VT1682 程序設計指南 V1.10

2. 修訂歷史

Revision	Date	Remark
V1.0	2005/08/	初始版
V1.1	2005/12/22	UART 狀態端口由$2119 移動到$211B. 刪除 CCIR 灰階拍攝和藍屏效果. 更改 8.2 ~ 8.6 IO 設置表格. 刪除休眠/甦醒部份. 刪除 Eye-Function IRQ. 刪除 MCU 界面部份. 更改讀取$2007 給卡通塊 RAM 和$2004 給 VRAM 更改 $2110~$2113 讀取端口. 刪除卡通塊縱向放大功能.
V1.2	2006/03/27	Page23:更改 $210B D7 TSYNEN 狀態
V1.3	2006/05/02	Page12:修正在 EXT2421 地址模式的錯誤. Page23:修正計時器公式的錯誤. Page29,30:修正 BK1/BK2 VRAM 管理. Page22 and 68: 修正 ALU 除法器的餘數.
V1.4	2006/06/01	Page18:修正 210D.D5
V1.5	2006/06/26	Page45:修正光槍的程序順序
V1.6	2006/12/26	修正打字錯誤 Page25 CSPU->SCPU,Page73:0X4121->0X212E
V1.7	2006/04/10	修正打字錯誤 page68 14.8.2 Divider ALU_Muti_operand-→ALU_Div_operand
V1.8	2007/10/1	Page 13: 增加 CPU Vector → System Reset 和 0x7FFFC, 0x7FFFD Page 53: 修改列 XIOE[0,1,2]的 IOEOE=0 欄位, INPUT floating → INPUT (12K ohms pull-down resistor) Page 56: 修改 XIOE0, 1, 2 的 OUTPUT 欄位, --- → VR, VG, VB Page 57: 修正 0x2127® → 0x2127(R)
V1.9	2007/10/30	Page 62: 修改"CS_Control"的範圍
V1.10	2009/01/19	Page 46: 取消 0x2004 和 0x2007 的讀取功能

3. 一般功能描述

VT1682 包括 主 CPU, 圖像處理器, 聲音的 CPU, 內部的 SRAM(8K 字節給程序用,4K 字節給圖像用),內部的 ROM(4K 字節)及 一些 I/O 控制裝置. VT1682 可以分為兩個系統,一個用於程序的,另一個用於影像的處理.

主 CPU(Main CPU)是整個程式系統的主要角色. 它可以對內部的隨機存儲器 PRAM 和外部的系統軟件存儲器(ROM 或是 Flash)進行尋址以取出需要的訊息進行運算處理. 系統軟件的存儲器(ROM 或是 Flash)被儲存程序命令, 程序指引和一些聲音資料.而 VT1682 內部的 8K 字節的程序動態隨機存儲器(PRAM)是第零頁 RAM, 儲存空間 及一些 CPU 的記憶體. 程式系統控制學習機的執行, 包括圖案, 語音, 及字幕. 也就是說 CPU 將控制視頻系統顯示指定的圖案.

圖像單元是影像系統的主要角色. 它能夠對內部的動態隨機存儲器(VRAM)和圖形塊(character)存儲器(ROM 或是 Flash) 進行尋址以取出需要的訊息進行運算處理後自動地顯示一些圖案. 除了內部的 PRAM 之外,VT1682 內部有另外的 4K 字節的 VRAM, 影像的動態隨機存儲器(VRAM)存儲許多指到圖形塊(Character)ROM 或是 Flash 內圖形的圖形序號. VRAM 儲存圖形序號,它可以在屏幕上作 2 層背景的顯示. 圖形塊(Character) ROM 儲存許多卡通塊圖形.

聲音 CPU(Sound CPU)和主 C PU 共用內部的 ROM 和 4K 字節的程序 SRAM.它有各自的 IO 和 ALU.它的操作速度比主 CPU 快 4 倍而且它適合於不同的應用.

3.1 特點(Feature)

系統(System)

- 工作電壓: 3.0~3.6 V
- 主 CPU: 6502 @5.3693MHz in NTSC and 5.3203MHz in PAL
- 內部的可隨意調整程序記憶體(ROM): 4K 字節
- 內部的主 CPU 的程序 RAM: 8K 字節 (4K 字節為專用的 RAM 和 4K 字節為共用的 RAM)
- 內部的 Video RAM: 4K 字節
- 記憶體直接尋址(DMA) Sprite RAM / VRAM / Program RAM / 外部的存儲器
- 單一的 16 位數據匯流排
- 掃瞄線 IRQ / 16-bits 定時器 IRQ / 外部的 IRQ
- 透過三條地址線(CSB)解碼可擴充外部的存儲器到 32M 字節.
- T.V. 信號輸出(NTSC, PAL, PAL-M, PAL-N)
- 擴充 5 IRQ 伺服入口
- 56 個 GPIO 端口, 40 個給主 CPU, 其他 16 個給 Sound CPU.

周邊(Peripheral)

- ADC: 8bits, 5 個時間分割複合(Times-Division-Multiplex)通道和聲音增大(Voice Gain)控制
- 4 階段低電壓檢查
- 主動裝置(Master)/從動裝置(Slave) SPI 界面
- UART 界面
- TFT LCD 界面.
- STN LCD 界面
- IIS 界面
- IIC 界面 (主動裝置(Master)方式)
- CCIR656/601 界面
- 加強 ALU, 16 乘 16 乘法器和 32 除 16 除法器

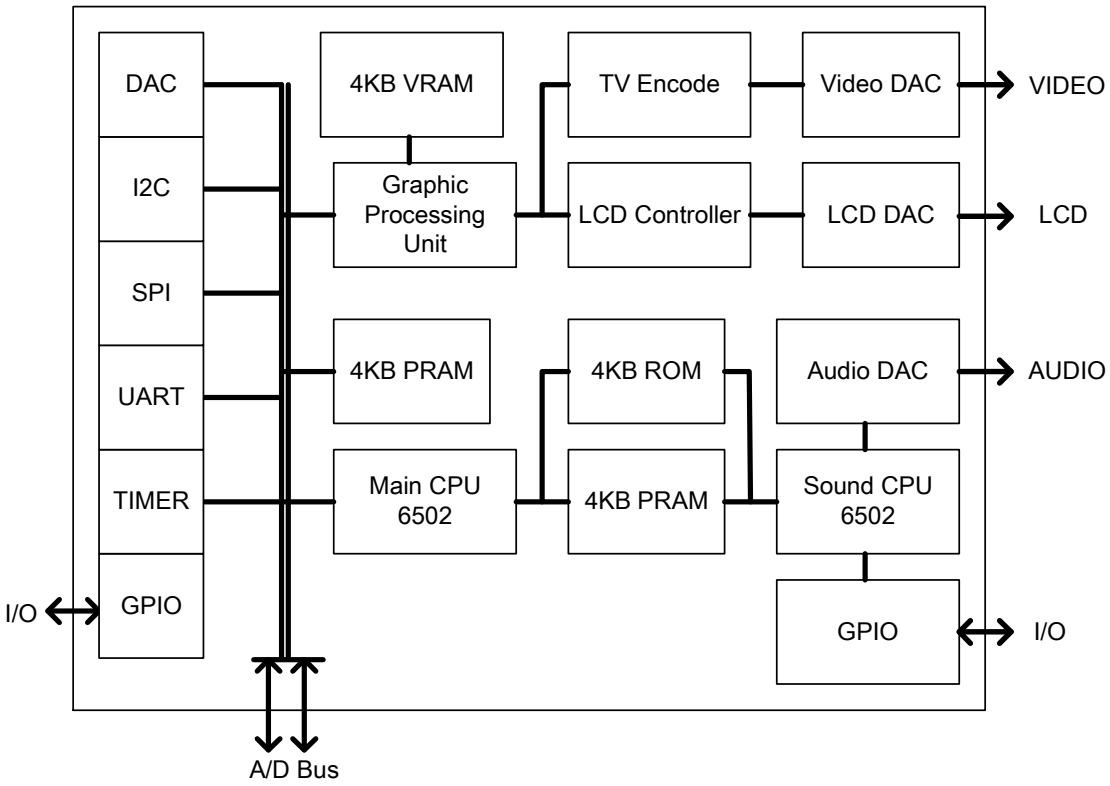
Graphic 處理器

- 分辨率: TV 256x240 點
- 在同一個畫面(Frame)有 240 個卡通塊,在橫向最大的卡通塊數是 16 個
- 2 個獨立的背景層.
- 背景字符方式: 16/64/256 索引顏色方式.
- 背景數位映像的方式: 16/64/256 索引顏色方式.或是 32768 色直接顏色方式
- 卡通塊(Sprites)為 16 色.
- 兩個 256 上色顏色調色板, 最大顯示的索引顏色: 512
- 背景縱向的寬度: x1/x1.5/x2
- 背景橫向的線單獨捲動: -128~+127

Sound CPU

- CPU 6502 @21.4772MHz in NTSC and 26.6017MHz in PAL
- 4K 字節共用 RAM
- 4K 字節可隨意調整的內部的 ROM
- 16 GPIO 端口
- 16 bits Timer x2
- ALU, 16 乘 16 乘法器和 32 除 16 除法器

3.2 結構圖(BLOCK DIAGRAM)



3.3 IC 腳位的描述

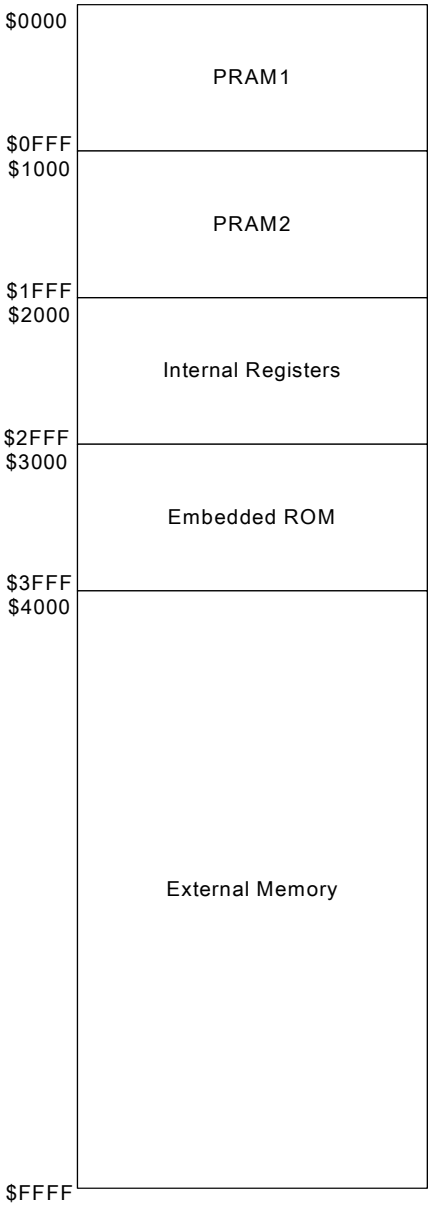
符　　號	型 式	描　　述
XA[23:0]	O	地址匯流排
XD[15:0]	I/O	數據匯流排
XROMCSB	O	1st 外部存儲器蕊片片選訊號
XRAMCSB	O	2nd 外部存儲器蕊片片選訊號
XROMOEB	O	外部存儲器 OEB 訊號
XRAMRWB	O	外部存儲器 RWB 訊號
XDEBUGNMI	I	除錯方式的 NMI
XDEBUGCSB	O	除錯方式存儲器蕊片片選訊號
XIOA[3:0]	I/O	一般的 I/O
XIOB[3:0]	I/O	一般的 I/O
XIOC[3:0]	I/O	一般的 I/O
XIOD[3:0]	I/O	一般的 I/O
XIOE[3:0]	I/O	一般的 I/O
XIOF[3:0]	I/O	一般的 I/O
XUIOA[7:0]	I/O	一般的 I/O
XUIOB[7:0]	I/O	一般的 I/O
XSCPUIOA[7:0]	I/O	一般的 I/O
XSCPUIOB[7:0]	I/O	一般的 I/O
XTAL1	I	晶振腳位
XTAL2	O	晶振腳位
XBOOTINIT	I	內部的 ROM 啟動方式
XPLLVCO	I/O	PLL 參考電壓
XPLLVREF	I/O	PLL 參考電壓
XVIDEO	O	混合視頻訊號
XAUDIOR	O	音頻訊號的右聲道
XAUDIOL	O	音頻訊號的左聲道

4. 主 CPU

在 VT1682 的主 CPU 是 8-bits 的 6502,操作頻率為 5MHz.

4.1 存儲器映射(Memory Map)

存儲器的映射如底下的圖形所示. PRAM1 (Program RAM-1)是給主 CPU 部份的 Program RAM 為 4K 字節. PRAM2 是音頻 CPU 和主 CPU 共用的 RAM 也是 4K 字節. 在 0x2000 和 0x20FF 之間的地址是 Graphic 端口,而其他的部份是由系統或是周邊來控制. 內部有一個 4K 字節的 ROM(Embedded ROM) 給主 CPU 或是 Sound CPU 用. 它可以充當 BIOS, 加密, 程序的 ROM 或是數據的 ROM. $4000 ~ $FFFF 是映射到 6 個程序儲存空間去擴充地址線到 8M 字節的外部的存儲器.



4.2 地址方式(Address Mode)

這個 16 位 6502 CPU 地址線被擴充到 25 位的物理地址線是根據以下的編輯程序樣式. 在下面的表格中, 6502 的地址線是 A[15:0] 和物理地址線是 {PA[24:13], A[12:0]}.

PQ2EN	COMR6	A15	A14	A13	TP20	TP19	TP18	TP17	TP16	TP15	TP14	TP13
0	0	1	0	0	Program_Bank0_Register0							
0	0	1	0	1	Program_Bank0_Register1							
0	0	1	1	0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1
0	1	1	0	0	1	1	1	1	1	1	1	0
0	1	1	0	1	Program_Bank0_Register1							
0	1	1	1	0	Program_Bank0_Register0							
0	1	1	1	1	1	1	1	1	1	1	1	1
1	0	1	0	0	Program_Bank0_Register0							
1	0	1	0	1	Program_Bank0_Register1							
1	0	1	1	0	Program_Bank0_Register2							
1	0	1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	0	Program_Bank0_Register2							
1	1	1	0	1	Program_Bank0_Register1							
1	1	1	1	0	Program_Bank0_Register0							
1	1	1	1	1	1	1	1	1	1	1	1	1
--	--	0	1	1	Program_Bank0_Register5							
--	--	0	1	0	Program_Bank0_Register4							

*TP[20:13]將會被轉換在下面的表格

Program_Bank0_select[2:0]			PA20	PA19	PA18	PA17	PA16	PA15	PA14	PA13
0	0	0	PQ37	PQ36	TP18	TP17	TP16	TP15	TP14	TP13
0	0	1	PQ37	PQ36	PA35	TP17	TP16	TP15	TP14	TP13
0	1	0	PQ37	PQ36	PQ35	PQ34	TP16	TP15	TP14	TP13
0	1	1	PQ37	PQ36	PQ35	PQ34	PA33	TP15	TP14	TP13
1	0	0	PQ37	PQ36	PQ35	PQ34	PQ33	PQ32	TP14	TP13
1	0	1	PQ37	PQ36	PQ35	PQ34	PQ33	PQ32	PA31	TP13
1	1	0	PQ37	PQ36	PQ35	PQ34	PQ33	PQ32	PQ31	PQ30
1	1	1	TP20	TP19	TP18	TP17	TP16	TP15	TP14	TP13

* ProgramBank0_Register3 = {PQ37, PQ36, PQ35, PQ34, PQ33, PQ32, PQ31, PQ30,}.

EXT2421	PQ2EN	COMR6	A15	A14	A13	PA24	PA23	PA22	PA21
1	x	x	1	x	x	Program_Bank1_Register3			
0	0	0	1	0	0	Program_Bank1_Register0			
0	0	0	1	0	1	Program_Bank1_Register1			
0	0	0	1	1	0	Program_Bank1_Register3			
0	0	0	1	1	1	Program_Bank1_Register3			
0	0	1	1	0	0	Program_Bank1_Register3			
0	0	1	1	0	1	Program_Bank1_Register1			
0	0	1	1	1	0	Program_Bank1_Register0			
0	0	1	1	1	1	Program_Bank1_Register3			
0	1	0	1	0	0	Program_Bank1_Register0			
0	1	0	1	0	1	Program_Bank1_Register1			

0	1	0	1	1	0	Program_Bank1_Register2
0	1	0	1	1	1	Program_Bank1_Register3
0	1	1	1	0	0	Program_Bank1_Register2
0	1	1	1	0	1	Program_Bank1_Register1
0	1	1	1	1	0	Program_Bank1_Register0
0	1	1	1	1	1	Program_Bank1_Register3
X	X	X	0	1	1	Program_Bank1_Register5
x	x	X	0	1	0	Program_Bank1_Register4

參考寄存器(Reference Registers)

	D7	D6	D5	D4	D3	D2	D1	D0
0x2100(W)					Program_Bank1_Register3			
0x2100(R)					Program_Bank1_Register3			
0x2105(W)	---	COMR6						
0x2107(W)	Program_Bank0_Register0							
0x2107(R)	Program_Bank0_Register0							
0x2108(W)	Program_Bank0_Register1							
0x2108(R)	Program_Bank0_Register1							
0x2109(W)	Program_Bank0_Register2							
0x2109(R)	Program_Bank0_Register2							
0x210A(W)	Program_Bank0_Register3							
0x210A(R)	Program_Bank0_Register3							
0x210B		PQ2EN				Program_Bank0_select		
0x210C(W)					Program_Bank1_Register2			
0x210C(R)					Program_Bank1_Register2			
0x2110(W)					Program_Bank1_Register0			
0x2112(R)					Program_Bank1_Register0			
0x2111(W)					Program_Bank1_Register1			
0x2113(R)					Program_Bank1_Register1			
0x2112(W)	Program_Bank0_Register4							
0x2110(R)	Program_Bank0_Register4							
0x2113(W)	Program_Bank0_Register5							
0x2111(R)	Program_Bank0_Register5							
0x2118(W)	Program_Bank1_Register5				Program_Bank1_Register4			
0x2118(R)	Program_Bank1_Register5				Program_Bank1_Register4			
0x211C(W)			EXT2421EN					

4.3 CPU 序號(CPU Vectors)

在主 CPU 的序號包括 NMI, RESET 和 5 IRQ,如以下的表格.

Vector Name	vector address
NMI	0x7FFFA, 0x7FFFB
System Reset	0x7FFFC, 0x7FFFD
Ext_IRQ	0x7FFFE, 0x7FFFF
Timer_IRQ	0x7FFF8, 0x7FFF9
SCPU_IRQ	0x7FFF6, 0x7FFF7
UART_IRQ	0x7FFF4, 0x7FFF5
SPI_IRQ	0x7FFF2, 0x7FFF3

5. 界面

VT1682 主 CPU 控制的界面包括電視複合信號的輸出, LCD, UART, SPI, IIC 和 CCIR 界面.

5.1 電視複合信號輸出

VT1682 可以支持 4 種電視系統和准許調整電視信號的飽和度(Saturation)和光亮度(Luminance). 在 0x2106 寫 "1" 到 TV_ON 和在 0x211D 寫 "1" 到 VDAC_EN 去使這個複合視頻輸出功能致能.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2106	---	---	SCPURN	SCPU_ON	SPI_ON	UART_ON	TV_ON	LCD_ON
0x211D	LVDEN			VDAC_EN	ADAC_EN	PLL_EN	LCDACEN	---

5.1.1 電視系統的表面配置

NTSC, PAL, PAL-M and PAL-N 電視系統是利用於 VT1682 內設置 0x2105 端口和在 IC 腳位 XTAL1 和 XTAL2 更換相關的晶振(crystal). 它們的相關性如下面的表格所列.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2105	---	COMR6	TV_SYS_SE:[1:0]		CCIR_SEL	Dual_Speed	ROM_SEL	PRAM

TV_SYS_SEL[1:0]	TV system	Crystal Frequency
0	NTSC	21.4772MHz
1	PAL_M	21.453669MHz
2	PAL_N	21.492336MHz
3	PAL	26.601712MHz

5.1.2 電視的參數

在 VT1682 裡面有 32 個明亮度(Brightness)和 8 個對比(Contrast)階段可以經由設置 0x2021 和 0x2022 端口由程序來控制. 好好配合 0x2021 的設置可以讓電視的畫面產生漸現(漸隱)的效果.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2021	----	----	Luminance_offset					
0x2022	----	----	VCOMIO	RGB_DAC	CCIR_OUT	Saturation		

光亮度補償值(Luminance_offset) : 電視輸出光亮度調整控制, 代表 2's 非.

2's 非: -A = ~A + 1,

Ex: -5 = ~(000101) + 1 = 111010 + 1 = 111011 = 59

有效的值 -32--------- -16 ------ -1 –0 – 1-------- 15 ------- 31

最暗----------暗--------正 常--------明亮-----最亮

Data to 0x2021 32----------48---------63--0--1--------15---------31

飽和度(Saturation) = 0 : 灰階輸出, 默認值為 4.

在 0x2022 的值 1-----2-----3-----4------5------6------7

飽和度(Saturation) 高<----------正常----------→ 低

5.2 LCD 界面

VT1682 可以提供不同類型的 LCD 界面, 包括 AUO-051(數位), AUO-052(數位), 類比 TFT LCD 和 CSTN 界面.在 0x200C 的 LCD_MODE 有 4 種主要的輸出擬定可被選擇.它們的映射信號如下面的表格所列.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2008	LCD_Y							
0x2009	LCD_X[7:0]							
0x200A	LCD_FR [7:0]							
0x200B							LCD_FR[8]	LCD_X[8]
0x200C	F_RATE		LCD_CLK		UPS052		LCD_MODE	
0x200D	LCDEN	Dot240	Reverse		Color_Sequence			
0x2022	----	----	VCOMIO	RGB_DAC	CCIR_OUT	Saturation		

LCD_Y : LCD 顯示屏縱向的補償,以橫向的掃瞄線為基礎.

