乐升半导体
LEVETOP

LT758x

High Performance TFT-LCD Graphics Controller

Data Sheet

V1.2

www.levetop.cn

Levetop Semiconductor Co., Ltd.

Revision

Version	Date	Description
V1.0	2024/11/8	● Preliminary version
V1.2	2025/06/12	● Update REG[06h], REG[08h], REG[0Ah] ● Update Application Circuit

Copyright

Contents

Figure

Table

1. Introduction

LT758x is a series of high-performance TFT-LCD graphic accelerated display controllers. It supports 16bits(5/6/5) and 24bits(8/8/8) RGB LCD panels with reolutions ranging from 480*480 to 1280*1024 (SXGA), and the supported color depth is up to Alpha RGB:8888. This series includes three chips: LT7586, LT7583, and LT7580, with the packaging of LQFP-128, LQFP-100, and QFN-80 respectively.

LT758x supports a variety of MCU interfaces, including SPI, I2C, and 8/16bits parallel interfaces. It has a built-in 128Mb Display RAM, which can support display colors of 16/24-bits per pixel. With built-in geometric drawing engine, LT758x supports drawing lines, curves, ellipse, triangle, rectangle, rounded rectangle, and other functions. In addition, LT758x has an embedded hardware graphics acceleration engine (BTE), which provides command-type graphic operations such as Picture-in-Picture, graphics blending, transparent display, and much more. Also, by utilizing the external QSPI Flash and the built-in JPG decoder, LT758x can even refresh the display (1024*600) at the rate of more than 25 frames per second. These functions can greatly enhance the display performance without increasing the MCU workload.

The powerful LT758x is ideal for embedded systems with TFT-LCD displays such as smart home appliances, industrial controls, electronic instruments, medical devices, human-machine interfaces, industrial equipment, inspection equipment, charging stations, multi-function machines, elevator, check-in gate, etc.

1.1 Internal Block Diagram

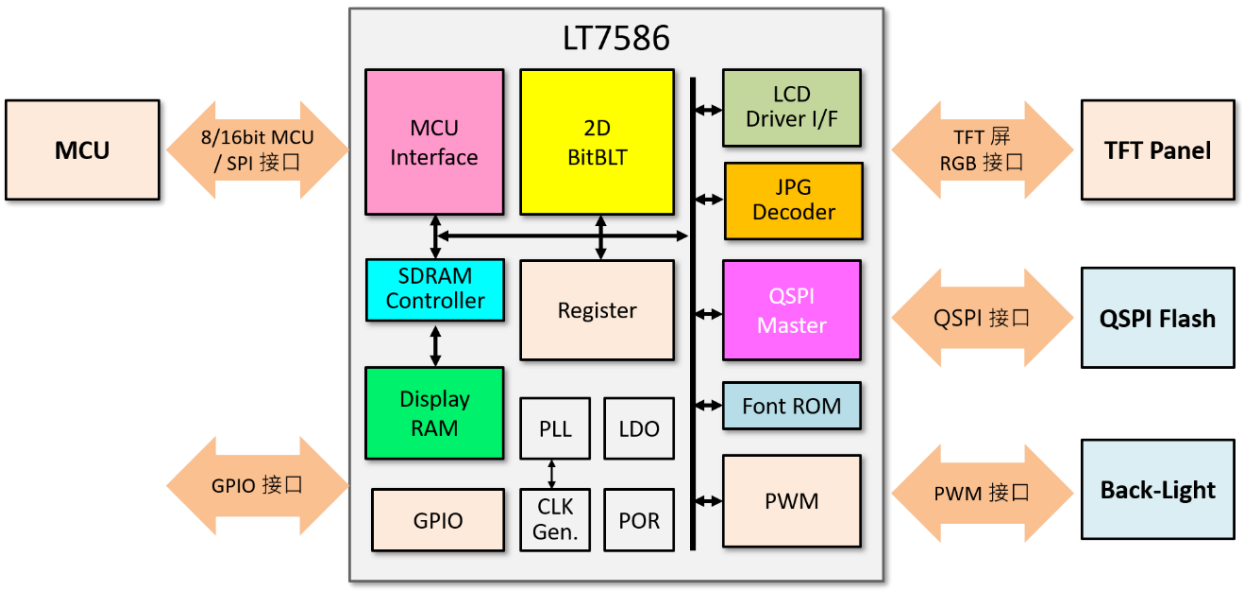


Figure 1-1: LT7586 Internal Block Diagram

1.2 System Block Diagram

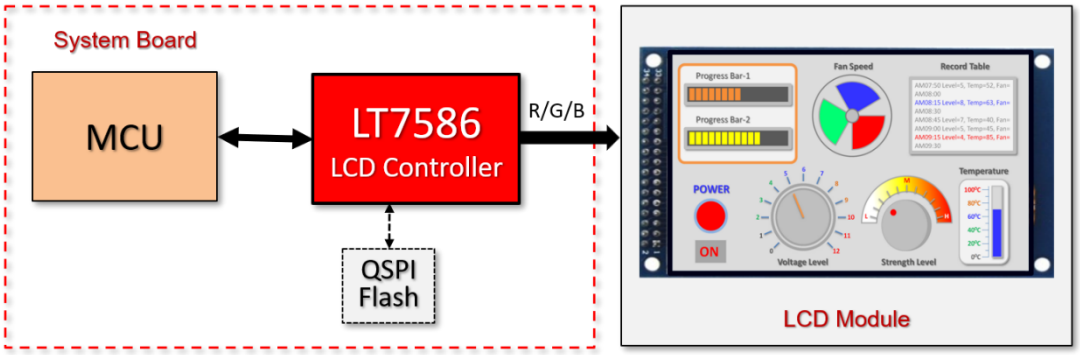


Figure 1-2: LT7586 Designed on System Board

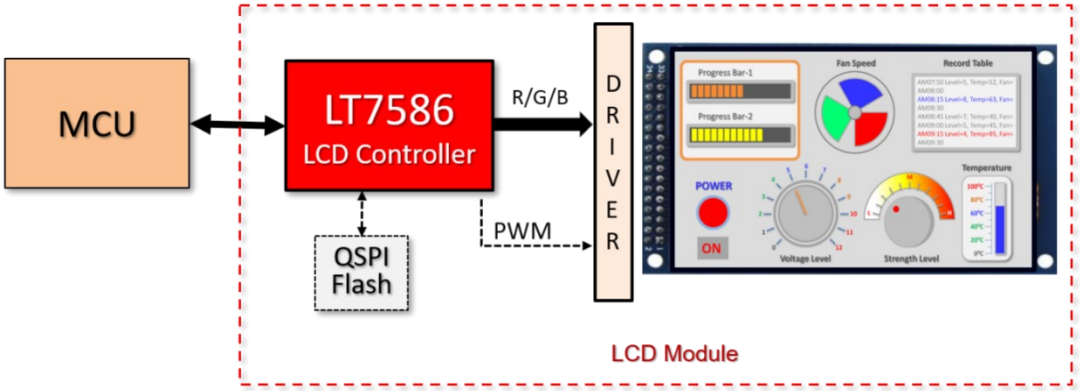


Figure 1-3: LT7586 Designed on TFT-LCD Module

1.3 Model Name

Table 1-1: Model Selection

Model	Package	Embedded Display RAM	Resolution	Colors
LT7586	LQFP-128	128Mb	1280*1024	16.7M
LT7583	LQFP-100	128Mb	1024*768	16.7M
LT7580	QFN-80	128Mb	1024*768	16.7M

1.4 Features

Host Interface

- Support 8/16bits Paraller Interface
  - 8080 Bus Interface
  - 6800 Bus Interface
- Support 3-wire & 4-wire SPI
- Support I^2C Bus Interface

Display RAM

- Built-in 128Mb

Display Data Formats

- 16bpp：RGB 5:6:5（2bytes/Pixel）
- 24bpp：RGB 8:8:8（3bytes/Pixel or 4bytes/Pixel）
  - αRGB 4:4:4:4（4096 Colors/Pixel with Opacity Attribute）
- **32bpp：αRGB 8:8:8:8（4bytes/Pixel）**

Panel Interface & Resolution

- Support 16/24bits RGB panel
- Support Resolution：
  - WQVGA：**480*480,** 16/24bits TFT Panel
  - VGA：**640*480,** 16/24bits TFT Panel
  - WVGA：**800*480,** 16/24bits TFT Panel
  - SVGA：**800*600,** 16/24bits TFT Panel
  - QHD：**960*540,** 16/24bits TFT Panel
  - WSVGA：**1024*600,** 16/24bits TFT Panel
  - XGA：**1024*768,** 16/24bits TFT Panel
  - SXGA：**1280*1024,** 16/24bits TFT Panel

Display Functions

- **Built-in JPG decoder**
- Multiple Display Buffer: Multi buffering allows the main display window to be switched among buffers. Multi buffering allows a simple animation display to be performed by switching the buffers
- Virtual Display: Virtual display can be utilized to show an image which is larger than the LCD panel size. The image may scroll easily in any direction.
- Mirror and Rotation Functions are Available for Image Data Writes
- Picture-in-Picture(PIP)：Support two PIP windows area. Enabled PIP windows are always displayed on top of the Main window. The PIP1 window is always on top of the PIP2 window.
- Provide four User-defined 32*32 Pixels Graphic Cursor.
- Wake-up Display: To quickly show the display data stored in the Display RAM. This feature is used when returning from the Standby mode and Suspend mode.
- Color Bar: To display color bars on the panel directly. Default resolution is 640 * 480.

Bit Block Transfer Engine（BitBLT）

- Built-in 2D BitBLT Engine
- Copy Imagewith Raster Operators
- Color Depth Conversion
- Solid Fill & Pattern Fill:
  - Provide User-defined Patterns with 8*8 pixels and 16*16 pixels
- Opacity (Alpha-Blend) Control: It blends two images and then generates a new image:
  - Chroma-Keying Function: Mixes images with applying the specified RGB color according to the transparency rate.
  - Window Alpha-Blending Function: Mixes two images based on the transparency rate in the specified region.
  - Dot Alpha-Blending Function: Mixes images according to the transparency rate when the target is a graphic image in the RGB format.

Geometric Drawing Engin

- Provide Smart Drawing Features: Line, Rectangle, Triangle, Polygon, Poly-Line, Circle, Ellipse, Arc, and Rounded-Rectangle.

Text Features

- Embedded 8*16, 12*24, 16*32 Character Sets of ISO/IEC 8859-1/2/4/5
- User-defined Characters Support Half Size & Full Size for 8*16, 12*24 and 16*32
- Programmable Text Cursor
- Character Enlargement Function *1, *2, *3, *4 for Horizontal/Vertical Direction
- Support Character to Rotate 90 Degrees

QSPI Master Interface

- Support direct data transfer from the external Serial Flash to the frame buffer.
- **Compatible with standard QSPI spec (NOR/NAND Flash)**
- Provide 16bytes Read FIFO and 16bytes Write FIFO.
- Provide Interrupt when Tx FIFO is empty and SPI Tx/Rx engine is idle.
- **Support NandFlash Bad Block Management**.
- **Supoort the Host MCU to direct access SPI Flash (by-pass mode).**

I2C Interface

- Support I2C interface to connect with external devices.
- Support Standard Mode (100kbps) and Fast Mode (400kbps)

PWM Interface

- Embedded two 16bits Timers
- Programmable Duty control of Output Waveform

GPIO

- Provide up to 28 GPIO interface.

Clock Source

- Embedded Programmable PLL for Core Clock, LCD Pixel Clock, and Fram Buffer Clock.

Power Saving

- Support three kind of Power Saving Mode: Standby, Suspend, and Sleep Mode.
- Support Wakeup function by the Host and External Events.

Reset

- Provide Power On Reset, External Hardware Reset, and Software Command Reset.

Power Supply

- VDD：3.3V +/- 0.3V
- Embedded 1.2V LDO

Package

- LQFP-128Pin
- LQFP-100Pin
- QFN-80Pin

Temperature

- -40℃~85℃

1.5 Pin Assignment

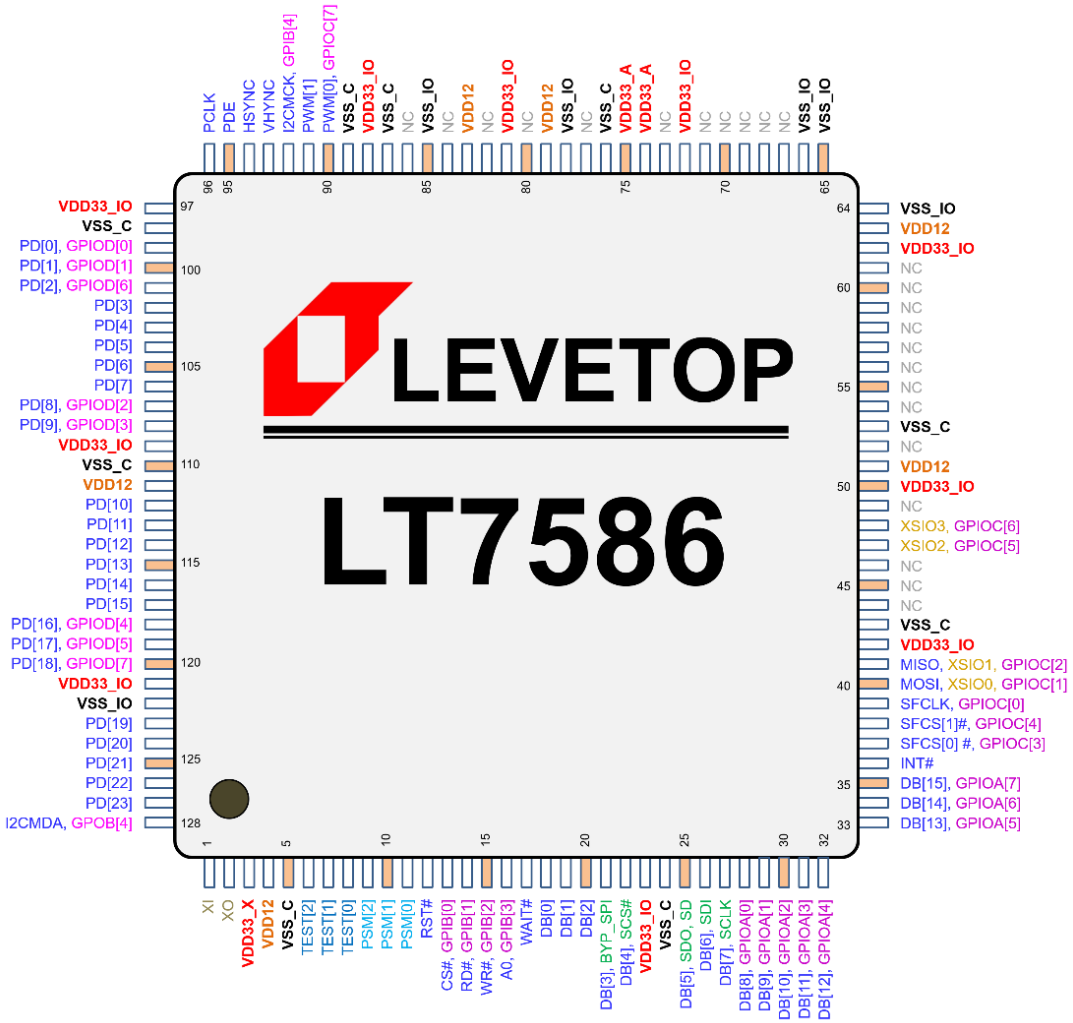


Figure 1-4: LT7586 Pin Assignment (LQFP-128Pin)

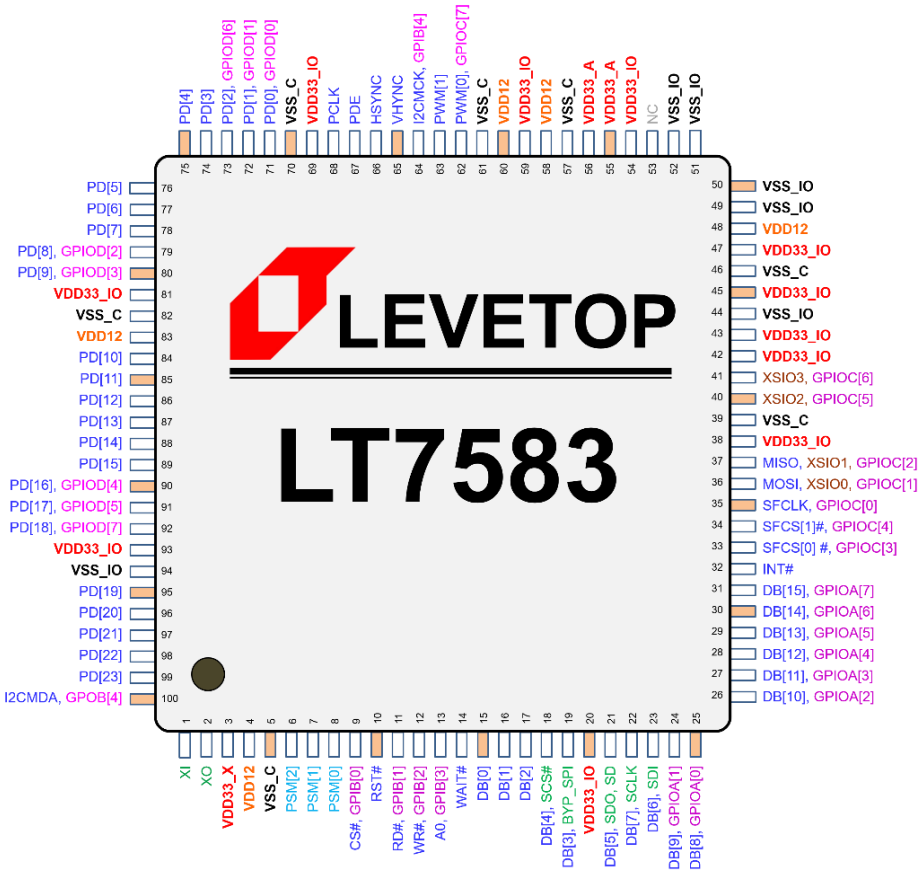


Figure 1-5: LT7583 Pin Assignment (LQFP-100Pin)

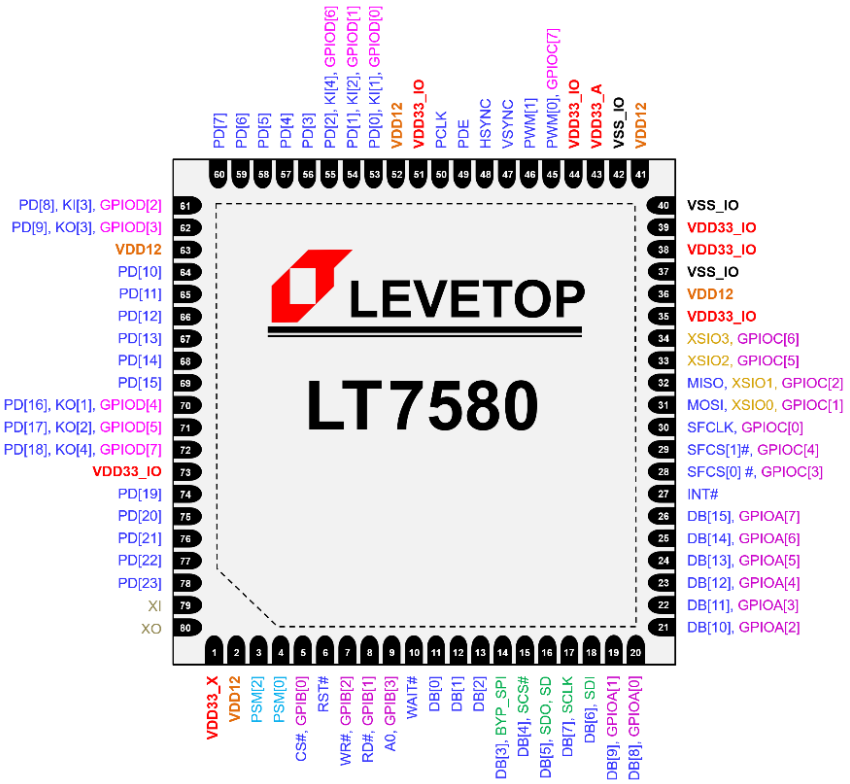


Figure 1-6: LT7580 Pin Assignment (QFN-80Pin)

2. Pin Description

2.1 MCU Interface

Table 2-1: MCU I/F Selection

<table>
<tr><th colspan="3">Pin Number</th><th rowspan="2">Pin Name</th><th rowspan="2">I/O</th><th rowspan="2">Description</th></tr>
<tr><th>LT7586</th><th>LT7583</th><th>LT7580</th></tr>
<tr><td>9
10
11</td><td>6
7
8</td><td>3
--
4</td><td>PSM[2]
PSM[1]
PSM[0]</td><td>I</td><td>MCU I/F Selection
<table>
<tr><th>PSM[2:0]</th><th>MCU I/F Mode</th></tr>
<tr><td>0 0 X</td><td>8/16bits, 8080 parallel interface mode</td></tr>
<tr><td>0 1 X</td><td>8/16bits, 6800 parallel interface mode</td></tr>
<tr><td>1 0 0</td><td>3-Wire SPI mode</td></tr>
<tr><td>1 0 1</td><td>4-Wire SPI mode (support By-Pass mode)</td></tr>
<tr><td>1 1 0</td><td>I2C mode (w/o the Continuous Read function)</td></tr>
<tr><td>1 1 1</td><td>I2C mode (with the Continuous Read function)</td></tr>
</table>
If the MCU interface is set to the parallel mode, then PSM[0] is used as an external interrupt input pin.
For LT7580, its PSM[1] pin is grounded internally, and it only supports 8/16bits parallel mode, 3-Wire SPI, and 4-Wire SPI mode.</td></tr>
</table>

2.2 MCU Parallel I/F Signals

Table 2-2: MCU Parallel I/F Signals

<table>
<tr><th colspan="3">Pin Number</th><th rowspan="2">Pin Name</th><th rowspan="2">I/O</th><th rowspan="2">Description</th></tr>
<tr><th>LT7586</th><th>LT7583</th><th>LT7580</th></tr>
<tr><td>35~25,
22~18</td><td>31~26,
24, 25,
22, 23,
21, 18,
19, 17~15</td><td>26~21,
19, 20,
17, 18,
16~11</td><td>DB[15:0]</td><td>IO</td><td>MCU Data Bus
These are the data bus for the data transfer between the MCU and LT758x.
DB[15:8] can be used as GPIO when the MCU interface is set to 8bit parallel mode.
DB[7:0] are multiplex pins that share with serial interface control pins. When the MCU interface is set to the serial mode, then DB[7:0] are defined as the control pins of the serial interface. Please refer to Table 2-1 for more information.</td></tr>
</table>

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
13	9	5	CS# GPIB[0]	I PU	**Chip Select Input** When CS# = 0, it means MCU can access LT758x for Command or Data transmission. If the MCU interface is set to the serial mode, then this pin can be set as GPIB[0]. This pin has an pull-high resistor.
14	11	8	RD# EN GPIB[1]	I PU	**Read / Enable Input** **RD#:** When the MCU interface is set to 8080 mode, this pin is a Read input signal, active low. **EN:** When the MCU interface is set to 6800 mode,this pin is an Enable input signal, active high. If the MCU interface is set to serial mode, then this pin can be set as GPIB[1]. This pin has an pull-high resistor.
15	12	7	WR# RW# GPIB[2]	I PU	**Write / Read-Write Input?** **WR#:** When the MCU interface is set to 8080 mode, this pin is a Write input signal, active low. **RW#:** When the MCU interface is set to 6800 mode, this pin is a Read-Write input signal, where RW# = 1 → MCU read cycle RW# = 0 → MCU write cycle If the MCU interface is set to the serial mode, then this pin can be set as GPIB[2]. This pin has an internal pull-high resistor.
16	13	9	A0 GPIB[3]	I	**Command / Data Select Input** A0 = 0, Status Read or Command Write cycle A0 = 1, Data Read or Data Write cycle If the MCU interface is set to the serial mode, then this pin can be set as GPIB[3]. This pin has an internal pull-high resistor.
36	32	27	INT#	O	**Interrupt Output Signal** When the preset interrupt condition is satisfied, this pin will become low for indicating the status to the MCU.
17	14	10	WAIT#	O	**Wait Output Signal** WAIT# = 0, LT758x is busy. WAIT# = 1, LT758x is ready to transfer data. The WAIT# signal is only valid for parallel host interface.

2.3 MCU Serial I/F Signals

Table 2-3: MCU Serial I/F Signals

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
27	22	17	SCLK (DB[7])	I	**SPI or I2C Clock** This pin is used as the clock signal when the MCU interface is set to 3-Wire, 4-Wire, or I2C This is multiplex pin that share with parallel data bus DB[7].
26	23	18	SDI I2C_SD (DB[6])	I	**I2C Data / 4-wire SPI Data Input** **SDI:** Data input pin of 4-wire SPI I/F. Connect to MCU's MOSI. **I2C_SDA:** Bi-direction data pin of I2C I/F Connect this pin to Gound if using 3-wire SPI I/F. This is a multiplex pin that shares with parallel data bus DB[6].
25	21	16	SDO SD I2CA[5] (DB[5])	IO	**4-wire SPI Data Output / 3-wire SPI Data / I2C Slave Address Select** **SDO:** Data output pin of 4-wire SPI I/F. Connect to MCU's MISO. **SD:** Bi-direction data pin of 3-wire SPI I/F. **I2CA[5]:** I2C device address bit[5] of I2C I/F. This is a multiplex pin that shares with parallel data bus DB[5].
22	18	15	SCS# I2CA[4] (DB[4])	I	**SPI Chip Select / I2C Slave Address Select** **SCS#:** Chip Select pin for 3-wire and 4-wire serial I/F. **I2CA[4]:** I2C device address bit[4]. This is a multiplex pin that shares with parallel data bus DB[4].
21	19	14	BYP_SPI I2CA[3] (DB[3])	I	**By-Pass Mode Control Signal / I2C Slave Address Select** **BYP_SPI:** When the MCU I/F uses 4-wire SPI, if this pin is set to high level, then the MCU can directly access the SPI Flash connected with LT758x. Set REG[B9h] bit5 to select the SPI Flash (SFCS[0]# or SFCS[1]#). I2CA[3]: I2C device address bit[3]. This is a multiplex pin that shares with parallel data bus DB[3]. Connect this pin to ground when using 3-wire SPI mode.

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
20~18	17~15	13~11	I2CA[2:0] (DB[2:0])	I	**I2C Slave Address Select** I2CA[2:0]: I2C device address bit[2:0] These are multiplex pins that share with parallel data bus DB[2:0]. Connect these pins to ground when using 3-wire SPI mode.

2.4 External Serial Flash / SPI Master Signals

Table 2-4: External Serial Flash / SPI Master Signals

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
37	33	28	SFCS[0]# GPIOC[3]	IO	**Chip Select 0 for External Serial Flash or SPI device** If SPI master I/F is disabled then this pin can be programmed as GPIOC[3], and its default function is input.
38	34	29	SFCS[1]# GPIOC[4]	IO	**Chip Select 1 for External Serial Flash or SPI device** If SPI master I/F is disabled then this pin can be programmed as GPIOC[4], and its default function is input.
39	35	30	SFCLK GPIOC[0]	IO	**External SPI Frequency Clock** Serial clock output for external Serial Flash or SPI device. If SPI master I/F is disabled then this pin can be programmed as GPIOC[0], and its default function is input.
40	36	31	MOSI XSIO0 GPIOC[1]	IO	**Master Output Slave Input (MOSI)** **Single Mode:** Data input for the serial Flash or SPI device. For LT758x, this is the data output pin. **Dual Mode:** The signal is used as bi-direction data #0 (XSIO0). Only valid for SPI Flash DMA mode. If SPI master I/F is disabled, then this pin can be programmed as GPIOC[1], and its default function is input.

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
41	37	32	MISO XSIO1 GPIOC[2]	IO	**Master Input Salve Output (MISO)** **Single Mode:** Data output for the serial Flash or SPI device. For LT758x, this is the data input pin. **Dual Mode:** The signal is used as bi-direction data #1 (XSIO1). Only valid for SPI Flash DMA mode. If SPI master I/F is disabled, then this pin can be programmed as GPIOC[2], and its default function is input.
47	40	33	XSIO2 GPIOC[5]	IO	**Quad SPI Mode，XSIO2 (#WP)** **Dual Mode:** The signal is used as bi-direction data #2 (XSIO2). Only valid for SPI Flash DMA mode. If SPI master I/F is disabled, then this pin can be programmed as GPIOC[5], and its default function is input.
48	41	34	XSIO3 GPIOC[6]	IO	**Quad SPI Mode，XSIO3 (#HOLD)** **Dual Mode:** The signal is used as bi-direction data #3 (XSIO3). Only valid for SPI Flash DMA mode. If SPI master I/F is disabled, then this pin can be programmed as GPIOC[6], and its default function is input.

Note: Due to the high communication speed, the SPI Flash related components and circuits should be placed as close to LT758x as possible.

2.5 PWM Signals

Table 2-5: PWM Signals

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
90	62	45	PWM[0] GPIOC[7] CCLK	IO	**PWM #0 Output Signal** This PWM output signal is programmable. It can be used to control TFT panel's backlight. The output mode can be set through configuration register. If the PWM function is disabled, then this pin can be programmed as GPIOC[7], and act as an input pin or output pin for core clock signal.
91	63	46	PWM[1]	IO	**PWM #1 Output Signal** This PWM output signal is programmable. It can be used to control TFT panel's backlight. The output mode can be set through configuration register.

2.6 LCD Interface Signals

Table 2-6: LCD I/F Signals

<table>
<tr><th colspan="3">Pin Number</th><th rowspan="2">Pin Name</th><th rowspan="2">I/O</th><th rowspan="2">Description</th></tr>
<tr><th>LT7586</th><th>LT7583</th><th>LT7580</th></tr>
<tr><td>127~123,
120~112,
108~99</td><td>99~95,
92~84,
80~71</td><td>78~74,
72~64,
62~53</td><td>PD[23:0]</td><td>IO</td><td>
LCD Panel Data Bus
These pins output RGB data to the TFT panle data bus. Users may set the register for corresponding RGB data bus.
<table>
<tr><th rowspan="2">Pin Name</th><th colspan="3">TFT-LCD RGB Interface</th></tr>
<tr><th>11b (GPIO)</th><th>10b (16bits)</th><th>00b (24bits)</th></tr>
<tr><td>PD[0]</td><td colspan="2">GPIOD[0]</td><td>B0</td></tr>
<tr><td>PD[1]</td><td colspan="2">GPIOD[1]</td><td>B1</td></tr>
<tr><td>PD[2]</td><td colspan="2">GPIOD[6]</td><td>B2</td></tr>
<tr><td>PD[3]</td><td>GPIOE[0]</td><td>B0</td><td>B3</td></tr>
<tr><td>PD[4]</td><td>GPIOE[1]</td><td>B1</td><td>B4</td></tr>
<tr><td>PD[5]</td><td>GPIOE[2]</td><td>B2</td><td>B5</td></tr>
<tr><td>PD[6]</td><td>GPIOE[3]</td><td>B3</td><td>B6</td></tr>
<tr><td>PD[7]</td><td>GPIOE[4]</td><td>B4</td><td>B7</td></tr>
<tr><td>PD[8]</td><td colspan="2">GPIOD[2]</td><td>G0</td></tr>
<tr><td>PD[9]</td><td colspan="2">GPIOD[3]</td><td>G1</td></tr>
<tr><td>PD[10]</td><td>GPIOE[5]</td><td>G0</td><td>G2</td></tr>
<tr><td>PD[11]</td><td>GPIOE[6]</td><td>G1</td><td>G3</td></tr>
<tr><td>PD[12]</td><td>GPIOE[7]</td><td>G2</td><td>G4</td></tr>
<tr><td>PD[13]</td><td>GPIOF[0]</td><td>G3</td><td>G5</td></tr>
<tr><td>PD[14]</td><td>GPIOF[1]</td><td>G4</td><td>G6</td></tr>
<tr><td>PD[15]</td><td>GPIOF[2]</td><td>G5</td><td>G7</td></tr>
<tr><td>PD[16]</td><td colspan="2">GPIOD[4]</td><td>R0</td></tr>
<tr><td>PD[17]</td><td colspan="2">GPIOD[5]</td><td>R1</td></tr>
<tr><td>PD[18]</td><td colspan="2">GPIOD[7]</td><td>R2</td></tr>
<tr><td>PD[19]</td><td>GPIOF[3]</td><td>R0</td><td>R3</td></tr>
<tr><td>PD[20]</td><td>GPIOF[4]</td><td>R1</td><td>R4</td></tr>
<tr><td>PD[21]</td><td>GPIOF[5]</td><td>R2</td><td>R5</td></tr>
<tr><td>PD[22]</td><td>GPIOF[6]</td><td>R3</td><td>R6</td></tr>
<tr><td>PD[23]</td><td>GPIOF[7]</td><td>R4</td><td>R7</td></tr>
</table>
Some pins of the LCD panel data bus share with GPIO pins. For example, if the data bus is set to 16bpp, then PD[18:16/9:8/2:0] are defined as GPIO pins.
</td></tr>
</table>

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
96	68	50	PCLK	O	**LCD Panel Scan Clock** Generic TFT interface signal for the LCD panel scan clock. This signal derives from the LT758x internal PLL.
93	65	47	VSYNC LVDS_PD#	O	**VSYNC Pulse** Generic TFT interface signal for the vertical synchronous pulse. When Ext_FlatInk(REG[00h] bit1 = 1, LT758x will enter DE mode, and the VSYNC signal becomes LVDS_PD#.
94	66	48	HSYNC LVDS_PD	O	**HSYNC Pulse** Generic TFT interface signal for the horizontal synchronous pulse. When Ext_FlatInk(REG[00h] bit1 = 1, LT758x will enter DE mode, and the HSYNC signal becomes LVDS_PD.
95	67	49	PDE	O	**Panel Data Enable** Generic TFT interface signal for data valid or data enable.

2.7 I2C Master

Table 2-7: I2C Master Signals

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
92	64	--	I2CMCK	I	**I2C Master Clock** This pin acts as I2C master clock signal.
128	100	--	I2CMDA	O	**I2C Master Data** This pin acts as I2C master data signal, which is set as Open-Drain output mode.

2.8 GPIO Signals

Table 2-8: GPIO Signals

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
35~28	31~26, 24, 25	26~19	GPIOA[7:0]	IO	**GPIO A Group** GPIOA[7:0] are general purpose I/O. These are multiplex pins that share with DB[15:8]. They are only available when the MCU interface is set to 8bits parallel mode or serial mode. The output mode of these pins can be configured through registers.
92, 128, 16~13	64, 1008, 13, 11, 12, 9	-- -- 9, 7, 8, 5	GPIB[4], GPOB[4], GPIB[3:0]	IO	**GPIO B Group** GPIB[4] shares the same pin with I2CMCK. GPOB[4] shares the same pin with I2CMCA. GPIB[3:0] are multiplex pins that share with {A0, WR#, RD#, CS#} GPIOB[3:0] are ready only under serial MCU mode.
90, 48, 47, 38, 37, 41, 40, 39	62, 41, 40, 34, 33, 37~35	45, 34, 33, 29, 28, 32, 31, 30	GPIOC[7], GPIOC[6:5], GPIOC[4:0]	IO	**GPIO C Group** GPIOC[7] shares the same pin with PWM[0], and is only valid when the PWM function is disabled.. GPIOC[6:0] share the same pins with {XSIO3, XSIO2, SFCS[1]#, SFCS[0]#, MISO, MOSI, SFCLK}, and are only valid when the SPI Master function is disabled.
120, 101 119, 118 108, 107 100, 99	92, 73, 91, 90, 80, 79, 72, 71	72, 55, 71, 70, 62, 61, 54, 53	GPIOD[7:0]	IO	**GPIO D Group** GPIOD[7:0] share the same pins with PD[18, 2, 17, 16, 9, 8, 1, 0], and are only valid when the LCD panel data bus is set to 16bits.

2.9 Reset and Test Signals

Table 2-9: Reset and Test Signals

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
12	10	6	RST#	I/O PU	**Reset Signal Input** When RST# = 0 and the reset signal lasts more than 32 OSC clocks, LT758x will be reset.
6~8	--	--	TEST[2:0]	I PD	**Test Mode Signals** These pins are for testing purposes, and should be grounded during regular operations. When TEST[0] = 1, the internal PLL will be disabled, and the OSC clock should be provided externally. When TEST[2:1] = b01, the SPI Master signals (connected to the Serial Flash) will be floating. This feature allows an external device to program the Serial Flash directly. (i.e. ISP, In-System-Programming) Users may also apply the "By-Pass Mode" to perform the ISP function. Refer to Section 13.2.1 for more details.

Note: PU: Pull-up; PD: Pull-Down; NP: No Pull

2.10 Power and Clock Signals

Table 2-10: Power and Clock Signals

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
1	1	79	XI	I	**Crystal / External Clock Input Signal** This pin is used to connected to an external oscillator clock input signal. The proper OSC range is between 5M and 30MHz, and 12MHz is recommended.
2	2	80	XO	O	**Crystal Clock Output** This pin is used to connected to an external oscillator clock output signal.
4, 111	4, 83	2, 63	VDD12	PWR	**Internal LDO 1.2V Output** Each pin must connect to a 10nF capacitor to ground.
51, 63, 79, 83	48, 58, 60	36, 41, 52	VDD12	PWR	**Internal LDO 1.2V Output** Each pin must connect to a 10nF and a 0.1uF capacitors to ground.
3	3	1	VDD33_X	PWR	**Internal OSC 3.3V Input** This pin must connect to a 0.1uF and a 2.2uF capacitors to ground. In addition, the power supply should be connected to this pin through a Bead.

Pin Number			Pin Name	I/O	Description
LT7586	LT7583	LT7580			
74, 75	55, 56	43	VDD33_A	PWR	**Internal LDO 3.3V Input** This pin must connect to a 0.1uF and a 2.2uF capacitors to ground. Also, the power supply must NOT connect with VDD33_IO, and should be connected to this pin through a Bead.
23, 42, 50, 62, 72, 81, 88, 97, 109, 121	20, 38, 42, 43, 45, 47, 54, 59, 69, 81, 93	35, 38, 39, 44, 51, 73	VDD33_IO	PWR	**3.3V I/O Input** Each pin must connect to a 0.1uF and a 2.2uF capacitors to ground.
5, 24, 43, 53, 76, 87, 89, 98, 110	5, 39, 46, 57, 61, 70, 82,	81(1)	VSS_C	PWR	**Ground pins for internal core power**
64, 65, 66, 78, 85, 122	44, 49, 50, 51, 52, 94	37, 40, 42	VSS_IO	PWR	**Ground pins for I/O power**

Note(1): This is also the Thermal Pad Zone for LT7580, which must be connected to VSS or GND. Special attention should be paid to the solder surface design of the pads when planning the PCB layouts. Refer to Section 18.4 for more information.

3. Electrical Characteristics

3.1 Absolute Maximum Ratings

Table 3-1: Absolute Maximum Ratings

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage Range	-0.3 ~ 4.0	V
V_{IN}	Input Voltage Range	-0.3 ~ VDD+0.3	V
V_{OUT}	Output Voltage Range	-0.3 ~ VDD+0.3	V
P_D	Power Dissipation	≦500	mW
T_{OPR}	Operation Temperature Range	-40 ~ 85	℃
T_{ST}	Storage Temperature	-45 ~ 125	℃
T_{SOL}	Soldering Temperature	260	℃

Note: If used beyond the absolute maximum ratings, LT758x may be permanently damaged. It is strongly suggested that the device is used within the electrical characteristics. If exposed to the condition not within the electrical characteristics, it may affect the reliability of the device. The specification does not guarantee the accuracy of the parameters without a given upper and lower limit value, but its typical value reasonably reflects the device performance.

3.2 Static Electrical Characteristics

Condition: VDD = 3.3V, TA = 25 ℃

Table 3-2: Electrical Characteristics

Symbol	Parameter	Condition	Min.	Typ.	Max.	Unit
V_{DD}	Operation Voltage		3.0	3.3	3.6	V
C_{VDD}	Loading Capacitor		1	-	10	uF
I_{OPR}	Operation Current	Note 1		50		mA
I_{STB}	Standby Current	Note 1		15		mA
I_{SUSP}	Suspend Current	Note 1		12		mA
I_{SLP}	Sleep Current	Note 1		5		mA
T_{RMP}	Power Ramp Up Time	VDD Ramp Up to 3.3 V	3.5		35	ms
OSC / PLL						
F_{OSC}	Oscillator Clock	V_{DD} = 3.3V	3.5	12	35	MHz
F_{VCO}	VCO Output Clock Frequency		200		400	MHz
T_{LOCK}	Lock Time	Note 2			50	us
CLK_{MPLL}	MPLL Output Clock (MCLK)	Note 1	25	133	180	MHz
CLK_{CPLL}	CPLL Output Clock (CCLK)	Note 1	25	133	170	MHz
CLK_{PPLL}	PPLL Output Clock (PCLK)	Note 1	12.5	30	100	MHz
Gm	The Oscillator Cell Gain.	HLMC HL55 DS[2:0]=b100	20.59	16.46	11.80	mA/V

Symbol	Parameter	Condition	Min.	Typ.	Max.	Unit
Serial Master Interface						
CLK_{SPI}	SPI Input Clock			40	50	MHz
Input / Output (CMOS 3-State Output pad with Schmitt Trigger Input, Pull-Up/Down)						
V_{IH}	Input High Voltage		2		3.6	V
V_{IL}	Input Low Voltage		-0.3		0.8	V
V_{OH}	Output High Vlotage		2.4			V
V_{OL}	Output Low Voltage				0.4	V
R_{PU}	Pull up Resistance		50	76	120	KΩ
R_{PD}	Pull down Resistance		40	63	110	KΩ
V_{TP}	Schmitt Trigger Low to High Threshold		1.5		2.1	V
V_{TN}	Schmitt Trigger High to Low Threshold		0.8		1.3	V
V_{HVS}	Hysteresis Voltage		200			mV
I_{LEAK}	Input Leakage Current		-10		+10	μA
V_{SLEW}	Rise/Fall Slew Rate			1.5		V/ns

Note 1: Testing condition: Based on 800x480 TFT panel, 8bit 8080 MCU I/F, and 16bpp. VDD= 3.3V, TA= 25℃, without extra load.

Note 2: Time needed from power-up till the internal PLL has stable clock outputs.

Table 3-3: Power Characteristics

Item	Symbol	Min	Typical	Max	Unit
Chip Voltage	VDD33_A	2.97	3.3	3.63	V
	VDD33_IO	2.97	3.3	3.63	V
Internal OSC Voltage	VDD33_X	2.97	3.3	3.63	V
LCD Controller Core Voltage (LDO O/P)	LCD_V12	1.1	1.2	1.3	V
RTC Voltage	VBAT	2.7	3.3	3.6	V

Table 3-4: Thermal Characteristics

Symbol	Parameter	Value	Unit
ReJC	Junction to Case	3 ~ 8	°C/W
ReJA	Junction to Ambient	20 ~ 25	°C/W

3.3 ESD Protection

Table 3-5: ESD Protection Specification

Test Item	Symbol	Max	Unit	Criteria
Human Body Model	HBM	4,000	V	ANSI/ESDA/JEDEC JS-001-2017
Machine Model	MM	200	V	JEDEC JESD22-A115C-2010
Charged Device Model	CDM	800	V	ANSI/ESDA/JEDEC JS-002-2022
Latch Up	LU	200	mA	JEDEC JESD78F.01-2022, @105°C

Note: While doing manual welding, it is suggested that anti-static measures should be well taken, including appropriate temperature and humidity environment, equipment grounding, anti-static workbench, and anti-static wrist straps for the technicians.

4. Clock and Reset

4.1 Clock

LT758x has embedded PLL circuits which need an external oscillator to provide the clock source, as shown in Figure 4-1. 12MHz oscillator is suggested for regular applications.

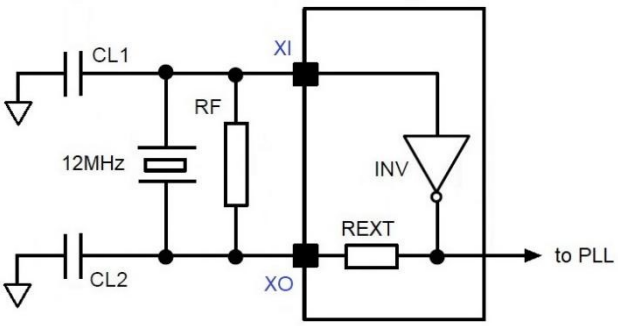


Figure 4-1：Oscillator Circuit

There are three sets of PLL circuits which provide three sets of clock signals:

- **CPLL ：** Provide **CCLK** (Core Clock) for MCU interface, BTE Engine, Graphic Engine, and Text DMA Engine etc. When using 12MHz oscillator, the default frequency will be 96MHz.
- **MPLL ：** Provide **MCLK** (Memory Clock) for internal Display Memory. The default frequency is 96MHz.
- **PPLL ：** Provide **PCLK** (Pixel Clock) for LCD panel scan frequency. The default frequency is 36MHz.

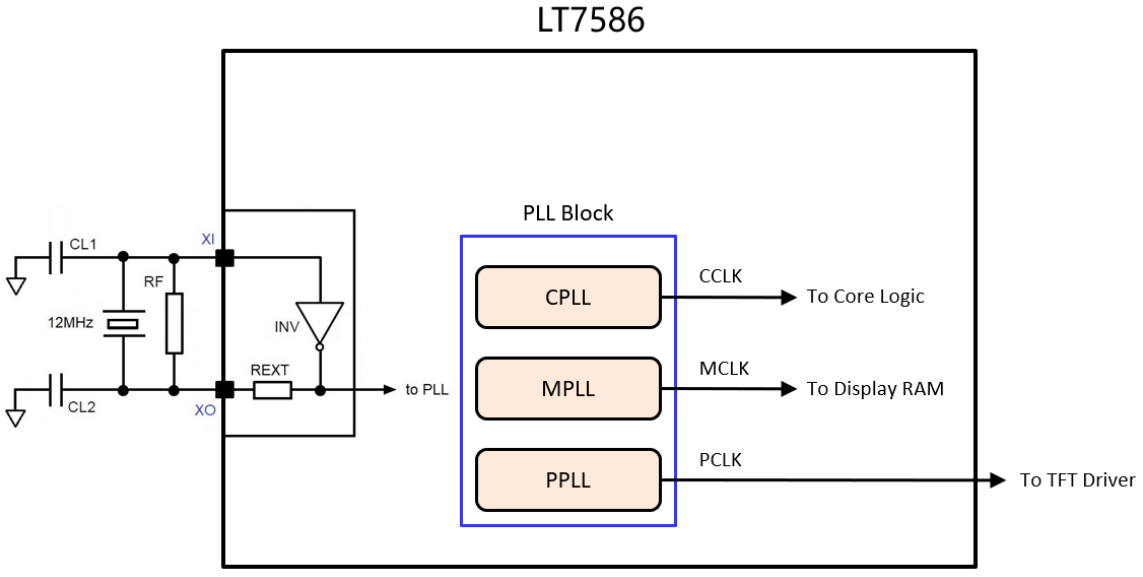


Figure 4-2：Three PLL Circuits

Each PLL output frequency (F_{OUT}) can be calculated from the following formula:

$$F_{OUT} = XI * (M \div N) \div OD$$

XI represents the external oscillator / clock input signal, the recommended value is 12MHz. **M** is the Feedback Divider Ratio of Loop. It includes 7bits and the value ranges from 4 to 127. **N** is the Input Divider Ratio, ranging from 1 to 15. **OD** is the Output Divider Ratio, and can be set to 1, 2, 4, or 8.

Other than the above formula, the following conditions must be met when setting the PLL register:

a) $3.5MHz \le XI \div N \le 35MHz$

b) $200MHz \le F_{OUT} * OD \le 400MHz$

c) $M \ge 4$; $N \ge 1$;

Table 4-1: PLL Register - Feedback Divider Ratio (M)

M[6:0]	Feedback Divider Ratio (M)
000_0000	NA
000_0001	NA
000_0010	NA
000_0011	NA
000_0100	4
000_0101	5
000_0110	6
⋮	⋮
111_1101	125
111_1110	126
111_1111	127

Table 4-2: PLL Register - Input Divider Ratio (N)

N[3:0]	Input Divider Ratio (N)
0000	NA
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

N[3:0]	Input Divider Ratio (N)
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

Table 4-3: PLL Register - Ouput Divider Ratio (OD)

OD[1:0]	Ouput Divider Ratio (OD)
00	1
01	2
10	4
11	8

In general, TFT manufacturers will advise the Pixel Clock(PCLK) for the best display effect, based on their TFT characteristics. Therefore, users can setup the register accordingly, and then apply the above rules to setup CCLK and MCLK.

The PLL (Fout) should be set according to the LCD panel resolution. For example, if an LCD panel resolution is 640*480, and the recommended PCLK is 30MHz, then users may set MCLK = 60MHz, CCLK = 60MHz. The PLL output and the register settings will be as following:

PCLK = XI * (M ÷ N) ÷ OD
= 12MHz * (60 ÷ 3) ÷ 8
= 30MHz

REG[06h] [7:0] (M) = 3Ch,
REG[05h] [4:1] (N) = 0011b,
REG[05h] [7:6] (OD) = 11b,

MCLK = XI * (M ÷ N) ÷ OD
= 12MHz * (60 ÷ 3) ÷ 4
= 60MHz

REG[08h] [7:0] (M) = 3Ch,
REG[07h] [4:1] (N) = 0011b,
REG[07h] [7:6] (OD) = 10b,

CCLK = XI * (M ÷ N) ÷ OD
= 12MHz * (60 ÷ 3) ÷ 4
= 60MHz

REG[0Ah] [7:0] (M) = 3Ch,
REG[09h] [4:1] (N) = 0011b,
REG[09h] [7:6] (OD) = 10b,

If an LCD panel resolution is 800*480, and the recommended PCLK is 50MHz, then users may set MCLK = 100MHz, and CCLK = 100MHz. The PLL output and the register settings will be as following:

PCLK = XI * (M ÷ N) ÷ OD
= 12MHz * (50 ÷ 3) ÷ 4
= 50MHz

REG[06h] [7:0] (M) = 32h,
REG[05h] [4:1] (N) = 0011b,
REG[05h] [7:6] (OD) = 10b,

MCLK = XI * (M ÷ N) ÷ OD
= 12MHz * (100 ÷ 3) ÷ 4
= 100MHz

REG[08h] [7:0] (M) = 64h,
REG[07h] [4:1] (N) = 0011b,
REG[07h] [7:6] (OD) = 10b,

CCLK = XI * (M ÷ N) ÷ OD
= 12MHz * (100 ÷ 3) ÷ 4
= 100MHz

REG[0Ah] [7:0] (M) = 64h,
REG[09h] [4:1] (N) = 0011b,
REG[09h] [7:6] (OD) = 10b,

Table 4-4: PLL Register Setting Example

PLL Register	Resolution 640*480	Resolution 800*480
REG[05h], PPLLC1	1100_0110b	1000_0110b
REG[06h], PPLLC2	0011_1100b (3Ch)	0013_0010b (32h)
REG[07h], MPLLC1	1000_0110b	1000_0110b
REG[08h], MPLLC2	0011_1100b (3Ch)	0100_0011b (64h)
REG[09h], CPLLC1	1000_0110b	1000_0110b
REG[0Ah], CPLLC2	0011_1100b (3Ch)	0100_0011b (64h)

4.2 Reset

4.2.1 Power-on Reset

LT758x has an embedded POR (Power On Reset) circuit. POR can issue an active low signal to synchronize the whole system through the RST# pin. When the system power (3.3V) on, the internal reset will be activated for at least 32 OSC clocks until the internal power is stable.

4.2.2 External Reset

External reset signal RST# allows LT758x to synchronize with the external systems. The external reset signal must be stabilized for at least 32 OSC clocks to be asserted. The Host MCU should check the bit1 (working mode status indication bit) of the Status Register before setting up LT758x to ensure that LT758x is in “Normal Operation State”.

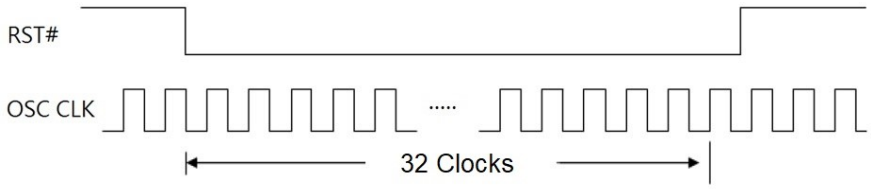


Figure 4-3：Reset Signal

4.2.3 Software Reset

If the Host MCU writes 1 to the register REG[00h] bit0, LT758x will be reset by software. The software reset will only reset the internal state machine of LT758x, and the other registers’ values will not be affected or cleared. After the software reset is complete, the REG[00h] bit0 will be cleared to 0 automatically.

5. Host Interface

LT758x is controlled by the external Host MCU. LT758x can be connected to a Host MCU through various parallel or seral interfaces which can be setup through PSM[2:0] pins, as listed in Table 5-1.

Table 5-1: Host Interface Mode

PSM[2:0]	Host Interface
0 0 X	8/16 bits 8080 Parallel Interface Mode
0 1 X	8/16 bits 6800 Parallel Interface Mode
1 0 0	3-Wire SPI Mode
1 0 1	4-Wire SPI Mode (Support By-Pass Mode)
1 1 0	I2C Mode (without Continuous Read)
1 1 1	I2C Mode (Support Continuous Read)

Since different MCU interfaces cannot be used at the same time, the unused interface pins can be set as GPIO pins. Please refer to the following table:

Table 5-2: The Pin Definition of the Host Interfaces

Pin Name	8080 I/F		6800 I/F		SPI 3-Wires	SPI 4-Wires	I2C
	8-bits	16-bits	8-bits	16-bits			
DB[15:8]	--	DB[15:0]	--	DB[15:0]	GPIOA[0:7]	GPIOA[0:7]	GPIOA[0:7]
DB[7]	DB[7:0]		DB[7:0]		SCLK	SCLK	SCLK
DB[6]					接地	SDI	I2C_SDA
DB[5]					SD	SDO	I2CA[5]
DB[4]					SCS#	SCS#	I2CA[4]
DB[3]					接地	**BYP_SPI**(1)	I2CA[3]
DB[2:0]					接地	接地	I2CA[2:0]
CS#	CS#		CS#		GPIB[0]	GPIB[0]	GPIB[0]
RD#	RD#		EN		GPIB[1]	GPIB[1]	GPIB[1]
WR#	WR#		RW#		GPIB[2]	GPIB[2]	GPIB[2]
A0	A0		A0		GPIB[3]	GPIB[3]	GPIB[3]
INT#	INT#		INT#		INT#	INT#	INT#
WAIT#	WAIT#		WAIT#		--	--	--

Note(1): See Section 13.2.1 for more detail.

When using Paralle Interface Mode, users may select 8bit or 16bit interface by setting the register REG[01h] bit0. If bit0 = 0, then 8bits is selected. If bit0 = 1, then 16bits is selected.

The following table shows the interfaces supported by LT758x series.

Table 5-3：Host Interfaces supported by LT758x series

No.	MCU Interface Mode	LT7586 LT7583	LT7580
1	8bits 8080 Parallel Interface Mode	V	V
2	16bits 8080 Parallel Interface Mode	V	V
3	8bits 6800 Parallel Interface Mode	V	--
4	16bits 6800 Parallel Interface Mode	V	--
5	3-Wire SPI Mode	V	V
6	4-Wiere SPI Mode	V	V
7	I2C Mode	V	--

Note: The accesses to all the registers are 8bits long only, except for the MRWDP register (REG[04h]). If the registers are accessed through the 16bits paralle interface, only LSB (DB[7:0] will be valid (except for the MRWDP register). Please refer to Chapter 17 for more detail about the MRWDP regsiter.

5.1 Paralle Host Interface

The application circuit and the timing chart of 8080/6800 parallel intefaces are explained in this section.

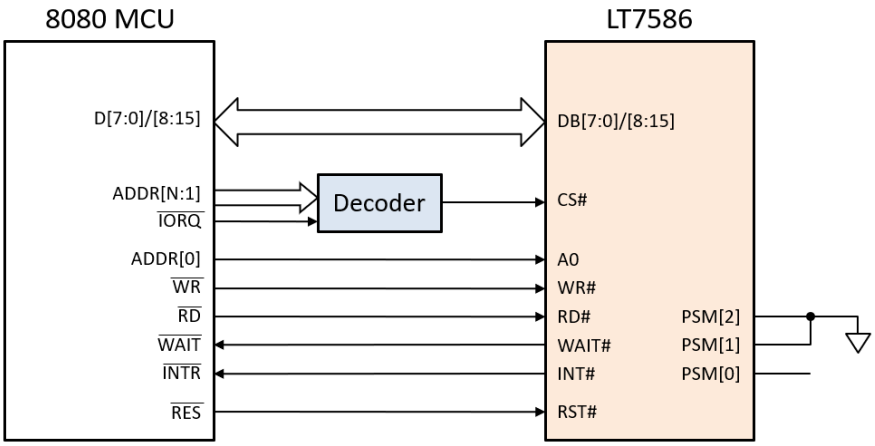


Figure 5-1: 8080 Parallel Host Interface

If WAIT# signal is not connected, then the time between each cycle has to be longer than 5 system clocks to avoid access fail. If the Host's reset signal is active low, then it can be connected to RST# pin of LT758x. The RST# pin can also be controlled by the Host's I/O pin; or be connected to an RC circuit that generate a low pluse when power on. No matter which of the above reset methods is applied, the reset signal has to last at least 32 OSC clocks. After reset, the Host should check bit1 of the Status Register to ensure that LT758x is in "Normal Operating State".

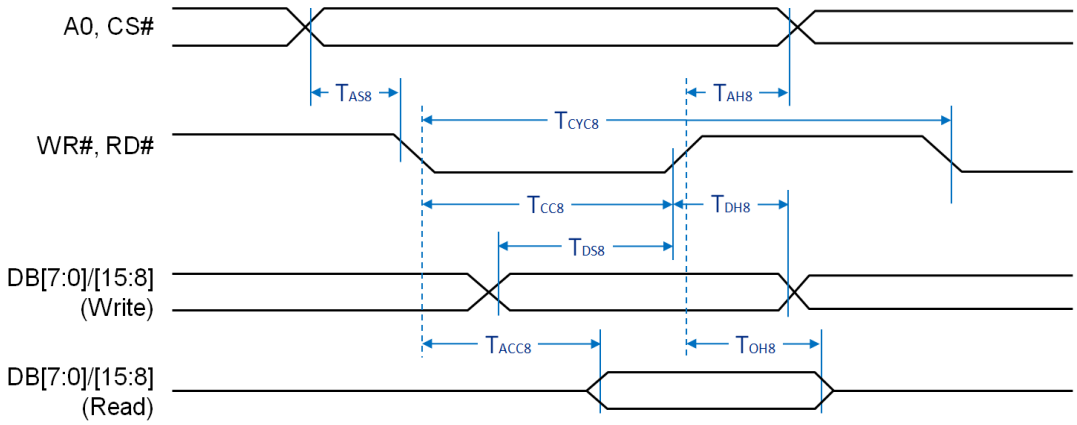


Figure 5-2: Timing Chart of 8080 Parallel Host Interface

Table 5-4: Timing Parameter for 8080 Parallel Host Interface

Symbol	Parameter	Rating		Unit	Note
		Min.	Max.		
T_{CYC8}	Cycle Time	50	--	ns	t_{CYC8} is one system clock (CCLK) period: t_{CYC8} = 1/CCLK
T_{CC8}	Strobe Pulse Width	20	--	ns	
T_{AS8}	Address Setup Time	1	--	ns	
T_{AH8}	Address Hold Time	1	--	ns	
T_{DS8}	Data Setup Time	20	--	ns	
T_{DH8}	Data Hold Time	10	--	ns	
T_{ACC8}	Data Output Access Time	0	20	ns	
T_{OH8}	Data Output Hold Time	0	20	ns	

Note 1: (Tcyc8 – Tcc8) >= 3* Tcclk

Note 2: In power saving mode, Tcclk = Tosc

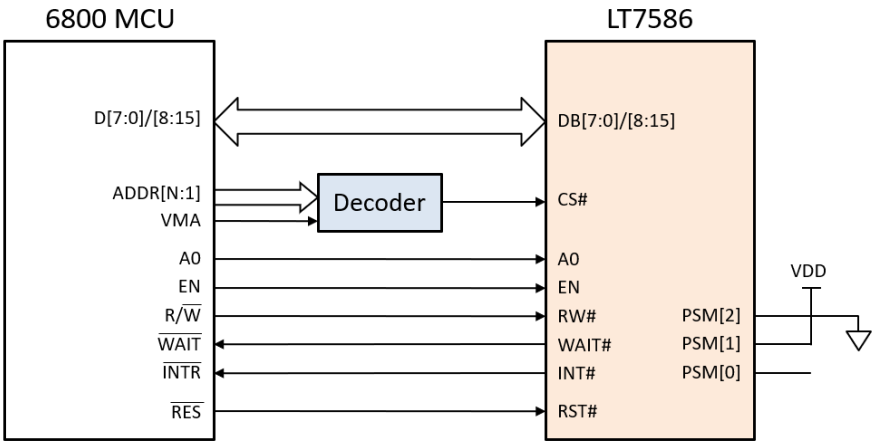


Figure 5-3: 6800 Parallel Host Interface

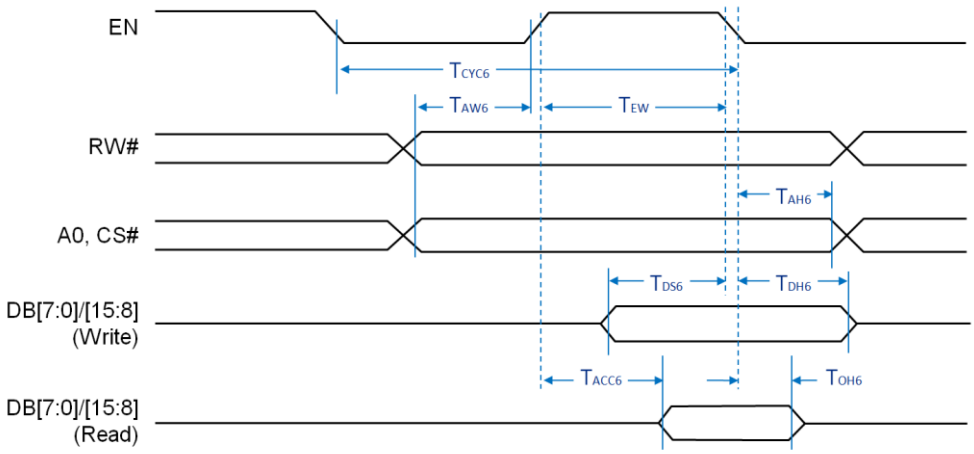


Figure 5-4: Timing Chart of 6800 Parallel Host Interface

Table 5-5：Timing Parameter for 6800 Parallel Host Interface

Symbol	Parameter	Rating		Unit	Note
		Min.	Max.		
T_{CYC6}	Cycle Time	50	--	ns	t_{CYC6} is one system clock (CCLK) period: t_{CYC6} = 1/CCLK
T_{EW}	Strobe Pulse Width	20	--	ns	
T_{AW6}	Address Setup Time	1	--	ns	
T_{AH6}	Address Hold Time	1	--	ns	
T_{DS6}	Data Setup Time	20	--	ns	
T_{DH6}	Data Hold Time	10	--	ns	
T_{ACC6}	Data Output Access Time	0	20	ns	
T_{OH6}	Data Output Hold Time	0	20	ns	

Host accesses LT758x registers and Display Memory through the host interface. LT758x has a Status Register and 256 Instruction Registers (i.e. REG[00h] ~ REG[FF]). The access procedure is as following:

- Register Write:
  1. Address Write：Write the Register's address. For example, 00h (i.e., REG[00h]), 01h (i.e., REG[01h]) etc.
  2. Data Write：Write data to the designated Register
- Register Read:
  1. Address Write：Write the Register's address.
  2. Data Read：Read the data from the designated register.

