/\*

u8g\_dev\_pcd8544\_84x48.c

Display: Nokia 84x48

Status: Tested with PCF8812 Display

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.

\*/

#include "u8g.h"

#define WIDTH 84

#define HEIGHT 48

#define PAGE\_HEIGHT 8

static const uint8\_t u8g\_dev\_pcd8544\_init\_seq[] PROGMEM = {

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

U8G\_ESC\_ADR(0), /\* instruction mode \*/

U8G\_ESC\_RST(1), /\* do reset low pulse with (1\*16)+2 milliseconds \*/

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

0x021, /\* activate chip (PD=0), horizontal increment (V=0), enter extended command set (H=1) \*/

0x006, /\* temp. control: b10 = 2 \*/

0x013, /\* bias system 1:48 \*/

0x0c0, /\* medium Vop \*/

0x020, /\* activate chip (PD=0), horizontal increment (V=0), enter normal command set (H=0) \*/

0x00c, /\* display on, normal operation \*/

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

0x020, /\* activate chip (PD=0), horizontal increment (V=0), enter normal command set (H=0) \*/

0x00d, /\* display on, invert \*/

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

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

0x020, /\* activate chip (PD=0), horizontal increment (V=0), enter normal command set (H=0) \*/

0x00c, /\* display on, normal \*/

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

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

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

};

static const uint8\_t u8g\_dev\_pcd8544\_sleep\_on[] PROGMEM = {

U8G\_ESC\_ADR(0), /\* instruction mode \*/

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

0x020, /\* activate chip (PD=0), horizontal increment (V=0), enter normal command set (H=0) \*/

0x00c, /\* display on, normal \*/

U8G\_ESC\_CS(0), /\* disable chip, bugfix 12 nov 2014 \*/

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

};

static const uint8\_t u8g\_dev\_pcd8544\_sleep\_off[] PROGMEM = {

U8G\_ESC\_ADR(0), /\* instruction mode \*/

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

0x020, /\* activate chip (PD=0), horizontal increment (V=0), enter normal command set (H=0) \*/

0x008, /\* display blank \*/

0x024, /\* power down (PD=1), horizontal increment (V=0), enter normal command set (H=0) \*/

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

U8G\_ESC\_CS(0), /\* disable chip, bugfix 12 nov 2014 \*/

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

};

uint8\_t u8g\_dev\_pcd8544\_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\_400NS);

u8g\_WriteEscSeqP(u8g, dev, u8g\_dev\_pcd8544\_init\_seq);

break;

case U8G\_DEV\_MSG\_STOP:

break;

case U8G\_DEV\_MSG\_PAGE\_NEXT:

{

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

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

u8g\_SetChipSelect(u8g, dev, 1);

u8g\_WriteByte(u8g, dev, 0x020 ); /\* activate chip (PD=0), horizontal increment (V=0), enter normal command set (H=0) \*/

u8g\_WriteByte(u8g, dev, 0x080 ); /\* set X address \*/

u8g\_WriteByte(u8g, dev, 0x040 | pb->p.page); /\* set Y address \*/

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

if ( u8g\_pb\_WriteBuffer(pb, u8g, dev) == 0 )

return 0;

u8g\_SetChipSelect(u8g, dev, 0);

}

break;

case U8G\_DEV\_MSG\_CONTRAST:

/\* the contrast adjustment does not work, needs to be analysed \*/

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

u8g\_SetChipSelect(u8g, dev, 1);

u8g\_WriteByte(u8g, dev, 0x021); /\* command mode, extended function set \*/

u8g\_WriteByte(u8g, dev, 0x080 | ( (\*(uint8\_t \*)arg) >> 1 ) );

u8g\_SetChipSelect(u8g, dev, 0);

return 1;

case U8G\_DEV\_MSG\_SLEEP\_ON:

u8g\_WriteEscSeqP(u8g, dev, u8g\_dev\_pcd8544\_sleep\_on);

return 1;

case U8G\_DEV\_MSG\_SLEEP\_OFF:

u8g\_WriteEscSeqP(u8g, dev, u8g\_dev\_pcd8544\_sleep\_off);

return 1;

}

return u8g\_dev\_pb8v1\_base\_fn(u8g, dev, msg, arg);

}

U8G\_PB\_DEV(u8g\_dev\_pcd8544\_84x48\_sw\_spi , WIDTH, HEIGHT, PAGE\_HEIGHT, u8g\_dev\_pcd8544\_fn, U8G\_COM\_SW\_SPI);

U8G\_PB\_DEV(u8g\_dev\_pcd8544\_84x48\_hw\_spi , WIDTH, HEIGHT, PAGE\_HEIGHT, u8g\_dev\_pcd8544\_fn, U8G\_COM\_HW\_SPI);