備註: 0xFF 和 0 是不許用的.

LCD_X : LCD 顯示屏橫向的補償,以 LCD dot clock 為基礎.

備註: 0x1FF 和 0 是不許用的.

LCD_FR: STN LCD 交互的信號 toggle rate 以橫向的掃瞄線為基礎.

F_RATE: LCD 控制器參數, STN 框架速率(frame rate)控制.

0 : 60(N) / 50(P) FPS 1 : 120(N) / 100(P) FPS

LCD_CLK: LCD 控制器參數, LCD dot clock 選擇

0 : 21.4772 / 4 MHz 1 : 21.4772 / 2 MHz

2 : 21.4772 MHz 3 : External clock*

UPS052 : LCD 控制器參數, AUO UPS052 TCON 方式選擇

0 : 非 UPS052 方式 1 : UPS052 方式

LCDEN : LCD 致能控制

0 : 無作用. 1 : 致能.

Dot240 : LCD 依比率決定控制, 依比率決定比率到 15/16.

0 : No scaling 1 : Scaling

反轉 : LCD 控制器參數,點陣數據輸出反轉

0 : 正常 1 : 反轉

LCD_MODE : LCD 控制器參數, 控制器輸出擬定選擇

0 : ANALOG 方式 1 : CSTN 方式

2 : LTPS 方式 3 : DIGITAL 方式

VCOMIO : LCD 輸入 VCOM 方式

0 : 無作用 1 : 致能

類比方式(ANALOG Mode)

腳位名稱	信號名稱	設　　置	描　　述
XIOA0	DIO	IOA_ENB=1, IOA_OE=1	Vertical start pulse
XIOA1	XOE	IOA_ENB=1, IOA_OE=1	Output enable for scan driver
XIOA2	CPH1	IOA_ENB=1, IOA_OE=1	Sample and shift clock pulse for data driver
XIOA3	CPH3	IOA_ENB=1, IOA_OE=1	Sample and shift clock pulse for data driver
XIOB0	Q2H	IOB_ENB=1, IOB_OE=1	Analog rotate signal
XIOB1	VCOM	IOB_ENB=1, IOB_OE=1	Common electrode driving signal
XIOB2	CPV	IOB_ENB=1, IOB_OE=1	Shift clock for scan driver
XIOB3	INH	IOB_ENB=1, IOB_OE=1	Output enable for data driver
XIOC0	CPH2	IOC_ENB=1, IOC_OE=1	Sample and shift clock pulse for data driver
XIOC1	STH	IOC_ENB=1, IOC_OE=1	Start pulse of horizontal scan line
XIOE0	VR	IOE0OE = 0	Analog R
XIOE1	VG	IOE1OE = 0	Analog G
XIOE2	VB	IOE2OE = 0	Analog B

LTPS Mode

腳位名稱	信號名稱	設　　置	描　　述
XIOA0	DIO	IOA_ENB=1, IOA_OE=1	Frame start signal
XIOA1	XOE	IOA_ENB=1, IOA_OE=1	Inverse of frame start signal
XIOA2	CPH1	IOA_ENB=1, IOA_OE=1	Dot clock
XIOB0	Q2H	IOB_ENB=1, IOB_OE=1	Line clock
XIOB1	VCOM	IOB_ENB=1, IOB_OE=1	Common electrode driving signal
XIOB2	CPV	IOB_ENB=1, IOB_OE=1	Inverse of line clock
XIOB3	INH	IOB_ENB=1, IOB_OE=1	Inverse of line start signal
XIOC0	CPH2	IOC_ENB=1, IOC_OE=1	Inverse of dot clock
XIOC1	STH	IOC_ENB=1, IOC_OE=1	Line start signal
XIOE0	VR	IOE0OE = 0	Analog R
XIOE1	VG	IOE1OE = 0	Analog G
XIOE2	VB	IOE2OE = 0	Analog B

數位方式(Digital Mode)

腳位名稱	信號名稱	設　　置	描　　述
XIOA0	DIO	IOA_ENB=1, IOA_OE=1	Vertical SYNC
XIOA1	XOE	IOA_ENB=1, IOA_OE=1	Horizontal SYNC
XIOA2	CPH1	IOA_ENB=1, IOA_OE=1	Dot clock
XIOA3	CPH3	IOA_ENB=1, IOA_OE=1	DATA[0]
XIOB0	Q2H	IOB_ENB=1, IOB_OE=1	DATA[1]
XIOB1	VCOM	IOB_ENB=1, IOB_OE=1	DATA[3]
XIOB2	CPV	IOB_ENB=1, IOB_OE=1	DATA[2]
XIOB3	INH	IOB_ENB=1, IOB_OE=1	DATA[4]

CSTN 方式

腳位名稱	信號名稱	設　　置	描　　述
XIOA0	DIO	IOA_ENB=1, IOA_OE=1	Frame Start
XIOA1	XOE	IOA_ENB=1, IOA_OE=1	Line Data Load
XIOA2	CPH1	IOA_ENB=1, IOA_OE=1	DCLK
XIOA3	CPH3	IOA_ENB=1, IOA_OE=1	AC Signal
XIOB0	Q2H	IOB_ENB=1, IOB_OE=1	DATA[0]
XIOB1	VCOM	IOB_ENB=1, IOB_OE=1	DATA[2]
XIOB2	CPV	IOB_ENB=1, IOB_OE=1	DATA[1]
XIOB3	INH	IOB_ENB=1, IOB_OE=1	DATA[3]

5.3 UART 界面

在 VT1682 的 UART 控制器可以提供達到 11520 bps 接收/傳遞的 baud-rate, 可程序控制接收/傳送的 IRQ, 同位檢查(parity check). UART 傳送 (TX) 和 UART 接收(RX) 界面是在 XIOC2 和 XIOC3.

5.3.1 UART 控制寄存器

在 0x2106 的 UART_ON 必須在任何一個寄存器設置到 UART 端口之前先被設置為致能. 當 TXIRQEn 或是 RXIRQEn, TXIRQ 和 RXIRQ 兩者將會供給到 CPU IRQ3, UART_IRQ, with IRQ vectors 0x7FFF4, 0x7FFF5.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2106	---	---	SCPURN	SCPU_ON	SPI_ON	UART_ON	TV_ON	LCD_ON
0x2119	--	CarriEn	UARTEN	TXIRQEn	RXIRQEn	ParityEn	OddEven	9bitmode

UARTEN : UART 致能

0 : 無作用 1 : 致能

TXIRQEn : TX IRQ 致能.當 TXIRQEn=1 和 TX_Status=1 時, UART_IRQ 會出來.

0 :無作用 1 : 致能

RXIRQEn : RX IRQ 致能. 當 RXIRQEn=1 和 RX_Status=1 時,UART_IRQ 會出來.

0 : 無作用 1 : 致能

ParityEn : RX 同位檢查致能,這個檢查結果是於 UART 狀態端口表示在 ParityErr.

0 : 無作用 1 : 致能

OddEven : 當 ParityEn 被設置, 單數(Odd) / 偶數(Even)同位必須被設置.

0 : 單數(Odd)同位 1 : 偶數(Even)同位

9bitmode : RX / TX 於 9 bits 方式下傳送. 當 ParityEn 被設置, 這個 9th bit 會是同位 bit (odd / even). 不然的話, 這個 9th bit 會只是在 OddEven 的數據.

UART 狀態 (只能讀取(Read Only))

0x211B	--	--	RxError	TX_Status	RX_Status	ParityErr	--	--

RxError : RX 錯誤標誌.

0 : 無錯誤 1 : Baudrate 不匹配或是超過停止位.

TX_Status : TX 的狀態標誌. 寫 Tx_Data 會清除此狀態標誌.

0 : UART 正在傳送 1 : UART 完成傳送

RX_Status : RX 的狀態標誌. 讀 Rx_Data 會清除此狀態標誌.

0 : 無數據在 Rx_Data 1 : 在 Rx_Data 數據是有效的

ParityErr : 同位檢查錯誤標誌. ParityEn 必須先被致能.

0 : 同位正確 1 : 同位錯誤

0x211A(W)	Tx_Data[7:0]
0x211A(R)	Rx_Data[7:0]

寫 0x211A 會經由 TX 傳送數據. 當 RX_Status=1 時,讀 0x211A 會得到 RX 數據.

5.3.2 波特速率(Baud Rate) 設置

在 VT1682 的 UART 控制器可以提供達到 11520 bps 接收/傳送的波特速率(baud-rate).此外, UART 界面也可以在 TX 輸出提供載波參數給紅外線(Infra-red)應用. 在 0x211B 端口設置 Carier_Frequency 來調整載波頻率. 在 0x2119 設置 CarriEN 使載波功能致能.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2114	Baud_phase[7:0]							
0x2115	Baud_phase[15:8]							
0x211B	Carrier_frequency[7:0]							

For NTSC system,

Baud_phase[15:0] = 65536 * Baud_rate / 5.3693 x10^6

Carrier_Frequency[7:0] = 256 – (5.3693 x10^6 / $F_{carrier}$)

For PAL system,

Baud_phase[15:0] = 65536 * Baud_rate / 5.3203424 x10^6

Carrier_Frequency[7:0] = 256 – (5.3203 x10^6 / $F_{carrier}$)

哪裡的波特速率(Baud_rate)是 115200, 57600, 38400, 19200, 9600, 4800, 2400 和 $F_{carrier}$ 是這個需要運送的頻率.

5.3.3 UART 信號

UART 腳位, TX 和 RX 是共用 XIOC2 和 XIOC3.

	腳位名稱	設 置	描 述
TX	XIOC2	IOC_ENB=1, IOC_OE=1, UART_ON	UART Transmit (OUTPUT)
RX	XIOC3	IOC_ENB=1, IOC_OE=1, UART_ON	UART Receive (INPUT)

程序設計註解:

UART_TX 會在 XIOC2 和 UART_RX 在 XOIC3. 以下的寄存器必須在 UART 寄存器之前被設置.

```
$2106.D2 = 1;          // UART 模組致能
$210D.D4 = 1;          // IOC 輸出致能
$210D.D5 = 1;          // IOC 輸出
```

5.4 SPI 界面

SPI (標準的周邊界面)可以操作在主動裝置(master)和從動裝置(slave) 方式. 它可以操作在 4 種不同的速度,由 0x2116 的 CK_FREQ 控制. 有 4 個型式的通信協議,如下表所描述.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2116	16bitMode	SPIEN	SPI_RST	M/SB	CKPHASE	CKPOLAR	CK_FREQ[1:0]	
0x2117(W)	SPI_TX_Data							
0x2117(R)	SPI_RX_Data							

CK_FREQ : SPI clock 操作頻率.

CK_FREQ	SPI SCK Frequency
0	2.5MHz
1	1.25MHz
2	639KHz
3	320KHz

CKPOLAR : SPI clock 極性控制.

CKPHASE : SPI clock 樣本相位控制.

M/SB : SPI 主動裝置(Master) / 從動裝置(slave) 方式選擇

SPI_RST : SPI 模組復位信號

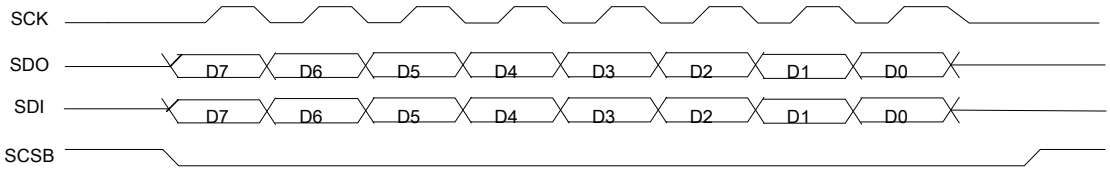
0 : 正常操作　　1 : SPI 復位

SPIEN : SPI 模組致能

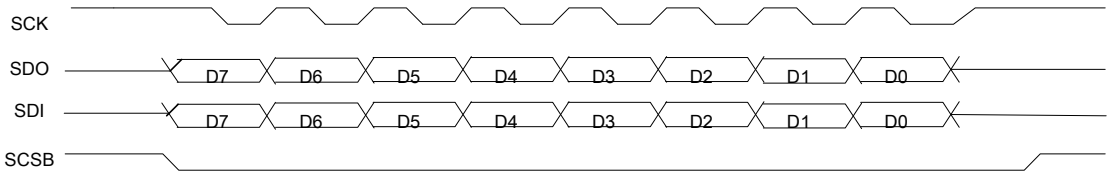
0 : 無作用　　1 : 致能

16bitMode : 8 / 16-bits SPI 方式選擇

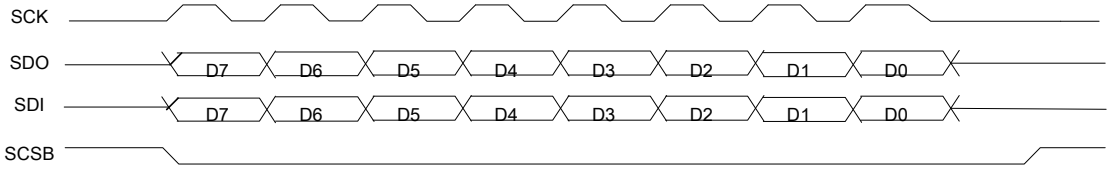
0 : 8-bits 方式　　1 : 16-bits 方式



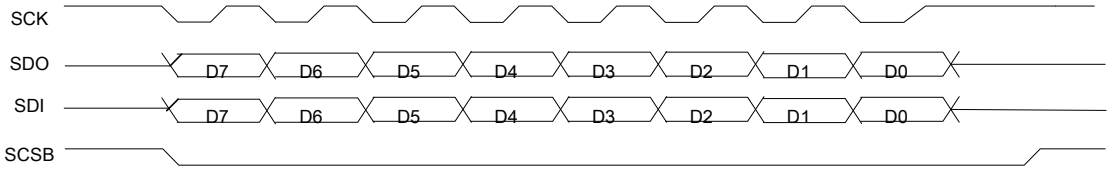
CKPOLAR=0, CKPHASE =0



CKPOLAR=1, CKPHASE =0



CKPOLAR=0, CKPHASE =1



CKPOLAR=1, CKPHASE =1

PIN NAME	Master Mode IOD_ENB=1, IOD_OE=1SPI_ON = 1	Slave Mode IOD_ENB=1, IOD_OE=0SPI_ON = 1	描　　述
XIOD0	OUTPUT:SCK	INPUT:SCK	SPI CLOCK
XIOD1	INPUT:SDI	INPUT:SDI	SPI DATA INPUT
XIOD2	OUTPUT:SDO	OUTPUT:SDO	SPI DATA OUTPUT
XIOD3	OUTPUT:SCSB	INPUT:SCSB	SPI CSB

5.5 I2C

在 VT1682 主動裝置(Master)方式的 I2C 功能是有效的. 每一個 I2C 命令包括 IIC_ID, IIC_ADDR 和 IIC_Data 如下面的圖所示.當 IIC_ID 的低位(LSB)是偶數時,在 0x2142 的數據會輸出在第三個相位.不然的話 IIC_DATA 會是在 0x2142 端口讀到的流動輸入數據.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2140	IIC_ID							
0x2141	IIC_ADDR							
0x2142(W)	IIC_DATA							
0x2142(R)	IIC_DATA							
0x2143							IIC_CLK_SEL	

IIC_CLK_SEL : I2C SCK 頻率選擇

0 : 1.34MHz　　1 : 670KHz

2 : 335KHz　　3 : 禁止使用

5.6 CCIR 通信協議(protocol)

VT1682 提供兩組 CCIR 的界面,CCIR656 和 CCIR 601 界面, SET0 和 SET1 給影像輸入.這個 CCIR 輸入必須是 60 FPS for NTSC 或是 50 FPS for PAL system.

5.6.1 CCIR 輸入腳位

這兩組的 CCIR 輸入是在 XUIOA, XUIOB, XSCPUIOA 和 XSCPUIOB ,如下面的表格所示.當這個 CCIR 界面被使用時,這些相關的 IO 腳必須被設置到輸入方式. CCIR SET0 或是 SET1 是由 0x2105 的 CCIR_SEL 來選擇.

SIGNAL NAME	SET0	SET1	Description
CCIR_D0	XUIOA0	XSCPUIOA0	DATA[0]
CCIR_D1	XUIOA1	XSCPUIOA1	DATA[1]
CCIR_D2	XUIOA2	XSCPUIOA2	DATA[2]
CCIR_D3	XUIOA3	XSCPUIOA3	DATA[3]
CCIR_D4	XUIOA4	XSCPUIOA4	DATA[4]
CCIR_D5	XUIOA5	XSCPUIOA5	DATA[5]
CCIR_D6	XUIOA6	XSCPUIOA6	DATA[6]
CCIR_D7	XUIOA7	XSCPUIOA7	DATA[7]

CCIR_CK	XUIOB0	XSCPUIOB0	CCIR CLOCK (27MHz)
CCIR_HS	XUIOB1	XSCPUIOB1	CCIR601 HSYNC
CCIR_VS	XUIOB2	XSCPUIOB2	CCIR 601 VSYNC

5.6.2 CCIR 控制寄存器

下面的寄存器是被使用來控制 CCIR 的界面.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2000				Capture	SLAVE	---	---	NMI_EN
0x2028	----	----	CCIR_Y					
0x2029	----	----	----	CCIR_X				
0x202A	VS_Phase	HS_Phase	YC_Swap	CbCrswap	SYNCMOD	YUV_RGB	Field_OEn	Field_On
0x2105	---	COMR6	TV_SYS_SE:[1:0]		CCIR_SEL	Double	ROM_SEL	PRAM

SLAVE : 從動裝置(slave)方式致能控制, VT1682 畫面和聲音會與 CCIR 界面一致.

0 : 正常方式　　1 : 從動裝置(Slave)方式(CCIR 輸入)

CCIR_Y : CCIR 縱向的補償通信協議與橫向的掃描線的起點.

Effective offset = 64 - CCIR_Y

CCIR_X : CCIR 橫向的補償通信協議與晶振周期的數目.

有效的補償(Effective offset) = 32 - CCIR_H

註釋 : 在 CCIR_V 和 CCIR_H 0 是禁止用的.

Field_On : CCIR 參數, 底色方式致能.

0 : 無作用　　1 : 致能

Field_OEn : CCIR 參數,當 Field_On 是 1 時有效.

YUV_RGB : CCIR 參數,輸入數據格式的選擇.

0 : YUV 4:2:2 方式　　1 : GBR 4:2:2 方式

SYNC_MOD : CCIR 參數, 同步的方式的選擇.

0 : CCIR656　　1 : CCIR601

CbCrSwap : CCIR 參數, Cb 和 Cr 命令的控制.

0 : CbYCrY　　1 : CrYCbY

YC_Swap : CCIR 參數, Y 和 CrCb 命令的控制.

0 : CbYCrY　　1 : YcbYCr

HS_Phase : CCIR 參數, 在 CCIR601 內 hsync 極性的控制.

0 : 低脈衝(Low pulse)　　1 : 高脈衝(High pulse)

VS_Phase, CCIR 參數, 在 CCIR601 內 vsync 極性的控制.

0 : 低脈衝(Low pulse)　　1 : 高脈衝(High pulse)

6. 周邊(PERIPHERAL)

6.1 提高計算(Arithmetic)單元 –乘法器(Multiplier)和除法器(Divider)

VT1682 提供兩個計算的單元,乘法器和除法器,一個是給主 CPU 而另外一個是給 Sound CPU. 這個乘法器是 16 位乘 16 位,這個除法器是 32 位除 16 位. 這個乘法需要 16 個 CPU clock cycle 才能完成操作,而除法需要 32 個 CPU clock cycle 才能完成操作.

6.1.1 乘法器 (16x16)

這個乘法操作是:

ALU_ Multi _operand6, ALU_ Multi _operand5

X) ALU_operand2, ALU_operand1

= ALU_out4, ALU_out3, ALU_out2, ALU_out1

當 ALU_Multi_operand6 被寫入時這個操作開始.在乘法運算之後是在 ALU_Multi_operand5 和 ALU_Multi_operand6 的值被改變, 而不是在 ALU_operand1 和 ALU_operand2 的值.

0x2130(W)	ALU_operand1
0x2131(W)	ALU_operand2
0x2132(W)	ALU_operand3
0x2133(W)	ALU_operand4
0x2134(W)	ALU_Multi_operand5
0x2135(W)	ALU__Multi_operand6

0x2130(R)	ALU_out1
0x2131(R)	ALU_out2
0x2132(R)	ALU_out3
0x2133(R)	ALU_out4

6.1.2 除法器

這個除法操作是:

ALU_operand4, ALU_operand3, ALU_operand2, ALU_operand1

ALU__Multi_operand6, ALU__Multi_operand5

= { ALU_out4, ALU_out3, ALU_out2, ALU_out1} + { ALU_out6, ALU_out5}或是+{{ ALU_out6, ALU_out5}*2 - { ALU_out4, ALU_out3, ALU_out2, ALU_out1}}

當 LSB(Least Significant Bit) 為 “1”時, 餘數會是 :

{ ALU_out6, ALU_out5}*2 - { ALU_out4, ALU_out3, ALU_out2, ALU_out1}

當 LSB 為 “0”時, 餘數會是 { ALU_out6, ALU_out5}.

