RBELib

2.0

Generated by Doxygen 1.8.5

Fri Jul 18 2014 12:55:32

Contents

Chapter 1

3001 RBELib

1.1 Welcome to RBE 3001

Here you will find all the documentation that you need for working with RBELib which will contain function prototypes for everything that you will need throughout the course. While it is possible to complete the course without using RBELib, using it will make your code easier to maintain as you get closer to the final project as well as follow proper coding practice as outlined in the syllabus.

As you progress through the course, it is encouraged that you keep your own version of RBELib with your SVN repository so that you can add to things such as the To Do list and function descriptions as you go.

1.2 RBELib Files

ADC **Pages 233 - 252**

Here you will find everything relating to the Analog to Digital Converter (ADC) that you should need. You will have to make your own corresponding functions for these prototypes.

Pages 65 - 85	
Port A	Port B
Port C	Port D

Port A/B/C/D contains the code for manipulating the ports on the chip that you can access via the breakout board. Keep in mind that some of these are also connected to components on the board such as the SPI line, if you have a problem using a pin and on-board components, **CHECK THE DATASHEET AND BOARD LAYOUT** to make sure that you are not using the same pin for your buttons that you are for the MOSI while using SPI or etc..

Printing

This contains all the printing statements that you should need for debugging your code. Whenever you have a problem and can't find out what it is, first try to print out all of your code and add long delays so that you can see what is happening.

You do NOT need to make any functions for printing, it is all done for you!

Peripherals **See respective datasheets for each peripheral.**

Here is where all code for peripheral devices such as the IR sensor and accelerometer go. You will have to make your own functions for these prototypes.

PID

This is where all PID code goes for the calculation. You will need to create your own calculation using the formulas that you used in class and optimize them for running on and embedded system.

RBELib Macros

Here are all of the includes for RBELib as well as some of the macros that you may find useful to use (such as INPUT/OUTPUT instead of 0/1). You do not need to create anything additional.

Reg Structs

Reg Strucs contain a lot of useful shorthand notations that you can use for specifying bits when manipulating registers. Each register has the datasheet page you can look at to find out more information of what each of the settings do. You do not need to create anything additional.

Servo (Conveyer/Gripper)

This is used for moving the conveyer belt and opening/closing the gripper. You do not need to create anything additional.

SPI **Pages 154 - 163**

For initializing the SPI and and sending data through it. You need to create the functions for the prototypes.

Timers **Pages 86 - 153 (Timers 0-2)**

Allows you to initialize any one of the timers and set their comparative values for when they reset. You need to create the functions for the prototypes.

USART **Pages 164 - 190 (In SPI mode, pages 191 - 199)**

This should be the very first thing that you work on. This allows you to use serial printing within your code through the use of the USART. You do need to create your functions for the prototypes.

Potentiometers

These functions can be used to get the potentiometer values in degrees, voltage, or ADC counts. You need to make your own functions for the prototypes.

1.3 Other Helpful Links

Bug List

This is a place where any bugs in RBELib and your code should be documented.

To Do

Things that still need to be done in the RBELib and or your code.

Data Types

This lets you know the number of bytes in any given data type.

Chapter 2

Data Types

2.1 1 Byte

char

2.2 2 Bytes

short

int

2.3 4 Bytes

long

float

double

long double

2.4 8 Bytes

long long

Chapter 3

Todo List

Global calcPID (int setPoint, int actPos, char link)

Make a function to calculate the PID value for a link.

Global changeADC (unsigned char channel)

Create a way to switch ADC channels.

Global ClearADC (unsigned char channel)

Create the corresponding function to clear the last ADC calculation register.

Global DebugUSARTInit (unsigned long baudrate)

Create the function that will initialize the USART for debugging use.

Global encCount (int chan)

Make a function to find the encoder counts on a given channel.

Global encInit (char chan)

Make a function that can setup both encoder chips on the board.

Global GenericSPIInit (void)

Create the function that will allow you to initialize the SPI.

Global getAccel (char axis)

Create a function that is able to find the acceleration of a given axis.

Global GetADC (unsigned char channel)

Create the corresponding function to obtain the value of the last calculation.

Global gotoAngles (char lowerTheta, char upperTheta)

Make a way to drive the links to a desired angle.

Global gotoXY (int x, int y)

Use kinematic equations to move the end effector to the desired position.

Global InitADC (unsigned char channel)

Create the corresponding function to initialize the ADC using the channel parameter.

Global InitTimer (unsigned char number, unsigned char mode, unsigned short int comp)

Create a function that initializes the desired timer in a given mode and set the compare value.

Global IRDist (char chan)

Make a function that is able to get the ADC value of the IR sensor.

Global potADC (unsigned char pot)

Create a function to get the ADC value of the potentiometer.

Global potVolts (unsigned char pot)

Create a function to get the voltage of the potentiometer.

Global putCharDebug (char byteToSend)

Make the function that will put a character on the USART line.

Global setCompValue (unsigned char number, unsigned short int comp)

Create a function that will set the compare value for the given time.

Global setConst (unsigned char link, float Kp, float Ki, float Kd)

Create a function to the the PID constants for a given link/

Global setDAC (unsigned char DACn, unsigned int SPIVal)

Make the function that is able to set the DAC to a given value from 0 - 4095.

Global SPITransceive (BYTE data)

Make a function that will send a byte of data through the SPI.

Global stopMotors ()

Create way to stop the motors using the DAC.

Global USART_Receive (void)

Make the function that will listen for input on the USART.

Chapter 4

Bug List

File SPI.h

Some SS lines may need to be toggled low -> high -> low (or high -> low -> high) due to some bug with the AVR. When enabling/disabling SS lines, toggle them to makes sure that they work if you encounter any problems.

Chapter 5

Data Structure Index

5.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

Chapter 7

Data Structure Documentation

7.1 __8bitreg_t Union Reference

Generic byte register.

`#include <ports.h>`

Data Fields

- struct {
  unsigned _P0:1
  unsigned _P1:1
  unsigned _P2:1
  unsigned _P3:1
  unsigned _P4:1
  unsigned _P5:1
  unsigned _P6:1
  unsigned _P7:1
  };

- struct {
  unsigned _w:8
  };

7.1.1 Detailed Description

Generic byte register.

Definition at line 14 of file ports.h.

7.1.2 Field Documentation

7.1.2.1 struct { ... }

7.1.2.2 struct { ... }

7.1.2.3 unsigned __8bitreg_t::_P0

Definition at line 16 of file ports.h.

7.1.2.4 unsigned __8bitreg_t::_P1

Definition at line 17 of file ports.h.

7.1.2.5 unsigned __8bitreg_t::_P2

Definition at line 18 of file ports.h.

7.1.2.6 unsigned __8bitreg_t::_P3

Definition at line 19 of file ports.h.

7.1.2.7 unsigned __8bitreg_t::_P4

Definition at line 20 of file ports.h.

7.1.2.8 unsigned __8bitreg_t::_P5

Definition at line 21 of file ports.h.

7.1.2.9 unsigned __8bitreg_t::_P6

Definition at line 22 of file ports.h.

7.1.2.10 unsigned __8bitreg_t::_P7

Definition at line 23 of file ports.h.

7.1.2.11 unsigned __8bitreg_t::_w

Definition at line 26 of file ports.h.

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

- include/RBELib/ports.h

7.2 __SPIPORTbits_t Union Reference

SPI port.

```
#include <ports.h>
```

Data Fields

- struct {
    unsigned __pad0__:5
    unsigned _MOSI:1
    unsigned _MISO:1
    unsigned _SCK:1
  };

- struct {
  unsigned _w:8
  };

7.2.1 Detailed Description

SPI port.

Definition at line 33 of file ports.h.

7.2.2 Field Documentation

7.2.2.1 struct { ... }

7.2.2.2 struct { ... }

7.2.2.3 unsigned __SPIPORTbits_t::__pad0__

Definition at line 35 of file ports.h.

7.2.2.4 unsigned __SPIPORTbits_t::_MISO

Definition at line 37 of file ports.h.

