/\*

u8g.h

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.

\*/

#ifndef \_U8G\_H

#define \_U8G\_H

/\* uncomment the following line to support displays larger than 240x240 \*/

//#define U8G\_16BIT 1

/\* comment the following line to generate more compact but interrupt unsafe code \*/

#define U8G\_INTERRUPT\_SAFE 1

#include <stddef.h>

#ifdef \_\_18CXX

typedef unsigned char uint8\_t;

typedef signed char int8\_t;

typedef unsigned short uint16\_t;

typedef signed short int16\_t;

#else

#include <stdint.h>

#endif

#if defined(\_\_AVR\_\_)

#include <avr/pgmspace.h>

#endif

/\*

use the com interface directly on any systems which are not AVR or ARDUINO

\*/

#if defined(\_\_AVR\_\_) || defined(ARDUINO)

#define U8G\_WITH\_PINLIST

#endif

#define U8G\_WITH\_PINLIST

#ifdef \_\_cplusplus

extern "C" {

#endif

/\*===============================================================\*/

#ifdef \_\_GNUC\_\_

# define U8G\_NOINLINE \_\_attribute\_\_((noinline))

# define U8G\_PURE \_\_attribute\_\_ ((pure))

# define U8G\_NOCOMMON \_\_attribute\_\_ ((nocommon))

# define U8G\_SECTION(name) \_\_attribute\_\_ ((section (name)))

# if defined(\_\_MSPGCC\_\_)

/\* mspgcc does not have .progmem sections. Use -fdata-sections. \*/

# define U8G\_FONT\_SECTION(name)

# endif

# if defined(\_\_AVR\_\_)

# define U8G\_FONT\_SECTION(name) U8G\_SECTION(".progmem." name)

# endif

#else

# define U8G\_NOINLINE

# define U8G\_PURE

# define U8G\_NOCOMMON

# define U8G\_SECTION(name)

# define U8G\_FONT\_SECTION(name)

#endif

#ifndef U8G\_FONT\_SECTION

# define U8G\_FONT\_SECTION(name)

#endif

/\*===============================================================\*/

/\* flash memory access \*/

#if defined(\_\_AVR\_\_)

/\* U8G\_PROGMEM is used by the XBM example \*/

#define U8G\_PROGMEM U8G\_SECTION(".progmem.data")

typedef uint8\_t PROGMEM u8g\_pgm\_uint8\_t;

typedef uint8\_t u8g\_fntpgm\_uint8\_t;

#define u8g\_pgm\_read(adr) pgm\_read\_byte\_near(adr)

#define U8G\_PSTR(s) ((u8g\_pgm\_uint8\_t \*)PSTR(s))

#else

#define U8G\_PROGMEM

#define PROGMEM

typedef uint8\_t u8g\_pgm\_uint8\_t;

typedef uint8\_t u8g\_fntpgm\_uint8\_t;

#define u8g\_pgm\_read(adr) (\*(const u8g\_pgm\_uint8\_t \*)(adr))

#define U8G\_PSTR(s) ((u8g\_pgm\_uint8\_t \*)(s))

#endif

/\*===============================================================\*/

/\* interrupt safe code \*/

#if defined(U8G\_INTERRUPT\_SAFE)

# if defined(\_\_AVR\_\_)

extern uint8\_t global\_SREG\_backup; /\* u8g\_state.c \*/

# define U8G\_ATOMIC\_START() do { global\_SREG\_backup = SREG; cli(); } while(0)

# define U8G\_ATOMIC\_END() SREG = global\_SREG\_backup

# define U8G\_ATOMIC\_OR(ptr, val) do { uint8\_t tmpSREG = SREG; cli(); (\*(ptr) |= (val)); SREG = tmpSREG; } while(0)

# define U8G\_ATOMIC\_AND(ptr, val) do { uint8\_t tmpSREG = SREG; cli(); (\*(ptr) &= (val)); SREG = tmpSREG; } while(0)

# else

# define U8G\_ATOMIC\_OR(ptr, val) (\*(ptr) |= (val))

# define U8G\_ATOMIC\_AND(ptr, val) (\*(ptr) &= (val))

# define U8G\_ATOMIC\_START()

# define U8G\_ATOMIC\_END()

# endif /\* \_\_AVR\_\_ \*/

#else

# define U8G\_ATOMIC\_OR(ptr, val) (\*(ptr) |= (val))

# define U8G\_ATOMIC\_AND(ptr, val) (\*(ptr) &= (val))

# define U8G\_ATOMIC\_START()

# define U8G\_ATOMIC\_END()

#endif /\* U8G\_INTERRUPT\_SAFE \*/

/\*===============================================================\*/

/\* forward \*/

typedef struct \_u8g\_t u8g\_t;

typedef struct \_u8g\_dev\_t u8g\_dev\_t;

typedef struct \_u8g\_dev\_arg\_pixel\_t u8g\_dev\_arg\_pixel\_t;

typedef struct \_u8g\_dev\_arg\_bbx\_t u8g\_dev\_arg\_bbx\_t;

typedef struct \_u8g\_box\_t u8g\_box\_t;

typedef struct \_u8g\_dev\_arg\_irgb\_t u8g\_dev\_arg\_irgb\_t;

/\*===============================================================\*/

/\* generic \*/

#if defined(U8G\_16BIT)

typedef uint16\_t u8g\_uint\_t;

typedef int16\_t u8g\_int\_t;

#else

typedef uint8\_t u8g\_uint\_t;

typedef int8\_t u8g\_int\_t;

#endif

#ifdef OBSOLETE

struct \_u8g\_box\_t

{

u8g\_uint\_t x0, y0, x1, y1;

};

typedef struct \_u8g\_box\_t u8g\_box\_t;

#endif /\* OBSOLETE \*/

/\*===============================================================\*/

/\* device structure \*/

#ifdef \_\_XC8

/\* device prototype \*/

typedef uint8\_t (\*u8g\_dev\_fnptr)(void \*u8g, void \*dev, uint8\_t msg, void \*arg);

/\* com prototype \*/

typedef uint8\_t (\*u8g\_com\_fnptr)(void \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr);

#else

/\* device prototype \*/

typedef uint8\_t (\*u8g\_dev\_fnptr)(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* com prototype \*/

typedef uint8\_t (\*u8g\_com\_fnptr)(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr);

#endif

struct \_u8g\_dev\_t

{

u8g\_dev\_fnptr dev\_fn; /\* device procedure \*/

void \*dev\_mem; /\* device memory \*/

u8g\_com\_fnptr com\_fn; /\* communication procedure \*/

};

/\*===============================================================\*/

/\* device list \*/

/\* Size: 128x64 SDL, u8g\_dev\_sdl.c \*/

extern u8g\_dev\_t u8g\_dev\_sdl\_1bit;

extern u8g\_dev\_t u8g\_dev\_sdl\_1bit\_h;

extern u8g\_dev\_t u8g\_dev\_sdl\_2bit;

extern u8g\_dev\_t u8g\_dev\_sdl\_2bit\_double\_mem;

extern u8g\_dev\_t u8g\_dev\_sdl\_8bit;

extern u8g\_dev\_t u8g\_dev\_sdl\_hicolor;

extern u8g\_dev\_t u8g\_dev\_sdl\_fullcolor;

int u8g\_sdl\_get\_key(void);

/\* Size: 70x30 monochrom, stdout \*/

extern u8g\_dev\_t u8g\_dev\_stdout;

/\* Size: monochrom, writes "u8g.pbm" \*/

extern u8g\_dev\_t u8g\_dev\_pbm;

extern u8g\_dev\_t u8g\_dev\_pbm\_8h1;

extern u8g\_dev\_t u8g\_dev\_pbm\_8h2; /\* grayscale simulation \*/

/\* Size: 128x64 monochrom, no output, used for performance measure \*/

extern u8g\_dev\_t u8g\_dev\_gprof;

/\* Display: EA DOGS102, Size: 102x64 monochrom \*/

extern u8g\_dev\_t u8g\_dev\_uc1701\_dogs102\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1701\_dogs102\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1701\_dogs102\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1701\_dogs102\_2x\_hw\_spi;

/\* Display: Mini12864 (dealextreme), Size: 128x64 monochrom \*/

extern u8g\_dev\_t u8g\_dev\_uc1701\_mini12864\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1701\_mini12864\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1701\_mini12864\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1701\_mini12864\_2x\_hw\_spi;

/\* Display: EA DOGM132, Size: 128x32 monochrom \*/

extern u8g\_dev\_t u8g\_dev\_st7565\_dogm132\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_dogm132\_hw\_spi;

/\* Display: EA DOGM128, Size: 128x64 monochrom \*/

extern u8g\_dev\_t u8g\_dev\_st7565\_dogm128\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_dogm128\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_dogm128\_parallel;

extern u8g\_dev\_t u8g\_dev\_st7565\_dogm128\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_dogm128\_2x\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_dogm128\_2x\_parallel;

/\* EA DOGM 240-6 \*/

extern u8g\_dev\_t u8g\_dev\_uc1611\_dogm240\_i2c;

extern u8g\_dev\_t u8g\_dev\_uc1611\_dogm240\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1611\_dogm240\_sw\_spi;

/\* EA DOGXL 240 \*/

extern u8g\_dev\_t u8g\_dev\_uc1611\_dogxl240\_i2c;

extern u8g\_dev\_t u8g\_dev\_uc1611\_dogxl240\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1611\_dogxl240\_sw\_spi;

/\* Display: Topway LM6059 128x64 (Adafruit) \*/

extern u8g\_dev\_t u8g\_dev\_st7565\_lm6059\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_lm6059\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_lm6059\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_lm6059\_2x\_hw\_spi;

/\* Display: Topway LM6063 128x64 \*/

extern u8g\_dev\_t u8g\_dev\_st7565\_lm6063\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_lm6063\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_lm6063\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_lm6063\_2x\_hw\_spi;

/\* Display: Newhaven NHD-C12864 \*/

extern u8g\_dev\_t u8g\_dev\_st7565\_nhd\_c12864\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_nhd\_c12864\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_nhd\_c12864\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_nhd\_c12864\_2x\_hw\_spi;

/\* Display: Newhaven NHD-C12832 \*/

extern u8g\_dev\_t u8g\_dev\_st7565\_nhd\_c12832\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_nhd\_c12832\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_nhd\_c12832\_parallel;

extern u8g\_dev\_t u8g\_dev\_st7565\_nhd\_c12832\_hw\_usart\_spi;

/\* Display: Displaytech 64128N \*/

extern u8g\_dev\_t u8g\_dev\_st7565\_64128n\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_64128n\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_64128n\_parallel;

extern u8g\_dev\_t u8g\_dev\_st7565\_64128n\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_64128n\_2x\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7565\_64128n\_2x\_parallel;

/\* Display: LCD-AG-C128032R-DIW W/KK E6 PBF \*/

extern u8g\_dev\_t u8g\_dev\_uc1601\_c128032\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1601\_c128032\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1601\_c128032\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1601\_c128032\_2x\_hw\_spi;

/\* East Rising/buy-display.com ERC24064-1 \*/

extern u8g\_dev\_t u8g\_dev\_uc1608\_240x64\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1608\_240x64\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1608\_240x64\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1608\_240x64\_2x\_hw\_spi;

/\* UC1608 240x128 \*/

extern u8g\_dev\_t u8g\_dev\_uc1608\_240x128\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1608\_240x128\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1608\_240x128\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1608\_240x128\_2x\_hw\_spi;

/\* dfrobot 128x64 Graphic LCD (SKU:FIT0021) \*/

extern u8g\_dev\_t u8g\_dev\_st7920\_128x64\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_128x64\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_128x64\_8bit;

extern u8g\_dev\_t u8g\_dev\_st7920\_128x64\_custom;

extern u8g\_dev\_t u8g\_dev\_st7920\_128x64\_4x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_128x64\_4x\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_128x64\_4x\_8bit;

extern u8g\_dev\_t u8g\_dev\_st7920\_128x64\_4x\_custom;

/\* NHD-19232WG \*/

extern u8g\_dev\_t u8g\_dev\_st7920\_192x32\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_192x32\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_192x32\_8bit;

extern u8g\_dev\_t u8g\_dev\_st7920\_192x32\_4x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_192x32\_4x\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_192x32\_4x\_8bit;

/\* CrystalFontz CFAG20232 \*/

extern u8g\_dev\_t u8g\_dev\_st7920\_202x32\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_202x32\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_202x32\_8bit;

extern u8g\_dev\_t u8g\_dev\_st7920\_202x32\_4x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_202x32\_4x\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7920\_202x32\_4x\_8bit;

/\* LC7981 160x80 display \*/

extern u8g\_dev\_t u8g\_dev\_lc7981\_160x80\_8bit;

/\* LC7981 240x64 display \*/

extern u8g\_dev\_t u8g\_dev\_lc7981\_240x64\_8bit;

/\* LC7981 240x128 display \*/

extern u8g\_dev\_t u8g\_dev\_lc7981\_240x128\_8bit;

/\* LC7981 320x64 display \*/

extern u8g\_dev\_t u8g\_dev\_lc7981\_320x64\_8bit;

