VT268 Programming Guide V1.0

1.TABLE OF CONTENT

2. REVISION HISTORY

Revision	Date	Remark
1.0	2010.2.11	First edition

3. GENERAL DESCRIPTION

VT268 includes the main CPU, Graphic Processor, Sound CPU, internal SRAM, ROM, and some I/O controllers. There are three main systems in VT268, they are program, sound and video systems.

Main CPU plays the key role in program system. It can access the internal and external program memories(NOR FLASH or SRAM). The program memory stores the program command, instructions, and sound data. VT268 is equipped with 12K Bytes SRAM as internal program memory. This program RAM will be the zero page RAM, STACK and some memory of CPU. Program system controls the operations of whole system, including graphic and the sound.

Graphic Processor is the main role of the video system. It can access the video memory automatically to display some figures. VT268 is equipped the other 4K Bytes SRAM (VRAM) used for the background layer display. Another 1.5K bytes SRAM(SPRAM) are used for the sprite display. There are another 1K bytes SRAM(PaletteRAM) are used to provide up to 512 color palette components display. All graphic is displayed through either the Composited video signal or the LCD panel.

Sound CPU includes an enhanced 6502 CPU and 4K bytes local program SRAM. It has the individual IO and ALU to provide kinds of voice applications.

3.1 Feature

System

- Working Voltage 3.0~3.6 V
- Main CPU: 6502 @10.7368MHz in NTSC and 10.64068MHz in PAL
- Internal optional Data / Boot / Program ROM: 4K Bytes
- Internal Main CPU 12K Bytes RAM, includes 6K bytes dedicated RAM and 6K bytes shared RAM)
- Internal 4K Bytes Shared Video RAM
- Direct Memory Access (DMA) Sprite RAM / Program RAM / External memory up to 10Mbytes/sec.
- Single 16bits external data bus
- Support 4 type of T.V. signal output (NTSC, PAL, PAL-M, PAL-N)
- Sleep mode and Auto wake up mode
- 56 individual controlled GPIO ports
- Cartridge combination function
- External EEPROM boot up function

Peripheral

- ADC: 10bits, 8 Times-Division-Multiplex channels with microphone, and battery detection
- Master/Slave SPI Interface:
- UART duplex Interface up to 115200 bps
- TFT LCD Interface.
- STN LCD Interface
- SD card interface
- Master IIC interface with 16bits addressing mode

- ITU656/601 Video Input Interface
- Enhanced ALU, 16 by 16 multiplier and 32 by 16 divider
- RTC (Real-Time-Clock) with Alarm function
- 3-chanel LED PWM output

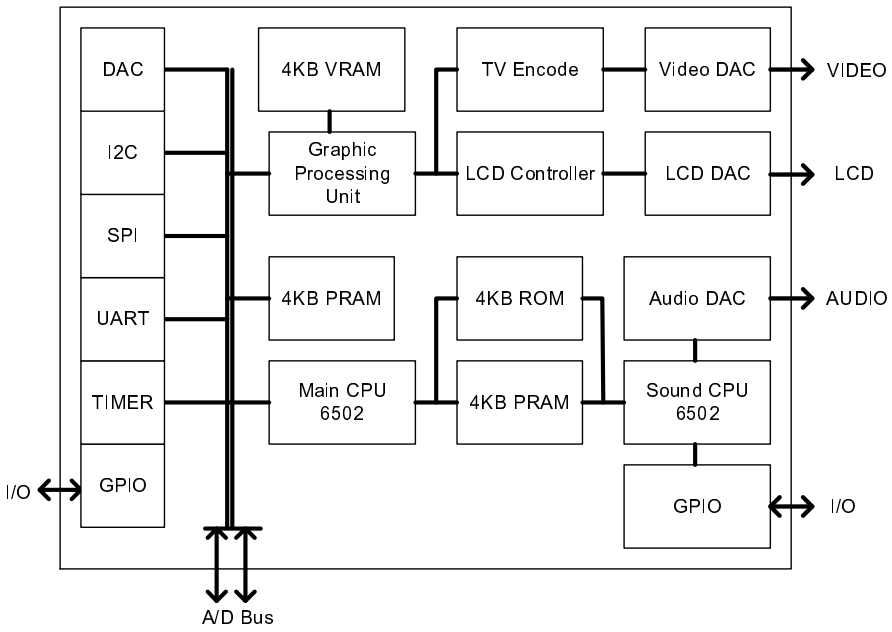
Graphic Processor

- Background resolution: TV 256dots x240 lines
- 3 independent background layers.
- Background 1/2 character mode (8x8 / 16x16): 4/16/64/256 indexed color mode.
- Background 3 character mode (16x16): 4/16 indexed color mode.
- Background 1/2 bitmap mode: 4/16/64/256 indexed color mode
- Background 1 bitmap 32768 colors hi color (direct color) mode
- Background 1/2/3 independent vertical extension: x1/x1.5/x2
- Background 1/2/3 horizontal scan line line-based scrolling: -128~+127
- Sprites are 16/64 colors.
- Sprite character size includes 8x8 / 8x16 / 8x32 / 16x8 / 16x16 / 16x32 / 32x8 / 32x16 / 32x32 (H/V).
- Sprite vertical extension (Global): x1/x1.5/x2
- Max Sprite index number is 65536.
- Sprite resolution 256dots x 240 lines / 512 dots x 240 lines
- 512 Color palette
- Dual graphic Output (TV + LCD or LCD + LCD).
- Dig-Hole, Fade in / fade out, Blend or gray effect
- RGB independent 32 level fade-in fade-out effect

Sound CPU

- CPU 6502 @21.4772MHz in NTSC and 21.28136 MHz in PAL
- 6K bytes dedicated RAM includes 4 K bytes local memory and 2K bytes shared memory
  - Stereo 10bits audio DAC
  - Provides CELP, ADPCM and MIDI driver.

3.2 BLOCK DIAGRAM



3.3 Pin Description

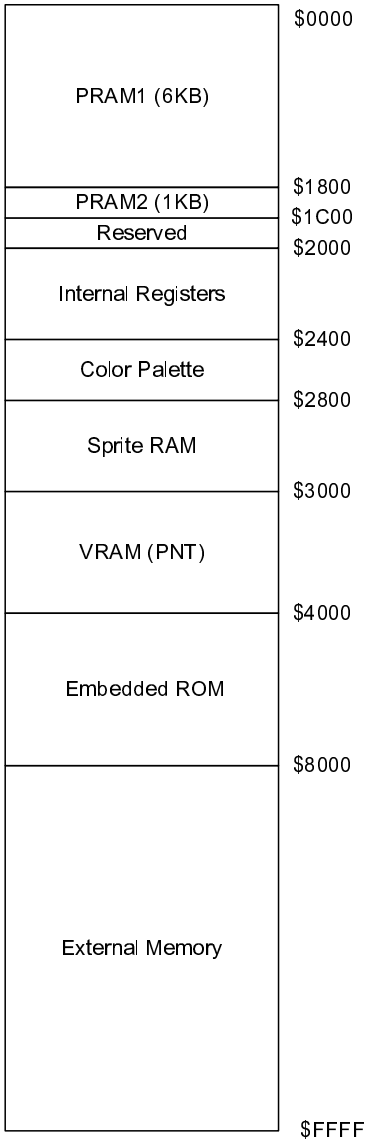
SYMBOL	TYPE	DESCRIPTION
A[21:0]	O	Address bus
D[15:0]	I/O	Data bus
ROMCSB	O	1st external memory CSB signal
RAMCSB	O	2nd external memory CSB signal
ROMOEB	O	External memory OEB signal
RAMRWB	O	External memory RWB signal
IDECK	I	Debug Mode interface
IDEDI	I	Debug Mode interface
IDEDODO	O	Debug Mode interface
IDECSB	I	Debug Mode interface
UIOA[7:0]	I/O	Universal I/O
UIOB[7:0]	I/O	Universal I/O
UIOC[7:0]	I/O	Universal I/O
UIOD[7:0]	I/O	Universal I/O
UIOE[7:0]	I/O	Universal I/O
UIOF[7:0]	I/O	Universal I/O
UIOG[7:0]	I/O	Universal I/O
XTAL1	I	System Crystal 21.4772MHz / 26.6017MHz pin
XTAL2	O	System Crystal 21.4772MHz / 26.6017MHz pin
XTAL3	I	32768 Crystal pin
XTAL4	O	32768 Crystal pin
VIDEO	O	Composite video signal
AUDIOL	O	Right channel audio signal
AUDIOR	O	Left channel audio signal
BOOTINIT	I	Internal ROM Boot up mode
PLLVCO	I/O	PLL reference voltage
TVSEL	I	TV system select

4. MAIN CPU

Main CPU in the VT1682 is an enhanced 8-bits 6502-based CPU operates at 10MHz. Except the 6502 standard instruction set, 90 instructions are provided for the fully linear addressing(8M bytes) with independent instruction and data bank. For the detail operation please reference the documents “VT268 CPU Instruction Set”.

4.1 Memory Map

The partition of the memory map is shown in the following diagram. PRAM1 (Program RAM-1) is 6KB for Main CPU local Program RAM. PRAM2 is 1KB shared by Main CPU and Sound CPU. Address between 0x2000 and 0x20FF is the Graphic ports and 0x2100 and 0x23FF is for the system or peripheral control. $2400~$3FFF are for the graphic memories include 512 color palettes, sprite memory and VRAM. There is a also a 4KB embedded ROM could be the BIOS, security, program ROM or data ROM.



When the new memory access instruction with data bank register(D:) is used, all the addressing area are mapped to the external memory.

4.2 Address Mode

The 6502 CPU in VT268 adds some new instructions to provide the 24 bits linear address mode. For the detail operation, please reference the “VT268 CPU assembly instruction set”.

4.3 CPU Interrupt Service

VT268 provides 6 interrupt(IRQ) and 1 non-maskable interrupt(NMI). These IRQs include external interrupt(EXT_IRQ), timer interrupt(Timer_IRQ), sound processor interrupt(SCPU_IRQ), UART interrupt(UART_IRQ), Real-Timer-Clock or graphic position interrupt(RTC_IRQ) and SPI interrupt(SPI_IRQ). When the interrupt is detected, the service routine address stored in the mapped vector address is executed. When the interrupts occur at the same time, their priority would be EXT_IRQ > TMR_IRQ > SCPU_IRQ > UART_IRQ > SPI_IRQ > RTC_IRQ. Please noted that only when the interrupt mask flag(I_FLAG) in CPU is clear (using CLI instruction), the interrupt, except NMI, would be detected by CPU.

Vector Name	vector address
NMI	0x00FFFA, 0x00FFFB
Ext IRQ	0x00FFFE, 0x00FFFF
Timer IRQ	0x00FFF8, 0x00FFF9
SCPU IRQ	0x00FFF6, 0x00FFF7
UART IRQ	0x00FFF4, 0x00FFF5
SPI IRQ	0x00FFF2, 0x00FFF3
RTC IRQ	0x00FFF0, 0x00FFF1

4.3.1 Non-Maskable Interrupt

The NMI in VT268 is always the same as graphic vertical blanking period. It’s 60Hz in NTSC system and 50HZ is PAL system. It’s unable to be masked by the CPU i-flag but it could be disable through the register NMIEN in P_CONTROL1.

NMIEN	NMI
0	Disabled
1	Enabled, and synchronized with the graphic vertical blanking period.

When the NMI occurs, the PC(program counter) would jump to the address stored in vector address 0x00FFFA and 0x00FFFB.And the 8MSB A24..A16 would be all zero. For example, when the NMI service routine is located in 0x001234, then the data stored in the vector address [0x00FFFA] = $34, [0x00FFFB] = $12. Please note that the 8-MSB are zero.

NMI initialization :

```
    LDA     P_CONTROL1
    ORA     #$01                ; Enable NMI
  STA    P_CONTROL1
```

4.3.2 External interrupt

When EXTIRQEN in C_EXT_IRQ is enabled, the external interrupt would occur when the UIOG0 input signal transition from high to low(falling edge). When the external interrupt occurs, the PC(program counter) would jump to the address stored in vector address 0x00FFFE and 0x00FFFF.And the 8MSB A24..A16 would be all zero.

EXTIRQEN	External interrupt
0	Disabled
1	Enabled at the UIOG0 falling edge

External interrupt initialization sub-routine:

```
LDA  C_UIOG_DIR      ;   Set UIOG0 to input mode
AND  #$FE
STA  C_UIOG_DIR
LDA  #$3             ; ENABLE EXT_IRQ
STA  C_EXT_IRQ
CLI                  ; Clear i-Flag in CPU
```

External interrupt service sub-routine:

```
LDA  C_EXT_IRQ
ORA  #$2
STA  C_EXT_IRQ           ; Clear EXT_IRQ status flag
```

4.3.3 Timer interrupt

Timer interrupt is the periodical interrupt that generated by the timer. As to the detail operation of the timer, please reference the section “Timer” in the later chapter. When the timer interrupt occurs, the PC(program counter) would jump to the address stored in vector address 0x00FFF8 and 0x00FFF9.And the 8MSB A24..A16 would be all zero.

Timer interrupt initialization sub-routine:

```
LDA  #TIMER_VALUE_L      ; Set timer period
STA  C_TMR_L
LDA  #TIMER_VALUE_H      ; Set timer period
STA  C_TMR_H
STA  C_TMR_CLR
LDA  #$07(or #$03)       ; Enable Timer and enable interrupt
STA  C_TMR_SETTING
CLI                      ; Clear i-Flag in CPU
```

Timer interrupt service sub-routine:

```
STA  C_TMR_CLR           ; Clear timer interrupt status flag
```

4.3.4 UART interrupt

UART interrupt occurs when the UART module received or transmit data. As to the detail operation of the UART, please reference the section “UART Interface” in the later chapter. When the UART interrupt occurs, the PC(program counter) would jump to the address stored in vector address 0x00FFF4 and 0x00FFF5.And the 8MSB A24..A16 would be all zero.

4.3.5 SP interrupt

SPI interrupt occurs when the SPI module received or transmit data. As to the detail operation of the SPI, please reference the section “SPI Interface” in the later chapter. When the SPI interrupt occurs, the PC(program counter) would jump to the address stored in vector address 0x00FFF2 and 0x00FFF3.And the 8MSB A24..A16 would be all zero.

4.3.6 Rea Time Clock interrupt

There are two real timer clock interrupt source, one is the 32768HZ-based timer, and the other us the alarm interrupt. When WAKEIRQEN is enabled, the 32768Hz-based counter would periodical generate interrupt. When ALMIRQEN is enabled, the interrupt would occur when the RTC time(RTC_HOUR, RTC_MINUTE, RTC_SECOND) is the same as the ALARM time(ALARM_HOUR, ALARM_MINUTE, ALARM_SECOND). When the interrupt occurs, the PC(program counter) would jump to the address stored in vector address 0x00FFF0 and 0x00FFF1.And the 8MSB A24..A16 would be all zero. This interrupt is shared with the graphic position interrupt. In other word, either real time clock interrupt or graphic position interrupt would use the service routine defined by 0x00FFF0 and 0x00FFF1. Reading the ALMSTAT and WAKESTAT would help to identify the interrupt source, even when the WAKEIRQEN or ALMIRQEN is disabled.

Status Flag	Status Description
ALMSTAT = 0	Alarm not occurs
ALMSTAT = 1	Alarm occurs
WAKESTAT = 0	Wakeup timer not occurs
WAKESTAT = 1	Wakeup timer occurs

32768Hz-based Timer interrupt initialization sub-routine:

```
LDA     #CNT_PERIOD
STA     C_WAKE_PERD       ; Set timer period
LDA     #$01
STA     C_KEY_WAKE_MD     ; Set clock source
LDA     C_RTL_IRQ
ORA     #$4
STA     C_RTL_IRQ         ; clear RTC IRQ status
ORA     #$1
```

```
STA     C_RTL_IRQ       ; Enable interrupt
CLI
    :
```

32768Hz-based Timer interrupt service sub-routine:

```
LDA     C_RTL_IRQ       ; clear RTC IRQ status
STA     C_RTL_IRQ
```

Alarm interrupt initialization sub-routine:

```
LDA     #ALARM_HOUR
STA     C_ALARM_HR      ; Set ALARM HOUR
LDA     # ALARM_MINUTE
STA     C_ALARM_MN      ; Set ALARM MINUTE
LDA     # ALARM_SECOND
STA     C_ALARM_SC      ; Set ALARM SECOND
LDA     C_RTL_IRQ
ORA     #$8
STA     C_RTL_IRQ       ; clear IRQ status
ORA     #$8
STA     C_RTL_IRQ       ; Enable interrupt
    :
```

Alarm interrupt service sub-routine:

```
LDA     C_RTL_IRQ       ; clear IRQ status
STA     C_RTL_IRQ
```

4.3.7 Graphic position interrupt

When the composited video signal or the LCD display stream arrive some pre-defined location, the graphic position interrupt would occur. When the interrupt occurs, the PC(program counter) would jump to the address stored in vector address 0x00FFF0 and 0x00FFF1.And the 8MSB A24..A16 would be all zero. The graphic resolution of the VT268 is 256x240. If XYIRQEN is enable, the interrupt would occurs when the display stream arrive the horizontal position P_XY_IRQ_H and the vertical position P_XY_IRQ_V.

Graphic position interrupt initialization sub-routine:

```
LDA     #XYIRQ_X_VALUE      ; Set horizontal position
STA     P_XY_IRQ_H
LDA     #XYIRQ_Y_VALUE      ; Set vertical position
STA     P_XY_IRQ_V
LDA     P_XY_IRQ
ORA     #$2
```

```
STA     P_XY_IRQ                ; Clear IRQ status
ORA     #$1
STA     P_XY_IRQ                ; Enable IRQ
CLI
  :
```

Graphic position interrupt service routine:

```
LDA     P_XY_IRQ                    ; Clear IRQ status
STA     P_XY_IRQ
LDA     #XYIRQ_NEXT _Y_VALUE        ; Set the next vertical position
STA     P_XY_IRQ_V
LDA     #XYIRQ_NEXT_X_VALUE         ; Set the next horizontal position
STA     P_XY_IRQ_H
```

5. Peripheral and Interface

Interfaces in VT268 include TV composite output, LCD, UART, SPI, IIC, SD and CCIR interface.

5.1 TV System Configuration

VT268 supports 4 types of TV system, NTSC, PAL, PAL-M and PAL-N, selected the two pins TVSEL0 and TVSEL1. Please note that the crystal should be changed to the related frequency.

