/\*

u8g\_dev\_ili9325d\_320x240.c

Universal 8bit Graphics Library

Copyright (c) 2011, olikraus@gmail.com

All rights reserved.

Redistribution and use in source and binary forms, with or without modification,

are permitted provided that the following conditions are met:

\* Redistributions of source code must retain the above copyright notice, this list

of conditions and the following disclaimer.

\* Redistributions in binary form must reproduce the above copyright notice, this

list of conditions and the following disclaimer in the documentation and/or other

materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND

CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,

INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE

DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR

CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,

SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT

NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;

LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER

CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,

STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)

ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF

ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Color format

Red: 5 Bit

Green: 6 Bit

Blue: 5 Bit

\*/

#include "u8g.h"

#define WIDTH 240

#if defined(U8G\_16BIT)

#define HEIGHT 320

#else

/\* if the user tries to compile the 8Bit version of the lib, then restrict the height to something which fits to 8Bit \*/

#define HEIGHT 240

#endif

#define PAGE\_HEIGHT 4

/\*

reference board for this device:

http://iteadstudio.com/store/index.php?main\_page=product\_info&cPath=57\_58&products\_id=55

documentation:

http://iteadstudio.com/Downloadfile/ITDB02\_material.rar

datasheet

http://www.newhavendisplay.com/app\_notes/ILI9325D.pdf

other libs

http://henningkarlsen.com/electronics/library.php

init sequence

http://code.google.com/p/itdb02/, ITDB02.cpp, iteadstudio.com

\*/

static const uint8\_t u8g\_dev\_ili9325d\_320x240\_init\_seq[] PROGMEM = {

U8G\_ESC\_CS(0), /\* disable chip \*/

U8G\_ESC\_DLY(50), /\* delay 50 ms \*/

U8G\_ESC\_RST(15), /\* do reset low pulse with (15\*16)+2 milliseconds (=maximum delay)\*/

U8G\_ESC\_DLY(50), /\* delay 50 ms \*/

U8G\_ESC\_RST(15), /\* do reset low pulse with (15\*16)+2 milliseconds (=maximum delay)\*/

U8G\_ESC\_DLY(50), /\* delay 50 ms \*/

U8G\_ESC\_CS(1), /\* enable chip \*/

U8G\_ESC\_DLY(50), /\* delay 50 ms \*/

//U8G\_ESC\_ADR(0), 0x000, 0x0E5, /\* only used for none D version: set SRAM internal timing \*/

//U8G\_ESC\_ADR(1), 0x078, 0x0f0,

U8G\_ESC\_ADR(0), 0x000, 0x001, /\* Driver Output Control, bits 8 & 10 \*/

U8G\_ESC\_ADR(1), 0x001, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x002, /\* LCD Driving Wave Control, bit 9: Set line inversion \*/

U8G\_ESC\_ADR(1), 0x002, 0x000, /\* ITDB02 none D verion: 0x007, 0x000 \*/

U8G\_ESC\_ADR(0), 0x000, 0x003, /\* Entry Mode, GRAM write direction and BGR=1 \*/

U8G\_ESC\_ADR(1), 0x010, 0x030,

U8G\_ESC\_ADR(0), 0x000, 0x004, /\* Resize register \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x008, /\* Display Control 2: set the back porch and front porch \*/

U8G\_ESC\_ADR(1), 0x002, 0x007,

U8G\_ESC\_ADR(0), 0x000, 0x009, /\* Display Control 3 \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x00a, /\* Display Control 4: FMARK \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x00c, /\* RGB Display Interface Control 1 \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x00d, /\* Frame Maker Position \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x00f, /\* RGB Display Interface Control 2 \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x010, /\* Power Control 1: SAP, BT[3:0], AP, DSTB, SLP, STB \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x011, /\* Power Control 2: DC1[2:0], DC0[2:0], VC[2:0] \*/

U8G\_ESC\_ADR(1), 0x000, 0x007,

U8G\_ESC\_ADR(0), 0x000, 0x012, /\* Power Control 3: VREG1OUT voltage \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x013, /\* Power Control 4: VDV[4:0] for VCOM amplitude \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x007, /\* Display Control 1: Operate, but do not display \*/

U8G\_ESC\_ADR(1), 0x000, 0x001,

U8G\_ESC\_DLY(100), /\* delay 100 ms \*/ /\* ITDB02 none D verion: 50ms \*/

U8G\_ESC\_DLY(100), /\* delay 100 ms \*/

U8G\_ESC\_ADR(0), 0x000, 0x010, /\* Power Control 1: SAP, BT[3:0], AP, DSTB, SLP, STB \*/

U8G\_ESC\_ADR(1), 0x016, 0x090, /\* ITDB02 none D verion: 0x010, 0x090 \*/

U8G\_ESC\_ADR(0), 0x000, 0x011, /\* Power Control 2: SAP, BT[3:0], AP, DSTB, SLP, STB \*/

