

# ER-TFTMC070-4

## TFT LCD Module Datasheet



## EastRising Technology Co., Limited

**Attention:**

Please pay more attention to "INSPECTION CRITERIA" in this datasheet. We assume you already agree with these criterions when you place an order with us. No more recommendations.

| REV | Description         | Release Date |
|-----|---------------------|--------------|
| 1.0 | Preliminary Release | Apr-08-2021  |
|     |                     |              |
|     |                     |              |

## Content

|   |    |
|---|----|
| Content.....  | 2  |
| Order Information .....   | 5  |
| 1.1 Order Number .....  | 5  |
| 1.2 Image .....   | 5  |
| Specification .....   | 6  |
| 2.1 Display Specification .....                                     | 6  |
| 2.2 Mechanical Specification .....                                  | 6  |
| 2.3 Electrical Specification .....                                  | 6  |
| 2.4 Optical Specification .....                                     | 6  |
| Outline Drawing .....   | 7  |
| 3.1 ER-TFTMC070-4 with No Touch Panel Outline Drawing .....         | 7  |
| 3.2 ER-TFTMC070-4 with Resistive Touch Panel Outline Drawing .....  | 8  |
| 3.3 ER-TFTMC070-4 with Capacitive Touch Panel Outline Drawing ..... | 9  |
| 4. Electrical Spec.....   | 10 |
| 4.1 Pin Configuration .....   | 10 |
| 4.2 Jump Point Description.....                                     | 12 |
| 4.3 Absolute Maximum Ratings .....                                  | 12 |
| 4.4 Electrical Characteristics.....                                 | 12 |
| 5. Function Description .....                                       | 13 |
| 5.1 Clock and Reset .....   | 13 |
| 5.1.1 Clock.....  | 13 |
| 5.1.2 Reset .....   | 15 |
| 5.2 Host Interface .....  | 16 |
| 5.2.1 Parallel Host Interface .....                                 | 18 |
| 5.2.2 Serial Host Interface .....                                   | 21 |
| 5.2.3 Display Input Data Format.....                                | 25 |
| 5.3 Display Memory.....   | 30 |
| 5.3.1 Display RAM Data Structure .....                              | 31 |
| 5.3.2 Color Palette RAM.....  | 33 |
| 5.4 LCD Interface .....   | 34 |
| 5.5 Display Function .....  | 36 |
| 5.5.1 Color Bar.....  | 36 |
| 5.5.2 Main Window .....   | 36 |
| 5.5.3 Picture-In-Picture (PIP) .....                                | 40 |
| 5.5.4 Image Rotate and Mirror .....                                 | 42 |
| 5.6 Geometric Drawing Engine .....                                  | 45 |
| 5.6.1 Drawing Circle and Ellipse .....                              | 45 |
| 5.6.2 Drawing Curve.....  | 46 |
| 5.6.3 Drawing Rectangle.....  | 47 |
| 5.6.4 Draw Line.....  | 48 |
| 5.6.5 Drawing Triangle.....   | 49 |
| 5.6.6 Drawing Rounded-Rectangle.....                                | 50 |
| 5.7 Block Transfer Engine (BTE) .....                               | 51 |
| 5.7.1 BTE Basic Settings.....                                       | 53 |
| 5.7.2 Color Palette RAM.....  | 54 |
| 5.7.3 BTE Operation Overview .....                                  | 55 |
| 5.7.4 BTE Memory Access Method .....                                | 56 |
| 5.7.5 BTE Chroma Key (Transparency Color) Function .....            | 56 |
| 5.7.6 BTE Operation Detail.....                                     | 57 |
| 5.8 Display Text.....   | 77 |
| 5.8.1 Internal CGROM .....  | 77 |
| 5.8.2 User-defined Character Graphic (UCG).....                     | 80 |
| 5.8.3 Character Rotation by 90 Degree.....                          | 88 |
| 5.8.4 Size Enlargement .....  | 88 |

|  |     |
|--|-----|
| 5.8.5 Background Transparency.....                             | 89  |
| 5.8.6 Character Full-Alignment .....                           | 89  |
| 5.8.7 Automatic Line Feed .....                                | 90  |
| 5.8.8 Cursor .....   | 90  |
| 5.9 Pulse Width Modulation (PWM).....                          | 94  |
| 5.9.1 PWM Clock Source .....                                   | 94  |
| 5.9.2 PWM Output.....  | 95  |
| 5.10 I2C Master .....  | 97  |
| 5.11 Keypad-Scan .....   | 100 |
| 5.11.1 Keypad-scan Operation .....                             | 100 |
| 5.12 GPIO Interface .....                                      | 104 |
| 5.13 Power Management .....                                    | 105 |
| 5.13.1 Normal Mode.....  | 105 |
| 5.13.2 Standby Mode.....                                       | 105 |
| 5.13.3 Suspend Mode.....                                       | 106 |
| 5.13.4 Sleep Mode .....  | 106 |
| 5.14 Register Description.....                                 | 107 |
| 5.14.1 Status Register .....                                   | 107 |
| 5.14.2 Configuration Registers .....                           | 109 |
| 5.14.3 PLL Setting Register.....                               | 114 |
| 5.14.4 Interrupt Control Register .....                        | 116 |
| 5.14.5 LCD Display Control Registers.....                      | 121 |
| 5.14.6 Geometric Engine Control Registers .....                | 132 |
| 5.14.7 PWM Control Registers .....                             | 141 |
| 5.14.8 Bit Block Transfer Engine (BTE) Control Registers ..... | 144 |
| 5.14.9 Text Engine Registers .....                             | 152 |
| 5.14.10 Power Management Control Register.....                 | 157 |
| 5.14.11 Display RAM Control Register .....                     | 158 |
| 5.14.12 I2C Master Register .....                              | 162 |
| 5.14.13 GPIO Register .....                                    | 164 |
| 5.14.14 Keypad-scan Control Register .....                     | 166 |
| 6. Inspection Criteria .....                                   | 168 |
| 6.1 Acceptable Quality Level .....                             | 168 |
| 6.2 Definition of Lot.....                                     | 168 |
| 6.3 Condition of Cosmetic Inspection .....                     | 168 |
| 6.4 Module Cosmetic Criteria.....                              | 169 |
| 6.5 Screen Cosmetic Criteria (Non-Operating).....              | 170 |
| 6.6 Screen Cosmetic Criteria (Operating) .....                 | 171 |
| 7. Precautions for Using.....                                  | 173 |
| 7.1 Handling Precautions .....                                 | 173 |
| 7.2 Power Supply Precautions .....                             | 173 |
| 7.3 Operating Precautions.....                                 | 174 |
| 7.4 Mechanical/Environmental Precautions .....                 | 174 |
| 7.5 Storage Precautions.....                                   | 174 |
| 7.6 Others.....  | 174 |
| 8. Using LCD Modules .....                                     | 175 |
| 8.1 Liquid Crystal Display Modules .....                       | 175 |
| 8.2 Installing LCD Modules.....                                | 175 |
| 8.3 Precaution for Handling LCD Modules .....                  | 175 |
| 8.4 Electro-Static Discharge Control.....                      | 176 |
| 8.5 Precaution for Soldering to Eastrising LCM .....           | 176 |
| 8.6 Precaution for Operation .....                             | 176 |
| 8.7 Limited Warranty.....                                      | 177 |
| 8.8 Return Policy .....  | 177 |
| 9. Image Sticking .....  | 178 |
| 9.1 What is Image Sticking? .....                              | 178 |

|   |     |
|---|-----|
| 9.2 What causes Image Sticking? .....                           | 178 |
| 9.3 How to Avoid Image Sticking? .....                          | 179 |
| 9.4 How to Fix the Image Sticking? .....                        | 179 |
| 9.5 Is Image Sticking Covered by Eastrising RMA Warranty? ..... | 179 |

## 1. Order Information

### 1.1 Order Number

| Order Number  | Description                                       |
|---------------|---|
| ER-TFTMC070-4 | 7" TFT LCD Display with Controller Board          |
| ER-TP070-1    | 7 " Resistive Touch Panel with Connector Type FPC |

### 1.2 Image



← ER-TFTMC070-4 with No Touch Panel

ER-TFTMC070-4 with Resistive Touch Panel →



## 2. Specification

### 2.1 Display Specification

| Item                  | Standard Value                               | Unit   |
|-----------------------|--|--------|
| Display Format        | 800x480                                      | Pixels |
| Display Connector     | FPC  | --     |
| FPC Connector         | 40 Pin 1.0mm Pitch Top Contact FPC Connector | --     |
| Operating Temperature | -20 ~ +70                                    | °C     |
| Storage Temperature   | -30 ~ +80                                    | °C     |
| Touch Panel Optional  | Yes  | --     |
| Sunlight Readable     | No   | --     |

### 2.2 Mechanical Specification

| Item                              | Standard Value       | Unit |
|-----------------------------------|----------------------|------|
| Diagonal Size                     | 7                    | Inch |
| Outline Dimension with FPC Folded | 164.90(W) x100.00(H) | mm   |
| Visual Area                       | 156.56(W)x88.96(H)   | mm   |
| Active Area                       | 154.08(W)x85.92(H)   | mm   |
| Dot Pitch                         | 0.192(W)x0.179(H)    | mm   |

### 2.3 Electrical Specification

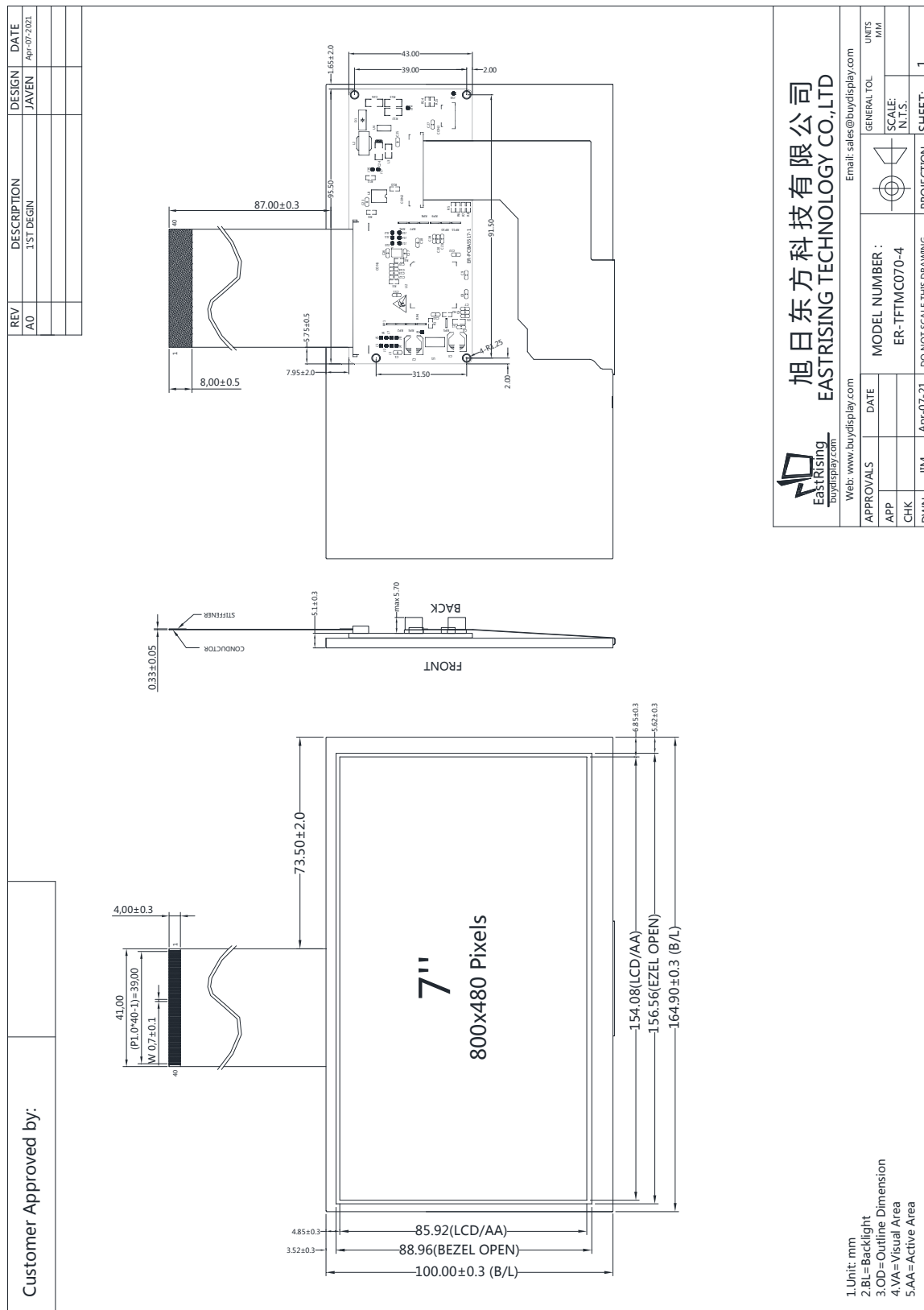
| Item                | Standard Value  | Unit |
|---------------------|---|------|
| IC Package          | COG   | --   |
| Interface           | 8080/6800 8-bit/16-bit Parallel, 3-wire,4-wire SPI ,I2C | --   |
| Response Time (Typ) | 12  | MS   |

### 2.4 Optical Specification

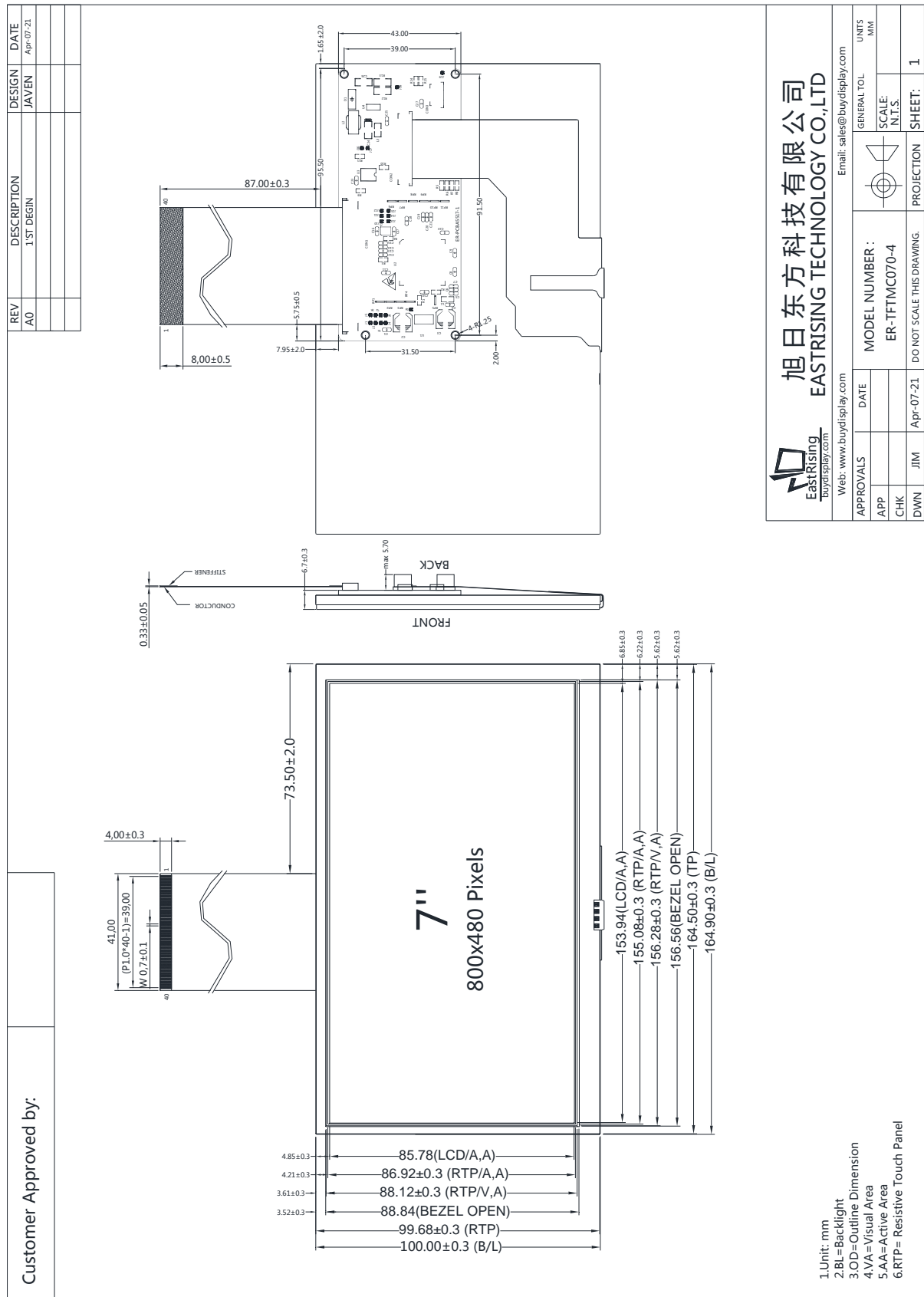
| Item                 | Standard Value                       | Unit  |
|----------------------|--------------------------------------|-------|
| LCD Type             | TFT-LCD / Transmissive / TN          | --    |
| Viewing Angle Range  | Left:60 , Right:60 , Up:75 , Down:60 | Deg   |
| Colors               | 65k/16.7M                            | --    |
| Contrast Ratio (Typ) | 500:1                                | --    |
| Brightness (Typ)     | 200                                  | cd/m2 |

### 3. Outline Drawing

#### 3.1 ER-TFTMC070-4 with No Touch Panel Outline Drawing



## 3.2 ER-TFTMC070-4 with Resistive Touch Panel Outline Drawing





3.3 ER-TFTMC070-4 with Capacitive Touch Panel Outline Drawing

Not available.

## 4. Electrical Spec

### 4.1 Pin Configuration

| Pin No | Symbol                               | Descriptions  |
|--------|--------------------------------------|---|
| 1      | VSS                                  | Ground  |
| 2      | VSS                                  | Ground  |
| 3      | VDD                                  | Power Supply  |
| 4      | VDD                                  | Power Supply  |
| 5      | Parallel Mode<br>E <sub>/</sub> RD   | Enable/Read Enable<br>When MCU interface (I/F) is 8080 series, this pin is used as RD# signal (Data Read) , active low.<br>When MCU I/F is 6800 series, this pin is used as EN signal (Enable), active high   |
|        | Serial Mode<br>/SCS                  | Serial Mode Chip Select, Low active chip select pin.<br>Chip select pin for 3-wire , 4-wire serial .<br>XI2CA[4], I2C device address bit [4],internal pull-high<br>This is a multiplex pin that share with Parallel Host Data Bus DB[4].                                |
| 6      | Parallel Mode<br>R/W <sub>/</sub> WR | Write/Read-Write<br>When MCU interface is 8080 series, this pin is used as WR# signal (data write) , active low.<br>When MCU interface is 6800 series, this pin is used as RW# signal (data Read/Write control) . Active high for read and active low for write.        |
|        | Serial Mode<br>SDO                   | 4-wire SPI interface: SDO. Data output.<br>3-wire SPI interface: SDA. Bi-direction data.<br>IIC interface: I2C device address bit [5], internal pull-high<br>This is a multiplex pin that share with Parallel Host Data Bus DB[5].                                      |
| 7      | Parallel Mode /CS                    | Parallel Mode Chip Select Input<br>Low active chip select pin.  |
|        | Serial Mode<br>SDI                   | IIC data /4-wire SPI Data Input<br>4-wire SPI interface: SDI. Data input for serial interface<br>3-wire SPI interface: NC, please connect it to GND.<br>IIC interface: SDA. Bi-direction data.<br>This is a multiplex pin that share with Parallel Host Data Bus DB[6]. |
| 8      | Parallel Mode<br>RS                  | Command / Data Select Input<br>The pin is used to select data/command cycle. RS = 1, data Read/Write cycle is selected. RS = 0, status read/command write cycle is selected.  |
|        | Serial Mode<br>SCLK                  | SPI Clock<br>3-wire, 4-wire Serial or IIC interface clock<br>This is a multiplex pin that share with Parallel Host Data Bus DB[7].  |
| 9      | WAIT                                 | Wait Signal Output<br>When high, it indicates that the chip is ready to transfer data. When low,then microprocessor is in wait state.   |
| 10     | INT                                  | Interrupt Signal Output   |

|       |            |   |
|-------|------------|---|
|       |            | The interrupt output for MCU to indicate the status.  |
| 11    | /RESET     | Reset Signal Input<br>This is a active low Reset pin for chip. To avoid noise interfere and cause fake reset behavior, this pin is active at least 256 OSC clocks.  |
| 12    | NC         | No Connection.  |
| 13    | VSS        | Ground  |
| 14    | BL_CONTROL | Backlight control signal input. When using the internal PWM signal this pin floating.   |
| 15-30 | DB0-DB15   | Host Data Bus<br>These are data bus for data transfer between Host and chip.<br>DB[15:8] will become GPIO (GPIOA[7:0]) when parallel Host 8080/6800 16-bits data bus mode doesn't set.<br>DB[7:0] are multiplex pins that share with Serial Host control pins. When serial host mode set then DB[7:0] are defined as the control pins of serial host. Please refer to Host Interface section. |
| 31    | VSS        | Ground  |
| 32    | RTP_/CS    | Chip Select Input. Active Low Logic Input. This input provides the dual function of initiating conversions on the XPT2046 and also enables the serial input/output register.  |
| 33    | RTP_/PEN   | Pen Interrupt. CMOS logic open-drain output   |
| 34    | RTP_DIN    | Data In. Logic input. Data to be written to the XPT2046 control register is provided on this input and is clocked into the register on the rising edge of DCLK (see the Control Register section).  |
| 35    | RTP_SCLK   | External Clock Input. Logic Input. DCLK provides the serial clock for accessing data from the part. This clock input is also used as the clock source for the XPT2046 conversion process.   |
| 36    | RTP_DOUT   | Data Out. Logic Output. The conversion result from the XPT2046 is provided on this output as a serial data stream. The bits are clocked out on the falling edge of the DCLK input. This output is high impedance when CS is high.   |
| 37    | VDD        | Power Supply  |
| 38    | VDD        | Power Supply  |
| 39    | VSS        | Ground  |
| 40    | VSS        | Ground  |

Note: CTP means Capacitive Touch Panel. RTP means 4-wire Resistive Touch Panel

#### 4.2 Jump Point Description

| Function Descriptions   | Jump Method  |
|-------------------------|--|
| Power Supply Switch     | Vdd=3.3V Power Supply : J9 Short<br>Vdd=5V Power Supply : J9 Open  |
| 8080 Parallel Interface | J1,J2,J3,J4,J13,J14,J15 Short<br>J5,J6,J7,J8,J10,J11,J12 Open  |
| 6800 Parallel Interface | J1,J2,J3,J4,J11,J13,J15 Short<br>J5,J6,J7,J8,J10,J12,J14 Open  |
| I2C Interface           | J5,J6,J7,J8,J10,J11,J15 Short<br>J1,J2,J3,J4,J12,J13,J14 Open  |
| 3-wire Serial Interface | J5,J6,J7,J8,J10,J14,J15 Short<br>J1,J2,J3,J4,J11,J12,J13 Open  |
| 4-wire Serial Interface | J5,J6,J7,J8,J10,J12,J14 Short<br>J1,J2,J3,J4,J11,J13,J15 Open  |
| Backlight Control       | J16 Short,J17 Open: Select Backlight Control Signal with External Input<br>J16 Open,J17 Short: Select Backlight Control Signal with internal PWM |

#### 4.3 Absolute Maximum Ratings

| Item                  | Symbol | Min  | Typ | Max          | Unit |
|-----------------------|--------|------|-----|--------------|------|
| Power Supply Voltage  | VCC    | -0.3 | -   | 4            | V    |
| Operating Temperature | Top    | -20  | -   | 70           | °C   |
| Storage Temperature   | TST    | -30  | -   | 80           | °C   |
| Humidity              | RH     | -    |     | 90%(Max60°C) | RH   |

#### 4.4 Electrical Characteristics

| Item                     | Symbol    | Min. | Typ. | Max. | Unit |
|--------------------------|-----------|------|------|------|------|
| Power Supply Voltage     | VDD       | 3.0  | 3.3  | 3.6  | V    |
|                          |           | 4.8  | 5.0  | 5.5  | V    |
| Logic Signal I/O Voltage | VDDIO     | 3.0  | 3.3  | 3.6  | V    |
| Input Voltage 'H' Level  | VIH       | 2.0  | -    | 3.6  | V    |
| Input Voltage 'L' Level  | VIL       | -0.3 | -    | 0.8  | V    |
| Output Voltage 'H' Level | VOH       | 2.4  | -    | 3.6  | V    |
| Output Voltage 'L' Level | VCL       | 0    | -    | 0.4  | V    |
| Module Current           | IDD(3.3V) | --   | --   | 450  | mA   |
|                          | IDD(5.0V) | --   | --   | 280  | mA   |

## 5. Function Description

### 5.1 Clock and Reset

#### 5.1.1 Clock

ER-TFTMC070-4 embedded three PLL circuit to generate three clock source for internal circuit operation:

- CPLL : Provide **CCLK** for Host interface, BTE Engine, Graphics Engine, and Text DMA data transfer etc...
- MPLL : Provide **MCLK** for internal Display RAM
- PPLL : Provide **PCLK** for TFT-LCD's ScanClock.

The three PLL are operation independent. The PLL output frequency is calculated from the following formula:

$$F_{OUT} = X_I \times (N / R) \div OD$$

In above formula,  $X_I$  is the external Oscillator / Clock input. The input frequency " $X_I/R$ " is no less than 1MHz, and the default value is 1MHz. "R" is Input Divider Ratio, it between 2 ~ 31. "OD" is Output Divider Ratio that must be 1, 2 or 4. "N" is the Feedback Divider Ratio of Loop that indicated by 9bits which between 2 ~ 511.

Table 1-1: PLL Register Setting (1)

| R[4:0] | Input Divider Ratio (R) | N[8:0]    | Feedback Divider Ratio (N) |
|--------|-------------------------|-----------|----------------------------|
| 00010  | 2                       | 000000010 | 2                          |
| 00011  | 3                       | 000000011 | 3                          |
| 00101  | 4                       | 000000101 | 4                          |
| :      | :                       | :         | :                          |
| :      | :                       | :         | :                          |
| :      | :                       | :         | :                          |
| 11101  | 29                      | 111111101 | 509                        |
| 11110  | 30                      | 111111110 | 510                        |
| 11111  | 31                      | 111111111 | 511                        |

Table 1-2: PLL Register Setting (2)

| OD[1:0] | Input Divider Ratio (OD) |
|---------|--------------------------|
| 00      | 1                        |
| 01      | 2                        |
| 10      | 3                        |
| 11      | 4                        |

For example,  $X_I$  is 10MHz, R[4:0] is 01010 (i.e. 10), N[8:0] is 100000000 (i.e. 256), OD[1:0] is 11 (i.e. 4), then:

$$F_{OUT} = 10\text{MHz} \times (256 / 10) \div 4 = 64\text{MHz}$$

The design rule of three clock are: :

1.  $CCLK \times 2 \geq MCLK \geq CCLK$
2.  $CCLK \geq PCLK \times 1.5$

Usually TFT manufacturers will be based on their TFT characteristics to inform the best display of Pixel Clock (PCLK). Therefore, user can setup the register according to the requirements of the PCLK. And according to the above rule to setup CCLK and MCLK.

According to the different resolution of LCD panel, the PLL output should generate different clock frequency. For example, if LCD panel resolution is 640x480, the recommend values are: PCLK = 20MHz, MCLK = 40MHz, CCLK = 40MHz, then the register are setting as following:

|   |   |
|---|---|
| $PCLK = XI \times (N / R) \div OD$ $= 10MHz \times (80 / 10) \div 4$ $= 20MHz$  | REG[05h] : OD = 11b, R = 01010b<br>REG[06h] : N = 01010000b |
| $MCLK = XI \times (N / R) \div OD$ $= 10MHz \times (160 / 10) \div 4$ $= 40MHz$ | REG[07h] : OD = 11b, R = 01010b<br>REG[08h] : N = 10100000b |
| $CCLK = XI \times (N / R) \div OD$ $= 10MHz \times (160 / 10) \div 4$ $= 40MHz$ | REG[09h] : OD = 11b, R = 01010b<br>REG[0Ah] : N = 10100000b |

Another example, if resolution is 800x480, the recommend values are: PCLK = 25MHz, MCLK = 50MHz, CCLK = 50MHz. We can setup the registers values as following.

|   |   |
|---|---|
| $PCLK = XI \times (N / R) \div OD$ $= 10MHz \times (100 / 10) \div 4$ $= 25MHz$ | REG[05h] : OD = 11b, R = 01010b<br>REG[06h] : N = 01100100b |
| $MCLK = XI \times (N / R) \div OD$ $= 10MHz \times (200 / 10) \div 4$ $= 50MHz$ | REG[07h] : OD = 11b, R = 01010b<br>REG[08h] : N = 11001000b |
| $CCLK = XI \times (N / R) \div OD$ $= 10MHz \times (200 / 10) \div 4$ $= 50MHz$ | REG[09h] : OD = 11b, R = 01010b<br>REG[0Ah] : N = 11001000b |

Table 1-3: PLL Register Setting Example

| Registers       | 640x480   | 800x480   |
|-----------------|-----------|-----------|
| REG[05h], PLLC1 | 11010100b | 11010100b |
| REG[06h], PLLC2 | 01010000b | 01100100b |
| REG[07h], PLLC1 | 11010100b | 11010100b |
| REG[08h], PLLC2 | 10100000b | 11001000b |
| REG[09h], PLLC1 | 11010100b | 11010100b |
| REG[0Ah], PLLC1 | 10100000b | 11001000b |

## 5.1.2 Reset

### 5.1.2.1 Power-on Reset

ER-TFTMC070-4 embedded a Power-On-Reset for core system. It is an active low signal and may output to external circuits by RST# pin to synchronize whole system. When system power (3.3V) on, internal reset will active until internal power stable and then de-active after 256 OSC(X"tal Oscillator) clocks.

### 5.1.2.2 External Reset

ER-TFTMC070-4 has capability to receive external reset(RST#) event to synchronize with external system. The external reset event will be admitted when RST# keep low and stable at least 256 OSC clocks.

Before the start to access ER-TFTMC070-4, Host should check it's Status Register(STSR) bit [1], i.e. operation mode status bit, and make sure it's in "Normal operation state".

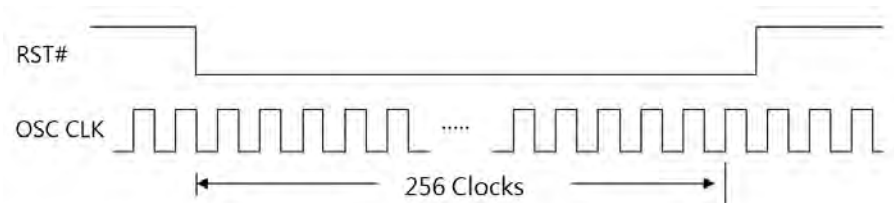


Figure 1-1: External Reset Signal

### 5.1.2.3 Software Reset

If the Host write registers REG[00h] bit0 to 1, the ER-TFTMC070-4 will be reset by software. The software reset will only reset the internal state machine of ER-TFTMC070-4, and the other registers values will not be affected or cleared. After the software reset is complete, the REG[00h] bit0 will automatically be cleared to 0.

## 5.2 Host Interface

ER-TFTMC070-4 is control by external Host and through the host interface to access ER-TFTMC070-4's Registers and Display Memory. ER-TFTMC070-4 provide 8bits or 16bits parallel mode, serial SPI mode, and I2C mode for Host's communication. These interface mode is setup by PSM[2:0] pins:

Table 2-1: Host Interface Mode

| PSM[2:0] | Host Interface                               |
|----------|--|
| 0 0 X    | 8bits or 16bits 8080 Parallel Interface Mode |
| 0 1 X    | 8bits or 16bits 6800 Parallel Interface Mode |
| 1 0 0    | 3-Wire SPI Mode                              |
| 1 0 1    | 4-Wire SPI Mode                              |
| 1 1 X    | I2C Mode                                     |

Because the different MCU interfaces cannot be used at the same time, so ER-TFTMC070-4 provides a shared pin mode. When use Serial Mode, the other parallel pins can also be set to GPIO use. Please refer to the following table:

Table 2-2: The Pin Definition of Host Interface

| 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]  |          | DB[15:0]    | SCLK        | SCLK       | SCLK      |
| DB[6]    |          |          |          |          |             | GND         | SDI        | I2C_SDA   |
| DB[5]    |          |          |          |          |             | SD          | SDO        | I2CA[5]   |
| DB[4]    |          |          |          |          |             | SCS#        | SCS#       | I2CA[4]   |
| DB[3:0]  |          |          |          |          |             | GND         | GND        | I2CA[3:0] |
| CS#      | CS#      |          | CS#      |          | GPIOB[0]    | GPIOB[0]    | GPIOB[0]   |           |
| RD#      | RD#      |          | EN       |          | GPIOB[1]    | GPIOB[1]    | GPIOB[1]   |           |
| WR#      | WR#      |          | RW#      |          | GPIOB[2]    | GPIOB[2]    | GPIOB[2]   |           |
| A0       | A0       |          | A0       |          | GPIOB[3]    | GPIOB[3]    | GPIOB[3]   |           |
| INT#     | INT#     |          | INT#     |          | INT#        | INT#        | INT#       |           |
| WAIT#    | WAIT#    |          | WAIT#    |          | --          | --          | --         |           |

When using the Parallel Host mode, 8bits or 16bits data transfer is determined by the bit0 of Register REG[01h]. When this bit0=0, then 8bits data transfer was selected. And if this bit0=1 then 16bits selected. The ER-TFTMC070-4 supports different host interfaces, and was selected by pins PSM[2:0].



Table 2-3: Host Interface Supporting List of ER-TFTMC070-4

| No. | Host Interface Mode                 | ER-TFTMC070-4 |
|-----|-------------------------------------|---------------|
| 1   | 8bits 8080 Parallel Interface Mode  | √             |
| 2   | 16bits 8080 Parallel Interface Mode | √             |
| 3   | 8bits 6800 Parallel Interface Mode  | √             |
| 4   | 16bits 6800 Parallel Interface Mode | √             |
| 5   | 3-Wire SPI Mode                     | √             |
| 6   | 4-Wire SPI Mode                     | √             |
| 7   | I2C Mode                            | √             |

## 5.2.1 Parallel Host Interface

The following are the application circuit and timing of 8080/6800 parallel interface:

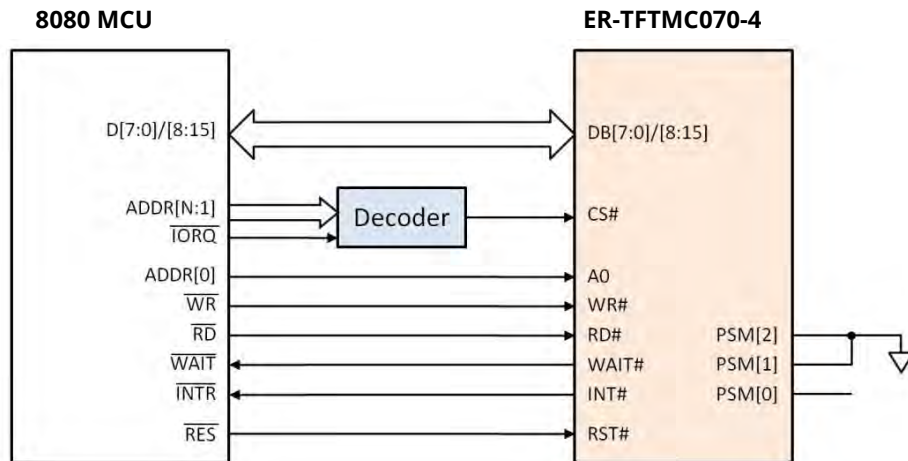


Figure 2-1: 8080 Parallel Mode Interface

The WAIT# signal is used to indicate ER-TFTMC070-4 is ready to transfer data or not. If WAIT# signal did not connect, then the Host access cycle time has to length than five CCLK clocks to avoid access fail. If the Host's Reset signal is active low, then it can connect to RST# of ER-TFTMC070-4. Of course, the RST# can also control by the I/O pin of host, or connect a RC circuit to generate a low pulse. However, either way to confirm RST#'s active cycle has to keep at least 256 system clock cycle. While using ER-TFTMC070-4, Host should first confirm the state register bit1, to know whether the ER-TFTMC070-4 in the standard operating state.

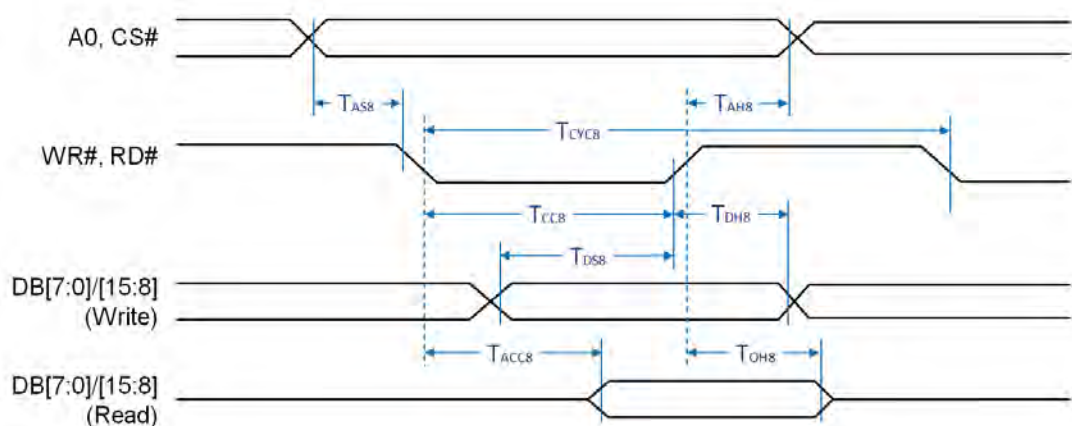


Figure 2-2: 8080 Parallel Mode Interface Timing

| Symbol                        | Parameter               | Rating |      | Unit | Note   |
|-------------------------------|-------------------------|--------|------|------|--|
|                               |                         | Min.   | Max. |      |  |
| T <sub>CYC</sub> <sub>8</sub> | Cycle Time              | 50     | --   | ns   | tc is one system clock period:<br>tc = 1/SYS_CLK |
| T <sub>CC8</sub>              | Strobe Pulse Width      | 20     | --   | ns   |  |
| T <sub>AS8</sub>              | Address Setup Time      | 0      | --   | ns   |  |
| T <sub>AH8</sub>              | Address Hold Time       | 10     | --   | ns   |  |
| T <sub>DS8</sub>              | Data Setup Time         | 20     | --   | ns   |  |
| T <sub>DH8</sub>              | Data Hold Time          | 10     | --   | ns   |  |
| T <sub>ACC</sub> <sub>8</sub> | Data Output Access Time | 0      | 20   | ns   |  |
| T <sub>OH8</sub>              | Data Output Hold Time   | 0      | 20   | ns   |  |

Table 2-3: 8080 Parallel Mode Interface Timing Parameter

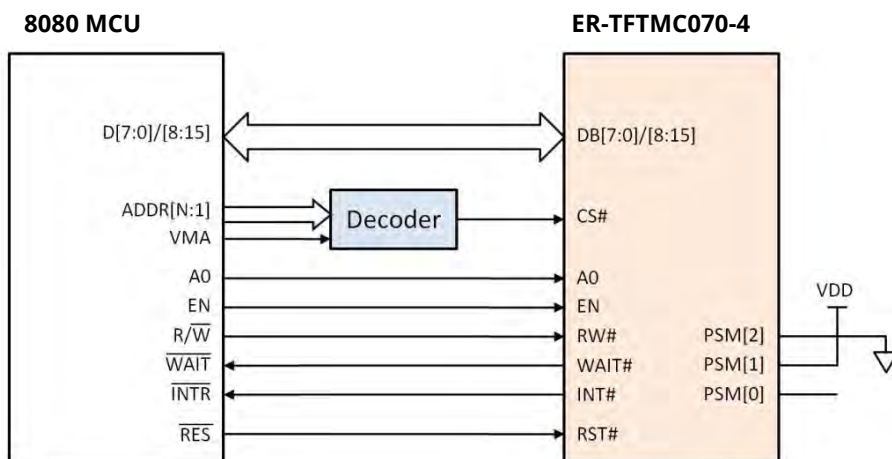


Figure 2-4: 6800 Parallel Mode Interface

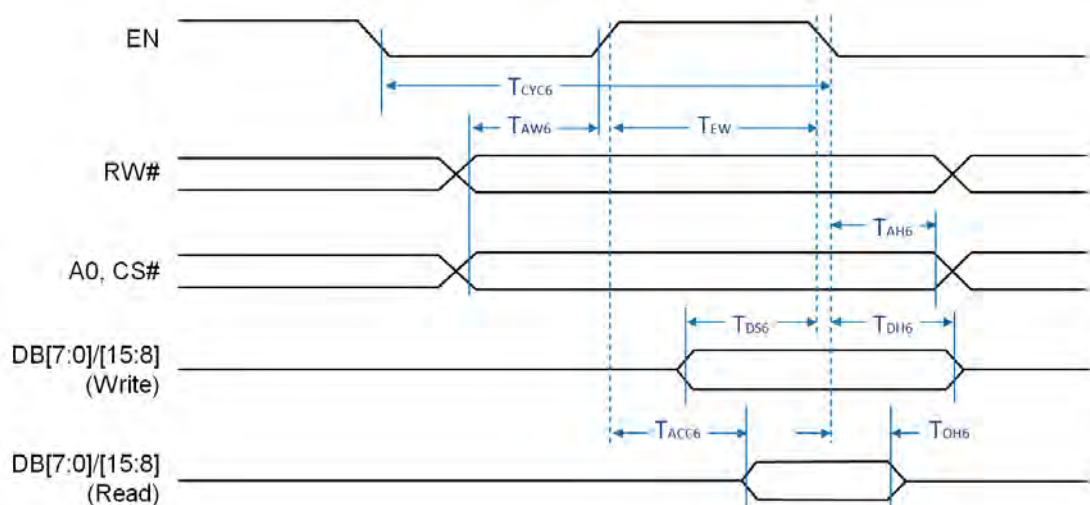


Figure 2-5: 6800 Parallel Mode Interface Timing

Table 2-6: 6800 Parallel Mode Interface Timing Parameter

| Symbol            | Parameter               | Rating |      | Unit | Note   |
|-------------------|-------------------------|--------|------|------|--|
|                   |                         | Min.   | Max. |      |  |
| TCYC <sub>6</sub> | Cycle Time              | 50     | --   | ns   | tc is one system clock period:<br>tc = 1/SYS_CLK |
| TEW               | Strobe Pulse Width      | 20     | --   | ns   |  |
| TAW <sub>6</sub>  | Address Setup Time      | 0      | --   | ns   |  |
| TAH <sub>6</sub>  | Address Hold Time       | 10     | --   | ns   |  |
| TDS <sub>6</sub>  | Data Setup Time         | 20     | --   | ns   |  |
| TDH <sub>6</sub>  | Data Hold Time          | 10     | --   | ns   |  |
| TACC <sub>6</sub> | Data Output Access Time | 0      | 20   | ns   |  |
| TOH <sub>6</sub>  | Data Output Hold Time   | 0      | 20   | ns   |  |

Host through the host interface to access ER-TFTMC070-4's Registers and Display Memory. ER-TFTMC070-4 has one Status Register and 256 Instruction Registers (i.e. .REG[00h] ~ REG[FF]). The access procedure are as following:

## Register Write:

1. Address Write: Write the Register's Address. For example, 00h i.e. REG[00h], 01h i.e. REG[01h], 02h i.e. REG[02h] .....
2. Data Write: Write Data to the Register

## Register Read:

1. Address Write: Write the Register's Address
2. Data Write: Read Data from the Register

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

## Display RAM Write:

1. Set the Active Window Registers before writing any image data.
2. Perform an register write to Graphic R/W Position Register 0, REG[5Fh]).
3. Repeat step 2 until setup all the Active Window & Graphic R/W Position Coordinates.
4. Perform an address write to point to Memory Data Port Register (REG[04h])
5. Perform data writes to fill the window. Each write to the Memory Data Port will auto-increment the internal memory address.

### 5.2.2 Serial Host Interface

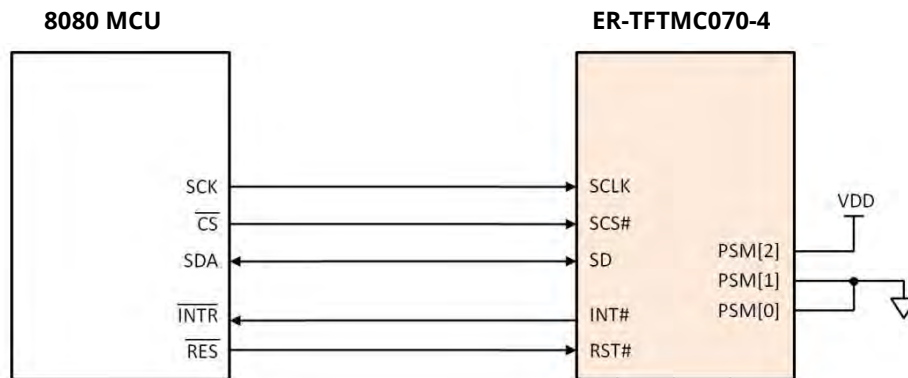


Figure 2-5: 3-Wire SPI Interface

The above circuit is the ER-TFTMC070-4's 3-Wires SPI interface with Host. SD signal is a bi-direction data pin for data access. The access timing and procedure are as below:

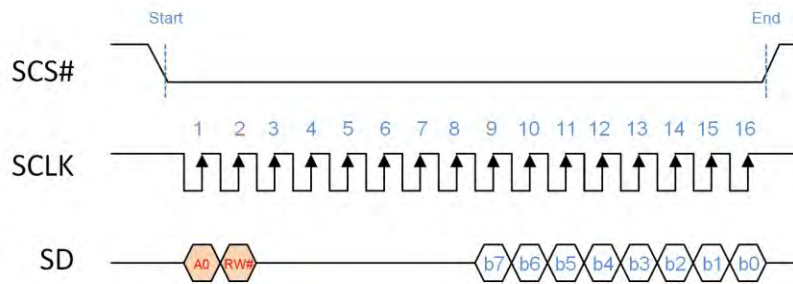


Figure 2-6: 3-Wire SPI Interface Timing

#### Status Register Read:

1. Host drive SCS#(Low) and SCLK(SPI Clock).
2. Host drive A0(Low), then drive RW#(High).
3. ER-TFTMC070-4 will drive the Data of Status Register (b7 ~ b0) at 9<sup>th</sup> ~ 16<sup>th</sup> Clock. Then Host will get the content of Status Register.

#### Write Register's Address:

1. Host drive SCS#(Low) and SCLK.
2. Host drive A0(Low), then drive RW#(Low).
3. Host drive the Register's Address (b0 ~ b7) at 9<sup>th</sup> ~ 16<sup>th</sup> Clock to ER-TFTMC070-4.

#### Write Data to Register or Memory:

1. Host drive SCS#(Low) and SCLK.
2. Host drive A0(High), then drive RW#(Low).
3. Host drive the Data at 9<sup>th</sup> ~ 16<sup>th</sup> Clock to ER-TFTMC070-4. i.e. Data will be stored in Register or Memory.

## Read Register's Data:

1. Host drive SCS#(Low) and SCLK.
2. Host drive A0(High), then drive RW#(High).
3. ER-TFTMC070-4 will drive the Data of Register at 9<sup>th</sup> ~ 16<sup>th</sup> Clock. Then Host will get the content of Register.

The 4-Wires SPI is almost same as 3-Wires. The difference is its data line input and output are separate. The interface circuit diagram and Timing are as follows:

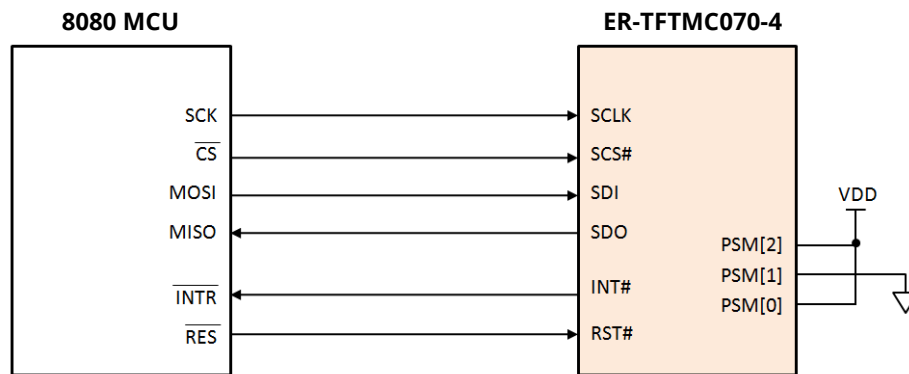


Figure 2-7: 4-Wire SPI Interface

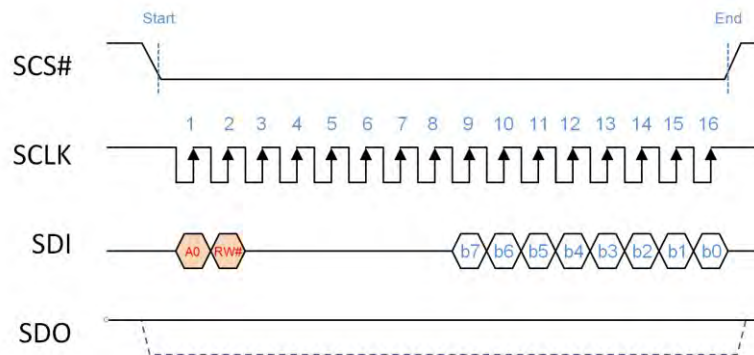


Figure 2-8: 4-Wire SPI Interface Write Timing

The above Timing diagram is the Write Cycle of 4-Wires SPI. When Host drive A0(Low) and RW#(Low), that's means Host write Register's Address. When Host drive A0(High), then RW#(Low) that's means Host write data to Register or Display RAM.

The following Timing diagram is the Read Cycle of 4-Wires SPI. When Host drive A0(Low) and RW#(High), that's means Host want to read the data of Status Register. ER-TFTMC070-4 will drive the Data of Status Register (b7 ~ b0) at 9<sup>th</sup> ~ 16<sup>th</sup> Clock. Then Host will get the data of Status Register. When Host drive A0(High), then RW#(High) that's means Host want to read the data of Command Register. ER-TFTMC070-4 will drive the Data of Command Register (b7 ~ b0) at 9<sup>th</sup> ~ 16<sup>th</sup> Clock for Host. Of course, Host will get the content of Command Register.