/\* T6963, all t6963 devices have double page (2x) \*/

extern u8g\_dev\_t u8g\_dev\_t6963\_240x128\_8bit;

extern u8g\_dev\_t u8g\_dev\_t6963\_128x128\_8bit;

extern u8g\_dev\_t u8g\_dev\_t6963\_240x64\_8bit;

extern u8g\_dev\_t u8g\_dev\_t6963\_128x64\_8bit;

/\* Display: EA DOGXL160, Size: 160x104 monochrom & gray level \*/

extern u8g\_dev\_t u8g\_dev\_uc1610\_dogxl160\_bw\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1610\_dogxl160\_bw\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1610\_dogxl160\_gr\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1610\_dogxl160\_gr\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1610\_dogxl160\_2x\_bw\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1610\_dogxl160\_2x\_bw\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1610\_dogxl160\_2x\_gr\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_uc1610\_dogxl160\_2x\_gr\_hw\_spi;

/\* Display: Generic KS0108b, Size: 128x64 monochrom \*/

extern u8g\_dev\_t u8g\_dev\_ks0108\_128x64; /\* official Arduino Library interface \*/

extern u8g\_dev\_t u8g\_dev\_ks0108\_128x64\_fast; /\* faster, but uses private tables from the Arduino Library \*/

/\* Nokia 84x48 Display with PCD8544 \*/

extern u8g\_dev\_t u8g\_dev\_pcd8544\_84x48\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_pcd8544\_84x48\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_tls8204\_84x48\_sw\_spi;

/\* Nokia 96x65 Display with PCF8812 \*/

extern u8g\_dev\_t u8g\_dev\_pcf8812\_96x65\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_pcf8812\_96x65\_hw\_spi;

/\* NHD-2.7-12864UCY3 OLED Display with SSD1325 Controller \*/

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_bw\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_bw\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_bw\_parallel;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_gr\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_gr\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_2x\_bw\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_2x\_bw\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_2x\_bw\_parallel;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_2x\_gr\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1325\_nhd27oled\_2x\_gr\_hw\_spi;

/\* LY120 OLED with SSD1327 Controller (tested with Seeedstudio module) \*/

extern u8g\_dev\_t u8g\_dev\_ssd1327\_96x96\_gr\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1327\_96x96\_gr\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1327\_96x96\_gr\_i2c;

extern u8g\_dev\_t u8g\_dev\_ssd1327\_96x96\_2x\_gr\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1327\_96x96\_2x\_gr\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1327\_96x96\_2x\_gr\_i2c;

/\* NHD-3.12-25664 OLED Display with SSD1322 Controller \*/

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_bw\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_bw\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_bw\_parallel;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_2x\_bw\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_2x\_bw\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_gr\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_gr\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_gr\_parallel;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_2x\_gr\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1322\_nhd31oled\_2x\_gr\_hw\_spi;

/\* OLED 128x64 Display with SSD1306 Controller \*/

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x64\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x64\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x64\_i2c;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_adafruit\_128x64\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_adafruit\_128x64\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_adafruit\_128x64\_i2c;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x64\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x64\_2x\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x64\_2x\_i2c;

/\* OLED 128x64 Display with SH1106 Controller \*/

extern u8g\_dev\_t u8g\_dev\_sh1106\_128x64\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_sh1106\_128x64\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_sh1106\_128x64\_i2c;

extern u8g\_dev\_t u8g\_dev\_sh1106\_128x64\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_sh1106\_128x64\_2x\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_sh1106\_128x64\_2x\_i2c;

/\* OLED 128x64 Display with SSD1309 Controller \*/

extern u8g\_dev\_t u8g\_dev\_ssd1309\_128x64\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1309\_128x64\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1309\_128x64\_i2c;

/\* OLED 128x32 Display with SSD1306 Controller \*/

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x32\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x32\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x32\_i2c;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x32\_2x\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x32\_2x\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1306\_128x32\_2x\_i2c;

/\* OLED 60x32 Display with LD7032 Controller \*/

extern u8g\_dev\_t u8g\_dev\_ld7032\_60x32\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ld7032\_60x32\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ld7032\_60x32\_parallel;

/\* experimental 65K TFT with st7687 controller \*/

extern u8g\_dev\_t u8g\_dev\_st7687\_c144mvgd\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_st7687\_c144mvgd\_8bit;

/\* SBN1661/SED1520 display with 122x32 \*/

extern u8g\_dev\_t u8g\_dev\_sbn1661\_122x32;

/\* flip disc matrix \*/

extern u8g\_dev\_t u8g\_dev\_flipdisc\_2x7;

void u8g\_SetFlipDiscCallback(u8g\_t \*u8g, void (\*cb)(uint8\_t id, uint8\_t page, uint8\_t width, uint8\_t \*row1, uint8\_t \*row2));

/\* ILI9325D based TFT \*/

extern u8g\_dev\_t u8g\_dev\_ili9325d\_320x240\_8bit;

/\* SSD1351 OLED (breakout board from http://www.kickstarter.com/projects/ilsoftltd/colour-oled-breakout-board) \*/

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_332\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_332\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_4x\_332\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_4x\_332\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_idx\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_idx\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_hicolor\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_hicolor\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_4x\_hicolor\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128\_4x\_hicolor\_hw\_spi;

/\* SSD1351 OLED (Freetronics, GPIOs set to high level) \*/

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128gh\_332\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128gh\_332\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128gh\_4x\_332\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128gh\_4x\_332\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128gh\_hicolor\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128gh\_hicolor\_hw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128gh\_4x\_hicolor\_sw\_spi;

extern u8g\_dev\_t u8g\_dev\_ssd1351\_128x128gh\_4x\_hicolor\_hw\_spi;

/\* HT1632 \*/

extern u8g\_dev\_t u8g\_dev\_ht1632\_24x16;

/\* A2 Micro Printer \*/

extern u8g\_dev\_t u8g\_dev\_a2\_micro\_printer\_384x240;

extern u8g\_dev\_t u8g\_dev\_a2\_micro\_printer\_192x120\_ds;

extern u8g\_dev\_t u8g\_dev\_a2\_micro\_printer\_192x360\_ds;

extern u8g\_dev\_t u8g\_dev\_a2\_micro\_printer\_192x720\_ds;

/\* u8g\_virtual\_screen.c \*/

extern u8g\_dev\_t u8g\_dev\_vs;

/\*===============================================================\*/

/\* device messages \*/

struct \_u8g\_dev\_arg\_pixel\_t

{

u8g\_uint\_t x, y; /\* will be modified \*/

uint8\_t pixel; /\* will be modified, pixel sequence or transparency value \*/

uint8\_t dir;

uint8\_t color; /\* color or index value, red value for true color mode \*/

uint8\_t hi\_color; /\* high byte for 64K color mode, low byte is in "color", green value for true color mode \*/

uint8\_t blue; /\* blue value in true color mode \*/

};

/\* typedef struct \_u8g\_dev\_arg\_pixel\_t u8g\_dev\_arg\_pixel\_t; \*/ /\* forward decl \*/

/\* range for r,g,b: 0..255 \*/

#define U8G\_GET\_HICOLOR\_BY\_RGB(r,g,b) (((uint16\_t)((r)&0x0f8))<<8)|(((uint16\_t)((g)&0x0fc))<<3)|(((uint16\_t)((b)>>3)))

struct \_u8g\_dev\_arg\_bbx\_t

{

u8g\_uint\_t x, y, w, h;

};

/\* typedef struct \_u8g\_dev\_arg\_bbx\_t u8g\_dev\_arg\_bbx\_t; \*/ /\* forward decl \*/

struct \_u8g\_box\_t

{

u8g\_uint\_t x0, y0, x1, y1;

};

/\* typedef struct \_u8g\_box\_t u8g\_box\_t; \*/ /\* forward decl \*/

struct \_u8g\_dev\_arg\_irgb\_t

{

u8g\_uint\_t idx, r, g, b; /\* index with rgb value \*/

};

/\* typedef struct \_u8g\_dev\_arg\_irgb\_t u8g\_dev\_arg\_irgb\_t; \*/ /\* forward decl \*/

#define U8G\_DEV\_MSG\_INIT 10

#define U8G\_DEV\_MSG\_STOP 11

/\* arg: pointer to uint8\_t, contranst value between 0 and 255 \*/

#define U8G\_DEV\_MSG\_CONTRAST 15

#define U8G\_DEV\_MSG\_SLEEP\_ON 16

#define U8G\_DEV\_MSG\_SLEEP\_OFF 17

#define U8G\_DEV\_MSG\_PAGE\_FIRST 20

#define U8G\_DEV\_MSG\_PAGE\_NEXT 21

/\* arg: u8g\_dev\_arg\_bbx\_t \* \*/

/\* new algorithm with U8G\_DEV\_MSG\_GET\_PAGE\_BOX makes this msg obsolete \*/

/\* #define U8G\_DEV\_MSG\_IS\_BBX\_INTERSECTION 22 \*/

/\* arg: u8g\_box\_t \*, fill structure with current page properties \*/

#define U8G\_DEV\_MSG\_GET\_PAGE\_BOX 23

/\*

#define U8G\_DEV\_MSG\_PRIMITIVE\_START 30

#define U8G\_DEV\_MSG\_PRIMITIVE\_END 31

\*/

/\* arg: u8g\_dev\_arg\_pixel\_t \* \*/

#define U8G\_DEV\_MSG\_SET\_TPIXEL 44

#define U8G\_DEV\_MSG\_SET\_4TPIXEL 45

#define U8G\_DEV\_MSG\_SET\_PIXEL 50

#define U8G\_DEV\_MSG\_SET\_8PIXEL 59

#define U8G\_DEV\_MSG\_SET\_COLOR\_ENTRY 60

#define U8G\_DEV\_MSG\_SET\_XY\_CB 61

#define U8G\_DEV\_MSG\_GET\_WIDTH 70

#define U8G\_DEV\_MSG\_GET\_HEIGHT 71

#define U8G\_DEV\_MSG\_GET\_MODE 72

/\*===============================================================\*/

/\* device modes \*/

#define U8G\_MODE(is\_index\_mode, is\_color, bits\_per\_pixel) (((is\_index\_mode)<<6) | ((is\_color)<<5)|(bits\_per\_pixel))

#define U8G\_MODE\_UNKNOWN 0

#define U8G\_MODE\_BW U8G\_MODE(0, 0, 1)

#define U8G\_MODE\_GRAY2BIT U8G\_MODE(0, 0, 2)

#define U8G\_MODE\_R3G3B2 U8G\_MODE(0, 1, 8)

#define U8G\_MODE\_INDEX U8G\_MODE(1, 1, 8)

/\* hicolor is R5G6B5 \*/

#define U8G\_MODE\_HICOLOR U8G\_MODE(0, 1, 16)

/\* truecolor \*/

#define U8G\_MODE\_TRUECOLOR U8G\_MODE(0, 1, 24)

#define U8G\_MODE\_GET\_BITS\_PER\_PIXEL(mode) ((mode)&31)

#define U8G\_MODE\_IS\_COLOR(mode) (((mode)&32)==0?0:1)

#define U8G\_MODE\_IS\_INDEX\_MODE(mode) (((mode)&64)==0?0:1)

/\*===============================================================\*/

/\* com options \*/

/\* uncomment the following line for Atmega HW SPI double speed, issue 89 \*/

/\* #define U8G\_HW\_SPI\_2X 1 \*/

/\* com messages \*/

#define U8G\_COM\_MSG\_STOP 0

#define U8G\_COM\_MSG\_INIT 1

#define U8G\_COM\_MSG\_ADDRESS 2

/\* CHIP\_SELECT argument: number of the chip which needs to be activated, so this is more like high active \*/

#define U8G\_COM\_MSG\_CHIP\_SELECT 3

#define U8G\_COM\_MSG\_RESET 4

#define U8G\_COM\_MSG\_WRITE\_BYTE 5

#define U8G\_COM\_MSG\_WRITE\_SEQ 6

#define U8G\_COM\_MSG\_WRITE\_SEQ\_P 7

/\* com driver \*/

uint8\_t u8g\_com\_null\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_null.c \*/

uint8\_t u8g\_com\_arduino\_std\_sw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_std\_sw\_spi.c \*/

uint8\_t u8g\_com\_arduino\_hw\_usart\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_atmega\_hw\_usart\_spi.c \*/

uint8\_t u8g\_com\_arduino\_sw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_sw\_spi.c \*/

uint8\_t u8g\_com\_arduino\_hw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_hw\_spi.c \*/

uint8\_t u8g\_com\_arduino\_ATtiny85\_std\_hw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_arduino\_ATTiny85\_std\_hw\_spi.c \*/

uint8\_t u8g\_com\_arduino\_st7920\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_st7920\_spi.c \*/

uint8\_t u8g\_com\_arduino\_st7920\_custom\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_st7920\_custom.c \*/

uint8\_t u8g\_com\_arduino\_st7920\_hw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_st7920\_hw\_spi.c \*/

uint8\_t u8g\_com\_arduino\_parallel\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_parallel.c \*/

uint8\_t u8g\_com\_arduino\_fast\_parallel\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_fast\_parallel.c \*/

uint8\_t u8g\_com\_arduino\_port\_d\_wr\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_port\_d\_wr.c \*/

uint8\_t u8g\_com\_arduino\_no\_en\_parallel\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_no\_en\_parallel.c \*/

uint8\_t u8g\_com\_arduino\_ssd\_i2c\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_ssd\_i2c.c \*/

uint8\_t u8g\_com\_arduino\_uc\_i2c\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr);

uint8\_t u8g\_com\_arduino\_t6963\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_arduino\_t6963.c \*/

uint8\_t u8g\_com\_atmega\_hw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_atmega\_hw\_spi.c \*/

uint8\_t u8g\_com\_atmega\_sw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_atmega\_sw\_spi.c \*/

uint8\_t u8g\_com\_atmega\_st7920\_sw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_atmega\_st7920\_spi.c \*/

uint8\_t u8g\_com\_atmega\_st7920\_hw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr);

uint8\_t u8g\_com\_atmega\_parallel\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_atmega\_parallel.c \*/

uint8\_t u8g\_com\_raspberrypi\_hw\_spi\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_rasperrypi\_hw\_spi.c \*/

uint8\_t u8g\_com\_raspberrypi\_ssd\_i2c\_fn(u8g\_t \*u8g, uint8\_t msg, uint8\_t arg\_val, void \*arg\_ptr); /\* u8g\_com\_raspberrypi\_ssd\_i2c.c \*/

