



---

## **SPG290 REGISTER GUIDE v1.0**

**12/15/2005**

SUNPLUS TECHNOLOGY CO. reserves the right to change this documentation without prior notice. Information provided by SUNPLUS TECHNOLOGY CO. is believed to be accurate and reliable. However, SUNPLUS TECHNOLOGY CO. makes no warranty for any errors which may appear in this document. Contact SUNPLUS TECHNOLOGY CO. to obtain the latest version of device specifications before placing your order. No responsibility is assumed by SUNPLUS TECHNOLOGY CO. for any infringement of patent or other rights of third parties which may result from its use. In addition, SUNPLUS products are not authorized for use as critical components in life support devices/ systems or aviation devices/systems, where a malfunction or failure of the product may reasonably be expected to result in significant injury to the user, without the express written approval of Sunplus.

SUNPLUS TECHNOLOGY CO., LTD. 19, Innovation First Road, Science-Based Industrial Park, Hsin-Chu, Taiwan, R. O. C.

% 886-3-578-6005

☎ 886-3-578-4418

🌐 [www.sunplus.com.tw](http://www.sunplus.com.tw)

# 1 TABLE OF CONTENT

<b>1</b>	<b>TABLE OF CONTENT .....</b>	<b>2</b>
<b>2</b>	<b>REVISION HISTORY .....</b>	<b>23</b>
<b>3</b>	<b>PROCESSOR BRIEFING .....</b>	<b>24</b>
3.1	ENDIANESS .....	24
3.2	KEY FEATURES.....	24
<b>4</b>	<b>DEBUGGING IN SPG290A.....</b>	<b>26</b>
4.1	OVERVIEW .....	26
4.2	SJTAG MODULE.....	26
4.3	RS232.....	27
4.4	REFERENCE.....	27
<b>5</b>	<b>S<sup>+</sup>CORE IDE .....</b>	<b>28</b>
5.1	INSTALLATION S <sup>+</sup> CORE IDE.....	28
5.2	EXTERNAL DEVICES FOR SIMULATOR .....	28
<b>6</b>	<b>SCENEDIRECTOR - GRAPHIC TOOL .....</b>	<b>29</b>
6.1	FEATURES .....	29
6.2	INSTALLING SCENEDIRECTOR .....	29
6.2.1	<i>System Requirement.....</i>	<i>29</i>
6.2.2	<i>Installing.....</i>	<i>29</i>
<b>7</b>	<b>SUNMIDIAR - SOUND TOOL .....</b>	<b>31</b>
7.1	INSTALLING SUNMIDIAR.....	31
7.1.1	<i>Installing.....</i>	<i>31</i>
7.2	TECHNICAL SUPPORT .....	31
<b>8</b>	<b>SYSTEM OVERVIEW .....</b>	<b>32</b>
8.1	GENERAL DESCRIPTION .....	32
8.2	FEATURES .....	32
8.3	BLOCK DIAGRAM.....	34
<b>9</b>	<b>PLLS AND CKG.....</b>	<b>35</b>
9.1	PLLS .....	35
9.2	CKG .....	35
9.3	CONTROL REGISTERS .....	36
9.3.1	<i>P_CPU_CONF(0x88210000): CPU Clock Config.....</i>	<i>36</i>
9.3.2	<i>P_CKG_SEL_CPU(0x88210004): CPU Clock Frequency Select.....</i>	<i>36</i>
9.3.3	<i>P_CPUAHB_CONF(0x88210008): CPU AHB Clock Config.....</i>	<i>37</i>
9.3.4	<i>P_CKG_SEL_CPUAHB(0x8821000C): AHB Bus Frequency Select.....</i>	<i>37</i>
9.3.5	<i>P_MIU_CONF(0x88210010): MIU Clock Config .....</i>	<i>37</i>

9.3.6	<i>P_CSI_CONF(0x88210018): CSI Clock Config</i>	38
9.3.7	<i>P_CKG_SEL_CSI(0x8821001C): CSI Clock Select</i>	38
9.3.8	<i>P_PPU_CONF(0x88210020): PPU Clock Config</i>	38
9.3.9	<i>P_CKG_SEL_PPU(0x88210024): PPU Clock Frequency Select</i>	39
9.3.10	<i>P_JPG_CONF(0x88210028): JPG Clock Config</i>	39
9.3.11	<i>P_CKG_SEL_JPG(0x8821002C): JPG Clock Frequency Select</i>	39
9.3.12	<i>P_TVE_CONF(0x88210030): TVE Clock Config</i>	40
9.3.13	<i>P_LCD_CONF(0x88210034): LCD Clock Config</i>	40
9.3.14	<i>P_CKG_SEL_LCD(0x88210038): LCD Clock Frequency Select</i>	40
9.3.15	<i>P_SPU_CONF(0x8821003C): SPU Clock Config</i>	41
9.3.16	<i>P_CDS_CONF(0x88210044): CD Servo Clock Config</i>	41
9.3.17	<i>P_BLNDMA_CONF(0x88210050): BLNDMA Clock Config</i>	41
9.3.18	<i>P_APBDMA_CONF(0x88210058): APBDMA Clock Config</i>	41
9.3.19	<i>P_UART_CONF(0x8821005C): UART Clock Config</i>	42
9.3.20	<i>P_USB_CONF(0x88210064): USB Clock Config</i>	42
9.3.21	<i>P_TIMER1_CONF(0x8821006C): Timer1 Clock Config</i>	42
9.3.22	<i>P_TIMER2_CONF(0x88210070): Timer2 Clock Config</i>	43
9.3.23	<i>P_TIMER3_CONF(0x88210074): Timer3 Clock Config</i>	43
9.3.24	<i>P_TIMER4_CONF(0x88210078): Timer4 Clock Config</i>	43
9.3.25	<i>P_TIMER5_CONF(0x8821007C): Timer5 Clock Config</i>	43
9.3.26	<i>P_TIMER6_CONF(0x88210080): Timer6 Clock Config</i>	43
9.3.27	<i>P_WDOG_CONF(0x88210084): Watch Dog Clock Config</i>	44
9.3.28	<i>P_RTC_CONF(0x88210088): RTC Clock Config</i>	44
9.3.29	<i>P_I2S_CONF(0x8821008C): I2S Clock Config</i>	44
9.3.30	<i>P_CKG_SEL_I2S(0x88210090): I2S Master Clock Divided Setting</i>	45
9.3.31	<i>P_I2C_CONF(0x88210094): I2C Clock Config</i>	45
9.3.32	<i>P_SPI_CONF(0x88210098): SPI Clock Config</i>	45
9.3.33	<i>P_SIO_CONF(0x882100A0): SIO Clock Config</i>	45
9.3.34	<i>P_SD_CONF(0x882100A4): SDCard Clock Config</i>	46
9.3.35	<i>P_FLASH_CONF(0x882100A8): NAND-Type Flash Clock Config</i>	46
9.3.36	<i>P_ADC_CONF(0x882100AC): ADC Clock Config</i>	46
9.3.37	<i>P_CKG_SEL_ADC(0x882100B0): ADC Source Clock Select</i>	46
9.3.38	<i>P_PLLV_EN(0x882100B4): Video PLL Enable Setting</i>	46
9.3.39	<i>P_PLLV_FREQUENCY(0x882100B8): Video PLL output Clock Frequency Select</i>	47
9.3.40	<i>P_PLLAU_CONF(0x882100BC): Audio and USB PLL Config</i>	47
9.3.41	<i>P_CLR_KEY_CHANGE(0x882100C0): KEYCHG_WAKEUP Clear Register</i>	48
9.3.42	<i>P_LVR_EN(0x882100C4): Low Voltage Reset Enable</i>	48

9.3.43	<i>P_BUFCTL_CONF(0x882100C8): Buffer Control Clock Config</i>	48
9.3.44	<i>P_LDMDMA_CONF(0x882100CC): LDMDMA Clock Config</i>	48
9.3.45	<i>P_IRQ_CONF(0x882100D0): IRQ Clock Config</i>	48
9.3.46	<i>P_EPP_CONF(0x882100D4): EPP Clock Config</i>	48
9.3.47	<i>P_CKG_SEL_CDS (0x882100D8): CD-Servo Clock Select</i>	49
9.3.48	<i>P_CKG_CONF(0x882100DC): Sleep Mode Clock Select</i>	49
9.3.49	<i>P_TBASE_CONF(0x882100E0): Time Base Clock Config</i>	49
9.3.50	<i>P_CKG_SEL_TIMER (0x882100E4): Each Timer Source Clock Select</i>	49
9.3.51	<i>P_WDOG_STATUS(0x882100E8): Watch Dog Interrupt Status</i>	50
9.3.52	<i>P_MP4_CONF(0x882100EC): MP4 Clock Config</i>	50
9.3.53	<i>P_CKG_SEL_MP4(0x882100F0): MP4 Clock Select</i>	50
9.3.54	<i>P_USB_DETECT_SET (0x882100F4): USB Detection Setting</i>	51
9.3.55	<i>P_UART_DETECT_SET (0x882100F8): UART Detection Setting</i>	51
9.3.56	<i>P_SFTCFG_CONF(0x882100FC): Software Config Clock Config</i>	51
9.3.57	<i>P_ECC_CONF(0x88210100): ECC Clock Config</i>	51
9.3.58	<i>P_PLL_STIME(0x88210104): Stable PLL Output Clock Config</i>	52
9.3.59	<i>P_CKG_STATUS(0x88210110): Wakeup Status for USB/ UART</i>	52
9.3.60	<i>P_CRYSTAL_EN (0x88210114): Crystal Pad Enable Setting</i>	52
<b>10</b>	<b>MEMORY INTERFACE UNIT - MIU</b>	<b>53</b>
10.1	MEMORY MAPPING	54
10.2	SYSTEM REGISTERS : EACH MODULE USED 32K BYTES RANGE.	54
10.3	CONTROL REGISTERS	55
10.3.1	TVE frame buffer start address	55
10.3.2	LCD frame buffer start address	55
10.3.3	<i>P_SPU_START_ADR(0x88070054): SPU Buffer Start Address</i>	56
10.3.4	<i>P_SDRAM_POWER_DOWN (0x8807005C): SDRAM Power Setting</i>	56
10.3.5	<i>P_MIU1_SDRAM_SETTING(0x88070060): MIU1 SDRAM Settings</i>	56
10.3.6	<i>P_MIU1_STATUS(0x8807006C): MIU1 SDRAM Status</i>	57
10.3.7	MP4 RAW Image frame buffer	57
10.3.8	MP4 write frame buffer	57
10.3.9	MP4 VLC buffer	58
10.3.10	<i>P_MP4_FRAME_BUF_HSIZE(0x88070090): MP4 Frame Buffer H-Size Setting</i>	58
10.3.11	<i>P_SDRAM_SETTING2(0x88070094): tRP &amp;tRCD Cycle setup for SDRAM</i>	58
10.4	ARBITER	58
<b>11</b>	<b>BUFFER CONTROL - BUFCTL</b>	<b>59</b>
11.1	CONTROL REGISTERS	59
11.1.1	<i>P_C2P_SETTING(R/W)(0x88090000): CSI to PPU Setting</i>	59

11.1.2	<i>P_PTR_SETTING(R/W)(0x88090004): Control type .....</i>	59
11.1.3	<i>P_CSI_BUF_PTR(R/W)(0x88090008): Software buffer pointer for CSI.....</i>	61
11.1.4	<i>P_TEXT1_BUF_PTR(R/W)(0x8809000C): Software buffer pointer for Text1 .....</i>	61
11.1.5	<i>P_TEXT2_BUF_PTR(R/W)(0x88090010): Software buffer pointer for Text2.....</i>	61
11.1.6	<i>P_TEXT3_BUF_PTR(R/W)(0x88090014): Software buffer pointer for Text3.....</i>	62
11.1.7	<i>P_PPU_BUF_PTR(R/W)(0x88090018): Software buffer pointer for PPU .....</i>	62
11.1.8	<i>P_TVE_BUF_PTR(R/W)(0x88090020): Software buffer pointer for TVE .....</i>	62
11.1.9	<i>P_LCD_BUF_PTR(R/W)(0x88090024): Software buffer pointer for LCD.....</i>	62
11.1.10	<i>P_BUFCTL_STATUS(R/W)(0x88090028): Buffer Control Status .....</i>	63
11.1.11	<i>P_PPU_FRAME_CNT (R/W)(0x8809002C): PPU Frame Counter.....</i>	63
11.1.12	<i>P_PPU_FCNT_INC (R/W)(0x88090030): PPU Frame Counter Increase value .....</i>	63
11.1.13	<i>P_TVE_FRAME_CNT (R/W)(0x88090034): TVE Frame Counter.....</i>	63
11.1.14	<i>P_P2T_SETTING(R/W)(0x8809003C): PPU to TVE buffer control register.....</i>	63
11.1.15	<i>P_P2L_SETTING(R/W)(0x88090040): PPU to LCD buffer control register.....</i>	64
11.1.16	<i>P_MP4RAW_BUF_PTR (R/W)(0x88090044): MP4 RAW Image Buffer Pointer.....</i>	64
11.1.17	<i>P_MP4W_BUF_PTR (R/W)(0x88090048): MP4 Re-Constructed Buffer Pointer .....</i>	64
11.1.18	<i>P_BUFCTL_SETTING(R/W)(0x8809004C): Buffer Controller Interrupt Mask .....</i>	64
11.1.19	<i>P_MP4R_BUF_PTR (R/W)(0x88090050): MP4 reference buffer pointer .....</i>	64
11.1.20	<i>P_MP4V_BUF_PTR(R/W)(0x88090054): MP4 VLC buffer pointer .....</i>	65
11.1.21	<i>P_HW4_SETTING(R/W)(0x88090058): HW4 Hardware Buffer Setting.....</i>	65
11.1.22	<i>P_FRAMENUM_MP4ENC(R/W)(0x8809005C): MP4 encode base on TVE frame number</i>	65
		65
<b>12</b>	<b>2D PICTURE PROCESS UNIT - PPU .....</b>	<b>66</b>
12.1	ORDER OF BYTES .....	67
12.2	COLOR MODE .....	67
12.2.1	Color modes of SPRITE.....	68
12.2.2	Color modes of TEXT .....	68
12.3	PALETTE MEMORY .....	68
12.3.1	Palette Bank.....	68
12.3.2	Pixel format of Palette .....	68
12.4	ATTRIBUTES .....	69
12.4.1	Palette.....	69
12.4.2	Depth .....	69
12.4.3	Character Flip.....	72
12.4.4	Character Size .....	72
12.4.5	Character Data.....	73
12.5	TEXT SCREEN.....	74

12.5.1	Character Mode.....	74
12.5.2	Bitmap Mode.....	74
12.6	TEXT COORDINATE .....	75
12.7	SPRITE COORDINATE.....	78
12.8	TEXT HORIZONTAL MOVEMENT CONTROL .....	79
12.9	TEXT COMPRESSION/EXTENSION CONTROL.....	81
12.9.1	Vertical compression.....	81
12.9.2	Horizontal compression.....	84
12.9.3	Horizontal and Vertical Compression.....	86
12.9.4	Horizontal/Vertical Compression and Movement .....	86
12.10	SPRITE MEMORY .....	87
12.11	TEXT MEMORY .....	87
12.12	WALLPAPER MODE .....	89
12.13	BLENDING EFFECT .....	90
12.14	32768/65536 HIGH COLOR MODE.....	90
12.15	CHARACTER ATTRIBUTES .....	90
12.16	BITMAP MODE ATTRIBUTES.....	91
12.17	ADDRESSING MODE .....	92
12.18	PPU REGISTER SET .....	93
<b>13</b>	<b>DISPLAY UNIT – TV ENCODER .....</b>	<b>107</b>
13.1	DESCRIPTION .....	107
13.2	TV SYSTEM AND SCREEN MODE.....	107
13.3	TEXT SCREEN INTRODUCTION.....	107
13.3.1	Q-VGA Mode .....	107
13.3.2	H-VGA Mode .....	107
13.3.3	VGA Mode .....	108
13.4	FRAME BUFFER.....	108
13.5	DISPLAY .....	108
13.6	TVE REGISTERS.....	108
13.6.1	P_TV_Control(0x88030000): .....	108
13.6.2	P_TV_Saturation(0x88030004) & P_TV_Hue(0x88030008):.....	109
13.6.3	P_TV_FADE(0x8803000C): Fade Effect Control Registers.....	109
13.6.4	P_LP_FILTER_SEL(0x88030010): Low Pass Filter Select Control Registers.....	110
13.6.5	P_LINE_COUNTER(0x88030020): .....	110
13.6.6	Light Gun Interface.....	110
<b>14</b>	<b>SOUND PROCESS UNIT - SPU .....</b>	<b>112</b>
14.1	GENERAL DESCRIPTION .....	112

14.2	FEATURE .....	112
14.3	BLOCK DIAGRAM.....	113
14.4	INTERNAL MEMORY MAPPING .....	114
14.5	GAIN CONTROL.....	115
14.6	FUNDAMENTALS .....	116
14.7	CONTROL REGISTER.....	118
14.8	INTERNAL SRAM .....	133
14.9	S/W PROGRAMMING FLOW EXAMPLE.....	147
<b>15</b>	<b>APBDMA.....</b>	<b>149</b>
15.1	ARCHITECTURE.....	149
15.2	APB BRIDGE.....	149
15.3	DMA CONTROLLER .....	149
15.4	CONTROL REGISTERS .....	150
15.4.1	<i>P_CH_BUSY_STS(0x88080000): Channel Busy Status .....</i>	<i>150</i>
15.4.2	<i>P_CH_IRQ_STS(0x88080004): Channel Interrupt Status .....</i>	<i>150</i>
15.4.3	<i>Buffer A Start Address.....</i>	<i>151</i>
15.4.4	<i>Buffer A End Address.....</i>	<i>151</i>
15.4.5	<i>APB Module Access Port Start Address.....</i>	<i>151</i>
15.4.6	<i>Buffer B Start Address.....</i>	<i>151</i>
15.4.7	<i>Buffer B End Address.....</i>	<i>152</i>
15.4.8	<i>P_CH1_SETTING (0x8808006C): Channel One Setting .....</i>	<i>152</i>
15.4.9	<i>P_CH2_SETTING (0x88080070): Channel Two Setting .....</i>	<i>152</i>
15.4.10	<i>P_CH3_SETTING (0x88080074): Channel Three Setting.....</i>	<i>153</i>
15.4.11	<i>P_CH4_SETTING (0x88080078): Channel Four Setting.....</i>	<i>154</i>
15.4.12	<i>P_CH_SW_RST (0x8808007C): Channel Software Reset .....</i>	<i>155</i>
15.5	DMA END.....	155
<b>16</b>	<b>BITBLT DMA WITH ALPHA BLENDING OPERATION - BLNDMA.....</b>	<b>156</b>
16.1	FEATURES .....	156
16.1.1	<i>DRAM-to-DRAM DMA(BitBlt) Function .....</i>	<i>156</i>
16.1.2	<i>DMA Pattern Fill Function.....</i>	<i>156</i>
16.1.3	<i>Transparent Filter (Color Key) Function .....</i>	<i>156</i>
16.1.4	<i>Blending Function .....</i>	<i>157</i>
16.1.5	<i>Clipping Function.....</i>	<i>157</i>
16.1.6	<i>YUV2RGB Conversion Function .....</i>	<i>157</i>
16.2	ARCHITECTURE.....	158
16.3	CONTROL REGISTERS .....	159
16.3.1	<i>P_BLNDMA_SRCA_ADDR(R/W) (0x880D0000): DMA Source A Starting Address.....</i>	<i>159</i>

16.3.2	<i>P_BLNDMA_SRCB_ADDR(R/W) (0x880D0004): DMA Source B Starting Address.....</i>	159
16.3.3	<i>P_BLNDMA_DEST_ADDR(R/W) (0x880D0008): DMA Destination Starting Address....</i>	159
16.3.4	<i>P_BLNDMA_WIDTH_HEIGH(R/W) (0x880D000C): DMA Transfer Width .....</i>	159
16.3.5	<i>P_BLNDMA_FILL_PAT(R/W) (0x880D0010): DMA Fill Pattern .....</i>	159
16.3.6	<i>P_BLNDMA_CONTROL_1(R/W) (0x880D0014): DMA Operation Mode .....</i>	159
16.3.7	<i>P_BLNDMA_IRQ_CONTROL(R/W) (0x880D0018): DMA Interrupt Control .....</i>	160
16.3.8	<i>P_BLNDMA_BLEND_FACTOR(R/W) (0x880D001C): DMA Blending Factor.....</i>	160
16.3.9	<i>P_BLNDMA_TRANSPARENT(R/W) (0x880D0020): DMA Transparent Filter Pattern ...</i>	160
16.3.10	<i>P_BLNDMA_ADDR_MODE(R/W)(0x880D0024): DMA Address Mode Control.....</i>	160
16.3.11	<i>P_BLNDMA_CONTROL_2(R/W) (0x880D0028): DMA Control 2 .....</i>	161
16.3.12	<i>P_BLNDMA_ABASE_ADDR(R/W) (0x880D0030): DMA Source A Base Address.....</i>	161
16.3.13	<i>P_BLNDMA_AOFFSET_XY(R/W) (0x880D0034): DMA Source A Offset Address XY</i>	161
16.3.14	<i>P_BLNDMA_A_BG(R/W) (0x880D0038): DMA Source A Background Width/Height</i>	161
16.3.15	<i>P_BLNDMA_BBASE_ADDR(R/W) (0x880D0040): DMA Source B Base Address.....</i>	162
16.3.16	<i>P_BLNDMA_BOFFSET_XY(R/W) (0x880D0044): DMA Source B Offset Address XY</i>	162
16.3.17	<i>P_BLNDMA_B_BG(R/W) (0x880D0048): DMA Source B Background Width/Height</i>	162
16.3.18	<i>P_BLNDMA_DBASE_ADDR(R/W) (0x880D0050): DMA Destination Base Address .</i>	162
16.3.19	<i>P_BLNDMA_DOFFSET_XY(R/W) (0x880D0054): DMA Destination Offset Address XY</i>	163
16.3.20	<i>P_BLNDMA_D_BG(R/W)(0x880D0058): DMA Destination Background Width/Heigh</i>	163
<b>17</b>	<b>LOCAL DATA MEMORY - LDM .....</b>	<b>164</b>
17.1	ARCHITECTURE.....	164
17.2	LDMDMA CONTROLLER .....	164
17.3	CONTROL REGISTERS .....	165
17.3.1	<i>P_DMA_CTRL (0x880C0000): DMA Control.....</i>	165
17.3.2	<i>P_DMA_STATUS (0x880C0004): DMA Status.....</i>	165
17.3.3	<i>P_MIU_START_ADDR (0x880C0008): MIU Start Address.....</i>	165
17.3.4	<i>P_MIU_END_ADDR (0x880C000C): MIU End Address .....</i>	165
17.3.5	<i>P_LDM_START_ADDR (0x880C0010): LDM Start Address .....</i>	166
17.3.6	<i>P_LDM_END_ADDR (0x880C0014): LDM End Address .....</i>	166
<b>18</b>	<b>PERIPHERAL INTERRUPT CONTROLLER.....</b>	<b>167</b>
18.1	INTRODUCTION.....	167
18.2	FEATURE .....	167
18.3	CONTROL REGISTER.....	167
18.3.1	<i>P_INTPND (R) (0x880A0000): Interrupt Request Status Register. ....</i>	167
18.3.2	<i>P_INTPND_H (R) (0x880A0004): Interrupt Request Status High Byte .....</i>	168

18.3.3	<i>P_I_PMST (R/W) (0x880A0008):</i>	168
18.3.4	<i>IRQ Priority Register:</i>	168
18.4	VECADDR	169
18.5	INTERRUPT SOURCES	169
<b>19</b>	<b>SOFTWARE CONFIG - SFTCFG:</b>	<b>171</b>
19.1	FUNCTION DESCRIPTION	171
19.2	ARCHITECTURE	171
19.3	CONTROL REGISTERS	172
19.3.1	<i>P_GPIO_DEVICE_SET(0x88200000): Software Configuration for Device Selection</i>	172
19.3.2	<i>P_GPIO_FUNC_SET(0x88200004): Software Configuration for Function Selection</i>	172
19.3.3	<i>P_GPIO_DRAM_SET(0x88200008): Software Configuration for DRAM Selection</i>	173
19.3.4	<i>P_GPIO_OUTPUT(0x8820000C): Software Configuration for Transceiver Output Selection</i>	173
19.3.5	<i>P_GPIO_LIGHTPEN_PORT(0x88200010): Software Configuration for LightPen Selection</i>	173
19.3.6	<i>P_GPIO_TFT_PORT(0x88200014): TFT GPIO Port Data</i>	174
19.3.7	<i>P_GPIO_TFT_OE(0x88200018): TFT GPIO Port Output Setting</i>	174
19.3.8	<i>P_GPIO_TFT_PU(0x8820001C): TFT GPIO Port Pull Up Setting</i>	174
19.3.9	<i>P_GPIO_TFT_PD(0x88200020): TFT GPIO Port Pull Down Setting</i>	174
19.3.10	<i>P_GPIO_CSI_PORT(0x88200024): CSI GPIO Port Control</i>	174
19.3.11	<i>P_GPIO_CSI_PULL(0x88200028): CSI GPIO Port Pull Status</i>	175
19.3.12	<i>P_GPIO_NFLASH_PORT(0x8820002C): NFLASH GPIO Port Control</i>	175
19.3.13	<i>P_GPIO_NFLASH_PULL(0x88200030): NFLASH GPIO Port Pull Status</i>	175
19.3.14	<i>P_GPIO_JTAG_PORT(0x88200034): JTAG GPIO Port Control</i>	175
19.3.15	<i>P_GPIO_GLOBAL_PORT(0x88200038): GLOBAL GPIO Port Control</i>	176
19.3.16	<i>P_GPIO_USB_PORT(0x8820003C): USB GPIO Port Control</i>	176
19.3.17	<i>P_GPIO_UART_PORT(0x88200040): UART GPIO Port Control</i>	176
19.3.18	<i>P_GPIO_I2C_PORT(0x88200044): I2C GPIO Port Control</i>	177
19.3.19	<i>P_GPIO_ADC_PORT(0x88200048): ADC GPIO Port Control</i>	177
19.3.20	<i>P_GPIO_CDSERVO_PORT(0x8820004C): CD-Servo GPIO Port Control</i>	177
19.3.21	<i>P_GPIO_DRAM_PORT(0x88200050): DRAM GPIO Port Control</i>	178
19.3.22	<i>P_GPIO_ADC_AEN(0x88200054): ADC GPIO Port Analog Input Control</i>	178
19.3.23	<i>P_GPIO_TFT_INPUT(0x88200064): TFT GPIO Input Control</i>	178
19.3.24	<i>P_GPIO_CSI_INPUT(0x88200068): CSI GPIO Input Control</i>	178
19.3.25	<i>P_GPIO_NFLASH_INPUT(0x8820006C): NFLASH GPIO Input Control</i>	179
19.3.26	<i>P_GPIO_JCD_INPUT(0x88200070): JTAG/CDServo/DRAM GPIO Input Control</i>	179
19.3.27	<i>P_GPIO_GUUI_INPUT(0x88200074):</i>	179

19.3.28	<i>P_GPIO_ADC_INPUT(0x88200078)</i> .....	179
19.3.29	<i>P_GPIO_TFT_INT(0x88200080)</i> .....	179
19.3.30	<i>P_GPIO_CSI_INT(0x88200084)</i> .....	180
19.3.31	<i>P_GPIO_NFLASH_INT(0x88200088)</i> .....	180
19.3.32	<i>P_GPIO_JTAG_INT(0x8820008C)</i> .....	181
19.3.33	<i>P_GPIO_GLOBAL_INT(0x88200090)</i> .....	181
19.3.34	<i>P_GPIO_UART_INT(0x88200094)</i> .....	182
19.3.35	<i>P_GPIO_I2C_INT(0x88200098)</i> .....	182
19.3.36	<i>P_GPIO_ADC_INT(0x8820009C)</i> .....	183
19.3.37	<i>P_GPIO_USBDDET_INT(0x882000A0)</i> .....	183
19.3.38	<i>P_GPIO_NFSDSPI_EN(0x882000A4)</i> .....	184
19.3.39	<i>P_GPIO_LVD_CTL(0x882000A8)</i> .....	184
<b>20</b>	<b>TIMER.....</b>	<b>186</b>
20.1	FEATURES .....	186
20.2	ARCHITECTURE.....	186
20.3	CONTROL REGISTERS .....	188
20.3.1	<i>P_TimerXCtrl(0x8816X000)</i> .....	188
20.3.2	<i>P_CPP_CTRL(0x8816X004)</i> .....	189
20.3.3	<i>P_Preload_REGS(0x8816X008)</i> .....	189
20.3.4	<i>P_CPP_REGS(0x8816X00C)</i> .....	189
20.3.5	<i>P_CPP_UPCOUNT(0x8816X010)</i> .....	189
<b>21</b>	<b>REAL TIME CLOCK - RTC.....</b>	<b>190</b>
21.1	FEATURES .....	190
21.2	ARCHITECTURE.....	190
21.3	CONTROL REGISTERS.....	190
21.3.1	<i>P_RTC_SEC(0x88166000): RTC Second Setup Registers.</i> .....	190
21.3.2	<i>P_RTC_MIN(0x88166004): RTC Minute Setup Registers.</i> .....	190
21.3.3	<i>P_RTC_HOUR(0x88166008): RTC Hour Setup Registers.</i> .....	190
21.3.4	<i>P_ALM_SEC(0x8816600C): Alarm Second Setup Registers.</i> .....	191
21.3.5	<i>P_ALM_MIN(0x88166010): Alarm Minute Setup Registers.</i> .....	191
21.3.6	<i>P_ALM_HOUR(0x88166014): Alarm Hour Setup Registers.</i> .....	191
21.3.7	<i>P_RTC_CTRL(0x88166018): RTC Enable Registers.</i> .....	191
21.3.8	<i>P_RTC_STATUS(0x8816601C): RTC Status Registers.</i> .....	191
21.3.9	<i>P_TMB_Ctrl(0x88166020): Time Base Control Registers.</i> .....	192
21.3.10	<i>P_TMB_Status(0x88166024): Time Base Status Registers.</i> .....	193
21.3.11	<i>P_TMB_Reset(0x88166028): Time Base Reset Registers.</i> .....	194
<b>22</b>	<b>WATCH DOG .....</b>	<b>195</b>

22.1	FEATURES .....	195
22.2	ARCHITECTURE.....	195
22.3	CONTROL REGISTERS.....	195
22.3.1	<i>P_WDOG_CTRL(0x88170000): Watch Dog Control Registers.</i> .....	195
22.3.2	<i>P_WDOG_CYCLE(0x88170004): Reset Counter Number.</i> .....	195
22.3.3	<i>P_WDOG_CLEAR(0x88170008): Reset Counter Number.</i> .....	195
<b>23</b>	<b>SLEEP/WAKEUP .....</b>	<b>196</b>
23.1	SLEEP.....	196
23.2	WAKEUP .....	196
<b>24</b>	<b>ANALOG TO DIGITAL CONVERTER - ADC.....</b>	<b>198</b>
24.1	FEATURES .....	198
24.2	ARCHITECTURE.....	198
24.3	CONTROL REGISTERS .....	199
24.3.1	<i>P_ADC_SETUP(0x881A0000): ADC Setup</i> .....	199
24.3.2	<i>P_MIC_GAIN(0x881A0004): MIC GAIN</i> .....	200
24.3.3	<i>P_ADC_CLOCK_SET(0x881A0008): ADC Clock Setup</i> .....	200
24.3.4	<i>P_ADC_HOLD_SETUP(0x881A000C): Sample Hold Setup</i> .....	200
24.3.5	<i>P_ADC_MIC_CTRL(0x881A0010):</i> .....	200
24.3.6	<i>P_ADC_MIC_CTRL2(0x881A0014):</i> .....	201
24.3.7	<i>P_ADC_Read_Data(0x881A0018):</i> .....	202
24.3.8	<i>P_ASP_Read_Data(0x881A001C):</i> .....	202
24.3.9	<i>P_MIC_Read_Data(0x881A0020):</i> .....	203
<b>25</b>	<b>INTERACTIVE INFORMATION SERVICES - I2S .....</b>	<b>204</b>
25.1	FEATURES .....	204
25.2	ARCHITECTURE.....	204
25.3	CONTROL REGISTERS .....	204
25.3.1	<i>P_I2S_CR(R/W)(0x88140000): I2S control</i> .....	204
25.3.2	<i>P_I2S_IRQ_STS(R/W)(0x88140004): I2S IRQ Status</i> .....	205
25.3.3	<i>P_I2S_FIFO(R/W)(0x88140008): I2S FIFO Config</i> .....	205
25.3.4	<i>P_I2S_RDATA(R/W)(0x8814000C): I2S receive data</i> .....	206
<b>26</b>	<b>INTER-INTEGRATED CIRCUIT - I2C .....</b>	<b>207</b>
26.1	OVERVIEW .....	207
26.2	INITIALIZATION .....	207
26.3	CONTROL REGISTERS .....	208
26.3.1	<i>P_I2C_CR(R/W)(0x88130020): I2C configuration register.</i> .....	208
26.3.2	<i>P_I2C_INTR(R/W)(0x88130024): I2C interrupt register.</i> .....	209
26.3.3	<i>P_I2C_CVR(R/W)(0x88130028): I2C clock setting register.</i> .....	209

26.3.4	<i>P_I2C_ID(R/W)(0x8813002C): I2C ID register.....</i>	209
26.3.5	<i>P_I2C_ADDR(R/W)(0x88130030): I2C port address register.....</i>	209
26.3.6	<i>P_I2C_WDATA(R/W)(0x88130034): I2C write data register.....</i>	209
26.3.7	<i>P_I2C_RDATA(R/W)(0x88130038): I2C read data register.....</i>	209
<b>27</b>	<b>SERIAL PERIPHERAL INTERFACE - SPI .....</b>	<b>210</b>
27.1	GENERAL DESCRIPTION .....	210
27.2	FEATURES .....	210
27.3	ARCHITECTURE.....	211
27.4	CONTROL REGISTERS.....	211
27.4.1	<i>P_SPI_Control(0x88110000).....</i>	211
27.4.2	<i>P_SPI_TX_Status (0x88110004).....</i>	212
27.4.3	<i>P_SPI_TX_DATA (0x88110008).....</i>	212
27.4.4	<i>P_SPI_RX_Status (0x8811000C).....</i>	212
27.4.5	<i>P_SPI_RX_DATA (0x88110008).....</i>	213
27.4.6	<i>P_SPI_RX_Status (0x8811000C).....</i>	213
<b>28</b>	<b>SERIAL INTERFACE I/O - SIO.....</b>	<b>217</b>
28.1	GENERAL DESCRIPTION .....	217
28.2	SIO PROTOCOL .....	217
28.3	SIO FEATURE .....	217
28.4	SIO CONTROL REGISTERS.....	218
28.4.1	<i>P_SIO_Control(0x88120000).....</i>	218
28.4.2	<i>P_SIO_AutoTx (0x88120004): Automatic transmit word number.....</i>	219
28.4.3	<i>P_SIO_Addr(0x88120008): SIO Start Address Register .....</i>	220
28.4.4	<i>P_SIO_DATA(0x8812000C): SIO Data Register.....</i>	220
28.5	SIO INTERRUPT PROCESSING .....	220
28.6	SIO USAGE PROCEDURE .....	221
<b>29</b>	<b>UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER-UART .....</b>	<b>225</b>
29.1	BLOCK DIAGRAM.....	225
29.2	CONTROL REGISTERS.....	225
29.2.1	<i>P_UART_Data(R/W)(0x88150000): UART Data Tx/Rx Register.....</i>	225
29.2.2	<i>P_UART_ERR(R/W)(0x88150004):UART Tx &amp; Rx Error Flag Register.....</i>	226
29.2.3	<i>P_UART_Ctrl(R/W)(0x88150008):UART Control Register.....</i>	227
29.2.4	<i>P_UART_BaudRate(R/W)(0x8815000C):UART Baud Rate Setup Register.....</i>	228
29.2.5	<i>P_UART_Status(R/W)(0x88150010):UART Status Register .....</i>	228
<b>30</b>	<b>UNIVERSAL SERIAL BUS – USB 1.1 .....</b>	<b>230</b>
30.1	FEATURES .....	230
30.2	USB DEVICE.....	230

30.3	USB MINI HOST .....	231
30.4	USB DEVICE CONTROL REGISTER .....	231
30.4.1	<i>P_USBD_Config(0x881B00C0): USB Configuration Register.....</i>	<i>231</i>
30.4.2	<i>P_USBD_Device(0x881B015C): USB Device Register.....</i>	<i>231</i>
30.4.3	<i>P_USBD_Function(0x881B00C4): USB Function Register. ....</i>	<i>232</i>
30.4.4	<i>P_USBD_DMAINT(0x881B0164): USB DMA Interrupt Register.....</i>	<i>233</i>
30.4.5	<i>P_USBD_PMR(0x881B00C8): USB Power Management Register.....</i>	<i>233</i>
30.4.6	<i>P_USBD_EP0Data(0x881B00CC): USB Endpoint0 Data Register.....</i>	<i>234</i>
30.4.7	<i>P_USBD_BIData(0x881B00D0): USB Bulk In Data Register. ....</i>	<i>234</i>
30.4.8	<i>P_USBD_BOData(0x881B00D4): USB Bulk Out Data Register. ....</i>	<i>234</i>
30.4.9	<i>P_USBD_INTINData(0x881B00D8): USB Interrupt In Data Register.....</i>	<i>234</i>
30.4.10	<i>P_USBD_NullPkt(0x881B0160): USB Null Packet Register.....</i>	<i>234</i>
30.4.11	<i>P_USBD_EPEvent(0x881B00DC): USB Endpoint Event Register. ....</i>	<i>234</i>
30.4.12	<i>P_USBD_GLOINT(0x881B00E0): USB Global Interrupt Register.....</i>	<i>236</i>
30.4.13	<i>P_USBD_INTEN(0x881B00E4): USB Interrupt Enable Register. ....</i>	<i>237</i>
30.4.14	<i>P_USBD_INT(0x881B00E8): USB Interrupt Register.....</i>	<i>238</i>
30.4.15	<i>P_USBD_SCINTEN(0x881B00EC): USB Standard Command Interrupt Enable Register.</i>	<i>240</i>
30.4.16	<i>P_USBD_SCINT(0x881B00F0): USB Standard Command Interrupt Register.....</i>	<i>241</i>
30.4.17	<i>_USBD_EPAutoSet(0x881B00F4): USB Endpoint Auto Set Register. ....</i>	<i>242</i>
30.4.18	<i>P_USBD_EPSetStall(0x881B00F8): USB Endpoint Set Stall Register.....</i>	<i>242</i>
30.4.19	<i>P_USBD_EPBufClear(0x881B00FC): USB Endpoint Buffer Clear Register.....</i>	<i>243</i>
30.4.20	<i>P_USBD_EPEvntClear(0x881B0100): USB Endpoint Event Clear Register.....</i>	<i>243</i>
30.4.21	<i>P_USBD_EP0WrtCount(0x881B0104): USB Endpoint0 Write Count Register.....</i>	<i>243</i>
30.4.22	<i>P_USBD_BOWrtCount(0x881B0108): USB Bulk Out Write Count Register.....</i>	<i>244</i>
30.4.23	<i>P_USBD_EP0BufPointer(0x881B010C): USB Endpoint0 Buffer Pointer Register.....</i>	<i>244</i>
30.4.24	<i>P_USBD_BIBufPointer(0x881B0110): USB Bulk In Buffer Pointer Register.....</i>	<i>244</i>
30.4.25	<i>P_USBD_BOBufPointer(0x881B0114): USB Bulk Out Buffer Pointer Register. ....</i>	<i>244</i>
30.4.26	<i>P_USBD_EP0RTR(0x881B0118): USB Endpoint0 bmRequestType Register.....</i>	<i>244</i>
30.4.27	<i>P_USBD_EP0RR(0x881B011C): USB Endpoint0 bRequest Register.....</i>	<i>245</i>
30.4.28	<i>P_USBD_EP0VR(0x881B0120): USB Endpoint0 wValue Register.....</i>	<i>245</i>
30.4.29	<i>P_USBD_EP0IR(0x881B0124): USB Endpoint0 wIndex Register. ....</i>	<i>245</i>
30.4.30	<i>P_USBD_EP0LR(0x881B0128): USB Endpoint0 wLength Register. ....</i>	<i>245</i>
30.4.31	<i>P_USBD_DMAWrtCount(0x881B0140): USB DMA Byte Count Register.....</i>	<i>245</i>
30.4.32	<i>P_USBD_DMAACKCount(0x881B0144): USB DMAACK Count Register.....</i>	<i>245</i>
30.4.33	<i>P_USB_EPStall(0x881B0150): USB Endpoint Stall Bit Register. ....</i>	<i>246</i>
30.5	USB HOST CONTROL REGISTER.....	246

30.5.1	<i>P_USBH_Config(0x881C0000): USB Host Configuration Register.</i>	246
30.5.2	<i>P_USBH_TimeConfig(0x881C0004): USB Host Timing Configuration Register.</i>	247
30.5.3	<i>P_USBH_Data(0x881C0008): USB Host Data Register.</i>	247
30.5.4	<i>P_USBH_Transfer(0x881C000C): USB Host Transfer Register.</i>	248
30.5.5	<i>P_USBH_DveAddr(0x881C0010): USB Device Address Register.</i>	248
30.5.6	<i>P_USBH_DveEP(0x881C0014): USB Device Endpoint Register.</i>	248
30.5.7	<i>P_USBH_TXCount(0x881C0018): USB Host Transmit Count Register.</i>	249
30.5.8	<i>P_USBH_RXCount(0x881C001C): USB Receive Count Register.</i>	249
30.5.9	<i>P_USBH_FIFOInPointer(0x881C0020): USB Host FIFO Input Pointer Register.</i>	249
30.5.10	<i>P_USBH_FIFOutPointer(0x881C0024): USB Host FIFO Output Pointer Register.</i>	249
30.5.11	<i>P_USBH_AutoInByteCount(0x881C0028): USB Host Automatic In Transaction Byte Count Register.</i>	249
30.5.12	<i>P_USBH_AutoOutByteCount(0x881C002C): USB Host Automatic Out Transaction Byte Count Register.</i>	249
30.5.13	<i>P_USBH_AutoTrans(0x881C0030): USB Host Auto Transfer Register.</i>	250
30.5.14	<i>P_USBH_Status(0x881C0034): USB Host Status Register.</i>	250
30.5.15	<i>P_USBH_INT(0x881C0038): USB Host Interrupt Register.</i>	252
30.5.16	<i>P_USBH_INTEN(0x881C003C): USB Host Interrupt Enable Register.</i>	254
30.5.17	<i>P_USBH_SoftRST(0x881C0044): USB Software Reset Register/Device Plug Out Register.</i>	255
30.5.18	<i>P_USBH_INAckCount(0x881C005C): USB IN ACK Count Register.</i>	256
30.5.19	<i>P_USBH_OutAckCount(0x881C0060): USB Out ACK Count Register.</i>	256
30.5.20	<i>P_USBH_RSTAckCount(0x881C0064): USB Reset ACK Count Register.</i>	256
30.5.21	<i>P_USBH_Dreadback(0x881C006C): USB D+/D- Readback Register.</i>	256
<b>31</b>	<b>SECURE DIGITAL CARD – SDCARD CONTROLLER</b>	<b>257</b>
31.1	FEATURES	257
31.2	BLOCK DIAGRAM	258
31.3	CONTROL REGISTER	258
31.3.1	<i>P_SD_DATATx (R/W)(0x88180000) : SD Card Data Transmit register</i>	258
31.3.2	<i>P_SD_DATARx(R) (0x88180004): SD Card Data Receive register</i>	258
31.3.3	<i>P_SD_COMMAND(R/W)(0x88180008): SD Card Command register</i>	259
31.3.4	<i>P_SD_ARGUMENT(R/W)(0x8818000C): SD Card argument register</i>	260
31.3.5	<i>P_SD_STATUS (R)(0x88180014): SD Card Controller status register</i>	260
31.3.6	<i>P_SD_RESPONSE (R) (0x88180010): SD Card Response register</i>	261
31.3.7	<i>P_SD_CONTROL(R/W)(0x88180018): SD Card Control Register</i>	262
31.3.8	<i>13.10 P_SD_INTEN(R/W) (0x8818001C): SD Card Interrupt Register</i>	262
<b>32</b>	<b>UNIVERSAL FILE SYSTEM – UFAT LIBRARY</b>	<b>264</b>

32.1	STRUCTURE.....	264
32.2	API FUNCTIONS .....	264
32.2.1	<i>open</i> .....	264
32.2.2	<i>close</i> .....	265
32.2.3	<i>read</i> .....	265
32.2.4	<i>write</i> .....	266
<b>33</b>	<b>NAND-TYPE FLASH .....</b>	<b>267</b>
33.1	INTRODUCTION.....	267
33.2	FEATURE.....	267
33.3	BLOCK DIAGRAM .....	267
33.4	INTERFACE SIGNAL.....	267
33.4.1	<i>APB Bus Signals</i> .....	267
33.4.2	<i>Flash Bus Signals</i> .....	267
33.4.3	<i>Other Signals</i> .....	268
33.5	CONTROL REGISTERS .....	268
33.5.1	<i>P_FL_CR(R/W)(0x8800F000): Control Register</i> .....	268
33.5.2	<i>P_FL_CLE(R/W)(0x8800F004): Command Latch enable register</i> .....	269
33.5.3	<i>P_FL_ALE(R/W)(0x8800F008): Address latch enable register</i> .....	270
33.5.4	<i>P_FL_WD(R/W)(0x8800F00C): Write data register</i> .....	270
33.5.5	<i>P_FL_RD(R/W)(0x8800F010): Read data register</i> .....	270
33.5.6	<i>P_FL_INTEN(R/W)(0x8800F014): Interrupt enable register</i> .....	270
33.5.7	<i>P_FL_INTSTS(R/C)(0x8800F018): Interrupt status register</i> .....	271
33.5.8	<i>P_FL_TRUELP(R/W)(0x8800F01C): True line parity register</i> .....	272
33.5.9	<i>P_FL_TRUECP(R/W)(0x8800F020): True column parity register</i> .....	272
33.5.10	<i>P_FL_CALLP(R)(0x8800F024): Calculate line a parity register</i> .....	273
33.5.11	<i>P_FL_CALCP(R)(0x8800F028): Calculate column parity register</i> .....	273
33.5.12	<i>P_FL_ECCSTS(R)(0x8800F02C): ECC status register</i> .....	273
33.5.13	<i>P_FL_CALECC(W)(0x8800F030): ECC control register</i> .....	274
<b>34</b>	<b>TFT LCD CONTROL.....</b>	<b>275</b>
34.1	INTRODUCTION .....	275
34.2	OUTPUT INTERFACE .....	275
34.3	SUPPORTED DATA FORMAT.....	275
34.4	CLOCK FREQUENCY .....	275
34.5	REGISTER DEFINITION .....	275
34.5.1	<i>P_TFT_CTRL(0x88040000):</i> .....	275
34.5.2	<i>P_DATA_FMT(0x88040004):</i> .....	276
34.5.3	<i>P_HOR_ACT(0x88040008):</i> .....	276

34.5.4	<i>P_HOR_FBLK(0x8804000C):</i> .....	276
34.5.5	<i>P_HOR_BBLK(0x88040010):</i> .....	276
34.5.6	<i>P_HOR_SYNCW(0x88040014):</i> .....	277
34.5.7	<i>P_VER_ACT(0x88040018):</i> .....	277
34.5.8	<i>P_VER_FBLK(0x8804001C):</i> .....	277
34.5.9	<i>P_VER_BBLK(0x88040020):</i> .....	277
34.5.10	<i>P_VER_SYNCW(0x88040024):</i> .....	277
34.5.11	<i>P_FB_ATTRIB(0x88040028):</i> .....	278
34.5.12	<i>P_STR_LNO(0x8804002C):</i> .....	278
34.5.13	<i>P_STR_PNO(0x88040030):</i> .....	278
34.5.14	<i>P_PIX_NUM(0x88040034):</i> .....	278
34.5.15	<i>P_DUMMY_PIX (0x88040038):</i> .....	278
34.5.16	<i>P_TFT_ST (0x8804003C):</i> .....	279
34.5.17	<i>P_DATA_SEQ (0x88040040):</i> .....	279
34.5.18	<i>P_TFT_INT (0x88040050):</i> .....	280
<b>35</b>	<b>STN LCD CONTROL.....</b>	<b>281</b>
35.1	INTRODUCTION .....	281
35.2	FEATURE .....	281
35.3	INTERFACE SIGNALS .....	281
35.4	STN LCD CONTROL REGISTER.....	281
35.4.1	<i>P_STN_CTRL (0x88041000):</i> .....	281
35.4.2	<i>P_LCDCLK (0x88041004):</i> .....	282
35.4.3	<i>P_STN_SEG_NUM (0x88041008):</i> .....	283
35.4.4	<i>P_STN_COM_NUM (0x8804100C):</i> .....	283
35.4.5	<i>P_STN_STR_LNO(0x88041010):</i> .....	283
35.4.6	<i>P_STN_STR_PNO(0x88041014):</i> .....	283
35.4.7	<i>P_STN_STR_PNO(0x88041018):</i> .....	284
35.4.8	<i>P_STN_LT(0x8804101C):</i> .....	284
35.4.9	<i>P_STN_FM(0x88041020):</i> .....	285
35.4.10	<i>P_STN_BUF_CTRL(0x88041024):</i> .....	286
35.4.11	<i>P_STN_ATTR (0x88041028):</i> .....	286
35.4.12	<i>P_STN_MSC(0x8804102C):</i> .....	287
<b>36</b>	<b>CD SERVO.....</b>	<b>288</b>
36.1	BLOCK DIAGRAM OF CD SERVO .....	288
36.2	CONTROL REGISTERS .....	288
36.2.1	<i>P_IF51_CONFIG (R/W) (0X88060000) :</i> .....	288
36.2.2	<i>P_IF51_ADDR (R/W) (0X88060004) :</i> .....	289

36.2.3	<i>P_IF51_DATA</i> (R/W) (0X88060008) : .....	289
36.2.4	<i>P_IF51_CONTROL</i> (R/W) (0X8806000C) : .....	289
36.2.5	<i>P_CDDSP_CONFIG</i> (R/W) (0X88060040) : .....	289
36.2.6	<i>P_CDDSP_CONTROL</i> (R/W) (0X88060044) : .....	291
36.2.7	<i>P_CDDSP_SEEK_MM</i> (R/W) (0X88060048) : .....	291
36.2.8	<i>P_CDDSP_SEEK_SS</i> (R/W) (0X8806004C) : .....	291
36.2.9	<i>P_CDDSP_SEEK_FF</i> (R/W) (0X88060050) : .....	292
36.2.10	<i>P_CDDSP_STATUS</i> (R/W) (0X88060054) : .....	292
36.2.11	<i>P_CD_BUF_BTMM_ADDR</i> (R/W) (0X88060060) : .....	292
36.2.12	<i>P_CD_BUF_TOP_ADDR</i> (R/W) (0X88060064) : .....	292
36.2.13	<i>P_CD_WRITE_PTR</i> (R/W) (0X88060068) : .....	292
36.2.14	<i>P_CD_FRAME_SIZE</i> (R/W) (0X8806006C) : .....	293
36.2.15	<i>P_CD_FRAME_CNT</i> (R/W)(0x88060070) : .....	293
36.3	INTERRUPT .....	293
<b>37</b>	<b>CMOS SENSOR INTERFACE – CSI APPLICATION .....</b>	<b>294</b>
37.1	FRAME BUFFER CONTROL .....	294
37.2	CONTROL REGISTERS .....	294
37.3	TIME GENERATOR .....	294
37.3.1	<i>P_TG_CR</i> (0x08000000): <i>Time Generator Control</i> .....	295
37.3.2	<i>P_TG_LSTART</i> (0x08000004): .....	297
37.3.3	<i>P_TG_START</i> (0x08000008): .....	297
37.3.4	<i>P_TG_END</i> (0x0800000C): .....	298
37.3.5	<i>P_TG_BLACK</i> (0x0800000C): .....	298
37.3.6	<i>P_TG_BSUPPER</i> (0x08000014): .....	299
37.3.7	<i>P_TG_BSLOWER</i> (0x08000018): .....	299
37.3.8	<i>P_TG_TRANSP</i> (0x0800001C): .....	300
37.3.9	<i>P_TG_FBSADDR1</i> (0x08000020): <i>Frame Buffer 1's start address</i> .....	300
37.3.10	<i>P_TG_FBSADDR2</i> (0x08000024): <i>Frame Buffer 2's start address</i> .....	300
37.3.11	<i>P_TG_FBSADDR3</i> (0x08000028): <i>Frame Buffer 3's start address</i> .....	300
37.3.12	<i>P_TG_CAP</i> (0x0800002C): <i>Frame Buffer 3's start address</i> .....	301
37.3.13	<i>P_TG_CUTSETUP</i> (0x08000060): .....	301
37.4	MOTION DETECT CONTROL .....	302
37.4.1	<i>P_MD_CR</i> (0x88000030): <i>Motion Detect control register</i> .....	302
37.4.2	<i>P_MD_SADDR</i> (0x88000034): .....	303
37.4.3	<i>P_MD_POS</i> (0x88000038): .....	303
37.4.4	<i>P_MD_SADDR1</i> (0x8800003C): .....	303
37.4.5	<i>P_MD_CTABLE0</i> (0x88000040): .....	304

37.4.6	<i>P_MD_CTABLE1(0x88000044):</i>	304
37.4.7	<i>P_MD_CTABLE2(0x88000048):</i>	304
37.4.8	<i>P_MD_CTABLE3(0x8800004C):</i>	304
37.4.9	<i>P_MD_REG1(0x88000050):</i>	304
37.4.10	<i>P_MD_REG2(0x88000054):</i>	304
37.4.11	<i>P_MD_REG3(0x88000058):</i>	304
37.4.12	<i>P_MD_TH(0x8800005C):</i>	305
37.4.13	<i>P_MD_RGB(0x88000074):</i>	305
37.5	INTERRUPT	305
37.5.1	<i>P_CSI_IRQEN(0x88000078):</i>	305
37.5.2	<i>P_CSI_IRQSTS(0x8800007C):</i>	306
37.6	COLOR MODE	307
37.6.1	<i>P_CSI_Y2R_A1(0x880000E8):</i>	307
37.6.2	<i>P_CSI_Y2R_A2(0x880000EC):</i>	307
37.6.3	<i>P_CSI_Y2R_A3(0x880000F0):</i>	307
37.6.4	<i>P_CSI_R2Y_A1(0x880000E8):</i>	308
37.6.5	<i>P_CSI_R2Y_A2(0x880000EC):</i>	308
37.6.6	<i>P_CSI_R2Y_A3(0x880000F0):</i>	308
37.7	MOTION DETECTION	309
37.7.1	<i>Recognition Sample Type</i>	309
37.7.2	<i>Recognition Reference Filed / Frame</i>	310
<b>38</b>	<b>MPEG-4/JPEG CODEC</b>	<b>311</b>
38.1	FUNCTIONAL DESCRIPTION	311
38.1.1	<i>MPEG Encoding</i>	311
38.1.2	<i>MPEG Decoding</i>	314
38.1.3	<i>JPEG Encoding</i>	316
38.1.4	<i>JPEG Decoding</i>	318
38.1.5	<i>Coding Note</i>	320
38.2	MPEG ENGINE CONTROL REGISTER DEFINITION	320
38.2.1	<i>P_MP4_MJWIDTHL(0x88220000): Image Width Low Byte Control Registers</i>	320
38.2.2	<i>P_MP4_MJWIDTHH(0x88220004): Image Width High Byte Control Registers</i>	320
38.2.3	<i>P_MP4_MJHEIGHTL(0x88220008): Image Height Low byte Control Registers</i>	320
38.2.4	<i>P_MP4_MJHEIGHTH(0x8822000C): Image Height High byte Control Registers</i>	320
38.2.5	<i>P_MP4_MJHOFFSETL(0x88220010): Image Horizontal Offset Low Byte Control Registers</i>	321
38.2.6	<i>P_MP4_MJHOFFSETH(0x88220014): Image Horizontal Offset Control Registers</i>	321
38.2.7	<i>P_MP4_MJVOFFSETL(0x88220018): Image Vertical Offset Low Byte Control Registers</i>	

321	
38.2.8	<i>P_MP4_MJVOFFSET(0x8822001C): Image Vertical Offset Control Registers. .... 321</i>
38.2.9	<i>P_MP4_VLCOFFADDRL(0x88220020): VLC Bit Stream Starting Offset Address Low Byte</i>
321	
38.2.10	<i>P_MP4_VLCOFFADDRM(0x88220024):VLC Bit Stream Starting Offset Address Middle Byte 321</i>
38.2.11	<i>P_MP4_VLCOFFADDR(0x88220028):VLC Bit Stream Starting Offset Address High Byte</i>
321	
38.2.12	<i>P_MP4_TMBOFFADDRL(0x88220040): Thumb Nail Starting Address Low Byte..... 321</i>
38.2.13	<i>P_MP4_TMBOFFADDRM(0x88220044): Thumb Nail Starting Address Middle Byte 322</i>
38.2.14	<i>P_MP4_TMBOFFADDR(0x88220048): Thumb Nail Starting Address High Byte</i>
	<i>Registers. 322</i>
38.2.15	<i>P_MP4_YUVSEL(0x8822004C): YUV Group Selection Registers. .... 322</i>
38.2.16	<i>P_MP4_CONTROL(0x88220050): MJPG Control Registers. .... 322</i>
38.2.17	<i>P_MP4_SETTING(0x88220054): MJPG Setting Registers..... 323</i>
38.2.18	<i>P_MP4_HSFACOR(0x88220058): Horizontal Scale Factor Registers. .... 324</i>
38.2.19	<i>P_MP4_VSFACOR(0x8822005C): Vertical Scale Factor Registers..... 324</i>
38.2.20	<i>P_MP4_GOPVAL(0x88220060): P Frame number for MPEG Compress mode Select 324</i>
38.2.21	<i>P_MP4_SETEN(0x88220064): Setting Enable..... 324</i>
38.2.22	<i>P_MP4_VARDTHRL(0x88220068): Variance Threshold LSB Byte ..... 325</i>
38.2.23	<i>P_MP4_VARDTHRM(0x8822006C): Variance Threshold MSB Byte..... 325</i>
38.2.24	<i>P_MP4_INIAVGACT (0x88220070): Initial Average Activity Value ..... 325</i>
38.2.25	<i>P_MP4_AVGACTWEIL(0x88220078): Normalize Average Activity Value Weighting LSB Byte 326</i>
38.2.26	<i>P_MP4_AVGACTWEIM(0x8822007C):Normalize Average Activity Value Weight MSB Byte 326</i>
38.2.27	<i>P_MP4_RESET(0x88220080):Software Reset For MJPG Register..... 326</i>
38.2.28	<i>P_MP4_BUFCTRL(0x88220084): MJPG Buffer Controll Register. .... 326</i>
38.2.29	<i>P_MP4_IFRAME(0x88220088): Frame Mode Register. .... 327</i>
38.2.30	<i>P_MP4_TBL(0x8822008C): Thumb Nail Burst Length Register. .... 327</i>
38.2.31	<i>P_MP4_MESRAM(0x88220090): SRAM Setting Register..... 327</i>
38.2.32	<i>P_MP4_PROBEMODE(0x88220094): H/W Probe Mode Register. .... 327</i>
38.2.33	<i>P_MP4_MEMCTRL (0x88220098): Memory Control Register. .... 327</i>
38.2.34	<i>P_MP4_SOFCTRL(0x882200A0): Start of Compressing Control Register.. .... 328</i>
38.2.35	<i>P_MP4_GOPCOUNTER(0x882200A4): GOP Counter Monitor..... 328</i>
38.2.36	<i>P_MP4_AVGACT(0x882200A8): Normalize Average Activity Value for Every Frame 329</i>
38.2.37	<i>P_MP4_AVGVARD(0x882200AC): Average Difference Varance..... 329</i>

38.2.38	<i>P_MP4_MEBISTFAIL(0x882200B0): ME Bist Test Fail</i>	329
38.2.39	<i>P_MP4_MCBISTFAIL(0x882200B4): MC Bist Test Fail</i>	329
38.2.40	<i>P_MP4_MEMCBISTFINISH(0x882200B8): ME/MC Bist Test Finish</i>	329
38.2.41	<i>P_MP4_HSFINI(0x882200C0): Horizontal Initial Scaling Factor</i>	329
38.2.42	<i>P_MP4_VSFINI(0x882200C0): Vertical Initial Scaling Factor</i>	329
38.2.43	<i>P_MP4_UVMODE(0x882200C0): MC Write UV Transfer Mode</i>	329
38.2.44	<i>P_MP4_SFL(0x882200CC): MC/ME Scaling Factor Burst Length</i>	330
38.2.45	<i>P_MP4_INTMASK(0x882200D0): Interrupt Mask Register</i>	330
38.2.46	<i>P_MP4_INTCLR(0x882200D4): Interrupt Clear Register</i>	330
38.2.47	<i>P_MP4_DISGATED(0x882200D8): Disable for Gated Clock Register</i>	331
38.2.48	<i>MPEG4/JPEG VLC Buffer Size</i>	331
38.2.49	<i>P_MP4_DECWIDTHL(0x88220100): Decode Image Width Low Byte Control Registers</i>	331
38.2.50	<i>P_MP4_DECWIDTH(0x88220104): Decode Image Width High Byte Control Registers</i>	331
38.2.51	<i>P_MP4_MJHEIGHTL(0x88220108): Decode Image Height Low byte Control Registers</i>	331
38.2.52	<i>P_MP4_DECHEIGHT(0x8822010C): Decode Image Height High byte Control Registers</i>	332
38.2.53	<i>Decode VLC Bit Stream Starting Offset</i>	332
38.2.54	<i>Last Memory Address</i>	332
38.2.55	<i>P_MP4_LASTLENGTH(0x88220128): Last Memory Length</i>	332
38.3	<b>TEXTURE ENGINE CONTROL REGISTER DEFINITION</b>	333
38.3.1	<i>Quantization Table 1</i>	333
38.3.2	<i>Quantization Table 2</i>	333
38.3.3	<i>P_MP4_NOQTBL(0x88220600): Q-Table Currently Used Register</i>	333
38.3.4	<i>P_MP4_QTBLSET(0x88220604): Q-Table Setting Register</i>	333
38.3.5	<i>P_MP4_QSRAMEN(0x88220608): Q-Table SRAM Control Register</i>	334
38.3.6	<i>P_MP4_ENTHUMB (0x8822060C): Thumb Nail Image Generation Control Register</i>	334
38.3.7	<i>P_MP4_JFIF (0x88220610): JFIF Compatible Control</i>	334
38.3.8	<i>P_MP4_TRUNTRES(0x88220614):</i>	334
38.3.9	<i>P_MP4_VLCBIT (0x88220618):</i>	335
38.3.10	<i>P_MP4_JFIFEND (0x8822061C):</i>	335
38.3.11	<i>Restart MCU</i>	335
38.3.12	<i>P_MP4_IFRAMEQSCALE(0x88220640): Quantizer Scale Value for I-Frame</i>	335
38.3.13	<i>P_MP4_PFRAMEQSCALE(0x88220644): Quantizer Scale Value for P-Frame</i>	335
38.3.14	<i>P_MP4_MATCHCNT (0x88220648): Match Code Function Control</i>	335

38.3.15	<i>Match Code</i> .....	336
38.3.16	<i>P_MP4_OFFSET(0x8822065C): Offset Value</i> .....	336
38.3.17	<i>P_MP4_VOPTIMEINC (0x88220660): VOP Time Increment Mode Selection</i> .....	336
38.3.18	<i>P_MP4_MSCNT (0x88220664): Mini Second Counter</i> .....	336
38.3.19	<i>P_MP4_MSPLUSCNTEN (0x88220668): Enable one extra mini second added counter</i> 336	
38.3.20	<i>P_MP4_MSPLUSCTRL(0x8822066C): Extra mini second added Control</i> .....	336
38.3.21	<i>VOP Time Increment Resolution</i> .....	337
38.3.22	<i>VOP Time Increment Unit</i> .....	337
38.3.23	<i>P_MP4_VOPTimeIncLen(0x88220690): Length of VOP time Increment</i> .....	337
38.3.24	<i>TM5 Model: Reaction Parameter</i> .....	337
38.3.25	<i>TM5 Model: Virtual Buffer Fullness Initial Values for I picture</i> .....	337
38.3.26	<i>TM5 Model: The Predict byte Counter Each Macro In I Frame</i> .....	337
38.3.27	<i>TM5 Model: The Predict byte Counter for Each Macro In P Frame</i> .....	337
38.3.28	<i>TM5 Model: Q-Scale Bound</i> .....	338
38.3.29	<i>TM5 Model: Virtual Buffer Fullness Initial Values for P picture</i> .....	338
38.3.30	<i>Compressed Data Size</i> .....	338
38.3.31	<i>TM5 Model: Sum of Every Macro Block Q-Scale of a Frame</i> .....	338
38.3.32	<i>P_MP4_SRAM_CS_N(0x882206F8):</i> .....	338
38.3.33	<i>P_MP4_AUTO_RESET(0x882206FC):</i> .....	338
38.3.34	<i>P_MP4_SRAM_TEST_CTRL (0x88220780):</i> .....	339
38.3.35	<i>P_MP4_JPG_SEL (0x88220784): JPEG Probe Bus Selection</i> .....	339
38.3.36	<i>P_MP4_BistMode (0x88220788): SRAM Bist Mode Control</i> .....	339
38.3.37	<i>P_MP4_BistFail (0x8822078C): SRAM Bist Mode Fail Information Control</i> .....	339
38.3.38	<i>P_MP4_BlockEnd (0x88220790):</i> .....	339
38.3.39	<i>P_MP4_DeHuffmanen (0x88220798): Dehuffman mode Control</i> .....	339
38.3.40	<i>P_MP4_DeHuffmanrdy (0x8822079C):</i> .....	340
38.3.41	<i>Firmware Dehuffman</i> .....	340
38.3.42	<i>P_MP4_VIDEORstMode (0x882207A8): Video Reset</i> .....	340
38.3.43	<i>P_MP4_NACTEN (0x882207AC): Enable activity value from ME</i> .....	340
38.3.44	<i>Number of IntraMB in current Frame</i> .....	340
38.3.45	<i>Number of InterMB in current Frame</i> .....	340
38.3.46	<i>Number of Skipped MB in current Frame</i> .....	341
38.3.47	<i>P_MP4_H263_CTRL(0x882207C8): H263 format Control Registers.</i> .....	341
38.3.48	<i>P_MP4_H263_Struct(0x882207CC):</i> .....	341
38.3.49	<i>Sum of Absoulte Horizontal Motion Vectors(HalfPel Unit)</i> .....	341
38.3.50	<i>Sum of Absoulte Vertical Motion Vectors(HalfPel Unit)</i> .....	342

38.3.51	<i>P_MP4_TSRAM_CTRL(0x882207E0):</i> <i>TSRAM Control Register</i> .....	342
38.3.52	<i>P_MP4_GRegion_CTRL(0x882207E4):</i> .....	342
38.3.53	<i>P_MP4_MJPG_CTRL (0x882207E8):</i> .....	342
38.3.54	<i>P_MP4_ROIMBXOffsetLSB(0x882207EC):</i> <i>LSB of Horizontal Starting MB Coordinate of ROI</i>	343
38.3.55	<i>P_MP4_ROIMBXDestLSB(0x882207F0):</i> <i>LSB of Horizontal End MB Coordinate of ROI</i>	343
38.3.56	<i>P_MP4_ROIMBYOffsetLSB(0x882207F4):</i> <i>LSB of Vertical Starting MB Coordinate of ROI</i>	343
38.3.57	<i>P_MP4_ROIMBYDestLSB(0x882207F8):</i> <i>LSB of Vertical End MB Coordinate of ROI</i>	343
38.3.58	<i>P_MP4_ROIMBMSB(0x882207FC):</i> <i>MSB Coordinate of ROI</i> .....	343
38.3.59	<i>Huffman Table</i> .....	343
38.3.60	<i>Luminance DC code word</i> .....	343
38.3.61	<i>Luminance address offset for DC first codeword</i> .....	343
38.3.62	<i>Luminance DC Huffman table content</i> .....	344
38.3.63	<i>Chrominance DC codeword</i> .....	344
38.3.64	<i>Chrominance address offset for DC codeword</i> .....	344
38.3.65	<i>Chrominance DC Huffman table content</i> .....	344
38.3.66	<i>Luminance AC Huffman table codeword</i> .....	344
38.3.67	<i>Luminance AC Huffman table address offset for AC first 1-bit codeword</i> .....	344
38.3.68	<i>Chrominance AC Huffman table codeword</i> .....	344
38.3.69	<i>Chrominance AC Huffman table address offset for codeword</i> .....	345
38.4	<b>MPEG/JPEG ENGINE LIMITATION</b> .....	345
38.4.1	<i>JPEG Engine Limitation</i> .....	345
38.4.2	<i>MPEG Engine Limitation</i> .....	346

---

---

## 2 REVISION HISTORY

---

---

Revision	Date	By	Remark
1.0	12/15/2005	Jackie Chang	Add CD-Servo/MPEG4/DMA function/SFTCFG/BUFCFG.....

### 3 PROCESSOR BRIEFING

The SPG290A is equipped with *S<sup>+</sup>core® 7*, the latest 32-bit CPU developed by SUNPLUS, pronounced as *score-seven*. The *S<sup>+</sup>core® 7* is a 32-bit RISC with Sunplus-owned instruction set architecture (ISA). The ISA has 32/16-bit hybrid instruction mode and parallel conditional execution for high code density, high performance and versatile application. The micro-architecture includes AMBA bus for SoC integration, coprocessor and custom engine interface for function flexibility, and SJTAG for efficient debugging and In-Circuit Emulation (ICE).

The *S<sup>+</sup>core® 7* is a single issue, 7-pipeline stage, high performance and high speed 32-bit RISC. For SPG290A, *S<sup>+</sup>core® 7* can run up to 162 MHz. It supports 4-KB two-way set associative I/D-cache and 4KB LIM/LDM (local instruction/data memory). The MMU (memory management unit) is also supported for RTOS. We also define two custom engine and three coprocessor interfaces for user defined function extension. The custom engine supports 32-bit sign/un-sign multiply and divider. The bus interface of the processor is compliance to the AHB v2.0 for easy integration into SoC implementation.

Up to sixty-three prioritized vector interrupts are supported for fast interrupt service. The high performance Sunplus-patented unaligned load/store and pre/post increment addressing instructions are provided for string copy or memory moving. The powerful bit operation instructions and branch instructions using the counter register as loop iteration counter are provided for control application, and sleep instruction is provided for low-power application

#### 3.1 Endianess

All data access in SPG290 32 bits system are **Little Endian**. The most significant byte in a WORD, 32 bits data, is at the last place of a 4 bytes string.

For data read operations, a simple example is show below for word(32 bits), half-word(16-bits), and byte(8-bits) access:

<b>Memory Address</b>	0x00	0x01	0x02	0x03	<b>Read word</b> 0x00	0x78563412			
<b>Data Content</b>	0x12	0x34	0x56	0x78	<b>Read half word</b> 0x00m 0x03	0x3412	0x7856		
					<b>Read byte</b> 0x00~0x03	0x12	0x34	0x56	0x78

#### 3.2 Key Features

The features of *S<sup>+</sup>Core* are listed as following:

- 32/16-bits Hybrid Execution Mode

- Parallel Conditional Execution(patent pending)
- Capability for Further Software Security Design(Patent pending)
- A Harvard Architecture(I-Cache/ D-Cache) solution
- MMU (Fixed mode and Full-function mode)
- Compliance to the AMBA Specification (Rev 2.0) for easy integration into SOC implementation
- Vectored Interrupt
- SJTAG (Sunplus JTAG)

---

## 4 DEBUGGING IN SPG290A

---

### 4.1 Overview

The debug solution in processor includes the following features:

- Off-chip debug memory access  
SJTAG allows processor in the debug mode access the instructions or data that are not physically on the chip. Memory access to the special segment will be transferred to JTAG port and then sent to the debug probe that controlled by debug host PC. Debug probe will handle the memory access by redirect the access to its local memory. Then, the accessed data will feedback to processor via JTAG again.
- Hardware breakpoint  
Two types of hardware breakpoint are included which can be configured to cause debug exception on:
  - 1) Instruction fetch of a specified address(Breakpoint)
  - 2) Data fetch of a specified address and the access data value(Watchpoint)
- Software breakpoint instruction  
Two additional instructions are added to achieve system debug: Software Debug Breakpoint(SDBBP) and Debug Exception Return(DRET).
- Single step execution  
A dedicated single step exception in processor is provided for single step debugging.
- Debug interrupt  
A debug interrupt is provided to force processor to enter debug mode at any time.
- DMA Access  
A DMA channel that controlled by SJTAG directly access system bus through BIU. This feature is very useful when user need to download code to system memory.

### 4.2 SJTAG Module

The feature of SJTAG module includes:

- The hardware point unit will assert the breakpoint exception request to core.
- A memory controller handles the memory request from core if the address is located in debug segment.  
If the access is in debug register segment, it will directly access the debug registers such as breakpoint unit registers or debug control register. If the access is in debug memory segment, it will pass the access to debug probe through JTAG interface.
- A JTAG controller handles the communication between probe and SJTAG module.

#### **4.3 RS232**

The SPG290A still provides a RS232 for debugging purposes. The user should be acquainted with the I/O function printf(). In addition, during the development phase, a simple command monitor is implemented to allow user to issue commands via the RS232. For example, the user can issue a dump command to dump memory content.

#### **4.4 Reference**

- **S<sup>+</sup>Core 7 Processor Core Technical Reference Manual( for Software Use) V2.1 – Feb. 01, 2005, by Sunplus Technology Co., Ltd.**
- **S<sup>+</sup>Core IDE User Manual, by Sunplus Technology Co., Ltd.**

---

---

## 5 S<sup>+</sup>CORE IDE

---

---

S<sup>+</sup>Core IDE, a 32-bit powerful Integrated Development Environment(IDE) for developing application in C or assembler for S<sup>+</sup>Core series CPU, which owns fully integrated Windows development tool.

### 5.1 Installation S<sup>+</sup>Core IDE

The S<sup>+</sup>Core IDE can be run on Windows98<sup>®</sup>, Windows2000<sup>®</sup> and WindowsXP<sup>®</sup>. Follow the on-screen instruction and S<sup>+</sup>Core IDE will be installed on your computer. The following two points must be considered:

1. You must have administrator privilege when you install S<sup>+</sup>Core IDE on Win2000/XP.
2. The printer-port mode of "SPP" must be set in BIOS setting, and hardware address must be set to 378H as well.

### 5.2 External Devices for Simulator

In Simulator mode, the S<sup>+</sup>Core IDE provides two external devices for easy debugging your program.

- LCD

LCD displaying can be show during running after LCD is enabled from [Project]->[Setting]->[Simulator]-[Peripheral equipments].

NOTES:

- i. LCD displaying can be used in simulator mode only.
- ii. Default LCD is off

- UART

UART printf can be enabled during running after UART is enabled from [Project] [Setting] [Simulator] [Peripheral equipments]. For more information about UART printf usage, please refer to example "UART printf" provided in installation kit.

NOTES:

1. UART printf can be used in simulator mode only.
2. Default UART is off

---

## 6 SCENEDIRECTOR - GRAPHIC TOOL

---

SceneDirector™, a powerful tool authored by Sunplus Tech Co., Ltd mainly for developing games, offers effective ways to import and edit pictures, to setup and view an animation (especially, to view an animation on an ICE), to export animation data to PC, to manage cells (small picture), and to edit palettes. With the user-friendly interface and flexible editing modes, the SceneDirector is capable of helping you to work easily and efficiently.

### 6.1 Features

- Single-project architecture for organizing Textures, Bitmaps, JPEGs, Layers, and Sprites(frames)
- Supports the following formats: \*.bmp \*.jpg \*.pcx \*.gif
- Provides two modes for importing pictures: sliced(into cells), unsliced.
- Two cell libraries: Scene Library and System Library
- Two editable, exportable, independent palette bars, up to 512 colors.
- RGB555/RGB565 mode for Bitmaps
- Powerful edit function on textures, bitmaps, layers, sprites, and frames.
- Flexible animation setup and preview function with one background(Layers item) and several sprites.
- Up to 3 layers for a background.
- Various Download function: download image(texture, frame, bitmap, JPEG)/ animation, download to PC/ICE
- Two export modes: ASM, BIN.
- Two display modes for preview: CIF, VGA.

### 6.2 Installing SceneDirector

#### 6.2.1 System Requirement

The current version of the tools is capable of running under Windows98® and Windows2000®. And it is recommended to run under Windows2000®. The propositional minimum system requirements are:

- CPU clock: PII or later
- Capacity of memory: 256 MB
- Free hard disk space: 500 MB

#### 6.2.2 Installing

Taking the following steps and you can install this tool on your computer:

1. Execute the installation file (.exe).
2. Follow on-screen prompts.

During installation, you will be asked to select one type of installation: **Typical**, **Compact** or **Custom**. SUNPLUS highly recommends you to install the tool with the "Typical" type which should apply for most of users. Selecting "Compact" results in the tool of minimal configuration installed on your computer. The "Custom" option is usually for advanced users.

---

---

## 7 SUNMIDIAR - SOUND TOOL

---

---

SunMidiar, a powerful tool, authored by Sunplus Tech Co., Ltd mainly for making & editing Tone Colors and playing them on an ICE, offers a more effective way than ever to manage MIDI files and Instrument Library, to make and edit tone colors, and to play MIDI and tone colors on PC/ICE. With the user-friendly interface and flexible editing modes, the SunMidiar is capable of helping you to work easily and efficiently, and allows you to create and refine your application in only one place.

### 7.1 Installing SunMidiar

The current version of the tool is capable of running under Windows98® and Windows2000®. The propositional minimum system requirements are:

- CPU clock: 133MHz
- Capacity of memory: 64MB
- Free hard disk space: 20MB

#### 7.1.1 Installing

Take the following steps, you can install the SunMidiar on your computer:

1. Extract the zip file and execute the installation file (setup. exe).
2. Follow the on-screen prompts and the SunMidiar will be installed on your computer.

During installation, you will be asked to select one installation type: Typical, Compact, or Custom. SUNPLUS highly recommends you to install the tool with the "Typical" which applies for most users. Selecting "Compact" results in the minimal configuration installed on your computer. The "Custom" option is usually for advanced users.

### 7.2 Technical Support

SunMidiar is designed to use easily. However, on some occasion you may need some explanation or help with an issue. In most cases, you'll find the answer to your questions through on-line help inserted this tool. In addition, you'll get available support material around-the-clock at our Web site(<http://www.sunplus.com>).

For the furthermore technical support, you can contact with your Sunplus sales representatives.

---

## 8 SYSTEM OVERVIEW

---

### 8.1 General Description

SPG290A, an SoC designed specifically for TV game and handheld game products, is composed of **S<sup>+</sup>core®** (a 32-bit CPU developed by SUNPLUS Technology), **2D Picture Processing Unit (PPU)**, **24 channel Sound Processing Unit (SPU)** and other primary functions for video game and ELA applications. SPG290A is able to generate graphics and sound for the television system (NTSC or PAL) and LCD. SPG290A features the integrated **CD servo** circuit for reading CD contents. With the ability to access CD titles, the huge amount of data for generating gorgeous video images and pleasing sound can be stored on the disc. **SD card** and **NAND-type flash** are also supported for mass data storage. The operating voltage supply range is 3.0V through 3.6V and CPU speed is 27~162 MHz. Plus, it offers 76 configurable general purpose I/O, **six 16-bit timers**, **32768Hz Real Time Clock**, **Low Voltage Detect**, **Low Voltage Reset**, **12-bit ADC**, **UART interface**, **SPI interface**, **SIO interface**, **I<sup>2</sup>C master interface** and many other features that facilitate connecting with varieties of I/O devices such as **TFT LCD**, **color STN LCD**, **CMOS image sensor**, **TV decoder**, **Light Pen**, **Touch panel**...etc. The SPG290A provides not only the latest video game technology, but also the full service and support of SUNPLUS.

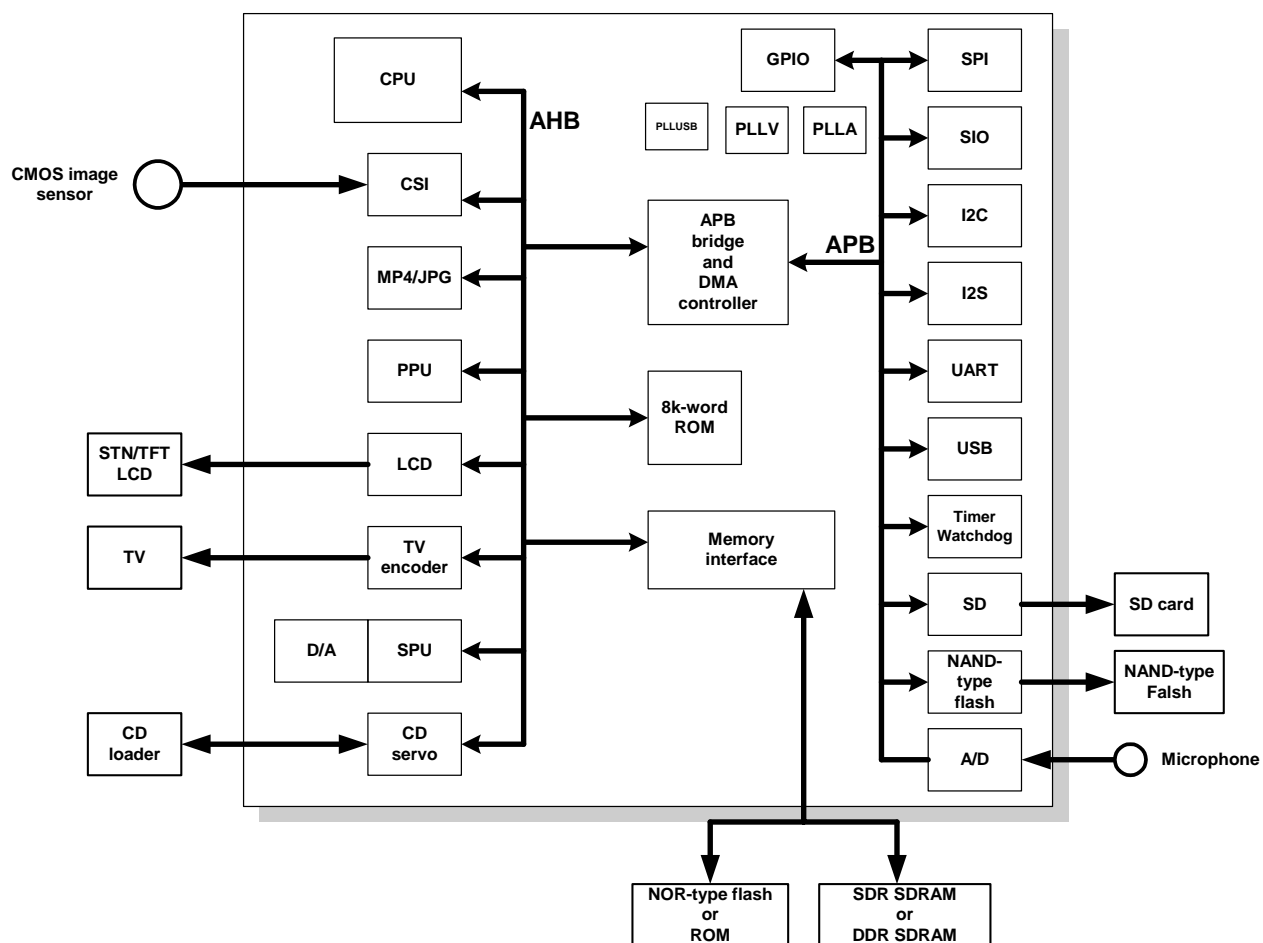
### 8.2 Features

- Operating Voltage: I/O VDD is 3.0V ~ 3.6V, core VDD is 1.62V~1.98V
- CPU speed: 27 ~ 162 MHz
- Supports SDR DRAM and DDR DRAM with capacity up to 16 Mbytes
- Supports 32-bit and 16-bit SDRAM data buses
- External ROM capacity up to 8 Mbytes
- Integrated CD servo: supports CD-DA, CD-ROM, and CD-ROM/XA
- Supports interlace/non-interlace NTSC/PAL composite Graphic resolution: VGA(640 pixels x 480 pixels), HVGA(640 pixels x 240 pixels) and QVGA(320 pixels x 240 pixels) video output
- Programmable 4/16/64/256/512/1024 index color mode (512 colors for texts, 512 colors for sprites)
- Provides 32768/65536 direct color mode
- Three layer texts with size 1024 pixels x 512 pixels
- Provides character and bitmap operation modes
- Maximum 512 pieces of sprite are available
- Supports different size of character for sprite and text
- Hardware Horizontal / Vertical Flip function
- Texts and sprites provide 64 levels blending control
- Provides fade-in and fade-out effect with 256 levels.
- Hardware MPEG-4/JPEG codec
- MPEG-4 frame rate: up to 30 frame/s at QVGA resolution
- Provides 4-channel APB DMA: APB peripherals to DRAM or DRAM to APB peripherals

- Provides DRAM DMA: DRAM-to-DRAM data transfer by hardware
- Supports vertical blanking, any position, DMA end IRQ sources
- Provide 24 PCM/ADPCM sound channels
- Automatically volume control function (Compressor)
- Built-in 4-band digital equalizer
- Two 16-bit high speed DACs for stereo sound quality
- Fully ADSR envelope control
- Three programmable built-in PLLs to provide system clocks
- 27MHz crystal for NTSC/PAL system
- Real Time Clock (RTC): 32768Hz
- Six 16-bit timers/counters (Programmable and Auto Reload)
- 16 interrupt sources: SPU, PPU, Timer, Time-base, External Input, Key wakeup
- Key wake-up function
- 8+1 channels 12-bit AD converter with 9-bit accuracy
- USB 1.1 host or USB 1.1 device
- UART: Universal Asynchronous Receiver and Transmitter
- SPI: Serial Peripheral Interface(master and slave)
- SIO: Sunplus Serial Input/Output Interface
- Built-in Watchdog function
- Two Light Pen interface
- TFT LCD interface
- CSTN LCD interface
- Supports Sunplus CMOS image sensor interface to connect with Sunplus CMOS image sensor device
- Support CCIR-601/656 CMOS image sensor / TV decoder interface
- Supports SD card and NAND-type flash for data storage

### 8.3 Block Diagram

## SPG290



## 9 PLLS AND CKG

### 9.1 PLLS

There are three PLL sources embedded in SPG290, called PLLV, PLLA and PLLU respectively. PLLV generates system clock that is used by the whole system, PLLA generates clock source to CD Servo and I2S module and PLLU generates clock used specifically by USB Module. The output frequency of PLLV can be adjusted through input pins. It can generate a clock with frequency of multiple 27MHz, the maximum frequency in SPG290 is 162MHz. PLLA generates a 67.7376MHz clock for I2S, then divided by 2 in CKG module to generate 33.8688MHz clock to feed CD Servo Module. PLLU generates a 48MHz clock to USB Module.