7.2.2.5 unsigned __SPIPORTbits_t::_MOSI

Definition at line 36 of file ports.h.

7.2.2.6 unsigned __SPIPORTbits_t::_SCK

Definition at line 38 of file ports.h.

7.2.2.7 unsigned __SPIPORTbits_t::_w

Definition at line 41 of file ports.h.

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

- include/RBELib/ports.h

7.3 pidConst Struct Reference

PID constants.

`#include <PID.h>`

Data Fields

- float Kp_H
  Upper link Kp.
- float Ki_H

Upper link Ki.

- float Kd_H

  Upper link Kd.

- float Kp_L

  Lower link Kp.

- float Ki_L

  Lower link Ki.

- float Kd_L

  Lower link Kd.

7.3.1 Detailed Description

PID constants.

Obtain value using:

```
pidConst.Kp_H;
```

for the value desired.

Definition at line 18 of file PID.h.

7.3.2 Field Documentation

7.3.2.1 float pidConst::Kd_H

Upper link Kd.

Definition at line 30 of file PID.h.

7.3.2.2 float pidConst::Kd_L

Lower link Kd.

Definition at line 42 of file PID.h.

7.3.2.3 float pidConst::Ki_H

Upper link Ki.

Definition at line 26 of file PID.h.

7.3.2.4 float pidConst::Ki_L

Lower link Ki.

Definition at line 38 of file PID.h.

7.3.2.5 float pidConst::Kp_H

Upper link Kp.

Definition at line 22 of file PID.h.

7.3.2.6 float pidConst::Kp_L

Lower link Kp.

Definition at line 34 of file PID.h.

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

- include/RBELib/PID.h

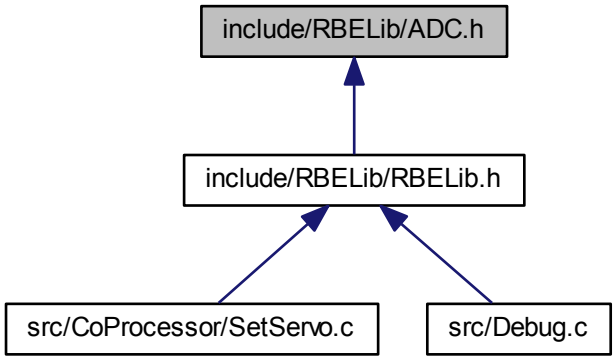
Chapter 8

File Documentation

8.1 include/RBELib/ADC.h File Reference

The header file and function prototypes for the ADC.

This graph shows which files directly or indirectly include this file:



Functions

- void InitADC (unsigned char channel)

  Initializes one and only one channel of the ADC.
- void ClearADC (unsigned char channel)

  Disables ADC functionality from one and only one pin.
- unsigned short GetADC (unsigned char channel)

  Run a conversion on and get the analog value from one ADC channel if using polling.
- void changeADC (unsigned char channel)

  Change the channel the ADC is sampling if using interrupts.

8.1.1 Detailed Description

The header file and function prototypes for the ADC. For single ended conversion, the ADC value an be found from the voltage using:

$$\frac{V_{in} * 1024}{V_{ref}}$$

Author

Kevin Harrington

Date

February 11, 2010

Author

Justin Barrett

Date

August 23, 2011

Author

Eric Willcox

Date

August 19, 2013

Definition in file ADC.h.

8.1.2 Function Documentation

8.1.2.1 void changeADC (unsigned char *channel*)

Change the channel the ADC is sampling if using interrupts.

Parameters

channel	The ADC channel to switch to.

Todo Create a way to switch ADC channels.

8.1.2.2 void ClearADC (unsigned char *channel*)

Disables ADC functionality from one and only one pin.

Parameters

channel	The ADC channel to disable.

Returns

void

Todo Create the corresponding function to clear the last ADC calculation register.

8.1.2.3 unsigned short GetADC (unsigned char *channel*)

Run a conversion on and get the analog value from one ADC channel if using polling.

Parameters

channel	The ADC channel to run a conversion on.

Returns

unsigned short value The 10-bit value returned by the ADC conversion.

Todo Create the corresponding function to obtain the value of the last calculation.

8.1.2.4 void InitADC (unsigned char *channel*)

Initializes one and only one channel of the ADC.

Parameters

channel	The ADC channel to initialize.

Returns

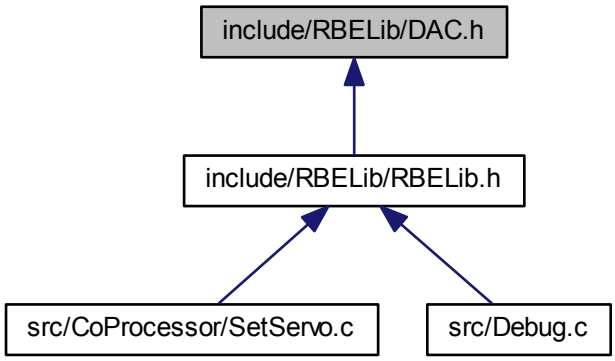
void

Todo Create the corresponding function to initialize the ADC using the channel parameter.

8.2 include/RBELib/DAC.h File Reference

The header file and function prototypes for the DAC.

This graph shows which files directly or indirectly include this file:



Functions

- void setDAC (unsigned char DACn, unsigned int SPIVal)
  Set the DAC to the given value on the chosen channel.

8.2.1 Detailed Description

The header file and function prototypes for the DAC. Use these functions to control the behavior of the DAC.

Author

Eric Willcox

Date

August 21, 2013

Definition in file DAC.h.

8.2.2 Function Documentation

8.2.2.1 void setDAC (unsigned char *DACn,* unsigned int *SPIVal*)

Set the DAC to the given value on the chosen channel.

Parameters

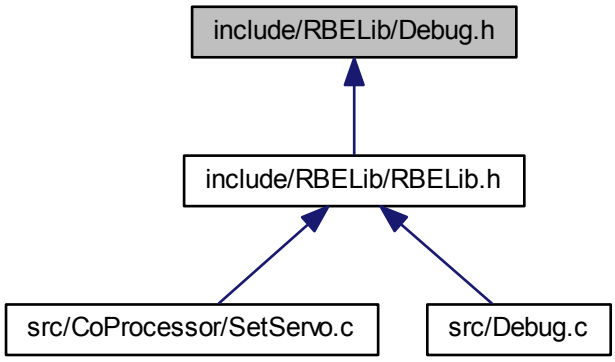
DACn	The channel that you want to set.
SPIVal	The value you want to set it to.

Todo Make the function that is able to set the DAC to a given value from 0 - 4095.

8.3 include/RBELib/Debug.h File Reference

Allows for the use of printf() on the 3001 board. Simply rebinds stdout to the UART.

This graph shows which files directly or indirectly include this file:



Functions

- int printfRBE (char var, FILE ∗stream)

  Calls the students function 'putCharDebug()' to output the stream to. You should not call this function directly, instead use the standard function printf().
- void initRBELib ()

  Rebinds stdout to call printfRBE() and initializes communication with the coprocessor. This function should be called once at the start of your code once you have putCharDebug() written in Lab 1. If you do not call this, printf() and SetServo() will not work.

8.3.1 Detailed Description

Allows for the use of printf() on the 3001 board. Simply rebinds stdout to the UART.

Author

Eric Willcox

Date

July 9, 2014

Definition in file Debug.h.

8.3.2 Function Documentation

8.3.2.1 void initRBELib ( )

Rebinds stdout to call printfRBE() and initializes communication with the coprocessor. This function should be called once at the start of your code once you have putCharDebug() written in Lab 1. If you do not call this, printf() and SetServo() will not work.

Definition at line 20 of file Debug.c.

References InitAltCom().

Here is the call graph for this function:

8.3.2.2 int printfRBE (char *var*, FILE * *stream*)

Calls the students function 'putCharDebug()' to output the stream to. You should not call this function directly, instead use the standard function printf().

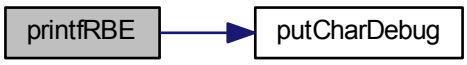
Parameters

var	Character to output
**stream*	Place to put the character (stdout)

Definition at line 14 of file Debug.c.

