intLib

Generated by Doxygen 1.8.6

Mon Mar 31 2014 16:47:50

Contents

Chapter 1

uIntPLib

Universal Integrated Peripheral Library

This is a library made with functions masks to medium level programming. Intended to make code more portable, while maintaning its performance.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Data Structure Documentation

4.1 CommandInstance Struct Reference

```
#include <cmd_sort.h>
```

Data Fields

- uint8_t charIn
- uint8_t cmdBuffer [MAX_BUFFER_SIZE]
- uint16_t charOut [MAX_BUFFER_SIZE]
- uint8_t charOutPtr

4.1.1 Detailed Description

Definition at line 16 of file cmd_sort.h.

4.1.2 Field Documentation

4.1.2.1 uint8_t CommandInstance::charIn

Definition at line 17 of file cmd_sort.h.

4.1.2.2 uint16_t CommandInstance::charOut[MAX_BUFFER_SIZE]

Definition at line 19 of file cmd_sort.h.

4.1.2.3 uint8_t CommandInstance::charOutPtr

Definition at line 20 of file cmd_sort.h.

4.1.2.4 uint8_t CommandInstance::cmdBuffer[MAX_BUFFER_SIZE]

Definition at line 18 of file cmd_sort.h.

The documentation for this struct was generated from the following file:

- my_lib/cmd_sort.h

4.2 IRInstance Struct Reference

```
#include <ir.h>
```

Data Fields

- uint16_t Mode
- uint8_t CarrierFrequency
- uint16_t CarrierPeriod
- uint32_t TxPin
- uint32_t TxPort
- uint32_t RxPin
- uint32_t RxPort
- uint16_t ReceiveAddress
- uint16_t ReceiveBuffer
- uint16_t Pulses
- uint8_t LastData

4.2.1 Detailed Description

Definition at line 87 of file ir.h.

4.2.2 Field Documentation

4.2.2.1 uint8_t IRInstance::CarrierFrequency

Definition at line 89 of file ir.h.

4.2.2.2 uint16_t IRInstance::CarrierPeriod

Definition at line 90 of file ir.h.

4.2.2.3 uint8_t IRInstance::LastData

Definition at line 98 of file ir.h.

4.2.2.4 uint16_t IRInstance::Mode

Definition at line 88 of file ir.h.

4.2.2.5 uint16_t IRInstance::Pulses

Definition at line 97 of file ir.h.

4.2.2.6 uint16_t IRInstance::ReceiveAddress

Definition at line 95 of file ir.h.

4.2.2.7 uint16_t IRInstance::ReceiveBuffer

Definition at line 96 of file ir.h.

4.2.2.8 uint32_t IRInstance::RxPin

Definition at line 93 of file ir.h.

4.2.2.9 uint32_t IRInstance::RxPort

Definition at line 94 of file ir.h.

4.2.2.10 uint32_t IRInstance::TxPin

Definition at line 91 of file ir.h.

4.2.2.11 uint32_t IRInstance::TxPort

Definition at line 92 of file ir.h.

The documentation for this struct was generated from the following file:

- my_lib/ir.h

4.3 LCDStatus Struct Reference

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

Data Fields

- uint8_t row
- uint8_t col
- uint8_t display
- uint8_t shift
- uint8_t cgramAdress
- uint8_t specialChar [8]

4.3.1 Detailed Description

Definition at line 120 of file lcd.h.

4.3.2 Field Documentation

4.3.2.1 uint8_t LCDStatus::cgramAdress

Definition at line 126 of file lcd.h.

4.3.2.2 uint8_t LCDStatus::col

Definition at line 123 of file lcd.h.

4.3.2.3 uint8_t LCDStatus::display

Definition at line 124 of file lcd.h.

4.3.2.4 uint8_t LCDStatus::row

Definition at line 122 of file lcd.h.

4.3.2.5 uint8_t LCDStatus::shift

Definition at line 125 of file lcd.h.

4.3.2.6 uint8_t LCDStatus::specialChar[8]

Definition at line 127 of file lcd.h.

The documentation for this struct was generated from the following file:

- my_lib/lcd.h

4.4 UARTInstance Struct Reference

```
#include <myUart.h>
```

Data Fields

- uint8_t RxBuffer [UART_BUFFER_SIZE]
- uint8_t RxBufferPtr
- uint8_t TxBuffer [UART_BUFFER_SIZE]
- uint8_t TxBufferPtr
- uint16_t Mode
- uint8_t TxLastSent [UART_BUFFER_SIZE]
- uint8_t TxLastSentPtr

4.4.1 Detailed Description

Definition at line 25 of file myUart.h.

4.4.2 Field Documentation

4.4.2.1 uint16_t UARTInstance::Mode

Definition at line 30 of file myUart.h.

4.4.2.2 uint8_t UARTInstance::RxBuffer[UART_BUFFER_SIZE]

Definition at line 26 of file myUart.h.

4.4.2.3 uint8_t UARTInstance::RxBufferPtr

Definition at line 27 of file myUart.h.

4.4.2.4 uint8_t UARTInstance::TxBuffer[UART_BUFFER_SIZE]

Definition at line 28 of file myUart.h.

4.4.2.5 uint8_t UARTInstance::TxBufferPtr

Definition at line 29 of file myUart.h.

4.4.2.6 uint8_t UARTInstance::TxLastSent[UART_BUFFER_SIZE]

Definition at line 31 of file myUart.h.

4.4.2.7 uint8_t UARTInstance::TxLastSentPtr

Definition at line 32 of file myUart.h.

The documentation for this struct was generated from the following file:

- my_lib/myUart.h

Chapter 5

File Documentation

5.1 depl_spc/chip_specific.c File Reference

```
#include "chip_specific.h"
```

5.2 chip_specific.c

```
/*
 * chip_specific.c
 *
 *  Created on: Mar 25, 2014
 *      Author: rikardo
 */


#include "chip_specific.h"





```

5.3 depl_spc/chip_specific.h File Reference

```
#include "includeAll_sw.h"
#include "includeAll_hw.h"
```

5.4 chip_specific.h

```
/*
 * chip_specific.h
 *
 *  Created on: Mar 25, 2014
 *      Author: rikardo
 */

#ifndef CHIP_SPECIFIC_H_
#define CHIP_SPECIFIC_H_

#include "includeAll_sw.h"
#include "includeAll_hw.h"



#endif /* CHIP_SPECIFIC_H_ */
```

5.5 depl_spc/cmd_list.h File Reference

5.6 cmd_list.h

```
/*
 * cmd_list.h
 *
 *  Created on: Mar 25, 2014
 *      Author: rikardo
 */

#ifndef CMD_LIST_H_
#define CMD_LIST_H_



#endif /* CMD_LIST_H_ */
```

5.7 depl_spc/device_init/hardwareInit.c File Reference

```
#include "hardwareInit.h"
```

Functions

- void HardwareInit (void)

5.7.1 Function Documentation

5.7.1.1 void HardwareInit (void)

Definition at line 13 of file hardwareInit.c.

5.8 hardwareInit.c

```
00001 /*
00002  * hardwareInit.c
00003  *
00004  *  Created on: Feb 5, 2014
00005  *      Author: rikardo
00006  */
00007
00008
00009
00010 #include "hardwareInit.h"
00011
00012
00013 void HardwareInit(void)
00014 {
00015
00016 }
00017
00018
00019
```

5.9 depl_spc/device_init/hardwareInit.h File Reference

```
#include "depl_spc/includeAll_hw.h"
```

Functions

- void HardwareInit (void)

5.9.1 Function Documentation

5.9.1.1 void HardwareInit (void)

Definition at line 13 of file hardwareInit.c.

5.10 hardwareInit.h

```
/*
 * hardwareInit.h
 *
 *  Created on: Feb 5, 2014
 *      Author: rikardo
 */

#ifndef HARDWAREINIT_H_
#define HARDWAREINIT_H_

#include "depl_spc/includeAll_hw.h"







void HardwareInit(void);

#endif /* HARDWAREINIT_H_ */
```

5.11 depl_spc/device_init/softwareInit.c File Reference

```
#include "softwareInit.h"
```

Functions

- void SoftwareInit (void)

5.11.1 Function Documentation

5.11.1.1 void SoftwareInit (void)

Definition at line 12 of file softwareInit.c.

5.12 softwareInit.c

```
/*
 * softwareInit.c
 *
 *  Created on: Feb 5, 2014
 *      Author: rikardo
 */

#include "softwareInit.h"


```

```
00011
00012 void SoftwareInit(void)
00013 {
00014
00015 }
00016
```

5.13 depl_spc/device_init/softwareInit.h File Reference

```
#include "depl_spc/includeAll_sw.h"
```

Functions

- void SoftwareInit (void)

5.13.1 Function Documentation

5.13.1.1 void SoftwareInit (void)

Definition at line 12 of file softwareInit.c.

5.14 softwareInit.h

```
/*
 * softwareInit.h
 *
 *  Created on: Feb 5, 2014
 *      Author: rikardo
 */

#ifndef SOFTWAREINIT_H_
#define SOFTWAREINIT_H_

#include "depl_spc/includeAll_sw.h"






void SoftwareInit(void);

#endif /* SOFTWAREINIT_H_ */
```

5.15 depl_spc/globalParam.h File Reference

Macros

- #define PROJECT_NAME ("your projects name here")
- #define LCD_SPLASHSCREEN1 1
- #define LCD_SPLASHSCREEN 1
- #define CPU_CLOCK 48
- #define BUS_CLOCK CPU_CLOCK/2
- #define CPUHZ_CLOCK 48000000
- #define BUSHZ_CLOCK CPUHZ_CLOCK/2

5.15.1 Macro Definition Documentation

5.15.1.1 #define BUS_CLOCK CPU_CLOCK/2

Definition at line 19 of file globalParam.h.

5.15.1.2 #define BUSHZ_CLOCK CPUHZ_CLOCK/2

Definition at line 21 of file globalParam.h.

5.15.1.3 #define CPU_CLOCK 48

Definition at line 18 of file globalParam.h.

5.15.1.4 #define CPUHZ_CLOCK 48000000

Definition at line 20 of file globalParam.h.

5.15.1.5 #define LCD_SPLASHSCREEN 1

Definition at line 15 of file globalParam.h.

5.15.1.6 #define LCD_SPLASHSCREEN1 1

Definition at line 14 of file globalParam.h.

5.15.1.7 #define PROJECT_NAME ("your projects name here")

Definition at line 13 of file globalParam.h.

5.16 globalParam.h

```
00001 /*
00002  * globalParam.h
00003  *
00004  *  Created on: Mar 26, 2014
00005  *      Author: rikardo
00006  */
00007
00008 #ifndef GLOBALPARAM_H_
00009 #define GLOBALPARAM_H_
00010
00011
00012
00013 #define PROJECT_NAME        ("your projects name here")
00014 #define LCD_SPLASHSCREEN1   1   //enables proejct name in 2 secs splash
00015 #define LCD_SPLASHSCREEN    1   //enables date and time of compilation
00016
00017
00018 #define CPU_CLOCK   48
00019 #define BUS_CLOCK   CPU_CLOCK/2
00020 #define CPUHZ_CLOCK 48000000
00021 #define BUSHZ_CLOCK CPUHZ_CLOCK/2
00022
00023
00024
00025
00026
00027 #endif /* GLOBALPARAM_H_ */
```

5.17 depl_spc/includeAll_hw.h File Reference

```
#include "globalParam.h"
#include "depl_spc/device_init/hardwareInit.h"
#include "depl_spc/lib_comp/external_cons.h"
#include "chip_specific.h"
```

5.18 includeAll_hw.h

```
/*
 * includeAll_hw.h
 *
 *  Created on: Feb 5, 2014
 *      Author: rikardo
 */

#ifndef INCLUDEALL_HW_H_
#define INCLUDEALL_HW_H_

//program definitions
#include "globalParam.h"


//masks for chip

//functions for peripherals



#include "depl_spc/device_init/hardwareInit.h"
#include "depl_spc/lib_comp/external_cons.h"
#include "chip_specific.h"


#endif /* INCLUDEALL_HW_H_ */
```

5.19 depl_spc/includeAll_sw.h File Reference

```
#include "stdint.h"
#include "stdbool.h"
#include "depl_spc/lib_comp/libraryCompatible.h"
#include "my_use.h"
#include "lcd.h"
#include "depl_spc/device_init/softwareInit.h"
#include "depl_spc/device_init/hardwareInit.h"
```

5.20 includeAll_sw.h

```
/*
 * includeAll_sw.h
 *
 *  Created on: Mar 25, 2014
 *      Author: rikardo
 */

#ifndef INCLUDEALL_SW_H_
#define INCLUDEALL_SW_H_

#include "stdint.h"
#include "stdbool.h"

//basic low level functions
#include "depl_spc/lib_comp/libraryCompatible.h"
#include "my_use.h"

```

```
//external peripherals
#include "lcd.h"



#include "depl_spc/device_init/softwareInit.h"
#include "depl_spc/device_init/hardwareInit.h"

#endif /* INCLUDEALL_SW_H_ */
```

5.21 depl_spc/lib_comp/external_cons.h File Reference

Macros

- #define LCD_RS_Port PTE_BASE_PTR
- #define LCD_RS_Pin IOPin_30
- #define LCD_EN_Port PTE_BASE_PTR
- #define LCD_EN_Pin IOPin_29
- #define LCD_DTA_Port PTE_BASE_PTR
- #define LCD_DTA_Pin IOPin_22
- #define LCD_CLK_Port PTE_BASE_PTR
- #define LCD_CLK_Pin IOPin_23
- #define LCD_row_num 2
- #define LCD_col_num 16
- #define LCD_char_heigh 8
- #define LCD_char_width 5

5.21.1 Macro Definition Documentation

5.21.1.1 #define LCD_char_heigh 8

Definition at line 29 of file external_cons.h.