### 9.2 CKG

There is no clock generation circuit inside each module. All clock generation and distribution circuit are inside CKG. The maximum frequency of CPU clock is 162MHz, and the maximum frequency of AHB clock is 108MHz. The clock frequencies of each module are as follows, that  $N=1,2,3,4,5$ :

Module	Main clock	AHB master clock	AHB slave clock	APB clock
CPU	PLLV / N	CPU main clock / M		
MIU to CPU			CPU main clock / M	
MIU to other modules	PLLV / L		PLLV / L	
JPG	PLLV / A	PLLV / L	CPU main clock / M	
CD	33.8688MHz	PLLV / L	CPU main clock / M	
CSI	27 MHz	PLLV / L	CPU main clock / M	
PPU	PLLV / B	PLLV / L	CPU main clock / M	
SPU	27 MHz	PLLV / L	CPU main clock / M	
TV	27 MHz	PLLV / L	CPU main clock / M	
LCD	27 MHz	PLLV / L	CPU main clock / M	
UART	27~67.5 MHz			27MHz
TIMER	32768 Hz			27MHz
WDOG	32768 Hz			27MHz

The table above indicates:

- 1). All AHB slave (connect to CPU) and AHB master (connects to MIU) of each modules except CPU are running at the same clock frequency with maximum frequency up to 108MHz.
- 2). AHB master clock frequency of MJPG module is multiple of core clock frequency. Suppose the AHB master clock frequency is higher than Core clock and the maximum frequency of Core is 54MHz.
- 3). AHB master clock of PPU is asynchronous with core clock, therefore, it need asynchronous circuit implementation. Suppose AHB master's clock frequency is no lower than core clock and the maximum frequency of Core is 108MHz.
- 4). LDMDMA's AHB master is using MIU\_CLK, and the LDMDMA's LDM Interface is using CPU\_CLK. These two have multiple relations, and also have asynchronous circuit implementation. Suppose the CPU\_CLK's clock frequency is no lower than MIU\_CLK.
- 5). Asynchronous circuit implementation is a must between AHB master of CSI, TV, LCD and SPU. We can suppose that AHB master's clock is no lower than main clock, but have no multiple relations. As for connection between AHB slave and core, as AHB slave only set control register's value, only the enable register class need to take care of synchronicity.

### 9.3 Control registers

#### 9.3.1 P\_CPU\_CONF(0x88210000): CPU Clock Config

NAME	D6	D5	D4	D3	D2	D1	D0
P_CPU_CONF	CPUALLRST	CPUWRST	CPUPRST	CPULLIRQ	CPUCKMIU	CPUCKIRQ	CPUCK

CPUCK:	Temporarily Stop CPUCK for Changing Frequency						
0:	Enable(Stop)						
1:	Disable						
CPUCKIRQ:	Permanently Stop CPUCK until IRQ or Key Pressing Happens Setting						
0:	Enable(Stop)						
1:	Disable						
CPUCKMIU:	Permanently Stop CPUCK and MIU Clock until IRQ or Key Pressing Happen Setting						
0:	Enable(Stop)						
1:	Disable						
CPULLIRQ:	Permanently Stop CPU and MIU Clock(stops at 0) and then Disable PLL until IRQ or Key Pressing Happen						
0:	Disable						
1:	Enable(Stop)						
CPUPRST:	Temporarily Reset CPU(to Poweron_n port)						
0:	Disable						
1:	Enable(Reset)						
CPUWRST:	Temporarily Reset CPU(to warmrst_n port)						
0:	Disable						
1:	Enable(Reset)						
CPUALLRST:	Temporarily Reset CPU (to warmrst_n port) and All Other Modules						
0:	Disable						
1:	Enable(Reset)						

#### 9.3.2 P\_CKG\_SEL\_CPU(0x88210004): CPU Clock Frequency Select

NAME	D3-D0
P_CKG_SEL_CPU	CPU_CLOCK

CPU_CLOCK:	select CPU clock frequency		
0000:	video PLL output divided by 1		
0001:	video PLL output divided by 2		
0010:	video PLL output divided by 3		
0011:	video PLL output divided by 4		
0100:	video PLL output divided by 6		

- 0101: video PLL output divided by 8
- 0110: audio PLL output divided by 1
- 0111: audio PLL output divided by 2
- 1000: audio PLL output divided by 3
- 1001: audio PLL output divided by 4
- 1010: audio PLL output divided by 6
- 1011: audio PLL output divided by 8

### 9.3.3 P\_CPUAHB\_CONF(0x88210008): CPU AHB Clock Config

NAME	D0
P_CPUAHB_CONF	CPUAHB_CLK_Stop

CPUAHB\_CLK\_Stop: Temporarily Stop CPUAHB\_CLK for Changing Frequency

- 0: Enable(Stop)
- 1: Disable

### 9.3.4 P\_CKG\_SEL\_CPUAHB(0x8821000C): AHB Bus Frequency Select

NAME	D1-D0
P_CKG_SEL_CPUAHB	CPUAHB_CLK

CPUAHB\_CLK: Select AHB Bus Frequency

- 00: CPU clock divided by 1
- 01: CPU clock divided by 2
- 10: CPU clock divided by 3
- 11: CPU clock divided by 4

### 9.3.5 P\_MIU\_CONF(0x88210010): MIU Clock Config

NAME	D2	D1	D0
P_MIU_CONF	MIU_RST	MIU_DRAM	MIU_STOP

MIU\_STOP: Stop the MIU Clock but Don't Force SDRAM to Enter the Self-Refresh Mode

- 0: Enable(Stop)
- 1: Disable

MIU\_DRAM: Force DRAM to Enter the Self-Refresh Mode and Stop MIU Clock Until Wakeup Happens

- 0: Enable(Stop)
- 1: Disable

MIU\_RST: Reset MIU Setting

- 0: Enable(Reset)
- 1: Disable

### 9.3.6 P\_CSI\_CONF(0x88210018): CSI Clock Config

NAME	D1	D0
P_CSI_CONF	CSI_RST	CSI_STOP

CSI\_STOP: Stop CSI Clock

0: Enable(Stop)

1: Disable

CSI\_RST: Reset CSI Module

0: Enable(Reset)

1: Disable

### 9.3.7 P\_CKG\_SEL\_CSI (0x8821001C): CSI Clock Select

NAME	D3	D2	D1-D0
P_CKG_SEL_CSI	CLOCK_FORM	CSI_SOURCE	CSI_CLOCK

CSI\_CLOCK: Select the CSI Clock

00: CSI source clock divided by 1

01: CSI source clock divided by 2

10: CSI source clock divided by 4

11: CSI source clock divided by 8

CSI\_SOURCE: Select the CSI Source Clock

0: 27MHz clock

1: 24MHz clock

CSI\_FORM: Select the Clock Form.

0: Select the inverted clock from CMOS sensor as the pixel clock

1: Select the clock from CMOS sensor as the pixel clock

### 9.3.8 P\_PPU\_CONF(0x88210020): PPU Clock Config

NAME	D1	D0
P_PPU_CONF	PPU_RST	PPU_STOP

PPU\_STOP: Stop PPU Clock

0: Enable(Stop)

1: Disable

PPU\_RST: Reset PPU Module

0: Enable(Reset)

1: Disable

### 9.3.9 P\_CKG\_SEL\_PPU(0x88210024): PPU Clock Frequency Select

NAME	D3-D0
P_CKG_SEL_PPU	PPU_CLOCK

PPU\_CLOCK: Select **PPU x 2** clock frequency

0000:	video PLL output divided by 1
0001:	video PLL output divided by 2
0010:	video PLL output divided by 3
0011:	video PLL output divided by 4
0100:	video PLL output divided by 6
0101:	video PLL output divided by 8
0110:	audio PLL output divided by 1
0111:	audio PLL output divided by 2
1000:	audio PLL output divided by 3
1001:	audio PLL output divided by 4
1010:	audio PLL output divided by 6
1011:	audio PLL output divided by 8

### 9.3.10 P\_JPG\_CONF(0x88210028): JPG Clock Config

NAME	D1	D0
P_JPG_CONF	JPG_RST	JPG_STOP

JPG\_STOP: Stop JPG Clock

0:	Enable(Stop)
1:	Disable

JPG\_RST: Reset JPG Module

0:	Enable(Reset)
1:	Disable

### 9.3.11 P\_CKG\_SEL\_JPG(0x8821002C): JPG Clock Frequency Select

NAME	D3-D0
P_CKG_SEL_JPG	JPG_CLOCK

JPG\_CLOCK: Select JPG clock frequency

0000:	video PLL output divided by 1
0001:	video PLL output divided by 2
0010:	video PLL output divided by 3
0011:	video PLL output divided by 4
0100:	video PLL output divided by 6
0101:	video PLL output divided by 8
0110:	audio PLL output divided by 1

0111: audio PLL output divided by 2  
 1000: audio PLL output divided by 3  
 1001: audio PLL output divided by 4  
 1010: audio PLL output divided by 6  
 1011: audio PLL output divided by 8

### 9.3.12 P\_TVE\_CONF(0x88210030): TVE Clock Config

NAME	D1	D0
P_TVE_CONF	TVE_RST	TVE_STOP

TVE\_STOP: Stop TVE Clock

0: Enable(Stop)

1: Disable

TVE\_RST: Reset TVE Module

0: Enable(Reset)

1: Disable

### 9.3.13 P\_LCD\_CONF(0x88210034): LCD Clock Config

NAME	D1	D0
P_LCD_CONF	LCD_RST	LCD_STOP

LCD\_STOP: Stop LCD Clock

0: Enable(Stop)

1: Disable

LCD\_RST: Reset LCD Module

0: Enable(Reset)

1: Disable

### 9.3.14 P\_CKG\_SEL\_LCD(0x88210038): LCD Clock Frequency Select

NAME	D5-D3	D2-D0
P_LCD_CONF	TFT_CLK	STN_CLK

STN\_CLK: Select CSTN clock frequency

000: video PLL output divided by 1

001: video PLL output divided by 2

010: video PLL output divided by 3

011: video PLL output divided by 4

100: video PLL output divided by 6

101: video PLL output divided by 8

TFT\_CLK: Select TFT clock frequency

000: 27MHz

001: USB PLL output divided by 2

010: USB PLL output divided by 3

011: USB PLL output divided by 4

100: USB PLL output divided by 6

101: USB PLL output divided by 8

### 9.3.15 P\_SPU\_CONF(0x8821003C): SPU Clock Config

NAME	D1	D0
P_SPU_CONF	SPU_RST	SPU_STOP

SPU\_STOP: Stop SPU Clock

0: Enable(Stop)

1: Disable

SPU\_RST: Reset SPU Module

0: Enable(Reset)

1: Disable

### 9.3.16 P\_CDS\_CONF(0x88210044): CD Servo Clock Config

NAME	D1	D0
P_CDS_CONF	CDS_RST	CDS_STOP

CDS\_STOP: Stop CD Servo Clock

0: Enable(Stop)

1: Disable

CDS\_RST: Reset CD Servo Module

0: Enable(Reset)

1: Disable

### 9.3.17 P\_BLNDMA\_CONF(0x88210050): BLNDMA Clock Config

NAME	D1	D0
P_BLNDMA_CONF	BLNDMA_RST	BLNDMA_STOP

BLNDMA\_STOP: Stop BLNDMA Clock

0: Enable(Stop)

1: Disable

BLNDMA\_RST: Reset BLNDMA Module

0: Enable(Reset)

1: Disable

### 9.3.18 P\_APBDMA\_CONF(0x88210058): APBDMA Clock Config

NAME	D1	D0
P_APBDMA_CONF	APBDMA_RST	APBDMA_STOP

APBDMA\_STOP: Stop APBDMA Clock

0: Enable(Stop)

1: Disable

APBDMA\_RST: Reset APBDMA Module

0: Enable(Reset)

1: Disable

### 9.3.19 P\_UART\_CONF(0x8821005C): UART Clock Config

NAME	D1	D0
P_UART_CONF	UART_RST	UART_STOP

UART\_STOP: Stop UART Clock

0: Enable(Stop)

1: Disable

UART\_RST: Reset UART Module

0: Enable(Reset)

1: Disable

### 9.3.20 P\_USB\_CONF(0x88210064): USB Clock Config

NAME	D2	D1	D0
P_USB_CONF	USB_RST	USBH_STOP	USBD_STOP

USBD\_STOP: Stop USB Device Clock

0: Enable(Stop)

1: Disable

USBH\_STOP: Stop USB Host Clock

0: Enable(Stop)

1: Disable

USB\_RST: Reset USB Module

0: Enable(Reset)

1: Disable

### 9.3.21 P\_TIMER1\_CONF(0x8821006C): Timer1 Clock Config

NAME	D1	D0
P_TIMER1_CONF	TIMER1_RST	TIMER1_STOP

TIMER1\_STOP: Stop TIMER1 Clock

0: Enable(Stop)

1: Disable

TIMER1\_RST: Reset TIMER1 Module

0: Enable(Reset)

1: Disable

### 9.3.22 P\_TIMER2\_CONF(0x88210070): Timer2 Clock Config

NAME	D1	D0
P_TIMER2_CONF	TIMER2_RST	TIMER2_STOP

TIMER2\_STOP: Stop TIMER2 Clock

0: Enable(Stop)

1: Disable

TIMER2\_RST: Reset TIMER2 Module

0: Enable(Reset)

1: Disable

### 9.3.23 P\_TIMER3\_CONF(0x88210074): Timer3 Clock Config

NAME	D1	D0
P_TIMER3_CONF	TIMER3_RST	TIMER3_STOP

TIMER3\_STOP: Stop TIMER3 Clock

0: Enable(Stop)

1: Disable

TIMER1\_RST: Reset TIMER3 Module

0: Enable(Reset)

1: Disable

### 9.3.24 P\_TIMER4\_CONF(0x88210078): Timer4 Clock Config

NAME	D1	D0
P_TIMER4_CONF	TIMER4_RST	TIMER4_STOP

TIMER4\_STOP: Stop TIMER4 Clock

0: Enable(Stop)

1: Disable

TIMER4\_RST: Reset TIMER4 Module

0: Enable(Reset)

1: Disable

### 9.3.25 P\_TIMER5\_CONF(0x8821007C): Timer5 Clock Config

NAME	D1	D0
P_TIMER5_CONF	TIMER5_RST	TIMER5_STOP

TIMER5\_STOP: Stop TIMER5 Clock

0: Enable(Stop)

1: Disable

TIMER5\_RST: Reset TIMER5 Module

0: Enable(Reset)

1: Disable

### 9.3.26 P\_TIMER6\_CONF(0x88210080): Timer6 Clock Config

NAME	D1	D0
------	----	----

P_TIMER6_CONF	TIMER6_RST	TIMER6_STOP
---------------	------------	-------------

TIMER6\_STOP: Stop TIMER6 Clock

0: Enable(Stop)

1: Disable

TIMER6\_RST: Reset TIMER6 Module

0: Enable(Reset)

1: Disable

### 9.3.27 P\_WDOG\_CONF(0x88210084): Watch Dog Clock Config

NAME	D1	D0
P_WDOG_CONF	WDOG_RST	WDOG_STOP

WDOG\_STOP: Stop Watch Dog Clock

0: Enable(Stop)

1: Disable

WDOG\_RST: Reset Watch Dog Module

0: Enable(Reset)

1: Disable

### 9.3.28 P\_RTC\_CONF(0x88210088): RTC Clock Config

NAME	D1	D0
P_RTC_CONF	RTC_RST	RTC_STOP

RTC\_STOP: Stop RTC Clock

0: Enable(Stop)

1: Disable

RTC\_RST: Reset RTC Module

0: Enable(Reset)

1: Disable

### 9.3.29 P\_I2S\_CONF(0x8821008C): I2S Clock Config

NAME	D3	D2	D1	D0
P_I2S_CONF	I2S_RST	I2SM_STOP	I2Sx256_STOP	I2SAPB_STOP

I2SAPB\_STOP: Stop I2S APB Clock

0: Enable(Stop)

1: Disable

I2Sx256\_STOP: Stop I2S x 256 Clock

0: Enable(Stop)

1: Disable

I2SM\_STOP: Stop I2S Master Clock

0: Enable(Stop)

1: Disable

I2S\_RST: Reset I2S Module  
0: Enable(Reset)  
1: Disable

### 9.3.30 P\_CKG\_SEL\_I2S(0x88210090): I2S Master Clock Divided Setting

NAME	D5-D1	D0
P_CKG_SEL_I2S	DividedNum	I2SM_CLK

I2SM\_CLK: Select I2S Master Clock Divided Number  
0: select audio PLL output divided by 2 as the I2S master clock  
1: select audio PLL output divided by 4 as the I2S master clock  
DividedNum: I2S x256 clock is audio PLL output divided by (DividedNum +1),  
DividedNum =1 is not allowed

### 9.3.31 P\_I2C\_CONF(0x88210094): I2C Clock Config

NAME	D1	D0
P_I2C_CONF	I2C_RST	I2C_STOP

I2C\_STOP: Stop I2C Clock  
0: Enable(Stop)  
1: Disable  
I2C\_RST: Reset I2C Module  
0: Enable(Reset)  
1: Disable

### 9.3.32 P\_SPI\_CONF(0x88210098): SPI Clock Config

NAME	D1	D0
P_SPI_CONF	SPI_RST	SPI_STOP

SPI\_STOP: Stop SPI Clock  
0: Enable(Stop)  
1: Disable  
SPI\_RST: Reset SPI Module  
0: Enable(Reset)  
1: Disable

### 9.3.33 P\_SIO\_CONF(0x882100A0): SIO Clock Config

NAME	D1	D0
P_SIO_CONF	SIO_RST	SIO_STOP

SIO\_STOP: Stop SIO Clock  
0: Enable(Stop)  
1: Disable  
SIO\_RST: Reset SIO Module  
0: Enable(Reset)  
1: Disable

### 9.3.34 P\_SD\_CONF(0x882100A4): SDCard Clock Config

NAME	D1	D0
P_SD_CONF	SD_RST	SD_STOP

SD\_STOP: Stop SD Clock

0: Enable(Stop)

1: Disable

SD\_RST: Reset SD Module

0: Enable(Reset)

1: Disable

### 9.3.35 P\_FLASH\_CONF(0x882100A8): NAND-Type Flash Clock Config

NAME	D1	D0
P_FLASH_CONF	FLASH_RST	FLASH_STOP

FLASH\_STOP: Stop NAND-type FLASH Clock

0: Enable(Stop)

1: Disable

FLASH\_RST: Reset NAND-type FLASH Module

0: Enable(Reset)

1: Disable

### 9.3.36 P\_ADC\_CONF(0x882100AC): ADC Clock Config

NAME	D1	D0
P_ADC_CONF	ADC_RST	ADC_STOP

ADC\_STOP: Stop ADC Clock

0: Enable(Stop)

1: Disable

ADC\_RST: Reset ADC Module

0: Enable(Reset)

1: Disable

### 9.3.37 P\_CKG\_SEL\_ADC(0x882100B0): ADC Source Clock Select

NAME	D8	D7-D0
P_CKG_SEL_ADC	ADC_Source	DividedNUM

ADC\_Source: Select ADC Source Clock

0: Select audio PLL output as the ADC source clock

1: Select USB PLL output the ADC source clock

DividedNUM: ADC clock is ADC source clock divided by (DividedNUM + 1),

DividedNUM = 1 is not allowed

### 9.3.38 P\_PLLV\_EN(0x882100B4): Video PLL Enable Setting

NAME	D0
------	----

P_PLLV_EN	PLLV_EN
-----------	---------

PLLV\_EN: Video PLL Enable

0: Disable

1: Enable

### 9.3.39 P\_PLLV\_FREQUENCY(0x882100B8): Video PLL output Clock Frequency Select

NAME	D3-D0
P_PLLV_FREQUENCY	FREQUENCY

FREQUENCY: Decide video PLL output clock frequency

0000: Reserved

0001: Reserved

0011: 81MHz

0100: 87.75MHz

0101: 97.5MHz

0110: 101.25MHz

0111: 108MHz

1000: 114.75MHz

1001: 121.5MHz

1010: 128.25MHz

1011: 135MHz

1100: 141.75MHz

1101: 148.5MHz

1110: 155.25MHz

1111: 162MHz

### 9.3.40 P\_PLLAU\_CONF(0x882100BC): Audio and USB PLL Config

NAME	D2	D1	D0
P_ADC_CONF	USBPLL_EN	PLLA_CLK	PLLA_EN

PLLA\_EN: Audio PLL Enable

0: Disable Audio PLL

1: Enable Audio PLL

PLLA\_CLK: Audio PLL Output Clock Select

0: Audio PLL output clock is 73.728MHz

1: Audio PLL output clock is 67.6766MHz

USBPLL\_EN: USB PLL Enable

0: Disable USB PLL

1: Enable USB PLL

### 9.3.41 P\_CLR\_KEY\_CHANGE(0x882100C0): KEYCHG\_WAKEUP Clear Register

NAME	D0
P_CLR_KEY_CHANGE	KEYCHG_CLR

KEYCHG\_CLR: 0: Clear KEYCHG\_WAKEUP register

### 9.3.42 P\_LVR\_EN(0x882100C4): Low Voltage Reset Enable

NAME	D0
P_LVRE_EN	LVR_EN

LVR\_EN: LVR Enable

0: Disable LVR reset

1: Enable LVR reset

### 9.3.43 P\_BUFCTL\_CONF(0x882100C8): Buffer Control Clock Config

NAME	D1	D0
P_BUFCTL_CONF	BUFCTL_RST	BUFCTL_STOP

BUFCTL\_STOP: Stop BUFCTL Clock

0: Enable(Stop)

1: Disable

BUFCTL\_RST: Reset BUFCTL Module

0: Enable(Reset)

1: Disable

### 9.3.44 P\_LDMDMA\_CONF(0x882100CC): LDMDMA Clock Config

NAME	D1	D0
P_LDMDMA_CONF	LDMDMA_RST	LDMDMA_STOP

LDMDMA\_STOP: Stop LDMDMA Clock

0: Enable(Stop)

1: Disable

LDMDMA\_RST: Reset LDMDMA Module

0: Enable(Stop)

1: Disable

### 9.3.45 P\_IRQ\_CONF(0x882100D0): IRQ Clock Config

NAME	D0
P_IRQ_CONF	IRQ_RST

IRQ\_RST: Reset IRQCTL Module

0: Enable(Reset)

1: Disable

### 9.3.46 P\_EPP\_CONF(0x882100D4): EPP Clock Config

NAME	D1	D0
------	----	----

P_EPP_CONF	EPP_RST	EPP_STOP
------------	---------	----------

EPP\_STOP: Stop EPP Clock

0: Enable(Stop)

1: Disable

EPP\_RST: Reset EPP Module

0: Enable(Reset)

1: Disable

#### 9.3.47 P\_CKG\_SEL\_CDS (0x882100D8): CD-Servo Clock Select

NAME	D0
P_CKG_SEL_CDS	CDS_Clock

CDS\_Clock: Set CD-Servo Clock Inverted or Non-inverted

0: Select inverted bit clock from CD servo

1: Select non-inverted bit clock from CD servo

#### 9.3.48 P\_CKG\_CONF(0x882100DC): Sleep Mode Clock Select

NAME	D0
P_CKG_CONF	SLEEP_CLK

SLEEP\_CLK Sleep Mode Clock Select

0: Select video PLL output divided by 8 as the clock used in sleep mode

1: Select 32768Hz clock as the clock used in sleep mode

#### 9.3.49 P\_TBASE\_CONF(0x882100E0): Time Base Clock Config

NAME	D2	D1	D0
P_TBASE_CONF	TBASE_RST	--	TBASE_STOP

TBASE\_STOP: Stop TBASE Clock

0: Enable(Stop)

1: Disable

TBASE\_RST: Reset TBASE Module

0: Enable(Reset)

1: Disable

#### 9.3.50 P\_CKG\_SEL\_TIMER (0x882100E4): Each Timer Source Clock Select

NAME	D13	D12	D11	D10	D9	D8	D7-D0
P_CKG_SEL_TIMER	Timer6C	Timer5C	Timer4C	Timer3C	Timer2C	Timer1C	DividedNum

DivideNum: TIMER source clock is 27MHz clock divided by (DivideNum +1)

Timer1C: TIMER1 Source Clock Select

0: Select TIMER source clock as the clock for TIMER 1

1: Select 32768Hz clock as the clock for TIMER 1

Timer2C: TIMER2 Source Clock Select

0: Select TIMER source clock as the clock for TIMER 2

- 1: Select 32768Hz clock as the clock for TIMER 2
- Timer3C: TIMER3 Source Clock Select
- 0: Select TIMER source clock as the clock for TIMER 3
- 1: Select 32768Hz clock as the clock for TIMER 3
- Timer4C: TIMER4 Source Clock Select
- 0: Select TIMER source clock as the clock for TIMER 4
- 1: Select 32768Hz clock as the clock for TIMER 4
- Timer5C: TIMER5 Source Clock Select
- 0: Select TIMER source clock as the clock for TIMER 5
- 1: Select 32768Hz clock as the clock for TIMER 5
- Timer6C: TIMER6 Source Clock Select
- 0: Select TIMER source clock as the clock for TIMER 6
- 1: Select 32768Hz clock as the clock for TIMER 6

### 9.3.51 P\_WDOG\_STATUS(0x882100E8): Watch Dog Interrupt Status

NAME	D1	D0
P_WDOG_STATUS	WDOG_INT	WDOG_ERRINT

WDOG\_ERRINT: Read 1: Reset Caused by Watch Dog error has occurred

Write 1: Clear this flag

WDOG\_INT: Read 1: Reset Caused by Watch Dog has occurred

### 9.3.52 P\_MP4\_CONF(0x882100EC): MP4 Clock Config

NAME	D2	D1	D0
P_MP4_CONF	MP4_RST	MP4M_STOP	MP4E_STOP

MP4E\_STOP: Stop MP4 Extra Clock

0: Enable(Stop)

1: Disable

MP4M\_STOP: Stop MP4 Main Clock

0: Enable(Stop)

1: Disable

MP4\_RST: Reset MP4 Module

0: Enable(Reset)

1: Disable

### 9.3.53 P\_CKG\_SEL\_MP4(0x882100F0): MP4 Clock Select

NAME	D1-D0
P_CKG_SEL_MP4	MP4_CLK

MP4\_CLK: 00: MP4 clock is MIU clock divided by 1

01: MP4 clock is MIU clock divided by 2

10: MP4 clock is MIU clock divided by 3

11: MP4 clock is MIU clock divided by 4

### 9.3.54 P\_USB\_DETECT\_SET (0x882100F4): USB Detection Setting

NAME	D1	D0
P_USB_DETECT_SET	USB_CLR	USB_EN

USB\_EN: USB Detection Enable Setting

0: Disable

1: Enable

USB\_CLR: USB Detection Clear Setting

0: Disable

1: Enable(Clear)

### 9.3.55 P\_UART\_DETECT\_SET (0x882100F8): UART Detection Setting

NAME	D1	D0
P_UART_DETECT_SET	UART_CLR	UART_EN

UART\_EN: UART Detection Enable Setting

0: Disable

1: Enable

UART\_CLR: UART Detection Clear Setting

0: Disable

1: Enable(Clear)

### 9.3.56 P\_SFTCFG\_CONF(0x882100FC): Software Config Clock Config

NAME	D1	D0
P_SFTCFG_CONF	SFTCFG_RST	SFTCFG_STOP

SFTCFG\_STOP: Stop SFTCFG Clock

0: Enable(Stop)

1: Disable

SFTCFG\_RST: Reset SFTCFG Module

0: Enable(Reset)

1: Disable

### 9.3.57 P\_ECC\_CONF(0x88210100): ECC Clock Config

NAME	D1	D0
P_ECC_CONF	ECC_RST	ECC_STOP

ECC\_STOP: Stop ECC Clock

0: Enable(Stop)

1: Disable

ECC\_RST: Reset ECC Module

- 0: Enable(Reset)  
1: Disable

**9.3.58 P\_PLL\_STIME(0x88210104): Stable PLL Output Clock Config**

NAME	D2-D0
P_PLL_STIME	Num

Num: After the video PLL is enabled from the disabled mode, counter triggered by video PLL output divided by 8 must count.

**9.3.59 P\_CKG\_STATUS(0x88210110): Wakeup Status for USB/ UART**

NAME	D1	D0
P_CKG_STATUS	UART_Wakeup	USB_Wakeup

USB\_Wakeup: USB Wakeup Status

- 0: Not occurred  
1: Occurred

UART\_Wakeup: UART Wakeup Status

- 0: Not occurred  
1: Occurred

**9.3.60 P\_CRYSTAL\_EN (0x88210114): Crystal Pad Enable Setting**

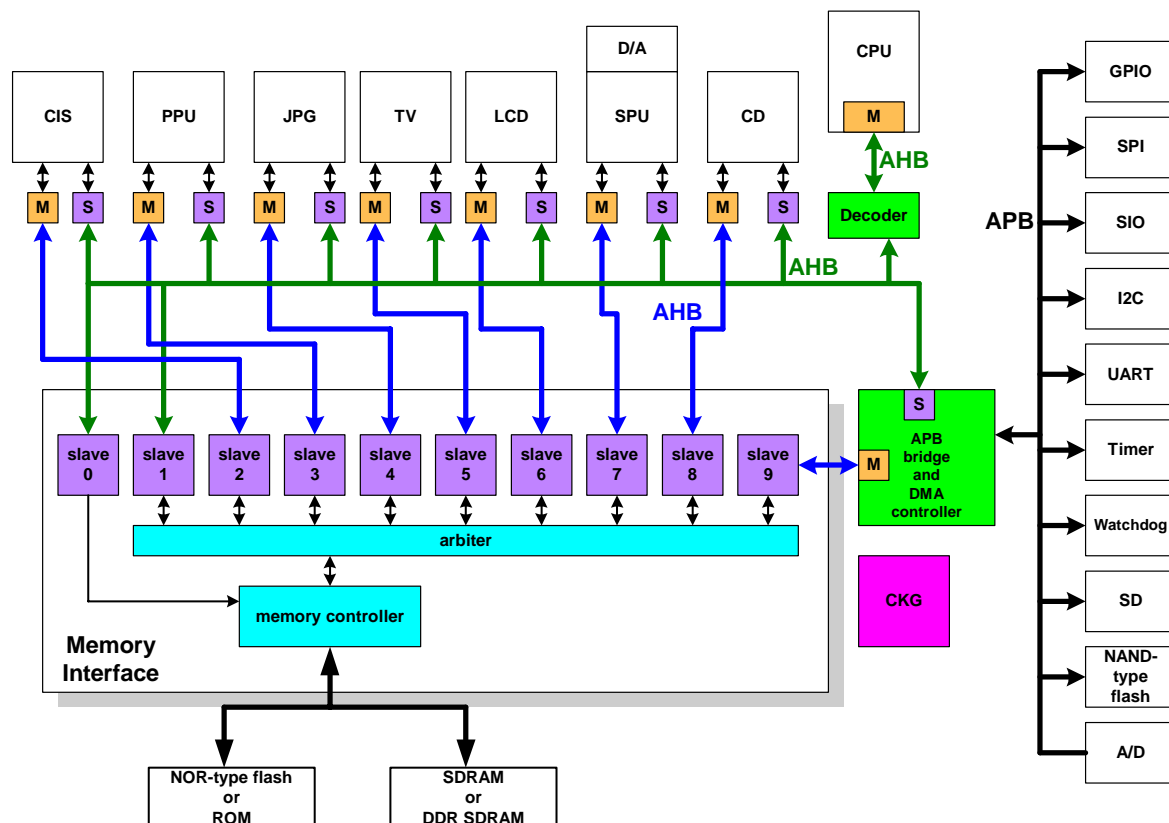
NAME	D0
P_CRYSTAL_EN	Crystal_En

Crystal\_En: 32768Hz Crystal Pad Enable

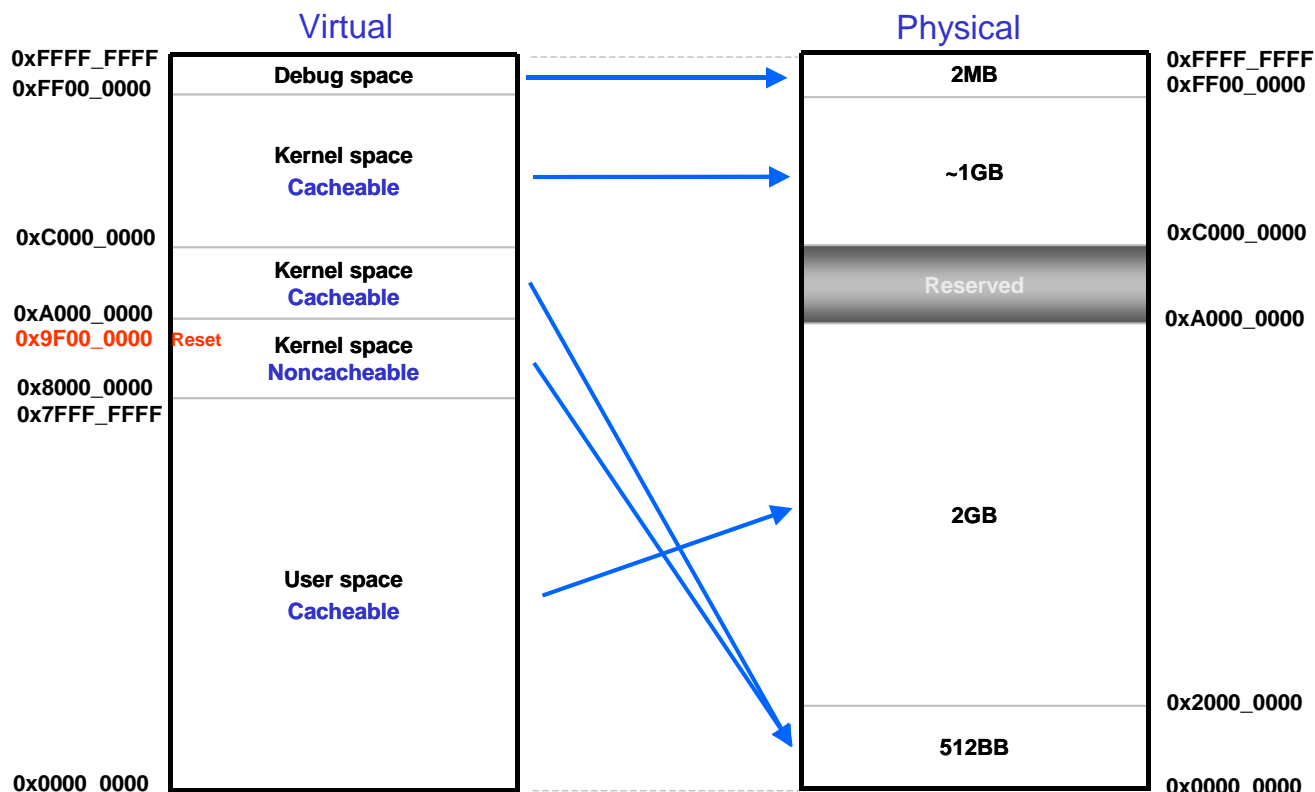
- 0: Disable  
1: Enable

## 10 MEMORY INTERFACE UNIT - MIU

MIU supports several different types of external memories, SDRAM, EDO DRAM, Parallel ROM, and also Nand Flash to let users use it to manage and design their application systems with more flexible selection of memory chips. In addition, MIU also manages two internal embedded memory blocks inside SPG290: 64K bits internal SRAM as **Local Data Memory** and 256K bits ROM as embedded **BOOT ROM**.



## 10.1 Memory Mapping



The memory map of AHB decoder is as follows:

0x0000\_0000 ~ 0x1FFF\_FFFF is non-cacheable region,

0x2000\_0000 ~ 0x3FFF\_FFFF is cacheable region.

Take notice that the region with color only represents the address memory space. Take

0x0A000000 ~ 0x0AFFFFFFF for instance, this 16M Bytes space only means this region points to the embedded SRAM, not real implemented Memory space embedded for S<sup>+</sup>Core is as follows:

Name	Address Range	Size
Internal boot ROM	0xB0000000 ~ 0xBFFFFFFF	32k bytes
Internal SRAM	0xA0000000 ~ 0xAFFFFFFF	0
LDM	Configurable	8k bytes

## 10.2 System registers : Each module used 32k bytes range.

Range	Describe
0x0800_0000 ~ 0x0800_FFFF	CSI
0x0801_0000 ~ 0x0801_FFFF	PPU
0x0802_0000 ~ 0x0802_FFFF	JPG
0x0803_0000 ~ 0x0803_FFFF	TV
0x0804_0000 ~ 0x0804_FFFF	LCD
0x0805_0000 ~ 0x0805_FFFF	SPU
0x0806_0000 ~ 0x0806_FFFF	CD
0x0807_0000 ~ 0x0807_FFFF	MIU(system reg)

0x0808_0000 ~ 0x0808_FFFF	<b>APBDMA(system reg)</b>
0x0809_0000 ~ 0x0809_FFFF	<b>BUFCTL</b>
0x080A_0000 ~ 0x080A_FFFF	<b>IRQCTL</b>
0x080C_0000 ~ 0x080C_FFFF	<b>LDMDMA</b>
0x080D_0000 ~ 0x080D_FFFF	<b>BLNDMA</b>
0x080F_0000 ~ 0x080F_FFFF	<b>AHBDEC</b>
0x0810_0000 ~ 0x0810_FFFF	<b>GPIO</b>
0x0811_0000 ~ 0x0811_FFFF	<b>SPI</b>
0x0812_0000 ~ 0x0812_FFFF	<b>SIO</b>
0x0813_0000 ~ 0x0813_FFFF	<b>I2C</b>
0x0814_0000 ~ 0x0814_FFFF	<b>I2S</b>
0x0815_0000 ~ 0x0815_FFFF	<b>UART</b>
0x0816_0000 ~ 0x0816_0FFF	<b>TIMER1</b>
0x0816_1000 ~ 0x0816_1FFF	<b>TIMER2</b>
0x0816_2000 ~ 0x0816_2FFF	<b>TIMER3</b>
0x0816_3000 ~ 0x0816_3FFF	<b>TIMER4</b>
0x0816_4000 ~ 0x0816_4FFF	<b>TIMER5</b>
0x0816_5000 ~ 0x0816_5FFF	<b>TIMER6</b>
0x0816_6000 ~ 0x0816_6FFF	<b>RTC</b>
0x0817_0000 ~ 0x0817_FFFF	<b>WDOG</b>
0x0818_0000 ~ 0x0818_FFFF	<b>SD</b>
0x0819_0000 ~ 0x0819_FFFF	<b>FLASH</b>
0x081A_0000 ~ 0x081A_FFFF	<b>ADC</b>
0x081B_0000 ~ 0x081B_FFFF	<b>USB device</b>
0x081C_0000 ~ 0x081C_FFFF	<b>USB host</b>
0x0820_0000 ~ 0x0820_FFFF	<b>SFTCFG</b>
0x0821_0000 ~ 0x0821_FFFF	<b>CKG</b>
0x0822_0000 ~ 0x0822_FFFF	<b>MP4</b>
0x0823_0000 ~ 0x0823_FFFF	<b>MIU2(system reg)</b>
0x0824_0000 ~ 0x0824_FFFF	<b>ECC</b>

### 10.3 Control Registers

MIU have some control registers, which is read/write by AHB slave 0 by CPU.

#### 10.3.1 TVE frame buffer start address

NAME	ADDR	D23-D0
<b>P_TV_START_ADR1</b>	0x88070000	TV_START_ADR1
<b>P_TV_START_ADR2</b>	0x88070004	TV_START_ADR2
<b>P_TV_START_ADR3</b>	0x88070008	TV_START_ADR3

TV\_START\_ADRx: The start address of TV frame buffer x

The D0-D7 must be zero

#### 10.3.2 LCD frame buffer start address

NAME	ADDR	D23-D0
<b>P_LCD_START_ADR1</b>	0x8807000C	PPU_LCD_ADR1
<b>P_LCD_START_ADR2</b>	0x88070010	PPU_LCD_ADR2
<b>P_LCD_START_ADR3</b>	0x88070014	PPU_LCD_ADR3

LCD\_START\_ADRx: The start address of LCD frame buffer x

The D0-D7 must be zero

### 10.3.3 P\_SPU\_START\_ADR(0x88070054): SPU Buffer Start Address

NAME	D23-D0
P_SPU_START_ADR	PPU_SPU_ADR

PPU\_SPU\_ADR: The start address of SPU buffer

The D0-D7 must be zero

### 10.3.4 P\_SDRAM\_POWER\_DOWN (0x8807005C): SDRAM Power Setting

NAME	D3	D2	D1	D0
P_SDRAM_POWER_DOWN	DEEP	REDO_MRS	POWER_DW	SELF_REF

SELF\_REF: Force SDRAM to enter self-refresh mode

0: Disable

1: Enable

POWER\_DW: Force SDRAM to enter power-down mode

0: Disable

1: Enable

REDO\_MRS: Re-Do MRS

0: Disable

1: Enable

DEEP: Force SDRAM to enter deep power-down mode

0: Disable

1: Enable

### 10.3.5 P\_MIU1\_SDRAM\_SETTING(0x88070060): MIU1 SDRAM Settings

NAME	D14-D11	D10-D3	D2	D1	D0
P_MIU1_SDRAM_SETTING	RFR_CYCL E_NUM	ATRF_R_P RIOD	CAS_LA_C CLE	tRCD_CYC LE	tRP_CYCLE
NAME	D31	D30-D27	D26-D25	D24-D23	D22-D15
P_MIU1_SDRAM_SETTING	MIU_EN	SDCLK_SEL	SDRAM_RN	SDRAM_CW	SFRFR_P RIOD

tRP\_CYCLE: tRP cycles setup

0: tRP is 1 cycle

1: tRP is 2 cycles(Default)

tRCD\_CYCLE: tRCD cycles setup

0: tRCD is 1 cycle

1: tRCD is 2 cycles(Default)

CAS\_LA\_CYCLE: Cas latency cycles setup

0: Cas Latency is 2 cycles(Default)

1: Cas Latency is 3 cycles

ATRF\_R\_PERIOD: Every( ATRFR\_PERIOD x 16) cycles, a auto-refresh is issued

RFR\_CYCLE\_NUM: Decide how many cycles is needed for a auto-refresh

SFRFR_PERIOD:	If ADG_REQ doesn't occur for (SFRFR_PERIOD x 256 x ATRFR_PERIOD x 16) cycles, a hardware-triggered self-refresh is issued
SDRAM_CW:	SDRAM Col Width
00:	8bits word SDRAM is used
01:	16bits word SDRAM is used
10:	32bits word SDRAM is used
11:	Reserved
SDRAM_RN:	SDRAM Row Number
00:	1024 rows per bank
01:	2048 rows per bank
10:	4096 rows per bank
11:	8192 rows per bank
SDCLK_SEL:	SDRAM clock selection
	Select the phase delay of SDRAM_CLK
MIU_EN:	MIU enable setup
0:	Disable
1:	Enable

### 10.3.6 P\_MIU1\_STATUS(0x8807006C): MIU1 SDRAM Status

NAME	D0
P_MIU1_SDRAM_SETTING	MIU_STATUS

MIU_STATUS:	0:	Non-occurred
	1:	SDRAM is in the self-refresh mode or power-down mode

### 10.3.7 MP4 RAW Image frame buffer

NAME	ADDR	D23-D0
P_MP4RAW_START_ADR1	0x88070070	MP4RAW_START_ADR1
P_MP4RAW_START_ADR2	0x88070074	MP4RAW_START_ADR2
P_MP4RAW_START_ADR3	0x88070078	MP4RAW_START_ADR3

MP4RAW\_START\_ADRx: Start address of MP4 raw image frame buffer x

The D0-D7 must be zero

### 10.3.8 MP4 write frame buffer

NAME	ADDR	D23-D0
P_MP4W_START_ADR1	0x8807007C	MP4W_START_ADR1
P_MP4W_START_ADR2	0x88070080	MP4W_START_ADR2
P_MP4W_START_ADR3	0x88070084	MP4W_START_ADR3

MP4W\_START\_ADRx: Start address of MP4 write frame buffer x

The D0-D7 must be zero

### 10.3.9 MP4 VLC buffer

NAME	ADDR	D23- D0
P_MP4V_START_ADR1	0x88070088	MP4V_START_ADR1
P_MP4V_START_ADR2	0x8807008C	MP4V_START_ADR2

MP4V\_START\_ADRx: Start address of MP4 VLC buffer x

The D0-D9 must be zero

### 10.3.10 P\_MP4\_FRAME\_BUF\_HSIZE(0x88070090): MP4 Frame Buffer H-Size Setting

NAME	D9-D2	D1-D0
P_MP4_FRAME_BUF_HSIZE	MP4_FB_HSIZE	

MP4\_FB\_HSIZE: MP4 Frame Buffer H-Size

The number of pixels on a scan line.

### 10.3.11 P\_SDRAM\_SETTING2(0x88070094): tRP &tRCD Cycle setup for SDRAM

NAME	D1	D0
P_SDRAM_SETTING2	tRCD_CYCLE	tRP_CYCLE

tRP\_CYCLE: tRP cycles setup

0: See MIU1\_SDRAM\_SETTING[0]

1: tRP is 3 cycles

tRCD\_CYCLE: tRCD cycles setup

0: See MIU1\_SDRAM\_SETTING[1]

1: tRCD is 3 cycles

DDR\_INV\_CLK\_OFF: Turn off the inverse clock for DDR

0: Disable

1: Enable(Turn Off)

## 10.4 Arbiter

Memory request priority is TV > LCD > CSI, other module's priority is decided by round-robin.

## 11 BUFFER CONTROL - BUFCTL

Buffer control unit is a comprehensive hardware logic to coordinate all double or triple buffers switch in TV encode unit and PPU TEXTs. When PPU outputs double buffers with hardware buffer control function, buffer control unit will change the display area pointer in the TV encoder to the other PPU output buffer whenever it is fully repainted. Therefore, TV encoder will always project the updated PPU output frame buffer to TV screen automatically without any CPU program required for buffer switching. This will simplify the program flow and reduce the possible timing or interrupt function overhead problems in the system.

### 11.1 Control Registers

#### 11.1.1 P\_C2P\_SETTING(R/W)(0x88090000): CSI to PPU Setting

Name	D4	D3-D2	D1	D0
<b>P_C2P_SETTING</b>	MD_Write	MD_Read	Buffer_Mode	C2P_HW

MD\_Write: Select the Module which writes the buffer

- 0: CSI
- 1: MP4

MD\_Read: Select the Module which reads the buffer

- 0: PPU
- 1: MP4
- 2: TVE
- 3: LCD

Buffer\_Mode: Select Buffer Mode

- 0: Double buffer
- 1: Triple buffer

C2P\_HW: CSI to PPU Hardware buffer control Enable

- 0: Disable
- 1: Enable

#### 11.1.2 P\_PTR\_SETTING(R/W)(0x88090004): Control type

Name	D7-D6	D5-D4	D3	D2	D1	D0
<b>P_PTR_SETTING</b>	TVE_Ctrl	PPU_Ctrl	Text3_Ctrl	Text2_Ctrl	Text1_Ctrl	CSI_Ctrl
Name	D31-D18	D17-D16	D15	D14-D12	D11-D10	D9-D8
<b>P_PTR_SETTING</b>		MP4_Ctrl	MP4_VLC	MP4_Write	MP4_RAW	LCD_Ctrl

CSI\_Ctrl: Software/Hardware buffer control select

- 0: Use software buffer control for CSI
- 1: Use hardware buffer control for CSI

Text1_Ctrl:	Software/Hardware buffer control select
	0: Use software buffer control for PPU TEXT1
	1: Use hardware buffer control for PPU TEXT1
Text2_Ctrl:	Software/Hardware buffer control select
	0: Use software buffer control for PPU TEXT2
	1: Use hardware buffer control for PPU TEXT2
Text3_Ctrl:	Software/Hardware buffer control select
	0: Use software buffer control for PPU TEXT3
	1: Use hardware buffer control for PPU TEXT3
PPU_Ctrl:	Software/Hardware buffer control select
	0: Use software buffer control for PPU
	1: Use P2T (PPU to TV encoder) hardware buffer control for PPU
	2: Use P2L (PPU to LCD) hardware buffer control for PPU
	3: Use HW4 hardware buffer control for PPU
TVE_Ctrl:	Software/Hardware buffer control select
	0: Use software buffer control for TVE
	1: Use C2P(CSI to PPU) hardware buffer control for TVE
	2: Use P2T (PPU to TV encoder) hardware buffer control for TVE
	3: Use HW4 hardware buffer control for TVE
LCD_Ctrl:	Software/Hardware buffer control select
	0: Use software buffer control for LCD
	1: Use C2P(CSI to PPU) hardware buffer control for LCD
	2: Use P2L(PPU to LCD) hardware buffer control for LCD
	3: Reserved
MP4_RAW:	MP4 RAW image Buffer control select
	0: Use software buffer control for MP4 raw image
	1: Use C2P_W_PTR_M1 for MP4 raw image
	2: Use C2P_R_PTR as the MP4 raw image buffer pointer
	3: Use C2P_W_PTR_M1_SPL for MP4 raw image
MP4_Write:	MP4 write buffer control select
	0: Use software buffer control for MP4 write buffer
	1: Use C2P hardware buffer control for MP4 write buffer
	2: Use P2T hardware buffer control for MP4 write buffer
	3: Use P2L hardware buffer control for MP4 write buffer

MP4_VLC:	4:	Use hardware buffer control for MP4 write buffer
	MP4 VLC buffer control select	
MP4_Ctrl:	0:	Use software buffer control for MP4 VLC buffer
	1:	Use hardware buffer control for MP4 VLC buffer
	MP4 control	
	0:	No hardware triggering signal for MP4 encoding
	1:	Use CSI's frame end as the triggering signal for MP4 encoding
	2:	Use TVE's frame end as the triggering signal for MP4 encoding

\* P2T mode means TV encoder will access one of the PPU output buffer which has just been updated by PPU for displaying to TV screen automatically

#### 11.1.3 P\_CSI\_BUF\_PTR(R/W)(0x88090008): Software buffer pointer for CSI

Name	D1-D0
P_CSI_BUF_PTR	CSI_PTR

CSI\_PTR: Write this register to set which is the active frame buffer from 0 to 2 of CSI  
Read this register to get which is the active frame buffer from 0 to 2 of CSI

0:	CSI frame buffer 0
1:	CSI frame buffer 1
2:	CSI frame buffer 2

#### 11.1.4 P\_TEXT1\_BUF\_PTR(R/W)(0x8809000C): Software buffer pointer for Text1

Name	D1-D0
P_TEXT1_BUF_PTR	TEXT1_PTR

TEXT1\_PTR: Write this register to set which is the active frame buffer from 0 to 2 of PPU TEXT1;  
Read this register to get which is the active frame buffer from 0 to 2 of PPU TEXT1

0:	TEXT1 frame buffer0
1:	TEXT1 frame buffer1
2:	TEXT1 frame buffer2

#### 11.1.5 P\_TEXT2\_BUF\_PTR(R/W)(0x88090010): Software buffer pointer for Text2

Name	D1-D0
P_TEXT2_BUF_PTR	TEXT2_PTR

TEXT2\_PTR: Write this register to set which is the active frame buffer from 0 to 2 of PPU TEXT2;  
Read this register to get which is the active frame buffer from 0 to 2 of PPU TEXT2

0:	TEXT2 frame buffer0
----	---------------------

1: TEXT2 frame buffer1

2: TEXT2 frame buffer2

#### 11.1.6 P\_TEXT3\_BUF\_PTR(R/W)(0x88090014): Software buffer pointer for Text3

Name	D1-D0
P_TEXT3_BUF_PTR	TEXT3_PTR

TEXT3\_PTR: Write this register to set which is the active frame buffer from 0 to 2 of PPU TEXT3;

Read this register to get which is the active frame buffer from 0 to 2 of PPU TEXT3

0: TEXT3 frame buffer0

1: TEXT3 frame buffer1

2: TEXT3 frame buffer2

#### 11.1.7 P\_PPU\_BUF\_PTR(R/W)(0x88090018): Software buffer pointer for PPU

Name	D1-D0
P_PPU_BUF_PTR	PPU_PTR

PPU\_PTR: Write this register to set which is the active frame buffer from 0 to 2 of PPU

Read this register to get which is the active frame buffer from 0 to 2 of PPU

0: PPU frame buffer0

1: PPU frame buffer1

2: PPU frame buffer2

#### 11.1.8 P\_TVE\_BUF\_PTR(R/W)(0x88090020): Software buffer pointer for TVE

Name	D1-D0
P_TVE_BUF_PTR	TVE_PTR

TVE\_PTR: Write this register to set which is the active frame buffer from 0 to 2 of TVE;

Read this register to get which is the active frame buffer from 0 to 2 of TVE

0: TVE frame buffer0

1: TVE frame buffer1

2: TVE frame buffer2

#### 11.1.9 P\_LCD\_BUF\_PTR(R/W)(0x88090024): Software buffer pointer for LCD

Name	D1-D0
P_LCD_BUF_PTR	LCD_PTR

LCD\_PTR: Write this register to set which is the active frame buffer from 0 to 2 of LCD;

Read this register to get which is the active frame buffer from 0 to 2 of LCD

0: LCD frame buffer0

1: LCD frame buffer1

## 2: LCD frame buffer2

**11.1.10 P\_BUFCTL\_STATUS(R/W)(0x88090028): Buffer Control Status**

Name	D0
<b>P_BUFCTL_STATUS</b>	CSI_Floss

CSI\_Floss: CSI frame loss interrupt flag

Write 0: Clear the interrupt flag

**11.1.11 P\_PPU\_FRAME\_CNT (R/W)(0x8809002C): PPU Frame Counter**

Name	D3-D0
<b>P_PPU_FRAME_CNT</b>	FrameCounter

FrameCounter: PPU Frame Counter

Write any value to reset the counter to 0.

**11.1.12 P\_PPU\_FCNT\_INC (R/W)(0x88090030): PPU Frame Counter Increase value**

Name	D31-D0
<b>P_PPU_FCNT_INC</b>	FCNT_INC

FCNT\_INC: PPU Frame Counter Increase value

Write any value to increase the PPU\_FRAME\_CNT counter by 1

**11.1.13 P\_TVE\_FRAME\_CNT (R/W)(0x88090034): TVE Frame Counter**

Name	D7-D0
<b>P_TVE_FRAME_CNT</b>	FrameCounter

FrameCounter: TVE Frame Counter

Write any value to reset the counter to 0.

**11.1.14 P\_P2T\_SETTING(R/W)(0x8809003C): PPU to TVE buffer control register**

Name	D2	D1	D0
<b>P_P2T_SETTING</b>	T2_HARD	T1_HARD	P2T_EN

P2T\_EN: Enable P2T hardware buffer control

0: Disable

1: Enable

T1\_HARD: Select the module which writes data

0: PPU

1: MP4

T2\_HARD: Select the module which reads data

0: TVE

1: LCD

**11.1.15 P\_P2L\_SETTING(R/W)(0x88090040): PPU to LCD buffer control register**

Name	D2	D1	D0
<b>P_P2L_SETTING</b>	T2_HARD	T1_HARD	P2L_EN

P2L\_EN: Enable P2L hardware buffer control

0: Disable

1: Enable

T1\_HARD: Select the module which writes data

0: PPU

1: MP4

T2\_HARD: Select the module which reads data

0: LCD

1: TVE

**11.1.16 P\_MP4RAW\_BUF\_PTR (R/W)(0x88090044): MP4 RAW Image Buffer Pointer**

Name	D1-D0
<b>P_MP4RAW_BUF_PTR</b>	MP4RAW_BUF

MP4RAW\_BUF: Software buffer pointer for MP4 RAW Image buffer

**11.1.17 P\_MP4W\_BUF\_PTR (R/W)(0x88090048): MP4 Re-Constructed Buffer Pointer**

Name	D1-D0
<b>P_MP4W_BUF_PTR</b>	MP4W_BUF

MP4W\_BUF: Software buffer pointer for MP4 re-constructed buffer

**11.1.18 P\_BUFCTL\_SETTING(R/W)(0x8809004C): Buffer Controller Interrupt Mask**

Name	D0
<b>P_BUFCTL_SETTING</b>	BUFCTL_SET

BUFCTL\_SET: Mask Buffer Control Interrupt to CPU

0: Disable

1: Enable(Mask)

**11.1.19 P\_MP4R\_BUF\_PTR (R/W)(0x88090050): MP4 reference buffer pointer**

Name	D1-D0
<b>P_MP4R_BUF_PTR</b>	MP4R_BUF

MP4R\_BUF: Software buffer pointer for MP4 reference buffer

**11.1.20 P\_MP4V\_BUF\_PTR(R/W)(0x88090054): MP4 VLC buffer pointer**

Name	D0
<b>P_MP4V_BUF_PTR</b>	MP4V_BUF

MP4V\_BUF: Software buffer pointer for MP4 VLC buffer

**11.1.21 P\_HW4\_SETTING(R/W)(0x88090058): HW4 Hardware Buffer Setting**

Name	D0
<b>P_HW4_SETTING</b>	HW4_EN

HW4\_EN: Enable/Disable HW4 hardware buffer control

0: Disable

1: Enable

**11.1.22 P\_FRAMENUM\_MP4ENC(R/W)(0x8809005C): MP4 encode base on TVE frame number**

Name	D1-D0
<b>P_FRAMENUM_MP4ENC</b>	FRAMENUM

FRAMENUM: Use Time of N+1 TVE frames to encode 1 MP4 frame

## 12 2D PICTURE PROCESS UNIT - PPU

PPU, Picture Process Unit, is a character base engine to render 512 foreground animation SPRITES and three background layers which are named TEXT in SPG290 system into a specified rectangle area for displaying to TV screen in the main external DRAM area. The PPU engine of SPG290 is inherited from SPG220 and enhances its computing efficiency to let SPG290 PPU can render up to 30 frames per second with 512 SPRITES and three 16bits colors TEXT layers in VGA mode. In addition, a 64 grade Alpha-Blending computing engine is included in SPG290 PPU, which can support each TEXT with different factor settings to let Game developers can add more special visual effect on the screen without slowing down the frame rate. The major features include:

- \* Support three screen modes:  
VGA (640x480 interlaced),  
QVGA (320x240 interlaced and progressive),  
HVGA screen mode (640x240 interlaced and progressive).
- \* All three TEXT layers support 64 order TEXT to TEXT alpha blending operation, 4/16/64/256 index color modes with characters and 16-bit BITMAP modes. The rectangular TEXT window is much larger than the physical screen size, so it can do screen scrolling more easily by changing the left-top coordinate of screen mapping on the TEXT window. The TEXT window size is dependent with screen mode. TEXT windows will be 1024 pixels wide and 512 pixels high in VGA mode, and 512 pixels wide and 256 pixels high in QVGA mode.
- \* Variable Character Size  
Size (H x V) 8x8 / 8x16 / 8x32 / 8x64  
16x8 / 16x16 / 16x32 / 16x64  
32x8 / 32x16 / 32x32 / 32x64  
64x8 / 64x16 / 64x32 / 64x64 pixels
- \* 512 SPRITE memory entries, each of them can have different SPRITE size, any one of above 16 variant character size table. The color mode of each SPRITE can also be different from others. In addition, with frame buffer architecture, there is no limitation of SPRITE numbers which can be shown in a frame or a line simultaneously.

Vertical Size (pixel)	Horizontal Size (pixel)			
	8	16	32	64
8	8 x 8	8 x 16	8 x 32	8 x 64
16	16 x 8	16 x 16	16 x 32	16 x 64
32	32 x 8	32 x 16	32 x 32	32 x 64
64	64 x 8	64 x 16	64 x 32	64 x 64

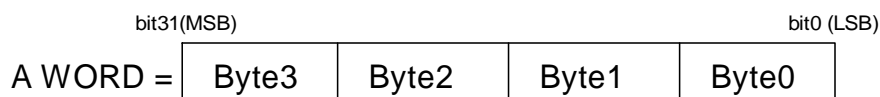
- \* Maximum 1024 color palette for indexed color
  - 512 palette for Texture entries for indexed color and each of them can be a transparent color index.
  - 512 palette for Sprite entries for indexed color and each of them can be a transparent color index.
- \* Maximum 256/32768/65536 colors simultaneous output
- \* Support 32768/65536 bitmap mode
- \* TEXT Horizontal lines Movement Control
- \* TEXT Compression / Extension function

## 12.1 Order of bytes

Data access in SPG290 32 bits system is **Little Endian**, including PPU. The most significant byte in a WORD, 32 bits data, is at the last place of a 4 bytes string.

Address or File order

Byte0
Byte1
Byte2
Byte3
Byte4



## 12.2 Color Mode

There are 6 color modes are available in SPG290. 4 modes with indexed color method, they are 2, 4, 6, 8 bits per pixel modes or 4, 16, 64, 256 indexed color mode. In BITMAP mode, 2 kinds of RGB pixel format can be applied, one is ARGB (1:5:5:5) where A at MSB is the enable bit of transparency and the other is RGB565 (5:6:5).

### 12.2.1 Color modes of SPRITE

SPRITE consists of Characters and 4/16/64/256 indexed color mode, BITMAP mode is not available for SPRITE.

### 12.2.2 Color modes of TEXT

TEXT can support both Character mode and BITMAP mode. With Character mode, 4/16/64/256 indexed color modes are all available. A TEXT with BITMAP mode means the frame buffer of TEXT is no longer filled with Character data, instead each 16 bits in it represent a physical pixel's color data. There are two of RGB pixel format defined in SPG290, one is ARGB and the other is RGB565.

## 12.3 Palette Memory

Pixel data of a character or a SPRITE are all Index numbers of Palette, not a real RGB color value. With different color modes, 4, 16, 64, or 256 colors, pixel formats in bits are also different. For example, in 16-color mode character, each pixel needs 4 bits, and 8 bits when 256 color mode. Palette Memory is a block of internal static RAM in SPG290 with 16 bits per Palette and totally 512 entries. In addition, with different color modes, Palette Memory can divide into several Banks so as let users can manage it more efficiently. The Palette Bank is also used in TEXT structure or attribute and SPRITE structure to calculate the real color address in Palette RAM.

### 12.3.1 Palette Bank

Color modes	Total Palette Banks
2 bits 4 color mode	32
4 bits 16 color mode	32
6 bits 64 color mode	8
8 bits 256 color mode	2

Physical Palette address = 0x81001000 + Palette Offset ;

Palette Offset = (Palette Bank x Color Mode(4/16/64/256) + Pixel Color Index) x 4

\* SPG290 use 4 bits to store 4 color mode pixel like 16 color mode

**Palette Bank Registers in PPU :**

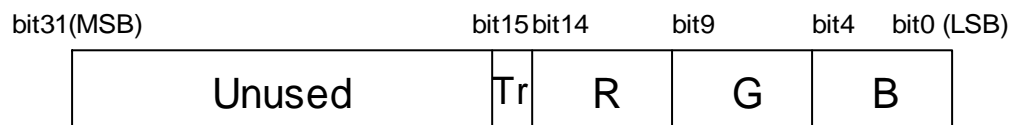
		B12	B11	B10	B9	B8
P_Tx1_attribute	0x8801_0028	Palette Bank of TEXT1				
P_Tx2_attribute	0x8801_0044	Palette Bank of TEXT2				
P_Tx3_attribute	0x8801_0060	Palette Bank of TEXT3				
P_Sprite(N)_Attribute	0x8801_4004+(8*N)	Palette Bank of Sprite(N)				

### 12.3.2 Pixel format of Palette

Each Palette entry is a 32-bit word, but only last significant half word is valid for storing color values.

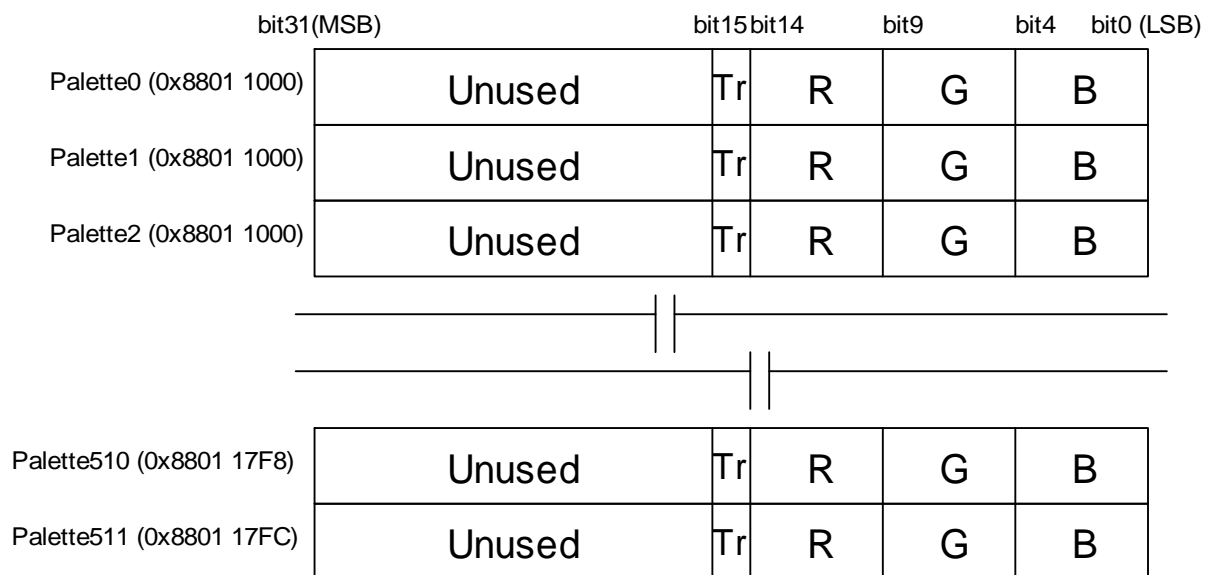
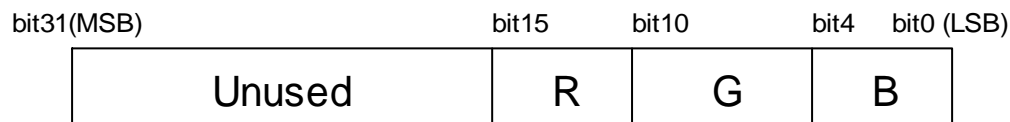
There are two RGB color pixel format defined in SPG290.

RGB555 mode:



Tr(bit15) : Transparent Bit. 1 : Transparent Color ; 0 : Solid Color

RGB565 mode:



## 12.4 Attributes

Each character/sprite comprises many attributes including color mode, depth, horizontal / vertical size, horizontal / vertical flip, blend and palette bank selection.

### 12.4.1 Palette

Denote the Palette Bank of Character, SPRITE or TEXT.

		B12	B11	B10	B9	B8
P_Tx1_attribute	0x8801_0028	Palette Bank of TEXT1				
P_Tx2_attribute	0x8801_0044	Palette Bank of TEXT2				
P_Tx3_attribute	0x8801_0060	Palette Bank of TEXT3				
P_Sprite(N)_Attribute	0x8801_4004+(8*N)	Palette Bank of Sprite(N)				

### 12.4.2 Depth

Depth defines the relevant position for characters and layers. There are four depth layers for both the sprites and the texts, as shown in the figure. The layer with larger depth number will be displayed

when two elements are overlapped in different depth. If Text1 and Text2 and Text3 are located at the same layer, the Text3 would be dominated in the overlap area. If two or more sprites are located at the same layer, the sprite with larger sprite memory address number is also dominated in the overlap area. In the figure below, the priority of sprite and text is: Sprite Layer3 > Text Layer3 > Sprite Layer 2 > Text Layer2 > Sprite Layer 1 > Text Layer 1 > Sprite Layer 0 > Text Layer 0 where ">" means on top.

		B14	B13
P_Tx1_attribute	0x8801_0028	Depth1	
P_Tx2_attribute	0x8801_0044	Depth2	
P_Tx3_attribute	0x8801_0060	Depth3	
P_Sprite(N)_Attribute	0x8801_4004+(8*N)	Sprite(N)_Depth	

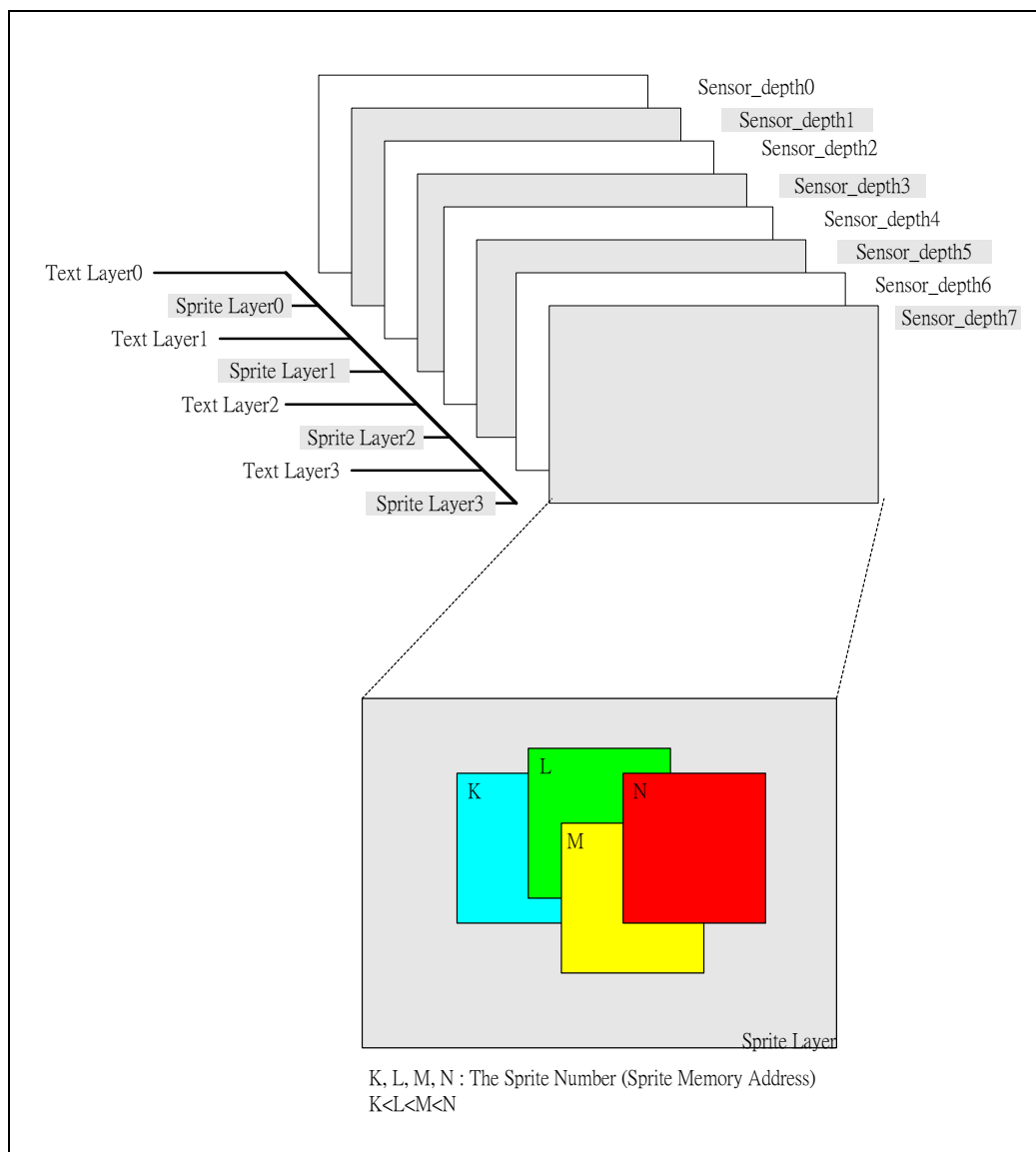
**Depth1:** Text1 Depth

**Depth2:** Text2 Depth

**Depth3:** Text3 Depth

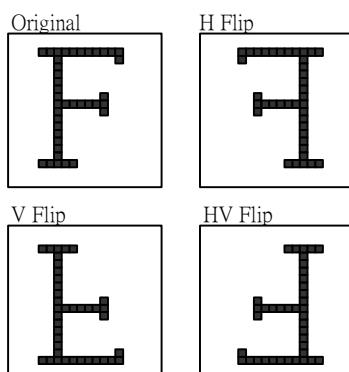
**Sprite(N)\_Depth:** The Nth Sprite Depth

00: Deepest layer ~ 511 11: Shallowest layer



### 12.4.3 Character Flip

Flip function means the mirror effect. The vertical and horizontal flips can be controlled independently. This function operates character by character individually. Please see the figures below for illustration.



		B3	B2	B1	B0
P_Tx1_attribute	0x8801_0028	Flip1			
P_Tx2_attribute	0x8801_0044	Flip2			
P_Tx3_attribute	0x8801_0060	Flip3			
P_Sprite(N)_Attribute	0x8801_4004+(8*N)	Sprite(N)_Flip			

**Flip1:** Text1 Character Flip

**Flip2:** Text2 Character Flip

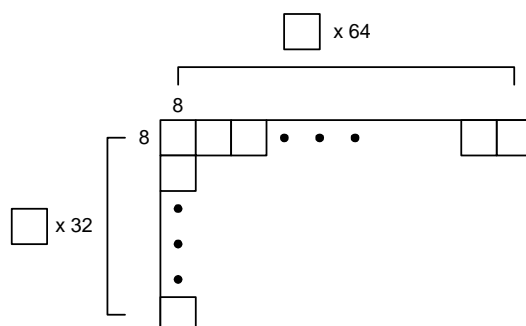
**Flip3:** Text3 Character Flip

**Sprite(N)\_Flip:** Nth Sprite Flip

00: No Flip      01: Horizontal Flip  
10: Vertical Flip    11: Vertical and Horizontal Flip

### 12.4.4 Character Size

Character size defines the number of pixel for each character. Each sprite has its own character size, in other words, sprite can be different character sizes. In contrast, a text (Text1 ,Text2 or Text3) can only allow one character size in it. For example, suppose 8x8 character-size is chosen for a text screen. As a result, the entire text screen is all filled with 8x8 character-size.



The following character sizes are available in SPG290.

Vertical Size (pixel)	Horizontal Size (pixel)			
	8	16	32	64
8	8 x 8	8 x 16	8 x 32	8 x 64
16	16 x 8	16 x 16	16 x 32	16 x 64
32	32 x 8	32 x 16	32 x 32	32 x 64
64	64 x 8	64 x 16	64 x 32	64 x 64

		B7	B6	B5	B4
P_Tx1_attribute	0x8801_0028	Vs1		Hs1	
P_Tx2_attribute	0x8801_0044	Vs2		Hs2	
P_Tx3_attribute	0x8801_0060	Vs3		Hs3	
P_Sprite(N)_Attribute	0x8801_4004+(8*N)	Sprite(N)_vsize		Sprite(N)_hsize	

**Vs1:** Text1 Character Vertical Size

**Hs1:** Text1 Character Horizontal Size

**Vs2:** Text2 Character Vertical Size

**Hs2:** Text2 Character Horizontal Size

**Vs3:** Text3 Character Vertical Size

**Hs3:** Text3 Character Horizontal Size

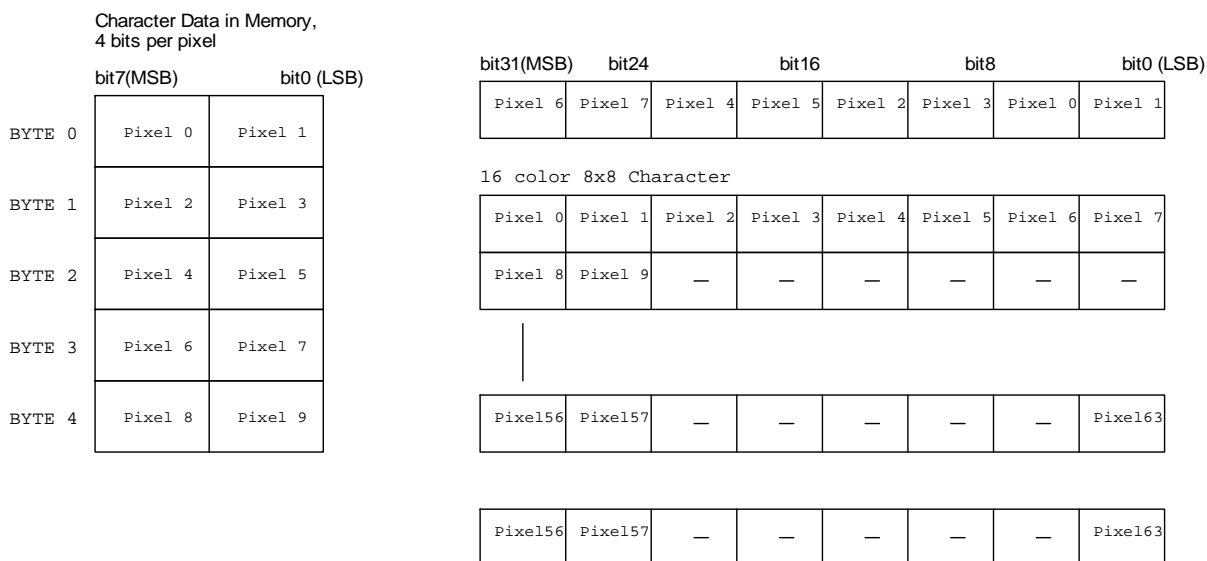
**Sprite(N)\_vsize:** Nth Sprite Vertical Size

**Sprite(N)\_hsize:** Nth Sprite Horizontal Size

00: 8 pixels      01: 16 pixels      10: 32 pixels      11: 64 pixels

#### 12.4.5 Character Data

Character data is a sequence of indexed color pixels in bytes. For example, a 16-bits 8x8 character data stored in Frame Buffer memory are as following sample.

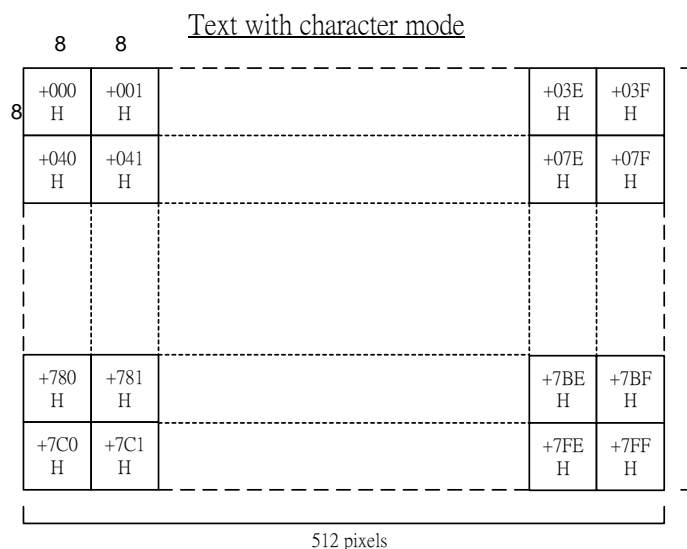


## 12.5 Text Screen

Text Screen consists of a set of characters (character mode) or lines (bitmap mode). The characters within a text must be the same size. This section describes the relationship between Text Memory content and Text Screen. The character mode and bitmap mode will be illustrated individually.

### 12.5.1 Character Mode

Text is formed by a set of characters. There are 16 character-size modes, but the size of a Text Screen is always 512 (H) x 256 (V) pixels in QVGA mode or 1024 (H) x 512 (V) pixels in VGA mode. The following diagram is an example with characters-size of 8x8 to form a text in QVGA mode.



### 12.5.2 Bitmap Mode

Text is formed by a set of pixels horizontally (from left to right), called bitmap mode. As a result, there are 512 horizontal pixels with 256 vertical pixels in CIF mode and 1024 horizontal pixels with 512 vertical pixels of VGA mode in bitmap mode. When this mode is used, the Vertical/Horizontal Flip Effect is not effective. In addition, the bitmap addressing Mode is automatically chosen. In bitmap addressing Mode, the pointer and attribute arrays store the address of each line. Each line has its pointer and attribute. The three text layers can be chosen as the bitmap mode or the character mode independently.

		<b>B3</b>	<b>B2</b>	<b>B1</b>	<b>B0</b>
<b>P_Tx1_control</b>	<b>0x8801_002c</b>	Txe1			Linr1
<b>P_Tx2_control</b>	<b>0x8801_0048</b>	Txe2			Linr2
<b>P_Tx3_control</b>	<b>0x8801_0064</b>	Txe3			Linr3

**Txe1:** Text1 Enable

0: Disable (Non-Visible)

1: Enable (Visible)

**Linr1:** Text1 Bitmap Mode Enable

0: Character Mode

1: Bitmap Mode

**Txe2:** Text2 Enable                      0: Disable (Non-Visible)                      1: Enable (Visible)

**Linr2:** Text2 Bitmap Mode Enable    0: Character Mode                      1: Bitmap Mode

**Txe3:** Text3 Enable                      0: Disable (Non-Visible)                      1: Enable (Visible)

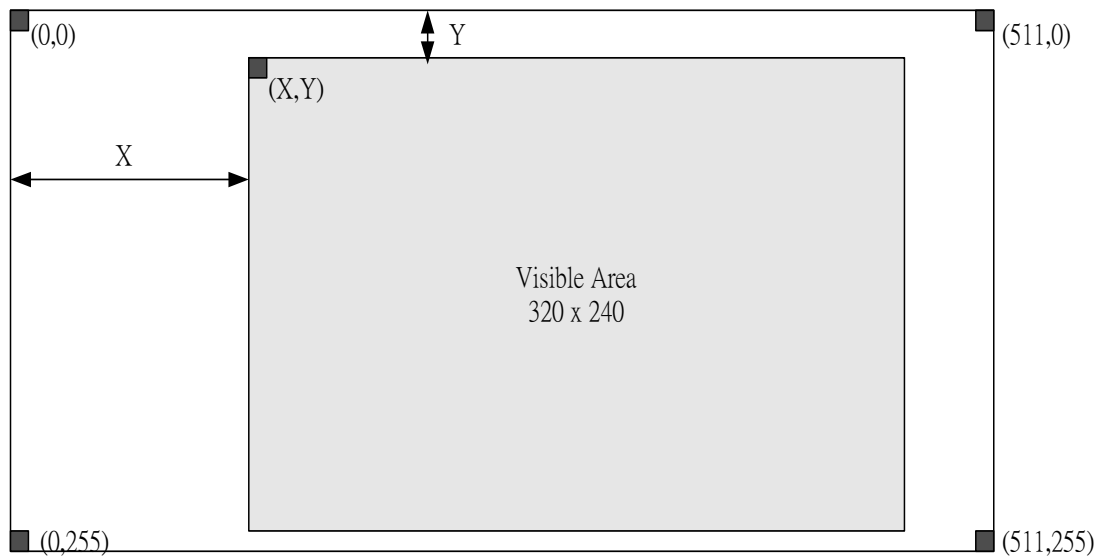
**Linr3:** Text3 Bitmap Mode Enable    0: Character Mode                      1: Bitmap Mode

## 12.6 Text Coordinate

The origin of Text Screen is located at the top-left corner. The area of Text Screen defined in the Text Memory is 512x256 in CIF Mode, while 320x240 is the visible area. The (X, Y) defines the position of visible window, shown in the diagram.

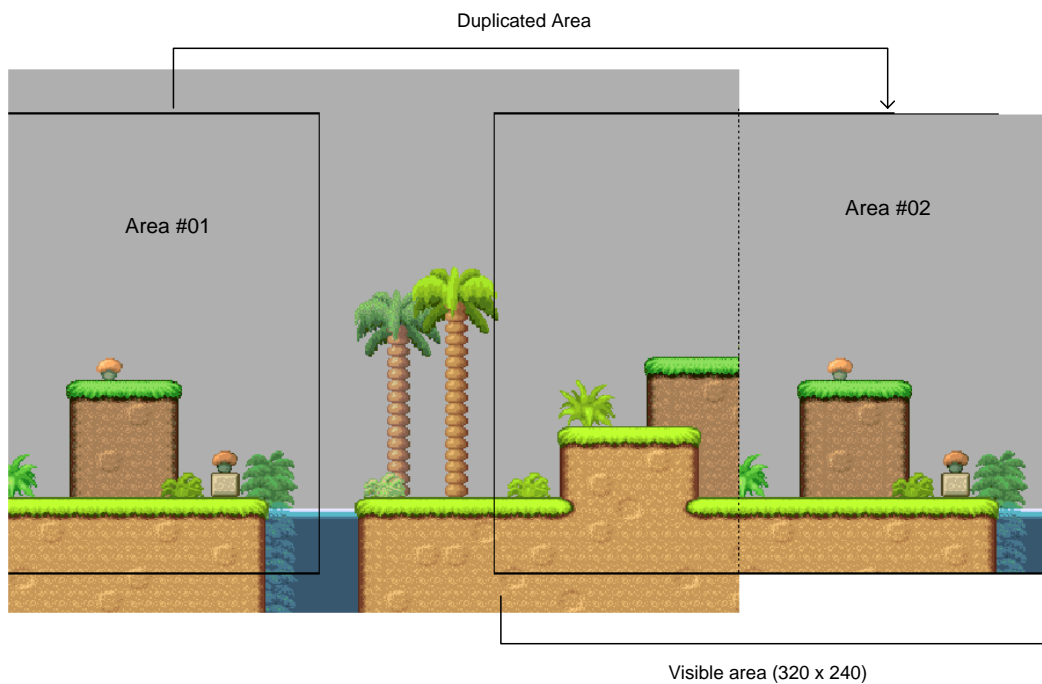
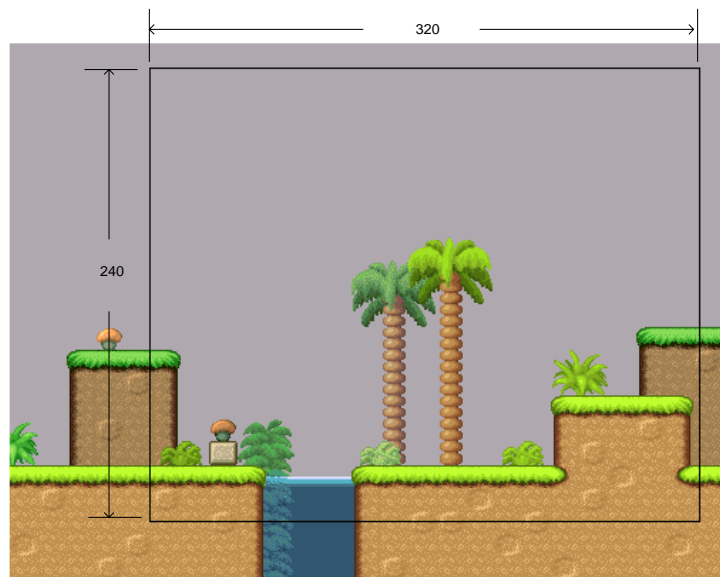
		B8	B7	B6	B5	B4	B3	B2	B1	B0
P_Tx1_X_position	0x8801_0020	X1								
P_Tx1_Y_position	0x8801_0024		Y1							
P_Tx2_X_position	0x8801_003c	X2								
P_Tx2_Y_position	0x8801_0040		Y2							
P_Tx3_X_position	0x8801_0058	X3								
P_Tx3_Y_position	0x8801_005c		Y3							

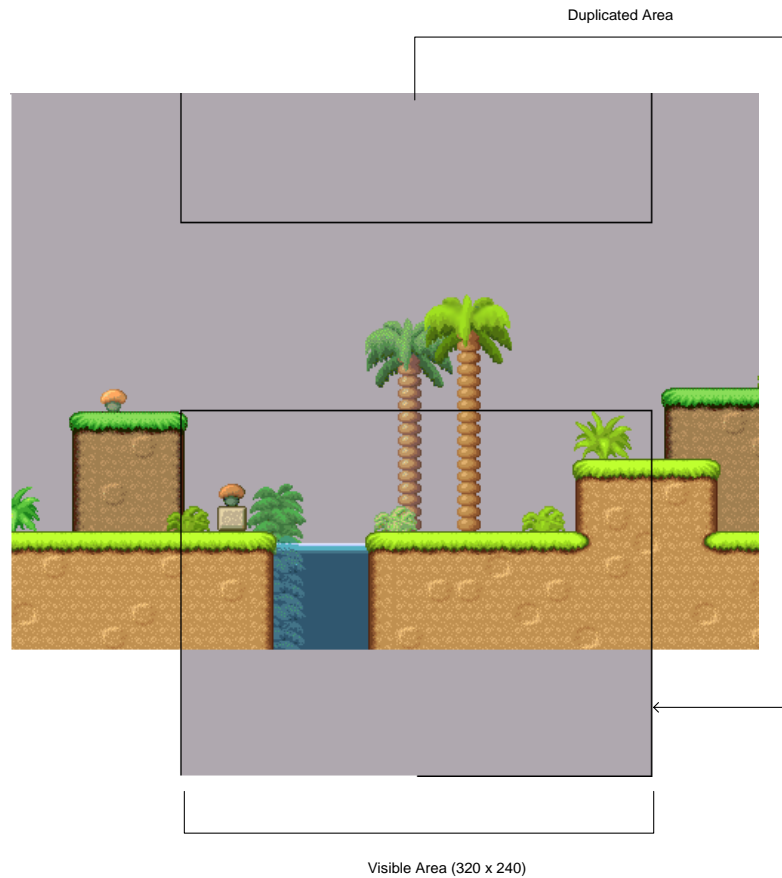
Text Coordinate (X,Y)



Text Screen loops up-down and left-right sides.

