AVR libraries

Generated by Doxygen 1.8.13

Contents

1	Maiı	n Page			1
2	Mod	lule Inde	ех		3
	2.1	Module	9 s		3
3	File	Index			5
	3.1	File Lis	st		5
4	Mod	lule Doc	cumentatio	on	7
	4.1	LCD lil	orary < lcd.	h>	7
		4.1.1	Detailed	Description	9
		4.1.2	Macro De	efinition Documentation	10
			4.1.2.1	LCD_CONTROLLER_KS0073	10
			4.1.2.2	LCD_DELAY_BOOTUP	10
			4.1.2.3	LCD_DELAY_BUSY_FLAG	10
			4.1.2.4	LCD_DELAY_ENABLE_PULSE	10
			4.1.2.5	LCD_DELAY_INIT	10
			4.1.2.6	LCD_DELAY_INIT_4BIT	10
			4.1.2.7	LCD_DELAY_INIT_REP	10
			4.1.2.8	LCD_IO_MODE	11
			4.1.2.9	LCD_LINE_LENGTH	11
			4.1.2.10	LCD_RW_PIN	11
			4.1.2.11	LCD_RW_PORT	11
			4.1.2.12	LCD_START_LINE1	11
			41213	LCD START LINES	11

ii CONTENTS

		4.1.2.14	LCD_START_LINE3	11
		4.1.2.15	LCD_START_LINE4	11
		4.1.2.16	LCD_WRAP_LINES	12
	4.1.3	Function	Documentation	12
		4.1.3.1	lcd_clrscr()	12
		4.1.3.2	lcd_command()	12
		4.1.3.3	lcd_data()	12
		4.1.3.4	lcd_gotoxy()	13
		4.1.3.5	lcd_home()	13
		4.1.3.6	lcd_init()	13
		4.1.3.7	lcd_putc()	14
		4.1.3.8	lcd_puts()	14
		4.1.3.9	lcd_puts_p()	15
4.2	UART	Library <u< td=""><td>uart.h></td><td>16</td></u<>	uart.h>	16
	4.2.1	Detailed	Description	17
	4.2.2	Macro De	efinition Documentation	17
		4.2.2.1	UART_BAUD_SELECT	17
		4.2.2.2	UART_BAUD_SELECT_DOUBLE_SPEED	18
		4.2.2.3	UART_RX_BUFFER_SIZE	18
		4.2.2.4	UART_TX_BUFFER_SIZE	18
	4.2.3	Function	Documentation	18
		4.2.3.1	uart1_getc()	18
		4.2.3.2	uart1_init()	19
		4.2.3.3	uart1_putc()	19
		4.2.3.4	uart1_puts()	19
		4.2.3.5	uart1_puts_p()	19
		4.2.3.6	uart_getc()	20
		4.2.3.7	uart_init()	20
		4.2.3.8	uart_putc()	20
		4.2.3.9	uart_puts()	21
		4.2.3.10	uart puts p()	21
			→ → V	-

CONTENTS

5	File	Docum	entation		23
	5.1	/home/	/fryza/GIT/[Digital-electronics-2/Examples/library/include/common.h File Reference	23
		5.1.1	Detailed [Description	23
		5.1.2	Macro De	finition Documentation	23
			5.1.2.1	DDR	23
			5.1.2.2	PIN	23
	5.2	/home/	/fryza/GIT/[Digital-electronics-2/Examples/library/include/lcd.h File Reference	24
	5.3	/home/	/fryza/GIT/[Digital-electronics-2/Examples/library/include/twi.h File Reference	26
		5.3.1	Detailed [Description	26
		5.3.2	Macro De	finition Documentation	27
			5.3.2.1	F_SCL	27
			5.3.2.2	TWI_BIT_RATE_REG	27
			5.3.2.3	TWI_PORT	27
			5.3.2.4	TWI_READ	27
			5.3.2.5	TWI_SCL_PIN	27
			5.3.2.6	TWI_SDA_PIN	27
			5.3.2.7	TWI_WRITE	28
		5.3.3	Function	Documentation	28
			5.3.3.1	twi_init()	28
			5.3.3.2	twi_read_ack()	28
			5.3.3.3	twi_read_nack()	28
			5.3.3.4	twi_start()	28
			5.3.3.5	twi_stop()	29
			5.3.3.6	twi_write()	29
	5.4	/home/	/fryza/GIT/[Digital-electronics-2/Examples/library/include/uart.h File Reference	29
Inc	lex				31

Chapter 1

Main Page

Collection of libraries for AVR-GCC

Author

Peter Fleury pfleury@gmx.ch http://tinyurl.com/peterfleury

Copyright

(C) 2015 Peter Fleury, GNU General Public License Version 3

2 Main Page

Chapter 2

Module Index

2.1	M	0	d	ш	les
4 . I	IVI	u	u	u	

Here	is a	list	ot al	I mod	IIIIes

LCD library <lcd.h></lcd.h>	7
UART Library <uart.h></uart.h>	16

4 Module Index

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

/home/fryza/GIT/Digital-electronics-2/Examples/library/include/common.h	23
/home/fryza/GIT/Digital-electronics-2/Examples/library/include/lcd.h	24
/home/fryza/GIT/Digital-electronics-2/Examples/library/include/ Icd_definitions.h	??
/home/fryza/GIT/Digital-electronics-2/Examples/library/include/twi.h	26
/home/fryza/GIT/Digital-electronics-2/Examples/library/include/uart.h	29

6 File Index

Chapter 4

Module Documentation

4.1 LCD library < lcd.h>

Basic routines for interfacing a HD44780U-based character LCD display.

