```
/***************************************************************************
*Copyright (c) 2007 wormchen
*All rights reserved
*文 件 名: main.c
*说    明: Nokia5110液晶模块驱动
*
*主要硬件: Mega16(外部7.37628M)
*编译环境: AVRStudio + WinAVR 20070525
*当前版本: 1.0
*作    者: 陈崇
*完成日期: 2008年11月17日14:39:26
*取代版本:
*原作  者:
*完成日期:
***************************************************************************/
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <avr/pgmspace.h>
#define N5110_DC_SET PORTB |= (1 << PB3)
#define N5110_DC_CLR PORTB &= ~(1 << PB3)
#define N5110_CE_SET PORTB |= (1 << PB4)
#define N5110_CE_CLR PORTB &= ~(1 << PB4)

#define N5110_RST_SET PORTB |= (1 << PB6)
#define N5110_RST_CLR PORTB &= ~(1 << PB6)
#define HANZIKU_MAX 4

typedef  struct HANZI_STRUCT
{
    uint8_t data[24];
    uint8_t hanzi[2];

}hanzi_struct;

const uint8_t cBmpCode[504] PROGMEM =
{
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x00,0x00,
0x00,0x80,0xC0,0x40,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xE0,0xFC,0xFF,0xFF,0xFF,0xFF,
0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xC0,
0xE0,0xF8,0xF8,0xFC,0xFE,0xFE,0xFF,0xFF,0xFF,0x3F,0x1F,0x0F,0x0F,0x0F,0x0F,0x0F,
0xC7,0xE7,0x07,0x83,0xC3,0x03,0x03,0x00,0x03,0x01,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x40,0xC8,0x9C,0x94,0x16,0x12,0x27,0x67,0xC7,0x87,
0x8F,0x1F,0x1F,0x3F,0x3F,0x7F,0x7F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x3F,0x07,0x01,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x0F,0x1F,0x18,0x1F,0x0F,0x04,0x00,0x00,0x00,0x00,0x00,0x00,
0xF8,0x40,0x00,0x00,0xFC,0xFE,0x08,0x78,0xF0,0x00,0x00,0x01,0x7F,0xFF,0xFF,0xFF,
0xFF,0xFE,0xFF,0xFF,0x3B,0x0F,0x0E,0x08,0x00,0x06,0x07,0x07,0x03,0x03,0x83,0x07,
0x0F,0x1C,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x1F,0x7F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xC0,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x20,0x7F,0x07,0x06,0x7E,0x0C,0x00,0x00,
0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x07,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x07,0x07,0x03,0x01,0x01,0x01,0x00,0x00,0x80,0xC0,0xE0,0x60,0x00,0xC0,0xE0,
0xF8,0xFC,0xFF,0xCE,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x03,0x0F,0x0F,0x1F,
0x1F,0x3F,0x3F,0x3E,0x7E,0x7C,0x7C,0x7C,0x7C,0x7C,0x3C,0x3C,0x38,0x38,0x38,0x38,
0x30,0x10,0x40,0x40,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x02,0x04,0x08,0x02,0x02,0x0A,0x4A,0x0A,0x0B,0x09,0x09,0x08,0x0C,
0x0C,0x0C,0x0C,0x0C,0x0F,0x0F,0x3F,0x3F,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x0C,0x3C,0x3C,0xFC,0xFC,0xF8,0xFE,0xF6,0xF6,0xE0,0xE0,0xE4,0xE5,0xCF,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
};
const uint8_t Ascii[][5] PROGMEM =
{
    { 0x00, 0x00, 0x00, 0x00, 0x00 }, // sp
    { 0x00, 0x00, 0x2f, 0x00, 0x00 }, // !