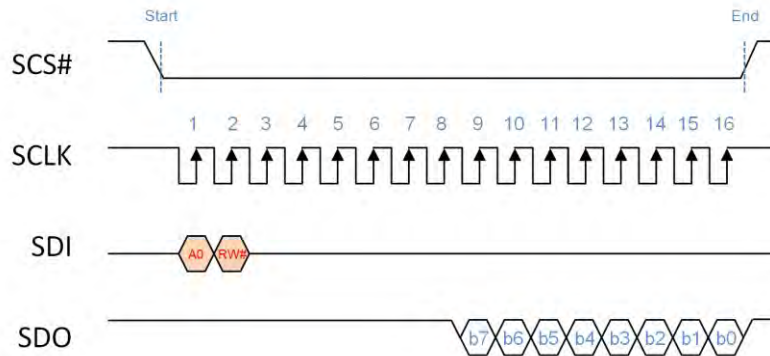
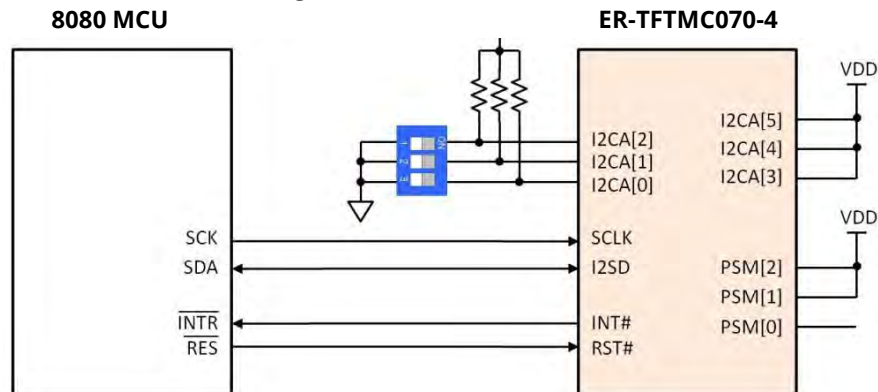


Figure 2-9: 4-Wire SPI Interface Read Timing

The serial I2C interface is also almost same as 3-Wires SPI interface. But I2C interface only need 2 wires for data transfer. The following is the application circuit of I2C interface. Signals I2CA[5:0] are used to setup ER-TFTMC070-4's Device ID, and to avoid confuse with other's I2C device. In this example circuit, I2CA[5:3] connect to VDD, and if all DIP Switch are "ON" state, then I2C Device ID is 111000b. i.e. 38h. Therefore if Host drive I2C timing with "111000b" Device ID, then ER-TFTMC070-4 will communicate in the I2C cycle time with Host.

Figure 2-10: I2C Interface



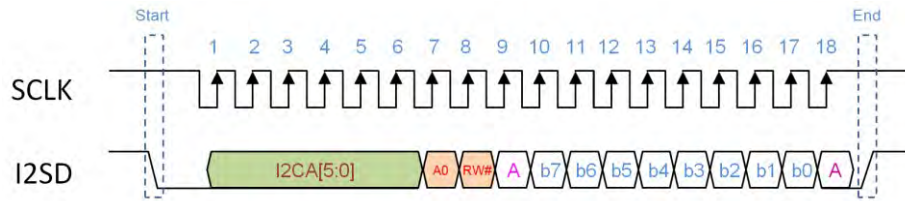


Figure 2-11: I2C Interface Timing

Figure 2-11 is the timing diagram of I2C interface. At first, Host has to know the Device ID of the I2C device. Then release the Device ID's data in the five clocks of beginning. In the example of Figure 2-10, the Device ID data is 111000. The definition of A0 and RW# are same as SPI interface. When Host release A0(High) and RW#(Low), that means Host will write data(b7 ~ b0, at 10th ~ 17th Clock) to Command Register or Display RAM. If Host release A0(High) and RW#(High), that means Host want to read date from Command Register. ER-TFTMC070-4 will release the content(b7 ~b0) of Command Register on 10th ~ 17th Clock. If Host release A0(Low) and RW#(High), that means Host want to read Status Data. ER-TFTMC070-4 will release the Status Data(B7~ B0) on 10th ~ 17th Clock. Host will get the data of ER-TFTMC070-4's Status Register.



### 5.2.3 Display Input Data Format

ER-TFTMC070-4 supports Monochrome, 256 color, 65K color and 16.7M color for TFT panel. The data for RGB colors are arranged in Display RAM as follows:

- 1bpp : Monochrome (1bit/pixel)
- 8bpp : Color RGB 3:3:2 (1 byte/pixel)
- 16bpp : Color RGB 5:6:5 (2bytes/pixel)
- 24bpp : Color RGB 8:8:8 (3bytes/pixel, or 4bytes/pixel)

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

#### 5.2.3.1 Input Data without Opacity (RGB)

Table 2-6: 8bits MCU, 1bpp Monochrom Mode

| Order | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|-------|------|------|------|------|------|------|------|------|
| 1     | P7   | P6   | P5   | P4   | P3   | P2   | P1   | P0   |
| 2     | P15  | P14  | P13  | P12  | P11  | P10  | P9   | P8   |
| 3     | P23  | P22  | P21  | P20  | P19  | P18  | P17  | P16  |
| 4     | P31  | P30  | P29  | P28  | P27  | P26  | P25  | P24  |
| 5     | P39  | P38  | P37  | P36  | P35  | P34  | P33  | P32  |
| 6     | P47  | P46  | P45  | P44  | P43  | P42  | P41  | P40  |

Table 2-7: 8bits MCU, 8bpp Mode (RGB 3:3:2)

| Order | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 1     | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> |
| 2     | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> |
| 3     | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> |
| 4     | R <sub>3</sub> <sup>7</sup> | R <sub>3</sub> <sup>6</sup> | R <sub>3</sub> <sup>5</sup> | G <sub>3</sub> <sup>7</sup> | G <sub>3</sub> <sup>6</sup> | G <sub>3</sub> <sup>5</sup> | B <sub>3</sub> <sup>7</sup> | B <sub>3</sub> <sup>6</sup> |
| 5     | R <sub>4</sub> <sup>7</sup> | R <sub>4</sub> <sup>6</sup> | R <sub>4</sub> <sup>5</sup> | G <sub>4</sub> <sup>7</sup> | G <sub>4</sub> <sup>6</sup> | G <sub>4</sub> <sup>5</sup> | B <sub>4</sub> <sup>7</sup> | B <sub>4</sub> <sup>6</sup> |
| 6     | R <sub>5</sub> <sup>7</sup> | R <sub>5</sub> <sup>6</sup> | R <sub>5</sub> <sup>5</sup> | G <sub>5</sub> <sup>7</sup> | G <sub>5</sub> <sup>6</sup> | G <sub>5</sub> <sup>5</sup> | B <sub>5</sub> <sup>7</sup> | B <sub>5</sub> <sup>6</sup> |

Table 2-8: 8bits MCU, 16bpp Mode (RGB 5:6:5)

| Order | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 1     | G <sub>0</sub> <sup>4</sup> | G <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>2</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> | B <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>4</sup> | B <sub>0</sub> <sup>3</sup> |
| 2     | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | R <sub>0</sub> <sup>4</sup> | R <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> |
| 3     | G <sub>1</sub> <sup>4</sup> | G <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>2</sup> | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> | B <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>4</sup> | B <sub>1</sub> <sup>3</sup> |
| 4     | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | R <sub>1</sub> <sup>4</sup> | R <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> |
| 5     | G <sub>2</sub> <sup>4</sup> | G <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>2</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> | B <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>4</sup> | B <sub>2</sub> <sup>3</sup> |
| 6     | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | R <sub>2</sub> <sup>4</sup> | R <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> |

Table 2-9: 8bits MCU, 24bpp Mode (RGB 8:8:8)

| Order | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 1     | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> | B <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>4</sup> | B <sub>0</sub> <sup>3</sup> | B <sub>0</sub> <sup>2</sup> | B <sub>0</sub> <sup>1</sup> | B <sub>0</sub> <sup>0</sup> |
| 2     | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>4</sup> | G <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>2</sup> | G <sub>0</sub> <sup>1</sup> | G <sub>0</sub> <sup>0</sup> |
| 3     | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | R <sub>0</sub> <sup>4</sup> | R <sub>0</sub> <sup>3</sup> | R <sub>0</sub> <sup>2</sup> | R <sub>0</sub> <sup>1</sup> | R <sub>0</sub> <sup>0</sup> |
| 4     | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> | B <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>4</sup> | B <sub>1</sub> <sup>3</sup> | B <sub>1</sub> <sup>2</sup> | B <sub>1</sub> <sup>1</sup> | B <sub>1</sub> <sup>0</sup> |
| 5     | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>4</sup> | G <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>2</sup> | G <sub>1</sub> <sup>1</sup> | G <sub>1</sub> <sup>0</sup> |
| 6     | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | R <sub>1</sub> <sup>4</sup> | R <sub>1</sub> <sup>3</sup> | R <sub>1</sub> <sup>2</sup> | R <sub>1</sub> <sup>1</sup> | R <sub>1</sub> <sup>0</sup> |

Table 2-10: 16bits MCU, 1bpp Mono Mode -1

| 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  | P7   | P6   | P5   | P4   | P3   | P2   | P1   | P0   |
| 2     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | P15  | P14  | P13  | P12  | P11  | P10  | P9   | P8   |
| 3     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | P23  | P22  | P21  | P20  | P19  | P18  | P17  | P16  |
| 4     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | P31  | P30  | P29  | P28  | P27  | P26  | P25  | P24  |
| 5     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | P39  | P38  | P37  | P36  | P35  | P34  | P33  | P32  |
| 6     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | P47  | P46  | P45  | P44  | P43  | P42  | P41  | P40  |

Table 2-11: 16bits MCU, 1bpp Mono Mode -2

| Order | Bit15 | Bit14 | Bit13 | Bit12 | Bit11 | Bit10 | Bit9 | Bit8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|-------|-------|-------|-------|-------|-------|-------|------|------|------|------|------|------|------|------|------|------|
| 1     | P15   | P14   | P13   | P12   | P11   | P10   | P9   | P8   | P7   | P6   | P5   | P4   | P3   | P2   | P1   | P0   |
| 2     | P31   | P30   | P29   | P28   | P27   | P26   | P25  | P24  | P23  | P22  | P21  | P20  | P19  | P18  | P17  | P16  |
| 3     | P47   | P46   | P45   | P44   | P43   | P42   | P41  | P40  | P39  | P38  | P37  | P36  | P35  | P34  | P33  | P32  |
| 4     | P63   | P62   | P61   | P60   | P59   | P58   | P57  | P56  | P55  | P54  | P53  | P52  | P51  | P50  | P49  | P48  |
| 5     | P79   | P78   | P77   | P76   | P75   | P74   | P73  | P72  | P71  | P70  | P69  | P68  | P67  | P66  | P65  | P64  |
| 6     | P95   | P94   | P93   | P92   | P91   | P90   | P89  | P88  | P87  | P86  | P85  | P84  | P83  | P82  | P81  | P80  |

Table 2-12: 16bits MCU, 8bpp Mode -1 (RGB 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 <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> |
| 2     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> |
| 3     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> |
| 4     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | R <sub>3</sub> <sup>7</sup> | R <sub>3</sub> <sup>6</sup> | R <sub>3</sub> <sup>5</sup> | G <sub>3</sub> <sup>7</sup> | G <sub>3</sub> <sup>6</sup> | G <sub>3</sub> <sup>5</sup> | B <sub>3</sub> <sup>7</sup> | B <sub>3</sub> <sup>6</sup> |
| 5     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | R <sub>4</sub> <sup>7</sup> | R <sub>4</sub> <sup>6</sup> | R <sub>4</sub> <sup>5</sup> | G <sub>4</sub> <sup>7</sup> | G <sub>4</sub> <sup>6</sup> | G <sub>4</sub> <sup>5</sup> | B <sub>4</sub> <sup>7</sup> | B <sub>4</sub> <sup>6</sup> |
| 6     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | R <sub>5</sub> <sup>7</sup> | R <sub>5</sub> <sup>6</sup> | R <sub>5</sub> <sup>5</sup> | G <sub>5</sub> <sup>7</sup> | G <sub>5</sub> <sup>6</sup> | G <sub>5</sub> <sup>5</sup> | B <sub>5</sub> <sup>7</sup> | B <sub>5</sub> <sup>6</sup> |

Table 2-13: 16bits MCU, 8bpp Mode -2 (RGB 3:3:2)

| Order | Bit15                        | Bit14                        | Bit13                        | Bit12                        | Bit11                        | Bit10                        | Bit9                         | Bit8                         | Bit7                         | Bit6                         | Bit5                         | Bit4                         | Bit3                         | Bit2                         | Bit1                         | Bit0                         |
|-------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|
| 1     | R <sub>1</sub> <sup>7</sup>  | R <sub>1</sub> <sup>6</sup>  | R <sub>1</sub> <sup>5</sup>  | G <sub>1</sub> <sup>7</sup>  | G <sub>1</sub> <sup>6</sup>  | G <sub>1</sub> <sup>5</sup>  | B <sub>1</sub> <sup>7</sup>  | B <sub>1</sub> <sup>6</sup>  | R <sub>0</sub> <sup>7</sup>  | R <sub>0</sub> <sup>6</sup>  | R <sub>0</sub> <sup>5</sup>  | G <sub>0</sub> <sup>7</sup>  | G <sub>0</sub> <sup>6</sup>  | G <sub>0</sub> <sup>5</sup>  | B <sub>0</sub> <sup>7</sup>  | B <sub>0</sub> <sup>6</sup>  |
| 2     | R <sub>3</sub> <sup>7</sup>  | R <sub>3</sub> <sup>6</sup>  | R <sub>3</sub> <sup>5</sup>  | G <sub>3</sub> <sup>7</sup>  | G <sub>3</sub> <sup>6</sup>  | G <sub>3</sub> <sup>5</sup>  | B <sub>3</sub> <sup>7</sup>  | B <sub>3</sub> <sup>6</sup>  | R <sub>2</sub> <sup>7</sup>  | R <sub>2</sub> <sup>6</sup>  | R <sub>2</sub> <sup>5</sup>  | G <sub>2</sub> <sup>7</sup>  | G <sub>2</sub> <sup>6</sup>  | G <sub>2</sub> <sup>5</sup>  | B <sub>2</sub> <sup>7</sup>  | B <sub>2</sub> <sup>6</sup>  |
| 3     | R <sub>5</sub> <sup>7</sup>  | R <sub>5</sub> <sup>6</sup>  | R <sub>5</sub> <sup>5</sup>  | G <sub>5</sub> <sup>7</sup>  | G <sub>5</sub> <sup>6</sup>  | G <sub>5</sub> <sup>5</sup>  | B <sub>5</sub> <sup>7</sup>  | B <sub>5</sub> <sup>6</sup>  | R <sub>4</sub> <sup>7</sup>  | R <sub>4</sub> <sup>6</sup>  | R <sub>4</sub> <sup>5</sup>  | G <sub>4</sub> <sup>7</sup>  | G <sub>4</sub> <sup>6</sup>  | G <sub>4</sub> <sup>5</sup>  | B <sub>4</sub> <sup>7</sup>  | B <sub>4</sub> <sup>6</sup>  |
| 4     | R <sub>7</sub> <sup>7</sup>  | R <sub>7</sub> <sup>6</sup>  | R <sub>7</sub> <sup>5</sup>  | G <sub>7</sub> <sup>7</sup>  | G <sub>7</sub> <sup>6</sup>  | G <sub>7</sub> <sup>5</sup>  | B <sub>7</sub> <sup>7</sup>  | B <sub>7</sub> <sup>6</sup>  | R <sub>6</sub> <sup>7</sup>  | R <sub>6</sub> <sup>6</sup>  | R <sub>6</sub> <sup>5</sup>  | G <sub>6</sub> <sup>7</sup>  | G <sub>6</sub> <sup>6</sup>  | G <sub>6</sub> <sup>5</sup>  | B <sub>6</sub> <sup>7</sup>  | B <sub>6</sub> <sup>6</sup>  |
| 5     | R <sub>9</sub> <sup>7</sup>  | R <sub>9</sub> <sup>6</sup>  | R <sub>9</sub> <sup>5</sup>  | G <sub>9</sub> <sup>7</sup>  | G <sub>9</sub> <sup>6</sup>  | G <sub>9</sub> <sup>5</sup>  | B <sub>9</sub> <sup>7</sup>  | B <sub>9</sub> <sup>6</sup>  | R <sub>8</sub> <sup>7</sup>  | R <sub>8</sub> <sup>6</sup>  | R <sub>8</sub> <sup>5</sup>  | G <sub>8</sub> <sup>7</sup>  | G <sub>8</sub> <sup>6</sup>  | G <sub>8</sub> <sup>5</sup>  | B <sub>8</sub> <sup>7</sup>  | B <sub>8</sub> <sup>6</sup>  |
| 6     | R <sub>11</sub> <sup>7</sup> | R <sub>11</sub> <sup>6</sup> | R <sub>11</sub> <sup>5</sup> | G <sub>11</sub> <sup>7</sup> | G <sub>11</sub> <sup>6</sup> | G <sub>11</sub> <sup>5</sup> | B <sub>11</sub> <sup>7</sup> | B <sub>11</sub> <sup>6</sup> | R <sub>10</sub> <sup>7</sup> | R <sub>10</sub> <sup>6</sup> | R <sub>10</sub> <sup>5</sup> | G <sub>10</sub> <sup>7</sup> | G <sub>10</sub> <sup>6</sup> | G <sub>10</sub> <sup>5</sup> | B <sub>10</sub> <sup>7</sup> | B <sub>10</sub> <sup>6</sup> |

Note: The Mode-1 and Mode 2 is determined by bit[7:6] of Register[02h], please refer to the Cptater-13 Reregister Description.

Table 2-14: 16bits MCU, 16bpp Mode (RGB 5:6:5)

| Order | Bit15                       | Bit14                       | Bit13                       | Bit12                       | Bit11                       | Bit10                       | Bit9                        | Bit8                        | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 1     | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | R <sub>0</sub> <sup>4</sup> | R <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>4</sup> | G <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>2</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> | B <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>4</sup> | B <sub>0</sub> <sup>3</sup> |
| 2     | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | R <sub>1</sub> <sup>4</sup> | R <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>4</sup> | G <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>2</sup> | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> | B <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>4</sup> | B <sub>1</sub> <sup>3</sup> |
| 3     | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | R <sub>2</sub> <sup>4</sup> | R <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> | G <sub>2</sub> <sup>4</sup> | G <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>2</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> | B <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>4</sup> | B <sub>2</sub> <sup>3</sup> |
| 4     | R <sub>3</sub> <sup>7</sup> | R <sub>3</sub> <sup>6</sup> | R <sub>3</sub> <sup>5</sup> | R <sub>3</sub> <sup>4</sup> | R <sub>3</sub> <sup>3</sup> | G <sub>3</sub> <sup>7</sup> | G <sub>3</sub> <sup>6</sup> | G <sub>3</sub> <sup>5</sup> | G <sub>3</sub> <sup>4</sup> | G <sub>3</sub> <sup>3</sup> | G <sub>3</sub> <sup>2</sup> | B <sub>3</sub> <sup>7</sup> | B <sub>3</sub> <sup>6</sup> | B <sub>3</sub> <sup>5</sup> | B <sub>3</sub> <sup>4</sup> | B <sub>3</sub> <sup>3</sup> |
| 5     | R <sub>4</sub> <sup>7</sup> | R <sub>4</sub> <sup>6</sup> | R <sub>4</sub> <sup>5</sup> | R <sub>4</sub> <sup>4</sup> | R <sub>4</sub> <sup>3</sup> | G <sub>4</sub> <sup>7</sup> | G <sub>4</sub> <sup>6</sup> | G <sub>4</sub> <sup>5</sup> | G <sub>4</sub> <sup>4</sup> | G <sub>4</sub> <sup>3</sup> | G <sub>4</sub> <sup>2</sup> | B <sub>4</sub> <sup>7</sup> | B <sub>4</sub> <sup>6</sup> | B <sub>4</sub> <sup>5</sup> | B <sub>4</sub> <sup>4</sup> | B <sub>4</sub> <sup>3</sup> |
| 6     | R <sub>5</sub> <sup>7</sup> | R <sub>5</sub> <sup>6</sup> | R <sub>5</sub> <sup>5</sup> | R <sub>5</sub> <sup>4</sup> | R <sub>5</sub> <sup>3</sup> | G <sub>5</sub> <sup>7</sup> | G <sub>5</sub> <sup>6</sup> | G <sub>5</sub> <sup>5</sup> | G <sub>5</sub> <sup>4</sup> | G <sub>5</sub> <sup>3</sup> | G <sub>5</sub> <sup>2</sup> | B <sub>5</sub> <sup>7</sup> | B <sub>5</sub> <sup>6</sup> | B <sub>5</sub> <sup>5</sup> | B <sub>5</sub> <sup>4</sup> | B <sub>5</sub> <sup>3</sup> |

Table 2-15: 16bits MCU, 24bpp Mode -1 (RGB 8:8:8)

| Order | Bit15                       | Bit14                       | Bit13                       | Bit12                       | Bit11                       | Bit10                       | Bit9                        | Bit8                        | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 1     | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>4</sup> | G <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>2</sup> | G <sub>0</sub> <sup>1</sup> | G <sub>0</sub> <sup>0</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> | B <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>4</sup> | B <sub>0</sub> <sup>3</sup> | B <sub>0</sub> <sup>2</sup> | B <sub>0</sub> <sup>1</sup> | B <sub>0</sub> <sup>0</sup> |
| 2     | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> | B <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>4</sup> | B <sub>1</sub> <sup>3</sup> | B <sub>1</sub> <sup>2</sup> | B <sub>1</sub> <sup>1</sup> | B <sub>1</sub> <sup>0</sup> | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | R <sub>0</sub> <sup>4</sup> | R <sub>0</sub> <sup>3</sup> | R <sub>0</sub> <sup>2</sup> | R <sub>0</sub> <sup>1</sup> | R <sub>0</sub> <sup>0</sup> |
| 3     | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | R <sub>1</sub> <sup>4</sup> | R <sub>1</sub> <sup>3</sup> | R <sub>1</sub> <sup>2</sup> | R <sub>1</sub> <sup>1</sup> | R <sub>1</sub> <sup>0</sup> | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>4</sup> | G <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>2</sup> | G <sub>1</sub> <sup>1</sup> | G <sub>1</sub> <sup>0</sup> |
| 4     | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> | G <sub>2</sub> <sup>4</sup> | G <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>2</sup> | G <sub>2</sub> <sup>1</sup> | G <sub>2</sub> <sup>0</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> | B <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>4</sup> | B <sub>2</sub> <sup>3</sup> | B <sub>2</sub> <sup>2</sup> | B <sub>2</sub> <sup>1</sup> | B <sub>2</sub> <sup>0</sup> |
| 5     | B <sub>3</sub> <sup>7</sup> | B <sub>3</sub> <sup>6</sup> | B <sub>3</sub> <sup>5</sup> | B <sub>3</sub> <sup>4</sup> | B <sub>3</sub> <sup>3</sup> | B <sub>3</sub> <sup>2</sup> | B <sub>3</sub> <sup>1</sup> | B <sub>3</sub> <sup>0</sup> | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | R <sub>2</sub> <sup>4</sup> | R <sub>2</sub> <sup>3</sup> | R <sub>2</sub> <sup>2</sup> | R <sub>2</sub> <sup>1</sup> | R <sub>2</sub> <sup>0</sup> |
| 6     | R <sub>3</sub> <sup>7</sup> | R <sub>3</sub> <sup>6</sup> | R <sub>3</sub> <sup>5</sup> | R <sub>3</sub> <sup>4</sup> | R <sub>3</sub> <sup>3</sup> | R <sub>3</sub> <sup>2</sup> | R <sub>3</sub> <sup>1</sup> | R <sub>3</sub> <sup>0</sup> | G <sub>3</sub> <sup>7</sup> | G <sub>3</sub> <sup>6</sup> | G <sub>3</sub> <sup>5</sup> | G <sub>3</sub> <sup>4</sup> | G <sub>3</sub> <sup>3</sup> | G <sub>3</sub> <sup>2</sup> | G <sub>3</sub> <sup>1</sup> | G <sub>3</sub> <sup>0</sup> |

Table 2-16: 16bits MCU, 24bpp Mode -2 (RGB 8:8:8)

| Order | Bit15                       | Bit14                       | Bit13                       | Bit12                       | Bit11                       | Bit10                       | Bit9                        | Bit8                        | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 1     | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>4</sup> | G <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>2</sup> | G <sub>0</sub> <sup>1</sup> | G <sub>0</sub> <sup>0</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> | B <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>4</sup> | B <sub>0</sub> <sup>3</sup> | B <sub>0</sub> <sup>2</sup> | B <sub>0</sub> <sup>1</sup> | B <sub>0</sub> <sup>0</sup> |
| 2     | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | R <sub>0</sub> <sup>4</sup> | R <sub>0</sub> <sup>3</sup> | R <sub>0</sub> <sup>2</sup> | R <sub>0</sub> <sup>1</sup> | R <sub>0</sub> <sup>0</sup> |
| 3     | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>4</sup> | G <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>2</sup> | G <sub>1</sub> <sup>1</sup> | G <sub>1</sub> <sup>0</sup> | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> | B <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>4</sup> | B <sub>1</sub> <sup>3</sup> | B <sub>1</sub> <sup>2</sup> | B <sub>1</sub> <sup>1</sup> | B <sub>1</sub> <sup>0</sup> |
| 4     | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | R <sub>1</sub> <sup>4</sup> | R <sub>1</sub> <sup>3</sup> | R <sub>1</sub> <sup>2</sup> | R <sub>1</sub> <sup>1</sup> | R <sub>1</sub> <sup>0</sup> |
| 5     | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> | G <sub>2</sub> <sup>4</sup> | G <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>2</sup> | G <sub>2</sub> <sup>1</sup> | G <sub>2</sub> <sup>0</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> | B <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>4</sup> | B <sub>2</sub> <sup>3</sup> | B <sub>2</sub> <sup>2</sup> | B <sub>2</sub> <sup>1</sup> | B <sub>2</sub> <sup>0</sup> |
| 6     | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | n/a                         | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | R <sub>2</sub> <sup>4</sup> | R <sub>2</sub> <sup>3</sup> | R <sub>2</sub> <sup>2</sup> | R <sub>2</sub> <sup>1</sup> | R <sub>2</sub> <sup>0</sup> |

### 5.2.3.2 Input Data with Opacity (aRGB)

ER-TFTMC070-4 provide a palette of 64 simultaneous colors from a total of 4096 different colors with opacity attribute for OSD (On-Screen-Display) application. User may load preferred color into embedded color palette then pick it up by index color. The □ value stands for opacity.

Table 2-17: 8bits MCU, 8bpp Mode (Index 2:6)

| Order | Bit7                         | Bit6                         | Bit5                   | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|-------|------------------------------|------------------------------|------------------------|------|------|------|------|------|
| 1     | □ <sub>1</sub> <sup>3</sup>  | □ <sub>1</sub> <sup>2</sup>  | Index color of pixel 0 |      |      |      |      |      |
| 2     | □ <sub>3</sub> <sup>3</sup>  | □ <sub>3</sub> <sup>2</sup>  | Index color of pixel 1 |      |      |      |      |      |
| 3     | □ <sub>5</sub> <sup>3</sup>  | □ <sub>5</sub> <sup>2</sup>  | Index color of pixel 2 |      |      |      |      |      |
| 4     | □ <sub>7</sub> <sup>3</sup>  | □ <sub>7</sub> <sup>2</sup>  | Index color of pixel 3 |      |      |      |      |      |
| 5     | □ <sub>9</sub> <sup>3</sup>  | □ <sub>9</sub> <sup>2</sup>  | Index color of pixel 4 |      |      |      |      |      |
| 6     | □ <sub>11</sub> <sup>3</sup> | □ <sub>11</sub> <sup>2</sup> | Index color of pixel 5 |      |      |      |      |      |

Table 2-18: 8bits MCU, 16bpp Mode (RGB 4:4:4)

| Order | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 1     | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>4</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> | B <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>4</sup> |
| 2     | □ <sub>0</sub> <sup>3</sup> | □ <sub>0</sub> <sup>2</sup> | □ <sub>0</sub> <sup>1</sup> | □ <sub>0</sub> <sup>0</sup> | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | R <sub>0</sub> <sup>4</sup> |
| 3     | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>4</sup> | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> | B <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>4</sup> |
| 4     | □ <sub>1</sub> <sup>3</sup> | □ <sub>1</sub> <sup>2</sup> | □ <sub>1</sub> <sup>1</sup> | □ <sub>1</sub> <sup>0</sup> | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | R <sub>1</sub> <sup>4</sup> |
| 5     | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> | G <sub>2</sub> <sup>4</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> | B <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>4</sup> |
| 6     | □ <sub>2</sub> <sup>3</sup> | □ <sub>2</sub> <sup>2</sup> | □ <sub>2</sub> <sup>1</sup> | □ <sub>2</sub> <sup>0</sup> | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | R <sub>2</sub> <sup>4</sup> |

Table 2-19: 16bits MCU, Index Mode (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  | □ <sub>0</sub> <sup>3</sup> | □ <sub>0</sub> <sup>2</sup> | Index color of pixel 0 |      |      |      |      |      |
| 2     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | □ <sub>1</sub> <sup>3</sup> | □ <sub>1</sub> <sup>2</sup> | Index color of pixel 1 |      |      |      |      |      |
| 3     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | □ <sub>2</sub> <sup>3</sup> | □ <sub>2</sub> <sup>2</sup> | Index color of pixel 2 |      |      |      |      |      |
| 4     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | □ <sub>3</sub> <sup>3</sup> | □ <sub>3</sub> <sup>2</sup> | Index color of pixel 3 |      |      |      |      |      |
| 5     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | □ <sub>4</sub> <sup>3</sup> | □ <sub>4</sub> <sup>2</sup> | Index color of pixel 4 |      |      |      |      |      |
| 6     | n/a   | n/a   | n/a   | n/a   | n/a   | n/a   | n/a  | n/a  | □ <sub>5</sub> <sup>3</sup> | □ <sub>5</sub> <sup>2</sup> | Index color of pixel 5 |      |      |      |      |      |

Table 2-20: 16bits MCU, 12bpp Mode (RGB 4:4:4:4)

| Order | Bit15           | Bit14           | Bit13           | Bit12           | Bit11   | Bit10   | Bit9    | Bit8    | Bit7    | Bit6    | Bit5    | Bit4    | Bit3    | Bit2    | Bit1    | Bit0    |
|-------|-----------------|-----------------|-----------------|-----------------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| 1     | $\square_{0^3}$ | $\square_{0^2}$ | $\square_{0^1}$ | $\square_{0^0}$ | $R_0^7$ | $R_0^6$ | $R_0^5$ | $R_0^4$ | $G_0^7$ | $G_0^6$ | $G_0^5$ | $G_0^4$ | $B_0^7$ | $B_0^6$ | $B_0^5$ | $B_0^4$ |
| 2     | $\square_{1^3}$ | $\square_{1^2}$ | $\square_{1^1}$ | $\square_{1^0}$ | $R_1^7$ | $R_1^6$ | $R_1^5$ | $R_1^4$ | $G_1^7$ | $G_1^6$ | $G_1^5$ | $G_1^4$ | $B_1^7$ | $B_1^6$ | $B_1^5$ | $B_1^4$ |
| 3     | $\square_{2^3}$ | $\square_{2^2}$ | $\square_{2^1}$ | $\square_{2^0}$ | $R_2^7$ | $R_2^6$ | $R_2^5$ | $R_2^4$ | $G_2^7$ | $G_2^6$ | $G_2^5$ | $G_2^4$ | $B_2^7$ | $B_2^6$ | $B_2^5$ | $B_2^4$ |
| 4     | $\square_{3^3}$ | $\square_{3^2}$ | $\square_{3^1}$ | $\square_{3^0}$ | $R_3^7$ | $R_3^6$ | $R_3^5$ | $R_3^4$ | $G_3^7$ | $G_3^6$ | $G_3^5$ | $G_3^4$ | $B_3^7$ | $B_3^6$ | $B_3^5$ | $B_3^4$ |
| 5     | $\square_{4^3}$ | $\square_{4^2}$ | $\square_{4^1}$ | $\square_{4^0}$ | $R_4^7$ | $R_4^6$ | $R_4^5$ | $R_4^4$ | $G_4^7$ | $G_4^6$ | $G_4^5$ | $G_4^4$ | $B_4^7$ | $B_4^6$ | $B_4^5$ | $B_4^4$ |
| 6     | $\square_{5^3}$ | $\square_{5^2}$ | $\square_{5^1}$ | $\square_{5^0}$ | $R_5^7$ | $R_5^6$ | $R_5^5$ | $R_5^4$ | $G_5^7$ | $G_5^6$ | $G_5^5$ | $G_5^4$ | $B_5^7$ | $B_5^6$ | $B_5^5$ | $B_5^4$ |

## 5.3 Display Memory

ER-TFTMC070-4 embedded a 32Mb Display RAM. The Host(MCU) through the instructions to save the displayed data to internal Display RAM. And ER-TFTMC070-4's internal display engine will continue to read the data of memory, then sent to TFT driver. The capacity of the Display RAM is also related to the resolutions and image layers supported. As shown in the following table:

Table 3-1: Embedded Display RAM Capacity

| Display RAM Capacity | Resolution (Max.) | Color (Max.) | Image Layer (@Max Color) |
|----------------------|-------------------|--------------|--------------------------|
| 32Mb                 | 320x240           | 16.7M        | 18                       |
|                      | 480x272           | 16.7M        | 10                       |
|                      | 640x480           | 16.7M        | 4                        |
|                      | 800x600           | 16.7M        | 2                        |
|                      | 1024x768          | 16.7M        | 1                        |

The Display RAM type of ER-TFTMC070-4 embedded is a kind of High-Speed SDRAM (Synchronous Dynamic Random Access Memory). Before the Host access the Display RAM, it must be based on the type of Display RAM to set the relevant register initialization as following steps:

- According to the Display RAM capacity, setup Register REG[E0h]. The value of REG[E0h] must be set according to the ER-TFTMC070-4 to avoid display anomalies and image confusion. Please refer to Table 14-5 of Chapter 14.
- According to the type of Display RAM, setup Register REG[E1H], REG[E2h], REG[E3h], including setup the CAS latency, refresh interval, etc..... The typical Display RAM refresh interval is 64ms. The value of these registers are recommended according to the ER-TFTMC070-4 used. Please refer to Table 14-6 of Chapter 14.
- Set Register REG[E4h] Bit0 is 1, start Display RAM initialization processing.
- Read Register REG[E4h] bit0, if it becomes 1, that means initialization is complete.

### 5.3.1 Display RAM Data Structure

The image data stored in the Display RAM will be stored in different arrangement formats depending on the color (1bpp, 8bpp, 16bpp, 24bpp). Therefore, the Host writes the image data must according to these formats, the following tables are 8/16/24bpp RGB data arrangement format:

#### 5.3.1.1 8bpp Display Data (RGB 3:3:2)

Table 3-2: 8bpp Display Data (RGB 3:3:2)

| Addr  | Bit1<br>5                    | Bit1<br>4                    | Bit13                        | Bit12                        | Bit11                        | Bit10                        | Bit9                         | Bit8                         | Bit7                         | Bit6                         | Bit5                         | Bit4                         | Bit3                         | Bit2                         | Bit1                         | Bit0                         |
|-------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|
| 0000h | R <sub>1</sub> <sup>7</sup>  | R <sub>1</sub> <sup>6</sup>  | R <sub>1</sub> <sup>5</sup>  | G <sub>1</sub> <sup>7</sup>  | G <sub>1</sub> <sup>6</sup>  | G <sub>1</sub> <sup>5</sup>  | B <sub>1</sub> <sup>7</sup>  | B <sub>1</sub> <sup>6</sup>  | R <sub>0</sub> <sup>7</sup>  | R <sub>0</sub> <sup>6</sup>  | R <sub>0</sub> <sup>5</sup>  | G <sub>0</sub> <sup>7</sup>  | G <sub>0</sub> <sup>6</sup>  | G <sub>0</sub> <sup>5</sup>  | B <sub>0</sub> <sup>7</sup>  | B <sub>0</sub> <sup>6</sup>  |
| 0002h | R <sub>3</sub> <sup>7</sup>  | R <sub>3</sub> <sup>6</sup>  | R <sub>3</sub> <sup>5</sup>  | G <sub>3</sub> <sup>7</sup>  | G <sub>3</sub> <sup>6</sup>  | G <sub>3</sub> <sup>5</sup>  | B <sub>3</sub> <sup>7</sup>  | B <sub>3</sub> <sup>6</sup>  | R <sub>2</sub> <sup>7</sup>  | R <sub>2</sub> <sup>6</sup>  | R <sub>2</sub> <sup>5</sup>  | G <sub>2</sub> <sup>7</sup>  | G <sub>2</sub> <sup>6</sup>  | G <sub>2</sub> <sup>5</sup>  | B <sub>2</sub> <sup>7</sup>  | B <sub>2</sub> <sup>6</sup>  |
| 0004h | R <sub>5</sub> <sup>7</sup>  | R <sub>5</sub> <sup>6</sup>  | R <sub>5</sub> <sup>5</sup>  | G <sub>5</sub> <sup>7</sup>  | G <sub>5</sub> <sup>6</sup>  | G <sub>5</sub> <sup>5</sup>  | B <sub>5</sub> <sup>7</sup>  | B <sub>5</sub> <sup>6</sup>  | R <sub>4</sub> <sup>7</sup>  | R <sub>4</sub> <sup>6</sup>  | R <sub>4</sub> <sup>5</sup>  | G <sub>4</sub> <sup>7</sup>  | G <sub>4</sub> <sup>6</sup>  | G <sub>4</sub> <sup>5</sup>  | B <sub>4</sub> <sup>7</sup>  | B <sub>4</sub> <sup>6</sup>  |
| 0006h | R <sub>7</sub> <sup>7</sup>  | R <sub>7</sub> <sup>6</sup>  | R <sub>7</sub> <sup>5</sup>  | G <sub>7</sub> <sup>7</sup>  | G <sub>7</sub> <sup>6</sup>  | G <sub>7</sub> <sup>5</sup>  | B <sub>7</sub> <sup>7</sup>  | B <sub>7</sub> <sup>6</sup>  | R <sub>6</sub> <sup>7</sup>  | R <sub>6</sub> <sup>6</sup>  | R <sub>6</sub> <sup>5</sup>  | G <sub>6</sub> <sup>7</sup>  | G <sub>6</sub> <sup>6</sup>  | G <sub>6</sub> <sup>5</sup>  | B <sub>6</sub> <sup>7</sup>  | B <sub>6</sub> <sup>6</sup>  |
| 0008h | R <sub>9</sub> <sup>7</sup>  | R <sub>9</sub> <sup>6</sup>  | R <sub>9</sub> <sup>5</sup>  | G <sub>9</sub> <sup>7</sup>  | G <sub>9</sub> <sup>6</sup>  | G <sub>9</sub> <sup>5</sup>  | B <sub>9</sub> <sup>7</sup>  | B <sub>9</sub> <sup>6</sup>  | R <sub>8</sub> <sup>7</sup>  | R <sub>8</sub> <sup>6</sup>  | R <sub>8</sub> <sup>5</sup>  | G <sub>8</sub> <sup>7</sup>  | G <sub>8</sub> <sup>6</sup>  | G <sub>8</sub> <sup>5</sup>  | B <sub>8</sub> <sup>7</sup>  | B <sub>8</sub> <sup>6</sup>  |
| 000Ah | R <sub>11</sub> <sup>7</sup> | R <sub>11</sub> <sup>6</sup> | R <sub>11</sub> <sup>5</sup> | G <sub>11</sub> <sup>7</sup> | G <sub>11</sub> <sup>6</sup> | G <sub>11</sub> <sup>5</sup> | B <sub>11</sub> <sup>7</sup> | B <sub>11</sub> <sup>6</sup> | R <sub>10</sub> <sup>7</sup> | R <sub>10</sub> <sup>6</sup> | R <sub>10</sub> <sup>5</sup> | G <sub>10</sub> <sup>7</sup> | G <sub>10</sub> <sup>6</sup> | G <sub>10</sub> <sup>5</sup> | B <sub>10</sub> <sup>7</sup> | B <sub>10</sub> <sup>6</sup> |

#### 5.3.1.2 16bpp Display Data (RGB 5:6:5)

Table 3-3: 16bpp Display Data (RGB 5:6:5)

| Addr  | Bit1<br>5                   | Bit1<br>4                   | Bit13                       | Bit12                       | Bit11                       | Bit1<br>0                   | Bit9                        | Bit8                        | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 0000h | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | R <sub>0</sub> <sup>4</sup> | R <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>4</sup> | G <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>2</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> | B <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>4</sup> | B <sub>0</sub> <sup>3</sup> |
| 0002h | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | R <sub>1</sub> <sup>4</sup> | R <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>4</sup> | G <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>2</sup> | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> | B <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>4</sup> | B <sub>1</sub> <sup>3</sup> |
| 0004h | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | R <sub>2</sub> <sup>4</sup> | R <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> | G <sub>2</sub> <sup>4</sup> | G <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>2</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> | B <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>4</sup> | B <sub>2</sub> <sup>3</sup> |
| 0006h | R <sub>3</sub> <sup>7</sup> | R <sub>3</sub> <sup>6</sup> | R <sub>3</sub> <sup>5</sup> | R <sub>3</sub> <sup>4</sup> | R <sub>3</sub> <sup>3</sup> | G <sub>3</sub> <sup>7</sup> | G <sub>3</sub> <sup>6</sup> | G <sub>3</sub> <sup>5</sup> | G <sub>3</sub> <sup>4</sup> | G <sub>3</sub> <sup>3</sup> | G <sub>3</sub> <sup>2</sup> | B <sub>3</sub> <sup>7</sup> | B <sub>3</sub> <sup>6</sup> | B <sub>3</sub> <sup>5</sup> | B <sub>3</sub> <sup>4</sup> | B <sub>3</sub> <sup>3</sup> |
| 0008h | R <sub>4</sub> <sup>7</sup> | R <sub>4</sub> <sup>6</sup> | R <sub>4</sub> <sup>5</sup> | R <sub>4</sub> <sup>4</sup> | R <sub>4</sub> <sup>3</sup> | G <sub>4</sub> <sup>7</sup> | G <sub>4</sub> <sup>6</sup> | G <sub>4</sub> <sup>5</sup> | G <sub>4</sub> <sup>4</sup> | G <sub>4</sub> <sup>3</sup> | G <sub>4</sub> <sup>2</sup> | B <sub>4</sub> <sup>7</sup> | B <sub>4</sub> <sup>6</sup> | B <sub>4</sub> <sup>5</sup> | B <sub>4</sub> <sup>4</sup> | B <sub>4</sub> <sup>3</sup> |
| 000Ah | R <sub>5</sub> <sup>7</sup> | R <sub>5</sub> <sup>6</sup> | R <sub>5</sub> <sup>5</sup> | R <sub>5</sub> <sup>4</sup> | R <sub>5</sub> <sup>3</sup> | G <sub>5</sub> <sup>7</sup> | G <sub>5</sub> <sup>6</sup> | G <sub>5</sub> <sup>5</sup> | G <sub>5</sub> <sup>4</sup> | G <sub>5</sub> <sup>3</sup> | G <sub>5</sub> <sup>2</sup> | B <sub>5</sub> <sup>7</sup> | B <sub>5</sub> <sup>6</sup> | B <sub>5</sub> <sup>5</sup> | B <sub>5</sub> <sup>4</sup> | B <sub>5</sub> <sup>3</sup> |

#### 5.3.1.3 24bpp Display Data (RGB 8:8:8)

Table 3-4: 24bpp Display Data (RGB 8:8:8)

| Addr  | Bit1<br>5                   | Bit1<br>4                   | Bit13                       | Bit12                       | Bit11                       | Bit1<br>0                   | Bit9                        | Bit8                        | Bit7                        | Bit6                        | Bit5                        | Bit4                        | Bit3                        | Bit2                        | Bit1                        | Bit0                        |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 0000h | G <sub>0</sub> <sup>7</sup> | G <sub>0</sub> <sup>6</sup> | G <sub>0</sub> <sup>5</sup> | G <sub>0</sub> <sup>4</sup> | G <sub>0</sub> <sup>3</sup> | G <sub>0</sub> <sup>2</sup> | G <sub>0</sub> <sup>1</sup> | G <sub>0</sub> <sup>0</sup> | B <sub>0</sub> <sup>7</sup> | B <sub>0</sub> <sup>6</sup> | B <sub>0</sub> <sup>5</sup> | B <sub>0</sub> <sup>4</sup> | B <sub>0</sub> <sup>3</sup> | B <sub>0</sub> <sup>2</sup> | B <sub>0</sub> <sup>1</sup> | B <sub>0</sub> <sup>0</sup> |
| 0002h | B <sub>1</sub> <sup>7</sup> | B <sub>1</sub> <sup>6</sup> | B <sub>1</sub> <sup>5</sup> | B <sub>1</sub> <sup>4</sup> | B <sub>1</sub> <sup>3</sup> | B <sub>1</sub> <sup>2</sup> | B <sub>1</sub> <sup>1</sup> | B <sub>1</sub> <sup>0</sup> | R <sub>0</sub> <sup>7</sup> | R <sub>0</sub> <sup>6</sup> | R <sub>0</sub> <sup>5</sup> | R <sub>0</sub> <sup>4</sup> | R <sub>0</sub> <sup>3</sup> | R <sub>0</sub> <sup>2</sup> | R <sub>0</sub> <sup>1</sup> | R <sub>0</sub> <sup>0</sup> |
| 0004h | R <sub>1</sub> <sup>7</sup> | R <sub>1</sub> <sup>6</sup> | R <sub>1</sub> <sup>5</sup> | R <sub>1</sub> <sup>4</sup> | R <sub>1</sub> <sup>3</sup> | R <sub>1</sub> <sup>2</sup> | R <sub>1</sub> <sup>1</sup> | R <sub>1</sub> <sup>0</sup> | G <sub>1</sub> <sup>7</sup> | G <sub>1</sub> <sup>6</sup> | G <sub>1</sub> <sup>5</sup> | G <sub>1</sub> <sup>4</sup> | G <sub>1</sub> <sup>3</sup> | G <sub>1</sub> <sup>2</sup> | G <sub>1</sub> <sup>1</sup> | G <sub>1</sub> <sup>0</sup> |
| 0006h | G <sub>2</sub> <sup>7</sup> | G <sub>2</sub> <sup>6</sup> | G <sub>2</sub> <sup>5</sup> | G <sub>2</sub> <sup>4</sup> | G <sub>2</sub> <sup>3</sup> | G <sub>2</sub> <sup>2</sup> | G <sub>2</sub> <sup>1</sup> | G <sub>2</sub> <sup>0</sup> | B <sub>2</sub> <sup>7</sup> | B <sub>2</sub> <sup>6</sup> | B <sub>2</sub> <sup>5</sup> | B <sub>2</sub> <sup>4</sup> | B <sub>2</sub> <sup>3</sup> | B <sub>2</sub> <sup>2</sup> | B <sub>2</sub> <sup>1</sup> | B <sub>2</sub> <sup>0</sup> |

|       |                             |                             |                             |                             |                             |                             |                             |                             |                             |                             |                             |                             |                             |                             |                             |                             |
|-------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| 0008h | B <sub>3</sub> <sup>7</sup> | B <sub>3</sub> <sup>6</sup> | B <sub>3</sub> <sup>5</sup> | B <sub>3</sub> <sup>4</sup> | B <sub>3</sub> <sup>3</sup> | B <sub>3</sub> <sup>2</sup> | B <sub>3</sub> <sup>1</sup> | B <sub>3</sub> <sup>0</sup> | R <sub>2</sub> <sup>7</sup> | R <sub>2</sub> <sup>6</sup> | R <sub>2</sub> <sup>5</sup> | R <sub>2</sub> <sup>4</sup> | R <sub>2</sub> <sup>3</sup> | R <sub>2</sub> <sup>2</sup> | R <sub>2</sub> <sup>1</sup> | R <sub>2</sub> <sup>0</sup> |
| 000Ah | R <sub>3</sub> <sup>7</sup> | R <sub>3</sub> <sup>6</sup> | R <sub>3</sub> <sup>5</sup> | R <sub>3</sub> <sup>4</sup> | R <sub>3</sub> <sup>3</sup> | R <sub>3</sub> <sup>2</sup> | R <sub>3</sub> <sup>1</sup> | R <sub>3</sub> <sup>0</sup> | G <sub>3</sub> <sup>7</sup> | G <sub>3</sub> <sup>6</sup> | G <sub>3</sub> <sup>5</sup> | G <sub>3</sub> <sup>4</sup> | G <sub>3</sub> <sup>3</sup> | G <sub>3</sub> <sup>2</sup> | G <sub>3</sub> <sup>1</sup> | G <sub>3</sub> <sup>0</sup> |



### 5.3.1.4 Index Display with Opacity (RGB 2:2:2:2)

Table 3-5: Index Display with Opacity (RGB 2:2:2:2)

| Addr  | Bit1<br>5          | Bit1<br>4          | Bit1<br>3               | Bit1<br>2 | Bit1<br>1 | Bit1<br>0 | Bit9 | Bit8 | Bit7                    | Bit6               | Bit5                    | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|-------|--------------------|--------------------|-------------------------|-----------|-----------|-----------|------|------|-------------------------|--------------------|-------------------------|------|------|------|------|------|
| 0000h | $\square_1^3$      | $\square_1^2$      | Index color of pixel 1  |           |           |           |      |      | $\square_0^3$           | $\square_0^2$      | Index color of pixel 0  |      |      |      |      |      |
| 0002h | $\square_3^3$      | $\square_3^2$      | Index color of pixel 3  |           |           |           |      |      | $\square_2^3$           | $\square_2^2$      | Index color of pixel 2  |      |      |      |      |      |
| 0004h | $\square_5^3$      | $\square_5^2$      | Index color of pixel 5  |           |           |           |      |      | $\square_4^3$           | $\square_4^2$      | Index color of pixel 4  |      |      |      |      |      |
| 0006h | $\square_7^3$      | $\square_7^2$      | Index color of pixel 7  |           |           |           |      |      | $\square_6^3$           | $\square_6^2$      | Index color of pixel 6  |      |      |      |      |      |
| 0008h | $\square_9^3$      | $\square_9^2$      | Index color of pixel 9  |           |           |           |      |      | $\square_8^3$           | $\square_8^2$      | Index color of pixel 8  |      |      |      |      |      |
| 000Ah | $\square_1^3$<br>1 | $\square_1^2$<br>1 | Index color of pixel 11 |           |           |           |      |      | $\square_3^3$<br>1<br>0 | $\square_1^2$<br>0 | Index color of pixel 10 |      |      |      |      |      |

### 5.3.1.5 12bpp Display with OpacityB

Table 3-6: 12bpp Display with OpacityB

| Addr  | Bit1<br>5     | Bit1<br>4     | Bit13         | Bit12         | Bit11   | Bit1<br>0 | Bit9    | Bit8    | Bit7    | Bit6    | Bit5    | Bit4    | Bit3    | Bit2    | Bit1    | Bit0    |
|-------|---------------|---------------|---------------|---------------|---------|-----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| 0000h | $\square_0^3$ | $\square_0^2$ | $\square_0^1$ | $\square_0^0$ | $R_0^7$ | $R_0^6$   | $R_0^5$ | $R_0^4$ | $G_0^7$ | $G_0^6$ | $G_0^5$ | $G_0^4$ | $B_0^7$ | $B_0^6$ | $B_0^5$ | $B_0^4$ |
| 0002h | $\square_1^3$ | $\square_1^2$ | $\square_1^1$ | $\square_1^0$ | $R_1^7$ | $R_1^6$   | $R_1^5$ | $R_1^4$ | $G_1^7$ | $G_1^6$ | $G_1^5$ | $G_1^4$ | $B_1^7$ | $B_1^6$ | $B_1^5$ | $B_1^4$ |
| 0004h | $\square_2^3$ | $\square_2^2$ | $\square_2^1$ | $\square_2^0$ | $R_2^7$ | $R_2^6$   | $R_2^5$ | $R_2^4$ | $G_2^7$ | $G_2^6$ | $G_2^5$ | $G_2^4$ | $B_2^7$ | $B_2^6$ | $B_2^5$ | $B_2^4$ |
| 0006h | $\square_3^3$ | $\square_3^2$ | $\square_3^1$ | $\square_3^0$ | $R_3^7$ | $R_3^6$   | $R_3^5$ | $R_3^4$ | $G_3^7$ | $G_3^6$ | $G_3^5$ | $G_3^4$ | $B_3^7$ | $B_3^6$ | $B_3^5$ | $B_3^4$ |
| 0008h | $\square_4^3$ | $\square_4^2$ | $\square_4^1$ | $\square_4^0$ | $R_4^7$ | $R_4^6$   | $R_4^5$ | $R_4^4$ | $G_4^7$ | $G_4^6$ | $G_4^5$ | $G_4^4$ | $B_4^7$ | $B_4^6$ | $B_4^5$ | $B_4^4$ |
| 000Ah | $\square_5^3$ | $\square_5^2$ | $\square_5^1$ | $\square_5^0$ | $R_5^7$ | $R_5^6$   | $R_5^5$ | $R_5^4$ | $G_5^7$ | $G_5^6$ | $G_5^5$ | $G_5^4$ | $B_5^7$ | $B_5^6$ | $B_5^5$ | $B_5^4$ |

### 5.3.2 Color Palette RAM

Table 3-7: Color Palette RAM

| 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 | $R_5^7$ | $R_5^6$ | $R_5^5$ | $R_5^4$ | $G_5^7$ | $G_5^6$ | $G_5^5$ | $G_5^4$ | $B_5^7$ | $B_5^6$ | $B_5^5$ | $B_5^4$ |

## 5.4 LCD Interface

ER-TFTMC070-4 Support 16, 18, 24bits RGB interface of TFT Panel, whether 24bpp (RGB 8:8:8), 16bpp (RGB 5:6:5) or 8bpp (RGB 3:3:2), the display data can be sent to the TFT Driver on the TFT panel through these RGB interfaces. The ER-TFTMC070-4 LCD Display data corresponds to the RGB data as shown in the below table. The Host can setup REG[01h] Bit[4:3] to select 16bits, 18bits or 24bits resolution.