Definition for LCD controller type

Use 0 for HD44780 controller, change to 1 for displays with KS0073 controller.

• #define LCD_CONTROLLER_KS0073 0

Definitions for Display Size

Change these definitions to adapt setting to your display

These definitions can be defined in a separate include file <code>lcd_definitions.h</code> instead modifying this file by adding - <code>D_LCD_DEFINITIONS_FILE</code> to the CDEFS section in the Makefile. All definitions added to the file <code>lcd_definitions.h</code> will override the default definitions from <code>lcd.h</code>

- #define LCD LINE LENGTH 0x40
- #define LCD START LINE1 0x00
- #define LCD START LINE2 0x40
- #define LCD START LINE3 0x14
- #define LCD START LINE4 0x54
- #define LCD_WRAP_LINES 0

Definitions for 4-bit IO mode

The four LCD data lines and the three control lines RS, RW, E can be on the same port or on different ports. Change LCD_RS_PORT, LCD_RW_PORT, LCD_E_PORT if you want the control lines on different ports.

Normally the four data lines should be mapped to bit 0..3 on one port, but it is possible to connect these data lines in different order or even on different ports by adapting the LCD_DATAX_PORT and LCD_DATAX_PIN definitions.

Adjust these definitions to your target.

These definitions can be defined in a separate include file lcd_definitions.h instead modifying this file by adding - D_LCD_DEFINITIONS_FILE to the CDEFS section in the Makefile. All definitions added to the file lcd_definitions.h will override the default definitions from lcd_h

- #define LCD IO MODE 1
- #define LCD_RW_PORT LCD_PORT
- #define LCD_RW_PIN 5

Definitions of delays

Used to calculate delay timers. Adapt the F_CPU define in the Makefile to the clock frequency in Hz of your target

These delay times can be adjusted, if some displays require different delays.

These definitions can be defined in a separate include file lcd_definitions.h instead modifying this file by adding - D_LCD_DEFINITIONS_FILE to the CDEFS section in the Makefile. All definitions added to the file lcd_definitions.h will override the default definitions from lcd_h

- #define LCD_DELAY_BOOTUP 16000
- #define LCD_DELAY_INIT 5000
- #define LCD_DELAY_INIT_REP 64
- #define LCD DELAY INIT 4BIT 64
- #define LCD DELAY BUSY FLAG 4
- #define LCD DELAY ENABLE PULSE 1

Definitions for LCD command instructions

The constants define the various LCD controller instructions which can be passed to the function lcd_command(), see HD44780 data sheet for a complete description.

- #define LCD_CLR 0 /* DB0: clear display */
- #define LCD_HOME 1 /* DB1: return to home position */
- #define LCD_ENTRY_MODE 2 /* DB2: set entry mode */
- #define LCD ENTRY INC 1 /* DB1: 1=increment, 0=decrement */
- #define LCD_ENTRY_SHIFT 0 /* DB2: 1=display shift on */
- #define LCD_ON 3 /* DB3: turn lcd/cursor on */
- #define LCD ON DISPLAY 2 /* DB2: turn display on */
- #define LCD_ON_CURSOR 1 /* DB1: turn cursor on */
- #define LCD ON BLINK 0 /* DB0: blinking cursor ? */
- #define LCD_MOVE 4 /* DB4: move cursor/display */
- #define LCD_MOVE_DISP 3 /* DB3: move display (0-> cursor) ? */
- #define LCD_MOVE_RIGHT 2 /* DB2: move right (0-> left) ? */
- #define LCD_FUNCTION 5 /* DB5: function set */
- #define LCD FUNCTION 8BIT 4 /* DB4: set 8BIT mode (0->4BIT mode) */
- #define LCD FUNCTION 2LINES 3 /* DB3: two lines (0->one line) */
- #define LCD FUNCTION 10DOTS 2 /* DB2: 5x10 font (0->5x7 font) */
- #define LCD_CGRAM 6 /* DB6: set CG RAM address */
- #define LCD_DDRAM 7 /* DB7: set DD RAM address */
- #define LCD_BUSY 7 /* DB7: LCD is busy */
- #define LCD_ENTRY_DEC 0x04 /* display shift off, dec cursor move dir */
- #define LCD_ENTRY_DEC_SHIFT 0x05 /* display shift on, dec cursor move dir */
- #define LCD ENTRY INC 0x06 /* display shift off, inc cursor move dir */
- #define LCD_ENTRY_INC_SHIFT 0x07 /* display shift on, inc cursor move dir */
- #define LCD_DISP_OFF 0x08 /* display off */
- #define LCD_DISP_ON 0x0C /* display on, cursor off */
- #define LCD_DISP_ON_BLINK 0x0D /* display on, cursor off, blink char */
- #define LCD_DISP_ON_CURSOR 0x0E /* display on, cursor on */
- #define LCD_DISP_ON_CURSOR_BLINK 0x0F /* display on, cursor on, blink char */
- #define LCD MOVE CURSOR LEFT 0x10 /* move cursor left (decrement) */
- #define LCD_MOVE_CURSOR_RIGHT 0x14 /* move cursor right (increment) */
- #define LCD_MOVE_DISP_LEFT 0x18 /* shift display left */
- #define LCD MOVE DISP_RIGHT 0x1C /* shift display right */
- #define LCD_FUNCTION_4BIT_1LINE 0x20 /* 4-bit interface, single line, 5x7 dots */
- #define LCD_FUNCTION_4BIT_2LINES 0x28 /* 4-bit interface, dual line, 5x7 dots */
- #define LCD FUNCTION 8BIT 1LINE 0x30 /* 8-bit interface, single line, 5x7 dots */
- #define LCD_FUNCTION_8BIT_2LINES 0x38 /* 8-bit interface, dual line, 5x7 dots */
- #define LCD_MODE_DEFAULT ((1 << LCD_ENTRY_MODE) | (1 << LCD_ENTRY_INC))

Functions

void lcd init (uint8 t dispAttr)

Initialize display and select type of cursor.

void lcd_clrscr (void)

Clear display and set cursor to home position.

void lcd home (void)

Set cursor to home position.

void lcd_gotoxy (uint8_t x, uint8_t y)

Set cursor to specified position.

