#include <Arduino.h>

#include <Wire.h>

#include "MicroLCD.h"

#define I2C\_ADDR 0x78 >> 1

void LCD\_SH1106::WriteCommand(unsigned char ins)

{

Wire.beginTransmission(I2C\_ADDR);//0x78 >> 1

Wire.write(0x00);//0x00

Wire.write(ins);

Wire.endTransmission();

}

void LCD\_SH1106::WriteData(unsigned char dat)

{

Wire.beginTransmission(I2C\_ADDR);//0x78 >> 1

Wire.write(0x40);//0x40

Wire.write(dat);

Wire.endTransmission();

}

void LCD\_SH1106::setCursor(unsigned char x, unsigned char y)

{

m\_col = x + 2;

m\_row = y;

WriteCommand(0xb0 + m\_row);

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

}

void LCD\_SH1106::clear(byte x, byte y, byte width, byte height)

{

WriteCommand(SSD1306\_SETLOWCOLUMN | 0x0); // low col = 0

WriteCommand(SSD1306\_SETHIGHCOLUMN | 0x0); // hi col = 0

WriteCommand(SSD1306\_SETSTARTLINE | 0x0); // line #0

height >>= 3;

width >>= 3;

y >>= 3;

#ifdef TWBR

uint8\_t twbrbackup = TWBR;

TWBR = 18; // upgrade to 400KHz!

#endif

for (byte i = 0; i < height; i++) {

// send a bunch of data in one xmission

WriteCommand(0xB0 + i + y);//set page address

WriteCommand((x + 2) & 0xf);//set lower column address

WriteCommand(0x10 | (x >> 4));//set higher column address

for(byte j = 0; j < 8; j++){

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte k = 0; k < width; k++) {

Wire.write(0);

}

Wire.endTransmission();

}

}

#ifdef TWBR

TWBR = twbrbackup;

#endif

setCursor(0, 0);

}

size\_t LCD\_SH1106::write(uint8\_t c)

{

if (c == '\n') {

setCursor(0, m\_row + ((m\_font == FONT\_SIZE\_SMALL) ? 1 : 2));

return 1;

} else if (c == '\r') {

m\_col = 0;

return 1;

}

#ifdef TWBR

uint8\_t twbrbackup = TWBR;

TWBR = 18; // upgrade to 400KHz!

#endif

#ifndef MEMORY\_SAVING

if (m\_font == FONT\_SIZE\_SMALL) {

#endif

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

if (c > 0x20 && c < 0x7f) {

c -= 0x21;

for (byte i = 0; i < 5; i++) {

byte d = pgm\_read\_byte(&font5x8[c][i]);

Wire.write(d);

if (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) Wire.write(d);

}

Wire.write(0);

} else {

for (byte i = (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 11 : 6; i > 0; i--) {

Wire.write(0);

}

}

Wire.endTransmission();

m\_col += (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 11 : 6;

if (m\_col >= 128) {

m\_col = 0;

m\_row ++;

}

#ifndef MEMORY\_SAVING

} else {

if (c > 0x20 && c < 0x7f) {

c -= 0x21;

WriteCommand(0xB0 + m\_row);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = 0; i <= 14; i += 2) {

byte d = pgm\_read\_byte(&font8x16\_terminal[c][i]);

Wire.write(d);

if (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) Wire.write(d);

}

Wire.endTransmission();

WriteCommand(0xB0 + m\_row + 1);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = 1; i <= 15; i += 2) {

byte d = pgm\_read\_byte(&font8x16\_terminal[c][i]);

Wire.write(d);

if (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) Wire.write(d);

}

Wire.endTransmission();

} else {

WriteCommand(0xB0 + m\_row);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 16 : 8; i > 0; i--) {

Wire.write(0);

}

Wire.endTransmission();

WriteCommand(0xB0 + m\_row + 1);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 16 : 8; i > 0; i--) {

Wire.write(0);

}

Wire.endTransmission();

}

m\_col += (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 17 : 9;

if (m\_col >= 128) {

m\_col = 0;

m\_row += 2;

}

}

#endif

#ifdef TWBR

TWBR = twbrbackup;

#endif

return 1;

}

void LCD\_SH1106::writeDigit(byte n)

{

#ifdef TWBR

uint8\_t twbrbackup = TWBR;

TWBR = 18; // upgrade to 400KHz!

#endif

if (m\_font == FONT\_SIZE\_SMALL) {

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

if (n <= 9) {

n += '0' - 0x21;

for (byte i = 0; i < 5; i++) {

Wire.write(pgm\_read\_byte(&font5x8[n][i]));

}

Wire.write(0);

} else {

for (byte i = 0; i < 6; i++) {

Wire.write(0);

}

}

Wire.endTransmission();

m\_col += 6;

} else if (m\_font == FONT\_SIZE\_MEDIUM) {

write(n <= 9 ? ('0' + n) : ' ');

#ifndef MEMORY\_SAVING

} else if (m\_font == FONT\_SIZE\_LARGE) {

if (n <= 9) {

byte i;

