Border Survillence Bot

Group V

Generated by Doxygen 1.7.4

Sat Apr 16 2011 23:30:54

Contents

Chapter 1

File Index

1.1 File List

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

Chapter 2

File Documentation

2.1 /Users/harish/Downloads/cs308-group05/C-code/main.c File Reference

```
#include "winavr_firebird.h"
```

Functions

- void init_devices ()
- int main (void)

Variables

- unsigned char **data**

2.1.1 Detailed Description

Written by: Group 5:

HARISH 8005052 SAMEER 8005056 RAJESH 8005041 PRADEEP 8005044

AVR Studio Version 4.17, Build 666

Application example: Robot control over serial port

Concepts covered: serial communication

Serial Port used: UART1

There are two components to the motion control: 1. Direction control using pins PORTA0 to PORTA3 2. Velocity control by PWM on pins PL3 and PL4 using OC5A and OC5B.

In this experiment for the simplicity PL3 and PL4 are kept at logic 1.

Pins for PWM are kept at logic 1.

Connection Details:

Motion control: L-1---->PA0; L-2---->PA1; R-1---->PA2; R-2---->PA3; PL3 (OC5A) ----> Logic 1; PL4 (OC5B) ----> Logic 1;

Serial Communication: PORTD 2 --> RXD1 UART1 receive for RS232 serial communication PORTD 3 --> TXD1 UART1 transmit for RS232 serial communication

PORTH 0 --> RXD2 UART 2 receive for USB - RS232 communication PORTH 1 --> TXD2 UART 2 transmit for USB - RS232 communication

PORTE 0 --> RXD0 UART0 receive for ZigBee wireless communication PORTE 1 --> TXD0 UART0 transmit for ZigBee wireless communication

PORTJ 0 --> RXD3 UART3 receive available on microcontroller expainsion board PORTJ 1 --> TXD3 UART3 transmit available on microcontroller expainsion board

Serial communication baud rate: 9600bps

This experiment enables the user to control the robot motion through Serial Comunication from the PC Wirelessly.

Byte Commands for respective direction are as Follows:

0x51 -----> FORWARD 0x52 -----> BACKWARD 0x53 -----> LEFT 0x54 -----> Right 0x50 -----> Stop

Note:

1. Make sure that in the configuration options following settings are done for proper operation of the code

Microcontroller: atmega2560 Frequency: 11059200 Optimization: -O0 (For more information read section: Selecting proper optimization options below figure 4.22 in the hardware manual)

2. Difference between the codes for RS232 serial, USB and wireless communication is only in the serial port number. Rest of the things are the same.

3. For USB communication check the Jumper 1 position on the ATMEGA2560 microcontroller adaptor board

****

the author needs to be taken.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Software released under Creative Commence cc by-nc-sa licence. For legal information refer to: http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

Definition in file main.c.

2.1.2 Function Documentation

2.1.2.1 void init_devices ( )

Function To Initialize All The Devices

Initializes all the ports

Initailize UART0 for serial communiaction

timer0 interrupt sources

timer1 interrupt sources

timer2 interrupt sources

timer3 interrupt sources

timer4 interrupt sources

timer5 interrupt sources

Enables the global interrupt

Definition at line 120 of file main.c.

2.1.2.2 int main (void)

Main Function

Definition at line 143 of file main.c.

2.2 /Users/harish/Downloads/cs308-group05/C-code/winavr_firebird.h File Reference