5.21.1.2 #define LCD_char_width 5

Definition at line 30 of file external_cons.h.

5.21.1.3 #define LCD_CLK_Pin IOPin_23

Definition at line 25 of file external_cons.h.

5.21.1.4 #define LCD_CLK_Port PTE_BASE_PTR

Definition at line 24 of file external_cons.h.

5.21.1.5 #define LCD_col_num 16

Definition at line 28 of file external_cons.h.

5.21.1.6 #define LCD_DTA_Pin IOPin_22

Definition at line 22 of file external_cons.h.

5.21.1.7 #define LCD_DTA_Port PTE_BASE_PTR

Definition at line 21 of file external_cons.h.

5.21.1.8 #define LCD_EN_Pin IOPin_29

Definition at line 19 of file external_cons.h.

5.21.1.9 #define LCD_EN_Port PTE_BASE_PTR

Definition at line 18 of file external_cons.h.

5.21.1.10 #define LCD_row_num 2

Definition at line 27 of file external_cons.h.

5.21.1.11 #define LCD_RS_Pin IOPin_30

Definition at line 16 of file external_cons.h.

5.21.1.12 #define LCD_RS_Port PTE_BASE_PTR

Definition at line 15 of file external_cons.h.

5.22 external_cons.h

```
00001 #ifndef external_cons_h
00002 #define external_cons_h
00003
00004
00005 /*
00006  * file used to declare masks to external peripherals
00007  *
00008  */
00009
00010
00011 /*
00012  * Definitions for LCD peripheral
00013  */
00014 //LCD
00015 #define LCD_RS_Port     PTE_BASE_PTR
00016 #define LCD_RS_Pin      IOPin_30
00017
00018 #define LCD_EN_Port     PTE_BASE_PTR
00019 #define LCD_EN_Pin      IOPin_29
00020
00021 #define LCD_DTA_Port        PTE_BASE_PTR
00022 #define LCD_DTA_Pin     IOPin_22
00023
00024 #define LCD_CLK_Port        PTE_BASE_PTR
00025 #define LCD_CLK_Pin     IOPin_23
00026
00027 #define LCD_row_num     2
00028 #define LCD_col_num     16
00029 #define LCD_char_heigh  8
00030 #define LCD_char_width  5
00031
00032
00033
00034 #endif//external_cons_h
```

5.23 depl_spc/lib_comp/libraryCompatible.h File Reference

```
#include "depl_spc/includeAll_hw.h"
#include "gpioPin_masks.h"
```

Macros

- #define PinSet(port, pin) (port##_PSOR = pin)
- #define PinClear(port, pin) (port##_PCOR = pin)
- #define PinToogle(port, pin) (port##_PTOR = pin)
- #define PinAddrSet(port, pin) (GPIO_PSOR_REG((GPIO_MemMapPtr)port) = pin)
- #define PinAddrClear(port, pin) (GPIO_PCOR_REG((GPIO_MemMapPtr)port) = pin)
- #define SysDelay(time) SysDelayFRDM(time)
- #define SysDelayUs(time) SysDelay((time∗BUS_CLOCK)/6)
- #define SysDelayMs(time) SysDelayUs(time∗1000)

5.23.1 Macro Definition Documentation

5.23.1.1 #define PinAddrClear(*port*, *pin*) (GPIO_PCOR_REG((GPIO_MemMapPtr)port) = pin)

Definition at line 22 of file libraryCompatible.h.

5.23.1.2 #define PinAddrSet(*port*, *pin*) (GPIO_PSOR_REG((GPIO_MemMapPtr)port) = pin)

Definition at line 21 of file libraryCompatible.h.

5.23.1.3 #define PinClear(*port*, *pin*) (port##_PCOR = pin)

Definition at line 17 of file libraryCompatible.h.

5.23.1.4 #define PinSet(*port*, *pin*) (port##_PSOR = pin)

Definition at line 16 of file libraryCompatible.h.

5.23.1.5 #define PinToogle(*port*, *pin*) (port##_PTOR = pin)

Definition at line 18 of file libraryCompatible.h.

5.23.1.6 #define SysDelay(*time*) SysDelayFRDM(time)

Definition at line 26 of file libraryCompatible.h.

5.23.1.7 #define SysDelayMs(*time*) SysDelayUs(time∗1000)

Definition at line 28 of file libraryCompatible.h.

5.23.1.8 #define SysDelayUs(*time*) SysDelay((time∗BUS_CLOCK)/6)

Definition at line 27 of file libraryCompatible.h.

5.24 libraryCompatible.h

```
/*
 * libraryCompatible.h
 *
 *  Created on: Feb 5, 2014
 *      Author: rikardo
 */

#ifndef LIBRARYCOMPATIBLE_H_
#define LIBRARYCOMPATIBLE_H_

#include "depl_spc/includeAll_hw.h"
#include "gpioPin_masks.h"


//direct setting, uses a pre-casted object
#define PinSet(port, pin)       (port##_PSOR = pin)
#define PinClear(port, pin)     (port##_PCOR = pin)
#define PinToogle(port, pin)    (port##_PTOR = pin)

//casts the address to the structure referenced in the memory mapping file
#define PinAddrSet(port, pin)   (GPIO_PSOR_REG((GPIO_MemMapPtr)port) = pin)
#define PinAddrClear(port, pin) (GPIO_PCOR_REG((GPIO_MemMapPtr)port) = pin)



#define SysDelay(time)          SysDelayFRDM(time)          //chip specific
#define SysDelayUs(time)        SysDelay((time*BUS_CLOCK)/6)
#define SysDelayMs(time)        SysDelayUs(time*1000)





#endif /* LIBRARYCOMPATIBLE_H_ */
```

5.25 depl_spc/variables.h File Reference

Macros

- #define variable_h

5.25.1 Macro Definition Documentation

5.25.1.1 #define variable_h

Definition at line 2 of file variables.h.

5.26 variables.h

```
00001 #ifndef variables_h
00002 #define variable_h
00003
00004
00005
00006 #endif
```

5.27 my_lib/ascii.h File Reference

Macros

- #define ASCII_NULL 0
- #define ASCII_SOH 1
- #define ASCII_STX 2

- #define ASCII_ETX 3
- #define ASCII_EOT 4
- #define ASCII_ENQ 5
- #define ASCII_ACK 6
- #define ASCII_BEL 7
- #define ASCII_BS 8
- #define ASCII_HT 9
- #define ASCII_LF 10
- #define ASCII_VT 11
- #define ASCII_FF 12
- #define ASCII_CR 13
- #define ASCII_SO 14
- #define ASCII_SI 15
- #define ASCII_DLE 16
- #define ASCII_DC1 17
- #define ASCII_DC2 18
- #define ASCII_DC3 19
- #define ASCII_DC4 20
- #define ASCII_NAK 21
- #define ASCII_SYN 22
- #define ASCII_ETB 23
- #define ASCII_CAN 24
- #define ASCII_EM 25
- #define ASCII_SUB 26
- #define ASCII_ESC 27
- #define ASCII_FS 28
- #define ASCII_GS 29
- #define ASCII_RS 30
- #define ASCII_US 31

5.27.1 Macro Definition Documentation

5.27.1.1 #define ASCII_ACK 6

Definition at line 22 of file ascii.h.

5.27.1.2 #define ASCII_BEL 7

Definition at line 23 of file ascii.h.

5.27.1.3 #define ASCII_BS 8

Definition at line 24 of file ascii.h.

5.27.1.4 #define ASCII_CAN 24

Definition at line 40 of file ascii.h.

5.27.1.5 #define ASCII_CR 13

Definition at line 29 of file ascii.h.

5.27.1.6 #define ASCII_DC1 17

Definition at line 33 of file ascii.h.

5.27.1.7 #define ASCII_DC2 18

Definition at line 34 of file ascii.h.

5.27.1.8 #define ASCII_DC3 19

Definition at line 35 of file ascii.h.

5.27.1.9 #define ASCII_DC4 20

Definition at line 36 of file ascii.h.

5.27.1.10 #define ASCII_DLE 16

Definition at line 32 of file ascii.h.

5.27.1.11 #define ASCII_EM 25

Definition at line 41 of file ascii.h.

5.27.1.12 #define ASCII_ENQ 5

Definition at line 21 of file ascii.h.

5.27.1.13 #define ASCII_EOT 4

Definition at line 20 of file ascii.h.

5.27.1.14 #define ASCII_ESC 27

Definition at line 43 of file ascii.h.

5.27.1.15 #define ASCII_ETB 23

Definition at line 39 of file ascii.h.

5.27.1.16 #define ASCII_ETX 3

Definition at line 19 of file ascii.h.

5.27.1.17 #define ASCII_FF 12

Definition at line 28 of file ascii.h.

5.27.1.18 #define ASCII_FS 28

Definition at line 44 of file ascii.h.

5.27.1.19 #define ASCII_GS 29

Definition at line 45 of file ascii.h.

5.27.1.20 #define ASCII_HT 9

Definition at line 25 of file ascii.h.

5.27.1.21 #define ASCII_LF 10

Definition at line 26 of file ascii.h.

5.27.1.22 #define ASCII_NAK 21

Definition at line 37 of file ascii.h.

5.27.1.23 #define ASCII_NULL 0

Definition at line 16 of file ascii.h.

5.27.1.24 #define ASCII_RS 30

Definition at line 46 of file ascii.h.

5.27.1.25 #define ASCII_SI 15

Definition at line 31 of file ascii.h.

5.27.1.26 #define ASCII_SO 14

Definition at line 30 of file ascii.h.

5.27.1.27 #define ASCII_SOH 1

Definition at line 17 of file ascii.h.

5.27.1.28 #define ASCII_STX 2

Definition at line 18 of file ascii.h.

5.27.1.29 #define ASCII_SUB 26

Definition at line 42 of file ascii.h.

5.27.1.30 #define ASCII_SYN 22

Definition at line 38 of file ascii.h.

5.27.1.31 #define ASCII_US 31

Definition at line 47 of file ascii.h.

5.27.1.32 #define ASCII_VT 11

Definition at line 27 of file ascii.h.