References putCharDebug().

Here is the call graph for this function:



8.4 include/RBELib/doxy_pages/datatypes.h File Reference

8.4.1 Detailed Description

Date

Aug 28, 2013

Author

Eric Willcox

Definition in file datatypes.h.

8.5 include/RBELib/doxy_pages/index.h File Reference

This is the index file for Doxygen..

8.5.1 Detailed Description

This is the index file for Doxygen..

Author

Kevin Harrington

Date

February 21, 2010

Author

Eric Willcox

Date

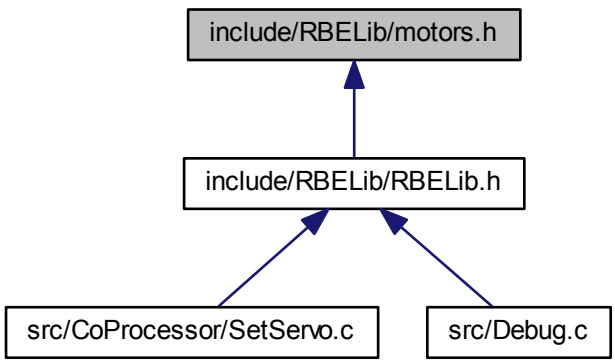
August 20, 2013

Definition in file index.h.

8.6 include/RBELib/motors.h File Reference

Motor driving functions for the arm.

This graph shows which files directly or indirectly include this file:



Functions

- void stopMotors ()

  Helper function to stop the motors on the arm.
- void gotoAngles (char lowerTheta, char upperTheta)

  Drive the arm to a desired angle.
- void gotoXY (int x, int y)

  Drive the end effector of the arm to a desired X and Y position in the workspace.

8.6.1 Detailed Description

Motor driving functions for the arm.

Author

Eric Willcox

Date

July 9, 2014

Definition in file motors.h.

8.6.2 Function Documentation

8.6.2.1 void gotoAngles (char *lowerTheta,* char *upperTheta*)

Drive the arm to a desired angle.

Parameters

lowerTheta	The desired angle for the lower link.
upperTheta	The desired angle for the upper link.

Todo Make a way to drive the links to a desired angle.

8.6.2.2 void gotoXY (int *x,* int *y*)

Drive the end effector of the arm to a desired X and Y position in the workspace.

Parameters

x	The desired x position for the end effector.
y	The desired y position for the end effector.

Todo Use kinematic equations to move the end effector to the desired position.

8.6.2.3 void stopMotors ( )

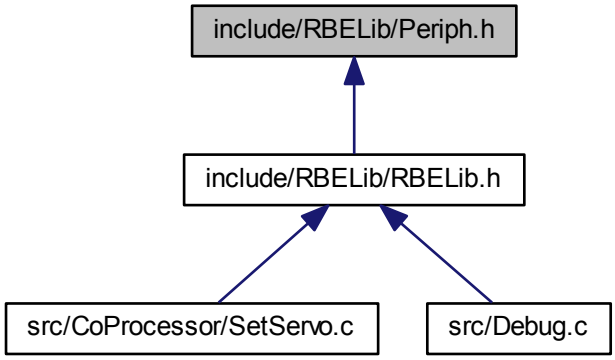
Helper function to stop the motors on the arm.

Todo Create way to stop the motors using the DAC.

8.7 include/RBELib/Periph.h File Reference

The header file and function prototypes for the peripherals (IR, encoder and accelerometer).

This graph shows which files directly or indirectly include this file:



Functions

- int getAccel (char axis)
  Find the acceleration in the given axis (X, Y, Z).
- int IRDist (char chan)
  Read an IR sensor and calculate the distance of the block.
- void encInit (char chan)
  Initialize the encoders with the desired settings.
- unsigned long encCount (int chan)
  Finds the current count of one of the encoders.

8.7.1 Detailed Description

The header file and function prototypes for the peripherals (IR, encoder and accelerometer). Each of these functions is for controlling the peripheral devices of the arm.

Author

Eric Willcox

Date

August 21, 2013

Definition in file Periph.h.

8.7.2 Function Documentation

8.7.2.1 unsigned long encCount (int *chan*)

Finds the current count of one of the encoders.

Parameters

chan	Channel that the encoder is on that you would like to read.

Returns

count The current count of the encoder.

Todo Make a function to find the encoder counts on a given channel.

8.7.2.2 void encInit (char *chan*)

Initialize the encoders with the desired settings.

Parameters

chan	Channel to initialize.

Todo Make a function that can setup both encoder chips on the board.

8.7.2.3 int getAccel (char *axis*)

Find the acceleration in the given axis (X, Y, Z).

Parameters

axis	The axis that you want to get the measurement of (x, y, z).

Returns

gVal Value of acceleration.

Todo Create a function that is able to find the acceleration of a given axis.

8.7.2.4 int IRDist (char *chan*)

Read an IR sensor and calculate the distance of the block.

Parameters

chan	The port that the IR sensor is on.

Returns

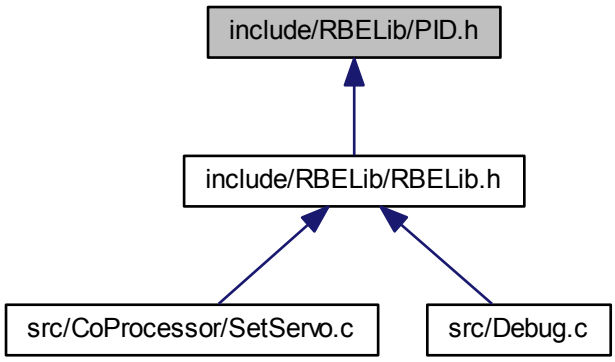
value The distance the block is from the sensor.

Todo Make a function that is able to get the ADC value of the IR sensor.

8.8 include/RBELib/PID.h File Reference

The header file for PID constants and calculations.

This graph shows which files directly or indirectly include this file:



Data Structures

- struct pidConst

  PID constants.

Functions

- void setConst (unsigned char link, float Kp, float Ki, float Kd)

  Sets the Kp, Ki, and Kd values for 1 link.
- int calcPID (int setPoint, int actPos, char link)

  Calculate the PID value.

Variables

- pidConst pidConsts

  Declaration for use in other files.

8.8.1 Detailed Description

The header file for PID constants and calculations. Sets the PID constants and calculate the PID value.

Author

Eric Willcox

Date

August 17 2013

Definition in file PID.h.

8.8.2 Function Documentation

8.8.2.1 int calcPID (int *setPoint,* int *actPos,* char *link*)

Calculate the PID value.

Parameters

setPoint	The desired position of the link.
actPos	The current position of the link.
link	Which link to calculate the error for.

Returns

void

Todo Make a function to calculate the PID value for a link.

8.8.2.2 void setConst (unsigned char *link*, float *Kp*, float *Ki*, float *Kd*)

Sets the Kp, Ki, and Kd values for 1 link.

to set the values, use the following style

```
*  pidConst.Kp = 1.3;
*
```

Parameters

link	The link you want to set the values for (H or L).
Kp	Proportional value.
Ki	Integral value.
Kd	Derivative value.

Returns

void

Todo Create a function to the the PID constants for a given link/

8.8.3 Variable Documentation

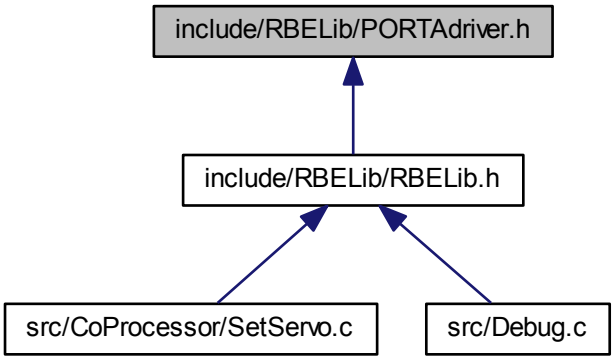
8.8.3.1 pidConst pidConsts

Declaration for use in other files.