U8G\_ESC\_ADR(1), 0x002, 0x027,

U8G\_ESC\_DLY(50), /\* delay 50 ms \*/

U8G\_ESC\_ADR(0), 0x000, 0x012, /\* Power Control 3: VCI: External, VCI\*1.80 \*/

U8G\_ESC\_ADR(1), 0x000, 0x00d, /\* ITDB02 none D verion: 0x000, 0x01f \*/

U8G\_ESC\_DLY(50), /\* delay 50 ms \*/

U8G\_ESC\_ADR(0), 0x000, 0x013, /\* Power Control 4: VDV[4:0] for VCOM amplitude \*/

U8G\_ESC\_ADR(1), 0x012, 0x000, /\* ITDB02 none D verion: 0x015, 0x000 \*/

U8G\_ESC\_ADR(0), 0x000, 0x029, /\* Power Control 7 \*/

U8G\_ESC\_ADR(1), 0x000, 0x00a, /\* ITDB02 none D verion: 0x000, 0x027 \*/

U8G\_ESC\_ADR(0), 0x000, 0x02b, /\* Frame Rate: 83 \*/

U8G\_ESC\_ADR(1), 0x000, 0x00d,

U8G\_ESC\_DLY(50), /\* delay 50 ms \*/

U8G\_ESC\_ADR(0), 0x000, 0x020, /\* Horizontal GRAM Address Set \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x021, /\* Vertical GRAM Address Set \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

/\* gamma control \*/

U8G\_ESC\_ADR(0), 0x000, 0x030,

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x031,

U8G\_ESC\_ADR(1), 0x004, 0x004,

U8G\_ESC\_ADR(0), 0x000, 0x032,

U8G\_ESC\_ADR(1), 0x000, 0x003,

U8G\_ESC\_ADR(0), 0x000, 0x035,

U8G\_ESC\_ADR(1), 0x004, 0x005,

U8G\_ESC\_ADR(0), 0x000, 0x036,

U8G\_ESC\_ADR(1), 0x008, 0x008,

U8G\_ESC\_ADR(0), 0x000, 0x037,

U8G\_ESC\_ADR(1), 0x004, 0x007,

U8G\_ESC\_ADR(0), 0x000, 0x038,

U8G\_ESC\_ADR(1), 0x003, 0x003,

U8G\_ESC\_ADR(0), 0x000, 0x039,

U8G\_ESC\_ADR(1), 0x007, 0x007,

U8G\_ESC\_ADR(0), 0x000, 0x03c,

U8G\_ESC\_ADR(1), 0x005, 0x004,

U8G\_ESC\_ADR(0), 0x000, 0x03d,

U8G\_ESC\_ADR(1), 0x008, 0x008,

U8G\_ESC\_ADR(0), 0x000, 0x050, /\* Horizontal GRAM Start Address \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x051, /\* Horizontal GRAM End Address: 239 \*/

U8G\_ESC\_ADR(1), 0x000, 0x0EF,

U8G\_ESC\_ADR(0), 0x000, 0x052, /\* Vertical GRAM Start Address \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x053, /\* Vertical GRAM End Address: 319 \*/

U8G\_ESC\_ADR(1), 0x001, 0x03F,

U8G\_ESC\_ADR(0), 0x000, 0x060, /\* Driver Output Control 2 \*/

U8G\_ESC\_ADR(1), 0x0a7, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x061, /\* Base Image Display Control: NDL,VLE, REV \*/

U8G\_ESC\_ADR(1), 0x000, 0x001,

U8G\_ESC\_ADR(0), 0x000, 0x06a, /\* Vertical Scroll Control \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x080, /\* Partial Image 1 Display Position \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x081, /\* Partial Image 1 RAM Start Address \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x082, /\* Partial Image 1 RAM End Address \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x083, /\* Partial Image 2 Display Position \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x084, /\* Partial Image 2 RAM Start Address \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x085, /\* Partial Image 2 RAM End Address \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x090, /\* Panel Interface Control 1 \*/

U8G\_ESC\_ADR(1), 0x000, 0x010,

U8G\_ESC\_ADR(0), 0x000, 0x092, /\* Panel Interface Control 2 \*/

U8G\_ESC\_ADR(1), 0x000, 0x000, /\* 0x006, 0x000 \*/

U8G\_ESC\_ADR(0), 0x000, 0x007, /\* Display Control 1: Operate, display ON \*/

U8G\_ESC\_ADR(1), 0x001, 0x033,

U8G\_ESC\_DLY(10), /\* delay 10 ms \*/

/\* write test pattern \*/

U8G\_ESC\_ADR(0), 0x000, 0x020, /\* Horizontal GRAM Address Set \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x021, /\* Vertical GRAM Address Set \*/