5.28 ascii.h

```
00001 /*
00002  * ascii.h
00003  *
00004  *  Created on: Nov 25, 2013
00005  *      Author: rikardo
00006  */
00007
00008 #ifndef ASCII_H_
00009 #define ASCII_H_
00010
00011 /*
00012  * File contaning ASCII Masks
00013  */
00014
00015
00016 #define ASCII_NULL          0           //Null Char
00017 #define ASCII_SOH           1           //Start of Header
00018 #define ASCII_STX           2           //Start of Text
00019 #define ASCII_ETX           3           //End of Text
00020 #define ASCII_EOT           4           //End of Transmission
00021 #define ASCII_ENQ           5           //Enquiry
00022 #define ASCII_ACK           6           //Ack
00023 #define ASCII_BEL           7           //Bell
00024 #define ASCII_BS            8           //BackSpace
00025 #define ASCII_HT            9           //Horizontal Tab
00026 #define ASCII_LF            10          //Line Feed
00027 #define ASCII_VT            11          //Vertical Tab
00028 #define ASCII_FF            12          //Form Feed
00029 #define ASCII_CR            13          //Carriage Return
00030 #define ASCII_SO            14          //Shift Out
00031 #define ASCII_SI            15          //Shift In
00032 #define ASCII_DLE           16          //Data Link Escape
00033 #define ASCII_DC1           17          //Device Control 1
00034 #define ASCII_DC2           18
00035 #define ASCII_DC3           19
00036 #define ASCII_DC4           20
00037 #define ASCII_NAK           21          //Negative Ack
00038 #define ASCII_SYN           22          //Synchronous idle
00039 #define ASCII_ETB           23          //End of Transmission Block
00040 #define ASCII_CAN           24          //Cancel
00041 #define ASCII_EM            25          //End of Medium
00042 #define ASCII_SUB           26          //Substitute
00043 #define ASCII_ESC           27          //Escape
00044 #define ASCII_FS            28          //File Separator
00045 #define ASCII_GS            29          //Group Separtor
00046 #define ASCII_RS            30          //Record Separator
00047 #define ASCII_US            31          //Unit Separator
00048
00049
00050
00051
00052
00053
00054
00055 #endif /* ASCII_H_ */
```

5.29 my_lib/cmd_sort.c File Reference

```
#include "cmd_sort.h"
```

Functions

- void CommandSort (uint8_t ∗cmdString)

5.29.1 Function Documentation

5.29.1.1 void CommandSort (uint8_t ∗ *cmdString*)

Definition at line 16 of file cmd_sort.c.

5.30 cmd_sort.c

```
00001 /*
00002  * cmd_sort.c
00003  *
00004  *  Created on: Nov 28, 2013
00005  *      Author: rikardo
00006  */
00007 
00008 #include "cmd_sort.h"
00009 
00010 
00011 /*
00012  * Processes a string as a command
00013  * todo: make software interrupt for routines, call from here
00014  * todo: return function pointer
00015  */
00016 void CommandSort(uint8_t *cmdString)
00017 {
00018 
00019 }
00020 
00021 
```

5.31 my_lib/cmd_sort.h File Reference

```
#include "depl_spc/includeAll_sw.h"
#include "depl_spc/cmd_list.h"
```

Data Structures

- struct CommandInstance

Macros

- #define MAX_BUFFER_SIZE 30

Functions

- void CommandSort (uint8_t ∗cmdString)

5.31.1 Macro Definition Documentation

5.31.1.1 #define MAX_BUFFER_SIZE 30

Definition at line 14 of file cmd_sort.h.

5.31.2 Function Documentation

5.31.2.1 void CommandSort (uint8_t ∗ *cmdString*)

Definition at line 16 of file cmd_sort.c.

5.32 cmd_sort.h

```
00001 /*
00002  * cmd_sort.h
00003  *
00004  *  Created on: Nov 28, 2013
00005  *      Author: rikardo
00006  */
00007
00008 #ifndef CMD_SORT_H_
00009 #define CMD_SORT_H_
00010
00011 #include "depl_spc/includeAll_sw.h"
00012 #include "depl_spc/cmd_list.h"
00013
00014 #define MAX_BUFFER_SIZE             30
00015
00016 typedef struct{
00017     uint8_t charIn;
00018     uint8_t cmdBuffer[MAX_BUFFER_SIZE];
00019     uint16_t charOut[MAX_BUFFER_SIZE];
00020     uint8_t charOutPtr;
00021 } CommandInstance;
00022
00023
00024
00025 void CommandSort(uint8_t *cmdString);
00026
00027
00028
00029
00030 #endif /* CMD_SORT_H_ */
```

5.33 my_lib/gpioPin_masks.h File Reference

Macros

- #define IOPin_0 0x00000001
- #define IOPin_1 0x00000002
- #define IOPin_2 0x00000004
- #define IOPin_3 0x00000008
- #define IOPin_4 0x00000010
- #define IOPin_5 0x00000020
- #define IOPin_6 0x00000040
- #define IOPin_7 0x00000080
- #define IOPin_8 0x00000100
- #define IOPin_9 0x00000200
- #define IOPin_10 0x00000400
- #define IOPin_11 0x00000800
- #define IOPin_12 0x00001000
- #define IOPin_13 0x00002000

- #define IOPin_14 0x00004000
- #define IOPin_15 0x00008000
- #define IOPin_16 0x00010000
- #define IOPin_17 0x00020000
- #define IOPin_18 0x00040000
- #define IOPin_19 0x00080000
- #define IOPin_20 0x00100000
- #define IOPin_21 0x00200000
- #define IOPin_22 0x00400000
- #define IOPin_23 0x00800000
- #define IOPin_24 0x01000000
- #define IOPin_25 0x02000000
- #define IOPin_26 0x04000000
- #define IOPin_27 0x08000000
- #define IOPin_28 0x10000000
- #define IOPin_29 0x20000000
- #define IOPin_30 0x40000000
- #define IOPin_31 0x08000000

5.33.1 Macro Definition Documentation

5.33.1.1 #define IOPin_0 0x00000001

Definition at line 5 of file gpioPin_masks.h.

5.33.1.2 #define IOPin_1 0x00000002

Definition at line 6 of file gpioPin_masks.h.

5.33.1.3 #define IOPin_10 0x00000400

Definition at line 15 of file gpioPin_masks.h.

5.33.1.4 #define IOPin_11 0x00000800

Definition at line 16 of file gpioPin_masks.h.

5.33.1.5 #define IOPin_12 0x00001000

Definition at line 17 of file gpioPin_masks.h.

5.33.1.6 #define IOPin_13 0x00002000

Definition at line 18 of file gpioPin_masks.h.

5.33.1.7 #define IOPin_14 0x00004000

Definition at line 19 of file gpioPin_masks.h.

5.33.1.8 #define IOPin_15 0x00008000

Definition at line 20 of file gpioPin_masks.h.

5.33.1.9 #define IOPin_16 0x00010000

Definition at line 21 of file gpioPin_masks.h.

5.33.1.10 #define IOPin_17 0x00020000

Definition at line 22 of file gpioPin_masks.h.

5.33.1.11 #define IOPin_18 0x00040000

Definition at line 23 of file gpioPin_masks.h.

5.33.1.12 #define IOPin_19 0x00080000

Definition at line 24 of file gpioPin_masks.h.

5.33.1.13 #define IOPin_2 0x00000004

Definition at line 7 of file gpioPin_masks.h.

5.33.1.14 #define IOPin_20 0x00100000

Definition at line 25 of file gpioPin_masks.h.

5.33.1.15 #define IOPin_21 0x00200000

Definition at line 26 of file gpioPin_masks.h.

5.33.1.16 #define IOPin_22 0x00400000

Definition at line 27 of file gpioPin_masks.h.

5.33.1.17 #define IOPin_23 0x00800000

Definition at line 28 of file gpioPin_masks.h.

5.33.1.18 #define IOPin_24 0x01000000

Definition at line 29 of file gpioPin_masks.h.

5.33.1.19 #define IOPin_25 0x02000000

Definition at line 30 of file gpioPin_masks.h.

5.33.1.20 #define IOPin_26 0x04000000

Definition at line 31 of file gpioPin_masks.h.

5.33.1.21 #define IOPin_27 0x08000000

Definition at line 32 of file gpioPin_masks.h.

5.33.1.22 #define IOPin_28 0x10000000

Definition at line 33 of file gpioPin_masks.h.

5.33.1.23 #define IOPin_29 0x20000000

Definition at line 34 of file gpioPin_masks.h.

5.33.1.24 #define IOPin_3 0x00000008

Definition at line 8 of file gpioPin_masks.h.

5.33.1.25 #define IOPin_30 0x40000000

Definition at line 35 of file gpioPin_masks.h.

5.33.1.26 #define IOPin_31 0x08000000

Definition at line 36 of file gpioPin_masks.h.

5.33.1.27 #define IOPin_4 0x00000010

Definition at line 9 of file gpioPin_masks.h.

5.33.1.28 #define IOPin_5 0x00000020

Definition at line 10 of file gpioPin_masks.h.

5.33.1.29 #define IOPin_6 0x00000040

Definition at line 11 of file gpioPin_masks.h.

5.33.1.30 #define IOPin_7 0x00000080

Definition at line 12 of file gpioPin_masks.h.

5.33.1.31 #define IOPin_8 0x00000100

Definition at line 13 of file gpioPin_masks.h.

5.33.1.32 #define IOPin_9 0x00000200

Definition at line 14 of file gpioPin_masks.h.

5.34 gpioPin_masks.h

```
00001 #ifndef GPIOPIN_MASKS
00002 #define GPIOPIN_MASKS
00003
00004
00005 #define IOPin_0     0x00000001
00006 #define IOPin_1     0x00000002
00007 #define IOPin_2     0x00000004
00008 #define IOPin_3     0x00000008
00009 #define IOPin_4     0x00000010
00010 #define IOPin_5     0x00000020
00011 #define IOPin_6     0x00000040
00012 #define IOPin_7     0x00000080
00013 #define IOPin_8     0x00000100
00014 #define IOPin_9     0x00000200
00015 #define IOPin_10    0x00000400
00016 #define IOPin_11    0x00000800
00017 #define IOPin_12    0x00001000
00018 #define IOPin_13    0x00002000
00019 #define IOPin_14    0x00004000
00020 #define IOPin_15    0x00008000
00021 #define IOPin_16    0x00010000
00022 #define IOPin_17    0x00020000
00023 #define IOPin_18    0x00040000
00024 #define IOPin_19    0x00080000
00025 #define IOPin_20    0x00100000
00026 #define IOPin_21    0x00200000
00027 #define IOPin_22    0x00400000
00028 #define IOPin_23    0x00800000
00029 #define IOPin_24    0x01000000
00030 #define IOPin_25    0x02000000
00031 #define IOPin_26    0x04000000
00032 #define IOPin_27    0x08000000
00033 #define IOPin_28    0x10000000
00034 #define IOPin_29    0x20000000
00035 #define IOPin_30    0x40000000
00036 #define IOPin_31    0x08000000
00037
00038
00039
00040 #endif //gpiopin_masks
```

5.35 my_lib/ir.c File Reference

```
#include "ir.h"
```

Functions

- void IRInit (IRInstance ∗instPtr)
- void IRSend (IRInstance ∗instPtr, uint16_t address, uint16_t byte)
- void __inline IRByteBySoftware (IRInstance ∗instPtr, uint16_t address, uint16_t byte)
- void __inline IRRepeat (uint32_t port, uint32_t pin, uint8_t pulses, uint16_t delay)

5.35.1 Function Documentation

5.35.1.1 void __inline IRByteBySoftware (IRInstance ∗ *instPtr,* uint16_t *address,* uint16_t *byte*)

Definition at line 146 of file ir.c.

5.35.1.2 void IRInit (**IRInstance** ∗ *instPtr*)

Definition at line 16 of file ir.c.

5.35.1.3 void __inline IRRepeat (uint32_t *port,* uint32_t *pin,* uint8_t *pulses,* uint16_t *delay*)

Definition at line 212 of file ir.c.

5.35.1.4 void IRSend (**IRInstance** ∗ *instPtr,* uint16_t *address,* uint16_t *byte*)

Definition at line 57 of file ir.c.