當 ALU__Div_operand6 被寫入時這個操作開始. 在除法運算之後是在 ALU_operand1 和 ALU_operand2, ALU_operand3 和 ALU_operand4 的值被改變,而不是在 ALU_Div_operand5 和 ALU_Div_operand6 的值.

0x2130(W)	ALU_operand1
0x2131(W)	ALU_operand2
0x2132(W)	ALU_operand3

0x2133(W)	ALU_operand4
0x2136(W)	ALU_Div_operand5
0x2137(W)	ALU_div_operand6

0x2130(R)	ALU_out1
0x2131(R)	ALU_out2
0x2132(R)	ALU_out3
0x2133(R)	ALU_out4
0x2134(R)	ALU_out5
0x2135(R)	ALU_out6

6.2 定時器(Timer)

寫 Timer_PreLoad 去初始化這個定時器的頻率. 寫 0x2104 會重新裝 Timer_Preload 到計時器, 所以 0x2101 必須要在 0x2104 之前被寫入.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2101(W)	Timer_Preload[7:0]							
0x2101(R)	Timer_Preload[7:0]							
0x2102							TMR_IRQ	TMR_EN
0x2103	Timer_IRQ_Clear							
0x2104(W)	Timer_Preload[15:8]							
0x2104(R)	Timer_Preload[15:8]							
0x210B	TSYNEN							

Timer_PreLoad[15:0], Timer IRQ 周期定義.

TSYNEN : 定時器基礎頻率的選擇

	TSYNEN	定時器基礎頻率
NTSC	1	15.746KHz
	0	5.3693 MHz
PAL	1	15.602KHz
	0	5.32034 MHz

當 TSYNEN = 0 時,

For NTSC,

Period = (65536 - Timer_PreLoad) / 5.3693MHz

Timer_PreLoad = 65536 – (Period(sec) * 5.3693 * 10^6)

For PAL

Period = (65536 - Timer_PreLoad) / 5.32034MHz

Timer_PreLoad = 65536 – (Period (sec) * 5.32034 * 10^6)

TMR_En : 計時器致能控制.

0 : 無作用 1 : 致能

TMR_IRQ, 計時器 IRQ 致能控制.

0 : 無作用 1 : 致能

Timer_IRQ_Clear : 計時器 IRQ 清除控制, 寫任何數據去清除計時器 IRQ.

6.3 隨機數產生器(Random Number Generator)

在 VT1682 有一個 8-bits 虛擬的隨機數產生器.它需要去寫一個非 0 的種子數字到 0x212C 開頭. 這個隨機數在讀取 0x212C 時會是有效的. 每一次一個新的種子數字被寫入,一個新的虛擬的隨機數序號會被產生出來.請注意當你開始對這個隨機數尋址時是不需要每次去寫這個種子數字的.只有在初始的順序或是改變這個隨機數的順序時你必須去更新這個發送數字.

	D7	D6	D5	D4	D3	D2	D1	D0
0x212C(W)	Pseudo_random_number_seed							
0x212C(R)	Pseudo_random_number							

6.4 類比到數位轉換器 (ADC)

在 VT1682 有一個 ADC,它擁有 5 個時間分割複合(timing-division-multiplex)通道和 8 位的分辨率. 其中的 4 個是標準的 ADC 端口.輸入電壓在(VCC-1) volt 和 GND 之間會被轉換到 255~0. 另外一個端口是給麥克風應用的,它包括有一個聲音增大控制器(VGC). 這個 VGC 增大是在 0.5 倍和 127.5 倍之間.當這些端口 XIOF0, XIOF1, XIOF2, XIOF3 或是 XIOE3 是 ADC 的輸入時,記得設置 IOFOE 到低電平.

	D7	D6	D5	D4	D3	D2	D1	D0
0x211E(W)	ADCEN	ADCSEL		UNUSE	IOFOE3	IOFOE2	IOFOE1	IOFOE0
0x211E®	ADC_Data[7:0]							
0x211F	VGCEN	VGCGAIN						

VGC Gain = VGCGAIN + 0.5

VGCEN : VGC 致能控制

0： 無作用　　1： 致能

ADCEN : ADC 致能控制

0： 關掉　　1： 正常操作

ADCSEL	ADC Input Source
0	XIOF0
1	XIOF1
2	XIOF2
3	XIOF3

程序設計註釋:

1. IOFOE0 = 0, ADCSEL = 0　　// 選擇 XIOF0
2. ADAC_EN = 1,　　// ADC 致能
3. 定期的讀取 ADC_Data　　// 取得 ADC 數據

6.5 相位鎖定迴路(Phase Lock Loop (PLL))

在 VT1682 內建一個 PLL 作為 LCD 應用. 它可以產生下面的效果

	D7	D6	D5	D4	D3	D2	D1	D0
0x211D	LVDEN	LVDS1	LVDS0	VDAC_EN	ADAC_EN	PLL_EN	LCDACEN	---
0x212D	SCALE_B				SCALE_M	SCALE_A		

F_{SCALE} = Fosc * (SCALE_A + 2) * (SCALE_M+1) / (SCALE_B + 2)

6.6 數位到類比轉換器(DAC)

在 VT1682 內有 6 個 DAC (數位到類比轉換器). 一個給視頻(Video), 兩個給音頻(Audio),另外三個給類比(Analog)LCD 界面. 然而,在應用時不是所有的 DAC 都可以被使用. 例如,如果你使用數位(Digital)LCD 界面,這三個 LCD ADC 將不能使用. 所以你可以把它們當額外的應用.

名稱	功 能	分辨率	反應時間	輸出 PAD	控制端口
Video DAC	Composite Video	6	50ns	XVIDEO	0x2030
LCD DAC1	Analog LCD	5	50ns	XIOE0	0x2031
LCD DAC2	Analog LCD	5	50ns	XIOE1	0x2032
LCD DAC3	Analog LCD	5	50ns	XIOE2	0x2033
Audio DAC1*	Audio	12	5us	XAUDIOL	0x2118, 0x2119(SCPU)
Audio DAC2*	Audio	12	5us	XAUDIOR	0x211A, 0x211B(SCPU)

* Audio DAC 端口 0x2118~0x211B 是只給 Sound CPU 尋址.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2030	----	VDACSW	VDAC_OUT[5:0]					
0x2031	----	----	RDACSW	RDAC_OUT[4:0]				
0x2032	----	----	GDACSW	GDAC_OUT[4:0]				
0x2033	----	----	BDACSW	BDAC_OUT[4:0]				

VDACSW : Video DAC 輸出數據開關.

0 : 電視合成的輸出(默認) 1: 輸出 VDAC_OUT

RDACSW : LCD DAC 輸出數據開關

0 : 輸出 LCD 控制器數據(默認) 1: 輸出 RDAC_OUT

GDACSW : LCD DAC 輸出數據開關

0 : 輸出 LCD 控制器數據(默認) 1: 輸出 GDAC_OUT

BDACSW : LCD DAC 輸出數據開關

0 : 輸出 LCD 控制器數據(默認) 1: 輸出 BDAC_OUT

註釋 1 : 如果你需要電視合成的輸出, VDACSW 必須設置在低電平(LOW).

註釋 2 : 如果類比的(Analog) RGB 數據被使用在 LCD 界面, 這些開關必須設置在低電平(LOW).

註釋 3 : 音頻(Audio)DAC 控制請參考"Sound CPU" 部份.

6.7 低電壓檢查 (LVD)

VT1682 有四個階段的低電壓檢查功能. 它們為 3.65V, 3.18V, 2.71V 和 2.24V. 當 IC 的電源比檢查出的標準低時, 在 0x211D 的 LVD 標誌會進入 "high". 寫 "1" 到 LVDEN 去使 LVD 功能致能.

	D7	D6	D5	D4	D3	D2	D1	D0
0x211D(W)	LVDEN	LVDS1	LVDS0	VDAC_EN	ADAC_EN	PLL_EN	LCDACEN	---
0x211D(R)	----	----	----	----	----	----	----	LVD

LVDEN : LVD 致能控制

0 : 無作用　　1 : 致能

6.8 內部的 ROM(Embedded ROM)

在 VT1682 內部有一個 4K 字節的 ROM(Embedded ROM) .它可以給主 CPU 或是 Sound CPU 兩者任一來尋址,由 0x2105 內的 ROM_SEL 來選擇而且這個 ROM 它的地址是在 0x3000 和 0x3FFF 之間.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2105(W)	---	COMR6	TV_SYS_SE:[1:0]		CCIR_SEL	Dual_Speed	ROM_SEL	PRAM

ROM_SEL	Main CPU	Sound CPU
0	0x3000 ~ 0x3FFF	Invalid
1	Invalid	0x3000 ~ 0x3FFF

這個 ROM 也可以是一個 Boot ROM. 當"XBOOTINIT" 這個腳位被置為高電平時, 這些 RESET, NMI, IRQ 序號(vectors)會從 0x7FFF0 ~ 0x7FFFF 改變到 0x00FF0 ~0x00FFF.

7. GRAPHIC –圖形處理單元(PPU)

7.1 特點(Feature)

- 分辨率: TV 256x240 點
- 在同一個畫面(Frame)有 240 個卡通塊, 在橫向最大的卡通塊數是 16 個
- 2 個獨立的背景層.
- 背景字符方式: 16/64/256 索引顏色方式.
- 背景數位映像的方式: 16/64/256 索引顏色方式.或是 32768 色直接顏色方式
- 卡通塊(Sprites)為 16 色.
- 兩個 256 上色顏色調色板, 最大顯示的索引顏色: 512
- 背景縱向的寬度: x1/x1.5/x2
- 背景橫向的線單獨捲動: -128~+127

7.2 畫面結構(Screen Structure)

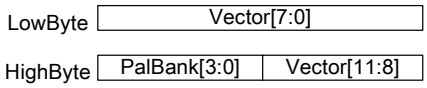
在 VT1682 顯示畫面(在電視機)圖像的分辨率是 256 x 240 點.然而這個 VRAM 畫面(這個圖形塊定義在給背景層 (background layer)的 VRAM)是 256 x256 點.

7.2.1 VRAM 畫面結構(Screen Structure)

去顯示一個背景在電視機的畫面,這個在 VRAM 的圖形塊必須要先被定義. 在 VRAM 有兩類的畫面結構,它們是字符(Character)和數位映像(Bitmap)方式.在字符(Character)方式, 這個畫面分割成字符塊.在數位映像方式,這個畫面分割成一條一條線.

7.2.1.1 字符方式(Character Mode)

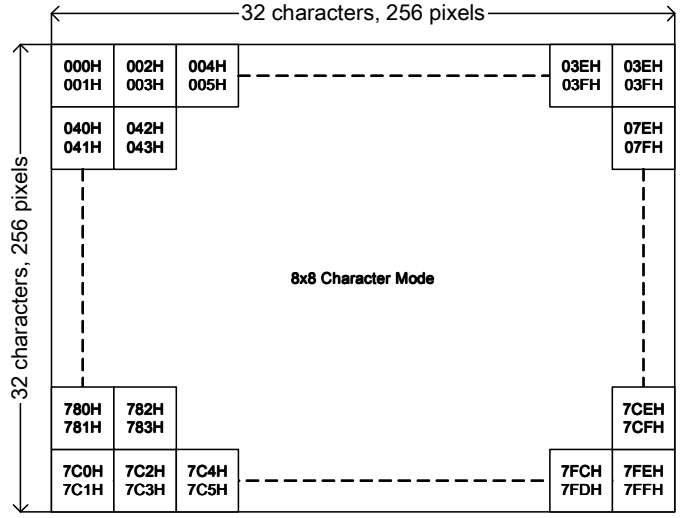
在背景的每一個字符需要兩個字節來定義. 如下面的圖形所示.這兩個字節包括 12 bits 的字符序號(character vector)和 4 bits 的調色板儲存空間參數(palette bank parameters). 序號(Vector) = 0 表示為透明的字符(transparent character).請參考圖像地址方式(Graphic Addressing)部份和顏色調色板部份有關於這些參數的詳細描述.



不同的字符大小會需要不同數量的字符圖形塊來定義一個畫面,如下面的表格所示:

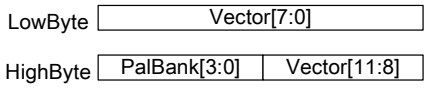
字符大小 (H x V)	需要的圖形塊數量	需要的 VRAM (字節)
8 x 8	32 x 32	2048
8 x 16	32 x 16	1024
16 x 8	16 x 32	1024
16 x 16	16 x 16	512

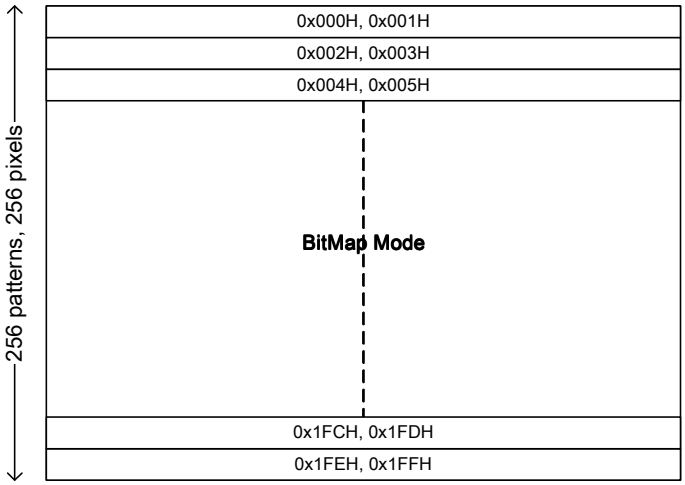
下面的例子是一個 8x8 字符(Character)方式它需要 32 x 32 的圖形塊去定義一個畫面.



7.2.1.2 數位映像(Bitmap)方式

在這個方式, 畫面分割到 256 條線(圖形塊) 而且每一條線需要兩個字節來定義, 如下面的圖表所示. 序號(Vector)= 0 表示在數位映像方式的透明線,但是不是在高顏色方式.





7.2.1.3 VRAM 管理(VRAM Management)

有一個 4K 字節的 VRAM 給背景層用. 在字符方式, 不同的字符大小會需要不同的 VRAM 區域 (**7.2.1.1 字符方式**). 底下的表格所列是 BK1 和 BK2 在不同的字符大小和捲軸方式的存儲器映射(memory map).

BK1

	V_scroll_en	H_scroll_en	Y[8]	X[8]	BK1
8x8	0	0	0	0	0x000 ~ 0x7FF
			0	1	0x800 ~ 0xFFF
			1	0	0x800 ~ 0xFFF
			1	1	0x800 ~ 0xFFF
	0	1	0	0	0x000 ~ 0x7FF, 0x800 ~ 0xFFF
			0	1	0x800 ~ 0xFFF, 0x000 ~ 0x7FF
			1	0	0x000 ~ 0x7FF, 0x800 ~ 0xFFF
			1	1	0x800 ~ 0xFFF, 0x000 ~ 0x7FF
	1	0	0	0	0x000 ~ 0x7FF, 0x800 ~ 0xFFF
			0	1	0x000 ~ 0x7FF, 0x800 ~ 0xFFF
			1	0	0x800 ~ 0xFFF, 0x000 ~ 0x7FF
			1	1	0x800 ~ 0xFFF, 0x000 ~ 0x7FF
	1	1	0	0	Forbidden
			0	1	Forbidden
			1	0	Forbidden
			1	1	Forbidden
16x16	0	0	0	0	0x000 ~ 0x1FF
			0	1	0x200 ~ 0x3FF
			1	0	0x400 ~ 0x5FF
			1	1	0x600 ~ 0x7FF
	0	1	0	0	0x000 ~ 0x1FF, 0x200 ~ 0x3FF
			0	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
			1	0	0x000 ~ 0x1FF, 0x200 ~ 0x3FF
			1	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
	1	0	0	0	0x000 ~ 0x1FF, 0x200 ~ 0x3FF
			0	1	0x000 ~ 0x1FF, 0x200 ~ 0x3FF
			1	0	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
			1	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
	1	1	0	0	0x000 ~ 0x1FF, 0x200 ~ 0x3FF, 0x400 ~ 0x5FF, 0x600 ~ 0x7FF,
			0	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF, 0x600 ~ 0x7FF, 0x400 ~ 0x5FF,
			1	0	0x400 ~ 0x5FF, 0x600 ~ 0x7FF, 0x000 ~ 0x1FF, 0x200 ~ 0x3FF,
			1	1	0x600 ~ 0x7FF, 0x400 ~ 0x5FF, 0x200 ~ 0x3FF, 0x000 ~ 0x1FF,

BK2

	V_scroll_en	H_scroll_en	Y[8]	X[8]	BK2
8x8	0	0	0	0	0x000 ~ 0x7FF
			0	1	0x800 ~ 0xFFF
			1	0	0x800 ~ 0xFFF
			1	1	0x800 ~ 0xFFF
	0	1	0	0	0x000 ~ 0x7FF, 0x800 ~ 0xFFF
			0	1	0x800 ~ 0xFFF, 0x000 ~ 0x7FF
			1	0	0x000 ~ 0x7FF, 0x800 ~ 0xFFF
			1	1	0x800 ~ 0xFFF, 0x000 ~ 0x7FF
	1	0	0	0	0x000 ~ 0x7FF, 0x800 ~ 0xFFF
			0	1	0x000 ~ 0x7FF, 0x800 ~ 0xFFF
			1	0	0x800 ~ 0xFFF, 0x000 ~ 0x7FF
			1	1	0x800 ~ 0xFFF, 0x000 ~ 0x7FF
	1	1	0	0	Forbidden
			0	1	Forbidden
			1	0	Forbidden
			1	1	Forbidden
16x16	0	0	0	0	0x800 ~ 0x9FF
			0	1	0xA00 ~ 0xBFF
			1	0	0xC00 ~ 0xDFF
			1	1	0xE00 ~ 0xFFF
	0	1	0	0	0x800 ~ 0x9FF, 0xA00 ~ 0xBFF
			0	1	0xA00 ~ 0xBFF, 0x800 ~ 0x9FF
			1	0	0x800 ~ 0x9FF, 0xA00 ~ 0xBFF
			1	1	0xA00 ~ 0xBFF, 0x800 ~ 0x9FF
	1	0	0	0	0x800 ~ 0x9FF, 0xA00 ~ 0xBFF
			0	1	0x800 ~ 0x9FF, 0xA00 ~ 0xBFF
			1	0	0xA00 ~ 0xBFF, 0x800 ~ 0x9FF
			1	1	0xA00 ~ 0xBFF, 0x800 ~ 0x9FF
	1	1	0	0	0x800 ~ 0x9FF, 0xA00 ~ 0xBFF, 0xC00 ~ 0xDFF, 0xE00 ~ 0xFFF,
			0	1	0xA00 ~ 0xBFF, 0x800 ~ 0x9FF, 0xE00 ~ 0xFFF, 0xC00 ~ 0xDFF,
			1	0	0xC00 ~ 0xDFF, 0xE00 ~ 0xFFF, 0x800 ~ 0x9FF, 0xA00 ~ 0xBFF,
			1	1	0xE00 ~ 0xFFF, 0xC00 ~ 0xDFF, 0xA00 ~ 0xBFF, 0x800 ~ 0x9FF,

在數位映像方式, BK1 只有 240/256 序號(vectors)是被需要的.

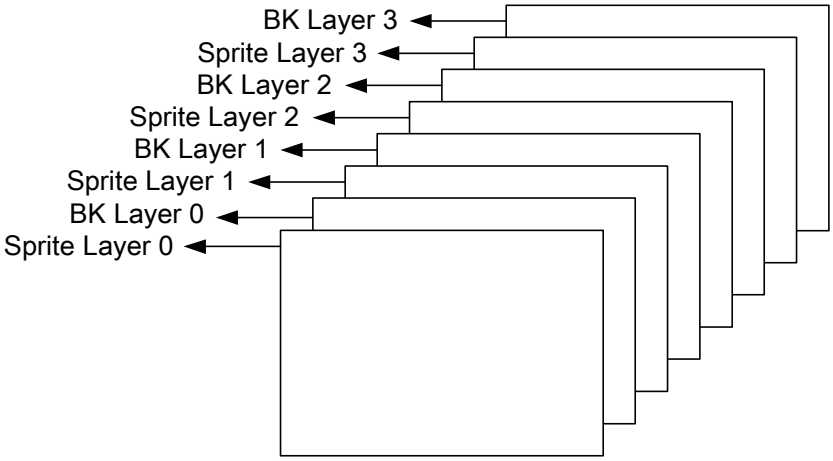
	V_scroll_en	H_scroll_en	Y[8]	X[8]	BK1
16x16	0	0	0	0	0x000 ~ 0x1FF
			0	1	0x200 ~ 0x3FF
			1	0	0x400 ~ 0x5FF
			1	1	0x600 ~ 0x7FF
	0	1	0	0	0x000 ~ 0x1FF, 0x200 ~ 0x3FF
			0	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
			1	0	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
			1	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
	1	0	0	0	0x000 ~ 0x1FF, 0x200 ~ 0x3FF
			0	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
			1	0	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
			1	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF
	1	1	0	0	0x000 ~ 0x1FF, 0x200 ~ 0x3FF, 0x400 ~ 0x5FF, 0x600 ~ 0x7FF,
			0	1	0x200 ~ 0x3FF, 0x000 ~ 0x1FF, 0x600 ~ 0x7FF, 0x400 ~ 0x5FF,
			1	0	0x400 ~ 0x5FF, 0x600 ~ 0x7FF, 0x000 ~ 0x1FF, 0x200 ~ 0x3FF,
			1	1	0x600 ~ 0x7FF, 0x400 ~ 0x5FF, 0x200 ~ 0x3FF, 0x000 ~ 0x1FF,

7.2.2 顯示畫面結構(Display Screen Structure)

TV / LCD 顯示畫面的分辨率為 256 x 240 點.即使 VRAM 畫面的分辨率為 256 x 256 點,部份的背景在電視畫面是看不到的.請參考“Graphic 座標部份”有顯示畫面和 VRAM 畫面相關性的描述.

