RAiO

RA8875

Character/Graphic

TFT LCD Controller

Application Note

Version 1.3

July 11, 2011

RAiO Technology Inc.
©Copyright RaiO Technology Inc. 2011

Update History		
Version	**Date**	**Description**
1.0	May, 03, 2011	First Release
1.1	May,09,2011	Schematic 2 updata ,initial code updata
1.2	June,22,2011	Schematic 2 updata
1.3	July,11,2011	Add chapter 6.appendix

Chapter Contents Page

1. Application Circuit

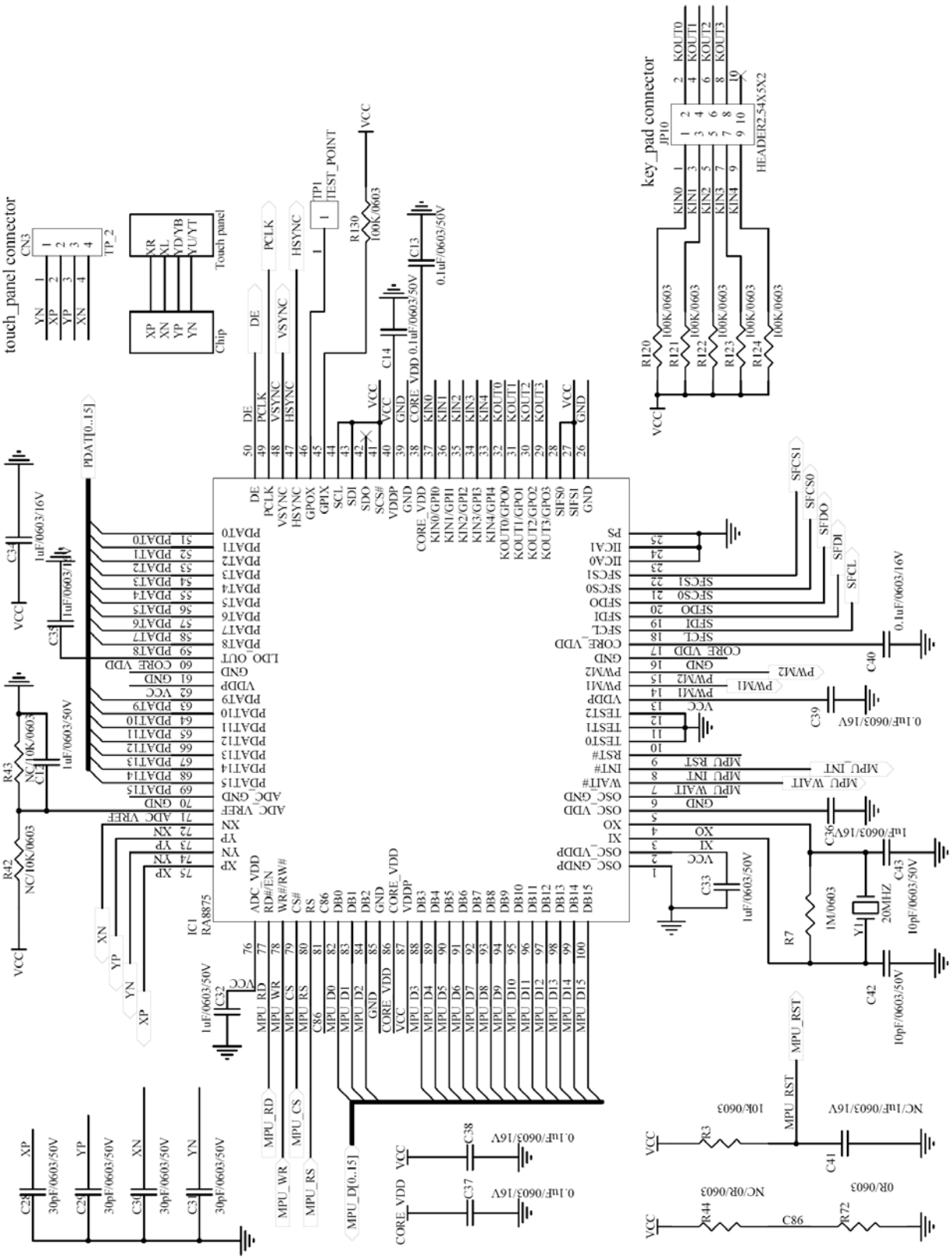


Figure 1-1

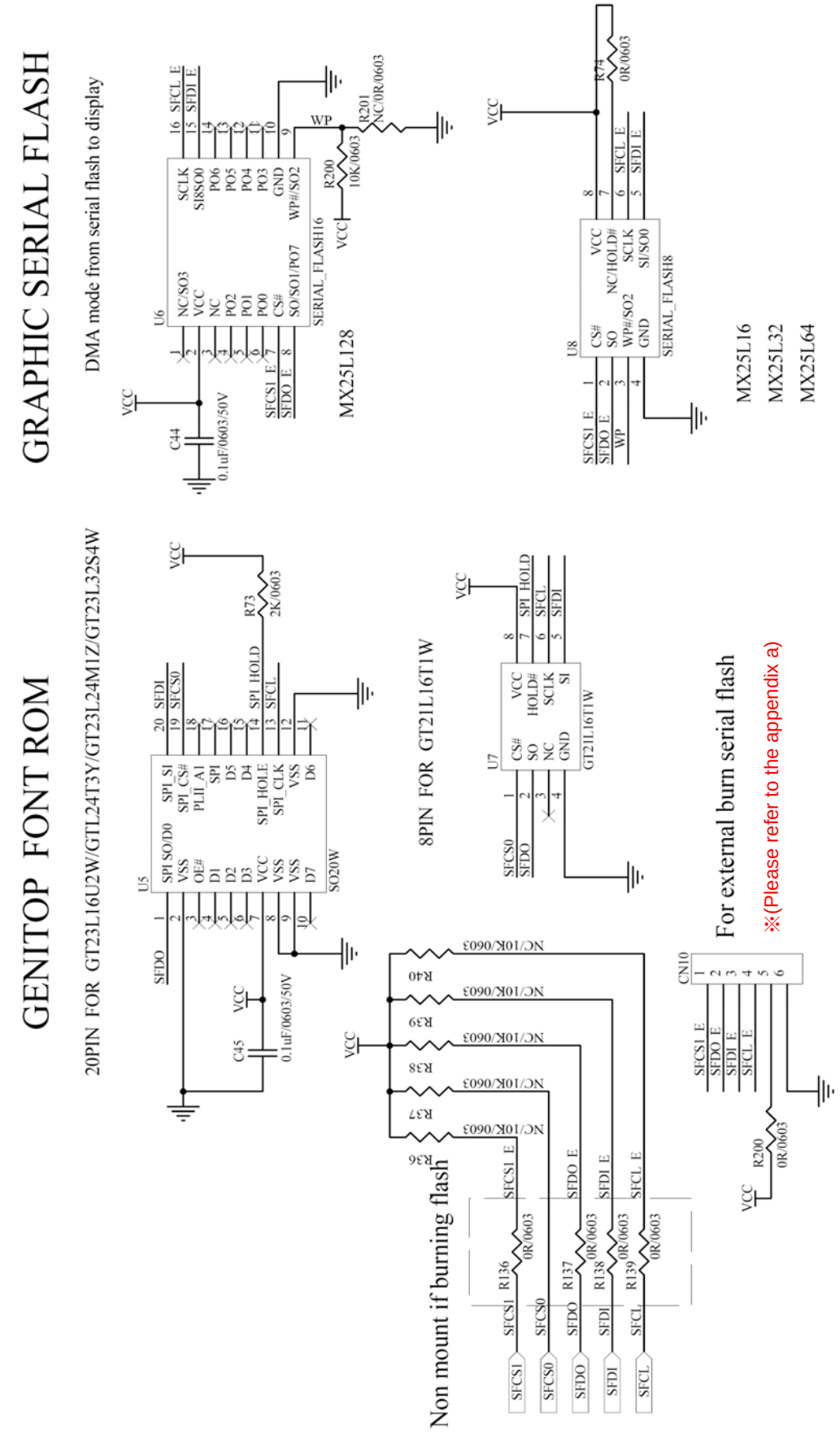

GRAPHIC SERIAL FLASH
DMA mode from serial flash to display
MX25L128
MX25L16
MX25L32
MX25L64
GENITOP FONT ROM
20PIN FOR GT23L16U2W/GTL24T3Y/GT23L24M1Z/GT23L32S4W
8PIN FOR GT21L16T1W
For external burn serial flash
※(Please refer to the appendix a)
Non mount if burning flash

