Data Book
POLLUX

Data Book

MDPOLLUXDB
Ver. 0.91 / Jan 29, 2008
POLLUX
Data Book for POLLUX Application Processor

Document Release 0.91
29/1/2008

IMPORTANT NOTICE

Information in this document is provided solely to enable system and software implementers to use MagicEyes products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.
MagicEyes reserves the right to make changes without further notice to any products herein.
MagicEyes makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does MagicEyes assumes any liability arising out of the application or use of any product or circuit and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in MagicEyes data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. MagicEyes does not convey any license under its patent rights nor the rights of others. MagicEyes products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the MagicEyes product could create a situation where personal injury or death may occur. Should Buyer purchase or use MagicEyes products for any such unintended or unauthorized application, Buyer shall indemnify and hold MagicEyes and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, costs, damages and expenses and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that MagicEyes was negligent regarding the design or manufacture of the part.

HOW TO REACH US:
MagicEyes Digital Co., Ltd.
463-824 4F, Uniquest Bldg., 271-2 Seohyeon-Dong, Bundang-Gu Seongnam-City, Gyeonggi-Do, Korea
TEL: 82-31-788-0300
FAX:82-31-707-5765

HOME PAGE:
http://www.mesdigital.com
DEVELOPER FORUM:
http://forum.mesdigital.com

INDEX

1. INTRODUCTION ---- 2

1.1. Overview ---- **2**
1.1.1. Features ---- 2
1.1.2. Block Diagram ---- 3

1.2. Functional Specification ---- **3**
1.2.1. Architecture ---- 3
1.2.2. Clock and Power Manager ---- 3
1.2.3. GPIO Controller ---- 3
1.2.4. Memory Controller ---- 3
1.2.5. DMA (Direct Memory Access) ---- 4
1.2.6. Interrupt Controller ---- 4
1.2.7. Timer ---- 4
1.2.8. RTC (Real Time Clock) ---- 4
1.2.9. Audio ---- 4
1.2.10. SD/ MMC ---- 4
1.2.11. UART ---- 4
1.2.12. USB Host/ Device ---- 5
1.2.13. SSP/ SPI (Serial Protocol / Serial Peripheral Interface) ---- 5
1.2.14. I^2C (Inter-Integrate Circuit) ---- 5
1.2.15. PWM (Pulse Width Modulation) ---- 5
1.2.16. ADC (Analog to Digital Converter) ---- 6
1.2.17. MLC ---- 6
1.2.18. DPC (Display Controller) ---- 6
1.2.19. 3D Graphic Accelerator ---- 6
1.2.20. CSC (Color Space Converter) ---- 7
1.2.21. Operating Voltage Range ---- 7
1.2.22. Operating Frequency ---- 7
1.2.23. Operating Temperature ---- 7
1.2.24. Package ---- 7

2. I/O PIN DESCRIPTION ---- 9

2.1. Mechanical Dimensions ---- **9**

2.2. Marking Information ---- **9**

2.3. POLLUX Package FBGA Ball Map ---- **11**

2.4. Function Description ---- **11**

2.5. GPIO Pin Functions ---- **16**

2.6. POLLUX I/O Pin Listing ---- **19**

2.7. System Configuration ---- **24**

3. ALIVE ---- 27

3.1. Features ---- **27**

3.2. Block Diagram ---- **28**

3.3. Power Isolation. ---- **28**
3.3.1. Power Pad (VDD Groups) ---- 28
3.3.2. Power Gating ---- 29

3.4. Alive Registers ---- 29
3.4.1. Alive GPIO Register ---- 29
3.4.2. Scratch Register ---- 29

3.5. Momentary Power Control ---- 29
3.5.1. CoreVDD Powering on ---- 29
3.5.2. CoreVDD Powering off ---- 29

3.6. Register Summary ---- 30

4. CLOCK AND POWER MANAGEMENT (RESET) ---- 33

4.1. Clock Manager Overview ---- 33
4.1.1. Clock Manager Features ---- 33
4.1.2. Clock Manager Block Diagram ---- 33

4.2. Clock Manager Operation ---- 33
4.2.1. PLL (Phase Locked Loop) ---- 34
4.2.2. Clock Generator ---- 36

4.3. Power Manager Overview ---- 38

4.4. Power Down Mode Operation ---- 39
4.4.1. IDLE Mode ---- 40
4.4.2. STOP Mode ---- 40
4.4.3. GPIO as a Wake Up Source ---- 41

4.5. Reset Generation ---- 42
4.5.1. Power On Reset Sequence ---- 42
4.5.2. Software Reset and GPIO Reset ---- 43
4.5.3. Watchdog Reset ---- 43
4.5.4. nPORST, Software Reset, Watchdog Reset and GPIO Reset ---- 43

4.6. Change PLL Value ---- 44

4.7. Register Summary ---- 45

5. ID REGISTER (ECID) ---- 61

5.1. ECID Overview ---- 61

5.2. ECID timing ---- 61

5.3. Register Summary ---- 61

6. GPIO CONTROLLER ---- 65

6.1. Overview ---- 65
6.1.1. Block Diagram ---- 65

6.2. Operation ---- 66
6.2.1. Input Operation ---- 66
6.2.2. Output Operation ---- 66
6.2.3. Alternate Function Operation ---- 66

6.3. GPIO PIN FUNCTIONS ---- 67
6.3.1. GPIOx Control Register (x = A, B, C) ---- 70

7. MEMORY CONTROLLER ---- 76

7.1. Overview ---- 76
7.1.1. Unified Memory Architecture (UMA) ---- 76
7.1.2. Block Diagram ---- 77
7.1.3. Pin Function Description ---- 77

7.2. Operation ---- 78
7.2.1. MCU-Y Bank Feature ---- 79
7.2.2. MCU-S Bank Feature ---- 79
7.2.3. Boot Mode ---- 79
7.2.4. Memory Map ---- 80

7.3. Block Address ---- 81
7.3.1. Linear Array ---- 81
7.3.2. Display Array ---- 82

7.4. Memory Controller Timing ---- 82
7.4.1. DDR-SDRAM Timing ---- 83
7.4.2. Static Memory Timing ---- 84

7.5. Interface ---- 86

7.6. NAND Boot Overview ---- 87
7.6.1. NAND Boot Features ---- 88
7.6.2. NAND Boot Sequence ---- 88
7.6.3. Normal Access Sequence ---- 88
7.6.4. ECC (BCH) ---- 89

7.7. MCU-Y Memory Control Register ---- 89

7.8. MCU-S Memory Control Register ---- 91

8. DIRECT MEMORY ACCESS CONTROLLER (DMA) ---- 102

8.1. Overview ---- 102
8.1.1. Features ---- 102
8.1.2. Block Diagram ---- 102

8.2. Operation ---- 103
8.2.1. Memory to Memory ---- 103
8.2.2. I/O to Memory ---- 103
8.2.3. Memory to I/O ---- 103

8.3. Command Buffer Mode ---- 103
8.3.1. Write Through Mode (Original Mode) ---- 103
8.3.2. Write Buffered Mode ---- 104
8.3.3. DMA Controller Base Address ---- 105
8.3.4. DMA Priority ---- 105
8.3.5. Peripheral DMA Request ID ---- 105

8.4. Register Summary ---- 107

9. INTERRUPT CONTROLLER ---- 111

9.1. Overview ---- 111
9.1.1. Block Diagram ---- 111

9.2. Operation ---- 111
9.2.1. F-bit and I-bit of PSR (Program Status Register) ---- 111
9.2.2. Interrupt Mode ---- 111
9.2.3. Interrupt Mask Register ---- 111
9.2.4. Interrupt Source ---- 111

9.2.5. Interrupt Priority ---- 113
9.2.6. Interrupt Mode Register (INTMODE) ---- 115
9.2.7. Interrupt MASK Register (INTMASK) ---- 115
9.2.8. Interrupt Pending Register (INTPEND) ---- 115

9.3. Register Summary ---- 116

10. TIMER ---- 122

10.1. Overview ---- 122
10.1.1. Features ---- 122
10.1.2. Block Diagram ---- 122

10.2. Operation ---- 122

10.3. Timer Setting Sequence ---- 122

10.4. Register Summary ---- 124

11. REAL TIME CLOCK (RTC) ---- 127

11.1. Overview ---- 127
11.1.1. Features ---- 127
11.1.2. Block Diagram ---- 127

11.2. Operation ---- 127
11.2.1. Backup Battery Operation ---- 127
11.2.2. RTC Operation ---- 128
11.2.3. Accessing the RTC Time Counter Setting/Read Register ---- 128
11.2.4. Interrupt Pending Register ---- 128

11.3. Register Summary ---- 129

12. AUDIO CONTROLLER ---- 132

12.1. Overview ---- 132
12.1.1. Features ---- 132
12.1.2. Block Diagram ---- 132
12.1.3. Pin Function Description ---- 132

12.2. Operation ---- 133
12.2.1. I^2S Controller Operation ---- 133
12.2.2. Timing Information ---- 136

12.3. Valid Data Structure ---- 136
12.3.1. Audio Buffer ---- 136
12.3.2. Output Buffer ---- 136
12.3.3. PCM Input Buffer ---- 137
12.3.4. Buffer Status ---- 137

12.4. Operation Flow ---- 138
12.4.1. I^2S ---- 138

12.5. Register Summary ---- 139

13. SD/MMC CONTROLLER ---- 143

13.1. Overview ---- 143
13.1.1. Features ---- 143
13.1.2. Block Diagram ---- 143

13.1.3. Pin Function Description ---- 143

13.2. SD/MMC Card Information ---- 144
13.2.1. SD and MMC Card Pin Assignments ---- 144
13.2.2. SD and MMC Card Registers ---- 144

13.3. Operation ---- 145
13.3.1. Bus protocol ---- 145
13.3.2. Initialization ---- 146
13.3.3. Error Detection ---- 147
13.3.4. Clocks ---- 147
13.3.5. SDIO Operation ---- 147
13.3.6. Operation Sequence Example ---- 148
13.3.7. Signal Timing ---- 150

13.4. Register Summary ---- 151

14. UART ---- 159

14.1. Overview ---- 159
14.1.1. Features ---- 159
14.1.2. Block Diagram ---- 160

14.2. Operation ---- 160
14.2.1. Data Transmission ---- 161
14.2.2. Data Reception ---- 161
14.2.3. Auto Flow Control (AFC) ---- 161
14.2.4. Non-AFC ---- 161
14.2.5. Loop Back Mode ---- 162
14.2.6. Interrupt Generation ---- 162
14.2.7. DMA Request Generation ---- 162
14.2.8. UART Error Status FIFO ---- 162
14.2.9. Baud-Rate Generation ---- 163
14.2.10. UART Boot ---- 163
14.2.11. SIR(Serial Infra-Red) Mode ---- 164
14.2.12. Signal Description ---- 166

14.3. Register Summary ---- 167

15. USB HOST/ DEVICE ---- 173

15.1. USB HOST ---- 173
15.1.1. Overview ---- 173
15.1.2. USB Host Clock Generator ---- 173
15.1.3. USB Init Config ---- 174
15.1.4. USB HOST Control Registers Summary ---- 175

15.2. USB Device Rev 2.0 ---- 177
15.2.1. Overview ---- 177
15.2.2. Pin Function Description ---- 177
15.2.3. Detect VBUS ---- 178
15.2.4. USB Device Clock Generator ---- 178
15.2.5. Flowchart ---- 179

15.3. USB DEVICE 2.0 Register Summary ---- 181

16. SSP/SPI ---- 193

16.1. Overview ---- 193
16.1.1. Features ---- 193
16.1.2. Block Diagram ---- 193

16.1.3. Pin Function Description ---- 194

16.2. Operation ---- 194
16.2.1. SPI Mode ---- 194
16.2.2. SSP Mode ---- 196
16.2.3. SSPSPI Burst Receive mode ---- 196

16.3. Register Summary ---- 197

17. I2C CONTROLLER (I2C) ---- 200

17.1. Overview ---- 200
17.1.1. Features ---- 200
17.1.2. Block Diagram ---- 200
17.1.3. Pin Function Description ---- 200

17.2. Operation ---- 200
17.2.1. I^2C Bus Interface ---- 201
17.2.2. READ-WRITE OPERATION ---- 203
17.2.3. Example of the operation in Master mode ---- 204
17.2.4. Example of the operation in Slave mode ---- 205

17.3. Register Summary ---- 207

18. PULSE WIDTH MODULATOR (PWM) ---- 211

18.1. Overview ---- 211
18.1.1. Features ---- 211
18.1.2. Block Diagram (n=0, 1, 2) ---- 211
18.1.3. Pin Function Description ---- 212

18.2. Operation ---- 212
18.2.1. PWM Signal Timing ---- 212
18.2.2. Flowchart ---- 213

18.3. Register Summary ---- 214

19. ANALOG TO DIGITAL CONVERTER (ADC) ---- 217

19.1. Overview ---- 217
19.1.1. Features ---- 217
19.1.2. ADC Block Diagram ---- 217

19.2. Operation ---- 217
19.2.1. Signal Description ---- 218
19.2.2. Absolute Maximum Ratings ---- 218
19.2.3. Recommended Operating Conditions ---- 218
19.2.4. DC Electrical Characteristics ---- 218
19.2.5. AC Electrical Characteristics ---- 219
19.2.6. I/O Chart ---- 219
19.2.7. Timing ---- 220
19.2.8. Analog Input Selection Table ---- 220
19.2.9. Flowchart ---- 221

19.3. Register Summary ---- 222

20. MULTI LAYER CONTROLLER ---- 225

20.1. Overview ---- 225
20.1.1. Features ---- 225

20.1.2. Block Diagram ---- 226

20.2. Dual Register Set Architecture ---- 227

20.3. MLC Global parameters ---- 227
20.3.1. Screen size ---- 227
20.3.2. Priority ---- 228
20.3.3. Field Mode ---- 228
20.3.4. Background color ---- 228

20.4. Per-layer parameters ---- 229
20.4.1. Enable ---- 229
20.4.2. Lock control ---- 230
20.4.3. Position ---- 230
20.4.4. Pixel format ---- 231
20.4.5. Layer blending ---- 233
20.4.6. Address generation ---- 234

20.5. Video layer specific parameters ---- 235
20.5.1. Scale function ---- 235
20.5.2. Color control ---- 236

20.6. Clock Generation ---- 239

20.7. Register Sumarry ---- 240

21. DISPLAY CONTROLLER ---- 253

21.1. Overview ---- 253
21.1.1. Features ---- 253
21.1.2. Block Diagram ---- 254

21.2. Pin configuration ---- 255

21.3. Dual Display ---- 255

21.4. Sync Generator ---- 256
21.4.1. TFT/Video Encoder Block Diagram ---- 256
21.4.2. Clock generation ---- 257
21.4.3. Format ---- 259
21.4.4. Sync signals ---- 261
21.4.5. Scan mode ---- 268
21.4.6. Delay ---- 268
21.4.7. Interrupt ---- 268

21.5. Internal Video Encoder and DAC ---- 270
21.5.1. Features ---- 270
21.5.2. General description ---- 270
21.5.3. Video Encoder Reset Sequence ---- 270
21.5.4. Block diagram ---- 271
21.5.5. Timing interface ---- 272
21.5.6. Video Standard Selection ---- 272
21.5.7. Basic Video Adjustments ---- 273
21.5.8. Programmable Bandwidth ---- 273
21.5.9. Analog Video output configuration ---- 273

21.6. Register Sumarry ---- 274

22. 3D GRAPHICS ENGINE ---- 286

22.1. Overview ---- 286

22.1.1. Register 3D ---- 286
22.1.2. Command Processor ---- 287
22.1.3. Primitive Processor ---- 288
22.1.4. GTE/Clipper (Geometry Transformation Engine/ Triangle Clipper) ---- 288
22.1.5. TSE (Triangle Setup Engine) ---- 288
22.1.6. Rasterizer ---- 288

22.2. Memory structure for POLLUX ---- 288
22.2.1. Segments ---- 288
22.2.2. Blocks ---- 289
22.2.3. Sub-Segments ---- 289
22.2.4. Block addressing Mode ---- 290
22.2.5. Address converting table ---- 291
22.2.6. Resources using Non-linear addressing ---- 291

22.3. Command Processor ---- 291
22.3.1. Command buffer/queue ---- 292
22.3.2. Command Reader ---- 294
22.3.3. Command Parser ---- 295

22.4. TSE ---- 296
22.4.1. 24bit Floating Point Number ---- 297
22.4.2. TSE operation mode ---- 297
22.4.3. Culling direction and inverse culling flag ---- 297
22.4.4. TSE input vertex format ---- 298

22.5. GTE ---- 298
22.5.1. GTE input parameters ---- 299
22.5.2. Registers ---- 300
22.5.3. GTE Instruction ---- 301
22.5.4. Example ---- 309

22.6. Clipper ---- 309
22.6.1. Viewport ---- 311

22.7. Rasterizer ---- 311
22.7.1. Memory fill ---- 311
22.7.2. Pixel Pipeline ---- 312
22.7.3. Render Target ---- 312
22.7.4. Depth-Buffer ---- 313

22.8. Texture ---- 313
22.8.1. Page ---- 313
22.8.2. Mipmap ---- 314
22.8.3. Texture size ---- 314
22.8.4. Texture setting ---- 314
22.8.5. Palette ---- 315
22.8.6. Image texture ---- 316
22.8.7. Tile indexed texture ---- 317
22.8.8. Tile ---- 318
22.8.9. Texture cache ---- 320

22.9. Primitive Processor ---- 320
22.9.1. Overview ---- 320
22.9.2. Primitive mode ---- 320
22.9.3. Primitive control ---- 321
22.9.4. Vertex buffer ---- 323
22.9.5. Index buffer ---- 325
22.9.6. Matrix palette ---- 325

22.10. Display (Page flipping) ---- 326
22.10.1. Page flipping ---- 327

22.10.2. Synchronous page flipping ---- 327

22.11. 3D Core Registers ---- 327
22.11.1. General / Control register map (C001_A000h ~ C001_A1FFh) ---- 328

22.12. Register Summary ---- 331

23. COLOR SPACE CONVERTER (CSC) ---- 353

23.1. Overview ---- 353

23.2. Operation ---- 353
23.2.1. Input/ Output data ---- 353
23.2.2. Input from MPEG4 decoder or VIP ---- 354
23.2.3. Output to Texture ---- 354
23.2.4. Transparency Test ---- 354
23.2.5. Clock Generation ---- 354

23.3. CSC Registers ---- 355
23.3.1. Register map (C000_3200h ~ C000_3227h) ---- 355
23.3.2. Registers ---- 355

24. ELECTRICAL CHARACTERISTIC ---- 359

24.1. ABSOLUTE MAXIMUM RATINGS ---- 359

24.2. RECOMMENDED OPERATION CONDITIONS ---- 359

24.3. D.C. ELECTRICAL CHARACTERISTICS ---- 360
24.3.1. Normal I/O – 3.3v ---- 360
24.3.2. DDR Normal I/O ---- 361
24.3.3. DDR Differential I/O ---- 361
24.3.4. USB I/O ---- 361
24.3.5. Normal Operation ---- 362

Figure

Figure 1-1. POLLUX Block Diagram 3
Figure 4-1. Block Diagram of CLOCK MANAGER 33
Figure 4-2. Block Diagram of PLL 34
Figure 4-3. 533MHz CPU Clock 37
Figure 4-4. System BUS Clock 37
Figure 4-5. Power Management Sequence 39
Figure 4-6. Power Down Mode Sequence7 41
Figure 4-7. Wake Up Block Diagram 42
Figure 4-8. Power-On Reset Sequence 43
Figure 6-1. GPIO Block Diagram 65
Figure 7-1. Memory Controller Block Diagram 77
Figure 7-2. Memory Map 81
Figure 7-3. Linear Array Mode 82
Figure 7-4. Block Address Mode 82
Figure 7-5. DDR-SDRAM Write Timing 83
Figure 7-6. DDR-SDRAM Read Timing 83
Figure 7-7. DDR-SDRAM Initial Timing 83
Figure 7-8. No Wait Timing (CLK=BCLK(Typ.100MHz)) 84
Figure 7-9. Wait Timing (CLK=BCLK(Typ.100MHz)) 85
Figure 7-10. Burst 4 Timing (CLK=BCLK(Typ.100MHz)) 85
Figure 7-11. NAND Flash Write Timing (CLK=BCLK(Typ.100MHz)) 85
Figure 7-12. NAND Flash Read Flash Timing (CLK=BCLK(Typ.100MHz)) 86
Figure 7-13. 16bit Data bus width DDR-SDRAM Interface 87
Figure 7-14. 16bit Data bus width DDR-SDRAM Interface 87
Figure 7-15. NAND Boot 88
Figure 8-1. DMA Controller Block Diagram 102
Figure 9-1. Interrupt Process Diagram 111
Figure 9-2. Priority Generating Block 114
Figure 10-1. Timer Block Diagram 122
Figure 10-2. Timer Sequence 123
Figure 11-1. RTC Block Diagram 127
Figure 12-1. Audio Block Diagram 132
Figure 12-2. I^2S Mode 133
Figure 12-3. Left-justified Mode 133
Figure 12-4. Right-justified Mode 133
Figure 12-5. I^2S Clock Select(Master Mode) 134
Figure 12-6. I^2S Controller State 135
Figure 12-7. Buffer Enable & In/Out Enable 136
Figure 12-8. OUTPUT MEMORY FORMAT 137
Figure 12-9. INPUT MEMORY FORMAT 137
Figure 12-10. I^2S Operation Flow 138
Figure 13-1. SD/MMC Block Diagram 143
Figure 13-2. Operation Without Data 145
Figure 13-3. Stream Read Operation 146
Figure 13-4. Multiple Block Read Operation 146
Figure 13-5. Stream Write Operation 146
Figure 13-6. Multiple Block Write Operation 146
Figure 14-1. UART Block Diagram 160
Figure 14-2. Auto Flow Control Interface 161
Figure 14-3. UART Clock Generator 163
Figure 14-4. SIR Function Block Diagram 164
Figure 14-5. Serial I/O Frame Timing Diagram (Normal UART) 164
Figure 14-6. Intra-Red Transmit Mode Frame Timing Diagram 165
Figure 14-7. Infra-Red Receive Mode Frame Timing Diagram 165
Figure 15-1. USB Host Controller Block Diagram 173
Figure 15-2. USB 1.1 Host Clock Generator 173
Figure 15-3. Pin and Control Signals 174
Figure 15-4. USB Device Rev 2.0 Block Diagram 177
Figure 15-5. USB 2.0 Device Clock Generator 178
Figure 15-6. OUT DMA Operation Flow 179
Figure 15-7. IN DMA Operation Flow 180
Figure 16-1. SSP/SPI Block Diagram 193
Figure 16-2. Format A & Continuous Data Transmission Timing 195
Figure 16-3. Format B & Continuous Data Transmission Timing Chart 195
Figure 16-4. SSP mode Timing Chart 196
Figure 16-5. Continuous Data Transmission Timing Chart 196
Figure 16-6. Burst Receive Timing Chart 196
Figure 17-1. I^2C Block Diagram 200

Figure 17-2. Start and Stop Condition 201
Figure 17-3. I2C-Bus Interface Data Format 202
Figure 17-4. Data Transfer on the I2C–Bus 202
Figure 17-5. Acknowledge on the I2C-Bus 203
Figure 17-6. Master Transmitter Mode Data Format 204
Figure 17-7. Master Receiver Mode Data Format 204
Figure 17-8. Slave Transmitter Mode Data Format 205
Figure 17-9. Slave Receiver Mode Data Format 205
Figure 18-1. PWM Block Diagram 211
Figure 18-2. PWM Signal Timing 212
Figure 18-3. Example of a timing chart in which the PWMnDUTY value is bigger than the PWMnPERIOD value 212
Figure 18-4. PWM Flowchart 213
Figure 19-1. ADC Block Diagram 217
Figure 19-2. Main Waveform 220
Figure 19-3. ADC Sequence Flowchart 221
Figure 20-1. Concept of Multi Layer Controller 225
Figure 20-2. MLC Block Diagram 226
Figure 20-3. Dual Reigister Set Architecture 227
Figure 20-4. Layer priority 228
Figure 20-5. Lock timing 230
Figure 20-6. Layer position 230
Figure 20-7. Separated YUV format 232
Figure 20-8. Transparency & Color inversion 233
Figure 20-9. Alpha blending 234
Figure 20-10. Bilinear Filter & Nearest Neighbor 235
Figure 20-11. Brightness 236
Figure 20-12. Contrast 237
Figure 20-13. The basic concept of Hue and Saturation control 237
Figure 20-14. Hue and Saturation 238
Figure 20-15. Clock generation 239
Figure 21-1. Display Controller 254
Figure 21-2. Sync Generator 256
Figure 21-3. Peripheral Clock Generation 257
Figure 21-4. Video Clock Generation 257
Figure 21-5. Clock polarity and delay 258
Figure 21-6. RGB Dithering 260
Figure 21-7. Horizontal timing 261
Figure 21-8. Vertical timing 263
Figure 21-9. Vertical timing 264
Figure 21-10. STN Horizontal timing 265
Figure 21-11. STN Vertical timing 265
Figure 21-12. Frame Rate Control 266
Figure 21-13.. Random Number Generator 266
Figure 21-14. Data stream format with SAV/EAV 267
Figure 21-15. Y/C clip 268
Figure 21-16. Interrupt block diagram and timing 269
Figure 21-17. Video Encoder and DAC block diagram 271
Figure 21-18. Timing inteface 272
Figure 21-19. Luma Bandwidth 273
Figure 21-20. Chroma Bandwidth 273
Figure 22-1. 3D Core Control Flow 286
Figure 22-2. 3D Core Data Flow 286
Figure 22-3. Register 3D 287
Figure 22-4. The two kinds of control signal flow 287
Figure 22-5. GTE/ Clipper Data Flow 288
Figure 22-6. Segment 289
Figure 22-7. 32bit address described in segment and block 289
Figure 22-8. Sub-Segment 290
Figure 22-9. Texture Mode 291
Figure 22-10. Address Converting (Screen to Sub-Segment) 291
Figure 22-11. Command Processor 292
Figure 22-12. Command Processor 292
Figure 22-13. Command Reader Operation 295
Figure 22-14. Command Parser 295
Figure 22-15. 24bit Floating Point format 297
Figure 22-16. TSE Operation Mode 297
Figure 22-17. Backface Culling Direction 298
Figure 22-18. Inverse Culling Flag Logic 298
Figure 22-19. GTE Block 299
Figure 22-20. GTE Input Parameter 300

Figure 22-21. Render Triangle Strip 300
Figure 22-22. Example of GTE Programming Rules 302
Figure 22-23. Source and Destination in GTE Instruction 304
Figure 22-24. Modifier 305
Figure 22-25. Example of Arithmetic Instruction with Delay 306
Figure 22-26. Example of NOP 306
Figure 22-27. Example of Delay 307
Figure 22-28. Example of SETLOOP/ LOOP 307
Figure 22-29. Example of RENDER 308
Figure 22-30. Example of RENDERFORCE and EXIT 308
Figure 22-31. Source0 Component Modifier SETRCP 308
Figure 22-32. Pipeline Timing of Sample Transform Example 309
Figure 22-33. Clipper 311
Figure 22-34. View Port 311
Figure 22-35. Pixel Writing 312
Figure 22-36. Page Segment 313
Figure 22-37. MipMap 314
Figure 22-38. Texture Setting 315
Figure 22-39. Palette 316
Figure 22-40. Text Format 317
Figure 22-41. Pixel Texture 318
Figure 22-42. Tile 319
Figure 22-43. Primitive Processor 320
Figure 22-44. Primitive Type 322
Figure 22-45. Triangle Strip at GTE 323
Figure 22-46. Index Data Format 325
Figure 22-47. Page Flipping 327
Figure 23-1. Color Space Conversion 353
Figure 23-2. Input/ Output Data Addressing 353
Figure 23-3. YCbCr 420 Mode (from MPEG Decoder) 354
Figure 23-4. Clock generation 354

Table

Table 2-1. POLLUX Pin Function Description 16
Table 2-2. GPIO Pin Functions 18
Table 4-1. Initial PDIV/ MDIV/ SDIV Value 35
Table 4-2. PDIV/ MDIV/ SDIV Value for PLL0 35
Table 4-3. PDIV/ MDIV/ SDIV Value for PLL1 36
Table 4-4. POLLUX Clock Summary 36
Table 4-5. Wake Up Condition and Power Down Mode Status 40
Table 4-6. Reset State 44
Table 6-1. Pull-Up Resistor Current 66
Table 6-2. GPIO Pin Functions 69
Table 7-1. Memory Controller Pin Function Description 78
Table 7-2. Boot Mode 80
Table 7-3. Linear Array Mode 82
Table 7-4. Block Address Mode 82
Table 7-5. Symbol Description 84
Table 7-6. Symbol Description 86
Table 8-1. DMA Controller Base Address 105
Table 8-2. DMA Channel Priority 105
Table 8-3. Peripheral DMA Request ID 106
Table 9-1 Interrupt Sources Description Table 113
Table 12-1. I^2S Pin Function Description 133
Table 12-2. Sampling Rates by CLKDIV Values in 256 fs (Suppose PLL1 is 98304000) 135
Table 13-1. SD/MMC Pin Function Description 143
Table 13-2. SD/MMC Card Pin Assingment 144
Table 13-3. SD/MMC Card registers 144
Table 13-4. Command Format 145
Table 13-5. Normal Response Format 145
Table 13-6. Long Response Format 145
Table 14-1. Interrupts in connection with FIFO 162
Table 14-2. Error Interrupts Table 163
Table 14-3. Signal Description 166
Table 15-1. OHCI REGISTERs for USB Host Controller 176
Table 15-2 USB2.0 Pin Function Description 178
Table 16-1. SSP/SPI Pin Function Description 194
Table 17-1. I^2C Pin Function Description 200
Table 18-1. PWM Pin Function Description 212
Table 19-1. Signal Description 218
Table 19-2. Absolute Maximum Ratings 218
Table 19-3. Recommended Operating Conditions 218
Table 19-4. DC Electrical Characteristic 218
Table 19-5. AC Electrical Characteristics 219
Table 19-6. I/O Chart 219
Table 19-7. Analog Input Selection Table 220
Table 20-1. Dirty flag 227
Table 20-2. Top controller registers 227
Table 20-3. Background color format 228
Table 20-4. RGB layer registers 229
Table 20-5. RGB layer format 231
Table 20-6. Video layer format 232
Table 20-7. TPCOLOR & INVCOLOR format 233
Table 20-8. RGB layer address & flip 234
Table 20-9. Segment addressing format 234
Table 20-10. Video layer specific parameters 235
Table 20-11. CONTRAST parameter 236
Table 20-12. PCLK mode 239
Table 20-13. BCLK mode 239
Table 21-1. Pin Configuration 255
Table 21-2. Register Base Address 255
Table 21-3. PCLK mode 257
Table 21-4. Recommend Clock Settings 258
Table 21-5. Clock delay 258
Table 21-6. Data format for Primary Sync Generator 259
Table 21-7. Recommend Setting for RGB Dithering 260
Table 21-8. Output order for ITU-R BT.656 or 601 260
Table 21-9. Horizontal timing symbols 261
Table 21-10. Horizontal timing registers 262
Table 21-11. Vertial timing symbols 263
Table 21-12. Vertical timing registers 264

Table 21-13. sync timing symbols 264
Table 21-14. STN Horizontal timing symbols 265
Table 21-15. STN Horizontal timing registers 265
Table 21-16. STN Vertical timing registers 266
Table 21-17. Random number generator value 267
Table 21-18. Embedded Sync Code 267
Table 21-19. Registers relative to scan mode 268
Table 21-20. Default delay value 268
Table 21-21. Default timing settings 272
Table 21-22. Settings for Various Output Formats 272
Table 21-23. Summary of DAC Voltage and Codes 273
Table 22-1. 32bit Address Description 289
Table 22-2. Display Mode Segment 290
Table 22-3. Texture Mode Segment 290
Table 22-4. Resources using Non-Linear addressing 291
Table 22-5. Command Processor Mode 293
Table 22-6. Command Format 293
Table 22-7. Command ID 293
Table 22-8. Register Fill Command 293
Table 22-9. Optimized Command 294
Table 22-10. TSE Operation Mode 297
Table 22-11. TSE Input Vertex Format 298
Table 22-12. GRP3D_CONTROL Register 299
Table 22-13. Registers 301
Table 22-14. GTE Instruction 301
Table 22-15. Instruction Format 301
Table 22-16. Three Sub-Operations 301
Table 22-17. Instructions by PIPELine Timing 301
Table 22-18. Arithmetic Instruction 302
Table 22-19. Arithmetic Instruction Format 302
Table 22-20. Vertex Number 303
Table 22-21. Vertex Component 303
Table 22-22. Modifier Table 306
Table 22-23. Control Instructions 306
Table 22-24. Example of NOP (Not Operation) 306
Table 22-25. Example of Delay 307
Table 22-26. Example of EXIT 307
Table 22-27. Example of SETLOOP 307
Table 22-28. Example of LOOP 307
Table 22-29. Example of RENDER 307
Table 22-30. Example of RENDERFORCE 308
Table 22-31. Example 309
Table 22-32. Clipper 310
Table 22-33. Texture Size 314
Table 22-34. Texture Setting 315
Table 22-35. Tile 318
Table 22-36. 16bpp Address 319
Table 22-37. 8bpp Address 319
Table 22-38. 4bpp Address 319
Table 22-39. Texture Cache 320
Table 22-40. Primitive Mode 321
Table 22-41. Examples of primitive control parameters 322
Table 22-42. Page Flipping 327
Table 22-43. Synchronous Page Flipping 327
Table 22-44. 3D Core Registers 328
Table 23-1. PCLK mode 354
Table 23-2. BCLK mode 354
Table 23-3. Register MAP 355
Table 24-3. Normal 3.3v DC Currents of Output Driver 360

CHAPTER 1.

INTRODUCTION

1. Introduction

1.1. Overview

This document describes the POLLUX 32bit RISC Processor developed by MagicEyes Digital Co., Ltd. POLLUX is a complete product designed for Handheld, Low Cost and Low Power Consumption products.
POLLUX incorporates a 32bit CPU Processor, 3D Graphic Accelerator, USB 2.0 Device and a variety of I/O peripheral components. POLLUX can significantly reduce system costs by eliminating not only the system control CPU, but also the Graphic IC, as well as the USB Host 1.1/ USB 2.0 Device IC. POLLUX helps system designers reduce engineering effort and time taken to develop a new system, by adding only memory and I/O devices such as the LCD panel and human I/F devices.

1.1.1. Features

- 90nm, CMOS Process Technology.
- 288 pin FBGA (Fine BGA) Package
- 32bit CPU Embedded Architecture : 533MHz ARM926EJ with 16KByte I-Cache and 16KByte D-Cache.
- High performance 3D Graphics Accelerator
- USB 1.1 Host Controller, USB 2.0 Device, SIR, 4CH. UART
- LCD Controller, 8CH. DMA, Timer, Interrupt Controller, RTC
- SD/MMC
- I^2S
- I^2C, SSP, ADC, GPIOs, PWM, Power Manager

1.1.2. Block Diagram

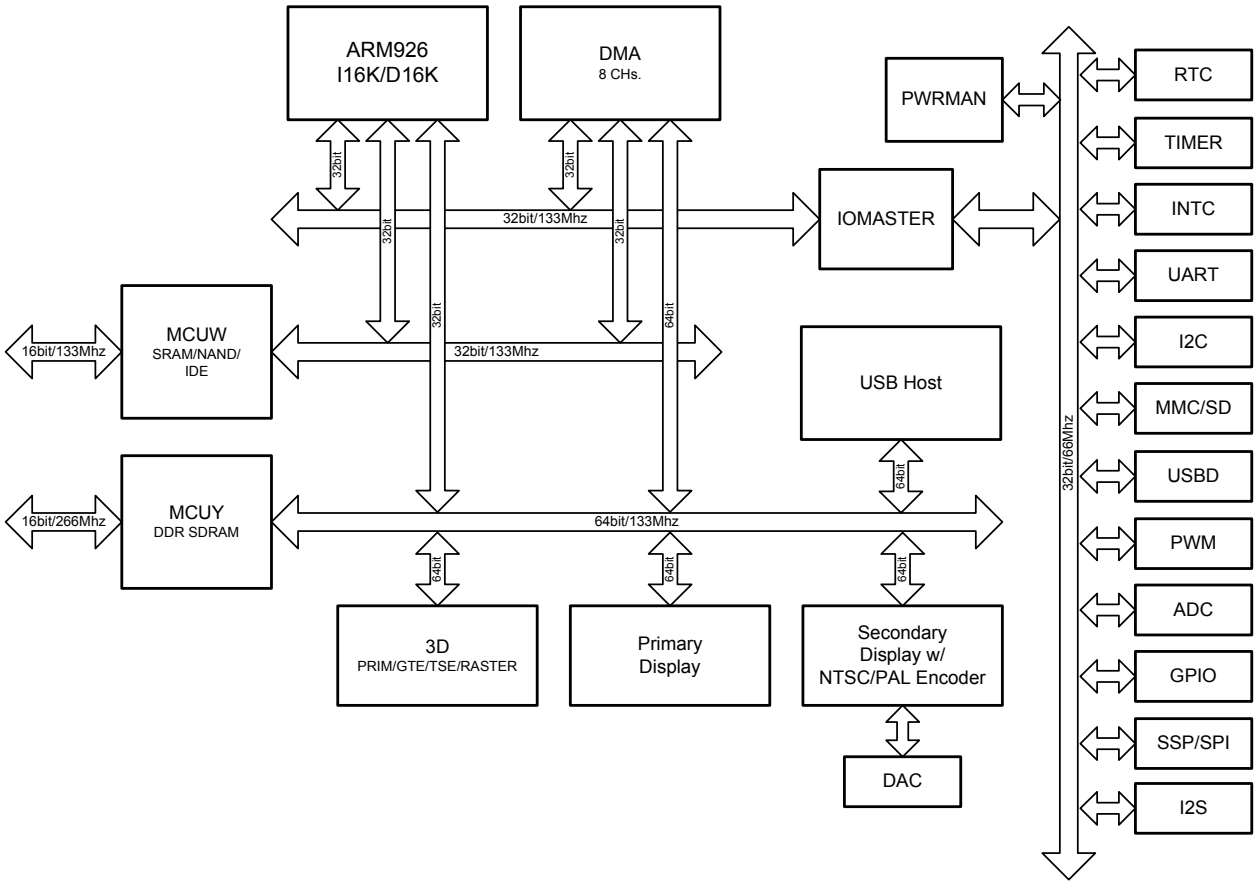


Figure 1-1. POLLUX Block Diagram

1.2. Functional Specification

1.2.1. Architecture

- 533MHz, ARM926EJ CPU Core
- 16KByte Instruction Cache, 16KByte Data Cache, Write Buffer to reduce the effect of main memory bandwidth and latency on performance.
- 16/32 bit RISC architecture and powerful instruction set with CPU core.

1.2.2. Clock and Power Manager

- Frequency is changed by Programmable Divider (PDIV, MDIV, SDIV)
- Number of PLLs are 2.

1.2.3. GPIO Controller

- Total 84 General Purpose I/O (GPIO) Pins
- Various GPIO Interrupt Mode (Rising Edge, Falling Edge, High Level and Low Level Detect)
- Individual Interrupt Generation Enabled/ disable is possible.

1.2.4. Memory Controller

- Memory Controller supports for single DDR-SDRAM Bank

- Static Memory Controller
  - Multiplexed Address
  - SRAM, ROM and NOR Flash
  - Programmable Wait Control
  - Burst Read/ Write Support
  - NAND Flash : NAND Booting and Hardware ECC Generation Support (both SLC/MLC)

1.2.5. DMA (Direct Memory Access)

- 8 Channel DMA
- Memory to Memory Transfer
- Memory to I/O Transfer
- I/O to Memory Transfer

1.2.6. Interrupt Controller

- Built-in Interrupt Controller
- Interrupt Controller Manages 64 Interrupt Sources
- Programmable Priority Control

1.2.7. Timer

- 32bit Built-in Timer/Counter
- WatchDog Function Supported

1.2.8. RTC (Real Time Clock)

- 32bit Counter
- Alarm Function : Alarm Interrupt or Wake Up from Power Down Mode
- Independent power pin (VDD_RTC)
- Support 1Hz Time interrupt for Power Down Mode

1.2.9. Audio

- I^2S
  - 16/ 18 bits
  - Master and Slave Mode
  - Master Mode : Up to 192 kHz (128, 192, 256, 384fs)
  - Slave Mode : Up to 192 kHz (128, 192, 256, 384fs)
  - I^2S, Left-Justified, Right-Justified Data Mode

1.2.10. SD/ MMC

- Secure Digital memory (SD mem Version 2.00)
- Secure Digital I/O (SDIO version 1.10)
- Multiedia Cards (MMC version 4.2)
- Support clock speed up to 52MHz
- Support PIO and DMA mode data transfer
- Support 1/4-bit data bus

1.2.11. UART

- UART0 ~ UART3 with DMA-Based or Interrupt-Based Operation.
- UART0 with Full modem function.

- Ability to add or delete standard asynchronous communications bit (Start, Stop and Parity) in the serial data.
- Independently controls transmission, reception, line status and data set interrupts.
- Programmable Baud Rate
- Modem control pins that allow flow control through software.
- Fully programmable serial interface :
  - 5, 6, 7, 8-bit characters
  - Even, Odd and no parity detection
  - 1 or 2 Stop bit generation
  - Baud rate generation up to around 3.1Mbps
- 16-byte transmit FIFO
- 16-byte receive FIFO
- Complete status reporting capability
- Ability to generate and detect line breaks
- Internal diagnostic capabilities that include
  - Loopback controls for communications link fault isolation
- Separates DMA requests for transmit and receive data services

1.2.12. USB Host/ Device

- USB Host 1.1
  - Compliant to USB 1.1 specification
  - 3 downstream ports.
- USB 2.0 Device
  - Compliant to USB2.0 specification.
  - Support FS/HS dual mode operation.
  - Conforms to UTMI (USB Transceiver Macrocell Interface)
  - Easy FIFO size configuration.
  - DMA interface capability.

1.2.13. SSP/ SPI (Serial Protocol / Serial Peripheral Interface)

- SSP Protocol compatible, SPI Protocol
- 32 x 16 FIFO
- Master& Slave mode
- Polling, Interrupt, DMA transfer mode
- Supports Standard four SPI Format
  - Format A, normal
  - Format A, inverse
  - Format B, normal
  - Format B, inverse
- 5-bit pre-scale counter
- 3 Channel SSP/SPI

1.2.14. I^2C (Inter-Integrate Circuit)

- 2 channel I^2C-bus
- 100Kbps ~ 1Mbps Speed (due to clock Pre-Scaler)
- Interrupt mode (Byte Transfer)
- Support Master & Slave mode

1.2.15. PWM (Pulse Width Modulation)

- 3 Channel Pulse Width Modulator channels

- 7-bit Clock divider & 10-bit Period counter
- 10-bit Duty counter

1.2.16. ADC (Analog to Digital Converter)

- Channels : 8 channels
- Resolution : 10 bit
- Maximum conversion rate : 500KSPS (samples per sec)
- Main clock : 2.5MHz (max.)
- Power supply : 3.3V ±0.3V, 1.3V ±0.1V(Digital I/O Interface)
- Input range : 0.0V ~ 3.3V ($3.3V_{P-P}$)
- Differential linearity Error : ±1.0 LSB
- Integral linearity Error : ±2.0 LSB
- Signal to Noise & Distortion Ratio : 54dB (Typ.)

1.2.17. MLC

- Two RGB Layers and One Video Layer
- RGB layers can be used as 3D layers.
- Dual Register-Set Architecture
- Various Pixel Formats
- Video Layer Priority
- Various Blending Effects between Layers
  - Per-layer or Per-pixel Alpha blending, Transparency, Inverse color
- Hardware Clipping
- Vertical flip
- Free Layer Position and Size in Pixel Unit
- Scale-up/Down (Video Layer Only)
  - Bi-linear interpolation or Nearest neighbor sampling scale-up/down
- Color control (Video layer only)
  - Brightness, Contrast, Hue, Saturation

1.2.18. DPC (Display Controller)

- Supports RGB, Multiplexed RGB (MRGB), ITU-R BT.601 and ITU-R BT.656
- Programmable HSYNC, VSYNC and DE (Data Enable)
- Programmable polarity for Sync signals
- Programmable delay for data, Sync signals and clock phase
- Supports dual display
- Supports NTSC/PAL TV output (CVBS only)
- Supports RGB dithering
- 10-bit Built-in Video DAC : Analog Composite Support
- Support Format : NTSC-J/N, PAL-B/D/G/H/I/M/N/Combination-N
- Support STN-LCD for Primary
- TV-out Up Scaler for Dual Display(LCD and TV)

1.2.19. 3D Graphic Accelerator

- Programmable floating point vector processing engine for geometric transform.
  - Variable mathematical operation.
  - 512-depth instruction memory
  - 256 Vector Input/Constant Registers
  - 16 Vector General Purpose Registers

- 2x2 Sub-Pixel Accuracy
- Features of Texturing
  - Perspective Correction
  - Multi-Texturing
  - Bi-Linear Filtering
  - MIPMAP
- Features of Pixel Operation
  - Per-Pixel Fogging
  - Hardware Dithering

1.2.20. CSC (Color Space Converter)

- Color Space Converter (CSC) module converts Y/Cb/Cr output from MPEG decoder or External Video Input Processor(VIP) to usable format for using as texture in 3D Core.

1.2.21. Operating Voltage Range

- Core : TBD
- Memory : 2.5V (DDRSDRAM)
- I/O : 3.3V

1.2.22. Operating Frequency

- 533MHz /133MHz/133MHz (CPU/Core/Memory)

1.2.23. Operating Temperature

- Commercial Temperature (0℃ ~ 70℃) : POLLUX

1.2.24. Package

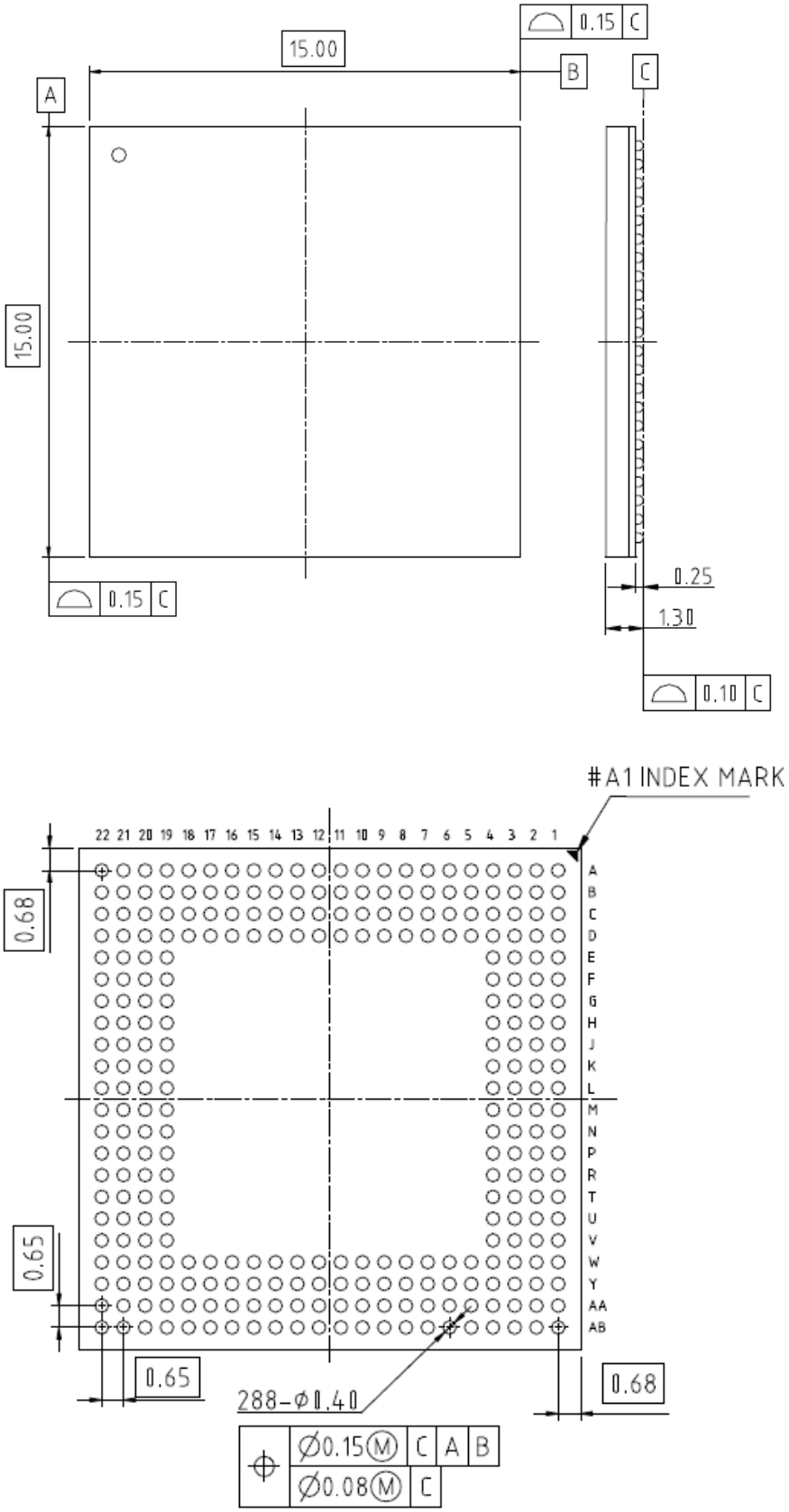
- 288-FBGA

CHAPTER 2.

I/O PIN DESCRIPTION

2. I/O PIN DESCRIPTION

2.1. Mechanical Dimensions



2.2. Marking Information

TBD

2.3. POLLUX Package FBGA Ball Map

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	
A	VDD10A_PLL0	VIDEO	VDD33A_DAC	USBXI	USBXO	VSS33A_USB2C	USBDP	USBDM	VSS33A_USB2A	ADCIN6	VDD33A_ADC	VSS33A_ADC	YA4	YA6	YA8	YA11	YDQM1	YCK	YCKE	YD7	YD5	YD4	A
B	VSS10A_PLL0	VDD10A_PLL1	VSS33A_DAC	VSSA33C_USB2	VDDA33C_USB2	VSS33A_USB2B	VDD33A_USB2B	VDD33A_USB2A	ADCIN4	ADCIN5	ADCIN7	ADCREF	YA5	YA7	YA9	YA12	YnCK	YDQM0	YDQS0	YD6	YD3	YD2	B
C	XTI	VSS10A_PLL1	VREF	IREF	USBTESTA	UP	UM	ADCIN2	ADCIN3	VDD33D_ADC	YD15	YD13	YD12	YD11	YD10	YD8	YnWE	YnCAS	YnRAS	YnCS0	YD1	YD0	C
D	XTO	GPIOALIVE1	VCOMP	VSS33AR_DAC	USBREXT	USBVBUS	ADCIN0	ADCIN1	VSS	VDDI10	YD14	YD9	YDQS1	Yref	VSS25	VDD25	VSS25	VDD25	VSS25	YBA0	YA10	YA0	D
E	GPIOALIVE3	GPIOALIVE2	GPIOALIVE0	VDD33AR_DAC															VDD25	YBA1	YA1	YA2	E
F	GPIOALIVE6	GPIOALIVE5	GPIOALIVE4	VDD33D_DAC															VDD25	SDDAT13	SDDAT12	YA3	F
G	XTIRTC	VDD_RTC	PORSEL	VDDI10_ALV															SDDAT11	SDDAT10	SDDAT03	SDDAT02	G
H	XTORTC	VDDPWRON	nBATF	VDD33_ALV															VSS	SDCMD1	SDDAT01	SDDAT00	H
J	TDO	TCK	nVDDPWRTOGGLE	VSS															VDDI10	SDCLK1	SDCMD0	SDCLK0	J
K	TMS	TDI	nTRST	VDDI10															VSS	SDA1	SCL1	SDA0	K
L	RX0	nPORST	JTAGMODE	VSS															VDD33_IO	SSPFRM0	SSPRXD0	SCL0	L
M	TX0	nEXTRST	TEST_EN	VDD33_IO															SSPFRM1	SSPCLK1	SSPTXD0	SSPCLK0	M
N	TX1	nCTS1	nRTS1	VSS															SSPRXD1	SSPTXD1	PVSYNC(YD)	PVCLK	N
P	RX1	nRIO1	nDCD1	VDDI10															VSS	PVD12	PDE(CL2)	PHSYNC(CL1)	P
R	TX2	RX2	TX3	nDSR1															VDD33_IO	PVD13	PVD1	PVD0	R
T	PWMOUT0	PWMOUT1	RX3	nDTR1															PVD14	PVD15	PVD3	PVD2	T
U	I2SDATO	I2SSYNC	PWMOUT2	VDD33_IO															PVD16	PVD17	PVD5	PVD4	U
V	I2SMCLK	I2SBCLK	I2SDATI	VSS															PVD19	PVD18	PVD7	PVD6	V
W	SA23	SA24	SA25	NC	VSS	VDDI10	VSS	nSCS4	VDD33_IO	VSS	VDDI10	VSS	VDD33_IO	VSS	RnB	VSS	VDD33_IO	VDDI10	VSS	PVD20	PVD9	PVD8	W
Y	SA21	SA22	nSCS9	NC	RDnWR	nSCS6	nSCS5	nSCS3	nSCS2	nSCS1	nSCS0	nSOE	nSWE	nNCS1	nNFOE	nNCS0	NFCLEi	NFALE	nNFWE	PVD23	PVD11	PVD10	Y
AA	SA20	nSCS8	LATADDR	nSWAIT	SA17	SA15	SA13	SA11	SA9	SA7	SA5	SA3	SA1	SD15	SD13	SD11	SD9	SD7	SD5	SD3	PVD22	PVD21	AA
AB	SA19	nSCS7	nSDQM1	SA18	SA16	SA14	SA12	SA10	SA8	SA6	SA4	SA2	SA0	SD14	SD12	SD10	SD8	SD6	SD4	SD2	SD1	SD0	AB
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	

2.4. Function Description

Pin Name	GPIO No	GPIO Function	Type	Description
Memory Controller (MCU-Y)				
YD[15:0]	-	-	I/O	MCU-Y Bank DDR-SDRAM Data bus. AD[15:0] is used for 16-bit data mode.
YA[12:0]	-	-	O	MCU-Y Bank DDR-SDRAM Address bus.
YBA[1:0]	-	-	O	MCU-Y Bank DDR-SDRAM Bank address.
YCK	-	-	O	MCU-Y Bank DDR-SDRAM clock.
YnCK	-	-	O	MCU-Y Bank DDR-SDRAM inverting clock.
YCKE	-	-	O	MCU-Y Bank DDR-SDRAM clock enable.
YnCS0	-	-	O	MCU-Y Bank DDR-SDRAM Chip Select 0. This signal should be connected to the chip select pin for DDR-SDRAM.
YnRAS	-	-	O	MCU-Y Bank DDR-SDRAM RAS. This signal should be connected to the row address strobe pin for all banks of DDR-SDRAM.

Pin Name	GPIO No	GPIO Function	Type	Description
YnCAS	-	-	O	MCU-Y Bank DDR-SDRAM CAS. This signal should be connected to the column address strobe pin for all banks of DDR-SDRAM.
YnWE	-	-	O	MCU-Y Bank DDR-SDRAM write enable. This signal should be connected to the write enables for DDR-SDRAM.
YDQS0	-	-	I/O	MCU-Y Bank DDR-SDRAM Data Strobe0.
YDQS1	-	-	I/O	MCU-Y Bank DDR-SDRAM Data Strobe1.
YDQM0	-	-	O	MCU-Y Bank DDR-SDRAM DQM for data bytes 0. These signals should be connected to the data output mask enable for DDR-SDRAM.
YDQM1	-	-	O	MCU-Y Bank DDR-SDRAM DQM for data bytes 1. These signals should be connected to the data output mask enable for DDR-SDRAM.
Yref			I	DDR PAD Reference Voltage (1.25v)
Static Memory				
SD[15:0]	-	-	I/O	Static Memory/NAND Data bus.
SA[0]/nSDQM0	-	-	O	Static Memory Address bus, This signal should be connected to the low byte enable for 16bit SRAM.
SA[1]	-	-	O	Static Memory Address bus.
SA[2](SALAT[19])	-	-	O	Static Memory Address bus.
SA[3](SALAT[20])	-	-	O	Static Memory Address bus.
SA[4](SALAT[21])	-	-	O	Static Memory Address bus.
SA[5](SALAT[22])	-	-	O	Static Memory Address bus.
SA[6](SALAT[23])	-	-	O	Static Memory Address bus.
SA[7](SALAT[24])	-	-	O	Static Memory Address bus.
SA[8](SALAT[25])	-	-	O	Static Memory Address bus.
SA[18:9]	-	-	O	Static Memory Address bus.
SA[25:19]	GPIOC[14:8]	ALT1	O	Static Memory Address bus.
nSDQM[1]	-	-	O	Static Memory High Byte Enable.
LATADDR	-	-	O	Static Memory Latch Address Enable.
RDnWR	-	-	O	Buffer Direction Control. Read/Write for static interface. Intended for use as a steering signal for buffering logic.
nSCS [0]_ nSCS [1]	-	-	O	Static Memory Chip Select.
nSCS [1]_ nSCS [0]	-	-	O	Static Memory Chip Select.
nSCS [2]	GPIOC[15]-	ALT1	O	Static Memory Chip Select.
nSCS [3]	GPIOC[16]	ALT1	O	Static Memory Chip Select.
nSCS [4]	GPIOC[17]	ALT1	O	Static Memory Chip Select.
nSCS [5]	GPIOC[18]	ALT1	O	Static Memory Chip Select.
nSCS [6]	GPIOC[19]	ALT1	O	Static Memory Chip Select.
nSCS [7]	GPIOC[0]	ALT1	O	Static Memory Chip Select.
nSCS[8]	GPIOC[1]	ALT1	O	Static Memory Chip Select.
nSCS[9]	GPIOC[2]	ALT1	O	Static Memory Chip Select.
nSWAIT	-	-	I	Wait Control for Static Memory. This signal is an input and is driven low by the Static Memory to extend the length of the transfers to/from applications processor.
nSOE	-	-	O	Static Memory Read Enable.
nSWE	-	-	O	Static Memory Write Enable.
nNCS[0]_ nNCS [1]	-	-	O	NAND Chip Select0. This is chip select to NAND Flash memory.

Pin Name	GPIO No	GPIO Function	Type	Description
nNCS[1]_ nNCS [0]	-	-	O	NAND Chip Select1. This is chip select to NAND Flash memory.
NFCLE	-	-	O	NAND CLE
NFALE	-	-	O	NAND ALE
RnB	-	-	I	NAND Ready & Busy. This is Ready/Busy signal of NAND Flash memory.
nNFOE	-	-	O	NAND Read Enable.
nNFWE	-	-	O	NAND Write Enable.
Dispay Controller				
PVD[7:0]	GPIOA[7:0]	ALT1	O	Video output data (RGB, Multiplexed-RGB, YCbCr)
PVD[23:8]	GPIOB[31:16]	ALT1	O	Video output data (RGB, Multiplexed-RGB, YCbCr)
PDE(CL2)	-	-	O	Data Enable'CSYNC.
PHSYNC(CL1)	-	-	O	Horizontal Sync
PVSYNC(YD)	-	-	O	Vertical Sync
PVCLK	-	-	I/O	Pixel Clock output/input.
NTSC/PAL				
VIDEO			O	Analog TV out.
VREF			I	Video DAC Reference
IREF			I	Video DAC Reference Current
VCOMP			I	Video Compensation Capacitor
UART				
RX0	-	-	I	UART0 Receive Pin./UART Boot Pin
TX0	GPIOA[8]	ALT1	O	UART0 Transmit Pin.
nRTS1	GPIOA[10]	ALT1	O	UART1 Ready-To-Send
nCTS1	GPIOA[9]	ALT1	I	UART1 Clear-To-Send
nDTR1	GPIOA[14]	ALT1	O	UART1 Data-Terminal-Ready
nDSR1	GPIOA[13]	ALT1	I	UART1 Data-Set-Ready Pin0.
nDCD1	GPIOA[12]	ALT1	I	UART1 Data-Carrier-Detect
nRIO1	GPIOA[11]	ALT1	I	UART1 Ring Indicator
RX1	GPIOA[16]	ALT1	I	UART1 Receive Pin.
TX1	GPIOA[15]	ALT1	O	UART1 Transmit Pin.
RX2	GPIOA[18]	ALT1	I	UART2 Receive Pin.
TX2	GPIOA[17]	ALT1	O	UART2 Transmit Pin.
RX3	GPIOA[20]	ALT1	I	UART3 Receive Pin
TX3	GPIOA[19]	ALT1	O	UART3 Transmit Pin.
MMC/SD Controller				
SDCLK0	GPIOB[0]	ALT1	IO	SD Clock 1 Channel
SDCMD0	GPIOB[1]	ALT1	I/O	SD Command 1 Channel
SDDATA0[3:0]	GPIOB[5:2]	ALT1	I/O	SD Data 1 Channel
SDCLK1	GPIOB[6]	ALT1	IO	SD Clock 2 Channel
SDCMD1	GPIOB[7]	ALT1	I/O	SD Command 2 Channel
SDDATA1[3:0]	GPIOB[11:8]	ALT1	I/O	SD Data 2 Channel

Pin Name	GPIO No	GPIO Function	Type	Description
I²S Controller				
I2SDATO	GPIOA[21]	ALT1	O	Audio Port Data Out (SDO)
I2SBCLK	GPIOA[22]	ALT1	I/O	Audio Port Bit Clock
I2SDATI	GPIOA[23]	ALT1	I	Audio Port Data In (SDI)
I2SSYNC	GPIOA[24]	ALT1	I/O	Audio Port Sync Signal (LRCK)
I2SMCLK	GPIOA[25]	ALT1	O	Audio Port MCLK (12 MHz Clock)
I²C Controller				
SDA0	GPIOA[26]	ALT1	I/O	I2C SDA0
SCL0	GPIOA[27]	ALT1	I/O	I2C SCL0
SDA1	GPIOA[28]	ALT1	I/O	I2C SDA1
SCL1	GPIOA[29]	ALT1	I/O	I2C SCL1
SSP / SPI Controller				
SSPRXD0	GPIOB[14]	ALT1	I	Synchronous Serial Port RX (SPI MISO)
SSPTXD0	GPIOB[15]	ALT1	O	Synchronous Serial Port TX (SPI MOSI)
SSPCLK0	GPIOB[13]	ALT1	I/O	Synchronous Serial Port clock (SPI SCLK)
SSPFRM0	GPIOB[12]	ALT1	I/O	Synchronous Serial Port Frame0 signal
SSPRXD1	GPIOC[5]	ALT1	I	Synchronous Serial Port RX 1l
SSPTXD1	GPIOC[6]	ALT1	O	Synchronous Serial Port TX 1
SSPCLK1	GPIOC[4]	ALT1	I/O	Synchronous Serial Port clock 1
SSPFRM1	GPIOC[3]	ALT1	I/O	Synchronous Serial Port Frame1 signal
SSPRXD2	GPIOB[1]	ALT2	I	Synchronous Serial Port RX 2
SSPTXD2	GPIOB[2]	ALT2	O	Synchronous Serial Port TX 2
SSPCLK2	GPIOB[0]	ALT2	I/O	Synchronous Serial Port clock 2
SSPFRM2	GPIOB[5]	ALT2	I/O	Synchronous Serial Port Frame 2 signal
PWM Controller				
PWM0	GPIOA[30]	ALT1	O	Pulse Width Modulation Output
PWM1	GPIOA[31]	ALT1	O	Pulse Width Modulation Output
PWM2	GPIOC[7]	ALT1	O	Pulse Width Modulation Output
USB				
USBDP	-	-	I/O	USB Host Only : Ch0
USBDM	-	-	I/O	USB Host Only : Ch0
UP	-	-	I/O	USB Device 2.0 Plus.
UM	-	-	I/O	USB Device 2.0 Minus.
USBREXT	-	-	O	Connection to the external 3.4k-ohm (+/- 1%) resistor. The 3.4K-ohm (+/- 1%) resistor must be referenced to the ground and placed as close as possible to the chip.
USBTESTA	-	-	O	Analog Test Pin. 10K-ohm (+/- 1%) resistor must be referenced to the GND.
USBVBUS	-	-	I	VBUS Detect. This VBUS indicator signal indicates that the VBUS signal on the USB cable is active.
USBXO	-	-	O	12MHz Crystal output.
USBXI	-	-	I	12MHz Crystal input.
ADC Controller				

Pin Name	GPIO No	GPIO Function	Type	Description
ADCIN[7:0]	-	-	I	ADC Analog Input
ADCREF	-	-	I	ADC Regerence Voltage.
Miscellaneous				
nPORST	-	-	I	Global Reset In. Active low input. nRESET is a level-sensitive input which is used to start the processor from a known address. A LOW level will cause the current instruction to terminate abnormally and all on-chip states to be reset. When nRESET is driven HIGH, the processor will re-start from address 0.
nEXTRST	-	-	O	Global Reset Output.
PORSEL	-	-	I	Decide to use internal POR (Power On Reset) 0 : Use Internal POR. 1 : Not use internal POR.
TEST_EN	-	-	I	Test Mode Enable.
VDDPWRON	-	-	O	Momentary Power Control (VDD Power On)
VDDPWRTOGGLE	-	-	I	Momentary Power Control (Toggle Switch)
JTAGMODE	-	-	I	Serial Data Input for JTAG Enable
PLL				
XTI	-	-	I	27MHz Oscillator Input
XTO	-	-	O	27MHz Oscillator Output
RTC				
XTIRTC	-	-	I	32.768KHz RTC Crystal input.
XTORTC	-	-	O	32.768KHz RTC Crystal output.
VDD_RTC	-	-	I	RTC Supply
Alive GPIO				
GPIOALIVE[6:0]	-	-	O	Active GPIO when Powered Off
Power Manager				
nBATF	-	-	I	Battery Fault
JTAG				
nTRST	-	-	I	Reset input for the JTAG Logic.
TDO	-	-	O	Serial output for test instructions and data for JTAG.
TDI	-	-	I	Serial Data input for JTAG.(TESTMODE2)
TMS	-	-	I	TMS controls the sequence of the TAP controller's state.(TESTMODE1)
TCK			I	Clock Input for JTAG Logic. (TESTMODE0)
Power and Ground : TBD				
VSS10A_PLL0	-	-	-	Analog GND for PLL TBD Power.
VDD10A_PLL0	-	-	-	Analog VCC for PLL TBD Power.
VSS10A_PLL1	-	-	-	Analog GND for PLL TBD Power.
VDD10A_PLL1	-	-	-	Analog VCC for PLL TBD Power.
VSS33A_USB2A	-	-	-	Analog GND for USB 3.3V Power.
VSS33A_USB2B	-	-	-	Analog GND for USB 3.3V Power.
VSS33A_USB2C	-	-	-	Analog GND for USB 3.3V Power.
VDD33A_USB2A	-	-	-	Analog VCC for USB 3.3V Power.
VDD33A_USB2B	-	-	-	Analog VCC for USB 3.3V Power.
VSSA33C_USB2	-	-	-	Digital GND for USB Clock 3.3V Power.

Pin Name	GPIO No	GPIO Function	Type	Description
VDDA33C_USB2	-	-	-	Digital VCC for USB Clock 3.3V Power.
VSS33A_ADC	-	-	-	Analog GND for ADC 3.3V Power.
VDD33A_ADC	-	-	-	Analog VCC for ADC 3.3V Power.
VDD33D_ADC	-	-	-	Digital VCC for ADC 3.3V Power.
VSS33A_DAC	-	-	-	Analog GND for DAC 3.3V Power.
VSS33AR_DAC	-	-	-	Analog GND for DAC 3.3V Power.
VDD33A_DAC	-	-	-	Analog VCC for DAC 3.3V Power.
VDD33D_DAC	-	-	-	Digital VCC for DAC 3.3V Power.
VSS25	-	-	-	GND for DDR-SDRAM 2.5V Power.
VDD25	-	-	-	VCC for DDR-SDRAM 2.5V Power.
VDD_RTC	-	-	-	VCC for RTC 2.5V ~ 3.6V Power.
VDDI10	-	-	-	VCC for Internal TBD Logic Power.
AREF	-	-		1.25V Reference VCC for MCU-Y bank.
IREF	-	-	-	External resistor connection (2.6kΩ against Ground) for generating reference current.
VREF	-	-	-	TBD Reference VCC for Internal Video DAC.
VCOMP	-	-	-	External capacitor connection to 3.3V Power.
ADCREF	-	-	-	3.3V Reference VCC for ADC.
VDD33	-	-	-	VCC for 3.3V I/O Power.
VSS	-	-	-	GND.

Table 2-1. POLLUX Pin Function Description

2.5. GPIO Pin Functions

GPIO Pins	Alternate Function 1	Alternate Function 2
GPIOA[31 : 0]		
GPIOA[31]	PWMOUT[1]	-
GPIOA[30]	PWMOUT[0]	
GPIOA[29]	SDA[1]	
GPIOA[28]	SCL[1]	
GPIOA[27]	SDA[0]	
GPIOA[26]	SCL[0]	
GPIOA[25]	I2SMCLK	
GPIOA[24]	I2SSYNC	
GPIOA[23]	I2SDATI	
GPIOA[22]	I2SBCLK	
GPIOA[21]	I2SDATO	
GPIOA[20]	RX[3]	
GPIOA[19]	TX[3]	
GPIOA[18]	RX[2]	
GPIOA[17]	TX[2]	
GPIOA[16]	RX[1]	

GPIO Pins	Alternate Function 1	Alternate Function 2
GPIOA[15]	TX[1]	
GPIOA[14]	NDTR[1]	
GPIOA[13]	NDSR[1]	
GPIOA[12]	NDCD[1]	
GPIOA[11]	NRIO[1]	
GPIOA[10]	NRTS[1]	
GPIOA[9]	NCTS[1]	
GPIOA[8]	TX[0]	
GPIOA[7]	PVD[7]	
GPIOA[6]	PVD[6]	
GPIOA[5]	PVD[5]	
GPIOA[4]	PVD[4]	
GPIOA[3]	PVD[3]	
GPIOA[2]	PVD[2]	
GPIOA[1]	PVD[1]	
GPIOA[0]	PVD[0]	
GPIOB[31 : 0]		
GPIOB[31]	PVD[23]	-
GPIOB[30]	PVD[22]	
GPIOB[29]	PVD[21]	
GPIOB[28]	PVD[20]	
GPIOB[27]	PVD[19]	
GPIOB[26]	PVD[18]	
GPIOB[25]	PVD[17]	
GPIOB[24]	PVD[16]	
GPIOB[23]	PVD[15]	
GPIOB[22]	PVD[14]	
GPIOB[21]	PVD[13]	
GPIOB[20]	PVD[12]	
GPIOB[19]	PVD[11]	
GPIOB[18]	PVD[10]	
GPIOB[17]	PVD[9]	
GPIOB[16]	PVD[8]	
GPIOB[15]	SSPTXD[0]	
GPIOB[14]	SSPRXD[0]	
GPIOB[13]	SSPCLK[0]	
GPIOB[12]	SSPFRM[0]	
GPIOB[11]	SDDAT1[3]	
GPIOB[10]	SDDAT1[2]	
GPIOB[9]	SDDAT1[1]	

GPIO Pins	Alternate Function 1	Alternate Function 2
GPIOB[8]	SDDAT1[0]	
GPIOB[7]	SDCMD1	
GPIOB[6]	SDCLK1	
GPIOB[5]	SDDAT0[3]	SSPFRM[2]
GPIOB[4]	SDDAT0[2]	
GPIOB[3]	SDDAT0[1]	
GPIOB[2]	SDDAT0[0]	SSPTXD[2]
GPIOB[1]	SDCMD0	SSPRXD[2]
GPIOB[0]	SDCLK0	SSPCLK[2]
GPIOC[22 : 0]		
GPIOC[19]	nSCS[6]	
GPIOC[18]	nSCS[5]	
GPIOC[17]	nSCS[4]	
GPIOC[16]	nSCS[3]	
GPIOC[15]	nSCS[2]	
GPIOC[14]	SA[25]	
GPIOC[13]	SA[24]	
GPIOC[12]	SA[23]	
GPIOC[11]	SA[22]	
GPIOC[10]	SA[21]	
GPIOC[9]	SA[20]	
GPIOC[8]	SA[19]	
GPIOC[7]	PWMOUT[2]	
GPIOC[6]	SSPTXD[1]	
GPIOC[5]	SSPRXD[1]	
GPIOC[4]	SSPCLK[1]	
GPIOC[3]	SSPFRM[1]	
GPIOC[2]	nSCS[9]	AVCLK
GPIOC[1]	nSCS[8]	
GPIOC[0]	nSCS[7]	

Table 2-2. GPIO Pin Functions

2.6. POLLUX I/O Pin Listing

- Power Balls will be added.

Ball	Name	Type	Func.After Reset	Primary Function	Alternate Function 1	Alternate Function 2
E3	GPIOALIVE[0]		GPIOALIVE[0]	GPIOALIVE[0]	-	-
D2	GPIOALIVE [1]		GPIOALIVE [1]	GPIOALIVE [1]	-	-
E2	GPIOALIVE [2]		GPIOALIVE [2]	GPIOALIVE [2]	-	-
E1	GPIOALIVE [3]		GPIOALIVE [3]	GPIOALIVE [3]	-	-
F3	GPIOALIVE [4]		GPIOALIVE [4]	GPIOALIVE [4]		
F2	GPIOALIVE [5]		GPIOALIVE [5]	GPIOALIVE [5]		
F1	GPIOALIVE [6]		GPIOALIVE [6]	GPIOALIVE [6]		
L2	nPORST		nPORST	nPORST	-	-
M2	nEXTRST		nEXTRST	nEXTRST	-	-
G1	XTIRTC		XTIRTC	XTIRTC	-	-
H1	XTORTC		XTORTC	XTORTC	-	-
G2	VDD_RTC		RTC_VDD	RTC_VDD	-	-
H3	nBATF		nBATF	nBATF	-	-
L3	JTAGMODE		JTAGMODE	JTAGMODE	-	-
H2	VDDPWRON		VDDPWRON	VDDPWRON	-	-
J3	VDDPWRTOGGLE		VDDPWRTOGGLE	VDDPWRTOGGLE	-	-
M3	TEST_EN		TEST_EN	TEST_EN	-	-
R22	GPIOA[0]/PVD[0]		GPIOA[0]	GPIOA[0]	PVD[0]	-
R21	GPIOA[1]/PVD[1]		GPIOA[1]	GPIOA[1]	PVD[1]	-
T22	GPIOA[2]/PVD[2]		GPIOA[2]	GPIOA[2]	PVD[2]	-
T21	GPIOA[3]/PVD[3]		GPIOA[3]	GPIOA[3]	PVD[3]	-
U22	GPIOA[4]/PVD[4]		GPIOA[4]	GPIOA[4]	PVD[4]	-
U21	GPIOA[5]/PVD[5]		GPIOA[5]	GPIOA[5]	PVD[5]	-
V22	GPIOA[6]/PVD[6]		GPIOA[6]	GPIOA[6]	PVD[6]	-
V21	GPIOA[7]/PVD[7]		GPIOA[7]	GPIOA[7]	PVD[7]	-
N22	PVCLK		PVCLK	PVCLK	-	-
A2	VIDEO		VIDEO	VIDEO	-	-
C3	VREF		VREF	VREF	-	-
C4	IREF		IREF	IREF	-	-
D3	VOUTCOMP		VOUTCOMP	VOUTCOMP	-	-
P21	PDE(CL2)		PDE(CL2)	PDE(CL2)	-	-
P22	PHSYNC(CL1)		PHSYNC(CL1)	PHSYNC(CL1)	-	-
N21	PVSYNC(YD)		PVSYNC(YD)	PVSYNC(YD)	-	-
L1	RX[0]		RX[0]	RX[0]		
M1	GPIOA[8]/TX[0]		GPIOA[8]	GPIOA[8]		
N2	GPIOA[9]/nCTS[1]		GPIOA[9]	GPIOA[9]	nCTS[1]	
N3	GPIOA[10]/nRTS[1]		GPIOA[10]	GPIOA[10]	nRTS[1]	
P2	GPIOA[11]nRIO[1]		GPIOA[11]	GPIOA[11]	nRIO[1]	

Ball	Name	Type	Func.After Reset	Primary Function	Alternate Function 1	Alternate Function 2
P3	GPIOA[12]/nDCD[1]		GPIOA[12]	GPIOA[12]	nDCD[1]	
R4	GPIOA[13]/nDSR[1]		GPIOA[13]	GPIOA[13]	nDSR[1]	
T4	GPIOA[14]/nDTR[1]		GPIOA[14]	GPIOA[14]	nDTR[1]	
N1	GPIOA[15]/TX[1]		GPIOA[15]	GPIOA[15]	TX[1]	
P1	GPIOA[16]/RX[1]		GPIOA[16]	GPIOA[16]	RX[1]	
R1	GPIOA[17]/TX[2]		GPIOA[17]	GPIOA[17]	TX[2]	
R2	GPIOA[18]/RX[2]		GPIOA[18]	GPIOA[18]	RX[2]	
R3	GPIOA[19]/TX[3]		GPIOA[19]	GPIOA[19]	TX[3]	
T3	GPIOA[20]/RX[3]		GPIOA[20]	GPIOA[20]	RX[3]	
AB22	SD[0]		SD[0]	SD[0]		
AB21	SD[1]		SD[1]	SD[1]		
AB20	SD[2]		SD[2]	SD[2]		
AA20	SD[3]		SD[3]	SD[3]		
AB19	SD[4]		SD[4]	SD[4]		
AA19	SD[5]		SD[5]	SD[5]		
AB18	SD[6]		SD[6]	SD[6]		
AA18	SD[7]		SD[7]	SD[7]		
AB17	SD[8]		SD[8]	SD[8]		
AA17	SD[9]		SD[9]	SD[9]		
AB16	SD[10]		SD[10]	SD[10]		
AA16	SD[11]		SD[11]	SD[11]		
AB15	SD[12]		SD[12]	SD[12]		
AA15	SD[13]		SD[13]	SD[13]		
AB14	SD[14]		SD[14]	SD[14]		
AA14	SD[15]		SD[15]	SD[15]		
AB13	SA[0]/nSDQM0		SA[0]	SA[0]		
AA13	SA[1]		SA[1]	SA[1]		
AB12	SA[2]/SA[19]		SA[2]	SA[2]		
AA12	SA[3]/SA[20]		SA[3]	SA[3]		
AB11	SA[4]/SA[21]		SA[4]	SA[4]		
AA11	SA[5]/SA[22]		SA[5]	SA[5]		
AB10	SA[6]/SA[23]		SA[6]	SA[6]		
AA10	SA[7]/SA[24]		SA[7]	SA[7]		
AB9	SA[8]/SA[25]		SA[8]	SA[8]		
AA9	SA[9]		SA[9]	SA[9]		
AB8	SA[10]		SA[10]	SA[10]		
AA8	SA[11]		SA[11]	SA[11]		
AB7	SA[12]		SA[12]	SA[12]		
AA7	SA[13]		SA[13]	SA[13]		
AB6	SA[14]		SA[14]	SA[14]		

Ball	Name	Type	Func.After Reset	Primary Function	Alternate Function 1	Alternate Function 2
AA6	SA[15]		SA[15]	SA[15]		
AB5	SA[16]		SA[16]	SA[16]		
AA5	SA[17]		SA[17]	SA[17]		
AB4	SA[18]		SA[18]	SA[18]		
AA3	LATADDR		LATADDR	LATADDR		
AA4	nSWAIT		nSWAIT	nSWAIT		
Y12	nSOE		nSOE	nSOE		
Y13	nSWE		nSWE	nSWE		
Y15	nNFOE		nNFOE	nNFOE		
Y19	nNFWE		nNFWE	nNFWE		
Y17	NFCLE		NFCLE	NFCLE		
Y18	NFALE		NFALE	NFALE		
W15	RnB		RnB	RnB		
Y5	RDnWR		RDnWR	RDnWR		
AB3	nSDQM1		nSDQM1	nSDQM1		
Y16	nNCS[0]_nNCS[1]		nNCS[0]	nNCS[0]		
Y14	nNCS[1]_nNCS[0]		nNCS[1]	nNCS[1]		
Y11	nSCS[0]_nSCS[1]		nSCS[0]	nSCS[0]		
Y10	nSCS[1]_nSCS[0]		nSCS[1]	nSCS[1]		
Y7	GPIOC[18]/nSCS[5]		GPIOC[18]	GPIOC[18]	nSCS[5]	
Y6	GPIOC[19]/nSCS[6]		GPIOC[19]	GPIOC[19]	nSCS[6]	
U1	GPIOA[21]/ I2SDATO		GPIOA[21]	GPIOA[21]	I2SDATO	
V2	GPIOA[22]/I2SBCLK		GPIOA[22]	GPIOA[22]	I2SBCLK	
V3	GPIOA[23]/ I2SDATI		GPIOA[23]	GPIOA[23]	I2SDATI	
U2	GPIOA[24]/I2SSYNC		GPIOA[24]	GPIOA[24]	I2SSYNC	
V1	GPIOA[25]/I2SMCLK		GPIOA[25]	GPIOA[25]	I2SMCLK	
L22	GPIOA[26]/SCL[0]		GPIOA[26]	GPIOA[26]	SCL[0]	
K22	GPIOA[27]/SDA[0]		GPIOA[27]	GPIOA[27]	SDA[0]	
K21	GPIOA[28]/SCL[1]		GPIOA[28]	GPIOA[28]	SCL[1]	
K20	GPIOA[29]/SDA[1]		GPIOA[29]	GPIOA[29]	SDA[1]	
T1	GPIOA[30]/PWMOUT[0]		GPIOA[30]	GPIOA[30]	PWMOUT[0]	
T2	GPIOA[31]/PWMOUT[1]		GPIOA[31]	GPIOA[31]	PWMOUT[1]	
J22	GPIOB[0]/SDCLK0/SSPCLK[2]		GPIOB[0]	GPIOB[0]	SDCLK0	SSPCLK[2]
J21	GPIOB[1]/SDCMD[0]/SSPRXD[2]		GPIOB[1]	GPIOB[1]	SDCMD0	SSPRXD2
H22	GPIOB[2]/SDDAT0[0]/SSPTXD[2]		GPIOB[2]	GPIOB[2]	SDDAT0[0]	SSPTXD2
H21	GPIOB[3]/SDDAT0[1]		GPIOB[3]	GPIOB[3]	SDDAT0[1]	
G22	GPIOB[4]/SDDAT0[2]/		GPIOB[4]	GPIOB[4]	SDDAT0[2]	
G21	GPIOB[5]/SDDAT0[3]/SSPFRM[2]		GPIOB[5]	GPIOB[5]	SDDAT0[3]	SSPRRM[2]
J20	GPIOB[6]/SDCLK1		GPIOB[6]	GPIOB[6]	SDCLK1	
H20	GPIOB[7]/SDCMD1		GPIOB[7]	GPIOB[7]	SDCMD1	

Ball	Name	Type	Func.After Reset	Primary Function	Alternate Function 1	Alternate Function 2
G20	GPIOB[8]/SDDAT1[0]		GPIOB[8]	GPIOB[8]	SDDAT1[0]	
G19	GPIOB[9]/SDDAT1[1]		GPIOB[9]	GPIOB[9]	SDDAT1[1]	
F21	GPIOB[10]/SDDAT1[2]		GPIOB[10]	GPIOB[10]	SDDAT1[2]	
F20	GPIOB[11]/SDDAT1[3]		GPIOB[11]	GPIOB[11]	SDDAT1[3]	
D22	YA[0]		YA[0]	YA[0]		
E21	YA[1]		YA[1]	YA[1]		
E22	YA[2]		YA[2]	YA[2]		
F22	YA[3]		YA[3]	YA[3]		
A13	YA[4]		YA[4]	YA[4]		
B13	YA[5]		YA[5]	YA[5]		
A14	YA[6]		YA[6]	YA[6]		
B14	YA[7]		YA[7]	YA[7]		
A15	YA[8]		YA[8]	YA[8]		
B15	YA[9]		YA[9]	YA[9]		
D21	YA[10]		YA[10]	YA[10]		
A16	YA[11]		YA[11]	YA[11]		
B16	YA[12]		YA[12]	YA[12]		
D20	YBA0		YBA0	YBA0		
E20	YBA1		YBA1	YBA1		
C22	YD[0]		YD[0]	YD[0]		
C21	YD[1]		YD[1]	YD[1]		
B22	YD[2]		YD[2]	YD[2]		
B21	YD[3]		YD[3]	YD[3]		
A22	YD[4]		YD[4]	YD[4]		
A21	YD[5]		YD[5]	YD[5]		
B20	YD[6]		YD[6]	YD[6]		
A20	YD[7]		YD[7]	YD[7]		
C16	YD[8]		YD[8]	YD[8]		
D12	YD[9]		YD[9]	YD[9]		
C15	YD[10]		YD[10]	YD[10]		
C14	YD[11]		YD[11]	YD[11]		
C13	YD[12]		YD[12]	YD[12]		
C12	YD[13]		YD[13]	YD[13]		
D11	YD[14]		YD[14]	YD[14]		
C11	YD[15]		YD[15]	YD[15]		
C20	YnCS0		YnCS0	YnCS0		
B19	YDQS0		YDQS0	YDQS0		
D13	YDQS1		YDQS1	YDQS1		
B18	YDQM0		YDQM0	YDQM0		
A17	YDQM1		YDQM1	YDQM1		

Ball	Name	Type	Func.After Reset	Primary Function	Alternate Function 1	Alternate Function 2
A18	YCK		YCK	YCK		
B17	YnCK		YnCK	YnCK		
A19	YCKE		YCKE	YCKE		
C17	YnWE		YnWE	YnWE		
C19	YnRAS		YnRAS	YnRAS		
C18	YnCAS		YnCAS	YnCAS		
D14	Yref		Yref	Yref		
L20	GPIOB[12]/SSPFRM[0]		GPIOB[12]	GPIOB[12]	SSPFRM[0]	
M22	GPIOB[13]/SSPCLK[0]		GPIOB[13]	GPIOB[13]	SSPCLK[0]	
L21	GPIOB[14]/SSPRXD[0]		GPIOB[14]	GPIOB[14]	SSPRXD[0]	
M21	GPIOB[15]/SSPTXD[0]		GPIOB[15]	GPIOB[15]	SSPTXD[0]	
D7	ADCIN[0]		ADCIN[0]	ADCIN[0]		
D8	ADCIN[1]		ADCIN[1]	ADCIN[1]		
C8	ADCIN[2]		ADCIN[2]	ADCIN[2]		
C9	ADCIN[3]		ADCIN[3]	ADCIN[3]		
B9	ADCIN[4]		ADCIN[4]	ADCIN[4]		
B10	ADCIN[5]		ADCIN[5]	ADCIN[5]		
A10	ADCIN[6]		ADCIN[6]	ADCIN[6]		
B11	ADCIN[7]		ADCIN[7]	ADCIN[7]		
B12	ADCREF		ADCREF	ADCREF		
A7	USBDP		USBDP	USBDP		
A8	USBDM		USBDM	USBDM		
C6	UP		UP	UP		
C7	UM		UM	UM		
D5	USBREXT		USBREXT	USBREXT		
A5	USBXO		USBXO	USBXO		
A4	USBXI		USBXI	USBXI		
C5	USBTESTA		USBTESTA	USBTESTA		
D6	USBVUS		USBVUS	USBVBUS		
C1	XTI		XTI	XTI		
D1	XTO		XTO	XTO		
J2	TCK		TCK	TCK		
K3	nTRST		nTRST	nTRST		
J1	TDO		TDO	TDO		
K1	TMS		TMS	TMS		
K2	TDI		TDI	TDI		
W22	GPIOB[16]/PVD[8]		GPIOB[16]	GPIOB[16]	PVD[8]	
W21	GPIOB[17]/PVD[9]		GPIOB[17]	GPIOB[17]	PVD[9]	
Y22	GPIOB[18]/PVD[10]		GPIOB[18]	GPIOB[18]	PVD[10]	
Y21	GPIOB[19]/PVD[11]		GPIOB[19]	GPIOB[19]	PVD[11]	

Ball	Name	Type	Func.After Reset	Primary Function	Alternate Function 1	Alternate Function 2
P20	GPIOB[20]/PVD[12]		GPIOB[20]	GPIOB[20]	PVD[12]	
R20	GPIOB[21]/PVD[13]		GPIOB[21]	GPIOB[21]	PVD[13]	
T19	GPIOB[22]/PVD[14]		GPIOB[22]	GPIOB[22]	PVD[14]	
T20	GPIOB[23]/PVD[15]		GPIOB[23]	GPIOB[23]	PVD[15]	
U19	GPIOB[24]/PVD[16]		GPIOB[24]	GPIOB[24]	PVD[16]	
U20	GPIOB[25]/PVD[17]		GPIOB[25]	GPIOB[25]	PVD[17]	
V20	GPIOB[26]/PVD[18]		GPIOB[26]	GPIOB[26]	PVD[18]	
V19	GPIOB[27]/PVD[19]		GPIOB[27]	GPIOB[27]	PVD[19]	
W20	GPIOB[28]/PVD[20]		GPIOB[28]	GPIOB[28]	PVD[20]	
AA22	GPIOB[29]/PVD[21]		GPIOB[29]	GPIOB[29]	PVD[21]	
AA21	GPIOB[30]/PVD[22]		GPIOB[30]	GPIOB[30]	PVD[22]	
Y20	GPIOB[31]/PVD[23]		GPIOB[31]	GPIOB[31]	PVD[23]	
AB2	GPIOC[0]/nSCS[7]		GPIOC[0]	GPIOC[0]	nSCS[7]	
AA2	GPIOC[1]/nSCS[8]		GPIOC[1]	GPIOC[1]	nSCS[8]	
Y3	GPIOC[2]/nSCS[9]/AVCLK		GPIOC[2]	GPIOC[2]	nSCS[9]	AVCLK
M19	GPIOC[3]/SSPFRM[1]		GPIOC[3]	GPIOC[3]	SSPFRM[1]	
M20	GPIOC[4]/SSPCLK[1]		GPIOC[4]	GPIOC[4]	SSPCLK[1]	
N19	GPIOC[5]/SSPRXD[1]		GPIOC[5]	GPIOC[5]	SSPRXD[1]	
N20	GPIOC[6]/SSPTXD[1]		GPIOC[6]	GPIOC[6]	SSPTXD[1]	
U3	GPIOC[7]/PWMOUT[2]		GPIOC[7]	GPIOC[7]	PWMOUT[2]	
AB1	GPIOC[8]/SA[19]		GPIOC[8]	GPIOC[8]	SA[19]	
AA1	GPIOC[9]/SA[20]		GPIOC[9]	GPIOC[9]	SA[20]	
Y1	GPIOC[10]/SA[21]		GPIOC[10]	GPIOC[21]	SA[21]	
Y2	GPIOC[11]/SA[22]		GPIOC[11]	GPIOC[11]	SA[22]	
W1	GPIOC[12]/SA[23]		GPIOC[12]	GPIOC[12]	SA[23]	
W2	GPIOC[13]/SA[24]		GPIOC[13]	GPIOC[13]	SA[24]	
W3	GPIOC[14]/SA[25]		GPIOC[14]	GPIOC[14]	SA[25]	
Y9	GPIOC[15]/nSCS[2]		GPIOC[15]	GPIOC[15]	nSCS[2]	
Y8	GPIOC[16]/nSCS[3]		GPIOC[16]	GPIOC[16]	nSCS[3]	
W8	GPIOC[17]/nSCS[4]		GPIOC[17]	GPIOC[17]	nSCS[4]	

2.7. System Configuration

Pin Name	Ball	Function Name	Description
GPIOA[2]/PVD[2]		CfgSDRBUSBW0	**MCU-Y bank data bus width** **00 : None** **01 : None** **10 : 16bit** **11 : None**
GPIOA[3]/PVD[3]		CfgSDRBUSBW1	
GPIOA[4]/PVD[4]		CfgSDRBW0	**SDRAM IC data bus width** **00 : None** **01 : 8bit** **10 : 16bit** **11 : None**
GPIOA[5]/PVD[5]		CfgSDRBW1	
GPIOA[6]/PVD[6]		CfgSDRCAP0	**SDRAM IC Capacity**

Pin Name	Ball	Function Name	Description
GPIOA[7]/PVD[7]		CfgSDRCAP1	00 : 64Mbit 01 : 128Mbit 10 : 256Mbit 11 : 512Mbit
SD[0]		CfgSDRLAT0	SDRAM CAS Latency 00 : 1CL 01 : 2CL 10 : 2.5CL 11: 3CL
SD[1]		CfgSDRLAT1	
SD[3]		CfgSDRREADLAT0	SDRAM Read Latency 00 : 1CL 01 : 2CL 10 : 2.5CL 11 : 3CL
SD[4]		CfgSDRREADLAT1	
SD[5]		CfgBOOTMODE0	Boot Mode 00 : UART 01 : NAND 10 : MEM 11 : None
SD[6]		CfgBOOTMODE1	
SD[7]		CfgSELCS	Select the funcion of Chip Select PAD 0 : NCS[0]_NCS[1] = NCS[0], NNCS[0]_NNCS[1] = NNCS[0], NCS[1]_NCS[0]= NCS[1] NNCS[1]_NNCS[0]= NNCS[1] 1 : NCS[1]_NCS[0] = NCS[0], NNCS[1]_NNCS[0] = NNCS[0], NCS[0]_NCS[1]= NCS[1] NNCS[0]_NNCS[1]= NNCS[1]
SD[8]		CfgNFType0	NAND Flash Address Type 00 : 3Address(Small Block) 01 : 4Address(Small Block) 10 : 4Address(Large Block) 11 : 5Address(Large Block)
SD[9]		CfgNFType1	
SD[10]		CfgBOOTSIZE	UART Boot Size 0 : 512Byte 1 : 16KByte
SD[11]		CfgSTBUSWidth	Static Bus Width 0 : 8bit 1 : 16bit
SD[12]		CfgSTLATADD	Static Latched Address 0 : None 1: Latched
SD[13]		CfgShadow	Shadow mode 0 : Shadow Disable (Static First) 1 : Shadow Enable (DRAM First)
SD[14]		CfgMSBAddr	Static MSB address 0 : GPIOC[14:8] 1 : sa[25:19]

-

PAD	CfgMSBAddr: 0 CfgSELCS : 0	CfgMSBAddr: 1 CfgSELCS : 0	CfgMSBAddr: 0 CfgSELCS : 1	CfgMSBAddr: 1 CfgSELCS : 1
SA[25:19]	GPIOC[14:8]	SA[25:19]	GPIOC[14:8]	SA[25:19]
nCS[0]_nCS[1] nNCS[0]_nNCS[1]	nCS[0] nNCS[0]	nCS[0] nNCS[0]	nCS[1] nNCS[1]	nCS[1] nNCS[1]
nCS[1]_nCS[0] nNCS[1]_nNCS[0]	nCS[1] nNCS[1]	nCS[1] nNCS[1]	nCS[0] nNCS[0]	nCS[0] nNCS[0]

CHAPTER 3.

ALIVE

3. ALIVE

In the status with eliminating core power supply of POLLUX, some PAD need power supply continuously and should keep driving PAD with certain value.
For example, The bit that controlling STN LCD should keep driving PAD with low in the status with eliminating core power supply.
32bit value can be saved in Scratch Register and the saved value maintains in the case of core power off.
User could on/off the system power by pressing toggle switch and ALIVE performs the necessary functions in these Momemtary power control.

3.1. Features

- AliveGPIO PAD uses independent power AliveVDD3.3V and AliveVDD1.0V.
- AliveGPIO PAD is all output PAD.
- The value of ALIVE Block maintains even in power off of Core Power.
- Alive GPIO does not use clock. To change the value, need to program set/reset pin of SR-flipflop directly.
- Scan chain is not inserted to Alive GPIO. When Mass Production, Alive GPIO would be tested with separated testmode in order to increase fault coverage.

3.2. Block Diagram

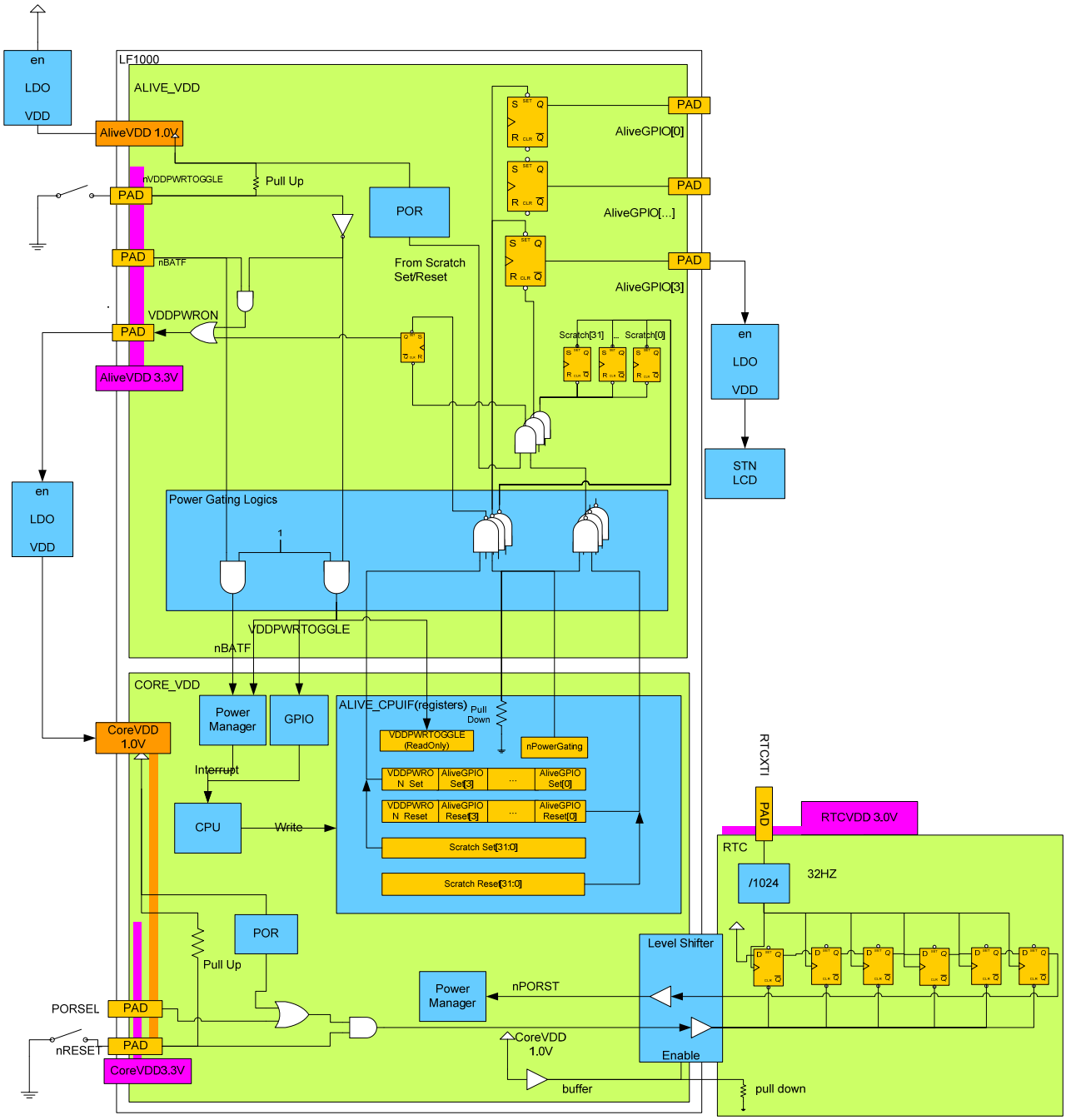


Figure 3-1. Block Diagram of Alive

3.3. Power Isolation.

3.3.1. Power Pad (VDD Groups)

The image above shows the interface between ALIVE Block and Core. The overall POWER is divided into four power like the foloowing.

- AliveVDD 1.0V
- AliveVDD 3.3V
- CoreVDD 1.0V
- CoreVDD 3.3V

In the case of power off of CoreVDD, AliveGPIO maintains its value since Alive VDD keep supplied. Pull-down register connected to nPowerGating performs the function of maintaining control bit of AliveGPIO securely in the interval of core power off or unstable.

3.3.2. Power Gating

The value of set/reset should be kept as low to maintains the value of AliveGPIO securely in the case of power off of CoreVDD. Therefore, it's designed to maintain low value in the case of core power off since nPowerGAting register is connected to pull-down register.

3.4. Alive Registers

3.4.1. Alive GPIO Register

AliveGPIO Register maintains the value in the case of power off of CoreVDD. Also, Control bit of STN LCD can be driven using AliveGPIO. Since Alive block does not operate based on clock, need to drive set/reset bit of flipflop directly when writing Alive GPIO. When doing this, you must not set/reset bit as '1' all together. You must set or reset (just one) as '1'. For example, the two BIT must be set like the following when wanting to change Alive GPIO 0 bit into '1'.

- GPIORST0=0, GPIOSET0=1

When CPU writes value in AliveGPIO, AliveGPIO internal register could acknowledge the value only in the condition of setting nPowerGating as '1' and writes. After CPU writing value in AliveGPIO, need to set nPowerGating bit '0' for the situation of power off of CoreVDD.

3.4.2. Scratch Register

Programmer could save any 32bit value in Scratch Register. The value of Scratch Register maintains in the case of power off of CoreVDD.

3.5. Momentary Power Control

3.5.1. CoreVDD Powering on

Under the system power off situation, let's assume that user press VDDPWRTOGGLE switch. While user pressing VDDPWRTOGGLE button, system starts to boot with LDO connecting to CoreVDD is powered on. After the system boot and user stop pressing the power toggle switch, CoreVDD need to be set to maintain 'power on' status by setting the value of VDDPWRON register as high.

3.5.2. CoreVDD Powering off

Let's assume that user press VDDPWRTOGGLE switch while system is operating. The event that user press VDDPWRTOGGLE button makes the interrupt occurred in CPU. CPU needs to make LDO connecting to CoreVDD powered off by setting the value of VDDPWRON register as low. At this time, the value of nPowrGating Register must be set as low to preserve the value of AliveGPIO and Scratch Register securely.

In the case of the current PowerMode is Stop Mode, the event that pressing VDDPWRTOGGLE button makes PowerManager waked up. After waking up, CPU takes care of the event that pressing VDDPWRTOGGLE button as interrupt.

3.6. Register Summary

Bit	R/W	Symbol	Description	Reset Value
ALIVE POWER GATING REGISTER (ALIVEPWRGATEREG) ***Address : C001_9000h***				
[31:1]	-	**RESERVED**	Reserved	0
[0]	R/W	**NPOWERGATING**	nPowerGating (negative active Power Gating). The default value is 0, disabling writing to AliveGPIO PADs, in order to keep the values of AliveGPIO PADs when Core Power 1.0V is off. 0 : Disable writing data to Alive GPIO 1 : Enable wrting data to Alive GPIO	1'b0
Alive GPIO Reset Register (ALIVEGPIORSTREG) ***Address : C001_9004h***				
[31:8]	-	**RESERVED**	Reserved	0
[7]	R/W	**VDDPWRONRST**	Reset VDDPWRON register. 0: none, 1: reset	1'b0
[6]	R/W	**GPIORST6**	Reset AliveGPIO6 register. 0: none, 1: reset	1'b0
[5]	R/W	**GPIORST5**	Reset AliveGPIO5 register. 0: none, 1: reset	1'b0
[4]	R/W	**GPIORST4**	Reset AliveGPIO4 register. 0: none, 1: reset	1'b0
[3]	R/W	**GPIORST3**	Reset AliveGPIO3 register. 0: none, 1: reset	1'b0
[2]	R/W	**GPIORST2**	Reset AliveGPIO2 register. 0: none, 1: reset	1'b0
[1]	R/W	**GPIORST1**	Reset AliveGPIO1 register. 0: none, 1: reset	1'b0
[0]	R/W	**GPIORST0**	Reset AliveGPIO0 register. 0: none, 1: reset	1'b0
Alive GPIO Set Register (ALIVEGPIOSETREG) ***Address : C001_9008h***				
[31:8]	-	**RESERVED**	Reserved	0
[7]	R/W	**VDDPWRONSET**	Set VDDPWRON register. Do not set 1 both **VDDPWRONRST** and **VDDPWRONSET** bit. Two bits must be set exclusively. Beause these bits are driving set/reset pins of SR-flipflop. 0: none, 1: set	1'b0
[6]	R/W	**GPIOSET6**	Set AliveGPIO6 register. Do not set 1 both **GPIORST6** and **GPIOSET6** bit. Two bits must be set exclusively. Beause these bits are driving set/reset pins of SR-flipflop. 0: none, 1: set	1'b0
[5]	R/W	**GPIOSET5**	Set AliveGPIO5 register. Do not set 1 both **GPIORST5** and **GPIOSET5** bit. Two bits must be set exclusively. Beause these bits are driving set/reset pins of SR-flipflop. 0: none, 1: set	1'b0
[4]	R/W	**GPIOSET4**	Set AliveGPIO4 register. Do not set 1 both **GPIORST4** and **GPIOSET4** bit. Two bits must be set exclusively. Beause these bits are driving set/reset pins of SR-flipflop. 0: none, 1: set	1'b0
[3]	R/W	**GPIOSET3**	Set AliveGPIO3 register. Do not set 1 both **GPIORST3** and **GPIOSET3** bit. Two bits must be set exclusively. Beause these bits are driving set/reset pins of SR-flipflop. 0: none, 1: set	1'b0
[2]	R/W	**GPIOSET2**	Set AliveGPIO2 register. Do not set 1 both **GPIORST2** and **GPIOSET2** bit. Two bits must be set	1'b0

Bit	R/W	Symbol	Description	Reset Value
			exclusively. Beause these bits are driving set/reset pins of SR-flipflop. 0: none, 1: set	
[1]	R/W	**GPIOSET1**	Set AliveGPIO1 register. Do not set 1 both **GPIORST1** and **GPIOSET1** bit. Two bits must be set exclusively. Beause these bits are driving set/reset pins of SR-flipflop. 0: none, 1: set	1'b0
[0]	R/W	**GPIOSET0**	Set AliveGPIO0 register. Do not set 1 both **GPIORST0** and **GPIOSET0** bit. Two bits must be set exclusively. Beause these bits are driving set/reset pins of SR-flipflop. 0: none, 1: set	1'b0
ALIVE GPIO Read Register (ALIVEGPIOREADREG) ***Address : C001_900Ch***				
[31:9]	-	**RESERVED**	Reserved	0
[8]	R	**VDDPWRTOGGLE**	Read VDDPWRTOGGLE PAD status 0: user does not pushed VDDPWRTOGGLE PAD, 1: user pushed VDDPWRTOGGLE PAD	1'b0
[7]	R	**VDDPWRONSET**	Read VDDPWRON register.	1'b0
[6]	R	**GPIOREAD6**	Read AliveGPIO6 register.	1'b0
[5]	R	**GPIOREAD5**	Read AliveGPIO5 register.	1'b0
[4]	R	**GPIOREAD4**	Read AliveGPIO4 register.	1'b0
[3]	R	**GPIOREAD3**	Read AliveGPIO3 register.	1'b0
[2]	R	**GPIOREAD2**	Read AliveGPIO2 register.	1'b0
[1]	R	**GPIOREAD1**	Read AliveGPIO1 register.	1'b0
[0]	R	**GPIOREAD0**	Read AliveGPIO0 register.	1'b0

ALIVE SCRATCH RESET REGISTER (ALIVESCRATCHRSTREG) ***Address : C001_9010h***				
[31 : 0]	R/W	**ALIVESCRATCHRST**	Reset Alive Scratch Register Each bit of **ALIVESCRATCHRST** drives Scratch Register's reset pin.	32'b0
ALIVE SCRATCH SET REGISTER (ALIVESCRATCHRSTREG) ***Address : C001_9014h***				
[31 : 0]	R/W	**ALIVESCRATCHSET**	Set Alive Scratch Register Each bit of **ALIVESCRATCHSET** drives Scratch Register's set pin	32'b0
ALIVE SCRATCH READ REGISTER (ALIVESCRATCHRSTREG) ***Address : C001_9018h***				
[31 : 0]	R	**ALIVESCRATCHREAD**	Read Alive Scratch Register	32'b0

CHAPTER 4.

CLOCK AND POWER MANAGEMENT (RESET)

4. CLOCK AND POWER MANAGEMENT (RESET)

4.1. Clock Manager Overview

The clock of the POLLUX is roughly divided into FCLK, HCLK, BCLK, PCLK. FCLK, HCLK and PCLK are used for the ARM CPU core, AHB bus peripherals and APB bus peripherals, respectively. In addition, BCLK is the clock for the POLLUX system bus and the clock for DDR-SDRAM memory. The 2-PLL of the POLLUX is called PLL0 and PLL1, respectively. The 2-PLL and EXTCLK are used to generate the above clocks (i.e. FCLK, HCLK, PCLK, BCLK). All PLLs are designed to operate with an X-TAL input of 27 MHz.

4.1.1. Clock Manager Features

- Embedded 2-PLL operating independently
- Output Frequency Range
  - PLL0: 16M ~ 533MHz
  - PLL1: 16M ~ 300MHz
- Frequency is changed by Programmable Divider(PDIV, MDIV, SDIV)
- Clock generation for all blocks in the chip
- The PLLs, except for PLL0, can be switched into Power Down mode by using the program.
- 32.768 KHz supported for Power Management
- Various Power Down Modes
  - IDLE mode and STOP mode
  - Various Wake Up sources

4.1.2. Clock Manager Block Diagram

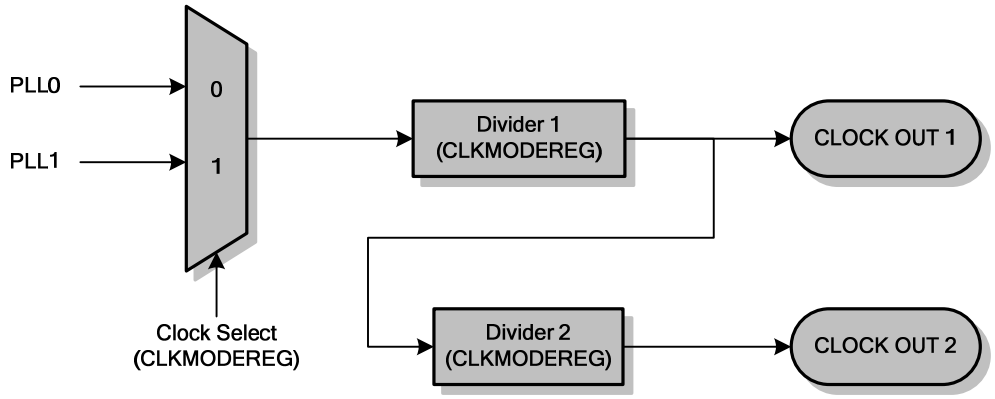


Figure 4-1. Block Diagram of CLOCK MANAGER

The above figure shows a diagram for the clock manager in the POLLUX. As shown in the above figure, the POLLUX has two PLLs: PLL0 and PLL1. The POLLUX receives the output of PLL0 and PLL1 and generates all system clocks, the memory clock and the CPU clock with the output frequency selected among two PLLs. (CLOCK OUT 1 and CLOCK OUT 2 in Figure 4-1 are notated arbitrarily. For more detailed information, refer to Chapter 4.)

4.2. Clock Manager Operation

4.2.1. PLL (Phase Locked Loop)

4.2.1.1. PMS Values

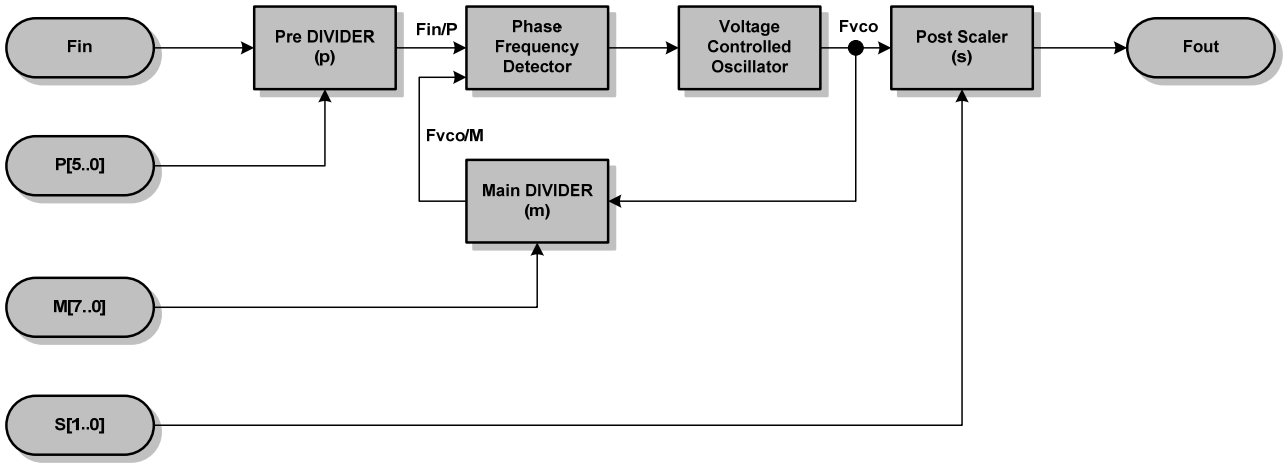


Figure 4-2. Block Diagram of PLL

For the aspect of PLL structure, Figure 4-2 shows the block diagram for one PLL. Fin and Fout indicate input frequency and output frequency respectively. The POLLUX has two PLLs and can generate various programmable clocks by using each PLL.

If the Pre Divider receives a Fin input of 27 MHz, it divides the Fin with 'P'. After that, Phase Frequency Detector (PFD) compares the difference between Fin/P (Reference Clock) and Fvco/M (Feedback Clock). The amplitude of the voltage varies depending on the difference of the values compared between Fin/P and Fvco/M. If the reference clock is faster feed back block, the Voltage Controlled Oscillator (VCO) increases in proportion to the difference. If the reference clock is delayed than the feed clock, VCO is decreased, and it generates an Fvco clock. That is because VCO is a voltage value and plays the role of controlling the clock speed to be faster or slower. At this point, the voltage value is determined by the difference of the values compared between the reference clock and the feedback clock. If the Fvco is not a desired clock, the feed back is recreated through the Main Divider and compared in PFD. These steps are repeated until the reference clock and the feedback clock become equal. If proper FVCO is out, the final Fout clock is created as the divide value(s) of a Post Scaler. Finally, the desired clock frequency is determined by the p, m and s values.

As described above, Fout can be variously set by Fin and p/m/s values (refer to Table 4-2 and 4-3) and the equation to specify p/m/s values is as follows: (Note that all PLL0/1 indicate Fout . Equation may vary for each case.)

- PLL0, PLL1 = (m * Fin)/(p * 2^s)
  (m = MDIV, p = PDIV, s = SDIV = 0, 1, 2, 3)

The range of the MDIV and PDIV values for PLL0 are as follows:

- Range of MDIV Value : 56 ≤ MDIV ≤ 1023
- Range of PDIV Value : 1 ≤ PDIV ≤ 63

The range of the MDIV and PDIV values for PLL1 are as follows:

- Range of MDIV Value : 13 ≤ MDIV ≤ 255
- Range of PDIV Value : 1 ≤ PDIV ≤ 63

The PDIV and the MDIV values should be selected by considering the VCO value and POLLUX's stable operation. The POLLUX has all three PLLs and each PLL has different default values and operation ranges. The basic frequencies for the POLLUX are listed in the table below: (Refer to Table 4-2 and Table 4-3.)

PLL	INITIAL FREQUENCY	RECOMMENDED FREQUENCY (Fvco)	RECOMMENDED FREQUENCY (Fout)	INITIAL PDIV/ MDIV/ SDIV VALUE		
				PDIV	MDIV	SDIV
PLL0	384.000000MHz	500 ~ 1066MHz	16 ~ 533MHz	9	256	1
PLL1	147.461538MHz	250 ~ 600MHz	16 ~ 300MHz	13	142	1

Table 4-1. Initial PDIV/ MDIV/ SDIV Value

For all blocks except for CPU, the operation status (Run/Stop) of the memory controller should be checked before changing the PLL output frequency. In addition, the PLL change bit (***PWRMODE.CHGPLL***) should be set as '1' after PLL Change***(PLLSETREG0, PLLSETREG1).***

4.2.1.2. PDIV/ MDIV/ SDIV Values for PLL0

Input Frequency	Output Frequency (MHz)	PDIV/ MDIV/ SDIV Value			Remark
		PDIV	MDIV	SDIV	
27MHz	533.000000	24	948	1	Maximum Available Frequency
27MHz	500.000000	27	1000	1	
27MHz	490.000000	27	980	1	
27MHz	470.000000	27	940	1	
27MHz	460.000000	27	920	1	
27MHz	470.000000	27	940	1	
27MHz	460.000000	27	920	1	
27MHz	450.000000	9	300	1	
27MHz	440.000000	27	880	1	
27MHz	430.000000	27	860	1	
27MHz	420.000000	9	280	1	
27MHz	410.000000	27	820	1	
27MHz	400.000000	27	800	1	
27MHz	399.000000	27	798	1	
27MHz	390.000000	9	260	1	
27MHz	384.000000	9	256	1	Default (PLL0)
27MHz	350.000000	27	700	1	
27MHz	330.000000	9	220	1	
27MHz	300.000000	**N/A**	**N/A**	**N/A**	Not Available in PLL0
27MHz	266.000000	**N/A**	**N/A**	**N/A**	Not Available in PLL0
27MHz	250.000000	27	1000	2	
27MHz	220.000000	27	880	2	
27MHz	200.000000	27	800	2	
27MHz	166.000000	27	664	2	
27MHz	147.460000	**N/A**	**N/A**	**N/A**	Not Available in PLL0
27MHz	133.000000	**N/A**	**N/A**	**N/A**	Not Available in PLL0
27MHz	100.000000	27	800	3	
27MHz	96.000000	9	256	3	
27MHz	48.000000	9	256	4	

Table 4-2. PDIV/ MDIV/ SDIV Value for PLL0

4.2.1.3. PDIV/ MDIV/ SDIV Values for PLL1

Input Frequency	Output Frequency	PDIV/ MDIV/ SDIV Value			Remark
		PDIV	MDIV	SDIV	

27MHz	300.000000	9	200	1	
27MHz	266.142857	7	138	1	
27MHz	249.750000	6	111	1	
27MHz	220.500000	6	98	1	
27MHz	200.045455	11	163	1	
27MHz	165.970588	17	209	1	
27MHz	147.461538	13	142	1	Default (PLL1)
27MHz	132.750000	6	59	1	
27MHz	100.022727	11	163	2	
27MHz	96.000000	9	128	2	
27MHz	48.000000	9	128	3	

Table 4-3. PDIV/ MDIV/ SDIV Value for PLL1

4.2.1.4. PLL Power Down

The POLLUX supports PLL Power Down mode to minimize power consumption. For example, if all system clocks are generated with PLL0 and PLL1 does not need to be used. Therefore, power does not need to be supplied to the PLL1. In such a case, the POLLUX switches PLL1 into power down mode to reduce the power consumption. However, PLL0 cannot enter to the power down mode. PLL0 power down can be achieved by writing '1' to the ***CLKMODEREG.PLLPWDN1***.

4.2.2. Clock Generator

4.2.2.1. Clocks Summary

The 4 clocks created in the POLLUX and the maximum frequencies for each clock are listed in the table below. The minimum frequency is not limited within the clock frequency limit creatable in PLL.

Clock Name	Min Frequency	Max Frequency	Description
FCLKCPU0	-	533MHz	ARM926EJ CORE CLOCK
HCLKCPU0	-	133MHz	ARM926EJ AHB BUS CLOCK
BCLK	Note*	133MHz	SYSTEM BUS CLOCK(CORE CLOCK) All CORE blocks operates on the basis of BCLK. (MPEG, MEMORY CONTROLLER, DMA, etc…) Note*. Minimum frequency of BCLK is determined by DDR-SDRAM specification.
PCLK	-	66.5MHz	PERIPHERAL BUS CLOCK CPU accesses a block register via I/O with PCLK.

Table 4-4. POLLUX Clock Summary

In Table 4-4, note that the size of PCLK should be the half size of the BCLK when the maximum/minimum frequency values are specified.

4.2.2.2. CPU0 Clock

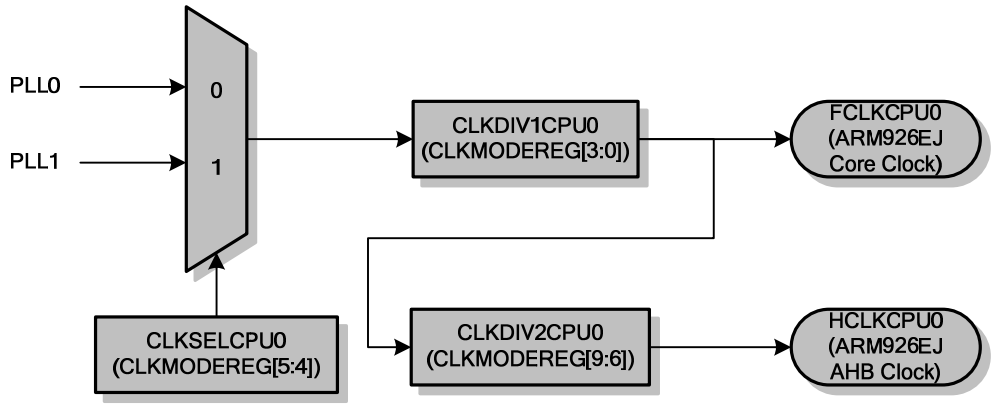


Figure 4-3. 533MHz CPU Clock

Figure 4-3 shows a block diagram that creates the clock supplied to 533MHz, which is the main CPU of the POLLUX. ***CLKMODEREG*** selects a desired PLL output from among the two PLLs. With the clock created from the selected PLL, the ***CLKDIV1CPU0*** register generates ***FCLKCPU0*** to be supplied to the core block of 533MHz and ***HCLKCPU0*** to be supplied to the AHB bus clock. Be careful not to set HCLKCPU0 over 133MHz. The maximum frequency of HCLKCPU0 is 133MHz. Be carefule not to set CLKDIV1CPU0. The frequency of FCLKCPU0 and HCLKCPU0 cannot be the same.

Any PLL can be used to generate the CPU0 clock, but it is recommended to use the PLL0.

4.2.2.3. System Bus Clock (CORE Clock)

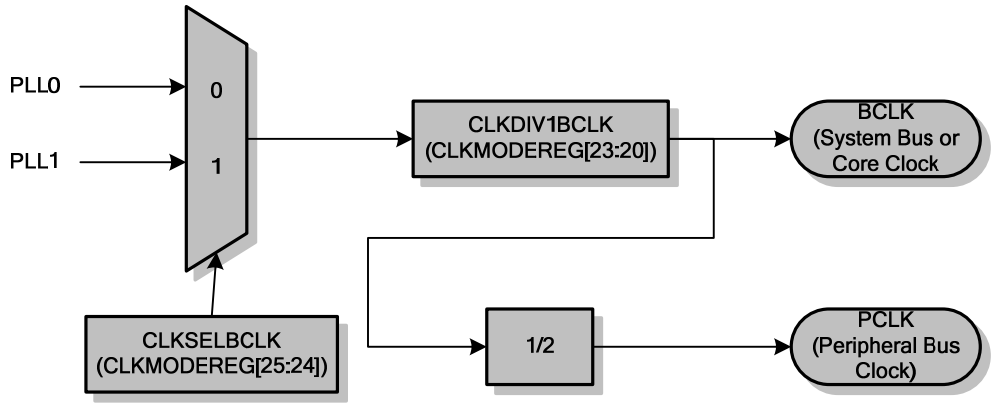


Figure 4-4. System BUS Clock

The system bus clock is called [BCLK] and the half clock of BCLK is called [PCLK]. BCLK is the clock for all SOC Core operations. PCLK is used when the CPU accesses each block register via I/O. Therefore, PCLK should not be applied to the blocks not being used. Every block has PCLK enable/disable Register.The blocks that PCLK is applied to have (refer to each chapter). These registers decide if PCLK is applied to a block only when the CPU accesses the corresponding block register or when it is always applied.

The BCLK frequency should be the double that of the PCLK frequency.

- BCLK Frequency = PCLK Frequency * 2

4.3. Power Manager Overview

The power manager of the POLLUX provides the following functions to operate the system stably and reduce the power consumption.

- Power Up Sequence
- Reset Generation
- Power Management
- Change PLL Value

The key functions of the power manager are to control the Power up Sequence to make the POLLUX stable after the power is supplied to the system and to manage the power effectively. Apart from this, it controls the reset configuration in initial operation.

In addition, this block generates various reset signals, such as External Reset Output (nRSTOUT), GPIO Reset and Soft Reset.

The POLLUX provides various Power Down modes to reduce the system power consumption. The three Power modes provided by the POLLUX are as follows:

- Normal Mode
- IDLE Mode
- Stop Mode

4.4. Power Down Mode Operation

Figure 4-5 shows the state diagram for the Power Management Block. The figure indicates the entry conditions for each Power Down mode and all Wake Up conditions.

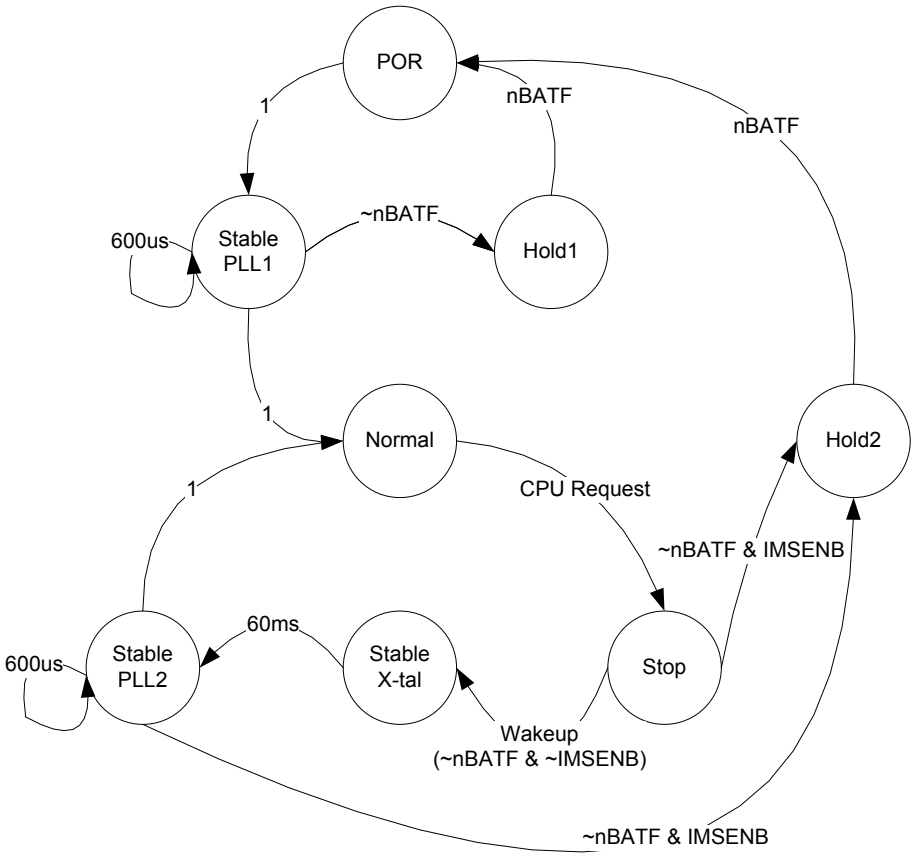


Figure 4-5. Power Management Sequence

<POLLUX State>

- POR : Power On Reset State
- StablePLL : Wait for PLL locking time
- NORMAL : Normal Operation State
- STOP : Stop Operation Mode
- StableX-tal : Wait for Crystal's stable oscillation
- Hold : Wait for nBATF= High.

<Wake Up Source>

- SWRST : Software Reset
- SWRSTENB : Software Reset Enable
- GPIOEvent : GPIO Wake Up Event
- CPUIRQ : Interrupt from CPU
- RTCIRQ : Interrupt from RTC
- PLLChg : PLL Change
- WRST : Watchdog Reset

4.4.1. IDLE Mode

In the IDLE mode, since the power and clocks are supplied to all blocks except for the CPU clock, power consumption can be reduced a bit. To enter to IDLE mode, the ***PWRMODE .CURPWRMODE*** Register should be set as "01". In Idle mode, the CPU clock is not supplied, but the power is normally supplied and PLLs operate normally.

Wake-Up Source can use all the POLLUX interrupts that can be generated by the Interrupt controller: GPIO Interrupt, Alive GPIO Interrupt, External Interrupt and RTC Interrupt. The interrupt for the Wake-Up Source should be enabled before entering to the IDLE Mode. The CPU returns to the previous status immediately after it is woken up in IDLE Mode.

4.4.2. STOP Mode

In STOP mode, the clock is not supplied to all bocks including the ARM Core, because the PLL also does not operate in the clock controller if the clock is not supplied to all blocks. However, the POLLUX converts DRAM into Self Refresh mode to protect memory data before entering to STOP mode. Like IDLE mode, the ***PWRMODE .CURPWRMODE*** should be set as '10' to enter to STOP mode.

The Wake Up source is slightly limited in STOP mode. The available Wake Up sources are RTC Interrupt, GPIO Interrupt, External Interrupt, etc. The Wake Up source is limited because the clock is not supplied to all the other blocks except for the power manager and RTC block. Since the RTC block uses a separate power and clock, only interrupts by the RTC clock can be used as a Wake Up source.

Unlike with IDLE mode, all PLLs stop when the system is woken up in STOP mode so that the system cannot return to the previous status, immediately. Therefore, the Wake Up time is about 70 ms longer than that of IDLE mode to stabilize the internal PLLs.