In Fig. 1, the visible area is 320 x 240 and the entire screen is 512 x 256 pixels. As the visible area moves to right and exceeds to the screen view, the background on the left (Area #01) will be duplicated to the right (Area #02), see Fig.2. Similarly, when the visible area moves down and exceeds the screen view, the background on the top will be duplicated to the bottom, see Fig. 3.

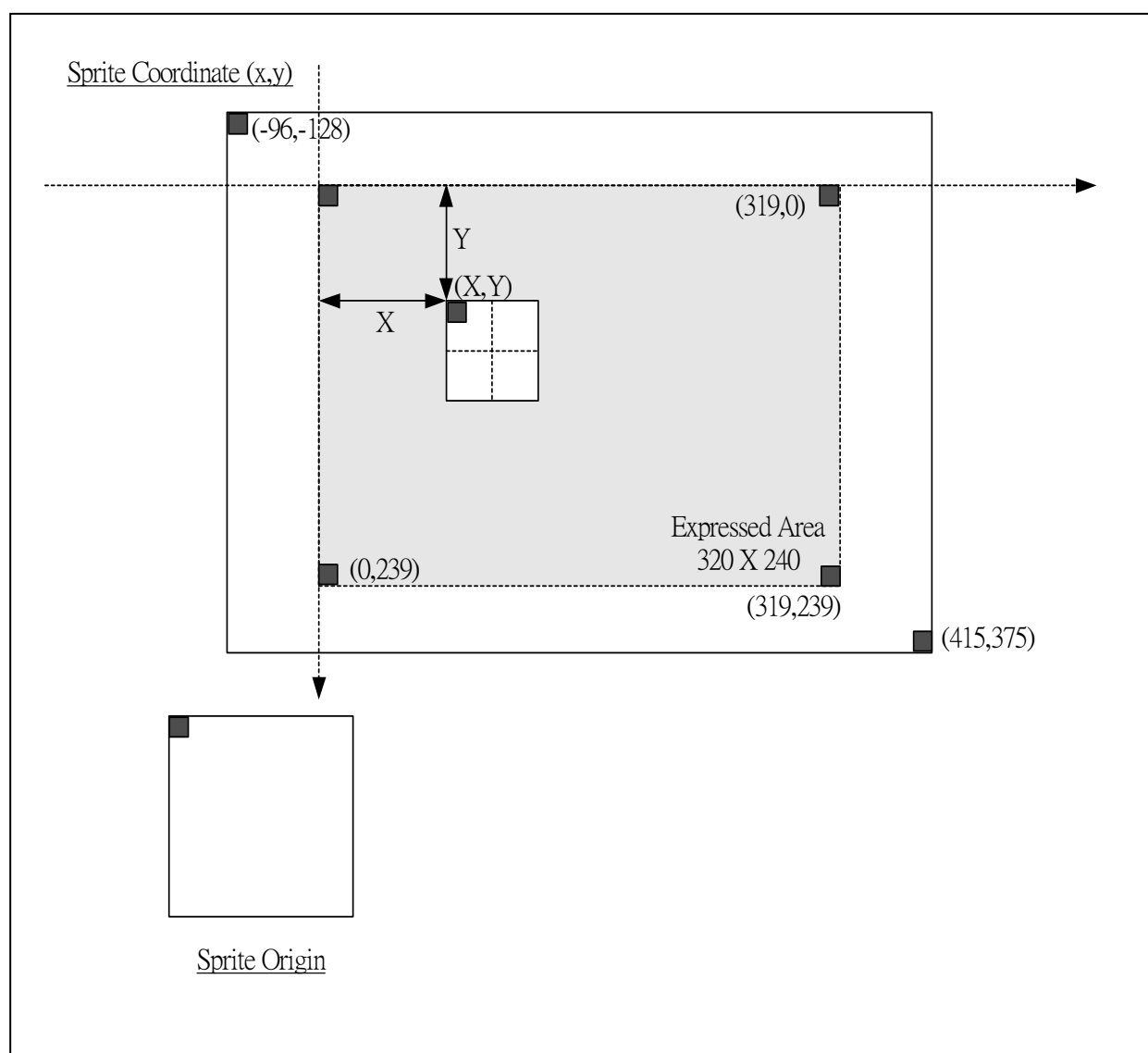




## 12.7 Sprite Coordinate

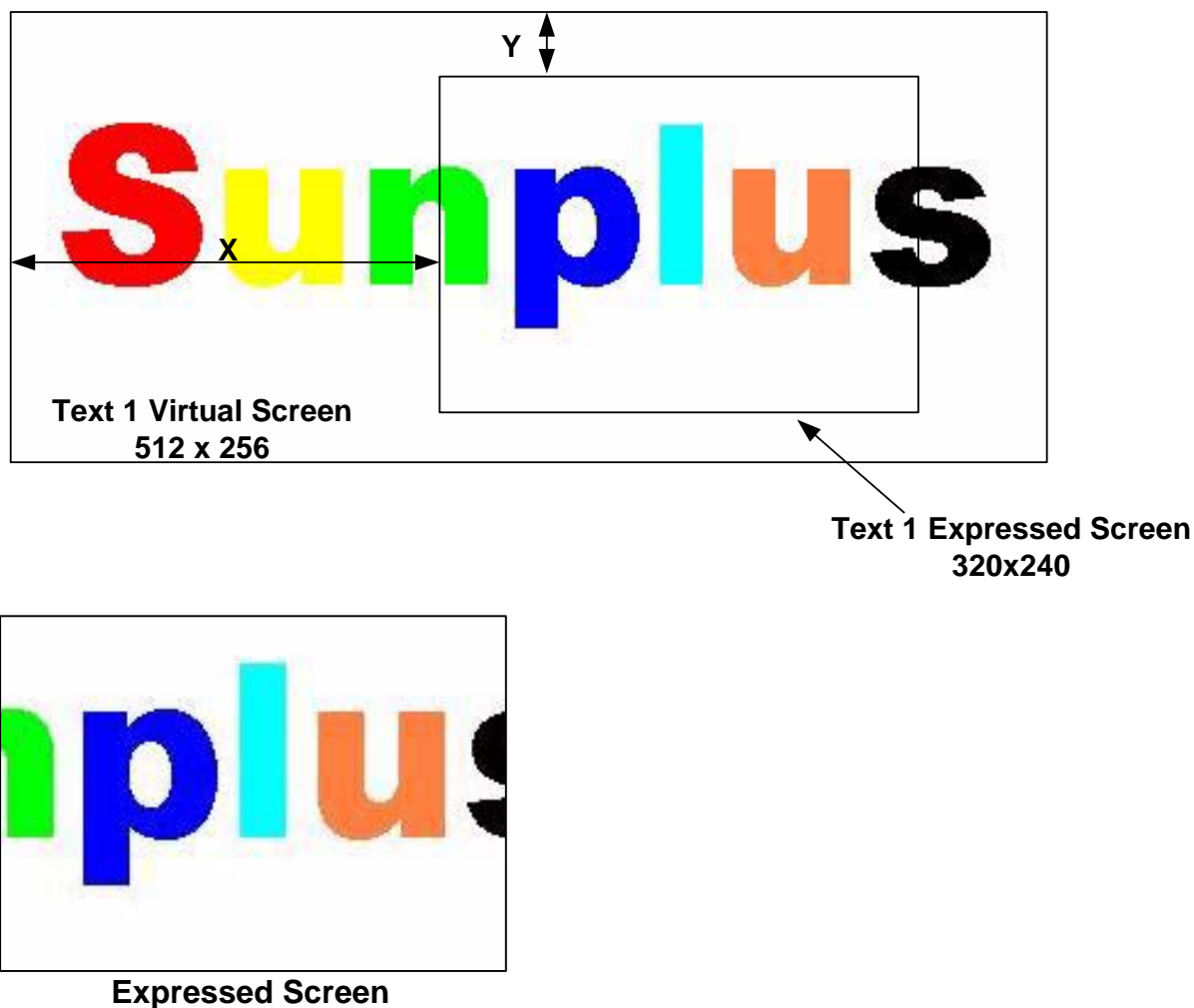
The origin of the sprite screen is at the left top corner of the visible area. The coordinate origin of sprite is at the left top of the sprite, as shown in the diagram.

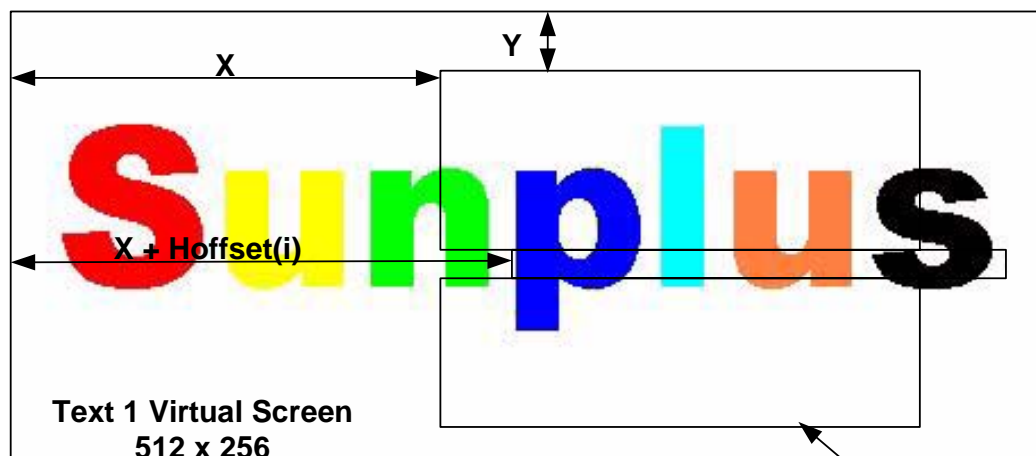
		D9	B8	B7	B6	B5	B4	B3	B2	B1	B0
<b>P_Sprite(N)_X</b>	<b>0x8801_4002+(8*N)</b>	Sprite(N)_X									
<b>P_Sprite(N)_Y</b>	<b>0x8801_4006+(8*N)</b>	Sprite(N)_Y									



## 12.8 Text Horizontal Movement Control

The **Horizontal Movement Control** provides the line-based individual horizontal movement control. Each horizontal scanning line has a horizontal movement offset value. The offset value will be added to the Text Position, described in the *Text Coordinate*.





**Text 1 Expressed Screen**  
320x240



**Expressed Screen**

These control registers Offset(i) are between 0x8801\_2000 to 0x8801\_27FC.

	B9 - b0
0x8801_2000	Offset0
0x8801_2004	Offset1
0x8801_27FC	Offset511

0x8801\_002c

b8	b7	b6	b5	b4	b3	b2	b1	b0
---	---	---	---	MVEN1	---	---	---	---

0x8801\_0048

b8	b7	b6	b5	b4	b3	b2	b1	b0
---	---	---	---	MVEN2	---	---	---	---

0x8801\_0064

b8	b7	b6	b5	b4	b3	b2	b1	b0
---	---	---	---	MVEN3	---	---	---	---

The MVEN is the switch for Horizontal movement control. When MVEN is given “high”, the offset value would be added to the Horizontal Position.

## 12.9 Text Compression/Extension Control

The **Text Compression Control** provides vertical compression (VCMP) and line-based individual horizontal compression (HCMP) controls. The whole text has a vertical compression value while each horizontal scanning line has its horizontal compression value.

### 12.9.1 Vertical compression

For vertical compression, VCMP1 & VCMP2 & VCMP3 in control registers 0x8801\_002c and 0x8801\_0048 and 0x8801\_0064 are the switches to enable/disable vertical compression. Moreover, the compression scale is given in 0x8801\_0074, and the line compression step is given in 0x8801\_0078. This scale and step are available for text1, text2, text3 or both text1 and text2 and text3, depending on the setting of VCMP1 & VCMP2 & VCMP3. However, there is only one scale for whole text.

0x8801\_002c

b8	b7	b6	b5	b4	b3	b2	b1	b0
---	---	VCMP1	---	---	---	---	---	---

0x8801\_0048

b8	b7	b6	b5	b4	b3	b2	b1	b0
---	---	VCMP2	---	---	---	---	---	---

0x8801\_0064

b8	b7	b6	b5	b4	b3	b2	b1	b0
---	---	VCMP3	---	---	---	---	---	---

0x8801\_0074

b8	b7	b6	b5	b4	b3	b2	b1	b0
VCMP Value								

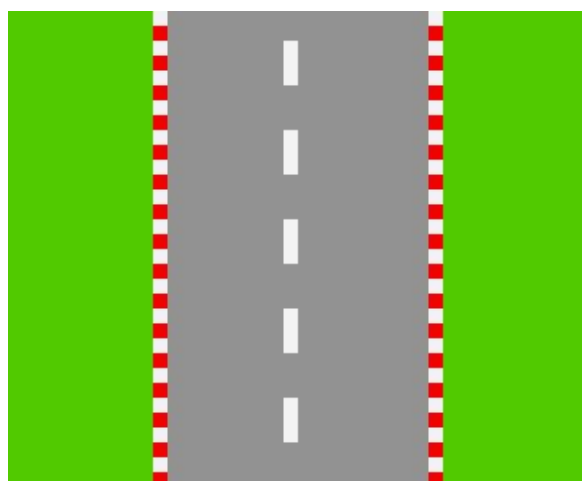
0x8801\_0078

B12 – b0								
VCMPoffset								

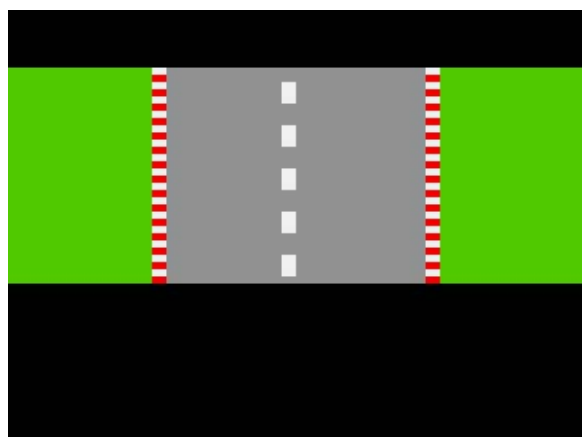
0x8801\_007c

b8	b7	b6	b5	b4	b3	b2	b1	b0
VCMP Step								

Note: When VCMP is given “high”, the corresponding text would be compressed by the scale set in 0x8801\_0074. “20h” means no compression; “40h” represents double compression and “10h” expands the size to double.

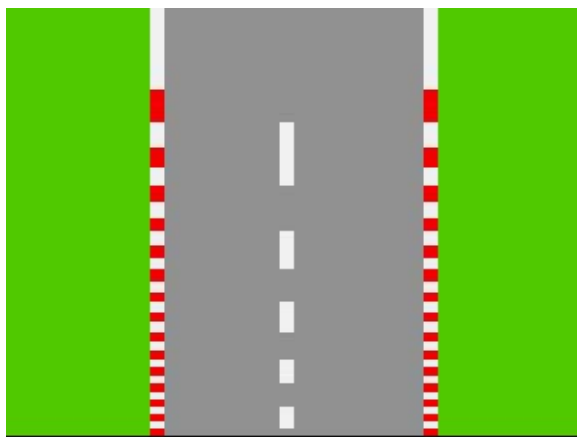


Vcmp\_Value = 20H  
Vcmp\_Step = 0H  
Vcmp\_Y\_offset = 0H

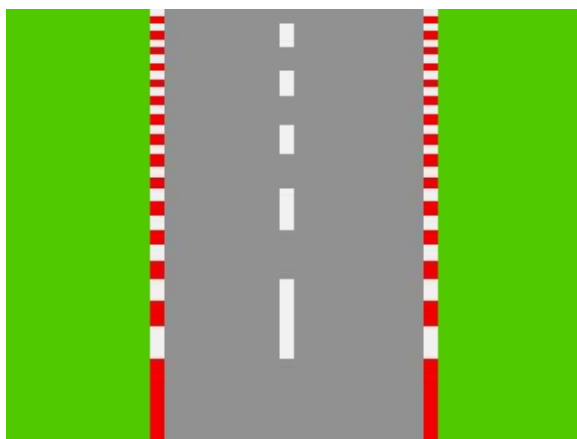


↓  
Vcmp\_Y\_offset = 20H  
↑

Vcmp\_Value = 40H  
Vcmp\_Step = 0H  
Vcmp\_Y\_offset = 20H



Vcmp\_Value = 0H  
Vcmp\_Step = 9H = +9  
Vcmp\_Y\_offset = 0H



Vcmp\_Value = 42H  
Vcmp\_Step = F7H = -9  
Vcmp\_Y\_offset = 0H

### 12.9.2 Horizontal compression

Different from vertical compression, users can choose different compression scale for each Horizontal scanning line in horizontal compression. The HCMP in control register 0x8801\_002c is the switch to control horizontal compression. The scales are set in 0x8801\_3000~0x8801\_37FC. *Note that horizontal compression is only available in text1 and character mode.*

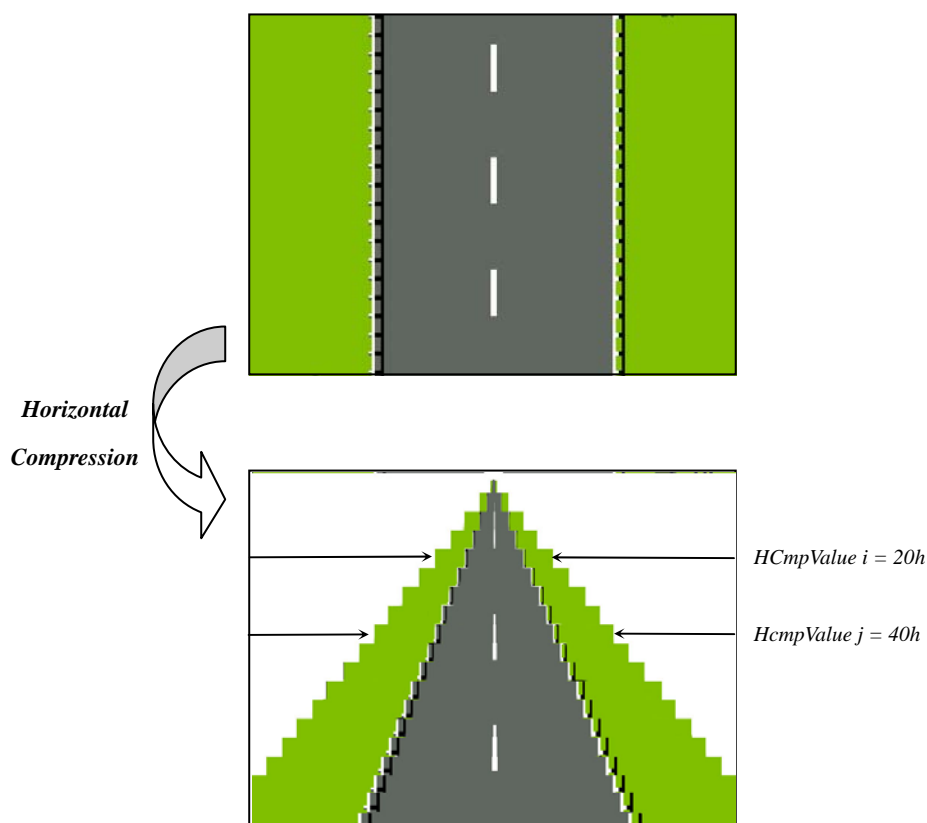
0x8801\_002c

b8	b7	b6	b5	b4	b3	b2	b1	b0
---			HCMP	---				

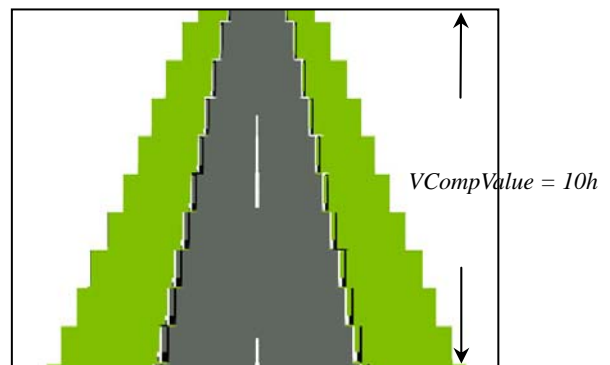
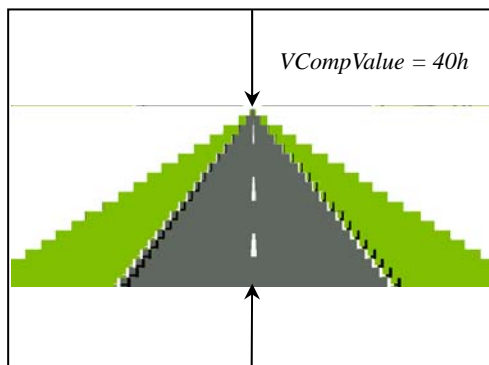
0x8801\_3000~0x8801\_37FC

	B8 - b0
0x8801_3000	HcmpValue0
0x8801_3004	HcmpValue1
0x8801_37FC	HcmpValue511

Note: When HCMP is given "high", compression for text1 is active. Each horizontal scanning line is compressed according to the scale given in 0x8801\_3000~0x8801\_37FC. "80h" means no compression; "40h" represents double compression.



Example:



### 12.9.3 Horizontal and Vertical Compression

Example:



### 12.9.4 Horizontal/Vertical Compression and Movement

Example:



## 12.10 Sprite Memory

Each sprite has its own parameters including **position**, **depth**, **palette**, **character size**, **number**, **flip**, **blending**, and **color mode**. There are 512 sprites stored in the Sprite Memory. CPU is able to access Sprite Memory directly. Every sprite contains four words to store parameters. The word1 stores the character number, word2 for storing the X position, word3 for storing the attributes, and word4 for storing Y position. The definition for each block is shown below.

Word1- Character Number

b15 – b0
Number

Word2 – X-Position

B25 – b16
PX

Word3 – Attribute

b15	b14	b13	b12	b11	b10	b9	b8
Blend	Depth		Palette Bank				
b7	b6	b5	b4	b3	b2	b1	b0
VSIZE		HSIZE		VFLIP	HFLIP	Color	

Word4 – Y-Position

B31-b26	B25 – b16
Blend_Level	PY

<b>Blend</b>	Blending effect control
<b>Depth</b>	Depth layer selection
<b>Palette</b>	Color Palette Bank of the sprite.
<b>VSIZE / HSIZE</b>	Vertical / horizontal character size.
<b>VFLIP / HFLIP</b>	Vertical / horizontal flip.
<b>Color</b>	Color Mode selection
<b>PX</b>	Horizontal position range is between the (–512 ~ 511).
<b>PY</b>	Vertical position range is between (–512 ~ 511).
<b>Number</b>	Character number. Zero represents transparent character.
<b>Blend_Level</b>	Blend Level value range is between(0~63).

## 12.11 Text Memory

Text screen consists of characters or lines. In addition, the information stored in Text Memory is called **Text Data Array**, which includes attribute array and character number array. Attribute Array stores the attribute of each character/line such as blend, flip and palette. Character number array stores the character number which is used for the addressing mode to access the character/pixel information from Pattern Memory. Number Array is always necessary and Attribute Array is only required in the case of Bitmap Mode or Attribute Array mode (RegisterMode=0).

The number of CharNumber Array Block and Attribute Array Block are dependent on the Text Screen Character Size and the Text Screen Bitmap Mode.

CIF Mode:

Character /Bitmap	Character Vsize (Pixels)	Character Hsize (Pixels)	No. CharNumber Array (Words)	No. of Attribute Array (Words)
Character	8	8	2048	1024
Character	8	16	1024	512
Character	8	32	512	256
Character	8	64	256	128
Character	16	8	1024	512
Character	16	16	512	256
Character	16	32	256	128
Character	16	64	128	64
Character	32	8	512	256
Character	32	16	256	128
Character	32	32	128	64
Character	32	64	64	32
Character	64	8	256	128
Character	64	16	128	64
Character	64	32	64	32
Character	64	64	32	16
Bitmap	--	--	256	128

VGA Mode:

Character /Bitmap	Character Vsize (Pixels)	Character Hsize (Pixels)	No. CharNumber Array (Words)	No. of Attribute Array (Words)
Character	8	8	8192	1024x4
Character	8	16	4096	512 x4
Character	8	32	2048	256 x4
Character	8	64	1024	128 x4
Character	16	8	4096	512 x4
Character	16	16	2048	256 x4
Character	16	32	1024	128 x4
Character	16	64	512	64 x4
Character	32	8	2048	256 x4
Character	32	16	1024	128 x4
Character	32	32	512	64 x4
Character	32	64	256	32 x4
Character	64	8	1024	128 x4
Character	64	16	512	64 x4
Character	64	32	256	32 x4
Character	64	64	128	16 x4
Bitmap	--	--	1024	128 x4

Location of Text Array is controlled by **Text Array Pointer Registers**. This register appoints the Text Memory Address.

0x8801\_0030 Text1 Character Number Array Pointer

D31	D30	.....	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
NumberPointer1													

0x8801\_004c Text1 Character Number Array Pointer

D31	D30	.....	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
NumberPointer1													

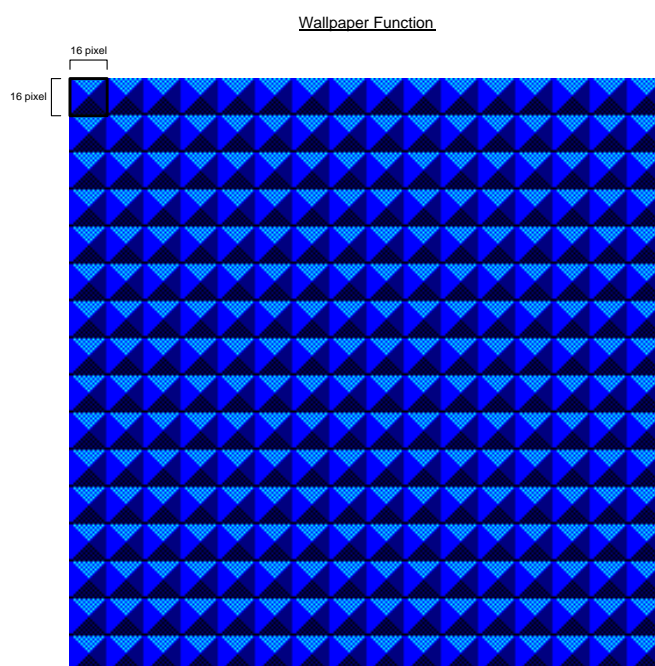
0x8801\_0068 Text1 Character Number Array Pointer

D31	D30	.....	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
NumberPointer1													

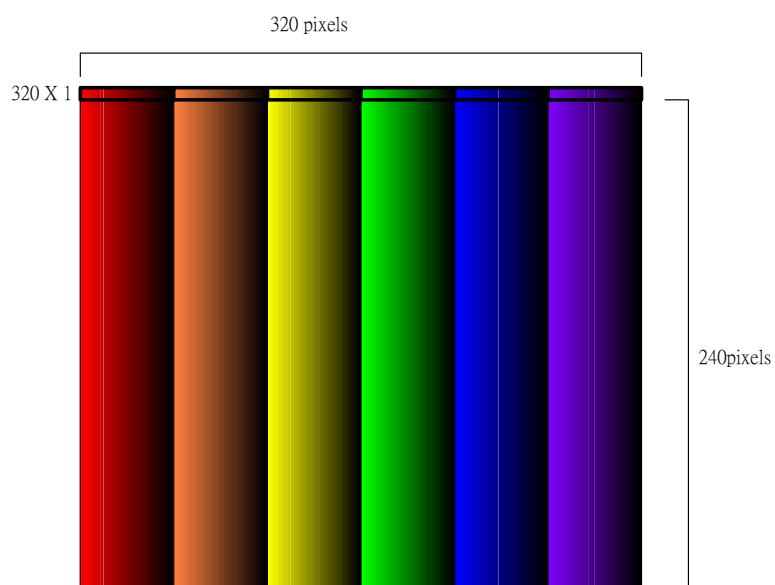
## 12.12 Wallpaper Mode

Wallpaper Mode loops single character or line (bitmap mode), as shown in the following figure. The attributes of the repeating character are the same as the first character/line defined in the Text Attribute.

### Character Mode



### Bitmap Mode



### 12.13 Blending Effect

Blending effect is used to mix three pictures as shown in the following figures. The blending effect control of the sprite is the bit15 of the sprite attribute in the sprite memory. The text blending effect is controlled by the bit8 in 0x8801\_002c and 0x8801\_0048 and 0x8801\_0064 in register mode or bit6 in the attribute array in attribute mode. When the blending effect of a sprite or text is enabled, it will mix with the pixels that with the depth lower than it.

64 blending levels can be selected by the Blend\_level at port 0x8801\_0038, 0x8801\_0054 and 0x8801\_0070. It can be selected by the Blend\_level at each sprite attribute register(0x8804\_0006 + N\*8)

### 12.14 32768/65536 High Color Mode

In SPG290 PPU, the 32768/65536 high color mode is provided to perform better image quality. This mode is controlled by the bit7/bit12 of 0x8801\_002c for text1 and 0x8801\_0048 for text2 and 0x8801\_0064 for text3. This mode only provides the bitmap mode. The programming methodology is similar to the bitmap mode.

### 12.15 Character Attributes

The following attribute parameters belong to Character for the Text Layer Character Mode and Sprites.

<b>Color Mode:</b>	4 / 16 / 64 / 256 colors per character
<b>Size:</b>	8x8 / 8x16 / 8x32 / 8x64 16x8 / 16x16 / 16x32 / 16x64 32x8 / 32x16 / 32x32 / 32x64 64x8 / 64x16 / 64x32 / 64x64 pixel size
<b>Flip:</b>	Vertical / Horizontal independent flip
<b>Palette:</b>	Palette Bank
<b>Depth:</b>	Depth control
<b>Blend:</b>	Blending control

In case of Text screen, these parameters are stored in Text memory and Register set. In case of Sprite, they are stored in Sprite memory. Some parameters are defined in plural elements, and only one parameter is effective. This rule is described as follows.

#### Text Screen

Color Mode:	Text Screen Color Mode in Register Set
Size :	Text Screen Character Size in Register Set
Flip :	if AddressMode = 0X Flip Function Off

```

        else if RegisterMode = 1
            Text Screen Flip in Register Set
        else
            Attribute in Text Memory
    Palette      : if RegisterMode = 1
        Text Screen Palette Bank in Register Set
    else
        Attribute in Text Memory
    Depth:
        Text Screen Depth in Register Set
    Blend      : if RegisterMode = 1
        Text Screen Blending control in Register Set
    else
        Attribute in Text Memory

```

### Sprite

```

    Color Mode:
        Sprite Color Mode in Sprite Memory
    Size:
        Sprite Size in Sprite Memory
    Flip:
        Sprite Flip Control in Sprite Memory
    Palette:
        Sprite Palette Bank in Sprite Memory
    Depth:
        Sprite Depth in Sprite Memory
    Blend:
        Sprite Blending control in Sprite Memory

```

## 12.16 Bitmap Mode Attributes

The following attribute parameters belong to the Text Layer Bitmap Mode.

<b>Color Mode:</b>	4 / 16 / 64 / 256 colors per character
<b>Palette:</b>	Palette Bank
<b>Depth:</b>	Depth control
<b>Blend:</b>	Blending control
<b>RGB555 Color Mode:</b>	32768 colors per pixel
<b>RGB565 Color Mode:</b>	65536 colors per pixel

In case of Text Screen, these parameters are stored in Register Set.

**Text Screen**

Color Mode:	Text Screen Color Mode in Register Set
Palette:	Text Screen Palette Bank in Register Set
Depth:	Text Screen Depth in Register Set

**12.17 Addressing Mode**

The 32-bit address pointer is required to access Pattern Data Block. The 32-bit address is sufficient to define any location in the memory map of picture processor or the CPU Memory. In any mode, the last 32-bit Address Pointer is generated through address conversion. The most suitable Addressing Mode must be chosen according to the specification.

- 1) Character Number Mode
- 2) Bitmap Address Mode

The Character Number Mode is for handling set of Characters and the Bitmap Address Mode is for the Bitmap Data.

**Picture Segment Registers** are necessary in address conversion. One Picture Segment Register consists of 32 bits, and 3 sets of Segment Register.

## 12.18 PPU Register Set

### 0x8801\_0000 PPU Control

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1-D0
P_PPU_Control				PPUEN											Resolution

**Resolution:** Select Resolution Mode

- 00: QVGA(320x240)
- 01: VGA(640x480)
- 10: HVGA(640x240)
- 11: VGA to CIF(320x240)

**PPUEN:** PPU Enable

- 0: Disable
- 1: Enable

### 0x8801\_0004 Sprite Control Registers

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_SP_Control															Coord_sel	Sp_en

**Coord\_sel:** Sprite Coordinate Select

- 0: Origin on center
- 1: Origin on top/left corner(Default)

**Spen:** Sprite Enable

- 0: Disable
- 1: Enable

### 0x8801\_0008 Sprite Max Num

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_SP_Max									Sp_Max							

**Sp\_Max:** Sprite Max available sprite numbers

Range: 0~511

### 0x8801\_000C Blend Sub

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Blend_Sub																Blend_Sub

**Blend\_Sub:** Blend formula select

- 0:  $C=A*\alpha + B*(1-\alpha)$
- 1:  $C=A*\alpha - B*(1-\alpha)$

**0x8801\_0010 Trans RGB**

Name	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Trans_RGB	TransRGB_En	TransRGB															

**TransRGB:** Transparent RGB[5:6:5]

If (Pixel\_Value == P\_Trans\_RGB)

Pixel Value is transparent color

**TransRGB\_En:** Transparent RGB enable

0: Disable

1: Enable

**0x8801\_0020 Text1 X Position**
**0x8801\_003C Text2 X Position**
**0x8801\_0058 Text3 X Position**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Tx1_X_position							X1									
P_Tx2_X_position							X2									
P_Tx3_X_position							X3									

**X:** Text Visible Area Horizontal Position

Range: -512~+511

**0x8801\_0024 Text1 Y Position**
**0x8801\_0040 Text2 Y Position**
**0x8801\_005C Text3 Y Position**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Tx1_Y_position							Y1									
P_Tx2_Y_position							Y2									
P_Tx3_Y_position							Y3									

**Y:** Text Visible Area Vertical Position

Range: -256~+255

**0x8801\_0028 Text1 Attribute**
**0x8801\_0044 Text2 Attribute**
**0x8801\_0060 Text3 Attribute**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Tx1_attribute		Depth1		Palette1				Vs1		Hs1		Flip1		Color1		
P_Tx2_attribute		Depth2		Palette2				Vs2		Hs2		Flip2		Color2		
P_Tx3_attribute		Depth3		Palette3				Vs3		Hs3		Flip3		Color3		

**Color:** Text Color Mode  
00: 2 bits / pixel (4 colors)                      01: 4 bits / pixel (16 colors)  
10: 6 bits / pixel (64 colors)                    11: 8 bits / pixel (256 colors)

**Flip:** Text Character Flip  
00: No Flip                      01: Horizontal Flip  
10: Vertical Flip              11: Vertical and Horizontal Flip

**Hs:** Text Character Horizontal Size  
00: 8 pixels                      01: 16 pixels  
10: 32 pixels                    11: 64 pixels

**Vs:** Text Character Vertical Size  
00: 8 pixels                      01: 16 pixels  
10: 32 pixels                    11: 64 pixels

**Palette:** Text Palette Select  
Range: 0~63

**Depth:** Text Depth  
00: Depth 0 (bottom)  
01: Depth 1  
02: Depth 2  
03: Depth 3 (top)

**0x8801\_002C Text1 Control**
**0x8801\_0048 Text2 Control**
**0x8801\_0064 Text3 Control**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Tx1_control				RGB565				Blend1	RGB555	VCmp1	HCmp	Mve1	Txe1	Wap1	Rgm1	Linr1
P_Tx2_control				RGB565				Blend2	RGB555	VCmp2		Mve2	Txe2	Wap2	Rgm2	Linr2
P_Tx3_control				RGB565				Blend3	RGB555	VCmp3		Mve3	Txe3	Wap3	Rgm3	Linr3

**Linr:** Text Bitmap Mode Enable  
0: Character Mode  
1: Bitmap Mode

**Rgm:** Text Register Mode Enable  
0: Character Attribute Array Effective  
1: Character Register Set Effective

**Wap:** Text Wallpaper Effect Enable  
0: Disable (Normal)  
1: Enable (Only First Character/Line attribute is effective)

- Txe:** Text Enable  
0: Disable (Non Visible)  
1: Enable (Visible)
- Mve:** Text Horizontal Movement Control Enable  
0: Disable  
1: Enable
- HCmp:** Text Horizontal extension/compression enable.  
**(Horizontal extension/compression is only available for Text1)**  
0: Disable  
1: Enable
- VCmp:** Text1 Vertical compression/extension enable  
0: Disable  
1: Enable
- RGB555:** Text 32768 Colors Mode( ARGB1555)  
0: Disable  
1: Enable
- Blend:** Text Blend control  
0: Normal mode      1: Blending mode
- RGB565:** Text 65536 Colors Mode( RGB565)  
0: Disable  
1: Enable

**0x8801\_0030** Text1 Character number array start address or bitmap line start address

**0x8801\_004C** Text2 Character number array start address or bitmap line start address

**0x8801\_0068** Text3 Character number array start address or bitmap line start address

Name	D31 – D0
P_Tx1_N_PTR	Ptrnum1[31:0]
P_Tx2_N_PTR	Ptrnum2[31:0]
P_Tx3_N_PTR	Ptrnum3[31:0]

**Ptrnum:** Text Screen Number Array Page

**0x8801\_0038 Text1 Blend Level**

**0x8801\_0054 Text2 Blend Level**

**0x8801\_0070 Text3 Blend Level**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Tx1_Blend_Level											Tx1_Blend_Level					
P_Tx2_Blend_Level											Tx2_Blend_Level					
P_Tx3_Blend_Level											Tx3_Blend_Level					

**Tx\_Blend\_Level:** Text Blend Level

Range: 0 ~ 63

**0x8801\_0074 Vertical compression/extension scale register**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_VComp_Value											VCmpValue					

**VCmpValue:** Text Screen vertical compression/extension value.

Range: 0 ~ 1024

Double size: 10h.

Unit gain: 20h.

Half size: 40h

**0x8801\_0078 Vertical compression/extension scale movement control register**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_VComp_Offset											VCmpOffset					

**VCmpOffset:** Text Screen vertical compression/extension movement value.

**0x8801\_007C Vertical compression/extension scale movement control register**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_VComp_Step											VCmpStep					

**VCmpStep:** Text Screen vertical compression/extension step value.

Range: -128 ~ +127

**0x8801\_0080 IRQ Control**

Name	D15-D3	D2	D1	D0
P_IRQ_control		HV_IRQ_EN	BLK_END_EN	BLK_ST_EN

**BLK\_ST\_EN:** VBlanking Start IRQ

0: Disable                      1: Enable

**BLK\_END\_EN:** VBlanking End IRQ

0: Disable                      1: Enable

**HV\_IRQ\_EN:** Horizontal and vertical Position hit IRQ

0: Disable                      1: Enable

**0x8801\_0084 IRQ Status**

Name	D15-D3	D2	D1	D0
P_IRQ_Status		HV_IRQ	BLK_IRQ_END	BLK_IRQ_ST

**BLK\_IRQ\_ST:** VBlanking Start IRQ status

0: None                        1: Occurred

**BLK\_IRQ\_END:** VBlanking End IRQ status

0: None                        1: Occurred

**HV\_IRQ:** Horizontal and vertical position hit IRQ status

0: None                        1: Occurred

**0x8801\_0088 Video Timing Vertical IRQ register**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_IRQTMV																

IRQTimingV

**IRQTimingV:** H/V position hit IRQ Vertical position

Range: 0 ~ 1023

**0x8801\_008C Video Timing Horizontal IRQ register**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_IRQTMH																

IRQTimingH

**IRQTimingH:** H/V position hit IRQ Horizontal position

Range: 0 ~ 1023

**0x8801\_0090 Blanking Time**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_VBLK_TIME																

VBlank\_Time

**VBlank\_Time:** Vertical blanking time extension

Range: 0~1023 lines

**0x8801\_0094 Service Line Counter**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Line_Counter							Line_Counter									

**Line\_Counter:**Service Line Counter

Range: 0~1023 lines

**0x8801\_00A0~0x8801\_00D0 Buffer Start Address**

Name	D31-D0
P_TX1_BUF_SA1	Tx1_BUF_Start_Address1[31:0]
P_TX1_BUF_SA2	Tx1_BUF_Start_Address2[31:0]
P_TX1_BUF_SA3	Tx1_BUF_Start_Address3[31:0]
P_TX2_BUF_SA1	Tx2_BUF_Start_Address1[31:0]
P_TX2_BUF_SA2	Tx2_BUF_Start_Address2[31:0]
P_TX2_BUF_SA3	Tx2_BUF_Start_Address3[31:0]
P_TX3_BUF_SA1	Tx3_BUF_Start_Address1[31:0]
P_TX3_BUF_SA2	Tx3_BUF_Start_Address2[31:0]
P_TX3_BUF_SA3	Tx3_BUF_Start_Address3[31:0]
P_Frame_BUF_SA1	Frame_BUF_Start_Address1[31:0]
P_Frame_BUF_SA2	Frame_BUF_Start_Address2[31:0]
P_Frame_BUF_SA3	Frame_BUF_Start_Address3[31:0]
P_SP_BUF_SA	Sprite_BUF_Start_Address[31:0]

**P\_Tx\_BUF\_SA1:** Character data of Text start address / Bitmap buffer 1

**\* When choosing character mode of TEXT, P\_Tx\_BUF\_SA1 will be the start address of character data in memory**

**P\_Tx\_BUF\_SA2:** Bitmap buffer 2

**P\_Tx\_BUF\_SA3:** Bitmap buffer 3

**P\_Frame\_BUF\_SA:**PPU output frame buffer 1~3 start address

**P\_SP\_BUF\_SA:** Sprite data buffer start address

**0x8801\_1000~0x8801\_17FC Texture Color Palette**

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_PPU_Color_Palet	Tr	Palette_R					Palette_G					Palette_B				

**Tr:** Transparent color control

0: Normal

1: Transparent

**Palette\_R:** Color Red component

**Palette\_G:** Color Green component

**Palette\_B:** Color Blue component

#### 0x8801\_1800~0x8801\_1FFC Sprite Color Palette

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_SP_Color_Palet	Tr	Palette_R						Palette_G				Palette_B				

**Tr:** Transparent color control

0: Normal

1: Transparent

**Palette\_R:** Color Red component

**Palette\_G:** Color Green component

**Palette\_B:** Color Blue component

#### 0x8801\_2000 ~ 0x8801\_27FC Text Horizontal movement control registers

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Tx_Hvoffset							Tx_Line_Hoffset_Value									

**Hvoffset0:** Text Screen line0 offset

**Hvoffset1:** Text Screen line1 offset

...

**Hvoffset511:** Text Screen line511 offset

#### 0x8801\_3000 ~ 0x8801\_37FC Horizontal extension/compression scale registers

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_HComp_Value							Tx_Line_HCmp_Value									

**HCmpValue0:** Text Screen line0 extension/ compression value

**HCmpValue 1:** Text Screen line1 extension/ compression value

...

**HCmpValue 511:** Text Screen line511 extension/ compression value.

Unit gain: 80h

Half size: 40h

#### 0x8801\_4000 ~ 0x8801\_4FFE Sprite Attributes

0x8801\_4000 P\_Sprite0\_Num\_X

0x8801\_4004 P\_Sprite0\_Att\_Y

0x8801\_4008 P\_Sprite1\_Num\_X

0x8801\_400C P\_Sprite1\_Att\_Y

:

:

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
------	-----	-----	-----	-----	-----	-----	----	----	----	----	----	----	----	----	----	----

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Sprite_Ch_Num	Sprite_Character_number															
Name	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
P_Sprite_X							Sprite_X									
Name	D15		D14	D13	D12-D8			D7	D6	D5	D4	D3	D2	D1	D0	
P_Sprite_Attribute	Sprite_Blend		Sprite_Depth		Sprite_Palette			Sprite_vsize		Sprite_hsize		Sprite_Flip		Sprite_Color		
Name	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
P_Sprite_Y	Sprite_Blend_Level						Sprite_Y									

<b>Sprite_Character_Number:</b>	Sprite Character Number
	Range: 0~ 65535
<b>Sprite_X:</b>	Sprite Horizontal Position
	Range: 0~ 1023
<b>Sprite_Blend:</b>	Blend control
	0: Disable      1: Enable
<b>Sprite_Paltte:</b>	Color Palette Select
	Range: 0~63
<b>Sprite_Depth:</b>	Sprite Depth
	00:      Depth 0 (bottom)
	01:      Depth 1
	02:      Depth 2
	03:      Depth 3 (top)
<b>Sprite_Vsize:</b>	Character Vertical Size
	00: 8 pixels      01: 16 pixels
	10: 32 pixels      11: 64 pixels
<b>Sprite_Hsize:</b>	Character Horizontal Size
	00: 8 pixels      01: 16 pixels
	10: 32 pixels      11: 64 pixels
<b>Sprite_Flip:</b>	Character Flip mode
	00: No Flip      01: Horizontal Flip
	10: Vertical Flip      11: Vertical and Horizontal Flip
<b>Sprite_Color :</b>	Color Mode
	00: 2 bits / pixel      01: 4 bits / pixel
	10: 6 bits / pixel      11: 8 bits / pixel
<b>Sprite_Blend_Level:</b>	Sprite blend level
	Range: 0~ 63
<b>Sprite_Y:</b>	Vertical Position
	Range: 0~ 1023

Name	Port 8801_XXXX	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_PPU_Control	0000				PPUEN											Resolution	
P_SP_Control	0004															Coord_s el	Sp_en
P_SP_Max	0008									Sp_Max							
P_Blend_Sub	000C																Blend_ Sub
P_Trans_RGB	0010	TransRGB_En[16]/Trans_RGB[15:0]															
Reserved	0014	Reserved															
Reserved	0018	Reserved															
Reserved	001C	Reserved															
P_Tx1_X_position	0020									X1							
P_Tx1_Y_position	0024									Y1							
P_Tx1_attribute	0028		Depth1		Palette1				Vs1		Hs1		Flip1		Color1		
P_Tx1_control	002C				RGB565				Blend1	RGB555	VCmp1	HCmp	Mve1	Txe1	Wap1	Rgm1	Linr1
P_Tx1_N_PTR	0030	Ptrnum1[31:0]															
Reserved	0034	Reserved															
P_Tx1_Blend_Level	0038									Tx1_Blend_Level							
P_Tx2_X_position	003C									X2							
P_Tx2_Y_position	0040									Y2							
P_Tx2_attribute	0044		Depth2		Palette2				Vs2		Hs2		Flip2		Color2		
P_Tx2_control	0048				RGB565				Blend2	RGB555	VCmp2		Mve2	Txe2	Wap2	Rgm2	Linr2
P_Tx2_N_PTR	004C	Ptrnum2[31:0]															
Reserved	0050	Reserved															

Name	Port 8801_XXXX	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
P_Tx2_Blend_Level	0054									Tx2_Blend_Level								
P_Tx3_X_position	0058								X3									
P_Tx3_Y_position	005C									Y3								
P_Tx3_attribute	0060		Depth3		Palette3					Vs3		Hs3		Flip3		Color3		
P_Tx3_control	0064				RGB565				Blend3	RGB555	VCmp3		Mve3	Txe3	Wap3	Rgm3	Linr3	
P_Tx3_N_PTR	0068	Ptrnum3[31:0]																
Reserved	006C	Reserved																
P_Tx3_Blend_Level	0070									Tx3_Blend_Level								
P_VComp_Value	0074								VCmpValue									
P_VComp_Offset	0078					VCmpOffset												
P_VComp_Step	007C									VCmpStep								
P_IRQ_control	0080													HV_IR Q_EN	BLK_E ND_EN	BLK_ST _EN		
P_IRQ_Status	0084													HV_IR Q	BLK_IR Q_END	BLK_IR Q_ST		
P_IRQTMV	0088								IRQTimingV									
P_IRQTMH	008C								IRQTimingH									
P_VBLK_TIME	0090								Vertical Blank Time (unit: lines)									
P_TX1_BUF_SA0	00A0	Tx1_BUF_Start_Address0[31:0]																
P_TX1_BUF_SA1	00A4	Tx1_BUF_Start_Address1[31:0]																
P_TX1_BUF_SA2	00A8	Tx1_BUF_Start_Address2[31:0]																
P_TX2_BUF_SA0	00AC	Tx2_BUF_Start_Address0[31:0]																

Name	Port 8801_XXXX	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_TX2_BUF_SA1	00B0	Tx2_BUF_Start_Address1[31:0]															
P_TX2_BUF_SA2	00B4	Tx2_BUF_Start_Address2[31:0]															
P_TX3_BUF_SA0	00B8	Tx3_BUF_Start_Address0[31:0]															
P_TX3_BUF_SA1	00BC	Tx3_BUF_Start_Address1[31:0]															
P_TX3_BUF_SA2	00C0	Tx3_BUF_Start_Address2[31:0]															
P_Frame_BUF_SA0	00C4	Frame_BUF_Start_Address0[31:0]															
P_Frame_BUF_SA1	00C8	Frame_BUF_Start_Address1[31:0]															
P_Frame_BUF_SA2	00CC	Frame_BUF_Start_Address1[31:0]															
P_SP_BUF_SA	00D0	Sprite_BUF_Start_Address[31:0]															
P_Color_Palet0	1000	Tr0	Palette_R_0					Palette_G_0					Palette_B_0				
P_Color_Palet1	1004	Tr0	Palette_R_1					Palette_G_1					Palette_B_1				
:	:	:	:					:					:				
P_Color_Palet510	17F8	Tr510	Palette_R_510					Palette_G_510					Palette_B_510				
P_Color_Palet511	17FC	Tr511	Palette_R_511					Palette_G_511					Palette_B_511				
P_Tx_Hvoffset0	2000							Tx_Line0_Hoffset_Value									
P_Tx_Hvoffset1	2004							Tx_Line1_Hoffset_Value									
....	....	....						....									

Name	Port 8801_XXXX	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Tx_Hvoffset510	27F8							Tx_Line510_Hoffset_Value									
P_Tx_Hvoffset511	27FC							Tx_Line511_Hoffset_Value									
Horizontal Compression Control Registers																	
P_HComp_Value0	3000							Tx_Line0_HCmp_Value									
P_HComp_Value1	3004							Tx_Line1_HCmp_Value									
:	:	:						:									
P_HComp_Value510	37F8							Tx_Line510_HCmp_Value									
P_HComp_Value511	37FC							Tx_Line511_HCmp_Value									
Sprite Control Registers																	
P_Sprite0_Ch_Num	4000	Sprite0_Character_number															
P_Sprite0_X	4002							Sprite0_X									
P_Sprite0_Attribute	4004	Sprite0_Blend	Sprite0_Depth		Sprite0_Palette				Sprite0_vsize		Sprite0_hsize		Sprite0_Flip		Sprite0_Color		
P_Sprite0_Y	4006	Sprite0_Blend_Level						Sprite0_Y									
P_Sprite1_Ch_Num	4008	Sprite1_Character_number															
P_Sprite1_X	400A							Sprite1_X									
P_Sprite1_Attribute	400C	Sprite1_Blend	Sprite1_Depth		Sprite1_Palette				Sprite1_vsize		Sprite1_hsize		Sprite1_Flip		Sprite1_Color		
P_Sprite1_Y	400E	Sprite1_Blend_Level						Sprite1_Y									
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:

Name	Port 8801_XXXX	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_Sprite511_Ch_Nu m	4FF8	Sprite511_Character_number															
P_Sprite511_X	4FFA									Sprite511_X							
P_Sprite511_Attribut e	4FFC	Sprite511_Blend	Sprite511_Depth	Sprite511_Palette						Sprite511_vsize		Sprite511_hsize		Sprite511_Flip		Sprite511_Color	
P_Sprite511_Y	4FFE	Sprite511_Blend_Level							Sprite511_Y								

## 13 DISPLAY UNIT – TV ENCODER

### 13.1 Description

The display unit, TV encoder unit in SPG290 is a multi-TV system and multi-screen mode TV encoder with 9 bits video DAC to generate composite video signal to TV screen with VGA resolution.

### 13.2 TV System and Screen Mode

The display unit can support several TV systems with different screen modes as the list shown below.

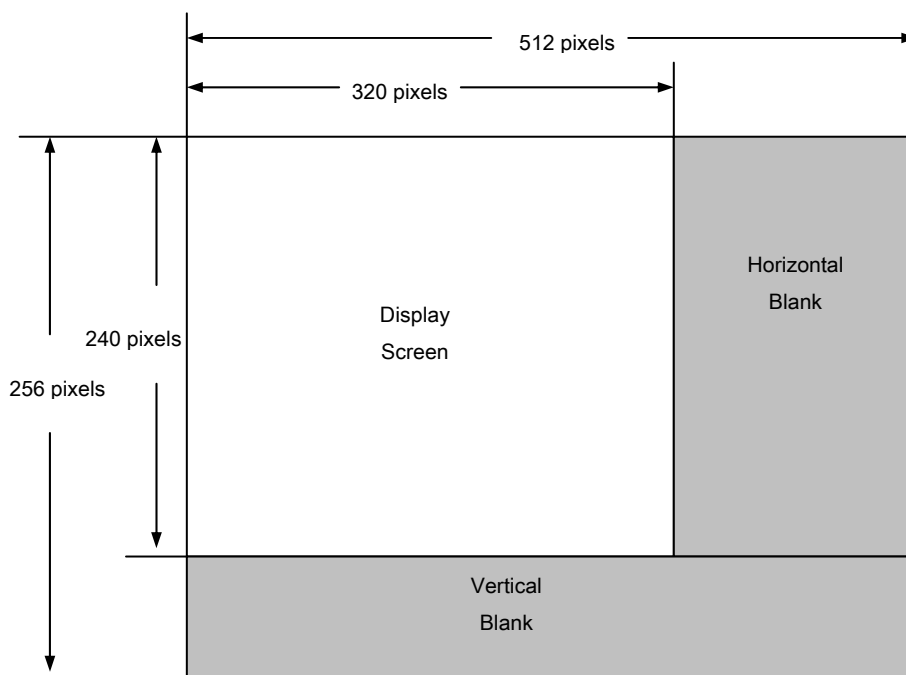
NTSC		PAL	
Interlace	Non-Interlace	Interlace	Non-Interlace
640 x 480	640 x 240	640 x 480	640 x 240
320 x 240	320 x 240	320 x 240	320 x 240
630x240	640x240	640x240	640x240

\* PAL system includes PAL-(B,D,G,H,I,N) modes

### 13.3 Text Screen Introduction

#### 13.3.1 Q-VGA Mode

The size of the entire text screen in SPG is 512x256 pixels, and the text screen for displaying (Display Screen) is 320x240 (Horizontal x Vertical) pixels. The graphic on Display Screen is shown on TV, but other parts are not. There are three text areas to be used: *Text1*, *Text2* and *Text3*. Each text can be defined in different mode (character mode/bitmap mode).



#### 13.3.2 H-VGA Mode

As the Q-VGA mode, but the size of the entire text screen in SPG is 1024x256 pixels, and

the text screen for displaying (Display Screen) is 640x240 (Horizontal x Vertical) pixels. That PPU function will extend the Vertical size to 480 pixels.

### 13.3.3 VGA Mode

As the Q-VGA mode, but the size of the entire text screen in SPG is 1024x512 pixels, and the text screen for displaying (Display Screen) is 640x480 (Horizontal x Vertical) pixels.

## 13.4 Frame Buffer

In SPG290, frame buffer can be rendered by PPU, CPU or MP4 decoder and with intelligent **Buffer Control Unit**, Frame Buffer can be allocated everywhere in the external Dynamic RAM area, such as SDRAM or DDR RAM.

## 13.5 Display

SPG290 can display arbitrary rectangle area within whole external Dynamic RAM area by setting the frame buffer start address register to the left-top pointer of the frame buffer which area is needed to be displayed ,

## 13.6 TVE Registers

SPG290 TV encoder unit provides two digital luminance filter, low pass filter, edge enhancement filter, and all pass filter to let user can depend on their content's characteristic to adjust the best TV signal output quality. For example, when displaying true color digital photos, enable low pass filter will let the image on TV screen looks more smoothly. On the contrary, the edge enhancement filter will be more suitable for the computer graphics. The all pass filter will bypass all digital filters, so user can place their own analog filter circuit on system PCB to adjust their idea TV output signal quality. In addition, the TV encoder unit can be disabled by programming to save the system power consumption whenever the TV composite output is not required, such as handheld applications.

### 13.6.1 P\_TV\_Control(0x88030000):

NAME	D12	D11	D10	D9	D8	D7-D6	D5-D4	D3-D2	D1	D0
P_TV_Control	TVEN	YCBCRMODE	RGBMODE	ENDIAN	MODE	NTSCYPE		Resolution	PAL	Intrlce

Intrlce: Interlace Mode Select  
0: Non-Interlace mode  
1: Interlace mode  
PAL: NTSC/PAL Mode Select  
0: NTSC Mode  
1: PAL Mode  
Resolution: TV Resolution Select  
0: QVGA(320x240)

1: VGA(640x480)  
 2: HVGA(640x240)  
 3: QVGA(320x240)  
 NTSCTYPE: NTSC Type Select  
 0: NTSC  
 1: NTSCJ  
 2: NTSC443  
 3: NTSC  
 MODE: Frame Buffer format Select  
 0: YCbCr(YCbCr422 or 4Y4U4Y4V)  
 1: RGB(RGB565 or ARGB1555)  
 ENDIAN: Endian Mode Select  
 0: Little Endian  
 1: Big Endian  
 RGBMODE: Frame Buffer RGB format Select  
 0: RGB565  
 1: ARGB1555  
 YCBCRMODE: Frame Buffer YCbCr format Select  
 0: YCbCr422  
 1: 4Y4U4Y4V for MPEG4  
 TVEN: TV encoder Enable  
 0: Disable  
 1: Enable

### 13.6.2 P\_TV\_Saturation(0x88030004) & P\_TV\_Hue(0x88030008):

Saturation & HUE level of TV encoder can be adjust by programming following two registers. Generally, they are initialized automatically when SPG290 boot. Be sure to leave them unchanged when running normal applications.

Name	D7-D0
<b>P_TV_Saturation</b>	Saturation
<b>P_TV_Hue</b>	Hue adjust

### 13.6.3 P\_TV\_FADE(0x8803000C): Fade Effect Control Registers.

Fade effect provides the fade-in and fade-out control. It's controlled by the register at 0x8803000C. The Luminance\_Adjust value defines the fade-out level. The following diagrams are the example of the fade effect. The right side diagram has the smaller Luminance\_Adjust value than those in the left side.

Name	D7-D0
P_TV_FADE	Lumince_Adjust



#### 13.6.4 P\_LP\_FILTER\_SEL(0x88030010): Low Pass Filter Select Control Registers.

Name	D7-D6	D5-D4	D3-D2	D1-D0
P_LP_FILTER_SEL	UPS_TYPE	UV_TYPE	YUPS_TYPE	Y_TYPE

UPS\_TYPE: U/V UPS Filter Type

UV\_TYPE: U/V Filter Type

YUPS\_TYPE: Y UPS Filter Type

Y\_TYPE: Y Filter Type

#### 13.6.5 P\_LINE\_COUNTER(0x88030020):

Line counter is synchronous with the PPU horizontal synchronous pulse (HSYNC). In NTSC system, its value is between 0 and 262. And it's between 0 and 312 in PAL system. Between the line 0 and line 239, the PPU expresses the image. In other words, the lines with the number greater than the 239 is vertical blanking period.

Name	D8-D0
P_LINE_COUNTER	Line_Counter

#### 13.6.6 Light Gun Interface

**Light Gun Interface** is mainly used for latching the luster position. The position of the luster can be read from 0x88030030 and 0x88030034(player one) 0x88030038 and 0x8803003C(player two). These two ports are for read only; write operation is invalid.

**P\_LIGHTGUN\_CTL(0x8803002C)**

**P\_LIGHTGUN0\_X(0x88030030)**

**P\_LIGHTGUN0\_Y(0x88030034)**

**P\_LIGHTGUN1\_X(0x88030038)**

**P\_LIGHTGUN1\_Y(0x8803003C)**

Name	D0
------	----

Name	D0
P_LIGHTGUN_CTL	Latch1st

Latch1st:      Latch First Pulse

0:      Latch each pulse in a frame

1:      Latch first pulse only in a frame

Name	D9-D0
P_LIGHTGUN0_X	LightGun0HPosition
P_LIGHTGUN0_Y	LightGun0VPosition
P_LIGHTGUN1_X	LightGun1HPosition
P_LIGHTGUN1_Y	LightGun1VPosition

---

---

## 14 SOUND PROCESS UNIT - SPU

---

---

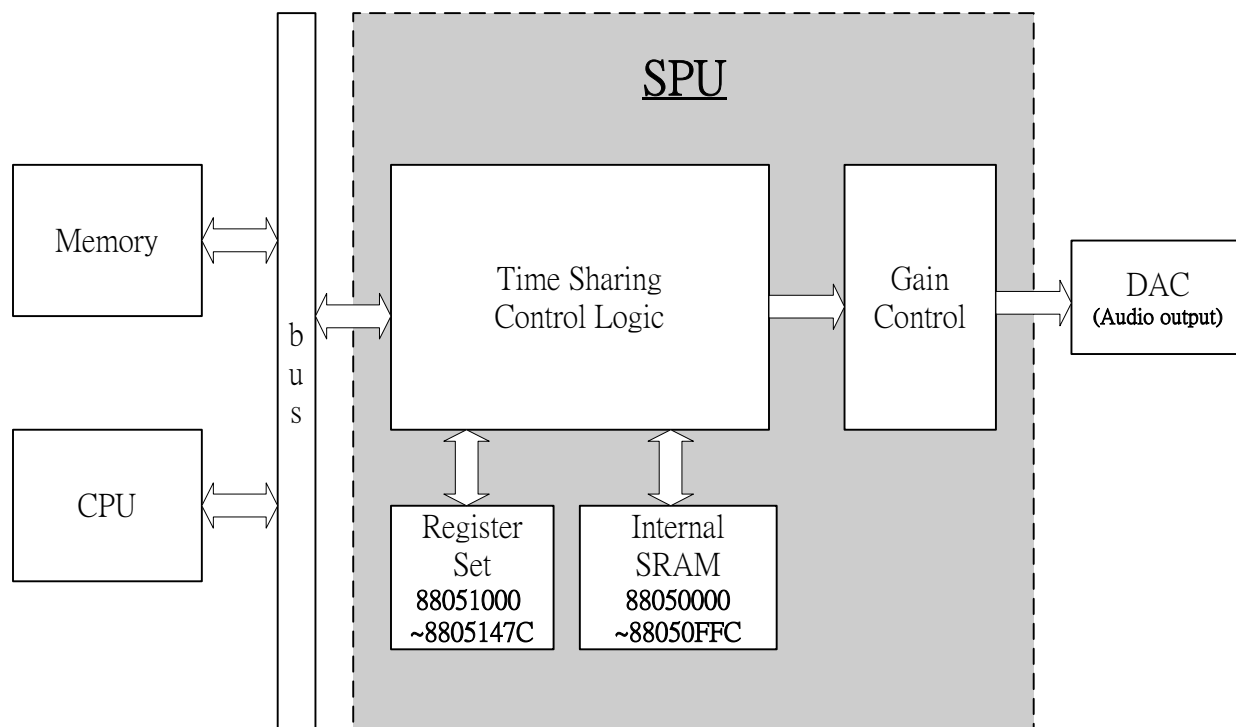
### 14.1 General Description

The Sound Processor built in the SPG290 is able to simulate all type of musical instruments by programming the tone ROM and controlling the envelope slope for each channel. Moreover, each channel can also be defined as a speech channel to provide percussion, animal sounds, gun, explosions and other sound effects in PCM format accompanied with the main music rhythm.

### 14.2 Feature

- 8-bit /16-bit (software mode) stereo PCM
- 4-bits ADPCM for each channel
- 24 channels
- Max. 4M x 16 addressing ability
- 7 bits Main volume control
- MIDI Format Gain control for each R/L channel of 24 channels
- Max. 128 piece-wise slope with repeat for envelope control
- 24-channel IRQ functions and Beat event IRQ and Envelope IRQ
- Tone-Color & envelope address not limited by bank (64K). The Tone-Color & envelope can be controlled by hardware or manual (software) mode respectively.
- Automatic tone-color interpolation function.
- Max. 281.25kHz tone-color sample rate.
- Hardware Release Tone Color function.
- Individual Channel Release function.
- Hardware Pitch Band function.
- Automatically volume control function (Compressor).
- 4-bands digital equalizer.

### 14.3 Block Diagram



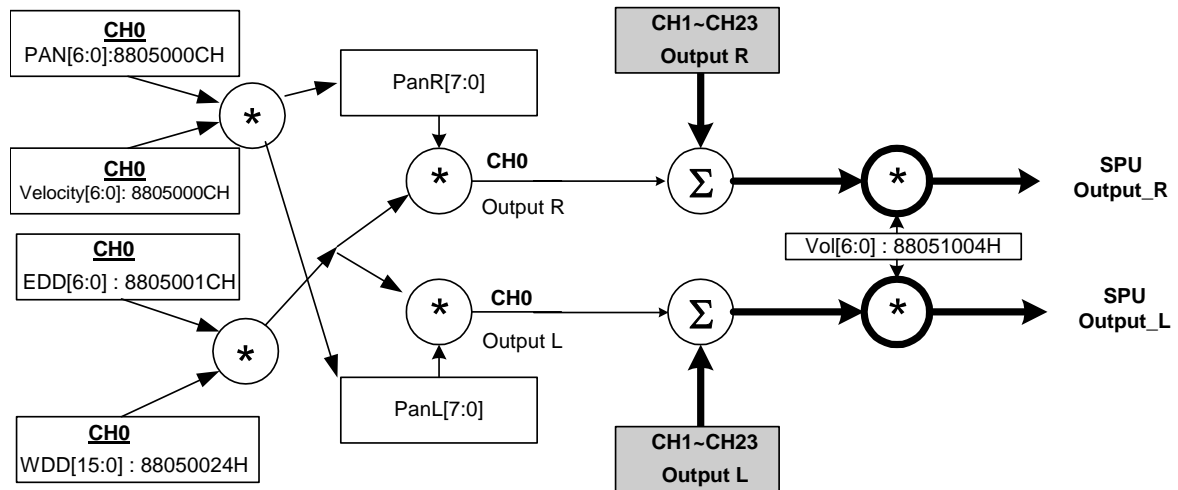
#### 14.4 Internal Memory Mapping

- Register Set: 88051000H ~ 8805107FH(Channel 0~15), 88051400H ~ 8805147FH(Channel 16~23)
- Internal SRAM: 24 channels

Channel	Phase Address Port(Hex)	Attribute Address Port(Hex)
0	88050800~8805083F	88050000~8805003F
1	88050840~8805087F	88050040~8805007F
2	88050880~880508BF	88050080~880500BF
3	880508C0~880508FF	880500C0~880500FF
4	88050900~8805093F	88050100~8805013F
5	88050940~8805097F	88050140~8805017F
6	88050980~880509BF	88050180~880501BF
7	880509C0~880509FF	880501C0~880501FF
8	88050A00~88050A3F	88050200~8805023F
9	88050A40~88050A7F	88050240~8805027F
10	88050A80~88050ABF	88050280~880502BF
11	88050AC0~88050AFF	880502C0~880502FF
12	88050B00~88050B3F	88050300~8805033F
13	88050B40~88050B7F	88050340~8805037F
14	88050B80~88050BBF	88050380~880503BF
15	88050BC0~88050BFF	880503C0~880503FF
16	88050C00~88050C3F	88050400~8805043F
17	88050C40~88050C7F	88050440~8805047F
18	88050C80~88050CBF	88050480~880504BF
19	88050CC0~88050CFF	880504C0~880504FF
20	88050D00~88050D3F	88050500~8805053F
21	88050D40~88050D7F	88050540~8805057F
22	88050D80~88050DBF	88050580~880505BF
23	88050DC0~88050DFF	880505C0~880505FF

## 14.5 Gain Control

The SPU provides 24 channels. The above structure diagram only illustrates the Channel 0.



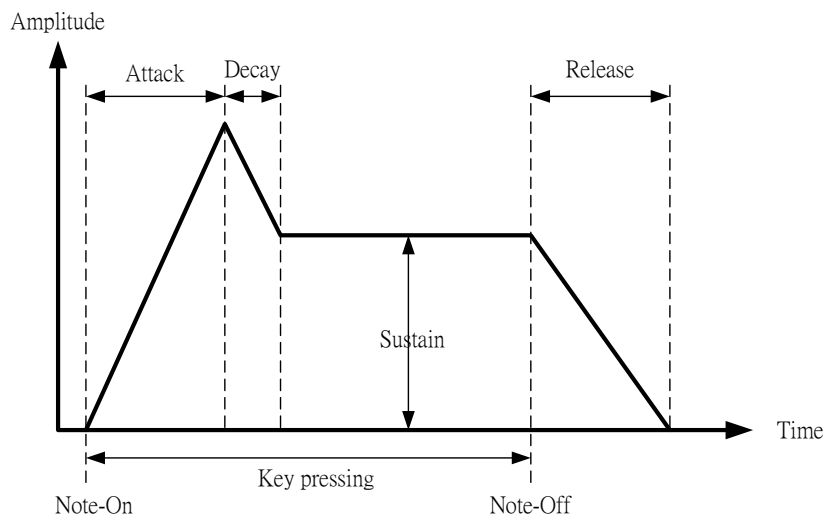
Similarities are applied for other channels. In the diagram, the multiplication result of tone-color and envelope is sent to each R/L channel to complete the panning effect. After all channels are added up, it outputs through the main volume control.

## 14.6 FUNDAMENTALS

Sound is composed of three essential elements: **pitch**, **tone-color**, and **envelope/ADSR**. The tone color can be classed to musical sound and noise in digital music.

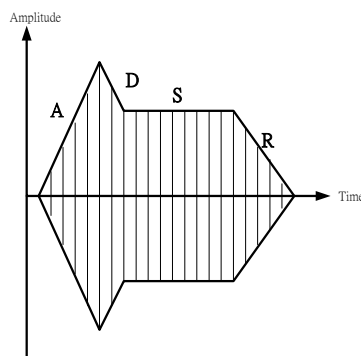
**Musical Sound:** With characteristics of cycles, vibration in certain pattern, e.g. piano, xylophone, violin, etc.

**Noise:** With the characteristic of vibration in random, e.g. animal sound.



### Musical sound

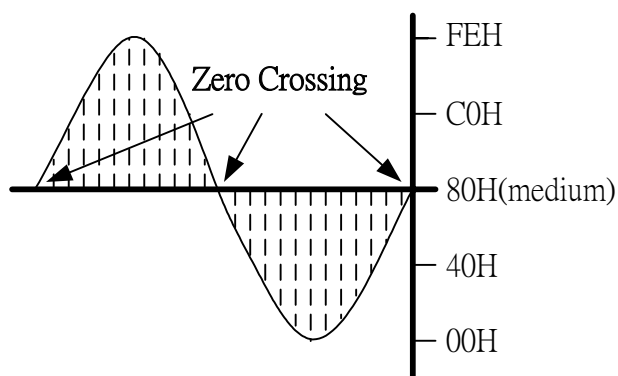
Since musical sound is a cycled wave, it is possible to be synthesized by repeating, which must be supported by hardware design. The instrument can be expressed by envelope and tone-color. The envelope can be four parts- **Attack**, **Decay**, **Sustain**, and **Release** or called **ADSR**, see the waveform above. See the waveform below for the modulations result from tone-color and envelope.



An envelope contains a series of repeated tone-color wave. The ADSR illustrated here is only a mode among all sound processing modes. Not all instruments are produced by this mode.

- 8 bits tone-color code is expressed by unsigned. The medium is "80H"; max is "FEH" and min is "00H". The "FFH" is defined as the end code of tone-color; as a result, data itself cannot be "FFH".

### Sin Wave of Tone-Color



*Note: FFH defined as the end code of tone-Color*

- The envelope data is expressed by 7 bits unsigned. The max. value is "7FH", and min. value is "00H".
- There are two envelop modes supported, envelope auto mode and manual mode.
  - Envelope Auto Mode:** The CPU writes required parameters to **register set** and **internal SRAM**. Then, the hardware loads tone-color and envelope data automatically based on the given parameters. The envelope slope of an instrument can be up to 128 types. In addition, it has the repeat function, which can be used to synthesize envelope LFO (Low Frequency Oscillation) easily.
  - Envelope Manual Mode:** The envelope data is controlled by software.

### 14.7 Control Register

Address	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Description	
88051000H	ChEn[15:0]																Channel Enable	
88051004H											Vol[6:0]						Main volume	
88051008H	ChFIQEn[15:0]																Channel FIQ Enable	
8805100CH	ChFIQSts[15:0]																Channel FIQ Status	
88051010H						BeatBaseCnt[10:0]											Beat base counter	
88051014H	BIE	BIS	BeatCnt[13:0]														Beat counter	
88051018H	EnvClk[15:0], Ch 3~0																Envelope interval select	
8805101CH	EnvClk[31:16], Ch 7~4																Envelope interval select	
88051020H	EnvClk[47:32], Ch 11~8																Envelope interval select	
88051024H	EnvClk[63:48], Ch 15~12																Envelope interval select	
88051028H	EnvRampDown[15:0]																Envelope fast ramp down	
8805102CH	ChStopSts[15:0]																Stop channel status	
88051030H	ChZeroCrossEn[15:0]																Zero cross Enable	
88051034H	Saturate		SoftCh	CompEn	NoHigh	NoInt	EQEn	VolSel	FOF		Init		MIXER_SEL	Control Flags				
88051038H	Peak	Threshold							AttScale	RelScale	DisZC	Ratio			Compressor Control			
8805103CH	ChSts[15:0]																Channel status	
88051040H	WaveInL[15:0]																Left channel mixer input	
88051044H	WaveInR[15:0]																Left channel mixer input	
88051048H	WaveOutL[15:0]																Left channel mixer output	
8805104CH	WaveOutR[15:0]																Left channel mixer output	
88051050H	ChRepeatEn[15:0]																Channel Repeat Enable control	
88051054H	ChEnvMode[15:0]																Channel Env Mode	
88051058H	ChToneRelease[15:0]																Channel Tone Release Control	
8805105CH	ChEnvIrqSts[15:0]																Channel Env Irq Status	
88051060H	ChPitchBandEn[15:0]																Channel Pitch Band Enable	
88051064H	Phase_Soft[15:0]																Software channel phase	
88051068H	AttackTime								ReleaseTime								Attack/Release Time Control	
8805106CH		CutOff1[6:0]								CutOff0[6:0]							Digital EQ Cut-Off frequency 0/1	
88051070H		CutOff2[6:0]								CutOff2[6:0]							Digital EQ Cut-Off frequency 2/3	
88051074H		Gain1[6:0]								Gain0[6:0]							Digital EQ Gain 0/1	
88051078H		Gain3[6:0]								Gain2[6:0]							Digital EQ Gain 2/3	

8805107CH	BankAddr[8:0]										Wave Table's Bank Address					
88051080H	SoftChBaseL										SoftCh Base Address [15:0]					
88051084H	SoftChBaseH										SoftCh Base Address [31:16]					
88051088H	SoftIrq	SoftIrqEn									Stereo	SOFTSIZE	SoftCh Control Register			

Note: All control registers are active high.

Address	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Description
88051400H									ChEn[23:16]								Channel Enable
88051404H																	-
88051408H									ChFIQEn[23:16]								Channel FIQ Enable
8805140CH									ChFIQSts[23:16]								Channel FIQ Status
88051410H																	-
88051414H																	-
88051418H									EnvClk[79:64], Ch 19~16								Envelope interval select
8805141CH									EnvClk[95:80], Ch 23~20								Envelope interval select
88051420H																	
88051424H																	
88051428H									EnvRampDown[23:16]								Envelope fast ramp down
8805142CH									ChStopSts[23:16]								Stop channel status
88051430H									ChZeroCrossEn[23:16]								Zero cross Enable
88051434H																	
8805143CH									ChSts[23:16]								Channel status
88051440H																	-
88051444H																	-
88051448H																	-
8805144CH																	-
88051450H									ChRepeatEn[23:16]								Channel Repeat Enable control
88051454H									ChEnvMode[23:16]								Channel Env Mode
88051458H									ChToneRelease[23:16]								Channel Tone Release Control
8805145CH									ChEnvIrqSts[23:16]								Channel Env Irq Status
88051460H									ChPitchBandEn[23:16]								Channel Pitch Band Enable

#### 88051000H; ChEn [15:0]

#### 88051400H; ChEn [23:16], Channel Enable Control and Status

1: Channel Enable

0: Channel Disable

- In H/W mode:

Before enabling a channel (ChEn [x]=1) , it is a **must** to initialize the Port[88050000 + (x-1)\*64 H] ~

Port[88050500 + x \* 64 – 1 H]. Please refer to **Programming Note** section for initialization.

- To enable a channel, check the corresponding **ChStopSts** (8805102CH) is cleared (writing “1” to ChStopSts corresponding bit) first. Write 0 to ChEn of a channel will set the ChStopSts of the channel to 1.

#### 88051004H; VOL [6:0]

Main volume control controls sound volume for entire system. It is designed that when all 24-channel is turn on with its max WDD, EDD, Velocity and global volumn, the output will not over flow.

#### 88051008H; ChFIQEn [15:0];

#### 88051408H; ChFIQEn [23:16]; Channel FIQ Enable

0: Channel FIQ disable

1: Channel FIQ enable.

Control FIQ enable. Refer to register 8805100CH.

#### 8805100CH; ChFIQSts [15:0]

#### 8805140CH; ChFIQSts [23:16]

Channel FIQ Status. ChFIQSts[x] occurs at the frequency of the sample rate of channel x, which depends on the channel phase.

Write 1: Clear FIQ

Write 0: No operation

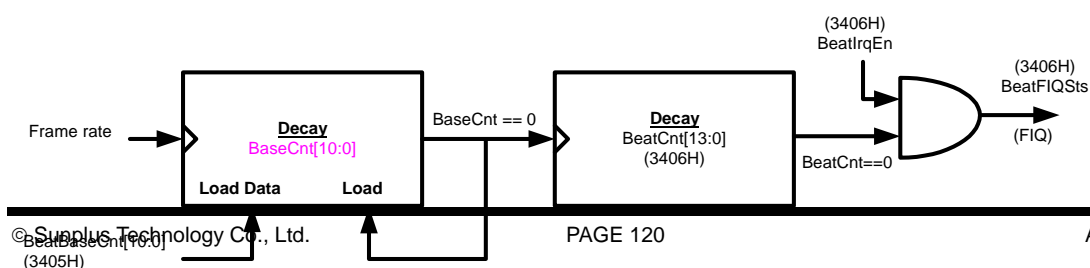
Read 1: Channel FIQ active

Read 0: Channel FIQ inactive

- FIQ occurs when the corresponding ChFIQEn[x] and ChFIQSts[x] be activated.

#### 88051010H; BeatBaseCnt [10:0]; Beat Base Counter (R/W)

b10	b9	b8	B7	b6	B5	b4	B3	b2	b1	b0
D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0



Beat count will be subtracted one every time when beat trigger count reach zero. Beat trigger count will be subtracted one every 4 frame.

#### 88051014H; Beat Counter

B15	b14	B13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
BeatIRQEn	BeatIRQSts	BeatCnt[13:0]													

BeatIRQEn **b15** (R/W): Beat IRQ Enable

0 Disable; Writing "0" to Beat(3105H) register clears the Beat FIQ.

1 Enable; When Beat event (Beat decay to 0) is enabled, FIQ activates.

BeatIRQSts **b14** (R); Current Beat IRQ (IRQ 4) event status.

0 Beat IRQ (IRQ 4) no request

1 Beat IRQ (IRQ 4) request

BeatIRQ\_Period = ( **BeatBaseCnt** \* **BeatCnt** ) \* 14.208us.

**88051018H, 8805101CH, 88051020H, 88051024H; EnvClk0[63:0];**  
**88051418H, 8805141CH; EnvClk0[95:64]; Envelope Interval selection**

Ch0 : EnvClk[3:0]

Ch1 : EnvClk[7:4]

.....

.....

Ch23: EnvClk[95:94]

- **Default:** ChxEnvClk0[1:0] = 0H
- **ChxEnvClk0[1:0] where frame = 1/frame-rate**
  - 0000: EnvClk Count once / 4 \* 4 frame
  - 0001: EnvClk Count once / 8 \* 4 frame
  - 0010: EnvClk Count once / 16 \* 4 frame
  - 0011: EnvClk Count once / 32 \* 4 frame
  - 0100: EnvClk Count once / 64 \* 4 frame
  - 0101: EnvClk Count once / 128 \* 4 frame
  - 0110: EnvClk Count once / 256 \* 4 frame
  - 0111: EnvClk Count once / 512 \* 4 frame
  - 1000: EnvClk Count once / 1024 \* 4 frame
  - 1001: EnvClk Count once / 2048 \* 4 frame
  - 1010: EnvClk Count once / 4096 \* 4 frame
  - 1011: EnvClk Count once / 8192 \* 4 frame
  - 1100: EnvClk Count once / 8192 \* 4 frame

1101: EnvClk Count once /  $8192 * 4$  frame

1110: EnvClk Count once /  $8192 * 4$  frame

1111: EnvClk Count once /  $8192 * 4$  frame

**88051028H; EnvRampDown[15:0];**

**88051428H; EnvRampDown[23:16]; For ChSts[x] control, see \$8805103CH.**

When **EnvRampDown** is set to 1, **EnvRampDownClk** will be read from Phase SRAM and Ramp down offset will be read from attribute SRAM then the envelop will ramp down to 0 in 0~3sec depends on the **EnvRampDownClk** and **RampDownOffset**.

**8805080CH, 80050C0CH; EnvRampDownClk:**

000: Envelope Ramp down one step every  $13 * 4 * 4$  frame = 0.738 ms

001: Envelope Ramp down one step every  $13 * 16 * 4$  frame = 2.955 ms

010: Envelope Ramp down one step every  $13 * 64 * 4$  frame = 11.821 ms

011: Envelope Ramp down one step every  $13 * 256 * 4$  frame = 47.284 ms

100: Envelope Ramp down one step every  $13 * 1024 * 4$  frame = 189.137 ms

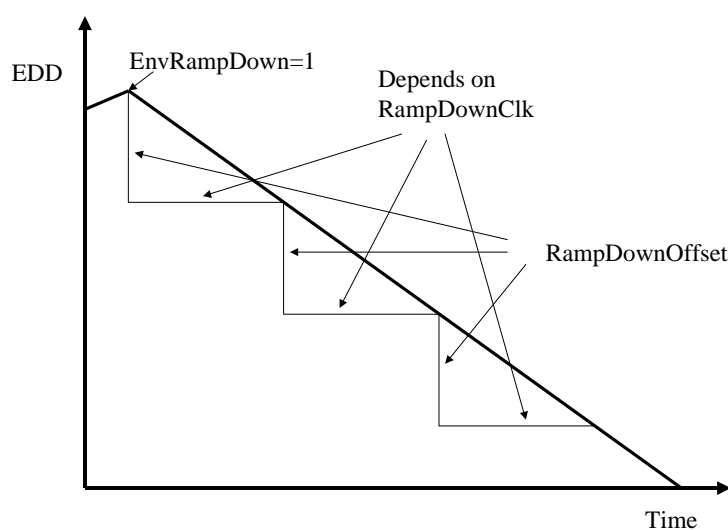
101: Envelope Ramp down one step every  $13 * 4096 * 4$  frame = 756.548 ms

110: Envelope Ramp down one step every  $13 * 8192 * 4$  frame = 1.513 s

111: Envelope Ramp down one step every  $13 * 8192 * 4$  frame = 1.513 s

**88050028H, 88050428H; RampDownOffset**

The decreasing value of EDD at each ramp down step.