WriteCommand(0xB0 + m\_row);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (i = 0; i < 16; i ++) {

byte d = pgm\_read\_byte(&digits16x16[n][i]);

Wire.write(d);

if (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) Wire.write(d);

}

Wire.endTransmission();

WriteCommand(0xB0 + m\_row + 1);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (; i < 32; i ++) {

byte d = pgm\_read\_byte(&digits16x16[n][i]);

Wire.write(d);

if (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) Wire.write(d);

}

Wire.endTransmission();

} else {

WriteCommand(0xB0 + m\_row);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 32 : 16; i > 0; i--) {

Wire.write(0);

}

Wire.endTransmission();

WriteCommand(0xB0 + m\_row + 1);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 32 : 16; i > 0; i--) {

Wire.write(0);

}

Wire.endTransmission();

}

m\_col += (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 30 : 16;

#endif

} else {

if (n <= 9) {

byte i;

WriteCommand(0xB0 + m\_row);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (i = 0; i < 16; i ++) {

byte d = pgm\_read\_byte(&digits16x24[n][i \* 3]);

Wire.write(d);

if (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) Wire.write(d);

}

Wire.endTransmission();

WriteCommand(0xB0 + m\_row + 1);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (i = 0; i < 16; i ++) {

byte d = pgm\_read\_byte(&digits16x24[n][i \* 3 + 1]);

Wire.write(d);

if (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) Wire.write(d);

}

Wire.endTransmission();

WriteCommand(0xB0 + m\_row + 2);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (i = 0; i < 16; i ++) {

byte d = pgm\_read\_byte(&digits16x24[n][i \* 3 + 2]);

Wire.write(d);

if (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) Wire.write(d);

}

Wire.endTransmission();

} else {

WriteCommand(0xB0 + m\_row);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 32 : 16; i > 0; i--) {

Wire.write(0);

}

Wire.endTransmission();

WriteCommand(0xB0 + m\_row + 1);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 32 : 16; i > 0; i--) {

Wire.write(0);

}

Wire.endTransmission();

WriteCommand(0xB0 + m\_row + 2);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte i = (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 32 : 16; i > 0; i--) {

Wire.write(0);

}

Wire.endTransmission();

}

m\_col += (m\_flags & FLAG\_PIXEL\_DOUBLE\_H) ? 30 : 16;

}

#ifdef TWBR

TWBR = twbrbackup;

#endif

}

void LCD\_SH1106::draw(const PROGMEM byte\* buffer, byte width, byte height)

{

#ifdef TWBR

uint8\_t twbrbackup = TWBR;

TWBR = 18; // upgrade to 400KHz!

#endif

WriteCommand(SSD1306\_SETLOWCOLUMN | 0x0); // low col = 0

WriteCommand(SSD1306\_SETHIGHCOLUMN | 0x0); // hi col = 0

WriteCommand(SSD1306\_SETSTARTLINE | 0x0); // line #0

const PROGMEM byte \*p = buffer;

height >>= 3;

width >>= 3;

for (byte i = 0; i < height; i++) {

// send a bunch of data in one xmission

WriteCommand(0xB0 + i + m\_row);//set page address

WriteCommand(m\_col & 0xf);//set lower column address

WriteCommand(0x10 | (m\_col >> 4));//set higher column address

for(byte j = 0; j < 8; j++){

Wire.beginTransmission(I2C\_ADDR);

Wire.write(0x40);

for (byte k = 0; k < width; k++, p++) {

Wire.write(pgm\_read\_byte(p));

}

Wire.endTransmission();

}

}

#ifdef TWBR

TWBR = twbrbackup;

#endif

m\_col += width;

}

void LCD\_SH1106::begin()

{

Wire.begin();

WriteCommand(0xAE); /\*display off\*/

WriteCommand(0x02); /\*set lower column address\*/

WriteCommand(0x10); /\*set higher column address\*/

WriteCommand(0x40); /\*set display start line\*/

WriteCommand(0xB0); /\*set page address\*/

WriteCommand(0x81); /\*contract control\*/

WriteCommand(0x80); /\*128\*/

WriteCommand(0xA1); /\*set segment remap\*/

WriteCommand(0xA6); /\*normal / reverse\*/

WriteCommand(0xA8); /\*multiplex ratio\*/

WriteCommand(0x3F); /\*duty = 1/32\*/

WriteCommand(0xad); /\*set charge pump enable\*/

WriteCommand(0x8b); /\*external VCC \*/

WriteCommand(0x30); /\*0X30---0X33 set VPP 9V liangdu!!!!\*/

WriteCommand(0xC8); /\*Com scan direction\*/

WriteCommand(0xD3); /\*set display offset\*/

WriteCommand(0x00); /\* 0x20 \*/

WriteCommand(0xD5); /\*set osc division\*/

WriteCommand(0x80);

WriteCommand(0xD9); /\*set pre-charge period\*/

WriteCommand(0x1f); /\*0x22\*/

WriteCommand(0xDA); /\*set COM pins\*/

WriteCommand(0x12);

WriteCommand(0xdb); /\*set vcomh\*/

WriteCommand(0x40);

WriteCommand(0xAF); /\*display ON\*/

clear();

}