	TVSEL0	TVSEL1	Crystal Frequency
NTSC	0	0	21.4772MHz
PAL-M	1	0	21.453669MHz
PAL-N	0	1	21.492336MHz
PAL	1	1	26.601712MHz

5.2 LCD Interface

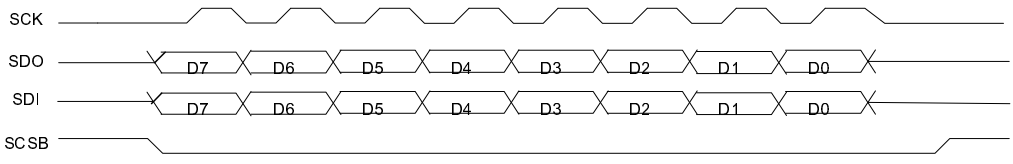
VT268 provides two sets of LCD output interface. The first set includes kinds of LCD interface, includes serial RGB, parallel RGB, analog TFT LCD, 80x80 and STN interface. While the second set provides only the serial RGB and STN interface. These LCD pins would use UIOA, UIOB, UIOC and UIOD shared with GPIO. The detail pin map please reference the later chapter “Shared IO”. The LCD control register is between 0x2040 ~ 0x204F, and please contact the VRT FAE for the setting procedure.

5.3 CCIR protocol

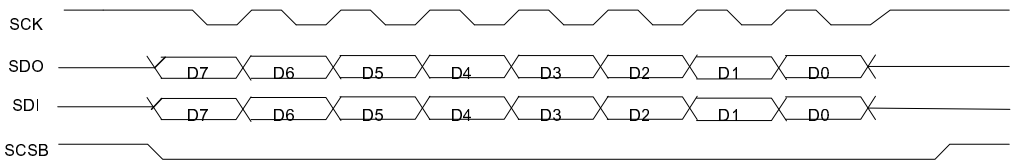
VT268 provides kinds of CCIR656 / 601 formats, include YUV422, RGB422, RGB555 and RGB565 format. These CCIR pins would use UIOE and UIOF shared with GPIO. The detail pin map please reference the later chapter “Shared IO”. The CCIR control register is between 0x2028 ~ 0x202B, and please contact the VRT FAE for the setting procedure.

5.4 SPI interface

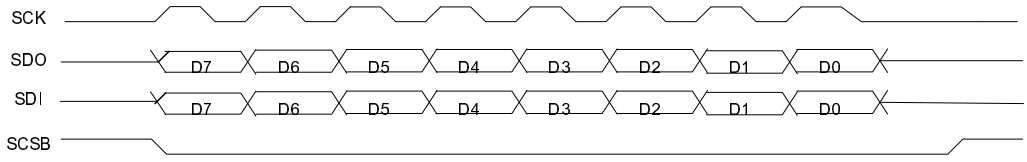
VT268 provide masker and slave mode SPI(Standard peripheral Interface). There are four transmission mode in master mode. Either master or slave mode, four type of communication protocols are valid as shown in the flowing diagram. SPI interface would use the pins ,UIOA0, UIOA1, UIOA2 and UIOA3.



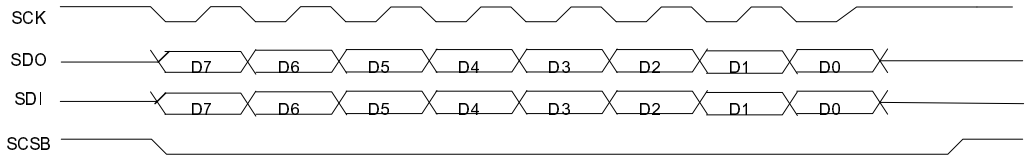
POLAR=0, PHASE =0



POLAR=1, PHASE =0



POLAR=0, PHASE =1



POLAR=1, PHASE =1

Data transmission rate

CLKSEL	Transmission rate (bit per second)
0	5 MHz
1	2.5 MHz
2	1.25 MHz
3	639 KHz

Master/Slave mode select

MASTER = 0	Slave mode
MASTER = 1	Master mode

8/16 bits mode select in master mode

D16Mode = 0	8bitsmode
D16Mode = 1	16 bitsmode

Interrupt control

IRQEN = 0	SPI IRQ disabled
IRQEN = 1	SPI IRQ enabled

SPI module reset control

RSTSPI = 0	Normal operation
RSTSPI = 1	SPI module reset

SPI RX DMA control

RxDmaEn = 0	SPI RX DMA disabled
RxDmaEn = 1	SPI RX DMA enabled

SPI TXDMA control

TxDmaEn = 0	SPI TX DMA disabled
TxDmaEn = 1	SPI TX DMA enabled

SPI status

READY = 0	SPI busy
READY = 1	Ready for next command
NewRec = 0	No new-received data
NewRec = 1	New data is received

Example : SPI application procedure

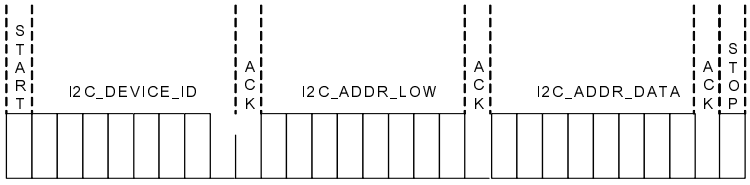
```
LDA     C_UIOA_DIR              ; Set UIOA0, UIOA1and UIOA3 to output mode
ORA     #$0B
AND     #$04                    ; Set UIOA2 to input mode
STA     C_UIOA_DIR
LDA     C_UIOA_ENB
ORA     #$0B
STA     C_UIOA_ENB              ; Set UIOA0..3 to SPI pins
LDA     SPI_CONTROL_VALUE
STA     C_SPI_SETTING           ; Set SPI operation mode
LDA     SPI_DATA_VALUE
STA     C_SPI_DATA              ; Transmit data……
```

5.5 I2C interface

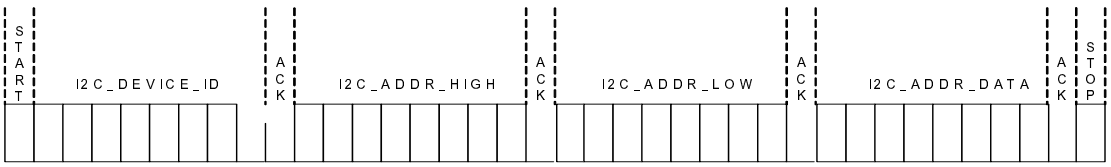
VT268 provides master mode I2C access protocol. Not only the standard I2c access mode, but also the 16-bits I2C access mode for the serail flash with higher density. I2C interface shares the UIOE2 and UIOE3. Four types of access mode are valid as shown in the following table and diagram.

I2C _ID[0]	D16Mode	I2C operation mode
0	0	8bits address write
0	1	16bits address write
1	0	8bits address read
1	1	16bits address read

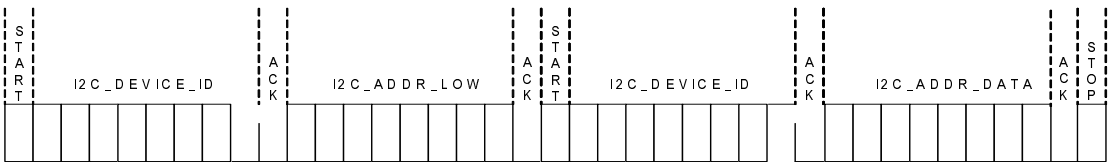
8bits address write timing diagram:



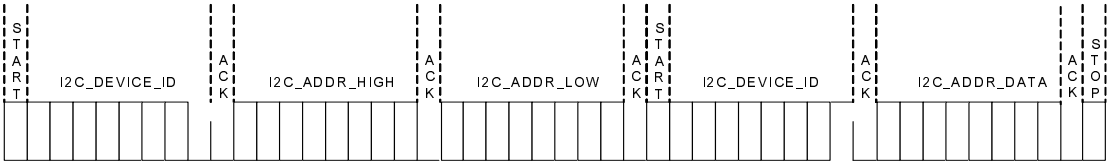
16bits address write timing diagram:



8bits address read timing diagram:



16bits address read timing diagram:



I2C data transmission rate:

CLKSEL	transmission rate (bit/sec)
0	2.5MHz
1	625KHz
2	312.5KHz
3	78.125KHz

I2C module operation status:

READY	Operation status
0	Data is transmitting
1	Data transmission complete

I2C communication status

ACKERR	Communication status
0	Normal communication
1	No response

I2C operation example:

```
LDA     C_UIOE_DIR              ; Set UIOE2 to output mode
ORA     #$04                    ; Set UIOE3 to input mode
STA     C_UIOE_DIR
LDA     C_UIOE_ENB
ORA     #$04
STA     C_UIOE_ENB              ;
LDA     I2C_DEVICE_ID_VALUE
STA     C_I2C_ID                ; Set I2C device ID and read/write select
LDA     I2C_ADDR_OW_VALUE
STA     C_I2C_ADDR_L            ; Set I2C address
LDA     I2C_CONTROL_VALUE
STA     C_I2C_CONTROL           ; Set I2C operation mode
LDA     I2C_DATA_VALUE
STA     C_I2C_W_DATA            ; Transmit data
```

5.6 SD card interface

VT268 provides the SD interface to access SD card through the SPI mode in SD card. It allows VT268 to send different command and access data on SD card. The related driver please contact with VRT FAE. There are nine SD card commands are provided in VT268, they are

CMD0: GO_IDLE_STATE

CMD9: SEND_CSD

CMD10: SEND_CID

CMD13: SEND_STATUS

CMD16: SET_BLOCKLEN

CMD17: READ_SINGLE_BLOCK

CMD24: WRITE_SINGLE_BLOCK

CMD55: APP_CMD

ACMD41: SEND_OP_COND

5.7 InfraRed remote controller

VT268 provides the learning mode InfraRed remote controller function. It could memory the remote controller signal and store it in internal RAM or external memory device, and playback. The related driver please contacts with the VRT FAE.

5.7.1 Receive and Storage

When the bottom of the remote controller is pressed, some particular infra-red stream is emitted. VT268 could receive it through the IR receiver, compress and store it in memory. The received data sample time is programmable through the register. Sampling rate is inverse proportion to the data size. Larger sample would possiblly result in the data loss at playback period. When VT268 get the learning command, it would automatically wait for the infra-red signal and then start to memory to save the memory space. There are three ways to end the learnng(memory) procedure. The first is the IR sigal emittion complete. The second is memory full(1024 bytes per storage). The third is break by software. When the IR signal is received, VT268 would automatically compress the data and store it to the software specified memory area

5.8 UART interface

VT268 provides up to 11520 bit-per-second full-time duplex transceiver. UART transmitter (TX) uses the UIOE0 and receiver in UIOE1. Individual receive/transmit IRQ, DMA access mode and parity check are also valid.

UART Enable control

UARTEN = 0	UART disabled
UARTEN = 1	UART enabled

UART interrupt setting

TXIRQEN = 0	UART TX IRQ disabled
TXIRQEN = 1	UART TX IRQ enable
RXIRQEN = 0	UART RX IRQ disabled
RXIRQEN = 1	UART RX IRQ enable

UART RX DMA setting

RxDmaEn = 0	UART RX DMA disabled
RxDmaEn = 1	UART RX DMA enabled

UART TX DMA setting

TxDmaEn = 0	UART TX DMA disabled
TxDmaEn = 1	UART TX DMA enabled

UART staus

TXStatus = 0	Data is transmitting
TXStatus = 1	Transmission complete
RXStatus = 0	Data is receiving
RXStatus = 1	Receive complete
ParityErr = 0	Parity check is correct
ParityErr = 1	Parity check is fail
RxError = 0	Baud rate match
RxError = 1	Baud rate mis-match

Baud rate table

	NTSC		PAL	
Baud Rate (bps)	Baud_rate_H	Baud_rate_L	Baud_rate_H	Baud_rate_L
115200	$02	$BF	$02	$C6
57600	$01	$60	$01	$63
38400	$00	$EA	$00	$ED
19200	$00	$75	$00	$76
9600	$00	$3A	$00	$3B
4800	$00	$1D	$00	$1E
2400	$00	$0F	$00	$0F

UART initialization procedure (TX is polling mode, and RX is interrupt mode)

```
LDA  #$BF                 // Set Baud rate
STA  C_UART_BAUD_L
LDA  #$02
STA  C_UART_BAUD_H
LDA  #$88
STA  C_UART_IRQ_CLR       // Clear IRQ flag and enable UART
LDA  #$40                 // Set UART operation mode
STA  $2118
LDA  C_UIOE_DIR           // Set UIOE0 and UIOE1
ORA  #$01                 // UIOE0 is output mode 輸出
```

```
        AND  #$FD                   // UIOE1 is input mode
        STA  C_UIOE_DIR
        LDA  C_UIOE_ENB
        ORA  #$01
        STA  C_UIOE_ENB
        CLI

;==== UART Write =====================
UART_WRITE:
        TAX                         // Register A is the data to be transmitted
L_wait_uart_tx:
        LDA  C_UART_STATUS
        AND  #$02
        BEQ  L_wait_uart_tx
        STX  C_UART_TX_DATA
        RTS

====== UART RX interrupt service routine ====================
UART_IRQ:
        PHA
        PHX
        PHY
        LDA  #$88
        STA  C_UART_IRQ_CLR     ; CLEAR RX IRQ
        LDA  C_UART_RX_DATA
        STA  r_UART_rx_buffer   ; RX buffer
        PLY
        PLX
        PLA
        RTI
```

6. DMA

VT268 provides high speed Direct Memory Access (DMA) data transfer rate up to 10M bytes per second. There are no bank limitation in DMA transfer operation. DMA source data could be located in any address area. They are two ways to issue the DMA function, one is the direct access that is control the DMA operation through $2122~$2128 directly. The other is in-direct access that stores all data in the RAM area, the hardware would automatically upload the data from RAM when the DMA is trigger, .

These two DMA methodology is suitable for different applications. The direct mode uses the shorter CPU instruction but the 0x2122~0x2128 would change after the trigger of DMA. So it's suitable to transport the

larger amount of data. The indirect mode would not change the value in RAM suits for the repeat or routine data transport.

Example:

While we are going to transport 32 bytes data from address $345678 to address $ABCDEF,

a. Use the direct mode

C_DMA_DT_AL = $EF

C_DMA_DT_AM= $CD

C_DMA_SR_AL = $78

C_DMA_SR_AM = $56

C_DMA_SR_AH = $34

C_DMA_DT_AH = $AB

C_DMA_NUM = 16 …….. 32 bytes = 16 Word, write C_DMA_NUM to trigger DMA

a. Use the indirect mode, and use the memory area 0x0250 ~0x0255 as buffer

0x0250 = $78..... The first address should be an even

0x0251 = $56

0x0252 = $EF

0x0253 = $CD

0x0254 = $34

0x0255 = $AB

then

C_DMA_LUT_AL = $50

C_DMA_LUT_AH = $02

C_DMA_LUT_NUM = 16 …….. 32 bytes = 16 Word, write C_DMA_NUM to trigger DMA

DMA address mapping area :

0x000000 ~ 0x007FFF……. internal memory or register

0x008000 ~ 0xFFFFFF…… external memory

7. GRAPHIC processor

7.1 Feature

- Resolution: TV 256x240 pixels
- 3 independent background layers and 256 sprites.
- Background 1/2 character mode (8x8 / 16x16): 4/16/64/256 indexed color mode.
- Background 3 character mode (16x16): 4/16 indexed color mode.
- Background 1/2 bitmap mode: 4/16/64/256 indexed color mode
- Background 1 bitmap 32768 colors direct color mode
- Background 1/2/3 individual vertical extension: x1/x1.5/x2
- Background 1/2/3 horizontal scan line line-based scrolling: -128~+127
- Sprites individual color mode 16/64 colors.
- Sprite individual character size includes 8x8 / 8x16 / 8x32 / 16x8 / 16x16 / 16x32 / 32x8 / 32x16 / 32x32 (H/V).
- Sprite global vertical extension (Global): x1/x1.5/x2
- 512 Color palette, maximum display indexed color: 512
- Dual graphic Output (TV + LCD or LCD + LCD).
- Dig-Hole, Fade in / fade out, Blend or gray effect
- RGB independent 32 level fade-in fade-out effect
- Background 1/2/3 and sprite vertical scaling

7.2 Screen Structure

The display screen (displayed on the TV screen) resolution of the Graphic in VT268 is 256 x 240 pixels.

7.2.1 VRAM Screen Structure

To display a background on the TV screen, the pattern(PNT) in the VRAM (0x003000 ~ 0x003FFF) should be defined first. There are two kinds of screen structure in VRAM, they are character and bitmap mode. Each PNT include 12bits character vector and 4-bits palette bank. When PNT=0, it would be a transparent character or line.

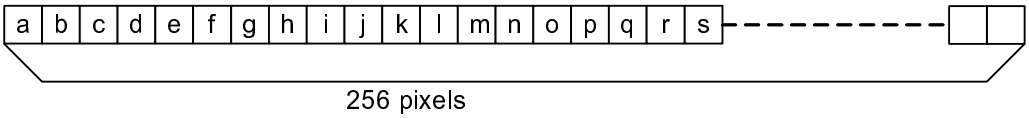
LowByte	Vector[7:0]	
HighByte	PalBank[3:0]	Vector[11:8]

7.2.2 Graphic data format

To display a background or sprite, graphic processor would decode the PNT and fetch the graphic pattern (PGT) from the external memory, and then translate by the Color Palette. PGT format are different by its color mode. Following chapter would illustrate the format of the PGT data.

7.2.2.1 Linear mode

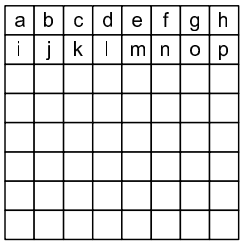
The graphic pattern is defined line by line whose sequence is as shown in the following diagram.



