HLib - An arm-based hardware library

Generated by Doxygen 1.8.5

Fri Sep 13 2013 13:05:49

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

Class Documentation

2.1 hd44780_c Class Reference

Public Member Functions

- hd44780_c ()

    Construction function. Do nothing.
- void Start (void)

    Initialize all HD44780 control pins, set default mode, and return home.
- void Clear (void)

    Send Clear Display command.
- void Home (void)

    Send Cursor Home command.
- void EntryMode (bool shift, bool increase)

    Send Entry Mode command.
- void OnOff (bool displayOn, bool cursorOn, bool cursorBlink)

    Send Display On Off command.
- void Goto (uint8_t charLine, uint8_t charCol)

    Set display cursor to a specified position.
- void Print (char printChar)

    Print one character to the LCD.
- void Print (char ∗printStr)

    Print one string to the LCD.
- void Print (uint32_t printNum, uint8_t radix)

    Print one unsigned number to the LCD.
- void Print (uint32_t printNum)

    Print one unsigned number to the LCD in decimal.
- void Print (int32_t printNum)

    Print one signed number to the LCD in decimal.

2.1.1 Constructor & Destructor Documentation

2.1.1.1 hd44780_c::hd44780_c (void)

Construction function. Do nothing.

Returns

None

2.1.2 Member Function Documentation

2.1.2.1 void hd44780_c::Clear (void)

Send Clear Display command.

Returns

None

2.1.2.2 void hd44780_c::EntryMode (bool *shift,* bool *increase*)

Send Entry Mode command.

Parameters

shift	TRUE the display will be shifted, FALSE the display will not be shifted
increase	TRUE increase cursor position, FALSE decrease cursor position

Returns

None

2.1.2.3 void hd44780_c::Goto (uint8_t *charLine,* uint8_t *charCol*)

Set display cursor to a specified position.

Parameters

charLine	LCD's line
charCol	LCD's column

Returns

None

2.1.2.4 void hd44780_c::Home (void)

Send Cursor Home command.

Returns

None

2.1.2.5 void hd44780_c::OnOff (bool *displayOn,* bool *cursorOn,* bool *cursorBlink*)

Send Display On Off command.

Parameters

displayOn	TRUE set the display on, FALSE set the display off

cursorOn	TRUE set the cursor on, FALSE set the cursor off
cursorBlink	TRUE the cursor is blinked, FALSE the cursor is not blinked

Returns

None

2.1.2.6 void hd44780_c::Print (char *printChar*)

Print one character to the LCD.

Parameters

printChar	Character to print

Returns

None

2.1.2.7 void hd44780_c::Print (char ∗ *printString*)

Print one string to the LCD.

Parameters

printString	String to print

Returns

None

2.1.2.8 void hd44780_c::Print (uint32_t *printNum,* uint8_t *radix*)

Print one unsigned number to the LCD.

Parameters

printNum	Unsigned number to print
radix	Valid values are 2, 8, 10, 16

Returns

None

2.1.2.9 void hd44780_c::Print (uint32_t *printNum*)

Print one unsigned number to the LCD in decimal.

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

printNum	Unsigned number to print

Returns

None

2.1.2.10 void hd44780_c::Print (int32_t *printNum*)

Print one signed number to the LCD in decimal.

Parameters

printNum	Unsigned number to print

Returns

None

2.1.2.11 void hd44780_c::Start (void)

Initialize all HD44780 control pins, set default mode, and return home.

Return values

None	

2.2 leds_c Class Reference

Public Member Functions

- leds_c (void)

  Construction. Enable clock, set output mode for LEDs' pins.
- void Set (uint8_t ledIndex, bool val)

  Set state of an LED.
- void On (uint8_t ledIndex)

  Turn on an LED.
- void Off (uint8_t ledIndex)

  Turn off an LED.
- void Toggle (uint8_t ledIndex)

  Toggle state of an LED.

2.2.1 Constructor & Destructor Documentation

2.2.1.1 leds_c::leds_c (void)

Construction. Enable clock, set output mode for LEDs' pins.