8.9 include/RBELib/PORTAdriver.h File Reference

The header file and function prototypes for PORTA.

This graph shows which files directly or indirectly include this file:



Functions

- void SetPORTADDR (unsigned char direction, unsigned char pin)
  Sets the direction (Input/Output) of the specified pin on PORTA.
- unsigned char GetPORTAPinValue (unsigned char pin)
  Returns the value on the specified pin of PORTA.
- void SetPORTAPinValue (unsigned char pin, unsigned char value)
  Sets the value on the specified pin of PORTA.

8.9.1 Detailed Description

The header file and function prototypes for PORTA.

Author

Kevin Harrington

Date

August 30, 2010

Author

Justin Barrett

Date

August 23, 2011

Author

Eric Willcox

Date

August 19, 2013

Definition in file PORTAdriver.h.

8.9.2 Function Documentation

8.9.2.1 unsigned char GetPORTAPinValue (unsigned char *pin*)

Returns the value on the specified pin of PORTA.

Parameters

pin	The pin on PORTA to retrieve the value of.

Returns

value The value of the specified pin on PORTA.

8.9.2.2 void SetPORTADDR (unsigned char *direction,* unsigned char *pin*)

Sets the direction (Input/Output) of the specified pin on PORTA.

Parameters

direction	The desired direction for the pin (Input or Output).
pin	The pin on PORTA to set the direction of.

Returns

void

8.9.2.3 void SetPORTAPinValue (unsigned char *pin,* unsigned char *value*)

Sets the value on the specified pin of PORTA.

Parameters

pin	The pin on PORTA to set the value of.
value	The value to set the specified pin on PORTA to.

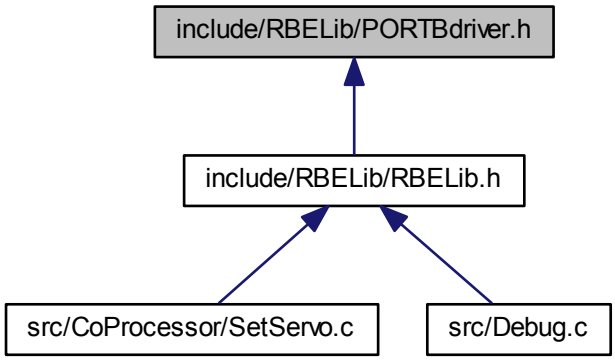
Returns

void

8.10 include/RBELib/PORTBdriver.h File Reference

The header file and function prototypes for PORTB.

This graph shows which files directly or indirectly include this file:



Functions

- void SetPORTBDDR (unsigned char direction, unsigned char pin)

  Sets the direction (Input/Output) of the specified pin on PORTB.
- unsigned char GetPORTBPinValue (unsigned char pin)

  Returns the value on the specified pin of PORTB.
- void SetPORTBPinValue (unsigned char pin, unsigned char value)

  Sets the value on the specified pin of PORTB.

8.10.1 Detailed Description

The header file and function prototypes for PORTB.

Author

Kevin Harrington

Date

August 30, 2010

Author

Justin Barrett

Date

August 23, 2011

Author

Eric Willcox

Date

August 19, 2013

Definition in file PORTBdriver.h.

8.10.2 Function Documentation

8.10.2.1 unsigned char GetPORTBPinValue (unsigned char *pin*)

Returns the value on the specified pin of PORTB.

Parameters

pin	The pin on PORTB to retrieve the value of.

Returns

value The value of the specified pin on PORTB.

8.10.2.2 void SetPORTBDDR (unsigned char *direction,* unsigned char *pin*)

Sets the direction (Input/Output) of the specified pin on PORTB.

Parameters

direction	The desired direction for the pin (Input or Output).
pin	The pin on PORTB to set the direction of.

Returns

void

8.10.2.3 void SetPORTBPinValue (unsigned char *pin,* unsigned char *value*)

Sets the value on the specified pin of PORTB.

Parameters

pin	The pin on PORTB to set the value of.
value	The value to set the specified pin on PORTB to.

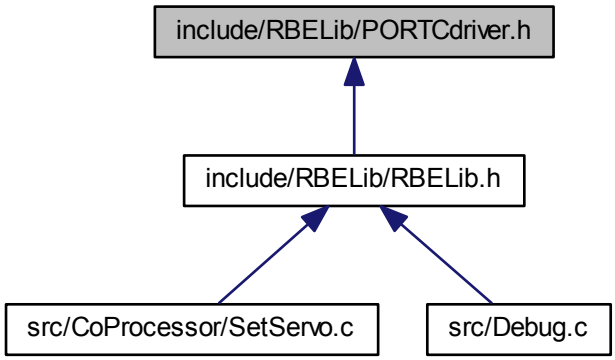
Returns

void

8.11 include/RBELib/PORTCdriver.h File Reference

The header file and function prototypes for PORTC.

This graph shows which files directly or indirectly include this file:



Functions

- void SetPORTCDDR (unsigned char direction, unsigned char pin)

  Sets the direction (Input/Output) of the specified pin on PORTC.
- unsigned char GetPORTCPinValue (unsigned char pin)

  Returns the value on the specified pin of PORTC.
- void SetPORTCPinValue (unsigned char pin, unsigned char value)

  Sets the value on the specified pin of PORTC.

8.11.1 Detailed Description

The header file and function prototypes for PORTC.

Author

Kevin Harrington

Date

August 30, 2010

Author

Justin Barrett

Date

August 23, 2011

Author

Eric Willcox

Date

August 19, 2013

Definition in file PORTCdriver.h.

8.11.2 Function Documentation

8.11.2.1 unsigned char GetPORTCPinValue (unsigned char *pin*)

Returns the value on the specified pin of PORTC.

Parameters

pin	The pin on PORTC to retrieve the value of.

Returns

value The value of the specified pin on PORTC.

8.11.2.2 void SetPORTCDDR (unsigned char *direction,* unsigned char *pin*)

Sets the direction (Input/Output) of the specified pin on PORTC.

Parameters

direction	The desired direction for the pin (Input or Output).
pin	The pin on PORTC to set the direction of.

Returns

void

8.11.2.3 void SetPORTCPinValue (unsigned char *pin,* unsigned char *value*)

Sets the value on the specified pin of PORTC.

Parameters

pin	The pin on PORTC to set the value of.
value	The value to set the specified pin on PORTC to.

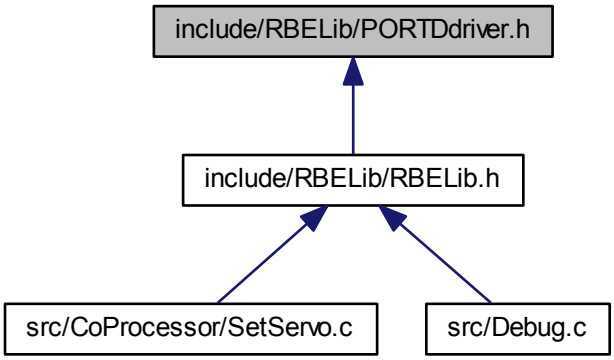
Returns

void

8.12 include/RBELib/PORTDdriver.h File Reference

The header file and function prototypes for PORTD.

This graph shows which files directly or indirectly include this file:



Functions

- void SetPORTDDDR (unsigned char direction, unsigned char pin)
  Sets the direction (Input/Output) of the specified pin on PORTD.
- unsigned char GetPORTDPinValue (unsigned char pin)
  Returns the value on the specified pin of PORTD.
- void SetPORTDPinValue (unsigned char pin, unsigned char value)
  Sets the value on the specified pin of PORTD.

8.12.1 Detailed Description

The header file and function prototypes for PORTD.

Author

Kevin Harrington

Date

August 30, 2010

Author

Justin Barrett

Date

August 23, 2011

Author

Eric Willcox

Date

August 19, 2013

Definition in file PORTDdriver.h.

8.12.2 Function Documentation

8.12.2.1 unsigned char GetPORTDPinValue (unsigned char *pin*)

Returns the value on the specified pin of PORTD.

Parameters

pin	The pin on PORTD to retrieve the value of.