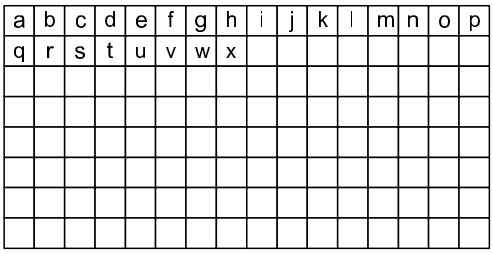
7.2.2.2 Character mode

Assume the display sequence is a,b,c,….h, then

8x8(HxV) / 8x16 / 8x32 character mode

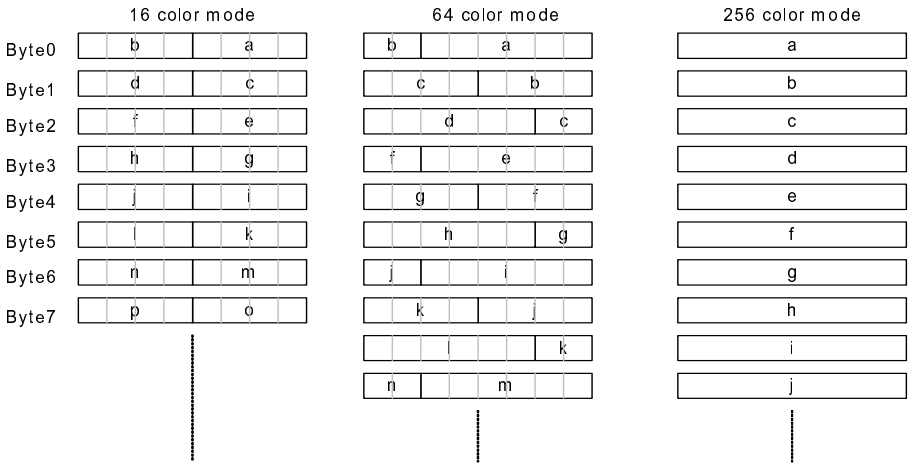


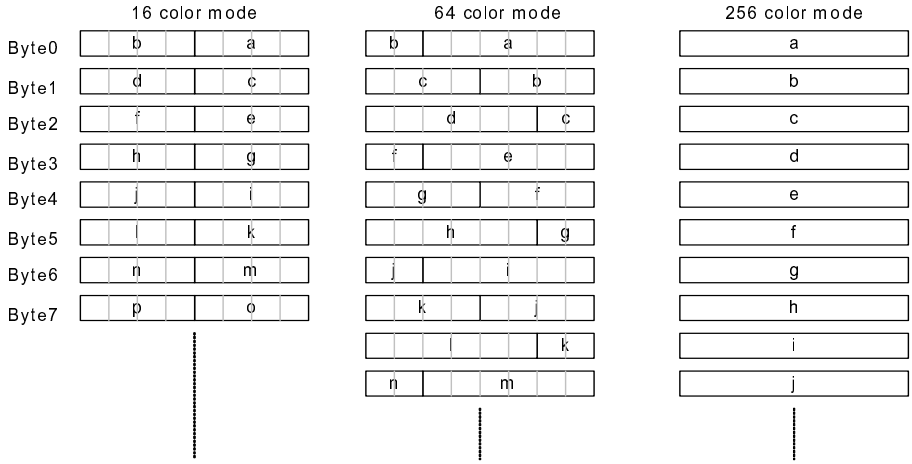
16x8(HxV) / 16x16 / 16x32 character mode



7.2.2.3 Color Mode

Graphic pattern with different color mode would have different format as shown in the following diagram.





In high color mode(32768 color mode), each pixel is defined by two bytes, that is

Low byte

D7 – D5	D4 – D0
Green[2:0]	Blue[4:0]

High byte

D15	D14 – D10	D9 – D8
TRPT	Red[4:0]	Green[4:3]

Each pixel contains 5bitsR, 5bits G and 5-bits B. The MSB, TRPT, is the transparent flag. When TRPT=1, the pixel would become transparent.

7.2.3 Color Palette Format

Color Palette is used to translate PGT indexed number into the real RGB color domain. There are 512 components in Color Palette. And each component includes 5bits Red, 5bits Green, 5-bits Blue and 1-bit EF flag which is used for the special effect.

Low byte

D7 – D5	D4 – D0
Green[2:0]	Blue[4:0]

High byte

D7	D6 – D2	D1 – D0
EF	Red[4:0]	Green[4:3]

7.3 Background Layer 1(BK1)

The control registers of background layer1(BK1) are listed in the following table.

Address	Symbol	R/W	default
0x002010	P_BK1_X	R/W	0
0x002011	P_BK1_Y	R/W	0
0x002012	P_BK1_SETTING1	R/W	0
0x002013	P_BK1_SETTING2	R/W	0

0x00201C	P_BK1_SEG_L	R/W	0
0x00201D	P_BK1_SEG_H	R/W	0

7.3.1 Enable BK1

When BK1EN is set to 1, the BK1 would be displayed according to its parameters.

7.3.2 Coordinate

The X-axis and Y-axis of the background layer coordinate are between -256 and 255, represented by 2'scomplment. When X is positive(BK1_X[8] = 0), the display background would scroll left with X pixels. Otherwise it scrolls right. When Y is positive(BK1_Y[8] = 0), the display background would scroll up with Y pixels. Otherwise it scrolls down.

7.3.3 Scrolling Page

BK1 provides 4 type of scrolling mode, as shown in the following table. They are single page mode, horizontal two page mode, vertical two page mode and four page mode..

	BK1HSCRL = 0	BK1HSCRL = 1
BK1VSCRL = 0	single page mode	horizontal two page mode
BK1VSCRL = 1	vertical two page mode	four page mode

In different mode, the effective X, Y coordinate are different.

Scrolling mode	BK1VSCRL	BK1HSCRL	Effective X	Effective Y
single page mode	0	0	0 ~ 255	0 ~ 255
horizontal two page mode	0	1	0 ~ 511	0 ~ 255
vertical two page mode	1	0	0 ~ 255	0 ~ 511
four page mode	1	1	0 ~ 511	0 ~ 511

7.3.4 Character mode

When BK1LINE = 0, BK1 operates in character mode. The size of each character could be either 8x8 or 16x16. Only four color mode 4, 16, 64, 256are valid in BK1. High color mode is invalid.

	Character size
BK1SIZE= 0	8x8
BK1SIZE = 1	16x16.

7.3.5 Linear mode

When BK1LINE = 1, BK1 operates in linear mode. In this mode the BK1SIZE should be set to 1. Both indexed color and high color mode are valid in linear mode.

7.3.5.1 Indexed color mode

In this mode, following register should be set as BK1LINE = 1, BK1SIZE = 1, BK1HC = 0, and color palette is necessary. Four kinds of color modes are valid, they are 4, 16 , 64 and 256 colors.

7.3.5.2 High color mode

In this mode, following register should be set as BK1LINE = 1, BK1SIZE = 1, BK1HC = 1. BK1COLOR = 3 and color palette is not necessary. And only 32768 color mode is valid.

7.3.6 Indexed color mode

Four kinds of color modes are valid in BK1, they are 4, 16 , 64 and 256 colors.

Indexed Color Mode	BK1COLOR
4	0
16	1
64	2
256	3

And It's Color Palette mapping address is

Indexed Color Mode	Color Palette address ($002400 ~ $0027FF)								
	A9	A8	A7	A6	A5	A4	A3	A2	A1
4	BK1PALSEL	PALBANK[3:0]				0	0	COLORDATA	
16	BK1PALSEL	PALBANK[3:0]				COLORDATA			
64	BK1PALSEL	PALBANK[3:2]		COLORDATA					
256	BK1PALSEL	COLORDATA							

7.3.7 Graphic display depth

BK1 provides 4 display depth, the top is 0 and the button is 3. When it has the same depth with BK2 pr BK3, their display priority would be BK1 > BK2 > BK3.

Depth	BK1DEPTH
Top	0
|	1
|	2
Button	3

7.3.8 VRAM memory space management

Before enable the BK1, the related PNT should be filled in to the mapped VRAM area. The mapped VRAM area is adjustable through the register BK1VRAMBANK.

7.3.8.1 16x16 character mode

Background single page mode (256x256),

BK1VRAMBANK	Mapped VRAM area
0	0x3000 ~ 0x31FF
1	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF

3	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF
5	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF
7	0x3E00 ~ 0x3FFF

Background horizontal two page mode(512x256),

BK1VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK1_X[8] = 0, and page1 is for BK1_X[8] = 1

Background vertical two page mode (256x512),

BK1VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK1_Y[8] = 0, and page1 is for BK1_Y[8] = 1

Background four page mode (512x512),

BK1VRAMBANK	Page0 mapped area	Page1 mapped area	Page2 mapped area	Page3 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
2	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF

4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
6	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK1_X[8] = 0, BK1_Y[8] = 0, page1 is for BK1_X[8] = 1, BK1_Y[8] = 0,

Page2 is for BK1_X[8] = 0, BK1_Y[8] = 1, page3 is for BK1_X[8] = 1, BK1_Y[8] = 1,

7.3.8.2 8x8 character mode

Background single page mode (256x256),

BK1VRAMBANK	mapped area
0	0x3000 ~ 0x37FF
1	0x3000 ~ 0x37FF
2	0x3000 ~ 0x37FF
3	0x3000 ~ 0x37FF
4	0x3800 ~ 0x3FFF
5	0x3800 ~ 0x3FFF
6	0x3800 ~ 0x3FFF
7	0x3800 ~ 0x3FFF

Background horizontal two page mode (512x256),

BK1VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
1	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
2	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
3	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
4	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
5	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
6	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
7	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF

Page0 is for BK1_X[8] = 0, and page1 is for BK1_X[8] = 1

Background vertical two page mode (256x512),

BK1VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
1	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
2	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
3	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF

4	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
5	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
6	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
7	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF

Page0 is for BK1_Y[8] = 0, and page1 is for BK1_Y[8] = 1

7.3.8.3 Linear Mode

Background single page mode (256x256),

BK1VRAMBANK	mapped area
0	0x3000 ~ 0x31FF
1	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF
3	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF
5	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF
7	0x3E00 ~ 0x3FFF

Background horizontal two page mode (512x256),

BK1VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK1_X[8] = 0, and page1 is for BK1_X[8] = 1

Background vertical two page mode (256x512),

BK1VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF

5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK1_Y[8] = 0, and page1 is for BK1_Y[8] = 1

Background four page mode (512x512),

BK1VRAMBANK	Page0 mapped area	Page1 mapped area	Page2 mapped area	Page3 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
2	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
6	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK1_X[8] = 0, BK1_Y[8] = 0, page1 is for BK1_X[8] = 1, BK1_Y[8] = 0,

Page2 is for BK1_X[8] = 0, BK1_Y[8] = 1, page3 is for BK1_X[8] = 1, BK1_Y[8] = 1,

7.3.9 Graphic Pattern Address

Graphic PGT address is calculated from the PNT in VRAM, PGT start address SEGMENT, character size and color mode.

Physical PGT address(Bytes) = Start address + Offset

Start address = (BK1SEGMENT << 13).

Offset = (PNT x character horizontal size(pixel) x character vertical size(lines) x bit color mode) / 8

character horizontal size could be 8 or 16,

character vertical size could be 8 or 16

bit color mode :

Color mode	Bit color mode
4	2
16	4
64	6
256	8

For example:

Assume the PGT is located from address 0x20000,

Then BK1SEGMENT = (0x20000) << 13 = 0x10

If BK1 character size is 8x8, 256 color, then the PGT whose PNT=3 would be located at

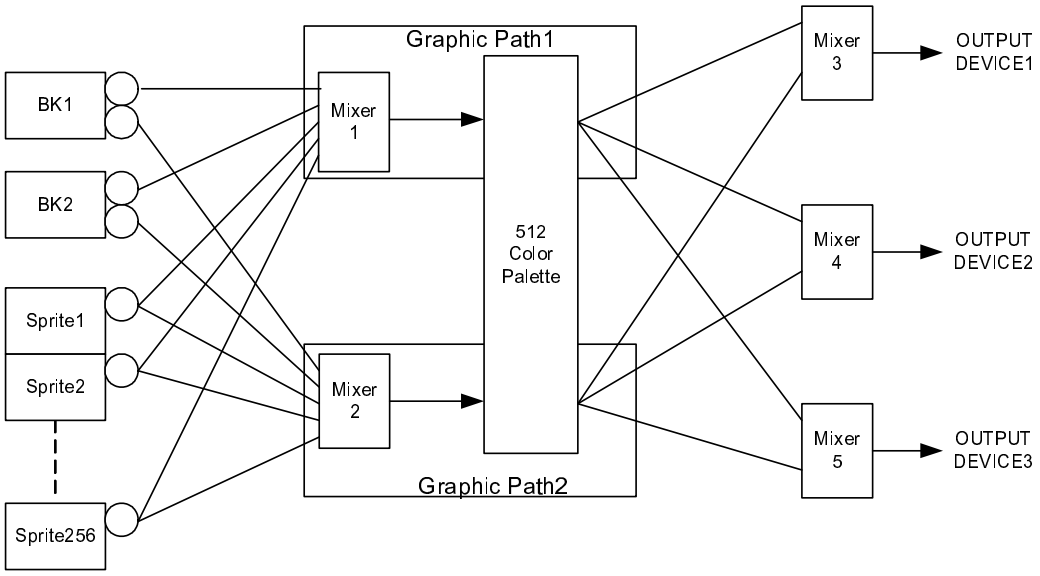
Offset = 3 x 8 x 8 x 8 = 0xC0

Physical PGT address = Start address + Offset = 0x200C0

7.3.10 Graphic path selection

VT268 provides two graphic data path as shown in the following diagram. BK1could choose one or both of them. Both path would share the same 512 color palette.

	Graphic Path1	Graphic path 2
BK1	BK1OutSel[0] = 1	BK1OutSel[1] = 1



7.3.11 Vertical scaling function

BK1 provides the vertical scaling function whose scaling ration is between 1/16 and 16 times separated into 256 levels. Please note that all BK1, BK2, BK3 and sprite use the same scaling ration whose vertical scaling function is enabled.

BK1VGain = 0	BK1VGain = 1
No Scaling function. (Scaling ration = 1)	The scaling would be started form the line VGAINSTARTLINE. All lines before VGAINSTARTLINE would be transparent. The scaling ration of the VGAINSTARTLINE line would be VGAININITIAL, then the ration of the following line would be increased 1/VGAINSTEP line by line.

7.3.12 Horizontal line scrolling function

BK1 horizontal line scaling function allows each horizontal line has an individual horizontal scrolling value, which is defined in a specified RAM area. This specified memory area address is started from { Scroll_Bank+1, 8'b0000_0000}, and each horizontal line maps to one byte whose scrolling value should be between –128 and +127, represented by 2's complement. Please note that this memory area is shared with BK1, BK2 and BK3 whose horizontal line scrolling function is enabled.

BK1LNEN = 0	BK1LNEN = 1
All lines' horizontal scrolling value is BK1_X	N^{th} horizontal scrolling value would be 爲 BK1_X + [Scroll_Bank+1, N]

7.3.13 Horizontal Scaling Function

Horizontal scaling function allows each horizontal line has the individual scaling ration, which is defined in a specified RAM area. This specified memory area address is started from { Scroll_Bank, 8'b0000_0000}, and each horizontal line maps to one byte whose scrolling value should be between 8 and 255 whose scaling ration would be mapped to 8/64 and255/64. Please note that when this function is enabled, all graphic uses the graphic path2 would be scaled.

HSCALEN = 0	HSCALEN = 1
Graphic on Graphic path2 is out of scaling	The scale ration of the N^{th} on Graphic path2 is {Scroll_Bank, N} / 64

7.4 Background Layer 2(BK2)

The control registers of background layer2(BK2) are listed in the following table.

Address	Symbol	R/W	default
0x002014	P_BK2_X	R/W	0
0x002015	P_BK2_Y	R/W	0
0x002016	P_BK2_SETTING1	R/W	0
0x002017	P_BK2_SETTING2	R/W	0
0x00201E	P_BK2_SEG_L	R/W	0
0x00201F	P_BK2_SEG_H	R/W	0

7.4.1 Enable BK2

When BK2EN is set to 1, the BK2 would be displayed according to its parameters.

7.4.2 Coordinate

The X-axis and Y-axis of the background layer coordinate are between -256 and 255, represented by 2'scomplment. When X is positive(BK2_X[8] = 0), the display background would scroll left with X pixels. Otherwise it scrolls right. When Y is positive(BK2_Y[8] = 0), the display background would scroll up with Y pixels. Otherwise it scrolls down.

7.4.3 Scrolling Page

BK2 provides 4 type of scrolling mode, as shown in the following table. They are single page mode, horizontal two page mode, vertical two page mode and four page mode..

	BK2HSCRL = 0	BK2HSCRL = 1

BK2VSCRL = 0	single page mode	horizontal two page mode
BK2VSCRL = 1	vertical two page mode	four page mode

In different mode, the effective X, Y coordinate are different.

Scrolling mode	BK2VSCRL	BK2HSCRL	Effective X	Effective Y
single page mode	0	0	0 ~ 255	0 ~ 255
horizontal two page mode	0	1	0 ~ 511	0 ~ 255
vertical two page mode	1	0	0 ~ 255	0 ~ 511
four page mode	1	1	0 ~ 511	0 ~ 511

7.4.4 Character mode

When BK2LINE = 0, BK2 operates in character mode. The size of each character could be either 8x8 or 16x16. Only four color mode 4, 16, 64, 256are valid in BK2. High color mode is invalid.

	Character size
BK2SIZE= 0	8x8
BK2SIZE = 1	16x16.

7.4.5 Linear mode

When BK2LINE = 1, BK2 operates in linear mode. In this mode the BK2SIZE should be set to 1. Only indexed color is valid in linear mode.

7.4.5.1 Indexed color mode

In this mode, following register should be set as BK2LINE = 1, BK2SIZE = 1, and color palette is necessary. Four kinds of color modes are valid, they are 4, 16 , 64 and 256 colors.

7.4.6 Indexed color mode

Four kinds of color modes are valid in BK2, they are 4, 16 , 64 and 256 colors.

Indexed Color Mode	BK2COLOR
4	0
16	1
64	2
256	3

And It's Color Palette mapping address is

Indexed Color Mode	Color Palette address ($002400 ~ $0027FF)								
	A9	A8	A7	A6	A5	A4	A3	A2	A1
4	BK2PALSEL	PALBANK[3:0]				0	0	COLORDATA	
16	BK2PALSEL	PALBANK[3:0]				COLORDATA			
64	BK2PALSEL	PALBANK[3:2]		COLORDATA					
256	BK2PALSEL	COLORDATA							

7.4.7 Graphic display depth

BK2 provides 4 display depth, the top is 0 and the button is 3. When it has the same depth with BK1 or BK3, their display priority would be BK1 > BK2 > BK3.

Depth	BK2DEPTH
Top	0
|	1
|	2
Button	3

7.4.8 VRAM memory space management

Before enable the BK2, the related PNT should be filled in to the mapped VRAM area. The mapped VRAM area is adjustable through the register BK2VRAMBANK.