Display Memory (Display RAM) is where the TFT screen image data is stored. Host can write data into the Display RAM through Host interface. The procedure of accessing Display RAM is as following:

- Display RAM Write：
  1. Setting Active Window Registers (REG[56h]~REG[5Eh])
  2. Setting Graphic Coordinate registers （REG[5Fh] ~ REG[62h]）。
  3. Setting Memory Data Port Register REG[04h])
  4. Perform data writes to fill the designated window. Each write to the Memory Data Port will auto increment the internal memory address.

5.2 Serial Host Interface

5.2.1 3-Wire SPI Host Interface

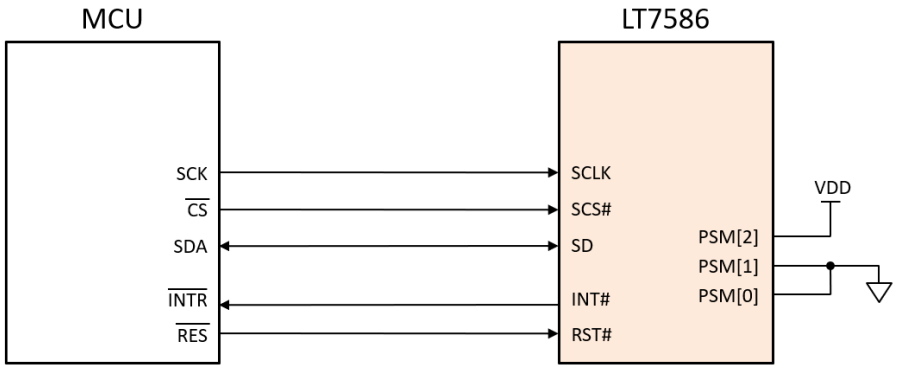


Figure 5-5: 3-Wire SPI Host Interface

Host can connect to LT758x through 3-Wire SPI interface. The SD signal is a bi-direction data pin for data access. The access timing and procedure are as below:

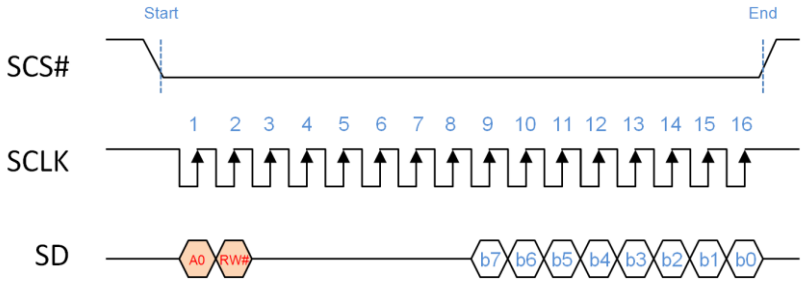


Figure 5-6: Timing Chart of 3-Wire SPI Host Interface

■ 3-Wire SPI Interface Protocol:

- Support Mode 3 only
- SCS# Low, CMDW(A0=0), REGx, DATRW(A0=1), Data0, Data1,…, SCS# High
- Continuous Read/Write is only available for memory access.
- SCS# Low, STSR(A0=0), Status, Status, Status …, SCS# High

■ Read Status Register:

1. Host drives SCS# (Low) and SCLK (SPI Clock)
2. Host drives A0 = 0, and then RW# = 1
3. LT758x will respond and send back the data of the Status Register (b7~ b0) at 9th ~ 16th Clock. The Host will then get the data of the the Status Register.

- Write Register's Address:
  1. Host drives SCS# (Low) and SCLK (SPI Clock)
  2. Host drives A0 = 0，and then RW# = 0
  3. Host drives the Register's address (b7 ~ b0) at 9th ~ 16th Clock to LT758x.

- Write Data to Register or Display Memory:
  1. Host drives SCS# (Low) and SCLK (SPI Clock)
  2. Host drives A0 = 1，and then RW# = 0
  3. Host drivers the data (b7 ~ b0) at 9th ~ 16th Clock to LT758x. The data will be written to the designated Register or display memory.
  4. When writing data to the display memory, if SCS# is kept Low, then the data can be written continuously without sending the instruction byte.

- Read Register's Data:
  1. Host drives SCS# (Low) and SCLK (SPI Clock)
  2. Host drives A0 = 1，and then RW# = 1。
  3. LT758x will drive the data (b7 ~ b0) of the designated Register at 9th ~ 16th Clock. Then the Host will get the content of the designated Register.
  4. When reading data from the display memory, if SCS# is kept Low, then the data can be read continuously without sending the instruction byte.

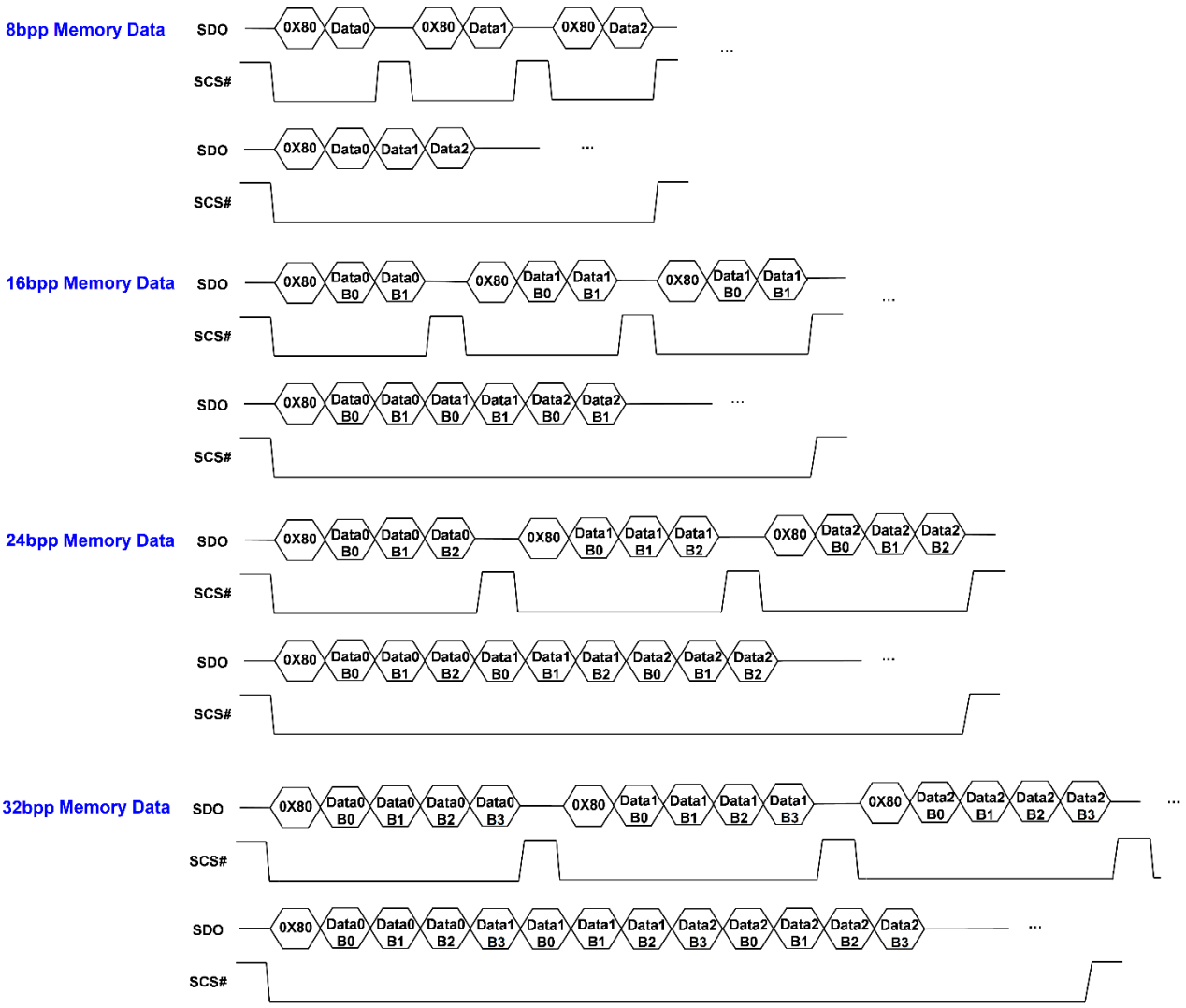


Figure 5-7: Data Formats of Writing to Display RAM through 3-Wire SPI Host Interface

5.2.2 4-Wire SPI Host Interface

The 4-Wire SPI is almost the same as 3-Wire. The difference is its input and output data lines are separated. The interface circuit diagram and timing charts are as following:

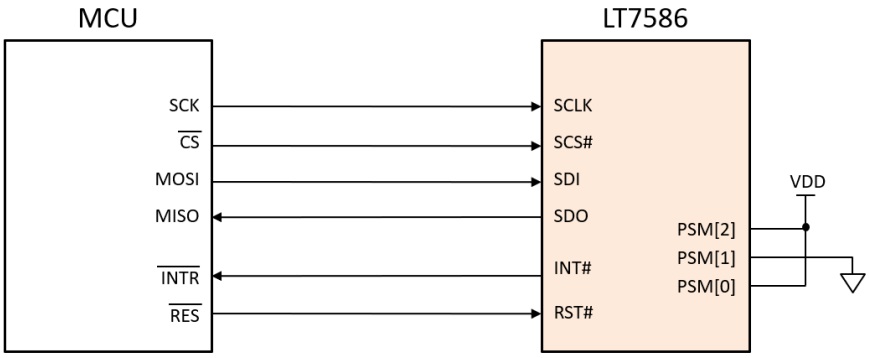


Figure 5-8: 4-Wire SPI Host Interface

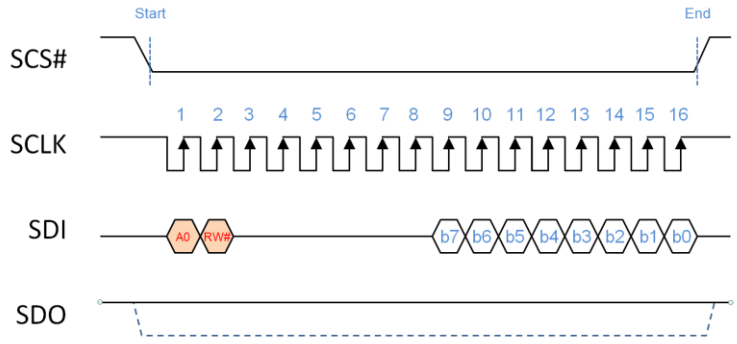


Figure 5-9: Write Timing Chart of 4-Wire SPI Host Interface

The above timing chart is the Write Cycle of 4-Wire SPI. When Host drives A0 = 0, and then RW# = 0, it means Host writes Register's Address. When Host drives A0 = 1, and then RW# = 0, it means Host writes data to Register or Display RAM.

The below timing chart shows the Read Cycle of 4-Wire SPI. When Host drives A0 = 0, and then RW# = 1, it means Host wants to read the data of the Status Register. LT758x will drive the data of the Status Register (b7 ~ b0) at 9^{th} ~ 16^{th} Clock. Then Host can get the data of the Status Register. When Host drives A0 = 1, and then RW# = 1, it means Host wants to read the data of the Command Register. LT758x will drive the data (b7 ~ b0) of the Command Register at 9^{th} ~ 16^{th} Clock. Host can then get the data of the command Register.

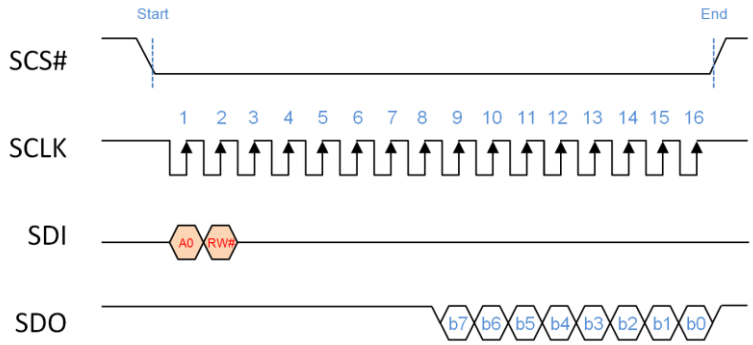


Figure 5-10: Read Timing Chart of 4-Wire SPI Host Interface

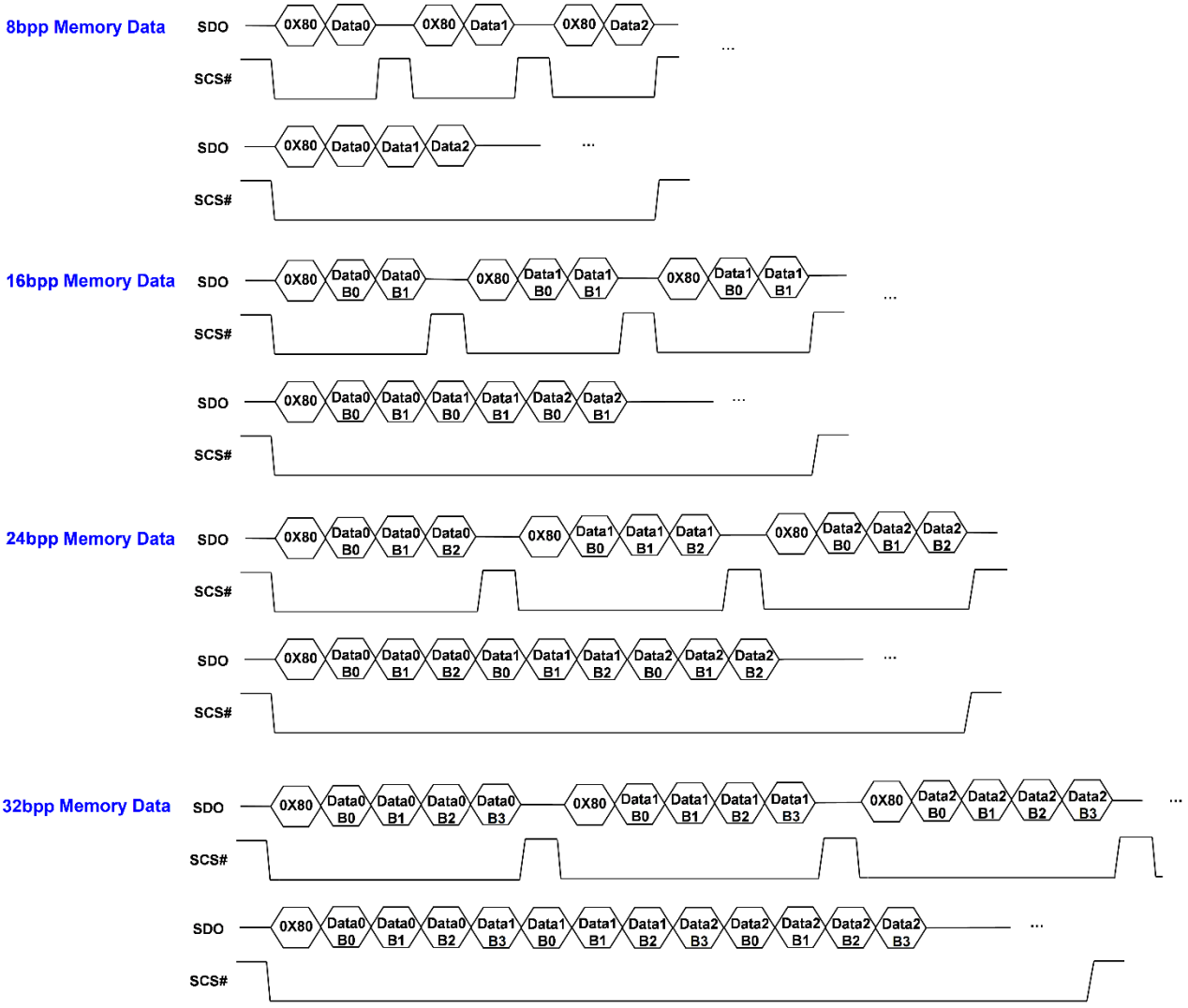


Figure 5-11: Data Formats of Writing to Display RAM through 4-Wire SPI Host Interface

5.2.3 I2C Host Interface

The I2C Host Interface is similar to 3-Wire SPI interface, yet I2C interface only needs 2 wires (SCK and SDA) for data transfer. The following is the application circuit of I2C interface. Singals I2CA[5:0] are used to setup LT758x's Device ID, and to avoid confusing with other I2C devices. In this example circuit, I2CA[5:3] are connected to VDD, and if all DIP Switches are at "ON" state, then the I2C Device ID is 111000b = 38h.

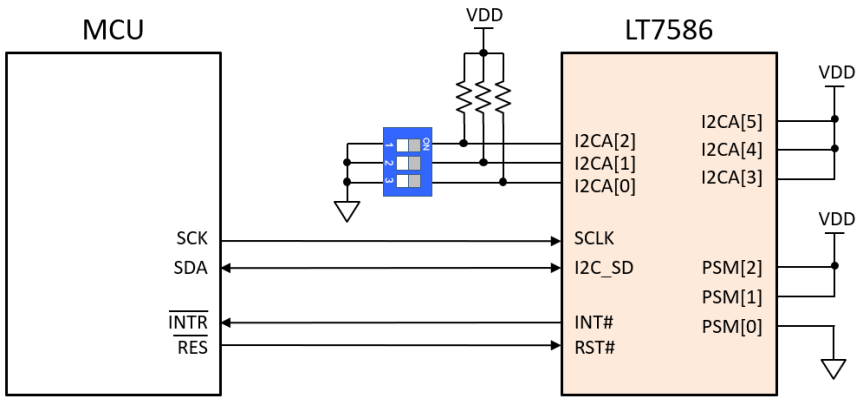


Figure 5-12: I2C Host Interface (without continuous read mode)

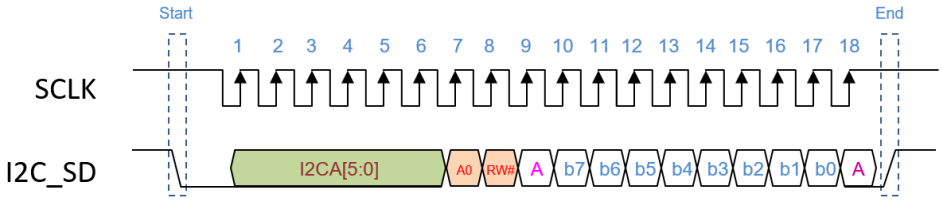


Figure 5-13: Timing Chart of I2C Host Interface

Figure 5-13 is the timing chart of I2C Host interface. Host has to find out the Device ID of the I2C device first, and then drives the Device ID in the first five clocks. Take Figure 5-12 as an example, the Device ID is 111000b. The definition of A0 and RW# is similar to SPI Host interface mentioned above. When Host drives A0 = 1 and RW# = 0, it means Host will write data (b7 ~ b0 at 10^{th} ~ 17^{th} Clock) to the Command Register or Display RAM. If Host drives A0 = 1 and RW# = 1, it means Host wants to read the data of the Command Register. LT758x will then drive the data (b7~b0) of the Command Register at 10^{th} ~ 17^{th} Clock. Host will then get the data of the Command Register. If Host drives A0 = 0 and RW# = 1, it means Host wants to read the data of the Status Register. LT758x will then drive the data (b7 ~ b0) of the Status Register at 10^{th} ~ 17^{th} Clock. Host will then get the data of LT758x's Status Register.

I2C clock is not supposed to exceed 1/10 of external OSC frequency. The protocol is as following:

- Single Read Mode: PSM = 110b, as shown in Figure 5-12. Each command can retrieve a set of data only, no matter which response (ack/nack) is received.
  - Start, DATR, {RData, ack/nack}, DATR, {RData.ack/nack}, ...{RData, ack/nack}, CMD.../Stop
  - Start CMDx, Data, CMDx, Data, ..., CMDx, Data/Stop
  - Start, CMDW, {reg_no, ack}, DATR/DATW/STSR ..... CMD...../Stop
  - Start, CMDx, reg_no, DATW, WData, WData, ... WData, CMD...../stop

- Continuous Read Mode: PSM = 111b, as shown in Figure 5-14. When nACK (ACK = 1) is received, the Host stops reading. In other words, when ACK is received (ACK = 0), the Host can read the next data.
  - **Note:** If a read action is not ended, LT758x operation may be stuck.
  - Start, STSR, {Status, ack}, {Status, ack}, ... , {Status, nack}, New CMD/Stop
  - Start, DATR, {RData, ack}, {RData, ack}, {RData, ...ack}, {RData, nack}, Stop, New CMD

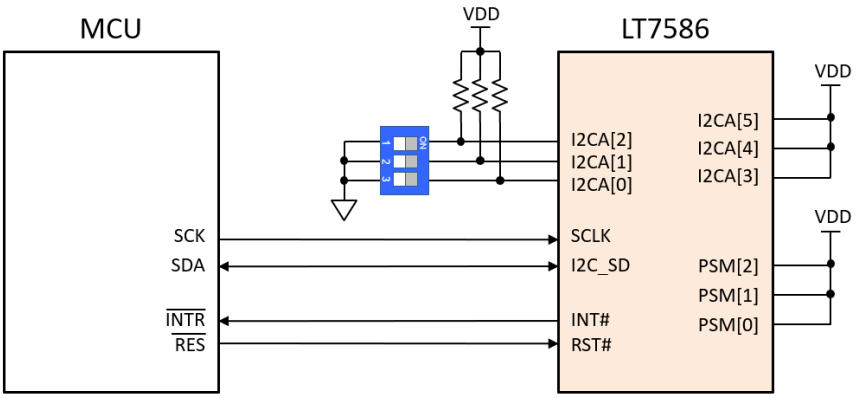


Figure 5-14: I2C Host Interface (Support continuous read mode)

5.3 Data Format of the Display Color

LT758x supports 256 color, 65K color, and 16.7M color. The data for RGB colors are arranged in Display RAM as following:

- 8bpp : RGB 3:3:2 (1 byte/pixel)
- 16bpp : RGB 5:6:5 (2bytes/pixel)
- 24bpp : RGB 8:8:8 (3bytes/pixel or 4bytes/pixel)
- 32bpp : αRGB 8:8:8:8 (4bytes/pixel, with opacity)

The following examples are based on 8bits MCU, 16bits MCU, and display color (RGB) in different data format.

5.3.1 Input Data without Opacity (RGB)

Table 5-6: 8bits MCU, 8bpp Mode (R:G:B=3:3:2)

Order	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	R_0^7	R_0^6	R_0^5	G_0^7	G_0^6	G_0^5	B_0^7	B_0^6
2	R_1^7	R_1^6	R_1^5	G_1^7	G_1^6	G_1^5	B_1^7	B_1^6
3	R_2^7	R_2^6	R_2^5	G_2^7	G_2^6	G_2^5	B_2^7	B_2^6
4	R_3^7	R_3^6	R_3^5	G_3^7	G_3^6	G_3^5	B_3^7	B_3^6
5	R_4^7	R_4^6	R_4^5	G_4^7	G_4^6	G_4^5	B_4^7	B_4^6
6	R_5^7	R_5^6	R_5^5	G_5^7	G_5^6	G_5^5	B_5^7	B_5^6

Table 5-7: 8bits MCU, 8bpp Gray Shade Mode (Gray256)

Order	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	Gray Color 0							
2	Gray Color 1							
3	Gray Color 2							
4	Gray Color 3							
5	Gray Color 4							
6	Gray Color 5							

Table 5-8: 8bits MCU, 8bpp Index Color Mode (Index-256)

Order	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	Index Color 0							
2	Index Color 1							
3	Index Color 2							
4	Index Color 3							
5	Index Color 4							
6	Index Color 5							

Table 5-9：8bits MCU，16bpp Mode（R:G:B=5:6:5）

Order	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	G_0^4	G_0^3	G_0^2	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3
2	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	G_0^7	G_0^6	G_0^5
3	G_1^4	G_1^3	G_1^2	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3
4	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	G_1^7	G_1^6	G_1^5
5	G_2^4	G_2^3	G_2^2	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3
6	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	G_2^7	G_2^6	G_2^5

Table 5-10：8bits MCU，24bpp Mode（R:G:B=8:8:8）

Order	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
2	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0
3	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
4	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0
5	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0
6	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0

Table 5-11：16bits MCU，8bpp Mode-1（R:G:B=3:3:2）

Order	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_0^7	R_0^6	R_0^5	G_0^7	G_0^6	G_0^5	B_0^7	B_0^6
2	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_1^7	R_1^6	R_1^5	G_1^7	G_1^6	G_1^5	B_1^7	B_1^6
3	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_2^7	R_2^6	R_2^5	G_2^7	G_2^6	G_2^5	B_2^7	B_2^6
4	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_3^7	R_3^6	R_3^5	G_3^7	G_3^6	G_3^5	B_3^7	B_3^6
5	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_4^7	R_4^6	R_4^5	G_4^7	G_4^6	G_4^5	B_4^7	B_4^6
6	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_5^7	R_5^6	R_5^5	G_5^7	G_5^6	G_5^5	B_5^7	B_5^6

Table 5-12：16bits MCU，16bpp Mode（R:G:B=5:6:5）

Order	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3
2	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3
3	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3
4	R_3^7	R_3^6	R_3^5	R_3^4	R_3^3	G_3^7	G_3^6	G_3^5	G_3^4	G_3^3	G_3^2	B_3^7	B_3^6	B_3^5	B_3^4	B_3^3
5	R_4^7	R_4^6	R_4^5	R_4^4	R_4^3	G_4^7	G_4^6	G_4^5	G_4^4	G_4^3	G_4^2	B_4^7	B_4^6	B_4^5	B_4^4	B_4^3
6	R_5^7	R_5^6	R_5^5	R_5^4	R_5^3	G_5^7	G_5^6	G_5^5	G_5^4	G_5^3	G_5^2	B_5^7	B_5^6	B_5^5	B_5^4	B_5^3

Table 5-13：16bits MCU，24bpp Mode-1 (R:G:B=8:8:8)

Order	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
2	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
3	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0
4	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	G_2^1	G_2^0	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3	B_2^2	B_2^1	B_2^0
5	B_3^7	B_3^6	B_3^5	B_3^4	B_3^3	B_3^2	B_3^1	B_3^0	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	R_2^2	R_2^1	R_2^0
6	R_3^7	R_3^6	R_3^5	R_3^4	R_3^3	R_3^2	R_3^1	R_3^0	G_3^7	G_3^6	G_3^5	G_3^4	G_3^3	G_3^2	G_3^1	G_3^0

Table 5-14：16bits MCU，24bpp Mode-2 (R:G:B=8:8:8)

Order	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
2	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
3	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0
4	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0
5	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	G_2^1	G_2^0	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3	B_2^2	B_2^1	B_2^0
6	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	R_2^2	R_2^1	R_2^0

5.3.2 Input Data with Opacity（α RGB）

LT758x provides a palette of 64 colors that can be chosen from the embedded 4096 different colors with opacity attribute. This palette can be applied on OSD (On Screen Display) function through BTE and index color. The α value represents a contrast value.

Table 5-15：8bits MCU，8bpp Mode (α:Index=2:6)

Order	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	α_1^3	α_1^2	Index Color Of Pixel 0					
2	α_3^3	α_3^2	Index Color Of Pixel 1					
3	α_5^3	α_5^2	Index Color Of Pixel 2					
4	α_7^3	α_7^2	Index Color Of Pixel 3					
5	α_9^3	α_9^2	Index Color Of Pixel 4					
6	α_{11}^3	α_{11}^2	Index Color Of Pixel 5					

α_x^3 α_x^2: 0→100%, 1→20/32, 2→11/32, 3→0

Table 5-16：8bits MCU，16bpp Mode（α:R:G:B=4:4:4:4）

Order	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	G_0^7	G_0^6	G_0^5	G_0^4	B_0^7	B_0^6	B_0^5	B_0^4
2	α_0^3	α_0^2	α_0^1	α_0^0	R_0^7	R_0^6	R_0^5	R_0^4
3	G_1^7	G_1^6	G_1^5	G_1^4	B_1^7	B_1^6	B_1^5	B_1^4
4	α_1^3	α_1^2	α_1^1	α_1^0	R_1^7	R_1^6	R_1^5	R_1^4
5	G_2^7	G_2^6	G_2^5	G_2^4	B_2^7	B_2^6	B_2^5	B_2^4
6	α_2^3	α_2^2	α_2^1	α_2^0	R_2^7	R_2^6	R_2^5	R_2^4

$\alpha_x^3\,\alpha_x^2\,\alpha_x^1\,\alpha_x^0$：0→100%，1→30/32，2→28/32，3→26/32，
4→24/32，……，12→8/32，13→6/32，14→4/32，15→0.

Table 5-17：8bits MCU，32bpp Mode（α:R:G:B=8:8:8:8）

Order	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
2	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0
3	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
4	α_0^7	α_0^6	α_0^5	α_0^4	α_0^3	α_0^2	α_0^1	α_0^0
5	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0
6	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0
7	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0
8	α_1^7	α_1^6	α_1^5	α_1^4	α_1^3	α_1^2	α_1^1	α_1^0

Table 5-18：16bits MCU，Index Mode（α:Index=2:6）

Order	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	α_0^3	α_0^2	Index color of pixel 0					
2	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	α_1^3	α_1^2	Index color of pixel 1					
3	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	α_2^3	α_2^2	Index color of pixel 2					
4	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	α_3^3	α_3^2	Index color of pixel 3					
5	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	α_4^3	α_4^2	Index color of pixel 4					
6	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	α_5^3	α_5^2	Index color of pixel 5					

$\alpha_x^3\,\alpha_x^2$：0→0，1→11/32，2→20/32，3→100%

Table 5-19：16bits MCU，12bpp Mode（α:R:G:B=4:4:4:4）

Order	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	α_0^3	α_0^2	α_0^1	α_0^0	R_0^7	R_0^6	R_0^5	R_0^4	G_0^7	G_0^6	G_0^5	G_0^4	B_0^7	B_0^6	B_0^5	B_0^4
2	α_1^3	α_1^2	α_1^1	α_1^0	R_1^7	R_1^6	R_1^5	R_1^4	G_1^7	G_1^6	G_1^5	G_1^4	B_1^7	B_1^6	B_1^5	B_1^4
3	α_2^3	α_2^2	α_2^1	α_2^0	R_2^7	R_2^6	R_2^5	R_2^4	G_2^7	G_2^6	G_2^5	G_2^4	B_2^7	B_2^6	B_2^5	B_2^4
4	α_3^3	α_3^2	α_3^1	α_3^0	R_3^7	R_3^6	R_3^5	R_3^4	G_3^7	G_3^6	G_3^5	G_3^4	B_3^7	B_3^6	B_3^5	B_3^4
5	α_4^3	α_4^2	α_4^1	α_4^0	R_4^7	R_4^6	R_4^5	R_4^4	G_4^7	G_4^6	G_4^5	G_4^4	B_4^7	B_4^6	B_4^5	B_4^4
6	α_5^3	α_5^2	α_5^1	α_5^0	R_5^7	R_5^6	R_5^5	R_5^4	G_5^7	G_5^6	G_5^5	G_5^4	B_5^7	B_5^6	B_5^5	B_5^4

$\alpha_x^3\,\alpha_x^2\,\alpha_x^1\,\alpha_x^0$：0→0，1→2/32，2→4/32，3→6/32，4→8/32，……，12→24/32，13→26/32，14→28/32，15→100%.

Table 5-20：16bits MCU，32bpp Mode（α:R:G:B=8:8:8:8）

Order	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
2	α_0^7	α_0^6	α_0^5	α_0^4	α_0^3	α_0^2	α_0^1	α_0^0	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
3	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0
4	α_1^7	α_1^6	α_1^5	α_1^4	α_1^3	α_1^2	α_1^1	α_1^0	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0
5	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	G_2^1	G_2^0	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3	B_2^2	B_2^1	B_2^0
6	α_2^7	α_2^6	α_2^5	α_2^4	α_2^3	α_2^2	α_2^1	α_2^0	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	R_2^2	R_2^1	R_2^0

6. Display Memory

LT758x has embedded 128Mb Display RAM. The Host can store the displayed data to LT758x Display RAM through designated instructions. LT758x will keep reading the image data stored in the Display RAM (main layer only), and sending them to the TFT driver for display. The amount of image layers is related to the Display RAM capacity and the picture resolution, as shown in the following table:

Table 6-1: LT758x Models vs. The amount of Image Layers

Model	Display RAM Capacity	Resolution (Max.)	Color (Max.)	Image Layer (@Max Color)
LT7586	128Mb	800*480	16.7M	14
		800*600	16.7M	11
		1024*600	16.7M	9
		1280*1024	16.7M	4
LT7583	128Mb	800*480	16.7M	14
		800*600	16.7M	11
		1024*600	16.7M	9
		1024*768	16.7M	7
LT7580	128Mb	800*480	16.7M	14
		800*600	16.7M	11
		1024*600	16.7M	9
		1024*768	16.7M	7

The embedded Display RAM of LT758x is a High-Speed SDRAM (Synchronous Dynamic Random Access Memory). Before the Display RAM can be accessed, the relevant register initialization must be done. Please refer to the initialization steps show below:

- Setup Register REG[E0h] according to the Display RAM capacity and model. The value of REG[E0h] must be set according to the LT758x model used to avoid display anomalies and image confusion. Please refer to Section 17.12 for more detail.
- Setup Register REG[E1h], REG[E2h], and REG[E3h] , including the CAS latency, refresh interval etc. The typical Display RAM refresh interval is 64ms. The value of these registers must be set according to the LT758x model used. Please refer to Section 17.12 for more detail.
- Set Register REG[E4h] bit0 to 1, and start Display RAM initialization process.
- Read Register REG[E4h] bit0, if it becomes 1, it means the initialization is done.

6.1 Display RAM Data Structure

The image data stored in the Display RAM will be stored in different arrangement formats depending on the color depths. Therefore, the Host must write the image data according to the formats, the following tables list various RGB data arrangement formats:

6.1.1 16bpp Display Data（RGB 5:6:5）

Table 6-2：16bpp Display Data（RGB 5:6:5)

Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0000h	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3
0002h	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3
0004h	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3
0006h	R_3^7	R_3^6	R_3^5	R_3^4	R_3^3	G_3^7	G_3^6	G_3^5	G_3^4	G_3^3	G_3^2	B_3^7	B_3^6	B_3^5	B_3^4	B_3^3
0008h	R_4^7	R_4^6	R_4^5	R_4^4	R_4^3	G_4^7	G_4^6	G_4^5	G_4^4	G_4^3	G_4^2	B_4^7	B_4^6	B_4^5	B_4^4	B_4^3
000Ah	R_5^7	R_5^6	R_5^5	R_5^4	R_5^3	G_5^7	G_5^6	G_5^5	G_5^4	G_5^3	G_5^2	B_5^7	B_5^6	B_5^5	B_5^4	B_5^3

6.1.2 24bpp Display Data（RGB 8:8:8）

Table 6-3：24bpp Display Data（RGB 8:8:8)

Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0000h	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
0002h	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
0004h	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0
0006h	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	G_2^1	G_2^0	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3	B_2^2	B_2^1	B_2^0
0008h	B_3^7	B_3^6	B_3^5	B_3^4	B_3^3	B_3^2	B_3^1	B_3^0	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	R_2^2	R_2^1	R_2^0
000Ah	R_3^7	R_3^6	R_3^5	R_3^4	R_3^3	R_3^2	R_3^1	R_3^0	G_3^7	G_3^6	G_3^5	G_3^4	G_3^3	G_3^2	G_3^1	G_3^0

6.1.3 Index Display Data with Opacity（α RGB 2：Index-64）

Table 6-4：Index Display Data with Opacity（αRGB 2：Index-64)

Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0000h	α_1^3	α_1^2	Index color of pixel 1						α_0^3	α_0^2	Index color of pixel 0					
0002h	α_3^3	α_3^2	Index color of pixel 3						α_2^3	α_2^2	Index color of pixel 2					
0004h	α_5^3	α_5^2	Index color of pixel 5						α_4^3	α_4^2	Index color of pixel 4					
0006h	α_7^3	α_7^2	Index color of pixel 7						α_6^3	α_6^2	Index color of pixel 6					
0008h	α_9^3	α_9^2	Index color of pixel 9						α_8^3	α_8^2	Index color of pixel 8					
000Ah	α_{11}^3	α_{11}^2	Index color of pixel 11						α_{10}^3	α_{10}^2	Index color of pixel 10					

$\alpha_x^3\alpha_x^2$: 0→0, 1→11/32, 2→20/32, 3→100%

6.1.4 12bpp Display Data with Opacity（αRGB 4:4:4:4）

Table 6-5：12bpp Display Data with Opacity（αRGB 4:4:4:4）

Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0000h	α_0^3	α_0^2	α_0^1	α_0^0	R_0^7	R_0^6	R_0^5	R_0^4	G_0^7	G_0^6	G_0^5	G_0^4	B_0^7	B_0^6	B_0^5	B_0^4
0002h	α_1^3	α_1^2	α_1^1	α_1^0	R_1^7	R_1^6	R_1^5	R_1^4	G_1^7	G_1^6	G_1^5	G_1^4	B_1^7	B_1^6	B_1^5	B_1^4
0004h	α_2^3	α_2^2	α_2^1	α_2^0	R_2^7	R_2^6	R_2^5	R_2^4	G_2^7	G_2^6	G_2^5	G_2^4	B_2^7	B_2^6	B_2^5	B_2^4
0006h	α_3^3	α_3^2	α_3^1	α_3^0	R_3^7	R_3^6	R_3^5	R_3^4	G_3^7	G_3^6	G_3^5	G_3^4	B_3^7	B_3^6	B_3^5	B_3^4
0008h	α_4^3	α_4^2	α_4^1	α_4^0	R_4^7	R_4^6	R_4^5	R_4^4	G_4^7	G_4^6	G_4^5	G_4^4	B_4^7	B_4^6	B_4^5	B_4^4
000Ah	α_5^3	α_5^2	α_5^1	α_5^0	R_5^7	R_5^6	R_5^5	R_5^4	G_5^7	G_5^6	G_5^5	G_5^4	B_5^7	B_5^6	B_5^5	B_5^4

$\alpha_x^3\ \alpha_x^2\ \alpha_x^1\ \alpha_x^0$：0→0，1→2/32，2→4/32，3→6/32，4→8/32，……，12→24/32，13→26/32，14→28/32，15→100%.

6.1.5 32bpp Display Data with Opacity（α RGB 8:8:8:8）

Table 6-6：32bpp Display Data with Opacity（αRGB 8:8:8:8）

Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0000h	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
0002h	α_0^7	α_0^6	α_0^5	α_0^4	α_0^3	α_0^2	α_0^1	α_0^0	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
0004h	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0
0006h	α_1^7	α_1^6	α_1^5	α_1^4	α_1^3	α_1^2	α_1^1	α_1^0	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0
0008h	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	G_2^1	G_2^0	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3	B_2^2	B_2^1	B_2^0
000Ah	α_2^7	α_2^6	α_2^5	α_2^4	α_2^3	α_2^2	α_2^1	α_2^0	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	R_2^2	R_2^1	R_2^0

6.2 Color Palette RAM

Table 6-7：Color Palette RAM（Index-4096）

Addr	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0000h	R_0^7	R_0^6	R_0^5	R_0^4	G_0^7	G_0^6	G_0^5	G_0^4	B_0^7	B_0^6	B_0^5	B_0^4
0002h	R_1^7	R_1^6	R_1^5	R_1^4	G_1^7	G_1^6	G_1^5	G_1^4	B_1^7	B_1^6	B_1^5	B_1^4
0004h	R_2^7	R_2^6	R_2^5	R_2^4	G_2^7	G_2^6	G_2^5	G_2^4	B_2^7	B_2^6	B_2^5	B_2^4
0006h	R_3^7	R_3^6	R_3^5	R_3^4	G_3^7	G_3^6	G_3^5	G_3^4	B_3^7	B_3^6	B_3^5	B_3^4
0008h	R_4^7	R_4^6	R_4^5	R_4^4	G_4^7	G_4^6	G_4^5	G_4^4	B_4^7	B_4^6	B_4^5	B_4^4
000Ah	R_5^7	R_5^6	R_5^5	R_5^4	G_5^7	G_5^6	G_5^5	G_5^4	B_5^7	B_5^6	B_5^5	B_5^4

7. LCD Interface

LT758x supports 16bits and 24bits RGB interface panels. No matter the color depth is 24bpp (RGB 8:8:8) or 16bpp (RGB 5:6:5), the display data can be sent to the TFT Driver on the TFT panel through these RGB interfaces. The LT758x LCD Display data that correspond to the RGB data are shown in the below Table 7-1. The Host can setup REG[01h] bit[4:3] to select 16bits or 24bits. Please refer to Table 7-2 for the RGB data supported by different models of LT758x.

Table 7-1: LT758x LCD Interface VS. RGB Display Data

LCD Data Bus	TFT-LCD RGB Interface	
	REG[01h] bit[4:3] = 10b (16bits)	REG[01h] bit[4:3] = 00b (24bits)
PD[0]		B0
PD[1]		B1
PD[2]		B2
PD[3]	B0	B3
PD[4]	B1	B4
PD[5]	B2	B5
PD[6]	B3	B6
PD[7]	B4	B7
PD[8]		G0
PD[9]		G1
PD[10]	G0	G2
PD[11]	G1	G3
PD[12]	G2	G4
PD[13]	G3	G5
PD[14]	G4	G6
PD[15]	G5	G7
PD[16]		R0
PD[17]		R1
PD[18]		R2
PD[19]	R0	R3
PD[20]	R1	R4
PD[21]	R2	R5
PD[22]	R3	R6
PD[23]	R4	R7

Table 7-2: RGB Data Types Supported by LT758x

Model	LCD Data Bus	RGB Data Types	Resolution	Colors
LT7586	PD[23~0]	R:G:B = 8:8:8	1280*1024	16.7M
	PD[23~19], PD[15~10], PD[7~3]	R:G:B = 5:6:5		65K
LT7583	PD[23~0]	R:G:B = 8:8:8	1024*768	16.7M
	PD[23~19], PD[15~10], PD[7~3]	R:G:B = 5:6:5		65K
LT7580	PD[23~0]	R:G:B = 8:8:8	1024*768	16.7M
	PD[23~19], PD[15~10], PD[7~3]	R:G:B = 5:6:5		65K

Figure 7-1 is the timing chart of the output signals from LT758x to the TFT-LCD. In addition to the RGB data lines mentioned above, LT758x also provides PCLK (Panel Scan Clock), VSYNC Pulse, HSYNC Pulse, and Data Enable signals. The PCLK frequency is setup by REG[05h] and REG[06h]. Please refer to Section 4.1 and Chapter 17 for more information.

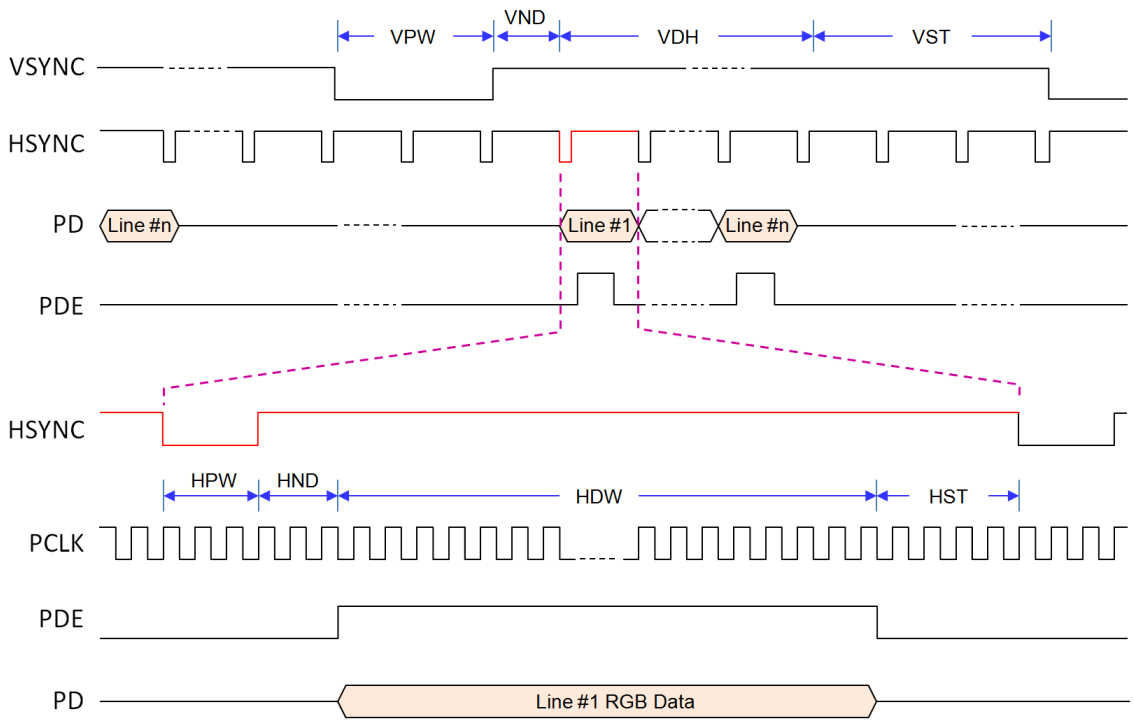


Figure 7-1: TFT-LCD RGB Interface Timing

8. Display Function

8.1 Color Bar

LT758x provides a color bar display function, which can be used as a display test and does not require display memory. The function can be performed by setting REG[12h] bit5 to 1.

Figure 8-1: Horizontal Color Bar

When REG[12h] bit4 (VDIR) = 0, the horizontal color bar will be displayed, as shown in Figure 8-1.
When REG[12h] bit4 (VDIR) = 1, the vertical color bar will be displayed, as shown in Figure 8-2.

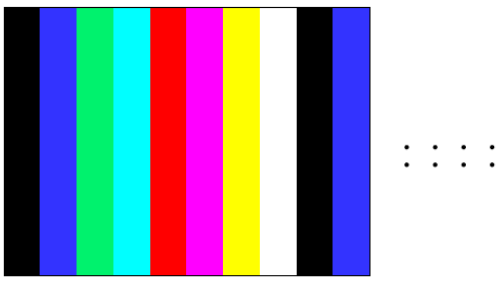


Figure 8-2: Vertical Color Bar

8.2 Main Window

The LCD main window size can be defined by setting Registers REG[14h] ~ REG[1Fh]. Users can store images to the memory buffers first. By settingthe related Registers (REG[20h] ~ REG[29h]), users can then display the images they want.

8.2.1 Configure Display Image Buffer

Display RAM is used to store images, and how many images can be sotred is determined by the image resolution and the color depth. For example, if the image resolution is 800*480 with 65K color (16bits), then 21 images can be sotred in 128Mbits Display RAM:

$$\text{Number of Images} = (128*1024*1024) / (800*480*16) = \mathbf{21}.8$$

If the image resolution is 1024*600 with 16.7M color (24bits), then 9 images can be stored in 128Mbits Display RAM:

$$\text{Number of Images} = (128*1024*1024) / (1024*600*24) = \mathbf{9}.1$$

Before writing the image data to the Display RAM, the starting position and the width of the Canvas window and the Active window range must be set. Please refer to the description of the register REG[50h] ~ REG[5Eh] in Chapter 17.

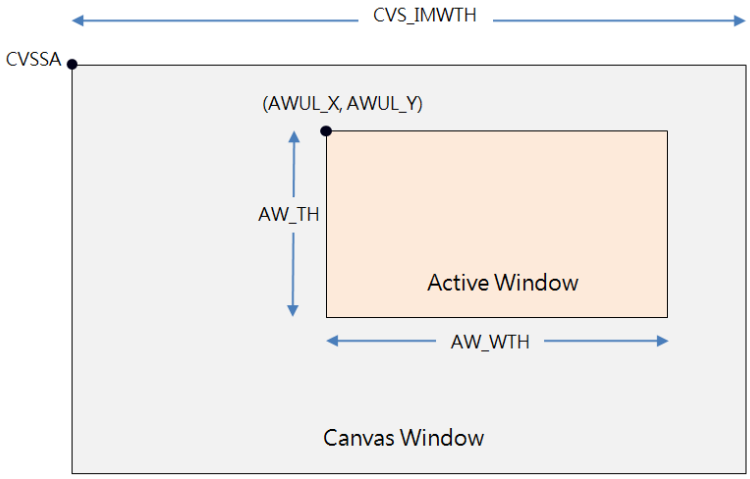


Figure 8-3：Canvas Window and Active Window

8.2.2 Write Image Data to Display RAM

Figure 8-4 shows the procedure of writing image data to the Display RAM and displaying the main window image onto the LCD screen.

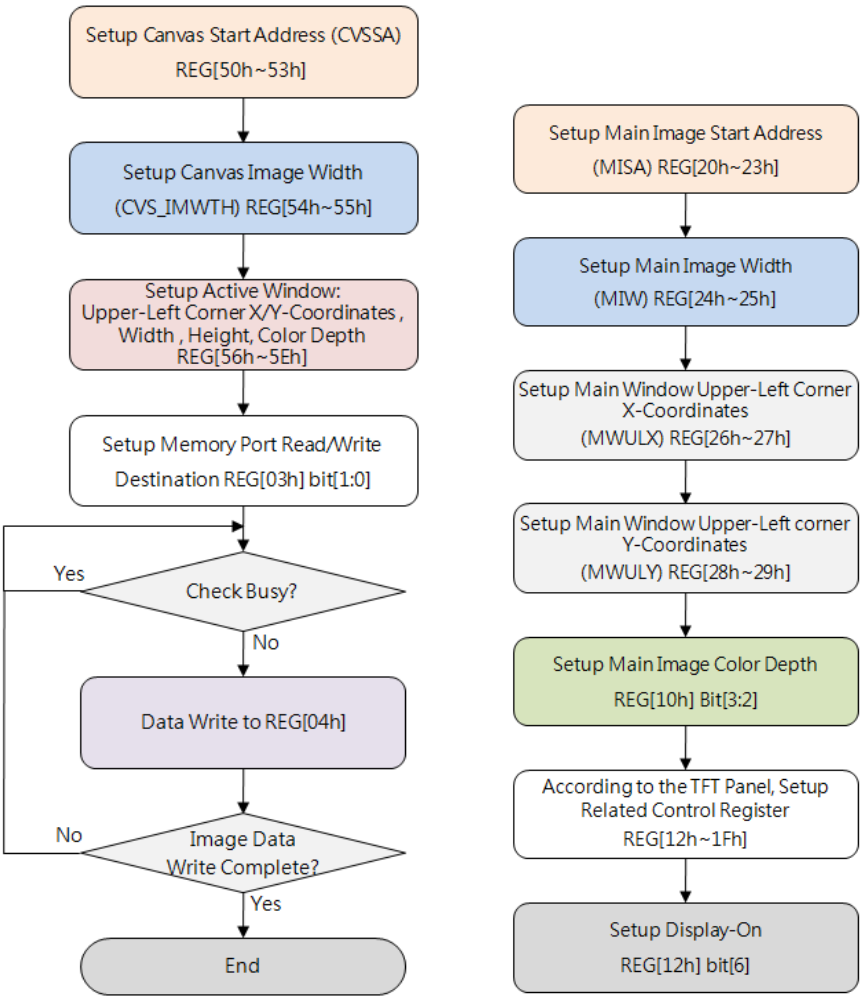


Figure 8-4：Write Image Data to Display RAM

8.2.3 Setup and Select the Main Window Image

Image data stored in the assigned Main Window area of the SDRAM will be displayed onto the TFT panel directly. The Main Window area can be assigned by setting the Registers REG[20h] ~ REG[29h]. Figure 8-5 shows the flow for setting up the main window.

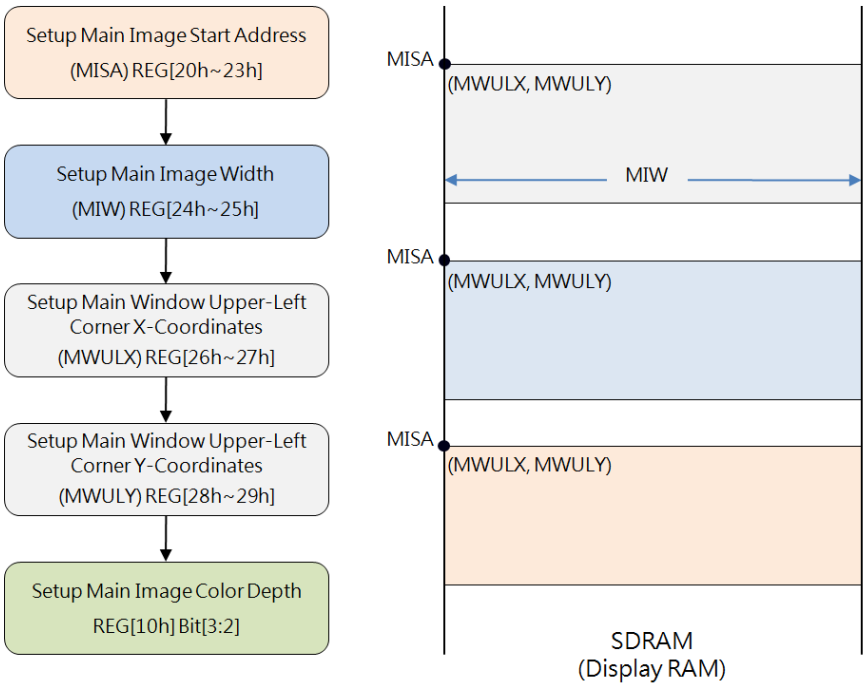


Figure 8-5：Setup and Select the Main Window Image

8.3 Picture-In-Picture (PIP)

LT758x supports 2 sets of Picture-in-Picture feature (PIP-1 & PIP-2). Users can display sub-image on the main window without overwriting the image data of the main window. If PIP-1 and PIP-2 are overlapping, then the PIP-1 image is always on the top of the PIP-2 image.

Note: The display priority is Graphic Cursor > Text Cursor > PIP1 > PIP2 > Main

The size and location of the PIP window are set by the Register REG[2Ah] ~ REG[3Bh] and REG[11h]. The parameters of PIP-1 and PIP-2 are using the same registers, whereas REG[10h] bit4 is used to decide whether the parameters are for PIP-1 or PIP-2. Before using the PIP function, the relevant parameters must be set. The unit of PIP windows sizes and start positions is 4 pixels in horizontal, and 1 pixel in vertical.

LT758x supports two PIP windows in the Main window. In the PIP window, overlapping of transparent display is aslo supported. Users may apply ROP functions in the PIP window to implement such effect.

8.3.1 PIP Window Setting

To apply the PIP feature, the Host has to setup the PIP Image Start Address, Image Width, Display X/Y coordinates, Image X/Y coordinates, PIP Window Color Depth, PIP Window Width, and PIP Window Height. Figure 8-6 shows the flow for setting and displaying the PIP window:

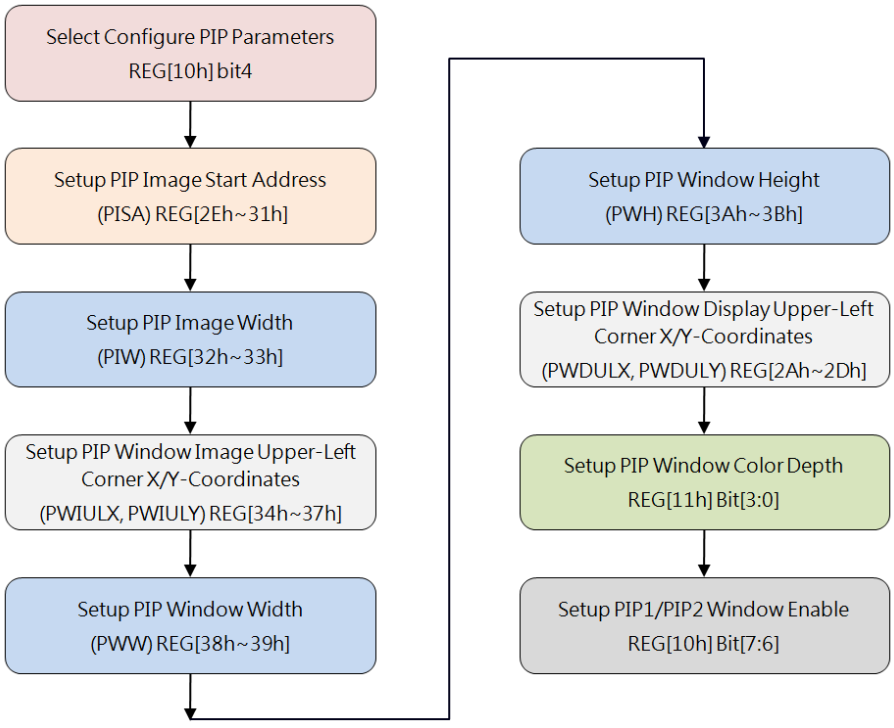


Figure 8-6: Setup PIP Window

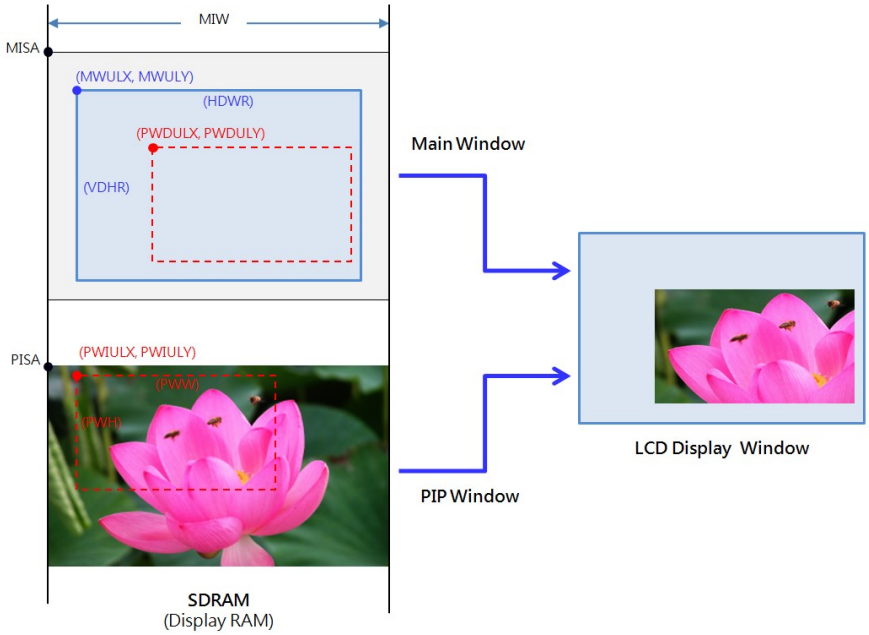


Figure 8-7：PIP Example

8.3.2 PIP Display Position and PIP Image Position

As described in the previous section, the final position of the PIP window on the LCD screen can be changed by setting PWDULX and PWDULY. In addition, setting PISA, PIW, PWIULX, and PWIULY can switch the PIP images that users want to display. These actions do not change any image data that existes in the Display RAM, but can easily switch the picture to be displayed. The following example shows a Main window with a PIP window in it. Users can display different PIP images by setting the PIP image Position Register (PWIULX and PWIULLY).