Figure 1-2

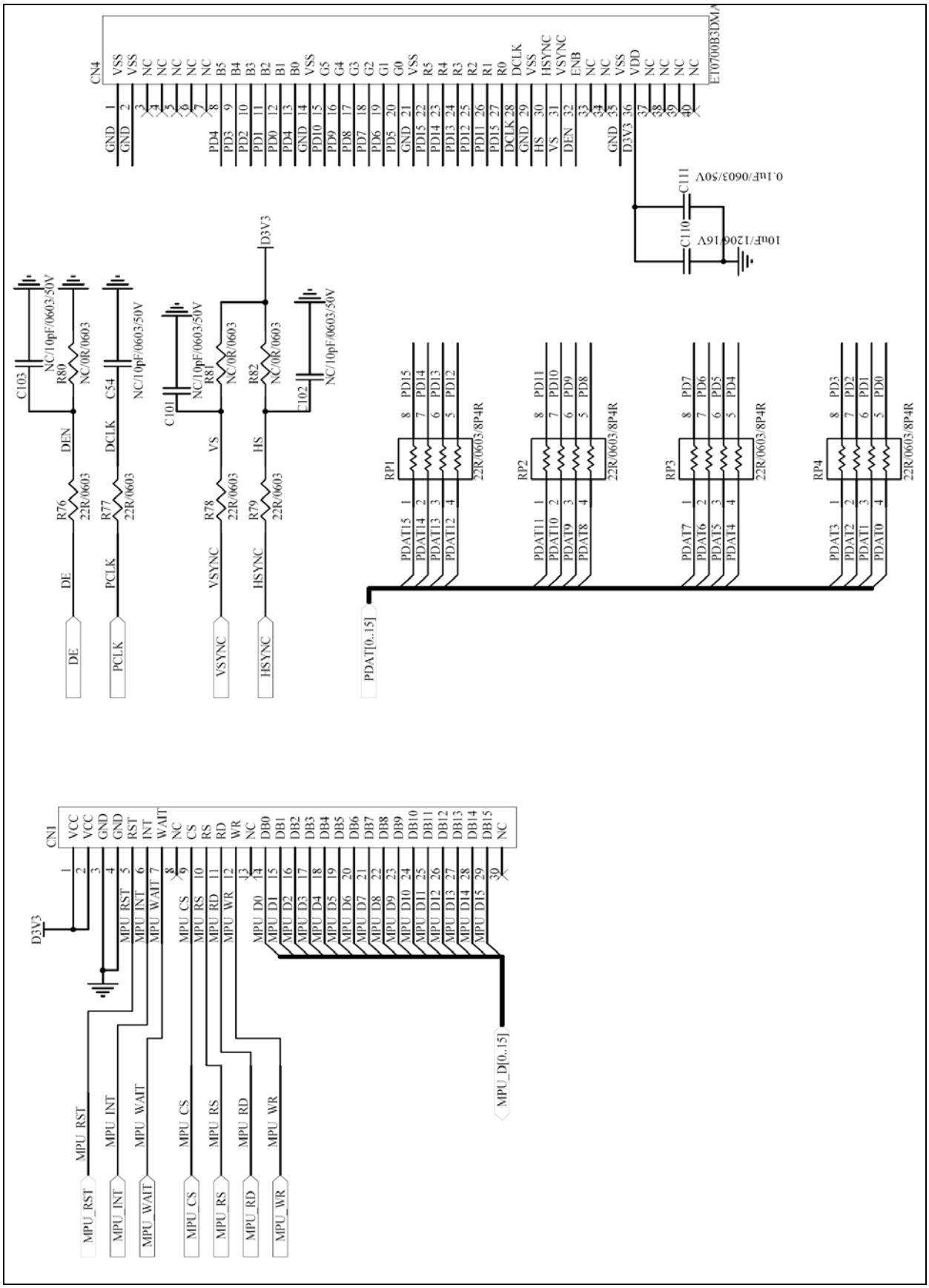

CN4
E1070083DMA
VSS VSS NC NC NC NC NC
B5 B4 B3 B2 B1 B0 VSS G5 G4 G3 G2 G1 G0 VSS R5 R4 R3 R2 R1 R0 DCLK VSS HSYNC VSYNC ENB NC NC VSS VDD NC NC NC NC
GND GND PD4 PD3 PD2 PD1 PD0 PD4 GND PD10 PD9 PD8 PD7 PD6 PD5 GND PD15 PD14 PD13 PD12 PD11 PD15 DCLK GND HS VS DEN NC NC GND D3V3
C111 0.1uF/0603/50V
C110 10uF/1206/16V
D3V3
C103 NC/10pF/0603/50V
R80 NC/0R/0603
C54 NC/10pF/0603/50V
C101 NC/10pF/0603/50V
R81 NC/0R/0603
R82 NC/0R/0603
C102 NC/10pF/0603/50V
R76 22R/0603
R77 22R/0603
R78 22R/0603
R79 22R/0603
DEN
DCLK
VS
HS
DE
PCLK
VSYNC
HSYNC
RP1 RP2 RP3 RP4
8 PD15 7 PD14 6 PD13 5 PD12
8 PD11 7 PD10 6 PD9 5 PD8
8 PD7 7 PD6 6 PD5 5 PD4
8 PD3 7 PD2 6 PD1 5 PD0
22R/0603/8P4R
PDAT15 PDAT14 PDAT13 PDAT12
PDAT11 PDAT10 PDAT9 PDAT8
PDAT7 PDAT6 PDAT5 PDAT4
PDAT3 PDAT2 PDAT1 PDAT0
PDAT[0..15]
CN1
VCC VCC GND GND RST INT WAIT NC CS RS RD WR NC DB0 DB1 DB2 DB3 DB4 DB5 DB6 DB7 DB8 DB9 DB10 DB11 DB12 DB13 DB14 DB15 NC
MPU_RST MPU_INT MPU_WAIT MPU_CS MPU_RS MPU_RD MPU_WR
MPU_D[0..15]

Figure 1-3

2. Initial Code

```
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
      //LCD_DataWrite(0x0F); // if 16bit MCU interface      and 65k color display
#ifdef P320x240
//==============   Display Window320x240 ==================
      LCD_CmdWrite(0x04);        //set PCLK invers
      LCD_DataWrite(0x03);
      Delay1ms(1);

       //Horizontal set
      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(0x04);//HSYNC Start Position(PCLK) = (HSTR + 1)*8
      LCD_CmdWrite(0x18); //HPWR//HSYNC Polarity ,The period width of HSYNC.
      LCD_DataWrite(0x03);//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(0xef);//Vertical pixels = VDHR + 1
      LCD_CmdWrite(0x1A); //VDHR1 //Vertical Display Height Bit [8]
      LCD_DataWrite(0x00);//Vertical pixels = VDHR + 1
      LCD_CmdWrite(0x1B); //VNDR0 //Vertical Non-Display Period Bit [7:0]
      LCD_DataWrite(0x05);//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(0x0e);//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(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); //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(0x01);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
    LCD_CmdWrite(0x17); //HSTR//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.
```

```
    LCD_DataWrite(0x05);//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(0x0f); //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(0x02);    //VSYNC Start Position(PCLK) = (VSTR + 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(0x07);//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(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)
    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(0x0F); //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
```

```
#ifdef P640x480
//==============  Display Window640x480 ==================
      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); //Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
      LCD_DataWrite(0x05);//Horizontal Non-Display Period Fine Tuning(HNDFT) [3:0]
      LCD_CmdWrite(0x16);//HNDR//Horizontal Non-Display Period Bit[4:0]
      LCD_DataWrite(0x0f);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
      LCD_CmdWrite(0x17);//HSTR//HSYNC Start Position[4:0]
      LCD_DataWrite(0x01);//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(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)
      LCD_CmdWrite(0x1D);//VSTR0 //VSYNC Start Position[7:0]
      LCD_DataWrite(0x0E);//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(0x01);//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(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


#ifdef 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(0x01);//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]
```

```
 LCD_DataWrite(0x16);//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(0x01);//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(0x1F); //Horizontal End Point of Active Window [7:0]
    LCD_CmdWrite(0x35); //Horizontal End Point 1 of Active Window (HEAW1)
    LCD_DataWrite(0x03); //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
}
```