/\*

Translation of system specific com drives to generic com names

At the moment, the following generic com drives are available

U8G\_COM\_HW\_SPI

U8G\_COM\_SW\_SPI

U8G\_COM\_PARALLEL

U8G\_COM\_T6963

U8G\_COM\_FAST\_PARALLEL

U8G\_COM\_SSD\_I2C

U8G\_COM\_UC\_I2C

defined(\_\_18CXX) || defined(\_\_PIC32MX)

\*/

/\* ==== HW SPI, Raspberry PI ====\*/

#if defined(U8G\_RASPBERRY\_PI)

#define U8G\_COM\_HW\_SPI u8g\_com\_raspberrypi\_hw\_spi\_fn

#define U8G\_COM\_SW\_SPI u8g\_com\_null\_fn

/\* I'm sure there must be some mad reason for needing this \*/

#define U8G\_COM\_ST7920\_SW\_SPI u8g\_com\_null\_fn

#define U8G\_COM\_ST7920\_HW\_SPI u8g\_com\_null\_fn

#endif

/\* ==== HW SPI, Arduino ====\*/

#if defined(ARDUINO)

#if defined(\_\_AVR\_\_)

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

#define U8G\_COM\_HW\_SPI u8g\_com\_arduino\_ATtiny85\_std\_hw\_spi\_fn

#define U8G\_COM\_ST7920\_HW\_SPI u8g\_com\_null\_fn

#else

#define U8G\_COM\_HW\_SPI u8g\_com\_arduino\_hw\_spi\_fn

#if defined(\_\_AVR\_ATmega32U4\_\_)

#define U8G\_COM\_HW\_USART\_SPI u8g\_com\_arduino\_hw\_usart\_spi\_fn

#endif /\* \_\_AVR\_ATmega32U4\_\_ \*/

#define U8G\_COM\_ST7920\_HW\_SPI u8g\_com\_arduino\_st7920\_hw\_spi\_fn

#endif /\* \_\_AVR\_ATtiny85\_\_ \*/

#elif defined(\_\_18CXX) || defined(\_\_PIC32MX)

#define U8G\_COM\_HW\_SPI u8g\_com\_null\_fn

#define U8G\_COM\_ST7920\_HW\_SPI u8g\_com\_null\_fn

#elif defined(\_\_SAM3X8E\_\_) /\* Arduino Due \*/

#define U8G\_COM\_HW\_SPI u8g\_com\_arduino\_hw\_spi\_fn

#define U8G\_COM\_ST7920\_HW\_SPI u8g\_com\_null\_fn

#endif

#endif

/\* ==== HW SPI, not Arduino ====\*/

#ifndef U8G\_COM\_HW\_SPI

#if defined(\_\_AVR\_\_)

#define U8G\_COM\_HW\_SPI u8g\_com\_atmega\_hw\_spi\_fn

#define U8G\_COM\_ST7920\_HW\_SPI u8g\_com\_atmega\_st7920\_hw\_spi\_fn

#endif

#endif

#ifndef U8G\_COM\_HW\_SPI

#define U8G\_COM\_HW\_SPI u8g\_com\_null\_fn

#define U8G\_COM\_ST7920\_HW\_SPI u8g\_com\_null\_fn

#endif

#ifndef U8G\_COM\_HW\_USART\_SPI

#define U8G\_COM\_HW\_USART\_SPI u8g\_com\_null\_fn

#endif

/\* ==== SW SPI, Arduino ====\*/

#if defined(ARDUINO)

#if defined(\_\_AVR\_\_)

#define U8G\_COM\_SW\_SPI u8g\_com\_arduino\_sw\_spi\_fn

#define U8G\_COM\_ST7920\_SW\_SPI u8g\_com\_arduino\_st7920\_spi\_fn

#elif defined(\_\_18CXX) || defined(\_\_PIC32MX)

#define U8G\_COM\_SW\_SPI u8g\_com\_arduino\_sw\_spi\_fn

#define U8G\_COM\_ST7920\_SW\_SPI u8g\_com\_arduino\_st7920\_spi\_fn

#elif defined(\_\_SAM3X8E\_\_) /\* Arduino Due \*/

//#define U8G\_COM\_SW\_SPI u8g\_com\_arduino\_std\_sw\_spi\_fn

#define U8G\_COM\_SW\_SPI u8g\_com\_arduino\_sw\_spi\_fn

#define U8G\_COM\_ST7920\_SW\_SPI u8g\_com\_arduino\_st7920\_spi\_fn

#elif defined(\_\_arm\_\_) /\* Teensy \*/

#define U8G\_COM\_SW\_SPI u8g\_com\_arduino\_std\_sw\_spi\_fn

#define U8G\_COM\_ST7920\_SW\_SPI u8g\_com\_arduino\_st7920\_spi\_fn

#endif

#endif

#ifndef U8G\_COM\_SW\_SPI

/\* ==== SW SPI, not Arduino ====\*/

#if defined(\_\_AVR\_\_)

#define U8G\_COM\_SW\_SPI u8g\_com\_atmega\_sw\_spi\_fn

#define U8G\_COM\_ST7920\_SW\_SPI u8g\_com\_atmega\_st7920\_sw\_spi\_fn

#endif

#endif

#ifndef U8G\_COM\_SW\_SPI

#define U8G\_COM\_SW\_SPI u8g\_com\_null\_fn

#define U8G\_COM\_ST7920\_SW\_SPI u8g\_com\_null\_fn

#endif

/\* ==== Parallel interface, Arduino ====\*/

#if defined(ARDUINO)

#if defined(\_\_AVR\_\_)

#define U8G\_COM\_PARALLEL u8g\_com\_arduino\_parallel\_fn

#define U8G\_COM\_FAST\_PARALLEL u8g\_com\_arduino\_fast\_parallel\_fn

#define U8G\_COM\_T6963 u8g\_com\_arduino\_t6963\_fn

#else /\* Arduino Due, Chipkit PIC32 \*/

#define U8G\_COM\_PARALLEL u8g\_com\_arduino\_parallel\_fn

#define U8G\_COM\_FAST\_PARALLEL u8g\_com\_arduino\_parallel\_fn

#define U8G\_COM\_T6963 u8g\_com\_null\_fn

#endif

#endif

#ifndef U8G\_COM\_PARALLEL

#if defined(\_\_AVR\_\_)

#define U8G\_COM\_PARALLEL u8g\_com\_atmega\_parallel\_fn

#define U8G\_COM\_FAST\_PARALLEL u8g\_com\_atmega\_parallel\_fn

#define U8G\_COM\_T6963 u8g\_com\_null\_fn

#endif

#endif

#ifndef U8G\_COM\_PARALLEL

#define U8G\_COM\_PARALLEL u8g\_com\_null\_fn

#define U8G\_COM\_FAST\_PARALLEL u8g\_com\_null\_fn

#define U8G\_COM\_T6963 u8g\_com\_null\_fn

#endif

#if defined(ARDUINO)

#if defined(\_\_AVR\_\_)

#define U8G\_COM\_SSD\_I2C u8g\_com\_arduino\_ssd\_i2c\_fn

#define U8G\_COM\_UC\_I2C u8g\_com\_arduino\_uc\_i2c\_fn

#endif

#endif

#ifndef U8G\_COM\_SSD\_I2C

#if defined(\_\_AVR\_\_) || defined(\_\_SAM3X8E\_\_)

/\* AVR variant and also DUE can use the arduino version at the moment \*/

#define U8G\_COM\_SSD\_I2C u8g\_com\_arduino\_ssd\_i2c\_fn

#endif

#endif

#ifndef U8G\_COM\_SSD\_I2C

#if defined(U8G\_RASPBERRY\_PI)

#define U8G\_COM\_SSD\_I2C u8g\_com\_raspberrypi\_ssd\_i2c\_fn

#endif

#endif

#ifndef U8G\_COM\_SSD\_I2C

#define U8G\_COM\_SSD\_I2C u8g\_com\_null\_fn

#endif

#ifndef U8G\_COM\_UC\_I2C

#if defined(\_\_AVR\_\_)

/\* AVR variant can use the arduino version at the moment \*/

#define U8G\_COM\_UC\_I2C u8g\_com\_arduino\_uc\_i2c\_fn

#endif

#endif

#ifndef U8G\_COM\_UC\_I2C

#define U8G\_COM\_UC\_I2C u8g\_com\_null\_fn

#endif

/\*===============================================================\*/

/\* com api \*/

#define U8G\_SPI\_CLK\_CYCLE\_50NS 1

#define U8G\_SPI\_CLK\_CYCLE\_300NS 2

#define U8G\_SPI\_CLK\_CYCLE\_400NS 3

#define U8G\_SPI\_CLK\_CYCLE\_NONE 255

uint8\_t u8g\_InitCom(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t clk\_cycle\_time);

void u8g\_StopCom(u8g\_t \*u8g, u8g\_dev\_t \*dev);

void u8g\_EnableCom(u8g\_t \*u8g, u8g\_dev\_t \*dev); /\* obsolete \*/

void u8g\_DisableCom(u8g\_t \*u8g, u8g\_dev\_t \*dev); /\* obsolete \*/

void u8g\_SetChipSelect(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t cs);

void u8g\_SetResetLow(u8g\_t \*u8g, u8g\_dev\_t \*dev);

void u8g\_SetResetHigh(u8g\_t \*u8g, u8g\_dev\_t \*dev);

void u8g\_SetAddress(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t address);

uint8\_t u8g\_WriteByte(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t val);

uint8\_t u8g\_WriteSequence(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t cnt, uint8\_t \*seq);

uint8\_t u8g\_WriteSequenceP(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t cnt, const uint8\_t \*seq);

#define U8G\_ESC\_DLY(x) 255, ((x) & 0x7f)

#define U8G\_ESC\_CS(x) 255, (0xd0 | ((x)&0x0f))

#define U8G\_ESC\_ADR(x) 255, (0xe0 | ((x)&0x0f))

#define U8G\_ESC\_RST(x) 255, (0xc0 | ((x)&0x0f))

#define U8G\_ESC\_VCC(x) 255, (0xbe | ((x)&0x01))

#define U8G\_ESC\_END 255, 254

#define U8G\_ESC\_255 255, 255

//uint8\_t u8g\_WriteEscSeqP(u8g\_t \*u8g, u8g\_dev\_t \*dev, u8g\_pgm\_uint8\_t \*esc\_seq);

uint8\_t u8g\_WriteEscSeqP(u8g\_t \*u8g, u8g\_dev\_t \*dev, const uint8\_t \*esc\_seq);

/\* u8g\_com\_api\_16gr.c \*/

uint8\_t u8g\_WriteByteBWTo16GrDevice(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t b);

uint8\_t u8g\_WriteSequenceBWTo16GrDevice(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t cnt, uint8\_t \*ptr);

uint8\_t u8g\_WriteByte4LTo16GrDevice(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t b);

uint8\_t u8g\_WriteSequence4LTo16GrDevice(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t cnt, uint8\_t \*ptr);

/\*===============================================================\*/

/\* u8g\_arduino\_common.c \*/

void u8g\_com\_arduino\_digital\_write(u8g\_t \*u8g, uint8\_t pin\_index, uint8\_t value);

void u8g\_com\_arduino\_assign\_pin\_output\_high(u8g\_t \*u8g);

/\*===============================================================\*/

/\* u8g\_com\_io.c \*/

/\* create internal number from port and pin \*/

uint8\_t u8g\_Pin(uint8\_t port, uint8\_t bitpos);

#define PN(port,bitpos) u8g\_Pin(port,bitpos)

/\* low level procedures \*/

void u8g\_SetPinOutput(uint8\_t internal\_pin\_number);

void u8g\_SetPinLevel(uint8\_t internal\_pin\_number, uint8\_t level);

void u8g\_SetPinInput(uint8\_t internal\_pin\_number);

uint8\_t u8g\_GetPinLevel(uint8\_t internal\_pin\_number);

/\* u8g level procedures, expect U8G\_PI\_xxx macro \*/

void u8g\_SetPIOutput(u8g\_t \*u8g, uint8\_t pi);

void u8g\_SetPILevel(u8g\_t \*u8g, uint8\_t pi, uint8\_t level);