    { 0x00, 0x07, 0x00, 0x07, 0x00 }, // "
    { 0x14, 0x7f, 0x14, 0x7f, 0x14 }, // #
    { 0x24, 0x2a, 0x7f, 0x2a, 0x12 }, // $
    { 0xc4, 0xc8, 0x10, 0x26, 0x46 }, // %
    { 0x36, 0x49, 0x55, 0x22, 0x50 }, // &
    { 0x00, 0x05, 0x03, 0x00, 0x00 }, // '
    { 0x00, 0x1c, 0x22, 0x41, 0x00 }, // (
    { 0x00, 0x41, 0x22, 0x1c, 0x00 }, // )
    { 0x14, 0x08, 0x3E, 0x08, 0x14 }, // *
    { 0x08, 0x08, 0x3E, 0x08, 0x08 }, // +
    { 0x00, 0x00, 0x50, 0x30, 0x00 }, // ,
    { 0x10, 0x10, 0x10, 0x10, 0x10 }, // -
    { 0x00, 0x60, 0x60, 0x00, 0x00 }, // .
    { 0x20, 0x10, 0x08, 0x04, 0x02 }, // /
    { 0x3E, 0x51, 0x49, 0x45, 0x3E }, // 0
    { 0x00, 0x42, 0x7F, 0x40, 0x00 }, // 1
    { 0x42, 0x61, 0x51, 0x49, 0x46 }, // 2
    { 0x21, 0x41, 0x45, 0x4B, 0x31 }, // 3
    { 0x18, 0x14, 0x12, 0x7F, 0x10 }, // 4
    { 0x27, 0x45, 0x45, 0x45, 0x39 }, // 5
    { 0x3C, 0x4A, 0x49, 0x49, 0x30 }, // 6
    { 0x01, 0x71, 0x09, 0x05, 0x03 }, // 7
    { 0x36, 0x49, 0x49, 0x49, 0x36 }, // 8
    { 0x06, 0x49, 0x49, 0x29, 0x1E }, // 9
    { 0x00, 0x36, 0x36, 0x00, 0x00 }, // :
    { 0x00, 0x56, 0x36, 0x00, 0x00 }, // ;
    { 0x08, 0x14, 0x22, 0x41, 0x00 }, // <
    { 0x14, 0x14, 0x14, 0x14, 0x14 }, // =
    { 0x00, 0x41, 0x22, 0x14, 0x08 }, // >
    { 0x02, 0x01, 0x51, 0x09, 0x06 }, // ?
    { 0x32, 0x49, 0x59, 0x51, 0x3E }, // @
    { 0x7E, 0x11, 0x11, 0x11, 0x7E }, // A
    { 0x7F, 0x49, 0x49, 0x49, 0x36 }, // B
    { 0x3E, 0x41, 0x41, 0x41, 0x22 }, // C
    { 0x7F, 0x41, 0x41, 0x22, 0x1C }, // D
    { 0x7F, 0x49, 0x49, 0x49, 0x41 }, // E
    { 0x7F, 0x09, 0x09, 0x09, 0x01 }, // F
    { 0x3E, 0x41, 0x49, 0x49, 0x7A }, // G
    { 0x7F, 0x08, 0x08, 0x08, 0x7F }, // H
    { 0x00, 0x41, 0x7F, 0x41, 0x00 }, // I
    { 0x20, 0x40, 0x41, 0x3F, 0x01 }, // J
    { 0x7F, 0x08, 0x14, 0x22, 0x41 }, // K
    { 0x7F, 0x40, 0x40, 0x40, 0x40 }, // L
    { 0x7F, 0x02, 0x0C, 0x02, 0x7F }, // M
    { 0x7F, 0x04, 0x08, 0x10, 0x7F }, // N
    { 0x3E, 0x41, 0x41, 0x41, 0x3E }, // O
    { 0x7F, 0x09, 0x09, 0x09, 0x06 }, // P
    { 0x3E, 0x41, 0x51, 0x21, 0x5E }, // Q
    { 0x7F, 0x09, 0x19, 0x29, 0x46 }, // R
    { 0x46, 0x49, 0x49, 0x49, 0x31 }, // S
    { 0x01, 0x01, 0x7F, 0x01, 0x01 }, // T
    { 0x3F, 0x40, 0x40, 0x40, 0x3F }, // U
    { 0x1F, 0x20, 0x40, 0x20, 0x1F }, // V
    { 0x3F, 0x40, 0x38, 0x40, 0x3F }, // W
    { 0x63, 0x14, 0x08, 0x14, 0x63 }, // X
    { 0x07, 0x08, 0x70, 0x08, 0x07 }, // Y
    { 0x61, 0x51, 0x49, 0x45, 0x43 }, // Z
    { 0x00, 0x7F, 0x41, 0x41, 0x00 }, // [
    { 0x55, 0x2A, 0x55, 0x2A, 0x55 }, // 55
    { 0x00, 0x41, 0x41, 0x7F, 0x00 }, // ]
    { 0x04, 0x02, 0x01, 0x02, 0x04 }, // ^
    { 0x40, 0x40, 0x40, 0x40, 0x40 }, // _
    { 0x00, 0x01, 0x02, 0x04, 0x00 }, // '
    { 0x20, 0x54, 0x54, 0x54, 0x78 }, // a
    { 0x7F, 0x48, 0x44, 0x44, 0x38 }, // b