U8G\_ESC\_ADR(1), 0x000, 0x010,

U8G\_ESC\_ADR(0), 0x000, 0x022, /\* Write Data to GRAM \*/

U8G\_ESC\_ADR(1), 0x0fe, 0x0fe,

0x000, 0x000,

0x0fe, 0x0fe,

0x000, 0x000,

0x0fe, 0x0fe,

0x000, 0x000,

0x0fe, 0x0fe,

0x000, 0x000,

0x0fe, 0x0fe,

0x000, 0x000,

0x0fe, 0x0fe,

0x000, 0x000,

0x0fe, 0x0fe,

0x000, 0x000,

0x0fe, 0x0fe,

0x000, 0x000,

0x0fe, 0x0fe,

U8G\_ESC\_CS(0), /\* disable chip \*/

U8G\_ESC\_END /\* end of sequence \*/

};

static const uint8\_t u8g\_dev\_ili9325d\_320x240\_page\_seq[] PROGMEM = {

U8G\_ESC\_CS(1), /\* enable chip \*/

U8G\_ESC\_ADR(0), 0x000, 0x020, /\* Horizontal GRAM Address Set \*/

U8G\_ESC\_ADR(1), 0x000, 0x000,

U8G\_ESC\_ADR(0), 0x000, 0x021, /\* Vertical GRAM Address Set \*/

U8G\_ESC\_ADR(1),

U8G\_ESC\_END /\* end of sequence \*/

};

/\* convert the internal RGB 332 to 65K high byte \*/

static uint8\_t u8g\_dev\_ili9325d\_get\_65K\_high\_byte(uint8\_t color)

{

uint8\_t h;

h = color;

h &= 0x0e0;

h |= h>>3;

h &= 0x0f8;

color>>=2;

color &= 7;

h |= color;

return h;

}

/\* convert the internal RGB 332 to 65K high byte \*/

static uint8\_t u8g\_dev\_ili9325d\_get\_65K\_low\_byte(uint8\_t color)

{

uint8\_t l;

l = color;

l <<= 3;

color &= 3;

color <<= 1;

l |= color;

return l;

}

uint8\_t u8g\_dev\_ili9325d\_320x240\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg)

{

switch(msg)

{

case U8G\_DEV\_MSG\_INIT:

u8g\_InitCom(u8g, dev, U8G\_SPI\_CLK\_CYCLE\_50NS);

//for(;;)

u8g\_WriteEscSeqP(u8g, dev, u8g\_dev\_ili9325d\_320x240\_init\_seq);

break;

case U8G\_DEV\_MSG\_STOP:

break;

case U8G\_DEV\_MSG\_PAGE\_NEXT:

{

uint8\_t i;

uint16\_t y, j;

uint8\_t \*ptr;

u8g\_pb\_t \*pb = (u8g\_pb\_t \*)(dev->dev\_mem);

y = pb->p.page\_y0;

ptr = pb->buf;

for( i = 0; i < pb->p.page\_height; i ++ )

{

u8g\_WriteEscSeqP(u8g, dev, u8g\_dev\_ili9325d\_320x240\_page\_seq);

u8g\_WriteByte(u8g, dev, y >> 8 ); /\* display ram (cursor) address high byte \*/

u8g\_WriteByte(u8g, dev, y & 255 ); /\* display ram (cursor) address low byte \*/

u8g\_SetAddress(u8g, dev, 0); /\* cmd mode \*/

u8g\_WriteByte(u8g, dev, 0 );

u8g\_WriteByte(u8g, dev, 0x022 ); /\* start gram data \*/

u8g\_SetAddress(u8g, dev, 1); /\* data mode \*/

for( j = 0; j < pb->width; j++ )

{

u8g\_WriteByte(u8g, dev, u8g\_dev\_ili9325d\_get\_65K\_high\_byte(\*ptr) );

u8g\_WriteByte(u8g, dev, u8g\_dev\_ili9325d\_get\_65K\_low\_byte(\*ptr) );

ptr++;

}

y++;

}

u8g\_SetChipSelect(u8g, dev, 0);

}

break;

}

return u8g\_dev\_pb8h8\_base\_fn(u8g, dev, msg, arg);

}

uint8\_t u8g\_ili9325d\_320x240\_8h8\_buf[WIDTH\*PAGE\_HEIGHT] U8G\_NOCOMMON ;

u8g\_pb\_t u8g\_ili9325d\_320x240\_8h8\_pb U8G\_NOCOMMON = { {PAGE\_HEIGHT, HEIGHT, 0, 0, 0}, WIDTH, u8g\_ili9325d\_320x240\_8h8\_buf};

u8g\_dev\_t u8g\_dev\_ili9325d\_320x240\_8bit U8G\_NOCOMMON = { u8g\_dev\_ili9325d\_320x240\_fn, &u8g\_ili9325d\_320x240\_8h8\_pb, u8g\_com\_arduino\_port\_d\_wr\_fn };

//u8g\_dev\_t u8g\_dev\_ili9325d\_320x240\_8bit = { u8g\_dev\_ili9325d\_320x240\_fn, &u8g\_ili9325d\_320x240\_8h8\_pb, u8g\_com\_arduino\_parallel\_fn };

//U8G\_PB\_DEV(u8g\_dev\_ili9325d\_320x240\_8bit, WIDTH, HEIGHT, PAGE\_HEIGHT, u8g\_dev\_ili9325d\_320x240\_fn, U8G\_COM\_PARALLEL);