Table 4-1: RGB Interface VS. RGB Display Data

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

Table 4-2: RGB Data Signals of ER-TFTMC070-4

| LCD Data Bus | RGB Signal Number | Colors      |
|--------------|-------------------|-------------|
| PD[23~0]     | R:G:B = 8:8:8     | 16.7M Color |

Figure 4-1 is the timing diagram of ER-TFTMC070-4 TFT-LCD output signals. In addition to the RGB data lines of above mentioned, ER-TFTMC070-4 also provides PCLK (Panel Scan Clock), VSYNC Pulse, HSYNC Pulse and Data Enable signal. The frequency of PCLK is setup by REG[05h] and REG[06h]. Please refer to the description in Section 1.1 and Chapter 14th.

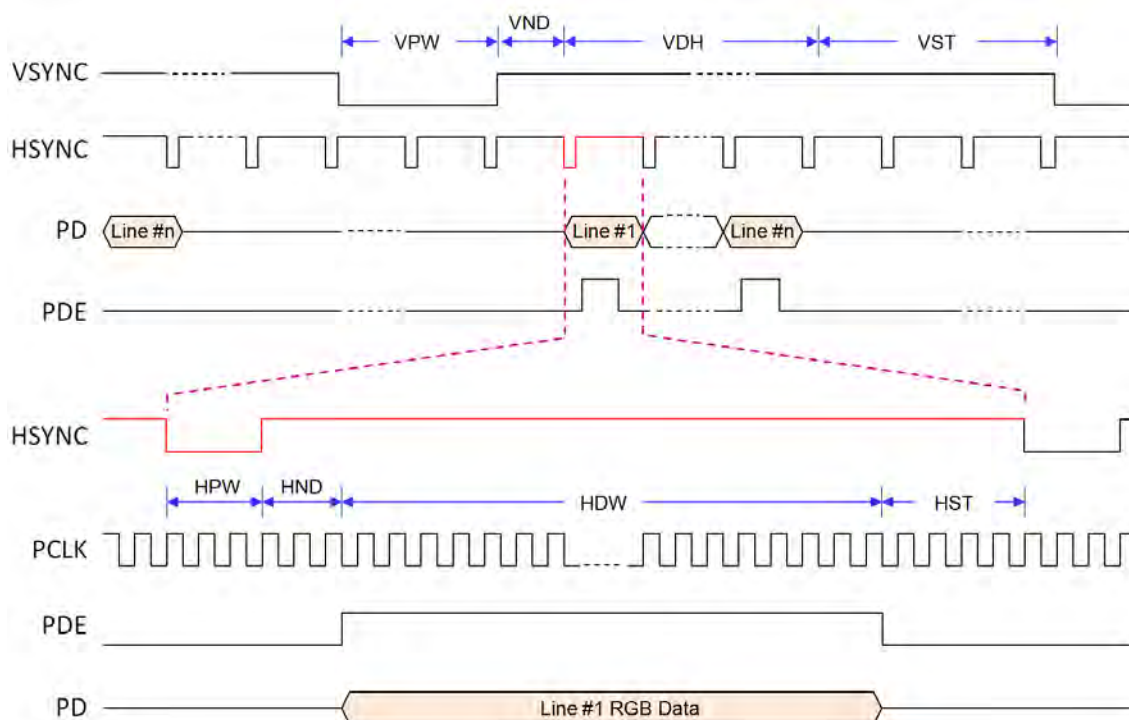


Figure 4-1: TFT-LCD Interface Timing

## 5.5 Display Function

### 5.5.1 Color Bar

The ER-TFTMC070-4 provides a color bar display, which can be used as a display test and does not require display memory. The function can be performed by Host to set REG[12h] bit5 to 1.

|             |   |                                     |
|-------------|---|-------------------------------------|
| Row number  |   |                                     |
| #1 ~ #32    |  | Color set to R(00h), G(00h), B(00h) |
| #33 ~ #64   |  | Color set to R(00h), G(00h), B(FFh) |
| #65 ~ #96   |  | Color set to R(00h), G(FFh), B(00h) |
| #97 ~ #128  |  | Color set to R(00h), G(FFh), B(FFh) |
| #129 ~ #160 |  | Color set to R(FFh), G(00h), B(00h) |
| #161 ~ #192 |  | Color set to R(FFh), G(00h), B(FFh) |
| #193 ~ #224 |  | Color set to R(FFh), G(FFh), B(00h) |
| #225 ~ #256 |  | Color set to R(FFh), G(FFh), B(FFh) |
| ⋮           | ⋮   | ⋮                                   |
| ⋮           | ( Repeat above eight Color )  | ⋮                                   |
| ⋮           | ⋮   | ⋮                                   |
| Last Row    |   |                                     |

Figure 5-1: Color Bar

### 5.5.2 Main Window

The LCD main window size can be defined by setting Registers REG[14h] to REG[1Fh]. At first, Host can save different images in memory buffer. Then setup the related Registers (REG[20h] ~ REG[29h]) to select a different buffer for show different images.

#### 5.5.2.1 Configure Display Image Buffer

Display RAM is used to store the displayed image. And how many images can be stored is determined by the image resolution and color. For example, the image resolution is 640x480, with 65K color (16bits), then 13 image data can be stored in 64Mbits Display RAM:

$$\text{Number of Image} = (64 \times 1024 \times 1024) / (640 \times 480 \times 16) = 13.6$$

If the image resolution is 800x600, with 256 color (8bits), then 34 image data can be stored in 64Mbits Display RAM:

$$\text{Number of Image} = (128 \times 1024 \times 1024) / (800 \times 600 \times 8) = 34.9$$

ER-TFTMC070-4 must set the Starting Position of the Canvas, Width and the Working window range before writing the image data to Display RAM. And also set the Working window range. Please refer the registers REG[50H] to REG[5Eh] for detail description.

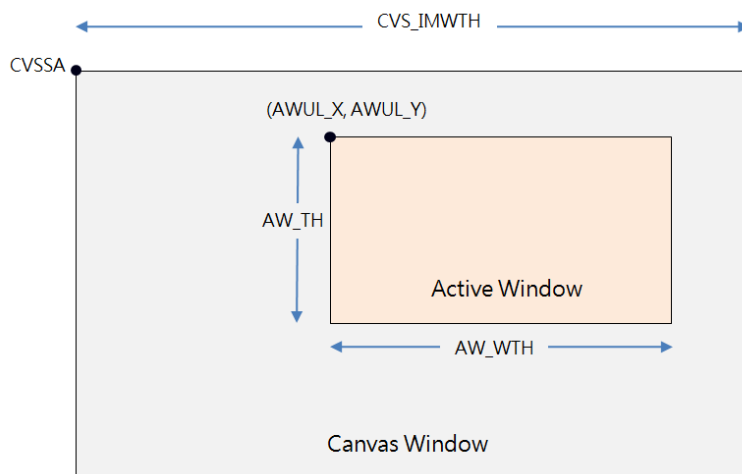


Figure 5-1: Canvas Window and Active Window

#### 5.5.2.2 Write Image Data to Display Image Buffer

The following diagram is a flowchart that writes image data to Display RAM and displays the main window image on an LCD screen:

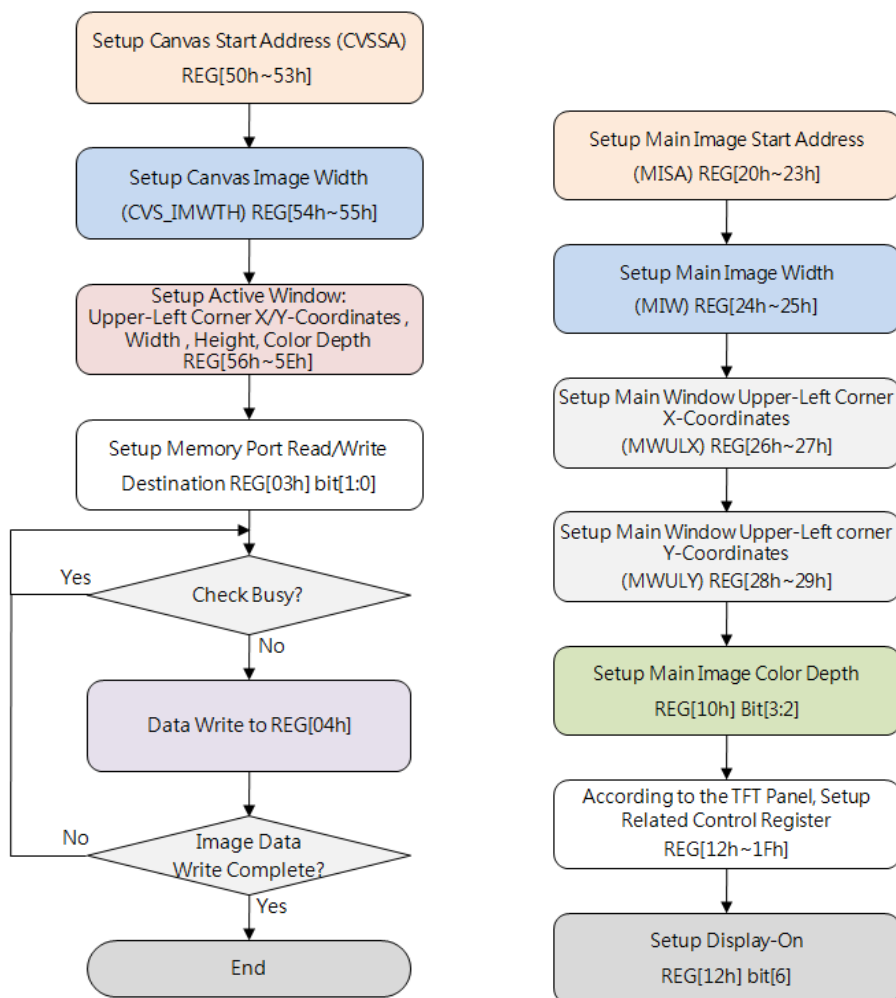


Figure 5-3: Write Image Data to Display Image Buffer

### 5.5.2.3 Display Main Window Image

Main window displays the image was selected by setting the Registers REG[20h] ~ REG[29h]. That is, by the register setting to select image within the Display RAM which images to be displayed. The following figure is the process for setting up the main window:

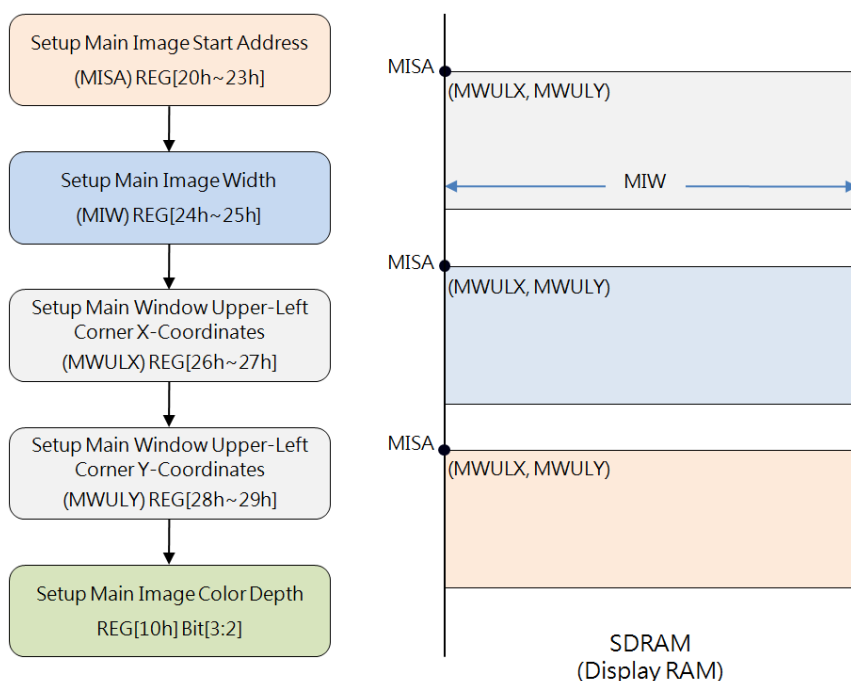


Figure 5-4: Setup and Select Main Window Image

### 5.5.3 Picture-In-Picture (PIP)

ER-TFTMC070-4 supports Picture-in-Picture feature. User can display secondary image(PIP-1) on the main screen(PIP-2) without to overwrite the image data of the main display window. If both images are overlapping, then the PIP-1 images is always on the top of PIP-2.

The size and location of the picture window is set by the register REG[2Ah] ~ REG[3Bh] and REG[11h]. Because the parameters of PIP-1 and PIP-2 are using the same registers, so REG[10H] Bit4 is used to choose PIP-1 or PIP-2 will be setup by these registers. And in the use of PIP function must first set the relevant parameters of the display window.

The unit of PIP windows sizes and start positions is 4 pixels in horizontal, and 1 pixel in vertical. In PIP mode, ER-TFTMC070-4 does not support the overlapping of transparent display, and when REG[12h] Bit3 VDIR = 1, then PIP windows, graphics cursors and text cursors functions will be automatically prohibited.

#### 5.5.3.1 PIP Window Setting

For using PIP feature, Host has to setup the PIP Image Start Address, Image Width, Display X/Y coordinates, Image X/Y coordinates, PIP Windows Color Depth, PIP Window Width and PIP Window Height registers. The following figure is the flowchart for setting the PIP and show the corresponding display to main window.

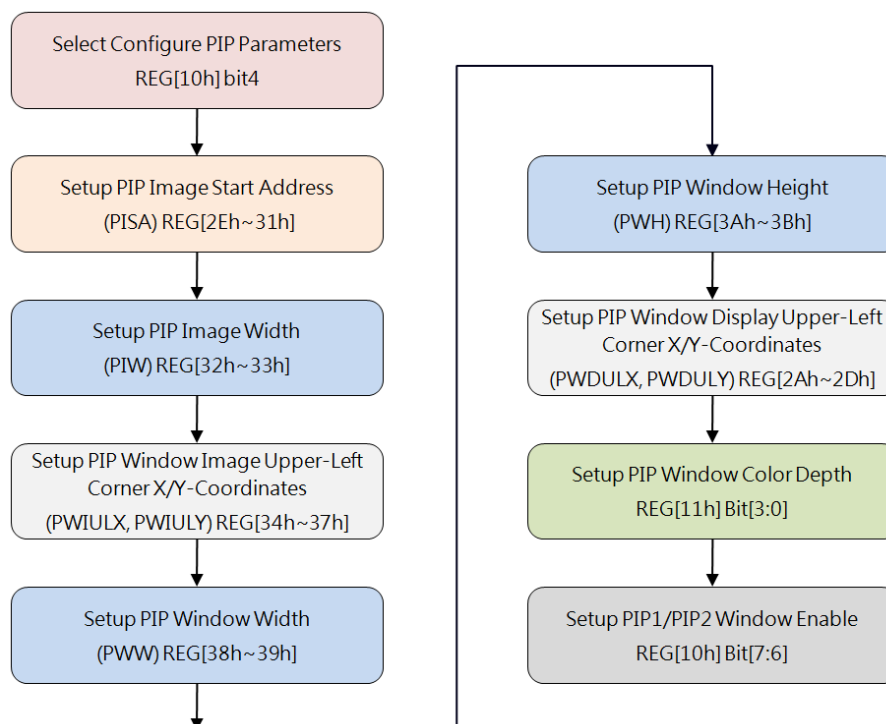


Figure 5-5: Initialize PIP Function



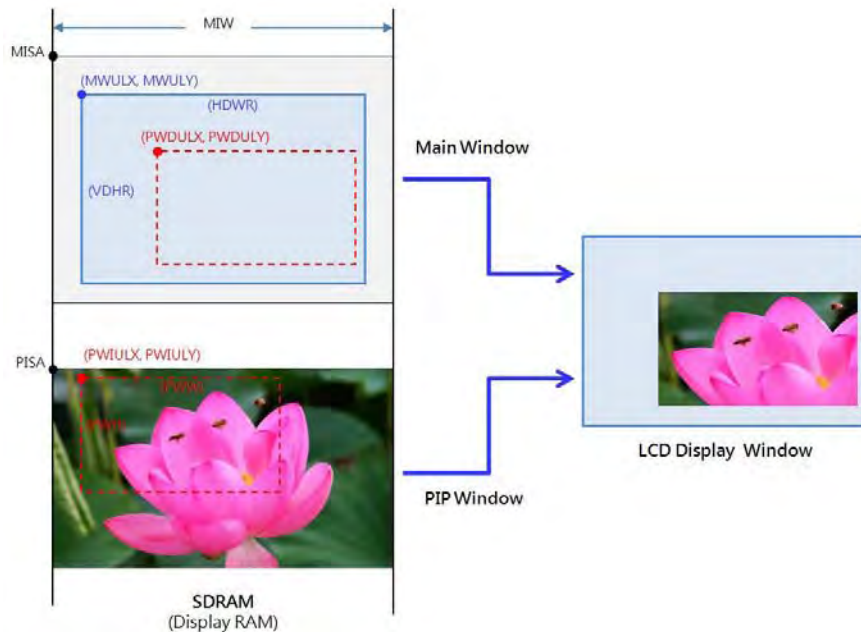


Figure 5-6: PIP Example

### 5.5.3.2 PIP Display Position and PIP Image Position

In the previous section of the PIP display flowchart, you can know that the display window can be set PWDULX and PWDULY to change the final position on the LCD screen. Setting PISA, PIW, PWIULX, PWIULY can change the position of the PIP images that you want to display. These actions do not change any image data that exists in Display RAM, but can be easily changed to be displayed picture on the panel. The following example shows a main window with a PIP window. You can show the different PIP image by changing the PIP Position Register(PWDULX and PWDULY) of the display window.

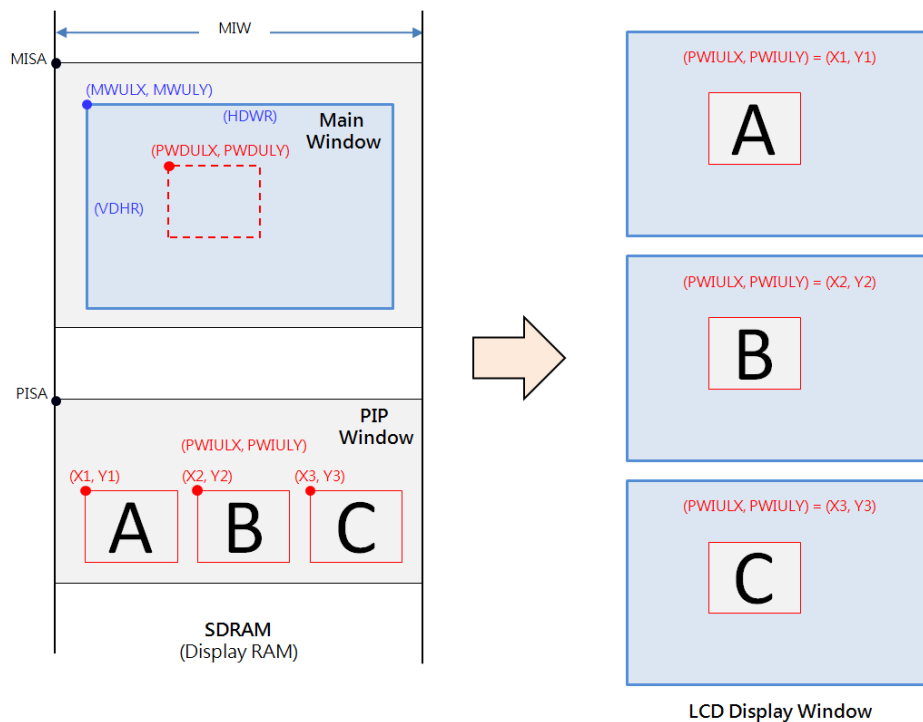


Figure 5-7: Select Different PIP image on the Screen

### 5.5.4 Image Rotate and Mirror

Usually LCD displays are refreshed horizontally (from left to right and from top to bottom), and the displayed images are stored in the same way. ER-TFTMC070-4 provides the ability to Rotate and Mirror, in which the rotate feature rotates the displayed image in the counterclockwise direction of 90° or 180°. The mirror feature is the image that is displayed from right to left to design mirroring. ER-TFTMC070-4 embedded a hardware controller to execute Rotation and Mirroring features, so it's greatly saving the Host processing time.

The REG[02H] bit[2:1] are used to control the Memory Store Direction in which the Host write Image Data to Display RAM. And these two bits is available 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 right 90° & Horizontal flip) 11b: Bottom

☐ Top, then Left ☐ Right (Rotate left 90°)

The following are some examples for Image Rotate and Mirror:



Figure 5-8: Original Image – without Rotation

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

When set REG[02H] bit[2:1] to 00b, its definition is to write image data from left to right and then top to bottom. This will show the original image that same as above. If set REG[02H] bit[2:1] to 01b, it means writing image data from right to left and then from top to bottom. So the image displayed will be a horizontal mirror image of the following:



Figure 5-9: Horizontal Mirror Image

When set REG[02H] bit[2:1] to 10b, it means writing image data from top to bottom and then left to right. So the displayed image will be rotated to the right 90° and then flipped horizontally as following Figure:



Figure 5-10: Rotated to the Right 90° and Flipped Horizontally

When set REG[02H] bit[2:1] to 11b, the write image is written from bottom to top and then left to right. So the display image will rotate 90° to the left as following Figure:



Figure 5-11: Rotate 90° to the Left

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

When set REG[02h] bit[2:1] to 00b, the display image will be as following Figure:



Figure 5-12: Flip with Vertical

When set REG[02h] bit[2:1] to 01b, the display image will be rotated 180°:



Figure 5-13: Rotated 180°

When set REG[02h] bit[2:1] to 10b, The displayed image will rotate 90° to the left:



Figure 5-14: Rotate 90° to the Left

When set REG[02h] bit[2:1] to 11b, the display image will be as following Figure:



Figure 5-15: Rotated to the Left 90° then Vertically Mirrored

## 5.6 Geometric Drawing Engine

### 5.6.1 Drawing Circle and Ellipse

ER-TFTMC070-4 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 the Color of the Circle (REG[D2h ~ D4h]), then specify REG[76h] bit[5:4] to 00b. Finally enable the Drawing Circle function (REG[76h] bit7 = 1), then ER-TFTMC070-4 will drawing a Circle or Ellipse on the screen in automatically.

Host can also set REG[76h] Bit6 to 1 to fill a Circle or an Ellipse. Therefore, Host(MCU) doesnot need use many resources to calculate the location and a series of writing data to display memory. The Ellipse has two radius. For drawing Circle, just set the long radius and short radius register to the same value ( $R1 = R2$ ).

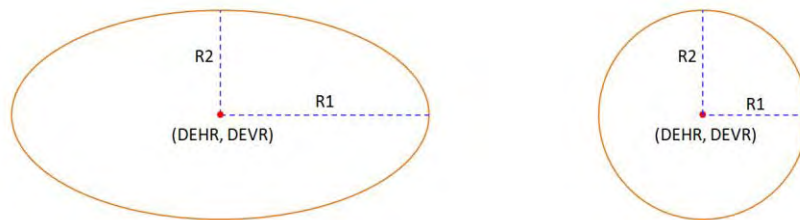


Figure 6-1: Drawing Ellipse and Circle

The flowchart for drawing circles and drawing ellipses is as follows. The left flowchart is draw a circle or ellipses without fill, and the right flowchart is with fill. Please note the center point of Circle must be located in the Active Window.

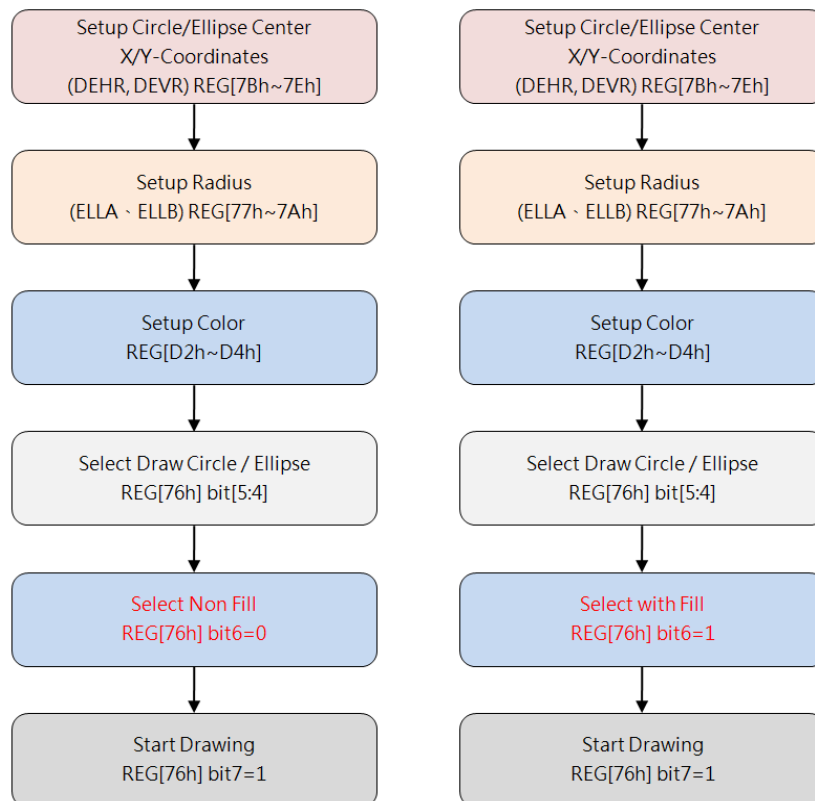


Figure 6-2: Flowchart of Drawing Circle and Ellipse

## 5.6.2 Drawing Curve

ER-TFTMC070-4 supports the function of drawing Curve. As long as the Host sets the Curve Center (REG[7Bh ~ 7Eh]), Radius (REG[77h ~ 7Ah]), and the Color of the Curve (REG[D2h ~ D4h]), then specify REG[76h] bit[5:4] to 01b. Finally enable the Drawing Curve function (REG[76h] bit7 = 1), then ER-TFTMC070-4 will drawing a one-fourth Curve on the screen in automatically.



Figure 6-3: Drawing Curve and one-fourth Ellipse

The flowchart for drawing a Curve or drawing one-fourth Ellipses is as follows. The left flowchart is draw a Curve or one-fourth Ellipse without fill, and the right flowchart is with fill. Please note the center point of Curve must be located in the Active Window.

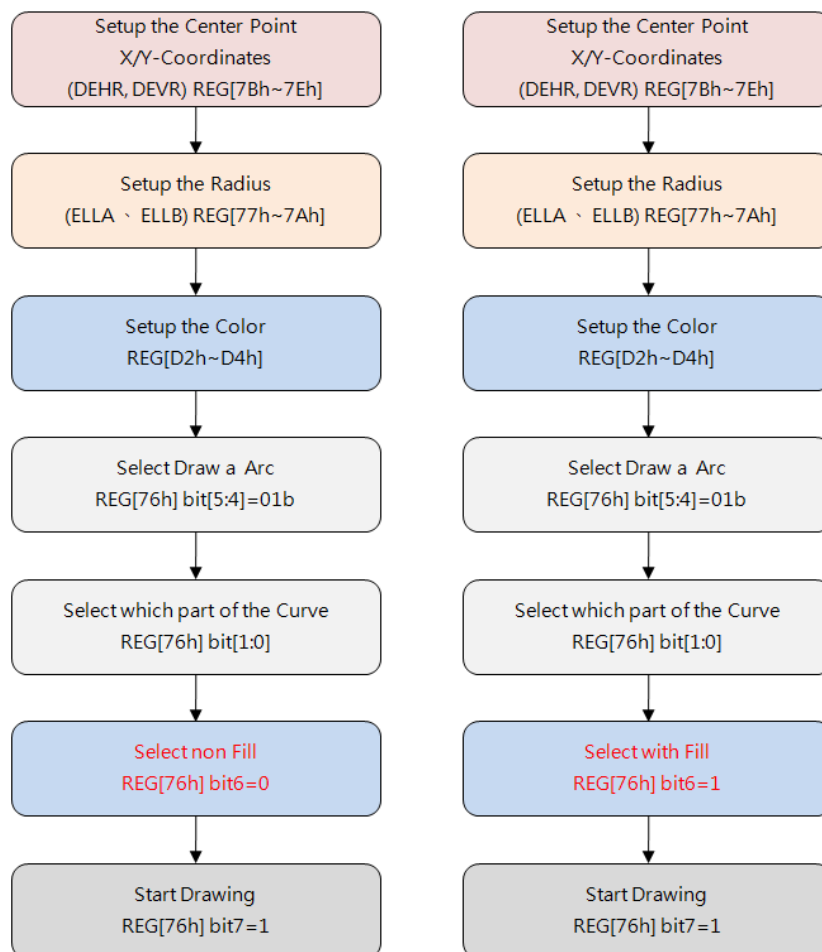


Figure 6-4: Flowchart of Drawing Curve and one-fourth Ellipse

### 5.6.3 Drawing Rectangle

For Drawing a Rectangle, the Host has to sets the Start Point Coordinates (REG[68h ~ 6Bh]), Stop Point Coordinates (REG[6Ch ~ 6Fh]), and the Color of the Rectangle (REG[D2h ~ D4h]), then specify REG[76h] bit[5:4] to 10b. Finally enable the drawing function (REG[76h] bit7 = 1), then ER-TFTMC070-4 will drawing a Rectangle on the screen in automatically. Host can also set REG[76h] Bit6 to 1 to fill the Rectangle with specified color.

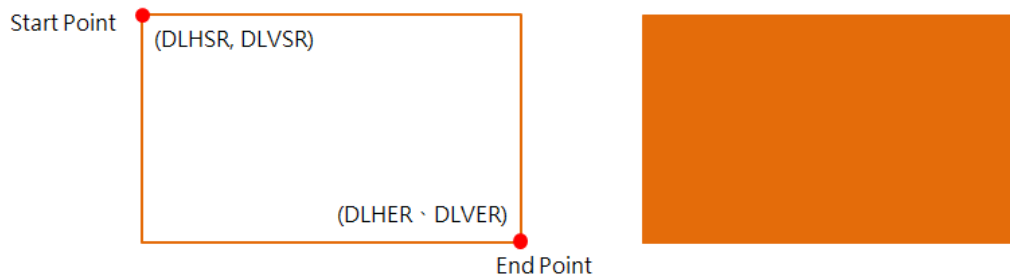


Figure 6-5: Drawing Rectangle

The following figure is the flowchart of drawing Rectangle. The left flowchart is draw a Rectangle without fill, and the right flowchart is with fill. Please note the Start and Stop point must be located in the Active Window.

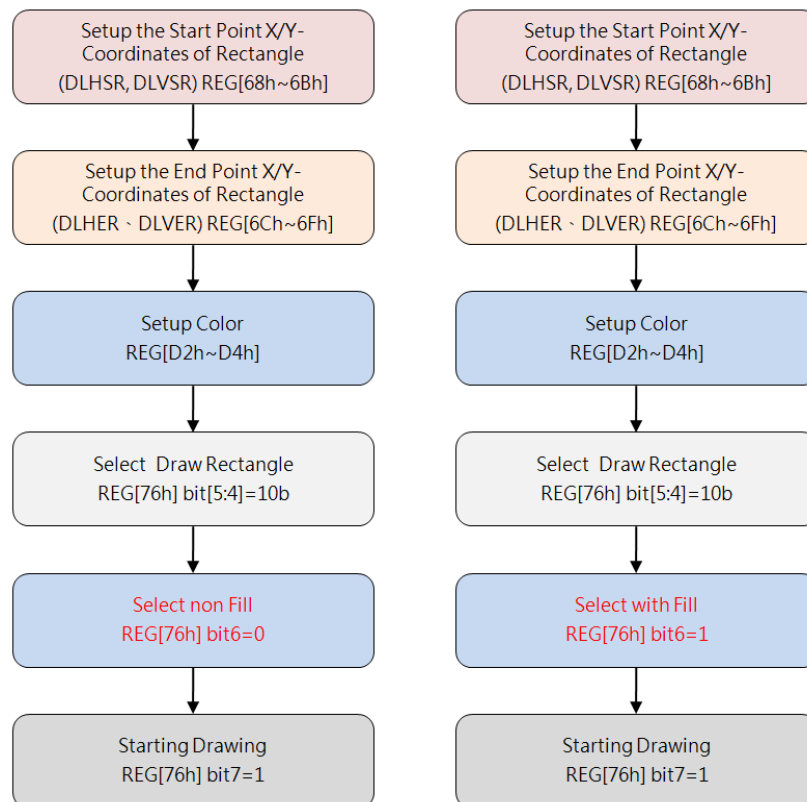


Figure 6-6: Flowchart of Drawing a Rectangle

#### 5.6.4 Draw Line

For Drawing a Line, the Host has to sets the Start Point Coordinates (REG[68h ~ 6Bh]), Stop Point Coordinates (REG[6Ch ~ 6Fh]), and the Color of the Line (REG[D2h ~ D4h]), then specify and specify REG[67h] bit1 to 0. Finally enable the Line Drawing function (REG[67h] bit7 = 1), then ER-TFTMC070-4 will drawing a Line on the screen in automatically.

The following figure is the flowchart of Line drawing. Please note the Start and Stop point must be located in the Active Window.

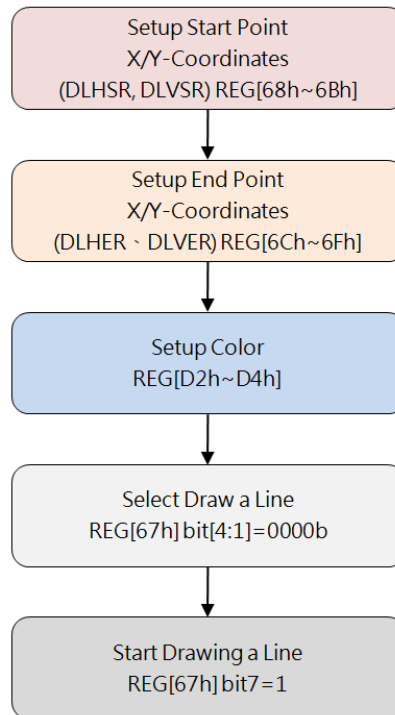


Figure 6-7: Flowchart of Drawing a Line



### 5.6.5 Drawing Triangle

For Drawing a Triangle, the Host has to sets the 1<sup>ST</sup> Point Coordinates (REG[68h ~ 6Bh]) of Triangle, 2<sup>nd</sup> Point Coordinates (REG[6Ch ~ 6Fh]), 3<sup>rd</sup> Point Coordinates (REG[70h ~ 73h]), and the Color of the Triangle (REG[D2h ~ D4h]), then specify REG[67h] bit1 to 1. Finally enable the drawing function (REG[67h] bit7 = 1), then ER-TFTMC070-4 will drawing a Triangle on the screen in automatically. Host can also set REG[67h] bit5 to 1 to fill the Rectangle with specified color.

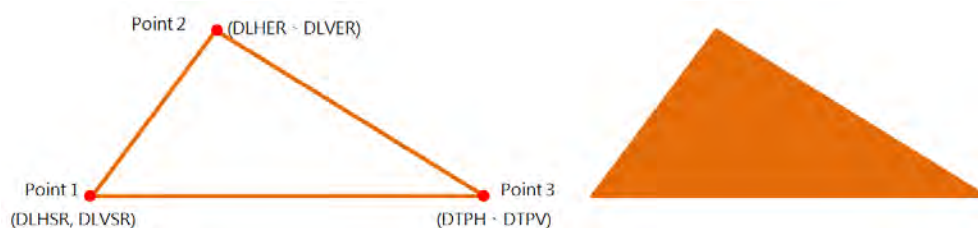


Figure 6-8: Drawing Triangle

The following figure is the flowchart of Triangle drawing. The left flowchart is draw a Triangle without fill, and the right flowchart is with fill. Please note the three points must be located in the Active Window.

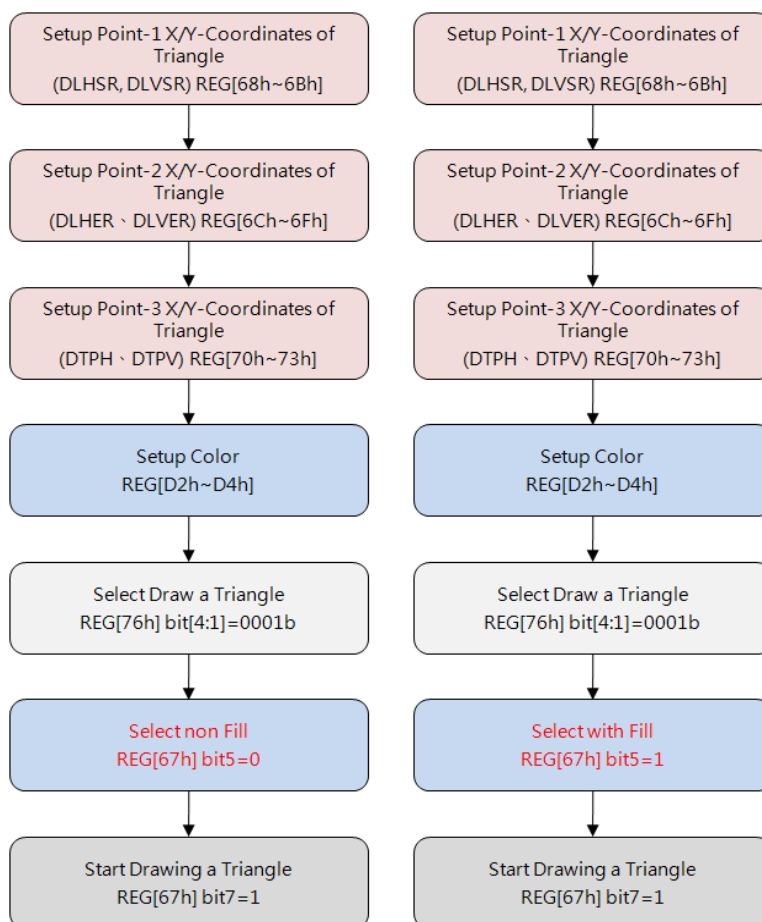


Figure 6-9: Flowchart of Drawing a Triangle

### 5.6.6 Drawing Rounded-Rectangle

For Drawing a Rounded-Rectangle, the Host has to sets the Start Point Coordinates (REG[68h ~ 6Bh]), Stop Point Coordinates (REG[6Ch ~ 6Fh]), the Rounded Radius (REG[77h ~ 7Ah]) and the Color (REG[D2h ~ D4h]), then specify REG[76h] bit[5:4] to 11b. Finally enable the drawing function (REG[76h] bit7 = 1), then ER-TFTMC070-4 will drawing a Rounded-Rectangle on the screen in automatically. Host can also set REG[76h] Bit6 to 1 to fill the Rounded-Rectangle with specified color. The following figure is a schematic of a Rounded-Rectangle, in which the R1 represents the long axis radius, and the R2 represents the short axis radius.

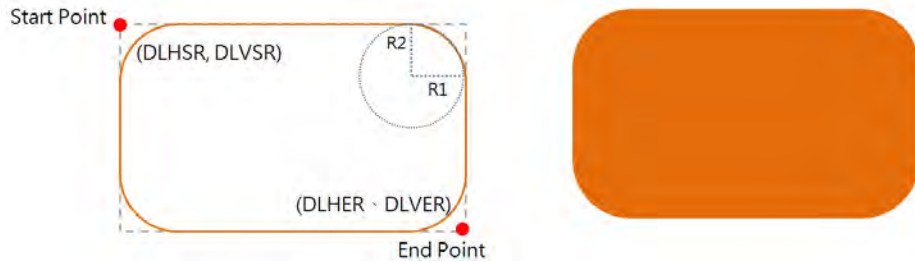


Figure 6-10: Drawing Rounded Triangle

**Note1:** DLHER-DLHSR must large than  $2 \times R1 + 1$

**Note2:** DLVER-DLVSR must large than  $2 \times R2 + 1$

The following figure is the flowchart of drawing Rounded-Rectangle. The left flowchart is draw a Triangle without fill, and the right flowchart is with fill. Please note the Start and Stop point must be located in the Active Window.

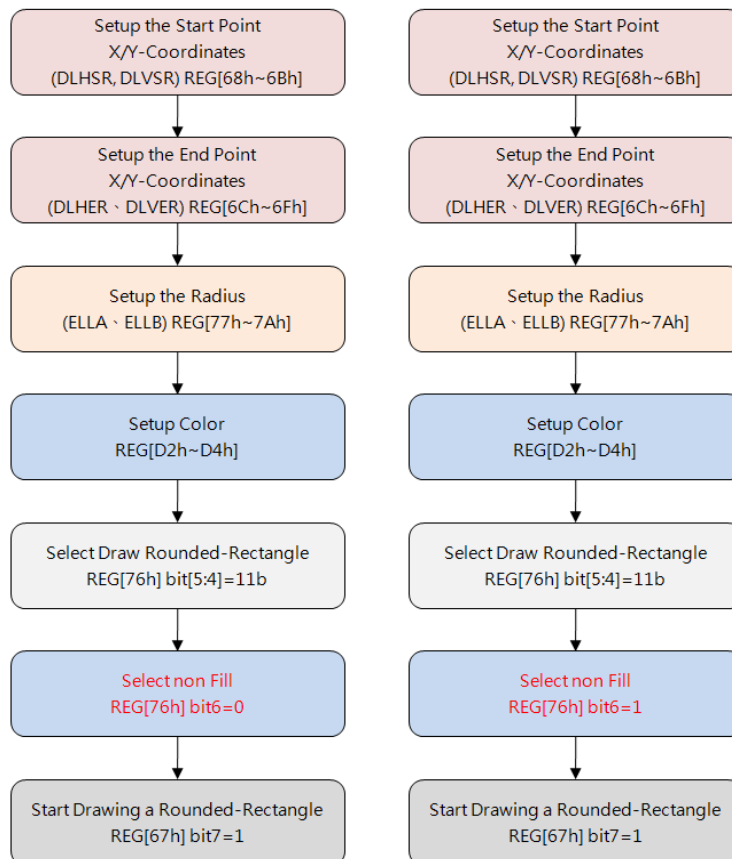


Figure 6-11: Flowchart of Drawing a Rounded-Rectangle

## 5.7 Block Transfer Engine (BTE)

ER-TFTMC070-4 has a embedded high performance hardware engine - Block Transfer Engine (BTE), it is designed to accelerate data Loading, Transferring, and additional Logic Processing. BTE supports 13 basic BTE Operations (Table 7-1), as well as additional functions of Raster Operation (ROP, Table 7-2), Chroma Key, Color Expansion, and so on. After getting properly set and enabled, BTE can perform the required operations and functions automatically and rapidly regardless of MCU. BTE can extremely release MCU working loads and resources, further to greatly improve the performance for all the system.

If a data block needs to be loaded, moved, processed at one times, user could consider to use BTE, by any available combinations of basic BTE Operations and additional Functions as well as various available logic combinations of Source and Destination, further to achieve many useful application purposes. Once the BTE started under proper settings, it will keep active working (busy) until the working completed. There are two ways to get BTE working status, the one is to monitor Hardware Interruption: connecting INT# to MCU, once the interruption presented, then MCU responds and starts the routine to get the Interruption Source by checking REG[0Ch] ; the another one is to check corresponding Flag Bits of register BTE\_CTRL0 (REG[90h]) Bit4, or Status Register (STSR) Bit3.

Table 7-1: BTE Operation

| BTE Operation Code<br>REG[91h] Bits [3:0] | BTE Operation Description  |
|---|--|
| 0000b                                     | MCU Write with ROP.  |
| 0010b                                     | Memory Copy (move) with ROP.                                     |
| 0100b                                     | MCU Write with chroma keying (without ROP)                       |
| 0101b                                     | Memory Copy (move) with chroma keying (without ROP)              |
| 0110b                                     | Pattern Fill with ROP  |
| 0111b                                     | Pattern Fill with chroma keying (without ROP)                    |
| 1000b                                     | MCU Write with Color Expansion (without ROP)                     |
| 1001b                                     | MCU Write with Color Expansion and chroma keying (without ROP)   |
| 1010b                                     | Memory Copy with opacity (without ROP)                           |
| 1011b                                     | MCU Write with opacity (without ROP)                             |
| 1100b                                     | Solid Fill (without ROP)   |
| 1110b                                     | Memory Copy with Color Expansion (without ROP)                   |
| 1111b                                     | Memory Copy with Color Expansion and chroma keying (without ROP) |
| Other Combinations                        | Reserved   |

Table 7-2: ROP Function

| ROP Function Code<br>REG[91h] bit[7:4] | Function Description (Boolean)              |
|--|---|
| 0000b                                  | 0 (Blackness)                               |
| 0001b                                  | $\sim S0 \cdot \sim S1$ or $\sim (S0+S1)$   |
| 0010b                                  | $\sim S0 \cdot S1$                          |
| 0011b                                  | $\sim S0$                                   |
| 0100b                                  | $S0 \cdot \sim S1$                          |
| 0101b                                  | $\sim S1$                                   |
| 0110b                                  | $S0 \wedge S1$                              |
| 0111b                                  | $\sim S0 + \sim S1$ or $\sim (S0 \cdot S1)$ |
| 1000b                                  | $S0 \cdot S1$                               |
| 1001b                                  | $\sim (S0 \wedge S1)$                       |
| 1010b                                  | $S1$  |
| 1011b                                  | $\sim S0 + S1$                              |
| 1100b                                  | $S0$  |
| 1101b                                  | $S0 + \sim S1$                              |
| 1110b                                  | $S0 + S1$                                   |
| 1111b                                  | 1 (Whiteness)                               |

Note:

1. Source 0 (S0) Data: comes from MCU Writing or Memory
2. Source 1 (S1) Data: comes from Memory,
3. Destination (DT) Data: be wrote to Memory
4. Memory means internal Display RAM
5. Memory, S0, S1, DT, these Short Names will be always quoted in this Chapter. For Example: If ROP function REG[91h] bit[7:4]=0xCh, then "DT = S0", means "Transfer Source 0 data to Destination"; If ROP function REG[91h] bit[7:4]=0xEh, then "DT = S0 + S1", means "Source 0 data + Source 1 data then transfer to Destination"

Table 7-3: Color Expansion Function

| ROP Function<br>REG[91h] bit[7:4] | Start Bit Position for Color Expansion BTE<br>operation code = 1000/1001/1110/1111 |                     |
|-----------------------------------|--|---------------------|
|                                   | 16bits MCU Interface   | 8bits MCU Interface |
| 0000b                             | Bit0   | Bit0                |
| 0001b                             | Bit1   | Bit1                |
| 0010b                             | Bit2   | Bit2                |
| 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             |

### 5.7.1 BTE Basic Settings

The BTE basic settings for all two Sources and one Destination, including Memory Start Address, Image Width, and Start Point Position (Coordinate) for each, could be defined by following registers:

1. S0 Address Registers: REG [93h], REG[94h], REG[95h], REG[96h], REG[97h], REG [98h], REG[99h], REG[9Ah], REG[9Bh], REG[9Ch]
2. S1 Address Registers: REG [9Dh], REG[9Eh], REG[9Fh], REG[A0h], REG [A1h], REG[A2h], REG[A3h], REG[A4h], REG[A5h], REG[A6h]
3. DT Address Registers: REG [A7h], REG[A8h], REG[A9h], REG[AAh], REG[ABh], REG[ACH], REG[ADh], REG[A Eh], REG[AFh], REG[B0h]

### 5.7.2 Color Palette RAM

The ER-TFTMC070-4 has a embedded Color Palette RAM for 8-bits Alpha Blend function, organized by 64x12bits. Color Index is an Address of Color Palette RAM space to save a series of 12-bits Word of needed color data (refer to Figure 7-1). Color Palette RAM must be written by 128 Bytes (8-bits per Byte) sequence in one times, but cannot be read, bit[7:4] of the Byte with Even Sequence Number will be discarded. The diagram Figure 7-2 shows the initialization flow.

