RAiO

RA8875

Character/Graphic

TFT LCD Controller

Application Note

Version 1.3

July 11, 2011
<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>May, 03, 2011</td>
<td>First Release</td>
</tr>
<tr>
<td>1.1</td>
<td>May, 09, 2011</td>
<td>Schematic 2 update, initial code update</td>
</tr>
<tr>
<td>1.2</td>
<td>June, 22, 2011</td>
<td>Schematic 2 update</td>
</tr>
<tr>
<td>1.3</td>
<td>July, 11, 2011</td>
<td>Add chapter 6 appendix</td>
</tr>
<tr>
<td>Chapter</td>
<td>Contents</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>1. Application Circuit</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>2. Initial Code</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>3. Display on Sequence</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>4. Sleep Mode Sequence</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>5. Display RAM pure data write example</td>
<td></td>
<td>18</td>
</tr>
<tr>
<td>6. Appendix</td>
<td></td>
<td>19</td>
</tr>
</tbody>
</table>
1. Application Circuit

Figure 1-1
Figure 1-2

GA8875

(RA8875)

RAiO TECHNOLOGY INC.

5/19

www.raio.com.tw
2. Initial Code

```c
void RA8875_PLL_ini(void)
{
    #ifdef P320x240
        LCD_CmdWrite(0x88);
        LCD_DataWrite(0x0a);
        Delay1ms(1);
        LCD_CmdWrite(0x89);
        LCD_DataWrite(0x02);
        Delay1ms(1);
    #endif

    #ifdef P480x272
        LCD_CmdWrite(0x88);
        LCD_DataWrite(0x0a);
        Delay1ms(1);
        LCD_CmdWrite(0x89);
        LCD_DataWrite(0x02);
        Delay1ms(1);
    #endif

    #ifdef P640x480
        LCD_CmdWrite(0x88);
        LCD_DataWrite(0x0b);
        Delay1ms(1);
        LCD_CmdWrite(0x89);
        LCD_DataWrite(0x02);
        Delay1ms(1);
    #endif

    #ifdef P800x480
        LCD_CmdWrite(0x88);
        LCD_DataWrite(0x0b);
        Delay1ms(1);
        LCD_CmdWrite(0x89);
        LCD_DataWrite(0x02);
        Delay1ms(1);
    #endif
}
```
void LCD_Initial(void)
{
    RA8875_PLL_ini();
    LCD_CmdWrite(0x10);  //SYSR   bit[4:3]= 00 256 color  bit[2:1]=  00 8bit MPU interface
    LCD_DataWrite(0x0c);   // if 8bit MCU interface   and 65k color display
    #ifdef P320x240
//============== Display Window320x240 ==================
    LCD_CmdWrite(0x04);     //set PCLK invers
    LCD_DataWrite(0x03);
    Delay1ms(1);
    LCD_CmdWrite(0x14); //HDWR//Horizontal Display Width Setting Bit[6:0]
    LCD_DataWrite(0x27);//Horizontal display width(pixels) = (HDWR + 1)*8
    LCD_CmdWrite(0x15); //Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
    LCD_DataWrite(0x00);//Horizontal Non- Display Period Fine Tuning(HNDFT) [3:0]
    LCD_CmdWrite(0x16); //HNDR//Horizontal Non-Display Period Bit[4:0]
    LCD_DataWrite(0x05);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
    LCD_CmdWrite(0x17); //HSTR//HSYNC Start Position[4:0]
    LCD_DataWrite(0xef);//Vertical pixels = VDHR + 1
    LCD_CmdWrite(0x1A); //VDHR1 //Vertical Display Height Bit [8]
    LCD_CmdWrite(0x00);//Vertical pixels = VDHR + 1
    LCD_CmdWrite(0x1B); //VNDR0 //Vertical Non-Display Period Bit [7:0]
    LCD_CmdWrite(0x05);//Vertical Non-Display area = (VNDR + 1)
    LCD_CmdWrite(0x1C); //VNDR1 //Vertical Non-Display Period Bit [8]
    LCD_CmdWrite(0x00); //Vertical Non-Display area = (VNDR + 1)
    LCD_CmdWrite(0x1D); //VSTR0 //VSYNC Start Position[7:0]
    LCD_CmdWrite(0xe0); //VSYNC Start Position(PCLK) = (VSTR + 1)
    LCD_CmdWrite(0x1E); //VSTR1 //VSYNC Start Position[8]
    LCD_CmdWrite(0x00); //VSYNC Start Position(PCLK) = (VSTR + 1)
    LCD_CmdWrite(0x1F); //VPWR //VSYNC Polarity ,VSYNC Pulse Width[6:0]
RA8875