7.3 深度層次(Depth Layer)

每一個卡通塊或是背景有一個深度層次參數.它定義適當的位置的顯示物體.在 VT1682 有 8 個深度層次可以利用, 4 個給背景而其他的給卡通塊,它們的關係如下面圖表所示.當數個物體重疊在一起時,這個層次有較小的深度數會被顯示. 當兩個背景層有相同的深度層次時, 背景 1(Background1)會被顯示. 當兩個卡通塊有相同的深度層次時,有較低卡通塊數的卡通塊(這個映射地址在卡通塊 RAM, 0~239)會被顯示.



7.4 圖像圖形塊格式(Graphic Pattern Format)

7.4.1 字符(Character)方式

字符(Character) 8x8 方式

在 8x8 方式,在橫向的數據順序為 a,b,c,….h 如下面的圖表所示.

a	b	c	d	e	f	g	h
i	j	k	l	m	n	o	p

字符(Character) 8_H x 16_V 方式

這個數據順序和圖形塊數據除了這個圖形塊數據的長度之外跟 8x8 方式是一樣的.

字符(Character) 16_H x 8_V 方式

在橫向的數據順序為 a,b,c,….h 如下面的圖表所示.

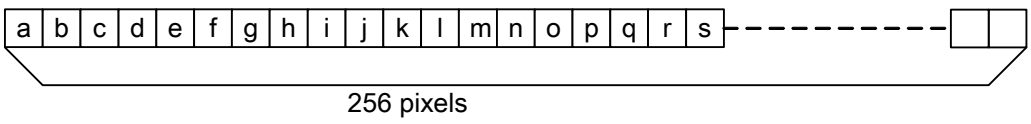
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
q	r	s	t	u	v	w	x								

字符(Character) 16_H x 16_V 方式

這個數據順序和圖形塊數據除了這個圖形塊數據的長度之外跟 16_H x 8_V 方式是一樣的.

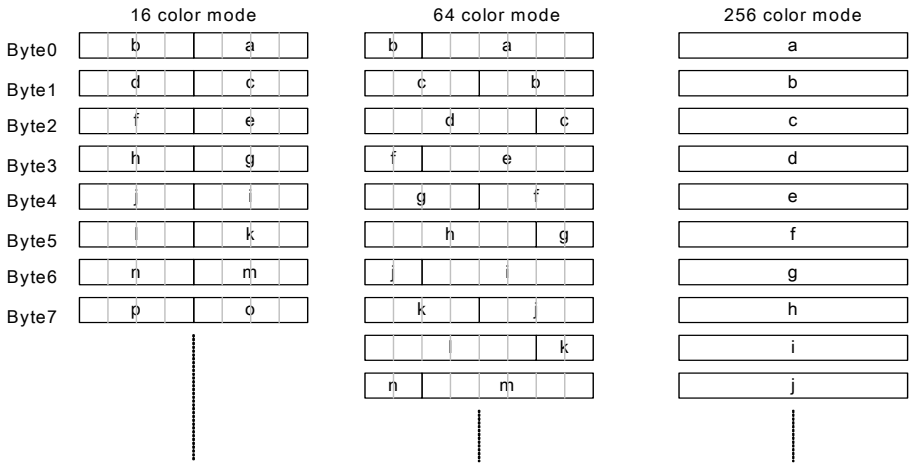
7.4.2 數位映像(Bitmap)方式

在數位映像方式, 這個圖形塊數據的定義是線靠線(line-by-line),它的順序如下面的圖所示.



7.4.3 顏色方式(Color Mode)

在 VT1682 有 4 種顏色方式,它們是 16, 64, 256 和 32768 色方式. 背景 1(Background 1)可以是 16, 64, 256 和 32768 色方式之中的一種. 背景 2(Background 2)可以是 16, 64, 和 256 色方式之中的一種.而卡通塊(Sprite)只有在 16 色方式有效. 這個圖形塊對應不同的顏色方式如下面的圖表所示:



7.4.4 透明的顏色(Transparent Color)

在索引顏色方式,有 16/64/256 色方式,這個透明的索引會是 0. 例如,在 16 色方式下帶有數據 0x00 的點會是透明的.

7.4.5 高顏色方式(High Color Mode)

在高顏色(32768 色) 方式, 每一個點需要兩個字節,它的格式如下:

低字節(Low Byte)

D7 – D5	D4 – D0
Green[2:0]	Blue[4:0]

高字節(High Byte)

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

在這個方式,如在顏色調色板內的內容,每一個點有 5 bits R, G, B 成分. 這個 MSB TRPT 是給透明的標誌使用的.當這個 TRPT 標誌為"1" 時,這個點將會變成透明的.當一個點被設置為透明的表示這些其他的具有最少深度層次的點將會被顯示.如果沒有其他的層次在它的下面,這個 R, G, B 顏色會被顯示.

7.5 顏色調色板(Color Palette)

顏色調色板是被使用來轉換索引顏色為物理的 RGB 顏色. 在 VT1682 有兩個獨立的顏色調色板. 在單一個顯示畫面每一個調色板可以從 32768 色選擇 256 色來顯示.

7.5.1 調色板內容(Palette Content)

這個顏色主要是 R, G, B, 每一個成分分辨率為 5 bits. 如下面的圖表所示,每一個顏色在調色板需要兩個字節來定義, 包括 5-bits Red, 5-bits Green, 5-bits Blue 和一個 DG 標誌給特殊效果.關於這個 DG 的應用請參考圖像特殊效果部份(Graphic Special Effect).

低字節(Low Byte):

D7 – D5	D4 – D0
Green[2:0]	Blue[4:0]

高字節(High Byte):

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

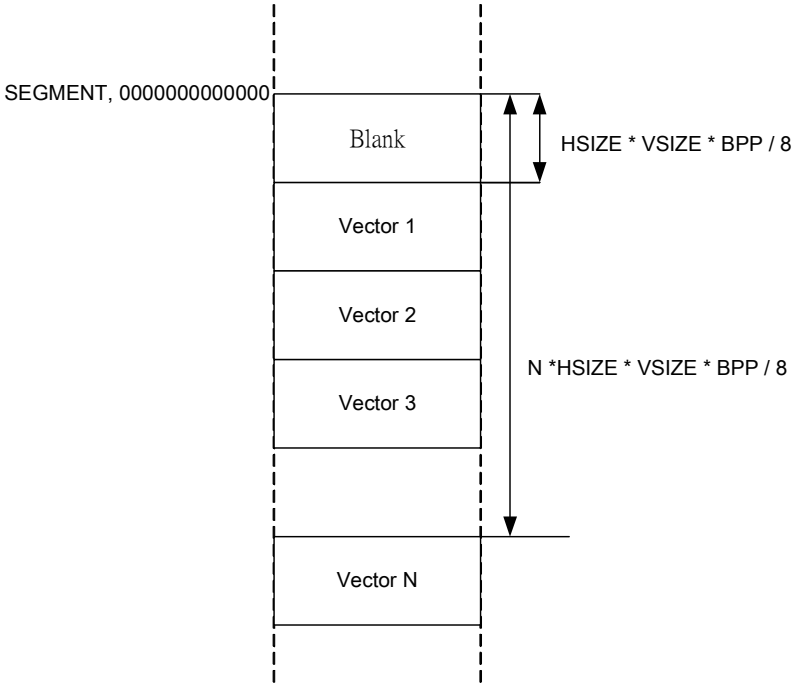
7.5.2 調色板儲存空間(Palette Bank)

在調色板內有 256 色.在 16 色方式,它可以分割成 16 個儲存空間(banks).在 64 色方式, 它可以分割成 4 個儲存空間(banks) .這個儲存空間(bank)是由“PalBank” 參數來定義.每一個圖形塊在 VRAM 有個別的“PalBank” ,定義在 D7:D4 的高字節圖形塊(High byte pattern).

	D7	D6	D5	D4	D3	D2	D1	D0
16 colors mode	PalBank[3:0]				D3 : D0			
64 colors mode	PalBank[3:2]		D5 : D0					
256 colors mode	D7 : D0							

7.6 圖像地址方式(GRAPHIC ADDRESSING MODE)

這個圖像圖形的地址是以 SEGMENT 為起點,序號(Vector)號碼如下面的圖示.有三組的 SEGMENT 給卡通塊(sprites),背景 1(background1 (BK1))和背景 2(background2 (BK2)).



物理地址= (Segment << 13) + Offset

Offset = Vector x H_Size x V_Size x Color_Mode / 8

V_Size 是 8 或是 16,

H_size 是 8 或是 16

Color_Mode 是 4 為 16 色方式, 6 為 64 色方式, 8 為 256 色方式.

Segment = {Segment_H, Segment_L}(Sprite/BK1/BK2)

例如:

如果 BK1 字符數據被開始具有地址 0x20000,

那麼 segment = (0x20000) >> 13 = 0x10

如果 BK1 字符為 8x8 具有256色具有 vector 3,那麼

Offset = 3 x 8 x 8 x 8 / 8 = 0xC0,

這個字符數據會從地址 0x200C0開始.

註釋 1： 在數位映像方式或是高顏方式, V_Size 和 H_Size 兩者都是 16.

註釋 2： 在高顏色方式, Color_Mode 是 8.

註釋 3： 所有的序號(vectors)在高顏色方式都是偶數.

	D7	D6	D5	D4	D3	D2	D1	D0
0x201A	SP_SEGMENT[7:0]							
0x201B					SP_SEGMENT[11:8]			
0x201C	BK1_SEGMENT[7:0]							
0x201D					BK1_SEGMENT[11:8]			
0x201E	BK2_SEGMENT[7:0]							
0x201F	----	----	----	----	BK2_SEGMENT[11:8]			

SP_SEGMENT : Segmentation number (8KB bank) 代表卡通塊(Sprites)

BK1_SEGMENT : Segmentation number (8KB bank) 代表背景 1(BK1)

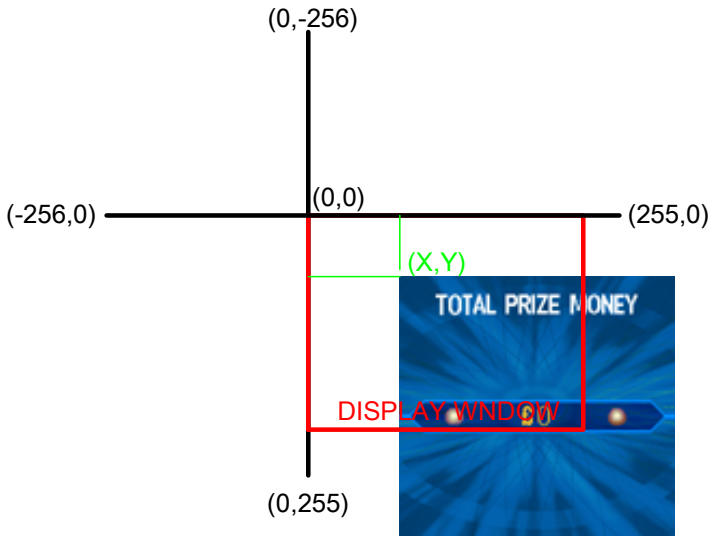
BK2_SEGMENT : Segmentation number (8KB bank) 代表背景 2 (BK2)

7.7 背景層(Background Layer)

在 VT1682 有兩個背景層. 它們是背景 1(background1 (BK1))和背景 2(background2 (BK2)).

7.7.1 座標(coordinate)

背景層的 X-軸是介於 –256 和 255 之間, Y-軸也是. (X, Y) 是表示在 2's 非.當這個 X 是正的(X[8] =0)時,這個顯示背景會捲到右邊. 當這個 Y 是正的(Y[8] = 0)時,這個顯示背景會捲到下面.如下面的圖示:



7.7.2 捲軸(Scrolling)

BKx_X[8:0] 和 BKx_Y[8:0] 是被使用來控制背景層的捲軸. BKx_Scroll_En 參數在每一個背景層提供 4 種捲軸方式.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2010	BK1_X[7:0]							
0x2011	BK1_Y[7:0]							
0x2012				BK1_HCLR	BK1_Scroll_En		BK1_Y8	BK1_X8
0x2014	BK2_X[7:0]							
0x2015	BK2_Y[7:0]							
0x2016					BK2_Scroll_En		BK2_Y8	BK2_X8

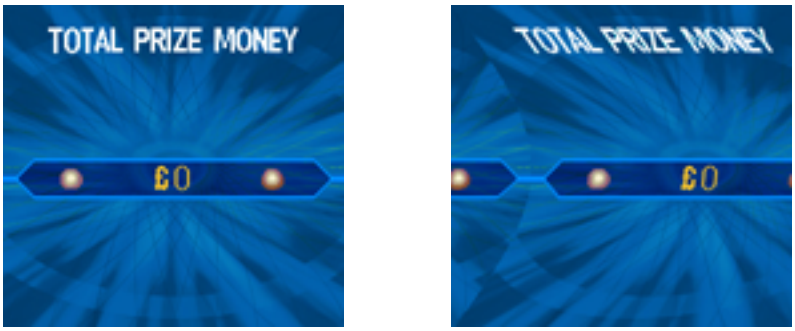
7.7.2.1 畫面捲軸(Frame Scroll)

每一個背景(background)有 9-bits X 和 Y 座標, 提供達到 512x512 捲軸功能.取決於捲軸方式, BK1_Scroll_En 和 BK2_Scroll_En, 它們定義在 0x2012 和 0x2016, 這個捲軸效果將會不同.

捲軸方式(Scrolling Mode)	BK_Scroll_En	Effective X	Effective Y
固定頁方式	0	0 ~ 255	0 ~ 255
橫向兩頁方式	1	-256 ~ 255	0 ~ 255
縱向兩頁方式	2	0 ~ 255	-256 ~ 255
四頁方式	3	-256 ~ 255	-256 ~ 255

7.7.2.2 線捲軸(Line Scroll)

VT1682 提供背景層線對線(line-by-line)橫向的捲軸方式, 如下面的圖示:



線捲軸方式可以應用在 BK1 或是 BK2 兩個之中任一個,或是兩個都用程序來寫寄存器 0x2020 的 BK1_L_EN 和 BK2_L_EN . 240 個字節的捲軸序號會在這個功能致能之前會被存放在介於{ Scroll_Bank, 8'b0000_0000} 和 { Scroll_Bank, 8'b1110_1111}之間的 PRAM. 每一個序號映射到每一條橫向的掃描線. 這個序號值會是介於 –128 和 +127 之間而且它表示在 2's 非.這個序號會被加到背景層的 X.請注意這跟背景縱向的座標 BK_Y 是獨立的.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2020	----	----	BK2_L_EN	BK1_L_En	Scroll_Bank			

BK1_L_En : BK1 線捲軸方式控制

BK2_L_En : BK2 線捲軸方式控制

0 : 無作用 1 : 致能

Scroll_Bank : 當作線捲軸方式序號的 PRAM 存儲空間.

{ Scroll_Bank, 8'b0000_0000} 是當作 1st 線 (上面), 和 { Scroll_Bank, 8'b1111_0111} 是當作 240th 線 (下面).

7.7.3 顏色方式(Color Mode)

在背景層的顏色方式可以是 16, 64 或是 256 色.換句話說, 在背景層的每一個點是表示在 4, 6 或是 8 位. 每一個字符(線)有各自的調色板存儲空間,那個定義在這個 VRAM 的高字節 bit7~bit4,如下面的表格所示.調色板存儲空間(Palette Bank) (PalBank, 參考 **7.5.2 Palette Bank**),有兩種組合當作不同的應用,如下面的表格所示:

背景 1(Background1)

BK1_Pal_Sel	16 色方式	64 色方式	256 色方式
0	Depth = 0x2013[D5:D4]	Depth = 0x2013[D5:D4]	Depth = 0x2013[D5:D4]
	PalBank = VRAM[15:12]	PalBank = VRAM[15:14]	----
1	Depth = VRAM[13:12]	Depth = VRAM[13:12]	Depth = VRAM[13:12]
	PalBank = {0x2013[5:4],VRAM[15:14] }	PalBank = VRAM[15:14]	----

背景 2(Background2)

BK2_Pal_Sel	16 色方式	64 色方式	256 色方式
0	Depth = 0x2017[5:4]	Depth = 0x2017[5:4]	Depth = 0x2017[5:4]
	PalBank = VRAM[15:12]	PalBank = VRAM[15:14]	----
1	Depth = VRAM[13:12]	Depth = VRAM[13:12]	Depth = VRAM[13:12]
	PalBank = { 0x2017[5:4],VRAM[15:14] }	PalBank = VRAM[15:14]	----

7.7.4 字符方式(Character Mode)

BK1_Line = 0; BK1_HCLR=0;

每一個背景層可以被 8x8 字符或是 16x16 字符之中任何一個來形成.當這個字符方式被選擇時, 這個在 0x2013 和 0x2017 的 BKx_Line 必須被置為 0, 而 BKx_Size 是當作字符大小的選擇.在 VRAM 它會拿 32x32 字(words)來定義一個 8x8 字符方式而在 16x16 字符方式為 16x16 字(words).

	D7	D6	D5	D4	D3	D2	D1	D0
0x2013	BK1_EN	BK1_Pal	BK1_Depth		BK1_Color		BK1_Line	BK1_Size
0x2017	BK2_EN	BK2_Pal	BK2_Depth		BK2_Color		----	BK2_Size

7.7.5 數位映像方式(Bitmap Mode)

BK1_Line = 1; BK1_Size = 1; BK1_EN = 1; BK1_HCLR=0

在數位映像方式, 這個背景層被分割成 256 條橫向的(水平的)線.在背景層需 256 條橫向的(水平的)線要 256 字(words) .請注意這個數位映像方式只有在背景 1(Background1)是有效的.

7.7.6 高顏色方式(High Color Mode)

BK1_Line = 1; BK1_Size = 1; BK1_EN = 1; BK1_HCLR=1; BK2_EN = 0;

在背景 1(Background1)高顏色方式是去顯示達到 32768 色.在上面的數位映像描述,給每一個背景層的這個顏色方式可以是 16, 64 或是 256 色.在高顏色方式,每一個點被用兩個字節定義,它們的數據格式如下面的圖示.請注意這個數位映像方式只有在背景 1(Background1)是有效的.當背景 1(background1)是在高顏色方式,背景 2(background2)將會無作用.除此之外,這個在 VRAM 的序號必須是偶數.

數據格式(Data Format):

低字節(Low Byte)

D7 – D5	D4 – D0
Green[2:0]	Blue[4:0]

高字節(High Byte)

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

7.7.7 深度(Depth)

深度參數是去定義顯示的優先順序.有較低優先順序的物件會被顯示在這個畫面.每一個背景層有四個深度層次.在背景的所有的字符(線)會有一個共同的深度參數或是各自的深度,如下面的圖示:

背景 1(Background1)

BK1_Pal_Sel	16 色方式	64 色方式	256 色方式
0	Depth = 0x2013[D5:D4]	Depth = 0x2013[D5:D4]	Depth = 0x2013[D5:D4]
	PalBank = VRAM[15:12]	PalBank = VRAM[15:14]	----
1	Depth = VRAM[13:12]	Depth = VRAM[13:12]	Depth = VRAM[13:12]
	PalBank = {0x2013[5:4],VRAM[15:14] }	PalBank = VRAM[15:14]	----

背景 2(Background2)

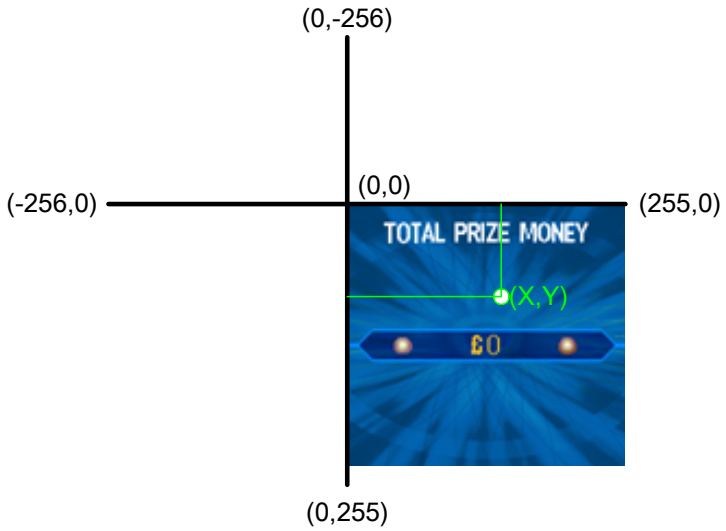
BK2_Pal_Sel	16 色方式	64 色方式	256 色方式
0	Depth = 0x2017[5:4]	Depth = 0x2017[5:4]	Depth = 0x2017[5:4]
	PalBank = VRAM[15:12]	PalBank = VRAM[15:14]	----
1	Depth = VRAM[13:12]	Depth = VRAM[13:12]	Depth = VRAM[13:12]
	PalBank = { 0x2017[5:4],VRAM[15:14] }	PalBank = VRAM[15:14]	----

7.8 卡通塊層(Sprite Layer)

在一個顯示畫面有 240 個卡通塊. 每一個橫向的(水平的)方向線可以顯示 16 個卡通塊,每一個卡通塊有各自的深度, 序號, X, Y, 調色板存儲空間, 調色板選擇參數.