/\*===============================================================\*/

/\* page \*/

struct \_u8g\_page\_t

{

u8g\_uint\_t page\_height;

u8g\_uint\_t total\_height;

u8g\_uint\_t page\_y0;

u8g\_uint\_t page\_y1;

uint8\_t page;

};

typedef struct \_u8g\_page\_t u8g\_page\_t;

void u8g\_page\_First(u8g\_page\_t \*p) U8G\_NOINLINE; /\* u8g\_page.c \*/

void u8g\_page\_Init(u8g\_page\_t \*p, u8g\_uint\_t page\_height, u8g\_uint\_t total\_height ) U8G\_NOINLINE; /\* u8g\_page.c \*/

uint8\_t u8g\_page\_Next(u8g\_page\_t \*p) U8G\_NOINLINE; /\* u8g\_page.c \*/

/\*===============================================================\*/

/\* page buffer (pb) \*/

struct \_u8g\_pb\_t

{

u8g\_page\_t p;

u8g\_uint\_t width; /\* pixel width \*/

void \*buf;

};

typedef struct \_u8g\_pb\_t u8g\_pb\_t;

/\* u8g\_pb.c \*/

void u8g\_pb\_Clear(u8g\_pb\_t \*b);

uint8\_t u8g\_pb\_IsYIntersection(u8g\_pb\_t \*pb, u8g\_uint\_t v0, u8g\_uint\_t v1);

uint8\_t u8g\_pb\_IsXIntersection(u8g\_pb\_t \*b, u8g\_uint\_t v0, u8g\_uint\_t v1);

uint8\_t u8g\_pb\_IsIntersection(u8g\_pb\_t \*pb, u8g\_dev\_arg\_bbx\_t \*bbx);

void u8g\_pb\_GetPageBox(u8g\_pb\_t \*pb, u8g\_box\_t \*box);

uint8\_t u8g\_pb\_Is8PixelVisible(u8g\_pb\_t \*b, u8g\_dev\_arg\_pixel\_t \*arg\_pixel);

uint8\_t u8g\_pb\_WriteBuffer(u8g\_pb\_t \*b, u8g\_t \*u8g, u8g\_dev\_t \*dev);

/\*

note on \_\_attribute\_\_ ((nocommon))

AVR scripts often use --gc-sections on the linker to remove unused section.

This works fine for initialed data and text sections. In principle .bss is also

handled, but the name##\_pb definition is not removed. Reason is, that

array definitions are placed in the COMMON section, by default

The attribute "nocommon" removes this automatic assignment to the

COMMON section and directly puts it into .bss. As a result, if more

than one buffer is defined in one file, then it will be removed with --gc-sections

.. not sure if Arduino IDE uses -fno-common... if yes, then the attribute is

redundant.

\*/

#define U8G\_PB\_DEV(name, width, height, page\_height, dev\_fn, com\_fn) \

uint8\_t name##\_buf[width] U8G\_NOCOMMON ; \

u8g\_pb\_t name##\_pb = { {page\_height, height, 0, 0, 0}, width, name##\_buf}; \

u8g\_dev\_t name = { dev\_fn, &name##\_pb, com\_fn }

void u8g\_pb8v1\_Init(u8g\_pb\_t \*b, void \*buf, u8g\_uint\_t width) U8G\_NOINLINE;

void u8g\_pb8v1\_Clear(u8g\_pb\_t \*b) U8G\_NOINLINE;

uint8\_t u8g\_pb8v1\_IsYIntersection(u8g\_pb\_t \*b, u8g\_uint\_t v0, u8g\_uint\_t v1);

uint8\_t u8g\_pb8v1\_IsXIntersection(u8g\_pb\_t \*b, u8g\_uint\_t v0, u8g\_uint\_t v1);

uint8\_t u8g\_pb8v1\_WriteBuffer(u8g\_pb\_t \*b, u8g\_t \*u8g, u8g\_dev\_t \*dev);

uint8\_t u8g\_dev\_pb8v1\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb16v1.c \*/

uint8\_t u8g\_dev\_pb16v1\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb14v1.c \*/

uint8\_t u8g\_dev\_pb14v1\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb8v2.c \*/

uint8\_t u8g\_dev\_pb8v2\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb16v2.c (double memory of pb8v2) \*/

uint8\_t u8g\_dev\_pb16v2\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb8h1.c \*/

uint8\_t u8g\_dev\_pb8h1\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb16h1.c \*/

uint8\_t u8g\_dev\_pb16h1\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb32h1.c \*/

uint8\_t u8g\_dev\_pb32h1\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb8h2.c 8 pixel rows, byte has horzontal orientation \*/

uint8\_t u8g\_dev\_pb8h2\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb16h2.c \*/

uint8\_t u8g\_dev\_pb16h2\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb8h1f.c \*/

uint8\_t u8g\_dev\_pb8h1f\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pb8h8.c \*/

uint8\_t u8g\_dev\_pb8h8\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pbxh16.c \*/

uint8\_t u8g\_dev\_pbxh16\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\* u8g\_pbxh24.c \*/

uint8\_t u8g\_dev\_pbxh24\_base\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

/\*===============================================================\*/

/\* u8g\_ll\_api.c \*/

/\* cursor draw callback \*/

typedef void (\*u8g\_draw\_cursor\_fn)(u8g\_t \*u8g);

/\* vertical reference point calculation callback \*/

typedef u8g\_uint\_t (\*u8g\_font\_calc\_vref\_fnptr)(u8g\_t \*u8g);

/\* state backup and restore procedure \*/

typedef void (\*u8g\_state\_cb)(uint8\_t msg);

/\* PI = Pin Index \*/

/\* reset pin, usually optional \*/

#define U8G\_PI\_RESET 0

/\* address / data or instruction \*/

#define U8G\_PI\_A0 1

#define U8G\_PI\_DI 1

/\* chip select line \*/

#define U8G\_PI\_CS 2

#define U8G\_PI\_CS1 2

#define U8G\_PI\_CS2 3

/\* Feb 2013: A0 state moved from 7 to 3 for t6963 controller\*/

#define U8G\_PI\_A0\_STATE 3

/\* enable / clock signal \*/

#define U8G\_PI\_EN 4

#define U8G\_PI\_CS\_STATE 4

#define U8G\_PI\_SCK 4

#define U8G\_PI\_SCL 4

#define U8G\_PI\_RD 4

/\* data pins, shared with SPI and I2C pins \*/

#define U8G\_PI\_D0 5

#define U8G\_PI\_MOSI 5

#define U8G\_PI\_SDA 5

#define U8G\_PI\_D1 6

#define U8G\_PI\_MISO 6

#define U8G\_PI\_D2 7

#define U8G\_PI\_D3 8

#define U8G\_PI\_SET\_A0 8

#define U8G\_PI\_D4 9

#define U8G\_PI\_D5 10

#define U8G\_PI\_I2C\_OPTION 11

#define U8G\_PI\_D6 11

#define U8G\_PI\_D7 12

/\* read/write pin, must be the last pin in the list, this means U8G\_PIN\_LIST\_LEN = U8G\_PI\_RW + 1\*/

#define U8G\_PI\_WR 13

#define U8G\_PI\_RW 13

#define U8G\_PIN\_LIST\_LEN 14

#define U8G\_PIN\_DUMMY 254

#define U8G\_PIN\_NONE 255

#define U8G\_FONT\_HEIGHT\_MODE\_TEXT 0

#define U8G\_FONT\_HEIGHT\_MODE\_XTEXT 1

#define U8G\_FONT\_HEIGHT\_MODE\_ALL 2

struct \_u8g\_t

{

u8g\_uint\_t width;

u8g\_uint\_t height;

u8g\_dev\_t \*dev; /\* first device in the device chain \*/

const u8g\_pgm\_uint8\_t \*font; /\* regular font for all text procedures \*/

const u8g\_pgm\_uint8\_t \*cursor\_font; /\* special font for cursor procedures \*/

uint8\_t cursor\_fg\_color, cursor\_bg\_color;

uint8\_t cursor\_encoding;

uint8\_t mode; /\* display mode, one of U8G\_MODE\_xxx \*/

u8g\_uint\_t cursor\_x;

u8g\_uint\_t cursor\_y;

u8g\_draw\_cursor\_fn cursor\_fn;

int8\_t glyph\_dx;

int8\_t glyph\_x;

int8\_t glyph\_y;

uint8\_t glyph\_width;

uint8\_t glyph\_height;

u8g\_font\_calc\_vref\_fnptr font\_calc\_vref;

uint8\_t font\_height\_mode;

int8\_t font\_ref\_ascent;

int8\_t font\_ref\_descent;

uint8\_t font\_line\_spacing\_factor; /\* line\_spacing = factor \* (ascent - descent) / 64 \*/

uint8\_t line\_spacing;

u8g\_dev\_arg\_pixel\_t arg\_pixel;

/\* uint8\_t color\_index; \*/

#ifdef U8G\_WITH\_PINLIST

uint8\_t pin\_list[U8G\_PIN\_LIST\_LEN];

#endif

u8g\_state\_cb state\_cb;

u8g\_box\_t current\_page; /\* current box of the visible page \*/

};

#define u8g\_GetFontAscent(u8g) ((u8g)->font\_ref\_ascent)

#define u8g\_GetFontDescent(u8g) ((u8g)->font\_ref\_descent)

#define u8g\_GetFontLineSpacing(u8g) ((u8g)->line\_spacing)

uint8\_t u8g\_call\_dev\_fn(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t msg, void \*arg);

uint8\_t u8g\_InitLL(u8g\_t \*u8g, u8g\_dev\_t \*dev);

void u8g\_FirstPageLL(u8g\_t \*u8g, u8g\_dev\_t \*dev);

uint8\_t u8g\_NextPageLL(u8g\_t \*u8g, u8g\_dev\_t \*dev);

uint8\_t u8g\_SetContrastLL(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t contrast);

void u8g\_DrawPixelLL(u8g\_t \*u8g, u8g\_dev\_t \*dev, u8g\_uint\_t x, u8g\_uint\_t y);

void u8g\_Draw8PixelLL(u8g\_t \*u8g, u8g\_dev\_t \*dev, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t dir, uint8\_t pixel);

void u8g\_Draw4TPixelLL(u8g\_t \*u8g, u8g\_dev\_t \*dev, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t dir, uint8\_t pixel);

uint8\_t u8g\_IsBBXIntersectionLL(u8g\_t \*u8g, u8g\_dev\_t \*dev, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h); /\* obsolete \*/

u8g\_uint\_t u8g\_GetWidthLL(u8g\_t \*u8g, u8g\_dev\_t \*dev);

u8g\_uint\_t u8g\_GetHeightLL(u8g\_t \*u8g, u8g\_dev\_t \*dev);

void u8g\_UpdateDimension(u8g\_t \*u8g);

uint8\_t u8g\_Begin(u8g\_t \*u8g); /\* reset device, put it into default state and call u8g\_UpdateDimension() \*/

uint8\_t u8g\_Init(u8g\_t \*u8g, u8g\_dev\_t \*dev); /\* only usefull if the device only as hardcoded ports \*/

uint8\_t u8g\_InitComFn(u8g\_t \*u8g, u8g\_dev\_t \*dev, u8g\_com\_fnptr com\_fn); /\* Init procedure for anything which is not Arduino or AVR (e.g. ARM, but not Due, which is Arduino) \*/

uint8\_t u8g\_InitSPI(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t sck, uint8\_t mosi, uint8\_t cs, uint8\_t a0, uint8\_t reset);

uint8\_t u8g\_InitHWSPI(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t cs, uint8\_t a0, uint8\_t reset);

uint8\_t u8g\_InitI2C(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t options); /\* use U8G\_I2C\_OPT\_NONE as options \*/

uint8\_t u8g\_Init8BitFixedPort(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t en, uint8\_t cs, uint8\_t di, uint8\_t rw, uint8\_t reset);

uint8\_t u8g\_Init8Bit(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t d0, uint8\_t d1, uint8\_t d2, uint8\_t d3, uint8\_t d4, uint8\_t d5, uint8\_t d6, uint8\_t d7,

uint8\_t en, uint8\_t cs1, uint8\_t cs2, uint8\_t di, uint8\_t rw, uint8\_t reset);

uint8\_t u8g\_InitRW8Bit(u8g\_t \*u8g, u8g\_dev\_t \*dev, uint8\_t d0, uint8\_t d1, uint8\_t d2, uint8\_t d3, uint8\_t d4, uint8\_t d5, uint8\_t d6, uint8\_t d7,

uint8\_t cs, uint8\_t a0, uint8\_t wr, uint8\_t rd, uint8\_t reset);

void u8g\_FirstPage(u8g\_t \*u8g);

uint8\_t u8g\_NextPage(u8g\_t \*u8g);

uint8\_t u8g\_SetContrast(u8g\_t \*u8g, uint8\_t contrast);

void u8g\_SleepOn(u8g\_t \*u8g);

void u8g\_SleepOff(u8g\_t \*u8g);

void u8g\_DrawPixel(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y);

void u8g\_Draw8Pixel(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t dir, uint8\_t pixel);

void u8g\_Draw4TPixel(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t dir, uint8\_t pixel);

uint8\_t u8g\_Stop(u8g\_t \*u8g);

void u8g\_SetColorEntry(u8g\_t \*u8g, uint8\_t idx, uint8\_t r, uint8\_t g, uint8\_t b);

void u8g\_SetColorIndex(u8g\_t \*u8g, uint8\_t idx);

void u8g\_SetHiColor(u8g\_t \*u8g, uint16\_t rgb);

void u8g\_SetHiColorByRGB(u8g\_t \*u8g, uint8\_t r, uint8\_t g, uint8\_t b);

void u8g\_SetRGB(u8g\_t \*u8g, uint8\_t r, uint8\_t g, uint8\_t b);

uint8\_t u8g\_GetColorIndex(u8g\_t \*u8g);

uint8\_t u8g\_GetDefaultForegroundColor(u8g\_t \*u8g);

void u8g\_SetDefaultForegroundColor(u8g\_t \*u8g);

uint8\_t u8g\_GetDefaultBackgroundColor(u8g\_t \*u8g);

void u8g\_SetDefaultBackgroundColor(u8g\_t \*u8g);

uint8\_t u8g\_GetDefaultMidColor(u8g\_t \*u8g);

void u8g\_SetDefaultMidColor(u8g\_t \*u8g);