    { 0x38, 0x44, 0x44, 0x44, 0x20 }, // c
    { 0x38, 0x44, 0x44, 0x48, 0x7F }, // d
    { 0x38, 0x54, 0x54, 0x54, 0x18 }, // e
    { 0x08, 0x7E, 0x09, 0x01, 0x02 }, // f
    { 0x0C, 0x52, 0x52, 0x52, 0x3E }, // g
    { 0x7F, 0x08, 0x04, 0x04, 0x78 }, // h
    { 0x00, 0x44, 0x7D, 0x40, 0x00 }, // i
    { 0x20, 0x40, 0x44, 0x3D, 0x00 }, // j
    { 0x7F, 0x10, 0x28, 0x44, 0x00 }, // k
    { 0x00, 0x41, 0x7F, 0x40, 0x00 }, // l
    { 0x7C, 0x04, 0x18, 0x04, 0x78 }, // m
    { 0x7C, 0x08, 0x04, 0x04, 0x78 }, // n
    { 0x38, 0x44, 0x44, 0x44, 0x38 }, // o
    { 0x7C, 0x14, 0x14, 0x14, 0x08 }, // p
    { 0x08, 0x14, 0x14, 0x18, 0x7C }, // q
    { 0x7C, 0x08, 0x04, 0x04, 0x08 }, // r
    { 0x48, 0x54, 0x54, 0x54, 0x20 }, // s
    { 0x04, 0x3F, 0x44, 0x40, 0x20 }, // t
    { 0x3C, 0x40, 0x40, 0x20, 0x7C }, // u
    { 0x1C, 0x20, 0x40, 0x20, 0x1C }, // v
    { 0x3C, 0x40, 0x30, 0x40, 0x3C }, // w
    { 0x44, 0x28, 0x10, 0x28, 0x44 }, // x
    { 0x0C, 0x50, 0x50, 0x50, 0x3C }, // y
    { 0x44, 0x64, 0x54, 0x4C, 0x44 }  // z
};
const hanzi_struct Hanziku[HANZIKU_MAX] PROGMEM =
{
    0x19,0xE2,0x14,0x42,0xF2,0x2E,0x72,0x8F,
    0xAA,0x7A,0x02,0x00,0x01,0x07,0x00,0x00,
    0x07,0x04,0x04,0x02,0x01,0x02,0x04,0x00, "液",
    /* (12 X 12 , 宋体 )*/
    0x00,0xC0,0x40,0x5F,0xD5,0x15,0xD5,0x55,
    0x5F,0x40,0xC0,0x00,0x00,0x07,0x05,0x05,
    0x07,0x00,0x07,0x05,0x05,0x05,0x07,0x00, "晶",
    /* (12 X 12 , 宋体 )*/
    0x21,0x3D,0x21,0x3F,0xE0,0x00,0xFF,0x89,
    0x51,0x71,0x8D,0x00,0x01,0x05,0x05,0x06,
    0x01,0x00,0x07,0x04,0x04,0x04,0x05,0x00, "驱",
    /* (12 X 12 , 宋体 )*/
    0x92,0x52,0x32,0x52,0x92,0x10,0x08,0xFF,
    0x08,0x08,0xF8,0x00,0x03,0x01,0x01,0x01,
    0x05,0x02,0x01,0x04,0x04,0x06,0x01,0x00, "动",

};

uint8_t Ziku_FindHanzi(uint8_t *pcBuff)
{
    hanzi_struct *pcHanziku;
    uint8_t i;
    pcHanziku = Hanziku;
    for(i = 0; i < HANZIKU_MAX ; i++)
    {
        if(pgm_read_byte(pcHanziku->hanzi) ==  pcBuff[0] &&
            pgm_read_byte(pcHanziku->hanzi + 1) ==  pcBuff[1])
        {
            return i;
        }
        else
        {
            pcHanziku++;
        }
    }
    return 0;
}
/**************************************************************************
*名称: DelayMS
*功能: 延时nms
*参数: 无
*返回: 无
**************************************************************************/
void Delay_MS(uint16_t dMS)
{
    while(dMS--)
    {
        _delay_loop_2(2000);    // 延时1ms(粗略)
    }
}

/**************************************************************************
*名称: void PORT_Init(void)
*功能: 端口初始化
*参数: 无
*返回: 无
**************************************************************************/
void PORT_Init(void)
{

    //PORTB |= (1 << PB3) | (1 << PB4) |(1 << PB5) | (1 << PB6) | (1 << PB7);
    DDRB |= (1 << PB3) | (1 << PB4) |(1 << PB5) | (1 << PB6) | (1 << PB7);

}

