iRobot Final

Generated by Doxygen 1.8.4

Sun May 4 2014 16:56:01

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 object_s Struct Reference

Data Fields

- int degrees_start
- int degrees_end
- float distance
- float width

3.1.1 Field Documentation

3.1.1.1 int degrees_end

3.1.1.2 int degrees_start

3.1.1.3 float distance

3.1.1.4 float width

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

- iRobot Final.c

3.2 oi_t Struct Reference

iRobot Create Sensor Data

```
#include <open_interface.h>
```

Data Fields

- uint8_t bumper_right: 1
- uint8_t bumper_left: 1
- uint8_t wheeldrop_right: 1
- uint8_t wheeldrop_left: 1
- uint8_t wheeldrop_caster: 1
- uint8_t wall

- uint8_t cliff_left
- uint8_t cliff_frontleft
- uint8_t cliff_frontright
- uint8_t cliff_right
- uint8_t virtual_wall
- uint8_t overcurrent_ld1: 1
- uint8_t overcurrent_ld0: 1
- uint8_t overcurrent_ld2: 1
- uint8_t overcurrent_driveright: 1
- uint8_t overcurrent_driveleft: 1
- uint16_t unused_bytes
- uint8_t infrared_byte
- uint8_t button_play: 2
- uint8_t button_advance: 1
- int16_t distance
- int16_t angle
- uint8_t charging_state
- uint16_t voltage
- int16_t current
- int8_t temperature
- uint16_t charge
- uint16_t capacity
- uint16_t wall_signal
- uint16_t cliff_left_signal
- uint16_t cliff_frontleft_signal
- uint16_t cliff_frontright_signal
- uint16_t cliff_right_signal
- uint8_t cargo_bay_io0: 1
- uint8_t cargo_bay_io1: 1
- uint8_t cargo_bay_io2: 1
- uint8_t cargo_bay_io3: 1
- uint8_t cargo_bay_baud: 1
- uint16_t cargo_bay_voltage
- uint8_t internal_charger_on: 1
- uint8_t home_base_charger_on: 1
- uint8_t oi_mode
- uint8_t song_number
- uint8_t song_playing
- uint8_t number_packets
- int16_t requested_velocity
- int16_t requested_radius
- int16_t requested_right_velocity
- int16_t requested_left_velocity

3.2.1 Detailed Description

iRobot Create Sensor Data

3.2.2 Field Documentation

3.2.2.1 int16_t angle

3.2.2.2 uint8_t bumper_left

3.2.2.3 uint8_t bumper_right

3.2.2.4 uint8_t button_advance

3.2.2.5 uint8_t button_play

3.2.2.6 uint16_t capacity

3.2.2.7 uint8_t cargo_bay_baud

3.2.2.8 uint8_t cargo_bay_io0

3.2.2.9 uint8_t cargo_bay_io1

3.2.2.10 uint8_t cargo_bay_io2

3.2.2.11 uint8_t cargo_bay_io3

3.2.2.12 uint16_t cargo_bay_voltage

3.2.2.13 uint16_t charge

3.2.2.14 uint8_t charging_state

3.2.2.15 uint8_t cliff_frontleft

3.2.2.16 uint16_t cliff_frontleft_signal

3.2.2.17 uint8_t cliff_frontright

3.2.2.18 uint16_t cliff_frontright_signal

3.2.2.19 uint8_t cliff_left

3.2.2.20 uint16_t cliff_left_signal

3.2.2.21 uint8_t cliff_right

3.2.2.22 uint16_t cliff_right_signal

3.2.2.23 int16_t current

3.2.2.24 int16_t distance

3.2.2.25 uint8_t home_base_charger_on

3.2.2.26 uint8_t infrared_byte

3.2.2.27 uint8_t internal_charger_on

3.2.2.28 uint8_t number_packets

3.2.2.29 uint8_t oi_mode

3.2.2.30 uint8_t overcurrent_driveleft

3.2.2.31 uint8_t overcurrent_driveright

3.2.2.32 uint8_t overcurrent_ld0

3.2.2.33 uint8_t overcurrent_ld1

3.2.2.34 uint8_t overcurrent_ld2

3.2.2.35 int16_t requested_left_velocity

3.2.2.36 int16_t requested_radius

3.2.2.37 int16_t requested_right_velocity

3.2.2.38 int16_t requested_velocity

3.2.2.39 uint8_t song_number

3.2.2.40 uint8_t song_playing

3.2.2.41 int8_t temperature

3.2.2.42 uint16_t unused_bytes

3.2.2.43 uint8_t virtual_wall

3.2.2.44 uint16_t voltage

3.2.2.45 uint8_t wall

3.2.2.46 uint16_t wall_signal

3.2.2.47 uint8_t wheeldrop_caster

3.2.2.48 uint8_t wheeldrop_left

3.2.2.49 uint8_t wheeldrop_right

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

- open_interface.h

Chapter 4

File Documentation

4.1 iRobot Final.c File Reference

```
#include <string.h>
#include <avr/io.h>
#include "util.h"
#include "lcd.h"
#include "usart.h"
#include "open_interface.h"
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
```