7.8.1 卡通塊座標(Sprite Coordinate)

每一個卡通塊有 9-bits X 和 Y 座標.這個起始座標是在這個顯示畫面的左上角,如下的圖示:



7.8.2 卡通塊大小(Sprite Size)

卡通塊的大小可以是 8x8, 8x16, 16x8 或是 16x16. 所有的卡通塊有相同的字符大小由 0x2018 內的 SP_Size 來定義.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2018					SPALSEL	SP_EN	SP_SIZE	

SP_size : 卡通塊大小(V x H)

0 : 8x8　　1 : 8x16

2 : 16x8　　3 : 16x16

7.8.3 卡通塊控制(Sprite Control)

在 0x2018 設置 SP_EN 使卡通塊顯示致能.如果在橫向的線上的卡通塊數目超過 16 個, 這個在 0x2001 的超過標誌 SP_ERR 會是 “1”. 這個標誌是線對線來修改,而且只有在橫向的空白期間是有效的.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2001(R)	VBLANK	SP_ERR						

SP_ERR : 在單一的橫向的線上超過 16 個卡通塊.

0 : 沒超過　　1 :超過

7.8.4 卡通塊調色板選擇(Sprite Palette Selection)

在 VT1682 有兩個顏色調色板(參考 **7.10 Palette Select**).每一個卡通塊有各自的參數,在卡通塊 RAM 的 PalSel 來選擇顏色的調色板.設置這個 SPALSEL 可以容許所有的卡通塊在同一時間顯示在 Palette1 和 Palette2 .

SPALSEL	PalSel in SpriteRAM	Palette1	Palette2
0	0	Yes	No
	1	No	Yes
1	0/1	Yes	Yes

7.8.5 卡通塊 RAM 的格式(Sprite RAM data format)

每一個卡通塊需要 6 個字節來定義它的屬性,包括序號, X, Y, 深度層次,縱向的快速翻動,橫向的快速翻動,調色板選擇內部的圖形塊方式.它的形態如下面的表格所示.當這個 DMA 被使用來傳送卡通塊 RAM 數據時,這個傳送來源地址順序會是

00—01—02—03—04—05—06—07—08—09—0A—0B—0C--……

目的地卡通塊 RAM 地址順序會是

00—01—02—03—04—05—08—09---0A--0B---0C---0D—10--……

在卡通塊 RAM 的卡通塊數據形態

	D7	D6	D5	D4	D3	D2	D1	D0
SP*8	Vector[7:0]							
SP*8 + 1	Palette[3:0]				Vector[11:8]			
SP*8 + 2	X[7:0]							
SP*8 + 3				Depth[1:0]		Flip[1:0]		X[8]
SP*8 + 4	Y[7:0]							
SP*8 + 5						VRCH	PalSel	Y[8]

* SP 是在卡通塊 RAM 的卡通塊索引數,介於 0 和 239 之間.

7.9 CCIR 層(CCIRLayer)

VT1682 有兩組 CCIR 通信協議給輸入這個圖像的影像,稱之為 CCIR 層.請參考"CCIR 通信協議(Protocol)" 部份有這個信號的相關連結和寄存器的設置. CCIR 會把它當作第三個背景來看,但是只有在 Palette2 是有效的.如背景, CCIR 層有 8 個深度層次(CCIR_Depth) ,它是卡通塊和背景深度層次的合成.

7.9.1 CCIR 顏色效果(Color Effects)

有幾個顏色的特殊效當作 CCIR 層, 例如灰階顏色, 半調色顏色和顏色的罩幕.這個灰階顏色功能是去讓在 CCIR 層的色度(Chrominance)無作用,只有當在 0x202A 的 YUV_RGB 為 0 時,這個功能是有效的. 半調色功能是一半的色度成分.在顏色罩幕功能有三個 RGB 顏色罩幕去產生不同的顏色輸出. CCIR 層也有一些其他的特殊應用,例如影像感應器,這些功能會各自描述在下面的表格.

	D7	D6	D5	D4	D3	D2	D1	D0

0x202A	VS_Phase	HS_Phase	YC_Swap	CbCrswap	SYNCMOD	YUV_RGB	Field_OEn	Field_On
0x202B	R_MSK	G_MSK	B_MSK	HalfTone	Gray	CCIR_Depth		
0x202E	TRC_EN	CCIR_EN	BlueScr_EN	Touch_EN	CCIR_TH			

R_MSK : CCIR 特殊的效果, 紅色(red)成分罩幕控制.

0 : 正常 1 : 無紅色成分

G_MSK : CCIR 特殊的效果, 綠色(green)成分罩幕控制.

0 : 正常 1 : 無綠色成分

B_MSK : CCIR 特殊的效果, 藍色(blue)成分罩幕控制.

0 : 正常 1 : 無藍色成分

HalfTone : CCIR 特殊的效果, 輸入顏色分成一半.

0 : 正常 I 1 :一半

Gray : CCIR 特殊的效果, 改變輸入影像為灰階(gray-levels).

0 : 彩色 1 : 灰階

7.9.2 CCIR 影像拍攝(Image Capture)

CCIR 影像可以拍攝和儲存在外部的 SRAM.在這個拍攝期間,卡通塊不會被影響.只有當 VT1682 在從動裝置(SLAVE)方式時,拍攝功能可以被正確的操作.當在 0x2000 的拍攝命令被設置時,這個影像透過 CCIR 界面會是閃光燈和被儲存在外部的 SRAM.這個拍攝命令可以在 VBLANK 期間 致能/無作用.這個拍攝影像可以是高顏色或是 16 階灰階影像兩者任一.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2000				Capture	SLAVE	---	---	NMI_EN

Capture : 在 CCIR 界面拍攝影像

0 : 無操作 1: 影像拍攝

7.9.2.1 高顏色影像(High Color Image)

這個拍攝資料的格式是 256 x 240 點陣, 每一個點使用一個字(word) (兩個字節),而每一個字(word)包括紅 (D14-D10), 綠(D9-D5) 和 藍(D4-D0). 這些下面的寄存器必須在進入拍攝功能之前要被設置.

BK1_Line = 1; BK1_Size = 1; BK1_EN = 1; BK1_HCLR=1; BK2_EN = 0; BK1_X=9; BK1_Y=0; BK1_Scroll_En=0;

這個拍攝影像將會被儲存在背景 1(Background1)高顏色方式的圖形塊區域.所以在 VRAM 的背景 1(BK1)序號會被設置在高顏色方式的格式, 240 個線序號.

7.10 調色板選擇(Palette Select)

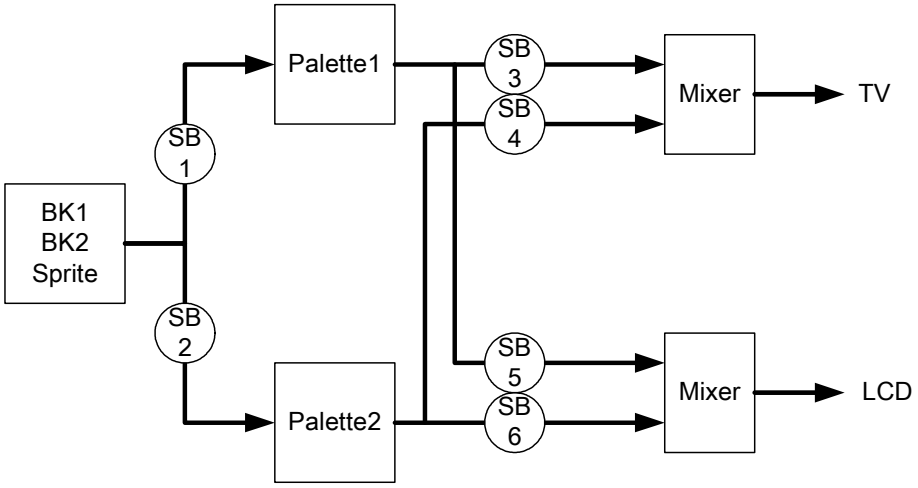
有兩個 256 色的顏色調色板可以支持單一顯示畫面達到 512 色.

背景 1(BK1),背景 2(BK2)和卡通塊(Sprite)可以選擇各自的調色板.

如下面的圖示, SB1 和 SB2 是被使用來選擇調色板.

當 SB1 致能時, 這個調色板 1(Palette 1)被選到.

當 SB2 致能時, 這個調色板 2(Palette 2)被選到.



背景(Background) :

BK1	SB1	0x200F.D0
	SB2	0x200F.D1
BK2	SB1	0x200F.D2
	SB2	0x200F.D3

	D7	D6	D5	D4	D3	D2	D1	D0
0x200F					BK2_Pal_Sel		BK1_Pal_Sel	

BK1_Pal_Sel: 背景 1(BK1)調色板選擇

0 : BK1 無作用　　1 : BK1 使用調色板 1(palette1)

2 : BK1 使用調色板 2(palette2)　　3 : BK1 使用調色板 1 和調色板 2

BK2_Pal_Sel:背景 2(BK2)調色板選擇

0 : BK2 無作用　　1 : BK2 使用調色板 1(palette1)

2 : BK2 使用調色板 2(palette2)　　3 : BK2 使用調色板 1 和調色板 2

卡通塊(Sprite) :

每一個卡通塊有各自的調色板選擇位,那是被在卡通塊 RAM 的 “PalSel” 控制,而這個 SPALSEL 是在 0x2018.當 SPALSEL 是高電平時,所有的 SB1 和 SB2 的卡通塊會有作用,這個 PalSel 會無作用.當 SPALSEL 為低電平時,這個 PalSel 將會定義 SB1 和 SB2,當 PalSel=0 時, SB1 被選擇,反之 SB2 被選擇.

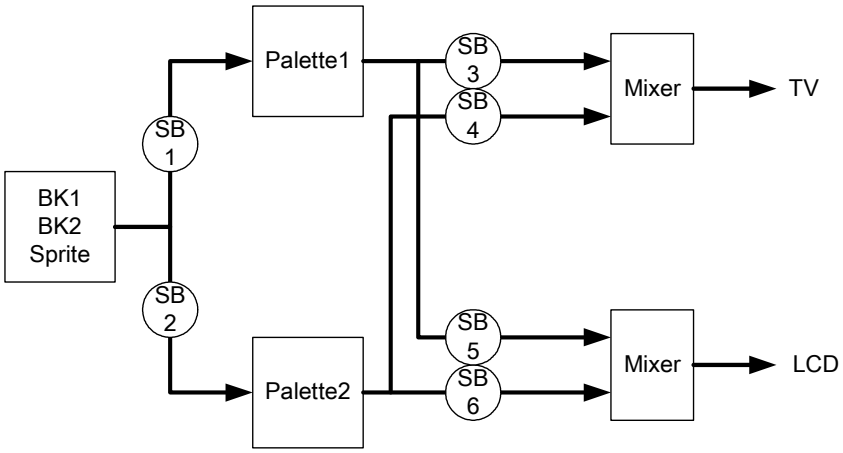
	D7	D6	D5	D4	D3	D2	D1	D0
0x2018					SPALSEL	SP_EN	SP_SIZE	

在卡通塊 RAM 的數據格式

	D7	D6	D5	D4	D3	D2	D1	D0
SP*8	Vector[7:0]							
SP*8 + 1	Palette[3:0]				Vector[11:8]			
SP*8 + 2	X[7:0]							
SP*8 + 3				Layer[1:0]		Flip[1:0]		X[8]
SP*8 + 4	Y[7:0]							
SP*8 + 5						VRCH	PalSel	Y[8]

7.11 輸出選擇(Output Select)

由於有兩個 256 色顏色調色板輸出(調色板 1(Palette1) 和調色板 2(Palette2)),它們可以改方向到兩個各自的輸出通信協議(LCD 或是 TV) ,如下面的圖示. SB3, SB4, SB5 和 SB6 是被用來選擇這個輸出的元件. 當 SB3 被致能時,所有用調色板 1(Palette1)的數據會被顯示在電視上.當 SB5 被致能時,所有用調色板 1(Palette1)的數據會被顯示在 LCD 上. SB3 和 SB5 可以各自被控制.同樣的 SB4 和 SB6 也是.當 SB3 和 SB4 同時被致能時,用調色板 1 和調色板 2 的數據會在電視的調整裝置上被混合.兩種混合方法是有效的, 一種是起點的混合(重疊),是以深度層次為起點.另外一種是顏色的混合.它會從調色板 1 和調色板 2 用 50%的混合比重來混合這些數據.請參考 “圖像混合控制(Graphic Blend Control)” 有詳細的描述.



開關名稱	端口
SB3	0x200E.D1
SB4	0x200E.D3
SB5	0x200E.D0
SB6	0x200E.D2

	D7	D6	D5	D4	D3	D2	D1	D0
0x200E			Blend2	Blend1	Pal2_Out_Sel		Pal1_Out_Sel	

Pal1_Out_Sel: 調色板 1 選擇

0 : 輸出無作用　　1 : 只有輸出到 LCD

2 : 只有輸出到電視　　3 : 輸出到電視和 LCD

Pal2_Out_Sel:調色板 2 輸出選擇

0 : 輸出無作用　　1 : 只有輸出到 LCD

2 : 只有輸出到電視　　3 : 輸出到電視和 LCD

Blend1 : 電視輸出混合致能

0 : 重疊　　1 :混合

Blend2 : LCD 輸出混合致能

0 : 重疊　　1 :混合

7.12 圖像縱向的放大(Graphic Vertical Scaling)

有兩個各自的縱向的放大係數對於背景 1,和背景 2 是有效的.它們是 x1, x1.5 和 x2 ,如下面的圖示,由 0x2019 端口來選擇.

Gain = 1　　Gain = 1.5　　Gain = 2

	D7	D6	D5	D4	D3	D2	D1	D0
0x2019					BK2_Gain		BK1_Gain	

BK2_Gain : 背景 2(BK2)縱向的放大倍率

BK1_Gain : 背景 1(BK1)縱向的放大倍率

0 : x1　　1 : x1

2 : x1.5　　3 : x2

7.13 光槍界面(Light Gun Interface) (脈衝鎖定(Pulse Latch))

VT1682 提供兩組的光槍界面(Light-Gun-Interface) (脈衝鎖定功能(Pulse Latch function))作為雙光槍應用.這兩個輸入的脈衝是經由 **XIOE0** 和 **XIOE1**.在 0x2023 被寫入之後這個 **XIOE0** 右邊的第一個上升邊緣那個座標會被鎖定在 0x2024 和 0x2025.對於 **XIOE1** 也是一樣,那個座標會被鎖定在 0x2026 和 0x2027. 0x2024 和 0x2026 的值會是介於 0 和 119 之間,然而 0x2025 和 0x2027 的值會是介於 0 和 126 之間.當這個座標是(0,0)時,它代表在偵測時無輸入上升邊緣.

這個操作流程會是

1. 設置 **XIOE0** 為輸入方式.(設置 **XIOE1** 為輸入方式,如果需要的話)
2. 在 VBLANK NMI 期間讀取 0x2024 和 0x2025 (0x2026 和 0x2027 如果需要的話).
3. (0x2024 *2) 是 Gun1 的物理的 Y 位置而 (0x2025 *2) 是 X 位置.
4. (0x2026 *2) 是 Gun2 的物理的 Y 位置而 (0x2027 *2) 是 X 位置.
5. 寫任一個值到 0x2023.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2023	Light_Gun_Reset							
0x2024	Light_Gun1_Y							
0x2025	Light_Gun1_X							
0x2026	Light_Gun2_Y							
0x2027	Light_Gun2_X							

7.14 圖像存儲器尋址(Graphic Memory Access)

圖像存儲器包括卡通塊 RAM(卡通塊屬性工作區) 和 VRAM (背景和調色板).它可以由 DMA 功能快速的更新或是由 CPU 用字節-方式來更新.這個 DMA 功能將不會在這裡描述,請參考 “DMA” 部份.

7.14.1 卡通塊 RAM(Sprite RAM)

卡通塊存儲器是用來儲存卡通塊屬性的工作區. SPRAM_ADDR[10:0] 是給 CPU 尋址的卡通塊 RAM 的地址.寫數據到 0x2004 會寫在卡通塊 RAM 具有地址 SPRAM_ADDR.當 0x2004 被寫入時, SPRAM_ADDR 會自動增加.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2002						SPRAM_ADDR[2:0]		
0x2003	SPRAM_ADDR[10:3]							
0x2004(W)	SPRAM_DATA[7:0]							

在卡通塊 RAM 的數據格式

SPRAM_ADDR	D7	D6	D5	D4	D3	D2	D1	D0
SP*8	Vector[7:0]							
SP*8 + 1	Palette[3:0]				Vector[11:8]			
SP*8 + 2	X[7:0]							
SP*8 + 3				Layer[1:0]		Flip[1:0]		X[8]
SP*8 + 4	Y[7:0]							
SP*8 + 5						VRCH	PalSel	Y[8]
SP*8 + 6	Reserved							
SP*8 + 7	Reserved							

Note : SP 是介於 0 和 239 之間的數.

7.14.2 VRAM

VRAM 的尋址跟卡通塊 RAM 一樣.寫數據到 0x2007 會寫在 VRAM 具有地址 VRAM_ADDR. 當 0x2007 被寫入時,VRAM_ADDR 會自動增加.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2005	VRAM_ADDR[7:0]							
0x2006	VRAM_ADDR[15:8]							
0x2007(W)	VRAM_DATA[7:0]							

VRAM 地址映射(VRAM address map)

7.15 特殊效果(Special Effect)

7.15.1 掘取顏色(Dig Color)

在 VT1682 的顏色調色板提供掘取顏色功能,那個功能會從調色板 1 移除顏色去顯示相關的點到調色板 2. 同樣地也可以從調色板 2 移除顏色去顯示相關的點到調色板 1. DG 標誌會在調色板之後表現這個掘取功能在圖像上.不同於透明顏色, DG 標誌是去移除所有層次的顏色和顯示這些顏色在其他的調色板.透明顏色是去顯示這個具有最低層次的顏色在同一個調色板.當這個 DG 為 “0”時,這個點會被顯示具有顏色[R, G, B] 和相關的索引.當 DG 為 “1”時,這個顯示顏色會是[R, G, B] 或是這個從其他調色板顏色的點顏色.參考 t “調色板選擇(Palette Select)” 部份.如果沒有顯示點來自其他的調色板,這個顏色[R, G, B] 會被顯示,不然的話,這個來自其他調色板顏色的點顏色會被顯示.

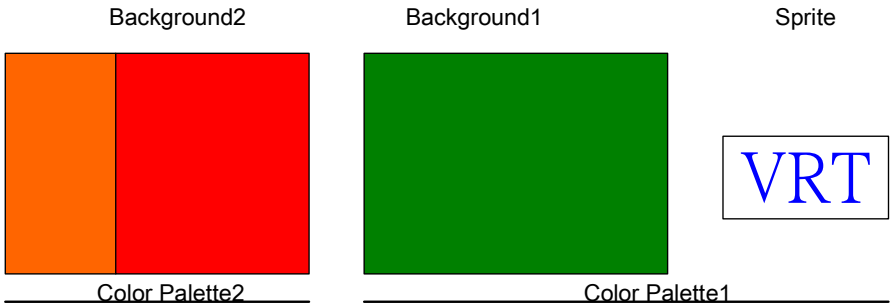
低字節(Low Byte):

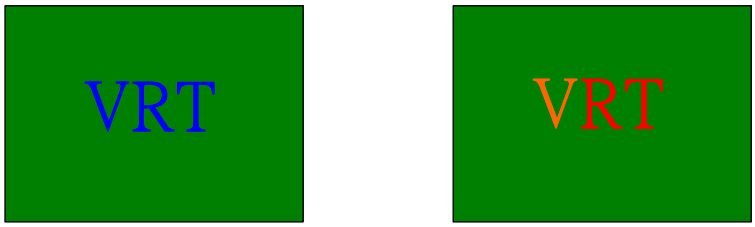
D7 – D5	D4 – D0
綠[2:0]	藍[4:0]

高字節(High Byte):

D7	D6 – D2	D1 – D0
DG	紅[4:0]	綠[4:3]

下面的圖示是一個描述掘取顏色效果的例子.假定那個背景 1(BK1)和卡通塊(Sprite)是來自於顏色調色板 1(Color Palette1),而背景 2(BK2)來自於顏色調色板 2(Color Palette2).當這個在顏色調色板 1 的卡通塊藍色掘取顏色標誌為 0 時,這個顏色調色板 1 的輸出會是藍色的 “VRT”.當這個掘取顏色標誌為 1 時,這個 “VRT” 會變成顏色調色板 2 的顏色(BK2 在這個例子).





DG = 0　　DG = 1

7.15.2 混合效果(Blending Effect)

混合效果是顏色調色板 1 和顏色調色板 2 的顏色混合.這個混合的比率是 50% 的顏色調色板 1 和 50% 的顏色調色板 2.有兩種混合的效果,一個是給電視輸出,另一個是給 LCD.