LCD_DataWrite(0x02); // VSYNC Pulse Width (PCLK) = (VPWR + 1)

// Active window set
// setting active window X

LCD_CmdWrite(0x30); // Horizontal Start Point 0 of Active Window (HSAW0)
LCD_DataWrite(0x00); // Horizontal Start Point of Active Window [7:0]
LCD_CmdWrite(0x31); // Horizontal Start Point 1 of Active Window (HSAW1)
LCD_DataWrite(0x00); // Horizontal Start Point of Active Window [9:8]
LCD_CmdWrite(0x34); // Horizontal End Point 0 of Active Window (HEAW0)
LCD_DataWrite(0x3F); // Horizontal End Point of Active Window [7:0]
LCD_CmdWrite(0x35); // Horizontal End Point 1 of Active Window (HEAW1)
LCD_DataWrite(0x01); // Horizontal End Point of Active Window [9:8]

// setting active window Y

LCD_CmdWrite(0x32); // Vertical Start Point 0 of Active Window (VSAW0)
LCD_DataWrite(0x00); // Vertical Start Point of Active Window [7:0]
LCD_CmdWrite(0x33); // Vertical Start Point 1 of Active Window (VSAW1)
LCD_DataWrite(0x00); // Vertical Start Point of Active Window [8]
LCD_CmdWrite(0x36); // Vertical End Point of Active Window 0 (VEAW0)
LCD_DataWrite(0xef); // Vertical End Point of Active Window [7:0]
LCD_CmdWrite(0x37); // Vertical End Point of Active Window 1 (VEAW1)
LCD_DataWrite(0x00); // Vertical End Point of Active Window [8]

#endif

#ifdef P480x272

LCD_CmdWrite(0x04); // set PCLK invers
LCD_DataWrite(0x82);
Delay1ms(1);

// Horizontal set

LCD_CmdWrite(0x14); // HDWR // Horizontal Display Width Setting Bit [6:0]
LCD_DataWrite(0x3B); // Horizontal display width (pixels) = (HDWR + 1)*8
LCD_CmdWrite(0x15); // HNSWR // Horizontal Non-Display Period Fine Tuning Option Register (DNDFTR)
LCD_DataWrite(0x00); // Horizontal Non-Display Period Fine Tuning (HNDFT) [3:0]
LCD_CmdWrite(0x16); // HNDR // Horizontal Non-Display Period Bit [4:0]
LCD_DataWrite(0x01); // Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
LCD_CmdWrite(0x17); // HSYNC Start Position [4:0]
LCD_DataWrite(0x00); // HSYNC Start Position (PCLK) = (HSTR + 1)*8
LCD_CmdWrite(0x18); // HPWR // HSYNC Polarity, The period width of HSYNC.
// Vertical set

// Vertical display height set
LCD_CmdWrite(0x19); // VDHR0 // Vertical Display Height Bit [7:0]
LCD_DataWrite(0x0f); // Vertical pixels = VDHR + 1
LCD_CmdWrite(0x1a); // VDHR1 // Vertical Display Height Bit [8]
LCD_DataWrite(0x01); // Vertical pixels = VDHR + 1

