AVR-ATMEGA32A LCD16x8 DRIVER DOCUMENTATION
DRIVER #2

Written on ATMEL STUDIO 7.0
Edited on VSCODE 1.54.3
Generated by Doxygen 1.9.1

Chapter 1

File Index

1.1 File List

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

Chapter 2

File Documentation

2.1 HAL_LCD_interface.h File Reference

This is the h file that is used for our macros, function prototypes and declaration used in our LCD16x8 Driver. !PLEASE DON'T FORGET TO USE DELAYS!

Macros

- #define **HAL_LCD_ROW00** (0x80)
- #define **HAL_LCD_ROW01** (0xC0)
- #define **HAL_LCD_COL00** (0)
- #define **HAL_LCD_COL01** (1)
- #define **HAL_LCD_COL02** (2)
- #define **HAL_LCD_COL03** (3)
- #define **HAL_LCD_COL04** (4)
- #define **HAL_LCD_COL05** (5)
- #define **HAL_LCD_COL06** (6)
- #define **HAL_LCD_COL07** (7)
- #define **HAL_LCD_COL08** (8)
- #define **HAL_LCD_COL09** (9)
- #define **HAL_LCD_COL10** (10)
- #define **HAL_LCD_COL11** (11)
- #define **HAL_LCD_COL12** (12)
- #define **HAL_LCD_COL13** (13)
- #define **HAL_LCD_COL14** (14)
- #define **HAL_LCD_COL15** (15)

Functions

- void HAL_LCD_init (void)

  HAL_LCD_init is the LCD16x8 Initializing function.
- void HAL_LCD_displayCharacter (u8_t au8_charData)

  HAL_LCD_displayCharacter is a function that displays the character passed as a parameter on the LCD16x8.
- void HAL_LCD_displayString (u8_t ∗pu8_srtData)

  HAL_LCD_displayString is a function that displays a given string on the LCD16x8 screen (!Be careful of character overflows!)
- void HAL_LCD_putAtLoc (u8_t au8_row, u8_t au8_col)

  HAL_LCD_putAtLoc is a function that moves the cursor of the LCD16x8 to the DDRAM address passed.
- void HAL_LCD_clearDisplay (void)

  HAL_LCD_clearDisplay is a function that clears the display of the LCD16x8.

2.1.1 Detailed Description

This is the h file that is used for our macros, function prototypes and declaration used in our LCD16x8 Driver. !PLEASE DON'T FORGET TO USE DELAYS!

Author

Mohamed El Barbary (mohmbarbary@gmail.com)

Version

1.0

Date

29-01-2021 10:19:20 PM

Copyright

Copyright GPL(c) 2021

2.1.2 Function Documentation

2.1.2.1 HAL_LCD_displayCharacter()

```
void HAL_LCD_displayCharacter (
            u8_t au8_charData )
```

HAL_LCD_displayCharacter is a function that displays the character passed as a parameter on the LCD16x8.

Parameters

au8_charData	is the data that will be passed to the LCD16x8.

2.1.2.2 HAL_LCD_displayString()

```
void HAL_LCD_displayString (
            u8_t * pu8_srtData )
```

HAL_LCD_displayString is a function that displays a given string on the LCD16x8 screen (!Be careful of character overflows!)

Parameters

pu8_srtData	is a pointer to the 8 bits of dat that will be displayed.

2.1.2.3 HAL_LCD_init()

```
void HAL_LCD_init (
            void  )
```

HAL_LCD_init is the LCD16x8 Initializing function.

SUPER IMPORTANT!//

2.1.2.4 HAL_LCD_putAtLoc()

```
void HAL_LCD_putAtLoc (
            u8_t au8_row,
            u8_t au8_col )
```

HAL_LCD_putAtLoc is a function that moves the cursor of the LCD16x8 to the DDRAM address passed.

Parameters

au8_row	is the variable containing the row where we want to move (!DO NOT FORGET TO USE THE MACROS!)
au8_col	is the variable containing the column where we want to move (!DO NOT FORGET TO USE THE MACROS!)

2.2 HAL_LCD_private.h File Reference

This is the h file that is used to store the private variables and declarations of our LCD16x8 Driver. !PLEASE DON'T FORGET TO USE DELAYS!