Power Down Mode	Power Supply	CPU Clock Supply	Other Clock Supply	SDRAM Mode	Wake Up Condition
IDLE MODE	ON	OFF	ON	NORMAL	RTC Interrupt, GPIO Interrupt, All Interrupt to Interrupt Controller, External IRQ
STOP MODE	ON	OFF	OFF	Self Refresh	RTC Interrupt, GPIO Interrupt, External IRQ

Table 4-5. Wake Up Condition and Power Down Mode Status

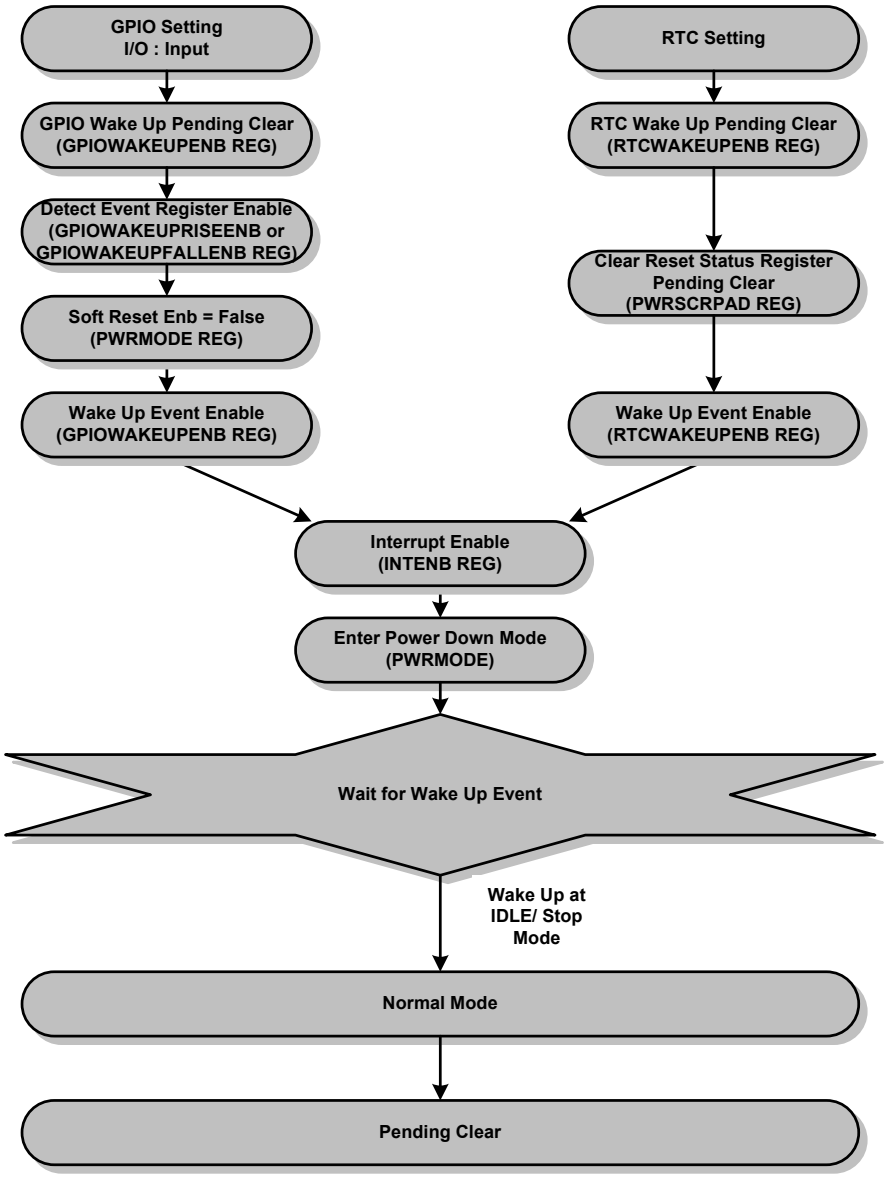


Figure 4-6. Power Down Mode Sequence

Figure 4-6. shows the sequence to enter the Power Down mode and the Wake Up procedure. First, the Wake Up source selects a desired event (interrupt) and specifies the attribute of the event. If the Wake Up Source is GPIO, the setting is changed into Input and the Pending Clear is performed. In addition, the status to detect that an event (interrupt) is specified and Software Reset Enb is set as False for the worst case (If the Software Reset Enb switch is not implemented in terms of Hardware, False does not need to be specified). Finally, the system enables the relevant interrupt (if an interrupt is used) and enters a Power Down mode.

In this Power Down mode, the POLLUX awaits a Wake Up event (interrupt). If the Wake Up event (Interrupt) occurs, the POLLUX returns to normal mode and clears the relevant interrupt pending in terms of Software.

4.4.3. GPIO as a Wake Up Source

GPIO is available for all Power Down modes. However, since the internal power and the clock status are not equal for each power modes, the operation status is a little different at each power mode.

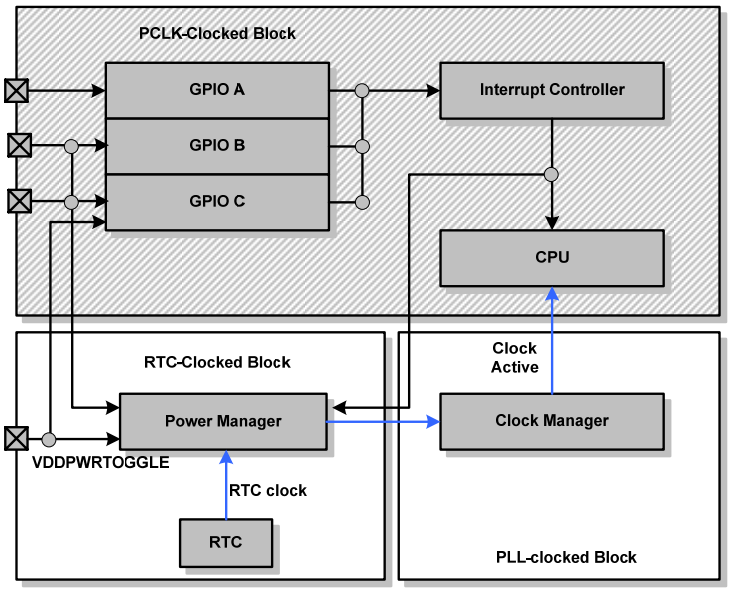


Figure 4-7. Wake Up Block Diagram

As shown in Figure 4-7, GPIO and Power Manager use different clock. Since RTC-clock is always supplied to Power Manager block, the PADs can be used as a Wake Up Source. In addition, the GPIO and the Power Manager blocks receive the same signals coming from GPIOB [31:24], GPIOC [19:0] and VDDPWRTOGGLE.

The description of the Wake Up procedures in Power Down mode is as follows:

- Wake Up in IDLE Mode

During IDLE mode, clock and the power for the other blocks are supplied normally except CPU clock. Therefore, the input received in GPIO is applied to the interrupt controller and wakes up the CPU.

- Wake Up in STOP Mode

In STOP mode, all clocks except for the RTC clock are not supplied (PLL and XTI are included). As shown in Figure 4.7. the interrupt controller does not operate in STOP mode. At this time, if a signal is entered to Power Manager, the power manager wakes up the clock manager, first. As a result of this Wake Up, all clocks, such as the PLL and PCLK, BCLK and FCLK, are enabled and supplied to the CPU and the whole system. In other words, the system is woken up. The Wake Up time is about 70 ms longer than that of IDLE mode to stabilize the PLLs.

4.5. Reset Generation

4.5.1. Power On Reset Sequence

Power management block has the reset generation block. The reset generation block uses the nPORST which is sampled at RTC clock (32.768KHz). And the RTC clock is used as main clock for power management. So Even if the RTC Function is not used, the RTC clock must be supplied.

Figure 4-8 shows the clock and reset behavior during the power-on reset sequence.

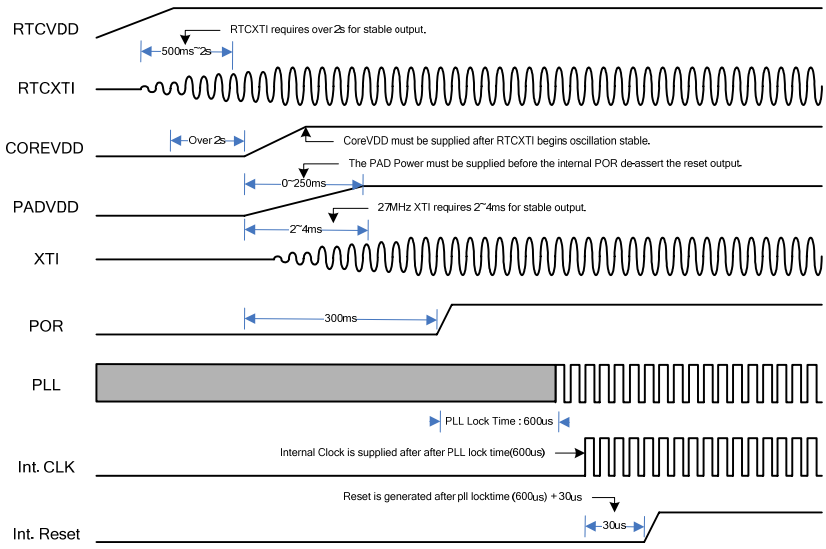


Figure 4-8. Power-On Reset Sequence

nPORST(Power on reset) PAD must be hold to Low(GND) while the crystal oscillator begins stable oscillation after power on. It takes a least 60ms to stabilize oscillations. When nPORST is released after the stabilization of OSC pad (XTI, RTCXTI), the internal PLL start to operate according to the default PLL configuration. But PLL needs the locking time before the stable clock is output. While locking time, power manage doesn't supply the clock for internal logic, After PLL clock is stable, PLL clock output is supplied for each block, then The internal reset and external reset are released after 30us.

4.5.2. Software Reset and GPIO Reset

POLLUX supports Software Reset that CPU can reset itself with Software reset. To generate Software Reset, ***GPIOSWRSTENB*** bit must be set to 1 before setting ***PWRMODE.SWRST*** bit. Software Reset mode does not need the time for stabilization clock because the software reset is requested in stable state differently from power on reset.

POLLUX supports user defined GPIO Reset. The Power management block generates reset when the GPIO pad defined as GPIO Reset source is asserted or de-asserted. The GPIO pad is used as GPIO Reset source are defined at Wakeup Source Register. The ***PWRRSTSTATUS.GPIORESET*** bit set to 1 enables the GPIO Reset source feature.

4.5.3. Watchdog Reset

The Watchdog timer block is used to resume the controller operation whenever it is disturbed by malfunctions such as system error, etc. When power management block detects the event from watchdog timer, it generates exactly the same reset as power on reset because the watchdog reset event occurs in malfunctions and unknown state.

4.5.4. nPORST, Software Reset, Watchdog Reset and GPIO Reset

POLLUX has four reset states as below.

Blocks	Power On Reset	Watchdog Reset	GPIO Reset (Software Reset)	Wake Up (Idle, Stop)
Clock Manager	Reset	Reset	Reset	X
All Core(CPU and etc…)	Reset	Reset	Reset	X
GPIO	Reset	Reset	Reset	X
Power Manager (Except ***LASTPWRMODE*** Register)	Reset	Reset	Reset	X
LASTPWRMODE Register	Reset	X	X	X
RTC Registers (Except ***RTCCNTREAD*** Register)	Reset	Reset	Reset	X

RTCCNTREAD Register	X	X	X	X
nGRESETOUT (Output to PAD)	Reset	Reset	Reset	X

Table 4-6. Reset State

4.6. Change PLL Value

When CPU want to change the PLL divider value. The PLL Change Bit (***PWRMODE.CHGPLL*** bit) must be set to 1 after setting the PLL Setting Reset(***PLLSETREG0, PLLSETREG1***) to appropriate value.

Power management and Clock Controller blocks up the clock supplied to internal controllers because PLLs are unstable when PLL divider value is changed. After locking time, these blocks supply clock. CPU must check whether the blocks run or stop such as STOP mode.

4.7. Register Summary

Bit	R/W	Symbol	Description	Reset Value
CLOCK MODE REGISTER (CLKMODEREG) ***Address : C000_F000h***				
[31]	-	**RESERVED**	Reserved	1'b0
[30]	R/W	**PLLPWDN1**	PLL1 Power Down. 0 : Normal Operation 1 : Power Down	1'b0
[29:26]	R/W	**RESERVED**	Reserved	4'b1
[25:24]	R/W	**CLKSELBCLK**	Select the Clock Source. 00 : PLL0 01 : PLL1 10 : None 11 : None	2'b0
[23:20]	R/W	**CLKDIV1BCLK**	Divide value to create BCLK. For 'N' clock divide, enter an [N-1] value. 0000 ~ 1111 : Divide Value = 1 ~ 16 Ex) For eight clock divide : [0111b]	4'b3
[19:10]	R/W	**RESERVED**	Reserved	4'b0
[9:6]	R/W	**CLKDIV2CPU0**	Divide value to create the AHB bus clock of CPU0 (533MHz). For 'N' clock divide, enter an [N-1] value. The default frequency of AHB bus clock of CPU0 is 96MHz. Be careful that the frequency of AHB bus clock must be lesser than133MHz.. 0001 ~ 1111 : Divide Value = 2 ~ 16 Ex) For eight clock divide : [0111b]	4'b3
[5:4]	R/W	**CLKSELCPU0**	CLKSELCPU0 : Select a clock source. 00 : PLL0 01 : PLL1 10 : None 11 : None	2'b0
[3:0]	R/W	**CLKDIVCPU0**	Divide value to create the core clock of CPU0 (533MHz). For 'N' clock divide, enter an [N-1] value. 0000 ~ 1111 : Divide Value = 1 ~ 16 Ex) For eight clock divide : [0111b]	4'b0
<Note> The CHGPLL bit of the PWRMODE register should be set as '1' to apply the register.				
PLL0 SETTING REGISTER (PLLSETREG0) ***Address : C000_F004h***				
[31:24]	-	**RESERVED**	Reserved	8'h0
[23:18]	R/W	**PDIV**	Pre Divider Value. Divider initially dividing 27 MHz to make PLL0 1 ≤ PDIV ≤ 63	8'h09
[17:8]	R/W	**MDIV**	Main Divider Value. If the value of Fvco is not a desired clock, the value is divided into MDIV again after feedback. (This loop is continuously circulated until a desired clock frequency is made.) 56≤ MDIV ≤ 1023	10'h100
[7:0]	R/W	**SDIV**	Post Scale Divider. Divide the value of Fvco by SDIV to obtain the desired PLL0 value finally. 0 ≤ SDIV ≤ 5	8'h01
PLL1 SETTING REGISTER (PLLSETREG1) ***Address : C000_F008h***				
[31:24]	-	**RESERVED**	Reserved	8'h0
[23:16]	R/W	**PDIV**	Pre Divider Value. Divider initially dividing 27 MHz to make PLL1 1 ≤ PDIV ≤ 63	8'h0D
[15:8]	R/W	**MDIV**	Main Divider Value. If the value of Fvco is not a desired clock, the value is divided into MDIV again after feedback. (This loop is continuously circulated until the desired clock frequency is made.) 13 ≤ MDIV ≤ 255	8'h 8E
[7:0]	R/W	**SDIV**	Post Scale Divider. Divide the value of Fvco to SDIV to obtain the desired PLL1 value finally. 0 ≤ SDIV ≤ 4	8'h01
GPIO Wakeup Enable Register (GPIOWAKEUPENB)				

Bit	R/W	Symbol	Description	Reset Value
Address : C000_F040h				
[31]	R/W	**WKUPEN31**	VDDPWRTOGGLE Wakeup Source Enable.. This bit enables wakeup from power down modes, when user pushed VDDPWRTOGGLE PAD 0 : Disable 1 : Enable	1'b0
[30]	R/W	**WKUPEN30**	Reseved	1'b0
[29]	R/W	**WKUPEN29**	Reseved	1'b0
[28]	R/W	**WKUPEN28**	Reseved	1'b0
[27]	R/W	**WKUPEN27**	GPIOC19 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[26]	R/W	**WKUPEN26**	GPIOC18 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[25]	R/W	**WKUPEN25**	GPIOC17 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[24]	R/W	**WKUPEN24**	GPIOC16 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[23]	R/W	**WKUPEN23**	GPIOC15 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[22]	R/W	**WKUPEN22**	GPIOC14 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[21]	R/W	**WKUPEN21**	GPIOC13 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[20]	R/W	**WKUPEN20**	GPIOC12 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[19]	R/W	**WKUPEN19**	GPIOC11 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[18]	R/W	**WKUPEN18**	GPIOC10 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[17]	R/W	**WKUPEN17**	GPIOC9 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[16]	R/W	**WKUPEN16**	GPIOC8 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[15]	R/W	**WKUPEN15**	GPIOC7 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[14]	R/W	**WKUPEN14**	GPIOC6 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[13]	R/W	**WKUPEN13**	GPIOC5 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[12]	R/W	**WKUPEN12**	GPIOC4 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[11]	R/W	**WKUPEN11**	GPIOC3 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[10]	R/W	**WKUPEN10**	GPIOC2 Wakeup Source Enable	1'b0

Bit	R/W	Symbol	Description	Reset Value
			0 : Disable 1 : Enable	
[9]	R/W	**WKUPEN9**	GPIOC1 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[8]	R/W	**WKUPEN8**	GPIOC0 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[7]	R/W	**WKUPEN7**	GPIOB31 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[6]	R/W	**WKUPEN6**	GPIOB30 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[5]	R/W	**WKUPEN5**	GPIOB29 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[4]	R/W	**WKUPEN4**	GPIOB28 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[3]	R/W	**WKUPEN3**	GPIOB27 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[2]	R/W	**WKUPEN2**	GPIOB26 Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
[1]	R/W	**WKUPEN1**	GPIOB25 Wakeup Source Enable 0 : Disable 1 : Enable	1'b1
[0]	R/W	**WKUPEN0**	GPIOB24 Wakeup Source Enable 0 : Disable 1 : Enable	1'b1

RTC WAKEUP ENABLE REGISTER (RTCWAKEUPENB) *Address : C000_F044h*				
[31 : 1]	-	**RESERVED**	Reserved	31'b0
[0]	R/W	**WKRTC**	RTC Interrupt Wakeup Source Enable 0 : Disable 1 : Enable	1'b0
RISING EDGE DETECT ENABLE REGISTER (GPIOWAKEUPRISEENB) *Address : C000_F048h*				
[31]	R/W	**RISEWKSRC31**	Wakeup Source(VDDPWRTOGGLE) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[30]	R/W	**RISEWKSRC30**	Reserved	1'b0
[29]	R/W	**RISEWKSRC29**	Reserved	1'b0
[28]	R/W	**RISEWKSRC28**	Reserved	1'b0
[27]	R/W	**RISEWKSRC27**	Wakeup Source(GPIOC19) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[26]	R/W	**RISEWKSRC26**	Wakeup Source(GPIOC18) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[25]	R/W	**RISEWKSRC25**	Wakeup Source(GPIOC17) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[24]	R/W	**RISEWKSRC24**	Wakeup Source(GPIOC16) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[23]	R/W	**RISEWKSRC23**	Wakeup Source(GPIOC15) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[22]	R/W	**RISEWKSRC22**	Wakeup Source(GPIOC14) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[21]	R/W	**RISEWKSRC21**	Wakeup Source(GPIOC13) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[20]	R/W	**RISEWKSRC20**	Wakeup Source(GPIOC12) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[19]	R/W	**RISEWKSRC19**	Wakeup Source(GPIOC11) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[18]	R/W	**RISEWKSRC18**	Wakeup Source(GPIOC10) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[17]	R/W	**RISEWKSRC17**	Wakeup Source(GPIOC9) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[16]	R/W	**RISEWKSRC16**	Wakeup Source(GPIOC8) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[15]	R/W	**RISEWKSRC15**	Wakeup Source(GPIOC7) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[14]	R/W	**RISEWKSRC14**	Wakeup Source(GPIOC6) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[13]	R/W	**RISEWKSRC13**	Wakeup Source(GPIOC5) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[12]	R/W	**RISEWKSRC12**	Wakeup Source(GPIOC4) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[11]	R/W	**RISEWKSRC11**	Wakeup Source(GPIOC3) Rising Edge Detect Enable	1'b0

			0 : Disable 1 : Enable	
[10]	R/W	**RISEWKSRC10**	Wakeup Source(GPIOC2) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[9]	R/W	**RISEWKSRC9**	Wakeup Source(GPIOC1) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[8]	R/W	**RISEWKSRC8**	Wakeup Source(GPIOC0) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[7]	R/W	**RISEWKSRC6**	Wakeup Source(GPIOB31) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[6]	R/W	**RISEWKSRC6**	Wakeup Source(GPIOB30) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[5]	R/W	**RISEWKSRC5**	Wakeup Source(GPIOB29) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[4]	R/W	**RISEWKSRC4**	Wakeup Source(GPIOB28) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[3]	R/W	**RISEWKSRC3**	Wakeup Source(GPIOB27) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[2]	R/W	**RISEWKSRC2**	Wakeup Source(GPIOB26) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 0
[1]	R/W	**RISEWKSRC1**	Wakeup Source(GPIOB25) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 1
[0]	R/W	**RISEWKSRC0**	Wakeup Source(GPIOB24) Rising Edge Detect Enable 0 : Disable 1 : Enable	1'b 1
FALLING EDGE DETECT ENABLE REGISTER (GPIOWAKEUPFALLENB) ***Address : C000_F04Ch***				
[31]	R/W	**FALLWKSRC31**	Wakeup Source(VDDPWRTOGGLE) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[30]	R/W	**FALLWKSRC30**	Reserved	
[29	R/W	**FALLWKSRC29**	Reserved	
[28]	R/W	**FALLWKSRC28**	Reserved	
[27]	R/W	**FALLWKSRC27**	Wakeup Source(GPIOC19) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[26]	R/W	**FALLWKSRC26**	Wakeup Source(GPIOC18) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[25]	R/W	**FALLWKSRC25**	Wakeup Source(GPIOC17) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[24]	R/W	**FALLWKSRC24**	Wakeup Source(GPIOC16) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[23]	R/W	**FALLWKSRC23**	Wakeup Source(GPIOC15) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[22]	R/W	**FALLWKSRC22**	Wakeup Source(GPIOC14) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[21]	R/W	**FALLWKSRC21**	Wakeup Source(GPIOC13) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[20]	R/W	**FALLWKSRC20**	Wakeup Source(GPIOC12) Falling Edge Detect Enable 0 : Disable 1 : Enable	

[19]	R/W	**FALLWKSRC19**	Wakeup Source(GPIOC11) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[18]	R/W	**FALLWKSRC18**	Wakeup Source(GPIOC10) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[17]	R/W	**FALLWKSRC17**	Wakeup Source(GPIOC9) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[16]	R/W	**FALLWKSRC16**	Wakeup Source(GPIOC8) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[15]	R/W	**FALLWKSRC15**	Wakeup Source(GPIOC7) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[14]	R/W	**FALLWKSRC14**	Wakeup Source(GPIOC6) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[13]	R/W	**FALLWKSRC13**	Wakeup Source(GPIOC5) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[12]	R/W	**FALLWKSRC12**	Wakeup Source(GPIOC4) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[11]	R/W	**FALLWKSRC11**	Wakeup Source(GPIOC3) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[10]	R/W	**FALLWKSRC10**	Wakeup Source(GPIOC2) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[9]	R/W	**FALLWKSRC9**	Wakeup Source(GPIOC1) Falling Edge Detect Enable 0 : Disable 1 : Enable	
[8]	R/W	**FALLWKSRC8**	Wakeup Source(GPIOC0) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[7]	R/W	**FALLWKSRC7**	Wakeup Source(GPIOB31) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[6]	R/W	**FALLWKSRC6**	Wakeup Source(GPIOB30) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[5]	R/W	**FALLWKSRC5**	Wakeup Source(GPIOB29) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[4]	R/W	**FALLWKSRC4**	Wakeup Source(GPIOB28) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[3]	R/W	**FALLWKSRC3**	Wakeup Source(GPIOB27) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[2]	R/W	**FALLWKSRC2**	Wakeup Source(GPIOB26) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b0
[1]	R/W	**FALLWKSRC1**	Wakeup Source(GPIOB25) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b1
[0]	R/W	**FALLWKSRC0**	Wakeup Source(GPIOB24) Falling Edge Detect Enable 0 : Disable 1 : Enable	1'b1
WAKEUP SOURCE DETECT PENDING REGISTER (GPIOPEND) ***Address : C000_F050h***				
[31]	R/W	**WKDET31**	Wakeup Source(VDDPWRTOGGLE) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[30]	R/W	**WKDET30**	Reserved	1'b0

[29]	R/W	**WKDET29**	Reserved	1'b0
[28]	R/W	**WKDET28**	Reserved	1'b0
[27]	R/W	**WKDET27**	Wakeup Source(GPIOC19) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[26]	R/W	**WKDET26**	Wakeup Source(GPIOC18) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[25]	R/W	**WKDET25**	Wakeup Source(GPIOC17) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[24]	R/W	**WKDET24**	Wakeup Source(GPIOC16) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[23]	R/W	**WKDET23**	Wakeup Source(GPIOC15) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[22]	R/W	**WKDET22**	Wakeup Source(GPIOC14) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[21]	R/W	**WKDET21**	Wakeup Source(GPIOC13) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[20]	R/W	**WKDET20**	Wakeup Source(GPIOC12) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[19]	R/W	**WKDET19**	Wakeup Source(GPIOC11) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[18]	R/W	**WKDET18**	Wakeup Source(GPIOC10) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[17]	R/W	**WKDET17**	Wakeup Source(GPIOC9) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[16]	R/W	**WKDET16**	Wakeup Source(GPIOC8) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[15]	R/W	**WKDET15**	Wakeup Source(GPIOC7) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[14]	R/W	**WKDET14**	Wakeup Source(GPIOC6) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[13]	R/W	**WKDET13**	Wakeup Source(GPIOC5) Detect Read > 0 : None 1 : Interrupt Pended Write > 0 : Not Clear 1 : Clear	1'b0
[12]	R/W	**WKDET12**	Wakeup Source(GPIOC4) Detect	1'b0

			Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	
[11]	R/W	**WKDET11**	Wakeup Source(GPIOC3) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[10]	R/W	**WKDET10**	Wakeup Source(GPIOC2) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[9]	R/W	**WKDET9**	Wakeup Source(GPIOC1) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[8]	R/W	**WKDET8**	Wakeup Source(GPIOC0) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[7]	R/W	**WKDET7**	Wakeup Source(GPIOB31) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[6]	R/W	**WKDET6**	Wakeup Source(GPIOB30) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[5]	R/W	**WKDET5**	Wakeup Source(GPIOB29) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[4]	R/W	**WKDET4**	Wakeup Source(GPIOB28) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[3]	R/W	**WKDET3**	Wakeup Source(GPIOB27) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[2]	R/W	**WKDET2**	Wakeup Source(GPIOB26) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
[1]	R/W	**WKDET1**	Wakeup Source(GPIOB25) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 1
[0]	R/W	**WKDET0**	Wakeup Source(GPIOB24) Detect Read > 0 : None Write > 0 : Not Clear	1 : Interrupt Pended 1 : Clear	1'b 0
<Note> The Wakeup Source Detect Pending Register should be changed to No Clear status after clearing.					
INTERRUPT PENDING & SCRATCH PAD REGISTER (INTPENDSPAD) ***Address : C000_F058h***					
[31 : 15]	-	**RESERVED**	Reserved		18'b0
[14]	W	**BATFW**	Clear GPIO Wakeup Event or Software Reset Event Detect status. 0 : No Effect	1 : Clear	1'bx
[13]	W	**GPIORESETW**	Clear GPIO Wakeup Event or Software Reset Event Detect status. 0 : No Effect	1 : Clear	1'bx
[12]	W	**WATCHDOGRSTW**	Clear Watchdog Reset booting status. 0 : No Effect	1 : Clear	1'bx

[11]	R/W	**POWERONRSTW**	Clear Power-On Reset booting status 0 : No Effect 1 : Clear	1'bx
[10 : 0]	R/W	**SCRPAD**	General purpose register to store data regardless power mode. This register has no effect on other blocks.	11'b0
RESET STATUS REGISTER (PWRRSTSTATUS) ***Address : C000_F05Ch***				
[31 : 3]	-	**RESERVED**	Reserved	29'b0
[3]	R	**BATF**	BATF(Battery Fault) Wakeup Event or Software Reset Event Detect Pending status. BATF Event is low level active. 0 : None 1 : BATF Wakeup or SW Reset Detect	1'b0
[2]	R	**GPIORESETR**	GPIO Wakeup Event or Software Reset Event Detect Pending status. 0 : None 1 : GPIO Wakeup or SW Reset Detect	1'b0
[1]	R	**WATCHDOGRSTR**	Inform Watchdog Reset Booting 0 : None 1 : Watchdog Reset Booting	1'b0
[0]	R	**POWERONRSTR**	Inform Power-On Reset Booting 0 : None 1 : POR Booting	1'b1
<Note> Before the Power mode, the PWRRSTSTATUS register should be cleared.				
INTERRUPT ENABLE REGISTER (INTENB) ***Address : C000_F060h***				
[31 : 3]	-	**RESERVED**	Reserved	31'b0
[2]	R/W	**BATFINTENB**	BATF(Battery Fault) Event Interrupt Enable (Or Software Reset Interrupt) Interrupt occurs when BATF is low level. 0 : Disable 1 : Enable	1'b0
[1]	R/W	**RESERVED**	Reserved	1'b0
[0]	R/W	**GPIOINTENB**	GPIO Event Interrupt Enable (Or Software Reset Interrupt) 0 : Disable 1 : Enable	1'b0
POWER MODE CONTROL REGISTER (PWRMODE) ***Address : C000_F07Ch***				
[31 : 16]	-	**RESERVED**	Reserved	16'b0
[15]	R/W	**CHGPLL**	Change PLL Value with new value defined in PLL Setting Register (PLL0set, PLL1set, PLL2set) in clock Controller Read > 0 : Stable 1 : PLL is Unstable Write > 0 : None 1 : PLL Value Change	1'b0
[14]	R/W	**RESERVED**	Reserved However, PWRMODE[14] bit should be always be '0'.	1'b0
[13]	R/W	**GPIOSWRSTENB**	Soft Reset Enable (Softreset contains GPIO Reset and Software Reset.) 0 : Disable 1 : Enable	1'b0
[12]	W	**SWRST**	Software Reset, SRSTENB should be 1 for reflection 0 : Do Not Reset 1 : Reset	1'b0
[11 : 7]	-	**RESERVED**	Reserved	5'b0
[6 : 4]	R	**LASTPWRMODE**	Get Last Power Mode. Check the last power mode before Watchdog Reset, GPIO Reset and Wakeup. 000 : NORMAL Mode 001 : IDLE Mode 010 : STOP Mode 011 ~ 111 : Reserved	3'b0
[3 : 2]	-	**RESERVED**	Reserved	2'b0
[1 : 0]	R/W	**CURPWRMODE**	Set New Power Down Mode. Set this register if you wish to change the power mode. In Wake Up from a power down mode, the mode is changed into normal mode. 00 : NORMAL Mode 01 : IDLE Mode 10 : STOP Mode 11 : Reserved	2'b0

PAD STRENGTH GPIOA LOW REGISTER (PADSTRENGTHGPIOAL) *Address : C000_F100h*				
[31 : 30]	R/W	**GPIOA15_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA See Appendix for I/O Swithcing Characteristics	01b
[29 :28]	R/W	**GPIOA14_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[27 :26]	R/W	**GPIOA13_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[25 :24]	R/W	**GPIOA12_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[23 :22]	R/W	**GPIOA11_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[21 :20]	R/W	**GPIOA10_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[19 :18]	R/W	**GPIOA9_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[17 :16]	R/W	**GPIOA8_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[15 :14]	R/W	**GPIOA7_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	11b
[13 :12]	R/W	**GPIOA6_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	11b
[11 :10]	R/W	**GPIOA5_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	11b
[9 : 8]	R/W	**GPIOA4_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	11b
[7 : 6]	R/W	**GPIOA3_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	11b
[5 : 4]	R/W	**GPIOA2_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	11b
[3 : 2]	R/W	**GPIOA1_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	11b
[1 : 0]	R/W	**GPIOA0_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	11b
PAD STRENGTH GPIOA HIGH REGISTER (PADSTRENGTHGPIOAH) ***Address : C000_F104h***				
[31 : 30]	R/W	**GPIOA31_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[29 :28]	R/W	**GPIOA30_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA	01b

			10b : 6mA 11b : 8mA	
[27 :26]	R/W	**RESERVED**	Reserved	01b
[25 :24]	R/W	**RESERVED**	Reserved	01b
[23 :22]	R/W	**RESERVED**	Reserved	01b
[21 :20]	R/W	**RESERVED**	Reserved	01b
[19 :18]	R/W	**GPIOA25_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[17 :16]	R/W	**GPIOA24_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[15 :14]	R/W	**GPIOA23_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[13 :12]	R/W	**GPIOA22_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[11 :10]	R/W	**GPIOA21_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[9 : 8]	R/W	**GPIOA20_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[7 : 6]	R/W	**GPIOA19_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[5 : 4]	R/W	**GPIOA18_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[3 : 2]	R/W	**GPIOA17_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[1 : 0]	R/W	**GPIOA16_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
PAD STRENGTH GPIOB LOW REGISTER (PADSTRENGTHGPIOBL) ***Address : C000_F108h***				
[31 : 30]	R/W	**GPIOB15_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[29 :28]	R/W	**GPIOB14_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[27 :26]	R/W	**GPIOB13_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[25 :24]	R/W	**GPIOB12_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[23 :22]	R/W	**GPIOB11_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[21 :20]	R/W	**GPIOB10_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA 10b : 6mA 11b : 8mA	01b
[19 :18]	R/W	**GPIOB9_DRV**	Set Output Drive Strength 00b : 2mA 01b : 4mA	01b

			10b : 6mA	11b : 8mA	
[17 :16]	R/W	**GPIOB8_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[15 :14]	R/W	**GPIOB7_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[13 :12]	R/W	**GPIOB6_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[11 :10]	R/W	**GPIOB5_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[9 : 8]	R/W	**GPIOB4_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[7 : 6]	R/W	**GPIOB3_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[5 : 4]	R/W	**GPIOB2_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[3 : 2]	R/W	**GPIOB1_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[1 : 0]	R/W	**GPIOB0_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
PAD STRENGTH GPIOB HIGH REGISTER (PADSTRENGTHGPIOBH) ***Address : C000_F10Ch***					
[31 : 30]	R/W	**GPIOB31_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[29 :28]	R/W	**GPIOB30_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[27 :26]	R/W	**GPIOB29_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[25 :24]	R/W	**GPIOB28_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[23 :22]	R/W	**GPIOB27_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[21 :20]	R/W	**GPIOB26_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[19 :18]	R/W	**GPIOB25_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[17 :16]	R/W	**GPIOB24_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[15 :14]	R/W	**GPIOB23_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[13 :12]	R/W	**GPIOB22_DRV**	Set Output Drive Strength 00b : 2mA	01b : 4mA	11b

			10b : 6mA	11b : 8mA	
[11 :10]	R/W	**GPIOB21_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[9 : 8]	R/W	**GPIOB20_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[7 : 6]	R/W	**GPIOB19_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[5 : 4]	R/W	**GPIOB18_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[3 : 2]	R/W	**GPIOB17_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[1 : 0]	R/W	**GPIOB16_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
PAD STRENGTH GPIOC LOW REGISTER (PADSTRENGTHGPIOCL) ***Address : C000_F110h***					
[31 : 30]	R/W	**GPIOC15_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[29 :28]	R/W	**GPIOC14_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[27 :26]	R/W	**GPIOC13_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[25 :24]	R/W	**GPIOC12_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[23 :22]	R/W	**GPIOC11_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[21 :20]	R/W	**GPIOC10_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[19 :18]	R/W	**GPIOC9_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[17 :16]	R/W	**GPIOC8_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[15 :14]	R/W	**GPIOC7_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[13 :12]	R/W	**GPIOC6_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[11 :10]	R/W	**GPIOC5_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[9 : 8]	R/W	**GPIOC4_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[7 : 6]	R/W	**GPIOC3_DRV**	Set Output Drive Strength 00b : 2mA	01b : 4mA	01b

			10b : 6mA	11b : 8mA	
[5 : 4]	R/W	**GPIOC2_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[3 : 2]	R/W	**GPIOC1_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
[1 : 0]	R/W	**GPIOC0_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	01b
PAD STRENGTH GPIOC HIGH REGISTER (PADSTRENGTHGPIOCH) ***Address : C000_F114h***					
[31 : 8]	R	**RESERVED**	Reserved		-
[7 : 6]	R/W	**GPIOC19_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[5 : 4]	R/W	**GPIOC18_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[3 : 2]	R/W	**GPIOC17_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[1 : 0]	R/W	**GPIOC16_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
PAD STRENGTH BUS REGISTER (PADSTRENGTHBUS) ***Address : C000_F118h***					
[31 : 30]	R/W	**DDR_CNTL_DRV**	Set Output Drive Strength 00b : 4mA 10b : 10mA	01b : 8mA 11b : 14mA	01b
[29 :28]	R/W	**DDR_ADDR _DRV**	Set Output Drive Strength 00b : 4mA 10b : 10mA	01b : 8mA 11b : 14mA	01b
[27 :26]	R/W	**DDR_DATA_DRV**	Set Output Drive Strength 00b : 4mA 10b : 10mA	01b : 8mA 11b : 14mA	01b
[25 :24]	R/W	**DDR_CLK_DRV**	Set Output Drive Strength 00b : 6mA 10b : 18mA	01b : 12mA 11b : 24mA	01b
[23 :22]	R/W	**STATIC_CNTL_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[21 :20]	R/W	**STATIC_ADDR_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[19 :18]	R/W	**STATIC_DATA_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[17 :8]	R/W	**RESERVED**	Reserved		-
[7 : 6]	R/W	**VSYNC_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[5 : 4]	R/W	**HSYNC_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b
[3 : 2]	R/W	**DE_DRV**	Set Output Drive Strength 00b : 2mA	01b : 4mA	11b

			10b : 6mA	11b : 8mA	
[1 : 0]	R/W	**PVCLK_DRV**	Set Output Drive Strength 00b : 2mA 10b : 6mA	01b : 4mA 11b : 8mA	11b

CHAPTER 5.

ID REGISTER(ECID)

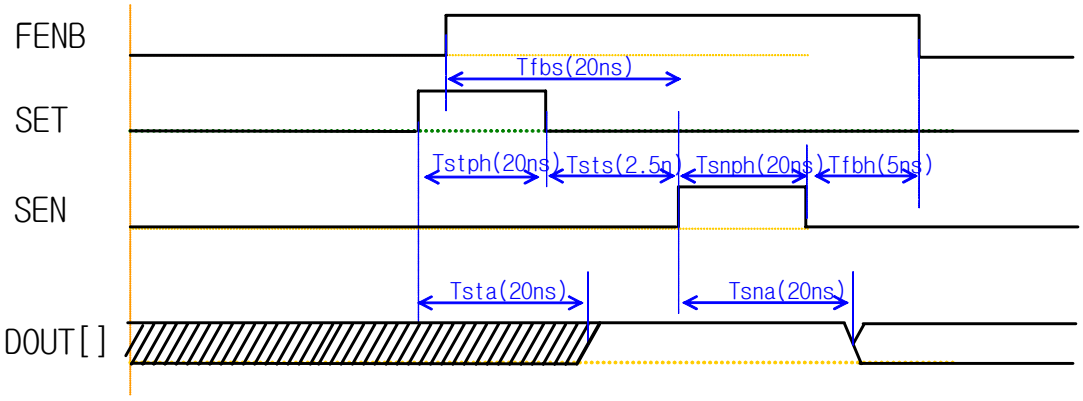
5. ID Register (ECID)

5.1. ECID Overview

ECID module of the POLLUX stores the 128bit DIEID information on a e-fuse ROM. Each die will have its own DIEID to identify its lot number, wafer number and position on wafer.

●

5.2. ECID timing



FVDD:hi Z SDI:don`t care
ACK:low BCK:low
FEN:low SDOUT:hold
Violation : Fail

5.3. Register Summary

Bit	R/W	Symbol	Description	Reset Value
ID0 REGISTER (ID0REG) *Address : C001_F800h*				
[31:0]	R	**ECID0**	ECID Block output date[31:0]	–
ID1 REGISTER (ID1REG) *Address : C001_F804h*				
[31:0]	R	**ECID1**	ECID Block output date[63:32]	–
ID2 REGISTER (ID2REG) *Address : C001_F808h*				
[31:0]	R	**ECID2**	ECID Block output date[95:64]	–
ID3 REGISTER (ID3REG) *Address : C001_F80Ch*				
[31:0]	R	**ECID3**	ECID Block output date[127:96]	–
CHIP NAME REGISTER (CHIPNAME3~0) *Address : C001_F810h*				
[31:0]	R	**CHIPNAME3_0**	Chip Name 3~0	-

Bit	R/W	Symbol	Description	Reset Value
			CHIPNAME3_0 to **CHIPNAME47_44** is chip name register. Chip name is 48bytes string, "MAGICEYES-POLLUX——————".	
CHIP NAME REGISTER (CHIPNAME7~4) *Address : C001_F814h*				
[31:0]	R	**CHIPNAME7_4**	Chip Name 7~4	–
CHIP NAME REGISTER (CHIPNAME11~8) *Address : C001_F818h*				
[31:0]	R	**CHIPNAME11_8**	Chip Name 11~8	–
CHIP NAME REGISTER (CHIPNAME15~12) *Address : C001_F81Ch*				
[31:0]	R	**CHIPNAME15_12**	Chip Name 15~12	–
CHIP NAME REGISTER (CHIPNAME19~16) *Address : C001_F820h*				
[31:0]	R	**CHIPNAME19_16**	Chip Name 19~16	–
CHIP NAME REGISTER (CHIPNAME23~20) *Address : C001_F824h*				
[31:0]	R	**CHIPNAME23_20**	Chip Name 23~20	–
CHIP NAME REGISTER (CHIPNAME27~24) *Address : C001_F828h*				
[31:0]	R	**CHIPNAME27_24**	Chip Name 27~24	–
CHIP NAME REGISTER (CHIPNAME31~28) *Address : C001_F82Ch*				
[31:0]	R	**CHIPNAME31_28**	Chip Name 31~28	–
CHIP NAME REGISTER (CHIPNAME35~32) *Address : C001_F830h*				
[31:0]	R	**CHIPNAME35_32**	Chip Name 35~32	–
CHIP NAME REGISTER (CHIPNAME39~36) *Address : C001_F834h*				
[31:0]	R	**CHIPNAME39_36**	Chip Name 39~36	–
CHIP NAME REGISTER (CHIPNAME43~40) *Address : C001_F838h*				
[31:0]	R	**CHIPNAME43_40**	Chip Name 43~40	–
CHIP NAME REGISTER (CHIPNAME47~44) *Address : C001_F83Ch*				
[31:0]	R	**CHIPNAME47_44**	Chip Name 47~44	–
RESERVED (Reserved) *Address : C001_F840h*				
[31:0]	R	**RESERVED**	Reserved	–
GUID0 REGISTER *Address : C001_F844h*				
[31:0]	R	**GUID_DATA1**	GUID.Data1 GUID0, GUID1, GUID2, GUID3 registers consist of 128 bit GUID. The GUID structure in C-language is like this:	0x89b6495c

Bit	R/W	Symbol	Description	Reset Value
			struct { unsigned long Data1; unsigned short Data2; unsigned short Data3; unsigned char Data4[8]; } GUID; 128 GUID value is 128'h89B6495C_4298_42E0_BA73_F59670752302	
GUID1 REGISTER ***Address : C001_F848h***				
[31:16]	R	**GUID_DATA2**	GUID.Data2	0x4298
[15: 0]	R	**GUID_DATA3**	GUID.Data3	0x42e0
GUID2 REGISTER ***Address : C001_F84Ch***				
[31:24]	R	**GUID_DATA4_3**	GUID.Data4[3]. See GUID0 Register	0xba
[23:16]	R	**GUID_DATA4_2**	GUID.Data4[2]	0x73
[15: 8]	R	**GUID_DATA4_1**	GUID.Data4[1]	0xf5
[7: 0]	R	**GUID_DATA4_0**	GUID.Data4[0]	0x96
GUID3 REGISTER ***Address : C001_F850h***				
[31:24]	R	**GUID_DATA4_7**	GUID.Data[7]. See GUID0 Register	0x70
[23:16]	R	**GUID_DATA4_6**	GUID.Data[6]	0x75
[15: 8]	R	**GUID_DATA4_5**	GUID.Data[5]	0x23
[7: 0]	R	**GUID_DATA4_4**	GUID.Data[4]	0x2
ECID CONTROL (EC) ***Address : C001_F854h***				
[31:7]	R	**RESERVED**	Reserved	–
[6]	R/W	**FENB**	Enable the blowing	0x0
[5]	R/W	**SET**	Initialize the latches prior to fuse sense	0x0
[4]	R/W	**SEN**	Sense the fuse	0x0
[3: 2]	R	**RESERVED**	Reserved	–
[1: 0]	R	**BONDINGID**	Bonding ID Read	0x0

CHAPTER 6.

GPIO CONTROLLER

6. GPIO Controller

6.1. Overview

The POLLUX has a total of 87 GPIO pins. Among them, 34 GPIO pins are available as Dedicated GPIO.
The entire GPIO is divided into three GPIO groups as shown in Figure 6-1.

6.1.1. Block Diagram

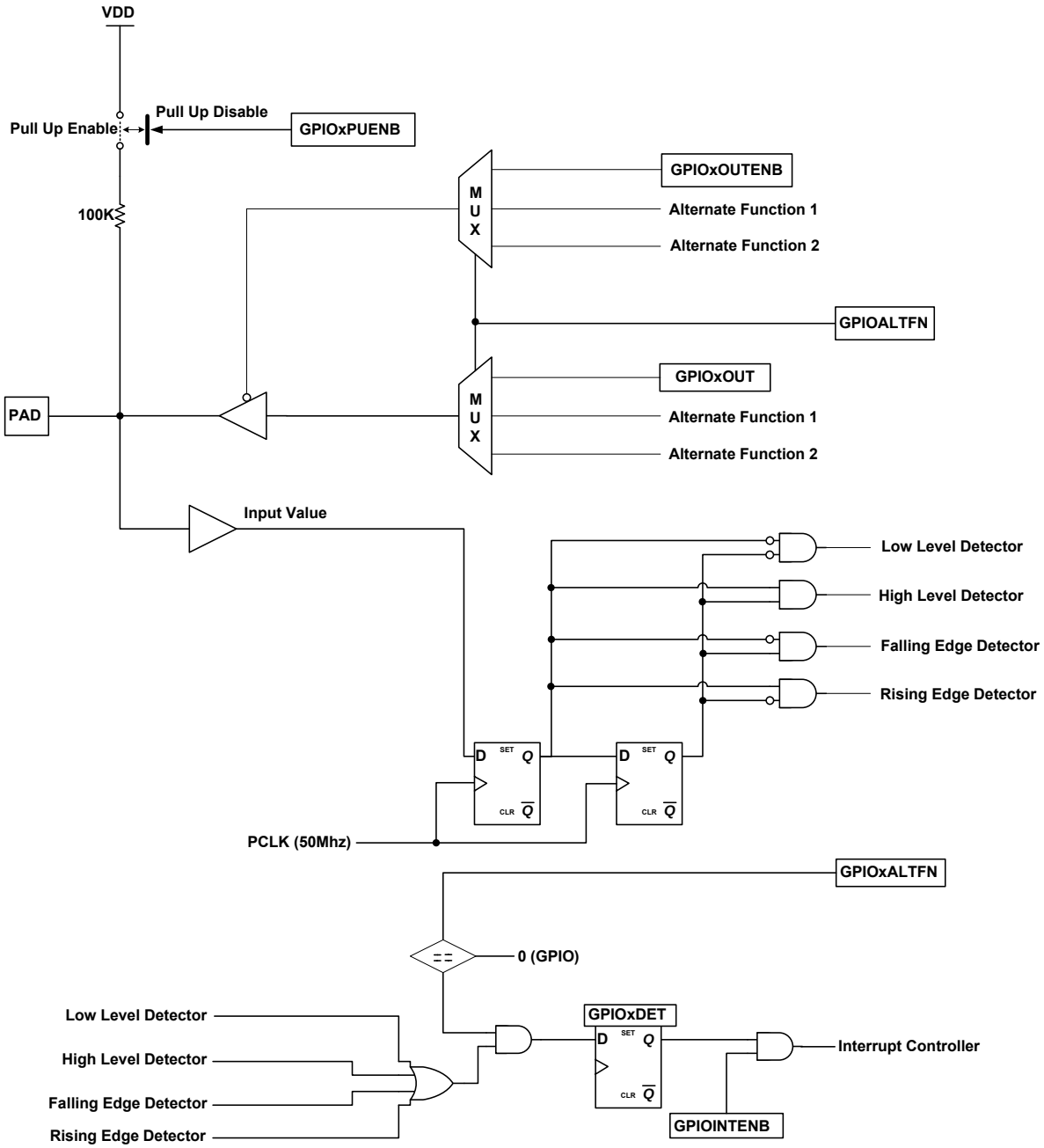


Figure 6-1. GPIO Block Diagram

6.2. Operation

POLLUX GPIO Pins have an internal pull-up resistance of 100K ohm. The current of the pull-up resistance (for VDD = 3.3 V and V(PAD) = 0V) is listed in the table below:

Pull Up	MIN	TYP	MAX	UNIT
ENABLE	10	33	72	uA
DISABLE	-	-	0.1	uA

Table 6-1. Pull-Up Resistor Current

Most POLLUX GPIO ports contain the Alternate function (some ports supports up to Alternate Function2). All GPIO ports should be set as GPIO function or Alternate Function suitable for the user's purposes and this setting can easily be performed with GPIO registers. In addition, all GPIO pull-up resistances are enabled/disabled. This setting operates when the system is fully booted and does not affect the system in initial booting. If there is a value which needs to be determined in system booting, the value is given by inserting a pull up/down resistance from outside.

6.2.1. Input Operation

To use GPIO for input, the GPIO function should be selected by setting the relevant bit of the GPIO Alternate Function Select register as b'00 to select the GPIO function. In addition, the GPIO Input mode should be, also, selected by the GPIOx Output Enable register(***GPIOxOUTENB***) as '0'.

An input signal is detected by selecting a desired detection type with the GPIOx Event Detect Mode register. Four types of input signal can be detected: Low Level, High Level, Falling edge and Rising edge. The GPIOx Event Detect Mode registers consist of GPIOx Event Detect Mode register0 (***GPIOxDETMODE0***) and GPIOx Event Detect Mode register1 (***GPIOxDETMODE1***).

To use interrupt, set the GPIOx Interrupt Enable Register (***GPIOxINTENB***) as '1'.

The GPIOx Event Detect Register (***GPIOxDET***) enables checking the generation of an event via GPIO and may be used as the Pending Clear function when an interrupt occurs.

When the GPIOx PAD Status Register (***GPIOxPAD***) is set as GPIO Input mode, the level of the relevant GPIOx PAD can be checked.

The Open drain pins (GPIOB[21], GPIOB[22] and GPIOB[23]) operate in Input mode only when the GPIOx Output register (*GPIOxOUT*) is set as '1'. The Open drain pins are operated by the GPIOx Output Register (*GPIOxOUT*) even if the GPIOx Output Enable register (*GPIOxOUTENB*) is set as Input mode.

6.2.2. Output Operation

To use GPIO for output, the GPIO function should be selected by setting the relevant bit of the GPIOx. Alternate Function Select register should be set as b'00 to select the GPIO function. In addition, the GPIOx Output mode should also be selected by setting the GPIOx Output Enable register as '1'.

If you set a desired output value (low level: '0', high level: '1') with the GPIOx Output Register (GPIOxOUT), the value is reflected to the corresponding bit.

The Open drain pins (GPIOB[21], GPIOB[22] and GPIOB[23]) operate in output mode only when the GPIOx Output register (*GPIOxOUT*) is set as '0'. The Open drain pins are operated by the GPIOx Output Register (*GPIOxOUT*) even if the GPIOx Output Enable register (*GPIOxOUTENB*) is set as Input mode.

6.2.3. Alternate Function Operation

Among the 87 GPIO pins of the POLLUX, most GPIO pins have an Alternate function. However, the Alternate function and the GPIO function should not be used simultaneously. Therefore, Alternate Function1 and Alternate Function2 are operated by setting the corresponding bits of the GPIOx Alternate Function Select register as b'01 and b'10, respectively.

6.3. GPIO PIN FUNCTIONS

GPIO Pins	Alternate Function 1	Alternate Function 2
GPIOA[31 : 0]		
GPIOA[31]	PWMOUT[1]	-
GPIOA[30]	PWMOUT[0]	
GPIOA[29]	SDA[1]	
GPIOA[28]	SCL[1]	
GPIOA[27]	SDA[0]	
GPIOA[26]	SCL[0]	
GPIOA[25]	I2SMCLK	
GPIOA[24]	I2SSYNC	
GPIOA[23]	I2SDIN	
GPIOA[22]	I2SBCLK	
GPIOA[21]	I2SDOUT	
GPIOA[20]	RX[3]	
GPIOA[19]	TX[3]	
GPIOA[18]	RX[2]	
GPIOA[17]	TX[2]	
GPIOA[16]	RX[1]	
GPIOA[15]	TX[1]	
GPIOA[14]	NDTR[1]	
GPIOA[13]	NDSR[1]	
GPIOA[12]	NDCD[1]	
GPIOA[11]	NRIO[1]	
GPIOA[10]	NRTS[1]	
GPIOA[9]	NCTS[1]	
GPIOA[8]	TX[0]	
GPIOA[7]	PVD[7]	
GPIOA[6]	PVD[6]	
GPIOA[5]	PVD[5]	
GPIOA[4]	PVD[4]	
GPIOA[3]	PVD[3]	
GPIOA[2]	PVD[2]	
GPIOA[1]	PVD[1]	
GPIOA[0]	PVD[0]	
GPIOB[31 : 0]		
GPIOB[31]	PVD[23]	-
GPIOB[30]	PVD[22]	
GPIOB[29]	PVD[21]	
GPIOB[28]	PVD[20]	

GPIO Pins	Alternate Function 1	Alternate Function 2
GPIOB[27]	PVD[19]	
GPIOB[26]	PVD[18]	
GPIOB[25]	PVD[17]	
GPIOB[24]	PVD[16]	
GPIOB[23]	PVD[15]	
GPIOB[22]	PVD[14]	
GPIOB[21]	PVD[13]	
GPIOB[20]	PVD[12]	
GPIOB[19]	PVD[11]	
GPIOB[18]	PVD[10]	
GPIOB[17]	PVD[9]	
GPIOB[16]	PVD[8]	
GPIOB[15]	SSPTXD[0]	
GPIOB[14]	SSPRXD[0]	
GPIOB[13]	SSPCLK[0]	
GPIOB[12]	SSPFRM[0]	
GPIOB[11]	SDDAT1[3]	
GPIOB[10]	SDDAT1[2]	
GPIOB[9]	SDDAT1[1]	
GPIOB[8]	SDDAT1[0]	
GPIOB[7]	SDCMD1	
GPIOB[6]	SDCLK1	
GPIOB[5]	SDDAT0[3]	
GPIOB[4]	SDDAT0[2]	
GPIOB[3]	SDDAT0[1]	
GPIOB[2]	SDDAT0[0]	
GPIOB[1]	SDCMD0	
GPIOB[0]	SDCLK0	
GPIOC[20 : 0]		
GPIOC[20]	**VDDPWRTOGGLE (input mode only)** **VDDPWRTOGGLE PAD status could be detected as program the GPIOC[20] bit of GPIO Control Registers. General GPIOs are able to drive the PAD, i.e output mode, but GPIOC[c] is not.**	
GPIOC[19]	nSCS[6]	
GPIOC[18]	nSCS [5]	
GPIOC[17]	nSCS [4]	
GPIOC[16]	nSCS [3]	nICE2
GPIOC[15]	nSCS [2]	nICE1
GPIOC[14]	SA[25]	
GPIOC[13]	SA[24]	
GPIOC[12]	SA[23]	

GPIO Pins	Alternate Function 1	Alternate Function 2
GPIOC[11]	SA[22]	
GPIOC[10]	SA[21]	
GPIOC[9]	SA[20]	
GPIOC[8]	SA[19]	
GPIOC[7]	PWMOUT[2]	
GPIOC[6]	SSPTXD[1]	
GPIOC[5]	SSPRXD[1]	
GPIOC[4]	SSPCLK[1]	
GPIOC[3]	SSPFRM[1]	
GPIOC[2]	NSCS[9]	
GPIOC[1]	NSCS[8]	
GPIOC[0]	NSCS[7]	

Table 6-2. GPIO Pin Functions

Register Summary

6.3.1. GPIOx Control Register (x = A, B, C)

Bit	R/W	Symbol	Description	Reset Value
GPIOx OUTPUT REGISTER (GPIOxOUT) ***Address : GPIOA: C000_A000h / GPIOB: C000_A040h / GPIOC: C000_A080h***				
[31:0]	R/W	**GPIOXOUT**	GPIOx[31:0]: Specifies the output value in GPIOx output mode. Note: This bit should be set as '1' (Input mode) or '0' (Output mode) to use the Open drain pins in Input/Output mode. 0:Low Level 1:High Level	32'h0
GPIOx OUTPUT ENABLE REGISTER (GPIOxOUTENB) ***Address : GPIOA: C000_A004h / GPIOB: C000_A044h/ GPIOC: C000_A084h***				
[31:0]	R/W	**GPIOXOUTENB**	GPIOx[31:0]: Specifies GPIOx In/Out mode. Note: The Open drain pins are operated in Input/Output mode by the GPIOxOUTPUT register (GPIOxOUT) and not by this bit. 0:Input Mode 1:Output Mode	32'h0
GPIOx EVENT DETECT MODE REGISTER 0 (GPIOxDETMODE0) ***Address : GPIOA: C000_A008h/ GPIOB: C000_A048h/ GPIOC: C000_A088h***				
[31:30]	R/W	**GPIOXDETMODE0_15**	GPIOx[15]: Specifies Detect mode when GPIOx 15pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[29:28]	R/W	**GPIOXDETMODE0_14**	GPIOx[14]: Specifies Detect mode when GPIOx 14pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[27:26]	R/W	**GPIOXDETMODE0_13**	GPIOx[13]: Specifies Detect mode when GPIOx 13pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[25:24]	R/W	**GPIOXDETMODE0_12**	GPIOx[12]: Specifies Detect mode when GPIOx 12pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[23:22]	R/W	**GPIOXDETMODE0_11**	GPIOx[11]: Specifies Detect mode when GPIOx 11pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[21:20]	R/W	**GPIOXDETMODE0_10**	GPIOx[10]: Specifies Detect mode when GPIOx 10pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[19:18]	R/W	**GPIOXDETMODE0_9**	GPIOx[9]: Specifies Detect mode when GPIOx 9pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[17:16]	R/W	**GPIOXDETMODE0_8**	GPIOx[8]: Specifies Detect mode when GPIOx 8pin is in Input mode.. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[15:14]	R/W	**GPIOXDETMODE0_7**	GPIOx[7]: Specifies Detect mode when GPIOx 7pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[13:12]	R/W	**GPIOXDETMODE0_6**	GPIOx[6]: Specifies Detect mode when GPIOx 6pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00

Bit	R/W	Symbol	Description	Reset Value
[11:10]	R/W	**GPIOXDETMODE0_5**	GPIOx[5]: Specifies Detect mode when GPIOx 5pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[9:8]	R/W	**GPIOXDETMODE0_4**	GPIOx[4]: Specifies Detect mode when GPIOx 4pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[7:6]	R/W	**GPIOXDETMODE0_3**	GPIOx[3]: Specifies Detect mode when GPIOx 3pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[5:4]	R/W	**GPIOXDETMODE0_2**	GPIOx[2]: Specifies Detect mode when GPIOx 2pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[3:2]	R/W	**GPIOXDETMODE0_1**	GPIOx[1]: Specifies Detect mode when GPIOx 1pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[1:0]	R/W	**GPIOXDETMODE0_0**	GPIOx[0]: Specifies Detect mode when GPIOx 0pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
GPIOx EVENT DETECT MODE REGISTER 1 (GPIOxDETMODE1) ***Address : GPIOA: C000_A00Ch/ GPIOB: C000_A04Ch/ GPIOC: C000_A08Ch***				
[31:30]	R/W	**GPIOXDETMODE1_31**	GPIOx[31]: Specifies Detect mode when GPIOx 31pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[29:28]	R/W	**GPIOXDETMODE1_30**	GPIOx[30]: Specifies Detect mode when GPIOx 30pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[27:26]	R/W	**GPIOXDETMODE1_29**	GPIOx[29]: Specifies Detect mode when GPIOx 29pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[25:24]	R/W	**GPIOXDETMODE1_28**	GPIOx[28]: Specifies Detect mode when GPIOx 28pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[23:22]	R/W	**GPIOXDETMODE1_27**	GPIOx[27]: Specifies Detect mode when GPIOx 27pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[21:20]	R/W	**GPIOXDETMODE1_26**	GPIOx[26]: Specifies Detect mode when GPIOx 26pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[19:18]	R/W	**GPIOXDETMODE1_25**	GPIOx[25]: Specifies Detect mode when GPIOx 25pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[17:16]	R/W	**GPIOXDETMODE1_24**	GPIOx[24]: Specifies Detect mode when GPIOx 24pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[15:14]	R/W	**GPIOXDETMODE1_23**	GPIOx[23]: Specifies Detect mode when GPIOx 23pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00

Bit	R/W	Symbol	Description	Reset Value
[13:12]	R/W	**GPIOXDETMODE1_22**	GPIOx[22]: Specifies Detect mode when GPIOx 22pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[11:10]	R/W	**GPIOXDETMODE1_21**	GPIOx[21]: Specifies Detect mode when GPIOx 21pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[9:8]	R/W	**GPIOXDETMODE1_20**	GPIOx[20]: Specifies Detect mode when GPIOx 20pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[7:6]	R/W	**GPIOXDETMODE1_19**	GPIOx[19]: Specifies Detect mode when GPIOx 19pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[5:4]	R/W	**GPIOXDETMODE1_18**	GPIOx[18]: Specifies Detect mode when GPIOx 18pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[3:2]	R/W	**GPIOXDETMODE1_17**	GPIOx[17]: Specifies Detect mode when GPIOx 17pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
[1:0]	R/W	**GPIOXDETMODE1_16**	GPIOx[16]: Specifies Detect mode when GPIOx 16pin is in Input mode. 00 : Low Level 01 : High Level 10 : Falling Edge 11 : Rising Edge	2'b00
GPIOx INTERRUPT ENABLE REGISTER (GPIOxINTENB) ***Address : GPIOA: C000_A010h/ GPIOB: C000_A050h/ GPIOC: C000_A090h***				
[31:0]	R/W	**GPIOXINTENB**	GPIOx[31:0]: Specifies the use of an interrupt when a GPIOx Event occurs. The events specified in GPIOxDETMODE0 and GPIOxDETMODE1 are used. 0 : Disable 1 : Enable	32'h0
GPIOx EVENT DETECT REGISTER (GPIOxDET) ***Address : GPIOA: C000_A014h/ GPIOB: C000_A054h/ GPIOC: C000_A094h***				
[31:0]	R/W	**GPIOXDET**	GPIOx[31:0]: Shows if an event is detected in accordance with Event Detect mode in GPIOx Input Mode. Set '1' to clear the relevant bit. GPIOx[31:0] is used as a Pending register when an interrupt occurs. Read : 0 : Not Detect 1 : Detected Write : 0: Not Clear 1: Clear	32'h0
GPIOx PAD STATUS REGISTER (GPIOxPAD) ***Address : GPIOA: C000_A018h/ GPIOB: C000_A058h/ GPIOC: C000_A098h***				
[31:0]	R	**GPIOXPAD**	GPIOx[31:0]: Can read the level pf PAD in GPIOx Input mode. The data read in this register is the data not passing a filter and reflects the PAD status itself. 0 : Low Level 1 : High Level	32'h0
GPIOx PULL UP ENABLE REGISTER (GPIOxPUENB) ***Address : GPIOA: C000_ A01Ch/ GPIOB: C000_A05Ch/ GPIOC: C000_ A09Ch***				
[31:0]	R/W	**GPIOXPUENB**	GPIOx[31:0]: Decides the use of the Pull-up resistance (100 Kohm) of GPIOx PAD. 0: Pull-Up Disable 1: Pull-Up Enable	32'h0
GPIOx ALTERNATE FUNCTION SELECT REGISTER 0 (GPIOxALTFN0) ***Address : GPIOA: C000_A020h/ GPIOB: C000_ A060h/ GPIOC: C000_A0A0h***				
[31:30]	R/W	**GPIOXALTFN0_15**	GPIOx[15]: Selects the function of GPIOx 15pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00

Bit	R/W	Symbol	Description	Reset Value
[29:28]	R/W	**GPIOXALTFN0_14**	GPIOx[14]: Selects the function of GPIOx 14pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[27:26]	R/W	**GPIOXALTFN0_13**	GPIOx[13]: Selects the function of GPIOx 13pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[25:24]	R/W	**GPIOXALTFN0_12**	GPIOx[12]: Selects the function of GPIOx 12pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[23:22]	R/W	**GPIOXALTFN0_11**	GPIOx[11]: Selects the function of GPIOx 11pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[21:20]	R/W	**GPIOXALTFN0_10**	GPIOx[10]: Selects the function of GPIOx 10pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[19:18]	R/W	**GPIOXALTFN0_9**	GPIOx[9]: Selects the function of GPIOx 9pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[17:16]	R/W	**GPIOXALTFN0_8**	GPIOx[8]: Selects the function of GPIOx 8pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[15:14]	R/W	**GPIOXALTFN0_7**	GPIOx[7]: Selects the function of GPIOx 7pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[13:12]	R/W	**GPIOXALTFN0_6**	GPIOx[6]: Selects the function of GPIOx 6pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[11:10]	R/W	**GPIOXALTFN0_5**	GPIOx[5]: Selects the function of GPIOx 5pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[9:8]	R/W	**GPIOXALTFN0_4**	GPIOx[4]: Selects the function of GPIOx 4pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[7:6]	R/W	**GPIOXALTFN0_3**	GPIOx[3]: Selects the function of GPIOx 3pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[5:4]	R/W	**GPIOXALTFN0_2**	GPIOx[2]: Selects the function of GPIOx 2pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[3:2]	R/W	**GPIOXALTFN0_1**	GPIOx[1]: Selects the function of GPIOx 1pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[1:0]	R/W	**GPIOXALTFN0_0**	GPIOx[0]: Selects the function of GPIOx 0pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00

GPIOx ALTERNATE FUNCTION SELECT REGISTER 1 (GPIOxALTFN1)

Address : GPIOA: C000_A024h/ GPIOB: C000_A064h/ GPIOC: C000_A0A4h

Bit	R/W	Symbol	Description	Reset Value
[31:30]	R/W	**GPIOXALTFN1_31**	GPIOx[31]: Selects the function of GPIOx 31pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[29:28]	R/W	**GPIOXALTFN1_30**	GPIOx[30]: Selects the function of GPIOx 30pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[27:26]	R/W	**GPIOXALTFN1_29**	GPIOx[29]: Selects the function of GPIOx 29pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[25:24]	R/W	**GPIOXALTFN1_28**	GPIOx[28]: Selects the function of GPIOx 28pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[23:22]	R/W	**GPIOXALTFN1_27**	GPIOx[27]: Selects the function of GPIOx 27pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[21:20]	R/W	**GPIOXALTFN1_26**	GPIOx[26]: Selects the function of GPIOx 26pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[19:18]	R/W	**GPIOXALTFN1_25**	GPIOx[25]: Selects the function of GPIOx 25pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[17:16]	R/W	**GPIOXALTFN1_24**	GPIOx[24]: Selects the function of GPIOx 24pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[15:14]	R/W	**GPIOXALTFN1_23**	GPIOx[23]: Selects the function of GPIOx 23pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[13:12]	R/W	**GPIOXALTFN1_22**	GPIOx[22]: Selects the function of GPIOx 22pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[11:10]	R/W	**GPIOXALTFN1_21**	GPIOx[21]: Selects the function of GPIOx 21pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[9:8]	R/W	**GPIOXALTFN1_20**	GPIOx[20]: Selects the function of GPIOx 20pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[7:6]	R/W	**GPIOXALTFN1_19**	GPIOx[19]: Selects the function of GPIOx 19pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[5:4]	R/W	**GPIOXALTFN1_18**	GPIOx[18]: Selects the function of GPIOx 18pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[3:2]	R/W	**GPIOXALTFN1_17**	GPIOx[17]: Selects the function of GPIOx 17pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00
[1:0]	R/W	**GPIOXALTFN1_16**	GPIOx[16]: Selects the function of GPIOx 16pin. 00 : GPIO 01 : ALT Function1 10 : ALT Function2 11 : Reserved	2'b00

CHAPTER 7.

MEMORY CONTROLLER

7. MEMORY CONTROLLER

7.1. Overview

The POLLUX Memory Controller is based on a Unified Memory Architecture (UMA). This Controller consists of two control units: MCU-Y, MCU-S. Each unit has dedicated control pins.

7.1.1. Unified Memory Architecture (UMA)

- Supported Memory controller
  - DDR-SDRAM memory
  - Static memory
- Single Bank of DDR-SDRAM(16bit data bus width)
  - Support Active Power down mode
  - Support Self Refresh mode
- Ten Static Memory Chip Selects
- NAND Flash Interface
- Two Separate Memory Controller
  - MCU-Y : DDR-SDRAM
  - MCU-S : Static Memory
- 26-bit address support using latch address
- NAND Boot function (Large/ Small block support)
- 512-byte NAND copy function to 0 address
- SLC NAND, MLC NAND with ECC(Support BCH-algorithm)

7.1.2. Block Diagram

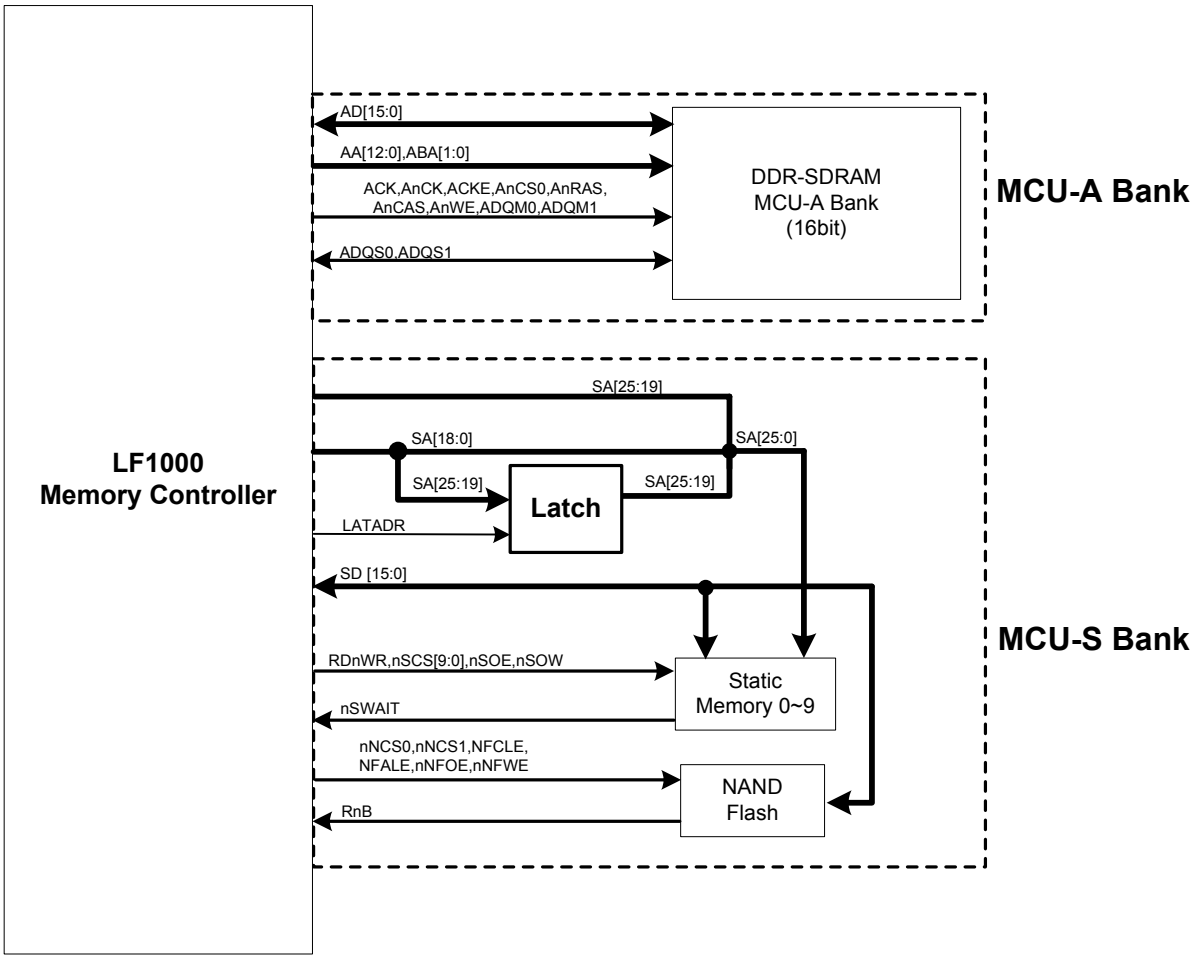


Figure 7-1. Memory Controller Block Diagram

7.1.3. Pin Function Description

Pin Name	GPIO No	GPIO Function	Type	Description
MCU-Y Bank				
YD[15:0]	-	-	I/O	MCU-Y Bank DDR-SDRAM Data bus. YD[15:0] is used for 16-bit data mode.
YA[12:0]	-	-	O	MCU-Y Bank DDR-SDRAM Address bus. This bus signals the address requested for memory accesses.
YBA[1:0]	-	-	O	MCU-Y Bank DDR-SDRAM Bank address.
YCK.PAD	-	-	O	MCU-Y Bank DDR-SDRAM clock.
YCK.PADB	-	-	O	MCU-Y Bank DDR-SDRAM inverting clock.
YKE	-	-	O	MCU-Y Bank DDR-SDRAM clock enable.
YnCS[0]	-	-	O	MCU-Y Bank DDR-SDRAM CS for bank0. This signal be connected to the chip select pin for DDR-SDRAM.
YnRAS	-	-	O	MCU-Y Bank DDR-SDRAM RAS. This signal should be connected to the row address strobe pin for all bank of DDR-SDRAM.
YnCAS	-	-	O	MCU-Y Bank DDR-SDRAM RAS. This signal should be connected to the column address strobe pin for all bank of DDR-SDRAM.
YnWE	-	-	O	MCU-Y Bank DDR-SDRAM write enable. This signal should be connected to the write enable for DDR-SDRAM.
YDQS[0]	-	-	I/O	MCU-Y Bank DDR-SDRAM Data strobe0.
YDQS[1]	-	-	I/O	MCU-Y Bank DDR-SDRAM Data strobe1.

Pin Name	GPIO No	GPIO Function	Type	Description
YDQM[0]	-	-	O	MCU-Y Bank DDR-SDRAM DQM for data bytes 0. These signals should be connected to the data output mask enable for DDR-SDRAM.
YDQM[1]	-	-	O	MCU-Y Bank DDR-SDRAM DQM for data bytes 1. These signals should be connected to the data output mask enable for DDR-SDRAM.
MCU-S(Static Memory)				
SD[15:0]	-	-	I/O	Static Memory/NAND Data bus.
SA[25:1]	-	-	O	Static Memory Address bus
SA[0](nSDQM[0])	-	-	O	Static Memory Address bus. This signal should be connected to the low byte enable for 16bit SRAM.
SA[1]	-		O	Static Memory Address bus.
SA[8:2]/SA25:19]	-	-	O	Static Memory Address bus.
SA[18:8]	-	-	O	Static Memory Address bus.
nSDQM[1]	-	-	O	Static Memory High Byte Enable.
LATADDR	-	-	O	Static Memory Latch Address Enable.
RDnWR	-	-	O	Buffer Direction Control. Read/Write for static interface. Intended for use as a steering signal for buffering logic.
nSCS[0]	-	-	O	Static Memory Bnak0 Chip Select.
nSCS[1]	-	-	O	Static Memory Bnak1 Chip Select.
nSCS[2]	-	-	O	Static Memory Bnak2 Chip Select.
nSCS[3]	-	-	O	Static Memory Bnak3 Chip Select.
nSCS[4]	-	-	O	Static Memory Bnak4 Chip Select.
nSCS[5]	-	-	O	Static Memory Bnak5 Chip Select.
nSCS[6]	-	-	O	Static Memory Bnak6 Chip Select.
nSCS[7]	GPIOC[0]	ALT1	O	Static Memory Bnak7 Chip Select.
nSCS[8]	GPIOC[1]	ALT1	O	Static Memory Bnak8 Chip Select.
nSCS[9]	GPIOC[1]	ALT1	O	Static Memory Bnak9 Chip Select.
nSWAIT	-	-	I	Wait Control for Static Memory. This signal is an input and is driven low by the Static Memory to extend the length of the transfers to/from applications processor.
nSOE	-	-	O	Static Memory Read Enable.
nSWE	-	-	O	Static Memory Write Enable.
MCU-S(NAND)				
SD[15:0]	-	-	I/O	NAND/Static Memory Data bus.
nNCS[0]	-	-	O	NAND Chip Select0. This is chip selects to NAND Flash memory.
nNCS[1]	-	-	O	NAND Chip Select1. This is chip selects to NAND Flash memory.
NFCLE	-	-	O	NAND CLE IO Read Enable. This is command latch enable to NAND Flash memory.
NFALE	-	-	O	NAND ALE IO Write Enable. This is address latch enable to NAND Flash memory.
RnB	-	-	I	NAND Ready & Busy. This is Ready/Busy signal of NAND Flash memory.
nNFOE	-	-	O	NAND Read Enable.
nNFWE	-	-	O	NAND Write Enable.

Table 7-1. Memory Controller Pin Function Description

7.2. Operation

The memory controller area of the POLLUX is divided into an MCU-Y and MCU-S bank. The MCU-Y bank is connected to DDR-

SDRAM, which is the main memory of the POLLUX and 16-bit data bus width.
The MCU-S bank is the static bank and can be connected to Static Memory/Device, NAND (Refer to Figure5-2 Memory Map)

7.2.1. MCU-Y Bank Feature

- The MCU-Y bank have one chip select signal.
- The data bus widths of the MCU-Y bank 16-bit.
  - For system configuration setting for the MCU-Y bank, refer to **POLLUX Application Note – SYSTEM CONFIGURATION.**
- The capacity of a DDR-SDRAM IC that can be connected to the MCU-Y bank is between 64Mbit ~ 512Mbit.
- The total capacity of DDR-SDRAM memory is up to 128Mbyte.
- Supported DDR Type
  - Burst Length : 2
  - Burst Type : Sequencial
  - CAS Letency : 2, 2.5, 3
- DDR-Command
  - DESELECT(NOP)
  - NO OPERATION (NOP)
  - ACTIVE (Select band and active row) (ACT)
  - READ (Select bank and column and start READ burst) (READ)
  - WRITE (Select bank and column and start WRITE burst) (WRITE)
  - PRECHARGE, PRECHARGE ALL,
  - BURST TERMINATE
  - AUTO REFRESH
  - LOAD MODE REGISTER (LMR), LOAD EXTENDED MODE REGISTER

7.2.2. MCU-S Bank Feature

- Latched Addressing
  - The number of pins that are connected to the outside is ADDR [18:0]. Since, however, the total address of the MCU-S Bank is 26-bit. ADDR[8:2] and ADDR[25:19] are allotted to the same pin, the system has a structure in which addresses are output two times. If the system uses ADDR[19] or more, the setting of higher address (ADDR[25:19]) is possible via System Configuration Pin (CfgSTLATADD). In this event, ADDR[25:19] which is configured by using external Latch IC first should be latched. (For more detailed information, refer to **POLLUX Application Note – MCU-S.**)
- 16-bit data bus width
  - Static memory register except NAND flash(8bits) has bus width select registers
- Static memory Controller
  - Normal static memory (SRAM, ROM and FRAM) or static devices are connected.
  - Up to ten Static Chip Select signals exist.
- NAND Flash Controller
  - It supports both the small and large block NAND flash memories.
  - Up to two NAND flash memories can be connected.
  - Supports both SLC and MLC NAND flash memories.
  - Up to 4-bit error correction/512byte using Binary-BCH coding

7.2.3. Boot Mode

POLLUX provides various Boot modes as follows:

CfgBOOTENB	CfgShadow	CfgBOOTMODE	CfgNFBUSWidth	CfgSTBUSWidth	Boot Mode
X	1'b0	2'b10	x	1'b0	8bit Static(NOR/ROM) Boot
X	1'b0	2'b10	x	1'b1	16bit Static(NOR/ROM) Boot

CfgBOOTENB	CfgShadow	CfgBOOTMODE	CfgNFBUSWidth	CfgSTBUSWidth	Boot Mode
1'b1	1' b1	2' b01	1' b0	x	8bit NAND Boot
1'b1	1' b1	2' b00	x	x	UART Boot

Table 7-2. Boot Mode

- 8bit Static Boot case
  - Shadow Mode(CfgShadow) : 1'b0(Shadow disable)
  - Boot Mode(CfgBOOTMODE) : 2'b10(Static boot mode)
  - Static data bus width(CfgSTBUSWidth) : 1'b0(8bit bus width Static memory)
- 16bit Static Boot case
  - Shadow Mode(CfgShadow) : 1'b0(Shadow disable)
  - Boot Mode(CfgBOOTMODE) : 2'b10(Static boot mode)
  - Static data bus width(CfgSTBUSWidth) : 1'b1(16bit bus width Static memory)
- 8bit NAND Boot case (POLLUX do not support 16bit NAND Boot)
  - Shadow Mode(CfgShadow) : 1'b1(Shadow enable)
  - Boot Mode(CfgBOOTMODE) : 2'b01(NAND boot mode)
  - NAND Flash data bus width(CfgNFBUSWidth) : 1'b0(8bit bus width NAND flash)
  - NAND Boot Size(CfgBOOTENB) : 1'b1(Boot Mode Enable) only
- UART Boot case (For more detailed information, refer to **Chapter14. UART – 14.2.10. UART Boot**)
  - Shadow Mode(CfgShadow) : 1'b1(Shadow enable)
  - Boot Mode(CfgBOOTMODE) : 2'b00(UART boot mode)
  - UART Boot Size(CfgBOOTENB) : 1'b1(Boot Mode Enable) only

7.2.4. Memory Map

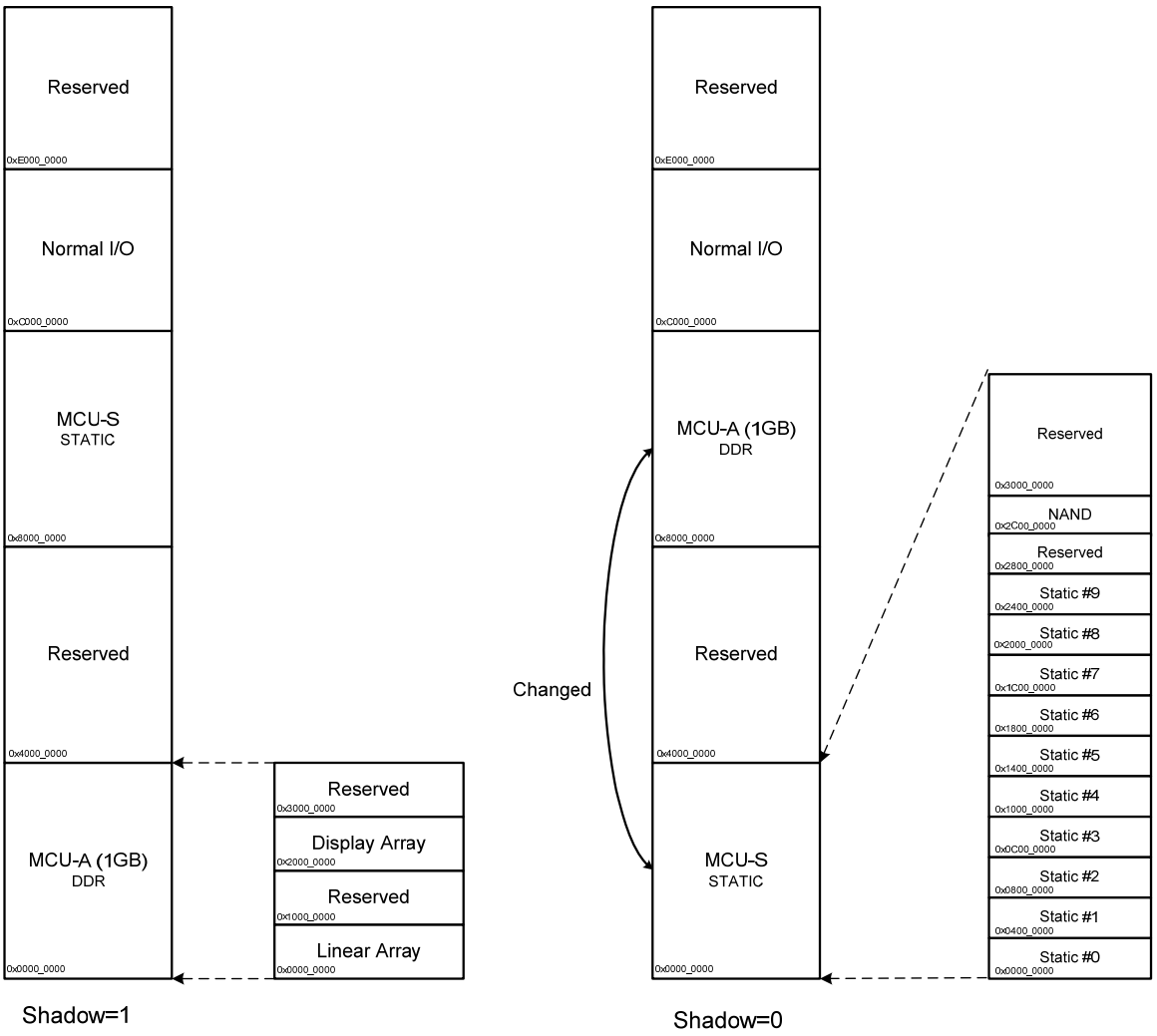


Figure 7-2. Memory Map

The memory map is roughly divided into one DDR-SDRAM Bank (MCU-Y) and one static bank (MCU-S). The static bank consists of NAND Flash controller, Static Memory controller. The MCU-Y bank consist of a Linear Array area and Display Array area.

The locations of the DDR-SDRAM bank (MCU-Y) and the Static Memory bank (MCU-S) are changed depending on the CfgShadow (Shadow mode) setting.

For a system using static boot, its Shadow mode should be set as '0'. For a system using NAND boot or UART boot, however the Shadow mode should be set as '1'.

7.3. Block Address

As shown in Figure 7-2, the memory map sizes of the MCU-Y bank are 1Gbyte, respectively. However, this size is the sum of the two reserved areas, which are the Linear Array area and Display Array area. In actual fact, the total sums of DDR-SDRAM memories that can be connected to MCU-Y banks is 256 Mbytes (Linear Array area), respectively. In addition, the access to the mirrored area, Display Array area, is possible. Since the access to the Display Array area is performed with the block address method, not the linear address method, the access via the Display Array area can obtain a higher operation speed then that obtained via the Linear Array area for the display buffer memory in a program. (In the access to the Display Array area, the memory controller optimizes the DDR-SDRAM access method in the block address method in accordance with the video data format.) If, for example, shadow is '1' and the display buffer memory address is 0x0100_0000 to 0x0120_0000, the program should access 0x2100_0000 to 0x2120_0000, not 0x0100_0000 to 0x0120_0000. In other words, the program should access the memory area where the offset address of the Display Array area, which is 0x2000_000, is added to Display Buffer memory address. ADDR[29:28] is used for the selection of the Linear Array and the Display Array area and ADDR[31:30] is used for the selection of MCU-Y, MCU-S and normal I/O within the total memory map of POLLUX.

7.3.1. Linear Array

The POLLUX accesses the memory into linear address (i.e. access to ADDR[29:28] to 2'b00), rather than converting an address.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Segment				Y[11:0]												X[11:0]											
Bank		Mode		Super Segment				Y Coordinates in super segment[11:0]												X Coordinates in super segment[11:0]											

Table 7-3. Linear Array Mode

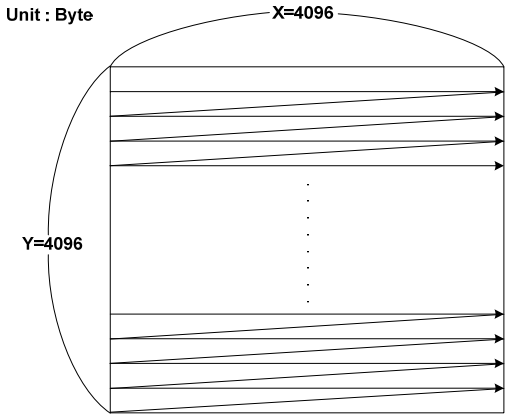


Figure 7-3. Linear Array Mode

7.3.2. Display Array

If the POLLUX accesses to the memory area, that is, the offset of 0x2000_0000 is added to the Linear Array area address (i.e. access to ADDR[29:28] to 2'b10), the Display Array area is accessed. At this point, the memory access is performed with the Block address method, instead of general Linear address method. For this type of access to the display array area by the block address method, however, the ***MEMCONTROL.BLOCKDISP*** set as '1'. Note that the access to the Display Array area is performed with Linear Address method, not the Block Address method, if the register is set as '0'.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	Segment				Y[11:5]						X[11:6]							Y[4:0]					X[5:0]					
Bank		Mode		Super Segment				Block index in super segment[12:0]													Block Address [10:0]										

Table 7-4. Block Address Mode

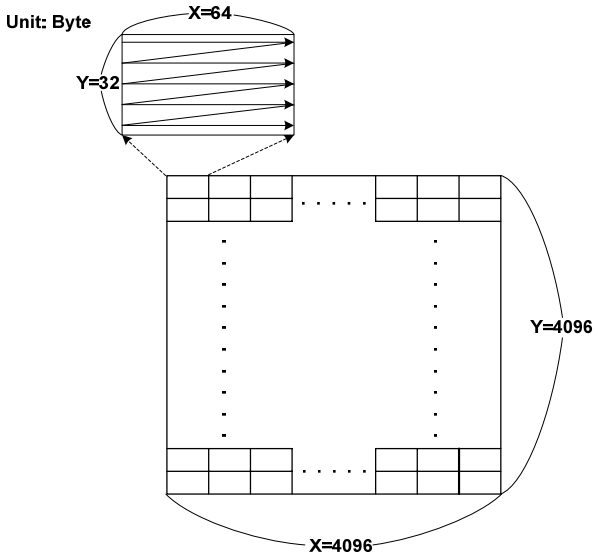


Figure 7-4. Block Address Mode

7.4. Memory Controller Timing

7.4.1. DDR-SDRAM Timing

7.4.1.1. DDR-SDRAM Write Timing

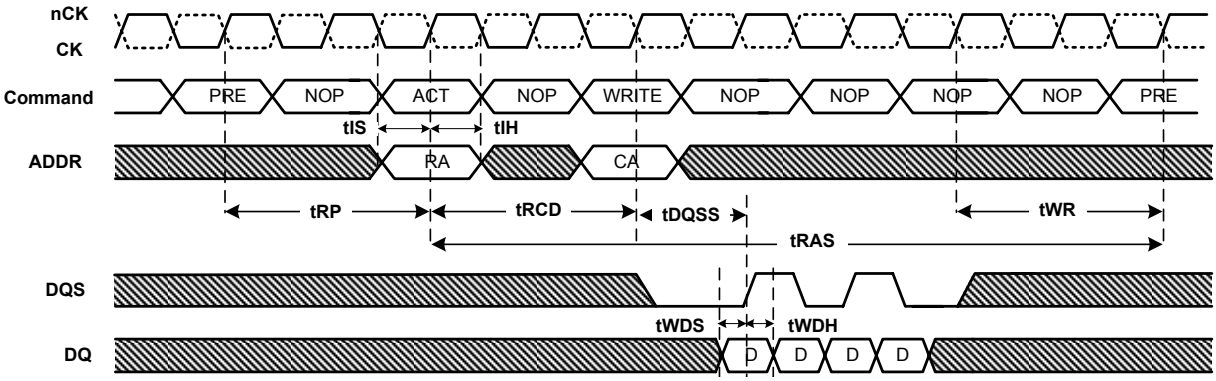


Figure 7-5. DDR-SDRAM Write Timing

7.4.1.2. DDR-SDRAM Read Timing

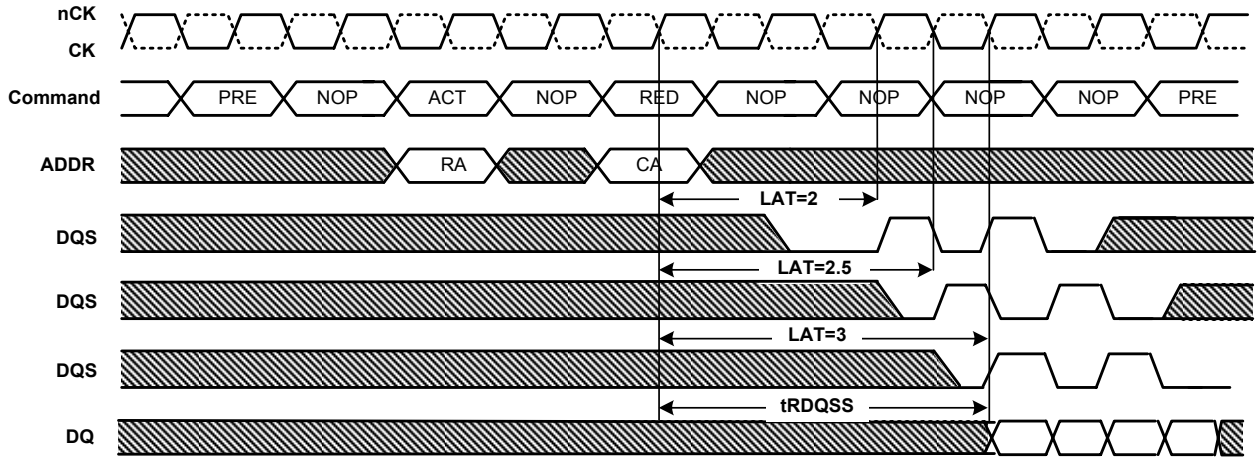


Figure 7-6. DDR-SDRAM Read Timing

7.4.1.3. DDR-SDRAM Initial Timing

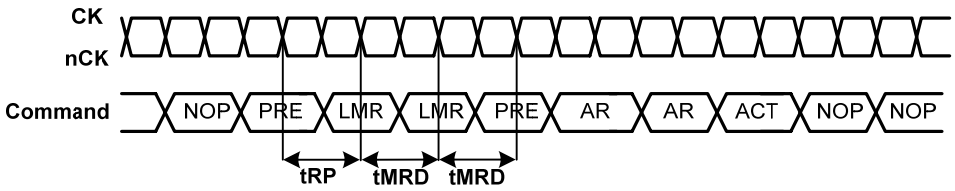


Figure 7-7. DDR-SDRAM Initial Timing

7.4.1.4. DDR-SDRAM Timing Diagram Symbol Description

Symbol	Min	Max	Units	Description
tCK	7.5	10	ns	Clock cycle time
tCKL	tCK/2 – 0.3	tCK /2+ 0.3	ns	CK Low Level Width
tCKH	tCK/2 – 0.3	tCK/2 + 0.3	ns	CK High Level Width
tIS	2.36	3.6	ns	Command input setup time
tIH	2.75	3.99	ns	Command input hold time
tDQSS	tCK * 2 – 0.3	tCK * 2 + 0.3	ns	Write command to first DQS time
tWDS	1.6	2.9	ns	Write data setup time

tWDH	1.08	2.32	ns	Write data hold time
tRDQSS	LAT – 0.3	LAT + 0.3	ns	Read command to first DQS time
tRDS	-0.4	-	ns	Read data setup time
tRDH	0.5	-	ns	Read data hold time
tRP	1	16	tCK	Row Precharge Time
tRCD	1	16	tCK	RAS to CAS Delay Time
tRAS	1	16	tCK	Row Active Time
tWR	0	15	tCK	Write Recovery Time
LAT	1* tCK	3* tCK	ns	CAS Latency
tMRD	1	16-	TCK	Mode Register cycle time

Table 7-5. Symbol Description

7.4.2. Static Memory Timing

7.4.2.1. No Wait Static Memory Timing

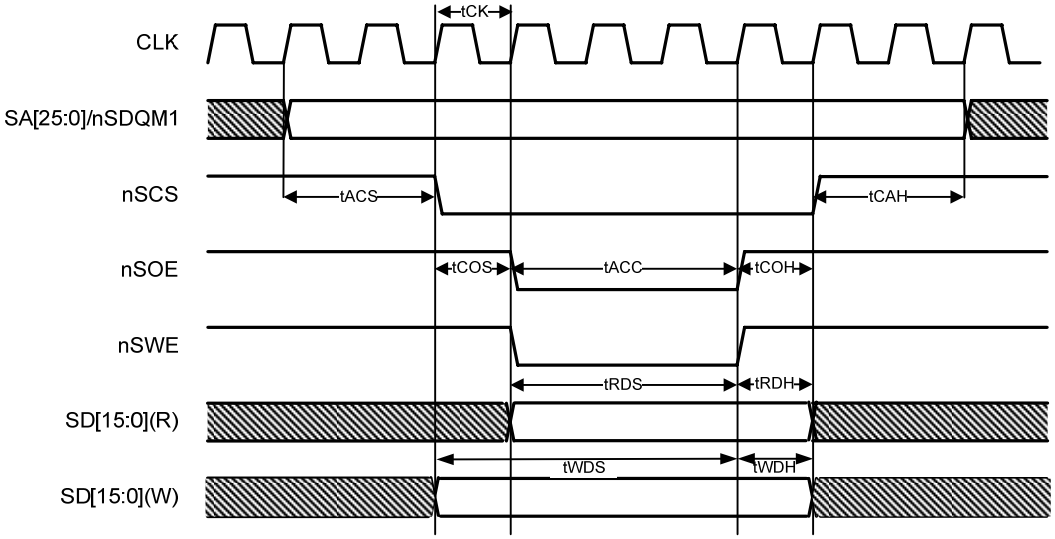


Figure 7-8. No Wait Timing (CLK=BCLK (Typ.100MHz))

7.4.2.2. Wait Static Memory Timing

Wait cycles is not available during burst reading and writing operation.

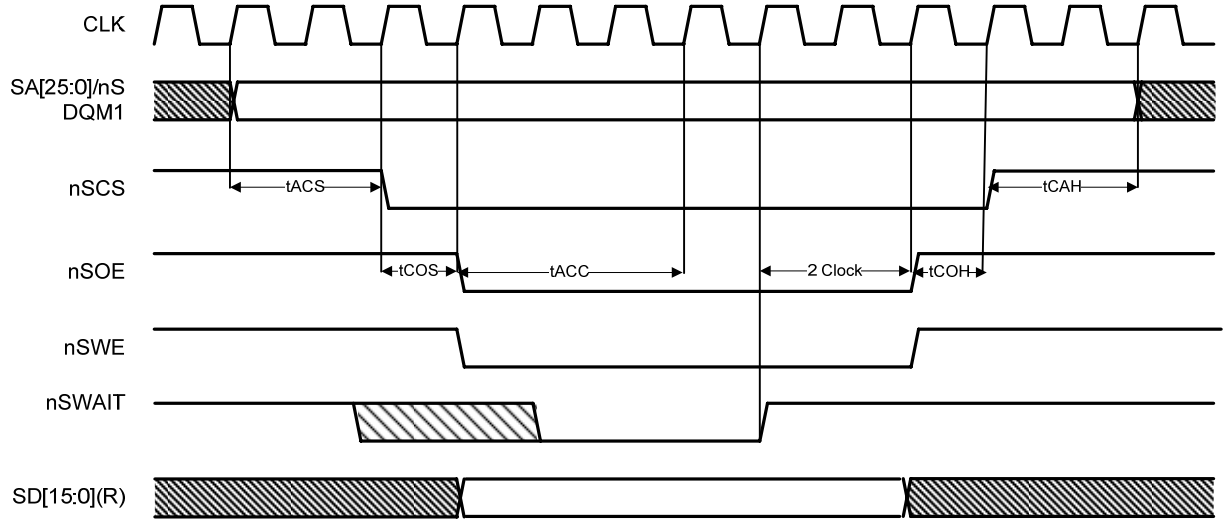


Figure 7-9. Wait Timing (CLK=BCLK(Typ.100MHz))

7.4.2.3. Burst Static Memory Timing

tCOH is not available when there is no next command in the middle of reading and writing operations, or at the end of reading and writing operations.

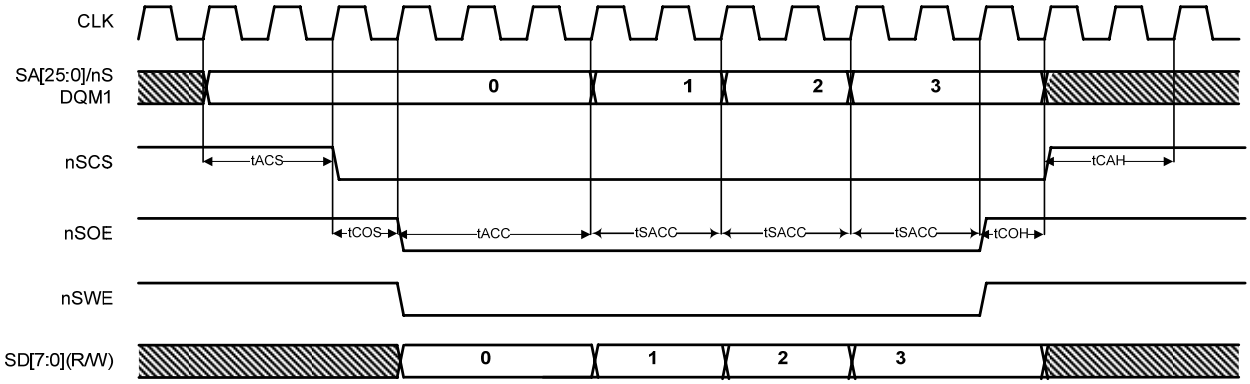


Figure 7-10. Burst 4 Timing (CLK=BCLK(Typ.100MHz))

7.4.2.4. NAND Flash Write Timing

Nand flash memory tACS/tCAH is applied on the internal signal (Internal_nNCS). Outgoing signal nNCS[1:0] stays Low by NFBANK.(Outgoing nNCS is not shown in the following figure.)

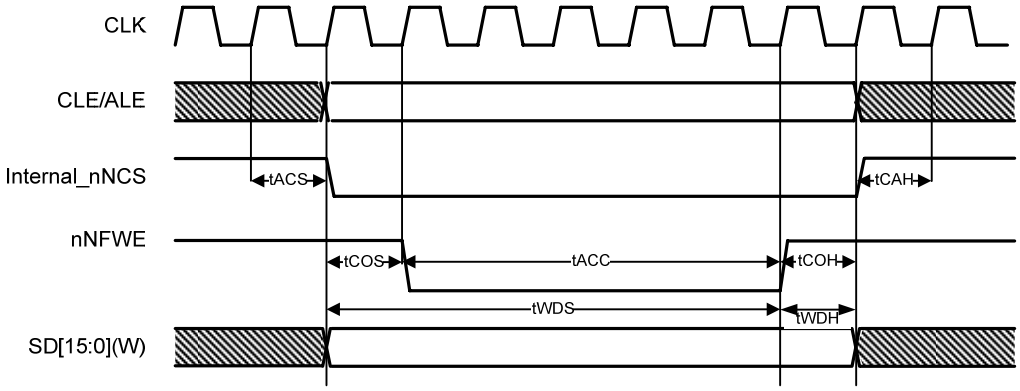


Figure 7-11. NAND Flash Write Timing (CLK=BCLK(Typ.100MHz))

7.4.2.5. NAND Flash Read Timing

If wait control corresponding to each static memory bank is enabled. The Access duration should be prolonged by the external RnB (Ready&Busy) pin while the memory bank is active

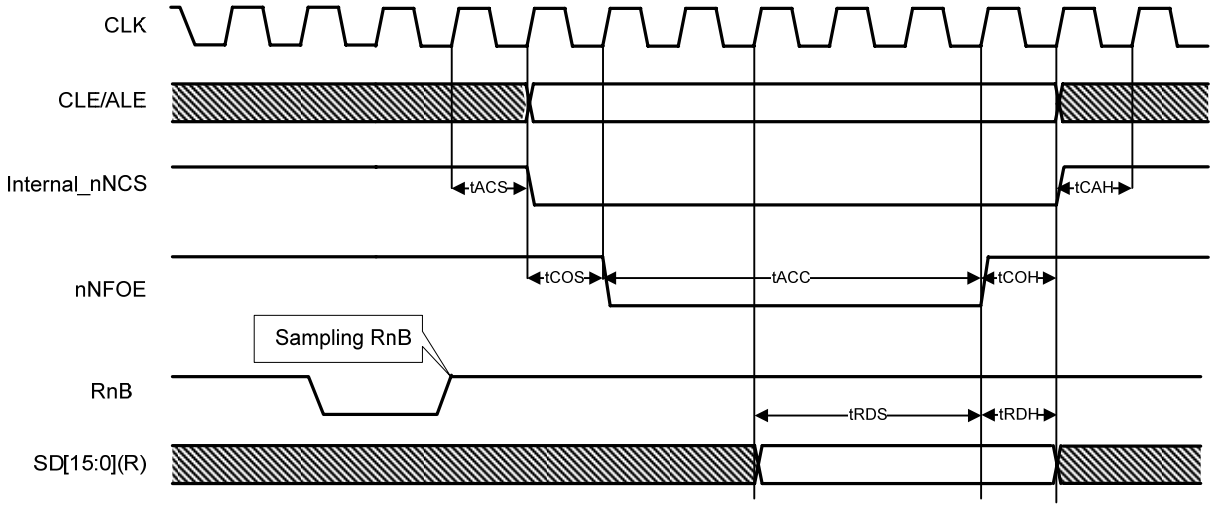


Figure 7-12. NAND Flash Read Flash Timing (CLK=BCLK(Typ.100MHz))

7.4.2.6. Static Memory Timing Diagram Symbol Description

Symbol	min	max	Unit	Description
tCK	7.5	10	ns	Clock cycle
tACS	-3.29	tCK *3 +0.29	ns	Address Setup time before nCS is asserted
tCAH	1.25	tCK *3 +3.29	ns	Address hold time after nCS is not asserted
tCOS	-0.64	tCK *3 +1.45	ns	Chip select setup time before nOE/nWE is asserted
tACC	tCK	tCK *16	ns	Access Cycle
tCOH	0.2	tCK *3 +1.45	ns	Chip Select hold time after nOE/nWE is not asserted
tWDS	tCOS+tACC– 0.64	tCOS+tACC+2.93	ns	Write data setup time
tWDH	tCOH+0.9	tCOH+ 2.93	ns	Write data hold time
tRDS	tCK + 4.53	-	ns	Read data setup time
tRDH	0	-	ns	Read data hold time

Table 7-6. Symbol Description

7.5. Interface

7.5.1.1. 16bit Data bus width DDR-SDRAM Interface of MCU-Y Bank

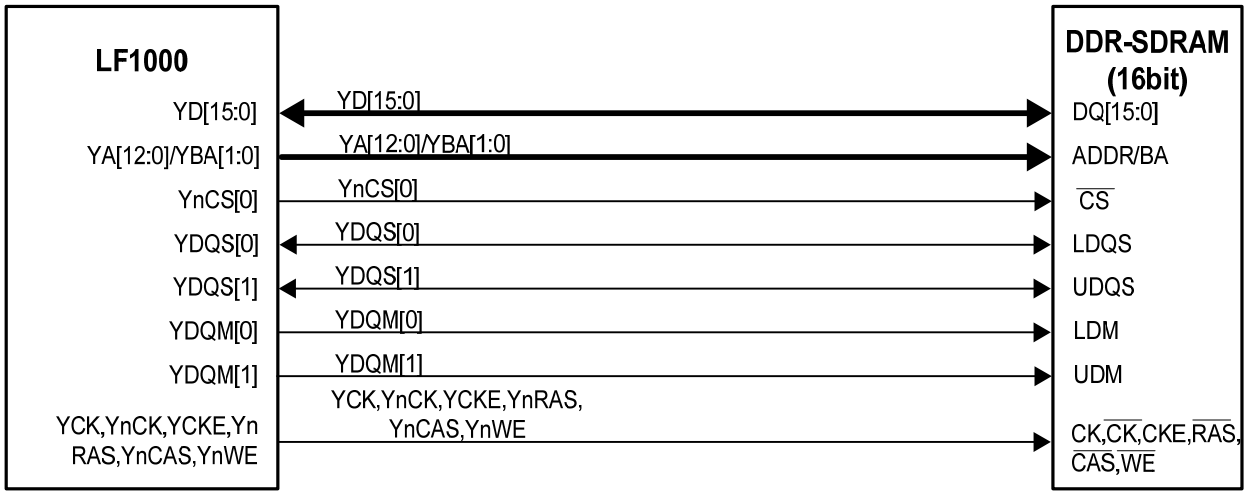


Figure 7-13. 16bit Data bus width DDR-SDRAM Interface

7.5.1.2. 8bit Data bus width DDR-SDRAM Interface of MCU-Y Bank

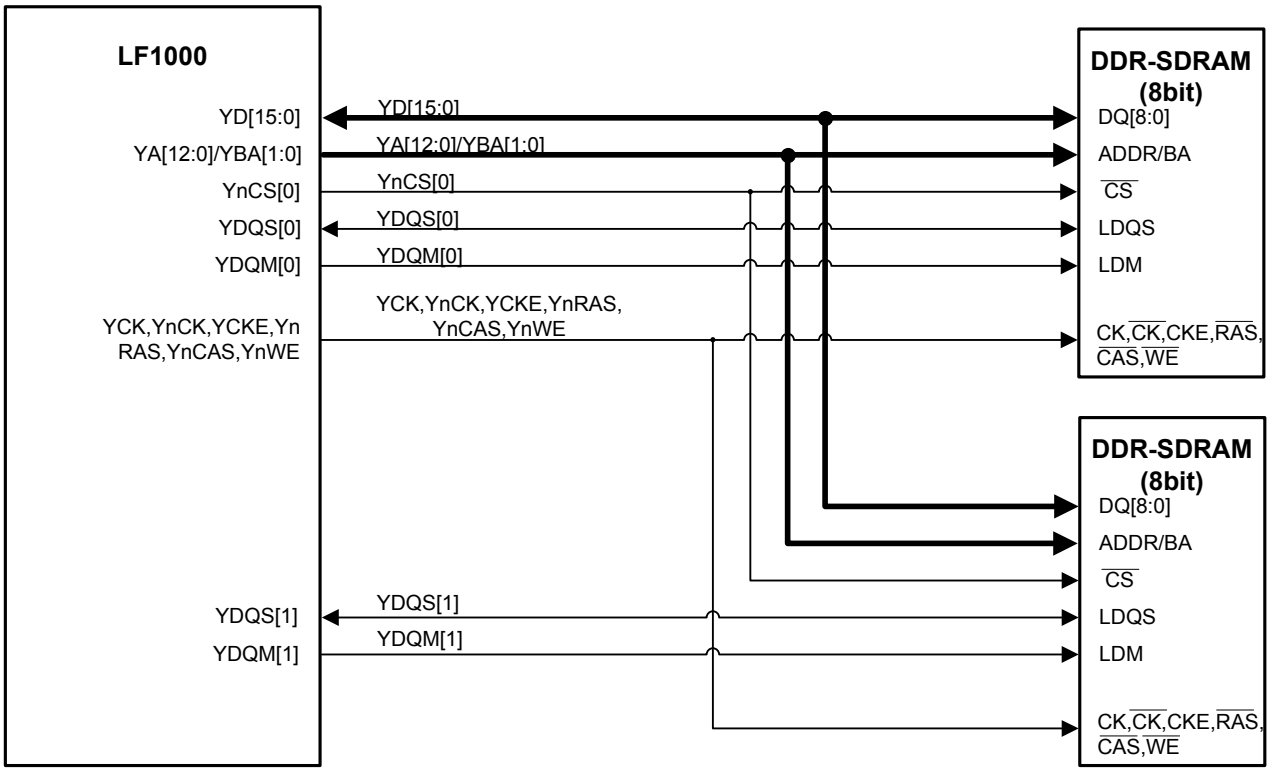


Figure 7-14. 16bit Data bus width DDR-SDRAM Interface

7.6. NAND Boot Overview

POLLUX Supports the Booting function with NAND Flash. The following figure shows the data path of NAND booting.

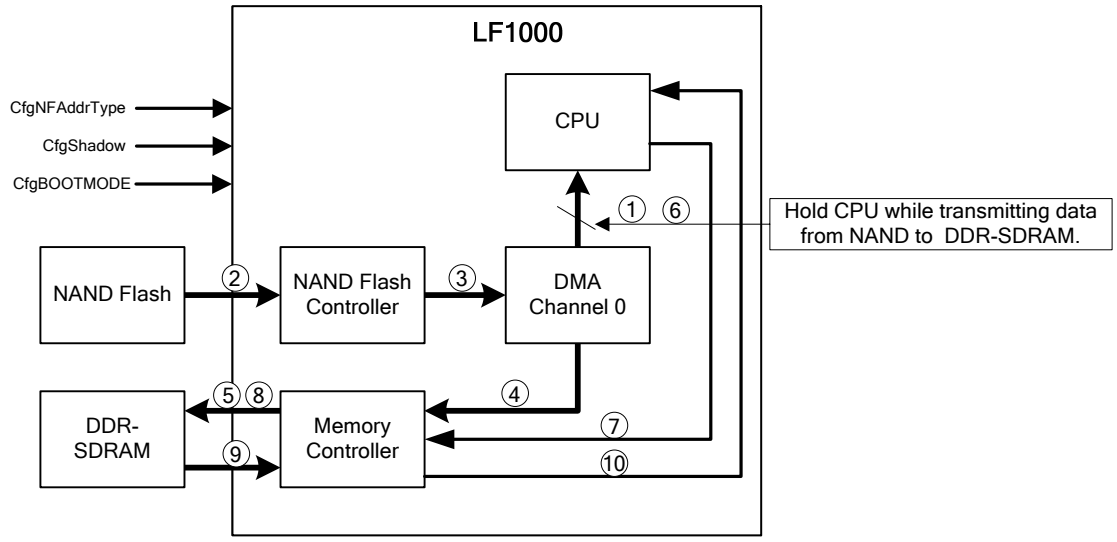


Figure 7-15. NAND Boot

7.6.1. NAND Boot Features

- NAND Flash bus width : Only support for 8-bit bus width NAND flash
- Contain Hardware ECC Block
- Support Small Block NAND Booting & Large Block NAND Booting

7.6.2. NAND Boot Sequence

- In case CfgBOOTMODE is '1', DMA prohibits main memory access as DMA holds CPU.
- NAND Flash Controller which reads the data of external NAND Flash saves at internal buffer.
- NAND Flash requests DMA transmission to DMA Channel 0
- DMA transmits to main memory
- Execute 512Byte or 2048Byte repeatedly according to CfgBOOTENB
- If the transmission is completed, make CPU become normal operation as lifting CPU hold.

7.6.3. Normal Access Sequence

7.6.3.1. Read Cycle

- Write NAND Flash Command at ***NFCMD*** Register.
- Write Address to NFADDR Register with respect to NAND Flash Address Type. (Refer to NAND Flash data book)
- Check ***IRQPEND*** bit.
- Read ECC Data from NAND Flash Spare Array and Write ***ORGECC*** Register.
- Read Data(512byte) from NAND Flash Main Array.
- Check ***NFDecECCDone*** Register
- Check ***NFCheckError*** Register

7.6.3.2. Write Cycle

- Write NAND Flash Command to ***NFCMD*** Register. (Refer to NAND Flash data book)
- Access address of memory which tries to access to ***NFADDR*** Register by 3~5 time according to the sort of NAND Flash. (Refer to NAND Flash data book)
- Write data(512byte) through ***NFDATA*** Register.
- Check ***NFEncECCDone*** Register.
- Read the result of ECC through ***NFECC*** Register. (Small block only)
- Write NAND Flash Command to ***NFCMD*** Register. (Refer to NAND Flash data book)
- Check whether NAND Flash is ready or not by reading ***NFCONTROL.IRQPEND*** bit. (For the exact sequence of NAND Flash, Please refer to NAND Flash data book)

7.6.4. ECC (BCH)

7.6.4.1. Feature

Hardware ECC generation, detection and indication (software correction)

4bit error correction and detection

- Error Detection Code/Error Correction Code(EDC/ECC)

  NAND-based MLC flash has error weakness therefore error handling method is required. POLLUX can perform EDC(error detecting codes) and 4bit ECC(error correction codes) and both based on BCH(Bose-Chadhuri-Hocquenghem) algorithm. EDC is run by hardware to reduce CPU overload and increase the CPU performance. In constrast, ECC is run by software. Parity bit is calculated on every 512 byte page. Syndrome is calculated when each data is read from NAND flash and the value is used in error correction operation.

Hardware ECC generation reset

- This reset is asserted when NAND address or command register is written by any value.
- This reset is also asserted when ECCRST bit (NFCONTROL register [11] bit) register is set to '1'
- This reset initializes NFECCL, NFECCH, NFCNT, NFECCSTATUS, NFSYNDRONE31/75 Registers.

7.7. MCU-Y Memory Control Register

Bit	R/W	Symbol	Description	Reset Value
MEMORY CONFIGURATION REGISTER(MEMCFG) ***Address : MCU-Y : C001_4800h***				
[15]	-	**RESERVED**	Reserved	1'b0
[14:13]	-	**RESERVED**	Reserved	2'b00
[12:10]	-	**RESERVED**	Reserved	3'b000
[9]	R/W	**DS**	Output Driver Impedance Control (Normal DDR only). The normal drive strength for all output are specified to be SSTL2, ClassII. Only when MODESET is set as '1', the written DS setting is applied. 0: Normal 1: Weak	1'b0
[8]	R/W	**DISDLL**	Disable DLL (Data Link Layer). DLL enable is required during power-up initialization and upon returning to normal operation after having disabled the DLL for the purpose of debug or evaluation. Only when MODESET is set as '1', the written DS setting is applied. 0: Enable 1: Disable	1'b0
[7:6]	R/W	**SDRTYPE**	Set DDR-SDRAM type 00: Not Support 01: DDR-SDRAM 10: Not Support 11: Not Support	CfgSDRTYPE
[5:4]	R/W	**SDRBUSBW**	Set data bit width of bank 00 : Reserved 01 : Not Support 10 : 16bit 11 : Reserved	CfgSDR-BUSBW
[3:2]	R/W	**SDRBW**	Set DDR-SDRAM data bus width 00 : Reserved 01 : 8bit 10 : 16bit 11 : Reserved	CfgSDRBW
[1:0]	R/W	**SDRCAP**	Set DDR-SDRAM capacity 00 : 64Mbit 01 : 128Mbit 10 : 256Mbit 11 : 512Mbit	CfgSDRCAP
MEMORY TIMING #0 REGISTER (MEMTIME0) ***Address : MCU-Y : C001_4802h***				
[15:12]	R/W	**TMRD**	Set mode register cycle time (0~15). The unit is a memory clock	4'b0010
[11:8]	-	**RESERVED**	Reserved. This value has no affect.	4'b1111
[7:4]	R/W	**TRP**	Set Low precharge time (0~15). The unit is a memory clock	4'b0100
[3:0]	R/W	**TRCD**	Set RAS to CAS delay (0~15). The unit is a memory clock	4'b0100
MEMORY TIMING #1 REGISTER (MEMTIME1) ***Address : MCU-Y : C001_4804h***				
[15]	R/W	**MODESET**	Read : Read the old register value before the register is set.. 0: complete 1: busy Write: If MODESET is set as '1', a new register value is set to the Mode register and the Extended Mode register of DDR-SDRAM. After the register setting, the value of MODESET is changed into '0' automatically. (Refer to each DDR datasheet).	1'b0

Bit	R/W	Symbol	Description	Reset Value
			0: No affect 1: Apply	
[14]	-	**RESERVED**	Reserved	1'b0
[13:12]	R/W	**LAT**	CAS latency. Although a CAS latency value is written to this bit, the CAS latency is not applied immediately. Only when MODESET is set as '1', the written CAS latency is applied. 00: Reserved 01: 2 cycle 10: 2.5 cycle 11: 3 cycle	CfgSDRLAT
[11:8]	R/W	**TRC**	Auto Refresh to RAS Delay(0~15). The unit is memory clock TRC = tRC – 2	4'b1010
[7:4]	R/W	**TRAS**	RAS active time (0~15). The unit is a memory clock TRAS = tRAS - 2	4'b0111
[3:0]	R/W	**TWR**	Write recovery time (0~15). The unit is a memory clock TWR = tWR	4'b0010

RESERVED
Address : MCU-Y : C001_4806h

MEMORY REFRESH CONTROL REGISTER(MEMREFRESH)
Address : MCU-Y : C001_4808h

[15:0]	-	**REFPERD**	Set refresh period (0~255). Actual refresh period = (REFRESH + 2) * BLCK * 2, disable = 0xFFFF	16'h0100

MEMORY CONTROL REGISTER(MEMCONTROL)
Address : MCU-Y : C001_480Ah

[15:10]	-	**RESERVED**	Reserved	6'h0
[9:8]	R	**UPLAT**	Read the current Read Latency value. 00: Reserved 01: 2 cycle 10: 2.5 cycle 11: 3 cycle	CfgSDR READLAT
[7:6]	R/W	**RDLAT**	Write a desired Read latency value. Although Read latency values are written to this bit, the value is not applied immediately. Only when MODESET is set as '1', the written value is applied. If the MODESET value is automatically changed into '0', the RDLAT value is updated as UPLAT. 00: Reserved 01: 2 cycle 10: 2.5 cycle 11: 3 cycle	CfgSDR READLAT
[5:3]	-	**RESERVED**	Reserved	3'b100
[2]	-	**RESERVED**	Reserved . However, always set to '0'.	1'b0
[1]	-	**RESERVED**	Reserved	1'b1
[0]	R/W	**BLOCKDISP**	Display Array Enable. When the BLOCKDISP is disabled, all the Array areas of the MCU-Y banks are only used only as Linear Array areas. 0: Disable 1: Enable	1'b1

MEMORY CLOCK OUTPUT DELAY REGISTER (MEMCLKDELAY)
Address : MCU-Y : C001_480Ch

[15:4]	-	**RESERVED**	Reserved	12'h0
[3]	-	**RESERVED**	Reserved. However, MEMCLKDELAY[3] should be always be '0'.	1'b0
[2:0]	R/W	**DLYCK**	Adjust the clock delay of MCU-Y. Although a value is written to this bit, the value is not reflected to the operation. The value is applied only after bit15 of the PWRMODE register in the Clock & Power management is set as '1'. (Refer to Ch4. Clock & Power management). 0 : 0ns 1 : 0.5ns 2 : 1.0ns 3 : 1.5ns 4 : 2.0ns 5 : 2.5ns 6 : 3.0ns 7 : 3.5ns Read : Read the current clock delay value. Write : Write the clock delay value to be applied after PLL change.	3'b010

MEMORY DQS OUTPUT DELAY REGISTER (MEMDQSOUTDELAY)
Address : MCU-Y : C001_480Eh

[15:8]	-	**RESERVED**	Reserved for future use. This Value has no affect at current system.	8'h22
[7]	-	**RESERVED**	Reserved. However, MEMDQSOUTDELAY[7] should be always be '0'.	1'b0
[6:4]	R/W	**DLYOUTDQS1**	Adjust the DQS1 output delay of MCU-Y/B. Although a value is written to this bit, the value is not reflected to the operation. The value is applied only after bit15 of the PWRMODE register in the Clock & Power management is set as '1'. (Refer to Ch4. Clock & Power management)	3'b010

Bit	R/W	Symbol	Description	Reset Value
			0 : 0ns 1 : 0.5ns 2 : 1.0ns 3 : 1.5ns 4 : 2.0ns 5 : 2.5ns 6 : 3.0ns 7 : 3.5ns Read : Read the current output DQS1 delay value. Write : Write the output DQS1 delay value to be applied after PLL change.	
[3]	-	**RESERVED**	Reserved. However, MEMDQSOUTDELAY[3] should be always be '0'.	1'b0
[2:0]	R/W	**DLYOUTDQS0**	Adjust the DQS0 output delay of MCU-Y/B. Although a value is written to this bit, the value is not reflected to the operation. The value is applied only after bit15 of the PWRMODE register in the Clock & Power management is set as '1'. (Refer to Ch4. Clock & Power management) 0 : 0ns 1 : 0.5ns 2 : 1.0ns 3 : 1.5ns 4 : 2.0ns 5 : 2.5ns 6 : 3.0ns 7 : 3.5ns Read : Read the current output DQS0 delay value. Write : Write the output DQS0 delay value to be applied after PLL change.	3'b010
MEMORY DQS INPUT DELAY REGISTER (MEMDQSINDELAY) ***Address : MCU-Y : C001_4810h***				
[15:8]	-	**RESERVED**	Reserved for future use. This Value has no affect at current system.	8'h22
[7]	-	**RESERVED**	Reserved. However, MEMDQSINDELAY[7] should be always be '0'.	1'b0
[6:4]	R/W	**DLYINDQS1**	Adjust the DSQ1 input delay of MCU-Y. Although a value is written to this bit, the value is not reflected to the operation. The value is applied only after bit15 of the PWRMODE register in the Clock & Power management is set as '1'. (Refer to Ch4. Clock & Power management.) 0 : 0ns 1 : 0.5ns 2 : 1.0ns 3 : 1.5ns 4 : 2.0ns 5 : 2.5ns 6 : 3.0ns 7 : 3.5ns Read : Read the current output DQS1 delay value. Write : Write the output DQS1 delay value to be applied after PLL change.	3'b010
[3]	-	**RESERVED**	Reserved. However, MEMDQSINDELAY[3] should be always be '0'.	1'b0
[2:0]	R/W	**DLYINDQS0**	Adjust the DSQ0 input delay of MCU-Y/B. Although a value is written to this bit, the value is not reflected to the operation. The value is applied only after bit15 of the PWRMODE register in the Clock & Power management is set as '1'. (Refer to Ch4. Clock & Power management). 0 : 0ns 1 : 0.5ns 2 : 1.0ns 3 : 1.5ns 4 : 2.0ns 5 : 2.5ns 6 : 3.0ns 7 : 3.5ns Read : Read the current output DQS0 delay value. Write : Write the output DQS1 delay value to be applied after PLL change.	3'b010

7.8. MCU-S Memory Control Register

Bit	R/W	Symbol	Description	Reset Value
MEMORY BUS WIDTH REGISTER (MEMBW) ***Address : C001_5800h***				
[31:10]	-	**RESERVED**	Reserved	22'h0
[9]	R/W	**SR9BW**	Set data bus width of static #9 0: Byte (8bit) 1: Half-word (16bit)	1'b1
[8]	R/W	**SR8BW**	Set data bus width of static #8 0: Byte (8bit) 1: Half-word (16bit)	1'b1
[7]	R/W	**SR7BW**	Set data bus width of static #7 0: Byte (8bit) 1: Half-word (16bit)	1'b1
[6]	R/W	**SR6BW**	Set data bus width of static #6 0: Byte (8bit) 1: Half-word (16bit)	1'b1
[5]	R/W	**SR5BW**	Set data bus width of static #5 0: Byte (8bit) 1: Half-word (16bit)	1'b1
[4]	R/W	**SR4BW**	Set data bus width of static #4 0: Byte (8bit) 1: Half-word (16bit)	1'b1
[3]	R/W	**SR3BW**	Set data bus width of static #3 0: Byte (8bit) 1: Half-word (16bit)	1'b1

Bit	R/W	Symbol	Description	Reset Value
[2]	R/W	**SR2BW**	Set data bus width of static #2 0: Byte (8bit) 1: Half-word (16bit)	1'b1
[1]	R/W	**SR1BW**	Set data bus width of static #1 0: Byte (8bit) 1: Half-word (16bit)	1'b0
[0]	R/W	**SR0BW**	Set data bus width of static #0 0: Byte (8bit) 1: Half-word (16bit)	CfgSTBUSWidth
MEMORY TIMING FOR TACS REGISTER (MEMTIMEACS) ***Address : C001_5804h***				
[31:24]	-	**RESERVED**	Reserved	8h'0
[23:22]	R/W	**TACS11**	tACS of NAND Flash Memory 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[21:20]	R/W	**RESERVED**	Reserved	2'b00
[19:18]	R/W	**TACS9**	tACS of static #9 (Unit : BCLK) 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[17:16]	R/W	**TACS8**	tACS of static #8 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[15:14]	R/W	**TACS7**	tACS of static #7 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[13:12]	R/W	**TACS6**	tACS of static #6 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[11:10]	R/W	**TACS5**	tACS of static #5 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[9:8]	R/W	**TACS4**	tACS of static #4 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[7:6]	R/W	**TACS3**	tACS of static #3 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[5:4]	R/W	**TACS2**	tACS of static #2 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[3:2]	R/W	**TACS1**	tACS of static #1 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b01
[1:0]	R/W	**TACS0**	tACS of static #0 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b01
MEMORY TIMING FOR TCOS REGISTER (MEMTIMECOS) ***Address : C001_5808h***				
[31:24]	-	**RESERVED**	Reserved	8h'0
[23:22]	R/W	**TCOS11**	tCOS of NAND Flash Memory 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[21:20]	R/W	**RESERVED**	Reserved	2'b00
[19:18]	R/W	**TCOS9**	tCOS of static #9 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[17:16]	R/W	**TCOS8**	tCOS of static #8 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[15:14]	R/W	**TCOS7**	tCOS of static #7 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00

Bit	R/W	Symbol	Description	Reset Value
[13:12]	R/W	**TCOS6**	tCOS of static #6 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[11:10]	R/W	**TCOS5**	tCOS of static #5 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[9:8]	R/W	**TCOS4**	tCOS of static #4 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[7:6]	R/W	**TCOS3**	tCOS of static #3 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[5:4]	R/W	**TCOS2**	tCOS of static #2 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[3:2]	R/W	**TCOS1**	tCOS of static #1 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b01
[1:0]	R/W	**TCOS0**	tCOS of static #0 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b01
MEMORY TIMING FOR TACC LOW REGISTER (MEMTIMEACCL) ***Address : C001_580Ch***				
[31:28]	R/W	**TACC7**	tACC (Access cycle) of static #7 tACC = (n + 1) cycle	4'b0000
[27:24]	R/W	**TACC6**	tACC of static #6	4'b0000
[23:20]	R/W	**TACC5**	tACC of static #5	4'b0000
[19:16]	R/W	**TACC4**	tACC of static #4	4'b0000
[15:12]	R/W	**TACC3**	tACC of static #3	4'b0000
[11:8]	R/W	**TACC2**	tACC of static #2	4'b0000
[7:4]	R/W	**TACC1**	tACC of static #1	4'b0100
[3:0]	R/W	**TACC0**	tACC of static #0	4'b1111
MEMORY TIMING FOR TACC HIGH REGISTER (MEMTIMEACCH) ***Address : C001_5810h***				
[31:16]	R	**RESERVED**	Reserved	-
[15:12]	R/W	**TACC11**	tACC of NAND Flash	4'b0100
[11:8]	R	**RESERVED**	Reserved	4'b0000
[7:4]	R/W	**TACC9**	tACC of static #9	4'b0000
[3:0]	R/W	**TACC8**	tACC of static #8	4'b0000
MEMORY TIMING FOR TSACC LOW REGISTER (MEMTIMESACCL) ***Address : C001_5814h***				
[31:28]	R/W	**TSACC7**	tSACC of static #7	4'b0000
[27:24]	R/W	**TSACC6**	tSACC of static #6	4'b0000
[23:20]	R/W	**TSACC5**	tSACC of static #5	4'b0000
[19:16]	R/W	**TSACC4**	tSACC of static #4	4'b0000
[15:12]	R/W	**TSACC3**	tSACC of static #3	4'b0000
[11:8]	R/W	**TSACC2**	tSACC of static #2	4'b0000
[7:4]	R/W	**TSACC1**	tSACC of static #1	4'b0011
[3:0]	R/W	**TSACC0**	tSACC of static #0	4'b0000

Bit	R/W	Symbol	Description	Reset Value
MEMORY TIMING FOR TSACC HIGH REGISTER (MEMTIMESACCH) ***Address : C001_5818h***				
[31:8]	R/W	**RESERVED**	Reserved	-
[7:4]	R/W	**TSACC9**	tSACC of static #1	4'b0000
[3:0]	R/W	**TSACC8**	tSACC of static #0	4'b0000
RESERVED ***Address : C001_581Ch ~ C001_5820h***				
MEMORY TIMING FOR TCOH LOW REGISTER (MEMTIMECOH) ***Address : C001_5824h***				
[31:24]	-	**RESERVED**	Reserved	8h'0
[23:22]	R/W	**TCOH11**	tCOH of NAND Flash Memory 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[21:20]	R/W	**RESERVED**	Reserved	2'b00
[19:18]	R/W	**TCOH9**	tCOH of static #9 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[17:16]	R/W	**TCOH8**	tCOH of static #8 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[15:14]	R/W	**TCOH7**	tCOH of static #7 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[13:12]	R/W	**TCOH6**	tCOH of static #6 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[11:10]	R/W	**TCOH5**	tCOH of static #5 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[9:8]	R/W	**TCOH4**	tCOH of static #4 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[7:6]	R/W	**TCOH3**	tCOH of static #3 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[5:4]	R/W	**TCOH2**	tCOH of static #2 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[3:2]	R/W	**TCOH1**	tCOH of static #1 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b01
[1:0]	R/W	**TCOH0**	tCOH of static #0 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b01
MEMORY TIMING FOR TCAH HIGH REGISTER (MEMTIMECAH) ***Address : C001_5828h***				
[31:24]	-	**RESERVED**	Reserved	8h'0
[23:22]	R/W	**TCAH11**	tCAH of NAND Flash Memory 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[21:20]	R/W	**RESERVED**	Reserved	2'b00
[19:18]	R/W	**TCAH9**	tCAH of static #9 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[17:16]	R/W	**TCAH8**	tCAH of static #8 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[15:14]	R/W	**TCAH7**	tCAH of static #7	2'b00

Bit	R/W	Symbol	Description	Reset Value
			00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	
[13:12]	R/W	**TCAH6**	tCAH of static #6 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[11:10]	R/W	**TCAH5**	tCAH of static #5 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[9:8]	R/W	**TCAH4**	tCAH of static #4 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[7:6]	R/W	**TCAH3**	tCAH of static #3 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[5:4]	R/W	**TCAH2**	tCAH of static #2 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b00
[3:2]	R/W	**TCAH1**	tCAH of static #1 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b01
[1:0]	R/W	**TCAH0**	tCAH of static #0 00: 1 cycle 01: 2 cycle 10: 3 cycle 11: 0 cycle	2'b01
MEMORY BURST CONTROL LOW REGISTER (MEMBURSTL) ***Address : C001_582Ch***				
[31:30]	R/W	**BWRITE7**	Write Access Control of static #7 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[29:28]	R/W	**BREAD7**	Read Access Control of static #7 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[27:26]	R/W	**BWRITE6**	Write Access Control of static #6 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[25:24]	R/W	**BREAD6**	Read Access Control of static #6 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[23:22]	R/W	**BWRITE5**	Write Access Control of static #5 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[21:20]	R/W	**BREAD5**	Read Access Control of static #5 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[19:18]	R/W	**BWRITE4**	Write Access Control of static #4 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[17:16]	R/W	**BREAD4**	Read Access Control of static #4 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[15:14]	R/W	**BWRITE3**	Write Access Control of static #3 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[13:12]	R/W	**BREAD3**	Read Access Control of static #3 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00

Bit	R/W	Symbol	Description	Reset Value
[11:10]	R/W	**BWRITE2**	Write Access Control of static #2 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b01
[9:8]	R/W	**BREAD2**	Read Access Control of static #2 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b01
[7:6]	R/W	**BWRITE1**	Write Access Control of static #1 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[5:4]	R/W	**BREAD1**	Read Access Control of static #1 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b01
[3:2]	R/W	**BWRITE0**	Write Access Control of static #0 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[1:0]	R/W	**BREAD0**	Read Access Control of static #0 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
MEMORY BURST CONTROL HIGH REGISTER (MEMBURSTH) ***Address : C001_5830h***				
[31:8]	R/W	**RESERVED**	Reserved	24'hx
[7:6]	R/W	**BWRITE9**	Write Access Control of static #9 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[5:4]	R/W	**BREAD9**	Read Access Control of static #9 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[3:2]	R/W	**BWRITE8**	Write Access Control of static #8 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
[1:0]	R/W	**BREAD8**	Read Access Control of static #8 00: Disable 01: 4 byte burst Access 10: 8 byte burst Access 11: 16 byte burst Access	2'b00
MEMORY WAIT CONTROL REGISTER (MEMWAIT) ***Address : C001_5834h***				
[31:20]	-	**RESERVED**	Reserved	12'h0
[19]	R/W	**WAITENB9**	Wait Enable control of static #9 0: Disable Wait Control 1: Enable Wait Control	1'b0
[18]	R/W	**WAITPOL9**	Wait Polarity control of static #9 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0
[17]	R/W	**WAITENB8**	Wait Enable control of static #8 0: Disable Wait Control 1: Enable Wait Control	1'b0
[16]	R/W	**WAITPOL8**	Wait Polarity control of static #8 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high.	1'b0

Bit	R/W	Symbol	Description	Reset Value
			0: High active wait signal 1: Low active wait signal	
[15]	R/W	**WAITENB7**	Wait Enable control of static #7 0: Disable Wait Control 1: Enable Wait Control	1'b0
[14]	R/W	**WAITPOL7**	Wait Polarity control of static #7 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0
[13]	R/W	**WAITENB6**	Wait Enable control of static #6 0: Disable Wait Control 1: Enable Wait Control	1'b0
[12]	R/W	**WAITPOL6**	Wait Polarity control of static #6 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0
[11]	R/W	**WAITENB5**	Wait Enable control of static #5 0: Disable Wait Control 1: Enable Wait Control	1'b0
[10]	R/W	**WAITPOL5**	Wait Polarity control of static #5 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0
[9]	R/W	**WAITENB4**	Wait Enable control of static #4 0: Disable Wait Control 1: Enable Wait Control	1'b0
[8]	R/W	**WAITPOL4**	Wait Polarity control of static #4 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0
[7]	R/W	**WAITENB3**	Wait Enable control of static #3 0: Disable Wait Control 1: Enable Wait Control	1'b0
[6]	R/W	**WAITPOL3**	Wait Polarity control of static #3 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0
[5]	R/W	**WAITENB2**	Wait Enable control of static #2 0: Disable Wait Control 1: Enable Wait Control	1'b0
[4]	R/W	**WAITPOL2**	Wait Polarity control of static #2 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0
[3]	R/W	**WAITENB1**	Wait Enable control of static #1 0: Disable Wait Control 1: Enable Wait Control	1'b0
[2]	R/W	**WAITPOL1**	Wait Polarity control of static #1 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0
[1]	R/W	**WAITENB0**	Wait Enable control of static #0 0: Disable Wait Control 1: Enable Wait Control	1'b0
[0]	R/W	**WAITPOL0**	Wait Polarity control of static #0 If it is set to low active wait signal, the wait is enabled when the input signal is low. If it is set to high active wait signal, the wait is enabled when the input signal is high. 0: High active wait signal 1: Low active wait signal	1'b0

Bit	R/W	Symbol	Description	Reset Value
RESERVED *Address : C001_5838h ~ C001_5870h*				
NAND FLASH CONTROL REGISTER (NFCONTROL) *Address : C001_5874h*				
[31:16]	-	**RESERVED**	Reserved	16'h0
[15]	R/W	**IRQPEND**	Interrupt pending bit of RnB signal detect. Read : 0 : Not pended 1 : Pended Write : 0 : No affect 1 : Clear	1'bx
[14:12]	-	**RESERVED**	Reserved	3'h0
[11]	W	**ECCRST**	HW ECC block reset NFECCL, NFECCH, NFCNT, NFECCSTATUS, NFSYNDRONE31/75 Registers Reset	1'b0
[10]	-	**RESERVED**	Reserved	1'bx
[9]	R	**RNB**	Ready/Busy check of NAND Flash operation. 0: Busy 1 : Ready	1'bx
[8]	R/W	**IRQENB**	Set interrupt enable/disable at the rising edge of RnB signal of NAND Flash. 0 : Disable 1 : Enable	1'b0
[7:6]	-	**RESERVED**	Reserved	2'b00
[5]	R/W	**NFBOOTENB**	Set NAND Flash Booting On/Off. 0 : Disable 1 : Enable	CfgBOOTMODE
[4:3]	R/W	**NFTYPE**	Set NAND Flash Type for NAND Booting. 00 : Small block 3 address NAND 01 : Small block 4 address NAND 10 : Large block 4 address NAND 11 : Large block 5 address NAND	CfgNFType
[2:1]	-	**RESERVED**	Reserved	2'b00
[0]	R/W	**NFBANK**	Set NAND Flash bank for access. This bit determines which one will be selected out of nNCS[1:0]. Selected nNCS applied after NFBANK is changed and static memory is accessed. 0 : nNCS[0] 1 : nNCS[1]	1'b0
NAND FLASH ECC LOW REGISTER (NFECCL) *Address : C001_5878h*				
[31:0]	R	**ECCL**	Represents 512 byte ECC parity code during Write operation by H/W ECC generation block	32'h0
NAND FLASH ECC HEIGHT REGISTER (NFECCH) *Address : C001_587Ch*				
[31:24]	R	**RESERVED**	Reserved	8'h0
[23:0]	R	**ECCH**	Represents 512 byte ECC parity code during Write operation by H/W ECC generation block.	24'h0
NAND FLASH ORIGIN ECC LOW REGISTER (NFORGECCL) *Address : C001_5880h*				
[31:0]	R/W	**ORGECCL**	When NAND Read operates, firstly read Original ECC Data already saved in the NAND Spare area and then wrties in Register.	32'h0
NAND FLASH ORIGIN ECC HEIGHT REGISTER (NFORGECCH) *Address : C001_5884h*				
[31:24]	R	**RESERVED**	Reserved	8'h0
[23:0]	R/W	**ORGECCH**	When NAND Read operates, firstly read Original ECC Data already saved in the NAND Spare area and then wrties in Register.	24'h0
NAND FLASH DATA COUNT REGISTER (NFCNT) *Address : C001_5888h*				

Bit	R/W	Symbol	Description	Reset Value
[31:26]	R	**RESERVED**	Reserved	6'h0
[25:16]	R	**NFWRCNT**	NAND Flash Write Data Count	10'h0
[15:10]	R	**RESERVED**	Reserved	6'h0
[9:0]	R	**NFRDCNT**	NAND Flash Read Data Count	10'hx
NAND FLASH ECC STATUS REGISTER (NFECCSTATUS) ***Address : C001_588Ch***				
[31:3]	R	**RESERVED**	Reserved	29'h0
[2]	R	**NFCHECKERROR**	When completing NAND Read operating, error check on Read Data. If Read Data Error occurs, NFCHECKERROR is set as '1'. Then NAND Address/Command wirites, the register is cleared. 0 : No Error 1 :Data Error	1'bx
[1]	R	**NFECCDECDONE**	When reading NAND Data with 512 byte plus 51 Cycles (BCLK), NFECCDECDONE is set as '1'. When NAND Address/Command wirites, it's cleared. 0 : IDLE or RUN 1 : End	1'bx
[0]	R	**NFECCENCDONE**	After writing NAND DATA with 512 byte, NFECCENCDONE is set as '1'. When reading ECC Register Data, it's cleared. 0 : IDLE or RUN 1 : End	1'b0
NAND FLASH ECC SYNDROME VALUE31 REGISTER (NFSYNDRONE31) ***Address : C001_5890h***				
[31:26]	R	**RESERVED**	Reserved	6'b0
[25:13]	R	**SYNDROM3**	ECC Decoder Result Odd Syndrome Data3	13'hx
[12:0]	R	**SYNDROM1**	ECC Decoder Result Odd Syndrome Data1	13'hx
NAND FLASH ECC SYNCDROME VALUE75 REGISTER (NFSYNDRONE75) ***Address : C001_5894h***				
[31:26]	R	**RESERVED**	Reserved	6'b0
[25:13]	R	**SYNCDROM7**	ECC Decoder Result Odd Syndrome Data7	13'hx
[12:0]	R	**SYNCDROM5**	ECC Decoder Result Odd Syndrome Data5	13'hx
NAND FLASH DATA REGISTER (NFDATA) ***Address : Shadow 0 : 2C00_0000h/ Shadow 1 : AC00_0000h***				
[31:0]	R/W	**NFDATA**	Nand flash data register. In case of 16 bit access on this register, it generates 8 bit access cycle in twice automatically. In case of 32 bit access on this register, it also generates four 8 bit access cycles automatically.	32'hx
NAND FLASH COMMAND REGISTER (NFCMD) ***Address : Shadow 0 : 2C00_0010h/ Shadow 1 : AC00_0010h***				
[15:8]	-	**RESERVED**	Reserved	8'hx
[7:0]	W	**NFCMD**	Nand flash command register. Writing on this register generates a command cycle with CLE signal and transfers this value on data bus automatically. You have to write only 8 bit data on this register. Do not access this register with 16/32 bit data.	8'hx
NAND FLASH ADDRESS REGISTER (NFADDR) ***Address : Shadow 0 : 2C00_0018h/ Shadow 1 : AC00_0018h***				
[15:8]	-	**RESERVED**	Reserved	8'hx
[7:0]	W	**NFADDR**	Nand flash address register. Writing on this register generates an address cycle with ALE signal and transfers this value on data bus automatically. You have to write only 8 bit data on this register. Do not access this register with 16/32 bit access. Only byte access is avaliable.	8'hx

CHAPTER 8.