7.3.8.1 16x16 character mode

Background single page mode (256x256),

BK2VRAMBANK	Mapped VRAM area
0	0x3000 ~ 0x31FF
1	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF
3	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF
5	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF
7	0x3E00 ~ 0x3FFF

Background horizontal two page mode(512x256),

BK2VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK2_X[8] = 0, and page1 is for BK2_X[8] = 1

Background vertical two page mode (256x512),

BK2VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK2_Y[8] = 0, and page1 is for BK2_Y[8] = 1

Background four page mode (512x512),

BK2VRAMBANK	Page0 mapped area	Page1 mapped area	Page2 mapped area	Page3 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
2	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
6	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK2_X[8] = 0, BK2_Y[8] = 0, page1 is for BK2_X[8] = 1, BK2_Y[8] = 0,

Page2 is for BK2_X[8] = 0, BK2_Y[8] = 1, page3 is for BK2_X[8] = 1, BK2_Y[8] = 1,

7.4.8.2 8x8 character mode

Background single page mode (256x256),

BK2VRAMBANK	mapped area
0	0x3000 ~ 0x37FF
1	0x3000 ~ 0x37FF
2	0x3000 ~ 0x37FF
3	0x3000 ~ 0x37FF
4	0x3800 ~ 0x3FFF
5	0x3800 ~ 0x3FFF
6	0x3800 ~ 0x3FFF
7	0x3800 ~ 0x3FFF

Background horizontal two page mode (512x256),

BK2VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
1	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
2	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
3	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
4	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
5	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
6	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
7	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF

Page0 is for BK2_X[8] = 0, and page1 is for BK2_X[8] = 1

Background vertical two page mode (256x512),

BK2VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
1	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
2	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
3	0x3000 ~ 0x37FF	0x3800 ~ 0x3FFF
4	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
5	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
6	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF
7	0x3800 ~ 0x3FFF	0x3000 ~ 0x37FF

Page0 is for BK2_Y[8] = 0, and page1 is for BK2_Y[8] = 1

7.4.8.3 Linear Mode

Background single page mode (256x256),

BK2VRAMBANK	mapped area
0	0x3000 ~ 0x31FF
1	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF
3	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF
5	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF
7	0x3E00 ~ 0x3FFF

Background horizontal two page mode (512x256),

BK2VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK2_X[8] = 0, and page1 is for BK2_X[8] = 1

Background vertical two page mode (256x512),

BK2VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK2_Y[8] = 0, and page1 is for BK2_Y[8] = 1

Background four page mode (512x512),

BK2VRAMBANK	Page0 mapped area	Page1 mapped area	Page2 mapped area	Page3 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
2	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
6	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK12_X[8] = 0, BK2_Y[8] = 0, page1 is for BK2_X[8] = 1, BK1_Y[8] = 0,

Page2 is for BK2_X[8] = 0, BK2_Y[8] = 1, page3 is for BK2_X[8] = 1, BK1_Y[8] = 1,

7.4.9 Graphic Pattern Address

Graphic PGT address is calculated from the PNT in VRAM, PGT start address SEGMENT, character size and color mode.

Physical PGT address(Bytes) = Start address + Offset

Start address = (BK2SEGMENT << 13).

Offset = (PNT x character horizontal size(pixel) x character vertical size(lines) x bit color mode) / 8

character horizontal size could be 8 or 16,

character vertical size could be 8 or 16

bit color mode :

Color mode	Bit color mode
4	2
16	4
64	6
256	8

For example:

Assume the PGT is located from address 0x20000,

Then BK2SEGMENT = (0x20000) << 13 = 0x10

If BK1 character size is 8x8, 256 color, then the PGT whose PNT=3 would be located at

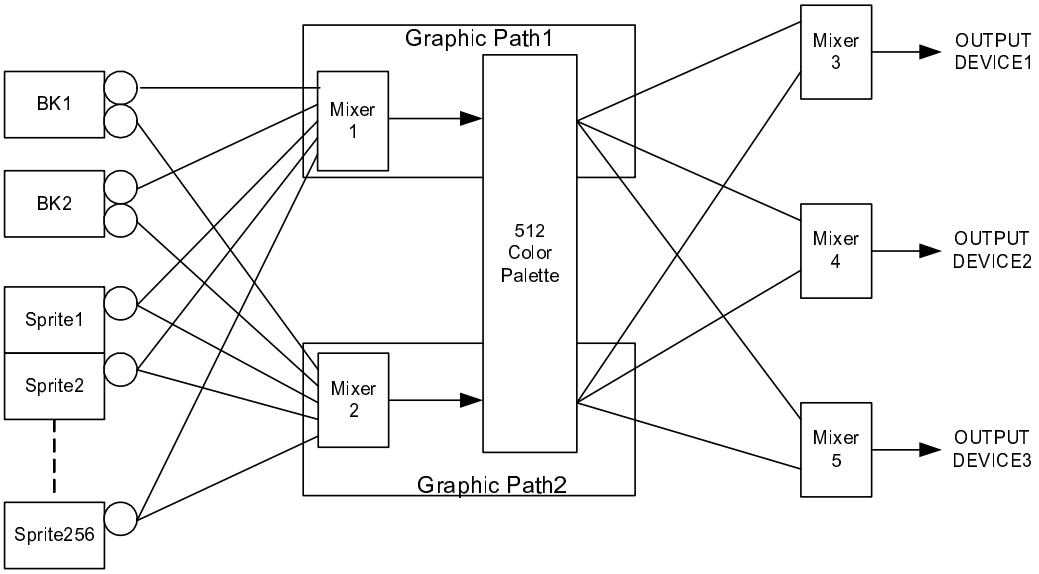
Offset = 3 x 8 x 8 x 8 = 0xC0

Physical PGT address = Start address + Offset = 0x200C0

7.4.10 Graphic path selection

VT268 provides two graphic data path as shown in the following diagram. BK2could choose one or both of them. Both path would share the same 512 color palette.

	Graphic Path1	Graphic path 2
BK2	BK2OutSel[0] = 1	BK2OutSel[1] = 1



7.4.11 Vertical scaling function

BK2 provides the vertical scaling function whose scaling ration is between 1/16 and 16 times separated into 256 levels. Please note that all BK1, BK2, BK3 and sprite use the same scaling ration whose vertical scaling function is enabled.

BK2VGain = 0	BK2VGain = 1
No Scaling function. (Scaling ration = 1)	The scaling would be started form the line VGAINSTARTLINE. All lines before VGAINSTARTLINE would be transparent. The scaling ration of the VGAINSTARTLINE line would be VGAININITIAL, then the ration of the following line would be increased 1/VGAINSTEP line by line.

7.4.12 Horizontal line scrolling function

BK2 horizontal line scaling function allows each horizontal line has an individual horizontal scrolling value, which is defined in a specified RAM area. This specified memory area address is started from { Scroll_Bank+1, 8'b0000_0000}, and each horizontal line maps to one byte whose scrolling value should be between –128 and +127, represented by 2's complement. Please note that this memory area is shared with BK1, BK2 and BK3 whose horizontal line scrolling function is enabled.

BK2LNEN = 0	BK2LNEN = 1
All lines' horizontal scrolling value is BK2_X	N^{th} horizontal scrolling value would be 爲 BK2_X + [Scroll_Bank+1, N]

7.4.13 Horizontal Scaling Function

Horizontal scaling function allows each horizontal line has the individual scaling ration, which is defined in a specified RAM area. This specified memory area address is started from { Scroll_Bank, 8'b0000_0000}, and each horizontal line maps to one byte whose scrolling value should be between 8 and 255 whose

scaling ration would be mapped to 8/64 and255/64. Please note that when this function is enabled, all graphic uses the graphic path2 would be scaled.

HSCALEN = 0	HSCALEN = 1
Graphic on Graphic path2 is out of scaling	The scale ration of the N^{th} on Graphic path2 is {Scroll_Bank, N} / 64

7.5 Background Layer 3(BK3)

The control registers of background layer3(BK3) are listed in the following table.

Address	Symbol	R/W	default
0x002002	P_BK3_X	R/W	0
0x002003	P_BK3_Y	R/W	0
0x002004	P_BK3_SETTING1	R/W	0
0x002005	P_BK3_SETTING2	R/W	0
0x002006	P_BK3_SEG_L	R/W	0
0x002007	P_BK3_SEG_H	R/W	0

7.5.1 Enable BK3

When BK3EN is set to 1, the BK3 would be displayed according to its parameters.

7.5.2 Coordinate

The X-axis and Y-axis of the background layer coordinate are between -256 and 255, represented by 2'scomplment. When X is positive(BK3_X[8] = 0), the display background would scroll left with X pixels. Otherwise it scrolls right. When Y is positive(BK3_Y[8] = 0), the display background would scroll up with Y pixels. Otherwise it scrolls down.

7.5.3 Scrolling Page

BK3 provides 4 type of scrolling mode, as shown in the following table. They are single page mode, horizontal two page mode, vertical two page mode and four page mode..

	BK3HSCRL = 0	BK3HSCRL = 1
BK3VSCRL = 0	single page mode	horizontal two page mode
BK3VSCRL = 1	vertical two page mode	four page mode

In different mode, the effective X, Y coordinate are different.

Scrolling mode	BK3VSCRL	BK3HSCRL	Effective X	Effective Y
single page mode	0	0	0 ~ 255	0 ~ 255
horizontal two page mode	0	1	0 ~ 511	0 ~ 255
vertical two page mode	1	0	0 ~ 255	0 ~ 511
four page mode	1	1	0 ~ 511	0 ~ 511

7.5.4 Character mode

When BK3LINE = 0, BK3 operates in character mode. The size of each character is 16x16. Only three color mode 4, 16, 64are valid in BK3 High color mode is invalid.

7.5.5 Linear mode

When BK3LINE = 1, BK3 operates in linear mode. Only indexed color is valid in linear mode.

7.5.5.1 Indexed color mode

In this mode, following register should be set as BK3LINE = 1, and color palette is necessary. Three kinds of color modes are valid, they are 4, 16 , and 64 colors.

7.5.6 Indexed color mode

Three kinds of color modes are valid in BK3, they are 4, 16 and 64 colors.

Indexed Color Mode	BK2COLOR
4	0
16	1
64	2

And It's Color Palette mapping address is

Indexed Color Mode	Color Palette address ($002400 ~ $0027FF)								
	A9	A8	A7	A6	A5	A4	A3	A2	A1
4	BK3PALSEL	PALBANK[3:0]				0	0	COLORDATA	
16	BK3PALSEL	PALBANK[3:0]				COLORDATA			
64	BK3PALSEL	PALBANK[3:2]		COLORDATA					

7.5.7 Graphic display depth

BK3 provides 4 display depth, the top is 0 and the button is 3. When it has the same depth with BK1 or BK3, their display priority would be BK1 > BK2 > BK3.

Depth	BK3DEPTH
Top	0
|	1
|	2
Button	3

7.4.8 VRAM memory space management

Before enable the BK3, the related PNT should be filled in to the mapped VRAM area. The mapped VRAM area is adjustable through the register BK3VRAMBANK.

7.3.8.1 16x16 character mode

Background single page mode (256x256),

BK3VRAMBANK	Mapped VRAM area
0	0x3000 ~ 0x31FF
1	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF
3	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF
5	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF
7	0x3E00 ~ 0x3FFF

Background horizontal two page mode(512x256),

BK3VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK3_X[8] = 0, and page1 is for BK3_X[8] = 1

Background vertical two page mode (256x512),

BK3VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK3_Y[8] = 0, and page1 is for BK3_Y[8] = 1

Background four page mode (512x512),

BK3VRAMBANK	Page0 mapped area	Page1 mapped area	Page2 mapped area	Page3 mapped area

0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
2	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
6	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK3_X[8] = 0, BK3_Y[8] = 0, page1 is for BK3_X[8] = 1, BK3_Y[8] = 0,

Page2 is for BK3_X[8] = 0, BK3_Y[8] = 1, page3 is for BK3_X[8] = 1, BK3_Y[8] = 1,

7.5.8.2 Linear Mode

Background single page mode (256x256),

BK3VRAMBANK	mapped area
0	0x3000 ~ 0x31FF
1	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF
3	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF
5	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF
7	0x3E00 ~ 0x3FFF

Background horizontal two page mode (512x256),

BK3VRAMBANK	Page0 mapped area	Page1 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK3_X[8] = 0, and page1 is for BK3_X[8] = 1

Background vertical two page mode (256x512),

BK3VRAMBANK	Page0 mapped area	Page1 mapped area

0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF
2	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF
6	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK3_Y[8] = 0, and page1 is for BK3_Y[8] = 1

Background four page mode (512x512),

BK3VRAMBANK	Page0 mapped area	Page1 mapped area	Page2 mapped area	Page3 mapped area
0	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
1	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
2	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
3	0x3000 ~ 0x31FF	0x3200 ~ 0x33FF	0x3400 ~ 0x35FF	0x3600 ~ 0x37FF
4	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
5	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
6	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF
7	0x3800 ~ 0x39FF	0x3A00 ~ 0x3BFF	0x3C00 ~ 0x3DFF	0x3E00 ~ 0x3FFF

Page0 is for BK3_X[8] = 0, BK3_Y[8] = 0, page1 is for BK3_X[8] = 1, BK3_Y[8] = 0,

Page2 is for BK3_X[8] = 0, BK3_Y[8] = 1, page3 is for BK3_X[8] = 1, BK3_Y[8] = 1,

7.5.9 Graphic Pattern Address

Graphic PGT address is calculated from the PNT in VRAM, PGT start address SEGMENT, character size and color mode.

Physical PGT address(Bytes) = Start address + Offset

Start address = (BK3SEGMENT << 13).

Offset = (PNT x character horizontal size(pixel) x character vertical size(lines) x bit color mode) / 8

character horizontal size is 16,

character vertical size is 16

bit color mode :

Color mode	Bit color mode
4	2
16	4
64	6

For example:

Assume the PGT is located from address 0x20000,

Then BK3SEGMENT = (0x20000) << 13 = 0x10

If BK3 character size is 8x8, 16 color, then the PGT whose PNT=3 would be located at

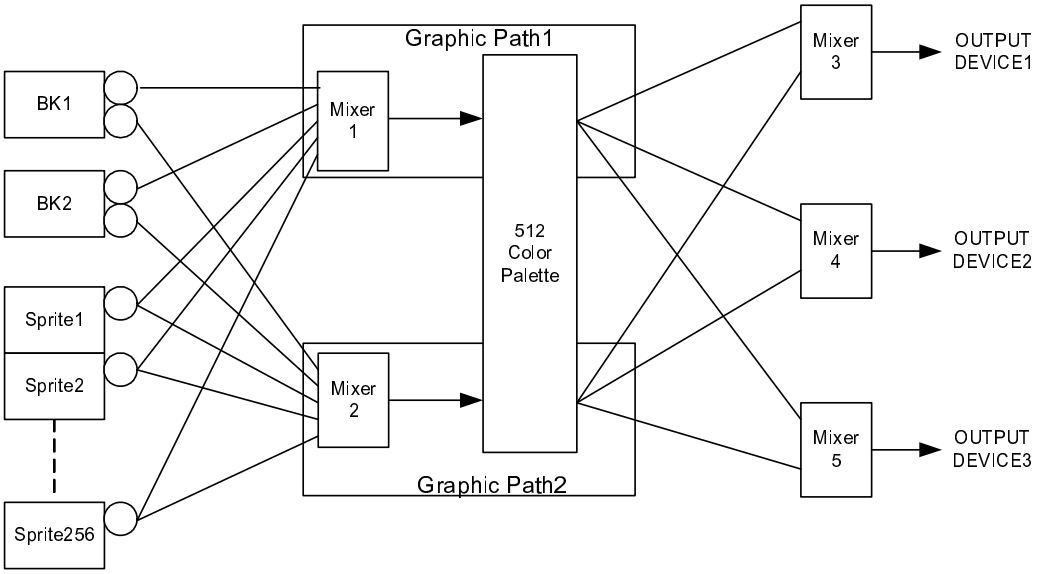
Offset = 3 x 16 x 16 x 4 = 0x180

Physical PGT address = Start address + Offset = 0x20180

7.5.10 Graphic path selection

VT268 provides two graphic data path as shown in the following diagram. BK3could choose one or both of them. Both path would share the same 512 color palette.

	Graphic Path1	Graphic path 2
BK3	BK3OutSel[0] = 1	BK3OutSel[1] = 1



7.5.11 Vertical scaling function

BK3 provides the vertical scaling function whose scaling ration is between 1/16 and 16 times separated into 256 levels. Please note that all BK1, BK2, BK3 and sprite use the same scaling ration whose vertical scaling function is enabled.

BK3VGain = 0	BK3VGain = 1
No Scaling function. (Scaling ration = 1)	The scaling would be started form the line VGAINSTARTLINE. All lines before VGAINSTARTLINE would be transparent. The scaling ration of the VGAINSTARTLINE line would be VGAININITIAL, then the ration of the following line would be increased 1/VGAINSTEP line by line.

7.5.12 Horizontal line scrolling function

BK3 horizontal line scaling function allows each horizontal line has an individual horizontal scrolling

value, which is defined in a specified RAM area. This specified memory area address is started from { Scroll_Bank+1, 8'b0000_0000}, and each horizontal line maps to one byte whose scrolling value should be between –128 and +127, represented by 2's complement. Please note that this memory area is shared with BK1, BK2 and BK3 whose horizontal line scrolling function is enabled.

BK3LNEN = 0	BK3LNEN = 1
All lines' horizontal scrolling value is BK3_X	N^{th} horizontal scrolling value would be 爲 BK3_X + [Scroll_Bank+1, N]

7.5.13 Horizontal Scaling Function

Horizontal scaling function allows each horizontal line has the individual scaling ration, which is defined in a specified RAM area. This specified memory area address is started from { Scroll_Bank, 8'b0000_0000}, and each horizontal line maps to one byte whose scrolling value should be between 8 and 255 whose scaling ration would be mapped to 8/64 and255/64. Please note that when this function is enabled, all graphic uses the graphic path2 would be scaled.

HSCALEN = 0	HSCALEN = 1
Graphic on Graphic path2 is out of scaling	The scale ration of the N^{th} on Graphic path2 is {Scroll_Bank, N} / 64

7.6 Sprite Layer

There are 256 sprites in display screen. Each sprite cell includes PGT vector, XY coordinate, depth layer, vertical flip, horizontal flip, palette select, effect area definition, color mode, vertical and horizontal size and graphic processor selection

7.6.1 Sprite RAM data format

The sprite memory area is between 0x002800 ~ 0x002FFF. Every sprite cell is combined by six bytes in every eight bytes memory area as shown in the following example. In other word, there are only 256x6 bytes in this 256x8 bytes memory area. There are two data format for different applications selected by SPTYPE.

When SPTYPE = 0,

	D7	D6	D5	D4	D3	D2	D1	D0
SP*8	Vector[7:0]							
SP*8 + 1	V_SIZE[1:0]		H_SIZE[1:0]		Vector[11:8]			
SP*8 + 2	X[7:0]							
SP*8 + 3			COLOR	Depth[1:0]		VFLIP	HFLIP	X[8]
SP*8 + 4	Y[7:0]							
SP*8 + 5	Palette[4:0]					OUTSEL	EFFCT	Y[8]
SP*8 + 6	NULL							
SP*8 + 7	NULL							

When SPTYPE = 1,

	D7	D6	D5	D4	D3	D2	D1	D0
SP*8	Vector[7:0]							
SP*8 + 1	V_SIZE[1:0]		Vector[13:8]					
SP*8 + 2	X[7:0]							
SP*8 + 3	H_SIZE[1:0]		COLOR	Depth[1:0]		VFLIP	HFLIP	X[8]
SP*8 + 4	Y[7:0]							
SP*8 + 5	Palette[4:0]					OUTSEL	EFFCT	Y[8]
SP*8 + 6	NULL							
SP*8 + 7	NULL							

Here is description of each sprite parameter

VECTOR[13….0] : The PGT of sprite, and $000 means the transparent sprite

V_SIZE[1..0] : The vertical size of sprite cell.