5.36 ir.c

```
00001 #include "ir.h"
00002
00003 #include "ir.h"
00004
00005
00006
00007
00008
00009
00010 /*
00011  * Calls IR init
00012  * Modes: IR_BY_SOFTWARE
00013  *         IR_BY_UART
00014  *         IR_BY_TIMER
00015  */
00016 void IRInit(IRInstance *instPtr)
00017 {
00018
00019     if((instPtr->Mode&IR_BY_SOFTWARE)!=0)
00020     {
00021         instPtr->CarrierPeriod = (uint16_t) 1000/instPtr->
      CarrierFrequency;
00022         #ifdef IR_BY_SOFTWARE_EN
00023         /*
00024          * for software modulation, configure delay timing
00025          */
00026         if((instPtr->Mode & (IR_NEC_PROTOCOL|IR_NEC_EXTENDED))!=0)
00027             instPtr->Pulses = (uint16_t) (NEC_PULSE_TIME*((uint16_t) instPtr->
      CarrierFrequency))/2000;
00028         if((instPtr->Mode&IR_RC5_PROTOCOL)!=0)
00029             instPtr->Pulses = (uint16_t) (RC5_PULSE_TIME*((uint16_t) instPtr->
      CarrierFrequency))/1000;
00030         #endif
00031     }
00032
00033
00034     if(instPtr->Mode == IR_BY_UART)
00035     {
00036         #ifdef IR_BY_UART_EN
00037
00038         #endif
00039     }
00040
00041
00042     if(instPtr->Mode == IR_BY_TIMER)
00043     {
00044         #ifdef IR_BY_TIMER_EN
00045
00046         #endif
00047     }
00048
00049
00050 }
00051
00052
00053
00054 /*
00055  * Sends IR data according to instance
00056  */
00057 void IRSend(IRInstance *instPtr, uint16_t address, uint16_t byte)
00058 {
00059     uint32_t data;
```

```
    uint8_t tempAddress=0;
    uint8_t tempByte=0;

#ifdef IR_BY_SOFTWARE_EN
    uint16_t pulses;
    uint8_t roller;
    uint16_t delay = instPtr->CarrierPeriod/2;
#endif

    if((instPtr->Mode&IR_NEC_PROTOCOL)!=0)                       //inversdo enderee dados
    {
        tempAddress = ~address;
        address = ((address&0xFF))|((tempAddress&0xFF)<<8);
        tempByte = ~byte;
        byte = ((byte&0xFF))|((tempByte&0xFF)<<8);
    }
    if((instPtr->Mode&(IR_NEC_PROTOCOL|IR_NEC_EXTENDED))!=0)
            data = address|byte<<16;

#ifdef IR_BY_SOFTWARE_EN
    if((instPtr->Mode&IR_BY_SOFTWARE)!=0)
    {
        if((instPtr->Mode&(IR_NEC_EXTENDED|IR_NEC_PROTOCOL))!=0)      // padrde envio
        {
            pulses = instPtr->Pulses*32;
            roller = 32;
            while(pulses>0)                     //start signal send 9ms
            {
                IRPinSet(instPtr->TxPort, instPtr->TxPin);
                IRDelayUs(delay);
                IRPinClear(instPtr->TxPort, instPtr->TxPin);
                IRDelayUs(delay);
                pulses--;
            }
            IRDelayMs(4);                  //protocol wait time
            IRDelayUs(500);
            while(roller>0)
            {
                pulses = instPtr->Pulses;
                while(pulses>0)                 //carrier send
                {
                    IRPinSet(instPtr->TxPort, instPtr->TxPin);
                    IRDelayUs(delay);
                    IRPinClear(instPtr->TxPort, instPtr->TxPin);
                    IRDelayUs(delay);
                    pulses--;
                }
                if((data&0x1)!=0)
                    IRDelayUs(NEC_PULSE_TIME*2);
                IRDelayUs(NEC_PULSE_TIME);
                data >>= 1;
                roller --;
            }
            pulses = instPtr->Pulses;
            while(pulses>0)                 //end signal send 562.5 us
            {
                IRPinSet(instPtr->TxPort, instPtr->TxPin);
                IRDelayUs(delay);
                IRPinClear(instPtr->TxPort, instPtr->TxPin);
                IRDelayUs(delay);
                pulses--;
            }
        }
        if((instPtr->Mode&IR_RC5_PROTOCOL)!=0)
        {
            //todo: to be implemented. sem saco anymore.
        }
    }
#endif
#ifdef IR_BY_UART_EN
    if((instPtr->Mode&IR_BY_UART)!=0)
    {

        //todo: make uart send buffer/command
    }
#endif
#ifdef IR_BY_TIMER_EN

#endif
}


#ifdef IR_BY_SOFTWARE
/*
 * sends a modulated bit
 */
void __inline IRByteBySoftware(IRInstance *instPtr, uint16_t address, uint16_t byte)
{
    uint8_t tempAddress=0;
    uint8_t tempByte=0;
    uint16_t pulses;
    uint32_t data;
    uint8_t roller;
    uint16_t delay = instPtr->CarrierPeriod/2;

    if((instPtr->Mode&IR_NEC_PROTOCOL)!=0)                       //inversdo enderee dados
    {
        tempAddress = ~address;
        address = ((address&0xFF))|((tempAddress&0xFF)<<8);
        tempByte = ~byte;
        byte = ((byte&0xFF))|((tempByte&0xFF)<<8);
    }
    if((instPtr->Mode&(IR_NEC_EXTENDED|IR_NEC_PROTOCOL))!=0)      //padrde envio
    {
        data = address|byte<<16;
        pulses = instPtr->Pulses*32;
        roller = 32;
        while(pulses>0)                     //start signal send 9ms
        {
            IRPinSet(instPtr->TxPort, instPtr->TxPin);
            IRDelayUs(delay);
            IRPinClear(instPtr->TxPort, instPtr->TxPin);
            IRDelayUs(delay);
            pulses--;
        }
        IRDelayMs(4);                   //protocol wait time
        IRDelayUs(500);
        while(roller>0)
        {
            pulses = instPtr->Pulses;
            while(pulses>0)                 //carrier send
            {
                IRPinSet(instPtr->TxPort, instPtr->TxPin);
                IRDelayUs(delay);
                IRPinClear(instPtr->TxPort, instPtr->TxPin);
                IRDelayUs(delay);
                pulses--;
            }
            if((data&0x1)!=0)
                IRDelayUs(NEC_PULSE_TIME*2);
            IRDelayUs(NEC_PULSE_TIME);
            data >>= 1;
            roller --;
        }
        pulses = instPtr->Pulses;
        while(pulses>0)                 //end signal send 562.5 us
        {
            IRPinSet(instPtr->TxPort, instPtr->TxPin);
            IRDelayUs(delay);
            IRPinClear(instPtr->TxPort, instPtr->TxPin);
            IRDelayUs(delay);
            pulses--;
        }
    }
    if((instPtr->Mode&IR_RC5_PROTOCOL)!=0)
    {
        //todo: to be implemented. sem saco anymore.
    }
}



void __inline IRRepeat(uint32_t port, uint32_t pin, uint8_t pulses, uint16_t delay)
{
    uint8_t tempPulses;
    //fixme: repeat codes should be sent at 108ms intervals
    tempPulses = pulses;
    pulses *= 16;
    delay /= 2;
    while(pulses>0)                 //start signal send 9ms
    {
        IRPinSet(port, pin);
        IRDelayUs(delay);
        IRPinClear(port, pin);
        IRDelayUs(delay);
        pulses--;
    }
    IRDelayMs(2);
    IRDelayUs(250);
    pulses = tempPulses;
    while(pulses>0)                 //end signal send 562.5 us
    {
        IRPinSet(port, pin);
        IRDelayUs(delay);
        IRPinClear(port, pin);
        IRDelayUs(delay);
        pulses--;
    }
}




#endif //ir_by_software
```

5.37 my_lib/ir.h File Reference

```
#include "depl_spc/includeAll_sw.h"
#include "depl_spc/includeAll_hw.h"
#include "stdint.h"
#include "stdbool.h"
```

Data Structures

- struct IRInstance

Macros

- #define IR_MAX_INSTANCES 4
- #define IR_BY_SOFTWARE 0x0001
- #define IR_BY_UART 0x0002
- #define IR_BY_TIMER 0x0004
- #define IR_BY_EXTERNAL_TIEMR 0x0008
- #define IR_NEC_PROTOCOL 0x0010
- #define IR_NEC_EXTENDED 0x0020
- #define IR_MY_PROTOCOL 0x0040
- #define IR_RC5_PROTOCOL 0x0080
- #define IR_REPEAT_COMMAND_ENABLE 0x0100
- #define IR_REPEAT_COMMAND_DISABLE 0x0000
- #define NEC_PULSE_TIME 562
- #define RC5_PULSE_TIME 889
- #define IRPinSet(port, pin) GPIOPinWrite(port, pin, pin)
- #define IRPinClear(port, pin) GPIOPinWrite(port, pin, 0)
- #define IRDelayMs(delay) SysDelayMs(delay)
- #define IRDelayUs(delay) SysDelayUs(delay)
- #define IRDelay(delay) SysDelay(delay)

Functions

- void IRInit (IRInstance ∗instPtr)
- void IRSend (IRInstance ∗instPtr, uint16_t address, uint16_t byte)
- void IRByteBySoftware (IRInstance ∗instPtr, uint16_t address, uint16_t byte)
- void IRRepeat (uint32_t port, uint32_t pin, uint8_t pulses, uint16_t delay)

5.37.1 Macro Definition Documentation

5.37.1.1 #define IR_BY_EXTERNAL_TIEMR 0x0008

Definition at line 67 of file ir.h.

5.37.1.2 #define IR_BY_SOFTWARE 0x0001

Definition at line 64 of file ir.h.

5.37.1.3 #define IR_BY_TIMER 0x0004

Definition at line 66 of file ir.h.

5.37.1.4 #define IR_BY_UART 0x0002

Definition at line 65 of file ir.h.

5.37.1.5 #define IR_MAX_INSTANCES 4

Definition at line 59 of file ir.h.

5.37.1.6 #define IR_MY_PROTOCOL 0x0040

Definition at line 71 of file ir.h.

5.37.1.7 #define IR_NEC_EXTENDED 0x0020

Definition at line 70 of file ir.h.

5.37.1.8 #define IR_NEC_PROTOCOL 0x0010

Definition at line 69 of file ir.h.

5.37.1.9 #define IR_RC5_PROTOCOL 0x0080

Definition at line 72 of file ir.h.

5.37.1.10 #define IR_REPEAT_COMMAND_DISABLE 0x0000

Definition at line 75 of file ir.h.

5.37.1.11 #define IR_REPEAT_COMMAND_ENABLE 0x0100

Definition at line 74 of file ir.h.

5.37.1.12 #define IRDelay(*delay*) SysDelay(delay)

Definition at line 106 of file ir.h.

5.37.1.13 #define IRDelayMs(*delay*) SysDelayMs(delay)

Definition at line 104 of file ir.h.

5.37.1.14 #define IRDelayUs(*delay*) SysDelayUs(delay)

Definition at line 105 of file ir.h.

5.37.1.15 #define IRPinClear(*port,* *pin*) GPIOPinWrite(port, pin, 0)

Definition at line 103 of file ir.h.

5.37.1.16 #define IRPinSet(*port,* *pin*) GPIOPinWrite(port, pin, pin)

Definition at line 102 of file ir.h.

5.37.1.17 #define NEC_PULSE_TIME 562

Definition at line 81 of file ir.h.

5.37.1.18 #define RC5_PULSE_TIME 889

Definition at line 83 of file ir.h.

5.37.2 Function Documentation

5.37.2.1 void IRByteBySoftware (IRInstance ∗ *instPtr,* uint16_t *address,* uint16_t *byte*)

Definition at line 146 of file ir.c.

5.37.2.2 void IRInit (IRInstance ∗ *instPtr*)

Definition at line 16 of file ir.c.

5.37.2.3 void IRRepeat (uint32_t *port,* uint32_t *pin,* uint8_t *pulses,* uint16_t *delay*)

Definition at line 212 of file ir.c.

5.37.2.4 void IRSend (IRInstance ∗ *instPtr,* uint16_t *address,* uint16_t *byte*)

Definition at line 57 of file ir.c.