DIRECT MEMORY ACCESS CONTROLLER (DMA)

8. DIRECT MEMORY ACCESS CONTROLLER (DMA)

8.1. Overview

The POLLUX supports a 8-channel DMA located between the System Bus and the I/O Bus. Each DMA controller freely exchanges data between devices in the system bus and the I/O Bus. The DMA transfers data in three cases:

- When both the source and the destination are in the System Bus.
- When the source is in the System Bus and the destination is in the I/O Bus.
- When the source is in the I/O Bus and the destination is in the System Bus.

The main feature of the DMA is that it can transfer data without the intervention of the CPU. The DMA can be initialized with software and is available on request from internal peripherals or external request pins.

8.1.1. Features

- 8 Channel DMA
- Memory to Memory Transfer
- Memory to I/O Transfer
- I/O to Memory Transfer
- Command Buffer Mode

8.1.2. Block Diagram

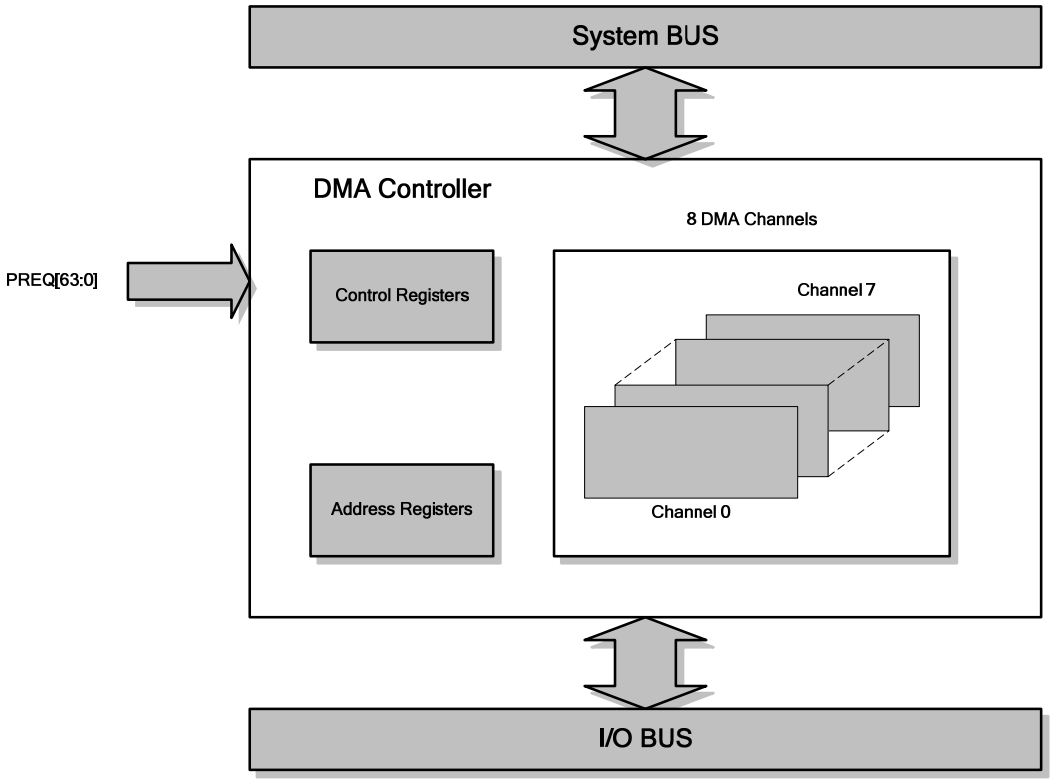


Figure 8-1. DMA Controller Block Diagram

8.2. Operation

The POLLUX has a 8-channel DMA and each channel can be set in three ways: Memory to Memory, I/O to Memory and Memory to I/O. In addition, the POLLUX has 64 DMA Request IDs so that it can select IDs to each peripheral. In default, the DMA provides the communication function between the memories or memory and an I/O device without the intervention of the CPU. The function reduces the CPU load and enhances the data transfer rate. During the DMA data transfer, the CPU does not have the control over the System Bus. During this time, the DMA controller has the control authority for the bus and controls the communication between memory and an I/O device. This operation is generated by the signal like PREQ. If PREQ signal is input, the DMA controller requests the bus control to CPU. Then, CPU terminates the current operation and hands over the bus control to the DMA controller.

The DMA controller of the POLLUX has an Address register group and a Control register group. The Address group can specify source/destination addresses and the Control register group can specify transfer data length, DMA request ID and DMA mode. (For detailed information on these registers, refer to Register Summary attached to this chapter.)

First, the DMA controller assigns a desired DMA channel and specifies a source address and a destination address. In addition, it assigns the whole data length to be transferred in the unit of bytes. The lengths of each address and data depend on the I/O devices. After these settings, DMA Run is performed by setting the ***DMAMODE.RUN*** bit as '1'.

8.2.1. Memory to Memory

Source/ Destination Addresses should be 64-bit aligned.

- Burst Mode : Burst Mode Support at all times
- Source : 64bit Read, Burst Operation
- Destination : 64bit Write, Burst Operation

8.2.2. I/O to Memory

The source address should be the base address of the relevant peripheral. The destination address should be aligned depending on I/O size. (Ex : If the I/O size for a device is 16-bit, the destination address should be aligned in 16-bit.)

- Burst Mode : Source Single Transfer, Destination Single Transfer
- Source : 8/ 16/ 32bit I/O Read, Single Operation. No Burst
- Destination : Same bit width as Source I/O Bus width, No Burst

8.2.3. Memory to I/O

The destination address should be the base address of the relevant peripheral and the source address should be dependant on I/O size. (Ex: If the I/O Size for a device is 16-bit, the source address should be aligned in 16-bit.)

- Burst Mode : Source Burst Transfer, Destination Single Transfer
- Source : 64bit Read, Burst Operation. Start Address Alignment Depending on I/O Width
- Destination : 8/ 16/ 32bit I/O Write, Single Operation, No Burst

8.3. Command Buffer Mode

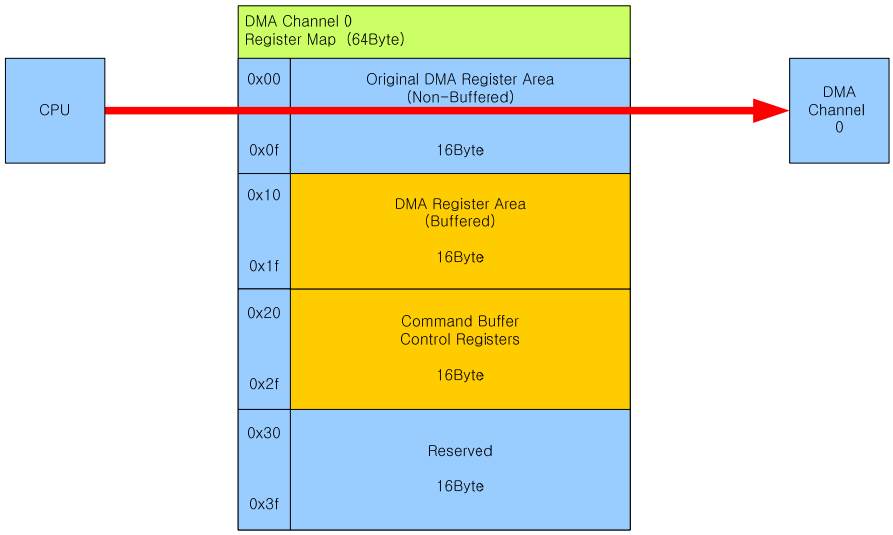
In the previous implementation, next DMA-transfer operation must wait until current DMA-transfer operation is ended. This inefficiency is remove by introducing `Command Buffer Mode`. In the command buffer mode, next DMA-trasfer operation is buffered onto command FIFO(first-in-first-out). When currnet DMA-transfer is ended, waiting commands in the FIFO are written to DMA-register.

8.3.1. Write Through Mode (Original Mode)

Follwing picture shows register address map of DMA channel number 0. Area is divided by four as follows.

- 1st area is original register area of DMA channel 0. (offset 0x00~0x0f)
- 2nd area is buffered register area of DMA channel 0. (offset 0x10~0x1f)
- 3rd area is control registers of command buffer. (offset 0x20~0x2f)
- 4th area is reserved.

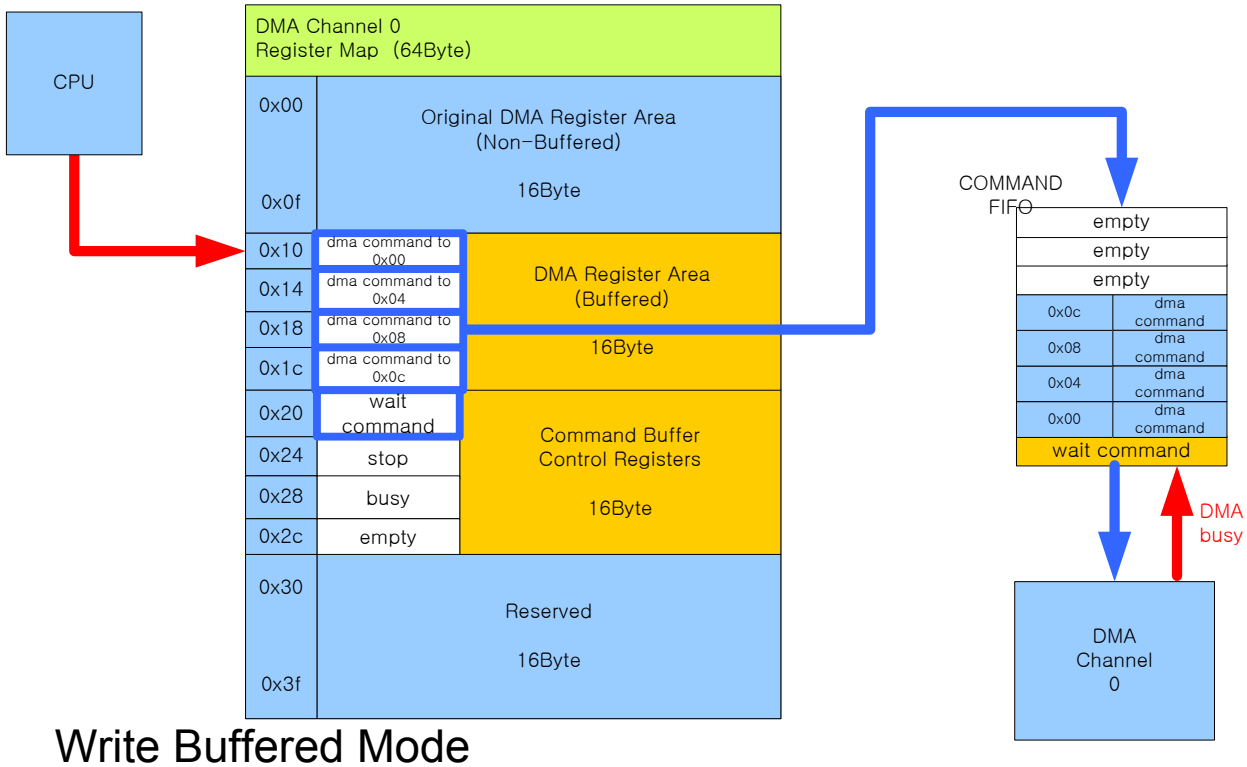
CPU enables 'Write Through Mode' just by writing DMA-register between offset-0x00 and offset-0x0f. This is the same manner as writing registers of DMA channel directly.



Write Through Mode
(CPU accesses original DMA register area)

8.3.2. Write Buffered Mode

Follwing picture shows register address map of DMA channel number 0. CPU enables 'Write Buffered Mode' just by writing buffered DMA register area between offset-0x10 and offset-0x1c. Original DMA register address between 0x00 and 0x0c is mirrored to between 0x10 and 0x1c. Do not forget to write `wait-command` to offset 0x20 before write `buffered-DMA-command`. Without `wait-command`, all commands in the FIFO would be written into registers of DMA channel. `wait-command` hold COMMAND-FIFO until DMA channel is busy. One must be careful buffered DMA register area is write-only area. If one needs to read DMA register, one must read it from non-buffered area. Be careful not to buffer command reading. Buffered command reading is not supported.



Write Buffered Mode
(CPU accesses Buffered DMA register area)

8.3.3. DMA Controller Base Address

Channel	DMA Base Address	Function
0	C000_0000h	DMA Channel 0
1	C000_0080h	DMA Channel 1
2	C000_0100h	DMA Channel 2
3	C000_0180h	DMA Channel 3
4	C000_0200h	DMA Channel 4
5	C000_0280h	DMA Channel 5
6	C000_0300h	DMA Channel 6
7	C000_0380h	DMA Channel 7

Table 8-1. DMA Controller Base Address

8.3.4. DMA Priority

Group	Channels	Priority (Service Percentage)
0	0, 1, 2, 3	TBD
1	4, 5, 6, 7	TBD

Table 8-2. DMA Channel Priority

8.3.5. Peripheral DMA Request ID

Index	Description	Index	Description
0	UART0 Tx	32	Reserved
1	UART0 Rx	33	Reserved
2	UART1 Tx	34	Reserved
3	UART1 Rx	35	Reserved
4	UART2 Tx	36	Reserved
5	UART2 Rx	37	Reserved
6	UART3 Tx	38	Reserved
7	UART3 Rx	39	Reserved
8	Reserved	40	Reserved
9	Reserved	41	Reserved
10	Reserved	42	Reserved
11	Reserved	43	Reserved
12	USB End Point1 Device	44	Reserved
13	USB End Point2 Device	45	Reserved
14	Reserved	46	Reserved
15	Reserved	47	Reserved
16	SD0 Read/Write	48	Reserved
17	Reserved	49	Reserved
18	SSP/SPI0 Tx	50	Reserved
19	SSP/SPI0 Rx	51	Reserved

20	SSP/SPI1 Tx	52	Reserved
21	SSP/SPI1 Rx	53	Reserved
22	SSP/SPI2 TX	54	Reserved
23	SSP/SPI2 RX	55	Reserved
24	PCM OUT	56	Reserved
25	Reserved	57	Reserved
26	PCM IN	58	Reserved
27	Reserved	59	Reserved
28	Reserved	60	Reserved
29	Reserved	61	Reserved
30	SD1 Read/Write	62	Reserved
31	Reserved	63	Reserved

Table 8-3. Peripheral DMA Request ID

8.4. Register Summary

Bit	R/W	Symbol	Description	Reset Value
DMA SOURCE ADDRESS REGISTER (DMASRCADDR) ***Address : DMA Base Address + 00h : WORD***				
[31 : 0]	R/W	**DMASRCADDR**	Specifies the source address for DMA transfer. - I/O to Memory : The base address of a peripheral should be specified. - Memory to I/O : The source address should be aligned depending on I/O Size. - Memory to Memory : The source address should be 64-bit aligned.	32'bx
DMA DESTINATION ADDRESS REGISTER (DMADSTADDR) ***Address : DMA Base Address + 04h : WORD***				
[31 : 0]	R/W	**DMADSTADDR**	Specifies the destination address for DMA transfer. - Memory to I/O : The base address of the peripheral should be specified. - I/O to Memory : The destination address should be aligned depending on I/O Size. - Memory to Memory : The destination address should be 64-bit aligned.	32'bx
DMA TRANSFER LENGTH REGISTER (DMALENGTH) ***Address : DMA Base Address + 08h : HWORD***				
[15 : 0]	R/W	**DMALENGTH**	Specifies the number of bytes to be transferred. DMALENGTH should specify N-1 of the actual transfer byte numbers. For Memory to Memory, the transfer byte numbers should be specified depending on I/O Size. 0000 ~ FFFF : 1 ~ 64Kbyte transfer	16'b0
DMA REQUEST ID REGISTER (DMAREQID) ***Address : DMA Base Address + 0Ah : HWORD***				
[15 : 6]	-	**RESERVED**	Reserved	10'b0
[5 : 0]	R/W	**DMAREQID**	Specifies the DMA request ID of the peripheral for I/O transfer from among the 64 DMA request pins,	6'b0
DMA OPERATION MODE REGISTER (DMAMODE) ***Address : DMA Base Address + 0Ch : WORD***				
[31]	-	**RESERVED**	Reserved (Test Purpose Only) However, DMAMODE[31] bit should be always be '0'.	1'b0
[30 : 21]	-	**RESERVED**	Reserved	11'b0
[20]	R/W	**STOP**	Compulsion Stop of DMA Run status This bit allows stopping before DMA transfer operation is ended. After this STOP bit is written by '1', one must read RUN bit to decide DMA channel is safely stopped. This bit is not automatically cleared to '0'. Be careful not to forget writting '0' to this bit after DMA is stopped during transfer operation. 0 : RUN 1 : Stop	1'b0
[19]	R/W	**RUN**	Decides the execution of DMA operation. For forced termination during DMA Run, write '0' to this bit. (In DMA RUN mode, the other bits cannot be accessed.) 0 : STOP 1 : RUN	1'b0
[18]	R/W	**INTENB**	Decides the interrupt generation when DMA Run is completed. 0 : Disable 1 : Enable	1'b0
[17]	R/W	**INTPEND**	Shows the DMA operation completion status. Write '1' to clear this bit. (Although INTENB is disabled, this bit always has an effective value.) Read > 0 : None 1: Interrupt Pended Write > 0 : None 1 : Pending Clear	1'b0
[16]	R/W	**BUSY**	Shows the DMA operation status. 0 : IDLE 1 : Busy	1'b0
[15 : 14]	-	**RESERVED**	Reserved	2'b0
[13]	R/W	**DSTNOTREQCHK**	Checks request when DSTIOMODE = 1 0 : Check Request 1 : Not Check Request	1'b0
[12]	R/W	**DSTNOTIMC**	Decides the increase of the destination address. 0 : Increment 1 : Not Increment	1'b0
[11]	-	**RESERVED**	Reserved	1'b0
[10]	R/W	**DESTIMODE**	Sets the destination mode. 0 : Memory 1 : I/O	1'b0
[9 : 8]	R/W	**DSTIOSIZE**	Specifies the I/O size in I/O mode transfer. 00 : Byte 01 : HWORD (16bit)	2'b0

Bit	R/W	Symbol	Description	Reset Value
			10 : WORD(32bit) 11 : Reserved	
[7 : 6]	-	**RESERVED**	Reserved	2'b0
[5]	R/W	**SRCNOTREQCHK**	Does not use the request (Acknowledge) of the relevant device when SRCIOMODE is '1'. Checks request when SRCIOMODE = 1 0 : Check (Req, Ack) 1 : Not Check (PSEL)	1'b0
[4]	R/W	**SRCNOTINC**	Decides the increase of the source address. 0 : Increment 1 : Not Increment	1'b0
[3]	R/W	**PACKMODE**	Reserved	1'b0
[2]	R/W	**SRCIOMODE**	Sets the source mode. 0 : Memory 1 : I/O	1'b0
[1 : 0]	R/W	**SRCIOSIZE**	Specifies the I/O size in I/O mode transfer. 00 : Byte 01 : HWORD (16bit) 10 : WORD (32bit) 11 : Reserved	2'b0
Buffered DMA SOURCE ADDRESS REGISTER (DMASRCADDR) ***Address : DMA Base Address + 10h : WORD***				
[31 : 0]	W	**DMASRCADDRBUF**	This is the buffered area which mirros **DMA SOURCE ADDRESS REGISTER** Be careful this is write-only register.	32'bx
Buffered DMA DESTINATION ADDRESS REGISTER (DMADSTADDR) ***Address : DMA Base Address + 14h : WORD***				
[31 : 0]	W	**DMADSTADDRBUF**	This is the buffered area which mirros **DMA DESTINATION ADDRESS REGISTER** Be careful this is write-only register.	
Buffered DMA TRANSFER LENGTH REGISTER (DMALENGTH) ***Address : DMA Base Address + 18h : HWORD***				
[15 : 0]	W	**DMALENGTHBUF**	This is the buffered area which mirros **TRANSFER LENGTH REGISTER** Be careful this is write-only register.	
Buffered DMA REQUEST ID REGISTER (DMAREQID) ***Address : DMA Base Address + 1Ah : HWORD***				
[15 : 6]	-	**RESERVED**	Reserved	
[5 : 0]	W	**DMAREQIDBUF**	This is the buffered area which mirros **DMA REQUEST ID REGISTER** Be careful this is write-only register.	
Buffered DMA OPERATION MODE REGISTER (DMAMODE) ***Address : DMA Base Address + 1Ch : WORD***				
[31:0]	-	**DMAMODEBUF**	This is the buffered area which mirros **DMA OPERATION MODE REGISTER** Be careful this is write-only register.	
DMA COMMAND WAIT REGISTER (DMACMDWAIT) ***Address : DMA Base Address + 20h : HWORD***				
[31:1]	-	**RESERVED**	Reserved	
[0]	W	**DMACMDWAIT**	**Command Wait** Value '1' must be written to this bit, before wrtting buffered-commands for next DMA operation. This 'CommandWait' holds COMMAND FIFO while current DMA operation is busy. When current DMA is ended, 'CommandWait' is cleared and next commands in the FIFO are written to DMA registers.	
DMA COMMAND STOP REGISTER (DMACMDSTOP) ***Address : DMA Base Address + 24h : HWORD***				
[31:1]	-	**RESERVED**	Reserved	
[0]	W	**DMACMDSTOP**	**Command Stop** 1: All buffered command in the COMMAND FIFO is canceled. 0: None	1
DMA COMMAND BUSY REGISTER (DMACMDBUSY) ***Address : DMA Base Address + 28h : HWORD***				
[31:1]	-	**RESERVED**	Reserved	

Bit	R/W	Symbol	Description	Reset Value
[0]	R	**DMACMDBUSY**	**Command Busy** 1: Commands in the FIFO is waiting for next DMA transfer operation. 0: Command FIFO is empty.	0
DMA COMMAND EMPTY SPACE REGISTER (DMACMDSPACE) ***Address : DMA Base Address + 2Ch : HWORD***				
[31:4]	-	**RESERVED**	Reserved	
[3 : 0]	R	**DMACMDSPACE**	This bit shows how many spaces are empty in the COMMAND-FIFO. Do not write command into FIFO when this bit is zero.	8

CHAPTER 9.

INTERRUPT CONTROLLER

9. INTERRUPT CONTROLLER

9.1. Overview

43 interrupt sources from internal peripherals, including the DMA controller, UART, I^2C and GPIO, etc. supply requests to an Interrupt Controller.

The POLLUX has one interrupt controller used for the 533MHz.

An FIQ or an IRQ (interrupt requests) are signaled by the Interrupt Controller to 533MHz. After arbitrating multiple requests from internal peripherals and GPIO, the Controller requests an interrupt.

The hardware arbitration logic decides the interrupt arbitration process and the results are recorded in the interrupt pending registers.

9.1.1. Block Diagram

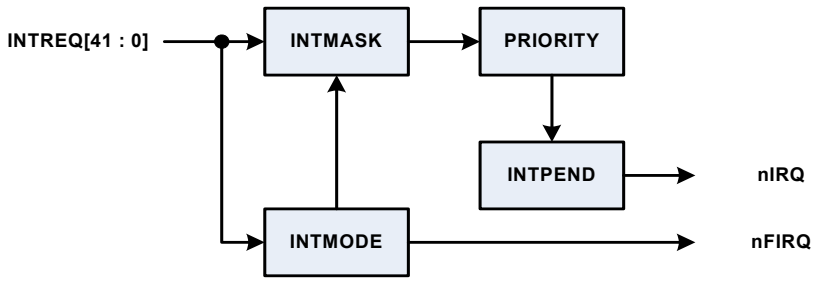


Figure 9-1. Interrupt Process Diagram

9.2. Operation

The interrupt controller consists of five control registers including the interrupt pending register, priority register, mask register, interrupt mode register and source pending register.

The first registration of all the interrupt requests from the interrupt source is done in the source pending register. Based on the interrupt mode register, interrupt requests are divided into two groups: FIQ requests and IRQ requests. Arbitration is performed for multiple IRQ requests according to the priority register.

9.2.1. F-bit and I-bit of PSR (Program Status Register)

A FIQ (fast interrupt request) from the Interrupt Controller is not accepted by the CPU when the F-bit of PSR (program status register in 533MHZ CPU) is set to 1. The IRQ (interrupt request) from the interrupt controller is not accepted by the CPU when the I-bit of PSR (program status register in 533MHZ CPU) is set to 1. Therefore, the PSR F-bit or I-bit must be cleared to 0 and the corresponding bit of INTMASK(***INTMASKL, INTMASKH***) must be set to 0 to enable the interrupt reception.

9.2.2. Interrupt Mode

533MHZ CPU has 2 types of interrupt mode, FIQ or IRQ. All the interrupt sources determine the mode of interrupt to be used at interrupt request.

9.2.3. Interrupt Mask Register

Interrupt Mask Register indicates that an interrupt has been disabled if the corresponding mask bit is '1'. If an interrupt mask bit of ***INTMASK*** is '0', the interrupt will be serviced normally. If the corresponding mask bit is '1' and the interrupt is generated, the source pending bit will be set.

9.2.4. Interrupt Source

Interrupt Number	Source	Description
0	PDISPLAY	Primary Display Interrupt

Interrupt Number	Source	Description
1	SDISPLAY	Secondary Display Interrupt
2	N/A	N/A
3	DMA	DMA Interrupt
4	TIMER0	Timer0 Interrupt
5	SYSCTRL	System Control Interrupt
6	N/A	N/A
7	N/A	N/A
8	N/A	N/A
9	N/A	N/A
10	UART0	UART 0 Interrupt
11	TIMER1	Timer Interrupt 1
12	SSPSPI0	SSP/ SPI0 Interrupt
13	GPIO	GPIO Interrupt
14	SDMMC0	SD/ MMC0 Interrupt
15	TIMER2	Timer 2 Interrupt
16	N/A	N/A
17	N/A	N/A
18	N/A	N/A
19	N/A	N/A
20	UDC	USB Device 2.0 Interrupt
21	TIMER3	Timer 3 Interrupt
22	N/A	N/A
23	N/A	N/A
24	AUDIOIF	Audio Interface Interrupt
25	ADC	ADC Interrupt
26	MCUSTATIC	MCU-S Interrupt
27	GRP3D	3D Graphic Controller Interrupt
28	UHC	USB Host Interrupt
29	N/A	N/A
30	N/A	N/A
31	RTC	RTC Interrupt
32	I^2C0	I^2C0 Interrupt
33	I^2C1	I^2C1 Interrupt
34	UART1	UART 1 Interrupt
35	UART2	UART 2 Interrupt
36	UART3	UART 3 Interrupt
37	N/A	N/A
38	N/A	N/A
39	SSPSPI1	SSP/ SPI 1 Interrupt
40	SSPSPI2	SSP/ SPI 2 Interrupt

Interrupt Number	Source	Description
41	CSC	Color Space Converter
42	SDMMC1	SD/ MMC1 Interrupt
43	TIMER4	Timer 4 Interrupt

Table 9-1 Interrupt Sources Description Table

9.2.5. Interrupt Priority

The POLLUX has 12 Arbiters for selecting the interrupt priority. Among these, each Arbiter from 0 to 10 has six interrupt sources. So users can select the priority suitable for their purpose. In addition, Arbiter11 can determine the priorities after receiving the interrupt requests generated from Arbiter0 to Arbiter10. The priorities are determined by the ***PRIORDER*** registers.

The priorities from Arbiter0 to Arbiter10 are determined as follows:

- The order of priority is subREQ0, subREQ1, subREQ2, subREQ3, subREQ4 and subREQ5 when PRIORDER xx bits are 00b.
- The order of priority is subREQ0, subREQ2, subREQ3, subREQ4, subREQ1 and subREQ5 when PRIORDER xx bits are 01b.
- The order of priority is subREQ0, subREQ3, subREQ4, subREQ1, subREQ2 and subREQ5 when PRIORDER xx bits are 10b.
- The order of priority is subREQ0, subREQ4, subREQ1, subREQ2, subREQ3 and subREQ5 when PRIORDER xx bits are 11b.

Arbiter11 receives a total of eleven interrupt requests and their priorities can be determined as follows:

- The order of priority is REQ0, REQ1, REQ2, REQ3, REQ4, REQ5, REQ6, REQ7, REQ8, REQ9, REQ10 when PRIORDER[24 : 22] bits are 000b.
- The order of priority is REQ0, REQ2, REQ3, REQ4, REQ5, REQ6, REQ7, REQ8, REQ1, REQ9, REQ10 when PRIORDER[24 : 22] bits are 001b.
- The order of priority is REQ0, REQ3, REQ4, REQ5, REQ6, REQ7, REQ8, REQ1, REQ2, REQ9, REQ10 when PRIORDER[24 : 22] bits are 010b.
- The order of priority is REQ0, REQ4, REQ5, REQ6, REQ7, REQ8, REQ1, REQ2, REQ3, REQ9, REQ10 when PRIORDER[24 : 22] bits are 011b.
- The order of priority is REQ0, REQ5, REQ6, REQ7, REQ8, REQ1, REQ2, REQ3, REQ4, REQ9, REQ10 when PRIORDER[24 : 22] bits are 100b.
- The order of priority is REQ0, REQ6, REQ7, REQ8, REQ1, REQ2, REQ3, REQ4, REQ5, REQ9, REQ10 when PRIORDER[24 : 22] bits are 101b.
- The order of priority is REQ0, REQ7, REQ8, REQ1, REQ2, REQ3, REQ4, REQ5, REQ6, REQ9, REQ10 when PRIORDER[24 : 22] bits are 110b.
- The order of priority is REQ0, REQ8, REQ1, REQ2, REQ3, REQ4, REQ5, REQ6, REQ7, REQ9, REQ10 when PRIORDER[24 : 22] bits are 111b.

Figure-9.2 shows the POLLUX interrupt priorities for the above description.

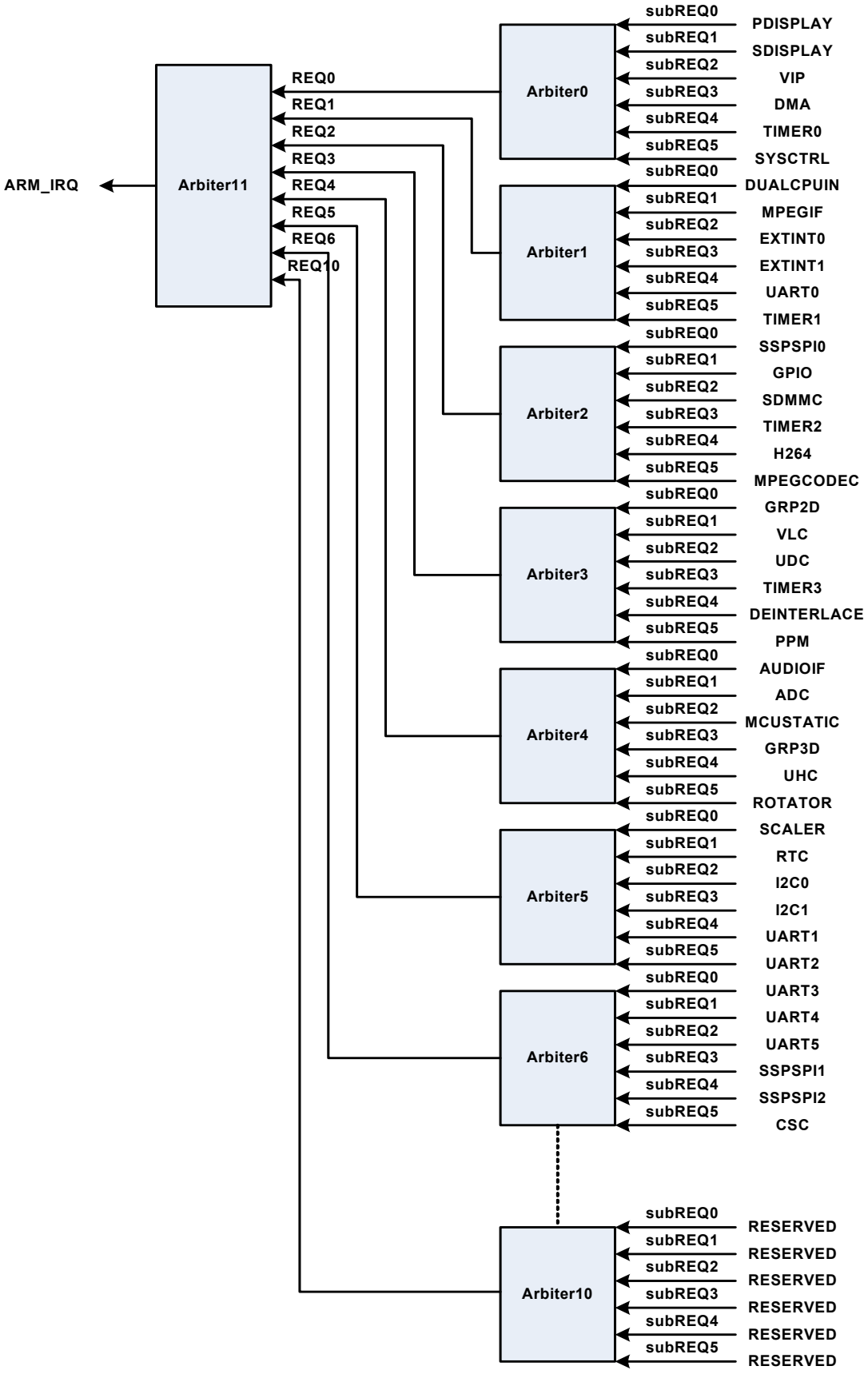


Figure 9-2 Priority Generating Block

9.2.6. Interrupt Mode Register (INTMODE)

There are 43 interrupt source related bits in this register. The corresponding interrupt is processed in the FIQ(Fast Interrupt) mode when a specific is set to '1'. If not, it is processed in the IRQ(Normal Interrupt) mode.

<Note>

INTPEND and ***INTOFFSET*** registers will not be affected by the FIQ interrupt when an interrupt mode is set to FIQ mode in ***INTMODE*** register. The ***INTPEND*** and ***INTOFFSET*** registers are valid only for IRQ mode interrupt source.

In the interrupt controller only one interrupt source can be served in the FIQ mode. (For the urgent interrupt FIQ mode should only be used.) And only one ***INTMODE*** bit can be set to '1'.

9.2.7. Interrupt MASK Register (INTMASK)

An interrupt source is related to each of the 43bits(32bit + 11bit) in the MASK register. CPU does not serve the interrupt request from the corresponding interrupt source when the specific bit is set to '1', even in the case of the corresponding ***SRCPEND*** register bit is set to '1'. The interrupt service is made when the mask bit is '0'

9.2.8. Interrupt Pending Register (INTPEND)

In the interrupt pending register, each of the 43bits(32bit + 11bit) indicates if the corresponding interrupt request is the unmasked highest priority one and waits for the interrupt to be served. Only one bit can be set to '1' as ***INTPEND*** is located after the priority logic, which is the interrupt request producing IRQ to CPU. This register can be read to determine the interrupt source to be served among 42 sources in interrupt service routine for IRQ.

A specific ***INTPEND*** register bit can be cleared by writing a data to this register. Only ***INTPEND*** bit position corresponding to those set to '1' in the data can be cleared. The corresponding bit positions to those that are set to '0' in data remain unchanged.

<Note>

This register is only for the IRQ sources. This register does not show the FIQ request under being served.

9.3. Register Summary

Bit	R/W	Symbol	Description	Reset Value
RESERVED *Address C000_0800h ~ C000_0807h*				
INTERRUPT MODE LOW REGISTER (INTMODEL) *Address : C000_0808h*				
[31]	R/W	**RTC**	0 : IRQ 1 : FIQ	1'b0
[30]	R/W	**RESERVED**	Reserved	1'b0
[29]	R/W	**RESERVED**	Reserved	1'b0
[28]	R/W	**UHC**	0 : IRQ 1 : FIQ	1'b0
[27]	R/W	**GRP3D**	0 : IRQ 1 : FIQ	1'b0
[26]	R/W	**MCUSTATIC**	0 : IRQ 1 : FIQ	1'b0
[25]	R/W	**ADC**	0 : IRQ 1 : FIQ	1'b0
[24]	R/W	**AUDIOIF**	0 : IRQ 1 : FIQ	1'b0
[23]	R/W	**RESERVED**	Reserved	1'b0
[22]	R/W	**RESERVED**	Reserved	1'b0
[21]	R/W	**TIMER3**	0 : IRQ 1 : FIQ	1'b0
[20]	R/W	**UDC**	0 : IRQ 1 : FIQ	1'b0
[19]	R/W	**RESERVED**	Reserved	1'b0
[18]	R/W	**RESERVED**	Reserved	1'b0
[17]	R/W	**RESERVED**	Reserved	1'b0
[16]	R/W	**RESERVED**	Reserved	1'b0
[15]	R/W	**TIMER2**	0 : IRQ 1 : FIQ	1'b0
[14]	R/W	**SDMMC**	0 : IRQ 1 : FIQ	1'b0
[13]	R/W	**GPIO**	0 : IRQ 1 : FIQ	1'b0
[12]	R/W	**SSPSPI0**	0 : IRQ 1 : FIQ	1'b0
[11]	R/W	**TIMER1**	0 : IRQ 1 : FIQ	1'b0
[10]	R/W	**UART0**	0 : IRQ 1 : FIQ	1'b0
[9]	R/W	**RESERVED**	Reserved	1'b0
[8]	R/W	**RESERVED**	Reserved	1'b0
[7]	R/W	**RESERVED**	Reserved	1'b0
[6]	R/W	**RESERVED**	Reserved	1'b0
[5]	R/W	**SYSCTRL**	0 : IRQ 1 : FIQ	1'b0
[4]	R/W	**TIMER0**	0 : IRQ 1 : FIQ	1'b0
[3]	R/W	**DMA**	0 : IRQ 1 : FIQ	1'b0
[2]	R/W	**RESERVED**	Reserved	1'b0
[1]	R/W	**SDISPLAY**	0 : IRQ 1 : FIQ	1'b0
[0]	R/W	**PDISPLAY**	0 : IRQ 1 : FIQ	1'b0
INTERRUPT MODE HIGH REGISTER (INTMODEH) *Address : C000_080Ch*				
[31 : 12]	-	**RESERVED**	Reserved	20'b0

Bit	R/W	Symbol	Description		Reset Value
[11]	R/W	**TIMER4**	0 : IRQ	1 : FIQ	1'b0
[10]	R/W	**SDMMC1**	0 : IRQ	1 : FIQ	1'b0
[9]	R/W	**CSC**	0 : IRQ	1 : FIQ	1'b0
[8]	R/W	**SSPSPI2**	0 : IRQ	1 : FIQ	1'b0
[7]	R/W	**SSPSPI1**	0 : IRQ	1 : FIQ	1'b0
[6]	R/W	**RESERVED**	Reserved		1'b0
[5]	R/W	**RESERVED**	Reserved		1'b0
[4]	R/W	**UART3**	0 : IRQ	1 : FIQ	1'b0
[3]	R/W	**UART2**	0 : IRQ	1 : FIQ	1'b0
[2]	R/W	**UART1**	0 : IRQ	1 : FIQ	1'b0
[1]	R/W	**I2C1**	0 : IRQ	1 : FIQ	1'b0
[0]	R/W	**I2C0**	0 : IRQ	1 : FIQ	1'b0
INTERRUPT MASK LOW REGISTER (INTMASKL) ***Address : C000_0810h***					
[31]	R/W	**RTC**	0 : Serviced	1 : Masked	1'b0
[30]	R/W	**RESERVED**	Reserved		1'b0
[29]	R/W	**RESERVED**	Reserved		1'b0
[28]	R/W	**UHC**	0 : Serviced	1 : Masked	1'b0
[27]	R/W	**GRP3D**	0 : Serviced	1 : Masked	1'b0
[26]	R/W	**MCUSTATIC**	0 : Serviced	1 : Masked	1'b0
[25]	R/W	**ADC**	0 : Serviced	1 : Masked	1'b0
[24]	R/W	**AUDIOIF**	0 : Serviced	1 : Masked	1'b0
[23]	R/W	**RESERVED**	Reserved		1'b0
[22]	R/W	**RESERVED**	Reserved		1'b0
[21]	R/W	**TIMER3**	0 : Serviced	1 : Masked	1'b0
[20]	R/W	**UDC**	0 : Serviced	1 : Masked	1'b0
[19]	R/W	**RESERVED**	Reserved		1'b0
[18]	R/W	**RESERVED**	Reserved		1'b0
[17]	R/W	**RESERVED**	Reserved		1'b0
[16]	R/W	**RESERVED**	Reserved		1'b0
[15]	R/W	**TIMER2**	0 : Serviced	1 : Masked	1'b0
[14]	R/W	**SDMMC**	0 : Serviced	1 : Masked	1'b0
[13]	R/W	**GPIO**	0 : Serviced	1 : Masked	1'b0
[12]	R/W	**SSPSPI0**	0 : Serviced	1 : Masked	1'b0
[11]	R/W	**TIMER1**	0 : Serviced	1 : Masked	1'b0
[10]	R/W	**UART0**	0 : Serviced	1 : Masked	1'b0
[9]	R/W	**RESERVED**	Reserved		1'b0
[8]	R/W	**RESERVED**	Reserved		1'b0
[7]	R/W	**RESERVED**	Reserved		1'b0
[6]	R/W	**RESERVED**	Reserved		1'b0

Bit	R/W	Symbol	Description		Reset Value
[5]	R/W	**SYSCTRL**	0 : Serviced	1 : Masked	1'b0
[4]	R/W	**TIMER0**	0 : Serviced	1 : Masked	1'b0
[3]	R/W	**DMA**	0 : Serviced	1 : Masked	1'b0
[2]	R/W	**RESERVED**	Reserved		1'b0
[1]	R/W	**SDISPLAY**	0 : Serviced	1 : Masked	1'b0
[0]	R/W	**PDISPLAY**	0 : Serviced	1 : Masked	1'b0
INTERRUPT MASK HIGH REGISTER (INTMASKH) ***Address : C000_0814***					
[31 : 11]	-	**RESERVED**	Reserved		21'b0
[10]	R/W	**TIMER4**	0 : Serviced	1 : Masked	1'b0
[9]	R/W	**CSC**	0 : Serviced	1 : Masked	1'b0
[8]	R/W	**RESERVED**	Reserved		1'b0
[7]	R/W	**SSPSPI1**	0 : Serviced	1 : Masked	1'b0
[6]	R/W	**RESERVED**	Reserved		1'b0
[5]	R/W	**RESERVED**	Reserved		1'b0
[4]	R/W	**UART3**	0 : Serviced	1 : Masked	1'b0
[3]	R/W	**UART2**	0 : Serviced	1 : Masked	1'b0
[2]	R/W	**UART1**	0 : Serviced	1 : Masked	1'b0
[1]	R/W	**I2C1**	0 : Serviced	1 : Masked	1'b0
[0]	R/W	**I2C0**	0 : Serviced	1 : Masked	1'b0
PRIORITY ORDER REGISTER (PRIORDER) ***Address : C000_0818h***					
[31 : 25]	-	**RESERVED**	Reserved		7'b0
[24 : 22]	R/W	**ARBSEL11**	Arbiter Group 11 priority order set 000 : REQ 0-1-2-3-4-5-6-7-8-9-10 001 : REQ 0-2-3-4-5-6-7-8-1-9-10 010 : REQ 0-3-4-5-6-7-8-1-2-9-10 011 : REQ 0-4-5-6-7-8-1-2-3-9-10 100 : REQ 0-5-6-7-8-1-2-3-4-9-10 101 : REQ 0-6-7-8-1-2-3-4-5-9-10 110 : REQ 0-7-8-1-2-3-4-5-6-9-10 111 : REQ 0-8-1-2-3-4-5-6-7-9-10		3'b0
[21 : 20]	R/W	**ARBSEL10**	Arbiter Group 10 priority order set 00 : REQ 0-1-2-3-4-5 10 : REQ 0-3-4-1-2-5	01 : REQ 0-2-3-4-1-5 01 : REQ 0-4-1-2-3-5	2'b0
[19 : 18]	R/W	**ARBSEL9**	Arbiter Group 9 priority order set 00 : REQ 0-1-2-3-4-5 10 : REQ 0-3-4-1-2-5	01 : REQ 0-2-3-4-1-5 01 : REQ 0-4-1-2-3-5	2'b0
[17 : 16]	R/W	**ARBSEL8**	Arbiter Group 8 priority order set 00 : REQ 0-1-2-3-4-5 10 : REQ 0-3-4-1-2-5	01 : REQ 0-2-3-4-1-5 01 : REQ 0-4-1-2-3-5	2'b0
[15 : 14]	R/W	**ARBSEL7**	Arbiter Group 7 priority order set 00 : REQ 0-1-2-3-4-5 10 : REQ 0-3-4-1-2-5	01 : REQ 0-2-3-4-1-5 01 : REQ 0-4-1-2-3-5	2'b0
[13 : 12]	R/W	**ARBSEL6**	Arbiter Group 6 priority order set 00 : REQ 0-1-2-3-4-5 10 : REQ 0-3-4-1-2-5	01 : REQ 0-2-3-4-1-5 01 : REQ 0-4-1-2-3-5	2'b0
[11 : 10]	R/W	**ARBSEL5**	Arbiter Group 5 priority order set 00 : REQ 0-1-2-3-4-5 10 : REQ 0-3-4-1-2-5	01 : REQ 0-2-3-4-1-5 01 : REQ 0-4-1-2-3-5	2'b0
[9 : 8]	R/W	**ARBSEL4**	Arbiter Group 4 priority order set 00 : REQ 0-1-2-3-4-5 10 : REQ 0-3-4-1-2-5	01 : REQ 0-2-3-4-1-5 01 : REQ 0-4-1-2-3-5	2'b0

Bit	R/W	Symbol	Description	Reset Value
[7 : 6]	R/W	**ARBSEL3**	Arbiter Group 3 priority order set 00 : REQ 0-1-2-3-4-5 01 : REQ 0-2-3-4-1-5 10 : REQ 0-3-4-1-2-5 01 : REQ 0-4-1-2-3-5	2'b0
[5 : 4]	R/W	**ARBSEL2**	Arbiter Group 2 priority order set 00 : REQ 0-1-2-3-4-5 01 : REQ 0-2-3-4-1-5 10 : REQ 0-3-4-1-2-5 01 : REQ 0-4-1-2-3-5	2'b0
[3 : 2]		**ARBSEL1**	Arbiter Group 1 priority order set 00 : REQ 0-1-2-3-4-5 01 : REQ 0-2-3-4-1-5 10 : REQ 0-3-4-1-2-5 01 : REQ 0-4-1-2-3-5	2'b0
[1 : 0]		**ARBSEL0**	Arbiter Group 0 priority order set 00 : REQ 0-1-2-3-4-5 01 : REQ 0-2-3-4-1-5 10 : REQ 0-3-4-1-2-5 01 : REQ 0-4-1-2-3-5	2'b0
RESERVED ***Address : C000_081Ch***				
[31 : 0]	-	**RESERVED**	Reserved	32'b0
INTERRUPT PENDING LOW REGISTER (INTPENDL) ***Address : C000_0820h***				
[31]	R/W	**RTC**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[30]	R/W	**RESERVED**	Reserved	1'b0
[29]	R/W	**RESERVED**	Reserved	1'b0
[28]	R/W	**UHC**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[27]	R/W	**GRP3D**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[26]	R/W	**MCUSTATIC**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[25]	R/W	**ADC**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[24]	R/W	**AUDIOIF**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[23]	R/W	**RESERVED**	Reserved	1'b0
[22]	R/W	**RESERVED**	Reserved	1'b0
[21]	R/W	**TIMER3**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[20]	R/W	**UDC**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[19]	R/W	**RESERVED**	Reserved	1'b0
[18]	R/W	**RESERVED**	Reserved	1'b0
[17]	R/W	**RESERVED**	Reserved	1'b0
[16]	R/W	**RESERVED**	Reserved	1'b0
[15]	R/W	**TIMER2**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[14]	R/W	**SDMMC**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[13]	R/W	**GPIO**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[12]	R/W	**SSPSPI0**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[11]	R/W	**TIMER1**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[10]	R/W	**UART0**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended Write > 0 : Don't care 1 : Pending Clear	1'b0
[9]	R/W	**RESERVED**	Reserved	1'b0
[8]	R/W	**RESERVED**	Reserved	1'b0
[7]	R/W	**RESERVED**	Reserved	1'b0
[6]	R/W	**RESERVED**	Reserved	1'b0
[5]	R/W	**SYSCTRL**	Read > 0 : Interrupt is not Pended 1 : Interrupt is Pended	1'b0

Bit	R/W	Symbol	Description		Reset Value
			Write > 0 : Don't care	1 : Pending Clear	
[4]	R/W	**TIMER0**	Read > 0 : Interrupt is not Pended Write > 0 : Don't care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[3]	R/W	**DMA**	Read > 0 : Interrupt is not Pended Write > 0 : Don't care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[2]	R/W	**RESERVED**	Reserved		1'b0
[1]	R/W	**SDISPLAY**	Read > 0 : Interrupt is not Pended Write > 0 : Don't care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[0]	R/W	**PDISPLAY**	Read > 0 : Interrupt is not Pended Write > 0 : Don't care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
INTERRUPT PENDING HIGH REGISTER (INTPENDH) ***Address : C000_0824h***					
[31 : 11]	-	**RESERVED**	Reserved		21'b0
[10]	R/W	**TIMER4**	Read > 0 : Interrupt is not Pended Write > 0 : Don't Care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[9]	R/W	**CSC**	Read > 0 : Interrupt is not Pended Write > 0 : Don't Care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[8]	R/W	**RESERVED**	Reserved		1'b0
[7]	R/W	**SSPSPI1**	Read > 0 : Interrupt is not Pended Write > 0 : Don't Care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[6]	R/W	**RESERVED**	Reserved		1'b0
[5]	R/W	**RESERVED**	Reserved		1'b0
[4]	R/W	**UART3**	Read > 0 : Interrupt is not Pended Write > 0 : Don't Care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[3]	R/W	**UART2**	Read > 0 : Interrupt is not Pended Write > 0 : Don't Care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[2]	R/W	**UART1**	Read > 0 : Interrupt is not Pended Write > 0 : Don't Care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[1]	R/W	**I2C1**	Read > 0 : Interrupt is not Pended Write > 0 : Don't Care	1 : Interrupt is Pended 1 : Pending Clear	1'b0
[0]	R/W	**I2C0**	Read > 0 : Interrupt is not Pended Write > 0 : Don't Care	1 : Interrupt is Pended 1 : Pending Clear	1'b0

CHAPTER 10.

TIMER

10. TIMER

10.1. Overview

The POLLUX has five independent timers and each timer has a 32-bit counter. Each timer channel for the POLLUX has a Timer Counter register (***TMRCOUNT***) and the register value automatically increases until the value reaches the value of the Time Match register (***TMRMATCH***). If the values of the two registers become the same, an interrupt (or Watch Dog Reset) occurs and the ***TMRCOUTER*** is automatically cleared down to '0'. The features and the block diagram of each the POLLUX timers are as follows:

10.1.1. Features

- Embedded 5-channel 32-bit counter (one channel is for WatchDog Only)
- WatchDog function

10.1.2. Block Diagram

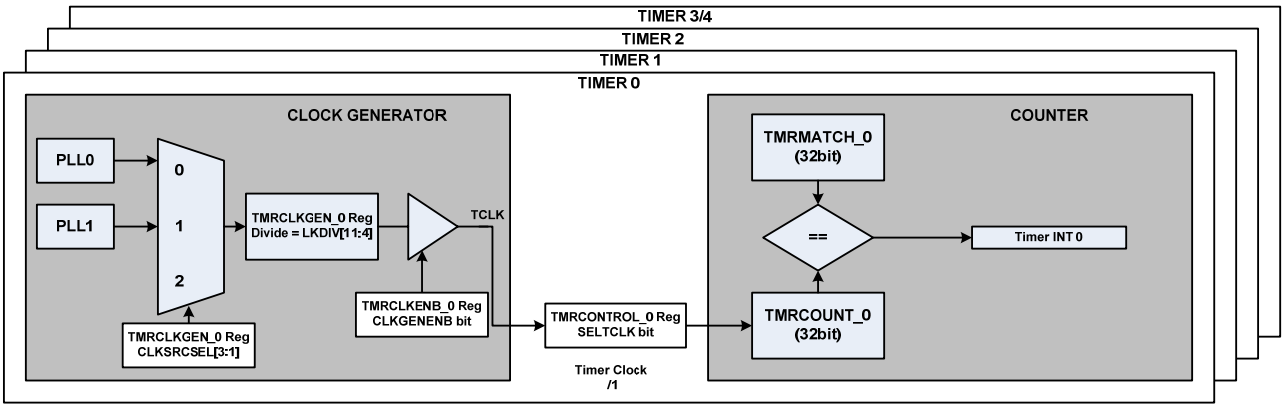


Figure 10-1. Timer Block Diagram

<Note>

In the above figure, 'n' indicates the number of the timer channel and four channels have the same block diagrams.

10.2. Operation

Figure 10-1 shows the overall block diagram of a timer. The POLLUX has four channels and each channel operates as a 32-bit Up Counter. All timers have the same blocks (timer0 to timer4) and also contain the Watch Dog function. In addition, the operation of the timers is determined by using ***TMRCONTROL.WDENB*** bit.

The CPU can change the value of a timer counter at any time, but it may read the wrong value due to setup or hold problems caused by Async when it reads the timer value. To prevent error, the CPU should read timer values using the following procedure:

Step 1) Set ***TMRCONTROL.LDCNT*** bit to '1'

Step 2) Read ***TMRCOUNT*** Register value

The POLLUX timer block basically receives inputs from the 2-PLL and selects one of the three PLLs using the ***TMRCLKGEN.CLKSRCSEL***[3:1]. After that, the block generates the clock for timers using the ***TMRCLKGEN.CLKDIV***[11:4] bit. At this point, the block determines the final clock from the clock generator by using the ***TMRCLKENB.CLKGENENB*** bit. Each of the POLLUX timers has a Divider, Timer Counter register and Timer Match register. All of them can be selected by the Base address. If the POLLUX writes a count value to the Timer Match register and enables a timer, the Time Counter register value automatically increases. If the ***TMRCOUNT*** register value reaches the setting value of the ***TMRMATCH*** register, an interrupt or Watch Dog occurs. If an interrupt occurs in these circumstances, the ***TMRCOUNTER*** value is automatically cleared down to '0'.

10.3. Timer Setting Sequence

- Selection of a Desired Timer (Selection of Timer Base Address)
- PLL Selection, Divide Value Setting

- ***TMRCOUNT***, ***TMRMATCH*** Register Value Setting (Clock Divider Value Setting)
- Interrupt Pending Clear
- Interrupt/ Watch Dog Enable
- Timer Enable
-

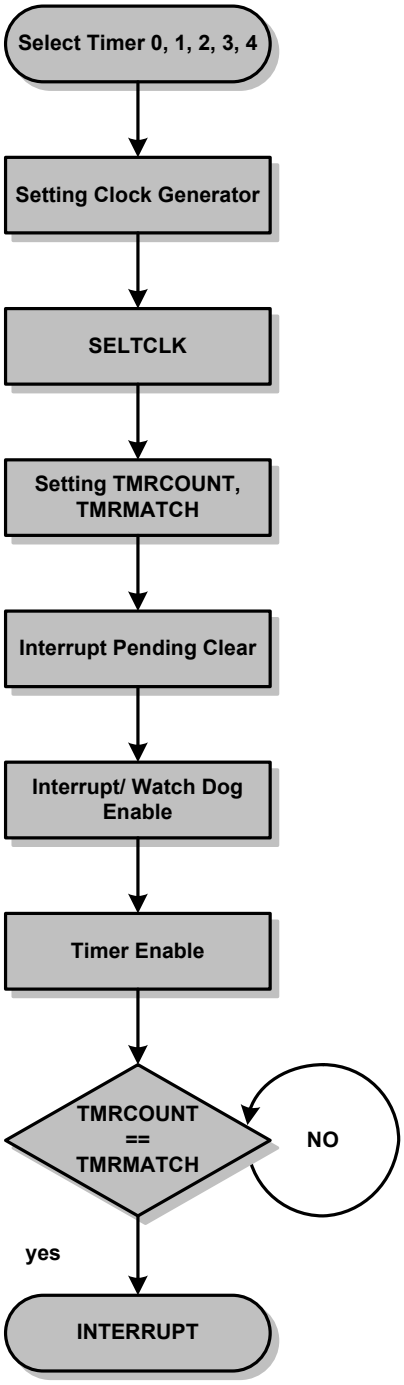


Figure 10-2. Timer Sequence

10.4. Register Summary

Bit	R/W	Symbol	Description	Reset Value
TIMER COUNTER REGISTER (TMRCOUNT) ***Address : TIMER0 : C000_1800h/ TIMER1 : C000_1880h/ TIMER2 : C000_1900h/ TIMER3 : C000_1980h/TIMER4: C000_1A00h***				
[31 : 0]	R/W	**TMRCOUNT**	Register to write the current timer counter value. The value can be upgraded by using an up-counter whose value automatically increases. However, the LDCNT bit should be set as '1' before reading this register.	32'bx
TIMER MATCH REGISTER (TMRMATCH) ***Address : TIMER0 : C000_1804h/ TIMER1 : C000_1884h/ TIMER2 : C000_1904h/ TIMER3 : C000_1984h/ TIMER4: C000_1A04h***				
[31 : 0]	R/W	**TMRMATCH**	Writes the match counter value. If the TMRCOUNT value reaches the value written in this register, an interrupt or Watch Dog Reset occurs.	32'bx
TIMER CONTROL REGISTER (TMRCONTROL) ***Address : TIMER0 : C000_1808h/ TIMER1 : C000_1888h/ TIMER2 : C000_1908h/ TIMER3 : C000_1988h/ TIMER4: C000_1A08h***				
[31 : 7]	-	**RESERVED**	Reserved	25'b0
[6]	W	**LDCNT**	Reads the value of the TMRCOUNT register. 0 : Don't Care 1 : Load a Timer Counter Value	1'b0
[5]	R/W	**INTPEND**	Enables the Interrupt Pending status to be checked and the value cleared. Read > 0 : Not Pended 1 : Interrupt Pended Write> 0 : Don't Care 1 : Pending Clear	1'b0
[4]	R/W	**INTENB**	Determines the generation of an interrupt when the TMRCOUNT value reaches the TMRMATCH value. 0 : Interrupt Disable 1 : Interrupt Enable	1'b0
[3]	R/W	**RUN**	Operates the timer. 0 : Timer STOP 1 : Timer RUN	1'b0
[2]	R/W	**WDENB**	Operates the Watch Dog function. 0 : Normal Timer Operation 1 : Watch Dog Operation	1'b0
[1 : 0]	R/W	**SELTCLK**	Can further divide the clock generated in the Timer Clock Generator by 1, 2, 4 and 8. 00 : Timer Clock / 2 01 : Timer Clock / 4 10 : Timer Clock/ 8 11 : Timer Clock	2'b0
RESERVED ***Address : TIMER0 : C000_180Ch/ TIMER1 : C000_188Ch/ TIMER2 : C000_190Ch/ TIMER3 : C000_198Ch/ TIMER4: C000_1A0Ch*** ***~ TIMER0 : C000_183Fh/ TIMER1 : C000_18BFh/ TIMER2 : C000_193Fh/ TIMER3 : C000_19BFh / TIMER4: C000_1A3Fh***				
TIMER CLOCK GENERATION ENABLE REGISTER (TMRCLKENB) ***Address : TIMER0 : C000_1840h/ TIMER1 : C000_18C0h/ TIMER2 : C000_1940h/ TIMER3 : C000_19C0h / TIMER4: C000_1A40h***				
[31 : 4]	-	**RESERVED**	Reserved	28'b0
[3]	R/W	**TCLKMODE**	Determines TCLK operation mode 0 : TCLK is enabled only in CPU access. 1 : Always.	1'b0
[2]	R/W	**CLKGENENB**	Enables/Disables the operation of the Clock Generation. Since the Timer registers have the structure to be written after synchronizing with TCLK, write is not allowed for the Timer registers when TCLK is not supplied. 0 : Disable 1 : Enable	1'b0
[1 : 0]	-	**RESERVED**	Reserved	2'b0
TIMER CLOCK GENERATION CONTROL REGISTER (TMRCLKGEN) ***Address : TIMER0 : C000_1844h/ TIMER1 : C000_18C4h/ TIMER2 : C000_1944h/ TIMER3 : C000_19C4h / TIMER4: C000_1A44h***				
[31 : 12]	-	**RESERVED**	Reserved	20'b0
[11 : 4]	R/W	**CLKDIV**	Clock Divider Value. For 'N' clock divide, enter a value of 'N-1' 00h ~ FFh(N-1) : 1 to 256 clock divide (N clock divide)	8'b0

Bit	R/W	Symbol	Description	Reset Value
			(Always only 1 or even number available for divide value N) It is recommended that CLKDIV be set as '2'.	
[3 : 1]	R/W	**CLKSRCSEL**	Clock Source Selection 000 : PLL0 001 : PLL1 010 : Reserved 011 ~ 111 : Reserved	3'b0
[0]	-	**RESERVED**	Reserved	1'b0

<Note>

- The TCLK output should be slower than 50 MHz.

CHAPTER 11.

REAL TIME CLOCK (RTC)

11. REAL TIME CLOCK (RTC)

11.1. Overview

The Real Time Clock (RTC) block can be operated by the Backup Battery while the system power is off. The RTC block is composed of 32bit free counter register and works with an external 32.768KHz Crystal and also can perform the alarm function.

11.1.1. Features

- 32bit Counter
- Alarm Function : Alarm Interrupt or Wake Up from Power Down Mode
- Independent power pin (VDD_RTC)
- Support 1Hz Time interrupt for Power Down Mode

11.1.2. Block Diagram

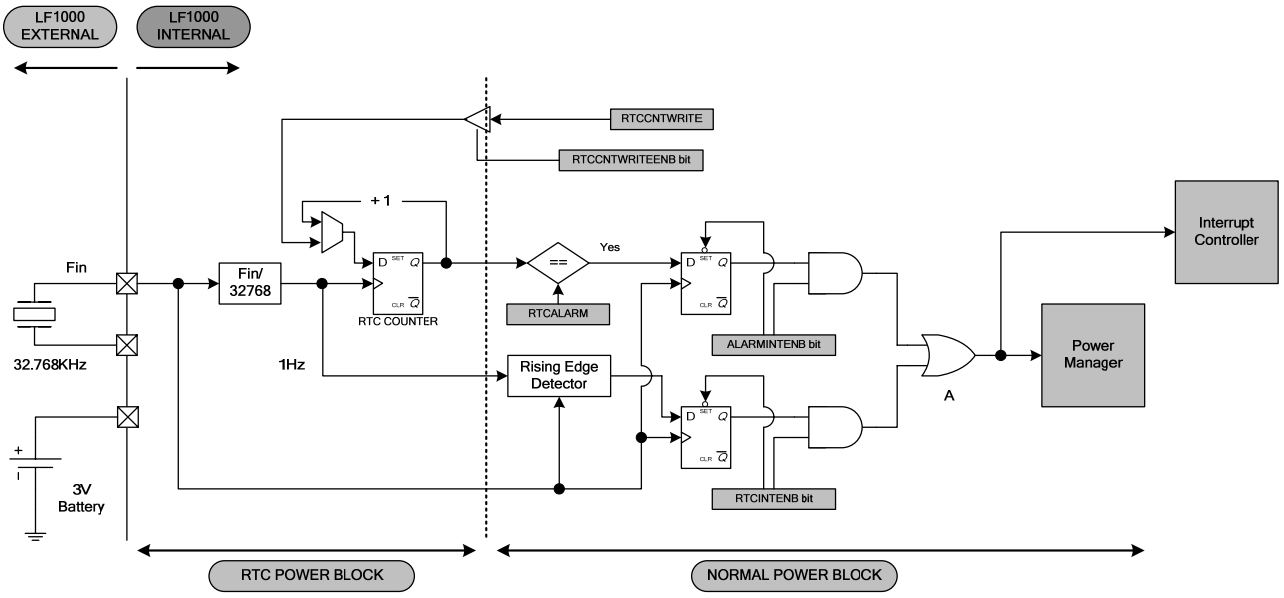


Figure 11-1. RTC Block Diagram

Figure 11-1 shows the RTC block diagram. The RTC block receives an external clock of 32.768 KHz and divides it into 1 Hz with 32.768 KHz. The RTC Counter operates depending on the external clock.

<Note 1>

As shown in Figure 11-1, the left and right parts of the central dotted line use RTC Power and Normal Power, separately. The RTC Power Block uses a mercury battery, but the block actually using the mercury battery is the RTC Counter in the RTC Power Block. The battery life is about five years.

In Figure 11-1, the output in point [A] is applied to the Power Manager or the Interrupt Controller. The output is applied to the Interrupt Controller in Normal mode and applied to the Power Manager in Power mode.

<Note 2>

Even if RTC is not used, RTC power and RTC clock should be supplied.

11.2. Operation

11.2.1. Backup Battery Operation

As shown in Figure 11-1, since the RTC block uses a separate power source (Coin Battery), the RTC block operates even when the external power is turned off.

The RTC Logic can be driven by the Backup Battery, which supplies the power through the VDD_RTC pin into the RTC Block, even if the system power is off. When the system power is off, the interfaces of the CPU and RTC logic should be blocked and the backup battery only drives the oscillation circuit and the internal 32bit RTC counter to minimize power dissipation. In other words, the RTC block can be used as the Wake Up Source when the POLLUX is converted into Power Down mode.

The use of the RTC block as a Wake-Up source requires that the ***RTCCTRL.ACCESSENB*** bit is set as '0' before the system enters Power Down mode. Setting it as '0' is performed to use the RTC as the Wake-Up source even when the system enters the Power Down Mode. (For detailed information on the Power Down Mode, refer to Chapter 4.)
Even if the RTC block is not used, the RTC Clock must be connected to POLLUX because the RTC clock is used as the clock for power management operation.

11.2.2. RTC Operation

The RTC generates an alarm signal at a specified time in the Power Down Mode or Normal Operation Mode. In Normal Operation Mode, the Alarm Interrupt is activated. In Power Down Mode, the Power Management Wake Up Signal is activated as well as the ***RTCALARM***. The ALARM Time Set register(***RTCALARM***) determines the condition of the alarm time setting and the ***RTCINTENB.ALARMINTENB*** bit determines the alarm enable/disable status.
The procedure to generate an alarm interrupt is as follows:
First, write a counter value to the ***RTCCNTWRITE*** register. (To this end, the busy status of the ***RTCCTRL.RTCCNTWAIT*** should be checked in advance. The written value is applied to the register after two 32.768 KHz clock cycles.) After that, write the value of the point at which you wish to generate an interrupt to the ***RTCALARM*** register. The RTC counter increases the counter value at intervals of 1 Hz. If the values of the two registers (***RTCCNTWRITE*** and ***RTCAKARM*** registers) become equal when the ***RTCINTENB.ALARMINTENB*** bit is set as '1', an interrupt occurs.
In a similar way, the RTC interrupt is detected in a rising edge of 1Hz. In this case, the interrupt is generated by setting the ***RTCINTENB.RTCINTENB*** bit as '1'.
In addition, The ***RTCINTENB*** register contains the Pending Clear function and the Pending Clear is performed by writing '0'.

11.2.3. Accessing the RTC Time Counter Setting/Read Register

To access RTC Time Count Read Register(***RTCCNTREAD***) and RTC Time Count Setting Register(***RTCCNTWRITE***), the ***RTCCTRL.RTCCNTWRITEENB*** bit is set to '1' before accessing these register. When the CPU completes to access these register, the CPU should set the ***RTCCTRL.RTCCNTWRITEENB*** bit to '0' to protect the content of RTC counter from unknown problem in abnormal state. The ***RTCCNTWRITEENB*** bit determines the reflection of the ***RCCNTWRITE*** register value to the RTC counter.

11.2.4. Interrupt Pending Register

Only the "READ" function is available for the ***RTCINTPND*** register of the POLLUX, but the current pending status can be read.
Since the ***RTCINTPND*** register only has a "READ" function, the Pending Clear function is controlled by the ***RTCINTENB*** register. The Interrupt Pending status is cleared by disabling the relevant interrupt. Therefore, if the corresponding bit of the ***RTCINTENB*** register is set as '1', the relevant interrupt is enabled. If the corresponding bit is set as '0', the interrupt is disabled and the pending bit is also cleared.

11.3. Register Summary

Bit	R/W	Symbol	Description	Reset Value
RTC TIME COUNT SETTING REGISTER (RTCCNTWRITE) ***Address : C000_F080h***				
[31 : 0]	W	**RTCCNTWRITE**	Set RTC Counter Value. (Unit : 1Hz) The ***RTCCNTWAIT*** bit of the RTCCONTROL register should be checked before a value is written in this bit. If the ***RTCCNTWAIT*** bit is '1', this written value is not reflected. The written value is reflected to this register at least two cycles of 32768 Hz after it is changed to '0'. To write a value to this register, the RTCCNTWRITEENB bit should be set as '1'.	32'bx
RTC TIME COUNT READ REGISTER (RTCCNTREAD) ***Address : C000_F084h***				
[31 : 0]	R	**RTCCNTREAD**	Read Current RTC Counter Value. (Unit : 1Hz) The value of the RTC counter is continuously changed.	32'bx
ALARM TIME COUNT SET REGISTER (RTCALARM) ***Address : C000_F088h***				
[31 : 0]	R/W	**RTCALARM**	ALARM Time Set Register. (Unit : 1Hz) The ***ALARMCNTWAIT*** bit of the ***RTCCONTROL*** register should be checked before a value is written in this bit. If the ***RTCCNTWAIT*** bit is '1', this written value is not reflected. The written value is reflected to this register at least two cycles of 32768 Hz after it is changed to '0'.	32'b0
RTC CONTROL REGISTER (RTCCTRL) ***Address : C000_F08Ch***				
[31 : 5]	-	**RESERVED**	Reserved	27'b0
[4]	R	**RTCCNTWAIT**	***RTCCNTWAIT*** : Register to check if the previously requested "WRITE" is completed when writing a value to the ***RTCCNTWRITE*** register. The bit below indicates the status of the ***RTCCNTWRITE*** register. 0 : IDLE 1 : Busy	1'b0
[3]	R	**ALARMCNTWAIT**	***ALARMCNTWAIT*** : Register to check if the previously requested "WRITE" is completed when writing a value to the ***RTCALARM*** register. The bit below indicates the status of the ***RTCALARMWRITE*** register 0 : IDLE 1 : Busy	1'b0
[2]	-	**RESERVED**	Reserved	1'b0
[1]	-	**RESERVED**	Reserved. However, '0' should be written.	1'b0
[0]	R/W	**RTCCNTWRITEENB**	***RTCCNTWRITEENB*** : Control Power isolation and connection of RTC block. This bit should be '0' before Power Down Mode for normal operation of RTC in Power Down Mode. 0 : Disable (Power Isolation) 1 : Enable (Power Connection) To access the RTCCNTREAD and RTCCNTWRITE registers, this bit should be set as '1'.	1'b0
RTC INTERRUPT ENABLE REGISTER (RTCINTENB) ***Address : C000_F090h***				
[31 : 2]	-	**RESERVED**	Reserved	30'b0
[1]	R/W	**ALARMINTENB**	***ALARMINTENB*** : Set ALARM Interrupt On/ Off and Pending Clear/ Interrupt Enable READ > 0 : Interrupt Disable 1 : Interrupt Enable WRITE> 0 : Pending Clear & Interrupt Disable 1 : Interrupt Enable	1'b0
[0]	R/W	**RTCINTENB**	***RTCINTENB*** : Set RTC(1Hz Only) Interrupt On/ Off and Pending Clear/ Interrupt Enable READ > 0 : Interrupt Disable 1 : Interrupt Enable WRITE> 0 : Pending Clear & Interrupt Disable 1 : Interrupt Enable	1'b0
RTC INTERRUPT PENDING REGISTER (RTCINTPND) ***Address : C000_F094h***				
[31 : 2]	-	**RESERVED**	Reserved	30'b0

Bit	R/W	Symbol	Description	Reset Value
[1]	R	**ALARMINTPEND**	***ALARMINTEPND*** : ALARM Interrupt Pending bit. 0 : None 1 : Interrupt Pended	1'b0
[0]	R	**RTCINTPEND**	***RTCINTPEND*** : Set RTC (1Hz Only) Interrupt Pending bit. 0 : None 1 : Interrupt Pended	1'b0

CHAPTER 12.

AUDIO CONTROLLER

12. AUDIO CONTROLLER

12.1. Overview

The Audio controller transmits the PCM data in memory to Audio Codec and/or receives the PCM data from Audio codec to memory. The Audio Controller of the POLLUX consists of an I^2S Controller. The I^2S Controller transmits/recieves data in I^2S, Left-Justified and Right-Justified data mode and supports various sampling rates.

12.1.1. Features

- I^2S FEATURES
  - 16/ 18 bits (18 bits is supported for playback only)
  - Master and Slave Mode
  - Master Mode : up to 192 kHz (128, 192, 256, 384fs)
  - Slave Mode : up to 192 kHz (128, 192, 256, 384fs)
  - I^2S, Left-Justfied, Right-Justified Data Mode

12.1.2. Block Diagram

The Audio block roughly consists of Clock Control, Top Control, Buffer & DMA Interface and I^2S Control blocks. The Audio Clock Control block receives the PLL and creates the clock for internal operation and external Codec Interface. The Audio Top Control block decides the I^2S and Audio Buffer operation. The Audio Buffer & DMA I/F block contains the data input/output buffer passing the Audio Controller and controls the operation. The Audio block has two buffers (a PCM output buffer, a PCM input buffer). The I^2S Control block controls data transfer with the I^2S Codec. The I^2S Controller read output data from the Audio buffer or write input data to the Audio buffer.

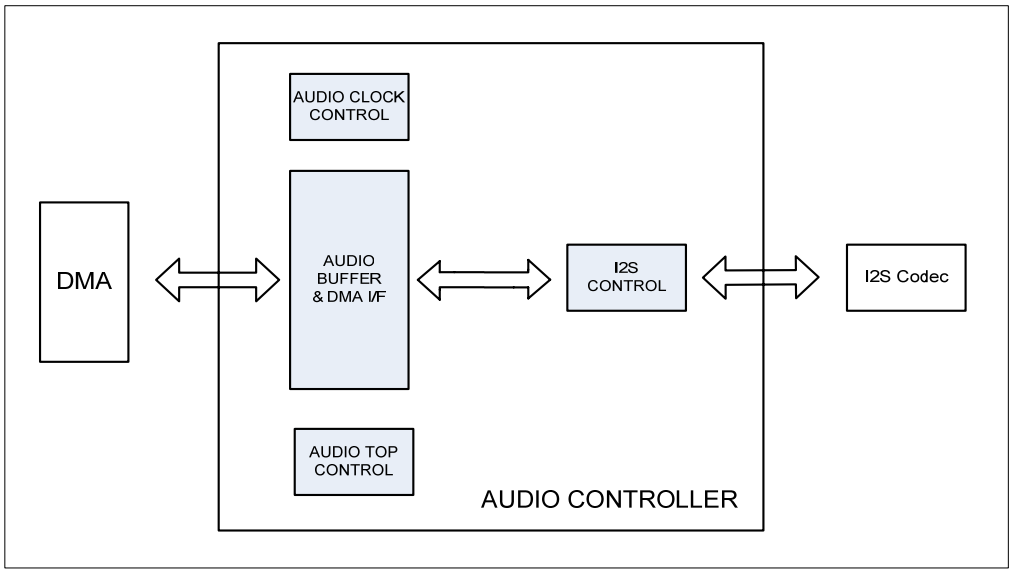


Figure 12-1. Audio Block Diagram

12.1.3. Pin Function Description

The Audio block uses five GPIO pins.

- I^2S

For data exchange between the I^2S Controller and I^2S Codec, a link between them should be set as ALT1 Function. The following five signals are connected to the I^2S Codec and could be used as GPIO pins:

Pin Name	GPIO No	GPIO Function	Type	Description
SYSCLK	A[25]	ALT1	O	SYSCLK is codec's operation clock. Used by codec only
I2S_SYNC	A[24]	ALT1	I/O	Left/Right audio data identifier. If controller master mode, this signal is output. If controller slave mode, this signal is input.
I2SDATA_IN	A[23]	ALT1	I	Serial audio Input data

Pin Name	GPIO No	GPIO Function	Type	Description
BITCLK	A[22]	ALT1	I/O	bit-rate clock. If controller master mode, this signal is output. If controller slave mode, this signal is input.
I2SDATA_OUT	A[21]	ALT1	O	Serial audio output data

Table 12-1. I²S Pin Function Description

12.2. Operation

12.2.1. I²S Controller Operation

The I²S Controller transmits data, synchronizing with the Sync signal. The I²S Controller supports both Master mode and Slave mode. The I²S Controller supports 128 fs, 192 fs, 256 fs and 384 fs. The I²S Controller is performed via five GPIO pins. The I²S Controller data format supports I²S, Left-justified and Right-justified mode. In I²S mode, data is transferred one BCLK clock after the Sync signal occurrence. Up to 18-bit data is supported by the I²S Controller.

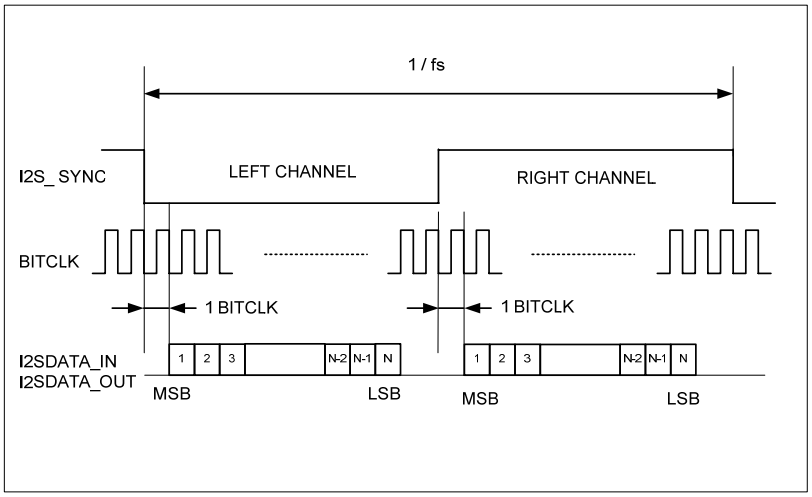


Figure 12-2. I²S Mode

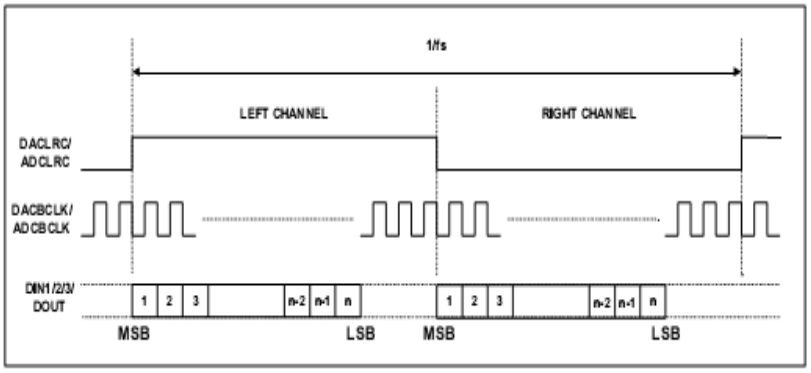


Figure 12-3. Left-justified Mode

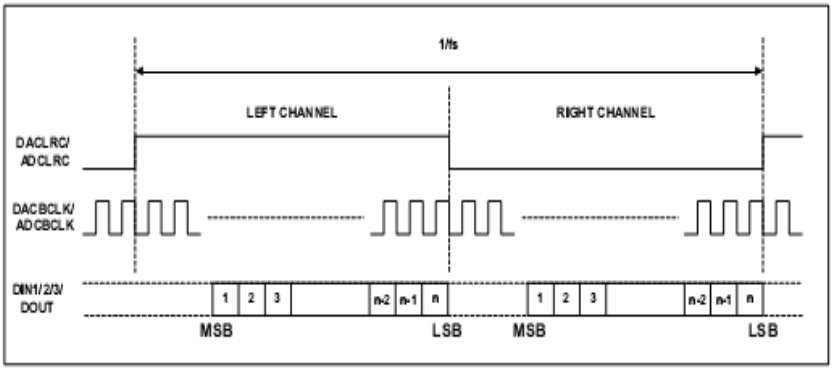


Figure 12-4. Right-justified Mode

12.2.1.1. GPIO

Before using the I²S Controller, the GPIO pins should be set properly. The GPIO setting is performed by using five GPIOA[25:21] pins, and the GPIO pins should be set as ALT1 Function.

12.2.1.2. Clock and Sampling Frequencies

The settings for the source clock and the clock divide value are required to create SYSCLK and BITCLK for the I²S operation. In Master mode, internal PLL, BIT or AV clock can be used as the source clock. In Slave mode, BITCLK of an external Codec can be used as the source clock. As shown in the figure below, The BITCLK value should be a quarter of the MCLK value (system clock). The value of ***CLKGEN1.CLKDIV1*** is set to divide-by-4. The figure below shows the source clock that can be selected for the I²S Controller. BITCLK, SYSCLK ratio is not restricted to 4 for both master and slave mode. For example, If sampling frequency is 192fs, BITCLK divider can be 3 (LRC is 64fs) or 4 (LRC is 48fs).

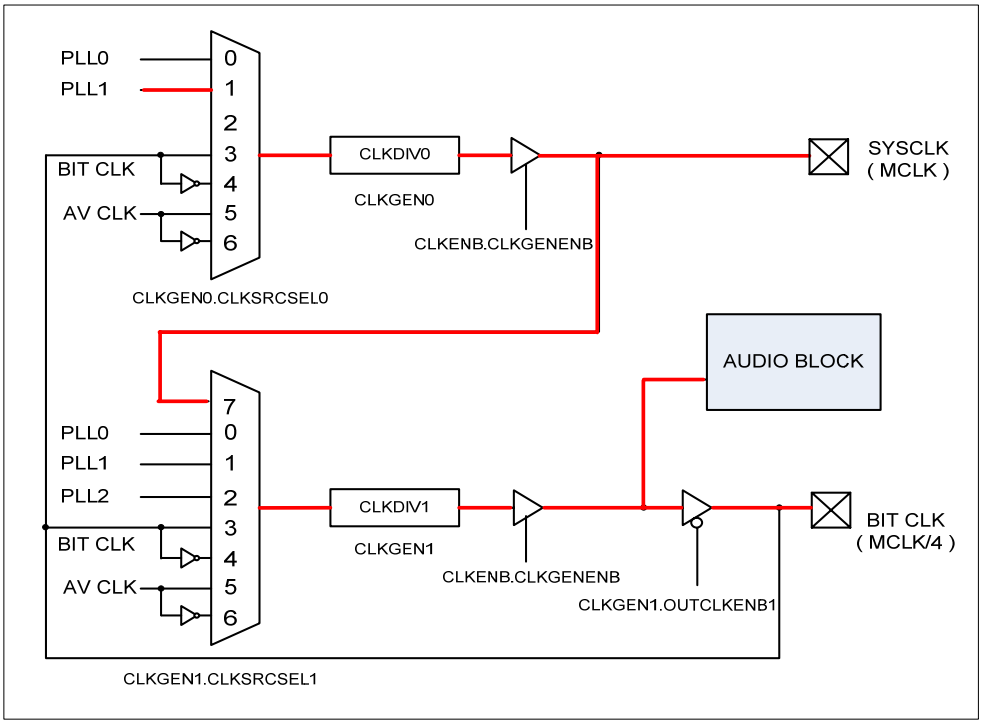


Figure 12-5. I²S Clock Select(Master Mode)

If the I²S Controller is used in Master mode (256 fs) and the source clock uses the internal PLL1 (98304000Hz), the value of CLKDIV0 to create a 48 KHz sampling frequency is calculated as follows:

SYSCLK(MCLK) = PLL1 / CLKDIV0
= 98304000Hz/ CLKDIV0(8) = 12288000Hz

BITCLK = MLCK / CLKDIV1
= 12288000Hz / CLKDIV(4) = 3072000Hz

SYNC (Sampling Frequency) = BITCLK/ Bit Clock Cycle
= 3072000Hz / 64 = 48000Hz

The values of the Bit Clock cycle are 32, 48, 64 and 96 for sampling frequencies of 128 fs, 192 fs, 256 fs and 384 fs, respectively. Therefore, you can get sampling rates in 256 fs as listed in the table below. The divider value of ***CLKGEN0(1).CLKDIV0(1)*** is set as Divider - 1.

Sampling Rate	CLKDIV0	CLKDIV1
96KHz	4	4
48KHz	8	4

32KHz	12	4

Table 12-2. Sampling Rates by CLKDIV Values in 256 fs (Suppose PLL1 is 98304000)

12.2.1.3. Initialization

- I²S Controller

The I²S Controller is initialized in Main Reset and Controller Reset modes. The Main Reset mode indicates the status in which the power is initially supplied or the Hardware Reset pin is pressed. In Controller Reset mode, the controller is initialized when Bit 0 of ***I2S_CTRL.I2S_ENB*** register is set as '0'. At this point, the Sync signal and data output go to '0' and the controller enters Idle mode. Although the controller is initialized, the Audio Data Buffer is not initialized. The Audio Data Buffer is initialized by clearing the relevant bit of the ***AUDIO_BUFF_CTRL*** register.

- I²S Codec

I²S Codec cannot be directly controlled via the I²S Controller. For the initialization and control of the I²S Codec, the control protocols (I²C, SSP/SPI, etc.) of the corresponding Codec should be used

12.2.1.4. I²S State

- Idle State : Except for the ***I2S_CTRL*** register, all registers have the reset values.
- Ready State : Access to the I²S Codec is allowed, but the link is disabled.
- Run State : I²S Link is enabled.

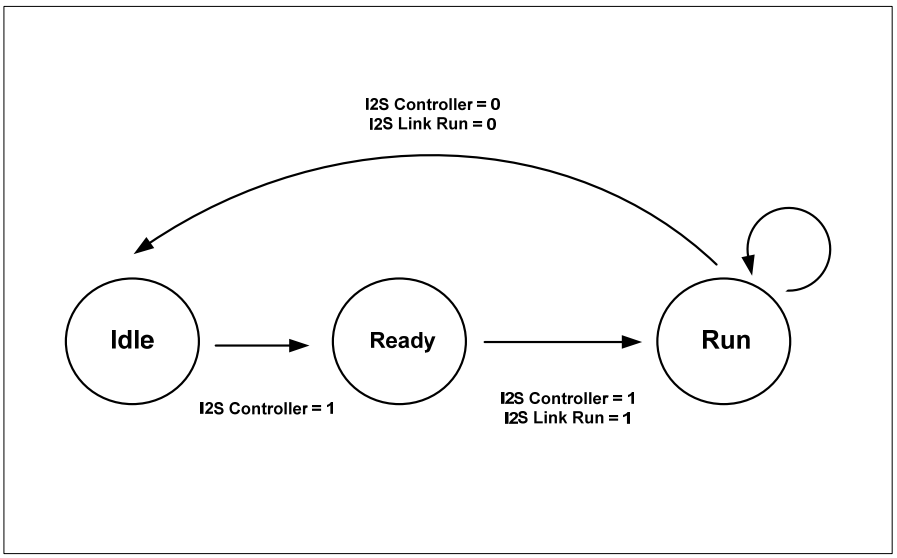
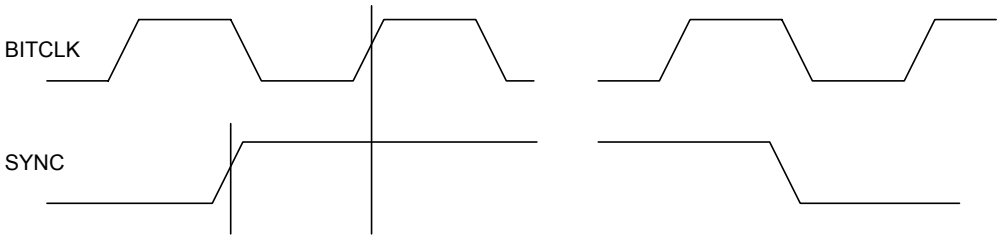


Figure 12-6. I²S Controller State

12.2.2. Timing Information

12.2.2.1. SYNC to BITCLK Timing



SYNC to BITCLK : (BITCLK period/2) – 1 ns

12.3. Valid Data Structure

12.3.1. Audio Buffer

The Audio Buffer consists of two buffers and each buffer uses separate DMA channels. The two buffers are the PCM Output, PCM Input buffers.

In initial mode, the output Buffer is in the Not Ready state and the Input Bufferis in the Ready state. The buffers are initialized in the following states.

- In Main Reset state
- When I^2S Controller is Off
- When the value of the corresponding Buffer Enable (***AUDIO_BUFF_CTRL*** register) is '0'

Data are stored in the Audio Buffer through DMA and the data saved in the Audio Buffer are transferred to each Codec by the I^2S Controller. In this case, the ***AUDIO_BUFF_CTRL*** register is used to select data transfer to the Audio Buffer through DMA.

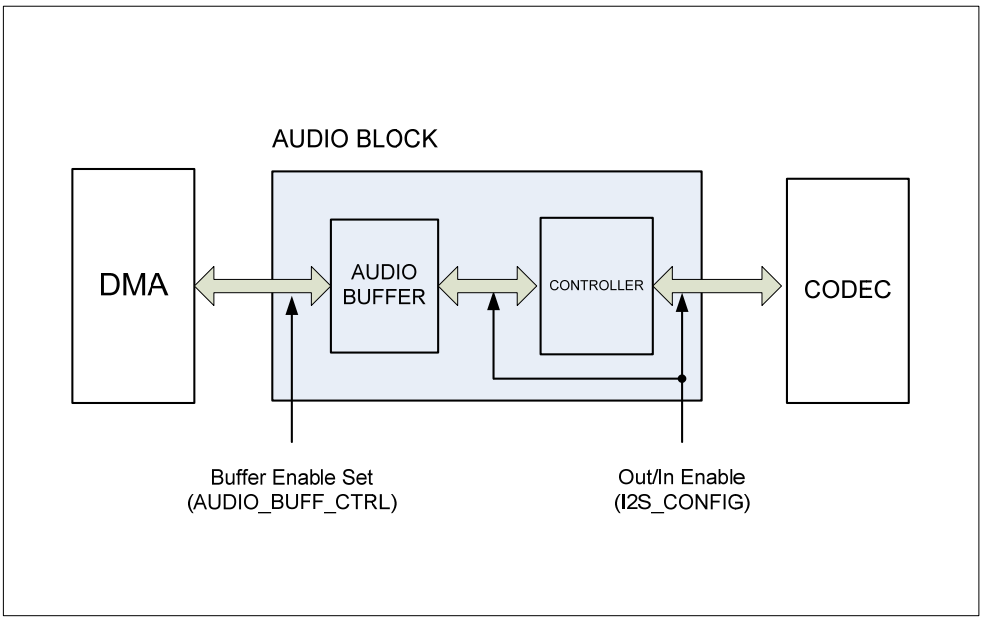


Figure 12-7. Buffer Enable & In/Out Enable

12.3.2. Output Buffer

The Audio controller has Output buffers: PCM Output buffer.

12.3.2.1. PCM Output Buffer

The I^2S Controller uses the PCM Output Buffer (32 x 18 bit FIFO). The DMA Controller reads and stores the data to send out to the PCM Output buffer from memory. If the PCM Output data request is received from the Audio Controller, the data in the Audio buffer is

transmitted to the controller. For normal PCM output, data array should satisfy the following items:

1) The left data should be input first.

- If Data Width = 16-bit, the data is based on 32-bit words.
  - The lower 16 bits are Left Data and:
  - The upper 16 bits are Right Data.
- If Data Width > 16-bit, the data is based on 32-bit words.
  - Even address words are Left Data and:
  - Odd address words are Right Data.

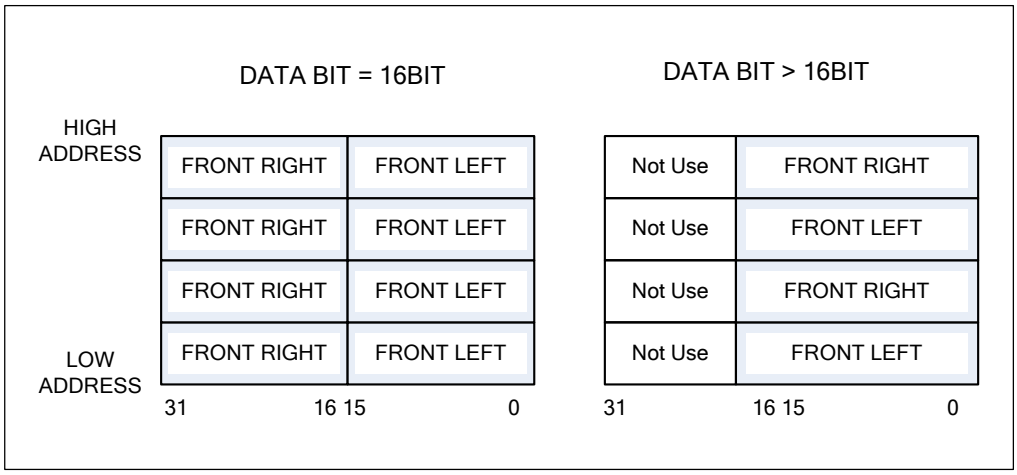


Figure 12-8. OUTPUT MEMORY FORMAT

12.3.3. PCM Input Buffer

The Input Buffer consists of PCM Input buffer.

- PCM Input Buffer : 32 x 16 Bit FIFO.

If Data Width is 16-bit and data input is PCM input, the left data are stored in memory first and afterward the right data are stored.

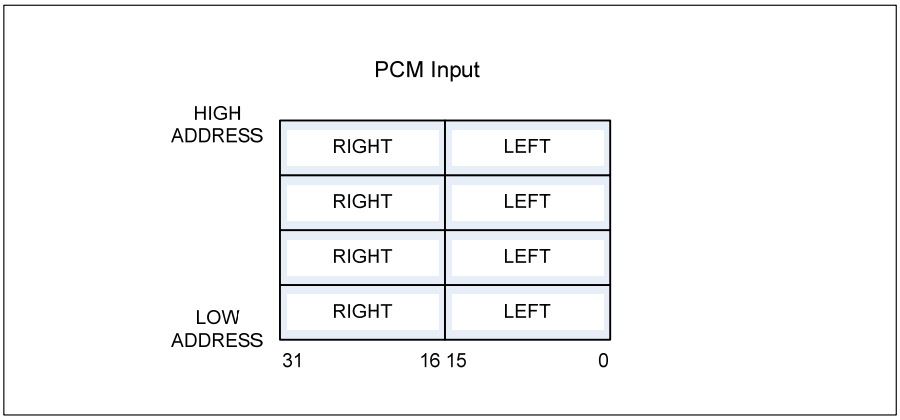


Figure 12-9. INPUT MEMORY FORMAT

12.3.4. Buffer Status

The IRQ Pending register (***AUDIO_IRQ_PEND***) and Audio Status1 register (***AUDIO_STATUS1***) are used to check the buffer status. Regardless of the IRQ Pending Register Enable status, the corresponding bit of the Pending register is set if Buffer Underrun or Overrun occurs. The Ready status of each Buffer can be checked via the ***AUDIO_STATUS1*** register. ***AUDIO_IRQ_PEND*** and ***AUDIO_STATUS1*** are different in the following respect: if a specific bit of ***AUDIO_IRQ_PEND*** is set as '1', the bit holds the value, '1', until the CPU clears the bit, but the value of the corresponding bit of ***AUDIO_STATUS1*** is changed depending on the buffer status.

Let's suppose that an Underrun event occurred in the PCM Output Buffer, but the Underrun event is cleared by the input of the next output data via DMA. The Pending bit of ***AUDIO_IRQ_PEND*** holds the value, '1', until the CPU clears it. However, the value of the Ready bit in ***AUDIO_STATUS1*** is changed from '0' to '1' if the Underrun event is cleared. In other words, ***AUDIO_STATUS1*** indicates the normal

status. If the output buffer is in the Underrun status, the last effective data is repeatedly output. If the input buffer is in the Overrun status, the following input data are ignored.

12.4. Operation Flow

12.4.1. I²S

For data and codec transfer through the I²S Controller, the GPIO, DMA, AUDIO BUFFER and I²S Controller should be set as follows:

- GPIO
  Set the relevant GPIO pin as I²S Mode. The I²S Controller is performed by using five GPIOA[25:21] pins and should be set as ALT1 Function.
- DMA
  Input or output the data exchanged with I²S Codec via DMA. The DMA default setting should be completed before the corresponding buffer of the Audio Buffer block is enabled.
- Audio Buffer
  After setting Data Width, enable a buffer to be used. If the output buffer is enabled, data fetch is started. Therefore, data should be prepared before the buffer is enabled and DMA should be set to fetch memory data. In addition, AUDIO IRQ should be set as Enable.
- I²S Controller
  After setting the I²S Controller as Enable, set the I²S operation mode (Master or Slave). The sampling frequency is set by using the Clock (MCLK and BITCLK). Since Data Width follows the AUDIO BUFFER BLOCK setting, the additional setting is not required. When ***I2S_CTRL*** is set as 0x03, the I²S - link is enabled.

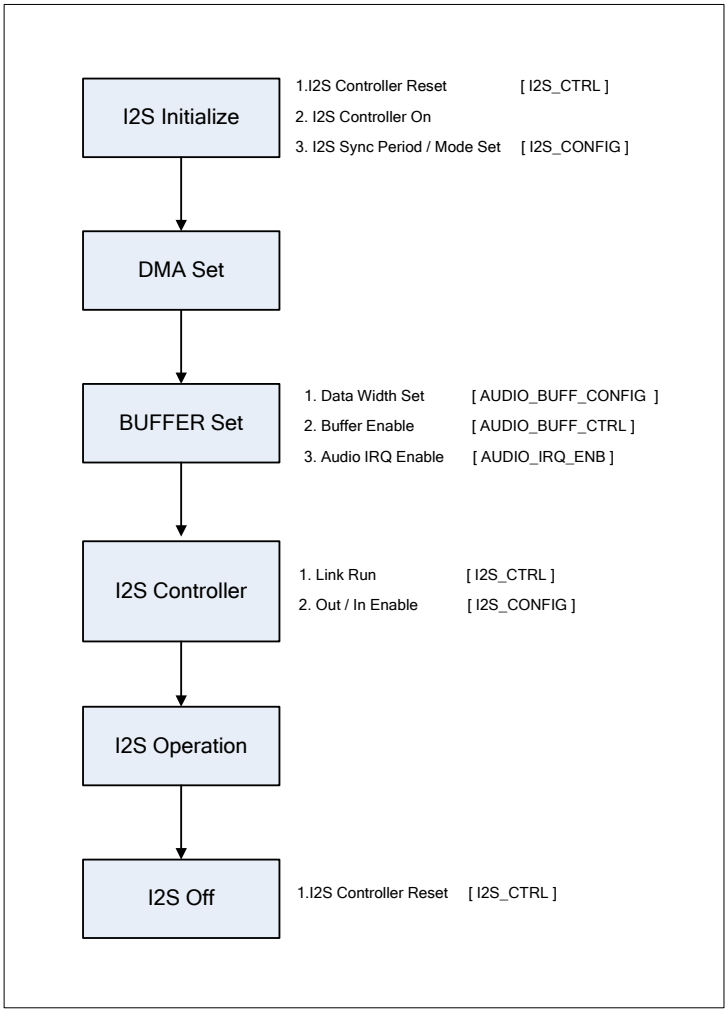


Figure 12-10. I²S Operation Flow

12.5. Register Summary

Bit	R/W	Symbol	Description	Reset Value
Reserved				
Address : C000_D800h				
[15:0]	-	**RESERVED**	Reserved	16'b0
Reserved				
Address : C000_D802h				
[15:0]	-	**RESERVED**	Reserved	16'b0
I²S CONTROL REGISTER (I2S_CTRL)				
Address : C000_D804h				
[15:3]	-	**RESERVED**	Reserved	14'b0
[1]	R/W	**I2SLINK_RUN**	I²S RUN (Link Enable) 0 : None 1 : Run (Link On)	1'b0
[0]	R/W	**I2S_EN**	Set I²S Controller 0 : Controller Reset 1 : Controller On	1'b0
I²S CONFIG REGISTER (I2S_CONFIG)				
Address : C000_D806h				
[15:8]	-	**RESERVED**	Reserved	8'b0
[7:6]	R/W	**IF_MODE**	00 : I2S mode 01 : forbidden 10 : Left-Justified 11 : Right-Justified	2'b0
[5 : 4]	R/W	**SYNC_PERIOD**	The setting is required in Master mode. 00 = 32fs 01 = 48fs 10 = 64fs	2'b0
[3]	R/W	**LOOP_BACK**	Loop Back Mode (Input data is directly send to output data) 0 : Disable 1 : Enable	1'b0
[2]	R/W	**I2SI_EN**	I²S Link Input Enable 0 : Disable 1 : Enable	1'b0
[1]	R/W	**I2SO_EN**	I²S Link Output Enable 0 : Disable (Mute output & No Buffer Request) 1 : Enable	1'b0
[0]	R/W	**MST_SLV**	Controller Mode. 0 : Master Mode 1 : Slave Mode (Codec Master)	1'b0
BUFFER CONTROL REGISTER (AUDIO_BUFF_CTRL)				
Address : C000_D808h				
[15:2]	-	**RESERVED**	Reserved	14'b0
[1]	R/W	**PCMIBUF_EN**	PCM IN Buffer Enable 0 : Disable (No DMA Request) 1 : Enable	1'b0
[0]	R/W	**PCMOBUF_EN**	PCM OUT Buffer Enable 0 : Disable (No DMA Request) 1 : Enable	1'b0
BUFFER CONFIG REGISTER (AUDIO_BUFF_CONFIG)				
Address : C000_D80Ah				
[15:6]	-	**RESERVED**	Reserved	10'b0
[5 :4]	R/W	**PI_WIDTH**	PCM IN Width. 00 : 16 bit others : reserved	2'b0
[3 : 2]	R/W	**RESERVED**	Reserved	2'b0
[1 : 0]	R/W	**PO_WIDTH**	PCM OUT Width. 00 : 16 bit 01: reserved 11 : 18 bit others : reserved	2'b0
IRQ ENABLE REGISTER (AUDIO_IRQ_ENA)				
Address : C000_D80Ch				
[15:2]	-	**RESERVED**	Reserved	14'b0
[1]	R/W	**PIOVER_IRQ_EN**	PCM IN Buffer IRQ Enable 0 : Disable 1 : Enable	1'b0
[0]	R/W	**POUDR_IRQ_EN**	PCM OUT Buffer IRQ Enable 0 : Disable 1 : Enable	1'b0
IRQ STATUS REGISTER (AUDIO_IRQ_PEND)				
Address : C000_D80Eh				

Bit	R/W	Symbol	Description	Reset Value
[15:2]	-	**RESERVED**	Reserved	14'b0
[1]	R/W	**PIOVR_PEND**	PCM IN Buffer OVER RUN Read > 0 : Not Over Run 1 : Over Run Pending Write > 0 : None 1 : Pending Clear	1'b0
[0]	R/W	**POUDR_PEND**	PCM OUT Buffer UNDER RUN Read > 0 : Not Under Run 1 : Under Run Pending Write > 0 : None 1 : Pending Clear	1'b0
Reserved ***Address : C000_D810h***				
[15 : 0]	-	**RESERVED**	Reserved	16'b0
Reserved ***Address : C000_D812h***				
[15 : 0]	-	**RESERVED**	Reserved	16'b0
Reserved ***Address : C000_D816h***				
[15 : 0]	-	**RESERVED**	Reserved	16'b0
Reserved ***Address : C000_D816h***				
[15:0]	-	**RESERVED**	Reserved	16'b0
AUDIO STATUS 1 (AUDIO_STATUS1) ***Address : C000_D818h***				
[15:2]	-	**RESERVED**	Reserved	14'b0
[1]	R	**PIBUF_RDY**	PCM IN Buffer Ready 0 : None 1 : Buffer Ready	1'b1
[0]	R	**POBUF_RDY**	PCM OUT Buffer Ready 0 : None 1 : Buffer Ready	1'b0
RESERVED ***Address : C000_D81Ah ~ C000_D87Eh***				
[15 : 0]	-	**RESERVED**	Reserved	16'bx
RESERVED ***Address : C000_D880h ~ C000_DBBEh***				
[15 : 0]	-	**RESERVED**	Reserved	16'bx
CLOCK ENABLE REGISTER (CLKENB) ***Address : C000_DBC0h***				
[31: 4]	-	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	PCLK operating mode 0 : Disable 1: Enable	1'b0
[2]	R/W	**CLKGENENB**	Clock Generation Enable 0 : Disable 1: Enable	1'b0
[1: 0]	R/W	**RESERVED**	Reserved	2'b0
CLOCK GENERATE REGISTER 0 (CLKGEN0) ***Address : C000_DBC4h***				
[15 : 10]	-	**RESERVED**	Reserved	6'b0
[9 : 4]	R/W	**CLKDIV0**	Clock Divisor. Write a value of (the actual clock divide – 1). Set as 0(1 divide)in I^2S Master Mode Max. Range [0, 63] : 1 clock divide ~ 64 clock divide	6'b0
[3 : 1]	R/W	**CLKSRCSEL0**	Clock Source Selection 000 : PLL0 001 : PLL1 010 : Reserved 011 : Bit Clock 100 : Inversion of Bit Clock 101 : AV Clock 110 : Inversion of AV Clock 111 : None	3'b0

Bit	R/W	Symbol	Description	Reset Value
[0]	R/W	**OUTCLKINV0**	Output Clock Inversion 0 : Normal 1: Inverting	1'b0
CLOCK GENERATE REGISTER 1 (CLKGEN1) ***Address : C000_DBC8h***				
[15]	R/W	**OUTCLKENB1**	Output Clock Enable This bit decides the I/O direction when the output clock is connected to a bidirectional PAD. 0 : Output 1 : Input	1'b0
[14 : 10]	-	**RESERVED**	Reserved	5'b0
[9 : 4]	R/W	**CLKDIV1**	Clock Divisor. Write a value of (the actual clock divide – 1). Set as 3(4 divide)in I^2S Master Mode Max. Range [0, 63] : 1 clock divide ~ 64 clock divide	6'b0
[3 : 1]	R/W	**CLKSRCSEL1**	Clock Source Selection 000 : PLL0 001 : PLL1 010 : Reserved 011 : Bit Clock 100 : Inversion of Bit Clock 101 : AV Clock 110 : Inversion of AV Clock 111 : CLKGEN0 output clock	3'b0
[0]	R/W	**OUTCLKINV1**	Output Clock Inversion 0: Normal 1: Inverting	1'b0

CHAPTER 13.

SD/MMC CONTROLLER

13. SD/MMC CONTROLLER

13.1. Overview

Multi-Media Card(MMC), Secure Digital(SD) memory card and SDIO device are supported by the SD/MMC controller. MMC specification version 4.2, SD memory card specification version 2.0 and SDIO card specification version 1.10 are appropriate for the SD/MMC controller.

13.1.1. Features

- Compatible with the Multi-Media Card System Specification, version 4.2
- Compatible with the SD Memory Card Specification, version 2.0
- Compatible with the SDIO Card Specification, version 1.10
- Supports clock speeds up to 50 MHz+
- Contains an Internal Clock Pre-Scaler
- Contains 64 Bytes of FIFO for data Receive/Transmit
- Supports PIO and DMA Mode data transfer
- Supports 4-bit data bus
- Two Channels of SD/MMC

13.1.2. Block Diagram

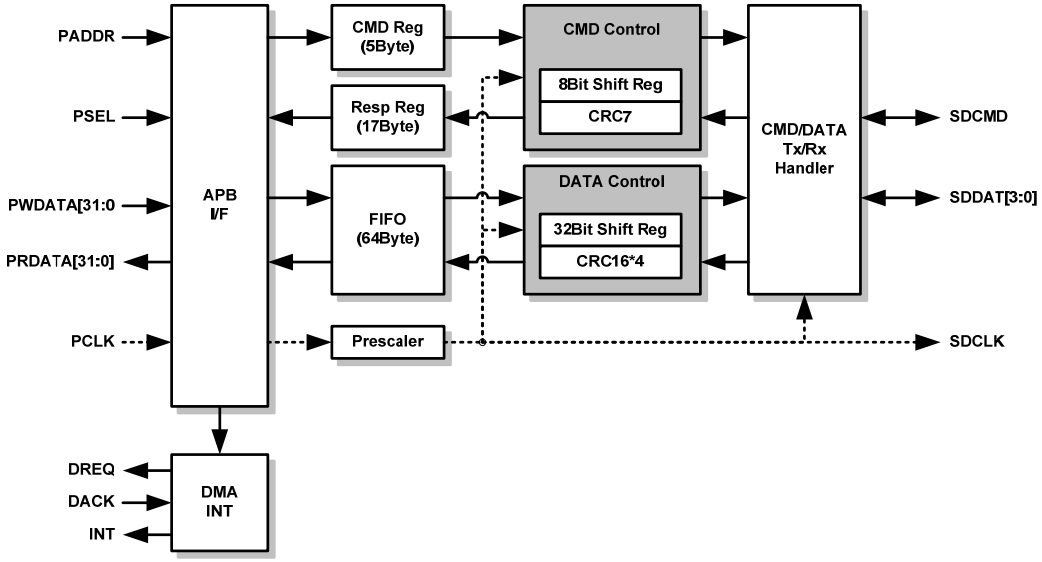


Figure 13-1. SD/MMC Block Diagram

13.1.3. Pin Function Description

Pin Name	GPIO No	GPIO Function	Type	Description
SDDAT0[3:0]	B[5:2]	ALT1	I/O	SD/MMC0 Data[3:0]
SDCLK0	B[0]	ALT1	O	SD/MMC0 Clock
SDCMD0	B[1]	ALT1	I/O	SD/MMC0 Command
SDDAT1[3:0]	B[11:8]	ALT1	I/O	SD/MMC1 Data[3:0]
SDCLK1	B[6]	ALT1	O	SD/MMC1 Clock
SDCMD1	B[7]	ALT1	I/O	SD/MMC1 Command

Table 13-1. SD/MMC Pin Function Description

13.2. SD/MMC Card Information

13.2.1. SD and MMC Card Pin Assignments

SD and MMC cards are 7 pin or 9 pin cards that operate as external memory storage for the POLLUX. The pin assignment and form factor are shown in Table 13-2.

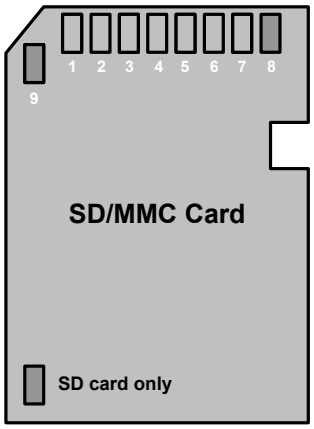
Card Shape	Pin Number	MMC Card	SD Card	
			1-Bit Mode	4-Bit Mode
	1	Reserved	Card Detect	Data Line DAT [3]
	2	Command / Response (CMD)		
	3	Supply Voltage Ground (Vss1)		
	4	Supply Voltage (Vdd)		
	5	Clock (CLK)		
	6	Supply Voltage Ground (Vss2)		
	7	Data Line DAT [0]		
	8		Interrupt (IRQ)	Data Line DAT [1] or Interrupt (IRQ)
	9		Read Wait (RW)	Data Line DAT [2] or Read Wait (RW)

Table 13-2. SD/MMC Card Pin Assingment

13.2.2. SD and MMC Card Registers

SD and MMC cards have a set of information registers. The Card registers be shown in Table 13-3

Name	Width	Description
CID	128	Card identification number; card individual number for identification. (Mandatory.)
RCA1	16	Relative card address; local system address of a card, dynamically suggested by the card and approved by the host during initialization. (Mandatory)
DSR	16	Driver Stage Register; to configure the card's output drivers. (Optional)
CSD	128	Card Specific Data; information about the card operation conditions. (Mandatory)
SCR	64	SD Configuration Register; information about the SD Memory Card's Special Fea-tures capabilities. (Mandatory)
OCR	32	Operation condition register. (Mandatory)

Table 13-3. SD/MMC Card registers

13.3. Operation

The clock frequency, for the transmission of the controller and the device, can be adjusted by the clock generate register (***CLKGEN***) and clock divider register (***CLKDIV***) of the SD controller. The samplings of commands and data are synchronized by this clock.

13.3.1. Bus protocol

Communication over the SD bus is based on command and data bit streams which are initiated by a start bit and terminated by a stop bit.

- **Command**: A command is an operation starting token. It's sent from the controller to a card and is transferred serially on the CMD line.

  Command Format : Command consists of 48 bits(6Byte)

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width(bits)	1	1	6	32	7	1
Value	'0'	'1'	x	x	x	'1'
Description	Start bit	Transmission bit	Command index	argument	CRC7	End bit

Table 13-4. Command Format

- **Response**: A response is a token that is sent from a card as an answer to a previously received command to the controller and is transferred serially on the CMD line.

  Normal Response Format : Response consists of 48 bits(6Byte)

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width(bits)	1	1	6	32	7	1
Value	'0'	'1'	X	x	x	'1'
Description	Start bit	Transmission bit	Command index	Card status	CRC7	End bit

Table 13-5. Normal Response Format

Long Response Format : Response consists of 136 bits(17Byte)

Bit position	135	134	[133:128]	[127:1]	0
Width(bits)	1	1	6	127	1
Value	'0'	'1'	'111111'	x	'1'
Description	Start bit	Transmission bit	reserved	CID or CSD register incl. internal CRC7	End bit

Table 13-6. Long Response Format

- **Data**: Data can be transferred from the controller to the card or vice versa and it is transferred via the data line.

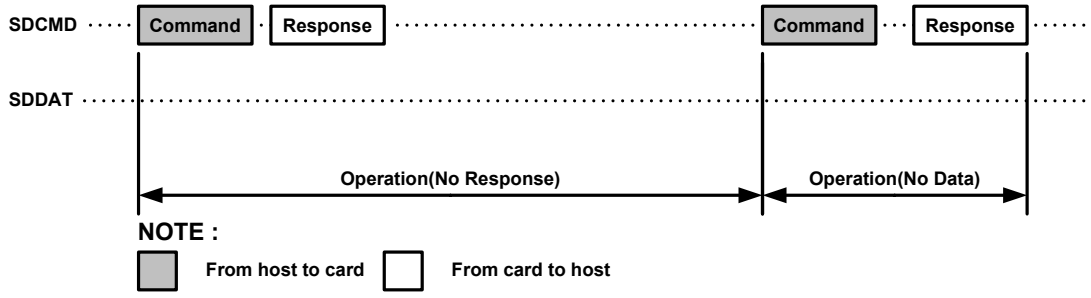


Figure 13-2. Operation Without Data

Card addressing is implemented using a session address, assigned to the card during the initialization phase. The basic transaction on the SD bus is the command/response transaction (refer to Figure 13-2). This type of bus transactions transfers their information directly within the command or response structure. Inaddition, some operations have a data token.

Data transfers from/to the card are done in block. CRC bits always follow Data blocks that defines the single and multiple block operations. For faster write operations, the multiple block operation mode is better. The controller terminates a multiple block transmission when a stop command follows on the CMD line. The host can configure data transfer to use single or multiple data lines.

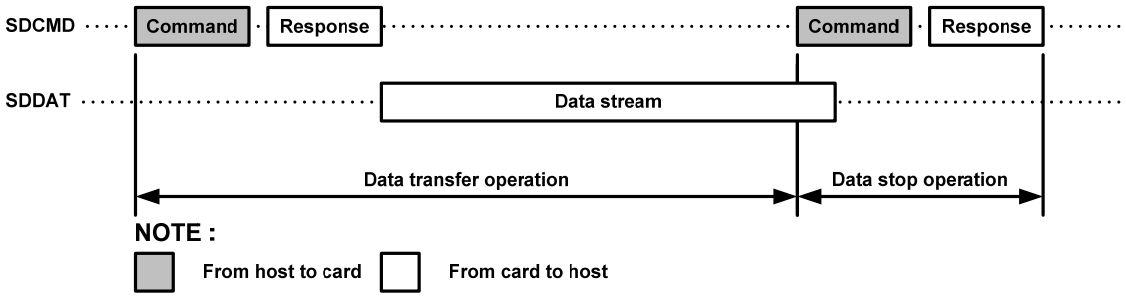


Figure 13-3. Stream Read Operation

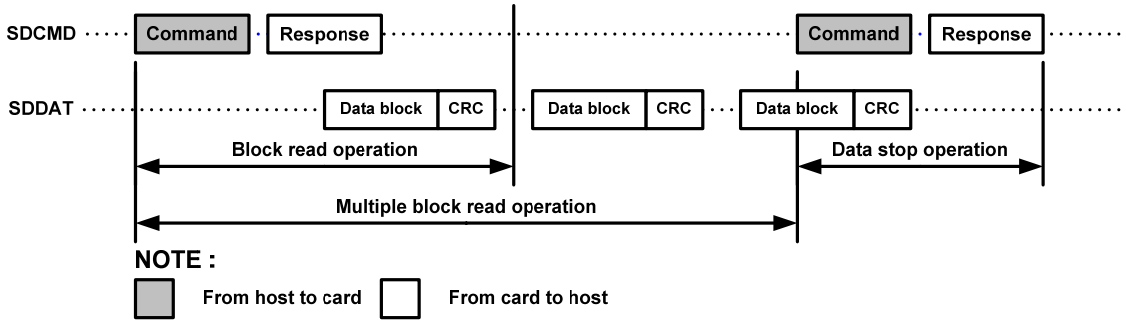


Figure 13-4. Multiple Block Read Operation

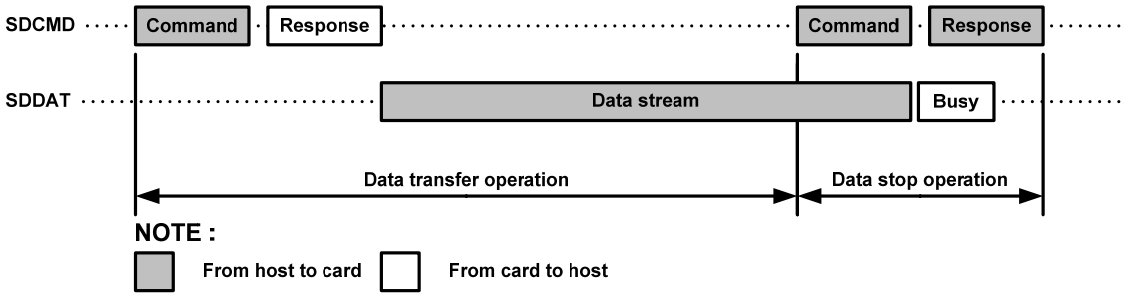


Figure 13-5. Stream Write Operation

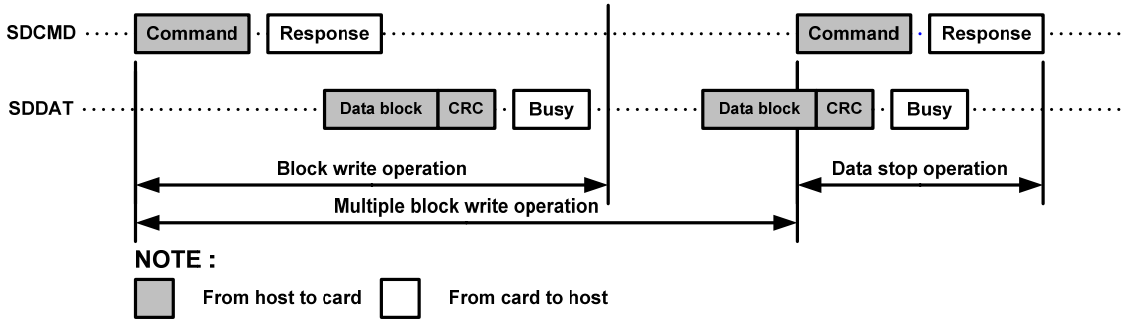


Figure 13-6. Multiple Block Write Operation

13.3.2. Initialization

To program the SDI modules, follow these basic steps:

1. Set ***CTRL*** to clear FIFO, DMA interface and internal FSM.
2. Set ***CTYPE*** and ***MINTSTS*** to configure properly with card type and interrupt enable
3. Set ***CLKGEN*** and ***CLKDIV*** to configure with a proper value.
4. Initialization delay : The maximum of 1msec, 74 SDCLK clock cycles in order to initialize the card.