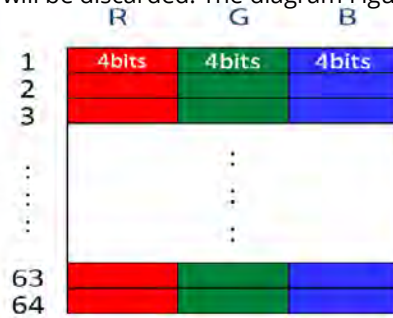


Figure 7-1: Color Palette RAM Organization

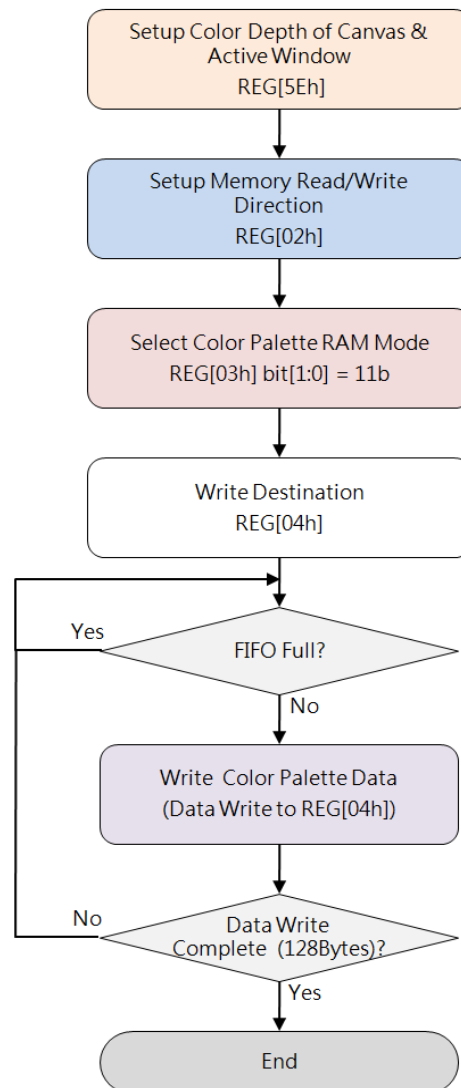


Figure 7-2: Flow Chart of Color Palette RAM Initialization

### 5.7.3 BTE Operation Overview

#### **MCU Write with ROP**

This Operation supports 16 ROP functions, BTE engine will perform the ROP function automatically then write the result data to DT.

#### **Memory Copy with ROP**

This Operation supports 16 ROP functions, but supports data transfer in positive direction only.

#### **Solid Fill**

This Operation fills a specified Rectangle Area of DT with a Solid Color data defined in the Foreground Color Register.

#### **Pattern Fill**

This Operation duplicates to fill DT with an 8x8 or 16x16 pixel Pattern.

#### **Pattern Fill with Chroma Key**

This Operation duplicates to fill DT with an 8x8 or 16x16 pixel Pattern, combined with Chroma Key function but without ROP function. If Pattern Color is equal to the Chroma Key Color, which is defined in Background Color Register, destination BTE Window will not be changed.

#### **MCU Write with Chroma Key**

This Operation supports data transfer from S0 (MCU Write) to DT. If S0 Color is equal to the Chroma Key Color, which is defined in Background Color Register, destination BTE Window will not be changed, no ROP applied for this operation.

#### **Memory Copy with Chroma Key**

This Operation supports data transfer in positive direction only, and requires all source data and destination data located in Memory. If S0 Color is equal to Chroma Key color, which is defined in Background Color Register, destination BTE Window will not be changed, no ROP applied for this operation.

#### **MCU Write with Color Expansion**

This Operation performs Color Expansion for the S0 (the data from MCU), from Monochrome 1 bpp format to Color 8/16/24 bpp format. The source data\_1b will expand to the color data defined in the Foreground Color Register. The source data\_0b will expand to the color data defined in the Background Color Register. If background transparency is enabled, then the Color of destination BTE Window will remain no change.

**Note:** No matter background transparency is enabled or not, must set the Different Color data in Foreground Color Register (D2h~D4h) and Background Color Register (D5h~D7h).

#### **Memory Copy with Color Expansion**

This Operation performs Color Expansion for Source 0 data from Memory,.

#### **Memory Copy with Opacity**

This Operation performs Alpha Blending for S0 data and S1 data and transferring the result data to the destination BTE Window, it requires S0, S1, DT located in Memory. The Alpha Blending has 2 modes, Picture Mode: for all pixels, blending by a mutual Alpha the level data saved in the register. Pixel Mode: for each pixel, blending by individual Alpha Level of each pixel.

#### **MCU Write with Opacity**

This Operation performs Alpha Blending for S0 data and S1 data and transferring the result data to destination BTE Window, it requires S0 coming from MCU and S1/DT from Memory, please refer to 7.3.10 for the descriptions of 2 modes.

### 5.7.4 BTE Memory Access Method

BTE accesses data of Sources and Destination by Block Method, and the size of the Data Block is defined by BTE Window. Below diagram shows how to define S0 / S1 / DT / BTE\_Window, and shows the Directions of Memory Access:

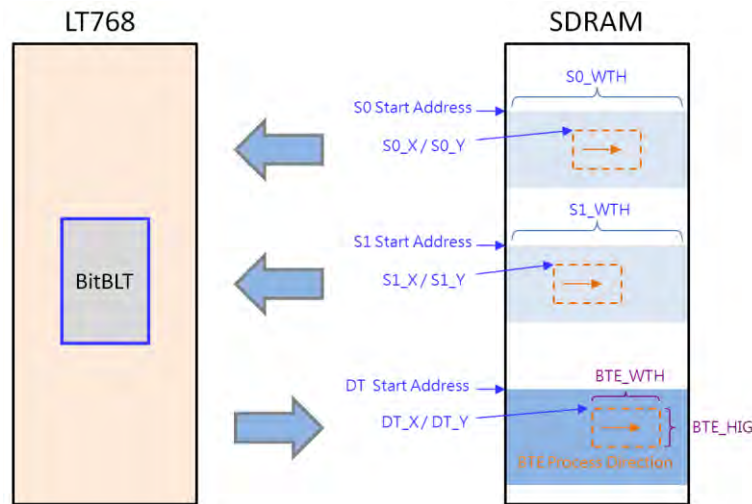


Figure 7-3: BTE Memory Access Method

### 5.7.5 BTE Chroma Key (Transparency Color) Function

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

Below shows the available bits of the Background Color Registers against different Color Depth:

#### 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 = 65,536

- 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 = 16,777,216

- 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]



## 5.7.6 BTE Operation Detail

### 5.7.6.1 MCU Write with ROP

This Operation performs to transfer S0 (form MCU Write) to DT, and supports all 16 ROP functions. Below diagram is an example for the operation result while BTE Control Register 1 REG[91h] set as 0xC0h.

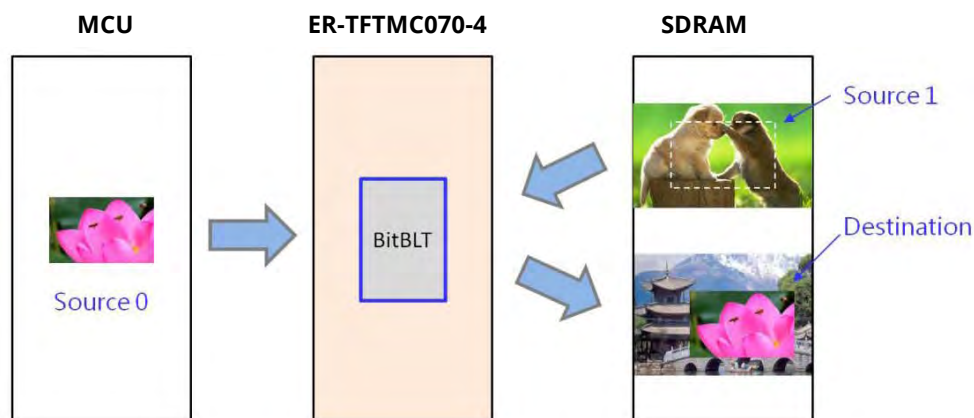


Figure 7-4: Example of MCU Write with ROP

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

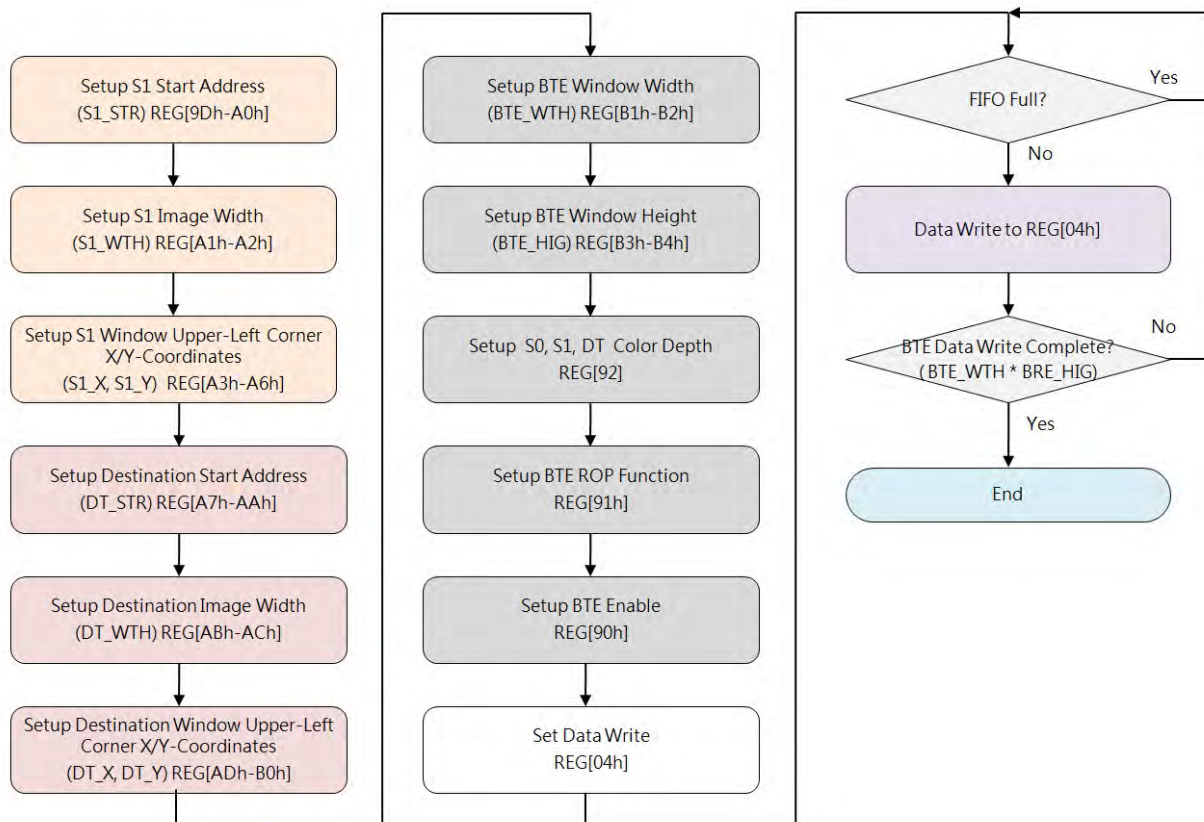


Figure 7-5: Flow Chart of MCU Write with ROP

### 5.7.6.2 Memory Copy (move) with ROP

This Operation performs Memory Copy from S0 (from Memory) to DT. Below diagram is an example for the operation result while BTE Control Register 1 REG[91h] set as 0xC2h.

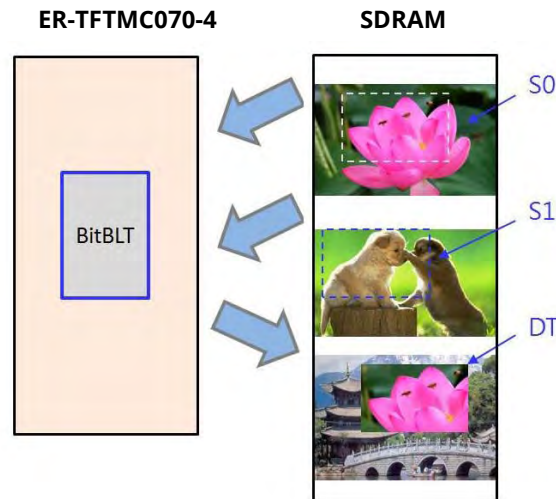


Figure 7-6: Example of Memory Copy with ROP

Figure 7-7 is Flow Chart by checking Status Register STSR bit3 to monitor BTE busy or not.  
Figure 7-8 is Flow Chart by responding Hardware Interruption to get BTE processing result.

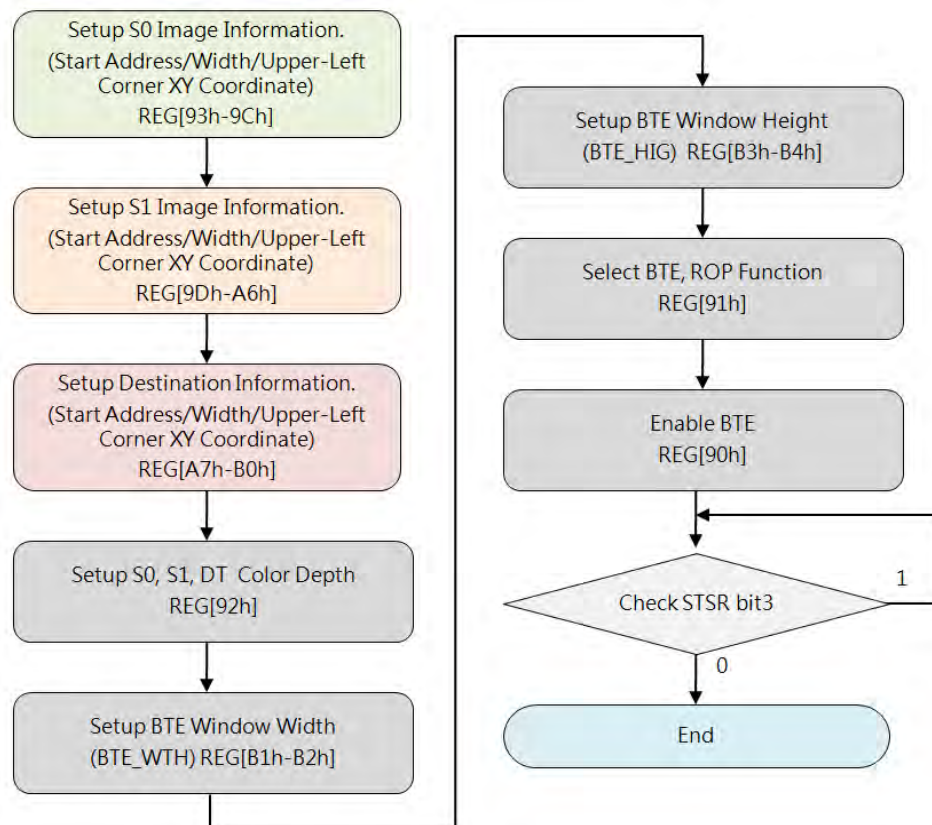


Figure 7-7: Flow Chart 1 of Memory Copy with ROP

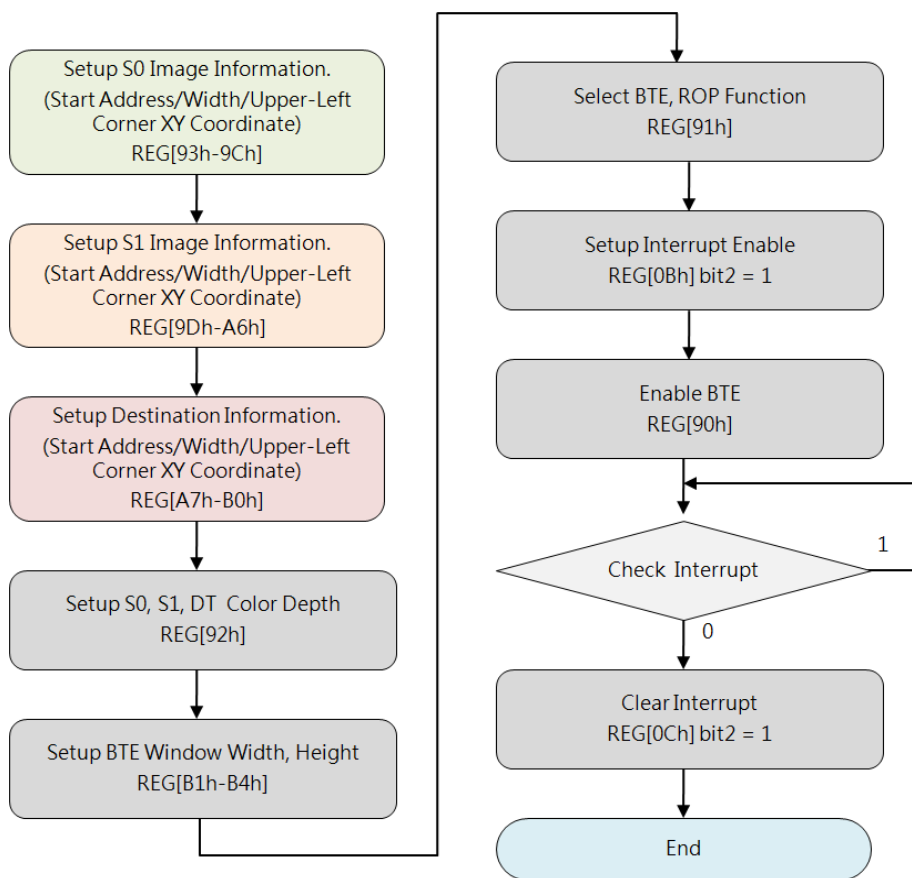


Figure 7-8: Flow Chart 2 of Memory Copy with ROP

### 5.7.6.3 MCU Write with Chroma Key (w/o ROP)

This Operation performs to transfer S0 (from MCU Write) to DT, and supports Chroma Key function (referring to 7.5).

If S0 Color Data is equal to the Chroma Key (the color Data defined in the Background Color Registers REG[D5h-D7h]), BTE will take that Color as Transparent, means BTE will discard writing that Color data to DT. Below Example shows GREEN is background color of RED "TOP", and GREEN is set as Chroma Key, BTE will write RED "TOP" only to DT:

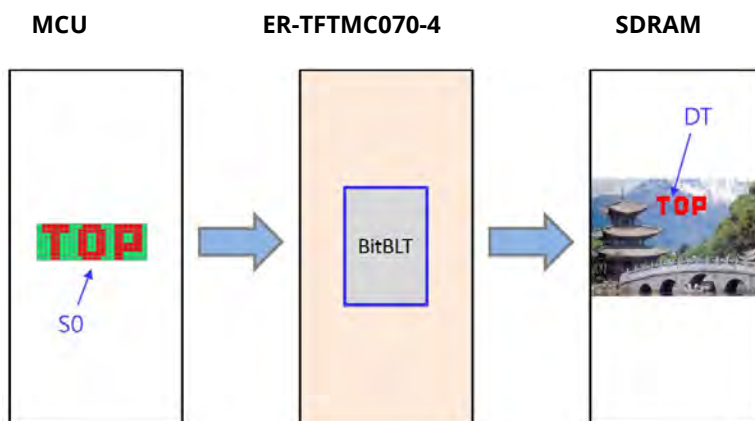


Figure 7-9: Example of MCU Write with Chroma Key

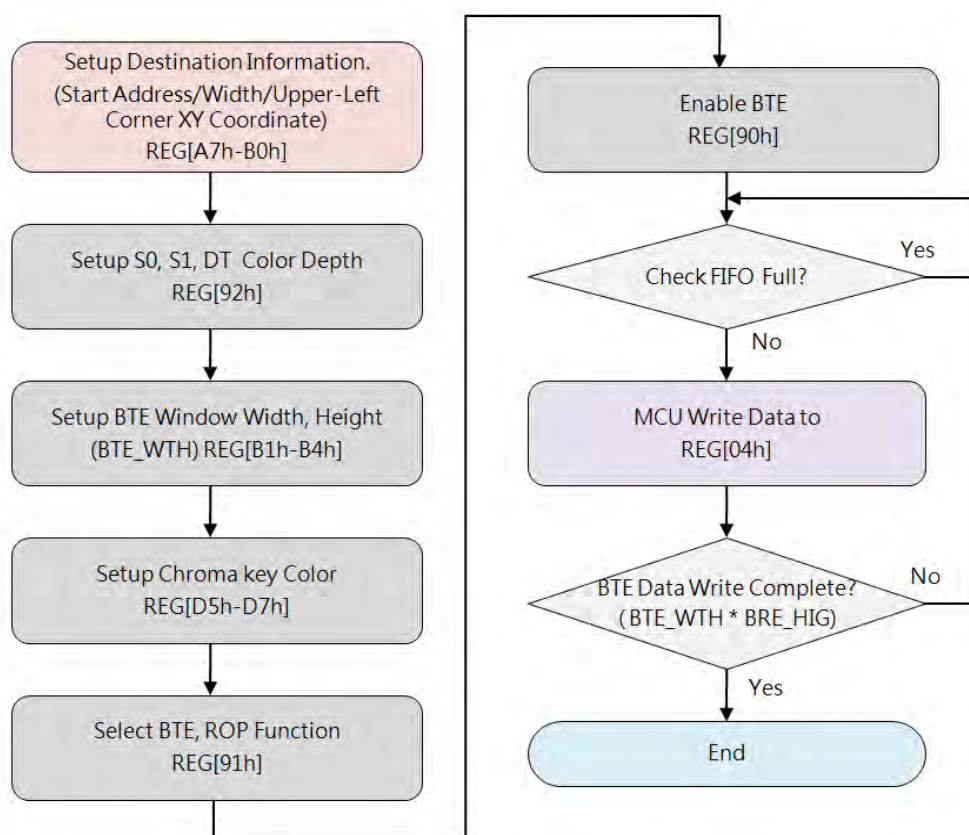


Figure 7-10: Flow Chart of MCU Write with Chroma Key

#### 5.7.6.4 Memory Copy with Chroma Key (w/o ROP)

This Operation performs Memory Copy from S0 (from Memory) to DT, and supports Chroma Key function. The difference comparing with "MCU Write with Chroma Key" is that S0 comes from Memory but not MCU Write.

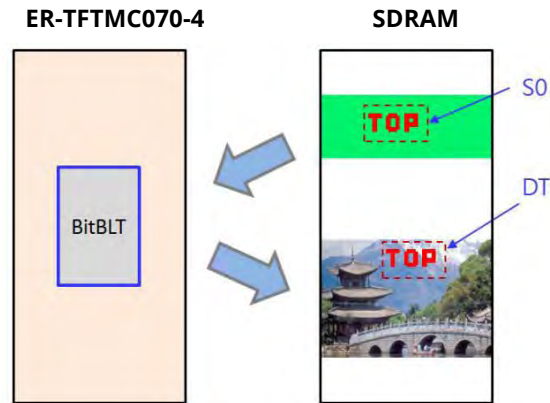


Figure 7-11: Example of Memory Copy with Chroma Key

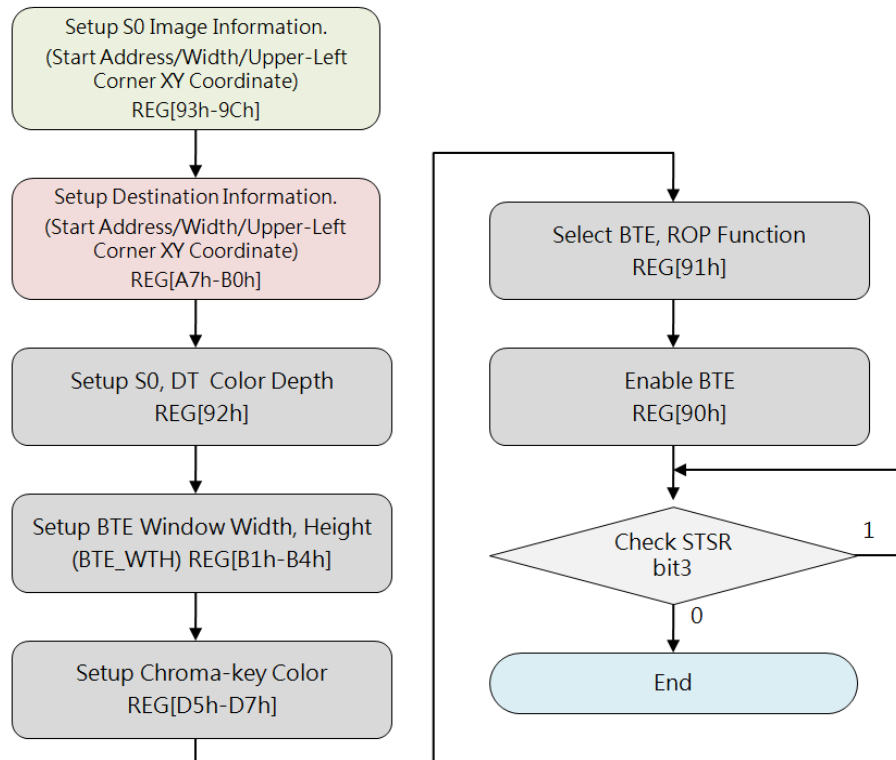


Figure 7-12: Flow Chart of Memory Copy with Chroma Key



### 5.7.6.5 Pattern Fill with ROP

This Operation performs to fill a pattern (as S0) repeatedly to a specified Rectangular Area of Destination (also known as BTE Window). The pattern should be an array of 8x8 or 16x16 pixels stored in Memory. The pattern can be logically processed with S1 by using one of the 16 ROP functions. This Operation can be used to speed up duplicating a matrix of pattern into an area, such as background paste function.

Below example shows a six fills with 8x8 Pattern, ROP REG[91h] bit[7:4] set as 0x0C:

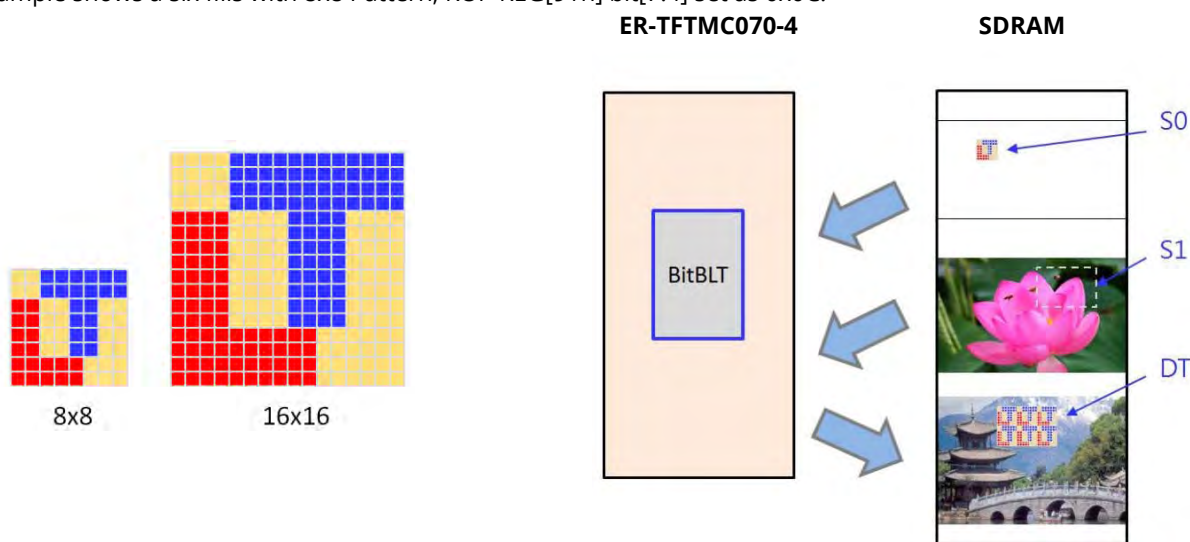


Figure 7-13: Pattern Format

Figure 7-14: Example of Pattern Fill with ROP

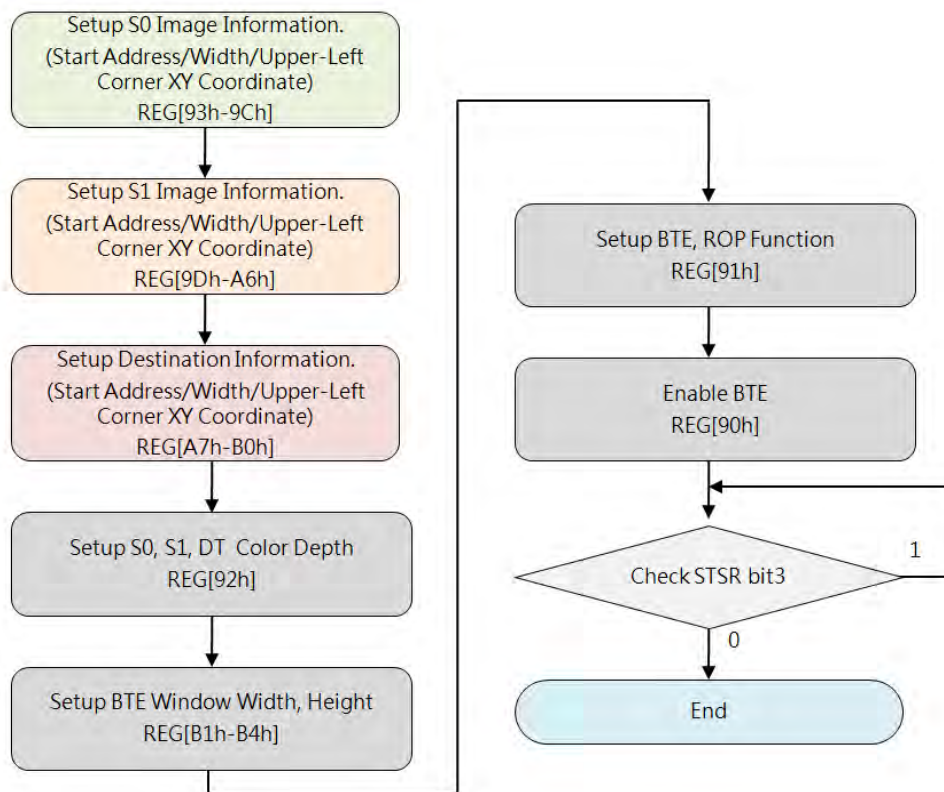


Figure 7-15: Flow Chart of Pattern Fill with ROP

#### 5.7.6.6 Pattern Fill with Chroma Key

This Operation performs to fill a pattern (as S0) repeatedly to a specified Rectangular Area of DT (also known as BTE Window), and supports Chroma Key function (referring to Section 7.5). If any partial Color Data of the pattern is equal to the Chroma Key data, BTE will discard that partial fillings.

Below Example shows ORANGE is background color of RED "T", and ORANGE is set as Chroma Key, BTE will fill RED "T" only to DT one by one:

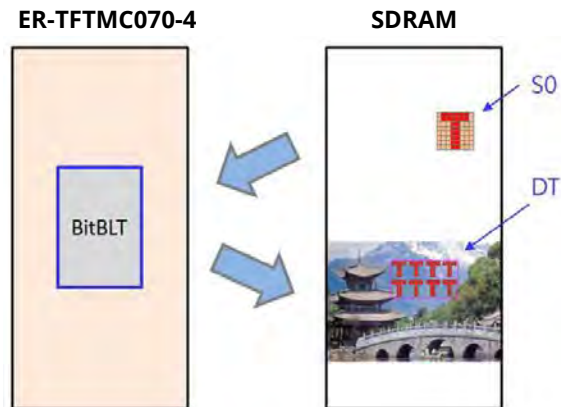


Figure 7-16: Example of Pattern Fill Chroma Key

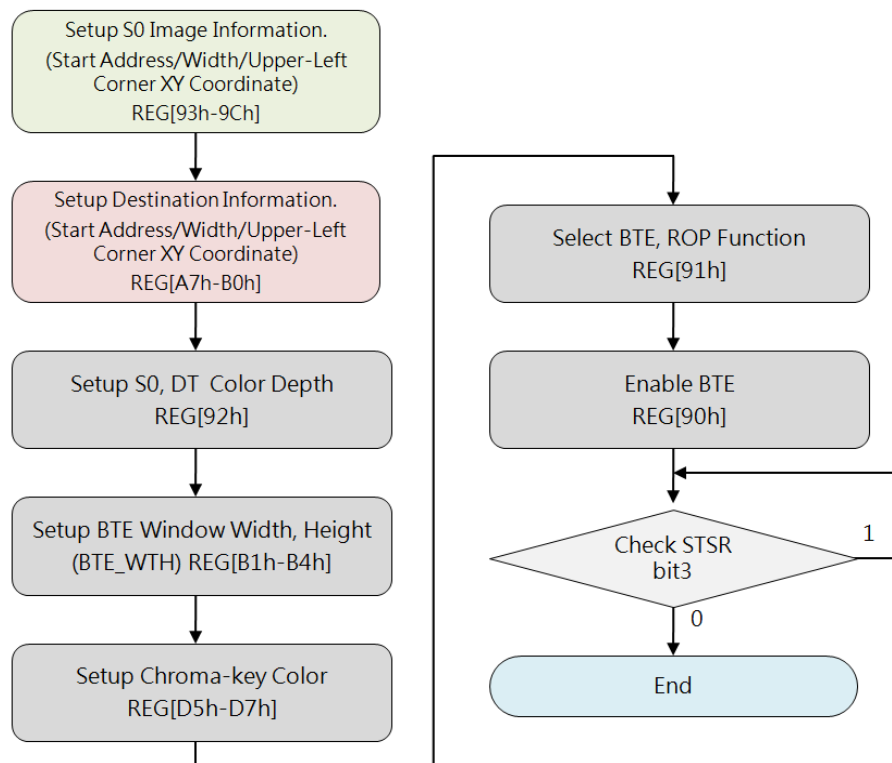


Figure 7-17: Flow Chart of Pattern Fill with Chroma Key

#### 5.7.6.7 MCU Write with Color Expansion

This Operation performs Color Expansion for S0 (from MCU Write), it's useful to translate bit-wise monochrome data to byte-wise color data. In this operation, S0 Color Depth REG[92h] Bit[6:5] is ignored, BTE takes MCU Interface Width as S0 Color Depth (Word Width), such as 8-bits MCU Interface the Color Depth is 8-bits and 16-bits MCU Interface the Color Depth is 16-bits. User should set needed Start Bit to REG[91h] Bit[7:4] against the corresponding Color Depth, so Bit[7]~Bit[0] are available for 8-bits depth and Bit[15]~Bit[0] are available for 16-bits depth. BTE disassembles Word by Word to Bit sequences (from LSB to MSB) against the ROW Line of source image (from left to right), and sequently expands bit by bit until the BTE Width reached ending. The result to DT is that data\_1b expanded to Foreground Color and data\_0b expanded to Background Color. Any Bit 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), so the result to DT is RED "TOP" together with BLUE background:

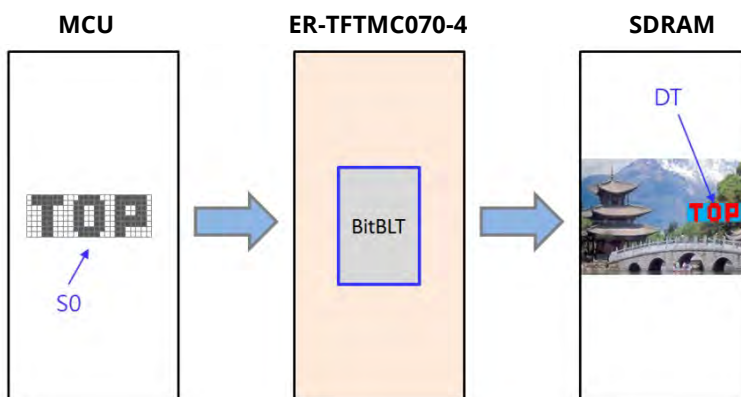


Figure 7-18: Example of MCU Write with Color Expansion



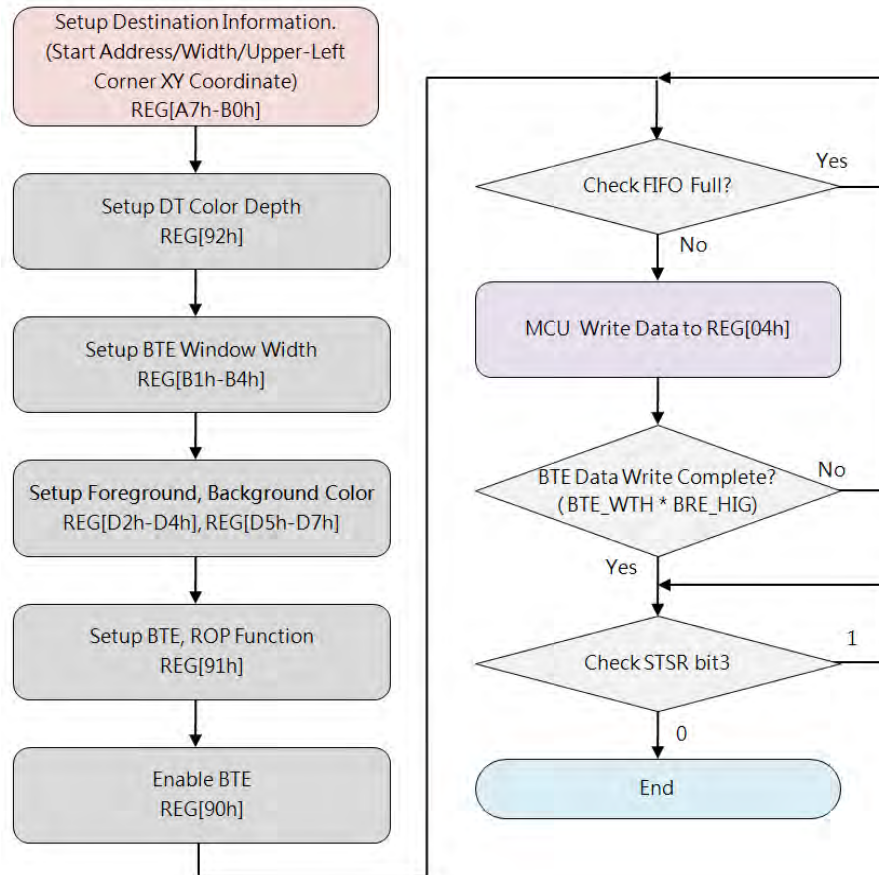


Figure 7-19: Flow Chart of MCU Write with Color Expansion

For 8-bits MCU Interface, if Foreground Color set to RED, Background Color set to KHAKI, and BTE Width set to 23, more examples please refer to Figure 7-20 (ROP = 7) and Figure 7-21 (ROP = 3).

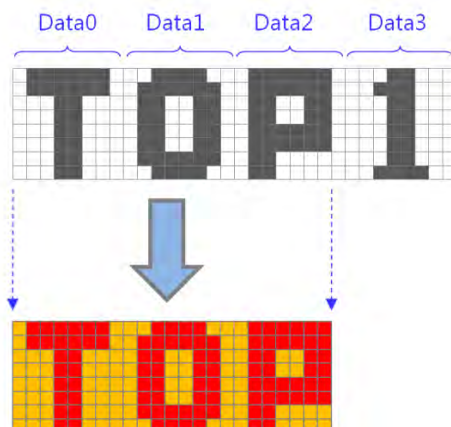


Figure 7-20: Example 1 of MCU Write with Color Expansion (ROP=7)

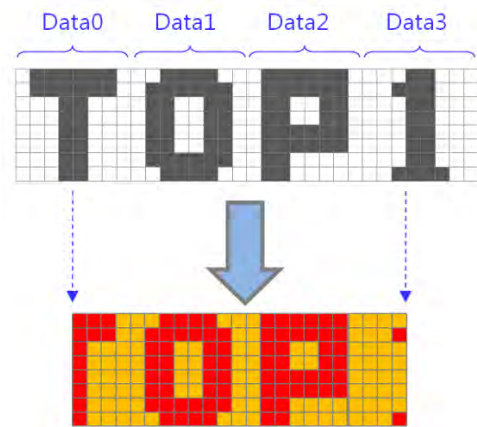


Figure 7-21: Example 2 of MCU Write with Color Expansion (ROP=3)

Note:

Sent Word\_ Numbers\_per\_Row

= [ BTE\_Width + (MCU\_Interface\_bits - Start\_bit - 1) ] / (MCU\_Interface\_bits)

(Take integer with unconditional carry)

Word\_Number\_Total = (Word\_ Numbers\_per\_Row) x BTE Height

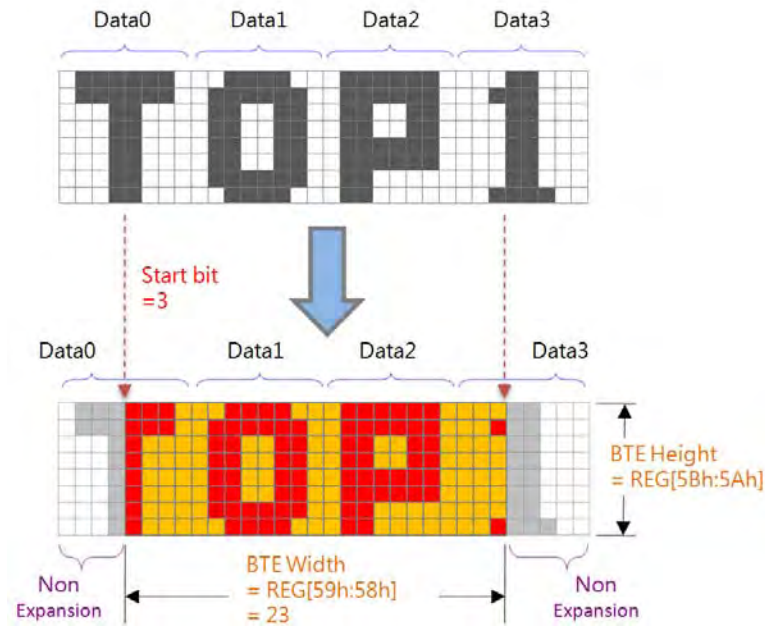


Figure 7-22: Data Format for Color Expansion

Example 1: "BTE 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: "BTE 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

#### 5.7.6.8 MCU Write with Color Expansion and Chroma key

This Operation is similar with MCU Write with Color Expansion, but the difference is that data\_0b will be discarded, BTE expands only data\_1b to Foreground Color.

Below example shows expanding data\_1b to RED (Foreground Color) and data\_0b discarded, so the result to DT is RED "TOP" with TRANSPARENT background:

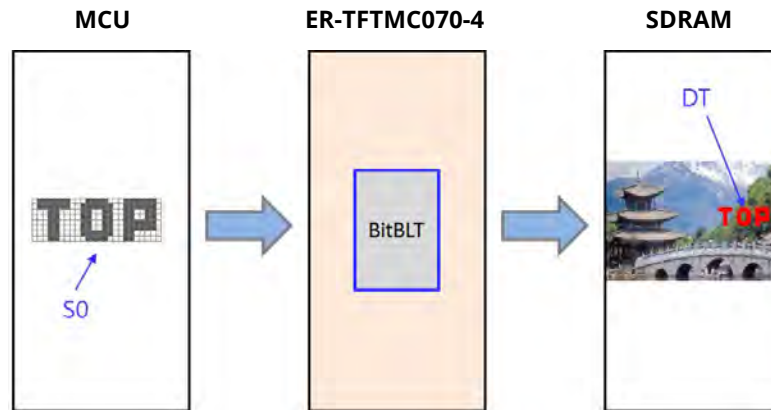


Figure 7-23: Example of MCU Write with Color Expansion and Chroma key

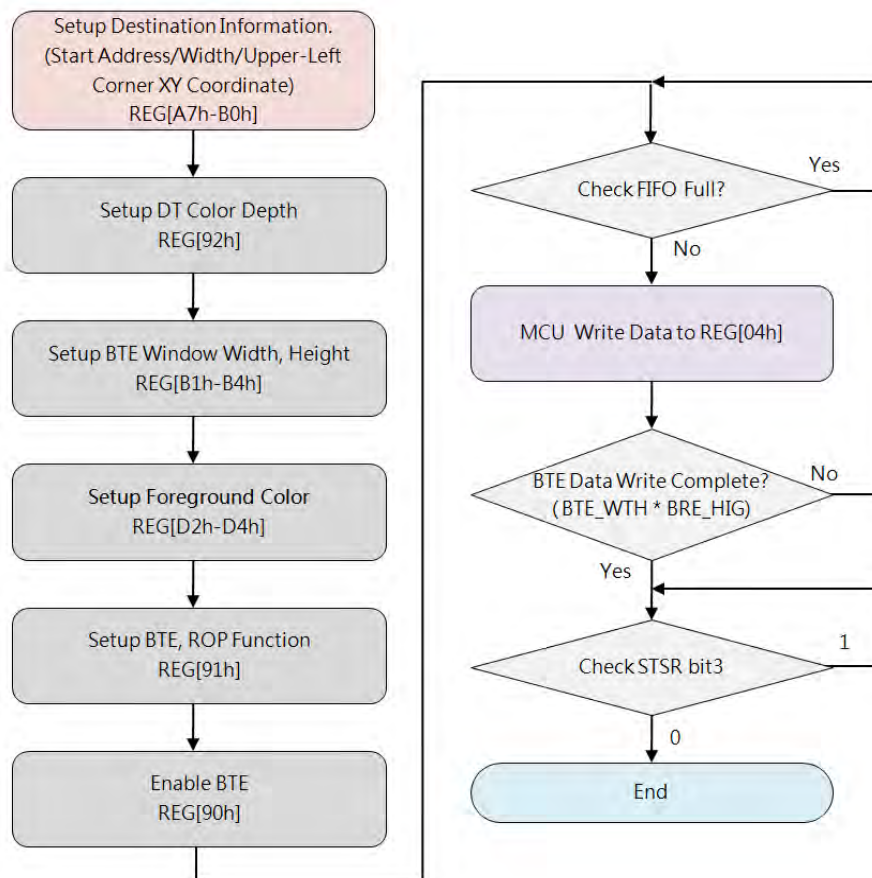


Figure 7-24: Flow Chart of MCU Write with Color Expansion and Chroma key

#### 5.7.6.9 Memory Copy with Opacity

This Operation performs blending S0 and S1 data and transferring the result to DT, two mode are available: Picture Mode and Pixel Mode.