• void lcd putc (char c)

Display character at current cursor position.

void lcd_puts (const char *s)

Display string without auto linefeed.

void lcd_puts_p (const char *progmem_s)

Display string from program memory without auto linefeed.

void lcd_command (uint8_t cmd)

Send LCD controller instruction command.

• void lcd data (uint8 t data)

Send data byte to LCD controller.

#define lcd_puts_P(__s) lcd_puts_p(PSTR(__s))

macros for automatically storing string constant in program memory

4.1.1 Detailed Description

Basic routines for interfacing a HD44780U-based character LCD display.

```
#include <lcd.h>
```

LCD character displays can be found in many devices, like espresso machines, laser printers. The Hitachi HD44780 controller and its compatible controllers like Samsung KS0066U have become an industry standard for these types of displays.

This library allows easy interfacing with a HD44780 compatible display and can be operated in memory mapped mode (LCD_IO_MODE defined as 0 in the include file lcd.h.) or in 4-bit IO port mode (LCD_IO_MODE defined as 1). 8-bit IO port mode is not supported.

Memory mapped mode is compatible with old Kanda STK200 starter kit, but also supports generation of R/W signal through A8 address line.

See also

The chapter $Interfacing \ a \ HD44780 \ Based \ LCD \ to \ an \ AVR \ on \ my \ home \ page, which shows example circuits how to connect an LCD to an AVR controller.$

Author

Peter Fleury pfleury@gmx.ch http://tinyurl.com/peterfleury

Version

2.0

Copyright

(C) 2015 Peter Fleury, GNU General Public License Version 3

4.1.2 Macro Definition Documentation

```
4.1.2.1 LCD_CONTROLLER_KS0073
```

```
#define LCD_CONTROLLER_KS0073 0
```

Use 0 for HD44780 controller, 1 for KS0073 controller

4.1.2.2 LCD_DELAY_BOOTUP

```
#define LCD_DELAY_BOOTUP 16000
```

delay in micro seconds after power-on

4.1.2.3 LCD_DELAY_BUSY_FLAG

```
#define LCD_DELAY_BUSY_FLAG 4
```

time in micro seconds the address counter is updated after busy flag is cleared

4.1.2.4 LCD_DELAY_ENABLE_PULSE

```
#define LCD_DELAY_ENABLE_PULSE 1
```

enable signal pulse width in micro seconds

4.1.2.5 LCD_DELAY_INIT

```
#define LCD_DELAY_INIT 5000
```

delay in micro seconds after initialization command sent

4.1.2.6 LCD_DELAY_INIT_4BIT

```
#define LCD_DELAY_INIT_4BIT 64
```

delay in micro seconds after setting 4-bit mode

4.1.2.7 LCD_DELAY_INIT_REP

```
#define LCD_DELAY_INIT_REP 64
```

delay in micro seconds after initialization command repeated

4.1.2.8 LCD_IO_MODE

#define LCD_IO_MODE 1

0: memory mapped mode, 1: IO port mode

4.1.2.9 LCD_LINE_LENGTH

#define LCD_LINE_LENGTH 0x40

internal line length of the display

4.1.2.10 LCD_RW_PIN

#define LCD_RW_PIN 5

pin for RW line

4.1.2.11 LCD_RW_PORT

#define LCD_RW_PORT LCD_PORT

port for RW line

4.1.2.12 LCD_START_LINE1

#define LCD_START_LINE1 0x00

DDRAM address of first char of line 1

4.1.2.13 LCD_START_LINE2

#define LCD_START_LINE2 0x40

DDRAM address of first char of line 2

4.1.2.14 LCD_START_LINE3

#define LCD_START_LINE3 0x14

DDRAM address of first char of line 3

4.1.2.15 LCD_START_LINE4

#define LCD_START_LINE4 0x54

DDRAM address of first char of line 4

```
4.1.2.16 LCD_WRAP_LINES
```

```
#define LCD_WRAP_LINES 0
```

0: no wrap, 1: wrap at end of visibile line

4.1.3 Function Documentation

```
4.1.3.1 lcd_clrscr()
```

```
void lcd_clrscr (
    void )
```

Clear display and set cursor to home position.

Returns

none

4.1.3.2 lcd_command()

Send LCD controller instruction command.

Parameters

cmd instruction to send to LCD controller, see HD44780 data sheet

Returns

none

```
4.1.3.3 lcd_data()
```

Send data byte to LCD controller.

Similar to Icd_putc(), but without interpreting LF

Parameters

data byte to send to LCD controller, see HD44780 data shee
--

Returns

none

4.1.3.4 lcd_gotoxy()

Set cursor to specified position.

Parameters

X	horizontal position (0: left most position)
У	vertical position (0: first line)

Returns

none

4.1.3.5 lcd_home()

```
void lcd_home (
     void )
```

Set cursor to home position.

Returns

none

4.1.3.6 lcd_init()

Initialize display and select type of cursor.