00 : 8 lines, 01:16 lines, 10:64 lines, 11:forbidden

H_SIZE[1..0] : The horizontal size of sprite cell.

00 : 8 dots, 01:16dots, 10:64 dots, 11:forbidden

X[8..0] : X coordinate of sprite cell, use 2's complement.

Y[8..0] : Y coordinate of sprite cell, use 2's complement.

VFLIP : Sprite cell vertical flip enable control

0 : normal 1 : vertical flip

HFLIP : Sprite cell horizontal flip enable control

0 : normal 1 : horizontal flip

DEPTH[1..0] : Define the depth layer of the sprite cell

00 : top …… 11 : button

COLOR : Sprite cell color mode

0 : 16 color (4bits/pixel) 1 : 64 color(6bits/pixel)

EFFECT : Effect area definition

0 : normal graphic character 1 : effect area indicator

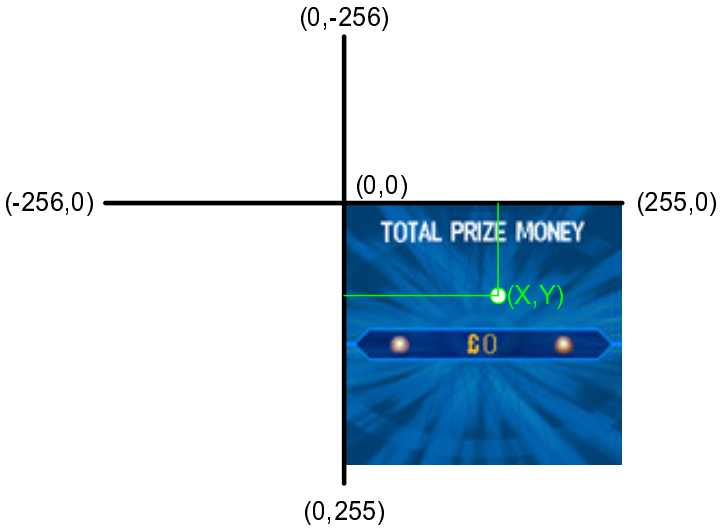
OUTSEL : graphic path selection

0 : graphic path1 1 : graphic path2

PALETTE[4..0] : Color Palette bank selection

7.6.2 Sprite Coordinate

The X-axis and Y-axis of the background layer coordinate are between 0 and 255, represented by 2'scomplment. Position (0,0) is the left top corner of the display screen



7.6.3 Sprite Enable

When the SPEN is P_SP_SETTING is set, all sprite cells with non-zero PGT would be display on the display screen

7.6.4 Graphic Address

Graphic PGT address is calculated from the PNT in sprite RAM, PGT start address SEGMENT, sprite cell size and color mode.

Physical PGT address(Bytes) = Start address + Offset

Start address = (SPSEGMENT << 13).

Offset = (PNT x cell horizontal size(pixel) x cell vertical size(lines) x bit color mode) / 8

character horizontal size could be 8, 16 or 32

character vertical size could be 8, 16 or 32

bit color mode :

Color mode	Bit color mode
4	2
16	4
64	6

For example:

Assume the PGT is located from address 0x20000,

Then SPSEGMENT = (0x20000) << 13 = 0x10

If sprite cell size is 8x8, 16 color, then the PGT whose PNT=3 would be located at

Offset = 3 x 16 x 16 x 4 = 0x180

Physical PGT address = Start address + Offset = 0x20180

When the physical address area is between $000000 and $000FFF, the memory area would map to the internal VRAM area $003000 ~ $003FFF. When the physical address is greater than $001000, the

memory area would map to the external memory.

7.6.5 Sprite 512 dots resolution mode

In normal mode, both background layers and sprite graphic resolution of VT268 is 256 dots x 240 lines. When the SP512M is set, the horizontal resolution of sprite would change to 512 dots which is the combination of the two graphic path, each one is 256 dots. The graphic data from graphic path1 would be located at the left half of the screen. While the graphic data from the graphic path2 would be located at the right half. The sprite cells that appears on the half screen should be duplicated on both graphic paths.

7.6.6 Vertical scaling function

Sprite provides the vertical scaling function whose scaling ration is between 1/16 and 16 times separated into 256 levels. Please note that all BK1, BK2, BK3 and sprite use the same scaling ration whose vertical scaling function is enabled.

SPVGain = 0	SPVGain = 1
No Scaling function. (Scaling ration = 1)	The scaling would be started form the line VGAINSTARTLINE. All lines before VGAINSTARTLINE would be transparent. The scaling ration of the VGAINSTARTLINE line would be VGAININITIAL, then the ration of the following line would be increased 1/VGAINSTEP line by line.

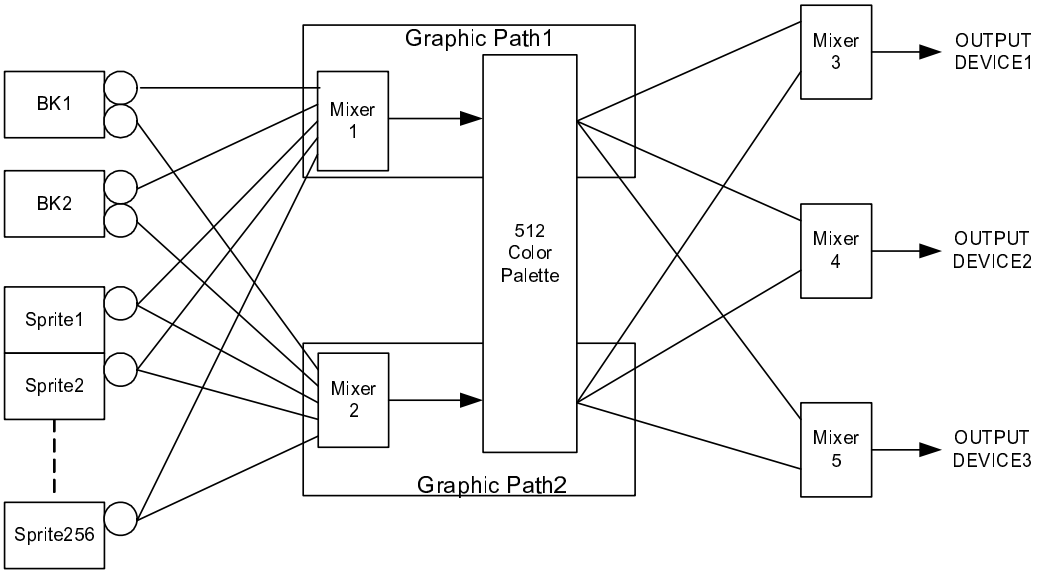
7.6.7 Graphic path selection

VT268 provides two graphic data path as shown in the following diagram. Sprite could choose one of them. Both paths would share the same 512 color palette.

	Graphic Path1	Graphic path 2
Sprite	OUTSEL=0	OUTSEL = 1

7.7 Graphic path selection

VT268 provides two graphic paths and three graphic output device. Suitable combination could provide different graphic effects. The combination includes overlap and alpha blending as shown in the following diagram.



Graphic output1 control

	BLD1EN = 0	BLD1EN = 1
OVLP1EN = 0	Output graphic path 1	Output the 50% alpha blending of graphic path1 and graphic path2
OVLP1EN = 1	Output the overlap of graphic path1 and graphic path2 based on their depth	Output the overlap of graphic path1 and graphic path2 based on their depth

Graphic output2 control

	BLD2EN = 0	BLD2EN = 1
OVLP2EN = 0	Output graphic path 2	Output the 50% alpha blending of graphic path1 and graphic path2
OVLP2EN = 1	Output the overlap of graphic path1 and graphic path2 based on their depth	Output the overlap of graphic path1 and graphic path2 based on their depth

7.8 Graphic output device

VT268 provides three output devices, they are TV composite video, LCD1 and LCD2. Each output device could choose one of the graphic output described above.

	Graphic otuput1	Graphic output2
LCD1	LCD1SEL = 0	LCD1SEL = 1
LCD2	LCD2SEL = 0	LCD2SEL = 1
Composite Video	TVSEL = 0	TVSEL = 1

7.9 CRT Light Gun interface (Pulse Latch)

VT268 has two pulse latch interface used in the CRT light-gun application. The rising edge of the input pulse(UIOG6 or UIOG7) would trigger the CPU to memory the current X and Y position in the P_LIGHTGUN_X and P_LIGHTGUN_Y.

Initialization:

```
LDA  C_UIOG_DIR          ; Set UIOG6 to input mode
AND  $BF
STA  C_UIOG_DIR
:                        ; Set the pull-high or pull-low if necessary
STA  P_LIGHTGUN_CLR      ; Clear XY memory
```

NMI example:

```
LDA  P_LIGHTGUN_Y        ; Take vertical position
ASL  A
BEQ  No_Light_Gun_Detect ; if P_LIGHTGUN_Y=0, no pulse is detected
STA  LIGHT_GUN_Y         ; Store vertical position
LDA  P_LIGHTGUN_X        ; Take horizontal position
STA  LIGHT_GUN_X         ; Store horizontal position
```

7.10 Special effects

VT268 provides kinds of graphic special effect for different applications. The effects include dig-hole effect, fade effect, blending effect, brown picture effect, halftone effect and gray effect. The effects of two graphic channels are controlled independently.

7.10.1 Effect Area definition

There are two ways to define the effect area, one is using the sprite with setting the “EFFCT” bit, and the other is using the “EF” flag in Color Palette. Please note that “Dig-Hole” effect is only valid for sprite effect bit. All other effects are valid in sprite effect bit and palette effect bit. Only one of the definition method could exist in one graphic path. The effect area could be applied on the exclusive area as shown in the flowing table.

Graphic path1 special effect area definition

FADE1ALL	EFF1XOR	EFF1SRC	Special effect area
0	0	0	All pixels use the palette with EF=1
0	1	0	All pixels use the palette with EF=0
0	0	1	Non-transparent pixels in sprite whose EFFCT=1
0	1	1	All sprite pixels except the non-transparent pixels in sprite whose EFFCT=1
1	0	0	All pixels

*The special effect area is valid only when the EFF1SEL is non-zero.

Graphic path2 special effect area definition

FADE2ALL	EFF2XOR	EFF2SRC	Special effect area
0	0	0	All pixels use the palette with EF=1
0	1	0	All pixels use the palette with EF=0
0	0	1	Non-transparent pixels in sprite whose EFFCT=1
0	1	1	All sprite pixels except the non-transparent pixels in sprite whose EFFCT=1
1	0	0	All pixels

*The special effect area is valid only when the EFF2SEL is non-zero.

7.10.2 Special effect selection

Graphic path1 special effect

EFF1SEL	Special effect	Description
0	non	---
1	Fade in / fade out	Related to FADEINOUT and FADE1_OFFSET Generated up to 32768 color levels, please reference the "fade-in fade-out effect" in the later chapter
2	Gray level	Change to 32 gray-level
3	Color tone	Provide half-tone or brown effect FADEINOUT = 0, half-tone effect FADEINOUT = 1, brown effect

Graphic path2 special effect

EFF2SEL	Special effect	Description
0	non	---
1	Fade in / fade out	Related to FADE2NOUT and FADE2_OFFSET Generated up to 32768 color levels
2	Gray level	Change to 32 gray-level
3	Color tone	Provide half-tone or brown effect FADE2NOUT = 0, half-tone effect FADE2NOUT = 1, brown effect

7.10.3 Dig-Hole effect

Except the effects selected by the EFF1SEL /SFF2SEL, there is the other special effect dig-hole whose special effect area is only be defined by EFFCT in sprite cell. When EFFCT = 1, the marked area would become completely transparent. This effect is suggested to use two graphic path with overlap mode.

Digging hole on the first graphic path, and overlapping on the second graphic path is the suggestion application method.

DIG1EN = 0	Graphic path1 dig-hole effect enable
DIG1EN = 1	Graphic path1 dig-hole effect disable
DIG2EN = 0	Graphic path2 dig-hole effect enable
DIG2EN = 1	Graphic path2 dig-hole effect disable

7.10.4 Fade-in Fade-out effect

VT268 provide Red, Green and Blue color individually 32 level fade-in and fade-out control, that is totally32768 color levels. Fade-in from black screen, fade-out to white screen, fade-in from while screen or fade-out to black screen are all available. It could also be used to adjust the color level of the screen. The special effect area is described In the previous chapter. With well control of the graphic output, two different fade level are also possible.

Graphic path1 example 1, fade-in from black screen:

1. Set FADEINOUT = 0, EFF1SEL = 1, and mark the special effect area
2. Set FADE1_RED (FADE1_OFFSET[14:10]) = 31, FADE1_GREEN (FADE1_OFFSET[9:5]) = 31, FADE1_BLUE(FADE1_OFFSET[4:0]) = 31
3. Use the NMI to control the fade-in speed
4. Let FADE1_RED = FADE1_RED –1, FADE1_ GREEN = FADE1_ GREEN–1, FADE1_ BLUE = FADE1_ BLUE–1
5. Repeat step 3 and step 4 until FADE1_RED = 0

Note : Programmer could fine tune the FADE1_OFFSET change table to generate the best fade-in color.

Graphic path1 example 2, fade-in from white screen:

1. Set FADEINOUT = 1, EFF1SEL = 1, and mark the special effect area
2. Set FADE1_RED (FADE1_OFFSET[14:10]) = 31, FADE1_GREEN (FADE1_OFFSET[9:5]) = 31, FADE1_BLUE(FADE1_OFFSET[4:0]) = 31
3. Use the NMI to control the fade-in speed
4. Let FADE1_RED = FADE1_RED –1, FADE1_ GREEN = FADE1_ GREEN–1, FADE1_ BLUE = FADE1_ BLUE–1
5. Repeat step 3 and step 4 until FADE1_RED = 0

Note : Programmer could fine tune the FADE1_OFFSET change table to generate the best fade-in color.

Graphic path1 example 2, fade-out to black screen:

1. Set FADEINOUT = 0, EFF1SEL = 1, and mark the special effect area
2. Set FADE1_RED (FADE1_OFFSET[14:10]) = 0, FADE1_GREEN (FADE1_OFFSET[9:5]) = 0, FADE1_BLUE(FADE1_OFFSET[4:0]) = 0
3. Use the NMI to control the fade-in speed

4. Let FADE1_RED = FADE1_RED +1, FADE1_ GREEN = FADE1_ GREEN+1,
   FADE1_ BLUE = FADE1_ BLUE+1
5.Repeat step 3 and step 4 until FADE1_RED = 31
Note : Programmer could fine tune the FADE1_OFFSET change table to generate the best fade-in color.

Graphic path1 example 2, fade-out to white screen:
1. Set FADEINOUT = 1, EFF1SEL = 1, and mark the special effect area
2. Set FADE1_RED (FADE1_OFFSET[14:10]) = 0, FADE1_GREEN (FADE1_OFFSET[9:5]) = 0,
   FADE1_BLUE(FADE1_OFFSET[4:0]) = 0
3. Use the NMI to control the fade-in speed
4. Let FADE1_RED = FADE1_RED +1, FADE1_ GREEN = FADE1_ GREEN+1,
   FADE1_ BLUE = FADE1_ BLUE+1
5.Repeat step 3 and step 4 until FADE1_RED = 31
Note : Programmer could fine tune the FADE1_OFFSET change table to generate the best fade-in color.

7.11 CCIR Layer

VT268 has CCIR(ITU601 / ITU656) protocol to input the graphic image named CCIR layer. CCIR layer could be outputted to either or both of the two graphic paths, and is always located at the button of all image layers. The input format could be YUV422, RGB422, RGB565 or RGB555. For the detail CCIR setting and application note, please contact with the VRT FAE.

7.11.1 *i*-interface

VT268 provides the comparison of the two continuous CCIR image for the software application. CCIR image is partition into 240 blocks (16x15), the luminance of the center of each block is stored in memory area {EYE_PRAM_BANK, 7'b000_0000} used for the comparison in the next CCIR image. When the difference is greater than EYE_THRESHOLD, the flag in P_EYE_STATUS_L or P_EYE_STATUS_H would be set. It's suggested to read the flags in the NMI.

7.11.2 CCIR Pixel Color Capture

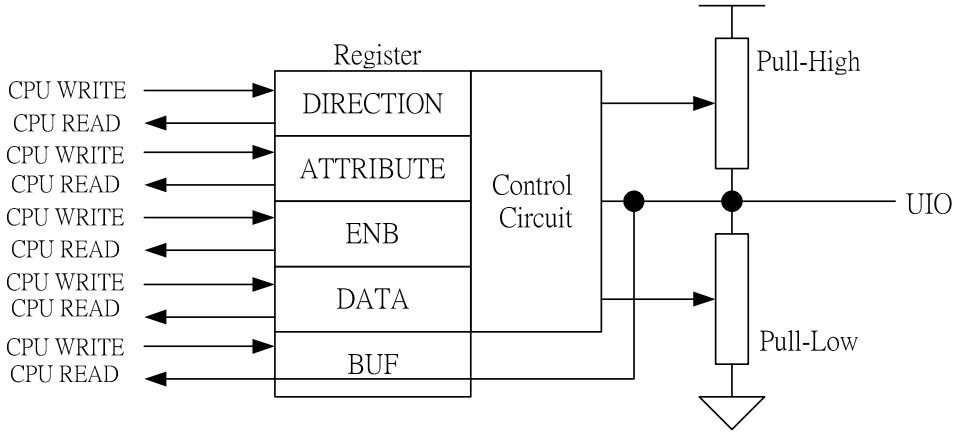
VT268 could strobe the RGB color at CCIR specified position. The position is defined by P_STROBE_X and P_STROBE_Y, whose coordinate is based on the graphic(BK or sprite). The strobe color information(RGB555 format) is valid at P_STROBE_DATAL and STROBE_DATAH.

7.12 Horizontal scan line number

The current display horizontal scan line number is readable at the port P_HSCALNUM. During the display area, the value range would be between 0x01 ~ 0xF0.

8. I/O

There are 56 I/Os are valid in VT268, they are UIOA[7:0], UIOB[7:0], UIOC[7:0], UIOD[7:0], UIOE[7:0], UIOF[7:0], and UIOG[7:0]. Each of them is bit-wise controlled. Each bit has an individual direction (IN / OUT) and attribute (pull-high, pull-low / floating) parameters. Parts of peripheral interface shared these I/O ports, such as LCD, UART, SPI and I2C also bit-wise select as listed in the following table.