For the sequences to execute their actual operations, refer the operation sequence examples in 13.3.6

13.3.3. Error Detection

The controller checks the following errors and stores the results into the Raw Interrupt Status Register (***RINTSTS***) and the Masked Interrupt Status Register (***MINTSTS***) if corresponding interrupt is enabled.

- **Response CRC error:** Command response shows the calculation of CRC error.
- **Response error :** Response is invalid error.
- **Response time out:** No response begins before the specific number of clocks
- **Data CRC error:** A CRC status error on the data.
- **Read time out**: No read data operation begins before the specified number of clocks.
- **Data starvation by host timeout error** : No transmit data supplied by the host before the specific number of clocks
- **Start bit error** : Invalid start bit.
- **End bit error or No CRC error** : Invalid end bit(read) or no CRC (write)

13.3.4. Clocks

The Processor enables/disables the SD bus clock (SDCLK) by means of the bit 0 (***CLKENA.CCLKENA***) of the Clock Enable Register (***CLKENA***). The SD bus clock frequency is adjusted by the value of the Clock Generate Register (***CLKGEN***) and Clock Divider Register (***CLKDIV***). The clock frequency is determined by two steps.

1st step : Determine cclk_in input clock frequency by the value of the Clock Generate Register.

Maximum frequency of cclk_in is 50 MHz+.

Prescaler Value (CLKGEN)	cclk_in Frequency	Prescaler Value (CLKGEN)	cclk_in Frequency	Prescaler Value (CLKGEN)	cclk_in Frequency	Prescaler Value (CLKGEN)	cclk_in Frequency
1	50MHz	2	25MHz	3	16.6MHz	4	12.5MHz

2nd step : Determine card interface clock frequency by the value of the Clock Divider Register.

The clock frequency is determined by the following expression.

SDCLK = cclk_in/(2*CLKDIV)

Example) If cclk_in is 25 MHz and CLKDIV = 1, then SDCLK is 12.5 MHz.

(if CLKDIV = 0, then SDCLK frequency is same as cclk_in)

13.3.5. SDIO Operation

13.3.5.1. SDIO Interrupt

SDIO cards signal an interrupt by asserting SDDAT low during the interrupt period. An interrupt period is always valid for non-active or non-selected cards and 1-bit data mode for the selected card. An interrupt period for a wide-bus active or selected card is valid for the following conditions:

- Card is idle
- Non-data transfer command in progress
- Third clock after end bit of data block between two data blocks
- From two clocks after end bit of last data until end bit of next data transfer command

Bear in mind that, in the following situations, the SDMMC controller does not sample the SDIO interrupt of the selected card when the card data width is 4bits. Since the SDIO interrupt is level-triggered, it is sampled in a further interrupt period and the host does not lose any SDIO interrupt from the card.

1. Read/Write Resume

The card interface treats the resume command as a normal data transfer command. SDIO interrupts during the resume command are handled similarly to other data commands.

According to the SDIO specification, for the normal data command the interrupt period ends after the command end bit of the data command; for the resume command, it ends after the response end bit. In the case of the resume command, the controller stops the interrupt sampling period after the resume command end bit, instead of stopping after the response end bit of the resume command.

2. Suspend during read transfer
If the read data transfer is suspended by the host, the host sets the abort_read_data bit (***CTRL[8]***) to reset the data state machine. In the controller, the SDIO interrupts are handled such that the interrupt sampling starts after the abort_read_data bit is set by the host. In this case the controller does not sample SDIO interrupts between the period from response of the suspend command to setting the abort_read_data bit and starts sampling after setting the abort_read_data bit.

13.3.5.2. Read Wait Sequence

Read Wait is used with only the SDIO card and can temporarily stall the data transfer--either from function or memory--and allow the host to send commands to any function within the SDIO device.
The host can stall this transfer for as long as required. The controller provides the facility to signal this stall transfer to the card. The steps for doing this are:
1. Check if the card supports the read_wait facility; read SRW (bit2) of the CCCR register. If this bit is 1, then all functions in the card support the read_wait facility. Use CMD52 to read this bit.
2. If the card supports the read_wait signal, then assert it by setting the READWAIT bit (CTRL[6]).
3. Clear the READWAIT bit(CTRL[6]).

13.3.6. Operation Sequence Example

13.3.6.1. Card Identification Process

A. Detects the cards
B. Resets all cards that are in the card identification state
C. Validates operation voltage range
D. Identifies the cards

13.3.6.2. Multiple Block Write with polling and single data line

The write block operation sequence with a single data line, multiple blocks and a polling case is as follows:
1. Clear ***CTRL.DMAEN*** (bit[5]).
2.CMD7 selection of specific sdcard.
   1) Set ***CMDARG*** (card number<<16).
   2) Set ***CMD*** 801f0147h & (card number<<16)
   3) Wait until Command Done (***RINTSTS.CD***==1 when polling mode, or Command done interrupt issued when IRQ mode)
   4) Clear ***RINTSTS.CD*** bit
3. Write 64 bytes of data for Tx to ***DAT*** register
4. CMD25 write multiple block
   1) Set ***CMDARG*** with data address
   2) Set ***CMD*** 801f2759 & (card number<<16)
   3) Wait until Command Done (***RINTSTS.CD***==1 when polling mode, or Command done interrupt issued when IRQ mode)
   4) Clear ***RINTSTS.CD*** bit
5. Until ***RINTSTS.DTO***==1, repeat following rutines.
   1) Check Tx FIFO watermark status (***STATUS.TXWM***)
   2) If ***STATUS.TXWM***==1, write 32 bytes of data to ***DAT*** register
6. If ***RINTSTS.DTO***==1, reset FIFO (***CTRL.FIFORST***=1) and clear ***RINTSTS.DTO***

13.3.6.3. Multiple Block Read with polling and single data line

The block operation sequence with single data line, multiple blocks and polling case is as follows:
1. Clear ***CTRL.DMAEN*** (bit[5]).
2. CMD7 selection of specific sdcard.
   1) Set ***CMDARG*** (card number<<16).
   2) Set ***CMD*** 801f0147h & (card number<<16)
   3) Wait until Command Done (***RINTSTS.CD***==1 when polling mode, or Command done interrupt issued when IRQ mode)
   4) Clear ***RINTSTS.CD*** bit

3. CMD18 read multiple block
   1) Set ***CMDARG*** with data address
   2) Set ***CMD*** 801f2352 & (card number<<16)
   3) Wait until Command Done (***RINTSTS.CD***==1 when polling mode, or Command done interrupt issued when IRQ mode)
   4) Clear ***RINTSTS.CD*** bit
4. Until ***RINTSTS.DTO***==1, repeat following rutines.
   1) Check Rx FIFO watermark status (***STATUS.RXWM***)
   2) If ***STATUS.RXWM***==1, read 32 bytes of data from ***DAT*** register
5. If ***RINTSTS.DTO***==1,
   1) read ***STATUS.FCOUNT*** (bit[29:17])
   2) read (***STATUS.FCOUNT****4) bytes from ***DAT*** register
6. Reset FIFO (***CTRL.FIFORST***=1) and clear ***RINTSTS.DTO***

13.3.6.4. Multiple Block Read with DMA and single data line

The block operation sequence with single data line, multiple blocks and DMA case is as follows:

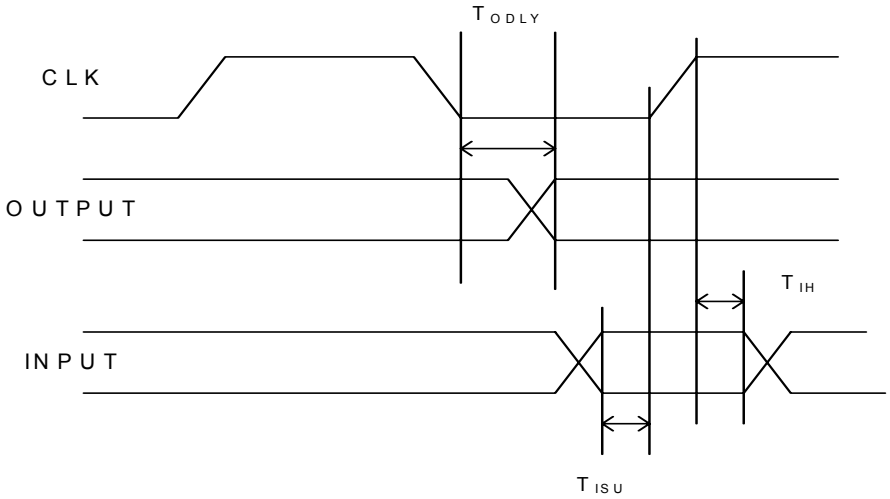
1. Set ***CTRL.DMAEN*** (bit[5]).
2.CMD7 selection of specific sdcard.
   1) Set ***CMDARG*** (card number<<16).
   2) Set ***CMD*** 801f0147h & (card number<<16)
   3) Wait until Command Done (***RINTSTS.CD***==1 when polling mode, or Command done interrupt issued when IRQ mode)
   4) Clear ***RINTSTS.CD*** bit
3. CMD25 write multiple block
   1) Set ***CMDARG*** with data address
   2) Set ***CMD*** 801f2759 & (card number<<16)
   3) Wait until Command Done (***RINTSTS.CD***==1 when polling mode, or Command done interrupt issued when IRQ mode)
   4) Clear ***RINTSTS.CD*** bit
4. Wait until ***RINTSTS.DTO***==1.
5. If ***RINTSTS.DTO***==1, reset FIFO (***CTRL.FIFORST***=1) and clear ***RINTSTS.DTO***

13.3.6.5. Multiple Block Read with DMA and single data line

The block operation sequence with single data line, multiple blocks and DMA case is as follows:

1. Set ***CTRL.DMAEN*** (bit[5]).
2. CMD7 selection of specific sdcard.
   1) Set ***CMDARG*** (card number<<16).
   2) Set ***CMD*** 801f0147h & (card number<<16)
   3) Wait until Command Done (***RINTSTS.CD***==1 when polling mode, or Command done interrupt issued when IRQ mode)
   4) Clear ***RINTSTS.CD*** bit
3. CMD18 read multiple block
   1) Set ***CMDARG*** with data address
   2) Set ***CMD*** 801f2352 & (card number<<16)
   3) Wait until Command Done (***RINTSTS.CD***==1 when polling mode, or Command done interrupt issued when IRQ mode)
   4) Clear ***RINTSTS.CD*** bit
4. Wait until ***RINTSTS.DTO***==1.
5. Wait until ***STATUS.FEMPTY***==1.
6. Reset FIFO (***CTRL.FIFORST***=1) and clear ***RINTSTS.DTO***

13.3.7. Signal Timing



Parameter	Symbol	Min	Max
Output Delay Time	T_{ODLY}		2
Input Set-up Time Requirement	T_{ISU}	0	
Input Hold Time Requirement	T_{IH}	10	

Note : All time unit is ns.

13.4. Register Summary

Bit	R/W	Symbol	Description	Reset Value
SDI CONTROL REGISTER (CTRL) ***Address : SD0 : C000_9800h/ SD1 : C000_C800h***				
[31:12]	-	**RESERVED**	Reserved	20'b0
[11:9]	R/W	**RESERVED**	Resetved. Must be '0'	3'b0
[8]	R?W	**ABORT_RDATA**	0 : No change. 1 : After suspend command is issued during read-transfer, software polls card to find when suspend happened. Once suspend occurs, software sets bit to reset data state-machine, which is waiting for next block of data. Bit automatically clears once data state- machine resets to idle. Used in SDIO card suspend sequence.	1'b0
[7]	R/W	**SEND_IRQ_RESP**	0 : No change 1 : Send auto IRQ response Bit automatically clears once response is sent. To wait for MMC card interrupts, host issues CMD40 and the controller waits for interrupt response from MMC card(s). In meantime, if host wants the controller to exit waiting for interrupt state, it can set this bit, at which time the controller command state-machine sends CMD40 response on bus and returns to idle state.	
[6]	R/W	**READ_WAIT**	For sending read-wait to SDIO cards. 0 : Clear read wait 1 : Assert read wait	
[5]	R/W	**DMA_ENA**	0 : Disable DMA transfer mode 1 : Enable DMA transfer mode	
[4]	R/W	**INT_ENA**	0 : Disable interrupt 1 : Enable interrupt	1'b0
[3]	R/W	**RESERVED**	Reserved.	1'b0
[2]	R/W	**DMARST**	Reset internal DMA interface logic. This bit is automatically cleared 0 : No change 1 : Reset DMA logic	1'b0
[1]	R/W	**FIFORST**	Reset FIFO value. This bit is automatically cleared. 0 : Normal mode 1 : FIFO reset	1'b0
[0]	R/W	**CTRLRST**	Reset sdmmc controller intrnal logic. Does not affect any registers or DMA interface, or FIFO or host interrupts. This bit is automatically cleared. 0 : Normal mode 1 : Reset controller	1'b0
RESERVED ***Address : SD0 :C000_9804h/ SD1 : C000_C804h***				
[31:0]	-	**RESERVED**	Reserved	32'b0
SDI CLOCK DIVIDER REGISTER (CLKDIV) ***Address : SD0 :C000_9808h/ SD1 : C000_C808h***				
[31:8]	-	**RESERVED**	Reserved	24'b0
[7:0]	R/W	**DIVIDER**	Clock divider value. Clock division is 2*n. For example, value of 0 means bypass, value 1 means divide by 2, value of 255 means divide by 510.	
RESERVED ***Address : SD0 :C000_980Ch/ SD1 : C000_C80Ch***				
[31:0]	-	**RESERVED**	Reesrved	32'b0
SDI CLOCK ENABLE REGISTER (CLKENA) ***Address : SD0 :C000_9810h/ SD1 : C000_C810h***				
[31:17]	-	**RESERVED**	Reserved	15'b0
[16]	R/W	**LOWPWR**	Low power control. If enabled, stop clock when card in idle status. 0 : Low power disable 1 : Low power enable	1'b0

Bit	R/W	Symbol	Description	Reset Value
[15:1]	R/W	**RESERVED**	Reserved	15'b0
[0]	R/W	**CLKENA**	Clock enable control 0 : Disable sd clock 1 : Enable sd clock	1'b0
SDI TIMEOUT REGISTER (TMOUT) ***Address : SD0 :C000_9814h/ SD1 : C000_C814h***				
[31:8]	R/W	**DTMOUT**	Value for card data read timeout. Same value also used for data starvation by host timeout. Value is in number of card output clocks.	24'hFF_FFFF
[7:0]	R/W	**RSPTMOUT**	Response timeout value. Value is in number of card output clocks.	8'h40
SDI CARD TYPE REGISTER (CTYPE) ***Address : SD0 :C000_9818h/ SD1 : C000_C818h***				
[31:17]	-	**RESERVED**	Reserved	15'b0
[16]	R/W	**8BIT**	8 bit interface mode. In POLLUX, this bit must be 0.	1'b0
[15:1]	-	**RESERVED**	Reserved.	15'b0
[0]	R/W	**WIDTH**	Card bus width. 0 : 1-bit mode 1 : 4-bit mode	1'b0
SDI BLOCK SIZE REGISTER (BLKSIZ) ***Address : SD0 :C000_981Ch/ SD1 : C000_C81Ch***				
[31:16]	-	**RESERVED**	Reserved	16'b0
[15:0]	R/W	**BLKSIZE**	Block size in bytes.	16'h200
SDI BYTE COUNT REGISTER (BYTCNT) ***Address : SD0 :C000_9820h/ SD1 : C000_C820h***				
[31:0]	R/W	**BYTCNT**	Number of bytes to be transferred. This value should be integer multiple of BLKSIZ for block transters. For undefined number of byte transfers, byte count should be set to 0.	32'h200
SDI INTERRUPT MASK REGISTER (INTMASK) ***Address : SD0 :C000_9824h/ SD1 : C000_C824h***				
[31:17]	-	**RESERVED**	Reserved	15'b0
[16]	R/W	**MSKSDIOINT**	SDIO interrupt mask. 0 : Masked 1 : Enabled	1'b0
[15]	R/W	**MSKEBE**	End bit error(read), Write no CRC (write) interrupt mask.	1'b0
[14]	R/W	**MSKACD**	Auto command done interrupt mask.	1'b0
[13]	R/W	**MSKSBE**	Start bit error interrupt mask	1'b0
[12]	R/W	**MSKHLE**	Hardware locked write error interrupt mask	1'b0
[11]	R/W	**MSKFRUN**	FIFO underrun/overrun error interrupt mask	1'b0
[10]	R/W	**MSKHTO**	Data starvation by host timeout interrupt mask	1'b0
[9]	R/W	**MSKDRTO**	Data read timeout interrupt mask	1'b0
[8]	R/W	**MSKRTO**	Response timeout interrupt mask	1'b0
[7]	R/W	**MSKDCRC**	Data CRC error interrupt mask	1'b0
[6]	R/W	**MSKRCRC**	Response CRC error interrupt mask	1'b0
[5]	R/W	**MSKRXDR**	Receive FIFO data request interrupt mask	1'b0
[4]	R/W	**MSKTXDR**	Transmit FIFO data request interrupt mask	1'b0
[3]	R/W	**MSKDTO**	Data transfer over interrupt mask	1'b0

Bit	R/W	Symbol	Description	Reset Value
[2]	R/W	**MSKCD**	Command done interrupt mask	1'b0
[1]	R/W	**MSKRE**	Response error interrupt mask	1'b0
[0]	W	**RESERVED**	This bit must be '0'	1'b0
SDI COMMAND ARGUMENT REGISTER (CMDARG) ***Address : SD0 :C000_9828h/ SD1 : C000_C828h***				
[31:0]	R/W	**CMDARG**	Command argument to be passed to card	32'b0
SDI COMMAND REGISTER (CMD) ***Address : SD0 :C000_982Ch/ SD1 : C000_C82Ch***				
[31]	R/W	**STARTCMD**	Start command. Once command is taken by card interface part, this bit is cleared. When this bit is set, host should not attempt to write to any command registers. If write is attempted, hardware lock error is set in raw interrput status register.	1'b0
[30:24]	-	**RESERVED**	Reserved	7'b0
[23:22]	-	**RESERVED**	Must be '0'	2'b0
[21]	R/W	**UPDATECLKONLY**	Update clock registers only. When set, the controller does not send command and following register values are transferred into card clock domain: CLKDIV, CLKENA. 0 : Normal command sequence 1 : Update clock registers only	1'b0
[20:16]	R/W	**CARDNUM**	Card number in use. In POLLUX, this value must be 0.	5'b0
[15]	R/W	**SENDINIT**	If this bit is set, send 80 clocks to card for initialization before sending command to card. This bit should be set while sending first command to card. 0 : Do not send initialization sequence before sending command 1 : Send initialization sequence before sending command	1'b0
[14]	R/W	**STOPABORT**	Stop or Abort command control. 0 : Neither stop nor abort command to top current data transter in progress 1 : Stop or abort command intended to stop current data transfer in progress	1'b0
[13]	R/W	**WAITPRVDAT**	If this bit is set, wait for previous data transfer completion. 0 : Send command at once, even if previous data transfer has not completed. 1 : Wait for previous data transter completion before sending command	1'b0
[12]	R/W	**SENDAUTOSTOP**	When set, the controller sends stop command to sd card at end of data transfer. Don't care if no data expected. 0 : No stop command sent at end of data transfer 1 : Send stop command at end of data transfer	1'b0
[11]	R/W	**TRMODE**	Transfer mode. Don't care if no data expected. 0 : Block data transfer mode 1 : Stream data transfer mode	1'b0
[10]	R/W	**RW**	Read/Write mode selection. Don't care if no data expected. 0 : Read from card 1 : Write to card	1'b0
[9]	R/W	**DATEXP**	Data transfer expected flag. 0 : No data transfer expected 1 : Data transfer expected	1'b0
[8]	R/W	**CHKRSPCRC**	Check response CRC or not. Some of command responses do not return valid CRC bits. Software should disable CRC checks for those commands in order to disable CRC checking by controller. 0 : Do not check response CRC 1 : Check response CRC	1'b0
[7]	R/W	**RSPLEN**	Response length selection. 0 : Short response 1 : Long response	1'b0
[6]	R/W	**RSPEXP**	Response expected flag. 0 : No response expected 1 : Response expected	1'b0
[5:0]	R/W	**CMDINDEX**	Command index.	6'b0

Bit	R/W	Symbol	Description	Reset Value
SDI RESPONSE REGISTER 0 (RESP0) ***Address : SD0 :C000_9830h/ SD1 : C000_C830h***				
[31:0]	R	**RESPONSE0**	Bit [31:0] of response	32'b0
SDI RESPONSE REGISTER 1 (RESP1) ***Address : SD0 :C000_9834h/ SD1 : C000_C834h***				
[31:0]	R	**RESPONSE1**	Bit [63:32] of long response.	32'b0
SDI RESPONSE REGISTER 2 (RESP2) ***Address : SD0 :C000_9838h/ SD1 : C000_C838h***				
[31:0]	R	**RESPONSE2**	Bit [95:64] of long response.	32'b0
SDI RESPONSE REGISTER 3 (RESP3) ***Address : SD0 :C000_983Ch/ SD1 : C000_C83Ch***				
[31:0]	R	**RESPONSE3**	Bit [127:96] of long response.	32'b0
SDI MASKED INTERRUPT STATUS REGISTER (MINTSTS) ***Address : SD0 :C000_9840h/ SD1 : C000_C840h***				
[31:17]	-	**RESERVED**	Reserved	15'b0
[16]	R	**SDIOINT**	SDIO interrupt from card interrupt. Notice) MINTSTS = INTMASK & RINTSTS	1'b0
[15]	R	**EBEINT**	End bit error(read), Write no CRC (write) interrupt.	1'b0
[14]	R	**ACDINT**	Auto command done interrupt.	1'b0
[13]	R	**SBEINT**	Start bit error interrupt	1'b0
[12]	R	**HLEINT**	Hardware locked write error interrupt	1'b0
[11]	R	**FRUNINT**	FIFO underrun/overrun error interrupt	1'b0
[10]	R	**HTOINT**	Data starvation by host timeout interrupt	1'b0
[9]	R	**DRTOINT**	Data read timeout interrupt	1'b0
[8]	R	**RTOINT**	Response timeout interrupt	1'b0
[7]	R	**DCRCINT**	Data CRC error interrupt	1'b0
[6]	R	**RCRCINT**	Response CRC error interrupt	1'b0
[5]	R	**RXDRINT**	Receive FIFO data request interrupt	1'b0
[4]	R	**TXDRINT**	Transmit FIFO data request interrupt	1'b0
[3]	R	**DTOINT**	Data transfer over interrupt	1'b0
[2]	R	**CDINT**	Command done interrupt	1'b0
[1]	R	**REINT**	Response error interrupt	1'b0
[0]	-	**RESERVED**	Reserved	1'b0
SDI RAW INTERRUPT STATUS REGISTER (RINTSTS) ***Address : SD0 :C000_9844h/ SD1 : C000_C844h***				
[31:17]	-	**RESERVED**	Reserved	15'b0
[16]	R/W	**SDIOINT**	SDIO interrupt from card	1'b0
[15]	R/W	**EBE**	End bit error(read), Write no CRC (write) error status	1'b0
[14]	R/W	**ACD**	Auto command done status.	1'b0
[13]	R/W	**SBE**	Start bit error status	1'b0

Bit	R/W	Symbol	Description	Reset Value
[12]	R/W	**HLE**	Hardware locked write error status	1'b0
[11]	R/W	**FRUN**	FIFO underrun/overrun error status	1'b0
[10]	R/W	**HTO**	Data starvation by host timeout status	1'b0
[9]	R/W	**DRTO**	Data read timeout status	1'b0
[8]	R/W	**RTO**	Response timeout status	1'b0
[7]	R/W	**DCRC**	Data CRC error status	1'b0
[6]	R/W	**RCRC**	Response CRC error status	1'b0
[5]	R/W	**RXDR**	Receive FIFO data request status	1'b0
[4]	R/W	**TXDR**	Transmit FIFO data request status	1'b0
[3]	R/W	**DTO**	Data transfer over status	1'b0
[2]	R/W	**CD**	Command done status	1'b0
[1]	R/W	**RE**	Response error status	1'b0
[0]	-	**RESERVED**	Reserved	1'b0
SDI STATUS REGISTER (STATUS) ***Address : SD0 :C000_9848h/ SD1 : C000_C848h***				
[31]	R	**DMAREQ**	DMA request signal state	1'b0
[30]	R	**DMAACK**	DMA acknowledge signal state	1'b0
[29:22]	-	**RESERVED**	Reserved	8'b0
[21:17]	R	**FIFOCOUNT**	Number of filled locations in FIFO	5'b0
[16:11]	R	**RSPINDEX**	Index of previous response, including any auto-stop sent by core	6'b0
[10]	R	**FSMBUSY**	Data transmit or receive state-macine is busy	1'b0
[9]	R	**DATBUSY**	Selected card data busy	1'bx
[8]	R	**CPRESENT**	Card present flag	1'bx
[7:4]	R	**CMDFSM**	Command FSM states. 0 : Idle 1 : Send init sequence 2 : Tx cmd start bit 3 : Tx cmd tx bit 4 : Tx cmd index + arg 5 : Tx cmd crc7 6 : Tx cmd end bit 7 : Rx resp start bit 8 : Rx resp IRQ response 9 : Rx resp tx bit 10 : Rx resp cmd idx 11 : Rx resp data 12 : Rx resp crc7 13 : Rx resp end bit 14 : Cmd path with NCC 15 : Wait	4'b0
[3]	R	**FIFOFULL**	FIFO is full	1'b0
[2]	R	**FIFOEMPTY**	FIFO is empty	1'b1
[1]	R	**TXWMARK**	FIFO reached transmit watermark level	1'b1
[0]	R	**RXWMARK**	FIFO reached receive watermark level	1'b0
SDI FIFO THRESHOLD WATERMARK REGISTER (FIFOTH) ***Address : SD0 :C000_984Ch/ SD1 : C000_C84Ch***				
[31:20]	-	**RESERVED**	Reserved	12'b0
[19:16]	R/W	**RXTH**	FIFO threshold watermark level when receiving data from card. When FIFO data count reaches greater than this number, DMA/FIFO request is raised. During end of packet, request is generated regardless of threshold programming in order to complete any reamining data. In non-DMA mode, when receiver FIFO threshold (RXDR) interrupt is enabled, then interrupt is	4'hF

Bit	R/W	Symbol	Description	Reset Value
			generated instead of DMA request. During end of packet, interrupt is not generated if threshold programming is larger than any remaining data. It is responsibility of host to read remaining bytes on seeing Data Transfer Done interrupt. In DMA mode, at end of packet, even if remaining bytes are less than threshold, DMA request does single transfers to flush out any remaining bytes before Data Transfer Done interrupt is set. Limitation : RXTH <= FIFO Depth – 2 Recommended : (FIFO Depth/2) - 1	
[15:4]	-	**RESERVED**	Reserved	12'b0
[3:0]	R/W	**TXTH**	FIFO threshold watermark level when transmitting data to card. When FIFO data count is less than of equal to this number, DMA/FIFO request is raised . During end of packet, request or interrupt is generated regardless of threshold programming. In non-DMA mode, when transmit FIFO threshold interrupt is enabled, then interrupt is generated instead of DMA request. During end of packet, on last interrupt, host is responsible for filling FIFO with only required remaining bytes. In DMA mode, at end of packet, if last transfer is less than burst size, DMA controller does single cycles until required bytes are transferred. Limitation : TXTH >= 1 Recommended : FIFO Depth/2	3'b0
RESERVED ***Address : SD0 :C000_9850h~C000_9858h/ SD1 : C000_C850~C000_C858h***				
SDI TRANSFERRED CARD BYTE COUNT REGISTER (TCBCNT) ***Address : SD0 :C000_985Ch/ SD1 : C000_C85Ch***				
[31:0]	R	**TRCBCNT**	Number of bytes transferred by card interface part to card.	32'b0
SDI TRANSFERRED HOST BYTE COUNT REGISTER (TBBCNT) ***Address : SD0 :C000_9860h/ SD1 : C000_C860h***				
[31:0]	R	**TRFBCNT**	Number of bytes transferred between host/DMA memory and FIFO	32'b0
RESERVED ***Address : SD0 :C000_9864h/ SD1 : C000_C864h***				
[31:0]	-	**RESERVED**	Reserved	32'b0
RESERVED ***Address : SD0 :C000_9868h~C000_98FCh/ SD1 : C000_C868h~C000_C8FCh***				
SDI DATA REGISTER (DAT) ***Address : SD0 :C000_9900h/ SD1 : C000_C900h***				
[31:0]	R/W	**DATA**	Data write to or read from FIFO	32'bx
RESERVED ***Address : SD0 :C000_9904h~C000_99FC/ SD1 : C000_C904h~C000_C9FCh***				
RESERVED ***Address : SD0 :C000_9A00h/ SD1 : C000_CA00h***				
[31:0]	-	**RESERVED**	Reserved	32'b0
RESERVED ***Address : SD0 :C000_9A04h ~ C000_9FBCh/ SD1 : C000_CA04h ~ C000_CFBCh***				
SDI SYSTEM CLOCK ENABLE REGISTER (SYSCLKENB) ***Address : SD0 :C000_9FC0h/ SD1 : C000_CFC0h***				
[31:4]	-	**RESERVED**	Reserved	28'b0

Bit	R/W	Symbol	Description	Reset Value
[3]	R/W	**PCLKMODE**	PCLK operating mode. 0 : Disable 1 : Enable	1'b0
[2]	R/W	**CLKGENENB**	Clock generation enable 0 : Disable 1 : Enable	1'b0
[1:0]	-	**RESERVED**	Reserved	2'b0
SDI CLOCK GENERATOR REGISTER (CLKGEN) ***Address : SD0 :C000_9FC4h/ SD1 : C000_CFC4h***				
[31:7]	-	**RESERVED**	Reserved	22'b0
[6:4]	R/W	**CLKDIV0**	Clock divisor. Write a value of (actual divisor – 1)	3'b0
[3:1]	R/W	**CLKSRCSEL0**	Clock source selection. 0 : PCLK 1 : PLL0 2 : PLL1 else : N.A	3'b0
[0]	-	**RESERVED**	Reserved	1'b0

CHAPTER 14.

UART

14. UART

14.1. Overview

The POLLUX has four independent UART channels and supports Full Function (UART1) and Simple Function (UART0, UART2 and UART3).

Each channel for communication between the CPU and UART can operate in Interrupt-based mode or DMA-based mode. Communication via Interrupt or DMA should use FIFO mode and if Non-FIFO mode is used, only transmission in Polling mode is available. Each channel has two 16-byte FIFO registers for data transmission and reception and the data to be transmitted to/received from other devices to/from the POLLUX contain various attributes, such as Baud-rate, Stop bit and Parity check.

<Note>

The terms [Full/ Half/ Simple Function], are special terms used for the POLLUX Databook. Therefore, although this terminology is used consistently in this document, the meanings may differ from the use of the same terms by different companies. Here, Full Function indicates support for all signals that are used in the modem, that is to say RI, DCD, DSR, DTR, CTS and RTS as well as Tx and Rx. Half Function indicates that only Tx, Rx, CTS and RTS are supported. Finally, Simple Function indicates that only Tx and Rx are supported.

- Full Function : Tx, Rx, RI, DCD, DSR, DTR, CTS, RTS signal.
- Simple Function : Tx, Rx signal

14.1.1. Features

- UART0 ~ UART3 with DMA-Based or Interrupt-Based Operation.
- UART1 with Full modem function.
- Ability to add or delete standard asynchronous communications bits (Start, Stop and Parity) in the serial data.
- Independently controls transmission, reception, line status and data set interrupts.
- Programmable Baud Rate
- Modem control pins that allow flow control through software.
- Fully programmable serial interface :
  - 5, 6, 7, 8-bit characters
  - Even, Odd and no parity detection
  - 1 or 2 Stop bit generation
  - Baud rate generation up to around 3.1Mbps
- 16-Byte transmit FIFO
- 16-Byte receive FIFO
- Complete status reporting capability
- Ability to generate and detect line breaks
- Internal Diagnostic Capability that include : Loopback controls for communications link fault isolation
- Separates DMA requests for transmit and receive data services

14.1.2. Block Diagram

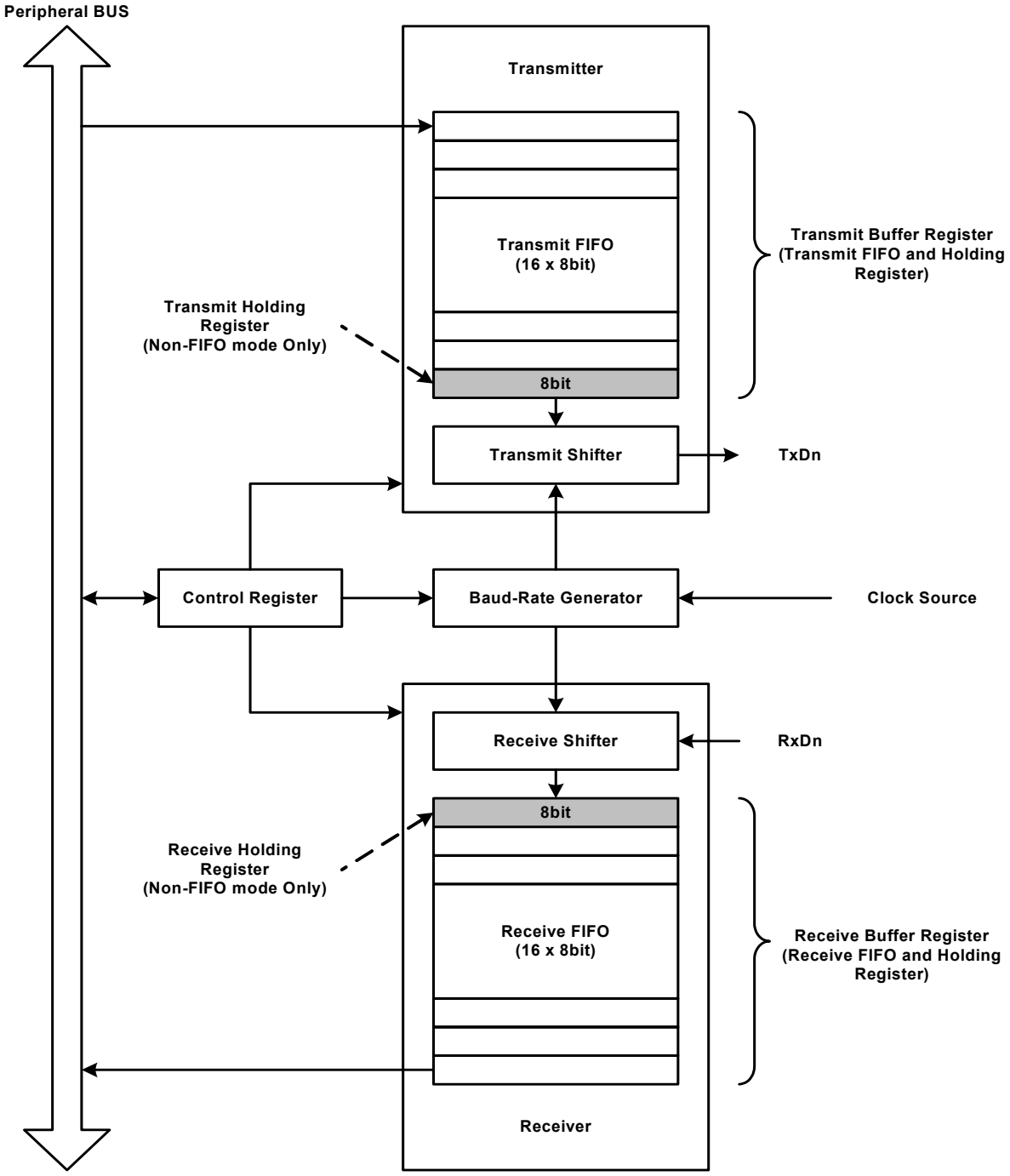


Figure 14-1. UART Block Diagram

Figure 14-1 shows one of the POLLUX's UART blocks. This figure shows a block diagram corresponding to a UART channel. As shown in the above figure, the UART block has 16-byte FIFO registers for each Tx and Rx and can set Stop bit, Parity bit, Baud-rate, etc. by using a control register. The block can use all 16-byte FIFO (as many as are set in the Trigger level) in FIFO mode, but can use only 1-byte (Transmit/Receive Holding register) in Non-FIFO mode. Transmit (Receive) Shifter performs the function of transmitting the data written in 1-byte by using a Tx (or Rx) port step by step.

14.2. Operation

The POLLUX has four UART Block channels and all channels can operate independently. Each independent UART channel has an individual Interrupt Pending register, so that any channel can generate an interrupt. The UART channels also provide functions, such as Auto Flow Control (AFC), Loop Back, UART Boot.

14.2.1. Data Transmission

A frame contains a Start bit, 5 to 8 Data bits, a Parity bit and one or two Stop bits for transmitting data. All bits in the frame can be controlled via a program and controlled by the Line Control register (LCON0 ~ LCON3). The Break conditions can also be produced by the transmitter. The serial output is forced by the break condition to a logic '0' state for one frame transmission time. The break signal is transmitted by this block after the present transmission word transmits perfectly. Data are continuously transmitted into the Tx FIFO after the break signal transmission.

<Note>

Break Condition : The break is defined as a continuous low level signal for one frame transmission time on the transmit data output.

14.2.2. Data Reception

As with data transmission, data frame for reception can be controlled via a program. The data frame contains a Start bit, 5 to 8 Data bits, a Parity bit and one or two Stop bits. In addition, an Error Flag is used to set the Overrun Error, Parity Error, Frame Error and Break Condition and these can be detected by using a receiver.

- If new data is overwritten before the previous data are read, an Overrun Error occurs.
- If unexpected Parity status is detected, a Parity Error occurs.
- If the received data does not contain the normal stop bit, a Frame Error occurs.
- If RxD holds the logical '0' for longer than the frame transmission time, a Break Condition occurs.

Receive Time out Condition occurs when the ***LCONn*** does not receive data during the value of timeout register word time(The setting of the Word Length bit is followed by this interval) and the RX FIFO is not empty in the FIFO mode.

14.2.3. Auto Flow Control (AFC)

The POLLUX UART blocks support an AFC function. Both nRTS pin and nCTS pin are used together in AFC and the AFC block diagram is as shown in the figure below:

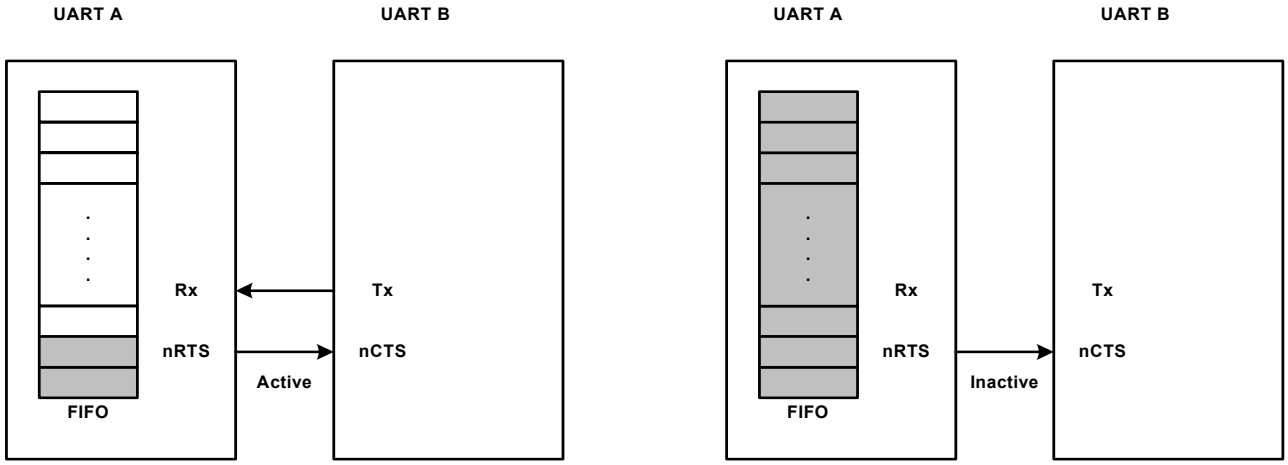


Figure 14-2. Auto Flow Control Interface

To use AFC mode, the ***MCON.AFC*** bit should be set as '1'.
If FIFO of UART A is not full, the Active signal ('0') is transmitted from nRTS pin of UART A to nCTS pin of UART B. If FIFO of UART A is full, the Inactive signal ('1') is transmitted from nRTS pin of UART A to nCTS pin of UART B.

14.2.4. Non-AFC

14.2.4.1. Rx Operation with FIFO

1) Select Receive Mode at UART A (Interrupt or DMA Mode)
2) The value of Rx FIFO count in ***FSTATUS*** Register is checked and the user must set the value of ***RTS_ACTIVE*** bit to '1' when the value is less than 15 and the user must set the value to '0'(Inactive nRTS) when the value is equal to or bigger than 15.
3) Repeat step 2.

14.2.4.2. Tx Operation with FIFO

1) Select Transmit Mode at UART B (Interrupt or DMA Mode)
2) The Value of CTS (***MSTATUS*** Register) is checked. If the value of ***MSTATUS.CTS*** is '1' (Active), UART B transmits data continuously.
3) If the value of ***MSTATUS.CTS*** becomes '0' (Inactive), data transmission is stopped. Otherwise, Step 2 is repeated.

14.2.5. Loop Back Mode

The POLLUX supports Loop Back mode to test a UART in a channel. The data transmitted via Tx is received in Rx of the same channel immediately. In other words, Tx and Rx are internally connected to each other in a UART block. Therefore, it is possible to test if the current UART port has a problem in Tx and Rx transmission. This Loop Back mode is executed by setting the ***UCONn.LOOPBACK_MODE*** bit as '1'.

14.2.6. Interrupt Generation

Each POLLUX UART channel has its own Interrupt Status register and the register contains ***INTSTATUSREG.TXPEND, INTSTATUSREG.RXPEND, INTSTATUSREG.ERRPEND, INTSTATUSREG.IRQTXENB, INTSTATUSREG.IRQRXENB, INTSTATUSREG.ERRENB*** and ***INTSTATUSREG.MENB*** bits.
Since an Interrupt Pending register exists in each UART channel in the POLLUX, the interrupt in each channel can be detected.

Type	FIFO Mode	Non-FIFO Mode
Rx Interrupt	Each time receive data reaches the trigger level of receive FIFO, the Rx Interrupt will be generated When the number of data in FIFO does not reaches Rx FIFO trigger Level and does not receive data during 3word time (This interval follows the setting of Word Length bit), the Rx interrupt will be generated (Receive time out)	Each time receive data becomes full, the receive holding register generates an interrupt
Tx Interrupt	Each time the transmit data reaches the trigger level of transmit FIFO(Tx FIFO trigger Level), the Tx Interrupt will be generated.	Each time the transmit data becomes empty, the transmit holding register generates an interrupt.
Error Interrupt	Frame error, Parity error and Break signal are detected and will generate an error interrupt. When it gets to the top of the receive FIFO without reading out data in it, the error interrupt will be generated(Overrun Error).	All errors generate an error interrupt immediately. However if another error occurs at the same time, only one interrupt is generated.

Table 14-1. Interrupts in connection with FIFO

14.2.7. DMA Request Generation

Data transmission through DMA requires the following prerequisites:

- DMA On
- FIFO On
- Trigger level decision by Rx and Tx of FIFO

The Tx/Rx trigger level can be set as Empty, 4-Byte, 8-Byte and 12-Byte. If there is no data to be transmitted in Tx, data are filled in the FIFO buffer in proportion to the trigger level. If, however, transmitted data are filled in Rx in proportion to the trigger level, Rx transmits the data by using DMA. In Rx operation, even if the size of the last data does not reach the trigger level, Rx automatically transmits the data after the time of the Time Out register (***TIMEOUTREG***) is passed.UART Error Status FIFO

14.2.8. UART Error Status FIFO

UART has the status FIFO in addition to the Rx FIFO register. The status FIFO indicates which data, among FIFO registers, is received with an error. Only when the error having data is ready to read out, the error interrupt will be issued. The ***RHBn***(Receive Buffer Register) Register with an error and ***ESTATUSn*** Register must be read out to clear the status FIFO. Even if the ***RHBn*** Register is read out instead of the ***ESTATUSn*** Register, the Error Status IO is cleared.
For example :
The fact that the UART FIFO receives A, B, C, D, E characters sequentially and the frame error occurs while receiving 'B' and the parity error occurs while receiving 'D' is assumed.

The error interrupt will not be generated even though the actual UART error occurred because the character, which was received with an error, has not been read yet. The error interrupt will be occurred when the character is read out.
The address of the Error Status FIFO has been synchronized with the address of the Rx-FIFO.

Time	Sequence Flow	Error Interrupt	Note
#0	When no Character is read out	-	-
#1	After 'A' is read out	The frame error(in B) interrupt occurs	The 'B' has to be read out
#2	After 'B' is read out	-	-
#3	After 'C' is read out	The frame error(in D) interrupt occurs	The 'D' has to be read out
#4	After 'D' is read out	-	-
#5	After 'E' is read out	-	-

Table 14-2. Error Interrupts Table

14.2.9. Baud-Rate Generation

Each UART's baud-rate generator provides the serial clock for transmitter and receiver. The source clock for the baud-rate generator can be selected with the POLLUX internal system clock. The baud-rate clock is generated by dividing the source clock by 16 and a 16-bit divisor specified in the UART baud-rate divisor register (***BRDn***). The ***BRDn*** can be determined as follows:

BRDn = (int)(UART Source Clock /(bps x 16)) - 1

where the divisor should be from 1 to (2^{16}- 1).

For example, if the baud-rate is 115200 bps and Source CLK is 96 MHz(PLL1), Divide = 1, ***BRDn*** is :

BRDn = (int)(48000000/(115200 x 16)) -1
= (int)(26.04) -1
= 26 -1
= 25.

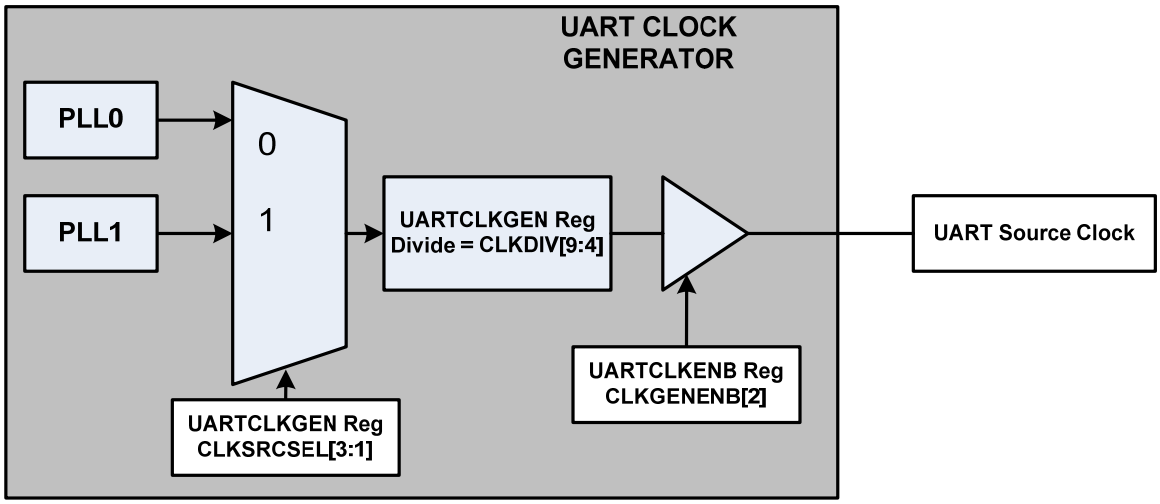


Figure 14-3. UART Clock Generator

All four UART channels have the same clock generator, as in Figure 14-3. UART can only use PLL1 and creates the UART source clock by using the Divider and the Clock Generator Enable. The value of the UART source clock can be set by using the UART CLOCK Enable register and the UART CLOCK Generate register.

14.2.10. UART Boot

The POLLUX supports UART Boot mode. The POLLUX can execute a UART Boot via the Rx pin of UART 0 from among the four UART channels. The UART Boot mode is set by the System Configuration pin and the POLLUX determines a variety of information including Booting mode and Memory size and type by using the values of the System Configuration pins set by an external device. (Refer to the POLLUX Application Note – SYSTEM CONFIGURATION document, for more detail information related to System Configuration settings).

Among the System Configuration pins, the CfgBOOTMODE[1:0] pins decide the Boot mode of the POLLUX. If the value of

CfgBOOTMODE[1:0] is set as '2', the POLLUX boots UART by using the UART Boot function. The setting of CfgBOOTMODE[1:0] is decided by an external resistance (pull-up or pull-down) connected to the POLLUX pins.

Only PLL1 is available for the initial booting and the value is 147.46 MHz. In addition, the clock divide value is '41'. Therefore, the value of the UART source clock in UART booting becomes 3.5966 MHz.

147.46M/ 41 = 3.5966MHz

<Note>

Refer to Chapter 2. I/O Description, for more detail information related to System Configuration setting).

14.2.11. SIR(Serial Infra-Red) Mode

SIR(serial infra-red) transmission and reception are supported by the POLLUX UART and can be selected by setting the SIR(Serial infra-red mode bit) in the UART Line Control Register and SIR signal control mode in the Serial Infra-red mode set register depending upon IrDA transceiver type.

Then the UART block operates in the normal Tx/ Rx mode. Figure 14-5 shows the implementation of the SIR mode.

The transmit period is pulsed at the normal serial transmit rate(when the transmit data bit is zero) of 3/16, in IR transmit mode. The receive must detect the 3/6 pulsed period to recognize a zero value, in the IR receive mode(refer to the frame timing diagrams shown in Figures 14-6 and 14-7, 14-8).

The SIR mode can be used in all UART channels; the mode can be set by using the ***LCONn.SIR_MODE*** bit of the corresponding Line Control register. As in Figure 14.4, the UART Block is to the left of the center dotted line. All UART Blocks have an SIR Tx Encoder and SIR Rx Decoder and have a ***LCONn.SIR_MODE*** bit to select them.

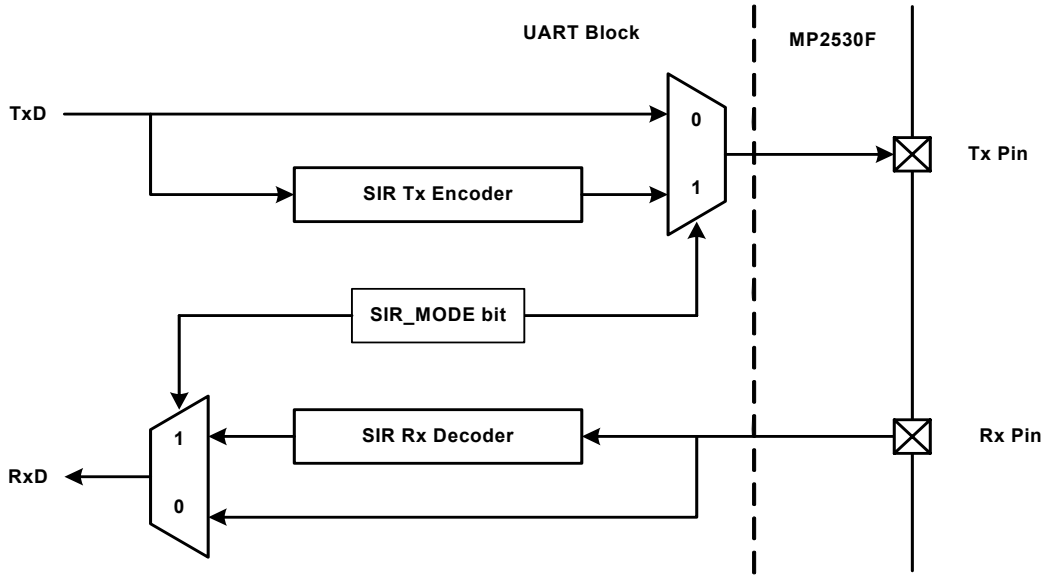


Figure 14-4. SIR Function Block Diagram

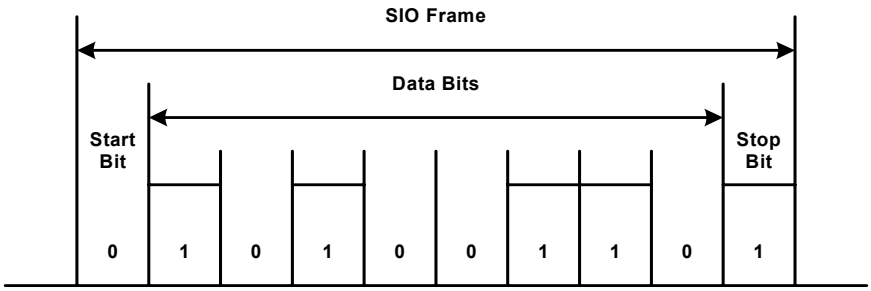


Figure 14-5. Serial I/O Frame Timing Diagram (Normal UART)

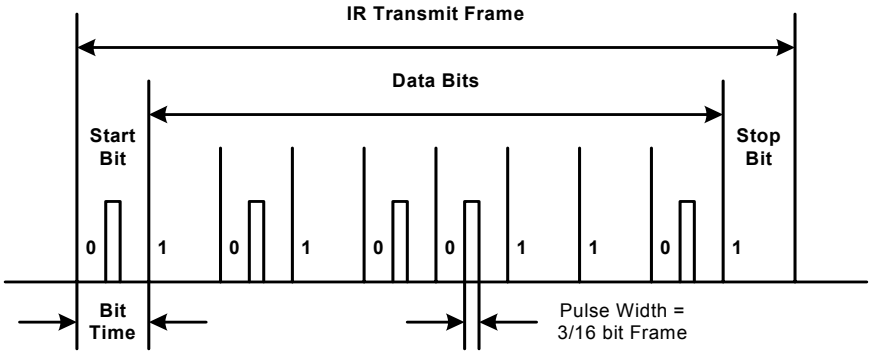


Figure 14-6. Intra-Red Transmit Mode Frame Timing Diagram

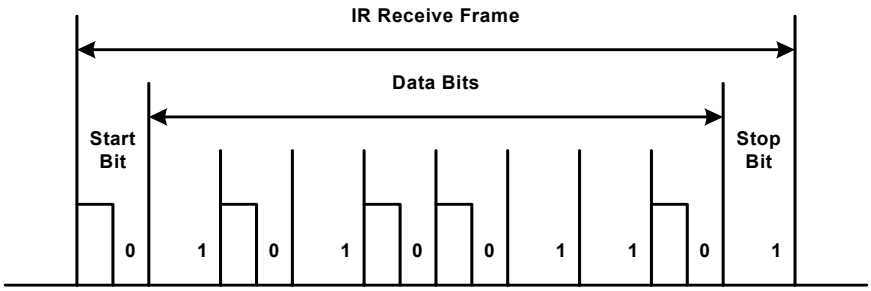


Figure 14-7. Infra-Red Receive Mode Frame Timing Diagram

14.2.12. Signal Description

Signal	GP-IO	Alt Function	I/O	Description
RX[0]			I	UART0 Receive Pin./UART Boot Pin
TX[0]	GPIOA[8]	ALT1	O	UART0 Transmit Pin.
nRTS[1]	GPIOA[10]	ALT1	O	UART1 Ready-To-Send Pin.
nCTS[1]	GPIOA[9]	ALT1	I	UART1 Clear-To-Send Pin.
nDTR[1]	GPIOA[14]	ALT1	O	UART1 Data-Terminal-Ready Pin.
nDSR[1]	GPIOA[13]	ALT1	I	UART1 Data-Set-Ready Pin.
nDCD[1]	GPIOA[12]	ALT1	I	UART1 Data-Carrier-Detect Pin.
nRI[1]	GPIOA[11]	ALT1	I	UART1 Ring Indicator Pin.
RX[1]	GPIOA[16]	ALT1	I	UART1 Receive Pin.
TX[1]	GPIOA[15]	ALT1	O	UART1 Transmit Pin.
RX[2]	GPIOA[18]	ALT1	I	UART2 Receive Pin.
TX[2]	GPIOA[17]	ALT1	O	UART2 Transmit Pin.
RX[3]	GPIOA[20]	ALT1	I	UART3 Receive Pin
TX[3]	GPIOA[19]	ALT1	O	UART3 Transmit Pin.

Table 14-3. Signal Description

14.3. Register Summary

Bit	R/W	Symbol	Description	Reset Value
LINE CONTROL REGISTER (LCON0, LCON1, LCON2, LCON3) ***Address :* UART0 *: C001_6000h/ UART1 : C001_6080h/ UART2 : C001_6800h/ UART3 : C001_6880h***				
[15 : 8]	-	**RESERVED**	Reserved	8'b0
[7]	R/W	**SYNC_PENDCLR**	This bit clears IRQ Pending signals (Rx and Tx) initially. <Note> Before using the interrupt of the UART block, this bit should be set as '1' after setting the Pending Clear bit of the Interrupt Status register as '1'. 0 : None. 1: Clear(Auto Clear)	1'b0
[6]	R/W	**SIR_MODE**	The SIR Mode determines whether or not to use the Infra-Red Mode 0 : Normal Mode(UART Mode) 1 : Infra Red Mode	1'b0
[5 : 3]	R/W	**PARITY_MODE**	The parity mode specifies how parity generation and checking are to be performed during the UART transmit and receive operation. 000 ~ 011 : No Parity 100 : Odd Parity 101 : Even Parity 110 : Parity forced/ checked as 1 111 : Parity forced/ checked as 0	3'b0
[2]	R/W	**STOPBIT**	The number of STOP bit. –Specifies how many stop bits are to be used to signal end-of-frame 0 : One stop bit per frame 1 : Two stop bit per frame	1'b0
[1 : 0]	R/W	**WORD_LEN**	The word length indicates the number of data bits to be transmitted or received per frame. 00 : 5-bits 01 : 6-bits 10 : 7-bits 11 : 8-bits	2'b0
CONTROL REGISTER (UCON0, UCON1, UCON2, UCON3) ***Address : UART0 : C001_6002h/ UART1 : C001_6082h/ UART2 : C001_6802h/ UART3 : C001_6882h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9]	R/W	**TX_INT**	Interrupt Request Type 0 : Pulse (Interrupt is requested as soon as Tx buffer becomes empty in Non-FIFO mode or Tx FIFO reaches Trigger Level in FIFO Mode) 1 : Level (Interrupt is requested while Tx buffer is empty in Non-FIFO mode or Tx FIFO reached in Trigger Level in FIFO mode) <Note> When the Interrupt Request Type is Level, this bit requests an interrupt in [High Level].	
[8]	R/W	**RX_INT**	Interrupt Request Type 0 : Pulse (Interrupt is requested as soon as Rx buffer becomes empty in Non-FIFO mode or Rx FIFO reaches Trigger Level in FIFO Mode 1 : Level (Interrupt is requested while Rx buffer is empty in Non-FIFO mode or Rx FIFO reached in Trigger Level in FIFO mode) <Note> When the Interrupt Request Type is Level, this bit requests an interrupt in [High Level].	1'b0
[7]	R/W	**RX_TIMEOUT**	Enable/ Disable RX time out interrupt when UART FIFO is enabled. The interrupt is a receive interrupt. 0 : Disable 1 : Enable	1'b0
[6]	R/W	**RX_ERRSTATUS**	This bit enables the UART to generate an interrupt if an exception, such as a break, frame error, parity error, or overrun error occurs during a receive operation. 0 : Do not Generate 1 : Generate	1'b0
[5]	R/W	**LOOPBACK_MODE**	Setting loop-back bit to 1 cause the UART to enter the loop-back mode. This mode is provided for test purposes only. 0 : Normal operation 1 : Loop-back mode	1'b0
[4]	R/W	**SEND_BREAK**	Setting this bit causes the UART to send a break during 1 frame time (auto-cleared after sending the break signal). 0 : Normal transmit 1 : Send break signal	1'b0
[3 : 2]	R/W	**TRANS_MODE**	These two bits determine which function is currently able to write TX data to the UART transmit buffer register. 00 : Disable 01 : Interrupt request or polling mode 10 : DMA request 11 : RESERVED	2'b0
[1 : 0]	R/W	**RECEIVE_MODE**	These two bits determine which function is currently able to read data from UART receive buffer register. 00 : Disable 01 : Interrupt request or polling mode	2'b0

Bit	R/W	Symbol	Description	Reset Value
			10 : DMA request 11 : RESERVED	
FIFO CONTROL REGISTER (FCON0, FCON1, FCON2, FCON3) ***Address : UART0 : C001_6004h/ UART1 : C001_6084h/ UART2 : C001_6804h/ UART3 : C001_6884h***				
[15 : 8]	-	**RESERVED**	Reserved	8'b0
[7: 6]	R/W	**TX_FIFO_TRIGGER**	These two bits determine the trigger level of transmit FIFO. 00 : empty 01 : 4-byte 10 : 8-byte 11 : 12-byte	2'b0
[5 : 4]	R/W	**RX_FIFO_TRIGGER**	These two bits determine the trigger level of receive FIFO. 00 : 1-byte 01 : 4-byte 10 : 8-byte 11 : 12-byte	2'b0
[3]	-	**RESERVED**	Reserved	1'b0
[2]	R/W	**TX_FIFO_RESET**	This bit is auto-cleared after resetting FIFO. 0 : Normal 1 : TX FIFO reset (Auto Clear)	1'b0
[1]	R/W	**RX_FIFO_RESET**	This bit is auto-cleared after resetting FIFO. 0 : Normal 1 : RX FIFO reset (Auto Clear)	1'b0
[0]	R/W	**FIFO_EN**	This bit decides the use of FIFO mode. If FIFO mode is in disable status, operation is allowed in only Polling mode. 0 : FIFO Disable 1 : FIFO Enable	1'b0
<Note> When the UART does not reach the FIFO trigger level and does not receive data during 3 word time in DMA receive mode with FIFO, the RX interrupt will be generated (receive timeout) and the users should check the FIFO status and read out the rest.				
MODEM CONTROL REGISTER (MCON0, MCON1) ***Address : UART0 : C001_6006h/ UART1 : C001_6086h/ UART2 : C001_6806h/ UART3 : C001_6886h***				
[15 : 8]	-	**RESERVED**	Reserved	8'b0
[7]	R/W	**HALF_CH_ENB**	This bit sets the operation of Full function / Half function. This setting is only applied in UART0 and UART1. 0 : Full UART 1 : Half UART	1'b0
[6]	R/W	**SCRXENB**	This bit enables ISO-7816 to receive. 0 : ISO7816 Tx 1 : UART / ISO7816 Rx	1'b1
[5]	R/W	**SCTXENB**	This bit enables ISO-7816 to transmit. 0 : UART / ISO7816 RX 1 : ISO7816 Tx	1'b1
[4]	R/W	**AFC**	This bit enables Auto Flow Control. 0 : Disable 1 : Enable	1'b0
[3 : 2]	-	**RESERVED**	Reserved. (These bits must be written to 0's)	1'b0
[1]	R/W	**DTR_ACTIVE**	Data Terminal Ready 0 : Inactive 1 : Active	1'b0
[0]	R/W	**RTS_ACTIVE**	If the AFC bit is enabled, this value will be ignored. In this case the POLLUX will control nRTS automatically. If the AFC bit is disabled, nRTS must be controlled by S/W. 0 : 'H' level(Inactivate nRTS pin) 1 : 'L' level(Activate nRTS pin)	1'b0
Tx/ Rx STATUS REGISTER (TRSTATUS0, TRSTATUS1, TRSTATUS2, TRSTATUS3) ***Address : UART0 : C001_6008h/ UART1 : C001_6088h/ UART2 : C001_6808h/ UART3 : C001_6888h***				
[15 : 3]	-	**RESERVED**	Reserved	13'b0
[2]	R	**TRANSMITTER_EMPTY**	This bit is automatically set to 1 when the transmit buffer register has no valid data to transmit and the transmit shift register is empty. 0 : Not empty 1 : Transmit buffer & shifter register empty	1'b1
[1]	R	**TRANSMIT_BUFFER_EMPTY**	This bit is automatically set to 1 when the transmit buffer register is empty. 0 : Not empty 1 : Transmit buffer Empty (In Non-FIFO mode, Interrupt or DMA is requested. In FIFO mode, Interrupt or DMA is requested when TX FIFO Trigger Level is set to 00 (Empty)) If the UART uses the FIFO, users should check TX FIFO Count bits and TX FIFO Full bit in the UFSTAT register instead of this bit.	1'b1
[0]	R	**RECEIVE_BUFFER_**	This bit is automatically set to 1 whenever the receive buffer register contains valid	1'b0

Bit	R/W	Symbol	Description	Reset Value
		DATA_READY	data received over the RXDn port. 0 = Empty 1 = The buffer register has a received data (In Non-FIFO mode, Interrupt or DMA is requested) If the UART uses the FIFO, users should check RX FIFO Count bits and RX FIFO Full bit in the UFSTAT register instead of this bit.	
ERROR STATUS REGISTER (ESTATUS0, ESTATUS1, ESTATUS2, ESTATUS3) ***Address:UART0 : C001_600Ah/ UART1 : C001_608Ah/ UART2 : C001_680Ah/ UART3 : C001_688Ah***				
[15 : 4]	-	**RESERVED**	Reserved	12'b0
[3]	R	**BREAK_DETECT**	This bit is automatically set to 1 to indicate that a break signal has been received. 0 : No break receive 1 : Break receive(Interrupt is requested)	1'b0
[2]	R	**FRAME_ERROR**	This bit is automatically set to 1 whenever a frame error occurs during receive operation. 0 : No frame error during receive 1 : Frame error(Interrupt is requested)	1'b0
[1]	R	**PARITY_ERROR**	This bit is automatically set to 1 whenever a parity error occurs during receive operation. 0 : No parity error during receive 1 : Parity error(Interrupt is requested)	1'b0
[0]	R	**OVERRUN_ERROR**	This bit is automatically set to 1 whenever an overrun error occurs during receive operation. 0 : No overrun error during receive 1 : Overrun error(Interrupt is requested)	1'b0
<Note> : These bits (ESTATUSn[3:0]) are automatically cleared to 0 when the UART error status register is read.				
FIFO STATUS REGISTER (FSTATUS0, FSTATUS1, FSTATUS2, FSTATUS3) ***Address:UART0 : C001_600Ch/ UART1 : C001_608Ch/ UART2 : C001_680Ch/ UART3 : C001_688Ch***				
[15:11]	-	**RESERVED**	Reserved	5'b0
[10]	R	**RX_FIFO_ERROR**	Error in RX FIFO Detect 0 : No Error Receive 1 : Error Receive	1'b0
[9]	R	**TX_FIFO_FULL**	This bit is automatically set to 1 whenever Transmit FIFO is full during transmit operation 0 : 0-byte ≤ TX FIFO Data ≤ 15-byte 1 : Full	1'b0
[8]	R	**RX_FIFO_FULL**	This bit is automatically set to 1 whenever Receive FIFO is full during receive operation 0 : 0-byte ≤ RX FIFO Data ≤ 15-byte 1 : Full	1'b0
[7 : 4]	R	**TX_FIFO_COUNT**	Number of data in TX FIFO	4'b0
[3 : 0]	R	**RX_FIFO_COUNT**	Number of data in RX FIFO	4'b0
MODEM STATUS REGISTER (MSTATUS0, MSTATUS1, MSTATUS2) ***Address:UART0 : C001_600Eh/ UART1 : C001_608Eh/ UART2 : C001_680Eh/ UART3 : C001_688Eh***				
[16 : 8]	-	**RESERVED**	Reserved	8'b0
[7]	R	**DELTA_DCD**	Delta DCD input 0 : Has not changed 1 : Has changed	-
[6]	R	**DELTA_RI**	Delta RI input 0 : Has not changed 1 : Has changed	-
[5]	R	**DELTA_DSR**	Delta DSR (Delta Set Ready) input 0 : Has not changed 1 : Has changed	-
[4]	R	**DELTA_CTS**	Delta CTS input. This bit indicates that the nCTS input to POLLUX has changed state since the last time it was read by the CPU 0 : Has not changed 1 : Has changed	-
[3]	R	**DCD**	DCD (Data Carrier Detect) Input 0 : nDCD pin is High 1 : nDCD pin is Low	-
[2]	R	**RI**	RI (Ring Indicator) Input 0 : nRI pin is High	-

Bit	R/W	Symbol	Description	Reset Value
			1 : nRI pin is Low	
[1]	R	**DSR**	DSR (Data Set Ready) Input 0 : nDSR pin is High 1 : nDSR pin is Low	-
[0]	R	**CTS**	CTS (Clear to Send) Input 0 : CTS signal is not activated (nCTS pin: High, Can't Receive Data) 1 : CTS signal is activated (nCTS pin: Low, Can Receive Data)	-
TRANSMIT BUFFER REGISTER (HOLDING REGISTER & FIFO REGISTER. THBn : THB0, THB1, THB2, THB3) ***Address : UART0 : C001_6010h/ UART1 : C001_6090h/ UART2 : C001_6810h/ UART3 : C001_6890h***				
[16 : 8]	-	**RESERVED**	Reserved	8'b0
[7 : 0]	W	**THBn**	Transmit data for UARTn	8'b0
RECEIVE BUFFER REGISTER (HOLDING REGISTER & FIFO REGISTER. RHBn : RHB0, RHB1, RHB2, RHB3) ***Address : UART0 : C001_6012h/ UART1 : C001_6092h/ UART2 : C001_6812h/ UART3 : C001_6892h***				
[15 : 8]	-	**RESERVED**	Reserved	8'b0
[7 : 0]	R	**RHBn**	Receive data for UARTn	8'b0
<Note> When an overrun error occurs, the RHBn must be read. If not, the next received data will also make an overrun error, even though the overrun bit of ESTATUSn had been cleared.				
THE VALUE STORED IN THE BAUD RATE DIVISOR REGISTER (BRDn : BRD0, BRD1, BRD2, BRD3) ***Address : UART0 : C001_6014h/ UART1 : C001_6094h/ UART2 : C001_6814h/ UART3 : C001_6894h***				
[15 : 0]	R/W	**BRDn**	Baud Rate division value BRDn = (int)(UART Source Clock /(bps x 16)) - 1 Note : BRDn > 0	-
RECEIVE TIMEOUT REGISTER (TIMEOUTREG0, TIMEOUTREG1, TIMEOUTREG2, TIMEOUTREG3) ***Address : UART0 : C001_6016h/ UART1 : C001_6096h/ UART2 : C001_6816h/ UART3 : C001_6896h***				
[15 : 0]	R/W	**TIMEOUTREGn**	Receive Time Out Period Value The value is set in bit units. Put the value of [the actual counter value-1] into the input value. If the Reset value is 0x001F, a Time Out interrupt occurs after transferring 32-bit	16'h1F
INTERRUPT CONTROL REGISTER (INTSTATUSREG0, INTSTATUSREG1, INTSTATUSREG2, INTSTATUSREG3) ***Address : UART0 : C001_6018h/ UART1 : C001_6098h/ UART2 : C001_6818h/ UART3 : C001_6898h***				
[15 : 8]	-	**RESERVED**	Reserved	8'b0
[7]	R/W	**MENB**	Modem Interrupt Enable 0 : Interrupt Disable 1 : Interrupt Enable	1'b0
[6]	R/W	**ERRENB**	Error Interrupt Enable 0 : Interrupt Disable 1 : Interrupt Enable	1'b0
[5]	R/W	**IRQRXENB**	Rx Interrupt Enable 0 : Interrupt Disable 1 : Interrupt Enable	1'b0
[4]	R/W	**IRQTXENB**	Tx Interrupt Enable 0 : Interrupt Disable 1 : Interrupt Enable	1'b0
[3]	R/W	**MPEND**	Indicate the Modem Interrupt request status. Set as '1' to clear this bit. Although MENB is disabled, this bit has effective value. Read > 0 : None 1 : Interrupt Pended Write > 0 : None 1 : Pending Clear	1'b0
[2]	R/W	**ERRPEND**	Indicate the Error Interrupt request status. Set as '1' to clear this bit. Although ERRENB is disabled, this bit has effective value. Read > 0 : None 1 : Interrupt Pended Write > 0 : None 1 : Pending Clear	1'b0
[1]	R/W	**RXPEND**	Indicate the Rx Interrupt request status. Set as '1' to clear this bit. Although IRQRXENB is disabled, this bit has effective value. Read > 0 : None 1 : Interrupt Pended Write > 0 : None 1 : Pending Clear	1'b0
[0]	R/W	**TXPEND**	Indicate the Tx Interrupt request status. Set as '1' to clear this bit. Although IRQTXENB is disabled, this bit has effective	1'b1

Bit	R/W	Symbol	Description	Reset Value
			value. Read > 0 : None 1 : Interrupt Pended Write > 0 : None 1 : Pending Clear	
UART CLOCK ENABLE REGISTER (UARTCLKENB0, UARTCLKENB1, UARTCLKENB2, UARTCLKENB3) ***Address : UART0 : C001_6040h/ UART1 : C001_60C0h/ UART2 : C001_6840h/ UART3 : C001_68C0h***				
[31 : 4]	-	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	PCLK operating mode 0 : Disable 1 : Enable	1'b1
[2]	R/W	**UARTCLKGENENB**	UART Clock Generation Enable 0 : Disable 1 : Enable	1'b1
[1 : 0]	-	**RESERVED**	Reserved	2'b11
UART CLOCK GENERATE REGISTER (UARTCLKGEN0, UARTCLKGEN1, UARTCLKGEN2, UARTCLKGEN3) ***Address : UART0 : C001_6044h/ UART1 : C001_60C4h/ UART2 : C001_6844h/ UART3 : C001_68C4h***				
[31 : 10]	-	**RESERVED**	Reserved	22'b0
[9 : 4]	R/W	**UARTCLKDIV**	UART Clock Divisor If the clock divisor is N, write an (N-1) value to the register. Default value of UART Clock Divisor is 40. This means that default vaule of this register is 39. 000000 ~ 111111(0 ~ 63) : 1 clock divider ~ 64 clock divider (N)	6'h27
[3 : 1]	R/W	**UARTCLKSRCSEL**	UART Clock Source Selection 000 : PLL0 001 : PLL1 010 : Not Used 011 ~ 111 : Not Used Note : Only PLL0 and PLL1 is available. The default value of PLL1's frequency is 147.456MHz and the default value of Uart Clock Divisor is 40. So UART is clocked by 3.6864MHz. See following equation. UART clock = (PLL clock) / (Uart Clock Divisor) = 147.456MHz / 40 = 3.6864MHz	1'b1
[0]	-	**RESERVED**	Reserved	1'b0

CHAPTER 15.

USB HOST/ DEVICE

15. USB HOST/ DEVICE

15.1. USB HOST

15.1.1. Overview

A 1-port USB host interface is supported by the POLLUX as follows:

- ✓ Open Rev.1.0 compatible.
- ✓ USB Rev. 1.1 compatible.
- ✓ 1 downstream ports.

Support for both LowSpeed and HighSpeed USB devices.

15.1.1.1. Block Diagram

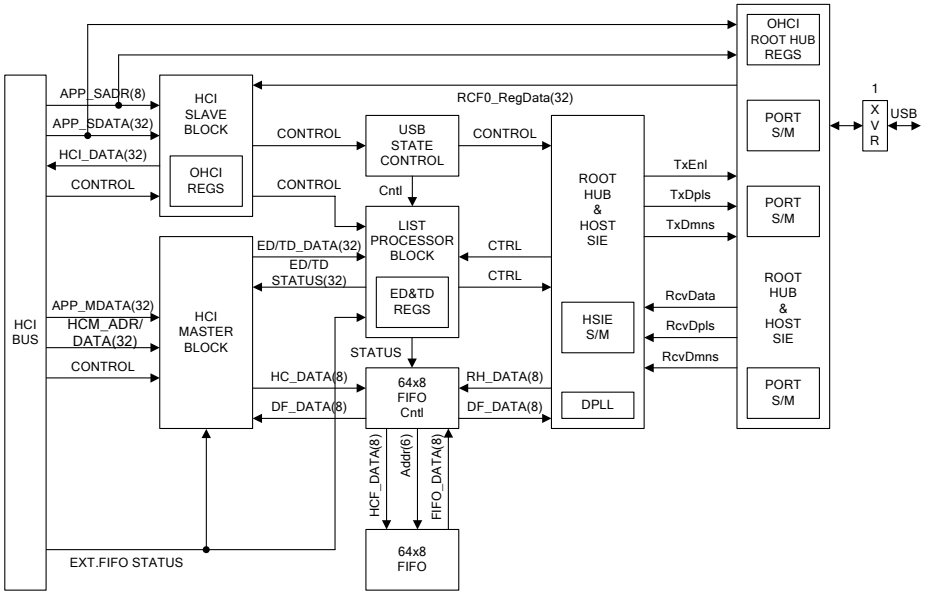


Figure 15-1. USB Host Controller Block Diagram

15.1.2. USB Host Clock Generator

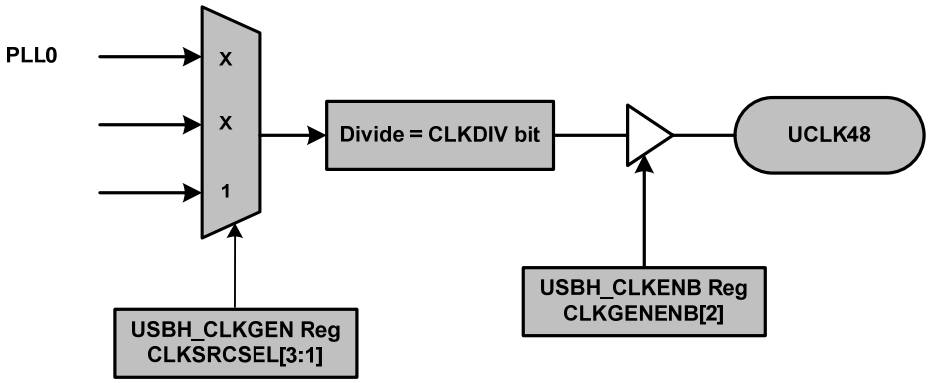


Figure 15-2. USB 1.1 Host Clock Generator

15.1.3. USB Init Config

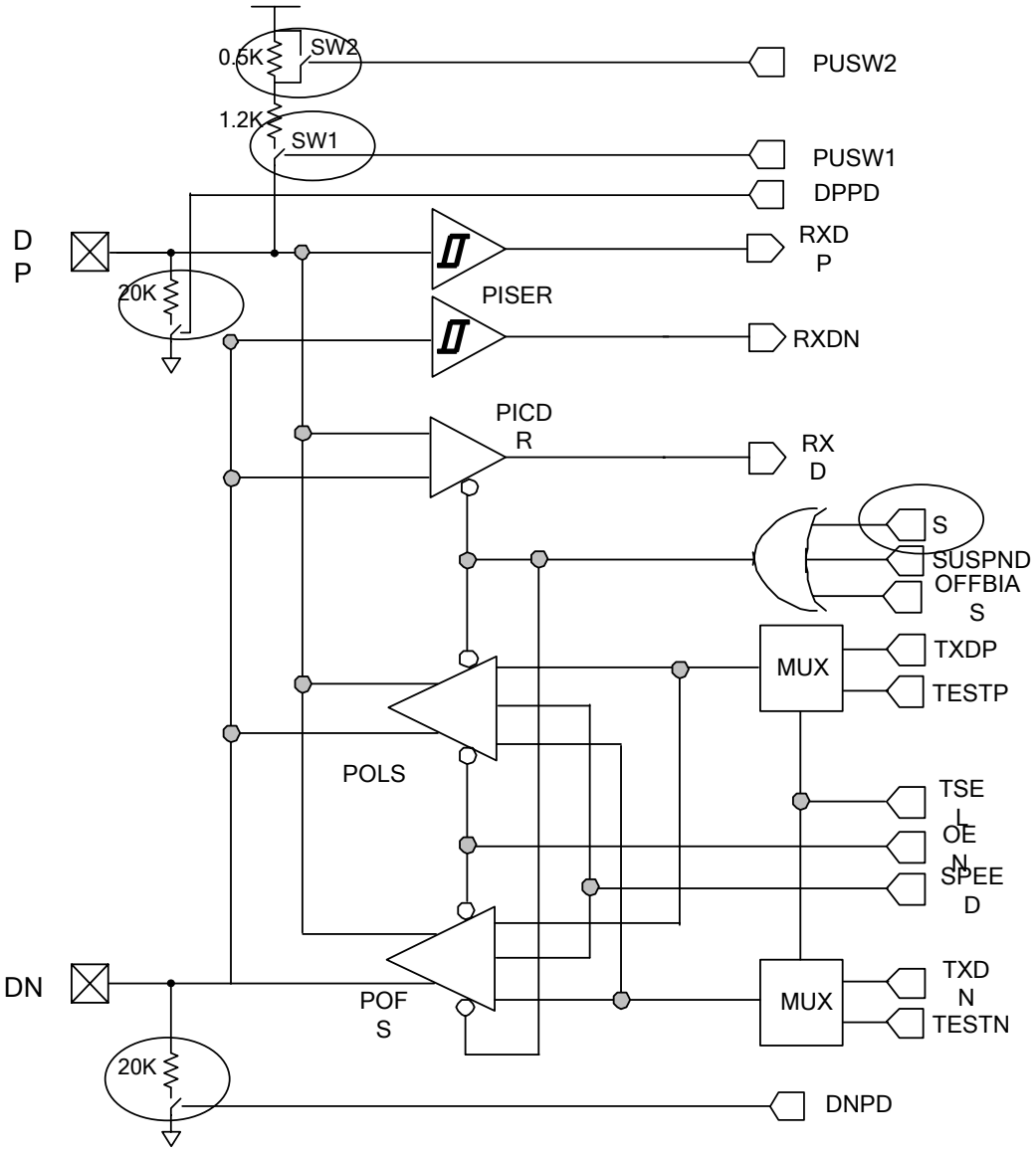


Figure 15-3. pin and control signals

Following table explains the operation for pull-down, pull-up switchs and S(sleep) signal as shown in figure 15-3.

Pin Name	In/Out	Description
DPPD	DI	Pull down of DP control; DPPD=1, DP is pulled down
DNPD	DI	Pull down of DN control; DNPD=1, DN is pulled down
PUSW1	DI	Pull up s/w 1 control; PUSW1=1, pull up s/w1 is on
PUSW2	DI	Pull up s/w 2 control; PUSW2=1, pull up s/w2 is on
S	DI	S=1, sleep mode S=0, working mode

15.1.4. USB HOST Control Registers Summary

The POLLUX USB Host controller meets OPEN HCI, Rev. 1.0. For detail information, refer to the Open Host Controller Interface, Rev. 1.0.

Register	Address	Description
HcRevision	0xC000_D000	Control and status group
HcControl	0xC000_D004	
HcCommonStatus	0xC000_D008	
HcInterruptStatus	0xC000_D00C	
HcInterruptEnable	0xC000_D010	
HcInterruptDisable	0xC000_D014	
HcHCCA	0xC000_D018	Memory pointer group
HcPeriodCurrentED	0xC000_D01C	
HcControlHeadED	0xC000_D020	
HcControlCurrentED	0xC000_D024	
HcBulkHeadED	0xC000_D028	
HcBulkCurrentED	0xC000_D02C	
HcDoneHead	0xC000_D030	
HcRmInterval	0xC000_D034	Frame counter group
HcFmRemaining	0xC000_D038	
HcFmNumber	0xC000_D03C	
HcPeriodicStart	0xC000_D040	
HcLSThreshold	0xC000_D044	
HcRhDescriptorA	0xC000_D048	Root hub group
HcRhDescriptorB	0xC000_D04C	
HcRhStatus	0xC000_D050	
HcRhPortStatus1	0xC000_D054	
HcRhPortStatus2	0xC000_D058	
Reserved	0xC000_D05C~0xC000_D07C	
HcInitCfg	0xC000_D080	USB Host Initial Config *refer to figure 15-3

DNPD	DPPD	PUSW2	PUSW1	SLEEP
4	3	2	1	0

0 : off 1 : on default : 5'b11001

USB Host CLOCK Enable REGISTER (USBH_CLKENB)

Address : C000_D0C0h

[15 : 4]	R/W	**RESERVED**	Reserved	12'b0
[3]	R/W	**PCLKMODE**	PCLK operating mode 0 : clock is enabled only when CPU accesses. 1 : Always	1'b0
[2]	R/W	**CLKGENENB**	Clock Generation Enable	1'b0
[1 : 0]	R/W	**USBH_CLKENB**	USBH_CLK Enable 00 : Disable 01 : Reserved 10 : Dynamic 11 : Always	2'b0

USB Host CLOCK Generate REGISTER (USBH_CLKGEN) *Address : C000_D0C2h*				
[15 : 10]	R/W	**RESERVED**	Reserved	6'b0
[9 : 4]	R/W	**CLKDIV**	Clock Divider Value. For 'N' clock divide, enter a value of 'N-1' 00h ~ 3Fh (N-1) : 1 to 64 clock divide (N clock divide)	6'b0
[3 : 1]	R/W	**CLKSRCSEL**	Clock Source Selection 000 : Not Used 001 : Not Used 010 : PLL2 Others : Not Used Note : Only PLL2 is available.	3'b0
[0]	R/W	**RESERVED**	Reserved	1'b0

Table 15-1. OHCI REGISTERs for USB Host Controller

15.2. USB Device Rev 2.0

15.2.1. Overview

The USB 2.0 Core is designed to aid the rapid implementation of the USB 2.0 peripheral device. The core supports both High and Full speed mode with 16-bit interface. The core can support up to 16 endpoints with programmable INTERRUPT, BULK and ISOCHRONOUS transfer mode. FIFO size for each endpoint can be easily configured.

15.2.1.1. Features

- ✓ Compliant to USB2.0 specification.
- ✓ Supports FS/HS dual mode operation.
- ✓ Conforms to UTMI(USB Transceiver Macrocell Interface)
- ✓ Easy FIFO size configuration.
- ✓ DMA interface capability.
- ✓ Support Interrupt, Bulk, Isochronous Transfer.

15.2.1.2. Block Diagram

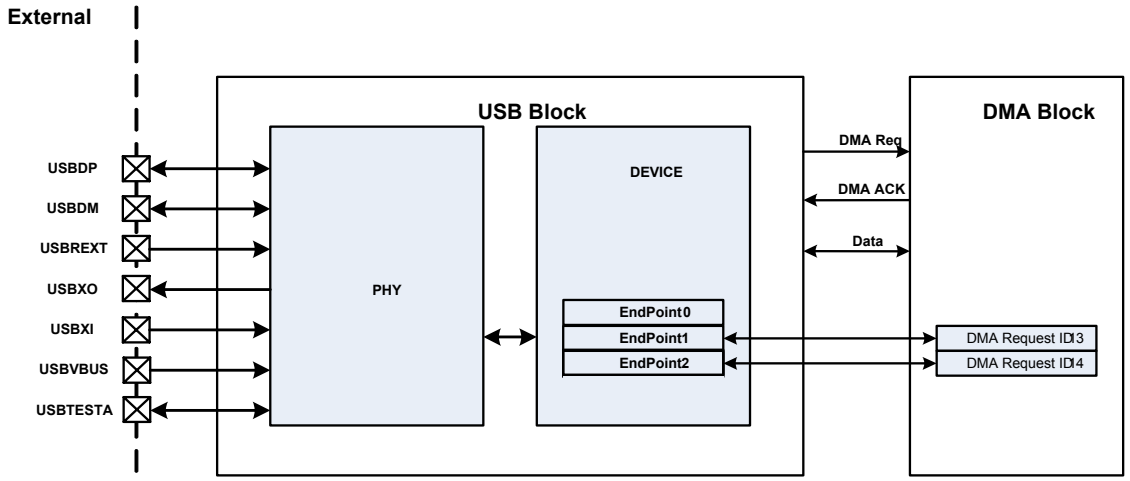


Figure 15-4. USB Device Rev 2.0 Block Diagram

Figure 15-2 shows a block diagram for the USB device Rev 2.0 Block to the POLLUX. Basically, the POLLUX performs high-speed communication with DMA. The use of DMA has the merit of reducing CPU load.

As shown in the above figure, the USB Block is roughly divided into a PHY block and a Device block. The PHY block deals with seven pins connected to the outside and data transmitted through USBDP and USBDM among these pins are written to a particular EndPoint in the Device block. The EndPoint consists of four channels and each channel is mapped to the corresponding DMA channel.

Note : Refer to Application Note.

15.2.2. Pin Function Description

Pin Name	GPIO No	GPIO Function	Type	Description
USBDP	-	-	I/O	Data Plus Signal from the USB Cable
USBDM	-	-	I/O	Data Minus Signal from the USB Cable
USBREXT	-	-	I	Connection to the external 3.4K-ohm(+/- 1%) resistor. The signal must not go though a series resistance in the pad (an ESD resistance is included in the macro). The pad should have ESD, PMOS and NMOS clamp devices. The 3.4K-ohm(+/- 1%) resistor must be referenced to the VSSA33C ground supply and placed as close as possible to the chip. Total capacitance should be less than 8pF, including board traces.
USBXO	-	-	O	Crystal Oscillator XO signal. If XO_EXT_CLK_ENBN is asserted, a 12MHz clock signal meeting the USB jitter specifications should be presented to this pin (< 100ps maximum jitter). If XO_ EXT_CLK_ENBN is deasserted, this signal is connected to one side of a 12MHz crystal.

Pin Name	GPIO No	GPIO Function	Type	Description
USBXI	-	-	I	Crystal Oscillator XI signal. If XO_EXT_CLK_ENBN is asserted, this pad is not needed. If XO_EXT_CLK_ENBN is deasserted, this signal is connected to one side of a 12MHz crystal. If a board clock is used, this pin should be tied to ground.
USBTESTA	-	-	I/O	Analog test input/ output. Voltage on this node is determined by the test interface mode and the VATEST_ENB signal. Depending on the production test strategy, this pin might not be required.
USBVBUS	-	-	I	This VBUS indicator signal indicates that the VBUS signal on the USB cable is active. For the serial interface, this signal controls the Pull-Up resistance on the D+ line in Device mode only. 1 : Pull-Up resistance on the D+ line is enabled based on the speed of operation 0 : Pull-Up resistance on the D+ line is disable.

Table 15-2 USB2.0 Pin Function Description

15.2.3. Detect VBUS

The USBVBUS pad is a logic-level input. When the USB2 PHY is operating as a device, The USBVBUS acts as a gating signal for many of the internal USB2 PHY modules. Therefore, it is important that transitions on the USBVBUS are clean and well-defined.

15.2.4. USB Device Clock Generator

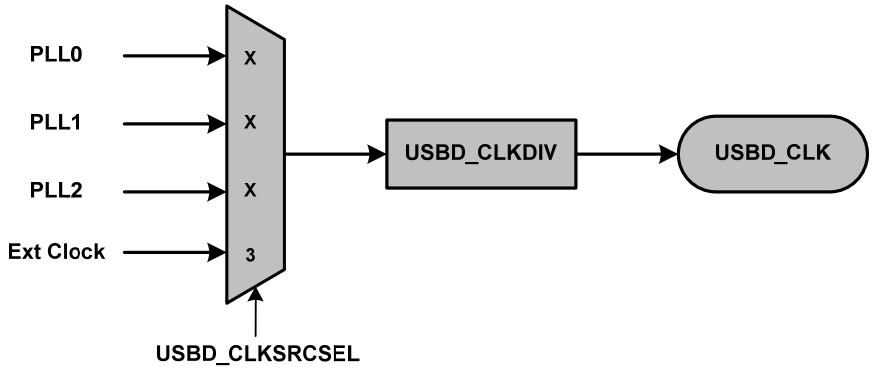


Figure 15-5. USB 2.0 Device Clock Generator

The POLLUX provides a clock generator for a USB 2.0 Device. The clock generator actually has four types of clock source, but can only use an external clock. The source clock can be selected by setting the ***USBD_CLKSRCSEL*** bit of the ***USBD_CLKGEN*** register. The selected 27MHz Ext Clock is divided by the ***USBD_CLKDIV*** bit in Divide. At this time, [value to divide – 1] is entered as an input value. In other words, enter [divide-by-1 – 1] = 0 for divide-by-1 and [divide-by-16 – 1] = 15 or 1111b for divide-by-16. Finally, enable BCLK by using the ***USBD_CLKENB*** bit of the ***USBD_CLKENB*** register. (The default value of the ***USBD_CLKENB*** bit is always enabled as 11b).

15.2.5. Flowchart

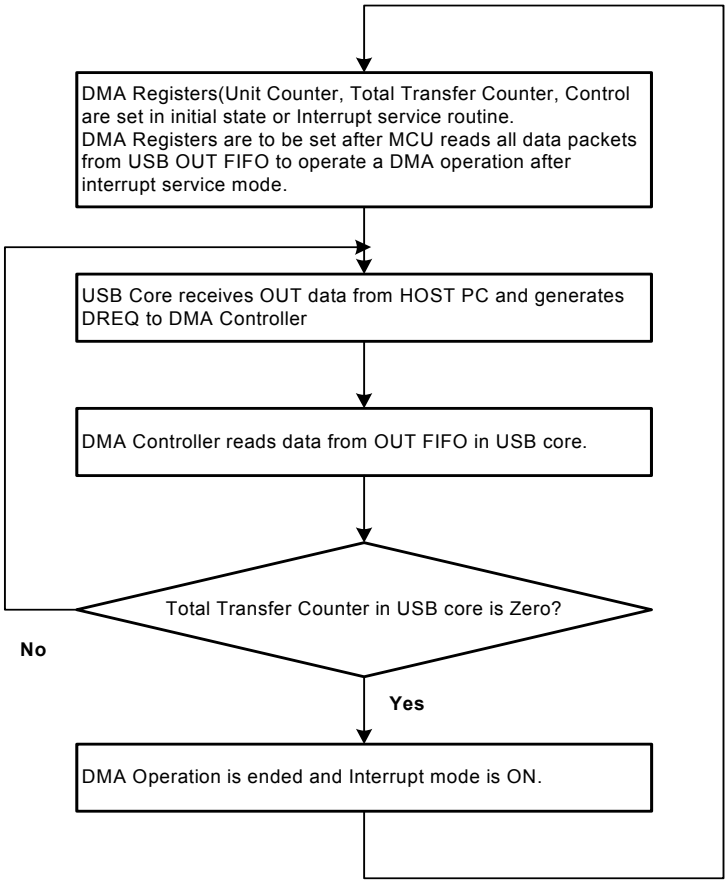


Figure 15-6. OUT DMA Operation Flow

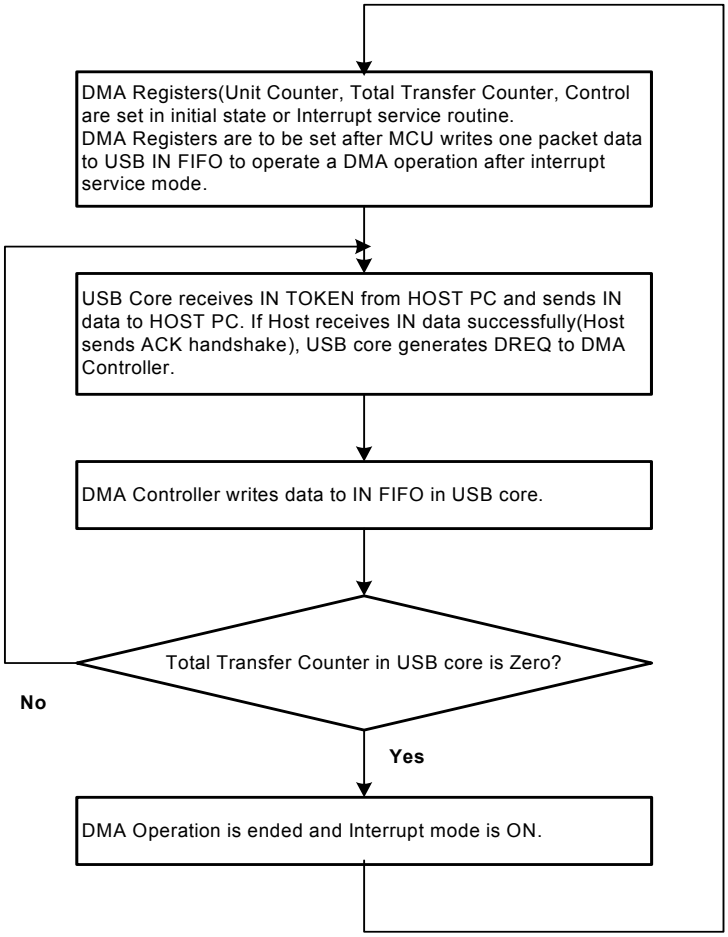


Figure 15-7. IN DMA Operation Flow

15.3. USB DEVICE 2.0 Register Summary

Bit	R/W	Symbol	Description	Reset Value
ENDPOINT INDEX REGISTER (IR) ***Address : C001_8000h***				
[15 : 3]	-	**RESERVED**	Reserved	13'b0
[2]	R/W	**INDEX[2]**	**Endpoint Number Select Register (Endpoint Number : 0 ~ 15)**	1'b0
[1]	R/W	**INDEX[1]**	0000 : Endpoint 0	1'b0
[0]	R/W	**INDEX[0]**	0001 : Endpoint 1 0010 : Endpoint 2 0011 ~1111 : Reserved	1'b0
ENDPOINT INTERRUPT REGISTER (EIR) ***Address : C001_8002h***				
[15 : 3]	-	**RESERVED**	Reserved	13'b0
[2]	R/W	**EP2INT**	Endpoint 2 Interrupt Flag Read > 0 : None 1 : EP2 Interrupt Pended Write > 0 : None 1 : EP2 Interrupt Pending Clear	1'b0
[1]	R/W	**EP1INT**	Endpoint 1 Interrupt Flag Read > 0 : None 1 : EP1 Interrupt Pended Write > 0 : None 1 : EP1 Interrupt Pending Clear	1'b0
[0]	R/W	**EP0INT**	Endpoint 0 Interrupt Flag Read > 0 : None 1 : EP0 Interrupt Pended Write > 0 : None 1 : EP0 Interrupt Pending Clear	1'b0
ENDPOINT INTERRUPT ENABLE REGISTER (EIR) ***Address : C001_8004h***				
[15 : 3]	-	**RESERVED**	Reserved	13'b0
[2]	R/W	**EP2INTENB**	Endpoint 2 Interrupt Flag 0 : EP2 Interrupt flag Disable 1 : EP2 Interrupt flag Enable	1'b0
[1]	R/W	**EP1INTENB**	Endpoint 1 Interrupt Flag 0 : EP1 Interrupt flag Disable 1 : EP1 Interrupt flag Enable	1'b0
[0]	R/W	**EP0INTENB**	Endpoint 0 Interrupt Flag 0 : EP0 Interrupt flag Disable 1 : EP0 Interrupt flag Enable	1'b0
FUNCTION ADDRESS REGISTER (FAR) ***Address : C001_8006h***				
[15 : 7]	-	**RESERVED**	Reserved	9'b0
[6]	R	**FNCADDR6**	MCU can read a unique USB Function Address from this register. The address is transferred from USB Host through set_address command.	1'b0
[5]	R	**FNCADDR5**		1'b0
[4]	R	**FNCADDR4**		1'b0
[3]	R	**FNCADDR3**		1'b0
[2]	R	**FNCADDR2**		1'b0
[1]	R	**FNCADDR1**		1'b0
[0]	R	**FNCADDR0**		1'b0
FRAME NUMBER REGISTER (FNR) ***Address : C001_8008h***				

Bit	R/W	Symbol	Description	● Reset Value
[15]	-	**RESERVED**	Reserved	-
[14]	R	**FRMTIMELOCK**	Frame Timer Lock.(FTL) FTL is activated when the device frame timer is locked to the host frame timer. When this bit is set, Frame Number is valid. This bit is set by USB after device receives two valid SOF.	-
[13]	R	**SM**	SOF Missing SM is activated when frame timer locking between device frame timer and Host frame timer fails	-
[12:11]	-	**RESERVED**	Reserved	-
[10]	R	**FN[10]**	[10 : 0] bits store the frame count number, which increases per every SOF packet.	-
[9]	R	**FN[9]**		-
[8]	R	**FN[8]**		-
[7]	R	**FN[7]**		-
[6]	R	**FN[6]**		-
[5]	R	**FN[5]**		-
[4]	R	**FN[4]**		-
[3]	R	**FN[3]**		-
[2]	R	**FN[2]**		-
[1]	R	**FN[1]**		-
[0]	R	**FN[0]**		-
ENDPOINT DIRECTION REGISTER (EDR) ***Address : C001_800Ah***				
[15]	R/W	**EP15DS**	Endpoint 15 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[14]	R/W	**EP14DS**	Endpoint 14 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[13]	R/W	**EP13DS**	Endpoint 13 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[12]	R/W	**EP12DS**	Endpoint 12 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[11]	R/W	**EP11DS**	Endpoint 11 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[10]	R/W	**EP10DS**	Endpoint 10 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[9]	R/W	**EP9DS**	Endpoint 9 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[8]	R/W	**EP8DS**	Endpoint 8 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[7]	R/W	**EP7DS**	Endpoint 7 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[6]	R/W	**EP6DS**	Endpoint 6 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[5]	R/W	**EP5DS**	Endpoint 5 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[4]	R/W	**EP4DS**	Endpoint 4 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0

Bit	R/W	Symbol	Description	● Reset Value
[3]	R/W	**EP3DS**	Endpoint 3 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[2]	R/W	**EP2DS**	Endpoint 2 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[1]	R/W	**EP1DS**	Endpoint 1 Direction Select 0 : Rx Endpoint 1 : Tx Endpoint	1'b0
[0]	-	**RESERVED**	Reserved	1'b0
TEST REGISTER (TR) ***Address : C001_800Ch***				
[15]	R	**VBUS**	VBUS signal status 0 : VBUS Off 1 : VBUS On	1'b0
[14]	-	**RESERVED**	Reserved	1'b0
[13]	R/W	**EUERR**	EB UNDERRUN Error. If error interrupt enable bit of SCR register is set to '1', EUERR is set to '1' when EB underrun error in transceiver is detected.	1'b0
[12]	R/W	**PERR**	PID Error. If error interrupt enable bit of SCR register is set to '1', PERR is set to '1' when PID error is detected.	1'b0
[11]	R/W	**FDWR**	FIFO Direct Write Read Enable 0 : Disable, 1 : Enable	1'b0
[10 : 8]	-	**RESERVED**	Reserved	7'b0
[7:6]	R/W	**SPDSEL**	SPEED SELECT MODE 00 : Normal Operration 01 : High speed Fixed. 10 : Full speed Fixed. 11 : Reserved.	2'b0
[5]	-	**RESERVED**	Reserved	-
[4]	R/W	**TMD**	Test Mode When TMP is set to '1', the core is forced into the test mode. Following TPS, TKS, TJS, TSNS bit are meaningful in test mode.	1'b0
[3]	R/W	**TPS**	Test Packets. If this bit is set, the USB repetitively transmit the test packets to Host. This bit can be set when TMD bit is set.	1'b0
[2]	R/W	**TKS**	Test K select If this bit is set, the transceiver port enters into the high speed K state. This bit can be set when TMD bit is set.	1'b0
[1]	R/W	**TJS**	Test J select If this bit is set, the transceiver port enters into the high speed J state. This bit can be set when TMD bit is set.	1'b0
[0]	R/W	**TSNS**	Test SE0 NAK Select. If this bit is set, the transceiver enters into the high speed receive mode and must respond to any IN token with NAK handshake. This bit can be set when TMD bit is set.	1'b0
SYSTEM STATUS REGISTER (SSR) ***Address : C001_800Eh***				
[15]	R/C	**BAERR**	Byte Align Error If error interrupt enable bit of SCR register is set to '1', BAERR is set to '1' when Byte Alignment Error is detected 0 : No Error 1 : Error Detected	1'b0

Bit	R/W	Symbol	Description	● Reset Value
[14]	R/C	**TMERR**	Timeout Error If error interrupt enable bit of SCR register is set to '1', TMERR is set to '1' when Timeout Error is detected 0 : No Error 1 : Error Detected	1'b0
[13]	R/C	**BSERR**	Bit Stuff Error If error interrupt enable bit of SCR register is set to '1', BSERR is set to '1' when Bit Stuff Error is detected 0 : No Error 1 : Error Detected	1'b0
[12]	R/C	**TCERR**	Token CRC Error If error interrupt enable bit of SCR register is set to '1', TCERR is set to '1' when CRC Error in Token Packet is detected 0 : No Error 1 : Error Detected	1'b0
[11]	R/C	**DCERR**	Data CRC Error If error interrupt enable bit of SCR register is set to '1', DCERR is set to '1' when CRC Error in Data Packet is detected 0 : No Error 1 : Error Detected	1'b0
[10]	R/C	**EOERR**	EB OVERRUN Error If error interrupt enable bit of SCR register is set to '1', EOERR is set to '1' when EB OVERRUN Error in Transceiver is detected 0 : No Error 1 : Error Detected	1'b0
[9]	R/C	**VBUSOFF**	VBUS Off If VBUS Off Interrupt enable bit of SCR register is set to '1', VBUSOFF is set to '1' When VBUS is Low. 0 :	1'b0
[8]	R/C	**VBUSON**	VBUS On If VUS interrupt enable bit of SCR set to '1', VBUSON is set to '1' when VBUS is high.	1'b0
[7]	R/C	**TBM**	Toggle Bit Mismatch. If error interrupt enable bit of SCR register is set to '1', TBM is set to '1' when Toggle mismatch is detected.	1'b0
[6]	R	**DP**	DP Data Line State. DP informs the status of D+ Line.	1'b0
[5]	R	**DM**	DM Data Line State. DM informs the status of D- Line.	1'b0
[4]	R	**HSP**	Host Speed 0 : Full Speed 1 : High Speed	1'b0
[3]	R/C	**SDE**	Speed Detection End. SDE is set by the core when the HS Detect Handshake process is ended.	1'b0
[2]	R/C	**HFRM**	Host Forced Resume. HFRM is set by the core in suspend state when Host sends resume signaling.	1'b0
[1]	R/C	**HFSUSP**	Host Forced Suspend. HFSUSP is set by the core when the SUSPEND signaling from host is detected.	1'b0
[0]	R/C	**HFRES**	Host Forced Reset. HFRES is set by the core when the RESET signaling from host is detected.	1'b0
SYSTEM CONTROL REGISTER (SCR) ***Address : C001_8010h***				
[15]	-	**RESERVED**	Reserved	1'b0
[14]	R/W	**DTZIEN**	DMA Total Counter Zero Interrupt Enable. When set to '1' DMA Total Counter Zero Interrupt is generated. 0 : Disable 1 : Enable	1'b0
[13]	-	**RESERVED**	Reserved	1'b0

Bit	R/W	Symbol	Description	● Reset Value
[12]	R/W	**DIEN**	DUAL Interrupt Enable. When set to '1', Interrupt is activated until Interrupt source is cleared. 0 : Disable 1 : Enable	1'b0
[11]	R/W	**VBUSOFFEN**	VBUS Off Enable 0 : Disable 1 : Enable	1'b0
[10]	R/W	**VBUSONEN**	VBUS On Enable 0 : Disable 1 : Enable	1'b0
[9]	R/W	**RWDE**	Reserve Write Data Enable. 0 : Low Byte data(MCU_DATA[7 : 0]) is first sent to Host. 1 : High Byte data(MCU_DATA[15 : 8]) is first sent to Host.	1'b0
[8]	R/W	**EIE**	Error Interrupt Enable. This bit must be set to '1' to enable error interrupt.	1'b0
[7]	R/W	**BIS**	Bus Interface Select. The MCU bus width is selected by BIS. 0 : bus width is 8bit. 1 : bus width is 16bit	1'b0
[6]	R/W	**SPDEN**	Speed Detect End Interrupt Enable. When set to '1', Speed Detection Interrupt is generated. 0 : Disable 1 : Enable	1'b0
[5]	R/W	**RRDE**	Reverse Read Data Enable 0 : First received byte is loaded in High byte field (MCU_DATA[15 : 8]) 1 : First received byte is loaded in Low byte field (MCU_DATA[7 : 0])	1'b0
[4]	R/W	**IPS**	Interrupt Polarity Select. The signal polarity of the interrupt from the core is changed through IPS. 0 : The Interrupt is considered to be active in low state. 1 : The interrupt is considered to be active in High state.	1'b0
[3]	R/W	**SPDC**	Speed detection Control 0 : Enable 1 : Disable	1'b0
[2]	R/W	**MFRM**	Resume by MCU. If this bit is set, the suspended core generates a resume signal. 0 : Cleared 1 : Resumed	1'b0
[1]	R/W	**HSUSPE**	Suspend Enable. When set to '1', core can respond to the Suspend signaling by USB host.	1'b0
[0]	R/W	**HRESE**	Reset Enable. When set to '1', core can respond to the Reset signaling by USB host.	1'b0
EP0 STATUS REGISTER (EP0SR) ***Address : C001_8012h***				
[15 : 7]	-	**RESERVED**	Reserved	9'b0
[6]	R	**LWO**	Last Word Odd LWO informs that the last word of a packet in FIFO has an invalid upper byte. This bit is cleared automatically after the MCU reads it from the FIFO.	1'b0
[5]	-	**RESERVED**	Reserved	1'b0
[4]	R/C	**SHT**	Stall Handshake Transmitted SHT informs that STALL handshake due to stall condition is sent to Host. This bit is an interrupt source. This bit is cleared when the MCU writes '1'.	1'b0
[3 : 2]	-	**RESERVED**	Reserved	2'b0
[1]	R/C	**TST**	Tx Successfully Transmitted. TST is set by core after core sends TX data to Host and receives ACK successfully. TST is one of the interrupt sources.	1'b0
[0]	R/C	**RSR**	Rx Successfully Received. RSR is set by core after core receives error free packet from Host and sent ACK back to Host successfully. RSR is one of the Interrupt sources.	1'b0

Bit	R/W	Symbol	Description	● Reset Value
EP0 CONTROL REGISTER (EP0CR) ***Address : C001_8014h***				
[15 : 4]	-	**RESERVED**	Reserved	12'b0
[3]	R/W	**TTE**	Tx Test Enable. This bit can be used for Test. TTE enables MCU to control Tx data toggle bit and Tx zero length set bit. The Tx data toggle bit and Tx zero length set bit do not need to be set in normal operation. 0 : Disable 1 : Enable	1'b0
[2]	R/W	**TTS**	Tx Toggle Set. TTS is useful for Test. TTS is used to set or reset data PID. TTS can be managed when Tx Test Enable(TTE) bit is set. 0 : Data PID 0 1 : Data PID 1	1'b0
[1]	R/W	**ESS**	Endpoint Stall Set. Ess is set by MCU when it intends to send STALL handshake to Host. This bit is cleared when the MCU writes '0' on it. ESS is needed to be set '0' after MCU writes '1' on it.	1'b0
[0]	R/W	**TZLS**	Tx Zero Length Set. TZLS is set by MCU when it intends to send Tx zero length data to Host. TZLS can be managed when Tx Test Enable bit is set. This bit is cleared when the MCU writes '0' on it.	1'b0
BUFFER REGISTER (BR) ***Address : C001_8030h ~ C001_804Eh***				
[15 : 0]	R/W	**-**	Buffer register holds Tx/ Rx data between MCU and the core.	16'bX
ENDPOINT STATUS REGISTER (BSR) ***Address : C001_8016h***				
[15]	R/C	**FUDR**	FIFO Underflow. FUDR is only used for ISO mode. FUDR is set when FIFO is empty and Host sends IN token. This bit is cleared when the MCU writes '1'	1'b0
[14]	R/C	**FOVF**	FIFO Overflow. FOVF is only used for ISO mode. FOVF is set when FIFO is full and Host sends OUT data. This bit is cleared when the MCU writes '1'	1'b0
[13 : 12]	-	**RESERVED**	Reserved	2'b0
[11]	R/W	**FPID**	First OUT Packet Interrupt Disable in OUT DMA operation. First Received OUT packet generates interrupt if this bit is disabled and DEN in DMA control register is enabled. 0 : Disable 1 : Enable	1'b0
[10]	R/C	**OSD**	OUT Start DMA. OSD is set when First OUT packet is received after Registers related DMA operation are set.	1'b0
[9]	R/C	**DTCZ**	DMA Total Count Zero. DTCZ is set when DMA total counter reach to '0'. This bit is cleared when the MCU writes '1' on it.	1'b0
[8]	R/C	**SPT**	Short Packet Received. SPT informs that OUT endpoint receives short packet during OUT DMA operation. This bit is cleared when the MCU writes '1' on it.	1'b0
[7]	R	**DOM**	Dual Operation Mode. DOM is set when the max packet size of corresponding endpoint is equal to a half FIFO size. This bit is read only. Endpoint 0 does not support dual mode.	1'b0
[6]	R/C	**FFS**	FIFO Flushed.	1'b0

Bit	R/W	Symbol	Description	● Reset Value
			FFS informs that FIFO is flushed. This bit is an interrupt source. This bit is cleared when the MCU clears FLUSH bit in Endpoint Control Register.	
[5]	R/C	**FSC**	Functional Stall Condition. FSC informs that STALL handshake due to functional stall condition is sent to Host. This bit is set when endpoint stall set bit is set by the MCU. This bit is cleared when the MCU writes '1' on it.	1'b0
[4]	R	**LWO**	Last Word Odd. LWO informs that the lower byte of last word is only valid. This bit is automatically cleared after the MCU reads packet data received Host.	1'b0
[3 : 2]	R/C	**PSIF**	Packe Status In FIFO. 00 : No packet in FIFO 01 : One packet in FIFO 10 : Two packets in FIFO 11 : invalid value.	2'b0
[1]	R/C	**TPS**	Tx Packet Success. RPS is used for Single or Dual transfer mode. TPS is activated when one packet data in FIFO was successfully transferred to Host and received ACK from Host. This bit should be cleared by writing '1' on it after being read by the MCU.	1'b0
[0]	R	**RPS**	Rx Packet Success. RPS is used for Single or Dual transfer mode. RPS is activated when the FIFO has a packet data to receive. RPS is automatically cleared when MCU reads all packet(one or two) from FIFO. MCU can identify the packet size through BYTE READ COUNT REGISTER (BRCR).	1'b0
ENDPOINT CONTROL REGISTER (ECR) ***Address : C001_8018h***				
[15]	R/W	**SRE**	Software Reset Enable 0 : Disable 1 : Enable	2'b0
[14:13]		**RESERVED**		2'b0
[12]	R/W	**INPKTHLD**	The MCU can control Tx FIFO status through this bit. If this bit is set to one, USB does not send IN data to Host. 0 : The USB can send IN data to Host according to IN FIFO status (Normal Operation) 1 : The USB sends NAK handshake to Host regardless of IN FIFO status.	1'b0
[11]	R/W	**OUTPKTHLD**	The MCU can control Rx FIFO status through this bit. If this bit is set to one, USB does not accept OUT data from Host. 0 : The USB can accept OUT data from Host according to OUT FIFO status(Normal Operation). 1 : The USB does not accept OUT data from Host.	1'b0
[10]	R/W	**TNPMF_1**	Transaction Number Per Micro Frame. TNPMF is useful for ISO transfer. 00 : Invalid value. 01 : 1 Transaction Per Micro Frame. 10 : 2 Transaction Per Micro Frame. 11 : 3 Transaction Per Micro Frame.	2'b0
[9]	R/W	**TNPMF_0**		
[8]	R/W	**IME**	ISO Mode Endpoint. IME determines the transfer type of an endpoint. 0 : Bulk(Interrupt) Mode. 1 : ISO mode.	1'b0
[7]	R/W	**DUEN**	Dual FIFO Mode Enable. 0 : Dual Disable (Single Mode) 1 : Dual Enable	1'b0
[6]	R/W	**FLUSH**	FIFO Flush. FIFO is flushed when this bit is set to '1'. This bit is automatically cleared after MCU writes '1'.	1'b0
[5]	R/W	**TTE**	Tx Toggle Enable. The MCU can force Tx data toggle bit with TTE. This bit is useful for test. The Tx data toggle bit changes automatically in normal operation.	1'b0

Bit	R/W	Symbol	Description	● Reset Value
			0 : Disable 1 : Enable	
[4]	R/W	**TTS_1**	Tx Toggle Select. TTS is used for Test. This is valid when Tx Toggle Enable(TTE) is set. 00 : DATA PID 0 01 : DATA PID 1 10 : DATA PID 2(Only in ISO Mode) 11 : DATA PID M(Only in ISO Mode)	2'b0
[3]	R/W	**TTS_0**		
[2]	R/W	**CDP**	Clear Data PID. - In Rx Mode When this bit is set to '1', data toggle bit in core to be compared with the data PID of received packet is reset to '0'. This bit is automatically cleared after MCU writes '1'. - In Tx Mode Tx data PID to be transmitted to host is reset to '0' when this bit is set to '1'. This bit is automatically cleared after MCU writes '1'.	1'b0
[1]	R/W	**ESS**	Endpoint Stall Set. ESS is set by the MCU when the MCU intends to send STALL handshake to Host. This bit is cleared when the MCU writes '0' in it.	1'b0
[0]	R/W	**IEMS**	Interrupt Endpoint Mode Set IEMS determines the transfer type of an endpoint. 0 : interrupt Transfer mode Disable, 1: interrupt Transfer mode Enablef	1'b0
BYTE READ COUNT REGISTER (BRCR) ***Address : C001_801Ah***				
[15(7) : 10(2)]	-	**RESERVED**	Reserved	6'b0
[9(1)]	R	**RDCNT_9**	FIFO Read Byte Count [9 : 0] RDCNT is read only. The BRCR inform the amount of received data from host. In 16 bit Interface, RDCNT informs the amount of data in word(16bit) unit. Through the LWO bit of EP0SR, the MCU can determine valid byte in last data word. In 8bit Interface, RDCNT keeps the byte size of received data.	1'b0
[8(0)]	R	**RDCNT_8**		1'b0
[7]	R	**RDCNT_7**		1'b0
[6]	R	**RDCNT_6**		1'b0
[5]	R	**RDCNT_5**		1'b0
[4]	R	**RDCNT_4**		1'b0
[3]	R	**RDCNT_3**		1'b0
[2]	R	**RDCNT_2**		1'b0
[1]	R	**RDCNT_1**		1'b0
[0]	R	**RDCNT_0**		1'b0
BYTE WRITE COUNT REGISTER (BWCR) ***Address : C001_801Ch***				
[15(7) : 10(2)]	-	**RESERVED**	Reserved	6'b0
[9(1)]	R/W	**WRTCNT_9**	Through BWCR, the MCU must load the byte counts of a Tx data packet to the core. The core uses this count value to determine the end of packet. The count value to this register must be less than MAXP.	1'b0
[8(0)]	R/W	**WRTCNT_8**		1'b0
[7]	R/W	**WRTCNT_7**		1'b0
[6]	R/W	**WRTCNT_6**		1'b0
[5]	R/W	**WRTCNT_5**		1'b0
[4]	R/W	**WRTCNT_4**		1'b0
[3]	R/W	**WRTCNT_3**		1'b0
[2]	R/W	**WRTCNT_2**		1'b0

Bit	R/W	Symbol	Description	● Reset Value
[1]	R/W	**WRTCNT_1**		1'b0
[0]	R/W	**WRTCNT_0**		1'b0
MAX PACKET REGISTER (MPR) ***Address : C001_801Eh***				
[15(7) : 10(2)]	-	**RESERVED**	Reserved	6'b0
[10(2)]	R/W	**MAXP_10**		1'b0
[9(1)]	R/W	**MAXP_9**	Max Packet [9 : 0]. The max packet size of each endpoint is determined by MAX packet register. The range of max packet is from '0' to '1024' byte. 000_0000_0000 : Max Packet 0. 000_0000_1000 : Max Packet 8. 000_0001_0000 : Max Packet 16. 000_0010_0000 : Max Packet 32. 000_0100_0000 : Max Packet 64. 000_1000_0000 : Max Packet 128. 001_0000_0000 : Max Packet 256. 010_0000_0000 : Max Packet 512. 100_0000_0000 : Max Packet 1024.	
[8(0)]	R/W	**MAXP_8**		1'b0
[7]	R/W	**MAXP_7**		1'b0
[6]	R/W	**MAXP_6**		1'b0
[5]	R/W	**MAXP_5**		1'b0
[4]	R/W	**MAXP_4**		1'b0
[3]	R/W	**MAXP_3**		1'b0
[2]	R/W	**MAXP_2**		1'b0
[1]	R/W	**MAXP_1**		1'b0
[0]	R/W	**MAXP_0**		1'b0
DMA CONTROL REGISTER (DCR) ***Address : C001_8020h***				
[15(7) : 6]	-	**RESERVED**	Reserved	10'b0
[5]	R/W	**ARDRD**	Auto Rx DMA Run set disable. This bit is cleared when DMA operation is ended. 0 : Disable 1 : Enable	1'b0
[4]	R/W	**FMDE**	Fly Mode DMA Enable. This bit is used to run Fly mode DMA operation. 0 : Fly mode Disable 1 : Fly mode Enable	1'b0
[3]	R/W	**DMDE**	Demand Mode DMA Enable. This bit is used to run Demand mode DMA operation. 0 : Demand mode Disable. 1 : Demand mode Enable	1'b0
[2]	R/W	**TDR**	Tx DMA Run. This bit is used to set start DMA operation for Rx Endpoint (In endpint) 0 : DMA Stop 1 : DMA Run	1'b0
[1]	R/W	**RDR**	Rx DMA Run. This bit is used to start DMa operation for Rx Endpoint(OUT endpoint). This bit is automatically set when USB receives OUT packet data and DEN bit is set to '1' and ARDRD bit is set to '0'. To operate DMA after OUT packet data received, MCU must set RDR to '1'. 0 : DMa Stop 1 : DMA Run	1'b0
[0]	R/W	**DEN**	DMA Mode Enable This bit is used to set the DMA mode. 0 : Interrupt Operation mode. 1 : DMA Operation mode.	1'b0
DMA TRANSFER COUNTER REGISTER (DTCR) ***Address : C001_8022h***				
[15 : 0]	R/W	**-**	This 16 bit counter keeps the DMA transfer unit. If the value of counter is set to 16'h0002, there will be two bytes(one word) transfer per each DMA request. To operate single mode DMA transfer, DTCR is needed to be set 16'h0002 and Demand mode in DCR	16'b0

Bit	R/W	Symbol	Description	● Reset Value
			is disabled. In case of Demand or Fly mode, the MCU should set max packet value.	
DMA FIFO COUNTER REGISTER (DFCR) ***Address : C001_8024h***				
[15 : 0]	R/W	-	In case of OUT Endpoint, the size value of received packet will be loaded in this register automatically when Rx_DMA_run is enabled. In case of IN Endpoint, the MCU should set max packet value.	16'b0
DMA TOTAL TRANSFER COUNTER REGISTER (DTTCR) ***Address : C001_8026h, C001_8028h***				
[15 : 0]	R/W	-	This register should have total byte size to be transferred using DMA. DMA Total Transfer Counter 1 : Low half word value. DMA Total Transfer Counter 2 : High half word value. The max value is up to 2^{32}	16'b0
ENDPOINT STATUS REGISTER2 ESR2) ***Address : C001_802Ah***				
[15 : 0]	R		These bits does not need to be controlled or checked in normal operation and are only used for debugging.	[15 : 0]
SYSTEM CONTROL REGISTER (SCR) ***Address : C001_802Ch***				
[15 : 3]	-	**RESERVED**	Reserved	13'b0
[2]	R/W	**EP2RST**	Endpoint 2 Software Reset 0 : Reset Enable 1 : Reset Disable	1'b1
[1]	R/W	**EP1RST**	Endpoint 1 Software Reset 0 : Reset Enable 1 : Reset Disable	1'b1
[0]	R/W	**EP0RST**	Endpoint 0 Software Reset 0 : Reset Enable 1 : Reset Disable	1'b1
PHY LINK INTERFACE CONTROL REGISTER (PLICR) ***Address : C001_8050h***				
[15 : 12]	-	**RESERVED**	Reserved	4'b0
[11:8]	R/W	**PLC**	Link to PHY interface Control. The region of PLC is from 4'b0000 to 4'b1111.	4'b0010
[7:4]	R/W	**LPC**	Link to PHY interface Control. The region of LPC is from 4'b0000 to 4'b1111.	4'b0100
[3:0]	-	**RESERVED**	Reserved	4'b0
PHY CONTROL REGISTER (PCR) ***Address : C001_8052h***				
[15 : 12]	-	**RESERVED**	Reserved	4'b0
[7]	R/W	**URSTC**	UTMI_RESET Contro (PHY Software Reset) 0 : UTMI_RESET is 0 (Reset enable) 1 : UTMI_RESET is 1.	1'b0
[6]	R/W	**SIDC**	SIDDQ Control 0 : SIDDQ is 0 1 : SIDDQ is 1.I	1'b0
[5:4]	R/W	**OPMC**	OPMODE Control 00 : Normal. 01 : Non-Driving 10 : Disable bit stuffing and NRZI encoding	2'b0

Bit	R/W	Symbol	Description	● Reset Value
			11 : Reserved.	
[3]	R/W	**TMSC**	TERMSEL Control 0 : TERMSEL is 0. 1 : TERMSEL is 1	1'b0
[2]	R/W	**XCRC**	XCVRSEL Control 0 : XCVRSEL is 0. 1 : XCVRSEL is 1	1'b0
[1]	R/W	**SUSPC**	SUSPENDM Control 0 : SUSPENDM is 0. 1 : SUSPENDM is 1	1'b0
[0]	R/W	**PCE**	PHY Control Enable 0 : Control Disable. 1 : Control Enable	1'b0
USB CLOCK ENABLE REGISTER ***Address : C001_88C0h***				
[31 : 4]	-	**RESERVED**	Reserved	28'b0
[3]	R/W	**USB_PCLKMODE**	PCLK Operating Mode 0 : Clock is enabled only when CPU accesses 1 : Always	1'b0
[2]	R/W	**USBD_CLKGENENB**	Clock Generation Enable 0 : Disable 1 : Enable	1'b0
[1 : 0]	R/W	**USBD_CLKENB**	USBD Clock Enable 00 : Disable 01 : Reserved 10 : Dynamic 11 : Always	2'b11
USB CLOCK GENERATE REGISTER ***Address : C001_88C4h***				
[31 : 10]	-	**RESERVED**	Reserved	22'b0
[9 : 4]	R/W	**USBD_CLKDIV**	Clock Divisor. For divide – by - N, enter an [N – 1] value. 000000 ~ 111111(N-1) : Divide Value (N) = Divide by 1 to 64 Ex) For divide-by-2 : [0001b]	1'b0
[3 : 1]	R/W	**USBD_CLKSRCSEL**	Clock Source Selection 000 : Reserved 001 : Resrved 010 : Reserved 011 : Ext Clock 100 ~ 111 : Reserved	3'b0
[0]	-	**RESERVED**	Reserved	1'b0
UserTest ***Address : C001_884Ch***				
[15]	R/W	**VBUSENB**	External VBUS Enbale 0 : Disable 1: Enable	1'b0
[14:0]	-	**RESERVED**	Reserved	15'b0

CHAPTER 16.

SSP/SPI

16. SSP/SPI

16.1. Overview

The SSP/SPI is a full-duplex synchronous serial interface. It supports Synchronous Serial Protocol (SSP) and Serial Peripheral Interface (SPI) protocol. It can connect to a variety of external converter, serial memory and many other device which use serial protocols for transferring data.

There are 4 I/O pin signals associated with SSP/SPI transfers: the SSPCLK, the SSPRXD data receive line, the SSPTXD data transfer line, SSPFRM (Frame Indicator in SSP mode, Chip Select in SPI mode).

The POLLUX has three SSP/SPI ports and each port can operate in Master and Slave mode.

16.1.1. Features

- SSP Protocol, SPI Protocol
- 32x16 FIFO
- Master&Slave mode
- Polling, Interrupt, DMA transfer mode
- Support Standard four SPI Format
  - Format A, Normal
  - Format A, Invert
  - Format B, Normal
  - Format B, Invert
- 5-bit pre-scale counter
- 3 Channel SSP/SPI
- Programmable clock phase and polarity
- Independent programmable baud rate generator (in Master mode)
- Supports data characters from 8 to 16 bits long
- Max Operation Frequency : 25MHz

16.1.2. Block Diagram

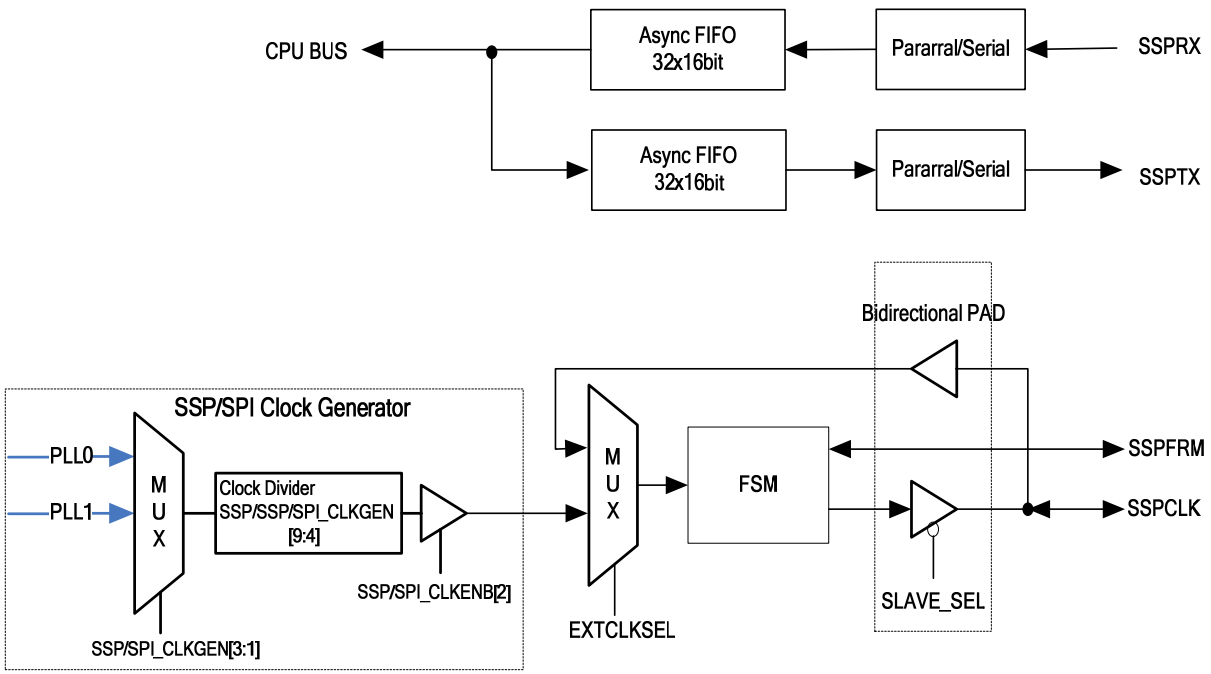


Figure 16-1. SSP/SPI Block Diagram

16.1.3. Pin Function Description

Pin Name	GPIO No	GPIO Function	Type	Description
SSPCLK2	B[0]	ALT2	I/O	
SSPCLK1	C[4]	ALT1	I/O	
SSPCLK0	B[13]	ALT1	I/O	
SSPFRM2	B[5]	ALT2	I/O	
SSPFRM1	C[3]	ALT1	I/O	
SSPFRM0	B[12]	ALT1	I/O	
SSPRXD2	B[1]	ALT2	I	
SSPRXD1	C[5]	ALT1	I	
SSPRXD0	B[14]	ALT1	I	
SSPTXD2	B[2]	ALT1	O	
SSPTXD1	C[6]	ALT1	O	
SSPTXD0	B[15]	ALT1	O	

Table 16-1. SSP/SPI Pin Function Description

16.2. Operation

The SSP/SPI Block transfers Serial Data from / to External device via FIFO in the SSP/SPI Block. The transfer operation is initiated by CPU, using the programmed I/O system or the DMA, to / from the system memory. The SSP/SPI Data transfer is performed in full duplex.
When the SSP/SPI sends Data to the PIO mode, the transfer operation is completed by Reading 'Read FIFO' or Writing to 'Write FIFO' by means of a program.
In the case of communication with DMA mode, In DMA transfer mode, the ***DMAENB*** bit in the DMA and the ***SSPSPICONT0*** bit are set as '1' to enable the DMA transfer.
If the block operates DMA mode by using an interrupt, the operation undergoes a lengthy delay before the processor jumps to the interrupt handler. When the Rx FIFO is full of data, the next data arriving may be lost as a result of the delay before jumping to the interrupt handler. Hence the interrupt operation needs the use of IRQE. However the external device speed should be 1 Mbps or less.
The POLLUX transmits a clock/frame signal to external devices in case that it operates in Master mode, while it receives a clock/frame signal from an external device in case that it operates in Slave mode.