5.38 ir.h

```
#ifndef ir_h
#define ir_h



//main file header
#include "depl_spc/includeAll_sw.h"
#include "depl_spc/includeAll_hw.h"

#include "stdint.h"
#include "stdbool.h"


/*
 * need to declare:
 *

#define IRTX_FREQ           38000
#define IRTX_PORT           GPIO_PORTB_BASE
#define IRTX_PIN            GPIO_PIN_1
#define IRRX_PORT           GPIO_PORTB_BASE
#define IRRX_PIN            GPIO_PIN_0
//larger compiled file
#define IR_BY_SOFTWARE_EN
#define IR_BY_UART_EN
#define IR_UART_BASE        UART1_BASE
#define IR_BY_TIMER_EN
#define IR_TIMER_BASE       TIMER3_BASE



 */



/*
 * InfraRed transceiver library
 * todo:    uart
 *          timer
 *          software
 *
 * -> uart peripheral support
 *      initiated for stellaris/tivaC uCs, uart IR coding support
 *  -> timer peripheral support
 *      common to all uCs
 *      carrier frequency generation
 *  -> software support
 *      support for full software control and emulation
 *      delay by cpu time use
 *
 *
 *  devBy: rnm (17/11/13)
 */


/*
 * Op. Param.
 */
#define IR_MAX_INSTANCES            4

/*
 * Op. Masks
 */
#define IR_BY_SOFTWARE              0x0001
#define IR_BY_UART                  0x0002
#define IR_BY_TIMER                 0x0004
#define IR_BY_EXTERNAL_TIEMR        0x0008

#define IR_NEC_PROTOCOL             0x0010
#define IR_NEC_EXTENDED             0x0020
#define IR_MY_PROTOCOL              0x0040
#define IR_RC5_PROTOCOL             0x0080

#define IR_REPEAT_COMMAND_ENABLE    0x0100
#define IR_REPEAT_COMMAND_DISABLE   0x0000

/*
 * Protocol Definitions
 */

#define NEC_PULSE_TIME              562

#define RC5_PULSE_TIME              889

```

```


typedef struct{
    uint16_t Mode;              //IR_BY_XX | IR_XX_PROTOCOL
    uint8_t CarrierFrequency;   //in kHZ
    uint16_t CarrierPeriod;        // in uS
    uint32_t TxPin;
    uint32_t TxPort;
    uint32_t RxPin;
    uint32_t RxPort;
    uint16_t ReceiveAddress;
    uint16_t ReceiveBuffer;
    uint16_t Pulses;
    uint8_t LastData;
}IRInstance;


#define IRPinSet(port, pin)     GPIOPinWrite(port, pin, pin)
#define IRPinClear(port, pin)   GPIOPinWrite(port, pin, 0)
#define IRDelayMs(delay)        SysDelayMs(delay)
#define IRDelayUs(delay)        SysDelayUs(delay)
#define IRDelay(delay)          SysDelay(delay)



void IRInit(IRInstance *instPtr);
void IRSend(IRInstance *instPtr, uint16_t address, uint16_t byte);
void IRByteBySoftware(IRInstance *instPtr, uint16_t address, uint16_t byte);
void IRRepeat(uint32_t port, uint32_t pin, uint8_t pulses, uint16_t delay);



#endif// if_h
```

5.39 my_lib/lcd.c File Reference

```
#include "lcd.h"
```

Macros

- #define true 1
- #define false 0
- #define trueDefinedLCD
- #define lcd_vector_index 9

Functions

- void LCDInit (void)

  Initializes the LCD Module.
- void LCDSendCmd (uint8_t cmd)
- void LCDSendChar (uint8_t txt)

  send single character to LCD.
- __inline void LCDSend (uint8_t send)
- void LCDPosition (uint8_t row, uint8_t col)
- void LCDPositionNoDelay (uint8_t row, uint8_t col)
- void LCDSendString (uint8_t *string, uint8_t breakLine)
- void LCDSendNumStrict (int64_t num, uint8_t length, uint8_t isSigned, uint8_t showZeros)
- void LCDSendNum (int64_t num, uint8_t length, uint8_t isSigned, uint8_t showZeros)
- void LCDSendNumArray (uint8_t *index)
- void LCDClear (void)
- void LCDDisplayOn (uint8_t onOff)
- void LCDSendHex (uint8_t *array)
- void numToArray (int32_t num, uint8_t *array, uint8_t length, uint16_t base)

- void LCDRegisterSpecial (uint8_t number, uint8_t ∗character)
- void LCDShift (uint8_t shift)
- void LCDHome (void)
- void arrayToNum (uint8_t ∗array, uint32_t ∗num, uint8_t base)
- void LCDSendVU (uint32_t num, uint32_t base)

Variables

- const char LCD_CmdInit_Vector [lcd_vector_index]
- const unsigned int LCD_InitDelay_Vector [lcd_vector_index]

5.39.1 Macro Definition Documentation

5.39.1.1 #define false 0

Definition at line 12 of file lcd.c.

5.39.1.2 #define lcd_vector_index 9

Definition at line 21 of file lcd.c.

5.39.1.3 #define true 1

Checks and defines boolean values.

Definition at line 11 of file lcd.c.

5.39.1.4 #define trueDefinedLCD

Definition at line 13 of file lcd.c.

5.39.2 Function Documentation

5.39.2.1 void arrayToNum (uint8_t ∗ *array,* uint32_t ∗ *num,* uint8_t *base*)

Definition at line 422 of file lcd.c.

5.39.2.2 void LCDClear (void)

Definition at line 286 of file lcd.c.

5.39.2.3 void LCDDisplayOn (uint8_t *onOff*)

Definition at line 300 of file lcd.c.

5.39.2.4 void LCDHome (void)

Definition at line 414 of file lcd.c.

5.39.2.5 void LCDInit (void)

Initializes the LCD Module.

Called once at startup. Takes no parameters.

Returns

None.

Definition at line 47 of file lcd.c.

5.39.2.6 void LCDPosition (uint8_t *row,* uint8_t *col*)

Definition at line 117 of file lcd.c.

5.39.2.7 void LCDPositionNoDelay (uint8_t *row,* uint8_t *col*)

Definition at line 135 of file lcd.c.

5.39.2.8 void LCDRegisterSpecial (uint8_t *number,* uint8_t ∗ *character*)

Definition at line 387 of file lcd.c.

5.39.2.9 __inline void LCDSend (uint8_t *send*)

Definition at line 102 of file lcd.c.

5.39.2.10 void LCDSendChar (uint8_t *txt*)

send single character to LCD.

Parameters

txt	char type data to be sent, 8 bits.

Definition at line 91 of file lcd.c.

5.39.2.11 void LCDSendCmd (uint8_t *cmd*)

Definition at line 77 of file lcd.c.

5.39.2.12 void LCDSendHex (uint8_t ∗ *array*)

Definition at line 314 of file lcd.c.

5.39.2.13 void LCDSendNum (int64_t *num,* uint8_t *length,* uint8_t *isSigned,* uint8_t *showZeros*)

Definition at line 235 of file lcd.c.

5.39.2.14 void LCDSendNumArray (uint8_t ∗ *index*)

Definition at line 273 of file lcd.c.

5.39.2.15 void LCDSendNumStrict (int64_t *num,* uint8_t *length,* uint8_t *isSigned,* uint8_t *showZeros*)

Definition at line 188 of file lcd.c.

5.39.2.16 void LCDSendString (uint8_t ∗ *string,* uint8_t *breakLine*)

Definition at line 153 of file lcd.c.

5.39.2.17 void LCDSendVU (uint32_t *num,* uint32_t *base*)

Definition at line 434 of file lcd.c.

5.39.2.18 void LCDShift (uint8_t *shift*)

Definition at line 405 of file lcd.c.

5.39.2.19 void numToArray (int32_t *num,* uint8_t ∗ *array,* uint8_t *length,* uint16_t *base*)

Definition at line 347 of file lcd.c.

5.39.3 Variable Documentation

5.39.3.1 const char LCD_CmdInit_Vector[lcd_vector_index]

Initial value:

```
=
        {
            0x03, 0x38, 0x38, 0x38, 0x01, LCD_DISPLAY_CONFIG,
      LCD_DISPLAY_INCREMENT, 0x01, 0x02,
        }
```

Definition at line 22 of file lcd.c.

5.39.3.2 const unsigned int LCD_InitDelay_Vector[lcd_vector_index]

Initial value:

```
=
        {
            8000, 800, 800, 800, 600, 600, 200, 200, 200
        }
```

LCD Init command delay vector, in uS

Definition at line 32 of file lcd.c.