Data Structures

- struct object_s

Typedefs

- typedef struct object_s object_t

Functions

- int main (void)

4.1.1 Typedef Documentation

4.1.1.1 typedef struct object_s object_t

4.1.2 Function Documentation

4.1.2.1 int main (void)

Big Scan

Small Scan

Move Forward

Turn Right

Turn Left

Move Backward

Report Sensor Data

Play Song

4.2 lcd.c File Reference

```
#include <avr/io.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "util.h"
#include "lcd.h"
```

Macros

- #define HD_LCD_CLEAR 0x01
- #define HD_RETURN_HOME 0x02
- #define HD_CURSOR_SHIFT_DEC 0x05
- #define HD_CURSOR_SHIFT_INC 0x07
- #define HD_DISPLAY_CONTROL 3
- #define HD_DISPLAY_ON 2
- #define HD_CURSOR_ON 1
- #define HD_BLINK_ON 0
- #define HD_CURSOR_MOVE_LEFT 0x10
- #define HD_CURSOR_MOVE_RIGHT 0x14
- #define HD_DISPLAY_SHIFT_LEFT 0x18
- #define HD_DISPLAY_SHIFT_RIGHT 0x1C
- #define LCD_WIDTH 20
- #define LCD_HEIGHT 4
- #define LCD_TOTAL_CHARS (LCD_WIDTH∗LCD_HEIGHT)

Functions

- void lcd_toggle_clear (char delay)

  Triggers loading of bits by LCD controller and clears bits after toggle.
- void lcd_home_anyloc (unsigned char location)

  Sets character position to any valid location.
- void lcd_init (void)

  Initializes PORTA to communicate with LCD controller.
- void lcd_command (char data)

  Submits command to LCD controller.
- void lcd_clear (void)

  Clears the LCD.
- void lcd_home_line1 (void)

  Sets character position to first line first position.
- void lcd_home_line2 (void)

  Sets character position to second line first position.

- void lcd_home_line3 (void)

  Sets character position to third line first position.
- void lcd_home_line4 (void)

  Sets character position to fourth line first position.
- void lcd_display_shift_left (void)

  Shift display content left.
- void lcd_puts (char *string)

  Prints string to lcd, starting at the current cursor position.
- void lcd_putc (char data)

  Prints one character at the current cursor position.
- void lprintf (const char *format,...)

  Print a formatted string to the LCD screen.

4.2.1 Macro Definition Documentation

4.2.1.1 #define HD_BLINK_ON 0

4.2.1.2 #define HD_CURSOR_MOVE_LEFT 0x10

4.2.1.3 #define HD_CURSOR_MOVE_RIGHT 0x14

4.2.1.4 #define HD_CURSOR_ON 1

4.2.1.5 #define HD_CURSOR_SHIFT_DEC 0x05

4.2.1.6 #define HD_CURSOR_SHIFT_INC 0x07

4.2.1.7 #define HD_DISPLAY_CONTROL 3

4.2.1.8 #define HD_DISPLAY_ON 2

4.2.1.9 #define HD_DISPLAY_SHIFT_LEFT 0x18

4.2.1.10 #define HD_DISPLAY_SHIFT_RIGHT 0x1C

4.2.1.11 #define HD_LCD_CLEAR 0x01

lcd.c: functions for displaying content to the LCD screen

Date

06/26/2012

4.2.1.12 #define HD_RETURN_HOME 0x02

4.2.1.13 #define LCD_HEIGHT 4

4.2.1.14 #define LCD_TOTAL_CHARS (LCD_WIDTH*LCD_HEIGHT)

4.2.1.15 #define LCD_WIDTH 20

4.2.2 Function Documentation

4.2.2.1 void lcd_clear (void)

Clears the LCD.

4.2.2.2 void lcd_command (char *data*)

Submits command to LCD controller.

4.2.2.3 void lcd_display_shift_left (void)

Shift display content left.

4.2.2.4 void lcd_home_anyloc (unsigned char *location*)

Sets character position to any valid location.