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

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

##### Picture Mode:

Alpha\_Level = REG[B5h]

DT Data = ( S0 x Alpha\_Level ) + ( S1 x ( 1 - Alpha\_Level ) )

##### Pixel Mode 8bpp:

Alpha\_Level = S1\_Bit[7:6]

DT Data = ( S0 x Alpha\_Level ) + ( Palette\_Color\_RAM[ S1\_Bit[5:0] ] x ( 1 - Alpha\_Level ) )

##### Pixel Mode 16bpp:

Alpha\_Level = S1\_Bit[15:12]

DT Data = ( S0 x Alpha\_Level ) + ( S1\_Bit[11:0] x ( 1 - Alpha\_Level ) )

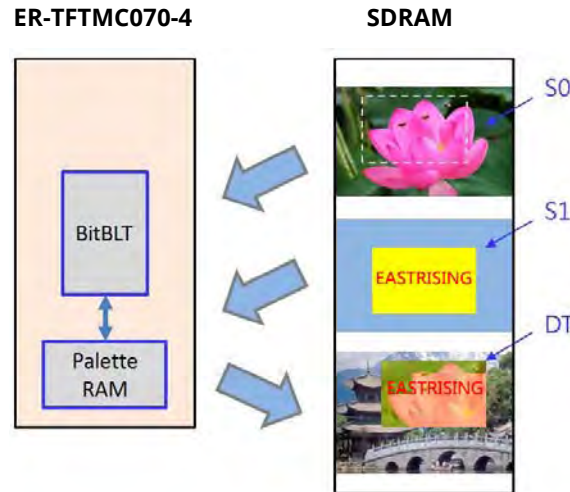


Figure 7-25: Example of Pixel Mode - 8bpp

Table 7-4: Alpha Blending of Pixel Mode - 8bpp

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

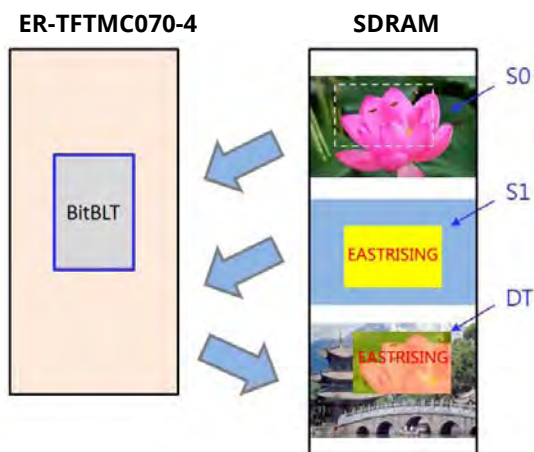


Figure 7-26: Example of Pixel Mode - 16bpp

Table 7-5: Alpha 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           |



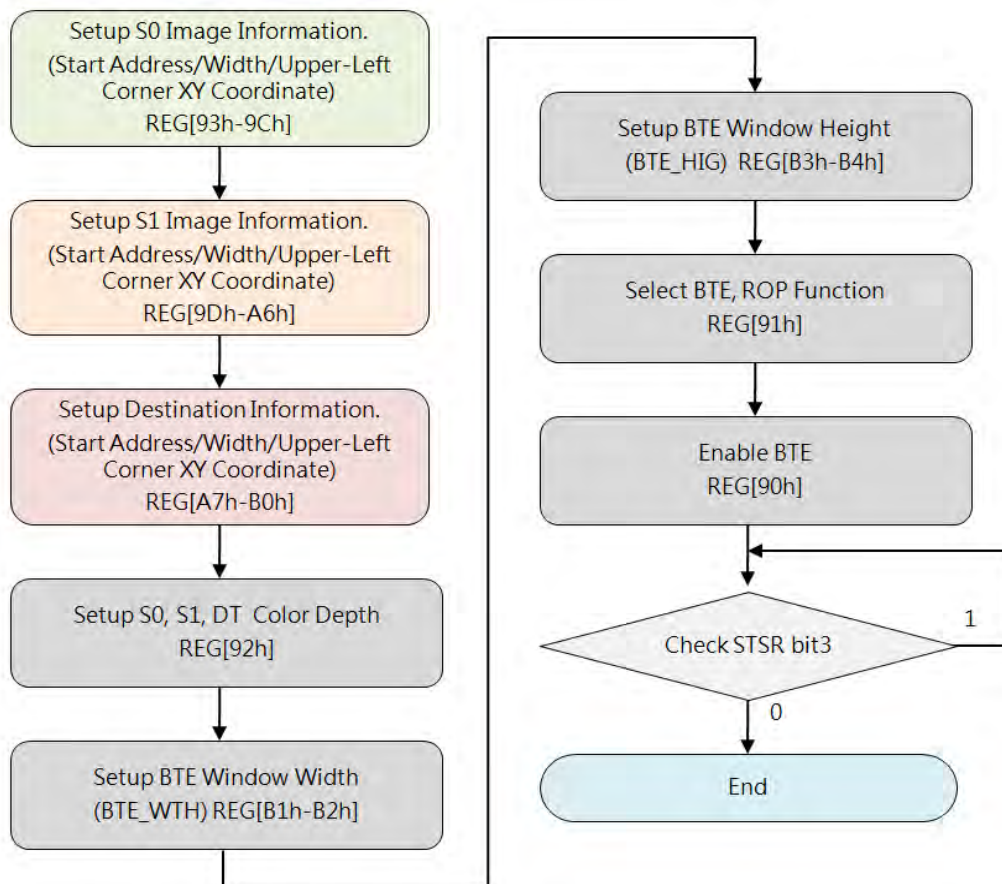


Figure 7-27: Flow Chart of Memory Copy with Opacity -Pixel Mode

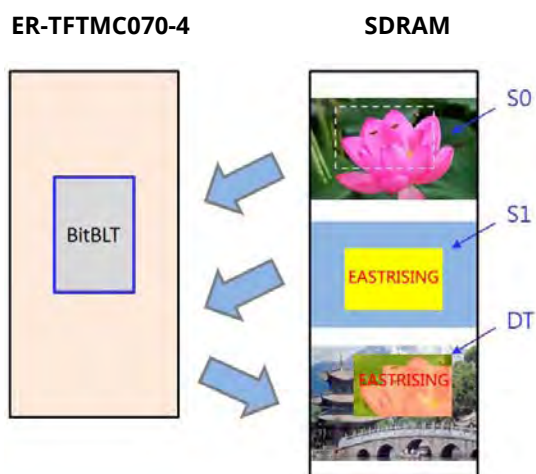


Figure 7-28: Example of Memory Copy with Opacity - Picture Mode

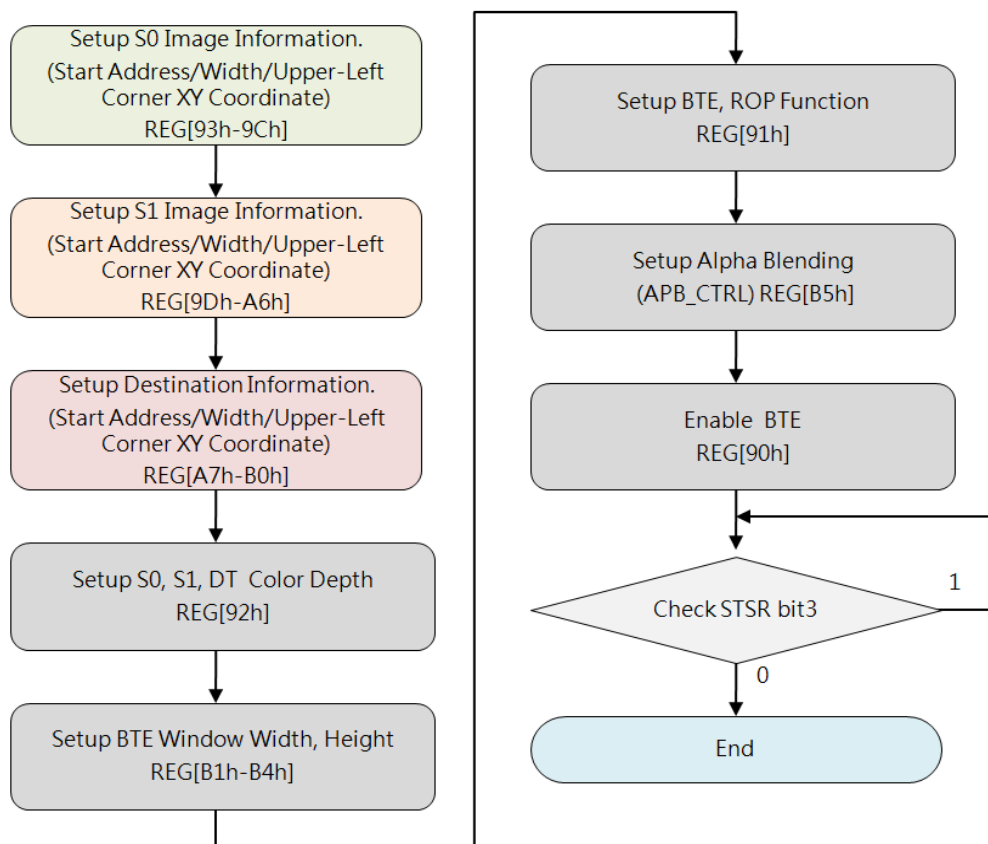


Figure 7-29: Flow Chart of Memory Copy with Opacity - Picture Mode

#### 5.7.6.10 MCU Write with Opacity

This Operation is similar with Memory Copy with Opacity, but the difference is that S0 is from MCU Write, and has same Picture Mode and Pixel Mode, for more descriptions please refer to Section 7.6.9.

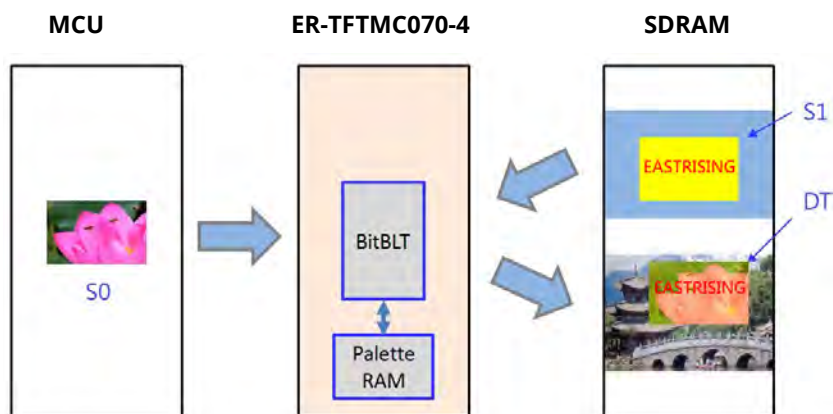


Figure 7-30: Example of MCU Write with Opacity

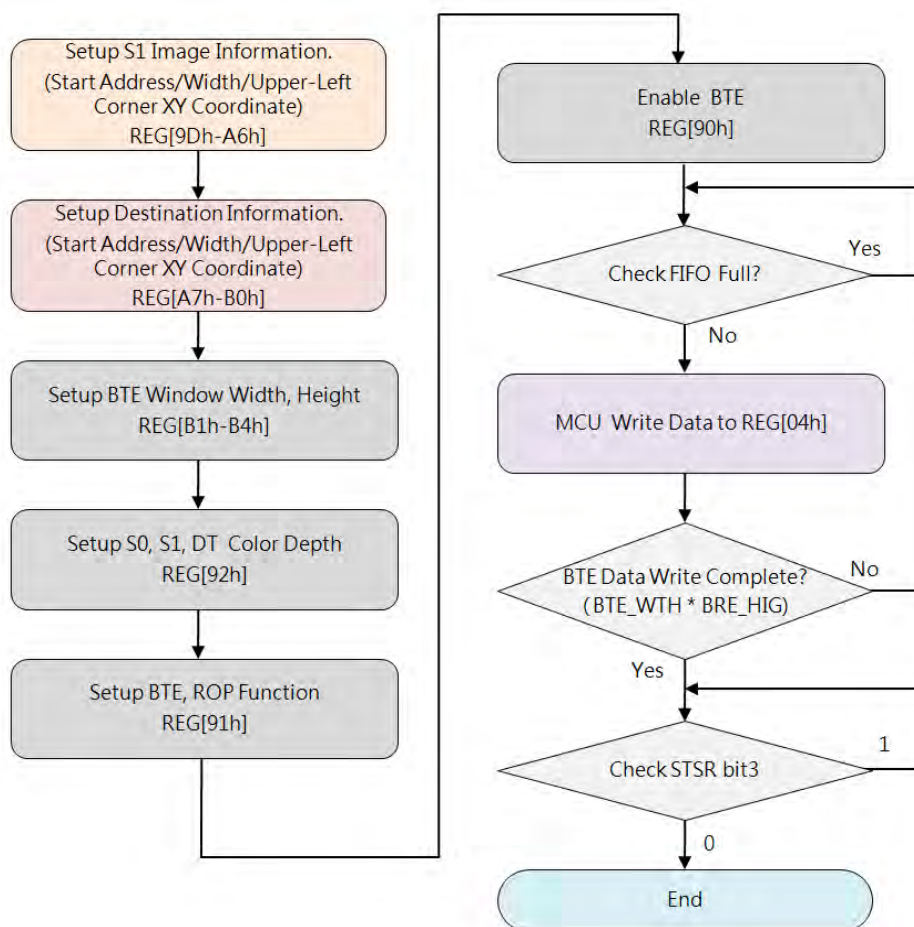


Figure 7-31: Flow Chart of MCU Write with Opacity



#### 5.7.6.11 Memory Copy with Color Expansion

This Operation performs Color Expansion for the S0 data (from Memory), it's useful to translate bit-wise monochrome data to byte-wise color data. In this operation, S0 data Color Depth (Word Width) is defined in REG[92h] Bit[6:5]. User should set needed Start Bit to REG[91h] Bit[7:4] against the corresponding Color Depth, Bit[7]~Bit[0] are available for 8-bits depth and Bit[15]~Bit[0] are available for 16-bits depth. BTE disassembles Word by Word to Bit sequences (from MSB to LSB) against the ROW Line of source image (from left to right), and sequentially expands bit by bit until the BTE Width reached ending as well. The result to DT is that data\_1b expanded to Foreground Color and data\_0b expanded to Background Color. Any Bit 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), so the result to DT is RED "TOP" together with BLUE background:

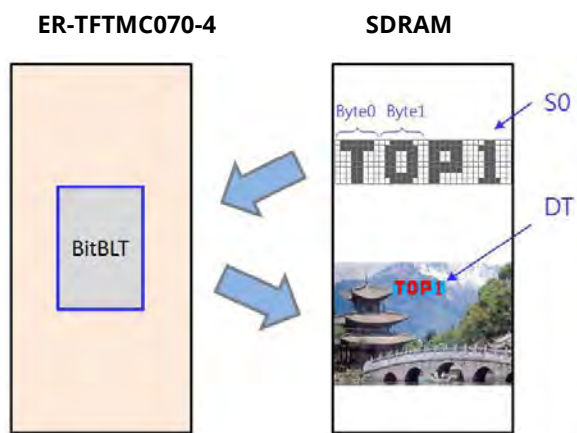


Figure 7-32: Example of Memory Copy with Color Expansion

While: S0 Color Depth = 8, Foreground Color = RED, Background Color = KHAKI, BTE window Width = 23, please refer to Figure 7-33 (ROP = 7) and Figure 7-34 (ROP = 4) as examples.

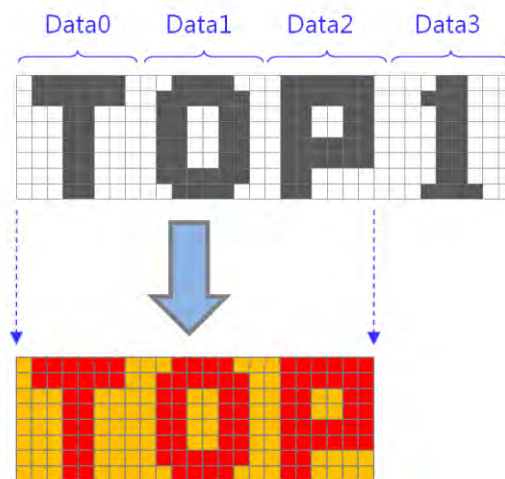


Figure 7-33: Example 1 of Memory Copy with Color Expansion (ROP=7)

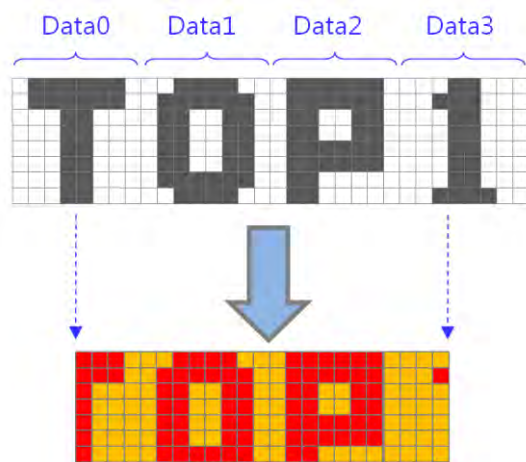


Figure 7-34: Example 2 of Memory Copy with Color Expansion (ROP=4)

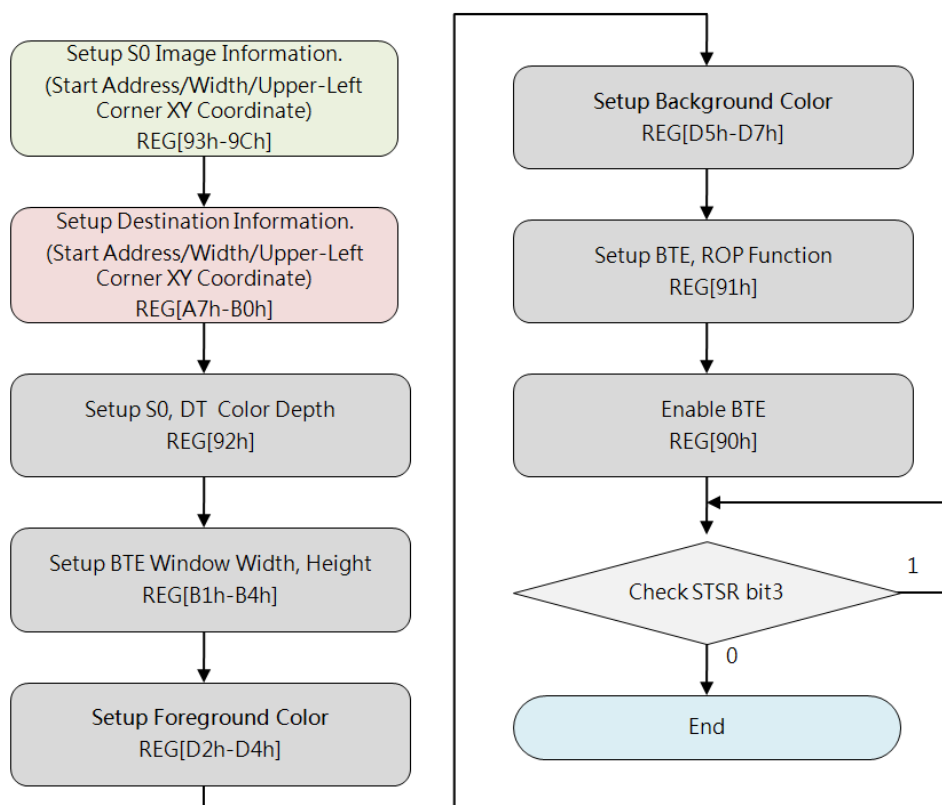


Figure 7-35: Flow Chart of Memory Copy with Color Expansion

#### 5.7.6.12 Memory Copy with Color Expansion and Chroma Key

This Operation is similar with Memory Copy with Color Expansion, but the difference is that data\_0b will be discarded, BTE expands only data\_1b to Foreground Color.

Below example shows expanding data\_1b to RED (Foreground Color) and data\_0b discarded, so the result to DT is RED "TOP" with TRANSPARENT background:

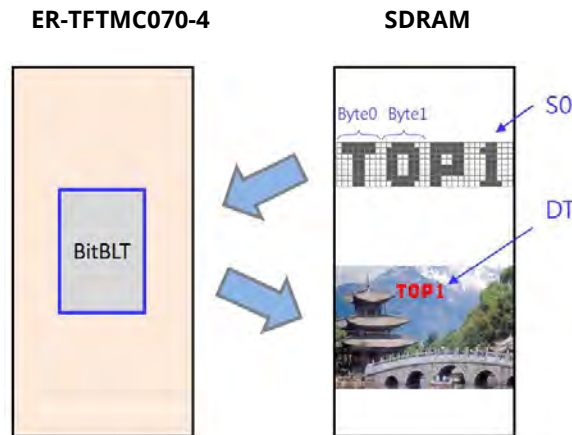


Figure 7-36: Example of Memory Copy with Color Expansion and Chroma Key

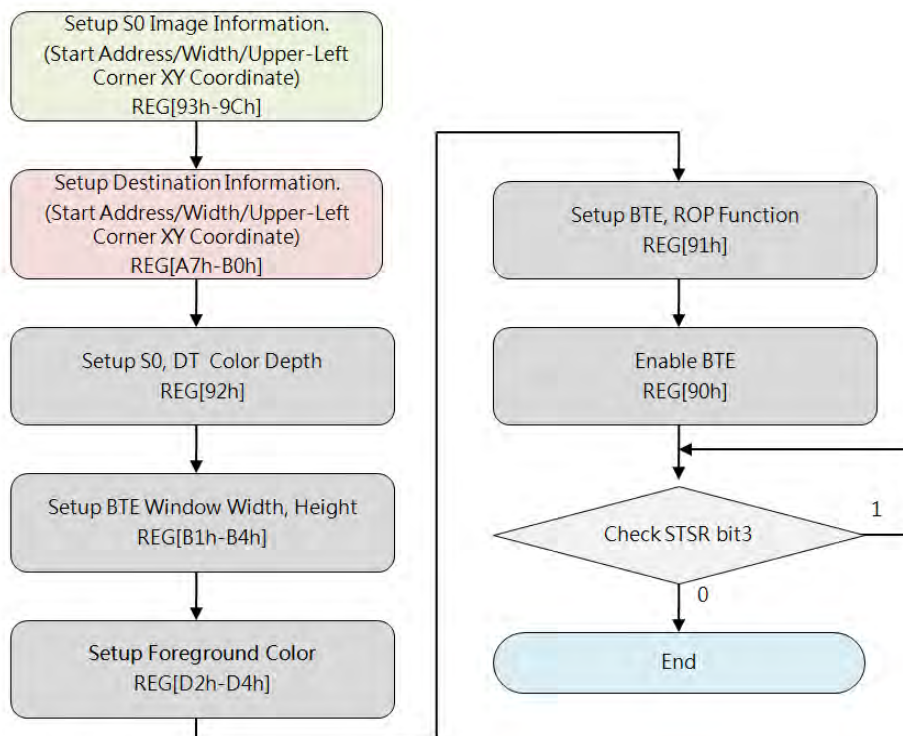


Figure 7-37: Flow Chart of Memory Copy with Color Expansion and Chroma Key

### 5.7.6.13 Solid Fill

This operation fills a specified Rectangle Area (BTE Window) of DT, with a Solid Color the data defined in the Foreground Color Register.

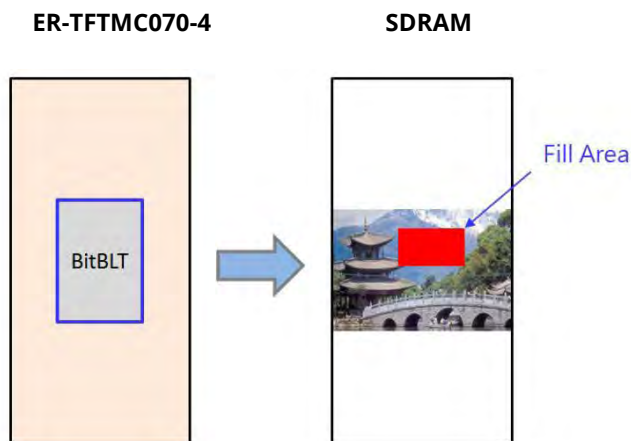


Figure 7-38: Example of Solid Fill

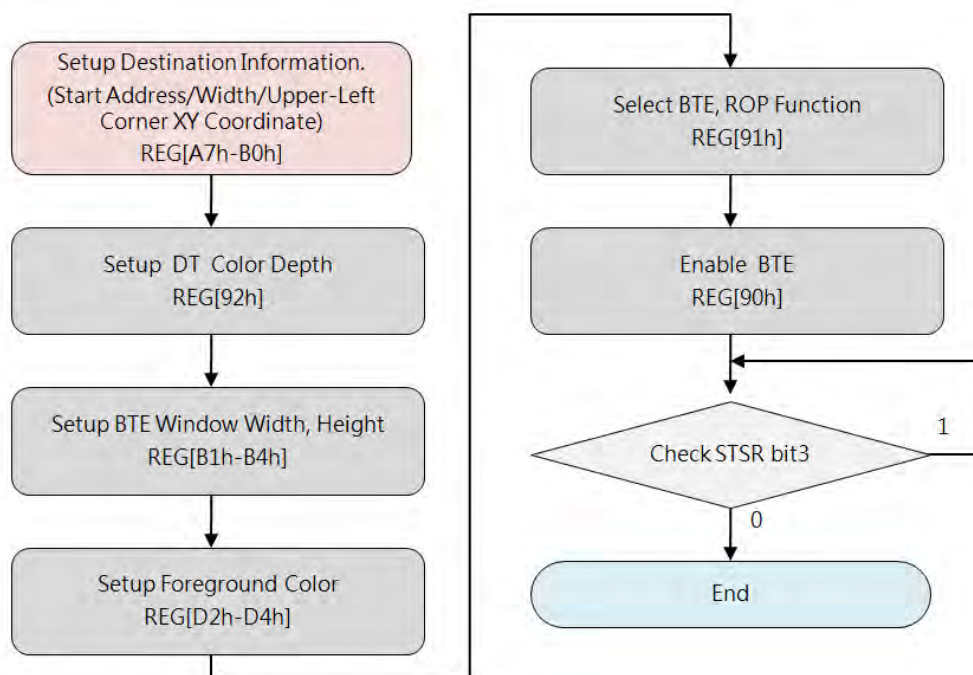


Figure 7-39: Flow Chart of Solid Fill

## 5.8 Display Text

ER-TFTMC070-4 supports 2 sources of Text (Character and Symbols):

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

ER-TFTMC070-4 internal CGROM supports four sets of embedded Characters and Symbols of ASCII code, and internal CGRAM supports user to create own Characters and Symbols sets when needed. The registers REG[CCh] ~REG[DEh] are for 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.

### 5.8.1 Internal CGROM

The ER-TFTMC070-4 built in three ASCII font types with different resolutions (character sizes): 8 x 16, 12 x 24, 16 x 32. The MCU just simply writes the font code to easily make the ASCII word display on the LCD panel. The size of the ASCII characters displayed is set by REG[CCh] [5:4]. The ASCII code is corresponding to the standard ISO/IEC 8859-1/2/4/5 coding (table 8-1 to Table 8-4 below). What type of ISO/IEC 8859 font displayed on the TFT panel is setting by REG [CCh] [1:0]. In addition, the user 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). You can refer to the following program flowchart:

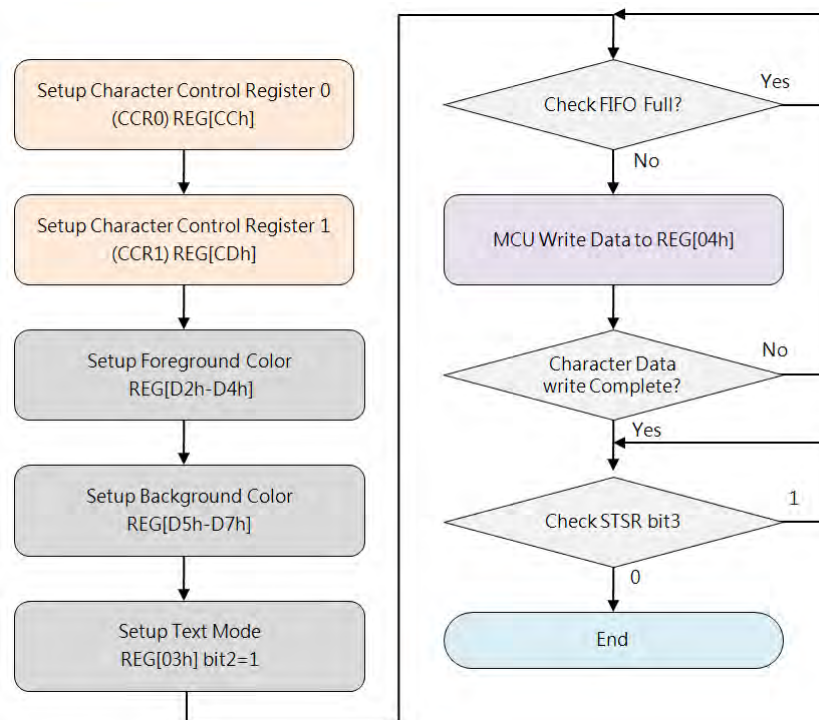


Figure 8-1A: Flow Chart of CGROM Basic Programming

Table 8-1 shows the standard character encoding of ISO/IEC 8859-1. ISO means 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 consisting of 192 characters from the Latin script in range 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 also can use ISO/IEC 8859-1. In addition, it is also commonly used in many languages outside Europe, such as Swahili, Indonesian, Malaysian and Tagalog.

In the Table 8-1, character codes 0x80~0x9F are defined by Microsoft Windows, also called CP1252 (WinLatin1).

Table 8-1: ISO/IEC 8859-1

|   | 0 | 1 | 2 | 3  | 4  | 5   | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|----|----|-----|---|---|---|---|---|---|---|---|---|---|
| 0 |   | ☺ | ☻ | ♥  | ♦  | ♣   | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ |
| 1 | ▶ | ◀ | ↑ | !! | ¶  | §   | ■ | ↑ | ↓ | → | ← | ↶ | ↷ | ↶ | ↷ | ↶ |
| 2 | ! | " | # | \$ | %  | &   | ' | ( | ) | * | + | , | - | . | / |   |
| 3 | 0 | 1 | 2 | 3  | 4  | 5   | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4 | @ | A | B | C  | D  | E   | F | G | H | I | J | K | L | M | N | O |
| 5 | P | Q | R | S  | T  | U   | V | W | X | Y | Z | [ | \ | ] | ^ | _ |
| 6 | ` | a | b | c  | d  | e   | f | g | h | i | j | k | l | m | n | o |
| 7 | p | q | r | s  | t  | u   | v | w | x | y | z | { |   | } | ~ | ␣ |
| 8 | € |   | . | f  | .. | ... | † | ‡ | § | ¶ | § | < | € | Ž |   |   |
| 9 | ' | ' | " | "  | .  | -   | - | ™ | š | > | œ | ž | Ÿ |   |   |   |
| A | ı | ç | £ | ¤  | ¥  | ı   | § | ™ | © | ª | « | ¬ | - | ® |   |   |
| B | ° | ± | ² | ³  | ´  | µ   | ¶ | · | ¸ | ¹ | º | » | ¼ | ½ | ¾ | ¿ |
| C | À | Á | Â | Ã  | Ä  | Å   | Æ | Ç | È | É | Ê | Ë | Ì | Í | Î | Ï |
| D | Ð | Ñ | Ò | Ó  | Ô  | Õ   | Ö | × | Ø | Ù | Ú | Û | Ü | Ý | Þ | ß |
| E | à | á | â | ã  | ä  | å   | æ | ç | è | é | ê | ë | ì | í | î | ï |
| F | ð | ñ | ò | ó  | ô  | õ   | ö | ÷ | ø | ù | ú | û | ü | ý | þ | ÿ |

Table 8-2: ISO/IEC 8859-2

|   | 0 | 1 | 2 | 3  | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|----|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 |   | ☺ | ☻ | ♥  | ♦ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ |
| 1 | ▶ | ◀ | ↑ | !! | ¶ | § | ■ | ↑ | ↓ | → | ← | ↶ | ↷ | ↶ | ↷ | ↶ |
| 2 | ! | " | # | \$ | % | & | ' | ( | ) | * | + | , | - | . | / |   |
| 3 | 0 | 1 | 2 | 3  | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4 | @ | A | B | C  | D | E | F | G | H | I | J | K | L | M | N | O |
| 5 | P | Q | R | S  | T | U | V | W | X | Y | Z | [ | \ | ] | ^ | _ |
| 6 | ` | a | b | c  | d | e | f | g | h | i | j | k | l | m | n | o |
| 7 | p | q | r | s  | t | u | v | w | x | y | z | { |   | } | ~ | ␣ |
| 8 |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |   |
| 9 |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |   |
| A | À | Á | Â | Ã  | Ä | Å | Æ | Ç | È | É | Ê | Ë | Ì | Í | Î | Ï |
| B | à | á | â | ã  | ä | å | æ | ç | è | é | ê | ë | ì | í | î | ï |
| C | Ā | Ą | Ć | Č  | Ĉ | Ċ | Ď | Ě | Ɔ | Ǽ | Ǽ | Ǽ | Ǽ | Ǽ | Ǽ | Ǽ |
| D | Ď | Ď | Ď | Ď  | Ď | Ď | Ď | Ď | Ď | Ď | Ď | Ď | Ď | Ď | Ď | Ď |
| E | Ě | Ě | Ě | Ě  | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě |
| F | Ě | Ě | Ě | Ě  | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě | Ě |

Table 8-2 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 8-3 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 8-3: ISO/IEC 8859-4

|   | 0 | 1 | 2 | 3  | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|----|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 |   | ☺ | ☹ | ♥  | ♦ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ |
| 1 | ▶ | ◀ | ↕ | !! | ¶ | § | ■ | ↑ | ↓ | → | ← | ↶ | ↷ | ↶ | ↷ | ↶ |
| 2 | ! | " | # | \$ | % | & | ' | ( | ) | * | + | , | - | . | / |   |
| 3 | 0 | 1 | 2 | 3  | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4 | @ | A | B | C  | D | E | F | G | H | I | J | K | L | M | N | O |
| 5 | P | Q | R | S  | T | U | V | W | X | Y | Z | [ | \ | ] | ^ | _ |
| 6 | ` | a | b | c  | d | e | f | g | h | i | j | k | l | m | n | o |
| 7 | p | q | r | s  | t | u | v | w | x | y | z | { |   | } | ~ | ␣ |
| 8 |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |   |
| 9 |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |   |
| A | À | Á | Â | Ã  | Ä | Å | Æ | Ç | È | É | Ê | Ë | Ì | Í | Î | Ï |
| B | Ĳ | ā | ŗ | ĩ  | ı | ı | ı | š | ē | ġ | t | đ | ž | ŋ |   |   |
| C | Ĳ | Ĳ | Ĳ | Ĳ  | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ |
| D | Ĳ | Ĳ | Ĳ | Ĳ  | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ |
| E | Ĳ | Ĳ | Ĳ | Ĳ  | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ |
| F | Ĳ | Ĳ | Ĳ | Ĳ  | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ | Ĳ |

Table 8-4: ISO/IEC 8859-5

|   | 0 | 1 | 2 | 3  | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|----|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 |   | ☺ | ☹ | ♥  | ♦ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ | ♠ | ♣ |
| 1 | ▶ | ◀ | ↕ | !! | ¶ | § | ■ | ↑ | ↓ | → | ← | ↶ | ↷ | ↶ | ↷ | ↶ |
| 2 | ! | " | # | \$ | % | & | ' | ( | ) | * | + | , | - | . | / |   |
| 3 | 0 | 1 | 2 | 3  | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4 | @ | A | B | C  | D | E | F | G | H | I | J | K | L | M | N | O |
| 5 | P | Q | R | S  | T | U | V | W | X | Y | Z | [ | \ | ] | ^ | _ |
| 6 | ` | a | b | c  | d | e | f | g | h | i | j | k | l | m | n | o |
| 7 | p | q | r | s  | t | u | v | w | x | y | z | { |   | } | ~ | ␣ |
| 8 |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |   |
| 9 |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |   |
| A | Ё | Ђ | Ѓ | Є  | Ѕ | І | Ї | Ј | Љ | Њ | Ћ | Ќ | Ў | Љ | Њ | Ћ |
| B | А | Б | В | Г  | Д | Е | Ж | З | И | Й | К | Л | М | Н | О | П |
| C | Р | С | Т | У  | Ф | Х | Ц | Ч | Ш | Щ | Ъ | Ы | Ь | Э | Ю | Я |
| D | а | б | в | г  | д | е | ж | з | и | й | к | л | м | н | о | п |
| E | р | с | т | у  | ф | х | ц | ч | ш | щ | ъ | ы | ь | э | ю | я |
| F | ѐ | ђ | ѓ | є  | ѕ | і | ї | ј | љ | њ | ќ | џ | џ | џ | џ | џ |

Table 8-4 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 8-1B: Internal ASCII Font for 8x16, 12x24 and 16x32

## 5.8.2 User-defined Character Graphic (UCG)

User can create and utilize UCG when needed. ER-TFTMC070-4 supports Half Width size (8x16, 12x24, 16x32 dot-matrix graphic) and Full Width size (16x16, 24x24, 32x32 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 someone UCG, just need MCU to write corresponding CODE of the UCG to ER-TFTMC070-4, ER-TFTMC070-4 can resolve the CODE (relative ADDRESS of CGRAM) to corresponding absolute ADDRESS of Memory where the UCG data is really saved, then transfer the graphic data to display Memory Buffer. Of course, user can define Foreground Color by setting the REG[D2h]~REG[D4h] and Background Color by setting the REG[D5h]~REG[D7h] in advance.

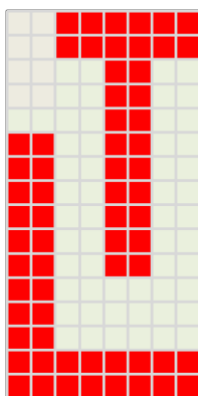
To create UCG and Initialize CGRAM, need to format data of dot-matrix graphic and allocate Memory Space at first, please refer to below sections.

### 5.8.2.1 8x16 UCG Data Format

UCG with 8x16 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 and table show the way to calculate 8x16 UCG Address in Memory, data Format and data byte Sequence:

$$\text{UCG\_ADD} = \text{CGRAM\_Start\_ADD} + (\text{UCG\_Code} \times 16)$$

Table 8-5: Data Format and Byte Sequence of 8x16 UCG





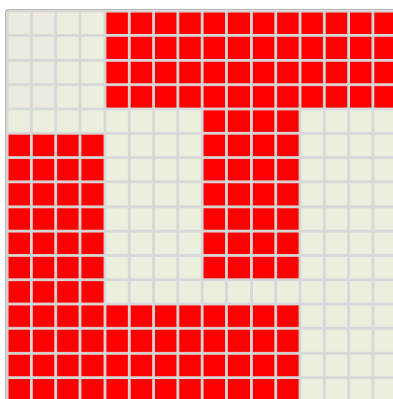
### 5.8.2.2 16x16 UCG Data Format

UCG with 16x16 size needs 32 bytes data. For example, CGRAM start address is 1000h and the first UCG encoding is 0000h, the data will be saved in 1000h~101Fh, then the second UCG encoding will be 0001h and its data will be saved in 1020h~103Fh. Below formula and table show the way to calculate 16x16 UCG Address in Memory, data Format and data byte Sequence:

$$\text{UCG\_ADD} = \text{CGRAM\_Start\_ADD} + (\text{UCG\_Code} \times 32)$$

Table 8-6: Data Format and Byte Sequenc of 16x16 UCG

| UCG Code: 0000h |             |         |             |
|-----------------|-------------|---------|-------------|
| 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 |



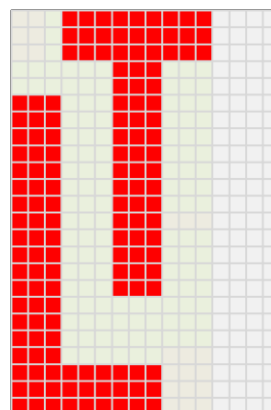
### 5.8.2.3 12x24 UCG Data Format

UCG with 12x24 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 saved in 1000h~102Fh, then the second UCG encoding will be 0001h and its data will be saved in 1030h~105Fh. Below formula and table show the way to calculate 12x24 UCG Address in Memory, data Format and data byte Sequence:

$$\text{UCG\_ADD} = \text{CGRAM\_Start\_ADD} + (\text{UCG\_Code} \times 48)$$

Table 8-7: Data Format and Byte Sequenc of 12x24 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 |



#### 5.8.2.4 24x24 UCG Data Format

UCG with 24x24 size needs 72 bytes data. For example, CGRAM start address is 1000h and the first UCG encoding is 0000h, the data will be saved in 1000h~1047h, then the second UCG encoding will be 0001h and its data will be saved in 1048h~108Fh. Below formula and table show the way to calculate 24x24 UCG Address in Memory, data Format and data byte Sequence:

$$\text{UCG\_ADD} = \text{CGRAM\_Start\_ADD} + (\text{UCG\_Code} \times 72)$$

Table 8-8: Data Format and Byte Sequence of 24x24 UCG

| UCG Code: 0000h |        |         |        |         |        |
|-----------------|--------|---------|--------|---------|--------|
| 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 |

#### 5.8.2.5 16x32 UCG Data Format

UCG with 16x32 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 and table show the way to calculate 16x32 UCG Address in Memory, data Format and data byte Sequence:

$$\text{UCG\_ADD} = \text{CGRAM\_Start\_ADD} + (\text{UCG\_Code} \times 64)$$

Table 8-9: Data Format and Byte Sequence of UGC

| 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 |

### 5.8.2.6 32x32 UCG Data Format

UCG with 32x32 size needs 128 bytes data. For example, CGRAM start address is 1000h and the first UCG encoding is 0000h, the data will be saved in 1000h~107Fh, then the second UCG encoding will be 0001h and its data will be saved in 1080h~10FFh. Below formula and table show the way to calculate 32x32 UCG Address in Memory, data Format and data byte Sequence:

$$\text{UCG\_ADD} = \text{CGRAM\_Start\_ADD} + (\text{UCG\_Code} \times 128)$$

Table 8-10: Data Format and Byte Sequenc of 32x32 UCG

| UCG Code: 0000h |         |         |         |         |         |         |         |
|-----------------|---------|---------|---------|---------|---------|---------|---------|
| 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 |

### 5.8.2.7 Initialize CGRAM from MCU

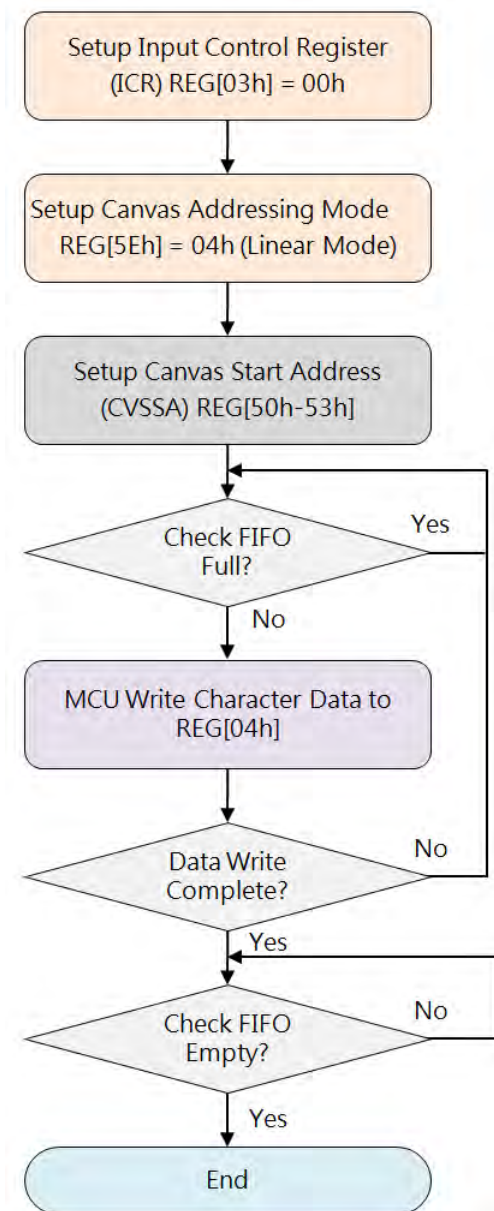


Figure 8-2: Flow Chart of CGRAM Initialization from MCU

### 5.8.2.8 Initialize CGRAM from Serial Flash

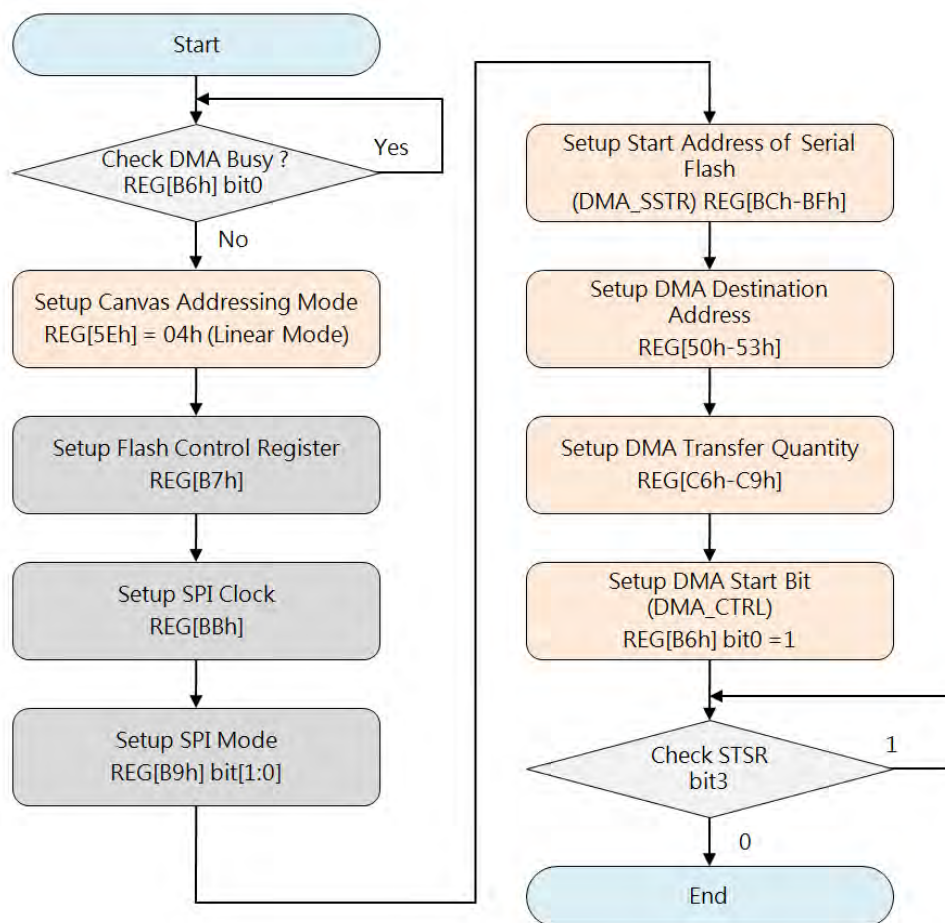
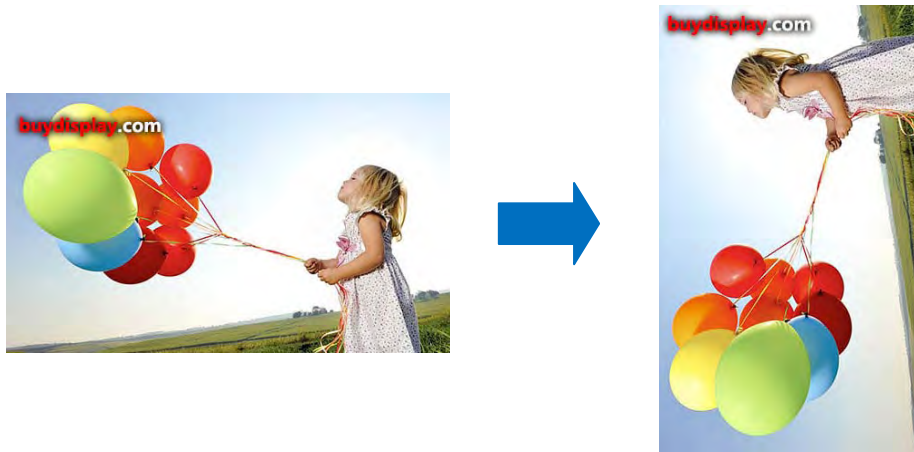


Figure 8-3: Flow Chart of CGRAM Initialization from Serial Flash

### 5.8.3 Character Rotation by 90 Degree

ER-TFTMC070-4 supports to rotate character display 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 rotate counterclockwise 90 degree and flip in vertical, as well as text direction will change to from top to bottom then from left to right. But to see correct display result, need to further change Display Scan Direction (set VDIR REG[12h] bit3=1, but please note that Text Cursor and Graphic Cursor as well as PIP are disabled automatically under this setting). Below is an example of Character Rotation by 90 degree:

Figure 8-4: Example of Character Rotation



### 5.8.4 Size Enlargement

ER-TFTMC070-4 supports linear x1, x2, x3, x4 character size enlargement for Height and/or Width, controlled by REG[CDh] bit[3:0].

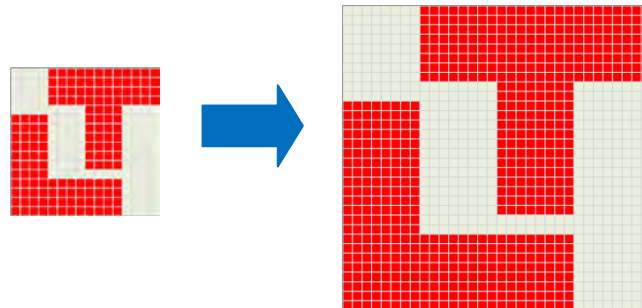


Figure 8-5: Example of Size Enlargement



### 5.8.5 Background Transparency

ER-TFTMC070-4 supports character Background transparent, controlled by REG[CDh] bit6.



Figure 8-6: Example of Background Transparency

### 5.8.6 Character Full-Alignment

ER-TFTMC070-4 supports character full-alignment that makes the character to align each other when input and display Half or Full size characters, set REG[CDh] bit7=1.

這是一款高效能TFT LCD图形加速显示芯片。其主要的功能就是协助MCU将所要显示到TFT屏的内容传递给TFT驱动器，并且提供PIP、图形加速、几何图形绘图等功能，除了提升显示效率外，还降低MCU处理图形显示所花费的时间。



這是一款高效能TFT LCD 图形加速显示芯片。其主要的功能就是协助MCU 将所要显示到TFT 屏的内容传递给TFT 驱动器，并且提供PIP、图形加速、几何图形绘图等功能，除了提升显示效率外，还降低MCU 处理图形显示所花费的时间。

Figure 8-7: Example of Character Full-Alignment

### 5.8.7 Automatic Line Feed

ER-TFTMC070-4 supports sequent text input and display, and can perform automatically Line Feed at active window boundary.

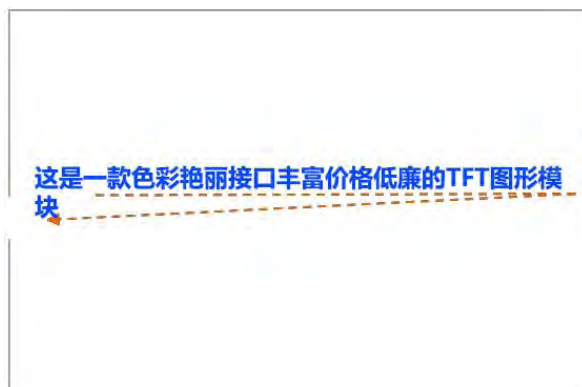


Figure 8-8: Example of Automatic Line Feed

### 5.8.8 Cursor

ER-TFTMC070-4 supports 2 types of Cursor, Graphic Cursor and Text Cursor. The Graphic Cursor is 32x32 pixel graphic with 2-bits color index which can be displayed at an user-defined position. The Text Cursor is bit-wise graphic with 32x32 as maximum size, to point text input position. Note that when Vertical Scan Direction set to "From bottom to Top" (VDIR REG[12h] bit=1), Text Cursor and Graphic Cursor as well as PIP will be disabled automatically.

#### 5.8.8.1 Text Cursor

Text Cursor has Auto-move, Blinking, and Enlargement functions, once enabled the Text Cursor appears in waiting input position, and can automatically move to next input position after current input completed. Auto-move also supports Auto Line Feed, but is dominated by Active Window, so Text Cursor must be positioned in active window and in Text Mode, moving distance and direction is same with Character input settings.

Table 8-11: Regiaters 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 X coordinate                  |
| REG[D0h]         | FLDR          | Text Line Gap Setting                                |

#### Text Cursor Blinking

By GTCCR ( REG[3Ch] ) to set Blinking enabled ( bit[0]=1) or disabled ( bit[0]=0 ), use below formula to calculate blinking interval:

$$\text{Blink\_Time (sec)} = \text{BTCR}[3Dh] \times (1/\text{Frame\_Rate})$$



Figure 8-9: 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] ). If Character enlargement is enabled, Text Cursor enlargement is also enabled automatically according same enlargement settings.