**8805102CH; ChStopSts[15:0];**

**8805142CH; ChStopSts[23:16]; channel stop status**

Write 1: Clear Stop status

Write 0: No operation

Read 1: Channel is stopped

Read 0: Channel is ready

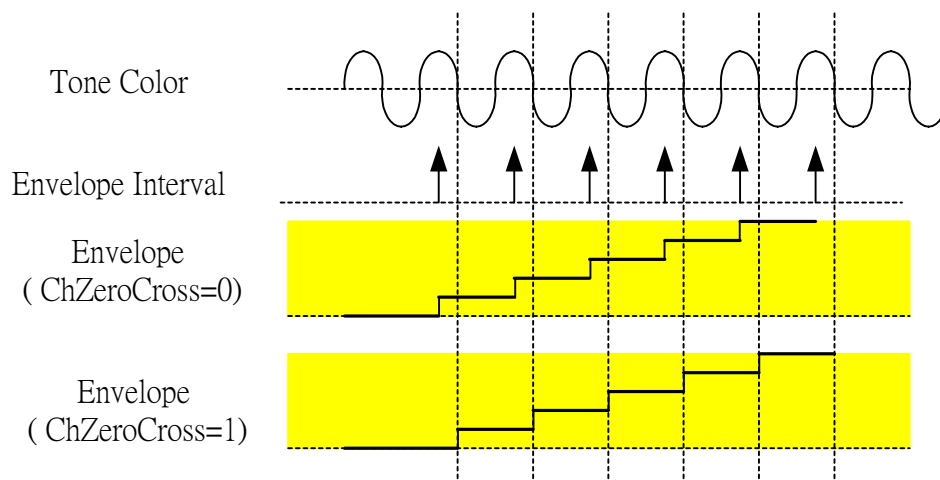
This stop bit is able to avoid channel to be ON accidentally. Before enabling (writing "1") channel [x], "1" must be written to the corresponding STOP ch[x]. Such process assures the channel is enabled successfully.

**88051030H; ChZeroCrossEn[15:0];**

**88051430H; ChZeroCrossEn[23:16]; Channel zero crossing control.**

0: Envelope data changes according to the setup in Envelope Interval selection (88051018 H –88051024H, 88051418H –8805141CH).

1: Envelope data changes only when the tone-color is at the zero-crossing point (default).



**88051034H [b3]; Init**

Sound processor channels accumulator initial control. To make sure that the system would be operated correctly, this initial command is required in the initialization of the sound processor. It's better to initialize the accumulator every time when the processor is enabled or disabled.

Write 0: no operation

Write 1: initial accumulator

**88051034H [b5]; FOF: FIFO over flow flag.**

This flag is used to indicate the SPU is unable to deal with such high sample rate and cause data lost.

This bit is read only and writing 1 can clear this bit.

**88051034H [b7~b6]; VolSel; High volume select**

When VolSel is zero, the volume of a single channel will be 1/24 of max volume in 24-ch mode, 1/16 in 16-ch mode, or 1/8 in 8-ch mode. When VolSel is set to a non-zero value, the volume of a single channel become larger according to the following table

VolSel Setting	Maximun Proportion of a Single Channel in Overall Output		
	24-ch Mode	16-ch Mode	8/6-ch Mode
00	1/24	1/16	1/8
01	1/8	1/4	1/2
10	1/2	1	2
11	1	2	4

**88051034H [b8]; EQEn; Digital Equalizer Enable bit**

When LPEn is zero, the internal digital equalizer will be disabled, and the output will be the original wave data. When LPEn is one, the internal digital equalizer will be enabled.

**88051034H [b9]; NoInt; Interpolation on/off control bit**

When NoInt is zero, the internal interpolation logic will used to smooth the output data when the phase is smaller than 0x10000, which causes the total bandwidth lower. If the FOF bit is set when program is running, set NoInt to one might help to improve the overall performance.

**88051034H [b10]; NoHigh; High Quality Interpolation on/off control bit**

When NoHigh is zero, the internal interpolation logic will used to compensate the error induced by the phase-jitter effect and this will dramatically increase the sound quality of output wave data. Write 1 to this bit will disable this feature.

**88051034H [b12]; SoftCh; Software channel enable bit**

When SoftCh is zero, the software channel will not have FIFO, i.e. the data written into D040H, D044H will be mixed with SPU's output directly. When SoftCh is one, an embedded 16-depth FIFO will be used for stored the wave data. Additionally, the setting in D044H and D064H will be used to control the output frequency of this software channel. The software channel has the capability to fired FIQ when data stored in FIFO in lower than threshold level set in D044H.

**88051034H [b1~b0]; Mixer\_Sel; Software channel Left/Right mixer selection control**

These bits will only effect when SoftCh is set to 1.

Mixer\_Sel[1] :

0: Software channel keeps silence output at Left channel.

1: Software channel output the data written in FIFO at Left channel.

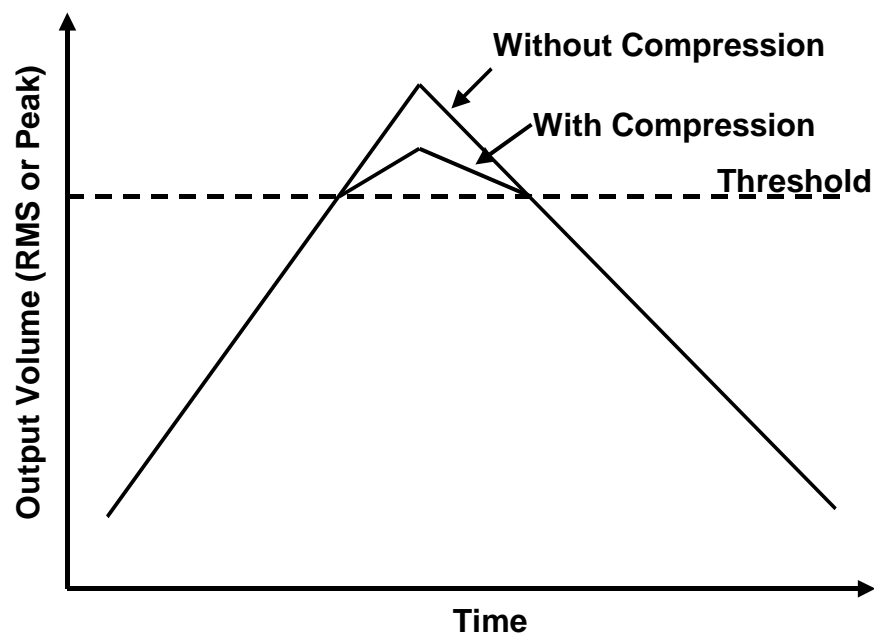
Mixer\_Sel[0] :

0: Software channel keeps silence output at Right channel.

1: Software channel output the data written in FIFO at Right channel.

#### 88051034H [b11]; CompEn; Compressor on/off control bit

When CompEn is one, the internal compressor will be activated and used the parameter stored in 88051038H and 88051064H to dynamically control the output volume. When the VolSel is set to a non-zero value, there is a possibility the output wave will saturate, the compressor can increase the over-all volume without saturate in output.



#### 88051034H [b15]; Saturate; Output signal saturation flag

This bit is used to indicate if the output signal is saturate. The saturate means the output signal is exceed the maximum range and the output will be clipped. When this bit is set to 1, it means a saturation condition is happened and it will not be cleared until the programmer writes '1' to this bit. The saturate will happen only when VolSel is set to a non-zero value. To avoid this, use the compressor properly to increase the volume without affect the sound quality.

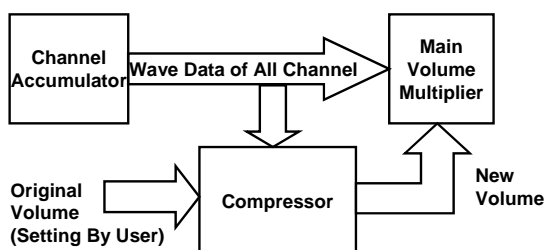
#### 88051038H [b15]; Peak; Peak/RMS Mode Selection of Compressor

When Peak is one, the compressor will detect the peak value of output wave data, when the peak value is larger than threshold value setting in Threshold register, the compressor will start to compress the output.

When Peak is zero, the compressor will detect the RMS value of output wave data. User can try both mode to determine which is better for the application.

#### 88051038H [b14~b8]; Threshold; Threshold Level of Compressor

The compressor detects the output level “before” the main volume’s multiplier, and it will control the main volume dynamically. Look the following diagram for detail.



The threshold has maximum value 7FH and minimum value 01H, 00H is not allowed. Suppose the maximum value of wave data of all channels is 1, when peak or RMS of wave data of all channels is large than (threshold/80H), the compressor will start to active.

#### 88051038H [b7~b6]; AttScale; Attack Time Scale Control Register

#### 88051068H [b15~b8]; AttackTime[7:0]; Attack Time Register

The definition of attack time is how fast the compressor response for the wave data’s peak or RMS over the threshold. The fast attack time will result in fast response to the wave data, which means the compress will start in a short time. This is good for prevent the saturate but the transient part in the wave may be lost. So fast attack time is good for melody but bad for drum. The slow attack time will result in slow response to the wave data, which means that the compress will start after a while. This is good for preserve the transient part in the wave but bad to prevent the saturation. Use can try both setting and find the best setting for the application.

The real attack time is 14-bits wide according to the following table

:

AttScale	Real Attack Time
00b	AttackTime[7:0] * 1
01b	AttackTime[7:0] * 4
10b	AttackTime[7:0] * 16
11b	AttackTime[7:0] * 64

Real attack time is 1 means the attack time is around 1.5 ms. Real attack time is 100 means the attack time is around 150 ms and so on.

#### 88051038H [b5~b4]; RelScale; Release Time Scale Control Register

#### 88051068H [b7~b0]; ReleaseTime[7:0]; Release Time Register

The definition of release time is how fast the compressor responses for the wave data's peak or RMS lower than the threshold. The fast release time will result in the pumping of breathing effect happen, but the long release time will result in the over-all volume decrease. Use can try both setting and find the best setting for the application.

The real release time is 14-bits wide according to the following table

:

RelScale	Real Release Time
00b	ReleaseTime[7:0] * 1
01b	ReleaseTime[7:0] * 4
10b	ReleaseTime[7:0] * 16
11b	ReleaseTime[7:0] * 64

Real release time is 1 means the release time is around 1.5 ms. Real release time is 100 means the release time is around 150 ms and so on.

#### 88051038H [b3]; DisZC; Disable Zero Cross Function of Compressor

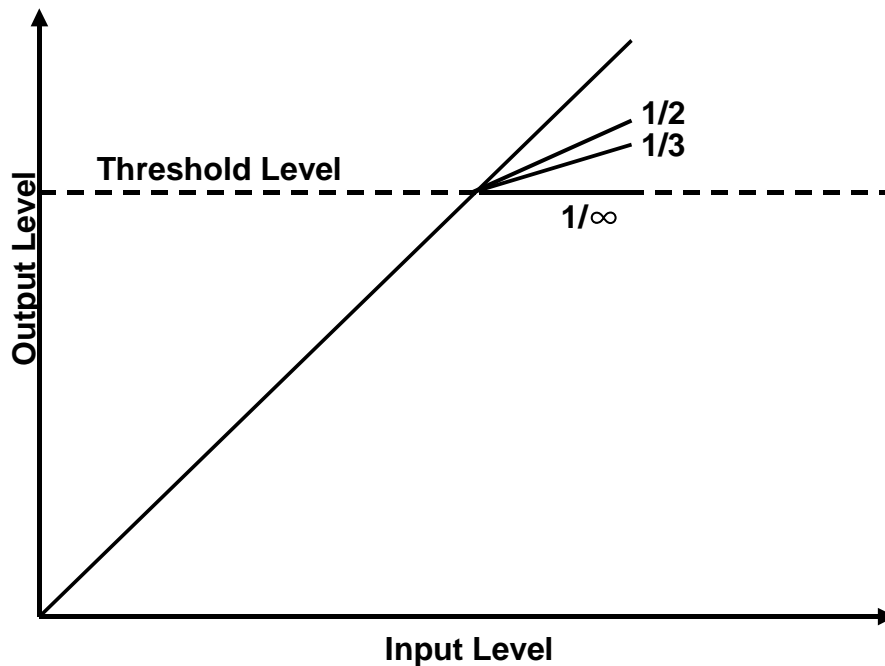
As described before, the compressor will adjust the main volume automatically. This bit is used to control if the volume change wait to the zero cross or not.

0: Volume change will wait to the zero cross happen. We suggest use this configuration for better result.

1: Volume change will not wait to the zero cross happen. Glitch sound might happen when the volume is changed.

#### 88051038H [b2~b0]; Ratio; Compress Ratio Setting

The ratio is used to control how much does it compress when the level of output wave is high than the threshold level. Look the following figure for detail.



The following table shows the relationship between the ratio setting and the real compress ratio.

Ratio Setting	Compress Ratio
0	1/2
1	1/3
2	1/4
3	1/5
4	1/6
5	1/7
6	1/8
7	$1/\infty$ (Limiter)

Since the compress ratio is highly depended on the setting of VolSel, the following table can be used for selecting the suitable ratio.

**Suggested Threshold Level and Ratio Setting under different kind of VolSel Setting.**

VolSel	Minimum Channel Number to Cause Saturation			Effect	Suggested Threshold Level	Suggested Ratio Setting
	24-ch	16-ch	8/6ch			
00 (x1)	Not possible			-	Turn-off Compressor	Turn-off Compressor
01 (x4)	8	4	2	Compress	10H (1/8)	0 (1/2)
				Limit	20H (1/4)	7 ( $1/\infty$ )
10 (x16)	2	1	0.5	Compress	04H (1/32)	0 (1/2)
				Limit	08H (1/16)	7 ( $1/\infty$ )
11 (x32)	1	0.5	0.25	Compress	02H (1/64)	0 (1/2)
				Limit	04H (1/32)	7 ( $1/\infty$ )

User who wish to use compress ratio other than 1/2 and  $1/\infty$  can use the threshold level in between these

two setting to get the best result. The larger compress ratio can result in the overall volume reduce but with less saturation possibility.

**8805103CH; ChSts [15:0];**

**8805143CH; ChSts [31:16]; Channel status**

Read 0: channel is valid

Read 1: channel is busy

**88051040H; WaveInL [15:0];**

**88051044H; WaveInR [15:0]; An additional software channel.**

The data in these two ports will be mixed with the SPU output.

**88051048H; WaveOutL[15:0];**

**8805104CH; WaveOutR[15:0]; The 16-bits output of SPU.**

The data in these two register is the final result of 24-channel + software channel when SoftCh is set to zero.

**88051064H; Phase\_Soft[15:0]; Software channel phase.**

This register is used to control the output period of software channel. The relationship between Phase\_Soft and sample rate is show as follows.

$$\text{Phase} = 27M / \text{sample-rate} - 1.$$

**88051080H; SoftChBaseL; Software channel external buffer base address [15:0]**

**88051084H; SoftChBaseH; Software channel external buffer base address [31:16]**

These registers indicated the base address of external buffer of soft channel. In SPG290, only [22:0] is valid, the high address is control by MIU.

**88051088H [b15]; SoftIrqSts; Software channel FIQ Status**

This bit will be set when the number of the rest data is less than the FIFO\_Irq\_Threshold. The SPU will assert FIQ to CPU only when SoftIrqEn is set to 1.

**88051088H [b14]; SoftIrqEn; Software channel FIQ enable bit**

0: Soft channel will not assert FIQ to CPU when the number of the rest data is less than the FIFO\_Irq\_Threshold.

1: Soft channel will not assert FIQ to CPU when the number of the rest data is less than the FIFO\_Irq\_Threshold.

**88051088H [b2]; Stereo; Software channel Stereo mode**

- 0: Soft channel will treat data in external memory as 16 bits mono data  
1: Soft channel will treat data in external memory as 16 bits stereo data

**88051088H [b1~b0]; SoftChSize; Software external memory buffer size**

- 00: 1K bytes buffer in external memory  
01: 2K bytes buffer in external memory  
10: 4K bytes buffer in external memory  
11: 8K bytes buffer in external memory

To correct initial the software channel, the following step must be followed.

1. Setup **88051080H**, **88051084H**.
2. Setup **88051064H** to determine the play speed.
3. Setup **88051088H**, this will determine the stereo mode/buffer size, enable fiq
4. Fill the buffer to full with size determined by **88051088H**
5. Setup **88051034H** to initial software channel and mixer\_sel.
6. wait fiq
7. Fill first half of the buffer.
8. wait fiq
9. Fill last half of the buffer.
10. Jump to step 6.

**88051050H; ChRepeatEn [15:0];****88051450H; ChRepeatEn [31:16]; Channel repeat enable.**

This register is used to control the repeat feature of envelope data.

- 0: Envelope repeat feature is disabled. RpEn and RpCnt is useless.  
1: Envelope repeat feature is enabled (default). RpEn and RpCnt is usefull.

**88051054H; ChEnvMode [15:0];****88051454H; ChEnvMode [31:16]; Channel envelope mode.**

This register is used to control the envelope mode of each channel. Programmer can switch a channel from auto mode to manual mode any time. If programmer want to switch a channel from manual mode to auto mode, the attribute memory of the channel need to be programmed first.

- 0: Envelope of this channel is in manual mode.  
1: Envelope of this channel is in auto mode.

**88051058H; ChToneRelease [15:0];**

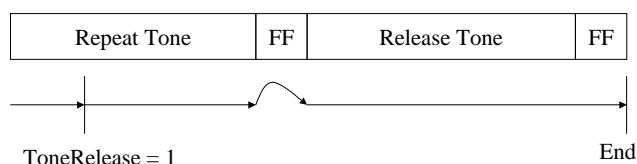
**88051458H; ChToneRelease [31:16]; Channel tone release control register.**

This register is used to control the channel tone release feature. After write 1 to this bit, the channel will complete the current tone color and then begin to play the release tone then complete.

0: Do nothing.

1: Tone Release

This bit will be cleared automatically after channel stop.



**8805105CH; ChEnvIrqSts [15:0]**

**8805145CH; ChEnvIrqSts [31:16]; Channel envelope IRQ status.**

0: No Envelope IRQ.

1: Envelope IRQ is set.

Write 1 clear this bit. Envelope IRQ will be set if **IrqEn** is 1 and **Eaoffset** match **IrqFireAddress**.

**88051060H; ChPitchBandEn [15:0]**

**88051460H; ChPitchBandEn [23:16]; Channel pitch band enable**

This register is used to control the pitch band feature of each channel.

0: Pitch band feature is disabled. TargetPhase[18:0], PhaseOffset[11:0], PhaseTimeStep[2:0], PhaseSign is discarded.

1: Pitch band feature is enabled. TargetPhase[18:0], PhaseOffset[11:0], PhaseTimeStep[2:0], PhaseSign will be used to increase/decrease phase. Please refer to **Internal SRAM** for detail.

**8805106CH; CutOff0[6:0], CutOff1[6:0]**

**88051070H; CutOff2[6:0], CutOff3[6:0]; Cut-off frequency setting of internal digital equalizer.**

These registers are used to control the cut-off frequency of internal digital equalizer. The internal digital

equalizer has 4-bands and cut-off frequency of each band can be adjusted individually.

For CutOff0/CutOff1/CutOff2: 0 = 0Hz, 7F = 17.578125Hz

For CutOff3: 0 = 0Hz, 7F = 70.3125Hz

**88051074H; Gain0[6:0], Gain1[6:0]**

**88051078H; Gain2[6:0], Gain3[6:0]; Gain setting of internal digital equalizer.**

These registers are used to control the gain of internal digital equalizer. The internal digital equalizer has 4-bands and gain of each band can be adjusted individually.

00H: -12dB

40H: 0dB

7FH: 12dB

## 14.8 Internal SRAM

Internal Attribute SRAM Channel 0~15 Format:

Address	Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
88050000H + x*64	Wave Address	Waddr[15:0]															
88050004H + x*64	Mode	ADPCM	16M	ToneMode	LoopAddr[21:16]						Waddr[21:16]						
88050008H + x*64	Loop Address	LoopAddr[15:0]															
8805000CH + x*64	Pan		Pan[6:0]									ChVolumn[6:0]					
88050010H + x*64	Envelope0	Repeat Period	EnvTarget[6:0]								EnvSig n	EnvInc[6:0]					
88050014H + x*64	Envelope Data	EnvCnt[7:0]									EDD[6:0]						
88050018H + x*64	Envelope1	RpCnt							Rpt	EnvLoad[7:0]							
8805001CH + x*64	Envelope Address	IrqFireAddress[8:0]									IrqEn	Eaddr[21:16]					
88050020H + x*64	Envelope Address	Eaddr[15:0]															
88050024H + x*64	Wave Data 0	WDD0[15:0]															
88050028H + x*64	Envelope Loop Control	RampDownoffset[6:0]								Eaoffset[8:0]							
8805002CH + x*64	Wave Data	WDD[15:0]															
88050030H + x*64	-																
88050034H + x*64	ADPCM Sel	ADPCM36	PointNumber														
88050038H + x*64	-																
8805003CH + x*64	-																

X = channel number

Internal Phase SRAM Channel 0~15 Format:

Address	Name	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
88050800H + x*64	Phase	Phase[18:0]																		
88050804H + x*64	Phase Accumulator	PhaseAcc[18:0]																		
88050808H + x*64	Target Phase	Target Phase[18:0]																		
8805080CH + x*64	Phase Control	RampDownClk			PhaseTimeStep				Sign	PhaseOffset										

X = channel number

Internal Attribute SRAM Channel 16~23 Format:

Address	Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
88050400H + x*64	Wave Address	Waddr[15:0]																
88050404H + x*64	Mode	ADPCM	16M	ToneMode	LoopAddr[21:16]						Waddr[21:16]							
88050408H + x*64	Loop Address	LoopAddr[15:0]																
8805040CH + x*64	Pan		Pan[6:0]							ChVolumn[6:0]								
88050410H + x*64	Envelope0	Repeat Period	EnvTarget[6:0]						EnvSig n	EnvInc[6:0]								
88050414H + x*64	Envelope Data	EnvCnt[7:0]							EDD[6:0]									
88050418H + x*64	Envelope1	RpCnt						Rpt	EnvLoad[7:0]									
8805041CH + x*64	Envelope Address	IrqFireAddress[8:0]								IrqEn	Eaddr[21:16]							
88050420H + x*64	Envelope Address	Eaddr[15:0]																
88050424H + x*64	Wave Data 0	WDD0[15:0]																
88050428H + x*64	Envelope Loop Control	RampDownoffset[6:0]						Eaoffset[8:0]										
8805042CH + x*64	Wave Data	WDD[15:0]																
88050430H + x*64	-																	
88050434H + x*64	ADPCM Sel	ADPCM36	PointNumber															
88050438H + x*64	-																	
8805043CH + x*64	-																	

X = channel number -16

Internal Phase SRAM Channel 16~23 Format:

Address	Name	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
88050C00H + x*64	Phase	Phase[18:0]																		
88050C04H + x*64	Phase Accumulator	PhaseAcc[18:0]																		
88050C08H + x*64	Target Phase	Target Phase[18:0]																		
88050C0CH + x*64	Phase Control	RampDownClk			PhaseTimeStep				Sign	PhaseOffset										

X = channel number -16

**88050000H, 88050004H (88050400H, 88050404H); Waddr[21:0];** Wave (Tone color) data start address.

**88050004H, 88050008H (88050404H, 88050408H); LoopAddr[21:0];** Wave data loop address.

This address is used for the tone color automatic repeat mode.

#### 88050004H

b15	b14	b13 – b12	b11 – b6	b5 – b0
ADPCM	ToneColor16	ToneMode[1:0]	---	---

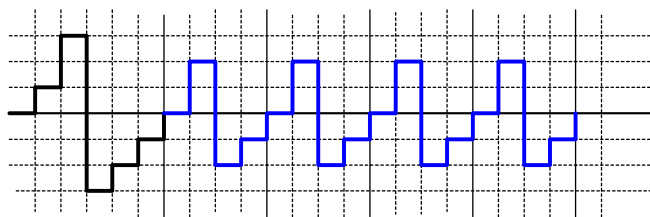
- **ADPCM:** ADPCM mode on/off
  - 0: Wave data is treated as PCM data.
  - 1: Wave data is treated as ADPCM data.

In ADPCM mode, user should set the ToneColor16 to one. Only Auto-End and Auto-Repeat mode is available in ADPCM mode. When the end code (0xFFFF) is reach, this bit will be cleared to zero automatically, i.e. back to PCM mode. This means only wave data in first region will be treated as ADPCM data, the following will be treated as PCM data. The PCM data can be 16-bits or 8-bits depends on the setting in ToneColor16. Two kinds of algorithm can be selected, please referred C034H for detail.
- **ToneColor16:** Tone color data resolution control
  - 0: 8-bits tone color data mode
  - 1: 16-bits tone color data mode

In 8-bits tone color data mode, wave data is accessed byte by byte. In 16-bits tone color data mode, data is accessed by word.
- **ToneMode[1:0]:** Tone-Color mode control
  - 00: S/W mode. Tone-Color S/W mode. In S/W mode, the tone color value is the value in WDD that is written by CPU.
  - 01: H/W auto-end mode. Tone-Color H/W output with auto end. SPU auto load tone-color and stop playing when tone-color value is end-code (FFH for 8-bits tone color data mode and FFFFH for 16-bits tone color data mode).
  - 10: H/W auto-repeat mode. Tone-Color H/W output with auto repeat. SPU reload tone-color automatically when end-code is reached. The data between the loop address and the end-code would be repeated until turn off the channel. When ADPCM36 Mode is used, this modes means the data in the loop area is in ADPCM format.
  - 11: H/W auto-repeat mode 1. Tone-Color H/W output with auto repeat. SPU reload tone-color automatically when end-code is reached. The data between the loop address and the end-code would be repeat till the off of the channel. When ADPCM36 Mode is used, this modes means the data in the loop area is in PCM format. The mode of PCM (8 or 16) is selected by the ToneColor16 bit.



Address	Data
WAddr + 0	9080H
WAddr + 1	50B0H
WAddr + 2	7060H
WAddr + 3	A080H
WAddr + 4	7060H
WAddr + 5	FFFFH



In ADPCM36 mode, when the ToneMode is set to 10b, the data format is in the following diagram:

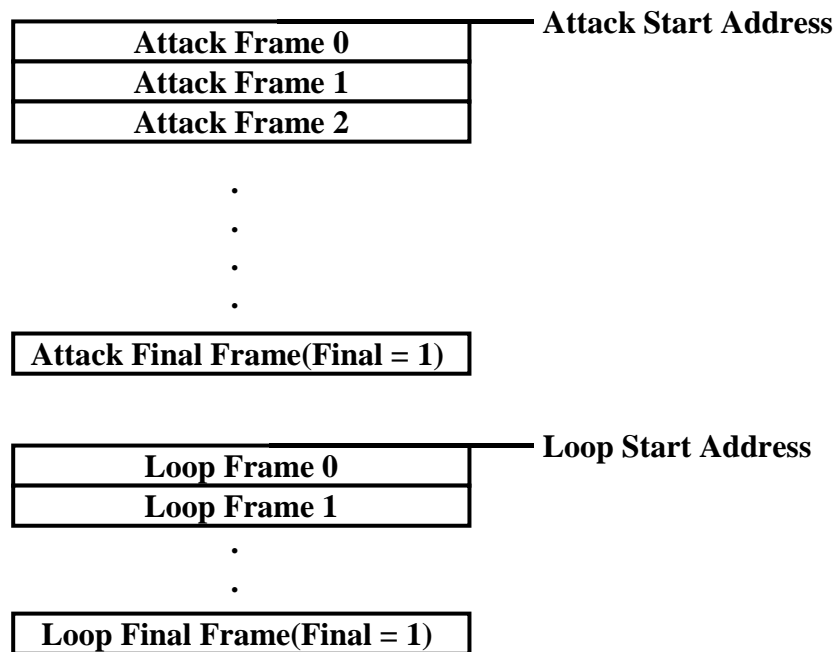


Figure. ToneMode = 10b, ADPCM36 => ADPCM36 mode

In ADPCM36 mode, when the ToneMode is set to 11b, an addition end-code (0xFFFF) must be added to the end of the attack data, the data format is in the following diagram:

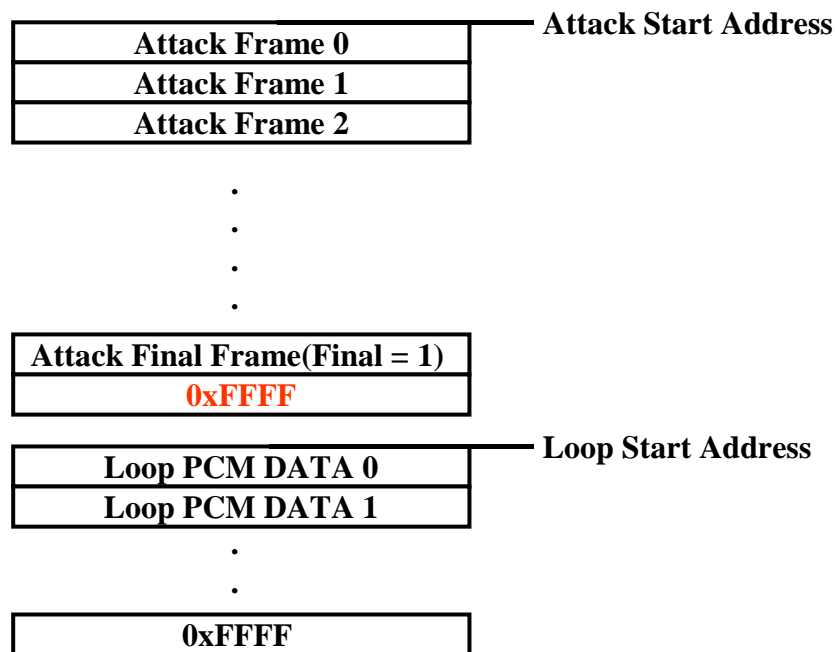


Figure. ToneMode = 11b, ADPCM36 => PCM mode

**8805000CH, 8805040CH; Pan[6:0]:** Ratio control for right and left channel.

**8805000CH, 8805040CH; Velocity[6:0]:** Velocity control for right and left channel.

When Pan < 64

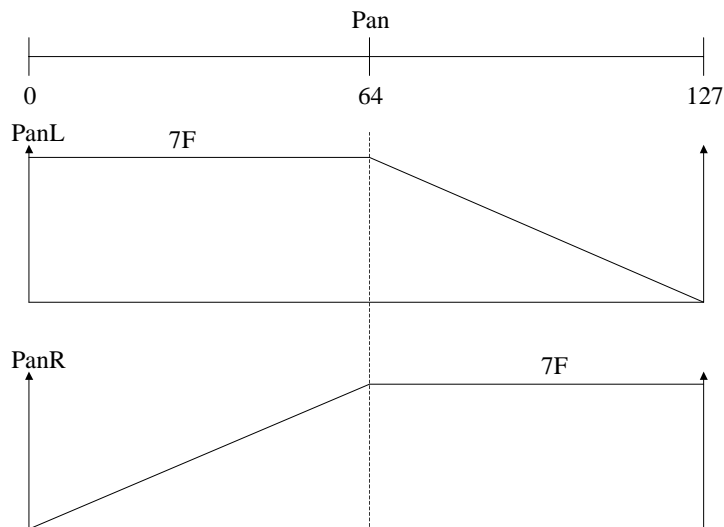
PanL = 0x7F \* ChVolumn

PanR = Pan \* 2 \* ChVolumn

When Pan >= 64

PanL = (127-Pan) \* 2 \* ChVolumn

PanR = 0x7F \* ChVolumn



#### 8805010H; Envelope 0 for envelope tracking

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	B2	b1	b0
---	EnvTargetValue [6:0]							Envsign	EnvInc [6:0]						

- **EnvTargetValue[6:0]:** Envelop Target Value for envelop tracking.

When envelop data EDD reaches EnvTargetValue [6:0], processor automatically update the envelope data from exteranl momary.

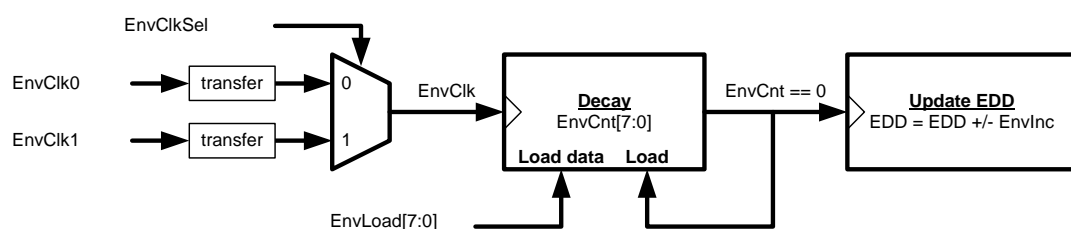
- **EnvSign: envelop sign bit**

0: envelop increment direction is positive.

1: envelop increment direction is negative.

- **EnvInc[6:0]:** envelop increment value for Envelop Tracking.

$EDD = EDD \pm EnvInc[6:0]$  , depend on **EnvSign(0/1)** bit.



Note : "transfer" is EnvClkx mapping to EnvClk. Refer 3106H~3109H register description

#### 80050014H; Envelope Data

B15	b14	b13	b12	B11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EnvCnt[7:0]								---	EDD [6:0]						

- **EnvCnt[7:0]: Envelop Count**

The count is controlled by hardware in Envelope automatic mode. The clock source for envelop counter is

from EnvClk. It records the period for counting envelope data (EDD) once.

- **EDD[6:0]**

If envelope data reaches “0”, the corresponding channel will be stopped automatically.

### 80050018H; Envelope 1 for envelope tracking

b15	b14	b13	b12	b11	B10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RpCnt[6:0]							Repeat	EnvLoad[7:0]							

- **Repeat** : Envelope repeat control

0: Envelop normal mode

1: Envelop repeat mode

In repeat mode, an extra word (**EnvelopLoopControl** in 88050028H) would be accessed from the external memory as shown in the following example. The repeat address (EAoffset) is stored in that word. The envelope would be repeated between the Eaoffset and repeat point RpCnt times.

- **EnvLoad [7:0]**: EnvLoad[7:0] (88050018H) is the value loaded into EnvCnt [7:0] (88050014H). After EnvCnt reaches “0”, the EnvLoad is re-loaded automatically.

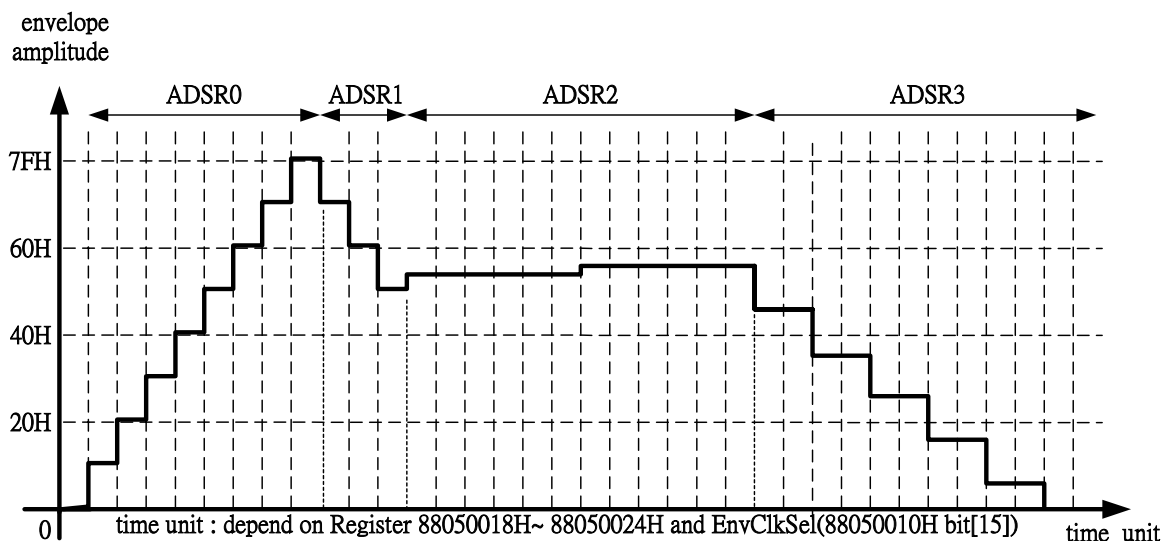
Initially, sound processor auto-loads EnvLoad[7:0] to internal register, **EnvCnt [7:0]**. The processor, according to the EnvClk0[1:0] definition or EnvClk1[1:0] in EnvClkSel, decays EnvCnt. When EnvCnt is decayed to “0”, the processor will calculate the next EDD (envelop) value.  $EDD = EDD \pm \text{EnvInc}[6:0]$ , depend on **EnvSign(0/1)** bit. If EDD value reaches EnvTargetValue[6:0], the processor automatically update the data in envelope 0/1 (Port 88050010H / 88050018H).

- **RpCnt[6:0]**: Envelop repeat counter

The registers of 88050010H, 88050014H and 88050018H are used to control the envelope. In Envelope automatic mode (88050004H), these registers are automatically reload/update by hardware from the external memory. In Envelope manual mode, CPU has to update the envelope data (EDD in 88050014H). To control the envelope in automatic mode, the envelope is controlled by Envelope0 (88050010H) and Envelope1 (88050018H) that is stored in external memory.

- Example for Envelop Format without Envelop Repeat

N <sup>th</sup> ADSR	Memory Offset[8:0] address	Envelop0/1 (88050010H/88050018H)
0	0	Envelope 0
	1	Envelope 1
1	2	Envelope 0
	3	Envelope 1
2	4	Envelope 0
	5	Envelope 1
...	...	...
	...	...



Suppose the waveform of envelop is shown as above. The start address of the envelope is as follows:

N <sup>th</sup> ADSR	Memory Offset[8:0] address	Envelop0/1 ( 8805010H/8805018H)
0	0	7F10H
	1	0000H
1	2	5090H
	3	0000H
2	4	5804H
	5	0005H
3	6	0090H
	7	0001H

1. In ADSR0, it is increased from 0 to 7FH by 8 time units. Therefore, the following values are obtained:  
 $\text{EnvTargetValue} = 7\text{FH}$ ,  $\text{EnvSign} = 0$ ,  $\text{EnvInc} = (7\text{FH} - 0) / 8 = 10\text{H}$ ,  $\text{EnvCnt} = 0$  (EDD counts each time unit).
2. In ADSR1, it is decreased from 7FH to 50H by 3 time units. Therefore, the following values are obtained:  
 $\text{EnvTargetValue} = 50\text{H}$ ,  $\text{EnvSign} = 1$ ,  $\text{EnvInc} = (7\text{FH} - 50\text{H}) / 3 = 10\text{H}$ ,  $\text{EnvCnt} = 0$  (EDD counts each time unit).
3. In ADSR2, it is increased from 50H to 58H by 12 time units. Therefore, the following values are obtained:  
 $\text{EnvTargetValue} = 58\text{H}$ ,  $\text{EnvSign} = 0$ ,  $\text{EnvInc} = (58\text{H} - 50\text{H}) / 2 = 04\text{H}$ ,  $\text{EnvCnt} = 5$  (EDD counts each 6 time units).
4. In ADSR3, it is decreased from 58H to 00H by 12 time units. Therefore,  $\text{EnvTargetValue} = 00\text{H}$ ,  $\text{EnvSign} = 1$ ,  
 $\text{EnvInc} = (58\text{H} - 00\text{H}) / 6 = 10\text{H}$ ,  $\text{EnvCnt} = 1$  (EDD counts each 2 time units).

**Note:** For time unit definition, refer to **Envelope Interval selection**.

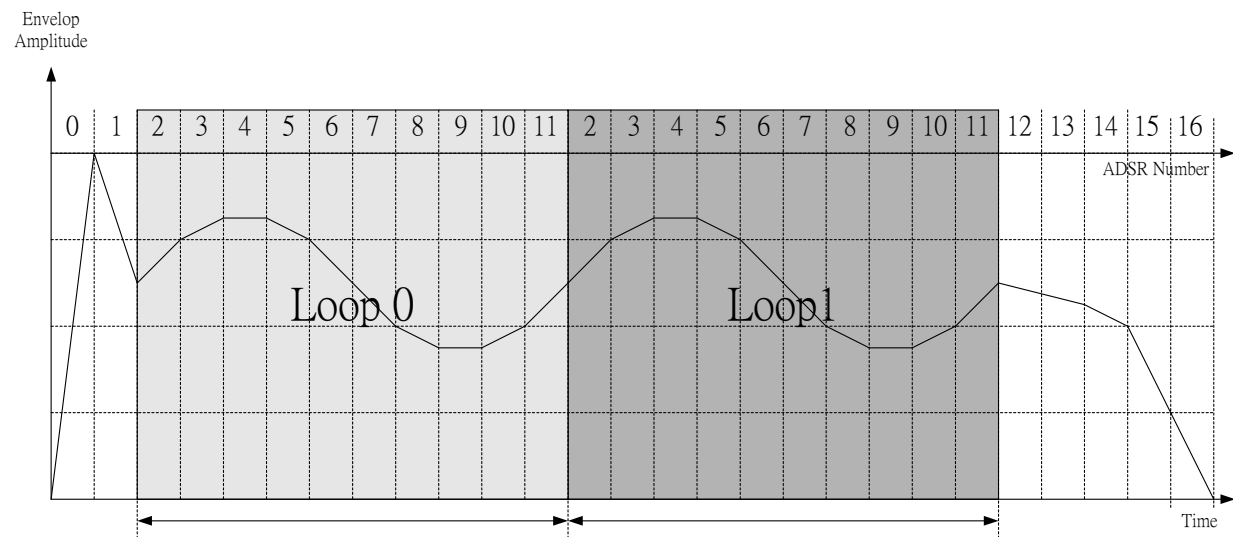
● Example for Envelop Format with Envelop Repeat

N <sup>th</sup> ADSR	Memory Offset[8:0] address	Envelop0/1 ( 8805010H/8805018H)
0	0	Envelope 0
	1	Envelope 1
1	2	Envelope 0
	3	Envelope 1
2	4	Envelope 0
	5	Envelope 1

3	6	Envelope 0
	7	Envelope 1
4	8	Envelope 0
	9	Envelope 1
5	A	Envelope 0
	B	Envelope 1
6	C	Envelope 0
	D	Envelope 1
7	E	Envelope 0
	F	Envelope 1
8	10	Envelope 0
	11	Envelope 1
9	12	Envelope 0
	13	Envelope 1
10	14	Envelope 0
	15	Envelope 1
11	16	Envelope 0
	17	Envelope 1 (*1)
	18	Envelop Loop Control (*2)
12	19	Envelope 0
	1A	Envelope 1
...	...	...
	...	...

\*1: Repeat = 1.

\*2: In this example, EOffset[8:0] = 04H, RpCnt[6:0]=01H.



#### 8805001CH, 88050020H (8805041CH, 88050420H); Envelope Base Address

\$8805001CH						\$88050020H															
b5	b4	b3	b2	b1	b0	b15	B14	B13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EAddr [21:0]																					

When envelope is set to auto mode, the H/W will use address in this register to get envelope data from external memory.

**8805001CH (8805041CH); Envelope Irq Fire Address**

b15	b14	b13	b12	b11	B10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IrqFireAddress[8:0]									IrqEn	EAddr[21:16]					

- **IrqFireAddress[8:0]:** When Eaoffset match the IrqFireAddress and IrqEn is set to 1, the envelope IRQ(IRQ 4) will fired and EnvIrqSts[x] will be set.
- **IrqEn:** Envelope IRQ enable bit.

**88050024H (88050424H); WDD1[15:0]: Previous Tone-color data value (wave data)**

B15	b14	B13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WDD1[15:0]															

Programmer must write 0x8000 to this register when using the ADPCM36 mode; otherwise this register can be ignored.

**88050028H (88050428H); Envelope Loop Control**

b15	b14	b13	b12	b11	B10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RampDownOffset[6:0]								EAoffset[8:0]							

- **RampDownOffset[6:0]:** Envelop ramp down offset, this value is used to decrease the EDD when EnvRampDown of a channel is set. A read from external memory because of envelope repeat will not affect this register.
- **EAoffset[8:0]:** . Envelop repeat address offset.

**8805002CH (8805042CH); WDD[15:0]: Tone-color data value (wave data)**

B15	b14	B13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WDD[15:0]															

In tone color automatic modes (88050004H), WDD is fetched automatically by hardware. In software mode, WDD is written by CPU. When 8-bits tone color mode(88050004H, 88050404H) is set, the wave data should be placed at **WDD[15:8]**. And in 16-bits tone color mode the tone color data is located in WDD[15:0].

**88050034H; ADPCM Sel: ADPCM Mode Select**

b15	b14	b13	b12	b11	B10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADPCM36	PointNum[4:0]						Reserved								

- **ADPCM36:** When ADPCM bit in 30x1H is set to 1, use this bit to select between two kinds of algorithm.  
0: Original algorithm in SPG2xx series, which has data rate around 32kbytes/sec.  
1: New algorithm with higher quality, which data rate around 36kbytes/sec.
- **PointNum[4:0]:** The point numbers of first frame in ADPCM36 mode, this field must be filled with the correct value before play a ADPCM36 tone colors. This field must be filled with 0 when the tonecolor is not in the ADPCM36 mode.  
0: The first frame of ADPCM36 tone colors has 1 point.

- 1: The first frame of ADPCM36 tone colors has 2 point.
- 2: The first frame of ADPCM36 tone colors has 3 point.
- 3: The first frame of ADPCM36 tone colors has 4 point.
  
- 31: The first frame of ADPCM36 tone colors has 32 point.

**88050800H (88050C00H); Phase [18:0]; control tone-color pitch (similar to sound pitch)**

B2	b1	b0	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Phase[18:0]																		

Phase = sample-rate \*  $2^{19}$  / 281.25 kHz.

Although frame rate is not the same in 8- 16- 24-Ch mode, above formula is correct no matter what channel mode you use, Phase will be divided by 2 in 16-Ch mode and divided by 4 in 8-Ch mode automatically.

**88050804H (88050C04H); Phase Accumulator [18:0]; Tone-color pitch accumulator**

B2	b1	b0	b15	b14	b13	b12	b11	b10	b9	B8	b7	b6	b5	b4	b3	b2	b1	b0
PhaseAcc[18:0]																		

To have the lowest latency between the channel enable(ChEn[x]), and the play of the first tone color, it's suggested to fit the PhaseAcc[18:0] as the 7FFFFH.

**88050808H (88050C08H); TargetPhase [18:0]; Pitch Band Target Phase**

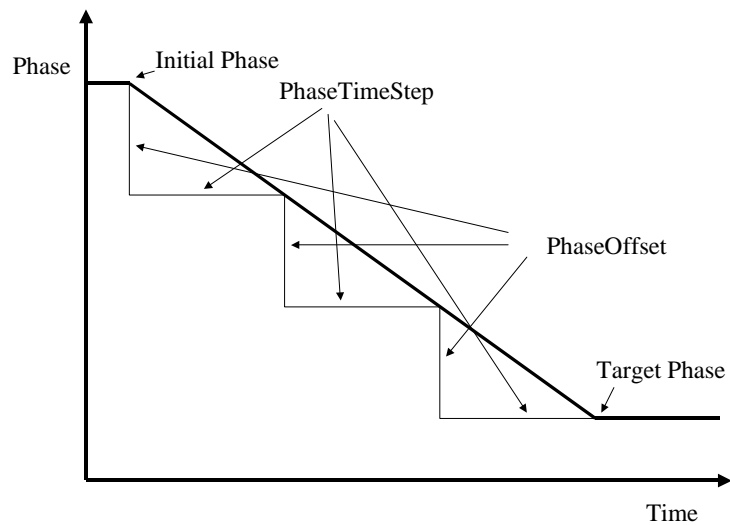
B2	b1	b0	b15	b14	b13	b12	b11	b10	b9	B8	B7	b6	b5	b4	b3	b2	b1	b0
TargetPhase[18:0]																		

The pitch band target phase, if PitchBandEn[x] is 1, the channel will use PhaseOffset[11:0] and PhaseSign and PhaseTimeStep to increase/decrease Phase to the TargetPhase.

**8805080CH (88050C0CH); Pitch Band Control and Ramp Down Control**

B15	b14	B13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PhaseTimeStep			Sign	PhaseOffset											

- PhaseOffset: This is the phase offset each time PhaseTimeStep is reached.
- PhaseSign: This is the increase decrease selection of phase.  
0: Increase phase.  
1: Decrease phase.
- PhaseTimeStep : This register is used to control the period of phase change.  
000: Phase Change Every  $8 * 4$  frame = 0.114ms  
001: Phase Change Every  $16 * 4$  frame = 0.227ms  
010: Phase Change Every  $32 * 4$  frame = 0.455ms  
011: Phase Change Every  $64 * 4$  frame = 0.909ms  
100: Phase Change Every  $128 * 4$  frame = 1.819ms  
101: Phase Change Every  $256 * 4$  frame = 3.637ms  
110: Phase Change Every  $512 * 4$  frame = 7.274ms  
111: Phase Change Every  $1024 * 4$  frame = 14.549ms



#### Programming Notes

- While initializing SPU SRAM, data must be written into Attribute SRAM and Phase SRAM. The Port[88050014H] and Port[88050028H] should be set to zero. For 8 bits tone color application, Port[88050024H] = 8000H (middle level), and Port[8805002CH] = 8080H or the first tone-color. For 16 bits tone color application, Port[8805002CH] = 8000H (middle level), and Port[8805002CH] = 8000H or the first tone-color. Besides, "1" should be written into Port[88050434H].b3 to set up the related SRAM initial value.
- Note: When update data during beat event, make sure the corresponding bit is "0" in (ChSts).

**Otherwise, hardware control is unpredictable.**

There are three methods to disable ChSts[x] :

1. Disable ChEn[x] and the channel would be disable in the next tone-color zero-crossing.
2. S/W writes EnvRampDown[x]=1. H/W will ramp down in speed defined in EnvRampDownClk[2:0] and offset in EnvRampDownOffset[6:0] until envelope is equal to zero
- 3: When the envelop data EDD(88050014H) is 0.
4. When the tone color reaches the end point(FFH) in the H/W auto-end mode.

- Frame Rate**

Channel max sample rate is 281.25kHz and the frame time is 3.55 us

**8805080CH (88050C0CH); Envelope Ramp Down Clock Selection Control**

														b18	b17	b16
---														EnvRampDownClk		

This register is used to control the ramp down period when EnvRampDown[x] is set to 1.

000: Envelope Ramp down one step every  $13 * 4 * 4$  frame = 0.738 ms

001: Envelope Ramp down one step every  $13 * 16 * 4$  frame = 2.955 ms

010: Envelope Ramp down one step every  $13 * 64 * 4$  frame = 11.821 ms

011: Envelope Ramp down one step every  $13 * 256 * 4$  frame = 47.284 ms

100: Envelope Ramp down one step every  $13 * 1024 * 4$  frame = 189.137 ms

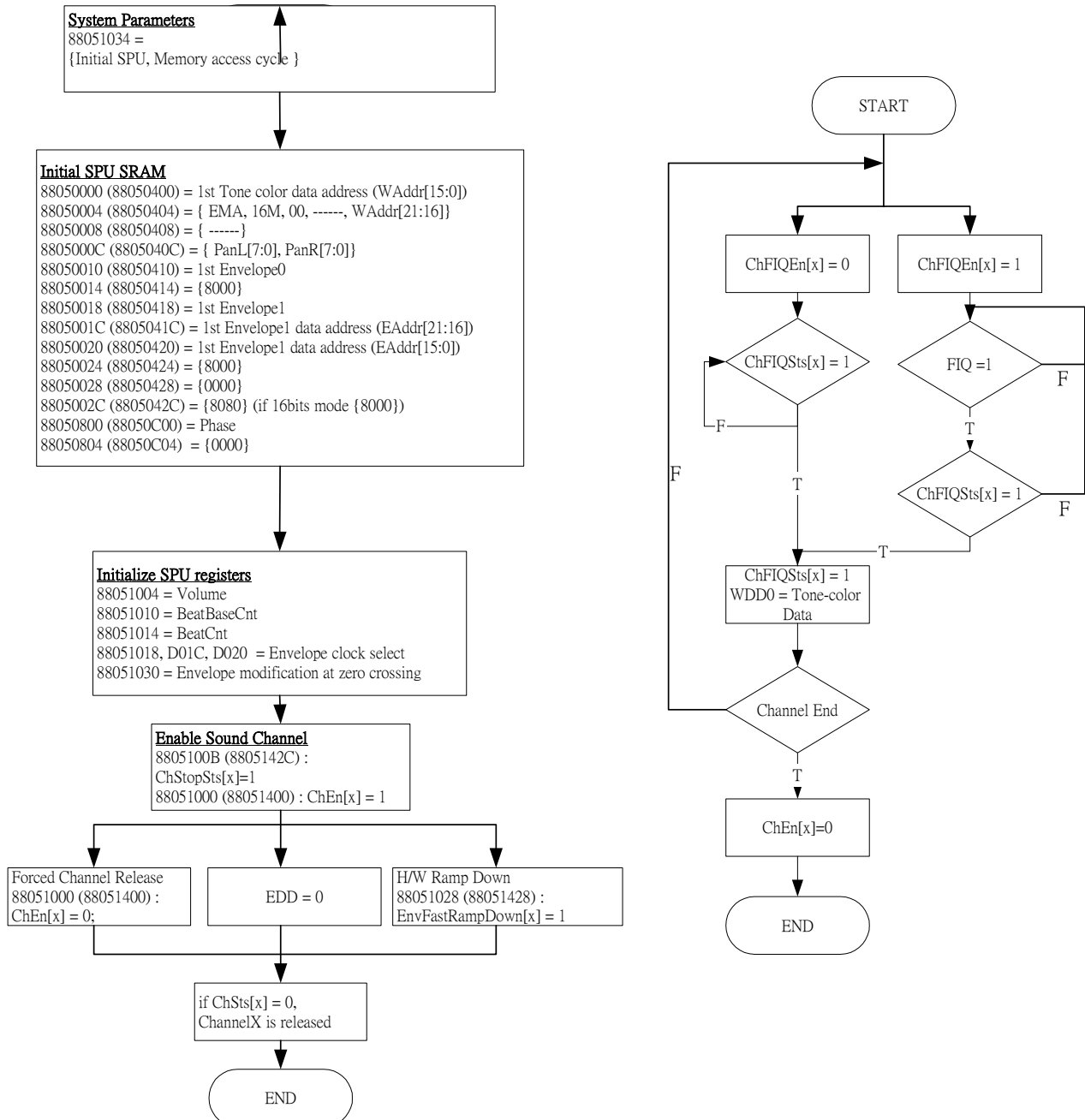
101: Envelope Ramp down one step every  $13 * 4096 * 4$  frame = 756.548 ms

110: Envelope Ramp down one step every  $13 * 8192 * 4$  frame = 1.513 s

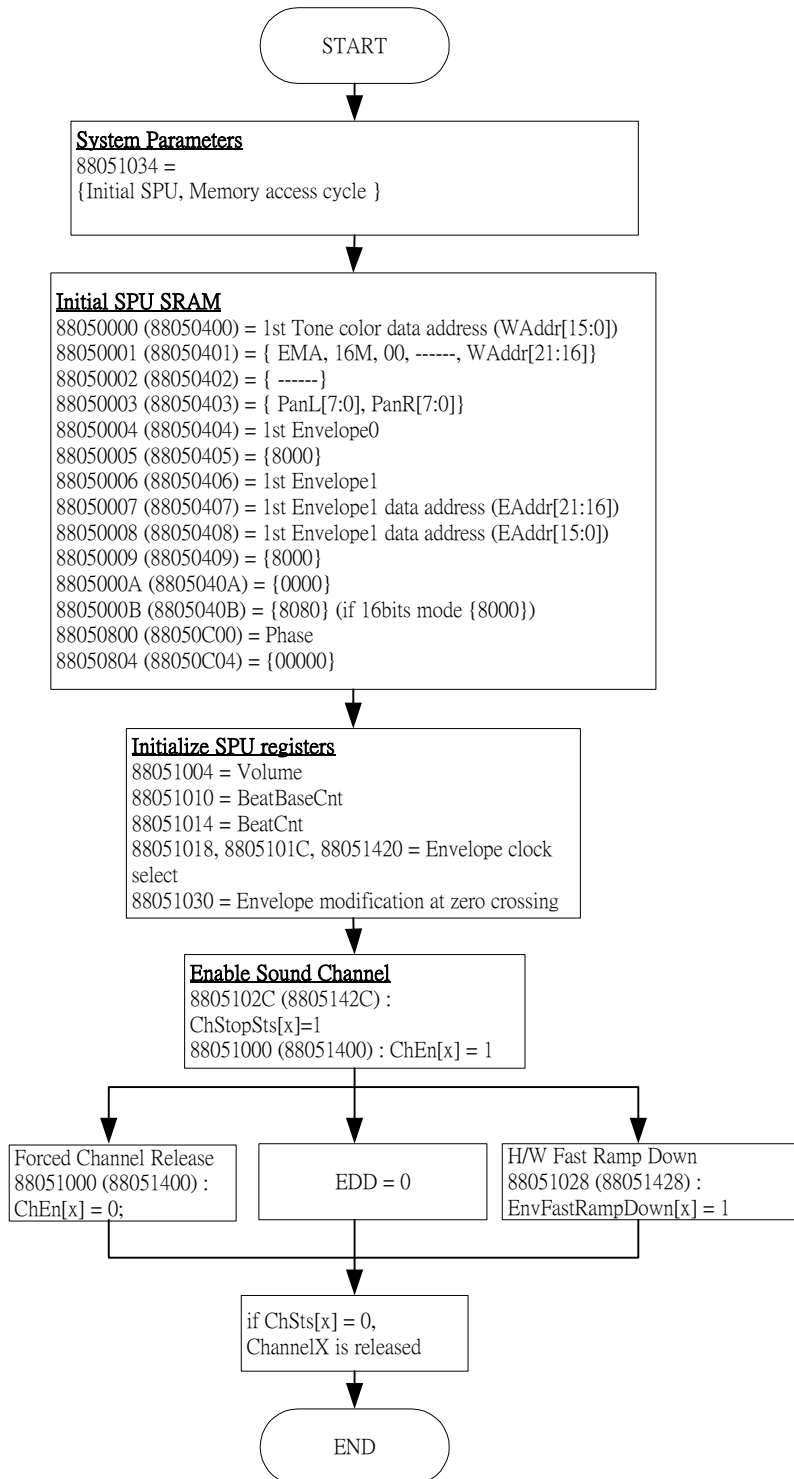
111: Envelope Ramp down one step every  $13 * 8192 * 4$  frame = 1.513 s

## 14.9 S/W Programming Flow Example

- S/W mode:



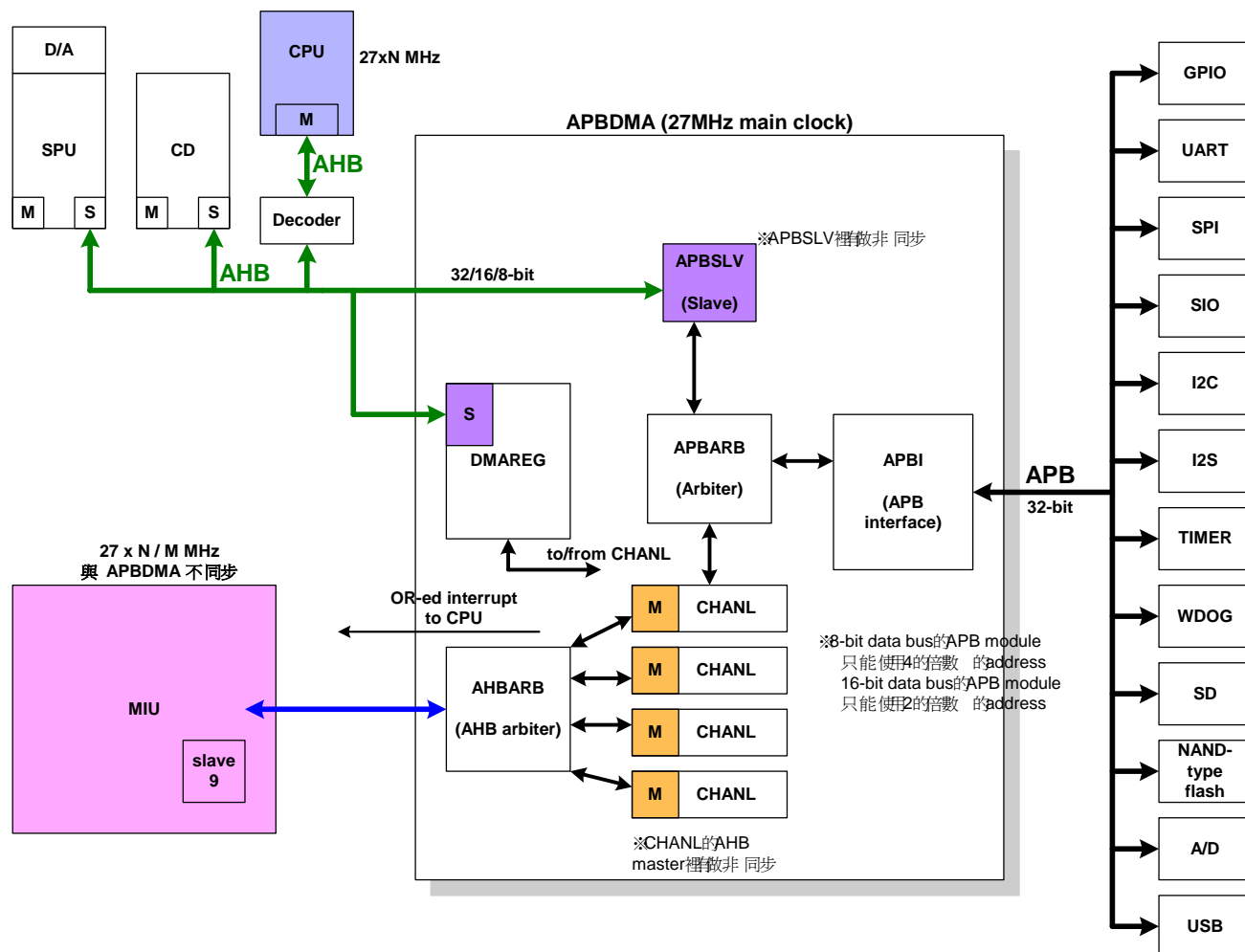
● H/W mode



## 15 APBDMA

## 15.1 Architecture

### APB Bridge and DMA Controller:



## 15.2 APB Bridge

CPU can read and write each APB peripheral Module through APB Bridge

### 15.3 DMA Controller

DMA controller can offer four Channels to R/W MIU memory for APB peripheral modules at the same time, each channel can be established into four kinds of transmission means:

- 1) 8 bits single transfer
- 2) 16 bits single transfer
- 3) 32 bits single transfer
- 4) 32 bits burst transfer.

DMA has two kinds of starting mode:

Case 1: When channel enable is set to 1, DMA controller begins to read or write the data

successively and terminates when it is done.

Case 2: Channel enable is set to 1, DMA controller carries out Read or Write movements once when APB peripheral modules send out REQ and terminate when all REQ finish sending to DMA.

The space planning of Memory has two ways when DMA R/W MIU:

Case 1: Single buffering. By specifying the start and end of DMA address, DMA controller only reads or writes this memory space, and is terminated by sending out IRQ when read or write operation completes.

Case 2: Double buffering. By specifying the start and end address of BUFFER A and BUFFER B at the same time, DMA will read or write BUFFERA and BUFFERB alternately. It will generate an IRQ signal when either transfer completes. Set channel enable back to '0' can generate a DMA operation cease request, and DMA controller will terminate its operation when the current block transfer completes..

There are two kinds of addressing for DMA to that APB peripheral modules R/W making location ways of PORT:

Case 1: Regular address

Case 2: Continuous address

## 15.4 Control registers

### 15.4.1 P\_CH\_BUSY\_STS(0x88080000): Channel Busy Status

NAME	D3	D2	D1	D0
P_CH_BUSY_STS	CH3_BUSY	CH2_BUSY	CH1_BUSY	CH0_BUSY

CH0\_BUSY: Channel 0 Busy Status

CH1\_BUSY: Channel 1 Busy Status

CH2\_BUSY: Channel 2 Busy Status

CH3\_BUSY: Channel 3 Busy Status

### 15.4.2 P\_CH\_IRQ\_STS(0x88080004): Channel Interrupt Status

NAME	D3	D2	D1	D0
P_CH_IRQ_STS	CH3_IRQ	CH2_IRQ	CH1_IRQ	CH0_IRQ

CH0\_IRQ: Channel 0 IRQ Status

Read 0: DMA IRQ not occurred

Read 1: DMA IRQ occurred

Write 0: No operation

Write 1: DMA IRQ Clear

CH1\_IRQ: Channel 1 IRQ Status

Read 0: DMA IRQ not occurred

Read 1: DMA IRQ occurred

Write 0: No operation

Write 1: DMA IRQ Clear

CH2\_IRQ: Channel 2 IRQ Status

Read 0: DMA IRQ not occurred

Read 1: DMA IRQ occurred

Write 0: No operation

Write 1: DMA IRQ Clear

CH3\_IRQ: Channel 3 IRQ Status

Read 0: DMA IRQ not occurred

Read 1: DMA IRQ occurred

Write 0: No operation

Write 1: DMA IRQ Clear

#### 15.4.3 Buffer A Start Address

NAME	ADDRESS	D31-D0
P_AHB_START_ADDR1_A	88080008H	BUFFER A Start Address For Channel1
P_AHB_START_ADDR2_A	8808000CH	BUFFER A Start Address For Channel2
P_AHB_START_ADDR3_A	88080010H	BUFFER A Start Address For Channel3
P_AHB_START_ADDR4_A	88080014H	BUFFER A Start Address For Channel4

#### 15.4.4 Buffer A End Address

NAME	ADDRESS	D31-D0
P_AHB_END_ADDR1_A	88080018H	BUFFER A End Address For Channel1
P_AHB_END_ADDR2_A	8808001CH	BUFFER A End Address For Channel2
P_AHB_END_ADDR3_A	88080020H	BUFFER A End Address For Channel3
P_AHB_END_ADDR4_A	88080024H	BUFFER A End Address For Channel4

#### 15.4.5 APB Module Access Port Start Address

NAME	ADDRESS	D31-D0
P_APB_START_ADDR1	88080028H	APB Module Access Port Start Address For Channel1
P_APB_START_ADDR2	8808002CH	APB Module Access Port Start Address For Channel2
P_APB_START_ADDR3	88080030H	APB Module Access Port Start Address For Channel3
P_APB_START_ADDR4	88080034H	APB Module Access Port Start Address For Channel4

#### 15.4.6 Buffer B Start Address

NAME	ADDRESS	D31-D0
P_AHB_START_ADDR1_B	8808004CH	BUFFER B Start Address For Channel1
P_AHB_START_ADDR2_B	88080050H	BUFFER B Start Address For Channel2
P_AHB_START_ADDR3_B	88080054H	BUFFER B Start Address For Channel3
P_AHB_START_ADDR4_B	88080058H	BUFFER B Start Address For Channel4

#### 15.4.7 Buffer B End Address

NAME	ADDRESS	D31-D0
<b>P_AHB_END_ADDR1_B</b>	<b>8808005CH</b>	BUFFER B End Address For Channel1
<b>P_AHB_END_ADDR2_B</b>	<b>88080060H</b>	BUFFER B End Address For Channel2
<b>P_AHB_END_ADDR3_B</b>	<b>88080064H</b>	BUFFER B End Address For Channel3
<b>P_AHB_END_ADDR4_B</b>	<b>88080068H</b>	BUFFER B End Address For Channel4

#### 15.4.8 P\_CH1\_SETTING (0x8808006C): Channel One Setting

NAME	D7	D6	D5	D4	D3	D2	D1	D0
<b>P_CH1_SETTING</b>	CH1_EN	CH1_IRQ	CH1_TRANS		CH1_MEM	CH1_MODE	CH1_DMA	CH1_DIR

CH1\_DIR: Channel1 direction for R/W

0: MIU TO APB

1: APB TO MIU

CH1\_DMA: Channel1 DMA Mode Selection

0: Auto mode

1: Polling mode

CH1\_MODE: Channel1 DMA's APB peripheral Modules location mode

0: Continuous mode

1: Regular mode

CH1\_MEM: Channel1 MIU Memory mode

0: Single Buffer

1: Double Buffer

CH1\_TRANS: Channel1 transfer mode

00: 8 bits single transfer

01: 16 bits single transfer

10: 32 bits single transfer

11: 32 bits burst transfer

CH1\_IRQ: Channel1 IRQ Mask

0: Disable DMA IRQ

1: Enable DMA IRQ

CH1\_EN: Channel1 Enable

0: Disable DMA

1: Enable DMA

#### 15.4.9 P\_CH2\_SETTING (0x88080070): Channel Two Setting

NAME	D7	D6	D5	D4	D3	D2	D1	D0
------	----	----	----	----	----	----	----	----

P_CH2_SETTING	CH2_EN	CH2_IRQ	CH2_TRANS	CH2_MEM	CH2_MODE	CH2_DMA	CH2_DIR
---------------	--------	---------	-----------	---------	----------	---------	---------

CH2_DIR:	Channel2 direction for R/W						
	0: MIU TO APB						
	1: APB TO MIU						
CH2_DMA:	Channel2 DMA Mode Selection						
	0: Auto mode						
	1: Polling mode						
CH2_MODE:	Channel2 DMA's APB peripheral Modules location mode						
	0: Continuous mode						
	1: Regular mode						
CH2_MEM:	Channel2 MIU Memory mode						
	0: Single Buffer						
	1: Double Buffer						
CH2_TRANS:	Channel2 transfer mode						
	00: 8 bits single transfer						
	01: 16 bits single transfer						
	10: 32 bits single transfer						
	11: 32 bits burst transfer						
CH2_IRQ:	Channel2 IRQ Mask						
	0: Disable DMA IRQ						
	1: Enable DMA IRQ						
CH2_EN:	Channel2 Enable						
	0: Disable DMA						
	1: Enable DMA						

#### 15.4.10 P\_CH3\_SETTING (0x88080074): Channel Three Setting

NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_CH3_SETTING	CH3_EN	CH3_IRQ	CH3_TRANS	CH3_MEM	CH3_MODE	CH3_DMA	CH3_DIR	

CH3_DIR:	Channel3 direction for R/W							
	0: MIU TO APB							
	1: APB TO MIU							
CH3_DMA:	Channel3 DMA Mode Selection							
	0: Auto mode							
	1: Polling mode							
CH3_MODE:	Channel3 DMA's APB peripheral Modules location mode							
	0: Continuous mode							
	1: Regular mode							
CH3_MEM:	Channel3 MIU Memory mode							

0: Single Buffer  
 1: Double Buffer  
 CH3\_TRANS: Channel3 transfer mode  
 00: 8 bits single transfer  
 01: 16 bits single transfer  
 10: 32 bits single transfer  
 11: 32 bits burst transfer  
 CH3\_IRQ: Channel3 IRQ Mask  
 0: Disable DMA IRQ  
 1: Enable DMA IRQ  
 CH3\_EN: Channel3 Enable  
 0: Disable DMA  
 1: Enable DMA

#### 15.4.11 P\_CH4\_SETTING (0x88080078): Channel Four Setting

NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_CH4_SETTING	CH4_EN	CH4_IRQ	CH4_TRANS	CH4_MEM	CH4_MODE	CH4_DMA	CH4_DIR	

CH4\_DIR: Channel4 direction for R/W  
 0: MIU TO APB  
 1: APB TO MIU  
 CH4\_DMA: Channel4 DMA Mode Selection  
 0: Auto mode  
 1: Polling mode  
 CH4\_MODE: Channel4 DMA's APB peripheral Modules location mode  
 0: Continuous mode  
 1: Regular mode  
 CH4\_MEM: Channel4 MIU Memory mode  
 0: Single Buffer  
 1: Double Buffer  
 CH4\_TRANS: Channel4 transfer mode  
 00: 8 bits single transfer  
 01: 16 bits single transfer  
 10: 32 bits single transfer  
 11: 32 bits burst transfer  
 CH4\_IRQ: Channel4 IRQ Mask  
 0: Disable DMA IRQ  
 1: Enable DMA IRQ

CH4\_EN: Channel4 Enable  
0: Disable DMA  
1: Enable DMA

#### 15.4.12 P\_CH\_SW\_RST (0x8808007C): Channel Software Reset

NAME	D3	D2	D1	D0
P_CH_SW_RST	CH3_REST	CH2_REST	CH1_REST	CH0_REST

CH0\_REST: Channel 0 Software Reset

0: Disable  
1: Enable

CH1\_REST: Channel 1 Software Reset

0: Disable  
1: Enable

CH2\_REST: Channel 2 Software Reset

0: Disable  
1: Enable

CH3\_REST: Channel 3 Software Reset

0: Disable  
1: Enable

### 15.5 DMA END

When DMA finishes, users need to write '1' to the corresponding IRQ STATUS bit to clear DMA interrupt and write '0' to the corresponding CHANNEL ENABLE

---

## 16 BITBLT DMA WITH ALPHA BLENDING OPERATION - BLNDMA

---

### 16.1 Features

“BLNDMA” is the abbreviation of the “Blending and DMA controller”. There are the following functions for the BLNDMA controller.

- DRAM-to-DRAM DMA (BitBlt) function
- DMA Pattern Fill function
- Transparent filter (Color Key) function
- Blending function
- Clipping function
- YUV2RGB conversion function

#### 16.1.1 DRAM-to-DRAM DMA(BitBlt) Function

The BLNDMA Controller supports a DRAM-to-DRAM DMA function. This function performs data transfer from source address to destination address in the SDRAM. Source address, destination address and transfer word count must be programmed before the transfer. The transfer is word alignment. One word includes two pixel data.

The BLNDMA Controller supports the linear address mode and the block address mode. If it selects the block address mode to perform DMA function, it is just BitBlt function. BitBlt is the abbreviation of BIT Block Transfer. The BitBlt function may be combined with color key, blending and clipping function. The BitBlt background width size could be 256/320/512/640/1024/2048 (unit is pixel). The BitBlt background height size could be 240/256/480/512/1024/2048 (unit is pixel). The background width and height must be programmed before performing the BitBlt function. If it performs the DRAM-to-DRAM DMA function, the transfer word count is decided by the “DMA\_TRANS\_WIDTH” and “DMA\_TRANS\_HEIGHT” for the register address 0x880D000C. They are 12 bits for each other, that is, the maximum transfer size is 4096x4096 words. The “DMA\_TRANS\_WIDTH” must be the multiple of the 8 words.

#### 16.1.2 DMA Pattern Fill Function

Pattern Fill function is to fill a rectangle with constantly color value. If it performs the DMA Pattern Fill Function, Pattern is decided by the 32 bit Register “DMA\_FILL\_PAT” for the register address 0x880D0010.

#### 16.1.3 Transparent Filter (Color Key) Function

If it performs the DRAM-to-DRAM DMA with the transparent filter function (transparent BitBlt), the transparent filter pattern is decided by the 16 bits Register “DMA\_FILTER\_PAT” for the register

address 0x880D0020. If the transferred data is matched with the transparent filter pattern, then it does not write into the destination address in the SDRAM.

#### 16.1.4 Blending Function

The Blending function is based on the two constant alpha values (SrcA\_Factor and SrcB\_Factor). The alpha value is decided by the 6 bit Register "SrcA\_Factor and SrcB\_Factor" for the register address 0x880D001C. There are two ways to perform the blending function and make the overlapping effect. One is that two source bitmaps (SrcA and SrcB) do the blending and write into the destination bitmap.

$$\text{Dst.R} = \text{round}((\text{SrcA.R} * \text{SrcA\_Factor} \pm \text{SrcB.R} * \text{SrcB\_Factor}) / 64)$$

$$\text{Dst.G} = \text{round}((\text{SrcA.G} * \text{SrcA\_Factor} \pm \text{SrcB.G} * \text{SrcB\_Factor}) / 64)$$

$$\text{Dst.B} = \text{round}((\text{SrcA.B} * \text{SrcA\_Factor} \pm \text{SrcB.B} * \text{SrcB\_Factor}) / 64)$$

The other is that one source bitmap (SrcA) and the destination bitmap (Dst) do the blending and write back to the destination bitmap.

$$\text{Dst.R} = \text{round}((\text{SrcA.R} * \text{SrcA\_Factor} \pm \text{Dst.R} * \text{SrcB\_Factor}) / 64)$$

$$\text{Dst.G} = \text{round}((\text{SrcA.G} * \text{SrcA\_Factor} \pm \text{Dst.G} * \text{SrcB\_Factor}) / 64)$$

$$\text{Dst.B} = \text{round}((\text{SrcA.B} * \text{SrcA\_Factor} \pm \text{Dst.B} * \text{SrcB\_Factor}) / 64)$$

#### 16.1.5 Clipping Function

The clipping function clips the copied source bitmap according to the BitBlt background width and height size. It only can perform the clipping function in the block address mode. If it is the linear address mode, there is not the clipping function.

#### 16.1.6 YUV2RGB Conversion Function

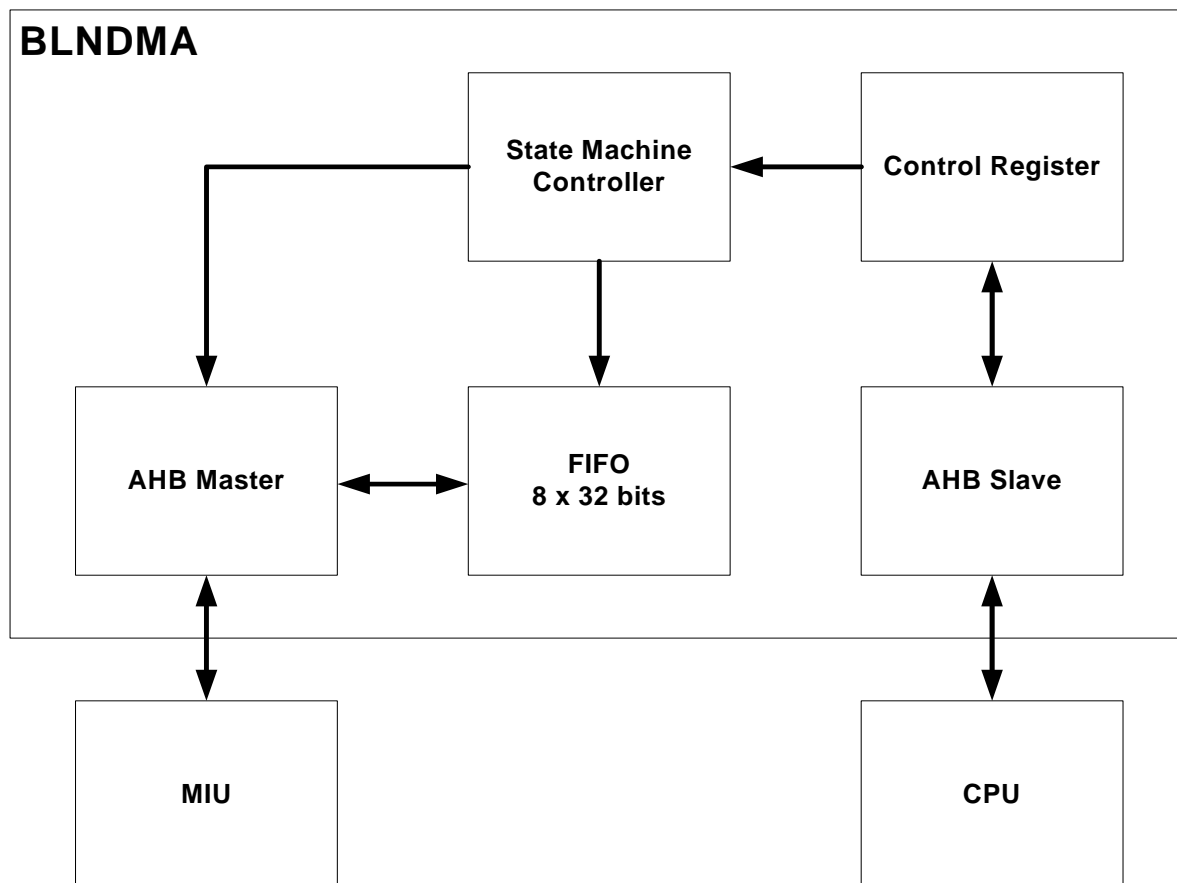
The YUV2RGB function is to convert the image from YUV422 format to the RGB555 or ARGB1555.

The BLNDMA output color format is 16 bits (RGB565 or ARGB1555) and the source of bitblt and blending may be 16 bits RGB565 or 16 bits ARGB1555 except the YUV2RGB function.

When the BLNDMA controller performs the above functions, there are the two way to let CPU know the BLNDMA status. One is the Polling Mode, CPU can read the bit 8 "DMA\_STATUS" for the register address 0x880D0018 and monitors the BLNDMA Controller whether the data transfer completes or not. The second is the Interrupt Mode. When the data transfer completes, the BLNDMA Controller sends the interrupt to the CPU.

## 16.2 Architecture

The architecture of the BLNDMA Controller is shown below.



The BLNDMA Controller is consisted of the AHB Slave, State Machine Controller and AHB Master.

CPU can access the internal control register of the BLNDMA controller by the AHB Slave of the BLNDMA controller. It can activate the state machine controller by the control register setting. Then the state machine controller controls the AHB Master of the BLNDMA controller and access the data in the SDRAM through the MIU controller. For each time the AHB Master reads 8 words data from MIU into the internal FIFO of the BLNDMA controller and performs the function of the BLNDMA controller, then writes into the SDRAM.

## 16.3 Control registers

### 16.3.1 P\_BLNDMA\_SRC\_A\_ADDR(R/W) (0x880D0000): DMA Source A Starting Address

NAME	D27-D0
P_BLNDMA_SRC_A_ADDR	DMA_SRC_A_ADDR

### 16.3.2 P\_BLNDMA\_SRC\_B\_ADDR(R/W) (0x880D0004): DMA Source B Starting Address

NAME	D27-D0
P_BLNDMA_SRC_B_ADDR	DMA_SRC_B_ADDR

### 16.3.3 P\_BLNDMA\_DEST\_ADDR(R/W) (0x880D0008): DMA Destination Starting Address

NAME	D27-D0
P_BLNDMA_DEST_ADDR	DMA_DEST_ADDR

### 16.3.4 P\_BLNDMA\_WIDTH\_HEIGHT(R/W) (0x880D000C): DMA Transfer Width

NAME	D26-D16	D15-D11	D10-D0
P_BLNDMA_WIDTH_HEIGHT	DMA_TRANS_HEIGHT	-	DMA_TRANS_WIDTH

Note: 1. Unit is word, one word including two pixel data.

2. Must be the multiple of the 8 words.

### 16.3.5 P\_BLNDMA\_FILL\_PAT(R/W) (0x880D0010): DMA Fill Pattern

NAME	D31-D0
P_BLNDMA_FILL_PAT	DMA_FILL_PAT

### 16.3.6 P\_BLNDMA\_CONTROL\_1(R/W) (0x880D0014): DMA Operation Mode

NAME	D24	D23-D17	D16	D15-D9	D8	D7-D3	D2-D0
P_BLNDMA_CONTROL_1	DMA_START		FILTER_MODE		BLEND_MODE		OP_MODE

OP\_MODE: DMA operation mode

0: Idle mode

1: Copy A and Paste to Destination

2: Copy A, B to do Blending and Paste to Destination

3: Fill Pattern to Destination

4: Copy A. Format conversion (YUV2RGB) and paste to Destination

BLEND\_MODE: DMA Blending mode

0:  $(A \times A\_FACTOR + B \times B\_FACTOR) \gg 6$

1:  $(A \times A\_FACTOR - B \times B\_FACTOR) \gg 6$

FILTER\_MODE: DMA transparent filter mode

0: Disable

1: Enable

DMA\_START: DMA operating function start

- 0: Disable  
 1: Enable(Start)

### 16.3.7 P\_BLNDMA\_IRQ\_CONTROL(R/W) (0x880D0018): DMA Interrupt Control

NAME	D24	D23-D17	D16	D15-D9	D8	D7-D1	D0
<b>P_BLNDMA_IRQ_CONTROL</b>	<b>DMA_INT_CLEAR</b>		<b>DMA_INT_EN</b>		<b>DMA_BUSY</b>		<b>DMA_IRQ</b>

- DMA\_IRQ: DMA Interrupt Status  
 0: DMA Interrupt does not occur  
 1: DMA Interrupt occurs
- DMA\_BUSY: DMA Operation Busy  
 0: DMA Operation is Idle  
 1: DMA Operation is Busy
- DMA\_INT\_EN: DMA Interrupt Control  
 0: Disable  
 1: Enable
- DMA\_INT\_CLEAR: DMA Interrupt Clear  
 0: none  
 1: Clear DMA Interrupt

### 16.3.8 P\_BLNDMA\_BLEND\_FACTOR(R/W) (0x880D001C): DMA Blending Factor

NAME	D13-D8	D7-D6	D5-D0
<b>P_BLNDMA_BLEND_FACTOR</b>	<b>DMA_B_FACTOR</b>		<b>DMA_A_FACTOR</b>

- DMA\_A\_FACTOR: Blend source A Alpha value
- DMA\_B\_FACTOR: Blend source B Alpha value

### 16.3.9 P\_BLNDMA\_TRANSPARENT(R/W) (0x880D0020): DMA Transparent Filter Pattern

NAME	D15-D0
<b>P_BLNDMA_TRANSPARENT</b>	<b>DMA_FILTER_PAT</b>

- DMA\_FILTER\_PAT: DMA Transparent Filter Pattern

### 16.3.10 P\_BLNDMA\_ADDR\_MODE(R/W)(0x880D0024): DMA Address Mode Control

NAME	D16	D15-D9	D8	D7-D1	D0
<b>P_BLNDMA_ADDR_MODE</b>	<b>DEST_ADDR</b>		<b>B_ADDR</b>		<b>A_ADDR</b>

- A\_ADDR: DMA Source A Addressing Mode  
 0: Linear Mode  
 1: Block Mode
- B\_ADDR: DMA Source B Addressing Mode  
 0: Linear Mode  
 1: Block Mode

DEST\_ADDR: DMA Destination Addressing Mode

0: Linear Mode

1: Block Mode

#### 16.3.11 P\_BLNDMA\_CONTROL\_2(R/W) (0x880D0028): DMA Control 2

NAME	D18-D16	D15-D9	D8	D7-D1	D0
P_BLNDMA_CONTROL_2	DMA_STATE		DMA_COLOR_MODE		DMA_ALPHA

DMA\_ALPHA: Alpha bit for the ARGB1555 mode

0: Disable

1: Enable

DMA\_COLOR\_MODE: RGB type select

0: RGB565

1: ARGB1555

DMA\_STATE: DMA State Machine Status

0: DMA is Idle State

1: DMA is Read A State

2: DMA is Read B State

3: DMA is Blend State or YUV2RGB Convert State

4: DMA is Write State

#### 16.3.12 P\_BLNDMA\_ABASE\_ADDR(R/W) (0x880D0030): DMA Source A Base Address

NAME	D27-D0
P_BLNDMA_ABASE_ADDR	DMA_ABASE_ADDR

#### 16.3.13 P\_BLNDMA\_AOFFSET\_XY(R/W) (0x880D0034): DMA Source A Offset Address XY

NAME	D26-D16	D15-D11	D10-D0
P_BLNDMA_AOFFSET_XY	OFFSETA_Y		OFFSETA_X

Note: 1. Unit is word, one word including two pixel data.

