/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Arduino Text Display Library for Multiple LCDs

\* Distributed under GPL v2.0

\* Copyright (c) 2013 Stanley Huang <stanleyhuangyc@live.com>

\* All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <Arduino.h>

#include <Wire.h>

#include "MicroLCD.h"

// fonts data

const PROGMEM unsigned char digits16x24[][48] = {

{0x00,0x00,0x00,0xF0,0xFF,0x0F,0xFC,0xFF,0x3F,0xFE,0xFF,0x7F,0xFE,0xFF,0x7F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x07,0x00,0xE0,0x07,0x00,0xE0,0x07,0x00,0xE0,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFE,0xFF,0x7F,0xFE,0xFF,0x7F,0xFC,0xFF,0x3F,0xF0,0xFF,0x0F},/\*"0",0\*/

{0x00,0x00,0x00,0x70,0x00,0x00,0x70,0x00,0x00,0x70,0x00,0x00,0x78,0x00,0x00,0xF8,0x00,0x00,0xFC,0xFF,0xFF,0xFE,0xFF,0xFF,0xFE,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00},/\*"1",0\*/

{0x00,0x00,0x00,0xF8,0x00,0xE0,0xFC,0x00,0xF8,0xFE,0x00,0xFE,0xFE,0x80,0xFF,0xFF,0xC0,0xFF,0x07,0xF0,0xFF,0x07,0xFC,0xFF,0x07,0xFF,0xEF,0xFF,0xFF,0xE3,0xFF,0xFF,0xE1,0xFE,0x7F,0xE0,0xFE,0x3F,0xE0,0xFC,0x0F,0xE0,0xF0,0x03,0x00,0x00,0x00,0x00},/\*"2",2\*/

{0x00,0x00,0x00,0xF8,0x80,0x1F,0xFE,0x80,0x3F,0xFE,0x80,0x7F,0xFF,0x80,0x7F,0xFF,0x80,0xFF,0xFF,0x9C,0xFF,0xFF,0x9C,0xFF,0x07,0x1C,0xE0,0x07,0x3E,0xE0,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFE,0xFF,0x7F,0xFE,0xF7,0x7F,0xFC,0xF7,0x3F,0xF0,0xE3,0x1F},/\*"3",3\*/

{0x00,0xF0,0x0F,0x00,0xFE,0x0F,0x80,0xFF,0x0F,0xE0,0xFF,0x0F,0xFC,0xBF,0x0F,0xFF,0x87,0x0F,0xFF,0x81,0x0F,0x3F,0x80,0x0F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x80,0x0F,0x00,0x80,0x0F},/\*"4",4\*/

{0x00,0x00,0x00,0xFF,0xC7,0x0F,0xFF,0xC7,0x3F,0xFF,0xC7,0x7F,0xFF,0xC7,0x7F,0xFF,0xC7,0xFF,0xFF,0xC7,0xFF,0x87,0x01,0xE0,0xC7,0x01,0xE0,0xC7,0x01,0xE0,0xC7,0xFF,0xFF,0xC7,0xFF,0xFF,0xC7,0xFF,0x7F,0x87,0xFF,0x7F,0x87,0xFF,0x3F,0x07,0xFE,0x1F},/\*"5",5\*/

{0x00,0x00,0x00,0xF0,0xFF,0x0F,0xFC,0xFF,0x3F,0xFE,0xFF,0x7F,0xFE,0xFF,0x7F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x07,0x06,0xE0,0x07,0x07,0xE0,0x07,0x07,0xE0,0x3F,0xFF,0xFF,0x3F,0xFF,0xFF,0x3E,0xFF,0x7F,0x3E,0xFE,0x7F,0x3C,0xFE,0x3F,0x38,0xF8,0x1F},/\*"6",6\*/

{0x00,0x00,0x00,0x07,0x00,0x00,0x07,0x00,0x00,0x07,0x00,0xC0,0x07,0x00,0xF8,0x07,0x00,0xFF,0x07,0xE0,0xFF,0x07,0xFE,0xFF,0xC7,0xFF,0xFF,0xFF,0xFF,0x3F,0xFF,0xFF,0x07,0xFF,0xFF,0x00,0xFF,0x0F,0x00,0xFF,0x01,0x00,0x1F,0x00,0x00,0x00,0x00,0x00},/\*"7",1\*/

{0x00,0x00,0x00,0xF0,0xE3,0x1F,0xFC,0xF7,0x3F,0xFE,0xFF,0x7F,0xFE,0xFF,0x7F,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x07,0x1C,0xE0,0x07,0x1C,0xE0,0x07,0x1C,0xE0,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFE,0xFF,0x7F,0xFE,0xF7,0x7F,0xFC,0xF7,0x3F,0xF0,0xE3,0x1F},/\*"8",8\*/

{0x00,0x00,0x00,0xF8,0x1F,0x1C,0xFC,0x7F,0x3C,0xFE,0x7F,0x7C,0xFE,0xFF,0x7C,0xFF,0xFF,0xFC,0xFF,0xFF,0xFC,0x07,0xE0,0xE0,0x07,0xE0,0xE0,0x07,0x60,0xE0,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFE,0xFF,0x7F,0xFE,0xFF,0x7F,0xFC,0xFF,0x3F,0xF0,0xFF,0x0F},/\*"9",9\*/