	D7	D6	D5	D4	D3	D2	D1	D0
0x200E			Blend2	Blend1	Pal2_Out_Sel		Pal1_Out_Sel	

Pal1_Out_Sel: 調色板 1 選擇

0：輸出無作用　　1：只有輸出到 LCD

2：只有輸出到電視　　3：輸出到電視和 LCD

Pal2_Out_Sel:調色板 2 輸出選擇

0：輸出無作用　　1：只有輸出到 LCD

2：只有輸出到電視　　3：輸出到電視和 LCD

Blend1：電視輸出混合致能

0：重疊　　1:混合

Blend2 : LCD 輸出混合致能

0：重疊　　1:混合

7.16 縱向的空白(Vertical Blanking) (NMI)

主 CPU 和 Sound CPU 的 NMI,兩者都是在縱向的空白處.去使 Sound CPU 或是主 CPU 的 NMI 致能, 必須要設置在 0x2000 內的 NMI_EN.當這個 NMI 無作用時,主 CPU 會讀取在 0x2001 內的 VBLANK 標誌來分辨識這個縱向的空白期間.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2000				Capture	SLAVE	---	---	NMI_EN
0x2001(R)	VBLANK	SP_ERR						

NMI_EN : VBLANK NMI 致能控制

0： 無作用　　1： 致能

VBLANK： 縱向的空白處有效期間和讀取來清除 NMI

0： 非-VBLANK 期間　　1 : VBLANK 期間

7.17 圖像顯示畫面大小(Graphic Display Screen Size)

在正常的操作之下,這個顯示圖像的分辨率是 256 x 240 點.當作一些特殊的橫向的捲軸控制時, VT1682 提供另一種顯示畫面大小為 248 x 240 點.這個最左邊的 8 點, 1st ~ 8th 將不會被顯示在畫面上.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2001(W)					EXT_CLK_DIV		SP_INI	BK_INI

SP_INI： 卡通塊顯示畫面大小

BK_INI： 背景顯示畫面大小

0 : 256 x 240 點　　1 : 248 x 240 點

8. I/O

在 VT1682 有 56 個 I/O 可以使用, 40 個由主 CPU 來控制,其他的 16 個是給 Sound CPU 用的.這 40 個 I/O 是 IOA[3:0], IOB[3:0], IOC[3:0], IOD[3:0], IOE[3:0], IOF[3:0], UIOA[7:0] 和 UIOB[7:0]. 它們每一個都有不同的屬性,如下面的表格所示.除此之外,所有的這些 I/O 端口是共用的腳位,它們可以是 GPIO 或是 LCD, UART, SPI, I2C, …… 界面.

	輸出 High	輸出 Low	輸入 Floating	Input w/ Pull High 電阻	Input w/ Pull Low 電阻	共用功能
IOA	O	O	X	O	O	LCD
IOB	O	O	X	O	O	LCD
IOC	O	O	X	O	O	LCD / UART
IOD	O	O	X	O	O	SPI
IOE	O	O	O	X	X	LCD / GunPort
IOF	O	O	O	X	X	ADC / I2C
UIOA	O	O	O	O	O	LCD / CCIR
UIOB	O	O	O	O	O	CSB / Ext IRQ / JoyStick

8.1 IOA

IOA 是和LCD 界面共用.當這個LCD 控制器被使用時,這個IOA_ENB 和 IOA_OE 必須被置為1.在這個 GPIO(IOA_ENB = 0) 方式, IOA 可以是輸出高電平(High),輸出低電平(Low),有拉到高電平(pull-high)電阻的輸入或是有拉到低電平(pull-low)電阻的輸入方式的其中之一,如下面的表格所列:

XIOA[0,1,2,3]	IOAOE = 1	IOAOE = 0
IOAENB = 0	OUTPUT IOA_O	IOA_O[x] = 0 : INPUT w/ pull-low 電阻 IOA_O[x] = 1 : INPUT w/ pull-high 電阻
IOAENB = 1	OUTPUT LCD signals	IOA_O[x] = 0 : INPUT w/ pull-low 電阻 IOA_O[x] = 1 : INPUT w/ pull-high 電阻

	D3	D2	D1	D0
0x210D(W)	IOPBENB	IOBOE	IOAENB	IOAOE
0x210E(W)	IOA_O[3:0]			
0x210E(R)	IOA_I[3:0]			

8.2 IOB

IOB 是和 LCD 界面共用. 當這個 LCD 控制器被使用時, 這個 IOB_ENB 和 IOB_OE 必須被置為 1.在這個 GPIO(IOB_ENB = 0) 方式, IOB 可以是輸出高電平(High),輸出低電平(Low),有拉到高電平(pull-high)電阻的輸入或是有拉到低電平(pull-low)電阻的輸入方式的其中之一,如下面的表格所列:

XIOB[0,2,3]	IOBOE = 1	IOBOE = 0
IOBENB = 0	OUTPUT IOB_O	IOB_O[x] = 0 : INPUT w/ pull-low 電阻 IOB_O[x] = 1 : INPUT w/ pull-high 電阻
IOBENB = 1	OUTPUT LCD	IOB_O[x] = 0 : INPUT w/ pull-low 電阻 IOB_O[x] = 1 : INPUT w/ pull-high 電阻

XIOB[1]	IOBOE = 1	IOBOE = 0
IOBENB = 0	VCOM_OEN=0, OUTPUT IOB_O	IOB_O[1] = 0 : INPUT w/ pull-low 電阻 IOB_O[1] = 1 : INPUT w/ pull-high 電阻
IOBENB = 1	VCOM_OEN=0, OUTPUT LCD VCOM_OEN=1, INPUT LCD	IOB_O[1] = 0 : INPUT w/ pull-low 電阻 IOB_O[1] = 1 : INPUT w/ pull-high 電阻

Note : VCOM_OEN : 0x2022.D5

	D7	D6	D5	D4	D3	D2	D1	D0
0x210D(W)					IOPBENB	IOBOE		
0x210E(W)	IOB_O[3:0]							
0x210E(R)	IOB_I[3:0]							

8.3 IOC

IOC 是和 LCD 及 UART 界面共用. 當這個 LCD 控制器是在 LTPS 或是 ANALOG 方式時或是 UART 界面被使用時, 這個 IOC_ENB 和 IOC_OE 必須被置為 1.在這個 GPIO(IOC_ENB = 0) 方式, IOC 可以是輸出高電平(High),輸出低電平(Low),有拉到高電平(pull-high)電阻的輸入或是有拉到低電平(pull-low)電阻的輸入方式的其中之一,如下面的表格所列:

XIOC[0,1,2]	IOCOE = 1	IOCOE = 0
IOCENB = 0	OUTPUT IOC_O	IOC_O[x] = 0 : INPUT w/ pull-low 電阻 IOC_O[x] = 1 : INPUT w/ pull-high 電阻
IOCENB = 1	OUTPUT LCD/UART	IOC_O[x] = 0 : INPUT w/ pull-low 電阻 IOC_O[x] = 1 : INPUT w/ pull-high 電阻

XIOC[3]	IOCOE = 1	IOCOE = 0
IOCENB = 0	UART_ON=0, OUTPUT IOC_O	IOC_O[3] = 0 : INPUT w/ pull-low 電阻 IOC_O[3] = 1 : INPUT w/ pull-high 電阻
IOCENB = 1	UART_ON=1, INPUT UART:RX	IOC_O[3] = 0 : INPUT w/ pull-low 電阻 IOC_O[3] = 1 : INPUT w/ pull-high 電阻

	D7	D6	D5	D4	D3	D2	D1	D0
0x210D(W)			IOCENB	IOCOE				
0x210F(W)					IOC_O[3:0]			
0x210F(R)					IOC_I[3:0]			

8.4 IOD

IOD 是和 SPI 界面共用. 當這個 SPI 界面是在 Master 方式時, 這個 IOD_ENB 和 IOD_OE 必須被置為 1. 當這個 SPI 界面是在 Slave 方式時, 這個 IOD_OE 必須被置為 0.在這個 GPIO(IOD_ENB = 0)方式, IOD 可以是輸出高電平(High),輸出低電平(Low),有拉到高電平(pull-high)電阻的輸入或是有拉到低電平(pull-low)電阻的輸入方式的其中之一,如下面的表格所列:

XIOD[0,3]	IODOE = 1	IODOE =0
IODENB = 0	OUTPUT IOD_O	IOD_O[x] = 0 : INPUT w/ pull-low 電阻 IOD_O[x] = 1 : INPUT w/ pull-high 電阻
IODENB = 1	OUTPUT SPI	IOD_O[x] = 0 : INPUT w/ pull-low 電阻 IOD_O[x] = 1 : INPUT w/ pull-high 電阻

XIOD[1]	IODOE = 1	IODOE = 0
IODENB = 0	SPI_ON=0, OUTPUT IOD_O SPI_ON=1, INPUT:SPI:DI	IOD_O[1] = 0 : INPUT w/ pull-low 電阻 IOD_O[1] = 1 : INPUT w/ pull-high 電阻
IODENB = 1	SPI_ON=0, OUTPUT IOD_O SPI_ON=1, INPUT:SPI:DI	IOD_O[1] = 0 : INPUT w/ pull-low 電阻 IOD_O[1] = 1 : INPUT w/ pull-high 電阻

XIOD[2]	IODOE = 1	IODOE = 0
IODENB = 0	OUTPUT IOD_O	IOD_O[2] = 0 : INPUT w/ pull-low 電阻 IOD_O[2] = 1 : INPUT w/ pull-high 電阻
IODENB = 1	OUPUT:SPI:DO	IOD_O[2] = 0 : INPUT w/ pull-low 電阻 IOD_O[2] = 1 : INPUT w/ pull-high 電阻

	D7	D6	D5	D4	D3	D2	D1	D0
0x210D(W)	IODENB	IODOEN						
0x210F(W)	IOD_O[3:0]							
0x210F(R)	IOD_I[3:0]							

8.5 IOE

IOE 是和 LCD DAC 輸出及光槍界面共用. 當這個 SPI 界面是在 Master 方式時, 這個 IOD_ENB 和 IOD_OE 必須被置為 1. 當這個 LCD 控制器是在 LTPS 或是 ANALOG 方式或是 LCD DAC 時,光槍功能被使用時, 這個 IOE_OE 必須被置為 0.當這個 ADC 麥克風方式被使用時, 在 0x211E 的 IOEOE3 必須被置為 0,不然的話 IOE 可以是輸出高電平(High),輸出低電平(Low),輸入漂浮(floating)方式的其中之一,如下面的表格所列:

	IOEOE = 1	IOEOE = 0
XIOE[0,1,2]	LCDACEN=0, OUTPUT IOE_O LCDACEN =1, OUTPUT LCDAC	LCDACEN=0, INPUT (12K ohms pull-down resistor) LCDACEN =1, OUTPUT LCDAC

	IOEOE3 = 1	IOEOE3 = 0
XIOE[3]	OUTPUT IOE_O	INPUT floating

	IOEOE = 1, LCDACEN = 0	IOEOE = 0, LCDACEN = 0	LCDACEN = 1
XIOE0	OUTPUT / IOE_O[0]	INPUT / IOE_I[0] /GunPort[0]	OUTPUT / LCD:VR
XIOE1	OUTPUT / IOE_O[1]	INPUT / IOE_I[1] / GunPort[1]	OUTPUT / LCD:VG
XIOE2	OUTPUT / IOE_O[2]	INPUT / IOE_I[2]	OUTPUT / LCD:VB

	IOEOE3 = 1,	IOEOE3 = 0,
XIOE3	OUTPUT / IOE_O[3]	INPUT / IOE_I[3]

	D7	D6	D5	D4	D3	D2	D1	D0
0x214C						----	----	IOEOE
0x214D(W)					IOE_O[3:0]			
0x214D®					IOE_I[3: 0]			
0x211E(W)				IOEOE3				

8.6 IOF

IOF 是和 ADC 輸入及 IIC 界面共用. 當這個 ADC 被使用時, 這些相關的 IOF_OE[x] 必須被置為 0. 當 IIC 被使用時, IOFENB 必須被置為 1. 不然的話, IOF 可以是輸出高電平(High),輸出低電平(Low),輸入漂浮(floating)的方式的其中之一,如下面的表格所列:

	IOF_OE[X] = 1	IOF_OE[X] = 0
XIOF0	OUTPUT:IOF_O[0]	INPUT FLOAT / ADC0
XIOF1	OUTPUTLIOF_O[1]	INPUT FLOAT / ADC1

	IOF_OE[2] = 1	IOF_OE[2] = 0
XIOF2	IOF_ENB=0, OUTPUT:IOF_O[2] IOF_ENB=1, OUTPUT: IIC:SCK	INPUT FLOAT / ADC2

	IOF_OE[3] = 1	IOF_OE[3] = 0
XIOF3	IOF_ENB=0, OUTPUT:IOF_O[3] IOF_ENB=1, INPUT: IIC:SDA	INPUT FLOAT / ADC3

	D7	D6	D5	D4	D3	D2	D1	D0
0x214C					IOFENB			
0x214D(R)	IOF[3:0]				IOE[3:0]			
0x211E(W)					IOFOE3	IOFOE2	IOFOE1	IOFOE0

8.7 UIOA

UIOA 是 bit-方式屬性控制,每一個 UIOA 有一個各自的方向性(IN / OUT)和屬性(拉到高電平(pull high)或是拉到低電平(pull low),漂浮(floating)) 參數,如下面的表格所列. XUIOA 是跟 LCD 和 CCIR 界面共用. 當這個 CCIR 界面有作用時, UIOA_DIR 必須置為輸入漂浮的方式.

DIR	ATTR	DATA_OUT	Description
0	0	0	Input floating
0	0	1	Input floating
0	1	0	Input with pull-low resistor
0	1	1	Input with pull-high resistor
1	0	0	Output low
1	0	1	Output high
1	1	0	Output low
1	1	1	Output high

	D7	D6	D5	D4	D3	D2	D1	D0
0x2129(W)	UIOA_DATA_OUT							
0x2129(R)	UIOA_DATA_IN							
0x212A(W)	UIOA_DIR							
0x212B	UIOA_ATTR							
0x2148(W)							UIOA_MODE	

8.7.1 UIOA 共用功能(UIOA Shared Function)

輸出方式(OUTPUT MODE)

	UIOA_MODE[0]=0	UIOA_MODE[0]=1
XUIOA0	OUTPUT:UIOA_DATA_OUT[0]	OUTPUT:LCD:D0
XUIOA1	OUTPUT:UIOA_DATA_OUT[1]	OUTPUT:LCD:D1
XUIOA2	OUTPUT:UIOA_DATA_OUT[2]	OUTPUT:LCD:D2
XUIOA3	OUTPUT:UIOA_DATA_OUT[3]	OUTPUT:LCD:D3

XUIOA4	OUTPUT:UIOA_DATA_OUT[4]	OUTPUT:LCD:D4

	UIOA_MODE[1]=0	UIOA_MODE[1]=1
XUIOA5	OUTPUT:UIOA_DATA_OUT[5]	OUTPUT:LCD:D5*
XUIOA6	OUTPUT:UIOA_DATA_OUT[6]	OUTPUT:LCD:D6*
XUIOA7	OUTPUT:UIOA_DATA_OUT[7]	OUTPUT:LCD:D7*

*LCD 的詳細描述, 請參考 LCD 部份.

輸入方式(INPUT MODE)

	Function
XUIOA0	UIOA_DATA_IN[0] / CCIR_D0
XUIOA1	UIOA_DATA_IN[1] / CCIR_D1
XUIOA2	UIOA_DATA_IN[2] / CCIR_D2
XUIOA3	UIOA_DATA_IN[3] / CCIR_D3
XUIOA4	UIOA_DATA_IN[4] / CCIR_D4
XUIOA5	UIOA_DATA_IN[5] / CCIR_D5
XUIOA6	UIOA_DATA_IN[6] / CCIR_D6
XUIOA7	UIOA_DATA_IN[7] / CCIR_D7

8.8 UIOB

UIOB 是 bit-方式屬性控制,每一個 UIOB 有一個各自的方向性(IN / OUT)和屬性(拉到高電平(pull high)或是拉到低電平(pull low),漂浮(floating)) 參數,如下面的表格所列. XUIOB 是跟外部的 IRQ,LCD 和 CCIR 界面共用.當這個 CCIR 界面有作用時, UIOB_DIR [2:0] 必須置為輸入漂浮的方式.當外部的 IRQ 有作用時, XUIOB3 必須被置為帶有拉到高電平(pull high)電阻的輸入方式.

0x2148(W)	UIOB_SEL[7:3]		
0x2149(W)	UIOB_DATA_OUT		
0x2149®	UIOB_DATA_IN		
0x214A	UIOB_DIRECTION		
0x214B	UIOB_ATTRIBUTE		

8.8.1 UIOB 共用功能(UIOB Shared Function)

輸出(OUTPUT)

	UIOA_MODE[1]=0	UIOA_MODE[1]=1
XUIOB0	OUTPUT:UIOB_DATA_OUT[0]	OUTPUT:LCD:D8*
XUIOB1	OUTPUT:UIOB_DATA_OUT[1]	OUTPUT:LCD:D9*

	UIOB_MODE[3]=0	UIOB_MODE[3]=1
XUIOB3	OUTPUT:UIOB_DATA_OUT[3]	OUTPUT:JOY_CK

	UIOB_MODE[4]=0	UIOB_MODE[4]=1
XUIOB4	OUTPUT:UIOB_DATA_OUT[4]	OUTPUT:JOY_CK2

	UIOB_MODE[5]=0	UIOB_MODE[5]=1
XUIOB5	OUTPUT:UIOB_DATA_OUT[5]	OUTPUT:CSYNC*

	UIOB_MODE[7]=0	UIOB_MODE[7]=1
XUIOB7	OUTPUT:UIOB_DATA_OUT[7]	OUTPUT:ROMCSB2

8.9 IO 共用腳位表(IO Shared Pin Table)

PIN NAME	OUTPUT	INPUT
XIOA0	LCD	----
XIOA1	LCD	----
XIOA2	LCD	----
XIOA3	LCD	----
XIOB0	LCD	----
XIOB1	LCD	----
XIOB2	LCD	----
XIOB3	LCD	----
XIOC0	LCD	----
XIOC1	LCD	----
XIOC2	UART_TX	----
XIOC3	----	UART_RX
XIOD0	SPI_CKO	SPI_CKI
XIOD1	----	SPI_DI
XIOD2	SPI_DO	----
XIOD3	SPI_CSBO	SPI_CSBI
XIOE0	VR	GunPort
XIOE1	VG	GunPort
XIOE2	VB	----
XIOE3	----	----
XIOF0	----	ADC0
XIOF1	----	ADC1
XIOF2	IIC_SCK	ADC2
XIOF3	IIC_SDA	ADC3
XUIOA0	LCD_D0 /	CCIR_D0
XUIOA1	LCD_D1 / CSTN_D0	CCIR_D1
XUIOA2	LCD_D2 / CSTN_D1	CCIR_D2
XUIOA3	LCD_D3 / CSTN_D2	CCIR_D3
XUIOA4	LCD_D4 / CSTN_D3	CCIR_D4
XUIOA5	LCD_D0 / CSTN_CP	CCIR_D5
XUIOA6	LCD_D1 / CSTN_LP	CCIR_D6

XUIOA7	LCD_D2 / CSTN_FP	CCIR_D7
XUIOB0	LCD_D3 / CSTN_FM	CCIR_CK
XUIOB1	LCD_D4 /	CCIR_HS
XUIOB2	----	CCIR_VS
XUIOB3	JOY_CK	EXT_IRQ
XUIOB4	JOY_CK2	----
XUIOB5	CSYNC	----
XUIOB6	----	----
XUIOB7	ROMCSB2	----

9. DMA

VT1682 提供 4 種存儲器直接尋址(DMA)路徑來加速數據的傳送.它們是外部的存儲器到程序的 RAM, 外部的存儲器到程序的 VRAM,程序的 RAM 到 VRAM 和程序的 RAM 到外部的存儲器.程序的 RAM 包括 4K 字節的主 CPU 的專用的 PRAM 和 4K 字節的共用 RAM,而 VRAM 包括卡通塊 RAM,顏色調色板和背景的 VRAM.它們是由寄存器 0x2122 ~ 0x2128 來控制,如下面的表格所示.這些 DMA_DT_Addr, DMA_SR_Addr 和 DMA_SR_Bank 是以“字節(byte)”為基礎,但是 DMA_Number 是以”字(Word)”為基礎”(兩個字節(Bytes)). DMA 的傳送是由寫 DMA_Number 所發送.這個 DMA 它的目的地是 VRAM 一直到這個縱向的空白(VBLANK) 期間時將不會被開始.換句話說,如果你在非 VBLANK 期間發送一個 VRAM DMA,這個 DMA 將會無作用一直到 VBLANK 開始時.在 DMA 期間這個 CPU 將會停止.這個 DMA_Status 是 DMA 狀態標誌作為你追蹤 DMA 的操作.當你傳送數據到 VRAM 時,不只有這裡的這些 DMA 寄存器你必須去寫它們而且這些寄存器 0x2002, 0x2003, 0x2005 和 0x2006 也要.當這個 DMA 的目的地是卡通塊(Sprite) RAM 時,寄存器端口 0x2002 和 0x2003 是被使用來定義這個目的地地址.當這個 DMA 的目的地是 VRAM 時,寄存器端口 0x2005 和 0x2006 必須被定義.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2122	DMA_DT_Addr[7:0]							
0x2123	DMA_DT_Addr[15:8]							
0x2124	DMA_SR_AddrT[7:0]							
0x2125	DMA_SR_Addr[15:8]							
0x2126	DMA_SR_Bank[22:15]							
0x2127(W)	DMA_Number							
0x2127(R)								DMA_Status
0x2128(W)							DMA_SR_Bank[24:23]	

DMA_DT_Addr : DMA 目的地地址.

DMA_SR_Addr : DMA 來源地址.

註釋 : DMA_SR_Addr[0] 和 DMA_DT_Addr[0] 必須為 0.

DMA_Number : DMA 傳送”字(word)”的數目.

DMA_Status : DMA 狀態標誌.

0 : DMA 準備就續(ready) 1 : DMA 忙碌的(busy.)