Macros

- #define **HAL_LCD_CTRL_PORT** (PORTB)
- #define **HAL_LCD_RS_PIN** (PIN1)
- #define **HAL_LCD_RW_PIN** (PIN2)
- #define **HAL_LCD_EN_PIN** (PIN3)
- #define **HAL_LCD_DATA_PORT** (PORTA)
- #define **HAL_LCD_D0_PIN** (PIN0)
- #define **HAL_LCD_D1_PIN** (PIN1)
- #define **HAL_LCD_D2_PIN** (PIN2)
- #define **HAL_LCD_D3_PIN** (PIN3)
- #define **HAL_LCD_D4_PIN** (PIN4)
- #define **HAL_LCD_D5_PIN** (PIN5)
- #define **HAL_LCD_D6_PIN** (PIN6)
- #define **HAL_LCD_D7_PIN** (PIN7)

2.2.1 Detailed Description

This is the h file that is used to store the private variables and declarations of our LCD16x8 Driver. !PLEASE DON'T FORGET TO USE DELAYS!

Author

Mohamed El Barbary (mohmbarbary@gmail.com)

Version

1.0

Date

29-01-2021 10:19:20 PM

Copyright

Copyright GPL(c) 2021

2.3 HAL_LCD_program.c File Reference

HAL_LCD_program.c is the file that contains the implementation for the function prototypes found in the HAL_LCD_interface.h file.

```
#include "LSTD_BITMATH.h"
#include "LSTD_TYPES.h"
#include "MCAL_GPIO_interface.h"
#include "HAL_LCD_private.h"
#include "HAL_LCD_interface.h"
#include "util/delay.h"
```

Macros

- #define **F_CPU** 16000000UL

Functions

- void HAL_LCD_init (void)

  HAL_LCD_init is the LCD16x8 Initializing function.
- void HAL_LCD_displayCharacter (u8_t au8_charData)

  HAL_LCD_displayCharacter is a function that displays the character passed as a parameter on the LCD16x8.
- void HAL_LCD_displayString (u8_t ∗pu8_srtData)

  HAL_LCD_displayString is a function that displays a given string on the LCD16x8 screen (!Be careful of character overflows!)
- void HAL_LCD_putAtLoc (u8_t au8_row, u8_t au8_col)

  HAL_LCD_putAtLoc is a function that moves the cursor of the LCD16x8 to the DDRAM address passed.
- void HAL_LCD_clearDisplay (void)

  HAL_LCD_clearDisplay is a function that clears the display of the LCD16x8.

2.3.1 Detailed Description

HAL_LCD_program.c is the file that contains the implementation for the function prototypes found in the HAL_LCD_interface.h file.

Author

Mohamed El Barbary (mohmbarbary@gmail.com)

Version

1.0

Date

25-03-2021 10:19:20 PM

Copyright

Copyright GPL(c) 2021

2.3.2 Function Documentation

2.3.2.1 HAL_LCD_displayCharacter()

```
void HAL_LCD_displayCharacter (
            u8_t au8_charData )
```

HAL_LCD_displayCharacter is a function that displays the character passed as a parameter on the LCD16x8.

Parameters

au8_charData	is the data that will be passed to the LCD16x8.

2.3.2.2 HAL_LCD_displayString()

```
void HAL_LCD_displayString (
            u8_t * pu8_srtData )
```

HAL_LCD_displayString is a function that displays a given string on the LCD16x8 screen (!Be careful of character overflows!)

Parameters

pu8_srtData	is a pointer to the 8 bits of dat that will be displayed.

2.3.2.3 HAL_LCD_init()

```
void HAL_LCD_init (
            void  )
```

HAL_LCD_init is the LCD16x8 Initializing function.

SUPER IMPORTANT!//

2.3.2.4 HAL_LCD_putAtLoc()

```
void HAL_LCD_putAtLoc (
            u8_t au8_row,
            u8_t au8_col )
```

HAL_LCD_putAtLoc is a function that moves the cursor of the LCD16x8 to the DDRAM address passed.

Parameters

au8_row	is the variable containing the row where we want to move (!DO NOT FORGET TO USE THE MACROS!)
au8_col	is the variable containing the column where we want to move (!DO NOT FORGET TO USE THE MACROS!)

2.4 LSTD_BITMATH.h File Reference

This is a standard library layer file that contains bitmath macros that can come in handy while coding.