8.1 UIOA

UIOA provides the output high, output low, input with pull-up, input with pull-low, input floating and some particular output function. The particular function please reference the chapter " Shared I/O".

DIR	ATTR	DATA	ENB	IO mode
0	1	X	X	input floating
0	0	0	X	Input with pull low resistor
0	0	1	X	Input with pull high resistor(default)
1	X	0	0	Input with pull high resistor
			1	Output Low
1	X	1	0	Specified function
			1	Output High

For example, if we are going to set UIOA0 output high, UIOA1 input floating and UIOA2 input with pull-high, then

```
LDA  C_UIOA_DIR
ORA  #$01          ; UIOA0 as output
AND  #$06          ; UIOA1/2 as input
STA  C_UIOA_DIR
LDA  C_UIOA_ATTR
ORA  $04           ; UIOA2 pull-high/pull-low
AND  $02           ; UIOA1 floating
STA  C_UIOA_ATTR
LDA  C_UIOA_DATA
```

```
ORA  #$05                ; UIOA2 pull-high, UIOA0 output high
STA  C_UIOA_DATA
```

8.2 UIOB

UIOB provides the output high, output low, input with pull-up, input with pull-low, input floating and some particular output function. The particular function please reference the chapter " Shared I/O".

DIR	ATTR	DATA	ENB	IO mode
0	1	X	X	input floating
0	0	0	X	Input with pull low resistor
0	0	1	X	Input with pull high resistor(default)
1	X	0	0	Input with pull high resistor
			1	Output Low
1	X	1	0	Specified function
			1	Output High

8.3 UIOC

UIOC provides the output high, output low, input with pull-up, input with pull-low, input floating and some particular output function. The particular function please reference the chapter " Shared I/O".

DIR	ATTR	DATA	ENB	IO mode
0	1	X	X	input floating
0	0	0	X	Input with pull low resistor
0	0	1	X	Input with pull high resistor(default)
1	X	0	0	Input with pull high resistor
			1	Output Low
1	X	1	0	Specified function
			1	Output High

8.4UIOD

UIOD provides the output high, output low, input with pull-up, input with pull-low, input floating and some particular output function. The particular function please reference the chapter " Shared I/O".

DIR	ATTR	DATA	ENB	IO mode
0	1	X	X	input floating
0	0	0	X	Input with pull low resistor
0	0	1	X	Input with pull high resistor(default)
1	X	0	0	Input with pull high resistor
			1	Output Low
1	X	1	0	Specified function

			1	Output High

8.5 UIOE

UIOE provides the output high, output low, input with pull-up, input with pull-low, input floating and some particular output function. The particular function please reference the chapter " Shared I/O".

DIR	ATTR	DATA	ENB	IO mode
0	1	X	X	Input floating
0	0	0	X	Input with pull low resistor
0	0	1	X	Input with pull high resistor(default)
1	X	0	0	Input with pull high resistor
			1	Output Low
1	X	1	0	Specified function
			1	Output High

8.6 UIOF

UIOF provides the output high, output low, input with pull-up, input with pull-low, input floating, some particular output function, analog(for analog LCD) and cap switch application. The particular function please reference the chapter " Shared I/O".

DIR	ATTR	DATA	ENB	IO mode
0	1	0	X	Input floating
0	1	1	X	Analog mode
0	0	0	X	Input with pull low resistor(default)
0	0	1	X	Input with pull high resistor
1	1	X	0	Cap switch mode
			1	Reserved
1	0	1	0	Output Low
			1	Specified function
1	0	0	0	Output High
			1	Specified function

8.7 UIOG

UIOG provides the output high, output low, input with pull-up, input with pull-low, input floating, some particular output function, analog(for ADC) and cap switch application. The particular function please reference the chapter " Shared I/O".

DIR	ATTR	DATA	ENB	IO mode
0	1	0	X	Input floating
0	1	1	X	Analog mode

0	0	0	X	Input with pull low resistor(default)
0	0	1	X	Input with pull high resistor
1	1	X	0	Cap switch mode
			1	Reserved
1	0	1	0	Output Low
			1	Specified function
1	0	0	0	Output High
			1	Specified function

8.8 Shared I/O

Some particular functions are shared with GPIO, include SPI, LCD, IRRC, SD, UART, IIC, CCIR, or ADC.

8.8.1 SPI interface

IO port	IO mode	Interface pin name	
UIOA0	Specified function	SPI CSB	
UIOA1	Specified function	SPI CK	
UIOA2	Input floating		SPIDI
UIOA3	Specified function	SPI DO	

8.8.2 LCD interface

IO port	IO mode	Interface pin name
UIOA4	Specified function	LCD1.D0
UIOA5	Specified function	LCD1.D1
UIOA6	Specified function	LCD1.D2
UIOA7	Specified function	LCD1.D3
UIOB0	Specified function	LCD1.D4
UIOB1	Specified function	LCD1.D5
UIOB2	Specified function	LCD1.D6
UIOB3	Specified function	LCD1.D7
UIOB4	Specified function	LCD1.D8
UIOB5	Specified function	LCD1.D9
UIOB6	Specified function	LCD1.D10
UIOB7	Specified function	LCD1.D11
UIOC0	Specified function, C_UIOC_SEL[0] = 0	LCD1.D12
UIOC1	Specified function	LCD1.D13
UIOC2	Specified function, C_UIOC_SEL[2] = 0	LCD1.D14
UIOC3	Specified function, C_UIOC_SEL[3] = 0	LCD1.D15
UIOC4	Specified function, C_UIOC_SEL[4] = 0	LCD1.D16

UIOC5	Specified function	LCD1.D17
UIOC6	Specified function	LCD1.D18
UIOD0	Specified function	LCD2.D0
UIOD1	Specified function	LCD2.D1
UIOD2	Specified function	LCD2.D2
UIOD3	Specified function	LCD2.D3
UIOD4	Specified function	LCD2.D4
UIOD5	Specified function	LCD2.D5
UIOD6	Specified function	LCD2.D6
UIOD7	Specified function	LCD2.D7

8.8.3 IRRC interface

IO port	IO mode	Interface pin name	
UIOC0	Specified function, C_UIOC_SEL[0] = 1	IRRCTX	
UIOC1	Input floating		IRRCRX

8.8.4 SD interface

IO port	IO mode	Interface pin name	
UIOC2	Specified function, C_UIOC_SEL[2] = 1	SDCK	
UIOC3	Specified function, C_UIOC_SEL[3] = 1	SDCSB	
UIOC4	Specified function, C_UIOC_SEL[4] = 1	SDDO	
UIOC5	Input floating		SDDI

8.8.5 UART interface

IO port	IO mode	Interface pin name	
UIOE0	Specified function	UARTTX	
UIOE1	Input floating		UARTRX

8.8.6 IIC interface

IO port	IO mode	Interface pin name
UIOE2	Input floating	IICCK
UIOE3	Input floating	IICDA

8.8.7 CCIR interface

IO port	IO mode	Interface pin name
UIOE5	Input floating	CCIRVS
UIOE6	Input floating	CCIRHS

UIOE7	Input floating	CCIRCK
UIOF0	Input floating	CCIRD0
UIOF1	Input floating	CCIRD1
UIOF2	Input floating	CCIRD2
UIOF3	Input floating	CCIRD3
UIOF4	Input floating	CCIRD4
UIOF5	Input floating	CCIRD5
UIOF6	Input floating	CCIRD6
UIOF7	Input floating	CCIRD7

8.8.8 PWM interface

IO port	IO mode	Interface pin name
UIOG4	Specified function	PWM1
UIOG5	Specified function	PWM2
UIOG6	Specified function	PWM3

8.8.9 Others

IO port	IO mode	Interface pin name
UIOC7	Specified function	ROMCSB2
UIOE4	Specified function	CSYNC

9. Multiplier and Divider

The multiplier is 16 bit by 16 bits takes 16 CPU clock to complete the operation. The division is 32 bits by 16 bits takes 32 CPU clock to complete the operation.

9.1 Multiplier

The multiply operation is,

MULTIPLIER_IN2 x MULTIPLIER_IN1 = MULTIPLIER_OUT

For example, for the operation $1234 x $5678, then

MULTIPLIER_IN1 = $1234

MULTIPLIER_IN2 = $5678

The operation is started when the MULTIPLIER_IN2[15:8] is written. After 16 CPU clock, the result would be valid at MULTIPLIER_OUT. Please note that do not change the 0x02130~0x02135 until the operation is complete.

address	Symbol	Description
0x2130(W)	C_MUL_IN1_L	MULTIPLIER_IN1[7:0]
0x2131(W)	C_MUL_IN1_H	MULTIPLIER_IN1[15:8]
0x2134(W)	C_MUL_IN2_L	MULTIPLIER_IN2[7:0]

0x2135(W)	C_MUL_IN2_H	MULTIPLIER_IN2[15:8]
0x2138(R)	C_MUL_OUT_L	MULTIPLIER_OUT[7:0]
0x2139(R)	C_MUL_OUT_M1	MULTIPLIER_OUT[15:8]
0x213A(R)	C_MUL_OUT_M2	MULTIPLIER_OUT[23:16]
0x213B(R)	C_MUL_OUT_H	MULTIPLIER_OUT[31:24]

9.2 Divider

The division operation is,

DIVIDER_IN1 / DIVIDER_IN2 = DIVIDER_OUT1(quotient) + DIVIDER_OUT2(remainder)

For example, for the operation $12345678 / $ABCD, then

DIVIDER_IN1 = $12345678,

DIVIDER_IN2 = $ABCD

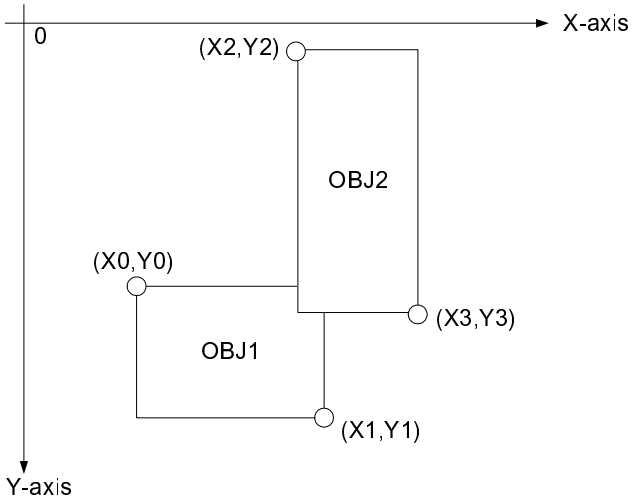
The operation is started when the DIVIDER_IN2 [15:8] is written. After 32 CPU clock, the quotient would be valid at DIVIDER_OUT1, and the remainder would be at DIVIDER_OUT2. Please note that do not change the 0x02130~0x02135 until the operation is complete.

address	Symbol	Description
0x2130(W)	C_DIV_IN1_L	DIVIDER_IN1[7:0]
0x2131(W)	C_DIV_IN1_M1	DIVIDER_IN1[15:8]
0x2132(W)	C_DIV_IN1_M2	DIVIDER_IN1[23:16]
0x2133(W)	C_DIV_IN1_H	DIVIDER_IN1[31:24]
0x2136(W)	C_DIV_IN2_L	DIVIDER_IN2[7:0]
0x2137(W)	C_DIV_IN2_H	DIVIDER_IN2[15:8]
0x2138(R)	C_DIV_OUT1_L	DIVIDER_OUT1[7:0]
0x2139(R)	C_DIV_OUT1_M1	DIVIDER_OUT1[15:8]
0x213A(R)	C_DIV_OUT1_M2	DIVIDER_OUT1[23:16]
0x213B(R)	C_DIV_OUT1_H	DIVIDER_OUT1[31:24]
0x213C(R)	C_DIV_OUT2_L	DIVIDER_OUT2[7:0]
0x213D(R)	C_DIV_OUT2_H	DIVIDER_OUT2[15:8]

10. Rectangular Overlap Detection

VT268 provides the hardware rectangular overlap detection to speed up the sprite collision detection. Writing two set the coordinate of the detected object corners, as shown in the following diagram. And get the collision status from the status port. Please note that this module could only process the rectangular with coordinate between 0 and 255.

For example :



It means

0x2190 = X0,

0x2191 = X1,

0x2192 = Y0,

0x2193 = Y1,

0x2194 = X2,

0x2195 = X3,

0x2196 = Y2,

0x2197 = Y3,

Then read the status C_COL_STATUS.

BUSY	Collision	Description
1	X	Operation is proceeding
0	0	No collision
0	1	Collision

11. Real Time Clock and Alarm

VT268 provides the RTC function and one set of alarm. When the RTC function is applied, the crystal 32768Hz should be added on the XTAL3 and XTAL4. Current time is valid from C_RTC_HR, C_RTC_MIN and C_RTC_SC. When the RTC reach the alarm time, programmer could get it through the RTC_IRQ or the status flag ALMSTAT.

Real Time Clock initialization:

```
LDA  #$3F
STA  C_RTL_PAD_CTRL
LDA  #HOUR
STA  C_RTC_HR        ; hour
LDA  #MINUTE
STA  C_RTC_MN        ; min
LDA  #SECOND
STA  C_RTC_SC        ; second
STA  C_RTC_LOAD      ; upload to RTC
LDA  #1
STA  C_RTC_ENABLE ; enable the RTC
```

Alarm initialization:

```
LDA  #ALARM_HOUR
STA  C_ALARM_HR          ; hour
LDA  #ALARM_MINUTE
STA  C_ALARM_MN          ; minute
LDA  #ALARM_SECOND
STA  C_ALARM_SC          ; second
```

12. Sleep and Wakeup

VT268 could enter the sleep mode to save the power consumption. When the sleep or wakeup function is applied, the crystal 32768Hz should be added on the XTAL3 and XTAL4. There are three ways to wake up from the sleep mode. The first is periodical wake up, the second is alarm, and the third is particular IO changed. The wake up mode could be set to either or all of them.

12.1 Periodical wakeup

Example:

```
        LDA  #$3F
        STA  C_RTL_PAD_CTRL
        LDA  #$1
        STA  C_RTL_ENABLE       ; Enable RTC
        LDA  #WAKE_PERIOD       ; Set wakeup period
        STA  C_WAKE_PERD
        LDA  #WAKE_CLOCK        ; Set wakeup clock source
        ORA  #$10               ; Enable periodical wakeup
        STA  C_KEY_WAKE_MD
        LDA  #$55               ; enter sleep mode
        STA  C_ENTER_SLEEP
        LDX  #$0
L_Wait_Sleep:
        NOP
        INX
        BNE  L_Wait_Sleep
```

WAKECNTSEL	Wakeup clock source	Wakeup period
0	OFF	---
1	1/32768 sec	(256- #WAKE_PERIOD) /32768 sec
2	1/128 sec	(256- #WAKE_PERIOD) /128 sec
3	1 sec	(256- #WAKE_PERIOD) sec

12.2 Alarm Wakeup

Example:

```
        LDA  #$3F
        STA  C_RTL_PAD_CTRL
        LDA  #ALARM_HOUR        ; Set Alarm
        STA  C_ALARM_HR         ; hour
        LDA  # ALARM_MINUTE
```

```
        STA  C_ALARM_MN          ; minute
        LDA  # ALARM_SECOND
        STA  C_ALARM_SC          ; second
        LDA  #$1
        STA  C_RTL_ENABLE        ; enable RTC
        LDA  C_KEY_WAKE_MD
        ORA  #$4                 ; enable Alarm wakeup
        STA  C_KEY_WAKE_MD
        LDA  #$55                ; enter sleepmode
        STA  C_ENTER_SLEEP
L_Wait_Sleep:
        NOP
        INX
        BNE  L_Wait_Sleep
```

ALMWAKE	Alarm wak p mode
0	OFF
1	ON

12.3 IO Trigger

Initialization:

```
    LDA  #$3F
    STA  C_RTL_PAD_CTRL
    LDA  #$02                ; IO trigger wake up source select
    STA  C_KEY_WAKE_SEL
    LDA  C_KEY_WAKE_MD       ; IO trigger wake up enable
    ORA  #$08
    STA  C_KEY_WAKE_MD
    LDA  #$55                ; enter sleep mode
    STA  C_ENTER_SLEEP
L_Wait_Sleep:
    NOP
    INX
    BNE  L_Wait_Sleep
```

IOCWAKE	IO trigger wake up mode
0	UIOC[2:0] trigger would not wake up
1	When UIOC2, UIOC1 or UIOC0 trigger, CPU wake up*

IOGWAKE	IO trigger wake up mode
0	UIOG[2:0] trigger would not wake up
1	When UIOG2, UIOG1 or UIOG0 trigger, CPU wake up*

* It's suggested to set the trigger IO to input mode. For example, when UIOG0 is trigger from 0 to 1, CPU would wake up. Then

```
        LDA  #$3F
        STA  C_RTL_PAD_CTRL
        LDA  C_UIOG_DIR
        AND  #$FE
        STA  C_UIOG_DIR
        LDA  C_UIOG_DATA
        AND  #$FE
        STA  C_UIOG_DATA
        LDA  C_UIOG_ATTR
        AND  #$FE
        STA  C_UIOG_ATTR            ; Set to input pull-low
        LDA  #$02                   ; UIOG[2:0] trigger
        STA  C_KEY_WAKE_SEL
        LDA  C_KEY_WAKE_MD          ; IO trigger wake up enable
        ORA  #$08
        STA  C_KEY_WAKE_MD
        LDA  #$55                   ; enter sleep mode
        STA  C_ENTER_SLEEP
L_Wait_Sleep:
        NOP
        INX
        BNE  L_Wait_Sleep
```

13. Timer

Timer provides the periodical interrupt or flag to CPU. The clock source of timer could be either CPU clock or horizontal scan line. Writing C_TMR_SETH would reload the timer.

C_TMR_SETL and C_TMR_SETH is the period of the timer, the formula of the timer is

Timer period = (65536 - Timer_PreLoad) / Timer Clock frequency

That is

Timer_PreLoad = 65536 – (Timer period * Timer Clock frequency)

其中

Timer_PreLoad = {TIMER_SETH, TIMER_SETL}

Timer clock frequency is related to the TMR_SR_SEL and TV system, as listed in the following table

	TMR_SR_SEL	Timer Clock frequency
NTSC	1	15.746 KHz
	0	10.7386 MHz
PAL	1	15.602 KHz
	0	10.6407 MHz

Please note that C_TMR_SETH should be written after C_TMR_SETL. Programmer could read the current value of timer through the C_TMR_SETL and C_TMR_SETH.