5.40 lcd.c

```

#include "lcd.h"




/**
 * Checks and defines boolean values.
 */
#ifndef true
#define true 1
#define false 0
#define trueDefinedLCD
#endif


/*
 * Initialization Sequence:
 * TODO: create masks for LCD commands
 */
#define lcd_vector_index    9
const char LCD_CmdInit_Vector [lcd_vector_index] = \
        {
            0x03, 0x38, 0x38, 0x38, 0x01, LCD_DISPLAY_CONFIG, LCD_DISPLAY_INCREMENT, 0x01, 0x02,
        };
/*
 * Delay time in uSs
 */
/**
 * LCD Init command delay vector, in uS
 */
const unsigned int LCD_InitDelay_Vector[lcd_vector_index] = \
        {
            8000, 800, 800, 800, 600, 600, 200, 200, 200
        };




/**
 *  \brief Initializes the LCD Module
 *
 *  Called once at startup. Takes no parameters.
 *
 *  \return None.
 */
void LCDInit(void)
{
    uint8_t Vector_Scan = 0;
    LCDDelay(15000);
    for(Vector_Scan=0; Vector_Scan < lcd_vector_index; Vector_Scan++)
    {
        LCDSendCmd(LCD_CmdInit_Vector[Vector_Scan]);
        LCDDelay(LCD_InitDelay_Vector[Vector_Scan]);
    }
    //splash screen
#if LCD_SPLASHSCREEN1 == 1
    LCDPosition(1,1);
    LCDSendString(LCD_splashscreen_row1, false);
    LCDPosition(2,1);
    LCDSendString(LCD_splashscreen_row2, false);
    LCDDelay(2*1000*1000);
#endif
#if LCD_SPLASHSCREEN2 == 1
    LCDPosition(1,1);
    LCDSendString(LCD_splashscreen2_row1, false);
    LCDPosition(2,1);
    LCDSendString(LCD_splashscreen2_row2, false);
    LCDDelay(2*1000*1000);
#endif
    LCDClear();
}

/*
 * Send a Command to the LCD
 */
void LCDSendCmd(uint8_t cmd)
{
    LCD_RS_Low;
    LCDSend(cmd);
}

/*
 * Send a Text to the LCD
 */
/**
 * \brief send single character to LCD.
 *
 * \param txt char type data to be sent, 8 bits.
 */
void LCDSendChar(uint8_t txt)
{
    LCD_RS_High;
    LCDSend(txt);
    LCD0Status.col ++;
    LCD_RS_Low;
}

/*
 * Write on serial shifter, pulse LCD EN
 */
__inline void LCDSend(uint8_t send)
{
    LCD_EN_Low;
    LCD_DTA_Send(send);
    LCDDelay(4);
    LCD_EN_High;
    LCDDelay(4);
    LCD_EN_Low;
}

/*
 * Changes the LCD Cursor Position
 * Updates LCDStatus
 * PARAM: row, col
 */
void LCDPosition(uint8_t row, uint8_t col)
{
    LCD0Status.row = row;
    LCD0Status.col = col;
    col--;
    if(row==1)
        row = 0x80;
    if(row==2)
        row = 0xC0;
    LCDSendCmd(row+col);
    LCDDelay(20);
}

/*
 * Changes the LCD Cursor Position
 * Updates LCDStatus
 * PARAM: row, col
 */
void LCDPositionNoDelay(uint8_t row, uint8_t col)
{
    LCD0Status.row = row;
    LCD0Status.col = col;
    col--;
    if(row==1)
        row = 0x80;
    if(row==2)
        row = 0xC0;
    LCDSendCmd(row+col);
}


/*
 * Writes a string of characteres on display
 * Processes according to the ASCII code
 * 0 - NULL
 */
void LCDSendString(uint8_t *string, uint8_t breakLine)
{
    while(*string)
    {
        LCDSendChar(*string);
        string++;
        if(LCD0Status.col==LCD_col_num && breakLine==true)
            if(LCD0Status.row<=LCD_row_num)
                LCDPosition(LCD0Status.row+1, 1);
            else
                LCDPosition(0, 1);
    }
}




/*
 * TODO: make function to print string and remaining spaces in LCD
 */


/*
 * Prints a number, from a variable, to the LCD
 * PARAM: num, length, isSigned, showZeros
 * IF signed
 *      Limits to a max of 10 digits to a positive number
 *      Limits to a max of 9 digits to a negative number
 * ELSE
 *      limits to 10 chars
 * IF showzeros
 *      shows all leading zeros
 * else
 *      supresses zeros; places space instead
 */
void LCDSendNumStrict(int64_t num, uint8_t length, uint8_t isSigned, uint8_t showZeros)
{
    uint8_t index =0;
    uint8_t out;
    uint64_t multiple = 1;
    limitCeilValue(length,10);
    if(num<0 && isSigned==true)
    {
        LCDSendChar('-');
        num *= -1;
        length--;
    }
    index = length;
    while(length>1)
    {
        multiple *= 10;
        length--;
    }
    while(index >= 1)
    {
        out = (uint32_t) (num/multiple);
        num -= out*(multiple);
        if(out!=0)
            showZeros = true;
        if(out==0 && showZeros==false)
            out -= 16;
        LCDSendChar(out+48);
        multiple /= 10;
        index--;
    }
}



/*
 * Prints a number, from a variable, to the LCD
 * PARAM: num, length, isSigned, showZeros
 * IF signed
 *      Limits to a max of 10 digits to a positive number
 *      Limits to a max of 9 digits to a negative number
 * ELSE
 *      limits to 10 chars
 * IF showzeros
 *      shows all leading zeros
 * else
 *      supresses zeros; places space instead
 */
void LCDSendNum(int64_t num, uint8_t length, uint8_t isSigned, uint8_t showZeros)
{
    uint8_t index =0;
    uint8_t out = ' ';
    uint64_t multiple = 1;
    limitCeilValue(length,10);
    if(num<0 && isSigned==true)
    {
        out = '-';
        num *= -1;
    }
    LCDSendChar(out);
    index = length;
    while(length>1)
    {
        multiple *= 10;
        length--;
    }
    while(index >= 1)
    {
        out = (uint32_t) (num/multiple);
        num -= out*(multiple);
        if(out!=0)
            showZeros = true;
        if(out==0 && showZeros==false)
            out -= 16;
        LCDSendChar(out+48);
        multiple /= 10;
        index--;
    }
}


/*
```

```
 * Sends numerical values to LCD
 * Values between 0 and base;
 * max base value is defined as 32 (32bit wide buses)
 */
void LCDSendNumArray(uint8_t *index)
{
    while(*index<33)
    {
        LCDSendChar(*index+’0’);
        index++;
    }
}

/*
 * Clears display
 * Updates LCDStatus
 */
void LCDClear(void)
{
    LCDSendCmd(0x01);
    LCD0Status.row=1;
    LCD0Status.col=1;
    LCDDelay(800);
}

/*
 * Turns
 * LCD_DISPLAY_ON/OFF
 * LCD_CURSOR_ON/OFF
 * LCD_BLINK_ON/OFF
 */
void LCDDisplayOn(uint8_t onOff)
{
    LCD0Status.display = onOff;
    LCDSendCmd(onOff);
}




/*
 * Prints the value of the array in hex format
 * As HEX, it’ll print in base 16
 * Takes out 2 leading digits
 */
void LCDSendHex(uint8_t *array)
{
    uint8_t offset, temp;
    LCDSendChar(’0’);
    LCDSendChar(’x’);
    array += 2;
    while(*array<=32)
    {
        temp = *array;
        if(temp>9)
        {
            temp -= 10;
            offset = ’A’;
        }
        else
            offset = ’0’;
        LCDSendChar(temp+offset);
        array++;
    }
}



//void LCDSendNum(long num, char length, uint8_t isSigned, uint8_t showZeros)

/*
 * Passes a number to a vector
 * num -> number
 * vector -> pointer to vector
 * base -> base of output (max: 32)
 *
 * Last number in vector output is 33
 */
void numToArray(int32_t num, uint8_t *array, uint8_t length, uint16_t base)
{
    uint16_t index =1;
    uint8_t out;
    uint64_t multiple = 1;

    limitCeilValue(length, (unsigned char) 1<<64/base);
    limitCeilValue(length, maxLengthOut);


    //create multiple number
    while(index<length)
    {
        multiple *= base;
        index++;
    }
    //sort multiples
    while(index >= 1)
    {
        //determines the multiple
        out = (uint8_t) (num/multiple);
        //takes out multiple
        num -= out*(multiple);

        //escreve no vetor, desloca indice
        *array = out;
        array++;
        multiple /= base;
        //change multiple position
        index--;
    }
    *array = 33;
}

/*
 * registers special characteres
 * number -> from 0 to 7
 *character -> first index to 8 bytes long vector
 *             scans char downward
 */
void LCDRegisterSpecial(uint8_t number, uint8_t *character)
{
    uint8_t scan=0, data=0;
    LCDSendCmd(0x40+(number<<3));
    do
    {
        data = *(character+scan);
        LCDDelay(640);
        LCDSendChar(data&0x1F);
        scan++;
    }
    while(scan<8);
    LCDDelay(320);
}

/*
 * Shifts data on LCD Display
 */
void LCDShift(uint8_t shift)
{
    LCDSendCmd(shift|LCD_SHIFT);
}

/*
 * Sends LCD cursor to home position
 * PARAM: none
 */
void LCDHome(void)
{
    LCDSendCmd(0x02);
    LCDDelay(1500);
}



void arrayToNum(uint8_t *array, uint32_t *num, uint8_t base)
{
    while(*array<33)
    {
        *num += *array * base;
        array++;
    }
}




void LCDSendVU(uint32_t num, uint32_t base)
{
    uint8_t index, pass=1;
    num  = (unsigned int) num*(LCD_col_num*LCD_char_width)/base;
    while(num>0)
    {
        index = LCD_char_width;
        while(num<LCD_char_width)
        {
            index--;
            num++;
        }
        LCDSendChar(index);
        num -= LCD_char_width;
        pass++;
    }
    while(pass<=LCD_col_num)
    {
        pass++;
        LCDSendChar(0);
    }
}




#ifdef trueDefinedLCD
#undef true
#undef false
#endif



```

5.41 my_lib/lcd.h File Reference

```
#include "depl_spc/includeAll_hw.h"
#include "depl_spc/includeAll_sw.h"
```

Data Structures

- struct LCDStatus

Macros

- #define LCD_splashscreen_row1 PROJECT_NAME

  geneartion of project name in LCD
- #define LCD_splashscreen_row2 ("rnm sys undvpd")

  creator's watermark
- #define LCD_splashscreen2_row1 __DATE__

  compile date, used as program version
- #define LCD_splashscreen2_row2 __TIME__

  compile time, used as program version
- #define LCDDelay(x) SysDelayUs(x)
- #define LCDPinSet(x, y) PinAddrSet(x, y)
- #define LCDPinClear(x, y) PinAddrClear(x,y)
- #define LCD_RS_High LCDPinSet(LCD_RS_Port, LCD_RS_Pin)
- #define LCD_RS_Low LCDPinClear(LCD_RS_Port, LCD_RS_Pin)
- #define LCD_EN_High LCDPinSet(LCD_EN_Port, LCD_EN_Pin)
- #define LCD_EN_Low LCDPinClear(LCD_EN_Port, LCD_EN_Pin)
- #define LCD_CLK_High LCDPinSet(LCD_CLK_Port, LCD_CLK_Pin)
- #define LCD_CLK_LoW LCDPinClear(LCD_CLK_Port, LCD_CLK_Pin)
- #define LCD_DTA_Send(text)
- #define LCD_DISPLAY_ON 0x0C
- #define LCD_DISPLAY_OFF 0x08
- #define LCD_CURSOR_ON 0x0A
- #define LCD_CURSOR_OFF 0x08
- #define LCD_BLINK_ON 0x09

- #define LCD_BLINK_OFF 0x08
- #define LCD_SHIFT 0x10
- #define LCD_SHIFT_DISPLAY 0x08
- #define LCD_SHIFT_CURSOR 0x02
- #define LCD_SHIFT_RIGHT 0x04
- #define LCD_SHIFT_LEFT 0x00
- #define LCD_SET_CGRAM 0x40
- #define LCD_INCREMENT 0X04
- #define LCD_INCREMENT_NO_SHIFT 0x00
- #define LCD_INCREMENT_SHIFT 0x01
- #define LCD_INCREMENT_POSITIVE 0x02
- #define LCD_INCREMENT_NEGATIVE 0x00
- #define LCD_DISPLAY_CONFIG (LCD_DISPLAY_ON|LCD_CURSOR_OFF|LCD_BLINK_OFF)
- #define LCD_DISPLAY_INCREMENT (LCD_INCREMENT|LCD_INCREMENT_NO_SHIFT)
- #define maxLengthOut 16

Functions

- void LCDInit (void)

  Initializes the LCD Module.
- void LCDSendCmd (uint8_t cmd)
- void LCDSendChar (uint8_t txt)

  send single character to LCD.
- void LCDSend (uint8_t send)
- void LCDPosition (uint8_t row, uint8_t col)
- void LCDPositionNoDelay (uint8_t row, uint8_t col)
- void LCDSendString (uint8_t ∗string, uint8_t breakLine)
- void LCDSendNumStrict (int64_t num, uint8_t length, uint8_t isSigned, uint8_t showZeros)
- void LCDSendNum (int64_t num, uint8_t length, uint8_t isSigned, uint8_t showZeros)
- void LCDSendNumArray (uint8_t ∗vector)
- void LCDClear (void)
- void LCDSendHex (uint8_t ∗array)
- void numToArray (int32_t num, uint8_t ∗array, uint8_t length, uint16_t base)
- void LCDRegisterSpecial (uint8_t number, uint8_t ∗character)
- void LCDShift (uint8_t shift)
- void LCDHome (void)
- void arrayToNum (uint8_t ∗array, uint32_t ∗num, uint8_t base)
- void LCDSendVU (uint32_t num, uint32_t base)

Variables

- LCDStatus LCD0Status

5.41.1 Macro Definition Documentation

5.41.1.1 #define LCD_BLINK_OFF 0x08

Definition at line 101 of file lcd.h.

5.41.1.2 #define LCD_BLINK_ON 0x09

Definition at line 100 of file lcd.h.

5.41.1.3 #define LCD_CLK_High LCDPinSet(LCD_CLK_Port, LCD_CLK_Pin)

Definition at line 87 of file lcd.h.

5.41.1.4 #define LCD_CLK_LoW LCDPinClear(LCD_CLK_Port, LCD_CLK_Pin)

Definition at line 88 of file lcd.h.

5.41.1.5 #define LCD_CURSOR_OFF 0x08

Definition at line 99 of file lcd.h.

5.41.1.6 #define LCD_CURSOR_ON 0x0A

Definition at line 98 of file lcd.h.

5.41.1.7 #define LCD_DISPLAY_CONFIG (LCD_DISPLAY_ON|LCD_CURSOR_OFF|LCD_BLINK_OFF)

Definition at line 116 of file lcd.h.

5.41.1.8 #define LCD_DISPLAY_INCREMENT (LCD_INCREMENT|LCD_INCREMENT_NO_SHIFT)

Definition at line 117 of file lcd.h.

5.41.1.9 #define LCD_DISPLAY_OFF 0x08

Definition at line 97 of file lcd.h.

5.41.1.10 #define LCD_DISPLAY_ON 0x0C

Definition at line 96 of file lcd.h.

5.41.1.11 #define LCD_DTA_Send(*text*)

Value:

```
ShiftSerialSend(LCD_DTA_Port,\
                                LCD_DTA_Pin,\
                                LCD_CLK_Port,\
                                LCD_CLK_Pin, text)
```

Definition at line 89 of file lcd.h.

5.41.1.12 #define LCD_EN_High LCDPinSet(LCD_EN_Port, LCD_EN_Pin)

Definition at line 85 of file lcd.h.

5.41.1.13 #define LCD_EN_Low LCDPinClear(LCD_EN_Port, LCD_EN_Pin)

Definition at line 86 of file lcd.h.

5.41.1.14 #define LCD_INCREMENT 0X04

Definition at line 108 of file lcd.h.

5.41.1.15 #define LCD_INCREMENT_NEGATIVE 0x00

Definition at line 112 of file lcd.h.

5.41.1.16 #define LCD_INCREMENT_NO_SHIFT 0x00

Definition at line 109 of file lcd.h.

5.41.1.17 #define LCD_INCREMENT_POSITIVE 0x02

Definition at line 111 of file lcd.h.

5.41.1.18 #define LCD_INCREMENT_SHIFT 0x01

Definition at line 110 of file lcd.h.

5.41.1.19 #define LCD_RS_High LCDPinSet(LCD_RS_Port, LCD_RS_Pin)

Definition at line 83 of file lcd.h.

5.41.1.20 #define LCD_RS_Low LCDPinClear(LCD_RS_Port, LCD_RS_Pin)

Definition at line 84 of file lcd.h.