};

const PROGMEM unsigned char digits8x8[][8] = {

{0x3C,0x7E,0x83,0x81,0x81,0x7E,0x3C,0x00},/\*0\*/

{0x84,0x84,0x82,0xFF,0xFF,0x80,0x80,0x00},/\*1\*/

{0x84,0xC6,0xE1,0xA1,0xB1,0x9F,0x8E,0x00},/\*2\*/

{0x42,0xC3,0x81,0x89,0x89,0xFF,0x76,0x00},/\*3\*/

{0x20,0x38,0x24,0x22,0xFF,0xFF,0x20,0x00},/\*4\*/

{0x5F,0xDF,0x99,0x89,0x89,0xF9,0x70,0x00},/\*5\*/

{0x3C,0x7E,0x89,0x89,0x89,0xFB,0x72,0x00},/\*6\*/

{0x01,0x01,0xE1,0xF9,0x1D,0x07,0x01,0x00},/\*7\*/

{0x6E,0xFF,0x89,0x89,0x99,0xFF,0x76,0x00},/\*8\*/

{0x4E,0xDF,0x91,0x91,0x91,0x7F,0x3E,0x00},/\*9\*/

};

// The 7-bit ASCII character set...

const PROGMEM unsigned char font5x8[][5] = {

{ 0x00, 0x00, 0x5f, 0x00, 0x00 }, // 21 !

{ 0x00, 0x07, 0x00, 0x07, 0x00 }, // 22 "

{ 0x14, 0x7f, 0x14, 0x7f, 0x14 }, // 23 #

{ 0x24, 0x2a, 0x7f, 0x2a, 0x12 }, // 24 $

{ 0x23, 0x13, 0x08, 0x64, 0x62 }, // 25 %

{ 0x36, 0x49, 0x55, 0x22, 0x50 }, // 26 &

{ 0x00, 0x05, 0x03, 0x00, 0x00 }, // 27 '

{ 0x00, 0x1c, 0x22, 0x41, 0x00 }, // 28 (

{ 0x00, 0x41, 0x22, 0x1c, 0x00 }, // 29 )

{ 0x14, 0x08, 0x3e, 0x08, 0x14 }, // 2a \*

{ 0x08, 0x08, 0x3e, 0x08, 0x08 }, // 2b +

{ 0x00, 0x50, 0x30, 0x00, 0x00 }, // 2c ,

{ 0x08, 0x08, 0x08, 0x08, 0x08 }, // 2d -

{ 0x00, 0x60, 0x60, 0x00, 0x00 }, // 2e .

{ 0x20, 0x10, 0x08, 0x04, 0x02 }, // 2f /

{ 0x3e, 0x51, 0x49, 0x45, 0x3e }, // 30 0

{ 0x00, 0x42, 0x7f, 0x40, 0x00 }, // 31 1

{ 0x42, 0x61, 0x51, 0x49, 0x46 }, // 32 2

{ 0x21, 0x41, 0x45, 0x4b, 0x31 }, // 33 3

{ 0x18, 0x14, 0x12, 0x7f, 0x10 }, // 34 4

{ 0x27, 0x45, 0x45, 0x45, 0x39 }, // 35 5

{ 0x3c, 0x4a, 0x49, 0x49, 0x30 }, // 36 6

{ 0x01, 0x71, 0x09, 0x05, 0x03 }, // 37 7

{ 0x36, 0x49, 0x49, 0x49, 0x36 }, // 38 8

{ 0x06, 0x49, 0x49, 0x29, 0x1e }, // 39 9

{ 0x00, 0x36, 0x36, 0x00, 0x00 }, // 3a :

{ 0x00, 0x56, 0x36, 0x00, 0x00 }, // 3b ;

{ 0x08, 0x14, 0x22, 0x41, 0x00 }, // 3c <

{ 0x14, 0x14, 0x14, 0x14, 0x14 }, // 3d =

{ 0x00, 0x41, 0x22, 0x14, 0x08 }, // 3e >

{ 0x02, 0x01, 0x51, 0x09, 0x06 }, // 3f ?