Parameters

dispAttr	LCD_DISP_OFF display off
	LCD_DISP_ON display on, cursor off
	LCD_DISP_ON_CURSOR display on, cursor on
	LCD_DISP_ON_CURSOR_BLINK display on, cursor on flashing

Returns

none

```
4.1.3.7 lcd_putc()
```

```
void lcd_putc ( {\tt char}\ c\ )
```

Display character at current cursor position.

Parameters

c character to be displayed

Returns

none

```
4.1.3.8 lcd_puts()
```

```
void lcd_puts ( {\tt const\ char\ *\ s\ )}
```

Display string without auto linefeed.

Parameters

s string to be displayed

Returns

none

4.1.3.9 lcd_puts_p()

Display string from program memory without auto linefeed.

Parameters

progmem←	string from program memory be be displayed
S	

Returns

none

See also

lcd_puts_P

4.2 UART Library < uart.h>

Interrupt UART library using the built-in UART with transmit and receive circular buffers.

Macros

#define UART_BAUD_SELECT(baudRate, xtalCpu) (((xtalCpu) + 8UL * (baudRate)) / (16UL * (baudRate))
 - 1UL)

UART Baudrate Expression.

#define UART_BAUD_SELECT_DOUBLE_SPEED(baudRate, xtalCpu) (((((xtalCpu) + 4UL * (baudRate)) / (8UL * (baudRate)) - 1UL)) | 0x8000)

UART Baudrate Expression for ATmega double speed mode.

• #define UART RX BUFFER SIZE 32

Size of the circular receive buffer, must be power of 2.

#define UART TX BUFFER SIZE 32

Size of the circular transmit buffer, must be power of 2.

#define UART_FRAME_ERROR 0x1000

Framing Error by UART.

#define UART_OVERRUN_ERROR 0x0800

Overrun condition by UART.

#define UART_PARITY_ERROR 0x0400

Parity Error by UART.

#define UART BUFFER OVERFLOW 0x0200

receive ringbuffer overflow

#define UART NO DATA 0x0100

no receive data available

#define uart_puts_P(__s) uart_puts_p(PSTR(__s))

Macro to automatically put a string constant into program memory.

#define uart1_puts_P(_s) uart1_puts_p(PSTR(_s))

Macro to automatically put a string constant into program memory.

Functions

void uart_init (unsigned int baudrate)

Initialize UART and set baudrate.

unsigned int uart_getc (void)

Get received byte from ringbuffer.

void uart_putc (unsigned char data)

Put byte to ringbuffer for transmitting via UART.

void uart_puts (const char *s)

Put string to ringbuffer for transmitting via UART.

void uart puts p (const char *s)

Put string from program memory to ringbuffer for transmitting via UART.

void uart1 init (unsigned int baudrate)

Initialize USART1 (only available on selected ATmegas)

• unsigned int uart1_getc (void)

Get received byte of USART1 from ringbuffer. (only available on selected ATmega)

void uart1_putc (unsigned char data)

Put byte to ringbuffer for transmitting via USART1 (only available on selected ATmega)

void uart1_puts (const char *s)

Put string to ringbuffer for transmitting via USART1 (only available on selected ATmega)

void uart1 puts p (const char *s)

Put string from program memory to ringbuffer for transmitting via USART1 (only available on selected ATmega)

4.2.1 Detailed Description

Interrupt UART library using the built-in UART with transmit and receive circular buffers.

```
#include <uart.h>
```

This library can be used to transmit and receive data through the built in UART.

An interrupt is generated when the UART has finished transmitting or receiving a byte. The interrupt handling routines use circular buffers for buffering received and transmitted data.

The UART_RX_BUFFER_SIZE and UART_TX_BUFFER_SIZE constants define the size of the circular buffers in bytes. Note that these constants must be a power of 2. You may need to adapt these constants to your target and your application by adding CDEFS += -DUART_RX_BUFFER_SIZE=nn -DUART_TX_BUFFER_SIZE=nn to your Makefile.

Note

Based on Atmel Application Note AVR306

Author

```
Peter Fleury pfleury@gmx.ch http://tinyurl.com/peterfleury
```

Copyright

(C) 2015 Peter Fleury, GNU General Public License Version 3

4.2.2 Macro Definition Documentation

4.2.2.1 UART_BAUD_SELECT

UART Baudrate Expression.

Parameters

xtalCpu	system clock in Mhz, e.g. 4000000UL for 4Mhz
baudRate	baudrate in bps, e.g. 1200, 2400, 9600

4.2.2.2 UART_BAUD_SELECT_DOUBLE_SPEED

UART Baudrate Expression for ATmega double speed mode.

Parameters

xtalCpu	system clock in Mhz, e.g. 4000000UL for 4Mhz
baudRate	baudrate in bps, e.g. 1200, 2400, 9600

4.2.2.3 UART_RX_BUFFER_SIZE

```
#define UART_RX_BUFFER_SIZE 32
```

Size of the circular receive buffer, must be power of 2.

You may need to adapt this constant to your target and your application by adding CDEFS += -DUART_RX_BUF← FER_SIZE=nn to your Makefile.

4.2.2.4 UART_TX_BUFFER_SIZE

```
#define UART_TX_BUFFER_SIZE 32
```

Size of the circular transmit buffer, must be power of 2.

You may need to adapt this constant to your target and your application by adding CDEFS += -DUART_TX_BUF \leftarrow FER_SIZE=nn to your Makefile.