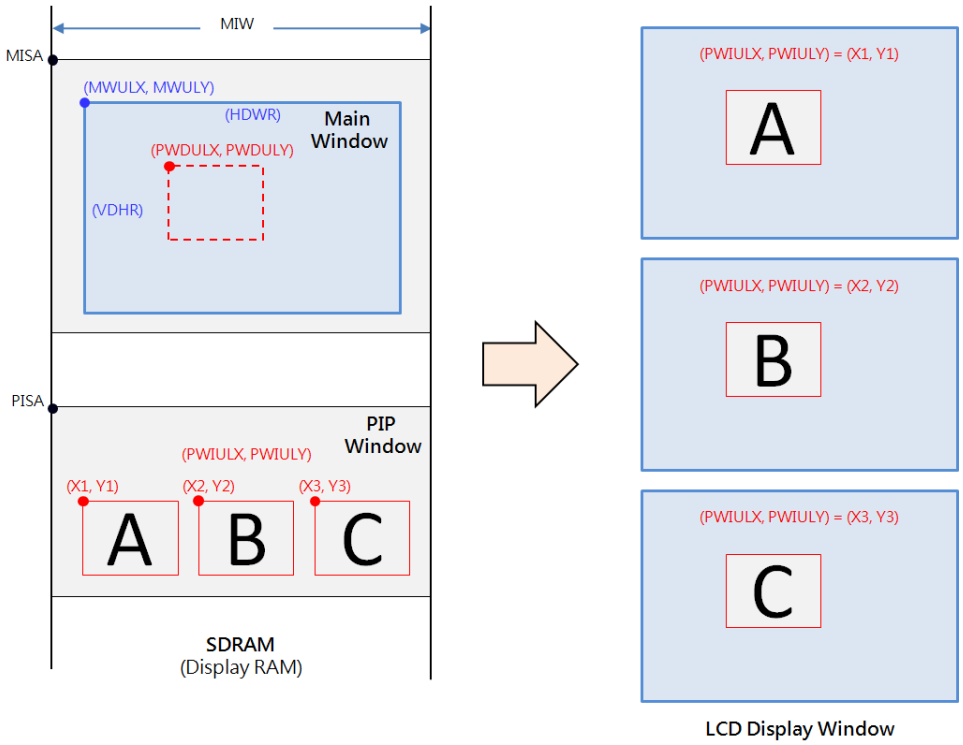


Figure 8-8：Select different PIP image to be displayed

8.4 Image Rotate and Mirror

Usuually LCD displays are refreshed horizontally (from left to right and then from top to bottom), and the displayed image data are stored the same way in the Display RAM. However, by setting related registers, the memory storing direction of the image data written by the Host can be changed, and therefore the written image can be displayed in a Rotate or Mirror manner.

REG[02h] bit[2:1] are used to control the Memory storing direction. These two bits are effective only for Graphic Mode.

00b: Left → Right, then Top → Bottom (Original)

01b: Right → Left, then Top → Bottom (Horizontal flip)

10b: Top → Bottom, then Left → Right (Rotate 90° to the right & Horizontal flip)

11b: Bottom → Top, then Left → Right (Rotate 90° to the left)

Followings are some examples for Image Rotate and Mirror:

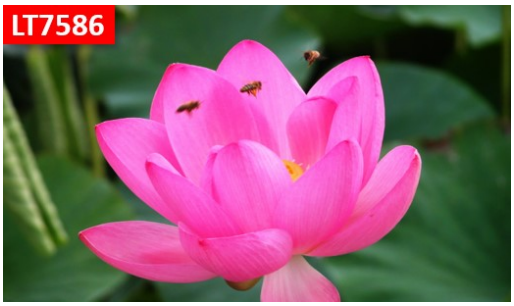


Figure 8-9: Original Image (REG[02h] bit[2:1]=00b)

1. When VDIR (REG[12h] bit3) = 0

When REG[02h] bit[2:1] = 00b, the written image data will be stored from left to right and then from top to bottom. The image will be displayed as shown in Firgure 8-9.

When REG[02h] bit[2:1] = 01b, the written image data will be stored from right to left and then from top to bottom. The image will be displayed as a horizontal mirror image, as shown in Figure 8-10.

Figure 8-10: Horizontal Mirror Image (REG[02h] bit[2:1]=01b)

When REG[02h] bit[2:1] = 10b, the written image data will be stored from top to bottom and then from left to right. The image will be displayed as shown in Figure 8-11.

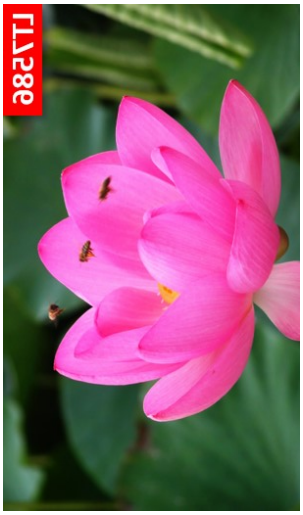


Figure 8-11：Rotated 90° to the right and Flipped horizontally（REG[02h] bit[2:1]=10b）

When REG[02h] bit[2:1] = 11b, the written image data will be stored from bottom to top and then from left to right. The image will be displayed as shown in Figure 8-12.

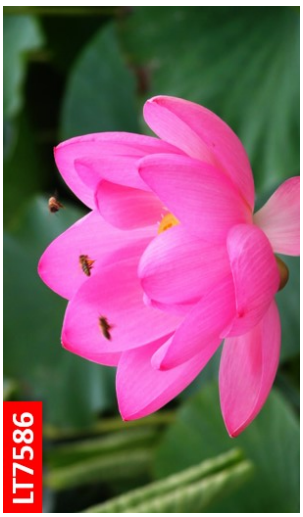


Figure 8-12：Rotate 90° to the left（REG[02h] bit[2:1]=11b）

2. When VDIR (REG[12h] bit3) = 1

Figure 8-13：Displayed Image （REG[02h] bit[2:1]=00b）

Figure 8-14：Displayed Image （REG[02h] bit[2:1]=01b）

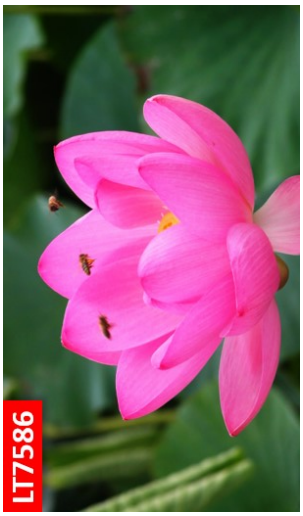


Figure 8-15：Displayed Image （REG[02h] bit[2:1]=10b）

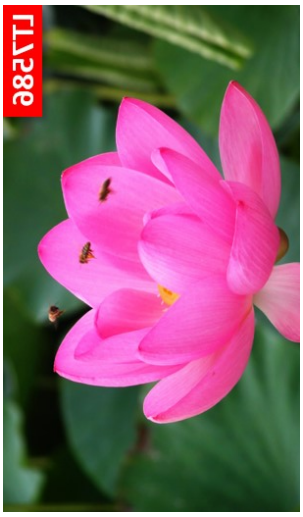


Figure 8-16：Displayed Image（REG[02h] bit[2:1]=11b）

9. Geometric Drawing Engine

9.1 Drawing Circle and Ellipse

LT758x supports the function of drawing Circles and Ellipses. As long as the Host sets the Circle or the Ellipse Center (REG[7Bh ~ 7Eh]), Radius (REG[77h ~ 7Ah]), and Color (REG[D2h ~ D4h]), then set REG[76h] bit[5:4] = 00b, and finally enables the Drawing Circle function by setting REG[76h] bit7 = 1, then LT758x will draw the specified Circle or Ellipse on the TFT panel automatically.

Host can also set REG[76h] bit6 to 1 to fill the Circle or Ellipse. There is no need for the Host to write image data for the fill operation. An Ellipse has two radius (R1 and R2). To draw a circle, simply set the long radius and short radius to the same value (R1 = R2).

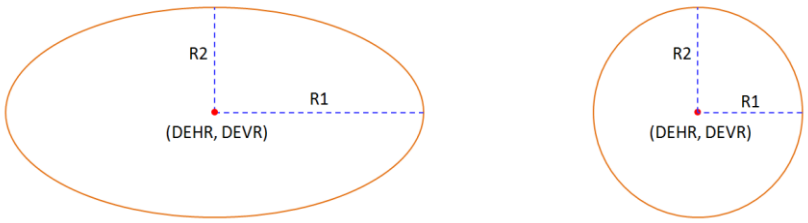


Figure 9-1：Drawing Circle and Ellipse

Figure 9-2 shows the flow of drawing a circle/ellipse. The left one is to draw a circle/ellipse withoutfill, and the right one is to draw a circle/ellipse with fill. Note that the center point of th circle must be located within the Active Window.

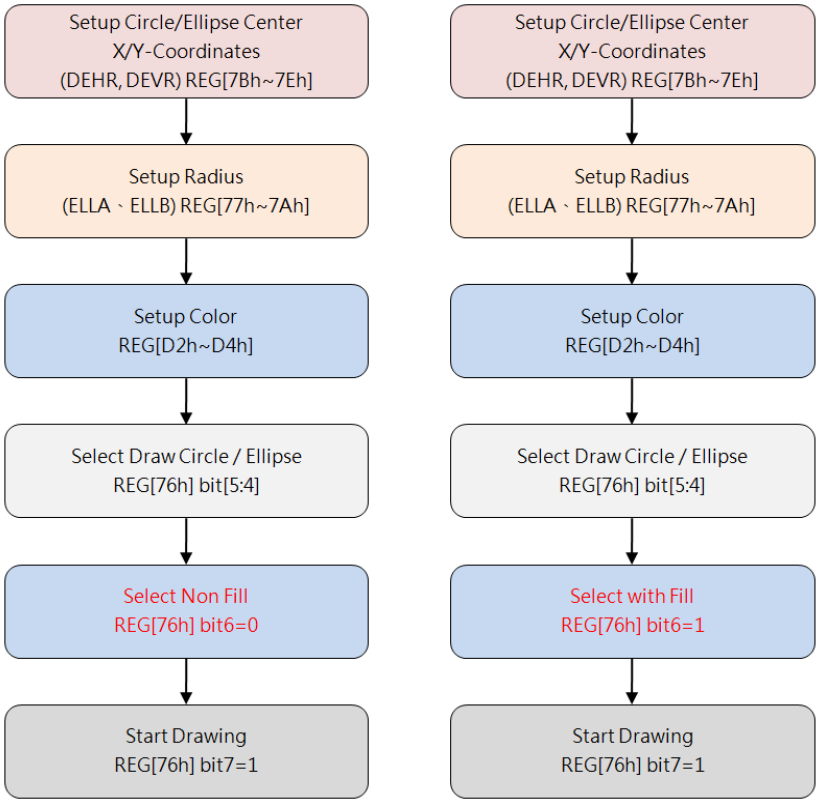


Figure 9-2：Flowchart of Draiwng Circle and Ellipse

9.2 Drawing Curve

LT758x supports the function of drawing Curves. As long as the Host sets the Curve Center (REG[7Bh ~ 7Eh], Radius (REG[77h ~ 7Ah]), and Color (REG[D2 ~ D4h]), then set REG[76h] bit[5:4] = 01b, and finally enables the Drawing Curve function by setting REG[76h] bit7 = 1, then LT758x will draw the specified Curve on the TFT panel automatically. The fill operation is also supported.

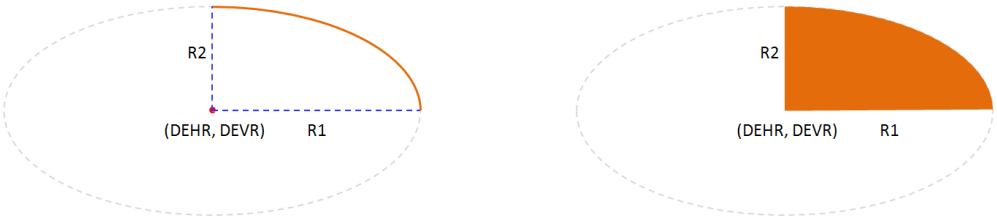


Figure 9-3: Drawing Curve and one-fourth Ellipse

Figure 9-4 shows the flow of drawing a curve/one-fourth ellipse. The left one is to draw a curve/one-fourth ellipse withoutfill, and the right one is to draw a curve/one-fourth ellipse with fill. Note that the center point of the Curve must be located within the Active Window.

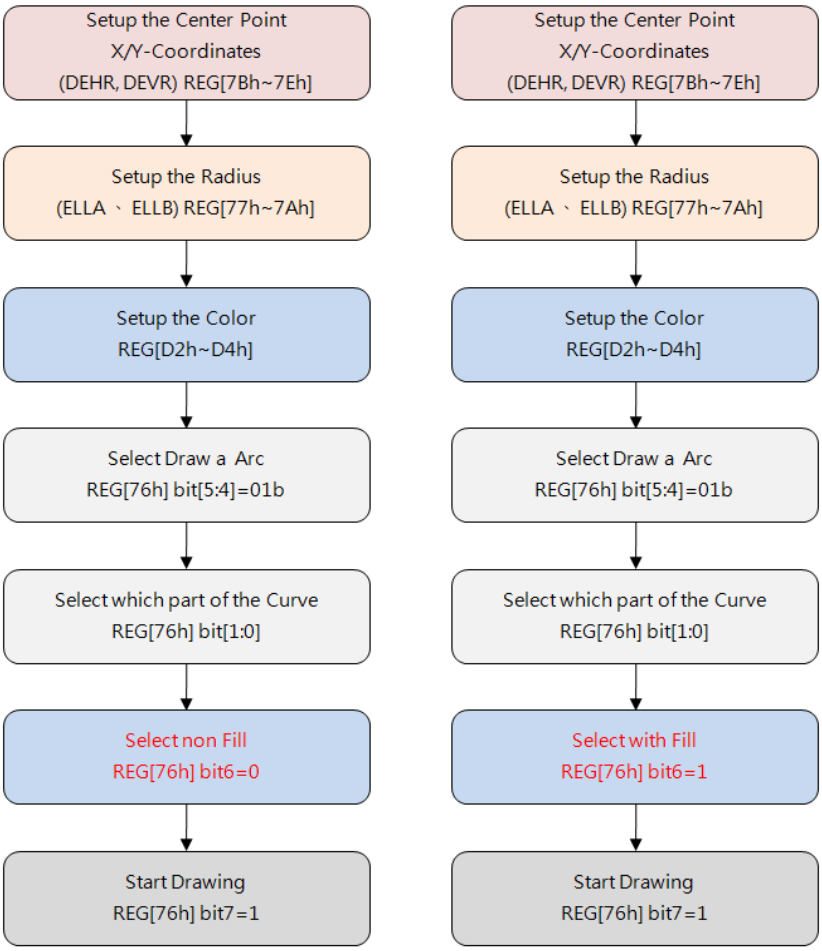


Figure 9-4: Flowchart of Drawing Curve and one-fourth Ellipse

9.3 Drawing Rectangle

To draw a rectangle, the Host has to set the Start coordinate (REG[68h ~ 6Bh]), End coordinate (REG[6Ch ~ 6Fh]), and the color of the rectangle (REG[D2h ~ D4h]), then set REG[76h] bit[5:4] = 10b, and finally enables the drawing function by setting REG[76h] bit7 =1, then LT758x will draw the specified rectangle on the TFT panel automatically. Host can also set REG[76h] bit6 to 1 to fill the rectangle with a specified color.

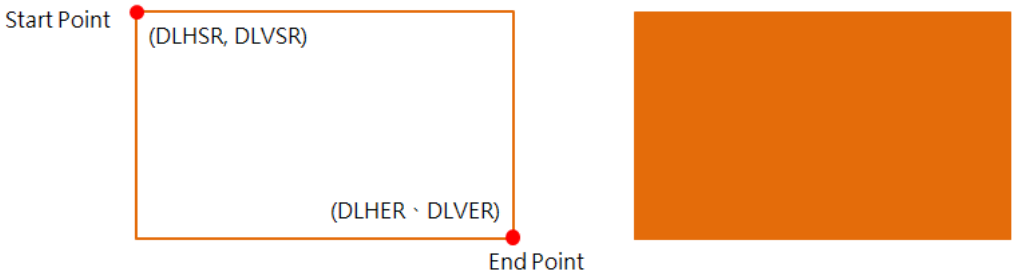


Figure 9-5：Drawing Rectangle

Figure 9-6 shows the flow of drawing a rectangle. The left one is to draw a rectangle without fill, and the right one is to draw a rectangle with fill. Note that the Start and End points must be located within the Active Window.

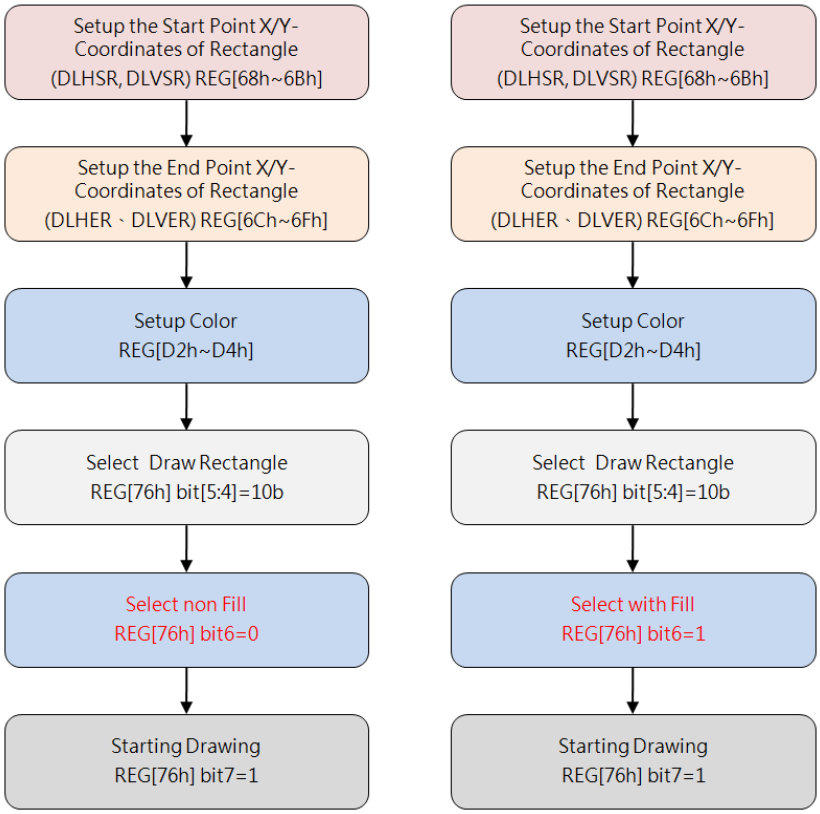


Figure 9-6：Flowchart of Drawing a Rectangle

9.4 Drawing Line

To draw a Line, the Host has to set the Start coordinate (REG[68h ~ 6Bh]), End coordinate (REG[6Ch~ 6Fh]), and the color of the line (REG[D2h ~ D4h]), then set REG[67h] bit[4:1] = 0000b, and finally enables the drawing function by setting REG[67h] bit7 = 1, then LT758x will draw a Line on the TFT panle automatically.

Figure 9-7 shows the flow of drawing a line. Note that the Start and End points must be located within the Active Window.

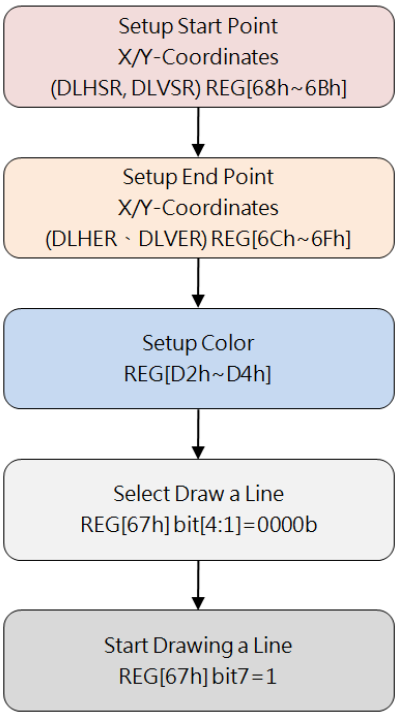


Figure 9-7：Flowchart of Drawing a Line

9.5 Drawing Triangle

To draw a triangle, the Host has to set the 1st Point coordinate (REG[68h ~ 6Bh]) of the triangle, the 2nd Point coordinate (REG[6Ch ~ 6Fh]), the 3rd Point coordinate (REG[70h ~ 73h]), and the color of the triangle (REG[D2h ~ D4h]), then set REG[67h] bit[4:1] = 0001b, and finally enables the drawing function by setting REG[67h] bit7 =1, then LT758x will draw the specified triangle on the TFT panel automatically. Host can also set REG[67h] bit5 = 1 to fill the triangle with a specified color.

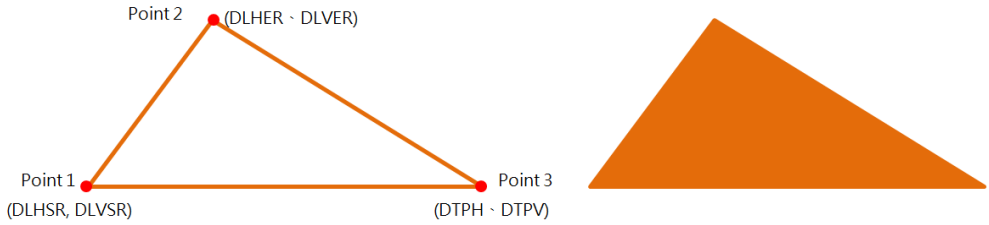


Figure 9-8：Drawing Triangle

Figure 9-9 shows the flow of drawing a triangle. The left one is to draw a triangle without fill, and the right one is to draw a triangle with fill. Note that Point 1, Point 2, and Point 3 must be located within the Active Window.

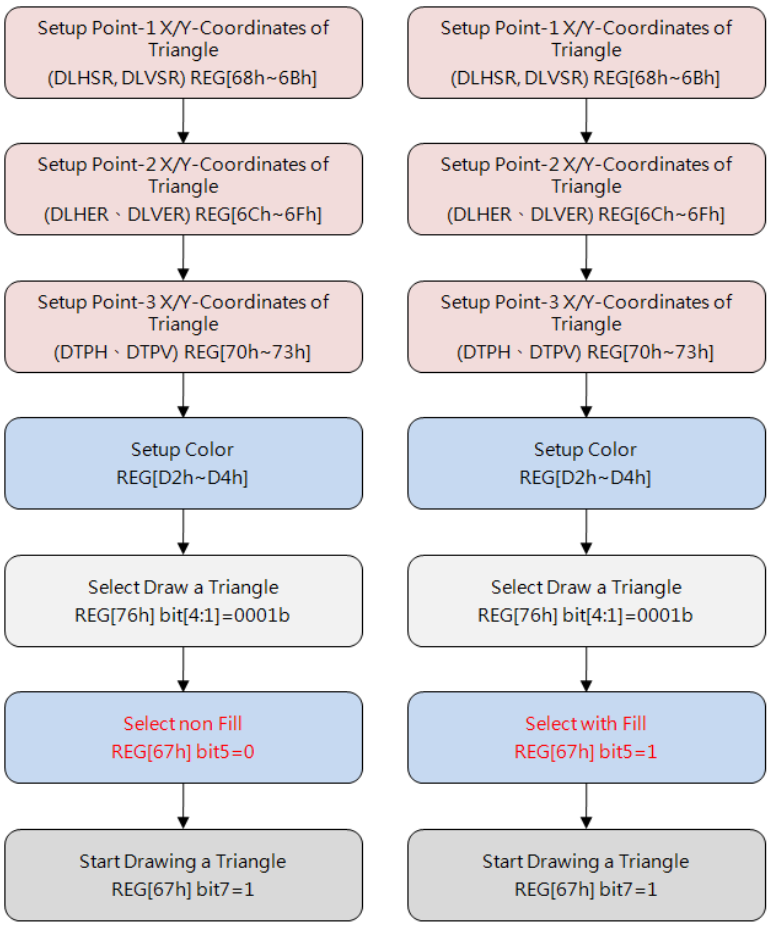


Figure 9-9：Flowchart of Drawing a Triangle

9.6 Drawing Rounded-Rectangle

To draw a rounded-rectangle, the Host has to set the Start coordinate (REG[68h ~ 6Bh]), End coordinate (REG[6Ch ~ 6Fh]), the rounded Radius (REG[77h ~ 7Ah]), and the color (REG[D2h ~ D4h]), then set REG[76h] bit[5:4] = 11b, and finally enables the drawing function by setting REG[76h] bit7 = 1, then LT758x will draw a rounded rectangle on the TFT panel automatically. Host can also set REG[76h] bit6 = 1 to fill the rounded-rectangle with a specified color. As shown in Figure 9-10, R1 is the long axis radius, and R2 is the short axis radius.

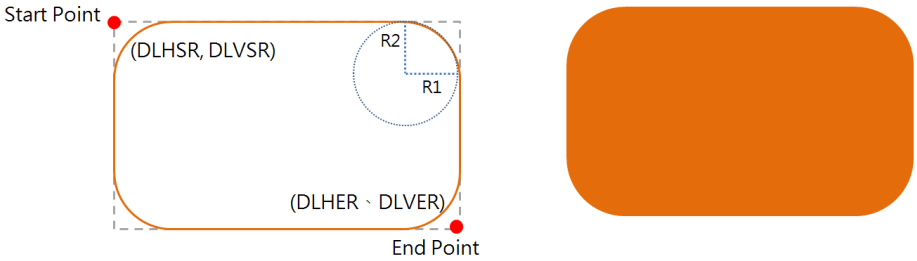


Figure 9-10：Drawing Rounded-Rectangle

Note1: DLHER-DLHSR must be larger than 2*R1 + 1
Note2: DLVER-DLVSR must be larger than 2*R2 + 1

Figure 9-11 shows the flow of drawing a rounded rectangle. The left flowchart is to draw a rounded-rectangle without fill, and the right flowchart is with fill. Note that the Start and End points must be located within the Active Window.

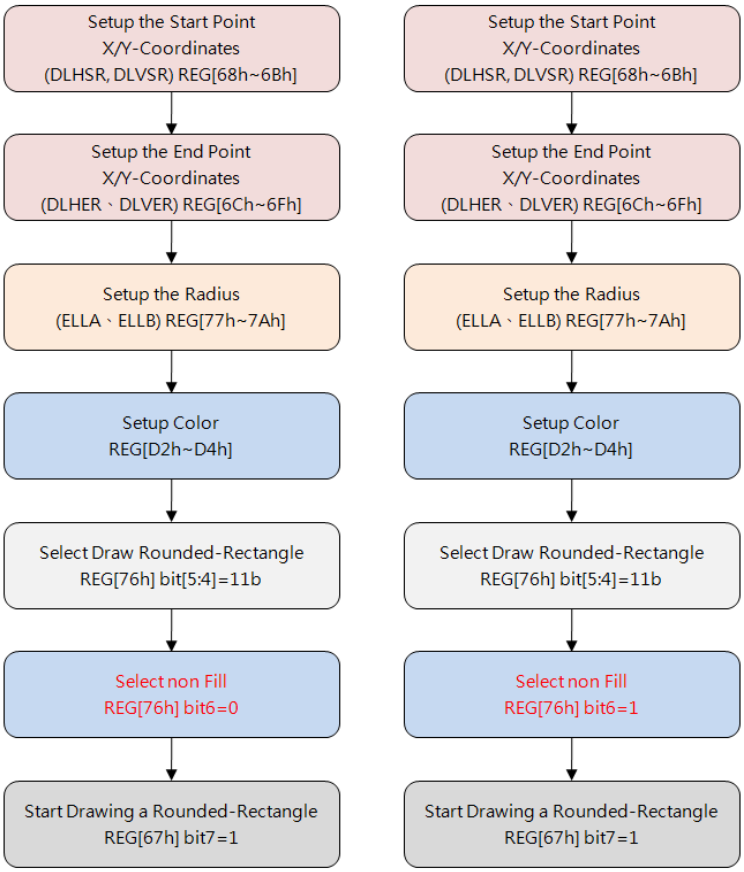


Figure 9-11：Flowchart of Drawing a Reounded-Rectanble

10. BitBlock Transfer Engine (BTE)

LT758x has an embedded high performance hardware engine – BitBlock Transfer Engine (BTE), which is designed to accelerate data loading, transferring, and additional logic processing. BTE supports basic data processing operations as listed in Table 10-1, and additional Raster operations as listed in Table 10-2. Using BTE can speed up the data processing without involving the Host, and therefore the Host workload can be reduced.

If a data block needs to be loaded, moved, and processed at the same time, users can get it done by BTE. Users can combine BTE operations and ROP functions to implement various data processing functions. Once the BTE is properly set and started, it will keep active(busy) until the task is done. There are two ways to get BTE working status. One is to check BTE_CTRL0 (REG[90h] bit4) or the Status Register (STSR) bit3. The other is to monitor hardware interrupt: connecting INT# pin to the Host, once an interrupt is detected, check REG[0Ch] bit2 to find out if BTE operation is complete.

。

Table 10-1：BTE Operation

Operation Code REG[91h] Bits [3:0]	Description
0000b	MCU Write with ROP.
0001b	Not used
0010b	Memory Copy（move）with ROP.
0011b	Not used
0100b	MCU Write with Chroma Keying（w/o ROP）
0101b	Memory Copy（move）with Chroma Keying（w/o ROP）
0110b	Pattern Fill with ROP
0111b	Pattern Fill with Chroma Keying（w/o ROP）
1000b	MCU Write with Color Expansion（w/o ROP）
1001b	MCU Write with Color Expansion and Chroma Keying（w/o ROP）
1010b	Memory Copy with Opacity（w/o ROP）
1011b	MCU Write with Opacity（w/o ROP）
1100b	Solid Fill（w/o ROP）
1101b	Not used
1110b	Memory Copy with Color Expansion（w/o ROP）
1111b	Memory Copy with Color Expansion and Chroma Keying（w/o ROP）

Table 10-2：ROP Function

ROP Bits REG[91h] bit[7:4]	Boolean Function Operation
0000b	0 (Blackness)
0001b	~S0・~S1 or ~ (S0+S1)
0010b	~S0・S1
0011b	~S0
0100b	S0・~S1
0101b	~S1
0110b	S0^S1
0111b	~S0+~S1 or ~ (S0・S1)
1000b	S0・S1
1001b	~ (S0^S1)
1010b	S1
1011b	~S0+S1
1100b	S0
1101b	S0+~S1
1110b	S0+S1
1111b	1 (Whiteness)

Note：

1. For ROP function,
   S0：data from Source 0，S1：data from Source 1，D：Data for the Destination
2. For pattern fill functions，the source data indicates the pattern data。

Examples：

If ROP = Ch, D = S0

If ROP = Eh, D = S0 + S1

If ROP = 2h, D = ~S0・S1

If ROP = Ah, D = S1

Table 10-3：Color Expansion Function

ROP Bits REG[91h] bit[7:4]	Start Bit Position for Color Expansion BitBLT Operation Code = 1000/1001/1110/1111	
	16bits MCU Interface	8bits MCU Interface
0000b	Bit0	Bit0
0001b	Bit1	Bit1
0010b	Bit2	Bit2

ROP Bits REG[91h] bit[7:4]	Start Bit Position for Color Expansion BitBLT Operation Code = 1000/1001/1110/1111	
	16bits MCU Interface	8bits MCU Interface
0011b	Bit3	Bit3
0100b	Bit4	Bit4
0101b	Bit5	Bit5
0110b	Bit6	Bit6
0111b	Bit7	Bit7
1000b	Bit8	Invalid
1001b	Bit9	Invalid
1010b	Bit10	Invalid
1011b	Bit11	Invalid
1100b	Bit12	Invalid
1101b	Bit13	Invalid
1110b	Bit14	Invalid
1111b	Bit15	Invalid

10.1 BTE Settings

S0, S1, and D can be set as any memory addresses. In adition, before processing the ROP operation, the start position (X, Y coordinates) must be designated.

- S0 Address Register: REG[93h], REG[94h], REG[95h], REG[96h], REG[97h], REG[98h], REG[99h], REG[9Ah], REG[9Bh], REG[9Ch]
- S1 Address Register: REG[9Dh], REG[9Eh], REG[9Fh], REG[A0h], REG[A1h], REG[A2h], REG[A3h], REG[A4h], REG[A5h], REG[A6h]
- D Address Register: REG[A7h], REG[A8h], REG[A9h], REG[AAh], REG[ABh], REG[ACh], REG[ADh], REG[AEh], REG[AFh], REG[B0h]

10.2 Color Palette RAM

LT758x has an embedded Color Palette RAM for 8-bits Alpha Blending function. Real colors can be retrieved by searching Color Palette RAM. As shown in Figure 10-1, Color Palette RAM is organized by 64*12bits. Since MCU writes 8bits data at a time, when it writes every even byte, only the lower 4bits will be stored to the RAM. When using Color Palette RAM, the Register REG[03h] bit[1:0] has to be set to 11b. Color Palette RAM must be written by 128 Bytes (8-bits per Byte) sequence at a time, and it is not readable . Figure 10-2 shows the initialization flow.

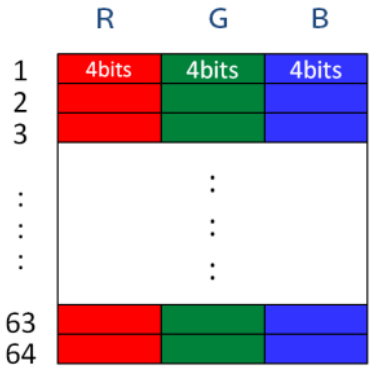


Figure 10-1: Color Palette RAM

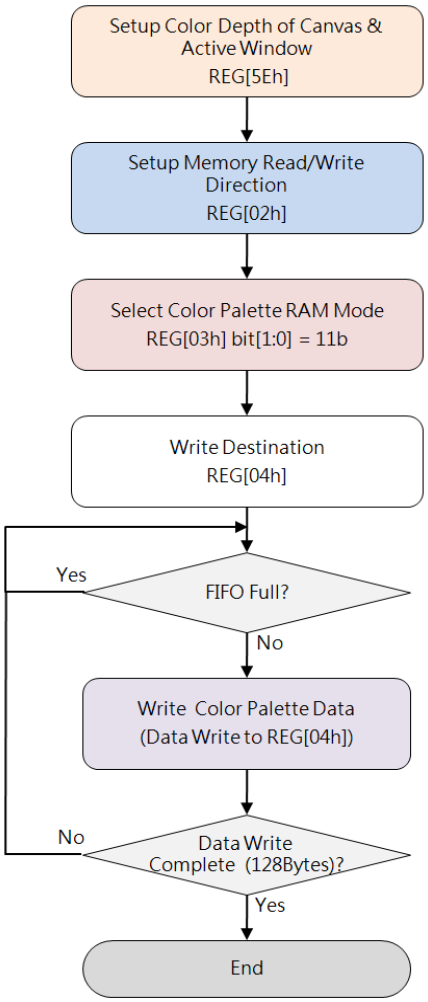


Figure 10-2: Flowchart of Color Palette RAM Initialization

10.3 BTE Memory Access Method

BTE accesses the data of Source and Destination by Block Method. Figure 10-3 shows how users should define S0/S1/D, and shows the directions of Memory Access:

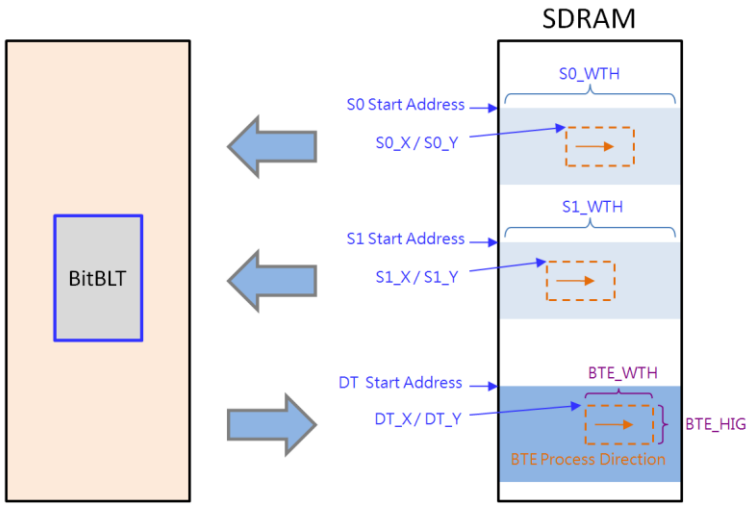


Figure 10-3: BTE Memory Access Example

10.4 BTE Chroma Key (Transparent Color) Function

If the Chroma Key function is enabled, BTE will take the Background Color (defined in the Background Color Register) as the the Chroma Key (Transparent Color), and compare the Chroma Key Data against the S0 data one by one. If the compared result is equal, then BTE will not overwrite the Destination, otherwise write the S0 Data to the Destination.

- **Source Color Depth = 256**
  - Compare S0 Red color vs. REG[D5h] bit[7:5]
  - Compare S0 Green color vs. REG[D6h] bit [7:5]
  - Compare S0 Blue color vs. REG[D7h] bit[7:6]

- **Source Color Depth = 65K**
  - Compare S0 Red color vs. REG[D5h] bit[7:3]
  - Compare S0 Green color vs. REG[D6h] bit[7:2]
  - Compare S0 Blue color vs. REG[D7h] bit[7:3]

- **Source Color Depth = 16.7M**
  - Compare S0 Red color vs. REG[D5h] bit[7:0]
  - Compare S0 Green color vs. REG[D6h] bit[7:0]
  - Compare S0 Blue color vs. REG[D7h] bit[7:0]

10.5 BTE Operation Detail

10.5.1 MCU Write with ROP

This BTE operation is selected by setting REG[91h] bits[3:0] = 0000b. The operation is to transfer Source 0 data (from MCU write) to the Destination, and supports all 16 ROP functions. Figure 10-4 shows an example of the operation result while the ROP register (REG[91h] bits[7:4]) is set to 1100b.

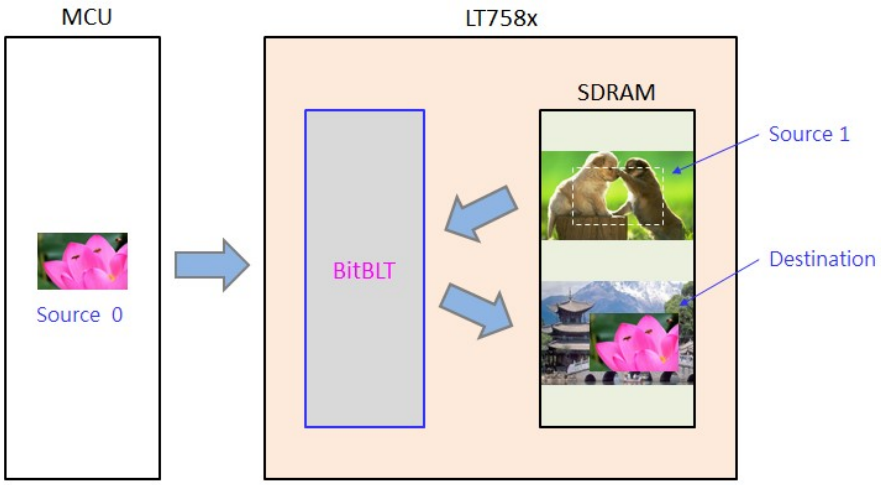


Figure 10-4: Example of MCU Write with ROP

The suggested programming steps and register settings are listed below for reference.

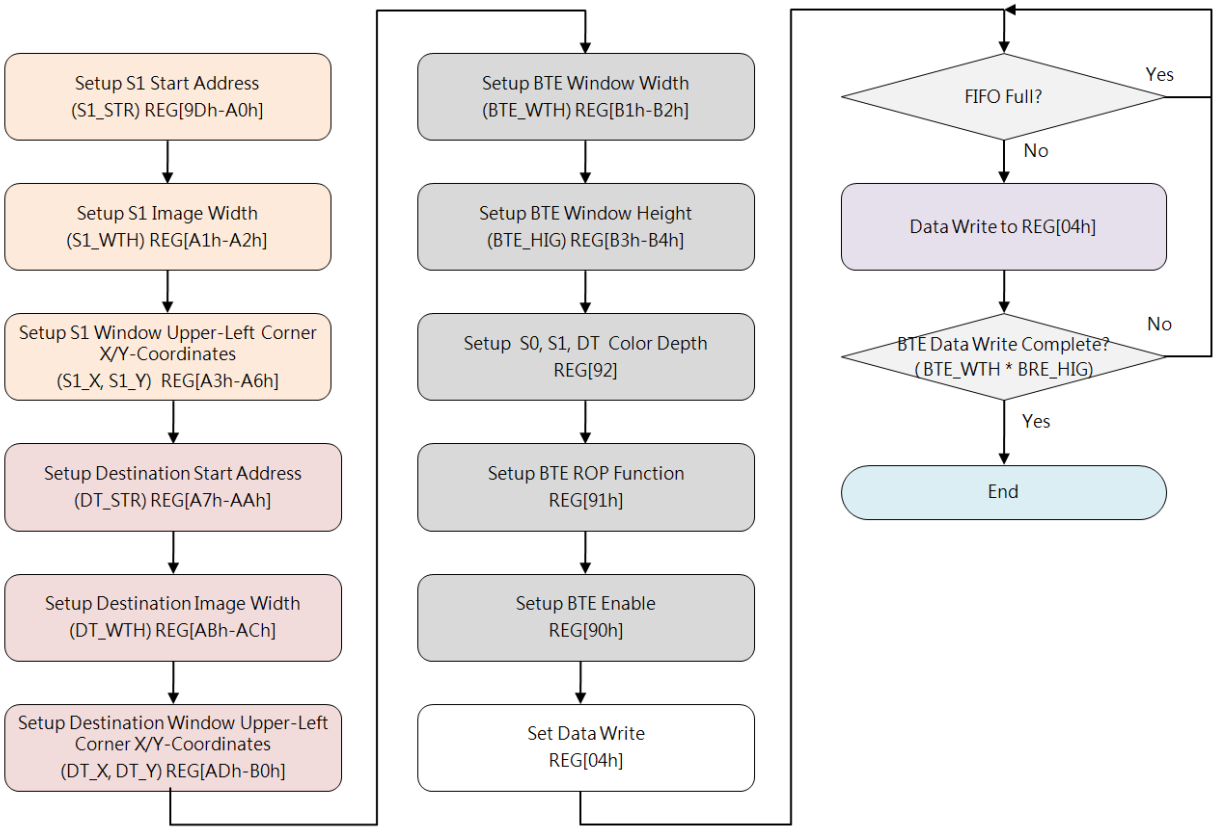


Figure 10-5: Flowchart of MCU Write with ROP

10.5.2 Memory Copy (move) with ROP

This BTE operation is selected by setting REG[91h] bits[3:0] = 0010b. The operation performs Memory Copy from S0 (from Memory) to the Destination. Below diagram is an example of the operation result while the ROP register (REG[91h] bit[7:4]) is set to 1100b.

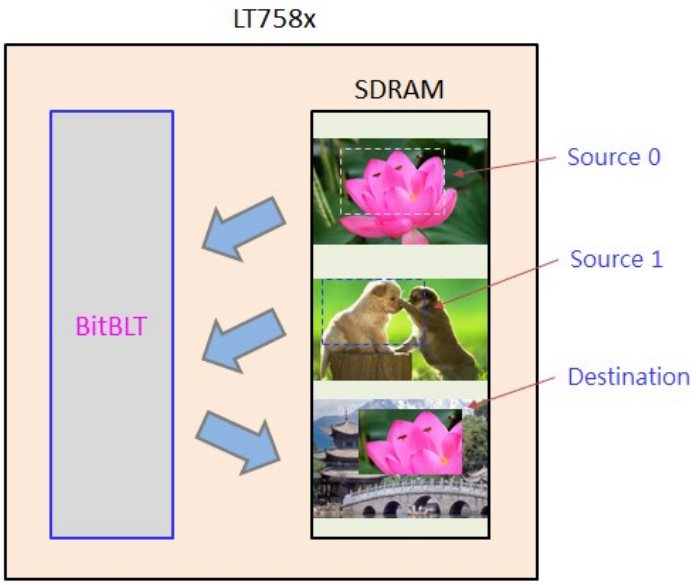


Figure 10-6: Example of Memory Copy with ROP

There are two ways to get BTE status. Figure 10-7 shows one way to get the status by checking the Status Register STSR bit3, and Figure 10-8 shows another way by checking the hardware interrupt.

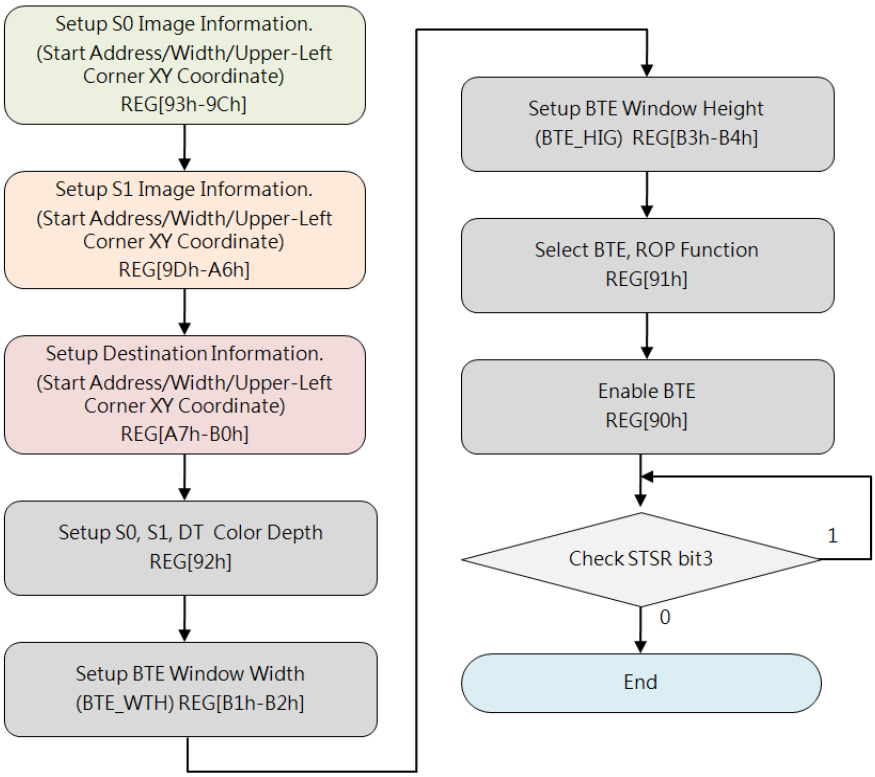


Figure 10-7: Flowchart of Memory Copy with ROP (1)

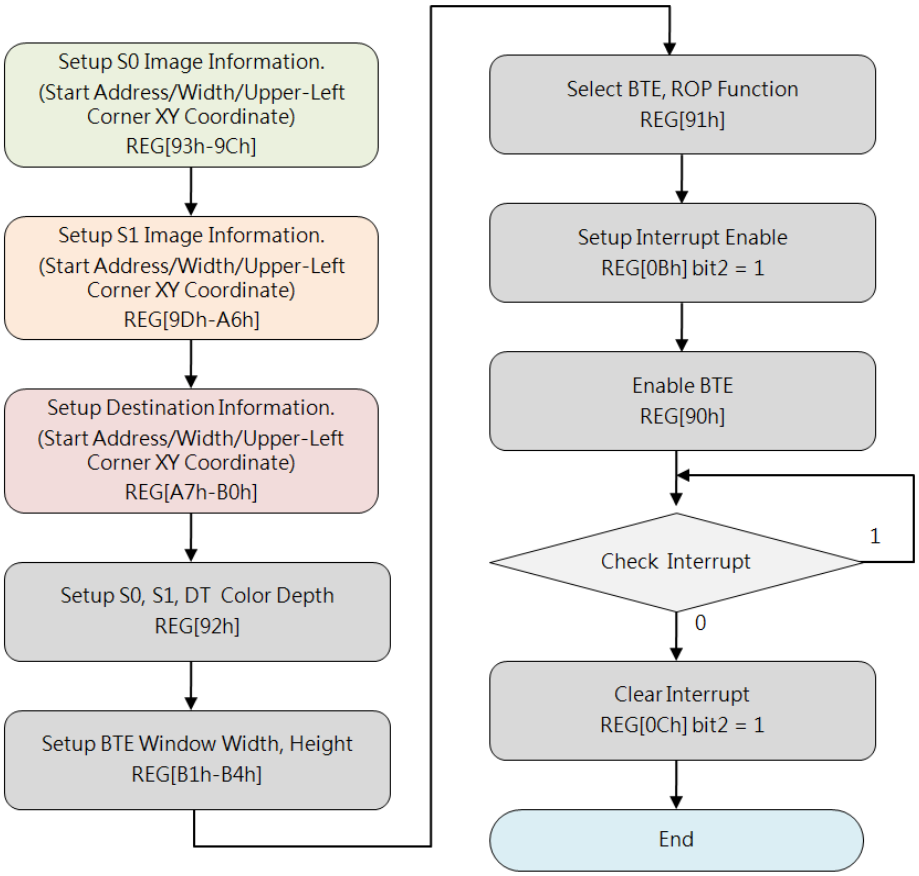


Figure 10-8：Flowchart of Memory Copy with ROP (2)

10.5.3 MCU write with Chroma Key (w/o ROP)

This BTE operation is selected by setting REG[91h] bits[3:0] = 0100b. The operation is to transfer Source 0 (from MCU write) to the Destination, and supports Chroma Key function (referring to Section 10.4). If the Source 0 data is equal to the Chroma Key (the color Data defined in the Background Color Registers REG[D5h ~ D7h]), BTE will take it as a transparent color, which means BTE will not write that color data to the Destination. Below example shows the GREEN color is the background color of Source 0, and is set as the Chroma Key. Therefore, the BTE operation will write the red “TOP” only to the Destination. Please note this BTE operation does not support ROP functions.

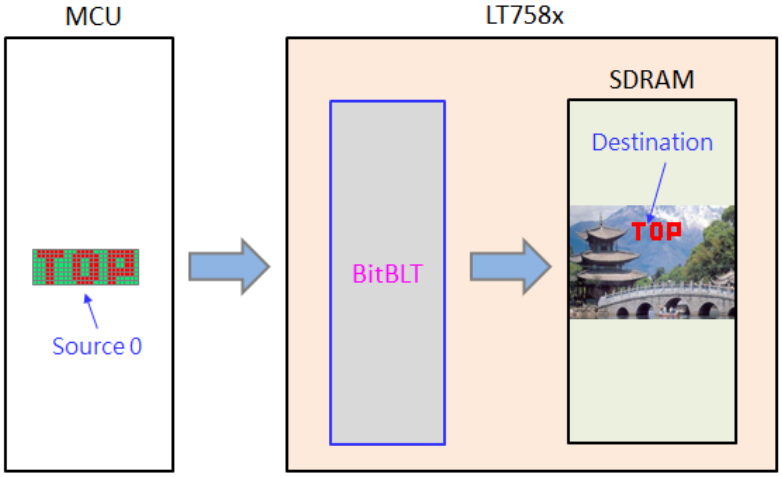


Figure 10-9: Example of MCU Write with Chroma Key

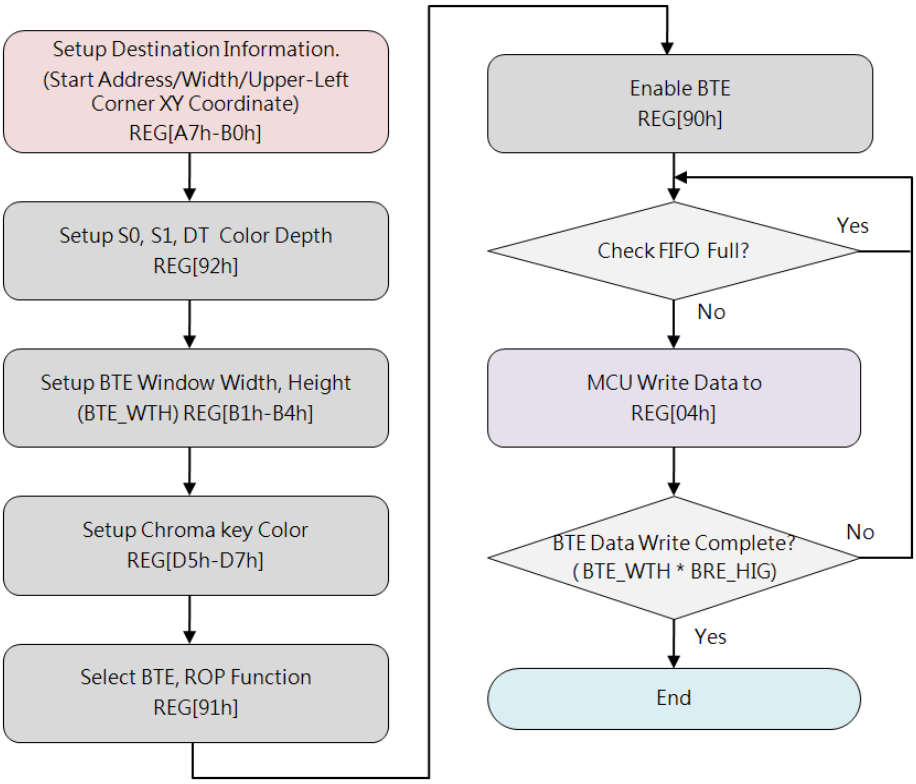


Figure 10-10: Flowchart of MCU Write with Chroma Key

10.5.4 Memory Copy with Chroma Key (w/o ROP)

This BTE operation is selected by setting REG[91h] bits[3:0] = 0101b. The operation performs Memory Copyfrom Source 0 (from Memory) to the Destination, and support Chroma Key function (referring to the Section 10.4).

The operation difference comparing with “MCU Write with Chroma Key” is that Source 0 data comes from the Memory instead of the MCU. Please refer to Section 10.5.3 for more descriptions about the operation, and Figure 10-11 for an example.

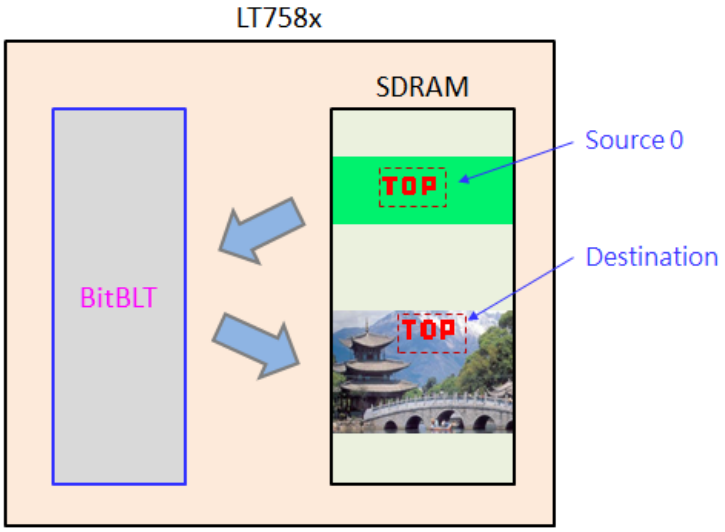


Figure 10-11: Example of Memory Copy with Chroma Key

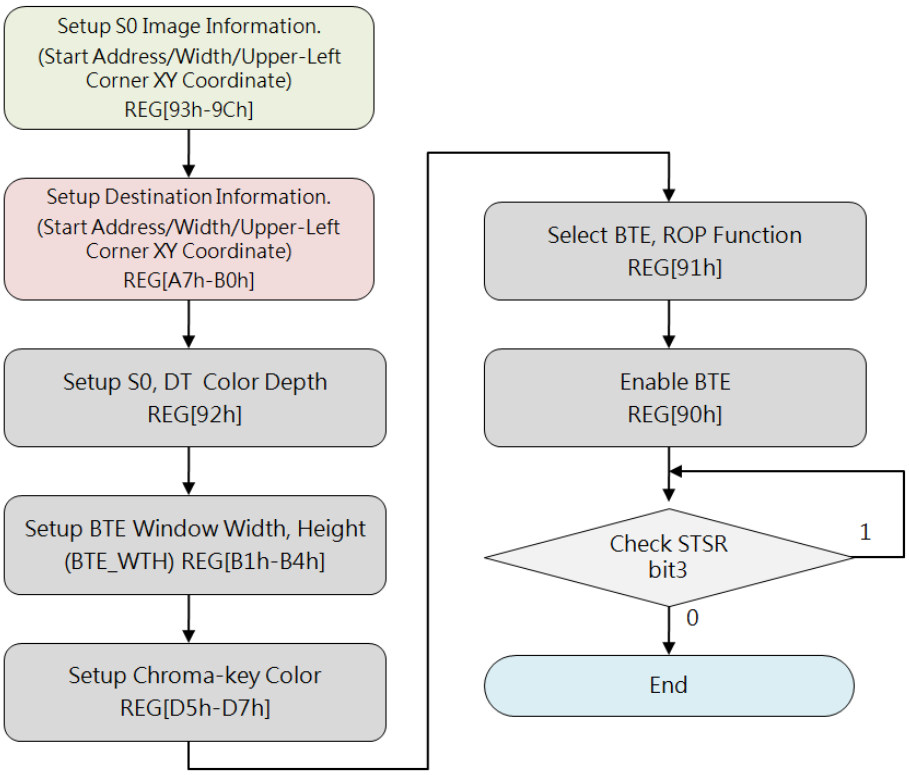


Figure 10-12: Flowchart of Memory Copy with Chroma Key

10.5.5 Pattern Fill with ROP

This BTE operation is selected by setting REG[91h] bits[3:0] = 0110b. The operation is to fill a pattern (from Source 0) repeatedly to a specified rectangular area of the Destination (also known as the BTE window). The pattern should be an array of 8x8 or 16x16 pixels stored in the Memory. The operation can also combine with the 16 ROP functions. This operation can be used to speed up duplicating a matrix of patten into an area. Figure 10-14 shows the result of a six fills with 8x8 pattern, where the ROP register REG[91h] bit[7:4] is set as 1100b.

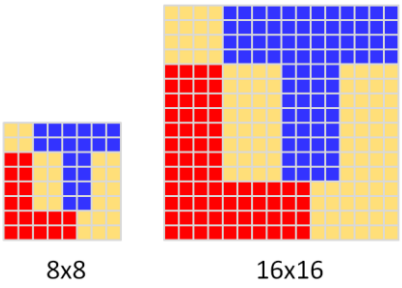


Figure 10-13: Pattern Format

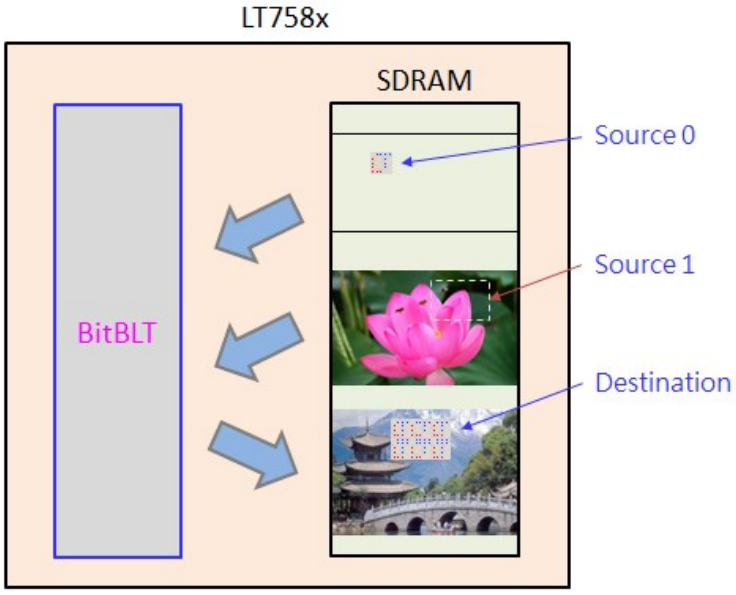


Figure 10-14: Example of Pattern Fill with ROP

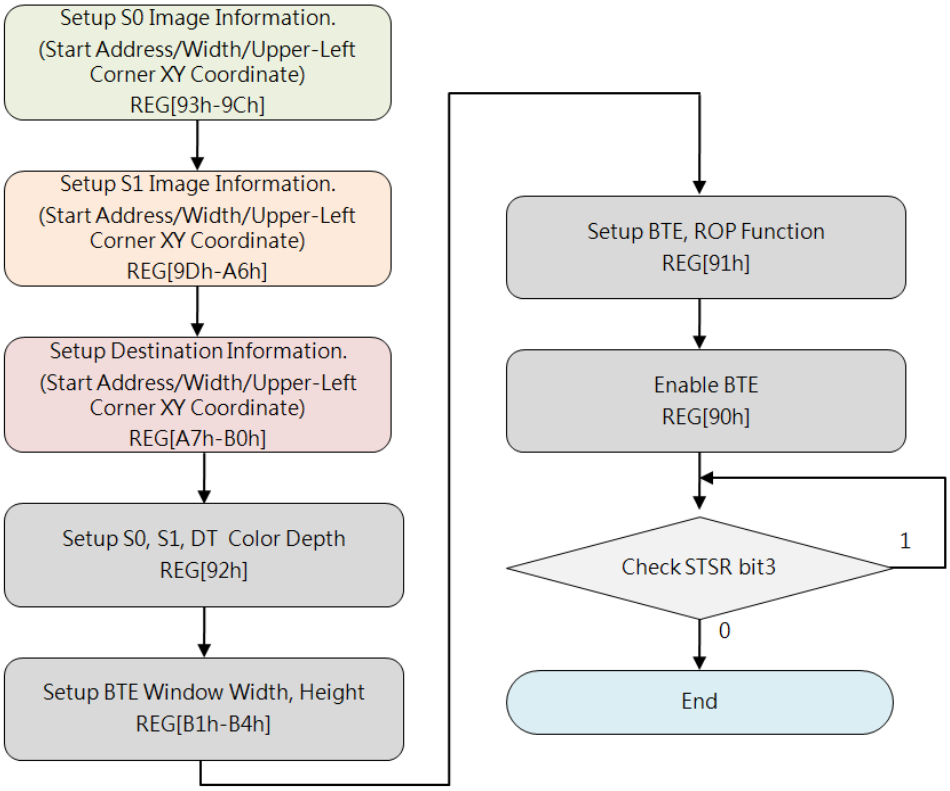


Figure 10-15: Flowchart of Pattern Fill with ROP

10.5.6 Pattern Fill with Chroma Key

This BTE operation is selected by setting REG[91h] bits[3:0] = 0111b. The operation is to fill a pattern of 8x8 or 16x16 pixels (from Source 0) repeatedly to a specified regtangular area of the Destination (also known as the BTE window), and supports Chroma Key function (referring to Section 10.4). If any color data of the patten is equal to the Chroma Key data, BTE will not write that part of data to the Destination. Please note this BTE operation does not support ROP functions.