Returns

value The value of the specified pin on PORTD.

8.12.2.2 void SetPORTDDDR (unsigned char *direction,* unsigned char *pin*)

Sets the direction (Input/Output) of the specified pin on PORTD.

Parameters

direction	The desired direction for the pin (Input or Output).
pin	The pin on PORTD to set the direction of.

Returns

void

8.12.2.3 void SetPORTDPinValue (unsigned char *pin,* unsigned char *value*)

Sets the value on the specified pin of PORTD.

Parameters

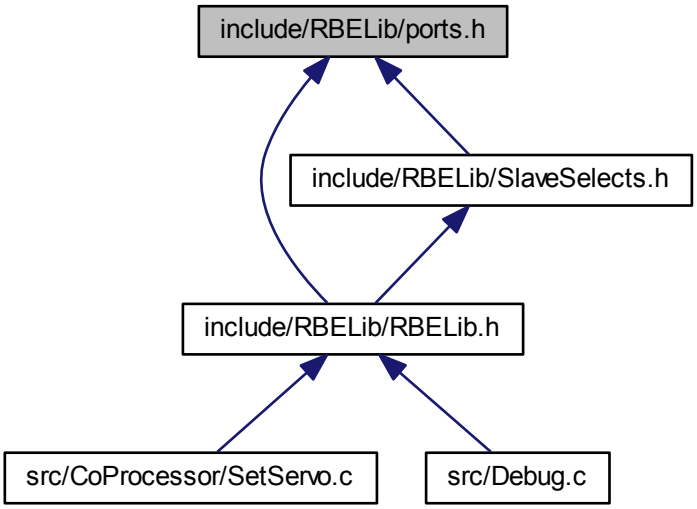
pin	The pin on PORTD to set the value of.
value	The value to set the specified pin on PORTD to.

Returns

void

8.13 include/RBELib/ports.h File Reference

This graph shows which files directly or indirectly include this file:



Data Structures

- union __8bitreg_t
  Generic byte register.
- union __SPIPORTbits_t
  SPI port.

Functions

- volatile __8bitreg_t PINAbits __asm__ ("0x20") __attribute__((section("sfr")))
- volatile __8bitreg_t DDRAbits __asm__ ("0x21") __attribute__((section("sfr")))
- volatile __8bitreg_t PORTAbits __asm__ ("0x22") __attribute__((section("sfr")))
- volatile __8bitreg_t PINBbits __asm__ ("0x23") __attribute__((section("sfr")))
- volatile __8bitreg_t DDRBbits __asm__ ("0x24") __attribute__((section("sfr")))
- volatile __8bitreg_t PORTBbits __asm__ ("0x25") __attribute__((section("sfr")))
- volatile __8bitreg_t PINCbits __asm__ ("0x26") __attribute__((section("sfr")))
- volatile __8bitreg_t DDRCbits __asm__ ("0x27") __attribute__((section("sfr")))
- volatile __8bitreg_t PORTCbits __asm__ ("0x28") __attribute__((section("sfr")))
- volatile __8bitreg_t PINDbits __asm__ ("0x29") __attribute__((section("sfr")))
- volatile __8bitreg_t DDRDbits __asm__ ("0x2A") __attribute__((section("sfr")))
- volatile __8bitreg_t PORTDbits __asm__ ("0x2B") __attribute__((section("sfr")))

8.13.1 Function Documentation

8.13.1.1 volatile __8bitreg_t PINAbits __asm__ ("0x20")

8.13.1.2 volatile __8bitreg_t DDRAbits __asm__ ("0x21")

8.13.1.3 volatile __8bitreg_t PORTAbits __asm__ ("0x22")

8.13.1.4 volatile __8bitreg_t PINBbits __asm__ ("0x23")

8.13.1.5 volatile __SPIPORTbits_t SPIDDRbits __asm__ ("0x24")

8.13.1.6 volatile __SPIPORTbits_t SPIPORTbits __asm__ ("0x25")

8.13.1.7 volatile __8bitreg_t PINCbits __asm__ ("0x26")

8.13.1.8 volatile __8bitreg_t DDRCbits __asm__ ("0x27")

8.13.1.9 volatile __8bitreg_t PORTCbits __asm__ ("0x28")

8.13.1.10 volatile __8bitreg_t PINDbits __asm__ ("0x29")

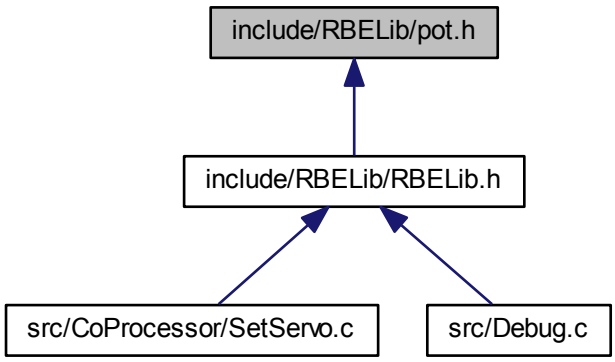
8.13.1.11 volatile __8bitreg_t DDRDbits __asm__ ("0x2A")

8.13.1.12 volatile __8bitreg_t PORTDbits __asm__ ("0x2B")

8.14 include/RBELib/pot.h File Reference

The header file and function prototypes for the potentiometers.

This graph shows which files directly or indirectly include this file:



Functions

- int potADC (unsigned char pot)

  Find the ADC value of the given potentiometer.

- int potVolts (unsigned char pot)

  Find the voltage value of the given potentiometer.

8.14.1 Detailed Description

The header file and function prototypes for the potentiometers. Use these functions to read the values from the pots.

Author

Eric Willcox

Date

August 17 2013

Definition in file pot.h.

8.14.2 Function Documentation

8.14.2.1 int potADC (unsigned char *pot*)

Find the ADC value of the given potentiometer.

Parameters

pot	The pot to get the value of.

Returns

adcVal Value of potentiometer.

Todo Create a function to get the ADC value of the potentiometer.

8.14.2.2 int potVolts (unsigned char *pot*)

Find the voltage value of the given potentiometer.

Parameters

pot	The pot to get the value of.

Returns

volts Voltage of potentiometer.

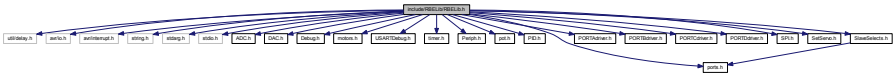
Todo Create a function to get the voltage of the potentiometer.

8.15 include/RBELib/RBELib.h File Reference

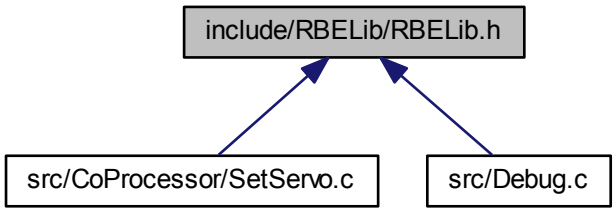
This is a meta-header. It includes all the other header files that are needed for RBELib as well as some macros.

```
#include <util/delay.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <string.h>
#include <stdarg.h>
#include <stdio.h>
#include "ADC.h"
#include "DAC.h"
#include "Debug.h"
#include "motors.h"
#include "USARTDebug.h"
#include "timer.h"
#include "Periph.h"
#include "pot.h"
#include "PID.h"
#include "ports.h"
#include "PORTAdriver.h"
#include "PORTBdriver.h"
#include "PORTCdriver.h"
#include "PORTDdriver.h"
#include "SPI.h"
#include "SetServo.h"
#include "SlaveSelects.h"
```

Include dependency graph for RBELib.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define OUTPUT 1
- #define INPUT 0
- #define ON 1
- #define OFF 0
- #define HIGH 1
- #define LOW 0
- #define TRUE 1

- #define FALSE 0
- #define BOOL unsigned char
- #define BYTE unsigned char
- #define UINT32 unsigned long
- #define INT32 long
- #define UINT16 unsigned short int
- #define WORD unsigned short int

8.15.1 Detailed Description

This is a meta-header. It includes all the other header files that are needed for RBELib as well as some macros.