5.41.1.21 #define LCD_SET_CGRAM 0x40

Definition at line 107 of file lcd.h.

5.41.1.22 #define LCD_SHIFT 0x10

Definition at line 102 of file lcd.h.

5.41.1.23 #define LCD_SHIFT_CURSOR 0x02

Definition at line 104 of file lcd.h.

5.41.1.24 #define LCD_SHIFT_DISPLAY 0x08

Definition at line 103 of file lcd.h.

5.41.1.25 #define LCD_SHIFT_LEFT 0x00

Definition at line 106 of file lcd.h.

5.41.1.26 #define LCD_SHIFT_RIGHT 0x04

Definition at line 105 of file lcd.h.

5.41.1.27 #define LCD_splashscreen2_row1 __DATE__

compile date, used as program version

Definition at line 13 of file lcd.h.

5.41.1.28 #define LCD_splashscreen2_row2 __TIME__

compile time, used as program version

Definition at line 14 of file lcd.h.

5.41.1.29 #define LCD_splashscreen_row1 PROJECT_NAME

geneartion of project name in LCD

Definition at line 10 of file lcd.h.

5.41.1.30 #define LCD_splashscreen_row2 ("rnm sys undvpd")

creator's watermark

Definition at line 11 of file lcd.h.

5.41.1.31 #define LCDDelay(*x*) SysDelayUs(x)

Definition at line 76 of file lcd.h.

5.41.1.32 #define LCDPinClear(*x*, *y*) PinAddrClear(x,y)

Definition at line 78 of file lcd.h.

5.41.1.33 #define LCDPinSet(*x*, *y*) PinAddrSet(x, y)

Definition at line 77 of file lcd.h.

5.41.1.34 #define maxLengthOut 16

Definition at line 155 of file lcd.h.

5.41.2 Function Documentation

5.41.2.1 void arrayToNum (uint8_t ∗ *array*, uint32_t ∗ *num*, uint8_t *base*)

Definition at line 422 of file lcd.c.

5.41.2.2 void LCDClear (void)

Definition at line 286 of file lcd.c.

5.41.2.3 void LCDHome (void)

Definition at line 414 of file lcd.c.

5.41.2.4 void LCDInit (void)

Initializes the LCD Module.

Called once at startup. Takes no parameters.

Returns

None.

Definition at line 47 of file lcd.c.

5.41.2.5 void LCDPosition (uint8_t *row,* uint8_t *col*)

Definition at line 117 of file lcd.c.

5.41.2.6 void LCDPositionNoDelay (uint8_t *row,* uint8_t *col*)

Definition at line 135 of file lcd.c.

5.41.2.7 void LCDRegisterSpecial (uint8_t *number,* uint8_t ∗ *character*)

Definition at line 387 of file lcd.c.

5.41.2.8 void LCDSend (uint8_t *send*)

Definition at line 102 of file lcd.c.

5.41.2.9 void LCDSendChar (uint8_t *txt*)

send single character to LCD.

Parameters

txt	char type data to be sent, 8 bits.

Definition at line 91 of file lcd.c.

5.41.2.10 void LCDSendCmd (uint8_t *cmd*)

Definition at line 77 of file lcd.c.

5.41.2.11 void LCDSendHex (uint8_t ∗ *array*)

Definition at line 314 of file lcd.c.

5.41.2.12 void LCDSendNum (int64_t *num,* uint8_t *length,* uint8_t *isSigned,* uint8_t *showZeros*)

Definition at line 235 of file lcd.c.

5.41.2.13 void LCDSendNumArray (uint8_t ∗ *vector*)

Definition at line 273 of file lcd.c.

5.41.2.14 void LCDSendNumStrict (int64_t *num,* uint8_t *length,* uint8_t *isSigned,* uint8_t *showZeros*)

Definition at line 188 of file lcd.c.

5.41.2.15 void LCDSendString (uint8_t ∗ *string,* uint8_t *breakLine*)

Definition at line 153 of file lcd.c.

5.41.2.16 void LCDSendVU (uint32_t *num,* uint32_t *base*)

Definition at line 434 of file lcd.c.

5.41.2.17 void LCDShift (uint8_t *shift*)

Definition at line 405 of file lcd.c.

5.41.2.18 void numToArray (int32_t *num,* uint8_t ∗ *array,* uint8_t *length,* uint16_t *base*)

Definition at line 347 of file lcd.c.

5.41.3 Variable Documentation

5.41.3.1 LCDStatus LCD0Status

5.42 lcd.h

```
#ifndef lcd_h
#define lcd_h


#include "depl_spc/includeAll_hw.h"
#include "depl_spc/includeAll_sw.h"



#define LCD_splashscreen_row1   PROJECT_NAME            //!< geneartion of project name in LCD
#define LCD_splashscreen_row2   ("rnm sys undvpd")      //!< creator's watermark

#define LCD_splashscreen2_row1  __DATE__                //!< compile date, used as program version
#define LCD_splashscreen2_row2  __TIME__                //!< compile time, used as program version



/*
NEED TO DECLARE

//LCD
#define LCD_RS          J1_05   //E5
#define LCD_RS_Port     GPIO_PORTE_BASE
#define LCD_RS_Pin      GPIO_PIN_5

#define LCD_EN          J1_06   //E4
#define LCD_EN_Port     GPIO_PORTE_BASE
#define LCD_EN_Pin      GPIO_PIN_4

#define LCD_DTA         J2_09   //A2
#define LCD_DTA_Port    GPIO_PORTA_BASE
#define LCD_DTA_Pin     GPIO_PIN_3

#define LCD_CLK         J2_10   //A3
```

```
#define LCD_CLK_Port    GPIO_PORTA_BASE
#define LCD_CLK_Pin     GPIO_PIN_2

#define LCD_row_num     2
#define LCD_col_num     16

LCDStatus LCD0Status;


#define LCD_splashscreen_row1   ("odqd")
#define LCD_splashscreen_row2   ("rnm sys undvpd")



uint8_t specialChar[8][8] = {   //ultima coluna, linha de baixo, reservada para cursor
        0x1F, 0x0F, 0x07, 0x03, 0x01, 0x07, 0x00, 0x00,\
        0x1F, 0x0F, 0x07, 0x03, 0x01, 0x06, 0x01, 0x00,\
        0x1F, 0x0F, 0x07, 0x03, 0x01, 0x05, 0x02, 0x00,\
        0x1F, 0x0F, 0x07, 0x03, 0x01, 0x04, 0x03, 0x00,\
        0x1F, 0x0F, 0x07, 0x03, 0x00, 0x03, 0x04, 0x00,\
        0x1F, 0x0F, 0x07, 0x03, 0x01, 0x02, 0x05, 0x00,\
        0x1F, 0x0F, 0x07, 0x03, 0x01, 0x01, 0x06, 0x00,\
        0x1F, 0x0F, 0x07, 0x03, 0x01, 0x00, 0x07, 0x00
        };



*/




/*
 * LIBRARY FOR LCD USE
 * SERIAL COMM
 * USE OF SHIFT REGISTERS FOR DATA
 */



//sub-function masks
//External Function Masks
#define LCDDelay(x)             SysDelayUs(x)
#define LCDPinSet(x, y)         PinAddrSet(x, y)
#define LCDPinClear(x, y)       PinAddrClear(x,y)



//sub-function masks
#define LCD_RS_High             LCDPinSet(LCD_RS_Port, LCD_RS_Pin)
#define LCD_RS_Low              LCDPinClear(LCD_RS_Port, LCD_RS_Pin)
#define LCD_EN_High             LCDPinSet(LCD_EN_Port, LCD_EN_Pin)
#define LCD_EN_Low              LCDPinClear(LCD_EN_Port, LCD_EN_Pin)
#define LCD_CLK_High            LCDPinSet(LCD_CLK_Port, LCD_CLK_Pin)
#define LCD_CLK_LoW             LCDPinClear(LCD_CLK_Port, LCD_CLK_Pin)
#define LCD_DTA_Send(text)      ShiftSerialSend(LCD_DTA_Port,\
                                LCD_DTA_Pin,\
                                LCD_CLK_Port,\
                                LCD_CLK_Pin, text)


//LCD Command Masks
#define LCD_DISPLAY_ON          0x0C
#define LCD_DISPLAY_OFF         0x08
#define LCD_CURSOR_ON           0x0A
#define LCD_CURSOR_OFF          0x08
#define LCD_BLINK_ON            0x09
#define LCD_BLINK_OFF           0x08
#define LCD_SHIFT               0x10
#define LCD_SHIFT_DISPLAY       0x08
#define LCD_SHIFT_CURSOR        0x02
#define LCD_SHIFT_RIGHT         0x04
#define LCD_SHIFT_LEFT          0x00
#define LCD_SET_CGRAM           0x40
#define LCD_INCREMENT           0X04
#define LCD_INCREMENT_NO_SHIFT  0x00
#define LCD_INCREMENT_SHIFT     0x01
#define LCD_INCREMENT_POSITIVE  0x02
#define LCD_INCREMENT_NEGATIVE  0x00


//LCD Command Initial State - Config
#define LCD_DISPLAY_CONFIG      (LCD_DISPLAY_ON|LCD_CURSOR_OFF|LCD_BLINK_OFF)
#define LCD_DISPLAY_INCREMENT   (LCD_INCREMENT|LCD_INCREMENT_NO_SHIFT)


typedef struct
{
    uint8_t row;
    uint8_t col;
    uint8_t display;
    uint8_t shift;
    uint8_t cgramAdress;
    uint8_t specialChar[8];     //defined by fonts
}LCDStatus;



// \TODO: fix this shit. decide if is to be used with structs or no use at all.
extern LCDStatus   LCD0Status;




//functions declarations
void LCDInit(void);
void LCDSendCmd(uint8_t cmd);
void LCDSendChar(uint8_t txt);
void LCDSend(uint8_t send);
void LCDPosition(uint8_t row, uint8_t col);
void LCDPositionNoDelay(uint8_t row, uint8_t col);
void LCDSendString(uint8_t *string, uint8_t breakLine);
void LCDSendNumStrict(int64_t num, uint8_t length,\
        uint8_t isSigned, uint8_t showZeros);
void LCDSendNum(int64_t num, uint8_t length,\
        uint8_t isSigned, uint8_t showZeros);
void LCDSendNumArray(uint8_t *vector);
void LCDClear(void);
void LCDSendHex(uint8_t *array);

//limited by uint64 max counting
#define maxLengthOut    16
void numToArray(int32_t num, uint8_t *array,\
                uint8_t length, uint16_t base);

void LCDRegisterSpecial(uint8_t number,\
                        uint8_t *character);
void LCDShift(uint8_t shift);
void LCDHome(void);


void arrayToNum(uint8_t *array, uint32_t *num, uint8_t base);


void LCDSendVU(uint32_t num, uint32_t base);

#endif
```

5.43 my_lib/my_use.c File Reference

`#include "my_use.h"`

Functions

- void __inline ShiftSerialSend (uint32_t data_port, uint32_t data_pin, uint32_t clk_port, uint32_t clk_pin, uint8_t text)

5.43.1 Function Documentation

5.43.1.1 void __inline ShiftSerialSend (uint32_t *data_port,* uint32_t *data_pin,* uint32_t *clk_port,* uint32_t *clk_pin,* uint8_t *text*)

Definition at line 9 of file my_use.c.

5.44 my_use.c

```
00001 #include "my_use.h"
00002
```

```
/*
 * Shift Serial Send function
 */
void __inline ShiftSerialSend(uint32_t data_port, uint32_t data_pin, \
                    uint32_t clk_port, uint32_t clk_pin, uint8_t text)
{

    PinAddrClear(clk_port, clk_pin);
    char i=8;
    while(i>0)
    {
        if((text&0x80)==0)
            PinAddrClear(data_port, data_pin);
        else
            PinAddrSet(data_port, data_pin);
        text <<= 1;
        i--;
        PinAddrSet(clk_port, clk_pin);
        SysDelay(2);
        PinAddrClear(clk_port, clk_pin);
        SysDelay(2);
    }

}
```

5.45 my_lib/my_use.h File Reference

```
#include "depl_spc/includeAll_sw.h"
#include "depl_spc/includeAll_hw.h"
```

Macros

- #define bTrue0 0x01
- #define bTrue1 0x02
- #define bTrue2 0x04
- #define bTrue3 0x08
- #define bTrue4 0x10
- #define bTrue5 0x20
- #define bTrue6 0x40
- #define bTrue7 0x80
- #define charDecadeLength 3
- #define charBinaryLength 8
- #define shortDecadeLength 5
- #define shortBinaryLength 16
- #define intDecadeLength 10
- #define intBinaryLength 32
- #define limitCeilValue(value, lim)
- #define limitCycleValueUpZero(value, lim)
- #define limitCycleValueUpOff(value, lim, reset)
- #define limitFloorValue(value, lim)

Functions

- void ShiftSerialSend (uint32_t data_port, uint32_t data_pin, uint32_t clk_port, uint32_t clk_pin, uint8_t text)

5.45.1 Macro Definition Documentation

5.45.1.1 #define bTrue0 0x01

Definition at line 7 of file my_use.h.