#define u8g\_GetWidth(u8g) ((u8g)->width)

#define u8g\_GetHeight(u8g) ((u8g)->height)

#define u8g\_GetMode(u8g) ((u8g)->mode)

/\*

U8G\_MODE\_GET\_BITS\_PER\_PIXEL(u8g\_GetMode(u8g))

U8G\_MODE\_IS\_COLOR(u8g\_GetMode(u8g))

\*/

/\* u8g\_state.c \*/

#define U8G\_STATE\_ENV\_IDX 0

#define U8G\_STATE\_U8G\_IDX 1

#define U8G\_STATE\_RESTORE 0

#define U8G\_STATE\_BACKUP 1

#define U8G\_STATE\_MSG\_COMPOSE(cmd,idx) (((cmd)<<1) | (idx))

#define U8G\_STATE\_MSG\_RESTORE\_ENV U8G\_STATE\_MSG\_COMPOSE(U8G\_STATE\_RESTORE,U8G\_STATE\_ENV\_IDX)

#define U8G\_STATE\_MSG\_BACKUP\_ENV U8G\_STATE\_MSG\_COMPOSE(U8G\_STATE\_BACKUP,U8G\_STATE\_ENV\_IDX)

#define U8G\_STATE\_MSG\_RESTORE\_U8G U8G\_STATE\_MSG\_COMPOSE(U8G\_STATE\_RESTORE,U8G\_STATE\_U8G\_IDX)

#define U8G\_STATE\_MSG\_BACKUP\_U8G U8G\_STATE\_MSG\_COMPOSE(U8G\_STATE\_BACKUP,U8G\_STATE\_U8G\_IDX)

#define U8G\_STATE\_MSG\_GET\_IDX(msg) ((msg)&1)

#define U8G\_STATE\_MSG\_IS\_BACKUP(msg) ((msg)&2)

void u8g\_state\_dummy\_cb(uint8\_t msg);

void u8g\_backup\_spi(uint8\_t msg); /\* backup SPI state controller \*/

/\* backward compatible definition \*/

#define u8g\_backup\_avr\_spi u8g\_backup\_spi

void u8g\_SetHardwareBackup(u8g\_t \*u8g, u8g\_state\_cb backup\_cb);

/\* u8g\_clip.c \*/

uint8\_t u8g\_IsBBXIntersection(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h);

/\* u8g\_rot.c \*/

void u8g\_UndoRotation(u8g\_t \*u8g);

void u8g\_SetRot90(u8g\_t \*u8g);

void u8g\_SetRot180(u8g\_t \*u8g);

void u8g\_SetRot270(u8g\_t \*u8g);

/\* u8g\_scale.c \*/

void u8g\_UndoScale(u8g\_t \*u8g);

void u8g\_SetScale2x2(u8g\_t \*u8g);

/\* u8g\_font.c \*/

size\_t u8g\_font\_GetSize(const void \*font);

uint8\_t u8g\_font\_GetFontStartEncoding(const void \*font) U8G\_NOINLINE;

uint8\_t u8g\_font\_GetFontEndEncoding(const void \*font) U8G\_NOINLINE;

void u8g\_SetFont(u8g\_t \*u8g, const u8g\_fntpgm\_uint8\_t \*font);

uint8\_t u8g\_GetFontBBXWidth(u8g\_t \*u8g);

uint8\_t u8g\_GetFontBBXHeight(u8g\_t \*u8g);

int8\_t u8g\_GetFontBBXOffX(u8g\_t \*u8g);

int8\_t u8g\_GetFontBBXOffY(u8g\_t \*u8g);

uint8\_t u8g\_GetFontCapitalAHeight(u8g\_t \*u8g);

uint8\_t u8g\_IsGlyph(u8g\_t \*u8g, uint8\_t requested\_encoding);

int8\_t u8g\_GetGlyphDeltaX(u8g\_t \*u8g, uint8\_t requested\_encoding);

int8\_t u8g\_draw\_glyph(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t encoding); /\* used by u8g\_cursor.c \*/

int8\_t u8g\_DrawGlyphDir(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t dir, uint8\_t encoding);

int8\_t u8g\_DrawGlyph(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t encoding);

int8\_t u8g\_DrawGlyph90(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t encoding);

int8\_t u8g\_DrawGlyph180(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t encoding);

int8\_t u8g\_DrawGlyph270(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t encoding);

int8\_t u8g\_DrawGlyphFontBBX(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t dir, uint8\_t encoding);

u8g\_uint\_t u8g\_DrawStr(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const char \*s);

u8g\_uint\_t u8g\_DrawStr90(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const char \*s);

u8g\_uint\_t u8g\_DrawStr180(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const char \*s);

u8g\_uint\_t u8g\_DrawStr270(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const char \*s);

u8g\_uint\_t u8g\_DrawStrDir(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t dir, const char \*s);

u8g\_uint\_t u8g\_DrawStrP(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const u8g\_pgm\_uint8\_t \*s);

u8g\_uint\_t u8g\_DrawStr90P(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const u8g\_pgm\_uint8\_t \*s);

u8g\_uint\_t u8g\_DrawStr180P(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const u8g\_pgm\_uint8\_t \*s);

u8g\_uint\_t u8g\_DrawStr270P(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const u8g\_pgm\_uint8\_t \*s);

void u8g\_SetFontRefHeightText(u8g\_t \*u8g);

void u8g\_SetFontRefHeightExtendedText(u8g\_t \*u8g);

void u8g\_SetFontRefHeightAll(u8g\_t \*u8g);

void u8g\_SetFontLineSpacingFactor(u8g\_t \*u8g, uint8\_t factor);

u8g\_uint\_t u8g\_font\_calc\_vref\_font(u8g\_t \*u8g);

u8g\_uint\_t u8g\_font\_calc\_vref\_bottom(u8g\_t \*u8g);

u8g\_uint\_t u8g\_font\_calc\_vref\_top(u8g\_t \*u8g);

u8g\_uint\_t u8g\_font\_calc\_vref\_center(u8g\_t \*u8g);

void u8g\_SetFontPosBaseline(u8g\_t \*u8g);

void u8g\_SetFontPosBottom(u8g\_t \*u8g);

void u8g\_SetFontPosCenter(u8g\_t \*u8g);

void u8g\_SetFontPosTop(u8g\_t \*u8g);

u8g\_uint\_t u8g\_GetStrPixelWidth(u8g\_t \*u8g, const char \*s);

u8g\_uint\_t u8g\_GetStrPixelWidthP(u8g\_t \*u8g, const u8g\_pgm\_uint8\_t \*s);

int8\_t u8g\_GetStrX(u8g\_t \*u8g, const char \*s);

int8\_t u8g\_GetStrXP(u8g\_t \*u8g, const u8g\_pgm\_uint8\_t \*s);

u8g\_uint\_t u8g\_GetStrWidth(u8g\_t \*u8g, const char \*s) U8G\_NOINLINE;

u8g\_uint\_t u8g\_GetStrWidthP(u8g\_t \*u8g, const u8g\_pgm\_uint8\_t \*s);

u8g\_uint\_t u8g\_DrawStrFontBBX(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, uint8\_t dir, const char \*s);

void u8g\_GetStrMinBox(u8g\_t \*u8g, const char \*s, u8g\_uint\_t \*x, u8g\_uint\_t \*y, u8g\_uint\_t \*width, u8g\_uint\_t \*height);

void u8g\_GetStrAMinBox(u8g\_t \*u8g, const char \*s, u8g\_uint\_t \*x, u8g\_uint\_t \*y, u8g\_uint\_t \*width, u8g\_uint\_t \*height);

u8g\_uint\_t u8g\_DrawAAStr(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, const char \*s);

/\* u8g\_rect.c \*/

void u8g\_draw\_box(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h) U8G\_NOINLINE;

void u8g\_DrawHLine(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w) U8G\_NOINLINE;

void u8g\_DrawVLine(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w) U8G\_NOINLINE;

void u8g\_DrawFrame(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h) U8G\_NOINLINE;

void u8g\_DrawBox(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h) U8G\_NOINLINE;

void u8g\_DrawRFrame(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h, u8g\_uint\_t r) U8G\_NOINLINE;

void u8g\_DrawRBox(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h, u8g\_uint\_t r) U8G\_NOINLINE;

/\* u8g\_bitmap.c \*/

void u8g\_DrawHBitmap(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t cnt, const uint8\_t \*bitmap);

void u8g\_DrawHBitmapP(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t cnt, const u8g\_pgm\_uint8\_t \*bitmap);

void u8g\_DrawBitmap(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t cnt, u8g\_uint\_t h, const uint8\_t \*bitmap);

void u8g\_DrawBitmapP(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t cnt, u8g\_uint\_t h, const u8g\_pgm\_uint8\_t \*bitmap);

void u8g\_DrawXBM(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h, const uint8\_t \*bitmap);

void u8g\_DrawXBMP(u8g\_t \*u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_uint\_t w, u8g\_uint\_t h, const u8g\_pgm\_uint8\_t \*bitmap);

/\* u8g\_line.c \*/

void u8g\_DrawLine(u8g\_t \*u8g, u8g\_uint\_t x1, u8g\_uint\_t y1, u8g\_uint\_t x2, u8g\_uint\_t y2);

/\* u8g\_circle.c \*/

/\* the following, commented code has been rewritten or is not yet finished

#define U8G\_CIRC\_UPPER\_RIGHT 0x01

#define U8G\_CIRC\_UPPER\_LEFT 0x02

#define U8G\_CIRC\_LOWER\_LEFT 0x04

#define U8G\_CIRC\_LOWER\_RIGHT 0x08

#define U8G\_CIRC\_ALL (U8G\_CIRC\_UPPER\_RIGHT|U8G\_CIRC\_UPPER\_LEFT|U8G\_CIRC\_LOWER\_RIGHT|U8G\_CIRC\_LOWER\_LEFT)

void u8g\_DrawEmpCirc(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t rad, uint8\_t option);

void u8g\_DrawFillCirc(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t rad, uint8\_t option);

void u8g\_DrawEllipseRect(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t x1, u8g\_uint\_t y1);

\*/

#define U8G\_DRAW\_UPPER\_RIGHT 0x01

#define U8G\_DRAW\_UPPER\_LEFT 0x02

#define U8G\_DRAW\_LOWER\_LEFT 0x04

#define U8G\_DRAW\_LOWER\_RIGHT 0x08

#define U8G\_DRAW\_ALL (U8G\_DRAW\_UPPER\_RIGHT|U8G\_DRAW\_UPPER\_LEFT|U8G\_DRAW\_LOWER\_RIGHT|U8G\_DRAW\_LOWER\_LEFT)

void u8g\_draw\_circle(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t rad, uint8\_t option) U8G\_NOINLINE;

void u8g\_draw\_disc(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t rad, uint8\_t option) U8G\_NOINLINE;

void u8g\_DrawCircle(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t rad, uint8\_t option);

void u8g\_DrawDisc(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t rad, uint8\_t option);

/\* u8g\_ellipse.c \*/

void u8g\_DrawEllipse(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t rx, u8g\_uint\_t ry, uint8\_t option);

void u8g\_DrawFilledEllipse(u8g\_t \*u8g, u8g\_uint\_t x0, u8g\_uint\_t y0, u8g\_uint\_t rx, u8g\_uint\_t ry, uint8\_t option);

/\* u8g\_clip.c \*/

uint8\_t u8g\_is\_box\_bbx\_intersection(u8g\_box\_t \*box, u8g\_dev\_arg\_bbx\_t \*bbx);

/\* u8g\_cursor.c \*/

void u8g\_SetCursorFont(u8g\_t \*u8g, const u8g\_pgm\_uint8\_t \*cursor\_font);

void u8g\_SetCursorStyle(u8g\_t \*u8g, uint8\_t encoding);

void u8g\_SetCursorPos(u8g\_t \*u8g, u8g\_uint\_t cursor\_x, u8g\_uint\_t cursor\_y);

void u8g\_SetCursorColor(u8g\_t \*u8g, uint8\_t fg, uint8\_t bg);

void u8g\_EnableCursor(u8g\_t \*u8g);

void u8g\_DisableCursor(u8g\_t \*u8g);

void u8g\_DrawCursor(u8g\_t \*u8g);

