# Autonomous Defensive Robot

Generated by Doxygen 1.7.4

Tue Apr 5 2011 21:47:44

# Contents

# Chapter 1

# File Index

## 1.1 File List

Here is a list of all files with brief descriptions:

# Chapter 2

# File Documentation

## 2.1 group12.c File Reference

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

### Functions

- void buzzer_pin_config (void)
- void port_init ()
- void buzzer_on (void)
- void buzzer_off (void)
- void uart0_init (void)
- SIGNAL (SIG_USART0_RECV)
- void init_devices ()
- int main (void)

### Variables

- unsigned char data
- int count = 0
- int degree = 0

### 2.1.1 Function Documentation

#### 2.1.1.1 void buzzer_off ( void )

function to stop the buzzer.

#### 2.1.1.2 void buzzer_on ( void )

Function that starts the buzzer.

#### 2.1.1.3 void buzzer_pin_config ( void )

Setting the congfigrations for the buzzer.

#### 2.1.1.4 void init_devices ( )

Function To Initialize all The Devices.

#### 2.1.1.5 int main ( void )

Main function that does the initializations and brings the motors to initial position.

#### 2.1.1.6 void port_init ( )

Function to initialize ports

#### 2.1.1.7 SIGNAL ( SIG_USART0_RECV )

Function that actually receives the signals(commands) from the computer over the zigbee communication channel. It takes in the degree values and rotates the servo motors on the bot accordingly.

the degree sent from the computer is half the actual angle of rotation so that it can be sent in one byte. Hence, the actual degree is 2 times the received degree value.

#### 2.1.1.8 void uart0_init ( void )

Function To Initialize UART0 desired baud rate:9600 actual baud rate:9600 (error 0.0%) char size: 8 bit parity: Disabled

### 2.1.2 Variable Documentation

#### 2.1.2.1 int count = 0

#### 2.1.2.2 unsigned char data

initialization of globals used in the program

#### 2.1.2.3 int degree = 0

## 2.2 lcd.h File Reference

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

### Defines

- #define FCPU 11059200ul
- #define RS 0
- #define RW 1
- #define EN 2
- #define lcd_port PORTC
- #define sbit(reg, bit) reg |= (1<<bit)
- #define cbit(reg, bit) reg &= ∼(1<<bit)

### Functions

- void init_ports ()
- void lcd_reset ()
- void lcd_init ()
- void lcd_wr_command (unsigned char)
- void lcd_wr_char (char)
- void lcd_line1 ()
- void lcd_line2 ()
- void lcd_string (char ∗)

- void lcd_port_config (void)
- void lcd_port_init ()
- void lcd_set_4bit ()
- void lcd_home ()
- void lcd_cursor (char row, char column)
- void lcd_print (char row, char coloumn, unsigned int value, int digits)
- void init_lcd (void)

## Variables

- unsigned int temp
- unsigned int unit
- unsigned int tens
- unsigned int hundred
- unsigned int thousand
- unsigned int million

### 2.2.1 Define Documentation

#### 2.2.1.1 #define cbit( *reg*, *bit* ) reg &= ∼(1<<bit)

#### 2.2.1.2 #define EN 2

#### 2.2.1.3 #define FCPU 11059200ul

#### 2.2.1.4 #define lcd_port PORTC

#### 2.2.1.5 #define RS 0

#### 2.2.1.6 #define RW 1

#### 2.2.1.7 #define sbit( *reg,* *bit* ) reg |= (1<<bit)

### 2.2.2 Function Documentation

#### 2.2.2.1 void init_lcd ( void )

Function to initialize the lcd

#### 2.2.2.2 void init_ports ( )

#### 2.2.2.3 void lcd_cursor ( char *row,* char *column* )

Position the LCD cursor at "row", "column".