/**************************************************************************
*名称: SPI_MasterInit
*功能: SPI主机模式初始化
*参数: 无
*返回: 无
**************************************************************************/
void SPI_MasterInit(void)
{
    SPCR |= (1 << SPE) | (1 << MSTR);
                    //使能SPI 主机模式，设置时钟速率为fck/2
    SPSR |= _BV(SPI2X); //倍速
}
/**************************************************************************
*名称: void SPI_MasterTransmit(uint8_t cData)
*功能: SPI主机模式传送数据
*参数: cData 要传输的字节数据
*返回: SPI接收值
**************************************************************************/
uint8_t SPI_MasterTransmit(unsigned char cByte)
{
    SPDR = cByte; //* 启动数据传输
    while(!(SPSR & (1<<SPIF)));//等待传输结束
    return SPDR; //返回SPI接收的数据
}
/************************************************************************
*名称: N5110_WriteByte
*功能: 往5110写一字节数据
*参数: cByte 写入数据，cCommand 0指令，1数据
*返回: 无
************************************************************************/
void N5110_WriteByte(uint8_t cByte,uint8_t cCommand)
{
    N5110_CE_CLR;
    if(cCommand == 0)
    {
        N5110_DC_CLR;
    }
    else
    {
        N5110_DC_SET;
    }
    SPI_MasterTransmit(cByte);
    N5110_CE_SET;
}
/************************************************************************
*名称: N5110_Clear
*功能: N5110清屏
*参数: 无
*返回: 无
************************************************************************/
void N5110_Clear(void)
{
    uint16_t i;

    N5110_WriteByte(0x80, 0);
    N5110_WriteByte(0x40, 0);
    N5110_WriteByte(0x80, 0);
    for (i=0; i<504; i++)
    {
        N5110_WriteByte(0x00, 1);
    }
}
/************************************************************************
*名称: N5110_Init
*功能: N5110初始化
*参数: 无
*返回: 无
************************************************************************/
void N5110_Init(void)
{
    N5110_RST_CLR;
    asm("nop"::);
    asm("nop"::);
    //Delay_MS(1);
    N5110_RST_SET;
    N5110_CE_CLR;
    asm("nop"::);
    asm("nop"::);
    //Delay_MS(1);
    N5110_CE_SET;
    asm("nop"::);
    asm("nop"::);
    //Delay_MS(1);
    N5110_WriteByte(0x21, 0);   // 使用扩展命令设置LCD模式
```

```c
    N5110_WriteByte(0xc8, 0);   // 设置偏置电压
    N5110_WriteByte(0x06, 0);   // 温度校正
    N5110_WriteByte(0x13, 0);   // 1:48
    N5110_WriteByte(0x20, 0);   // 使用基本命令
    N5110_Clear();
    N5110_WriteByte(0x0C, 0);   // 设定显示模式，正常显示
    N5110_CE_CLR;
}
/*************************************************************************
*名称: N5110_PutChar
*功能: N5110输出一字符
*参数: x 列0-83, 行0-5, cData输出字符
*返回: 无
*************************************************************************/
void N5110_PutChar(uint8_t x, uint8_t y, uint8_t cData)
{
    uint8_t *pcData;
    uint8_t i;
    x |= 0x80;
    y |= 0x40;
    pcData = &Ascii;
    pcData += (cData - 32) * 5; //获取显示数据的位置
    N5110_WriteByte(x, 0);
    N5110_WriteByte(y, 0);