{ 0x32, 0x49, 0x79, 0x41, 0x3e }, // 40 @

{ 0x7e, 0x11, 0x11, 0x11, 0x7e }, // 41 A

{ 0x7f, 0x49, 0x49, 0x49, 0x36 }, // 42 B

{ 0x3e, 0x41, 0x41, 0x41, 0x22 }, // 43 C

{ 0x7f, 0x41, 0x41, 0x22, 0x1c }, // 44 D

{ 0x7f, 0x49, 0x49, 0x49, 0x41 }, // 45 E

{ 0x7f, 0x09, 0x09, 0x09, 0x01 }, // 46 F

{ 0x3e, 0x41, 0x49, 0x49, 0x7a }, // 47 G

{ 0x7f, 0x08, 0x08, 0x08, 0x7f }, // 48 H

{ 0x00, 0x41, 0x7f, 0x41, 0x00 }, // 49 I

{ 0x20, 0x40, 0x41, 0x3f, 0x01 }, // 4a J

{ 0x7f, 0x08, 0x14, 0x22, 0x41 }, // 4b K

{ 0x7f, 0x40, 0x40, 0x40, 0x40 }, // 4c L

{ 0x7f, 0x02, 0x0c, 0x02, 0x7f }, // 4d M

{ 0x7f, 0x04, 0x08, 0x10, 0x7f }, // 4e N

{ 0x3e, 0x41, 0x41, 0x41, 0x3e }, // 4f O

{ 0x7f, 0x09, 0x09, 0x09, 0x06 }, // 50 P

{ 0x3e, 0x41, 0x51, 0x21, 0x5e }, // 51 Q

{ 0x7f, 0x09, 0x19, 0x29, 0x46 }, // 52 R

{ 0x46, 0x49, 0x49, 0x49, 0x31 }, // 53 S

{ 0x01, 0x01, 0x7f, 0x01, 0x01 }, // 54 T

{ 0x3f, 0x40, 0x40, 0x40, 0x3f }, // 55 U

{ 0x1f, 0x20, 0x40, 0x20, 0x1f }, // 56 V

{ 0x3f, 0x40, 0x38, 0x40, 0x3f }, // 57 W

{ 0x63, 0x14, 0x08, 0x14, 0x63 }, // 58 X

{ 0x07, 0x08, 0x70, 0x08, 0x07 }, // 59 Y

{ 0x61, 0x51, 0x49, 0x45, 0x43 }, // 5a Z

{ 0x00, 0x7f, 0x41, 0x41, 0x00 }, // 5b [

{ 0x02, 0x04, 0x08, 0x10, 0x20 }, // 5c backslash

{ 0x00, 0x41, 0x41, 0x7f, 0x00 }, // 5d ]

{ 0x04, 0x02, 0x01, 0x02, 0x04 }, // 5e ^

{ 0x40, 0x40, 0x40, 0x40, 0x40 }, // 5f \_

{ 0x00, 0x01, 0x02, 0x04, 0x00 }, // 60 `

{ 0x20, 0x54, 0x54, 0x54, 0x78 }, // 61 a

{ 0x7f, 0x48, 0x44, 0x44, 0x38 }, // 62 b

{ 0x38, 0x44, 0x44, 0x44, 0x20 }, // 63 c

{ 0x38, 0x44, 0x44, 0x48, 0x7f }, // 64 d

{ 0x38, 0x54, 0x54, 0x54, 0x18 }, // 65 e

{ 0x08, 0x7e, 0x09, 0x01, 0x02 }, // 66 f

{ 0x0c, 0x52, 0x52, 0x52, 0x3e }, // 67 g

{ 0x7f, 0x08, 0x04, 0x04, 0x78 }, // 68 h

{ 0x00, 0x44, 0x7d, 0x40, 0x00 }, // 69 i

{ 0x20, 0x40, 0x44, 0x3d, 0x00 }, // 6a j

{ 0x7f, 0x10, 0x28, 0x44, 0x00 }, // 6b k

{ 0x00, 0x41, 0x7f, 0x40, 0x00 }, // 6c l

{ 0x7c, 0x04, 0x18, 0x04, 0x78 }, // 6d m

{ 0x7c, 0x08, 0x04, 0x04, 0x78 }, // 6e n

{ 0x38, 0x44, 0x44, 0x44, 0x38 }, // 6f o

{ 0x7c, 0x14, 0x14, 0x14, 0x08 }, // 70 p

{ 0x08, 0x14, 0x14, 0x18, 0x7c }, // 71 q

{ 0x7c, 0x08, 0x04, 0x04, 0x08 }, // 72 r

{ 0x48, 0x54, 0x54, 0x54, 0x20 }, // 73 s

{ 0x04, 0x3f, 0x44, 0x40, 0x20 }, // 74 t

{ 0x3c, 0x40, 0x40, 0x20, 0x7c }, // 75 u

{ 0x1c, 0x20, 0x40, 0x20, 0x1c }, // 76 v

{ 0x3c, 0x40, 0x30, 0x40, 0x3c }, // 77 w

{ 0x44, 0x28, 0x10, 0x28, 0x44 }, // 78 x

{ 0x0c, 0x50, 0x50, 0x50, 0x3c }, // 79 y

{ 0x44, 0x64, 0x54, 0x4c, 0x44 }, // 7a z

{ 0x00, 0x08, 0x36, 0x41, 0x00 }, // 7b {

{ 0x00, 0x00, 0x7f, 0x00, 0x00 }, // 7c |

{ 0x00, 0x41, 0x36, 0x08, 0x00 }, // 7d }

{ 0x10, 0x08, 0x08, 0x10, 0x08 }, // 7e ~

};

#ifndef MEMORY\_SAVING

const PROGMEM unsigned char digits16x16[][32] = {

{0x00,0xE0,0xF8,0xFC,0xFE,0x1E,0x07,0x07,0x07,0x07,0x1E,0xFE,0xFC,0xF8,0xF0,0x00,0x00,0x07,0x0F,0x3F,0x3F,0x7C,0x70,0x70,0x70,0x70,0x7C,0x3F,0x1F,0x1F,0x07,0x00},/\*0\*/

{0x00,0x00,0x00,0x06,0x07,0x07,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7F,0x7F,0x7F,0x7F,0x00,0x00,0x00,0x00,0x00,0x00},/\*1\*/