In the example shown below, the background color of the pattern (red “T”) is orange. Since the orange color is set as the Chroma Key, BTE fills red “T” only to the Desitantion.

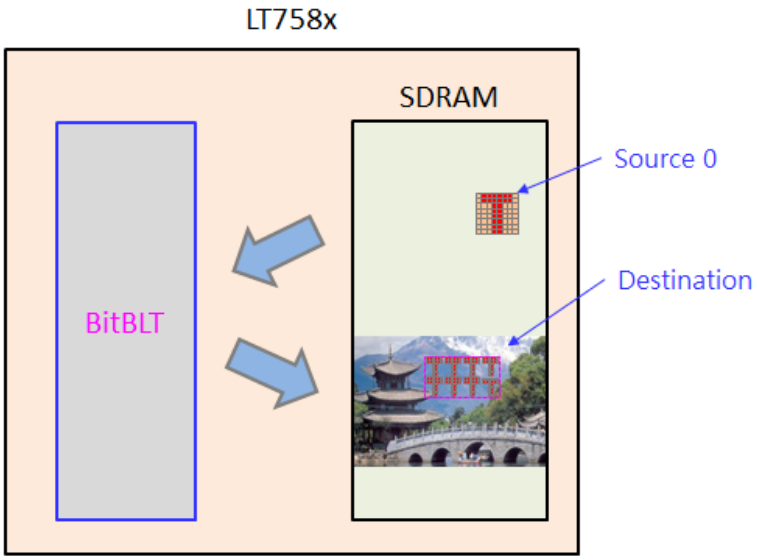


Figure 10-16: Example of Pattern Fill with Chroma Key

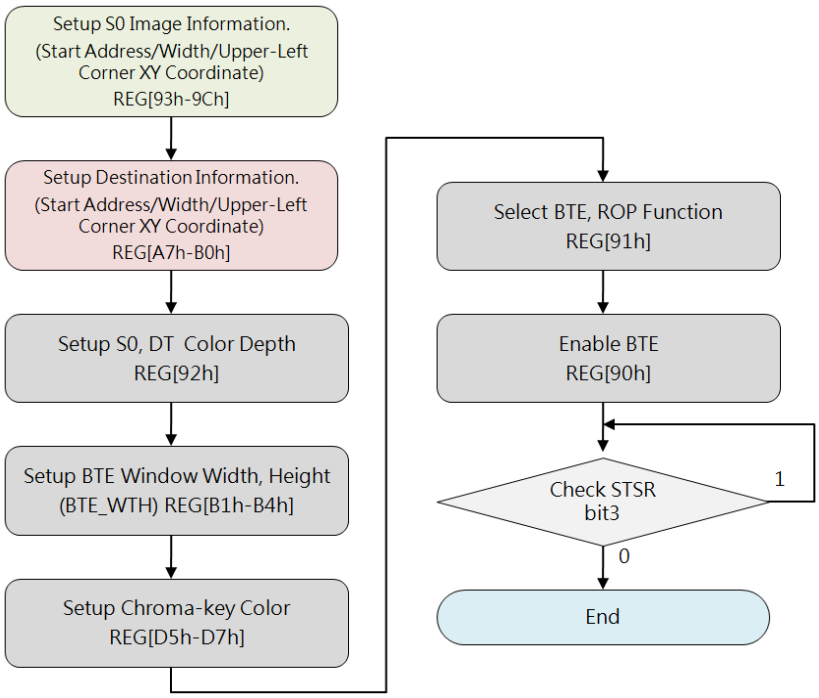


Figure 10-17: Flowchart of Pattern Fill with Chroma Key

10.5.7 MCU Write with Color Expansion

This BTE operation is selected by setting REG[91h] bits[3:0] = 1000b. The operation is to translate the bit-wise monochrome data (Source 0 from MCU Write) to the byte-wise color data, and then store the color data to the Destination in SDRAM. In this operation, Source 0 color depth (REG[92h] bits[6:5]) is ignored. BTE takes MCU interface width as Source 0 color depth (Word width), that is, if MCU interface is 8bits, then the color depth is 8bits, and if the MCU interface is 16bits, then the color depth is 16bits. Users should set the needed Start Bit to REG[91h] bit[7:4] against the corresponding color depth. As shown in Table 10-3, bit[7] ~ bit[0] are available for 8bits MCU interface, and bit[15] ~ bit[0] are available for 16bits MCU Interface. BTE disassembles word by word to bit sequences (from MSB to LSB) against the Row Line of the source image (from left to right), and sequently expands bit by bit until the BTE width reaches the end. The result to the Destination is that BTE expands data_1b to Foreground color and data_0b to Background color. Any bits before the Start Bit and other bits uncovered by the BTE Window will be discarded.

Below example shows expanding data_1b to RED (Foreground color), and data_0b to BLUE (Background color). Therefore, BTE fills red “TOP” together with BLUE background to the Destination:

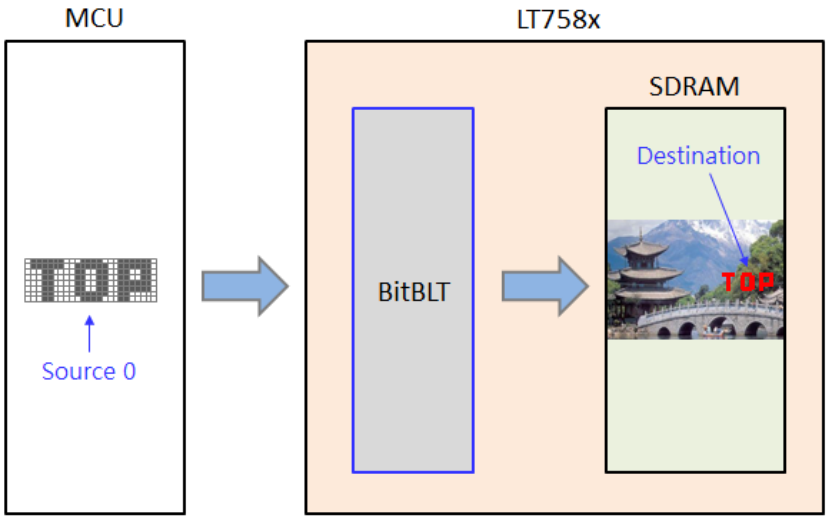


Figure 10-18: Example of MCU Write with Color Expansion

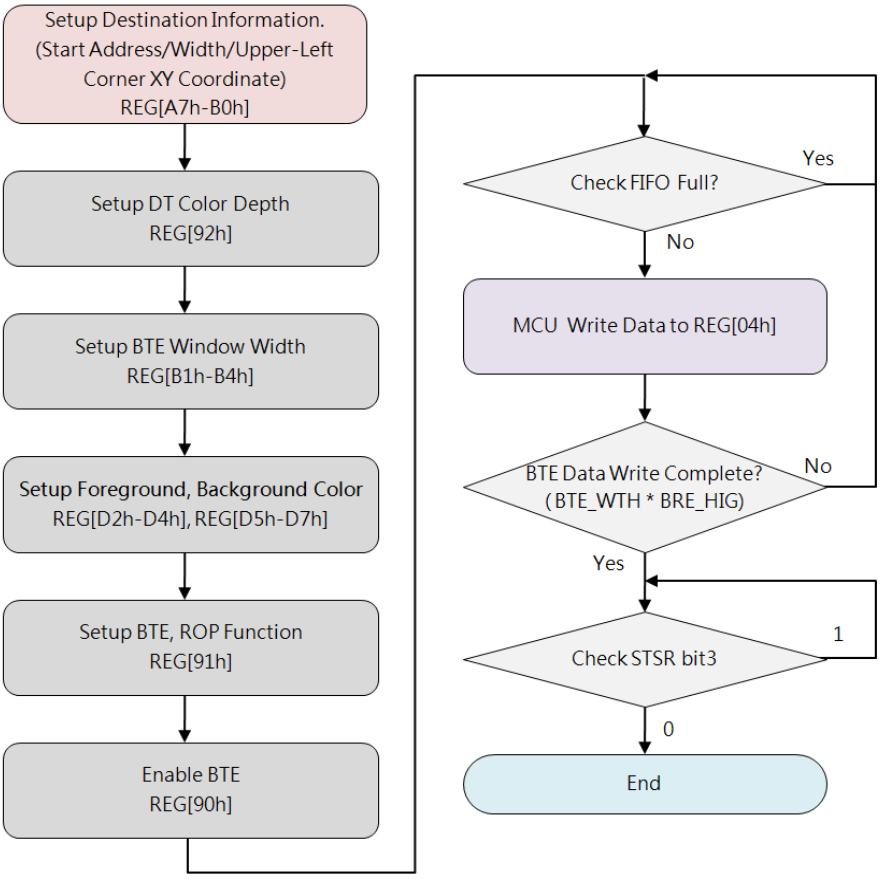


Figure 10-19: Flowchart of MCU Write with Color Expansion

If ROP = 7, Foreground Color is Red, Background Color is Khaki, and BTE width = 23, then the color expansion result is as shown in Figure 10-20.
If ROP = 3, and the rest of settings remains the same, then the result is shown as Figure 10-21.

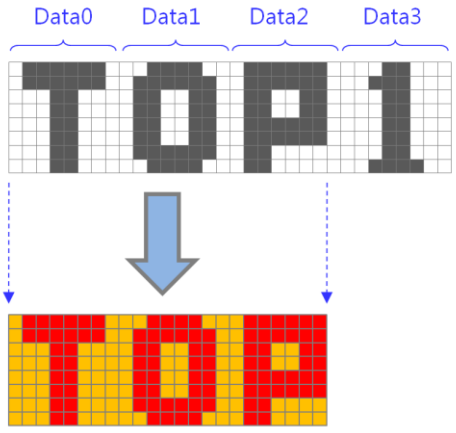


Figure 10-20: Example of MCU Write with Color Expansion (ROP = 7)

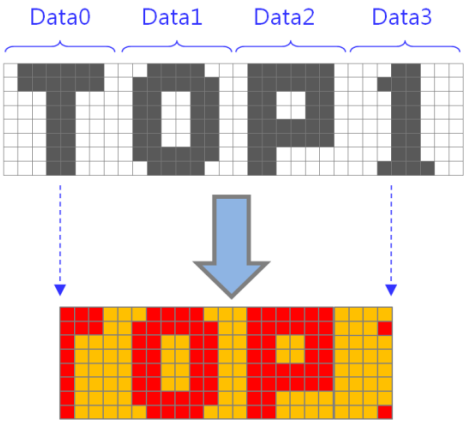


Figure 10-21: Example of MCU Write with Color Expansion (ROP = 3)

Note:

1. Sent Data Numbers per Row
   = [BitBLT Width + (MCU I/F bits – Start bit - 1)] / (MCU I/F bits)
   (Take integer with unconditional carry)
2. Total Data Number = (Sent Data Numbers per Row) * BitBLT Height

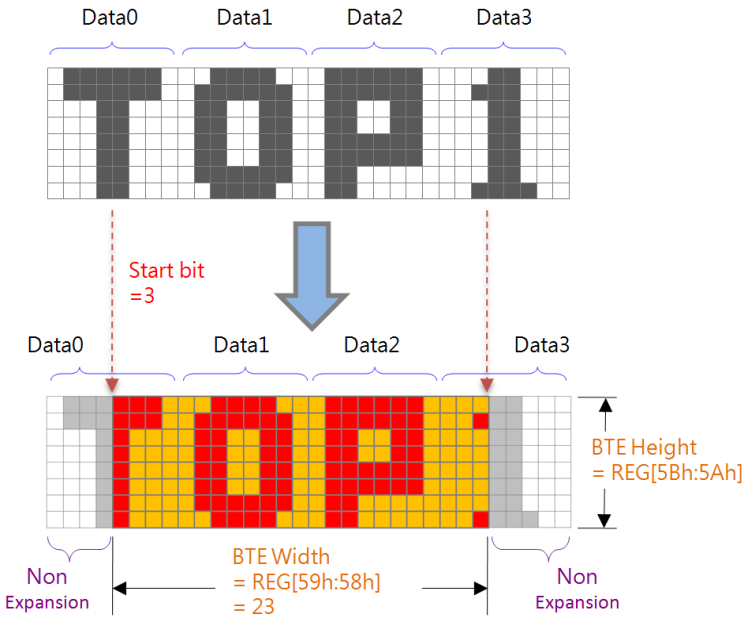


Figure 10-22: Data Format for Color Expansion

Example1: "BitBLT Width"= 50, "MCU I/F bits" = 8 bits,

If Start bit=7, then:

Sent Data Numbers per Row = [{ 50 + (8 - 7 - 1) } / 8] = 7 Bytes

If Start bit=4, then:

Sent Data Numbers per Row = [{ 50 + (8 - 4 - 1) } / 8] = 7 Bytes

Example 2: "BitBLT Width"= 50, "MCU I/F bits" = 16 bits,

If Start bit =15, then:

Sent Data Numbers per Row = [{ 50 + (16 - 15 - 1) } / 16] = 4 Bytes

If Start bit=0, then:

Sent Data Numbers per Row = [{ 50 + (16 - 0 - 1) } / 16] = 5 Bytes

10.5.8 MCU Write with Color Expansion and Chroma Key

This BTE operation is selected by setting REG[91h] bits[3:0] = 1001b. This operation is similar to MCU Write with Color Expansion, yet the difference is that data_0b (background color) will be discarded, BTE expands data_1b only to Foreground Color setting.

Below example shows that data_1b is expanded to RED (Foreground Color), and data_0b is discarded, therefore the result to Destination is RED “TOP” with TRANSPARENT background.

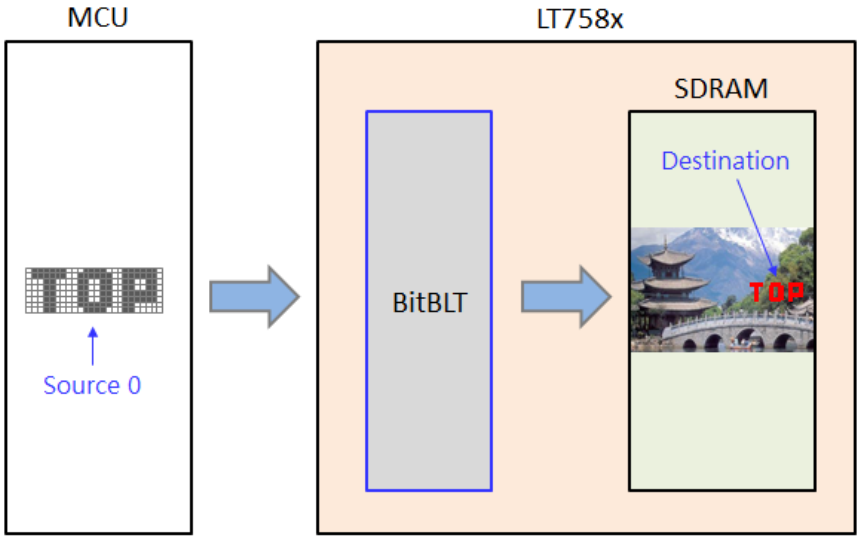


Figure 10-23: Example of MCU Write with Color Expansion and Chroma Key

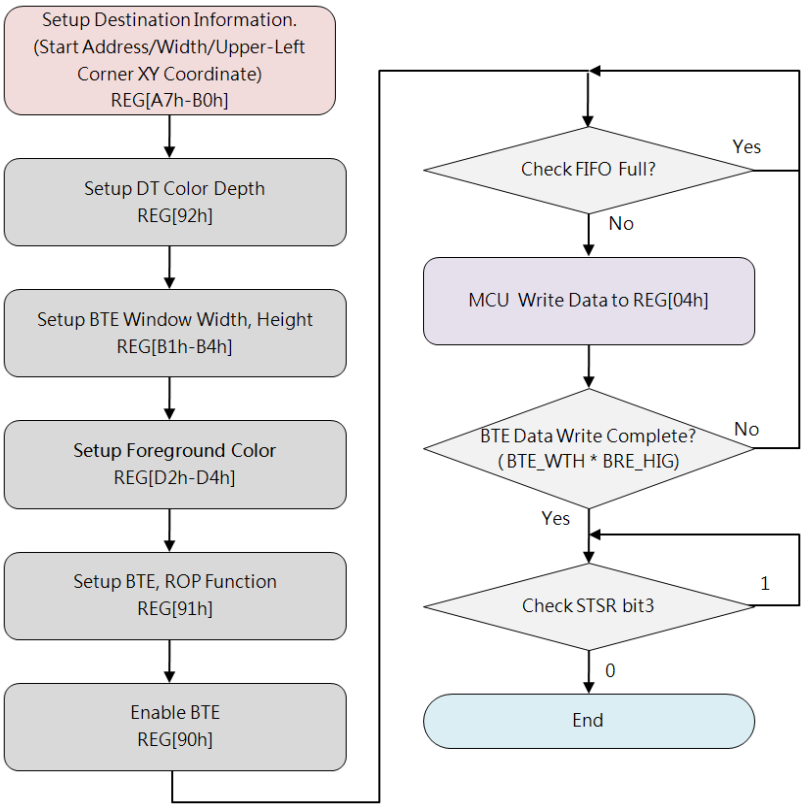


Figure 10-24: Flowchart of MCU Write with Color Expansion and Chroma Key

10.5.9 Memory Copy with Opacity

This BTE operation is selected by setting REG[91h] bits[3:0] = 1010b. This operation performs bleding Source 0 and Source 1, and transferring the result to Destination. Two modes are available: Picture Mode and Pixel Mode.

Pixture Mode is for 8bpp/16bpp/24bpp format and it uses the same Opacity Value (Alpha Level) for the whole bitmap picture. Opacity Value of Picture Mode is defined in REG[B5h].

Pixel Mode is for 8bpp/16bpp format and it uses individual Opacity Value of Source 1, each pixel of Source 1 has its own Opacity Value such as: for each 16bpp data, the bit[15:12] is Opacity Value, and the bit[11:0] is the color data; for each 8bpp data of S1, the bit[7:6] is Opacity Value, and the bit[5:0] is the Index (Address) of Palette Color RAM pointing to initialized 12-bits Color Depth data.

- **Picture Mode:**

  Destination Data = （Source 0 * Alpha Level） + [Source 1 * （1- Alpha Level）];

- **Pixel Mode 8bpp**:

  Destination Data = （Source 0 * Alpha Level） + [Index palette （Source 1[5:0]） * （1 - Alpha Level）]

- **Pixel Mode 16bpp**:

  Destination Data = （Source 0 * Alpha Level） + [Source 1 [11:0] * （1 - Alpha Level）]

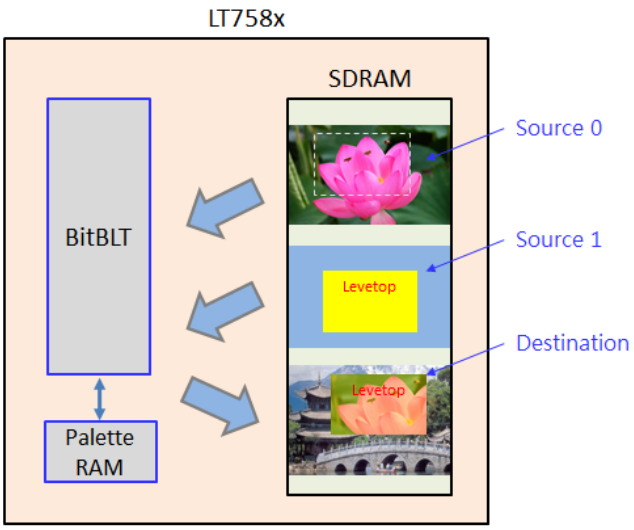


Figure 10-25： Example of Pixel Mode – 8bpp

Table 10-4： Alpha Blending Level of Pixel Mode – 8bpp

Bit[7:6]	Alpha Level
0h	0
1h	10/32
2h	21/32
3h	1

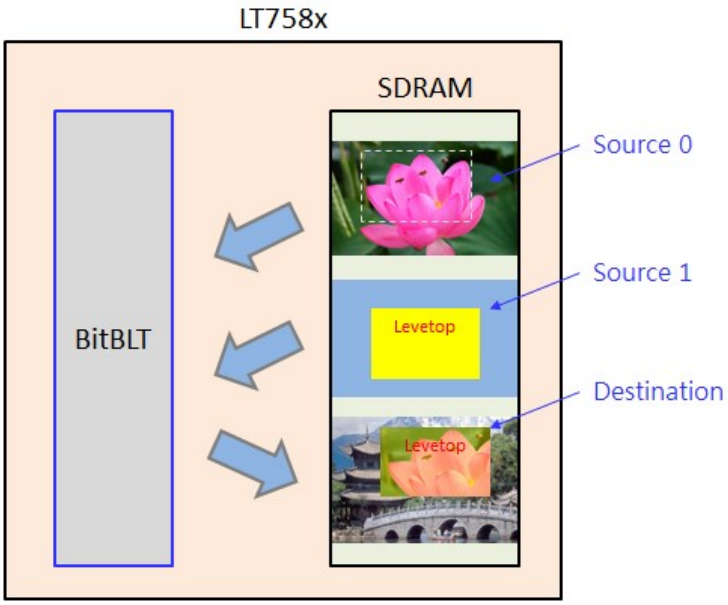


Figure 10-26: Example of Pixel Mode – 16bpp

Table 10-5: Alpha Blending Level of Pixel Mode – 16bpp

Bit[15:12]	Alpha Level
0h	0
1h	2/32
2h	4/32
3h	6/32
4h	8/32
5h	10/32
6h	12/32
7h	14/32
8h	16/32
9h	18/32
Ah	20/32
Bh	22/32
Ch	24/32
Dh	26/32
Eh	28/32
Fh	1

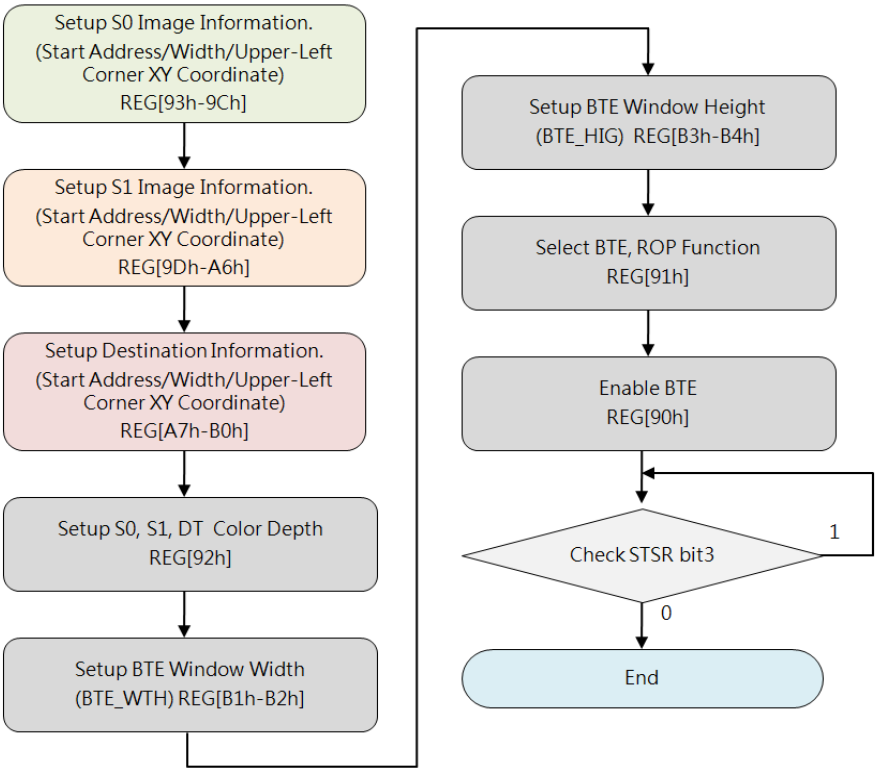


Figure 10-27: Flowchart of Memory Copy with Opacity – Pixel Mode

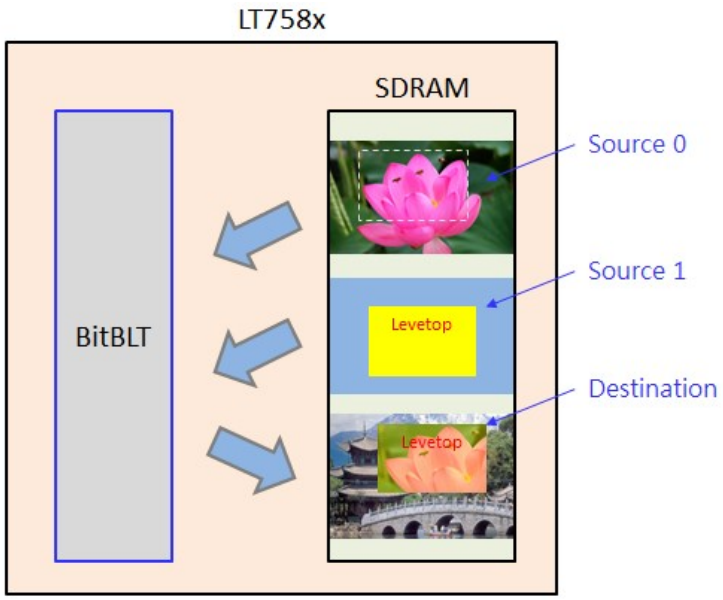


Figure 10-28: Example of Picture Mode

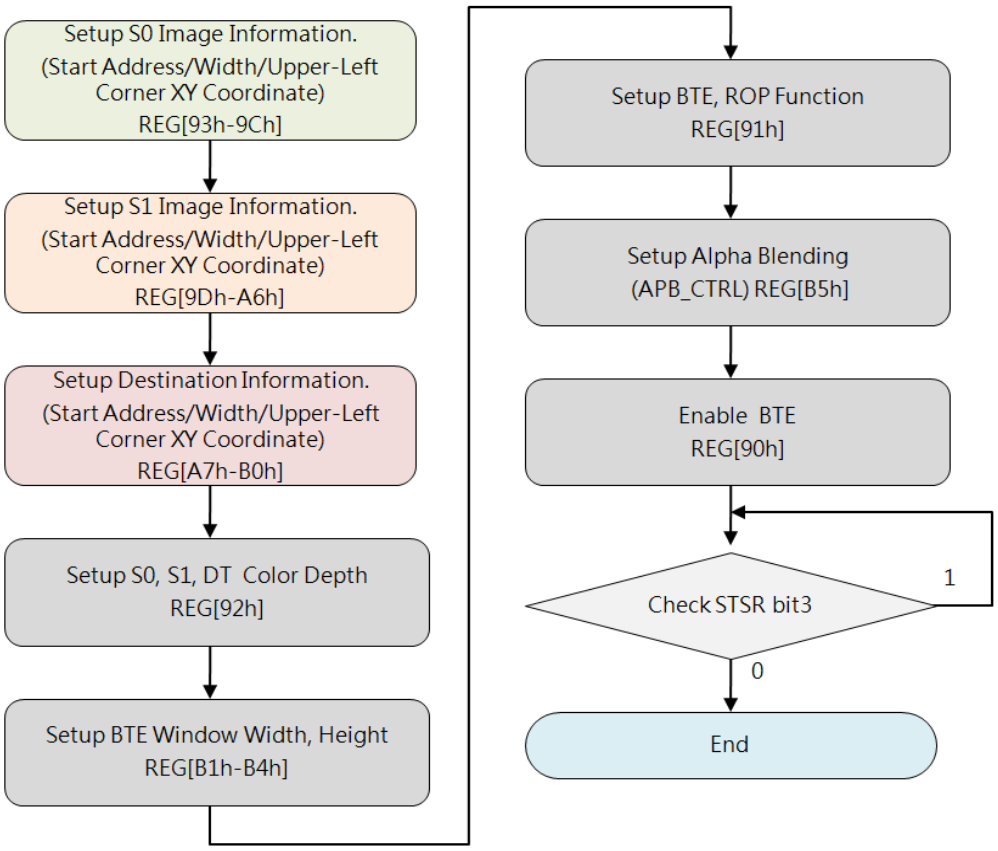


Figure 10-29: Flowchart of Memory Copy with Opacity – Picture Mode

10.5.10 MCU Write with Opacity

This BTE operation is selected by setting REG[91h] bits[3:0] = 1011b. This operation is similar to Memory Copy with Opacity. The difference is that Source 0 is from MCU Write. The Picture Mode and Pixel Mode are also available. Refer to Section 10.5.9 for more operation detail.

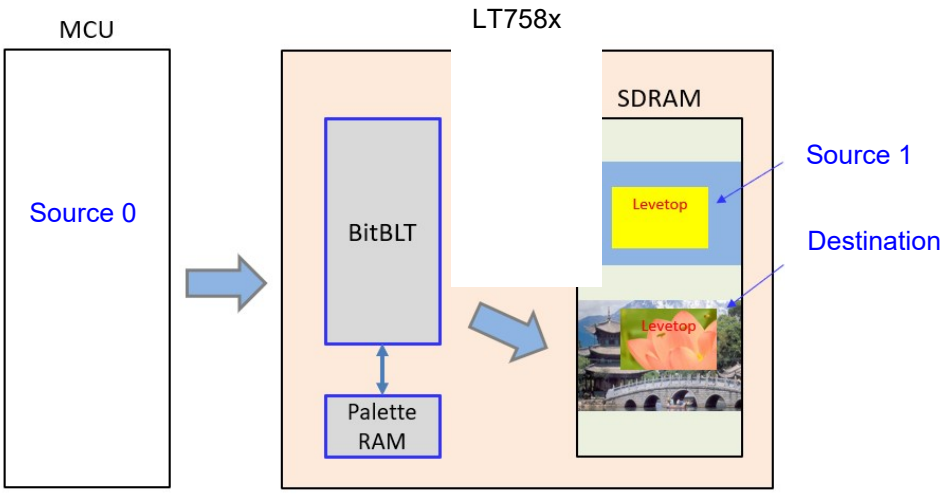


Figure 10-30: Example of MCU Write with Opacity

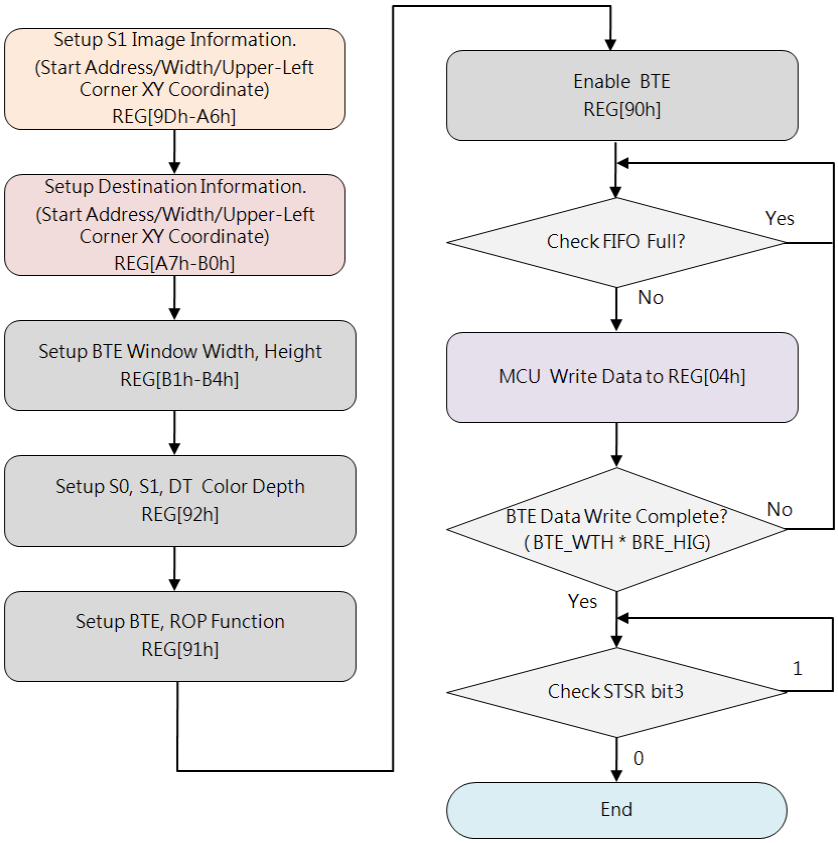


Figure 10-31: Flowchart of MCU Write with Opacity

10.5.11 Memory Copy with Color Expansion

This BTE operation is selected by setting REG[91h] bits[3:0] = 1110b. This operation performs Color Expansion for the Source 0 data (from Memory), it is userful to translate bit-wise monochrome data to byte-wise color data. In this operation, Source 0 data Color Depth (Word Width) is defined in REG[92h] bit[6:5]. Users should set the needed Start Bit to REG[91h] bit[7:4] based on the correspoinding color depth. The bit[7] ~ bit[0] are available for 8bits depth, and bit[15] ~ bit[0] are available for 16bits depth. BTE disassembles word by word to bit sequences (from MSB to LSB) against the Row Line of the source image (from left to right), and sequently expands bit by bit until the BTE width reaches the end. The result to the Destination is that BTE expands data_1b to Foreground color and data_0b to Background color. Any bits before the Start Bit and other bits uncovered by the BTE Window will be discarded.

Below example shows expanding data_1b to RED (Foreground color), and data_0b to BLUE (Background color). Therefore, BTE fills red “TOP” together with BLUE background to the Destination:

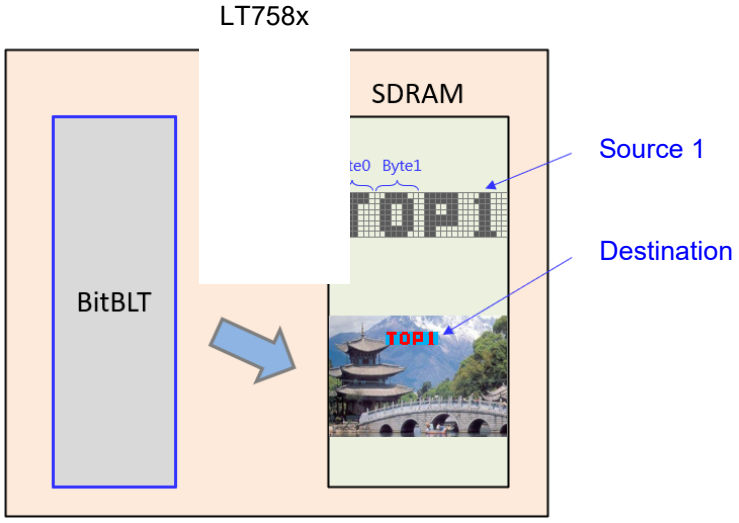


Figure 10-32: Example of Memory Copy with Color Expansion

If ROP = 7, Foreground Color is Red, Background Color is Khaki, and BTE width = 23, then the color expansion result is as shown in Figure 10-33.

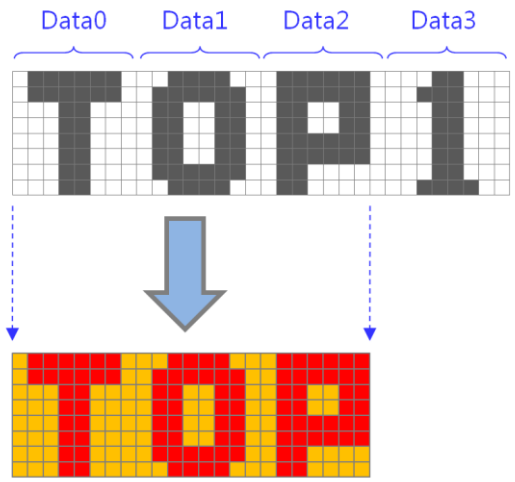


Figure 10-33: Example of Memory Copy with Color Expansion (ROP=7)

If ROP = 4, and the rest of settings remains the same, then the color expansion result is as shown Figure 10-34.

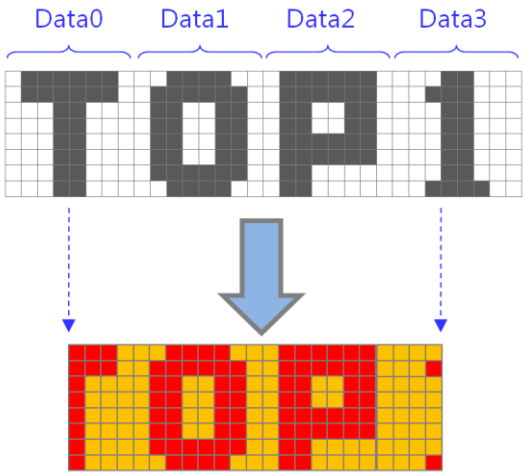


Figure 10-34: Example of Memory Copy with Color Expansion (ROP=3)

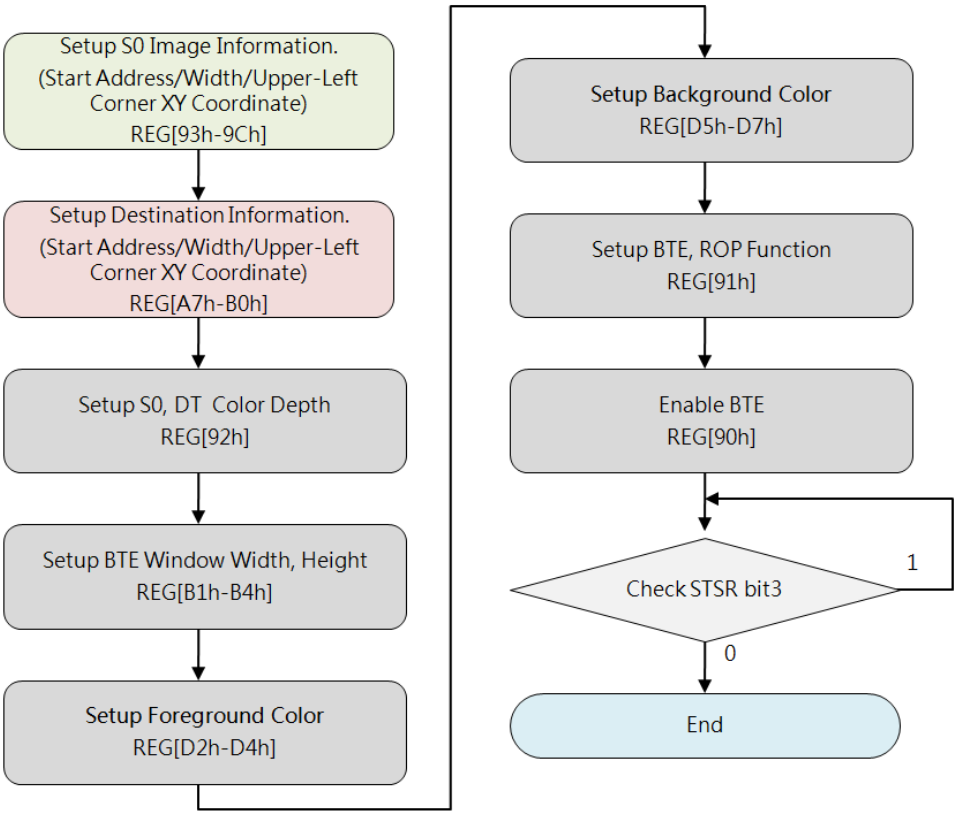


Figure 10-35: Flowchart of Memory Copy with Color Expansion

10.5.12 Memory Copy with Color Expansion and Chroma Key

This BTE operation is selected by setting REG[91h] bits[3:0] = 1111b. This operation is similar to Memory Copy with Color Expansion. The difference is that data_0b will be discarded, and BTE expands only data_1b to Foreground Color.

Below example shows that data_1b is expanded to RED (Foreground Color), and data_0b is discarded. Therefore, the result to the Destination is red “TOP” with transparent background.

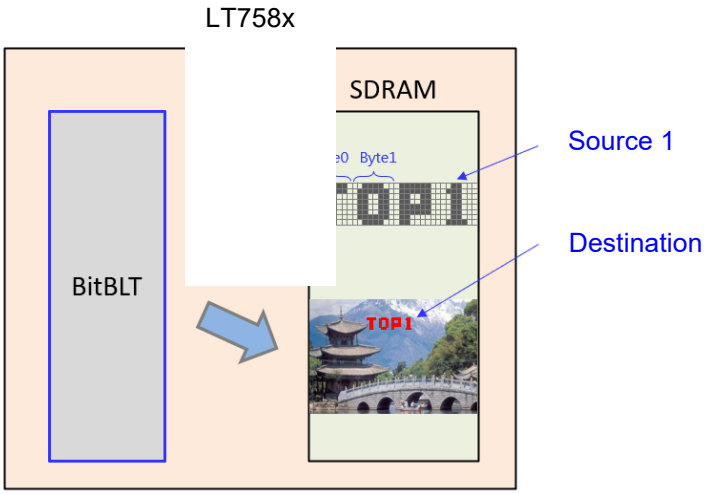


Figure 10-36: Example of Memory Copy with Color Expansion and Chroma Key

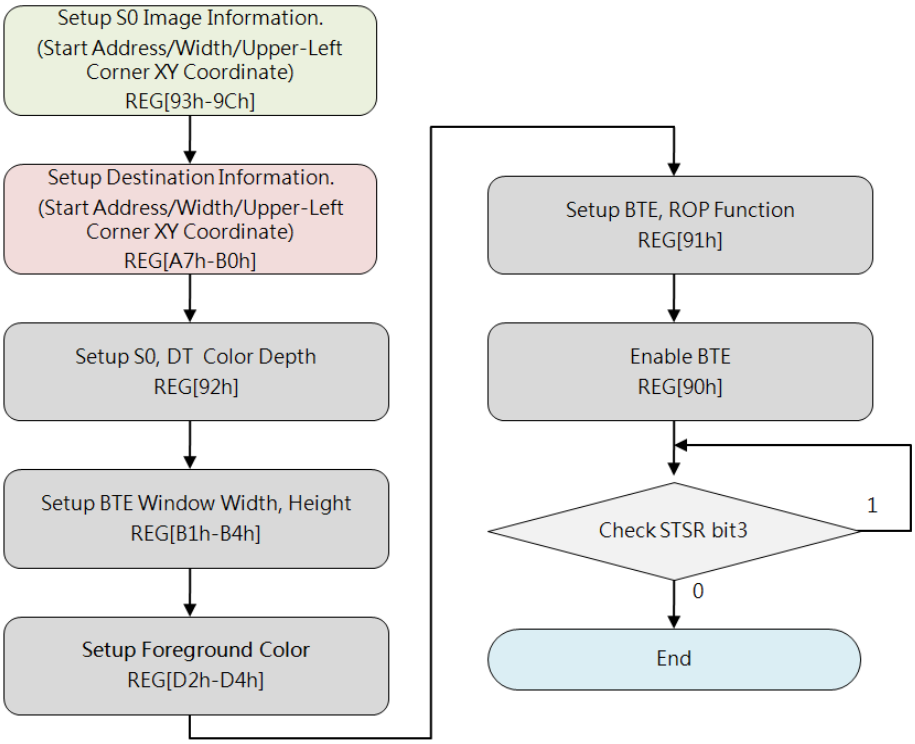


Figure 10-37: Flowchart of Memory Copy with Color Expansion and Chroma Key

10.5.13 Solid Fill

This BTE operation is selected by setting REG[91h] bits[3:0] = 1100b. This operation is to fill a specified rectangle area (BTE Window) of Desitination with a solid color whose data is defined in the Foreground Color Register.

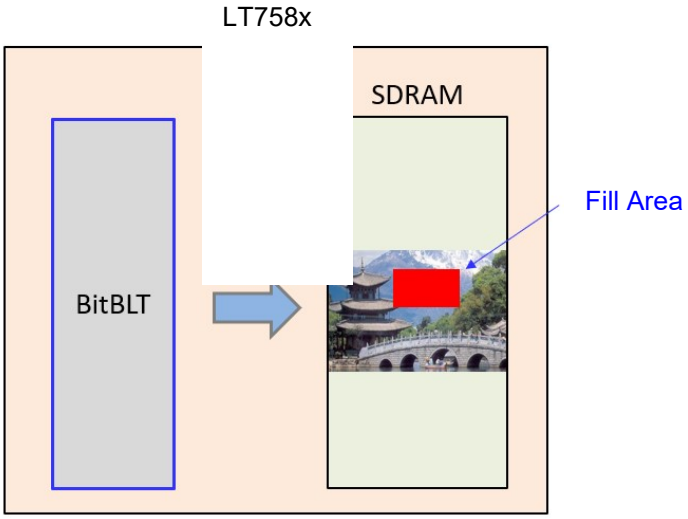


Figure 10-38: Example of Solid Fill

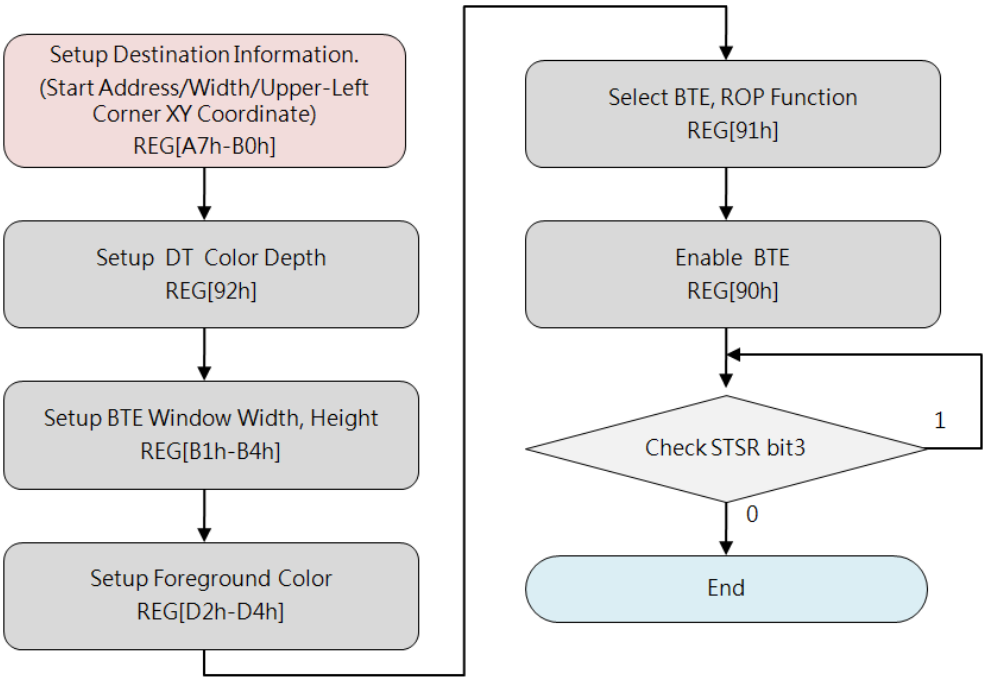


Figure 10-39: Flowchart of Solid Fill

11. Display Text

LT758x supports 2 sources of Text (Characters and Symbols):

- CGROM(Character Graphic ROM): Embedded ISO/IEC 8859 Character Sets
- CGRAM(Character Graphic RAM): User-defined Character Sets

LT758x internal CGROM supports four sets of embedded Characters and Symbols of ASCII code, and internal CGRAM supports users to create their own Characters and Symbols sets when needed. The registers REG[CCh] ~REG[DEh] are for the purpose of User-defined Characters. The Foreground Color registers REG[D2h] ~REG[D4h] and the Background Color registers REG[D5h] ~ REG[D7h] are also employable to define Color for characters from any source.

11.1 Internal CGROM

LT758x has three built-in ASCII font types with different resolutions (character sizes): 8*16, 12*24, 16*32. The MCU can simply writes the font code to display the ASCII code on the LCD panel. The size of the displayed ASCII characters is set by REG[CCh] [5:4]. The ASCII code is corresponding to the standard ISO/IEC 8859-1/2/4/5 coding (Table 11-1 to Table 11-4 below). The type of ISO/IEC 8859 font displayed on the TFT panel is set by REG [CCh] [1:0]. In addition, users can select the color of the text by setting the front-view register REG (D2h to D4h) and the background color register (REG (D5h~D7h). Please refer to the following program flowchart:

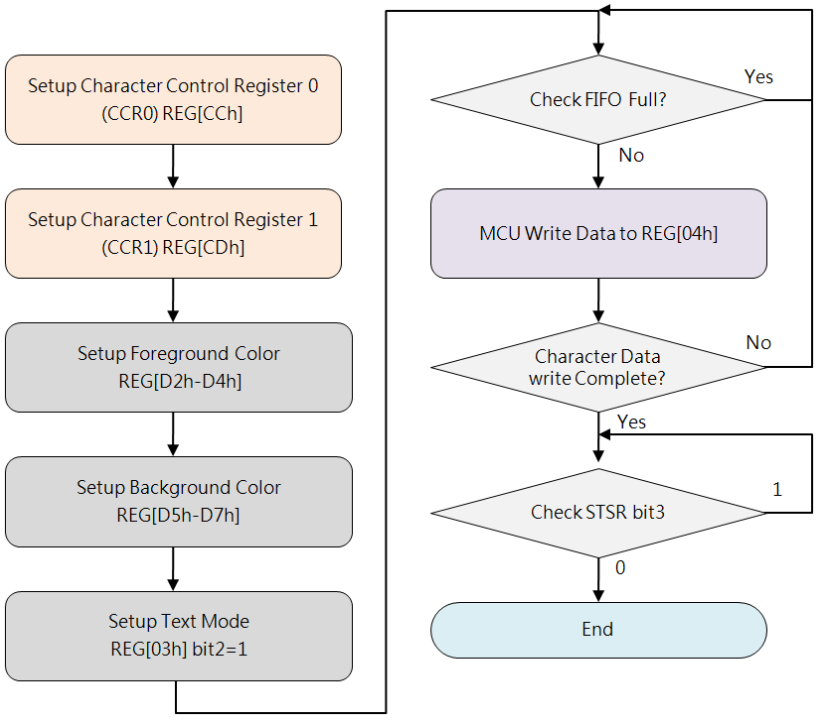


Figure 11-1: Flowchart of CGROM Programming

Table 11-1 shows the standard character encoding of ISO/IEC 8859-1. ISO stands for International Standardization Organization. The ISO/IEC 8859-1, generally known as “Latin-1”. It is the first sets of 8-bit coded character encoding developed by the ISO, and referred to ASCII that consists of 192 characters from the Latin script in the range of 0xA0-0xFF. This character coding is used throughout Western Europe, including Albanian, Afrikaans, Breton, Danish, Faroese, Frisian, Galician, German, Greenlandic, Icelandic, Irish, Italian, Latin, Luxembourgish, Norwegian, Portuguese, Rhaeto-Romanic, Scottish Gaelic, Spanish, Swedish. English letters without accent marks are also available in ISO/IEC 8859-1. In addition, it is also commonly used in many languages outside Europe, such as Swahili, Indonesian, Malaysian and Tagalong.

In Table 11-1, character codes 0x80~0x9F are defined by Microsoft Windows, also called CP1252 (WinLatin1).

Table 11-1：ISO/IEC 8859-1

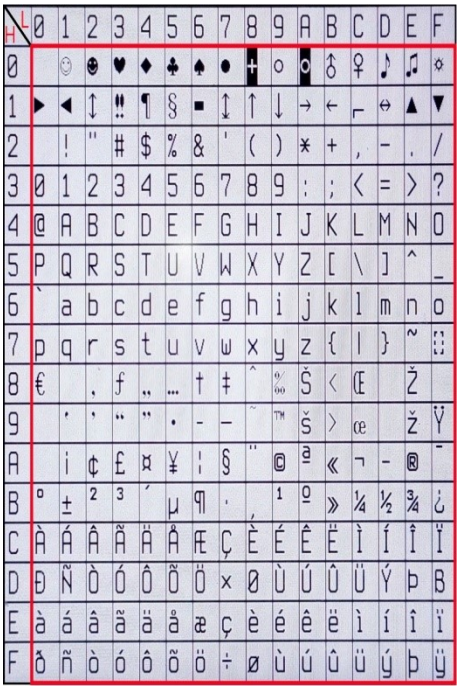


Table 11-2: ISO/IEC 8859-2

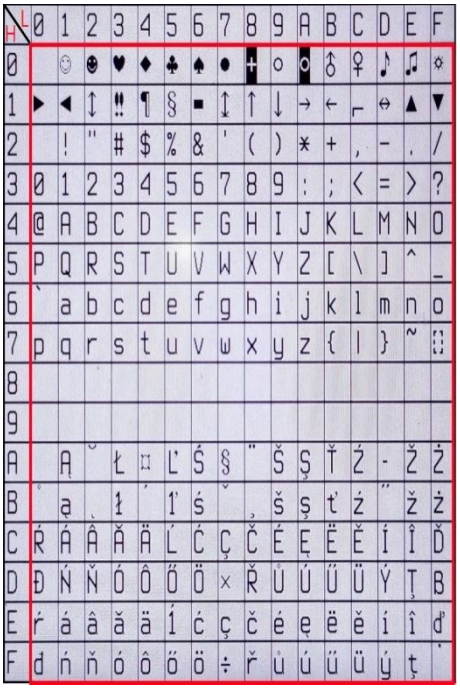


Table 11-2 shows the standard characters of ISO/IEC 8859-2, also known as Latin-2, it is the second sets of the 8-bit character encoding developed by ISO. This code sets can be used in almost any data interchange system to communicate in the following European languages: Croatian, Czech, Hungarian, Polish, Slovak, Slovenian, and Upper Sorbian. The Serbian, English, German, Latin can use ISO/IEC 8859-2 as well. Furthermore it is suitable to represent some western European languages like Finnish (with the exception of å used in Swedish and Finnish)

Table 11-3 shows the standard characters of ISO/IEC 8859-4, also known as Latin-4 or “North European”, it is the fourth sets of the ISO/IEC 8859 8-bit character encoding. It was designed originally to cover Estonian, Greenlandic, Latvian, Lithuanian, and Sami. This character set also supports Danish, English, Finnish, German, Latin, Norwegian, Slovenian, and Swedish.

Table 11-3: ISO/IEC 8859-4

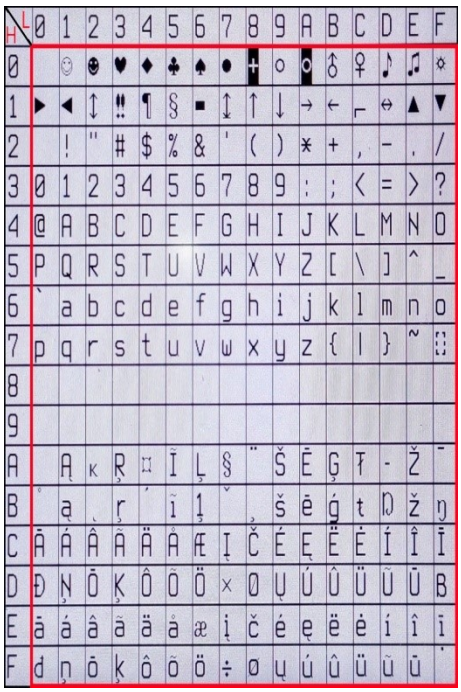


Table 11-4: ISO/IEC 8859-5

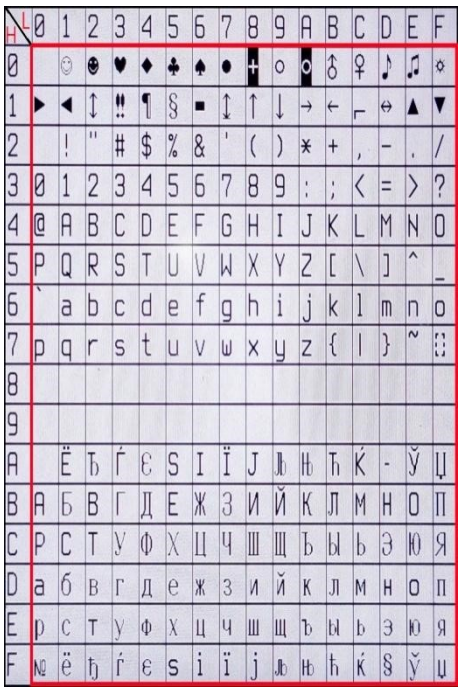


Table 11-4 shows the standard characters of ISO/IEC 8859-5, also known as the fifth sets of the ISO/IEC 8859 8-bit character encoding. It was designed originally to cover Bulgarian ,Belarusian, Russian, Serbian and Macedonian.

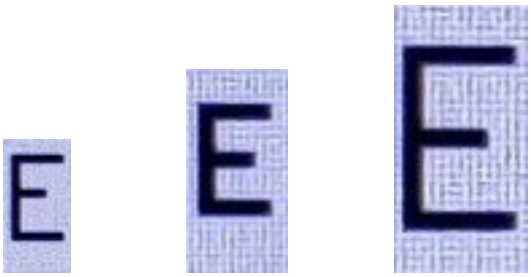


Figure 11-2：Internal ASCII Font for 8*16 / 12*24 / 16*32

11.2 User-defined Character Graphic (UCG)

Users can create and utilize UCG when needed. LT758x supports Half Width size (8*16, 12*24, 16*32 dot-matrix graphic) and Full Width size (16*16, 24*24, 32*32 dot-matrix graphic), and supports up to 32,768 UCGs with half width by encoding from 0000h up to 7FFFh, and up to 32,768 UCGs with full width by encoding from 8000h up to FFFFh.

To display a specific UCG, MCU just needs to write the corresponding CODE (The higher byte first, and then the lower byte) of the UCG to LT758x, LT768x can resolve the CODE (relative ADDRESS of CGRAM) to correspond the absolute ADDRESS of Memory where the UCG data is really saved, then transfer the graphic data to display Memory Buffer. Users can define Foreground Color by setting the REG[D2h] ~REG[D4h] and Background Color by setting the REG[D5h]~REG[D7h] in advance.

Note: The half width characters and the full width characters are using the same memory space.