Author

Kevin Harrington

Date

February 21, 2010

Author

Justin Barrett

Date

August 23, 2011

Author

Eric Willcox

Date

August 19, 2013

Definition in file RBELib.h.

8.15.2 Macro Definition Documentation

8.15.2.1 #define BOOL unsigned char

Definition at line 39 of file RBELib.h.

8.15.2.2 #define BYTE unsigned char

Definition at line 40 of file RBELib.h.

8.15.2.3 #define FALSE 0

Definition at line 36 of file RBELib.h.

8.15.2.4 #define HIGH 1

Definition at line 33 of file RBELib.h.

8.15.2.5 #define INPUT 0

Definition at line 30 of file RBELib.h.

8.15.2.6 #define INT32 long

Definition at line 42 of file RBELib.h.

8.15.2.7 #define LOW 0

Definition at line 34 of file RBELib.h.

8.15.2.8 #define OFF 0

Definition at line 32 of file RBELib.h.

8.15.2.9 #define ON 1

Definition at line 31 of file RBELib.h.

8.15.2.10 #define OUTPUT 1

Definition at line 29 of file RBELib.h.

8.15.2.11 #define TRUE 1

Definition at line 35 of file RBELib.h.

8.15.2.12 #define UINT16 unsigned short int

Definition at line 43 of file RBELib.h.

8.15.2.13 #define UINT32 unsigned long

Definition at line 41 of file RBELib.h.

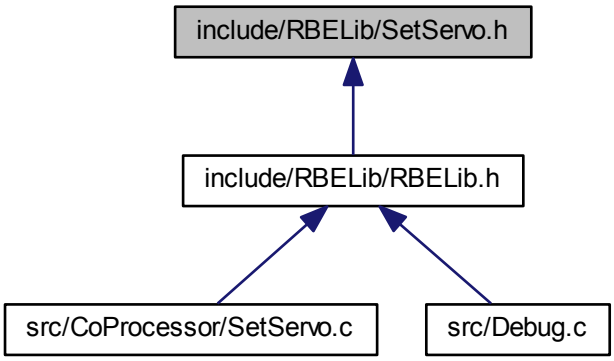
8.15.2.14 #define WORD unsigned short int

Definition at line 44 of file RBELib.h.

8.16 include/RBELib/SetServo.h File Reference

This file allows for using the SerSevo function to move the conveyor and gripper.

This graph shows which files directly or indirectly include this file:



Macros

- #define CLK 18432000

Functions

- void SetServo (int Pin, int Value)

  The communication to set servo values simply.
- void InitAltCom (unsigned long baudrate)

  Used to initialize UART1 for communication with the coprocessor. It should never be called manually.
- void setCharDebug (char byteToSend)

  Used to put a char on UART1. It should never be called manually.
- void dprintfDEBUG_NNL (char ∗str)

  String to send to the coprocessor via UART1 with no new line.

8.16.1 Detailed Description

This file allows for using the SerSevo function to move the conveyor and gripper.

Author

cwrus

Date

Jun 28, 2012

Author

Eric Willcox

Date

July 9, 2014

Definition in file SetServo.h.

8.16.2 Macro Definition Documentation

8.16.2.1 #define CLK 18432000

Definition at line 16 of file SetServo.h.

8.16.3 Function Documentation

8.16.3.1 void dprintfDEBUG_NNL (char * *str*)

String to send to the coprocessor via UART1 with no new line.

Parameters

**str*	String to send.

Returns

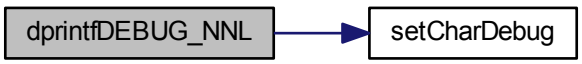
void

Definition at line 70 of file SetServo.c.

References setCharDebug().

Referenced by SetServo().

Here is the call graph for this function:



8.16.3.2 void InitAltCom (unsigned long *baudrate*)

Used to initialize UART1 for communication with the coprocessor. It should never be called manually.

Parameters

baudrate	Baud rate of the communication line in bps.

Returns

void

Definition at line 27 of file SetServo.c.

Referenced by initRBELib().

8.16.3.3 void setCharDebug (char *byteToSend*)

Used to put a char on UART1. It should never be called manually.

Parameters

byteToSend	Character to send

Returns

void

Definition at line 56 of file SetServo.c.

Referenced by dprintfDEBUG_NNL().

8.16.3.4 void SetServo (int *Pin,* int *Value*)

The communication to set servo values simply.

Parameters

Pin	Pin number.
Value	Value to set the pin to.

Returns

void

Definition at line 16 of file SetServo.c.

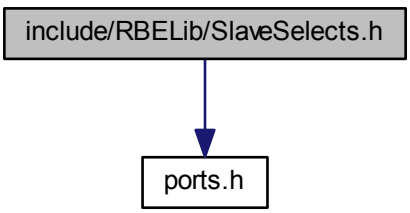
References dprintfDEBUG_NNL().

Here is the call graph for this function:

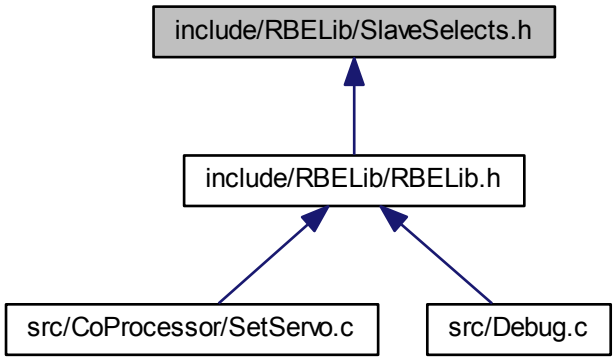
8.17 include/RBELib/SlaveSelects.h File Reference

```
#include "ports.h"
```

Include dependency graph for SlaveSelects.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define SPI_MISO PORTBbits._P6
  SPI MISO.
- #define SPI_MISO_DDR DDRBbits._P6
  SPI MISO.
- #define SPI_MOSI PORTBbits._P5
  SPI MOSI.
- #define SPI_MOSI_DDR DDRBbits._P5
  SPI MOSI.
- #define SPI_SCK PORTBbits._P7
  SPI Clock.
- #define SPI_SCK_DDR DDRBbits._P7
  SPI Clock.
- #define SPI_MASTER_SS DDRBbits._P4
  SPI master SS.
- #define ENCODER_SS_0 PORTCbits._P5
  SPI Slave Select Encoder 0.
- #define ENCODER_SS_1 PORTCbits._P4
  SPI Slave Select Encoder 1.
- #define ENCODER_SS_2 PORTCbits._P3
  SPI Slave Select Encoder 2.
- #define ENCODER_SS_3 PORTCbits._P2
  SPI Slave Select Encoder 3.
- #define DAC_SS PORTDbits._P4
  SPI Slave Select DAC 1.
- #define AUX_DAC_SS PORTCbits._P6
  SPI Slave Select DAC 2.
- #define COPROC_SS PORTCbits._P1
  SPI Slave Co-Processor.
- #define COPROC_BUSY PINCbits._P0

SPI Slave Select Busy Co-Processor.

- #define SPARE_SS PORTCbits._P0

  SPI Slave Select Unuesd.
- #define ENCODER_SS_0_ddr DDRCbits._P5

  SPI Slave Select Encoder 0.
- #define ENCODER_SS_1_ddr DDRCbits._P4

  SPI Slave Select Encoder 1.
- #define ENCODER_SS_2_ddr DDRCbits._P3

  SPI Slave Select Encoder 2.
- #define ENCODER_SS_3_ddr DDRCbits._P2

  SPI Slave Select Encoder 3.
- #define DAC_SS_ddr DDRDbits._P4

  SPI Slave Select DAC 1.
- #define AUX_DAC_SS_ddr DDRCbits._P6

  SPI Slave Select DAC 2.
- #define COPROC_SS_ddr DDRCbits._P1

  SPI Slave Select Co-Processor.
- #define COPROC_BUSY_ddr DDRCbits._P0

  SPI Slave Select busy Co-Processor.
- #define SPARE_SS_ddr DDRCbits._P0

  SPI Slave Select Unused.
- #define ENCODER_IRQ PORTBbitd._P2

  Interrupt line for all encoders.
- #define ENCODER_IRQ_ddr DDRBbitd._P2

  Interrupt line for all encoders.