3. Display on Sequence

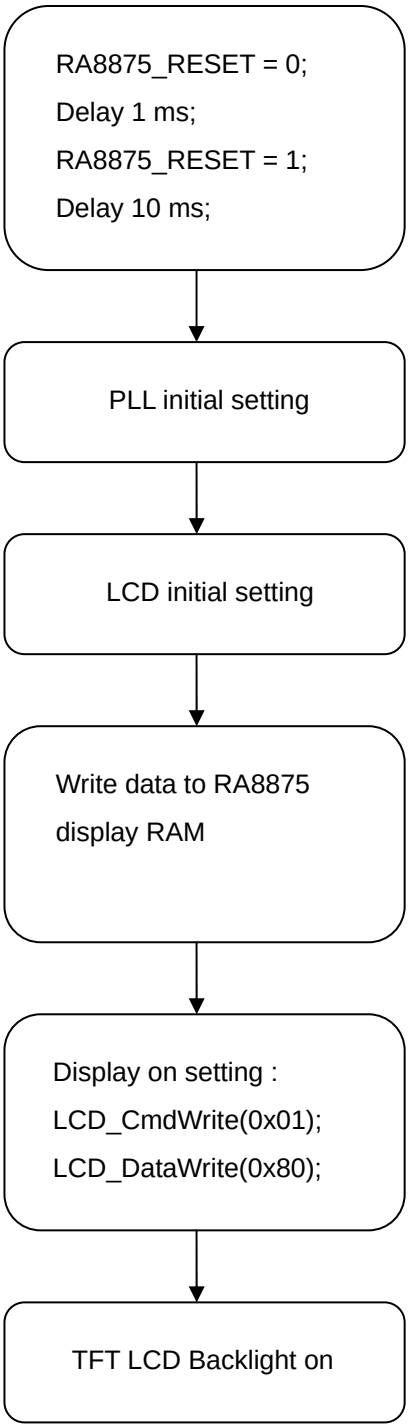


Figure 3-1

4. Sleep Mode Sequence

Enter Sleep :

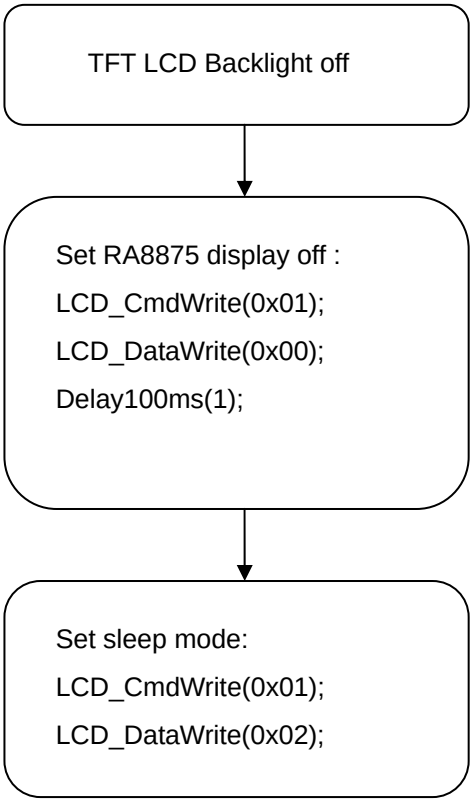


Figure 4-1

*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 !

Exit Sleep :

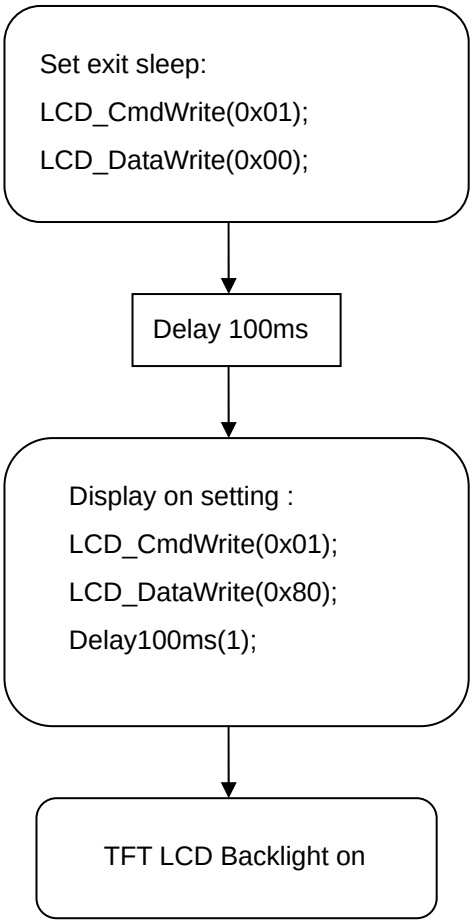


Figure 4-2

5. Display RAM pure data write example

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

If MCU 8 bit interface :

```
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 :