{0x00,0x38,0x3C,0x3E,0x3E,0x0F,0x07,0x07,0x07,0xCF,0xFF,0xFE,0xFE,0x38,0x00,0x00,0x00,0x40,0x40,0x60,0x70,0x78,0x7C,0x7E,0x7F,0x77,0x73,0x71,0x70,0x70,0x00,0x00},/\*2\*/

{0x00,0x18,0x1C,0x1E,0x1E,0x0F,0xC7,0xC7,0xE7,0xFF,0xFE,0xBE,0x9C,0x00,0x00,0x00,0x00,0x0C,0x1C,0x3C,0x3C,0x78,0x70,0x70,0x70,0x79,0x7F,0x3F,0x1F,0x0F,0x00,0x00},/\*3\*/

{0x00,0x00,0x80,0xC0,0xE0,0x70,0x38,0x1C,0x1E,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x06,0x07,0x07,0x07,0x06,0x06,0x06,0x06,0x06,0x7F,0x7F,0x7F,0x7F,0x06,0x06,0x00},/\*4\*/

{0x00,0x00,0x00,0x00,0xF0,0xFF,0xFF,0xFF,0xE7,0xE7,0xE7,0xE7,0xC7,0x87,0x00,0x00,0x00,0x00,0x38,0x78,0x71,0x70,0x70,0x70,0x70,0x70,0x39,0x3F,0x3F,0x1F,0x0F,0x00},/\*5\*/

{0x00,0x80,0xE0,0xF0,0xF8,0xFC,0x7F,0x7F,0x6F,0x67,0xE1,0xE1,0xC0,0x80,0x00,0x00,0x00,0x0F,0x1F,0x3F,0x3F,0x78,0x70,0x70,0x70,0x70,0x78,0x3F,0x3F,0x1F,0x0F,0x00},/\*6\*/

{0x00,0x07,0x07,0x07,0x07,0x07,0xC7,0xE7,0xF7,0xFF,0x7F,0x3F,0x1F,0x07,0x03,0x01,0x00,0x20,0x38,0x7C,0x7E,0x3F,0x0F,0x07,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00},/\*7\*/

{0x00,0x00,0x00,0x1C,0xBE,0xFE,0xFF,0xE7,0xC3,0xC3,0xE7,0xFF,0xFE,0xBE,0x1C,0x00,0x00,0x00,0x0E,0x3F,0x3F,0x7F,0x71,0x60,0x60,0x60,0x71,0x7F,0x3F,0x3F,0x0F,0x00},/\*8\*/

{0x00,0x78,0xFC,0xFE,0xFE,0x8F,0x07,0x07,0x07,0x07,0x8F,0xFE,0xFE,0xFC,0xF8,0x00,0x00,0x00,0x00,0x01,0x43,0x43,0x73,0x7B,0x7F,0x7F,0x1F,0x0F,0x07,0x03,0x00,0x00},/\*9\*/

};

const PROGMEM unsigned char font8x16\_terminal[][16] = {

{0x00,0x00,0x00,0x00,0x7C,0x00,0xFE,0x1B,0xFE,0x1B,0x7C,0x00,0x00,0x00,0x00,0x00},/\*"!",0\*/

{0x00,0x00,0x0E,0x00,0x1E,0x00,0x00,0x00,0x00,0x00,0x1E,0x00,0x0E,0x00,0x00,0x00},/\*""",1\*/

{0x20,0x01,0xFC,0x0F,0xFC,0x0F,0x20,0x01,0x20,0x01,0xFC,0x0F,0xFC,0x0F,0x20,0x01},/\*"#",2\*/

{0x38,0x06,0x7C,0x0C,0x44,0x08,0xFF,0x3F,0xFF,0x3F,0x84,0x08,0x8C,0x0F,0x18,0x07},/\*"$",3\*/

{0x1C,0x18,0x14,0x1E,0x9C,0x07,0xE0,0x01,0x78,0x1C,0x1E,0x14,0x06,0x1C,0x00,0x00},/\*"%",4\*/

{0xBC,0x1F,0xFE,0x10,0x42,0x10,0xC2,0x10,0xFE,0x1F,0x3C,0x0F,0x80,0x19,0x80,0x10},/\*"&",5\*/

{0x00,0x00,0x00,0x00,0x10,0x00,0x1E,0x00,0x0E,0x00,0x00,0x00,0x00,0x00,0x00,0x00},/\*"'",6\*/

{0x00,0x00,0x00,0x00,0xF0,0x07,0xFC,0x1F,0x0E,0x38,0x02,0x20,0x00,0x00,0x00,0x00},/\*"(",7\*/

{0x00,0x00,0x00,0x00,0x02,0x20,0x0E,0x38,0xFC,0x1F,0xF0,0x07,0x00,0x00,0x00,0x00},/\*")",8\*/

{0x80,0x00,0xA0,0x02,0xE0,0x03,0xC0,0x01,0xC0,0x01,0xE0,0x03,0xA0,0x02,0x80,0x00},/\*"\*",9\*/

{0x80,0x00,0x80,0x00,0x80,0x00,0xE0,0x03,0xE0,0x03,0x80,0x00,0x80,0x00,0x80,0x00},/\*"+",10\*/