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
```

Defines

- #define **FCPU** 11059200ul

Functions

- void motion_pin_config ()
- void left_encoder_pin_config (void)

  Function to configure INT4 (PORTE 4) pin as input for the left position encoder.
- void right_encoder_pin_config (void)

  Function to configure INT5 (PORTE 5) pin as input for the right position encoder.
- void left_position_encoder_interrupt_init (void)

  Interrupt 4 enable.
- void right_position_encoder_interrupt_init (void)
- void port_init ()

  Function to Initialize PORTS.
- ISR (INT5_vect)

  ISR for right position encoder.
- ISR (INT4_vect)

  TSR for left position encoder.
- void motion_set (unsigned char Direction)

  Function used for setting motor's direction.
- void forward (void)

  both wheels forward
- void back (void)

  both wheels backward
- void left (void)

  Left wheel backward, Right wheel forward.
- void right (void)

  Left wheel forward, Right wheel backward.
- void **stop** (void)
- void timer5_init ()
- void velocity (unsigned char left_motor, unsigned char right_motor)

  Function for velocity control.
- void angle_rotate (unsigned int Degrees)

Function used for turning robot by specified degrees.

- void linear_distance_mm (unsigned int DistanceInMM)

  Function used for moving robot forward by specified distance.
- void forward_mm (unsigned int DistanceInMM)

  move forward by specified distance
- void back_mm (unsigned int DistanceInMM)

  move backward by specified distance
- void left_degrees (unsigned int Degrees)

  rotate left by specified degrees
- void right_degrees (unsigned int Degrees)

  rotate right by specified degrees
- void buzzer_pin_config (void)

  Function to configure the buzzer.
- void buzzer_on (void)

  Function to switch the buzzer on.
- void buzzer_off (void)

  Function to switch the buzzer off.
- void uart1_init (void)
- SIGNAL (SIG_USART1_RECV)

  ISR for receive complete interrupt.

Variables

- unsigned long int ShaftCountLeft = 0

  to keep track of left position encoder
- unsigned long int ShaftCountRight = 0

  to keep track of right position encoder
- unsigned int Degrees

  to accept angle in degrees for turning
- unsigned char data

  to receive data through serial communication

2.2.1 Detailed Description

Written by: Group 5:

HARISH 8005052 SAMEER 8005056 RAJESH 8005041 PRADEEP 8005044

AVR Studio Version 4.17, Build 666

Application example: Robot control over serial port

Concepts covered: serial communication

Serial Port used: UART1

There are two components to the motion control: 1. Direction control using pins PORTA0 to PORTA3 2. Velocity control by PWM on pins PL3 and PL4 using OC5A and OC5B.

In this experiment for the simplicity PL3 and PL4 are kept at logic 1.

Pins for PWM are kept at logic 1.

Connection Details:

Motion control: L-1---->PA0; L-2---->PA1; R-1---->PA2; R-2---->PA3; PL3 (OC5A) ----> Logic 1; PL4 (OC5B) ----> Logic 1;

Serial Communication: PORTD 2 --> RXD1 UART1 receive for RS232 serial communication PORTD 3 --> TXD1 UART1 transmit for RS232 serial communication

PORTH 0 --> RXD2 UART 2 receive for USB - RS232 communication PORTH 1 --> TXD2 UART 2 transmit for USB - RS232 communication

PORTE 0 --> RXD0 UART0 receive for ZigBee wireless communication PORTE 1 --> TXD0 UART0 transmit for ZigBee wireless communication

PORTJ 0 --> RXD3 UART3 receive available on microcontroller expainsion board PORTJ 1 --> TXD3 UART3 transmit available on microcontroller expainsion board

Serial communication baud rate: 9600bps

This experiment enables the user to control the robot motion through Serial Comunication from the PC Wirelessly.

Byte Commands for respective direction are as Follows:

0x51 -----> FORWARD 0x52 -----> BACKWARD 0x53 -----> LEFT 0x54 -----> Right 0x50 -----> Stop

Note:

1. Make sure that in the configuration options following settings are done for proper operation of the code

Microcontroller: atmega2560 Frequency: 11059200 Optimization: -O0 (For more information read section: Selecting proper optimization options below figure 4.22 in the hardware manual)

2. Difference between the codes for RS232 serial, USB and wireless communication is only in the serial port number. Rest of the things are the same.

3. For USB communication check the Jumper 1 position on the ATMEGA2560 microcontroller adaptor board

vided with the distribution.

Neither the name of the copyright holders nor the names of contributors may be used to endorse or promote products derived from this software without specific prior written permission.