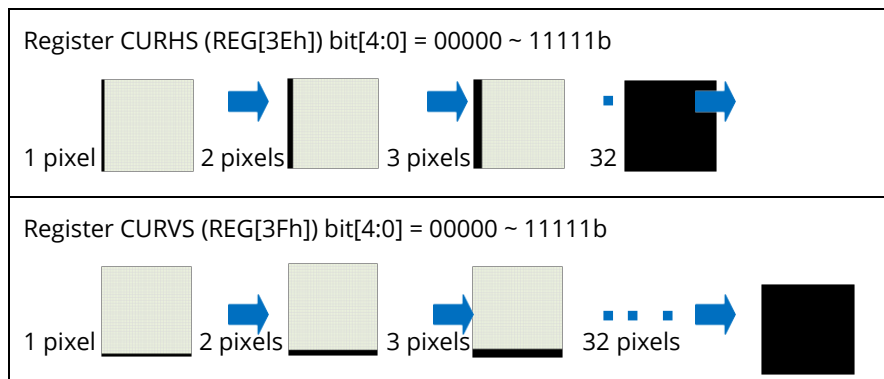


Figure 8-10: Example of Text Cursor Height and Width Settings

### 5.8.8.2 Graphic Cursor

ER-TFTMC070-4 Graphic Cursor is 32x32 pixel graphic with 2-bits Color Index, display color data is redirected to register GCC0(REG[44h]), GCC1(REG[45h]), Background Color, Inversed Background color:

Table 8-12: Graphic Color Definition

|              |                                |
|--------------|--------------------------------|
| <b>2'b00</b> | Color-0 (defined by REG[44h] ) |
| <b>2'b01</b> | Color-1 (defined by REG[45h])  |
| <b>2'b10</b> | Background Color               |
| <b>2'b11</b> | Inversed Background Color      |

To create a Graphic Cursor needs 256 bytes (32x32x2/8), Figure 8-11 shows the data format and byte sequence. ER-TFTMC070-4 supports 4 sets of graphic cursor for selection ( GTCCR REG[3Ch] bit[3:2] ). Display position of graphic cursor is controlled by register GCHP0 (REG[40h]), GCHP1(REG[41h]), GCVPO(REG[42h]) and GCVPI(REG[43h]).

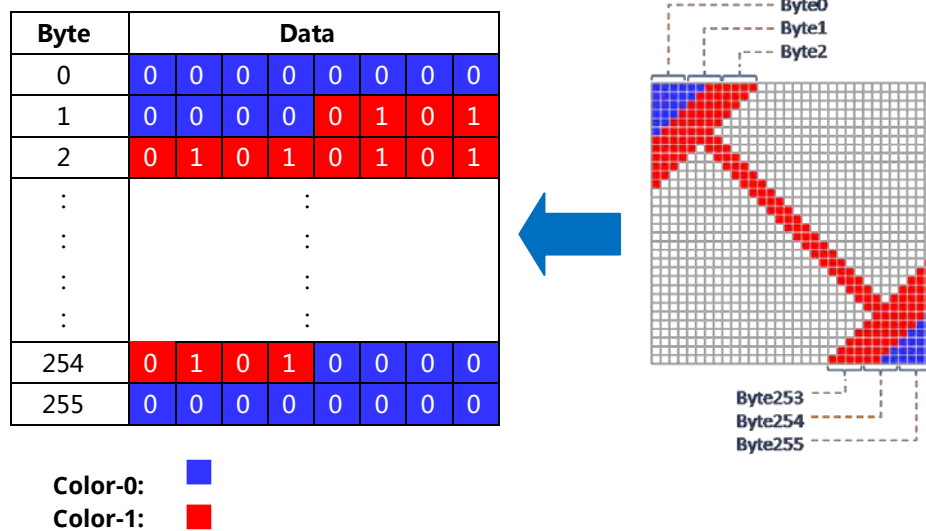


Figure 8-11: Graphic Cursor Data Format and Example

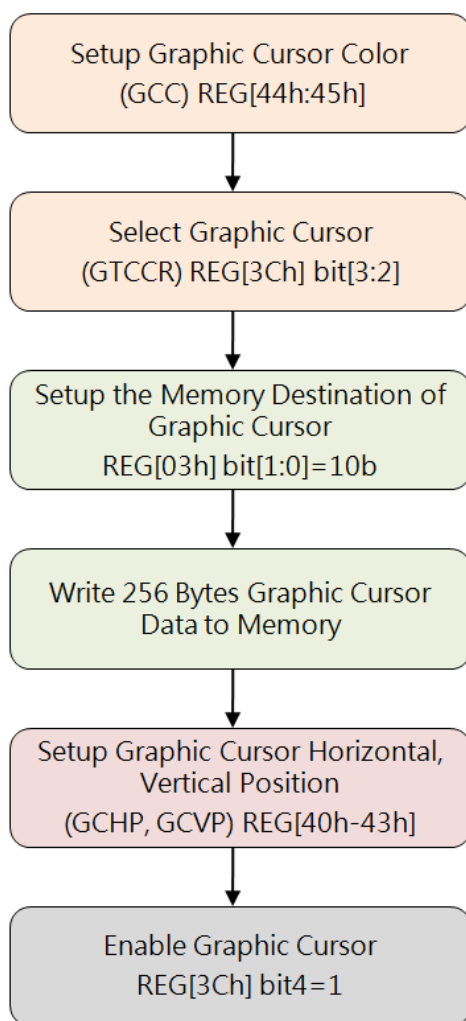


Figure 8-12: Flow Chart of Graphic Cursor Creation

## 5.9 Pulse Width Modulation (PWM)

ER-TFTMC070-4 built-in PWM function, and provide two PWM signals output: PWM0 and PWM1. The ER-TFTMC070-4 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 use 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 active. That's mean 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 to 0 then an interrupt will generated. The PWM0's state will back to the original 0, and also automatically reload TCNTB0 value. The above procedure is represent a complete PWM cycle.

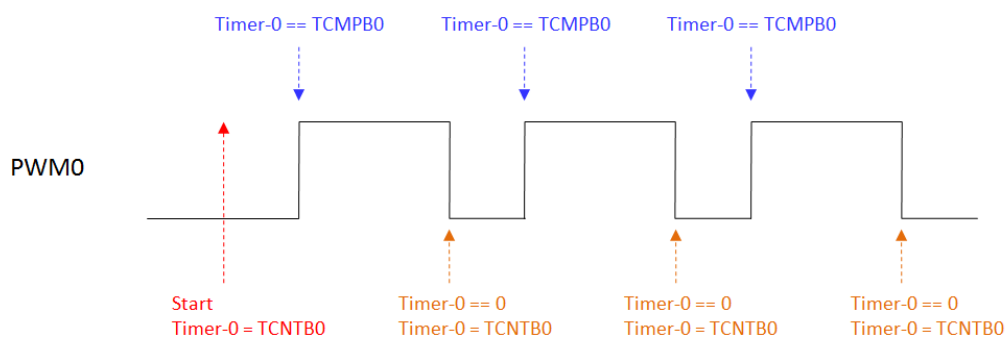


Figure 9-1: PWM0 Output Wave Form

Base on the above waveform and description, actually the duty cycle of PWM0 is determined by comparison registers (TCMPB0, REG[88h-89h]). For example, you want to generate a voltage of approximate DC potential by PWM0. When the PWM0 is initially set to 0, if TCMPB0 value is set to be large then we can get higher equivalent voltage on PWM0. Conversely, when the PWM0 is initially set to 1, then TCMPB0 value has to set smaller for getting higher equivalent voltage on PWM0.

**Note:** The automatic reload feature of PWM0 (REG[86h] bit1) must be turned on, 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.

### 5.9.1 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]):

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

The clock source of Timer is determined by the respective frequency registers (REG[85h]). The Timer Divisor provides four options: 1, 1/2, 1/4, 1/8 for the Timer's clock. For example the REG[85h] Bit[5:4] = 10b, then the Timer-0 count Clock = System\_clock/4. Please refer to the Register Description of next chapter for REG[84H] and REG[85h].

### 5.9.2 PWM Output

The output of PWM can also be set to a fixed high or low level. For PWM0, first to turn off the automatic overload feature (REG[86h] bit1 = 0), and stop Timer-0 count (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 shared output pins that can be used for other purposes, please refer to the following description of Register REG[85h] bit[3:0].

Table 9-1: REG[85h] Description

| Bit | Description  |
|-----|--|
| 3-2 | <b>PWM[1] Function Control</b><br>0xb: PWM[1] output system error flag (Scan FIFO POP error or Memory access out of range)<br>10b: PWM[1] output PWM timer 1 event or invert of PWM timer 0<br>11b: PWM[1] output Oscillator Clock |
| 1-0 | <b>PWM[0] Function Control</b><br>0xb: PWM[0] becomes GPIOC[7]<br>10b: PWM[0] output PWM Timer 0<br>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 is followed by the setting and control of PWM0, except that it is a PWM0 reverse output state:

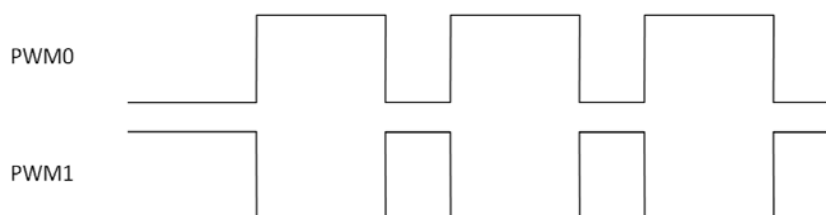


Figure 9-2: The Complementary Outputs of PWM0 and PWM1

In some applications of using Complementary Outputs of PWM0 and PWM1 , PWM0 and PWM1 change state at the same time will cause excessive current. The ER-TFTMC070-4 provides a dead-zone timing control to stagger the output state of PWM0 and PWM1 transfer at same time. The dead-zone length of PWM is set by register REG[87h] (DZ\_LENGTH):

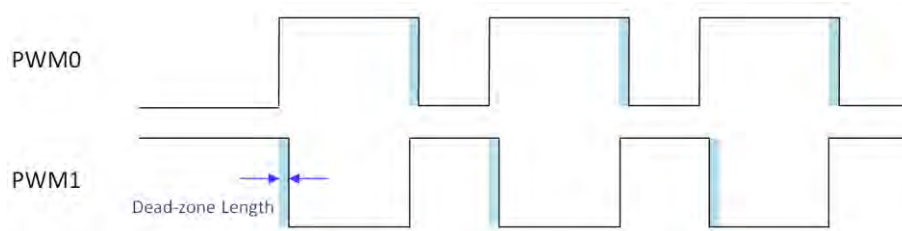


Figure 9-3: dead-zone of PWM0 and PWM1



## 5.10 I2C Master

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

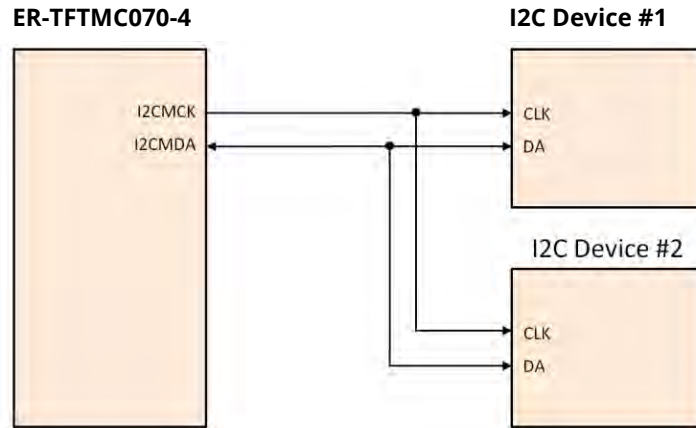


Figure 10-1: Application Circuit of ER-TFTMC070-4 I2C Master

ER-TFTMC070-4 support 100Kbps and 400Kbps transfer rate for I2C bus. The formula of I2C Master clock(I2CMCK) is as the following:

$$I2CMCK = CCLK / [5 \times (\text{Prescaler} + 2)]$$

For example, if I2CMCK is 100 KHz and CCLK is 50 MHz, pre-scalar must be set to 98. Data transfer between I2C Master and Slave is synchronously 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 processing during the high period of I2CMCK so that the I2CMDA could be change only during the low period of I2CMCK and must be held stable during the high period of I2CMCK.

The standard Communication Protocol of I2C consists of four parts:

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

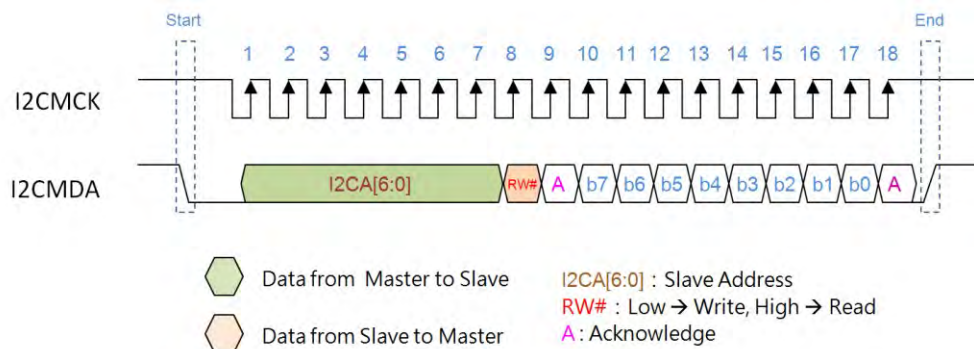


Figure 10-2: I2C Communication Protocol

Example 1. Write 1 Byte data to Slave Device:

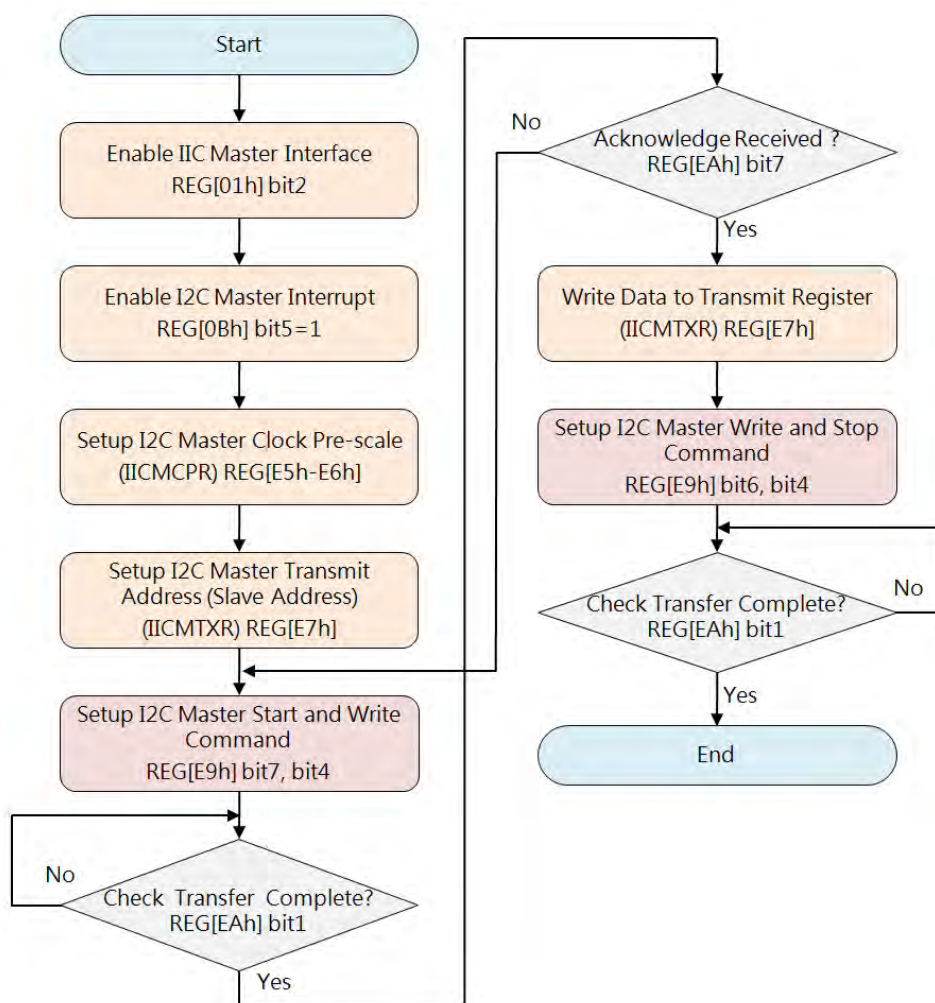


Figure 10-3: Write Data to Slave

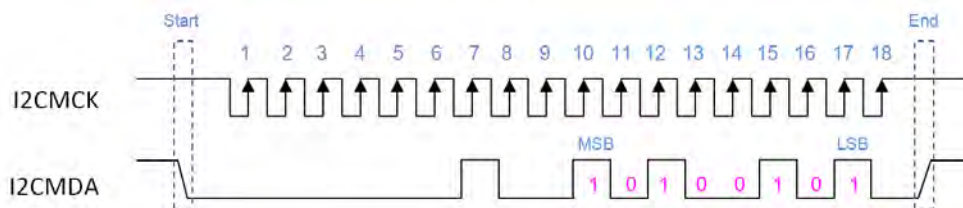


Figure 10-4: Write Data "A5" to Slave (Address=0x01)

Example 2. Read 1 Byte Data from Slave Device:.

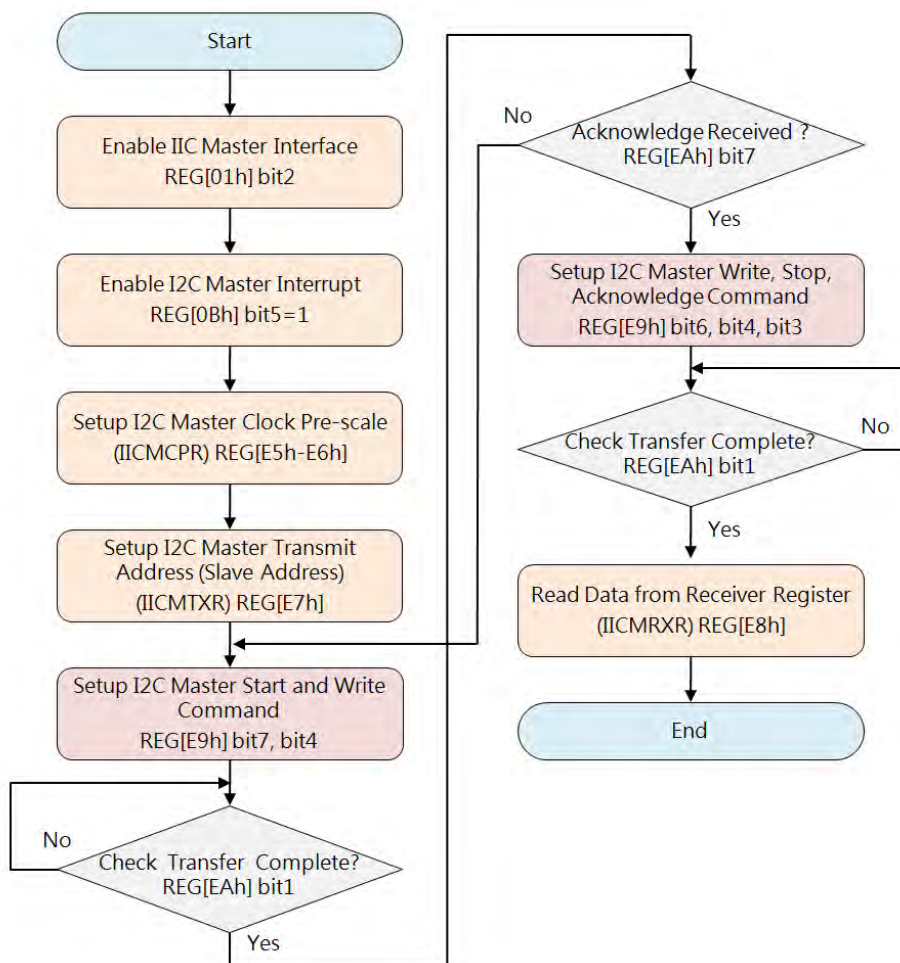


Figure 10-5: Read Data from Slave

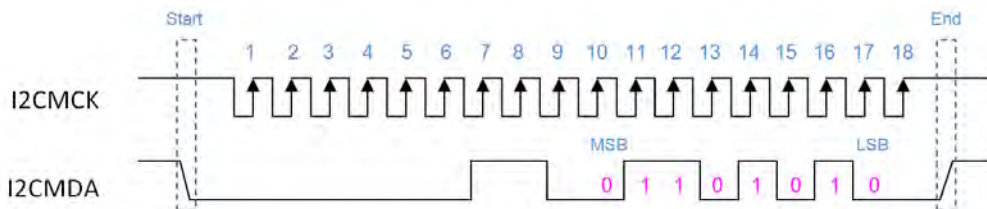


Figure 10-6: Read Data "6A" from Slave (Address=0x01)

Note: The I2CMCK and I2CMDA are multiplex pins that share with KI[0] and KO[0].

## 5.11 Keypad-Scan

ER-TFTMC070-4 provides a set of 5x5 keyboard scanning circuits. In addition to saving MCU IO interface, it can also reduce the hardware and software loading of MCU. As long as the setting of several registers, Host can make MCU easy to get the keyboard data. The following diagram is the basic keyboard application circuit.

**ER-TFTMC070-4**

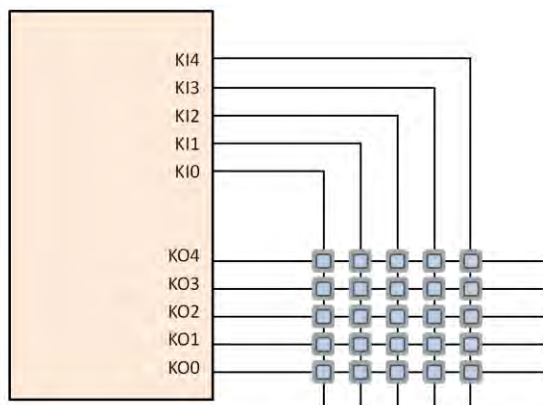


Figure 11-1: Application Circuit of Keypad-scan

### 5.11.1 Keypad-scan Operation

The embedded Keypad-scan features of ER-TFTMC070-4 are as below:

- Support up to 5x5 Keypad matrix
- Programmable setting of sampling times and scan frequency of Keypad-scan
- Adjustable long Key-press timing
- Support Multi Key-press
- Support Key-stroke wakeup function

The Keypad State Register KSCR is used to control the function of Keypad-scan, such as sampling time, sampling frequency, and enabling long keystrokes. If the key is pressed, The Host can also get the interrupt to know whether any keypad pressed. The Register KSCR2 bit[1:0] record the number of keystrokes pressed, then Host can get the corresponding key-code of pressed keys by reading register KSDR.

Table 11-1: Key-code of Normal-press Key

|     | K10 | K11 | K12 | K13 | K14 |
|-----|-----|-----|-----|-----|-----|
| KO0 | 00h | 01h | 02h | 03h | 04h |
| KO1 | 10h | 11h | 12h | 13h | 14h |
| KO2 | 20h | 21h | 22h | 23h | 24h |
| KO3 | 30h | 31h | 32h | 33h | 34h |
| KO4 | 40h | 41h | 42h | 43h | 44h |

Table 11-1 and Table 11-2 are show the corresponding Key-code of Normal-press key and Long-press key. The Key-code will be stored in the Registers KSDR0 ~ KSDR2 when there is any key pressed. If user press the Keypad button for a long time, then ER-TFTMC070-4 will corresponding different key-code which saved in Registers (KSDR0 ~ KSDR2).

Table 11-2: Key-code of Long-press Key

|     | KI0 | KI1 | KI2 | KI3 | KI4 |
|-----|-----|-----|-----|-----|-----|
| KO0 | 80h | 81h | 82h | 83h | 84h |
| KO1 | 90h | 91h | 92h | 93h | 94h |
| KO2 | A0h | A1h | A2h | A3h | A4h |
| KO3 | B0h | B1h | B2h | B3h | B4h |
| KO4 | C0h | C1h | C2h | C3h | C4h |

When multiple keystrokes are pressed, up to three key-code will be present in Register KSDR0, KSDR1 and KSDR2.

**Note:** The key-codes which stored in Register (KSDR0 ~ KSDR2) is related to the location of key keys or keys, but not to the sequence of keys. For example, if press the key code 0x34, 0x00, 0x22 at the same time, then the KSDR0 ~ KSDR2 may stored the key-code as follows:

KSDR0 = 0x00    KSDR1 = 0x22    KSDR2 = 0x34

The following is the related Register of Keypad-scan functions:

Table 11-3: The Related Register of Keypad-scan

| Register Address | Register Name | Description                                 |
|------------------|---------------|---|
| REG[FBh]         | KSCR1         | bit6: Long-press Key Enable                 |
|                  |               | bit[5:4]: Short Key De-bounce Times         |
|                  |               | bit[2:0]: Keypad Row Scan Time              |
| REG[FCh]         | KSCR2         | bit7: Keypad-Scan Wakeup Enable             |
|                  |               | bit[4:2]: Long-press Key Recognition Factor |
|                  |               | bit[1:0]: Numbers of Key Pressed            |
| REG[FDh ~ FFh]   | KSDR0 ~ 2     | Key Strobe Data1 ~ Data3                    |
| REG[01h]         | CCR           | bit5: Keypad-Scan Enable/Disable            |
| REG[0Bh]         | INTEN         | bit3: Keypad-Scan Interrupt Enable          |
| REG[0Ch]         | INTF          | bit3: Keypad-Scan Interrupt Flag            |

The Host can use two ways to check if there is any keypad pressed:

- Host check the Keypad-scan Status Register
- Host check the Interrupt Signal

If the Keypad Interrupt Enable (REG[0Bh] bit3=1), then interrupt signal will active when keypad is pressed. When interrupts occur, the Keypad-scan interrupt state flag (REG[0Bh] bit3) will be 1. So user must clear this Interrupt status flag after reading the key-code, otherwise there will be no next interruptions.

In addition, ER-TFTMC070-4 supports "key-press wakeup" in power-saving mode. After the register are setup complete, any keystroke triggers can awaken the ER-TFTMC070-4 from sleep mode. In order to recognize the Wake-up event, Host can use software to polling ER-TFTMC070-4 interrupt status bit. As follows Figure 11-2 flowchart shown.

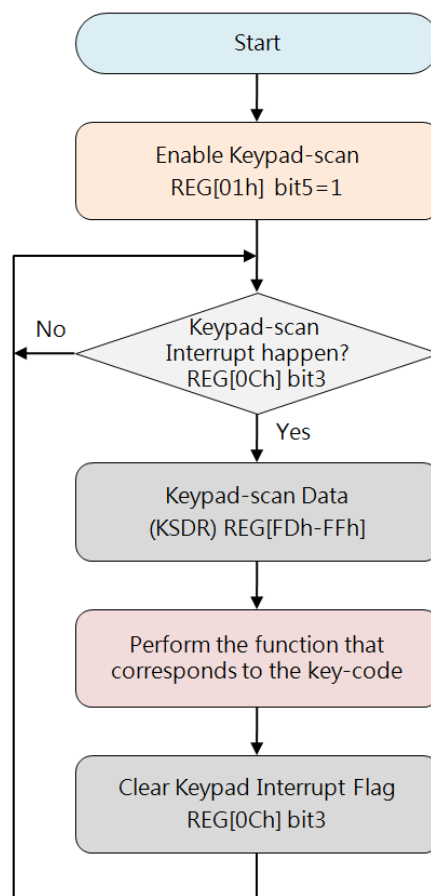


Figure 11-2: Flowchart to check Key-pad Interrupt ( S/W Polling Mode)



The Host can also be notified by hardware interrupt signal INT#, as shown in the Figure 11-3 flowchart.

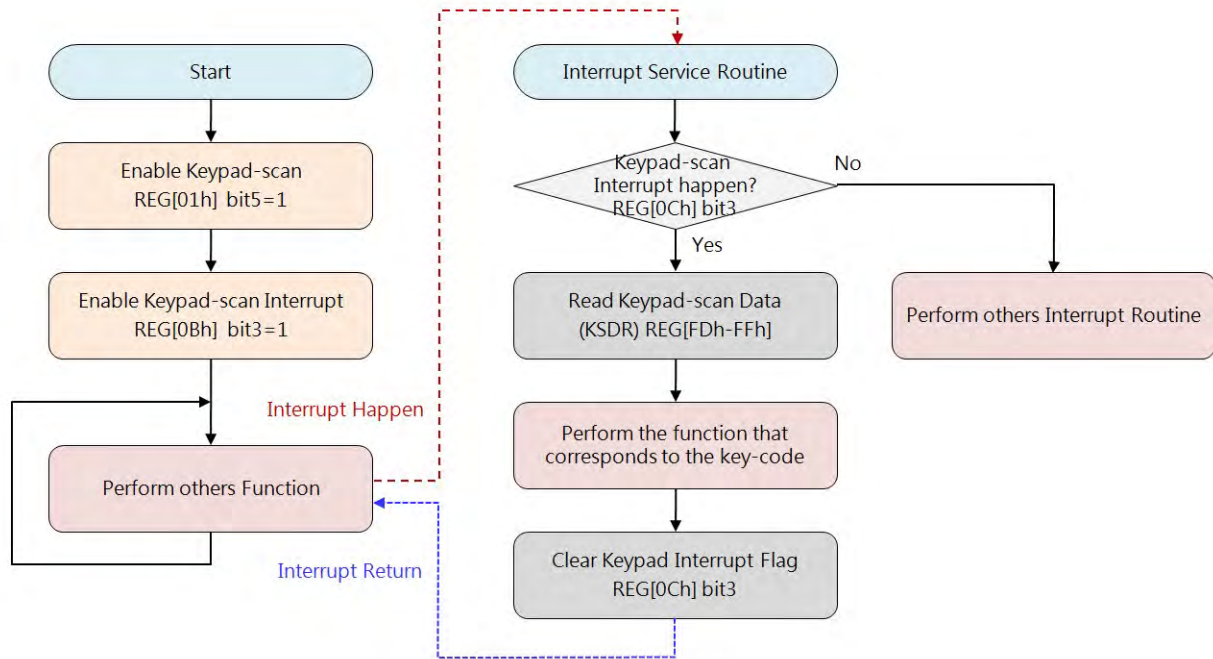


Figure 11-3: Flowchart to check Key-pad Interrupt (H/W Interrupt Mode)

## 5.12 GPIO Interface

ER-TFTMC070-4 provides many GPIO signals that can be extended as MCU I/O interfaces. Usually these GPIO signals are shared with other control signals. Please refer to Table 12-1 as below. And note these GPIO signals are available only when their mapping control signals were disabled.

Table 12-1: GPIO Port vs. Shared Signals

| GPIO Port  | Shared Signals                |
|------------|-------------------------------|
| GPIOA[7:0] | DB[15:8]                      |
| GPIOB[4],  | KI[0]                         |
| GPOB[4]    | KO[0]                         |
| GPIOB[3:0] | {A0, WR#, RD#, CS#}           |
| GPIOC[7]   | PWM[0]                        |
| IOD[7:0]   | PD[18, 2, 17, 16, 9, 8, 1, 0] |

Table 12-2 is a supported GPIO signals list of ER-TFTMC070-4. These GPIO signals are set to output or input, as well as to output data or read input data, are controlled by REG[F0h-F6h]. Please refer to Chapter 14 - Register Description.

Table 12-2: The Supported GPIO Port for ER-TFTMC070-4

| GPIO Number | GPIO Port   |
|-------------|---|
| 23          | GPIOA[7:0]<br>GPIOB[4], GPOB[4], GPIOB[3:0]<br>GPIOC[7], GPIOD[7:0] |



## 5.13 Power Management

ER-TFTMC070-4 has a total of four power modes of operation, based on the power consumption from high to Low: Normal mode, Standby mode, Suspend mode and sleep mode. These four working modes are set by register REG[DFh]. The following are four working modes of clock action comparison tables:

Table 13-1: Power Management vs. Clock Active

| Item        | Normal Mode | Standby Mode |            | Suspend Mode |            | Sleep Mode   |            |
|-------------|-------------|--------------|------------|--------------|------------|--------------|------------|
|             | PLL Enable  | Parallel MCU | Serial MCU | Parallel MCU | Serial MCU | Parallel MCU | Serial MCU |
| <b>MCLK</b> | MPLL Clock  | MPLL Clock   | MPLL Clock | OSC          | OSC        | Stop         | Stop       |
| <b>CCLK</b> | CPLL Clock  | OSC          | OSC        | Stop         | OSC        | Stop         | OSC        |
| <b>PCLK</b> | PPLL Clock  | Stop         | Stop       | Stop         | Stop       | Stop         | Stop       |
| <b>CPLL</b> | ON          | ON           | ON         | OFF          | OFF        | OFF          | OFF        |
| <b>MPLL</b> | ON          | ON           | ON         | OFF          | OFF        | OFF          | OFF        |
| <b>PPLL</b> | ON          | ON           | ON         | OFF          | OFF        | OFF          | OFF        |

Note:

1. When ER-TFTMC070-4 enters the power saving mode, the LCD interface will not output the signal. Therefore, before entering the power-saving mode, Host need to do the LCD module display off or power off to avoid LCD polarization damage.
2. The "OSC" means external X<sup>tal</sup> oscillator.

### 5.13.1 Normal Mode

In this mode, three of the internal PLL functions are working normally. That is, Host setup Registers to control three PLL respectively produce CCLK (Core Clock), MCLK (Display RAM Clock) and PCLK (LCD scan Clock). Because the PLL need some time to work stable, therefore, Host must first through the register 01h bit7 to know whether the PLL frequency is in a stable state.

### 5.13.2 Standby Mode

If setup REG[DFh] bit[1:0] to 01b, ER-TFTMC070-4 will enter Standby mode. The System Clock (CCLK) and LCD-Scan Clock (PCLK) will stop. The Display RAM Clock is active and provides by MCLK.

Enter Standby Mode:

- ◆ Select Standby Mode (REG[DFh] bit[1:0] = 01b)
- ◆ Setup REG[DFh] bit7 to 1 (Enter Standby Mode)
- ◆ Host may check the bit1 of Status Register (STSR), and wait this bit become to 1 to make sure ER-TFTMC070-4 enter Standby mode.

Back to Normal Mode:

- ◆ Setup REG[DFh] bit7 to 0 (Quit from Standby Mode)
- ◆ The Host may check the bit1 of Status Register (STSR), and wait this bit become to 0 to make sure ER-TFTMC070-4 back to Normal Mode.

### 5.13.3 Suspend Mode

If setup REG[DFh] bit[1:0] to 10b, ER-TFTMC070-4 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 provides by OSC Clock..

#### Enter Suspend Mode:

- ◆ According to OSC frequency to setup appropriate Display RAM(SDRAM) Refresh Clock
- ◆ Select Suspend Mode( REG[DFh] bit[1:0] = 10b)
- ◆ Setup REG[DFh] bit7 to 1 (Enter Suspend Mode)
- ◆ The Host may check the bit1 of Status Register (STSR), and wait this bit become to 1 to make sure ER-TFTMC070-4 enter Suspend Mode.

#### Back to Normal Mode:

- ◆ Setup REG[DFh] bit7 to 0 (Quit from Suspend Mode)
- ◆ The Host may check the bit1 of Status Register (STSR), and wait this bit become to 0 to make sure ER-TFTMC070-4 back to Normal Mode.

### 5.13.4 Sleep Mode

If setup REG[DFh] bit[1:0] to 11b, ER-TFTMC070-4 will enter Sleep Mode. And all of Clocks and PLL will be stop operate.

#### Enter Sleep Mode:

- ◆ Select Sleep Mode (REG[DFh] bit[1:0] = 11b)
- ◆ Setup REG[DFh] bit7 to 1 (Enter Sleep Mode)
- ◆ If REG[E0h] bit7 is 0, the Display RAM will enter Power Down mode that before ER-TFTMC070-4 enter Sleep mode. If REG[E0h] bit7 is 1, the Display RAM will enter Refreshmode.
- ◆ If Host interface is Parallel Mode, then ER-TFTMC070-4 will stop OSC. If Host interface is Serial Mode, then OSC will not be stop.
- ◆ The Host may check the bit1 of Status Register (STSR), and wait this bit become to 1 to make sure ER-TFTMC070-4 enter Sleep Mode.

#### Back to Normal Mode:

- ◆ Setup REG[DFh] bit7 to 0 (Quit from Sleep Mode)
- ◆ If OSC was be stopped in Sleep mode, the Host have to enable OSC.
- ◆ The Host may check the bit1 of Status Register (STSR), and wait this bit become to 0 to make sure ER-TFTMC070-4 back to Normal Mode.

## 5.14 Register Description

ER-TFTMC070-4 provides a compatible 4 forms of Host interface, and the internal registers are read and written through these Host interface cycles to complete. ER-TFTMC070-4 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. "Command Write" specifies the address of the register, and then the "Data Write" period can be written to the specified register. And when the 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 the set register address, "Data Read" is to read register data.

Table 14-1: Host Interface Cycle

| Host Interface Cycle | CS# | A0 | 8080 Type MCU |            | 6800 Type MCU |            | Action Description                |
|----------------------|-----|----|---------------|------------|---------------|------------|-----------------------------------|
|                      |     |    | RD#<br>EN     | WR#<br>RW# | RD#<br>EN     | WR#<br>RW# |                                   |
| Command Write        | 0   | 0  | 1             | 0          | 1             | 0          | Write Address of Register         |
| 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 |

All of the Registers of ER-TFTMC070-4 are describe as below. Each register contains 8bits data. In the register table include the detail description, default value and access attribute (RO: Read Only, WO: Write Only, RW: Readable and Writeable).

### 5.14.1 Status Register

Table 14-2: Status Register (STSR)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Memory Write FIFO Full Indicate</b><br>0: The FIFO of Memory Write not Full 1: The FIFO of Memory Write was Full<br>Host can write data to memory only if the Memory Write FIFO is not full.   | 0       | RO     |
| 6   | <b>Memory Write FIFO Empty Indicate</b><br>0: The FIFO of Memory Write not Empty 1: The FIFO of Memory Write was Empty<br>When the Memory Write FIFO is empty, Host can continuous to write 8bpp data with 64 pixels, 16bpp data with 32 pixels or 24bpp data with 16 pixels. | 1       | RO     |

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 5   | <b>Memory Read FIFO Full Indicate</b><br>0: The FIFO of Memory Read not Full 1: The FIFO of Memory Read was Full<br>When the Memory Read FIFO was Full, Host can continuous to read 8bpp data with 64 pixels, 16bpp data with 32 pixels or 24bpp data with 16 pixels.   | 0       | RO     |
| 4   | <b>Memory Read FIFO Empty Indicate</b><br>0: The FIFO of Memory Read not Empty 1: The FIFO of Memory Read was Empty<br>When the Memory Read FIFO not Empty, Host can continuous read pixel data from Memory.  | 1       | RO     |
| 3   | <b>Core Task is Busy, Fontwr_Busy</b><br>This bit represents whether the action is completed for ER-TFTMC070-4 BTE, Geometry engine, DMA, text writing, or graphic writing.<br>0: The action is complete or idle.<br>1: The action is not completed, in a busy state.<br>When the Host program switches text and graphics mode, or changes the base diagram related settings, the program must first verify that the ER-TFTMC070-4 is idle.<br><br><b>Note:</b> In text mode, if 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   | <b>Display RAM Ready for Access</b><br>0: Display RAM is not ready for access 1: Display RAM is ready for access<br>Before check this bit, Host has to setup REG[E4h] bit0 "SDR_INIT" as 1.   | 0       | RO     |
| 1   | <b>Operation Mode Status</b><br>0: Normal operation state 1: Inhibit operation state<br>Inhibit operation state means internal reset event keep running or initial display still running or chip enter power saving state. In power saving state, this bit becomes 1 until PLL clock stop.  | 0       | RO     |
| 0   | <b>Interrupt Pin State</b><br>0: without interrupt active 1: interrupt active   | 0       | RO     |

### 5.14.2 Configuration Registers

REG[00h] Software Reset Register (SRR)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Reconfigure PLL frequency</b><br>Write "1" to this bit will reconfigure PLL frequency.<br><b>Note:</b><br>1. When user change PLL relative parameters, PLL clock won't change immediately, user must set this bit as "1" again.<br>2. User may read (check) this bit to know whether system already switch to PLL clock or not yet.  | 0       | RW     |
| 6-1 | Reserved  | 0       | RO     |
| 0   | <b>Software Reset (Reconfigure PLL Frequency)</b><br>0: Normal Operation.<br>1: Software Reset. The bit will auto clear after reset.<br>Software Reset only reset internal state machine. Configuration Registers value won't be reset. So all read-only flag in the register will return to its initial value. User should have proper action to make sure flag is in desired state. | 0       | WO     |
| 0   | <b>Warning Condition Flag</b><br>0: No warning operation occurred<br>1: Warning condition occurred. Please refer to REG[E4h] bit3 description.  | 0       | RO     |

REG[01h] Chip Configuration Register (CCR)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Check PLL Ready</b><br>Host may read (check) this bit to know whether system already switch to PLL clock or not yet. Read "1" means PLL clock ready and switch successfully.   | 0       | RW     |
| 6   | <b>Mask WAIT# on CS# De-assert</b><br>0: No Mask, WAIT# keep assert if internal state keep busy and cannot accept next R/W cycle, no matter CS# assert/de-assert. If Host cycle cannot be extended while WAIT# keep low, Host program should poll WAIT# and wait it goes high then start next access.<br>1: Mask, WAIT# de-assert when CS# de-assert. Use in Host cycle can be extended by WAIT# automatically. | 1       | RW     |
| 5   | <b>Keypad-scan Enable/Disable</b><br>0: Disable<br>1: Enable  | 0       | RW     |

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 4-3 | <b>TFT Panel I/F Setting</b> 00b:<br>24bits TFT Output 01b:<br>18bits TFT Output 10b:<br>16bits TFT Output<br>11b: Without TFT Output<br>Other unused TFT output pins are set as GPIO or Key-Scan function.   | 01b     | RW     |
| 2   | <b>I2C Master Interface Enable/Disable</b><br>0: Disable (GPIO Function)<br>1: Enable (I2C Master Function)<br>This bit has higher priority than Keypad-scan Enable bit. i.e. if I2C master and Keypad-scan are enable simultaneously then KI[0] and KO[0] will become I2C function & other KI/KO pins still keep Keypad-scan function. | 0       | RW     |
| 1   | <b>Serial Flash or SPI Interface Enable/Disable</b><br>0: Disable (GPIO Function)<br>1: Enable (SPI Master Function)  | 0       | RW     |
| 0   | <b>Host Data Bus Width Selection</b><br>0: 8bits Data Bus 1:<br>16bits Data Bus<br>If Serial Host I/F selected or in initial display operation period, ER-TFTMC070-4 will force this bit as 0 and only allow 8bits access.  | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-6 | <b>Host Read/Write Image Data Format</b><br>0xb: Direct Write, for below I/F format:<br>1. 8bits MCU I/F.<br>2. 16bits MCU I/F with 8bpp data mode 1 & 2.<br>3. 16bits MCU I/F with 16/24bpp data mode 1.<br>4. Serial SPI/I2C I/F.<br>10b: Mask high byte of each data (ex. 16 bit MPU I/F with 8-bpp data mode 1)<br>11b: Mask high byte of even data (ex. 16 bit MPU I/F with 24-bpp data mode 2)   | 0       | RW     |
| 5-4 | <b>Host Read Memory Direction (Only for Graphic Mode)</b><br>00b: Left <input type="checkbox"/> Right then Top <input type="checkbox"/> Bottom.<br>01b: Right <input type="checkbox"/> Left then Top <input type="checkbox"/> Bottom.<br>10b: Top <input type="checkbox"/> Bottom then Left <input type="checkbox"/> Right.<br>11b: Bottom <input type="checkbox"/> Top then Left <input type="checkbox"/> Right.<br>Ignored if canvas in linear addressing mode.  | 0       | RW     |
| 3   | NA   | 0       | RO     |
| 2-1 | <b>Host Write Memory Direction (Only for Graphic Mode)</b><br>00b: Left <input type="checkbox"/> Right then Top <input type="checkbox"/> Bottom (Original)<br>01b: Right <input type="checkbox"/> Left then Top <input type="checkbox"/> Bottom (Horizontal Flip)<br>10b: Top <input type="checkbox"/> Bottom then Left <input type="checkbox"/> Right (Rotate right 90° & Horizontal flip)<br>11b: Bottom <input type="checkbox"/> Top then Left <input type="checkbox"/> Right (Rotate left 90°)<br>Ignored if canvas in linear addressing mode. | 0       | RW     |
| 0   | NA (must keep it as 0)   | 0       | RO     |

REG[03h] Input Control Register (ICR)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Interrupt Pin Active Level</b><br>0: Low Active<br>1: High Active  | 0       | RW     |
| 6   | <b>External Interrupt Signal - PSM[0] Pin De-bounce</b><br>0: Without De-bounce<br>1: Enable De-bounce (1,024 OSC Clock)  | 0       | RW     |
| 5-4 | <b>External Interrupt Signal - PSM[0] Pin Trigger Type</b><br>00b: low level trigger<br>01b: falling edge trigger<br>10b: high level trigger<br>11b: rising edge trigger  | 00b     | RW     |
| 3   | NA  | 0       | RW     |
| 2   | <b>Text Mode Enable</b><br>0: Graphic mode<br>1: Text mode<br>Before toggle this bit user must make sure core task busy bit in status register is done or idle.<br>This bit always 0 (in graphic mode) if canvas" address mode is linear mode.  | 0       | RW     |
| 1-0 | <b>Memory Port Read/Write Destination Selection</b><br>00b: Image buffer (Display RAM) for image data, pattern, user-characters. Support Read-modify-Write.<br>01b: Gamma table for Color Red/Green/Blue. Each color"s gamma table has 256 bytes. User need specify desired gamma table and continuous write 256 bytes.<br>10b: Graphic Cursor RAM (only accept low 8-bits MPU data, similar normal register data r/w.), not support Graphic Cursor RAM read. It contains 4 graphic cursor sets. Each set has 128x16 bits. User need specify target graphic cursor set and continue write 256 bytes.<br>11b: Color palette RAM. It is 64x12 bits SRAM, so even address" data only low 4 bits are valid. Not support Color palette RAM read. User need continue write 128 bytes. | 0       | RW     |



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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <p><b>Write Function: Memory Write Data</b></p> <p>Data to write in memory corresponding to the setting of REG[03h][1:0]. Continuous data write cycle can be accepted in bulk data write case.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>A. Image data in Display RAM: according MPU I/F bit width setting (8/16-bits), Host R/W image data format, canvas color depth and set canvas in block mode.</li> <li>B. Pattern data for BTE operation in Display RAM: according MPU I/F bit width setting (8/16-bits), Host R/W image data format, canvas color depth and set canvas in block mode. Active window's width and height should set as 8x8 or 16x16 depend on user required.</li> <li>C. User-characters in Display RAM: according MPU I/F bit width setting (8/16-bits), Host R/W image data format and set canvas in linear mode.</li> <li>D. Character code: only accept low 8-bits MPU data, similar to normal register R/W. For two bytes character code, input high byte first. To user defined Character, code &lt; 8000h is half size, code &gt;= 8000h is full size.</li> <li>E. Gamma table data: only accept low 8-bits MPU data. User must set "Select Gamma table sets([3Ch] Bit6-5)" to clear internal Gamma table's address counter then start to write data. User should program 256 bytes data to memory data port.</li> <li>F. Graphic Cursor RAM data: only accept low 8-bits MPU data. User must set "Select Graphic Cursor sets" bits to clear internal Graphic Cursor RAM address counter then start to write data.</li> <li>G. Color palette RAM data: only accept low 8-bits MPU data. User must program full Color palette RAM in a continuous 128 byte data write to memory data port and cannot change register address.</li> </ul> <p><b>Read Function: Memory Read Data</b></p> <p>Data to read from memory corresponding to the setting of REG[03h][1:0]. Continuous data read cycle can be accepted in bulk data read case.</p> <p><b>Note1:</b> if you set this port address from different port address, must issue a dummy read, the first data read cycle is dummy read and data should be ignored. Graphic Cursor RAM &amp; Color palette RAM data are not support data read function.</p> <p><b>Note2:</b> read memory data is 4 bytes alignment no matter color depth setting.</p> <p><b>Note3:</b> If user write data to Display RAM user must make sure write FIFO is empty before he change register number or core task busy status bit becomes idle.</p> | --      | RW     |

### 5.14.3 PLL Setting Register

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-6 | <b>PCLK Output Divider Ratio, OD[1:0]</b><br>00b: Divided by 1.<br>01b: Divided by 2.<br>10b: Divided by 3.<br>11b: Divided by 4. | 0       | RW     |
| 5-1 | <b>PCLK Input Divider Ratio, R[4:0]</b><br>The value should be 2~31.  | 10      | RW     |
| 0   | <b>PCLK Feedback Divider Ratio of Loop, N[8]</b><br>Total 9 bits, the value should be 2~511..                                     | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>PCLK PLLDIVN[7:0]</b><br>Total 9 bits, the value should be 2~511. | 60      | RW     |

**Note:** PCLK is used by TFT panel's scan clock and derived from CCLK.

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-6 | <b>MCLK output divider Ratio, OD[1:0]</b><br>00b: Divided by 1.<br>01b: Divided by 2.<br>10b: Divided by 3.<br>11b: Divided by 4. | 0       | RW     |
| 5-1 | MCLK Input Divider Ratio, R[4:0]<br>The value should be 2~31.   | 10      | RW     |
| 0   | MCLK Feedback Divider Ratio of Loop, N[8]<br>Total 9 bits, the value should be 2~511.   | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>MCLK PLLDIVN[7:0]</b><br>Total 9 bits, the value should be 2~511. | 133     | RW     |

**Note:** MCLK is used by internal Display RAM's clock.

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-6 | <b>CCLK Output Divider Ratio, OD[1:0]</b><br><br>00b: Divided by 1.<br>01b: Divided by 2.<br>10b: Divided by 3.<br>11b: Divided by 4. | 0       | RW     |
| 5-1 | <b>CCLK Input Divider Ratio, R[4:0]</b><br>The value should be 2~31.  | 10      | RW     |
| 0   | <b>CCLK Feedback Divider Ratio of Loop, N[8]</b><br>Total 9 bits, the value should be 2~511.  | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>CCLK PLLDIVN[7:0]</b><br>Total 9 bits, the value should be 2~511. | 133     | RW     |

**Note1:** CCLK is used by Core's Clock.

**Note2:** PLL output frequency is calculated from the following the equation:

$$F_{OUT} = X_I \times (N/R) \div OD$$

The input frequency  $X_I/R$  is no less than 1MHz. For example,  $IN = 10\text{MHz}$ ,  $R[4:0] = 01010$ ,  $N[8:0] = 100000000$ ,  $OD[1:0] = 11$ , then

$$F_{OUT} = 10\text{MHz} \times (256 / 10) \div 4 = 64 \text{ MHz}$$

#### 5.14.4 Interrupt Control Register

REG[0Bh] Interrupt Enable Register (INTEN)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Wakeup/Resume Interrupt Enable</b><br>0: Disable<br>1: Enable  | 0       | RW     |
| 6   | <b>External Interrupt Input - PSM[0] Enable)</b><br>0: Disable<br>1: Enable   | 0       | RW     |
| 5   | <b>I2C Master Interrupt Enable</b><br>0: Disable<br>1: Enable   | 0       | RW     |
| 4   | <b>VSYNC Time Base Interrupt Enable</b><br>0: Disable Interrupt<br>1: Enable Interrupt<br>This interrupt event may provide the host processor with Vsync signal information for tearing effect. | 0       | RW     |
| 3   | <b>Keypad-scan Interrupt Enable</b><br>0: Disable Interrupt<br>1: Enable Interrupt  | 0       | RW     |
| 2   | <b>Serial Flash DMA Complete / Draw Task Finished / BTE Process Complete etc. Interrupt Enable</b><br>0: Disable Interrupt<br>1: Enable Interrupt   | 0       | RW     |
| 1   | <b>PWM Timer-1 Interrupt Enable</b><br>0: Disable Interrupt<br>1: Enable Interrupt  | 0       | RW     |
| 0   | <b>PWM Timer-0 Interrupt Enable</b><br>0: Disable Interrupt<br>1: Enable Interrupt  | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Wakeup/Resume Interrupt Flag</b><br><b>Write:</b> Clear Interrupt Flag<br>0: No Operation<br>1: Clear Wakeup/Resume Interrupt Flag<br><b>Read:</b> Read Interrupt Flag<br>0: No Wakeup/Resume Interrupt Happens 1: Wakeup/Resume Interrupt Happens                 | 0       | RW     |
| 6   | <b>External Interrupt Input - PSM[0] Flag</b><br><b>Write:</b> Clear PSM[0] Pin Interrupt Flag<br>0: No Operation<br>1: Clear PSM[0] Interrupt Flag<br><b>Read:</b> Read PSM[0] Pin Interrupt Flag<br>0: No PSM[0] Interrupt Happens<br>1: PSM[0] Interrupt Happens   | 0       | RW     |
| 5   | <b>I2C Master Interrupt Flag</b><br><b>Write:</b> Clear I2C Master Interrupt Flag 0:<br>No Operation<br>1: Clear I2C Master Interrupt Flag<br><b>Read:</b> Read I2C Master Interrupt Flag<br>0: No I2C Master Interrupt Happens 1: I2C Master Interrupt Happens       | 0       | RW     |
| 4   | <b>VSYNC Time Base Interrupt Flag</b><br><b>Write:</b> Clear VSYNC Interrupt Flag<br>0: No Operation<br>1: Clear VSYNC Interrupt Flag<br><b>Read:</b> Read VSYNC Interrupt Flag<br>0: No VSYNC Interrupt Happens 1: VSYNC Interrupt Happens                           | 0       | RW     |
| 3   | <b>Keypad-scan Interrupt Flag</b><br><b>Write:</b> Clear Keypad-scan Interrupt Flag 0:<br>No Operation<br>1: Clear Keypad-scan Interrupt Flag<br><b>Read:</b> Read Keypad-scan Interrupt Flag<br>0: No Keypad-scan Interrupt Happens 1: Keypad-scan Interrupt Happens | 0       | RW     |

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 2   | <b>Serial Flash DMA Complete   Draw Task Finished   BTE Process Complete etc. Interrupt Flag</b><br><b>Write:</b> Clear Process Complete Interrupt Flag 0:<br>No Operation<br>1: Clear Interrupt Flag<br><b>Read:</b> Read Process Complete Interrupt Flag 0:<br>No Interrupt Happens<br>1: Interrupt Happens | 0       | RW     |
| 1   | <b>PWM1 Timer Interrupt Flag Write:</b><br>Clear PWM1 Interrupt Flag<br>0: No Operation<br>1: Clear PWM1 Interrupt Flag<br><b>Read:</b> Read PWM1 Interrupt Flag<br>0: No PWM1 Interrupt Happens<br>1: PWM1 Interrupt Happens   | 0       | RW     |
| 0   | <b>PPWM0 Timer Interrupt Flag</b><br><b>Write:</b> Clear PWM0 Interrupt Flag<br>0: No Operation<br>1: Clear PWM0 Interrupt Flag<br><b>Read:</b> Read PWM0 Interrupt Flag<br>0: No PWM0 Interrupt Happens<br>1: PWM0 Interrupt Happens   | 0       | RW     |

**Note:** If the Host received interruption, but the flag of this register show not interrupted happen, then Host should be to confirm SPI Master State Register Interrupt Flag of REG[BAh].