DMA Source	DMA Target	來源(Source) 開始地址(Start Address)	目標(Target) 地址(Address)
EXT	PRAM	{DMA_SR_Bank, DMA_SR_Addr[14:0]} (*1)	DMA_DT_Addr
EXT	SpriteRAM	{DMA_SR_Bank, DMA_SR_Addr[14:0]} (*1)	DMA_DT_Addr(*3)
EXT	VRAM	{DMA_SR_Bank, DMA_SR_Addr[14:0]} (*1)	DMA_DT_Addr(*4)
PRAM	SpriteRAM	DMA_SR_Addr[14:0] (*2)	DMA_DT_Addr(*3)
PRAM	VRAM	DMA_SR_Addr[14:0] (*2)	DMA_DT_Addr(*4)
PRAM	PRAM	DMA_SR_Addr[14:0] (*2)	DMA_DT_Addr
PRAM	EXT	DMA_SR_Addr[14:0] (*2)	{DMA_SR_Bank, DMA_DT_Addr[14:0]}(*5)

*1 : DMA_SR_Addr[15] = 1.

*2 : DMA_SR_Addr[15] = 0.

*3 : DMA_DT_Addr = 0x2004.

*4 : DMA_DT_Addr = 0x2007.

*5 : DMA_DT_Addr[15] = 1.

10. 省電功能(POWER SAVING)

為了節省電源損耗,在 VT1682 大部份的模組都是可以分開的來關掉,包括 Sound CPU, SPI, UART, 電視編碼器(TV encoder), LCD 控制器, LVD 模組, Video DAC, Audio DAC, LCD DAC, PLL 和 ADC.在默認的方式下,所有這些模組都是無作用的.

0x2106	---	---	SCPURN	SCPU_ON	SPI_ON	UART_ON	TV_ON	LCD_ON
0x211D	LVDEN			VDAC_EN	ADAC_EN	PLL_EN	LCDACEN	---

LCD_ON : LCD 控制器模組致能控制.

TV_ON : 電視編碼器(TV encoder) 模組致能控制

UART_ON : UART 致能控制

SPI_ON : SPI 模組致能控制

LVDEN : 低電壓檢測(Low Voltage Detect)模組致能控制

VDAC_EN : Video DAC 模組致能控制

ADAC_EN, :Audio DAC 模組致能控制

PLL_EN : 相位鎖定迴路(Phase Lock Loop)模組致能控制

LCDDACEN : 類比(analog)LCD DAC 模組致能控制

SCPU_ON : Sound CPU 致能控制

0 : 無作用 1 : 致能

SCPRN : Sound CPU Reset 控制

0 : Reset Sound CPU 1 : 正常操作

11. 中斷(INTERRUPT)

在 VT1682 有 6 個 IRQ 來源和 5 個 IRQ 序號(vectors).這些 IRQ 來源是外部的 IRQ,定時器(Timer) IRQ, SPCU IRQ, UART IRQ 和 SPI IRQ.它們的 IRQ vectors 如下面的表格所示.當在 IRQ 發生時期當中,這個伺服的優先順序會是 Ext_IRQ > Timer_IRQ > SCPU_IRQ > UART_IRQ > SPI_IRQ.

序號名稱(Vector Name)	序號地址(vector address)
NMI	0x7FFFA, 0x7FFFB
Ext_IRQ	0x7FFFE, 0x7FFFF
Timer_IRQ	0x7FFF8, 0x7FFF9
SCPU_IRQ	0x7FFF6, 0x7FFF7
UART_IRQ	0x7FFF4, 0x7FFF5
SPI_IRQ	0x7FFF2, 0x7FFF3

11.1 非罩幕式中斷(Non Maskable Interrupt (NMI))

這個 NMI 來源是這個圖像縱向的空白(VBLANK).這些詳細的描述請參考 “7.17 縱向的空白(Vertical Blanking)(NMI)” 部份.

11.2 外部的 IRQ(External IRQ)

外部的 IRQ 是來自於 XUIOB3 的下降邊緣觸發.在外部的 IRQ 被致能之前,記得去設置這個 XUIOB3 為帶有 Pull-high 電阻的輸入方式.當這個在 0x2121 的 Ext_MSK 被置為 1 時,在 XUIOB3 的下降邊緣會使 CPU 進入這個 IRQ 伺服慣例定義在 0x7FFFE 和 0x7FFFF.在伺服慣例寫“1” 到 Ext_MSK 會清除這個 IRQ 標誌去等待下一個觸發.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2121	--	--	--	SPI_MSK	UART_MSK	SPU_MSK	IRQ1_MSK	Ext_MSK

Ext_MSK: 外部的 IRQ 致能控制

0 : IRQ 致能　　1 : IRQ 無作用

寫 “1” 到 Ext_MSK 會清除 IRQ 標誌.

11.3 定時器 IRQ(Timer IRQ)

定時器 IRQ(Timer IRQ)的 IRQ 序號(vector), 0x7FFF8 和 0x7FFF9,而且這個 IRQ 是可以罩幕的,是由 0x2121 的 IRQ1_MSK 控制.

請參考 **“6.2 定時器(Timer)”** 部份有詳細的操作.去使這個 Timer IRQ 致能,必須要把在 0x2102 的 TMR_IRQ 和在 0x2121 的 IRQ1_MSK 兩個都先置為 “1” .當這個定時器 IRQ 發生時, 寫 0x2103 的 (Timer_IRQ_Clear) 會清除這個現在的定時器 IRQ 標誌去等待下一個定時器 IRQ.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2102							TMR_IRQ	TMR_EN
0x2121	--	--	--	SPI_MSK	UART_MSK	SPU_MSK	IRQ1_MSK	Ext_MSK
0x2103	Timer_IRQ_Clear							

TMR_IR: 定時器 IRQ 致能控制.

0 : 無作用　　1 : 致能

Timer_IRQ_Clear : 定時器 IRQ 清除控制, 寫入任一數據去清除定時器 IRQ.

11.4 Sound CPU IRQ

IRQ 是被使用來當作 Sound CPU 和主 CPU 之間的通訊.它們兩個都可以在 IRQ 傳送給彼此.

11.4.1 接收 Sound CPU IRQ

當 Sound CPU 送 IRQ 到主 CPU 時, 如果在 0x2121 的 SPU_MSK 被致能,主 CPU 將進入伺服慣例, 由 0x7FFF6 和 0x7FFF7 來定義. 請參考“**14.3.2.2 與主 CPU 通訊**” 部份作為 Sound CPU IRQ.在 Sound CPU IRQ 伺服慣例, 寫“1” 到 SPU_MSK 將會清除 IRQ 標誌一直到下一個 Sound CPU IRQ.

11.4.2 傳送 Sound CPU IRQ

當主 CPU 正準備有來自於 Sound CPU 的需求時,它會送一個 IRQ 去中斷 Sound CPU.在 0x211C 的 SCPU_IRQ 寫一個正的脈衝產生這個 IRQ 脈衝.

	D7	D6	D5	D4	D3	D2	D1	D0
0x211C(W)				SCPUIRQ				

11.5 UART IRQ

有兩個 IRQ 來源來自於 UART,一個是 RX (接收) IRQ 和另一個是 TX (傳送) IRQ.在 0x2119 的 RXIRQEn, TXIRQEn 控制這兩個 IRQ. UART IRQ 也是可以罩幕的是由在 0x2121 的 UART_MSK 控制. 只有當 UART_MSK 為 "1" 時和 RXIRQEn 和 TXIRQEn 其中之一被致能時,這個 UART_IRQ 會發生.只要從 0x211B 去讀取 TX-Status 和 RX_Status 的標誌就可以區分來自於 RX 和 TX 的 IRQ.請參考"**5.3 UART 界面**" 作詳細的 UART 設置.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2119-W	--	CarriEn	UARTEN	TXIRQEn	RXIRQEn	ParityEn	OddEven	9bitmode
0x211B-R	--	--	RxError	TX_Status	RX_Status	ParityErr	--	--
0x2121	--	--	--	SPI_MSK	UART_MSK	SPU_MSK	IRQ1_MSK	Ext_MSK

11.6 SPI IRQ

SPI IRQ 是可以罩幕的,是由在 0x2121 的 SPI_MSK 控制.當這個 SPI 準備就緒去傳送或是準備就緒去接收時,這個 IRQ 會發生.

12. 外部的存儲器控制(EXTERNAL MEMORY CONTROL)

外部的存儲器匯流排, XA[23:0], XD[15:0], XROMCSB, XRAMCSB, XRAMRWB, XROMOEB 當做特殊應用時是可以用程序控制去進入三態(tri-state)方式的.外部的存儲器尋址時間有兩種型式, 180ns 和 80ns 在 VT1682 都是容許的.為了節省應用時的 BOM 費用,有多達三個外部的存儲器 CSB 是可以利用的.

12.1 外部的存儲器匯流排尋址時間(External Memory Bus Access Time)

在默認的方式,CPU 的效能會被圖像所妨礙.更高的圖像品質會導致更低的 CPU 效能.為了解決這個問題, VT1682 容許使用者去更換更快速的外部的存儲器元件(SRAM / FLASH / ROM)來改善 CPU 的效能.除此之外, VT1682 可以改變存儲器匯流排 (XA, XD, XROMCSB, XRAMCSB, XRAMRWB 和 XROMOEB) 進入三態(tri-state)去允許外部的元件對這個存儲器尋址.在這個期間, CPU 可以在 PRAM 工作而且不用關掉顯示畫面並且顯示簡單的圖像在 VRAM.

匯流排尋址頻率(Bus Access Frequency)

兩種類型的尋址速度可利用在 VT1682,由在 0x2105 的 "Double" 來控制.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2105(W)	---	COMR6	TV_SYS_SE:[1:0]		CCIR_SEL	Double	ROM_SEL	PRAM

Double	外部的存儲器尋址時間(Access Time)
0	180 ns
1	80 ns

12.2 匯流排三態控制(Bus Tri-state Control)

VT1682 可以隔離系統匯流排 XA, XD, XROMCSB, XRAMCSB, XRAMRWB 和 XROMOEB.在這個方式下,外部的元件是被允許來對 VT1682 的外部存儲器進行尋址的.當匯流排是在三態(tri-sate)時,在 XA, XROMCSB, XRAMCSB, XRAMRWB 和 XROMOEB 有一個 50K ohm 的拉到高電平(Pull-high)電阻.

	D7	D6	D5	D4	D3	D2	D1	D0
0x210B	TSYNEN	PQ2EN	BUSTRI	CS_Control[1:0]		Program_Bank0_select		

0： 正常操作　　1 : XA 三態(tri-state)方式

12.3 外部的存儲器蕊片選擇信號控制(External Memory Chip Select Signal Control)

VT1682 可以提供達 3 個 CSB 的控制腳給外部的存儲器,它們是 XROMCSB, XRAMCSB 和 XUIOB7. 有四種存儲器映射(memory map)方式給這三支腳,它們是由在 0x210B 的 CS_Control 來控制.當這個 XUIOB7 是被使用當做外部存儲器的 CSB 時,記得去更改它的屬性,設置 UIOB_SEL[7] = 1 和 UIOB_DIR[7] = 1.

	D7	D6	D5	D4	D3	D2	D1	D0
0x210B	TSYNEN	PQ2EN	BUSTRI	CS_Control[1:0]		Program_Bank0_select		

CS_Control	XROMCSB	XRAMCSB	XUIOB7
0	$000000 ~ $FFFFFF	----	----
1	$000000 ~ $7FFFFF	$800000 ~ $FFFFFF	----
2	$000000 ~ $3FFFFF	$40000 ~ $7FFFFF	$800000 ~ $FFFFFF
3	$000000 ~ $1FFFFF	$200000 ~ $3FFFFF	$400000 ~ $7FFFFF

13. 搖桿通信協議(JOYSTICK PROTOCOL)

VT1682 可以提供達到兩組的三條線搖桿通信協議.每一組包括 SCK, SDO 和 SDI.這個 SCK 時鐘會輸出在 XUIOB4 和 XUIOB5.當這個 0x2129 端口被讀取時,一個 200ns 寬的正脈衝會被產生在 XUIOB4. 同樣在 0x212A 端口被讀取時,一個 200ns 寬的正脈衝會被產生在 XUIOB5. XUIOB4 和 XUIOB5 的設置如下面表格所列. SDI 和 SDO 會是任何一個 UIOA 或是除了 UIOB[5:4]之外的任何一個 UIOB.

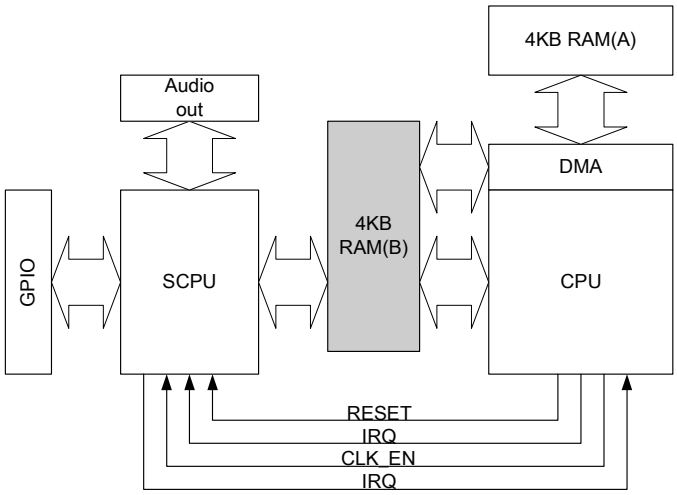
0x2129®	送_JOY_CLK		
0x2129®	UIOA_DATA_IN		
0x2148(W)	UIOB_SEL[7:3]		UIOA_MODE

腳位	信號名稱	設置
XUIOB4	JOY_CK	UIOB_SEL[4] = 1, UIOB_DIR[4] = 1
XUIOB5	JOY_CK2	UIOB_SEL[5] = 1, UIOB_DIR[5] = 1

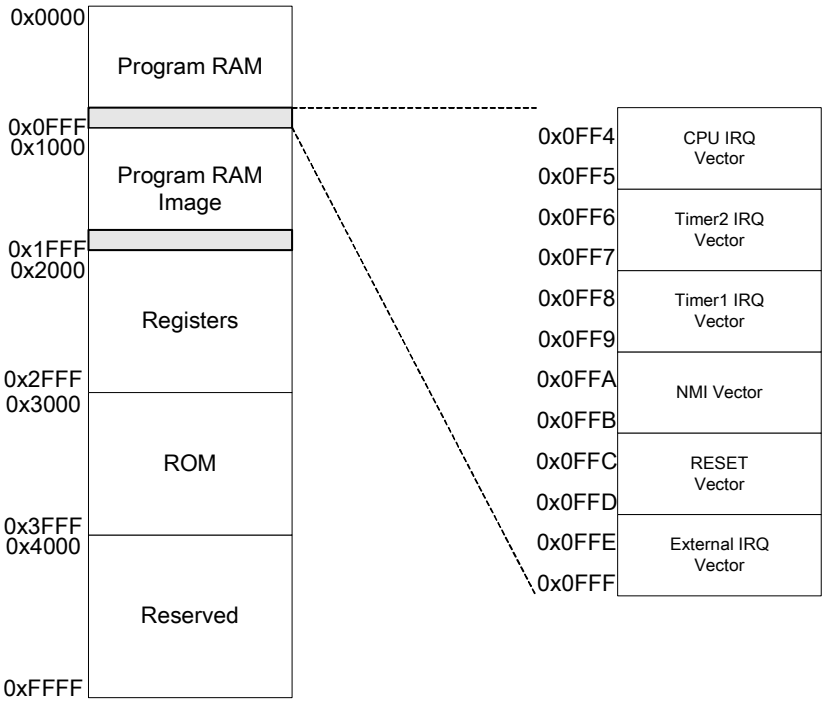
14. SOUND CPU

在 VT1682 有兩個 6502 CPU，一個是主 CPU,另一個是 Sound CPU (SCPU). SCPU 於 NTSC 系統下操作在 21.4772MHz 和於 PAL 系統下操作在 26.6027MHz. SCPU 和主 CPU 共用 4K 字節的 SRAM.主 CPU 和 SCPU 可以透過這個共用的 SRAM 或是 IRQ 信號來通訊.

14.1 結構圖(Block Diagram)



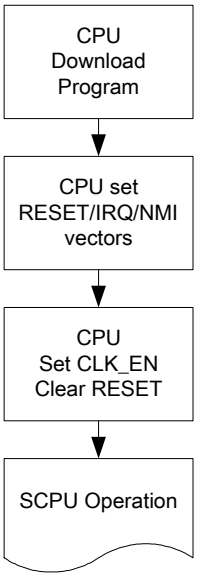
14.2 存儲器映射(Memory Map)



14.3 操作程序(Operation Procedure)

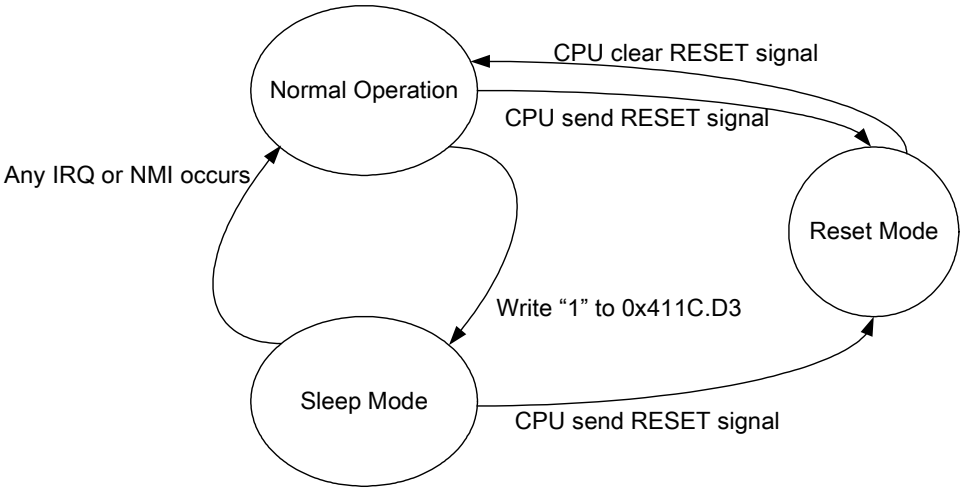
14.3.1 Power-On / Reset 程序

自主 CPU 控制 SCPU,所有在下面圖示的指令,在這一部份是給主 CPU 的.這個 SCPU 的 power-on (reset) 流程如下面的圖所示.主 CPU 必須先下載(download)SCPU 的程序到這個共用的 SRAM,並且在主 CPU 的存儲器設置這個 SCPU 序號(vector)介於 0x1FF4 ~ 0x1FFF 之間.然後使在 0x2106 的 SCPU 時鐘致能,清除這個 reset 標誌, SCPU 會開始工作.



14.3.2. SCPU 操作流程

在 SCPU 有三種操作方式,如下面的圖示.



14.3.2.1 休眠方式(Sleep Mode)

SCPU 可以進入休眠方式去節省電源損耗. 這個 SCPU 的甦醒是由 IRQ 控制,它們是定時器 IRQ, CPU IRQ, NMI 和外部的 IRQ.

注意這個甦醒 IRQ(wakeup IRQ)罩幕必須要打開.

	D7	D6	D5	D4	D3	D2	D1	D0

0x211C(W)				IRQ_OUT	SLEEP	ExtIRQSel	NMI_EN	ExtMask

SLEEP : 寫 “1” 去進入休眠方式

NMI_EN : NMI 罩幕

0 : NMI 不是甦醒(wakeup)來源　　1 : NMI 是甦醒(wakeup)來源

ExtMask : 外部的 IRQ 罩幕

0 :外部的 IRQ 不是甦醒(wakeup)來源　　1 : 外部的 IRQ 是甦醒(wakeup)來源

14.3.2.2 與主 CPU 通訊(Communicate with Main CPU)

有兩種方法作為 SCPU 來與主 CPU 通訊.第一個方法是在共用的 RAM 去定義一個整體的存儲器區域.這個 SCPU 和主 CPU 的共用的存儲器區域是介於 0x1000 和 0x1FFF 之間.另一個方法是 IRQ. 主 CPU 可以在 0x211C 的 SCPUIRQ 寫一個正的脈衝給 SCPU.同樣地, SCPU 也可以在 0x211C 的 IRQ_OUT 寫一個正的脈衝給主 CPU.在 SCPU 的 CPU_IRQ 伺服慣例,為了下一個 CPU IRQ,SCPU 必須要讀取 0x211C 端口去清除 IRQ 標誌.

14.4 定時器(Timer)

在 SCPU 有兩組定時器.

14.4.1 TimerA

寫 Timer_A_PreLoad 來初始化定時器的頻率.寫 0x2101 會重新下載 TimerA_Preload 進入定時器,所以 0x2100 必須要在 0x2101 之前被寫入.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2100	Timer_A_PreLoad[7:0]							
0x2101	Timer_A_PreLoad[15:8]							
0x2102							TMRA_IRQ	TMRA_EN
0x2103	TimerA_IRQ_Clear							

Timer_PreLoad[15:0]: 定時器 IRQ 期間(period)定義.

For NTSC,

Period = (65536 - Timer_A_PreLoad) / 21.4772MHz

Timer_A_PreLoad = 65536 – (Period(sec) * 21.4772 * 10^6)

For PAL

Period = (65536 - Timer_A_PreLoad) / 26.601712MHz

Timer_A_PreLoad = 65536 – (Period(sec) * 26.601712 * 10^6)

TMRA_En : TimerA 致能控制.

0 : 無作用　　1 : 致能

TMRA_IRQ: TimerA IRQ 致能控制.

0 : 無作用　　1 : 致能.

TimerA_IRQ_Clear : TimerA IRQ 清除控制, 寫任一個值來清除 TimerA IRQ.

14.4.2 TimerB

寫 Timer_B_PreLoad 來初始化定時器的頻率.寫 0x2111 會重新下載 TimerB_Preload 進入定時器,所

以 0x2110 必須要在 0x2111 之前被寫入.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2110	Timer_B_PreLoad[7:0]							
0x2111	Timer_B_PreLoad[15:8]							
0x2112							TMRB_IRQ	TMRB_EN
0x2113	TimerB_IRQ_Clear							

Timer_PreLoad[15:0]: 定時器 IRQ 期間(period)定義.