4.2.3 Function Documentation

4.2.3.1 uart1_getc()

Get received byte of USART1 from ringbuffer. (only available on selected ATmega)

See also

uart_getc

```
4.2.3.2 uart1_init()
void uart1_init (
              unsigned int baudrate )
Initialize USART1 (only available on selected ATmegas)
See also
     uart_init
4.2.3.3 uart1_putc()
void uart1_putc (
              unsigned char data )
Put byte to ringbuffer for transmitting via USART1 (only available on selected ATmega)
See also
     uart_putc
4.2.3.4 uart1_puts()
void uart1_puts (
              const char * s )
Put string to ringbuffer for transmitting via USART1 (only available on selected ATmega)
See also
     uart_puts
4.2.3.5 uart1_puts_p()
void uart1_puts_p (
              const char *s)
Put string from program memory to ringbuffer for transmitting via USART1 (only available on selected ATmega)
See also
     uart_puts_p
```

4.2.3.6 uart_getc()

Get received byte from ringbuffer.

Returns in the lower byte the received character and in the higher byte the last receive error. UART_NO_DATA is returned when no data is available.

Returns

lower byte: received byte from ringbuffer

higher byte: last receive status

- 0 successfully received data from UART
- · UART NO DATA

no receive data available

UART BUFFER OVERFLOW

Receive ringbuffer overflow. We are not reading the receive buffer fast enough, one or more received character have been dropped

• UART_OVERRUN_ERROR

Overrun condition by UART. A character already present in the UART UDR register was not read by the interrupt handler before the next character arrived, one or more received characters have been dropped.

• UART_FRAME_ERROR

Framing Error by UART

4.2.3.7 uart_init()

Initialize UART and set baudrate.

Parameters

baudrate | Specify baudrate using macro UART_BAUD_SELECT()

Returns

none

4.2.3.8 uart_putc()

```
void uart_putc (
          unsigned char data )
```

Put byte to ringbuffer for transmitting via UART.

Parameters

data byte to be transmitted

Returns

none

4.2.3.9 uart_puts()

```
void uart_puts ( {\tt const\ char\ *\ s\ )}
```

Put string to ringbuffer for transmitting via UART.

The string is buffered by the uart library in a circular buffer and one character at a time is transmitted to the UART using interrupts. Blocks if it can not write the whole string into the circular buffer.

Parameters

```
s string to be transmitted
```

Returns

none

4.2.3.10 uart_puts_p()

Put string from program memory to ringbuffer for transmitting via UART.

The string is buffered by the uart library in a circular buffer and one character at a time is transmitted to the UART using interrupts. Blocks if it can not write the whole string into the circular buffer.

Parameters

s program memory string to be transmitted

Returns

none

See also

uart_puts_P

Chapter 5

File Documentation

5.1 /home/fryza/GIT/Digital-electronics-2/Examples/library/include/common.h File Reference

Macros

- #define DDR(_x) (*(&_x 1))#define PIN(_x) (*(&_x 2))
- 5.1.1 Detailed Description

Common functions and defines.

Copyright

(c) 2019 Tomas Fryza Dept. of Radio Electronics, Brno University of Technology, Czechia This work is licensed under the terms of the MIT license.

5.1.2 Macro Definition Documentation

5.1.2.1 DDR

```
#define DDR( \_x \ ) \ (*(\&\_x \ - \ 1))
```

Define address of Data Direction Register of port _x.

5.1.2.2 PIN

Define address of input register of port _x.

24 File Documentation

5.2 /home/fryza/GIT/Digital-electronics-2/Examples/library/include/lcd.h File Reference

```
#include <inttypes.h>
#include <avr/pgmspace.h>
#include "lcd_definitions.h"
```

Macros

Definition for LCD controller type

Use 0 for HD44780 controller, change to 1 for displays with KS0073 controller.

• #define LCD CONTROLLER KS0073 0

Definitions for Display Size

Change these definitions to adapt setting to your display

These definitions can be defined in a separate include file **lcd_definitions.h** instead modifying this file by adding -D_LCD_DEFINITIONS_FILE to the CDEFS section in the Makefile. All definitions added to the file **lcd_← definitions.h** will override the default definitions from **lcd.h**

- #define LCD_LINE_LENGTH 0x40
- #define LCD START LINE1 0x00
- #define LCD_START_LINE2 0x40
- #define LCD_START_LINE3 0x14
- #define LCD_START_LINE4 0x54
- #define LCD_WRAP_LINES 0

Definitions for 4-bit IO mode

The four LCD data lines and the three control lines RS, RW, E can be on the same port or on different ports. Change LCD_RS_PORT, LCD_RW_PORT, LCD_E_PORT if you want the control lines on different ports.

Normally the four data lines should be mapped to bit 0..3 on one port, but it is possible to connect these data lines in different order or even on different ports by adapting the LCD_DATAx_PORT and LCD_DATAx_PIN definitions.

Adjust these definitions to your target.

These definitions can be defined in a separate include file Icd_definitions.h instead modifying this file by adding -D_LCD_DEFINITIONS_FILE to the CDEFS section in the Makefile. All definitions added to the file Icd_definitions.h will override the default definitions from Icd_h

- #define LCD_IO_MODE 1
- #define LCD RW PORT LCD PORT
- #define LCD_RW_PIN 5

Definitions of delays

Used to calculate delay timers. Adapt the F_CPU define in the Makefile to the clock frequency in Hz of your target

These delay times can be adjusted, if some displays require different delays.