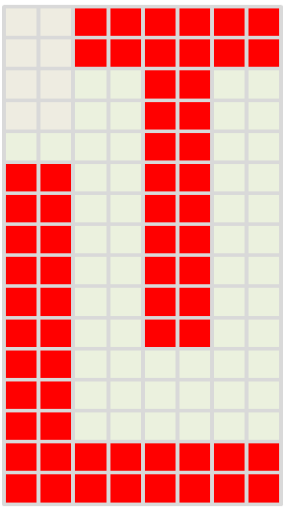
11.2.1 8*16 UCG Data Format

UCG with 8*16 size needs 16 bytes data. For example, CGRAM start address is 1000h and the first UCG encoding is 0000h, the data will be saved in 1000h~100Fh, then the second UCG encoding will be 0001h and its data will be saved in 1010h~101Fh. Below formula shows how the 8*16 UCG address in Memory is calculated, and Table 11-5 explains the data format and data byte sequence:

CGRAM Address = UCG_Start_Address + (UCG_Code & 0x7FFF * 16)

Table 11-5: Data Format and Byte Sequence of 8*16 UCG

UCG Code: 0000h		UCG Code: 0001h	
Address	Data	Address	Data
1000h	Byte0: 3Fh	1010h	Byte0
1001h	Byte1: 3Fh	1011h	Byte1
1002h	Byte2: 0Ch	1012h	Byte2
1003h	Byte3: 0Ch	1013h	Byte3
:	:	:	:
:	:	:	:
:	:	:	:
100Dh	Byte14: C0h	101Dh	Byte14
100Eh	Byte14: FFh	101Eh	Byte14
100Fh	Byte15: FFh	101Fh	Byte15



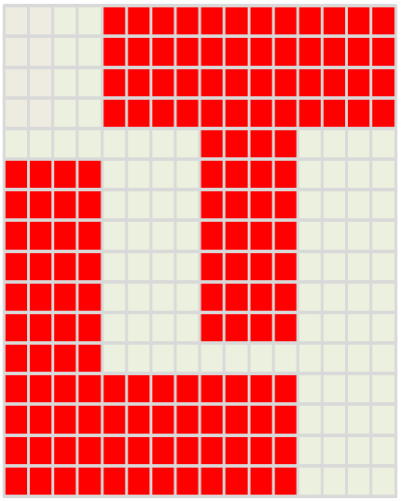
11.2.2 16*16 UCG Data Format

UCG with 16*16 size needs 32 bytes data. For example, CGRAM start address is 1000h and the first UCG encoding is 8000h, the data will be saved in 1000h~101Fh, then the second UCG encoding will be 8001h and its data will be saved in 1020h~103Fh. Below formula shows how the 16*16 UCG address in Memory is calculated, and Table 11-6 explains the data format and data byte sequence:

CGRAM Address = UCG_Start_Address + (UCG_Code * 32)

Table 11-6: Data Format and Byte Sequence of 16*16 UCG

UCG Code: 8000h			
Address	Data	Address	Data
1000h	Byte0: 0Fh	1001h	Byte1: FFh
1002h	Byte2: 0Fh	1003h	Byte3: FFh
1004h	Byte4: 0Fh	1005h	Byte5: FFh
1006h	Byte6: 0Fh	1007h	Byte7: FFh
:	:	:	:
:	:	:	:
:	:	:	:
101Ah	Byte26: FFh	101Bh	Byte27: F0h
101Ch	Byte28: FFh	101Dh	Byte29: F0h
101Eh	Byte30: FFh	101Fh	Byte31: F0h



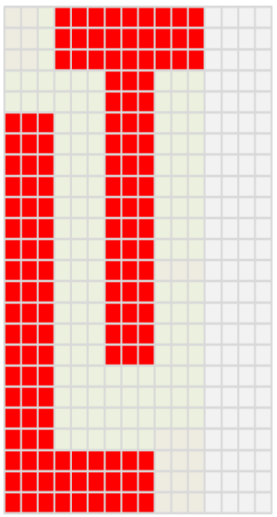
11.2.3 12*24 UCG Data Format

UCG with 12*24 size needs 48 bytes data, note that bit[3:0] of the byte with Odd Sequence Number is ignored. For example, CGRAM start address is 1000h and the first UCG encoding is 0000h, the data will be stored in 1000h~102Fh, then the second UCG encoding will be 0001h and its data will be stored in 1030h~105Fh. Below formula shows how the 12*24 UCG address in Memory is calculated, and Table 11-7 explains the data format and data byte sequence:

CGRAM Address = UCG_Start_Address + (UCG_Code & 0x7FFF * 48)

Table 11-7: Data Format and Byte Sequence of 12*24 UCG

UCG Code: 0000h			
Address	Data	Address	Data
1000h	Byte0: 1Fh	1001h	Byte1: F0h
1002h	Byte2: 1Fh	1003h	Byte3: F0h
1004h	Byte4: 1Fh	1005h	Byte5: F0h
1006h	Byte6: 03h	1007h	Byte7: 80h
:	:	:	:
:	:	:	:
:	:	:	:
102Ah	Byte42: FFh	102Bh	Byte43: 80h
102Ch	Byte44: FFh	102Dh	Byte45: 80h
102Eh	Byte46: FFh	102Fh	Byte47: 80h



11.2.4 24*24 UCG Data Format

UCG with 24*24 size needs 72 bytes data. For example, CGRAM start address is 1000h and the first UCG encoding is 8000h, the data will be stored in 1000h~1047h, then the second UCG encoding will be 8001h and its data will be saved in 1048h~108Fh. Below formula shows how the 24*24 UCG address in Memory is calculated, and Table 11-8 explains the data format and data byte sequence:

CGRAM Address = UCG_Start_Address + (UCG_Code * 72)

Table 11-8: Data Format and Byte Sequence of 24*24 UCG

UCG Code: 8000h					
Address	Data	Address	Data	Address	Data
1000h	Byte0	1001h	Byte1	1002h	Byte2
1003h	Byte3	1004h	Byte4	1005h	Byte5
1006h	Byte6	1007h	Byte7	1008h	Byte8
1009h	Byte9	100Ah	Byte10	100Bh	Byte11
: : :	: : :	: : :	: : :	: : :	: : :
103Fh	Byte63	1040h	Byte64	1041h	Byte65
1042h	Byte66	1043h	Byte67	1044h	Byte68
1045h	Byte69	1046h	Byte70	1047h	Byte71

11.2.5 16*32 UCG Data Format

UCG with 16*32 size needs 64 bytes data. For example, CGRAM start address is 1000h and the first UCG encoding is 0000h, the data will be saved in 1000h~103Fh, then the second UCG encoding will be 0001h and its data will be saved in 1040h~107Fh. Below formula shows how the 16*32 UCG address in Memory is calculated, and Table 11-9 explains the data format and data byte sequence:

CGRAM Address = UCG_Start_Address + (UCG_Code & 0x7FFF * 64)

Table 11-9: Data Format and Byte Sequence of 16*32 UCG

UCG Code: 0000h			
Address	Data	Address	Data
1000h	Byte0	1001h	Byte1
1002h	Byte2	1003h	Byte3
1004h	Byte4	1005h	Byte5
1006h	Byte6	1007h	Byte7
: : :	: : :	: : :	: : :
103Ah	Byte58	103Bh	Byte59
103Ch	Byte60	103Dh	Byte61
103Eh	Byte62	103Fh	Byte63

11.2.6 32*32 UCG Data Format

UCG with 32*32 size needs 128 bytes data. For example, CGRAM start address is 1000h and the first UCG encoding is 8000h, the data will be stored in 1000h~107Fh, then the second UCG encoding will be 8001h and its data will be stored in 1080h~10FFh. Below formula shows how the 32*32 UCG address in Memory is calculated, and Table 11-10 explains the data format and data byte sequence:

CGRAM Address = UCG_Start_Address + (UCG_Code * 128)

Table 11-10: Data Format and Byte Sequence of 32*32 UCG

UCG Code: 8000h							
Address	Data	Address	Data	Address	Data	Address	Data
1000h	Byte0	1001h	Byte1	1002h	Byte2	1003h	Byte3
1004h	Byte4	1005h	Byte5	1006h	Byte6	1007h	Byte7
1008h	Byte8	1009h	Byte9	100Ah	Byte10	100Bh	Byte11
100Ch	Byte12	100Dh	Byte13	100Eh	Byte14	100Fh	Byte15
: : :	: : :	: : :	: : :	: : :	: : :	: : :	: : :
1074h	Byte116	1075h	Byte117	1076h	Byte118	1077h	Byte119
1078h	Byte120	1079h	Byte121	107Ah	Byte122	107Bh	Byte123
107Ch	Byte124	107Dh	Byte125	107Eh	Byte126	107Fh	Byte127

11.2.7 Initialize CGRAM from MCU

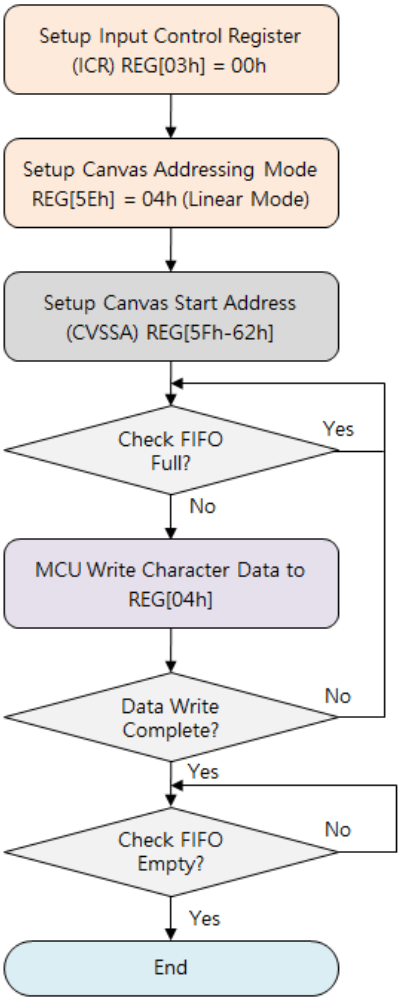


Figure 11-3：Flowchart of CGRAM Initialization from MCU

11.2.8 Initialize CGRAM from Serial Flash

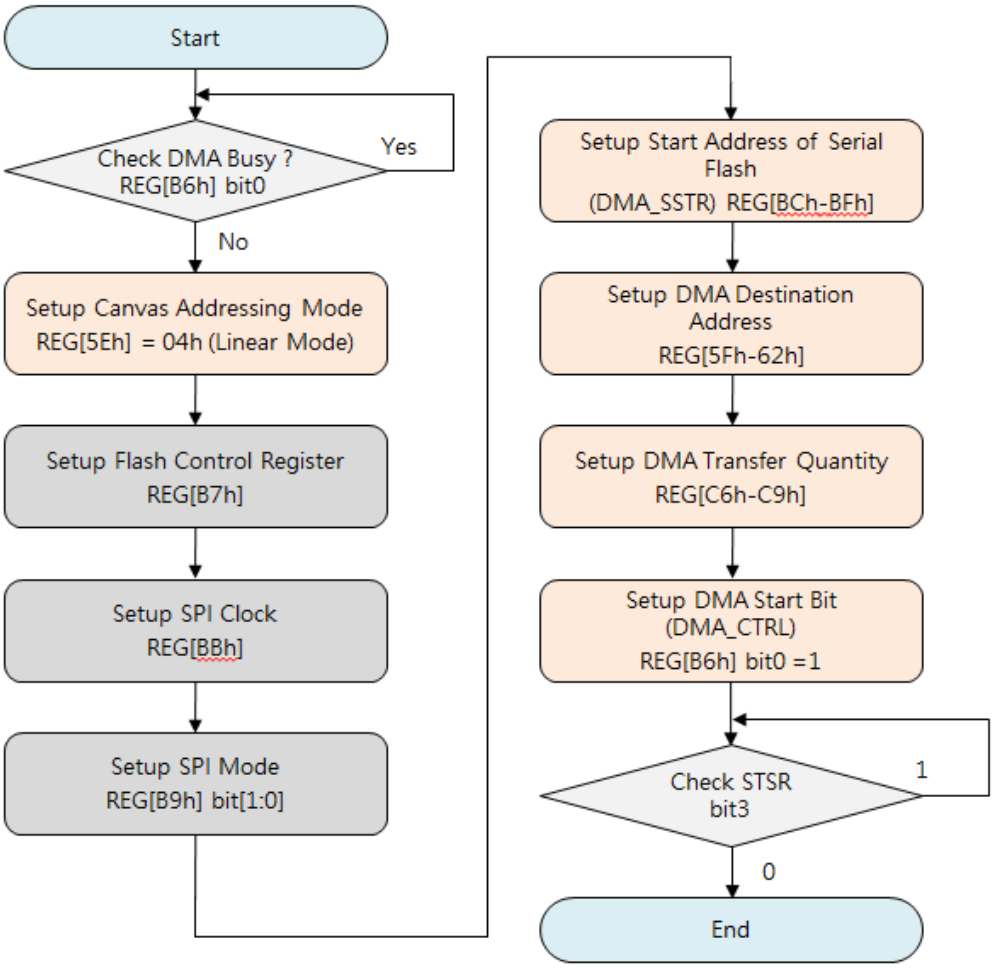


Figure 11-4：Flowchart of CGRAM Initialization from Serial Flash

11.3 Character Rotation by 90 Degree

LT758x supports character rotation by counterclockwise 90 degree. Normal (REG[CDh] bit4=0) text direction is from left to right then from top to bottom. If set REG[CDh] bit4=1, the character will be counterclockwise rotated by 90 degree. Also, the text direction will be changed to from top to bottom then from left to right. However, to get the correct display result, the Display Scan Direction must be changed by setting VDIR Reg[12h] bit3=1 (Please note that Text Cursor, Graphic Cursor, and PIP will be disabled automatically under this setting). Below example shows a set of characters rotated by 90 degree:

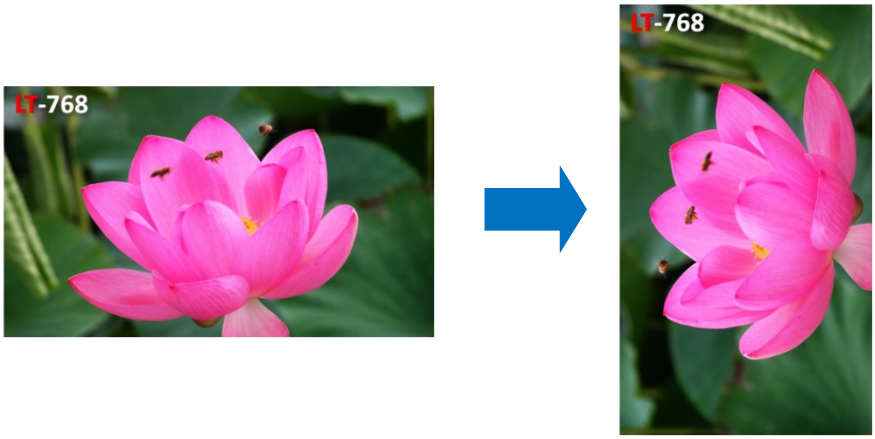


Figure 11-5：Example of Character Rotation

11.4 Font Size Enlargement

LT758x supports linear character size enlargement for Height and/or Width, controlled by REG[CDh] bit[3:0].

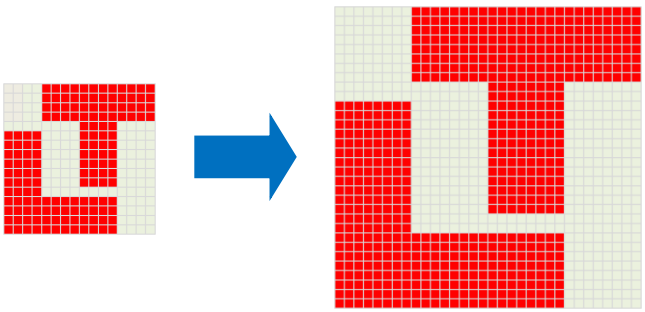


Figure 11-6：Font Size Enlargement

11.5 Background Transparency

LT758x supports character background transparent, controlled by REG[CDh] bit6.

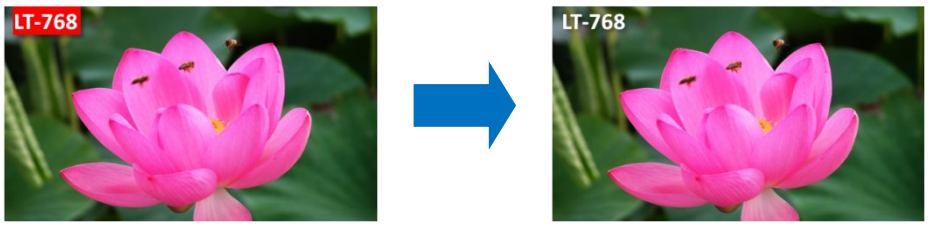


Figure 11-7：Background Transparency

11.6 Automatic Line Feed

LT758x supports sequent text input and display, and can perform automatically Line Feed at active window boundary.

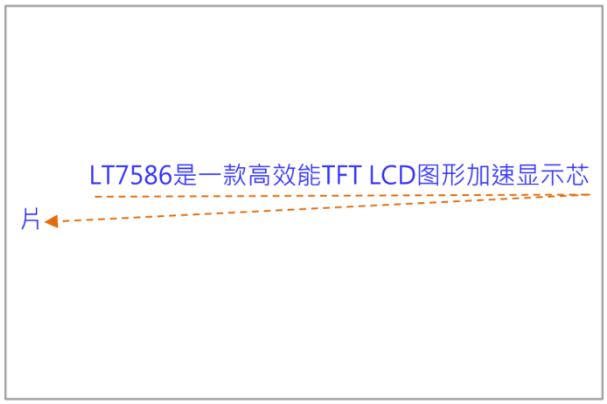


Figure 11-8：Automatic Line Feed

11.7 Character Full-Alignment

LT758x supports character full-alignment function that makes the characters to align each other. Set REG[CDh] bit7=1 to activate this function.

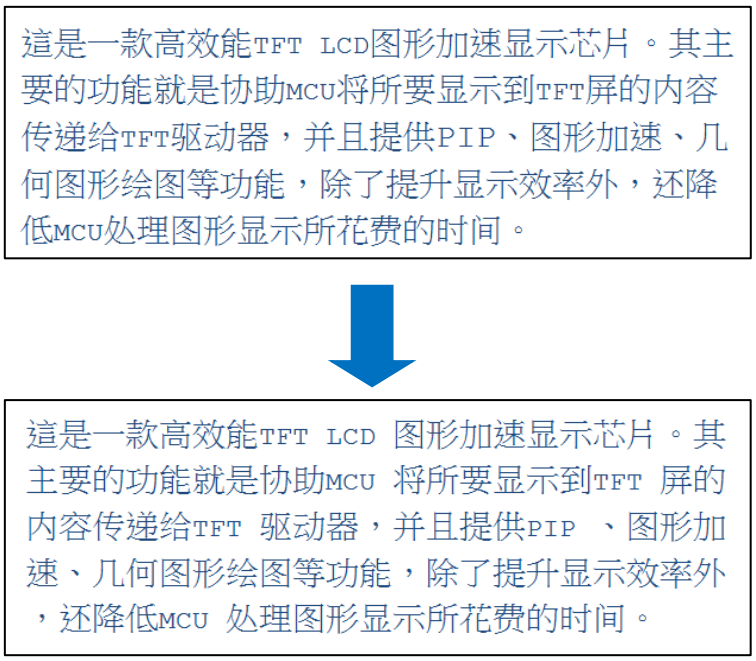


Figure 11-9：Character Full-Alignment

11.8 Cursor

LT758x supports 2 types of Cursor, Graphic Cursor and Text Cursor. The Graphic Cursor is of 32*32 or 64*64 pixel graphic which can be displayed at an user-defined position. The Text Cursor is used to point the text input position. The width and height of a Text Cursor can be setup.

11.8.1 Text Cursor

Text Cursor has Auto-move, Blinking, and Enlargement functions. Once enabled, the Text Cursor appears at the input position, and can automatically move to the next input position after the current input is completed. Auto-move also supports Auto Line Feed, but is dominated by the Active Window. Therefore, Text Cursor must be positioned within the active window and be used in Text Mode. Settings for moving distance and direction are the same as Character input settings.

Note: The display priority is Graphic Cursor > Text Cursor > PIP1 > PIP2 > Main

Table 11-11: Registers Related with Text Cursor

Register Address	Register Name	Description
REG[03h]	ICR	bit2: Graphic/Text Mode Selection (Text Mode Enable)
REG[3Ch]	GTCCR	bit1: Text Cursor Enable
		bit0: Text Cursor Blinking Enable
REG[64h:63h]	F_CURX	X_Position: Text Input X coordinate
REG[66h:65h]	F_CURY	Y_Position: Text Input Y coordinate
REG[D0h]	FLDR	Text Line Gap Setting

■ Text Cursor Blinking

Text Cursor Blinking can be enabled by setting GTCCR (REG[3Ch]) bit[0]=1, and disabled by setting bit[0]=0. Blinking interval can be obtained through below formula:

Blink Time (sec) = BTCR[3Dh] * (1/Frame_Rate)

As the example shown below, the Text Cursor will stay behind the last character:

Figure 11-10：Example of Text Cursor Blinking

■ Text Cursor Height and Width

Text Cursor Height and Width are controlled by CURHS (REG[3Eh]) and CURVS (REG[3Fh]).
In addition, if Character enlargement is enabled, Text Cursor enlargement is also enabled automatically according to the same enlargement settings.

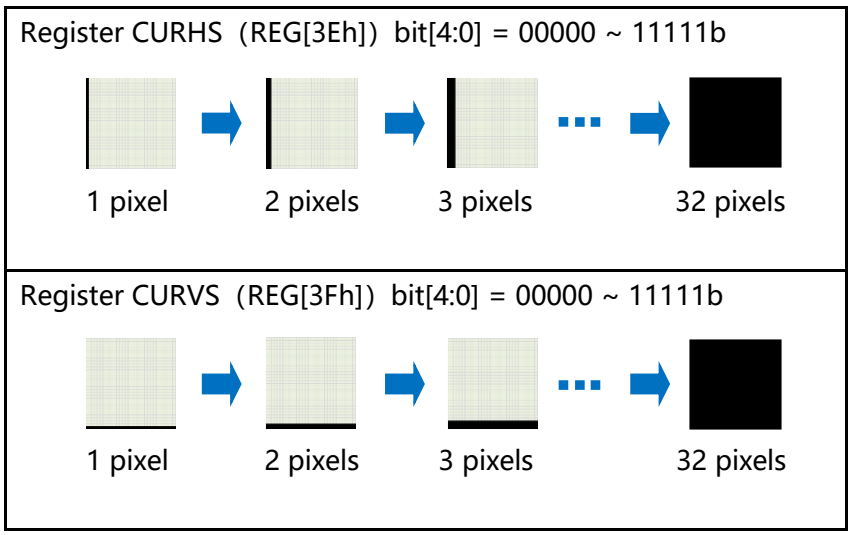


Figure 11-11：Settings for Text Cursor Height and Width

11.8.2 Graphic Cursor

LT758x Graphic Cursor is of 32*32 or 64*64 pixel graphic, and the color of each pixel is represented by 2 bits data. The color data are defined as shown in Table 11-12:

Table 11-12：Graphic Color Definition

Curor Pixel	Pixel Definition
2′ b00	Color-0, defined by GCC0 (REG[44h])
2′ b01	Color-1, defined by GCC1 (REG[45h])
2′ b10	Background Color
2′ b11	Inversed Background Color

It takes 256 bytes (32x32x2/8) to create a 32x32 Graphic Cursor, and 1Kbytes (64x64x2/8) to create a 64x64 Graphic Cursor. LT758x supports 4 sets of 32x32 graphic cursor or 1 set of 64x64 graphic cursor that users can choose from by setting GTCCR REG[3Ch] bit[3:2]. Display position of the graphic cursor is controlled by register GCHP0 (REG[40h]), GCHP1(REG[41h]), GCVP0(REG[42h]) and GCVP1(REG[43h]). Figure 11-12 shows the data format and byte sequence of a 32x32 graphic cursor.

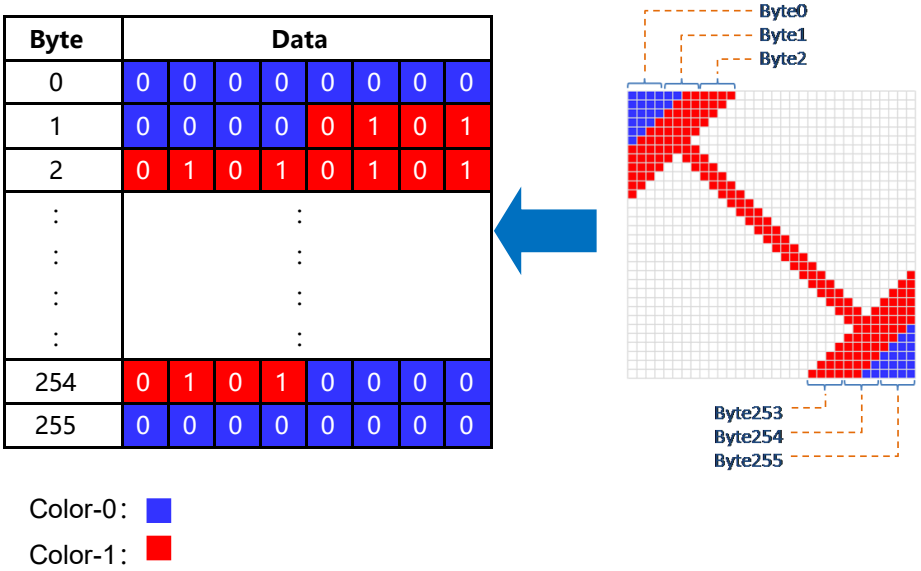


Figure 11-12：Graphic Cursor Data Format

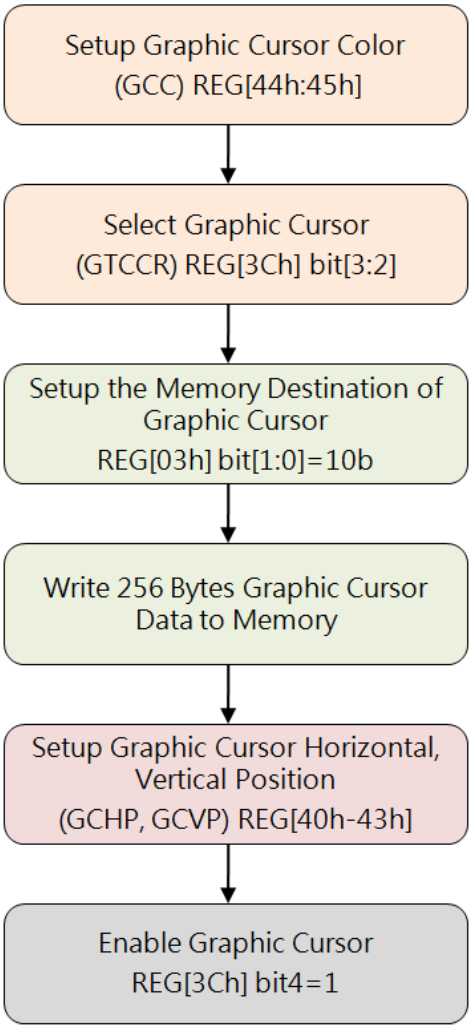


Figure 11-13：Flowchart of Creating a 32x32 Graphic Cursor

Note: The display priority is Graphic Cursor > Text Cursor > PIP1 > PIP2 > Main

One 64x64 Graphic Cursor is composed by 4 sets of 32x32 Graphic Cursors. As shown in Figure 11-14, the arranging sequence is from left to right and then top to bottom.

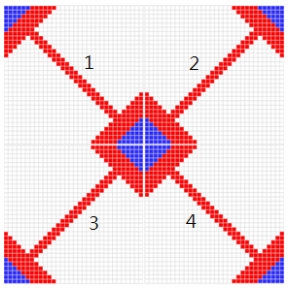


Figure 11-14：64x64 Graphic Cursor Example

12. Pulse Width Modulation (PWM)

LT758x has built-in PWM functions, and provides two PWM signal outputs: PWM0 and PWM1. LT758x also has embedded two 16bits counters, Timer-0 and Timer-1, and its action is related to the output state of the PWM signals. For example, before using the PWM0, the Host must set Timer-0 count registers (TCNTB0, REG[8Ah-8Bh],) and Timer-0 count comparison registers (TCMPB0, REG[88h-89h]). After the PWM function is enabled, the Timer-0 counter will first load the TCNTB0 value and start counting down according to the PWM clock frequency. When the value of the Timer-0 counter equals the value of the TCMPB0 register, PWM will be active. This means if the original state of PWM0 is 0, it will change to 1. The Timer-0 counter will continue to count down, and when Timer-0 equals 0, then an interrupt will be generated. The state of PWM0 will be back to the original 0, and also automatically reload TCNTB0 value. The above procedure represents a complete PWM cycle.

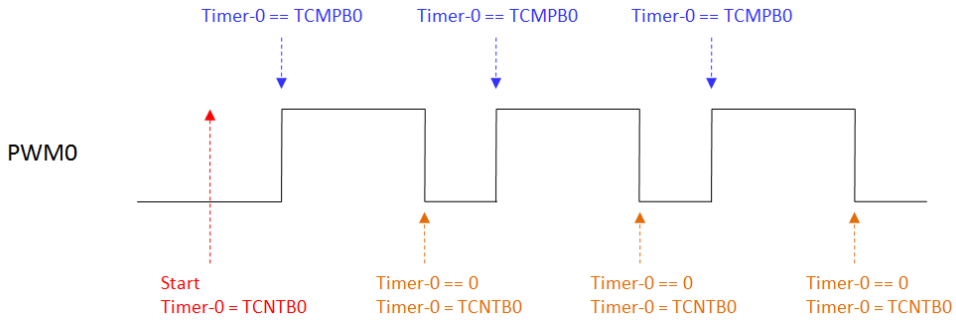


Figure 12-1：PWM Output Wave Form

Based on the above waveform and description, the duty cycle of PWM0 is determined by comparison registers (TCMPB0, REG[88h-89h]). For example, to generate a voltage of approximate DC potential by PWM0, when the PWM0 is initially set to 0, the larger value of the TCMPB0 is set, the higher equivalent voltage on PWM0 will be. On the other hand, when the PWM0 is initially set to 1, the smaller value of the TCMPB0 is set, the higher equivalent voltage on PWM0 will be.

Note: The automatic reload feature of PWM0 (REG[86h] bit1) must be activated, and when Timer-0 equals 0, it will reload the value of the TCNTB0 automatically. Therefore, if the MCU changes TCNTB0 or TCMPB0 value before Timer-0 equals 0, PWM can produce different duty-cycle waveform.

12.1 PWM Clock Source

The PWM's clock source comes from CCLK(System Clock), while the Timer-0 and Timer-1 base frequencies are determined by register PSCLR (REG[84h]):

$$PWM\_CLK = CCLK / (Prescaler + 1)$$

The clock source of the Timer is determined by the register (REG[85h]). The Timer Divisor of each timer provides four options: 1, 1/2, 1/4, 1/8 for the timer's clock. For example, if the REG[85h] Bit[5:4] = 10b, then the Timer-0 count Clock = System_clock/4. Refer to the Register Description chapter for more detail.

12.2 PWM Output

The output of PWM can also be set to a fixed high or low level. For PWM0, turn off the automatic reload feature (REG[86h] bit1 = 0) first, and stop counting down Timer-0 (REG[86h] bit0 = 0), if Timer-0 < TCMP0, then PWM output high. If Timer-0 > TCMP0, then PWM output low (assuming the reverse phase is closed, REG[86h] bit2=0). The output state of the PWM0 can be set to the inverse phase by REG[86h] bit2.

In addition, PWM0 and PWM1 are multiplex output pins that can be used for other purposes, please refer to the following description of Register REG[85h] bit[3:0].

Table 12-1: REG[85h] Description

Bit	Description
3-2	**PWM[1] Function Control** 0xb: PWM[1] output system error flag (Scan FIFO POP error or Memory access out of range) 10b: PWM[1] output PWM timer 1 event or invert of PWM timer 0 11b: PWM[1] output Oscillator Clock
1-0	**PWM[0] Function Control** 0xb: PWM[0] becomes GPIOC[7] 10b: PWM[0] output PWM Timer 0 11b: PWM[0] output Core Clock (CCLK)

PWM0 and PWM1 can also be set to complementary outputs. In this case, the output state of PWM1 follows the setting and control of PWM0, except that it is a PWM0 reverse output state:

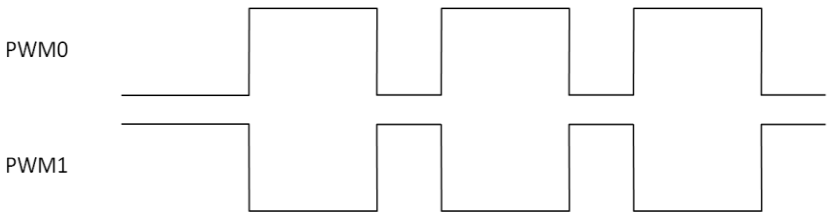


Figure 12-2: The Complementary Outputs of PWM0 and PWM1

In some applications, when using Complementary Outputs of PWM0 and PWM1, changing state of PWM0 and PWM1 at the same time may cause excessive currents. LT758x provides a dead-zone timing control to stagger the output state of PWM0 and PWM1. The dead-zone length of PWM is set by register REG[87h] (DZ_LENGTH):

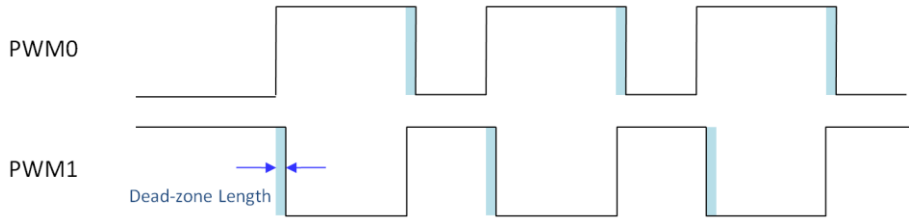


Figure 12-3：Dead-zone of PWM0 and PWM1

13. Serial Bus Master

13.1 SPI Master (SPIM)

When transferring data through SPI master of LT758x, the two serial data lines will be sampled and synchronized with the serial clock signal SFCLK. Master will place the relevant information on the SFDO signal line for the slave device to catch data in the first half of the clock edge. There are four possible protocol modes of CPOL and CPHA that can be choosen by setting the bit[1:0] of Serial Master Control Register [SPIMCR2]. The master and slave device must operate under the same frequency.

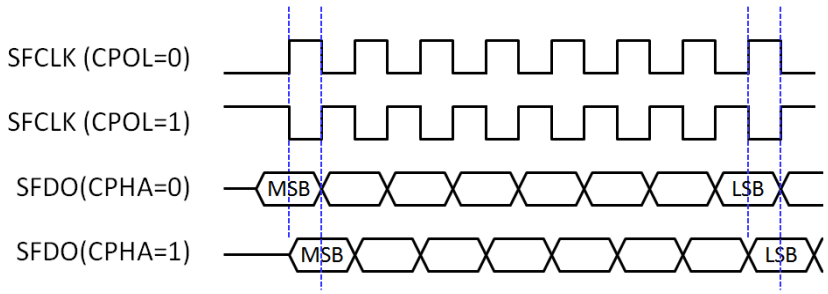


Figure 13-1: Master SPI Data Transmission

■ Transmitting Data Bytes

The SPI transmission can be initialized after the control register is setup. The method to initialize data is to write data to the Register SPIDR (REG[B8h]). The data written to SPDIR is actually written to a FIFO which has 16 bytes depths, also known as “Write-FIFO”. Each write access adds a data byte to the Write-FIFO. When the SS_Active is set to 1 and the FIFO is not empty, LT758x will start to transfer the data that is first written into “Write-FIFO” to the Slave.

■ Receiving Data Bytes

Receiving data is done simultaneously with transmitting data; whenever a data byte is transmitted, a data byte is received. For each byte that needs to be read from a device, a dummy byte has to be written to the Write FIFO. This means using a dummy data write cycle to activate the SPI while receiving the data. Whenever the transfer is done, the received data is written into the "Read-FIFO". The "Read-FIFO" is the counterpart of the Write-FIFO. It is an independent 16 bytes deep FIFO. The FIFO contents can be read by reading from the Register [SPIDR] (REG[B8h]).

■ FIFO Overrun

When FIFO is full, writing a new data into “Write-FIFO” will overwrite the oldest data. When writing data to “Write-FIFO” via [SPIDR] registers causes overflow, it will result in data errors. Then the data transmitted using the SPI interface will not be the first data entered, but the data that finally enter the FIFO. As shown in the example of Figure 13-2, WP is the write pointer, when "Write-FIFO" is full, and then next data written in FIFO will overwrite the first one. RP is the read pointer. If “Read-FIFO” is not performed before “Write-FIFO” is overflow, RP will stay at the first place. Therefore, once the overflow occurs, “Read-FIFO” will get wrong data.

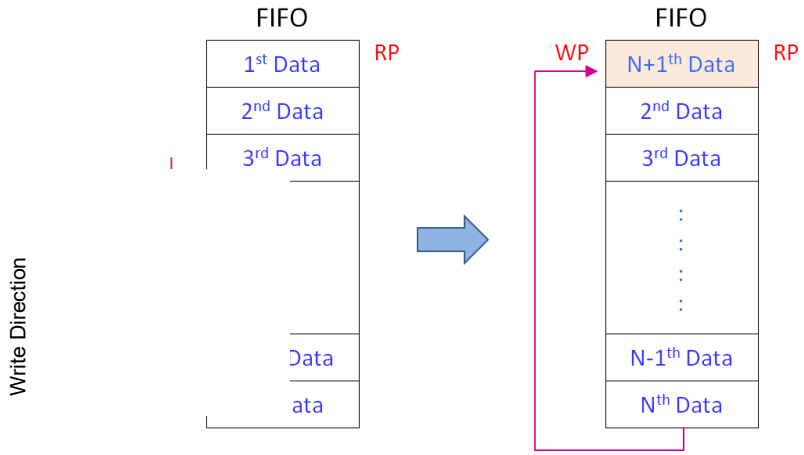


Figure 13-2: FIFO Overrun Example

The only way to recover from this situation is to reset the Write Buffer. Both the Read FIFO and the Write FIFO are reset when the Slave Select signal active [SS_ACTIVE] bit is cleared ('0'). Read FIFO overruns might be less destructive. Especially when the SPI bus is used to transmit data only, and the received data are simply ignored. So the Read FIFO overruns are irrelevant. But, if the SPI bus is used to transmit and receive data, it is important to keep the Read FIFO aligned. The easiest way to do this is to perform a number of dummy reads. The number of dummy reads should be equal to the amount of bytes transmitted modulo 16.

$$N_{dummy\_reads} = N_{transmitted\_bytes} \bmod 16$$

Note: If the "Read-FIFO" is not empty, storing 16 of data is bound to cause overwritten. Therefore, before receiving every 16 bytes of data, the host must confirm if "Read-FIFO" is empty or not.

Reference Code for SPI Master Loop Test (Connect MOSI to MISO)

```
REG_WR ('hBB, 8'h1f) ;              //Divisor, configure SPI clock frequency
REG_WR ('hB9, 8'b0001_1111) ;       // {1'b0, mask, SS#_sel, ss_active, ovfirqen, emtirqen,
                                    // cpol, cpha}, SS# Low
REG_WR ('hB8, 8'h55) ;              // TX
REG_WR ('hB8, 8'haa) ;              // TX
REG_WR ('hB8, 8'h87) ;              // TX
REG_WR ('hB8, 8'h78) ;              // TX
wait (INT#) ;
REG_RD ('hBA, acc) ;
while (acc != 8'h84) begin
      $display ("wait for FIFO empty ...") ;
      REG_RD ('hBA, acc) ;
end
REG_WR ('hBA, 8'h04) ;              // clear interrupt flag
REG_RD ('hB8, 8'h55) ;              // RX
REG_RD ('hB8, 8'haa) ;              // RX
REG_RD ('hB8, 8'h87) ;              // RX
REG_RD ('hB8, 8'h78) ;              // RX
REG_WR ('hB9, 8'b0000_1111) ;       // {1'b0, mask, SS#_sel, ss_active, ovfirqen, emtirqen,
                                    // cpol, cpha}, SS# Hign.
```


13.2 Serial Flash Controller

LT758x has a built-in SPI Master interface for external Serial Flash, and supports 4-BUS (Normal Read), 5-BUS (FAST Read), Dual mode 0, Dual mode1, and Mode 0/Mode 3. Serial Flash functions can be used for Font mode and DMA mode. Font mode means that the external Serial Flash is treated as a source of character bitmap. DMA mode means that the external Serial Flash is treated as the data source of DMA (Direct Memory Access). Host can speed up the data transmission to the display memory, and does not need to intervene in DMA mode. From the hardware point of view, LT758x supports 2-Wire-SPI or 4-Wire QSPI interface. See the application diagrams shown below:

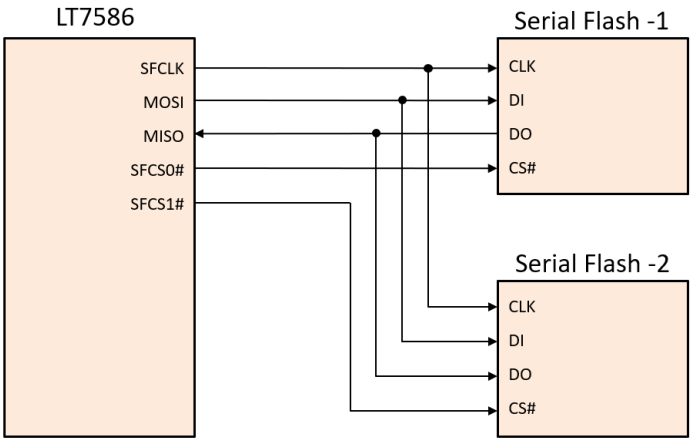


Figure 13-3: LT758x SPI Flash Application Diagram

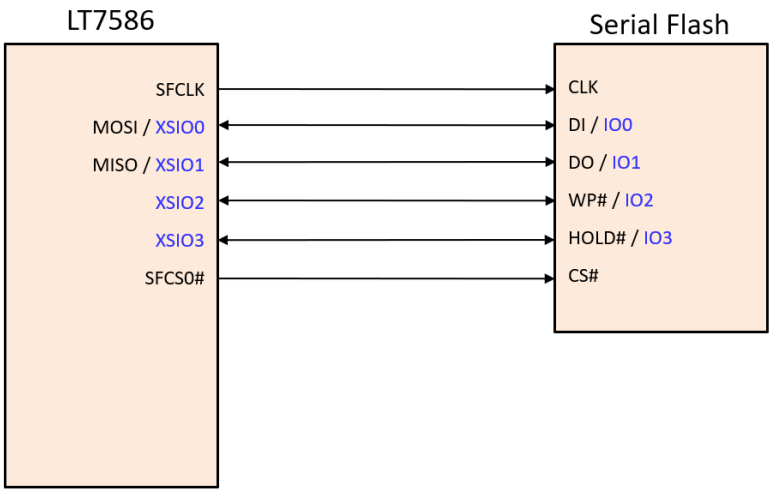


Figure 13-4: LT758x 4-Wire QSPI Flash Application Diagram

13.2.1 By-Pass Mode

By-Pass Mode is only available when the MCU connects to LT758x through 4-Wire SPI (i.e. PSM[2:0] = 101b). When the MCU connects to LT758x through 4-Wire SPI, if the DB[3] pin of LT758x is set to High, the MCU can then directly access the SPI Flash connected to the LT758x. Users may choose the SPI Flash (SFCS[0]# or SFCS[1]#) to be accessed by setting REG[B9h] bit5.

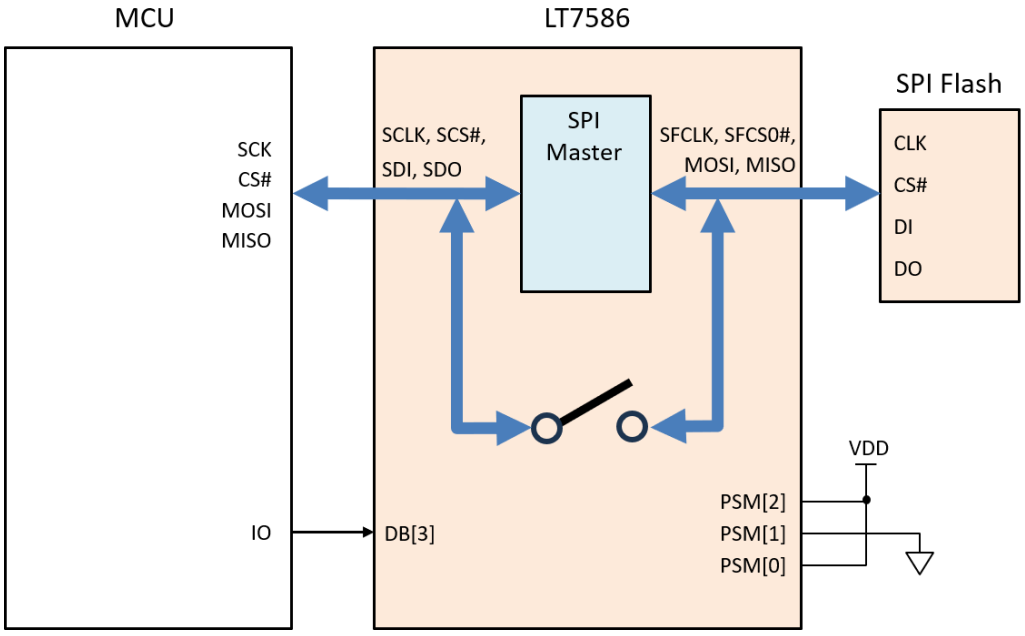


Figure 13-5: LT758x By-Pass Mode

The followings code shows an example of how MCU may read the Flash ID by applying the By-Pass Mode:

```
unsigned char MFID, KGD, EID[6];        // Flash ID related variables
Select_nSS_drive_on_xnsfcs0();          // Set REG[B9h] bit5 to choose the targetting SPI Flash
GPIO_SetBits(GPIOD, GPIO_Pin_1);        // Set DB[3] pin to high => Activate By-Pass Mode

SS_RESET;                               // Set CS pin low to select the SPI Flash
SPI2_ReadWriteByte(ReadDeviceID);       // Write [Read Flash Device ID] command
SPI2_ReadWriteByte(0x00);
SPI2_ReadWriteByte(0x00);
SPI2_ReadWriteByte(0x00);

MFID = SPI2_ReadWriteByte(0xFF);        // Read MFID
KGD = SPI2_ReadWriteByte(0xFF);         // Read KGD
for(int i=0; i<6; i++)
   EID[i] = SPI2_ReadWriteByte(0xFF);   // Read EID

SS_SET;                                 // Set CS pin high to deselect the SPI Flash

GPIO_ResetBits(GPIOD, GPIO_Pin_1);      // Reset DB[3] pin to low => deactivate By-Pass Mode
```


Serial NOR Flash

About Serial Flash read command and protocol setting, please refer to the table shown below:

Table 13-1: Read Command and Protocol of SPI Flash

REG [B7h] Bit[3:0]	Read Command Code & Behavior Selection
0000b	**1x Read Command 03h/13h** Normal read command. The command and address are output through MOSI, and the data are received from MISO. No dummy cycles needed between the address and data. Applicable for NOR/NAND Flash.
0100b	**1x Read Command 0Bh/0Ch** Faster read command. The command and address are output through MOSI, and the data are received from MISO. LT758x will insert 8 dummy cycles between the address and data. Applicable for NOR/NAND Flash.
1000b	**1x Read Command 1Bh** Fastest read command. The command and address are output through MOSI, and the data are received from MISO. LT758x will insert 16 dummy cycles between the address and data. Applicable for NOR Flash.
0010b	**2x Read Command 3Bh/3Ch** Fast Read Dual Output command. The command and address are output through MOSI, and the data are received from MOSI and MISO. LT758x will insert 8 dummy cycles (Dual Mode 0) between the address and data. Applicable for NOR/NAND Flash.
0011b	**2x Read Command BBh/BCh** Fast Read Dual I/O command. The command are output from MOSI. The output address and received data are from MOSI and MISO. LT758x will insert 4 dummy cycles (Dual Mode 1) between the address and data. Applicable for NOR/NAND Flash.
0110b	**4x Read Command 0Bh** Faster Read **in QSPI Mode** command.The command, address and data are all transmitted through MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 2 dummy cycles between the address and data. Applicable for NOR Flash. **Note:** Before using Faster Read in QSPI mode command, the “Enable QSPI Mode command” (38h) must be input through SPIM.
0111b	**4x Read Command 6Bh/6Ch** Fast Read Quad Output command. The command and address are output from MOSI, and the data are received from MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 8 dummy cycles (Quad Mode 0) between the address and data. Applicable for NOR/NAND Flash.

REG [B7h] Bit[3:0]	Read Command Code & Behavior Selection
1001b	**4x Read Command EBh/ECh** Fast Read Quad I/O command. The command is output from MOSI. The address and data are received from MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 4 dummy cycles (Quad Mode 1) between the address and data. The number of dummy cycles on the Flash side should be set accordingly by the Flash command, “Set Read Parameters” (C0h). Applicable for NOR/NAND Flash.
1010b	**4x Read Command EBh/ECh** Fast Read Quad I/O command. The command is output from MOSI. The address and data are received from MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 6 dummy cycles (Quad Mode 1) between the address and data. The number of dummy cycles on the Flash side should be set accordingly by the Flash command, “Set Read Parameters” (C0h). Applicable for NOR/NAND Flash.
1011b	**4x Read Command EBh** Fast Read Quad I/O **in QSPI Mode** command. The command, address, and data are transmitted through MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 6 dummy cycles (Quad Mode 2) between the address and data. Applicable for NOR Flash. **Note:** Before using Faster Read in QSPI mode command, the “Enable QSPI Mode command” (38h) must be input through SPIM. In addition, the number of dummy cycles on the Flash side should be set accordingly by the Flash command, “Set Read Parameters” (C0h).
1110b	**4x Read Command EBh** Fast Read Quad I/O **in QSPI Mode** command. The command, address, and data are transmitted through MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 8 dummy cycles (Quad Mode 2) between the address and data. Applicable for NOR Flash.。 **Note:** Before using Faster Read in QSPI mode command, the “Enable QSPI Mode command” (38h) must be input through SPIM. In addition, the number of dummy cycles on the Flash side should be set accordingly by the Flash command, “Set Read Parameters” (C0h).

Note 1: For NOR Flash, LT758x will adapt the proper “Read Command” based on the addressing mode (24bits or 32bits). For example, when REG[B7h] bit[3:0] is set as 1010b, then the Read Command will be EBh for 24bits addressing mode, and ECh for 32bits addressing mode.

Note 2: Not all of the SPI Flash chips support the above commands. Users should check the Flash spec for the proper commands.

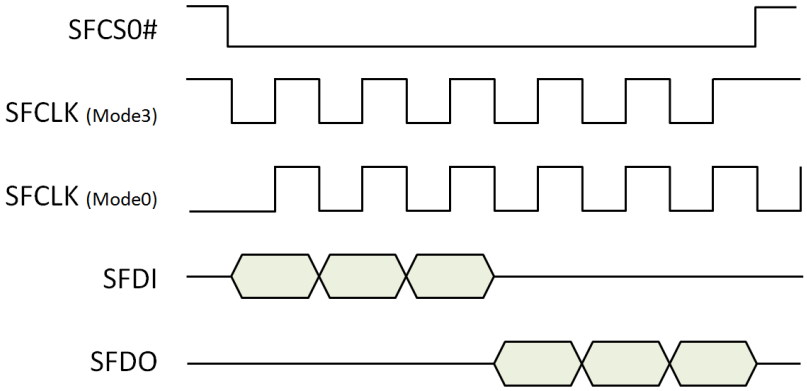


Figure 13-6: Mode 0 and Mode 3 Protocol

Figure 13-7 is a Timing diagram of Read Commands for SPI Flash Memory. If REG[B7h] bit5=0, the address line is 24bits and requires 24 clocks. If REG[B7h] bit5=1, the address line is 32bits and requires 32 clocks.

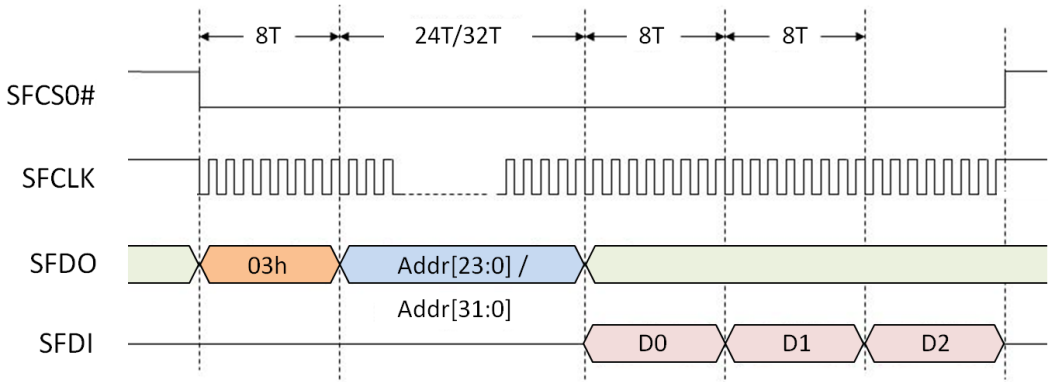


Figure 13-7: SPI Flash – Normal Read Command

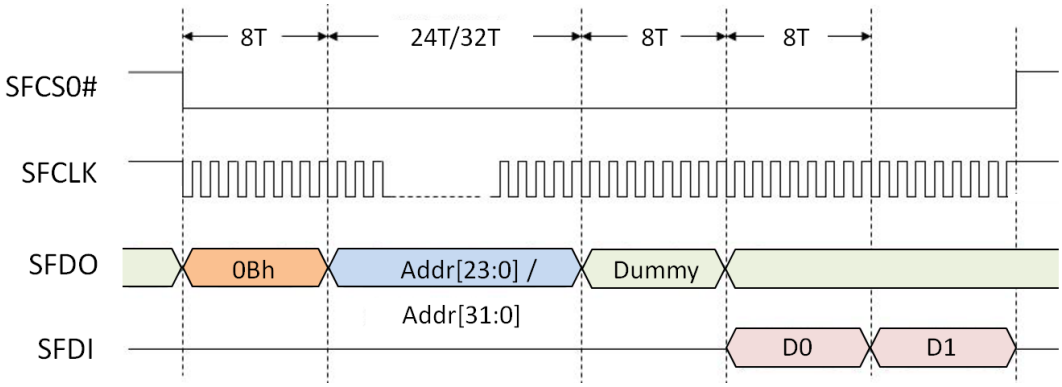


Figure 13-8: SPI Flash – Faster Read Command

Figure 13-9 shows that flash to LT758x data input is as interleaved input, and the data input appears on MISO and MOSI pins. If REG[B7h] bit5=0, the address line is 24bits and requires 24 clocks. If REG[B7h] bit5=1, the address line is 32bits and requires 32 clocks.

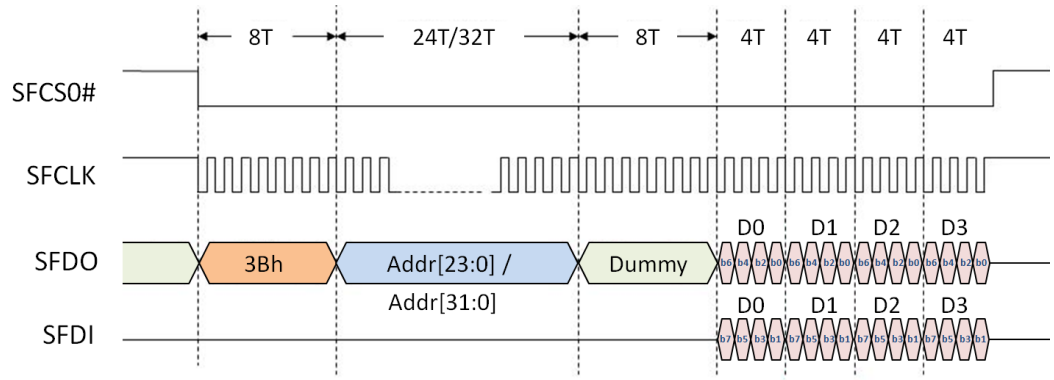


Figure 13-9: SPI Flash Read Command (Data are interleaved)

Figure 13-10 shows that LT758x address output and data input are interleaved. The data and address are interleaved on MISO and MOSI pins. If REG[B7h] bit5=0, it means the address line is 24bits. However, since the transmission is interleaved, only 12 clocks are required. If REG[B7h] bit5=1, it means the address line is 32bits, and the transimmsion only requires 16 clocks.

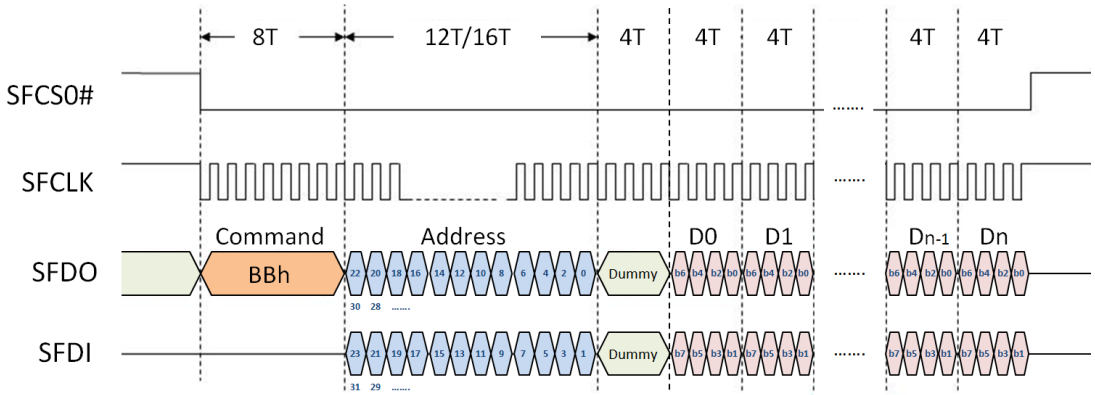


Figure 13-10: SPI Flash Read Command (Data and Address are interleaved)

13.2.2 External Serial Flash

External Flash Memory can be used as a source of the image data. In Graphics Mode, Host can use DMA (Direct Memory access) mode to access the data stored in the external Serial Flash. It means the Flash Memory can be used as the source of DMA, and developer can store a large amount of display data in advance. Therefore, Host does not need to take a lot of time to transfer large and commonly used graphics data to LT758x's Display RAM.

The data format of external Serial Flash must be consistent with the format of the Display RAM. The graphic data format for Flash Memory are as following:

Table 13-2：8bpp Image Data Format of Serial Flash (R:G:B=3:3:2)

Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Addr	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0001h	R_1^7	R_1^6	R_1^5	G_1^7	G_1^6	G_1^5	B_1^7	B_1^6	0000h	R_0^7	R_0^6	R_0^5	G_0^7	G_0^6	G_0^5	B_0^7	B_0^6
0003h	R_3^7	R_3^6	R_3^5	G_3^7	G_3^6	G_3^5	B_3^7	B_3^6	0002h	R_2^7	R_2^6	R_2^5	G_2^7	G_2^6	G_2^5	B_2^7	B_2^6
0005h	R_5^7	R_5^6	R_5^5	G_5^7	G_5^6	G_5^5	B_5^7	B_5^6	0004h	R_4^7	R_4^6	R_4^5	G_4^7	G_4^6	G_4^5	B_4^7	B_4^6
0007h	R_7^7	R_7^6	R_7^5	G_7^7	G_7^6	G_7^5	B_7^7	B_7^6	0006h	R_6^7	R_6^6	R_6^5	G_6^7	G_6^6	G_6^5	B_6^7	B_6^6
0009h	R_9^7	R_9^6	R_9^5	G_9^7	G_9^6	G_9^5	B_9^7	B_9^6	0008h	R_8^7	R_8^6	R_8^5	G_8^7	G_8^6	G_8^5	B_8^7	B_8^6
000Bh	R_{11}^7	R_{11}^6	R_{11}^5	G_{11}^7	G_{11}^6	G_{11}^5	B_{11}^7	B_{11}^6	000Ah	R_{10}^7	R_{10}^6	R_{10}^5	G_{10}^7	G_{10}^6	G_{10}^5	B_{10}^7	B_{10}^6

Table 13-3：16bpp Image Data Formate of Serial Flash (R:G:B=5:6:5)

Order	Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Addr	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	0001h	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	G_0^7	G_0^6	G_0^5	0000h	G_0^4	G_0^3	G_0^2	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3
2	0003h	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	G_1^7	G_1^6	G_1^5	0002h	G_1^4	G_1^3	G_1^2	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3
3	0005h	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	G_2^7	G_2^6	G_2^5	0004h	G_2^4	G_2^3	G_2^2	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3
4	0007h	R_3^7	R_3^6	R_3^5	R_3^4	R_3^3	G_3^7	G_3^6	G_3^5	0006h	G_3^4	G_3^3	G_3^2	B_3^7	B_3^6	B_3^5	B_3^4	B_3^3
5	0009h	R_4^7	R_4^6	R_4^5	R_4^4	R_4^3	G_4^7	G_4^6	G_4^5	0008h	G_4^4	G_4^3	G_4^2	B_4^7	B_4^6	B_4^5	B_4^4	B_4^3
6	000Bh	R_5^7	R_5^6	R_5^5	R_5^4	R_5^3	G_5^7	G_5^6	G_5^5	000Ah	G_5^4	G_5^3	G_5^2	B_5^7	B_5^6	B_5^5	B_5^4	B_5^3

Table 13-4：24bpp Image Data Format of Serial Flash (R:G:B=8:8:8)

Order	Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Addr	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	0001h	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0	0000h	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
2	0003h	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0	0002h	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
3	0005h	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0	0004h	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0
4	0007h	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	G_2^1	G_2^0	0006h	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3	B_2^2	B_2^1	B_2^0
5	0009h	B_3^7	B_3^6	B_3^5	B_3^4	B_3^3	B_3^2	B_3^1	B_3^0	0008h	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	R_2^2	R_2^1	R_2^0
6	000Bh	R_3^7	R_3^6	R_3^5	R_3^4	R_3^3	R_3^2	R_3^1	R_3^0	000Ah	G_3^7	G_3^6	G_3^5	G_3^4	G_3^3	G_3^2	G_3^1	G_3^0

Table 13-5：32bpp Image Data Format of Serial Flash (α:R:G:B=8:8:8:8)

Order	Addr	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Addr	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	0001h	G_0^7	G_0^6	G_0^5	G_0^4	G_0^3	G_0^2	G_0^1	G_0^0	0000h	B_0^7	B_0^6	B_0^5	B_0^4	B_0^3	B_0^2	B_0^1	B_0^0
2	0003h	α_0^7	α_0^6	α_0^5	α_0^4	α_0^3	α_0^2	α_0^1	α_0^0	0002h	R_0^7	R_0^6	R_0^5	R_0^4	R_0^3	R_0^2	R_0^1	R_0^0
3	0005h	G_1^7	G_1^6	G_1^5	G_1^4	G_1^3	G_1^2	G_1^1	G_1^0	0004h	B_1^7	B_1^6	B_1^5	B_1^4	B_1^3	B_1^2	B_1^1	B_1^0
4	0007h	α_1^7	α_1^6	α_1^5	α_1^4	α_1^3	α_1^2	α_1^1	α_1^0	0006h	R_1^7	R_1^6	R_1^5	R_1^4	R_1^3	R_1^2	R_1^1	R_1^0
5	0009h	G_2^7	G_2^6	G_2^5	G_2^4	G_2^3	G_2^2	G_2^1	G_2^0	0008h	B_2^7	B_2^6	B_2^5	B_2^4	B_2^3	B_2^2	B_2^1	B_2^0
6	000Bh	α_2^7	α_2^6	α_2^5	α_2^4	α_2^3	α_2^2	α_2^1	α_2^0	000Ah	R_2^7	R_2^6	R_2^5	R_2^4	R_2^3	R_2^2	R_2^1	R_2^0

DMA function provides a faster method for users to update/transfer mass data to display memory. The only source of DMA function in LT758x is the external Serial Flash. There are two kinds of data type defined for the DMA. One is Linear Mode and the other is Block Mode. It provides a flexible selection for user applications. The destination of DMA function is dominated by the active window in display memory. When DMA function is active, the specific data from Serial Flash/ROM will be transferred one by one to Display Memory by LT758x automatically. After the DMA function is completed, LT758x will generate an interrupt to notify the Host. Please refer to the following sections for more information.

13.2.3 DMA in Linear Mode – External Serial Flash

The DMA Linear Mode is used to send CGRAM data to Display RAM. The color depth of the Active Window must be set as 8bpp.