For NTSC,

Period = (65536 - Timer_B_PreLoad) / 21.4772MHz

Timer_B_PreLoad = 65536 – (Period(sec) * 21.4772 * 10^6)

For PAL

Period = (65536 - Timer_B_PreLoad) / 26.601712MHz

Timer_B_PreLoad = 65536 – (Period(sec) * 26.601712 * 10^6)

TMRB_En : TimerB 致能控制.

0： 無作用 1： 致能

TMRB_IRQ: TimerB IRQ 致能控制.

0： 無作用 1： 致能

TimerB_IRQ_Clear : TimerB IRQ 清除控制, 寫任一個值來清除 TimerB IRQ.

14.5 音頻輸出(Audio Output)

有兩種方法從 VT1682 來輸出音頻(Audio).第一種方法是 IIS 界面,如下一個部份的表示.另一種方式是內部的音頻(Audio) DAC.它有 12 位的精度,所以只有這 12 個 MSB [15:4] 會被輸出.請記得在你準備透過音頻(Audio) DAC 輸出音頻之前經由主 CPU 來打開這個 audio DAC.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2118	Audio_DAC_L[7:0]							
0x2119	Audio_DAC_L[15:8]							
0x211A	Audio_DAC_R[7:0]							
0x211B	Audio_DAC_R[15:8]							

Audio_DAC_L[15:0] : Audio DAC 左通道輸出數據.

Audio_DAC_R[15:0] : Audio DAC 右通道輸出數據.

14.6 IIS 界面

VT1682 為更高的音頻質量輸出應用提供 16 位雙工通道的 IIS 輸出. IIS 信號是輸出在 XSCPUIOB6, XSCPUIOB5 和 XSCPUIOB4. 給 IIS 界面致能,請記得確認 IIS_EN 必須置為"1"和 SCPUIOB[6:4]必須設置為輸出方式.

	D7	D6	D5	D4	D3	D2	D1	D0
0x211D							IIS_Mode	IIS_EN

IIS_Mode : IIS 格式選擇

0 : MSB 認可格式　　1 : IIS-bus 格式

IIS_EN : IIS 界面致能控制.

0 : 無作用　　1 : 致能

14.7 IRQ 控制

在 SCPU 有四個 IRQ,它們是外部的 IRQ, Timer-A IRQ, Timer-B IRQ 和 CPU IRQ. 它們的序號 (vector) 地址如下表所列.

IRQ	IRQ vector 地址 (字節)
NMI	0x0FFA, 0x0FFB
Ext_IRQ	0x0FFE, 0x0FFF
TimerA_IRQ	0x0FF8, 0x0FF9
TimerB_IRQ	0x0FF6, 0x0FF7
CPU_IRQ	0x0FF4, 0x0FF5
RESET	0x0FFC, 0x0FFD

除了 CPU_IRQ 之外,每一個 IRQ 有一個罩幕標誌(mask flag). Timer-A 和 Timer-B 的罩幕是在 0x2102 和 0x2112, 而 NMI 和外部的 IRQ 的罩幕是在 0x211C. 當這個罩幕是"0"時,這個 IRQ 不會被 SCPU 偵測到. IRQ_OUT 是 IRQ 輸出到主 CPU 作為兩個 CPU 之間的通訊用.寫一個正的脈衝到 IRQ_OUT 中斷主 CPU.讀取寄存器 0x211C 會清除 CPU IRQ,而且這個動作必須要在 CPU IRQ 伺服慣例來做.不然的話,下一個 CPU IRQ 將不會發生.在 0x211C 的 EXTIRQSel 是用來選擇外部的 IRQ 來源,它可以是來自於腳位 XSCPUIOB7 的低準位觸發兩者其中之一.如果在 SCPU 的 NMI 被使用,在主 CPU(0x2000) 的 NMI 必須被致能.

	D7	D6	D5	D4	D3	D2	D1	D0
0x211C(W)				IRQ_OUT	SLEEP	ExtIRQSel	NMI_EN	ExtMask
0x211C(R)	Clear_CPU_IRQ							

IRQ_OUT :送 IRQ 到 CPU.

0 : 清除 IRQ　　1 : 設置 IRQ

SLEEP :休眠方式致能, 寫 "1" 去進入休眠方式.

註釋 : 任一個發出的 IRQ 會甦醒 CPU.

EXTIRQSel : 外部的 IRQ 來源選擇控制

0 : SCPUIOB[7] 下降邊緣　　1 : 外部的 CPU 寫傳送完成

NMI_EN : NMI 致能控制

0 : 無作用　　1 : 致能

ExtMask : IRQ 罩幕給外部的 IRQ 來源

0 : 外部的 IRQ 無作用　　1 : 外部的 IRQ 致能

Clear_CPU_IRQ : 從 CPU 清除 IRQ 標誌, 寫任一值來清除 IRQ 標誌.

14.8 提高運算單元(Enhanced Arithmetic Unit) –乘法器(Multiplier)和除法器(Divider)

SCPU 有一個專門的 ALU 作為乘法和除法. 這個乘法器是 16 位乘以 16 位,這個除法器是 32 位除以

16 位. 這個乘的運算需要 16 個 CPU 時鐘週期來完成此操作而除的運算需要 32 個.

14.8.1 乘法器(Multiplier) (16x16)

這個乘法的操作是,

ALU_ Multi _operand6, ALU_ Multi _operand5

XI) ALU_operand2, ALU_operand1

= ALU_out4, ALU_out3, ALU_out2, ALU_out1

當 ALU_Multi_operand6 被寫入時,這個操作開始.在乘法運算之後是在 ALU_Multi_operand5 和 ALU_Multi_operand6 的值被改變, 而不是在 ALU_operand1 和 ALU_operand2 的值.

0x2130(W)	ALU_operand1
0x2131(W)	ALU_operand2
0x2132(W)	ALU_operand3
0x2133(W)	ALU_operand4
0x2134(W)	ALU_Multi_operand5
0x2135(W)	ALU__Multi_operand6

0x2130(R)	ALU_out1
0x2131(R)	ALU_out2
0x2132(R)	ALU_out3
0x2133(R)	ALU_out4

14.8.2 除法器(Divider)

這個除法操作是:

ALU_operand4, ALU_operand3, ALU_operand2, ALU_operand1

ALU__Div_operand6, ALU__Div_operand5

= { ALU_out4, ALU_out3, ALU_out2, ALU_out1} + { ALU_out6, ALU_out5}或是+{{ ALU_out6, ALU_out5}*2 - { ALU_out4, ALU_out3, ALU_out2, ALU_out1}}

當 LSB(Least Significant Bit) 為 “1”時, 餘數會是 :

{ ALU_out6, ALU_out5}*2 - { ALU_out4, ALU_out3, ALU_out2, ALU_out1}

當 LSB 為 “0”時, 餘數會是 { ALU_out6, ALU_out5}.

當 ALU__Div_operand6 被寫入時這個操作開始. 在除法運算之後是在 ALU_operand1 和 ALU_operand2, ALU_operand3 和 ALU_operand4 值被改變, 而不是在 ALU_Div_operand5 和 ALU_Div_operand6 的值.

0x2130(W)	ALU_operand1
0x2131(W)	ALU_operand2
0x2132(W)	ALU_operand3
0x2133(W)	ALU_operand4
0x2136(W)	ALU_Div_operand5
0x2137(W)	ALU_Div_operand6

0x2130(R)	ALU_out1
0x2131(R)	ALU_out2
0x2132(R)	ALU_out3
0x2133(R)	ALU_out4
0x2134(R)	ALU_out5
0x2135(R)	ALU_out6

14.9 IO

在 SCPU 有 16 個 IO 口.它們每一個都是位方式控制,它可以是拉到高電平(pull-high)輸入, 拉到低電平(pull-low)輸入,漂浮輸入(floating input), 高電平輸出(output high)或是低電平輸出(output low).次要的 CCIR 輸入, SCPU 外部的 IRQ 和 IIS 界面,都是共用這些 IO 腳位.

DIR	ATTR	DATA	Operation Mode
0	0	0	Input floating
0	0	1	Input floating
0	1	0	Input with pull-low resistor
0	1	1	Input with pull-high resistor
1	0	0	Output low
1	0	1	Output high
1	1	0	Output low
1	1	1	Output high

14.9.1 IOA

IOA 可以是 GPIO 或是次要的 CCIR 界面.當它是作為次要的 CCIR 界面時, 記得要去設置 IOA 為輸入漂浮(input floating)方式.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2140(W)	IOA_Data							
0x2140(R)	IOA_Data							
0x2141	IOA_DIR							
0x2142	IOA_ATTR							

共用腳位表

腳位名稱	信號名稱
XSCPUIOA0	CCIR_D0
XSCPUIOA1	CCIR_D1
XSCPUIOA2	CCIR_D2
XSCPUIOA3	CCIR_D3
XSCPUIOA4	CCIR_D4
XSCPUIOA5	CCIR_D5
XSCPUIOA6	CCIR_D6
XSCPUIOA7	CCIR_D7

14.9.2 IOB

IOB 可以是 GPIO,次要的 CCIR 界面, IIS 或是外部的 IRQ. 當它是 CCIR 時, 這些相關的腳位必須被設置為輸入漂浮方式(input floating). 當外部的 IRQ 被使用時,XSCPUIOB7 必須設置為帶有 Pull-high 電阻的輸入方式. 當 IIS 有作用時, XSPUIOB6, XSPUIOB5 和 XSPUIOB4 必須被設置為輸出方式.

	D7	D6	D5	D4	D3	D2	D1	D0
0x2144(W)	IOB_Data							
0x2144(R)	IOB_Data							
0x2145	IOB_DIR							
0x2146	IOB_ATTR							

共用腳位表

	輸出	輸入
SCPU_IOB1	----	EXT_CPU_CSB / CCIR_HS
SCPU_IOB2	----	CCIR_VS
SCPU_IOB3	----	----
SCPU_IOB4	IIS_CK	----
SCPU_IOB5	IIS_DA	----
SCPU_IOB6	IIS_SW	----
SCPU_IOB7	----	SCPU_IRQN

15. 寄存器表(REGISTER TABLE)

圖像的寄存器(Graphic Registers)

	D7	D6	D5	D4	D3	D2	D1	D0
0x2000				Capture	SLAVE	---	---	NMI_EN
0x2001(W)					EXT_CLK_DIV		SP_INI	BK_INI
0x2001(R)	VBLANK	SP_ERR						
0x2002						SPRAM_ADDR[2:0]		
0x2003	SPRAM_ADDR[10:3]							
0x2004	SPRAM_DATA[7:0]							
0x2005	VRAM_ADDR[7:0]							
0x2006	VRAM_ADDR[15:8]							
0x2007	VRAM_DATA[7:0]							
0x2008	LCD_VS_DELAY							
0x2009	LCD_HS_DELAY							
0x200A	LCD_FR_DELAY[7:0]							
0x200B	CH2_Odd_line_color		CH2_Even_line_color		CH2_SEL	CH2_REV	LCD_FR[8]	LCD_HS[8]
0x200C	F_RATE	DotODR	LCD_CLK		UPS052	Field_AC	LCD_MODE	
0x200D	LCDEN	Dot240	Reverse	VCOM	Odd_line_color		Even_line_color	
0x200E			Blend2	Blend1	Pal2_Out_Sel		Pal1_Out_Sel	
0x200F					BK2_Pal_Sel		BK1_Pal_Sel	
0x2010	BK1_X[7:0]							
0x2011	BK1_Y[7:0]							
0x2012				BK1_HCLR	BK1_Scroll_En		BK1_Y[8]	BK1_X8
0x2013	BK1_EN	BK1_Pal	BK1_Depth		BK1_Color		BK1_Line	BK1_Size
0x2014	BK2_X[7:0]							
0x2015	BK2_Y[7:0]							
0x2016					BK2_Scroll_En		BK2_Y[8]	BK2_X8
0x2017	BK2_EN	BK2_Pal	BK2_Depth		BK2_Color		----	BK2_Size
0x2018					SPALSEL	SP_EN	SP_SIZE	
0x2019					BK2_Gain		BK1_Gain	
0x201A	SP_SEGMENT[7:0]							
0x201B					SP_SEGMENT[11:8]			
0x201C	BK1_SEGMENT[7:0]							
0x201D					BK1_SEGMENT[11:8]			
0x201E	BK2_SEGMENT[7:0]							
0x201F	----	----	----	----	BK2_SEGMENT[11:8]			
0x2020	----	----	BK2_L_EN	BK1_L_En	Scroll_Bank			
0x2021	----	----	Luminance_offset					
0x2022	----	----	VCOMIO	RGB_DAC	CCIR_OUT	Saturation		
0x2023	Light_Gun_Reset							
0x2024	Light_Gun1_Y							
0x2025	Light_Gun1_X							
0x2026	Light_Gun2_Y							
0x2027	Light_Gun2_X							
0x2028	----	----	CCIR_Y					
0x2029	----	----	----	CCIR_X				

0x202A	VS_Phase	HS_Phase	YC_Swap	CbCrswap	SYNCMOD	YUV_RGB	Field_OEn	Field_On
0x202B	R_EN	G_EN	B_EN	HalfTone	B/W	CCIR_Depth		
0x202E	TRC_EN	CCIR_EN	BlueScr_EN	Touch_EN	CCIR_TH			
0x2030	----	VDACSW	VDAC_OUT[5:0]					
0x2031	----	----	RDACSW	RDAC_OUT[4:0]				
0x2032	----	----	GDACSW	GDAC_OUT[4:0]				
0x2033	----	----	BDACSW	BDAC_OUT[4:0]				

系統的寄存寄存器(System Registers)

	D7	D6	D5	D4	D3	D2	D1	D0
0x2100(W)					Program_Bank1_Register3			
0x2100(R)					Program_Bank1_Register3			
0x2101(W)	Timer_Preload							
0x2101(R)	Timer_Preload							
0x2102							TMR_IRQ	TMR_EN
0x2103	Timer_IRQ_Clear							
0x2104(W)	Timer_Preload[15:8]							
0x2104(R)	Timer_Preload[15:8]							
0x2105(W)	---	COMR6	TV_SYS_SE:[1:0]		CCIR_SEL	Double	ROM_SEL	PRAM
0x2106	---	---	SCPURN	SCPU_ON	SPI_ON	UART_ON	TV_ON	LCD_ON
0x2107(W)	Program_Bank0_Register0							
0x2107(R)	Program_Bank0_Register0							
0x2108(W)	Program_Bank0_Register1							
0x2108(R)	Program_Bank0_Register1							
0x2109(W)	Program_Bank0_Register2							
0x2109(R)	Program_Bank0_Register2							
0x210A(W)	Program_Bank0_Register3							
0x210A(R)	Program_Bank0_Register3							
0x210B	TSYNEN	PQ2EN	BUSTRI	CS_Control[1:0]		Program_Bank0_select		
0x210C(W)					Program_Bank1_Register2			
0x210C(R)					Program_Bank1_Register2			
0x210D	IODENB	IODOEN	IOCENB	IOCOE	IOPBENB	IOBOE	IOAENB	IOAOE
0x210E(W)	IOB_O[3:0]				IOA_O[3:0]			
0x210E(R)	IOB_I[3:0]				IOA_I[3:0]			
0x210F(W)	IOD_O[3:0]				IOC_O[3:0]			
0x210F(R)	IOD_I[3:0]				IOC_I[3:0]			
0x2110(W)					Program_Bank1_Register0			
0x2112(R)					Program_Bank1_Register0			
0x2111(W)					Program_Bank1_Register1			
0x2113(R)					Program_Bank1_Register1			
0x2112(W)	Program_Bank0_Register4							
0x2110(R)	Program_Bank0_Register4							
0x2113(W)	Program_Bank0_Register5							
0x2111(R)	Program_Bank0_Register5							
0x2114	Baud_rate[7:0]							
0x2115	Baud_rate[15:8]							
0x2116	16bitMode	SPIEN	SPI_RST	M/SB	CKPHASE	CKPOLAR	CK_FREQ[1:0]	
0x2117(W)	SPI_TX_Data							

0x2117(R)	SPI_RX_Data							
0x2118(W)	Program_Bank1_Register5				Program_Bank1_Register4			
0x2118(R)	Program_Bank1_Register5				Program_Bank1_Register4			
0x2119(W)	--	CarriEn	UARTEN	TxIRQEn	RxIRQEn	ParityEn	OddEven	9bitmode
0x211A(W)	Tx_Data[7:0]							
0x211A(R)	Rx_Data[7:0]							
0x211B	Carrier_frequency[7:0]							
0x211B(R)	--	--	RxError	Tx_Status	Rx_Status	ParityErr	--	--
0x211C	AutoWake	KeyWake	EXT2421EN	SCPUIRQ	SLEEPM	----	SLEEPSel	CLKSEL
	Clear_SCPU_IRQ							
0x211D(W)	LVDEN	LVDS1	LVDS0	VDAC_EN	ADAC_EN	PLL_EN	LCDACEN	---
0x211D®	----	-----	----	----	----	----	----	LVD
0x211E(W)	ADCEN	ADCS1	ADCS0	UNUSE	IOFOEN3	IOFOEN2	IOFOEN1	IOFOEN0
0x211E®	ADC_Data[7:0]							
0x211F	VGCEN	VGCA6	VGCA5	VGCA4	VGCA3	VGCA2	VGCA1	VGCA0
0x2120	SLEEP_PERIOD							
0x2121	--	--	--	SPI_MSK	UART_MSK	SPU_MSK	TMR_MSK	Ext_MSK
0x212E						Clear_Ext		Clear_SPU
0x2122	DMA_DT_Addr[7:0]							
0x2123	DMA_DT_Addr[15:8]							
0x2124	DMA_SR_AddrT[7:0]							
0x2125	DMA_SR_Addr[15:8]							
0x2126	DMA_SR_Bank[22:15]							
0x2127(W)	DMA_Number							
0x2127®								DMAStatus
0x2128							DMA_SR_Bank[24:23]	
0x2129®	Send_JOY_CLK							
0x2129(W)	UIOA_DATA_OUT							
0x2129®	UIOA_DATA_IN							
0x212A®	Send_JOY_CLK2							
0x212A(W)	UIOA_DIRECTION							
0x212B	UIOA_ATTRIBUTE							
0x212C(W)	Pseudo_random_number_seed							
0x212C®	Pseudo_random_number							
0x212D(W)	PLL_B				PLL_M	PLL_A		
0x2130(W)	ALU_operand1							
0x2131(W)	ALU_operand2							
0x2132(W)	ALU_operand3							
0x2133(W)	ALU_operand4							
0x2134(W)	ALU_Multi_operand5							
0x2135(W)	ALU__Multi_operand6							
0x2136(W)	ALU_Div_operand5							
0x2137(W)	ALU_div_operand6							
0x2130®	ALU_out1							
0x2131®	ALU_out2							
0x2132®	ALU_out3							
0x2133®	ALU_out4							

0x2134®	ALU_out5						
0x2135®	ALU_out6						
0x2140	IIC_ID						
0x2141	IIC_ADDR						
0x2142(W)	IIC_DATA						
0x2142(R)	IIC_DATA						
0x2143							IIC_CLK_SEL
0x2148(W)	UIOB_SEL[7:3]						UIOA_MODE
0x2149(W)	UIOB_DATA_OUT						
0x2149®	UIOB_DATA_IN						
0x214A	UIOB_DIRECTION						
0x214B	UIOB_ATTRIBUTE						
0x214C			KeyChangeEN	IOFEN	----	----	IOEOEN
0x214D(W)	IOF[3:0]			IOE[3:0]			
0x214D®	IOF[3:0]			IOE[3:0]			

SCPU 寄存器(SCPU Registers)

	D7	D6	D5	D4	D3	D2	D1	D0
0x2100	Timer_A_PreLoad[7:0]							
0x2101	Timer_A_PreLoad[15:8]							
0x2102							TMRA_IRQ	TMRA_EN
0x2103	Timer_A_IRQ_Clear							
0x2110	Timer_B_PreLoad[7:0]							
0x2111	Timer_B_PreLoad[15:8]							
0x2112							TMRB_IRQ	TMRB_EN
0x2113	Timer_B_IRQ_Clear							
0x2118	Audio_DAC_L[7:0]							
0x2119	Audio_DAC_L[15:8]							
0x211A	Audio_DAC_R[7:0]							
0x211B	Audio_DAC_R[15:8]							
0x211C(W)				IRQ_OUT	SLEEP	ExtIRQSel	NMI_EN	ExtMask
0x211C(R)	Clear_CPU_IRQ							
0x211D							IIS_Mode	IIS_EN
0x211E(W)								
0x211E(W)								
0x2130(W)	ALU_operand1							
0x2131(W)	ALU_operand2							
0x2132(W)	ALU_operand3							
0x2133(W)	ALU_operand4							
0x2134(W)	ALU_Multi_operand5							
0x2135(W)	ALU__Multi_operand6							
0x2136(W)	ALU_Div_operand5							
0x2137(W)	ALU_div_operand6							
0x2130(R)	ALU_out1							
0x2131(R)	ALU_out2							
0x2132(R)	ALU_out3							
0x2133(R)	ALU_out4							

0x2134(R)	ALU_out5
0x2135(R)	ALU_out6
0x2140(W)	IOA_Data
0x2140(R)	IOA_Data
0x2141	IOA_IO_DIR
0x2142	IOA_R_PLH
0x2144(W)	IOB_Data
0x2144(R)	IOB_Data
0x2145	IOB_IO_DIR
0x2146	IOB_R_PLH