These definitions can be defined in a separate include file Icd_definitions.h instead modifying this file by adding -D_LCD_DEFINITIONS_FILE to the CDEFS section in the Makefile. All definitions added to the file Icd_definitions.h will override the default definitions from Icd_h

- #define LCD_DELAY_BOOTUP 16000
- #define LCD_DELAY_INIT 5000
- #define LCD_DELAY_INIT_REP 64
- #define LCD_DELAY_INIT_4BIT 64

```
#define LCD_DELAY_BUSY_FLAG 4#define LCD_DELAY_ENABLE_PULSE 1
```

Definitions for LCD command instructions

The constants define the various LCD controller instructions which can be passed to the function lcd_← command(), see HD44780 data sheet for a complete description.

```
    #define LCD_CLR 0 /* DB0: clear display */

    #define LCD HOME 1 /* DB1: return to home position */

• #define LCD_ENTRY_MODE 2 /* DB2: set entry mode */
• #define LCD_ENTRY_INC 1 /* DB1: 1=increment, 0=decrement */
• #define LCD_ENTRY_SHIFT 0 /* DB2: 1=display shift on */
• #define LCD_ON 3 /* DB3: turn lcd/cursor on */

    #define LCD_ON_DISPLAY 2 /* DB2: turn display on */

• #define LCD_ON_CURSOR 1 /* DB1: turn cursor on */
• #define LCD_ON_BLINK 0 /* DB0: blinking cursor ? */

    #define LCD_MOVE 4 /* DB4: move cursor/display */

• #define LCD_MOVE_DISP 3 /* DB3: move display (0-> cursor) ? */
• #define LCD MOVE RIGHT 2 /* DB2: move right (0-> left) ? */
• #define LCD FUNCTION 5 /* DB5: function set */
• #define LCD_FUNCTION_8BIT 4 /* DB4: set 8BIT mode (0->4BIT mode) */
• #define LCD FUNCTION 2LINES 3 /* DB3: two lines (0->one line) */

    #define LCD_FUNCTION_10DOTS 2 /* DB2: 5x10 font (0->5x7 font) */

    #define LCD_CGRAM 6 /* DB6: set CG RAM address */

    #define LCD DDRAM 7 /* DB7: set DD RAM address */

• #define LCD_BUSY 7 /* DB7: LCD is busy */
• #define LCD_ENTRY_DEC 0x04 /* display shift off, dec cursor move dir */
• #define LCD_ENTRY_DEC_SHIFT 0x05 /* display shift on, dec cursor move dir */

    #define LCD_ENTRY_INC_0x06 /* display shift off, inc cursor move dir */
    #define LCD_ENTRY_INC_SHIFT 0x07 /* display shift on, inc cursor move dir */

• #define LCD_DISP_OFF 0x08 /* display off */
• #define LCD_DISP_ON 0x0C /* display on, cursor off */
• #define LCD_DISP_ON_BLINK 0x0D /* display on, cursor off, blink char */
• #define LCD_DISP_ON_CURSOR 0x0E /* display on, cursor on */
• #define LCD_DISP_ON_CURSOR_BLINK 0x0F /* display on, cursor on, blink char */
• #define LCD_MOVE_CURSOR_LEFT 0x10 /* move cursor left (decrement) */
• #define LCD_MOVE_CURSOR_RIGHT 0x14 /* move cursor right (increment) */
• #define LCD_MOVE_DISP_LEFT 0x18 /* shift display left */
• #define LCD MOVE DISP_RIGHT 0x1C /* shift display right */
• #define LCD FUNCTION 4BIT 1LINE 0x20 /* 4-bit interface, single line, 5x7 dots */
• #define LCD FUNCTION 4BIT 2LINES 0x28 /* 4-bit interface, dual line, 5x7 dots */
• #define LCD_FUNCTION_8BIT_1LINE 0x30 /* 8-bit interface, single line, 5x7 dots */
• #define LCD_FUNCTION_8BIT_2LINES 0x38 /* 8-bit interface, dual line, 5x7 dots */

    #define LCD_MODE_DEFAULT ((1 << LCD_ENTRY_MODE) | (1 << LCD_ENTRY_INC) )</li>
```

Functions

```
#define lcd_puts_P(_s) lcd_puts_p(PSTR(_s))

macros for automatically storing string constant in program memory
void lcd_init (uint8_t dispAttr)

Initialize display and select type of cursor.
void lcd_clrscr (void)

Clear display and set cursor to home position.
void lcd_home (void)

Set cursor to home position.
void lcd_gotoxy (uint8 t x, uint8 t y)
```

• void lcd putc (char c)

Set cursor to specified position.

26 File Documentation

Display character at current cursor position.

void lcd_puts (const char *s)

Display string without auto linefeed.

• void lcd_puts_p (const char *progmem_s)

Display string from program memory without auto linefeed.

void lcd_command (uint8_t cmd)

Send LCD controller instruction command.

void lcd_data (uint8_t data)

Send data byte to LCD controller.

5.3 /home/fryza/GIT/Digital-electronics-2/Examples/library/include/twi.h File Reference

```
#include <avr/io.h>
```

Macros

- #define TWI PORT PORTC
- #define TWI_SDA_PIN 4
- #define TWI SCL PIN 5
- #define F_SCL 50000
- #define TWI_BIT_RATE_REG ((F_CPU/F_SCL 16) / 2)
- #define TWI_READ 1
- #define TWI_WRITE 0

Functions

- void twi_init (void)
- uint8_t twi_start (uint8_t slave_address)
- void twi_write (uint8_t data)
- uint8 t twi read ack (void)
- · uint8 t twi read nack (void)
- void twi_stop (void)

5.3.1 Detailed Description

TWI library for AVR-GCC.

The library defines functions for the TWI (I2C) communication between AVR and slave device(s). Functions use TWI module of AVR.