16.2.1. SPI Mode

- Format A, Normal
- Format A, Invert
- Format B, Normal
- Format B, Invert

16.2.1.1. Format A

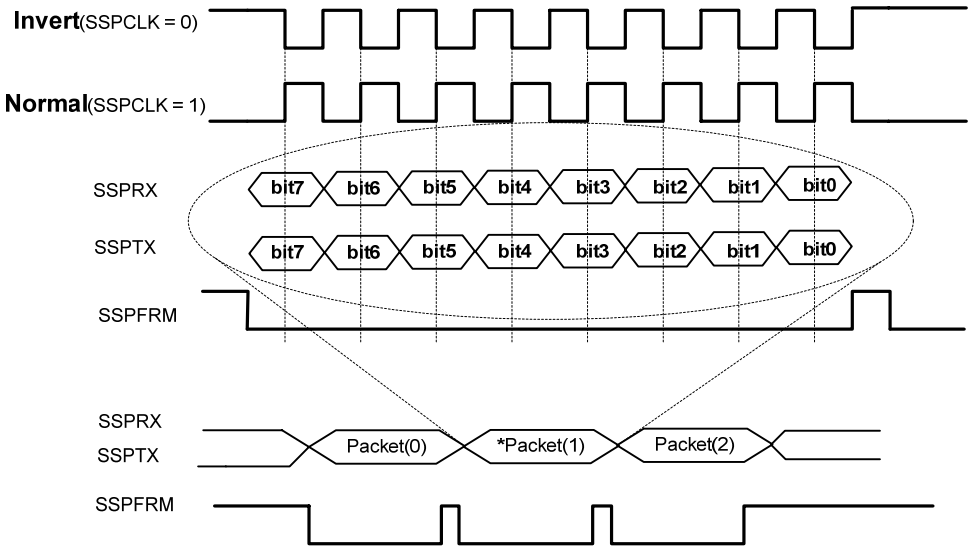


Figure 16-2. Format A & Continuous Data Transmission Timing

In invert mode (SSPCLK = 0), the SSPFRM is low and data is fetched on the falling edge of SSPCLK. In normal mode (SSPCLK = 1), the SSPFRM is low and data is fetched on the rising edge of the SSPCLK.

In continuous data transmission/reception, SSPERM is toggled whenever a packet is transmitted/received.

*Packet : Data bit width to be transmitted/received at the same time. (Looking at Figure 15-2, the packet size is 8-bit.).

16.2.1.2. Format B

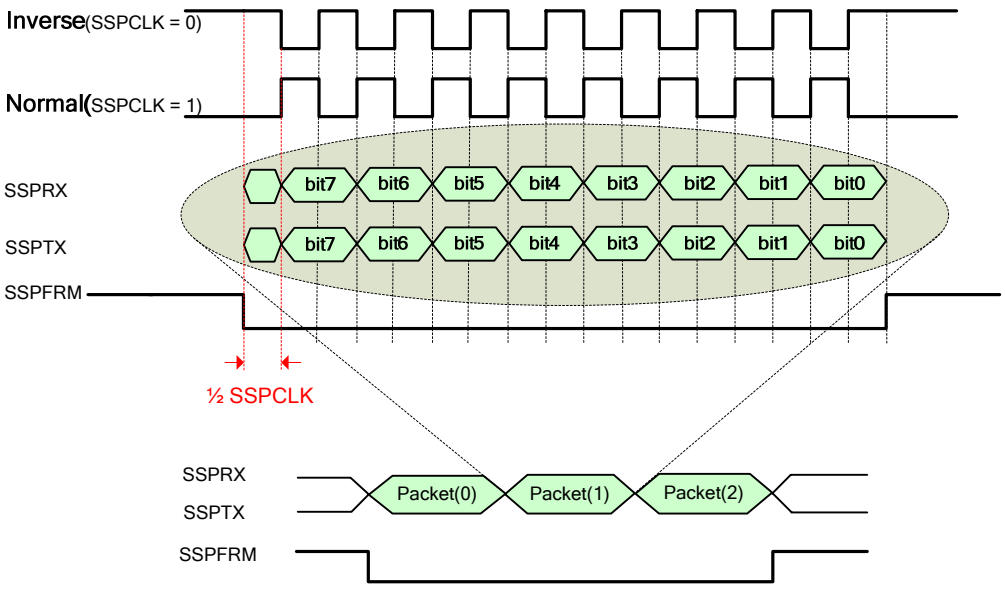


Figure 16-3. Format B & Continuous Data Transmission Timing Chart

In format B, when SSPERM asserts as low, data is transmitted/received after a half cycle of SSPCLK. In invert mode (SSPCLK = 0), data is fetched on the rising edge of SSPCLK. In normal mode (SSPCLK = 1), data is fetched on the falling edge of SSPCLK.

In continuous data transmission/reception, SSPFRM is not toggled until all packets are transmitted/received. After the completion of the transmission/reception, SSPERM asserts as high.

16.2.2. SSP Mode

16.2.2.1. SSP Timing

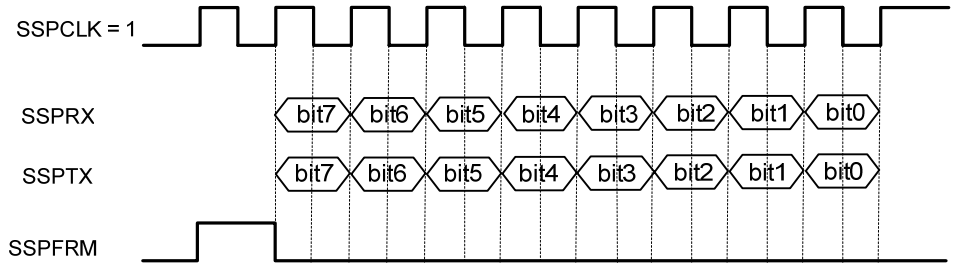


Figure 16-4. SSP mode Timing Chart

IN SSP mode, the SSPERM signal is toggled from low to high for a SSPCLK cycle before all the data transmission/reception. In addition, when SSPCLK is on the falling edge, data is fetched. Figure 15-5 shows the time diagram for transmission/reception of a packet.

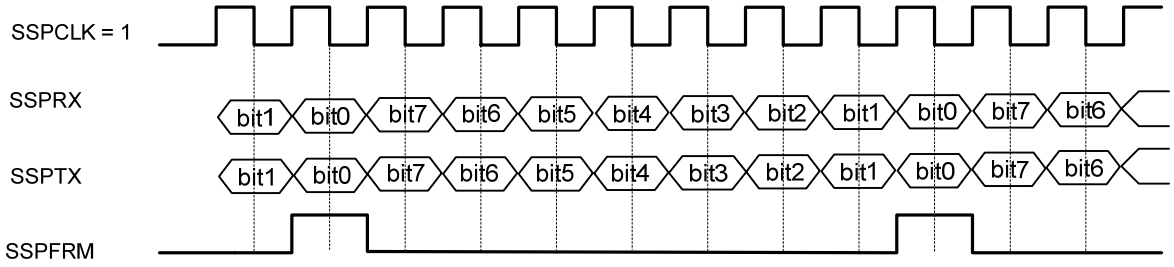


Figure 16-5. Continuous Data Transmission Timing Chart

In continuous data transmission/reception, SSPERM is toggled to high at the previous packet's LSB of the packet to be transmitted/received. Therefore, when the last packet is transmitted/received, SSPFRM is not toggled to high at the LSB of the packet.

16.2.3. SSPSPI Burst Receive mode

16.2.3.1. SSPSPI Burst Recive Timing

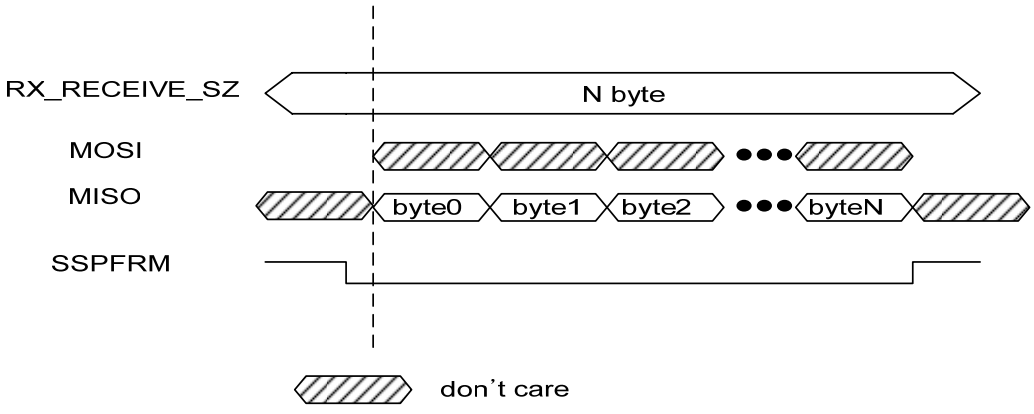


Figure 16-6. Burst Receive Timing Chart

Burst Receive can be used only when SPI Format B is in Master mode.

To use Burst Receive,

i) Set BRECEIVE bit to '1'

ii) Set RX_RECEIVE_SZ(decrease 1 from actual receiving data value) according to slave device.

iii) Set ST_RXBRECEIVE as '1'. Simply set ST_RXBRECEIVE as '1', if you want to receive data from slave as much as (RX_RECEVE_SZ+1) after master transfers data to DMA as much as (RX_RECEIVE_SZ+1).

16.3. Register Summary

Bit	R/W	Symbol	Description	Reset Value
SSP/SPI CONTROL REGISTER0 (SSPSPICONT0) ***Address : SSP/SPI0 : C000_7800h / SSP/SPI 1: C000_8000h / SSP/SPI2:C000_8800h***				
[15]	R/W	**ST_RXBRECEIVE**	Starts RX Burst Receive in master mode. 0: not start. 1: Rx Burst Receive start (Auto Clear)	1'b0
[14]	R/W	**RESERVED**		1'b1
[13]	R/W	**BRECEIVE**	Stands for Burst Transfer. It's meanful when Master mode. It maintains low status until SSPFRM signal receive BURST_SZ all. It only supports FORMATB. 0: Burst Receive disable 1: Burst Receive enable	1'b0
[12]	R/W	**DMAENB**	DMA Mode Enable 0 : PIO Mode 1 : DMA Mode	1'b0
[11]	R/W	**ENB**	SPI/SSP Enable 0: Disable 1: Enable	1'b0
[10]	R/W	**FFCLR**	Initialize internal FIFO. Toggle is required. 0 : IDLE 1 : RESET	1'b0
[9]	R/W	**EXTCLKSEL**	Use an External Clock. 0: Internal Clock 1: I_SSPEXTCLK	1'b0
[8:5]	R/W	**NUMBIT**	Specify the data bit width of a data packet to be transmitted. (n+1) bits, Max. 16 bits	4'b111
[4:0]	R/W	**DIVCNT**	Specify the divide value of the internal clock. The value should be greater than 2. ex) DIVCNT=3 : 4 Divide Ratio DIVCNT=4 : 5 Divide Ratio	5'b10
SSP/SPI CONTROL REGISTER1 (SSPSPICONT1) ***Address : SSP/SPI0 : C000_7802h / SSP/SPI 1: C000_8002h / SSP/SPI2:C000_8802h***				
[15:6]	–	**RESERVED**	Reserved	11'b0
[5]	R/W	**BYTE_SWAP**	Switch location of Data byte in 16bit FIFO. It allows using bandwidth more efficiently in 8bit reception device. 0: no byte swap 1: byte swap	1'b0
[4]	R/W	**SLAVE_SEL**	Decide the slave operation. 0 : master 1 : slave	1'b0
[3]	R/W	**SCLKPOL**	Specify the polarity of outgoing SSPCLK. 0 : Invert 1 : Normal	1'b0
[2]	R/W	**SCLKSH**	Specify the phase of outgoing SPICLK. 0 : Format A 1 : Format B	1'b0
[1:0]	R/W	**TYPE**	SSP/SPI Mode 00 : SSP 01 : SPI 1x: Reserved	2'b0
SSP/SPI DATA REGISTER (SSPSPIDATA) ***Address : SSP/SPI0 : C000_7804h / SSP/SPI 1: C000_8004h / SSP/SPI2:C000_8804h***				
[15:0]	R/W	**DATA**	Data to be transmitted in writing operation and received data in reading operation	16'bX
SSP/SPI STATUS REGISTER (SSPSPISTAT) ***Address : SSP/SPI0 : C000_7806h / SSP/SPI 1: C000_8006h/ SSP/SPI:C000_8806h***				
[15]	R/W	**IRQEENB**	IRQE Enable Register 0 : Disable 1 : Enable	1'b0
[14]	R/W	**IRQWENB**	IRQW Enable Register 0 : Disable 1 : Enable	1'b0
[13]	R/W	**IRQRENB**	IRQR Enable Register 0 : Disable 1 : Enable	1'b0
[12]	R/W	**RESERVED**		1'b0

Bit	R/W	Symbol	Description	Reset Value
[11:9]	–	**RESERVED**	Reserved	6'b0
[8]	R	**TXSHIFTEREMPTY**	When Tx Shift Register is empty, it becomes '1'	1'b1
[7]	R/W	**RESERVED**		1'bX
[6]	R/W	**IRQE**	Raise an interrupt when an Rx/Tx is completed.	1'bX
[5]	R/W	**IRQW**	Raise an interrupt when the transmission buffer is empty. Write '1' to clear.	1'bX
[4]	R/W	**IRQR**	Raise an interrupt when the reception buffer is full. Read '1' to clear.	1'bX
[3]	R	**WFFFULL**	Set as '1' when the transmission buffer is full.	1'b0
[2]	R	**WFFEMPTY**	Set as '1' when the transmission buffer is empty. This bit does'n reflects a valid state under SPI format B on slave mode.	1'b1
[1]	R	**RFFFULL**	Set as '1' when the reception buffer is full.	1'b0
[0]	R	**RFFEMPTY**	Set as '1' when the reception buffer is empty.	1'b1
SSP/SPI BURST RECEIVE SIZE (SSP_RX_BRECEIVE_SZ) ***Address : SSP/SPI0 : C000_7808h / SSP/SPI1 : C000_8008h / SSP/SPI2:C000_8808h***				
[15:0]	R/W	**RX_BRECEIVE_SZ**	Burst Receive Size. It is meanful only if it's master mode. 0x00 ~ 0xFFFF (1 ~ 65536 byte)	16'b255
RESERVED ***Address : SSP/SPI0 : C000_7808h ~ C000_783Ch / SSP/SPI1 : C000_8008h ~C000_803Ch / SSP/SPI2:C000_8808h~C000_883Ch***				
SSP/SPI CLOCK ENABLE REGISTER (SSP_CLKENB) ***Add Address : SSP/SPI0 : C000_7840h / SSP/SPI1 : C000_8040h / SSP/SPI2:C000_8840h***				
[31 : 4]	-	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	PCLK operating mode 0 : Disable 1 : Enable	1'b0
[2]	R/W	**CLKGENENB**	Clock generation Enable 0 : Disable 1 : Enable	1'b0
[1:0]	-	**RESERVED**	Reserved	2'b0
SSP/SPI CLOCK GENERATE REGISTER (SSP_CLKGEN) ***Address : SSP/SPI0 : C000_7844h / SSP/SPI 1: C000_8044h / SSP/SPI2:C000_8844h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9:4]	R/W	**CLKDIV**	Clock divisor: Since the value of CLKDIV +1 is put as the divider value, divide-by-I through 64 are available. The output clock of CLKGEN - PLLn/(CLKDIV+1) - must be less than 100Mhz	6'b0
[3:1]	R/W	**CLKSRCSEL**	Clock source selection 000 : PLL0 001 : PLL1 010 : Reserved 011 ~ 111: Reserved	3'b0
[0]	-	**RESERVED**	Reserved	1'b0

CHAPTER 17.

I2C CONTROLLER (I2C)

17. I2C Controller (I2C)

17.1. Overview

The POLLUX Microprocessor can support a multi-master I^2C-bus serial interface. Information between bus masters and peripheral devices that are connected to the I^2C-bus is carried by a dedicated serial data line (SDA) and a serial clock line (SCL). The SDA and SCL lines are bi-directional. In multi-master I^2C-bus mode Serial data to or from slave devices can be received or transmitted by multiple POLLUX Microprocessors. The master POLLUX, by which a data transfer can be initiated over the I^2C-bus, is responsible for terminating the transfer. This I^2C-bus in POLLUX uses the standard bus arbitration procedure.

17.1.1. Features

- 2 channel I^2C-bus
- Speed 100Kbps ~ 1Mbps (due to clock prescaler)
- Interrupt mode (byte transfer)
- Support Master & Slave mode

17.1.2. Block Diagram

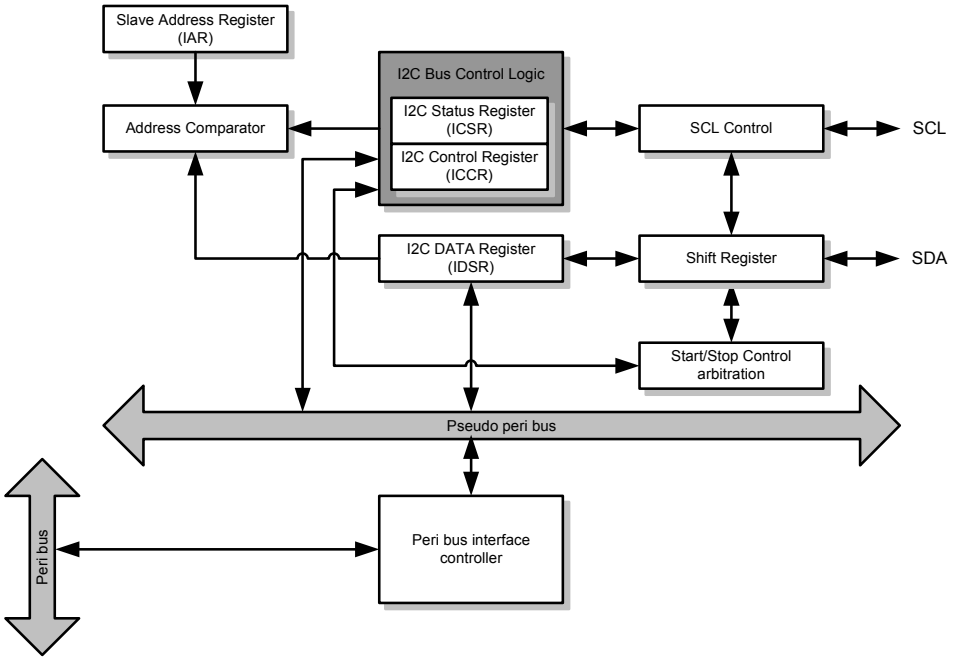


Figure 17-1. I^2C Block Diagram

17.1.3. Pin Function Description

Pin Name	GPIO No	GPIO Function	Type	Description
SDA0	A[27]-	ALT1	I/O	Serial Data Line 0
SCL0	A[26]	ALT1	I/O	Serial Clock Line 0
SDA1	A[29]	ALT1	I/O	Serial Data Line 1
SCL1	A[28]	ALT1	I/O	Serial Clock Line 1

Table 17-1. I^2C Pin Function Description

17.2. Operation

The SDA and SCL lines should both be at High level when the I^2C-bus is free. A Start condition can be initiated by a High-to-Low transition of SDA. A Stop condition can be initiated by a Low-to-High transition of SDA while SCL remains steady at High Level.

Start and Stop conditions can always be generated by Master devices. The slave device that the bus master device has selected can be

determined by a 7-bit address value in the first data byte, which is put onto the bus after the Start condition has been initiated. The direction of the transfer (read or write) is determined by the 8th bit.

Each SDA line data byte should total eight bits. The number of bytes that can be received or sent during the bus transfer operation is not limited. An acknowledge (ACK) bit should immediately follow every byte and data is always sent from the most significant bit (MSB) first.

The Master device controls the start/stop of the data transmission through the I^2C-bus and creates the clock signal (SCL). If byte data are successfully transmitted, the receiver generates the ACK signal.

I^2C data are transmitted in two ways: IRQ mode (non-burst). In IRQ mode, IRQ is generated at every 1 byte data transmission/reception and the CPU controls byte transmission.

17.2.1. I^2C Bus Interface

There are four operation modes in the POLLUX I^2C-bus interface:

- Master transmitter mode
- Master receiver mode
- Slave transmitter mode
- Slave receiver mode

The following describes functional relationships among these operating modes:

17.2.1.1. START AND STOP CONDITIONS

The I^2C-bus is usually in slave mode when it is inactive. That is, the interface should be in slave mode before detecting a Start condition on the SDA line.(A High-to-Low transition of the SDA line can be initiated by a Start condition while the clock signal of SCL is High). A data transfer on the SDA line can be initiated and SCL signal generated when the interface state is changed to master mode.

A single byte serial data over the SDA line can be transferred after a Start condition and the data transfer can be terminated by a Stop condition. A Stop condition is a Low-to-High transition of the SDA line while SCL is High. Stop and Start conditions are always generated by the master. When a Start condition is generated, the I^2C-bus is busy. The I^2C-bus will be free again within a few clocks after a Stop condition.

A slave address is sent to notify the slave device when a master initiates a Start condition. A 7-bit address and a 1-bit transfer direction indicator (write or read) are included in the single byte of the address field. A write operation (transmit operation) is indicated when bit 8 is 0; request for data read (receive operation) is indicated when bit 8 is 1.

The transfer operation will be completed by the master through transmitting a Stop condition. Another Start condition and a slave address should be generated when the master wants to continue data transmission to bus. So the read-write operation can be performed in various formats.

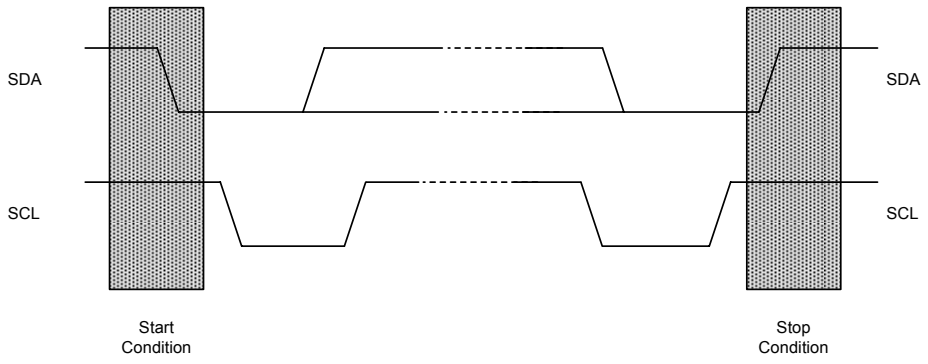
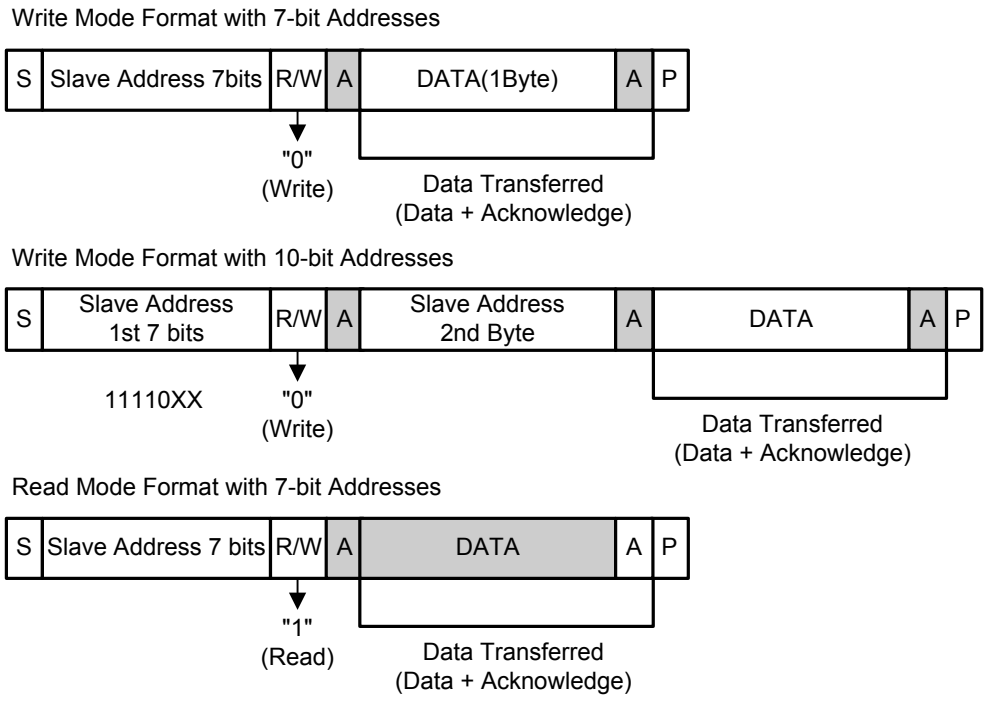


Figure 17-2. Start and Stop Condition

17.2.1.2. DATA TRANSFER FORMAT

Every Data will be transmitted on the SDA line byte should be 8 bits long. The number of bytes that can be transmitted per transfer is unlimited. The address filed should be contained in the first byte following a Start condition. When the I^2C-bus is operating in master mode, the master can transmit the address field. An acknowledge (ACK) bit should follow each byte. The MSB bit of the serial data and addresses are always sent first.



NOTES:

1. S: Start, rS: Repeat Start, P: Stop, A: Acknowledge
2. ☐ : From Master to Slave, ☐: from Slave to Master

Figure 17-3. I²C-Bus Interface Data Format

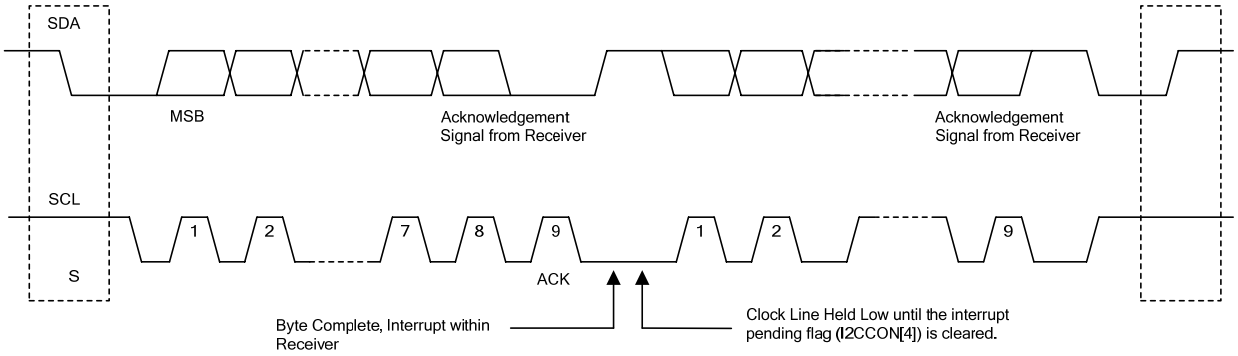


Figure 17-4. Data Transfer on the I²C–Bus

17.2.1.3. ACK SIGNAL TRANSMISSION

An ACK should be sent by the receiver to the transmitter Master to finalize a single byte transfer operation. The ACK pulse should occur at the ninth clock of the SCL line. For the single byte data transfer eight clocks are required. The clock pulse required to transmit the ACK bit should be generated by the master.

The transmitter should release the SDA line by making the SDA line High when the ACK clock pulse is received, should release the SDA line. The SDA line should also be driven Low by the receiver during the ACK clock pulse so that SDA is Low during the High period of ninth SCL pulse.

Software (***ICSR***) can enable or disable the ACK bit transmit function. But completion of a single byte data transfer operation requires the ACK pulse on the ninth clock of SCL.

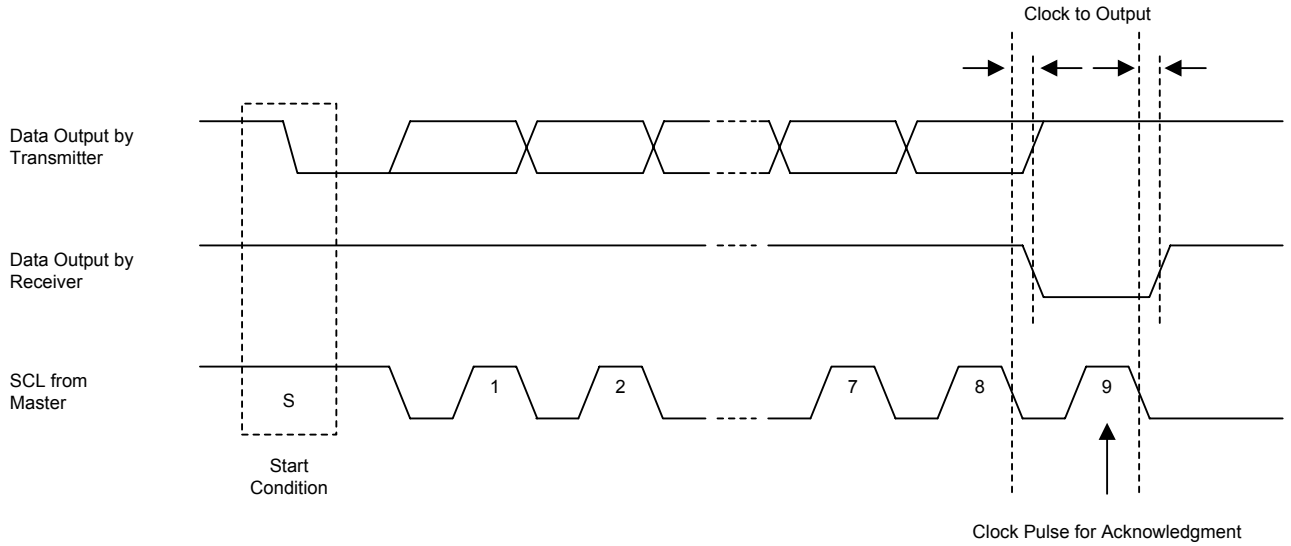


Figure 17-5. Acknowledge on the I²C-Bus

17.2.2. READ-WRITE OPERATION

Until new data is written to the ***IDSR*** (I²C-bus Data Shift Register) the I²C-bus interface will wait after the transmission of the data in transmitter mode. The SCL line will be held Low until the new data is written. The SCL line will be released after the new data is written to the ***IDSR*** Register. The interrupt should be held by the POLLUX to identify competition of the current data transfer. New data should be written to ***IDSR*** again after the interrupt is received by the CPU.

Until the ***IDSR*** Register is read, the I²C-bus interface will wait after data is received in receive mode. The SCL line will be held Low until the new data is read. The SCL line will be released after the new data is read from the ***IDSR*** Register. The interrupt should be held by the POLLUX to identify the completion of the new data reception. The data should be read from ***IDSR*** after the interrupt request is received by the CPU.

17.2.2.1. BUS ARBITRATION PROCEDURES

To prevent conflict between two masters on the bus, arbitration takes place on the SDA line. A data transfer by a master with an SDA High level will not be initiated if another master with an SDA active Low level is not detected because the current level on the bus does not correspond to its own level. Until the SDA line turns High, the arbitration procedure will be extended.

But each master should evaluate whether or not the mastership is allocated to itself when each master simultaneously lowers the SDA line. The address bits are detected by each master for the purpose of evaluation. The address bit is also detected on the SDA line while the save address is generated by each master because the lowering of the SDA line is stronger than maintaining High on the line. For instance, a Low is generated by one master as the first address bit, while the other master is maintaining High - in this case, both masters will detect Low on the bus because Low is stronger than High even if the first master is trying to maintain High on the line. When this happens, the Low (as the first bit of the address) generating master will get the mastership and the High (as the first bit of the address) generating master should withdraw from mastership. When Low is generated by both masters as the first bit of the address, there should be arbitration for the second address bit again. This arbitration will continue to the end of the last address bit.

17.2.2.2. ABORT CONDITIONS

The SDA line level should be maintained at High when a slave receiver cannot acknowledge the slave address confirmation. In this case, to abort the transfer, the master should generate a Stop condition.

The end of the slave transmit operation should be signaled when a master receiver is involved in the aborted transfer by canceling the generation of an ACK after the last data byte is received from the slave. To allow a master to generate a Stop condition, the slave transmitter should then release the SDA.

17.2.2.3. CONFIGURING THE I²C –BUS

In the ***ICCR*** Register, the 4-bit prescaler value can be programmed to control the frequency of the serial clock (SCL). The I²C-bus interface address is stored in the I²C-bus address register, IAR. (By default, the value of I²C-bus interface address is unknown.)

17.2.2.4. SDA UPDATE AND FETCH TIMING

(1) SDA update timing when transmit mode

In transmit mode, SDA is updated after 6 cycle of PCLK from SCL transition High to Low for stable operation. If more delay is needed to SDA signal line, Quarter Period Count Max register (***QCNT_MAX***) can be used. SDA update delay is (6 cycle + QCNT_MAX cycle) of PCLK.

(2) SDA fetch timing when receive mode

In receive mode, SDA is fetched after 4 cycle of PCLK from SCL transition Low to High for stable operation. If more delay is needed to SDA signal line, Quarter Period Count Max register (***QCNT_MAX***) can be used. SDA update delay is (4 cycle + QCNT_MAX cycle) of PCLK.

17.2.3. Example of the operation in Master mode

17.2.3.1. Master Transmitter and Receiver Mode data format

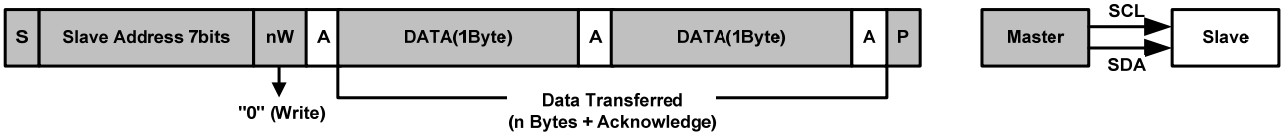


Figure 17-6. Master Transmitter Mode Data Format

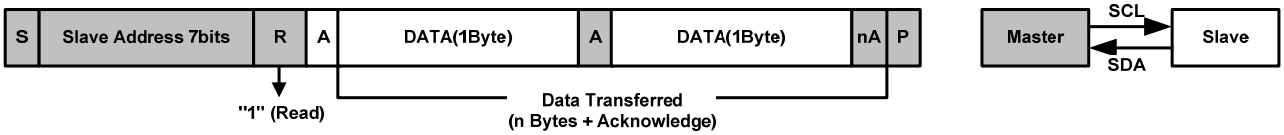


Figure 17-7. Master Receiver Mode Data Format

17.2.3.2. Programming Examples

- **Initialize the I²C controller.**

1. Clear the I²C control register (***ICCR***).
2. Set the clock source (***ICCR.CLK_SRC = 0 → PCLK/16, ICCR.CLK_SRC = 1 → PCLK/256)***
3. Set clock prescaler (***ICCR.CLK_SCALER)***
4. Set the transition timing to the ***QCNT_MAX*** register. (recommend 1~5)
5. Write '1010h' to the I²C status register (***ICSR***).
6. Write ***IRQ_PEND.OP_HOLD*** =1 to generate a STOP condition

(1)

- **Write to slave device**

1. Write a slave address to ***IDSR***.
2. Set the ***ICSR*** as the Master Transmitter mode. **(*ICSR* &= 1F0Fh, *ICSR* |= 0x10F0h)**
3. Write ***IRQ_PEND.OP_HOLD*** =1 to generate a START condition.
4. Wait until an interrupt occurs. (***IRQ_PEND.PEND***).
5. If an interrupt occurs, check the ACK signal.
6. If the ACK signal is received normally, write a sub address to the IDSR.
7. Write ***IRQ_PEND.OP_HOLD*** =1 to transmit the sub address.
8. Check the ACK signal.
9. If the ACK signal is received normally, the data is transmitted depending on the mode.
10. IRQ mode data transmission.
    1) Write data to the ***IDSR***.
    2) Write '1' to ***IRQ_PEND.PEND*** to clear the Interrupt Pend register.
    3) Write '1' to ***IRQ_PEND.OP_HOLD*** to transmit data.
    4) Wait for IRQ.
    5) Repeat Steps 1) to 4) until all data are transmitted.

6) After the transmission, set the STOP condition to the ***ICSR***. **(*ICSR* &= 1F0Fh, *ICSR* |= 0x10D0h)**
7) Write '1' to ***IRQ_PEND.OP_HOLD*** to generate the STOP condition.
8) Clear all registers to disable the I^2C controller.

- **Read from slave device**

1. Write a slave address to ***IDSR***
2. Set the ***ICSR*** as the Master Receiver mode **(*ICSR* &= 1F0Fh and *ICSR* |= 10B0h).**
3. Write ***IRQ_PEND.OP_HOLD*** = 1 to generate a START condition.
4. Wait until an interrupt occurs. (***IRQ_PEND.PEND***= =1).
5. If an interrupt occurs, check the ACK signal.
6. If the ACK signal is received normally, write a sub address to the ***IDSR***.
7. Write '1' to ***IRQ_PEND.OP_HOLD*** to transmit the sub address.
8. Check the ACK signal.
9. If the ACK signal is received normally, the data starts to be read.
10. IRQ mode data reception
    1) Read data from the ***IDSR***.
    2) Check if the data is the most recent data.
    3) If the data is the most recent, write '1' to ***ICCR.MASTER.SLV*** to prevent the generation of the ACK signal.
    4) Otherwise, write '1' to ***IRQ_PEND.OP_HOLD*** to receive the next data.
    5) Wait for IRQ
    6) Repeat Step 1 to 5) until reception is finished.
    7) After reception, set the STOP condition to the ***ICSR***. **(*ICSR* &= 1F0Fh, *ICSR* |= 1090h)**
    8) Write '1' to ***IRQ_PEND.OP_HOLD*** to generate the Stop condition.
    9) Clear all registers to disable the I^2C controller.

17.2.4. Example of the operation in Slave mode

17.2.4.1. Slave Transmitter and Receiver Mode data format

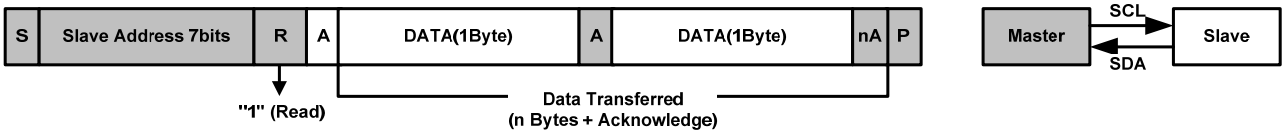


Figure 17-8. Slave Transmitter Mode Data Format

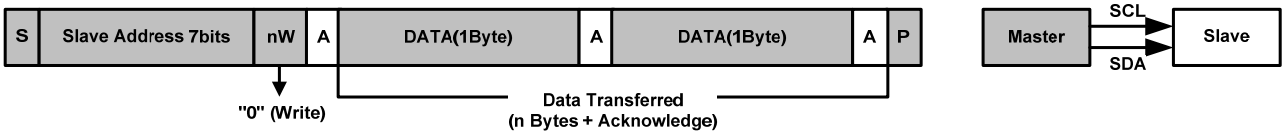


Figure 17-9. Slave Receiver Mode Data Format

17.2.4.2. Programming Examples

- **Initialize the I^2C controller.**

1. Clear the I^2C control register (***ICCR***).
2. Set the clock source (***ICCR.CLK_SRC = 0 → PCLK/16, ICCR.CLK_SRC = 1 → PCLK/256)***
3. Set clock prescaler (***ICCR.CLK_SCALER)***
4. Set the transition timing to the ***QCNT_MAX*** register. (recommend 1~5)
5. Set a slave address to ***IAR (Ex : Write ID(F8), Read ID(F9))***
6. Write '1010h' to the I^2C status register (***ICSR***).
7. Write ***IRQ_PEND.OP_HOLD*** =1 to generate a STOP condition

- **Write to master device**

1. Set the ***ICSR*** as the Slave Transmitter mode (Ack generation)**(*ICSR* &= 1F0Fh, *ICSR* |=1070h).**
2. Wait until an interrupt occurs (***IRQ_PEND.PEND*** = 1).
3. If an interrupt is generated, check if the slave address match occurs, too. (***ICSR.SLAVE_MATCH_OCCUR*** = 1).
4. If ***ICSR.SLAVE_MATCH_OCCUR*** = 1, the data is transmitted to the master device.
5. IRQ mode data transmission.
   1) Check ***ICSR.ACK_STATUS*** and repeat Steps 2) to 5) until the data transmission is finished.
   2) Write data to ***IDSR***.
   3) Clear the pending bit to resume.
   4) Write '1' to ***IRQ_PEND.OP_HOLD*** to transmit data.
   5) Wait for IRQ.
   6) When the data transmission is finished, initialize SCL and SDA. **(*ICSR* &=1010h)**
   7) Write '1' to ***IRQ_PEND.OP_HOLD***.
   8) Disable the I^2C controller.

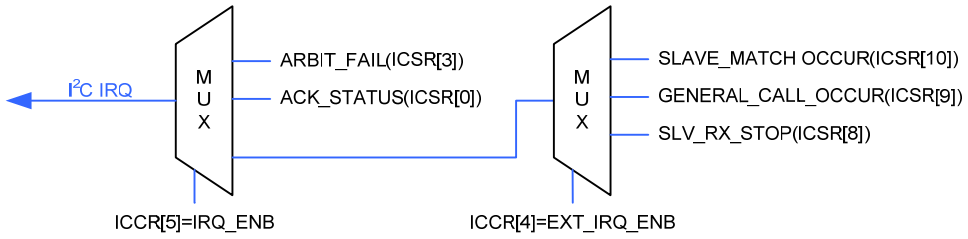
- **Read from master device**

1. Set the ICSR as the Slave receiver mode (Ack generation) **(*ICSR* &= 1F0Fh and *ICSR* |= 1010h).**
2. Wait until an interrupt occurs (***IRQ_PEND.PEND***= =1).
3. If an interrupt occurs, check if the Slave address match occurs, too. (***ICSR.SLAVE.MATCH.OCCUR*** = 1).
4. If a Slave address match occurs, read data.
5. IRQ mode data reception.
   1) Check ***ICSR.ACK_STATUS*** and repeat Steps 2) to 5) until the data transmission is finished.
   2) Read data from the ***IDSR***.
   3) Clear a pending bit to resume
   4) Write '1' to ***IRQ_PEND.OP_HOLD*** to transmit data.
   5) Wait for IRQ.
   6) When the data transmission is finished, initialize SCL and SDA **(*ICSR* &= 1010h).**
   7) Write '1' to ***IRQ_PEND.OP_HOLD***.
   8) Disable the I^2C controller.

17.3. Register Summary

Bit	R/W	Symbol	Description	Reset Value
I²C CONTROL REGISTER (ICCR) ***Address : I2C0:C000_E000h / I2C1:C000_E800h***				
[31:8]	-	**RESERVED**	Reserved	24'b0
[7]	R/W	**ACK_GEN**	ACK generation enable/disable (Use only for Receiver mode.) 0 : Disable ACK generation 1 : Enable ACK generation	1'b0
[6]	R/W	**CLK_SRC**	I²C Clock source select. 0 : PCLK/16 1 : PCLK/256	1'b0
[5]	R/W	**IRQ_ENB** (1)	Tx/RX IRQ enable/disable 0 : Disable IRQ 1 : Enable IRQ I2C Interrupt occurs whenever every byte transfers if ICCR[IRQ_ENB] is set.	1'b0
[4]	R/W	**RESERVED**	Reserved. This bit must be 0.	1'b 0
[3:0]	R/W	**CLK_SCALER** (2)	Clock prescaler. Master SCL clock = clock source/(CLK_SCALER+ 1) Clock source : : Is selected by bit 6 (CLK_SRC) of ICCR. For CLK_SRC = 0 : The clock source is PCLK/16 where the range of the CLK_SCALER values should be between 1 and 15. For CLK_SRC = 1 : The clock source is PCLK/256 where the range of the CLK_SCALER values should be between 0 and 15.	4'b 0

NOTES
(1) I²C IRQ

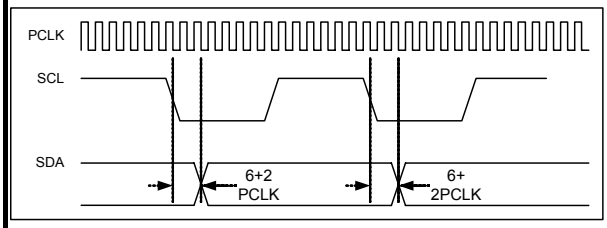
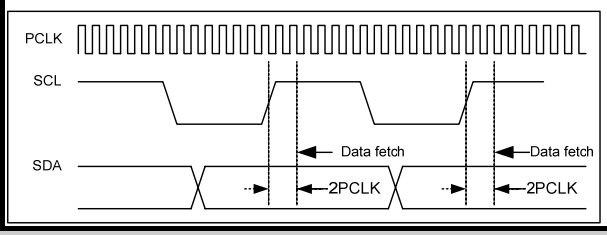


(2) Example) I²C Clock Table (where PCLK = 50MHz), I²C Clock = Clock Sourc(CLK_SCALER + 1)

ICCR[6]:CLK_SRC	ICCR[3:0]:CLK_SCALER	I²C Clock (Hz)	ICCR[6]:CLK_SRC	ICCR[3:0]:CLK_SCALER	I²C Clock (Hz)
-	-	–	1	0	195312
0	1	1562500	1	1	97656
0	2	1041667	1	2	65104
0	3	781250	1	3	48828
0	4	625000	1	4	39062
0	5	520833	1	5	32552
0	6	446428	1	6	27901
0	7	390625	1	7	24414
0	8	347222	1	8	21701
0	9	312500	1	9	19531
0	10	284090	1	10	17755
0	11	260416	1	11	16276
0	12	240384	1	12	15024
0	13	223214	1	13	I13950
0	14	208333	1	14	13020
0	15	195312	1	15	12207

I²C STATUS REGISTER (ICSR) ***Address : I2C0:C000_E004h / I2C1:C000_E804h***				
[31:13]	-	**RESERVED**	Reserved	18'b0
[12]	W	**ST_ENB**	Start/Stop enable : In ICSR write operations, the start/stop generation of ICSR[5] is enabled only when this bit is '1'. 0 : Start/Stop disable 1 : Start/Stop enable	1'b0

Bit	R/W	Symbol	Description	Reset Value
[11]	-	**RESERVED**	Reserved	1'b0
[10]	R/W	**SLAVE_MATCH_OCCUR**	Slave address match occurs : When the addressed value stored in IAR and the address value of IDSR are equal, this bit is set as '1'. (Clear ST_ENB bit before setting) READ> 0 : Not match occur 1 : Match occur WRITE> 0 : Clear 1 : Don't write Do not write '1' to this bit. If this bit is written as '1', an interrupt condition is created and IRQ occurs.	1'b0
[9]	R/W	**GENERAL_CALL_OCCUR**	General call occurs : In Slave mode, this bit is set as '1' when IAR = 0. (Clear ST_ENB bit before setting) READ> 0 : Not general call 1 : General call (slave address=0) WRITE> 0 : Clear 1 : Don't write Do not write '1' to this bit. If this bit is written as '1', an interrupt condition is created and IRQ occurs.	1'b0
[8]	R/W	**SLV_RX_STOP**	Slave Rx stop occurs : In Slave Receiver mode, this bit is set as '1' when the STOP condition occurs from the master device. (Clear ST_ENB bit before setting) READ> 0 : Continue 1 : Stop WRITE> 0 : Clear 1 : Don't write Do not write '1' to this bit. If this bit is written as '1', an interrupt condition is created and IRQ occurs.	1'b0
[7]	R/W	**MASTER_SLV**	Master/Slave mode Select 0 : Slave mode 1 : Master mode All values of ICSR [7:4] should be set at a time.	1'b0
[6]	R/W	**TX_RX**	Transmit//Receive mode Select 0 : Receive 1 : Transmit This bit is automatically changed to the proper mode by the LSB of slave address which is the first byte transferred just after the start condition under the slave mode(ICSR[MASTER_SLV]=0). All values of ICSR [7:4] should be set at a time.	1'b0
[5]	R/W	**ST_BUSY**	Busy status Read> 0 : Not busy 1 : Busy (Bus is in operating) Master Mode : Write> 0 : Stop signal generation 1: Start signal generation Slave Mode : Don't care When bit 12 of ICSR is '1', Start/Stop generation is enabled. All values of ICSR [7:4] should be set at a time.	1'b0
[4]	R/W	**TXRX_ENB**	Tx/Rx enable (Define bus usage) 0 : Disable Tx/Rx 1 : Enable Tx/Rx All values of ICSR [7:4] should be set at a time.	1'b0
[3]	R	**ARBIT_FAIL**	Arbitration fail status (It is cleared when I2C Start or Stop) 0 : Bus arbitration success 1 : Bus arbitration fail	1'b0
[2]	-	**RESERVED**	Reserved	1'b0
[1]	-	**RESERVED**	Reserved	1'b0
[0]	R	**ACK_STATUS**	ACK status. (Invoke interrupt when NACK occurs) 0 : ACK received 1 : ACK not received	1'b0
I²C ADDRESS REGISTER (IAR) ***Address : I2C0:C000_E008h / I2C1:C000_E808h***				
[31:8]	-	**RESERVED**	Reserved	24'b0
[7:1]	W	**SLV_ADDR**	7 bit slave address : IAR is used only when the I²C mode is selected to slave Receiver/Transmitter mode. This value can be written only when ICSR[4](TXRX_ENB) is 0.	7'b0
[0]	-	**RESERVED**	The bit[0] of data which is transferred just after the start condition is determined by the mode selection bit. If the mode selection bit is 'Receive', the bit[0] will be 1(read case). If the mode selection bit is 'Transmit', the bit[0] will be 0(write case)	1'b0
I²C DATA REGISTER (IDSR) ***Address : I2C0:C000_E00Ch / I2C1:C000_E80Ch***				
[31:8]	-	**RESERVED**	Reserved	24'b0
[7:0]	R/W	**BYTE_DAT**	Byte data for Tx/Rx IDSR does not contain the slave address which is the first byte transferred just after the start condition under the slave mode(ICSR[MASTER_SLV]=0).	8'b0
I²C QUARTER PERIOD COUNT MAX REGISTER (QCNT_MAX) ***Address : I2C0:C000_E010h / I2C1:C000_E810h***				

Bit	R/W	Symbol	Description	Reset Value
[31:5]	-	**RESERVED**	Reserved	25'b0
[4:0]	R/W	**CNT_MAX**	Quarter period count max. (SDA Timing Control Function) Tx case : If the PCLK counter increases from the falling edge of SCL until the counter equals the value of QCNT_MAX, SDA is output. Example) If CLK_SRC = 0(PCLK/16), CLK_SCALER = 9, PCLK = 50MHz and QCNT_MAX = 2, one clock cycle of SCL and PCLK is 32us and 20 ns, respectively.  Rx case : : If the PCLK counter increases from the rising edge of SCL until the counter equals the value of QCNT_MAX, SDA is input. Example) If CLK_SRC = 0(PCLK/16), CLK_SCALER = 9, PCLK = 50MHz and QCNT_MAX = 2, one clock cycle of SCL and PCLK is 32us and 20 ns, respectively. 	5'b0
RESERVED ***Address : I2C0:C000_E014h~C000_E020h / I2C1:C000_E818h~C000_E820h***				
[31:0]	-	**RESERVED**	Reserved	32'b0
I²C IRQ PEND (IRQ_PEND) ***Address : I2C0:C000_E024h / I2C1:C000_E824h***				
[31:2]	-	**RESERVED**	Reserved	30'b0
[1]	W	**OP_HOLD**	Operation hold. WRITE> 0 : No affect 1 : Clear operation hold. This bit is simultaneously set with the IRQ_PEND bit. If this bit is cleared, the next byte starts to be transmitted/received. Clear in the last step of the IRQ routine.	1'b0
[0]	R/W	**PEND**	I²C interrupt pend register. (It is generated by each 1byte transfer) READ> 0 : None 1 : Interrupt pended WRITE> 0 : No affect 1 : Clear pending bit	1'b0
RESERVED ***Address : I2CBase Address + 28h ~ Address : I2CBase Address + FCh*** ***Address : I2C0:C000_E028h~C000_E0FCh / I2C1:C000_E828h~C000E8FCh***				
I²C CLOCK ENABLE REGISTER (I2C_CLKENB) ***Address : I2C0:C000_E100h / I2C1:C000E900h***				
[31:4]	-	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	PCLK operation mode 0 : Disable Clock 1 : Enable Clock	1'b1
[2:0]	-	**RESERVED**	Reserved	3b100

CHAPTER 18.

PULSE WIDTH MODULATOR (PWM)

18. PULSE WIDTH MODULATOR (PWM)

18.1. Overview

The POLLUX has three Pulse Width Modulators (PWM). Each PWM operates respectively and is controlled by its own register set. Each channel outputs the pulse width modulated signal via an external pin. Four PWMs have the same circuit architectures and, hereinafter, will be described as PWMn (n = 0, 1, 2)

Each PWM Channel has a 7-bit Prescaler. The Prescaler receives the Clock provided by the Power Management and divides the Clock with the value specified by the ***PWMnPERIOD*** and then produces the Clock for carrying out the PWM operation.

In addition, each PWM Channel has a Counter composed of 10 bits. The CPU can produce the desired signal by specifying the period and the duty of a PWM Waveform in the ***PWMnPERIOD*** and ***PWMnDUTY*** registers.

When the specified value in the ***PWMnPERIOD*** corresponds to the Counter value, the PWM resets the Counter and sets the PWM Signal to '0'.

When the specified value in the ***PWMnDUTY*** corresponds to the Counter value, the PWM sets the PWM Signal to '1'.

The ***PWMnPOL*** determines whether to Invert or Bypass the value before the PWM Signal produced in the pertinent Channel goes out through the external PAD.

18.1.1. Features

- 3 Channel Pulse Width Modulator channels
- 7-bit Clock divider & 10-bit Period counter
- 10-bit Duty counter

18.1.2. Block Diagram (n=0, 1, 2)

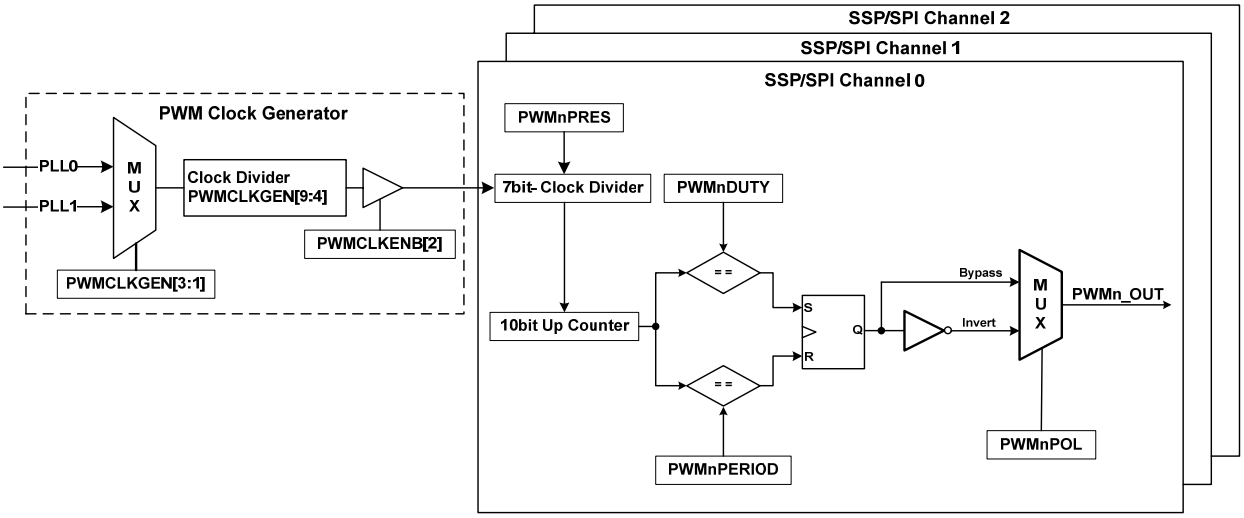


Figure 18-1. PWM Block Diagram

18.1.3. Pin Function Description

Pin Name	GPIO No	GPIO Function	Type	Description
PWMOUT[2]	C[7]	ALT1	O	Pulse Width Modulator channel 2
PWMOUT[1]	A[31]	ALT1	O	Pulse Width Modulator channel 1
PWMOUT[0]	A[30]	ALT1	O	Pulse Width Modulator channel 0

Table 18-1. PWM Pin Function Description

18.2. Operation

As shown below, the 10-bit Counter is reset by the ***PWMnPERIOD*** and the PWM Signal is changed to '0'. When the ***PWMnDUTY*** register value corresponds to the Counter value, the PWM Signal is set to '1'. In bypass mode, if the value specified by the ***PWMnDUTY*** is bigger than the ***PWMnPERIOD*** value, the PWM Signal is '0' and in the invert mode, if the value specified by the ***PWMnDUTY*** is bigger than the ***PWMnPERIOD*** value, the PWM Signal is '1'. In Bypass mode, if the ***PWMnDUTY*** value is specified as '0', the PWM signal is always output as '1'. In the Invert mode, if the ***PWMnDUTY*** value is specified as '0', the PWM signal is always output as '0'.
PWM signal output can be selected in Bypass mode or Invert mode by using the PWM Prescaler register (***PWMnPOL***).

18.2.1. PWM Signal Timing

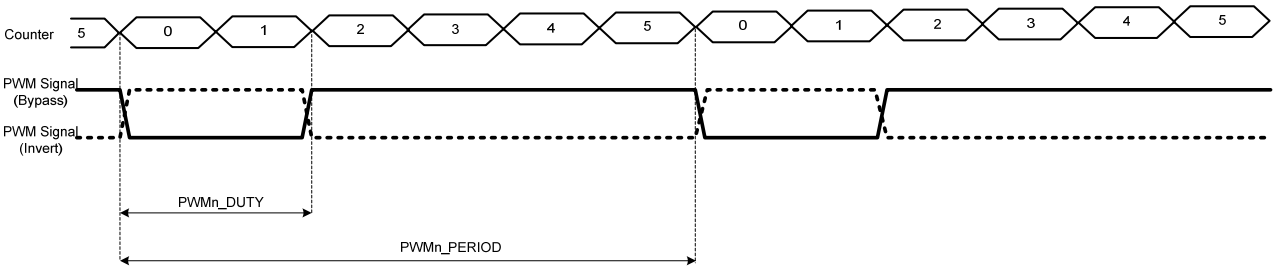


Figure 18-2. PWM Signal Timing

Figure 18-2 shows a timing chart in which the ***PWMnDUTY*** and the ***PWMnPERIOD*** values are specified as '2' and '6', respectively.

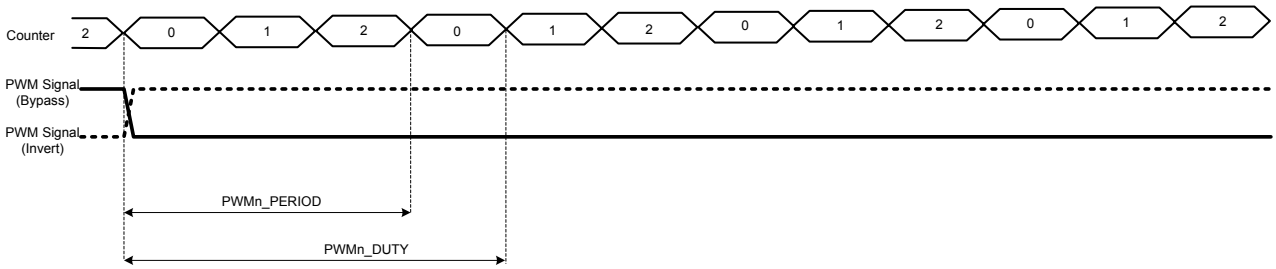


Figure 18-3. Example of a timing chart in which the PWMnDUTY value is bigger than the PWMnPERIOD value

Figure 18-3 shows a PWM signal where the ***PWMnDUTY*** and the ***PWMnPERIOD*** values are specified as '3'. If the ***PWMnDUTY*** value is bigger than the ***PWMnPERIOD*** value, the counter is reset by the ***PWMnPERIOD*** before the counter is set by the ***PWMnDUTY***. In other words, since the SR Flip-Flop in the block diagram of Figure 18-1 is always in Reset mode, the flip-flop outputs '0' and the PWM signal always maintains '0' (Bypass case) or '1' (Invert case).

18.2.2. Flowchart

Figure 18-4 shows a flowchart when PWM Channel 0 is used.

1) Select GPIOB[28] as Alternate Function1.

2) Set the clock by using the PWM clock generator.

3) Set ***PWMCLKENB***.

4) Set ***PWM0PRES***.

5) Set ***PWM0DUTY*** and ***PWM0PERIOD*** values.

6) Output the Invert or Bypass signals.

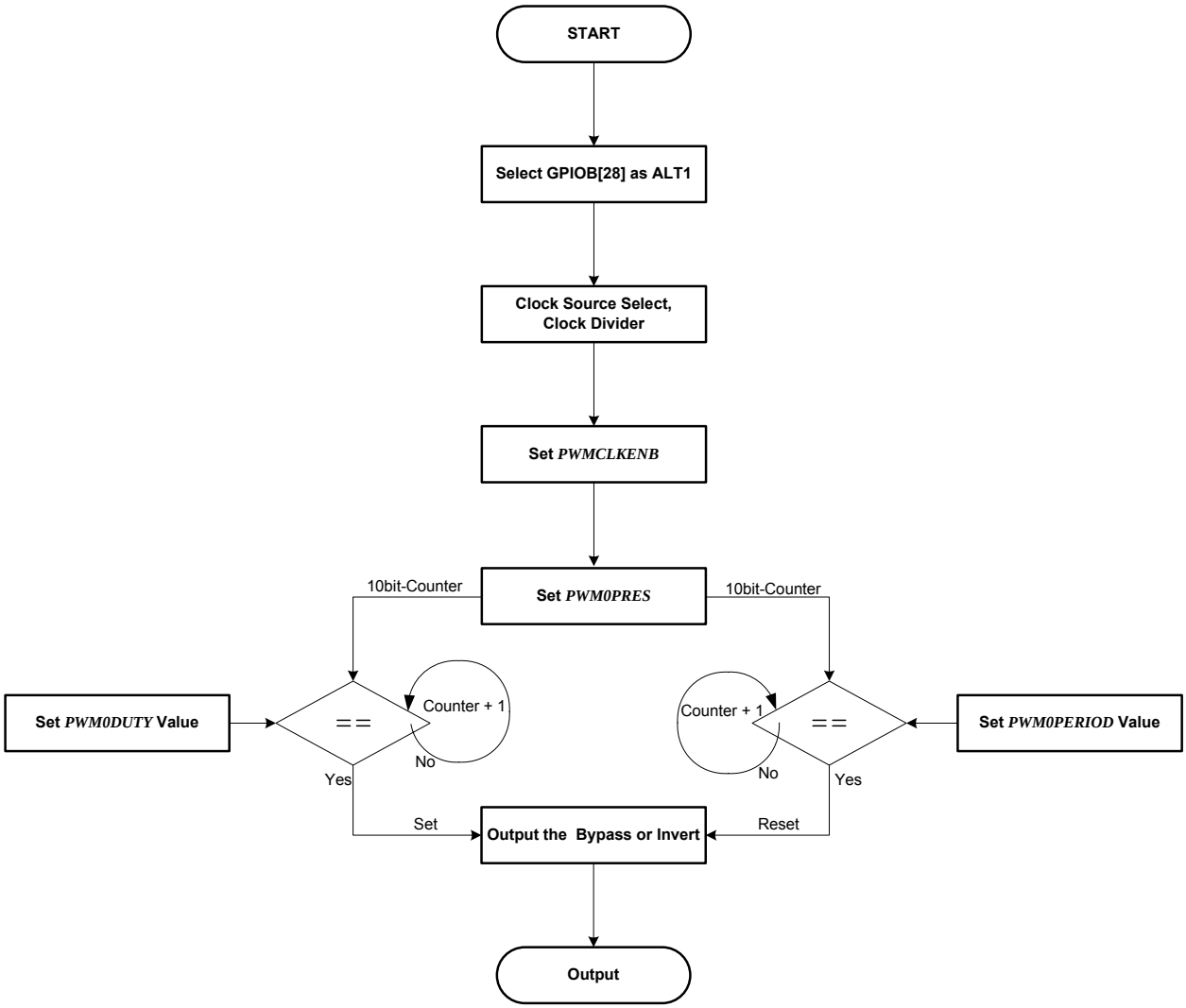


Figure 18-4. PWM Flowchart

18.3. Register Summary

Bit	R/W	Symbol	Description	Reset Value
PWM PRESCALER0/1 REGISTER (PWM01PRES) ***Address: C000_C000h***				
[15]	R/W	**PWM1POL**	This bit determines the polarity of PWM1 signal 0 : Invert 1 : Bypass	1'b0
[14:8]	R/W	**PWM1PRESCALE**	These 7 bits determine the prescaler value for PWM1 Ex.) The Prescale value is between 0 and 127.	7'b0
[7]	R/W	**PWM0POL**	This bit determines the polarity of PWM0 signal 0 : Invert 1 : Bypass	1'b0
[6:0]	R/W	**PWM0PRESCALE**	These 7 bits determine the prescaler value for PWM0 Ex.) The Prescale value is between 0 and 127.	7'b0
PWM0 DUTY CYCLE (PWM0DUTY) ***Address: C000_C002h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9:0]	R/W	**PWM0DUTY**	These bits determine the range of the PWM0DUTY Signal. Caution) In Bypass mode, if PWM0DUTY is set as '0', the PWM0 output is always '1'. In Invert mode, if PWM0DUTY is set as '0', the PWM0 output is always '0'. Caution) In Bypass mode, if PWM0DUTY is set as a bigger value than that of PWM0PERID, the PWM0 output always is '0'. In Invert mode, if PWM0DUTY is set as a bigger value than that of PWM0PERID, the PWM0 output is always '1'.	10'b0
PWM1 DUTY CYCLE (PWM1DUTY) ***Address: C000_C004h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9:0]	R/W	**PWM1DUTY**	These bits determine the range of the PWM1DUTY signal. Caution) In Bypass mode, if PWM1DUTY is set as '0', the PWM1 output is always '1'. In Invert mode, if PWM1DUTY is set as '0', the PWM1 output is always '0'. Caution) In Bypass mode, if PWM1DUTY is set as a bigger value than that of PWM1PERID, the PWM1 output is always '0'. In Invert mode, if PWM1DUTY is set as a bigger value than that of PWM1PERID, the PWM1 output is always '1'.	10'b0
PWM0 PERIOD CYCLE (PWM0PERIOD) ***Address: C000_C006h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9:0]	R/W	**PWM0PERID**	Set the Period of the PWM0 Signal	10'b0
PWM1 PERIOD CYCLE (PWM1PERIOD) ***Address: C000_C008h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9:0]	R/W	**PWM1PERID**	Set the Period of the PWM1Signal	10'b0
RESERVED ***Address: C000_C00Ah ~ C000_C00Eh***				
PWM PRESCALER 2 REGISTER (PWM2PRES) ***Address: C000_C010h***				
[15:8]	R/W	**RESERVED**	Reserved	8'b0
[7]	R/W	**PWM2POL**	This bit determines the polarity of PWM2 signal 0 : Invert 1 : Bypass	1'b0

Bit	R/W	Symbol	Description	Reset Value
[6:0]	R/W	**PWM2PRESCALE**	These 7 bits determine the prescaler value for PWM2 Ex.) The Prescale value is between 0 and 127.	7'b0
PWM2 DUTY CYCLE (PWM2DUTY) ***Address: C000_C012h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9:0]	R/W	**PWM2DUTY**	These bits determine the range of the PWM2DUTY signal. Caution) In Bypass mode, if PWM2DUTY is set as '0', the PWM2 output is always '1'. In Invert mode, if PWM2DUTY is set as '0', the PWM2 output is always '0'. Caution) In Bypass mode, if PWM2DUTY is set as a bigger value than that of PWM2PERID, the PWM2 output is always '0'. In Invert mode, if PWM2DUTY is set as a bigger value than that of PWM2PERID, the PWM2 output is always '1'.	10'b0
RESERVED ***Address: C000_C014h***				
[15:0]	-	**RESERVED**	Reserved	16'b0
PWM2 PERIOD CYCLE (PWM2PERIOD) ***Address: C000_C016h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9:0]	R/W	**PWM2PERID**	Set the Period of the PWM2 Signal	10'b0
RESERVED ***Address: C000_C018h***				
[15:0]	-	**RESERVED**	Reserved	16'b0
RESERVED ***Address : C000_C01Ah ~ C000_C060h***				
PWM CLOCK ENABLE REGISTER (PWMCLKENB) ***Address : C000_C040h***				
[31 : 4]	-	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	PCLK operating mode 0 : Clock is Enabled only when CPU accesses 1 : Always	1'b0
[2]	R/W	**CLKGENENB**	Clock generation Enable 0 : Disable 1 : Enable	1'b0
[1:0]	-	**RESERVED**	Reserved	2'b0
PWM CLOCK GENERATE REGISTER (PWMCLKGEN) ***Address : C000_C044h***				
[15:10]	-	**RESERVED**	Reserved	6'b0
[9:4]	R/W	**CLKDIV**	Clock divider: The value of (CLKDIV +1) is given as the divider value; divides by 1 to 64 are available. CLK /[(CLKDIV : 0~63) + 1]	6'b0
[3:1]	R/W	**CLKSRCSEL**	Clock source selection 000 : PLL0 001 : PLL1 010 : PLL2 011 ~ 111 : Reserved	3'b0
[0]	-	**RESERVED**	Reserved	1'b0

CHAPTER 19.

ANALOG TO DIGITAL CONVERTER (ADC)

19. Analog to Digital Converter (ADC)

19.1. Overview

The ADC of POLLUX is a CMOS 3.3V, 10-bit Analog-to-Digital Converter. It converts the analog input signal into 10-bit binary digital codes at a maximum conversion rate of 500KSPS with 2.5MHz clock.

The ADC block is a recycling type monolithic ADC with an on-chip sample-and-hold function. The POLLUX has eight ADC input channels. However, it cannot perform AD conversion for all channels simultaneously. It can only perform AD conversion for one channel at a time. In addition, the POLLUX only supports Power Down mode for the ADC block.

19.1.1. Features

- Resolution : 10bit
- Maximum Conversion Rate : 500KSPS (Samples Per Sec)
- Main Clock : 2.5MHz (Max.)
- Power Supply : 3.3V ±0.3V, 1.3V ±0.1V(Digital I/O Interface)
- Total Current : 20uA (Standby mode)
  2.0mA (Normal operation)
- Input Range : 0.0V ~ 3.3V ($3.3V_{P-P}$)
- Differential Linearity Error : ±1.0 LSB
- Integral Linearity Error : ±2.0 LSB
- Signal to Noise & Distortion Ratio : 54dB (Typ.)

19.1.2. ADC Block Diagram

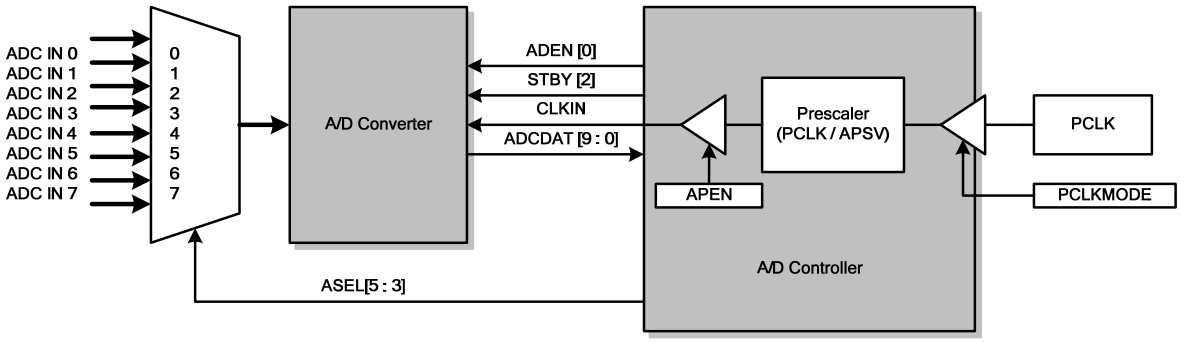


Figure 19-1. ADC Block Diagram

19.2. Operation

The POLLUX can receive eight ADC inputs and select one of the ADC inputs by using ***ADCCON.ASEL***[5:3]. The sizes of the eight ADC inputs depend on the size of V_{REF} and an ADC input voltage between 2.0 V and 3.6 V can be applied. (Recommended value: 3.3V)

The ADC controller uses PCLK and the PCLK is divided in the Prescaler and applied to the ADC. At this time, the ***ADCCON.APEN*** bit is used for the application of CLKIN. Clock divide values by the Prescaler are available from 20 to 256. (Actually, since the register input value is [Clock Divide Value – 1], smaller divide values make the sampling more detailed. The clock divide value is determined by ***ADCCON.APSV***bit.

If the ADC block continuously accepts after power is applied, it consumes current unnecessarily. In this case, it is better to power down the A/D converter by using the ***ADCCON.STBY*** bit. The ***ADCCON.STBY*** bit determines the power input for the ADC block. If the ***ADCCON.STBY*** bit is '0', the ADC block waits for ADC input after power on. After that, if the ***ADCCON.ADEN*** bit is set as '1' to accept ADC input, the ADC operation is actually performed. On the other hand, if the ***ADCCON.STBY*** bit is '1', the ADC block goes to power down status, that is, power is not applied. This is called Standby mode. (In Standby mode, only about 20 uA current is consumed.

If the ADC block is not used, set the ***ADCCON.STBY*** bit as '1' and power off the ADC block. In this way, unnecessary power consumption by the ADC block can be reduced. In addition, since the supply of the clock can be determined by using the ***ADCCON.APEN*** bit, power consumption can be reduced further by stopping the clock supply when supply is unnecessary.

19.2.1. Signal Description

Name	I/O Type	I/O PAD	Pin Description
V_{REF}	Analog Input(AI)	Phia_abb	Reference Voltage. 2.0V ~ 3.6V. (Recommend Value : 3.3V)
AGND	Analog Input(AI)	Phia_abb	Reference GND (0.0V)
ADC IN[7 : 0]	Analog Input(AI)	Phiar50_abb	Analog Input. (Input Range : 0.0V ~ 3.3V)

Table 19-1. Signal Description

<Note>

The analog input is single-ended type and the range is from V_{REF} to AGND. This analog input voltage follows reference voltage range fundamentally. So, if you want to alter into another input range, you should change the voltage value of V_{REF}.

You can use the analog input voltage whose minimum range is 2.0V. In this case, the V_{REF} is 2.0V

19.2.2. Absolute Maximum Ratings

Characteristics	Symbol	Value	Unit
Supply Voltage	VDD33	4.8	V
Analog Input Voltage	AI	VSS to VDD	V
Digital Input Voltage	CLKIN	VSS to VDD	V
Reference Voltage	VREF/ AGND	VSS to VDD	V
Storage Temperature Range	Tstg	-45 to 150	℃
Operating Temperature Range	Topr	-40 to 85	℃

Table 19-2. Absolute Maximum Ratings

<Note>

Absolute maximum rating specifies the values beyond which the device may be damaged permanently. Exposure to absolute maximum rating conditions for extended periods may affect reliability. Each condition value is applied with the other values kept within the following operating conditions and function operation under any of these conditions is not implied.

- All voltages are measured with respect to VSS unless otherwise specified.
- 100pF capacitor is discharged through a 1.5KΩ resistor (Human body model)
- VDD33 = VDD33A_ADC, VDD33D_ADC, VSS = VSS33A_ADC, VSS33D_ADC

19.2.3. Recommended Operating Conditions

Characteristics	Symbol	Min	Typ	Max	Unit
Supply Voltage	VDD33A_ADC VDD33D_ADC	3.0	3.3	3.6	V
	VDD13A	1.17	1.3	1.43	V
Reference Input Voltage	V_{REF} VSS	2.0 0.0	3.3 0.0	3.6 0.0	V
Analog Input Voltage	ADC IN	0.0	V_{REF}		V
Operating Temperature	Toper	-40		85	℃

Table 19-3. Recommended Operating Conditions

<Note>

It is strongly recommended that all the supply pins(VDD33A_ADC, VDD33D_ADC) be powered from the same source to avoid power latch-up.

19.2.4. DC Electrical Characteristics

Characteristics	Symbol	Min	Typ	Max	Unit	Test Condition
Differential Nonlinearity	DNL	-			LSB	V_{REF} = 3.3V AGND = 0.0V
Integral Nonlinearity	INL	-			LSB	V_{REF} = 3.3V AGND = 0.0V
Offset Voltage	TOPOFF BOTOFF	-			LSB	V_{REF} = 3.3V AGND = 0.0V

Table 19-4. DC Electrical Characteristic

<Note>

Converter Specifications : VDD33A_ADC = VDD33D_ADC = 3.3V, VDD13A = 1.3V, VSS33A_ADC, VSS33D_ADC = 0V, Toper = 25℃, V_{REF} = 3.3V, AGND = 0V unless otherwise specified.

19.2.5. AC Electrical Characteristics

Characteristics	Symbol	Min	Typ	Max	Unit	Test Condition
Maximum Conversion Rate	f_C				KSPS	f_{CKIN} = 2.5MHz
Standby Supply Current	-				uA	STBY = VDD
Dynamic Supply Current	IVDD				mA	f_{CKIN} = 2.5MHz (Without system load)
Reference Current	IREF				mA	V_{REF} = 3.3V
Total Harmonic Distortion	THD				dB	f_{CKIN} = 2.5MHz AINT = 50KHz
Signal-to-Noise & Distortion Ratio	SNDR				dB	f_{CKIN} = 2.5MHz AINT = 50KHz

Table 19-5. AC Electrical Characteristics

<Note>

Converter Specifications : VDD33A_ADC = VDD33D_ADC = 3.3V, VDD13A = 1.3V, VSS33A_ADC, VSS33D_ADC = 0V, Toper = 25℃, V_{REF} = 3.3V, VSS = 0V unless otherwise specified.

19.2.6. I/O Chart

Index	ADC Input (V)	Digital Output	
0	~ 0.00322	00_0000_0000	1LSB = 3.22mV V_{REF} = 3.3V AGND = 0.0V
1	0.00322 ~ 0.00644	00_0000_0001	
2	0.00644 ~ 0.00967	00_0000_0010	
~	~	~	
511	1.64678 ~ 1.65000	01_1111_1111	
512	1.65000 ~ 1.65322	10_0000_0000	
513	1.65322 ~ 1.65644	10_0000_0001	
~	~	~	
1021	3.29033 ~ 3.29355	11_1111_1101	
1022	3.29355 ~ 3.29678	11_1111_1110	
1023	3.29678 ~	11_1111_1111	

Table 19-6. I/O Chart

19.2.7. Timing

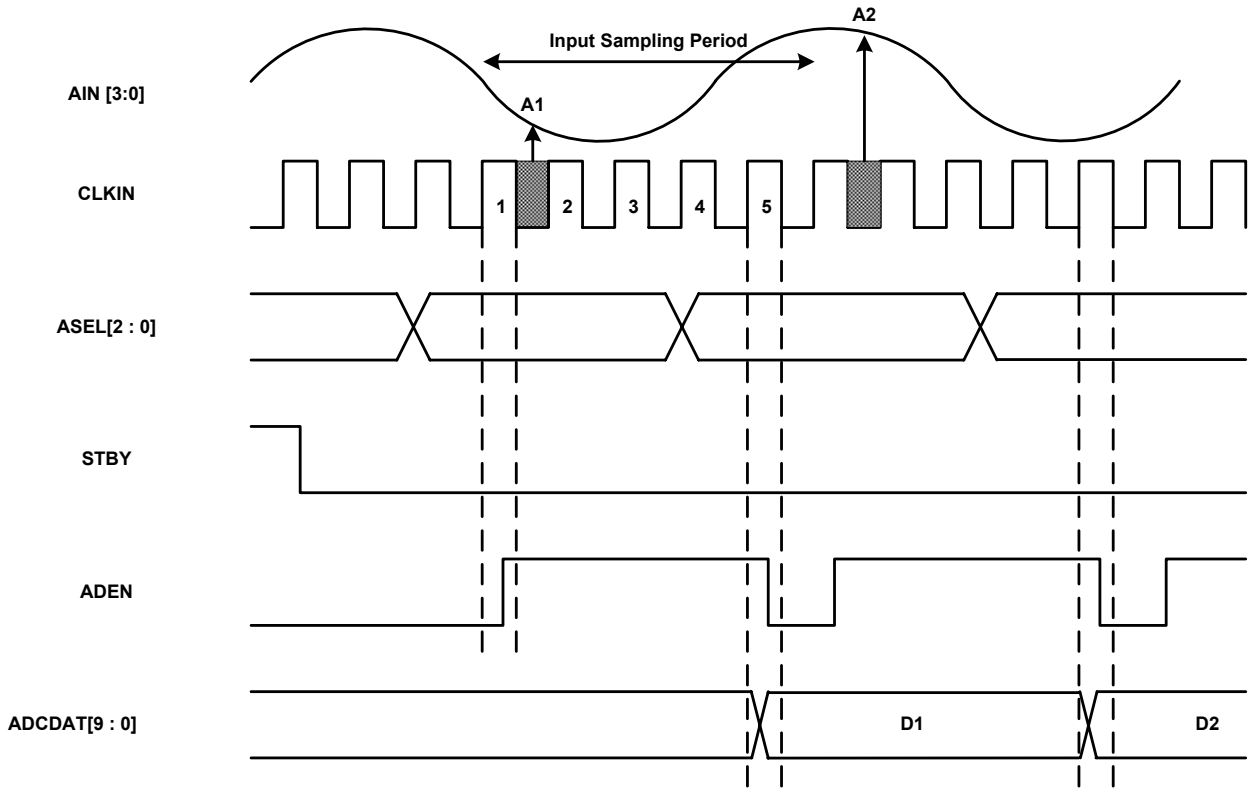


Figure 19-2. Main Waveform

Figure 19-2 shows the timing chart for the ADC. AIN[7:0] is continuously input from the outside and CLKIN is supplied via the ***ADCCON.APEN*** bit. After AIN[7:0] is selected, by using ASEL[2:0], the ***ADCCON.STBY*** bit is set as '0' to supply power to the ADC block. Finally, A/D conversion is progressed by setting the ***ADCCON.ADEN*** bit as '1'. After the conversion is completed, EDO occurs and the ***ADCCON.ADEN*** bit is automatically cleared to '0'. After that, A/D Converted Data (D1) can be read through ***ADCDAT.ADCDAT.*** Since it always takes 5-cycles for 10-bit conversion, the maximum conversion rate of the POLLUX is 500 KSPS. Set the ***ADCCON.ADEN*** bit as '1' to operate the ADC again.

19.2.8. Analog Input Selection Table

	ASEL[5]	ASEL[4]	ASEL[3]
Analog Input [0]	0	0	0
Analog Input [1]	0	0	1
Analog Input [2]	0	1	0
Analog Input [3]	0	1	1
Analog Input [4]	1	0	0
Analog Input [5]	1	0	1
Analog Input [6]	1	1	0
Analog Input [7]	1	1	1

Table 19-7. Analog Input Selection Table

19.2.9. Flowchart

1) PCLK Supply : ***CLKENB.PCLKMODE*** = 1
2) Analog Input Select : ***ADCCON.ASEL***
3) ADC Power On : ***ADCCON.STBY*** = 0
4) CLKIN Divide Value : ***ADCCON.APSV***
5) CLKIN On : ***ADCCON.APEN***
6) ADC Enable : ***ADCCON.ADEN***
7) A/D Conversion Process
8) Read ***ADCDAT.ADCDAT***
9) CLKIN Off
10) ADC Power Off

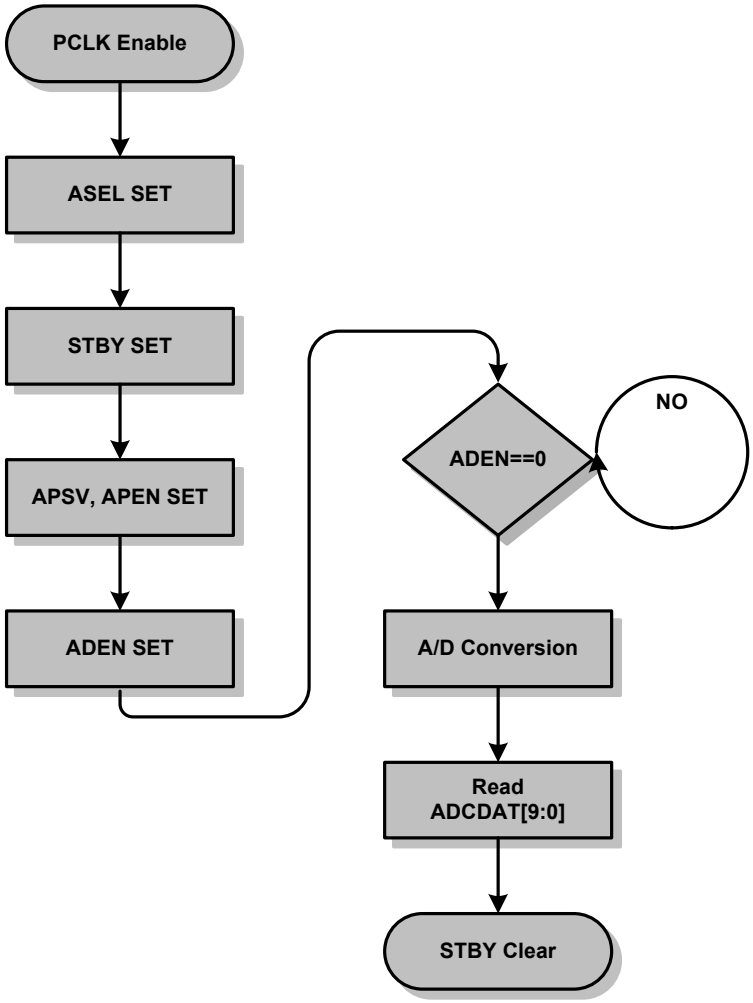


Figure 19-3. ADC Sequence Flowchart

19.3. Register Summary

Bit	R/W	Symbol	Description	Reset Value
ADC CONTROL REGISTER (ADCCON) ***Address : C000_5000h : WORD***				
[31:15]	R	**RESERVED**	Reserved	1'b1
[14]	R/W	**APEN**	Prescaler Enable. This bit determines the supply of the clock divided by the APSV register for the A/D converter. Before the APEN bit is enabled, the APSV register should be set. 0 : Disable 1 : Enable	1'b0
[13 : 6]	R/W	**APSV**	A/D Converter Clock Prescaler Value (8bit) - To write a value to this bit, APEN should be '0'. - The maximum value of the ADC CLK divided by the APSV value is 2.5 MHz (400 ns) (where PCLK is 50 MHz). - The minimum and the maximum value for the APSV bit are 19 and 255, respectively. (In effect, the range of the clock divide value is from 20 to 256). - Input Value = Clock Divide Value -1. For divide-by-20 and divide-by-100, (20-1) = 19 and (100-1) = 99 are input to APSV, respectively.	8'hFF
[5 : 3]	R/W	**ASEL**	These bits select ADCIN. POLLUX has four ADCINs and can select one of them. 000 : ADCIN_0 001 : ADCIN_1 010 : ADCIN_2 011 : ADCIN_3 100 : ADCIN_4 101 : ADCIN_5 110 : ADCIN_6 111 : ADCIN_7	3'b0
[2]	R/W	**STBY**	A/D Converter Standby Mode. If this bit is set as '0', power is actually applied to the A/D converter. 0 : ADC Power On 1 : ADC Power Off(Standby)	1'b1
[1]	-	**RESERVED**	Reserved	1'b0
[0]	R/W	**ADEN**	A/D Conversion Start - When the A/D conversion ends, this bit is cleared to '0'. Read > Check the A/D conversion operation. 0 : Idle 1 : Busy Write > Start the A/D conversion. 0 : None 1 : Start A/D conversion	1'b0
ADC OUTPUT DATA REGISTER (ADCDAT) ***Address : C000_5004h : WORD***				
[31 : 10]	-	**RESERVED**	Reserved	6'b0
[9 : 0]	R	**ADCDAT**	These bits are 10-bit data converted via the ADC.	10'b0
ADC INTERRUPT ENABLE REGISTER (ADCINTENB) ***Address : C000_5008h : WORD***				
[31 : 1]	-	**RESERVED**	Reserved	15'b0
[0]	R/W	**ADCINTENB**	ADC Interrupt Enable. (This bit determines the generation of an interrupt when EOC occurs.) This bit determines if interrupt occurs when the ADEN bit is '0'. 0 : Interrupt Disable 1 : Interrupt Enable	1'b0
ADC INTERRUPT PENDING AND CLEAR REGISTER (ADCINTCLR) ***Address : C000_500Ch : WORD***				
[31 : 1]	-	**RESERVED**	Reserved	15'b0
[0]	R/W	**ADCINTCLR**	EOC Interrupt Pending and Clear Read > 0 : None 1 : Interrupt Pended Write > 0 : None 1 : Pending Clear	1'b0

Bit	R/W	Symbol	Description	Reset Value
RESERVED ***Address : C000_5010h ~ 503Eh***				
CLOCK ENABLE REGISTER (CLKENB) ***Address : C000_5040h : WORD***				
[31 : 4]	-	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	PCLK Operating Mode. This bit should be set as '1' for the ADC operation. 0 : None 1 : Always Note : However, CLKENB[3] bit should be always be '1'.	1'b0
[2 : 0]	-	**RESERVED**	Reserved	3'b0

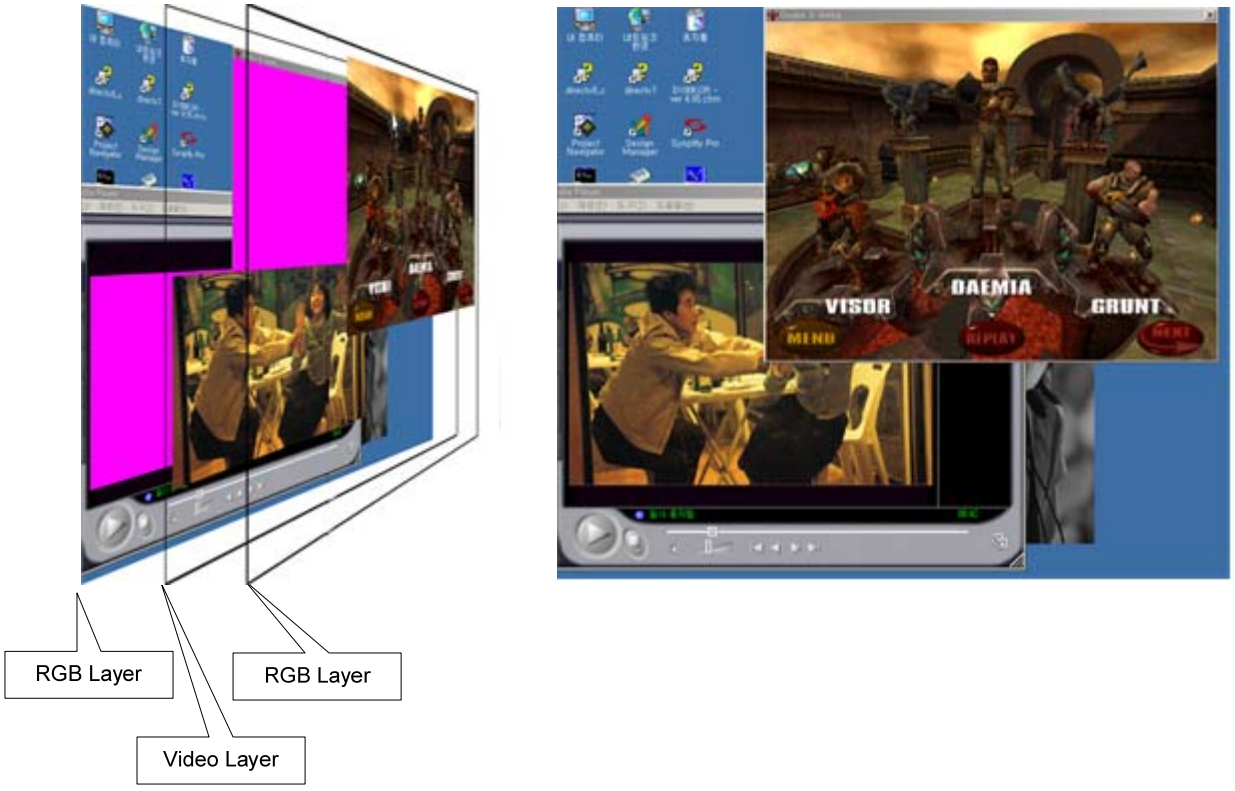
CHAPTER 20.

MULTI LAYER CONTROLLER (MLC)

20. MULTI LAYER CONTROLLER

20.1. Overview

The user screen is composed of complex components - RGB pictures, moving pictures, etc. These individual components have unique formats and are stored in their own memory spaces. The Multi Layer Controller (hereinafter, MLC) of the **POLLUX** reads and compounds various screen components in terms of Hardware, to organize a desired screen and transmits the result to the Display controller.



< Diaplay layers >

Figure 20-1. Concept of Multi Layer Controller

20.1.1. Features

- Dual register-set architecture
- Two RGB layers and one Video layer
- RGB layers can be used as 3D layers.
- Various pixel formats

  RGB layer : RGB/BGR 332, 444, 555, 565, 888 with/without Alpha (Palate Table 16x256)

  Video layer : 2D YUV 4:2:0
- Various blending effects between layers

  Per-layer or Per-pixel Alpha blending, Transparency, Inverse color
- Free layer position and size in pixel units
- Hardware clipping
- Vertical flip
- Video layer priority
- Scale-up/down (Video layer only)

  Bilinear interpolation, Nearest neighbor sampling scale-up/down
- Color control (Video layer only)

  Brightness, Contrast, Hue, Saturation

20.1.2. Block Diagram

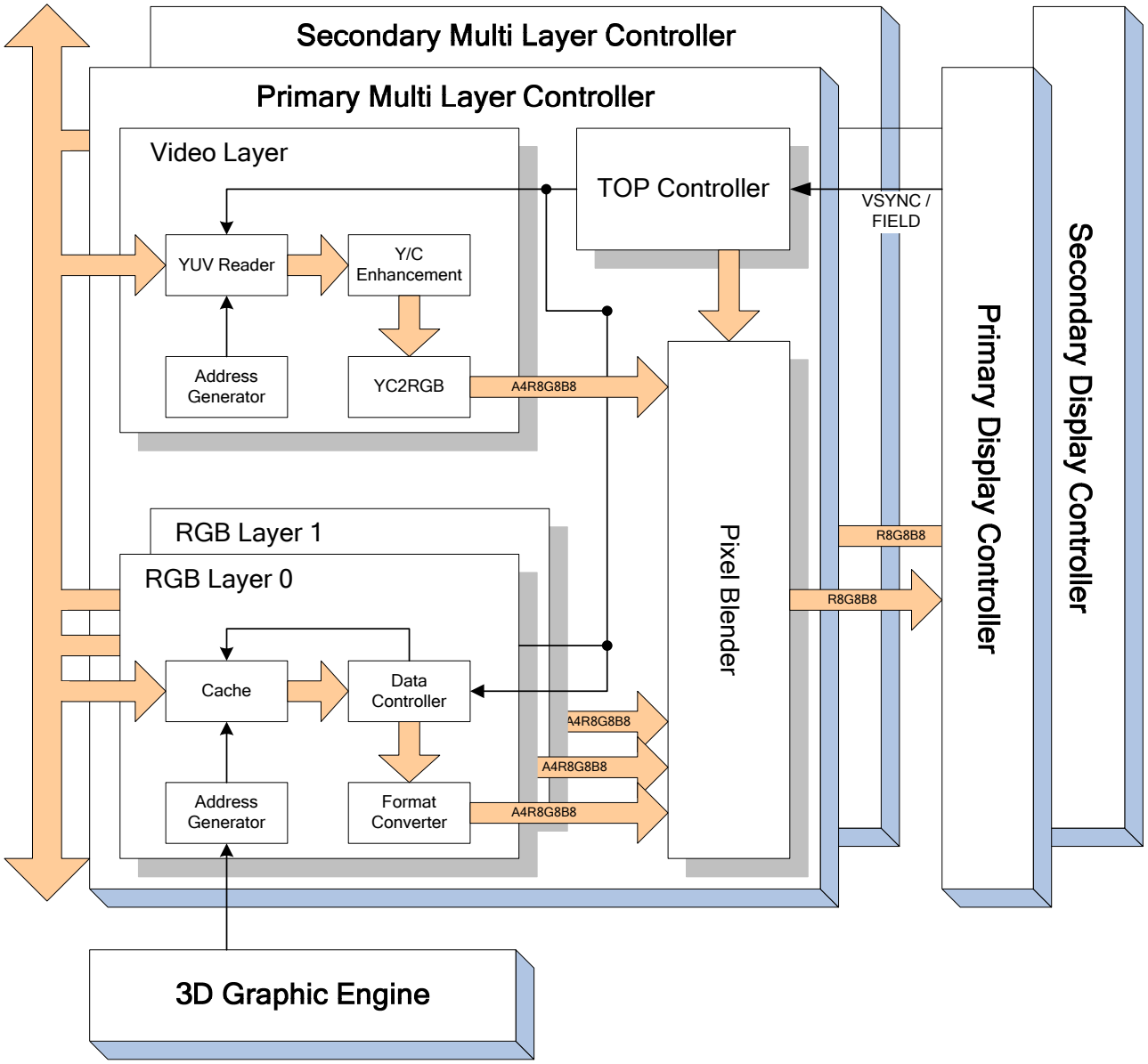


Figure 20-2. MLC Block Diagram

The MLC consists of two RGB layers and one Video layer. In the MLC, positions, pixel formats and various effects can be configured according to each layer. The Video layer supports the Scale function to display video images on various screen areas at certain sizes and various color control functions to provide optimal images. In addition, the **POLLUX** has a primary MLC and a secondary MLC to support 'dual display'.

20.2. Dual Register Set Architecture

The MLC of the **POLLUX** has dual register set architecture with a current working register group and a user side register group. All users can set registers via the user side register group. If the dirty flag is set as '1' after the user writes a desired setting to a register in the user side register group, the MLC copies the user side register group to the current working register group at the point when vertical sync occurs. Then the dirty flag is cleared to '0' and the user can continue to progress the next setting. In this way the changes in all registers are synchronized and applied to vertical sync so that the user can hide any abnormal screen, even if the user changes a register at any point.

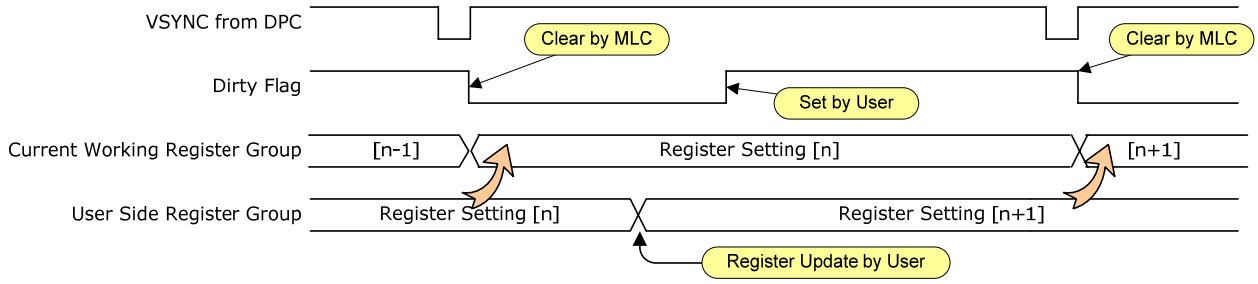


Figure 20-3. Dual Reigister Set Architecture

The Top controller and three layers of the MLC have separate dirty flag bits. Each dirty flag reflects the changes of the registers pertaining to the corresponding group.

Dirty flag	Numbers	Registers to be Affected
Top controller	1	MLCCONTROLT, MLCSCREENSIZE, MLCBGCOLOR
RGB layer	2	MLCLEFTRIGHTn, MLCTOPBOTTOMn, MLCCONTROLn, MLCHSTRIDEn, MLCVSTRIDEn, MLCTPCOLORn, MLCINVCOLORn, MLCADDRESSn, MLCLEFTRIGHTn_0, MLCTOPBOTTOMn_0, MLCLEFTRIGHTn_1, MLCTOPBOTTOMn_1
Video layer	1	MLCLEFTRIGHT2, MLCTOPBOTTOM2, MLCCONTROL2, MLCVSTRIDE2, MLCTPCOLOR2, MLCADDRESS2, MLCADDRESSCB, MLCADDRESSCR, MLCVSTRIDECB, MLCVSTRIDECR, MLCHSCALE, MLCVSCALE, MLCLUENH, MLCCHENH0, MLCCHENH1, MLCCHENH2, MLCCHENH3

Table 20-1. Dirty flag

20.3. MLC Global parameters

This section describes how to set the global parameters of the MLC.

Function	Symbol	Bit width	Register	Brief description
Priority	PRIORITY	2	MLCCONTROLT[9:8]	Specifies the priority of the Video layer.
Dual register set	DIRTYFLAGT	1	MLCCONTROLT[3]	Dirty Flag for MLC top controller.
Enable	MLCENB	1	MLCCONTROLT[1]	Specifies whether or not to enable MLC
Scan mode	FIEEDENB	1	MLCCONTROLT[0]	Specifies whether or not to enable Interlace mode
Screen size	SCREENWIDTH	12	MLCSCREENSIZE[11:0]	Specifies 'the whole screen width - 1'.
	SCREENHEIGHT	12	MLCSCREENSIZE[27:16]	Specifies 'the whole screen height - 1'.
Background color	BGCOLOR	24	MLCBGCOLOR[23:0]	Specifies the background color to be displayed on the screen in areas not covered by any of the layers

Table 20-2. Top controller registers

20.3.1. Screen size

The Screen size function enables users to specify the width and height of the whole screen to be displayed. The values of 'the whole screen width – 1' and 'the whole height – 1' are set to the ***SCREENWIDTH*** and ***SCREENHEIGHT***, respectively. Since each of the ***SCREENWIDTH*** and ***SCREENHEIGHT*** has 12-bit size units, the maximum available resolution is 2048 x 2048 pixels. The screen size is determined by setting the size in frame units regardless of whether the display is progressive or interlace.

20.3.2. Priority

The MLC consists of two RGB layers and one Video layer. Among the RGB layers, Layer 0 has the highest priority and Layer 1 has the lowest priority. These priorities cannot be changed, but the priority of the Video layer can be adjusted via the ***PRIORITY*** parameter at the user's discretion.

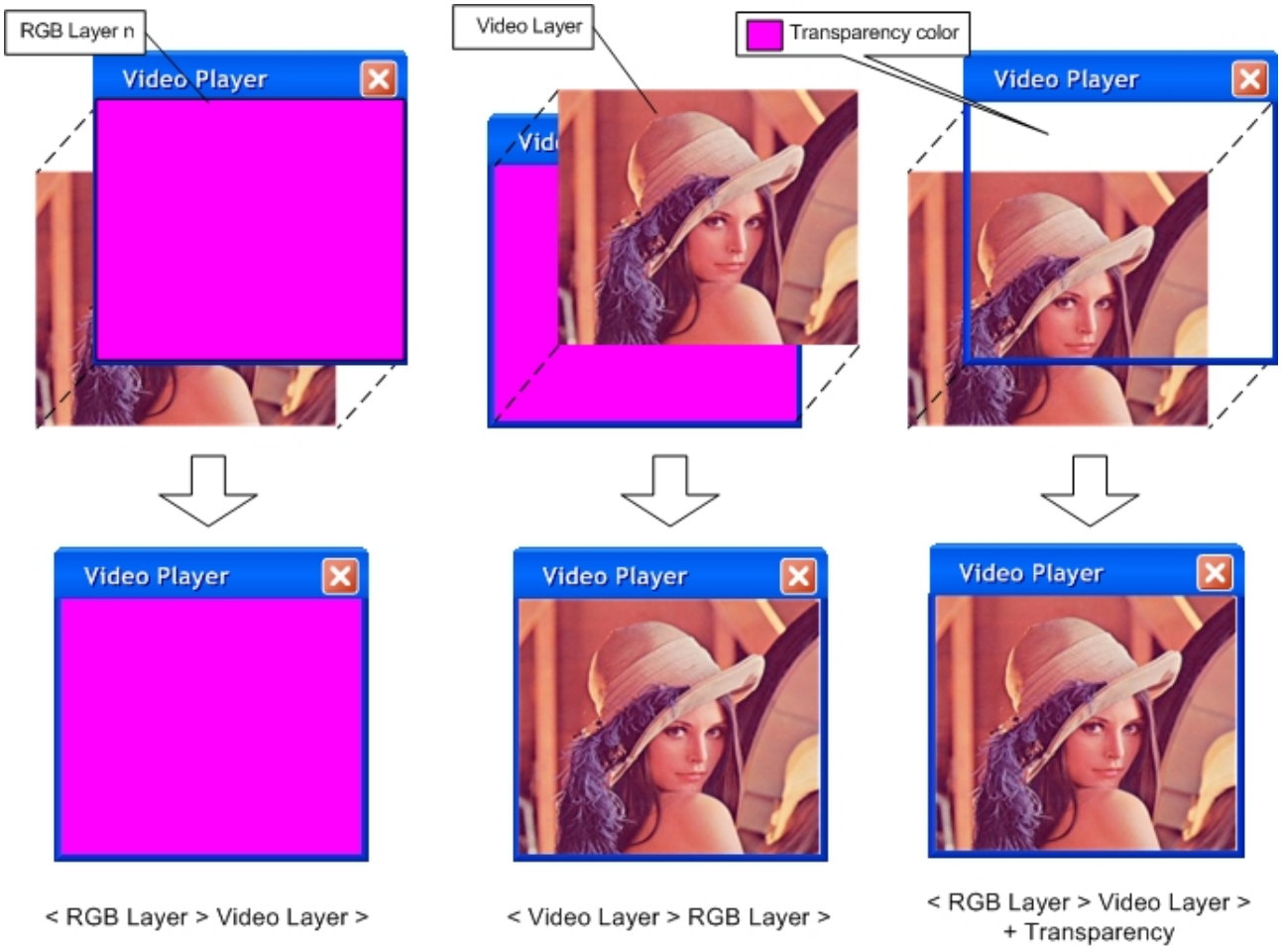


Figure 20-4. Layer priority

20.3.3. Field Mode

The MLC of the **POLLUX** supports an interlace display as well as a progressive display. All registers of the MLC should be set in frame units. For the progressive display, the ***FIELDENB*** bit of the Top controller should be set as '0'. For the interlace display, the ***FIELDENB*** bit of the Top controller should be set as '1'. For example, a 720 x 480 progressive display and a 720 x 480 interlace display have the same settings, except for the setting of the ***FIELDENB*** bit.

20.3.4. Background color

Each layer of the MLC can be positioned at any place on the screen. Therefore, it is possible for there to be an area not contained in any of the layers actually on the screen. The default color displayed in this area is called the background color and the background color is set to ***BGCOLOR***. If all layers are disabled, only the background color is displayed on the screen.
The bpp of the ***BGCOLOR*** is 24 and has the following format:

Bit	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Background color	R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4	G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0

Table 20-3. Background color format

20.4. Per-layer parameters

Function	Symbol	Bit width	Register	Brief description
Enable	LAYERENB	1	MLCCONTROLn[5]	Specifies whether or not to enable this layer.
Dual register set	DIRTYFLAG	1	MLCCONTROLn[4]	Dirty flag for this layer
Lock control	LOCKSIZE[1)]	2	MLCCONTROLn[13:12]	Specifies lock size for memory access.
Position	LEFT	12	MLCLEFTRIGHTn[27:16]	Specifies x-coordinate of upper-left corner.
	TOP	12	MLCTOPBOTTOMn [27:16]	Specifies y-coordinate of upper-left corner.
	RIGHT	12	MLCLEFTRIGHTn [11:0]	Specifies x-coordinate of lower-right corner.
	BOTTOM	12	MLCTOPBOTTOMn[11:0]	Specifies y-coordinate of lower-right corner.
InValid Position0	INVLIDENB[1)]	1	MLCLEFTRIGHTn_0[28]	InValid0 Area Enable
	LEFT[1)]	12	MLCLEFTRIGHTn_0[26:16]	Specifies x-coordinate of upper-left corner.
	TOP[1)]	12	MLCTOPBOTTOMn_0[26:16]	Specifies y-coordinate of upper-left corner.
	RIGHT[1)]	12	MLCLEFTRIGHTn_0[10:0]	Specifies x-coordinate of lower-right corner.
	BOTTOM[1)]	12	MLCTOPBOTTOMn_0[10:0]	Specifies y-coordinate of lower-right corner.
InValid Position1	INVLIDENB[1)]	1	MLCLEFTRIGHTn_1[28]	InValid1 Area Enable
	LEFT[1)]	12	MLCLEFTRIGHTn_1[26:16]	Specifies x-coordinate of upper-left corner.
	TOP[1)]	12	MLCTOPBOTTOMn_1[26:16]	Specifies y-coordinate of upper-left corner.
	RIGHT[1)]	12	MLCLEFTRIGHTn_1[10:0]	Specifies x-coordinate of lower-right corner.
	BOTTOM[1)]	12	MLCTOPBOTTOMn_1[10:0]	Specifies y-coordinate of lower-right corner.
Alpha blending	BLENDENB	1	MLCCONTROLn[2]	Specifies whether or not to enable alpha blending.
	ALPHA	4	MLCTPCOLORn[31:28]	Specifies alpha blending factor.
Color inversion	INVENB[1)]	1	MLCCONTROLn[1]	Specifies whether or not to enable color inversion.
	INVCOLOR[1)]	24	MLCINVCOLOR[23:0]	Specifies the color to be used for color inversion.
Transparency	TPENB[1)]	1	MLCCONTROLn[0]	Specifies whether or not to enable transparency.
	TPCOLOR[1)]	24	MLCTPCOLORn[23:0]	Specifies the color to be used as transparency color.
Address generation	ADDRESS	32	MLCADDRESSn[31:0]	Specifies the base address of image buffer.
	HSTRIDE[1)]	32	MLCHSTRIDEn[31:0]	Specifies the horizontal stride in bytes.
	VSTRIDE	32	MLCVSTRIDEn[31:0]	Specifies the vertical stride in bytes.
	ADDRESSCB[2)]	32	MLCADDRESSCB[31:0]	Specifies the base address of Cb image buffer
	ADDRESSCR[2)]	32	MLCADDRESSCR[31:0]	Specifies the base address of Cr image buffer.
	VSTRIDECB[2)]	32	MLCVSTRIDECB[31:0]	Specifies the vertical stride in bytes for Cb image buffer.
	VSTRIDECR[2)]	32	MLCVSTRIDECR[31:0]	Specifies the vertical stride in bytes for Cr image buffer.
Note 1) Only exists in RGB layers Note 2) Only exists in Video layer				

Table 20-4. RGB layer registers

20.4.1. Enable

Each layer has a LAYERENB bit to enable/disable each layer of the **POLLUX** at a certain point. If the ***LAYERENB*** bit is set as '1', the relevant layer becomes On. If the ***LAYERENB*** bit is set as '0', the relevant layer becomes Off and the other layer setting registers are not used. Since the setting of the ***LAYERENB*** bit is reflected by a dirty flag, the layer is toggled On/Off in accordance with a VSYNC signal, even if the user controls the ***LAYERENB*** bit at a certain point.

20.4.2. Lock control

Each RGB layer can adjust the data size to be read at any one time when a memory read is requested through the Lock control. The ***LOCKSIZE*** can specify 4, 8 or 16 and the unit size is 8 bytes. Therefore, if the ***LOCKSIZE*** is 4, the data size to be read at any one time is 32 bytes. For a resolution of 1280 x 1024 or higher, it is recommended to set the ***LOCKSIZE*** as 16.

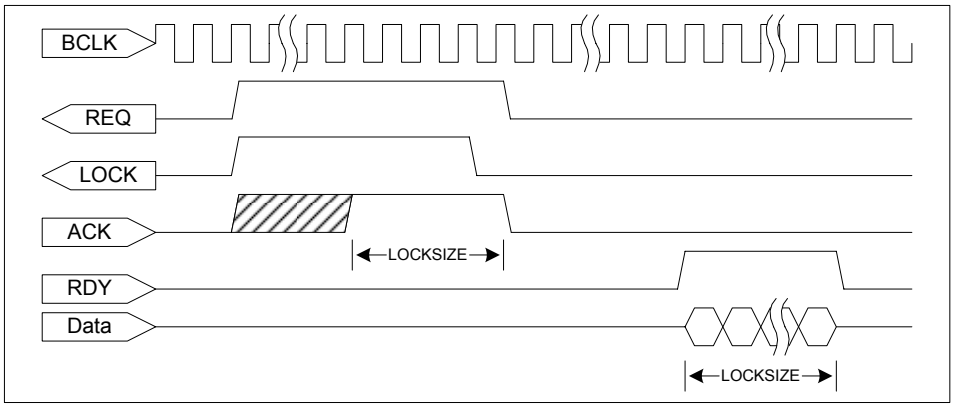


Figure 20-5. Lock timing

20.4.3. Position

The Position function enables users to specify the top-left (***LEFT*** and ***TOP***) and the bottom-right (***RIGHT*** and ***BOTTOM***) coordinates. Each coordinate can be positioned at any point within the range from -2048 to 2047, but only the layer contained in the area from (0, 0) and (ScreenWidth – 1, ScreenHeight -1) is displayed on the actual screen. The MLC of the **POLLUX** supports H/W clipping for any area outside of the screen area, so users do not need to carry out additional clipping processing. In addition, the MLC does not read the data in the clipped area and the area hidden by upper layer from memory for effective use of the memory bandwidth.
RGB Layer could appoint two invisible areas without using certain color. Appointed area is not read from Memory.

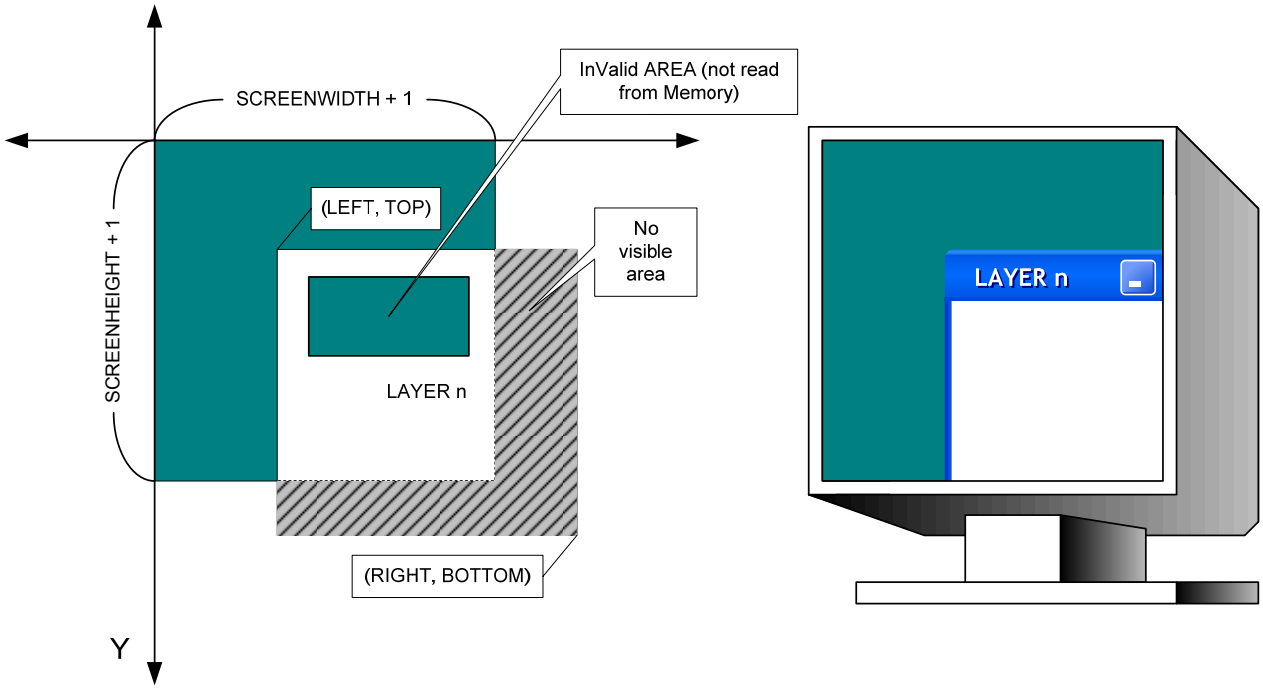


Figure 20-6. Layer position

20.4.4. Pixel format

20.4.4.1. RGB layer format

Each RGB layer supports various formats and the formats are listed in Table 20-5.

R : Red, G : Green, B : Blue, A : Alpha, X : Not used

Pixel format	FORMAT[15:0]	Bpp	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R5G6B5	4432h	16	R1[4:0]					G1[5:0]						B1[4:0]					R0[4:0]					G0[5:0]						B0[4:0]			
B5G6R5	C432h	16	B1[4:0]					G1[5:0]						R1[4:0]					B0[4:0]					G0[5:0]						R0[4:0]			
X1R5G5B5	4342h	16		R1[4:0]				G1[4:0]						B1[4:0]						R0[4:0]				G0[4:0]						B0[4:0]			
X1B5G5R5	C342h	16		B1[4:0]				G1[4:0]						R1[4:0]						B0[4:0]				G0[4:0]						R0[4:0]			
X4R4G4B4	4211h	16				R1[3:0]				G1[3:0]				B1[3:0]								R0[3:0]				G0[3:0]				B0[3:0]			
X4B4G4R4	C211h	16				B1[3:0]				G1[3:0]				R1[3:0]								B0[3:0]				G0[3:0]				R0[3:0]			
X8R3G3B2	4120h	16								R1[2:0]			G1[2:0]			B1[1:0]										R0[2:0]			G0[2:0]			B0[1:0]	
X8B3G3R2	C120h	16								B1[2:0]			G1[2:0]			R1[1:0]										B0[2:0]			G0[2:0]			R0[1:0]	
A1R5G5B5	3342h	16	A1	R1[4:0]				G1[4:0]						B1[4:0]					A0	R0[4:0]				G0[4:0]						B0[4:0]			
A1B5G5R5	B342h	16	A1	B1[4:0]				G1[4:0]						R1[4:0]					A0	B0[4:0]				G0[4:0]						R0[4:0]			
A4R4G4B4	2211h	16	A1[3:0]			R1[3:0]				G1[3:0]				B1[3:0]					A0[3:0]			R0[3:0]				G0[3:0]				B0[3:0]			
A4B4G4R4	A211h	16	A1[3:0]			B1[3:0]				G1[3:0]				R1[3:0]					A0[3:0]			B0[3:0]				G0[3:0]				R0[3:0]			
A8R3G3B2	1120h	16	A1[7:0]							R1[2:0]			G1[2:0]			B1[1:0]			A0[7:0]							R0[2:0]			G0[2:0]			B0[1:0]	
A8B3G3R2	9120h	16	A1[7:0]							B1[2:0]			G1[2:0]			R1[1:0]			A0[7:0]							B0[2:0]			G0[2:0]			R0[1:0]	
R8G8B8	4653h [1]	24	B1[7:0]							R0[7:0]									G0[7:0]							B0[7:0]								
B8G8R8	C653h [1]	24	R1[7:0]							B0[7:0]									G0[7:0]							R0[7:0]								
X8R8G8B8	4653h [1]	32								R[7:0]									G[7:0]							B[7:0]								
X8B8G8R8	C653h [1]	32								B[7:0]									G[7:0]							R[7:0]								
A8R8G8B8	0653h	32	A[7:0]							R[7:0]									G[7:0]							B[7:0]								
A8B8G8R8	8653h	32	A[7:0]							B[7:0]									G[7:0]							R[7:0]								
PTR5G6B5	443Ah	8	Palette Table Address3[7:0]							Palette Table Address2[7:0]									Palette Table Address1[7:0]							Palette Table Address0[7:0]								

Note 1) The format settings for R8G8B8 & X8R8G8B8 and B8G8R8 & X8B8G8R8 are the same. However, the ***HSTRIDE***s for R8G8B8 and B8G8R8 should be set as '3' because they are in 24 bpp modes, while the ***HSTRIDE***s for X8R8G8B8 and X8B8G8R8 should be set as '4'.

Table 20-5. RGB layer format

Every RGB layer does support the index format using a palette. The above formats are converted into A4R8G8B8 and are managed in each RGB layer. Each color component is converted into 8-bit size. The color components with the size smaller than 8-bit are extended to 8-bit size from the highest bit repeatedly. For example, the color with 5-bit size is converted to {[4:0], [4:2]} and the color with 3-bit size is converted to {[2:0], [2:0], [2:1]}. Each layer compares the color component internally extended with *TPCOLOR* or *INVCOLOR*. Therefore, a user should set the color that each color format is extended in R8G8B8 in *TPCOLO**R*** and ***I**NVCOLOR.*

User could assign whether to use Palette Table with FORMAT[3]. Each RGB Layer has Palette Table which is 16bits, 256Depth Size. And, it is set like HSTRIDE = '1' when using.

20.4.4.2. Video layer format

The Video layer manages YUV data and supports the linear YUV format and the 2D block addressing separated YUV format.

FORMAT[1:0]	Pack mode	Type	Y:UV	Addressing mode
0	Separate	YUV	4:2:0	2D Block

Table 20-6. Video layer format

The MLC of the **POLLUX** uses the A4R8G8B8 format internally. Therefore, the Video layer also reads YUV data from the memory and converts into RGB data internally. The formulas with which the Video layer converts YUV data into RGB data are as follows:

※ The formula for YCbCr to RGB conversion
- R = Y + (1.4020 * Cr)
- G = Y - (0.34414 * Cb) + (0.71414 * Cr)
- B = Y + (1.7720 * Cb)

▪ 2D block addressing separated YUV format

In the 2D block addressing separated YUV format, each of Y, U and V exists at separate memory spaces. In addition, the format is divided into 420 in proportion to U and V for Y. The 2D block addressing separated YUV format is the 2D block addressing format and each component has a size of 64 x 32 and linearity in block units. These features provide the **POLLUX**'s unique memory format, to enhance the effectiveness of memory access when the **POLLUX** manages data in macro block units through an algorithm to compress/decompress images like MPEG files.

According to each format, Y, U and V correspond to 2 x 2 pixels as follows:

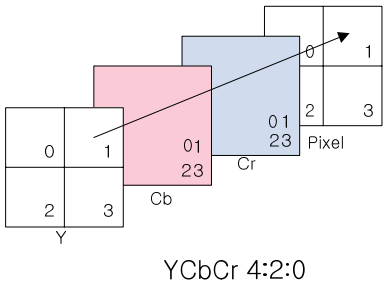


Figure 20-7. Separated YUV format

20.4.5. Layer blending

MLC consists of two RGB layers and one Video layer. Each RGB layer supports Transparency, Color Inversion and Alpha Blending functions but the Video layer only supports Alpha Blending functions. The Color Inversion and the Alpha Blending functions are only applied to between layers and not to background.

The Transparency function enables users to specify a particular color and handle the color as a transparent color. Therefore, an area filled with a transparent color shows through the lower layer and shows the layer as it is. Like the Transparency function, the Color Inversion function shows through the lower layer and shows the layer as it is, but the function is different in that it inverts and projects the layer's color. The Transparency and Color Inversion functions are useful for the implementation of the Cursor layer as shown in the figure below:

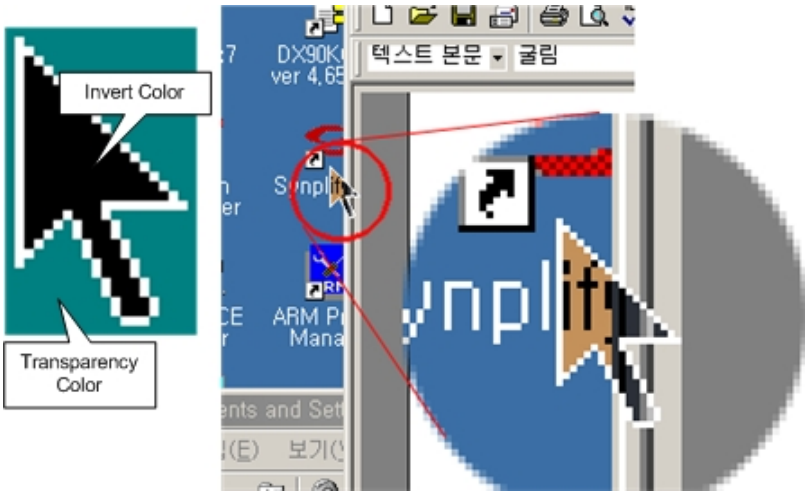


Figure 20-8. Transparency & Color inversion

The Transparency and the Color Inversion effects are applied by setting each of the ***TPENB*** and the ***INVENB*** bits of an RGB Layer as '1'. Both ***TPCOLOR*** and ***INVCOLOR*** have R8G8B8 formats.

Bit	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPCOLOR / INVCOLOR	R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4	G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0

Table 20-7. TPCOLOR & INVCOLOR format

The Alpha Blending function enables users to adjust the transparency of a desired layer. The Alpha level is adjusted by ***ALPHA***[3:0] and can be specified in the range between 0 and 15. If the Alpha level is 15, it means it is fully opaque. If the level is 0, it indicates full transparency. In addition, each RGB layer can use a pixel format including Alpha and is applied in Alpha per pixel units. In the Per-layer alpha or the Per-pixel Alpha formats, the Alpha Blending effect can only be applied by setting the ***BLENDENB*** bit as '1'.

Only one layer can use the alpha blending function at the same time, and when the alpha blending function of one layer is enabled, the alpha blending fucntion of the other layers must be disabled.

All layers use the A4R8G8B8 format internally and the formula for Alpha Blending is as follows:

※ The formula for Alpha blending function

If alpha is 0 then α is 0, else α is alpha + 1

Result color = this layer color * α / 16 + lower layer color * (16 – α) / 16

Figure 20-9. Alpha blending

20.4.6. Address generation

20.4.6.1. RGB layer address generation

The Address generation function enables users to specify the address and stride of the memory where images are stored.
Stride is divided into horizontal stride and vertical stride. The stride is the unit for increasing an address. The horizontal stride (***HSTRIDE***) is the value to be added to an address whenever its x-coordinate increases, while the vertical stride (***VSTRIDE***) is the value to be added to the address whenever its y-coordinate increases. In general, the horizontal stride is the number of the bytes per pixel and the vertical stride is the value that is the number of bytes per pixel multiplied by an image width. The vertical stride has 2's complement format. Thus a vertical flip function can be implemented by specifying the vertical stride as negative number.
The Image address (***ADDRESS***) usually specifies an address on the top left corner of the image. If the vertical stride for the vertical flip function has a negative number, the ***ADDRESS*** should specify an address on the bottom left corner of the image.

Vertical flip	Vertical stride	Base address
Off	> 0	Address on the top left corner of the image
On	< 0	Address on the bottom left corner of the image

Table 20-8. RGB layer address & flip

20.4.6.2. Video layer address generation

In the Video layer, the horizontal stride is not separately specified and is fixed internally. In addition, the vertical stride should always be a positive number. Since the vertical stride is always positive number, the vertical flip function is not supported.

- 2D block addressing separated YUV format

In the separated format, each Y, U and V is stored in different addresses of the memory. Thus the address for the Y component is specified by the ***ADDRESS3*** and the ***VSTRIDE3*** registers and the address for the U (Cb) and V (Cr) components is separately specified by the ***ADDRESSCB*** & the ***VSTRIDECB*** registers and the ***ADDRESSCR*** & the ***VSTRIDECR*** registers. Since the **POLLUX** uses segments with 4096 x 4096 pixel sizes in the 2D block addressing format, the vertical stride should be specified as 4096. In addition, the ***ADDRESS3/CB/CR*** registers set the separate format for segment addresses and not for normal linear addresses. The segment addressing format is listed in Table 20-9.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Address	0	0	1	0	Index of Segment				Y coordinate in segment [11:0]												X coordinate in segment [11:0]											

Table 20-9. Segment addressing format

The display array area on the memory map should be specified as a setting Address [31:28] of 4'b0010. In addition, the Memory controller should be enabled to use the display array area. See section '5.3.2. Display Array Area' for more detail.

20.5. Video layer specific parameters

The Video layer provides the Scale and Color Control functions additionally.

Function	Symbol	Bit width	Register	Brief description
Scale	VFILTERENB	1	MLCVSCALE[28]	Specifies whether or not to vertical scale enable bilinear filter.
	HFILTERENB	1	MLCHSCALE[28]	Specifies whether or not to horizontal scale enable bilinear filter.
	HSCALE	23	MLCHSCALE[22:0]	Specifies the horizontal scale ratio.
	VSCALE	23	MLCVSCALE[22:0]	Specifies the vertical scale ratio.
Color control	BRIGHTNESS	8	MLCLUENH[15:8]	Specifies the brightness value.
	CONTRAST	3	MLCLUENH[2:0]	Specifies the contrast value.
	HUECBnA	8	MLCCHENHn[7:0]	Specifies the cosine value for Cb component.
	HUECBnB	8	MLCCHENHn[15:8]	Specifies the sine value for Cb component.
	HUECRnA	8	MLCCHENHn[23:16]	Specifies the sine value for Cr component.
	HUECRnB	8	MLCCHENHn[31:24]	Specifies the cosine value for Cr component.