Macros

- #define **setBit**(REG, POS) (REG |= (1 << POS))
- #define **clearBit**(REG, POS) (REG &= ~(1 << POS))
- #define **toggleBit**(REG, POS) (REG ^= (1 << POS))
- #define **getBit**(REG, POS) ((REG >> POS) & 1)

2.4.1 Detailed Description

This is a standard library layer file that contains bitmath macros that can come in handy while coding.

Author

Mohamed El Barbary (Mohmbarbary@gmail.com)

Version

1.0

Date

2021-01-29 10:19:20 PM

Copyright

2.5 LSTD_TYPES.h File Reference

This is a standard library layer file that is used to make aliases for the standard data types inorder to make the code more portable and to avoid changes in data type sizes when using different compilers. giving our standard data types new aliases: unsigned char and signed char -> u8_t and s8_t. unsigned short int and signed short int -> u16_t and s16_t. unsigned long int and signed long int -> u32_t and s32_t. float -> f32_t. double -> f64_t.

Typedefs

- typedef unsigned char **u8_t**
- typedef signed char **s8_t**
- typedef unsigned short int **u16_t**
- typedef signed short int **s16_t**
- typedef unsigned long int **u32_t**
- typedef signed long int **s32_t**
- typedef float **f32_t**
- typedef double **f64_t**

2.5.1 Detailed Description

This is a standard library layer file that is used to make aliases for the standard data types inorder to make the code more portable and to avoid changes in data type sizes when using different compilers. giving our standard data types new aliases: unsigned char and signed char -> u8_t and s8_t. unsigned short int and signed short int -> u16_t and s16_t. unsigned long int and signed long int -> u32_t and s32_t. float -> f32_t. double -> f64_t.

Author

Mohamed El Barbary (Mohmbarbary@gmail.com)

Version

1.0

Date

2021-01-29 10:19:20 PM

Copyright

Copyright GPL(c) 2021

2.6 main.c File Reference

This is the main function that is used to test the functionality of our LCD16x8 Driver. !PLEASE DON'T FORGET TO USE DELAYS!

```
#include "LSTD_TYPES.h"
#include "LSTD_BITMATH.h"
#include "HAL_LCD_interface.h"
#include "util/delay.h"
```

Macros

- #define **F_CPU** 16000000UL
- #define **PUSHB0** PIN0
- #define **PUSHB1** PIN4
- #define **PUSHB2** PIN2

Functions

- int **main** (void)

2.6.1 Detailed Description

This is the main function that is used to test the functionality of our LCD16x8 Driver. !PLEASE DON'T FORGET TO USE DELAYS!

Author

Mohamed El Barbary (mohmbarbary@gmail.com)

Version

1.0

Date

5-02-2021 9:57:56 PM

Copyright

Copyright GPL(c) 2021

2.7 MCAL_GPIO_interface.h File Reference

This .h file contains the interfacing macros, declarations and function prototypes for the GPIO Driver.

Macros

- #define **PORTA** (0)
- #define **PORTB** (1)
- #define **PORTC** (2)
- #define **PORTD** (3)
- #define PIN0 (0b00000001)

  creating macros for the PIN registers, we will be writing them in binary, so that we can do bit operations on them for ease of use.
- #define **PIN1** (0b00000010)
- #define **PIN2** (0b00000100)
- #define **PIN3** (0b00001000)
- #define **PIN4** (0b00010000)
- #define **PIN5** (0b00100000)
- #define **PIN6** (0b01000000)
- #define **PIN7** (0b10000000)
- #define INPUT_FLOAT (0)

  creating a macro for the data direction types.
- #define **INPUT_PULLUP** (1)
- #define **OUTPUT** (2)
- #define LOW (0)

  creating a macro for the possible states.
- #define **HIGH** (1)

Functions

- void MCAL_GPIO_PinMode (u8_t au8_port, u8_t au8_pin, u8_t au8_type)

  MCAL_GPIO_PinMode is used to change the Mode of a pin/pins from any given port.
- void MCAL_GPIO_PinState (u8_t au8_port, u8_t au8_pin, u8_t au8_state)

  MCAL_GPIO_PinState is used to change the State of a pin/pins from any given port to HIGH or LOW.
- void MCAL_GPIO_TogglePin (u8_t au8_port, u8_t au8_pin)

  MCAL_GPIO_TogglePin is used to toggel the State of a pin given a port.
- u8_t MCAL_GPIO_GetPinState (u8_t au8_port, u8_t au8_pin)

  MCAL_GPIO_GetPinState is a function that gets the state of a gien PORT and PIN combination.