{0x00,0x00,0x00,0x00,0x00,0x40,0x00,0x78,0x00,0x38,0x00,0x00,0x00,0x00,0x00,0x00},/\*",",11\*/

{0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00},/\*"-",12\*/

{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x18,0x00,0x18,0x00,0x00,0x00,0x00,0x00,0x00},/\*".",13\*/

{0x00,0x18,0x00,0x1E,0x80,0x07,0xE0,0x01,0x78,0x00,0x1E,0x00,0x06,0x00,0x00,0x00},/\*"/",14\*/

{0xF8,0x07,0xFC,0x0F,0x06,0x18,0xC2,0x10,0xC2,0x10,0x06,0x18,0xFC,0x0F,0xF8,0x07},/\*"0",15\*/

{0x00,0x00,0x08,0x10,0x0C,0x10,0xFE,0x1F,0xFE,0x1F,0x00,0x10,0x00,0x10,0x00,0x00},/\*"1",16\*/

{0x04,0x1C,0x06,0x1E,0x02,0x13,0x82,0x11,0xC2,0x10,0x62,0x10,0x3E,0x18,0x1C,0x18},/\*"2",17\*/

{0x04,0x08,0x06,0x18,0x02,0x10,0x42,0x10,0x42,0x10,0x42,0x10,0xFE,0x1F,0xBC,0x0F},/\*"3",18\*/

{0xC0,0x01,0xE0,0x01,0x30,0x01,0x18,0x01,0x0C,0x11,0xFE,0x1F,0xFE,0x1F,0x00,0x11},/\*"4",19\*/

{0x7E,0x08,0x7E,0x18,0x42,0x10,0x42,0x10,0x42,0x10,0x42,0x10,0xC2,0x1F,0x82,0x0F},/\*"5",20\*/

{0xF8,0x0F,0xFC,0x1F,0x46,0x10,0x42,0x10,0x42,0x10,0x42,0x10,0xC0,0x1F,0x80,0x0F},/\*"6",21\*/

{0x06,0x00,0x06,0x00,0x02,0x00,0x02,0x1F,0xC2,0x1F,0xF2,0x00,0x3E,0x00,0x0E,0x00},/\*"7",22\*/

{0xBC,0x0F,0xFE,0x1F,0x42,0x10,0x42,0x10,0x42,0x10,0x42,0x10,0xFE,0x1F,0xBC,0x0F},/\*"8",23\*/

{0x3C,0x00,0x7E,0x10,0x42,0x10,0x42,0x10,0x42,0x10,0x42,0x18,0xFE,0x0F,0xFC,0x07},/\*"9",24\*/

{0x00,0x00,0x00,0x00,0x00,0x00,0x30,0x0C,0x30,0x0C,0x00,0x00,0x00,0x00,0x00,0x00},/\*":",26\*/

{0x00,0x00,0x00,0x00,0x00,0x20,0x60,0x3C,0x60,0x1C,0x00,0x00,0x00,0x00,0x00,0x00},/\*";",27\*/

{0x80,0x00,0xC0,0x01,0x60,0x03,0x30,0x06,0x18,0x0C,0x0C,0x18,0x04,0x10,0x00,0x00},/\*"<",28\*/

{0x40,0x02,0x40,0x02,0x40,0x02,0x40,0x02,0x40,0x02,0x40,0x02,0x40,0x02,0x40,0x02},/\*"=",29\*/

{0x04,0x10,0x0C,0x18,0x18,0x0C,0x30,0x06,0x60,0x03,0xC0,0x01,0x80,0x00,0x00,0x00},/\*">",30\*/

{0x04,0x00,0x06,0x00,0x02,0x00,0x82,0x1B,0xC2,0x1B,0x62,0x00,0x3E,0x00,0x1C,0x00},/\*"?",31\*/

{0xFC,0x0F,0xFE,0x1F,0x02,0x10,0x82,0x11,0xC2,0x13,0xE2,0x13,0xFE,0x13,0xFC,0x03},/\*"@",32\*/

{0xF0,0x1F,0xF8,0x1F,0x0C,0x01,0x06,0x01,0x06,0x01,0x0C,0x01,0xF8,0x1F,0xF0,0x1F},/\*"A",33\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x42,0x10,0x42,0x10,0x42,0x10,0xFE,0x1F,0xBC,0x0F},/\*"B",34\*/

{0xF8,0x07,0xFC,0x0F,0x06,0x18,0x02,0x10,0x02,0x10,0x02,0x10,0x06,0x18,0x0C,0x0C},/\*"C",35\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x02,0x10,0x02,0x10,0x06,0x18,0xFC,0x0F,0xF8,0x07},/\*"D",36\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x42,0x10,0x42,0x10,0xE2,0x10,0x06,0x18,0x06,0x18},/\*"E",37\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x42,0x10,0x42,0x00,0xE2,0x00,0x06,0x00,0x06,0x00},/\*"F",38\*/

{0xF8,0x07,0xFC,0x0F,0x06,0x18,0x02,0x10,0x82,0x10,0x82,0x10,0x86,0x0F,0x8C,0x1F},/\*"G",39\*/