/\* u8g\_polygon.c \*/

typedef int16\_t pg\_word\_t;

#define PG\_NOINLINE U8G\_NOINLINE

struct pg\_point\_struct

{

pg\_word\_t x;

pg\_word\_t y;

};

typedef struct \_pg\_struct pg\_struct; /\* forward declaration \*/

struct pg\_edge\_struct

{

pg\_word\_t x\_direction; /\* 1, if x2 is greater than x1, -1 otherwise \*/

pg\_word\_t height;

pg\_word\_t current\_x\_offset;

pg\_word\_t error\_offset;

/\* --- line loop --- \*/

pg\_word\_t current\_y;

pg\_word\_t max\_y;

pg\_word\_t current\_x;

pg\_word\_t error;

/\* --- outer loop --- \*/

uint8\_t (\*next\_idx\_fn)(pg\_struct \*pg, uint8\_t i);

uint8\_t curr\_idx;

};

/\* maximum number of points in the polygon \*/

/\* can be redefined, but highest possible value is 254 \*/

#define PG\_MAX\_POINTS 6

/\* index numbers for the pge structures below \*/

#define PG\_LEFT 0

#define PG\_RIGHT 1

struct \_pg\_struct

{

struct pg\_point\_struct list[PG\_MAX\_POINTS];

uint8\_t cnt;

uint8\_t is\_min\_y\_not\_flat;

pg\_word\_t total\_scan\_line\_cnt;

struct pg\_edge\_struct pge[2]; /\* left and right line draw structures \*/

};

void pg\_ClearPolygonXY(pg\_struct \*pg);

void pg\_AddPolygonXY(pg\_struct \*pg, u8g\_t \*u8g, int16\_t x, int16\_t y);

void pg\_DrawPolygon(pg\_struct \*pg, u8g\_t \*u8g);

void u8g\_ClearPolygonXY(void);

void u8g\_AddPolygonXY(u8g\_t \*u8g, int16\_t x, int16\_t y);

void u8g\_DrawPolygon(u8g\_t \*u8g);

void u8g\_DrawTriangle(u8g\_t \*u8g, int16\_t x0, int16\_t y0, int16\_t x1, int16\_t y1, int16\_t x2, int16\_t y2);

/\*===============================================================\*/

/\* u8g\_virtual\_screen.c \*/

void u8g\_SetVirtualScreenDimension(u8g\_t \*vs\_u8g, u8g\_uint\_t width, u8g\_uint\_t height);

uint8\_t u8g\_AddToVirtualScreen(u8g\_t \*vs\_u8g, u8g\_uint\_t x, u8g\_uint\_t y, u8g\_t \*child\_u8g);

/\*===============================================================\*/

void st\_Draw(uint8\_t fps);

void st\_Step(uint8\_t player\_pos, uint8\_t is\_auto\_fire, uint8\_t is\_fire);

/\*===============================================================\*/

/\* u8g\_com\_i2c.c \*/

/\* options for u8g\_i2c\_init() \*/

#define U8G\_I2C\_OPT\_NONE 0

#define U8G\_I2C\_OPT\_NO\_ACK 2

#define U8G\_I2C\_OPT\_DEV\_0 0

#define U8G\_I2C\_OPT\_DEV\_1 4

#define U8G\_I2C\_OPT\_FAST 16

/\* retrun values from u8g\_twi\_get\_error() \*/

#define U8G\_I2C\_ERR\_NONE 0x00

/\* the following values are bit masks \*/

#define U8G\_I2C\_ERR\_TIMEOUT 0x01

#define U8G\_I2C\_ERR\_BUS 0x02

void u8g\_i2c\_clear\_error(void) U8G\_NOINLINE;

uint8\_t u8g\_i2c\_get\_error(void) U8G\_NOINLINE;

uint8\_t u8g\_i2c\_get\_err\_pos(void) U8G\_NOINLINE;

void u8g\_i2c\_init(uint8\_t options) U8G\_NOINLINE; /\* use U8G\_I2C\_OPT\_NONE as options \*/

uint8\_t u8g\_i2c\_wait(uint8\_t mask, uint8\_t pos) U8G\_NOINLINE;

uint8\_t u8g\_i2c\_start(uint8\_t sla) U8G\_NOINLINE;

uint8\_t u8g\_i2c\_send\_byte(uint8\_t data) U8G\_NOINLINE;

uint8\_t u8g\_i2c\_send\_mode(uint8\_t mode) U8G\_NOINLINE;

void u8g\_i2c\_stop(void) U8G\_NOINLINE;

/\*===============================================================\*/

/\* u8g\_u8toa.c \*/

/\* v = value, d = number of digits \*/

const char \*u8g\_u8toa(uint8\_t v, uint8\_t d);

/\* u8g\_u8toa.c \*/

/\* v = value, d = number of digits \*/

const char \*u8g\_u16toa(uint16\_t v, uint8\_t d);

/\*===============================================================\*/

/\* u8g\_delay.c \*/

/\* delay by the specified number of milliseconds \*/

void u8g\_Delay(uint16\_t val);

/\* delay by one microsecond \*/

void u8g\_MicroDelay(void);

/\* delay by 10 microseconds \*/

void u8g\_10MicroDelay(void);

/\*===============================================================\*/

/\* chessengine.c \*/

#define CHESS\_KEY\_NONE 0

#define CHESS\_KEY\_NEXT 1

#define CHESS\_KEY\_PREV 2

#define CHESS\_KEY\_SELECT 3

#define CHESS\_KEY\_BACK 4

void chess\_Init(u8g\_t \*u8g, uint8\_t empty\_body\_color);

void chess\_Draw(void);

void chess\_Step(uint8\_t keycode);

/\*===============================================================\*/

/\* font definitions \*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_m2icon\_5[] U8G\_FONT\_SECTION("u8g\_font\_m2icon\_5");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_m2icon\_7[] U8G\_FONT\_SECTION("u8g\_font\_m2icon\_7");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_m2icon\_9[] U8G\_FONT\_SECTION("u8g\_font\_m2icon\_9");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_u8glib\_4[] U8G\_FONT\_SECTION("u8g\_font\_u8glib\_4");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_u8glib\_4r[] U8G\_FONT\_SECTION("u8g\_font\_u8glib\_4r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x12\_75r[] U8G\_FONT\_SECTION("u8g\_font\_6x12\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13\_75r[] U8G\_FONT\_SECTION("u8g\_font\_6x13\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13\_75r[] U8G\_FONT\_SECTION("u8g\_font\_7x13\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_8x13\_75r[] U8G\_FONT\_SECTION("u8g\_font\_8x13\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x15\_75r[] U8G\_FONT\_SECTION("u8g\_font\_9x15\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x18\_75r[] U8G\_FONT\_SECTION("u8g\_font\_9x18\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_cu12\_75r[] U8G\_FONT\_SECTION("u8g\_font\_cu12\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_75r[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_10x20\_75r[] U8G\_FONT\_SECTION("u8g\_font\_10x20\_75r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_10x20\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_10x20\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_10x20\_78\_79[] U8G\_FONT\_SECTION("u8g\_font\_10x20\_78\_79");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_10x20[] U8G\_FONT\_SECTION("u8g\_font\_10x20");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_10x20r[] U8G\_FONT\_SECTION("u8g\_font\_10x20r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_4x6[] U8G\_FONT\_SECTION("u8g\_font\_4x6");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_4x6r[] U8G\_FONT\_SECTION("u8g\_font\_4x6r");

//extern const u8g\_fntpgm\_uint8\_t u8g\_font\_4x6n[] U8G\_FONT\_SECTION("u8g\_font\_4x6n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_5x7[] U8G\_FONT\_SECTION("u8g\_font\_5x7");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_5x7r[] U8G\_FONT\_SECTION("u8g\_font\_5x7r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_5x8[] U8G\_FONT\_SECTION("u8g\_font\_5x8");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_5x8r[] U8G\_FONT\_SECTION("u8g\_font\_5x8r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x10[] U8G\_FONT\_SECTION("u8g\_font\_6x10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x10r[] U8G\_FONT\_SECTION("u8g\_font\_6x10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x12\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_6x12\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x12\_78\_79[] U8G\_FONT\_SECTION("u8g\_font\_6x12\_78\_79");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x12[] U8G\_FONT\_SECTION("u8g\_font\_6x12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x12r[] U8G\_FONT\_SECTION("u8g\_font\_6x12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_6x13\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13\_78\_79[] U8G\_FONT\_SECTION("u8g\_font\_6x13\_78\_79");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13B[] U8G\_FONT\_SECTION("u8g\_font\_6x13B");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13Br[] U8G\_FONT\_SECTION("u8g\_font\_6x13Br");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13[] U8G\_FONT\_SECTION("u8g\_font\_6x13");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13r[] U8G\_FONT\_SECTION("u8g\_font\_6x13r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13O[] U8G\_FONT\_SECTION("u8g\_font\_6x13O");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_6x13Or[] U8G\_FONT\_SECTION("u8g\_font\_6x13Or");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_7x13\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13\_78\_79[] U8G\_FONT\_SECTION("u8g\_font\_7x13\_78\_79");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13B[] U8G\_FONT\_SECTION("u8g\_font\_7x13B");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13Br[] U8G\_FONT\_SECTION("u8g\_font\_7x13Br");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13[] U8G\_FONT\_SECTION("u8g\_font\_7x13");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13r[] U8G\_FONT\_SECTION("u8g\_font\_7x13r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13O[] U8G\_FONT\_SECTION("u8g\_font\_7x13O");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x13Or[] U8G\_FONT\_SECTION("u8g\_font\_7x13Or");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x14B[] U8G\_FONT\_SECTION("u8g\_font\_7x14B");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x14Br[] U8G\_FONT\_SECTION("u8g\_font\_7x14Br");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x14[] U8G\_FONT\_SECTION("u8g\_font\_7x14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_7x14r[] U8G\_FONT\_SECTION("u8g\_font\_7x14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_8x13\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_8x13\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_8x13B[] U8G\_FONT\_SECTION("u8g\_font\_8x13B");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_8x13Br[] U8G\_FONT\_SECTION("u8g\_font\_8x13Br");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_8x13[] U8G\_FONT\_SECTION("u8g\_font\_8x13");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_8x13r[] U8G\_FONT\_SECTION("u8g\_font\_8x13r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_8x13O[] U8G\_FONT\_SECTION("u8g\_font\_8x13O");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_8x13Or[] U8G\_FONT\_SECTION("u8g\_font\_8x13Or");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x15\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_9x15\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x15\_78\_79[] U8G\_FONT\_SECTION("u8g\_font\_9x15\_78\_79");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x15B[] U8G\_FONT\_SECTION("u8g\_font\_9x15B");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x15Br[] U8G\_FONT\_SECTION("u8g\_font\_9x15Br");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x15[] U8G\_FONT\_SECTION("u8g\_font\_9x15");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x15r[] U8G\_FONT\_SECTION("u8g\_font\_9x15r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x18\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_9x18\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x18\_78\_79[] U8G\_FONT\_SECTION("u8g\_font\_9x18\_78\_79");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x18B[] U8G\_FONT\_SECTION("u8g\_font\_9x18B");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x18[] U8G\_FONT\_SECTION("u8g\_font\_9x18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x18Br[] U8G\_FONT\_SECTION("u8g\_font\_9x18Br");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_9x18r[] U8G\_FONT\_SECTION("u8g\_font\_9x18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_cursor[] U8G\_FONT\_SECTION("u8g\_font\_cursor");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_cursorr[] U8G\_FONT\_SECTION("u8g\_font\_cursorr");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_micro[] U8G\_FONT\_SECTION("u8g\_font\_micro");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_cu12\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_cu12\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_cu12\_78\_79[] U8G\_FONT\_SECTION("u8g\_font\_cu12\_78\_79");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_cu12[] U8G\_FONT\_SECTION("u8g\_font\_cu12");

/\*

Free-Universal Bold

r: Reduced char set (codes 32 - 128)

n: Numbers (codes 42 - 57)

no char: Full set (codes 32 - 255)

\*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub11[] U8G\_FONT\_SECTION("u8g\_font\_fub11");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub11r[] U8G\_FONT\_SECTION("u8g\_font\_fub11r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub11n[] U8G\_FONT\_SECTION("u8g\_font\_fub11n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub14[] U8G\_FONT\_SECTION("u8g\_font\_fub14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub14r[] U8G\_FONT\_SECTION("u8g\_font\_fub14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub14n[] U8G\_FONT\_SECTION("u8g\_font\_fub14n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub17[] U8G\_FONT\_SECTION("u8g\_font\_fub17");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub17r[] U8G\_FONT\_SECTION("u8g\_font\_fub17r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub17n[] U8G\_FONT\_SECTION("u8g\_font\_fub17n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub20[] U8G\_FONT\_SECTION("u8g\_font\_fub20");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub20r[] U8G\_FONT\_SECTION("u8g\_font\_fub20r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub20n[] U8G\_FONT\_SECTION("u8g\_font\_fub20n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub25[] U8G\_FONT\_SECTION("u8g\_font\_fub25");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub25r[] U8G\_FONT\_SECTION("u8g\_font\_fub25r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub25n[] U8G\_FONT\_SECTION("u8g\_font\_fub25n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub30[] U8G\_FONT\_SECTION("u8g\_font\_fub30");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub30r[] U8G\_FONT\_SECTION("u8g\_font\_fub30r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub30n[] U8G\_FONT\_SECTION("u8g\_font\_fub30n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub35n[] U8G\_FONT\_SECTION("u8g\_font\_fub35n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub42n[] U8G\_FONT\_SECTION("u8g\_font\_fub42n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fub49n[] U8G\_FONT\_SECTION("u8g\_font\_fub49n");