2.7.1 Detailed Description

This .h file contains the interfacing macros, declarations and function prototypes for the GPIO Driver.

Author

Mohamed El Barbary (Mohmbarbary@gmail.com)

Version

1.0

Date

29-01-2021 10:19:20 PM

Copyright

2.7.2 Function Documentation

2.7.2.1 MCAL_GPIO_GetPinState()

```
u8_t MCAL_GPIO_GetPinState (
            u8_t au8_port,
            u8_t au8_pin )
```

MCAL_GPIO_GetPinState is a function that gets the state of a gien PORT and PIN combination.

Parameters

au8_port	the given PORT from our macros list.
au8_pin	the given PIN from our macros list.

Returns

u8_t returns true if the state is HIGH and false if the state is LOW.

2.7.2.2 MCAL_GPIO_PinMode()

```
void MCAL_GPIO_PinMode (
            u8_t au8_port,
            u8_t au8_pin,
            u8_t au8_type )
```

MCAL_GPIO_PinMode is used to change the Mode of a pin/pins from any given port.

Parameters

au8_port	is the port to be selected from our macro list PORTA, PORTB, PORTC or PORTD.
au8_pin	is the port to be selected from our macro list PIN1 ... PIN7.
au8_type	is the mode selected from our macro list INPUT_FLOAT, INPUT_PULLUP or OUTPUT.

We will switch over the au_8port given and once we find it, we will switch over the au8_type and then we set up our mode.

The registers used in order to alter the I/P or O/P modes. MCAL_DDRx, MCAL_PORTx.

2.7.2.3 MCAL_GPIO_PinState()

```
void MCAL_GPIO_PinState (
            u8_t au8_port,
```

```
            u8_t au8_pin,
            u8_t au8_state )
```

MCAL_GPIO_PinState is used to change the State of a pin/pins from any given port to HIGH or LOW.

Parameters

au8_port	is the port to be selected from our macro list PORTA, PORTB, PORTC or PORTD.
au8_pin	is the port to be selected from our macro list PIN1 ... PIN7.
au8_state	is the mode selected from our macro list HIGH or LOW.

We will switch over the au_8port given and once we find it, we will switch over the au8_type and then we set up our state.

The registers used in order to alter the I/P or O/P states. MCAL_PORTx.

2.7.2.4 MCAL_GPIO_TogglePin()

```
void MCAL_GPIO_TogglePin (
             u8_t au8_port,
             u8_t au8_pin )
```

MCAL_GPIO_TogglePin is used to toggel the State of a pin given a port.

Parameters

au8_port	The PORT used in the toggling operation.
au8_pin	The PIN to be toggled.

2.8 MCAL_GPIO_private.h File Reference

This .h file contains the private macros and declarations for the GPIO Driver.

Macros

- #define MCAL_PORTA (∗(volatile u8_t∗)(0x3B))

  Header guard for the .h file.
- #define **MCAL_DDRA** (∗(volatile u8_t∗)(0x3A))
- #define **MCAL_PINA** (∗(volatile u8_t∗)(0x39))
- #define MCAL_PORTB (∗(volatile u8_t∗)(0x38))

  defining the memory mapped addresses for the PORTB, DDRB, PINB Registers.
- #define **MCAL_DDRB** (∗(volatile u8_t∗)(0x37))
- #define **MCAL_PINB** (∗(volatile u8_t∗)(0x36))
- #define MCAL_PORTC (∗(volatile u8_t∗)(0x35))

  defining the memory mapped addresses for the PORTC, DDRC, PINC Registers.
- #define **MCAL_DDRC** (∗(volatile u8_t∗)(0x34))
- #define **MCAL_PINC** (∗(volatile u8_t∗)(0x33))
- #define MCAL_PORTD (∗(volatile u8_t∗)(0x32))

  defining the memory mapped addresses for the PORTD, DDRD, PIND Registers.
- #define **MCAL_DDRD** (∗(volatile u8_t∗)(0x31))
- #define **MCAL_PIND** (∗(volatile u8_t∗)(0x30))

2.8.1 Detailed Description

This .h file contains the private macros and declarations for the GPIO Driver.