// Vertical non-display period set
LCD_CmdWrite(0x1b); // VNDR0 // Vertical Non-Display Period Bit [7:0]
LCD_DataWrite(0x02); // Vertical pixels = VNDR + 1
LCD_CmdWrite(0x1c); // VNDR1 // Vertical Non-Display Period Bit [8]

// Vertical non-display area set
LCD_CmdWrite(0x00); // Vertical Non-Display Area = (VNDR + 1)

// VSYNC start position set
LCD_CmdWrite(0x1d); // VSYNC Start Position [7:0]
LCD_DataWrite(0x07); // VSYNC Start Position (PCLK) = (VSTR + 1)
LCD_CmdWrite(0x1e); // VSYNC Start Position [8]
LCD_DataWrite(0x00); // VSYNC Start Position (PCLK) = (VSTR + 1)

// VSYNC polarity and pulse width set
LCD_CmdWrite(0x1f); // VPWR // VSYNC Polarity, VSYNC Pulse Width [6:0]
LCD_DataWrite(0x09); // VSYNC Pulse Width (PCLK) = (VPWR + 1)

// Active window set

// Setting active window X
LCD_CmdWrite(0x30); // Horizontal Start Point 0 of Active Window (HSAW0)
LCD_DataWrite(0x00); // Horizontal Start Point of Active Window [7:0]
LCD_CmdWrite(0x31); // Horizontal Start Point 1 of Active Window (HSAW1)
LCD_DataWrite(0x00); // Horizontal Start Point of Active Window [9:8]
LCD_CmdWrite(0x34); // Horizontal End Point 0 of Active Window (HEAW0)
LCD_DataWrite(0xDF); // Horizontal End Point of Active Window [7:0]
LCD_CmdWrite(0x35); // Horizontal End Point 1 of Active Window (HEAW1)

// Setting active window Y

// Vertical start point set
LCD_CmdWrite(0x32); // Vertical Start Point 0 of Active Window (VSAW0)
LCD_DataWrite(0x00); // Vertical Start Point of Active Window [7:0]
LCD_CmdWrite(0x33); // Vertical Start Point 1 of Active Window (VSAW1)

// Vertical end point set
LCD_CmdWrite(0x00); // Vertical End Point of Active Window 0 (VEAW0)
LCD_CmdWrite(0x0F); // Vertical End Point of Active Window [7:0]
LCD_CmdWrite(0x01); // Vertical End Point of Active Window 1 (VEAW1)

#if defined

LCобр DataWrite(0x05); // HSYNC Width [4:0] HSYNC Pulse width (PCLK) = (HPWR + 1)*8

#endif
#ifdef P640x480

//============ Display Window 640x480 ==============

 LCD_CmdWrite(0x04); //PCLK inverse
 LCD_DataWrite(0x01);
 Delay1ms(1);

//Horizontal set
 LCD_CmdWrite(0x14); //HDWR //Horizontal Display Width Setting Bit[6:0]
 LCD_DataWrite(0x4F); //Horizontal display width(pixels) = (HDWR + 1)*8
 LCD_CmdWrite(0x15); //HNDFTR //Horizontal Non-Display Period Fine Tuning Option Register(HNDFTR)
 LCD_DataWrite(0x05); //Horizontal Non-Display Period Fine Tuning(HNDF)(3:0)
 LCD_CmdWrite(0x16); //HSTR //Horizontal Non-Display Period Bit[4:0]
 LCD_DataWrite(0x00); //Horizontal Non-Display Period (pixels) = (HSTR + 1)*8

//Vertical set
 LCD_CmdWrite(0x19); //VDHR0 //Vertical Display Height Bit [7:0] 
 LCD_DataWrite(0xdf); //Vertical pixels = VDHR + 1
 LCD_CmdWrite(0x1A); //VDHR1 //Vertical Display Height Bit [8]
 LCD_DataWrite(0x01); //Vertical pixels = VDHR + 1
 LCD_CmdWrite(0x1B); //VNDR0 //Vertical Non-Display Period Bit [7:0]
 LCD_DataWrite(0x0A); //Vertical Non-Display area = (VNDR + 1)
 LCD_CmdWrite(0x1C); //VNDR1 //Vertical Non-Display Period Bit [8]
 LCD_DataWrite(0x00); //Vertical Non-Display area = (VNDR + 1)