    for(i = 0; i < 5; i++)
    {
        N5110_WriteByte(pgm_read_byte(pcData++), 1);
    }
}
/*************************************************************************
*名称: N5110_PutChar
*功能: N5110输出字符串
*参数: x 列0-83, 行0-5, pcBuff输出字符
*返回: 无
*************************************************************************/
void N5110_PutString(uint8_t x, uint8_t y, uint8_t *pcBuff)
{
    while(*pcBuff)
    {
        N5110_PutChar(x, y, *pcBuff++);
        x +=5;
    }
}
/************************************************************************
*函数: N5110_PutHanzi
*功能: 显示一个12*12汉字
*参数: x列地址(0-7), y行地址(0-3), pcBuff要显示的汉字
*返回: 无
************************************************************************/
void N5110_PutHanzi(uint8_t x, uint8_t y, uint8_t *pcBuff)
{
    uint8_t i;
    uint8_t j;
    hanzi_struct *pcHanzi;
    uint8_t *pcData;
    pcHanzi = &Hanziku;
    pcHanzi += Ziku_FindHanzi(pcBuff);
    pcData = pcHanzi->data;
    x = 0x80 + x * 12;
    switch(y)
    {
        case 0:
        {
            y = 0x40;
            break;
        }
        case 1:
        {
            y = 0x42;
            break;
        }
        case 2:
        {
            y = 0x44;
            break;
        }
        default: y = 0x40;
    }
    N5110_WriteByte(x, 0);
    N5110_WriteByte(y, 0);
    for(i = 0; i < 2; i++)
    {

        for(j = 0; j < 12; j++)
        {
            N5110_WriteByte(pgm_read_byte(pcData++), 1);

        }
        y++; //写汉字下半部分
        N5110_WriteByte(x, 0);
        N5110_WriteByte(y, 0);

    }

}
/*************************************************************************
*函数: N5110_PutHanziString
*功能: 显示12*12汉字字符串
*参数: x 列地址(0-7), y行地址(0-3), pcBuff要显示的汉字串
*返回: 无
*************************************************************************/
void N5110_PutHanziString(uint8_t x, uint8_t y, uint8_t *pcBuff)
{
    while(*pcBuff)
    {
        N5110_PutHanzi(x, y, pcBuff);
        pcBuff += 2;
        x += 1;
    }
}
/*************************************************************************
*名称: N5110_DrawBmp
*功能: N5110显示位图
*参数: pcBuff位图数据
*返回: 无
*************************************************************************/
void N5110_DrawBmp(uint8_t *pcBuff)
{
    uint16_t i;
    N5110_WriteByte(0x80, 0);
    N5110_WriteByte(0x40, 0);
    N5110_WriteByte(0x80, 0);
    for (i = 0; i < 504; i++)
    {
        N5110_WriteByte(pgm_read_byte(pcBuff++), 1);
    }
```

```
}
int main(void)
{
    PORT_Init();
    SPI_MasterInit();
    N5110_Init();

    while(1)
    {
        N5110_Clear();
        N5110_DrawBmp(cBmpCode);
        Delay_MS(2000);
        N5110_Clear();
        N5110_PutString(15, 0,"Nokia 5110");
        N5110_PutHanziString(1, 1, "液晶驱动");
        N5110_PutString(20, 4,"worm chen");
        N5110_PutString(25, 5,"2008.11");
        Delay_MS(2000);

    }
}
```