{0xFE,0x1F,0xFE,0x1F,0x40,0x00,0x40,0x00,0x40,0x00,0x40,0x00,0xFE,0x1F,0xFE,0x1F},/\*"H",40\*/

{0x00,0x00,0x02,0x10,0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x02,0x10,0x02,0x10,0x00,0x00},/\*"I",41\*/

{0x00,0x0C,0x00,0x1C,0x00,0x10,0x00,0x10,0x02,0x10,0xFE,0x1F,0xFE,0x0F,0x02,0x00},/\*"J",42\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0xE0,0x00,0xB0,0x01,0x18,0x03,0x0E,0x1E,0x06,0x1C},/\*"K",43\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x02,0x10,0x00,0x10,0x00,0x10,0x00,0x18,0x00,0x18},/\*"L",44\*/

{0xFE,0x1F,0xFE,0x1F,0x18,0x00,0xF0,0x00,0xF0,0x00,0x18,0x00,0xFE,0x1F,0xFE,0x1F},/\*"M",45\*/

{0xFE,0x1F,0xFE,0x1F,0x38,0x00,0x70,0x00,0xE0,0x00,0xC0,0x01,0xFE,0x1F,0xFE,0x1F},/\*"N",46\*/

{0xFC,0x0F,0xFE,0x1F,0x02,0x10,0x02,0x10,0x02,0x10,0x02,0x10,0xFE,0x1F,0xFC,0x0F},/\*"O",47\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x42,0x10,0x42,0x00,0x42,0x00,0x7E,0x00,0x3C,0x00},/\*"P",48\*/

{0xFC,0x0F,0xFE,0x1F,0x02,0x10,0x02,0x1C,0x02,0x38,0x02,0x70,0xFE,0x5F,0xFC,0x0F},/\*"Q",49\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x42,0x00,0x42,0x00,0xC2,0x00,0xFE,0x1F,0x3C,0x1F},/\*"R",50\*/

{0x1C,0x0C,0x3E,0x1C,0x62,0x10,0x42,0x10,0x42,0x10,0xC2,0x10,0x8E,0x1F,0x0C,0x0F},/\*"S",51\*/

{0x06,0x00,0x06,0x00,0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x02,0x10,0x06,0x00,0x06,0x00},/\*"T",52\*/

{0xFE,0x0F,0xFE,0x1F,0x00,0x10,0x00,0x10,0x00,0x10,0x00,0x10,0xFE,0x1F,0xFE,0x0F},/\*"U",53\*/

{0xFE,0x03,0xFE,0x07,0x00,0x0C,0x00,0x18,0x00,0x18,0x00,0x0C,0xFE,0x07,0xFE,0x03},/\*"V",54\*/

{0xFE,0x07,0xFE,0x1F,0x00,0x1C,0xC0,0x07,0xC0,0x07,0x00,0x1C,0xFE,0x1F,0xFE,0x07},/\*"W",55\*/

{0x0E,0x1C,0x1E,0x1E,0x30,0x03,0xE0,0x01,0xE0,0x01,0x30,0x03,0x1E,0x1E,0x0E,0x1C},/\*"X",56\*/

{0x1E,0x00,0x3E,0x00,0x60,0x10,0xC0,0x1F,0xC0,0x1F,0x60,0x10,0x3E,0x00,0x1E,0x00},/\*"Y",57\*/

{0x06,0x1E,0x06,0x1F,0x82,0x11,0xC2,0x10,0x62,0x10,0x32,0x10,0x1E,0x18,0x0E,0x18},/\*"Z",58\*/

{0x00,0x00,0x00,0x00,0xFE,0x1F,0xFE,0x1F,0x02,0x10,0x02,0x10,0x00,0x00,0x00,0x00},/\*"[",59\*/

{0x00,0x18,0x00,0x1E,0x80,0x07,0xE0,0x01,0x78,0x00,0x1E,0x00,0x06,0x00,0x00,0x00},/\*"/",60\*/

{0x00,0x00,0x00,0x00,0x02,0x10,0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x00,0x00,0x00,0x00},/\*"]",61\*/

{0x20,0x00,0x30,0x00,0x18,0x00,0x0C,0x00,0x18,0x00,0x30,0x00,0x20,0x00,0x00,0x00},/\*"^",62\*/

{0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x80},/\*"\_",63\*/

{0x00,0x00,0x00,0x00,0x00,0x00,0x38,0x00,0x78,0x00,0x40,0x00,0x00,0x00,0x00,0x00},/\*"`",64\*/

{0x00,0x0E,0x20,0x1F,0x20,0x11,0x20,0x11,0x20,0x11,0xE0,0x0F,0xC0,0x1F,0x00,0x10},/\*"a",65\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x0F,0x20,0x10,0x20,0x10,0x60,0x10,0xC0,0x1F,0x80,0x0F},/\*"b",66\*/

{0xC0,0x0F,0xE0,0x1F,0x20,0x10,0x20,0x10,0x20,0x10,0x20,0x10,0x60,0x18,0x40,0x08},/\*"c",67\*/

{0x80,0x0F,0xC0,0x1F,0x60,0x10,0x20,0x10,0x22,0x10,0xFE,0x0F,0xFE,0x1F,0x00,0x10},/\*"d",68\*/