2. Must be the multiple of the 1 words.

#### 16.3.14 P\_BLNDMA\_A\_BG(R/W) (0x880D0038): DMA Source A Background Width/Height

NAME	D10-D8	D7-D3	D2-D0
P_BLNDMA_A_BG	BG_HEIGH		BG_WIDTH

BG\_WIDTH: DMA Source A Background Width (Pixel)

0: 256

1: 320

2: 512

3: 640

4: 1024

5: 2048

BG\_HEIGH: DMA Source A Background Height (Pixel)

0: 240

1: 256

2: 480

3: 512

4: 1024

5: 2048

#### 16.3.15 P\_BLNDMA\_BBASE\_ADDR(R/W) (0x880D0040): DMA Source B Base Address

NAME	D27-D0
P_BLNDMA_BBASE_ADDR	DMA_BBASE_ADDR

#### 16.3.16 P\_BLNDMA\_BOFFSET\_XY(R/W) (0x880D0044): DMA Source B Offset Address XY

NAME	D26-D16	D15-D11	D10-D0
P_BLNDMA_BOFFSET_XY	OFFSETB_Y		OFFSETB_X

Note:1. Unit is word, one word including two pixel data.

2. Must be the multiple of the 1 words.

#### 16.3.17 P\_BLNDMA\_B\_BG(R/W) (0x880D0048): DMA Source B Background Width/Height

NAME	D10-D8	D7-D3	D2-D0
P_BLNDMA_B_BG	BG_HEIGH		BG_WIDTH

BG\_WIDTH: DMA Source B Background Width (Pixel)

0: 256

1: 320

2: 512

3: 640

4: 1024

5: 2048

BG\_HEIGH: DMA Source B Background Height (Pixel)

0: 240

1: 256

2: 480

3: 512

4: 1024

5: 2048

#### 16.3.18 P\_BLNDMA\_DBASE\_ADDR(R/W) (0x880D0050): DMA Destination Base Address

NAME	D27-D0
------	--------

<b>P_BLNDMA_DBASE_ADDR</b>	<b>DMA_DBASE_ADDR</b>
----------------------------	-----------------------

#### 16.3.19 P\_BLNDMA\_DOFFSET\_XY(R/W) (0x880D0054): DMA Destination Offset Address XY

NAME	D26-D16	D15-D11	D10-D0
<b>P_BLNDMA_DOFFSET_XY</b>	OFFSETD_Y		OFFSETD_X

Note:1. Unit is word, one word including two pixel data.

2. Must be the multiple of the 1 words.

#### 16.3.20 P\_BLNDMA\_D\_BG(R/W)(0x880D0058): DMA Destination Background Width/Heigh

NAME	D10-D8	D7-D3	D2-D0
<b>P_BLNDMA_D_BG</b>	BG_HEIGH		BG_WIDTH

BG\_WIDTH: DMA Source A Background Width (Pixel)

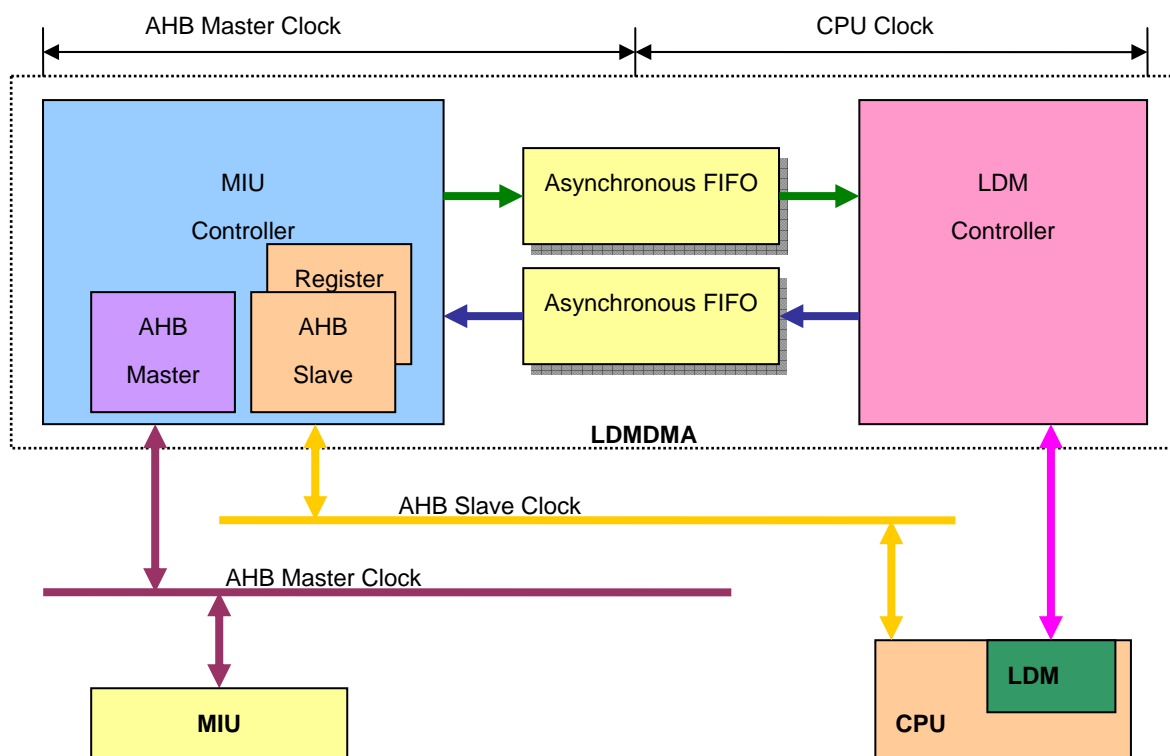
- 0: 256
- 1: 320
- 2: 512
- 3: 640
- 4: 1024
- 5: 2048

BG\_HEIGH: DMA Source A Background Height (Pixel)

- 0: 240
- 1: 256
- 2: 480
- 3: 512
- 4: 1024
- 5: 2048

## 17 LOCAL DATA MEMORY - LDM

### 17.1 Architecture



MIU controller, acting as AHB Master, is used to access MIU, and LDM controller comes to act as master to access Local Data Memory (LDM) within CPU. Besides, CPU could set registers within LDMDMA by AHB slave.

### 17.2 LDMDMA Controller

This DMA controller is half a duplexing way, and only permits a kind of route (MIU to LDM/ LDM to MIU) to exist at the same time. When LDMDMA finishes, it will send out an interrupt to notify CPU. There are four kinds of transmission ways that we support:

- 1) byte mode
- 2) half-word mode
- 3) word mode
- 4) 4-beat burst word mode

Because the whole system is that 32 bits bus in a main fact, that takes word address as the core, so will get better performance by 4-beat burst word mode or word mode. But if you want to use the transmission relying mainly on byte address or half-word address in system, SPG290 supports byte mode and half-word mode too. So SPG290 provides these four modes for users to make a better choice.

### 17.3 Control registers

Control registers in LDMDMA is accessed (R/W) by CPU through AHB slave.

#### 17.3.1 P\_DMA\_CTRL (0x880C0000): DMA Control

NAME	D31	D30	D29	D28	D27-D26	D25-D0
P_DMA_CTRL	MODULE_EN	IRQ_EN	DIRECT	-	TRANS_MODE	-

TRANS_MODE:	Transfer Mode
00:	byte mode (8bits)
01:	half-word mode (16bits)
10:	word mode (32bits)\
11:	4-beat burst word mode (32bits burst)
DIRECT:	Access direction
0:	MIU to LDM
1:	LDM to MIU
IRQ_EN:	IRQ Enable
0:	Disable
1:	Enable
MODULE_EN:	Module Enable
0:	Disable
1:	Enable

#### 17.3.2 P\_DMA\_STATUS (0x880C0004): DMA Status

NAME	D31	D30-D0
P_DMA_STATUS	INT_REQ	-

INT_REQ:	Interrupt request (IRQ) (MIU to LDM/ LDM to MIU finish)
Read 0:	not occurred
Read 1:	occurred
Write 0:	no operation
Write 1:	Clear Status

#### 17.3.3 P\_MIU\_START\_ADDR (0x880C0008): MIU Start Address

NAME	D27-D0
P_MIU_START_ADDR	MIU_START_ADDR (Byte address mode)

#### 17.3.4 P\_MIU\_END\_ADDR (0x880C000C): MIU End Address

NAME	D27-D0
P_MIU_END_ADDR	MIU_END_ADDR (Byte address mode)

**17.3.5 P\_LDM\_START\_ADDR (0x880C0010): LDM Start Address**

NAME	D17-D0
P_LDM_START_ADDR	LDM_START_ADDR (Byte address mode)

**17.3.6 P\_LDM\_END\_ADDR (0x880C0014): LDM End Address**

NAME	D17-D0
P_LDM_END_ADDR	LDM_END_ADDR (Byte address mode)

---

---

## 18 PERIPHERAL INTERRUPT CONTROLLER

---

---

### 18.1 INTRODUCTION

The interrupt controller provides masking capability for all of 40-interrupt sources and combines them into their final state for IRQ processor interrupt. The role of interrupt controller is to generate the IRQ interrupt request to the S\*Core processor after making arbitration process when there are multiple interrupt requests from internal peripherals and external interrupt request pins.

S\*Core processor only receives the IRQ interrupt events from the peripheral devices, but does not have a priority mechanism in it. Therefore the software like interrupt handler must handle this kind of issues with hardware in the interrupt controller. For example, assume that all of interrupt sources have been set as IRQ, and ten of them requested interrupt to processor at same time, the software can determine the interrupt service priority by reading interrupt pending register, which denotes what kind of interrupt request happens.

This kind of interrupt process requires long interrupt latency to jump to the corresponding service routine. To resolve those inefficient processes, S\*Core processor provides another interrupt processing mechanism called vectored interrupt mode, a general feature of the CISC type Microprocessor. When the multiple interrupt sources request interrupt, hardware priority logic determines which interrupt should be serviced. At same time, this hardware logic applies the offset value from the vector table base address, which has a jump instruction to the corresponding service routine. With hardware assistance, it only needs several instructions to fetch a vector table entry and reduce the interrupt latency dramatically.

### 18.2 FEATURE

- Compliance to the AMBA specification (Rev 2.0) for easy integration into System-on-a-chip (SoC) implementation.
- 40 interrupt sources.
- Each interrupt is high active, and level sensitive IRQ.
- Vectored interrupt support.
- Programmable Priority of each interrupt source.

### 18.3 Control Register

The following table shows the registers associated with the interrupt controller and physical address used to access them.

#### 18.3.1 P\_INTPND (R) (0x880A0000): Interrupt Request Status Register.

Each of the 40 bits in the interrupt pending register, INTPND, corresponds to an interrupt source. When an interrupt request is generated from the external device, the corresponding bit in the register is set to "1". Although several interrupt sources generate requests simultaneously, the INTPND indicates what interrupt sources generate the interrupt requests.

NAME	D31-D0
P_INTPND	INTPND

INTPND: Indicates the interrupt request status

0: The interrupt has not been requested

1: The interrupt source has asserted the interrupt request

### 18.3.2 P\_INTPND\_H (R) (0x880A0004): Interrupt Request Status High Byte

NAME	D7-D0
P_INTPND_H	INTPND_H

INTPND\_H: Indicates the interrupt request status

0: The interrupt has not been requested

1: The interrupt source has asserted the interrupt request

### 18.3.3 P\_I\_PMST (R/W) (0x880A0008):

This register determines the priority of among 4 IRQ service slave. Each IRQ service slave can accept 8 IRQ source.

NAME	D7-D0
P_I_PMST	X_PMST

X\_PMST: These 8 bits determine priorities among 4 slaves.

[1:0]: Priority of slave group 0

[3:2]: Priority of slave group 1

[5:4]: Priority of slave group 2

[7:6]: Priority of slave group 3

00 = 1<sup>st</sup>, 01 = 2<sup>nd</sup>, 02 = 3<sup>rd</sup>, 03 = 4<sup>th</sup>

### 18.3.4 IRQ Priority Register

This register determines the priority of each IRQ service slave. Each IRQ service slave can accept 8 IRQ source.

NAME	ADDR	D23-D21	D20-D18	D17-D15	D14-D12	D11-D9	D8-D6	D5-D3	D2-D0
P_I_PSLV0	0x880A0010	PSLV0_7	PSLV0_6	PSLV0_5	PSLV0_4	PSLV0_3	PSLV0_2	PSLV0_1	PSLV0_0
P_I_PSLV1	0x880A0014	PSLV1_7	PSLV1_6	PSLV1_5	PSLV1_4	PSLV1_3	PSLV1_2	PSLV1_1	PSLV1_0
P_I_PSLV2	0x880A0018	PSLV2_7	PSLV2_6	PSLV2_5	PSLV2_4	PSLV2_3	PSLV2_2	PSLV2_1	PSLV2_0
P_I_PSLV3	0x880A001C	PSLV3_7	PSLV3_6	PSLV3_5	PSLV3_4	PSLV3_3	PSLV3_2	PSLV3_1	PSLV3_0

PSLVX\_0: Priority of source 0

PSLVX\_1: Priority of source 1

PSLVX\_2: Priority of source 2

PSLVX\_3: Priority of source 3

PSLVX\_4: Priority of source 4

PSLVX\_5: Priority of source 5

PSLVX\_6: Priority of source 6

PSLVX\_7: Priority of source 7

NOTE: X = Slave X

000 = 1<sup>st</sup>, 001 = 2<sup>nd</sup>, 010 = 3<sup>rd</sup>, 011 = 4<sup>th</sup>

100 = 5<sup>th</sup>, 101 = 6<sup>th</sup>, 110 = 7<sup>th</sup>, 111 = 8<sup>th</sup>

## 18.4 VECADDR

The interrupt handler has the vector address to return to the processor respective to IRQ interrupt. The interrupt handler uses this vector value without calculating the offset address in the interrupt vector table to find the interrupt service routine. It allows the processor to skip the search process required in the non-vectored mode to get the offset address whenever it receives the interrupt. This mechanism makes a system improve the overall performance dramatically. The value in this register will not change until another interrupt happens.

## 18.5 Interrupt Sources

Slave Group	Source Number	Source	Vector address
0	0	SPU FIQ	63
	1	SPU BeatIRQ	62
	2	SPU EnvelopeIRQ	61
	3	CD servo	60
	4	ADC gain overflow / ADC recorder FIFO overflow	59
	5	General purpose ADC	58
	6	Timer base	57
	7	Timer	56
1	8	TV vblanking start	55
	9	LCD vblanking start	54
	10	PPU vblanking start	53
	11	Light Gun	52
	12	Sensor frame end	51
	13	Sensor coordinate hit	50
	14	Sensor motion frame end	49
	15	Sensor capture done + sensor debug IRQ	48
2	16	TV coordinate hit	47
	17	PPU coordinate hit	46
	18	USB host+device	45
	19	SIO	44
	20	SPI	43
	21	UART (IrDA)	42
	22	NAND	41
	23	SD	40
3	24	I <sup>2</sup> C master	39
	25	I <sup>2</sup> S slave	38
	26	APBDMA CH1	37
	27	APBDMA CH2	36
	28	LDM_DMA	35
	29	BLN_DMA	34
	30	APBDMA CH3	33
	31	APBDMA CH4	32

4	32	Alarm + HMS	31
	33	MP4	30
	34	C3 (ECC module)	29
	35	GPIO	28
	36	Bufctl (for debug) + TV/PPU vblanking end (for debug)	27
	37	RESERVED1	26
	38	RESERVED2	25
	39	RESERVED3	24

## 19 SOFTWARE CONFIG - SFTCFG

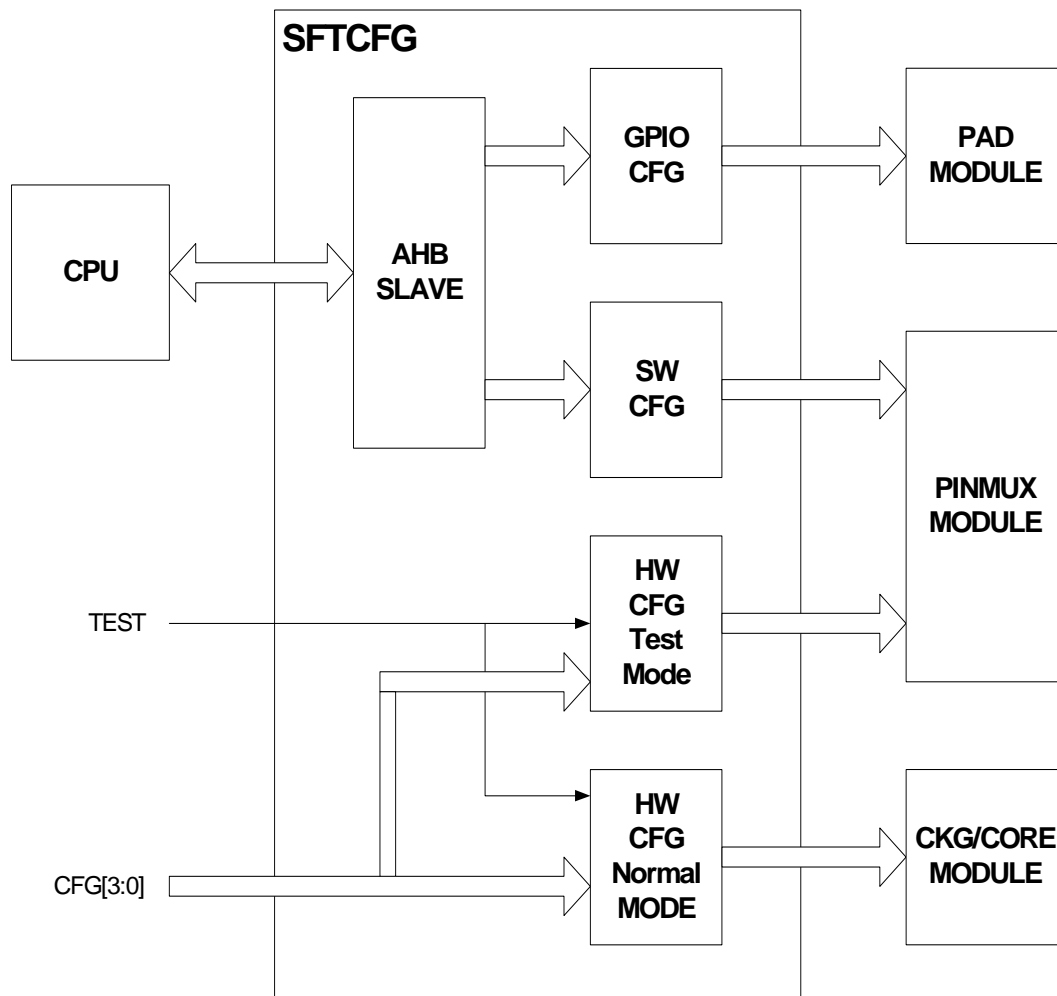
### 19.1 Function Description

There are the five groups of signals generated by the Software Configuration (SFTCFG) module. The first is the hardware configuration in the normal mode and generates the JTAG enable, warm configuration reset, boot mode and bypass pll signals. The second is the hardware configuration in the test mode and the generated signals are connected to the pin multiplex (PINMUX) module as the mode selection signals. The third is the software configuration by the CPU programming the registers and the generated signals are also connected to the pin multiplex (PINMUX) module as the mode selection signals. The fourth is the GPIO configuration by the CPU programming the registers and the generated signals including output port data, output enable, pull up and pull down signals are also connected to the input/output pads.

The fifth is that the GPIOs are as the external interrupt sources and they can be programmed to be rising of falling trigger. When the external interrupt source triggers, the SFTCFG module sends the interrupt to the CPU.

### 19.2 Architecture

The architecture of the SFTCFG module is shown below.



## 19.3 Control registers

### 19.3.1 P\_GPIO\_DEVICE\_SET(0x88200000): Software Configuration for Device Selection

NAME	D24	D23-D17	D16	D15-D10	D9-D8	D7-D2	D1-D0
P_GPIO_DEVICE_SET	SW_UART		SW_TVE		SW_CSI		SW_LCD

SW_LCD:	Software Configuration for LCD selection
00:	TFT_AUO
01:	TFT_TOPPOLY
10:	STN
11:	Reserved
SW_CSI:	Software Configuration for Sensor Interface Format selection
00:	CCIR601
01:	Sunplus Format
10:	CCIR656
11:	Reserved
SW_TVE:	Software Configuration for TV Encoder Hsync/Vsync Probe Selection
0:	Disable
1:	Enable
SW_UART:	Software Configuration for UART selection
0:	As GPIO
1:	UART Selection

### 19.3.2 P\_GPIO\_FUNC\_SET(0x88200004): Software Configuration for Function Selection

NAME	D24	D23-D17	D16	D15-D10	D9-D8	D7-D1	D0
P_GPIO_FUNC_SET	SW_EROM		SW_CSI_PROBE		SW_PERI		SW_I2C

SW_I2C:	Software Configuration for I2C Selection
0:	As GPIO
1:	I2C Selection
SW_PERI:	Software Configuration for Peripheral Selection
00:	JTAG
01:	SIO
10:	I2S Master Mode
11:	I2S Slave Mode
SW_CSI_PROBE:	Software Configuration for Sensor Hsync/Vsync/Field Probe Selection
0:	Disable
1:	Enable
SW_EROM:	Software Configuration for External ROM Selection
0:	As GPIO
1:	External ROM Selection

### 19.3.3 P\_GPIO\_DRAM\_SET(0x88200008): Software Configuration for DRAM Selection

NAME	D24	D23-D17	D16	D15-D9	D8	D7-D3	D2-D0
<b>P_GPIO_DRAM_SET</b>	SW_DRAM_BA1		SW_DRAM_A12		SW_DRAM_A11		SW_KEYC

SW\_KEYC: Software Configuration for Key Change Selection

- 1: TFT Interface
- 2: CSI Interface
- 3: ADC1 Configuration
- 4: ADC2 Configuration
- 5: NFLASH Interface

SW\_DRAM\_A11: Software Configuration for DRAM A11 Selection

- 0: As GPIO
- 1: DRAM A11 Selection

SW\_DRAM\_A12: Software Configuration for DRAM A12 Selection

- 0: As GPIO
- 1: DRAM A12 Selection

SW\_DRAM\_BA1: Software Configuration for DRAM BA1 Selection

- 0: As GPIO
- 1: DRAM BA1 Selection

### 19.3.4 P\_GPIO\_OUTPUT(0x8820000C): Software Configuration for Transceiver Output Selection

NAME	D24	D23-D17	D16	D15-D9	D8	D7-D1	D0
<b>P_GPIO_OUTPUT</b>	SW_SERVO_34		SW_SERVO_12		SW_VDAC_OUT		SW_USBTRX_OUT

SW\_USBTRX\_OUT: Software Configuration for USB transceiver Output

- 0: Disable
- 1: Enable

SW\_VDAC\_OUT: Software Configuration for VIDEO DAC Output

- 0: Disable
- 1: Enable

SW\_SERVO\_12: Software Configuration for CD Servo Test12

- 0: Disable
- 1: Enable

SW\_SERVO\_34: Software Configuration for CD Servo Test34

### 19.3.5 P\_GPIO\_LIGHTPEN\_PORT(0x88200010): Software Configuration for LightPen Selection

NAME	D24	D23-D22	D21-D16	D15-D10	D9	D8	D7-D1	D0
<b>P_GPIO_LIGHTPEN_PORT</b>	SW_AFE		SW_TCCP		SW_LP1	SW_LP0		SW_SPEED

SW\_SPEED: Software Configuration for Speed Test

- 0: Disable
- 1: Enable

SW\_LP0: Software Configuration for LightPen 0

- 0: Disable
- 1: Enable

SW\_LP1: Software Configuration for LightPen 1

0: Disable

1: Enable

SW\_TCCP: Software Configuration for Timer CCP

0: Disable

1: Enable

SW\_AFE: Software Configuration for AFE test

0: Disable

1: Enable

#### 19.3.6 P\_GPIO\_TFT\_PORT(0x88200014): TFT GPIO Port Data

NAME	D19-D0
P_GPIO_TFT_PORT	TFT_GPIO_O

TFT\_GPIO\_O: TFT GPIO Port Data

#### 19.3.7 P\_GPIO\_TFT\_OE(0x88200018): TFT GPIO Port Output Setting

NAME	D19-D0
P_GPIO_TFT_OE	TFT_GPIO_OE

TFT\_GPIO\_OE: TFT GPIO Port Output Enable

0: Disable

1: Enable

#### 19.3.8 P\_GPIO\_TFT\_PU(0x8820001C): TFT GPIO Port Pull Up Setting

NAME	D19-D0
P_GPIO_TFT_PU	TFT_GPIO_PU

TFT\_GPIO\_PU: TFT GPIO Port Pull Up Enable

0: Disable

1: Enable

#### 19.3.9 P\_GPIO\_TFT\_PD(0x88200020): TFT GPIO Port Pull Down Setting

NAME	D19-D0
P_GPIO_TFT_PD	TFT_GPIO_PD

TFT\_GPIO\_PD: TFT GPIO Port Pull Down Enable

0: Disable

1: Enable

#### 19.3.10 P\_GPIO\_CSI\_PORT(0x88200024): CSI GPIO Port Control

NAME	D28-D16	D15-D13	D12-D0
P_GPIO_CSI_PORT	CSI_GPIO_OE		CSI_GPIO_O

CSI\_GPIO\_O: CSI GPIO Port Data

CSI\_GPIO\_OE: CSI GPIO Port Output Enable

0: Disable  
1: Enable

#### 19.3.11 P\_GPIO\_CSI\_PULL(0x88200028): CSI GPIO Port Pull Status

NAME	D28-D16	D15-D13	D12-D0
P_GPIO_CSI_PULL	CSI_GPIO_PD		CSI_GPIO_PU

CSI\_GPIO\_PU: CSI GPIO Port Pull Up Enable  
0: Disable  
1: Enable

CSI\_GPIO\_PD: CSI GPIO Port Pull Down Enable  
0: Disable  
1: Enable

#### 19.3.12 P\_GPIO\_NFLASH\_PORT(0x8820002C): NFLASH GPIO Port Control

NAME	D31-D16	D15-D0
P_GPIO_NFLASH_PORT	NFLASH_GPIO_OE	NFLASH_GPIO_O

NFLASH\_GPIO\_O: NFLASH GPIO Port Data

NFLASH\_GPIO\_OE: NFLASH GPIO Port Output Enable  
0: Disable  
1: Enable

#### 19.3.13 P\_GPIO\_NFLASH\_PULL(0x88200030): NFLASH GPIO Port Pull Status

NAME	D31-D16	D15-D0
P_GPIO_NFLASH_PULL	NFLASH_GPIO_PD	NFLASH_GPIO_PU

NFLASH\_GPIO\_PU: NFLASH GPIO Port Pull Up Enable  
0: Disable  
1: Enable

NFLASH\_GPIO\_PD: NFLASH GPIO Port Pull Down Enable  
0: Disable  
1: Enable

#### 19.3.14 P\_GPIO\_JTAG\_PORT(0x88200034): JTAG GPIO Port Control

NAME	D28-D24	D23-D21	D20-D16	D15-D13	D12-D8	D7-D5	D4-D0
P_GPIO_JTAG_PORT	JTAG_GPIO_PD		JTAG_GPIO_PU		JTAG_GPIO_OE		JTAG_GPIO_O

JTAG\_GPIO\_O: JTAG GPIO Port Data

JTAG\_GPIO\_OE: JTAG GPIO Port Output Enable  
0: Disable  
1: Enable

JTAG\_GPIO\_PU: JTAG GPIO Port Pull Up Enable  
0: Disable  
1: Enable

JTAG\_GPIO\_PD: JTAG GPIO Port Pull Down Enable  
0: Disable  
1: Enable

**19.3.15 P\_GPIO\_GLOBAL\_PORT(0x88200038): GLOBAL GPIO Port Control**

NAME	D25-D24	D23-D18	D17-D16	D15-D10	D9-D8	D7-D2	D1-D0
<b>P_GPIO_GLOBAL_PORT</b>	XGPIO_PD		XGPIO_PU		GLOBAL_OE		GLOBAL_O

GLOBAL\_O: GLOBAL GPIO Port Data

GLOBAL\_OE: GLOBAL GPIO Port Output Enable  
0: Disable  
1: Enable

XGPIO\_PU: GLOBAL GPIO Port Pull Up Enable  
0: Disable  
1: Enable

XGPIO\_PD: GLOBAL GPIO Port Pull Down Enable  
0: Disable  
1: Enable

**19.3.16 P\_GPIO\_USB\_PORT(0x8820003C): USB GPIO Port Control**

NAME	D24	D23-D17	D16	D15-D9	D8	D7-D1	D0
<b>P_GPIO_USB_PORT</b>	XUSB_PD		XUSB_PU		USBDET_OE		USBDET_O

USBDET\_O: USBDET GPIO Port Data

USBDET\_OE: USBDET GPIO Port Output Enable  
0: Disable  
1: Enable

XUSB\_PU: USBDET GPIO Port Pull Up Enable  
0: Disable  
1: Enable

XUSB\_PD: USBDET GPIO Port Pull Down Enable  
0: Disable  
1: Enable

**19.3.17 P\_GPIO\_UART\_PORT(0x88200040): UART GPIO Port Control**

NAME	D25-D24	D23-D18	D17-D16	D15-D10	D9-D8	D7-D2	D1-D0
<b>P_GPIO_UART_PORT</b>	XUART_PD		XUART_PU		UART_OE		UART_O

UART\_O: UART GPIO Port Data

UART\_OE: UART GPIO Port Output Enable  
0: Disable  
1: Enable

XUART\_PU: UART GPIO Port Pull Up Enable  
0: Disable  
1: Enable

XUART\_PD: UART GPIO Port Pull Down Enable  
0: Disable  
1: Enable

**19.3.18 P\_GPIO\_I2C\_PORT(0x88200044): I2C GPIO Port Control**

NAME	D25-D24	D23-D18	D17-D16	D15-D10	D9-D8	D7-D2	D1-D0
<b>P_GPIO_I2C_PORT</b>	X I2C_PD		X I2C_PU		I2C_OE		I2C_O

I2C\_O: I2C GPIO Port Data  
 I2C\_OE: I2C GPIO Port Output Enable  
           0: Disable  
           1: Enable  
 XI2C\_PU: I2C GPIO Port Pull Up Enable  
           0: Disable  
           1: Enable  
 XI2C\_PD: I2C GPIO Port Pull Down Enable  
           0: Disable  
           1: Enable

**19.3.19 P\_GPIO\_ADC\_PORT(0x88200048): ADC GPIO Port Control**

NAME	D31-D24	D23-D16	D15-D8	D7-D0
<b>P_GPIO_ADC_PORT</b>	XADC_PD	XAD_PU	ADC_OE	ADC_O

ADC\_O: ADC GPIO Port Data  
 ADC\_OE: ADC GPIO Port Output Enable  
           0: Disable  
           1: Enable  
 XADC\_PU: ADC GPIO Port Pull Up Enable  
           0: Disable  
           1: Enable  
 XADC\_PD: ADC GPIO Port Pull Down Enable  
           0: Disable  
           1: Enable

**19.3.20 P\_GPIO\_CDSERVO\_PORT(0x8820004C): CD-Servo GPIO Port Control**

NAME	D29-D24	D23-D22	D21-D16	D15-D14	D13-D8	D7-D6	D5-D0
<b>P_GPIO_CDSERVO_PORT</b>	CDS_PD		CDS_PU		CDS_OE		CDS_O

CDS\_O: CD Servo Port Data  
 CDS\_OE: CD Servo GPIO Port Output Enable  
           0: Disable  
           1: Enable  
 CDS\_PU: CD Servo GPIO Port Pull Up Enable  
           0: Disable  
           1: Enable  
 CDS\_PD: CD Servo GPIO Port Pull Down Enable  
           0: Disable  
           1: Enable

### 19.3.21 P\_GPIO\_DRAM\_PORT(0x88200050): DRAM GPIO Port Control

NAME	D27-D24	D23-D17	D16	D15-D12	D11-D8	D7-D4	D3-D0
P_GPIO_DRAM_PORT	XDRAM_PD		XDRAM_PU		DRAM_OE		DRAM_O

DRAM\_O: DRAM GPIO Port Data

0: ROMCSN

1: DRAM\_A11

2: DRAM\_A12

3: DRAM\_BA1

DRAM\_OE: DRAM GPIO Port Output Enable

Bit 8:

0: Disable

1: Enable

Bit 9-Bit11:

0: Enable

1: Disable

XDRAM\_PU: DRAM GPIO Port Pull Up Enable

0: Disable

1: Enable

XDRAM\_PD: DRAM GPIO Port Pull Down Enable

0: Disable

1: Enable

### 19.3.22 P\_GPIO\_ADC\_AEN(0x88200054): ADC GPIO Port Analog Input Control

NAME	D16	D15-D9	D8	D7-D0
P_GPIO_ADC_AEN	SWEFU_ENB		SWEFU_READ	ADC_GPIO_AEN

ADC\_GPIO\_AEN: ADC GPIO Port Analog Input Enable

0: Disable

1: Enable

SWEFU\_READ: EFUSE Software Read Signal, Active High

SWEFU\_ENB: EFUSE Software Enable Signal, Active Low

### 19.3.23 P\_GPIO\_TFT\_INPUT(0x88200064): TFT GPIO Input Control

NAME	D19-D0
P_GPIO_TFT_INPUT	TFT_INPUT

TFT\_INPUT: TFT GPIO Input Value

### 19.3.24 P\_GPIO\_CSI\_INPUT(0x88200068): CSI GPIO Input Control

NAME	D12-D0
P_GPIO_CSI_INPUT	CSI_INPUT

CSI\_INPUT: CSI GPIO Input Value

### 19.3.25 P\_GPIO\_NFLASH\_INPUT(0x8820006C): NFLASH GPIO Input Control

NAME	D15-D0
P_GPIO_NFLASH_INPUT	NFLASH_INPUT

NFLASH\_INPUT: NFLASH GPIO Input Value

### 19.3.26 P\_GPIO\_JCD\_INPUT(0x88200070): JTAG/CDServo/DRAM GPIO Input Control

NAME	D19-D16	D15-D14	D13-D8	D7-D5	D4-D0
P_GPIO_JCD_INPUT	DRAM_INPUT		CDS_INPUT		JTAG_INPUT

JTAG\_INPUT: JTAG GPIO Input Value

CDS\_INPUT: CD Servo GPIO Input Value

DRAM\_INPUT: DRAM GPIO Input Value

### 19.3.27 P\_GPIO\_GUUI\_INPUT(0x88200074):

NAME	D25-D24	D23-D18	D17-D16	D15-D9	D8	D7-D2	D1-D0
P_GPIO_GUUI_INPUT	I2C_IN		UART_IN		USBDET_IN		GLOBAL_IN

GLOBAL\_IN: Global GPIO Input Value

USBDET\_IN: USBDET JTAG GPIO Input Value

UART\_IN: UART GPIO Input Value

I2C\_IN: I2C GPIO Input Value

### 19.3.28 P\_GPIO\_ADC\_INPUT(0x88200078)

NAME	D7-D0
P_GPIO_ADC_INPUT	ADC_INPUT

ADC\_INPUT: ADC GPIO Input Value

### 19.3.29 P\_GPIO\_TFT\_INT(0x88200080)

NAME	D27-D24	D23-D20	D19-D16	D15-D12	D11-D8	D7-D4	D3-D0
P_GPIO_TFT_INT	TFT_FI		TFT_RI		TFT_FIEN		TFT_RIEN

TFT\_RIEN: TFT GPIO Rising Interrupt Enable

0: Disable

1: Enable

TFT\_FIEN: TFT GPIO Falling Interrupt Enable

0: Disable

1: Enable

TFT\_RI: TFT GPIO Rising Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

TFT\_FI: TFT GPIO Falling Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

### 19.3.30 P\_GPIO\_CSI\_INT(0x88200084)

NAME	D27-D24	D23-D20	D19-D16	D15-D12	D11-D8	D7-D4	D3-D0
P_GPIO_CSI_INT	CSI_FI		CSI_RI		CSI_FIEN		CSI_RIEN

CSI\_RIEN: CSI GPIO Rising Interrupt Enable

0: Disable

1: Enable

CSI\_FIEN: CSI GPIO Falling Interrupt Enable

0: Disable

1: Enable

CSI\_RI: CSI GPIO Rising Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

CSI\_FI: CSI GPIO Falling Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

### 19.3.31 P\_GPIO\_NFLASH\_INT(0x88200088)

NAME	D27-D24	D23-D20	D19-D16	D15-D12	D11-D8	D7-D4	D3-D0
P_GPIO_NFLASH_INT	NFLASH_FI		NFLASH_RI		NFLASH_FIEN		NFLASH_RIEN

NFLASH\_RIEN: NFLASH GPIO Rising Interrupt Enable

0: Disable

1: Enable

NFLASH\_FIEN: NFLASH GPIO Falling Interrupt Enable

0: Disable

1: Enable

NFLASH\_RI: NFLASH GPIO Rising Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event  
 Write 0: No Action

NFLASH \_FI: NFLASH GPIO Falling Interrupt Event

Read 1: Interrupt Event Occurs  
 Read 0: No Interrupt Event

Write 1: Clear Interrupt Event  
 Write 0: No Action

### 19.3.32 P\_GPIO\_JTAG\_INT(0x8820008C)

NAME	D28-D24	D23-D21	D20-D16	D15-D13	D12-D8	D7-D5	D4-D0
P_GPIO_JTAG_INT	JTAG_FI		JTAG_RI		JTAG_FIEN		JTAG_RIEN

JTAG\_RIEN: JTAG GPIO Rising Interrupt Enable

0: Disable  
 1: Enable

JTAG\_FIEN: JTAG GPIO Falling Interrupt Enable

0: Disable  
 1: Enable

JTAG\_RI: JTAG GPIO Rising Interrupt Event

Read 1: Interrupt Event Occurs  
 Read 0: No Interrupt Event

Write 1: Clear Interrupt Event  
 Write 0: No Action

JTAG\_FI: JTAG GPIO Falling Interrupt Event

Read 1: Interrupt Event Occurs  
 Read 0: No Interrupt Event

Write 1: Clear Interrupt Event  
 Write 0: No Action

### 19.3.33 P\_GPIO\_GLOBAL\_INT(0x88200090)

NAME	D25-D24	D23-D18	D17-D16	D15-D10	D9-D8	D7-D2	D1-D0
P_GPIO_GLOBAL_INT	GLOBAL_FI		GLOBAL_RI		GLOBAL_FIEN		GLOBAL_RIEN

GLOBAL\_RIEN: GLOBAL GPIO Rising Interrupt Enable

0: Disable  
 1: Enable

GLOBAL\_FIEN: GLOBAL GPIO Falling Interrupt Enable

0: Disable  
 1: Enable

GLOBAL\_RI: GLOBAL GPIO Rising Interrupt Event

Read 1: Interrupt Event Occurs  
Read 0: No Interrupt Event  
Write 1: Clear Interrupt Event  
Write 0: No Action

GLOBAL \_FI: GLOBAL GPIO Falling Interrupt Event  
Read 1: Interrupt Event Occurs  
Read 0: No Interrupt Event  
Write 1: Clear Interrupt Event  
Write 0: No Action

#### 19.3.34 P\_GPIO\_UART\_INT(0x88200094)

NAME	D25-D24	D23-D18	D17-D16	D15-D10	D9-D8	D7-D2	D1-D0
P_GPIO_UART_INT	UART_FI		UART_RI		UART_FIEN		UART_RIEN

UART \_RIEN: UART GPIO Rising Interrupt Enable  
0: Disable  
1: Enable

UART \_FIEN: UART GPIO Falling Interrupt Enable  
0: Disable  
1: Enable

UART \_RI: UART GPIO Rising Interrupt Event  
Read 1: Interrupt Event Occurs  
Read 0: No Interrupt Event  
Write 1: Clear Interrupt Event  
Write 0: No Action

UART \_FI: UART GPIO Falling Interrupt Event  
Read 1: Interrupt Event Occurs  
Read 0: No Interrupt Event  
Write 1: Clear Interrupt Event  
Write 0: No Action

#### 19.3.35 P\_GPIO\_I2C\_INT(0x88200098)

NAME	D25-D24	D23-D18	D17-D16	D15-D10	D9-D8	D7-D2	D1-D0
P_GPIO_I2C_INT	I2C_FI		I2C_RI		I2C_FIEN		I2C_RIEN

I2C \_RIEN: I2C GPIO Rising Interrupt Enable  
0: Disable  
1: Enable

I2C \_FIEN: I2C GPIO Falling Interrupt Enable  
0: Disable

1: Enable

I2C \_RI: I2C GPIO Rising Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

I2C \_FI: I2C GPIO Falling Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

### 19.3.36 P\_GPIO\_ADC\_INT(0x8820009C)

NAME	D27-D24	D23-D20	D19-D16	D15-D12	D11-D8	D7-D4	D3-D0
P_GPIO_ADC_INT	ADC_FI		ADC_RI		ADC_FIEN		ADC_RIEN

ADC \_RIEN: ADC GPIO Rising Interrupt Enable

0: Disable

1: Enable

ADC \_FIEN: ADC GPIO Falling Interrupt Enable

0: Disable

1: Enable

ADC \_RI: ADC GPIO Rising Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

ADC \_FI: ADC GPIO Falling Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

### 19.3.37 P\_GPIO\_USBDDET\_INT(0x882000A0)

NAME	D24	D23-D17	D16	D15-D9	D8	D7-D1	D0
P_GPIO_USBDDET_INT	USBDDET_FI		USBDDET_RI		USBDDET_FIEN		USBDDET_RIEN

USBDDET \_RIEN: USBDDET GPIO Rising Interrupt Enable

0: Disable

1: Enable

USBDET\_FIEN: USBDET GPIO Falling Interrupt Enable

0: Disable

1: Enable

USBDET\_RI: USBDET GPIO Rising Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

USBDET\_FI: USBDET GPIO Falling Interrupt Event

Read 1: Interrupt Event Occurs

Read 0: No Interrupt Event

Write 1: Clear Interrupt Event

Write 0: No Action

#### 19.3.38 P\_GPIO\_NFSDSPI\_EN(0x882000A4)

NAME	D16	D15-D9	D8	D7-D1	D0
P_GPIO_NFSDSPI_EN	SD_EN		SPI_EN		NFLASH_EN

NFLASH\_EN: NFLASH Enable

0: Disable

1: Enable

SPI\_EN: SPI Enable

0: Disable

1: Enable

SD\_EN: SDCard Enable

0: Disable

1: Enable

#### 19.3.39 P\_GPIO\_LVD\_CTL(0x882000A8)

NAME	D25-D24	D23-D17	D16	D15-D10	D9-D8	D7-D1	D0
P_GPIO_LVD_CTL	LVRSEL		LVREN		LVDSEL		LV DEN

LV DEN: LVD Enable

0: Disable

1: Enable

LVDSEL: LVD Selection

0: VDD18<1.5V, LVD high

1: VDD1.8<1.6V, LVD high

2: VDD1.8<1.7V, LVD high

3: VDD1.8<1.8V, LVD high

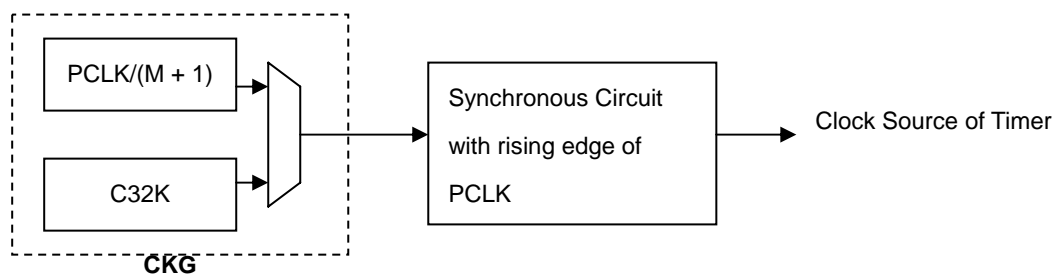
LVREN:	LVR Enable
	0: Disable
	1: Enable
LVRSEL:	LVR Selection

## 20 TIMER

### 20.1 Features

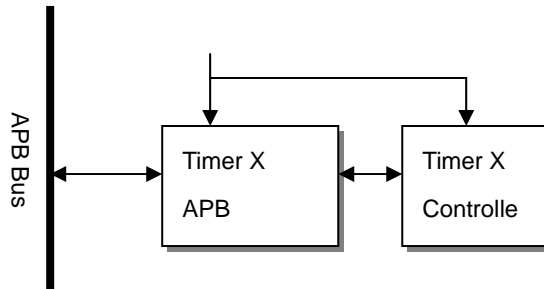
SPG290 contains six 16-bit timer/counters. All timers support Capture/Comparison/PWM (CCP) functions when cooperates with their own two 16-bit registers (preload register and ccp register). The clock source of these six timers can be programmed to internal clock (PCLK/2(max 13.5MHz) or C32K). Detail

- 1). PCLK/2 base and C32K base clock source selection for each timer source
- 2). Each Programmable clock must be synchronous with PCLK as for each timer
- 3). Each timer support Capture function / Comparison function / PWM function



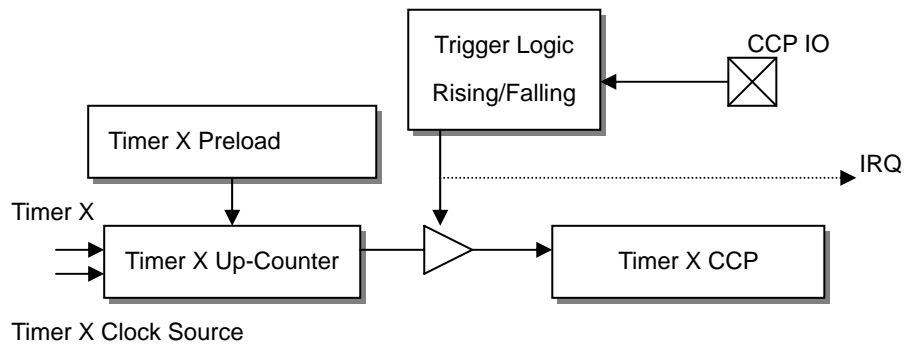
$M = 1 \sim 255$ , and  $M = 0$  is illegal

### 20.2 Architecture

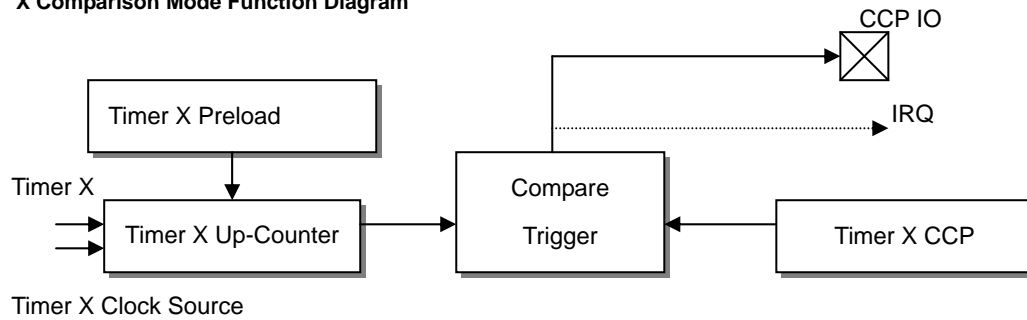


$X = 0 \sim 5$  (Timer 0 ~ Timer 5)

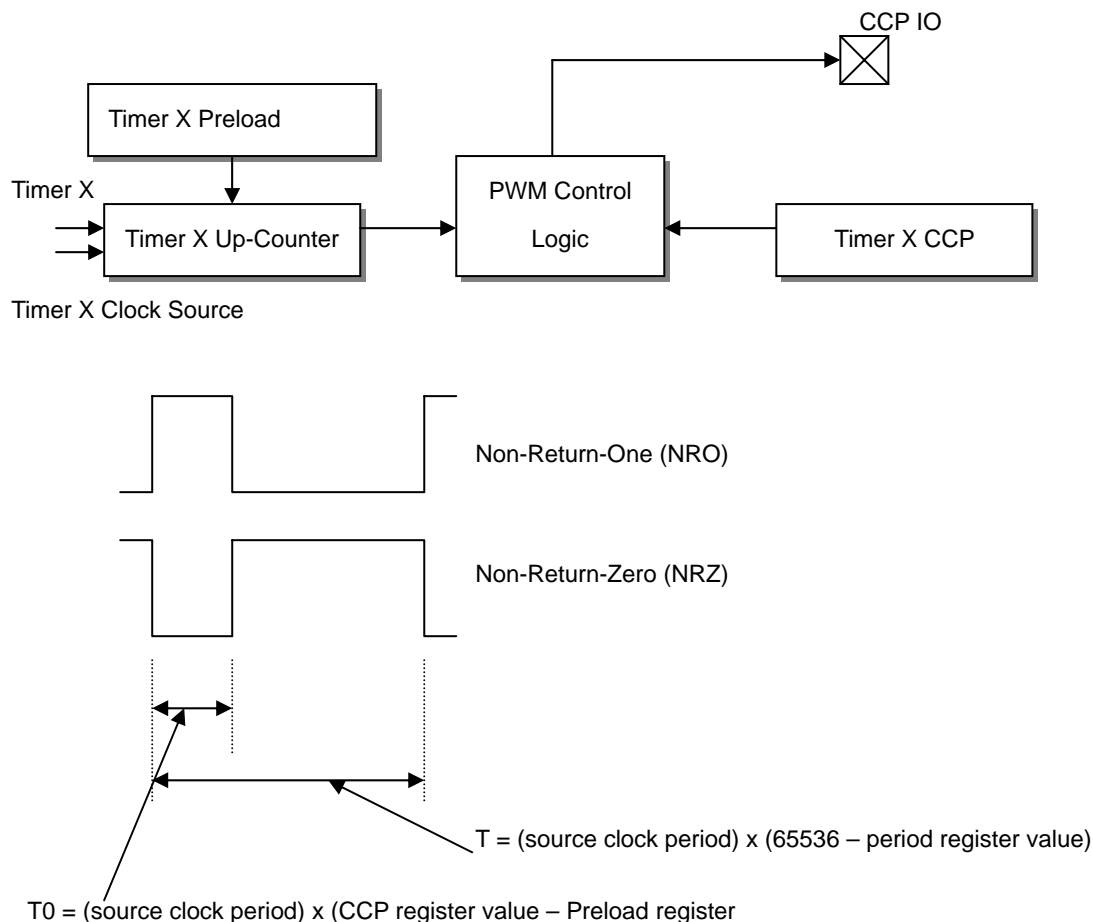
### X Capture Mode Function Diagram



### X Comparison Mode Function Diagram



X PWM Mode Function Diagram



## 20.3 Control Registers

### 20.3.1 P\_TimerXCtrl(0x8816X000)

NAME	D31	D30-D28	D27	D26	D25-D0
P_TimerXCtrl	TIMER_X_EN		TIMER_X_IRQ_EN	TIMER_X_IRQ_FLAG	

TIMER\_X\_IRQ\_FLAG: Timer X Interrupt Flag

0: Disable

1: Enable

TIMER\_X\_IRQ\_EN: Timer X Interrupt Enable

0: Disable

1: Enable

TIMER\_X\_EN: Timer X Enable

Read 0: Non Happen

Read 1: IRQ occurred

Write 1: Clear

### 20.3.2 P\_CPP\_CTRL(0x8816X004)

NAME	D31-D30	D29-D28	D27	D26	D25	D24-D0
P_CPP_CTRL	CCP_MODE		CAP_MODE	COM_MODE	PWM_MODE	

PWM\_MODE: PWM Mode Selection

0: NRO output

1: NRZ output

COM\_MODE: Comparison Mode Selection

0: high pulse

1: low pulse

CAP\_MODE: Capture Mode Selection

0: every falling

1: every rising

CCP\_MODE: CCP Mode Selection

00: Timer Mode

01: Capture Mode

10: Comparison Mode

11: PWM Mode

### 20.3.3 P\_Preload\_REGS(0x8816X008)

NAME	D31-D0
P_Preload_REGS	TIMER_X_PRELOAD_REGS

### 20.3.4 P\_CPP\_REGS(0x8816X00C)

NAME	D31-D0
P_CPP_REGS	TIMER_X_CCP_REGS

### 20.3.5 P\_CPP\_UPCOUNT(0x8816X010)

NAME	D31-D0
P_CPP_UPCOUNT	TIMER_X_UPCOUNT

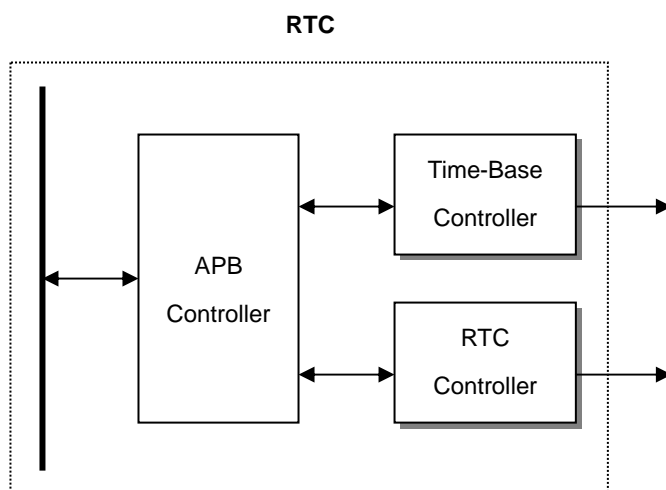
## 21 REAL TIME CLOCK - RTC

### 21.1 Features

The Real Time Clock (RTC) unit is operated in 32768Hz clock domain. The data includes second, minute, hour, date, and alarm. The system also supports a time-base function which is based on 32768Hz.

- Support half-second, one second, one minute, one hour interrupt request
- Support alarm function interrupt request
- Support 3 Time-Base Functions

### 21.2 Architecture



### 21.3 Control Registers.

#### 21.3.1 P\_RTC\_SEC(0x88166000): RTC Second Setup Registers.

NAME	D5 -D0
P_RTC_SEC	SECOND_SET

#### 21.3.2 P\_RTC\_MIN(0x88166004): RTC Minute Setup Registers.

NAME	D5 -D0
P_RTC_MIN	MINUTE_SET

#### 21.3.3 P\_RTC\_HOUR(0x88166008): RTC Hour Setup Registers.

NAME	D4 -D0
P_RTC_HOUR	HOUR_SET

#### 21.3.4 P\_ALM\_SEC(0x8816600C): Alarm Second Setup Registers.

NAME	D5 -D0
P_ALM_SEC	SECOND_SET

#### 21.3.5 P\_ALM\_MIN(0x88166010): Alarm Minute Setup Registers.

NAME	D5 -D0
P_ALM_MIN	MINUTE_SET

#### 21.3.6 P\_ALM\_HOUR(0x88166014): Alarm Hour Setup Registers.

NAME	D4 -D0
P_ALM_HOUR	HOUR_SET

#### 21.3.7 P\_RTC\_CTRL(0x88166018): RTC Enable Registers.

NAME	D31
P_ALM_HOUR	RTC_EN

RTC\_EN: RTC Enable

0: Disable

1: Enable

#### 21.3.8 P\_RTC\_STATUS(0x8816601C): RTC Status Registers.

NAME	D22	D21-D20	D19	D18	D17-D16	D15	D14
P_ALM_HOUR	MIN_INTEN		SEC_INT	SEC_INTEN		HSEC_INT	HSEC_INTEN
NAME	D31	D30	D29-D28	D27	D26	D25-D24	D23
P_ALM_HOUR	ALM_INT	ALM_INTEN		HOUR_INT	HOUR_INTEN		MIN_INT

HSEC\_INTEN: Half-Second Interrupt Enable

0: Disable

1: Enable

HSEC\_INT: Half-Second Interrupt Flag

Write 1: Clear the flag

SEC\_INTEN: Second Interrupt Enable

0: Disable

1: Enable

SEC\_INT: Second Interrupt Flag

Write 1: Clear the flag

MIN\_INTEN: Minute Interrupt Enable

0: Disable

1: Enable

MIN\_INT: Minute Interrupt Flag

Write 1: Clear the flag

HOUR\_INTEN: Hour Interrupt Enable  
                   0:           Disable  
                   1:           Enable  
  
 HOUR\_INT: Hour Interrupt Flag  
                   Write 1:   Clear the flag  
  
 ALM\_INTEN: Alarm Interrupt Enable  
                   0:           Disable  
                   1:           Enable  
  
 ALM\_INT: Alarm Interrupt Flag  
                   Write 1:   Clear the flag

### 21.3.9 P\_TMB\_Ctrl(0x88166020): Time Base Control Registers.

NAME	D31	D30-D28	D27-D24	D23	D22-D20	D19-D16	D15	D14-D12	D11-D8
P_TMB_CTRL	TMB0_EN		TMB0_SEL	TMB1_EN		TMB1_SEL	TMB2_EN		TMB2_SEL

TMB2\_SEL: Time-Base 2 Period Selection

0000: 1Hz  
 0001: 2Hz  
 0010: 4Hz  
 0011: 8Hz  
 0100: 16Hz  
 0101: 32Hz  
 0110: 64Hz  
 0111: 128Hz  
 1000: 256Hz  
 1001: 512Hz  
 1010: 1024Hz  
 1011: 2048Hz

TMB2\_EN: Time-Base 2 Enable

0:       Disable  
 1:       Enable

TMB1\_SEL: Time-Base 2 Period Selection

0000: 1Hz  
 0001: 2Hz  
 0010: 4Hz  
 0011: 8Hz  
 0100: 16Hz  
 0101: 32Hz

0110: 64Hz  
0111: 128Hz  
1000: 256Hz  
1001: 512Hz  
1010: 1024Hz  
1011: 2048Hz

TMB1\_EN: Time-Base 2 Enable  
0: Disable  
1: Enable

TMB0\_SEL: Time-Base 2 Period Selection  
0000: 1Hz  
0001: 2Hz  
0010: 4Hz  
0011: 8Hz  
0100: 16Hz  
0101: 32Hz  
0110: 64Hz  
0111: 128Hz  
1000: 256Hz  
1001: 512Hz  
1010: 1024Hz  
1011: 2048Hz

TMB0\_EN: Time-Base 2 Enable  
0: Disable  
1: Enable

#### 21.3.10 P\_TMB\_Status(0x88166024): Time Base Status Registers.

NAME	D31	D30	D29-D28	D27	D26	D25-D24	D23	D22	D21-D0
P_TMB_Status	TMB0_IN T	TMB0_IN TEN		TMB1_IN T	TMB1_IN TEN		TMB2_IN T	TMB2_IN TEN	

TMB2\_INTEN: Time-Base 2 Interrupt Enable  
0: Disable  
1: Enable

TMB2\_INT: Time-Base 2 Interrupt Flag  
Write 1: Clear the flag

TMB1\_INTEN: Time-Base 1 Interrupt Enable  
0: Disable

1: Enable

TMB1\_INT: Time-Base 1 Interrupt Flag

Write 1: Clear the flag

TMB0\_INTEN: Time-Base 0 Interrupt Enable

0: Disable

1: Enable

TMB0\_INT: Time-Base 0 Interrupt Flag

Write 1: Clear the flag

#### 21.3.11 P\_TMB\_Reset(0x88166028): Time Base Reset Registers.

NAME	D31-D0
P_TMB_Reset	RESET_COM

RESET\_COM: Time-Base Reset Command

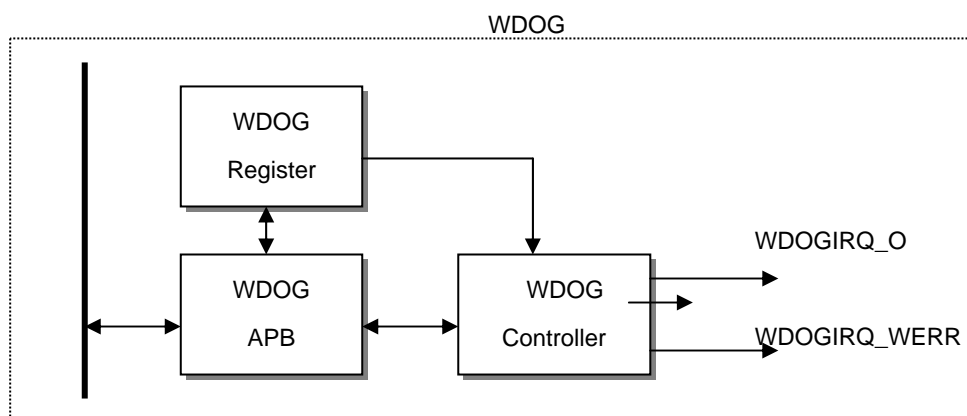
Write 0x5xxxxxx5 to reset Time-Base Counter.

## 22 WATCH DOG

### 22.1 Features

In order to avoid system crash, the watch-dog controller is always necessary. When watch-dog system is enable, we need to clear its counter per a period. And a period can be programmed by control registers. If the watch-dog counter is not cleared in a specific period, the system will be reset.

### 22.2 Architecture



### 22.3 Control Registers.

#### 22.3.1 P\_WDOG\_CTRL(0x88170000): Watch Dog Control Registers.

NAME	D31
P_WDOG_CTRL	WDog_EN

WDog\_EN: Watch-Dog Enable  
0: Disable  
1: Enable

#### 22.3.2 P\_WDOG\_CYCLE(0x88170004): Reset Counter Number

NAME	D31-D0
P_WDOG_CYCLE	RESET_COM

RESET\_COM: Reset Counter Number

#### 22.3.3 P\_WDOG\_CLEAR(0x88170008): Reset Counter Number

NAME	D31-D0
P_WDOG_CLEAR	CLEAR_COM

CLEAR\_COM: Clear Command

Write 0xaxxxxx5 to clear, write other values will reset system

## 23 SLEEP/WAKEUP

### 23.1 Sleep

After power on reset, CPU starts working until a sleep command is given. When a sleep signal is accepted, IC will turn off system clock (PLL) and enter sleep mode. After entering sleep, program counter will stop at the next address and will start to execute once wakeup activates.

According to the ON/OFF state of PLL, two sleep modes are classified and listed below:

- Wait Mode: The PLL is NOT turned off in this mode.
- Half Mode: The PLL is turned off in this mode.

The control register for entering sleep modes is 0X88210000, please see section about the PLL/CKG.

### 23.2 WakeUp

Waking up from sleep mode requires a wakeup signal to turn on the system clock (PLL). At the same time, a wakeup IRQ is also generated. The IRQ signal leads CPU to complete the wakeup process as well as initialization. After wakeup interrupt completed, program counter will continue to execute the next command.

In each sleep mode, the wakeup source can be key change signals or interrupts from any modules whose clock is not turned off. All wakeup sources are listed in the following table.

Wakeup sources	Description
SPU Fast INT	SPU fast Interrupt. Please refer to SPU for more detail.
SPU Beat	SPU beat Interrupt. Please refer to SPU for more detail.
SPU ENV	SPU envelope Interrupt. Please refer to SPU for more detail.
CD SERVO	DMA done from CD to DRAM.
MIC INT	ADC Microphone.
ADC INT	ADC General purpose Interrupt.
TB INT	Time Base Interrupt.
Timer INT	Timer1 or Timer2 or ... Timer 6 event trigger.
TVE INT1	TVE vertical blanking start.
LCD INT	LCD vertical blanking start.
PPU INT1	PPU vertical blanking start.
TVE INT3	TVE light gun hit.
CSI INT1	Sensor frame end.
CSI INT3	Sensor coordinate hit.
CSI INT2	Sensor motion frame end.
CSI INT0	Capture done.
TVE INT4	TVE coordinate hit.
PPU INT3	PPU coordinate hit.
USB INT	USB device or host Interrupt. Please refer to USB for more detail.
SIO INT	SIO Interrupt.
SPI INT	SPI Interrupt.

UART INT	UART Interrupt.
Flash INT	Nand-type Flash Interrupt.
SD INT	SD card Interrupt.
I2C INT	I2C master Interrupt.
I2S INT	I2S master or I2S slave Interrupt.
APBDMA INT1	APBDMA channel 1 done.
APBDMA INT2	APBDMA channel 2 done.
LDMDMA INT	LDM DMA done. (LDM to DRAM or DRAM to LDM)
BLN Int	Blending DMA done. (DRAM to DRAM)
APBDMA INT3	APBDMA channel 3 done.
APBDMA INT4	APBDMA channel 4 done.
RTC INT	Alarm or HMS. (per hour or per minute or per second event)
MPG INT	MPEG4 encoding or decoding a frame has done.
ECC INT	The event that ECC has found errors.
SFTCFG INT	For the rising or falling event of assigning PIN. Please see SFTCFG for more detail.
DEBUG INT	TVE vertical blanking start or PPU vertical blanking start.
KEYCHG INT	Key change Interrupt. See PINMUX section to know what pins can be used to generate key change signals.
LVD INT	Low voltage detecting Interrupt.

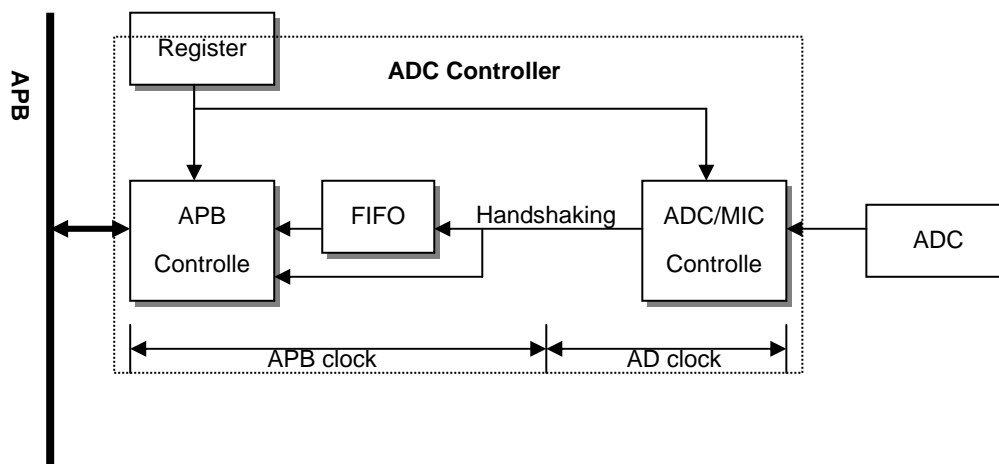
## 24 ANALOG TO DIGITAL CONVERTER - ADC

### 24.1 Features

The analog-to-digital converter provides voice record or general-purpose ad functions. There are total nine channels with one channel dedicated to microphone mode and other channels dedicated to 12-bit general-purpose ADC. The microphone mode ADC doesn't support auto gain control(AGC), therefore we need additional software to support it. Different from other general ADC channels, microphone channel data can only use DMA to transfer instead of the interrupt mode. The general-purpose mode support auto data by sampled at programmable sampling rate or manual data by sampled at sampling command. All general-purpose ADC only support interrupt mode to read data. ADC spec. is as follows:

- 1). General-purpose application: 2.0mA
- 2). Audio application without MICBIAS: 4.0mA / with MICBIAS: 7.5mA
- 3). Power down mode: 1uA
- 4). Nine channels 12-bit resolution for audio and general-purpose ADC application
- 5). On-chip microphone boost amplifier and programmable gain amplifier(33, 31.5, ..., -12,  $-\infty$  db)
- 6). On-chip anti-aliasing filter and bias-voltage output(MICBIAS)
- 7). Reference top voltage and bias current/voltage generators

### 24.2 Architecture



## 24.3 Control Registers

### 24.3.1 P\_ADC\_SETUP(0x881A0000): ADC Setup

NAME	D31	D30	D29	D28	D27	D26	D25	D24-D4	D3-D0
P_ADC_SETUP	ADC_S EL	ADC_E N	MIC_E N		MIC_B OOST	MIC_BI AS	TOP_V OL		CH_SE L

CH\_SEL:

Channel Select

0000: GPIO 0

0001: GPIO 1

0010: GPIO 2

0011: GPIO 3

0100: GPIO 4

0101: GPIO 5

0110: GPIO 6

0111: GPIO 7

1xxx: MIC

TOP\_VOL:

Top reference voltage internal or external

0: Internal

1: External

MIC\_BIAS:

MIC bias enable(only for MIC mode)

0: Disable

1: Enable

MIC\_BOOST:

MIC boost enable (only for MIC mode)

0: Disable

1: Enable

MIC\_EN:

MIC enable

0: Disable

1: Enable

ADC\_EN:

ADC enable

0: Disable

1: Enable

ADC\_SEL:

ADC Selection

0: Disable

1: Enable

### 24.3.2 P\_MIC\_GAIN(0x881A0004): MIC GAIN

NAME	D4-D0
P_MIC_GAIN	MIC_GAIN

MIC\_GAIN: MIC Gain value

00000: 33.0 db

00001: 31.5 db

00010: 30.0 db

.....

10101: 1.5 db

10110: 0.0 db

10111: -1.5 db

....

11110: -12.0 db

11111:  $-\infty$

### 24.3.3 P\_ADC\_CLOCK\_SET(0x881A0008): ADC Clock Setup

NAME	D15-D0
P_ADC_CLOCK_SET	ADC_CLOCK_CYCLE

ADC\_CLOCK\_CYCLE:  $[(15:0) + 1]$  cycles ACC

### 24.3.4 P\_ADC\_HOLD\_SETUP(0x881A000C): Sample Hold Setup

NAME	D31-D28	D27-D16	D15-D0
P_ADC_HOLD_SETUP	HOLD_WIDTH		HOLD_CYCLE

HOLD\_CYCLE:  $[(15:0) + 1]$  cycles Sample Hold Cycle must  $\geq 15$

HOLD\_WIDTH:  $[3:0]$  width Sample Hold Width must  $\geq 2$

**NOTE:** The method about setup ADC clock cycle and sample-hold width / cycle

If we need 44.1K sample rate for ADC and ADC controller clock = 33.8688 MHz

Sample cycle =  $33.8688\text{M} / 44.1\text{K} = 768$ , and we need Sample Hold cycle = 16.

Finally, ADC clock cycle =  $768 / 16 = 48$

So ACC = 47, SampleHoldCycle = 15, and SampleHoldWidth = 2

### 24.3.5 P\_ADC\_MIC\_CTRL(0x881A0010):

NAME	D23	D22	D21-D7	D6-D4	D3	D2-D0
P_ADC_MIC_CTRL	FIFO_OEN	AUTO_CLR		FIFO_INT		FIFO_RD
NAME	D31	D30-D28	D27	D26	D25	D24
P_ADC_MIC_CTRL	ADC_EN		AUTO_SAMPLE	MIC_MODE	MUTE_SEL	DMA_SIZE

ADC\_EN: ADC Controller enable

0: Disable

	1: Enable
AUTO_SAMPLE:	Auto Sampling for general-purpose mode
	0: manual
	1: auto
MIC_MODE:	Data Unsigned/Signed select for MIC Mode
	0: signed
	1: unsigned
MUTE_SEL:	Mute Select for MIC Mode
	0: normal
	1: Mute
DMA_SIZE:	DMA data size for MIC mode
	0: 32 bits
	1: 16 bits
FIFO_OEN:	FIFO overwrite Enable
	0: disable
	1: Enable
AUTO_CLR:	Auto clear irq flag for general-purpose mode
	0: disable
	1: Enable
FIFO_INT:	FIFO receive depth for Interrupt Level
	0: one receive will irq
	1: two receive will irq
	2: three receive will irq
	3: four receive will irq
	4: five receive will irq
	5: six receive will irq
	6: seven receive will irq
	7: eight receive will irq
FIFO_RD:	FIFO receive depth read

#### 24.3.6 P\_ADC\_MIC\_CTRL2(0x881A0014):

NAME	D25-D24	D23	D22	D21	D20-D16	D15
P_ADC_MIC_CTRL		FIFO_OFE	FIFO_FULL	FIFO_EMPTY		CONVERT_ADC
NAME	D31	D30	D29	D28	D27	D26
P_ADC_MIC_CTRL	ADC_INT	ADC_INTEN	ADC_AINT	ADC_AINTEN	MIC_OFINT	MIC_OFINTEN

ADC\_INT: ADC Manual Interrupt Flag  
Read 0: Occurred

	Write 1: Clear the flag
ADC_INTEN:	ADC Manual Interrupt Enable
	0: Disable
	1: Enable
ADC_AINT:	ADC Auto Interrupt Flag
	Read 0: Occurred
	Write 1: Clear the flag
ADC_AINTEN:	ADC Auto Interrupt Enable
	0: Disable
	1: Enable
MIC_OFINT:	MIC Overflow Interrupt Flag
	Read 0: Occurred
	Write 1: Clear the flag
MIC_OFINTEN:	MIC Overflow Interrupt Enable
	0: Disable
	1: Enable
FIFO_OFE:	FIFO receiver overflow Error
	Read 0: Occurred
	Write 1: Clear the flag
FIFO_FULL:	FIFO Full Flag
	0: Not Full
	1: Full
FIFO_EMPTY:	FIFO Empty Flag
	0: Not Empty
	1: Empty
CONVERT_ADC:	Start Convert for ADC manual mode
	Write 1: Start Convert

#### 24.3.7 P\_ADC\_Read\_Data(0x881A0018):

NAME	D31-D20
P_ADC_Read_Data	ADC_Self_Data

#### 24.3.8 P\_ASP\_Read\_Data(0x881A001C):

NAME	D31-D20
P_ASP_Read_Data	ADC_Auto_Data

## 24.3.9 P\_MIC\_Read\_Data(0x881A0020):

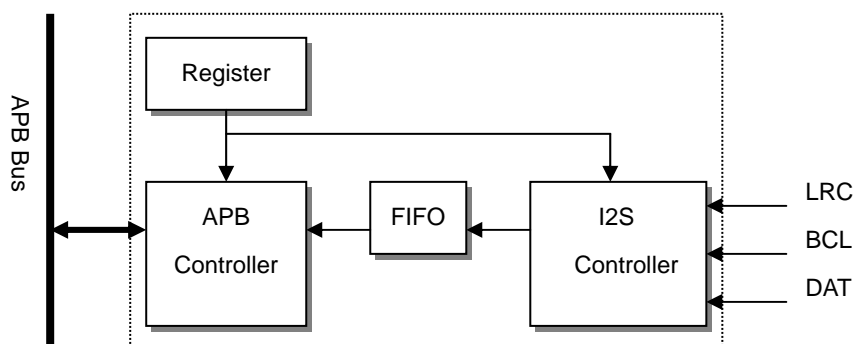
NAME	D31-D0
P_MIC_Read_Data	MIC_Data

## 25 INTERACTIVE INFORMATION SERVICES - I2S

### 25.1 Features

I<sup>2</sup>S is a protocol for digital stereo audio. The I<sup>2</sup>S controller embedded in SPG290 supports the receive function in the slave mode and maximum receive data size is 32-bit for left/right channel.

### 25.2 Architecture



### 25.3 Control Registers

#### 25.3.1 P\_I2S\_CR(R/W)(0x88140000): I2S control

NAME	D31	D30-D28	D27	D26	D25	D24	D23	D22	D21	D20	D19-D0
P_I2S_CR	RX_EN		LJM	LCE	RCE	USB	LRF	DMA	FIFO	AUTO	

AUTO: Auto Interrupt flag clear for IRQ mode

0: Disable

1: Enable

FIFO: FIFO receive overwrite enable

0: Disable

1: Enable

DMA: DMA Enable

0: IRQ mode

1: DMA mode

LRF: Left / Right first receive select

0: Left

1: Right

USB: USB Mode

0: Disable

1: Enable

RCE: Right channel enable

0: Disable

1: Enable

LCE: Left channel enable

0: Disable  
 1: Enable

LJM: I2S / Left justified mode select

0: I2S mode  
 1: Left mode

RX\_EN: I2S Receiver Enable

0: Disable  
 1: Enable

### 25.3.2 P\_I2S\_IRQ\_STS(R/W)(0x88140004): I2S IRQ Status

NAME	D31	D30	D29-D0
<b>P_I2S_IRQ_STS</b>	INT_FLAG	INT_EN	

INT\_EN: I2S interrupt enable

0: Disable  
 1: Enable

INT\_FLAG: I2S interrupt flag

Read 1: IRQ occurred  
 Write 1: Clear IRQ

### 25.3.3 P\_I2S\_FIFO(R/W)(0x88140008): I2S FIFO Config

NAME	D31	D30	D29	D28-D6	D5-D4	D3-D2	D1-D0
<b>P_I2S_FIFO</b>	OV_ERR	FULL_FLAG	EMPTY_FLAG		RX_INT_LV		RX_LV

RX\_LV: FIFO receiver level

RX\_INT\_LV: FIFO receive interrupt level

00: one receive will irq  
 01: two receive will irq  
 10: three receive will irq  
 11: four receive will irq

EMPTY\_FLAG: FIFO empty flag

0: not empty  
 1: empty

FULL\_FLAG: FIFO full flag

0: not full  
 1: full

OV\_ERR: FIFO receive overflow error

Read 1: Error occurred  
 Write 1: Clear

**25.3.4 P\_I2S\_RDATA(R/W)(0x8814000C): I2S receive data**

NAME	D31-D0
P_I2S_RDATA	I2S_RDATA

## 26 INTER-INTEGRATED CIRCUIT - I2C

### 26.1 Overview

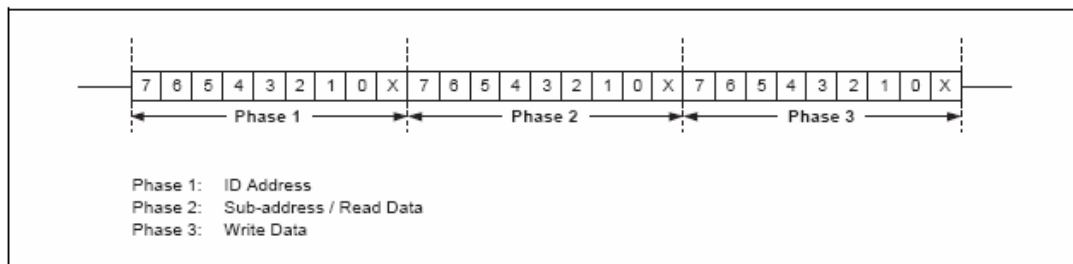
I<sup>2</sup>C is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange, minimizing the interconnection between devices. This bus is suitable for applications requiring occasional communications over a short distance between many devices. The flexible I<sup>2</sup>C allows additional devices to be connected to the bus for expansion and system development.

### 26.2 Initialization

Before starting I2C, initialization must be done. There are 2 major jobs:

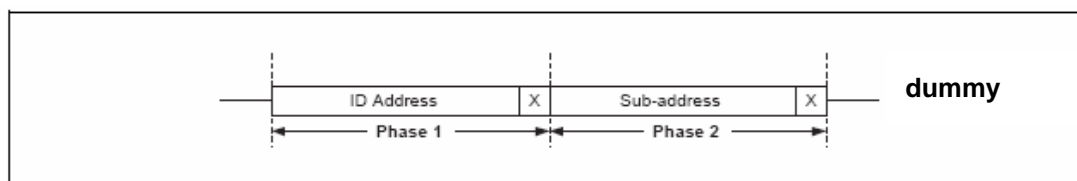
- (1) Decide the I2C clock rate by configuring I2CCVR
- (2) Decide the I2C slave ID Address by configuring I2CID

The SPG290 support I2C master only and 3 types of transaction : 8 bits type, 16 bits type and 8 bits \* n type. For 8 bits type: by writing I2CCR[0] to start the transaction and by reading I2CCR[3] to watch the result (if INTEN is 1, I2C interrupt happens after a full transaction). Here is an example for I2C 8 bits writing:

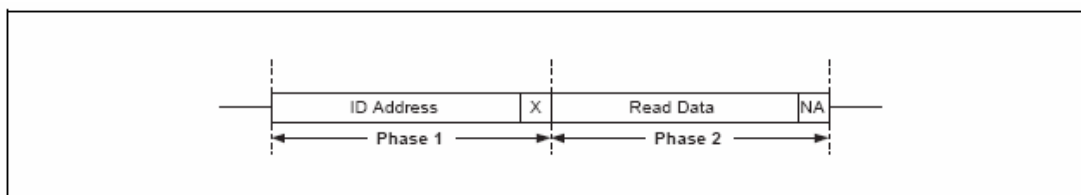


0<sup>th</sup> bit in ID Address must be 0

Besides, here is an example for I2C 8 bits reading:

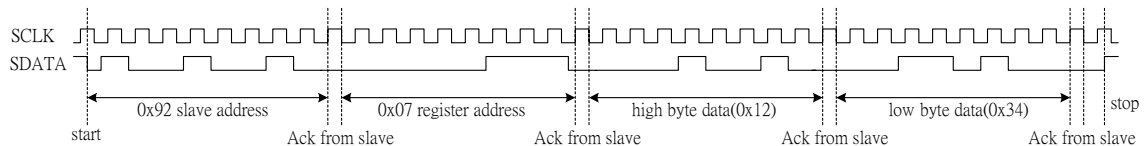


0<sup>th</sup> bit in ID Address must be 0

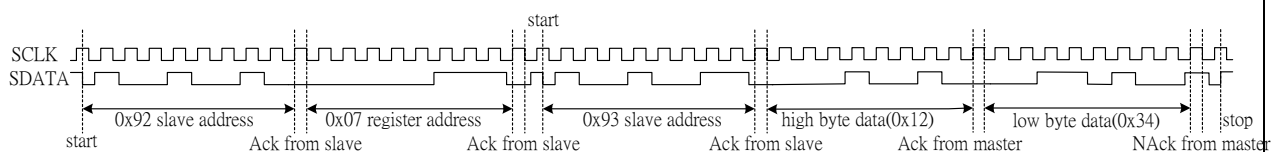


0<sup>th</sup> bit in ID Address must be 1

For 16 bits type : by writing I2CCR[1] to start the transaction and by reading I2CCR[4] to watch the result (if INTEN is 1, I2C interrupt happens after a full transaction). Here is an example for I2C 16 bits writing:



Here is an example for I2C 16 bits reading:



For 8 bits \* n type transaction : each transaction is looked as each phase transaction. The definition of “phase transaction” is presented as above. By writing I2CCR[2] to start each phase transaction and by reading I2CCR[5] to watch the result (if INTEN is 1, I2C interrupt happened after each phase transaction). In addition, by writing I2CCR[8] to stop the full transaction.

## 26.3 Control Registers

### 26.3.1 P\_I2C\_CR(R/W)(0x88130020): I2C configuration register.

NAME	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_I2C_CR	STOPN	GAK	TRANS_MODE	ACKN	ACK16	ACK8	STARTN	START16	START8

START8:	Start 8 bits transfer
START16:	Start 16 bits transfer
STARTN:	Start general transaction
ACK8:	ACK from 8 bits transfer
ACK16:	ACK from 16 bits transfer
ACKN:	ACK from general transaction
TRANS_MODE:	Read/ Write Mode selection
	0: Write Mode
	1: Read Mode
GAK:	ACK/NAK mode select
	0: Generate NAK at last transaction
	1: Generate ACK at last transaction
STOPN:	Stop general transaction

### 26.3.2 P\_I2C\_INTR(R/W)(0x88130024): I2C interrupt register.

NAME	D1	D0
P_I2C_INTR	INT_EN	INT_FLAG

INT\_FLAG: Interrupt flag

INT\_EN: Interrupt enable bit

### 26.3.3 P\_I2C\_CVR(R/W)(0x88130028): I2C clock setting register.

NAME	D9-D0
P_I2C_CVR	CLK_SET

### 26.3.4 P\_I2C\_ID(R/W)(0x8813002C): I2C ID register.

NAME	D7-D1
P_I2C_ID	ID_REG

### 26.3.5 P\_I2C\_ADDR(R/W)(0x88130030): I2C port address register.

NAME	D7-D0
P_I2C_ADDR	ADDR

### 26.3.6 P\_I2C\_WDATA(R/W)(0x88130034): I2C write data register.

NAME	D15-D0
P_I2C_WDATA	I2C_WDATA

### 26.3.7 P\_I2C\_RDATA(R/W)(0x88130038): I2C read data register.

NAME	D15-D0
P_I2C_RDATA	I2C_RDATA

---

---

## 27 SERIAL PERIPHERAL INTERFACE - SPI

---

---

### 27.1 General Description

A Serial Peripheral Interface (SPI) controller is built in SPG290 to facilitate communicating with other devices and components.

The SPI performs parallel-to-serial conversion on data written to an internal 8-bit wide transmit buffer. It also performs serial-to-parallel conversion on the serial input data, and buffers it in a receiving 8-bits wide buffer. The SPI asserts interrupts to request service to the transmitting buffer and to the receiving buffer, and to indicate an overrun condition in the receiving buffer.

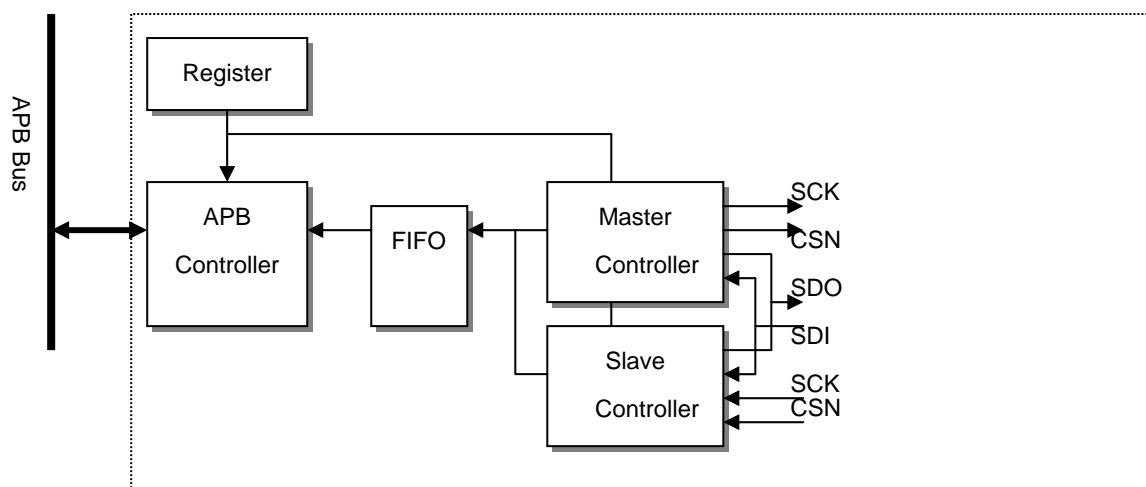
The SPI includes a programmable bit rate clock divider and pre-scalar to generate the serial output clock SCLK from the internal clock. The operating mode, frame format and size are programmable using control register P\_SPI\_Ctrl. The SSB output operates as an active LOW slave selection for SPI and Micro wire.