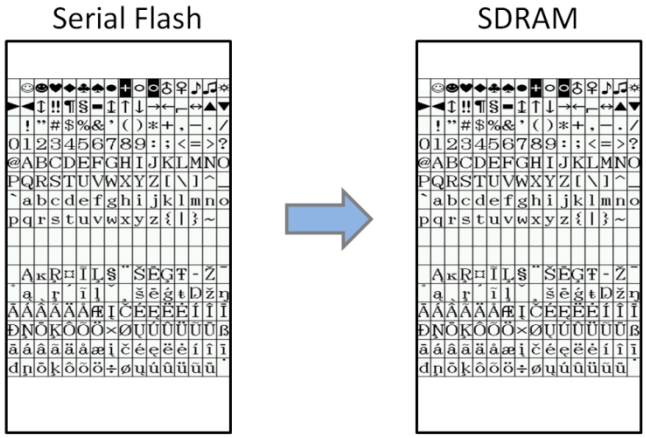


Figure 13-11: DMA in Linear Mode – External Serial Flash

13.2.4 DAM in Block Mode – External Serial Flash

DMA access in Block Mode is mainly used to transfer graphic data. The data is processed by pixel unit. Refer to the below figure and flowchart:

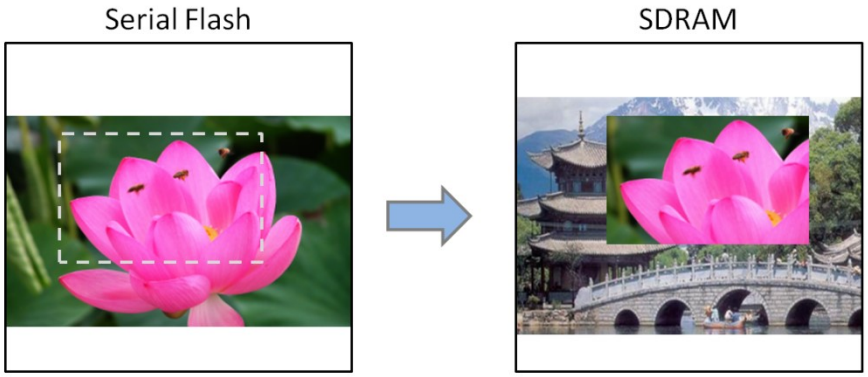


Figure 13-12: DMA in Block Mode – External Serial Flash

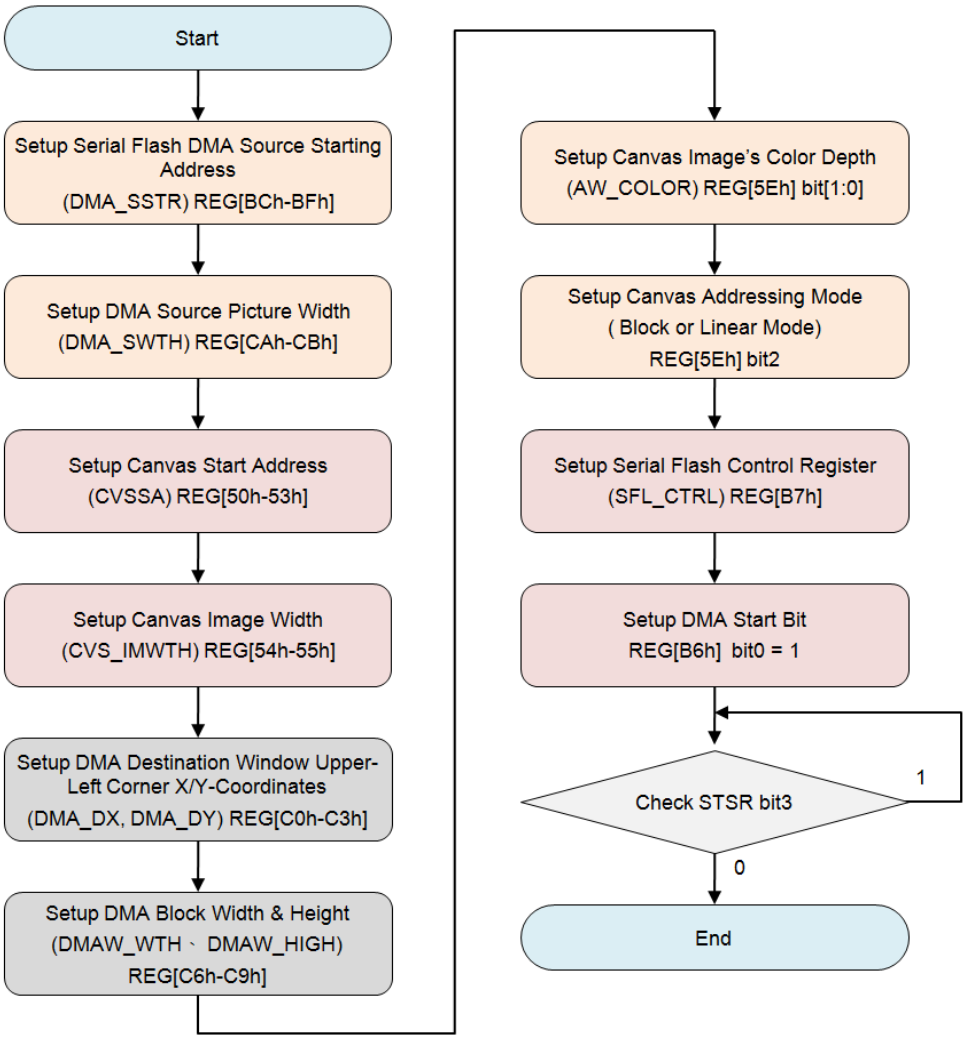


Figure 13-13：DMA Data Transfer Flowchart (Polling Mode)

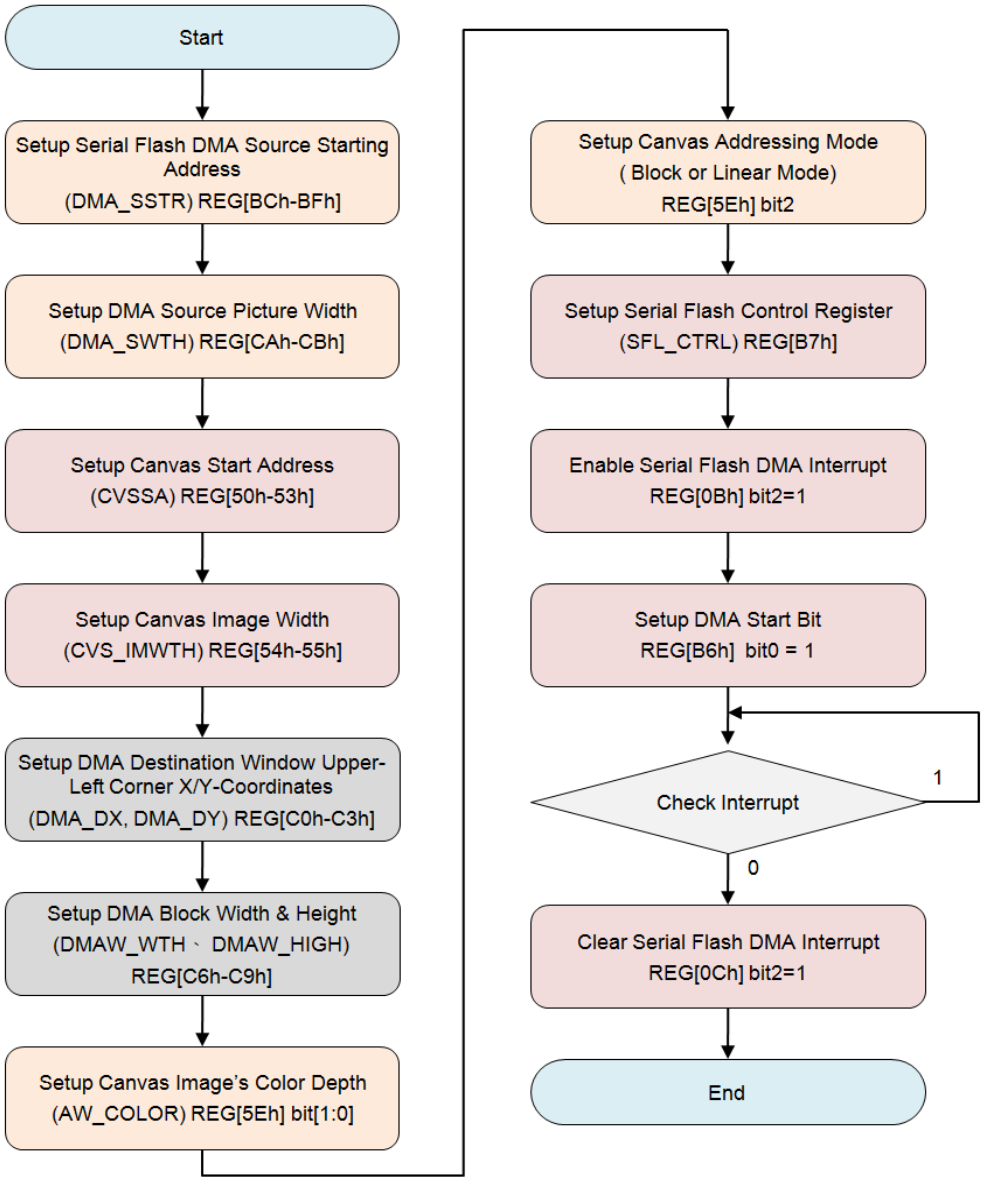


Figure 13-14: DMA Data Transfer Flowchart (Interrupt Mode)

Note: REG[B7h] bit4 is used to define the meaning of the data written to REG[C0h ~ C3h]:

REG[B7h] bit4=0: The data written to REG[C0h~C3h] is the left-top coordinate of the Destination window.

REG[B7h] bit4=1: The data written to REG[C0h~C3h] is the left-top coordinate of the Source window.

Destination: LT758x Display RAM

Source: SPI Flash

13.3 I2C Master

I2C Master is a two-wires, bi-directional serial bus that provides a simple and efficient method of data exchange between devices.

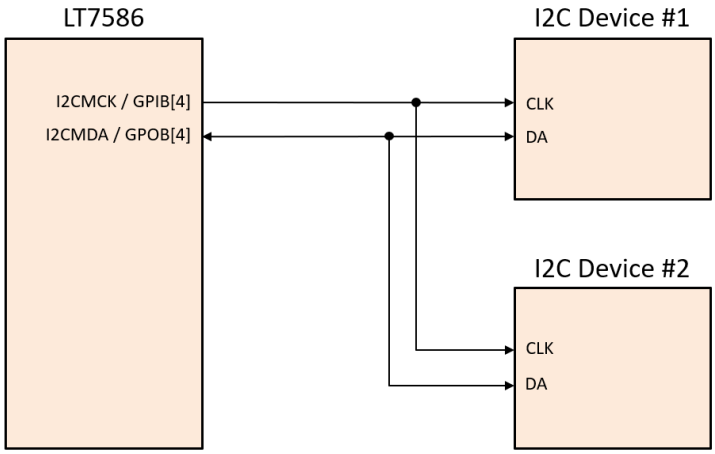


Figure 13-15: LT758x I2C Application Diagram

LT758x supports I2C bus of 100Kbps and 400Kbps transfer rates. The formula of I2C Master clock(I2CMCK) is as the following:

$$I2CMCK = CCLK / [5* (Prescaler + 2)]$$

For example, if I2CMCK is 100 KHz and CCLK is 100 MHz, pre-scalar (REG[E5h~E6h]) must be set to 200. Data transfer between I2C Master and Slave is synchronized by I2CMCK on the basis of byte. Each data byte is 8bit. There is one I2CMCK pulse for each I2CMDA bit and the MSB will be transmitted first. And then an acknowledge bit will be transmitted for each transferred byte. Each bit is processed during the high period of I2CMCK so that the I2CMDA could be changed only during the low period of I2CMCK and must be held stable during the high period of I2CMCK.

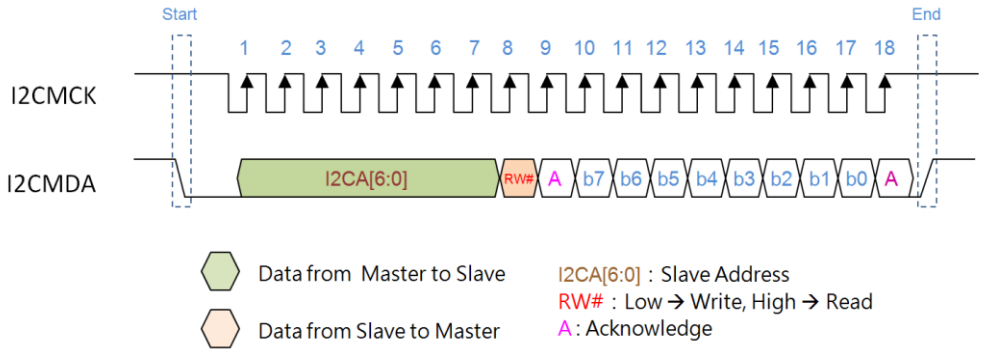


Figure 13-16: I2C Communication Protocol

The standard I2C communication protocol consists of four parts:

- Start Signal
- Slave Address Transfer
- Data Transfer
- End Signal

Example 1: Write 1 Byte data to Slave Device

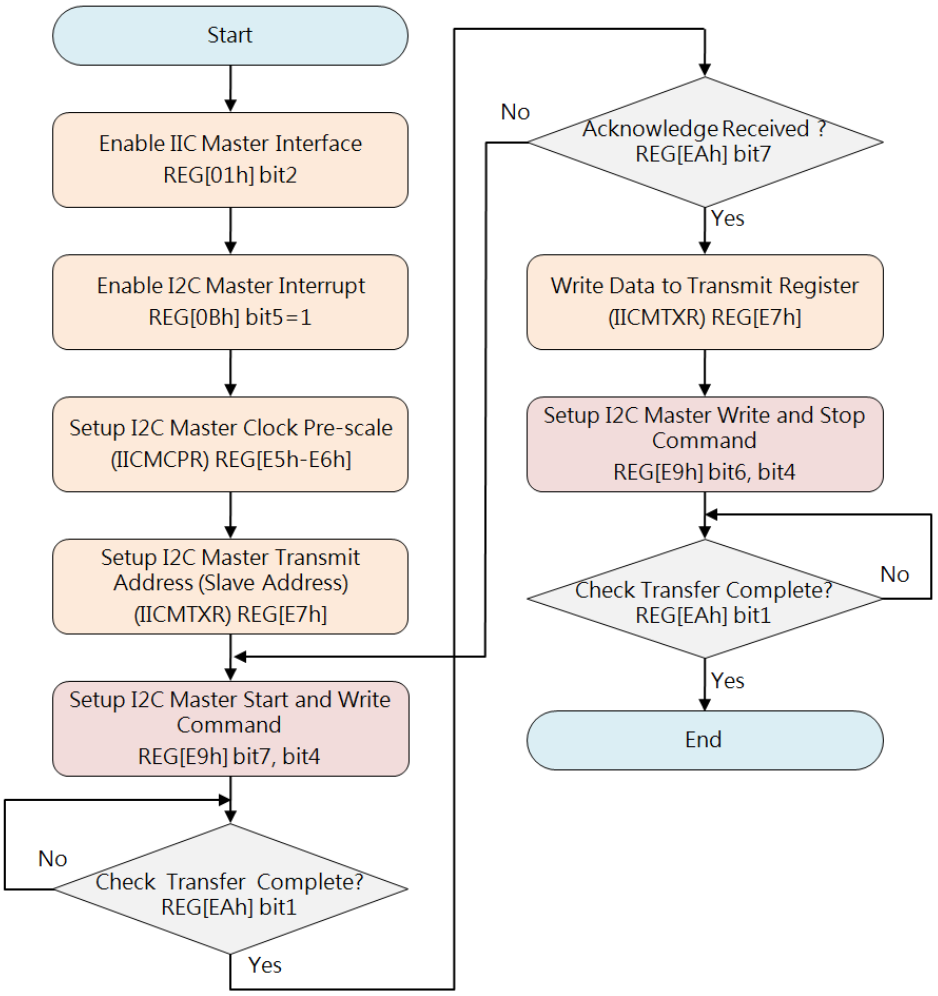


Figure 13-17: Write Data to Slave

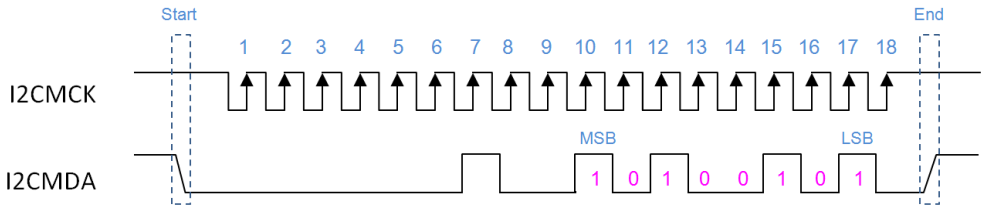


Figure 13-18: Write Data "A5" to Slave (Address= 0x01)

Example 2: Read 1 Byte Data from Slave Device

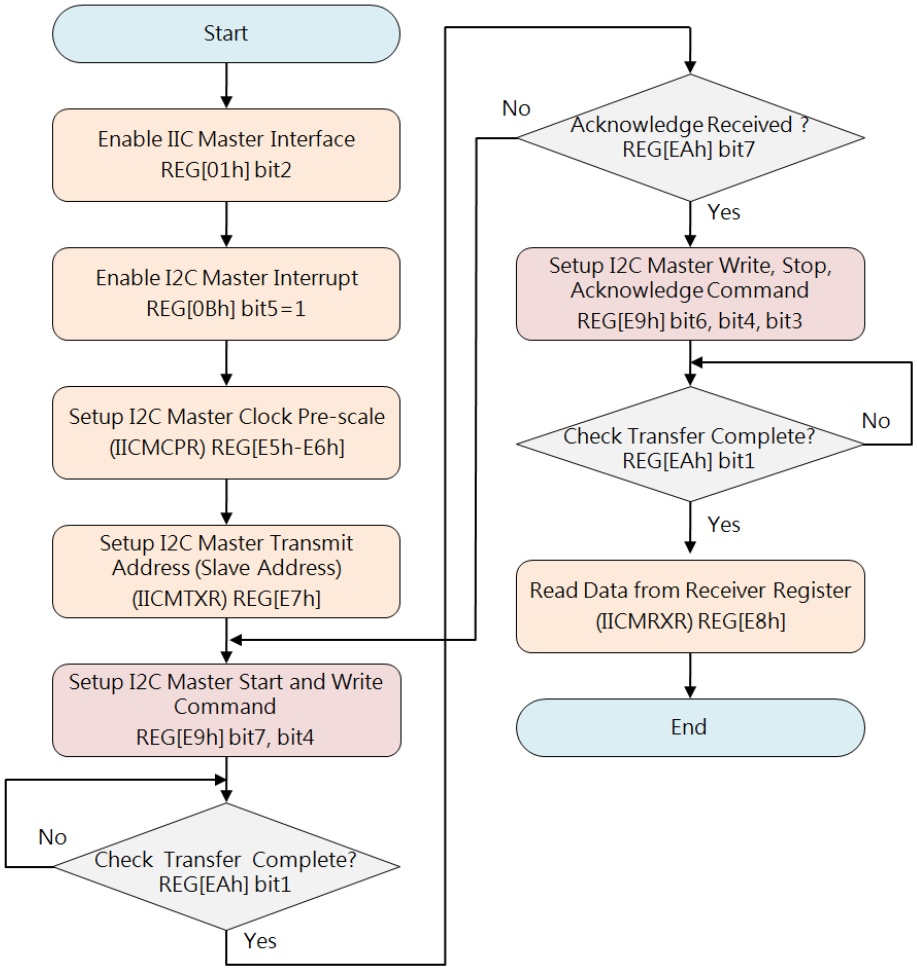


Figure 13-19: Read Data from Slave

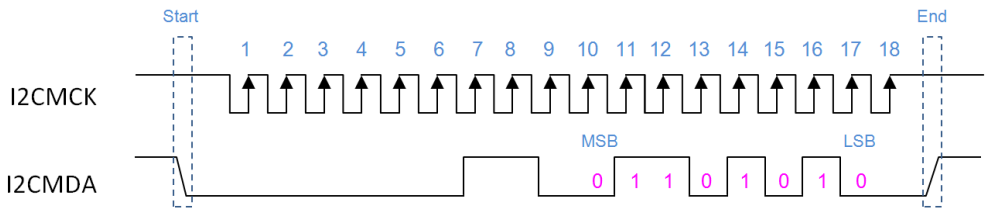


Figure 13-20: Read Data "6A" from Slave (Address=0x01)

14. Image Decode Unit

LT758x has a built-in Image Decode Unit for decoding JPEG pictures. JPEG (Joint Photographic Experts Group) is a common image compression format. It uses lossy compression methods to reduce the file size. LT758x supports standard JPEG Baseline format. Developers may retrieve the JPEG picture data stored in the Serial Flash (connected to LT758x), and display them onto the TFT display.

14.1 JPG Decoder

■ **Applicable to JPG Baseline Format:**

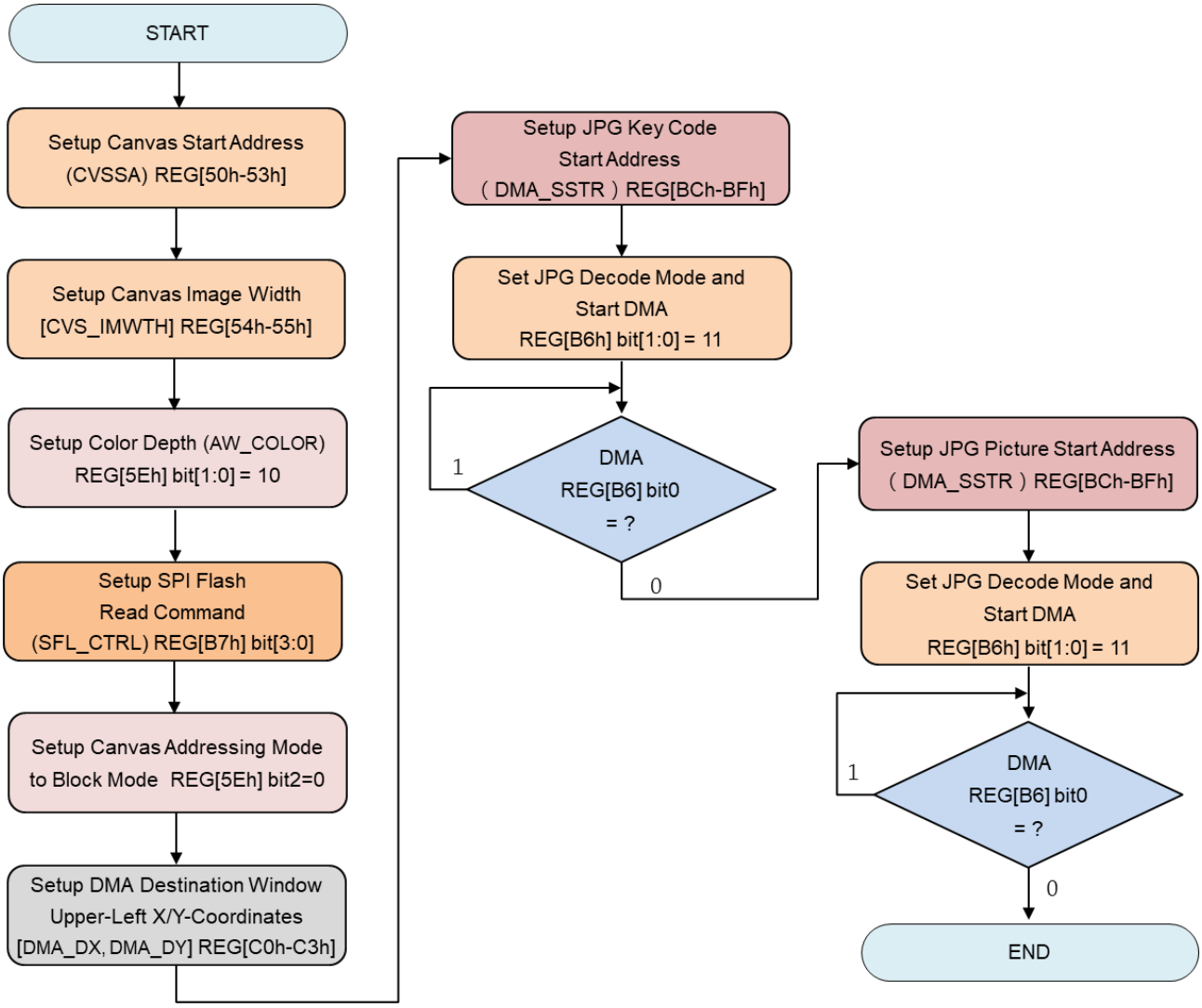


Figure 14-1: Flowchart of Displaying a JPG Picture

Example: To retrieve and display a 800x480 JPG picture from the SPI Flash.

```
Main_Image_Start_Address(0);          //Set Main Window: Display Layer
Main_Image_Width(800);                //Main Window Width: 800
Main_Window_Start_XY(0,0);            //Set Main Window Upper-Left Coordinate
Canvas_Image_Start_address(0);        //Set Canvas Window
Canvas_image_width(800);              //Canvas Window Width: 800
Active_Window_XY(0,0);                //Set Active Window Upper-Left Coordinate
Active_Window_WH(800, 480);           //Active Window Width & Height
Memory_24bpp_Mode();                  //Set Color Depth: 24bpp
Graphic_Mode();                       //Set as Graphic Mode, REG[03] bit2

Enable_SFlash_SPI();                  //Enable the SPI I/F for the Flash, REG[01h] bit1= 1
DMA_Read_6BH();                       //Set Flash Read Command , REG[B7h] bit[3:0]
Reset_CPOL();                         //SPI Protocol: Mode 0, REG[B9h] bit1
Reset_CPHA();                         //SPI Protocol: Mode 0, REG[B9h] bit0
SPI_Clock_Period(0);                  //Set SPI Clock, REG[BBh] bit[3:0]

Memory_XY_Mode();                     //Set as Block Mode, REG[5Eh] bit2 = 0

SFI_DMA_Destination_Upper_Left_Corner(X1,Y1); //Set DMA Destination, REG[C0h-C3h]

//Set the JPG KeyCode Start Address (in Flash)
SFI_DMA_Source_Start_Address(JPG_Key_Addr);     // REG[BCh-BFh]
Start_JPG_DMA();                      //Start DMA and Decode, REG[B6h] bit[1:0] = 11
Check_Busy_JPG_DMA();                 //Check if DMA is done, REG[B6h] bit0

//Set the JPG picture data Start Address (in Flash),
SFI_DMA_Source_Start_Address(JPG_Pic_Addr);     // REG[BCh-BFh]
Start_JPG_DMA();                      //Start DMA and Decode, REG[B6h] bit[1:0] = 11
Check_Busy_JPG_DMA();                 //Check if DMA is done, REG[B6h] bit0
```

Note:

1. Before decoding a JPG picture, the 16Bytes of JPG KeyCode has to be loaded to LT758x. Also, the 16Bytes of JPG KeyCode should be programmed into the SPI Flash so that LT758x can retrieve them afterwards.
2. JPG KeyCode (16Bytes): { FF DB FF E0 00 10 4A 45 52 52 00 01 01 01 00 0A }
3. Develpers may utilize the converting tool, UartTFT.exe, provided by Levetop to convert JPG pictures. The tool will add the 16Bytes of JPG KeyCode at the beginning of each JPG picture.

15. GPIO Port

LT758x provides many GPIO ports that can be extended as MCU I/O interfaces. Usually these GPIO ports are shared with other control signals. Please refer to Table 15-1 as below, and note these GPIO ports are available only when their mapping control signals are disabled.

Table 15-1: GPIO Port vs. Shared Signals

GPIO Port	Shared Signals
GPIOA[7:0]	DB[15:8]
GPIB[4]	I2CMCK
GPOB[4]	I2CMDA
GPIB[3:0]	{A0, WR#, RD#, CS#}
GPIOC[7]	PWM[0]
GPIOC[4:0],	{ SFCS[1]#, SFCS[0]#, MISO, MOSI, SFCLK }
IOD[7:0]	PD[18, 2, 17, 16, 9, 8, 1, 0]

Table 15-2 shows a list of the GPIO ports supported by different models of LT758x. These GPIO ports can be set to output or input.
The written and read data can be set or retrieved by REG[F0h~F6h]. Refer to Chapter 17 for more detail.

Table 15-2: GPIO Ports Supported by LT758x

Model	GPIO Number	GPIO Ports
LT7586 LT7583	28	GPIOA[7:0], GPIB[4], GPOB[4], GPIB[3:0], GPIOC[7], GPIOC[4:0], GPIOD[7:0]
LT7580	26	GPIOA[7:0], GPIB[3:0], GPIOC[7], GPIOC[4:0], GPIOD[7:0]

16. Power Management

LT758x has four operation modes, distinguished by the amount of power consumption, from high to Low: Normal mode, Standby mode, Suspend mode, and Sleep mode. The four modes of operation are set by the register REG[DFh]. Following is the comparison table for the related clock status of the four operation modes:

Table 16-1: Power Management vs. Clock Status

Item	Normal	Standby		Suspend		Sleep	
	PLL Enable	Parallel MCU	Serial MCU	Parallel MCU	Serial MCU	Parallel MCU	Serial MCU
MCLK	MPLL Clock	MPLL Clock	MPLL Clock	OSC	OSC	Stop	Stop
CCLK	CPLL Clock	OSC	OSC	Stop	OSC	Stop	OSC
PCLK	PPLL Clock	Stop	Stop	Stop	Stop	Stop	Stop
CPLL	ON	ON	ON	OFF	OFF	OFF	OFF
MPLL	ON	ON	ON	OFF	OFF	OFF	OFF
PPLL	ON	ON	ON	OFF	OFF	OFF	OFF

Note:

1. When LT758x enters the power saving mode, the LCD interface will not output the signals. Therefore, before entering the power-saving mode, Host needs to turn off (display off / power down) the LCD module to avoid LCD polarization damage.
2. "OSC" means external X'tal oscillator.

16.1 Normal Mode

In this mode, three of the internal PLL functions are working normally. That is, Host sets Registers to control three PLL to generate CCLK (Core Clock), MCLK (Display RAM Clock), and PCLK (LCD scan Clock) so that the whole system can work normally. Because it will take some time for the PLL to work steadily, Host must first check if the PLL frequency is in a stable state through REG[01h] bit7.

16.2 Standby Mode

If REG[DFh] bit[1:0] is set to 01b, LT758x will enter Standby mode. The System Clock (CCLK) and LCD-Scan Clock (PCLK) will stop. The Display RAM Clock will remain active and still be provided by MCLK.

- **Enter Standby Mode:**
  1. Select Standby Mode (REG[DFh] bit[1:0] = 01b)
  2. Setup REG[DFh] bit7 to 1 (Enter Standby Mode)
  3. Host may check the bit1 of Status Register (STSR) , and wait for this bit to become 1 to make sure LT758x enters Standby mode.
- **Back to Normal Mode:**
  1. Setup REG[DFh] bit7 to 0 (Exit Standby Mode)
  2. Host may check the bit1 of Status Register (STSR), and wait for this bit to become 0 to make sure LT758x is back to Normal Mode.

16.3 Suspend Mode

If REG[DFh] bit[1:0] is set to 10b, LT758x will enter Suspend mode. The System Clock (CCLK), Display RAM Clock(MCLK) and LCD-Scan Clock (PCLK) will stop. The clock of Display RAM will be provided by the OSC Clock.

- **Enter Suspend Mode:**
  1. Setup appropriate Display RAM(SDRAM) Refresh Clock based on the OSC frequency.
  2. Select Suspend Mode (REG[DFh] bit[1:0] = 10b)
  3. Setup REG[DFh] bit7 to 1 (Enter Suspend Mode)
  4. Host may check the bit1 of Status Register (STSR), and wait for this bit to become 1 to make sure LT758x enters Suspend Mode.
- **Back to Normal Mode:**
  1. Setup REG[DFh] bit7 to 0 (Exit Suspend Mode)
  2. Host may check the bit1 of Status Register (STSR), and wait for this bit to become 0 to make sure LT758x is back to Normal Mode.

16.4 Sleep Mode

If REG[DFh] bit[1:0] is set to 11b, LT758x will enter Sleep Mode, and all of Clocks and PLL will stop operating.

■ Enter Sleep Mode:

1. Select Sleep Mode (REG[DFh] bit[1:0] = 11b)
2. Setup REG[DFh] bit7 to 1 (Enter Sleep Mode)
3. If REG[E0h] bit7 is 0, the Display RAM will enter Power Down mode before LT758x enters Sleep mode. If REG[E0h] bit7 is 1, the Display RAM will enter Refresh mode.
4. If Host interface is Parallel Mode, then LT758x will stop OSC. If Host interface is Serial Mode, then OSC will not be stopped.
5. Host may check the bit1 of Status Register (STSR), and wait for this bit to become 1 to make sure LT758x enters Sleep Mode.

■ Back to Normal Mode:

1. Setup REG[DFh] bit7 to 0 (Exit Sleep Mode)
2. If OSC is stopped in Sleep mode, the Host has to enable OSC.
3. The Host may check the bit1 of Status Register (STSR), and wait for this bit to become 0 to make sure LT758x is back to Normal Mode.

17. Registers

LT758x provides a compatible 4 forms of Host interface, and the internal registers are read and written through these Host interface cycles. LT758x contains a Status Register and many Instruction Registers. Status Registers can read data through the state reading period, and it can only be read and cannot be written. Instruction Registers can control most of the functions through the "Command Write" cycle and the "Data Write" cycle. During the “Command Write” cycle, the address of the register is specified, and then during the “Data Wirte” cycle, data can be wriiten to the specified register. When a specified register data is to be read, the master will need to send the "Command write" cycle first. Then use the "Data Read" cycle to read the data. In other words, "Command Write" is to set the register address, and "Data Read" is to read the register data.

Table 17-1: MCU Interface Cycle

MCU Interface Cycle	CS#	A0	8080 Type MCU		6800 Type MCU		Action Description
			RD# EN	WR# RW#	RD# EN	WR# RW#	
Command	0	0	1	0	1	0	Set Register Address
Status Read	0	0	0	1	1	1	Read Status Register data
Data Write	0	1	1	0	1	0	Write Data to Register or Memory
Data Read	0	1	0	1	1	1	Read Data from Register or Memory

Note: It is suggested that users activate VSYNC interrupt, and change the register contents only after receiving the VSYNC interrupt, and finish accessing the registers in the timing between two VSYNC to avoid abnormal display.

17.1 Status Register

Table 17-2: Status Register (STSR)

Bit	Description	Default	Access
7	**Memory Write FIFO Full** 0: The FIFO of Memory Write is Not Full 1: The FIFO of Memory Write is Full Host can write data to memory only if the FIFO of Memory Write is not full.	0	RO
6	**Memory Write FIFO Empty** 0: The FIFO of Memory Write is Not Empty 1: The FIFO of Memory Write is Empty. When the FIFO of Memory Write is empty, Host can continuously write 32 pixels.	1	RO

Bit	Description	Default	Access
5	**Memory Read FIFO Full** 0：The FIFO of Memory Read is not Full 1：The FIFO of Memory Read is Full When the Memory Read FIFO is Full, Host can continuously read 32pixels.	0	RO
4	**Memory Read FIFO Empty** 0：The FIFO of Memory Read is not Empty 1：The FIFO of Memory Read is Empty When the Memory Read FIFO is not Empty, Host can continuously read pixel data from Memory.	1	RO
3	**Core Task is Busy, Fontwr_Busy** This bit represents whether the on-going action is completed (BTE, Geometry engine, DMA, text writing, or graphic writing). 0：The action is completed or idle 1：The action is not completed or in a busy state When the Host program switches text and graphic mode, or changes the base diagram related settings, the program must first verify if the LT758x is idle. **Note:** In text mode, before program changes text rotation, line spacing, character spacing, foreground color, background color, and text/graphics settings, Host must confirm this bit is 0.	0	RO
2	**SDRAM Ready for Access** 0：Display RAM is not ready for access 1：Display RAM is ready for access Before checking this bit, Host has to setup REG[E4h] bit0 “SDR_INIT” as 1	0	RO
1	**Operation Mode Status** 0：Normal operation state 1：Inhibit operation state Inhibit operation state means internal reset event is running, or initial display is running, or the chip enters power saving mode. In power saving mode, this bit becomes 1 until PLL clock stops.	0	RO
0	**Interrupt Pin State** 0：No interrupt activated 1：Interrupt activated	0	RO

Note： RO stands for Read Only

17.2 Configuration Registers

REG[00h] Software Reset Register (SRR)

Bit	Description	Default	Access
7	**Reconfigure PLL Frequency** Write "1" to this bit will reconfigure PLL frequency. **Note:** 1. When PLL relative parameters are modified, PLL clock will not be changed immediately, users must set this bit as "1" to sart the configuration. 2. Users may read this bit to check whether PLL clock is changed. 3. If the PLL related registers are not changed, then writing "1" to this bit will not reconfigure PLL.	1	RW
6-4	Reserved	0	RO
3	**Select Register Group** 0: Group 0 – Default register group 1: Group 1 – Expand register group This bit is not restricted by the register lock or the slected register group.	0	RW
2	**Lock Register** 0: Enable register access (R/W to all Registers) 1: Disable register write (Read Only to all Registers), except for this bit and REG[00h] bit3.	0	RW
1	**Ext_FlatInk** 1: Force to enter DE Mode, where VSYNC/HSYNC becomes PD # / PD signals to control the external LVDS conversion IC. 0: Regular status	0	RW
0	**Software Reset** 0: Normal Operation 1: Software Reset. The bit will be cleared to "0" after reset. Software Reset only resets internal state machine. Other Registers' values will not be reset.	0	WO
0	**Warning Condition Flag** 0: No warning occurred. 1: Warning occurred. Refer to REG[E4h] bit3 description. If this flag is disabled, then the access that exceeds the memory range will loop back to address 0 and overwrite it. If this flag is enabled, then the access that exceeds the memory range will be discarded.	0	RO

REG[01h] Chip Configuration Register (CCR)

Bit	Description	Default	Access
7	**Check PLL Ready** Host may read (check) this bit to find out whether PLL clock is ready or not. Read "1" means PLL clock is ready and switched successfully. When REG[01] bit7 = 1, the contents of REG[FDh~FFh] will be different, based on the value written to this bit: If "0" is written to this bit, the content of REG[FDh~FFh] is the ID value: LT7583: REG[FDh, FEh, FFh] = [05h, 83h, 76h] LT7586: REG[FDh, FEh, FFh] = [0Ah, 86h, 76h] If "1" is written to this bit, the content of REG[FDh~FFh] is Date [Y/M/D].	1	RO WO
6	**Mask WAIT# on CS# De-assert** 0 : No Mask, WAIT# will stay Low if LT758x internal state is busy, and LT758x cannot accept next R/W cycle. If Host cycle cannot be extended while WAIT# is low, Host program should poll WAIT# and wait for it to become high, then start next access. 1: Mask, WAIT# de-assert (become High) when CS# de-assert. Therefore, Host cycle has to be extended through WAIT#	1	RW
5	Reserved	0	RW
4-3	**TFT Panel I/F Setting** 00b: 24bits TFT output 01b: 18bits TFT output 10b: 16bits TFT output 11b: No TFT output Other unused TFT output pins are set as GPIO.	01b	RW
2	**I2C Master Interface Enable/Disable** 0: Disable (GPIO Function) 1: Enable (I2C Master Function)	0	RW
1	**Serial Flash or SPI Interface Enable/Disable** 0: Disable (Set as GPIO Function) 1: Enable (Set as SPI Master)	0	RW

Bit	Description	Default	Access
0	**MCU Data Bus Width Selection** 0：8bit Data Bus 1：16bit Data Bus If Serial MCU I/F is selected, this bit will be set to “0” by LT758x.	0	RW

REG[02h] Memory Access Control Register (MACR)

Bit	Description	Default	Access
7-6	**MCU Read/Write Image Data Format** 0xb: Direct Write, for below I/F format: 1. 8bits MCU I/F. 2. 16bits MCU I/F with 8bpp data mode 1 & 2. 3. 16bits MCU I/F with 16/24bpp data mode 1/32bpp 4. Serial SPI/I2C I/F. 10b: Mask high byte of each data (e.g. 16 bit MPU I/F with 8-bpp data mode 1) 11b: Mask high byte of even data (e.g. 16 bit MPU I/F with 24-bpp data mode 2)	0	RW
5-4	**MCU Read Memory Direction (Only for Graphic Mode)** 00b：Left→Right then Top→Bottom 01b：Right→Left then Top→Bottom 10b：Top→Bottom then Left→Right 11b：Bottom→Top then Left→Right These two bits can be ignored if the Canvas is set to Linear addressing mode.	0	RW
3	Reserved	0	RO
2-1	**MCU Write Memory Direction (Only for Graphic Mode)** 00b：Left→Right then Top→Bottom（Original）。 01b：Right→Left then Top→Bottom（Horizontal Flip） 10b：Top→Bottom then Left→Right（Rotate 90° to the right & Horizontal Flip） 11b：Bottom→Top then Left→Right（Rotate 90° to the left） These two bits can be ignored if the Canvas is set to Linear addressing mode.	0	RW
0	**Reserved (Must keep it as 0)**	0	RO

REG[03h] Input Control Register (ICR)

Bit	Description	Default	Access
7	**Interrupt Pin Active Level** 0: Low Active 1: High Active	0	RW
6	**External Interrupt Signal - PSM[0] Pin De-bounce** 0: No De-bounce required 1: Enable De-bounce	0	RW
5-4	**External Interrupt Signal - PSM[0] Trigger Type** 00b: low level trigger 01b: falling edge trigger 10b: high level trigger 11b: rising edge trigger	00b	RW
3	**Graphic Cursor Size** 0: 32x32 1: 64x64	0	RW
2	**Text Mode Enable** 0: Graphic Mode 1: Test Mode Before setting this bit, users must confirm if the Task Busy bit of the status register is idle. This bit will be "0" if the Canvas addressing mode is set to Linear mode.	0	RW
1-0	**Memory Port Read/Write Destination Selection** 00b: Select to write Display RAM for Image/Patten/User Defined Chracters. Support Read-Modify-Write. --- For below settings, the read/write of the data port is the same as general register read/write operations, only lower byte of the MCU bus is accepted. 01b: Select to write RGB Gamma table. Each color includes 256 bytes. MCU needs to specify the desired Gamma table and write 256 bytes continuously. 10b: Select to write Graphic Cursor. 4 sets of 32x32 Graphic Cursor color can be set. Each set has 128x16buts. MCU needs to specify th target graphic cursor and write 256 bytes continuously. 11b: Select to write Color Palette. In Non-Index color mode (REG10h[1] = 0): The destination is 64*12 bits SDRAM. Since MCU writes 8bits data at a time, when it writes every even byte only	0	RW

Bit	Description	Default	Access
	the lower 4bits will be stored to the RAM. Color Palette RAM does not supported read operation. MCU needs to write 128bytes continuously. In Index color mode（REG10h[1] = 1）: The destination is 256*24 bits SDRAM. MCU writes 3xN bytes of the index color (8bits B,G,R data) continuously, where N is the amount of the index color to be written, and N<=256. **Note:** When changing or reading the color palette SDRAM in Index mode, the color depth of Main/PIP1/PIP2 has to be set as non-8bpp temporarily.		

REG[04h] Memory Data Read/Write Port（MRWDP）

Bit	Description	Default	Access
7-0	**Write Function: Write Data to Memory** Write data to the SDRAM destination set in REG[03h][1:0]. Continuous write is acceptable for large amount of data. **Note:** **a. Image Data in SDRAM**: Set to 8/16-bits based on the MCU I/F bit width. Host R/W image data format can be set. Canvas color depth and related settings can also be set in block mode. **b. Pattern Data for BTE Operation in SDRAM**: Set to 8/16-bits based on the MCU I/F bit width. Host R/W image data format can be set. Canvas color depth and related settings can also be set in block mode. Active window's width and height should be set as 8x8 or 16x16 depending on users needs. **c. User-Characters in SDRAM**: Set to 8/16-bits based on the MCU I/F bit width. Host R/W image data format can be set, and canvas is set to linear mode. **d. Character Code**: only low 8-bits MCU data are accepted, similar to normal register R/W. For two bytes character code, input high byte first. For user defined Character, code < 8000h is half size, code >= 8000h is full size. **e. Gamma Table Data**: only low 8-bits MCU data are accepted. Users must set "Select Gamma table sets([3Ch] Bit[6-5])" to clear internal Gamma table's address counter then start to write data. Users should write 256 bytes data to SDRAM. **f. Graphic Cursor RAM Data**: only low 8-bits MPU data are accepted. Users must set "Select Graphic Cursor sets" bits to clear internal Graphic Cursor RAM address counter then start to write data. **g. Color Palette RAM Data**: In Non-Index color mode（REG10h[1] = 0）: only low 8-bits MPU data are accepted. Users must	--	RW

Bit	Description	Default	Access
	program full Color palette RAM in a continuous 128 byte data, and write to memory data port, and cannot change register address during the writing process. In Index color mode（REG10h[1] = 1）: MCU writes 3xN bytes of the index color (8bits B,G,R data) continuously, where N is the amount of the index color to be written, and N<=256. **Read Function: Read Data from Memory** To read data from memory, REG[03h][1:0] has to be set. Continuous data read cycle can be accepted in case of reading great amount of data. **Note 1:** When reading different port address, a dummy read must be issued. The first data read cycle is dummy read and the data should be ignored. Graphic Cursor RAM & Color palette RAM data do not support data read function. **Note 2:** No matter what the color depth setting is, data reading is based on 4 bytes. **Note 3:** If users want to change the address of the written registers, but there are data already written to Display RAM, users must make sure that Core Task Busy status bit becomes idle first.		

17.3 PLL Setting Register

REG[05h] PCLK PLL Control Register 1 (PPLLC1)

Bit	Description	Default	Access
7-6	**PCLK Output Divider Ratio, OD[1:0]** 00b: Divided by 1. 01b: Divided by 2. 10b: Divided by 4. 11b: Divided by 8.	11b	RW
5	**PCLK Extra Output Divider Ratio** 1: Add additional frequency divider circuit (divided by 2) besides PCLK Output Divider. 0: set to PCLK Output Divider	0	RW
4-1	**PCLK Input Divider Ratio, N[3:0]** The value should be 1~15	0011b (3)	RW
0	**PCLK Feedback Divider Ratio of Loop, PM_hl** This bit must be set as 0	0	RW

REG[06h] PCLK PLL Control Register 2 (PPLLC2)

Bit	Description	Default	Access
7	Reserved	0	RW
6-0	**PCLK Feedback Divider Ratio, M[6:0]** Total 7bits，the value should be 4 ~ 127	48h (72)	RW

Note: PCLK is the clock signal provided to the TFT driver. (Pixel Clock)

REG[07h] MCLK PLL Control Register 1 (MPLLC1)

Bit	Description	Default	Access
7-6	**MCLK Output Divider Ratio, OD[1:0]** 00b: Divided by 1. 01b: Divided by 2. 10b: Divided by 4. 11b: Divided by 8.	10b	RW
5	Reserved	0	RW
4-1	**MCLK Input Divider Ratio, N[3:0]** The value should be 1 ~ 15	0011b (3)	RW
0	**MCLK Feedback Divider Ratio of Loop, MM_hl** This bit must be set as 0	0	RW

REG[08h] MCLK PLL Control Register 2 (MPLLC2)

Bit	Description	Default	Access
7	Reserved	0	RW
6-0	**MCLK Feedback Divider Ratio, M[6:0]** Total 7bits，the value should be 4 ~ 127	60h (96)	RW

Note: MCLK is the clock signal for SDRAM (Memory Clock)

REG[09h] CCLK PLL Control Register 1 (CPLLC1)

Bit	Description	Default	Access
7-6	**CCLK Output Divider Ratio, OD[1:0]** 00b: Divided by 1. 01b: Divided by 2. 10b: Divided by 4. 11b: Divided by 8.	10b	RW
5	Reserved	0	RW
4-1	**CCLK Input Divider Ratio, N[3:0]** The value should be 1 ~ 15	0011b (3)	RW
0	**CCLK Feedback Divider Ratio of Loop, CM_hl** This be must be set as 0	0	RW

REG[0Ah] CCLK PLL Control Register 2 (CPLLC2)

Bit	Description	Default	Access
7	Reserved	0	RW
6-0	**CCLK Feedback Divider Ratio, M[6:0]** Total 7bits，the value should be 4 ~ 127	60h (96)	RW

Note: CCLK is the clock signal for System Core (Core Clock). To calculate PLL frequency, refer to Section 4.1

17.4 Interrupt Control Register

REG[0Bh] Interrupt Enable Register (INTEN)

Bit	Description	Default	Access
7	**Wakeup/Resume Interrupt Enable** 0: Disable 1: Enable	0	RW
6	**External Interrupt Input - PSM[0] Enable** 0: Disable 1: Enable (MCU I/F must be parallel, and PSM[2] = 0)	0	RW
5	**I2C Master Interrupt Enable** 0: Disable 1: Enable	0	RW
4	**VSYNC Time Base Interrupt Enable** 0: Disable Interrupt 1: Enable Interrupt This interrupt is to inform MCU that the VSYNC of the LCD has tearing effects.	0	RW
3	Reserved	0	RW
2	**Serial Flash DMA Complete / Draw Task Finished / BTE Process Complete etc. Interrupt Enable** 0: Disable Interrupt 1: Enable Interrupt	0	RW
1	**PWM Timer-1 Interrupt Enable** 0: Disable Interrupt 1: Enable Interrupt	0	RW
0	**PWM Timer-0 Interrupt Enable** 0: Disable Interrupt 1: Enable Interrupt	0	RW

REG[0Ch] Interrupt Event Flag Register (INTF)

Bit	Description	Default	Access
7	**Wakeup/Resume Interrupt Flag** **Write: Clear the Interrupt Flag** 0: No operation 1: Clear Wakeup / Resume Interrupt Flag **Read: Read the Interrupt Flag Status** 0: No Wakeup / Resume Interrupt triggered 1: Wakeup / Resume Interrupt triggered	0	RO WO
6	**External Interrupt Input - PSM[0] Flag** **Write: Clear PSM[0] Pin Interrupt Flag** 0: No operation 1: Clear PSM[0] Interrupt Flag **Read: Read the PSM[0] Pin Interrupt Status** 0: No PSM[0] Interrupt triggered 1: PSM[0] Interrupt triggered	0 (PSM[0])	RO WO
5	**I2C Master Interrupt Flag** **Write: Clear I2C Master Interrupt Flag** 0: No operation 1: Clear I2C Master Interrupt Flag **Read: Read I2C Master Interrupt Status** 0: No I2C Master Interrupt triggered 1: I2C Master Interrupt triggered	0	RO WO
4	**VSYNC Time Base Interrupt Flag** **Write: Clear VSYNC Interrupt Flag** 0: No operation 1: Clear VSYNC Interrupt Flag **Read: Read VSYNC Interrupt Flag Status** 0: No VSYNC Interrupt triggered 1: VSYNC Interrupt triggered	0	RO WO
3	Reserved	0	NA
2	**Serial Flash DMA Complete | Draw Task Finished | BTE Process Complete etc. Interrupt Flag** **Write: Clear Process Complete Interrupt Flag** 0: No operation 1: Clear Interrupt Flag **Read: Read Process Complete Interrupt Flag Status** 0: No Interrupt triggered 1: Interrupt triggered	0	RO WO

Bit	Description	Default	Access
1	**PWM1 Timer Interrupt Flag** **Write: Clear PWM1 Interrupt Flag** 0: No operation 1: Clear PWM1 Interrupt Flag **Read: Read PWM1 Interrupt Flag Status** 0: No PWM1 Interrupt triggered 1: PWM1 Interrupt triggered	0	RO WO
0	**PWM0 Timer Interrupt Flag** **Write: Clear PWM0 Interrupt Flag** 0: No operation 1: Clear PWM0 Interrupt Flag **Read: Clear PWM0 Interrupt Flag** 0: No PWM0 Interrupt triggered 1: PWM0 Interrupt triggered	0	RO WO

Note: If the MCU receives interrupts, but the flags of this register show no interrupts triggered, then the MCU should check REG[BAh] for the status of the SPI Master State Register Interrupt Flag.

REG[0Dh] Mask Interrupt Flag Register (MINTFR)

Bit	Description	Default	Access
7	**Mask Wakeup/Resume Interrupt Flag** 0: Unmask 1: Mask	0	RW
6	**External Interrupt Input - PSM[0] Flag** 0: Unmask 1: Mask (The Wakeup function is also masked.)	0	RW
5	**I2C Master Interrupt Flag** 0: Unmask 1: Mask	0	RW
4	**VSYNC Time Base Interrupt Flag** 0: Unmask 1: Mask	0	RW
3	Reserved	0	RW
2	**Serial Flash DMA Complete | Draw Task Finished | BTE Process Complete etc. Interrupt Flag** 0: Unmask 1: Mask	0	RW
1	**PWM1 Timer Interrupt Flag** 0: Unmask 1: Mask	0	RW

Bit	Description	Default	Access
0	**PWM0 Timer Interrupt Flag** 0：Unmask 1：Mask	0	RW

Note: If the MCU masks and disables the interrupt flags, then LT758x will not send the interrupt to the MCU. If only some unmasked but disabled interrupt flags are used, the MCU may check the interrupt flags to find out whether there is an interrupt. All interrupts will be masked during the display initialization.

REG[0Eh] Pull-High Control Register (PUENR)

Bit	Description	Default	Access
7-6	Reserved	0	RO
5	**GPIOF[7:0]/PD Pull-High Enable** 0：Disable Pull-High resistor 1：Enable Pull-High resistor	0	RW
4	**GPIOE[7:0]/PD Pull-High Enable** 0：Disable Pull-High resistor 1：Enable Pull-High resistor	0	RW
3	**GPIOD[7:0]/PD Pull-High Enable** 0：Disable Pull-High resistor 1：Enable Pull-High resistor	0	RW
2	**GPIO-C[4:0]/SPIM Pull-High Enable** 0：Disable Pull-High resistor 1：Enable Pull-High resistor	0	RW
1	**DB[15:8]/GPIOA Pull- High Enable** 0：Disable Pull-High resistor 1：Enable Pull-High resistor	0	RW
0	**DB[7:0] Pull- High Enable** 0：Disable Pull-High resistor 1：Enable Pull-High resistor	0	RW

Note: The bits[5:2] are only valid when the GPIO function is enabled.

REG[0Fh] PD for GPIO Function Select Register (PSFSR)

Bit	Description	Default	Access
7-0	These bits must be set as 0	0	RW

17.5 LCD Display Control Registers

REG[10h] Main/PIP Window Control Register (MPWCTR)

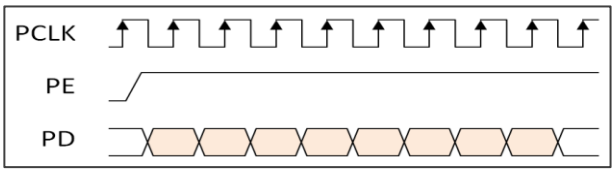
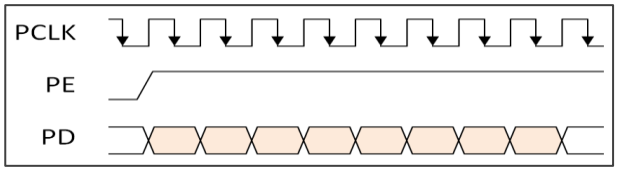
Bit	Description	Default	Access
7	**PIP-1 Window Enable/Disable** 0: Disable PIP-1 window 1: Enable PIP-1 window PIP-1 window is always on top of PIP-2 window.	0	RW
6	**PIP-2 Window Enable/Disable** 0: Disable PIP-2 window 1: Enable PIP-2 window PIP-1 window is always on top of PIP-2 window.	0	RW
5	**Ignore Alpha** 0: Alpha value is valid 1: Alpha value is ignored **Note:** This function is used for correctly displaying 32bpp BMP pictures.	0	RW
4	**Select Configure PIP-1 or PIP-2 Window's Parameters** PIP window's parameter includes Color Depth, Starting Address, Image Width, Display Coordinates, Window Coordinates, Window Width and Window Height. 0: To configure PIP 1's parameters. 1: To configure PIP 2's parameters.	0	RW
3-2	**Main Image Color Depth Setting** 00b: 8bpp Generic TFT (256 colors, RGB: 332)。 01b: 16bpp Generic TFT (65K colors, RGB:565)。 10b: 24bpp Generic TFT (1.67M colors, RGB:888)。 11b: 32bpp Generic TFT (1.67M colors+Alpha, ARGB: 8888)	1	RW
1	**Index color** 0: 8bpp-RGB332 mode 1: 8bpp-Index mode, used to convert the index colors to 24bpp colors **Note 1:** In 8bpp-Index mode, the ROP function will be invalid for 8bpp pictures. **Note 2:** ■ 8bpps-RGB332 pictures and 8bpp-Index pictures cannot be displayed at the same time as the colors will be abnormal. ■ Decoding 8bpp BMP pictures will change the index contents of Palette RAM. ■ The setting priority of Index color is higher than the 256 grayshade mode.	0	RW

Bit	Description	Default	Access
0	**To Control Panel's Synchronous Signals** 0：Sync Mode： Enable VSYNC, HSYNC, and PDE 1：DE Mode： Only enable PDE. VSYNC & HSYNC are in idle state. **Note:** This bit will be affected by REG[00h] bit[1] which forces LT758x to enter DE mode.	0	RW

REG[11h] PIP Window Color Depth Setting (PIPCDEP)

Bit	Description	Default	Access
7	**PIP1_VDIR: Vertical Scan Direction** 0：From top to bottom 1：From bottom to top **Note:** The direction is relative to PIP1 window	0	RW
6	**PIP2_VDIR: Vertical Scan Direction** 0：From top to bottom 1：From bottom to top **Note:** The direction is relative to PIP2 window	0	RW
5	**PIP1 HDIR: Horizontal Scan Direction** 0：From left to right 1：From right to left **Note:** The direction is relative to PIP1 window	0	RW
4	**PIP2 HDIR: Horizontal Scan Direction** 0：From left to right 1：From right to left **Note:** The direction is relative to PIP2 window	0	RW
3-2	**PIP-1 Window Color Depth Setting** 00b：8bpp Generic TFT 256 colors 01b：16bpp Generic TFT 65K colors 10b：24bpp Generic TFT 1.67M colors 11b：32bpp Generic TFT 1.67M colors + Alpah	1	RW
1-0	**PIP-2 Window Color Depth Setting** 00b：8bpp Generic TFT 256 colors 01b：16bpp Generic TFT 65K colors 10b：24bpp Generic TFT 1.67M colors 11b：32bpp Generic TFT 1.67M colors + Alpah	1	RW

REG[12h] Display Configuration Register (DPCR)

Bit	Description	Default	Access
7	**PCLK Inversion** 0: TFT Panel fetches PD at PCLK rising edge.  1: TFT Panel fetches PD at PCLK falling edge.  	0	RW
6	**Display ON/OFF** 0: Display Off 1: Display On	0	RW
5	**Display Test Color Bar** 0: Disable 1: Enable VDIR = 0, display horizontal color bar. VDIR = 1, display vertical color bar **Note:** This function has the highest priority.	0	RW
4	**HDIR: Horizontal Scan Direction** 0: From left to right 1: From right to left **Note:** The scan direction is relative to the TFT panel.	0	RW
3	**VDIR: Vertical Scan Direction** 0: From top to bottom 1: From bottom to top **Note:** The scan direction is relative to the TFT panel.	0	RW
2-0	**Parallel PD[23:0] Output Sequence** 000b: RGB 001b: RBG 010b: GRB 011b: GBR 100b: BRG 101b: BGR 110b: Gray 111b: Send out Idle State. All data are 0 (Black) or 1 (White). REG[13h] has to be set accordingly too.	0	RW

REG[13h] Panel Scan Clock and Data Setting Register (PCSR)

Bit	Description	Default	Access
7	**HSYNC Polarity** 0: Low Active 1: High Active	0	RW
6	**VSYNC Polarity** 0: Low Active 1: High Active	0	RW
5	**PDE Polarity** 0: High Active 1: Low Active	0	RW
4	**PDE Idle State** 0: Pin "PDE" output Low 1: Pin "PDE" output Hight This is used to setup the PDE output status in Power Saving mode or Display Off.	0	RW
3	**PCLK Idle State** 0: Pin "PCLK" output Low 1: Pin "PCLK" output High This is used to setup the PDE output status in Power Saving mode or Display Off.	0	RW
2	**PD Idle State** **(In Vertical/Horizontal Non-Display Period or Power Saving Mode or DISPLAY OFF)** 0: Pins "PD[23:0]" output Low 1: Pins "PD[23:0]" output High This is used to setup the PD output status in Vertical/Horizontal Non-Display Period, Power Saving mode or Display Off.	0	RW
1	**HSYNC Idle State** 0: Pin "HSYNC" output Low 1: Pin "HSYNC" output High This is used to setup the HSYNC output status in Power Saving mode or Display Off.	1	RW
0	**VSYNC Idle State** **(In Power Saving Mode or DISPLAY OFF)** 0: Pin "VSYNC" output Low 1: Pin "VSYNC" output High This is used to setup the VSYNC output status in Power Saving mode or Display Off.	1	RW

Note: The sum of (HST + HPW + HND) had better be larger than 64Pixels to prevent scanning FIFO empty. If both PIP1 and PIP2 are enabled and are very close to the left side of the window, there is only a small change in PIP1 and PIP2's UL-X. Please refer to Figure 7-1 TFT-LCD RGB interface Timing diagram.