{0xC0,0x0F,0xE0,0x1F,0x20,0x11,0x20,0x11,0x20,0x11,0x20,0x11,0xE0,0x19,0xC0,0x09},/\*"e",69\*/

{0x00,0x00,0x20,0x10,0xFC,0x1F,0xFE,0x1F,0x22,0x10,0x22,0x00,0x06,0x00,0x04,0x00},/\*"f",70\*/

{0xC0,0x4F,0xE0,0xDF,0x20,0x90,0x20,0x90,0x20,0x90,0xC0,0xFF,0xE0,0x7F,0x20,0x00},/\*"g",71\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x40,0x00,0x20,0x00,0x20,0x00,0xE0,0x1F,0xC0,0x1F},/\*"h",72\*/

{0x00,0x00,0x20,0x10,0x20,0x10,0xEC,0x1F,0xEC,0x1F,0x00,0x10,0x00,0x10,0x00,0x00},/\*"i",73\*/

{0x00,0x60,0x00,0xC0,0x20,0x80,0x20,0x80,0xEC,0xFF,0xEC,0x7F,0x00,0x00,0x00,0x00},/\*"j",74\*/

{0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x80,0x01,0x80,0x03,0xC0,0x06,0x60,0x1C,0x20,0x18},/\*"k",75\*/

{0x00,0x00,0x02,0x10,0x02,0x10,0xFE,0x1F,0xFE,0x1F,0x00,0x10,0x00,0x10,0x00,0x00},/\*"l",76\*/

{0xE0,0x1F,0xE0,0x1F,0x60,0x00,0xC0,0x0F,0xC0,0x0F,0x60,0x00,0xE0,0x1F,0xC0,0x1F},/\*"m",77\*/

{0x20,0x00,0xE0,0x1F,0xC0,0x1F,0x20,0x00,0x20,0x00,0x20,0x00,0xE0,0x1F,0xC0,0x1F},/\*"n",78\*/

{0xC0,0x0F,0xE0,0x1F,0x20,0x10,0x20,0x10,0x20,0x10,0x20,0x10,0xE0,0x1F,0xC0,0x0F},/\*"o",79\*/

{0x20,0x80,0xE0,0xFF,0xC0,0xFF,0x20,0x90,0x20,0x10,0x20,0x10,0xE0,0x1F,0xC0,0x0F},/\*"p",80\*/

{0xC0,0x0F,0xE0,0x1F,0x20,0x10,0x20,0x10,0x20,0x90,0xC0,0xFF,0xE0,0xFF,0x20,0x80},/\*"q",81\*/

{0x20,0x10,0xE0,0x1F,0xC0,0x1F,0x60,0x10,0x20,0x00,0x20,0x00,0x60,0x00,0x40,0x00},/\*"r",82\*/

{0xC0,0x08,0xE0,0x19,0x20,0x11,0x20,0x11,0x20,0x13,0x20,0x12,0x60,0x1E,0x40,0x0C},/\*"s",83\*/

{0x20,0x00,0x20,0x00,0xFC,0x0F,0xFE,0x1F,0x20,0x10,0x20,0x18,0x00,0x08,0x00,0x00},/\*"t",84\*/

{0xE0,0x0F,0xE0,0x1F,0x00,0x10,0x00,0x10,0x00,0x10,0xE0,0x0F,0xE0,0x1F,0x00,0x10},/\*"u",85\*/

{0xE0,0x03,0xE0,0x07,0x00,0x0C,0x00,0x18,0x00,0x18,0x00,0x0C,0xE0,0x07,0xE0,0x03},/\*"v",86\*/

{0xE0,0x0F,0xE0,0x1F,0x00,0x18,0x00,0x0F,0x00,0x0F,0x00,0x18,0xE0,0x1F,0xE0,0x0F},/\*"w",87\*/

{0x20,0x10,0x60,0x18,0xC0,0x0C,0x80,0x07,0x80,0x07,0xC0,0x0C,0x60,0x18,0x20,0x10},/\*"x",88\*/

{0xE0,0x8F,0xE0,0x9F,0x00,0x90,0x00,0x90,0x00,0x90,0x00,0xD0,0xE0,0x7F,0xE0,0x3F},/\*"y",89\*/

{0x60,0x18,0x60,0x1C,0x20,0x16,0x20,0x13,0xA0,0x11,0xE0,0x10,0x60,0x18,0x20,0x18},/\*"z",90\*/

{0x00,0x00,0x00,0x00,0x80,0x00,0xFC,0x1F,0x7E,0x3F,0x02,0x20,0x02,0x20,0x00,0x00},/\*"{",91\*/

{0x00,0x00,0x00,0x00,0x00,0x00,0x7C,0x3E,0x7C,0x3E,0x00,0x00,0x00,0x00,0x00,0x00},/\*"|",92\*/

{0x00,0x00,0x02,0x20,0x02,0x20,0x7E,0x3F,0xFC,0x1F,0x80,0x00,0x00,0x00,0x00,0x00},/\*"}",93\*/

};

#endif

void LCD\_Common::printInt(uint16\_t value, int8\_t padding)

{

uint16\_t den = 10000;

for (int8\_t i = 5; i > 0; i--) {

byte v = (byte)(value / den);

value -= v \* den;

den /= 10;

if (v == 0 && padding && den) {

if (padding >= i) {

writeDigit((m\_flags & FLAG\_PAD\_ZERO) ? 0 : -1);