8.17.1 Macro Definition Documentation

8.17.1.1 #define AUX_DAC_SS PORTCbits._P6

SPI Slave Select DAC 2.

Definition at line 67 of file SlaveSelects.h.

8.17.1.2 #define AUX_DAC_SS_ddr DDRCbits._P6

SPI Slave Select DAC 2.

Definition at line 105 of file SlaveSelects.h.

8.17.1.3 #define COPROC_BUSY PINCbits._P0

SPI Slave Select Busy Co-Processor.

Definition at line 75 of file SlaveSelects.h.

8.17.1.4 #define COPROC_BUSY_ddr DDRCbits._P0

SPI Slave Select busy Co-Processor.

Definition at line 113 of file SlaveSelects.h.

8.17.1.5 #define COPROC_SS PORTCbits._P1

SPI Slave Co-Processor.

Definition at line 71 of file SlaveSelects.h.

8.17.1.6 #define COPROC_SS_ddr DDRCbits._P1

SPI Slave Select Co-Processor.

Definition at line 109 of file SlaveSelects.h.

8.17.1.7 #define DAC_SS PORTDbits._P4

SPI Slave Select DAC 1.

Definition at line 63 of file SlaveSelects.h.

8.17.1.8 #define DAC_SS_ddr DDRDbits._P4

SPI Slave Select DAC 1.

Definition at line 101 of file SlaveSelects.h.

8.17.1.9 #define ENCODER_IRQ PORTBbitd._P2

Interrupt line for all encoders.

Definition at line 122 of file SlaveSelects.h.

8.17.1.10 #define ENCODER_IRQ_ddr DDRBbitd._P2

Interrupt line for all encoders.

Definition at line 126 of file SlaveSelects.h.

8.17.1.11 #define ENCODER_SS_0 PORTCbits._P5

SPI Slave Select Encoder 0.

Definition at line 46 of file SlaveSelects.h.

8.17.1.12 #define ENCODER_SS_0_ddr DDRCbits._P5

SPI Slave Select Encoder 0.

Definition at line 84 of file SlaveSelects.h.

8.17.1.13 #define ENCODER_SS_1 PORTCbits._P4

SPI Slave Select Encoder 1.

Definition at line 50 of file SlaveSelects.h.

8.17.1.14 #define ENCODER_SS_1_ddr DDRCbits._P4

SPI Slave Select Encoder 1.

Definition at line 88 of file SlaveSelects.h.

8.17.1.15 #define ENCODER_SS_2 PORTCbits._P3

SPI Slave Select Encoder 2.

Definition at line 54 of file SlaveSelects.h.

8.17.1.16 #define ENCODER_SS_2_ddr DDRCbits._P3

SPI Slave Select Encoder 2.

Definition at line 92 of file SlaveSelects.h.

8.17.1.17 #define ENCODER_SS_3 PORTCbits._P2

SPI Slave Select Encoder 3.

Definition at line 58 of file SlaveSelects.h.

8.17.1.18 #define ENCODER_SS_3_ddr DDRCbits._P2

SPI Slave Select Encoder 3.

Definition at line 96 of file SlaveSelects.h.

8.17.1.19 #define SPARE_SS PORTCbits._P0

SPI Slave Select Unuesd.

Definition at line 79 of file SlaveSelects.h.

8.17.1.20 #define SPARE_SS_ddr DDRCbits._P0

SPI Slave Select Unused.

Definition at line 117 of file SlaveSelects.h.

8.17.1.21 #define SPI_MASTER_SS DDRBbits._P4

SPI master SS.

Definition at line 42 of file SlaveSelects.h.

8.17.1.22 #define SPI_MISO PORTBbits._P6

SPI MISO.

Definition at line 16 of file SlaveSelects.h.

8.17.1.23 #define SPI_MISO_DDR DDRBbits._P6

SPI MISO.

Definition at line 20 of file SlaveSelects.h.

8.17.1.24 #define SPI_MOSI PORTBbits._P5

SPI MOSI.

Definition at line 25 of file SlaveSelects.h.

8.17.1.25 #define SPI_MOSI_DDR DDRBbits._P5

SPI MOSI.

Definition at line 29 of file SlaveSelects.h.

8.17.1.26 #define SPI_SCK PORTBbits._P7

SPI Clock.

Definition at line 34 of file SlaveSelects.h.

8.17.1.27 #define SPI_SCK_DDR DDRBbits._P7

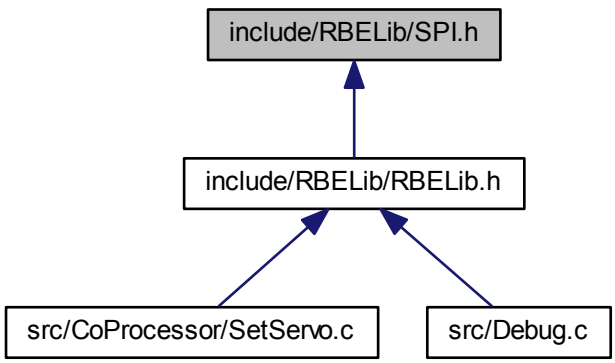
SPI Clock.

Definition at line 38 of file SlaveSelects.h.

8.18 include/RBELib/SPI.h File Reference

The header file and function prototypes for the SPI.

This graph shows which files directly or indirectly include this file:



Functions

- void GenericSPIInit (void)

  Initializes the SPI bus for a standard configuration.
- BYTE SPITransceive (BYTE data)

  Checks to see that the SPI is able to transmit and transmits a byte without changing any other configurations.

8.18.1 Detailed Description

The header file and function prototypes for the SPI.

Bug Some SS lines may need to be toggled low -> high -> low (or high -> low -> high) due to some bug with the AVR. When enabling/disabling SS lines, toggle them to makes sure that they work if you encounter any problems.

Author

kamiro87

Date

August 31, 2010

Author

Justin Barrett

Date

August 23, 2011

Author

Eric Willcox

Date

August 20, 2013

Definition in file SPI.h.

8.18.2 Function Documentation

8.18.2.1 void GenericSPIInit (void)

Initializes the SPI bus for a standard configuration.

Returns

void

Todo Create the function that will allow you to initialize the SPI.

8.18.2.2 BYTE SPITransceive (BYTE *data*)

Checks to see that the SPI is able to transmit and transmits a byte without changing any other configurations.

Parameters

data	The byte to send down the SPI bus.

Returns

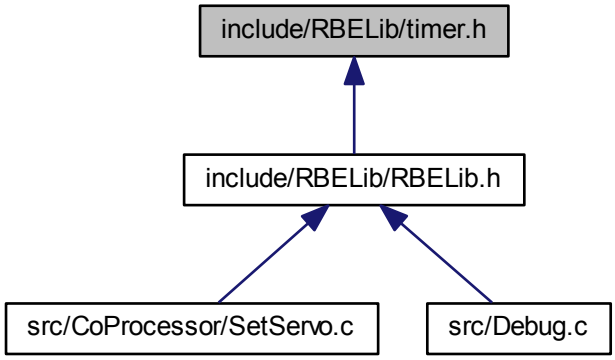
value The byte shifted in during transmit (SPI MUST send AND receive data, even if either is junk data).

Todo Make a function that will send a byte of data through the SPI.

8.19 include/RBELib/timer.h File Reference

The header file and function prototypes for Timers 0-2.

This graph shows which files directly or indirectly include this file:



Macros

- #define NORMAL 0
- #define CTC 1

Functions

- void InitTimer (unsigned char number, unsigned char mode, unsigned short int comp)

  Initializes the specified timer in the specified mode. This header file provides #defines for NORMAL operation mode and CTC operation mode, however there are many more modes of operation for the Timers that can be experimented with.
- void setCompValue (unsigned char number, unsigned short int comp)

  Only used when the specified timer is in CTC mode. Changes the value of the comparison register (OCRnA) to the new specified value.