//Active window set

//setting active window X
 LCD_CmdWrite(0x30); //Horizontal Start Point 0 of Active Window (HSAW0)
 LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [7:0]
 LCD_CmdWrite(0x31); //Horizontal Start Point 1 of Active Window (HSAW1)
 LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [9:8]
 LCD_CmdWrite(0x34); //Horizontal End Point 0 of Active Window (HEAW0)

//setting active window Y
 LCD_CmdWrite(0x32); //Vertical Start Point 0 of Active Window (VSAY0)
 LCD_DataWrite(0x00); //Vertical Start Point of Active Window [7:0]
 LCD_CmdWrite(0x33); //Vertical Start Point 1 of Active Window (VSAY1)
 LCD_DataWrite(0x00); //Vertical Start Point of Active Window [9:8]
 LCD_CmdWrite(0x36); //Vertical End Point 0 of Active Window (VSAE0)

//setting active window Size
 LCD_CmdWrite(0x38); //Size of Active Window (HSAS)
 LCD_DataWrite(0x4F); //Size of Active Window (pixels) = (HDWR + 1)*8
 LCD_CmdWrite(0x39); //Size of Active Window (VNAS)
 LCD_DataWrite(0x01); //Size of Active Window (pixels) = (HSTR + 1)*8
LCD_DataWrite(0x7f); //Horizontal End Point of Active Window [7:0]
LCD_CmdWrite(0x35); //Horizontal End Point 1 of Active Window (HEAW1)
LCD_DataWrite(0x02); //Horizontal End Point of Active Window [9:8]

//setting active window Y
LCD_CmdWrite(0x32); //Vertical Start Point 0 of Active Window (VSAW0)
LCD_DataWrite(0x00); //Vertical Start Point of Active Window [7:0]
LCD_CmdWrite(0x33); //Vertical Start Point 1 of Active Window (VSAW1)
LCD_DataWrite(0x00); //Vertical Start Point of Active Window [8]
LCD_CmdWrite(0x36); //Vertical End Point of Active Window 0 (VEAW0)
LCD_DataWrite(0xdf); //Vertical End Point of Active Window [7:0]
LCD_CmdWrite(0x37); //Vertical End Point of Active Window 1 (VEAW1)
LCD_DataWrite(0x01); //Vertical End Point of Active Window [8]