}

continue;

}

padding = 0;

writeDigit(v);

}

}

void LCD\_Common::printLong(uint32\_t value, int8\_t padding)

{

uint32\_t den = 1000000000;

for (int8\_t i = 10; i > 0; i--) {

byte v = (byte)(value / den);

value -= v \* den;

den /= 10;

if (v == 0 && padding && den) {

if (padding >= i) {

writeDigit((m\_flags & FLAG\_PAD\_ZERO) ? 0 : -1);

}

continue;

}

padding = 0;

writeDigit(v);

}

}

void LCD\_SSD1306::setCursor(byte column, byte line)

{

m\_col = column;

m\_row = line;

ssd1306\_command(0xB0 + m\_row);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

}

size\_t LCD\_SSD1306::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(\_i2caddr);

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;

ssd1306\_command(0xB0 + m\_row);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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();

ssd1306\_command(0xB0 + m\_row + 1);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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 {

ssd1306\_command(0xB0 + m\_row);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

Wire.write(0x40);

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

Wire.write(0);

}

Wire.endTransmission();

ssd1306\_command(0xB0 + m\_row + 1);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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\_SSD1306::writeDigit(byte n)

{

#ifdef TWBR

uint8\_t twbrbackup = TWBR;

TWBR = 18; // upgrade to 400KHz!

#endif

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

Wire.beginTransmission(\_i2caddr);

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;

ssd1306\_command(0xB0 + m\_row);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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();

ssd1306\_command(0xB0 + m\_row + 1);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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 {

ssd1306\_command(0xB0 + m\_row);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

Wire.write(0x40);

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

Wire.write(0);

}

Wire.endTransmission();

ssd1306\_command(0xB0 + m\_row + 1);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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;

ssd1306\_command(0xB0 + m\_row);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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();

ssd1306\_command(0xB0 + m\_row + 1);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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();

ssd1306\_command(0xB0 + m\_row + 2);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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 {

ssd1306\_command(0xB0 + m\_row);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

Wire.write(0x40);

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

Wire.write(0);

}

Wire.endTransmission();

ssd1306\_command(0xB0 + m\_row + 1);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

Wire.write(0x40);

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

Wire.write(0);

}

Wire.endTransmission();

ssd1306\_command(0xB0 + m\_row + 2);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

Wire.beginTransmission(\_i2caddr);

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\_SSD1306::draw(const PROGMEM byte\* buffer, byte width, byte height)

{

ssd1306\_command(SSD1306\_SETLOWCOLUMN | 0x0); // low col = 0

ssd1306\_command(SSD1306\_SETHIGHCOLUMN | 0x0); // hi col = 0

ssd1306\_command(SSD1306\_SETSTARTLINE | 0x0); // line #0

const PROGMEM byte \*p = buffer;

height >>= 3;

width >>= 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

ssd1306\_command(0xB0 + i + m\_row);//set page address

ssd1306\_command(m\_col & 0xf);//set lower column address

ssd1306\_command(0x10 | (m\_col >> 4));//set higher column address

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

Wire.beginTransmission(\_i2caddr);

Wire.write(0x40);

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

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

}

Wire.endTransmission();

}

}

m\_col += width;

#ifdef TWBR

TWBR = twbrbackup;

#endif

}

void LCD\_SSD1306::clearLine(byte line)

{

ssd1306\_command(SSD1306\_SETLOWCOLUMN | 0x0); // low col = 0

ssd1306\_command(SSD1306\_SETHIGHCOLUMN | 0x0); // hi col = 0

ssd1306\_command(SSD1306\_SETSTARTLINE | 0x0); // line #0

#ifdef TWBR

uint8\_t twbrbackup = TWBR;

TWBR = 18; // upgrade to 400KHz!

#endif

// send a bunch of data in one xmission

ssd1306\_command(0xB0 + line);//set page address

ssd1306\_command(0);//set lower column address

ssd1306\_command(0x10);//set higher column address

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

Wire.beginTransmission(\_i2caddr);

Wire.write(0x40);

for (byte k = 0; k < 16; k++) {

Wire.write(0);

}

Wire.endTransmission();

}

#ifdef TWBR

TWBR = twbrbackup;

#endif

}

void LCD\_SSD1306::clear(byte x, byte y, byte width, byte height)

{

ssd1306\_command(SSD1306\_SETLOWCOLUMN | 0x0); // low col = 0

ssd1306\_command(SSD1306\_SETHIGHCOLUMN | 0x0); // hi col = 0

ssd1306\_command(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

ssd1306\_command(0xB0 + i + y);//set page address

ssd1306\_command(x & 0xf);//set lower column address

ssd1306\_command(0x10 | (x >> 4));//set higher column address

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

Wire.beginTransmission(\_i2caddr);

Wire.write(0x40);

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

Wire.write(0);

}

Wire.endTransmission();

}

}

#ifdef TWBR

TWBR = twbrbackup;

#endif

setCursor(0, 0);

}

void LCD\_SSD1306::setContrast(byte Contrast)

{

ssd1306\_command(SSD1306\_SETCONTRAST);

ssd1306\_command(Contrast);

}