	TMR_EN = 0	TMR_EN = 1
TMR_IRQ_EN = 0	Timer OFF	Timer ON No TMR_IRQ Get timer status from C_TMR_STATUS
TMR_IRQ_EN = 1	Timer OFF	Timer ON TMR_IRQ enabled Get timer status from C_TMR_STATUS

Example of Initial TIMER:

```
LDA     #TMR_L
STA     C_TMR_SETL    ; Set Timer Period
LDA     #TMR_H
STA     C_TMR_SETH    ; Set Timer Period
STA     C_TMR_CLR     ; Clear Timer Status Flag
LDA     #$3
STA     C_TMR_SET     ; Enable TIMER / Enable IRQ / Set clock source
CLI                   ;
```

Example of TIMER IRQ Service Routine :

```
STA     $2103         ; Clear Timer IRQ Flag
:
```

14. Random Number Generator

There is a 15-bits pseudo-random number generator in VT268. A 15-bits non-zero seed number is required to write into the C_RANDOM_L and C_RANDOM_H in the initial sequence. The random number would be valid by reading 0x2108 and $2109. Every time a new seed number is written, a new pseudo random number sequence would be generated. Please note that it's not necessary to write the seed number every time when you are going to access the random number. Only in the initial sequence, or changing the random sequence, you have to update the send number.

Random number generator Initialization:

```
        LDA     #$55
        STA     $2108
        LDA     #$AA
        STA     $2109       ; Write the non-zero value into RANDOM_SEED
        :
```

Get random number:

```
        LDA  C_RANDOM_L
        LDA  C_RANDOM_H
```

15. Cartridge Combination

VT268 provides the cartridge combination function for programmer to integrate the small contents into a single one. The unit of the combination is 64 Kbytes. The physical address after the cartridge combination circuit is

ADDRESS(NEW) = ADDRESS(OLD) + (CARTRIDGE_OFFSET << 16).

All of the CPU PROGRAM, CPU DATA and graphic data are transferred to new address. Please note that the CARTRIDGE_OFFSET is written only when the PC is in the internal memory area (0x000000 ~ 0x007FFF).

16. Pulse-Width-Modulation (PWM) Control

VT268 provides three PWM outputs for the LED applications. These PWM signals output through the pins, UIOG4, UIOG5 and UIOG6, and they are all non-overlapped. Each PWM signal duty is controlled independently.

PWM_CLOCK_SEL	PWM clock (PWM_CLK)	
	NTSC	PAL
0	10.7368 MHz	10.6407 MHz
1	2.6842 MHz	2.6602 MHz
2	335.525 KHz	332.522 KHz
3	41.94 KHz	41.57 KHz
4*	32768 Hz	32768 Hz
5*	8192 Hz	8192 Hz
6*	1024 Hz	1024 Hz
7*	4 Hz	4 Hz

Pulse Width = 1/(PWM clock),

PWM R DUTY = 1/(PWM_R_PERIOD * 8)

PWM G DUTY = 1/(PWM_G_PERIOD * 8)

PWM B DUTY = 1/(PWM_B_PERIOD * 8)

	Description
PWM_EN = 0	PWM disabled
PWM_EN = 1	PWM enabled

17 Analog-To-Digital Converter(ADC)

There are 8 timing division multiplex ADC ports. One is dedicated for battery detection and one is for microphone application and the others are general purpose ADC ports.

17.1 Battery Detection Port

UIOG0 in VT268 is dedicated for the battery detection. Even when the power of VT268 is off and UIOG0 is connected to battery, it would not have power leakage problem. Please note that this port is not suitable for the general purpose ADC application which may lead to the sample distortion.

Initialization:

```
        LDA  C_UIOG_DIR          ; Set UIOG0 analog mode
        AND  #$FE                ;
        STA  C_UIOG_DIR          ;
        LDA  C_UIOG_ATTR         ;
        ORA  #$01                ;
        STA  C_UIOG_ATTR         ;
        LDA  C_UIOG_DATA         ;
        ORA  #$01                ;
        STA  C_UIOG_DATA         ;
        LDA  C_ADC_SETTING1      ; Set ADC
        ORA  #$06                ;
        STA  C_ADC_SETTING1      ;
        LDA  C_ADC_SETTING3      ;
        ORA  #$08                ;
        AND  #$F8                ;
        STA  C_ADC_SETTING3      ;
        LDA  C_ADC_ SETIING7     ;
        ORA  #$01                ;
        STA  C_ADC_ SETIING7     ;
```

Get ADC value:

```
        LDA  C_ADC_ SETIING7     ; Trigger ADC
        ORA  #$04                ;
        STA  C_ADC_ SETIING7     ;
        :
L_wait_ADC_ready:
        LDA  C_ADC_ SETIING7     ; Is ADC translate complete
        AND  #$04                ;
        BNE  L_wait_ADC_ready    ;
        LDA  C_ADC_DATAH         ; Translate complete, Get data high byte
```

```
        STA  ADC_DATA_H          ;
        LDA  C_ADC_DATAL         ; Get data low byte
        STA  ADC_DATA_L
```

17.2 Microphone port

UIOG7 is the microphone port in VT268 which also built-in the microphone bias circuit to reduce the complexity of external application circuit. Programmer could write the microphone Auto-Gain-Control (AGC) algorithm to control the Programmable Gain Amplifier, PGAG in C_ADC_SETTING2.

Initialization:

```
        LDA  C_UIOG_DIR          ; Set UIOG7 to analog mode
        AND  #$7F                ;
        STA  C_UIOG_DIR          ;
        LDA  C_UIOG_ATTR         ;
        ORA  #$80                ;
        STA  C_UIOG_ATTR         ;
        LDA  C_UIOG_DATA         ;
        ORA  #$80                ;
        STA  C_UIOG_DATA         ;
        LDA  C_ADC_SETTING1      ; Set ADC
        ORA  #$06                ;
        STA  C_ADC_SETTING1      ;
        LDA  C_ADC_SETTING3      ;
        ORA  #$08                ;
        AND  #$F8                ;
        ORA  #$04                ;
        STA  C_ADC_SETTING3      ;
        LDA  C_ADC_SETIING7      ;
        ORA  #$01                ;
        STA  C_ADC_SETIING7      ;
```

Get microphone data:

```
        LDA  C_ADC_SETIING7      ; Trigger ADC sample
        ORA  #$04                ;
        STA  C_ADC_SETIING7      ;
        :
L_wait_ADC_ready:
        LDA  C_ADC_ SETIING7     ; Is ADC complete translate
        AND  #$04                ;
        BNE  L_wait_ADC_ready    ;
```

```
LDA  C_ADC_DATAH      ; Get microphone high byte data
STA  ADC_DATA_H       ;
LDA  C_ADC_DATAL      ; Get microphone low byte data
STA  ADC_DATA_L
```

17.3 General purpose ADC port

UIOG1, UIOG2, UIOG3, UIOG4, UIOG5 and UIOG6 are general purpose ADC port in VT268.

Initilization:

```
LDA  C_UIOG_DIR       ; Set UIOG1 to analog mode
AND  #$FD             ;
STA  C_UIOG_DIR       ;
LDA  C_UIOG_ATTR      ;
ORA  #$02             ;
STA  C_UIOG_ATTR      ;
LDA  C_UIOG_DATA      ;
ORA  #$02             ;
STA  C_UIOG_DATA      ;
LDA  C_ADC_SETTING1   ; Set ADC
ORA  #$06             ;
STA  C_ADC_SETTING1   ;
LDA  C_ADC_SETTING3   ;
ORA  #$08             ;
AND  #$F8             ;
ORA  #$01             ;
STA  C_ADC_SETTING3   ;
LDA  C_ADC_ SETIING7  ;
ORA  #$01             ;
STA  C_ADC_ SETIING7  ;
```

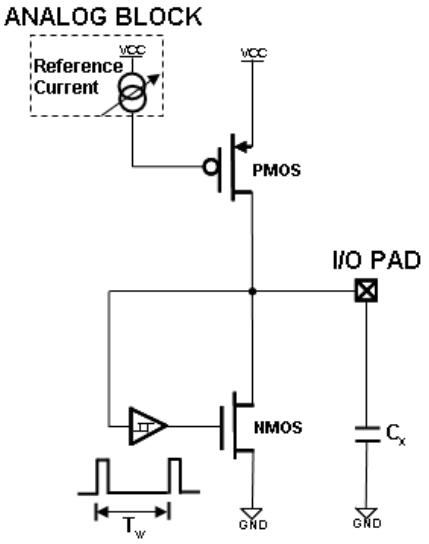
Get ADC data:

```
LDA  C_ADC_ SETIING7  ; Trigger ADC
ORA  #$04             ;
STA  C_ADC_ SETIING7  ;
:
L_wait_ADC_ready:
LDA  C_ADC_ SETIING7  ; Does ADC translate complete
AND  #$04             ;
BNE  L_wait_ADC_ready ;
LDA  C_ADC_DATAH      ; Get ADC high byte data
```

```
STA  ADC_DATA_H          ;
LDA  C_ADC_DATAL         ; Get ADC high byte data
STA  ADC_DATA_L
```

18 Capacitance Touched Switch

Capacitance switch use current to detect capacitance change on the external pad to check whether the pad is touched. There are 16 capacitance switch on UIOF and UIOG. The advantage of capacitance switch is that it's better reliability than traditional push switch, but requires more on the PCB layout and switch cover material. As to the capacitance switch layout note, please contact with VRT FAE. Following diagram is the basic block diagram of circuit in the VT268 capacitance switch detection. VT268 provides the constant current source, controlled by CCURSEL, to charge and discharge the external pad capacitance(Cx). Different Cx would have different charge and discharge time(C_CAP_DATAL and C_CAP_DATAH). Programmer could use this to detect the pad is touched or not.



CAPPD	Touch switch operation mode
0	Enabled
1	Disabled

CAPBSEL	Touch Switch Counter Setting
0	Use 8-bit counter, Operates at 21.47MHz in NTSC and 21.28MHz in PAL. Maximum counter number FFH, Counter value read through C_CAP_DATAL.
1	Use 14-bit counter, Operates at 21.47MHz in NTSC and 21.28MHz in PAL. Maximum counter number 3FFH, Counter value read through {C_CAP_DATAH , C_CAP_DATAL}

CCURSEL	Capacitance charge current
1	Current = CURMIN
2	Current = CURMIN x 2
:	:
30	CURMAX = CURMIN x 30
31	CURMAX = CURMIN x 31

Note : Smaller current is better for the detail measurement but requires longer measure time.

CSWITCH				Touch Switch Port Select
0	0	0	0	UIOF0
0	0	0	1	UIOF1
0	0	1	0	UIOF2
0	0	1	1	UIOF3
0	1	0	0	UIOF4
0	1	0	1	UIOF5
0	1	1	0	UIOF6
0	1	1	1	UIOF7
1	0	0	0	UIOG0
1	0	0	1	UIOG1
1	0	1	0	UIOG2
1	0	1	1	UIOG3
1	1	0	0	UIOG4
1	1	0	1	UIOG5
1	1	1	0	UIOG6
1	1	1	1	UIOG7

*Note:

When BSEL = 0, CSWITCH would increment after C_CAP_DATA_L is read.

When BSEL = 1, CSWITCH would increment after C_CAP_DATA_H is read.

CAPRDY	Touch Switch Status
0	Measuring
1	Measurement complete

Initialization:

```
LDA C_UIOF_ENB          ; Set UIOF[3..0] as touch switch
AND  #$F0
STA  C_UIOF_ENB
LDA  C_UIOF_DIR
```

```
        ORA  #$0F
        STA  C_UIOF_DIR
        LDA  C_UIOF_ATTR
        ORA  #$0F
        STA  C_UIOF_ATTR
        LDA  #$01                ; Set counter and current
        STA  C_CAP_SETTING1      ;
        LDA  #$00                ; Start from UIOF0
        STA  C_CAP_SETTING2      ;
```

Measure data access:

```
L_WAIT0:
    LDA   C_CAP_STATUSCAPRDY  ; Check measure status
    BEQ   L_WAIT0             ;
    LDA   C_CAP_DATAL         ; read measured data
    STA   CAP_DATA            ;
    LDA   C_CAP_SETTING2      ; Check UIOF3 is done
    CMP  #$3                  ;
    BNE   L_EXIT              ;
    LDA   #$0                 ; If UIOF3is done, change to UIOF0
    STA   C_CAP_SETTING2      ;
L_EXIT:                       ;
    :
```

19 External chip select Control

There are up to three chip select signals in VT26, they are ROMCSB, RAMCSB and ROMCSB2 in UIOC7.

CSMODE	ROMCSB	RAMCSB	UIOC7/ROMCSB2
0	128Mbits $000000 ~ $3FFFFF	X	X
1	64Mbits $000000 ~ $1FFFFF	64Mbits $200000 ~ $3FFFFF	X
2	32Mbits $000000 ~ $0FFFFF	32Mbits $100000 ~ $1FFFFF	64Mbits $200000 ~ $3FFFFF
3	32Mbits $000000 ~ $0FFFFF	32Mbits $100000 ~ $1FFFFF	64Mbits $200000 ~ $3FFFFF

Note : Above address is for VT268 address bus(16 bits address bus).

Appendix 1 : Register Table

Graphic Setting

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002000	P_CONTROL1	R/W	0						CAPTURE	SLAVE	NMIEN
0x002001	P_CONTROL2	W	0					EXT_CLK_DIV		VCOMIO	RGBDAC
0x002001	P_CONTROL2	R	0				VBLANK	EXT_CLK_DIV		VCOMIO	RGBDAC
0x00200E	P_CONTROL3	R/W	0	OVLP2EN	OVLP1EN	BLD2EN	BLD1EN	---	BK3PALSEL	BK2PALSEL	BK1PALSEL
0x00200F	P_CONTROL4	R/W	0			BK3OUTSEL		BK2OUTSEL		BK1OUTSEL	

Background1 Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002010	P_BK1_X	R/W	0	BK1_X [7:0]							
0x002011	P_BK1_Y	R/W	0	BK1_Y [7:0]							
0x002012	P_BK1_SETTING1	R/W	0	BK1_VRAM_BANK			BK1HC	BK1VSCRL	BK1HSCRL	BK1_Y[8]	BK1_X[8]
0x002013	P_BK1_SETTING2	R/W	0	BK1_EN	--	BK1DEPTH		BK1COLOR		BK1LINE	BK1SIZE
0x00201C	P_BK1_SEG_L	R/W	0	BK1SEGMENT[7:0]							
0x00201D	P_BK1_SEG_H	R/W	0					BK1SEGMENT[11:8]			

Background2 Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002014	P_BK2_X	R/W	0	BK2_X [7:0]							
0x002015	P_BK2_Y	R/W	0	BK2_Y [7:0]							

0x002016	P_BK2_SETTING1	R/W	0	BK2_VRAM_BANK			--	BK2VSCRL	BK2HSCRL	BK2_Y[8]	BK2_X[8]
0x002017	P_BK2_SETTING2	R/W	0	BK2_EN	--	BK2DEPTH		BK2COLOR		BK2LINE	BK2SIZE
0x00201E	P_BK2_SEG_L	R/W	0	BK2SEGMENT[7:0]							
0x00201F	P_BK2_SEG_H	R/W	0					BK2SEGMENT[11:8]			

Background3 Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002002	P_BK3_X	R/W	0	BK3_X [7:0]							
0x002003	P_BK3_Y	R/W	0	BK3_Y [7:0]							
0x002004	P_BK3_SETTING1	R/W	0	BK3_VRAM_BANK			--	BK3VSCRL	BK3HSCRL	BK3_Y[8]	BK3_X[8]
0x002005	P_BK3_SETTING2	R/W	0	BK3_EN	--	BK3DEPTH		BK3COLOR		BK3LINE	--
0x002006	P_BK3_SEG_L	R/W	0	BK3SEGMENT[7:0]							
0x002007	P_BK3_SEG_H	R/W	0					BK3SEGMENT[11:8]			

Sprite Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002018	P_SP_SETTING	R/W	0							SP512M	SPEN
0x00201A	P_SP_SEG_L	R/W	0	SPSEGMENT[7:0]							
0x00201B	P_SP_SEG_H	R/W	0					SPSEGMENT[11:8]			

Line Scroll Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0

0x00200C	P_BK_LNSCROLL	R/W	0	BK3LNEN	BK2LNEN	BK1LNEN	BKLNSCROLLBANK				

Vertical Scaling Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002019	P_VGAIN_ENABLE	R/W	0					SPVGAIN	BK3VGAIN	BK2VGAIN	BK1VGAIN
0x002060	P_VGAIN_LINE	R/W	0	VGAINSTARTLINE							
0x002061	P_VGAIN_INIT	R/W	0	VGAININITIAL							
0x002062	P_VGAIN_STEP	R/W	0	VGAINSTEP							

XY IRQ

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002020	P_XY_IRQ	W	0			EYE_THRESHOLD				XYIRQCLR	XYIRQEN
0x002020	P_XY_IRQ	R	0			EYE_THRESHOLD				XYIRQSTUS	XYIRQEN
0x002008	P_XY_IRQ_V	R/W	0	P_STROBE_Y							
0x002009	P_XY_IRQ_H	R/W	0	P_STROBE_X							

Eye Function

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002020	P_XY_IRQ	R/W	0			EYE_THRESHOLD				XYIRQCLR	XYIRQEN
0x002021	P_EYE_SET	R/W	0			EYEN	P_EYE_PRAM_BANK				
0x002022	P_EYE_STATUS_L	R		P_EYE_STATUS_L							
0x002023	P_EYE_STATUS_H	R		P_EYE_STATUS_H							

CCIR Node Strobe

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002008	P_STROBE_Y	R/W	0	P_STROBE_Y							
0x002009	P_STROBE_X	R/W	0	P_STROBE_X							
0x00200A	P_STROBE_DATAL	R	0	P_STROBE_DATAL							
0x00200B	P_STROBE_DATAH	R	0	P_STROBE_DATAH							

Light Gun Function

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002024	P_LIGHTGUN_CLR	W	0	P_LIGHTGUN_CLR							
0x002024	P_LIGHTGUN1_Y	R	0	P_LIGHTGUN1_Y							
0x002025	P_LIGHTGUN1_X	R	0	P_LIGHTGUN1_X							
0x002026	P_LIGHTGUN2_Y	R	0	P_LIGHTGUN2_Y							
0x002027	P_LIGHTGUN2_X	R	0	P_LIGHTGUN2_X							

CCIR Protocol Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002028	P_CCIR_V	R/W	0			CCIR_V					
0x002029	P_CCIR_H	R/W	0		CCIR_H						
0x00202A	P_CCIR_SETTING1	R/W	0								
0x00202B	P_CCIR_SETTING2	R/W	0								

Output Device Source Selection

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002056	P_OUTSEL	R/W	0						LCD2SEL	LCD1SEL	TVSEL

Current Horizontal Scan Line Number

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002058	P_HSCALNUM	R	-	HSCANLINENUM							