### 27.2 Features

There are four control signals on SPI, including SPICSN, SPICLK, SPIRX and SPITX. Main features of SPI include:

- 1). Support Master / Slave mode for single byte and consecutive bytes transferring.
- 2). Support overrun error indication
- 3). Support transmitting / receiving interrupt request
- 4). Programmable phase and polarity of master clock
- 5). Selectable data sampling time
- 6). Programmable master SCK clock frequency (System clock /2, /4, /8, /16, /32, /64, /128)
- 7). Built-in 8-depth 8bits FIFO in both transmit and receive direction, the interrupt level of those two FIFOs is programmable.

## 27.3 Architecture



## 27.4 Control Registers.

### 27.4.1 P\_SPI\_Control(0x88110000)

NAME	D31	D30-D28	D27	D26	D25	D24-D8	D7	D6	D5-D3	D2-D0
P_SPI_Control	SPI_EN		SPI_LB	SOFT_REST	SPI_MODE		SPI_CK_PH	SPI_CK_PO		SPI_CLOCK

SPI\_CLOCK: Master Clock Select

- 000: PCLK/2
- 001: PCLK/4
- 010: PCLK/8
- 011: PCLK/16
- 100: PCLK/32
- 101: PCLK/64
- 110: PCLK/128

SPI\_CK\_PO: Refer to timing diagram

SPI\_CK\_PH: Refer to timing diagram

SPI\_MODE: SPI Master/Slave Mode Select

- 0: Master
- 1: Slave

SOFT\_REST: SPI Software Reset

- 0: No effect
- 1: Reset

SOFT\_LB: SPI Loop Back Mode Select(Master)

- 0: Normal
- 1: SPIRX= SPITX

SOFT\_EN: SPI Enable

0: Disable  
1: Enable

#### 27.4.2 P\_SPI\_TX\_Status (0x88110004)

NAME	D31	D30	D29-D28	D27	D26-D7	D6-D4	D3	D2-D0
P_SPI_TX_Status	TX_FLAG	TX_EN		TX_EMPTY_FLAG		TX_FIFO_INT_LEVEL		FIFO_DATA_LEVEL

FIFO\_DATA\_LEVEL: SPI Tx FIFO Data Level

000: data number < 1, interrupt will happen  
001: data number < 2, interrupt will happen  
010: data number < 3, interrupt will happen  
011: data number < 4, interrupt will happen  
100: data number < 5, interrupt will happen  
101: data number < 6, interrupt will happen  
110: data number < 7, interrupt will happen  
111: data number < 8, interrupt will happen

TX\_EMPTY\_FLAG: SPI Tx FIFO Empty Flag

0: No-Empty  
1: Empty

TX\_EN: SPI Tx Interrupt Enable

0: Disable  
1: Enable

TX\_FLAG: SPI Tx Interrupt Flag

Read 0: No happen  
Read 1: Interrupt happened  
Write 1: Clear Flag

#### 27.4.3 P\_SPI\_TX\_DATA (0x88110008)

NAME	D7-D0
P_SPI_TX_DATA	SPI_TX_DATA

#### 27.4.4 P\_SPI\_RX\_Status (0x8811000C)

NAME	D31	D30	D29-D28	D27	D26	D25-D5	D6-D4	D3	D2-D0
P_SPI_RX_Status	R_FLAG	RX_EN		R_EMPTY_FLAG	RX_OR_ERR		R_FIFO_INT_LEVEL		FIFO_DATA_LEVEL

FIFO\_DATA\_LEVEL: SPI Rx FIFO Data Level

000: one data in FIFO, interrupt will happen  
001: two data in FIFO, interrupt will happen  
010: three data in FIFO, interrupt will happen  
011: four data in FIFO, interrupt will happen

100: five data in FIFO, interrupt will happen  
 101: six data in FIFO, interrupt will happen  
 110: seven data in FIFO, interrupt will happen  
 111: eight data in FIFO, interrupt will happen  
  
 RX\_OR\_ERR: SPI Rx overrun error  
 Read 0: no happen  
 Read 1: happened  
 Write 1: clear flag  
  
 RX\_FULL\_FLAG: SPI Rx FIFO Full Flag  
 0: No-Full  
 1: Full  
  
 RX\_EN: SPI Rx Interrupt Enable  
 0: Disable  
 1: Enable  
  
 Rx\_FLAG: SPI Rx Interrupt Flag  
 Read 0: No happen  
 Read 1: Interrupt happened  
 Write 1: Clear Flag

#### 27.4.5 P\_SPI\_RX\_DATA (0x88110008)

NAME	D7-D0
P_SPI_RX_DATA	SPI_RX_DATA

#### 27.4.6 P\_SPI\_RX\_Status (0x8811000C)

NAME	D31	D30	D29-D5	D4	D3	D2	D1	D0
P_SPI_RX_Status	SPI_OW_MODE	SPI_SMART		SPI_BUSY	RX_FULL	RX_EMPTY	TX_FULL	TX_EMPTY

TX\_EMPTY: SPI Transmit FIFO empty flag

0: No-Empty  
 1: Empty

TX\_FULL: SPI Transmit FIFO Full flag

0: full  
 1: not full

RX\_EMPTY: SPI receiver FIFO Empty flag

0: empty  
 1: no-empty

RX\_FULL: SPI receiver FIFO full flag

0: not full  
 1: full

SPI_BUSY:	SPI busy status
0:	Idle
1:	Busy
SPI_SMART:	SPI smart mode
0:	Manual Clear IRQ
1:	Auto Clear IRQ
SPI_OW_MODE:	SPI Overwrite mode
0:	skip data when OV
1:	overwrite data when OV

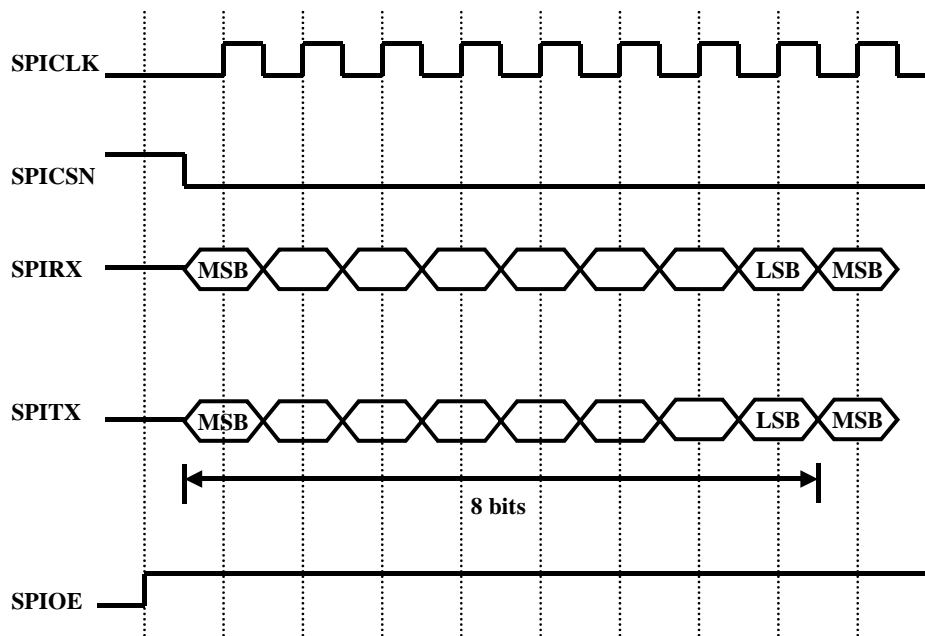


Fig1. Master Mode, SPO = 0, SPH=0

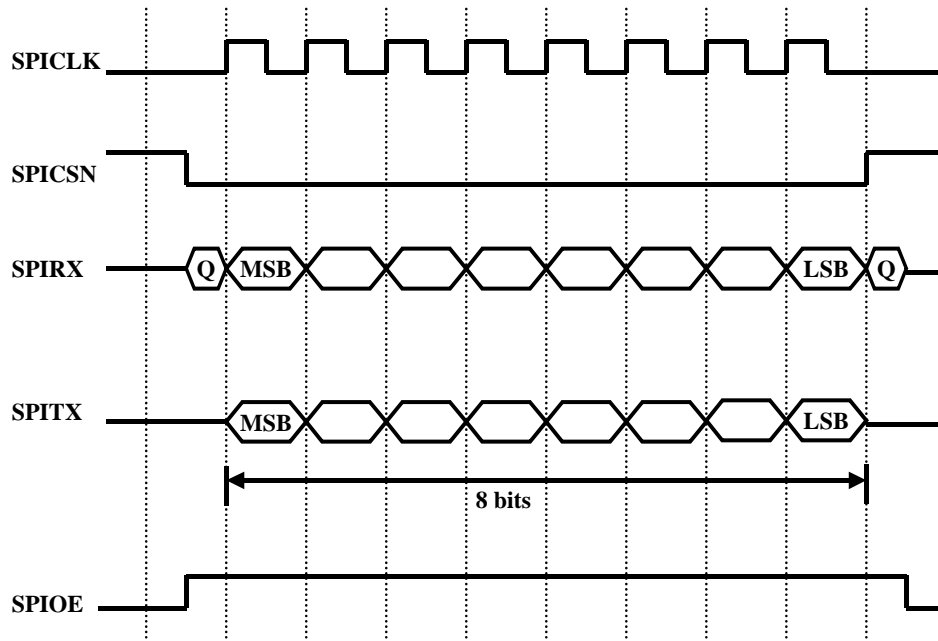


Fig2. Master Mode, SPO = 0, SPH=1

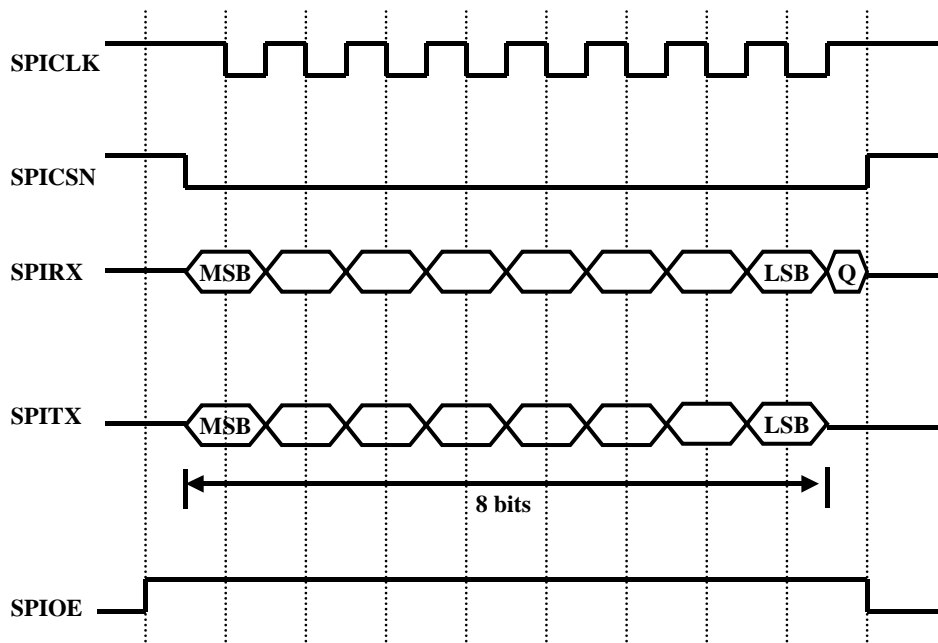


Fig3. Master Mode, SPO = 1, SPH=0

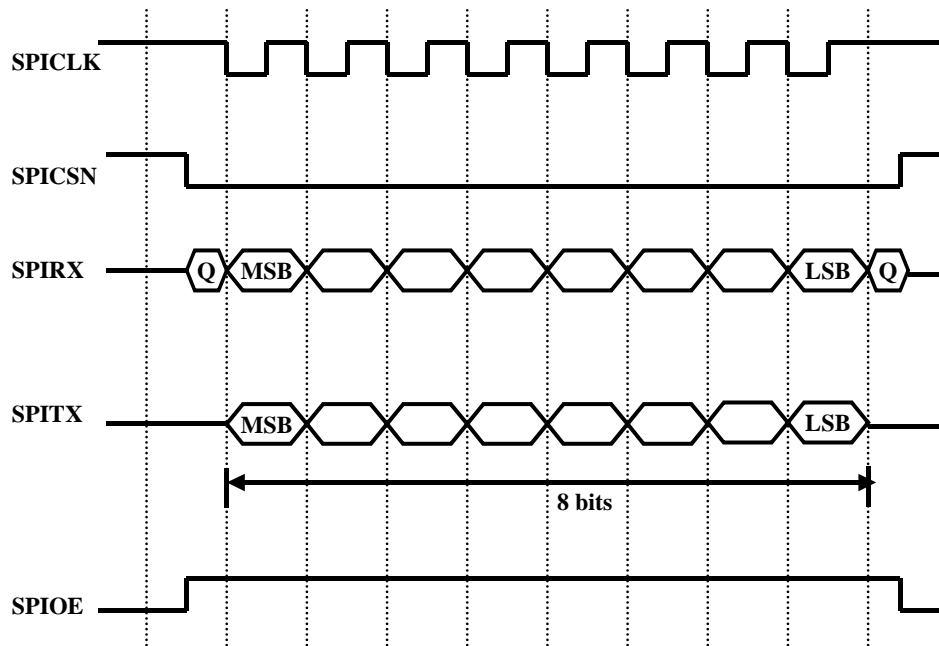


Fig4. Master Mode, SPO = 1, SPH=1

## 28 SERIAL INTERFACE I/O - SIO

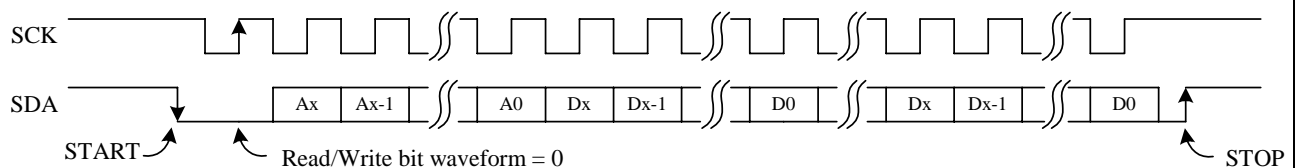
### 28.1 General Description

SIO (Serial Interface I/O) is Sunplus' s proprietary serial interface, which can be used to communicate with other devices. This serial interface is capable of transmitting or receiving data via 2 pins, SCK and SDA.

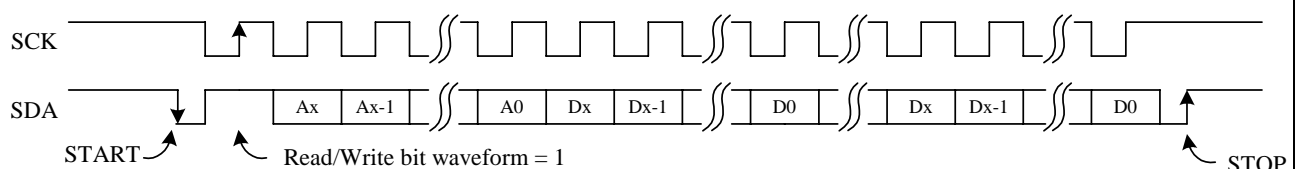
### 28.2 SIO Protocol

The SIO protocol is composed of start bit, read/write control bit, address word, data word, and stop bit. The figures below show the SIO protocol for the read and write mode respectively. If the SDA make a transition from 1 to 0 during the SCK is high, it stands for the “start bit” of the SIO transfer. If the SDA make a transition from 0 to 1 during the SCK is high, it means the “stop bit” occurred. In the SIO waveform other than the “start & stop bit” , the SDA can only change its logic level during the SCK is low. User should note that the address word & data word are transmitted with MSB first. And if the data word is 16 bits in width, then the **low** byte is transmitted/received first.

#### SIO Write Mode :



#### SIO Read Mode :



### 28.3 SIO Feature

The SIO interface provides the following features.

1. Support “master read” and “master write” function via SCK and SDA.
2. Support 4 address modes (No address, 8-bit address, 16-bit address, 24-bit address).
3. Support burst reading or writing function.
4. Support 8-bit and 16-bit data width.
5. Support interrupt and polling function.
6. Support 4 SIO baud rates.
7. Provide automatic bit-stream transmitting mode when communicating with SPDS301 IC.

## 28.4 SIO Control Registers

### 28.4.1 P\_SIO\_Control(0x88120000)

NAME	D7	D6-D3		D2	D1	D0
P_SIO_Control	SIO_IRQ_CLR			SIO_TRANSFER	SIO_RW_CTL	SIO_START
NAME	D15	D14	D13	D12	D11-D10	D9-D8
P_SIO_Control	SIO_DATA_WIDTH	APBDMA_EN	IRQ_EN	PROTOCOL	BAUDRATE	ADDR_MODE
NAME	D23-D16					
P_SIO_Control						
NAME	D31	D30	D29-D24			
P_SIO_Control	SIO_REQUEST	SIO_IRQ_STS				

SIO\_START: SIO Start/Stop control bit

- 0: Stop the SIO transmitting/receiving
- 1: Start the SIO transmitting/receiving

SIO\_RW\_CTL: SIO Read/Write control bit

- 0: SIO read mode
- 1: SIO write mode

SIO\_TRANSFER: Transfer data low byte first

- 0: SIO transfer Data Register High Portion(0x8812000D) before Data Register Low Portion(0x8812000C)
- 1: SIO transfer Data Register Low Portion(0x8812000C) before Data Register High Portion(0x8812000D)

SIO\_IRQ\_CLR: SIO Interrupt clear bit

ADDR\_Mode: SIO addressing mode selection bits.

- 00 = 16-bit address.
- 01 = No address.
- 10 = 8-bit address.
- 11 = 24-bit address.

BAUDRATE: SIO baud rate selection bits.

- 00 = 27/16 MHz.
- 01 = 27/4 MHz.
- 10 = 27/8 MHz.
- 11 = 27/32 MHz.

PROTOCOL: SIO protocol read/write bit waveform definition.

The definition of the SIO protocol read/write bit is shown in the "SIO Protocol" section.

0 = In write mode, the read/write bit waveform will be low.

In read mode, the read/write bit waveform will be high.

1 = In both write and read mode, the read/write bit waveform will be low.

IRQ\_EN: SIO interrupt enable control bit.

0 = SIO polling method.

1 = SIO interrupt enable. The SIO interrupt use IRQ44.

APBDMA\_EN: APBDMA enable

0 = SIO transfer data by CPU.

1 = Enable SIO transfer data by APBDMA. The detail step is referred to APBDMA.

SIO\_DATA\_WIDTH: SIO data width control bit.

0 = data width is 8 bits.

1 = data width is 16 bits.

SIO\_IRQ\_STS: SIO interrupt status bit.

If this bit is 1, it means the SIO interrupt occurs. This bit is read only.

SIO\_REQUEST: SIO data transfer request bit.

Read only. This bit is useful when you use the polling method instead of the interrupt method. If this bit is read as 1 in the write mode, it means the SIO interface circuit is waiting for the user to write a 8-bit/16-bit data into SIO\_DATA register. If this bit is read as 1 in the read mode, it means the SIO interface circuit has received a 8-bit/16-bit data and store it in the SIO\_DATA register. And user must read the SIO\_DATA register to get the received data.

**Note:** If “automatic bit-stream transmitting mode” is turn on, then this bit is always 0.

#### 28.4.2 P\_SIO\_AutoTx (0x88120004): Automatic transmit word number

NAME	D7-D4	D3-D2	D1	D0
P_SIO_Control	SIO_WORD_NUM		DS301_READY	AUTOMATIC_EN

AUTOMATIC\_EN: Automatic bit-stream transmitting mode for the SPDS301.

0 = disable the automatic bit-stream transmitting mode.

1 = start the automatic bit-stream transmitting mode.

The SIO interface circuit will keep on polling the SPDS301's status register until the “decode work bit” and “request ready bit” are both 1. Upon the completion of the polling, a SIO interrupt will occur, and the user should write a bit-stream data word to the SIO\_DATA register. After writing data to the SIO\_DATA register, this interrupt will be cleared automatically. In summary, each time the SIO interrupt occurs during the automatic bit-stream transmitting mode, just writes bit-stream data into the SIO\_DATA register. For the detailed information of the SPDS301, please refer to the SPDS301

programming guide.

DS301\_READY: Use DS301\_ready pin.

0 = Not use.

1 = Use.

The DS301\_ready pin may locate at IOB[0] or IOC[10] according to the SIO port select setting. This bit is only effective when

“automatic bit-stream transmitting mode for the SPDS301” is turn on. For proper usage of this control bit, refer to the “SIO usage procedure” section.

SIO\_WORD\_NUM: Automatic Transmit Word Number

Number of data word that will be transmitted to SPDS301 in one package. Default value is 15. This register is only effective when

“automatic bit-stream transmitting mode for the SPDS301” is turn on. The value of this register should **not** be set as 0. Otherwise, it will cause un-predictable result.

#### 28.4.3 P\_SIO\_Addr(0x88120008): SIO Start Address Register

NAME	D23-D16	D15-D8	D7-D0
P_SIO_Addr	SIO_ADDRH	SIO_ADDRM	SIO_ADDRL

SIO\_ADDRL: SIO Start Address Register Low Portion

It is used when addressing mode is not equal 01

SIO\_ADDRM: SIO Start Address Register Middle Portion

It is used when addressing mode is equal 00 and 11

SIO\_ADDRH: SIO Start Address Register High Portion

It is used when addressing mode is equal 11

#### 28.4.4 P\_SIO\_DATA(0x8812000C): SIO Data Register.

NAME	D15-D8	D7-D0
P_SIO_DATA	SIO_DATAH	SIO_DATAH

SIO\_DATAH: SIO Data Register High Portion

SIO\_DATAH: SIO Data Register High Portion

It is used when data width is equal 1.

### 28.5 SIO Interrupt Processing

The SIO interrupt signal is controlled by the 0x8812\_0001[5]. If 0x8812\_0001[5] is 1, the SIO interrupt

is enabled. User should note that setting 0x8812\_0000[7] could clear the SIO interrupt signal.

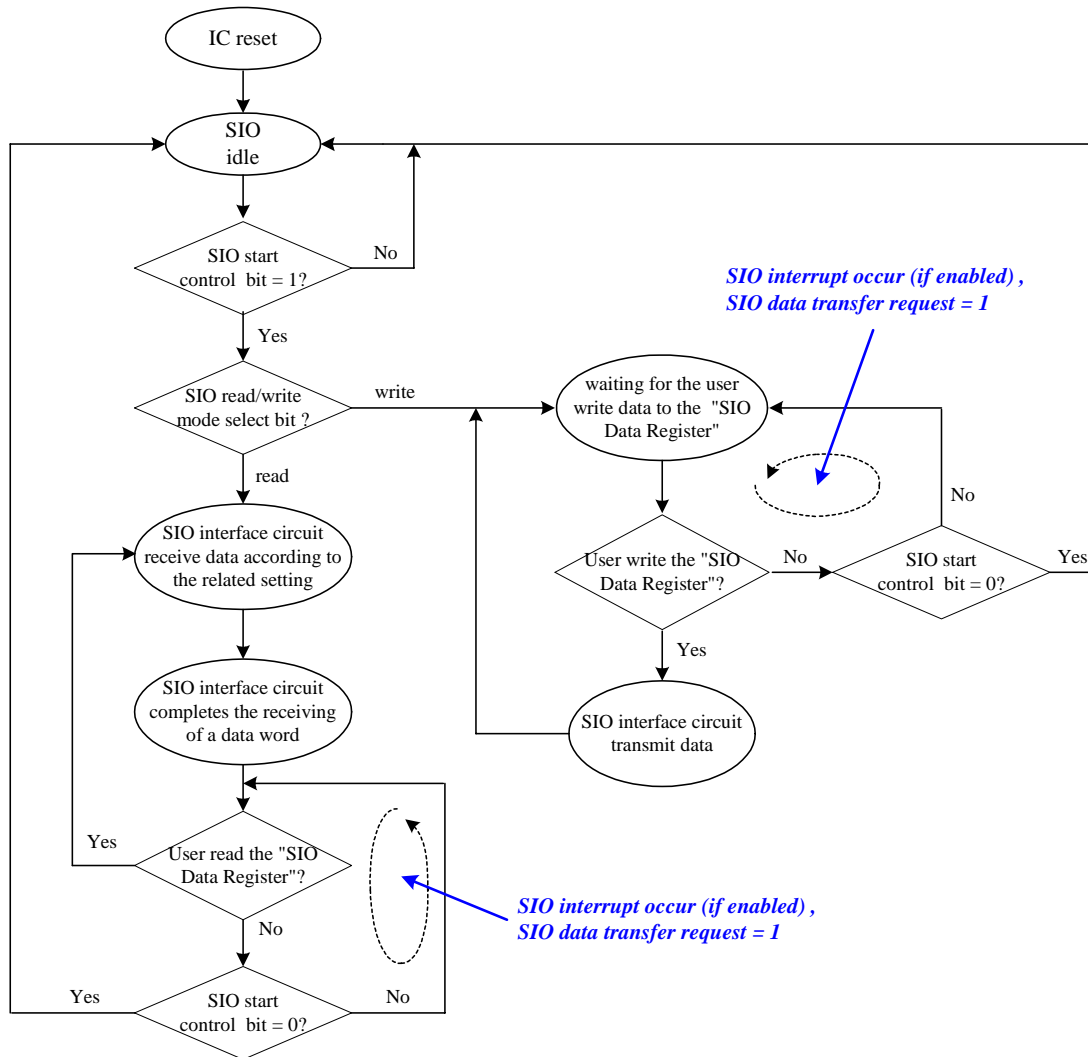
Following tables depict what the proper action is.

<b>Not using “automatic bit-stream transmitting mode”</b>	
<b>the SIO status when the interrupt occurs</b>	<b>proper action</b>
If SIO interrupt is generated during the “Read Mode”, it means the SIO interface has received a data byte/word.	(a) Reading the SIO_DATA register, and then setting the interrupt clear register clear interrupt. And the SIO interface will continue receiving data. (b) Even clear the start/stop control bit of the SIO, the interrupt is still cleared by setting the interrupt clear register.
If SIO interrupt is generated during the “Write Mode”, it means the SIO interface is waiting for data to be transmitted.	(a) By writing data to the SIO_DATA register, the interrupt will happen after finishing transferring data. And setting the interrupt clear register to clear interrupt. (b) Even clear the start/stop control bit of the SIO, the interrupt is still cleared by setting the interrupt clear register.

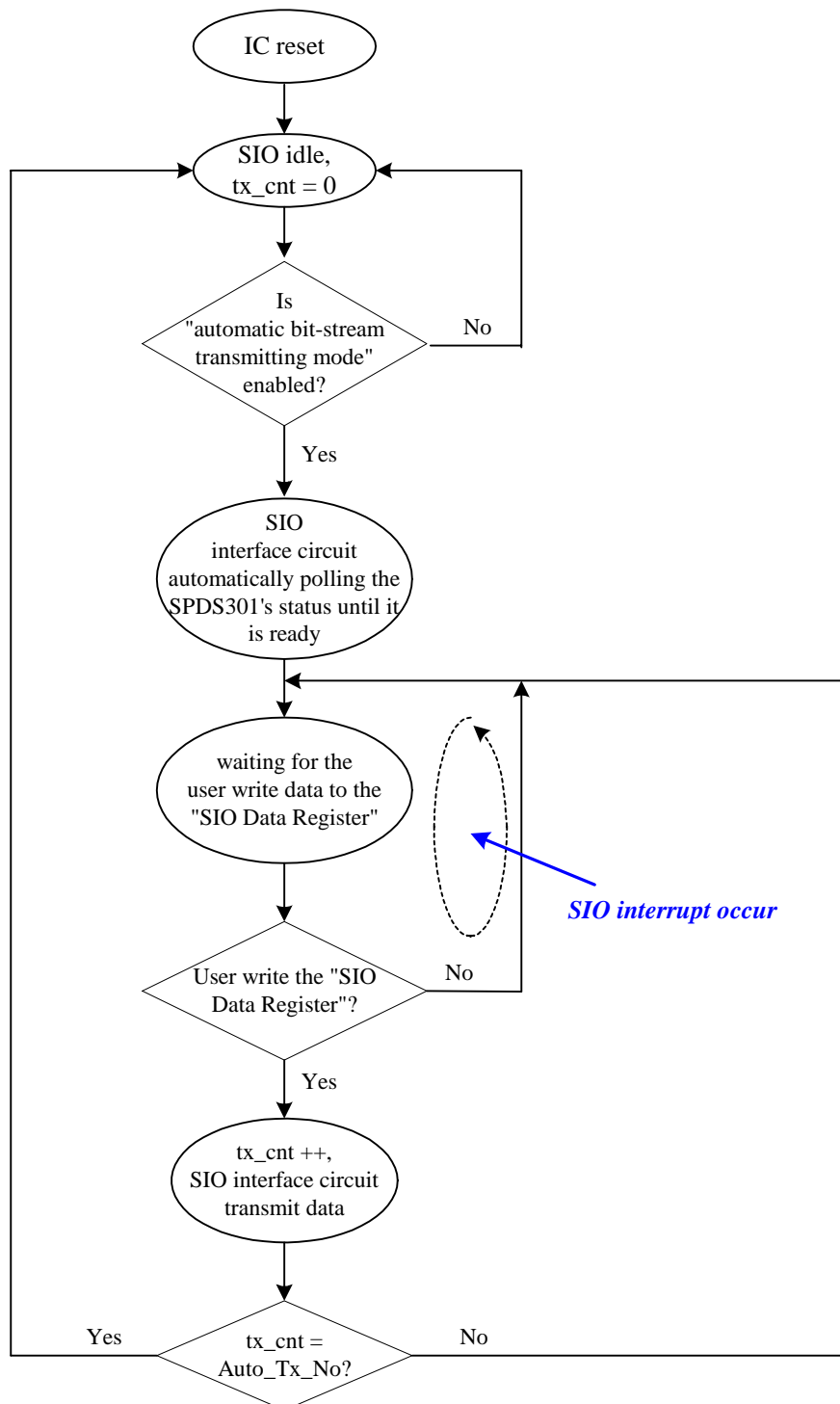
<b>Using “automatic bit-stream transmitting mode”</b>	
<b>the SIO status when the interrupt occurs</b>	<b>proper action</b>
If SIO interrupt is generated, it means the SIO interface is waiting for data to be transmitted.	By writing data to the SIO_DATA register, the interrupt will happen after the SIO interface transmitting this data. Setting the interrupt clear register to clear interrupt

## 28.6 SIO Usage Procedure

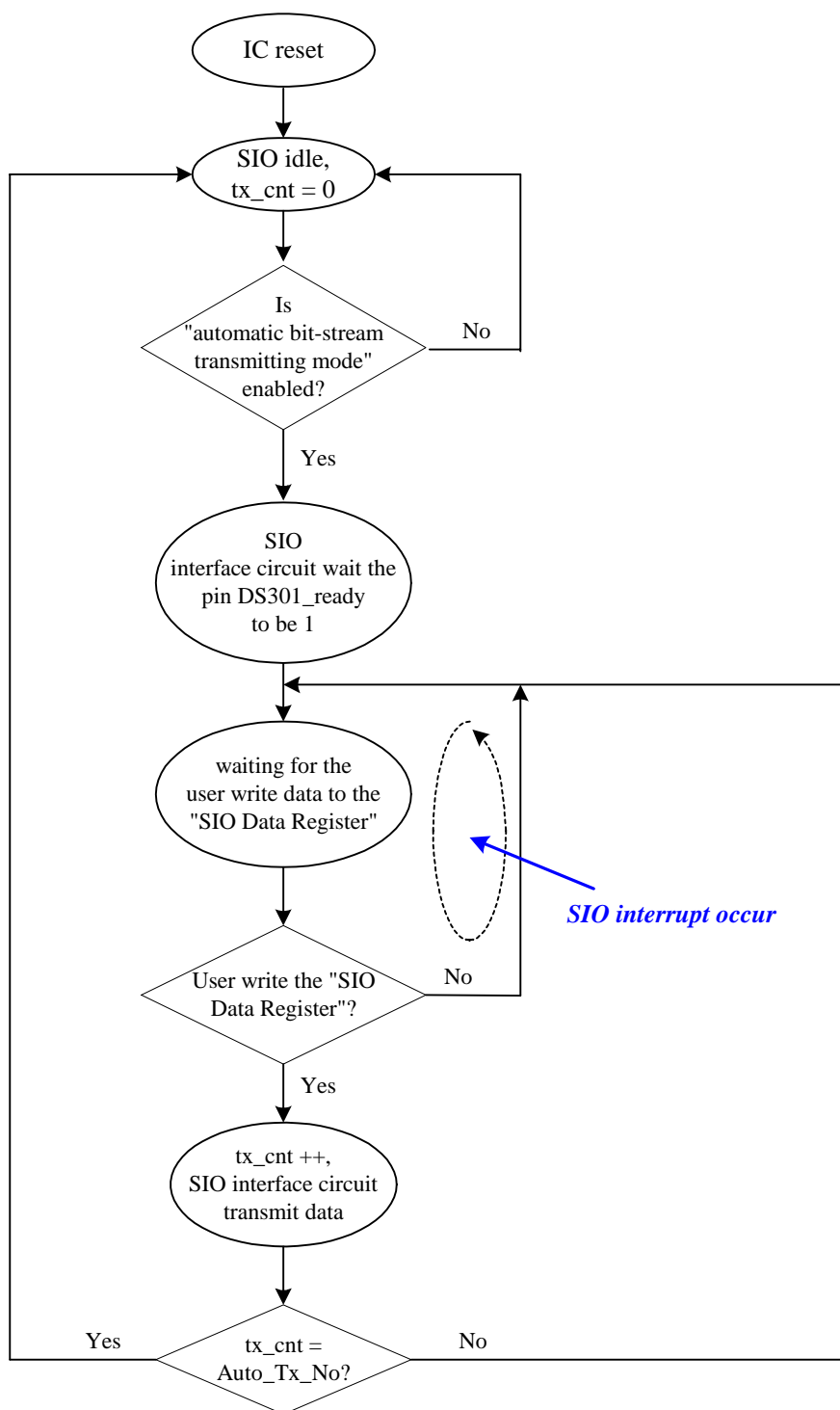
### (a) Not using “automatic bit-stream transmitting mode”



(b) Using “automatic bit-stream transmitting mode” & **not** using DS301\_ready pin



(c) Using both "automatic bit-stream transmitting mode" & DS301\_ready pin



## 29 UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER-UART

The UART of SPG290 is an enhanced serial communication unit for more powerful and flexible data exchanges between MCUs. Major features of UART include:

- \* The maximum baudrate is up to 460.8Kbps.
- \* Programmable of transmitting wait-time and receiving latency-time.
- \* Embedded 2 bytes transmit FIFO and embedded 8 bytes receiving FIFO.
- \* Programmable receiving FIFO size from no FIFO to 8 bytes.
- \* Independent mask of transmit FIFO, receiving FIFO, and receiver timeout interrupts.
- \* False start bit detection.
- \* Link break generation and detection.
- \* Standard MODEM control signals CTS, RTS, DSR, DTR and DCD included.
- \* Fully-programmable serial interface characteristics including:

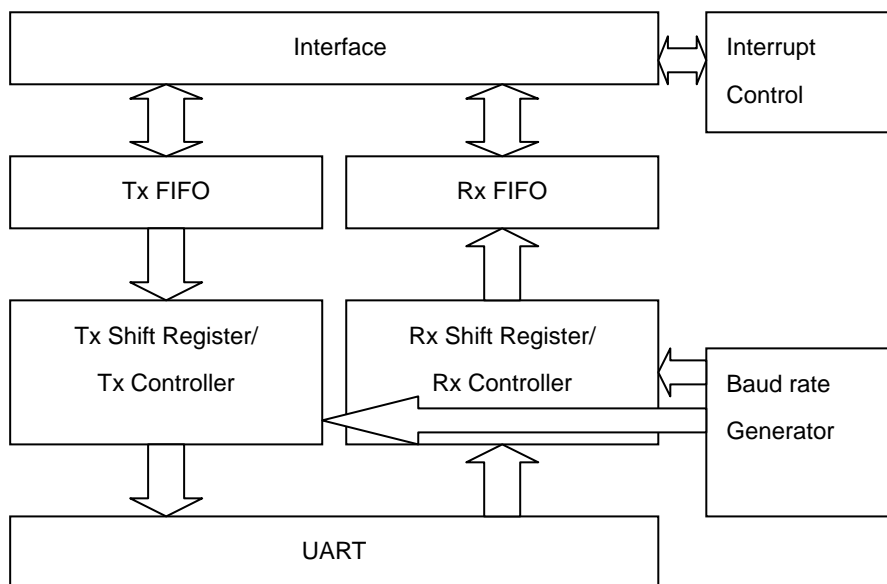
Data can be 5, 6, 7 or 8 bits

Even , odd or no-parity bit generation and detection

1 or 2 stop bit generation

### 29.1 Block Diagram

Block Diagram of UART Module.



### 29.2 Control Registers

#### 29.2.1 P\_UART\_Data(R/W)(0x88150000): UART Data Tx/Rx Register.

NAME	B7	B6	B5	B4	B3	B2	B1	B0
P_UART_DATA	UART data							

UART\_Data: UART data Read/Write Register

### 29.2.2 P\_UART\_ERR(R/W)(0x88150004):UART Tx & Rx Error Flag Register

Name	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
P_UART_ERR	RxD												OE	BE	PE	FE

RxD: UART RxD Signal

OE: Overrun Error

This bit is set to 1 if data is received and the FIFO is already full.

Read 0 = Not Occurs

Read 1 = Happened

Write 1 = Clear this error flag

BE: Break Error

This bit is set to 1 if a break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits).

Read 0 = Not Occurs

Read 1 = Happened

Write 1 = Clear this error flag

PE: Parity Error

This bit is set to 1 if the parity of the received data character does not match the parity selected in PSEL control bit. This bit is refreshing in every read. So, it is necessary to check this bit after DATA register is read.

Read 0 = Not Occurs

Read 1 = Happened

Write 1 = Clear the error flag.

FE: Frame Error

This bit is set to 1 if the received character did not have a valid stop bit (a valid stop bit is 1 bit). This bit is refreshing in every read. So, it is necessary to check this bit after DATA register is read.

Read 0 = Not Occurs

Read 1 = Happened

Write 1= Clear this Error Flag

### 29.2.3 P\_UART\_Ctrl(R/W)(0x88150008):UART Control Register

Name	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
P_UART_Ctrl	RIEN	TIEN	RT	UEN	MSIE					WLSEL	FEN	SBSEL	PSEL	PEN	SB	

RIEN: Receive Interrupt Enable

0= Disable 1= Enable

TIEN: Transmit Interrupt Enable

0= Disable 1= Enable

RT: Receive Timeout Interrupt Enable

0= Disable 1= Enable

UEN: UART Enable

0= Disable 1= Enable

WLSEL: Word Length Definition

Indicate number of data bits transmitted or received in a frame

00 = 5bits 01 = 6bits

10 = 7bits 11 = 8bits

FEN: FIFO Buffer Enable/Disable

Enable FIFO buffer during transmit and receive. When clear this bit to "0", the FIFO become 1-byte-deep holding registers

0= Disable 1= Enable

SBSEL: Stop Bit Size Selection

When this bit is set to "1", two stop bits are transmitted at the end of the frame.

The receive logic does not check for two stop bits being received.

0= 1 Stop Bit 1= 2 Stop Bit

PSEL: Parity Selection

If this bit is set to "1", even parity generation and checking is performed during transmission and reception, which checks for an even number of 1s in data and parity bits. When cleared to "0" the odd parity is performed which checks for an odd number of 1s. This bit has no effect when parity is disabled by PEN Control bit is clear to "0"

0= Odd Parity (if PEN= 1) 1= Even Parity (if PEN= 1)

PEN: Parity Enable

If this bit is set to "1", parity checking and generation is enabled, else parity is

disabled and no parity bit added to the data frame.

0= Disable                      1= Enable

SB:      Send Break

If this bit is set to “1”, a low level is continually output on the TX output pin, after completing transmission of the current character. This bit must be asserted for at least one complete frame transmission time in order to generate a break condition. The transmit FIFO contents remain unaffected during a break condition. For normal use, this bit must be cleared to “0”

0= Normal Operation              1= Send Break Signal

#### 29.2.4 P\_UART\_BaudRate(R/W)(0x8815000C):UART Baud Rate Setup Register

Name	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
P_UART_BaudRate	Baud rate															

To get a desired Baud Rate, write the corresponding value to the Port \$8815000CH according to the following Baud Rate equation.

$$\text{Baud Rate} = \text{Fosc/Baud Rate register} - 1$$

#### 29.2.5 P\_UART\_Status(R/W)(0x88150010):UART Status Register

Name	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
P_UART_Status	RI	TI	RT	MIT		URI	RTS	DTR	TE	RF	TF	RE	BY	DCD	DSR	CTS

RI:              Receive Interrupt Flag

Read 0= Not Occurs                      Read 1= Happened

TI:              Transmit Interrupt Flag

Read 0= Not Occurs                      Read 1= Happened

RT:              Receive Timeout Interrupt Flag

Read 0= Not Occurs                      Read 1= Happened

MIT:            Modem Status Interrupt Flag

Read 0= Not Occurs                      Read 1= Happened

URI:            This bit is the complement of the uUARTRI modem status input

0= uUARTRI is 1                      1= uUARTRI is 0

RTS:            Modem output

DTR:            Modem output

TE: Transmit FIFO Empty Flag

The meaning of this bit depends on the state of the FEN control bit.

If the FIFO is disabled, this bit is set to “1” when the transmit holding register is empty

If the FIFO is enabled, this bit is set to “1” when the transmit FIFO is empty

0= Not Empty                      1= Empty

RF: Receive FIFO Full Flag

The meaning of this bit depends on the state of the FEN control bit.

If the FIFO is disabled, this bit is set to “1” when the receive holding register is full

If the FIFO is enabled, this bit is set to “1” when the receive FIFO is full

0= Not Full                      1= Full

TF: Transmit FIFO Full Flag

The meaning of this bit depends on the state of the FEN control bit.

If the FIFO is disabled, this bit is set to “1” when the transmit holding register is full

If the FIFO is enabled, this bit is set to “1” when the transmit FIFO is full

0= Not Full                      1= Full

RE: Receive FIFO Empty Flag

The meaning of this bit depends on the state of the FEN control bit.

If the FIFO is disabled, this bit is set to “1” when the receive holding register is empty

If the FIFO is enabled, this bit is set to “1” when the receive FIFO is empty

0= Not Empty                      1= Empty

BY: BUSY

If this bit is set to “1”, the UART module is busy transmitting data. This bit remains set until the complete byte, including all the stop bits, has been sent from the shift register.

0= Not Busy                      1= Busy

DCD: This bit is the complement of the uUARTDCD modem status input

0= nUARTDCD is 1                      1= nUARTDCD is 0

**DSR:** This bit is the complement of the uUARTDSR modem status input

0= nUARTDSR is 1                      1= nUARTDSR is 0

**CTS:** This bit is the complement of the nUARTCTS modem status input.

0= nUARTCTS is 1                      1= nUARTCTS is 0

---

---

## 30 UNIVERSAL SERIAL BUS – USB 1.1

---

---

The USB controller built in SPG290 has two functions: to be a USB device or a USB mini-host. To become a USB device, there are 4 endpoints: control pipe, bulk in, bulk out and interrupt in. To become a USB mini host, basic transaction and function are supported. There is only a FIFO implemented by 128x8 bits single port SRAM. To enhance the speed of the transaction when bulk in/bulk out is enable, DMA function is supported and the FIFO is seem as dual FIFO. (Each one is 64 bytes). In addition, it's permitted that bulk in happened simultaneously with bulk out when DMA is disable.

### 30.1 Features

- USB 1.1 Version
- USB device is supported
- USB mini-host is supported
- Build-in USB transceiver
- There are 4 endpoints when USB device is enable
  - Control pipe for standard command
  - Bulk IN for a large number of data transfer
  - Bulk OUT for a large number of data transfer
  - Interrupt in for data transfer seldom happened
- A 8 bytes DFF FIFO for control pipe for USB device only
- For USB device a 128x8 bits single port SRAM for Bulk IN and Bulk OUT and for USB host all transmission are using this single port SRAM
- A 8 bytes DFF FIFO for Interrupt IN for USB device only
- Functions are supported when USB mini-host is enable
  - Setup command or data transaction
  - IN transaction / OUT transaction
  - Programmable with each packet delay time or timeout latency
  - SOF timer / frame number generator
  - Reset signal
- Interrupt mode or polling mode for driver

### 30.2 USB Device

When SPG290 is used as USB Device, it supports 4 endpoints, control pipe, bulk in, bulk out, and interrupt in. For control pipe, when SPG290 receives the standard command, it will auto reply it except Get / Set Descriptor. That is, when it received GET\_STATUS, CLEAR\_FEATURE, SET\_FEATURE, SET\_ADDRESS, GET\_CONFIGURATION, SET\_CONFIGURATION, GET\_INTERFACE, and SET\_INTERFACE, the SPG290 will auto reply these standard commands. For bulk in and bulk out, the maximum packet size is 64 bytes. SPG290 supports non-DMA or DMA transmit / receive. For non-DMA mode, it is 8 bits for MCU access. And it is 16 bits access for DMA mode.

### 30.3 USB Mini Host

SPG290 can be used as the USB mini host. It supports commands, IN, and OUT transfer. For command transfer, the maximum packet size is 64 bytes. For IN and OUT transfer, the maximum packet size is 64 bytes, and it is 8 bits for MCU access or 16 bits for DMA access. When it uses DMA mode, the data to be transferred must be multi-pipe of 64 bytes.

### 30.4 USB Device Control Register

#### 30.4.1 P\_USBD\_Config(0x881B00C0): USB Configuration Register.

NAME	D11-D8	D7-D6	D5	D4	D3	D2-D1	D0
P_USBD_Config	TNSPH	TNSPL	RWIPEN	SPWR	USBEN	-	BYPASS

TNSPH: USB Transceiver Pull High

TNSPL: USB Transceiver Pull Low

RWIPEN: Remote Wakeup Enable

0: Remote Wakeup is not supported

1: Remote Wakeup is supported

SPWR: Self Power of Device

0: USB device is bus-powered

1: USB device is self-powered

USBEN: USB Transceiver Enable

0: Disable

1: Enable

BYPASS: USB Bypass Mode

0: Bypass Disable

1: Bypass enable, the inner USB transceiver is disabled.

#### 30.4.2 P\_USBD\_Device(0x881B015C): USB Device Register.

NAME	D15-D14	D13-D12	D11-D10	D9-D8	D7	D6	D5	D4	D3-D1	D0
P_USBD_Device	EP4_Type	EP3_Type	EP2_Type	EP1_Type	EP4_IO	EP3_IO	EP2_IO	EP1_IO	-	MODE

EP4\_Type: Endpoint4 Type

00: Reserved

01: Reserved

10: Bulk

11: Interrupt

EP3\_Type: Endpoint3 Type

00: Reserved

01: Reserved

10: Bulk

11: Interrupt  
 EP2\_Type: Endpoint2 Type  
 00: Reserved  
 01: Reserved  
 10: Bulk  
 11: Interrupt  
 EP1\_Type: Endpoint1 Type  
 00: Reserved  
 01: Reserved  
 10: Bulk  
 11: Interrupt  
 EP4\_IO: Endpoint4 IN/OUT  
 0: OUT  
 1: In  
 EP3\_IO: Endpoint3 IN/OUT  
 0: OUT  
 1: In  
 EP2\_IO: Endpoint2 IN/OUT  
 0: OUT  
 1: In  
 EP1\_IO: Endpoint1 IN/OUT  
 0: OUT  
 1: In  
 MODE: Mode Selection  
 0: Normal mode  
 1: Debug Mode

#### 30.4.3 P\_USBD\_Function(0x881B00C4): USB Function Register.

NAME	D11	D10	D9	D8-D7	D6-D0
P_USBD_Function	SRST	DMA_BOEN	DMA_BIEN	ConfigVal	FNC_Addr

SRST: Software Reset

DMA\_BOEN: DMA Bulk Out Enable

0: Disable DMA function with bulk out

1: Enable DMA function with bulk out, Bulk In must disable.

DMA\_BIEN: DMA Bulk In Enable

0: Disable DMA function with bulk in

1: Enable DMA function with bulk in, Bulk out must disable

**ConfigVal:** Configure Value  
The USB configuration value of the device can be read from these two bits when received set configuration command.

**FNC\_Addr:** Function Address  
When the device gets "Set Address" command from the host, the address is stored in these bits.

#### 30.4.4 P\_USBD\_DMAINT(0x881B0164): USB DMA Interrupt Register.

NAME	D1	D0
P_USBD_DMAINT	DMAINTEN	DMAINT

**DMAINTEN:** DMA Interrupt Enable  
0: Disable, this interrupt will be masked.  
1: Enable, Hardware will issue an IRQ to CPU.

**DMAINT:** Bulk Out Interrupt  
Read 0: Not occurred  
Read 1: Occurred

#### 30.4.5 P\_USBD\_PMR(0x881B00C8): USB Power Management Register.

NAME	D4	D3	D2	D1	D0
P_USBD_PMR	RESWKE	RE_WA	RE_WAFEA	RST	SUS_MOD

**RESWKE:** Resume Wakeup Enable  
0: Disable  
1: Enable

**RE\_WA:** Remote Wakeup  
0: Disable  
1: Enable

**RE\_WAFEA:** Remote Wakeup Feature  
0: Disable  
1: Enable

**RST:** Reset, assigned to USBBUS reset  
0: USB reset is low  
1: USB reset is high

**SUS\_MOD:** Suspend Mode  
This bit is set by hardware when it enters suspend mode.  
This bit is cleared by hardware when it returns from suspend mode or the USB reset signal is generated.

#### 30.4.6 P\_USBD\_EP0Data(0x881B00CC): USB Endpoint0 Data Register.

NAME	D7-D0
P_USBD_EP0Data	EP0DATA

EP0DATA:      Endpoint0\_Data

#### 30.4.7 P\_USBD\_BIData(0x881B00D0): USB Bulk In Data Register.

NAME	D7-D0
P_USBD_BIData	BIDATA

BIDATA:          Bulk In Data

#### 30.4.8 P\_USBD\_BOData(0x881B00D4): USB Bulk Out Data Register.

NAME	D7-D0
P_USBD_BOData	BODATA

BODATA:          Bulk Out Data

#### 30.4.9 P\_USBD\_INTINData(0x881B00D8): USB Interrupt In Data Register.

NAME	D7-D0
P_USBD_INTINData	INTINDATA

INTINDATA:      Interrupt IN Data

#### 30.4.10 P\_USBD\_NullPkt(0x881B0160): USB Null Packet Register.

NAME	D0
P_USBD_NullPkt	NULLPKT

NULLPKT:          Send Null packet.

0:      disable that USB device send null packet function

1:      enable that USB device can send null packet

#### 30.4.11 P\_USBD\_EPEvent(0x881B00DC): USB Endpoint Event Register.

NAME	D15	D14	D13	D12	D11	D10	D9	D8
P_USBD_EPEvent	IINNA	IINPR	BONA	BOPR	BOPE	BINA	BIPC	BIPR
NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_USBD_EPEvent	E0SNA	E0SEN	E0INNA	E0INPR	E0ONA	E0OPR	E0OPE	E0SFR

IINNA:              Interrupt In NACK

Read 1:      Occurred

Read 0:      Not Occurred

	Write 0:	No Effect
	Write 1:	Clear the flag
IINPR:	Interrupt In Packet Ready	
	Write 1:	IN Packet is ready in the INTERRUPT_IN FIFO.
BONA:	Bulk Out NACK	
	Read 1:	If an OUT packet happened but the device send NAK
	Read 0:	Not Occurred
	Write 0:	No Effect
	Write 1:	Clear the flag
BOPR:	Bulk Out Packet Ready	
	Read 1:	Ready
	Read 0:	Not Ready
	Write 1:	Clear the flag
BOPE:	Bulk Out Packen Enable	
	Write 1:	Enable
BINA:	Bulk IN NACK	
	Read 1:	Occurred an IN request happened but the device sent NAK
	Read 0:	Not Occurred
	Write 1:	Clear the flag
BIPC:	Bulk IN Packet Clear	
	Read 1:	Occurred an IN packet is read from the host
	Read 0:	Not Occurred
	Write 1:	Clear the flag
BIPR:	Bulk IN Packet Ready	
	Write 1:	Packet is ready
E0SNA:	EP0 Status NACK	
	Read 1:	Occurred, if the request of status transaction happened but the device sent NAK.
	Read 0:	Not Occurred
	Write 1:	Clear the flag
E0INPR:	EP0 IN Packet Ready	
	Write 1 to indicate IN packet is ready in Endpoint0 FIFO	
E0ONA:	EP0 Out NACK	
	Read 1:	Occurred an OUT packet happened but the device sent NAK
	Read 0:	Not Occurred
	Write 1:	Clear the flag
E0OPR:	EP0 Out Packet Ready	

Read 1: Ready

Write 1: Clear the flag

E0OPE: EP0 Out Packet Enable

Write 1: Enable the incoming packet for OUT data.

E0SPR: EP0 Setup Packet Ready

Read 1: Occurred a non-standard setup command or get/set descriptor command is loaded into the endpoint0 FIFO.

Write 1: Clear the flag

#### 30.4.12 P\_USBD\_GLOINT(0x881B00E0): USB Global Interrupt Register.

NAME	D6	D5	D4	D3	D2	D1	D0
P_USBD_GLOINT	DMA	STANDARD	POWER	INT	BO	BI	EP0

DMA: DMA Interrupt

Read 1: Occurred

Write 1: Clear the flag

STANDARD: Standard Command Interrupt

Read 1: Occurred

Write 1: Clear the flag

Power: Power Management Interrupt

Read 1: Occurred

Write 1: Clear the flag

INT: Interrupt In Interrupt

Read 1: Occurred one of INTERRUPT\_IN interrupt happened

Write 1: Clear the flag

BO: Bulk Out Interrupt

Read 1: Occurred

Write 1: Clear the flag

BI: Bulk In Interrupt

Read 1: Occurred

Write 1: Clear the flag

EP0: Endpoint0 Interrupt

Read 1: Occurred

Write 1: Clear the flag

**30.4.13 P\_USBD\_INTEN(0x881B00E4): USB Interrupt Enable Register.**

NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_USBD_GLOINT	BIPC	E0SNA	E0SC	E0INNA	E0INPC	E0ONA	E0OPS	E0SPS
NAME	D15	D14	D13	D12	D11	D10	D9	D80
P_USBD_GLOINT	RST	RME	SUS	IINNA	IINPC	BONA	BOPS	BINA

RST: Reset Interrupt Enable

0: Disable

1: Enable

RME: Resume Interrupt Enable

0: Disable

1: Enable

SUS: Suspend Interrupt Enable

0: Disable

1: Enable

IINNA: Interrupt In NACK Interrupt Enable

0: Disable

1: Enable

IINPC: Interrupt In Packet Clear Interrupt Enable

0: Disable

1: Enable

BONA: Bulk Out NACK Interrupt Enable

0: Disable

1: Enable

BOPS: Bulk Out Packet Set Interrupt Enable

0: Disable

1: Enable

BINA: Bulk In NACK Interrupt Enable

0: Disable

1: Enable

BIPC: Bulk In Packet Clear Interrupt Enable

0: Disable

1: Enable

E0SNA: EP0 Status NACK Interrupt Enable

0: Disable

1: Enable

E0SC: EP0 Status Clear Interrupt Enable

0: Disable

1: Enable

E0INNA: EP0 In NACK Interrupt Enable  
 0: Disable  
 1: Enable

E0INPC: EP0 In Packet Clear Interrupt Enable  
 0: Disable  
 1: Enable

E0ONA: EP0 Out NACK Interrupt Enable  
 0: Disable  
 1: Enable

E0OPS: EP0 Out Packet Set Interrupt Enable  
 0: Disable  
 1: Enable

E0SPS: EP0 Setup Packet Set Interrupt Enable  
 0: Disable  
 1: Enable

#### 30.4.14 P\_USBD\_INT(0x881B00E8): USB Interrupt Register.

NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_USBD_INT	BIPC	E0SNA	E0SC	E0INNA	E0INPC	E0ONA	E0OPS	E0SPS
NAME	D15	D14	D13	D12	D11	D10	D9	D80
P_USBD_INT	RST	RME	SUS	IINNA	IINPC	BONA	BOPS	BINA

RST: Reset Interrupt Flag  
 Read 0: Not Occurred  
 Read 1: Occurred  
 Write 1: Clear the flag

RME: Resume Interrupt Flag  
 Read 0: Not Occurred  
 Read 1: Occurred  
 Write 1: Clear the flag

SUS: Suspend Interrupt Flag  
 Read 0: Not Occurred  
 Read 1: Occurred  
 Write 1: Clear the flag

IINNA: Interrupt In NACK Interrupt Flag  
 Read 0: Not Occurred  
 Read 1: Occurred  
 Write 1: Clear the flag

IINPC:	Interrupt In Packet Clear Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
BONA:	Bulk Out NACK Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
BOPS:	Bulk Out Packet Set Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
BINA:	Bulk In NACK Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
BIPC:	Bulk In Packet Clear Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
E0SNA:	EP0 Status NACK Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
E0SC:	EP0 Status Clear Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
E0INNA:	EP0 In NACK Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
E0INPC:	EP0 In Packet Clear Interrupt Flag
	Read 0: Not Occurred
	Read 1: Occurred
	Write 1: Clear the flag
E0ONA:	EP0 Out NACK Interrupt Flag

Read 0: Not Occurred  
Read 1: Occurred  
Write 1: Clear the flag

E0OPS: EP0 Out Packet Set Interrupt Flag  
Read 0: Not Occurred  
Read 1: Occurred  
Write 1: Clear the flag

E0SPS: EP0 Setup Packet Set Interrupt Flag  
Read 0: Not Occurred  
Read 1: Occurred  
Write 1: Clear the flag

#### 30.4.15 P\_USBD\_SCINTEN(0x881B00EC): USB Standard Command Interrupt Enable Register.

NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_USBD_SCINTEN	GSTS	CFEA	SFEA	SADD	GCON	SCON	GINT	SINT

GSTS: Get Status Interrupt Enable  
0: Disable  
1: Enable

CFEA: Clear Feature Interrupt Enable  
0: Disable  
1: Enable

SFEA: Set Feature Interrupt Enable  
0: Disable  
1: Enable

SADD: Set Address Interrupt Enable  
0: Disable  
1: Enable

GCON: Get Configuration Interrupt Enable  
0: Disable  
1: Enable

SCON: Set Configuration Interrupt Enable  
0: Disable  
1: Enable

GINT: Get Interface Interrupt Enable  
0: Disable

SINT:           1:    Enable  
                  Set Interface Interrupt Enable  
                  0:    Disable  
                  1:    Enable

#### 30.4.16 P\_USBD\_SCINT(0x881B00F0): USB Standard Command Interrupt Register.

NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_USBD_SCINT	GSTS	CFEA	SFEA	SADD	GCON	SCON	GINT	SINT

GSTS:           Get Status Interrupt Flag  
                  Read 0:   Not Occurred  
                  Read 1:   Occurred  
                  Write 1:   Clear the flag

CFEA:           Clear Feature Interrupt Flag  
                  Read 0:   Not Occurred  
                  Read 1:   Occurred  
                  Write 1:   Clear the flag

SFEA:           Set Feature Interrupt Flag  
                  Read 0:   Not Occurred  
                  Read 1:   Occurred  
                  Write 1:   Clear the flag

SADD:           Set Address Interrupt Flag  
                  Read 0:   Not Occurred  
                  Read 1:   Occurred  
                  Write 1:   Clear the flag

GCON:           Get Configuration Interrupt Flag  
                  Read 0:   Not Occurred  
                  Read 1:   Occurred  
                  Write 1:   Clear the flag

SCON:           Set Configuration Interrupt Flag  
                  Read 0:   Not Occurred  
                  Read 1:   Occurred  
                  Write 1:   Clear the flag

GINT:           Get Interface Interrupt Flag  
                  Read 0:   Not Occurred  
                  Read 1:   Occurred  
                  Write 1:   Clear the flag

SINT: Set Interface Interrupt Flag

Read 0: Not Occurred

Read 1: Occurred

Write 1: Clear the flag

#### 30.4.17 \_USBD\_EPAutoSet(0x881B00F4): USB Endpoint Auto Set Register.

NAME	D5	D4	D3	D2	D1	D0
<b>P_USBD_EPAutoSet</b>	EP0ASE	IAINPR	BAOPE	BAIPR	E0AIPR	E0AOPE

EP0ASE: EP0 Auto Status Enable Set  
Please refer to P\_USBD\_EPEvent[6]

IAINPR: Interrupt Auto In Packet Ready Set  
Please refer to P\_USBD\_EPEvent[14]

BAOPE: Bulk Auto Out Packet Enable Set  
Please refer to P\_USBD\_EPEvent[11] or [12]

BAIPR: Bulk Auto In Packet Ready Set  
Please refer to P\_USBD\_EPEvent[8]

E0AIPR: EP0 Auto In Packet Ready Set  
Please refer to P\_USBD\_EPEvent[4]

E0AOPE: EP0 Auto Out Packet Enable Set  
Please refer to P\_USBD\_EPEvent[1] or [2]

#### 30.4.18 P\_USBD\_EPSetStall(0x881B00F8): USB Endpoint Set Stall Register.

NAME	D3	D2	D1	D0
<b>P_USBD_EPSetStall</b>	IINS	BOS	BIS	EP0S

IINS: Interrupt In Set Stall  
1 = enable STALL response  
0 = disable STALL response.

BOS: Bulk Out Set Stall  
1 = enable STALL response  
0 = disable STALL response.

BIS: Bulk In Set Stall  
1 = enable STALL response  
0 = disable STALL response.

EP0S: EP0 Set Stall  
1 = enable STALL response  
0 = disable STALL response.

#### 30.4.19 P\_USBD\_EPBufClear(0x881B00FC): USB Endpoint Buffer Clear Register.

NAME	D3	D2	D1	D0
P_USBD_EPBufClear	IINBC	BOBC	BIBC	EP0BC

IINBC: Interrupt In Buffer Clear  
Write "1" to clear Interrupt In buffer

BOBC: Bulk Out Buffer Clear  
Write "1" to clear Bulk Out buffer

BIBC: Bulk In Buffer Clear  
Write "1" to clear Bulk In buffer

EP0BC: EP0 Buffer Clear  
Write "1" to clear EP0 buffer

#### 30.4.20 P\_USBD\_EPEvntClear(0x881B0100): USB Endpoint Event Clear Register.

NAME	D5	D4	D3	D2	D1	D0
P_USBD_EPEvntClear	IINPC	BOEC	BIPC	EP0SC	EP0IPC	EP0OEC

IINPC: Interrupt In Packet Clear  
Write "1" to this bit to clear P\_USBD\_EPEvent[14]

BOEC: Bulk Out Enable Clear  
Write "1" to this bit to clear P\_USBD\_EPEvent[11]

BIPC: Bulk In Packet Clear  
Write "1" to this bit to clear P\_USBD\_EPEvent[9]

EP0SC: EP0 Status Clear  
Write "1" to this bit to clear P\_USBD\_EPEvent[6]

EP0IPC: EP0 In Packet Clear  
Write "1" to this bit to clear P\_USBD\_EPEvent[4]

EP0OEC: EP0 Out Enable Clear  
Write "1" to this bit to clear P\_USBD\_EPEvent[1]

#### 30.4.21 P\_USBD\_EP0WrtCount(0x881B0104): USB Endpoint0 Write Count Register.

NAME	D3-D0
P_USBD_EP0WrtCount	EP0WC

EP0WC: EP0 Write Count  
Read these bits to indicate the number of data in the EP0 FIFO.

**30.4.22 P\_USBD\_BOWrtCount(0x881B0108): USB Bulk Out Write Count Register.**

NAME	D6-D0
P_USBD_BOWrtCount	BOWC

BOWC: Bulk Out Write Count

Read these bits to indicate the number of data in the Bulk Out FIFO.

**30.4.23 P\_USBD\_EP0BufPointer(0x881B010C): USB Endpoint0 Buffer Pointer Register.**

NAME	D5-D3	D2-D0
P_USBD_EP0BufPointer	EP0WBP	EP0RBP

EP0WBP: EP0 Write Buffer Pointer

The write pointer of EP0 FIFO

EP0RBP: EP0 Read Buffer Pointer

The read pointer of EP0 FIFO

**30.4.24 P\_USBD\_BIBufPointer(0x881B0110): USB Bulk In Buffer Pointer Register.**

NAME	D15-D8	D7-D0
P_USBD_BIBufPointer	BIBRP	BIBWP

BIBRP: Bulk IN Buffer Write Pointer

The write pointer of Bulk IN FIFO

BIBWP: Bulk IN Buffer Read Pointer

The read pointer of Bulk IN FIFO

**30.4.25 P\_USBD\_BOBufPointer(0x881B0114): USB Bulk Out Buffer Pointer Register.**

NAME	D15-D8	D7-D0
P_USBD_BOBufPointer	BOBRP	BOBWP

BOBRP: Bulk OUT Buffer Write Pointer

The write pointer of Bulk OUT FIFO

BOBWP: Bulk OUT Buffer Read Pointer

The read pointer of Bulk OUT FIFO

**30.4.26 P\_USBD\_EP0RTR(0x881B0118): USB Endpoint0 bmRequestType Register.**

NAME	D7-D0
P_USBD_EP0RTR	EP0RTR

EP0RTR: EP0 bmRequestType

The bmRequestType of setup command

#### 30.4.27 P\_USBD\_EP0RR(0x881B011C): USB Endpoint0 bRequest Register.

NAME	D7-D0
P_USBD_EP0RR	EP0RR

EP0RR: EP0 bRequest  
The bRequest of setup command

#### 30.4.28 P\_USBD\_EP0VR(0x881B0120): USB Endpoint0 wValue Register.

NAME	D15-D0
P_USBD_EP0VR	EP0VR

EP0VR: EP0 wValue  
The wValue of setup command

#### 30.4.29 P\_USBD\_EP0IR(0x881B0124): USB Endpoint0 wIndex Register.

NAME	D15-D0
P_USBD_EP0IR	EP0IR

EP0IR: EP0 wIndex  
The wIndex of setup command

#### 30.4.30 P\_USBD\_EP0LR(0x881B0128): USB Endpoint0 wLength Register.

NAME	D15-D0
P_USBD_EP0LR	EP0LR

EP0LR: EP0 wLength  
The wLength of setup command

#### 30.4.31 P\_USBD\_DMAWrtCount(0x881B0140): USB DMA Byte Count Register.

NAME	D23-D0
P_USBD_DMAWrtCount	DMAWC

DMAWC: DMA Write Count  
The register is used in DMA mode. Setting this register to indicate how many word data is transferred.

#### 30.4.32 P\_USBD\_DMAACKCount(0x881B0144): USB DMAACK Count Register.

NAME	D18-D0
P_USBD_DMAACKCount	DMAACK

DMAACK: DMA ACK Count

Write this port to reset DMAWC and DMAACK

### 30.4.33 P\_USB\_EPStall(0x881B0150): USB Endpoint Stall Bit Register.

NAME	D11	D10	D9	D8	D7-D4	D3	D2	D1	D0
P_USBD_EPStall	IISS	BOSS	BISS	EP0SS		IISB	BOSB	BISB	EP0SB

IISS: Interrupt IN

It indicates that USB device has replied the request by sending STALL

Write 1: Clear the flag

BOSS: Bulk OUT

It indicates that USB device has replied the request by sending STALL

Write 1: Clear the flag

BISS: Bulk IN

It indicates that USB device has replied the request by sending STALL

Write 1: Clear the flag

EP0SS: Endpoint0

It indicates that USB device has replied the request by sending STALL

Write 1: Clear the flag

IISB: Interrupt IN

If STALL happened, the bit is set. The bit is for debugging only.

BOSB: Bulk OUT

If STALL happened, the bit is set. The bit is for debugging only.

BISB: Bulk IN

If STALL happened, the bit is set. The bit is for debugging only.

EP0SB: Endpoint0

If STALL happened, the bit is set. The bit is for debugging only.

## 30.5 USB Host Control Register.

### 30.5.1 P\_USBH\_Config(0x881C0000): USB Host Configuration Register.

Bit 0 in [P\_USBH\_Config] must set to 1 when accessing the USB Host functions.

NAME	D5	D4	D3	D2	D1	D0
P_USBH_Config	SUS		ASOF	SOFTTR		HOSTEN

SUS: Host Suspend,

0 = HOST SUSPEND is disable.

1 = HOST SUSPEND is enable. USB Transceiver is in SUSPEND mode.

(The SUSPEND mode is controlled by software when HOST is enable.)

ASOF: Auto Generate SOF

0= Generate SOF by software  
1= Generate SOF by hardware

SOFTR: SOF Timer  
0= Disable SOF timer  
1= Enable SOF timer

HOSTEN: Host Enable  
0= Host is disabled  
1= Host is enabled (Device is disabled)

### 30.5.2 P\_USBH\_TimeConfig(0x881C0004): USB Host Timing Configuration Register.

NAME	D5	D4	D3-D2	D1-D0
P_USBH_TimeConfig	SAU	PAC	TC	IPD

SAU: Storage 1/2 Auto Mode  
Always set to 1, If this bit is set 1, the following behavior is not necessary:  
Setting up P\_USBH\_StorageRST after each transaction in non-DMA mode.

PAC: Pointer Auto Clear  
Always set to 1, If the bit is set 1, READ/WRITE pointer is automatically reset to 0 after any transaction. However, this mode is automatically disable in DMA mode.  
The DMA mode is entered by configuring P\_USBH\_AutoTrans[0] or P\_USBH\_AutoTrans[1].

TC: TimeOut Criteria  
00= 16T  
01= 18T  
10= 20T  
11= 22T

IPD: Inter Packet Delay  
00= 2T  
01= 4T  
10= 6T  
11= 8T

### 30.5.3 P\_USBH\_Data(0x881C0008): USB Host Data Register.

NAME	D7-D0
P_USBH_Data	HDATA

HDATA: Host Data

#### 30.5.4 P\_USBH\_Transfer(0x881C000C): USB Host Transfer Register.

NAME	D6	D5	D4	D3	D2	D1	D0
P_USBH_Transfer	RST	OD1	OD0	ID1	ID0	Setup	SOF

RST:	Reset Signal 1= Generate reset signal 0= Stop reset signal
OD1:	Out Data1 Transfer Write 1 to generate OUT DATA1 transfer. This bit is cleared automatically if the transfer is completed.
OD0:	Out Data0 Transfer Write 1 to generate OUT DATA0 transfer. This bit is cleared automatically if the transfer is completed.
ID1:	In Data1 Transfer Write 1 to generate IN DATA1 transfer. This bit is cleared automatically if the transfer is completed.
ID0:	In Data0 Transfer Write 1 to generate IN DATA0 transfer. This bit is cleared automatically if the transfer is completed.
Setup:	Setup Transfer Write 1 to generate SETUP transfer. This bit is cleared automatically if the transfer is completed.
SOF:	SOF Transfer Write 1 to generate SOF transfer. (SOF is generated by software) This bit is cleared automatically if the transfer is completed.

#### 30.5.5 P\_USBH\_DveAddr(0x881C0010): USB Device Address Register.

NAME	D6-D0
P_USBH_DevAddr	DAddr

DAddr:	Device Address Write the device address to these bits that host want to access.
--------	--

#### 30.5.6 P\_USBH\_DveEP(0x881C0014): USB Device Endpoint Register.

NAME	D3-D0
P_USBH_DveEP	DEP

DEP:	Device Endpoint Write the device endpoint to these bits that host want to access.
------	--

### 30.5.7 P\_USBH\_TXCount(0x881C0018): USB Host Transmit Count Register.

NAME	D6-D0
P_USBH_TXCount	TXC

TXC: Host Transmitting Count

Write the number of data to these bits that host wants to transmit.

### 30.5.8 P\_USBH\_RXCount(0x881C001C): USB Receive Count Register.

NAME	D6-D0
P_USBH_RXCount	RXC

RXC: Host Receiving Count

The number of data that host received is stored in these bits.

### 30.5.9 P\_USBH\_FIFOInPointer(0x881C0020): USB Host FIFO Input Pointer Register.

NAME	D7-D0
P_USBH_FIFOInPointer	HFIP

HFIP: Host FIFO Input Pointer

### 30.5.10 P\_USBH\_FIFOPutPointer(0x881C0024): USB Host FIFO Output Pointer Register.

NAME	D7-D0
P_USBH_FIFOPutPointer	HFOP

HFOP: Host FIFO Output Pointer

### 30.5.11 P\_USBH\_AutoInByteCount(0x881C0028): USB Host Automatic In Transaction Byte Count Register.

NAME	D15-D0
P_USBH_AutoInByteCount	HAIBC

HAIBC: Host Automatic IN Transaction Byte Count

Write the number of IN transaction that had to be generated in these bits.

For example, if the host is going to receive 512 bytes from the device, it has to generate  $512/64=8$  IN transaction, and write 8 to this register. This register is used only in DMA mode.

### 30.5.12 P\_USBH\_AutoOutByteCount(0x881C002C): USB Host Automatic Out Transaction Byte Count Register.

NAME	D15-D0
------	--------

<b>P_USBH_AutoOutByteCount</b>	<b>HAOBC</b>
--------------------------------	--------------

**HAOBC:** Host Automatic OUT Transaction Byte Count

Write the number of OUT transaction in these bits that had to be generated. For example, if the host is going to transmit 512 bytes to the device, it has to generate  $512/64=8$  OUT transaction, and write 8 to this register. This register is used only in DMA mode.

### 30.5.13 P\_USBH\_AutoTrans(0x881C0030): USB Host Auto Transfer Register.

<b>NAME</b>	<b>D3</b>	<b>D2</b>	<b>D1</b>	<b>D0</b>
<b>P_USBH_AutoTrans</b>	<b>CAO</b>	<b>CAI</b>	<b>AOX</b>	<b>AIX</b>

**CAO:** Clear Auto Out Transfer

Write "1" to this bit to clear P\_USBH\_AutoTrans[1]

**CAI:** Clear Auto IN Transfer

Write "1" to this bit to clear P\_USBH\_AutoTrans[0]

**AOX:** Auto Out Transfer

This bit is set 1 by entering DMA mode for Bulk out. When transaction is surely finished, this bit must be cleared.

Note: After DMA is finished (DMA counter reaches 0), the transaction may not be finished yet. Enable this bit, programming the UHAOBCR and then trigger the OUT transfer will start the OUT transaction.

This bit is cleared by software.

**AIX:** Auto In Transfer

This bit is set 1 by entering DMA mode for Bulk in. When transaction is surely finished, this bit must be cleared. Enable this bit, programming the UHAIBCR and then trigger the IN transfer will start the IN transaction. This bit is cleared by software.

### 30.5.14 P\_USBH\_Status(0x881C0034): USB Host Status Register.

<b>NAME</b>	<b>D7</b>	<b>D6</b>	<b>D5</b>	<b>D4</b>	<b>D3</b>	<b>D2</b>	<b>D1</b>	<b>D0</b>
<b>P_USBH_Status</b>	<b>TO</b>	<b>CRC</b>	<b>DE</b>	<b>BS</b>	<b>UP</b>	<b>SH</b>	<b>NH</b>	<b>AH</b>

**TO:** TimeOut, No Response

This flag is set when timeout or no response from the device

Read 0= Not occurred

Read 1= Occurred

Write 0= No effect

Write 1= Clear the flag

**CRC:** CRC16 Error Packet Received

	<p>This flag is set when In data packet is received with CRC16 error.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
DE:	<p>Data Sequence Error Packet Received</p> <p>This flag is set when In data packet is received with data sequence error.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
BS:	<p>Bit Stuffing Error Packet Received</p> <p>This flag is set when In data contains bit stuffing error.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
UP:	<p>Unkonwn PID Packet Received</p> <p>This flag is set when receiving a packet with unknown PID</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
SH:	<p>Stall Handshake Received</p> <p>This flag is set when receiving a stall handshake.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
NH:	<p>NACK Handshake Received</p> <p>This flag is set when receiving a NACK handshake.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
AH:	<p>ACK Handshake Received</p>

This flag is set when receiving an ACK handshake.

Read 0= Not occurred

Read 1= Occurred

Write 0= No effect

Write 1= Clear the flag

### 30.5.15 P\_USBH\_INT(0x881C0038): USB Host Interrupt Register.

NAME	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_USBH_INT	DPO	TRST	TSOFI	ITOK	TXO	VSC	AOX	AIX	RX	TX	SOF	DSC

DPO:

Device Plug Out Interrupt

This interrupt is asserted whenever host has detected device is plug\_out.

Write 1 to clear the interrupt.

Read 0= Not occurred

Read 1= Occurred

Write 0= No effect

Write 1= Clear the flag

TRST:

Transmit USB Reset Interrupt

This interrupt is asserted whenever host has sent USB RESET signal.

Write 1 to clear the interrupt.

Read 0= Not occurred

Read 1= Occurred

Write 0= No effect

Write 1= Clear the flag

TSOFI:

Transmit SOF Interrupt

This interrupt is asserted whenever host has sent a SOF. Write 1 to clear the interrupt.

Read 0= Not occurred

Read 1= Occurred

Write 0= No effect

Write 1= Clear the flag

ITOK:

IN Token Transmit Interrupt

This interrupt is asserted whenever host has sent an IN Token. Write 1 to clear the interrupt.

Read 0= Not occurred

Read 1= Occurred

Write 0= No effect

Write 1= Clear the flag

TXO:	<p>Transmit Data Interrupt</p> <p>This interrupt is asserted whenever host has sent a DATA packet. Write 1 to clear the interrupt.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
VSC:	<p>VBUS Status Change Interrupt</p> <p>This interrupt is asserted whenever VBUS status has changed.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
AOX:	<p>Automatic Out Transfer Interrupt</p> <p>This interrupt is asserted when the host has transmitted the data out to the device and host has received an ACK from the device.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
AIX:	<p>Automatic In Transfer Interrupt</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
RX:	<p>Receive Interrupt</p> <p>This interrupt is asserted whenever the host receives a packet form the device. Write 1 to clear the interrupt.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p> <p>Write 0= No effect</p> <p>Write 1= Clear the flag</p>
TX:	<p>Transmit Interrupt</p> <p>This interrupt is asserted whenever the TX task is completed. Write 1 to clear the interrupt.</p> <p>Read 0= Not occurred</p> <p>Read 1= Occurred</p>

Write 0= No effect  
Write 1= Clear the flag

SOF: SOF Interrupt

This interrupt periodically generated for every 1ms frame time. Write 1 to clear the interrupt.

Read 0= Not occurred  
Read 1= Occurred

Write 0= No effect  
Write 1= Clear the flag

DSC: DP DM Status Change Interrupt

D+/D- status change interrupt. This interrupt is used to detect the device connection when the host controller is in the idle state. Once the device is plug-in, this interrupt must be disabled. Write 1 to clear the interrupt.

Read 0= Not occurred  
Read 1= Occurred

Write 0= No effect  
Write 1= Clear the flag

#### 30.5.16 P\_USBH\_INTEN(0x881C003C): USB Host Interrupt Enable Register.

NAME	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_USBH_INTEN	DPO	TRST	TSOFI	ITOK	TXO	VSC	AOX	AIX	RX	TX	SOF	DSC

DPO: Device Plug Out Interrupt Enable

0= Disable

1= Enable

TRST: Transmit USB Reset Interrupt Enable

0= Disable

1= Enable

TSOFI: Transmit SOF Interrupt Enable

0= Disable

1= Enable

ITOK: IN Token Transmit Interrupt Enable

0= Disable

1= Enable

TXO: Transmit Data Interrupt Enable

0= Disable

1= Enable

VSC:	VBUS Status Change Interrupt Enable 0= Disable 1= Enable
AOX:	Automatic Out Transfer Interrupt Enable 0= Disable 1= Enable
AIX:	Automatic In Transfer Interrupt Enable 0= Disable 1= Enable
RX:	Receive Interrupt Enable 0= Disable 1= Enable
TX:	Transmit Interrupt Enable 0= Disable 1= Enable
SOF:	SOF Interrupt Enable 0= Disable 1= Enable
DSC:	DP DM Status Change Interrupt Enable 0= Disable 1= Enable

### 30.5.17 P\_USBH\_SoftRST(0x881C0044): USB Software Reset Register/Device Plug Out Register.

NAME	D8	D7-D1	D0
P_USBH_SoftRST	DPOE	DPOTV	SRST

DPOE:	Device Plug Out Timer Enable Write "1" to this bit to enable the timer
DPOTV:	Device Plug Out Timer Value If DEVICE_PLUG_OUT_TIMER_ENABLE is 1, the inside timer is enable. For each clock cycle, if D+ and D- are all 0, the timer is added by 1. Besides, if one of D+/D- is not 0, the timer is reset to 0. When it counts to DEVICE_PLUG_OUT_TIMER_VALUE, an interrupt is happened and DEVICE_PLUG_TIMER_OUT_ENABLE is reset.
SRST:	Software Reset

**30.5.18 P\_USBH\_INAckCount(0x881C005C): USB IN ACK Count Register.**

NAME	D15-D0
P_USBH_INAckCount	INACK

INACK: IN ACK Count

**30.5.19 P\_USBH\_OutAckCount(0x881C0060): USB Out ACK Count Register.**

NAME	D15-D0
P_USBH_OutAckCount	OUTACK

OUTACK: OUT ACK Count

**30.5.20 P\_USBH\_RSTAckCount(0x881C0064): USB Reset ACK Count Register.**

NAME	D1	D0
P_USBH_RSTAckCount	IARST	OARST

IARST: IN ACK Reset

Write "1" to reset P\_USBH\_INAckCount

OARST: OUT ACK Reset

Write "1" to reset P\_USBH\_OUTAckCount

**30.5.21 P\_USBH\_Dreadback(0x881C006C): USB D+/ D- Readback Register.**

NAME	D1	D0
P_USBH_Dreadback	DM	DP

DM: VPIN input signal

DP: VMIN input signal

---

---

## 31 SECURE DIGITAL CARD – SDCARD CONTROLLER

---

---

Secure Digital (SD) Memory card is a Flash-Based memory card that is specifically designed to meet the security, capacity, performance and environment requirements inherent in newly emerging audio and video consumer electronic device. The SD Memory Card communication is based on an advanced 9-pin interface (Clock, Command, 4xData and 3xPower lines) designed to operate in a low voltage range.

SD IO card is based on and compatible with the SD memory card. The intent of the SD IO card is to provide high-speed data I/O with low power consumption for mobile electronic devices.

The SD card controller built in SPG290 is designed to provide high performance transfer rate using DMA access which can achieve the best performance/cost ratio.

### 31.1 Features

- \* Fully compatible with SD Memory card specification
- \* Accept SD command directly which improve the compatibility.
- \* Programmable clock speed on the SD bus.
- \* SD bus clock control while buffer is full.
- \* Interrupt generation
- \* DMA R/W operation
- \* Both 1-bit, 4-bits SD mode supported
- \* SD IO card interrupt detection.

## 31.2 Block Diagram

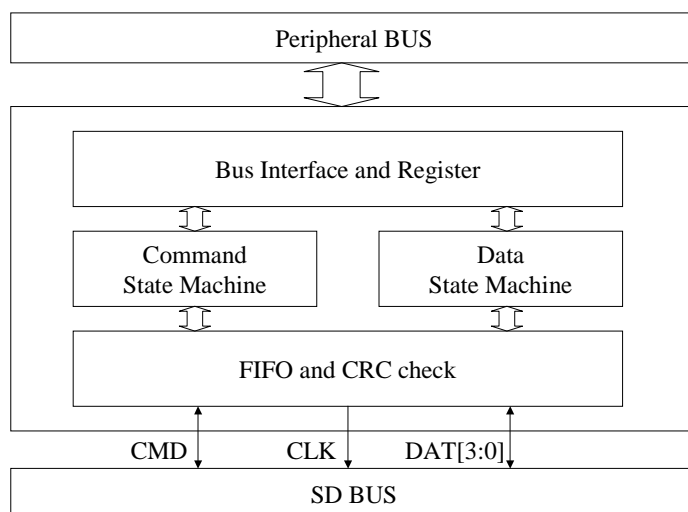


Figure 1. Functional Block Diagram of SD Memory Controller

## 31.3 Control Register

Host use two registers to transfer data to and from SD Card. Host need to poll the **DatBufFull** /**DatBufEmpty** bits in the status register to determine if the controller is able to receive/transfer data.

### 31.3.1 P\_SD\_DATATx (R/W)(0x88180000) : SD Card Data Transmit register

Host writes 32-bits data to this register and the controller will transmit it to SD card. When the data stored in the buffer is transmitted, **DATBUFEMPTY** bit in status register will be set or the DMA request will be issued. It should be noted data could be written to this register only when **DATBUFEMPTY** is 1.

Name	B31-B0
P_SD_DATATx	DataTx

DataTx: Write Data to SDCard, data can be written to this register only when **DATBUFEMPTY** is 1, default is zero.

### 31.3.2 P\_SD\_DATARx(R) (0x88180004): SD Card Data Receive register

This register is used to store the read data from the SD card. When 32-bits data is received, **DATBUFFULL** bit in status register will be set or the DMA request will be issued. It should be noted data could be read from this register only when **DATBUFFULL** is 1.

Name	B31-B0
------	--------

P_SD_DATARx	DataRx
-------------	--------

**DataRx:** Read data from SD card, read data from this register will only valid when the **DATBUFFULL** is set, otherwise, it will return zeros..

### 31.3.3 P\_SD\_COMMAND(R/W)(0x88180008): SD Card Command register

Host uses this register to control the behavior of controller. Controller will use the information in this register to determine what to do.

Name	B7	B6	B5	B4	B3	B2	B1	B0
P_SD_COMMAND	RUNCMD	STPCMD	CMDCODE					
Name	B15	B14	B13	B12	B11	B10	B9	B8
P_SD_COMMAND		RESPTYPE			INICARD	MULBLK	TxDATA	CMDWD

**CMDCODE:** The Command code host wishes to transfer

**STPCMD:** Write 1 to this bit will force the controller back to IDLE state. This bit will be clear to 0 after the controller back to IDLE state.

**RUNCMD:** Write '1' to this register will initiate the SD command on the SD bus according to current configuration of the controller. This bit will be cleared to '0' after the transaction start. You can start a new transaction only when BUSY bit is 0.

**CMDWD:** Indicate if this command with data.  
0 = Command without data.                      1 = Command with data.

**TRANDATA:** Indicate this command transfer or receive data.  
0 = Receive data (Read).                      1 = Transfer data (Write).

**MUL BLK:** Indicate multi-block transfer or not.  
0 = Single block transfer.                      1 = Multiple block transfer.