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Mask Wakeup/Resume Interrupt Flag</b><br>0: Unmask<br>1: Mask      | 0       | RW     |
| 6   | <b>External Interrupt Input - PSM[0] Flag</b><br>0: Unmask<br>1: Mask | 0       | RW     |
| 5   | <b>I2C Master Interrupt Flag</b><br>0: Unmask<br>1: Mask              | 0       | RW     |
| 4   | <b>VSYNC Time Base Interrupt Flag</b><br>0: Unmask<br>1: Mask         | 0       | RW     |
| 3   | <b>Keypad-scan Interrupt Flag</b><br>0: Unmask<br>1: Mask             | 0       | RW     |

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 2   | <b>Serial Flash DMA Complete   Draw Task Finished   BTE Process Complete etc. Interrupt Flag</b><br>0: Unmask<br>1: Mask | 0       | RW     |
| 1   | <b>PWM1 Timer Interrupt Flag</b><br>0: Unmask<br>1: Mask   | 0       | RW     |
| 0   | <b>PWM0 Timer Interrupt Flag</b><br>0: Unmask<br>1: Mask   | 0       | RW     |

**Note:** If the Host masking interrupt flag, then the IER-TFTMC070-4 will not send interruption to Host. If only use some not to be masked interrupt flag, Host can check interrupt flag to know whether there is interruption.

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-6 | NA.   | 0       | RO     |
| 5   | <b>GPIO-F[7:0] Pull-High Enable</b><br>0: without Pull-up Resistor<br>1: with Pull-up | 0       | RW     |
| 4   | <b>GPIO-E[7:0] Pull-High Enable</b><br>0: without Pull-up Resistor<br>1: with Pull-up | 0       | RW     |
| 3   | <b>GPIO-D[7:0] Pull-High Enable</b><br>0: without Pull-up Resistor<br>1: with Pull-up | 0       | RW     |
| 2   | <b>GPIO-C[4:0] Pull-High Enable</b><br>0: without Pull-up Resistor<br>1: with Pull-up | 0       | RW     |
| 1   | <b>DB[15:8] Pull- High Enable</b><br>0: without Pull-up Resistor<br>1: with Pull-up   | 0       | RW     |
| 0   | <b>DB[7:0] Pull- High Enable</b><br>0: without Pull-up Resistor<br>1: with Pull-up    | 0       | RW     |

**Note:** These bits are available only when GPIO function enabled.

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>PD[18] – Function Select</b><br>0: GPIO-D7<br>1: KO[4] | 0       | RW     |
| 6   | <b>PD[17] – Function Select</b><br>0: GPIO-D5<br>1: KO[2] | 0       | RW     |
| 5   | <b>PD[16] – Function Select</b><br>0: GPIO-D4<br>1: KO[1] | 0       | RW     |
| 4   | <b>PD[9] – Function Select</b><br>0: GPIO-D3<br>1: KO[3]  | 0       | RW     |
| 3   | <b>PD[8] – Function Select</b><br>0: GPIO-D2<br>1: KI[3]  | 0       | RW     |
| 2   | <b>PD[2] – Function Select</b><br>0: GPIO-D6<br>1: KI[4]  | 0       | RW     |
| 1   | <b>PD[1] – Function Select</b><br>0: GPIO-D1<br>1: KI[2]  | 0       | RW     |
| 0   | <b>PD[0] – Function Select</b><br>0: GPIO-D0<br>1: KI[1]  | 0       | RW     |

Some of the LCD Data pins are share with GPIO and Keypad-scan pins. If LCD I/F is not 24bpp mode, then PD[18:16 / 8:9 / 2:0] could be defined as GPIO or Keypad-scan pins.



### 5.14.5 LCD Display Control Registers

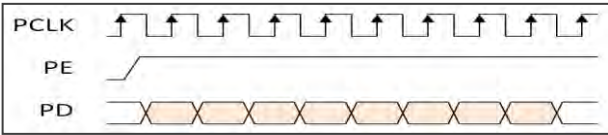
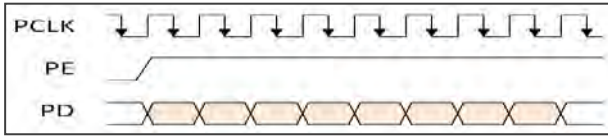
REG[10h] Main/PIP Window Control Register (MPWCTR)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | <b>PIP-1 Window Enable/Disable</b><br>0: PIP-1 Window Disable<br>1: PIP-1 Window Enable<br>PIP-1 window always on top of PIP-2 window.   | 0       | RW     |
| 6   | <b>PIP-2 Window Enable/Disable</b><br>0: PIP-2 Window Disable<br>1: PIP-2 Window Enable<br>PIP-1 window always on top of PIP-2 window  | 0       | RW     |
| 5   | NA   | 0       | RO     |
| 4   | <b>Select Configure PIP 1 or 2 Window's parameters</b><br>PIP window's parameter including Color Depth, Starting Address, Image Width, Display Coordinates, Window Coordinates, Window Width and Window Height.<br>0: To configure PIP 1's parameters. 1: To configure PIP 2's parameters. | 0       | RW     |
| 3-2 | <b>Main Image Color Depth Setting</b><br>00b: 8bpp Generic TFT (256 color) 01b:<br>16bpp Generic TFT (65K color) 1xb:<br>24bpp Generic TFT (1.67M color)   | 1       | RW     |
| 1   | NA   | 0       | RW     |
| 0   | <b>To Control Panel's Synchronous Signals</b><br>0: Sync Mode <input type="checkbox"/> Enable VSYNC, HSYNC, DE<br>1: DE Mode <input type="checkbox"/> Only DE enable, VSYNC & HSYNC in idle state.   | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-4 | NA   | 0       | RO     |
| 3-2 | <b>PIP-1 Window Color Depth Setting</b> 00b:<br>8bpp Generic TFT (256 color) 01b:<br>16bpp Generic TFT (65K color)<br>1xb: 24bpp Generic TFT (1.67M color) | 1       | RW     |
| 1-0 | <b>PIP-2 Window Color Depth Setting</b> 00b:<br>8bpp Generic TFT (256 color) 01b:<br>16bpp Generic TFT (65K color)<br>1xb: 24bpp Generic TFT (1.67M color) | 1       | RW     |

**REG[12h] Display Configuration Register (DPCR)**

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | <b>PCLK Inversion</b><br>0: TFT Panel fetches PD at PCLK rising edge.<br><br>1: TFT Panel fetches PD at PCLK falling edge.<br> | 0       | RW     |
| 6   | <b>Display ON/OFF</b><br>0b: Display Off<br>1b: Display On   | 0       | RW     |
| 5   | <b>Display Test Color Bar</b><br>0b: Disable<br>1b: Enable   | 0       | RW     |
| 4   | <b>This bit must be 0.</b>   | 0       | RW     |
| 3   | <b>VDIR: Vertical Scan Direction</b><br>0: From Top to Bottom<br>1: From bottom to Top   | 0       | RW     |
| 2-0 | <b>Parallel PD[23:0] Output Sequence</b><br>000b: RGB<br>001b: RBG<br>010b: GRB<br>011b: GBR<br>100b: BRG<br>101b: BGR<br>110b: Gray<br>111b: Send out Idle State. All data are 0 (Black) or 1(White). This option has to setup with REG[13h].   | 0       | RW     |

**Note:** When VDIR = 1, PIP window, Graphic Cursor and Text Cursor function will be disable automatically.

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | <b>HSYNC Polarity</b><br>0: Low Active<br>1: High Active   | 0       | RW     |
| 6   | <b>VSYNC Polarity</b><br>0: Low Active<br>1: High Active   | 0       | RW     |
| 5   | <b>PDE Polarity</b><br>0: High Active<br>1: Low Active   | 0       | RW     |
| 4   | <b>PDE Idle State</b><br>0: Pin "PDE" output Low 1:<br>Pin "PDE" output High<br>This is used to setup the PDE output status in Power Saving mode or Display Off.   | 0       | RW     |
| 3   | <b>PCLK Idle State</b><br>0: Pin "PCLK" output Low 1:<br>Pin "PCLK" output High<br>This is used to setup the PCLK output status in Power Saving mode or Display Off.   | 0       | RW     |
| 2   | <b>PD Idle State</b><br>0: Pins "PD[23:0]" output Low 1:<br>Pins "PD[23:0]" output High<br>This is used to setup the PD output status in Vertical/Horizontal Non-Display Period, Power Saving mode or Display Off. | 0       | RW     |
| 1   | <b>HSYNC Idle State</b><br>0: Pin "HSYNC" output Low 1:<br>Pin "HSYNC" output High<br>This is used to setup the HSYNC output status in Power Saving mode or Display Off.   | 1       | RW     |
| 0   | <b>VSYNC Idle State</b><br>0: Pin "VSYNC" output Low 1:<br>Pin "VSYNC" output High<br>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 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 4-1 TFT-LCD interface Timing diagram.

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Horizontal Display Width Setting</b><br>This register is set to a horizontal display width. Its specified LCD screen resolution is 8 pixels in one unit resolution.<br><br>$\text{Horizontal Display Width (pixels)} = (\text{HDWR} + 1) \times 8 + \text{HDFTR}$<br><br>HDFTR (REG[15h]) is the fine-tuning value for Horizontal Display Width. Each fine-tuning resolution is 1 pixels, and the maximum horizontal width is 2,048 pixels. | 4Fh     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-4 | NA   | 0       | RO     |
| 3-0 | <b>Horizontal Display Width Fine Tuning</b><br>This Register is the fine-tuning value for Horizontal Display Width, and each fine-tuning resolution is 1 pixels. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-5 | NA  | 0       | RO     |
| 4-0 | <b>Horizontal Non-Display Period</b><br>This register assign the Period of Horizontal Non-Display. It's also called "Back Porch".<br><br>$\text{Horizontal Non-Display Period (Pixels)} = (\text{HNDR} + 1) \times 8 + \text{HNDFTFTR}$<br><br>HNDFTFTR (REG[17h]) is the fine-tuning value for Horizontal Non-display Period. Each fine-tuning resolution is 1 pixels, and the maximum horizontal width is 2,048 pixels. | 03h     | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-4 | NA  | 0       | RO     |
| 3-0 | <b>Horizontal Non-Display Period Fine Tuning</b><br>This Register is the fine-tuning value for Horizontal Non-display Period, and each fine-tuning resolution is 1 pixels. it is used to support the SYNC mode panel. | 06h     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-5 | NA   | 0       | RO     |
| 4-0 | <b>HSYNC Start Position</b><br>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 8pixel. It's also called "Front Porch".<br><br>$\text{HSYNC Start Position} = (\text{HSTR} + 1) \times 8$ | 1Fh     | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-5 | NA  | 0       | RO     |
| 4-0 | <b>HSYNC Pulse Width</b><br><br>$\text{HSYNC Pulse Width (Pixels)} = (\text{HPW} + 1) \times 8$ | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Vertical Display Height</b><br>REG[1Ah] mapping to VDHR [7:0]<br>REG[1Bh] bit[2:0] mapping to VDHR [10:8], bit[7:3] are not used.<br>The height of the vertical display is in line, and the formula is as follows:<br><br>$\text{Vertical Display Height (Line)} = \text{VDHR} + 1$ | DFh     | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Vertical Non-Display Period</b><br>REG[1Ch] mapping to VNDR [7:0]<br>REG[1Dh] bit[1:0] mapping to VNDR [9:8], REG[1Dh] bit[7:2] are not used.<br>The formula is as follows:<br><br>$\text{Vertical Non-Display Period (Line)} = (\text{VNDR} + 1)$ | 15h     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>VSYNC Start Position</b><br>The starting position from the end of display area to the beginning of VSYNC.<br>$\text{VSYNC Start Position (Line)} = (\text{VSTR} + 1)$ | 0Bh     | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-6 | NA  | 0       | RO     |
| 5-0 | <b>VSYNC Pulse Width</b><br>$\text{VSYNC Pulse Width (Line)} = (\text{VPWR} + 1)$ | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Main Image Start Address</b><br>REG[20h] mapping to MISA[7:0], bit[1:0] must be 0.<br>REG[21h] mapping to MISA[15:8]<br>REG[22h] mapping to MISA[23:16]<br>REG[23h] mapping to MISA[31:24] | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Main Image Width</b><br>REG[24h] mapping to MIW[7:0], bit[1:0] must be 0.<br>REG[25h] bit[4:0] mapping to MIW[12:8], bit[7:5] are not used..<br>The unit is pixel, which is the pixel that represents the horizontal width of the actual LCD. The maximum setting is 8,192 pixels. | 0       | RW     |

**REG[27h-26h] Main Window Upper-Left Corner X-Coordinates (MWULX)**

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Main Window Upper-Left Corner X-Coordinates</b><br>REG[26h] mapping to MWULX[7:0], bit[1:0] must be 0.<br>REG[27h] bit[4:0] mapping to MWULX [12:8], bit[7:5] are not used.<br>The Unit is pixel. The sum of X-Coordinates plus Horizontal Display Width cannot be greater than 8,191. | 0       | RW     |

**REG[29h-28h] Main Window Upper-Left corner Y-Coordinates (MWULY)**

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Main Window Upper-Left Corner Y-Coordinates</b><br>REG[28h] mapping to MWULY[7:0]<br>REG[29h] bit[4:0] mapping to MWULY [12:8], bit[7:5] are not used.<br>Unit: Pixel. 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 |
|-----|--|---------|--------|
| 7-0 | <b>PIP Window Display Upper-Left Corner X-Coordinates</b><br>REG[2Ah] mapping to PWDULX[7:0], bit[1:0] must be 0.<br>REG[2Bh] bit[4:0] mapping to PWDULX[12:8], bit[7:5] are not used.<br>Unit: Pixel. Y-axis coordinates should less than vertical display height. According to bit of Select Configure PIP 1 or 2 Window's parameters(REG[10h]), value will be configured for relative PIP window. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>PIP Window Display Upper-Left Corner Y-Coordinates</b><br>REG[2Ch] mapping to PWDULY[7:0]<br>REG[2Dh] bit[4:0] mapping to PWDULY[12:8], bit[7:5] are not used.<br>Y-axis coordinates should less than vertical display height. According to bit of Select Configure PIP 1 or 2 Window's parameters(REG[10h]), value will be configured for relative PIP window. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>PIP Image Start Address</b><br>REG[2Eh] mapping to PISA[7:0], bit[1:0] must be 0.<br>REG[2Fh] mapping to PISA [15:8]<br>REG[30h] mapping to PISA [23:16]<br>REG[31h] mapping to PISA [31:24]<br>According to bit of Select Configure PIP 1 or 2 Window's parameters. Function bit will be configured for relative PIP window. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>PIP Image Width</b><br>REG[32h] mapping to PIW[7:0], bit[1:0] must be 0.<br>REG[33h] bit[5:0] mapping to PIW[13:8], bit[7:6] are not used.<br>Unit: Pixel. The value is physical width, and maximum value is 8192 pixels.<br>This width should less than horizontal display width. According to bit of Select Configure PIP 1 or 2 Window's parameters, value will be configured for relative PIP window. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>PIP Window 1 or 2 Image Upper-Left Corner X-Coordinates</b><br>REG[34h] mapping to PWIULX[7:0], bit[1:0] must be 0.<br>REG[35h] bit[4:0] mapping to PWIULX[12:8], bit[7:5] are not used.<br>Unit: Pixel. The sum of X-axis coordinates plus PIP image width must less or equal to 8,191. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>PIP Windows Display Upper-Left Corner Y-Coordinates</b><br>REG[36h] mapping to PWIULY[7:0], bit[1:0] must be 0.<br>REG[37h] bit[4:0] mapping to PWIULY[12:8], bit[7:5] are not used.<br>Unit: Pixel. The sum of Y-axis coordinates plus PIP window height should less or equal to 8,191. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>PIP Window Width</b><br>REG[38h] mapping to PWW[7:0], bit[1:0] must be 0.<br>REG[39h] bit[5:0] mapping to PWW[13:8], bit[7:6] are not used.<br>Unit: Pixel. Maximum value is 8,192 pixels. | 0       | RW     |



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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>PIP Window Height</b><br>REG[3Ah] mapping to PWH[7:0], bit[1:0] must be 0.<br>REG[3Bh] bit[5:0] mapping to PWH[13:8], bit[7:6] are not used.<br>Unit: Pixel. The value is physical pixel number and maximum value is 8,192 pixels. | 0       | RW     |

**Note 1:** Pip Window Size and Starting Position in the horizontal direction is 8 pixels, the vertical resolution is 1 line.

**Note 2:** The above registers REG[20h] ~ REG[3Bh] need to be written in turn by LSB to MSB. 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 in turn from LSB (REG[20h]) to MSB (REG[23h]). When REG[23h] was written, ER-TFTMC070-4 will write complete Main Image Start Address to the internal register in actually.

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | <b>Gamma Correction Enable</b><br>0: Disable<br>1: Enable<br>Gamma correction is the last output stage.  | 0       | RW     |
| 6-5 | <b>Gamma Table Select for Host Write Gamma Data</b><br>00b: Gamma table for Blue<br>01b: Gamma table for Green<br>10b: Gamma table for Red<br>11b: NA  | 0       | RW     |
| 4   | <b>Graphic Cursor Enable</b><br>0: Graphic Cursor Disable<br>1: Graphic Cursor Enable<br>Graphic cursor will auto disable if VDIR (REG[12h] bit3) set as "1".  | 0       | RW     |
| 3-2 | <b>Graphic Cursor Selection</b><br>Select one from four graphic cursor types. (00b to 11b) 00b:<br>Graphic Cursor Set 1<br>01b: Graphic Cursor Set 2<br>10b: Graphic Cursor Set 3<br>11b: Graphic Cursor Set 4 | 0       | RW     |
| 1   | <b>Text Cursor Enable</b><br>0: Disable<br>1: Enable<br><b>Note:</b> Text cursor & Graphic cursor cannot enable simultaneously. Graphic cursor has higher priority than Text cursor if enabled simultaneously. | 0       | RW     |

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 0   | <b>Text Cursor Blinking Enable</b><br>0: Disable<br>1: Enable | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Text Cursor Blink Time Setting</b><br>00h: 1 Frame Cycle Time 01h:<br>2 Frames Cycle Time 02h: 3<br>Frames Cycle Time<br>:<br>:<br>FFh: 256 frames Cycle Time | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-5 | NA   | 0       | RO     |
| 4-0 | <b>Text Cursor Horizontal Size Setting</b><br>00000b: 1 Pixel<br>00001b: 2 Pixels<br>:<br>:<br>11111b: 32 Pixels<br><b>Note:</b> When character is enlarged, the cursor setting will multiply the same times as the character enlargement. | 07h     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-5 | NA   | 0       | RO     |
| 4-0 | <b>Text Cursor Vertical Size Setting</b><br>Unit: Pixel, Zero-based number. Value "0" means 1 pixel.<br><b>Note:</b> When character is enlarged, the cursor setting will multiply the same times as the character enlargement. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Graphic Cursor Horizontal Position</b><br>REG[40h] mapping to GCHP[7:0]<br>REG[41h] bit[4:0] mapping to GCHP[12:8], bit[7:5] are not used. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Graphic Cursor Vertical Position</b><br>REG[42h] mapping to GCVP[7:0]<br>REG[43h] bit[4:0] mapping to GCVP[12:8], bit[7:5] are not used. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Graphic Cursor Color 0 with 256 Colors</b><br>RGB Format [7:0] = <b>RRRGGBB</b> | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Graphic Cursor Color 1 with 256 Colors</b><br>RGB Format [7:0] = <b>RRRGGBB</b> | 0       | RW     |

### 5.14.6 Geometric Engine Control Registers

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Start Address of Canvas</b><br>REG[50h] mapping to CVSSA[7:0], bit[1:0] must be 0.<br>REG[51h] mapping to CVSSA[15:8]<br>REG[52h] mapping to CVSSA[23:16]<br>REG[53h] mapping to CVSSA[31:24]<br>These Registers will be ignored if canvas in linear Addressing mode. | 0       | RW     |

REG[55h-54h] Canvas Image Width (CVS\_IMWTH)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Canvas Image Width</b><br>REG[54h] mapping to CVS_IMWTH[7:0], bit[1:0] must be 0.<br>REG[55h] bit[5:0] mapping to CVS_IMWTH[13:8], bit[7:6] are not used.<br>Width = Real Image Width<br>These Registers will be ignored if canvas in linear Addressing mode. | 0       | RW     |

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

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Active Window Upper-Left Corner X-Coordinates</b><br>REG[56h] mapping to AWUL_X[7:0]<br>REG[57h] bit[4:0] mapping to AWUL_X[12:8], bit[7:5] are not used.<br>Unit: Pixel. The sum of X-axis coordinates plus Active Window width cannot large than 8,191.<br>These Registers will be ignored if canvas in linear Addressing mode. | 0       | RW     |

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

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

REG[5Bh-5Ah] Active Window Width (AW\_WTH)

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

REG[5Dh-5Ch] Active Window Height (AW\_HT)

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

REG[5Eh] Color Depth of Canvas & Active Window (AW\_COLOR)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-4 | NA  | 0       | RO     |
| 3   | <b>Select What will Read Back from Graphic Read/Write Position Register</b><br>0: Read back Graphic Write position<br>1: Read back Graphic Read position (Pre-fetch Address)  | 0       | RW     |
| 2   | <b>Canvas Addressing Mode</b><br>0: Block mode (X-Y coordinates addressing) 1: Linear mode  | 0       | RW     |
| 1-0 | <b>Canvas Image's Color Depth &amp; Memory R/W Data Width</b><br><i>In Block Mode:</i><br>00b: 8bpp<br>01b: 16bpp<br>1xb: 24bpp<br><b>Note:</b> Monochrome data can input with any one color depth depends on proper image width.<br><i>In Linear Mode:</i><br>X0: 8-bits memory data read/write.<br>X1: 16-bits memory data read/write | 0       | RW     |

Note: **Please refer to Figure 5-2 Canvas and Active Windows.**

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Write: Set Graphic Read/Write X-Coordinate position</b><br><b>Read: Read Graphic Read/Write X-Coordinate position</b><br>Read back is Read position or Write position depends on REG[5Eh] bit3, Select to read back Graphic Read/Write position.<br>REG[5Fh] mapping to CURH[7:0]<br>REG[60h] bit[4:0] mapping to CURH[12:8], bit[7:5] are not used.<br><i>When DPRAM in Linear mode:</i><br>Memory Read/Write address [15:0], Unit: Byte.<br><i>When DPRAM in Block mode:</i><br>Graphic Read/Write X-Coordinate [12:0], Unit: Pixel. | 0       | RW     |

Note: Host should program proper active window related parameters before configure this register.

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <p><b>Write:</b> Set Graphic Read/Write X-Coordinate position<br/> <b>Read:</b> Read Graphic Read/Write X-Coordinate position<br/> Read back is Read position or Write position depends on REG[5Eh] bit3, Select to read back Graphic Read/Write position.<br/> REG[61h] mapping to CURV[7:0]<br/> REG[62h] bit[4:0] mapping to CURV[12:8], bit[7:5] are not used.</p> <p><b>When DPRAM In Linear Mode:</b><br/> Memory Read/Write address [31:16], Unit: Byte.</p> <p><b>When DPRAM In Block Mode:</b><br/> Graphic Read/Write Y-Coordinate [12:0], Unit: Pixel.</p> | 0       | RW     |

Note: **Host should program proper active window related parameters before configure this register.**

REG[64h-63h] Text Write X-Coordinates Register (F\_CURX)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <p><b>Text Write X-coordinates F_CURX[12:0]</b><br/> REG[63h] mapping to F_CURX[7:0]<br/> REG[64h] bit[4:0] mapping to F_CURX[12:8], bit[7:5] are not used.</p> <p><b>Write:</b> Set Text Write X-Coordinates position<br/> <b>Read:</b> Current Text Write X-Coordinates position</p> | 0       | RW     |

REG[66h-65h] Text Write Y-Coordinates Register (F\_CURY)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <p><b>Text Write Y-coordinates F_CURY[12:0]</b><br/> REG[65h] mapping to F_CURY[7:0]<br/> REG[66h] bit[4:0] mapping to F_CUR[12:8], bit[7:5] are not used.</p> <p><b>Write:</b> Set Text Write Y-Coordinates position<br/> <b>Read:</b> Current Text Write Y-Coordinates position</p> | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Draw Line / Triangle Start Control</b><br><i>Write:</i><br>0: Stop the drawing function.<br>1: Start the drawing function.<br><i>Read:</i><br>0: Drawing function complete.<br>1: Drawing function is processing.  | 0       | RW     |
| 6   | NA  | 0       | RO     |
| 5   | <b>Fill Function for Triangle</b><br>0: Non Fill<br>1: Fill   | 0       | RW     |
| 4-1 | <b>Draw Triangle or Line Select</b><br>0000b: Draw Line<br>0001b: Draw Triangle<br>0010b: Rectangle<br>0011b: Quadrilateral<br>0100b: Pentagon<br>0101b: Polyline (3EP)<br>0110b: Polyline (4EP)<br>0111b: Polyline (5EP)<br>1000b: Ellipse<br>1001b: Rounded-Rectangle<br>1010b, 1011b: NA<br>1100b: Oval Arc on upper-right / 1st Quadrant<br>1101b: Oval Arc on upper-left / 2nd Quadrant<br>1110b: Oval Arc on Lower-Left / 3rd Quadrant<br>1111b: Oval Arc on Lower-Right / 4th Quadrant | 0       | RW     |
| 0   | <b>Polyline Style</b><br>0: Open-end Polyline,<br>1: Closed Polyline (connect last point to start point)  | 0       | RO     |



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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Draw Line/Triangle Point 1 X-coordinates DLHSR[12:0]</b><br>REG[68h] mapping to DLHSR[7:0]<br>REG[69h] bit[4:0] mapping to DLHSR[12:8], bit[7:5] are not used.<br>Unit: Pixel.<br>When draw a rectangle, start point & end point cannot locate at same point or at same X- Coordinates or Y- Coordinates. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Draw Line/Triangle Point 1 Y-coordinates DLVSR[12:0]</b><br>REG[6Ah] mapping to DLVSR[7:0]<br>REG[6Bh] bit[4:0] mapping to DLVSR[12:8], bit[7:5] are not used. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Draw Line/Triangle Point 2 X-coordinates DLHER[12:0]</b><br>REG[6Ch] mapping to DLHER[7:0]<br>REG[6Dh] bit[4:0] mapping to DLHER[12:8], bit[7:5] are not used. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Draw Line/Triangle Point 2 Y-coordinates DLVER[12:0]</b><br>REG[6Eh] mapping to DLVER[7:0]<br>REG[6Fh] bit[4:0] mapping to DLVER[12:8], bit[7:5] are not used. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Draw Line/Triangle Point 3 X-coordinates DTPH[12:0]</b><br>REG[70h] mapping to DTPH[7:0]<br>REG[71h] bit[4:0] mapping to DTPH[12:8], bit[7:5] are not used. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Draw Line/Triangle Point 3 Y-coordinates DTPV[12:0]</b><br>REG[72h] mapping to DTPV[7:0]<br>REG[73h] bit[4:0] mapping to DTPV[12:8], bit[7:5] are not used. | 0       | RW     |

Note: **When Drawing triangle: If any two endpoints overlap will draw a line. And three endpoints overlap will draw a dot.**

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | <b>Draw Circle / Ellipse / Square / Circle Square Control</b><br><b>Write:</b><br>0: Stop the drawing function.<br>1: Start the drawing function.<br><b>Read:</b><br>0: Drawing function complete.<br>1: Drawing function is processing. | 0       | RW     |
| 6   | <b>Fill the Circle / Ellipse / Square / Rounded-rectangle Control</b><br>0: Non Fill<br>1: Fill  | 0       | RW     |
| 5-4 | <b>Draw Circle / Ellipse / Square / Ellipse Curve / Rounded-rectangle Select</b><br>00b: Draw Circle / Ellipse<br>01b: Draw Arc<br>10b: Draw Rectangle<br>11b: Draw Rounded-Rectangle  | 0       | RW     |
| 3-2 | NA   | 0       | RO     |
| 1-0 | <b>Draw Circle / Ellipse Curve Part Select (DECP)</b><br>00b: bottom-left Ellipse Curve<br>01b: upper-left Ellipse Curve<br>10b: upper-right Ellipse Curve<br>11b: bottom-right Ellipse Curve  | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Draw Circle/Ellipse/Rounded-rectangle Major-Radius</b><br>REG[77h] mapping to ELL_A[7:0]<br>REG[78h] bit[4:0] mapping to ELL_A[12:8], bit[7:5] are not used.<br>If draw a circle, then must set major radius equal to 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 |
|-----|---|---------|--------|
| 7-0 | <b>Draw Circle/Ellipse/Rounded-rectangle Minor-Radius</b><br>REG[79h] mapping to ELL_B[7:0]<br>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 |
|-----|---|---------|--------|
| 7-0 | <b>Draw Circle/Ellipse/Rounded-rectangle Center X-coordinates</b><br>REG[7Bh] mapping to DEHR[7:0]<br>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 |
|-----|---|---------|--------|
| 7-0 | <b>Draw Circle/Ellipse/Rounded-rectangle Center Y-coordinates</b><br>REG[7Dh] mapping to DEVR[7:0]<br>REG[7Eh] bit[4:0] mapping to DEVR[12:8], bit[7:5] are not used. | 0       | RW     |

Note: **The unit of REG[77h] ~ REG[7Eh] is Pixel.**

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Foreground Color - Red (for Draw, Text or Color Expansion)</b><br>256 Colors: Red mapping to bit[7:5] 65K<br>Colors: Red mapping to bit[7:3] 16.7M<br>Colors: Red mapping to bit[7:0] | FFh     | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Foreground Color – Green (for Draw, Text or Color Expansion)</b> <ul style="list-style-type: none"> <li>■ 256 Colors: Green mapping to bit[7:5]</li> <li>■ 65K Colors: Green mapping to bit[7:2]</li> <li>■ 16.7M Colors: Green mapping to bit[7:0]</li> </ul> | FFh     | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Foreground Color – Blue (for Draw, Text or Color Expansion)</b> <ul style="list-style-type: none"> <li>■ 256 Colors: Blue mapping to bit[7:6]</li> <li>■ 65K Colors: Blue mapping to bit[7:3]</li> <li>■ 16.7M Colors: Blue mapping to bit[7:0]</li> </ul> | FFh     | RW     |

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

### 5.14.7 PWM Control Registers

REG[84h] PWM Prescaler Register (PSCLR)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>PWM Pre-scaler Register</b><br>This register determine the pre-scaler value for Timer 0 and 1. The Base Frequency is:<br>$\text{Core\_Freq} / (\text{Prescaler} + 1)$ | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-6 | <b>Setup PWM Timer-1 Divisor</b> (Select 2 <sup>nd</sup> Clock Divider"s MUX Input for PWM Timer-1)<br>00b = 1<br>01b = 1/2<br>10b = 1/4<br>11b = 1/8  | 0       | RW     |
| 5-4 | <b>Setup PWM Timer-0 Divisor</b> (Select 2 <sup>nd</sup> Clock Divider"s MUX Input for PWM Timer-0)<br>00b = 1<br>01b = 1/2<br>10b = 1/4<br>11b = 1/8  | 0       | RW     |
| 3-2 | <b>PWM[1] Function Control</b><br>0xb: PWM[1] output system error flag (Scan FIFO POP error or Memory access out of range)<br>10b: PWM[1] output PWM timer 1 event or invert of PWM timer 0<br>11b: PWM[1] output Oscillator Clock | 0       | RW     |
| 1-0 | <b>PWM[0] Function Control</b><br>0xb: PWM[0] becomes GPIOC[7]<br>10b: PWM[0] output PWM Timer 0<br>11b: PWM[0] output Core Clock (CCLK).  | 0       | RW     |

REG[86h] PWM Configuration Register (PCFGR)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | NA   | 0       | RO     |
| 6   | <b>PWM Timer-1 Output Inverter On/Off</b><br>Determine the output inverter on/off for Timer 1.<br>0 = Inverter off<br>1 = Inverter on for PWM1   | 0       | RW     |
| 5   | <b>PWM Timer-1 Auto Reload On/Off</b><br>Determine auto reload on/off for Timer 1.<br>0: One-Shot Mode<br>1: Interval Mode (Auto Reload)   | 1       | RW     |
| 4   | <b>PWM Timer-1 Start/Stop</b><br>0: Stop<br>1: Start<br>In Interval Mode, Host need program it as 0 to stop PWM timer. In One-shot Mode, this bit will auto clear.<br>The Host may read this bit to know current PWMx is running or stopped. | 0       | RW     |
| 3   | <b>PWM Timer-0 Dead Zone Enable</b><br>0: Disable<br>1: Enable   | 0       | RW     |
| 2   | <b>PWM Timer-0 Output Inverter On/Off</b><br>Determine the output inverter on/off for Timer 0.<br>0 = Inverter off<br>1 = Inverter on for PWM0   | 0       | RW     |
| 1   | <b>PWM Timer-0 Auto Reload On/Off</b><br>Determine auto reload on/off for Timer 0.<br>0: One-Shot Mode<br>1: Interval Mode (Auto Reload)   | 1       | RW     |
| 0   | <b>PWM Timer-0 Start/Stop</b><br>0: Stop<br>1: Start<br>In Interval Mode, Host need program it as 0 to stop PWM timer. In One-shot Mode, this bit will auto clear.<br>The Host may read this bit to know current PWMx is running or stopped. | 0       | RW     |

REG[87h] Timer-0 Dead Zone Length Register [DZ\_LENGTH]

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Timer-0 Dead Zone Length Register</b><br>These 8 bits determine the dead zone length. The 1 unit time of the dead zone length is equal to Timer 0. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Timer-0 compare Buffer Register</b><br>REG[88h] mapping to TCMPB0 [7:0]<br>REG[89h] mapping to TCMPB0 [15:8]<br>The Timer-0 Compare Buffer Register have a total of 16bits. When the counter is equal to or less than the value of this register, and if INV_ON bit is Off, then PWM1 output is high level. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Timer-0 Count Buffer Register [15:0]</b><br>REG[8Ah] mapping to TCNTB0 [7:0]<br>REG[8Bh] mapping to TCNTB0 [15:8]<br>The Timer-0 count registers have a total of 16bit. When the counter down to 0 and the Reload_EN is enabled, the PWM will overloads 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[8Ch-8Dh] Timer-1 Compare Buffer Register [TCMPB1]

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Timer-1 Compare Buffer Register</b><br>REG[8Ch] mapping to TCMPB1 [7:0]<br>REG[8Dh] mapping to TCMPB1 [15:8]<br>The Timer-1 Compare Buffer Register have a total of 16bits. When the counter is equal to or less than the value of this register, and if INV_ON bit is Off, then PWM1 output is high level. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Timer-1 Count Buffer Register [15:0]</b><br>REG[8Eh] mapping to TCNTB1 [7:0]<br>REG[8Fh] mapping to TCNTB1 [15:8]<br>The Timer-1 count registers have a total of 16bit. When the counter down to 0 and the Reload_EN is enabled, the PWM will overloads 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     |

### 5.14.8 Bit Block Transfer Engine (BTE) Control Registers

REG[90h] BTE Function Control Register 0 (BLT\_CTRL0)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-5 | NA  | 0       | RO     |
| 4   | <b>BTE Function Enable / Status</b><br><i>Write:</i><br>0: No Action<br>1: BTE Enable<br><br><i>Read:</i><br>0: BTE function is idle. 1:<br>BTE function is busy.<br><br>When BTE function enable, Normal Host R/W memory through<br>canvas[active window] doesn't allow. | 0       | RW     |
| 3-1 | NA  | 0       | RO     |
| 0   | <b>Pattern Format</b><br>0: 8x8<br>1: 16x16   | 0       | RW     |



REG[91h] BTE Function Control Register1 (BLT\_CTRL1)

| Bit  | Description  | Default | Access |   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|--|--|---------|--------|---|-------------|-------|---------------|-------|---|-------|--------------------|-------|-----------|-------|--------------------|-------|-----------|-------|----------------|-------|---|-------|---------------|-------|-----------------------|-------|------|-------|----------------|-------|------|-------|----------------|-------|-----------|-------|---------------|
| 7-4  | <b>BTE ROP Code or Color Expansion Starting</b><br>ROP is the acronym for Raster Operation. Some of BTE operation code has to collocate with ROP for the detailed function.  | 0       | RW     |   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  |  |         |        |   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | <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><math>\sim S0 \cdot \sim S1</math> or <math>\sim (S0+S1)</math></td></tr><tr><td>0010b</td><td><math>\sim S0 \cdot S1</math></td></tr><tr><td>0011b</td><td><math>\sim S0</math></td></tr><tr><td>0100b</td><td><math>S0 \cdot \sim S1</math></td></tr><tr><td>0101b</td><td><math>\sim S1</math></td></tr><tr><td>0110b</td><td><math>S0 \wedge S1</math></td></tr><tr><td>0111b</td><td><math>\sim S0 + \sim S1</math> or <math>\sim (S0 \cdot S1)</math></td></tr><tr><td>1000b</td><td><math>S0 \cdot S1</math></td></tr><tr><td>1001b</td><td><math>\sim (S0 \wedge S1)</math></td></tr><tr><td>1010b</td><td><math>S1</math></td></tr><tr><td>1011b</td><td><math>\sim S0 + S1</math></td></tr><tr><td>1100b</td><td><math>S0</math></td></tr><tr><td>1101b</td><td><math>S0 + \sim S1</math></td></tr><tr><td>1110b</td><td><math>S0 + S1</math></td></tr><tr><td>1111b</td><td>1 (Whiteness)</td></tr></table> |         |        | Bit[7:4]                                    | Description | 0000b | 0 (Blackness) | 0001b | $\sim S0 \cdot \sim S1$ or $\sim (S0+S1)$ | 0010b | $\sim S0 \cdot S1$ | 0011b | $\sim S0$ | 0100b | $S0 \cdot \sim S1$ | 0101b | $\sim S1$ | 0110b | $S0 \wedge S1$ | 0111b | $\sim S0 + \sim S1$ or $\sim (S0 \cdot S1)$ | 1000b | $S0 \cdot S1$ | 1001b | $\sim (S0 \wedge S1)$ | 1010b | $S1$ | 1011b | $\sim S0 + S1$ | 1100b | $S0$ | 1101b | $S0 + \sim S1$ | 1110b | $S0 + S1$ | 1111b | 1 (Whiteness) |
|  | Bit[7:4]   |         |        | Description                                 |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 0000b  |         |        | 0 (Blackness)                               |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 0001b  |         |        | $\sim S0 \cdot \sim S1$ or $\sim (S0+S1)$   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 0010b  |         |        | $\sim S0 \cdot S1$                          |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 0011b  |         |        | $\sim S0$                                   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 0100b  |         |        | $S0 \cdot \sim S1$                          |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 0101b  |         |        | $\sim S1$                                   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 0110b  |         |        | $S0 \wedge S1$                              |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 0111b  |         |        | $\sim S0 + \sim S1$ or $\sim (S0 \cdot S1)$ |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 1000b  |         |        | $S0 \cdot S1$                               |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 1001b  |         |        | $\sim (S0 \wedge S1)$                       |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 1010b  |         |        | $S1$  |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 1011b  |         |        | $\sim S0 + S1$                              |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 1100b  |         |        | $S0$  |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 1101b  |         |        | $S0 + \sim S1$                              |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
|  | 1110b  |         |        | $S0 + S1$                                   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
| 1111b  | 1 (Whiteness)  |         |        |   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
| If BTE operation code function are color expansion with or without chroma key (08h / 09h / Eh / Fh), then these bits stand for starting bit on BTE window left boundary. MSB stands for left most pixel. For 8-bits Host, value should within 0 to 7. For 16-bits Host, value should within 0 to 15. |  |         |        |   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |
| <b>BTE Operation Code bit[3:0]</b><br>ER-TFTMC070-4 embedded a 2D BTE Engine, it can execute 13 BTE functions. Some of BTE Operation Code has to accommodate with the ROP code for the advance function.   | 0  | RW      |        |   |             |       |               |       |   |       |                    |       |           |       |                    |       |           |       |                |       |   |       |               |       |                       |       |      |       |                |       |      |       |                |       |           |       |               |

Table 14-4: BTE Operation Code

| REG[91h]<br>Bit[3:0] | Description  |
|----------------------|--|
| 0000b                | <b>MCU Write with ROP</b><br>S0: Data comes from Host S1: Data comes from Memory<br>D: According to ROP write to Memory  |
| 0001b                | Reserved   |
| 0010b                | <b>Memory Copy with ROP</b><br>S0: Data comes from Memory S1: Data comes from Memory<br>D: According to ROP Write to memory  |
| 0011b                | Reserved   |
| 0100b                | <b>MCU Write with Chroma Keying (without ROP)</b><br>S0: Data comes from Host<br>If the Host data is not the same color as the Chroma Key (Background Color Register), the data is written to the destination memory.  |
| 0101b                | <b>Memory Copy (move) with Chroma keying (without ROP)</b><br>S0: Data comes from Memory<br>If the S0 data is not the same color as the chroma key Background Color Register), the data is written to the destination.   |
| 0110b                | <b>Pattern Fill with ROP</b><br>Pattern was specified by S0.   |
| 0111b                | <b>Pattern Fill with Chroma Keying</b><br>Pattern was specified by S0.<br>If the data for S0 is not the same as the Chroma Key (Background Color) color, then it is written to the destination memory.   |
| 1000b                | <b>MCU Write with Color Expansion</b><br>S0: Data comes from Host<br>BTE converts it to the specified color and color depth and writes to the destination memory.  |
| 1001b                | <b>MCU Write with Color Expansion and Chroma Keying</b><br>The monochrome data required by S0 is written by MCU if the bit of monochrome data is 1. The processed data is a foreground color, and if the monochrome data is 0, it is not written. The data is also referenced in the destination memory setting. |
| 1010b                | <b>Memory Copy with Opacity</b><br>S0, S1 & D: Data come from Memory   |

Table 14-4: BTE Operation Code (Continued)

| REG[91h]<br>Bit[3:0] | Description   |
|----------------------|---|
| 1011b                | <b>MCU Write with Opacity</b><br>S0: Data comes from Host S1:<br>Data comes from Memory<br>D: Refer to the alpha blending operation and write to the destination memory.  |
| 1100b                | <b>Solid Fill</b><br>The write value is a register setting value and the target is written to the destination memory.   |
| 1101b                | Reserved  |
| 1110b                | <b>Memory Copy with Color Expansion</b><br>S0 and D are in memory and S1 is not in use.<br>S0 must be loaded with the microprocessor's write or DMA preload 8bpp or 16bpp of color depth into memory, so the S0 color depth should follow that color depth.   |
| 1111b                | <b>Memory Copy with Color Expansion and Chroma Keying</b><br>S0 and D are in memory and S1 is not in use.<br>S0 must be loaded with the microprocessor's write or DMA preload 8bpp or 16bpp of color depth into memory, so the S0 color depth should follow that color depth.<br>If the S0 data bit = 0, then D is not written to any data. 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   | NA  | 0       | RO     |
| 6-5 | <b>S0 Color Depth</b><br>00b: 256 Color (8bpp)<br>01b: 64k Color (16bpp)<br>1xb: 16M Color (24bpp)  | 0       | RW     |
| 4-2 | <b>S1 Color Depth</b><br>000b: 256 Color (8bpp)<br>001b: 64k Color (16bpp)<br>010b: 16M Color (24bpp)<br>011b: Constant color (S1 memory start address" setting definition change as S1 constant color definition)<br>100b: 8 bit pixel alpha blending<br>101b: 16 bit pixel alpha blending | 0       | RW     |

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

REG[93h-96h] Source 0 Memory Start Address (S0\_STR)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Source 0 Memory Start Address [31:2]</b><br>REG[93h] mapping to S0_STR [7:2]<br>REG[94h] mapping to S0_STR [15:8]<br>REG[95h] mapping to S0_STR [23:16]<br>REG[96h] mapping to S0_STR [31:24]<br><b>Note:</b> REG[93h] bit[1:0] fix at 0. | 0       | RW     |

REG[97h-98h] Source 0 Image Width (S0\_WTH)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Source 0 Image Width [12:2]</b><br>REG[97h] mapping to S0_WTH [7:2]<br>REG[98h] bit[4:0] mapping to S0_WTH [12:8], bit[7-5] are not used.<br><b>Note:</b> It must be divisible by 4, and REG[97h] bit[1:0] must fix at 0. Unit: Pixel. | 0       | RW     |

REG[99h-9Ah] Source 0 Window Upper-Left Corner X-Coordinates (S0\_X)

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

REG[9Bh-9Ch] Source 0 Window Upper-Left corner Y-Coordinates (S0\_Y)

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

REG[9Dh-A0h] Source 1 Memory Start Address 0 (S1\_STR)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Source 1 Memory Start Address [31:2]</b><br>REG[9Dh] bit[7:2] mapping to S1_STR[7:2]<br>REG[9Eh] mapping to S1_STR[15:8] REG[9Fh]<br>mapping to S1_STR[23:16] REG[A0h] mapping<br>to S1_STR[31:24]<br><b>Note1:</b> REG[9Dh] bit[1:0] must fix at 0.<br><b>Note2:</b> If source 1(S0) set as Constant Color then S1 memory start<br>address" setting definition change as S1 constant color definition:<br>REG[9Dh] is RED element (S1_RED), REG[9Eh] is Green element<br>(S1_GREEN), REG[9Fh] is Blue Element (S1_BLUE), REG[A0h] is not<br>used for color definition. | 0       | RW     |

REG[A1h-A2h] Source 1 Image Width (S1\_WTH)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Source 1 Image Width [12:2]</b><br>REG[A1h] [7:2] mapping to S1_WTH [7:2]<br>REG[A2h] bit[4:0] mapping to S1_WTH[12:8], bit[7:5] are not used.<br><b>Note:</b> It must be divisible by 4, and REG[A1h] bit[1:0] must fix at<br>0. Unit: Pixel. | 0       | RW     |

REG[A3h-A4h] Source 1 Window Upper-Left Corner X-Coordinates (S1\_X)

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

REG[A5h-A6h] Source 1 Window Upper-Left corner Y-Coordinates (S1\_Y)

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

REG[A7h-AAh] Destination Memory Start Address (DT\_STR)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-2 | <b>Destination Memory Start Address [31:2]</b><br>REG[A7h] bit[7:2] mapping to DT_STR [7:2]<br>REG[A8h] mapping to DT_STR [15:8] REG[A9h]<br>mapping to DT_STR [23:16] REG[AAh] mapping<br>to DT_STR [31:24]<br><b>Note:</b> REG[A7h] bit[1:0] must fix at 0. | 0       | RW     |

Note: The destination memory start address cannot be in the source 0 and Source 1 processing block, otherwise there will be an incorrect result output.