Table 20-10. Video layer specific parameters

20.5.1. Scale function

The scale-up and the scale-down of the Video layer are determined by the ***HSCALE*** and ***VSCALE*** parameters. Each parameter finds and sets the ratio between an input image size and an output image size. The setting formulae for the ***HSCALE*** and ***VSCALE*** parameters are as follows:

※ The formula for HSCALE and VSCALE

In case of the enlargement by using a Bilinear filter:

- HSCALE = (source width-1) * (1<<11) / (destination width-1)
- VSCALE = (source height-1) * (1<<11) / (destination height-1)

,else

- HSCALE = source width * (1<<11) / destination width
- VSCALE = source height * (1<<11) / destination height

Video Layer supports bilinear filtered scaling up and down. Nearest neighbor scaling is also supported. When bilinear scaling up and down is used, ***H/VFILTERENB*** should be set to '1' for bilinear filtered scaling. When nearest neighbor scaling down is used, ***H/VFILTERENB*** should be set to '0'. Nearest neighbor scaling up is not supported. So, ***VFILTERENB*** should be set to '1' when scaling up. The difference in images produced by the Bilinear filter method is as shown in Figure 20-10.

Figure 20-10. Bilinear Filter

20.5.2. Color control

The Video layer supports the Color Control function for output images. A user can adjust Brightness, Contrast, Hue and Saturation to compensate video data colors.

20.5.2.1. Luminance Enhancement

The Video layer can compensate luminance data by adjusting Brightness and Contrast.
The Brightness consists of 256 levels and is set by the ***BRIGHTNESS*** parameter. The ***BRIGHTNESS*** parameter has a 2's complement value and can be set between the -128 level and the +127 level. Figure 20-11 shows the image change depending on the Brightness change.

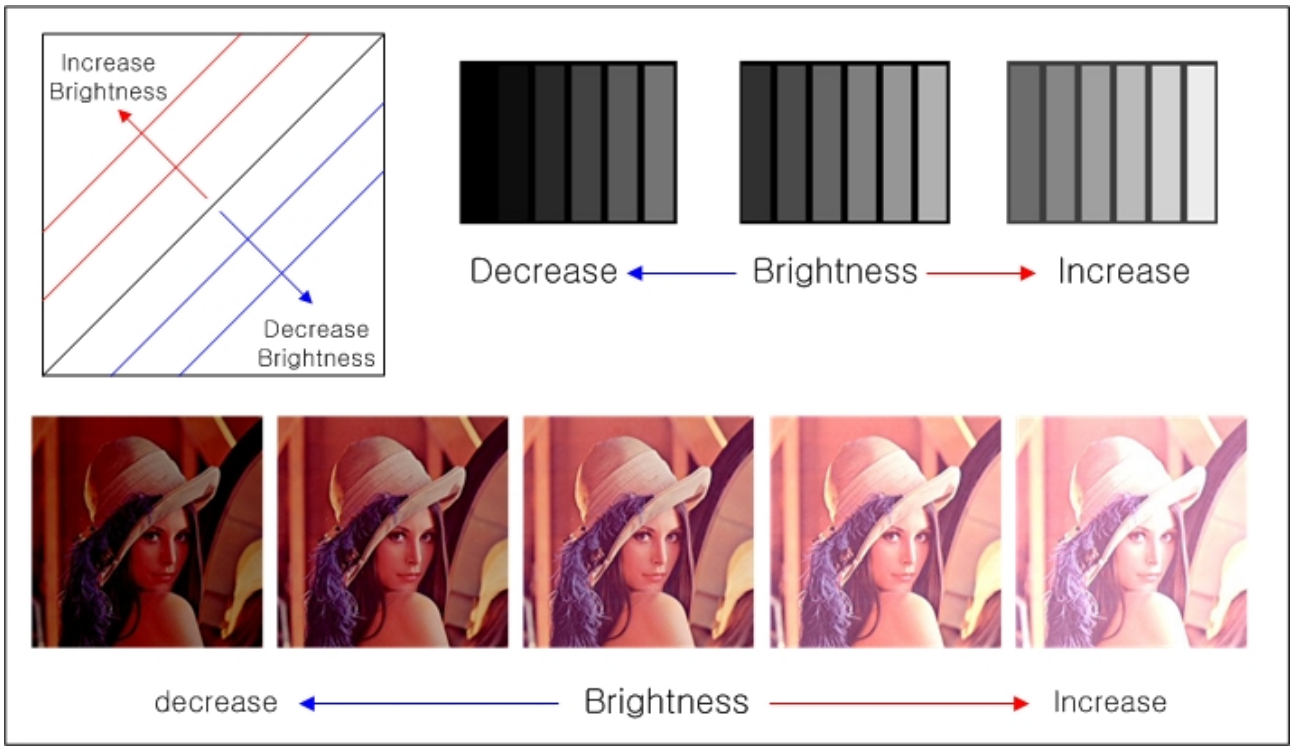


Figure 20-11. Brightness

The Contrast consists of 8 levels and is set by the ***CONTRAST*** parameter. The Video layer can adjust the contrast from 1.0 to 1.875 in increments of 0.125, but cannot reduce the contrast of the original image. Table 20-11 lists the contrast values corresponding to the ***CONTRAST*** parameters.

CONTRAST	0	1	2	3	4	5	6	7
Contrast value	1.0	1.125	1.25	1.375	1.5	1.625	1.75	1.875

Table 20-11. CONTRAST parameter

Figure 20-12 shows the image change depending on the contrast change. The following figure is originally intended to explain better about the contrast function as well as to show the effect of contrast reduction. Actually, the Video layer can increase the contrast, but not reduce it.

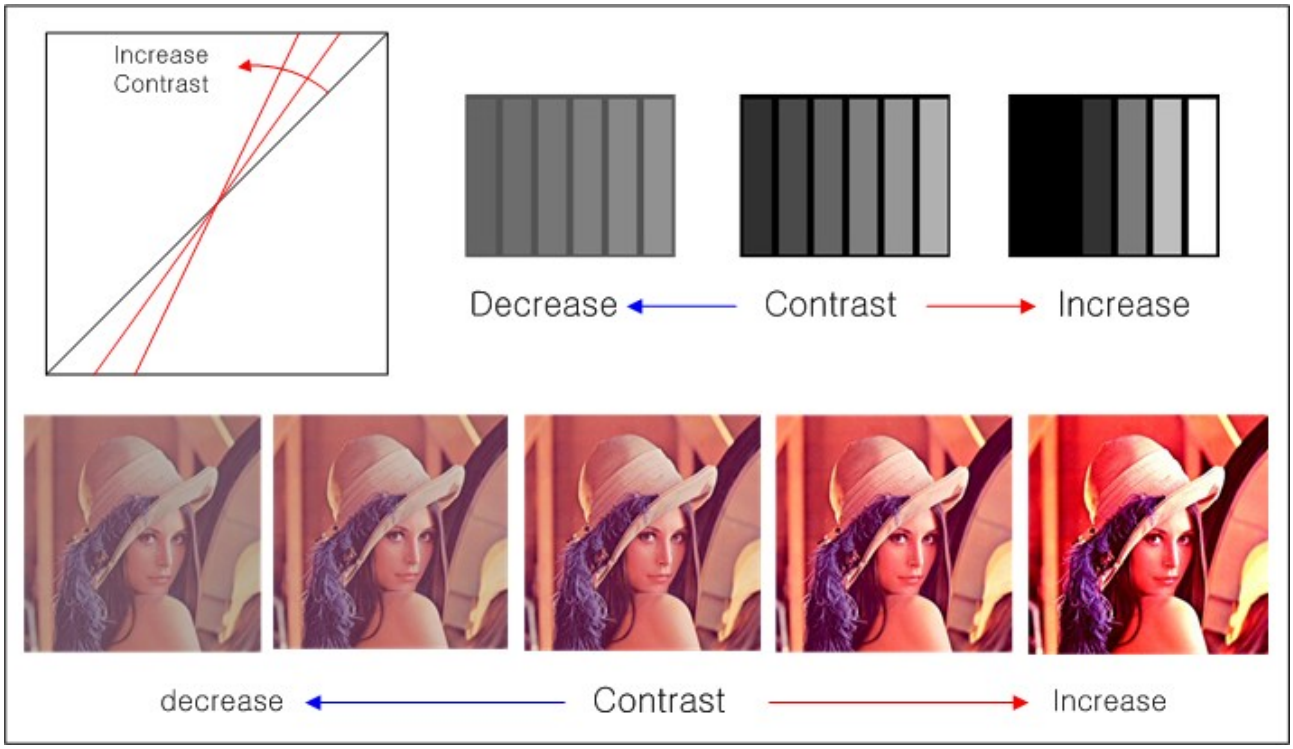


Figure 20-12. Contrast

20.5.2.2. Chrominance Enhancement

The Video layer can compensate Chrominance data by adjusting Hue and Saturation.

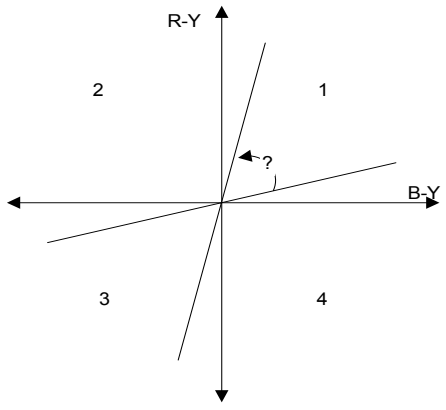
The Hue is adjusted by the following formulae:

$$(B\text{-}Y)' = (B\text{-}Y) * \cos(\theta) - (R\text{-}Y) * \sin(\theta)$$

$$(R\text{-}Y)' = (B\text{-}Y) * \sin(\theta) + (R\text{-}Y) * \cos(\theta)$$

The Saturation can be adjusted by multiplying a gain value by the above result value.

The Video layer can adjust the Hue and Saturation differently in each quadrant and has ***HUECBnA/B*** and ***HUECRnA/B*** parameters for each quadrant. Each parameter has the [S.1.6] format and is applied as follows:



1) 1st quadrant : (B-Y) > 0 and (R-Y) > 0
   (B-Y)' = (B-Y) * HUECB1A + (R-Y) * HUECB1B
   (R-Y)' = (B-Y) * HUECR1A + (R-Y) * HUECR1B
2) 2nd quadrant : (B-Y) < 0 and (R-Y) > 0
   (B-Y)' = (B-Y) * HUECB2A + (R-Y) * HUECB2B
   (R-Y)' = (B-Y) * HUECR2A + (R-Y) * HUECR2B
3) 3rd quadrant : (B-Y) < 0 and (R-Y) < 0
   (B-Y)' = (B-Y) * HUECB3A + (R-Y) * HUECB3B
   (R-Y)' = (B-Y) * HUECR3A + (R-Y) * HUECR3B
4) 4th quadrant : (B-Y) < 0 and (R-Y) < 0
   (B-Y)' = (B-Y) * HUECB4A + (R-Y) * HUECB4B
   (R-Y)' = (B-Y) * HUECR4A + (R-Y) * HUECR4B

Figure 20-13. The basic concept of Hue and Saturation control

Therefore, each parameter can be calculated by using the following formulas:

※ The formula for Hue and Saturation parameters

- HUECBnA = cos(θ) * 64 * gain, HUECBnB = - sin(θ) * 64 * gain
- HUECRnA = sin(θ) * 64 * gain, HUECRnB = cos(θ) * 64 * gain

, where θ is for hue and gain is for saturation from -2 to 1.99999X.

Figure 20-14 shows the image change depending on the changes to Hue and Saturation.

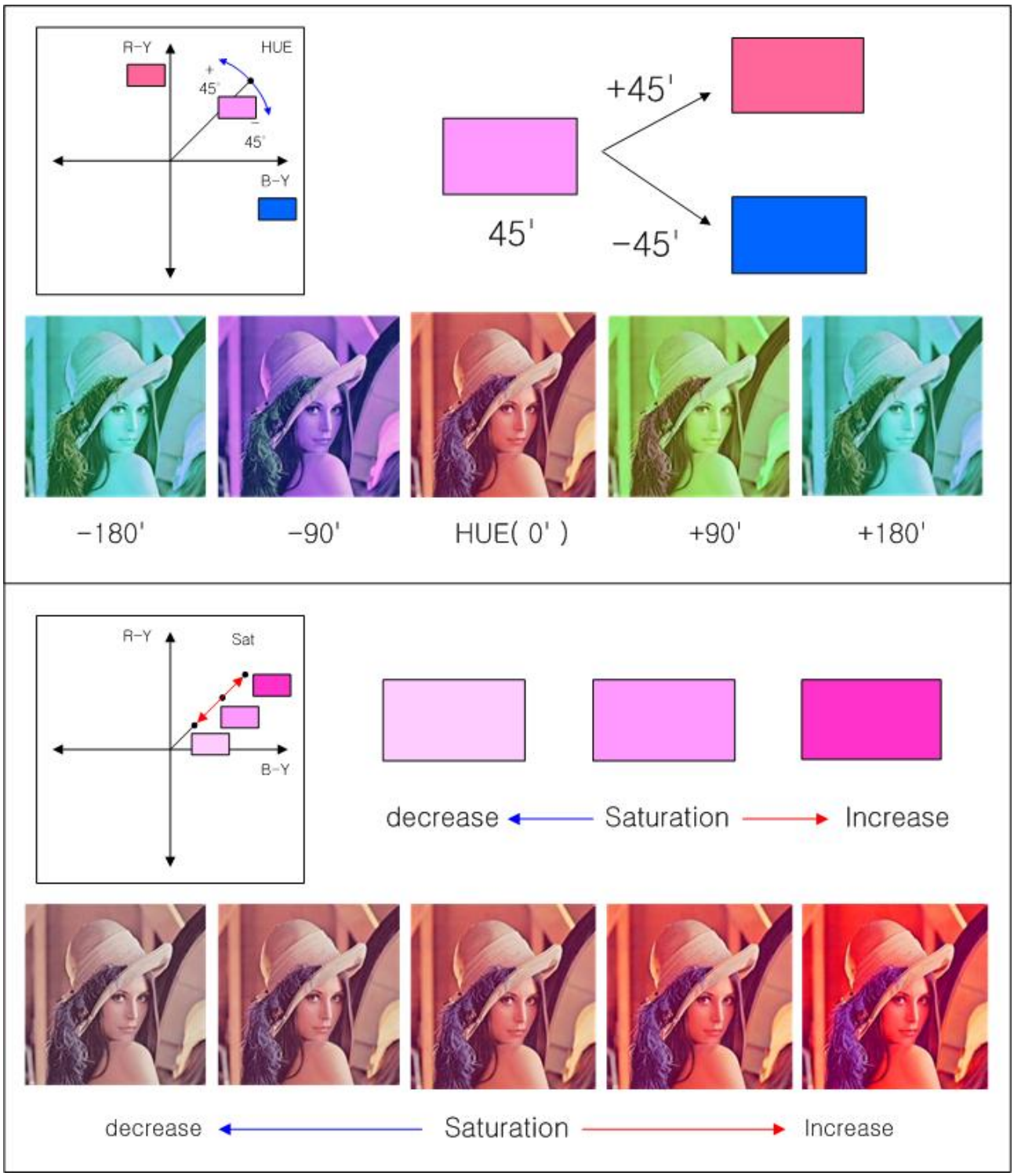


Figure 20-14. Hue and Saturation

20.6. Clock Generation

The MLC operates by using the PCLK and the BCLK. The PCLK is used when the CPU accesses the registers of the MLC. The BCLK is used as an internal clock or when the MLC accesses the memory. The MLC provides various operation modes for the PCLK and the BCLK. Therefore, users can adjust the clock for the MLC by setting the ***PCLKMODE*** and the ***BCLKMODE*** parameters according to their purpose.

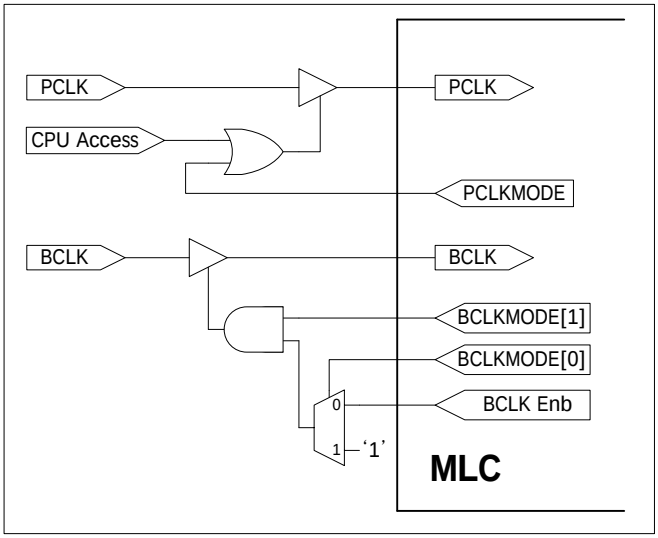


Figure 20-15. Clock generation

PCLKMODE	Brief Description
0	Only enabled when CPU accesses MLC Register.
1	Always enabled.

Table 20-12. PCLK mode

BLCKMODE		Brief Description
[1]	[0]	
0	0	Always disabled.
0	1	Reserved for future use.
1	0	Dynamic controlled by MLC
1	1	Always enabled.

Table 20-13. BCLK mode

20.7. Register Sumarry

Bit	R/W	Symbol	Description	Reset Value
MLC TOP CONTROL REGISTER (MLCCONTROLT) ***Address : C000_4000h(Primary) / C000_4400h(Secondary)***				
[31:14]	R	**RESERVED**	Reserved	18'b0
[13:12]	R/W	**RESERVED**	Reserved	2'b00
[11]	R/W	**PIXELBUFFER_PWD**	MLC Pixel Buffer Power On/Off it should be 'On' before MLC Enabled. 0 : Power Off 1 : Power On	1'b0
[10]	R/W	**PIXELBUFFER_SLD**	MLC Pixel Buffer Sleep Mode it is usable only when Power On. 0 : Sleep Mode Enable 1 : Sleep Mode Disable	1'b0
[9:8]	R/W	**PRIORITY**	Specifies the priority of the Video layer. 00 : video layer > layer0 > layer1 01 : layer0 > video layer > layer1 10 : layer0 > layer1 > video layer Others : Reserved for future use	2'b1
[7:4]	R	**RESERVED**	Reserved	4'bx
[3]	R/W	**DITTYFLAG**	Dirty Flag for MLC top controller. If this bit is set as '1', the register settings concerned with the Top controller in vertical sync mode are updated and this bit is cleared as '0' automatically. Read) 0 : Clean 1 : Dirty Write) 0 : No affect 1 : Apply modified settings	1'b0
[2]	R	**RESERVED**	Reserved	1'b0
[1]	R/W	**MLCENB**	Specifies whether or not to enable MLC Set when MLCENB Set/Clear together with set **MLCCONTROLT** DITTYFLAG. 0 : Disable 1 : Enable	1'b0
[0]	R/W	**FIELDENB**	Specifies whether or not to enable Interlace mode. 0 : progressive mode 1 : Interlace mode	1'b0
MLC SCREEN SIZE REGISTER (MLCSCREENSIZE) ***Address : C000_4004h / C000_4404h***				
[31:28]	R	**RESERVED**	Reserved	4'b0
[27:16]	R/W	**SCREENHEIGHT**	Specifies 'the whole screen height – 1'.	12'b0
[15:12]	R	**RESERVED**	Reserved	4'b0
[11:0]	R/W	**SCREENWIDTH**	Specifies 'the whole screen width – 1'.	12'b0
MLC BACKGROUND COLOR REGISTER (MLCBGCOLOR) ***Address : C000_4008h / C000_4408h***				
[31:24]	R	**RESERVED**	Reserved	8'b0
[23:0]	R/W	**DEFAULTCOLOR**	Specifies the color to be displayed on the screen in areas not covered by any of the layers Specifies the R8G8B8 format in 24 bpp mode.	24'b0
MLC RGB LAYER 0 LEFT RIGHT REGISTER (MLCLEFTRIGHT0) ***Address : C000_400Ch / C000_440Ch***				
[31:28]	R	**RESERVED**	Reserved	4'b0
[27:16]	R/W	**LEFT**	Specifies the x-coordinate of the upper-left corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
[15:12]	R	**RESERVED**	Reserved	4'b0
[11:0]	R/W	**RIGHT**	Specifies the x-coordinate of the lower-right corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
MLC RGB LAYER 0 TOP BOTTOM REGISTER (MLCTOPBOTTOM0) ***Address : C000_4010h / C000_4410h***				
[31:28]	R	**RESERVED**	Reserved	4'b0
[27:16]	R/W	**TOP**	Specifies the y-coordinate of the upper-left corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
[15:12]	R	**RESERVED**	Reserved	4'b0

Bit	R/W	Symbol	Description	Reset Value
[11:0]	R/W	**BOTTOM**	Specifies the y-coordinate of the lower-right corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
MLC RGB LAYER 0 INVALID AREA 0 LEFT RIGHT REGISTER (MLCLEFTRIGHT0_0) ***Address : C000_4014h / C000_4414h***				
[31:29]	R	**RESERVED**	Reserved	3'b0
[28]	R/W	**INVALIDENB0**	Shows the status of disable/enable about 1st invisible area of RGB Layer0. 0 : Disable 1 : Enable	1'b0
[27]	R	**RESERVED**	Reserved	1'b0
[26:16]	R/W	**LEFT**	Notify left coordination of X axis from the area to be invisible of RGB Layer0. Able to set coordination from 0 to 2047.	11'b0
[15:11]	R	**RESERVED**	Reserved	5'b0
[10:0]	R/W	**RIGHT**	Notify right coordination of X axis from the area to be invisible of RGB Layer0. Able to set coordination from 0 to 2047.	11'b0
MLC RGB LAYER 0 INVALID AREA 0 BOTTOM TOP REGISTER (MLCTOPBOTTOM0_0) ***Address : C000_4018h / C000_4218h***				
[31:27]	R	**RESERVED**	Reserved	5'b0
[26:16]	R/W	**TOP**	Notify upper coordination of Y axis from the area to be invisible of RGB Layer0. Able to set coordination from 0 to 2047.	11'b0
[15:11]	R	**RESERVED**	Reserved	5'b0
[10:0]	R/W	**BOTTOM**	Notify lower coordination of Y axis from the area to be invisible of RGB Layer0. Able to set coordination from 0 to 2047.	11'b0
MLC RGB LAYER 0 INVALID AREA 1 LEFT RIGHT REGISTER (MLCLEFTRIGHT0_1) ***Address : C000_401Ch / C000_441Ch***				
[31:29]	R	**RESERVED**	Reserved	3'b0
[28]	R/W	**INVALIDENB1**	Shows the status of disable/enable about 2nd invisible area of RGB Layer0. 0 : Disable 1 : Enable	1'b0
[27]	R	**RESERVED**	Reserved	1'b0
[26:16]	R/W	**LEFT**	Notify left coordination of X axis from the area to be invisible of RGB Layer0. Able to set coordination from 0 to 2047.	11'b0
[15:11]	R	**RESERVED**	Reserved	5'b0
[10:0]	R/W	**RIGHT**	Notify right coordination of X axis from the area to be invisible of RGB Layer0. Able to set coordination from 0 to 2047.	11'b0
MLC RGB LAYER 0 INVALID AREA 1 BOTTOM TOP REGISTER (MLCTOPBOTTOM0_1) ***Address : C000_4020h / C000_4420h***				
[31:27]	R	**RESERVED**	Reserved	5'b0
[26:16]	R/W	**TOP**	Notify upper coordination of Y axis from the area to be invisible of RGB Layer0. Able to set coordination from 0 to 2047.	11'b0
[15:11]	R	**RESERVED**	Reserved	5'b0
[10:0]	R/W	**BOTTOM**	Notify lower coordination of Y axis from the area to be invisible of RGB Layer0. Able to set coordination from 0 to 2047.	11'b0

Bit	R/W	Symbol	Description	Reset Value
MLC RGB LAYER 0 CONTROL REGISTER (MLCCONTROL0) ***Address : C000_4024h / C000_4424h***				
[31:16]	R/W	**FORMAT**	Specifies the RGB data format. For detailed information, refer to Table 21-5.	16'b0
[15]	R/W	**PALETTEPWD**	RGB Layer0 Palette Table's Power On/Off it should be 'On' before RGB Layer0 Enabled 0 : Power Off 1 : Power On	1'b0
[14]	R/W	**PALETTESLD**	RGB Layer0 Palette Table's Sleep Mode. It is usable only when **PALETTEPWD = '1'** 0 : Sleep Mode Enable 1 : Sleep Mode Disable	1'b0
[13:12]	R/W	**LOCKSIZE**	Adjusts the data size to be read at any one time when a memory read for RGB data is requested. For a resolution of 1280 x 1024, it is recommended to set it as 16. 0 : 4 1 : 8 2 : 16 3 : reserved	2'b0
[11:9]	R	**RESERVED**	Reserved	3'b0
[8]	R/W	**RESERVED**	Reserved but it should be written '0'	1'b0
[7:6]	R	**RESERVED**	Reserved	2'b0
[5]	R/W	**LAYERENB**	Enables/disables this layer. 0 : Disable 1 : Enable	1'b0
[4]	R/W	**DIRTYFLAG**	Dirty flag for this layer. If this bit is set as '1', the register settings concerned with this layer are updated in vertical sync mode and this bit is cleared as '0' automatically. Read) 0 : Clean 1 : Dirty Write) 0 : No affect 1 : Apply modified settings	1'b0
[3]	R	**RESERVED**	Reserved	1'b0
[2]	R/W	**BLENDENB**	Enables/disables the Alpha Blending function in this layer. In addition, this bit should be set as 'enable' to apply Alpha to a format with an Alpha channel. 0 : Disable 1 : Enable	1'b0
[1]	R/W	**INVENB**	Enables/disables the Color Inversion function in this layer. 0 : Disable 1 : Enable	1'b0
[0]	R/W	**TPENB**	Enables/disables the Transparency function in this layer. 0 : Disable 1 : Enable	1'b0
MLC RGB LAYER 0 HORIZONTAL STRIDE REGISTER (MLCHSTRIDE0) ***Address : C000_4028h / C000_4428h***				
[31]	R/W	**RESERVED**	Reserved for future use. You have to write '0' only.	1'b0
[30:0]	R/W	**HSTRIDE**	Specifies the horizontal stride in byte units. This value is the byte offset from the current pixel to the next unit and has the number of bytes per pixel in general.	31'b0
MLC RGB LAYER 0 VERTICAL STRIDE REGISTER (MLCVSTRIDE0) ***Address : C000_402Ch / C000_442Ch***				
[31:0]	R/W	**VSTRIDE**	Specifies the vertical stride in byte units. This value is the byte offset from the current line to the next line and has the number of bytes per line in general. This value has the 2's complement format and can be specified as a negative number for the Vertical Flip function.	32'b0
MLC RGB LAYER 0 TRANSPARENCY COLOR REGISTER (MLCTPCOLOR0) ***Address : C000_4030h / C000_4430h***				
[31:28]	R/W	**ALPHA**	Specifies an Alpha Blending factor. This value is valid only for an RGB format without Alpha channels. The formula for Alpha Blending is as follows: If ALPHA is 0 then a is 0, else a is ALPHA + 1. color = this layer color * a / 16 + lower layer color * (16-a) / 16	4'b0
[27:24]	R	**RESERVED**	Reserved	4'b0
[23:0]	R/W	**TPCOLOR**	Specifies the color for the Transparency. These bits have the R8G8B8 format in 24 bpp mode.	24'b0
MLC RGB LAYER 0 INVERSION COLOR REGISTER (MLCINVCOLOR0) ***Address : C000_4034h / C000_4434h***				
[31:24]	R	**RESERVED**	Reserved	8'b0
[23:0]	R/W	**INVCOLOR**	Specifies the color to be inverted. These bits have the R8G8B8 format in 24 bpp mode.	24'b0

Bit	R/W	Symbol	Description	Reset Value
MLC RGB LAYER 0 BASE ADDERSS REGISTER (MLCADDRESS0) ***Address : C000_4038h / C000_4438h***				
[31:0]	R/W	**ADDRESS**	Specifies the memory address where RGB data is stored. In general, the address on the top left of the image is specified, but the address on the bottom left corner is specified for Vertical Flip.	32'b0

Bit	R/W	Symbol	Description	Reset Value
MLC RGB LAYER 0 PALETTE ADDRESS DATA (MLCPALETTE0) ***Address : C000_403Ch / C000_443Ch***				
[31:24]	W	**PALETTEADDR**	RGB Layer 0 Palette Table Write Address	8'bx
[23:16]		**RESERVED**	Reserved	8'bx
[15:0]	W	**PALETTEDATA**	RGB Layer 0 Palette Table Write Data	16'bx
MLC RGB LAYER 1 LEFT RIGHT REGISTER (MLCLEFTRIGHT1) ***Address : C000_4040h / C000_4440h***				
[31:28]	R	**RESERVED**	Reserved	4'b0
[11:0]	R/W	**LEFT**	Specifies the x-coordinate of the upper-left corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
[15:12]	R	**RESERVED**	Reserved	4'b0
[27:16]	R/W	**RIGHT**	Specifies the x-coordinate of the lower-right corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
MLC RGB LAYER 1 TOP BOTTOM REGISTER (MLCTOPBOTTOM1) ***Address : C000_4044h / C000_4444h***				
[31:28]	R	**RESERVED**	Reserved	4'b0
[27:16]	R/W	**TOP**	Specifies the y-coordinate of the upper-left corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
[15:12]	R	**RESERVED**	Reserved	4'b0
[11:0]	R/W	**BOTTOM**	Specifies the y-coordinate of the lower-right corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
MLC RGB LAYER 1 INVALID AREA 0 LEFT RIGHT REGISTER (MLCLEFTRIGHT1_0) ***Address : C000_4048h / C000_4448h***				
[31:29]	R	**RESERVED**	Reserved	3'b0
[28]	R/W	**INVALIDENB0**	Shows the status of disable/enable about 1st invisible area of RGB Layer1. 0 : Disable 1 : Enable	1'b0
[27]	R	**RESERVED**	Reserved	1'b0
[26:16]	R/W	**LEFT**	Notify left coordination of X axis from the area to be invisible of RGB Layer1. Able to set coordination from 0 to 2047.	11'b0
[15:11]	R	**RESERVED**	Reserved	5'b0
[10:0]	R/W	**RIGHT**	Notify right coordination of X axis from the area to be invisible of RGB Layer1. Able to set coordination from 0 to 2047.	11'b0
MLC RGB LAYER 1 INVALID AREA 0 BOTTOM TOP REGISTER (MLCTOPBOTTOM1_0) ***Address : C000_404Ch / C000_444Ch***				
[31:27]	R	**RESERVED**	Reserved	5'b0
[26:16]	R/W	**TOP**	Notify upper coordination of Y axis from the area to be invisible of RGB Layer1. Able to set coordination from 0 to 2047.	11'b0
[15:11]	R	**RESERVED**	Reserved	5'b0
[10:0]	R/W	**BOTTOM**	Notify lower coordination of Y axis from the area to be invisible of RGB Layer1. Able to set coordination from 0 to 2047.	11'b0
MLC RGB LAYER 1 INVALID AREA 1 LEFT RIGHT REGISTER (MLCLEFTRIGHT1_1) ***Address : C000_4050h / C000_4450h***				
[31:29]	R	**RESERVED**	Reserved	3'b0
[28]	R/W	**INVALIDENB1**	Shows the status of disable/enable about 2nd invisible area of RGB Layer1. 0 : Disable 1 : Enable	1'b0
[27]	R	**RESERVED**	Reserved	1'b0

Bit	R/W	Symbol	Description	Reset Value
[26:16]	R/W	**LEFT**	Notify left coordination of X axis from the area to be invisible of RGB Layer1. Able to set coordination from 0 to 2047.	11'b0
[15:11]	R	**RESERVED**	Reserved	5'b0
[10:0]	R/W	**RIGHT**	Notify right coordination of X axis from the area to be invisible of RGB Layer1. Able to set coordination from 0 to 2047.	11'b0
MLC RGB LAYER 1 INVALID AREA 1 BOTTOM TOP REGISTER (MLCTOPBOTTOM1_1) ***Address : C000_4054h / C000_4454h***				
[31:27]	R	**RESERVED**	Reserved	5'b0
[26:16]	R/W	**TOP**	Notify upper coordination of Y axis from the area to be invisible of RGB Layer1. Able to set coordination from 0 to 2047.	11'b0
[15:11]	R	**RESERVED**	Reserved	5'b0
[10:0]	R/W	**BOTTOM**	Notify lower coordination of Y axis from the area to be invisible of RGB Layer1. Able to set coordination from 0 to 2047.	11'b0
MLC RGB LAYER 1 CONTROL REGISTER (MLCCONTROL1) ***Address : C000_4058h / C000_4458h***				
[31:16]	R/W	**FORMAT**	Specifies the RGB data format. For detailed information, refer to Table 21-5.	16'b0
[15]	R/W	**PALETTEPWD**	RGB Layer1 Palette Table's Power On/Off it should be 'On' before RGB Layer0 Enabled 0 : Power Off 1 : Power On	1'b0
[14]	R/W	**PALETTESLD**	RGB Layer1 Palette Table's Sleep Mode. It is usable only when **PALETTEPWD = '1'** 0 : Sleep Mode Enable 1 : Sleep Mode Disable	1'b0
[13:12]	R/W	**LOCKSIZE**	Adjusts the data size to be read at any one time when a memory read for RGB data is requested. For a resolution of 1280 x 1024, it is recommended to set it as 16. 0 : 4 1 : 8 2 : 16 3 : reserved	2'b0
[11:9]	R	**RESERVED**	Reserved	3'b0
[8]	R/W	**RESERVED**	Reserved but you have to write '0' only	1'b0
[7:6]	R	**RESERVED**	Reserved	2'b0
[5]	R/W	**LAYERENB**	Enables/disables this layer. 0 : Disable 1 : Enable	1'b0
[4]	R/W	**DIRTYFLAG**	Dirty flag for this layer. If this bit is set as '1', the register settings concerned with this layer are updated in vertical sync mode and this bit is cleared as '0' automatically. Read) 0 : Clean 1 : Dirty Write) 0 : No affect 1 : Apply modified settings	1'b0
[3]	R	**RESERVED**	Reserved	1'b0
[2]	R/W	**BLENDENB**	Enables/disables the Alpha Blending function in this layer. In addition, this bit should be set as 'enable' to apply Alpha to a format with Alpha channels. 0 : Disable 1 : Enable	1'b0
[1]	R/W	**INVENB**	Enables/disables the Color Inversion function in this layer. 0 : Disable 1 : Enable	1'b0
[0]	R/W	**TPENB**	Enables/disables the Transparency function in this layer. 0 : Disable 1 : Enable	1'b0

Bit	R/W	Symbol	Description	Reset Value
MLC RGB LAYER 1 HORIZONTAL STRIDE REGISTER (MLCHSTRIDE1) ***Address : C000_405Ch / C000_445Ch***				
[31]	R/W	**RESERVED**	Reserved for future use. You have to write '0' only.	1'b0
[30:0]	R/W	**HSTRIDE**	Specifies the horizontal stride in byte units. This value is the byte offset from the current pixel to the next unit and has the number of bytes per pixel in general.	31'b0
MLC RGB LAYER 1 VERTICAL STRIDE REGISTER (MLCVSTRIDE1) ***Address : C000_4060h / C000_4460h***				
[31:0]	R/W	**VSTRIDE**	Specifies the vertical stride in byte units. This value is the byte offset from the current line to the next line and has the number of bytes per line in general. This value has the 2's complement format and can be specified as a negative number for the Vertical Flip function.	32'b0
MLC RGB LAYER 1 TRANSPARENCY COLOR REGISTER (MLCTPCOLOR1) ***Address : C000_4064h / C000_4464h***				
[31:28]	R/W	**ALPHA**	Specifies an Alpha Blending factor. This value is valid only for an RGB format without an Alpha channel. The formula for Alpha Blending is as follows: If ALPHA is 0 then a is 0, else a is ALPHA + 1. color = this layer color * a / 16 + lower layer color * (16-a) / 16	4'b0
[27:24]	R	**RESERVED**	Reserved	4'b0
[23:0]	R/W	**TPCOLOR**	Specifies the color for the Transparency. These bits have the R8G8B8 format in 24 bpp mode.	24'b0
MLC RGB LAYER 1 INVERSION COLOR REGISTER (MLCINVCOLOR1) ***Address : C000_4068h / C000_4468h***				
[31:24]	R	**RESERVED**	Reserved	8'b0
[23:0]	R/W	**INVCOLOR**	Specifies the color to be inverted. These bits have the R8G8B8 format in 24 bpp mode.	24'b0
MLC RGB LAYER 1 BASE ADDERSS REGISTER (MLCADDRESS1) ***Address : C000_406Ch / C000_446Ch***				
[31:0]	R/W	**ADDRESS**	Specifies the memory address where RGB data is stored. In general, the address on the top left end of the image is specified, but the address on the bottom left corner is specified for Vertical Flip.	32'b0
MLC RGB LAYER 1 PALETTE ADDRESS DATA (MLCPALETTE1) ***Address : C000_4070h / C000_4470h***				
[31:24]	W	**PALETTEADDR**	RGB Layer 0 Palette Table Write Address	8'bx
[23:16]		**RESERVED**	Reserved	8'bx
[15:0]	W	**PALETTEDATA**	RGB Layer 0 Palette Table Write Data	16'bx
MLC VIDEO LAYER LEFT RUGHT REGISTER (MLCLEFRIGHT2) ***Address : C000_4074h / C000_4474h***				
[31:28]	R	**RESERVED**	Reserved	4'b0
[27:16]	R/W	**LEFT**	Specifies the x-coordinate of the upper-left corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
[15:12]	R	**RESERVED**	Reserved	4'b0
[11:0]	R/W	**RIGHT**	Specifies the x-coordinate of the lower-right corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
MLC VIDEO LAYER TOP BOTTOM REGISTER (MLCTOPBOTTOM2) ***Address : C000_4078h / C000_4478h***				
[31:28]	R	**RESERVED**	Reserved	4'b0
[27:16]	R/W	**TOP**	Specifies the y-coordinate of the upper-left corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0
[15:12]	R	**RESERVED**	Reserved	4'b0
[11:0]	R/W	**BOTTOM**	Specifies the y-coordinate of the lower-right corner of the layer. This value has 2's complement and from –2048 to 2047.	12'b0

Bit	R/W	Symbol	Description	Reset Value
MLC VIDEO LAYER CONTROL REGISTER (MLCCONTROL2) ***Address : C000_407Ch / C000_447Ch***				
[31:16]	R	**RESERVED**	Reserved	16'b0
[15]	R/W	**LIENBUFFER_PWD**	Video Layer Line Buffer's Power On/Off o : Power Off 1 : Power On	1'b0
[14]	R/W	**LIENBUFFER_SLMD**	Video Layer Line Buffer's Sleep Mode. It is usable only when **LIENBUFFER_PWD = '1'** 0 : Sleep Mode Enable 1 : Sleep Mode Disable	1'b0
[5]	R/W	**LAYERENB**	Enables/disables this layer. 0 : Disable 1 : Enable	1'b0
[4]	R/W	**DIRTYFLAG**	Dirty flag for this layer. If this bit is set as '1', the register settings concerned with this layer are updated in vertical sync mode and this bit is cleared as '0' automatically. Read) 0 : Clean 1 : Dirty Write) 0 : No affect 1 : Apply modified settings	1'b0
[3]	R	**RESERVED**	Reserved	1'b0
[2]	R/W	**BLENDENB**	Enables/disables the Alpha Blending function in this layer. 0 : Disable 1 : Enable	1'b0
[1:0]	R/W	**RESERVED**	Reserved for future use. You have to write '0' only.	2'b0
MLC VIDEO LAYER VERTICAL STRIDE REGISTER (MLCVSTRIDE3) ***Address : C000_4080h / C000_4480h***				
[31:0]	R/W	**VSTRIDE**	Specifies the vertical stride in byte units. This value is the byte offset from the current line to the next line and has the number of bytes per line in general. In the 2D block addressing separated YUV format, this value only corresponds to the Y component and should be set as 4096 in general.	32'b0

Bit	R/W	Symbol	Description	Reset Value
MLC VIDEO LAYER TRANSPARENCY COLOR REGISTER (MLCTPCOLOR3) ***Address : C000_4084h / C000_4484h***				
[31:28]	R/W	**ALPHA**	Specifies an Alpha Blending factor. The formula for Alpha Blending is as follows: If ALPHA is 0 then a is 0, else a is ALPHA + 1. color = this layer color * a / 16 + lower layer color * (16-a) / 16	4'b0
[27:0]	R	**RESERVED**	Reserved	28'b0
RESERVED ***Address : C000_4088h / C000_4488h***				
[31:0]	-	**RESERVED**	Reserved	-
MLC VIDEO LAYER BASE ADDERSS REGISTER (MLCADDRESS3) ***Address : C000_408Ch / C000_448Ch***				
[31:0]	R/W	**ADDRESS**	Specifies the memory address where YUB data is stored. In the 2D block addressing separated YUV format, this value only corresponds to the Y component and should be set as follows: • ADDRESS[31:24] : the index of segment • ADDRESS[23:12] : y-coordinate in segment • ADDRESS[11:0] : x-coordinate in segment, ADDRESS[2:0] must be '0'	32'b0
MLC VIDEO LAYER CB BASE ADDERSS REGISTER (MLCADDRESSCB) ***Address : C000_4090h / C000_4490h***				
[31:0]	R/W	**ADDRESSCB**	Specifies the memory address where Cb data is stored. These bits are valid only for the 2D block addressing separated YUV format and should be set as follows: • ADDRESS[31:24] : the index of segment • ADDRESS[23:12] : y-coordinate in segment • ADDRESS[11:0] : x-coordinate in segment, ADDRESS[2:0] must be '0'	32'b0
MLC VIDEO LAYER CR BASE ADDERSS REGISTER (MLCADDRESSCR) ***Address : C000_4094h / C000_4494h***				
[31:0]	R/W	**ADDRESSCR**	Specifies the memory address where Cr data is stored. These bits are valid only for the 2D block addressing separated YUV format and should be set as follows: • ADDRESS[31:24] : the index of segment • ADDRESS[23:12] : y-coordinate in segment • ADDRESS[11:0] : x-coordinate in segment, ADDRESS[2:0] must be '0'	32'b0
MLC VIDEO LAYER CB VERTICAL STRIDE REGISTER (MLCVSTRIDECB) ***Address : C000_4098h / C000_4498h***				
[31:0]	R/W	**VSTRIDECB**	Specifies the vertical stride for Cr data in byte units. These bits are valid only for the 2D block addressing separated YUV format and should be set as 4096 in general.	32'b0
MLC VIDEO LAYER CR VERTICAL STRIDE REGISTER (MLCVSTRIDECR) ***Address : C000_409Ch / C000_449Ch***				
[31:0]	R/W	**VSTRIDECR**	Specifies the vertical stride for Cr data in byte units. These bits are valid only for the 2D block addressing separated YUV format and should be set as 4096 in general.	32'b0
MLC VIDEO LAYER HORIZONTAL SCALE REGISTER (MLCHSCALE) ***Address : C000_40A0h / C000_44A0h***				
[31:29]	R	**RESERVED**	Reserved	3'b0
[28]	R/W	**HFILTERENB**	Decides whether to use bilinear filter when Video Layer horizontal scale. 0 : Disable (point sample) 1 : Enable (bilinear filter)	1'b0
[27:23]	R	**RESERVED**	Reserved	5'b0
[22:0]	R/W	**HSCALE**	Specifies the ratio for the horizontal scale. The formula to calculate this value is as follows: When FILTERENB is 1 and the destination width is wider than the source width: • HSCALE = (source width-1) * (1<<11) / (destination width-1) , else • HSCALE = source width * (1<<11) / destination width	23'h800

Bit	R/W	Symbol	Description	Reset Value
MLC VIDEO LAYER VERTICAL SCALE REGISTER (MLCVSCALE) ***Address : C000_40A4h / C000_44A4h***				
[31:29]	R	**RESERVED**	Reserved	3'b0
[28]	R/W	**VFILTERENB**	Decides whether to use bilinear filter when Video Layer vertical scale. VFILTERENB must be set as '1' when Veritcal Scale up 0 : Disable (Nearest sample) 1 : Enable (bilinear filter)	1'b0
[27:23]	R	**RESERVED**	Reserved	5'b0
[22:0]	R/W	**VSCALE**	Specifies the ratio for the vertical scale. The formula to calculate this value is as follows: When FILTERENB is 1, the destination height is higher than the source height: • VSCALE = (source height-1) * (1<<11) / (destination height-1) , else • VSCALE = source height * (1<<11) / destination height	23'h800
MLC VIDEO LAYER LUMINANACE ENHANCEMENT CONTROL REGISTER (MLCLUENH) ***Address : C000_40A8h / C000_44A8h***				
[31:16]	R	**RESERVED**	Reserved	16'b0
[15:8]	R/W	**BRIGHTNESS**	Specifies brightness values in 256 levels. These values are 2's complements and can be set between -128 and +127.	8'b0
[7:3]	R	**RESERVED**	Reserved	5'b0
[2:0]	R/W	**CONTRAST**	Specifies contrast levels with 8 levels. 0 : 1.0 1 : 1.125 2 : 1.25 3 : 1.375 4 : 1.5 5 : 1.625 6 : 1.75 7 : 1.875	3'b0
MLC VIDEO LAYER CHROMINANCE ENHANCEMENT CONTROL 0 REGISTER (MLCCHENH0) ***Address : C000_40ACh / C000_44ACh***				
[31:24]	R/W	**HUECR1B**	Specifies the factors for Hue and Saturation for the first quadrant. Each value has the 2's complement format and the format is [S.1.6] (here, S means a sign bit). The Hues and Saturations for each quadrant are determined by the following formulae: • (B-Y)' = (B-Y) * HUECBnA + (R-Y) * HUECBnB • (R-Y)' = (B-Y) * HUECRnA + (R-Y) * HUECRnB The formulae for each factor are as follows: • HUECBnA = cos(θ) * 64 * gain • HUECBnB = -sin(θ) * 64 * gain • HUECRnA = sin(θ) * 64 * gain • HUECRnB = cos(θ) * 64 * gain where the gain value is between 0 and 2.	8'h40
[23:16]	R/W	**HUECR1A**		8'b0
[15:8]	R/W	**HUECB1B**		8'b0
[7:0]	R/W	**HUECB1A**		8'h40
MLC VIDEO LAYER CHROMINANCE ENHANCEMENT CONTROL 1 REGISTER (MLCCHENH1) ***Address : C000_40B0h / C000_44B0h***				
[31:24]	R/W	**HUECR2B**	Specifies the factors for Hue and Saturation for the second quadrant. Each value has the 2's complement format and the format is [S.1.6] (here, S means a sign bit). The Hues and Saturations for each quadrant are determined by the following formulae: • (B-Y)' = (B-Y) * HUECBnA + (R-Y) * HUECBnB • (R-Y)' = (B-Y) * HUECRnA + (R-Y) * HUECRnB The formulae for each factor are as follows: • HUECBnA = cos(θ) * 64 * gain • HUECBnB = -sin(θ) * 64 * gain • HUECRnA = sin(θ) * 64 * gain • HUECRnB = cos(θ) * 64 * gain where the gain value is between 0 and 2.	8'h40
[23:16]	R/W	**HUECR2A**		8'b0
[15:8]	R/W	**HUECB2B**		8'b0
[7:0]	R/W	**HUECB2A**		8'h40
MLC VIDEO LAYER CHROMINANCE ENHANCEMENT CONTROL 2 REGISTER (MLCCHENH2) ***Address : C000_40B4h / C000_44B4h***				
[31:24]	R/W	**HUECR3B**	Specifies the factors for Hue and Saturation for the third quadrant. Each value has the 2's complement format and the format is [S.1.6] (here, S means a sign bit). The Hues and Saturations for each quadrant are determined by the following formulae: • (B-Y)' = (B-Y) * HUECBnA + (R-Y) * HUECBnB • (R-Y)' = (B-Y) * HUECRnA + (R-Y) * HUECRnB The formulae for each factor are as follows: • HUECBnA = cos(θ) * 64 * gain • HUECBnB = -sin(θ) * 64 * gain • HUECRnA = sin(θ) * 64 * gain • HUECRnB = cos(θ) * 64 * gain where the gain value is between 0 and 2.	8'h40
[23:16]	R/W	**HUECR3A**		8'b0
[15:8]	R/W	**HUECB3B**		8'b0
[7:0]	R/W	**HUECB3A**		8'h40
MLC VIDEO LAYER CHROMINANCE ENHANCEMENT CONTROL 3 REGISTER (MLCCHENH3) ***Address : C000_40B8h / C000_44B8h***				
[31:24]	R/W	**HUECR4B**	Specifies the factors for Hue and Saturation for the fourth quadrant. Each value has the 2's complement format and the format is [S.1.6] (here, S means a sign bit). The Hues and Saturations for each quadrant are determined by the following formulae: • (B-Y)' = (B-Y) * HUECBnA + (R-Y) * HUECBnB • (R-Y)' = (B-Y) * HUECRnA + (R-Y) * HUECRnB	8'h40
[23:16]	R/W	**HUECR4A**		8'b0

Bit	R/W	Symbol	Description	Reset Value
[15:8]	R/W	**HUECB4B**	The formulae for each factor are as follows: • HUECBnA = cos(θ) * 64 * gain • HUECBnB = -sin(θ) * 64 * gain • HUECRnA = sin(θ) * 64 * gain • HUECRnB = cos(θ) * 64 * gain where the gain value is between 0 and 2.	8'b0
[7:0]	R/W	**HUECB4A**		8'h40

Bit	R/W	Symbol	Description	Reset Value
RESERVED ***Address : C000_40BCh ~ 43BFh / C000_44BCh ~ 47BFh***				
[31:0]	-	**RESERVED**	Reserved	-
MLC CLOCK GENERATION ENABLE REGISTER (MLCCLKENB) ***Address : C000_43C0h / C000_47C0h***				
[31:24]	R	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	Specifies PCLK operating mode. 0 : PCLK is only enabled when CPU accesses this module 1 : PCLK is always enabled	1'b0
[2]	R	**RESERVED**	Reserved	1'b0
[1:0]	R/W	**BCLKMODE**	Specifies BCLK operating mode. 0 : BCLK is always disabled 1 : Reserved for future use 2 : BCLK is dynamic controlled by this module 3 : BCLK is always enabled	2'b0

CHAPTER 21.

DISPLAY CONTROLLER (DPC)

21. DISPLAY CONTROLLER

21.1. Overview

The Display controller (hereinafter, DPC) is a block that generates the signals to interface with external display devices, such as a TFT-LCD, STN_LCD, or video encoder. The DPC consists of a Sync generator, internal video encoder and DAC. The Sync generator transmits control signals to the Multi-Layer Controller (MLC) and receives RGB data from the MLC. Then the Sync generator converts the received RGB data into a suitable format and outputs to an exterior device or transfers to the internal video encoder. In addition, the Sync generator can support various types of LCD and video encoders adjusting various output formats and Sync signals.
The **POLLUX** has two DPCs for dual display. The primary DPC can connect to a TFT-LCD or a STN_LCD. The secondary DPC can connect to a TV via the CVBS output.

21.1.1. Features

- Supports RGB, Multiplexed RGB (MRGB), ITU-R BT.601 and ITU-R BT.656
- Programmable HSYNC, VSYNC and DE (Data Enable)
- Programmable polarity for Sync signals
- Programmable delay for data, Sync signals and clock phase
- Supports dual display
- Supports NTSC/PAL TV output(Only Secondary Display)
- Supports UPScaler (Only Secondary Display)
- Supports direct interface to STN LCD panel with FRM, CP1, CP2
- 4096 colors (4:4:4) with color STN-LCD, 16-gray level with Monochrome STN-LCD(Only Primary Display)
- Supports STN LCD Frame Rate Controller (Only Primary Display)
- Supports RGB dithering

21.1.2. Block Diagram

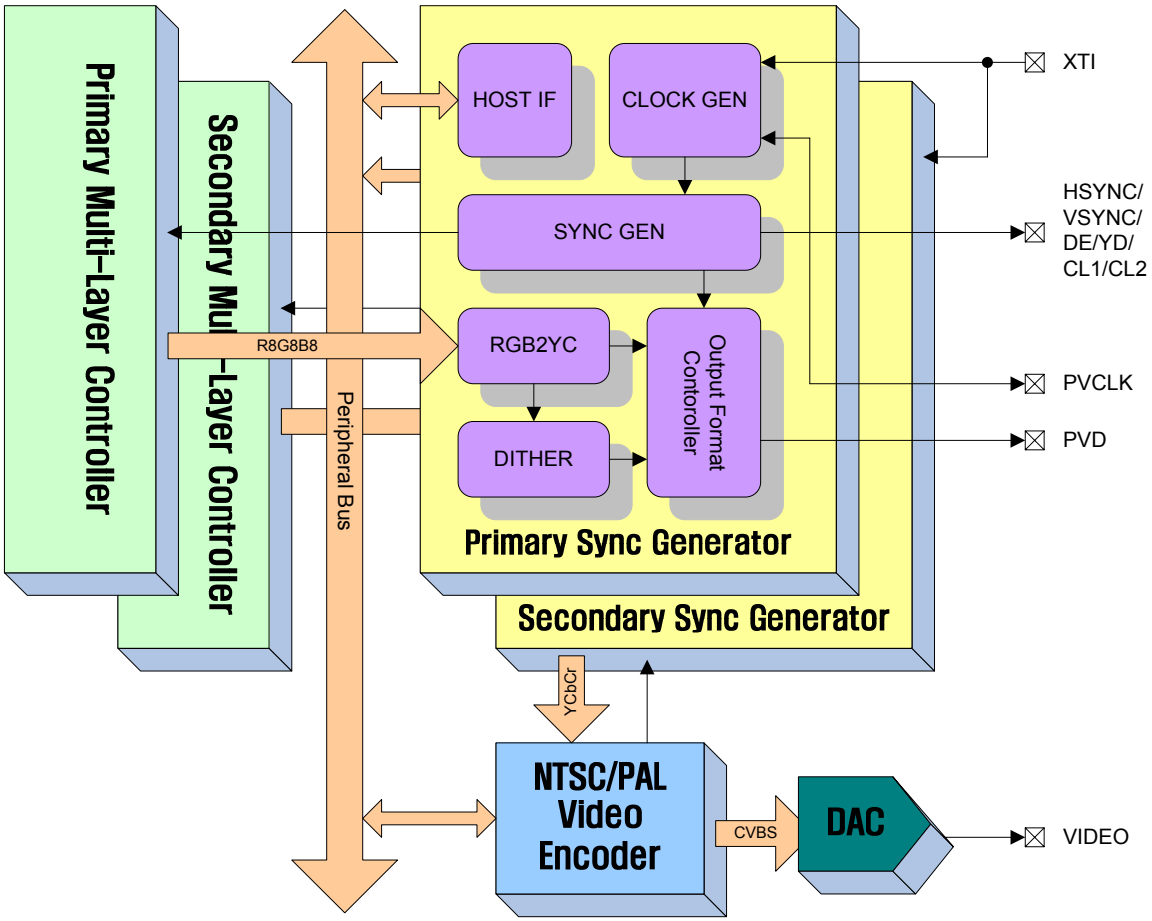


Figure 21-1. Display Controller

21.2. Pin configuration

Pins for the DPC are listed in Table 21-1. Since specific pins are shared with another function, the pins should be configured appropriately for the user's purpose.

Pin	GPIO	Func	Dir	Brief Description
External Clock				
XTI	–	–	I	System Crystal Input
AVCLK			I	Audio/Video External Clock Input
Primary Display Controller				
PVCLK	–	–	I/O	Primary Pixel Clock Input/Output
PVSYNC/YD	–	–	O	TFT or Video Encoder: Primary Vertical Sync signal STN : Primary Frame Start Signal
PHSYNC/CL1	–	–	O	TFT or Video Encoder : Primary Horizontal Sync signal STN : Primary Latch pulse of display data
PDE/CL2	–	–	O	TFT : Primary Data Enable signal STN : Primary Clock pulse for segment shift register
PVD[23:0]			O	Primary Video Output Data (RGB, Multiplexed-RGB, YCbCr)
Secondary Display Controller				
VIDEOOUT	–	–	AO	CVBS output (Analog TV Output)

Table 21-1. Pin Configuration

21.3. Dual Display

The **POLLUX** has two DPCs, a primary DPC and a secondary DPC, to support dual display. The primary DPC consists only of the primary Sync generator and creates various formats and interface signals for direct interface with external display devices. The secondary DPC consists of the secondary sync generator, a video encoder and DAC to support TV output (CVBS).

Display Controller	Base Address	Description
Primary Display Controller	**0xC000_3000**	**Without NTSC/PAL Encoder**
Secondary Display Controller	**0xC000_3400**	**With NTSC/PAL Encoder**

Table 21-2. Register Base Address

21.4. Sync Generator

21.4.1. TFT/Video Encoder Block Diagram

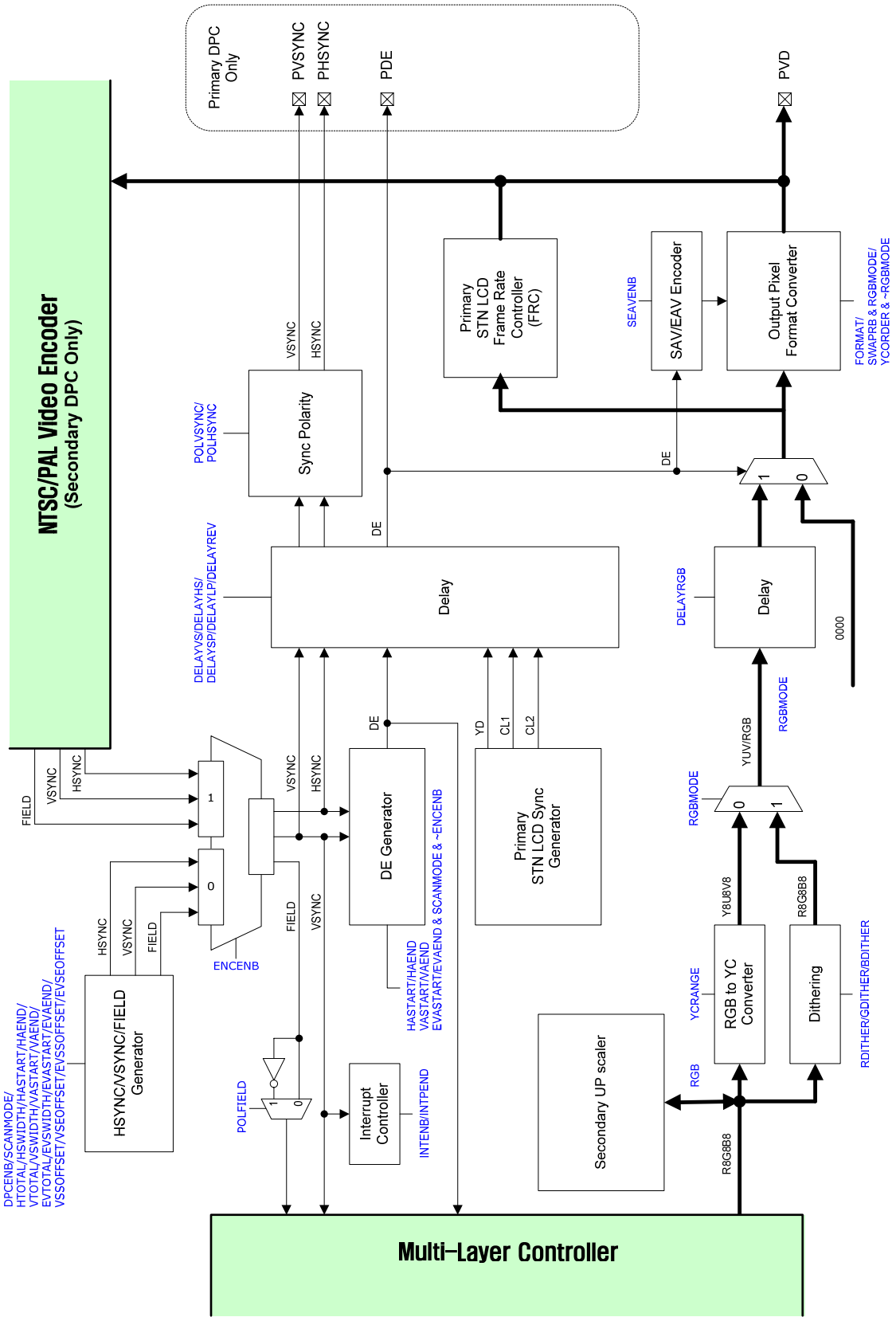


Figure 21-2. Sync Generator

21.4.2. Clock generation

21.4.2.1. Peripheral Clock Generation

The PCLK is used when the CPU accesses the registers of the DPC. Users can adjust the DPC clock by setting the ***PCLKMODE*** for the user's purpose.

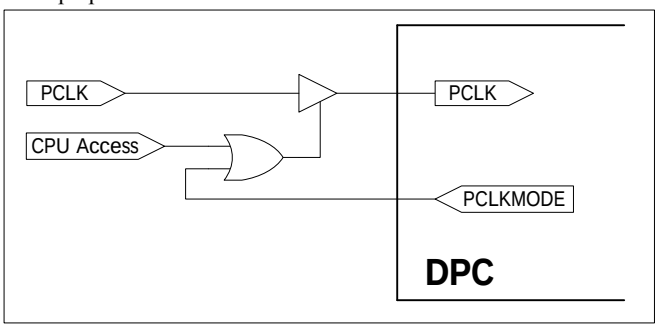


Figure 21-3. Peripheral Clock Generation

PCLKMODE	Brief Description
0	Only enabled when CPU accesses DPC Register.
1	Always enabled..

Table 21-3. PCLK mode

21.4.2.2. Video Clock Generation

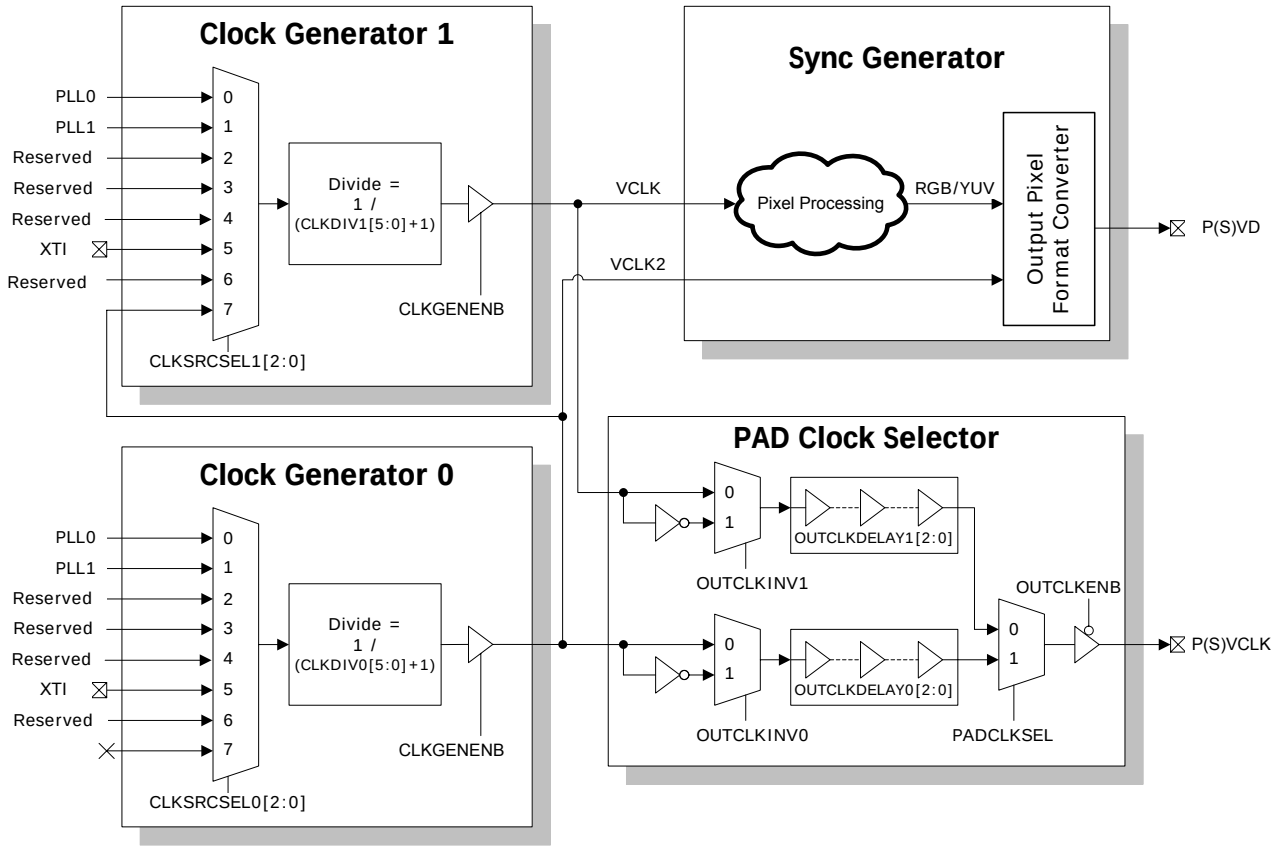


Figure 21-4. Video Clock Generation

The DPC can create an output clock via various clock sources. The clocks for creating an output clock are 2-PLL, XTI, etc.

The sync generator uses the VCLK and the VCLK2 as clock sources. The VCLK is a clock for the operation of a pixel unit. The VCLK2 is a clock for pixel output. Therefore, the VCLK is one clock per pixel and the VCLK2 is one or two clocks per pixel depending on the output format. In the RGB and ITU-R BT.601A formats, which output one pixel data per clock, the VCLK and the VCLK2 share the same clock. In the MRGB, the ITU-R BT.601B, the ITU-R BT.601(8 bit) and the ITU-R BT.656 formats, which output one pixel data per two clocks, the VCLK should divide the VCLK2 by 2. Since the VCLK should divide the VCLK2 by 2, Clock Generator1 should use the output of Clock Generator0 as the clock source. Users can select the output source by using the ***PADCLKSEL***. The selection of the ***PADCLKSEL*** is not related to the operation of the sync generator. In general, VCLK2 is used as the output clock. If, however, a display device using a dual edge is connected, the VCLK is used as the output clock.

Format	CLKSRCSEL0[2:0]	CLKDIV0[5:0]	CLKSRCSEL1[2:0]	CLKDIV1[5:0]	PADCLKSEL
RGB, ITU-R BT.601A	0 ~ 6	0 ~ 64	7	0	1
MRGB, ITU-R BT.601B, 601(8bit), 656	0 ~ 6	0 ~ 64	7	1	1
MRGB (Dual edge)	0 ~ 6	0 ~ 64	7	1	0
4069 Color/ 16 Level Gray	0 ~ 6	0 ~ 64	7	0	x

Table 21-4. Recommend Clock Settings

The DPC can adjust the polarity and phase of the output clock. The ***OUTCLKINV*** adjusts the polarity of the output clock and the ***OUTCLKDELAY*** adjusts the phase of the output clock.

Basically, the DPC outputs data to be fetched at the falling edge and the ***OUTCLKINV*** is set as '0'. For a display device that fetches the clock at the rising edge, the ***OUTCLKINV*** should be set as '1' to invert the output clock.

The ***OUTCLKDELAY*** can adjust the phase of the output clock by 0.5 ns and delay the output clock by up to 3.5 ns. The delays of the output clock for the ***OUTCLKDELAY*** are listed in Table 21-5.

OUTCLKDELAY	0	1	2	3	4	5	6	7
tDelay (ns)	0	0.5	1.0	1.5	2.0	2.5	3.0	3.5

Table 21-5. Clock delay

The output clock is changed by the ***OUTCLKINV*** and the ***OUTCLKDELAY*** as shown in Figure 21-5.

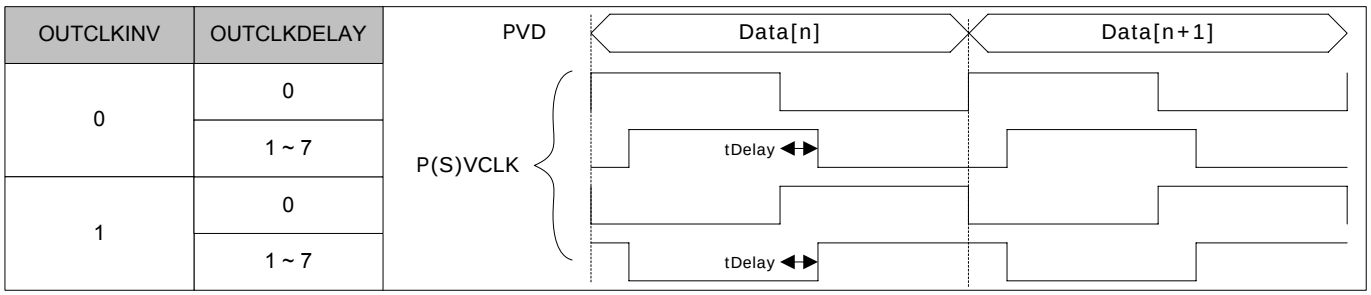


Figure 21-5. Clock polarity and delay

21.4.3. Format

The sync generator can receive RGB888 data from the MLC and display the data in various formats.

The formats that can be displayed by the primary sync generator are listed in Table 21-6.

TFT or Video Encoder Format

Output format	RGBMODE	FORMAT	CLK	PVD																							
				23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGB555	1	0	-										R4	R3	R2	R1	R0	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0
RGB565		1	-									R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0
RGB666		2	-							R5	R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B5	B4	B3	B2	B1	B0
RGB888		3	-	R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4	G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0
MRGB555A		4	1st																	G2	G1	G0	B4	B3	B2	B1	B0
			2nd																		R4	R3	R2	R1	R0	G4	G3
MRGB555B		5	1st																	G2	G1	G0	B4	B3	B2	B1	B0
			2nd																	R4	R3	R2	R1	R0		G4	G3
MRGB565		6	1st																	G2	G1	G0	B4	B3	B2	B1	B0
			2nd																	R4	R3	R2	R1	R0	G5	G4	G3
MRGB666		7	1st																G2	G1	G0	B5	B4	B3	B2	B1	B0
			2nd																R5	R4	R3	R2	R1	R0	G5	G4	G3
MRGB888A		8	1st													G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0
			2nd													R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4
MRGB888B		9	1st													G4	G3	G2	B7	B6	B5	B4	B3	G0	B2	B1	B0
			2nd													R7	R6	R5	R4	R3	G7	G6	G5	R2	R1	R0	G1
ITU-R BT.656 / ITU-R BT.601(8bit)	0	10	1st																	Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0
			2nd																	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
			3rd																	Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0
			4th																	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
ITU-R BT.601A		12	1st									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0	Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0
			2nd									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0	Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0
ITU-R BT.601B		13	1st									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0	Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0
			2nd																	Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0
4096 Color	X	1	-																	R0	G0	B0	R1	B1	G1	R2	G2
16 Level Gray	X	3	-																	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7

Table 21-6. Data format for Primary Sync Generator

Up to 24-bit data is available in the primary sync generator. If, however, the display format is not RGB666 or RGB888, the higher 8-bit is not used.

21.4.3.1. RGB Format

In the RGB format, data are displayed in the order of blue components, green components and red components based on the lower data. However, a user can swap the display of red components and blue components by setting the ***SWAPRB*** as '1'.

In addition, the DPC supports the Dithering effect in RGB format. All RGB data transmitted from the MLC have 8-bit data width. Therefore, if the data width of each component is less than 8-bit, as in RGB565 and RGB666 display, the lower bits are discarded. In such cases, the display quality can be compensated by the Dithering effect.

When RGB images are displayed as RGB565, the dithered image shows the difference from the image displayed, as shown in Figure 22-6, after the unused lower bits are simply discarded.

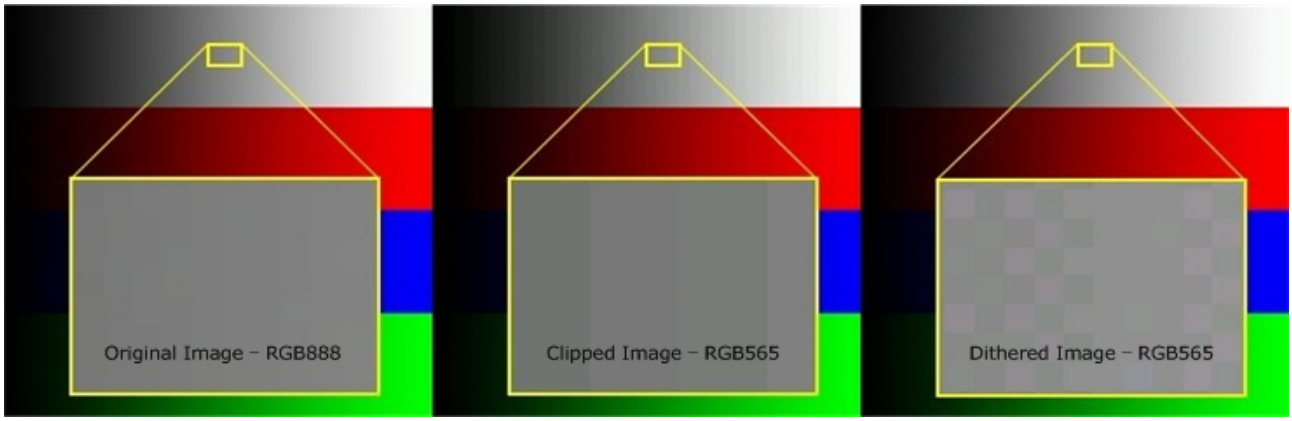


Figure 21-6. RGB Dithering

The RGB Dithering is set by ***RDITHER***, ***GDITHER*** and ***BDITHER*** and the setting values by output formats are listed in Table 21-7.

Output Format	RDITHER	GDITHER	BDITHER
RGB555, MRGB555A, MRGB555B	5 bit Dither	5 bit Dither	5 bit Dither
RGB565, MRGB565	5 bit Dither	6 bit Dither	5 bit Dither
RGB666, MRGB666	6 bit Dither	6 bit Dither	6 bit Dither
STN - 4096 Color, 16 Level Gray	4 bit Dither	4 bit Dither	4 bit Dither
RGB888, MRGB888A/B, ITU-R BT.656, ITU-R BT.601A/B	Bypass	Bypass	Bypass

Table 21-7. Recommend Setting for RGB Dithering

21.4.3.2. YCbCr Format

Since the DPC only receives RGB data from the MLC, it outputs the data after converting the RGB data to YCbCr data for the ITU-R BT.656 display or the ITU-R BT.601 display. The DPC converts RGB data to YCbCr data by using the following formulae:

※ The formula for RGB to YCbCr conversion

- Y = 0.229 * R + 0.587 * G + 0.114 * B
- Cb = – 0.169 * R – 0.331 * G + 0.5 * B
- Cr = 0.5 * R – 0.419 * G – 0.081 * B

When ITU-R BT.601 B external display device is used, the user can change the data output order by adjusting the ***YCORDER***. The output data for ***YCORDER*** can be changed as listed in Table 21-8.

YCORDER	0		1	
CLK	1st	2nd	1st	2nd
VD[15]	Y[7]		Y[7]	
VD[14]	Y[6]		Y[6]	
VD[13]	Y[5]		Y[5]	
VD[12]	Y[4]		Y[4]	
VD[11]	Y[3]		Y[3]	
VD[10]	Y[2]		Y[2]	
VD[9]	Y[1]		Y[1]	
VD[8]	Y[0]		Y[0]	
VD[7]	Cr[7]	Cb[7]	Cb[7]	Cr[7]
VD[6]	Cr[6]	Cb[6]	Cb[6]	Cr[6]

VD[5]	Cr[5]	Cb[5]	Cb[5]	Cr[5]
VD[4]	Cr[4]	Cb[4]	Cb[4]	Cr[4]
VD[3]	Cr[3]	Cb[3]	Cb[3]	Cr[3]
VD[2]	Cr[2]	Cb[2]	Cb[2]	Cr[2]
VD[1]	Cr[1]	Cb[1]	Cb[1]	Cr[1]
VD[0]	Cr[0]	Cb[0]	Cb[0]	Cr[0]

Table 21-8. Output order for ITU-R BT. 601 B

21.4.4. Sync signals

The sync generator creates sync signals with various timings. The primary sync generator transmits HSYNC, VSYNC and DE signals to the outside to provide timing interfaces for external display devices. Users can program each sync signal setting to create the timings required from external display devices.

21.4.4.1. Horizontal Timing Interface

HSYNC and DE signals are used for Horizontal Sync. The horizontal timing consists of tHSW, tHBP, tHFP and tAVW as shown in Figure 22-7.

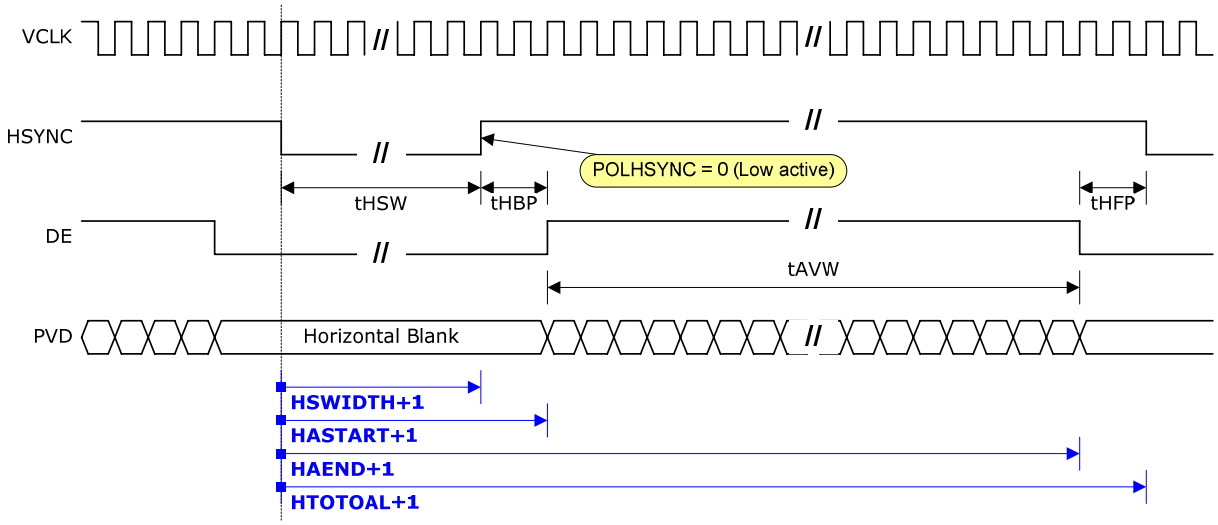


Figure 21-7. Horizontal timing

Each symbol in the above figure is described in Table 21-9.

Symbol	Brief	Remark
tHSW	Horizontal Sync Width	Number of VCLKs in the section where the horizontal sync pulse is active
tHBP	Horizontal Back Porch	Number of VCLKs in a section from the end point of the horizontal sync pulse to the start point of the horizontal active
tHFP	Horizontal Front Porch	Number of VCLKs in a section from the end point of the horizontal active to the start point of the horizontal sync pulse
tAVW	Active Video Width	Number of VCLKs in a horizontal active section

Table 21-9. Horizontal timing symbols

The horizontal timing setting registers are ***HSWIDTH***, ***HASTART***, ***HAEND*** and ***HTOTAL*** and each register setting is described in Table 21-10. Each unit of the registers is based on VCLK and each value is set as 'total number – 1'.

Register	Formula	Remark
HSWIDTH	tHSW – 1	Number of VCLKs in a section from the start point of the horizontal sync pulse to the end point of the horizontal sync pulse - 1
HASTART	tHSW + tHBP – 1	Number of VCLKs in a section from the start point of the horizontal sync pulse to the start point of the horizontal active - 1
HAEND	tHSW + tHBP + tAVW – 1	Number of VCLKs in a section from the start point of the horizontal sync pulse to the end point of the horizontal active - 1

HTOTAL	tHSW + tHBP + tAVW + tHFP – 1	Number of VCLKs in a section from the start point of the horizontal sync pulse to the start point of the next horizontal sync pulse - 1

Table 21-10. Horizontal timing registers

The ***POLHSYNC*** is used to change the polarity of the HSYNC signal to be output to the outside. The horizontal sync pulse is low active when the ***POLHSYNC*** is '0', while the horizontal sync pulse is high active when the ***POLHSYNC*** is '1'. The polarity of the Data Enable (DE) signal to be output to the outside cannot be changed and the section that outputs valid data comes into high state.

21.4.4.2. Vertical Timing Interface

The VSYNC signal is used for the Vertical Sync. The vertical timing consists of tVSW, tVBP, tVFP and tAVH as shown in Figure 22-8. In addition, the relation between the Vertical Sync and the Horizontal Sync is established by tVSSO and tVSEO.

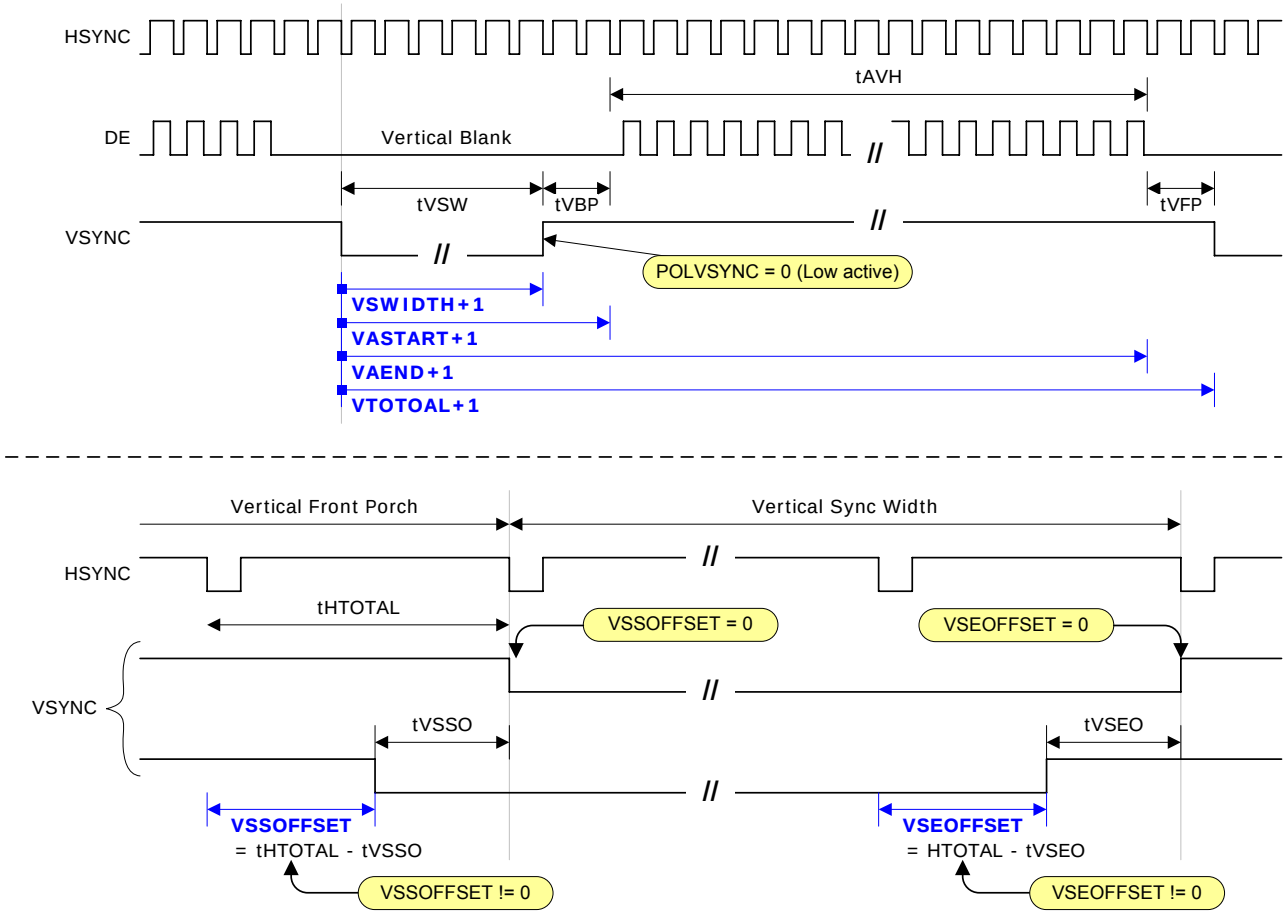


Figure 21-8. Vertical timing

Each symbol in the above figure is described in Table 21-11.

Symbol	Brief	Remark
tVSW	Vertical Sync Width	Number of lines in the section where the vertical sync pulse is active
tVBP	Vertical Back Porch	Number of lines in the section from the end point of the vertical sync pulse to the start point of the next vertical active
tVFP	Vertical Front Porch	Number of lines in the section from the end point of the vertical active to the start point of the vertical sync pulse
tAVH	Active Video Height	Number of lines in the vertical active section
tHTOTAL	Horizontal Total	Number of total VCLKs in a horizontal cycle where the horizontal active section and the horizontal blank section are added
tVSSO	Vertical Sync Start Offset	Number of VCLKs in a section from the start point of the vertical sync pulse to the start point of the horizontal sync pulse
tVSEO	Vertical Sync End Offset	Number of the VCLKs in a section from the end point of the horizontal sync pulse to the start point of the horizontal sync pulse

Table 21-11. Vertial timing symbols

The vertical timing setting registers are ***VSWIDTH***, ***VASTART***, ***VAEND*** and ***VTOTAL*** and each register setting is described in Table 21-12. The units of ***VSWIDTH***, ***VASTART***, ***VAEND*** and ***VTOTAL*** are based on the horizontal lines and their values are set as 'total number – 1'. The units of ***VSSOFFSET*** and ***VSEOFFSET*** are based on the VCLK.

Register	Formula	Remark
VSWIDTH	tVSW – 1	Number of lines in a section from the start point of the vertical sync pulse to the end point of the vertical sync pulse - 1
VASTART	tVSW + tVBP – 1	Number of VCLKs in a section from the start point of the vertical sync pulse to the start point of the vertical active - 1
VAEND	tVSW + tVBP + tAVH – 1	Number of lines in a section from the start point of the vertical sync pulse to the end point of the vertical active - 1
VTOTAL	tVSW + tVBP + tAVH + tVFP – 1	Number of lines in a section from the start point of the vertical sync pulse to the next point of the vertical sync pulse - 1
VSSOFFSET	If tVSSO is 0 then 0, else tHTOTAL – tVSSO	Number of VCLKs in a section from the start point of the last horizontal sync pulse in the Vertical Front Porch to the start point of the vertical sync pulse. If, however, VSSOFFSET is equal to tHTOTAL, the value is set as '0'.
VSEOFFSET	If tVSEO is 0 then 0, else tHTOTAL – tVSEO	Number of VCLKs in a section from the start point of the last horizontal sync pulse in the Vertical Front Porch to the start point of the vertical sync pulse. If, however, VSEOFFSET is equal to tHTOTAL, the value is set as '0'.

Table 21-12. Vertical timing registers

The ***POLVSYNC*** is used to change the polarity of the VSYNC signal to be output to the outside. The vertical sync pulse is low active when the ***POLVSYNC*** is '0', while the vertical sync pulse is high active when the ***POLVSYNC*** is '1'.

21.4.4.3. AC Timing

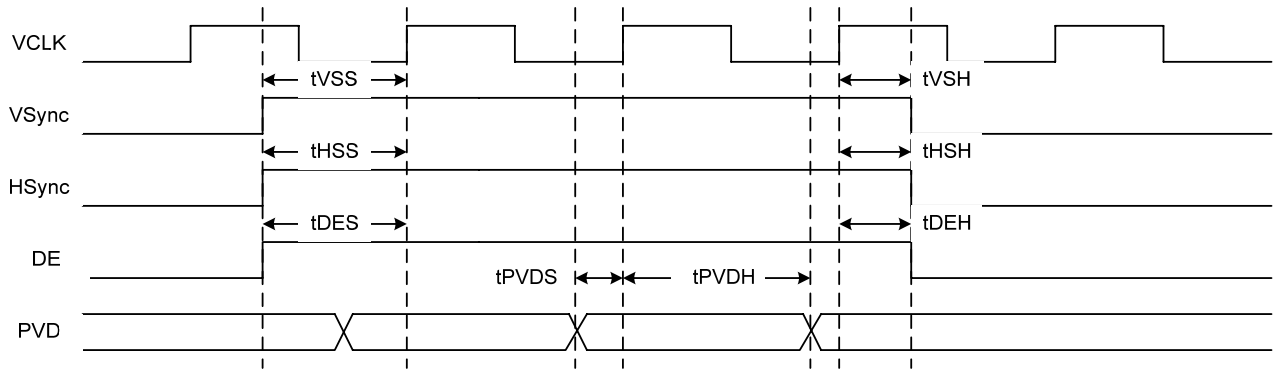


Figure 21-9. Vertical timing

Condition : VCLK 40MHz(25ns)

Symbol	Min	Max	Unit	Description
tVSS	12.13	-	ns	VSync setup time
tVSH	12.28	-	ns	VSync hold time
tHSS	12.27	-	ns	HSync setup time
tHSH	12.26	-	ns	HSync hold time
tDES	12.1	-	ns	DE setup time
tDEH	12.4	-	ns	DE hols time
tPVDS	8.8	-	ns	PVD setup time
tPVDH	12.87	-	ns	PVD hold time

Table 21-13. sync timing symbols

21.4.4.4. STN LCD Timing Interface(Only Primary Display)

The following is the Timing image based on the 320x240 Size, 4096 Color and STN LCD.

Horizontal Timing Interface

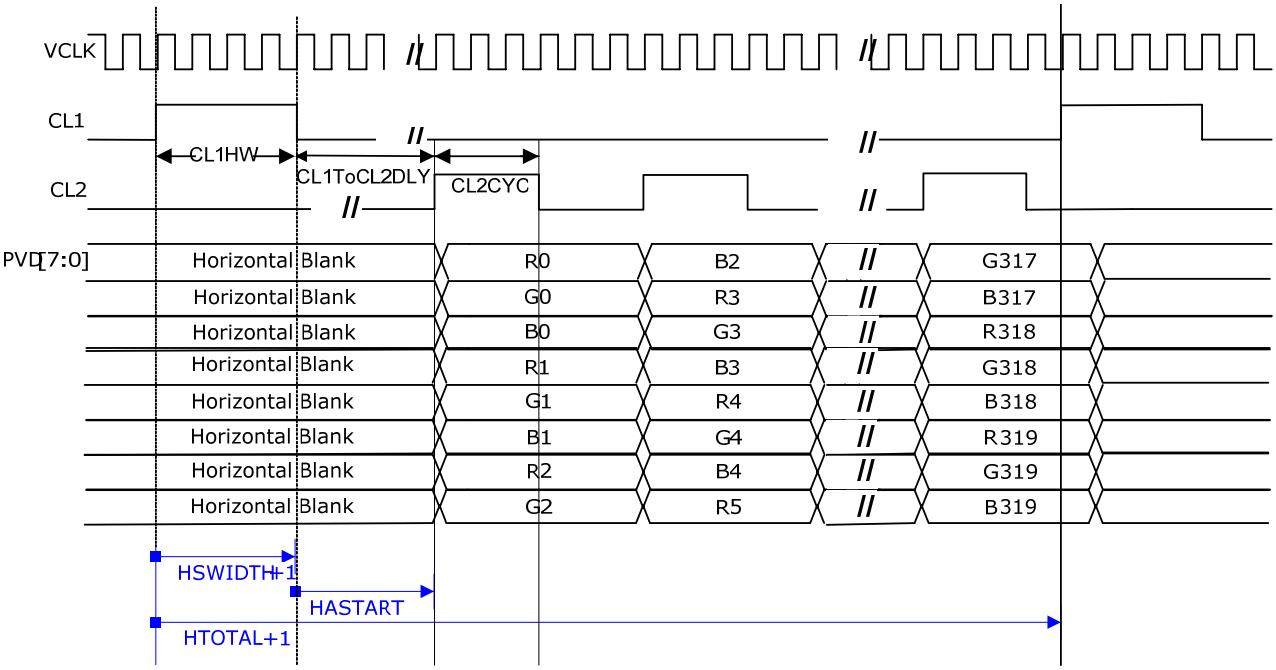


Figure 21-10. STN Horizontal timing

Symbol	Brief	Description
CL1HW	CL1 High Width	Define CL1 high region. (n-1)
CL1ToCL2DLY	CL1 to CL2 Delay	Define cycle from CL1 fall to first CL2 rise. (n)
CL2CYC	CL2 Cycle/2	Define CL2 Half Cycle. (n-1)

Table 21-14. STN Horizontal timing symbols

The horizontal timing setting registers are ***HSWIDTH***, ***HASTART***, ***HAEND*** and ***HTOTAL*** and each register setting is described in Table 21-10. Each unit of the registers is based on VCLK and each value is set as 'total number – 1'.

Register	Formula	Description
HSWIDTH	CL1HW – 1	Define CL1 hight region.
HASTART	CL1ToCL2DLY	Define cycle from CL1 fall to first CL2 rise.
HAEND	tAVW(Active pixel width) -1	Define total pixel of one line.
HTOTAL	CL1HW + CL1ToCL2DLY + ((CL2CYC*2)*(destwidth/BusWidth)) – 1	Define CL2 Half Cycle.

Table 21-15. STN Horizontal timing registers

Vertical Timing Interface

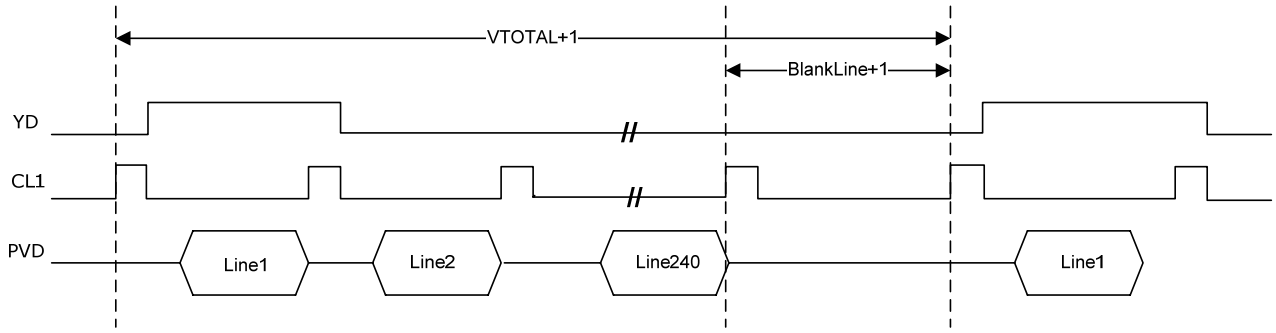


Figure 21-11. STN Vertical timing

Register	Formula	Description
VSWIDTH	BlankLine – 1	Define region from Last Line to first line.
VTOTAL	BlankLine + ActiveLineNumber -1	Define total line number (Active line number + blank line number)

Table 21-16. STN Vertical timing registers

21.4.4.5. Frame Rate Control(Only Primary Display)

FRC Block provides the function of Frame Dither & Random Number Generator and is the the block to eliminate Flicker.
Frame Dither consists of that 16 x 16 Table of each R/G/B Dither and could be set by using FDITHERTABLEADDR, FRDITHERVALUE, FGDITHERVALUE and FBDITHERVALUE. The value of Frame Dithered Pixel is shifted to the result of Random Number Generator and output thru PVD. It is selectable whether to use Random Number Generator as Enable or not.

Block Diagram

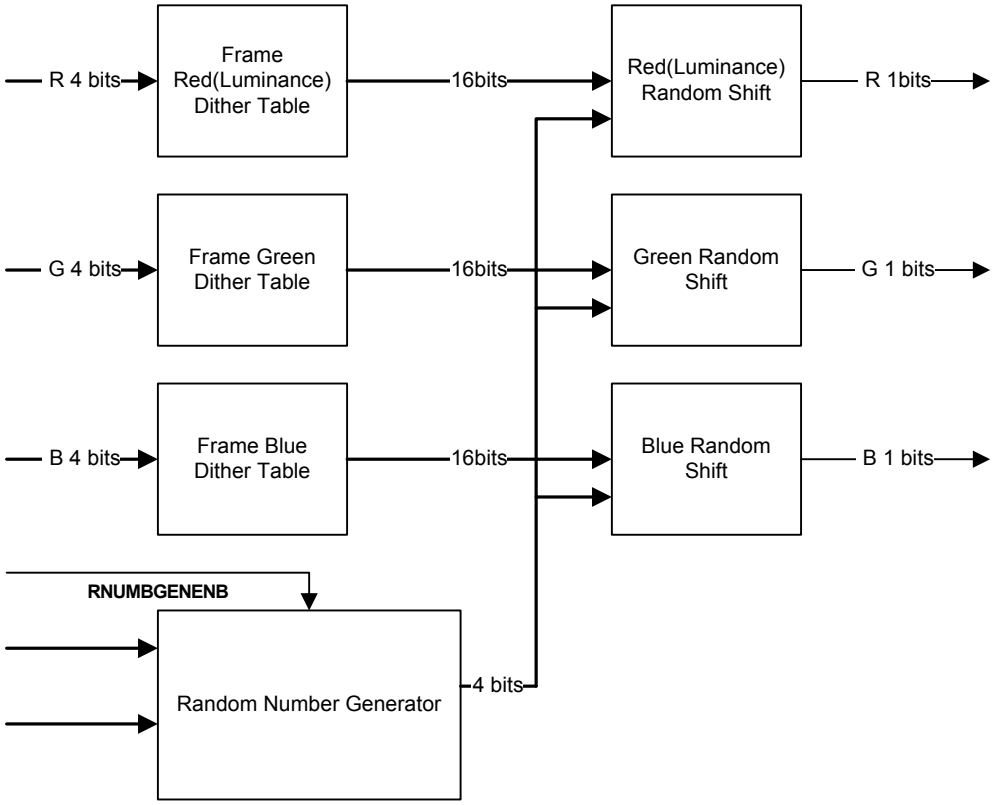


Figure 21-12. Frame Rate Control

Random Number Generator generates value to shift Frame Dither table data to make random number.
Generated shift value has 4bit data and shift Dither table output value by 0~15.

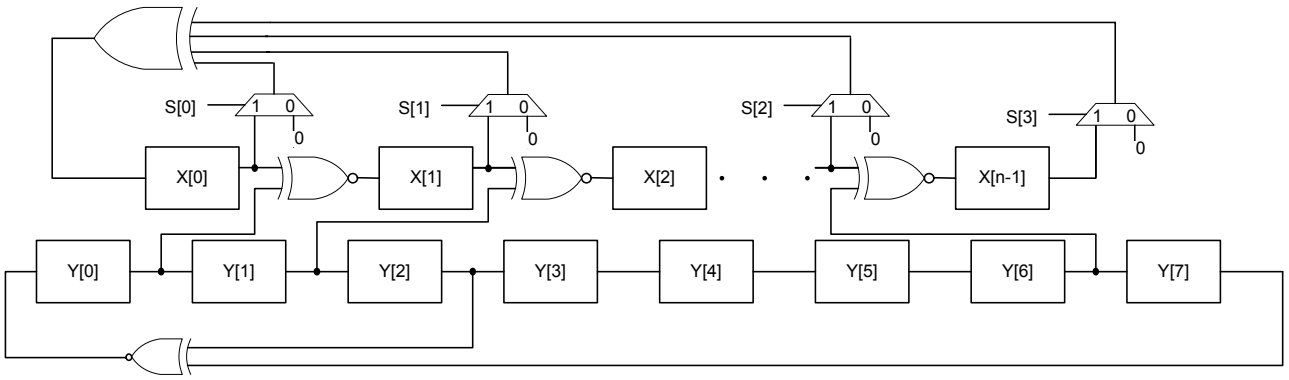


Figure 21-13.. Random Number Generator

n=32, S = {RNCONFormulselH[15:0], RNCONFormulselL[15:0]}
Each X[n-1:0] is initialized to RNCONValueL0[15:0], RNCONValueH0[15:0] when frame starts.
Each Y[7:0] is initialized to RNCONValue1[7:0] when frame starts.

Register simbol	bits	Value
{RNCONVALUEH0, RNCONVALUEL0}	[31:0]	32'heeeeeeee
{RNCONVALUEH0, RNCONVALUEL0}	[31:0]	32'heeeeeeee
RNCONVALUE1	[7:0]	8'hee

Table 21-17. Random number generator value

21.4.4.6. UPSCALER(Only Secondary Display)

Upscaler performs the function of horizontal bilinear filter up scale MLC OUT Layer. It is selectable whether to scale with UPSCALEENB Setting.
UPSCALEENB Setting.
UPSCALE = (MLCScreenWidth-1) * (1<<11) / (destination width-1)

21.4.4.7. Embedded Sync

The ITU-R BT.656 output does not need the separate Sync Signal pin to transmit signals to the outside. The Sync information is transmitted along with data via a data pin. At this time, the Sync information is inserted as a separate code before the start point of the valid data (SAV) and after the end point of the valid data (EAV). The Sync information included in data is as shown in Figure 21-14.

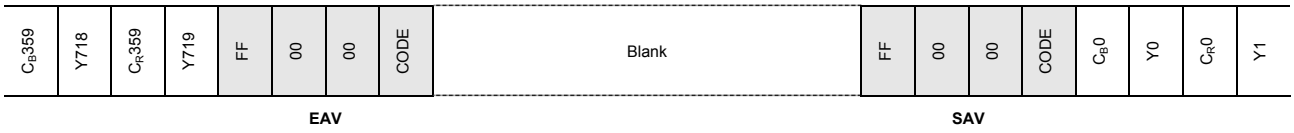


Figure 21-14. Data stream format with SAV/EAV

The SAV and EAV consist of [FF, 00, 00, CODE]. Each of the SAV and EAV codes contains Field(F), VSYNC(V) and HSYNC(H) data and each code is composed as follows:

Bit		7	6	5	4	3	2	1	0	Hex	Brief Description
Function		1	F	V	H	P3	P2	P1	P0		
(FVH)	0	1	0	0	0	0	0	0	0	80h	SAV of odd field
	1	1	0	0	1	1	1	0	1	9Dh	EAV of odd field
	2	1	0	1	0	1	0	1	1	ABh	SAV of odd blank
	3	1	0	1	1	0	1	1	0	B6h	EAV of odd blank
	4	1	1	0	0	0	1	1	1	C7h	SAV of even field
	5	1	1	0	1	1	0	1	0	DAh	EAV of even field
	6	1	1	1	0	1	1	0	0	ECh	SAV of even blank
	7	1	1	1	1	0	0	0	1	F1h	EAV of even blank

▪ F : Field select (0 : odd field, 1 : even field)
▪ V : Vertical blanking (0 : Active, 1 : blank)
▪ H : SAV/EAV (0 : SAV, 1 : EAV)
▪ Parity : P3 = V xor H, P2 = F xor H, P1 = F xor V, P0 = F xor V xor H

Table 21-18. Embedded Sync Code

In the ITU-R BT.656 format, SAV/EAV codes are inserted in data by setting the ***SEAVENB*** as '1'. However, since the SAV/EAV codes are transmitted via a data pin, the range of the data should be restricted, to distinguish the codes from the data. Users can restrict the data range by using ***YCRANGE***. Figure 2-15 shows that the result of the color space conversion changing RGB data to YCbCr data varies depending on YCRANGE.

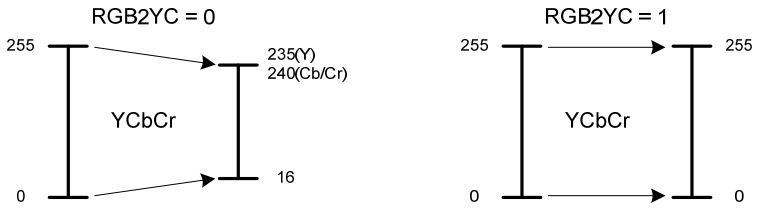


Figure 21-15. Y/C clip

If ***YCRANGE*** is set as '0', a Y value between 0 and 15 is changed to 16 and a Y value between 236 and 255 is changed to 235. In a similar way, Cb/Cr values between 0 and 15 are changed to 16 and Cb/Cr values between 241 and 255 are changed to 240. If ***YCRANGE*** is set as '1', all Y/Cb/Cr values are bypassed.

21.4.5. Scan mode

The sync generator supports interlace display as well as progressive display. The Scan mode is determined by ***SCANMODE.*** If ***SCANMODE*** is '0', the sync generator operates in progressive scan mode. If ***SCANMODE*** is '1', the sync generator operates in interlaced scan mode. In interlaced scan mode, the different vertical timing interfaces can be set in the odd and even fields separately. The odd field display and the progressive display share same registers and registers for the even field display exists separately. Registers to create vertical timing in accordance with the scan mode is listed in Table 21-19.

SCANMODE	Scan mode		Registers
0	Progressive		VTOTAL, VSWIDTH, VASTART, VAEND, VSSOFFSET, VSEOFFSET
1	Interlace	Odd field	
		Even field	EVTOTAL, EVSWIDTH, EVASTART, EVAEND, EVSSOFFSET, EVSEOFFSET

Table 21-19. Registers relative to scan mode

For more detailed descriptions for each register, refer to the section 21.4.4.2.

21.4.6. Delay

The pixel data is more delayed than the final output sync signal due to the processing in the sync generator. Therefore, users should delay Sync signal output to synchronize the Sync signals with the pixel data. When the Sync generator processes data, 4-clock is consumed for RGB data and 6-clock is consumed for YCbCr data. Therefore, for synchronization between Sync signals and data, the output of the Sync signals should be delayed. In RGB format and ITU-R BT.601A format with the same VCLK and VCLK2, delays of 4-clock and 6-clock should be set in each format, separately. In the MGRB and the ITU-R BT.656/601B formats, where the VCLK2 is twice the VCLK, delays of 8-clock and 12-clock should be set in each format, separately.

unit : VCLK2

Format	VCLK2 : VCLK	DELAYRGB	DELAYHS, DELAYVS, DELAYDE
RGB	1 : 1	0	Primary TFT : 7 Primary STN : 8 Secondary : 4
MRGB	2 : 1	0	Primary TFT : 14 Secondary : 8
ITU-R BT.601A	1 : 1	0	6
ITU-R BT.656, ITU-R BT.601B	2 : 1	0	12

Table 21-20. Default delay value

21.4.7. Interrupt

The DPC can generate an interrupt whenever VSYNC occurs.

Interrupt invokes each vertical Sync when TFT LCD Progressive operation. Interrupt invokes each 16 vertical sync when STN LCD

Progressive operation. Interrupt invokes each EVEN field(2 vertical sync) when interlace operation. The DPC sets ***INTPEND*** as '1' if VSYNC occurs and notifies the interrupt generation to the Interrupt controller if the ***INTENB*** is '1'. Therefore, users can acknowledge the generation of VSYNC via polling by using the status of the ***INTPEND*** regardless of the generation of interrupts. The ***INTPEND*** is cleared by writing '1' to it.

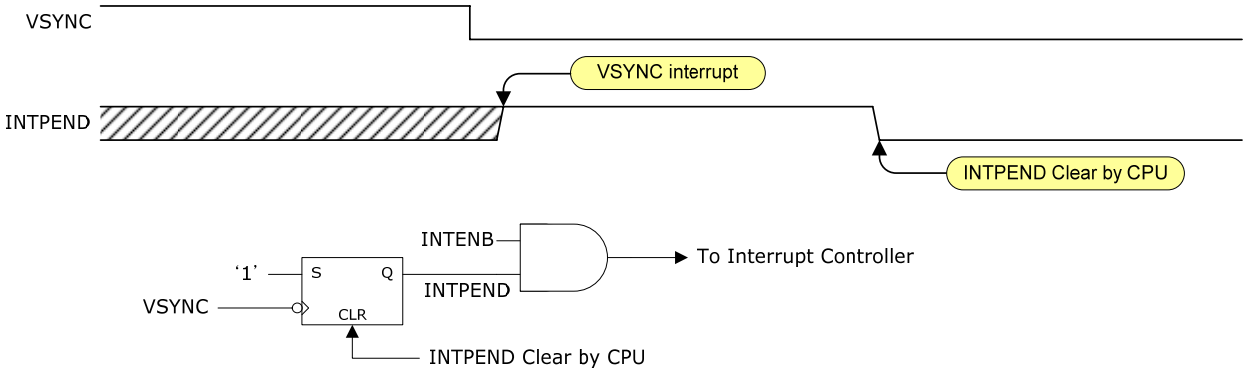


Figure 21-16. Interrupt block diagram and timing

21.5. Internal Video Encoder and DAC

21.5.1. Features

Video Encoder

- Programmable Luma and Chroma bandwidth
- Programmable Saturation, Hue, Contrast and Brightness
- Support all NTSC and PAL formats (NTSC-M/4.43, PAL-B/D/G/H/I/M/N/Combination N)
- Composite output 10 bits to DAC

Video DAC

- Maximum 54 MSPS update rate
- 10 bit current output transmit DAC
- 1.0 Vpp single output range
- 1.2V / 3.3V (thin/thick) 0.13 μm CMOS Process
- Power down mode (high active)

21.5.2. General description

The video encoder is designed to support all standards and variations of the NTSC and PAL encoding systems. This includes cross standards pseudo PAL and pseudo NTSC. The video encoder allows independent control of field rate, chroma subcarrier and the chroma encoding algorithm. Both the luma and chroma bandwidths can be varied to optimize for various data input conditions. The outputs are separated 10 bits for CVBS.

The ***ENCRST*** signal from the secondary sync generator is connected to RESET pin of the internal video encoder. Therefore you have to set ***ENCRST*** bit to access registers of the internal video encoder. And if ***ENCRST*** bit is clear then all registers of internal video encoder will be reset.

The video DAC is a 10 bit single CMOS Digital-to-Analog Converter for video system. Its maximum conversion rate is 54 MSPS. It operates at analog power, 3.0V to 3.6V and provides full scale output currents of 26.7mA at one channel with 37.5 ohm ohmic load for 1.0V. And it is also adaptable to high-speed application such as video system.

The ***ENCRST*** signal from the secondary sync generator is connected to Power Down pin of the internal video DAC. Therefore you have to set ***ENCRST*** bit to output video signal and clear ***ENCRST*** bit to reduce power consumption when it doesn't used.

21.5.3. Video Encoder Reset Sequence

To reset 'Video Encoder', following sequence is needed.

- Write '1' at ENCRST Register.
- Write '1' at CLKGENENB Register.
- Write '0' at ENCRST Register.
- Write '0' at CLKGENENB Register.
- Write '1' at ENCRST Register.

21.5.4. Block diagram

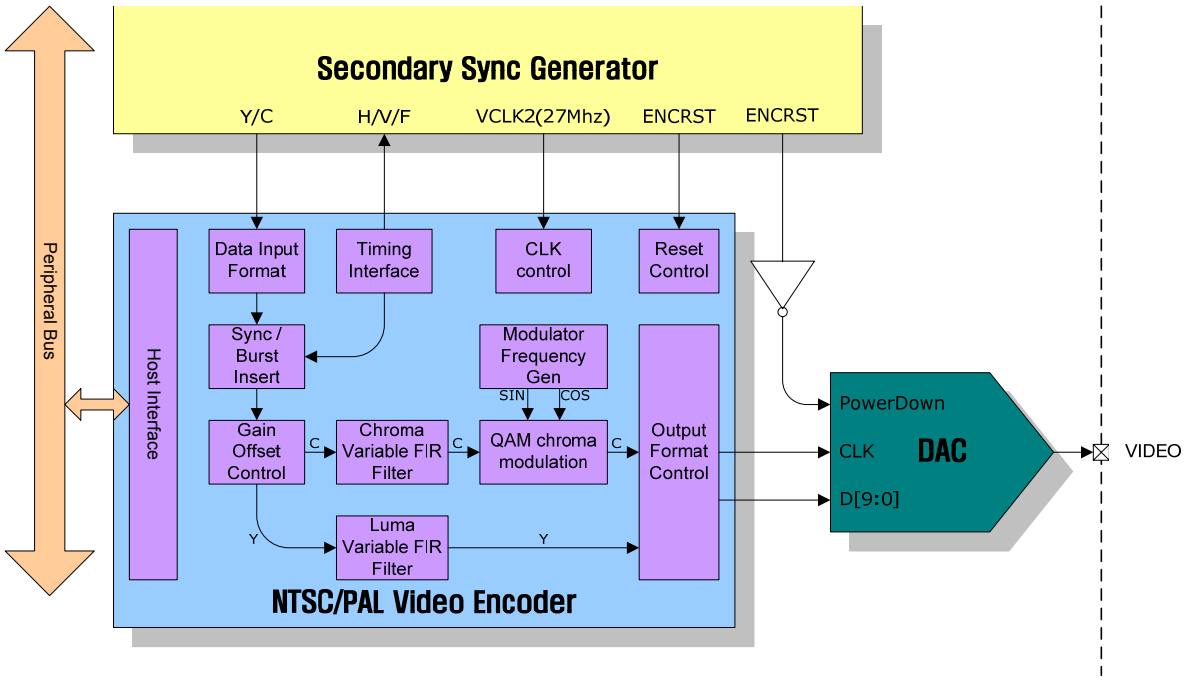


Figure 21-17. Video Encoder and DAC block diagram

21.5.5. Timing interface

The video encoder has 3 output timing signals (HSO, VSO and FSO) which can be used to extract data from a sync generator within the system. If ***ISYNC*** is programmed to '7', then the encoder will "free run" at the horizontal and vertical periods defined by the ***PIXSEL***, ***IFMT*** combination.

The horizontal timing HSO locations are controlled by the ***HSOS*** and ***HSOE*** beginning and ending locations. The programmed value is the pixel number for which the transition will occur. Vertical timing VSO transitions are controlled by the ***VSOS*** and ***VSOE*** register.

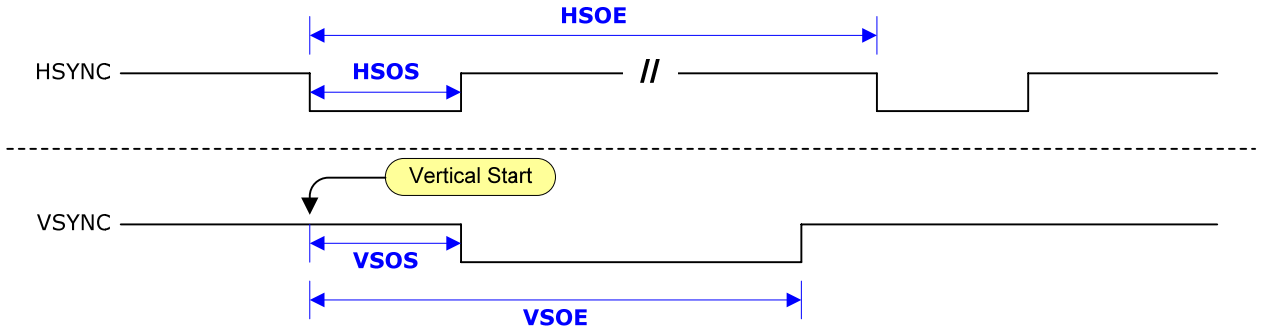


Figure 21-18. Timing inteface

Format	Sync Generator						Video Encoder			
	tHSW	tHBP	tHFP	tVSW	tVBP	tVFP	HSOS	HSOE	VSOS	VSOE
NTSC-M/4.43, PAL-M, Pseudo PAL	32	90	16	3	15	4	63	1715	0	3
NTSC-N, PAL-B/D/G/H/I/N/Combination N, Pseudo NTSC	42	90	12	2	21	3	83	1727	0	2

Table 21-21. Default timing settings

21.5.6. Video Standard Selection

The video encoder supports all NTSC and PAL standards throughout the world. The encoder supports independent control of the Chroma modulation frequency, selection of phase alternating Line encoded chroma and field frequency. In addition, for non SCH locked standards, the Chroma can be allowed to free run using the ***FDRST***. If the ***FSCADJ*** register is set to any value other than '0', it will not be possible to maintain the ***SCH*** relationship. For these setting it is also recommended that the ***FDRST*** bit be set to free run mode.

Requested output format			Required register settings			
Format	Field Rate	FSC	IFMT	FSCSEL	PHALT	FDRST
NTSC-M	59.95 Hz (525)	3.5795454	0	0	0	1
NTSC-N	50 Hz (625)	3.5795454	1	0	0	0
NTSC-4.43	60 Hz (525)	4.43361875	0	1	0	0
PAL-M	59.952 (525)	3.57561189	0	2	1	0
PAL-combination N	50 Hz (625)	3.58205625	1	3	1	0
PAL-B/D/G/H/I/N	50 Hz (625)	4.43361875	1	1	1	1
Pseudo PAL	60 Hz (525)	4.43361875	0	1	1	0
Pseudo NTSC	50 Hz (625)	3.5795454	1	0	0	0

Table 21-22. Settings for Various Output Formats

21.5.7. Basic Video Adjustments

The standard video adjustments for Chroma and Luma are included. Chroma controls include Saturation (***VENCSAT***) and Hue (***VENCHUE***). Luma adjustments include Contrast (***VENCCRT***) and Brightness (***VENCBRT***). An additional ***VENCSCH*** control is included that changes the Chroma subcarrier phase relative to the horizontal sync. Note that this feature operates only when ***FDRST*** = 0.

21.5.8. Programmable Bandwidth

The data bandwidth for the Luma and chroma paths is shown in the following frequency plots. The YBW control allows 3 different settings to optimize the output bandwidth to the receiver. The same applies to chroma bandwidth using the CBW control.

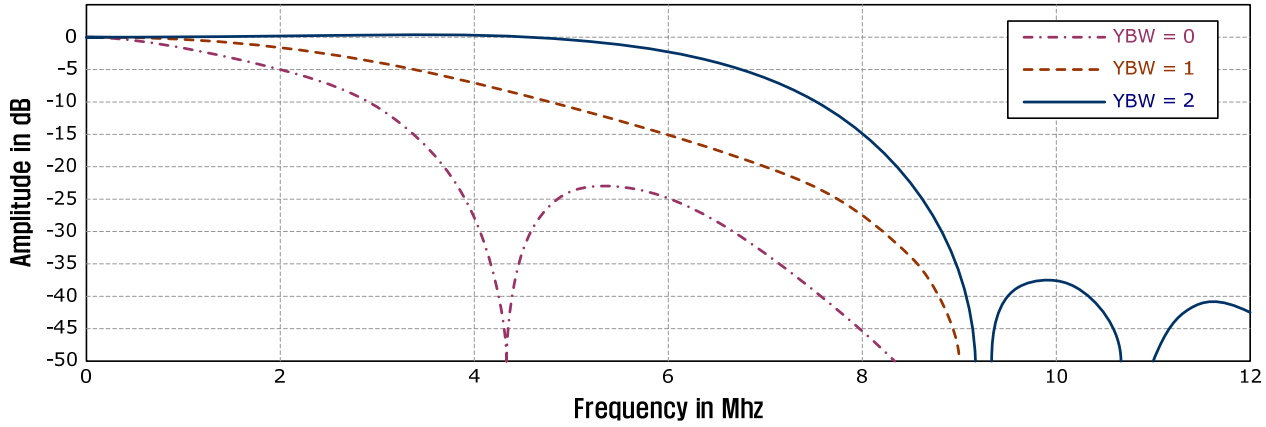


Figure 21-19. Luma Bandwidth

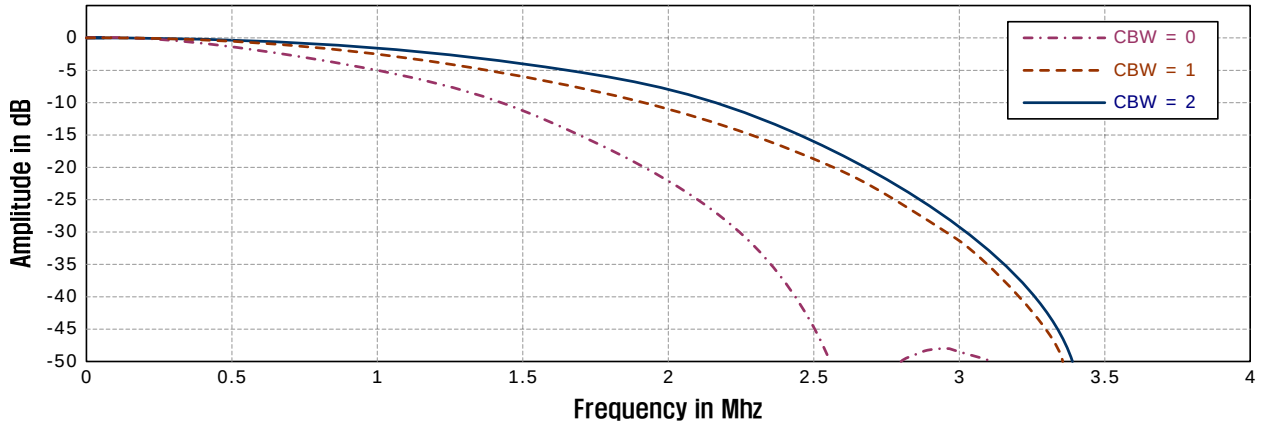


Figure 21-20. Chroma Bandwidth

21.5.9. Analog Video output configuration

The video encoder supports composite video output by the register ***DACSEL***. The DAC output levels and the associated digital codes are summarized below. DAC voltage assumes the standard 140IRE = 1v. Numbers are shown for NTSC type video with a pedestal.