REG[14h] Horizontal Display Width Register (HDWR)

Bit	Description	Default	Access
7-0	**Horizontal Display Width Setting** This register is used to set a horizontal display width. The specified LCD screen resolution is 8 pixels in one unit resolution. **Horizontal Display Width (pixels) = (HDWR + 1) * 8 + HDWFTR** HDWFTR (REG[15h]) is the fine-tuning value for Horizontal Display Width. Each fine-tuning resolution is 1 pixel, and the maximum horizontal width is 2,048 pixels.	4Fh	RW

REG[15h] Horizontal Display Width Fine Tune Register (HDWFTR)

Bit	Description	Default	Access
7-3	Reserved	0	RO
2-0	**Horizontal Display Width Fine Tuning** This Register is the fine-tuning value for Horizontal Display Width, and each fine-tuning resolution is 1 pixel. **Horizontal Display Width (pixels) = (HDWR + 1) * 8 + HDWFTR**	0	RW

REG[16h] Horizontal Non-Display Period Register (HNDR)

Bit	Description	Default	Access
7-6	Reserved	0	RO
5-0	**Horizontal Non-Display Period** This register assigns the period of Horizontal Non-Display. It is also called「**Back Porch**」 **Horizontal Non-Display Period (Pixels) = (HNDR + 1) * 8 + HNDFTR** HNDFTR (REG[17h]) is the fine-tuning value for Horizontal Non-display Period. Each fine-tuning resolution is 1 pixel, and the maximum horizontal width is 2,048 pixels.	03h	RW

REG[17h] Horizontal Non-Display Period Fine Tune Register (HNDFTR)

Bit	Description	Default	Access
7-4	Reserved	0	RO
3-0	**Horizontal Non-Display Period Fine Tuning** This register is for setting the fine-tuning value of the Horizontal Non-display Period, and each fine-tuning resolution is 1 pixel. it is used to support the SYNC mode panel.	06h	RW

REG[18h] HSYNC Start Position Register (HSTR)

Bit	Description	Default	Access
7-5	Reserved	0	RO
4-0	**HSYNC Start Position** This register specifies the starting address of the HSYNC. The starting point for the calculation is the point at which the end of the display area is to start producing HSYNC. The basic unit of each adjustment is 8 pixels. It is also called “Front Porch”. **HSYNC Start Position = (HSTR + 1) * 8**	1Fh	RW

REG[19h] HSYNC Pulse Width Register (HPWR)

Bit	Description	Default	Access
7-5	Reserved	0	RO
4-0	**HSYNC Pulse Width** **HSYNC Pulse Width (Pixels) = (HPW + 1) * 8**	0	RW

REG[1Bh-1Ah] Vertical Display Height Register (VDHR)

Bit	Description	Default	Access
15-0	**Vertical Display Height** REG[1Ah] mapping to VDHR [7:0] REG[1Bh] bit[2:0] mapping to VDHR [10:8], bit[7:3] are not used. The height of the vertical display is in the unit of line, and the formula is as following: **Vertical Display Height (Line) = VDHR + 1** **Note:** The maximum vertical height is 2048 pixels.	DFh 01h	RW

REG[1Dh-1Ch] Vertical Non-Display Period Register (VNDR)

Bit	Description	Default	Access
15-0	**Vertical Non-Display Period** REG[1Ch] mapping to VNDR [7:0] REG[1Dh] bit[1:0] mapping to VNDR [9:8], REG[1Dh] bit[7:2] are not used. The formula is as following: **Vertical Non-Display Period (Line) = VNDR + 1**	15h	RW

REG[1Eh] VSYNC Start Position Register (VSTR)

Bit	Description	Default	Access
7-0	**VSYNC Start Position** The starting position is from the end of the display area to the beginning of VSYNC. **VSYNC Start Position (Line) = (VSTR + 1)**	0Bh	RW

REG[1Fh] VSYNC Pulse Width Register (VPWR)

Bit	Description	Default	Access
7-6	Reserved	0	RO
5-0	**VSYNC Pulse Width** **VSYNC Pulse Width (Line) = (VPWR + 1)**	0	RW

REG[23h-20h] Main Image Start Address (MISA)

Bit	Description	Default	Access
31-0	**Main Image Start Address** REG[23h] mapping to MISA[31:24]。 REG[22h] mapping to MISA[23:16]。 REG[21h] mapping to MISA[15:8]。 REG[20h] mapping to MISA[7:0]， bit[1:0] must be 0	0	RW

REG[25h-24h] Main Image Width (MIW)

Bit	Description	Default	Access
15-0	**Main Image Width** REG[25h] bit[5:0] mapping to MIW[13:8], bit[7:6] are not used. REG[24h] mapping to MIW[7:0] The unit is pixel, which is the pixel that represents the horizontal width of the actual LCD. The maximum setting is 8,192 pixels. **Note:** The width must be a multiple of 4 when the color depth = 8bpp; the width must be even when the color depth = 16bpp; and the width can be any value when the color depth = 24bpp or 32bpp. That is, if bpp={0, 1, 2, 3} represents 8bpp, 16bpp, 24bpp & 32bpp, the width condition: (width * (bpp+1) * 8) % 32 == 0	0	RW

REG[27h-26h] Main Image Upper-Left Corner X-Coordinates (MIULX)

Bit	Description	Default	Access
15-0	**Main Image Upper-Left Corner X-Coordinates** REG[27h] bit[4:0] mapping to MIULX [12:8], bit[7:5] are not used. REG[26h] mapping to MIULX[7:0], bit[1:0] must be 0. The unit is pixel. The sum of X-Coordinates plus Horizontal Display Width cannot be greater than 8,191. **Note:** The X coordinate must be a multiple of 4 when the color depth = 8bpp; the X coordinate must be even when the color depth = 16bpp; and the X coordinate can be any value when the color depth = 24bpp or 32bpp. That is, if bpp={0, 1, 2, 3} represents 8bpp, 16bpp, 24bpp & 32bpp, the X coordinate condition: (UL_X * (bpp+1) * 8) % 4 == 0	0	RW

REG[29h-28h] Main Image Upper-Left corner Y-Coordinates (MIULY)

Bit	Description	Default	Access
15-0	**Main Image Upper-Left Corner Y-Coordinates** REG[29h] bit[4:0] mapping MIULY [12:8], bit[7:5] are not used. REG[28h] mapping to MIULY[7:0]。 Unit: Pixel. The range is between 0 and 8,191.	0	RW

REG[2Bh-2Ah] PIP Window 1 or 2 Display Upper-Left Corner X-Coordinates (PWDULX)

Bit	Description	Default	Access
15-0	**PIP Window Display Upper-Left Corner X-Coordinates** REG[2Bh] bit[2:0] mapping to PWDULX[10:8], bit[7:3] are not used. REG[2Ah] mapping to PWDULX[7:0] Unit: Pixel. X-axis coordinates should be less than the horizontal display width. This setting is relative to the PIP settings in the register of REG[10h].	0	RW

REG[2Dh-2Ch] PIP Window 1 or 2 Display Upper-Left corner Y-Coordinates (PWDULY)

Bit	Description	Default	Access
15-0	**PIP Window Display Upper-Left Corner Y-Coordinates** REG[2Dh] bit[2:0] mapping to PWDULY[10:8], bit[7:3] are not used. REG[2Ch] mapping to PWDULY[7:0] Y-axis coordinates should be less than the vertical display height. This setting is relative to the PIP settings in the register of REG[10h].	0	RW

REG[31h-2Eh] PIP Image 1 or 2 Start Address (PISA)

Bit	Description	Default	Access
31-0	**PIP Image Start Address** REG[31h] mapping to PISA [31:24] REG[30h] mapping to PISA [23:16] REG[2Fh] mapping to PISA [15:8] REG[2Eh] mapping to PISA[7:0], bit[1:0] must be 0 This setting is relative to the PIP settings in the register of REG[10h].	0	RW

REG[33h-32h] PIP Image 1 or 2 Width (PIW)

Bit	Description	Default	Access
15-0	**PIP Image Width** REG[33h] bit[5:0] mapping to PIW[13:8], bit[7:6] are not used. REG[32h] mapping to PIW[7:0] Unit: Pixel. The value is physical width, and maximum value is 8192 pixels. This width should be less than horizontal display width. This setting is relative to the PIP settings in the register of REG[10h].	0	RW

REG[35h-34h] PIP Window Image 1 or 2 Upper-Left Corner X-Coordinates (PWIULX)

Bit	Description	Default	Access
15-0	**PIP Window Image 1 or 2 Upper-Left Corner X-Coordinates** REG[35h] bit[4:0] mapping to PWIULX[12:8], bit[7:5] are not used. REG[34h] mapping to PWIULX[7:0] Unit: Pixel. The sum of X-axis coordinates and PIP image width must be less than or equal to 8,191. This setting is relative to the PIP settings in the register of REG[10h].	0	RW

REG[37h-36h] PIP Window Image 1 or 2 Upper-Left Corner Y-Coordinates (PWIULY)

Bit	Description	Default	Access
15-0	**PIP Window Image 1 or 2 Upper-Left Corner Y-Coordinates** REG[37h] bit[4:0] mapping to PWIULY[12:8], bit[7:5] are not used. REG[36h] mapping to PWIULY[7:0] Unit: Pixel. The sum of Y-axis coordinates and PIP window height should be less than or equal to 8,191. This setting is relative to the PIP settings in the register of REG[10h].	0	RW

REG[39h-38h] PIP Window 1 or 2 Width (PWW)

Bit	Description	Default	Access
15-0	**PIP Window Width** REG[39h] bit[3:0] mapping to PWW[11:8], bit[7:4] are not used. REG[38h] mapping to PWW[7:0] Unit: Pixel. Maximum value is2,048 pixels. This setting is relative to the PIP settings in the register of REG[10h].	0	RW

REG[3Bh-3Ah] PIP Window 1 or 2 Height (PWH)

Bit	Description	Default	Access
15-0	**PIP Window Height** REG[3Bh] bit[3:0] mapping to PWH[11:8], bit[7:4] are not used. REG[3Ah] mapping to PWH[7:0] Unit: Pixel. The value is physical pixel number and maximum value is 2,048 pixels. This setting is relative to the PIP settings in the register of REG[10h].	0	RW

Note 1: The resolution of Window Size and Starting Position in the horizontal direction is 1 pixel, the vertical resolution is 1 line.

Note 2: The above registers REG[20h] ~ REG[3Bh] need to be written from LSB to MSB in turn. Suppose we need to set the Main Image Start Address, this relative register are REG[20h] to REG[23h]. Then Host must write Address data from LSB (REG[20h]) to MSB (REG[23h]) in turn. When REG[23h] is written, LT758x will then write the complete Main Image Start Address to the internal register.

REG[3Ch] Graphic / Text Cursor Control Register (GTCCR)

Bit	Description	Default	Access
7	**Gamma Correction Enable** 0: Disable 1: Enable Gamma correction is the last output stage.	0	RW
6-5	**Gamma Table Select for MCU Write Gamma Data** 00b: Gamma table for Blue 01b: Gamma table for Green 10b: Gramma table for Red 11b: Reserved	0	RW
4	**Graphic Cursor Enable** 0: Graphic Cursor Disable 1: Graphic Cursor Enable Graphic cursor will be disable if VDIR (REG[12h] bit3) is set to 1.	0	RW
3-2	**Graphic Cursor Selection** To set a 32x32 Graphic Cursor, select one from the below four graphic cursors 00b: Graphic Cursor Set 1。 01b: Graphic Cursor Set 2。 10b: Graphic Cursor Set 3。 11b: Graphic Cursor Set 4。 Since the 64x64 Graphic Cursor is composed by the 4 sets of 32x32 Graphic cursor, when using the 64x64 Graphic Cursor, these two bits will be invalid.	0	RW

Bit	Description	Default	Access
1	**Text Cursor Enable** 0：Disable 1：Enable Text cursor & Graphic cursor cannot be enabled simultaneously. Graphic cursor has higher priority than Text cursor if enabled simultaneously.	0	RW
0	**Text Cursor Blinking Enable** 0：Disable 1：Enable	0	RW

REG[3Dh] Blink Time Control Register (BTCR)

Bit	Description	Default	Access
7-0	**Text Cursor Blink Time Setting** 00h：1 Frame cycle time 01h：2 Frames cycle time 02h：3 Frames cycle time ： ： FFh：256 frames cycle time	0	RW

REG[3Eh] Text Cursor Horizontal Size Register (CURHS)

Bit	Description	Default	Access
7-5	Reserved	0	RO
4-0	**Text Cursor Horizontal Size Setting** 00000b：1 Pixel 00001b：2 Pixels ： ： 11111b：32 Pixels Unit: Pixel. When the character is enlarged, the cursor will be enlarged at the same time.	07h	RW

REG[3Fh] Text Cursor Vertical Size Register (CURVS)

Bit	Description	Default	Access
7-5	Reserved	0	RO
4-0	**Text Cursor Vertical Size Setting** Unit: Pixel. When the character is enlarged, the cursor will be enlarged at the same time.	0	RW

REG[41h-40h] Graphic Cursor Horizontal Position Register (GCHP)

Bit	Description	Default	Access
15-0	**Graphic Cursor Horizontal Position** REG[41h] bit[2:0] mapping to GCHP[10:8], bit[7:3] are not used. REG[40h] mapping to GCHP[7:0]	0	RW

REG[43h-42h] Graphic Cursor Vertical Position Register (GCVP)

Bit	Description	Default	Access
15-0	**Graphic Cursor Vertical Position** REG[43h] bit[2:0] mapping to GCVP[10:8], bit[7:3] are not used REG[42h] mapping to GCVP[7:0]	0	RW

REG[44h] Graphic Cursor Color 0 (GCC0)

Bit	Description	Default	Access
7-0	**Graphic Cursor Color 0 with 256 Colors** RGB Format [7:0] = RRRGGGBB.	0	RW

REG[45h] Graphic Cursor Color 1 (GCC1)

Bit	Description	Default	Access
7-0	**Graphic Cursor Color 1 with 256 Colors** RGB Format [7:0] = RRRGGGBB.	0	RW

REG[46h] Main & PIP Raster Operation (PIP_ROP)

Bit	Description	Default	Access
7-4	**Main & PIP1 ROP** S0 represents Main, S1 represents PIP1 Refer to Table 17-3 for the ROP command list.	0	RW
3-0	**Main & PIP2 ROP** S0 represents Main, S1 represents PIP2 Refer to Table 17-3 for the ROP command list.	0	RW

Table 17-3: Main & PIPn ROP Command List

Bit[7:4] 或 Bit[3:0]	Exclusive Display GC > TC > PIP1 > PIP2 > Main
0001b	~S0・~S1 or ~(S0+S1)
0010b	~S0・S1
0011b	~S0
0100b	S0・~S1
0101b	~S1
0110b	S0^S1
0111b	~S0+~S1 or ~(S0・S1)
1000b	S0・S1
1001b	~ (S0^S1)
1010b	0 (Blackness)
1011b	~S0+S1
1100b	S0
1101b	S0+~S1
1110b	S0+S1
1111b	1 (Whiteness)

REG[47h] Main & PIP Vertical to Horizontal (xxxx_V2H)

Bit	Description	Default	Access
7	**Enable Main Layer Width_Height Setting** (MDWMDH_EN) 0: Display to the panel boundary 1: Display range is controlled by MDW / MDH	0	RW
6	**PIP1 256 Grayshade Mode (PIP1 Gray 256)** For PIP1 8bpp : 0: 8bpp color mode (RGB332 or RGB256 color index mode) 1: 8bpp grayshade mode (256 grayshade mode) - Index color priority is higher than 256 grayshade mode	0	RW
5	**PIP2 256 Grayshade Mode (PIP2 Gray 256)** For PIP1 8bpp : 0: 8bpp color mode (RGB332 or RGB256 color index mode) 1: 8bpp grayshade mode (256 grayshade mode) - Index color priority is higher than 256 grayshade mode	0	RW
4	Reserved	-	-
3	Reserved	-	-
2	Reserved	-	-
1	Reserved	-	-
0	Reserved	-	-

REG[49h-48h] Main Display Window Height (MDH)

Bit	Description	Default	Access
15-0	**Main Display Height** REG[49h] bit[5:0] mapping to MDH[13:8], bit[7:6] are not used. REG[48h] mapping to MDH[7:0] Unit: Pixel. The setting of this register represents the LCD height. The maximum value is 8,192 pixels.	00h	RW

REG[4Bh-4Ah] Main Display Window Width (MDW)

Bit	Description	Default	Access
15-0	**Main Display Width** REG[4Bh] bit[5:0] mapping to MDW[13:8], bit[7:6] are not used. REG[4Ah] mapping to MDW[7:0] Unit: Pixel. The setting of this register represents the LCD width. The maximum value is 8,192 pixels.	00h	RW

REG[4Dh-4Ch] Main Display Window Upper-Left Corner X-Coordinates (MIULX)

Bit	Description	Default	Access
15-0	**Main Display Window Upper-Left Corner X-Coordinates** REG[4Dh] bit[2:0] mapping to MDULX [10:8], bit[7:3] are not used. REG[4Ch] mapping to MDULX[7:0] Unit: Pixel. The coordinate should be 0~2047.	0	RW

REG[4Fh-4Eh] Main Display Window Upper-Left corner Y-Coordinates (MIULY)

Bit	Description	Default	Access
15-0	**Main Display Window Upper-Left Corner Y-Coordinates** REG[4Fh] bit[2:0] mapping to MDULY [10:8], bit[7:3] are not used. REG[4Eh] mapping to MDULY[7:0] Unit: Pixel. The coordinate should be 0~2047.	0	RW

17.6 Geometric Engine Control Registers

REG[53h-50h] Canvas Start Address (CVSSA)

Bit	Description	Default	Access
31-0	**Start Address of Canvas** REG[53h] mapping to CVSSA[31:24] REG[52h] mapping to CVSSA[23:16] REG[51h] mapping to CVSSA[15:8] REG[50h] mapping to CVSSA[7:0], bit[1:0] must be 0. These registers will be ignored if the canvas is in Linear Addressing mode.	0	RW

REG[55h-54h] Canvas Image Width (CVS_IMWTH)

Bit	Description	Default	Access
15-0	**Canvas Image Width** REG[55h] bit[5:0] mapping to CVS_IMWTH[13:8], bit[7:6] are not used. REG[54h] mapping to CVS_IMWTH[7:0] Width = Real Image Width; Max = 8192 **Note:** The width must be a multiple of 4 when the color depth = 8bpp or 24bpp; the width must be even when the color depth = 16bpp; and the width can be any value when the color depth = 32bpp. That is, if bpp={0, 1, 2, 3} represents 8bpp, 16bpp, 24bpp & 32bpp, the width condition: (width * (bpp+1)) % 4 == 0 These registers will be ignored if the canvas is in Linear Addressing mode.	0	RW

Note: The unit of REG[54h] ~ REG[5Dh] is Pixel.

REG[57h-56h] Active Window Upper-Left Corner X-Coordinates (AWUL_X)

Bit	Description	Default	Access
15-0	**Active Window Upper-Left Corner X-Coordinates** REG[57h] bit[4:0] mapping to AWUL_X[12:8], bit[7:5] are not used. REG[56h] mapping to AWUL_X[7:0] Unit: Pixel. The sum of X-axis coordinates and Active Window width cannot be larger than 8,191. These registers will be ignored if the canvas is in Linear Addressing mode.	0	RW

REG[59h-58h] Active Window Upper-Left Corner Y-Coordinates (AWUL_Y)

Bit	Description	Default	Access
15-0	**Active Window Upper-Left Corner Y-Coordinates** REG[59h] bit[4:0] mappting to AWUL_Y[12:8], bit[7:5] are not used. REG[58h] mapping to AWUL_Y[7:0] Unit: Pixel. The sum of Y-axis coordinates and Active Window height cannot be larger than 8,191. These Registers will be ignored if canvas is in linear Addressing mode.	0	RW

REG[5Bh-5Ah] Active Window Width (AW_WTH)

Bit	Description	Default	Access
15-0	**Active Window Width [13:8]** REG[5Bh] bit[5:0] mapping to AW_WTH[13:8], bit[7:6] are not used. REG[5Ah] mapping to AW_WTH[7:0] Unit: Pixel. The value is physical pixel width. Maximum pixel width is 8,192. These Registers will be ignored if canvas is in linear Addressing mode.	0	RW

REG[5Dh-5Ch] Active Window Height (AW_HT)

Bit	Description	Default	Access
15-0	**Height of Active Window [13:8]** REG[5Dh] bit[5:0] mapping to AW_HT[13:8], bit[7:6] are not used. REG[5Ch] mapping to AW_HT[7:0] Unit: Pixel. The value is physical pixel height. Maximum pixel width is 8,192. These Registers will be ignored if canvas is in linear Addressing mode.	0	RW

REG[5Eh] Color Depth of Canvas & Active Window (AW_COLOR)

Bit	Description	Default	Access
7-4	Reserved	0	RO
3	**Select What will Read Back from Graphic Read/Write Position Register** 0: Read back Graphic Write position 1: Read back Graphic Read position (Pre-fetch Address)	0	RW
2	**Canvas Addressing Mode** 0: Block mode (X-Y coordinates addressing) 1: Linear mode	0	RW
1-0	**Canvas Image's Color Depth & Memory R/W Data Width** **In Block Mode:** 00b: 8bpp 01b: 16bpp 10b: 24bpp 11b: 32bpp **In Linear Mode:** x0b: 8bits memory data read/write x1b: 16bits memory data read/write	0	RW

Note: Refer to Figure 8-3 Canvas Window and Active Window

REG[60h-5Fh] Graphic Read/Write X-Coordinate Register (CURH)

Bit	Descritpion	Default	Access
15-0	**Write: Set Graphic Read/Write X-Coordinate CURH[12:0]** **Read: Read Graphic Read/Write X-Coordinate CURH[12:0]** To decide whether it is "Read position" or "Write position", it depends on the setting of REG[5Eh] bit3. REG[60h] bit[4:0] mapping to CURH[12:8] bit[7:5] are not used. REG[5Fh] mapping to CURH[7:0] **When DPRAM in Linear Mode:** Memory Read/Wdrite adress [15:0], Unit: Byte. **When DPRAM in Block Mode:** Graphic Read/Write X-Coordinate [12:0], Unit: Pixel.	0	RW

Note: MCU should program proper active window related parameters before configuring this register.

REG[62h-61h] Graphic Read/Write Y-Coordinate Register (CURV)

Bit	Description	Default	Access
15-0	**Write: Set Graphic Read/Write Y-Coordinate CURV[12:0]** **Read: Read Graphic Read/Write Y-Coordinate CURV[12:0]** To decide whether it is "Read position" or "Write position", it depends on the setting of REG[5Eh] bits. REG[62h] bit[4:0] mapping to CURV[12:8], bit[7:5] are not used. REG[61h] mapping to CURV[7:0] **When DPRAM In Linear Mode:** Memory Read/Write address [31:16], Unit: Byte. **When DPRAM In Block Mode:** Graphic Read/Write Y-Coordinate [12:0], Unit: Pixel.	0	RW

Note: MCU should program proper active window related parameters before configuring this register.

REG[64h-63h] Text Write X-Coordinates Register (F_CURX)

Bit	Description	Default	Access
15-0	**Text Write X-Coordinates F_CURX[12:0]** REG[64h] bit[4:0] mapping to F_CURX[12:8], bit[7:5] are not used. REG[63h] mapping to F_CURX[7:0] Write: Set the X coordinate of the text to be written Read: Read the X coordinate of the written text	0	RO WO

REG[66h-65h] Text Write Y-Coordinates Register (F_CURY)

Bit	Description	Default	Access
15-0	**Text Write Y-Coordinates F_CURY[12:0]** REG[66h] bit[4:0] mapping to F_CURY[12:8], bit[7:5] are not used. REG[65h] mapping to F_CURY[7:0] Write: Set the Y coordinate of the text to be written Read: Read the Y coordinate of the written text	0	RO WO

REG[67h] Draw Line/Triangle Control Register 0 (DCR0)

Bit	Description	Default	Access
7	**Draw Line / Triangle Start Control** **Write:** 0: Stop Drawing 1: Start Drawing **Read:** 0: Drawing is done 1: Drawing is in process	0	RW
6	Reserved	0	RO
5	**Fill Function for Triangle** 0: Not-filled 1: Fill	0	RW
4-1	**Draw Triangle or Line Select** 0000b: Draw Line 0001b: Draw Triangle 0010b: Rectangle 0011b: Quadrilateral 0100b: Pentagon 0101b: Polyline (3EP) 0110b: Polyline (4EP) 0111b: Polyline (5EP) 1000b: Ellipse 1001b: Rounded-Rectangle 1010b, 1011b: Reserved 1100b: Oval Arc on upper-right/1st Quadrant 1101b: Oval Arc on upper-left /2nd Quadrant 1110b: Oval Arc on Lower-Left /3rd Quadrant 1111b: Oval Arc on Lower-Right/4th Quadrant	0	RW
0	**Polyline Style** 0: Open-end Polyline 1: Closed Polyline (connect the last point to the start point)	0	RW

REG[69h-68h] Draw Line/Rectangle/Triangle Point 1 X-Coordinates Register (DLHSR)

Bit	Description	Default	Access
15-0	**Draw Line/Rectangle/Triangle Point 1 X-Coordinates DLHSR[12:0]** REG[69h] bit[4:0] mapping to DLHSR[12:8], bit[7:5] are not used. REG[68h] mapping to DLHSR[7:0] **Note:** When drawing a rectangle, start point & end point cannot be located at the same point or at the same X-Coordinates or Y-Coordinates.	0	RW

Note: The unit is Pixel for the values set in REG[68h] ~ REG[72h] .

REG[6Bh-6Ah] Draw Line/Rectangle/Triangle Point 1 Y-Coordinates Register (DLVSR)

Bit	Description	Default	Access
15-0	**Draw Line/Rectangle/Triangle Point 1 Y-Coordinates DLVSR[12:0]** REG[6Bh] bit[4:0] mapping to DLVSR[12:8], bit[7:5] are not used. REG[6Ah] mapping to DLVSR[7:0]	0	RW

REG[6Dh-6Ch] Draw Line/Rectangle/Triangle Point 2 X-Coordinates Register (DLHER)

Bit	Description	Default	Access
15-0	**Draw Line/Rectangle/Triangle Point 2 X-Coordinates DLHER[12:0]** REG[6Dh] bit[4:0] mapping to DLHER[12:8], bit[7:5] are not used. REG[6Ch] mapping to DLHER[7:0]	0	RW

REG[6Fh-6Eh] Draw Line/Rectangle/Triangle Point 2 Y-Coordinates Register (DLVER)

Bit	Description	Default	Access
15-0	**Draw Line/Rectangle/Triangle Point 2 Y-Coordinates DLVER[12:0]** REG[6Fh] bit[4:0] mapping to DLVER[12:8], bit[7:5] are not used. REG[6Eh] mapping to DLVER[7:0]	0	RW

REG[71h-70h] Draw Triangle Point 3 X-Coordinates Register (DTPH)

Bit	Description	Default	Access
15-0	**Draw Triangle Point 3 X- Coordinates DTPH[12:0]** REG[71h] bit[4:0] mapping to DTPH[12:8], bit[7:5] are not used. REG[70h] mapping to DTPH[7:0]	0	RW

REG[73h-72h] Draw Triangle Point 3 Y-Coordinates Register (DTPV)

Bit	Description	Default	Access
15-0	**Draw Triangle Point 3 Y-Coordinates DTPV[12:0]** REG[73h] bit[4:0] mapping to DTPV[12:8], bit[7:5] are not used. REG[72h] mapping to DTPV[7:0]	0	RW

Note: When Drawing a triangle: If any two endpoints are overlapped, it will draw a line. If three endpoints are overlapped, it will draw a dot.

REG[74h~75h]: Reserved

REG[76h] Draw Circle/Ellipse/Ellipse Curve/Circle Square Control Register 1 (DCR1)

Bit	Description	Default	Access
7	**Draw Circle / Ellipse / Square /Circle Square Control** **Write Function** 0: Stop Drawing 1: Start Drawing **Read Function** 0: Drawing is done 1: Drawing is in process	0	RW
6	**Fill the Circle / Ellipse / Square / Circle Square Control** 0: Not-filled 1: Fill	0	RW
5-4	**Draw Circle / Ellipse / Square / Ellipse Curve / Circle Square Select** 00b: Draw Circle / Ellipse 01b: Draw Circle / Ellipse Curve 10b: Draw Square / Rectangle 11b: Draw Rounded-Rectangle	0	RW
3-2	Reserved	0	RO
1-0	**Draw Circle / Ellipse Curve Part Select, DECP** 00b: bottom-left Ellipse Curve 01b: upper-left Ellipse Curve 10b: upper-right Ellipse Curve 11b: bottom-right Ellipse Curve	0	RW

REG[78h-77h] Draw Circle/Ellipse/Rounded-Rectangle Major-Radius Register (ELL_A)

Bit	Description	Default	Access
15-0	**Draw Circle/Ellipse/Rounded-Rectangle Major-Radius ELL_A[12:0]** REG[77h] mapping to ELL_A[7:0] REG[78h] bit[4:0] mapping to ELL_A[12:8], bit[7:5] are not used. If drawing a circle, then the major radius must be equal to the minor radius (ELL_A = ELL_B).	0	RW

REG[7Ah-79h] Draw Circle/Ellipse/Rounded-Rectangle Minor-Radius Register (ELL_B)

Bit	Description	Default	Access
15-0	**Draw Circle/Ellipse/Rounded-Rectangle Minor-Radius ELL_B[12:0]** REG[79h] mapping to ELL_B[7:0] REG[7Ah] bit[4:0] mapping to ELL_B[12:8], bit[7:5] are not used.	0	RW

REG[7Ch-7Bh] Draw Circle/Ellipse/Rounded-Rectangle Center X-Coordinates Register (DEHR)

Bit	Description	Default	Access
15-0	**Draw Circle/Ellipse/Rounded-Rectangle Center X-Coordinates DEHR[12:0]** REG[7Bh] mapping to DEHR[7:0] REG[7Ch] bit[4:0] mapping to DEHR[12:8], bit[7:5] are not used.	0	RW

REG[7Eh-7Dh] Draw Circle/Ellipse/Rounded-Rectangle Center Y-Coordinates Register (DEVR)

Bit	Description	Default	Access
15-0	**Draw Circle/Ellipse/Rounded-Rectangle Center Y-Coordinates DEVR[12:0]** REG[7Dh] mapping to DEVR[7:0] REG[7Eh] bit[4:0] mapping to DEVR[12:8], bit[7:5] are not used.	0	RW

Note: The unit is Pixel for the values set in REG[77h] ~ REG[7Eh].

REG[7Fh]: Reserved

REG[D2h] Foreground Color Register - Red (FGCR)

Bit	Description	Default	Access
7-0	**Foreground Color – Red (for Graphic mode, Text mode, and Color Expansion mode)** 256 Colors: Red mapping to bit[7:5] 65K Colors: Red mapping to bit[7:3] 16.7M Colors: Red mapping to bit[7:0]	FFh	RW

REG[D3h] Foreground Color Register - Green (FGCG)

Bit	Description	Default	Access
7-0	**Foreground Color – Green (for Graphic mode, Text mode, and Color Expansion mode)** 256 Colors: Green mapping to bit[7:5] 65K Colors: Green mapping to bit[7:2] 16.7M Colors: Green mapping to bit[7:0]	FFh	RW

REG[D4h] Foreground Color Register - Blue (FGCB)

Bit	Description	Default	Access
7-0	**Foreground Color – Blue (for Graphic mode, Text mode, and Color Expansion mode)** 256 Colors: Blue mapping to bit[7:6] 65K Colors: Blue mapping to bit[7:3] 16.7M Colors: Blue mapping to bit[7:0]	FFh	RW

Note: For Background color setting, please refer to the Text Engine Registers REG[D5h–D7h].

REG[D8h] Foreground Alpha Color (FGCA)

Bit	Description	Default	Access
7-0	**Foreground Color – Alpha** Using 32bpp data format 0: Trasparent 1~254: Display by the transparent ratio 255: Non-transparent	FFh	RW

REG[D9h] Background Alpha Color (BGCA)

Bit	Description	Default	Access
7-0	**Background Color – Alpha** Using 32bpp data format 0: Trasparent 1~254: Display by the transparent ratio 255: Non-transparent	FFh	RW

17.7 PWM Control Registers

REG[84h] PWM Prescaler Register (PSCLR)

Bit	Description	Default	Access
7-0	**PWM Prescaler Register** This register determine the pre-scaler value for Timer 0 and 1. The Base Frequency is: Core_Freq / (Prescaler + 1)	0	RW

REG[85h] PWM Clock Mux Register (PMUXR)

Bit	Description	Default	Access
7-6	**PWM Timer-1 (Select 2nd Clock Divider's MUX Input for PWM Timer-1)** 00b = 1 01b = 1/2 10b = 1/4 11b = 1/8	0	RW
5-4	**PWM Timer-0 (Select 2nd Clock Divider's MUX Input for PWM Timer-0)** 00b = 1 01b = 1/2 10b = 1/4 11b = 1/8	0	RW
3-2	**PWM[1] Function Control** 0xb: PWM[1] output system error flag (Scan FIFO POP error or Memory access out of range) 10b: PWM[1] output PWM timer 1 waveform or invert waveform of PWM timer 0 (dead zone enable) 11b: PWM[1] output Oscillator Clock If TEST[0] = 1, PWM[1] will be the input of the panel scanning frequency.	0	RW
1-0	**PWM[0] Function Control** 0xb: PWM[0] becomes GPIO-C[7] 10b: PWM[0] output PWM Timer 0 11b: PWM[0] output Core Clock (CCLK)	0	RW

REG[86h] PWM Configuration Register (PCFGR)

Bit	Description	Default	Access
7	Reserved -- The value must be 0	0	RW
6	**PWM Timer-1 Output Inverter On/Off** Determine the output inverter to be On or Off for PWM1. 0 = Inverter off 1 = Inverter on for PWM1	0	RW
5	**PWM Timer-1 Auto Reload On/Off** Determine auto reload on/off for Timer 1. 0: One-Shot Mode 1: Interval Mode (Auto Reload)	1	RW
4	**PWM Timer-1 Start/Stop** 0: Stop 1: Start In Interval Mode, the MCU needs to program it as 0 to stop PWM timer. In One-shot Mode, this bit will be auto cleared. The MCU may read this bit to find out if the current PWMx is running or stopped.	0	RW
3	**PWM Timer-0 Dead Zone Enable** 0: Disable 1: Enable	0	RW
2	**PWM Timer-0 Output Inverter On/Off** Determine the output inverter to be On or Off for PWM0. 0 = Inverter off 1 = Inverter on for PWM0	0	RW
1	**PWM Timer- 0 Auto Reload On/Off** Determine auto reload on/off for Timer 0. 0: One-Shot Mode 1: Interval Mode (Auto Reload)	1	RW
0	**PWM Timer-0 Start/Stop** 0: Stop 1: Start In Interval Mode, the MCU needs to program it as 0 to stop PWM timer. In One-shot Mode, this bit will be auto cleared. The MCU may read this bit to find out if the current PWMx is running or stopped.	0	RW

REG[87h] Timer-0 Dead Zone Length Register [DZ_LENGTH]

Bit	Description	Default	Access
7-0	**Timer-0 Dead Zone Length Register** These 8 bits determine the dead zone length. One unit time of the dead zone length is equal to one complete counting cycle of Timer 0.	0	RW

REG[89h-88h] Timer-0 Compare Buffer Register [TCMPB0]

Bit	Description	Default	Access
15-0	**Timer-0 compare Buffer Register** REG[89h] mapping to TCMPB0 [15:8]。 REG[88h] mapping to TCMPB0 [7:0]。 The Timer-0 Compare Buffer Registers have a total of 16bits. When the counter value is equal to or less than the value of this register, and the output inverter of PWM0 is off, then PWM0 output is high level.	0	RW

REG[8Bh-8Ah] Timer-0 Count Buffer Register [TCNTB0]

Bit	Description	Default	Access
15-0	**Timer-0 Count Buffer Register [15:0]** REG[8Bh] mapping to TCNTB0 [15:8]。 REG[8Ah] mapping to TCNTB0 [7:0]。 The Timer-0 count registers have a total of 16bit. When the counter is equal to 0 and the Reload_EN is enabled, the PWM will reload the value of this register to the counter. When the PWM begins to count, the current count value can be read back through this register.	0	RW

REG[8Dh-8Ch] Timer-1 Compare Buffer Register [TCMPB1]

Bit	Description	Default	Access
15-0	**Timer-1 compare Buffer Register** REG[8Dh] mapping to TCMPB1 [15:8]。 REG[8Ch] mapping to TCMPB1 [7:0]。 The Timer-1 Compare Buffer Registers have a total of 16bits. When the counter value is equal to or less than the value of this register, and the output inverter of PWM1 is off, then PWM1 output is high level.	0	RW

REG[8Fh-8Eh] Timer-1 Count Buffer Register [TCNTB1]

Bit	Description	Default	Access
15-0	**Timer-1 Count Buffer Register [15:0]** REG[8Fh] mapping to TCNTB1 [15:8]。 REG[8Eh] mapping to TCNTB1 [7:0]。 The Timer-1 count registers have a total of 16bit. When the counter is equal to 0 and the Reload_EN is enabled, the PWM will reload the value of this register to the counter. When the PWM begins to count, the current count value can be read back through this register.	0	RW

17.8 BitBlock Transfer Engine (BTE) Control Registers

REG[90h] BitBLT Function Control Register 0 (BLT_CTRL0)

Bit	Description	Default	Access
7-5	Reserved	0	RO
4	**BTE Function Enable / Status** **Write** 0: No operation 1: BTE Enable **Read** 0: BTE is idle 1: BTE is busy When the BTE is enabled, the MCU is not allowed to access the memory of the canvas[active window] area.	0	RW
3-1	Reserved	0	RO
0	**Pattern Format** 0: 8*8 1: 16*16	0	RW

REG[91h] BitBLT Function Control Register1 (BLT_CTRL1)

<table>
<tr><th>Bit</th><th>Description</th><th>Default</th><th>Access</th></tr>
<tr><td>7-4</td><td>
BTE ROP Code or Color Expansion Starting
ROP stands for Raster Operation. Some of the BTE functions can collocate with ROP for advanced functions.

Table 17-4: ROP Code of BTE
<table>
<tr><th>Bit[7:4]</th><th>Description</th></tr>
<tr><td>0000b</td><td>0 (Blackness)</td></tr>
<tr><td>0001b</td><td>~S0 · ~S1 or ~(S0+S1)</td></tr>
<tr><td>0010b</td><td>~S0 · S1</td></tr>
<tr><td>0011b</td><td>~S0</td></tr>
<tr><td>0100b</td><td>S0 · ~S1</td></tr>
<tr><td>0101b</td><td>~S1</td></tr>
<tr><td>0110b</td><td>S0^S1</td></tr>
<tr><td>0111b</td><td>~S0+~S1 or ~(S0 · S1)</td></tr>
<tr><td>1000b</td><td>S0 · S1</td></tr>
<tr><td>1001b</td><td>~(S0^S1)</td></tr>
<tr><td>1010b</td><td>S1</td></tr>
<tr><td>1011b</td><td>~S0+S1</td></tr>
<tr><td>1100b</td><td>S0</td></tr>
<tr><td>1101b</td><td>S0+~S1</td></tr>
<tr><td>1110b</td><td>S0+S1</td></tr>
<tr><td>1111b</td><td>1 (Whiteness)</td></tr>
</table>
If BTE operation code is set to 8h / 9h / Eh / Fh (i.e. Color Expansion operation), then these bits stand for the starting bit on BTE window left boundary, and it is relative to the MCU interface. For 8-bits Host, the value should be within 0 to 7. For 16-bits Host, the value should be within 0 to 15.
</td><td>0</td><td>RW</td></tr>
<tr><td>3-0</td><td>
BTE Operation Code bit[3:0]
LT758x has an embedded a 2D BTE Engine, it can execute 13 BTE functions. Some of the BTE functions can collocate with the ROP code for advanced functions.
</td><td>0</td><td>RW</td></tr>
</table>

Table 17-5：BTE Operation Code

REG[91h] Bit[3:0]	Description
0000b	**MCU Write with ROP** S0：Data comes from the MCU S1：Data comes from the SDRAM D：Write to the SDRAM based on the designated ROP function
0001b	Reserved
0010b	**Memory Copy with ROP** S0：Data comes from the SDRAM S1：Data comes from the SDRAM D：Write the SDRAM based on the designated ROP function
0011b	Reserved
0100b	**MCU Write w/ Chroma Keying（w/o ROP)** S0：Data comes form the MCU If the data from the MCU is not the same as the Chroma key (set in the Background Color Register), then the data will be written to the destination memory (SDRAM).
0101b	**Memory Copy（move）w/ Chroma keying（w/o ROP)** S0：Data comes from the SDRAM (S1 is not required) If the S0 data is not the same as the Chroma key (set in the Background Color Register), then the data will be written to the destination memory (SDRAM).
0110b	**Pattern Fill with ROP** S0: Pattern stored in the memory (SDRAM)
0111b	**Pattern Fill with Chroma Keying** S0: Pattern stored in the memory (SDRAM) If the S0 data is not the same as the Chroma key (set in the Background Color Register), then the data will be written to the destination memory (SDRAM).
1000b	**MCU Write w/ Color Expansion** S0: Data comes from the MCU BTE converts the data to the specified color and color depth, and then writes the result to the desitnation memory (SDRAM).
1001b	**MCU Write w/ Color Expansion and Chroma Keying** The monochrome data required by S0 is written by the MCU. If the bit of monochrome data is 1, the processed data is a foreground color, and if the mochrome data is 0, it will not be written. The data written to the desination memory will also reference the color depth setting.
1010b	**Memory Copy with Opacity** S0, S1 & D：Data all come from the memory (SDRAM)

REG[91h] Bit[3:0]	Description
1011b	**MCU Write with Opacity** S0：Data comes from the MCU S1：Data comes form the SDRAM D：Refer to the alpha blending operation and write to the destination memory.
1100b	**Solid Fill** The written value is the register setting value, and the written target is the destination memory.
1101b	Reserved
1110b	**Memory Copy with Color Expansion** S0 and D are in memory and S1 is not in use. S0 must be loaded through the MCU's write or DMA to preload 8bpp or 16bpp of color depth into memory, i.e. the S0 color depth should follow that color depth.
1111b	**Memory Copy with Color Expansion and Chroma Keying** S0 and D are in memory and S1 is not in use. S0 must be loaded through the MCU's write or DMA to preload 8bpp or 16bpp of color depth into memory, i.e. the S0 color depth should follow that color depth. If the S0 data bit = 0, then no data will be written to D. If the S0 data bit = 1, the Foreground Color data will be written to D.

REG[92h] Source 0/1 & Destination Color Depth (BLT_COLR)

Bit	Description	Default	Access
7	Reserved	0	RO
6-5	**S0 Color Depth** 00b：256 Color（8bpp） 01b：64k Color（16bpp） 1xb：16M Color（24bpp）	0	RW
4-2	**S1 Color Depth** 000b：256 Color（8bpp） 001b：64k Color（16bpp） 010b：16M Color（24bpp） 011b：Constant Color（S1 memory start address setting must be defined as S1 constant color definition）definition） 100b：8 bit Pixel Alpha Blending（αIndex 2:6） 101b：16 bit Pixel Alpha Blending（αRGB 4:4:4:4） **111b：32 bit Pixel Alpha Blending**（αRGB 8:8:8:8）	0	RW

Bit	Description	Default	Access
1-0	**Destination Color Depth** 00b：256 Color（8bpp） 01b：64k Color（16bpp） 1xb：16M Color（24bpp）	0	RW

REG[96h-93h] Source 0 Memory Start Address（S0_STR）

Bit	Description	Default	Access
31-0	**Source 0 Memory Start Address [31:2]** REG[96h] mapping to S0_STR [31:24] REG[95h] mapping to S0_STR [23:16] REG[94h] mapping to S0_STR [15:8] REG[93h] mapping to S0_STR [7:2] **Note：** REG[93h] bit[1:0] is fixed to 0	0	RW

REG[98h-97h] Source 0 Image Width（S0_WTH）

Bit	Description	Default	Access
15-0	**Source 0 Image Width [13:2]** REG[98h] bit[5:0] mapping to S0_WTH [13:8], bit[7-6] are not used. REG[97h] mapping to S0_WTH [7:2] **Note:** It must be divisible by 4, and REG[97h] bit[1:0] must be fixed to 0. Unit: Pixel.	0	RW

REG[9Ah-99h] Source 0 Window Upper-Left Corner X-Coordinates（S0_X）

Bit	Description	Default	Access
15-0	**Source 0 Window Upper-Left Corner X-Coordinates [12:0]** REG[9Ah] bit[4:0] mapping to S0_X [12:8], bit[7-5] are not used. REG[99h] mapping to S0_X [7:0]	0	RW

REG[9Ch-9Bh] Source 0 Window Upper-Left corner Y-Coordinates（S0_Y）

Bit	Description	Default	Access
15-0	**Source 0 Window Upper-Left Corner Y-Coordinates [12:0]** REG[9Ch] bit[4:0] mapping to S0_Y[12:8], bit[7-5] are not used. REG[9Bh] mapping to S0_Y [7:0]	0	RW

REG[9Dh-A0h] Source 1 Memory Start Address 0 (S1_STR)

Bit	Description	Default	Access
31-0	**Source 1 Memory Start Address [31:2]** REG[9Dh] mapping to S1_STR[7:0] / RED REG[9Eh] mapping to S1_STR[15:8] / GREEN REG[9Fh] mapping to S1_STR[23:16] / BLUE REG[A0h] mapping to S1_STR[31:24] / NA **Note 1:** REG[9Dh] bit[1:0] must be fixed 0 **Note 2:** If source 1 (S1) is set as Constant Color then S1 memory start address' setting will be defined as S1 constant color definition: ● REG[9Dh] is RED element (S1_RED) ; ● REG[9Eh] is GREEN element (S1_GREEN) ; ● REG[9Fh] is BLUE element (S1_BLUE) ; ● REG[A0h] does not include color element	0	RW

REG[A2h-A1h] Source 1 Image Width (S1_WTH)

Bit	Description	Default	Access
15-0	**Source 1 Image Width [12:2])** REG[A2h] bit[5:0] mapping to S1_WTH[13:8], bit[7:6] are not used REG[A1h] [7:2] mapping to S1_WTH [7:2] **Note:** It must be divisible by 4, and REG[A1h] bit[1:0] must be fixed to 0. Unit: Pixel.	0	RW

REG[A4h-A3h] Source 1 Window Upper-Left Corner X-Coordinates (S1_X)

Bit	Description	Default	Access
15-0	**Source 1 Window Upper-Left Corner X-Coordinates [12:0]** REG[A4h] bit[4:0] mapping to S1_X [12:8], bit[7:5] are not used REG[A3h] mapping to S1_X [7:0]	0	RW

REG[A6h-A5h] Source 1 Window Upper-Left corner Y-Coordinates (S1_Y)

Bit	Description	Default	Access
15-0	**Source 1 Window Upper-Left Corner Y-Coordinates [12:0]** REG[A6h] bit[4:0] mapping to S1_Y [12:8], bit[7:5] are not used. REG[A5h] mapping to S1_Y [7:0]	0	RW

REG[A7h-AAh] Destination Memory Start Address (DT_STR)

Bit	Description	Default	Access
31-0	**Destination Memory Start Address [31:2]** REG[AAh] mapping to DT_STR [31:24] REG[A9h] mapping to DT_STR [23:16] REG[A8h] mapping to DT_STR [15:8] REG[A7h] bit[7:2] mapping to DT_STR [7:2] **Note:** REG[A7h] bit[1:0] must be fixed to 0	0	RW

Note: The destination memory start address cannot be within the source 0 or Source 1 block, otherwise the output will be incorrect.

((Image_Width) * (Image_Height) * ([1|2|3]Color Depth))

REG[ACh-ABh] Destination Image Width (DT_WTH)

Bit	Description	Default	Access
15-0	**Destination Image Width [12:2]** REG[ACh] bit[5:0] mapping to DT _WTH [13:8], bit[7:5] are not used REG[ABh] mapping to DT _WTH [7:2] **Note:** It must be divisible by 4. REG[ABh] bit[1:0] must be fixed to 0. Unit: Pixel.	0	RW

REG[AEh-ADh] Destination Window Upper-Left Corner X-Coordinates (DT_X)

Bit	Description	Default	Access
15-0	**Destination Window Upper-Left Corner X-Coordinates [12:0]** REG[AEh] bit[4:0] mapping to DT_X [12:8], bit[7-5] are not used. REG[ADh] mapping to DT_X [7:0]	0	RW

REG[B0h-AFh] Destination Window Upper-Left Corner Y-Coordinates (DT_Y)

Bit	Description	Default	Access
15-0	**Destination Window Upper-Left Corner Y-Coordinates [12:0]** REG[B0h] bit[4:0] mapping to DT_Y [12:8], bit[7-5] are not used. REG[AFh] mapping to DT_Y [7:0]	0	RW

REG[B2h-B1h] BitBLT Window Width (BLT _WTH)

Bit	Description	Default	Access
15-0	**BitBLT Window Width [12:0]** REG[B2h] bit[4:0] mapping to BLT_WTH [12:8], bit[7-5] are not used. REG[B1h] mapping to BLT_WTH [7:0] When all Image Fill (Pattern Fill) functions for BTE are enabled, the BTE window Width is ignored and automatically set to 8 or 16. Unit: Pixel.	0	RW

REG[B4h-B3h] BitBLT Window Height (BLT_HIG)

Bit	Description	Default	Access
15-0	**Destination Image Height [12:0]** REG[B4h] bit[4:0] mapping to BLT_HIG [12:8], bit[7-5] are not used. REG[B3h] mapping to BLT_HIG [7:0] When all Image Fill (Pattern Fill) functions for BTE are enabled, the BTE window Width is ignored and automatically set to 8 or 16. Unit: Pixel.	0	RW

REG[B5h] Alpha Blending (APB_CTRL)

Bit	Description	Default	Access
7-6	Reserved	0	RO
5-0	**Window Alpha Blending Effect for S0 & S1** The value of alpha in the color code ranges from 1.0 down to 0.0, where 1.0 represents a fully opaque color, and 0.0 represents a fully transparent color. 00h: 0 01h: 1/32 02h: 2/32 : : 1Eh: 30/32 1Fh: 31/32 2Xh: 1 **Output Effect** = [S0 image * (1 - Alpha Setting Value)] + (S1 Image * Alpha Setting Value)	0	RW

17.9 Serial Flash & SPI Master Control Register

REG[B6h] Serial Flash DMA Controller REG (DMA_CTRL)

Bit	Description	Default	Access
7-4	**The content of the bit[7-4] shows the decoding result of the JPG file stored in the SPI Flash.** 0: Succeeded Non-0: Failed --- The status will be renewed when decoding a new file. 1: Wrong File ID or Width/Height Over 8192 2~6: Reserved 7: JPG – Wrong Huffman bit Stream 8: JPG – File format Error 9: JPG – Wrong File ID 10: JPG – Wrong APP01 Contents 11: JPG – Wrong Q-table Size 12: JPG – Wrong SOF0 13: JPG – Invalid Huffman Table ID 14: JPG – Wrong SOS Format 15: JPG – Missing H-table or Q-table **Note:** If MCU writes 0 to REG[B6h] bit[0], the above error code will be cleared to 0.	NA	RO
3-2	**The content of bit[3-2] shows the color depth of the decoded JPG file stored in the SPI Flash** 00: 8bpp 01: 16bpp 10: 24bpp 11: 32bpp	NA	RO
1	**DMA Mode of the SPI Flash** 0: Direct DMA transimission, no JPG decoding. - The addressing mode (Block or Linear) for both SPI Flash and LT758x must be set as the same. 1: The JPG data stored in SPI Flash will be decoded first, then transferred to LT758x through DMA **(SFDMA_FDEC)** - The addressing mode of the SPI Flash is set to Linear mode; and the addressing mode of LT758x is set to Block mode.	0	RW

Bit	Description	Default	Access
0	**Write: DMA Start Bit** This bit can be written to 1 via the MCU, and the circuit will be automatically cleared to 0 right away. This bit cannot be used when writing characters. Therefore, if DMA is enabled, Text mode cannot be set for inputing character code. **Read Function: DMA Busy Check Bit** 0: Idle 1: Busy The DMA transmission of a Serial Flash must be in Graphic Mode, and it must first set the canvas destination location, destination width, color depth, and addressing mode in the Display RAM. **Note:** If MCU writes 0 to REG[B6h] bit[0], the above error code will be cleared to 0.	0	RW

REG[83h-80h] Media File Size (MDEC_FILESZ)

Bit	Description	Default	Access
31-0	**Return the size information of the decoded file** REG[83h] mapping to MDEC_FILESZ[31:24] REG[82h] mapping to MDEC_FILESZ[23:16] REG[81h] mapping to MDEC_FILESZ[15:8] REG[80h] mapping to MDEC_FILESZ[7:0]	0	RO

REG[B7h] Serial Flash Controller Register (SFL_CTRL)

Bit	Description	Default	Access
7	**Serial Flash Select** 0: Serial Flash 0 is selected 1: Serial Flash 1 is selected	0	RW
6	**Serial Flash Type** 0: Serial NOR Flash 1: Serial NAND Flash	0	RW
5	**Serial Flash Address Mode** 0: 24bits address mode 1: 32bits address mode To set 32bits address mode , the Flash command “Enter 4-Byte Mode” (B7h) must be written to the Flash through REG[B8h], and then set this bit to 1. MCU may also check this bit to find out whether the serial flash entered 32 bits address mode during initial display function.	0	RW

Bit	Description	Default	Access
4	**Set the source of the Upper-Left Coordinates of the DMA window** The content (upper-left coordinates) of REG[C0h ~ C3h] can be set to represent one of the below sources: 0：Destination window 1：Source window	0	RW
3-0	**Read Command Code & Behavior Selection** 0000b：1x Read Command 03h/13h Normal read command. The command and address are output through MOSI, and the data are received from MISO. No dummy cycles needed between the address and data. Applicable for NOR/NAND Flash. 0100b：1x Read Command 0Bh/0Ch Faster read command. The command and address are output through MOSI, and the data are received from MISO. LT758x will insert 8 dummy cycles between the address and data. Applicable for NOR/NAND Flash. 1000b：1x Read Command 1Bh Fastest read command. The command and address are output through MOSI, and the data are received from MISO. LT758x will insert 16 dummy cycles between the address and data. Applicable for NOR Flash. 0010b：2x Read Command 3Bh/3Ch Fast Read Dual Output command. The command and address are output through MOSI, and the data are received from MOSI and MISO. LT758x will insert 8 dummy cycles (Dual Mode 0) between the address and data. Applicable for NOR/NAND Flash. 0011b：2x Read Command BBh/BCh Fast Read Dual I/O command. The command are output from MOSI. The output address and received data are from MOSI and MISO. LT758x will insert 4 dummy cycles (Dual Mode 1) between the address and data. Applicable for NOR/NAND Flash. 0110b：4x Read Command 0Bh Faster Read **in QSPI Mode** command.The command, address and data are all transmitted through MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 2 dummy cycles between the address and data. Applicable for NOR Flash. **Note:** Before using Faster Read in QSPI mode command, the “Enable QSPI Mode command” (38h) must be input through SPIM.	0	R/W

Bit	Description	Default	Access
	0111b：4x Read Command 6Bh/6Ch Fast Read Quad Output command. The command and address are output from MOSI, and the data are received from MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 8 dummy cycles (Quad Mode 0) between the address and data. Applicable for NOR/NAND Flash. 1001b：4x Read Command EBh/ECh Fast Read Quad I/O command. The command is output from MOSI. The address and data are received from MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 4 dummy cycles (Quad Mode 1) between the address and data. The number of dummy cycles on the Flash side should be set accordingly by the Flash command, “Set Read Parameters” (C0h). Applicable for NOR/NAND Flash. 1010b：4x Read Command EBh/ECh Fast Read Quad I/O command. The command is output from MOSI. The address and data are received from MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 6 dummy cycles (Quad Mode 1) between the address and data. The number of dummy cycles on the Flash side should be set accordingly by the Flash command, “Set Read Parameters” (C0h). Applicable for NOR/NAND Flash. 1011b：4x Read Command EBh Fast Read Quad I/O **in QSPI Mode** command. The command, address, and data are transmitted through MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 6 dummy cycles (Quad Mode 2) between the address and data. Applicable for NOR Flash. **Note:** Before using Faster Read in QSPI mode command, the “Enable QSPI Mode command” (38h) must be input through SPIM. In addition, the number of dummy cycles on the Flash side should be set accordingly by the Flash command, “Set Read Parameters” (C0h). 1110b：4x Read Command EBh Fast Read Quad I/O **in QSPI Mode** command. The command, address, and data are transmitted through MOSI(XSIO0), MISO(XSIO1), XSIO2, and XSIO3. LT758x will insert 8 dummy cycles (Quad Mode 2) between the address and data. Applicable for NOR Flash.。 **Note:** Before using Faster Read in QSPI mode command, the “Enable QSPI Mode command” (38h) must be input through SPIM. In addition, the number of dummy cycles on the Flash side should be set		

Bit	Description	Default	Access
	accordingly by the Flash command, “Set Read Parameters” (C0h). **Note 1:** For NOR Flash, LT758x will adapt the proper “Read Command” based on the addressing mode (24bits or 32bits). For example, when REG[B7h] bit[3:0] is set as 1010b, then the Read Command will be EBh for 24bits addressing mode, and ECh for 32bits addressing mode. **Note 2:** Not all of the SPI Flash chips support the above commands. Users should check the Flash spec for the proper commands.		

REG[B8h] SPI Master Tx /Rx FIFO Data Register (SPIDR)

Bit	Description	Default	Access
7-0	**SPI Master Tx /Rx FIFO Data Register** After LT758x is correctly initialized, it can start transmitting data or commands through the SPI Master interface. A transfer is initiated by writing to the Serial Peripheral Data Register [SPIDR]. When the MCU wants to write the SPIDR register, it has to do so through the Write FIFO. Each write access adds a data byte to the Write FIFO. The writing procedure is: (1) Enable the SPI device (SS_ACTIVE); (2) Write the data if the Write FIFO is not full. Receiving data is done simultaneously with transmitting data; whenever a data byte is transmitted, a data byte is received. For each byte that needs to be read from a device, a dummy byte needs to be written to the Write FIFO. This instructs the core to initiate an SPI transfer, simultaneously transmitting the dummy byte and receiving the desired data. Whenever a transfer is finished, the received data byte is added to the Read FIFO. The Read FIFO is the counterpart of the Write FIFO. It is an independent 16 entries deep FIFO. The FIFO contents can be read by reading from the Serial Peripheral Data Register [SPIDR]	NA	RW

REG[B9h] SPI Master Control Register (SPIMCR2)

Bit	Description	Default	Access
7	**SPI NAND Bad Block Management Interrupt** 0: Disable 1: Enable	0	RW
6	**SPI Master Interrupt Enable** 0: Disable interrupt 1: Enable interrupt If the MCU disables SPI Master interrupt flag, then LT758x will not assert interrupt event to inform Host, but the MCU still may check interrupt event on SPIMSR.	0	RW
5	**Control Slave Select Drive on Which SFCS[0]# / SFCS[1]#** 0: Slave Select Signal (SS#) driven by SFCS[0]# 1: Slave Select Signal (SS#) driven by SFCS[1]#	0	RW

<table>
<tr><th>Bit</th><th>Description</th><th>Default</th><th>Access</th></tr>
<tr><td>4</td><td>Slave Select Signal Active [SS_ACTIVE]
0：Inactive （SS# outputs High signal）
1：Active （SS# outputs Low signal）
While SS_ACTIVE is set as Inactive, FIFO will be cleared and the engine will stay in idle state.
Note: It is suggested when SS_ACTIVE is set as Active, the settings of CPOL/CPHA should not be changed.</td><td>0</td><td>RW</td></tr>
<tr><td>3</td><td>Mask Interrupt for FIFO Overflow Error [OVFIRQMSK]
0：Unmask
1：Mask</td><td>1</td><td>RW</td></tr>
<tr><td>2</td><td>Mask Interrupt for While Tx FIFO Empty & SPI Engine/FSM Idle [EMTIRQMSK]
0：Unmask
1：Mask</td><td>1</td><td>RW</td></tr>
<tr><td>1:0</td><td>SPI Operation Mode
When DMA or external CGROM is enabled, only mode 0 & mode 3 will be supported.
Table 17-6： SPI Operation Mode
<table>
<tr><th>Mode</th><th>CPOL:
Clock Polarity Bit</th><th>CPHA:
Clock Phase Bit</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>2</td><td>1</td><td>0</td></tr>
<tr><td>3</td><td>1</td><td>1</td></tr>
</table></td><td>0</td><td>RW</td></tr>
</table>

■ Please refer to Section 13.1 for SPI Master description.