Note

Based on Microchip Atmel ATmega16 and ATmega328P manuals.

Copyright

(c) 2018-2019 Tomas Fryza Dept. of Radio Electronics, Brno University of Technology, Czechia This work is licensed under the terms of the MIT license.

5.3.2 Macro Definition Documentation

```
5.3.2.1 F_SCL
#define F_SCL 50000
TWI bit rate.
Warning
     Must be greater than 31000 kbps.
5.3.2.2 TWI_BIT_RATE_REG
#define TWI_BIT_RATE_REG ((F_CPU/F_SCL - 16) / 2)
TWI bit rate register value.
5.3.2.3 TWI_PORT
#define TWI_PORT PORTC
Port of TWI hardware unit.
5.3.2.4 TWI_READ
#define TWI_READ 1
Data direction for reading from TWI device.
5.3.2.5 TWI_SCL_PIN
#define TWI_SCL_PIN 5
SCL pin of TWI hardware unit.
5.3.2.6 TWI_SDA_PIN
#define TWI_SDA_PIN 4
```

Generated by Doxygen

SDA pin of TWI hardware unit.

28 File Documentation

5.3.2.7 TWI_WRITE

```
#define TWI_WRITE 0
```

Data direction for writing to TWI device.

5.3.3 Function Documentation

Initialize TWI, enable internal pull-up resistors, and set SCL frequency.

Implementation notes:

- AVR internal pull-up resistors at pins TWI_SDA_PIN and TWI_SCL_PIN are enabled
- TWI bit rate register value is calculated as follows fscl = fcpu/(16 + 2*TWBR)

```
5.3.3.2 twi_read_ack()
```

Read one byte from TWI slave device, followed by ACK.

Returns

Received data.

```
5.3.3.3 twi_read_nack()
```

Read one byte from TWI slave device, followed by NACK.

Returns

Received data.

```
5.3.3.4 twi_start()
```

Start communication on TWI bus and send address of TWI slave.

Parameters

slave_address	Address and transfer direction of TWI slave.
---------------	--

Return values

	0	- Slave device accessible.
ĺ	1	- Failed to access slave device.

Note

Function returns 0 only if 0x18 or 0x40 status code is detected. 0x18: SLA+W has been transmitted and ACK has been received. 0x40: SLA+R has been transmitted and ACK has been received.

5.3.3.5 twi_stop()

```
void twi_stop (
     void )
```

Generates stop condition on TWI bus.

5.3.3.6 twi_write()

Send one byte to TWI slave device.

Parameters

data Byte to be transmitted.

5.4 /home/fryza/GIT/Digital-electronics-2/Examples/library/include/uart.h File Reference

```
#include <avr/pgmspace.h>
```

Macros

#define UART_BAUD_SELECT(baudRate, xtalCpu) (((xtalCpu) + 8UL * (baudRate)) / (16UL * (baudRate))
 - 1UL)

UART Baudrate Expression.

30 File Documentation

#define UART_BAUD_SELECT_DOUBLE_SPEED(baudRate, xtalCpu) (((((xtalCpu) + 4UL * (baudRate)) / (8UL * (baudRate)) - 1UL)) | 0x8000)

UART Baudrate Expression for ATmega double speed mode.

#define UART RX BUFFER SIZE 32

Size of the circular receive buffer, must be power of 2.

• #define UART_TX_BUFFER_SIZE 32

Size of the circular transmit buffer, must be power of 2.

• #define UART FRAME ERROR 0x1000

Framing Error by UART.

• #define UART_OVERRUN_ERROR 0x0800

Overrun condition by UART.

#define UART_PARITY_ERROR 0x0400

Parity Error by UART.

• #define UART_BUFFER_OVERFLOW 0x0200

receive ringbuffer overflow

• #define UART_NO_DATA 0x0100

no receive data available

#define uart puts P(s) uart puts p(PSTR(s))

Macro to automatically put a string constant into program memory.

#define uart1_puts_P(_s) uart1_puts_p(PSTR(_s))

Macro to automatically put a string constant into program memory.

Functions

void uart init (unsigned int baudrate)

Initialize UART and set baudrate.

unsigned int uart_getc (void)

Get received byte from ringbuffer.

void uart_putc (unsigned char data)

Put byte to ringbuffer for transmitting via UART.

void uart_puts (const char *s)

Put string to ringbuffer for transmitting via UART.

void uart_puts_p (const char *s)

Put string from program memory to ringbuffer for transmitting via UART.

void uart1_init (unsigned int baudrate)

Initialize USART1 (only available on selected ATmegas)

• unsigned int uart1_getc (void)

Get received byte of USART1 from ringbuffer. (only available on selected ATmega)

• void uart1_putc (unsigned char data)

Put byte to ringbuffer for transmitting via USART1 (only available on selected ATmega)

void uart1_puts (const char *s)

Put string to ringbuffer for transmitting via USART1 (only available on selected ATmega)

void uart1_puts_p (const char *s)

Put string from program memory to ringbuffer for transmitting via USART1 (only available on selected ATmega)