/\*

Free-Universal Regular

r: Reduced char set (codes 32 - 128)

n: Numbers (codes 42 - 57)

no char: Full set (codes 32 - 255)

\*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur11[] U8G\_FONT\_SECTION("u8g\_font\_fur11");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur11r[] U8G\_FONT\_SECTION("u8g\_font\_fur11r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur11n[] U8G\_FONT\_SECTION("u8g\_font\_fur11n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur14[] U8G\_FONT\_SECTION("u8g\_font\_fur14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur14r[] U8G\_FONT\_SECTION("u8g\_font\_fur14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur14n[] U8G\_FONT\_SECTION("u8g\_font\_fur14n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur17[] U8G\_FONT\_SECTION("u8g\_font\_fur17");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur17r[] U8G\_FONT\_SECTION("u8g\_font\_fur17r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur17n[] U8G\_FONT\_SECTION("u8g\_font\_fur17n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur20[] U8G\_FONT\_SECTION("u8g\_font\_fur20");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur20r[] U8G\_FONT\_SECTION("u8g\_font\_fur20r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur20n[] U8G\_FONT\_SECTION("u8g\_font\_fur20n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur25[] U8G\_FONT\_SECTION("u8g\_font\_fur25");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur25r[] U8G\_FONT\_SECTION("u8g\_font\_fur25r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur25n[] U8G\_FONT\_SECTION("u8g\_font\_fur25n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur30[] U8G\_FONT\_SECTION("u8g\_font\_fur30");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur30r[] U8G\_FONT\_SECTION("u8g\_font\_fur30r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur30n[] U8G\_FONT\_SECTION("u8g\_font\_fur30n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur35n[] U8G\_FONT\_SECTION("u8g\_font\_fur35n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur42n[] U8G\_FONT\_SECTION("u8g\_font\_fur42n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fur49n[] U8G\_FONT\_SECTION("u8g\_font\_fur49n");

/\*

Gentium Bold

r: Reduced char set (codes 32 - 128)

n: Numbers (codes 42 - 57)

no char: Full set (codes 32 - 255)

\*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb11[] U8G\_FONT\_SECTION("u8g\_font\_gdb11");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb12[] U8G\_FONT\_SECTION("u8g\_font\_gdb12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb14[] U8G\_FONT\_SECTION("u8g\_font\_gdb14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb17[] U8G\_FONT\_SECTION("u8g\_font\_gdb17");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb20[] U8G\_FONT\_SECTION("u8g\_font\_gdb20");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb25[] U8G\_FONT\_SECTION("u8g\_font\_gdb25");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb30[] U8G\_FONT\_SECTION("u8g\_font\_gdb30");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb11r[] U8G\_FONT\_SECTION("u8g\_font\_gdb11r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb12r[] U8G\_FONT\_SECTION("u8g\_font\_gdb12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb14r[] U8G\_FONT\_SECTION("u8g\_font\_gdb14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb17r[] U8G\_FONT\_SECTION("u8g\_font\_gdb17r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb20r[] U8G\_FONT\_SECTION("u8g\_font\_gdb20r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb25r[] U8G\_FONT\_SECTION("u8g\_font\_gdb25r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb30r[] U8G\_FONT\_SECTION("u8g\_font\_gdb30r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb11n[] U8G\_FONT\_SECTION("u8g\_font\_gdb11n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb12n[] U8G\_FONT\_SECTION("u8g\_font\_gdb12n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb14n[] U8G\_FONT\_SECTION("u8g\_font\_gdb14n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb17n[] U8G\_FONT\_SECTION("u8g\_font\_gdb17n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb20n[] U8G\_FONT\_SECTION("u8g\_font\_gdb20n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb25n[] U8G\_FONT\_SECTION("u8g\_font\_gdb25n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdb30n[] U8G\_FONT\_SECTION("u8g\_font\_gdb30n");

/\*

Gentium Regular

r: Reduced char set (codes 32 - 128)

n: Numbers (codes 42 - 57)

no char: Full set (codes 32 - 255)

\*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr9[] U8G\_FONT\_SECTION("u8g\_font\_gdr9");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr10[] U8G\_FONT\_SECTION("u8g\_font\_gdr10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr11[] U8G\_FONT\_SECTION("u8g\_font\_gdr11");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr12[] U8G\_FONT\_SECTION("u8g\_font\_gdr12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr14[] U8G\_FONT\_SECTION("u8g\_font\_gdr14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr17[] U8G\_FONT\_SECTION("u8g\_font\_gdr17");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr20[] U8G\_FONT\_SECTION("u8g\_font\_gdr20");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr25[] U8G\_FONT\_SECTION("u8g\_font\_gdr25");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr30[] U8G\_FONT\_SECTION("u8g\_font\_gdr30");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr9r[] U8G\_FONT\_SECTION("u8g\_font\_gdr9r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr10r[] U8G\_FONT\_SECTION("u8g\_font\_gdr10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr11r[] U8G\_FONT\_SECTION("u8g\_font\_gdr11r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr12r[] U8G\_FONT\_SECTION("u8g\_font\_gdr12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr14r[] U8G\_FONT\_SECTION("u8g\_font\_gdr14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr17r[] U8G\_FONT\_SECTION("u8g\_font\_gdr17r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr20r[] U8G\_FONT\_SECTION("u8g\_font\_gdr20r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr25r[] U8G\_FONT\_SECTION("u8g\_font\_gdr25r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr30r[] U8G\_FONT\_SECTION("u8g\_font\_gdr30r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr9n[] U8G\_FONT\_SECTION("u8g\_font\_gdr9n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr10n[] U8G\_FONT\_SECTION("u8g\_font\_gdr10n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr11n[] U8G\_FONT\_SECTION("u8g\_font\_gdr11n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr12n[] U8G\_FONT\_SECTION("u8g\_font\_gdr12n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr14n[] U8G\_FONT\_SECTION("u8g\_font\_gdr14n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr17n[] U8G\_FONT\_SECTION("u8g\_font\_gdr17n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr20n[] U8G\_FONT\_SECTION("u8g\_font\_gdr20n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr25n[] U8G\_FONT\_SECTION("u8g\_font\_gdr25n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_gdr30n[] U8G\_FONT\_SECTION("u8g\_font\_gdr30n");

/\*

Old-Standard Bold

r: Reduced char set (codes 32 - 128)

n: Numbers (codes 42 - 57)

no char: Full set (codes 32 - 255)

\*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb18[] U8G\_FONT\_SECTION("u8g\_font\_osb18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb21[] U8G\_FONT\_SECTION("u8g\_font\_osb21");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb26[] U8G\_FONT\_SECTION("u8g\_font\_osb26");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb29[] U8G\_FONT\_SECTION("u8g\_font\_osb29");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb35[] U8G\_FONT\_SECTION("u8g\_font\_osb35");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb18r[] U8G\_FONT\_SECTION("u8g\_font\_osb18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb21r[] U8G\_FONT\_SECTION("u8g\_font\_osb21r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb26r[] U8G\_FONT\_SECTION("u8g\_font\_osb26r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb29r[] U8G\_FONT\_SECTION("u8g\_font\_osb29r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb35r[] U8G\_FONT\_SECTION("u8g\_font\_osb35r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb18n[] U8G\_FONT\_SECTION("u8g\_font\_osb18n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb21n[] U8G\_FONT\_SECTION("u8g\_font\_osb21n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb26n[] U8G\_FONT\_SECTION("u8g\_font\_osb26n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb29n[] U8G\_FONT\_SECTION("u8g\_font\_osb29n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osb35n[] U8G\_FONT\_SECTION("u8g\_font\_osb35n");

/\*

Old-Standard Regular

r: Reduced char set (codes 32 - 128)

n: Numbers (codes 42 - 57)

no char: Full set (codes 32 - 255)

\*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr18[] U8G\_FONT\_SECTION("u8g\_font\_osr18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr21[] U8G\_FONT\_SECTION("u8g\_font\_osr21");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr26[] U8G\_FONT\_SECTION("u8g\_font\_osr26");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr29[] U8G\_FONT\_SECTION("u8g\_font\_osr29");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr35[] U8G\_FONT\_SECTION("u8g\_font\_osr35");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr18r[] U8G\_FONT\_SECTION("u8g\_font\_osr18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr21r[] U8G\_FONT\_SECTION("u8g\_font\_osr21r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr26r[] U8G\_FONT\_SECTION("u8g\_font\_osr26r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr29r[] U8G\_FONT\_SECTION("u8g\_font\_osr29r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr35r[] U8G\_FONT\_SECTION("u8g\_font\_osr35r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr18n[] U8G\_FONT\_SECTION("u8g\_font\_osr18n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr21n[] U8G\_FONT\_SECTION("u8g\_font\_osr21n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr26n[] U8G\_FONT\_SECTION("u8g\_font\_osr26n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr29n[] U8G\_FONT\_SECTION("u8g\_font\_osr29n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr35n[] U8G\_FONT\_SECTION("u8g\_font\_osr35n");

//extern const u8g\_fntpgm\_uint8\_t u8g\_font\_osr41[] U8G\_FONT\_SECTION("u8g\_font\_osr41");

/\* GNU unifont \*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_18\_19[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_18\_19");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_72\_73[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_72\_73");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_67\_75[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_67\_75");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_76[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_76");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_77[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_77");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_78\_79[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_78\_79");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_86[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_86");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont[] U8G\_FONT\_SECTION("u8g\_font\_unifont");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifontr[] U8G\_FONT\_SECTION("u8g\_font\_unifontr");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_0\_8[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_0\_8");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_2\_3[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_2\_3");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_4\_5[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_4\_5");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_8\_9[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_8\_9");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_unifont\_12\_13[] U8G\_FONT\_SECTION("u8g\_font\_unifont\_12\_13");

/\* 04b fonts \*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_03b[] U8G\_FONT\_SECTION("u8g\_font\_04b\_03b");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_03bn[] U8G\_FONT\_SECTION("u8g\_font\_04b\_03bn");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_03br[] U8G\_FONT\_SECTION("u8g\_font\_04b\_03br");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_03[] U8G\_FONT\_SECTION("u8g\_font\_04b\_03");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_03n[] U8G\_FONT\_SECTION("u8g\_font\_04b\_03n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_03r[] U8G\_FONT\_SECTION("u8g\_font\_04b\_03r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_24[] U8G\_FONT\_SECTION("u8g\_font\_04b\_24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_24n[] U8G\_FONT\_SECTION("u8g\_font\_04b\_24n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_04b\_24r[] U8G\_FONT\_SECTION("u8g\_font\_04b\_24r");

/\* orgdot fonts \*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_orgv01[] U8G\_FONT\_SECTION("u8g\_font\_orgv01");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_orgv01r[] U8G\_FONT\_SECTION("u8g\_font\_orgv01r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_orgv01n[] U8G\_FONT\_SECTION("u8g\_font\_orgv01n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fixed\_v0[] U8G\_FONT\_SECTION("u8g\_font\_fixed\_v0");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fixed\_v0r[] U8G\_FONT\_SECTION("u8g\_font\_fixed\_v0r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_fixed\_v0n[] U8G\_FONT\_SECTION("u8g\_font\_fixed\_v0n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_tpssb[] U8G\_FONT\_SECTION("u8g\_font\_tpssb");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_tpssbr[] U8G\_FONT\_SECTION("u8g\_font\_tpssbr");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_tpssbn[] U8G\_FONT\_SECTION("u8g\_font\_tpssbn");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_tpss[] U8G\_FONT\_SECTION("u8g\_font\_tpss");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_tpssr[] U8G\_FONT\_SECTION("u8g\_font\_tpssr");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_tpssn[] U8G\_FONT\_SECTION("u8g\_font\_tpssn");

/\* contributed \*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_freedoomr25n[] U8G\_FONT\_SECTION("u8g\_font\_freedoomr25n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_freedoomr10r[] U8G\_FONT\_SECTION("u8g\_font\_freedoomr10r");