Special Effect Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002050	P_EFCT_CONTROL1	R/W	0		DIG1EN	FADEALL	FADEINOUT	EFF1XOR	EFF1SEL		EFF1SRC
0x002055	P_EFCT_CONTROL2	R/W	0	HSCALEN	DIG2EN	FADEALL	FADE2NOUT	EFF2XOR	EFF2SEL		EFF2SRC
0x002051	P_EFCT_FADE1L	R/W	0	FADE1_OFFSET[7:0]							
0x002052	P_EFCT_FADE1H	R/W	0		FADE2_OFFSET[14:8]						
0x002053	P_EFCT_FADE2L	R/W	0	FADE2_OFFSET[7:0]							
0x002054	P_EFCT_FADE2H	R/W	0		FADE2_OFFSET[14:8]						

LCD1 Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002040	P_LCD1_DOTN	R/W	0								
0x002041	P_LCD1_DOTP	R/W	0								

0x002042	P_LCD1_LNST	R/W	0								
0x002043	P_LCD1_LNN	R/W	0								
0x002044	P_LCD1_LNP	R/W	0								
0x002045	P_LCD1_SETTING1	R/W	0								
0x002046	P_LCD1_VS	R/W	0								
0x002047	P_LCD1_HS	R/W	0								
0x002048	P_LCD1_SETTING2	R/W	0								
0x002049	P_LCD1_SETTING3	R/W	0								

LCD2 Control

Address	Symbol	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x00204A	P_LCD2_DOTN	R/W	0								
0x00204B	P_LCD1_SETTING1	R/W	0								
0x00204C	P_LCD2_VS	R/W	0								
0x00204D	P_LCD2_HS	R/W	0								
0x00204E	P_LCD2_SETTING2	R/W	0								
0x00204F	P_LCD2_SETTING3	R/W	0								

Timer Register Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002100	C_TMR_L	W	0	TIMER_SETL							
0x002100	C_TMR_L	R	0	TIMER_L							
0x002101	C_TMR_H	W	0	TIMER_SETH							
0x002101	C_TMR_H	R	0	TIMER_H							
0x002102	C_TMR_SETTING	R/W	0						TMR_SR_SEL	TMR_IRQ_EN	TMR_EN
0x002103	C_TMR_CLR	W	0								TMR_CLR
0x002103	C_TMR_STATUS	R	0								TMR_DONE

Random Generator

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002108	C_RANDOM_L	R/W	0	RANDOM_L							
0x002109	C_RANDOM_H	R/W	0		RANDOM_H						

UART Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002118	C_UART_SETTING	R/W	0	TX_Dma_En	RX_Dma_En	Carrier_En	TX_Irq_En	RX_Irq_En	Parity_En	Parity_Sel	9bMode
0x002119	C_UART_IRQ_CLR	W	0	UARTEN			TX_Irq_Clr	RX_Irq_Clr			
0x002119	C_UART_STATUS	R		UARTEN	RX_Bit9	RX_Err	TX_Irq	RX_Irq	RX_Par_Err	TX_Status	RX_Status
0x00211A	C_UART_BAUD_L	R/W	0	UART_BAUD_L							

0x00211B	C_UART_BAUD_H	R/W	0	UART_BAUD_H
0x00211C	C_UART_CARRI	R/W	0	UART_CARRI
0x00211D	C_UART_TX_DATA	R/W	0	UART_TX_DATA
0x00211E	C_UART_RX_DATA	R/W	0	UART_RX_DATA

IIC Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002180	C_I2C_ID	R/W	0	I2C_ID							
0x002181	C_I2C_CONTROL	R/W	0						D16Mode	CLKSEL	
0x002182	C_I2C_ADDR_L	R/W	0	I2C_ADDR_L							
0x002183	C_I2C_ADDR_H	R/W	0	I2C_ADDR_H							
0x002184	C_I2C_W_DATA	W	0	I2C_W_DATA							
0x002184	C_I2C_R_DATA	R	0	I2C_R_DATA							
0x002185	C_I2C_STATUS	R	0							READY	ACKERR

SPI Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002188	C_SPI_SETTING	R/W	0	D16Mode	RSTSPI	IRQEN	Master	PHASE	POLAR	CLKSEL	
0x002189	C_SPI_TX_DATA	R/W	0	SPI_TX_DATA							
0x00218A	C_SPI_SET	W	0							RXDmaEn	TXDmaEn
0x00218A	C_SPI_SET	R	0					NewRec	READY	RXDmaEn	TXDmaEn
0x00218B	C_SPI_RX_DATA	R/W	0	SPI_RX_DATA							

Collision Detect

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002190	C_COL_X1L	R/W	0	COL_X1L							
0x002191	C_COL_X1H	R/W	0	COL_X1H							
0x002192	C_COL_Y1L	R/W	0	COL_Y1L							
0x002193	C_COL_Y1H	R/W	0	COL_Y1H							
0x002194	C_COL_X2L	R/W	0	COL_X2L							
0x002195	C_COL_X2H	R/W	0	COL_X2H							
0x002196	C_COL_Y2L	R/W	0	COL_Y2L							
0x002197	C_COL_Y2H	R/W	0	COL_Y2H							
0x002198	C_COL_SETTING	R	0							BUSY	Collision

PWM Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x00210C	C_PWM_1	R/W	0	PWMPERIOD_1							
0x00210D	C_PWM_2	R/W	0	PWMPERIOD_2							
0x00210E	C_PWM_3	R/W	0	PWMPERIOD_3							
0x00210F	C_PWM_SETTING	R/W	0			CLKSEL			PWM3EN	PWM2EN	PWM1EN

Cartridge Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0

0x00210A	C_CART_SET	R/W	0	CARTRIDGE_SET

SCPU IRQ Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002107	C_SCPUIRQ_SET	R/W	0								SCPUIRQ
0x00210B	C_SCPUIRQ_CLR	W	--	SCPUIRQCLR							

DMA Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002120	C_DMA_LUT_AL	R/W	0	DMA_LUT_ADDRESS[7:0]							
0x002121	C_DMA_LUT_AH	R/W	0			DMA_LUT_ADDRESS[13:8]					
0x002122	C_DMA_DT_AL	R/W	0	DMA_DT_ADDRESS[7:0]							
0x002123	C_DMA_DT_AM	R/W	0	DMA_DT_ADDRESS[15:8]							
0x002124	C_DMA_SR_AL	R/W	0	DMA_SR_ADDRESS[7:0]							
0x002125	C_DMA_SR_AM	R/W	0	DMA_SR_ADDRESS[15:8]							
0x002126	C_DMA_SR_AH	R/W	0	DMA_SR_ADDRESS[23:16]							
0x002127	C_DMA_DT_AH	R/W	0	DMA_DT_ADDRESS[23:16]							
0x002128	C_DMA_NUM	W	0	DMA_NUMBER							
0x002129	C_DMA_LUT_NUM	W	0	DMA_LUT_NUMBER							

Multiplier Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0

0x002130	C_MUL_IN1_L	W	0	MULTIPLIER_IN1[7:0]
0x002131	C_MUL_IN1_H	W	0	MULTIPLIER_IN1[15:8]
0x002134	C_MUL_IN2_L	W	0	MULTIPLIER_IN2[7:0]
0x002135	C_MUL_IN2_H	W	0	MULTIPLIER_IN2[15:8]
0x002138	C_MUL_OUT_L	R		MULTIPLIER_OUT[7:0]
0x002139	C_MUL_OUT_M1	R		MULTIPLIER_OUT[15:8]
0x00213A	C_MUL_OUT_M2	R		MULTIPLIER_OUT[23:16]
0x00213B	C_MUL_OUT_H	R		MULTIPLIER_OUT[31:24]

Division Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002130	C_DIV_IN1_L	W	0	DIVIDER_IN1[7:0]							
0x002131	C_DIV_IN1_M1	W	0	DIVIDER _IN1[15:8]							
0x002132	C_DIV_IN1_M2	W	0	DIVIDER_IN1[23:16]							
0x002133	C_DIV_IN1_H	W	0	DIVIDER _IN1[31:24]							
0x002136	C_ DIV _IN2_L	W	0	DIVIDER _IN2[7:0]							
0x002137	C_ DIV _IN2_H	W	0	DIVIDER _IN2[15:8]							
0x002138	C_ DIV _OUT1_L	R		DIVIDER _OUT1[7:0]							
0x002139	C_ DIV _OUT1_M1	R		DIVIDER _OUT1[15:8]							
0x00213A	C_ DIV _OUT1_M2	R		DIVIDER _OUT1[23:16]							
0x00213B	C_ DIV _OUT1_H	R		DIVIDER _OUT1[31:24]							
0x00213C	C_ DIV _OUT2_L	R		DIVIDER _OUT2[7:0]							

0x00213D	C_ DIV _OUT2_H	R		DIVIDER _OUT2[15:8]

UIOA Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002140	C_UIOA_DIR	R/W	0	UIOA_DIRECTION							
0x002141	C_UIOA_ATTR	R/W	0	UIOA_ATTRIBUTE							
0x002142	C_UIOA_DATA	R/W	FF	UIOA_DATA							
0x002143	C_UIOA_BUF	W	0	UIOA_DATA							
0x002143	C_UIOA_BUF	R	0	UIOA_PIN_DATA							
0x00215C	C_UIOA_ENB	R/W	0	UIOA_ENB							

UIOB Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002144	C_UIOB_DIR	R/W	0	UIOB _DIRECTION							
0x002145	C_ UIOB _ATTR	R/W	0	UIOB _ATTRIBUTE							
0x002146	C_ UIOB _DATA	R/W	FF	UIOB _DATA							
0x002147	C_ UIOB _BUF	W	0	UIOB _DATA							
0x002148	C_ UIOB _BUF	R	0	UIOB _PIN_DATA							
0x00215D	C_UIOB_ENB	R/W	0	UIOB_ENB							

UIOC Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002148	C_ UIOC _DIR	R/W	0	UIOC _DIRECTION							

0x002149	C_ UIOC _ATTR	R/W	0	UIOC _ATTRIBUTE							
0x00214A	C_ UIOC _DATA	R/W	FF	UIOC _DATA							
0x00214B	C_ UIOC _BUF	W	0	UIOC _DATA							
0x00214B	C_ UIOC _BUF	R	0	UIOC _PIN_DATA							
0x00215E	C_UIOC_ENB	R/W	0	UIOC_ENB							
0x002165	C_UIOC_SEL	R/W	0				UIOC_SEL				

UIOD Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x00214C	C_ UIOD _DIR	R/W	0	UIOD _DIRECTION							
0x00214D	C_ UIOD _ATTR	R/W	0	UIOD _ATTRIBUTE							
0x00214E	C_ UIOD _DATA	R/W	FF	UIOD _DATA							
0x00214F	C_ UIOD _BUF	W	0	UIOD _DATA							
0x00214F	C_ UIOD _BUF	R	0	UIOD _PIN_DATA							
0x00215F	C_UIOD_ENB	R/W	0	UIOD_ENB							

UIOE Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002150	C_ UIOE _DIR	R/W	0	UIOE _DIRECTION							
0x002151	C_ UIOE _ATTR	R/W	0	UIOE _ATTRIBUTE							
0x002152	C_ UIOE _DATA	R/W	FF	UIOE _DATA							
0x002153	C_ UIOE _BUF	W	0	UIOE _DATA							

0x002153	C_ UIOE _BUF	R	0	UIOE _PIN_DATA
0x002160	C_UIOE_ENB	R/W	0	UIOE_ENB

UIOF Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002154	C_ UIOF _DIR	R/W	0	UIOF _DIRECTION							
0x002155	C_ UIOF _ATTR	R/W	0	UIOF _ATTRIBUTE							
0x002156	C_ UIOF _DATA	R/W	0	UIOF _DATA							
0x002157	C_ UIOF _BUF	W	0	UIOF _DATA							
0x002157	C_ UIOF _BUF	R	0	UIOF _PIN_DATA							
0x002161	C_UIOF_ENB	R/W	0	UIOF_ENB							

UIOG Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002158	C_ UIOG _DIR	R/W	0	UIOG _DIRECTION							
0x002159	C_ UIOG _ATTR	R/W	0	UIOG _ATTRIBUTE							
0x00215A	C_ UIOG _DATA	R/W	0	UIOG _DATA							
0x00215B	C_ UIOG _BUF	W	0	UIOG _DATA							
0x00215B	C_ UIOG _BUF	R	0	UIOG _PIN_DATA							
0x002162	C_UIOG_ENB	R/W	0	UIOG_ENB							

ADC Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x0021B0	C_ADC_SETTING1	R/W	0						ENVREF	VREFSM	ENMICBIAS
0x0021B1	C_ADC_SETTING2	R/W	56	ENMIC	ENBST	ENZCD	PGAG				
0x0021B2	C_ADC_SETTING3	R/W	07			ENDAR	ENDAL	ENAD	ADCS		
0x0021B3	C_ADC_SETTING4	R/W	0				DARG				
0x0021B4	C_ADC_SETTING5	R/W	0				DALG				
0x0021B5	C_ADC_SETTING6	R/W	0			TMAD	TMDA	TADDA			
0x0021B7	C_ADC_SETTING7	W	0						ADCTRIG	ADCSEL	ADCLKEN
0x0021B7	C_ADC_SETTING7	R	--						ADCRDY	ADCSEL	ADCLKEN
0x0021B8	C_ADC_DATAL	R	--					ADC_DATA[3:0]			
0x0021B9	C_ADC_DATAH	R	--	ADC_DATA[11:4]							

ALARM Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002170	C_ALARM_HR	R/W	0				ALARM_HOUR				
0x002171	C_ALARM_MN	R/W	0			ALARM_MINUTE					
0x002172	C_ALARM_SC	R/W	0			ALARM_SECOND					

Real-Time Clock Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002174	C_RTC_HR	W	0				RTC_HOUR_INIT				
0x002174	C_RTC_HR	R	--				RTC_HOUR				

0x002175	C_ RTC_MN	W	0			RTC _MINUTE_INIT					
0x002175	C_ RTC_MN	R	--			RTC _MINUTE					
0x002176	C_ RTC_SC	W	0			RTC _SECOND_INIT					
0x002176	C_ RTC_SC	R	--			RTC _SECOND					
0x002178	C_ RTC_LOAD	W	--	RTC_LOAD_INIT							
0x00217A	C_ RTC_ENABLE	R/W									RTCEN
0x00217B	C_RTL_IRQ	W						ALMCLR	WAKECLR	ALMIRQEN	WAKEIRQEN
0x00217B	C_RTL_IRQ	R						ALMSTAT	WAKESTAT	ALMIRQEN	WAKEIRQEN
0x0021BA	C_RTL_PAD_CTRL	R/W					XPLDEN	XTALSTR	XTALEN	BIASEN	CLKSEC

External IRQ Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002164	C_ EXT_IRQ	W	0							EXTIRQCLR	EXTIRQEN
0x002164	C_ EXT_IRQ	R	0								EXTIRQEN

Sleep Mode Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002173	C_ ENTER_SLEEP	W	0	ENTERSLEEPMODE							

Wake up Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002163	C_ KEY_WAKE_SEL	R/W	0							IOGWAKE	IOCWAKE

0x002179	C_ KEY_WAKE_MD	R/W	0				TMRWAKE	KEYWAKE	ALMWAKE	WAKECNTSEL	
0x002177	C_WAKE_PERD	R/W	0	WAKEIPERIOD							

SCPU PBR Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x0021B6	C_ SCPU_PBR	R/W	0	SCPU_PBR							

System Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x002104	C_SYS_SETTING1	R/W	0				BOOTMOD	CSMODE		BUSMODE	0
0x002105	C_SYS_SETTING2	R/W	0			SCPURN	SCPUON	---	CCIRON	TVON	LCDON
0x002106	C_SYS_SETTING3	R/W	0						LVRPD	LCDACEN	VDACEN

Bridge Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x0021A0	C_BRDG1_SETTING	R/W	0	BR1RST	BR1EN	BUFFER1SIZE			BR1SRCSEL		
0x0021A1	C_BRDG1_ADDRL	R/W	0	BR1_ADDR[7:0]							
0x0021A2	C_BRDG1_ADDRH	R/W	0			BR1_ADDR[13:8]					
0x0021A3	C_BRDG1_WRNUM	R/W	0	BR1_WRITE_NUMBER							
0x0021A4	C_BRDG2_SETTING	R/W	0	BR2RST	BR2EN	BUFFER2SIZE			BR2SRCSEL		
0x0021A5	C_BRDG2_ADDRL	R/W	0	BR2_ADDR[7:0]							
0x0021A6	C_BRDG2_ADDRH	R/W	0			BR2_ADDR[13:8]					

0x0021A7	C_BRDG2_WRNUM	R/W	0	BR2_WRITE_NUMBER

CAPSWITCH Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x0021E0	C_ CAP_SETTING1	R/W	0x40		CAPPD	CAPBSEL	CCURSEL				
0x0021E1	C_ CAP_SETTING2	R/W	0					CSWITCHSEL			
0x0021E2	C_CAP_DATAH	R		CAPDATA[15:8]							
0x0021E3	C_CAP_DATAL	R		CAPDATA[7:0]							
0x0021E4	C_CAP_STATUS	R									CAPRDY

IRRC Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x0021C0	C_ IRRC_SETTING1	R/W	0x00	CARR[8]			SAMPLE			DATASIZE	
0x0021C1	C_ IRRC_SETTING2	R/W	0x00	CARR[7:0]							
0x0021C2	C_ IRRC_SETTING3	W	0x00	IRRCBUSY	IRRCLIMIT	IRRCERR		IRRCEND	IRRCSTOP	IRRCSTX	IRRCRX
0x0021C2	C_ IRRC_SETTING3	R	0x00								
0x0021C3	C_ IRRC_ADDRL	R/W	0x00	RAM_ADDRL							
0x0021C4	C_ IRRC_ADDRH	R/W	0x00				RAM_ADDRH				

SD Control

Address	Label	R/W	Default	D7	D6	D5	D4	D3	D2	D1	D0
0x0021D0	C_ SD_SETTING1	R/W	0x00	FUNCSEL	SPEED			TIMEOUTREG		NWRNECCYCLE	

0x0021D1	C_ SD_SETTING2	W	0x00	SDSTART							
0x0021D1	C_ SD_SETTING2	R	--	SDTXDONE	TIMEOUT1	RESPERR1	RESPERR2	TIMEOUT2	TIMEOUT3	TIMEOUT4	TIMEOUT5
0x0021D2	C_ SD_CMD1	R/W	0x00	SDCMD[15:9]							
0x0021D3	C_ SD_CMD2	R/W	0x00	SDCMD[23:16]							
0x0021D4	C_ SD_CMD3	R/W	0x00	SDCMD[31:24]							
0x0021D5	C_ SD_CMD4	R/W	0x00	SDCMD[39:32]							
0x0021D6	C_ SD_CMD5	R/W	0x00	SDCMD[47:40]							
0x0021D7	C_ SD_CMD6	R/W	0x00	SDCMDRESP							
0x0021D8	C_ SD_SETTING3	R/W	0x00	SDDATARESP							
0x0021D9	C_ SD_SETTING4	R/W	0x00	SDCHECKBUSY							