Source code can be used for academic purpose. For commercial use permission form the author needs to be taken.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Software released under Creative Commence cc by-nc-sa licence. For legal information refer to: http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

Definition in file winavr_firebird.h.

2.2.2 Function Documentation

2.2.2.1 void angle_rotate (unsigned int *Degrees*)

Function used for turning robot by specified degrees.

division by resolution to get shaft count

Definition at line 270 of file winavr_firebird.h.

2.2.2.2 void buzzer_pin_config (void)

Function to configure the buzzer.

Setting PORTC 3 as output

Setting PORTC 3 logic low to turnoff buzzer

Definition at line 349 of file winavr_firebird.h.

2.2.2.3 ISR (INT5_vect)

ISR for right position encoder.

increment right shaft position count

Definition at line 182 of file winavr_firebird.h.

2.2.2.4 ISR (INT4_vect)

TSR for left position encoder.

increment left shaft position count

Definition at line 189 of file winavr_firebird.h.

2.2.2.5 void left_degrees (unsigned int *Degrees*)

rotate left by specified degrees

88 pulses for 360 degrees rotation 4.090 degrees per count

Definition at line 332 of file winavr_firebird.h.

2.2.2.6 void left_encoder_pin_config (void)

Function to configure INT4 (PORTE 4) pin as input for the left position encoder.

Set the direction of the PORTE 4 pin as input

Enable internal pullup for PORTE 4 pin

Definition at line 139 of file winavr_firebird.h.

2.2.2.7 void left_position_encoder_interrupt_init (void)

Interrupt 4 enable.

Clears the global interrupt

INT4 is set to trigger with falling edge

Enable Interrupt INT4 for left position encoder

Enables the global interrupt

Definition at line 153 of file winavr_firebird.h.

2.2.2.8 void linear_distance_mm (unsigned int *DistanceInMM*)

Function used for moving robot forward by specified distance.

division by resolution to get shaft count

Definition at line 294 of file winavr_firebird.h.

2.2.2.9 void motion_pin_config ( )

Function To Initialize Ports

Motion control pins set as output

Inital value of the motion control pins set to 0

Setting PL3 and PL4 pins as output for PWM generation

Setting PL3 and PL4 pins as logic 1

Definition at line 129 of file winavr_firebird.h.

2.2.2.10 void motion_set (unsigned char *Direction*)

Function used for setting motor's direction.

removing upper nibbel for the protection

reading the PORTA original status

making lower direction nibbel to 0

adding lower nibbel for forward command and restoring the PORTA status

executing the command

Definition at line 196 of file winavr_firebird.h.

2.2.2.11 void right_degrees (unsigned int *Degrees*)

rotate right by specified degrees

88 pulses for 360 degrees rotation 4.090 degrees per count

Definition at line 341 of file winavr_firebird.h.

2.2.2.12 void right_encoder_pin_config (void)

Function to configure INT5 (PORTE 5) pin as input for the right position encoder.

Set the direction of the PORTE 4 pin as input

Enable internal pullup for PORTE 4 pin

Definition at line 146 of file winavr_firebird.h.

2.2.2.13 void right_position_encoder_interrupt_init (void)

Clears the global interrupt

INT5 is set to trigger with falling edge

Enable Interrupt INT5 for right position encoder

Enables the global interrupt

Definition at line 162 of file winavr_firebird.h.

2.2.2.14 SIGNAL (SIG_USART1_RECV)

ISR for receive complete interrupt.

making copy of data from UDR in data variable

echo data back to PC

forward

back

left

right

stop

buzzer

Definition at line 389 of file winavr_firebird.h.

2.2.2.15 void timer5_init ( )

Timer 5 initialised in PWM mode for velocity control Prescale:64 PWM 8bit fast, TOP=0x00FF Timer Frequency:674.988Hz

Stop

Counter higher 8-bit value to which OCR5xH value is compared with

Counter lower 8-bit value to which OCR5xH value is compared with

Output compare register high value for Left Motor

Output compare register low value for Left Motor

Output compare register high value for Right Motor

Output compare register low value for Right Motor

Output compare register high value for Motor C1

Output compare register low value for Motor C1

{COM5A1=1, COM5A0=0; COM5B1=1, COM5B0=0; COM5C1=1 COM5C0=0} For Overriding normal port functionalit to OCRnA outputs. {WGM51=0, WGM50=1} Along With WGM52 in TCCR5B for Selecting FAST PWM 8-bit Mode

WGM12=1; CS12=0, CS11=1, CS10=1 (Prescaler=64)

Definition at line 242 of file winavr_firebird.h.

2.2.2.16 void uart1_init (void)

UART1 initialization desired baud rate:9600 actual baud rate:9600 (0.0%) char size: 8 bit parity: Disabled

disable while setting baud rate

set baud rate lo

set baud rate hi

Definition at line 378 of file winavr_firebird.h.

Index