#endif

#elifdef P800x480

//AT070TN92 setting
//=---------------- Display Window800x480 =----------------=
    /*
      LCD_CmdWrite(0x04); //PCLK inverse
      LCD_DataWrite(0x81);
      Delay1ms(1);
      //Horizontal set
      LCD_CmdWrite(0x14); //HDWR//Horizontal Display Width Setting Bit[6:0]
      LCD_DataWrite(0x63);//Horizontal display width(pixels) = (HDWR + 1)*8
      LCD_CmdWrite(0x15); //Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
      LCD_DataWrite(0x03);//Horizontal Non-Display Period Fine Tuning(HNDFT) [3:0]
      LCD_CmdWrite(0x16); //HNDR//Horizontal Non-Display Period Bit[4:0]
      LCD_DataWrite(0x03);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
      LCD_CmdWrite(0x17); //HSTR//HSYNC Start Position[4:0]
      LCD_DataWrite(0x02);//HSYNC Start Position(PCLK) = (HSTR + 1)*8
      LCD_CmdWrite(0x18); //HPWR//HSYNC Polarity ,The period width of HSYNC.
      LCD_DataWrite(0x00);//HSYNC Width [4:0] HSYNC Pulse width(PCLK) = (HPWR + 1)*8
      //Vertical set
      LCD_CmdWrite(0x19); //VDHR0 //Vertical Display Height Bit [7:0]
      LCD_DataWrite(0xdf);//Vertical pixels = VDHR + 1
      LCD_CmdWrite(0x1a); //VDHR1 //Vertical Display Height Bit [8]
LCD_DataWrite(0x01); //Vertical pixels = VDHR + 1
LCD_CmdWrite(0x1b); //VNDR0 //Vertical Non-Display Period Bit [7:0]
LCD_DataWrite(0x14); //Vertical Non-Display area = (VNDR + 1)
LCD_CmdWrite(0x1c); //VNDR1 //Vertical Non-Display Period Bit [8]
LCD_DataWrite(0x00); //Vertical Non-Display area = (VNDR + 1)
LCD_CmdWrite(0x1d); //VSTR0 //VSYNC Start Position[7:0]
LCD_DataWrite(0x06); //VSYNC Start Position(PCLK) = (VSTR + 1)
LCD_CmdWrite(0x1e); //VSTR1 //VSYNC Start Position[8]
LCD_DataWrite(0x00); //VSYNC Start Position(PCLK) = (VSTR + 1)
LCD_CmdWrite(0x1f); //VPWR //VSYNC Polarity ,VSYNC Pulse Width[6:0]
LCD_DataWrite(0x00); //VSYNC Pulse Width(PCLK) = (VPWR + 1)
//LCD_CmdWrite(0xf2);
//LCD_DataWrite(0x01);
*/
//HSD050IDW1 setting
//============== Display Window800x480 ===============
LCD_CmdWrite(0x04); //PCLK inverse
LCD_DataWrite(0x81);
Delay1ms(1);
//Horizontal set
LCD_CmdWrite(0x14); //HDWR//Horizontal Display Width Setting Bit[6:0]
LCD_DataWrite(0x63);//Horizontal display width(pixels) = (HDWR + 1)*8
LCD_CmdWrite(0x15);//Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
LCD_DataWrite(0x00);//Horizontal Non-Display Period Fine Tuning(HNDFT) [3:0]
LCD_CmdWrite(0x16); //HNDR//Horizontal Non-Display Period Bit[4:0]
LCD_DataWrite(0x03);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
LCD_CmdWrite(0x17); //HSTR//HSYNC Start Position[4:0]
LCD_DataWrite(0x03);//HSYNC Start Position(PCLK) = (HSTR + 1)*8
LCD_CmdWrite(0x18); //HPWR//HSYNC Polarity ,The period width of HSYNC.
LCD_DataWrite(0x0B);//HSYNC Width [4:0] HSYNC Pulse width(PCLK) = (HPWR + 1)*8
//Vertical set
LCD_CmdWrite(0x19); //VDHR0 //Vertical Display Height Bit [7:0]
LCD_DataWrite(0xdf);//Vertical pixels = VDHR + 1
LCD_CmdWrite(0x1a); //VDHR1 //Vertical Display Height Bit [8]
LCD_DataWrite(0x01); //Vertical pixels = VDHR + 1
LCD_CmdWrite(0x1b); //VNDR0 //Vertical Non-Display Period Bit [7:0]
LCD_DataWrite(0x20);//Vertical Non-Display area = (VNDR + 1)
LCD_CmdWrite(0x1c); //VNDR1 //Vertical Non-Display Period Bit [8]
LCD_DataWrite(0x00); //Vertical Non-Display area = (VNDR + 1)
LCD_CmdWrite(0x1d); //VSTR0 //VSYNC Start Position[7:0]
LCDDataWrite(0x16); //VSYNC Start Position(PCLK) = (VSTR + 1)
LCDCmdWrite(0x1e); //VSTR //VSYNC Start Position[8]
LCDDataWrite(0x00); //VSYNC Start Position(PCLK) = (VSTR + 1)
LCDCmdWrite(0x1f); //VPWR //VSYNC Polarity,VSYNC Pulse Width[6:0]
LCDDataWrite(0x01); //VSYNC Pulse Width(PCLK) = (VPWR + 1)