Signal level	CVBS / LUMA Value	IRE Value	DAC Voltage
Max output	1023	105	1.0V
100 % White	982	100	958mV
Black	282	5.66	275mV
Sync	12	-30	11mV
White – Blank	742	100	724mV delta
White - sync	970	130	947mV delta
Color burst	228	30	222mV delta

Table 21-23. Summary of DAC Voltage and Codes

21.6. Register Sumarry

Bit	R/W	Symbol	Description	Reset Value
RESERVED ***Address : Not available(Primary) / C000_3400h(Secondary)***				
[15:0]	-	**RESERVED**	Reserved for future use.	-
VIDEO ENCODER CONTROL A REGISTER (VENCCTRLA) ***Address : Not available / C000_3402h***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7]	R/W	**PWDENC**	Power down mode. 0 : Normal operation 1 : Power down mode for the entire digital logic circuit of encoder digital core	1'b0
[6]	R/W	**FDRST**	SCH Chroma-Luma locking control. 0 : Constant relationship between color burst and horizontal sync maintained for appropriate video standards 1 : Chroma is free running as compared to horizontal sync(recommend)	1'b0
[5:4]	R/W	**FDCSEL**	Modulation frequency of chroma output. 0 : Color subcarrier frequency = 3.57954545 MHz for NTSC 1 : Color subcarrier frequency = 4.43361875 MHz for PAL-B,D,G,H,I,N 2 : Color subcarrier frequency = 3.57561149 MHz for PAL-M 3 : Color subcarrier frequency = 3.58205625 MHz for PAL-combination N	2'b0
[3]	R/W	**PED**	Define input pedestal format 0 : Video output has no pedestal 1 : Video output has a pedestal	1'b0
[3:2]	R/W	**RESERVED**	Reserved for future use. You don't have to write any value except 0.	2'b0
[1]	R/W	**IFMT**	Format of output data. 0 : 525 lines 1 : 625 lines	1'b0
[0]	R/W	**PHALT**	Phase alternate control for PAL encoded chroma signal output. 0 : NTSC encoded color 1 : PAL encoded color	1'b0
VIDEO ENCODER CONTROL B REGISTER (VENCCTRLB) ***Address : Not available / C000_3404h***				
[15:4]	R/W	**RESERVED**	Reserved for future use. You don't have to write any value except 0.	12'b0
[3:2]	R/W	**CBW**	Chroma bandwidth control. 0 : Low bandwidth(default) 1 : Medium bandwidth 2 : High bandwidth 3 : Reserved	2'b0
[1:0]	R/W	**YBW**	Luma bandwidth control. 0 : Low bandwidth(default) 1 : Medium bandwidth 2 : High bandwidth 3 : Reserved	2'b0
RESERVED ***Address : Not available / C000_3406h***				
[15:0]	-	**RESERVED**	Reserved for future use. You don't have to write any value.	-
VIDEO ENCODER SCH PHASE CONTROL REGISTER (VENCSCH) ***Address : Not available / C000_3408h***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**SCH**	Programs the color burst phase relative to the horizontal sync. '0' is the nominal value. Ths 8 bit control covers the entire 360 range as a 2's complement number.	8'b0
VIDEO ENCODER HUE PHASE CONTROL REGISTER (VENCHUE) ***Address : Not available / C000_340Ah***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**HUE**	Programs the active video color burst phase relative to color burst. Ths 8 bit control covers the entire 360 range as a 2's complement number.	8'b0

Bit	R/W	Symbol	Description	Reset Value
VIDEO ENCODER CHROMA SATURATION CONTROL REGISTER (VENCSAT) ***Address : Not available / C000_340Ch***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**SATURATION**	Controls the active video chroma gain relative to the color burst gain. Value is 2's complement with '0' the nominal value..	8'b0
VIDEO ENCODER LUMA GAIN CONTROL REGISTER (VENCCRT) ***Address : Not available / C000_340Ch***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**CONTRAST**	Controls Luma gain. Value is 2's complement with '0' the nominal value. This value is only valid when FDRST is '0'. The allowable constrast value is between -128 and 0.	8'b0
VIDEO ENCODER LUMA OFFSET CONTROL REGISTER (VENCBRT) ***Address : Not available / C000_3410h***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**BRIGHT**	Controls Luma offset. Value is 2's complement with '0' the nominal value. This value is only valid when FDRST is '0'. The allowable brightness value is between 0 and 127.	8'b0
VIDEO ENCODER COLOR BURST FREQUENCY ADJUSTMENT HIGH REGISTER (VENCFSCADJH) ***Address : Not available / C000_3412h***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**FSCADJH**	Allows the pixel clock to be varied by up to +/- 200 ppm of its nominal value. This allows dot crawl adjustment. This FSCADJ[15:0] is multiplied by 4 and added to the internal chroma frequency constant. This value is for FSCADJ[15:8].	8'b0
VIDEO ENCODER COLOR BURST FREQUENCY ADJUSTMENT LOW REGISTER (VENCFSCADJL) ***Address : Not available / C000_3414h***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**FSCADJL**	Allows the pixel clock to be varied by up to +/- 200 ppm of its nominal value. This allows dot crawl adjustment. This FSCADJ[15:0] is multiplied by 4 and added to the internal chroma frequency constant. This value is for FSCADJ[7:0].	8'b0
RESERVED ***Address : Not available / C000_3416h ~ 341Fh***				
[15:0]	-	**RESERVED**	Reserved for future use. You don't have to write any value.	-
VIDEO ENCODER DAC OUTPUT SELECT REGISTER (VENCDACSEL) ***Address : Not available / C000_3420h***				
[15:4]	R/W	**RESERVED**	Reserved for future use. You don't have to write any value except 0.	12'b0
[3:0]	R/W	**DACSEL**	Data type output selection for DAC. 0 : DAC digital output is disabled, Output is code '0' 1 : Data output in CVBS format - Composite Video Others : Reserved for future use	4'b0
RESERVED ***Address : Not available / C000_3422h ~ 343Fh***				
[15:0]	-	**RESERVED**	Reserved for future use. You don't have to write any value.	-
VIDEO ENCODER TIMING CONFIGURATION REGISTER (VENCICNTL) ***Address : Not available / C000_3440h***				
[15:3]	R/W	**RESERVED**	Reserved for future use. You don't have to write any value except 0.	13'b0
[2:0]	R/W	**ISYNC**	Timing configuration for Horizontal, Vertical and Field inputs. 7 : F, V, H Alignment is Free Running(Master mode) Others : Reserved for future use	3'b0
RESERVED ***Address : Not available / C000_3442h ~ 3447h***				
[15:0]	-	**RESERVED**	Reserved for future use. You don't have to write any value.	-

Bit	R/W	Symbol	Description	Reset Value
VIDEO ENCODER HORIZONTAL & VERTICAL SYNC REGISTER (VENCHSVSO) ***Address : Not available / C000_3448h***				
[15:7]	R	**RESERVED**	Reserved for future use.	9'b0
[6]	R/W	**VSOB[8]**	Specifies the number of lines for start of vertical sync output. This signal is not used inside the core. It is provided a reference for master mode timing. This value is for VSOB[8].	1'b0
[5:3]	R/W	**HSOB[10:8]**	Specifies the number of clocks for the horizontal sync width. The HSO signal is not used inside the core. It is provided as a source for master timing. The unit is VCLK2. This value is for HSOB[10:8]. • Horizontal Sync Width = HSOB[10:8] + 1	3'b0
[2:0]	R/W	**HSOE[10:8]**	Specifies the number of total clocks for a horizontal period. The HSO signal is not used inside the core. It is provided as a source for master timing. The unit is VCLK2. This value is for HSOE[10:8]. • Horizontal period = HSOE[10:8] + 1	3'b0
VIDEO ENCODER HORIZONTAL SYNC START REGISTER (VENCHSOS) ***Address : Not available / C000_344Ah***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**HSOS[7:0]**	Specifies the number of clocks for the horizontal sync width. The HSO signal is not used inside the core. It is provided as a source for master timing. The unit is VCLK2. This value is for HSOS[7:0]. • Horizontal Sync Width = HSOS[10:8] + 1	8'b0
VIDEO ENCODER HORIZONTAL SYNC END REGISTER (VENCHSOE) ***Address : Not available / C000_344Ch***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**HSOE[7:0]**	Specifies the number of total clocks for a horizontal period. The HSO signal is not used inside the core. It is provided as a source for master timing. The unit is VCLK2. This value is for HSOE[7:0]. • Horizontal period = HSOE[10:8] + 1	8'b0
VIDEO ENCODER VERTICAL SYNC START REGISTER (VENCVSOS) ***Address : Not available / C000_344Eh***				
[15:8]	R	**RESERVED**	Reserved for future use.	8'b0
[7:0]	R/W	**VSOS[7:0]**	Specifies the number of lines for start of vertical sync output. This signal is not used inside the core. It is provided a reference for master mode timing. This value is for VSOS[7:0].	8'b0
VIDEO ENCODER VERTICAL SYNC END REGISTER (VENCVSOE) ***Address : Not available / C000_3450h***				
[15:5]	R/W	**RESERVED**	Reserved for future use. You don't have to write any value except 0.	11'b0
[4:0]	R/W	**VSOE[4:0]**	Specifies the number of lines for end of vertical sync output. This signal is not used inside the core. It is provided a reference for master mode timing.	5'b0
RESERVED ***Address : Not available / C000_3452h ~ 347Bh***				
[15:0]	-	**RESERVED**	Reserved for future use. You don't have to write any value.	-
DPC HORIZONTAL TOTAL LENGTH REGISTER (DPCHTOTAL) ***Address : C000_307Ch / C000_347Ch***				
[15:0]	R/W	**HTOTAL**	Specifies the number of total VCLK clocks for a horizontal line. TFT or Video Encoder : HTOTAL = tHSW + tHBP + tHFP + tAVW – 1 Color STN : HTOTAL = CL1HW + CL1ToCL2DLY + {(tAVW x 3)/BitWidth}xCPLCYCx2}– 1 Monochrome STN : HTOTAL = CL1HW + CL1ToCL2DLY + {(tAVW x 1)/BitWidth} x CL2CYCx2}– 1	16'b0
DPC HORIZONTAL SYNC WIDTH REGISTER (DPCHSWIDTH) ***Address : C000_307Eh / C000_347Eh***				
[15:0]	R/W	**HSWIDTH**	TFT or Video Encoder : Specifies the number of VCLK clocks for the horizontal sync width. This value must be less than HASTART.	16'b0

Bit	R/W	Symbol	Description	Reset Value
			• HSWIDTH = tHSW – 1 STN : CL1 Hight Width HSWIDTH = CL1HW – 1	

Bit	R/W	Symbol	Description	Reset Value
DPC HORIZONTAL ACTIVE VIDEO START REGISTER (DPCHASTART) ***Address : C000_3080h / C000_3480h***				
[15:0]	R/W	**HASTART**	TFT or Video Encoder : Specifies the number of VCLK clocks from start of the horizontal sync to start of the active video. This value must be less than HAEND. • HASTART = tHSW + tHBP – 1 STN : Delay Cycle from posedge active of CL1 signal to CL2 posedge active. • HASTART = CL1ToCL2DLY	16'b0
DPC HORIZONTAL ACTIVE VIDEO END REGISTER (DPCHAEND) ***Address : C000_3082h / C000_3482h***				
[15:0]	R/W	**HAEND**	TFT or Video Encoder : Specifies the number of VCLK clocks from start of the horizontal sync to end of the active video. This value must be less than HTOTAL. • HAEND = tHSW + tHBP + tAVW – 1 STN : Active Width • HAEND = tAVW – 1	16'b0
DPC VERTICAL TOTAL LENGTH REGISTER (DPCVTOTAL) ***Address : C000_3084h / C000_3484h***				
[15:0]	R/W	**VTOTAL**	Specifies the number of total lines for a frame or field. TFT or Video Encoder : VTOTAL = tVSW + tVBP + tVFP + tAVH – 1 STN : VTOTAL = BLANKLINE + tAVH – 1	16'b0
DPC VERTICAL SYNC WIDTH REGISTER (DPCVSWIDTH) ***Address : C000_3086h / C000_3486h***				
[15:0]	R/W	**VSWIDTH**	TFT or Video Encoder : Specifies the number of lines for the vertical sync width. This value must be less than VASTART. When interlace mode, this value is used for odd field. • VSWIDTH = tVSW – 1 STN : Blank Line Number • VSWIDTH = BLANKLINE – 1	16'b0
DPC VERTICAL ACTIVE VIDEO START REGISTER (DPCVASTART) ***Address : C000_3088h / C000_3488h***				
[15:0]	R/W	**VASTART**	Specifies the number of lines from start of the vertical sync to start of the active video. This value must be less than VAEND. When interlace mode, this value is used for odd field. • VASTART = tVSW + tVBP - 1	16'b0
DPC VERTICAL ACTIVE VIDEO END REGISTER (DPCVAEND) ***Address : C000_308Ah / C000_348Ah***				
[15:0]	R/W	**VAEND**	Specifies the number of lines from start of the vertical sync to end of the active video. This value must be less than VTOTAL. When interlace mode, this value is used for odd field. • VAEND = tVSW + tVBP + tAVH – 1	16'b0
DPC CONTROL 0 REGISTER (DPCCTRL0) ***Address : C000_308Ch / C000_348Ch***				
[15]	R/W	**DPCENB**	Enable/Disable the display controller (DPC). 0 : Disable 1 : Enable	1'b0
[14]	R/W	**ENCMODE**	Primary : Run Primary Display synchronize with Secondary Display when run DualDisplay. 0 : ASync 1 : Sync Secondary : Internal Video Encoder Mode Select 0 : Reserved 1 : Master	1'b0
[13]	R/W	**ENCRST**	This bit controls reset of internal video encoder and power down of internal DAC. Set this bit to '1' when the registers of video encoder is written or read. Set this bit to '1' when CVBS output should be active. Set this bit to '0' when the registers of video encoder need to be reset. This bit is only valid in case of secondary display controller.	1'b0

Bit	R/W	Symbol	Description	Reset Value
			0 : Reset 1 : Active	
[12]	R/W	**RGBMODE**	Specifies the output pixel format. 0 : YCbCr 1 : RGB	1'b0
[11]	R/W	**INTENB**	Enable/Disable the VSYNC interrupt. The VSYNC interrupt will be issued at start of the VSYNC pulse. Therefore an interrupt will be occurred at every frame in progressive mode or at every field in interlace mode. 0 : Disable 1 : Enable	1'b0
[10]	R/W	**INTPEND**	Indicates whether the VSYNC interrupt is pended or not. This bit is always operated regardless of INTENB bit. Read> 0 : Not pended 1 : Pended Write> 0 : No affect 1 : Clear	1'b0
[9]	R/W	**SCANMODE**	Determines whether scan mode is progressive or interlace. 0 : Progressive scan mode 1 : Interlaced scan mode	1'b0
[8]	R/W	**SEAVENB**	Enable/Disable SAV/EAV signal into the data. This is used for ITU-R BT.656 format. 0 : Disable embedded sync 1 : Enable embedded sync	1'b0
[7:4]	R/W	**DELAYRGB**	Specifies the delay for RGB PAD output. The unit is VCLK2. Generally this value has '0' for normal operation. This value is only valid in case of primary display controller.	4'b0
[3]	R	**RESERVED**	Reserved	1'b0
[2]	R/W	**POLFIELD**	Specifies the polarity of the internal field signal. 0 : Normal(Low is odd field) 1 : Inversion(Low is even field)	1'b0
[1]	R/W	**POLVSYNC**	Specifies the polarity of the vertical sync output. This bit is only valid in case of primary display controller. 0 : Low active 1 : High active	1'b0
[0]	R/W	**POLHSYNC**	Specifies the polarity of the horizontal sync output. This bit is only valid in case of primary display controller. 0 : Low active 1 : High active	1'b0
DPC CONTROL 1 REGISTER (DPCCTRL1) ***Address : C000_308Eh / C000_348Eh***				
[15]	R/W	**SWAPRB**	Swap Red and Blue component. This value is only valid when the output is RGB. This bit is only valid in case of primary display controller. 0 : RGB 1 : BGR	1'b0
[14]	R	**RESERVED**	Reserved	1'b0
[13]	R/W	**YCRANGE**	Determines the YUV range for RGB to YUV conversion. Write 0 set of Video Encoder output 0 : Y = 16 ~ 235, Cb/Cr = 16 ~ 240 1 : Y/Cb/Cr = 0 ~ 255	1'b0
[12]	R	**RESERVED**	Reserved	1'b0
[11:8]	R/W	**FORMAT**	Specifies the data output format. This value must be 10 or 13 for secondary display controller. TFT or Video Encoder : 0 : RGB555 1 : RGB565 2 : RGB666 3 : RGB888 4 : MRGB555A 5 : MRGB555B 6 : MRGB565 7 : MRGB666 8 : MRGB888A 9 : MRGB888B 10 : ITU-R BT.656 or 601(8bit) 11 : Reserved 12 : ITU-R BT.601A 13 : ITU-R BT.601B(set YCORDER bit as '1') 14 : Reserved 15 : Reserved STN : 0 : Reserved 1 : 4096 Color 2 : Reserved 3 : 16 Gray Level other : Reserved	4'b0
[7]	R/W	**RESERVED**	Reserved but you have to write '0' only.	1'b0
[6]	R/W	**YCORDER**	Specifies the data output order in case of ITU-R BT. 601B. 0 : Cb Y Cr Y 1 : Cr Y Cb Y	1'b0
[5:4]	R/W	**BDITHER**	Specifies the dithering method of Blue component. This value is only valid in case of primary display controller. 0 : Bypass 1 : 4bit dither 2 : 5bit dither 3 : 6bit dither	2'b0
[3:2]	R/W	**GDITHER**	Specifies the dithering method of Green component. This value is only valid in case of primary display controller. 0 : Bypass 1 : 4bit dither 2 : 5bit dither 3 : 6bit dither	2'b0

Bit	R/W	Symbol	Description	Reset Value
[1:0]	R/W	**RDITHER**	Specifies the dithering method of Red component. This value is only valid in case of primary display controller. 0 : Bypass 1 : 4bit dither 2 : 5bit dither 3 : 6bit dither	2'b0
DPC EVEN FIELD VERTICAL TOTAL LENGTH REGISTER (DPCEVTOTAL) ***Address : C000_3090h / C000_3490h***				
[15:0]	R/W	**EVTOTAL**	Specifies the number of total lines for even field. This register is only used when interlace mode. • EVTOTAL = tEVSW + tEVBP + tEVFP + tEAVH - 1	16'b0
DPC EVEN FIELD VERTICAL SYNC WIDTH REGISTER (DPCEVSWIDTH) ***Address : C000_3092h / C000_3492h***				
[15:0]	R/W	**EVSWIDTH**	Specifies the number of lines for the vertical sync width of even field. This value must be less than EVASTART. This register is only used when interlace mode. • EVSWIDTH = tEVSW - 1	16'b0

Bit	R/W	Symbol	Description	Reset Value
DPC EVEN FIELD VERTICAL ACTIVE VIDEO START REGISTER (DPCEVASTART) ***Address : C000_3094h / C000_3494h***				
[15:0]	R/W	**EVASTART**	Specifies the number of lines from start of the vertical sync to start of the active video when even field. This value must be less than EVAEND. This register is only used when interlace mode. • EVASTART = tEVSW + tEVBP - 1	16'b0
DPC EVEN FIELD VERTICAL ACTIVE VIDEO END REGISTER (DPCEVAEND) ***Address : C000_3096h / C000_3496h***				
[15:0]	R/W	**EVAEND**	Specifies the number of lines from start of the vertical sync to end of the active video when even field. This value must be less than EVTOTAL. This register is only used when interlace mode. • EVAEND = tEVSW + tEVBP + tEAVH - 1	16'b0
DPC CONTROL 2 REGISTER (DPCCTRL2) ***Address : C000_3098h / C000_3498h***				
[15:12]	R/W	**CL2CYC**	Sets STN LCD CL2 (Shift clock) Cycle (Unit : VCLK) **CL2CYC = CL2CYC - 1**	4'b0
[11:10]	R	**RESERVED**	Reserved	2'b0
[9]	R/W	**STNLCDBITWIDTH**	STN LCD Data Bus Bit Width. 0 : Reserved 1 : 8 bit	1'b0
[8]	R/W	**LCDTYPE**	Declares External Display Device Type 0 : TFT or Video Encoder 1 : STN LCD	1'b0
[7:1]	R	**RESERVED**	Reserved	6'b0
[0]	R/W	**PADCLKSEL**	Specifies the PAD output clock. 0 : VCLK 1 : VCLK2	1'b0
DPC VERTICAL SYNC END OFFSET REGISTER (DPCVSEOFFSET) ***Address : C000_309Ah / C000_349Ah***				
[15:0]	R/W	**VSEOFFSET**	Specifies the number of clocks from start of the horizontal sync to end of the vertical sync, where the horizontal sync is last one in the vertical sync. If this value is 0 then end of the vertical sync synchronizes with start of the horizontal sync which is new one in the vertical back porch. This value has to be less than HTOTAL. When interlace mode, this vaule is used for odd field. • If tVSEO is 0 then VSEOFFSET = 0, else VSEOFFSET = HTOTAL - tVSEO	16'b0

Bit	R/W	Symbol	Description	Reset Value
DPC VERTICAL SYNC START OFFSET REGISTER (DPCVSSOFFSET) ***Address : C000_309Ch / C000_349Ch***				
[15:0]	R/W	**VSSOFFSET**	Specifies the number of clocks from start of the horizontal sync to start of the vertical sync, where the horizontal sync is last one in the vertical front porch. If this value is 0 then start of the vertical sync synchronizes with start of the horizontal sync which is new one in the vertical sync. This value has to be less than HTOTAL. When interlace mode, this vaule is used for odd field. • If tVSSO is 0 then VSSOFFSET = 0, else VSSOFFSET = HTOTAL - tVSSO	16'b0
DPC EVEN FIELD VERTICAL SYNC END OFFSET REGISTER (DPCEVSEOFFSET) ***Address : C000_309Eh / C000_349Eh***				
[15:0]	R/W	**EVSEOFFSET**	Specifies the number of clocks from start of the horizontal sync to end of the vertical sync, where the horizontal sync is last one in the vertical sync. If this value is 0 then end of the vertical sync synchronizes with start of the horizontal sync which is new one in the vertical back porch. This value has to be less than HTOTAL and is used for even field. • If tEVSEO is 0 then EVSEOFFSET = 0, else EVSEOFFSET = HTOTAL – tEVSEO	16'b0
DPC EVEN FIELD VERTICAL SYNC START OFFSET REGISTER (DPCEVSSOFFSET) ***Address : C000_30A0h / C000_34A0h***				
[15:0]	R/W	**EVSSOFFSET**	Specifies the number of clocks from start of the horizontal sync to start of the vertical sync, where the horizontal sync is last one in the vertical front porch. If this value is 0 then start of the vertical sync synchronizes with start of the horizontal sync which is new one in the vertical sync. This value has to be less than HTOTAL and is used for even field. • If tEVSSO is 0 then EVSSOFFSET = 0, else EVSSOFFSET = HTOTAL - tEVSSO	16'b0
DPC SYNC DELAY 0 REGISTER (DPCDELAY0) ***Address : C000_30A2h / Not available***				
[15:12]	R	**RESERVED**	Reserved	4'b0
[11:8]	R/W	**DELAYDE**	Specifies delay value of DE (data enable)/CP2 output. This value depends on the output format. The unit is VCLK2.	4'b0
[7:4]	R/W	**DELAYVS**	Specifies delay value of the vertical sync/FRM output. This value depends on the output format. The unit is VCLK2.	4'b0
[3:0]	R/W	**DELAYHS**	Specifies delay value of the horizontal sync/CP1 output. This value depends on the output format. The unit is VCLK2.	4'b0
DPC SYNC UPSCALE CONTROL REGISTER 0(DPUPSCALECON0) ***Address : Not available / C000_34A4h***				
[15:8]	R/W	**UPSCALEL**	Specifies the ratio for the horizontal scale. The formula to calculate this value is as follows: When FILTERENB is 1 and the destination width is wider than the source width: • UPSCALE = (source width-1) * (1<<11) / (destination width-1) **UPSCALEL =** UPSCALE[7:0]	8'b0
[7:1]	R	**RESERVED**	Reserved	7'b0
[0]	R/W	**UPSCALERENB**	Decide whether to enlarge Source Image horizontal width. 0 : Scaler Disable 1 : Scaler Enable	1'b0
DPC SYNC UPSCALE CONTROL REGISTER 1(DPUPSCALECON1) ***Address : Not available / C000_34A6h***				
[15]	R/W	**RESERVED**	Reserved	1'b0
[14:0]	R/W	**UPSCALEH**	Specifies the ratio for the horizontal scale. The formula to calculate this value is as follows: When FILTERENB is 1 and the destination width is wider than the source width: • UPSCALE = (MLCScreenwidth-1) * (1<<11) / (destination width-1) **UPSCALEH =** UPSCALE[22:8]	15'b0
DPC SYNC UPSCALE CONTROL REGISTER 2(DPUPSCALECON2) ***Address : Not available / C000_34A8h***				
[15:0]	R/W	**SOURCEWIDTH**	When using Up Scaler, Source image width (MLCScreenWidth) is set.	16'b0
DPC SYNC RANDOM NUMBER GENERATOR CONTROL REGISTER 0(DPRNUMGENCON0) ***Address : C000_30AAh / Not available***				

Bit	R/W	Symbol	Description	Reset Value
[15:0]	R/W	**RNCONVALUEL0**	Random Number Generator init Shift Value LOW	16'h0
DPC SYNC RANDOM NUMBER GENERATOR CONTROL REGISTER 1(DPRNUMGENCON1) ***Address : C000_30ACh / Not available***				
[15:0]	R/W	**RNCONVALUEH0**	Random Number Generator init Shift Value Higth	16'h0
DPC SYNC RANDOM NUMBER GENERATOR CONTROL REGISTER 2(DPRNUMGENCON2) ***Address : C000_30AEh / Not available***				
[15:13]	R	**RESERVED**	Reserved	3'b0
[12]	R/W	**RNUMBGENENB**	Random Number Generator Operation Enable 0 : Disable 1 : Enable	1'b0
[11:8]	R	**RESERVED**	Reserved	4'b0
[7:0]	R/W	**RNCONVALUE1**	Random Number Generator Sub Data	8'b0
DPC SYNC FRAME DIRHER TABLE ADDRESS REGISTER(DPCFDTADDR) ***Address : C000_30B0h / Not available***				
[15:4]	R	**RESERVED**	Reserved	12'b0
[3:0]	R/W	**FDITHERTABLEADDR**	Frame Dither Table Address	4'b0
DPC SYNC FRAME RED DIRHER TABLE VALUE REGISTER(DPCFRDITHERVALUE) ***Address : C000_30B2h / Not available***				
[15:0]	W	**FRDITHERVALUE**	Frame Red(Luminance) Dither Table Data	16'bx
DPC SYNC FRAME GREEN DIRHER TABLE VALUE REGISTER(DPCFGDITHERVALUE) ***Address : C000_30B4h / Not available***				
[15:0]	W	**FGDITHERVALUE**	Frame Green Dither Table Data	16'bx
DPC SYNC FRAME BLUE DIRHER TABLE VALUE REGISTER(DPCFBDITHERVALUE) ***Address : C000_30B6h / Not available***				
[15:0]	W	**FBDITHERVALUE**	Frame Blue Dither Table Data	16'bx

Bit	R/W	Symbol	Description	Reset Value
RESERVED ***Address : C000_30B8h ~ 31BFh / C000_34B8h ~ 35BFh***				
DPC CLOCK GENERATION ENABLE REGISTER (DPCCLKENB) ***Address : C000_31C0h / C000_35C0h***				
[31:4]	R	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	Specifies PCLK operating mode. 0 : PCLK is only enabled when CPU accesses this module 1 : PCLK is always enabled	1'b0
[2]	R/W	**CLKGENENB**	Enable/Disable to generate a clock. 0 : Disable 1 : Enable	1'b0
[1:0]	R	**RESERVED**	Reserved	2'b0
DPC CLOCK GENERATION CONTROL 0 REGISTER (DPCCLKGEN0) ***Address : C000_31C4h / C000_35C4h***				
[31:16]	R	**RESERVED**	Reserved	16'b0
[15]	R/W	**OUTCLKENB**	Specifies the direction of the PADVCLK pad. You have to set this bit when CLKSRCSEL0/1 is 3 or 4. 0 : Enable(Output) 1 : Reserved	1'b0
[14:12]	R/W	**OUTCLKDELAY0**	Specifies delay value of the clock output. This value is needed to adjust clock skew with respect to data signal. 0 : 0 ns 1 : 0.5 ns 2 : 1.0 ns 3 : 1.5 ns 4 : 2.0 ns 5 : 2.5 ns 6 : 3.0 ns 7 : 3.5 ns	3'b0
[11:10]	R	**RESERVED**	Reserved	2'b0
[9:4]	R/W	**CLKDIV0**	Specifies divider value of the source clock. Divider value = CLKDIV0 + 1	6'b0
[3:1]	R/W	**CLKSRCSEL0**	Specifies the source clock. 0 : PLL0 1 : PLL1 2 : Reserved 3 : Reserved 4 Reserved 5 : XTI 6 : Reserved 7 : Reserved	3'b0
[0]	R/W	**OUTCLKINV0**	Specifies whether to invert the clock output. 0 : Normal (Falling Edge) 1 : Invert (Rising Edge)	1'b0
DPC CLOCK GENERATION CONTROL 1 REGISTER (DPCCLKGEN1) ***Address : C000_31C8h / C000_35C8h***				
[31:15]	R/W	**RESERVED**	Reserved for future use. You don't have to write any value except 0.	17'b0
[14:12]	R/W	**OUTCLKDELAY1**	Specifies delay value of the clock output. This value is needed to adjust clock skew with respect to data signal. 0 : 0 ns 1 : 0.5 ns 2 : 1.0 ns 3 : 1.5 ns 4 : 2.0 ns 5 : 2.5 ns 6 : 3.0 ns 7 : 3.5 ns	3'b0
[11:10]	R	**RESERVED**	Reserved	2'b0
[9:4]	R/W	**CLKDIV1**	Specifies divider value of the source clock. Divider value = CLKDIV1 + 1	6'b0
[3:1]	R/W	**CLKSRCSEL1**	Specifies the source clock. 0 : PLL0 1 : PLL1 2 : Reserved 3 : Reserved 4 : Reserved 5 : XTI 6 : Reserved 7 : CLKGEN0 Output	3'b0
[0]	R/W	**OUTCLKINV1**	Specifies whether to invert the clock output. 0 : Normal (Falling Edge) 1 : Invert (Rising Edge)	1'b0

CHAPTER 22.

3D GRAPHICS ENGINE

22. 3D GRAPHICS ENGINE

22.1. Overview

The 3D core of the POLLUX consists of several functional blocks (sub-modules). 3D Graphic Engine block consistes of command processor, Premitive Processor, GTE/ Clipper, TSE and Rasterizer blocks.

The CPU controls directly all Sub-module[inconsistent with earlier spelling] operations and it can also request the Command processor to control them.

CPU can control the operation of sub modules either directly or through command processor.

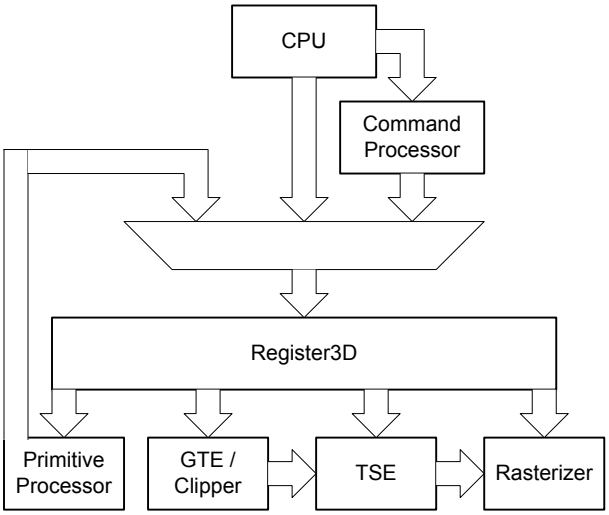


Figure 22-1. 3D Core Control Flow

The picture below shows the data flow for primitive rendering using a vertex buffer. The left side is for hardware vertex transformation/view volume clipping and the right side is for using the Transformed & Lit vertex (in the screen space).

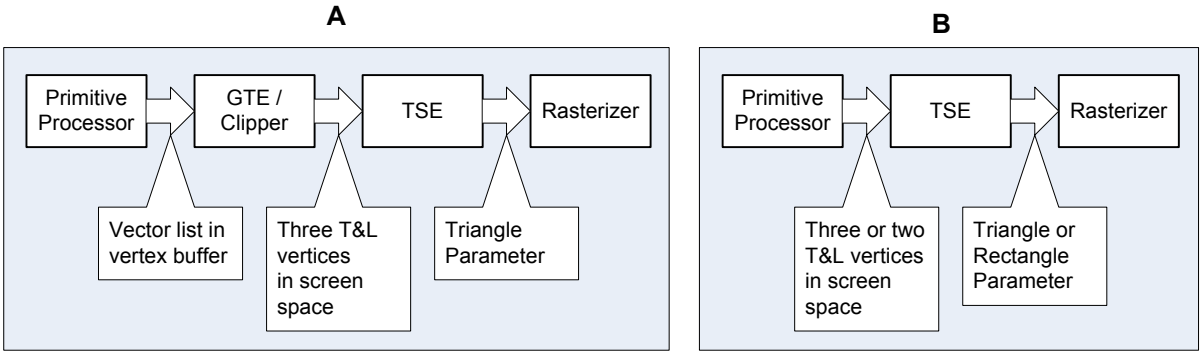


Figure 22-2. 3D Core Data Flow

The sub-modules and related primitive rendering operate with parameters and control signals from the Register3D. Similarly the special purpose DMA Command Processor, which can write to the 3D core register, also operates the 3D core.

22.1.1. Register 3D

The Register3D stores the parameters to control the Sub-Module. It generate operation signals to Sub-module, When external device such as CPU writes a data in control register. Sub-modules and sends operation signals to the Sub-module when an external device writes in the control register (***GRP3D_CONTROL***). The Command Processor and Primitive Processor can control sub-module through the Register3D.

The control priority is Primitive Processor > Command Processor > CPU

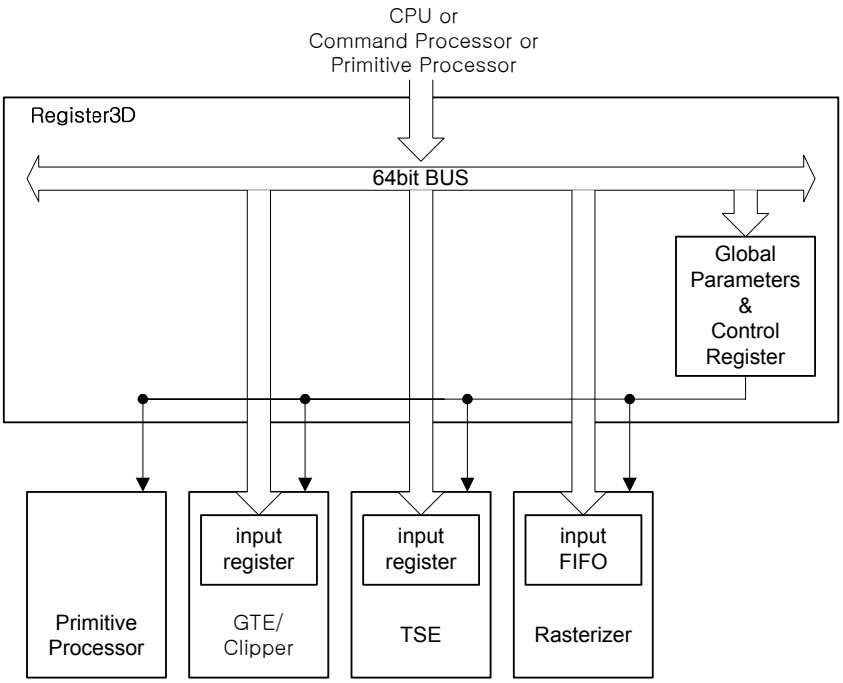


Figure 22-3. Register 3D

22.1.2. Command Processor

The Command Processor reads the Command buffer data and controls the sub-module accordingly. It is a kind of special purpose DMA, which reads data from memory and copies it to the Register3D. It can control

1) Sub-modules,
2) generate an interrupt after a copy operation,
3) it can be a waiting stage (flow control) after checking the sub-module,
4) vertical retrace status before a copy operation.

The Command Processor can operate all 3D operations independently - without the intervention of the CPU.

The Command buffer (or command queue) is a memory area that stores commands. It contains command and other additional information which will be copied into Register 3D block. The CPU assigns the command buffer area (refer to the Command buffer section).

The CPU controls the Command Processor directly, without the Register3D. For Register3D, the Command Processor has priority over the CPU. As shown Figure 22-3, CPU controls command processor directly, But when both CPU and command processor accesses Register 3D, CPU has lower priority than command processor and wait until command processor completes. The CPU cannot access the Register3D when the Command Processor is already accessing it.

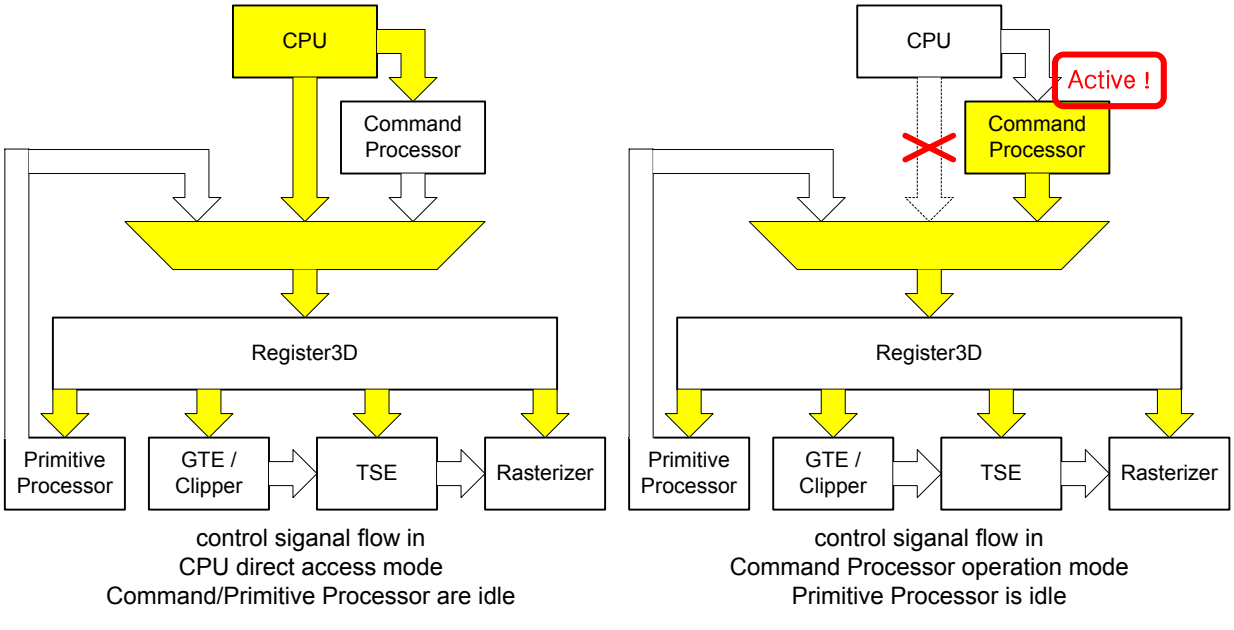


Figure 22-4. The two kinds of control signal flow

22.1.3. Primitive Processor

The Primitive Processor controls the GTE/TSE to render the sequence of geometric objects. The basic operation of the Primitive Processor is loading vertex data from memory to the input register of the GTE (or TSE) and controls the GTE (or TSE). Primitive processor consisits of vertex buffer and index buffer which stores vertex and index data, each. The Vertex buffer is a memory storing vertex data and the Index buffer is a memory storing Index data.
The Primitive Processor has higher priority then CPU and command processor when accessing the Register 3D.

22.1.4. GTE/Clipper (Geometry Transformation Engine/ Triangle Clipper)

The GTE/Clipper makes a triangle from the Vertex data and operates clipping/projection. Once Register 3D loads vector list into GTE input register, the GTE program calculates a vector list and delivers it to the input register of the Clipper.
The GTE is a vector processing unit, which can handle matrix and vector calculation. The GTE also can handle coordinates, color and texture coordinate calculation. The GTE can be used as a floating point Coprocessor because the CPU can read the calculation results.
The Clipper operates clipping and projection of triangle the following the request from the GTE and a requests rendering operation to TSE.

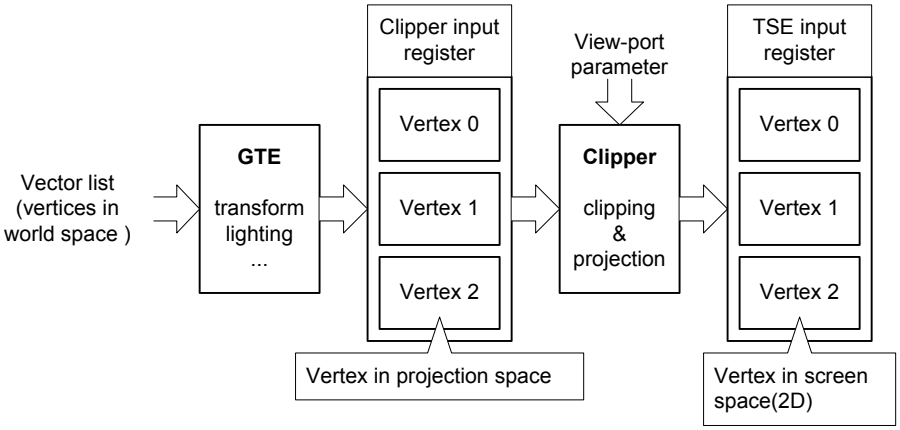


Figure 22-5. GTE/ Clipper Data Flow

22.1.5. TSE (Triangle Setup Engine)

The TSE converts vertex data in the Input Register to fit into the Rasterizer. The vertex data which is stored into three input registers from clipper block for the Rasterizer. The TSE is controlled from the Register3D or GTE/Clipper.

22.1.6. Rasterizer

The Rasterizer has two operations as given below.

- Render triangles or rectangles Rasterizer does two operations.
  Shading / Texture mapping / fog blending / alpha blending / z-buffering /…
- Memory fill
  Fill certain memory areas (rectangles) with certain values.
  Fast z-buffer/screen/texture clear

The Rasterizer is controlled by the Register3D and TSE. The Rasterizer has a 1-depth FIFO to input the next parameters during operations.

22.2. Memory structure for POLLUX

The POLLUX applies Non-linear addressing for some resources for maximum performance. Below is given the terminology used for the POLLUX memory addressing

22.2.1. Segments

Segment is a 16M byte of memory area which has 2 dimensional memory architecture(4096 x 4096 x 8bit). Resource addresses using Non-linear addressing are assigned as a form of segment and offset (U,V).

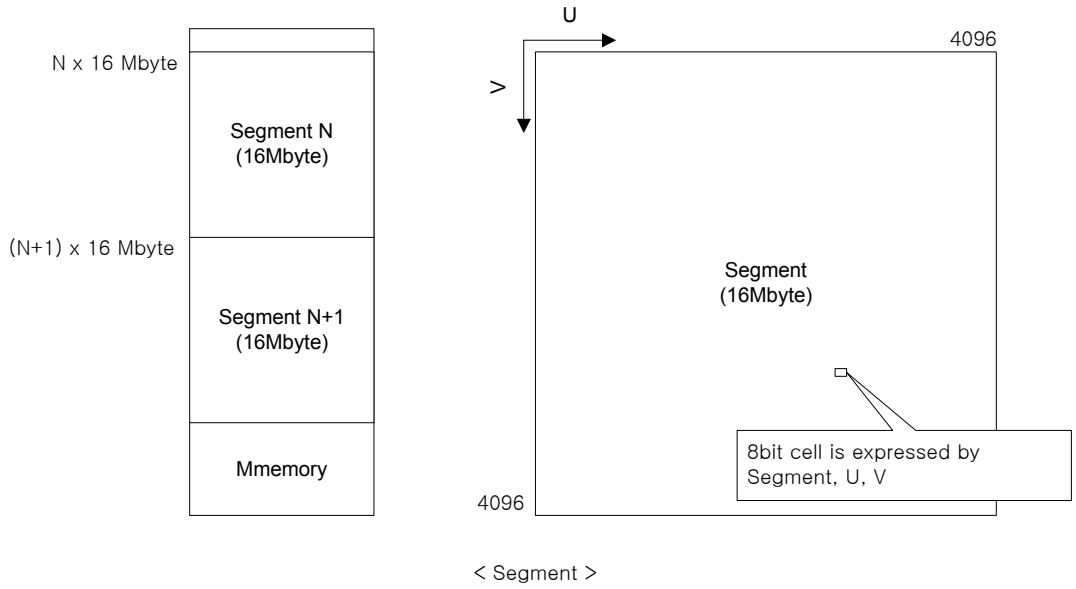


Figure 22-6. Segment

22.2.2. Blocks

To reduce memory latency, the POLLUX uses a Block Memory address instead of a linear address. One Segment consists of 64x128 Blocks and one Block consists of 64x32x8bit (2Kbyte).

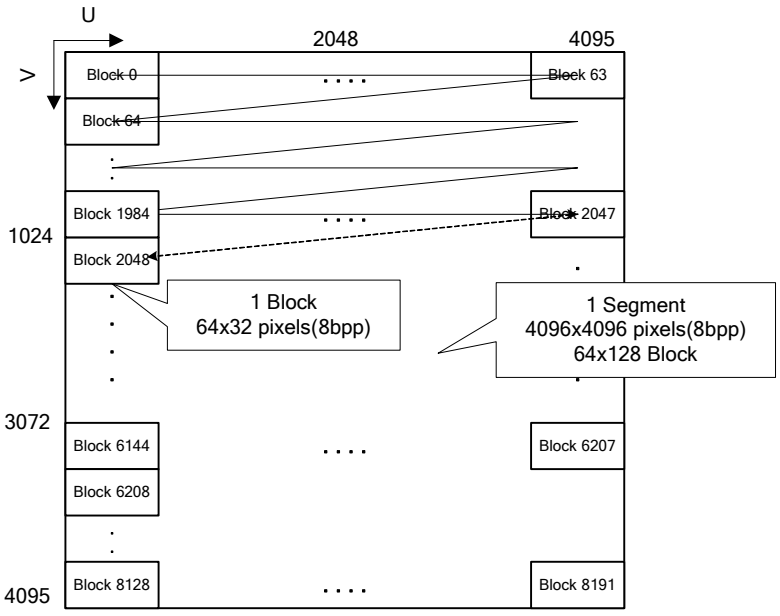


Figure 22-7. 32bit address described in segment and block

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Segment [7:0]								Block index in segment [12:0]													Block address [10:0]											

Table 22-1. 32bit Address Description

22.2.3. Sub-Segments

A Sub-segment is one eighth of the area of a segment.One segment consists of 8 sub-segments and each sub-segment size is 2Mbytes. One sub-segment consists of 1024 blocks.

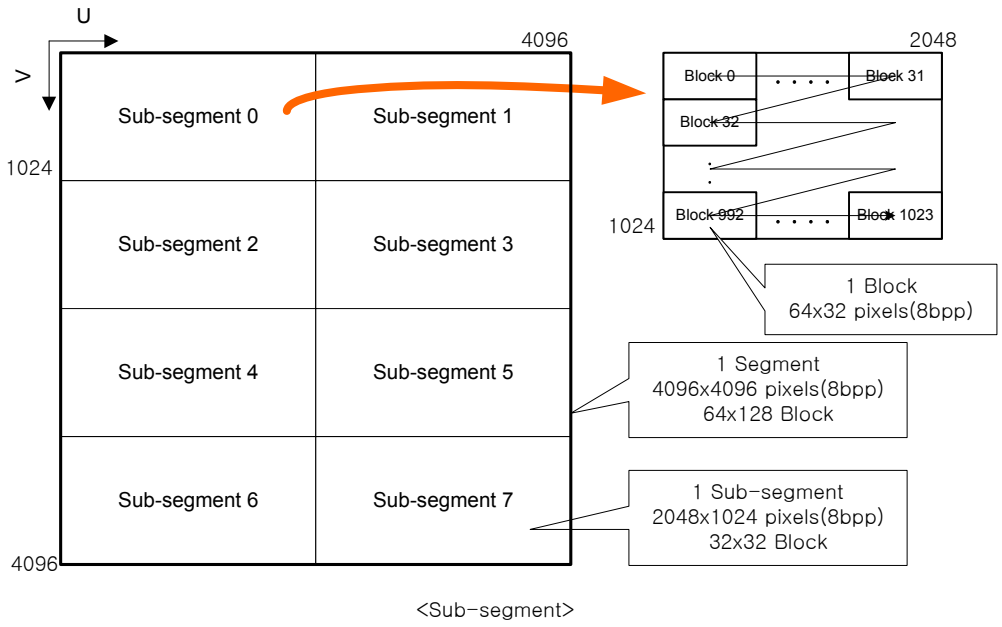


Figure 22-8. Sub-Segment

22.2.4. Block addressing Mode

22.2.4.1. Display mode

Display mode is one of addressing mode for block. It is a linear addressing and displays data to LCD/CRT directly.. The 8bit cell address is calculated as below in Display addressing mode. U,V are 8bpp coordinates in a segment
(0<=U<=4095, 0<=V<=4095)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Segment [7:0]								V[11:5]							U[11:6]						V[4:0]					U[5:0]					
								Block index in segment [12:0]													Block address [10:0]										

Table 22-2. Display Mode Segment

22.2.4.2. Texture mode

Texture mode is one of addressing mode that can be used for Texture/Palette. It divides a Blocks into 16x16 sub-blocks which has 4x2x8bit. An 8bit cell address is calculated as below in Texture addressing mode. U,V are 8bpp coordinates in a segment (0<=U<=4095, 0<=V<=4095)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Segment [7:0]								V[11:5]							U[11:6]						V[4:1]				U[5:2]				V[0]	U[1:0]	
								Block index in segment [12:0]																							

Table 22-3. Texture Mode Segment

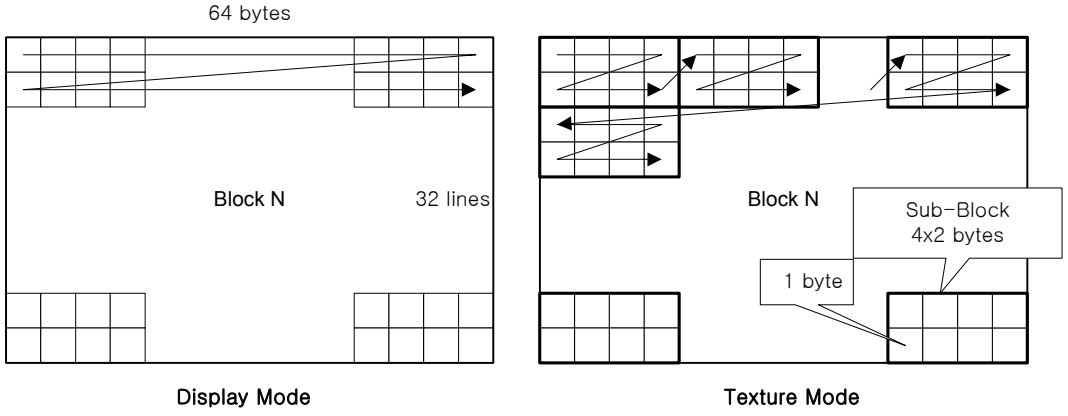


Figure 22-9. Texture Mode

22.2.5. Address converting table

Z-buffer, Display buffer and Render target use sub-segments memory area. To mapping the screen data to sub-segment, it requires address converting operation and 8bit address converting table converts screen address (X, Y) to sub-segment address (U, V) as sollows.

If x[10:9] == 0, U[10:0] = { tablex[1:0] , x[8:0] }
If x[10:9] == 1, U[10:0] = { tablex[3:2] , x[8:0] }
If x[10:9] == 2, U[10:0] = { tablex[5:4] , x[8:0] }
If x[10:9] == 3, U[10:0] = { tablex[7:6] , x[8:0] }
If y[9:8] == 0, V[9:0] = { tabley[1:0] , y[7:0] }
If y[9:8] == 1, V[9:0] = { tabley[3:2] , y[7:0] }
If y[9:8] == 2, V[9:0] = { tabley[5:4] , y[7:0] }
If y[9:8] == 3, V[9:0] = { tabley[7:6] , y[7:0] }

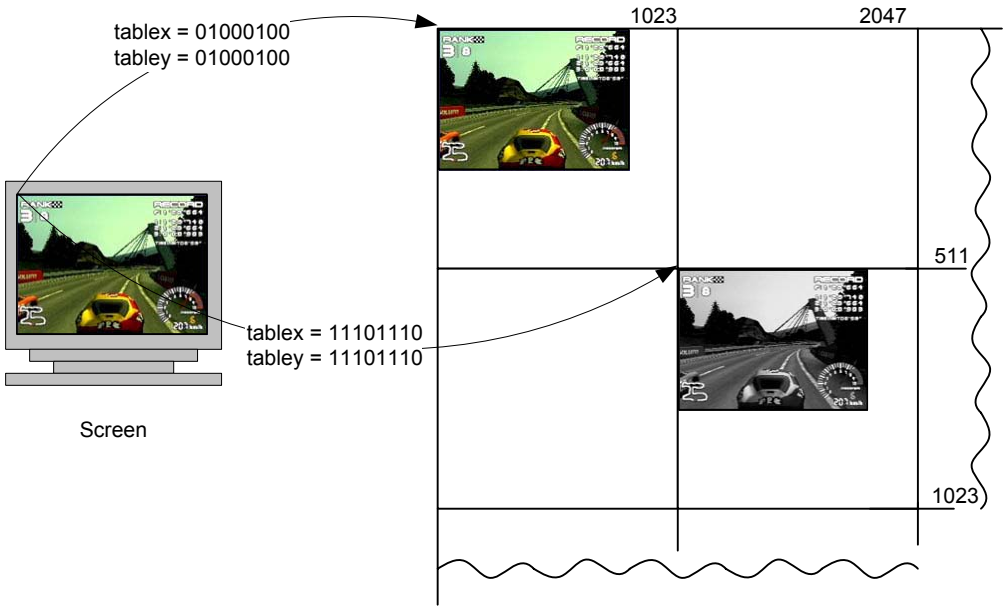


Figure 22-10. Address Converting (Screen to Sub-Segment)

22.2.6. Resources using Non-linear addressing

Block addressing mode	Directly displayed to LCD/CRT	Used as Texture	Resources
Display addressing mode	O	X	Render target Depth buffer
Texture addressing mode	X	O	Render target Texture Palette

Table 22-4. Resources using Non-Linear addressing

22.3. Command Processor

Command Processor can control 3D Graphic core in sted of CPU. Once CPU makes a command list and delivers it to the Command Processor and the Command Processor starts control 3D operation independent of the CPU. The Command Processor has two parts. One is the Command Reader, which reads data from memory, while the other, Command Parser, manages receive data.

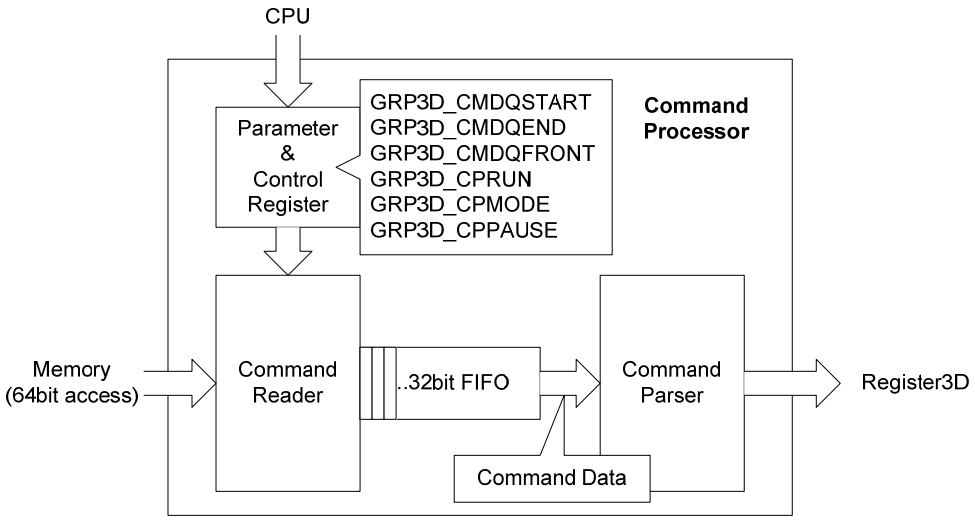


Figure 22-11. Command Processor

22.3.1. Command buffer/queue

Command buffer/ Command Queue is a memory area which contains 32bit command data. 32bit command data will be delivered to command paser. Command buffer and Command queue are named by the Command Processor operation mode (GRP3D_CPMODE). User can define the command buffer of command queue mode by setting the command processor mode register.
The picture below shows the concept and the difference between the Command buffer and Command queue.

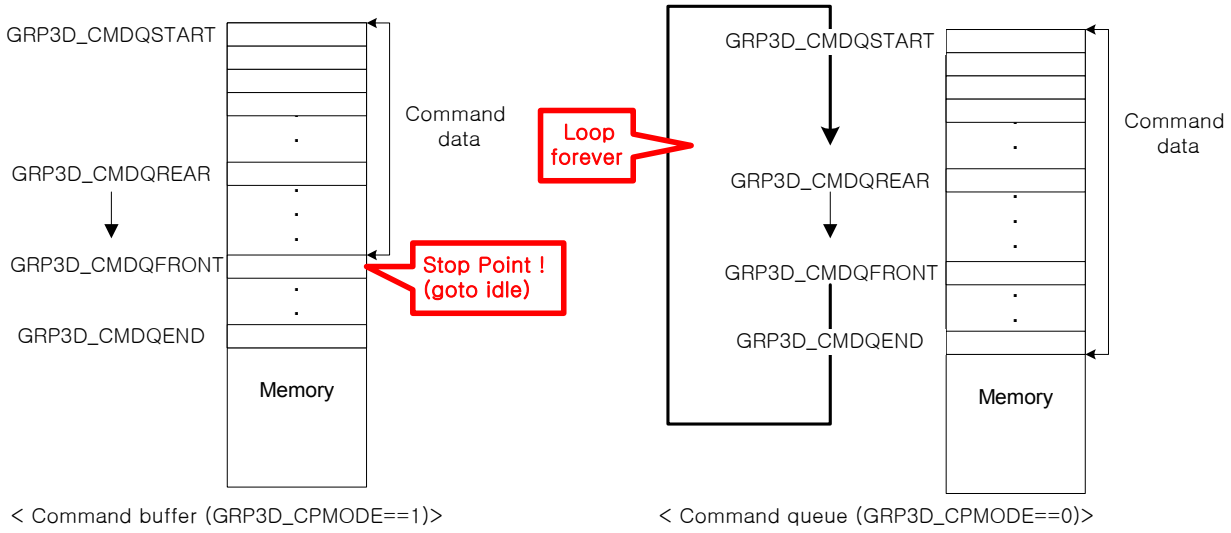


Figure 22-12. Command Processor

There are three registers, – ***GRP3D_CMDQSTART/ GRP3D_CMDQEND*** and ***GRP3D_CMDQFRONT*** which define starting point, end point and end of valid data of command buffer/ queue memory. They must satisfy the conditions below.

$$GRP3D\_CMDQSTART \leq GRP3D\_CMDQFRONT \leq GRP3D\_CMDQEND$$

GRP3D_CMDQREAR is a read only register indicating a read address and automatically is increased by or is initialized by the Command Reader. The basic operation of the Command Reader is reading memory while ***GRP3D_CMDQREAR*** is not equal to ***GRP3D_CMDQFRONT***.

Command Processor Mode (GRP3D_CPMODE)	Description
Command buffer mode (GRP3D_CPMODE==1)	Memory content operates one time and Command Processor stops. Command Reader goes to idle stage after reading memory area, which is assigned by GRP3D_CMDQSTART and GRP3D_CMDQFRONT. (GRP3D_CMDQSTART <= address < GRP3D_CMDQFRONT)

Command Processor Mode (GRP3D_CPMODE)	Description
Command queue mode (GRP3D_CPMODE==0)	Command Reader reads memory area, which is assigned by GRP3D_CMDQSTART and GRP3D_CMDQEND. (GRP3D_CMDQSTART <= address <= GRP3D_CMDQEND) If GRP3D_CMDQREAR is the same as GRP3D_CMDQFRONT, wait for GRP3D_CMDQFRONT to be increased

Table 22-5. Command Processor Mode

22.3.1.1. Command

Command is the basic unit of the Command buffer/queue. Command size is variable and will be defined by the command ID which is 4bit[31:28] of the 1'st command data. Command data may includes command ID, Address and data for Register 3D.
The number of Command is 12 and each Command is distinguished by the first 4bits]

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMDID																															

Table 22-6. Command Format

Name	CMDID	Target register	CMDSIZE (32bit)	Command type
CMDID_CONTROL	4	GRP3D_CONTROL	1	**Optimized command** Write Command into Register3D. This is very efficient compared to the Register fill command (CMDID_ REGFILL) because it does not have an extra header. Optimized commands wait for sub-module idle before writing but do not wait for Vertical retrace(fixed flow control). After writing, it automatically operates the sub-module following Command. CMDID_ LUTFILL : Rasterizer (LUT fill)
CMDID_RENDERSTATE	5	GRP3D_RENDERSTATE	1	
CMDID_ALPHABLEND	6	GRP3D_ALPHABLEND	1	
CMDID_TEXSUBSEGMENT	7	GRP3D_TEXSUBSEGMENT	1	
CMDID_MAPPARAM	8	GRP3D_TPCOLOR GRP3D_MIPMAPBIAS	2	
CMDID_LUTFILL	9	GRP3D_LUTPARAM	1	
CMDID_TEXINFO0	10	GRP3D_TEXINFO0_0 GRP3D_TEXINFO0_1	2	
CMDID_TEXINFO1	11	GRP3D_TEXINFO1_0 GRP3D_TEXINFO1_1	2	
CMDID_TEXBLEND0	12	GRP3D_TEXBLEND0	1	
CMDID_TEXBLEND1	13	GRP3D_TEXBLEND1	1	
CMDID_NOP	**14**		**6**	
CMDID_REGFILL	15	-	Number of 32bit data + 1	**Register fill command** This operates flow control following the Command header, writing the following data into the Register3D and invokes interrupt.s. The Command header includes data to write into the Register3D, register address and flow-control flags for Command processor flow control. (Synchronous page flipping uses this command) All 3D Core operation can be handled with this command, but it might be less efficient for small data because every command has a 32bit header.

Table 22-7. Command ID

22.3.1.2. Register fill command (CMDID_ REGFILL)

It operates flow control to follow the Command header and copies the following command data to the Register3D.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1				DATACOUNT32[8:0]									①	②	③	④	⑤	**REGADDR32[10:0]**										
DATA[0]																															
DATA [1]																															
…																															
DATA [DATACOUNT32]																															

Table 22-8. Register Fill Command

- ***DATACOUNT32[8:0] :*** Number of 32bit data = DATACOUNT32 + 1
- ***① NO_WAIT_RAST :*** No-wait check flag for Rasterizer. If this flag is 0, the Command processor waits for Rasterizer idle.
- ***② NO_WAIT_TSE :*** No-wait check flag for TSE. If this flag is 0, the Command processor waits for TSE idle.
- ***③ NO_WAIT_GTE :*** No-wait check flag for GTE. If this flag is 0, the Command processor waits for GTE idle.
- ***④ WAIT_SYNC :*** Wait check flag for Vertical retrace. If this flag is 1, the Command processor waits for vertical retrace.
- ***⑤ INTERRUPT :*** When GRP3D_REGFILLINTR=1, Request interrupt after copy.
- **REGADDR32[10 : 0]** :
  - Register address = C001_A000h + (REGADDR32 * 4)
  - REGADDR32 must be greater than 7
  - DATA[0] ~ DATA[DATACOUNT32] : 32bit Data
- Command processor operation
  - Wait for sub-module idle (NOWAIT_RAST, NOWAIT_TSE, NOWAIT_GTE)
  - Wait vertical retrace (WAIT_SYNC)
  - Register copy

    for(i=0; i < DATACOUNT32+1; i++) (unsigned long*)(E000_0000h)[REGADDR32 + i] = DATA[i];
  - If (GRP3D_REGFILLINTR) generates an Interrupt request

22.3.1.3. Optimized command

Wait until all of Sub-module is idle stage and then copy command to Register3D

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMDID				DATA[0]																											
DATA[1]																															
…																															

Table 22-9. Optimized Command

- CMDID
  - Command ID
  - 4<= CMDID <=**13**
- DATA[0] ~
  - Data that will be copied to Register3D
- Command processor operation
  - Wait for sub-module idle
  - Register copy

    for(i=0; i < CMDSIZE; i++) (unsigned long*)(E000_0000h)[CMDID*2+ i] = DATA[i];
  - Run sub-module

    if(CMDID == CMDID_ LUTFILL) Rasterizer (LUT fill)

22.3.2. Command Reader

The Command Reader reads Command data from the Command buffer/queue and delivers them to the Command Parser though 32bit FIFO to Command Parser.

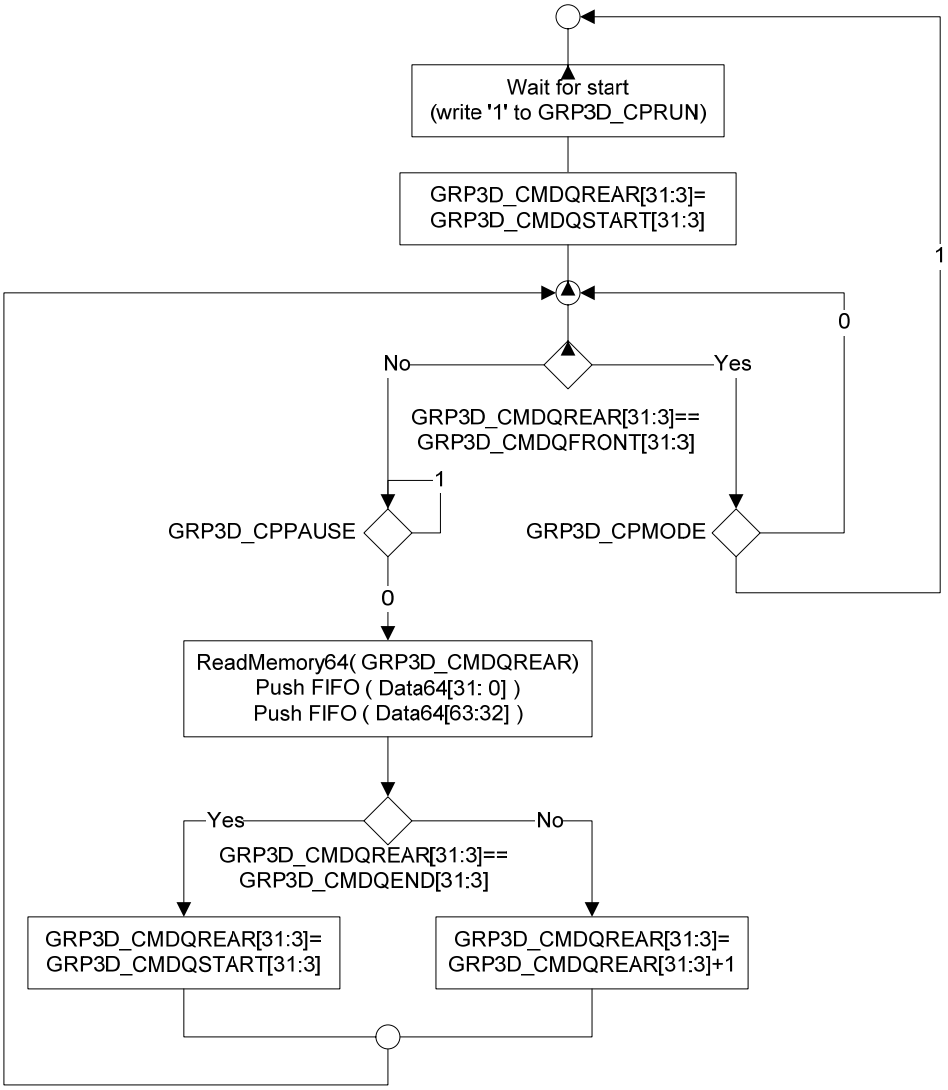


Figure 22-13. Command Reader Operation

The Command Reader reads memory to the point when ***GRP3D_CMDQREAR*** = ***GRP3D_CMDQFRONT*** and will do either (1) waits for the CPU to change the ***GRP3D_CMDQFRONT*** or (2) stops operating. Stage (1) is called Command queue mode and stage (2) is called Command buffer mode. (Command Processor Mode: ***GRP3D_CPMODE***)

22.3.3. Command Parser

After the Command Reader reads 32bit data from the memory, the Command Parser writes 32bit data to the Register3D. The Command Parser operates flow control or sub-modules depending on the conditions.

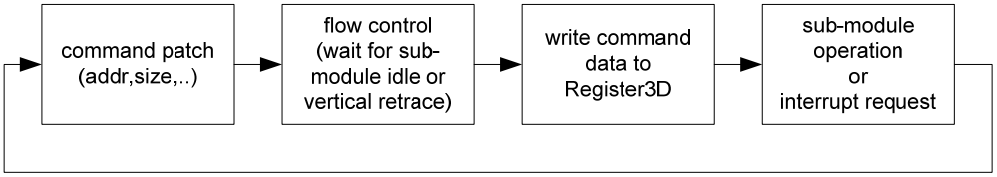


Figure 22-14. Command Parser

Below is given the pseudo-code of the Command Parser operation.

```
while(1)
{
        // command patch
            CMDID = CommandData[31:28];
```

```
        if( CMDID < 4 )
        {
            interrupt ( Invalid Command );
            Core down;
        }

        if( CMDID == 15 )
        {
            start_address32 = CommandData[10:0];
            num_of_data      = CommandData[24:16] + 1;
            wait_Rasterizer = ~ CommandData[15];
            wait_TSE         = ~ CommandData[14];
            wait_GTE         = ~ CommandData[13];
            wait_VSync       = CommandData[12];
            RegfillIntEnb    = CommandData[11];
            Pop FIFO; // get next 32bit data
        }
        else
        {
            start_address32 = CMDID*2;
            num_of_data      = CMDSIZE[ CMDID ];
            wait_Rasterizer = true;
            wait_TSE         = true;
            wait_GTE         = true;
            wait_VSync       = false;
        }

        // flow control
        wait for idle state of Primitive Proccessor
        if( wait_Rasterizer ) { wait for idle state of Rasterizer }
        if( wait_TSE )        { wait for idle state of TSE }
        if( wait_GTE )        { wait for idle state of GTE }
        if( wait_VSync )      { wait for Vertical retrace () }

    // write command data to Register3D
        for( i=0; i < num_of_data ; i++ )
        {
            Register3D_Write32bit (start_address32 + i, CommandData );
            Pop FIFO; // get next 32bit data
        }

        // sub-module operation
        if( CMDID ==  9 ) {  Fill LUT                } // CMDID_LUTFILL

        // Register fill interrupt
        if( CMDID == 15 && RegfillIntEnb ) { interrupt ( Register fill ); }
}
```

22.4. TSE

The TSE Module converts vertex data into a form that the Rasterizer can use.The Rasterizer needs a lot of information such as screen coordinates, vertex color, start value of texture coordinates, etc, to draw triangles or rectangles because Rasterizer can not use vertex data directly.

22.4.1. 24bit Floating Point Number

TSE is 24bit Floating point unit and uses upper 24bit of standard 32bit floating point number.
All floating point numbers used in the POLLUX are 24bit and all kind of floating point register in the POLLUX stores a 24bit floating point number. But application software can write 32bit floating point numbers because the 32bit register area is reserved for a floating point register.

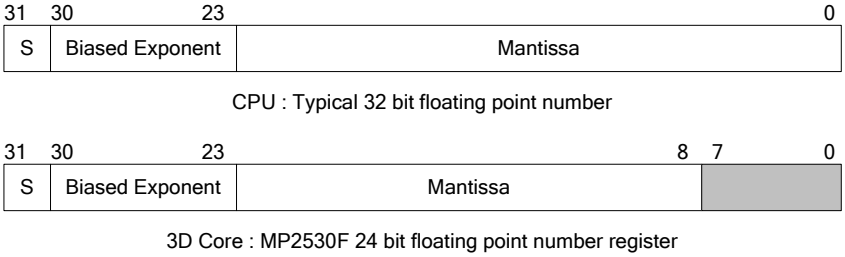


Figure 22-15. 24bit Floating Point format

22.4.2. TSE operation mode

The TSE has two operation modes. One is the triangle setup mode uses three vertexes and the other one is rectangle setup mode uses two vertexes.
TSE operation mode is defined by the TSE control module.

Controlled by	TSE operation mode
CPU or Command Processor	***GRP3DTSEMODE*** register 1: Triangle setup mode 0: Rectangle setup mode
Primitive Processor	***GRP3D_PRIMTYPE*** register 0,1,2: Triangle setup mode 3: Rectangle setup mode (RECTLIST)
GTE/Clipper	Triangle setup mode

Table 22-10. TSE Operation Mode

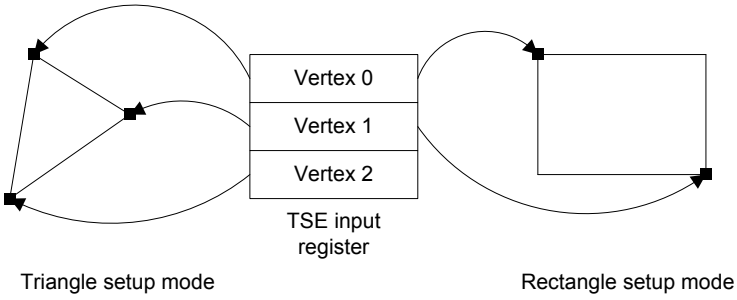


Figure 22-16. TSE Operation Mode

22.4.3. Culling direction and inverse culling flag

Backface culling is affected by the backface culling direction

- 1: CW (Cull backfaces with clockwise vertices)
- 0: CCW

The backface culling direction in GRP3D_RENDERSTATE is a global variable bit. Inverse culling flag is used temporary for changing the culling direction if necessary. For example, when render a triangle strip, the culling direction should be changed once per each triangle.
The Inverse culling flag inverses the GRP3D_CULLDIR effect.

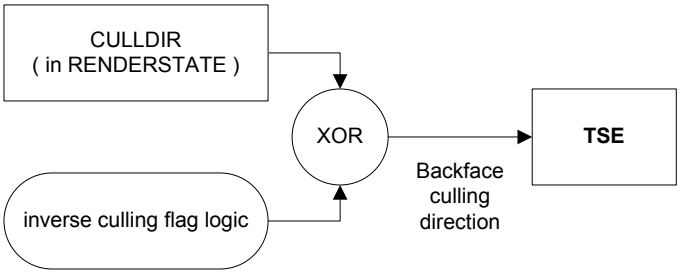


Figure 22-17. Backface Culling Direction

The Inverse culling flag made by the GRP3D_CONTROL register or Primitive processor can be modified by GTE instruction when the GTE is used.

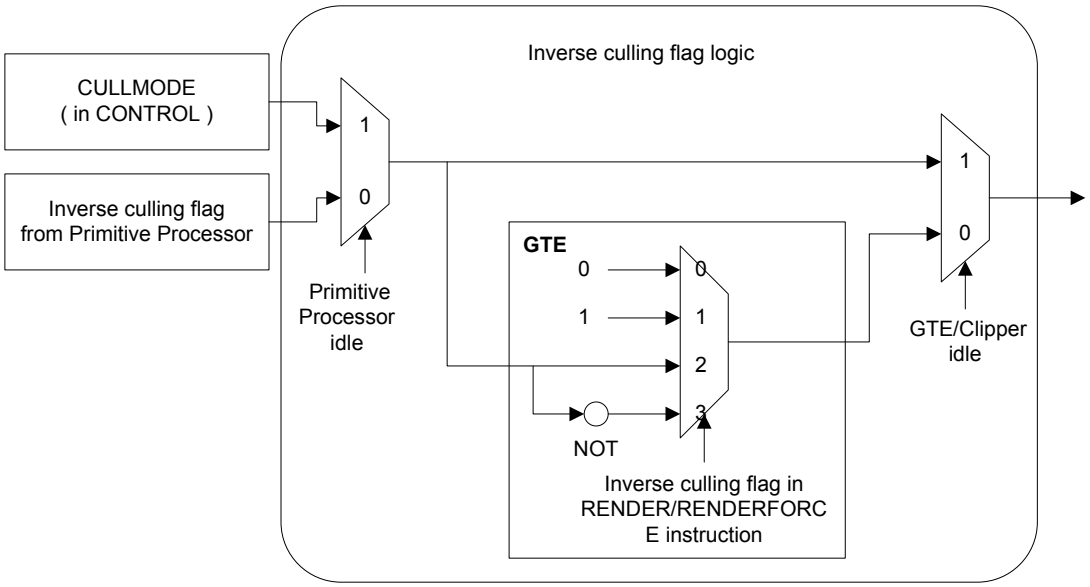


Figure 22-18. Inverse Culling Flag Logic

22.4.4. TSE input vertex format

The TSE has input registers (GRP3D_TSEINPUT) for the three vertices; each vertex consists of twelve floating point numbers.

Name	TSE operation mode	Range
X	The x-coordinate of vertex in screen space	-1023 ~ +1023
Y	The y-coordinate of vertex in screen space	-1023 ~ +1023
Z	The z-coordinate of vertex in screen space,	0 ~ 1
W	Reciprocal homogeneous w	$2^{-16} \sim 2^{12}$
A	The alpha component of vertex color	0 ~ 255
R	The red component of vertex color	0 ~ 255
G	The green component of vertex color	0 ~ 255
B	The blue component of vertex color	0 ~ 255
U0	The u-coordinate of vertex in texture (texture stage 0)	If texture addressing mode is clamp (- Texture size) ~ (Texture size x 2) If texture addressing mode is repeat $-2^{13} \sim 2^{13}$
V0	The v-coordinate of vertex in texture (texture stage 0)	
U1	The u-coordinate of vertex in texture (texture stage 1)	
V1	The v-coordinate of vertex in texture (texture stage 1)	

Table 22-11. TSE Input Vertex Format

22.5. GTE

The GTE module creates or changes vertices or triangles and transfers triangles, which are converted into projection space, to the Clipper.

The GTE is a Vector processing unit, which can handle four floating point numbers at the same time. The general purpose of GTE is multiplying vertices and transforming matrices(world/view/projection) and converting them into projection space and transferring [them?] to the Clipper. GTE is programmable and can manage other operations.

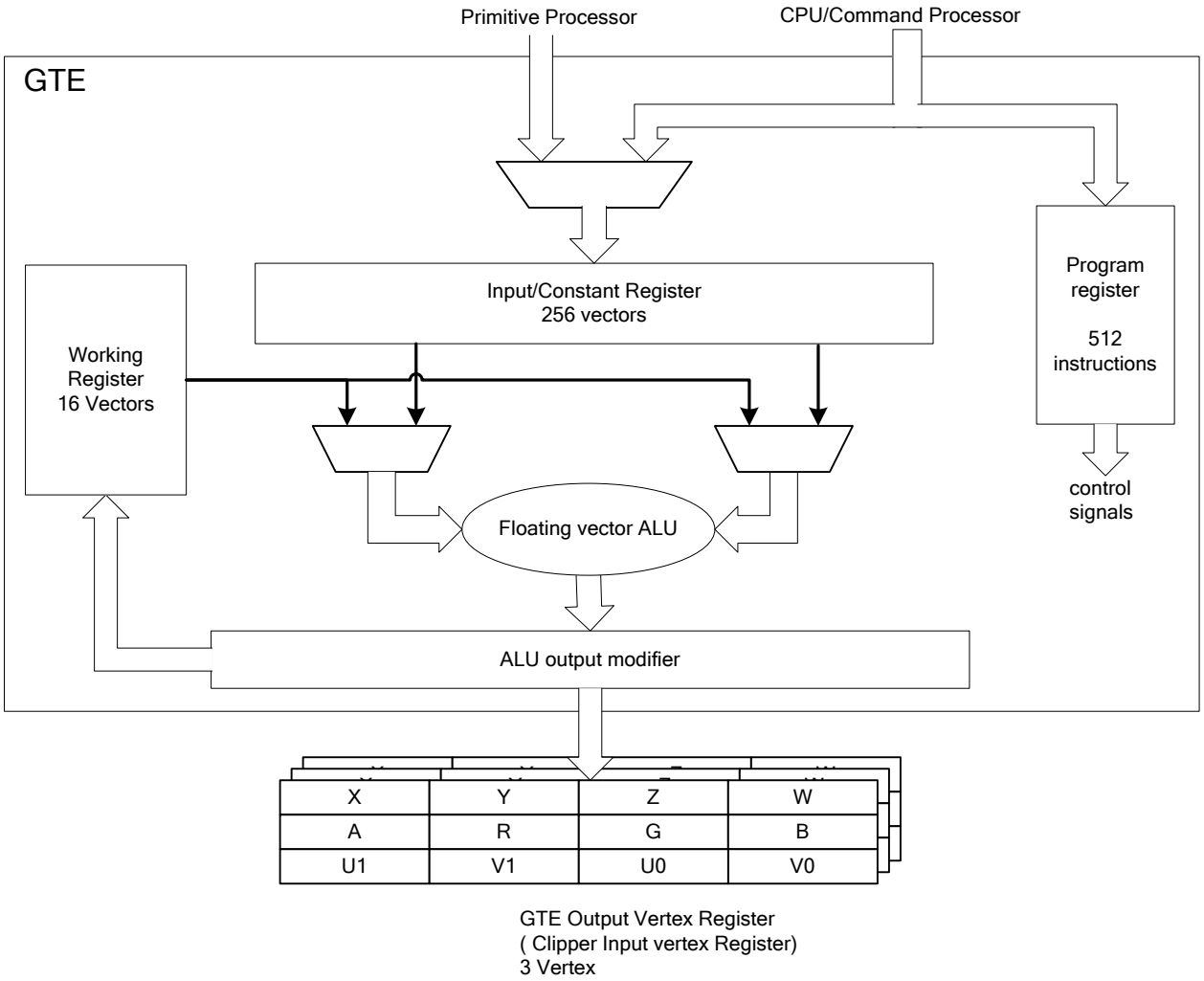


Figure 22-19. GTE Block

The GTE consists of four ALU for adding/multiplying and one divider and registers.

22.5.1. GTE input parameters

The GTE receives three input parameters from the ***GRP3D_CONTROL*** register or Primitive processor.

GTE input parameter	Description	Range
Output vertex number	Assign vertex of Output register. It can be used in GTE instruction or it can be disregarded.	0,1 or 2
Inverse culling flag	Inverse culling flag is transferred to TSE after it is modified by GTE instruction. (RENDER, RENDERFORCE)	0 or 1
Rendering flag	Decide Rendering go/no go of RENDER instruction. RENDER instruction operates Clipper when '1". 0: RENDER is treated as NOP 1: RENDER is treated as RENDERFORCE	0 or 1

Table 22-12. GRP3D_CONTROL Register

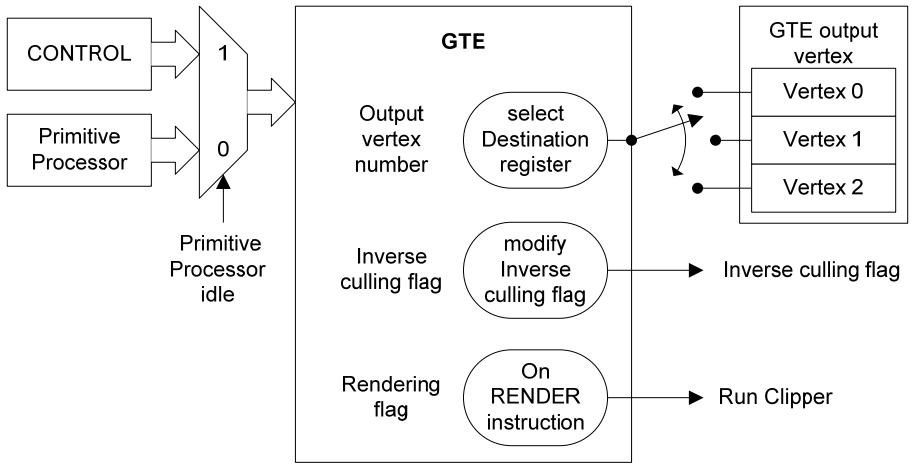


Figure 22-20. GTE Input Parameter

```
// Render triangle-strip
// Load GTE() - Load vertex to GTE input vector register
// Run GTE (output vertex number, inverse culling flag, rendering flag)
Load GTE()
Run GTE (0, x, false) // process vertex 0
Load GTE()
Run GTE (1, x, false) // process vertex 1
Load GTE()
    Run GTE (2, 0, true) // process vertex 2 & rendering
Load GTE()
    Run GTE (0, 1, true) // process vertex 0 & rendering
Load GTE()
    Run GTE (1, 0, true) // process vertex 1 & rendering
…
```

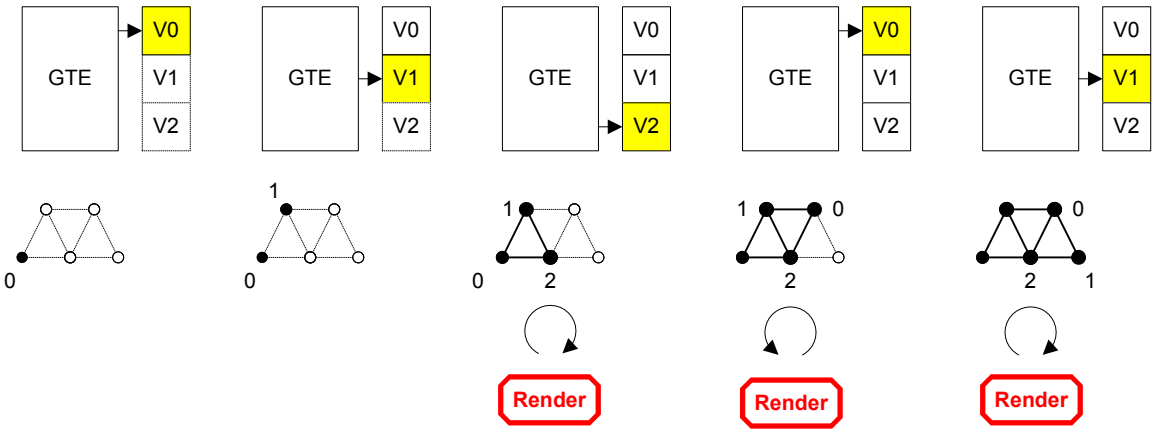


Figure 22-21. Render Triangle Strip

22.5.2. Registers

Register name	R/W (GTE program)	Size	Description
Input/Constant register	**R**	**252 vectors (vector : 4 float)**	**Stores geometric data, transform matrix data and lighting parameters. CPU/Command Processor/Primitive Processor writes new data when GTE operation ends.**
Working register	R/W	16 vectors	Stores interim result sof calculation. It cannot be accessed from outside. It preserves data unless GTE program changes the value.

Register name	R/W (GTE program)	Size	Description
Output register (Clipper input register)	W	3 vertices (vertex: 12 float)	Stores final results of calculation. It can be accessed from CPU only. It preserves data unless GTE program changes the value.
Program register	-	**512 instructions**	Stores GTE program. CPU/Command Processor writes data.

Table 22-13. Registers

22.5.3. GTE Instruction

Categories		Instructions
Arithmetic instructions	Add and multiply	ADD SUB MAD MUL DP3 DP4 DP3CLAMP DP4CLAMP
	Math	RSQRT SETRCP GETRCP
	Move	MOV0 MOV1
	Set On	SET CLR SGE SLT
	Minimum/maximum	MIN MAX
Control instructions	Loop	NOP DELAY EXIT SETLOOP LOOP
	Register paging	**SETPAGE**
	Rendering	RENDER RENDERFORCE

Table 22-14. GTE Instruction

22.5.3.1. Instruction format

GTE instruction is 32bit and each instruction can be distinguished by the top 5bits(instruction ID).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Instruction ID					Instruction body																										

Table 22-15. Instruction Format

22.5.3.2. Instruction cycle

The GTE instruction cycle has three sub-operations.

Sub-operation	Description
I	Instruction fetch (Read instruction)
A	ALU operation
E	Execute instruction (register writing, jump, …)

Table 22-16. Three Sub-Operations

The table below is instruction by PIPEline timing.

Pipeline Group (Pn)	Clock cycles							Instructions
	0	1	2	3	4	5	6	
P7	I	A	A	A	A	A	E	DP4 DP4CLAMP DP3 DP3CLAMP,**RSQRT**
P4	I	A	A	E				ADD SUB MAD SGE SLT MIN MAX
P3	I	A	E					MUL
P2	I	E						MOV0 MOV1 SETRCP
P1	I,E							NOP DELAY EXIT SETLOOP LOOP RENDER RENDERFORCE SET CLR GETRCP,**SETPAGE**

Table 22-17. Instructions by PIPELine Timing

Please observe the 1 rules below when programming the GTE because GTE Instruction has different pipeline timing.

- One destination write (E) for one cycle. (Write crush)

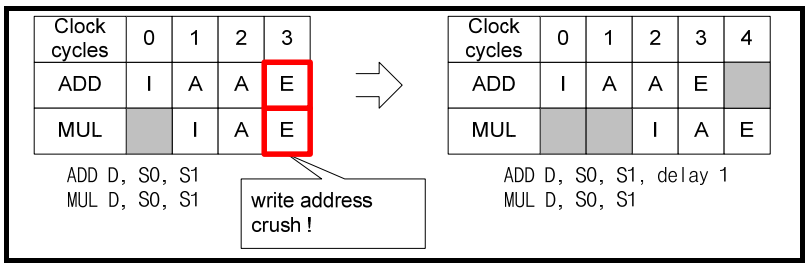


Figure 22-22. Example of GTE Programming Rules

22.5.3.3. Arithmetic instruction

Arithmetic instruction handles and stores the data of the Vector register.

Instruction	Instruction ID	Description	An	Pn
RSQRT	6	Dst = 1.0f / sqrt (| Src0.w |)	A0	P11
DP4	7	Dst = Src0.x * Src1.x + Src0.y * Src1.y + Src0.z * Src1.z + Src0.w * Src1.w	A1	P7
DP3	8	Dst = Src0.x * Src1.x + Src0.y * Src1.y + Src0.z * Src1.z	A2	P7
DP4CLAMP	9	value = Src0.x * Src1.x + Src0.y * Src1.y + Src0.z * Src1.z + Src0.w * Src1.w Dst = (value<0) ? 0: value	A1	P7
DP3CLAMP	10	value = Src0.x * Src1.x + Src0.y * Src1.y + Src0.z * Src1.z Dst = (value<0) ? 0: value	A2	P7
MAD	11	Dst = multiplier output + Src1	A3	P4
ADD	12	Dst = Src0 + Src1	A3	P4
SUB	13	Dst = Src0 - Src1	A3	P4
SGE	14	Dst = (Src0 >= Src1) ? 1.0f : 0.0f	A3	P4
SLT	15	Dst = (Src0 < Src1) ? 1.0f : 0.0f	A3	P4
MAX	16	Dst = (Src0 >= Src1) ? Src0 : Src1	A3	P4
MIN	17	Dst = (Src0 < Src1) ? Src0 : Src1	A3	P4
MUL	18	Dst = Src0 * Src1	A3	P3
SETRCP	19	Start calculation 1.0f / Src0.modifier.x (put Src0.modifier.x to reciprocal module)	-	P2
MOV0	20	Dst = Src0	-	P2
MOV1	21	Dst = Src1	-	P2
GETRCP	22	Dst = reciprocal module output	-	P1
SET	23	Dst = 1.0f	-	P1
CLR	24	Dst = 0.0f	-	P1

Table 22-18. Arithmetic Instruction

- Dst : destination register
- Src0 : the first argument register
- Src1 : the second argument register
- Modifier : refer to Arithmetic instruction format

22.5.3.4. Arithmetic instruction format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Instruction ID					Destination register					Source register 0					Source register 1					Modifier					Write enable				Delay		
					DR	Vector address				SR0	Vector address				SR1	Vector address									W	Z	Y	X			
						OVnum		OVcomp																								

Table 22-19. Arithmetic Instruction Format

- Instruction ID
  - Instruction identifier
- Destination register
  - Assign storing register for calculation results.
  - If DR is 0, store calculation results at Working register.
  - If DR is 1, store calculation results at Output register.
  - If calculation results stored in Output register, Vector address is split into two parts and managed.

Output Vertex number (OVnum)	Description
0	GTE output vertex 0
1	GTE output vertex 1
2	GTE output vertex 2
3	GTE output vertex number from CPU/Command processor/Primitive processor

Table 22-20. Vertex Number

Output Vertex component (OVcomp)	Description
0	Output position vector (x,y,z,w)
1	Output color vector (a,r,g,b)
2	Output texture coordinates (u0,v0,u1,v1)

Table 22-21. Vertex Component

- Source register 0
  - Assign register for Floating ALU source0
  - If SR0 is 0, use Working register as source0.
  - **If SR0 is 1, use Input/Constant register as source0. (Source page0 is used to specify high 4bit of register address)**
- Source register 1
  - Assign register for Floating ALU source1
  - If SR0 is 0, use Working register as source1.
  - **If SR0 is 1, use Input/Constant register as source1. (Source page1 is used to specify high 4bit of register address)**

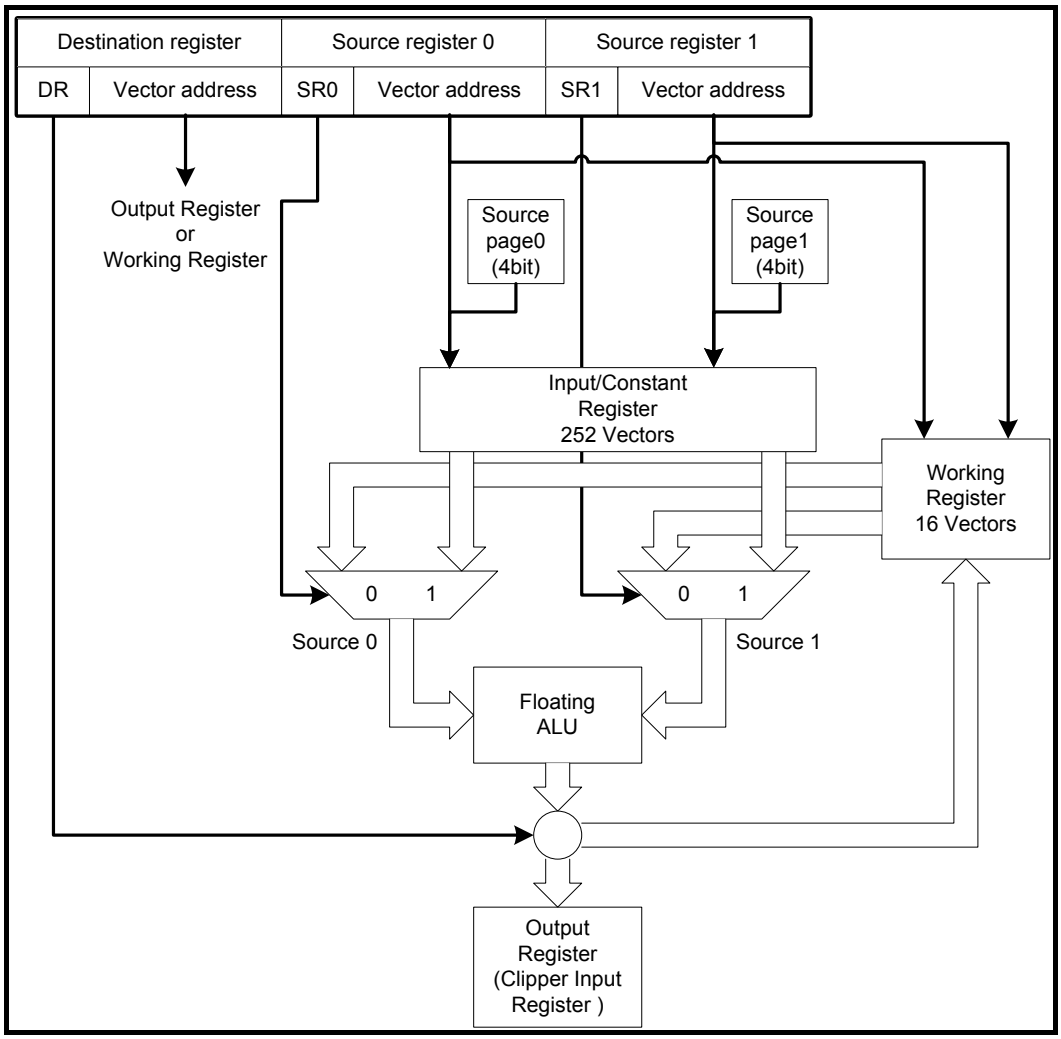
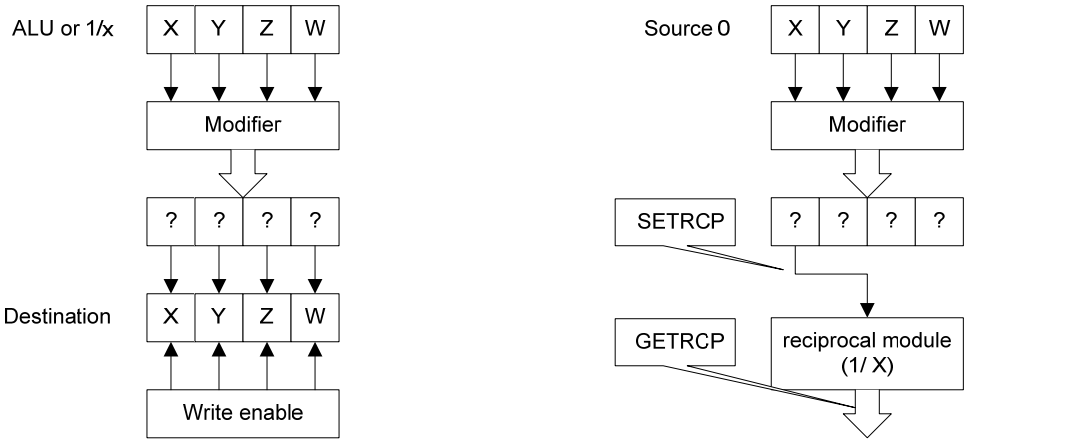


Figure 22-23. Source and Destination in GTE Instruction

- Modifier

Floating ALU output component modifier / Source0 component modifier for SETRCP



< ALU output component modifier & Destination write enable > < Source0 component modifier for SETRCP >

Figure 22-24. Modifier

Modifier	
0	xyzw
1	yzwx
2	zwxy
3	wxyz
4	xywz
5	ywzx
6	zxyw
7	wzxy
8	xzyw
9	ywxz
10	zywx
11	wxzy
12	xwyz
13	yzxw
14	zxwy
15	wyzx
16	xzwy
17	yxzw
18	zwyx
19	wyxz
20	xwzy
21	yxwz
22	zyxw
23	wzyx
24	**xxxx**
25	**yyyy**
26	**zzzz**

Modifier	
27	wwww
28	-x,-x,-x,-x
29	-y,-y,-y,-y
30	-z,-z,-z,-z
31	-w,-w,-w,-w

Table 22-22. Modifier Table

- Write enable

Destination component write enable

If this value = '0', Destination component value is not changed.

- Delay

Clock delay between instruction patch and next instruction patch. If '0', GET patch instruction at every clock.

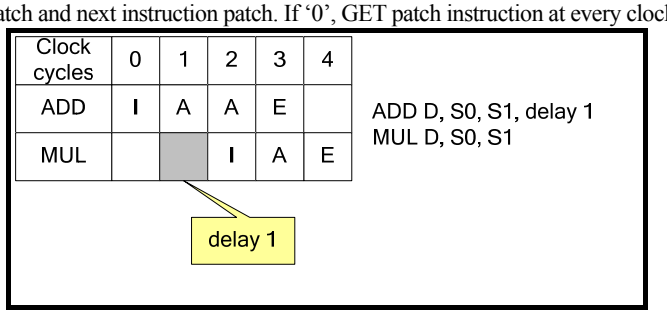


Figure 22-25. Example of Arithmetic Instruction with Delay

22.5.3.5. Control Instructions

Instruction	Instruction ID	Description	An	Pn
NOP	0	No-operation	-	P1
DELAY	25	No-operation with delay	-	P1
SETPAGE	**31**	**Set source page register.**	**-**	**P1**
EXIT	1	Exit GTE program	-	P1
SETLOOP	2	Set loop point & loop count	-	P1
LOOP	3	Jump to loop point as many as SETLOOP assigned.	-	P1
RENDER	4	Operate Clipper to render triangle if Rendering flag = '1'.	-	P1
RENDERFORCE	5	Operate Clipper regardless Rendering flag status.	-	P1

Table 22-23. Control Instructions

22.5.3.6. Control Instruction format & cycles

NOP

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0																															

Table 22-24. Example of NOP (Not Operation)

ADD D, S0, S1
NOP
MUL D, S0, S1

Clock cycles	0	1	2	3	4
ADD	I	A	A	E	
NOP		I,E			
MUL			I	A	E

Figure 22-26. Example of NOP

DELAY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
25																													Delay		

Table 22-25. Example of Delay

Clock cycles	0	1	2	3	4	5	6	7	8	9	10
DP3	I	A	A	A	A	A	E				
DELAY		I,E									
DP4					I	A	A	A	A	A	E

DP3 D, S0, S1
DELAY 2
DP4 D, S0, S1

Figure 22-27. Example of Delay

SETPAGE

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
31																				SRCPAGE1									SRCPAGE0		

Example of SETPAGE

SRCPAGE1: source page 1 register(0~14, reset value is 1)

SRCPAGE0: source page 0 register(0~14, reset value is 0)

EXIT

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1																															

Table 22-26. Example of EXIT

SETLOOP

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
2																									LPID		Loop count				

Table 22-27. Example of SETLOOP

LPID : Loop point ID (0~2)

Loop count : Number of jumps at LOOP instruction. If '0', it does not jump.

LOOP

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
3																									LPID						

Table 22-28. Example of LOOP

LPID : Loop point ID (0~2)

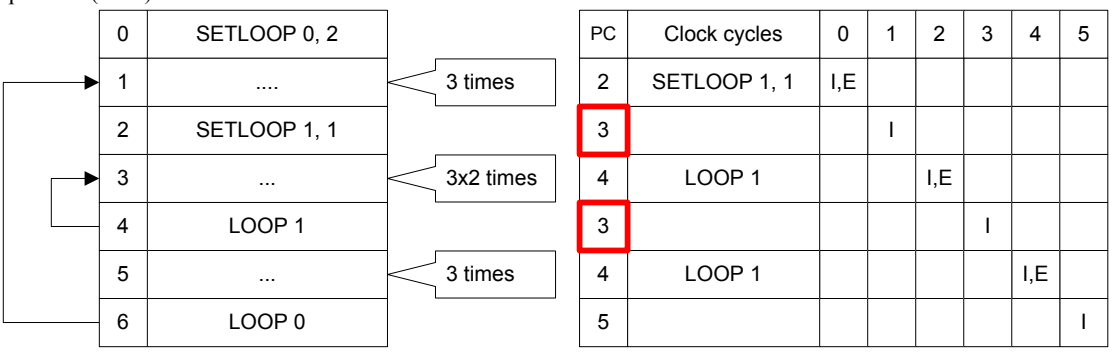


Figure 22-28. Example of SETLOOP/ LOOP

RENDER

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
4																														INVCULL	

Table 22-29. Example of RENDER

INVCULL : inverse culling flag (refer to 'Culling direction and inverse culling flag')

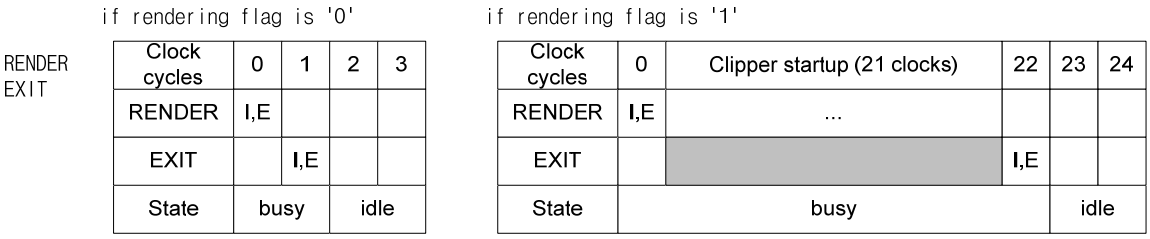


Figure 22-29. Example of RENDER

RENDERFORCE

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
5																														INVCULL	

Table 22-30. Example of RENDERFORCE

INVCULL : inverse culling flag (refer to 'Culling direction and inverse culling flag')

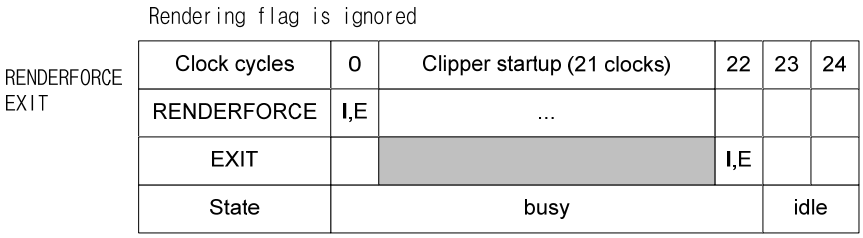


Figure 22-30. Example of RENDERFORCE and EXIT

22.5.3.7. SETRCP/GETRCP

1/x calculation cannot support vector processing because the GTE has one reciprocal module (1/x). Only one floating point number can be put in the reciprocal module (SETRCP) and it needs plenty of time (10clocks) to get a result. (GETRCP)

- SETRCP : Input one of the components (x, y. x, w) of the Source 0 register into a reciprocal module using a modifier.
- GETRCP: Write the reciprocal module output to the Destination register. (Using modifier, write enable)

Reciprocal module outputs reciprocal of input floating point number 10 clocks after SETRCP and holds output before next SETRCP operation. Other calculation can proceed after SETRCP because the reciprocal module operates independently of ALU. If the new SETRCP starts 10 clocks before the previous SETRCP, the previous SETRCP becomes invalid.

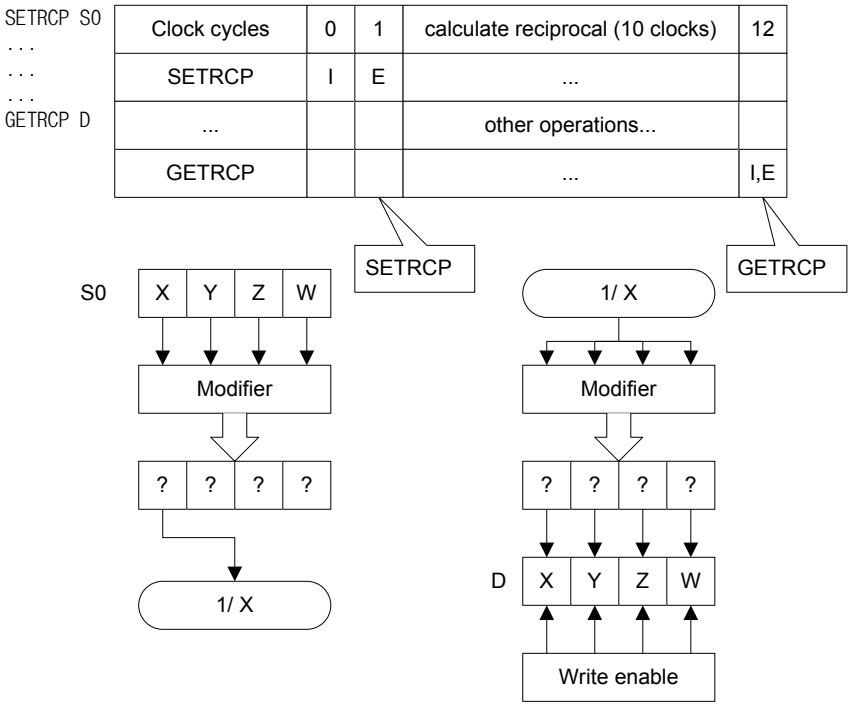


Figure 22-31. Source0 Component Modifier SETRCP

22.5.4. Example

- Simple transform

Output Position = **ICREG[0] * ICREG[16~19]**

Output Color = **ICREG[1]** // a,r,g,b = x,y,z,w

Output Tex = **ICREG[2]** // u0,v0,u1,v1 = x,y,z,w

- Where

ICREG[n] : Input/Constant vector register n

WREG[n] : Working vector register n

IOREG[n] : Output vertex (vertex number n , if n is 3, it is treated as an output vertex number from the GTE input parameter)

- Initial state

ICREG[0] : vertex position

ICREG[1] : vertex color

ICREG[2] : texture coordinates

ICREG[16~19] : total matrix (world x view x projection x clipping)

Description	Instruction code
SETPAGE 0, 1	0xF8000001
MOV0 OREG[3].xyzw, ICREG[1]	0x A7620078
DP4 OREG[3].x___, ICREG[0], ICREG[16]	0x 3F210008
DP4 OREG[3]._y__, ICREG[0], ICREG[17]	0x 3F211010
DP4 OREG[3].__z_, ICREG[0], ICREG[18]	0x 3F212020
DP4 OREG[3]. ___w, ICREG[0], ICREG[19]	0x 3F213040
MOV0 OREG[3], xyzw, ICREG[2], delay(4)	0x A7A4007C
RENDER	0x 20000002
EXIT	0x 08000000

Table 22-31. Example

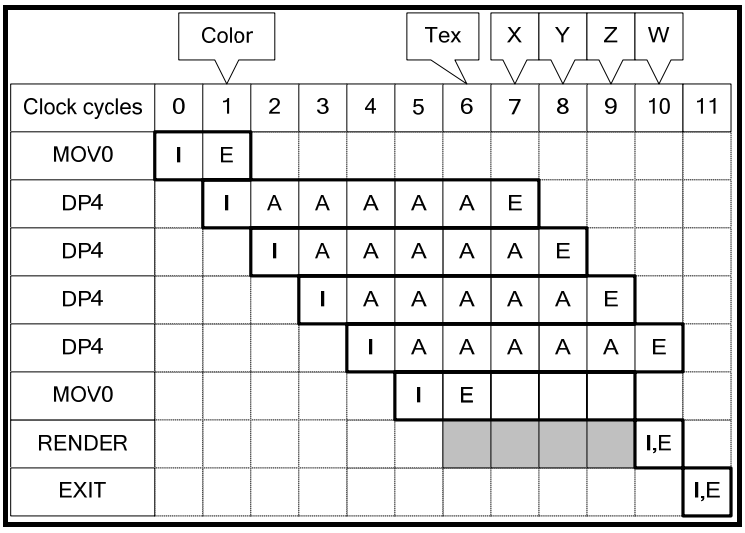


Figure 22-32. Pipeline Timing of Sample Transform Example

22.6. Clipper

The Clipper converts triangles in the projection space (GTE output) into the screen space (TSE input).

The Clipper gets a triangle from the GTE. If it is totally clipped, the Clipper drops it. If it is completely inside the clipping volume, The Clipper transfers it to the TSE in the raw. If it ranges over the clipping volume, The Clipper clips the triangle and transfers the clipped triangles to the TSE

Step	Description
1. Triangle Clipping	Create triangle fan by clipping triangle of Clipper Input vertex register the clipping volume for all points (Xc, Yc, Zc, Wc) in projection space is defined as: Xc : 0 ~ Wc Yc : 0 ~ Wc Zc : 0 ~ Wc
2. Viewport Move & Scaling	$Mvs = \begin{pmatrix} W & 0 & 0 & 0 \\ 0 & -H & 0 & 0 \\ 0 & 0 & 1 & 0 \\ X & Y+H & 0 & 1 \end{pmatrix}$ The viewport scale matrix (Mvs) scales coordinates to be within the viewport window and flips the y-axis from up to down Where: X, Y : Pixel coordinates of the upper-left corner of the viewport on the render target W, H : Dimensions of the viewport on the render target surface, in pixels
3. Projection Divide	Finally, screen coordinates are computed and passed to the TSE Xs = Xvp / Wvp Ys = Yvp / Wvp Zs = Zvp / Wvp Ws = 1 / Wvp Where: (Xvp,Yvp,Zvp,Wvp) = (Xc,Yc,Zc,Wc) x Mvs (Xs,Ys,Zs,Ws) : vertex coordinates in screen space (for TSE)

Table 22-32. Clipper

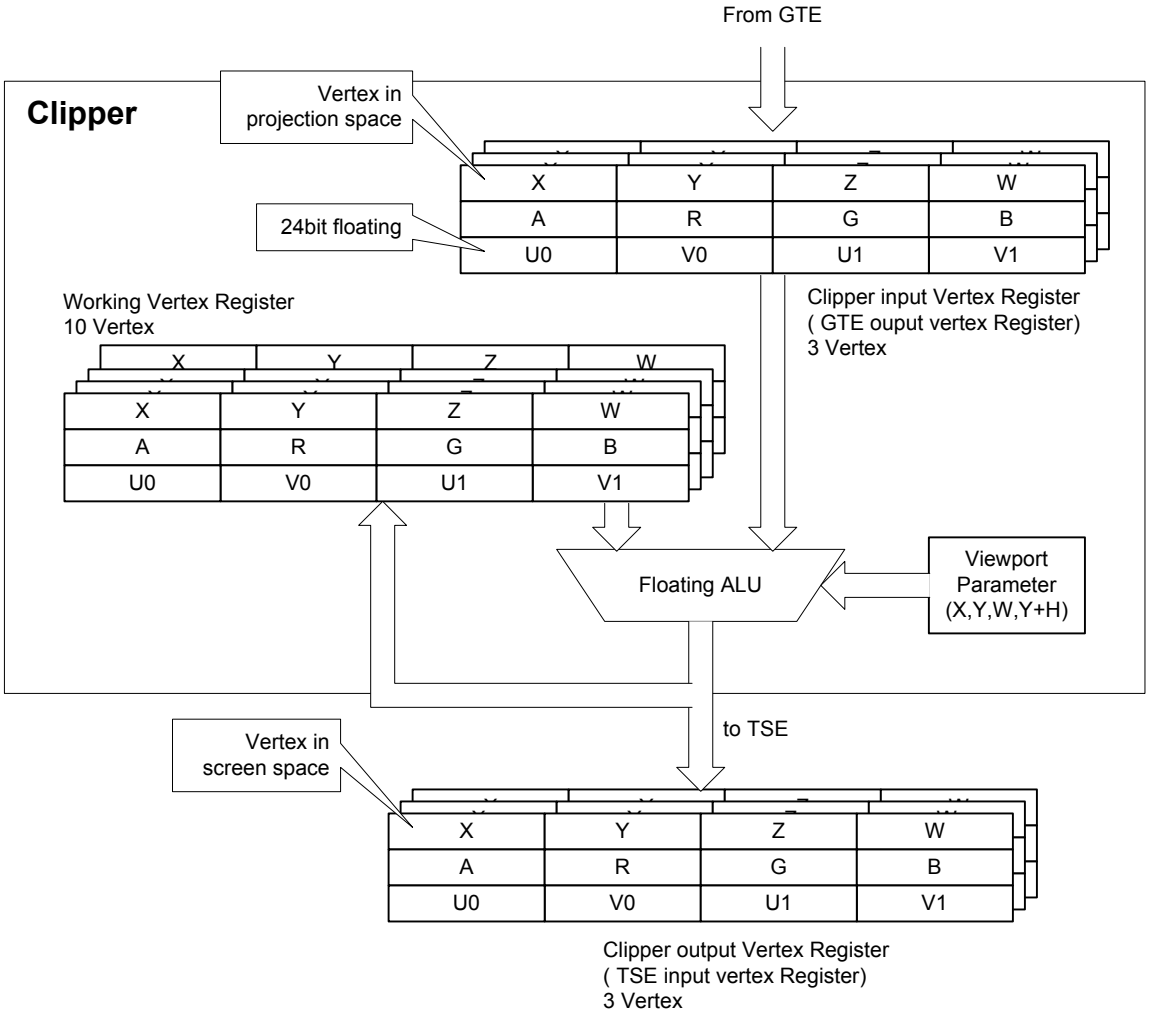


Figure 22-33. Clipper

22.6.1. Viewport

A Viewport defines the area of the screen into which a scene will be rendered. These values correspond to the destination rectangle, or viewport rectangle, as shown in the following illustration.

Refer to : ***GRP3D_VPORTBOT***, ***GRP3D_VPORTH***, ***GRP3D_VPORTLEFT***, ***GRP3D_VPORTW***

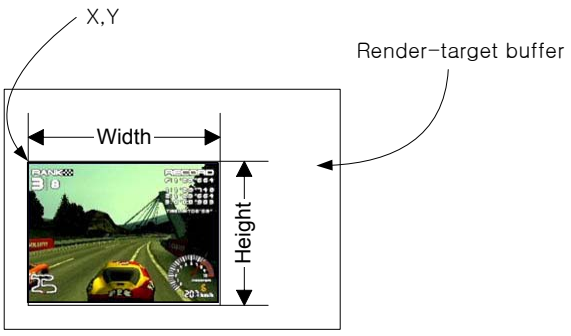


Figure 22-34. View Port

22.7. Rasterizer

22.7.1. Memory fill

Memory fill does filling the rectangle memory area with specific values. Memory fill executes at high speed because it does not use the Pixel pipeline. The clipping rectangle and address converting table have no influence on Memory fill. Memory fill is useful for clearing the Z-

Buffer / Render target / Texture at high speed.
Refer to : ***GRP3D_MFILLPARAM0, GRP3D_MFILLPARAM1, GRP3D_MFILLPARAM2***

22.7.2. Pixel Pipeline

- Depth test

Remove invisible pixels using the Z-buffer.
unsigned 16bit integer
Support 8 test function
Refer to : Depth-buffer, ***GRP3D_ZBINFO***

- Texture blending

Create Source color mixing Texture and Diffuse color
Refer to : ***GPR3D_TEXBLENDn***

- Fog blending

Blending Source color and Fog color using fog value.
Source color * (1 – f) + Fog Color * f
fog table stores fog value of depth corresponding to 0~***GRP3D_FOGMAXZ*** depth domain of camera area. If depth value of a pixel is larger than ***GRP3D_FOGMAXZ***, the fog value sets the last value of the fog table.
Refer to : ***GRP3D_FOGCOLOR, GRP3D_FOGMAXZ, GRP3D_FOGTBL***

- Alpha blending

Source color * ***GRP3D_SRCBLEND*** + Destination Color * ***GRP3D_DESTBLEND***
Refer to : ***GRP3D_ALPHABLEND***(***GRP3D_SRCBLEND,GRP3D_DESTBLEND***)

- Dithering

Convert R8G8B8 format Source color to R5G6B5 format.
It can be enabled/disabled using ***GRP3D_DITHERENB***.
Refer to : ***GRP3D_RENDERSTATE***(***GRP3D_DITHERENB***)

- Pixel writing

Renewal Render target and Z –buffer.

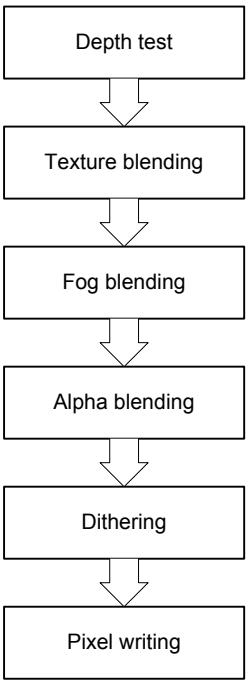


Figure 22-35. Pixel Writing

22.7.3. Render Target

- Render Target

The Render target is a sub-segment. The Screen is directly mapped to a memory sub-segment through an address converting table.
Maximum resolution is 1024x1024 (if antialiasing: 1024x512)

Refer to : Address converting table, ***GRP3D_RENDTRG0(GRP3D_RTSUBSEGMENT,GRP3D_RTTBLX,GRP3D_RTTBLY)***

- FSAA (Full screen antialiasing)

Use 2X method Full screen antialiasing. It needs twice memory area for the render target. (The Maximum vertical resolution of the render target is halved)

Refer to : Address converting table , ***GRP3D_RENDTRG0(GRP3D_RTAA), GRP3D_DISPINFO(GRP3D_DISPAA)***

- Renderable texture

Texture rendering is possible because a block addressing mode of the Render target can be set up.

Refer to : Block addressing mode, ***GRP3D_RENDTRG0(GRP3D_RTADDRMODE)***

- Clipping rectangle

It is a clipping rectangle in the screen space. Pixels outside of the region are not input to the pixel pipeline.

Refer to :***GRP3D_RENDTRG1, GRP3D_RENDTRG2***

22.7.4. Depth-Buffer

- Z-buffer

16bit unsigned integer

The Z-buffer is a Sub-segment. The Screen is directly mapped to a memory sub-segment through an address converting table.

Maximum resolution is 1024x1024 (if antialiasing: 1024x512)

Refer to : Address converting table, ***GRP3D_ZBINFO***

- Z scale

After Projection Transforming, multiply the Z scale and the normalized Z value and store in the Z-buffer (16bit unsigned integer)

Use less than 65535.0f for 24bit float point processing precision.

Refer to : ***GRP3D_ZSCALE***

- Z-valid buffer

The z-valid buffer is a bit buffer to minimize z-buffer reading

1bit corresponds to 8pixels (16bit) and it occupies a 64x128 pixel (16bit) rectangle area in the sub-segment.

If bit = 0, use ***GRP3D_ZVVALUE*** as the z value of the z-buffer without reading the z buffer.

Refer to : ***GRP3D_ZVBINFO***

22.8. Texture

22.8.1. Page

the Page is a 1/32 section of a Segment

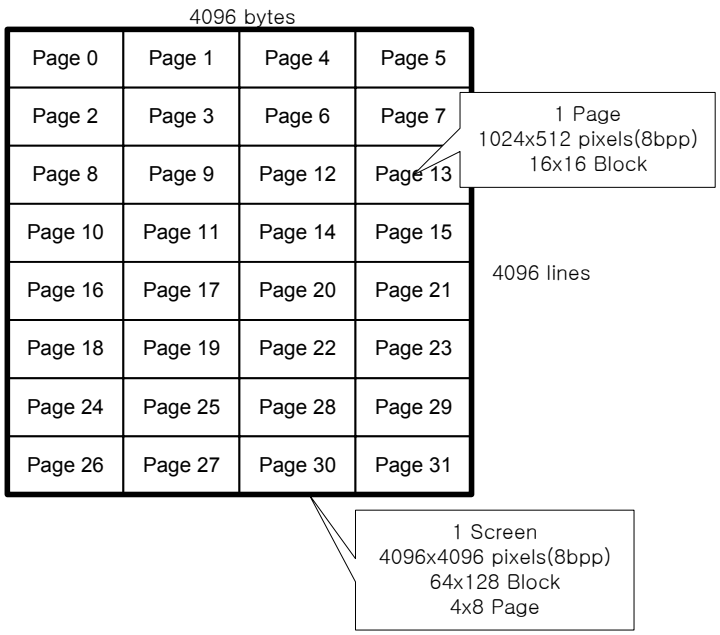


Figure 22-36. Page Segment

22.8.2. Mipmap

The lower right quadrant is reserved for mipmap. If Mipmmap is used, the POLLUX treats the region as another quadrant's downsampled mipmap texture.

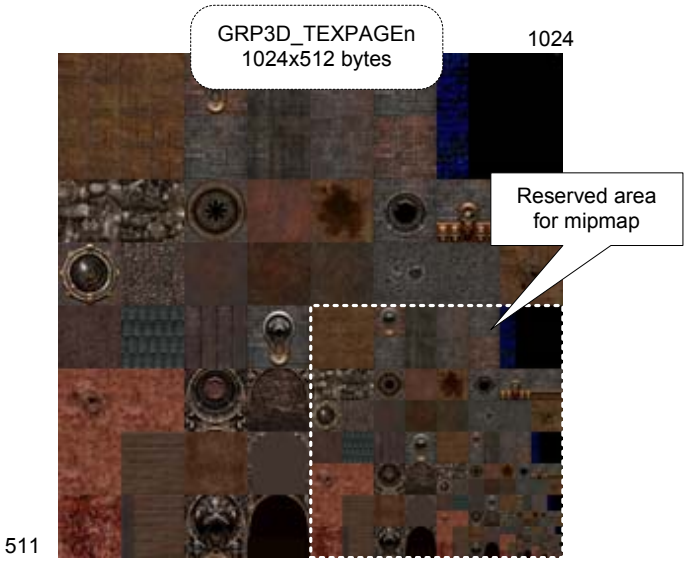


Figure 22-37. MipMap

22.8.3. Texture size

One texture (including the palette) must be fully included in one page.

If mipmap is used, the lower left 512x256(byte) region of the page is reserved for mipmap.

The unit size of texture is 4x2(bytes).

Bits per pixel	Maximum size (with mipmap)	Maximum size (without mipmap)	Minimum size
4	2048x256 or 1024x512	2048x512	8x2
8	1024x256 or 512x512	1024x512	4x2
16	512x256 or 256x512	512x512	2x2

Table 22-33. Texture Size

22.8.4. Texture setting

Texture is a rectangle area used in texture mapping. Texture must have widths and heights specified as powers of 2 and the position of a texture should be multiple of the size.

Refer to : ***GRP3D_TEXSUBSEGMENT, GRP3D_TEXINFOn_0, GRP3D_TEXINFOn_1***

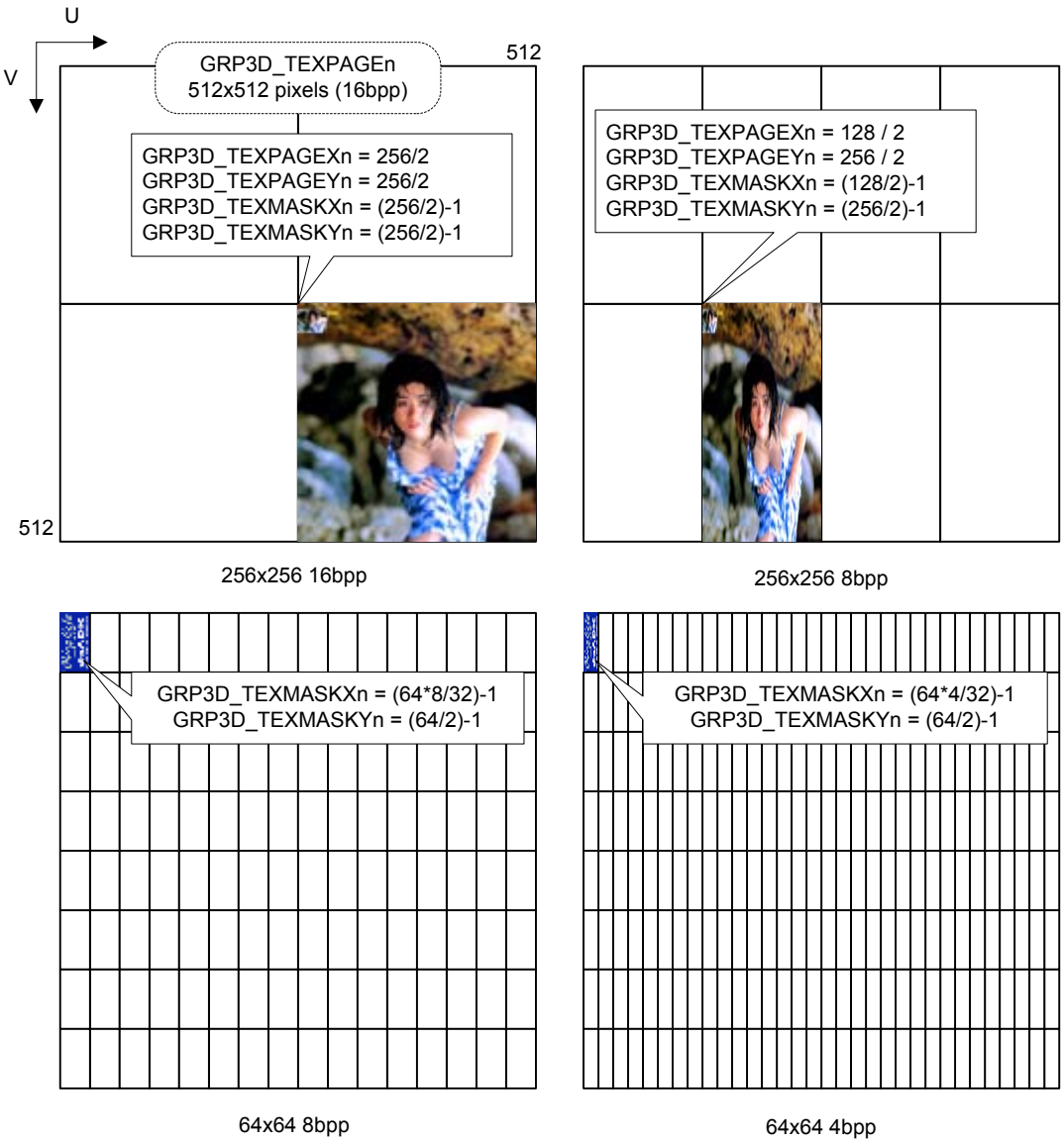


Figure 22-38. Texture Setting

(U,V) cell address

- U,V is 8bpp coordinates in a page (0<=U<=1023, 0<=V<=511)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GRP3D_TEXSUBSEGMENT[10:0]											①	V[8:5]				②	U[9:6]				V[4:1]				U[5:2]				③	④	
								Block index in segment													Block address											

Table 22-34. Texture Setting

GRP3D_TEXSUBSEGMENT[10:0] = {segment[7:0],page[4:2]}

① : bit 1 of page (page=GRP3D_TEXPAGEn)

② : bit 0 of page (page =GRP3D_TEXPAGEn)

③ : bit 0 of V

④ : bit [1:0] of U

22.8.5. Palette

The POLLUX has a Color LookUp Table (LUT) in 16bit R5G6B5 format. This LUT is loaded from memory on request from an application. The Palette is a color data list in the memory and it is expressed as 16x2 bytes or 16x32 bytes rectangles on a page.

Refer to : ***GRP3D_TEXSUBSEGMENT0, GRP3D_LUTPARAM, CMDID_LUTFILL***

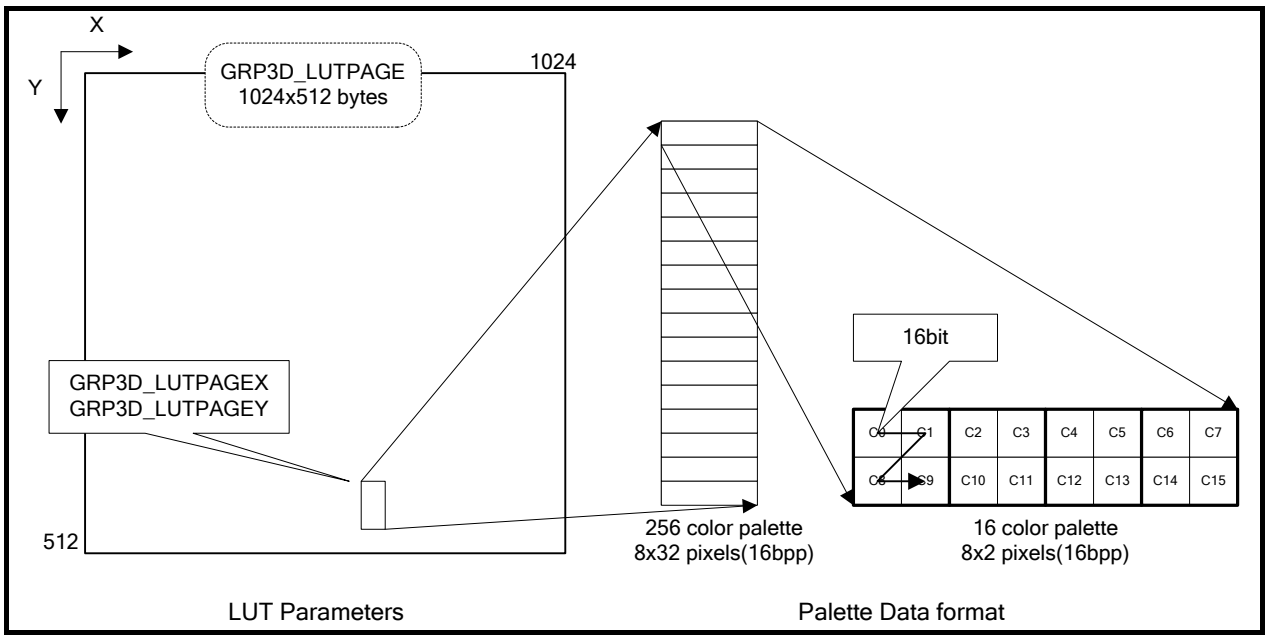


Figure 22-39. Palette

22.8.6. Image texture

Image textures store color data (or color index).

The data in Image textures are texture colors or color indices of LUT

Refer to : ***GRP3D_TEXINFOn_0(GRP3D_COLORMODEn)***

Color mode	Description	Associated Open GL|ES format
0	4bit indexed R5G6B5	GL_PALETTE4_R5_G6_B5_OES
1	4bit indexed R4G4B4A4	GL_PALETTE4_RGBA4_OES
2	4bit indexed R5G5B5A1	GL_PALETTE4_RGB5_A1_OES
3	4bit indexed A8L8	
4	4bit indexed A4R4G4B4	
5	4bit indexed A1R5G5B5	
6	4bit indexed L8A8	
9	A4	
15	L4	
16	8bit indexed R5G6B5	GL_PALETTE8_R5_G6_B5_OES
17	8bit indexed R4G4B4A4	GL_PALETTE8_RGBA4_OES
18	8bit indexed R5G5B5A1	GL_PALETTE8_RGB5_A1_OES
19	8bit indexed A8L8	
20	8bit indexed A4R4G4B4	
21	8bit indexed A1R5G5B5	
22	8bit indexed L8A8	
25	L4A4	
27	L8	GL_LUMINANCE
30	A8	GL_ALPHA
31	A4L4	
40	R5G6B5	GL_UNSIGNED_SHORT_5_6_5
41	R4G4B4A4	GL_UNSIGNED_SHORT_4_4_4_4

42	R5G5B5A1	GL_UNSIGNED_SHORT_5_5_5_1
43	A8L8	GL_LUMINANCE_ALPHA
44	A4R4G4B4	
45	A1R5G5B5	
46	L8A8	

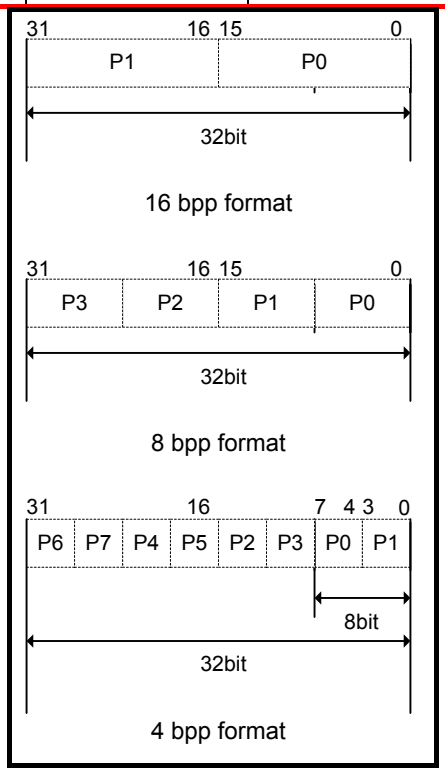


Figure 22-40. Text Format

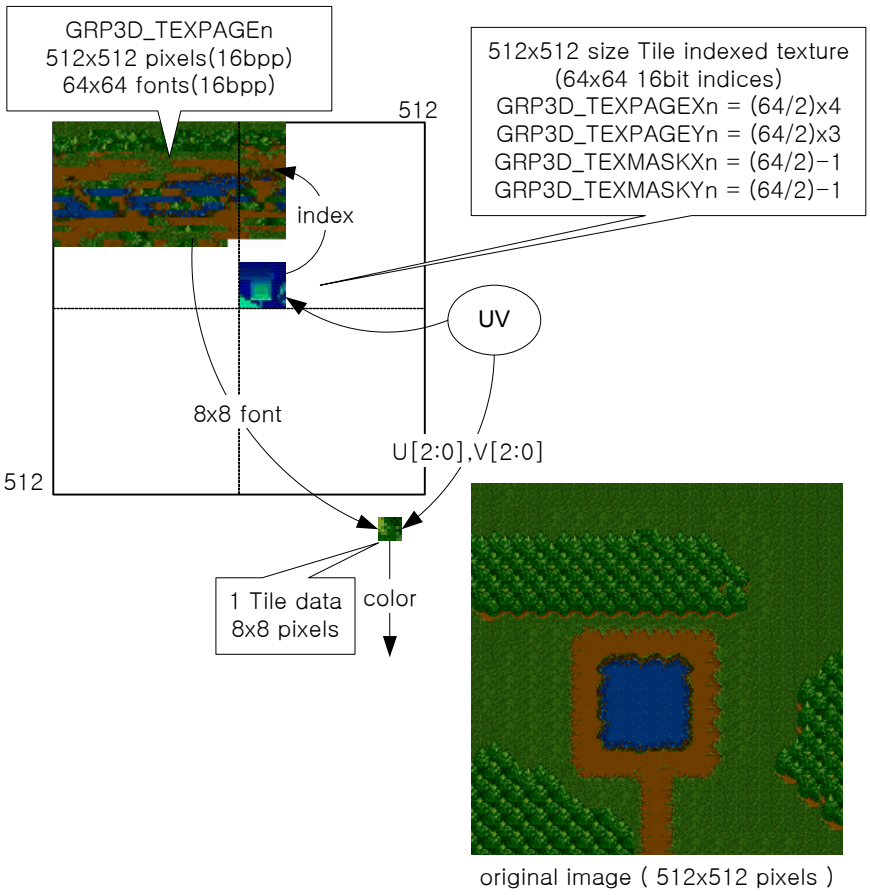
22.8.7. Tile indexed texture

Tile indexed textures store a 16bit tile index.

The data in Tile indexed textures are used in the indexing tile of a page.

Mipmap is not usable.

Refer to : ***GRP3D_TEXINFOn_0(GRP3D_TILEINDEXn)***



if GRP3D_TILEINDEXn is 1, texture is tile-index map. each pixel is 16bit tile index. Tile-index map that constructed by 64x64pixel 16bit indices is treated as 512x512 pixel texture

Figure 22-41. Pixel Texture

22.8.8. Tile

Tiles are 8x8 images indexed by Tile indexed textures. The POLLUX reads color data (or color indices) from the Tile reference index and tu[2:0],tv[2:0] when using Tile indexed textures.

A page is splitted into a set of tiles. The followings are the number of tiles in a page according to the color format.

Bits per pixel	Tile size	Maximum index number
4	2x8x16bit	16383 (256x64-1)
8	4x8x16bit	8191 (128x64-1)
16	8x8x16bit	4095 (64x64-1)

Table 22-35. Tile

Below is given the way to calculate the address of color data (or color indices) by using an index (Idx) and texture U. V read from Tile indexed textures.

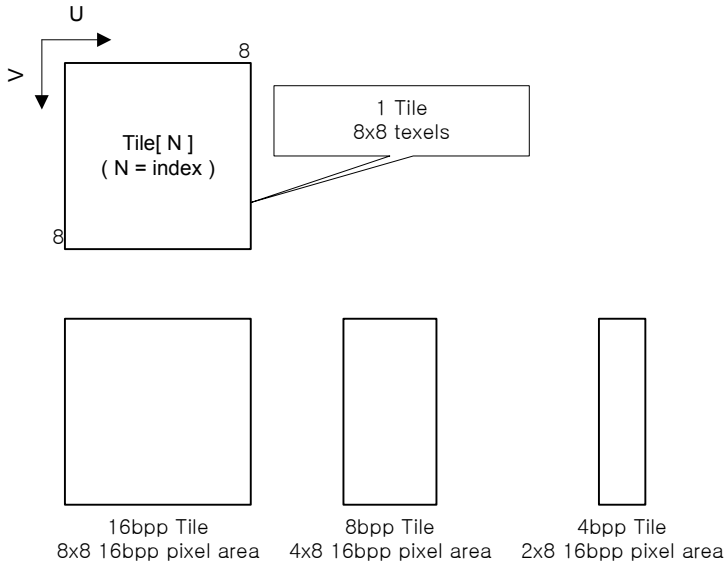


Figure 22-42. Tile

- 16bpp (Idx, U, V) address

- U,V is pixel coordinate in a tile(0<=U<=7, 0<=V<=7)

- Idx : A tile index, (0<= Idx <=4095)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GRP3D_TEXSUBSEGMENT											①	Idx[11:8]				②	Idx[5:2]				Idx[7:6]		V[2:1]		Idx[1:0]		U[2:1]		③	④	0
								Block index in segment													Block address										

Table 22-36. 16bpp Address

GRP3D_TEXSUBSEGMENT = {segment[7:0], page[4:2]}

① : bit 1 of ***GRP3D_TEXPAGE***

② : bit 0 of ***GRP3D_TEXPAGE***

③ : bit 0 of V

④ : bit 0 of U

- 8bpp (Idx, U, V) address
  - U,V is pixel coordinate in a tile (0<=U<=7, 0<=V<=7)
  - Idx : A tile index, (0<= Idx <=8191)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GRP3D_TEXSUBSEGMENT											①	Idx[12:9]				②	Idx[6:3]				Idx[8:7]		V[2:1]		Idx[2:0]			③	④	U[1:0]	
								Block index in segment													Block address										

Table 22-37. 8bpp Address

GRP3D_TEXSUBSEGMENT = {segment[7:0],page[4:2]}

① : bit 1 of ***GRP3D_TEXPAGE***

② : bit 0 of ***GRP3D_TEXPAGE***

③ : bit 2 of U

④ : bit 0 of V

- 4bpp (Idx, U, V) address
  - U,V is pixel coordinate in a tile (0<=U<=7, 0<=V<=7)
  - Idx : A tile index, (0<= Idx <=16383)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GRP3D_TEXSUBSEGMENT											①	Idx[13:10]				②	Idx[7:4]				Idx[9:8]		V[2:1]		Idx[3:0]				③	U[2:1]	
								Block index in segment													Block address										

Table 22-38. 4bpp Address

GRP3D_TEXSUBSEGMENT = {segment[7:0],page[4:2]}

① : bit 1 of ***GRP3D_TEXPAGE***

② : bit 0 of ***GRP3D_TEXPAGE***

③ : bit 0 of V

Palette Index 4bit = U[0] ? low_4bit: high_4bit

22.8.9. Texture cache

The POLLUX has an 8Kbyte texture cache.