Returns

None

2.2.2 Member Function Documentation

2.2.2.1 void leds_c::Off (uint8_t *ledIndex*)

Turn off an LED.

Parameters

ledIndex	Index or the LED

Returns

None

2.2.2.2 void leds_c::On (uint8_t *ledIndex*)

Turn on an LED.

Parameters

ledIndex	Index or the LED

Returns

None

2.2.2.3 void leds_c::Set (uint8_t *ledIndex,* bool *val*)

Set state of an LED.

Parameters

ledIndex	Index or the LED
val	TRUE turn on the LED, FALSE turn off the LED

Returns

None

2.2.2.4 void leds_c::Toggle (uint8_t *ledIndex*)

Toggle state of an LED.

Parameters

ledIndex	Index or the LED

Returns

None

2.3 pins_c Class Reference

Public Member Functions

- pins_c (void)

  Construction function. Enable GPIO clocks.
- void Release (uint8_t pinIndex)

  Release one pin to input floating state.
- err_t SetMode (uint8_t pinIndex, pin_mode_t mode, pin_type_t type)

Set operation mode for one pin.
- void SetOutVal (uint8_t pinIndex, bool val)

  Set output register to a specified value.
- void SetOutOne (uint8_t pinIndex)

  Set output register to one.
- void SetOutZero (uint8_t pinIndex)

  Set output register to zero.
- bool GetInput (uint8_t pinIndex)

  Get digital electronic value at one pin.

2.3.1 Constructor & Destructor Documentation

2.3.1.1 pins_c::pins_c (void)

Construction function. Enable GPIO clocks.

Returns

None

2.3.2 Member Function Documentation

2.3.2.1 bool pins_c::GetInput (uint8_t *pinIndex*)

Get digital electronic value at one pin.

Parameters

pinIndex	Index of the pin

Return values

true	pin is 1
false	pin is 0

2.3.2.2 void pins_c::Release (uint8_t *pinIndex*)

Release one pin to input floating state.

Parameters

pinIndex	Index of the pin

Returns

None

2.3.2.3 err_t pins_c::SetMode (uint8_t *pinIndex*, pin_mode_t *mode*, pin_type_t *type*)

Set operation mode for one pin.

Parameters

pinIndex	Index of the pin
mode	Operation mode. Please refer the table pin map for valid configuration
type	Type of pin

Returns

HL_OK, HL_INVALID

Attention

Please select type corresponding with selecte mode. If you make a wrong pin configuration, your system may behave in unpredictable manner. Thereforce, we strongly recommend that you check return value of the function and make sure it is HL_OK

2.3.2.4 void pins_c::SetOutOne (uint8_t *pinIndex*)

Set output register to one.

Parameters

pinIndex	Index of the pin

Returns

None

Attention

The actual value on the pin is also depend on pinMode, pull-up/pull-down resistor. Please ensure the pin is set as Output mode and pull-up/pull-down resistor is configured appropriately

2.3.2.5 void pins_c::SetOutVal (uint8_t *pinIndex,* bool *val*)

Set output register to a specified value.

Parameters

pinIndex	Index of the pin
val	TRUE set one, FALSE set zero

Returns

None

Attention

The actual value on the pin is also depend on pinMode, pull-up/pull-down resistor. Please ensure the pin is set as Output mode and pull-up/pull-down resistor is configured appropriately

2.3.2.6 void pins_c::SetOutZero (uint8_t *pinIndex*)

Set output register to zero.