■ When CPOL = 0， SCK is 0 in inactive

_ CPHA = 0: data are read on the rising edge (Low->High), and changed on the falling edge (Hign->Low)

_ CPHA = 1: data are read on the falling edge (High->Low), and changed on the rising edge (Low->High)

■ When CPOL = 1， SCK is 1 inactive （inversion of CPOL = 0）

_ CPHA = 0: data are read on ghe falling edge (Hign->Low), and changed on the rising edge (Low->High)

_ CPHA = 1: data are read on the rising edge (Low->High), and changed on the falling edge (Hign->Low)

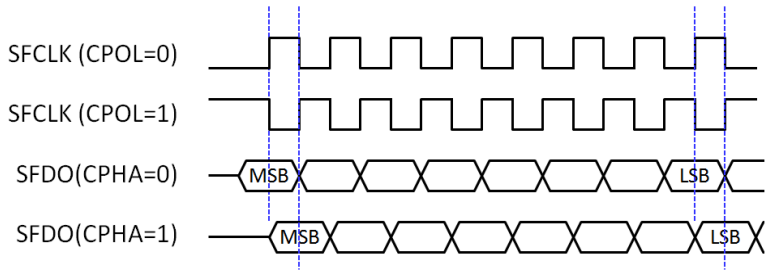


Figure 17-1： Master SPI Data Transmission

REG[BAh] SPI Master Status Register (SPIMSR)

Bit	Description	Default	Access
7	**Tx FIFO Empty Flag** 0: Not Empty 1: Empty	1	RO
6	**Tx FIFO Full Flag** 0: Not Full 1: Full	0	RO
5	**Rx FIFO Empty Flag** 0: Not Empty 1: Empty	1	RO
4	**Rx FIFO Full Flag** 0: Not Full 1: Full	0	RO
3	**Overflow Interrupt Flag** Write 1 will clear this flag	0	RW
2	**Tx FIFO Empty & SPI Engine/FSM Idle Interrupt Flag** Write 1 will clear this flag	0	RW
1	Reserved	0	RO
0	**Bad Block Management (BBM) Interrupt Flag** Writing any value to the Bad Block Switch Complete Register (REG[C4h]) will clear this flag.	0	RO

REG[BBh] SPI Clock Period (SPI_DIVSOR)

Bit	Description	Default	Access
7	**TX Mode Only / Disable RX Data Push** 0: To receive data simultaneously when driving SPI CLK 1: Stop receiving data when driving SPI CLK This setiing is used to avoid receiving unneeded data during sending Command, Address and Dummy stages.	0	RW
6	**Set XSIOn Input/Output State** 0: Output – Write (Drive) Set SPI interface based on Data Wire Mode settings to output state. 1: Input – Read (Tri-state) Set SPI interface based on Data Wire Mode settings to input state.	0	RW

Bit	Description	Default	Access
5-4	**Data Wire Mode** 00：Standared SPI；X1 MOSI：Keep at output state aftere data output MISO：Input state {XSIO3, XSIO2}：Keep at tri-state 01：Single SPI；X1 MOSI：Tri-state after data output MISO：Input state {XSIO3, XSIO2}：Keep at tri-state 10：Dual SPI；X2 {MISO, MOSI}：According to bit[6] (XSIO Input/Output state), tri-state after data output {XSIO3, XSIO2}：Keep at tri-state 11：Quad SPI；X4 {XSIO3, XSIO2, MISO, MOSI}：According to bit[6] (XSIO Input/Output state), tri-sate after data output.	0	RW
3-0	**SPI Clock Period** Set SPI clock based on the system clock and the clock requested by the SPI device. $F_{SCK} = F_{CORE} / ((Divisor + 1) *2)$	3	RW

REG[BFh-BCh] Serial Flash DMA Source Starting Address (DMA_SSTR)

Bit	Description	Default	Access
31-0	**Serial Flash DMA Source START Address[31:0]** REG[BFh] mapping to DMA_SSTR[31:24]。 REG[BEh] mapping to DMA_SSTR[23:16]。 REG[BDh] mapping to DMA_SSTR[15:8]。 REG[BCh] mapping to DMA_SSTR[07:0]。 These registers are used to set the SPI Flash address. The address directly points to the starting position of the source image.	0	RW

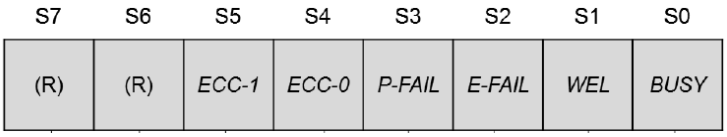
REG[C1h-C0h] DMA Source|Destination Window Upper-Left Corner X-Coordinates (DMA_SX |DMA_DX)

Bit	Description	Default	Access
15-0	**When REG[5Eh] bit2=0 (Block mode) and REG[B7h] bit4=0:** This register defines the DMA Destination Window Upper-Left Corner X-coordinates [12:0] on Canvas. This is also the starting position to store the decoded JPG pictures. REG[C0h] mapping to DMA_DX[7:0] REG[C1h] bit[4:0] mapping to DMA_DX[12:8], bit[7:5] are not used. **When REG[5Eh] bit2 = 1 (Linear Mode)** This register defines the destination memory address [15:0] REG[C0h] bit[7:2] mapping to DMA_MEMA[7:2], bit[1:0] are fixed to 00b. REG[C1h] mapping to DMA_MEMA[15:8] **When REG[5Eh] bit2=0 (Block mode) and REG[B7h] bit4=1:** This register defines the DMA Source Window Upper-Left Corner X-coordinates [12:0] in Flash. REG[C0h] mapping to DMA_SX[7:0] REG[C1h] bit[4:0] mapping to DMA_SX[12:8], bit[7:5] are not used. **Note:** If the Upper-Left Corner X-coordinates of the Destination window + the Block width of the DMA > the Canvas width, the data written byond the Canvas width will be discarded. If the Upper-Left Corner Y-coordinates of the Destination window + the Block height of the DMA > the Canvas height, the data written byond the Canvas height will be written across the Canvas to the subsequent memory area.	0	RW

REG[C3h-C2h] DMA Source|Destination Window Upper-Left Corner Y-Coordinates (DMA_SY |DMA_DY)

Bit	Description	Default	Access
15-0	**When REG[5Eh] bit2=0 (Block mode) and REG[B7h] bit4=0:** This register defines the DMA Destination Window Upper-Left Corner Y-coordinates [12:0] on Canvas. This is also the starting position to store the decoded JPG pictures. REG[C2h] mapping to DMA_DY[7:0] REG[C3h] bit[4:0] mapping to DMA_DY[12:8], bit[7:5] are not used. **When REG[5Eh] bit2 = 1 (Linear Mode)** This register defines the destination memory address [31:16] REG[C2h] mapping to DMA_MEMA[23:16] REG[C3h] mapping to DMA_MEMA[31:24] **When REG[5Eh] bit2=0 (Block mode) and REG[B7h] bit4=1:** This register defines the DMA Source Window Upper-Left Corner Y-coordinates [12:0] in Flash. REG[C2h] mapping to DMA_SY[7:0] REG[C3h] bit[4:0] mapping to DMA_SY[12:8], bit[7:5] are not used. **Note:** If the Upper-Left Corner X-coordinates of the Destination window + the Block width of the DMA > the Canvas width, the data written byond the Canvas width will be discarded. If the Upper-Left Corner Y-coordinates of the Destination window + the Block height of the DMA > the Canvas height, the data written byond the Canvas height will be written across the Canvas to the subsequent memory area.	0	RW

REG[C4h] R: SPI NAND Status Register / W: Bad Block Switch Complete Register

Bit	Description	Default	Access
7-0	**This register returns the value of the SPI NandFlash Status Register (0xC0) if a Bad Block is detected** Following is an example of Winbond Flash:  For Micron and XTX, S7 is CRBSY, and S6 is ECC-2	NA	RO
7-0	Writing any value to this register means the Bad Block Switch is done. The Flash DMA process will continue.	NA	WO

REG[C5h] Miscellaneous Setting

Bit	Description	Default	Access
7	Reserved	0	RW
6-4	**Tcsh: the shortest time of SPI SFCS# stays in High level** Minimum Deselect Time = (Val + 1) * period of XSCK	3	RW
3	Reserved	0	RO
2	**The width of the NAND Flash ECC status bits** 0: 2 bits (e.g. Winbond); applicable for the back block detection of low and medium sensitivity. 1: 3 bits (e.g. Micron); applicable for the bad block detection of high, medium, and low sensitivity.	0	RW
1-0	**Bad Block Management Mode – Sensitivity Settings** 00: Disable Bad Block Management 01: High Sensitivity → Inform the MCU when 1~3 bits data error are detected. 10: Medium Sensitivity → Inform the MCU when 4-6 bits data error are detected. 11: Low Sensitivity → Inform the MCU when 7-8 bits data error are detected.	0	RW

REG[EDh-EBh] ECC Failure Page Address

Bit	Description	Default	Access
23-0	**ECC Failure Page Address [23:0] = {EDh, ECh, EBh}**	0	RO

REG[C7h-C6h] DMA Block Width (DMAW_WTH)

Bit	Description	Default	Access
15-0	**When REG[5Eh] bit2=0 (Block Mode)** This register defines DMA Block Width, DMAW_WTH[15:0] REG[C7h] mapping to DMAW_WTH[15:8] REG[C6h] mapping to DMAW_WTH[7:0] **When REG[5Eh] bit2=1 (Linear Mode)** This register defines the DMA Transfer Number REG[C7h] mapping to DMAW_WTH[15:8] REG[C6h] mapping to DMAW_WTH[7:0] **Note 1:** If the Upper-Left Corner X-coordinates of the Destination window + the Block width of the DMA > the Canvas width, the data written byond the Canvas width will be discarded. **Note 2:** In JPG decoding mode (REG B6h[1]=1, SFDMA_FDEC), no need to set this register. LT758x will auto-detect the picture width and height. The MCU may read this register to get the picture width.	0	RW

REG[C9h-C8h] DMA Block Height（DMAW_HIGH）

Bit	Description	Default	Access
15-0	**When REG[5Eh] bit2=0 (Block Mode)** This register defines DMA Block Height, DMAW_HIGH[15:0] REG[C9h] mapping to DMAW_HIGH[15:8] REG[C8h] mapping to DMAW_HIGH[7:0] **When REG[5Eh] bit2=1 (Linear Mode)** This register defines the DMA Transfer Number REG[C9h] mapping to DMAW_HIGH[31:24] REG[C8h] mapping to DMAW_HIGH[23:16] **Note 1:** If the Upper-Left Corner Y-coordinates of the Destination window + the Block height of the DMA > the Canvas height, the data written byond the Canvas height will be written across the Canvas to the subsequent memory area. **Note 2:** In JPG decoding mode (REG B6h[1]=1, SFDMA_FDEC), no need to set this register. LT758x will auto-detect the picture width and height. The MCU may read this register to get the picture height.	0	RW

REG[CBh-CAh] DMA Source Picture Width（DMA_SWTH）

Bit	Decription	Default	Access
15-0	**Note:** The registers are not applicable under JPG decoding mode. **DMA Source Picture Width [13:0]** REG[CBh] bit[7:0] mapping to DMA_SWTH[15:8]。 REG[CAh] bit[7:0] mapping to DMA_SWTH[7:0]。 Unit: Pixel. Effective value: 1 ~ 8,192	0	RW

Please refer to Section 13.2 for the description of SPI Flash Controller.

17.10 Text Engine Registers

REG[CCh] Character Control Register 0 (CCR0)

Bit	Description	Default	Access
7:6	**Character Source Selection** 00b: Select internal CGROM Character 01b: User-defined Character (Source: SPI Flash) 10b: User-defined Character (CGRAM) 11b: Reserved	0	RW
5-4	**Character Height Setting** 00b: 16 dots; e.g. 8*16 / 16*16 01b: 24 dots; e.g. 12*24 / 24*24 10b: 32 dots; e.g. 16*32 / 32*32 **Note:** 1. User-defined character width is decided by character code; width for code < 8000h is 8/12/16. width for code >=8000h is 16/24/32. 2. Internal CGROM supports sizes of 8*16 / 12*24 / 16*32.	0	RW
3-2	Reserved	0	RO
1-0	**Character Selection for Internal CGROM** When bit[7:6] = 00b, the Internal CGROM is selected. It supports character sets with the standard coding of ISO/IEC 8859-1,2,4,5, which supports English and most of the European country languages 00b: ISO/IEC 8859-1 01b: ISO/IEC 8859-2 10b: ISO/IEC 8859-4 11b: ISO/IEC 8859-5	0	RW

REG[CDh] Character Control Register 1 (CCR1)

Bit	Description	Default	Access
7	**Full Alignment Selection** 0: Disable Full alignment 1: Enable Full alignment When Full alignment is enable, displayed character width is equal to (Character Height)/2 if character width is equal to or smaller than (Character Height)/2, otherwise displayed font width is equal to Character Height.	0	RW
6	**Chroma Keying Enable on Text Input** 0: Character's background displayed in the specified color. 1: Character's background displayed with original canvas' background	0	RW

Bit	Description	Default	Access
5	Reserved	0	RW
4	**Character Rotation** 0: Text direction is from left to right then from top to bottom 1: Counterclockwise 90 degree & vertical flip. Text direction is from top to bottom then from left to right. This attribute can be changed only when previous Text write is finished (Core_Busy = 0)	0	RW
3-2	**Character Width Enlargement Factor** 00b: x1 01b: x2 10b: x3 11b: x4	0	RW
1-0	**Character Height Enlargement Factor** 00b: x1 01b: x2 10b: x3 11b: x4	0	RW

REG[CFh-CEh] RESERVED

Bit	Description	Default	Access
7-0	Reserved	0	RO

REG[D0h] Character Line gap Setting Register (FLDR)

Bit	Description	Default	Access
7-5	Reserved	0	RO
4-0	**Character Line Gap Setting** Setting the character line gap. When entered character reaches the boundary of the active window, the controller jumps to the next line. (Unit: pixel) The color of the line gap will be the same as the background color, and tt will not be enlarged by the character enlargement function.	0	RW

REG[D1h] Character to Character Space Setting Register (F2FSSR)

Bit	Description	Default	Access
7-6	Reserved	0	RW

Bit	Description	Default	Access
5-0	**Character to Character Space Setting** 00h：0 pixel 01h：1 pixel 02h：2 pixels : : 3Fh：63 pixels The color of the space between characters will be the same as the background color. It will not be enlarged by character enlargement function.	0	RW

REG[D2h] Foreground Color Register - Red (FGCR)

Bit	Description	Default	Access
7-0	**Foreground Color – Red (for Draw, Text and Color Expansion)** 256 Colors: Red mapping to bit[7:5] 65K Colors: Red mapping to bit[7:3] 16.7M Colors: Red mapping to bit[7:0]	FFh	RW

REG[D3h] Foreground Color Register - Green (FGCG)

Bit	Description	Default	Access
7-0	**Foreground Color – Green (for Draw, Text, and Color Expansion)** 256 Colors: Red mapping to bit[7:5] 65K Colors: Red mapping to bit[7:2] 16.7M Colors: Red mapping to bit[7:0]	FFh	RW

REG[D4h] Foreground Color Register - Blue (FGCB)

Bit	Description	Default	Access
7-0	**Foreground Color - Blue (for Draw, Text, and color Expansion)** 256 Colors: Red mapping to bit[7:6] 65K Colors: Red mapping to bit[7:3] 16.7M Colors: Red mapping to bit[7:0]	FFh	RW

REG[D5h] Background Color Register - Red (BGCR)

Bit	Description	Default	Access
7-0	**Background Color - Red** 256 Colors: Red mapping to bit[7:5] 65K Colors: Red mapping to bit[7:3] 16.7M Colors: Red mapping to bit[7:0]	00h	RW

REG[D6h] Background Color Register - Green (BGCG)

Bit	Description	Default	Access
7-0	**Background Color - Green** 256 Colors: Red mapping to bit[7:5] 65K Colors: Red mapping to bit[7:2] 16.7M Colors: Red mapping to bit[7:0]	00h	RW

REG[D7h] Background Color Register - Blue (BGCB)

Bit	Description	Default	Access
7-0	**Background Color - Blue** 256 Colors: Red mapping to bit[7:5] 65K Colors: Red mapping to bit[7:3] 16.7M Colors: Red mapping to bit[7:0]	00h	RW

Note: No matter the Background transparency is enabled or not, please do not set it as the same value as the Foreground Color, otherwise, the images and text will become a square with Foreground Color. When using BTE function, these two values should not be set to the same either.

REG[D8h] Foreground Alpha Color (FGCA)

Bit	Description	Default	Access
7-0	**Foreground Color – Alpha** Using 32bpp data format 0：Trasparent 1~254：Display by the transparent ratio 255：Non-transparent	FFh	RW

REG[D9h] Background Alpha Color (BGCA)

Bit	Description	Default	Access
7-0	**Background Color – Alpha** Using 32bpp data format 0：Trasparent 1~254：Display by the transparent ratio 255：Non-transparent	FFh	RW

Note 1: Text coordinates registers (X, Y) = ([64h:63h], [66h:65h])

Note 2: Do not change the color paramters before the text is completely written to the memory as it may cause the text to be displayed in unexpected color.

REG[DAh]: Color Expansion

Bit	Description	Default	Access
7-2	Reserved	0	RO
1-0	**8bpp/16bpp Expands to 24bpp** **00b: Expanded by MSB** **Ex**. 8bpp: 10101110b (332) -> R:101b, G:011b, B:10b To 24bpp->R:10110110b, G: 01101101b, B: 10101010b **Ex**. 16bpp: 0110101110011001b (565) ->R: 01101b, G: 011100b, B: 11001b To 24bpp-> R: 01101011b, G: 01110001b, B: 11001110b **01b: Expanded by 0** **Ex**. 8bpp: 10101110b (332) -> R:101b, G:011b, B:10b To 24bpp->R:10100000b, G: 01100000b, B: 10000000b **Ex**. 16bpp: 0110101110011001b (565) ->R: 01101b, G: 011100b, B: 11001b To 24bpp-> R: 01101000b, G: 01110000b, B: 11001000b **10b: Expanded by 1** **Ex**. 8bpp: 10101110b (332) -> R:101b, G:011b, B:10b To 24bpp->R:10111111b, G: 01111111b, B: 1011111b **Ex**. 16bpp: 0110101110011001b (565) ->R: 01101b, G: 011100b, B: 11001b To 24bpp-> R: 01101111b, G: 01110011b, B: 11001111b	0	RW

REG[DEh-DBh] CGRAM Start Address (CGRAM_STRx)

Bit	Description	Default	Access
31-0	**CGRAM START ADDRESS [7:0]** User-defined Characters space REG[DEh] mapping to CGRAM_STR [31:24] REG[DDh] mapping to CGRAM_STR [23:16] REG[DCh] mapping to CGRAM_STR [15:8] REG[DBh] mapping to CGRAM_STR [7:0] Host must save CGRAM data based on the setting of canvas, and set CGRAM address so that Text engine knows where to fetch CGRAM data. **Note:** If Character Control Register REG[CCh] bit[7:6] = 01b, the specified CGRAM Start Address directs to the Serial Flash.	0	RW

Note: If the MCU wants to change the rotate attribute, character line gap, character-to-character space, foreground color, background color and Text/graphic mode settings, please make sure that Task_Busy (Status Register bit3) status bit is low.

17.11 Power Management Control Register

REG[DFh]: Power Management Register (PMU)

Bit	Description	Default	Access
7	**Enter Power Saving State** 0: Normal state or wake up from the power saving state 1: Enter power saving state. **Note:** There are 2 ways to wake up from power saving state: External interrupt event and Software wakeup. Writing this bit to 0 will generate Software wakeup. It will not be cleared until chip resume. MCU must wait until system is awekened before writting to other registers. The MCU may check this bit or check status bit1 (power saving) to find out if the system is back to the normal state.	0	RW
6-2	Reserved	0	RO
1-0	**Power Saving Mode Definition** 00b: Reserved 01b: Standby Mode, CCLK is provided by OSC, PCLK will stop, and MCLK will be provided by MPLL. (Parallel) 10b: Suspend Mode, CCLK & PCLK will stop, MCLK will be provided by OSC. (Parallel) 11b: Sleep Mode, All clocks and PLL will stop. (Parallel)	0	RW

17.12 Display RAM Control Register

REG[E0h] SDRAM Attribute Register (SDRAR)

Bit	Description	Default	Access
7	**SDRAM Power Saving** 0: Execute power down command to enter power saving mode 1: Execute self refresh command to enter power saving mode	0	RW
6	This bit must be set as 0.	0	RW
5	**SDRAM Bank Number, SDR_BANK** 0: 2 Banks (column addressing size only supports 256 words) 1: 4 Banks **Note:** This bit must set as 1 for normal operation.	1	RW
4-3	**SDRAM Row Addressing, SDR_ROW** 00b: 2K (A0-A10) 01b: 4K (A0-A11) 1xb: 8K (A0-A12) **Note:** The two bits must set as 01b (4K) for normal operation.	1	RW
2-0	**SDRAM Column Addressing, SDR_COL** 000b: 256 (A0-A7) 001b: 512 (A0-A8) 010b: 1,024 (A0-A9) 011b: 2,048 (A0-A9, A11) 1xxb: 4,096 (A0-A9, A11-A12) **Note:** The three bits must set as 001b (512) for normal operation.	0	RW

Note: For normal operation, REG[E0h] must be set as 0x29, based on LT758x SDRAM structure.

REG[E1h] SDRAM Mode Register & Extended Mode Register (SDRMD)

Bit	Description	Default	Access
7-3	bit[7:3] must be set as 0	0	RW
2-0	**SDRAM CAS Latency, SDR_CASLAT** 010b: 2 SDRAM clock 011b: 3 SDRAM clock Others: Reserved **Note:** The suggest setting value of this register is **03h.** This register is locked after SDR_INITDONE (REG[E4h] bit0) is set as 1.	011b	RW

REG[E3h-E2h] SDRAM Auto Refresh Interval (SDR_REF)

Bit	Description	Default	Access
15-0	**SDRAM Auto Refresh Interval** REG[E3h] mapping to SDR_REF [15:8]. REG[E2h] mapping to SDR_REF [7:0] The internal refresh time is determined by the refresh cycle of the SDRAM and the row size. For example, if the SDRAM frequency is 132MHz, SDRAM's refresh cycle Tref is 64ms, and the row size is 4,096, then the internal refresh time should be less than $64*10^{-3}$ / 4096 * $132*10^{6}$ ~= 2063 = 80Fh. Therefore the REG[E3h][E2h] is set to 080Fh. If the SDRAM frequency is 168MHz, then the internal refresh time should be less than $64*10^{-3}$ / 4096 * $168*10^{6}$ ~= 2625 = A41h. Therefore the REG[E3h][E2h] is set to 0A41h. **Note:** If this register is set to 0000h, then SDRAM self-refresh function will be disable.	00h	RW

Table 17-7: The Reference Setting of REG[E3h-E2h] when REG[E4] bit2=0

SDRAM Clock	REG[E3h]	REG[E2h]
CCLK=132MHz	08h	0Fh
CCLK=168MHz	0Ah	41h

REG[E4h] SDRAM Control Register (SDRCR)

Bit	Description	Default	Access
7-5	**[7:6] Length to Break a Burst Transfer** 00: 256 01: 128 10: 64 11: 32 **[5] SDRAM Bus Width Select: 0**	0	RW
4	**MCKE Status** SDRAM MCKE State 0: Low 1: High	1	RO
3	**Report Warning Condition** 0: Disable or Clear warning flag 1: Enable warning flag Warning conditions include (1) memory read cycle is close to the maximum address boundry of SDRAM (exceeding maximum address minus 512 bytes); (2) out of memory access range; (3) insufficient SDRAM bandwidth to fulfill panel's frame rate. Under the above situations, the warning event will be latched. The MCU can read back this bit to check if there is a warning flag. The warning flag could be cleared by setting this bit to 0. If the warning flag is disabled, the access that exceeds the memory range will loop back to the address 0, and overwrite the data. If the warning flag is enabled, the access that exceeds the memory range will be discarded.	0	RW
2	**SDRAM Timing Parameter Register Enable, SDR_PARAMEN** 0: Disable Display RAM timing parameter registers 1: Enable Display RAM timing parameter registers	0	RW
1	**Enter Power Saving Mode, SDR_PSAVING** 0 to 1 transition will enter power saving mode 1 to 0 transition will exit power saving mode The MCU may set REG[E0h] bit7 (SDRAM Power Saving) to decide how to enter the power saving mode: 0: Execute power down command to enter power saving mode 1: Execute self refresh command to enter power saving mode	0	RW

Bit	Description	Default	Access
0	**Start SDRAM Initialization Procedure, SDR_INITDONE** 0 to 1 transition: This will execute Display RAM initialization procedure. Read back this bit, if the value is '1' , it means Display RAM is initialized and ready for access. Once it is written as 1, it cannot be rewritten as 0. 1 to 0 transition: No operation.	0	RW

The following Display RAM Timing Registers (REG[E0h-E3h]) are available only when SDR_PARAMEN (REG[E4] bit2) is set to 1.

REG[E0h] SDRAM Timing Parameter 1

Bit	Description	Default	Access
7-4	Reserved	0	RO
3-0	**Time from Load Mode Command to Active/Refresh Command (T_{MRD})** 0000b: 1 SDRAM Clock 0001b: 2 SDRAM Clock 0010b: 3 SDRAM Clock : : 1111b: 16 SDRAM Clock	2	RW

REG[E1h] SDRAM Timing Parameter 2

Bit	Description	Default	Access
7-4	**Auto Refresh Period, T_{RFC}** 0h – Fh: 1 ~ 16 SDRAM Clocks (As REG[E0h] bit[3:0])	8	RW
3-0	**Time required to Exit SELF Refresh-to-ACTIVE Command (T_{XSR}), at least 70ns needed** 0h – Fh: 1 ~ 16 SDRAM Clocks	7	RW

REG[E2h] SDRAM Timing Parameter 3

Bit	Description	Default	Access
7-4	**Pre-charge Command Period (T_{RP}, 15/20ns), at least 15ns needed** 0h – Fh: 1 ~ 16 SDRAM Clocks	2	RW
3-0	**WRITE Recovery Time (T_{WR}), at least 2 SDRAM Clocks needed.** 0h – Fh: 1 ~ 16 SDRAM Clocks	0	RW

REG[E3h] SDRAM Timing Parameter 4

Bit	Description	Default	Access
7-4	**Delay Time of Active-to-Read/Write (T_{RCD}), at least 15ns needed** 0h – Fh: 1 ~ 16 SDRAM Clocks	2	RW
3-0	**Time of Active-to-Precharge (T_{RAS}), at least 40ns needed** 0h – Fh: 1 ~ 16 SDRAM Clocks	6	RW

Table 17-8: REG[E0h-E3h] Setting Examples when REG[E4] bit2=1REG[E0h-E3h]

Bit	SDRAM Timing Requirement	Ex1: MCLK=132MHz (Cycle Time = 7.6ns)	Ex2: MCLK=168MHz (Cycle Time ~= 6ns)
REG[E0h] bit3-0	Load Mode Command to Active / Refresh (T_{MRD}) → Must > 2 MCLK	**1** (>= 2 MCLK)	**1** (>= 2 MCLK)
REG[E1h] bit7-4	Auto Refresh Period, T_{RFC} → Must > 55ns	**7** (>= 8 MCLK)	**9** (>= 10 MCLK)
REG[E1h] bit3-0	Exit SELF Refresh-to-ACTIVE Command (T_{XSR}) → Must > 70ns	**9** (>= 10 MCLK)	**11** (>= 12 MCLK)
REG[E2h] bit7-4	Pre-charge Command (T_{RP}, 15/20ns) → Must > 15ns	**1** (>= 2 MCLK)	**2** (>= 3 MCLK)
REG[E2h] bit3-0	WRITE Recovery Time (T_{WR}) → Must > 2 MCLK	**1** (>= 2 MCLK)	**1** (>= 2 MCLK)
REG[E3h] bit7-4	Delay Tiem of Active-to-Read/Write (T_{RCD}) → Must > 15ns	**1** (>= 2 MCLK)	**2** (>= 3 MCLK)
REG[E3h] bit3-0	Active-to-Precharge (T_{RAS}) → Must > 40ns	**5** (>= 6 MCLK)	**6** (>= 7 MCLK)

17.13 I2C Master Registers

REG[E6h-E5h] I2C Master Clock Prescaler Register (I2CMCK)

Bit	Description	Default	Access
15-0	**I2C Master Clock Prescaler [15:0]** REG[E6h] mapping to I2CMCK[15:8]. REG[E5h] mapping to I2CMCK[7:0].	0	RW

REG[E7h] I2C Master Transmit Register (I2CMTXR)

Bit	Description	Default	Access
7-0	**I2C Master Transmit [7:0]**	0	RW

REG[E8h] I2C Master Receiver Register (I2CMRXR)

Bit	Description	Default	Access
7-0	**I2C Master Receiver [7:0]**	0	RO

REG[E9h] I2C Master Command Register (I2CMCMD)

Bit	Description	Default	Access
7	**Start Command** Write 1: Generate (repeated) start condition and be auto-cleared by hardware. **Note:** This bit is always read as 0.	0	RW
6	**Stop Command** Write 1: Generate stop condition and be auto-cleared by hardware. **Note:** This bit is always read as 0.	0	RW
5	**Read Command** Write 1: Read form slave and be auto-cleared by hardware. **Note:** This bit is always read as 0.	0	RW
4	**Write Command** Write 1: Write to slave and be auto-cleared by hardware. This bit is always read as 0. **Note:** Read and Write cannot be done at the same time.	0	RW
3	**Acknowledge Command** Write 0: Send ACK Write 1: Send NACK	0	RW
2-1	Reserved	0	RO
0	**Noise Filter** 0: Disable 1: Enable	0	RW

REG[EAh] I2C Master Status Register (I2CMST)

Bit	Description	Default	Access
7	**Received Acknowledge from Slave** 0: Acknowledge received. 1: No Acknowledge received.	0	RO
6	**I2C Bus is Busy** 0: Idle. This bit becomes 0 after Stop signal is detected. 1: Busy . This bit becomes 1 after Start signal is detected.	0	RO
5-2	Reserved	0	RO
1	**I2C Transfer in Progress** 0: when the transmission is done 1: when the transmission is in process	0	RO
0	**Arbitration Lost State** When LT758x lost Arbitration, this bit will be set to 1. **Note:** When a Stop signal is detected, but it is not required, then it means arbitration loss. The LT758x I2C Master will drive SDA to 1, but the other Master will drive SDA to 0.	0	RO

REG[EDh-EBh] ECC Failure Page Address)

Refer to Section 17.9.

REG[EFh-EEh] EINTR_DBCTM

Bit	Description	Default	Access
15-0	**PSM[0] De-bounce Time [15:0]** REG[EFh] mapping to ExtIntr_DBCTM[15:8]. REG[EEh] mapping to ExtIntr_DBCTM[7:0]. Zero based. 256 OSC as one unit, 0 represents 256 OSC.	3	RW

17.14 GPIO Registers

REG[F0h] GPIO-A Direction (GPIOAD)

Bit	Description	Default	Access
7-0	**GPIOA In/Out Control)** 0: Output 1: Input	FFh	RW

REG[F1h] GPIO-A (GPIOA)

Bit	Description	Default	Access
7-0	**GPIOA Data** Write: Output Data to GPIOA Read: Read Data from GPIOA GPIOA[7:0] are General Purpose I/O. These signals are shared with DB[15:8]. They are available only when Host is in 8bits parallel or Serial mode.	NA	RW

REG[F2h] GPIO-B (GPIOB)

Bit	Description	Default	Access
7-6	Reserved	NA	NA
5	**GPO Output Enable** 0: Enable GPIOB Output 1: Disable GPIOB Output (Floating)	0	RW
4	**GPIOB[4] Data** Write: Output Data to GPOB[4] Read: Read Data from GPIB[4] **Note:** GPOB[4] is shared with I2CMDA pin. GPIB[4] is shared with I2CMCK pin.	NA	RW
3-0	**GPIOB[3:0] Data** Write: when GPIOB output is enabled, these pins are output through { A0, WR#, RD#, CS# } Read: Read Data from GPIOB[3:0] **Note:** GPIB[3:0] are shared with { A0, WR#, RD#, CS# }. They are available only in Serial MCU I/F.	NA	RO

REG[F3h] GPIO-C Direction (GPIOCD)

Bit	Description	Default	Access
7-0	**GPIOC In/Out Control** 0: Output 1: Input	FFh	RW

REG[F4h] GPIO-C (GPIOC)

Bit	Description	Default	Access
7	**GPIOC[7] Data** Write: Output Data to GPIOC[7] Read: Read Data from GPIOC[7] GPIOC[7] is shared with PWM[0]. GPIOC is available only when PWM and SPI Master functions are disabled.	NA	RW
6-5	**GPIOC[6:5] Data** Write: Output Data to GPIOC[6:5] Read: Read Data from GPIOC[6:5] GPIOC[6:5] is shared with {XSIO3, XSIO2}. GPIOC[6:5] are available only when SPI Master function is disabled.	NA	RW
4-0	**GPIOC[4:0] Data** Write: Output Data to GPIOC[4:0] Read: Read Data from GPIOC[4:0] GPIOC[4:0] are shared with { SFCS[1]#, SFCS[0]#, MISO, MOSI, SFCLK } pins, they are only available when PWM and SPI Master functions are disabled.	NA	RW

REG[F5h] GPIO-D Direction (GPIODD)

Bit	Description	Default	Access
7-0	**GPIOD In/Out Control** 0: Output 1: Input	FFh	RW

REG[F6h] GPIO-D (GPIOD)

Bit	Description	Default	Access
7-0	**GPIOD Data** Write: Output Data to GPIOD[7:0] Read: Read Data from GPIOD[7:0] GPIOD[7:0] are shared with PD[18, 2, 17, 16, 9, 8, 1, 0] pins GPIOD[5,4,1,0] are available when LCD panel data bus is 16 or 12bits. GPIOD[7,6,3,2] are available when LCD panel data bus is 16bits.	NA	RW

REG[F7h] GPIO-E Direction (GPIOED)

Bit	Description	Default	Access
7-0	**GPIOE In/Out Control** 0: Output 1: Input	FFh	RW

REG[F8h] GPIO-E (GPIOE)

Bit	Description	Default	Access
7-0	**GPIOE Data** Only Available On Digital Display Package Type GPIOE[7:0] are General Purpose I/O. These pins are shared with PD[12, 11, 10, 7, 6, 5, 4, 3] pins	NA	RW

REG[F9h] GPIO-F Direction (GPIOFD)

Bit	Description	Default	Access
7-0	**GPIOF In/Out Control** 0: Output 1: Input	FFh	RW

REG[FAh] GPIO-F (GPIOF)

Bit	Description	Default	Access
7-0	**GPIOF Data** GPIOF[7:0] are General Purpose I/O. These pins are shared with PD[23, 22, 21, 20, 19, 15, 14, 13] pins	NA	RW

17.15 Extended Register（REG_00h[3] == 1）

REG[00h] Software Reset Register (SRR)

Bit	Description	Default	Access
7	**PLL Frequency Lock** 1: PLL Frequency is locked.	1	RO
6-4	Reserved	0	RO
3	**Select Register Group** 0：Group 0 – Select the default register group 1：Group 1– Select the extended register group This bit is not restricted by the register lock or the slected register group.	0	RW
2	**Lock Register** 0：R/W to all register 1：R Only to all register --- Except for this bit	0	RW
1	**Ext_FlatInk** 1: Force to enter DE Mode, where VSYNC/HSYNC becomes PD # / PD signals to control the external LVDS conversion IC. 0: Regular status	0	RO
0	**Warning Condition Flag** 0：No warning occurred. 1：Warning occurred. Refer to REG[E4h] bit3 description. If this flag is disabled, then the access that exceeds the memory range will loop back to address 0 and overwrite it. If this flag is enabled, then the access that exceeds the memory range will be discarded.	0	RO

SPI NAND Flash – Bad Block Remap Table

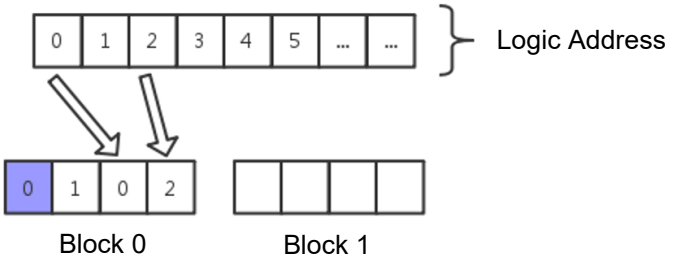


Figure 17 ND Fla lock F.. ..

SPI NAND Flash: Each Block siz es; the size is 2048bytes; and the Spare Area size varies by mod

REG[B3h-B0h] Bad Block Remap Link 0, REG[B7h-B4h] Bad Block Remap Link 1
REG[BBh-B8h] Bad Block Remap Link 2, REG[BFh-BCh] Bad Block Remap Link 3
REG[C3h-C0h] Bad Block Remap Link 4, REG[C7h-C4h] Bad Block Remap Link 5
REG[CBh-C8h] Bad Block Remap Link 6, REG[CFh-CCh] Bad Block Remap Link 7
REG[D3h-D0h] Bad Block Remap Link 8, REG[D7h-D4h] Bad Block Remap Link 9
REG[DBh-D8h] Bad Block Remap Link 10, REG[DFh-DCh] Bad Block Remap Link 11
REG[E3h-E0h] Bad Block Remap Link 12, REG[E7h-E4h] Bad Block Remap Link 13
REG[EBh-E8h] Bad Block Remap Link 14, REG[EFh-ECh] Bad Block Remap Link 15
REG[F3h-F0h] Bad Block Remap Link 16, REG[F7h-F4h] Bad Block Remap Link 17
REG[FBh-F8h] Bad Block Remap Link 18, REG[FFh-FCh] Bad Block Remap Link 19

Bit	Description	Default	Access
31-0	**Bad Block Remap Link x** Ex. Bit [31:0] == {83h, 82h, 81h, 80h} [31]: Valid Remap Link. [30:28]: NA [27:16]: Logical Block Address [15:12]: NA [11:0]: Physical Block Address	0	RW

Note: If the logical block address is repeated, the one with larger number has the higher priority.

18. Package Information

18.1 LT7586（LQFP-128Pin）

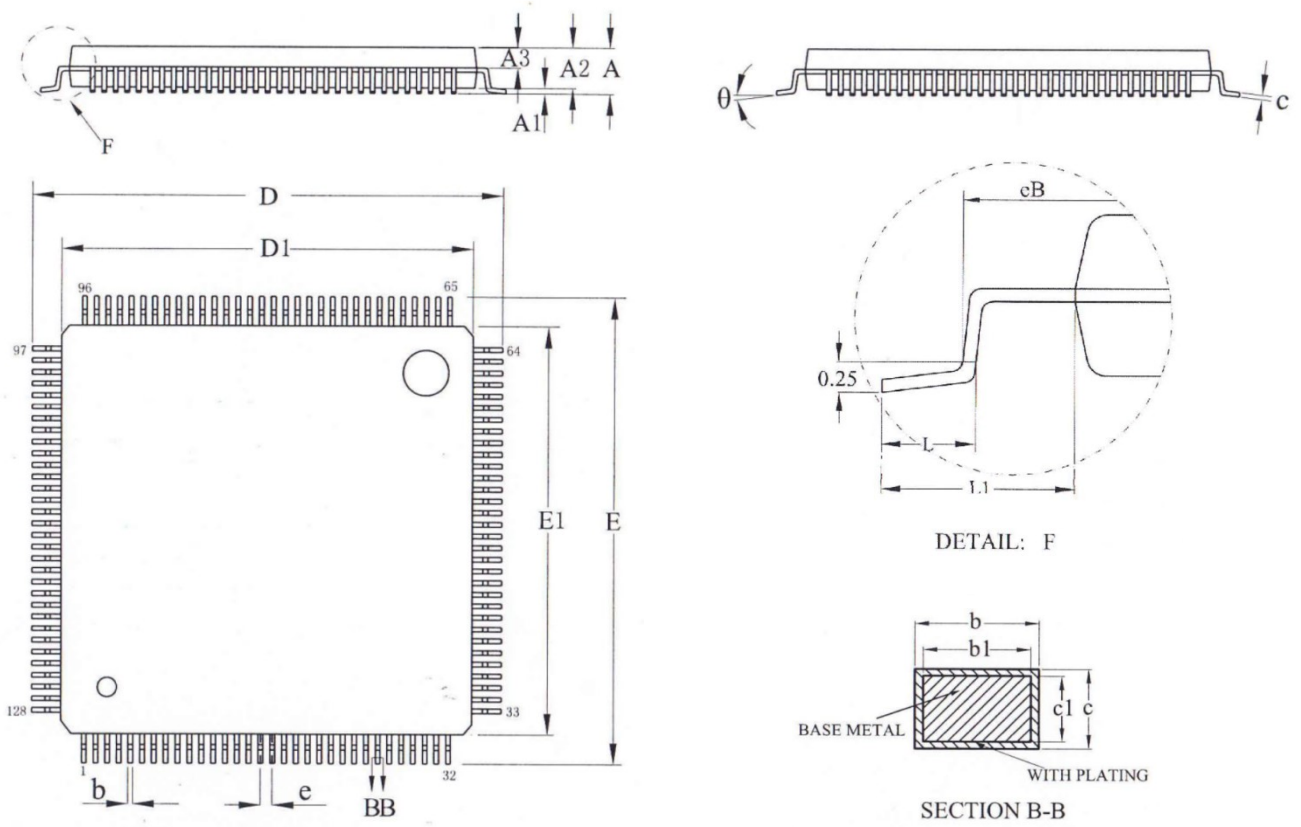


Figure 18-1：LQFP-128Pin Outline

Table 18-1：LQFP-128Pin Dimension

Symbol	Millimeter			Symbol	Millimeter		
	Min.	Nom.	Max		Min.	Nom.	Max
A	-	-	1.60	**D1**	13.9	14.0	14.1
A1	0.05	-	0.15	**E**	15.8	16.0	16.2
A2	1.35	1.40	1.45	**E1**	13.9	14.0	14.1
A3	0.59	0.64	0.69	**eB**	15.05	-	15.35
b	0.14	-	0.22	**e**	0.40BSC		
b1	0.13	0.16	0.19	**L**	0.45	-	0.75
c	0.13	-	0.17	**L1**	1.00REF		
c1	0.12	0.13	0.14	**θ**	0		7
D	15.8	16.00	16.2				

18.2 LT7583（LQFP-100Pin）

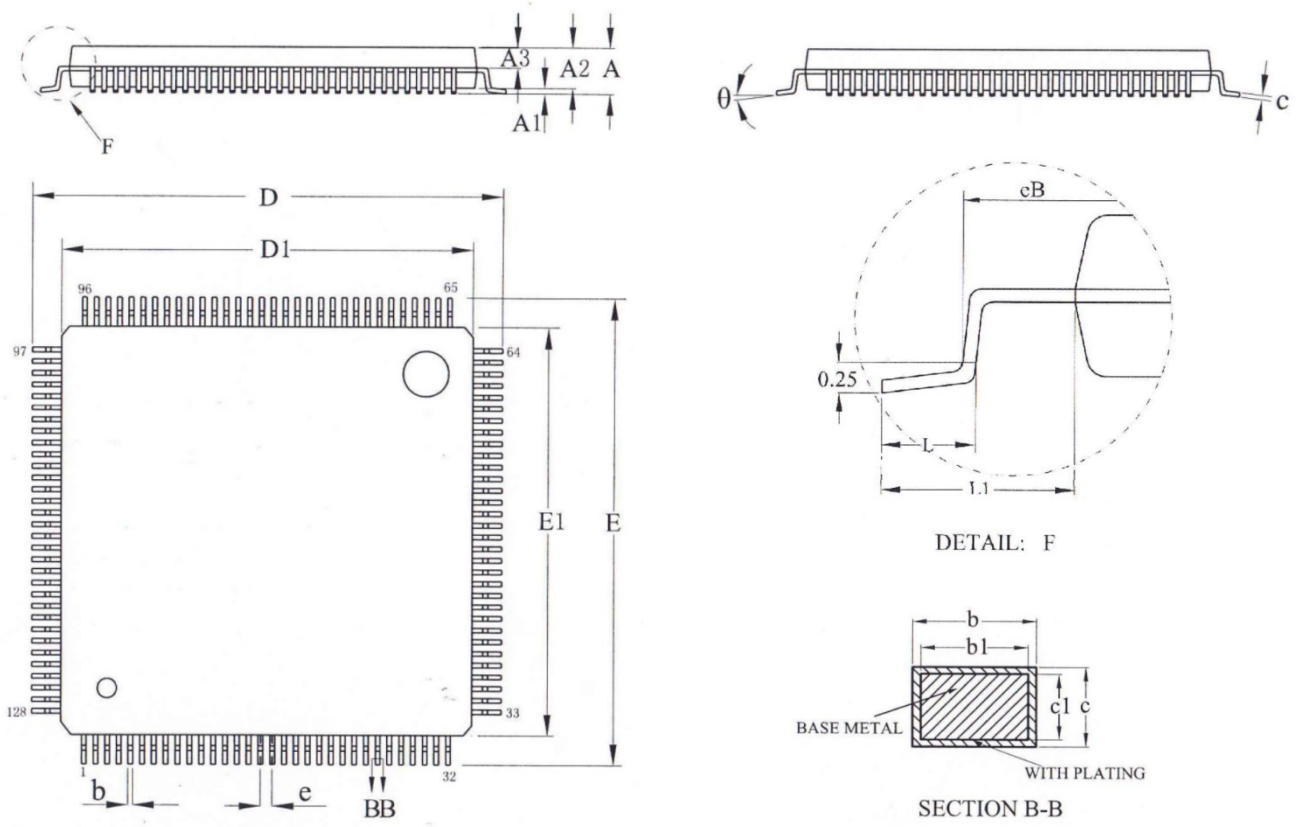


Figure 18-2：LQFP-100Pin Outline

Table 18-2：LQFP-100Pin Dimension

Symbol	Millimeter			Symbol	Millimeter		
	Min.	Nom.	Max		Min.	Nom.	Max
A	-	-	1.60	**D1**	13.9	14.0	14.1
A1	0.05	-	0.15	**E**	15.8	16.0	16.2
A2	1.35	1.40	1.45	**E1**	13.9	14.0	14.1
A3	0.59	0.64	0.69	**eB**	15.05	-	15.35
b	0.18	-	0.26	**e**	0.50BSC		
b1	0.17	0.20	0.23	**L**	0.45	-	0.75
c	0.13	-	0.17	**L1**	1.00REF		
c1	0.12	0.13	0.14	**θ**	0		7
D	15.8	16.00	16.2				

18.3 LT7580（QFN-80pin）

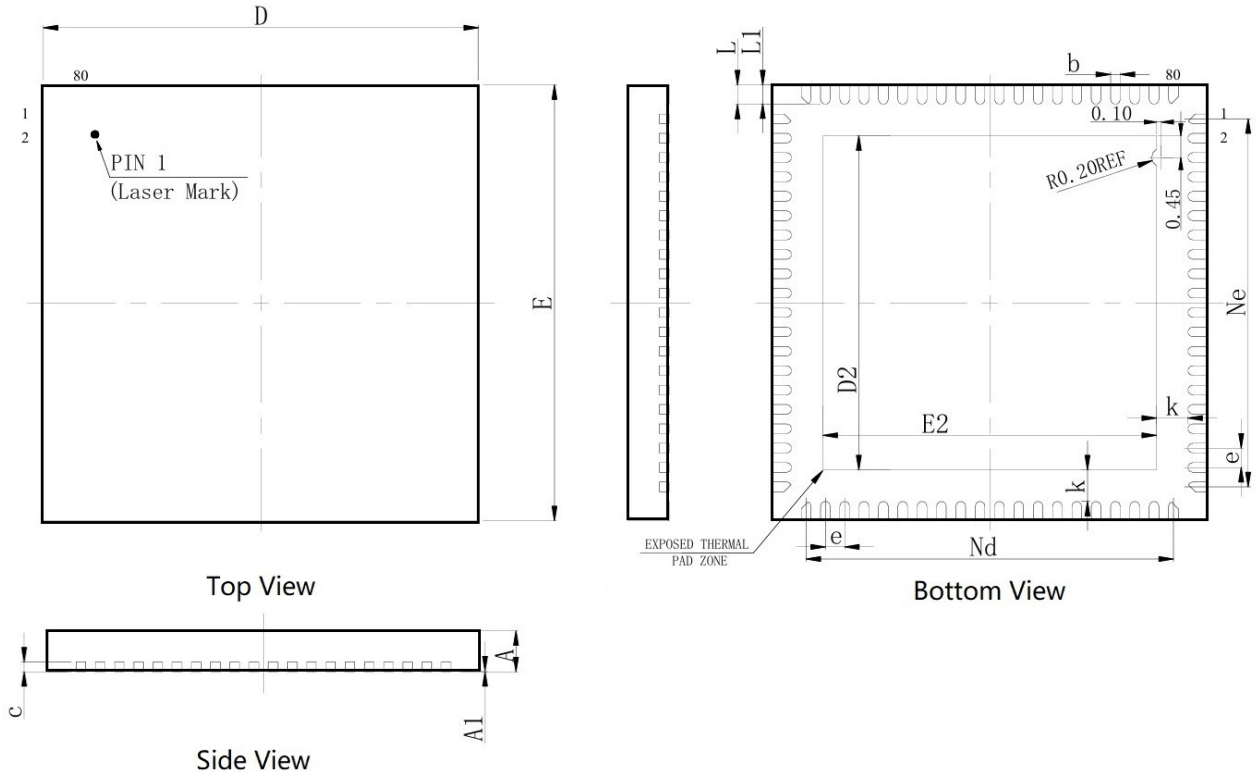


Figure 18-3：QFN-80Pin Outline

Note: When designing the PCB layout, the heat pad of LT7580's back (Thermal Pad Zone) must be directly grounded. Refer to Section 18.4 for more detail.

Table 18-3：QFN-80Pin Dimension

Symbol	Millimeter			Symbol	Millimeter		
	Min.	Nom.	Max		Min.	Nom.	Max
A	0.80	0.85	0.9	Ne	7.60BSC		
A1	0	0.02	0.05	E	8.90	9.00	9.10
b	0.15	0.20	0.25	E2	5.4	5.5	5.6
c	0.203REF			L	0.35	0.40	0.45
D	8.90	9.00	9.10	L1	0.29	0.39	0.49
D2	5.4	5.5	5.6	K	1.35REF		
e	0.40BSC			h	0.30	0.35	0.40
Nd	7.60BSC						

18.4 LT7580 Thermal Pad Layout Design Reference

LT7580 adopts QFN packaging, with ground (GND) heat dissipation pads on the back of the chip. In order to achieve better heat dissipation and reduce soldering risks, it is recommended to divide the PCB copper foil surface of the bottom solder pad of the LT7580 into four or more small solder surfaces (square or circular) when designing the PCB layout, and set the interval between each solder surface to~0.8mm to avoid incomplete soldering caused by the use of complete solder surfaces with the same or even larger size than the LT7580 solder pad, or chip deformation and poor contact caused by the PCB and the chip solder pad pulling after solder cooling.

The correct PCB pad layout is shown in the following example. The light yellow area in the middle is the grounding pad at the bottom of LT7580, and the gray area is the PCB grounding small pad (welding surface). Each pad can be grounded 1-2 times through a through-hole.

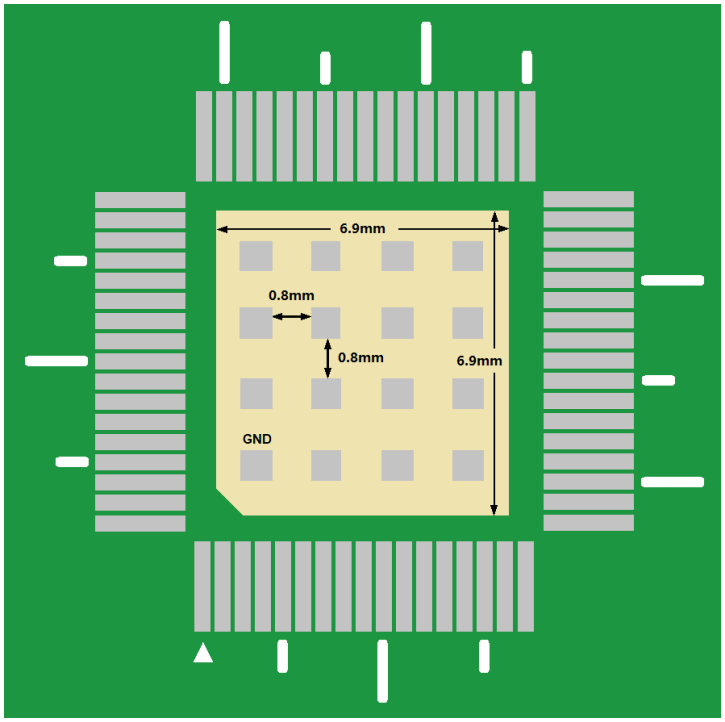


Figure 18-4：LT7580 Thermal Pad Layout Design Reference

19. Reference Schematics

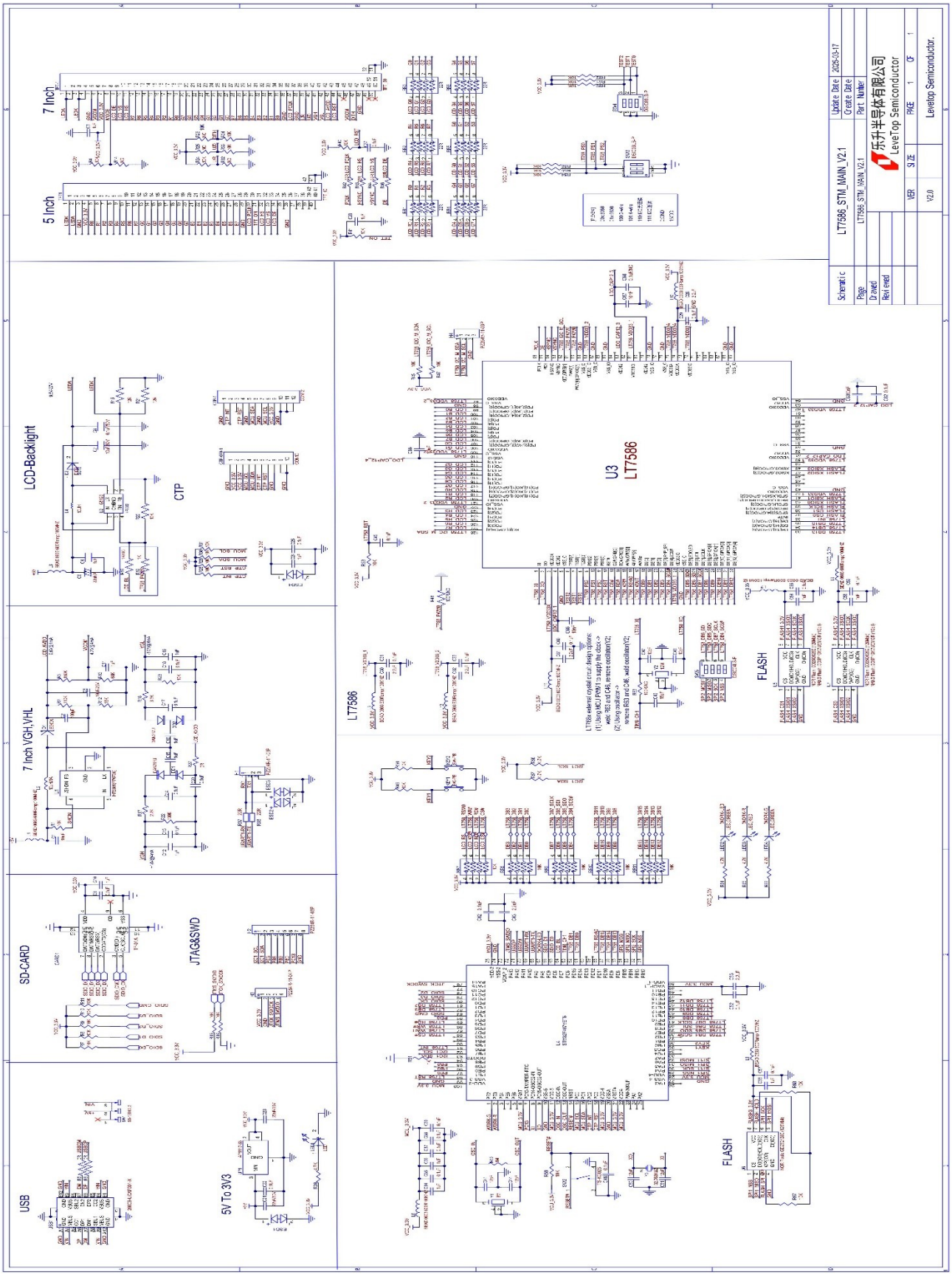


Figure 19-1：LT7586 Reference Schematics

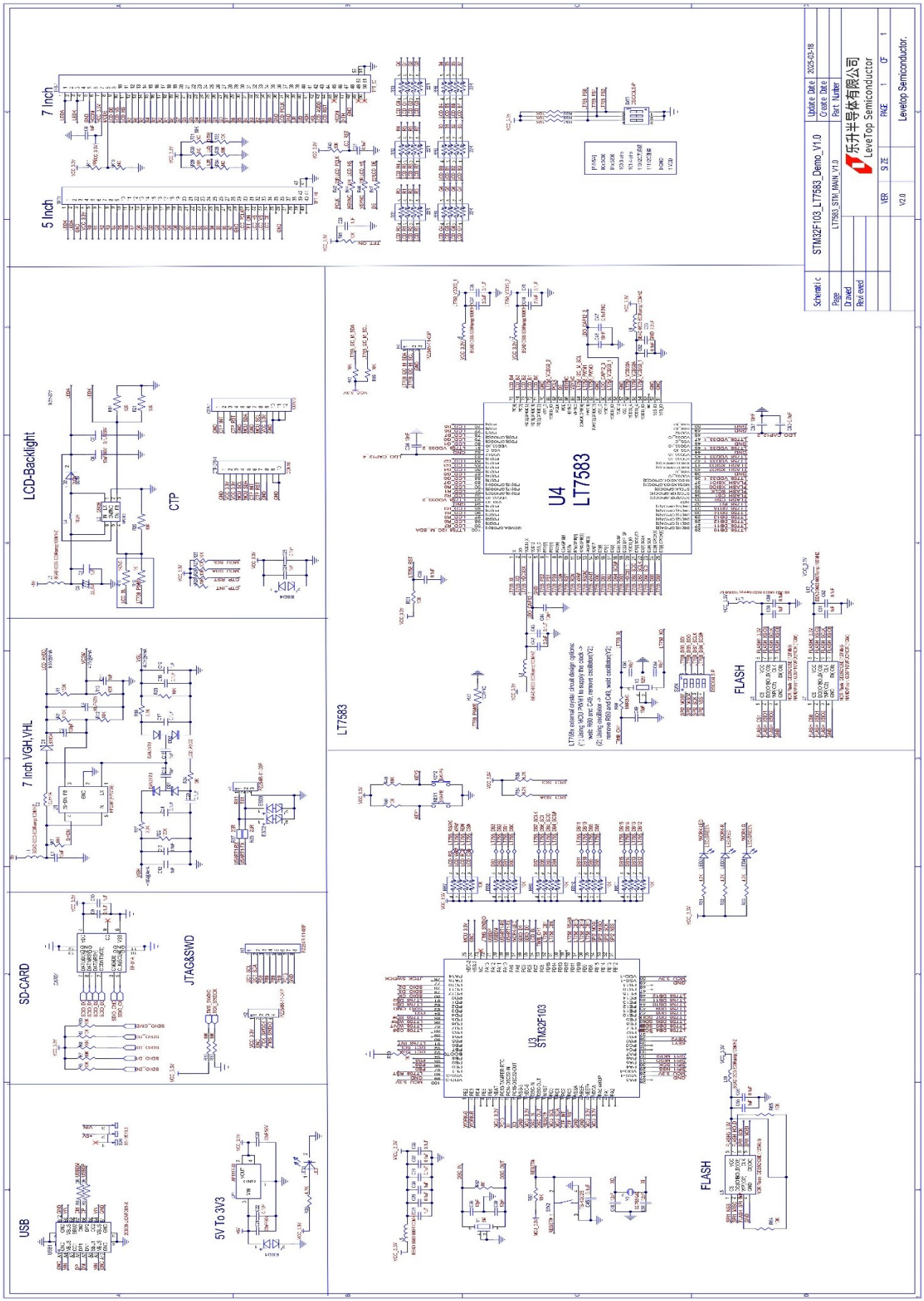


Figure 19-2: LT7583 Reference Schematics

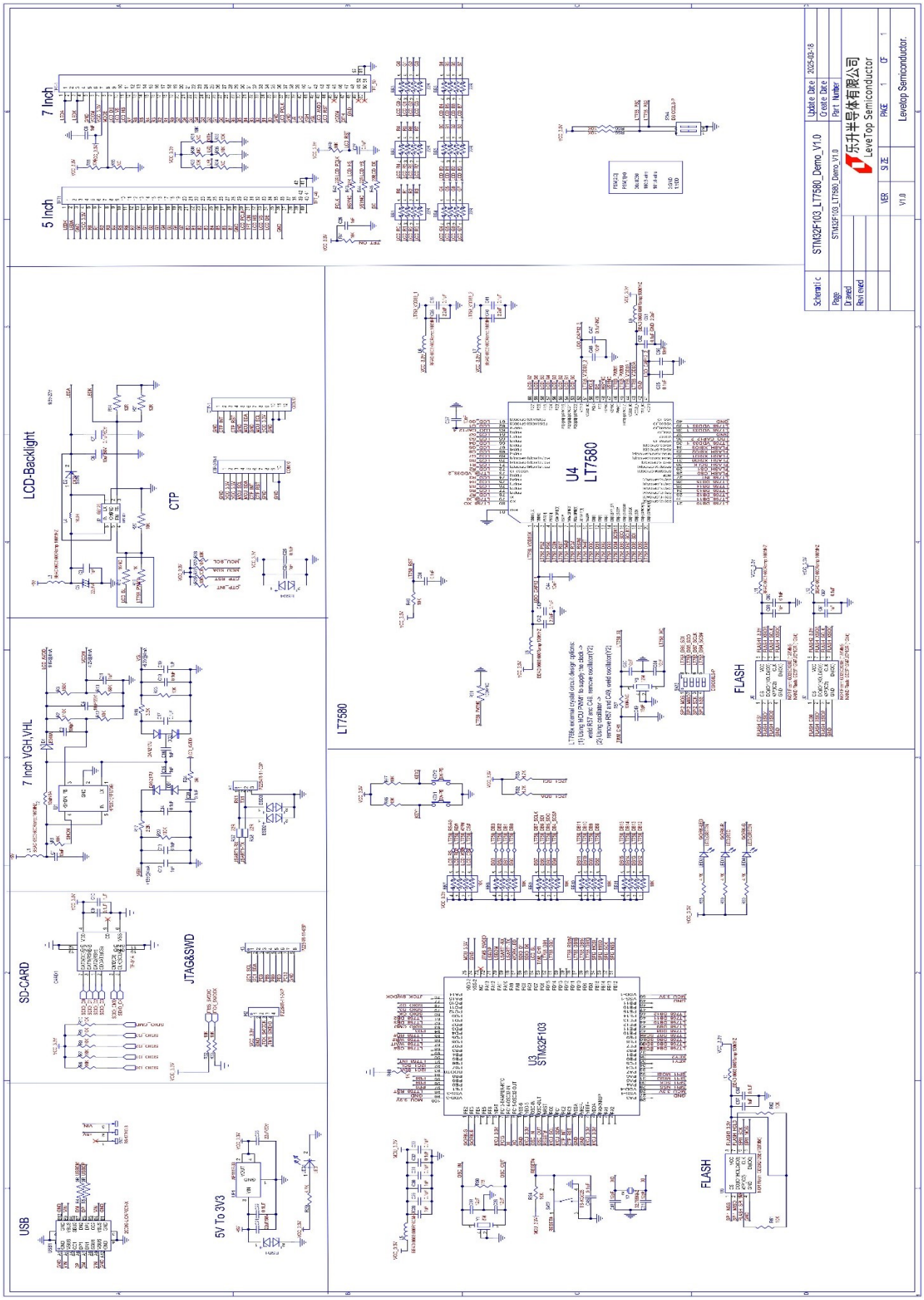


Figure 19-3：LT7580 Reference Schematics