#### 2.2.2.4 void lcd_home ( )

Function to bring cursor at home position

#### 2.2.2.5 void lcd_init ( )

Function to Initialize LCD

#### 2.2.2.6 void lcd_line1 ( )

#### 2.2.2.7 void lcd_line2 ( )

#### 2.2.2.8 void lcd_port_config ( void )

Function to configure LCD port

#### 2.2.2.9 void lcd_port_init ( )

Function to Initialize PORTS

#### 2.2.2.10 void lcd_print ( char *row,* char *coloumn,* unsigned int *value,* int *digits* )

Function To Print Any input value upto the desired digit on LCD

#### 2.2.2.11 void lcd_reset ( )

#### 2.2.2.12 void lcd_set_4bit ( )

Function to Reset LCD

#### 2.2.2.13 void lcd_string ( char ∗ *str* )

Function to Print String on LCD

#### 2.2.2.14 void lcd_wr_char ( char *letter* )

Function to Write Data on LCD

#### 2.2.2.15 void lcd_wr_command ( unsigned char *cmd* )

Function to Write Command on LCD

### 2.2.3 Variable Documentation

#### 2.2.3.1 unsigned int hundred

#### 2.2.3.2 unsigned int million

#### 2.2.3.3 unsigned int temp

#### 2.2.3.4 unsigned int tens

#### 2.2.3.5 unsigned int thousand

#### 2.2.3.6 unsigned int unit

## 2.3 servo_motor.h File Reference

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

### Functions

- void servo1_pin_config (void)
- void servo2_pin_config (void)
- void servo3_pin_config (void)
- void servo_port_init (void)
- void timer1_init (void)
- void init_servo (void)
- void servo_1 (unsigned char degrees)
- void servo_2 (unsigned char degrees)
- void servo_3 (unsigned char degrees)
- void servo_1_free (void)
- void servo_2_free (void)
- void servo_3_free (void)

### 2.3.1 Function Documentation

#### 2.3.1.1 void init_servo ( void )

Function to initialize all the peripherals

#### 2.3.1.2 void servo1_pin_config ( void )

Configure PORTB 5 pin for servo motor 1 operation

#### 2.3.1.3 void servo2_pin_config ( void )

Configure PORTB 6 pin for servo motor 2 operation

#### 2.3.1.4 void servo3_pin_config ( void )

Configure PORTB 7 pin for servo motor 3 operation

#### 2.3.1.5 void servo_1 ( unsigned char *degrees* )

Function to rotate Servo 1 by a specified angle in the multiples of 2.25 degrees

#### 2.3.1.6 void servo_1_free ( void )

servo_free functions unlocks the servo motors from the any angle and make them free by giving 100% duty cycle at the PWM. This function can be used to reduce the power consumption of the motor if it is holding load against the gravity.

#### 2.3.1.7 void servo_2 ( unsigned char *degrees* )

Function to rotate Servo 2 by a specified angle in the multiples of 2.25 degrees

#### 2.3.1.8 void servo_2_free ( void )

servo_free functions unlocks the servo motors from the any angle and make them free by giving 100% duty cycle at the PWM. This function can be used to reduce the power consumption of the motor if it is holding load against the gravity.

#### 2.3.1.9 void servo_3 ( unsigned char *degrees* )

Function to rotate Servo 3 by a specified angle in the multiples of 2.25 degrees

#### 2.3.1.10 void servo_3_free ( void )

servo_free functions unlocks the servo motors from the any angle and make them free by giving 100% duty cycle at the PWM. This function can be used to reduce the power consumption of the motor if it is holding load against the gravity.

#### 2.3.1.11 void servo_port_init ( void )

Initialize the ports

#### 2.3.1.12 void timer1_init ( void )

TIMER1 initialization in 10 bit fast PWM mode prescale:256 WGM: 7) PWM 10bit fast, TOP=0x03FF actual value: 42.187Hz

Output compare eegister high value for servo 3

{COM1A1=1, COM1A0=0; COM1B1=1, COM1B0=0; COM1C1=1 COM1C0=0} For Overriding normal port functionalit to OCRnA outputs. {WGM11=1, WGM10=1} Along With WGM12 in TCCR1B for Selecting FAST PWM Mode