**INICARD:** Write This Bit to 1 will start a 74 clock cycles on the clock line.

**RESPTYPE:** Indicate the response type of this command. Currently, only the response type R2 have response length 128 bits, all other response will have 32 bits length. Response type R1b will keep the controller to wait for busy signal on the SD bus.  
000 = No response.                      001 = Response type R1.  
010 = Response type R2.                      011 = Response type R3.  
110 = Response type R6.                      111 = Response type R1b

### 31.3.4 P\_SD\_ARGUMENT(R/W)(0x8818000C): SD Card argument register

Host writes the argument it wants to transfer to SD card in this register. Controller will use data in this register as the command argument transfer to the card. The SD Command needs a 32-bit long command.

Name	B31-B0
P_SD_ARGUMENT	Argument

Argument:      Argument[15:0] transfer to SD Card

### 31.3.5 P\_SD\_STATUS (R)(0x88180014): SD Card Controller status register

All the controller status and transaction status will be stored in this register. Host can get information of the controller to determine what to do.

Name	B7	B6	B5	B4	B3	B2	B1	B0
P_SD_STATUS	DATBUFFULL	CMDBUFFFULL	RSPCRCERR	RSPIDXERR	DATCOM	CMDCOM	CARDBUSY	BUSY
Name	B15	B14	B13	B12	B11	B10	B9	B8
P_SD_STATUS			CARDINT	CARDPRE	CARDWP	DATCRCERR	TIMEOUT	DATBUFEMPTY

BUSY:                      Indicate the controller is busy.

0 = Controller is idle.                      1 = Controller is busy.

CARDBUSY:              Indicate the SD card is busy (drive the DAT0 low). Host needs to poll this bit after a write command is issued.

0 = Card is not Busy.                      1 = Card is Busy.

CMDCOM:                Indicate corresponding response is received or timeout happened after send a command.

DATCOM:                Indicate data transfer/receive have completed.

RSPIDXERR:            Indicate the command index in the response is failed.

RSPCRCERR:            Indicate the CRC bits in the response is failed, this bit will be set if the CRC received is not 6'b111111 in the case of response R3.

CMDBUFFFULL:        Indicate the RESP register is full. Read to RESP register or Start a new transaction or set STPCMD in command register will clear this bit.

DATBUFFULL:           This bit will be set when data stored in FIFO is higher than the trigger level. This bit will be clear after appropriate number of data been read from the DATARx register or write 1 to STPCMD bit in command register.

DATBUFEMPTY:	This bit will be set when data stored in the FIFO is lower than the trigger level. This bit will be clear after appropriate number of data been written to the DATATx register or write 1 to STPCMD bit in command register.
TIMEOUT:	Indicate command response time out or read data response time out.
DATCRCERR:	Indicate read data CRC error or write data with CRC error response.
CARDWP:	Indicate the card is write protect. This bit only detect the write protect pin on the interface. Controller's behavior will not be affected by this bit. Host need to take the responsibility to protect the card.
CARDPRE:	Indicate the card is present. This bit only detect the DAT3 on the SD interface when the controller is idle. Controller's behavior will not be affected by this bit. Host can initial a transaction no matter what this bit is. Write 1 to this register will clear the pending interrupt of card present.
CARDINT:	Indicate a SD IO card interrupt is pending. This bit will be set only when IOEN in control register is 1. Host need to clear the interrupt using device specific command. Write 1 to this register will have no effect.

### 31.3.6 P\_SD\_RESPONSE (R ) (0x88180010): SD Card Response register

The response of the SD card will be store in this register. Note the controller will not interpret the information in this register. Host driver needs to take care this.

Command with response R1, R1b, R3, R6 have 6 bits command index and 32-bits response length. The Response will be stored in this register. Command with response R2 will have response length 128 bits. Host need to poll the CMDBUFFFULL bit in the status register to determine when to read this register. The data in this register is valid only when CMDBUFFFULL bit is 1.

Name	B31-B0
P_SD_RESPONSE	Response

Response: Response data from SD card, read data from this register will be valid only if the **CMDBUFFFULL** is set.

### 31.3.7 P\_SD\_CONTROL(R/W)(0x88180018): SD Card Control Register

The register control the clock speed on the SD bus, the block length is also control by this register when transfer or receive data. This register is changeable only when BUSY bit in status register is 0.

Name	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
P_SD_CONTROL					EN_SD	IOEN	DMAMODE	BUSWIDTH	CLKDIV							
Name	B31	B30	B29	B28	B27	B26	B25	B24	B23	B22	B21	B20	B19	B18	B17	B16
P_SD_CONTROL	BLKLEN															

**CLKDIV:** The Clock Speed on the SD bus is calculated from this register.  
 $F_{SDCLK} = F_{SYSCLK} / 2(CLKDIV + 1)$ , Default value is 0x54

**BUSWIDTH:** Indicate the data bus width during transfer.  
0 = 1 bit. 1 = 4 bits.

**DMAMODE:** Indicate DMA mode or not

**IOEN:** 0 = Disable SD IO card interrupt detection.  
1 = Enable SD IO card interrupt detection.

**EN\_SD:** Indicate if the IO is used by the SD controller. Programmer must write this bit to 1 before start using the SD controller.  
0x0: SDCLK, SDCMD, SDDAT0 is used as GPIO.  
0x1: SDCLK, SDCMD, SDDAT0 is used as SD control signal.  
If BUS\_WIDTH is set to 1, the SDDAT1, SDDAT2, SDDAT3 will become SD control signal when this bit is set to 1

**BLKLEN:** Data block length to be transferred, in the unit of bytes. The value in this register should be equal to the block length of the SD/MMC card.

### 31.3.8 13.10 P\_SD\_INTEN(R/W) (0x8818001C): SD Card Interrupt Register

This register is used to enable/disable the interrupt of the SD memory card controller.

Name	B15-B7	B6	B5	B4	B3	B2	B1	B0
P_SD_INTEN		INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0

**INTEN0:** 0:Disable CMDCOM interrupt  
1:Enable CMDCOM interrupt, fire interrupt when **CMDCOM** set.

INTEN1:	0:Disable <b>DATCOM</b> interrupt.
	1:Enable <b>DATCOM</b> interrupt, fire interrupt when <b>DATCOM</b> set.
INTEN2:	0:Disable <b>CMDBUFFFULL</b> interrupt.
	1:Enable <b>CMDBUFFFULL</b> interrupt, fire interrupt when <b>CMDBUFFFULL</b> set.
INTEN3:	0:Disable <b>DATBUFFFULL</b> interrupt.
	1:Enable <b>DATBUFFFULL</b> interrupt, fire interrupt when <b>DATBUFFFULL</b> set.
INTEN4:	0:Disable <b>DATBUFEMPTY</b> interrupt.
	1:Enable <b>DATBUFEMPTY</b> interrupt, fire interrupt when <b>DATBUFEMPTY</b> set.
INTEN5:	0:Disable card insert/remove interrupt.
	1:Enable card insert/remove interrupt.
INTEN6:	0:Disable SD IO card interrupt.
	1:Enable SD IO card interrupt..

## 32 UNIVERSAL FILE SYSTEM – UFAT LIBRARY

UFAT, Universal FAT, file system can be used in most of the embedded applications with file-like storage. The mainly UFAT API functions of SPG290 UFAT LIB are listed as following

API name	Description
Open	Open the specified file with the specified operation mode.
Read	Read bytes from the file specified by a file node index
Write	Write bytes into the file specified by a file node index
Close	Close the file specified by the file node index

### 32.1 Structure

UFAT is designed as the following structure:

<b>Interface Layer</b>	APIs for the application developers
<b>API Layer</b>	Operations for files and directories
<b>File System Layer</b>	Operation for FAT and clusters
<b>Logic Block Layer</b>	Sector based access and caching
<b>Driver Layer</b>	Low level device access

**Interface Layer:** A thin layer translates the API layer function calls and error numbers into POSIX function calls and error numbers.

**File System Layer:** Operates the FAT and clusters, follow the FAT storage rules.

**Logic Block Layer:** A sector base access interface is defined in this layer to read sectors or write sectors with cache.

**Driver Layer:** Provide a hardware independent access interface for the upper layer. SPG290 UFAT library support both **FAT16** and **FAT32** storage formats

### 32.2 API Functions

#### 32.2.1 open

Open a specified file.

int open (const char \*filename, int flags)

**Parameters:**

**Filename:** pointer to a path name string.

**Flags:**

Flags	Description
O_OPEN	Open a exists file
O_TRUNC	Open the file and truncate the file to zero length
O_CREAT	If set, the file will be created if it doesn't already exist
O_RDONLY	Open the file for read access
O_WRONLY	Open the file for write access
O_RDWR	Open the file for both reading and writing
O_EXCL	If set, the open operation will fail when trying to create and existent file

**Returns value:**

When success, the return value of open() is a non-negative integer file handle.

Return -1 when error occurs.

### 32.2.2 close

Close a file handle..

int close (int filehd)

**Parameters:**

**filehd:** integer file handle.

**Return value:**

When success, return 0. -1 when error occurs.

### 32.2.3 read

Read bytes from the specified file handle.

Int read (int filehd, void \*buffer, unsigned int size)

**Parameters:**

**filehd:** Integer file handle returned from open function.  
**buffer:** Pointer to the buffer for storing the data.  
**size:** bytes to read.

**Return values:**

When success, it will return the actual number of bytes readed from the specified file handle. Return -1 when error occurs.

#### 32.2.4 write

Write bytes into a specified file handle.

Int write ( int filehd, void \*buffer, unsigned int size)

**Parameters:**

**filehd:** Integer file handle returned from open function.  
**buffer:** Pointer to the buffer for writing.  
**size:** bytes to write.

**Return value:**

When success, it will return the actual number of bytes written to the specified file handle.  
Return -1 when error occurs.

## 33 NAND-TYPE FLASH

### 33.1 INTRODUCTION

There is a NAND-type flash controller embedded in SPG290. The NAND-type flash is suitable for mass storage application. The controller support standard 8-bits NAND flash equipped with ECC calculation circuit. The smart media card is also supported.

### 33.2 FEATURE

- Fully programmable CLE and ALE sequences which can support varies kind of NAND flash.
- Support polling/interrupt/DMA access method.
- Support page size from 528 bytes to 2112 bytes.
- Hardware ECC calculation circuit.
- Programmable read/write cycles.
- Programmable ALE cycles.

### 33.3 BLOCK DIAGRAM

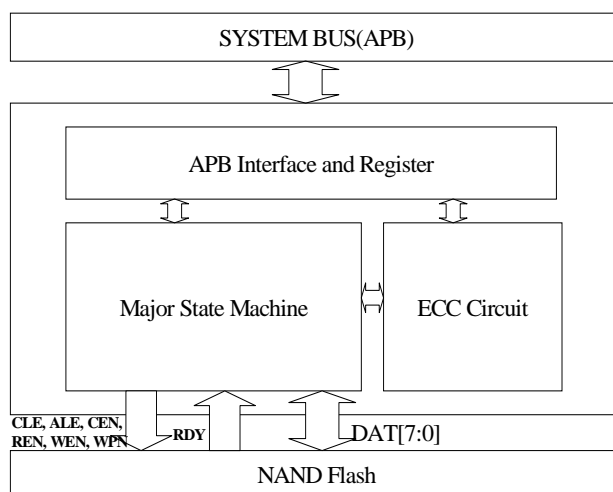


Figure 1. Functional Block Diagram of Flash Controller

### 33.4 INTERFACE SIGNAL

#### 33.4.1 APB Bus Signals

This part is the same as define in AMBA specification.

#### 33.4.2 Flash Bus Signals

CEN	O	Chip select of NAND flash, low active
CLE	O	Command latch enable, high active

ALE	O	Address latch enable, high active
WEN	O	Write enable, low active
REN	O	Read enable, low active
WPN	O	Write protect enable, low active
RDY	I	NAND flash ready input, high active
DATA[7:0]	I/O	Data bus

### 33.4.3 Other Signals

INT	O	Interrupt pin, high active
DMARQ	O	DMA request signal, high active

## 33.5 Control Registers

### 33.5.1 P\_FL\_CR(R/W)(0x8800F000): Control Register

Flash controller control register. This register is used to control the behavior of controller. For example, the timing setting of page size setting. Programming must program this register based on the specification of NAND flash. Host uses this register to control the behavior of the controller.

NAME	D7-D6	D5-D4	D3	D2	D1	D0
P_FL_CR	ALECYC	RDTYPE	WRCMD	FLWPN	FLCEN	FLEN
NAME	D31-D28	D27-D16	D15-D14	D13-D12	D11-D10	D9-D8
P_FL_CR		TBYTES	RDHIGH	RDLOW	WRHIGH	WRLOW

FLEN:	Flash interface enable register.
0:	Disable
1:	Enable
FLCEN:	CEN control register.
0:	CEN output 0
1:	CEN output 1
FLWPN:	WPN control register.
0:	WPN output 0
1:	WPN output 1
WRCMD:	Read/Write command control
0:	Read Command
1:	Write Command
RDTYPE:	Indicate when to start the read process
00:	Read after RDY rise
01:	Read after CLE fall
10:	Read after ALE fall
11:	Reserved

ALECYC:	Indicate how many writes will be issued when P_FL_ALE is written
00:	One write cycle
01:	Two write cycles
10:	Three write cycles
11:	Four write cycles
WRLOW:	Indicate the clock cycles of WEN low pulse
00:	One Clock cycle
01:	Two Clock cycle
10:	Three Clock cycles
11:	Four Clock cycles
WRHIGH:	Indicate the clock cycles of WEN high pulse.
00:	One clock cycle.
01:	Two clock cycles.
10:	Three clock cycles.
11:	Four clock cycles.
RDLOW:	Indicate the clock cycles of REN low pulse.
00:	One clock cycle.
01:	Two clock cycles.
10:	Three clock cycles.
11:	Four clock cycles.
RDHIGH:	Indicate the clock cycles of REN high pulse.
00:	One clock cycle.
01:	Two clock cycles.
10:	Three clock cycles.
11:	Four clock cycles.
TBYTES:	Total bytes -1 of this transfer

### 33.5.2 P\_FL\_CLE(R/W)(0x8800F004): Command Latch enable register

This register is used to control the CLE of NAND flash bus. When programmer write data to this register when controller is not busy, a CLE pulse will be issued with command data written in this register.

NAME	D7-D0
------	-------

P_FL_CLE	FL_CLE
----------	--------

FL\_CLE: When Programmer write data to this register when controller is not busy, a CLE pulse will be issued with command data written in this register

### 33.5.3 P\_FL\_ALE(R/W)(0x8800F008): Address latch enable register

This register is used to control the ALE of NAND flash bus. When programmer write data to this register when controller is not busy, a ALE pulse will be issued with address data written in this register. The cycles of ALE command is defined in FL\_CR register.

NAME	D31-D0
P_FL_ALE	FL_ALE

FL\_ALE: When programmer write data to this register when controller is not busy, a ALE pulse will be issued with command data written in this register.

### 33.5.4 P\_FL\_WD(R/W)(0x8800F00C): Write data register

This register is used to write data to NAND flash bus. Programmer must write data to this register when WREMPY is 1. Otherwise, the WRLOSS bit will be set and the written data will loss.

NAME	D31-D0
P_FL_WD	FL_WD

FL\_WD: This register is used to write data to NAND flash bus. Programmer must write data to this register when WREMPY is 1.

### 33.5.5 P\_FL\_RD(R/W)(0x8800F010): Read data register

This register is used to read data from NAND flash bus. Programmer must read data from this register when RDFULL is 1. Otherwise, the RDMISS bit will be set and the read data is invalid.

NAME	D31-D0
P_FL_RD	FL_RD

FL\_RD: This register is used to read data from NAND flash bus. Programmer must read data from this register when RDFULL is 1.

### 33.5.6 P\_FL\_INTEN(R/W)(0x8800F014): Interrupt enable register

Host uses this register to control the interrupt output.

NAME	D5	D4	D3	D2	D1	D0
P_FL_INTEN	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0

INTEN0: Write Empty Interrupt enable select

	0:	Disable
	1:	Enable
INTEN1:	Read Full Interrupt enable select	
	0:	Disable
	1:	Enable
INTEN2:	Ready Rise Interrupt enable select	
	0:	Disable
	1:	Enable
INTEN3:	Write Loss Interrupt enable select	
	0:	Disable
	1:	Enable
INTEN4:	Read Miss Interrupt enable select	
	0:	Disable
	1:	Enable
INTEN5:	Command Loss Interrupt enable select	
	0:	Disable
	1:	Enable

### 33.5.7 P\_FL\_INTSTS(R/C)(0x8800F018): Interrupt status register

The register is used to read-out the status of flash controller.

NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_FL_INTSTS	RDY	BUSY	CMDLOSS	RDMISS	WRLOSS	RDYRISE	RDFULL	WREMP TY

WREMP TY:	Write Empty Interrupt status							
	Read 0: Write buffer is full, further write to P_FL_WD will cause Write Loss become 1, and the write data will loss.							
	Read 1: Write buffer is empty, further write is allowed.							
	<b>NOTE:</b> This bit will be clear automatically when the write buffer is written.							
RD_FULL:	Read Full Interrupt status							
	Read 0: Read buffer is empty, further read from P_FL_RD will cause Read Miss become 1, and the read data is invalid.							
	Read 1: Read buffer is full, a read from FL_RD is allowed.							
	<b>NOTE:</b> This bit will be clear automatically when the read buffer is read.							
RDYRISE:	Ready Rise interrupt status							
	Read 0: RDY rise is not happen.							
	Read 1: RDY rise is happen.							
	Write 0: No effect.							
	Write 1: Clear this bit.							

WRLOSS:	Write Loss Interrupt status
	Read 0: Write loss event not happen.
	Read 1: Write loss event happen.
	Write 0: No effect.
	Write 1: Clear this bit.
RDMISS:	Read Miss Interrupt status
	Read 0: Read miss event not happen.
	Read 1: Read miss event happen.
	Write 0: No effect.
	Write 1: Clear this bit.
CMDLOSS:	Command Loss Interrupt status
	Read 0: Command loss event not happen.
	Read 1: Command loss miss event happen.
	Write 0: No effect.
	Write 1: Clear this bit.
BUSY:	Controllers busy flag
	0: Controller is not busy, ALE/CLE command is allowed.
	1: Controller is busy, ALE/CLE command is not allowed. If a ALE/CLE command is received in this state, CMDLOSS will be set to 1 and the command will loss.
RDY:	NAND flash busy flag, this bits will shows the status of RDY on the NAND flash bus.
	0: NAND flash is busy.
	1: NAND flash is not busy.

### 33.5.8 P\_FL\_TRUELP(R/W)(0x8800F01C): True line parity register

Programmer can use this register to write the true LP result for ECC circuit to calculate the error bits in a page.

NAME	D31-D16	D15-D0
P_FL_TRUELP	TRUELP1	TRUELP0

TRUELP0: The true line parity of bytes 0~255 in a page. This register is written by programmer and used in error line/bit calculation

TRUELP1: The true line parity of bytes 256~511 in a page. This register is written by programmer and used in error line/bit calculation.

### 33.5.9 P\_FL\_TRUECP(R/W)(0x8800F020): True column parity register

Programmer can use this register to write the true CP result for ECC circuit to calculate the error bits

in a page.

NAME	D11-D6	D5-D0
<b>P_FL_TRUECP</b>	TRUECP1	TRUECP0

TRUECP0: The true column parity of bytes 0~255 in a page. This register is written by programmer and used in error line/bit calculation.

TRUECP1: The true column parity of bytes 256~511 in a page. This register is written by programmer and used in error line/bit calculation.

### 33.5.10 P\_FL\_CALLP(R)(0x8800F024): Calculate line a parity register

Programmer can read the line parity of ECC calculation result from this register.

NAME	D31-D16	D15-D0
<b>P_FL_CALLP</b>	CALLP1	CALLP0

CALLP0: The line parity of bytes 0~255 in a page calculated by ECC circuit. This register is written by ECC circuit and used in error line/bit calculation.

CALLP1: The line parity of bytes 256~511 in a page calculated by ECC circuit. This register is written by ECC circuit and used in error line/bit calculation.

### 33.5.11 P\_FL\_CALCP(R)(0x8800F028): Calculate column parity register

Programmer can read the column parity of ECC calculation result from this register.

NAME	D11-D6	D5-D0
<b>P_FL_CALCP</b>	CALCP1	CALCP0

CALCP0: The column parity of bytes 0~255 in a page calculated by ECC circuit. This register is written by ECC circuit and used in error line/bit calculation.

CALCP1: The column parity of bytes 256~511 in a page calculated by ECC circuit. This register is written by ECC circuit and used in error line/bit calculation.

### 33.5.12 P\_FL\_ECCSTS(R)(0x8800F02C): ECC status register

The error line and error bit will be shown on this register.

NAME	D31-D30	D29-D28	D26-D24	D23-D16	D15-D14	D13-D11	D10-D8	D7-D0
<b>P_FL_ECCSTS</b>	ERR1		ERRBIT1	ERRBYT E1	ERR0		ERRBIT0	ERRBYT E0

ERRBYTE0: The error byte of bytes 0~255 detected by ECC circuit

ERRBIT0: The error bit of ERRBYTE0 detected by ECC circuit.

ERR0: Error detection of byte 0~255 in a page.

00: No error is detected

01: One error is detected, correctable.

10: Reserved

11: Two or more errors is detected, non-correctable.

ERRBYTE1: The error byte of bytes 256~511 detected by ECC circuit

ERRBIT1: The error bit of ERRBYTE1 detected by ECC circuit.

ERR1: Error detection of byte 256~511 in a page.

00: No error is detected

01: One error is detected, correctable.

10: Reserved

11: Two or more errors is detected, non-correctable.

### 33.5.13 P\_FL\_CALECC(W)(0x8800F030): ECC control register

This register is used to control the ECC circuit.

NAME	D2	D1	D0
P_FL_CALECC	CLRECC1	CLRECC0	CALECC

**CALECC:** Write 0: No effect.

Write 1: The ECC circuit will use FL\_TRUECP/FL\_TRUECP and FL\_CALLP/FL\_CALLP to find the error byte and error bit.

This bit will be clear after the calculation is done.

**CLRECC0:** Write 0: No effect.

Write 1: The ECC part 0 will be reset.

This bit will be clear after the calculation is done. This bit is used when the page size is larger than 528 bytes.

**CLRECC1:** Write 0: No effect.

Write 1: The ECC part 1 will be reset.

This bit will be clear after the calculation is done. This bit is used when the page size is larger than 528 bytes.

## 34 TFT LCD CONTROL

### 34.1 Introduction

The built-in TFT LCD interface supports multiple TFT LCD panel input format, such as DataEnable(DE) mode, Hsync/Vsync mode, 15-bit parallel RGB mode, 8-bit delta RGB mode and CCIR601/656 mode etc., which with resolution 320(H) X 240(V) and NTSC/PAL formats. The configurable position and width of synchronous signal can fit various TFT LCD panel specifications.

### 34.2 Output interface

The following is the IO pins related to TFT controller:

- Clk: TFT LCD clock
- Data\_en: data enable
- Vsync: vertical synchronous signal
- Hsync: horizontal synchronous signal
- Data[15:0]: data bus

### 34.3 Supported data format

The LCD TFT controller supports the following data formats:

- Parallel RGB
- Serial RGB
- Serial RGBDm
- CCIR-601/CCIR-656

### 34.4 Clock frequency

- System Clock(27 MHz)
- System Clock/2(13.5 MHz)
- System Clock/4(6.75 MHz)
- System Clock/8(3.375 MHz)

### 34.5 REGISTER DEFINITION

#### 34.5.1 P\_TFT\_CTRL(0x88040000):

NAME	D24	D23-D12	D11-D10	D9-D8	D7-D2	D1-D0
P_TFT_CTRL	TFT_EN		VER_SCALING	HOR_SCALING		TFT_CLK_SEL

TFT\_EN: TFT LCD Enable

VER\_SCALING: Image Scaling In Vertical

0= no scaling

1= the display image in vertical is scaling up to double size

2= the display image size in vertical is scaling down

HOR\_SCALING: Image scaling in horizontal

0= no scaling

1= the display image in horizontal is scaling up to double size

2= the display image is scaling down to half size

3= no scaling

TFT\_CLK\_SEL: TFT LCD output clock selection

0 : system clock (27 MHz)

1 : system clock /2 (13.5 MHz)

2 : system clock /4 (6.75 MHz)

3 : system clock/8 (3.875 MHz)

#### 34.5.2 P\_DATA\_FMT(0x88040004):

NAME	D8	D7-D3	D2-D0
P_DATA_FMT	CCIR656_M		TFT_FMT

CCIR656\_M: This function is for standard CCIR656.

When this bit sets to “1”, the controller will output 720 valid pixels. but it contains 80 pixels black in two side of picture.

When this bit sets to “0”, the controller will output 640 valid pixels.

TFT\_FMT: TFT LCD output data format

0 : parallel RGB

1 : serial RGB

2 : serial RGBDm

3 : CCIR 601

4 : CCIR 656

#### 34.5.3 P\_HOR\_ACT(0x88040008):

NAME	D10-D0
P_HOR_ACT	HOR_ACT

HOR\_ACT: The horizontal active area.

Unit : system clock

#### 34.5.4 P\_HOR\_FBLK(0x8804000C):

NAME	D9-D0
P_HOR_FBLK	HOR_FBLK

HOR\_FBLK: The front horizontal blanking area

Unit : system clock

#### 34.5.5 P\_HOR\_BBLK(0x88040010):

NAME	D9-D0
P_HOR_BBLK	HOR_BBLK

HOR\_BBLK: The back horizontal blanking area

Unit : system clock

#### 34.5.6 P\_HOR\_SYNCW(0x88040014):

NAME	D24	D23-D8	D7-D0
P_HOR_SYNCW	HS_P		HOR_SYNCW

HS\_P: The polarity of horizontal synchronous plus.

0 : negative

1 : positive

HOR\_SYNCW: The width of horizontal synchronous plus

Unit : system clock

#### 34.5.7 P\_VER\_ACT(0x88040018):

NAME	D9-D0
P_VER_ACT	VER_ACT

VER\_ACT: The vertical active area.

Unit : line

#### 34.5.8 P\_VER\_FBLK(0x8804001C):

NAME	D7-D0
P_VER_FBLK	VER_FBLK

VER\_FBLK: The front vertical blanking area.

Unit : line

#### 34.5.9 P\_VER\_BBLK(0x88040020):

NAME	D7-D0
P_VER_BBLK	VER_BBLK

VER\_BBLK: The back vertical blanking area.

Unit : line

#### 34.5.10 P\_VER\_SYNCW(0x88040024):

NAME	D24	D23-D5	D4-D0
P_VER_SYNCW	VS_P		VER_SYNCW

VS\_P: The polarity of vertical synchronous plus.

0 : negative

1 : positive

VER\_SYNCW: The width of vertical synchronous plus

Setting range : 1 ~ 31

#### 34.5.11 P\_FB\_ATTRIB(0x88040028):

NAME	D1-D0
P_FB_ATTRIB	FB_FMT

FB\_FMT: Frame buffer data format.  
0 : RGB565 (little endian)  
1 : RGB1555 (big endian)  
2 : YUYV (little endian)  
3 : 4Y4U4Y4V (little endian)

#### 34.5.12 P\_STR\_LNO(0x8804002C):

NAME	D9-D0
P_STR_LNO	STR_LNO

STR\_LNO: The number of start line in frame buffer.  
Unit : line

#### 34.5.13 P\_STR\_PNO(0x88040030):

NAME	D9-D0
P_STR_PNO	STR_PNO

STR\_PNO: The number of start pixel in frame buffer.  
it must be the multiple of 16.  
Unit : pixel

#### 34.5.14 P\_PIX\_NUM(0x88040034):

NAME	D9-D0
P_PIX_NUM	PIX_NUM

PIX\_NUM: The number of image pixel of one line in frame buffer.  
it must be the multiple of 16.  
Unit : pixel(16 bits)

#### 34.5.15 P\_DUMMY\_PIX (0x88040038):

NAME	D9-D0
P_DUMP_PIX	DUMP_PIX

DUMP\_PIX: The number of dummy pixel in frame buffer.  
it must be the multiple of 16.  
Unit : pixel(16 bits)

#### 34.5.16 P\_TFT\_ST (0x8804003C):

NAME	D24	D23-D17	D16	D15-D0
P_TFT_ST	BUF_ERR		TFT_FINISH	

BUF\_ERR: TFT LCD buffer underflow.  
 When reading is faster than writing in LCD buffer, it will be set "1" by TFT LCD controller.  
 Read 0= Not Occurs  
 Read 1= Happened  
 Write 1= Clear the flag

TFT\_FINISH: When this bit is set "1", it means the TFT controller complete internal operation. Then user can turn off the TFT clock.

#### 34.5.17 P\_DATA\_SEQ (0x88040040):

NAME	D24	D23-D19	D18-D16	D15-D11	D10-D8	D7-D3	D2-D0
P_DATA_SEQ	YUV_FMT		YUV_SEQ		RGB_OLS EQ		RGB_ELS EQ

YUV\_FMT: Output data is YUV/YCbCr.  
 0 : YCbCr  
 1 : YUV

YUV\_SEQ: YUV data output sequence for CCIR601 and CCIR656  
 000 : Y0U0Y1V0 / Y0Cb0Y1Cr0  
 001 : Y0V0Y1U0 / Y0Cr0Y1Cb0  
 010 : U0Y0V0Y1 / Cb0Y0Cr0Y1  
 011 : V0Y0U0Y1 / Cr0Y0Cb0Y1  
 100 : Y1U0Y0V0 / Y1Cb0Y0Cr0 (reserved for test)  
 101 : Y1V0Y0U0 / Y1Cr0Y0Cb0 (reserved for test)  
 110 : U0Y1V0Y0 / Cb0Y1Cr0Y0 (reserved for test)  
 111 : V0Y1U0Y0 / Cr0Y1Cb0Y0 (reserved for test)

RGB\_OLSEQ: RGB data output sequence in odd line for serial RGB and serial RGBDm.  
 000 : RGB (RGBDm)  
 001 : GBR (GBRDm)  
 010 : BRG (BRGDm)  
 011 : RBG (RBGDm)  
 100 : BGR (BGRDm)  
 101 : GRB (GRBDm)

RGB\_ELSEQ: RGB data output sequence in even line for serial RGB and serial RGBDm.  
 000 : RGB (RGBDm)  
 001 : GBR (GBRDm)

010 : BRG (BRGDm)

011 : RBG (RBGDm)

100 : BGR (BGRDm)

101 : GRB (GRBDm)

#### 34.5.18 P\_TFT\_INT (0x88040050):

NAME	D24	D23-D17	D16	D15-D0
P_TFT_INT	INT_EN		VLBK_INT	

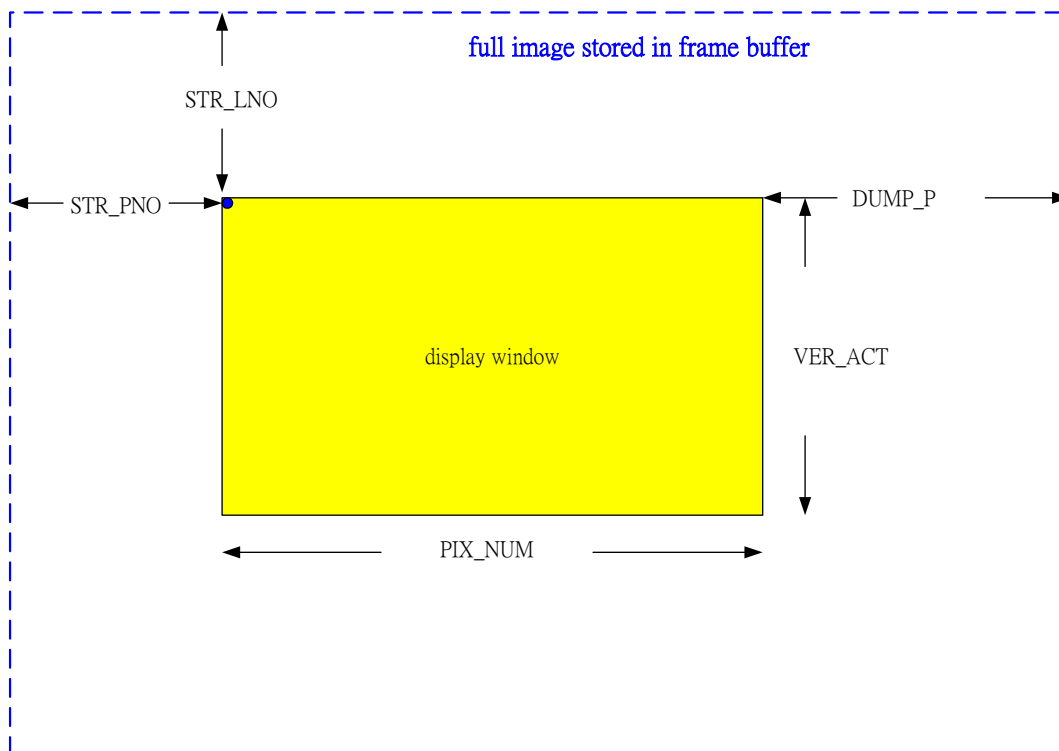
INT\_EN: The blanking interrupt of vertical enable.

VLBK\_INT: The blanking interrupt of vertical.

Read 0= Not Occurs

Read 1= Happened

Write 1= Clear the flag



## 35 STN LCD CONTROL

### 35.1 Introduction

The SPG290 contains a powerful LCD controller, which can support resolution up to 320(H) X 320(V). The LCD STN controller supports 4096 colors for color STN. It also contains a built-in hardware scroll function to reduce software overhead. Moreover, the interface supports flexible 4-bit or 8-bit data interface to connect with variety of LCD panels.

### 35.2 Feature

1. support resolution up to 320(H) X 320(V) LCD panel.
2. support 4096 color STN

### 35.3 Interface Signals

Name	I/O	Description
FP	O	LCD interface, frame signal
LP	O	LCD interface, line signal
LACD	O	LCD interface, alternating signal
LD[7:0]	O	LCD interface, data bus
LCLK	O	LCD interface, dot clock

### 35.4 STN LCD Control Register.

#### 35.4.1 P\_STN\_CTRL (0x88041000):

NAME	D28	D27-D25	D24	D22-D17	D16	D15-D9	D8	D7-D2	D1-D0
<b>P_STN_CTRL</b>	FPSIF		FPIEN		LCDEN		SELREF		BUSW

**FPSIF:** STN LCD FP Signal Interrupt Flag.  
 FP interrupt flag asserts when FP rising edge, and is clear by writing "1" to this bit.  
 Read 0= Not Occurs  
 Read 1= Happened  
 Write 1= Clear the flag

**FPIEN:** STN LCD FP Signal Interrupt Enable  
 If this bit is set to "1", and FP interrupt happened, hardware will issue an IRQ to CPU. If this bit is clear to "0", this interrupt will be mask

**LCDEN:** STN LCD Enable

**SELREF:** Self Refresh Mode Enable  
 0: Don't enter self-refresh mode  
 1: Enter self-refresh mode

When the LCD driver contains the built-in memory and supports the self-refresh mode, the LCDC module can be configured into self-refresh mode. The LCD driver shows the last display data, and the LCDC outputs LACD, FP, and LP signals only.

The LCDCLK and LD signal will keep at LOW state until SelfRef=0.

BUSW: LCD hardware data bus width configuration.

4/8 bit-width can be supported.

00= Reserved

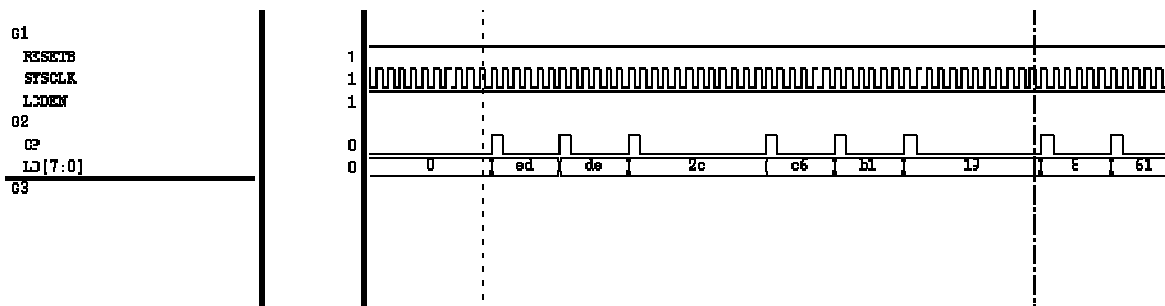
01= 4bit (LCDD[3:0] Valid)

10= 8bit (LCDD[7:0] Valid)

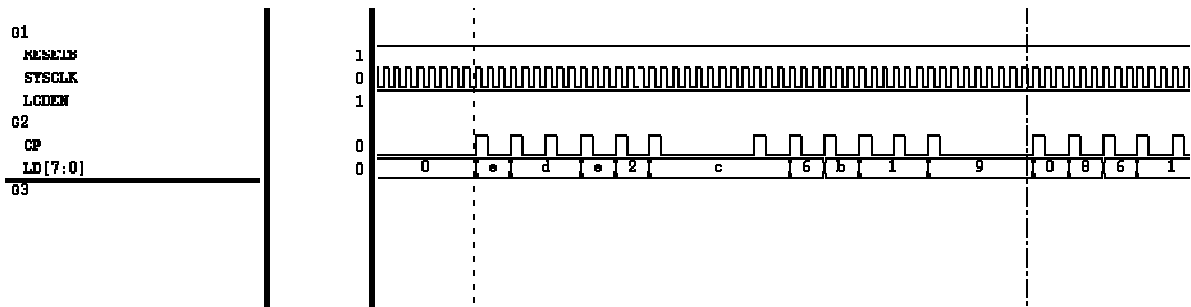
11= Reserved

The data is re-packaged into 4, or 8 bits data bus with the control of BUSW register and it is transferred to LCD driver through LD [3:0], or LD [7:0] respectively.

Color, Bus width=8



Color, Bus width=4



#### 35.4.2 P\_LCDCLK (0x88041004):

NAME	D9-D0
P_LCDCLK	LCDCLK

LCDCLK: Pixel Clock Divider

The SYSCLK signal is divided by N (= PCD [9:0] +2) to generate LCD clock.

where  $N = \text{SYSCLK} / (\text{COM} * \text{SEG} * \text{Frame Rate})$

For example: SYSCLK = 48MHZ

COM * SEG * Frame Rate	N
16 * 16 * 183	1025
80 * 80 * 178.6	42
160 * 160 * 187.5	10

COM: Number of common

SEG: Number of segment

Frame rate: the frequency of alternating signal (LACD).

#### 35.4.3 P\_STN\_SEG\_NUM (0x88041008):

NAME	D8-D0
P_STN_SEG_NUM	LCDSEG

LCDSEG: LCD Panel Segment Number Register

it must be the multiple of 16.

The horizontal size (in pixel) of the LCD panel, LCDSEG [8:0]= 16 ~ 320.

#### 35.4.4 P\_STN\_COM\_NUM (0x8804100C):

NAME	D8-D0
P_STN_SEG_NUM	LCDCOM

LDCOM: LCD Panel Common Number Register

LINEVAL [8:0]: The vertical size of the LCD panel where LINEVAL [8:0]= 16 ~ 320.

#### 35.4.5 P\_STN\_STR\_LNO(0x88041010):

NAME	D9-D0
P_STN_STR_LNO	STN_STR_LNO

STN\_STR\_LNO: The Start line number in frame buffer

Unit : line

#### 35.4.6 P\_STN\_STR\_PNO(0x88041014):

NAME	D9-D0
P_STN_STR_PNO	STN_STR_PNO

STN\_STR\_PNO: The Start pixel number in frame buffer

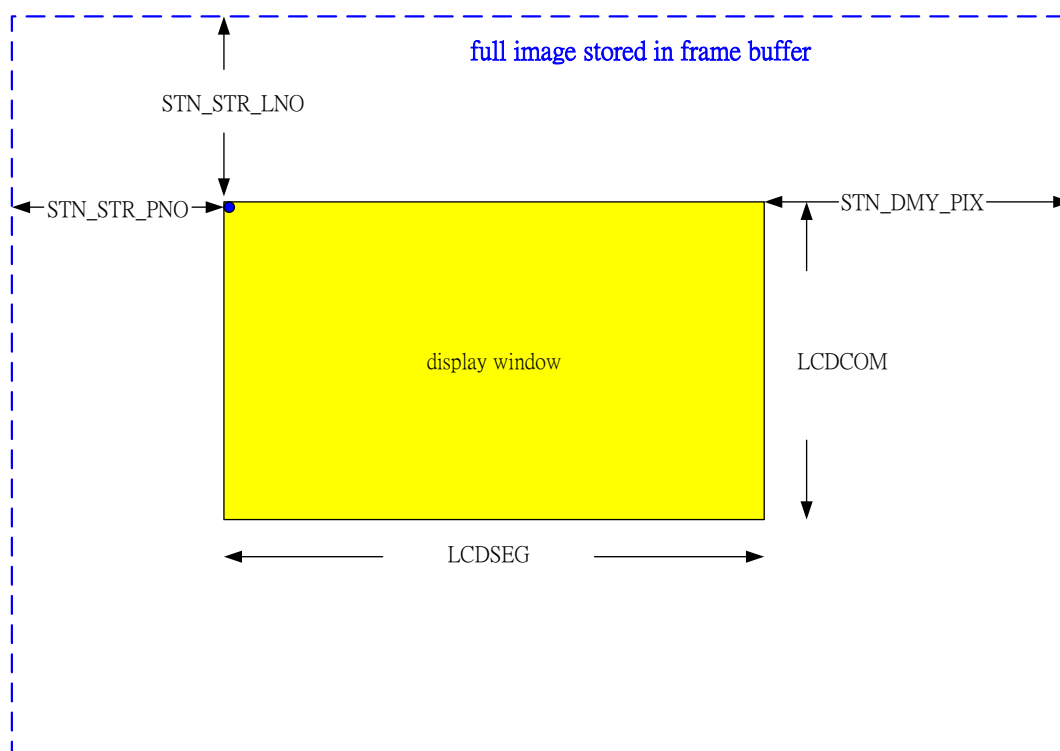
it must be the multiple of 16.

Unit : line

### 35.4.7 P\_STN\_STR\_PNO(0x88041018):

NAME	D9-D0
P_STN_DMY_PIX	STN_DMY_PIX

STN\_DMY\_PIX: The dummy pixel number in frame buffer  
it must be the multiple of 16.  
Unit : pixel(16 bits)



### 35.4.8 P\_STN\_LT(0x8804101C):

NAME	D19-D16	D15-D12	D11-D8	D7-D4	D3-D0
P_STN_LT	LBVL		LPW		LPCPD

LBVL: Line Blank Width  
 $T = (LBVL + 1) \times CLCPCLK$

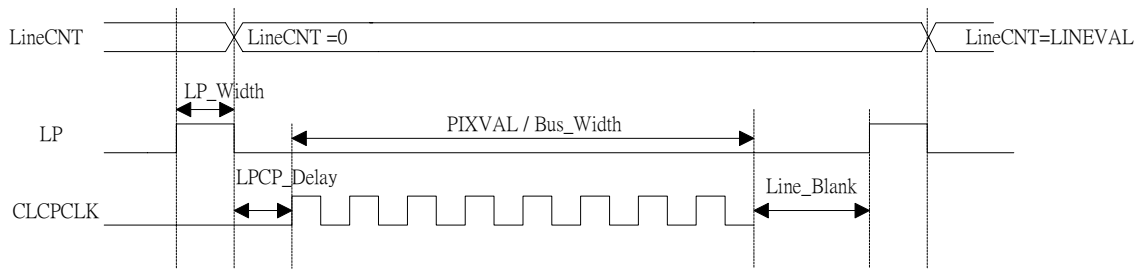
LPW: LP Pulse Width  
 $T = (LVW + 1) \times CLCPCLK$

LPCPD: LP to CP Delay  
 $T = (LPCPD + 1) \times CLCPCLK$

LBVL [3:0]: Line Blank

LPW [3:0]: LP pulse width

LPCPD [3:0]: LP to CP delay



1. LBVL = 0H ~ FH  
Actual Line Blank width= (LBVL + 1) \* CLCPCLK cycle time
2. LPW = 0H ~FH  
Actual LP Pulse Width= (LPW + 1) \* CLCPCLK cycle time
3. LPCPD = 0H ~FH  
Actual LP to CP Delay= (LPCPD + 1) \* CLCPCLK cycle time

#### 35.4.9 P\_STN\_FM(0x88041020):

NAME	D24	D23-D8	D7-D0
P_STN_FM	BCMOD		MVAL

BCMOD: LCD Frame Modulation Type Select

0= B type

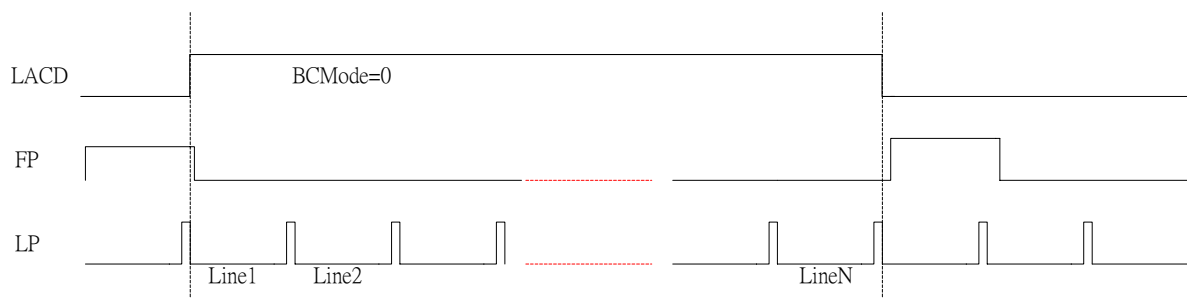
1= C type

MVAL: Define the frequency of Frame Modulation when C type is active

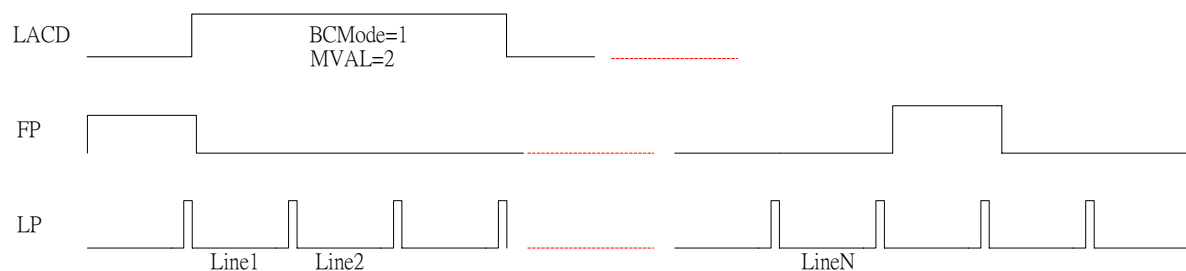
When BCMODE=1, the frequency of LACD depends on MVAL +1.

That is, LACD changes its state every MVAL + 1 line.

- When BCMoDe=0 (B type), LACD changes its state each FP signal.



- When BCMoDe=1 and MVAL=2, LACD changes its state each three (=MVAL+1) LP signals.



#### 35.4.10 P\_STN\_BUF\_CTRL(0x88041024):

NAME	D24	D23-D17	D16	D15-D0
P_STN_BUF_CTRL	BUF_ERR		STN_FINISH	

BUF\_ERR: TFT LCD buffer underflow.

When reading is faster than writing in LCD buffer, it will be set "1" by STN LCD controller.

Read 0= Not Occurs

Read 1= Happened

Write 1= Clear the flag

STN\_FINISH: When this bit is set "1", it means the STN controller complete internal operation. Then user can turn off the TFT clock.

#### 35.4.11 P\_STN\_ATTR (0x88041028):

NAME	D12	D11-D9	D8	D7-D2	D1-D0
P_STN_ATTR	VER_SCLDN		HOR_SCLDN		FB_FMT

VER\_SCLDN: 0: no scaling.  
1: the display image is scaling down to half size in vertical.

HOR\_SCLDN: 0: no scaling.  
1: the display image is scaling down to half size in horizontal.

FB\_FMT: Frame buffer data format  
0 : RGB565 (little endian)  
1 : RGB1555 (big endian)  
2 : YUYV (little endian)  
3 : 4Y4U4Y4V (little endian)

#### 35.4.12 P\_STN\_MSC(0x8804102C):

NAME	D24	D23-D11	D10-D8	D7-D3	D2-D0
P_STN_MSC	MOSAIC_EN		RGB_OLSEQ		RGB_ELSEQ

MOSAIC\_EN: Mosaic Color STN LCD Enable

RGB\_OLSEQ: RGB data output sequence in odd line for mosaic color STN LCD.

000 : RGB  
001 : GBR  
010 : BRG  
011 : RBG  
100 : BGR  
101 : GRB

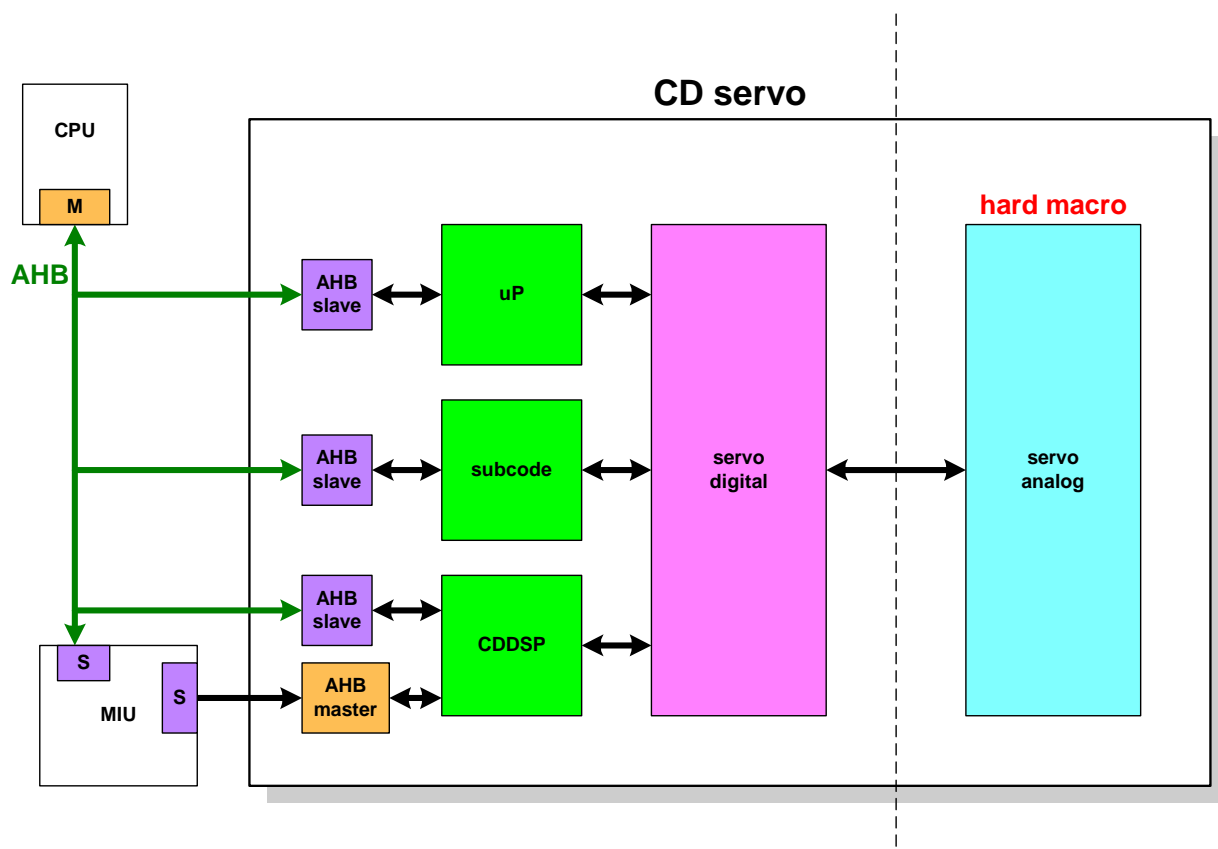
RGB\_ELSEQ: RGB data output sequence in even line for mosaic color STN LCD.

000 : RGB  
001 : GBR  
010 : BRG  
011 : RBG  
100 : BGR  
101 : GRB

## 36 CD SERVO

### 36.1 Block Diagram of CD Servo

## Block Diagram of CD Servo



### 36.2 Control registers

About the CD interface Control register, it can be described by three parts, CD Servo, CDDSP, CDDMA. First part, CD Servo, SPG290 must use 8051 interface to communicate with servo. There are four registers defined to do this. Second part, CDDSP, this is the unit that controls and processes the data read from CD. Third part, CDDMA, after CDDSP processes the correct data, CDDMA moves it to SDRAM.

#### 36.2.1 P\_IF51\_CONFIG (R/W) (0X88060000) :

Define each parameter of 8051 communication signal. Define one time at the beginning, and the suggestion value is 0xff.

Name	B7-B6	B5-B4	B3-B2	B1-B0
P_IF51_CONFIG	M_CYCLE	N_CYCLE	P_CYCLE	Q_CYCLE

M\_CYCLE: Define 8051 M cycle number 1 2 3 4. (pre\_ale, post\_ale, post\_rw)

N\_CYCLE: Define 8051 N cycle number 2 4 6 8 cycle (ale acitve high\_width)

P\_CYCLE: Define 8051 P cycle number 2 4 6 8 (pre\_rw)

Q\_CYCLE: Define 8051 Q cycle number 2 4 8 16 (rw active\_low width)

### 36.2.2 P\_IF51\_ADDR (R/W) (0X88060004) :

Name	B15-B0
P_IF51_ADDR	IF51_ADDR

IF51\_ADDR: Although SPG290 only use 12 bit, it still need to write 16 bit to the register.  
For example : 0x0407, 0x053d

### 36.2.3 P\_IF51\_DATA (R/W) (0X88060008) :

Name	B7-B0
P_IF51_DATA	IF51_DATA

IF51\_DATA: The data read or write from 8051 is only 8 bit.

### 36.2.4 P\_IF51\_CONTROL (R/W) (0X8806000C) :

Name	B0
P_IF51_CONTROL	IF51_CONTROL

IF51\_CONTROL: Write this register to read or write the register of SERVO.

write operation: 0x00

read operation: 0x01

Hardware will start to read or write the register of SERVO by using 8051 signal. Because of the signal of 8051 is asynchronous (it is slower than our system (AHB Bus)), it must to polling this register to make sure that read or write has been finished. If the value read by this register is 0x1, it means finished, on the other hand means not finished.

### 36.2.5 P\_CDDSP\_CONFIG (R/W) (0X88060040) :

Name	B6		B5	B4	B3	B2	B1	B0
P_CDDSP_CONFIG	C2PO		LRCKMODE	LRCK	DELAY	JUSTIFY	LSB/MSB	BCLK
Name	B15	B14	B13	B12-B11		B10	B9-B7	
P_CDDSP_CONFIG	SYNC	DATASWAP	DESCRAMBLE	FRMSIZE		SPECIALCRC	DATASIZE	

BCLK: 0: Sample I2S\_Data by using the falling edge of I2S\_BCK  
1: Sample I2S\_Data by using the rising edge of I2S\_BCK

LSB/MSB: I2S data is serial input, this register can set MSB sent first or LSB. In SPG290, the data is MSB first

	0: I2S data is LSB first
	1: I2S data is MSB first
JUSTIFY:	If frame size larger than data size, it must set the data is left justify or right justify. In SPG290, the data is right justified.
	0: data is right justified.
	1: data is left justified.
DELAY:	Setting the timing of sending I2S data. It can set delay I2S_LRCK one cycle or zero. In SPG290, it is zero delay.
	0: zero delay
	1: data is delay one cycle
LRCK:	Transferring L-channel or R-channel when I2S_LRCK is low. In SPG290, it is setting to R-channel.
	0: low is L-channel.
	1: low is R-channel.
LRCKMODE:	The trigger mode of LRCK. In SPG290, it is level trigger
	0: pulse mode
	1: level trigger
C2PO:	Not used
DATASIZE:	The meaningful data length of each L-channel or R-channel. In SPG290, it is 16bit.
	000: 16bit
	001: 17bit
	010: 18bit
	011: 19bit
	100: 20bit
	101: 21bit
	110: 22bit
	111: 24bit
SPECIALCRC:	There are no CRC data (all zero) in ancient CD. Therefore, there is no CRC error when CRC data are zero that detected by hardware.
FRMSIZE:	The data length of each L-channel or R-channel. In SPG290, it is 24bit
	00: 16bit
	01: 24bit
	10: 32bit
DESCRAMBLE:	Descrambling each frame or not when reading. In CD-DA, it no necessary to set descramble on.
	0: descramble on (CD-ROM)
	1: raw data (no descramble) (CD-DA)
DATASWAP:	Setting the data output of CDDSP is 16-bit little-endian or big-endian. In SPG290, it is big-endian. But when write to SDRAM, it also transfer to 32-bit little-endian.
	0: 16 bit big-endian

1: 16 bit little-endian

SYNC:                      Checking 12 byte sync code of each frame by CDDSP or not. In SPG290, it is set to search SYNC code.

0: Search SYNC.

1: no search SYNC.

### 36.2.6 P\_CDDSP\_CONTROL (R/W) (0X88060044) :

Name	B6	B5	B4	B3	B2	B1	B0
P_CDDSP_CONFIG	LRCKEDGE	FREERUN	RECEIVE	SEEK	PAUSE	STOP	RESET
Name	B15-B8						
P_CDDSP_CONFIG	SECTOR COUNT						

RESET:                      write 1 to reset CDDSP

STOP:                        write 1 to stop receiving immediately.

PAUSE:                      write 1 to pause at a sector boundary, it means complete the recent frame, and pause at next frame.

SEEK:                        write 1 to enable the seek function, it means start to write the frame to SDRAM, and the initial frame is setting at CDDSP\_SEEK\_MM, CDDSP\_SEEK\_SS, CDDSP\_SEEK\_FF.

RECEIVE:                    write 1 to receive continuously(still wait LRCK)

FREERUN:                    write 1 for free-running(no wait LRCK)

LRCKEDGE:                  write 1 to start at both LRCK edge (audio swap), SPG290 must set to zero

0: only start hardware in LRCK=0

1: both will do

SECTOR COUNT:            Define the number of frame read by CDDSP, it can be set from 1-255. If set to zero means infinite frame.

### 36.2.7 P\_CDDSP\_SEEK\_MM (R/W) (0X88060048) :

Name	B31-B0
P_CDDSP_SEEK_MM	SEEK_MM

SEEK\_MM:                    packed BCD format seeking sector MINUTE address.

For example: 59 minute, 0x59.

### 36.2.8 P\_CDDSP\_SEEK\_SS (R/W) (0X8806004C) :

Name	B31-B0
P_CDDSP_SEEK_SS	SEEK_SS

SEEK\_SS:                    packed BCD format seeking sector SECONDS address.

For example: 59 second, 0x59.

### 36.2.9 P\_CDDSP\_SEEK\_FF (R/W) (0X88060050) :

Name	B31-B0
P_CDDSP_SEEK_FF	SEEK_FF

SEEK\_FF: packed BCD format seeking sector FRAME address.

For example: 74th frame, 0x59.

### 36.2.10 P\_CDDSP\_STATUS (R/W) (0X88060054) :

CDDPS hardware status, write one to clear.

Name	B9	B8	B7	B3-B2	B1	B0
P_CDDSP_STATUS	SEEK_LOST	SEEK_FOUND	OVERFLOW	BERR	LAST_ERR	CRC_ERR

CRC\_ERR: The CRC\_ERR aggregate at last fully-received frame.

LAST\_ERR: The last CRC\_ERR of

BERR: unused.

OVERFLOW: write buffer overflow has been occurred

SEEK\_FOUND: The frame number setting at CDDSP\_SEEK\_MM, CDDSP\_SEEK\_SS, CDDSP\_SEEK\_FF has been found.

SEEK\_LOST: SEEK has been failed, the flag will be record in this register.

### 36.2.11 P\_CD\_BUF\_BTM\_ADDR (R/W) (0X88060060) :

Name	B31-B0
P_CD_BUF_BTM_ADDR	BTM_ADDR

BTM\_ADDR: ring buffer bottom address (e.g. 0xa020\_0000), only need to be setting at the beginning.

### 36.2.12 P\_CD\_BUF\_TOP\_ADDR (R/W) (0X88060064) :

Name	B31-B0
P_CD_BUF_TOP_ADDR	TOP_ADDR

TOP\_ADDR: ring buffer top address (e.g. 0xa02F\_FFFF), only need to be setting at the beginning.

### 36.2.13 P\_CD\_WRITE\_PTR (R/W) (0X88060068) :

Name	B31-B0
P_CD_WRITE_PTR	WRITE_PTR

**WRITE\_PTR:** current write pointer, the same as CD\_BUF\_BTMM\_ADDR, it only need to be setting one time. Hardware will change this value automatically when doing DMA. The value will be increase progressively, and return when the value equal to the setting of CD\_BUF\_TOP\_ADDR.

#### 36.2.14 P\_CD\_FRAME\_SIZE(R/W) (0X8806006C) :

Name	B15-B0
P_CD_FRAME_SIZE	FRAME_SIZE

**FRAME\_SIZE:** It can only be setting to 1 frame = 2368 byte (0x940), because of the application on CD-DA/CD-ROM.

#### 36.2.15 P\_CD\_FRAME\_CNT (R/W)(0x88060070) :

Name	B15-B0
P_CD_FRAME_SIZE	FRAME_CNT

**FRAME\_CNT:** The value must be the same as the setting of sector count in P\_CDDSP\_CONTROL and it will be decreased one when translate one frame data to SDRAM. The interrupt will be occurred when the value decrease to zero.

### 36.3 Interrupt

There are two registers to control the CD Servo interrupts. One is **P\_CD\_IRQEN**, which is a interrupt mask register. When this register is set to '1', the interrupt signal will be masked, that is the signal will remain inside of CD Servo module and the CPU can not receive this signal, therefore it will not enter into the corresponding interrupt service routine.

The other register is **P\_CD\_IRQSTS**, which is a interrupt status register. When interrupt event is received, the CPU will query this status register to determine which interrupt event is happen, and enter into the corresponding interrupt service routine. The CPU will clear the interrupt event flag automatically when it read the status register.

## 37 CMOS SENSOR INTERFACE – CSI APPLICATION

### 37.1 Frame Buffer control

When CMOS Image Sensor captures an image, it will transmits into DRAM using CMOS Sensor Interface (CSI), then all display related modules including PPU, TV, LCD or JPG will be able access to this image data, and directly or mix it with its own data to generate output. Users do not need to wait the CMOS Sensor data transfer to complete before using these data because CSI already has an independent Frame Buffer, this mechanism makes it possible for each of the PPU, TV, LCD and JPG module to work independently. BUFCTL is responsible for all the buffer control of each module, and the following Output Ports are for CSI module setting:

Name	I/O	Function description
CSI_FRAME_END	output	A pulse to indicate that CSI has completed the write transfer of a frame. Every frame must have a FRAME_END pulse, even the frame is disabled by the DISABLE_FRAME input.

CSI\_FRAME\_END is driven by AHB master clock.

BUFCTL module has the following I/O port :

Name	I/O	function description
CSI_FRAME_END	Input	A pulse to indicate that CSI has completed the write transfer of a frame. Every frame must have a FRAME_END pulse, even the frame is disabled by the DISABLE_FRAME input.
CSI_BUFFER_PTR[1:0]	Output	2'b00: write CSI frame buffer A 2'b01: write CSI frame buffer B 2'b10: write CSI frame buffer C
DISABLE_CSI_FRAME	Output	Disable the write transfer for a whole frame, CSI holds HTRANS at IDLE. This input changes after the CSI_FRAME_END pulse.

CSI\_BUFFER\_PTR connects to MIU , DISABLE\_CSI\_FRAME connects to CSI .

### 37.2 Control registers

CPU can access the internal control registers of the CSI module by the AHB slave. CSI has a disable register, it will wait the completion of MIU managed AHB transfer and stop the data transfer. Using this status register bit, CSI will notify to CPU that the MIU managed AHB data transfer is finished when the AHB transfer is complete.

### 37.3 Time Generator

In order to interface with specific sensor, the parameters of TG are necessary to be setup. There are several parameters that are necessary to be setup: TG\_CR, TG\_LSTART, TG\_START, TG\_END.

### 37.3.1 P\_TG\_CR (0x08000000): Time Generator Control

NAME	D7	D6	D5	D4	D3	D2	D1	D0
P_TG_CR	BSEN	YUVOUT	YUVIN	CLKINV	RGB565	VGAEN	MASEN	CSIEN
NAME	D15	D14	D13	D12	D11	D10	D9	D8
P_TG_CR	INTERLACE	FIELDSEL	YUV_TYPE	VRST_TYPE	VADD_TYPE	HRST_TYPE	FGET_TYPE	CCIR656
NAME	D23	D22	D21	D20	D19	D18	D17	D16
P_TG_CR		RGB1555	QVGA27	CSICLKINV	RESIZE	D_TYPE		
NAME	D23	D22	D29	D28	D27	D26	D25	D24
P_TG_CR			CUT_EN	HALF_VGA	BS_MODE	INV_UV	MP4OUT	BIGED

CSIEN: Sensor interface enable register

0: Disable

1: Enable

MASEN: SPG Master/Slave mode select

0: SPG Slave mode(if connected with OV7648/OV7649)

1: SPG master mode(if connected with S202/S201)

VGAEN: When SPG is slave mode (MASEN is 0)

0: QVGA(each line contains 320 pixels)

1: VGA(each line contains 640 pixels)

When SPG is master mode(MASEN is 1)

0: S202 is connected

1: S201 is connected

RGB565: The OV7648 has a RGB565 mode output, this register is used to control it is RGB888 or RGB565 mode.

0: RGB888 mode

1: RGB565 mode

This bit is valid only when MASEN is 0 and YUVIN is 0

CLKINV: Sensor Clock inverse select

0: Non-inverse input sensor clock

1: Inverse input sensor clock

YUVIN: YUV input mode

0: Input data is RGB

1: Input data is YUV

This bit is valid only when MASEN is 0(i.e. OV7648/OV7649 is connected)

YUVOUT: YUV output mode

0: Output data is RGBRGB

1: Output data is YUV422

BSEN: Blue Screen effect select

0: Disable Blue Screen Effect

	1: Enable Blue Screen Effect
CCIR656:	CCIR601/CCIR656 select
	0: CCIR601 Interface
	1: CCIR656 Interface
	This bit is valid only when MASEN is 0.
FGET_TYPE:	Field get type
	0: Referring to field at falling edge of VSYHC
	1: Referring to field at rising edge of VSYHC.
	It's set to 1 if connected with OV7648
HRST_TYPE:	Horizontal counter
	0: Reset at the falling edge of HSYNC
	1: Reset at the rising edge of HSYNC
VADD_TYPE:	Vertical counter
	0: Add at the falling edge of HSYNC
	1: Add at the rising edge of HSYNC
VRST_TYPE:	Vertical counter
	0: Reset at the falling edge of VSYNC
	1: Reset at the rising edge of VSYNC
YUV_TYPE:	Sensor data sequence
	0: UYVY(BGRB)
	1: YUYU(GBGR)
FIELDSEL:	The polarity of input FODD
	0: Reversing is not necessary
	1: Reverse the FODD signal
INTERLACE:	Non-Interlace/Interlace mode select
	0: Non-Interlace mode(Field is don't care)
	1: Interlace mode
D_TYPE:	The Parameter of data sequence control
RESIZE:	Re-Size frame buffer control
	0: No re-size is applied
	1: VGA => QVGA re-size will be applied when VGA sensor is connected
CSICKOINV:	Invert sensor output clock
QVGA27:	QVGA Input 27 MHz Mode, For OV7660's QVGA Mode
RGB1555:	RGB1555's mode, the bit 15 has two different functions.
BIGED:	Big-endian mode selection.
	0: Little-endian output
	1: Big-endian output

- MP4OUT:** MP4 Output format selection.
- 0: YUYV output
  - 1: 4Y4U4Y4V output.
- INV\_UV:** YUV or YcbCr Selection
- 0: YUV output
  - 1: YCbCr output
- BS\_MODE:** Blue Screen Mode
- 0: For SPG290, blue screen is determined by bit 15 in RGB1555 mode.  
Blue screen function is not work in other mode.
- HALF\_VGA:** Half VGA mode (640x240) mode
- 0: Normal VGA mode (640x480)
  - 1: Output frame buffer will have only 640x240:  
In interlace mode, the field input will be neglected. So the frame rate will be 60 when CSI clock is 27 MHz.  
In non-interlace mode, the even line will be neglected. So the frame rate will be 30 when CSI clock is 27 MHz.
- CUT\_EN:** Cut screen function enable bit
- 0: Normal operation, the output frame buffer size is 320x240 or 640x480
  - 1: Cut screen mode, only the selected output region will be written to the frame buffer.

### 37.3.2 P\_TG\_LSTART(0x08000004):

NAME	D29-D20	D19-D10	D9-D0
<b>P_TG_LSTART</b>	TG_VL1START	TG_VL0START	TG_HLSTART

**TG\_HLSTART:** The parameter of horizontal TG.

To get the actual starting pixel data at a line from sensor.

**TG\_VL0START:** The parameter of horizontal TG.

To get the actual starting line from sensor in field 0 (interlace mode) or in each frame (non-interlace mode)

**TG\_VL1START:** The parameter of horizontal TG.

To get the actual starting line from sensor in field 1 (interlace mode).

### 37.3.3 P\_TG\_START(0x08000008):

NAME	D24-D16	D15-D10	D9-D0
<b>P_TG_START</b>	TG_VSTART		TG_HSTART

**TG\_HSTART:** The horizontal start of a sensor window.

If it's 0, the left side contains 0 black column. If it's 20, the left side

contains 20 black columns where the black color is defined in TG\_BLACK.

TG\_VSTART: The vertical start of a sensor window.

If it's 0, the top side contains 0 black row. If it's 10, the top side contains 10 black rows where the black color is defined in TG\_BLACK.

#### 37.3.4 P\_TG\_END(0x0800000C):

NAME	D24-D16	D15-D10	D9-D0
P_TG_END	TG_VEND		TG_HEND

TG\_HEND: The horizontal end of a sensor window.

For example, in VGA mode. If it's 640, the right side contains 0 black column. If it's 620, the right side contains 20 black lines where the black color is defined in TG\_BLACK.

TG\_VEND: The vertical start of a sensor window.

If it's 0, the bottom side contains 0 black row. If it's 10, the bottom side contains 10 black rows where the black color is defined in TG\_BLACK.

#### 37.3.5 P\_TG\_BLACK(0x0800000C):

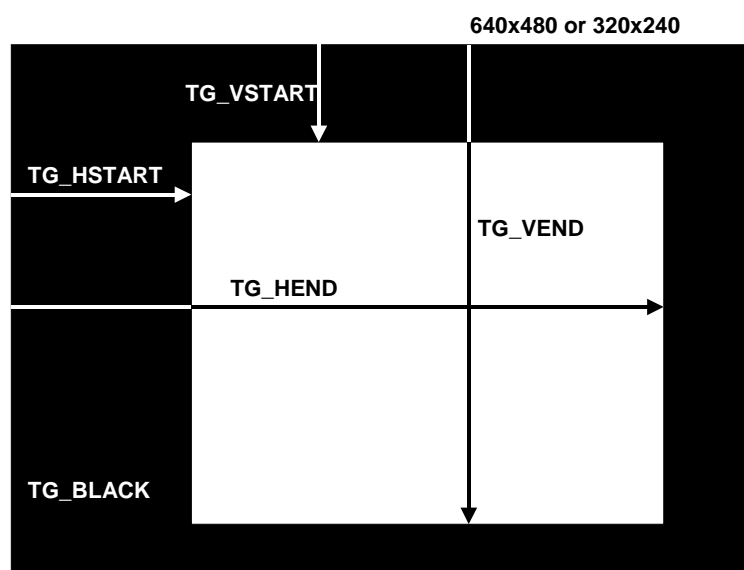
NAME	D23-D0
P_TG_BLACK	TG_BLACK

TG\_BLACK: The Black color definition.

When YUVOUT is 0, this is RGB888.

When YUVOUT is 1, this is YUV888

The following diagram shows the definition of these registers.



### 37.3.6 P\_TG\_BSUPPER(0x08000014):

NAME	D23-D16	D15-D8	D7-D0
P_TG_BSUPPER	TG_BSUPPER_B	TG_BSUPPER_G	TG_BSUPPER_R

TG\_BSUPPER\_R: The upper boundary of Red Color for Blue screen detection.

TG\_BSUPPER\_G: The upper boundary of Green Color for Blue screen detection.

TG\_BSUPPER\_B: The upper boundary of Blue Color for Blue screen detection.

### 37.3.7 P\_TG\_BSLOWER(0x08000018):

NAME	D23-D16	D15-D8	D7-D0
P_TG_BSLOWER	TG_BSLOWER_B	TG_BSLOWER_G	TG_BSLOWER_R

TG\_BSLOWER\_R: The lower boundary of Red Color for Blue screen detection.

TG\_BSLOWER\_G: The lower boundary of Green Color for Blue screen detection.

TG\_BSLOWER\_B: The lower boundary of Blue Color for Blue screen detection.

The blue screen region is defined in the following equation.

$(TG\_BSLOWER\_R < R/(R+G+B) < TG\_BSUPPER\_R)$  AND

$(TG\_BSLOWER\_G < G/(R+G+B) < TG\_BSUPPER\_G)$  AND

$(TG\_BSLOWER\_B < B/(R+G+B) < TG\_BSUPPER\_B)$  AND BSEN

### 37.3.8 P\_TG\_TRANSP(0x0800001C):

NAME	D23-D0
P_TG_TRANSP	TG_TRANSP

**TG\_TRANSP:** The transparent color definition.

When YUVOUT is 0, this is RGB888.

When YUVOUT is 1, this is YUV888.

This register will be used as the output data of sensor when a pixel is defined as the blue screen region.

### 37.3.9 P\_TG\_FBSADDR1(0x08000020): Frame Buffer 1's start address

NAME	D31-D0
P_TG_FBSADDR1	TG_FBSADDR1

### 37.3.10 P\_TG\_FBSADDR2(0x08000024): Frame Buffer 2's start address

NAME	D31-D0
P_TG_FBSADDR2	TG_FBSADDR2

### 37.3.11 P\_TG\_FBSADDR3(0x08000028): Frame Buffer 3's start address

NAME	D31-D0
P_TG_FBSADDR3	TG_FBSADDR3

**PS:** The frame buffer control is outside of sensor interface, the external hardware will determine which frame buffer is going to be written.

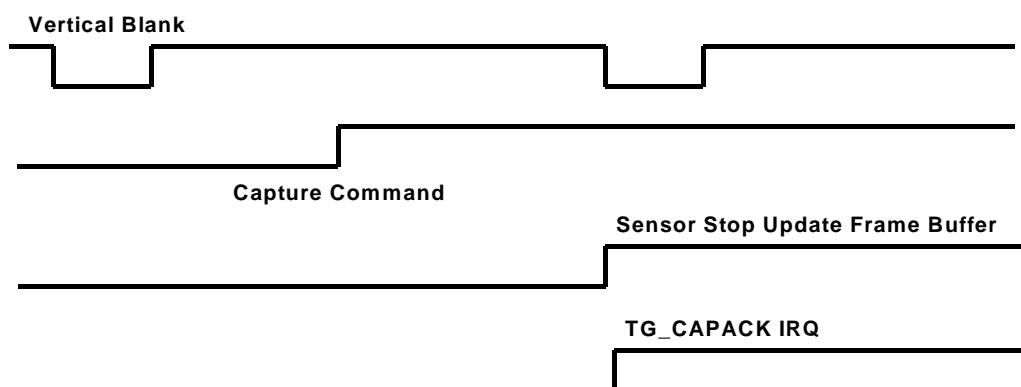
### 37.3.12 P\_TG\_CAP(0x0800002C): Frame Buffer 3's start address

NAME	D0
P_TG_CAP	TG_CAP

TG\_CAP: Capture control bit.

- 0: the sensor will return to normal function.
- 1: the sensor will stop update frame buffer after the current frame is updated completely. An interrupt will be asserted after the sensor is stop.

The following figure shows the capture flow.



### 37.3.13 P\_TG\_CUTSETUP(0x08000060):

NAME	D29-D24	D23-D21	D20-D16	D15-D14	D13-D8	D7-D5	D4-D0
P_TG_CUTSETUP	TG_HCURTSTART		TG_VCURTSTART		TG_HSIZE		TG_VSIZE

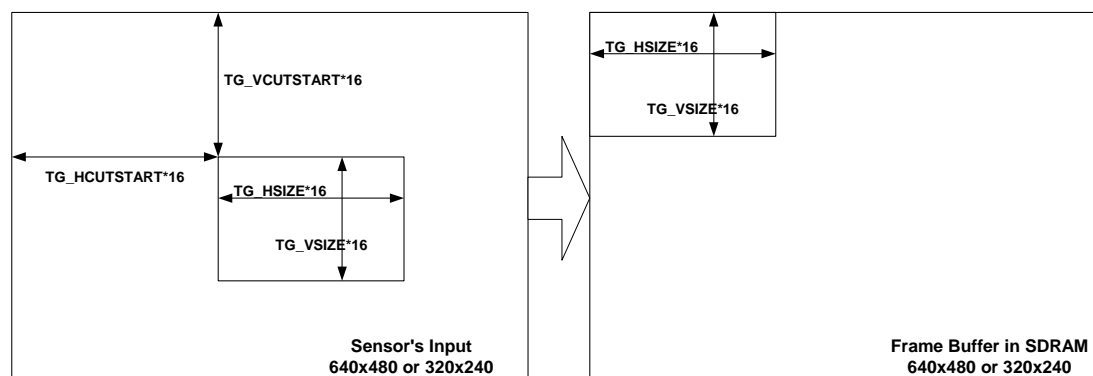
TG\_VSIZE: The Vertical Cut region size in unit of 16 points, only 1~30 is allowed.

TG\_HSIZE: The Horizontal Cut region size in unit of 16 points, only 1~40 is allowed.

TG\_VCURTSTART: The vertical cut region start address in unit of 16 points, only 1~30 is allowed.

TG\_HCURTSTART: The Horizontal cut region start address in unit of 16 points, only 1~40 is allowed.

The following diagram shows how the cut mechanism works.



You can change the position of the cut region in SDRAM by change the TG\_FBSADDR0~TG\_FBSADDR2.

### 37.4 Motion Detect Control

#### 37.4.1 P\_MD\_CR(0x88000030): Motion Detect control register.

NAME	D15-D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
P_MD_CR	DIFFY_THRES	SAMP_TYPE	CC_WHITE	CC_BLACK	FRAME_TYPE	MD_TYPE				

MD\_TYPE: Motion detect type selection register.

00: Motion detect function disable.

01: Reserved

10: SPG220's Y difference detect function.

11: SPG220's Y difference detect function and color classification function.

FRAME\_TYPE: Control the sampling frequency of motion detect function

00: Sample in each frame.

01: Sample in every two frame.

10: Sample in every four frame.

11: Sample in every eight frame.

CC\_BLACK: Black color definition of color classification circuit.

CC\_WHITE: White color definition of color classification circuit.

SAMP\_TYPE: Motion Detect size select

0 : 16x16

1 : 8x8, when MD\_TYPE is 0x11 and VGA mode is selected, this mode is not allowed.

DIFFY\_THRES: The recognition function is:

$$\text{DIFF} = |Y_{\text{new}} - Y_{\text{old}}|/2 \geq \text{DIFFY\_THRES}$$

Notes: DIFF bit is stored in address defined MD\_SADDR, for each word in memory:

{Y3[7:1], Y2[7:1], Y1[7:1], Y0[7:1], DIFF3, DIFF2, DIFF1, DIFF0}.

Sample 2 is at the right side of sample 1.

#### 37.4.2 P\_MD\_SADDR(0x88000034):

NAME	D31-D0
P_MD_SADDR	MD_SADDR

MD\_SADDR: Recognition start address.

##### In QVGA mode:

If recognition sample type 16x16 is used, the actual address mapping is:

$$\text{MD\_SADDR} \sim \text{MD\_SADDR} + 299$$

If recognition sample type 8x8 is used, the actual address mapping is:

$$\text{MD\_SADDR} \sim \text{MD\_SADDR} + 1199$$

##### In VGA mode:

If recognition sample type 16x16 is used, the actual address mapping is:

$$\text{MD\_SADDR} \sim \text{MD\_SADDR} + 1199$$

If recognition sample type 8x8 is used, the actual address mapping is:

$$\text{MD\_SADDR} \sim \text{MD\_SADDR} + 4799$$

#### 37.4.3 P\_MD\_POS(0x88000038):

NAME	D24-D16	D15-D10	D9-D0
P_MD_POS	MD_VPOS		MD_HPOS

MD\_HPOS: Horizontal IRQ position setting register

MD\_VPOS: Vertical IRQ position setting register

**NOTE:** When the sensor controller detect the current position is equal to {MD\_HPOS, MD\_VPOS}, an interrupt flag will be asserted and the color of this position will be stored in MD\_RGB register.

#### 37.4.4 P\_MD\_SADDR1(0x8800003C):

NAME	D31-D0
P_MD_SADDR1	MD_SADDR1

MD\_SADDR1: Color Classification start address.

##### In QVGA mode:

If recognition sample type 16x16 is used, the actual address mapping is:

$$\text{MD\_SADDR} \sim \text{MD\_SADDR} + 75$$

If recognition sample type 8x8 is used, the actual address mapping is:

$$\text{MD\_SADDR} \sim \text{MD\_SADDR} + 300$$

##### In VGA mode:

If recognition sample type 16x16 is used, the actual address mapping is:

MD\_SADDR ~ MD\_SADDR + 300

#### 37.4.5 P\_MD\_CTABLE0(0x88000040):

NAME	D31-D0
P_MD_CTABLE0	MD_CTABLE0

MD\_CTABLE0: Color Classification table 0, color of index 0 ~ 15

#### 37.4.6 P\_MD\_CTABLE1(0x88000044):

NAME	D31-D0
P_MD_CTABLE1	MD_CTABLE1

MD\_CTABLE1: Color Classification table 1, color of index 16 ~ 31

#### 37.4.7 P\_MD\_CTABLE2(0x88000048):

NAME	D31-D0
P_MD_CTABLE2	MD_CTABLE2

MD\_CTABLE2: Color Classification table 2, color of index 32 ~ 47

#### 37.4.8 P\_MD\_CTABLE3(0x8800004C):

NAME	D31-D0
P_MD_CTABLE3	MD_CTABLE3

MD\_CTABLE3: Color Classification table 3, color of index 48 ~ 54

#### 37.4.9 P\_MD\_REG1(0x88000050):

NAME	D31-D24	D23-D16	D15-D8	D7-D0
P_MD_REG1	L1	R1	T1	B1

B1: Bottom boundary of region 1, -128~128.

T1: Top boundary of region 1, -128~128.

R1: Right boundary of region 1, -128~128.

T1: Left boundary of region 1, -128~128.

#### 37.4.10 P\_MD\_REG2(0x88000054):

NAME	D31-D24	D23-D16	D15-D8	D7-D0
P_MD_REG2	L2	R2	T2	B2

B2: Bottom boundary of region 2, -128~128.

T2: Top boundary of region 2, -128~128.

R2: Right boundary of region 2, -128~128.

T2: Left boundary of region 2, -128~128.

#### 37.4.11 P\_MD\_REG3(0x88000058):

NAME	D31-D24	D23-D16	D15-D8	D7-D0
P_MD_REG3	L2	R3	T3	B3

B3: Bottom boundary of region 3, -128~128.

T3: Top boundary of region 3, -128~128.

R3: Right boundary of region 3, -128~128.

T3: Left boundary of region 3, -128~128.

#### 37.4.12 P\_MD\_TH(0x8800005C):

NAME	D31-D24	D23-D16	D15-D8	D7-D0
P_MD_TH	TH_B	TH_W	TH1	TH2

TH2: Threshold of region 2, 0~191.

TH1: Threshold of region 1, 0~191.

TH\_W: Threshold of white color, 0~191.

TH\_B: Threshold of black color, 0~191.

#### 37.4.13 P\_MD\_RGB(0x88000074):

NAME	D23-D0
P_MD_RGB	MD_RGB

MD\_RGB: The capture color value.

When YUVOUT is 1, the value in this register is YUV888.

When YUVOUT is 0, the value in this register is RGB888.

### 37.5 Interrupt

There are two registers to control the CSI related interrupts. One is P\_CSI\_IRQEN, which is a mask register for the CSI interrupt. When this register is set to '1', the interrupt signal will be masked, that is the signal will remain inside of CSI module and the CPU can not receive this signal, therefore it will not enter into interrupt service routine.

The other register is P\_CSI\_IRQSTS, which is a interrupt status register. When interrupt event is received, the CPU will query this status register to determine which interrupt event is happen, and enter into the corresponding interrupt service routine. The CPU will clear the interrupt event flag automatically when it read the status register.

#### 37.5.1 P\_CSI\_IRQEN(0x88000078)

NAME	D6	D5	D4	D3	D2	D1	D0
P_CSI_IRQEN	MD_UF_IRQ	FRAME_DISABLE_IRQ	POS_HIT_IRQ	MD_FRAME_IRQ	FRAME_END_IRQ	TG_CAPACK_IRQ	TG_OF_IRQ

TG\_OF\_IRQ: TG\_OF IRQ enable register.

0: Disable TG\_OF IRQ.

1: Enable TG\_OF IRQ.

TG\_CAPACK\_IRQ: TG\_CAPACK IRQ enable register.

0: Disable TG\_CAPACK IRQ.

1: Enable TG\_CAPACK IRQ.

FRAME\_END\_IRQ: FRAME\_END IRQ enable register.

0: Disable FRAME\_END IRQ.

1: Enable FRAME\_END IRQ.

MD\_FRAME\_IRQ: MD\_FRAME IRQ enable register.

0: Disable MD\_FRAME IRQ.

1: Enable MD\_FRAME IRQ.

POS\_HIT\_IRQ: POS\_HIT IRQ enable register.

0: Disable POS\_HIT IRQ.

1: Enable POS\_HIT IRQ.

FRAME\_DIS\_IRQ: FRAMD\_DIS IRQ enable register.

0: Disable FRAMD\_DIS IRQ.

1: Enable FRAMD\_DIS IRQ.

MD\_UF\_IRQ: MD\_UF IRQ enable register.

0: Disable MD\_UF IRQ.

1: Enable MD\_UF IRQ.

### 37.5.2 P\_CSI\_IRQSTS(0x8800007C)

NAME	D6	D5	D4	D3	D2	D1	D0
P_CSI_IRQSTS	MD_UF_STS	FRAME_DIS_STS	POS_HIT_STS	MD_FRAME_STS	FRAME_END_STS	TG_CAPACK_STS	TG_OVERFLOW_STS

TG\_OVERFLOW\_STS: TG\_OVERFLOW IRQ status register.