Author

Mohamed El Barbary (Mohmbarbary@gmail.com)

Version

1.0

Date

29-01-2021 10:19:20 PM

Copyright

2.8.2 Macro Definition Documentation

2.8.2.1 MCAL_PORTA

```
#define MCAL_PORTA (*(volatile u8_t*)(0x3B))
```

Header guard for the .h file.

defining the memory mapped addresses for the PORTA, DDRA, PINA Registers.

2.9 MCAL_GPIO_program.c File Reference

This c file contains the implementation for the function prototypes used in MCAL_GPIO_interface.h.

```
#include "LSTD_BITMATH.h"
#include "LSTD_TYPES.h"
#include "MCAL_GPIO_private.h"
#include "MCAL_GPIO_interface.h"
```

Functions

- void MCAL_GPIO_PinMode (u8_t au8_port, u8_t au8_pin, u8_t au8_type)

  MCAL_GPIO_PinMode is used to change the Mode of a pin/pins from any given port.
- void MCAL_GPIO_PinState (u8_t au8_port, u8_t au8_pin, u8_t au8_state)

  MCAL_GPIO_PinState is used to change the State of a pin/pins from any given port to HIGH or LOW.
- void MCAL_GPIO_TogglePin (u8_t au8_port, u8_t au8_pin)

  MCAL_GPIO_TogglePin is used to toggel the State of a pin given a port.
- u8_t MCAL_GPIO_GetPinState (u8_t au8_port, u8_t au8_pin)

  MCAL_GPIO_GetPinState is a function that gets the state of a gien PORT and PIN combination.

2.9.1 Detailed Description

This c file contains the implementation for the function prototypes used in MCAL_GPIO_interface.h.

Author

Mohamed El Barbary (Mohmbarbary@gmail.com)

Version

1.0

Date

29-01-2021 10:19:20 PM

Copyright

2.9.2 Function Documentation

2.9.2.1 MCAL_GPIO_GetPinState()

```
u8_t MCAL_GPIO_GetPinState (
            u8_t au8_port,
            u8_t au8_pin )
```

MCAL_GPIO_GetPinState is a function that gets the state of a gien PORT and PIN combination.

Parameters

au8_port	the given PORT from our macros list.
au8_pin	the given PIN from our macros list.

Returns

u8_t returns true if the state is HIGH and false if the state is LOW.

2.9.2.2 MCAL_GPIO_PinMode()

```
void MCAL_GPIO_PinMode (
            u8_t au8_port,
```

```
            u8_t au8_pin,
            u8_t au8_type )
```

MCAL_GPIO_PinMode is used to change the Mode of a pin/pins from any given port.

Parameters

au8_port	is the port to be selected from our macro list PORTA, PORTB, PORTC or PORTD.
au8_pin	is the port to be selected from our macro list PIN1 ... PIN7.
au8_type	is the mode selected from our macro list INPUT_FLOAT, INPUT_PULLUP or OUTPUT.

We will switch over the au_8port given and once we find it, we will switch over the au8_type and then we set up our mode.

The registers used in order to alter the I/P or O/P modes. MCAL_DDRx, MCAL_PORTx.

2.9.2.3 MCAL_GPIO_PinState()

```
void MCAL_GPIO_PinState (
            u8_t au8_port,
            u8_t au8_pin,
            u8_t au8_state )
```

MCAL_GPIO_PinState is used to change the State of a pin/pins from any given port to HIGH or LOW.

Parameters

au8_port	is the port to be selected from our macro list PORTA, PORTB, PORTC or PORTD.
au8_pin	is the port to be selected from our macro list PIN1 ... PIN7.
au8_state	is the mode selected from our macro list HIGH or LOW.

We will switch over the au_8port given and once we find it, we will switch over the au8_type and then we set up our state.

The registers used in order to alter the I/P or O/P states. MCAL_PORTx.

2.9.2.4 MCAL_GPIO_TogglePin()

```
void MCAL_GPIO_TogglePin (
            u8_t au8_port,
            u8_t au8_pin )
```

MCAL_GPIO_TogglePin is used to toggel the State of a pin given a port.

Parameters

au8_port	The PORT used in the toggling operation.
au8_pin	The PIN to be toggled.

Index