Parameters

pinIndex	Index of the pin

Returns

None

Attention

The actual value on the pin is also depend on pinMode, pull-up/pull-down resistor. Please ensure the pin is set as Output mode and pull-up/pull-down resistor is configured appropriately

2.4 port_pin_t Struct Reference

Public Attributes

- GPIO_TypeDef ∗ **port**
- uint16_t **pin**

2.5 uart_c Class Reference

Public Member Functions

- uart_c (uint8_t uartNum)

  Construction function.
- err_t Start (uint32_t baudRate, uint16_t stopBit)

  Start serial communicating.
- err_t Start (uint32_t baudRate)

  Start serial communicating in default mode with 8 data-bit, 1 stop-bit, no-parity.
- err_t Shutdown (void)

  Release I/O pins, stop UART clock.
- err_t Print (char outChar)

  Send one character.
- err_t Print (char ∗outStr)

  Send one string.
- err_t Print (uint32_t outNum, uint8_t radix)

  Convert one unsigned interger into string and then send it.
- err_t Print (uint32_t outNum)

  Convert one unsigned interger into string and then send it with default radix (decimal)
- err_t Print (int32_t outNum)

  Convert one signed interger into decimal string and then send it.
- err_t Out (uint8_t outNum)

  Send one raw 8-bit number.
- err_t Out (uint16_t outNum)

  Send one raw 16-bit number. The low-order part is sent first.
- err_t Out (uint32_t outNum)

  Send one raw 32-bit number. The low-order part is sent first.
- err_t Out (uint8_t outBuf[ ], uint32_t bufLen)

  Send one buffer.
- uint8_t Get (void)

  Get one received byte in receiving buffer.
- bool HasData (void)

  Check whether there is a new received byte in receiving buffer.

2.5.1 Constructor & Destructor Documentation

2.5.1.1 uart_c::uart_c (uint8_t *uartNum*)

Construction function.

Parameters

uartNum	UART will be used.

Returns

None

2.5.2 Member Function Documentation

2.5.2.1 uint8_t uart_c::Get (void)

Get one received byte in receiving buffer.

Returns

Received data

2.5.2.2 bool uart_c::HasData (void)

Check whether there is a new received byte in receiving buffer.

Return values

TRUE	has new data
FALSE	no new data

2.5.2.3 err_t uart_c::Out (uint8_t *outNum*)

Send one raw 8-bit number.

Parameters

outNum	8-bit number to send

Returns

HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.4 err_t uart_c::Out (uint16_t *outNum*)

Send one raw 16-bit number. The low-order part is sent first.

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

outNum	16-bit number to send.

Returns

HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.5 err_t uart_c::Out (uint32_t *outNum*)

Send one raw 32-bit number. The low-order part is sent first.

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

outNum	32-bit number to send.

Returns

HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.6 err_t uart_c::Out (uint8_t *outBuf[ ],* uint32_t *bufLen*)

Send one buffer.

Parameters

outBuf	Buffer to send
bufLen	Length in byte of the buffer

Returns

HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.7 err_t uart_c::Print (char *outChar*)

Send one character.

Parameters

outChar	Character to send

Returns

HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.8 err_t uart_c::Print (char $*$ *outStr*)

Send one string.

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

outStr	String to send

Returns

HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.9 err_t uart_c::Print (uint32_t *outNum,* uint8_t *radix*)

Convert one unsigned interger into string and then send it.

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

outNum	Unsigned interger number to send
radix	Valid values are 2, 8, 10, 16

Returns

HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.10 err_t uart_c::Print (uint32_t *outNum*)

Convert one unsigned interger into string and then send it with default radix (decimal)

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

outNum	Unsigned interger number to send

Returns

HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.11 err_t uart_c::Print (int32_t *outNum*)

Convert one signed interger into decimal string and then send it.

Parameters

outNum	Signed interger number to send @ HL_OK, HL_NOT_STARTED, HL_INVALID

2.5.2.12 err_t uart_c::Shutdown (void)

Release I/O pins, stop UART clock.

Returns

HL_OK, HL_INVALID

2.5.2.13 err_t uart_c::Start (uint32_t *baudRate,* uint16_t *stopBit*)

Start serial communicating.

Parameters

baudRate	UART's baud rate
stopBit	UART's stopBit

Returns

HL_OK , HL_INVALID

2.5.2.14 err_t uart_c::Start (uint32_t *baudRate*)

Start serial communicating in default mode with 8 data-bit, 1 stop-bit, no-parity.

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

baudRate	UART's baud rate

Returns

HL_OK, HL_INVALID

Index