Read 0: TG\_OVERFLOW IRQ not happen.

Read 1: The sensor output data overflow situation is happened.

Write 0: No effect.

Write 1: Clear this bit.

TG\_CAPACK\_IRQ: TG\_CAPACK IRQ status register.

Read 0: TG\_CAPACK IRQ not happen.

Read 1: The capture procedure is done.

Write 0: No effect.

Write 1: Clear this bit.

FRAME\_END\_IRQ: FRAME\_END IRQ status register.

Read 0: FRAME\_END IRQ not happen.

Read 1: A frame is end.

Write 0: No effect.

Write 1: Clear this bit.

MD\_FRAME\_IRQ: MD\_FRAME IRQ status register.

Read 0: MD\_FRAME IRQ not happen.

Read 1: A motion detect frame is end.  
Write 0: No effect.  
Write 1: Clear this bit.

POS\_HIT\_IRQ: POS\_HIT IRQ status register.  
Read 0: POS\_HIT IRQ not happen.  
Read 1: The luster is hit the position defined in {MD\_HPOS, MD\_VPOS}  
Write 0: No effect.  
Write 1: Clear this bit.

FRAME\_DIS\_IRQ: Frame disable interrupt.  
Read 0: FRAME\_DIS IRQ not happen.  
Read 1: A frame disable flag is received by CSI module.  
Write 0: No effect.  
Write 1: Clear this bit.

MD\_UF\_IRQ: Motion Detect Under Flow interrupt.  
Read 0: MD\_UF IRQ not happen.  
Read 1: A motion detect buffer under-run condition is happened.  
Write 0: No effect.  
Write 1: Clear this bit.

## 37.6 Color Mode

### 37.6.1 P\_CSI\_Y2R\_A1(0x880000E8)

NAME	D29-D20	D19-D10	D9-D0
P_CSI_Y2R_A1	Y2R_A11	Y2R_A12	Y2R_A13

Y2R\_A13: A13 parameter of YUV→RGB transfer

Y2R\_A12: A12 parameter of YUV→RGB transfer

Y2R\_A11: A11 parameter of YUV→RGB transfer

### 37.6.2 P\_CSI\_Y2R\_A2(0x880000EC)

NAME	D29-D20	D19-D10	D9-D0
P_CSI_Y2R_A2	Y2R_A21	Y2R_A22	Y2R_A23

Y2R\_A23: A23 parameter of YUV→RGB transfer

Y2R\_A22: A22 parameter of YUV→RGB transfer

Y2R\_A21: A21 parameter of YUV→RGB transfer

### 37.6.3 P\_CSI\_Y2R\_A3(0x880000F0)

NAME	D29-D20	D19-D10	D9-D0
------	---------	---------	-------

P_CSI_Y2R_A3	Y2R_A31	Y2R_A32	Y2R_A33
--------------	---------	---------	---------

Y2R\_A33: A33 parameter of YUV→RGB transfer

Y2R\_A32: A32 parameter of YUV→RGB transfer

Y2R\_A31: A31 parameter of YUV→RGB transfer

**NOTE: The parameter of YUV⇒ RGB transfer is all 10 bits signed integer. It will be divided by 256 in the hardware. In other word, the value of these parameters have the following range.**

$$-512/256 \leq Y2R\_Axx \leq 511/256$$

#### 37.6.4 P\_CSI\_R2Y\_A1(0x880000E8)

NAME	D29-D20	D19-D10	D9-D0
P_CSI_R2Y_A1	R2Y_A11	R2Y_A12	R2Y_A13

R2Y\_A13: A13 parameter of RGB→YUV transfer

R2Y\_A12: A12 parameter of RGB→YUV transfer

R2Y\_A11: A11 parameter of RGB→YUV transfer

#### 37.6.5 P\_CSI\_R2Y\_A2(0x880000EC)

NAME	D29-D20	D19-D10	D9-D0
P_CSI_R2Y_A2	R2Y_A21	R2Y_A22	R2Y_A23

R2Y\_A23: A23 parameter of RGB→YUV transfer

R2Y\_A22: A22 parameter of RGB→YUV transfer

R2Y\_A21: A21 parameter of RGB→YUV transfer

#### 37.6.6 P\_CSI\_R2Y\_A3(0x880000F0)

NAME	D29-D20	D19-D10	D9-D0
P_CSI_R2Y_A3	R2Y_A31	R2Y_A32	R2Y_A33

R2Y\_A33: A33 parameter of RGB→YUV transfer

R2Y\_A32: A32 parameter of RGB→YUV transfer

R2Y\_A31: A31 parameter of RGB→YUV transfer

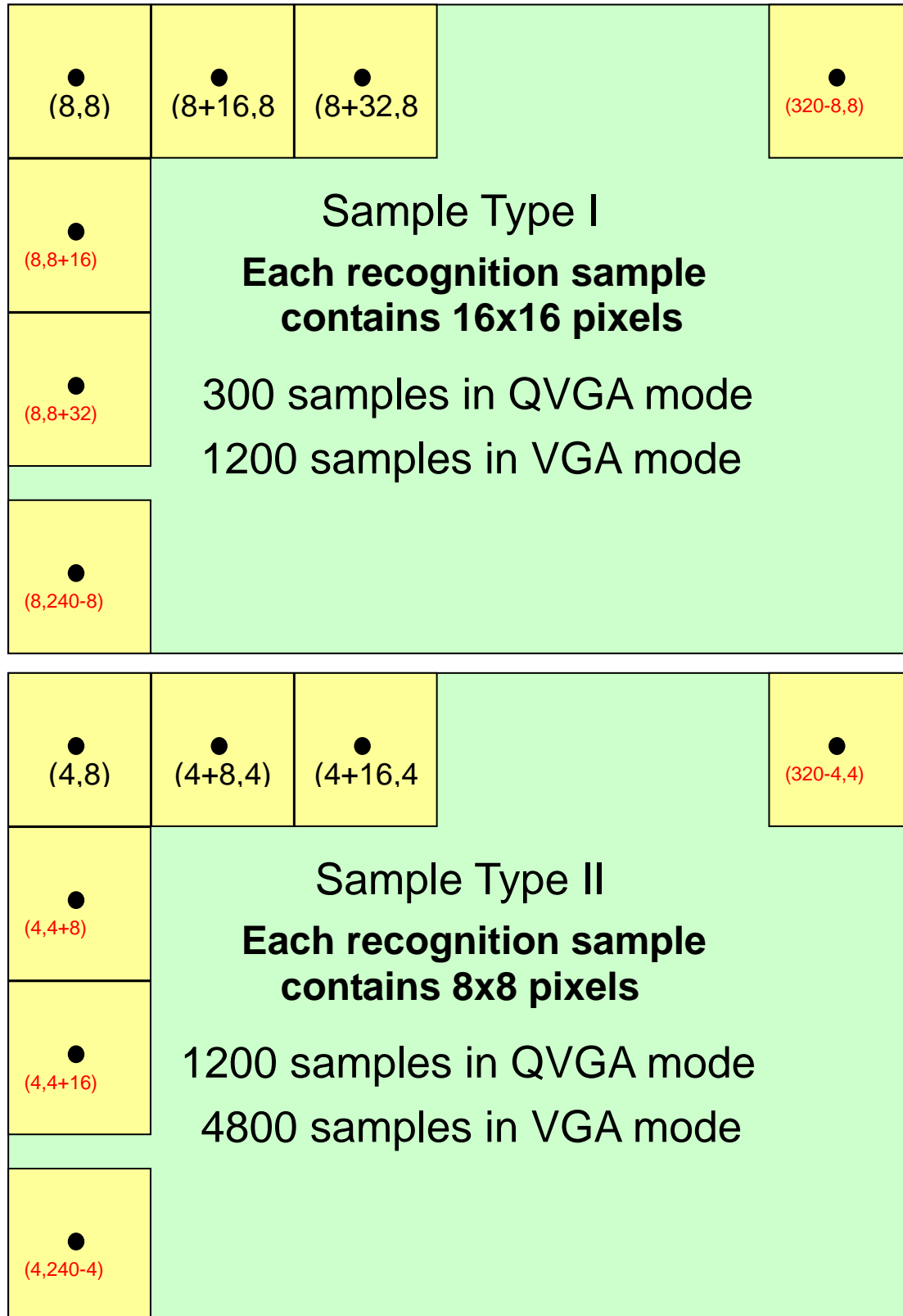
**NOTE: The parameter of RGB⇒ YUV transfer is all 9 bits signed integer. It will be divided by 256 in the hardware. In other word, the values of these parameters have the following range.**

$$-256/256 \leq R2Y\_Axx \leq 255/256$$

**CSI\_Y2R and CSI\_R2Y share the same register.** When YUVIN mode is selected, these registers must be filled by YUV⇒RGB parameter. When RGBIN mode is selected, these registers must be filled by RGB⇒YUV parameter.

### 37.7 Motion detection

#### 37.7.1 Recognition Sample Type



### 37.7.2 Recognition Reference Filed / Frame

If CPU does not have much time to handle motion detect algorithm, recognition data can not always calculated at each frame. The following table shows the reference field or frame decided by setting up parameters. For examples, if INTERLACE = 1 and RC\_FIELD = 0 and FRAME\_TYPE = 2, each reference frame is frame2 field0 instead of frame2 field1.

INTERLACE	RC_FIELD	FRAME_TYPE	Field0	Field1	Frame0	Frame1	Frame2	Frame3
0	0/1	0			HIT	HIT	HIT	HIT
0	0/1	1				HIT		HIT
0	0/1	2					HIT	
0	0/1	3						HIT
1	0	0	REF		HIT	HIT	HIT	HIT
1	0	1	REF			HIT		HIT
1	0	2	REF				HIT	
1	0	3	REF					HIT
1	1	0		REF	HIT	HIT	HIT	HIT
1	1	1		REF		HIT		HIT
1	1	2		REF			HIT	
1	1	3		REF				HIT

**Notes:** REF means which field is referred.

FRAME_TYPE	FRAME_COUNTER
<b>0</b>	Always 0
<b>1</b>	Always from 0 to 1
<b>2</b>	Always from 0 to 2
<b>3</b>	Always from 0 to 3

## 38 MPEG-4/JPEG CODEC

MPEG4: Moving Picture Experts Group 4

JPEG: Joint Photographic Experts Group

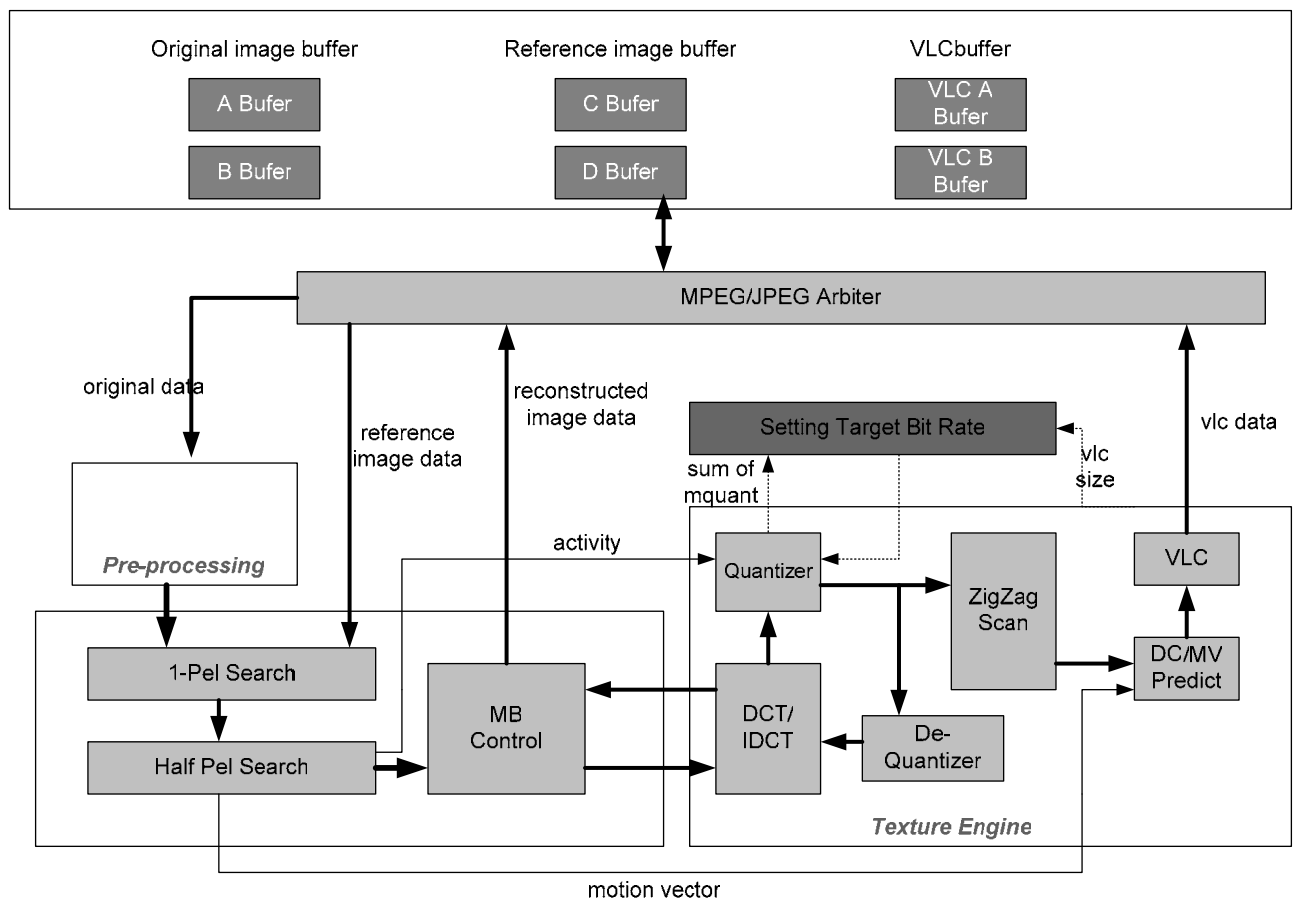
The hardware-implemented MPEG-4 codec supports the simple profile for decoding and encoding. The image resolution can be QVGA or VGA. When the codec is running at 54MHz, it can achieve 30f/s frame rate for encoding or decoding with QVGA resolution. The quantization step size is programmable to get the high compression ratio or to obtain the good image quality.

The same codec also provides JPEG function. It supports 4:2:2 and 4:2:0 data format for decoding and encoding.

The image resolution can be QVGA or VGA.

### 38.1 Functional Description

#### 38.1.1 MPEG Encoding



When you encode a series pictures to a video, you must setting and control the original image buffer, reference image buffer and VLC buffer memory start address. A series video image start address must be set in the registers **MP4RAW\_START\_ADRX** (reference Chapter about the MIU), and you can switch these image buffer by setting the register **MP4RAW\_BUF\_PTR\_REG**(reference Chapter about the BUFCTL). Here suggest that the image buffer start

address had better be 1K aligned. MPEG engine registers “**Mjhoffset**” and “**Mjvoffset**” let you adjust the start point of video image if you need.

The MPEG engine needs reference image buffers when it operates MPEG encoding. Giving two image buffer start addresses to registers **MP4W\_START\_ADR1** and **MP4W\_START\_ADR2** (reference Chapter 6 MIU) and the buffer size and the video image are the same. Of course, the start addresses of reference image buffers are 1K aligned. The control register which switches these start address is **MP4W\_BUF\_PTR\_REG** and **MP4R\_BUF\_PTR\_REG** (reference Chapter 11 BUFCTL) and their setting can't be the same. After completing a frame encoding, you must exchange their value each other.

You can assign where the encoded video code is stored by writing the buffer start address in register **MP4V\_START\_ADRX** (reference Chapter 6 MIU) and using the register **MP4V\_BUF\_PTR\_REG** (reference Chapter 11 BUFCTL) to switch these buffer start addresses.

You also can set the MPEG engine registers “**eVlcOffaddr**” which lets MPEG engine bypass the MPEG file header you create which store some information about this MPEG4 video information.

The MPEG engine will output the MPEG4 video object plane part and you need to add the MPEG4 visual object and video object layer before the codes which MPEG engine generates to complete a MPEG4 video file. Then you can make it up to a 3gp or avi file and save them to other media if you understand these files formats.

The next step is setting the video image size. Beside the MPEG engine registers “**mjwidth**” and “**mjheight**”, you need set the “**MP4\_FRAME\_BUF\_HSIZE**” (reference Chapter 6 MIU), too.

You must know this MPEG engine doesn't support the B-frame. About the limitation of the MPEG engine, you can reference 19.3.

The MPEG engine registers “**Gopval**” lets MPEG engine encodes one I-frame after the static number P-frames automatically. If you want to control when to encode I-frame or P-frame, you can use the register “**iframe**” to decide the frame type. You can switch the two choices by setting the register “**gopsetting**”. No matter what mode you operation, you must set nonzero value to the MPEG engine registers “**Gopval**”. When you generate I-frame or P-frame by the register “**iframe**”, you had better setting 2 to “**Gopval**”.

The recommend value to the register “**VOPTimeIncLen**” is 14.

Because the MPEG specific doesn't uses quantize table, you had better enable the register “**QscaleEn**”.

Then setting these registers “**IframeQscale**” and “**PframeQscale**”, the MPEG engine will generate Quantizer for video frame automatically. The greater Qscale will make little compressed code and poor video quality.

The other registers like as “**unrestricted**”, “**halfpelen**” and “**SkipType**” will affect the MPEG engine performance and you can get more information about MPEG4 and H.263 from ISO/IEC 14496-2.

After setting the MPEG related registers, you also need setting the MPEG engine register “**SRAM\_CS\_N**” =01 to turn on the internal static ram of MPEG texture engine.

Besides setting these registers **MP4W\_BUF\_PTR\_REG**, **MP4R\_BUF\_PTR\_REG**, **MP4RAW\_BUF\_PTR\_REG**

and **MP4V\_BUF\_PTR\_REG** by software to switch these buffers start addresses by which the MPEG engine operates, you also can let the hardware switch them automatically when one frame is encoded. Setting the register **PTR\_SETTING** (reference Chapter about the BUFCTL) to achieves that BUFCTL switch these buffers automatically. Be careful using the register or it may cause some error to system.

Besides setting automatically switch the MPEG engine's buffer, you also need to inform the MPEG engine then it will apply to this. Setting the register **SYS\_MPG\_CAMMODE=3** (see Chapter about the SFTCFG), “**doubleEn**” =1 and “**SOF mode**” =0 makes the MPEG engine start encoding the next frame after former frame is encoded automatically.

When the MPEG engine encodes MPEG4 video speed is slower than the picture source making, we must force the hardware to bypass some frame to be encoded when encoding frame is automatical, we can setting the register **FRAME\_NUM\_FOR\_MP4ENC** (reference Chapter about the BUFCTL).

You also can control all by software and it is more safely. The software gets the control by setting the register **SYS\_MPG\_CAMMODE=7** (see Chapter about the SFTCFG) and “**SOF mode**” =1 and clearing the register **PTR\_SETTING** (reference Chapter about the BUFCTL) and “**doubleEn**” .

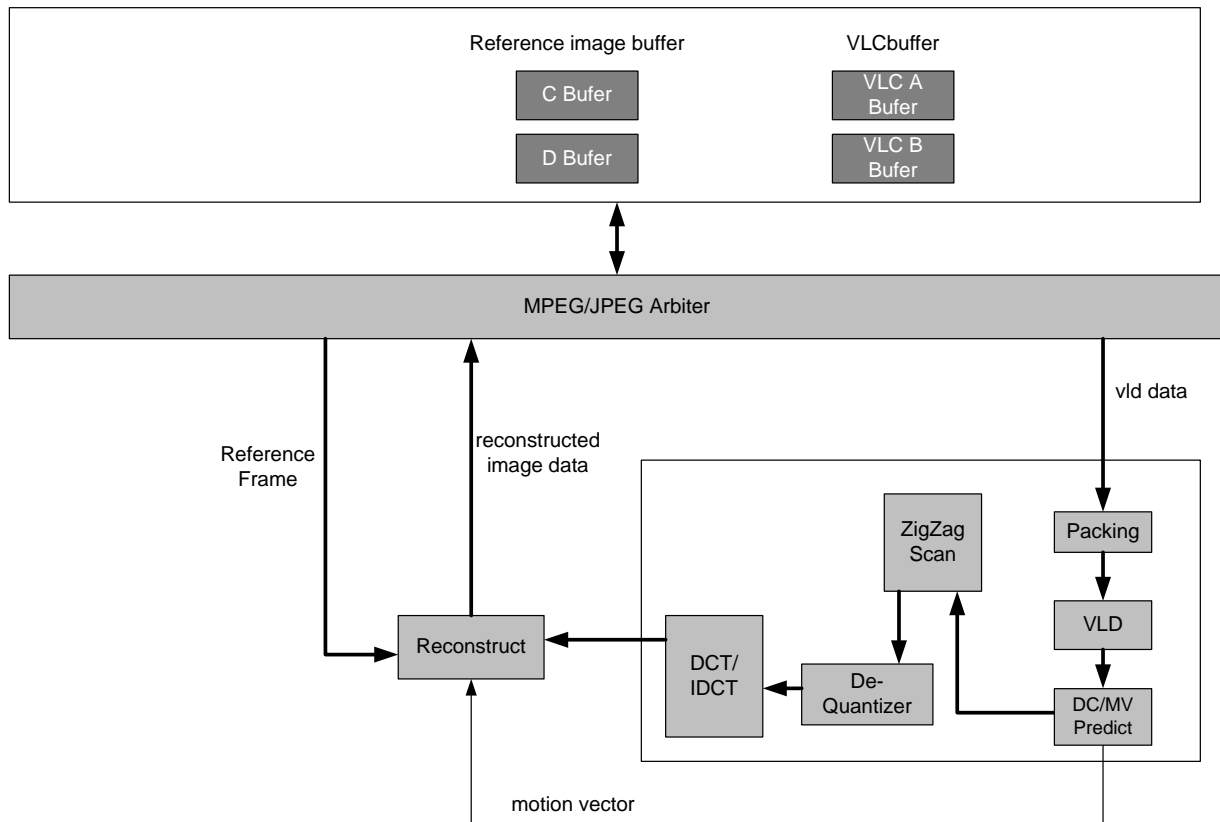
When you are sure that you have filled all registers for MPEG encoding function, the software can start the encoding process by setting the MPEG engine register “**mjpgcsof**” =1.

You can be acknowledged by polling the register “**encodedone**” or interrupt after the MPEG engine finishes its MPEG encoding work.

The MPEG engine interrupt is 33(see Chapter about the Interrupt Controller), and its interrupt mask is “**Eofen**”. If the MPEG interrupt occurs, you must clear it by writing the register “**Eofintclr**” .

The last step is reading register “**VLCSIZE**” for getting the MPEG code length.

### 38.1.2 MPEG Decoding



When you decode a series pictures from a MPEG video file, you just need to set and control the reference image buffer and VLC buffer memory. The video file format must meet the limit of the MPEG engine. You can check the items listed as fellow content (See 19.3).

The MPEG engine needs reference image buffers when it operates MPEG decoding. Giving two image buffer start addresses to registers **MP4W\_START\_ADR1** and **MP4W\_START\_ADR2** (reference Chapter 6 MIU) and the buffer sizes are the same sizes of the video images that you want to get. No doubt, the start addresses of reference image buffers are 1K aligned. The control register which switches these start address is

**MP4W\_BUF\_PTR\_REG** and **MP4R\_BUF\_PTR\_REG** (reference Chapter 11 BUFCTL) and their setting can't be the same. After completing a frame image decoding, you can get the video image from the start address (**MP4W\_START\_ADR1** or **MP4W\_START\_ADR2**) which **MP4W\_BUF\_PTR\_REG** points and you must put the value of

**MP4W\_BUF\_PTR\_REG** into the register **MP4R\_BUF\_PTR\_REG**. Then the newest decoded video image will be referenced by MPEG engine as it decodes a next P-frame encoded image.

You can assign where the video file is stored by writing the buffer start address in register **MP4V\_START\_ADRX** (reference Chapter 6 MIU) and using the register **MP4V\_BUF\_PTR\_REG** (reference Chapter 11 BUFCTL) to switch these buffer start addresses.

You also can set the MPEG engine register “ **dVlcOffaddr** ” which lets MPEG engine bypass the video file

header, and notice that the value set into register “ **dVlcOffaddr**” should not be too big.

The MPEG engine just can decodes MPEG4 video object plane part, so you must drop the code before MPEG4 video object plane in a MPEG4 video file no matter the file format is 3gp or avi.

The next step is setting the video image size that can be extracted from the video file header. Beside the MPEG engine registers “ **decmjwidth**” and “ **decmjheight**” , you need set the “ **MP4\_FRAME\_BUF\_HSIZE**” (reference Chapter 6 MIU), too.

You must enable the register “ **Uv420to422**” and “ **Uv420to422en**” , because SPG290 display interfaces (TVE or LCD) just support YUV422; otherwise the MPEG engine will output yuv420 image.

To tell the MPEG engine that we want to encode a video file, you need setting t MPEG4 decode mode in the MPEG engine register “**Optmode**”.

If you are decoding MPEG4 visual object plane, you had better set the visual object plane start code 0x000001b6 to the register “ **MatchCode**” .

If you are decoding H.263 video file, you had take care these registers “ **H263en**” , “ **H263format**” and “ **H263GOBen**” .

You can get more information about MPEG4 and H.263 in ISO/IEC 14496-2; the specific can give you more help than this document to understand MPEG4 and H.263.

Because the MPEG specific doesn't uses quantize table, you had better enable the register “**QscaleEn**”.

The value to the register “ **VOPTimeIncLen**” “ **FourMVen**” “ **AcpreDen**” may be need to be filled by user or the MPEG engine will get the information from MPEG4 visual object plane.

After setting the MPEG related registers, you also need setting the MPEG engine register “ **SRAM\_CS\_N**” =01 to turn on the internal static ram of MPEG texture engine.

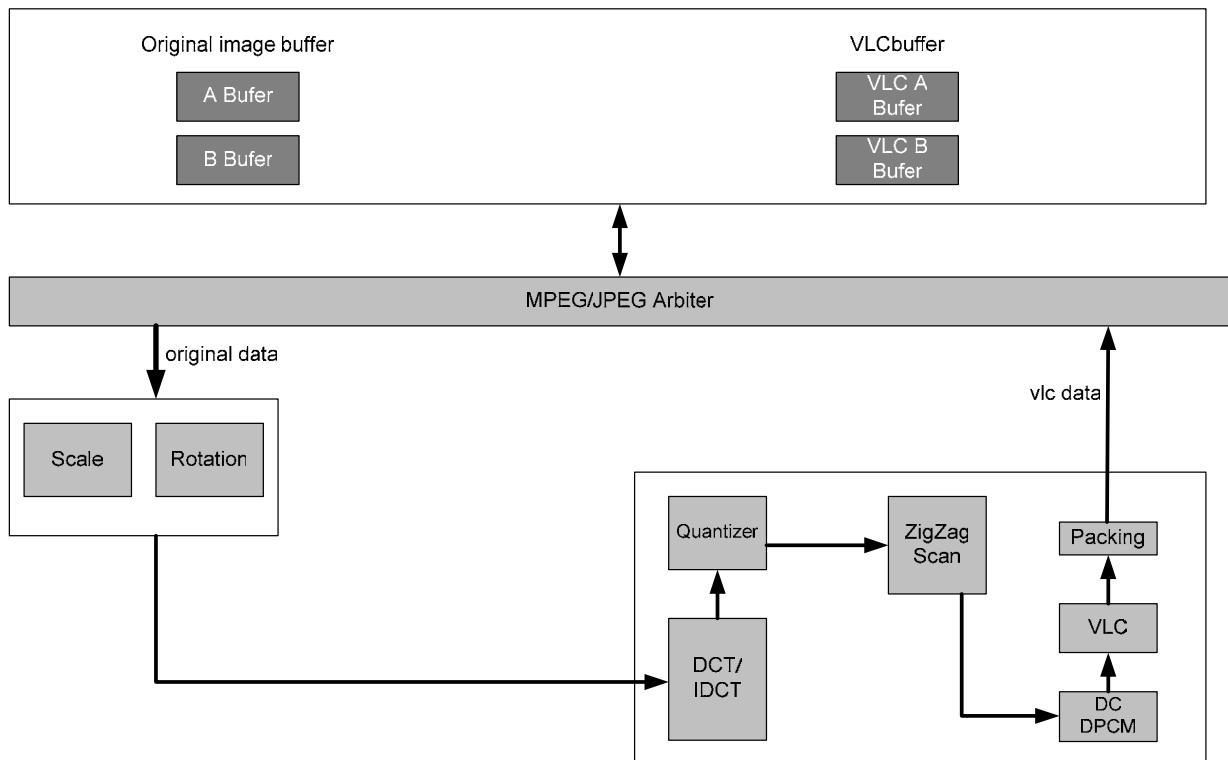
The software gets the control of video decoding by setting the register “**SOF mode**” =1.

When you need to start the MPEG engine decoding, you just setting the register “ **mjpgdsof**” .

You can be acknowledged by polling the register “ **decodedone**” or interrupt after the MPEG engine finishes its MPEG encoding work.

The MPEG engine interrupt is 33(see Chapter 20 Interrupt Controller), and its interrupt mask is “**Eofen**”. If the MPEG interrupt occurs, you must clear it by writing the register “ **Eofintclr**” .

### 38.1.3 JPEG Encoding



When you want to encode JPEG file, set the **MP4RAW\_START\_ADRX** (reference Chapter 6 MIU) firstly which tells the MPEG engine where the original picture stored. MPEG engine registers “**Mjhoffset**” and “**Mjvoffset**” let you adjust the start point of original picture if the image memory start address is not just at where **MP4RAW\_START\_ADRX** specified.

Then set **MP4V\_START\_ADRX** (reference Chapter 6 MIU) which tell the MPEG engine where you want to store the compressed JPG file. The start address should be word align. Be careful that MPEG engine will not fill the JPEG file header, you must generate it by yourself. You can set the MPEG engine registers “**eVlcOffaddr**” which lets MPEG engine bypass the JPEG file header you create.

The next step is setting the original picture size. Beside the MPEG engine registers “**mjwidth**” and “**mjheight**”, you need set the “**MP4\_FRAME\_BUF\_HSIZE**” (reference Chapter 6 MIU), too.

To tell the MPEG engine that we want to compress a picture, you need setting t JPG encode mode in the MPEG engine register “**Optmode**”. And tell the MPEG engine what sample type which your original picture is (YUV422, YUV420, gray-level ).

To enable the MPEG engine register “**JFIF**” for ensuring your compressed jpeg file can meet the JPEG File Interchange Format.

No matter what mode you operation, you must set nonzero value to the MPEG engine registers “**Gopval**”. If the MPEG engine is not in MPEG encode mode, you had better setting 2 to “**Gopval**”.

I don’ t suggest that enable the MPEG engine by hardware, unless you very understand the MIU and BUFCTL

operation. So I recommend setting the MPEG engine registers “ **SOF mode**” =1, “ **doubleEn**” =0.

Before you fill the Quantization Table by which you want to compress the original picture to JPEG format, you should turn on the SRAM which the MPEG engine save the Quantization Table by setting the MPEG engine register “ **SRAM CS\_N**” =01 and “ **QsramEn**” =1. The choice of JPEG file’ s Quantization Table will affect the JPEG’ s compression ratio and the quality. You must be careful about the order when filling Quantization Table. The order of Quantization Table saved in JPEG file header is zigzag, and the MPEG engine register “ **Qtable1**” (luminance quantization table) and “ **Qtable2**” (chrominance quantization table) are followed by Quantization Table matrix order.

For saving power consumption, I suggest turn off the MPEG move estimation sram of the MPEG engine power by setting “ **Mesramdis**” =1 when encoding or decoding JPEG file.

The JPEG encoding of the MPEG engine is using the static Huffman Table, so you don’ t need fill the Huffman Table.

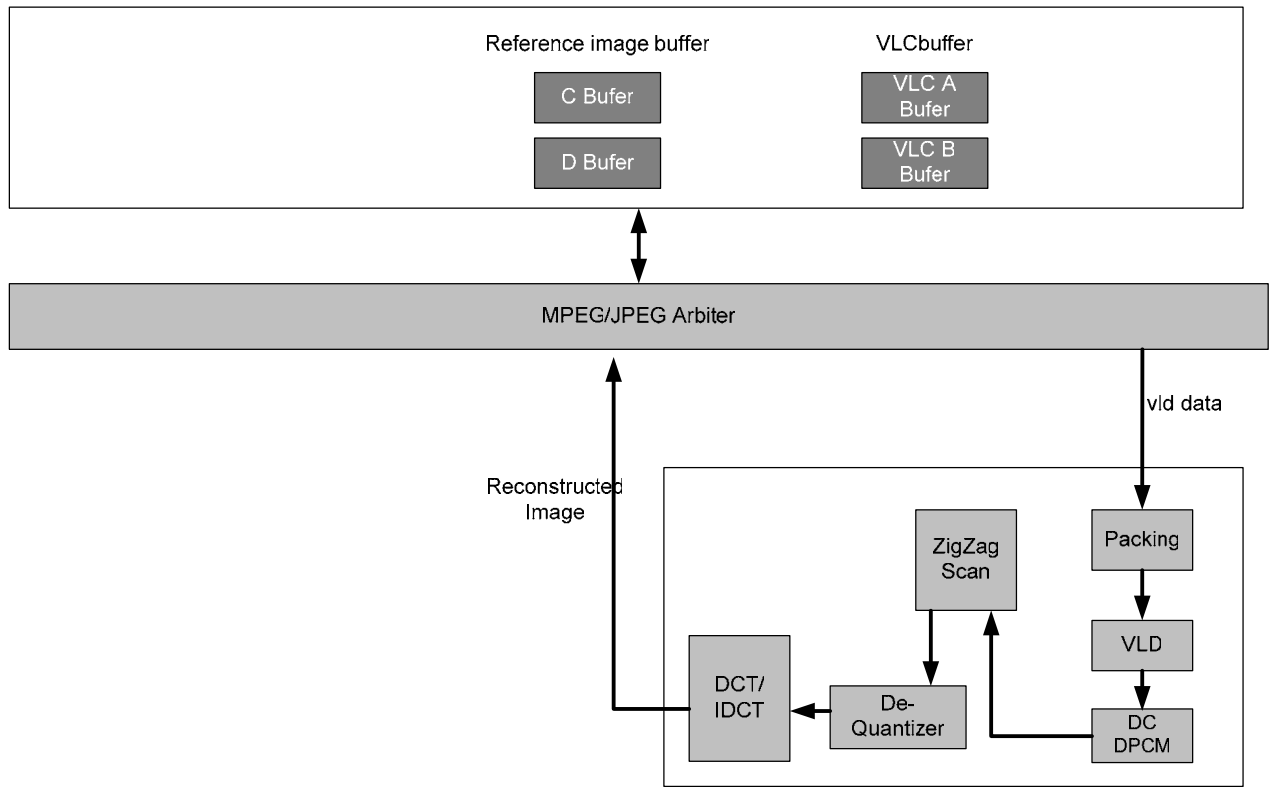
When you sure that you have filled all registers for JPEG encoding function, you can start the encoding process by setting the MPEG engine register “ **mjpgcsof**” =1.

You can be acknowledged by polling the register “ **encodedone**” or interrupt after the MPEG engine finishes its JPEG encoding work.

The MPEG engine interrupt is 33(see Chapter 20 Interrupt Controller), and its interrupt mask is “**Eofen**”. If the MPEG interrupt occurs, you must clear it by writing the register “ **Eofintclr**” .

The last step is reading register “**VLCSize**” for getting the JPEG file length.

### 38.1.4 JPEG Decoding



If you want to decode JPEG file, you should decode JPEG file header for getting some useful information filled the MPEG engine registers by yourself firstly. You can getting JPEG file's Quantization Table, Huffman Table, image size, YUV sample type ... etc from JPEG file header. Before you fill the Quantization Table by which gotten from JPEG file header, you should turn on the SRAM which the MPEG engine save the Quantization Table by setting the MPEG engine register " **SRAM CS\_N**" =01 and " **QsramEn**" =1.

You must be careful about the order when filling Quantization Table. The order of Quantization Table saved in JPEG file header is zigzag, and the MPEG engine register " **Qtable1**" (luminance quantization table) and " **Qtable2**" (chrominance quantization table) are followed by Quantization Table matrix order. After you filled Quantization Table, you should turn off the register " **QsramEn**" .

For saving power consumption, I suggest turn off the MPEG move estimation static ram of the MPEG engine power by setting " **Mesramdis**" =1 when decoding or decoding JPEG file.

The Huffman Tables stored in JPEG file header are not filled into the MPEG engine registers directly. You must find the code values, code sizes and code words of Huffman tables to fill these registers "YDCW", "YDCS", "YDCV"(Luminance DC Huffman Table), "CDCW", "CDCS", "CDCV" (Chrominance DC Huffman Table), "YACW", "YACS" (Luminance AC Huffman Table), "CACW", "CACS" (Chrominance AC Huffman Table)and Huffman Table static ram. You can reference the CCITT Recommendation T.81 Annex K to understand how to do this.

Before you writing the MPEG engine static ram in which stored luminance and chrominance ac Huffman table

code values, you should turn on the CPU access the static ram channel “**TsramMode**” =01. You can use the register “**TsramPage**” to switch the static ram pages and 64 bytes per page, the first four pages for luminance ac Huffman table code values and the last four pages for chrominance.

Set the **MP4V\_START\_ADRX** (reference Chapter 6 MIU) firstly which tells the MPEG engine where the JPEG file stored.

MPEG engine register “**dVlcOffaddr**” let you adjust the start point of JPEG file if you need to bypass the JPEG file header.

Then set **MP4V\_START\_ADRX** (reference Chapter 6 MIU) which tell the MPEG engine where you want to store decompressed image.

The next step is setting the original picture size that is the part of JPEG file header. Beside the MPEG engine registers “**decmjwidth**” and “**decmjheight**”, you need set the “**MP4\_FRAME\_BUF\_HSIZE**” (reference Chapter 6 MIU), too.

To tell the MPEG engine that we want to decompress a JPEG file, you need setting t JPG decode mode in the MPEG engine register “**Optmode**”. And tell the MPEG engine what sample type which your original picture is (YUV422, YUV420, gray-level). Please remember set the register “**Uv420to422**” and “**Uv420to422en**”.

To enable the MPEG engine register “**JFIF**” if the JPEG file meets the JPEG File Interchange Format.

No matter what mode you operation, you must set nonzero value to the MPEG engine registers “**Gopval**”. If the MPEG engine is not in MPEG encode mode, you had better setting 2 to “**Gopval**”.

I don’ t suggest that start the MPEG engine operation by hardware, unless you very understand the operation between MPEG engine, MIU and BUFCTL. So I recommend setting the MPEG engine registers “**SOF mode**” =1, “**doubleEn**” =0.

When you sure that you have filled all registers for JPEG decoding function, you can start the encoding process by setting the MPEG engine register “**mjpgdsof**” =1.

You can be acknowledged by polling the register “**decodedone**” or interrupt after the MPEG engine finishes its JPEG decoding work.

The MPEG engine interrupt is 33(see Chapter 20 Interrupt Controller), and its interrupt mask setting is “**Eofen**”. If the MPEG interrupt occurs, you must clear it by writing the register “**Eofintclr**”.

I have not tried the Playback re-scale function of this MPEG engine and its register **Pbrscal[2:0]**, but you can decode a 640x480 JPEG file to 320x240 picture by setting **Pbrscal[2:0]=4**.

I have not tried the rotate function of this MPEG engine and its registers “**rotchg**”, “**rotxdn**” and “**rotydn**”.

I have not tried the Region of Interest function of this MPEG engine and its registers

“**ROIEn**”, “**ROIMBXOffsetLSB**”, “**ROIMBXDestLSB**”, “**ROIMBYOffsetLSB**”, “**ROIMBYDestLSB**”, “**ROIMBYOffsetMSB**”, “**ROIMBYDestMSB**”, “**ROIMBXOffsetMSB**”, “**ROIMBXDestMSB**”.

I have not tried the Vertical scale factor register “**Vsf**” and Horizontal scale factor register “**Hsf**”.

I have not tried the JPEG Thumbnail function of this MPEG engine and its registers “**Enthumb**”, “**ThumbLength**”, “**Thumb422mode**” and “**tmbOffaddr**”.

### 38.1.5 Coding Note

If AHB bus frequency is equal to MPEG Engine frequency, set P2M\_SETTING (0x8820\_00c0) =6, please.

If AHB bus frequency is double MPEG Engine frequency, set P2M\_SETTING (0x8820\_00c0) =10, please(see Chapter 17 SFTCFG).

When set the start address of image or VLC buffer memory start address, please keep the start address 1K boundaries.

MPEG engine only deal with the 4Y4U4Y4V pixel format.

If you want to generate the RGB565 pixel format from MPEG 4Y4U4Y4V, you can reference the Chapter 15 BLNDMA YUV2RGB conversion.

## 38.2 MPEG Engine Control Register Definition

### 38.2.1 P\_MP4\_MJWIDTHL(0x88220000): Image Width Low Byte Control Registers.

NAME	D7-D0
P_MP4_MJWIDTHL	MJWIDTHL

MJWIDTHL: Image Width, low byte, must be multiple of 16.

If rotation or scaling occurs, this value will be image width after rotation and scaling.

### 38.2.2 P\_MP4\_MJWIDTH(0x88220004): Image Width High Byte Control Registers.

NAME	D11-D8
P_MP4_MJWIDTH	MJWIDTH

MJWIDTH: Image Width, High byte.

### 38.2.3 P\_MP4\_MJHEIGHTL(0x88220008): Image Height Low byte Control Registers.

NAME	D7-D0
P_MP4_MJHEIGHTL	MJHEIGHTL

MJHEIGHTL: Image Height, low byte, must be multiple of 16.

If rotation or scaling occurs, this value will be image width after rotation and scaling.

### 38.2.4 P\_MP4\_MJHEIGHT(0x8822000C): Image Height High byte Control Registers.

NAME	D11-D8
P_MP4_MJHEIGHT	MJHEIGHT

MJHEIGHT: Image HEIGHT, High byte.

**38.2.5 P\_MP4\_MJHOFFSETL(0x88220010): Image Horizontal Offset Low Byte Control Registers.**

NAME	D7-D0
P_MP4_MJHOFFSETL	MJHOFFSETL

MJHOFFSETL: Image Horizontal offset, low byte, byte boundary.

**38.2.6 P\_MP4\_MJHOFFSET(0x88220014): Image Horizontal Offset Control Registers.**

NAME	D11-D8
P_MP4_MJHOFFSET	MJHOFFSET

MJHOFFSET: Image Horizontal offset, high byte.

**38.2.7 P\_MP4\_MJVOFFSETL(0x88220018): Image Vertical Offset Low Byte Control Registers.**

NAME	D7-D0
P_MP4_MJVOFFSETL	MJVOFFSETL

MJVOFFSETL: Image Vertical offset, low byte, byte boundary.

**38.2.8 P\_MP4\_MJVOFFSET(0x8822001C): Image Vertical Offset Control Registers.**

NAME	D11-D8
P_MP4_MJVOFFSET	MJVOFFSET

MJVOFFSET: Image Vertical offset, high byte.

**38.2.9 P\_MP4\_VLCOFFADDRL(0x88220020): VLC Bit Stream Starting Offset Address Low Byte**

NAME	D7-D0
P_MP4_VLCOFFADDRL	VLCOFFADDRL

VLCOFFADDRL: VLC Bit Stream Starting Offset Address, low byte, byte boundary.

**38.2.10 P\_MP4\_VLCOFFADDRM(0x88220024): VLC Bit Stream Starting Offset Address Middle Byte**

NAME	D15-D8
P_MP4_VLCOFFADDRM	VLCOFFADDRM

VLCOFFADDRM: VLC Bit Stream Starting Offset Address, Middle byte.

**38.2.11 P\_MP4\_VLCOFFADDR(0x88220028): VLC Bit Stream Starting Offset Address High Byte**

NAME	D23-D16
P_MP4_VLCOFFADDR	VLCOFFADDR

VLCOFFADDR: VLC Bit Stream Starting Offset Address, High byte.

**38.2.12 P\_MP4\_TMBOFFADDRL(0x88220040): Thumb Nail Starting Address Low Byte**

NAME	D7-D0
P_MP4_TMBOFFADDRL	TMBOFFADDRL

TMBOFFADDRL: Thumb nail starting address, low byte, word boundary, bit 0 is ignored.

### 38.2.13 P\_MP4\_TMBOFFADDRM(0x88220044): Thumb Nail Starting Address Middle Byte

NAME	D15-D8
P_MP4_TMBOFFADDRM	TMBOFFADDRM

TMBOFFADDRM: Thumb nail starting address, Middle byte.

### 38.2.14 P\_MP4\_TMBOFFADDR(0x88220048): Thumb Nail Starting Address High Byte Registers.

NAME	D23-D16
P_MP4_TMBOFFADDR	TMBOFFADDR

TMBOFFADDR: Thumb nail starting address, High byte.

### 38.2.15 P\_MP4\_YUVSEL(0x8822004C): YUV Group Selection Registers.

NAME	D0
P_MP4_YUVSEL	YUV_TYPE

YUV\_TYPE: YUV Group Selection

0: YUV420/YUV422

1: YUV444/YUV411

### 38.2.16 P\_MP4\_CONTROL(0x88220050): MJPG Control Registers.

NAME	D7	D6	D5	D4	D3	D2-D0
P_MP4_CONTROL	FULLRAN GEUV	FULLRAN GEY	JPG_BW	BW_MOD E	YUV_SEL	OPT_MO DE

OPT\_MODE: MJPG Operation Mode Selection

000: JPG encode mode

001: JPG decode mode

010: MPEG 1 encode mode

011: MPEG 1 decode mode

110: MPEG 4 encode mode

111: MPEG 4 decode mode

YUV\_SEL: YUV Mode Select Base Mode Setting depends on YUV Group Select (0x8822004C)

If P\_MP4\_YUVSEL is 0

0: YUV420 mode

1: YUV422 mode

If P\_MP4\_YUVSEL is 1

0: YUV411 mode

1: YUV444 mode

BW\_MODE: Black/White mode for JPG and MPEG

0: Color Mode

1: Black/White mode

Note: For MPEG, UV fill zero.

For JPG it B/W mode depends on 0x88220050 bit 5.

JPEG\_BW: JPG Black/White mode selection

0: JPG Black/White standard mode

1: UV fill zero

FULLRANGEY: YUV full range selection for Y

0: Full range

$Y' = 0 \sim 255$

1: Not Full Range

$Y' = 16 \sim 235$

FULLRANGEUV: YUV Full Range Selection for UV

0: Full Range

$U' = U$

$V' = V$

1: Not Full Range

$U' = (U * 224) / 256$

$V' = (V * 224) / 256$

### 38.2.17 P\_MP4\_SETTING(0x88220054): MJPG Setting Registers.

NAME	D7	D6	D5	D4	D3	D2-D0
P_MP4_SETTING	RANGESAT	ROTYDN	ROTXDN	ROTCHG		PBRSCAL

PBRSCAL: Play Back Rescale Factor, Only use for JPEG decompress

000: 8/8

001: 1/8

010: 2/8

011: 3/8

100: 4/8

101: 5/8

110: 6/8

111: 7/8

ROTCHG: X/Y change, it is not valid for YUV422 image.

If this bit is 1, Image width and height (defined at 0x88220000~0x8822000C) are width and height after rotating

ROTXDN: X mirror

ROTYDN: Y mirror

NOTE: Rotchg, rotxdn, rotydn control direction of original image. They are only used at compress mode.

	ROTCHG	ROTXDN	ROTYDN
normal	0	0	0
Rotate 90'	1	0	1
Rotate 180'	0	1	1
Rotate 270'	1	1	0
Horizontal Mirror	0	1	0
Vertical Mirror	0	0	1
HOR. / VER. Mirror	0	1	1

RANGESAT: Saturate input YUV data >= 250 to 250

### 38.2.18 P\_MP4\_HSFACOR(0x88220058): Horizontal Scale Factor Registers.

NAME	D7-D0
P_MP4_HSFACOR	HSF

HSF: Horizontal scale factor for read original image. It is only used at compress mode for image scale up. Scaling method is linear interpolation.

0 : no scale

others :  $Hsf[7:0] = (original\ width - 1) / (output\ width - 1) * 256$

### 38.2.19 P\_MP4\_VSFACOR(0x8822005C): Vertical Scale Factor Registers.

NAME	D7-D0
P_MP4_VSFACOR	VSF

VSF: Vertical scale factor for read original image. It is only used at compress mode for image scale up. Scaling method is linear interpolation.

0 : no scale

others :  $Vsf[7:0] = (original\ height - 1) / (output\ height - 1) * 256$

### 38.2.20 P\_MP4\_GOPVAL(0x88220060): P Frame number for MPEG Compress mode Select

NAME	D7-D0
P_MP4_GOPVAL	GOPVAL

GOPVAL: P frame number for MPEG compress mode

0 : I only

1 : IPIP.....

2: IPPIPP...

...

### 38.2.21 P\_MP4\_SETEN(0x88220064): Setting Enable

NAME	D6	D5	D4	D3	D2	D1	D0
P_MP4_SETEN	GOPSET TING	UNREST RICT	SOFMOD E	RCINISE T	BR_CTR LEN	HPELEN	GOPRST t

GOPRST: Rest GOP Counter

0: No Reset, GOP counter will auto back to 0 if it counts to GOP value(REG 0x88220060)

1: H/W GOP counter reset to zero.

**HPELEN:** Half Pel Enable  
 0: Disable  
 1: Enable  
**BR\_CTRLLEN:** H/W Bit Rate Control Enable  
 0: Disable  
 1: Enable  
**RCINISSET:** Rate Control Initial Setting  
 0: Not Set  
 1: Set  
**SOFMODE:** SOF Mode  
 0: Trigger by HW at video clip and pc cam mode  
 1: Always trigger by CPU  
**UNRESTRICT:** Unrestricted MC Enable, It is only used at MPEG4 Encoding, this function will increase some search time.  
 0: Disable  
 1: Enable  
**GOPSETTING:** I/P Frame Decision Depend  
 0: I/P frame decision is depend on H/W GOP counter  
 1: I/P frame decision is depend on Register (0x88220088), bit 0

#### 38.2.22 P\_MP4\_VARDTHRL(0x88220068): Variance Threshold LSB Byte

NAME	D7-D0
P_MP4_VARDTHRL	VARDTHRL

VARDTHRL: Variance threshold for inter/intra type decision, LSB byte

#### 38.2.23 P\_MP4\_VARDTHRM(0x8822006C): Variance Threshold MSB Byte

NAME	D15-D8
P_MP4_VARDTHRM	VARDTHRM

VARDTHRM: Variance threshold for inter/intra type decision, MSB byte

#### 38.2.24 P\_MP4\_INIAVGACT (0x88220070): Initial Average Activity Value

NAME	D7-D0
P_MP4_INIAVGACT	INIAVGACT

INIAVGACT: Initial average activity value

Integer part is 2 bits.

### 38.2.25 P\_MP4\_AVGACTWEIL(0x88220078): Normalize Average Activity Value Weighting LSB Byte

NAME	D7-D0
P_MP4_AVGACTWEIL	AVGACTWEIL

AVGACTWEIL: Normalize average activity value weighting, LSB byte.

The center value is  $(\text{IMAGE\_WIDTH}/16) * (\text{IMAGE\_HEIGHT}/16)$ .

If we decrease this value, the weight of normalize average will increase

### 38.2.26 P\_MP4\_AVGACTWEIM(0x8822007C): Normalize Average Activity Value Weight MSB Byte

NAME	D15-D8
P_MP4_AVGACTWEIM	AVGACTWEIM

AVGACTWEIM: Normalize average activity value weighting, MSB byte.

The center value is  $(\text{IMAGE\_WIDTH}/16) * (\text{IMAGE\_HEIGHT}/16)$ .

If we decrease this value, the weight of normalize average will increase.

### 38.2.27 P\_MP4\_RESET(0x88220080): Software Reset For MJPG Register.

NAME	D0
P_MP4_RESET	MJPGRST

MJPGRST: Soft ware reset for MJPG. This SW reset will clear all counter and state machine at MJPG. But register setting will not reset.

0 : disable

1 : enable S/W reset.

### 38.2.28 P\_MP4\_BUFCTRL(0x88220084): MJPG Buffer Controll Register.

NAME	D7	D6-D5	D4	D3-D2	D1	D0
P_MP4_BUFCTRL	VCMODE		ZEROMV		DoubleMode	DoubleEN

DoubleEN: Double Buffer Control

0: Disable

1: Enable

DoubleMode: Double Buffer Update Mode

0: From SOF

1: From Static DRAM control

ZEROMV: Fore Zero Motion Vector

0: Disable

1: Enable

VCMODE: Video Conference mode

**38.2.29 P\_MP4\_IFRAME(0x88220088): Frame Mode Register.**

NAME	D0
P_MP4_IFRAME	IFRAME

IFRAME: Frame Mode Selection

0: P Frame

1: I Frame

**38.2.30 P\_MP4\_TBL(0x8822008C): Thumb Nail Burst Length Register.**

NAME	D4	D3	D2-D0
P_MP4_TBL	Thumb422Mode	Mesramdis	ThumbLength

ThumbLength: Thumb nail burst length

Mesramdis: ME SRAM Disable

Thumb422Mode: For YUV422 Image, Thumb nail mode Select

0: No mapping

1: Mapping to YYYYYYYYUUVV data

**38.2.31 P\_MP4\_MESRAM(0x88220090): SRAM Setting Register.**

NAME	D6	D5-D4	D3	D2-D0
P_MP4_MESRAM	MCSRAMTEST	MCSRAMSEL	MESRAMTEST	MESRAMSel

MESRAMSel: CPU direct write me static ram selection, ME static ram is mapping to 0x88220000~0x882200ff

MESRAMTEST: CPU Direct Write ME SRAM

0: Disable

1: Enable

MCSRAMSEL: CPU direct write MC Static RAM selection, ME static ram is mapping to 0x88220000~0x882200ff

MCSRAMTEST: CPU Direct Write MC Static RAM

0: Disable

1: Enable

**38.2.32 P\_MP4\_PROBEMODE(0x88220094): H/W Probe Mode Register.**

NAME	D7-D0
P_MP4_PROBEMODE	PROBEMODE

**38.2.33 P\_MP4\_MEMCTRL (0x88220098): Memory Control Register.**

NAME	D3	D2	D1-D0
P_MP4_MEMCTRL	MemGroupSel	BirstMode	

BirstMode: Birst Mode

0: Disable

1: Enable  
MemGroupSel: Birst Group Selector  
0: Disable  
1: Enable

#### 38.2.34 P\_MP4\_SOFCTRL(0x882200A0): Start of Compressing Control Register..

NAME	D4	D3	D2	D1	D0
P_MP4_SOFCTRL	MJPGDEC	SOFINFO	SKIPFRAME	EOFSTATUS	MJPGENC

MJPGENC: MJPG Encode Control

Read: when you write this bit, this bit will be 1. It will be set 0 until this image is done

Write: At video clip mode , Starting of Compressing one image is triggered by CDSP  
At other mode, Starting of Compressing / Decompress one image is writing this bit with one

EOFSTATUS: When write 0x028 bit 0 with one, this bit will reset to zero. It will be set to one by Hw when EOF occurs.

SKIPFRAME: Skip frame status

0: no skip  
1: skip frame.

SOFINFO: CPU Starting of Compressing information.

MJPGDEC: MJPG Decode Control

Read: It will be set 0 until this image is done

Write: At Video clip mode, Starting of Compressing one image is triggered by CDSP.  
At other mode, Starting of Compressing / Decompress one image is writing this bit with one

#### 38.2.35 P\_MP4\_GOPCOUNTER(0x882200A4): GOP Counter Monitor

NAME	D3-D0
P_MP4_GOPCOUNTER	IFRAME

IFRAME: GOP Counter Monitor

0: I Frame

Others: P Frame

**38.2.36 P\_MP4\_AVGACT(0x882200A8): Normalize Average Activity Value for Every Frame**

NAME	D7-D0
P_MP4_AVGACT	AVGACT

AVGACT: Normalize average activity value for every frame. It is likely to TM5 model.  
The difference is that we use sum of absolute difference, not sum of square difference. Integer part is 2 bits.

**38.2.37 P\_MP4\_AVGVARD(0x882200AC): Average Difference Variance**

NAME	D7-D0
P_MP4_AVGVARD	AVGVARD

**38.2.38 P\_MP4\_MEBISTFAIL(0x882200B0): ME Bist Test Fail**

NAME	D7-D0
P_MP4_MEBISTFAIL	MEBISTFAIL

**38.2.39 P\_MP4\_MCBISTFAIL(0x882200B4): MC Bist Test Fail**

NAME	D7-D0
P_MP4_MCBISTFAIL	MCBISTFAIL

**38.2.40 P\_MP4\_MEMCBISTFINISH(0x882200B8): ME/MC Bist Test Finish**

NAME	D1	D0
P_MP4_MEMCBISTFINISH	MCBISTFINISH	MEBISTFINISH

**38.2.41 P\_MP4\_HSFINI(0x882200C0): Horizontal Initial Scaling Factor**

NAME	D7-D0
P_MP4_HSFINI	HSFINI

**38.2.42 P\_MP4\_VSFINI(0x882200C0): Vertical Initial Scaling Factor**

NAME	D7-D0
P_MP4_VSFINI	VSFINI

**38.2.43 P\_MP4\_UVMODE(0x882200C0): MC Write UV Transfer Mode**

NAME	D5	D4	D3	D2-D0
P_MP4_UVMODE	UV420TO422EN	UV420T422	MCWRMODE	

MCWRMODE: MC Write Mode

UV420TO422: If UV420 to 422 Enable, UV420 to 422 is operated  
Else 0 when compress.

UV420TO422EN: Enable for UV420 to 422

#### 38.2.44 P\_MP4\_SFL(0x882200CC): MC/ME Scaling Factor Burst Length

NAME	D7-D4	D3-D0
P_MP4_SFL	MCHSFL	MEHSFL

MEHSEL: Me Scaling Factor Burst Length  
Length = 16 + OddSTA – MCHSFL

MCHSEL: Mc Scaling Factor Burst Length  
Length = 8 + OddSTA - MEHSFL

#### 38.2.45 P\_MP4\_INTMASK(0x882200D0): Interrupt Mask Register.

NAME	D5	D4	D3	D2	D1	D0
P_MP4_INTMASK	VLDLAST	DECERR	SOFEN	EOFEN	JPGEN	MCINTEN

MCINTEN: Interrupt Mask For MC

JPGEN: Interrupt Mask For JPEG

EOFEN: Interrupt Mask For EOF

SOFEN: Interrupt Mask For SOF

DECERR: Interrupt Mask For Decoder Error

VLDLAST: Interrupt Mask For VLD Stream Decompress

#### 38.2.46 P\_MP4\_INTCLR(0x882200D4): Interrupt Clear Register.

NAME	D5	D4	D3	D2	D1	D0
P_MP4_INTCLR	VLDRESTART ART	DECERRCLR LR	SOFINTCLR R	EOFINTCLR R		MCINTCLR R

MCINTCLR: MC Interrupt Control  
Read 0: MC Interrupt Status  
Write 1: Clear the Flag

EOFINTCLR: EOF Interrupt Control  
Read 0: EOF Interrupt Status  
Write 1: Clear the Flag

SOFINTCLR: SOF Interrupt Control  
Read 0: SOF Interrupt Status  
Write 1: Clear the Flag

DECERRCLR: Decode Error Interrupt Control  
Read 0: Decode Error Interrupt Status  
Write 1: Clear the Flag

VLDRESTART: VLD Restart Interrupt Control  
Read 0: VLD Restart Interrupt Status  
Write 1: Clear the Flag

### 38.2.47 P\_MP4\_DISGATED(0x882200D8): Disable for Gated Clock Register.

NAME	D1	D0
P_MP4_DISGATED	ENMDRST	DISGATED

DISGATED: Disable for Gated Clock

0: Disable

1: Enable

ENMDRST: Enable for Mode Reset

0: Disable

1: Enable

### 38.2.48 MPEG4/JPEG VLC Buffer Size

NAME	Address	D7-D0
P_MP4_VLCBUFAL	0x882200DC	VLCBUFAL
NAME	Address	D15-D8
P_MP4_VLCBUFAM	0x882200E0	VLCBUFAM
NAME	Address	D23-D16
P_MP4_VLCBUFA	0x882200E4	VLCBUFA
NAME	Address	D7-D0
P_MP4_VLCBUFBL	0x882200E8	VLCBUFBL
NAME	Address	D15-D8
P_MP4_VLCBUFBM	0x882200EC	VLCBUFBM
NAME	Address	D23-D16
P_MP4_VLCBUFB	0x882200F0	VLCBUFB

VLCBUFAL: VLC A/B/C Buffer Size low byte

VLCBUFAM: VLC A/B/C Buffer Size Middle Byte

VLCBUFA: VLC A/B/C Buffer Size High Byte

### 38.2.49 P\_MP4\_DECWIDTHL(0x88220100): Decode Image Width Low Byte Control Registers.

NAME	D7-D0
P_MP4_DECWIDTHL	DECWIDTHL

DECWIDTHL: Decode Image width, low byte, must be multiple of 16.

If rotation or scaling occurs, This value will be image width after rotation and scaling.

### 38.2.50 P\_MP4\_DECWIDTH(0x88220104): Decode Image Width High Byte Control Registers.

NAME	D11-D8
P_MP4_DECWIDTH	DECWIDTH

DECWIDTH: Decode Image height, high byte

### 38.2.51 P\_MP4\_MJHEIGHTL(0x88220108): Decode Image Height Low byte Control Registers.

NAME	D7-D0
P_MP4_DECHEIGHT	DECHEIGHT

DECHEIGHT: Decode Image height, low byte, must be multiple of 16.

If rotation or scaling occurs, This value will be image height after rotation and scaling.

### 38.2.52 P\_MP4\_DECHEIGHT(0x8822010C): Decode Image Height High byte Control Registers.

NAME	D11-D8
P_MP4_DECHEIGHT	DECHEIGHT

DECHEIGHT: Decode Image HEIGHT, High byte.

### 38.2.53 Decode VLC Bit Stream Starting Offset

NAME	Address	D7-D0
P_MP4_DVLCOFFADDRL	0x88220110	DVLCOFFADDRL
NAME	Address	D15-D8
P_MP4_DVLCOFFADDRM	0x88220114	DVLCOFFADDRM
NAME	Address	D23-D16
P_MP4_DVLCOFFADDR	0x88220118	DVLCOFFADDR

Note: Decode VLC Bit Stream Starting Offset Low/Middle/High is Byte Boundard.

### 38.2.54 Last Memory Address

NAME	Address	D7-D0
P_MP4_LASTMEMADDRL	0x8822011C	LASTMEMADDRL
NAME	Address	D15-D8
P_MP4_LASTMEMADDRM	0x88220120	LASTMEMADDRM
NAME	Address	D23-D16
P_MP4_LASTMEMADDR	0x88220124	LASTMEMADDR

Note: Last Memory Address(word alignment),

$$\text{LastMemAddr} = ((\text{Size} + \text{VLCOFFADDR}) - (\text{LastLength} + 1))$$

### 38.2.55 P\_MP4\_LASTLENGTH(0x88220128): Last Memory Length

NAME	D3-D0
P_MP4_LASTLENGTH	LastLength

LastLength: Last Memory Length

$$\text{LastLength} = (\text{Size} + \text{VLCOFFADDR} - 1) \% 16$$

### 38.3 Texture Engine Control Register Definition

#### 38.3.1 Quantization Table 1

NAME	Address
<b>P_MP4_QTABLE1_STR</b>	0x88220400
<b>P_MP4_QTABLE1_END</b>	0x882204FC

JPEG: Quantization Table for Y-Component

MPGE4: Invalid

The Values of Q are filled in the raster-scan order.

#### 38.3.2 Quantization Table 2

NAME	Address
<b>P_MP4_QTABLE2_STR</b>	0x88220500
<b>P_MP4_QTABLE2_END</b>	0x882205FC

JPEG: Quantization Table for C-Component

MPGE4: Invalid

The Values of Q are filled in the raster-scan order.

#### 38.3.3 P\_MP4\_NOQTBL(0x88220600): Q-Table Currently Used Register.

NAME	D7-D0
<b>P_MP4_NOQTBL</b>	NOQTBL

NOQTBL: Index of the Q-table Currently used, this register is for internal check only.

#### 38.3.4 P\_MP4\_QTBLSET(0x88220604): Q-Table Setting Register.

NAME	D3	D2	D1	D0
<b>P_MP4_QTBLSET</b>	EXHEADEN	SKIPTYPE	QSCALEEN	QVALUE

QVALUE: 0: Original  
1: Quantizer Coefficients are set to one

QSCALEEN: Enable qscale function.  
0: Use QTable1 & QTable2 as the quantize step size.  
1: Hardwire quantize step size.

SKIPTYPE: Used for MPEG skip MCU algorithm selection  
0: Skip MCU if the luminance blocks are zero.  
1: Skip MCU if all the blocks are zero.

EXHEADEN: Extra Information is appended before MPEG4 bit stream  
0: Disable  
1: Enable, the 0<sup>th</sup> byte is the number of skip frames  
the 1<sup>st</sup> to 3<sup>rd</sup> bytes is the frame time information

### 38.3.5 P\_MP4\_QSRAMEN(0x88220608): Q-Table SRAM Control Register.

NAME	D0
P_MP4_QSRAMEN	QSRAMEN

QSRAMEN: Q-Table SRAM Control

0: CPU cannot access Q-Table SRAM

1: Enable CPU to access Q-Table SRAM

### 38.3.6 P\_MP4\_ENTHUMB (0x8822060C): Thumb Nail Image Generation Control Register.

NAME	D1	D0
P_MP4_ENTHUMB	STOPEN	ENTHUMB

ENTHUMB: 0: Disable Thumb Nail Image Generation

1: Enable Thumb Nail Image Generation

STOPEN: 0: Normal operation.

1: Set this bit to enable stop mode. The decompression engine stops for every block until the flag " blockend" is cleared

### 38.3.7 P\_MP4\_JFIF (0x88220610): JFIF Compatible Control.

NAME	D0
P_MP4_JFIF	JFIF

JFIF: JFIF Compatible Control bit.

0: Original

1: Compatible with JFIF bit stream.

In a standard JFIF bit stream, 0XFF is used as a marker to indicated the start point of a special control sequence. To represent VLC data 0XFF, 0X00 must be appended to distinguish from the marker.

### 38.3.8 P\_MP4\_TRUNTRES(0x88220614):

NAME	D1	D0
P_MP4_JFIF	DCTresolution	Truncated

Truncated: 0: Rounded after quantization

1: Truncated after quantization

DCTresolution: 0: 9 bits decimal for intra macroblock, 8 bit decimal for non-intra macroblock.

1: 2 bits decimal for intra macroblock, 1 bit decimal for non-intra macroblock.

### 38.3.9 P\_MP4\_VLCBIT (0x88220618):

NAME	D2-D0
P_MP4_VLCBIT	VLCBIT

VLCBIT: Indicates the number of stuffing bits in the last byte of the VLC stream.

0: non-stuffing 1' s

1: 1 stuffing 1.

.....

7: 7 stuffing 1' s.

This information is passed to the PC in the Video mode for the software decoding to double-check the JPEG VLC stream data integrity.

### 38.3.10 P\_MP4\_JFIFEND (0x8822061C):

NAME	D0
P_MP4_JFIFEND	JFIFEND

JFIFEND: After JFIF-compliant data is decompressed, check this bit to identify if data stream is decompressed completely or not.

0: error happened.

1: decompressed completely.

### 38.3.11 Restart MCU

NAME	Address	D7-D0
P_MP4_RESTARTMCUL	0x88220620	RESTARTMCUL
NAME	Address	D15-D8
P_MP4_RESTARTMCU	0x88220624	RESTARTMCU

RESTARTMCUX: If the vlc stream includes MCU restart code, fill the restart MCU number.

Otherwise, set reset mcu number as 0 to disable restart code decompression.

### 38.3.12 P\_MP4\_IFFRAMEQSCALE(0x88220640): Quantizer Scale Value for I-Frame

NAME	D4-D0
P_MP4_IFFRAMEQSCALE	IFFRAMEQSCALE

### 38.3.13 P\_MP4\_PFRAMEQSCALE(0x88220644): Quantizer Scale Value for P-Frame

NAME	D4-D0
P_MP4_PFRAMEQSCALE	PFRAMEQSCALE

### 38.3.14 P\_MP4\_MATCHCNT (0x88220648): Match Code Function Control

NAME	D1-D0
P_MP4_MATCHCNT	MATCHCNT

MATCHCNT: Match Code Function Control

0: Disable match code function

- 1: Enable match 2 bytes  
 2: Enable match 4 bytes

### 38.3.15 Match Code

NAME	Address	D7-D0
P_MP4_MATCHCODE0	0x8822064C	MATCHCODE0
NAME	Address	D15-D8
P_MP4_MATCHCODE1	0x88220650	MATCHCODE1
NAME	Address	D23-D16
P_MP4_MATCHCODE2	0x88220654	MATCHCODE2
NAME	Address	D31-D24
P_MP4_MATCHCODE3	0x88220658	MATCHCODE3

### 38.3.16 P\_MP4\_OFFSET(0x8822065C): Offset Value

NAME	D7-D0
P_MP4_MATCHCODE0	OFFSET

### 38.3.17 P\_MP4\_VOPTIMEINC (0x88220660): VOP Time Increment Mode Selection

NAME	D7-D0
P_MP4_VOPTIMEINC	VOPTIMEINC

VOPTIMEINC: VOP time increment mode selection

- 0: VOP time increment is based on 0x8822\_0664 and 0x8822\_0668  
 1: VOP time increment is based on 0x8822\_0688 and 0x8822\_068C

### 38.3.18 P\_MP4\_MSCNT (0x88220664): Mini Second Counter

NAME	D5-D0
P_MP4_MSCNT	MSCNT

### 38.3.19 P\_MP4\_MSPLUSCNTEN (0x88220668): Enable one extra mini second added counter

NAME	D7
P_MP4_MSPLUSCNTEN	MSPLUSCNTEN

### 38.3.20 P\_MP4\_MSPLUSCTRL(0x8822066C): Extra mini second added Control

NAME	D7	D6-D5	D4-D0
P_MP4_MSPLUSCTRL	MSPLUSCNTEN		MSPLUSCNT

MSPLUSCNT: One Extra mini Second Added Counter

MSPLUSCNTEN: Enable One Extra mini Second Added Counter

### 38.3.21 VOP Time Increment Resolution

NAME	Address	D7-D0
P_MP4_VOPTimeIncResL	0x88220680	VOPTimeIncResL
NAME	Address	D15-D8
P_MP4_VOPTimeIncRes	0x88220684	VOPTimeIncRes

### 38.3.22 VOP Time Increment Unit

NAME	Address	D7-D0
P_MP4_VOPTimeIncL	0x88220688	VOPTimeIncL
NAME	Address	D15-D8
P_MP4_VOPTimeInc	0x8822068C	VOPTimeInc

### 38.3.23 P\_MP4\_VOPTimeIncLen(0x88220690): Length of VOP time Increment

NAME	D3-D0
P_MP4_VOPTimeIncLen	VOPTimeIncLen

VOPTimeIncLen= Length of VOP time Increment Resolution – 1

### 38.3.24 TM5 Model: Reaction Parameter

NAME	Address	D7-D0
P_MP4_REACTL	0x88220698	REACTL
NAME	Address	D15-D8
P_MP4_REACT	0x8822069C	REACT

### 38.3.25 TM5 Model: Virtual Buffer Fullness Initial Values for I picture

NAME	Address	D7-D0
P_MP4_D0_IFramL	0x882206A0	D0_IFramL
NAME	Address	D13-D8
P_MP4_D0_IFram	0x882206A4	D0_IFram

### 38.3.26 TM5 Model: The Predict byte Counter Each Macro In I Frame

NAME	Address	D7-D0
P_MP4_PerIFrameL	0x882206A8	PerIFrameL
NAME	Address	D9-D8
P_MP4_PerIFrame	0x882206AC	PerIFrame

### 38.3.27 TM5 Model: The Predict byte Counter for Each Macro In P Frame

NAME	Address	D7-D0
P_MP4_PerPFrameL	0x882206B0	PerPFrameL
NAME	Address	D9-D8
P_MP4_PerPFrame	0x882206B4	PerPFrame

### 38.3.28 TM5 Model: Q-Scale Bound

NAME	Address	D4-D0
P_MP4_QScaleUpperB	0x882206B8	QScaleUpperB
NAME	Address	D4-D0
P_MP4_QScaleLowerB	0x882206BC	QScaleLowerB

### 38.3.29 TM5 Model: Virtual Buffer Fullness Initial Values for P picture

NAME	Address	D7-D0
P_MP4_D0_PFrameL	0x882206C0	D0_PFrameL
NAME	Address	D13-D8
P_MP4_D0_PFrame	0x882206C4	D0_PFrame

### 38.3.30 Compressed Data Size

NAME	Address	D7-D0
P_MP4_VLCSizeL	0x882206C8	VLCSizeL
NAME	Address	D15-D8
P_MP4_VLCSizeM	0x882206CC	VLCSizeM
NAME	Address	D23-D16
P_MP4_VLCSize	0x882206D0	VLCSize

### 38.3.31 TM5 Model: Sum of Every Macro Block Q-Scale of a Frame

NAME	Address	D7-D0
P_MP4_QSumL	0x882206D4	QSumL
NAME	Address	D13-D8
P_MP4_QSum	0x882206D8	QSum

### 38.3.32 P\_MP4\_SRAM\_CS\_N(0x882206F8):

NAME	D0
P_MP4_SRAM_CS_N	SRAM_CS_N

SRAM\_CS\_N:      0:    Turn off static ram(default)  
                      1:    Turn on static ram

### 38.3.33 P\_MP4\_AUTO\_RESET(0x882206FC):

NAME	D0
P_MP4_AUTO_RESET	AUTO_RESET

AUTO\_RESET:    0:    Turn on auto reset when decompression(default)  
                      1:    Turn off auto reset when decompression

**38.3.34 P\_MP4\_SRAM\_TEST\_CTRL (0x88220780):**

NAME	D5-D4	D3-D2	D1-D0
<b>P_MP4_SRAM_TEST_CTRL</b>	SRAMPAGE		SRAMSEL

SRAMSEL: Internal SRAM test selection

SRAMPAGE: Internal SRAM test page selection

**38.3.35 P\_MP4\_JPG\_SEL (0x88220784): JPEG Probe Bus Selection**

NAME	D7-D0
<b>P_MP4_JPG_SEL</b>	JPG_SEL

**38.3.36 P\_MP4\_BistMode (0x88220788): SRAM Bist Mode Control**

NAME	D1	D0
<b>P_MP4_BistMode</b>	MEMGroupSel	BistMode

BistMode: SRAM Bist Mode Enable

0: Disable

1: Enable SRAM Bist Function

MEMGroupSel: SRAM Bist Mode Select

**38.3.37 P\_MP4\_BistFail (0x8822078C): SRAM Bist Mode Fail Information Control**

NAME	D4-D0
<b>P_MP4_BistFail</b>	BistFail

BistFail: SRAM Bist Mode Fail Information

0: Test Succeed

1: Test Fail

**38.3.38 P\_MP4\_BlockEnd (0x88220790):**

NAME	D0
<b>P_MP4_BlockEnd</b>	BlockEnd

In Stop Mode, active high after end of block. Write 0 to clear this flag.

**38.3.39 P\_MP4\_DeHuffmanen (0x88220798): Dehuffman mode Control**

NAME	D0
<b>P_MP4_DeHuffmanen</b>	Dehuffman

Dehuffman: Dehuffman mode selection

0: disable hardware de-huffman function

1: enable hardware de-huffman function

### 38.3.40 P\_MP4\_DeHuffmanrdy (0x8822079C):

NAME	D0
P_MP4_DeHuffmanrdy	Dehuffmanrdy

DeHuffmanrdy: Firmware dehuffman flow:

Check if inverse quantization engine is ready to receive VLD data extracted from firmware.

### 38.3.41 Firmware Dehuffman

NAME	Address	D7-D0
P_MP4_VLDDataL	0x882207A0	VLDDataL
NAME	Address	D10-D8
P_MP4_VLDData	0x882207A4	VLDData

Firmware dehuffman flow:

VLDData is feed 0x882207A0, 0x882207A4 data ports.

### 38.3.42 P\_MP4\_VIDEORstMode (0x882207A8): Video Reset

NAME	D0
P_MP4_VIDEORstMode	VRstMode

VRstMode: 0: Video Reset until the First frame Occurs

1: Video Reset until the First non-skip frame occurs

### 38.3.43 P\_MP4\_NACTEN (0x882207AC): Enable activity value from ME

NAME	D0
P_MP4_NACTEN	NACTEN

NACTEN: 0: Disable

1: Enable

### 38.3.44 Number of IntraMB in current Frame

NAME	Address	D7-D0
P_MP4_IMCUCNTL	0x882207B0	IMCUCNT L
NAME	Address	D15-D8
P_MP4_IMCUCNT	0x882207B4	IMCUCNT

### 38.3.45 Number of InterMB in current Frame

NAME	Address	D7-D0
P_MP4_PMCUCNTL	0x882207B8	PMCUCNT L
NAME	Address	D15-D8
P_MP4_PMCUCNT	0x882207BC	PMCUCNT

### 38.3.46 Number of Skipped MB in current Frame

NAME	Address	D7-D0
<b>P_MP4_SKIPMCUCNTL</b>	0x882207C0	SKIPMCUCNTL
NAME	Address	D15-D8
<b>P_MP4_SKIPMCUCNT</b>	0x882207C4	SKIPMCUCNT

### 38.3.47 P\_MP4\_H263\_CTRL(0x882207C8): H263 format Control Registers.

NAME	D7	D6	D5-D4	D3-D1	D0
<b>P_MP4_H263_CTRL</b>	ACPREDEn	FOURMVEn		H263Format	H263En

H263En: Enable H.263 Encoding/Decoding

0: Disable

1: Enable

H263Format: Specify H.263 Format

001: Sub-QCIF(128x96)

010: QCIF(176x144)

011: CIF(352x288)

100: 4CIF(704x576)

FOURMVEn: Enable 4MV Decoding mode

0: Disable

1: Enable

ACPREDEn: Enable AC Prediction decoding mode

0: Disable

1: Enable

### 38.3.48 P\_MP4\_H263\_Struct(0x882207CC):

NAME	D7-D4	D3-D1	D0
<b>P_MP4_H263_Struct</b>	MBNumLen		H263GOBEn

H263GOBEn: Enable H.263 GOB structure for Encoding/Decoding

MBNumLen: Specify number of bits needed for certain resolution.

### 38.3.49 Sum of Absoulte Horizontal Motion Vectors(HalfPel Unit)

NAME	Address	D7-D0
<b>P_MP4_HMVSumL</b>	0x882207D0	HMVSumL
NAME	Address	D14-D8
<b>P_MP4_HMVSum</b>	0x882207D4	HMVSum

### 38.3.50 Sum of Absoulte Vertical Motion Vectors(HalfPel Unit)

NAME	Address	D7-D0
<b>P_MP4_VMVSumL</b>	0x882207D8	VMVSumL
NAME	Address	D14-D8
<b>P_MP4_VMVSum</b>	0x882207DC	VMVSum

### 38.3.51 P\_MP4\_TSRAM\_CTRL(0x882207E0): TSRAM Control Register

NAME	D4-D2	D1-D0
<b>P_MP4_H263_Struct</b>	TSRAMPage	TSRAMMode

TSRAMMode: 0: Host Bridge R/W TSRAM  
 1: CPU R/W TSRAM  
 2: CPU R/W TSRAM Registers and JPEG Decoding  
 3: TSRAM is used for MV Histogram

TSRAMPage: TSRAM has 8 pages and 64 bytes per page

### 38.3.52 P\_MP4\_GRegion\_CTRL(0x882207E4):

NAME	D2	D1	D0
<b>P_MP4_GRegion_CTRL</b>	ROIEn		DISGatedClock

DISGatedClock: Gated Clock Disable

0: Enable Gated Clock  
 1: Disable Gated Clock

ROIEn: Enable Region of Interest mode

### 38.3.53 P\_MP4\_MJPG\_CTRL (0x882207E8):

NAME	D4	D3	D2	D1	D0
<b>P_MP4_MJPG_CTRL</b>	MBHoldRdy	DecErrEn	MBHoldChk	MBHoldInt	MBHoldEn

MBHoldEn: Enable MJPG Stops after Decoding each MB

0: Disable  
 1: Enable

MBHoldInt: Enable Sending a Interrupt When Stopped

0: Disable  
 1: Enable

MBHoldChk: Clear Hold Status and Continue Decoding

DecErrEn: Enable a halt when bit stream and Huffman table entry mismatched.

MBHoldRdy: Flag Indicates the MJPG has stopped

**38.3.54 P\_MP4\_ROIMBXOffsetLSB(0x882207EC):LSB of Horizontal Starting MB Coordinate of ROI**

NAME	D7-D0
P_MP4_ROIMBXOffsetLSB	ROIMBXOffsetLSB

**38.3.55 P\_MP4\_ROIMBXDestLSB(0x882207F0):LSB of Horizontal End MB Coordinate of ROI**

NAME	D7-D0
P_MP4_ROIMBXDestLSB	ROIMBXDestLSB

**38.3.56 P\_MP4\_ROIMBYOffsetLSB(0x882207F4):LSB of Vertical Starting MB Coordinate of ROI**

NAME	D7-D0
P_MP4_ROIMBYOffsetLSB	ROIMBYOffsetLSB

**38.3.57 P\_MP4\_ROIMBYDestLSB(0x882207F8):LSB of Vertical End MB Coordinate of ROI**

NAME	D7-D0
P_MP4_ROIMBYDestLSB	ROIMBYDestLSB

**38.3.58 P\_MP4\_ROIMBMSB(0x882207FC):MSB Coordinate of ROI**

NAME	D5	D4	D2-D2	D1	D0
P_MP4_ROIMBMSB	ROIYDest	ROIYOffset		ROIYDest	ROIYOffset

ROIYOffset: MSB of Vertical Starting MB Coordinate of ROI

ROIYDest: MSB of Vertical End MB Coordinate of ROI

ROIYOffset: MSB of Horizontal Starting MB Coordinate of ROI

ROIYDest: MSB of Horizontal End MB Coordinate of ROI

**38.3.59 Huffman Table**

NAME	Address
P_MP4_Huffman_Str	0x88220800
P_MP4_Huffman_End	0x882208FC

When JPEG Decoding(TSRAMMode = 0, 1 or 2), the TSRAM is used for Storing Huffman Table. The TSRAMPage Must be combined to reach all address range.

**38.3.60 Luminance DC code word**

NAME	Address
P_MP4_YDCW_Str	0x88220800
P_MP4_YDCW_End	0x8822085C

Luminance DC code word(TSRAMMode = 2), 8 bits register.

**38.3.61 Luminance address offset for DC first codeword**

NAME	Address
P_MP4_YDCS_Str	0x88220860
P_MP4_YDCS_End	0x8822087C

Luminance address offset for DC first codeword (TsrAMMode = 2), 8bits register.

### 38.3.62 Luminance DC Huffman table content

NAME	Address
P_MP4_YDCV_Str	0x88220880
P_MP4_YDCV_End	0x8822089C

Luminance DC Huffman table content (TsramMode = 2), 8 bits register.

### 38.3.63 Chrominance DC codeword

NAME	Address
P_MP4_CDCW_Str	0x882208C0
P_MP4_CDCW_End	0x8822091C

Chrominance DC codeword (TsramMode = 2), 8 bits register.

### 38.3.64 Chrominance address offset for DC codeword

NAME	Address
P_MP4_CDCS_Str	0x88220920
P_MP4_CDCS_End	0x8822093C

Chrominance DC codeword (TsramMode = 2), 8 bits register.

### 38.3.65 Chrominance DC Huffman table content

NAME	Address
P_MP4_CDCV_Str	0x88220920
P_MP4_CDCV_End	0x8822093C

Chrominance DC Huffman table content (TsramMode = 2), 8 bits register.

### 38.3.66 Luminance AC Huffman table codeword

NAME	Address
P_MP4_YACW_Str	0x88220980
P_MP4_YACW_End	0x882209DC

Luminance AC Huffman table codeword (TsramMode = 2), 8 bits register.

### 38.3.67 Luminance AC Huffman table address offset for AC first 1-bit codeword

NAME	Address
P_MP4_YACS_Str	0x882209E0
P_MP4_YACS_End	0x88220A34

Luminance AC Huffman table address offset for AC first 1-bit codeword (TsramMode = 2), 8 bits register.

### 38.3.68 Chrominance AC Huffman table codeword

NAME	Address
P_MP4_CACW_Str	0x88220A40
P_MP4_CACW_End	0x88220A90

Chrominance AC Huffman table codeword (TsramMode = 2), 8 bits register.

### 38.3.69 Chrominance AC Huffman table address offset for codeword

NAME	Address
P_MP4_CACS_Str	0x88220AA0
P_MP4_CACS_End	0x88220AF4

Chrominance AC Huffman table address offset for codeword (TsramMode = 2), 8 bits register.

## 38.4 MPEG/JPEG Engine Limitation

### 38.4.1 JPEG Engine Limitation

Engine	Mode	Description
JPEG engine	Capture	YUV422: Image width: Multiple of 16 (<4080) Image height: Multiple of 8 (<4080)  YUV420: Image width: Multiple of 16 (<4080) Image height: Multiple of 16 (<4080)  Cropping image: Yes  Rotation: Yes Flip: Yes  Scaling down: No Scaling up: Yes (factor 8 bits)
	Playback	YUV422: Image width: Multiple of 16 (<4080) Image height: Multiple of 8 (<4080) YUV420: Image width: Multiple of 16 (<4080) Image height: Multiple of 16 (<4080) YUV411: Image width: Multiple of 16 (<4080) Image height: Multiple of 8 (<4080) YUV444: Image width: Multiple of 16 (<4080) Image height: Multiple of 16 (<4080)  Cropping image: Yes  Rotation: No Flip: No  Scaling down: only 1/8 ~ 7/8 Scaling up: No

#### 38.4.2 MPEG Engine Limitation

Engine	Mode	Description
MPEG engine	Video clip,	<p>Image width: Multiple of 16 (<math>\leq 640</math>) Image height: Multiple of 16 (<math>\leq 480</math>)</p> <p>Cropping image: Yes</p> <p>Rotation: Yes Flip: Yes</p> <p>Scaling down: No Scaling up : Yes (Factor 8 bits)</p>
MPEG engine	Video Playback	<p>Image width: Multiple of 16 (<math>\leq 640</math>) Image height: Multiple of 16 (<math>\leq 480</math>)</p> <p>Cropping image: No</p> <p>Rotation: No Flip: No</p> <p>Scaling down: No Scaling up : No</p>