8.19.1 Detailed Description

The header file and function prototypes for Timers 0-2.

Author

Justin Barrett

Date

August 25, 2011

Author

Eric Willcox

Date

August 20, 2013

Definition in file timer.h.

8.19.2 Macro Definition Documentation

8.19.2.1 #define CTC 1

Runs the timer in Clear Timer on Compare (CTC) mode.

Definition at line 22 of file timer.h.

8.19.2.2 #define NORMAL 0

Runs the timer in Normal mode.

Definition at line 18 of file timer.h.

8.19.3 Function Documentation

8.19.3.1 void InitTimer (unsigned char *number,* unsigned char *mode,* unsigned short int *comp*)

Initializes the specified timer in the specified mode. This header file provides #defines for NORMAL operation mode and CTC operation mode, however there are many more modes of operation for the Timers that can be experimented with.

Parameters

number	The number of the timer to be initialized (0-2).
mode	The mode to initialize the specified timer in (see defines above).
comp	Only used in CTC mode. The value that the timer counts to before it triggers an interrupt and resets.

Returns

void

Todo Create a function that initializes the desired timer in a given mode and set the compare value.

8.19.3.2 void setCompValue (unsigned char *number,* unsigned short int *comp*)

Only used when the specified timer is in CTC mode. Changes the value of the comparison register (OCRnA) to the new specified value.

Parameters

number	The number of the timer to change (0-2).
comp	The value to set the OCRnA register to.

Returns

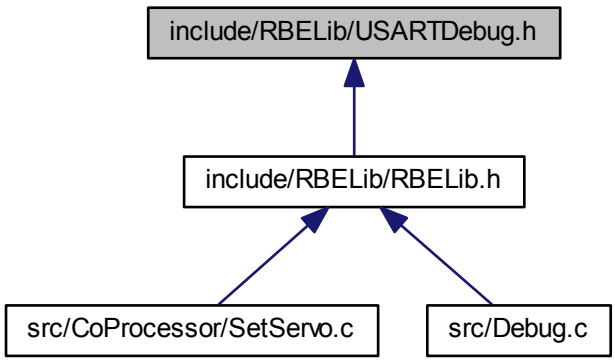
void

Todo Create a function that will set the compare value for the given time.

8.20 include/RBELib/USARTDebug.h File Reference

The header file and function prototypes for the USART.

This graph shows which files directly or indirectly include this file:



Macros

- #define DEFAULT_BAUD 115200

Functions

- void DebugUSARTInit (unsigned long baudrate)

  Initializes USART1 as a print terminal to the PC. This function must check the incoming baudrate against the valid baudrates from the data-sheet. If the baudrate is invalid, then the DEFAULT_BAUD (see above) must be used instead.
- void putCharDebug (char byteToSend)

  Sends one byte to the USART1 Tx pin (Transmits one byte). PREREQUISITE: DebugUSARTInit must be called first.
- unsigned char USART_Receive (void)

  Recieves one byte of data from the serial port (from the PC).

8.20.1 Detailed Description

The header file and function prototypes for the USART.

Author

Kevin Harrington

Date

August 20, 2010

Author

Justin Barrett

Date

August 25, 2011

Author

Eric Willcox

Date

August 20, 2013

Definition in file USARTDebug.h.

8.20.2 Macro Definition Documentation

8.20.2.1 #define DEFAULT_BAUD 115200

Definition at line 19 of file USARTDebug.h.

8.20.3 Function Documentation

8.20.3.1 void DebugUSARTInit (unsigned long *baudrate*)

Initializes USART1 as a print terminal to the PC. This function must check the incoming baudrate against the valid baudrates from the data-sheet. If the baudrate is invalid, then the DEFAULT_BAUD (see above) must be used instead.

Parameters

baudrate	The desired baudrate to set for USART1.

Returns

void

Todo Create the function that will initialize the USART for debugging use.

8.20.3.2 void putCharDebug (char *byteToSend*)

Sends one byte to the USART1 Tx pin (Transmits one byte). PREREQUISITE: DebugUSARTInit must be called first.

Parameters

byteToSend	The byte that is to be transmitted through USART1.

Returns

void

Todo Make the function that will put a character on the USART line.

Referenced by printfRBE().

8.20.3.3 unsigned char USART_Receive (void)

Recieves one byte of data from the serial port (from the PC).

Returns

charRec Character that was received on the USART.

Todo Make the function that will listen for input on the USART.

8.21 src/CoProcessor/SetServo.c File Reference

```
#include "RBELib/RBELib.h"
```

Include dependency graph for SetServo.c:

Functions

- void SetServo (int pin, int value)

  The communication to set servo values simply.
- void InitAltCom (unsigned long baudrate)

  Used to initialize UART1 for communication with the coprocessor. It should never be called manually.
- void setCharDebug (char byteToSend)

  Used to put a char on UART1. It should never be called manually.
- void dprintfDEBUG_NNL (char ∗str)

  String to send to the coprocessor via UART1 with no new line.

8.21.1 Function Documentation

8.21.1.1 void dprintfDEBUG_NNL (char ∗ *str*)

String to send to the coprocessor via UART1 with no new line.

Parameters

∗str	String to send.

Returns

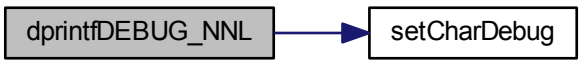
void

Definition at line 70 of file SetServo.c.

References setCharDebug().

Referenced by SetServo().

Here is the call graph for this function:



8.21.1.2 void InitAltCom (unsigned long *baudrate*)

Used to initialize UART1 for communication with the coprocessor. It should never be called manually.

Parameters

baudrate	Baud rate of the communication line in bps.

Returns

void

Definition at line 27 of file SetServo.c.

Referenced by initRBELib().

8.21.1.3 void setCharDebug (char *byteToSend*)

Used to put a char on UART1. It should never be called manually.

Parameters

byteToSend	Character to send

Returns

void

Definition at line 56 of file SetServo.c.

Referenced by dprintfDEBUG_NNL().

8.21.1.4 void SetServo (int *Pin,* int *Value*)

The communication to set servo values simply.

Parameters

Pin	Pin number.
Value	Value to set the pin to.

Returns

void

Definition at line 16 of file SetServo.c.

References dprintfDEBUG_NNL().

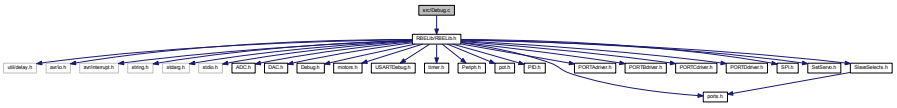
Here is the call graph for this function:

8.22 src/Debug.c File Reference

Allows for printf() and SetServo() capability.

```
#include "RBELib/RBELib.h"
```

Include dependency graph for Debug.c:



Functions

- int printfRBE (char var, FILE *stream)

  Calls the students function 'putCharDebug()' to output the stream to. You should not call this function directly, instead use the standard function printf().

- void initRBELib ()

  Rebinds stdout to call printfRBE() and initializes communication with the coprocessor. This function should be called once at the start of your code once you have putCharDebug() written in Lab 1. If you do not call this, printf() and SetServo() will not work.

8.22.1 Detailed Description

Allows for printf() and SetServo() capability.

Author

Eric Willcox

Date

July 9, 2014

Definition in file Debug.c.

8.22.2 Function Documentation

8.22.2.1 void initRBELib ( )

Rebinds stdout to call printfRBE() and initializes communication with the coprocessor. This function should be called once at the start of your code once you have putCharDebug() written in Lab 1. If you do not call this, printf() and SetServo() will not work.

Definition at line 20 of file Debug.c.

References InitAltCom().

Here is the call graph for this function:

8.22.2.2 int printfRBE (char *var*, FILE * *stream*)

Calls the students function 'putCharDebug()' to output the stream to. You should not call this function directly, instead use the standard function printf().

Parameters

var	Character to output
**stream*	Place to put the character (stdout)

Definition at line 14 of file Debug.c.

References putCharDebug().

Here is the call graph for this function: