/\*

u8g\_arduino\_ATtiny85\_std\_hw\_spi.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.

\*/

// Uses code from tinySPI Written by Nick Gammon

// March 2013

// ATMEL ATTINY45 / ARDUINO pin mappings

//

// +-\/-+

// RESET Ain0 (D 5) PB5 1| |8 Vcc

// CLK1 Ain3 (D 3) PB3 2| |7 PB2 (D 2) Ain1 SCK / USCK / SCL

// CLK0 Ain2 (D 4) PB4 3| |6 PB1 (D 1) pwm1 MISO / DO

// GND 4| |5 PB0 (D 0) pwm0 MOSI / DI / SDA

// +----+

#include "u8g.h"

#if defined(ARDUINO) && defined(\_\_AVR\_ATtiny85\_\_)

#if ARDUINO < 100

#include <WProgram.h>

#else

#include <Arduino.h>

#endif

const byte DI = 0; // D0, pin 5 Data In

const byte DO = 1; // D1, pin 6 Data Out (this is \*not\* MOSI)

const byte USCK = 2; // D2, pin 7 Universal Serial Interface clock

uint8\_t u8g\_arduino\_ATtiny85\_spi\_out(uint8\_t val)

{

USIDR = val; // byte to output

USISR = \_BV (USIOIF); // clear Counter Overflow Interrupt Flag, set count to zero

do

{

USICR = \_BV (USIWM0) // 3-wire mode

| \_BV (USICS1) | \_BV (USICLK) // Software clock strobe

| \_BV (USITC); // Toggle Clock Port Pin

}

while ((USISR & \_BV (USIOIF)) == 0); // until Counter Overflow Interrupt Flag set

return USIDR; // return read data

}

uint8\_t u8g\_com\_arduino\_ATtiny85\_std\_hw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr)

{

switch(msg)

{

case U8G\_COM\_MSG\_INIT:

u8g\_com\_arduino\_digital\_write(u8g, U8G\_PI\_CS, HIGH); // ensure SS stays high until needed

pinMode (USCK, OUTPUT);

pinMode (DO, OUTPUT);

pinMode (u8g->pin\_list[U8G\_PI\_CS], OUTPUT);

pinMode (u8g->pin\_list[U8G\_PI\_A0], OUTPUT);

USICR = \_BV (USIWM0); // 3-wire mode

u8g\_MicroDelay();

break;

case U8G\_COM\_MSG\_STOP:

break;

case U8G\_COM\_MSG\_RESET:

if ( u8g->pin\_list[U8G\_PI\_RESET] != U8G\_PIN\_NONE )

u8g\_com\_arduino\_digital\_write(u8g, U8G\_PI\_RESET, arg\_val);

break;

case U8G\_COM\_MSG\_CHIP\_SELECT:

if ( arg\_val == 0 )

{

/\* disable \*/

u8g\_MicroDelay();

u8g\_com\_arduino\_digital\_write(u8g, U8G\_PI\_CS, HIGH);

u8g\_MicroDelay();

}

else

{

/\* enable \*/

u8g\_com\_arduino\_digital\_write(u8g, U8G\_PI\_CS, LOW);

u8g\_MicroDelay();

}

break;

case U8G\_COM\_MSG\_WRITE\_BYTE:

u8g\_arduino\_ATtiny85\_spi\_out(arg\_val);

u8g\_MicroDelay();

break;

case U8G\_COM\_MSG\_WRITE\_SEQ:

{

register uint8\_t \*ptr = arg\_ptr;

while( arg\_val > 0 )

{

u8g\_arduino\_ATtiny85\_spi\_out(\*ptr++);

arg\_val--;

}

}

break;

case U8G\_COM\_MSG\_WRITE\_SEQ\_P:

{

register uint8\_t \*ptr = arg\_ptr;

while( arg\_val > 0 )

{

u8g\_arduino\_ATtiny85\_spi\_out(u8g\_pgm\_read(ptr));

ptr++;

arg\_val--;

}

}

break;

case U8G\_COM\_MSG\_ADDRESS: /\* define cmd (arg\_val = 0) or data mode (arg\_val = 1) \*/

u8g\_com\_arduino\_digital\_write(u8g, U8G\_PI\_A0, arg\_val);

u8g\_MicroDelay();

break;

}

return 1;

}

#else /\* ARDUINO \*/

uint8\_t u8g\_com\_arduino\_ATtiny85\_std\_hw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr)

{

return 1;

}

#endif /\* ARDUINO \*/