The Texture cache needs to be reset every time the texture segment changes, because it operates by segment.

Bits per pixel	Tile cache size (pixel)
4	(32x256) x 2 way
8	(32x128) x 2 way
16	(32x64) x 2 way

Table 22-39. Texture Cache

22.9. Primitive Processor

22.9.1. Overview

The Primitive processor is an active sub-module that renders several primitives using the vertex/index buffer. The Primitive processor has top priority for Register3D so the CPU/Command processor cannot control other modules when the primitive processor is working.

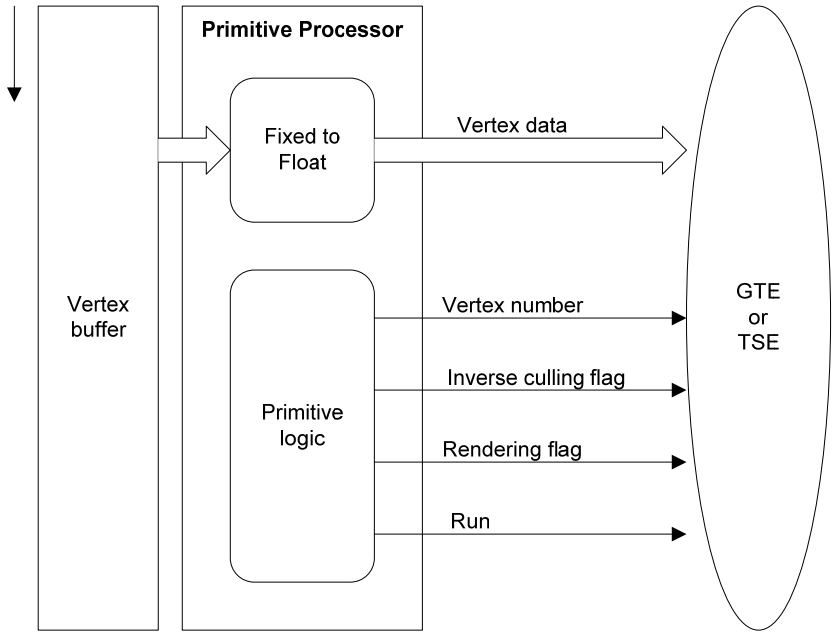


Figure 22-43. Primitive Processor

The basic operation of the Primitive processor is to operate the GTE (or TSE) after loading vertex data to the GTE (or TSE) input register from the vertex buffer.

22.9.2. Primitive mode

The Primitive processor has two operation modes. (Refer to ***GRP3D_PRIMMODE***)

Primitive mode	Description
TSE mode	Use TSE to render Primitive. Vertex data format should be the same as TSE input vertex format. Run TSE when Rendering is needed.
GTE mode	Use GTE to render Primitive. Vertex data should be composite to fit GTE program. Run GTE when Vertex processing is needed.

Table 22-40. Primitive Mode

22.9.3. Primitive control

The Primitive processor has control parameter table to specify control parameters of GTE/TSE.(refer to ***GRP3D_PPARAMn***)

GRP3D_PPARAMn register bits	Description
[4]	Rectangle flag
[3]	Inverse culling flag
[2]	Rendering flag
[1:0]	Vertex number

At each times when the primitive processor operates the GTE (or TSE), primitive control parameter table(***GRRP3D_PPARAMn***) is refered to control the GTE (or TSE).

The following code is a pseudo-code of primitive control operation. (refer to ***GRP3D_PPARAMn***, ***GRP3D_PPARAMEND***, ***GRP3D_PPARAMLOOP, GRP3D_PRIMCON1(GRP3D_USEGTE)*** registers)

```
primitive_control_index = 0;
while( there is one or more remained vertices )
{
    primitive_control = GRP3D_PPARAM [ primitive_control_index ]
    if( index == GRP3D_PPARAMEND ) primitive_control_index = GRP3D_PPARAMLOOP;
    else                           primitive_control_index = primitive_control_index + 1 ;

    vertexnumber      = GetBits( primitive_control, 1, 0 );
    render_flag       = GetBit ( primitive_control, 2 );
    inverseculling_flag = GetBit ( primitive_control, 3 );
    rectangle_flag    = GetBit ( primitive_control, 4 );

    // Read vertex index
    vertex_index = FetchVertexIndex();

    if( GRP3D_USEGTE ) // GTE operation
    {
        // Read vertex data from vertex buffer memory to GTE
        FetchVertexData_To_GTE(vertex_index);
        // Run GTE
        RunGTE( vertexnumber, inverseculling_flag, rectangle_flag );
    }
    else // TSE operation
    {
        // Read vertex data from vertex buffer memory to TSE input
        FetchVertexData_To_TSE(vertex_index,vertexnumber);
        // Run TSE
        if( rectangle_flag ) RunTSE( rectangle_flag, inverseculling_flag );
    }
}
```

Here are several examples of primitive control parameters to render normal primitives.

Primitive type	GRP3D_PPARAMn	GRP3D_PPARAMEND	GRP3D_PPARAMLOOP
Triangle list	0 (=={0,0,0,0})	2	0
	1 (=={0,0,0,1})		
	6 (=={0,0,1,2})		
Triangle fan	0 (=={0,0,0,0})	3	2
	1 (=={0,0,0,1})		
	6 (=={0,0,1,2})		
	13 (=={0,1,1,1})		

Primitive type	GRP3D_PPARAMn	GRP3D_PPARAMEND	GRP3D_PPARAMLOOP
Triangle strip	0 (=={0,0,0,0})	7	2
	1 (=={0,0,0,1})		
	6 (=={0,0,1,2})		
	12 (=={0,1,1,0})		
	5 (=={0,0,1,1})		
	14 (=={0,1,1,2})		
	4 (=={0,0,1,0})		
	13 (=={0,1,1,1})		
Rectangle list	0 (=={0,0,0,0})	1	0
	21 (=={1,0,1,1})		

Table 22-41. Examples of primitive control parameters

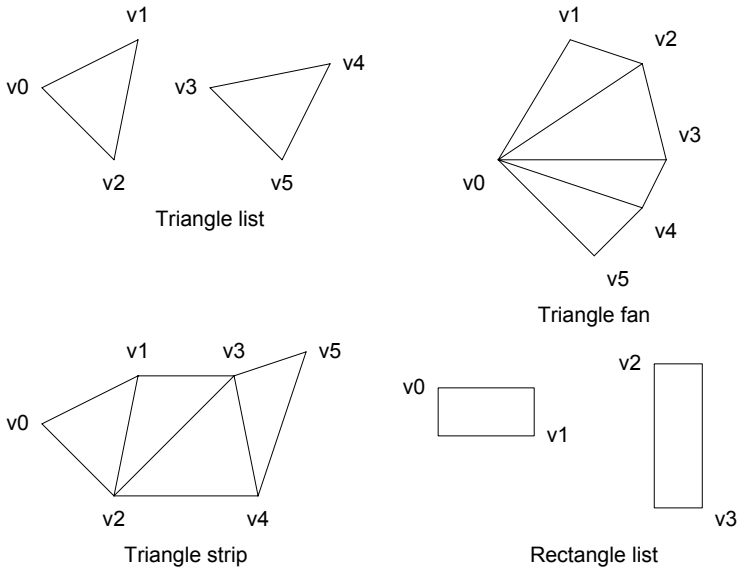


Figure 22-44. Primitive Type

```
// Render triangle-strip: TSE mode
// Load TSE(vertex number) - Load vertex to TSE input vertex register
// Run TSE( rectangle flag, inverse culling flag ) – TSE not use rendering flag
Load TSE(0)
Load TSE(1)
Load TSE(2)
    Run TSE( 0, 0 )
Load TSE(0)
    Run TSE( 0, 1 )
Load TSE(1)
    Run TSE( 0, 0 )
…
```

```
// Render triangle-strip: GTE mode
// Load GTE() - Load vertex to GTE input vector register
// Run GTE (output vertex number, inverse culling flag, rendering flag)
Load GTE()
    Run GTE (0, x, false) // process vertex 0
Load GTE()
    Run GTE (1, x, false) // process vertex 1
```

```
Load GTE()
    Run GTE (2, 0, true) // process vertex 2 & rendering
Load GTE()
    Run GTE (0, 1, true) // process vertex 0 & rendering
Load GTE()
    Run GTE (1, 0, true) // process vertex 1 & rendering
…
```

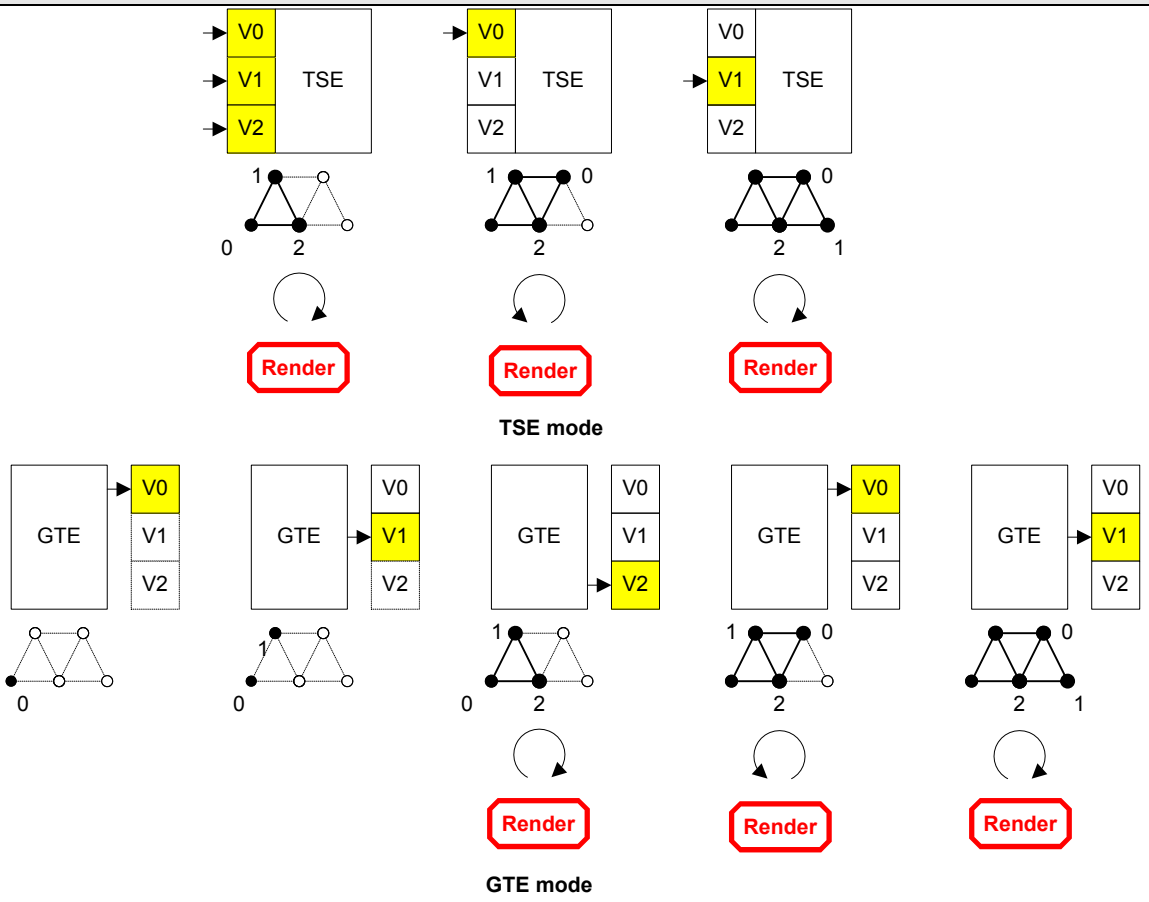


Figure 22-45. Triangle Strip at GTE

22.9.4. Vertex buffer

The Vertex buffer is a buffer containing data for the input register of GTE/TSE. It consists of a 8/16/32bit fixed point number or a 32bit floating point number. Each set of numbers is converted into a floating point number by the Primitive processor and transferred to GTE/TSE. You can use eight vertex buffers to specify a vertex data. (Refer to ***GRP3D_VSADDRn, GRP3D_VSSTRIDEn, GRP3D_VSPARAMn***)

GRP3D_VSADDRn register specifies the address of a vertex buffer. ***GRP3D_VSSTRIDEn*** register specifies the byte offset between consecutive array elements. ***GRP3D_VSPARAMn*** register specifies data type of array elements, number of components per array element and destination register offset of the GTE/TSE.

GRP3D_VSPARAMn register bits	Register name	Description
[31:20]	*GRP3D_VSDSTn*	Destination register offset of the GTE/TSE (unit 32bit) ▪ The destination offset of Input/Constant register of the GTE is (0x1000 / 4) ▪ The destination offset of Input register of the TSE is (0x0500 / 4).
[19:8]	*GRP3D_VSSIZEn*	Number of components per array element (N-1)

GRP3D_VSPARAMn register bits	Register name	Description
[7:0]	***GRP3D_VSTYPEn***	Data type of array elements

GRP_VSTYPE	Description	To float (in C/C++)
0	Normalized unsigned 8bit	((unsigned char)Data) / 255.0f
1	Integer unsigned 8bit	(float) ((unsigned char)Data)
2	Integer signed 8 bit	(float) ((char)Data)
3	Integer signed 16 bit	(float) ((short)Data)
4	Integer signed 32 bit	(float) ((int)Data)
5	Fixed point number s.15.16	((int)Data) / 65536.0f
6	Reserved	
7	Floating point number s.e8.23	Bypass

The following pseudo-code specifies the vertex fetch operation from vertex streams. (refer to ***GRP3D_STREAMVALID, GRP3D_VSADDRn, GRP3D_VSSTRIDEn, GRP3D_VSADDRn, GRP3D_VSSTRIDEn, GRP3D_VSTYPEn, GRP3D_PRIMCON1(GRP3D_USEGTE)*** registers)

```
void FetchVertexData_To_GTE( int vertex_index ) // 1 == GRP3D_PRIMCON1(GRP3D_USEGTE)
{
    for( int stream_index=0; stream_index<8; stream_index++ )
    {
        if( 0 == GRP3D_STREAMVALID[stream_index] ){ continue; }

        void* vertex_address = GRP3D_VSADDR[stream_index];
        int   vertex_offset  = GRP3D_VSSTRIDE[stream_index] * vertex_index;

        // BASE_ASSRESS_OF_3D_CORE is base address of 3D core. It is 0xC001A000 in POLLUX.
        void* dest_address = BASE_ASSRESS_OF_3D_CORE;
        int   dest_offset  = GRP3D_VSDST[stream_index] * 4;

        for( int size=0; size < GRP3D_VSSIZE[stream_index]; size++ )
        {
            // read a component
            float f = read_a_number( vertex_address, vertex_offset, GRP3D_VSTYPE[stream_index] );

            // write a component to GTE Input/Constant register
            write_a_number( dest_address, dest_offset, f );

            // get next component offset
            vertex_offset += typesize( GRP3D_VSTYPE[stream_index] );// bytes of a component
            dest_offset   += 4;
        }
    }
}
```

```
void FetchVertexData_To_TSE( int vertex_index, int vertexnumber ) // 0 == GRP3D_PRIMCON1(GRP3D_USEGTE)
{
    for( int stream_index=0; stream_index<8; stream_index++ )
    {
        if( 0 == GRP3D_STREAMVALID[stream_index] ){ continue; }

        void* vertex_address = GRP3D_VSADDR[stream_index];
        int   vertex_offset  = GRP3D_VSSTRIDE[stream_index] * vertex_index;

        // BASE_ASSRESS_OF_3D_CORE is base address of 3D core. It is 0xC001A000 in POLLUX.
        void* dest_address = BASE_ASSRESS_OF_3D_CORE + 0x500;
        int   dest_offset  = ( (vertexnumber * 16) + (GRP3D_VSDST[stream_index] % 16) ) * 4;

        for( int size=0; size < GRP3D_VSSIZE[stream_index]; size++ )
        {
            // read a component
            float f = read_a_number( vertex_address, vertex_offset, GRP3D_VSTYPE[stream_index] );

            // write a component to GTE Input/Constant register
            write_a_number( dest_address, dest_offset, f );
```

```
                // get next component offset
                vertex_offset += typesize( GRP3D_VSTYPE[stream_index] );// bytes of a component
                dest_offset   += 4;
            }
        }
    }
```

22.9.5. Index buffer

The Index buffer stores the vertex index. It consists of an 8bit, 16bit, 24bit or 32bit unsigned integer array. Each index can be used for indexing the vertex.

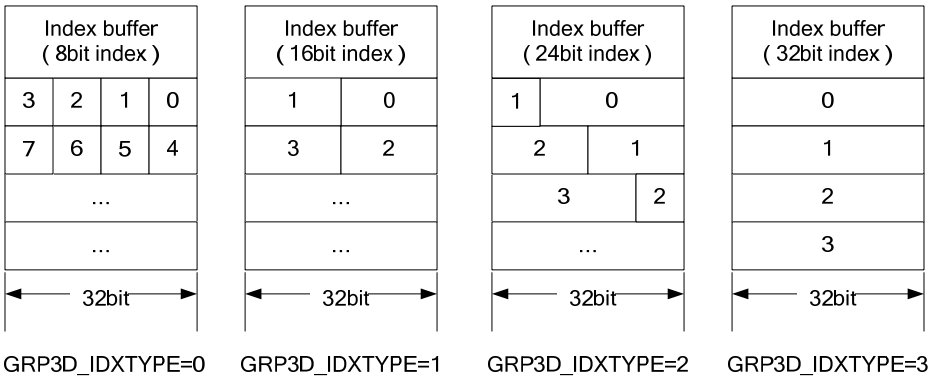


Figure 22-46. Index Data Format

The following pseudo-code specifies the index fetch operation from index stream. (Refer to ***GRP3D_IDXSTART, GRP3D_IDXEND, GRP3D_PRIMCON1(GRP3D_USEIDX), GRP3D_PRIMCON1(GRP3D_IDXTYPE)***)

```
int cur_index = GRP3D_IDXSTART;

int FetchVertexIndex( void )
{
    if( cur_index > GRP3D_IDXEND ){ return NO_MORE_INDEX; }

    int index;

    if( GRP3D_USEIDX )
    {
        // cur_index has the index buffer address
        index = read_a_index( (void*)cur_index, GRP3D_IDXTYPE );
        cur_index += typesize( GRP3D_IDXTYPE );
    }
    else
    {
        // cur_index is a index
        index = cur_index;
        cur_index++;
    }
    return index;
}
```

22.9.6. Matrix palette

If ***GRP3D_USEMPALETTE*** register is 1, vertex stream 0 is treated as matrix index buffer and vertex stream 7 is treated as matrix buffer. In this case, the type of vertex stream 0 must be 1(unsigned 8bit).

The following pseudo-code specifies the vertex data fetch operation from vertex stream to GTE as mentioned above. But this is a case of ***GRP3D_USEMPALETTE*** register is 1.

```
void FetchVertexData_To_GTE( int vertex_index )
{
    unsigned char matrix_index[];

    for( int stream_index=0; stream_index<8; stream_index++ )
    {
        if( 0 == GRP3D_STREAMVALID[stream_index] ){ continue; }
```

```
        void* vertex_address = GRP3D_VSADDR[stream_index];
        int     vertex_offset    = GRP3D_VSSTRIDE[stream_index] * vertex_index;

        // BASE_ASSRESS_OF_3D_CORE is base address of 3D core. It is 0xC001A000 in POLLUX.
        void* dest_address = BASE_ASSRESS_OF_3D_CORE;
        int     dest_offset    = GRP3D_VSDST[stream_index] * 4;

        for( int size=0; size < GRP3D_VSSIZE[stream_index]; size++ )
        {
            if( 0 == stream_index )
            {
                matrix_index[size] = read_a_byte( vertex_address, vertex_offset, GRP3D_VSTYPE[stream_index] );
                vertex_offset += typesize( GRP3D_VSTYPE[stream_index] );// bytes of a component
            }
            else
            {
                // read a component
                float f = read_a_number( vertex_address, vertex_offset, GRP3D_VSTYPE[stream_index] );

                // write a component to GTE Input/Constant register
                write_a_number( dest_address, dest_offset, f );

                // get next component offset
                vertex_offset += typesize( GRP3D_VSTYPE[stream_index] );// bytes of a component
                dest_offset     += 4;
            }
        }
    }

    if( 0 == GRP3D_STREAMVALID[7] ){ return; }
    void* vertex_address = GRP3D_VSADDR[7];
    void* dest_address     = BASE_ASSRESS_OF_3D_CORE;
    int     dest_offset      = GRP3D_VSDST[7] * 4;
    for( int matrix_count=0; matrix_count<GRP3D_VSSIZE[0]; matrix_count++ )
    {
        int     vertex_offset    = GRP3D_VSSTRIDE[7] * matrix_index[matrix_count];

        for( int size=0; size < GRP3D_VSSIZE[7]; size++ )
        {
            // read a component
            float f = read_a_number( vertex_address, vertex_offset, GRP3D_VSTYPE[7] );

            // write a component to GTE Input/Constant register
            write_a_number( dest_address, dest_offset, f );

            // get next component offset
            vertex_offset += typesize( GRP3D_VSTYPE[7] );// bytes of a component
            dest_offset     += 4;
        }
    }
}
```

22.10. Display (Page flipping)

Display contents to a CTR/LCD can be assigned by the sub-segment, address converting table. (Refer to ***GRP3D_DISPINFO***)

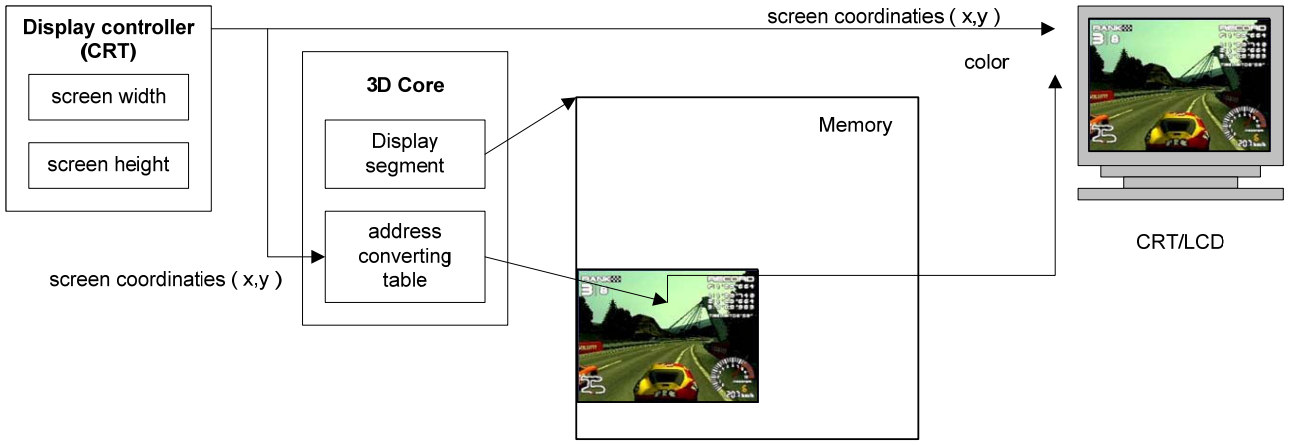


Figure 22-47. Page Flipping

22.10.1. Page flipping

Write a suitable value to display a register for Page flipping. It will display the corresponding memory contents to the screen. (Refer to DISPINFO)

Display information register (*DISPINFO*)	Description
Display antialiasing flag (GRP3D_DISPAA)	If '1', mix 2 lines and display 1 line. Should be '1' to display rendering image of target register antialiasing flag (GRP3D_RTAA) '1' to CRT/LCT.
Display segment (GRP3D_DISPSUBSEGMENT)	Sub-segment that has screen contents
Address converting table (GRP3D_DISPTBLX, GRP3D_DISPTBLY)	Converting table thatmaps CRT/LCD screen x, y coordinates to Memory segment. (refer to 'address converting table')

Table 22-42. Page Flipping

22.10.2. Synchronous page flipping

Once a data is written to the display register, the contents of the memory will be displayed on to the CRT/LCD. To avoid the tier phenomenon, the display register should be renewed to meet the vertical retrace.

There are two different methods to set the register to meet a vertical retrace that depend on using the Command queue/buffer.

Case	Description
Direct access	Use vertical retrace interrupt from CRT controller. After 3D core operation, receive interrupt and write value to register.
Use command queue/buffer	Use register fill command at Command queue/buffer. Using WAIT_SYNC option for changing register to synchronize to vertical retrace.

Table 22-43. Synchronous Page Flipping

22.11. 3D Core Registers

3D CORE REGISTER MAP

Address	Description
C001_A000 ~ C001_A0FF	General / Control registers
C001_A100 ~ C001_A1FF	General / Control registers
C001_A200 ~ C001_A2FF	Reserved (Do not access)
C001_A300 ~ C001_A3FF	Reserved (Do not access)
C001_A400 ~ C001_A4FF	Fog table
C001_A500 ~ C001_A5FF	TSE input vertex registers

C001_A600 ~ C001_A6FF	Reserved (Do not access)
C001_A700 ~ C001_A7FF	GTE output (Clipper input) registers
C001_A800 ~ C001_AFFF	GTE program registers (max 512 instructions)
C001_B000 ~ C001_BFFF	GTE constant/input vector registers (max 252 vectors)
C001_BFC0 ~ C001_BFFF	Clock enable register

Table 22-44. 3D Core Registers

22.11.1. General / Control register map (C001_A000h ~ C001_A1FFh)

Register Name	offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPCONTROL	00h																														CPCONTROL		
CMDQSTART	04h	CMDQSTART																													0	0	0
CMDQEND	08h	CMDQEND																													1	0	0
CMDQFRONT	0Ch	CMDQFRONT																														0	0
CMDQREAR	10h	CMDQREAR																													0	0	0
STATUS	14h											Sub-module idle flags						READVCNT															
INT	18h																																
	1Ch																																
CONTROL	20h	0	1	0	0																												
POWERCONTROL	24h																	POWERCONTROL															
RENDERSTATE	28h	0	1	0	1																												
	2Ch																																
ALPHABLEND	30h	0	1	1	0																			DESTBLEND					SRCBLEND				
	34h																																
TEXSUBSEGMENT	38h	0	1	1	1		TEXSUBSEGMENT1																TEXSUBSEGMENT0										
	3Ch																																
TPCOLOR	40h	1	0	0	0													TPCOLOR															
MIPMAPBIAS	44h																	MIPMAPBIAS															
LUTPARAM	48h	1	0	0	1				LUTBANK									LUTPAGEY							LUTPAGEX						0	0
	4Ch																																
TEXINFO0_0	50h	1	0	1	0			COLORMODE																									
TEXINFO0_1	54h	TEXMASKY								TEXMASKX								TEXPAGEY							TEXPAGEX								
TEXINFO1_0	58h	1	0	1	1			COLORMODE																									
TEXINFO1_1	5Ch	TEXMASKY								TEXMASKX								TEXPAGEY							TEXPAGEX								
TEXBLEND0	60h	1	1	0	0	RGBARG0				RGBARG1				RGBARG2				RGBOP				AARG0			AARG1			AARG2			AOP		
	64h																																
TEXBLEND1	68h	1	1	0	1	RGBARG0				RGBARG1				RGBARG2				RGBOP				AARG0			AARG1			AARG2			AOP		
	6Ch																																
RESERVED	70h																																
RESERVED	74h																																
RESERVED	78h																																
RESERVED	7Ch																																
RESERVED	80h																																
RESERVED	84h																																
ALPHATEST	88h																	ALPHAREF							ALPHAFUNC								
TEXCONST	8Ch	TEXCONST_A								TEXCONST_R								TEXCONST_G							TEXCONST_B								
FOGCOLOR	90h									FOGCOLOR																							

FOGMAXZ	94h	FOGMAXZ																														
DISPINFO	98h					DISPSUBSEGMENT											DISPTBLY											DISPTBLX					
	9Ch																															
MFILLPARAM0	A0h						MFSUBSEGMENT										MFDATA															
MFILLPARAM1	A4h							MFTOP															MFLEFT									
MFILLPARAM2	A8h							MFBOTTOM															MFRIGHT									
	ACh																															
RENDTRG0	B0h						RTSUBSEGMENT										RTTBLY											RTTBLX					
RENDTRG1	B4h							RTTOP															RTLEFT									
RENDTRG2	B8h							RTBOTTOM															RTRIGHT									
	BCh																															
ZBINFO	C0h		ZBFUNC				ZBSUBSEGMENT										ZBTBLY											ZBTBLX					
ZVBINFO	C4h									ZVBTOP		0		ZVBLEFT			ZVVALUE															
ZSCALE	C8h	ZSCALE																														
	CCh																															
RESERVED	D0h																															
RESERVED	D4h																															
RESERVED	D8h																															
RESERVED	DCh																															
RESERVED	E0h																															
	E4h																															
	E8h																															
	ECh																															
VPORTBOT	F0h	VPORTBOT																														
VPORTH	F4h	VPORTH																														
VPORTLEFT	F8h	VPORTLEFT																														
VPORTW	FCh	VPORTW																														
PRIMCON1	100h																STREAMVALID															
IDXSTART	104h	IDXSTART																														
IDXEND	108h	IDXEND																														
PPARAMLOOPCTRL	10Ch																PPARAMLOOP											PPARAMEND					
PPARAM03	110h	PPARAM3								PPARAM2							PPARAM1											PPARAM0					
PPARAM47	114h	PPARAM7								PPARAM6							PPARAM5											PPARAM4					
PPARAM8B	118h	PPARAMB								PPARAMA							PPARAM9											PPARAM8					
PPARAMCF	11Ch	PPARAMF								PPARAME							PPARAMD											PPARAMC					
VSADDR0	120h	VSADDR0																														
VSSTRIDE0	124h	VSSTRIDE0																														
VSPARAM0	128h	VSDST0															VSSIZE0											VSTYPE0					
VSADDR1	12Ch	VSADDR1																														
VSSTRIDE1	130h	VSSTRIDE1																														
VSPARAM1	134h	VSDST1															VSSIZE1											VSTYPE1					
VSADDR2	138h	VSADDR2																														
VSSTRIDE2	13Ch	VSSTRIDE2																														
VSPARAM2	140h	VSDST2															VSSIZE2											VSTYPE2					
VSADDR3	144h	VSADDR3																														
VSSTRIDE3	148h	VSSTRIDE3																														
VSPARAM3	14Ch	VSDST3															VSSIZE3											VSTYPE3					

VSADDR4	150h	VSADDR4		
VSSTRIDE4	154h	VSSTRIDE4		
VSPARAM4	158h	VSDST4	VSSIZE4	VSTYPE4
VSADDR5	15Ch	VSADDR5		
VSSTRIDE5	160h	VSSTRIDE5		
VSPARAM5	164h	VSDST5	VSSIZE5	VSTYPE5
VSADDR6	168h	VSADDR6		
VSSTRIDE6	16Ch	VSSTRIDE6		
VSPARAM6	170h	VSDST6	VSSIZE6	VSTYPE6
VSADDR7	174h	VSADDR7		
VSSTRIDE7	178h	VSSTRIDE7		
VSPARAM7	17Ch	VSDST7	VSSIZE7	VSTYPE7

22.12. Register Summary

Bit	R/W	Symbol	Description	Reset Value
COMMAND PROCESSOR CONTROL REGISTER (GRP3D_CPCONTROL) ***Address : C001_A000h***				
[31 : 3]	-	**RESERVED**	Reserved	-
[2]	W	**GRP3D_CPPAUSE**	Command/ Primitive Processor Pause State 0 : None 1 : Pause Command/ Processor	-
[1]	W	**GRP3D_CPMODE**	Command Processor Operation Mode 0 : Command Queue Mode If GRP3D_CMDQFRONT == GRP3D_CMDQREAR, Command Processor waits for a new Command. 1 : Command Buffer Mode If GRP3D_CMDQFRONT == GRP3D_CMDQREAR, Command Processor state becomes idle	-
[0]	W	**GRP3D_CPRUN**	0 : Just Change GRP3D_CPMODE or GRP3D_CPPAUSE 1 : Command Processor state operation	-
COMMAND QUEUE START ADDRESS REGISTER (GRP3D_CMDQSTART) ***Address : C001_A004h***				
[31 : 0]	R/W	**GRP3D_CMDQSTART**	Command Queue Start Address. This address must be aligned to 8Byte (64bit) (GRP3D_CMDQSTART[2 : 0] must be '0')	-
COMMAND QUEUE END ADDRESS REGISTER (GRP3D_CMDQEND) ***Address : C001_A008h***				
[31 : 0]	R/W	**GRP3D_CMDQEND**	Command Queue End Address. (GRP3D_CMDQEND[2 : 0] must be '0')	-
COMMAND QUEUE FRONT POINTER REGISTER (GRP3D_CMDQFRONT) ***Address : C001_A00Ch***				
[31 : 0]	R/W	**GRP3D_CMDQFRONT**	Command Queue Front Pointer. This address must be Aligned to 4Byte (32bit) (GRP3D_CMDQFRONT[1 : 0] must be '0')	-
COMMAND QUEUE REAR POINTER REGISTER (GRP3D_CMDQREAR) ***Address : C001_A010h***				
[31 : 0]	R	**GRP3D_CMDQREAR**	Command Queue Rear Pointer. If GRP3D_CMDQREAR == GRP3D_CMDQFRONT, Command queue is empty If GRP3D_CMDQREAR – 4 == GRP3D_CMDQFRONT, Command queue is full	-
3D CORE STATE REGISTER (GRP3D_STATUS) ***Address : C001_A014h***				
[31 :24]	-	**RESERVED**	Reserved	-
[23]	R	**GRP3D_RASTEMPTY**	Empty flag of Rasterizer input FIFO	1'b1
[22]	R	**GRP3D_RASTFULL**	Full Flag of Rasterizer input FIFO	1'b0
[21]	R	**GRP3D_CRIDLE**	Command Reader Idle Flag	1'b1
[20]	R	**GRP3D_CPIDLE**	Command Processor Idle Flag	1'b1
[19]	R	**GRP3D_PRIMIDLE**	Primitive Processor Idle Flag	1'b1
[18]	R	**GRP3D_RASTIDLE**	Rasterizer Idle Flag	1'b1
[17]	R	**GRP3D_TSEIDLE**	TSE Idle Flag	1'b1
[16]	R	**GRP3D_GTEIDLE**	GTE Idle Flag	1'b1

Bit	R/W	Symbol	Description	Reset Value
[15 : 0]	R	**GRP3D_READVCNT**	Number of vertices that are loaded by Primitive processor Goes to '0' status when Primitive Processor Starts operation	-
3D CORE INTERRUPT REGISTER (GRP3D_INT) ***Address : C001_A018h***				
[31 :10]	-	**RESERVED**	Reserved	-
[9]	R/W	**GRP3D_INVCMDINTR**	Invalid Command Interrupt Enable	1'b0
[8]	R/W	**GRP3D_REGFILLINTR**	Register Fill Interrupt Enable	1'b0
[7 : 2]	-	**RESERVED**	Reserved	-
[1]	R/W	**GRP3D_INVCMDPEND**	Invalid Command Interrupt Pending Read > 0 : No Interrupt is Generated 1 : Interrupt caused because invalid command was found Write > 0 : None 1 : Interrupt Pending bit (GRP3D_INVCMDPEND) clear	1'b0
[0]	R/W	**GRP3D_REGFILLPEND**	Register Fill Interrupt Pending Read > 0 : No Interrupt is Generated 1 : Interrupt caused because a Register fill command (with interrupt) was processed. Write > 0 : None 1 : interrupt pending bit (GRP3D_REGFILLPEND) clear	1'b0
3C CORE SUB-MODULE CONTROL REGISTER (GRP3D_CONTROL) ***Address : C001_A020h***				
[31 :11]	-	**RESERVED**	Reserved	-
[10]	W	**GRP3D_TSEMODE**	TSE operation mode 1: Triangle setup mode 0: Rectangle setup mode If GRP3D_*MODULEID* is not TSE(2), this bit is ignored.	-
[9]	W	**GRP3D_CULLMODE**	GTE/TSE culling mode 1: Backface culling is affected by ~(GRP3D_*CULLDIR*) 0: Backface culling is affected by GRP3D_*CULLDIR* If GRP3D_*MODULEID* is not GTE(1) or TSE(2), this bit is ignored.	1'b0
[8]	W	**GRP3D_CULLMODE**	GTE/TSE culling mode 1: Backface culling is affected by ~(GRP3D_*CULLDIR*) 0: Backface culling is affected by GRP3D_*CULLDIR* If GRP3D_*MODULEID* is not GTE(1) or TSE(2), this bit is ignored.	-
[7 : 6]	W	**GRP3D_GTEVNUM**	GTE output vertex number (0,1,2) If GRP3D_*MODULEID* is not GTE(1), these bits are ignored	-
[5]	W	**GRP3D_SPANDIR**	Rasterizer span direction. If this flag is 1, span direction is right. If GRP3D_RASTMODE is not 0, this flag is ignored. If GRP3D_*MODULEID* is not Rasterizer(3), this bit is ignored.	-
[4 : 3]		**GRP3D_RASTMODE**	Rasterizer operation mode 0 : Rendering 1 : Memory fill (for fast z-buffer/screen clear) 2 : LUT fill 3 : Texture cache clear If GRP3D_*MODULEID* is not Rasterizer(3), these bits are ignored.	-
[2]	W	**GRP3D_NOP**	NOP Flag	-

Bit	R/W	Symbol	Description	Reset Value
			If this bit is 1, all bits are ignored. If this bit is 1, CMDID_CONTROL just waits for the sub-module to become idle	
[1 : 0]	W	**GRP3D_MODULEID**	Sub_Module ID 00 : Primitive Processor 01 : GTE 10 : TSE 11 : Rasterizer	-
3C CORE POWER CONTROL REGISTER (GRP3D_POWERCONTROL) ***Address : C001_A024h***				
[31 :16]	-	**RESERVED**	Reserved	-
[15:14]	W	**GRP3D_POWER_VERTEXCACHE**	Power control mode of vertex cache 00: Power down mode 01: Sleep mode 10: Prohibit 11: Operation mode	2'b00
[13:12]	W	**GRP3D_POWER_GTEREG**	Power control mode of GTE register 00: Power down mode 01: Sleep mode 10: Prohibit 11: Operation mode	2'b00
[11:10]	W	**GRP3D_POWER_GTEPROG**	Power control mode of GTE instruction memory 00: Power down mode 01: Sleep mode 10: Prohibit 11: Operation mode	2'b00
[9:8]	W	**GRP3D_POWER_CLIPREG**	Power control mode of clipper local register 00: Power down mode 01: Sleep mode 10: Prohibit 11: Operation mode	2'b00
[7:6]	W	**GRP3D_POWER_PERSPECTIVE**	Power control mode of perspective correction ROM 00: Power down mode 01: Sleep mode 10: Prohibit 11: Operation mode	2'b00
[5:4]	W	**GRP3D_POWER_TCACHE0**	Power control mode of texture cache 00: Power down mode 01: Sleep mode 10: Prohibit 11: Operation mode	2'b00
[3:2]	W	**GRP3D_POWER_TCACHE1**	Power control mode of texture cache 00: Power down mode 01: Sleep mode 10: Prohibit 11: Operation mode	2'b00
[1:0]	W	**GRP3D_POWER_TPALETTE**	Power control mode of texture palette table 00: Power down mode 01: Sleep mode 10: Prohibit 11: Operation mode	2'b00
RENDER STATE REGISTER (GRP3D_RENDERSTATE) ***Address : C001_A028h***				
[31 :15]	-	**RESERVED**	Reserved	-
[14]	W	**GRP3D_ALPHATESTENB**	Alpha test enable	-
[13]	W	**GRP3D_CULLDIR**	Specifies how back-facing triangles are culled. 1: CW (Cull back faces with clockwise vertices) 0: CCW If GRP3D_CULLENB is 0, this bit is ignored	-
[12]	W	**GRP3D_CULLENB**	Back-face culling enable. 1: Cull back faces 0: Do not cull back faces	-
[11]	W	**GRP3D_DITHERENB**	Dithering Enable 0 : Disable 1 : Enable	-
[10]	W	**GRP3D_FOGENB**	Fog Blending Enable 0 : Disable 1 : Enable	-

Bit	R/W	Symbol	Description	Reset Value
[9]	W	**GRP3D_ZWENB**	Z-Buffer Write Enable 0 : Disable 1 : Enable	-
[8]	W	**GRP3D_ZENB**	Z-Buffer Test Enable 0 : Disable 1 : Enable	-
[7]	W	**GRP3D_ALPHAENB**	Alpha Blending Enable 0 : Disable 1 : Enable	-
[6]	W	**GRP3D_AFILTERENB**	Adaptive filter-cache enable Adaptive filter-cache enhances Rasterizer performance and slightly lowers Bilinear filtering quality. If GRP3D_FILTERENB is 0, this bit is ignored	-
[5]	W	**GRP3D_FILTERENB**	Bilinear filtering enable If GRP3D_TEXENB is 0, this bit must be 0	-
[4]	W	**GRP3D_SHADEENB**	Shade Enable Reduce Rasterizer power when vertex color is not used. If any GRP3D_TBLENDARGx is 1, this bit must be 1.	-
[3]	W	**GRP3D_TPENB**	Transparency color checking enable. If GRP3D_TEXENB is 0, this bit must be 0	-
[2]	W	**GRP3D_MTEXENB**	Multi-texturing enable If GRP3D_TEXENB is 0, this bit must be 0	-
[1]	W	**GRP3D_TEXENB**	Texture Mapping Enable 0 : Disable 1 : Enable	-
[0]	W	**GRP3D_PCENB**	Perspective correction enable If GRP3D_TEXENB is 0, this bit must be 0	-
ALPHA BLENDING FUNCTION REGISTER (GRP3D_ALPHABLEND) ***Address : C001_A030h***				
[31:10]	-	**RESERVED**	Reserved	-
[9 : 5]	W	**GRP3D_DESTBLEND**	Destination blending function 01h: Zero (00, 00, 00) 11h: One (FF, FF, FF) 02h: Source color (Rs, Gs, Bs) 12h: Inverse source color (~Rs,~Gs,~Bs) 04h: Source alpha (As, As, As) 14h: Inverse Source alpha (~As,~As,~As) 08h: Destination color (Rd, Gd, Bd) 18h: Inverse Destination color (~Rd,~Gd,~Bd)	-
[4 : 0]	W	**GRP3D_SRCBLEND**	Source Blending Function	-
TEXTURE SEGMENT REGISTER (GRP3D_TEXSUBSEGMENT) ***Address : C001_A038h***				
[31:27]	-	**RESERVED**	Reserved	-
[26 : 16]	W	**GRP3D_TEXSUBSEGMENT1**	Texture/Palette sub-segment. 1 If GRP3D_TEXSUBSEGMENT is changed, texture cache must be cleared.	-
[15:11]	-	**RESERVED**	Reserved	-
[10 : 0]	W	**GRP3D_TEXSUBSEGMENT0**	Texture/Palette sub-segment. 0 If GRP3D_TEXSUBSEGMENT is changed, texture cache must be cleared.	-
TRANSPARENCY COLOR REGISTER (GRP3D_TPCOLOR) ***Address : C001_A040h***				
[31:16]	-	**RESERVED**	Reserved	-

Bit	R/W	Symbol	Description	Reset Value
[15 : 0]	W	**GRP3D_TPCOLOR**	Transparency Color (R5G6B5)	-
MIPMAP BIAS REGISTER (GRP3D_MIPMAPBIAS) ***Address : C001_A044h***				
[31 :16]	-	**RESERVED**	Reserved	-
[15 : 0]	W	**GRP3D_MIPMAPBIAS**	Mipmap bias Resulting mipmap level = original mipmap level + bias Fixed point value : sign.7.8 If this value is positive(+), mipmap level is increased If this value is negative(-), mipmap level is decreased (Mipmap level '0' indicates using original texture.)	-
UPDATE LUT PARAMETER REGISTER (GRP3D_LUTPARAM) ***Address : C001_A048h***				
[31 :27]	-	**RESERVED**	Reserved	-
[26 : 25]	W	**GRP3D_LUTPAGE**	LSB 2bit of LUT page index in texture sub-segment (GRP3D_TEXSUBSEGMENT)	-
[24 : 21]	W	**GRP3D_LUTBANK**	LUT bank for 4bit-indexed If GRP3D_LUTFULL is '1', these bits are ignored.	-
[20 : 18]	-	**RESERVED**	Reserved	-
[17]	W	**GRP3D_LUTFULL**	LUT full update flag If this bit is 1, update all color tables (256 color). If this bit is 0, update only 1-bank (16 color).	-
[16]	W	**GRP3D_LUTSTAGE**	LUT stage number All texture stage has one LUT (LUT is R5G6B5 256 color table) this bit indicatesa texture stage number	-
[15 : 8]	W	**GRP3D_LUTPAGEY**	The y-coordinate of the upper-left corner of a piece of LUT data in a page GRP3D_LUTPAGEY = 16bpp pixel coordinate / 2	-
[7 : 0]	W	**GRP3D_LUTPAGEX**	The x-coordinate of the upper-left corner of a piece of LUT data in a page GRP3D_LUTPAGEX = 16bpp pixel coordinate / 2 GRP3D_LUTPAGEX must be a multiple of 4 (0 == GRP3D_LUTPAGEX[1:0])	-
<Note> LUT will be automatically updated when CMDID_LUTFILL is processed by the Command processor				
TEXTURE STAGE 0 INFORMATION REGISTER 0 (GRP3D_TEXINFO0_0) ***Address : C001_A050h***				
[31 :28]	-	**RESERVED**	Reserved	-
[27]	W	**GRP3D_TEXSEG0**	Segment selector. If this is 0, segment of texture stage 0 is described by GTP3D_TEXSUBSEGMENT0 If this is 1, segment of texture stage 0 is described by GTP3D_TEXSUBSEGMENT1	-
[26 : 25]	W	**GRP3D_TEXPAGE0**	LSB 2bit of Texture page index in texture sub-segment (GRP3D_TEXSUBSEGMENT)	-
[24 : 19]	W	**GRP3D_COLORMODE0**	Color mode	-

Color mode	Description	Associated Open GL|ES format
0	4bit indexed R5G6B5	GL_PALETTE4_R5_G6_B5_OES
1	4bit indexed R4G4B4A4	GL_PALETTE4_RGBA4_OES
2	4bit indexed R5G5B5A1	GL_PALETTE4_RGB5_A1_OES
3	4bit indexed A8L8	
4	4bit indexed A4R4G4B4	
5	4bit indexed A1R5G5B5	

Bit	R/W	Symbol	Description	Reset Value
			6 | 4bit indexed L8A8 | 9 | A4 | 15 | L4 | 16 | 8bit indexed R5G6B5 | GL_PALETTE8_R5_G6_B5_OES 17 | 8bit indexed R4G4B4A4 | GL_PALETTE8_RGBA4_OES 18 | 8bit indexed R5G5B5A1 | GL_PALETTE8_RGB5_A1_OES 19 | 8bit indexed A8L8 | 20 | 8bit indexed A4R4G4B4 | 21 | 8bit indexed A1R5G5B5 | 22 | 8bit indexed L8A8 | 25 | L4A4 | 27 | L8 | GL_LUMINANCE 30 | A8 | GL_ALPHA 31 | A4L4 | 40 | R5G6B5 | GL_UNSIGNED_SHORT_5_6_5 41 | R4G4B4A4 | GL_UNSIGNED_SHORT_4_4_4_4 42 | R5G5B5A1 | GL_UNSIGNED_SHORT_5_5_5_1 43 | A8L8 | GL_LUMINANCE_ALPHA 44 | A4R4G4B4 | 45 | A1R5G5B5 | 46 | L8A8 |	
[18]	W	**GRP3D_MIPMAPENB0**	Mipmap enable If GRP3D_COLORMODE0 is not 2, this bit must be 0	-
[17]	-	**RESERVED**	Reserved	-
[16]	W	**GRP3D_TILEINDEX0**	Tile-index flag If this bit is '1', this texture is a tile-index map.	-
[15 : 2]	-	**RESERVED**	Reserved	-
[1]	W	**GRP3D_TEXADDRMODEU0**	Texture addressing mode of V(vertical) direction 0 : Wrap Texture Address Mode. Repeats the texture. 1 : Clamp Texture Address Mode. It applies the texture once, then smears the color of edge pixels.	-
[0]	W	**GRP3D_TEXADDRMODEV0**	Texture addressing mode of U(horizontal) direction 0 : Wrap Texture Address Mode. Repeats the texture. 1 : Clamp Texture Address Mode. It applies the texture once, then smears the color of edge pixels.	-
TEXTURE STAGE 0 INFORMATION REGISTER 1 (GRP3D_TEXINFO0_1) ***Address : C001_A054h***				
[31 :24]	-	**GRP3D_TEXMASKY0**	Texture height parameter GRP3D_TEXIMASKY0 = (texture_height / 2) – 1	-
[23 : 16]	W	**GRP3D_TEXMASKX0**	Texture width parameter GRP3D_TEXIMASKX0 = (texture_width * bits_per_pixel / 32) – 1 bits_per_pixel = 4, 8 or 16 If GRP3D_TILEINDEX0 is 1, bits_per_pixel is treated as 16.	-
[15 : 8]	W	**GRP3D_TEXPAGEY0**	The y-coordinate of the upper-left corner of texture in a page GRP3D_TEXPAGEY0 = 16bpp pixel coordinate / 2	-

<table>
<tr><th>Bit</th><th>R/W</th><th>Symbol</th><th>Description</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td></td><td>GRP3D_TEXPAGEY0 must be a multiple of (texture_height / 2)</td><td></td></tr>
<tr><td>[7 : 0]</td><td>W</td><td>GRP3D_TEXPAGEX0</td><td>The x-coordinate of the upper-left corner of texture in a page
GRP3D_TEXPAGEX0 = 16bpp pixel coordinate / 2
GRP3D_TEXPAGEX0 must be a multiple of (texture_width * bits_per_pixel / 32)</td><td>-</td></tr>
<tr><td colspan="5"><Note> All textures must have widths and heights specified as powers of 2</td></tr>
<tr><td colspan="5">TEXTURE STAGE 1 INFORMATION REGISTER 0 (GRP3D_TEXINFO1_0)
Address : C001_A058h</td></tr>
<tr><td>[31 :28]</td><td>-</td><td>RESERVED</td><td>Reserved</td><td>-</td></tr>
<tr><td>[27]</td><td>W</td><td>GRP3D_TEXSEG1</td><td>Segment selector.
If this is 0, segment of texture stage 1 is described by GTP3D_TEXSUBSEGMENT0
If this is 1, segment of texture stage 1 is described by GTP3D_TEXSUBSEGMENT1</td><td>-</td></tr>
<tr><td>[26 : 25]</td><td>W</td><td>GRP3D_TEXPAGE1</td><td>LSB 2bit of Texture page index in texture sub-segment (GRP3D_TEXSUBSEGMENT)</td><td>-</td></tr>
<tr><td>[24 : 19]</td><td>W</td><td>GRP3D_COLORMODE1</td><td>Color mode
<table>
<tr><th>Color mode</th><th>Description</th><th>Associated Open GL|ES format</th></tr>
<tr><td>0</td><td>4bit indexed R5G6B5</td><td>GL_PALETTE4_R5_G6_B5_OES</td></tr>
<tr><td>1</td><td>4bit indexed R4G4B4A4</td><td>GL_PALETTE4_RGBA4_OES</td></tr>
<tr><td>2</td><td>4bit indexed R5G5B5A1</td><td>GL_PALETTE4_RGB5_A1_OES</td></tr>
<tr><td>3</td><td>4bit indexed A8L8</td><td></td></tr>
<tr><td>4</td><td>4bit indexed A4R4G4B4</td><td></td></tr>
<tr><td>5</td><td>4bit indexed A1R5G5B5</td><td></td></tr>
<tr><td>6</td><td>4bit indexed L8A8</td><td></td></tr>
<tr><td>9</td><td>A4</td><td></td></tr>
<tr><td>15</td><td>L4</td><td></td></tr>
<tr><td>16</td><td>8bit indexed R5G6B5</td><td>GL_PALETTE8_R5_G6_B5_OES</td></tr>
<tr><td>17</td><td>8bit indexed R4G4B4A4</td><td>GL_PALETTE8_RGBA4_OES</td></tr>
<tr><td>18</td><td>8bit indexed R5G5B5A1</td><td>GL_PALETTE8_RGB5_A1_OES</td></tr>
<tr><td>19</td><td>8bit indexed A8L8</td><td></td></tr>
<tr><td>20</td><td>8bit indexed A4R4G4B4</td><td></td></tr>
<tr><td>21</td><td>8bit indexed A1R5G5B5</td><td></td></tr>
<tr><td>22</td><td>8bit indexed L8A8</td><td></td></tr>
<tr><td>25</td><td>L4A4</td><td></td></tr>
<tr><td>27</td><td>L8</td><td>GL_LUMINANCE</td></tr>
<tr><td>30</td><td>A8</td><td>GL_ALPHA</td></tr>
<tr><td>31</td><td>A4L4</td><td></td></tr>
<tr><td>40</td><td>R5G6B5</td><td>GL_UNSIGNED_SHORT_5_6_5</td></tr>
<tr><td>41</td><td>R4G4B4A4</td><td>GL_UNSIGNED_SHORT_4_4_4_4</td></tr>
<tr><td>42</td><td>R5G5B5A1</td><td>GL_UNSIGNED_SHORT_5_5_5_1</td></tr>
<tr><td>43</td><td>A8L8</td><td>GL_LUMINANCE_ALPHA</td></tr>
<tr><td>44</td><td>A4R4G4B4</td><td></td></tr>
<tr><td>45</td><td>A1R5G5B5</td><td></td></tr>
<tr><td>46</td><td>L8A8</td><td></td></tr>
</table></td><td>-</td></tr>
<tr><td>[18]</td><td>W</td><td>GRP3D_MIPMAPENB1</td><td>Mipmap enable</td><td>-</td></tr>
</table>

Bit	R/W	Symbol	Description	Reset Value
			If GRP3D_COLORMODE1 is not 2, this bit must be 0	
[17]	-	**RESERVED**	Reserved	-
[16]	W	**GRP3D_TILEINDEX1**	Tile-index flag If this bit is 1, this texture is tile-index map.	-
[15 : 2]	-	**RESERVED**	Reserved	-
[1]	W	**GRP3D_TEXADDRMODEU1**	Texture addressing mode of V(vertical) direction 0 : Wrap Texture Address Mode. Repeats the texture. 1 : Clamp Texture Address Mode. It applies the texture once, then smears the color of edge pixels.	-
[0]	W	**GRP3D_TEXADDRMODEV1**	Texture addressing mode of U(horizontal) direction 0 : Wrap Texture Address Mode. Repeats the texture. 1 : Clamp Texture Address Mode. It applies the texture once, then smears the color of edge pixels.	-
TEXTURE STAGE 1 INFORMATION REGISTER 1 (GRP3D_TEXINFO1_1) ***Address : C001_A05Ch***				
[31 :24]	W	**GRP3D_TEXMASKY1**	Texture height parameter GRP3D_TEXMASKY1 = (texture_height / 2) – 1	-
[23 : 16]	W	**GRP3D_TEXMASKX1**	Texture width parameter GRP3D_TEXMASKX1 = (texture_width * bits_per_pixel / 32) – 1 bits_per_pixel = 4, 8 or 16	-
[15 : 8]	W	**GRP3D_TEXPAGEY1**	The y-coordinate of the upper-left corner of texture in a page GRP3D_TEXPAGEY1 = 16bpp pixel coordinate / 2 GRP3D_TEXPAGEY1 must be a multiple of (texture_height / 2)	-
[7 : 0]	W	**GRP3D_TEXPAGEX1**	The x-coordinate of the upper-left corner of texture in a page GRP3D_TEXPAGEX1 = 16bpp pixel coordinate / 2 GRP3D_TEXPAGEX1 must be a multiple of (texture_width * bits_per_pixel / 32)	-
<Note> All textures must have widths and heights specified as powers of 2				
TEXTURE STAGE 0 BLENDING FUNCTION REGISTER (GRP3D_TEXIBLEND0) ***Address : C001_A060h***				
[31:28]	-	**RESERVED**	Reserved.	-
[27:24]	W	**GRP3D_TB0_RGBARG0**	Texture blending RGB Arg0 0 : Cs 1 : 1-Cs 2 : As 3 : 1-As 4 : Cc 5 : 1-Cc 6 : Ac 7 : 1-Ac 8 : Cf 9 : 1-Cf 10 : Af 11 : 1-Af 12 : Cs 13 : Reserved 14 : As 15 : Reserved Cs,As : source color (texture color) Cc,Ac : constant color (refer to GRP3D_TEXCONST) Cf,Af : fragment color (vertex color)	-
[23:20]	W	**GRP3D_TB0_RGBARG1**	Texture blending RGB Arg1 0 : Cs 1 : 1-Cs 2 : As 3 : 1-As 4 : Cc 5 : 1-Cc 6 : Ac 7 : 1-Ac 8 : Cf 9 : 1-Cf 10 : Af 11 : 1-Af 12 : Cs 13 : Reserved 14 : As 15 : Reserved	-
[19:16]	W	**GRP3D_TB0_RGBARG2**	Texture blending RGB Arg2 0 : Cs 1 : 1-Cs 2 : As 3 : 1-As 4 : Cc 5 : 1-Cc 6 : Ac 7 : 1-Ac 8 : Reserved 9 : Reserved 10 : Reserved 11 : Reserved 12 : Reserved 13 : Reserved 14 : Reserved 15 : Reserved	-
[15:12]	W	**GRP3D_TB0_RGBOP**	Texture blending RGB operation 0: Arg0 1: Arg0 * Arg1	-

Bit	R/W	Symbol	Description	Reset Value
			2: Arg0 + Arg1 3: Arg0 + Arg1 – 0.5 4: Arg0 * Arg2 + Arg1 * (1- Arg2) 5: Arg0 – Arg1 6: 4 * ((Arg0.r – 0.5)*(Arg1.r – 0.5) + (Arg0.g – 0.5)*(Arg1.g – 0.5) + (Arg0.b – 0.5)*(Arg1.b – 0.5)) 7: 4 * ((Arg0.r – 0.5)*(Arg1.r – 0.5) + (Arg0.g – 0.5)*(Arg1.g – 0.5) + (Arg0.b – 0.5)*(Arg1.b – 0.5)) If it is 7, GRP3D_TB0_AOP is ignored and result of alpha is generated from GRP3D_TB0_RGBOP	
[11:9]	W	**GRP3D_TB0_AARG0**	Texture blending Alpha Arg0 0 : As 1 : 1-As 2 : Ac 3 : 1-Ac 4 : Af 5 : 1-Af 6 : As 7 : Reserved	-
[8:6]	W	**GRP3D_TB0_AARG1**	Texture blending Alpha Arg1 0 : As 1 : 1-As 2 : Ac 3 : 1-Ac 4 : Af 5 : 1-Af 6 : As 7 : Reserved	-
[5:3]	W	**GRP3D_TB0_AARG2**	Texture blending Alpha Arg2 0 : As 1 : 1-As 2 : Ac 3 : 1-Ac 4 : Af 5 : 1-Af 6 : As 7 : Reserved	-
[2:0]	W	**GRP3D_TB0_AOP**	Texture blending Alpha operation 0: Arg0 1: Arg0 * Arg1 2: Arg0 + Arg1 3: Arg0 + Arg1 – 0.5 4: Arg0 * Arg2 + Arg1 * (1- Arg2) 5: Arg0 – Arg1	-
TEXTURE STAGE 1 BLENDING FUNCTION REGISTER (GRP3D_TEXIBLEND1) ***Address : C001_A068h***				
[31:28]	-	**RESERVED**	Reserved.	-
[27:24]	W	**GRP3D_TB1_RGBARG0**	Texture blending RGB Arg0 0 : Cs 1 : 1-Cs 2 : As 3 : 1-As 4 : Cc 5 : 1-Cc 6 : Ac 7 : 1-Ac 8 : Cf 9 : 1-Cf 10 : Af 11 : 1-Af 12 : Cp 13 : Reserved 14 : Ap 15 : Reserved Cs,As : source color (texture color) Cc,Ac : constant color (refer to GRP3D_TEXCONST) Cf,Af : fragment color (vertex color) Cp,Ap: result color from previous texture blending stage.	-
[23:20]	W	**GRP3D_TB1_RGBARG1**	Texture blending RGB Arg1 0 : Cs 1 : 1-Cs 2 : As 3 : 1-As 4 : Cc 5 : 1-Cc 6 : Ac 7 : 1-Ac 8 : Cf 9 : 1-Cf 10 : Af 11 : 1-Af 12 : Cp 13 : Reserved 14 : Ap 15 : Reserved	-
[19:16]	W	**GRP3D_TB1_RGBARG2**	Texture blending RGB Arg2 0 : Cs 1 : 1-Cs 2 : As 3 : 1-As 4 : Cc 5 : 1-Cc 6 : Ac 7 : 1-Ac 8 : Reserved 9 : Reserved 10 : Reserved 11 : Reserved 12 : Reserved 13 : Reserved 14 : Reserved 15 : Reserved	-
[15:12]	W	**GRP3D_TB1_RGBOP**	Texture blending RGB operation 0: Arg0 1: Arg0 * Arg1	-

Bit	R/W	Symbol	Description	Reset Value
			2: Arg0 + Arg1 3: Arg0 + Arg1 – 0.5 4: Arg0 * Arg2 + Arg1 * (1- Arg2) 5: Arg0 – Arg1 6: 4 * ((Arg0.r – 0.5)*(Arg1.r – 0.5) + (Arg0.g – 0.5)*(Arg1.g – 0.5) + (Arg0.b – 0.5)*(Arg1.b – 0.5)) 7: 4 * ((Arg0.r – 0.5)*(Arg1.r – 0.5) + (Arg0.g – 0.5)*(Arg1.g – 0.5) + (Arg0.b – 0.5)*(Arg1.b – 0.5)) If it is 7, GRP3D_TB0_AOP is ignored and result of alpha is generated from GRP3D_TB0_RGBOP	
[11:9]	W	**GRP3D_TB1_AARG0**	Texture blending Alpha Arg0 0 : As 1 : 1-As 2 : Ac 3 : 1-Ac 4 : Af 5 : 1-Af 6 : Ap 7 : Reserved	-
[8:6]	W	**GRP3D_TB1_AARG1**	Texture blending Alpha Arg1 0 : As 1 : 1-As 2 : Ac 3 : 1-Ac 4 : Af 5 : 1-Af 6 : Ap 7 : Reserved	-
[5:3]	W	**GRP3D_TB1_AARG2**	Texture blending Alpha Arg2 0 : As 1 : 1-As 2 : Ac 3 : 1-Ac 4 : Af 5 : 1-Af 6 : Ap 7 : Reserved	-
[2:0]	W	**GRP3D_TB1_AOP**	Texture blending Alpha operation 0: Arg0 1: Arg0 * Arg1 2: Arg0 + Arg1 3: Arg0 + Arg1 – 0.5 4: Arg0 * Arg2 + Arg1 * (1- Arg2) 5: Arg0 – Arg1	-
RESERVED ***Address : C001_A070h~ C001_A087h***				
ALPHA TEST REGISTER (GRP3D_ALPHATEST) ***Address : C001_A088h***				
[31 : 16]	-	**RESERVED**	Reserved	-
[15:8]	W	**GRP3D_ALPHAREF**	Alpha reference value.	-
[7:0]	W	**GRP3D_ALPHAFUNC**	Alpha test function. 0 : NEVER Always fail the test 1 : LESS Accept the new pixel if its value is less than the value of the current pixel 2 : EQUAL Accept the new pixel if its value equals the value of the current pixel 3 : LESSEQUAL Accept the new pixel if its value is less than or equal to the value of the current pixel 4 : GREATER Accept the new pixel if its value is greater than the value of the current pixel 5 : NOT EQUAL Accept the new pixel if its value does not equal the value of the current pixel 6 : GREATER EQUAL Accept the new pixel if its value is greater than or equal to the value of the current pixel 7 : ALWAYS	-
TEXTURE CONST COLOR REGISTER (GRP3D_TEXCONST)				

Bit	R/W	Symbol	Description	Reset Value
Address : C001_A08Ch				
[31:24]	W	**GRP3D_TEXCONST_A**	Alpha component of texture constant color	-
[23:16]	W	**GRP3D_TEXCONST_R**	Red component of texture constant color	
[15:8]	W	**GRP3D_TEXCONST_G**	Green component of texture constant color	
[7:0]	W	**GRP3D_TEXCONST_B**	Blue component of texture constant color	
FOG COLOR REGISTER (GRP3D_FOGCOLOR) **Address : C001_A090h**				
[31 : 24]	-	**RESERVED**	Reserved	-
[23 : 0]	W	**GRP3D_FOGCOLOR**	Fog Color R8G8B8	-
FOG MAX Z REGISTER (GRP3D_FOGMAXZ) ***Address : C001_A094h***				
[31 : 8]	W	**GRP3D_FOGMAXZ**	Maximum z-value of applicable region of fog table. 2-12< GRP3D_FOGMAXZ< 216 If z >= GRP3D_FOGMAXZ, fog blend parameter = last fog table value. 24bit floating point number. This register has 8bit offset. In this way, you can use a 32bit floating point number, like this…(example in C) float Fog_max_z; *((volatile unsigned long*)(0xE0000094)) = (*(unsigned long*)&(Fog_max_z))	-
[7 : 0]	-	**RESERVED**	Reserved	-
DISPLAY INFORMATION REGISTER (GRP3D_DISPINFO) ***Address : C001_A098h***				
[31 : 28]	-	**RESERVED**	Reserved	-
[27]	W	**GRP3D_DISPAA**	Display Antialiasing Flag	-
[26 : 16]	W	**GRP3D_DISPSUBSEGMENT**	Display Sub-Segment	-
[15 : 8]	W	**GRP3D_DISPTBLY**	Display y-coordinate conversion table If Ydisplay [9:8] == 00, Ymemory[9:0] = { GRP3D_DISPTBLY[1:0] , Ydisplay[7:0]} If Ydisplay [9:8] == 01, Ymemory[9:0] = { GRP3D_DISPTBLY[3:2] , Ydisplay[7:0]} If Ydisplay [9:8] == 10, Ymemory[9:0] = { GRP3D_DISPTBLY[5:4] , Ydisplay[7:0]} If Ydisplay [9:8] == 11, Ymemory[9:0] = { GRP3D_DISPTBLY[7:6] , Ydisplay[7:0]}	-
[7 : 0]	W	**GRP3D_DISPTBLX**	Display x-coordinate conversion table If Xdisplay [9:8] == 00, Xmemory[9:0] = { GRP3D_DISPTBLX[1:0] , Xdisplay[7:0]} If Xdisplay [9:8] == 01, Xmemory[9:0] = { GRP3D_DISPTBLX[3:2] , Xdisplay[7:0]} If Xdisplay [9:8] == 10, Xmemory[9:0] = { GRP3D_DISPTBLX[5:4] , Xdisplay[7:0]} If Xdisplay [9:8] == 11, Xmemory[9:0] = { GRP3D_DISPTBLX[7:6] , Xdisplay[7:0]}	-
MEMORY FILL PARAMETER REGISTER 0 (GRP3D_MFILLPARAM0) ***Address : C001_A0A0h***				
[31 : 28]	-	**RESERVED**	Reserved	-
[27]	W	**GRP3D_MFADDRMODE**	Memory fill addressing mode 0 : display address mode 1 : texture address mode	-
[26 : 16]	W	**GRP3D_MFSUBSEGMENT**	Memory Fill Sub_Segment	-
[15 : 8]	W	**GRP3D_MFDATA**	Memory Fill Data	-
MEMORY FILL PARAMETER REGISTER 1 (GRP3D_MFILLPARAM1) ***Address : C001_A0A4h***				
[31 : 26]	-	**RESERVED**	Reserved	-

Bit	R/W	Symbol	Description	Reset Value
[25 : 16]	W	**GRP3D_MFTOP**	The y-coordinate of the upper-left corner of a rectangle in a Memory fill sub-segment.	-
[15 : 10]	-	**RESERVED**	Reserved	-
[9 : 0]	W	**GRP3D_MFLEFT**	The x-coordinate of the upper-left corner of a rectangle in a Memory fill sub-segment.	-
MEMORY FILL PARAMETER REGISTER 2 (GRP3D_MFILLPARAM2) ***Address : C001_A0A8h***				
[31 : 26]	-	**RESERVED**	Reserved	-
[25 : 16]	W	**GRP3D_MFBOTTOM**	The y-coordinate of the lower-right corner of a rectangle in a Memory fill sub-segment GRP3D_MFTOP <= Y range <= GRP3D_MFBOTTOM	-
[15 : 10]	-	**RESERVED**	Reserved	-
[9 : 0]	W	**GRP3D_MFRIGHT**	The x-coordinate of the lower-right corner of a rectangle in a Memory fill sub-segment GRP3D_MFLEFT <= X range <= GRP3D_MFRIGHT	-
RESERVED ***Address : C001_A0ACh***				
RENDER TARGET REGISTER 0 (GRP3D_RENDTRG0) ***Address : C001_A0B0h***				
[31 : 29]	-	**RESERVED**	Reserved	-
[25 : 16]	W	**GRP3D_RTAA**	Render target antialiasing flag	-
[27]	W	**GRP3D_RTADDRMODE**	Render target addressing mode 0 : display address mode 1: texture address mode	-
[26 : 16]	W	**GRP3D_RTSUBSEGMENT**	Render target sub-segment	-
[15 : 8]	W	**GRP3D_RTTBLY**	Rasterizer y-coordinate conversion table If Yrasterizer [9:8] == 00, Ymemory[9:0] = { GRP3D_RTTBLY[1:0] , Yrasterizer [7:0]} If Yrasterizer [9:8] == 01, Ymemory[9:0] = { GRP3D_RTTBLY[3:2] , Yrasterizer [7:0]} If Yrasterizer [9:8] == 10, Ymemory[9:0] = { GRP3D_RTTBLY[5:4] , Yrasterizer [7:0]} If Yrasterizer [9:8] == 11, Ymemory[9:0] = { GRP3D_RTTBLY[7:6] , Yrasterizer [7:0]}	-
[7 : 0]	W	**GRP3D_RTTBLX**	Rasterizer x-coordinate conversion table If Xrasterizer [9:8] == 0, Xmemory[9:0] = { GRP3D_RTTBLX[1:0] , Xrasterizer [7:0]} If Xrasterizer [9:8] == 1, Xmemory[9:0] = { GRP3D_RTTBLX[3:2] , Xrasterizer [7:0]} If Xrasterizer [9:8] == 2, Xmemory[9:0] = { GRP3D_RTTBLX[5:4] , Xrasterizer [7:0]} If Xrasterizer [9:8] == 3, Xmemory[9:0] = { GRP3D_RTTBLX[7:6] , Xrasterizer [7:0]}	-
RENDER TARGET REGISTER 1 (GRP3D_RENDTRG1) ***Address : C001_A0B4h***				
[31 : 26]	-	**RESERVED**	Reserved	-
[25 : 16]	W	**GRP3D_RTTOP**	The y-coordinate of the upper-left corner of a rectangle in a Render target sub-segment	-
[15 : 11]	-	**RESERVED**	Reserved	-
[10 : 0]	W	**GRP3D_RTLEFT**	The x-coordinate of the upper-left corner of a rectangle in a Render target sub-segment	-
RENDER TARGET REGISTER 2 (GRP3D_RENDTRG2) ***Address : C001_A0B8h***				
[31 : 26]	-	**RESERVED**	Reserved	-
[25 : 16]	W	**GRP3D_RTBOTTOM**	The y-coordinate of the lower-right corner of a rectangle in a Render target sub-segment GRP3D_RTTOP <= Y range <= GRP3D_RTBOTTOM	-
[15 : 11]	-	**RESERVED**	Reserved	-
[10 : 0]	W	**GRP3D_RTRIGTH**	The x-coordinate of the lower-right corner of a rectangle in a Render target sub-segment GRP3D_RTLEFT <= X range <= GRP3D_RTRIGHT	-

Bit	R/W	Symbol	Description	Reset Value
RESERVED ***Address : C001_A0BCh***				
Z BUFFER INFORMATION REGISTER (GRP3D_ZBINFO) ***Address : C001_A0C0h***				
[31]	-	**RESERVED**	Reserved	-
[30 : 28]	W	**GRP3D_ZBFUNC**	Z-buffer test function 000 : NEVER Always fail the test 001 : LESS Accept the new pixel if its value is less than the value of the current pixel 010 : EQUAL Accept the new pixel if its value equals the value of the current pixel 011 : LESSEQUAL Accept the new pixel if its value is less than or equal to the value of the current pixel 100 : GREATER Accept the new pixel if its value is greater than the value of the current pixel 101 : NOT EQUAL Accept the new pixel if its value does not equal the value of the current pixel 110 : GREATER EQUAL Accept the new pixel if its value is greater than or equal to the value of the current pixel 111 : ALWAYS Always pass the test	-
[27]	W	**GRP3D_ZBADDRMODE**	Z-buffer address mode 0: Display Address Mode 1: Texture Address Mode	-
[26 : 16]	W	**GRP3D_ZBSUBSEGMENT**	Z-buffer sub-segment	-
[15 : 8]	W	**GRP3D_ZBTBLY**	Rasterizer y-coordinate conversion table If Yrasterizer [9:8] == 00, Ymemory[9:0] = { GRP3D_ZBTBLY[1:0] , Yrasterizer [7:0]} If Yrasterizer [9:8] == 01, Ymemory[9:0] = { GRP3D_ZBTBLY[3:2] , Yrasterizer [7:0]} If Yrasterizer [9:8] == 10, Ymemory[9:0] = { GRP3D_ZBTBLY[5:4] , Yrasterizer [7:0]} If Yrasterizer [9:8] == 11, Ymemory[9:0] = { GRP3D_ZBTBLY[7:6] , Yrasterizer [7:0]}	
[7 : 0]	W	**GRP3D_ZBTBLX**	Rasterizer x-coordinate conversion table If Xrasterizer [9:8] == 00, Xmemory[9:0] = { GRP3D_ZBTBLX[1:0] , Xrasterizer [7:0]} If Xrasterizer [9:8] == 01, Xmemory[9:0] = { GRP3D_ZBTBLX[3:2] , Xrasterizer [7:0]} If Xrasterizer [9:8] == 10, Xmemory[9:0] = { GRP3D_ZBTBLX[5:4] , Xrasterizer [7:0]} If Xrasterizer [9:8] == 11, Xmemory[9:0] = { GRP3D_ZBTBLX[7:6] , Xrasterizer [7:0]}	
Z-VALID BUFFER INFORMATION REGISTER (GRP3D_ZVBINFO) ***Address : C001_A0C4h***				
[31 : 25]	-	**RESERVED**	Reserved	-
[24]	W	**GRP3D_ZVBENB**	Z-valid-buffer enable Z-valid buffer is a 64x128 pixel (16bit) (rectangle area) in a Z-buffer sub-segment. Z-valid buffer is for reducing Z-buffer access. It has 1valid bit per 8 pixels.	-
[23 : 20]	W	**GRP3D_ZVBTOP**	The y-coordinate of the upper-left corner of a Z-valid-buffer in a Z-buffer sub-segment GRP3D_ZVBTOP must be multiple of 2 (0==GRP3D_ZVBTOP[0]) 16bpp Y position in sub-segment = { GRP3D_ZVBTOP[3:1], 7'b000_000 }	-
[19 : 16]	W	**GRP3D_ZVBLEFT**	The x-coordinate of the upper-left corner of a Z-valid-buffer in aZ-buffer sub-segment 16bpp X position in sub-segment = { GRP3D_ZVBTOP[3:0], 6'b00_000 }	-
[15 : 0]	W	**GRP3D_ZVVALUE**	Z-valid-buffer invalid value If Z-valid-buffer bit is 0, destination z value is treated as GRP3D_ZVVALUE (Normally FFFFh)	

Bit	R/W	Symbol	Description	Reset Value
Z SCALE REGISTER (GRP3D_ZSCALE) ***Address : C001_A0C8h***				
[31 : 8]	W	**GRP3D_ZSCALE**	Z scale value TSE multiplies a projected z-value by GRP3D_ZSCALE to make the z-value 0~0ffff, because the A Projected z-value (TSE input z) is 0.0f ~ 1.0f 24bit floating point number. Although the register format is 24bit floating point, the user does not need to convert 32bit floating point to 24bit floating point ,as below. float Zscale = 65534.0f; *((volatile unsigned long*)(0xE00000C8)) = (*(unsigned long*)&(Zscale))	-
[7 : 0]	-	**RESERVED**	Reserved	-
RESERVED ***Address : C001_A0D0D0h ~A0EFh***				
VIEWPORT BOTTOM REGISTER (GRP3D_VPORTBOT) ***Address : C001_A0F0h***				
[31 : 8]	W	**GRP3D_VPORTBOT**	The bottom of the viewport 24bit floating point number. This register has 8bit offset. In this way, you can use a 32bit floating point number, like this…(example in C) float Bottom = viewport.y + viewport.height; *((volatile unsigned long*)(0xE00000F0)) = (*(unsigned long*)&(Bottom))	-
[7 : 0]	-	**RESERVED**	Reserved	-
VIEWPORT HEIGHT REGISTER (GRP3D_VPORTH) ***Address : C001_A0F4h***				
[31 : 8]	W	**GRP3D_VPORTH**	The height of the viewport 24bit floating point number. This register has 8bit offset. In this way, you can use a 32bit floating point number, like this…(example in C) float Height = viewport.height; *((volatile unsigned long*)(0xE00000F4)) = (*(unsigned long*)&(Height))	-
[7 : 0]	-	**RESERVED**	Reserved	-
VIEWPORT LEFT REGISTER (GRP3D_VPORTLEFT) ***Address : C001_A0F8h***				
[31 : 8]	W	**GRP3D_VPORTLEFT**	The left-bound coordinate of the viewport. 24bit floating point number. This register has 8bit offset. In this way, you can use a 32bit floating point number, like this…(example in C) float Left = viewport.x; *((volatile unsigned long*)(0xE00000F8)) = (*(unsigned long*)&(Left))	-
[7 : 0]	-	**RESERVED**	Reserved	-
VIEWPORT WIDTH REGISTER (GRP3D_VPORTW) ***Address : C001_A0FCh***				
[31 : 8]	W	**GRP3D_VPORTW**	The width of the viewport 24bit floating point number. This register has 8bit offset. In this way, you can use a 32bit floating point number, like this…(example in C) float Width = viewport.width; *((volatile unsigned long*)(0xE00000FC)) = (*(unsigned long*)&(Width))	-
[7 : 0]	-	**RESERVED**	Reserved	-
PRIMITIVE CONTROL REGISTER (GRP3D_PRIMCON1) ***Address : C001_A100h***				

Bit	R/W	Symbol	Description	Reset Value
[31 :20]	–	**RESERVED**	Reserved	-
[19:18]	W	**GRP3D_LOCKIDX**	Memory burst size when index buffer is accessed. 0: burst 4 1: burst 8 2: burst 16 3: reserved	2
[17:16]	W	**GRP3D_LOCKVTX**	Memory burst size when vertex buffer is accessed. 0: burst 4 1: burst 8 2: burst 16 3: reserved	2
[15:8]	W	**GRP3D_STREAMVALID**	Stream valid.	-
[7:6]	W	**GRP3D_IDXTYPE**	Type of index buffer data. 0 : 8bit 1 : 16bit 2 : 24bit 3 : Reserved	-
[5]	-	**RESERVED**	Reserved	-
[4]	W	**GRP3D_CLRRIMCACHE**	Clear primitive cache.	-
[3]	W	**GRP3D_LOOP**	Primitive loop flag. If this value is 1, vertex processor processes the first vertex after the last vertex data.	-
[2]	W	**GRP3D_USEIDX**	Use index buffer flag. If this value is 1, GRP3D_IDXSTART and GRP3D_IDXEND are start of index buffer address and end of index buffer address. If this value is 0, GRP3D_IDXSTART and GRP3D_IDXEND are start vertex number and end vertex number.	-
[1]	W	**GRP3D_USEMPALETTE**	Use matrix palette flag. If this value is 1, vertex stream 0 is treated as matrix index buffer and vertex stream 7 is treated as matrix buffer. If this value is 0, all vertex stream are treated as vertex data buffer.	-
[0]	W	**GRP3D_USEGTE**	Use GTE flag.	-
INDEX START REGISTER (GRP3D_IDXSTART) ***Address : C001_A104h***				
[31 : 0]	W	**GRP3D_IDXSTART**	Start of index. If GRP3D_USEIDX value is 1, this is start of index buffer address. If GRP3D_USEIDX value is 0, this is start vertex number.	-
INDEX END REGISTER (GRP3D_IDXEND) ***Address : C001_A108h***				
[31 : 0]	W	**GRP3D_IDXEND**	End of index. If GRP3D_USEIDX value is 1, this is end of index buffer address. If GRP3D_USEIDX value is 0, this is end vertex number.	-
PRIMITIVE PARAM LOOPCONTROL REGISTER (GRP3D_PPARAMLOOPCTRL) ***Address : C001_A10Ch***				
[31 : 0]	W	**RESERVED**	Reserved.	-
[15:8]	W	**GRP3D_PPARAMLOOP**	Primitive parameter loop point (0~15)	-
[7:0]	W	**GRP3D_PPARAMEND**	Primitive parameter end point (0~15)	-
PRIMITIVE PARAM DATA REGISTER (GRP3D_PRARAM03) ***Address : C001_A110h***				
[31 :24]	W	**GRP3D_PPARAM3**	Primitive parameter 3.	-
[23:16]	W	**GRP3D_PPARAM2**	Primitive parameter 2.	-
[15:8]	W	**GRP3D_PPARAM1**	Primitive parameter 1.	-
[7:0]	W	**GRP3D_PPARAM0**	Primitive parameter 0. { rectangle , inverse culling, rendering, vertex number[1:0] }	-
PRIMITIVE PARAM DATA REGISTER (GRP3D_PRARAM47)				

Bit	R/W	Symbol	Description	Reset Value
Address : C001_A114h				
[31 :24]	W	**GRP3D_PPARAM7**	Primitive parameter 7.	-
[23:16]	W	**GRP3D_PPARAM6**	Primitive parameter 6.	-
[15:8]	W	**GRP3D_PPARAM5**	Primitive parameter 5.	-
[7:0]	W	**GRP3D_PPARAM4**	Primitive parameter 4. { rectangle , inverse culling, rendering, vertex number[1:0] }	-
PRIMITIVE PARAM DATA REGISTER (GRP3D_PRARAM8B) *Address : C001_A118h*				
[31 :24]	W	**GRP3D_PPARAM11**	Primitive parameter 11.	-
[23:16]	W	**GRP3D_PPARAM10**	Primitive parameter 10.	-
[15:8]	W	**GRP3D_PPARAM9**	Primitive parameter 9.	-
[7:0]	W	**GRP3D_PPARAM8**	Primitive parameter 8. { rectangle , inverse culling, rendering, vertex number[1:0] }	-
PRIMITIVE PARAM DATA REGISTER (GRP3D_PRARAMCF) *Address : C001_A11Ch*				
[31 :24]	W	**GRP3D_PPARAM15**	Primitive parameter 15.	-
[23:16]	W	**GRP3D_PPARAM14**	Primitive parameter 14.	-
[15:8]	W	**GRP3D_PPARAM13**	Primitive parameter 13.	-
[7:0]	W	**GRP3D_PPARAM12**	Primitive parameter 12. { rectangle , inverse culling, rendering, vertex number[1:0] }	-
VERTEX STREAM0 ADDRESS REGISTER (GRP3D_VSDDR0) *Address : C001_A120h*				
[31 : 0]	W	**GRP3D_VSADDR0**	Vertex stream 0 address.	-
VERTEX STREAM0 STRIDE REGISTER (GRP3D_VSSTRIDE0) *Address : C001_A124h*				
[31 : 0]	W	**GRP3D_VSSTRIDE0**	Vertex stream 0 stride(number of bytes in a vertex)	-
VERTEX STREAM0 PARAMETER REGISTER (GRP3D_VSPARAM0) *Address : C001_A128h*				
[31 :20]	W	**GRP3D_ VSDST0**	Vertex stream 0 destination offset of GTE/TSE register. (unit: 32bit)	-
[19 : 8]	W	**GRP3D_VSSIZE0**	Vertex stream 0 size. (N-1, N is number of items in a vertex)	-
[7: 0]	W	**GRP3D_VSTYPE0**	Vertex stream 0 type. It describes number format of vertex stream data.	-
VERTEX STREAM1 ADDRESS REGISTER (GRP3D_VSDDR1) *Address : C001_A12Ch*				

GRP_VSTYPE	Description	To float (in C/C++)
0	Normalized unsigned 8bit	((unsigned char)Data) / 255.0f
1	Integer unsigned 8bit	(float) ((unsigned char)Data)
2	Integer signed 8 bit	(float) ((char)Data)
3	Integer signed 16 bit	(float) ((short)Data)
4	Integer signed 32 bit	(float) ((int)Data)
5	Fixed point number s.15.16	((int)Data) / 65536.0f
6	Reserved	
7	Floating point number s.e8.23	Bypass

<table>
<tr><th>Bit</th><th>R/W</th><th>Symbol</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSADDR1</td><td>Vertex stream 1 address.</td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM1 STRIDE REGISTER (GRP3D_VSSTRIDE1)
Address : C001_A130h</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSSTRIDE1</td><td>Vertex stream 1 stride(number of bytes in a vertex)</td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM1 PARAMETER REGISTER (GRP3D_VSPARAM1)
Address : C001_A134h</td></tr>
<tr><td>[31 :20]</td><td>W</td><td>GRP3D_ VSDST1</td><td>Vertex stream 1 destination offset of GTE/TSE register. (unit: 32bit)</td><td>-</td></tr>
<tr><td>[19 : 8]</td><td>W</td><td>GRP3D_VSSIZE1</td><td>Vertex stream 1 size.(N-1, N is number of items in a vertex)</td><td>-</td></tr>
<tr><td>[7: 0]</td><td>W</td><td>GRP3D_VSTYPE1</td><td>Vertex stream 1 type.
It describes number format of vertex stream data.
<table>
<tr><th>GRP_VSTYPE</th><th>Description</th><th>To float (in C/C++)</th></tr>
<tr><td>0</td><td>Normalized unsigned 8bit</td><td>((unsigned char)Data) / 255.0f</td></tr>
<tr><td>1</td><td>Integer unsigned 8bit</td><td>(float) ((unsigned char)Data)</td></tr>
<tr><td>2</td><td>Integer signed 8 bit</td><td>(float) ((char)Data)</td></tr>
<tr><td>3</td><td>Integer signed 16 bit</td><td>(float) ((short)Data)</td></tr>
<tr><td>4</td><td>Integer signed 32 bit</td><td>(float) ((int)Data)</td></tr>
<tr><td>5</td><td>Fixed point number s.15.16</td><td>((int)Data) / 65536.0f</td></tr>
<tr><td>6</td><td>Reserved</td><td></td></tr>
<tr><td>7</td><td>Floating point number s.e8.23</td><td>Bypass</td></tr>
</table></td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM2 ADDRESS REGISTER (GRP3D_VSDDR2)
Address : C001_A138h</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSADDR2</td><td>Vertex stream 2 address.</td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM2 STRIDE REGISTER (GRP3D_VSSTRIDE2)
Address : C001_A13Ch</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSSTRIDE2</td><td>Vertex stream 2 stride(number of bytes in a vertex)</td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM2 PARAMETER REGISTER (GRP3D_VSPARAM2)
Address : C001_A140h</td></tr>
<tr><td>[31 :20]</td><td>W</td><td>GRP3D_ VSDST2</td><td>Vertex stream 2 destination offset of GTE/TSE register. (unit: 32bit)</td><td>-</td></tr>
<tr><td>[19 : 8]</td><td>W</td><td>GRP3D_VSSIZE2</td><td>Vertex stream 2 size. (N-1, N is number of items in a vertex)</td><td>-</td></tr>
<tr><td>[7: 0]</td><td>W</td><td>GRP3D_VSTYPE2</td><td>Vertex stream 2 type.
It describes number format of vertex stream data.
<table>
<tr><th>GRP_VSTYPE</th><th>Description</th><th>To float (in C/C++)</th></tr>
<tr><td>0</td><td>Normalized unsigned 8bit</td><td>((unsigned char)Data) / 255.0f</td></tr>
<tr><td>1</td><td>Integer unsigned 8bit</td><td>(float) ((unsigned char)Data)</td></tr>
<tr><td>2</td><td>Integer signed 8 bit</td><td>(float) ((char)Data)</td></tr>
<tr><td>3</td><td>Integer signed 16 bit</td><td>(float) ((short)Data)</td></tr>
<tr><td>4</td><td>Integer signed 32 bit</td><td>(float) ((int)Data)</td></tr>
<tr><td>5</td><td>Fixed point number s.15.16</td><td>((int)Data) / 65536.0f</td></tr>
<tr><td>6</td><td>Reserved</td><td></td></tr>
<tr><td>7</td><td>Floating point number s.e8.23</td><td>Bypass</td></tr>
</table></td><td>-</td></tr>
</table>

<table>
<tr><th>Bit</th><th>R/W</th><th>Symbol</th><th>Description</th><th>Reset Value</th></tr>
<tr><td colspan="5">VERTEX STREAM3 ADDRESS REGISTER (GRP3D_VSDDR3)
Address : C001_A144h</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSADDR3</td><td>Vertex stream 3 address.</td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM3 STRIDE REGISTER (GRP3D_VSSTRIDE3)
Address : C001_A148h</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSSTRIDE3</td><td>Vertex stream 3 stride(number of bytes in a vertex)</td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM3 PARAMETER REGISTER (GRP3D_VSPARAM3)
Address : C001_A14Ch</td></tr>
<tr><td>[31 :20]</td><td>W</td><td>GRP3D_ VSDST3</td><td>Vertex stream 3 destination offset of GTE/TSE register. (unit: 32bit)</td><td>-</td></tr>
<tr><td>[19 : 8]</td><td>W</td><td>GRP3D_VSSIZE3</td><td>Vertex stream 3 size. (N-1, N is number of items in a vertex)</td><td>-</td></tr>
<tr><td>[7: 0]</td><td>W</td><td>GRP3D_VSTYPE3</td><td>Vertex stream 3 type.
It describes number format of vertex stream data.
<table>
<tr><th>GRP_VSTYPE</th><th>Description</th><th>To float (in C/C++)</th></tr>
<tr><td>0</td><td>Normalized unsigned 8bit</td><td>((unsigned char)Data) / 255.0f</td></tr>
<tr><td>1</td><td>Integer unsigned 8bit</td><td>(float) ((unsigned char)Data)</td></tr>
<tr><td>2</td><td>Integer signed 8 bit</td><td>(float) ((char)Data)</td></tr>
<tr><td>3</td><td>Integer signed 16 bit</td><td>(float) ((short)Data)</td></tr>
<tr><td>4</td><td>Integer signed 32 bit</td><td>(float) ((int)Data)</td></tr>
<tr><td>5</td><td>Fixed point number s.15.16</td><td>((int)Data) / 65536.0f</td></tr>
<tr><td>6</td><td>Reserved</td><td></td></tr>
<tr><td>7</td><td>Floating point number s.e8.23</td><td>Bypass</td></tr>
</table></td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM4 ADDRESS REGISTER (GRP3D_VSDDR4)
Address : C001_A150h</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSADDR4</td><td>Vertex stream 0 address.</td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM4 STRIDE REGISTER (GRP3D_VSSTRIDE4)
Address : C001_A154h</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSSTRIDE4</td><td>Vertex stream 0 stride(number of bytes in a vertex)</td><td>-</td></tr>
<tr><td colspan="5">VERTEX STREAM4 PARAMETER REGISTER (GRP3D_VSPARAM4)
Address : C001_A158h</td></tr>
<tr><td>[31 :20]</td><td>W</td><td>GRP3D_ VSDST4</td><td>Vertex stream 4 destination offset of GTE/TSE register. (unit: 32bit)</td><td>-</td></tr>
<tr><td>[19 : 8]</td><td>W</td><td>GRP3D_VSSIZE4</td><td>Vertex stream 4 size. (N-1, N is number of items in a vertex)</td><td>-</td></tr>
<tr><td>[7: 0]</td><td>W</td><td>GRP3D_VSTYPE4</td><td>Vertex stream 4 type.
It describes number format of vertex stream data.
<table>
<tr><th>GRP_VSTYPE</th><th>Description</th><th>To float (in C/C++)</th></tr>
<tr><td>0</td><td>Normalized unsigned 8bit</td><td>((unsigned char)Data) / 255.0f</td></tr>
<tr><td>1</td><td>Integer unsigned 8bit</td><td>(float) ((unsigned char)Data)</td></tr>
<tr><td>2</td><td>Integer signed 8 bit</td><td>(float) ((char)Data)</td></tr>
<tr><td>3</td><td>Integer signed 16 bit</td><td>(float) ((short)Data)</td></tr>
<tr><td>4</td><td>Integer signed 32 bit</td><td>(float) ((int)Data)</td></tr>
<tr><td>5</td><td>Fixed point number s.15.16</td><td>((int)Data) / 65536.0f</td></tr>
<tr><td>6</td><td>Reserved</td><td></td></tr>
</table></td><td>-</td></tr>
</table>

<table>
<tr><th>Bit</th><th>R/W</th><th>Symbol</th><th colspan="4">Description</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td></td><td></td><td>7</td><td>Floating point number s.e8.23</td><td>Bypass</td><td></td></tr>
<tr><td colspan="8">VERTEX STREAM5 ADDRESS REGISTER (GRP3D_VSDDR5)
Address : C001_A15Ch</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSADDR5</td><td colspan="4">Vertex stream 5 address.</td><td>-</td></tr>
<tr><td colspan="8">VERTEX STREAM5 STRIDE REGISTER (GRP3D_VSSTRIDE5)
Address : C001_A160h</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSSTRIDE5</td><td colspan="4">Vertex stream 5 stride(number of bytes in a vertex)</td><td>-</td></tr>
<tr><td colspan="8">VERTEX STREAM5 PARAMETER REGISTER (GRP3D_VSPARAM5)
Address : C001_A164h</td></tr>
<tr><td>[31 :20]</td><td>W</td><td>GRP3D_ VSDST5</td><td colspan="4">Vertex stream 5 destination offset of GTE/TSE register. (unit: 32bit)</td><td>-</td></tr>
<tr><td>[19 : 8]</td><td>W</td><td>GRP3D_VSSIZE5</td><td colspan="4">Vertex stream 5 size. (N-1, N is number of items in a vertex)</td><td>-</td></tr>
<tr><td>[7: 0]</td><td>W</td><td>GRP3D_VSTYPE5</td><td colspan="4">Vertex stream 5 type.
It describes number format of vertex stream data.
<table><tr><th>GRP_VSTYPE</th><th>Description</th><th>To float (in C/C++)</th></tr><tr><td>0</td><td>Normalized unsigned 8bit</td><td>((unsigned char)Data) / 255.0f</td></tr><tr><td>1</td><td>Integer unsigned 8bit</td><td>(float) ((unsigned char)Data)</td></tr><tr><td>2</td><td>Integer signed 8 bit</td><td>(float) ((char)Data)</td></tr><tr><td>3</td><td>Integer signed 16 bit</td><td>(float) ((short)Data)</td></tr><tr><td>4</td><td>Integer signed 32 bit</td><td>(float) ((int)Data)</td></tr><tr><td>5</td><td>Fixed point number s.15.16</td><td>((int)Data) / 65536.0f</td></tr><tr><td>6</td><td>Reserved</td><td></td></tr><tr><td>7</td><td>Floating point number s.e8.23</td><td>Bypass</td></tr></table></td><td>-</td></tr>
<tr><td colspan="8">VERTEX STREAM6 ADDRESS REGISTER (GRP3D_VSDDR6)
Address : C001_A168h</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSADDR6</td><td colspan="4">Vertex stream 6 address.</td><td>-</td></tr>
<tr><td colspan="8">VERTEX STREAM6 STRIDE REGISTER (GRP3D_VSSTRIDE60)
Address : C001_A16Ch</td></tr>
<tr><td>[31 : 0]</td><td>W</td><td>GRP3D_VSSTRIDE6</td><td colspan="4">Vertex stream 6 stride(number of bytes in a vertex)</td><td>-</td></tr>
<tr><td colspan="8">VERTEX STREAM6 PARAMETER REGISTER (GRP3D_VSPARAM6)
Address : C001_A170h</td></tr>
<tr><td>[31 :20]</td><td>W</td><td>GRP3D_ VSDST6</td><td colspan="4">Vertex stream 6 destination offset of GTE/TSE register. (unit: 32bit)</td><td>-</td></tr>
<tr><td>[19 : 8]</td><td>W</td><td>GRP3D_VSSIZE6</td><td colspan="4">Vertex stream 6 size. (N-1, N is number of items in a vertex)</td><td>-</td></tr>
<tr><td>[7: 0]</td><td>W</td><td>GRP3D_VSTYPE6</td><td colspan="4">Vertex stream 6 type.
It describes number format of vertex stream data.
<table><tr><th>GRP_VSTYPE</th><th>Description</th><th>To float (in C/C++)</th></tr><tr><td>0</td><td>Normalized unsigned 8bit</td><td>((unsigned char)Data) / 255.0f</td></tr><tr><td>1</td><td>Integer unsigned 8bit</td><td>(float) ((unsigned char)Data)</td></tr><tr><td>2</td><td>Integer signed 8 bit</td><td>(float) ((char)Data)</td></tr><tr><td>3</td><td>Integer signed 16 bit</td><td>(float) ((short)Data)</td></tr><tr><td>4</td><td>Integer signed 32 bit</td><td>(float) ((int)Data)</td></tr><tr><td>5</td><td>Fixed point number s.15.16</td><td>((int)Data) / 65536.0f</td></tr></table></td><td>-</td></tr>
</table>

Bit	R/W	Symbol	Description	Reset Value
			6 Reserved 7 Floating point number s.e8.23 Bypass	
VERTEX STREAM7 ADDRESS REGISTER (GRP3D_VSDDR7) ***Address : C001_A174h***				
[31 : 0]	W	**GRP3D_VSADDR7**	Vertex stream 7 address.	-
VERTEX STREAM7 STRIDE REGISTER (GRP3D_VSSTRIDE7) ***Address : C001_A178h***				
[31 : 0]	W	**GRP3D_VSSTRIDE7**	Vertex stream 7 stride.(number of bytes in a vertex)	-
VERTEX STREAM7 PARAMETER REGISTER (GRP3D_VSPARAM7) ***Address : C001_A17Ch***				
[31 :20]	W	**GRP3D_ VSDST7**	Vertex stream 7 destination offset of GTE/TSE register. (unit: 32bit)	-
[19 : 8]	W	**GRP3D_VSSIZE7**	Vertex stream 7 size. (N-1, N is number of items in a vertex)	-
[7: 0]	W	**GRP3D_VSTYPE0**	Vertex stream 7 type. It describes number format of vertex stream data. 0 Normalized unsigned 8bit ((unsigned char)Data) / 255.0f 1 Integer unsigned 8bit (float) ((unsigned char)Data) 2 Integer signed 8 bit (float) ((char)Data) 3 Integer signed 16 bit (float) ((short)Data) 4 Integer signed 32 bit (float) ((int)Data) 5 Fixed point number s.15.16 ((int)Data) / 65536.0f 6 Reserved 7 Floating point number s.e8.23 Bypass	-
FOG TABLE REGISTER (GRP3D_FOGTBL) ***Address : C001_A400h ~ A47Fh***				
[31 : 30]	-	**RESERVED**	Reserved	-
[29 : 24]	W	**GRP3D_FOGDIFF**	Difference of fog table value (1,3,5,7,…,63)	-
[23 : 16]	W	**GRP3D_FOGVAL**	Fog table value (1,3,5,7,…,63)	-
[15 : 14]	-	**RESERVED**	Reserved	-
[13 : 8]	W	**GRP3D_FOGDIFF**	Difference of fog table value (0,2,4,8,…,62)	-
[7 : 0]	-	**GRP3D_FOGVAL**	Fog table value (0,2,4,8,…,62)	-
<Note> for 0 <= n <= 62, GRP3D_FOGDIFF[n] = (GRP3D_FOGVAL[n+1] - GRP3D_FOGVAL[n]) / 4 GRP3D_FOGDIFF [63] must be 0				
TSE INPUT VERTEX REGISTER (GRP3D_TSEINPUT) ***Address : C001_A500h ~ A5AFh***				
[31 : 8]	W	**GRP3D_TSEINPUT**	TSE input (Clipper output) vertex 24bit floating point number.	-
[7 : 0]	-	**RESERVED**	Reserved	-
<Note> ((float*) GRP3D_TSEINPUT)[n * 16 + 0] : X of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 1] : Y of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 2] : Z of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 3] : W of input vertex[n]				

Bit	R/W	Symbol	Description	Reset Value
((float*) GRP3D_TSEINPUT)[n * 16 + 4] : A of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 5] : R of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 6] : G of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 7] : B of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 8] : U0 of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 9] : V0 of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 10] : U1 of input vertex[n] ((float*) GRP3D_TSEINPUT)[n * 16 + 11] : V1 of input vertex[n] (0 <= n <= 2 , GRP3D_TSEINPUT = C001_A500h)				
RESERVED ***Address : C001_A5B0h ~ A6FFh***				
GTE OUTPUT VERTEX REGISTER (GRP3D_GTEOUTPUT) ***Address : C001_A700h ~ A78Fh***				
[31 : 8]	W	**GRP3D_GTEOUTPUT**	GTE output (Clipper input) vertex 24bit floating point number.	-
[7 : 0]	-	**RESERVED**	Reserved	-
<Note> ((float*)GRP3D_GTEOUTPUT)[n * 12 + 0] : X of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 1] : Y of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 2] : Z of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 3] : W of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 4] : A of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 5] : R of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 6] : G of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 7] : B of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 8] : U0 of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 9] : V0 of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 10] : U1 of input vertex[n] ((float*)GRP3D_GTEOUTPUT)[n * 12 + 11] : V1 of input vertex[n] (0 <= n <= 2 , GRP3D_GTEOUTPUT = C001_A700h)				
RESERVED ***Address : C001_A790h ~ A7FFh***				
GTE PROGRAM REGISTER (GRP3D_GTECODE) ***Address : C001_A800h ~AFFFh***				
[31 : 0]	W	**GRP3D_GTECODE**	GTE Instruction	-
GTE CONSTANT VECTOR REGISTER (GRP3D_GTECONST) ***Address : C001_B000h ~ C001_BFBFh***				
[31 : 8]	W	**GRP3D_GTECONST**	GTE constant vector 24bit floating point number.	-
[7 : 0]	-	**RESERVED**	Reserved	-
<Note> ((float*)GRP3D_GTECONST)[n * 4 + 0] : X of const vector[n] ((float*)GRP3D_GTECONST)[n * 4 + 1] : Y of const vector[n] ((float*)GRP3D_GTECONST)[n * 4 + 2] : Z of const vector[n] ((float*)GRP3D_GTECONST)[n * 4 + 3] : W of const vector[n] (0 <= n <= 251, GRP3D_GTECONST = C001_B000h)				
GRP3D CLOCK ENABLE REGISTER (GRP3DCLKENB) ***Address : C001_BFC0h***				
[31:24]	R	**RESERVED**	Reserved	28'b0

Bit	R/W	Symbol	Description	Reset Value
[3]	R/W	**PCLKMODE**	Specifies PCLK operating mode. 0 : PCLK is only enabled when CPU accesses this module 1 : PCLK is always enabled	1'b0
[2]	R	**RESERVED**	Reserved	1'b0
[1:0]	R/W	**BCLKMODE**	Specifies BCLK operating mode. 0 : BCLK is always disabled 1 : Reserved for future use 2 : BCLK is dynamic controlled by this module 3 : BCLK is always enabled	2'b0

CHAPTER 23.

COLOR SPACE CONVERTER (CSC)

23. COLOR SPACE CONVERTER (CSC)

23.1. Overview

Color Space Converter (CSC) module converts YCbCr output from MPEG decoder (or JPEG decoder) to RGB format for using as a texture of 3D core. CSC is designed to convert YCbCr image which is generated by MPEG decoder(or JPEG decoder) into texture image for 3D processing.

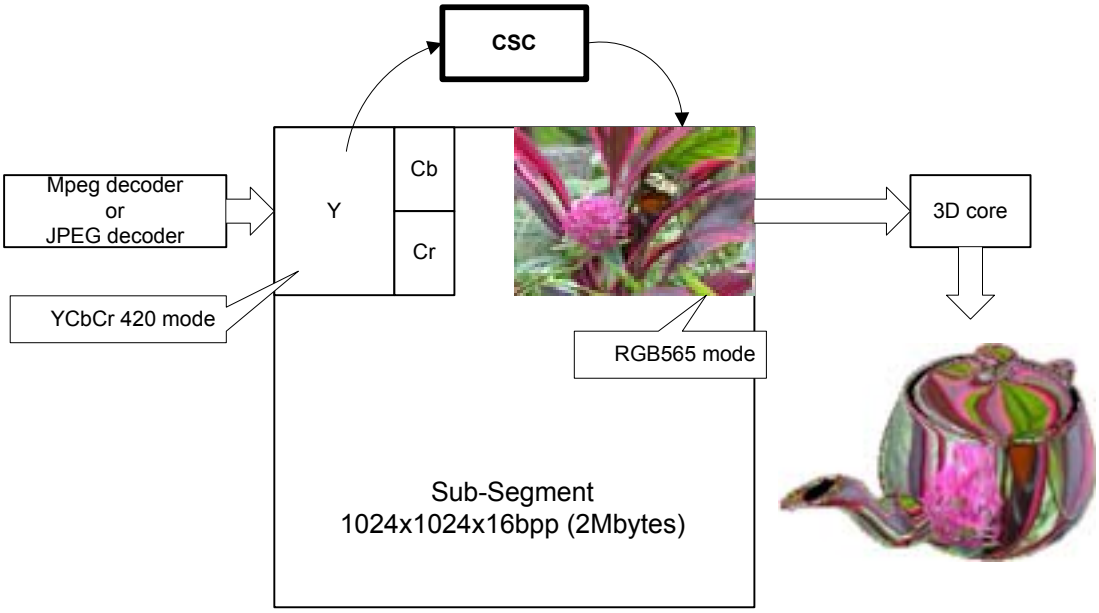


Figure 23-1. Color Space Conversion

23.2. Operation

23.2.1. Input/ Output data

Maximum convertible size is 1024x1024 pixel because CSC uses Sub-Segment/ Block addressing. (Refer to 3D Core memory structure for Sub-Segment/ Block addressing.)

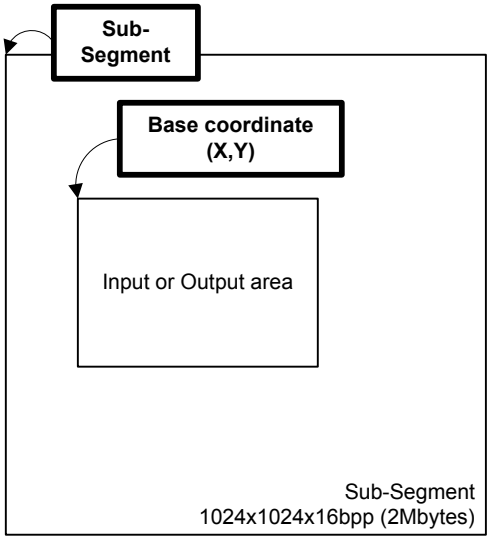


Figure 23-2. Input/ Output Data Addressing

23.2.2. Input from MPEG decoder or JPEG decoder

Image is stored coming from MPEG decoder(or JPEG decoder) into designated memory area. The output format of MPEG decoder(or JPEG decoder) is YCbCr 420 format that each component of it is separated.
Block addressing mode is for display mode (Refer to 3D Core memory structure for block addressing mode and display mode)

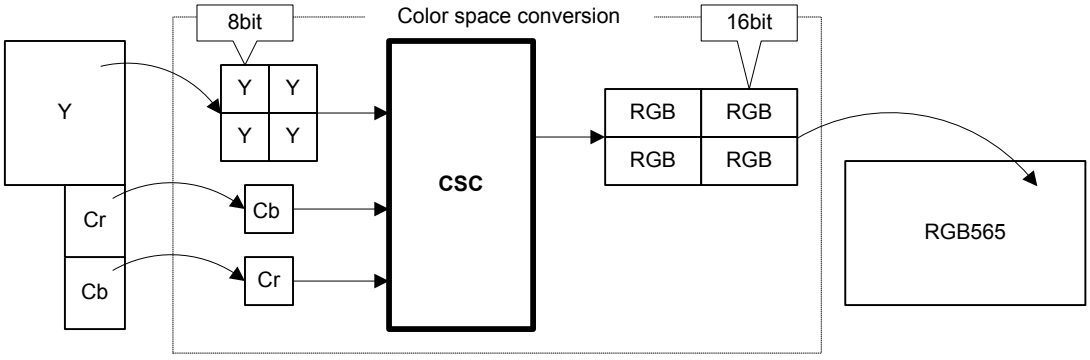


Figure 23-3. YCbCr 420 Mode (from MPEG Decoder)

23.2.3. Output to Texture

3D core can use CSC output as texture image. It fills rectangle area with color which was converted as RGB565 format when Sub-Segment/Base-coordinate of output position is assigned. Block addressing mode is for texture mode (Refer to Memory structure of 3D core for block addressing mode)

23.2.4. Transparency Test

If any component of input color (Y, Cb, Cr) belongs to transparency range, output color will be transparency color.

23.2.5. Clock Generation

The CSC operates by using the PCLK and the BCLK. The PCLK is used when the CPU accesses the register of the CSC. The BCLK is used for the CSC's memory access and as an internal clock. Since the CSC provides various operation modes for the PCLK and the BCLK, users can adjust the CSC clock by setting the ***PCLKMODE*** and the ***BCLKMODE*** parameters according to their purpose.

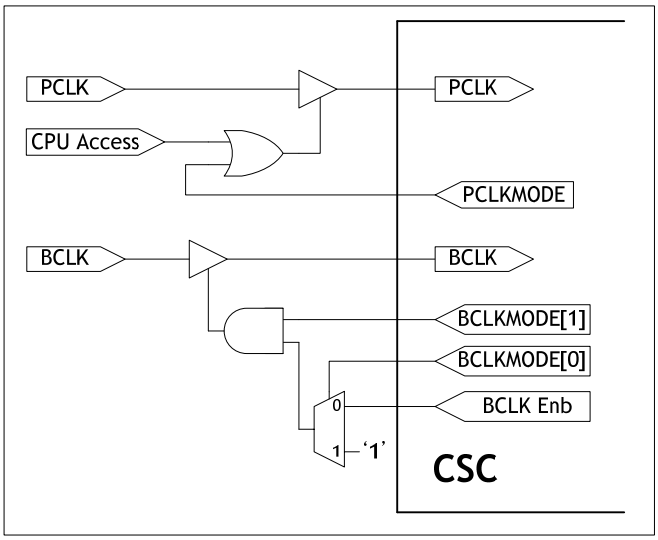


Figure 23-4. Clock generation

PCLKMODE	Brief Description
0	Only enabled when CPU accesses CSC Register.
1	Always enabled..

Table 23-1. PCLK mode

BLCKMODE		Brief Description
[1]	[0]	
0	0	Always disabled.
0	1	Reserved for future use.
1	0	Dynamic controlled by CSC
1	1	Always enabled.

Table 23-2. BCLK mode

23.3. CSC Registers

23.3.1. Register map (C000_3200h ~ C000_3227h)

Register Name	offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONTROL	00h																											D	Q	0	IP	IE	R
SIZE	04h							HEIGHT						1	1	1	1							WIDTH						1	1	1	1
ADDRY	08h	SEG_Y												OFFY_Y						0	0	0	OFFX_Y							0	0	0	0
ADDRCB	0Ch	SEG_CB												OFFY_CB						0	0	0	OFFX_CB							0	0	0	0
ADDRCR	10h	SEG_CR												OFFY_CR						0	0	0	OFFX_CR							0	0	0	0
ADDRTEX	14h	SEG_TEX												OFFY_TEX						0	0	0	OFFX_TEX							0	0	0	0
TPCTRL	18h											TPENB			TPINV			TPCOLOR															
TPY	1Ch																	TPMAXY										TPMINY					
TPCB	20h																	TPMAXCB										TPMINCB					
TPCR	24h																	TPMAXCR										TPMINCR					
Reserved	-																																
CSCCLKENB	1C0h																													PM		BM	

Note) CONTROL : R(RUN), IE(INTENB), IP(INTPEND), Q(QMODE), D(DITHER)
CSCCLKENB : BM(BCLKMODE), PM(PCLKMODE)

Table 23-3. Register MAP

23.3.2. Registers

Bit	R/W	Symbol	Description	Reset Value
Control Register (CONTROL) ***Address : C000_9000h***				
[31:16]	R	**RESERVED**	Reserved	16'bX
[15:6]	R	**RESERVED**	Reserved	11'b0
[5]	R/W	**DITHER**	Specifies whether the dithering function is enable or disable. 0 : Disable 1 : Enable	1'bx
[4]	R/W	**QMODE**	Specifies the quantization mode. 0: 220 quantization mode ▪ R = Y + 1.375 * Cr ▪ G = Y - 0.3359375 * Cb - 0.6953125 * Cr ▪ B = Y + 1.734375 * Cb 1: 256 quantization mode ▪ R = Y + 1.4020 * Cr ▪ G = Y - 0.34414 * Cb - 0.71414 * Cr ▪ B = Y + 1.7720 * Cb	1'bx
[3]	R	**RESERVED**	Reserved but you have to write 0 only	1'bx
[2]	R/W	**INTPEND**	Informs whether the interrupt is pended or not. Read > 0 : no interrupt is generated 1: interrupt caused because CSC done Write > 0 : No affect 1: interrupt pending bit(CSCPEND) clear	1'b0
[1]	R/W	**INTENB**	Specifies whether to issue the interrupt when CSC operation has done. 0 : Disable 1 : Enable	1'b0
[0]	R/W	**RUN**	Stat CSC or informs whether CSC is busy or not. Read > 0 : CSC is idle 1: CSC is busy Write > 0 : No affect 1: Start CSC	1'b0
Size Register (SIZE) ***Address : C000_9004h***				
[31:26]	-	**RESERVED**	Reserved	6'bx
[25:16]	W	**HEIGHT**	Specifies the image height, in pixels, which must be multiple of 16. ▪ HEIGHT = the image height – 1	10'bx
[15:10]	-	**RESERVED**	Reserved	6'bx

Bit	R/W	Symbol	Description	Reset Value
[9:0]	W	**WIDTH**	Specifies the image width, in pixels, which must be multiple of 16. ▪ WIDTH = the image width – 1	10'bx
ADDRESS Y CHANNEL REGISTER (ADDRY) ***Address : C000_9008h***				
[31:29]	W	**RESERVED**	Reserved but you have to write '0' only.	3'bx
[28:21]	W	**SEG_Y**	Specifies the index of sub-segment which contains Y channel of the source image. ▪ SEG_Y = { LinearAddr[31:22], LinearAddr[16] }	8'bx
[20:11]	W	**OFFY_Y**	Specifies the y-coordinate of the upper-left corner of the Y channel in sub-segment. It must be multiple of 2^N, where N is 0 to 9 and 2^N must be greater than or equal to the image height in pixels. ▪ OFFY_Y = { LinearAddr[21:17], LinearAddr[10:6] }	10'bx
[10:0]	W	**OFFX_Y**	Specifies the x-coordinate of the upper-left corner of the Y channel in sub-segment. It must be multiple of 2^N, where N is 0 to 10 and 2^N must be greater than or equal to the image width in pixels. ▪ OFFX_Y = { LinearAddr[15:11], LinearAddr[5:0] }	11'bx
ADDRESS CB CHANNEL REGISTER (ADDRCB) ***Address : C000_900Ch***				
[31:29]	W	**RESERVED**	Reserved but you have to write '0' only.	3'bx
[28:21]	W	**SEG_CB**	Specifies the index of sub-segment which contains Cb channel of the source image. ▪ SEG_CB = { LinearAddr[31:22], LinearAddr[16] }	8'bx
[20:11]	W	**OFFY_CB**	Specifies the y-coordinate of the upper-left corner of the Cb channel in sub-segment. It must be multiple of 2^N, where N is 0 to 9 and 2^N must be greater than or equal to the image height in pixels. ▪ OFFY _CB= { LinearAddr[21:17], LinearAddr[10:6] }	10'bx
[10:0]	W	**OFFX_CB**	Specifies the x-coordinate of the upper-left corner of the Cb channel in sub-segment. It must be multiple of 2^N, where N is 0 to 10 and 2^N must be greater than or equal to the image width in pixels. ▪ OFFX_CB = { LinearAddr[15:11], LinearAddr[5:0] }	11'bx
ADDRESS CR CHANNEL REGISTER (ADDRCR) ***Address : C000_9010h***				
[31:29]	W	**RESERVED**	Reserved but you have to write '0' only.	3'bx
[28:21]	W	**SEG_CR**	Specifies the index of sub-segment which contains Cr channel of the source image. ▪ SEG_CR = { LinearAddr[31:22], LinearAddr[16] }	8'bx
[20:11]	W	**OFFY_CR**	Specifies the y-coordinate of the upper-left corner of the Cr channel in sub-segment. It must be multiple of 2^N, where N is 0 to 9 and 2^N must be greater than or equal to the image height in pixels. ▪ OFFY_CR = { LinearAddr[21:17], LinearAddr[10:6] }	10'bx
[10:0]	W	**OFFX_CR**	Specifies the x-coordinate of the upper-left corner of the Cr channel in sub-segment. It must be multiple of 2^N, where N is 0 to 10 and 2^N must be greater than or equal to the image width in pixels. ▪ OFFX_CR = { LinearAddr[15:11], LinearAddr[5:0] }	11'bx
ADDRESS TEXTURE REGISTER (ADDRREX) ***Address : C000_9014h***				
[31:21]	W	**SEG_TEX**	Specifies the index of sub-segment which will be contain texture image. ▪ SEG_TEX = { LinearAddr[31:22], LinearAddr[16] }	11'bx
[20:11]	W	**OFFY_TEX**	Specifies the y-coordinate of the upper-left corner of the texture image in sub-segment. It must be multiple of 2^N, where N is 0 to 9 and 2^N must be greater than or equal to the image height in pixels. ▪ OFFY_TEX = { LinearAddr[21:17], LinearAddr[10:7], LinearAddr[2] }	10'bx
[10:0]	W	**OFFX_TEX**	Specifies the x-coordinate of the upper-left corner of the texture image in sub-segment. It must be multiple of 2^N, where N is 0 to 10 and 2^N must be greater than or equal to double of the image width in pixels. ▪ OFFX_TEX = { LinearAddr[15:11], LinearAddr[6:3], LinearAddr[1:0] }	11'bx

Bit	R/W	Symbol	Description	Reset Value
TRANSPARENCY CONTROL REGISTER (TPCTRL) ***Address : C000_9018h***				
[31:22]	-	**RESERVED**	Reserved	10'bx
[21:19]	W	**TPENB**	Specifies whether to test for transparency. 0 : Disable 7 : Enable 1~6: Reserved.	3'bx
[18]	W	**TPINVCR**	Specifies the transparency range of Cr channel when TPENBCR is set. The result color of CSC is replaced by TPCOLOR when, 0 : TPMINCR <= Cr <= TPMAXCR 1 : Cr < TPMINCR or Cr > TPMAXCR	1'bx
[17]	W	**TPINVCB**	Specifies the transparency range of Cb channel when TPENBCB is set. The result color of CSC is replaced by TPCOLOR when, 0 : TPMINCB <= Cb <= TPMAXCB 1 : Cb < TPMINCB or Cb > TPMAXCB	1'bx
[16]	W	**TPINVY**	Specifies the transparency range of Y channel when TPENBY is set. The result color of CSC is replaced by TPCOLOR when, 0 : TPMINY <= Y <= TPMAXY 1 : Y < TPMINY or Y > TPMAXY	1'bx
[15:0]	W	**TPCOLOR**	Specifies the transparency color which has RGB565 format.	16'bx
TRANSPARENCY Y RANGE REGISTER (TPY) ***Address : C000_901Ch***				
[31:16]	-	**RESERVED**	Reserved	16'bx
[15:8]	W	**TPMAXY**	Specifies maximum value of Y channel for transparency.	8'bx
[7:0]	W	**TPMINY**	Specifies minimum value of Y channel for transparency.	8'bx
TRANSPARENCY CB RANGE REGISTER (TPCB) ***Address : C000_9020h***				
[31:16]	-	**RESERVED**	Reserved	16'bx
[15:8]	W	**TPMAXCB**	Specifies maximum value of Cb channel for transparency.	8'bx
[7:0]	W	**TPMINCB**	Specifies minimum value of Cb channel for transparency.	8'bx
TRANSPARENCY CR RANGE REGISTER (TPCR) ***Address : C000_9024h***				
[31:16]	-	**RESERVED**	Reserved	16'bx
[15:8]	W	**TPMAXCR**	Specifies maximum value of Cr channel for transparency.	8'bx
[7:0]	W	**TPMINCR**	Specifies minimum value of Cr channel for transparency.	8'bx
TRANSPARENCY CR RANGE REGISTER (TPCR) ***Address : C000_9028h ~ C000_91BFh***				
[31:0]	-	**RESERVED**	Reserved	-
CSC CLOCK GENERATION ENABLE REGISTER (CSCCLKENB) ***Address : C000_97C0h***				
[31:24]	R	**RESERVED**	Reserved	28'b0
[3]	R/W	**PCLKMODE**	Specifies PCLK operating mode. 0 : PCLK is only enabled when CPU accesses this module 1 : PCLK is always enabled	1'b0
[2]	R	**RESERVED**	Reserved	1'b0
[1:0]	R/W	**BCLKMODE**	Specifies BCLK operating mode. 0 : BCLK is always disabled 1 : Reserved for future use 2 : BCLK is dynamic controlled by this module 3 : BCLK is always enabled	2'b0

CHAPTER 24.

ELECTRICAL CHARACTERISTIC

24. ELECTRICAL CHARACTERISTIC

24.1. ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Rating			Unit
		Descriptions	Min	Max	
V_{DD}	DC Supply Voltage	2.5V V_{DD}	-0.5	3.6	V
		3.3V V_{DD}	-0.5	4.6	
V_{IN}	DC Input Voltage	2.5V Input Buffer	-0.5	3.6	
		3.3V Input Buffer	-0.5	4.6	
		2.5V Interface/3.3V Tolerant Input Buffer	-0.5	3.8	
		3.3V Interface/5V Tolerant Input Buffer	-0.5	6.5	
V_{OUT}	DC Output Voltage	2.5V Output Buffer	-0.5	3.6	
		3.3V Output Buffer	-0.5	4.6	
		2.5V Interface/3.3V Tolerant Output Buffer	-0.5	3.8	
		3.3V Interface/5V Tolerant Output Buffer	-0.5	6.5	
$V_{I/O}$	In/out Current		±20		mA
T_A	Storage Temperature		-65 to 150		℃

Table 24-1. Absolute Maximum Rationgs

24.2. RECOMMENDED OPERATION CONDITIONS

Symbol	Parameter	Rating			Unit
		Descriptions	Min	Max	
V_{DD}	DC Supply Voltage for Internal	1.0V V_{DD}	0.9	1.1	V
		1.2V V_{DD}	1.1	1.3	
	DC Supply Voltage for I/O Block	2.5V V_{DD}	2.3	2.7	
		3.3V V_{DD}	3.0	3.6	
T_A	Commercial Temperature Range		0 to 70		℃
	Industrial Temperature Range		-40 to 85		℃

Table 24-2. Recommended Operation Conditions

24.3. D.C. ELECTRICAL CHARACTERISTICS

These DC Parameters guarantee the I/O cell characteristic at the static state only, not at the dynamic state.

24.3.1. Normal I/O – 3.3v

DC Currents of Output Driver

Parameter		Currents		
Driver Type		Worst VDD=3.00V T=125°C Process=Slow Isink at 0.20V Isource at 2.80V	Typical VDD=3.30V T=25°C Process=Nominal Isink at 0.20V Isource at 3.10V	Best VDD=3.60V T=-40°C Process=Fast Isink at 0.20V Isource at 3.40V
DriveStrength = 2mA	Isink	1.86mA	2.86mA	3.87mA
	Isource	-1.95mA	-2.74mA	-3.77mA
DriveStrength = 4mA	Isink	3.73mA	5.73mA	7.75mA
	Isource	-3.92mA	-5.49mA	-7.54mA
DriveStrength = 6mA	Isink	5.61mA	8.59mA	11.6mA
	Isource	-5.88mA	-8.24mA	-11.3mA
DriveStrength = 8mA	Isink	7.47mA	11.5mA	15.5mA
	Isource	-7.83mA	-10.1mA	-15.1mA

Table 24-1. Normal 3.3v DC Currents of Output Driver

DC Electric Specification, V_{DD} = 3.0 ~ 3.6V

Symbol	Parameter	Rating				Unit
		Condition	Min	Typ	Max	
V_{IH}	High Level Input Voltage (LVCMOS Interface)		$0.7V_{DD}$		V_{DD}+0.3	V
V_{IL}	Low Level Input Voltage (LVCMOS Interface)		-0.3		$0.3V_{DD}$	V
ΔV	Hysteresis Voltage			$0.15V_{DD}$		V
I_{IH}	High Level Input Current (Input Buffer)	V_{IN}= V_{DD}	-10		10	uA
I_{IL}	High Level Input Current (Input Buffer)	V_{IN}= V_{SS}	-10		10	uA
V_{OH}	High Level Output Voltage	I_{OH} = -100uA	V_{DD} -0.2			V
V_{OL}	Low Level Output Voltage	I_{OL} = 100uA			0.2	V

Table 24-4. DC Electric Specification

24.3.2. DDR Normal I/O – 2.5V

DC Currents of Output Driver

Parameter		Currents		
Driver Type		Worst VDD=2.30V T=125°C Process=Slow Isink at 0.20V Isource at 2.10V	Typical VDD=2.50V T=25°C Process=Nominal Isink at 0.20V Isource at 2.30V	Best VDD=2.70V T=-40°C Process=Fast Isink at 0.20V Isource at 2.50V
DriveStrength = 4mA	Isink	T.B.D	T.B.D	T.B.D
	Isource	T.B.D	T.B.D	T.B.D
DriveStrength = 8mA	Isink	T.B.D	T.B.D	T.B.D
	Isource	T.B.D	T.B.D	T.B.D
DriveStrength = 10mA	Isink	T.B.D	T.B.D	T.B.D
	Isource	T.B.D	T.B.D	T.B.D
DriveStrength = 14mA	Isink	T.B.D	T.B.D	T.B.D
	Isource	14mA	T.B.D	T.B.D

Table 24-5. DC Currents of Output Driver

DC Electric Specification, V_{DD} = 2.3 ~ 2.7V

Symbol	Parameter	Rating				Unit
		Condition	Min	Typ	Max	
V_{IH}	High Level Input Voltage (SSTL Interface)		Vref+0.15		V_{DD}+0.3	V
V_{IL}	Low Level Input Voltage (SSTL Interface)		-0.3		Vref-0.15	V
Vref	Input Reference Voltage		1.15	1.25	1.35	V
ΔV	Hysteresis Voltage			$0.15V_{DD}$		V
I_{IH}	High Level Input Current (Input Buffer)	V_{IN}= V_{DD}	10		60	uA
I_{IL}	High Level Input Current (Input Buffer)	V_{IN}= V_{SS}	10		60	uA
V_{OH}	High Level Output Voltage	I_{OH} = -100uA	V_{DD} -0.2			V
V_{OL}	Low Level Output Voltage	I_{OL} = 100uA			0.2	V

Table 24-6. DDR Normal I/O PAD DC Electrical Characteristics

24.3.3. USB Host 1.1 I/O

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
VDD	Supply voltage		3.0	3.3	3.6	V
VDI	Differential input sensitivity		0.2	-	-	V
VCM	Differential common mode voltage		0.8	-	2.5	V
VIL	Low level input voltage		-	-	0.8	V
VIH	High level input voltage		2.0	-	-	V
VOL	Low level output voltage	RL = 1.5Kohm to +3.6V	-	-	0.3	V
VOH	High level output voltage	RL = 15Kohm to GND	2.8	-	3.6	V
ILZ	Tri-state leakage current		-10	-	10	uA
Cin	Transceiver capacitance	Pin to GND	-	-	10	pF
RPD	Pull down resistance on pins DP/DM	Enable internal resistors	10	-	20	Kohm
RPU	Pull up resistance on DP	Enable internal resistor	1	-	2	Kohm
ZDRV	Driver output impedance(1)	Steady-state drive	39	-	44	Ohm
ZINP	Input impedance		10	-	-	Megaohm
VTERM	Termination voltage for upstream port pull up		3.0	-	3.6	V

Table 24-7. USB PAD DC Electrical Characteristics

24.3.4. Normal Operation

Parameter	Value	Unit	Condition
Operation Voltage	TBD/2.5/3.3	V	
Max. Operating Frequency (FCLK)	533	MHz	CPU Clock
Max. Operating Frequency (BCLK)	133	MHz	Bus Clock
Max. Operating Frequency (PCLK)	66	MHz	Internal Peripheral Clock
Typical normal mode power (VDDi Only)	TBD	mW	Full 3D Engine Operation ARM 200MHz MPEG Core Operation
Idle mode power (VDDi Only)	TBD	mW	CPU Stop, Other Logic running.
Stop mode power (VDDi Only)	TBD	mW	All Clock Stop

Table 24-8. Power Supply Voltage and Current

25. Appendix A - I/O Switching Characteristics.

25.1. Normal 3.3V I/O Switching Chracteristics

[Delays for worst process, 125℃, 3.00V] (CL: Capacitive Loads[pF])

Drive Strength	Parameter	Delay [ns] CL= 50.0pF	Delay Equations [ns]		
			Group1*	Group2*	Group3*
2mA	tR	15.419	0.774 + 0.291*CL	0.754 + 0.293*CL	0.741 + 0.294*CL
	tF	17.238	0.876 + 0.324*CL	0.851 + 0.327*CL	0.833 + 0.328*CL
	tPLH	8.155	1.748 + 0.128*CL	1.750 + 0.128*CL	1.751 + 0.128*CL
	tPHL	9.749	2.088 + 0.154*CL	2.094 + 0.153*CL	2.096 + 0.153*CL
4mA	tR	7.705	0.464 + 0.143*CL	_0.448 + 0.144*CL	_0.433 + 0.145*CL
	tF	8.614	0.533 + 0.160*CL	_0.517 + 0.161*CL	_0.501 + 0.162*CL
	tPLH	4.820	1.599 + 0.065*CL	1.609 + 0.064*CL	1.612 + 0.064*CL
	tPHL	5.751	1.883 + 0.079*CL	1.903 + 0.077*CL	1.910 + 0.077*CL
6mA	tR	5.183	0.449 + 0.094*CL	_0.448 + 0.094*CL_	0.439 + 0.095*CL
	tF	5.815	0.507 + 0.106*CL	_0.513 + 0.106*CL_0	.508 + 0.106*CL
	tPLH	3.904	1.641 + 0.049*CL	1.676 + 0.046*CL	1.698 + 0.044*CL
	tPHL	4.665	1.918 + 0.060*CL	1.965 + 0.055*CL	1.995 + 0.053*CL
8mA	tR	3.917	0.371 + 0.071*CL	_0.376 + 0.071*CL	_0.374 + 0.071*CL
	tF	4.403	0.415 + 0.081*CL	0.428 + 0.080*CL	_0.428 + 0.080*CL
	tPLH	3.322	1.572 + 0.039*CL	1.604 + 0.035*CL	1.627 + 0.034*CL
	tPHL	3.950	1.821 + 0.047*CL	1.864 + 0.043*CL	1.894 + 0.041*CL

Table 25-9. Delays for worst process