//Active window set
//setting active window X
  LCDCmdWrite(0x30); //Horizontal Start Point 0 of Active Window (HSAW0)
  LCDDataWrite(0x00); //Horizontal Start Point of Active Window [7:0]
  LCDCmdWrite(0x31); //Horizontal Start Point 1 of Active Window (HSAW1)
  LCDDataWrite(0x00); //Horizontal Start Point of Active Window [9:8]
  LCDCmdWrite(0x34); //Horizontal End Point 0 of Active Window (HEAW0)
  LCDDataWrite(0x1f); //Horizontal End Point of Active Window [7:0]
  LCDCmdWrite(0x35); //Horizontal End Point 1 of Active Window (HEAW1)
  LCDDataWrite(0x03); //Horizontal End Point of Active Window [9:8]

 //setting active window Y
  LCDCmdWrite(0x32); //Vertical Start Point 0 of Active Window (VSAW0)
  LCDDataWrite(0x00); //Vertical Start Point of Active Window [7:0]
  LCDCmdWrite(0x33); //Vertical Start Point 1 of Active Window (VSAW1)
  LCDDataWrite(0x00); //Vertical Start Point of Active Window [8]
  LCDCmdWrite(0x36); //Vertical End Point of Active Window 0 (VEAW0)
  LCDDataWrite(0xdf); //Vertical End Point of Active Window [7:0]
  LCDCmdWrite(0x37); //Vertical End Point of Active Window 1 (VEAW1)
  LCDDataWrite(0x01); //Vertical End Point of Active Window [8]
#endif
3. Display on Sequence

RA8875_RESET = 0;
Delay 1 ms;
RA8875_RESET = 1;
Delay 10 ms;

PLL initial setting

LCD initial setting

Write data to RA8875 display RAM

Display on setting:
LCD_CmdWrite(0x01);
LCD_DataWrite(0x80);

TFT LCD Backlight on

Figure 3-1
4. Sleep Mode Sequence

*Attention:
We suggest to confirm your TFT panel could be disable when the RA8875 enter sleep, to avoid liquid crystal polarization. RAiO shall not be held liable for any damages about TFT panel, when customers use sleep mode incorrect!

**Figure 4-1**

```
Enter Sleep:

TFT LCD Backlight off

Set RA8875 display off:
LCD_CmdWrite(0x01);
LCD_DataWrite(0x00);
Delay100ms(1);

Set sleep mode:
LCD_CmdWrite(0x01);
LCD_DataWrite(0x02);
```
Exit Sleep:

Set exit sleep:
LCD_CmdWrite(0x01);
LCD_DataWrite(0x00);

Delay 100ms

Display on setting:
LCD_CmdWrite(0x01);
LCD_DataWrite(0x80);
Delay100ms(1);

TFT LCD Backlight on

Figure 4-2
5. Display RAM pure data write example

Include RA8875_subrotine.c first, then refer the code below.

**If MCU 8 bit interface:**

```c
Graphic_Mode( );  //set to graphic mode
XY_Coordinate(0,0); //set write cursor position
LCD_CmdWrite(0x02); //set CMD [02h] before data write
for (i=384000;i>0;i--)
{ LCD_DataWrite(0xf8); // write color red data
  LCD_DataWrite(0x00);
}
```

**If MCU 16 bit interface:**

```c
Graphic_Mode( );  //set to graphic mode
XY_Coordinate(0,0); //set write cursor position
LCD_CmdWrite(0x02); // set CMD [02h] before data write
for (i=384000;i>0;i--)
{ LCD_DataWrite(0xf800); }// write color red data
```

You could download some example code and RA8875 subrotine on the website:

6. Appendix

a. How to program the Serial Flash Memory by the external programmer?
If the advance circuit is already designed by the LCM factory as the following picture1 and picture2, just leading the related pins of the serial flash out to the connector and then the user will be able to program the picture data (*.bin) by the external memory programmer. Please refer to the following picture3.

The resistors (R136~R139) should be took out when the serial flash memory is programmed by the external memory programmer.

Picture 1

Picture 2

Picture 3