5.45.1.2 #define bTrue1 0x02

Definition at line 8 of file my_use.h.

5.45.1.3 #define bTrue2 0x04

Definition at line 9 of file my_use.h.

5.45.1.4 #define bTrue3 0x08

Definition at line 10 of file my_use.h.

5.45.1.5 #define bTrue4 0x10

Definition at line 11 of file my_use.h.

5.45.1.6 #define bTrue5 0x20

Definition at line 12 of file my_use.h.

5.45.1.7 #define bTrue6 0x40

Definition at line 13 of file my_use.h.

5.45.1.8 #define bTrue7 0x80

Definition at line 14 of file my_use.h.

5.45.1.9 #define charBinaryLength 8

Definition at line 17 of file my_use.h.

5.45.1.10 #define charDecadeLength 3

Definition at line 16 of file my_use.h.

5.45.1.11 #define intBinaryLength 32

Definition at line 23 of file my_use.h.

5.45.1.12 #define intDecadeLength 10

Definition at line 22 of file my_use.h.

5.45.1.13 #define limitCeilValue(*value,* *lim*)

Value:

```
if(value>=lim)\
                                                    value=lim;
```

Definition at line 28 of file my_use.h.

5.45.1.14 #define limitCycleValueUpOff(*value,* *lim,* *reset*)

Value:

```
if(value>=lim)\
                                                    value=reset;
```

Definition at line 32 of file my_use.h.

5.45.1.15 #define limitCycleValueUpZero(*value,* *lim*)

Value:

```
if(value>=lim)\
                                                    value=0;
```

Definition at line 30 of file my_use.h.

5.45.1.16 #define limitFloorValue(*value,* *lim*)

Value:

```
if(value<=lim)\
                                                    value=lim;
```

Definition at line 34 of file my_use.h.

5.45.1.17 #define shortBinaryLength 16

Definition at line 20 of file my_use.h.

5.45.1.18 #define shortDecadeLength 5

Definition at line 19 of file my_use.h.

5.45.2 Function Documentation

5.45.2.1 void ShiftSerialSend (uint32_t *data_port,* uint32_t *data_pin,* uint32_t *clk_port,* uint32_t *clk_pin,* uint8_t *text*)

Definition at line 9 of file my_use.c.

5.46 my_use.h

```
#ifndef my_use_h
#define my_use_h

#include "depl_spc/includeAll_sw.h"
#include "depl_spc/includeAll_hw.h"

#define bTrue0              0x01
#define bTrue1              0x02
#define bTrue2              0x04
#define bTrue3              0x08
#define bTrue4              0x10
#define bTrue5              0x20
#define bTrue6              0x40
#define bTrue7              0x80

#define charDecadeLength    3
#define charBinaryLength    8

#define shortDecadeLength   5
#define shortBinaryLength   16

#define intDecadeLength     10
#define intBinaryLength     32



//function masks
#define limitCeilValue(value, lim)                  if(value>=lim)\
                                                    value=lim;
#define limitCycleValueUpZero(value,lim)            if(value>=lim)\
                                                    value=0;
#define limitCycleValueUpOff(value, lim, reset)     if(value>=lim)\
                                                    value=reset;
#define limitFloorValue(value, lim)                 if(value<=lim)\
                                                    value=lim;




/*
 * Function Declarations
 */
void ShiftSerialSend(uint32_t data_port, uint32_t data_pin, \
        uint32_t clk_port, uint32_t clk_pin, uint8_t text);










#endif// my_use_h
```

5.47 my_lib/myUart.c File Reference

```
#include "myUart.h"
```

Functions

- void myUARTSendString (uint32_t instance, uint8_t ∗string)

5.47.1 Function Documentation

5.47.1.1 void myUARTSendString (uint32_t *instance,* uint8_t ∗ *string*)

Definition at line 17 of file myUart.c.

5.48 myUart.c

```
/*
 * myUart.c
 *
 *  Created on: Nov 25, 2013
 *      Author: rikardo
 */


#include "myUart.h"



/*
 * Sends a string, NULL terminator
 * returns none
 */
void myUARTSendString(uint32_t instance, uint8_t *string)
{
    while(*string)
    {
        myUARTSend(instance, *string);
        string++;
        myUARTDelay(1);
    }
}

```

5.49 my_lib/myUart.h File Reference

```
#include "depl_spc/includeAll_sw.h"
#include "depl_spc/includeAll_hw.h"
```

Data Structures

- struct UARTInstance

Macros

- #define UART_NORMAL_OP_MODE 0x0001
- #define UART_DIRECT_TRANSFER_MODE 0x0002
- #define UART_BUFFER_SIZE 30
- #define myUARTPC UART0_BASE
- #define myUARTSend(instance, charToGo) MAP_UARTCharPutNonBlocking(instance, charToGo)
- #define myUARTDelay(delay) SysDelay(delay)
- #define myUARTPCSend(charToGo) myUARTSend(myUARTPC, charToGo)

Functions

- void myUARTSendString (uint32_t instance, uint8_t ∗string)

5.49.1 Macro Definition Documentation

5.49.1.1 #define myUARTDelay(*delay*) SysDelay(delay)

Definition at line 39 of file myUart.h.

5.49.1.2 #define myUARTPC UART0_BASE

Definition at line 36 of file myUart.h.

5.49.1.3 #define myUARTPCSend(*charToGo*) myUARTSend(myUARTPC, charToGo)

Definition at line 40 of file myUart.h.

5.49.1.4 #define myUARTSend(*instance,* *charToGo*) MAP_UARTCharPutNonBlocking(instance, charToGo)

Definition at line 38 of file myUart.h.

5.49.1.5 #define UART_BUFFER_SIZE 30

Definition at line 23 of file myUart.h.

5.49.1.6 #define UART_DIRECT_TRANSFER_MODE 0x0002

Definition at line 19 of file myUart.h.

5.49.1.7 #define UART_NORMAL_OP_MODE 0x0001

Definition at line 18 of file myUart.h.

5.49.2 Function Documentation

5.49.2.1 void myUARTSendString (uint32_t *instance,* uint8_t ∗ *string*)

Definition at line 17 of file myUart.c.

5.50 myUart.h

```
00001 /*
00002  * myUart.h
00003  *
00004  *  Created on: Nov 25, 2013
00005  *      Author: rikardo
00006  */
00007 
00008 #ifndef MYUART_H_
00009 #define MYUART_H_
00010 
00011 #include "depl_spc/includeAll_sw.h"
00012 #include "depl_spc/includeAll_hw.h"
00013 
00014 
00015 /*
00016  * UART operation mode masks
00017  */
00018 #define UART_NORMAL_OP_MODE                 0x0001
00019 #define UART_DIRECT_TRANSFER_MODE           0x0002
00020 
00021 
00022 
00023 #define UART_BUFFER_SIZE    30
00024 
00025 typedef struct{
00026     uint8_t RxBuffer[UART_BUFFER_SIZE];
00027     uint8_t RxBufferPtr;
00028     uint8_t TxBuffer[UART_BUFFER_SIZE];
00029     uint8_t TxBufferPtr;
```

```
    uint16_t Mode;
    uint8_t TxLastSent[UART_BUFFER_SIZE];
    uint8_t TxLastSentPtr;
}UARTInstance;


#define myUARTPC                                    UART0_BASE

#define myUARTSend(instance, charToGo)              MAP_UARTCharPutNonBlocking(instance, charToGo)
#define myUARTDelay(delay)                          SysDelay(delay)
#define myUARTPCSend(charToGo)                      myUARTSend(myUARTPC, charToGo)



void myUARTSendString(uint32_t instance, uint8_t *string);




#endif /* MYUART_H_ */
```

5.51 my_lib/uk_mapping.h File Reference

Macros

- #define UKM_SPACE 32
- #define UKM_BSPACE 127
- #define UKM_BS 8
- #define UKM_ENTER 13
- #define UKM_TILDA 126
- #define UKM_ESCAPE 27
- #define UKM_TAB 9
- #define UKM_CTRL_E 5
- #define UKM_ASCII_TAB 9
- #define UKM_ASCII_LF 10
- #define UKM_LF 10
- #define UKM_LINEFEED 10
- #define UKM_CR 13
- #define UKM_ASCII_VT 11
- #define UKM_VT 11
- #define UKM_ASCII_FF 12
- #define UKM_CLS 12

5.51.1 Macro Definition Documentation

5.51.1.1 #define UKM_ASCII_FF 12

Definition at line 28 of file uk_mapping.h.

5.51.1.2 #define UKM_ASCII_LF 10

Definition at line 22 of file uk_mapping.h.

5.51.1.3 #define UKM_ASCII_TAB 9

Definition at line 21 of file uk_mapping.h.

5.51.1.4 #define UKM_ASCII_VT 11

Definition at line 26 of file uk_mapping.h.

5.51.1.5 #define UKM_BS 8

Definition at line 15 of file uk_mapping.h.

5.51.1.6 #define UKM_BSPACE 127

Definition at line 14 of file uk_mapping.h.

5.51.1.7 #define UKM_CLS 12

Definition at line 29 of file uk_mapping.h.

5.51.1.8 #define UKM_CR 13

Definition at line 25 of file uk_mapping.h.

5.51.1.9 #define UKM_CTRL_E 5

Definition at line 20 of file uk_mapping.h.

5.51.1.10 #define UKM_ENTER 13

Definition at line 16 of file uk_mapping.h.

5.51.1.11 #define UKM_ESCAPE 27

Definition at line 18 of file uk_mapping.h.

5.51.1.12 #define UKM_LF 10

Definition at line 23 of file uk_mapping.h.

5.51.1.13 #define UKM_LINEFEED 10

Definition at line 24 of file uk_mapping.h.

5.51.1.14 #define UKM_SPACE 32

Definition at line 13 of file uk_mapping.h.

5.51.1.15 #define UKM_TAB 9

Definition at line 19 of file uk_mapping.h.

5.51.1.16 #define UKM_TILDA 126

Definition at line 17 of file uk_mapping.h.

5.51.1.17 #define UKM_VT 11

Definition at line 27 of file uk_mapping.h.

5.52 uk_mapping.h

```
00001 /*
00002  * uart_keyboard_mapping.h
00003  *
00004  *  Created on: Nov 27, 2013
00005  *      Author: rikardo
00006  */
00007
00008 #ifndef UART_KEYBOARD_MAPPING_H_
00009 #define UART_KEYBOARD_MAPPING_H_
00010
00011
00012
00013 #define UKM_SPACE           32
00014 #define UKM_BSPACE          127
00015 #define UKM_BS              8
00016 #define UKM_ENTER           13
00017 #define UKM_TILDA           126
00018 #define UKM_ESCAPE          27
00019 #define UKM_TAB             9
00020 #define UKM_CTRL_E          5
00021 #define UKM_ASCII_TAB       9
00022 #define UKM_ASCII_LF        10
00023 #define UKM_LF              10
00024 #define UKM_LINEFEED        10
00025 #define UKM_CR              13
00026 #define UKM_ASCII_VT        11
00027 #define UKM_VT              11
00028 #define UKM_ASCII_FF        12
00029 #define UKM_CLS             12
00030
00031
00032
00033 #endif /* UART_KEYBOARD_MAPPING_H_ */
```

5.53 README.md File Reference

5.54 README.md

```
00001 uIntPLib
00002 ========
00003
00004 Universal Integrated Peripheral Library
00005
00006 This is a library made with functions masks to medium level programming.
00007 Intended to make code more portable, while maintaning its performance.
```

Index