/\* adobe X11 \*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB08[] U8G\_FONT\_SECTION("u8g\_font\_courB08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB08r[] U8G\_FONT\_SECTION("u8g\_font\_courB08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB10[] U8G\_FONT\_SECTION("u8g\_font\_courB10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB10r[] U8G\_FONT\_SECTION("u8g\_font\_courB10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB12[] U8G\_FONT\_SECTION("u8g\_font\_courB12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB12r[] U8G\_FONT\_SECTION("u8g\_font\_courB12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB14[] U8G\_FONT\_SECTION("u8g\_font\_courB14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB14r[] U8G\_FONT\_SECTION("u8g\_font\_courB14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB18[] U8G\_FONT\_SECTION("u8g\_font\_courB18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB18r[] U8G\_FONT\_SECTION("u8g\_font\_courB18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB24[] U8G\_FONT\_SECTION("u8g\_font\_courB24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB24r[] U8G\_FONT\_SECTION("u8g\_font\_courB24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courB24n[] U8G\_FONT\_SECTION("u8g\_font\_courB24n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR08[] U8G\_FONT\_SECTION("u8g\_font\_courR08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR08r[] U8G\_FONT\_SECTION("u8g\_font\_courR08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR10[] U8G\_FONT\_SECTION("u8g\_font\_courR10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR10r[] U8G\_FONT\_SECTION("u8g\_font\_courR10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR12[] U8G\_FONT\_SECTION("u8g\_font\_courR12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR12r[] U8G\_FONT\_SECTION("u8g\_font\_courR12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR14[] U8G\_FONT\_SECTION("u8g\_font\_courR14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR14r[] U8G\_FONT\_SECTION("u8g\_font\_courR14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR18[] U8G\_FONT\_SECTION("u8g\_font\_courR18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR18r[] U8G\_FONT\_SECTION("u8g\_font\_courR18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR24[] U8G\_FONT\_SECTION("u8g\_font\_courR24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR24r[] U8G\_FONT\_SECTION("u8g\_font\_courR24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_courR24n[] U8G\_FONT\_SECTION("u8g\_font\_courR24n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB08[] U8G\_FONT\_SECTION("u8g\_font\_helvB08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB08r[] U8G\_FONT\_SECTION("u8g\_font\_helvB08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB08n[] U8G\_FONT\_SECTION("u8g\_font\_helvB08n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB10[] U8G\_FONT\_SECTION("u8g\_font\_helvB10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB10r[] U8G\_FONT\_SECTION("u8g\_font\_helvB10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB10n[] U8G\_FONT\_SECTION("u8g\_font\_helvB10n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB12[] U8G\_FONT\_SECTION("u8g\_font\_helvB12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB12r[] U8G\_FONT\_SECTION("u8g\_font\_helvB12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB12n[] U8G\_FONT\_SECTION("u8g\_font\_helvB12n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB14[] U8G\_FONT\_SECTION("u8g\_font\_helvB14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB14r[] U8G\_FONT\_SECTION("u8g\_font\_helvB14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB14n[] U8G\_FONT\_SECTION("u8g\_font\_helvB14n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB18[] U8G\_FONT\_SECTION("u8g\_font\_helvB18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB18r[] U8G\_FONT\_SECTION("u8g\_font\_helvB18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB18n[] U8G\_FONT\_SECTION("u8g\_font\_helvB18n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB24[] U8G\_FONT\_SECTION("u8g\_font\_helvB24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB24r[] U8G\_FONT\_SECTION("u8g\_font\_helvB24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvB24n[] U8G\_FONT\_SECTION("u8g\_font\_helvB24n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR08[] U8G\_FONT\_SECTION("u8g\_font\_helvR08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR08r[] U8G\_FONT\_SECTION("u8g\_font\_helvR08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR08n[] U8G\_FONT\_SECTION("u8g\_font\_helvR08n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR10[] U8G\_FONT\_SECTION("u8g\_font\_helvR10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR10r[] U8G\_FONT\_SECTION("u8g\_font\_helvR10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR10n[] U8G\_FONT\_SECTION("u8g\_font\_helvR10n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR12[] U8G\_FONT\_SECTION("u8g\_font\_helvR12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR12r[] U8G\_FONT\_SECTION("u8g\_font\_helvR12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR12n[] U8G\_FONT\_SECTION("u8g\_font\_helvR12n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR14[] U8G\_FONT\_SECTION("u8g\_font\_helvR14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR14r[] U8G\_FONT\_SECTION("u8g\_font\_helvR14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR14n[] U8G\_FONT\_SECTION("u8g\_font\_helvR14n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR18[] U8G\_FONT\_SECTION("u8g\_font\_helvR18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR18r[] U8G\_FONT\_SECTION("u8g\_font\_helvR18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR18n[] U8G\_FONT\_SECTION("u8g\_font\_helvR18n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR24[] U8G\_FONT\_SECTION("u8g\_font\_helvR24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR24r[] U8G\_FONT\_SECTION("u8g\_font\_helvR24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_helvR24n[] U8G\_FONT\_SECTION("u8g\_font\_helvR24n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB08[] U8G\_FONT\_SECTION("u8g\_font\_ncenB08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB08r[] U8G\_FONT\_SECTION("u8g\_font\_ncenB08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB10[] U8G\_FONT\_SECTION("u8g\_font\_ncenB10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB10r[] U8G\_FONT\_SECTION("u8g\_font\_ncenB10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB12[] U8G\_FONT\_SECTION("u8g\_font\_ncenB12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB12r[] U8G\_FONT\_SECTION("u8g\_font\_ncenB12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB14[] U8G\_FONT\_SECTION("u8g\_font\_ncenB14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB14r[] U8G\_FONT\_SECTION("u8g\_font\_ncenB14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB18[] U8G\_FONT\_SECTION("u8g\_font\_ncenB18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB18r[] U8G\_FONT\_SECTION("u8g\_font\_ncenB18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB24[] U8G\_FONT\_SECTION("u8g\_font\_ncenB24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB24r[] U8G\_FONT\_SECTION("u8g\_font\_ncenB24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenB24n[] U8G\_FONT\_SECTION("u8g\_font\_ncenB24n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR08[] U8G\_FONT\_SECTION("u8g\_font\_ncenR08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR08r[] U8G\_FONT\_SECTION("u8g\_font\_ncenR08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR10[] U8G\_FONT\_SECTION("u8g\_font\_ncenR10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR10r[] U8G\_FONT\_SECTION("u8g\_font\_ncenR10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR12[] U8G\_FONT\_SECTION("u8g\_font\_ncenR12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR12r[] U8G\_FONT\_SECTION("u8g\_font\_ncenR12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR14[] U8G\_FONT\_SECTION("u8g\_font\_ncenR14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR14r[] U8G\_FONT\_SECTION("u8g\_font\_ncenR14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR18[] U8G\_FONT\_SECTION("u8g\_font\_ncenR18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR18r[] U8G\_FONT\_SECTION("u8g\_font\_ncenR18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR24[] U8G\_FONT\_SECTION("u8g\_font\_ncenR24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR24r[] U8G\_FONT\_SECTION("u8g\_font\_ncenR24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_ncenR24n[] U8G\_FONT\_SECTION("u8g\_font\_ncenR24n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb08[] U8G\_FONT\_SECTION("u8g\_font\_symb08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb08r[] U8G\_FONT\_SECTION("u8g\_font\_symb08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb10[] U8G\_FONT\_SECTION("u8g\_font\_symb10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb10r[] U8G\_FONT\_SECTION("u8g\_font\_symb10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb12[] U8G\_FONT\_SECTION("u8g\_font\_symb12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb12r[] U8G\_FONT\_SECTION("u8g\_font\_symb12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb14[] U8G\_FONT\_SECTION("u8g\_font\_symb14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb14r[] U8G\_FONT\_SECTION("u8g\_font\_symb14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb18[] U8G\_FONT\_SECTION("u8g\_font\_symb18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb18r[] U8G\_FONT\_SECTION("u8g\_font\_symb18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb24[] U8G\_FONT\_SECTION("u8g\_font\_symb24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_symb24r[] U8G\_FONT\_SECTION("u8g\_font\_symb24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB08[] U8G\_FONT\_SECTION("u8g\_font\_timB08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB08r[] U8G\_FONT\_SECTION("u8g\_font\_timB08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB10[] U8G\_FONT\_SECTION("u8g\_font\_timB10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB10r[] U8G\_FONT\_SECTION("u8g\_font\_timB10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB12[] U8G\_FONT\_SECTION("u8g\_font\_timB12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB12r[] U8G\_FONT\_SECTION("u8g\_font\_timB12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB14[] U8G\_FONT\_SECTION("u8g\_font\_timB14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB14r[] U8G\_FONT\_SECTION("u8g\_font\_timB14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB18[] U8G\_FONT\_SECTION("u8g\_font\_timB18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB18r[] U8G\_FONT\_SECTION("u8g\_font\_timB18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB24[] U8G\_FONT\_SECTION("u8g\_font\_timB24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB24r[] U8G\_FONT\_SECTION("u8g\_font\_timB24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timB24n[] U8G\_FONT\_SECTION("u8g\_font\_timB24n");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR08[] U8G\_FONT\_SECTION("u8g\_font\_timR08");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR08r[] U8G\_FONT\_SECTION("u8g\_font\_timR08r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR10[] U8G\_FONT\_SECTION("u8g\_font\_timR10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR10r[] U8G\_FONT\_SECTION("u8g\_font\_timR10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR12[] U8G\_FONT\_SECTION("u8g\_font\_timR12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR12r[] U8G\_FONT\_SECTION("u8g\_font\_timR12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR14[] U8G\_FONT\_SECTION("u8g\_font\_timR14");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR14r[] U8G\_FONT\_SECTION("u8g\_font\_timR14r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR18[] U8G\_FONT\_SECTION("u8g\_font\_timR18");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR18r[] U8G\_FONT\_SECTION("u8g\_font\_timR18r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR24[] U8G\_FONT\_SECTION("u8g\_font\_timR24");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR24r[] U8G\_FONT\_SECTION("u8g\_font\_timR24r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_timR24n[] U8G\_FONT\_SECTION("u8g\_font\_timR24n");

/\* fontstruct \*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_p01type[] U8G\_FONT\_SECTION("u8g\_font\_p01type");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_p01typer[] U8G\_FONT\_SECTION("u8g\_font\_p01typer");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_p01typen[] U8G\_FONT\_SECTION("u8g\_font\_p01typen");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_lucasfont\_alternate[] U8G\_FONT\_SECTION("u8g\_font\_lucasfont\_alternate");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_lucasfont\_alternater[] U8G\_FONT\_SECTION("u8g\_font\_lucasfont\_alternater");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_lucasfont\_alternaten[] U8G\_FONT\_SECTION("u8g\_font\_lucasfont\_alternaten");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_chikita[] U8G\_FONT\_SECTION("u8g\_font\_chikita");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_chikitar[] U8G\_FONT\_SECTION("u8g\_font\_chikitar");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_chikitan[] U8G\_FONT\_SECTION("u8g\_font\_chikitan");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_pixelle\_micro[] U8G\_FONT\_SECTION("u8g\_font\_pixelle\_micro");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_pixelle\_micror[] U8G\_FONT\_SECTION("u8g\_font\_pixelle\_micror");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_pixelle\_micron[] U8G\_FONT\_SECTION("u8g\_font\_pixelle\_micron");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_trixel\_square[] U8G\_FONT\_SECTION("u8g\_font\_trixel\_square");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_trixel\_squarer[] U8G\_FONT\_SECTION("u8g\_font\_trixel\_squarer");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_trixel\_squaren[] U8G\_FONT\_SECTION("u8g\_font\_trixel\_squaren");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_robot\_de\_niro[] U8G\_FONT\_SECTION("u8g\_font\_robot\_de\_niro");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_robot\_de\_niror[] U8G\_FONT\_SECTION("u8g\_font\_robot\_de\_niror");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_robot\_de\_niron[] U8G\_FONT\_SECTION("u8g\_font\_robot\_de\_niron");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_baby[] U8G\_FONT\_SECTION("u8g\_font\_baby");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_babyr[] U8G\_FONT\_SECTION("u8g\_font\_babyr");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_babyn[] U8G\_FONT\_SECTION("u8g\_font\_babyn");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_blipfest\_07[] U8G\_FONT\_SECTION("u8g\_font\_blipfest\_07");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_blipfest\_07r[] U8G\_FONT\_SECTION("u8g\_font\_blipfest\_07r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_blipfest\_07n[] U8G\_FONT\_SECTION("u8g\_font\_blipfest\_07n");

/\* profont \*/

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont10[] U8G\_FONT\_SECTION("u8g\_font\_profont10");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont10r[] U8G\_FONT\_SECTION("u8g\_font\_profont10r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont11[] U8G\_FONT\_SECTION("u8g\_font\_profont11");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont11r[] U8G\_FONT\_SECTION("u8g\_font\_profont11r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont12[] U8G\_FONT\_SECTION("u8g\_font\_profont12");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont12r[] U8G\_FONT\_SECTION("u8g\_font\_profont12r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont15[] U8G\_FONT\_SECTION("u8g\_font\_profont15");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont15r[] U8G\_FONT\_SECTION("u8g\_font\_profont15r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont17[] U8G\_FONT\_SECTION("u8g\_font\_profont17");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont17r[] U8G\_FONT\_SECTION("u8g\_font\_profont17r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont22[] U8G\_FONT\_SECTION("u8g\_font\_profont22");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont22r[] U8G\_FONT\_SECTION("u8g\_font\_profont22r");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont29[] U8G\_FONT\_SECTION("u8g\_font\_profont29");

extern const u8g\_fntpgm\_uint8\_t u8g\_font\_profont29r[] U8G\_FONT\_SECTION("u8g\_font\_profont29r");

#ifdef \_\_cplusplus

}

#endif

#endif /\* \_U8G\_H \*/