```
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
```

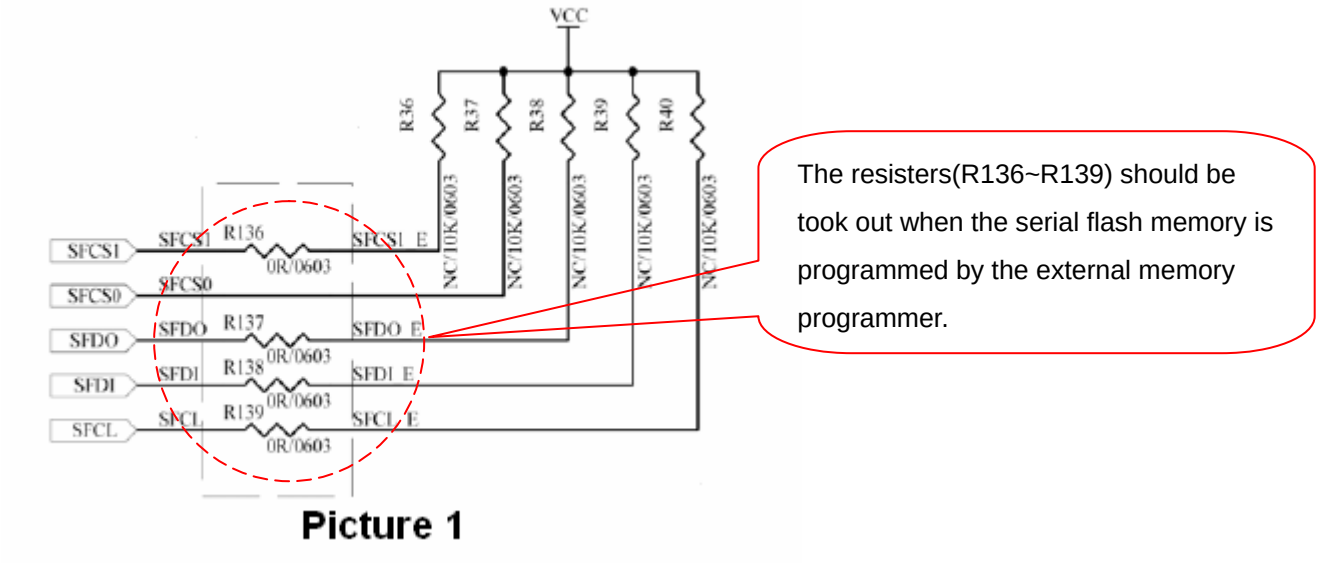
You could download some example code and RA8875 subrotine on the website :

www.raio.com.tw

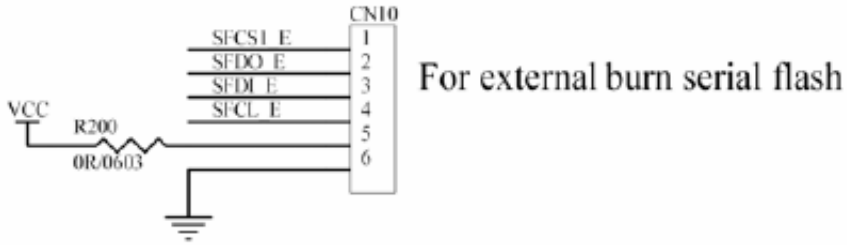
6. Appendix

a. How to program the Serial Flash Memory by the external programmer?

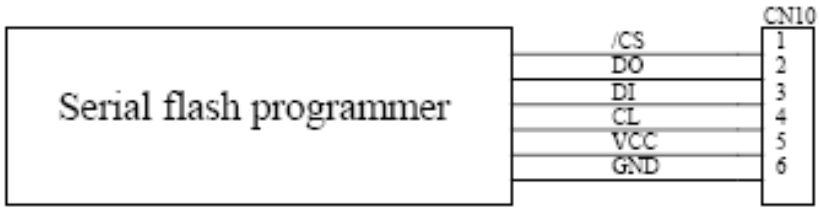
If the advance circuit is already designed by the LCM factory as the following picture1 and picture2, just leading the related pins of the serial flash out to the connector and then the user will be able to program the picture data(*.bin) by the external memory programmer. Please refer to the following pinture3.



Picture 1



Picture2



Picture3