Index

```
/home/fryza/GIT/Digital-electronics-2/Examples/library/include/cb@tintibrary < lcd.h > , 10
                                                    LCD DELAY ENABLE PULSE
/home/fryza/GIT/Digital-electronics-2/Examples/library/include/ldcD library < lcd.h > , 10
                                                    LCD_DELAY_INIT_4BIT
        h, 24
/home/fryza/GIT/Digital-electronics-2/Examples/library/include/twCD library < lcd.h >, 10
                                                    LCD_DELAY_INIT_REP
/home/fryza/GIT/Digital-electronics-2/Examples/library/include/ula@D-library < lcd.h >, 10
                                                    LCD_DELAY_INIT
        h, 29
                                                        LCD library <lcd.h>, 10
common.h
                                                    LCD IO MODE
    DDR, 23
                                                        LCD library <lcd.h>, 10
    PIN, 23
                                                    LCD_LINE_LENGTH
                                                        LCD library <lcd.h>, 11
DDR
                                                    LCD RW PIN
    common.h, 23
                                                        LCD library <lcd.h>, 11
                                                    LCD_RW_PORT
F SCL
                                                        LCD library <lcd.h>, 11
    twi.h, 27
                                                    LCD_START_LINE1
                                                        LCD library <lcd.h>, 11
LCD library <lcd.h>, 7
                                                    LCD START LINE2
    LCD_CONTROLLER_KS0073, 10
                                                        LCD library <lcd.h>, 11
    LCD_DELAY_BOOTUP, 10
                                                    LCD_START_LINE3
    LCD DELAY BUSY FLAG, 10
                                                        LCD library <lcd.h>, 11
    LCD_DELAY_ENABLE_PULSE, 10
                                                    LCD_START_LINE4
    LCD_DELAY_INIT_4BIT, 10
                                                        LCD library <lcd.h>, 11
    LCD DELAY INIT REP, 10
                                                    LCD_WRAP_LINES
    LCD_DELAY_INIT, 10
                                                        LCD library <lcd.h>, 11
    LCD_IO_MODE, 10
                                                    lcd clrscr
    LCD_LINE_LENGTH, 11
                                                        LCD library <lcd.h>, 12
    LCD_RW_PIN, 11
                                                    lcd command
    LCD_RW_PORT, 11
                                                        LCD library <lcd.h>, 12
    LCD_START_LINE1, 11
                                                    Icd data
    LCD_START_LINE2, 11
                                                        LCD library <lcd.h>, 12
    LCD START LINE3, 11
                                                    lcd_gotoxy
    LCD_START_LINE4, 11
                                                        LCD library <lcd.h>, 13
    LCD_WRAP_LINES, 11
                                                    lcd home
    lcd clrscr, 12
                                                        LCD library <lcd.h>, 13
    lcd_command, 12
                                                    lcd_init
    lcd_data, 12
                                                        LCD library <lcd.h>, 13
    lcd_gotoxy, 13
                                                    lcd putc
    lcd_home, 13
                                                        LCD library < lcd.h>, 14
    Icd_init, 13
                                                    lcd puts
    lcd putc, 14
                                                        LCD library <lcd.h>, 14
    lcd puts, 14
                                                    lcd puts p
    lcd_puts_p, 14
                                                        LCD library <lcd.h>, 14
LCD CONTROLLER KS0073
    LCD library <lcd.h>, 10
                                                    PIN
LCD DELAY BOOTUP
                                                        common.h, 23
    LCD library <lcd.h>, 10
LCD_DELAY_BUSY_FLAG
                                                    TWI_BIT_RATE_REG
```

32 INDEX

twi.h, 27	UART TX BUFFER SIZE
TWI PORT	UART Library <uart.h>, 18</uart.h>
 twi.h, 27	uart1 getc
TWI_READ	UART Library <uart.h>, 18</uart.h>
twi.h, 27	uart1_init
TWI_SCL_PIN	UART Library <uart.h>, 18</uart.h>
twi.h, 27	uart1_putc
TWI_SDA_PIN	UART Library <uart.h>, 19</uart.h>
twi.h, 27	uart1_puts
TWI_WRITE	UART Library <uart.h>, 19</uart.h>
twi.h, 27	uart1_puts_p
twi.h	UART Library <uart.h>, 19</uart.h>
F_SCL, 27	uart_getc
TWI_BIT_RATE_REG, 27	UART Library <uart.h>, 19</uart.h>
TWI_PORT, 27	uart_init
TWI_READ, 27	UART Library <uart.h>, 20</uart.h>
TWI_SCL_PIN, 27	uart_putc
TWI_SDA_PIN, 27	UART Library <uart.h>, 20</uart.h>
TWI_WRITE, 27	uart_puts
twi_init, 28	UART Library <uart.h>, 21</uart.h>
twi_read_ack, 28	uart_puts_p
twi_read_nack, 28	UART Library <uart.h>, 21</uart.h>
twi_start, 28	
twi_stop, 29 twi_write, 29	
twi init	
twi.h, 28	
twi_read_ack	
twi.h, 28	
twi_read_nack	
twi.h, 28	
twi start	
_ twi.h, 28	
twi_stop	
twi.h, 29	
twi_write	
twi.h, 29	
HADTI'I	
UART Library <uart.h>, 16</uart.h>	
UART_BAUD_SELECT_DOUBLE_SPEED, 17	
UART_BAUD_SELECT, 17	
UART_RX_BUFFER_SIZE, 18	
UART_TX_BUFFER_SIZE, 18 uart1_getc, 18	
uart1_jetc, 18	
uart1_putc, 19	
uart1_puts, 19	
uart1_puts_p, 19	
uart_getc, 19	
uart_init, 20	
uart_putc, 20	
uart_puts, 21	
uart_puts_p, 21	
UART_BAUD_SELECT_DOUBLE_SPEED	
UART Library <uart.h>, 17</uart.h>	
UART_BAUD_SELECT	
UART Library <uart.h>, 17</uart.h>	
UART_RX_BUFFER_SIZE	
UART Library <uart.h>, 18</uart.h>	