Start Address ~ S0/1's (Image\_Width)x (Image\_Height)x ([1 | 2 | 3]Color Depth)

REG[ABh-ACH] Destination Image Width (DT\_WTH)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Destination Image Width [12:2]</b><br>REG[ABh] mapping to DT _WTH [7:2]<br>REG[ACH] bit[4:0] mapping to DT _WTH [12:8], REG[ACH]<br>bit[7-5] are not used.<br><b>Note:</b> It must be divisible by 4. And REG[ABh] bit[1:0] must fix 0.<br>Unit: Pixel. | 0       | RW     |

REG[ADh-AEh] Destination Window Upper-Left Corner X-Coordinates (DT\_X)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Destination Window Upper-Left Corner X-Coordinates [12:0]</b><br>REG[ADh] mapping to DT_X [7:0]<br>REG[AUh] bit[4:0] mapping to DT_X [12:8], bit[7-5] are not used. | 0       | RW     |

REG[AFh-B0h] Destination Window Upper-Left Corner Y-Coordinates (DT\_Y)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Destination Window Upper-Left Corner X-Coordinates [12:0]</b><br>REG[AFh] mapping to DT_Y [7:0]<br>REG[B0h] bit[4:0] mapping to DT_Y [12:8], bit[7-5] are not used. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>BTE Window Width [12:0]</b><br>REG[B1h] mapping to BLT_WTH [7:0]<br>REG[B2h] bit[4:0] mapping to BLT_WTH [12:8], bit[7-5] are not used.<br>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[B3h-B4h] BTE Window Height (BLT\_HIG)

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

REG[B5h] Alpha Blending (APB\_CTRL)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-4 | NA  | 0       | RO     |
| 5-0 | <b>Window Alpha Blending Effect for S0 &amp; S1</b><br>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.<br>00h: 0<br>01h: 1/32<br>02h: 2/32<br>:<br>:<br>1Eh: 30/32<br>1Fh: 31/32<br>2Xh: 1<br><b>Output Effect</b><br><b>= [ S0 image x (1 - Alpha Setting Value) ] +</b><br><b>(S1 Image x Alpha Setting Value)</b> | 0       | RW     |

REG[B6h-CBh] RESERVED

| Bit | Description | Default | Access |
|-----|-------------|---------|--------|
| 7-0 | Not use     | 0       | RO     |

### 5.14.9 Text Engine Registers

REG[CCh] Character Control Register 0 (CCR0)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7:6 | <b>Character Source Selection</b><br>00b: Select internal CGROM Character<br>01b: Select external CGROM Character<br>10b: Select user-defined Character.<br>11b: NA   | 0       | RW     |
| 5-4 | <b>Character Height Setting for user-defined Character</b><br>00b: 16 dots, ex. 8x16 / 16x16<br>01b: 24 dots, ex. 12x24 / 24x24<br>10b: 32 dots, ex. 16x32 / 32x32<br><b>Note:</b><br>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.<br>2. Internal CGROM supports size 8x16 / 12x24 / 16x32. | 0       | RW     |
| 3-2 | N   | 0       | RO     |
| 1-0 | <b>Character Selection for Internal CGROM</b><br>When bit[7:6] = 00b, Internal CGROM supports character sets with the standard coding of ISO/IEC 8859-1,2,4,5, which supports English and most of European country languages<br>00b: ISO/IEC 8859-1<br>01b: ISO/IEC 8859-2<br>10b: ISO/IEC 8859-4<br>11b: ISO/IEC 8859-5  | 0       | RW     |

REG[CDh] Character Control Register 1 (CCR1)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Full Alignment Selection 0:</b><br>Full alignment disable 1:<br>Full alignment enable.<br>When Full alignment enable, displayed character width is equal to (Character Height)/2 if character width equal or small than (Character Height)/2, otherwise displayed font width is equal to Character Height. | 0       | RW     |
| 6   | <b>Chroma Keying Enable on Text Input</b><br>0: Character's background displayed with specified color.<br>1: Character's background displayed with original canvas" background.   | 0       | RW     |
| 5   | NA  | 0       | RO     |



| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 4   | <b>Character Rotation</b><br>0: Normal. Text direction from left to right then from top to bottom<br>1: Counterclockwise 90 degree & vertical flip. Text direction from top to bottom then from left to right (it should accommodate with set VDIR as 1). This attribute can be changed only when previous Text write finished (Core_Busy = 0) | 0       | RW     |
| 3-2 | <b>Character Width Enlargement Factor</b><br>00b: X1.<br>01b: X2.<br>10b: X3.<br>11b: X4.  | 0       | RW     |
| 1-0 | <b>Character Height Enlargement Factor</b><br>00b: X1.<br>01b: X2.<br>10b: X3.<br>11b: X4.   | 0       | RW     |

REG[CEh-CFh] RESERVED

| Bit | Description | Default | Access |
|-----|-------------|---------|--------|
| 7-0 | NA          | 0       | RO     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-5 | NA   | 0       | RO     |
| 4-0 | <b>Character Line Gap Setting</b><br>Setting the character line gap when meet active window boundary. (Unit: pixel)<br>Color of gap will fill-in background color. It won't be enlarged by character enlargement function. | 0       | RW     |

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

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-6 | NA  | 0       | RW     |
| 5-0 | <b>Character to Character Space Setting</b><br>00h: 0 pixel<br>01h: 1 pixel<br>02h: 2 pixels<br>:<br>:<br>3Fh: 63 pixels<br>Color of space will fill-in background color. It won't be enlarged by character enlargement function. | 0       | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Foreground Color – Red (for Draw, Text or Color Expansion)</b><br>■ 256 Colors: Red mapping to bit[7:5]<br>■ 65K Colors: Red mapping to bit[7:3]<br>■ 16.7M Colors: Red mapping to bit[7:0] | FFh     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Foreground Color – Green (for Draw, Text or Color Expansion)</b><br>■ 256 Colors: Green mapping to bit[7:5]<br>■ 65K Colors: Green mapping to bit[7:2]<br>■ 16.7M Colors: Green mapping to bit[7:0] | FFh     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Foreground Color – Blue (for Draw, Text or Color Expansion)</b><br>■ 256 Colors: Blue mapping to bit[7:6]<br>■ 65K Colors: Blue mapping to bit[7:3]<br>■ 16.7M Colors: Blue mapping to bit[7:0] | FFh     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Background Color – Red (for Text or Color Expansion)</b> <ul style="list-style-type: none"> <li>■ 256 Colors: Red mapping to bit[7:5]</li> <li>■ 65K Colors: Red mapping to bit[7:3]</li> <li>■ 16.7M Colors: Red mapping to bit[7:0]</li> </ul> <b>Note:</b> No matter background transparency is enabled or not, don't set same value with Foreground Color otherwise image or text will become a square with Foreground Color even BTE function. | 00h     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Background Color – Green (for Text or Color Expansion)</b> <ul style="list-style-type: none"> <li>■ 256 Colors: Green mapping to bit[7:5]</li> <li>■ 65K Colors: Green mapping to bit[7:2]</li> <li>■ 16.7M Colors: Green mapping to bit[7:0]</li> </ul> <b>Note:</b> No matter background transparency is enabled or not, don't set same value with Foreground Color otherwise image or text will become a square with Foreground Color even BTE function. | 00h     | RW     |

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>Background Color – Blue (for Text or Color Expansion)</b> <ul style="list-style-type: none"> <li>■ 256 Colors: Blue mapping to bit[7:6]</li> <li>■ 65K Colors: Blue mapping to bit[7:3]</li> <li>■ 16.7M Colors: Blue mapping to bit[7:0]</li> </ul> <b>Note:</b> No matter background transparency is enabled or not, don't set same value with Foreground Color otherwise image or text will become a square with Foreground Color even BTE function. | 00h     | RW     |

REG[D8h] – REG[DAh]: Reserved

| Bit | Description | Default | Access |
|-----|-------------|---------|--------|
| 7-0 | NA          | 0       | RO     |

REG[DBh] CGRAM Start Address 0 (CGRAM\_STR0)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>CGRAM Start ADDRESS [7:0]</b><br>User-defined Characters space<br>REG[DBh] mapping to CGRAM_STR [7:0]<br>REG[DCh] mapping to CGRAM_STR [15:8]<br>REG[DEh] mapping to CGRAM_STR [23:16]<br>REG[DEh] mapping to CGRAM_STR [31:24]<br>Host must use canvas image setting to organize CGRAM data and set CGRAM address to tell engine where to fetch CGRAM data. | 0       | RW     |

If user wants to change rotate attribute, character line gap, character-to-character space, foreground color, background color and Text/graphic mode setting, he must make sure Task\_Busy (Status Register bit3) status bit is low.

### 5.14.10 Power Management Control Register

**REG[DFh]: Power Management Register (PMU)**

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | <p><b>Enter Power Saving State</b></p> <p>0: Normal state or wake up from power saving state 1: Enter power saving state.</p> <p><b>Note:</b></p> <p>There are 3 ways to wake up from power saving state: External interrupt event, Key Scan wakeup and Software wakeup.</p> <p>Write this bit to 0 will cause Software wakeup. It will be cleared until chip resume. MPU must wait until system quit from power saving state than allow to write other registers. User may check this bit or check status bit1 (power saving)</p> | 0       | RW     |
| 6-2 | NA   | 0       | RO     |
| 1-0 | <p><b>Power Saving Mode Definition</b></p> <p>00b: NA</p> <p>01b: Standby Mode, CCLK &amp; PCLK will Stop. MCLK is provided by MPLL.</p> <p>10b: Suspend Mode, CCLK &amp; PCLK will Stop. MCLK is provided by OSC.</p> <p>11b: Sleep Mode, All of Clocks and PLL will Stop.</p>  | 3       | RW     |

### 5.14.11 Display RAM Control Register

**REG[E0h] SDRAM Attribute Register (SDRAR)**

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>SDRAM Power Saving</b><br>0: Execute power down command to enter power saving mode<br>1: Execute self refresh command to enter power saving mode | 0       | RW     |
| 6   | Must keep 0.  | 0       | RW     |
| 5   | <b>SDRAM Bank Number, SDR_BANK</b><br>1: 4 Banks<br><i>Must set to 1 after reset.</i>   | 1       | RW     |
| 4-3 | <b>SDRAM Row Addressing, SDR_ROW</b><br>00b: 2K (A0-A10)<br><i>Must set to 00 after reset.</i>  | 1       | RW     |
| 2-0 | <b>SDRAM Column Addressing, SDR_COL</b><br>000b: 256 (A0-A7)<br><i>Must set to 000 after reset.</i>   | 0       | RW     |

Table 14-5: The initialize of REG[E0h]

| Embedded Display RAM Type | REG[E0h]    | Description  |
|---------------------------|-------------|--|
| 32Mb(4MB, 2Mx16)          | <b>0x20</b> | Bank no : 4, Row Size : 2048,<br>Column Size : 256 |

**Note:** The value of register REG[E0h] must be set according to above table. Otherwise, the display of TFT panel will abnormal and the image is garbled.

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

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-3 | Must keep 0.   | 0       | RW     |
| 2-0 | <b>SDRAM CAS latency, SDR_CASLAT</b><br>010b: 2 SDRAM clock<br>011b: 3 SDRAM clock<br>Others: Reserved<br><b>Note:</b> The suggest setting value of this register is <b>03h</b> .<br>This register was locked after SDR_INITDONE (REG[E4h] bit0) was set as 1. | 011b    | RW     |

**REG[E2h-E3h] SDRAM Auto Refresh Interval (SDR\_REF)**

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>SDRAM Auto Refresh Interval</b><br>REG[E2h] mapping to SDR_REF [7:0].<br>REG[E3h] mapping to SDR_REF [15:8]<br>The internal refresh time is determined according to the period specification of the SDRAM's refresh and the row size. For example, if the SDRAM frequency is 100MHz, SDRAM's refresh period Tref is 64ms, and the row size is 4,096, then the internal refresh time should be less than $64 \times 10^{-3} / 4096 \times 100 \times 10^6 \approx 1562 = 61Ah$ . Therefore the REG[E3h][E2h] is set 030Dh.<br><b>Note:</b> If this register is set to 0000h, SDRAM automatic refresh will be prohibited. | 00h     | RW     |

Table 14-6: The Reference Setting of REG[E3h-E2h]

| Model         | REG[E3h] | REG[E2h] |
|---------------|----------|----------|
| ER-TFTMC070-4 | 06h      | 1Ah      |

**REG[E4h] SDRAM Control Register (SDRCR)**

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-4 | Must keep 0.   | 0       | RW     |
| 3   | <b>Report Warning Condition</b><br>0: Disable or Clear warning flag 1:<br>Disable or Clear warning flag<br>Warning condition are memory read cycle close to SDRAM maximum address boundary (may over maximum address minus 512 bytes) or out of range or SDRAM bandwidth insufficient to fulfill panel's frame rate, then this warning event will be latched, user could check this bit to do some judgments. That warning flag could be cleared by set this bit as 0. | 0       | RW     |
| 2   | <b>SDRAM Timing Parameter Register Enable, SDR_PARAMEN</b><br>0: Disable Display RAM timing parameter registers 1:<br>Enable Display RAM timing parameter registers  | 0       | RW     |
| 1   | <b>Enter Power Saving Mode, SDR_PSAVING</b><br>0 to 1 transition will enter power saving mode 1<br>to 0 transition will exit power saving mode   | 0       | RW     |
| 0   | <b>Start SDRAM Initialization Procedure, SDR_INITDONE</b><br>0 to 1 transition will execute Display RAM initialization procedure. Read value „1“ means Display RAM is initialized and ready for access. Once it was written as 1, it cannot be rewrite as 0.<br>1 to 0 transition without have any operation.<br>"Write 1" will execute Display RAM initialization procedure.  | 0       | RW     |

Note: 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 | NA   | 0       | RO     |
| 3-0 | <b>Time from Load Mode Command to Active/Refresh Command (TMRD)</b><br>0000b: 1 SDRAM Clock<br>0001b: 2 SDRAM Clock<br>0010b: 3 SDRAM Clock<br>:<br>:<br>1111b: 16 SDRAM Clock | 2       | RW     |



REG[E1h] SDRAM Timing Parameter 2

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-4 | <b>Auto Refresh Period, TRFC</b><br>0h – Fh: 1 ~ 16 SDRAM Clock (As REG[E0h] bit[3:0])   | 8       | RW     |
| 3-0 | <b>Time of Exit SELF Refresh-to-ACTIVE Command (TXSR)</b><br>0h – Fh: 1 ~ 16 SDRAM Clock | 7       | RW     |

REG[E2h] SDRAM Timing Parameter 3

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-4 | <b>Time of Pre-charge Command Period (TRP, 15/20ns)</b><br>0h – Fh: 1 ~ 16 SDRAM Clock | 2       | RW     |
| 3-0 | <b>Time of WRITE Recovery Time (TWR)</b><br>0h – Fh: 1 ~ 16 SDRAM Clock                | 0       | RW     |

REG[E3h] SDRAM Timing Parameter 4

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-4 | <b>Delay Time of Active-to-Read/Write (TRCD)</b><br>0h – Fh: 1 ~ 16 SDRAM Clock | 2       | RW     |
| 3-0 | <b>Time of Active-to-Precharge (TRAS)</b><br>0h – Fh: 1 ~ 16 SDRAM Clock        | 6       | RW     |

### 5.14.12 I2C Master Register

REG[E5h-E6h] I2C Master Clock Prescaler Register (I2CMCK)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>I2C Master Clock Pre-scaler [15:0]</b><br>REG[E5h] mapping to I2CMCK[7:0].<br>REG[E6h] mapping to I2CMCK[15:8]. | 0       | RW     |

REG[E7h] I2C Master Transmit Register (I2CMTXR)

| Bit | Description                      | Default | Access |
|-----|----------------------------------|---------|--------|
| 7-0 | <b>I2C Master Transmit [7:0]</b> | 0       | RW     |

REG[E8h] I2C Master Receiver Register (I2CMRXR)

| Bit | Description                      | Default | Access |
|-----|----------------------------------|---------|--------|
| 7-0 | <b>I2C Master Receiver [7:0]</b> | 0       | RW     |

REG[E9h] I2C Master Command Register (I2CMCMD)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | <b>Start Command</b><br>Generate (repeated) start condition and be cleared by hardware Automatically<br><b>Note:</b> This bit is always read as 0. | 0       | RW     |
| 6   | <b>Stop Command</b><br>Write 1: Generate stop condition and be cleared by hardware.<br><b>Note:</b> This bit is always read as 0.                  | 0       | RW     |
| 5   | <b>Read Command</b><br>Write 1: Read form slave and be cleared by hardware automatically.<br><b>Note:</b> This bit is always read as 0.            | 0       | RW     |
| 4   | <b>Write Command</b><br>Write 1: Write to slave and be cleared by hardware automatically.<br><b>Note:</b> This bit is always read as 0.            | 0       | RW     |
| 3   | <b>Acknowledge Command</b><br>Write 0: Send ACK<br>Write 1: Send NACK<br><b>Note:</b> This bit is always read as 0.                                | 0       | RW     |
| 2-1 | NA   | 0       | RO     |

| Bit | Description                                    | Default | Access |
|-----|--|---------|--------|
| 0   | <b>Noise Filter</b><br>0: Disable<br>1: Enable | 0       | RW     |

REG[EAh] I2C Master Status Register (I2CMST)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Received Acknowledge from Slave</b><br>0: Acknowledge received.<br>1: No Acknowledge received.   | 0       | RO     |
| 6   | <b>I2C Bus is Busy</b><br>0: Idle. „0“ after STOP signal detected<br>1: Busy. „1“ after START signal detected   | 0       | RO     |
| 5-2 | NA  | 0       | RO     |
| 1   | <b>I2C Transfer in Progress</b><br>0: when transfer complete<br>1: when transferring data   | 0       | RO     |
| 0   | <b>Arbitration Lost State</b><br>When ER-TFTMC070-4 lost Arbitration, this bit will be set 1.<br><b>Note:</b> When a Stop signal is detected, but is not required, then it means the condition of arbitration loss. The ER-TFTMC070-4 I2C Master will drive SDA to 1, but the other Master will drive SDA to 0. | 0       | RO     |

### 5.14.13 GPIO Register

REG[F0h] GPIO-A Direction (GPIOAD)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>PIOA In/Out Control</b><br>0: Output.<br>1: Input. | FFh     | RW     |

REG[F1h] GPIO-A (GPIOA)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>GPIOA Data</b><br>Write: Output Data to GPIOA<br>Read: Read Data from GPIOA<br>GPIOA[7:0] are General Purpose I/O. These signals are shared with DB[15:8]. And they are available only when Host is in 8bits parallel or Serial mode. | NA      | RW     |

REG[F2h] GPIO-B (GPIOB)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-5 | NA  | NA      | NA     |
| 4   | <b>GPIOB[4] Data</b><br>Write: Output Data to GPIOB[4]<br>Read: Read Data from GPIOB[4]<br>GPIOB[4] is shared with KO[0], GPIB[4] is shared with KI[0]. | NA      | RW     |
| 3-0 | <b>GPIB[3:0] Data</b><br>Read: Read Data from GPIB[3:0]<br>GPIB[3:0] are shared with { A0, WR#, RD#, CS# }. They are available only in Serial Host I/F. | NA      | RO     |

REG[F3h] GPIO-C Direction (GPIOCD)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>GPIOC In/Out Control</b><br>0: Output<br>1: Input. | FFh     | RW     |

REG[F4h] GPIO-C (GPIOC)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>GPIOC[7] Data</b><br>Write: Output Data to GPIOC[7]<br>Read: Read Data from GPIOC[7]<br>GPIOC[7] is shared with PWM[0]. GPIOC is available only when PWM and SPI Master functions disabled.                                    | NA      | RW     |
| 6-5 | NA  | NA      | RW     |
| 4-0 | <b>GPIOC[4:0] Data</b><br>Write: Output Data to GPIOC[4:0]<br>Read: Read Data from GPIOC[4:0]<br>GPIOC[4:0] are shared with { SFCS1#, SFCS0#, SFDI, SFDO, SFCLK }. They are available when PWM and SPI Master functions disabled. | NA      | RW     |

REG[F5h] GPIO-D Direction (GPIODD)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>GPIOD In/Out Control</b><br>0: Output<br>1: Input | FFh     | RW     |

REG[F6h] GPIO-D (GPIOD)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7-0 | <b>GPIOD Data</b><br>Write: Output Data to GPIOD[7:0]<br>Read: Read Data from GPIOD[7:0]<br>GPIOD[7:0] are shared with PD[18, 2, 17, 16, 9, 8, 1, 0].<br>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-FAh]: Reserved.

#### 5.14.14 Keypad-scan Control Register

REG[FBh] Keypad-scan Control Register 1 (KSCR1)

| Bit | Description  | Default | Access |
|-----|--|---------|--------|
| 7   | Must set as 0.   | 0       | 0      |
| 6   | <b>Long Key Enable</b><br>1: Enable. Long key period is set by KSCR2 bit[4:2]. 0: Disable.   | 0       | RW     |
| 5-4 | <b>Short Key de-bounce Times</b><br>De-bounce times of keypad scan frequency.<br>00b: 4<br>01b: 8<br>10b: 16<br>11b: 32  | 0       | RW     |
| 3   | <b>Repeatable Key Enable</b><br>0: Disable Repeatable Key 1:<br>Enable Repeatable Key<br>i.e., if key is always pressed, then controller will repeat issue key interrupt in every short key de-bounce time (long key disable) or long key recognition time (long key enable) after user clear interrupt flag.  | 0       | RW     |
| 2-0 | <b>Row Scan Time</b><br><br><b><math>T_{KEYCLK} = (1 / F_{SYSCLK}) \times 2,048</math></b><br><br>000b: Row_Scan_Time = $T_{KEYCLK}$ 001b:<br>Row_Scan_Time = $T_{KEYCLK} \times 2$ 010b:<br>Row_Scan_Time = $T_{KEYCLK} \times 4$ 011b:<br>Row_Scan_Time = $T_{KEYCLK} \times 8$ 100b:<br>Row_Scan_Time = $T_{KEYCLK} \times 16$ 101b:<br>Row_Scan_Time = $T_{KEYCLK} \times 32$ 110b:<br>Row_Scan_Time = $T_{KEYCLK} \times 64$ 111b:<br>Row_Scan_Time = $T_{KEYCLK} \times 128$<br>ER-TFTMC070-4's Keypad-scan controller supports 5x5 keys. Total Key pads can time = Row Scan Time x 5. | 0       | RW     |

REG[FCh] Keypad-scan Controller Register 2 (KSCR2)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7   | <b>Keypad-scan Wakeup Enable</b><br>0: Keypad-scan Wakeup function is disabled. 1: Keypad-scan Wakeup function is enabled               | 0       | R/W    |
| 6   | <b>Key Released Interrupt Enable</b><br>0: Without interrupt event when all key released 1: Generate an interrupt when all key released | 0       | RW     |

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 5   | NA  | 0       | RO     |
| 4-2 | <b>Long Key Recognition Factor</b><br>It determines long key recognition time since short key was recognized. Value from 0 to 7.<br><a href="#">LongKey Recognition Time</a><br>$= \text{RowScanTime} \times 5 \times (\text{Long Key Recognition Factor} + 1) \times 1,024$  | 0       | RW     |
| 1-0 | <b>Numbers of Key Hit</b><br>00b: No key is pressed<br>01b: One key is pressed, REG[FDh] for the key code.<br>10b: Two keys are pressed, REG[FEh] for the 2nd key code. 11b: Three keys are pressed, REG[FFh] for the 3rd key code.<br>It will auto return to 0 if without any keys are pressed for a de-bounce time. | 0       | RO     |

REG[FDh] Keypad-scan Data Register (KSDR1)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Key Strobe Data1</b><br>The corresponding key code 1 that is pressed. It will auto return to FFh if without any keys are pressed for a de-bounce time. | TBD     | RO     |

REG[FEh] Keypad-scan Data Register (KSDR2)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Key Strobe Data2</b><br>The corresponding key code 2 that is pressed. It will auto return to FFh if without any keys are pressed for a de-bounce time. | TBD     | RO     |

REG[FFh] Keypad-scan Data Register (KSDR3)

| Bit | Description   | Default | Access |
|-----|---|---------|--------|
| 7-0 | <b>Key Strobe Data3</b><br>The corresponding key code 3 that is pressed. It will auto return to FFh if without any keys are pressed for a de-bounce time. | TBD     | RO     |

## 6. Inspection Criteria

### 6.1 Acceptable Quality Level

Each lot should satisfy the quality level defined as follows

| Partition | AQL  | Definition   |
|-----------|------|--|
| A. Major  | 0.4% | Functional defective as product                                    |
| B. Minor  | 1.5% | Satisfy all functions as product but not satisfy cosmetic standard |

### 6.2 Definition of Lot

One lot means the delivery quantity to customer at one time.

### 6.3 Condition of Cosmetic Inspection

- ◆ INSPECTION AND TEST

- FUNCTION TEST
- APPEARANCE INSPECTION
- PACKING SPECIFICATION

- ◆ INSPECTION CONDITION

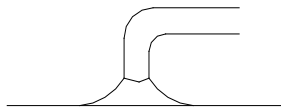
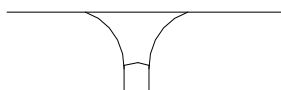
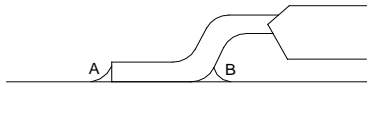
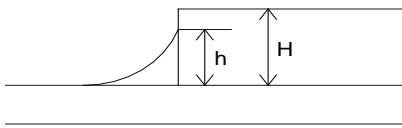
- Put under the lamp (20W) at a distance 100mm from
- Tilt upright 45 degree by the front (back) to inspect LCD appearance.

- ◆ AQL INSPECTION LEVEL

- SAMPLING METHOD: MIL-STD-105D
- SAMPLING PLAN: SINGLE
- MAJOR DEFECT: 0.4% (MAJOR)
- MINOR DEFECT: 1.5% (MINOR)
- GENERAL LEVEL: II/NORMAL



## 6.4 Module Cosmetic Criteria

| No. | Item                                 | Judgment Criterion  | Partition     |
|-----|--------------------------------------|---|---------------|
| 1   | Difference in Spec.                  | None allowed  | Major         |
| 2   | Pattern Peeling                      | No substrate pattern peeling and floating   | Major         |
| 3   | Soldering Defects                    | No soldering missing  | Major         |
|     |                                      | No soldering bridge   | Major         |
|     |                                      | No cold soldering   | Minor         |
| 4   | Resist Flaw on Substrate             | Invisible copper foil( $\phi$ 0.5mm or more)on substrate pattern  | Minor         |
| 5   | Accretion of Metallic Foreign Matter | No soldering dust   | Minor         |
|     |                                      | No accretion of metallic foreign matters(Not exceed $\phi$ 0.2mm)   |               |
| 6   | Stain                                | No stain to spoil cosmetic badly  | Minor         |
| 7   | Plate Discoloring                    | No plate fading, rusting and discoloring  | Minor         |
| 8   | Solder Amount<br>1.Lead Parts        | <p>a. Soldering side of PCB<br/>Solder to form a' Filet' all around t<br/>Solder should not hide the lead form</p>  <p>b.Components side<br/>(In case of 'Through Hole PCB' )<br/>Solder to reach the Components side of PCB</p>  | Minor         |
|     | 2.Flat Packages                      | <p>Either 'toe' (A) or 'heel' (B) of the lead to be covered by Filet'</p>  <p>Lead form to be assume over solder.</p>   | Minor         |
|     | 3.Chips                              | <p><math>(3/2) H \geq h \geq (1/2)H</math></p>    | Minor         |
| 9   | Backlight Defects                    | <p>1.Light fails or flickers.(Major)</p> <p>2. Color and luminance do not correspond to specifications. (Major)</p> <p>3.Exceeds standards for display' s blemishes, foreign matter, dark lines or scratches.(Minor)</p>  | See list<br>← |

|    |                   |   |               |
|----|-------------------|---|---------------|
| 10 | PCB Defects       | <p>Oxidation or contamination on connectors.*</p> <p>2. Wrong parts, missing parts, or parts not in specification.*</p> <p>3. Jumpers set incorrectly. (Minor)</p> <p>4. Solder (if any) on bezel, LED pad, zebra pad, or screw hole pad is not smooth. (Minor)</p> <p>*Minor if display functions correctly. Major if the display fails.</p> | See list<br>← |
| 11 | Soldering Defects | <p>1. Unmelted solder paste.</p> <p>2. Cold solder joints, missing solder connections, or oxidation.*</p> <p>3. Solder bridges causing short circuits.*</p> <p>4. Residue or solder balls.</p> <p>5. Solder flux is black or brown.</p> <p>*Minor if display functions correctly. Major if the display fails.</p>                             | Minor         |

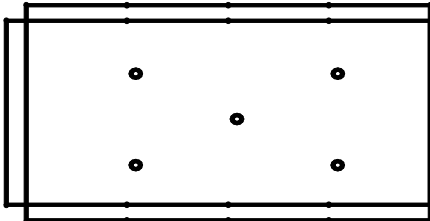
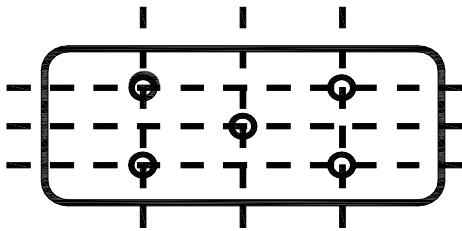
## 6.5 Screen Cosmetic Criteria (Non-Operating)

| No. | Defect               | Judgment Criterion  | Partition |
|-----|----------------------|---|-----------|
| 1   | Spots                | In accordance with Screen Cosmetic Criteria (Operating) No.1.   | Minor     |
| 2   | Lines                | In accordance with Screen Cosmetic Criteria (Operation) No.2.   | Minor     |
| 3   | Bubbles in Polarizer |   | Minor     |
|     |                      | Size: d mm  |           |
|     |                      | Acceptable Qty in active area   |           |
|     |                      | Disregard   |           |
|     |                      |   |           |
|     |                      | d ≤ 0.3   |           |
|     |                      | 0.3 < d ≤ 1.0   | 3         |
|     |                      | 1.0 < d ≤ 1.5   | 1         |
|     |                      | 1.5 < d   | 0         |
| 4   | Scratch              | In accordance with spots and lines operating cosmetic criteria, When the light reflects on the panel surface, the scratches are not to be remarkable. | Minor     |
| 5   | Allowable density    | Above defects should be separated more than 30mm each other.  | Minor     |
| 6   | Coloration           | Not to be noticeable coloration in the viewing area of the LCD panels.<br>Back-lit type should be judged with back-lit on state only.                 | Minor     |
| 7   | Contamination        | Not to be noticeable.   | Minor     |

## 6.6 Screen Cosmetic Criteria (Operating)

| No.  | Defect                        | Judgment Criterion   | Partition |                               |
|--|-------------------------------|--|-----------|-------------------------------|
| 1  | Spots                         | A) Clear   | Minor     |                               |
|  |                               | Size:d mm  |           | Acceptable Qty in active area |
|  |                               | d≤0.1  |           | Disregard                     |
|  |                               | 0.1<d≤0.2  |           | 6                             |
|  |                               | 0.2<d≤0.3  |           | 2                             |
|  |                               | 0.3<d  |           | 0                             |
| Note: Including pin holes and defective dots which must be within one pixel Size.<br>Unclear |                               |  |           |                               |
| Size:d mm  | Acceptable Qty in active area |  |           |                               |
| d≤0.2  | Disregard                     |  |           |                               |
| 0.2<d≤0.5  | 6                             |  |           |                               |
| 0.5<d≤0.7  | 2                             |  |           |                               |
| 0.7<d  | 0                             |  |           |                               |
| 2  | Lines                         | A) Clear   | Minor     |                               |
|  |                               | <div><div><div><div><div></div><div>L 5.0</div></div><div><div></div><div>2.0</div></div></div><div><div><div></div><div>∞</div></div><div><div></div><div>(6)</div></div><div><div></div><div>(0)</div></div></div><div><div>0.02</div><div>0.05</div><div>0.1</div><div>W</div></div><div>See No.1</div></div></div> <div>Note: () – Acceptable Qty in active area<br/>L - Length (mm)<br/>W -Width(mm)<br/>∞-Disregard</div> <div>B) Unclear</div> <div><div><div><div><div></div><div>L 10.0</div></div><div><div></div><div>2.0</div></div></div><div><div><div></div><div>∞</div></div><div><div></div><div>(6)</div></div><div><div></div><div>(0)</div></div></div><div><div>0.05</div><div>0.3</div><div>0.5</div><div>W</div></div><div>See No.1</div></div></div> |           |                               |

Clear’ = The shade and size are not changed by Vo.  
Unclear’ = The shade and size are changed by Vo.

| No. | Defect                                  | Judgment Criterion   | Partition |
|-----|---|--|-----------|
| 3   | Rubbing line                            | Not to be noticeable.  | Minor     |
| 4   | Allowable density                       | Above defects should be separated more than 10mm each other.   | Minor     |
| 5   | Rainbow                                 | Not to be noticeable.  | Minor     |
| 6   | Dot size                                | To be 95%~105%of the dot size (Typ.) in drawing.<br>Partial defects of each dot (ex.pin-hole) should be treated as spot.<br>(see Screen Cosmetic Criteria (Operating) No.1)  | Minor     |
| 7   | Brightness<br>(only back-lit<br>Module) | Brightness Uniformity must be $B_{MAX}/B_{MIN} \leq 2$<br>- $B_{MAX}$ : Max.value by measure in 5 points<br>- $B_{MIN}$ : Min.value by measure in 5 points<br>Divide active area into 4 vertically and horizontally.<br>Measure 5 points shown in the following figure.<br>   | Minor     |
| 8   | Contrast<br>Uniformity                  | Contrast Uniformity must be $B_{MAX}/B_{MIN} \leq 2$<br>Measure 5 points shown in the following figure.<br>Dashed lines divide active area into 4 vertically and horizontally.<br>Measuring points are located at the inter-sections of dashed line.<br><br><br>Note: $B_{MAX}$ – Max.value by measure in 5 points.<br>$B_{MIN}$ – Min.value by measure in 5 points.<br>O – Measuring points in $\phi 10mm$ . | Minor     |

Note:

(1) Size:  $d = (\text{long length} + \text{short length})/2$

(2) The limit samples for each item have priority.

(3) Complexed defects are defined item by item, but if the number of defects is defined in above table, the total number should not exceed 10.

(4) In case of 'concentration', even the spots or the lines of 'disregarded' size should not be allowed. Following three situations should be treated as 'concentration'.

-7 or over defects in circle of  $\phi 5mm$ .

-10 or over defects in circle of  $\phi 10mm$

-20 or over defects in circle of  $\phi 20mm$

## 7. Precautions for Using

### 7.1 Handling Precautions

- ♦ This device is susceptible to Electro-Static Discharge (ESD) damage. Observe Anti-Static precautions.
- ♦ Eastrising display panel is made of glass. Do not subject it to a mechanical shock by dropping it or impact.
- ♦ If Eastrising display panel is damaged and the liquid crystal substance leaks out, be sure not to get any in your mouth. If the substance contacts your skin or clothes, wash it off using soap and water.
- ♦ Do not apply excessive force to the Eastrising display surface or the adjoining areas since this may cause the color tone to vary.
- ♦ The polarizer covering the Eastrising display surface of the LCD module is soft and easily scratched. Handle this polarizer carefully.
- ♦ If Eastrising display surface becomes contaminated, breathe on the surface and gently wipe it with a soft dry cloth. If it is heavily contaminated, moisten cloth with one of the following Isopropyl or alcohol.
- ♦ Solvents other than those above-mentioned may damage the polarizer. Especially, do not use the Water.
- ♦ Exercise care to minimize corrosion of the electrode. Corrosion of the electrodes is accelerated by water droplets, moisture condensation or a current flow in a high-humidity environment.
- ♦ Install the Eastrising LCD Module by using the mounting holes. When mounting the LCD module make sure it is free of twisting, warping and distortion. In particular, do not forcibly pull or bend the cable or the backlight cable.
- ♦ Do not attempt to disassemble or process Eastrising LCD module.
- ♦ NC terminal should be open. Do not connect anything.
- ♦ If the logic circuit power is off, do not apply the input signals.
- ♦ To prevent destruction of the elements by static electricity, be careful to maintain an optimum work environment.
  - Be sure to ground the body when handling Eastrising LCD modules.
  - Tools required for assembling, such as soldering irons, must be properly grounded.
- To reduce the amount of static electricity generated, do not conduct assembling and other work under dry conditions.
- The LCD module is coated with a film to protect the display surface. Exercise care when peeling off this protective film since static electricity may be generated.

### 7.2 Power Supply Precautions

- ♦ Identify and, at all times, observe absolute maximum ratings for both logic and LC drivers. Note that there is some variance between models.
- ♦ Prevent the application of reverse polarity to VDD and VSS, however briefly.
- ♦ Use a clean power source free from transients. Power-up conditions are occasionally jolting and may exceed the maximum ratings of Eastrising modules.
- ♦ The VDD power of Eastrising module should also supply the power to all devices that may access the display. Don't allow the data bus to be driven when the logic supply to the module is turned off.

### 7.3 Operating Precautions

- ♦ DO NOT plug or unplug Eastrising module when the system is powered up.
- ♦ Minimize the cable length between Eastrising module and host MPU.
- ♦ For models with backlights, do not disable the backlight by interrupting the HV line. Unload inverters produce voltage extremes that may arc within a cable or at the display.
- ♦ Operate Eastrising module within the limits of the modules temperature specifications.

### 7.4 Mechanical/Environmental Precautions

- ♦ Improper soldering is the major cause of module difficulty. Use of flux cleaner is not recommended as they may seep under the electrometric connection and cause display failure.
- ♦ Mount Eastrising module so that it is free from torque and mechanical stress.
- ♦ Surface of the LCD panel should not be touched or scratched. The display front surface is an easily scratched, plastic polarizer. Avoid contact and clean only when necessary with soft, absorbent cotton dampened with petroleum benzene.
- ♦ Always employ anti-static procedure while handling Eastrising module.
- ♦ Prevent moisture build-up upon the module and observe the environmental constraints for storage tem
- ♦ Do not store in direct sunlight
- ♦ If leakage of the liquid crystal material should occur, avoid contact with this material, particularly ingestion. If the body or clothing becomes contaminated by the liquid crystal material, wash thoroughly with water and soap.

### 7.5 Storage Precautions

When storing the LCD modules, avoid exposure to direct sunlight or to the light of fluorescent lamps.

Keep Eastrising modules in bags (avoid high temperature / high humidity and low temperatures below 0 °C).

Whenever possible, Eastrising LCD modules should be stored in the same conditions in which they were shipped from our company.

### 7.6 Others

Liquid crystals solidify under low temperature (below the storage temperature range) leading to defective orientation or the generation of air bubbles (black or white). Air bubbles may also be generated if the module is subject to a low temperature.

If Eastrising LCD modules have been operating for a long time showing the same display patterns, the display patterns may remain on the screen as ghost images and a slight contrast irregularity may also appear. A normal operating status can be regained by suspending use for some time. It should be noted that this phenomenon does not adversely affect performance reliability.

To minimize the performance degradation of the LCD modules resulting from destruction caused by static electricity etc., exercise care to avoid holding the following sections when handling the modules.

- Exposed area of the printed circuit board.
- Terminal electrode sections.

## 8. Using LCD Modules

### 8.1 Liquid Crystal Display Modules

Eastrising LCD is composed of glass and polarizer. Pay attention to the following items when handling.

- ♦ Please keep the temperature within specified range for use and storage. Polarization degradation, bubble generation or polarizer peel-off may occur with high temperature and high humidity.
- ♦ Do not touch, push or rub the exposed polarizers with anything harder than an HB pencil lead (glass, tweezers, etc.).
- ♦ N-hexane is recommended for cleaning the adhesives used to attach front/rear polarizers and reflectors made of organic substances which will be damaged by chemicals such as acetone, toluene, ethanol and isopropyl alcohol.
- ♦ When Eastrising display surface becomes dusty, wipe gently with absorbent cotton or other soft material like chamois soaked in petroleum benzin. Do not scrub hard to avoid damaging the display surface.
- ♦ Wipe off saliva or water drops immediately, contact with water over a long period of time may cause deformation or color fading.
- ♦ Avoid contacting oil and fats.
- ♦ Condensation on the surface and contact with terminals due to cold will damage, stain or dirty the polarizers. After products are tested at low temperature they must be warmed up in a container before coming in contact with room temperature air.
- ♦ Do not put or attach anything on Eastrising display area to avoid leaving marks on.
- ♦ Do not touch the display with bare hands. This will stain the display area and degrade insulation between terminals (some cosmetics are determined to the polarizers).
- ♦ As glass is fragile. It tends to become or chipped during handling especially on the edges. Please avoid dropping.

### 8.2 Installing LCD Modules

- ♦ Cover the surface with a transparent protective plate to protect the polarizer and LC cell.
- ♦ When assembling the LCM into other equipment, the spacer to the bit between the LCM and the fitting plate should have enough height to avoid causing stress to the module surface, refer to the individual specifications for measurements. The measurement tolerance should be  $\pm 0.1\text{mm}$ .

### 8.3 Precaution for Handling LCD Modules

Since Eastrising LCM has been assembled and adjusted with a high degree of precision; avoid applying excessive shocks to the module or making any alterations or modifications to it.

- ♦ Do not alter, modify or change the shape of the tab on the metal frame.
- ♦ Do not make extra holes on the printed circuit board, modify its shape or change the positions of components to be attached.
- ♦ Do not damage or modify the pattern writing on the printed circuit board.
- ♦ Absolutely do not modify the zebra rubber strip (conductive rubber) or heat seal connector.
- ♦ Except for soldering the interface, do not make any alterations or modifications with a soldering iron.
- ♦ Do not drop, bend or twist Eastrising LCM.

## 8.4 Electro-Static Discharge Control

Since this module uses a CMOS LSI, the same careful attention should be paid to electrostatic discharge as for an ordinary CMOS IC.

- ♦ Make certain that you are grounded when handling LCM.
- ♦ Before remove LCM from its packing case or incorporating it into a set, be sure the module and your body have the same electric potential.
- ♦ When soldering the terminal of LCM, make certain the AC power source for the soldering iron does not leak.
- ♦ When using an electric screwdriver to attach LCM, the screwdriver should be of ground potentiality to minimize as much as possible any transmission of electromagnetic waves produced sparks coming from the commutator of the motor.
- ♦ As far as possible make the electric potential of your work clothes and that of the work bench the ground potential.
- ♦ To reduce the generation of static electricity be careful that the air in the work is not too dried. A relative humidity of 50%-60% is recommended.

## 8.5 Precaution for Soldering to Eastrising LCM

- ♦ Observe the following when soldering lead wire, connector cable and etc. to the LCM.
  - Soldering iron temperature :  $280^{\circ}\text{C} \pm 10^{\circ}\text{C}$
  - Soldering time: 3-4 sec.
  - Solder: eutectic solder.

If soldering flux is used, be sure to remove any remaining flux after finishing to soldering operation. (This does not apply in the case of a non-halogen type of flux.) It is recommended that you protect the LCD surface with a cover during soldering to prevent any damage due to flux spatters.

- ♦ When soldering the electroluminescent panel and PC board, the panel and board should not be detached more than three times. This maximum number is determined by the temperature and time conditions mentioned above, though there may be some variance depending on the temperature of the soldering iron.
- ♦ When remove the electroluminescent panel from the PC board, be sure the solder has completely melted, the soldered pad on the PCs board could be damaged.

## 8.6 Precaution for Operation

- ♦ Driving the Eastrising LCD in the voltage above the limit shortens its life.
- ♦ Response time is greatly delayed at temperature below the operating temperature range. However, this does not mean the LCD will be out of the order. It will recover when it returns to the specified temperature range.
- ♦ If Eastrising display area is pushed hard during operation, the display will become abnormal. However, it will return to normal if it is turned off and then back on.
- ♦ Condensation on terminals can cause an electrochemical reaction disrupting the terminal circuit. Therefore, it must be used under the relative condition of  $40^{\circ}\text{C}$ , 50% RH.
- ♦ When turning the power on, input each signal after the positive/negative voltage becomes stable.



## 8.7 Limited Warranty

Unless agreed between Eastrising and customer, Eastrising will replace or repair any of its LCD modules which are found to be functionally defective when inspected in accordance with Eastrising LCD acceptance standards (copies available upon request) for a period of one year from date of shipments. Cosmetic/visual defects must be returned to Eastrising within 90 days of shipment. Confirmation of such date shall be based on freight documents. The warranty liability of Eastrising limited to repair and/or replacement on the terms set forth above. Eastrising will not be responsible for any subsequent or consequential events.

## 8.8 Return Policy

No warranty can be granted if the precautions stated above have been disregarded. The typical examples of violations are:

- Broken LCD glass.
- PCB eyelet damaged or modified.
- PCB conductors damaged.
- Circuit modified in any way, including addition of components.
- PCB tampered with by grinding, engraving or painting varnish.
- Soldering to or modifying the bezel in any manner.

Module repairs will be invoiced to the customer upon mutual agreement. Modules must be returned with sufficient description of the failures or defects. Any connectors or cable installed by the customer must be removed completely without damaging the PCB eyelet' s, conductors and terminals.

## 9. Image Sticking

### 9.1 What is Image Sticking?

If you remain a fixed image on LCD Display for a long period of time, you may experience a phenomenon called Image Sticking. Image Sticking - sometimes also called "image retention" or "ghosting" - is a phenomenon where a faint outline of a previously displayed image remains visible on the screen when the image is changed. It can occur at variable levels of intensity depending on the specific image makeup, as well as the amount of time the core image elements are allowed to remain unchanged on the screen. In POS applications, for example, a button menu which remains fixed, or in which the "frame" elements (core image) remain fixed and the buttons may change, may be susceptible to image sticking. It is important to note that if the screen is used exclusively for this application, the user may never notice this phenomenon since the screen never displays other content. It is only when an image other than the "retained" image is shown on the screen that this issue becomes evident. Image sticking is different than the "burn-in" effect commonly associated with phosphor based devices.

### 9.2 What causes Image Sticking?

Image sticking is an intrinsic behavior of LCD displays due to the susceptibility to polarization of the interior materials (liquid crystals) when used under static, charged conditions (continuously displaying the same image). The individual liquid crystals in an LCD panel have unique electrical properties. Displaying a fixed pattern - such as the POS menu described above - over prolonged periods can cause a parasitic charge build-up (polarization) within the liquid crystals which affects the crystals' optical properties and ultimately prevents the liquid crystal from returning to its normal, relaxed state when the pattern is finally changed. This effect takes place at a cellular level within the LCD, and the effect can cause charged crystal alignment at the bottom or top of a crystal cell in the "z" axis, or even crystal migration to the edges of a cell, again based on their polarity. These conditions can cause image sticking over an entire area, or at boundaries of distinct color change respectively. In either case, when the liquid crystals in the pixels and sub-pixels utilized to display the static image are polarized such that they can not return fully to their "relaxed" state upon deactivation, the result is a faint, visible, retained image on the panel upon presentation of a new, different image. The actual rate of image retention depends on variation factors such as the specific image, how long it is displayed unchanged, the temperature within the panel and even the specific panel brand due to manufacturing differences amongst panel manufacturers.

### 9.3 How to Avoid Image Sticking?

- Try not to operate the LCD with a “fixed” image on the screen for more than 2 hours.
- If you are operating the monitor in an elevated temperature environment and with a displayed image which is contrary to the recommendations in “For Software Developers” below, image stick can occur in as little as 30 minutes. Adjust your screen saver settings accordingly.
- Power down the unit during prolonged periods of inactivity such as the hours a store is closed or a shift during which the piece of equipment isn’t used.
- Use a screensaver with a black or medium gray background that is automatically set to come on if the device is inactive for more than 5-10 minutes.
- Avoid placing the monitor in poorly ventilated areas or in areas that will create excess heat around the monitor for software developers.
- In defining the icons, buttons, or windows in the screen, try to utilize block patterns instead of distinct lines as borders for dividing the display into distinct areas.
- If it is necessary to display a static image, try to use colors that are symmetric to the middle grey level at the boundary of two different colors, and slightly shift the borders line once in a while.
- Try to utilize medium gray hues for those areas that will have prolonged display times or remain static as other menu elements change.

### 9.4 How to Fix the Image Sticking?

Unlike the usually irreversible “burn-in” effects commonly associated with direct view phosphor display devices such as CRTs, an image retained on an LCD display can be reversed – often to a point of total invisibility. However, the severity of the underlying causes (as described above) of the image retained on a specific display, as well as the variation factors (see “For Software Developers” above) under which the retained image was created, will dictate the final level of retention reversal. One way to erase a retained image on a panel is to run the screen (monitor “on” ) in an “all black” pattern for 4-6 hours. It is also helpful to do this in an elevated temperature environment of approximately 35° to 50°C. Again, utilizing a dynamic screen saver with an all black background during prolonged idle display periods is a good way to avoid image retention issues.

### 9.5 Is Image Sticking Covered by Eastrising RMA Warranty?

Image sticking is a phenomenon inherent to LCD Display technology itself, and as such, the occurrence of this “ghosting” effect is considered normal operation by the manufacturers of the LCD display modules which are integrated into today’s monitor solutions. Eastrising does not warrant any display against the occurrence of image sticking. We strongly advise that you follow the operating recommendations listed above to avoid the occurrence of this phenomenon.

That's the end of the datasheet.