4.2.2.5 void lcd_home_line1 (void)

Sets character position to first line first position.

Sets cursor position to left side of a given line.

4.2.2.6 void lcd_home_line2 (void)

Sets character position to second line first position.

4.2.2.7 void lcd_home_line3 (void)

Sets character position to third line first position.

4.2.2.8 void lcd_home_line4 (void)

Sets character position to fourth line first position.

4.2.2.9 void lcd_init (void)

Initializes PORTA to communicate with LCD controller.

4.2.2.10 void lcd_putc (char *data*)

Prints one character at the current cursor position.

4.2.2.11 void lcd_puts (char * *string*)

Prints string to lcd, starting at the current cursor position.

Prints a string of characters starting at the current cursor position.

4.2.2.12 void lcd_toggle_clear (char *delay*)

Triggers loading of bits by LCD controller and clears bits after toggle.

4.2.2.13 void lprintf (const char * *format*, ...)

Print a formatted string to the LCD screen.

Prints a string to the lcd; Google "printf" for documentation.

Mimics the C library function printf for writing to the LCD screen. The function is buffered; i.e. if you call lprintf twice with the same string, it will only update the LCD the first time.

Google "printf" for documentation on the formatter string.

Code from this site was also used: http://www.ozzu.com/cpp-tutorials/tutorial-writing-custom-printf-html

Author

Kerrick Staley & Chad Nelson

Date

05/16/2012

4.3 lcd.h File Reference

Functions

- void lcd_init (void)

  Initializes PORTA to communicate with LCD controller.
- void lcd_home_line1 (void)

  Sets cursor position to left side of a given line.
- void lcd_home_line2 (void)

  Sets character position to second line first position.
- void lcd_home_line3 (void)

  Sets character position to third line first position.
- void lcd_home_line4 (void)

  Sets character position to fourth line first position.
- void lprintf (const char *formatter,...)

  Prints a string to the lcd; Google "printf" for documentation.
- void lcd_puts (char *data)

  Prints a string of characters starting at the current cursor position.
- void lcd_putc (char data)

  Prints one character at the current cursor position.
- void lcd_clear (void)

  Clears the LCD.
- void lcd_command (char data)

  Submits command to LCD controller.

4.3.1 Function Documentation

4.3.1.1 void lcd_clear (void)

Clears the LCD.

4.3.1.2 void lcd_command (char *data*)

Submits command to LCD controller.

4.3.1.3 void lcd_home_line1 (void)

Sets cursor position to left side of a given line.

4.3.1.4 void lcd_home_line2 (void)

Sets character position to second line first position.

4.3.1.5 void lcd_home_line3 (void)

Sets character position to third line first position.

4.3.1.6 void lcd_home_line4 (void)

Sets character position to fourth line first position.

4.3.1.7 void lcd_init (void)

Initializes PORTA to communicate with LCD controller.

4.3.1.8 void lcd_putc (char *data*)

Prints one character at the current cursor position.

4.3.1.9 void lcd_puts (char * *string*)

Prints a string of characters starting at the current cursor position.

4.3.1.10 void lprintf (const char * *format, ...*)

Prints a string to the lcd; Google "printf" for documentation.

Prints a string to the lcd; Google "printf" for documentation.

Mimics the C library function printf for writing to the LCD screen. The function is buffered; i.e. if you call lprintf twice with the same string, it will only update the LCD the first time.

Google "printf" for documentation on the formatter string.

Code from this site was also used: http://www.ozzu.com/cpp-tutorials/tutorial-writing-custom-printf-html

Author

Kerrick Staley & Chad Nelson

Date

05/16/2012

4.4 open_interface.c File Reference

```
#include <stdlib.h>
#include "util.h"
#include "open_interface.h"
```

Functions

- oi_t * oi_alloc ()
  Allocate memory for a the sensor data.
- void oi_free (oi_t *self)
  Free memory from a pointer to the sensor data struct.
- void oi_init (oi_t *self)
  Initialize the Create.
- void oi_update (oi_t *self)
  Update the Create. This will update all the sensor data and store it in the oi_t struct.
- void oi_set_leds (uint8_t play_led, uint8_t advance_led, uint8_t power_color, uint8_t power_intensity)
  Sets the LEDs on the iRobot.
- void oi_set_wheels (int16_t right_wheel, int16_t left_wheel)
  Drive wheels directly; speeds are in mm / sec.
- void oi_load_song (int song_index, int num_notes, unsigned char *notes, unsigned char *duration)
  Loads a song onto the iRobot Create.
- void oi_play_song (int index)
  Plays a given song; use oi_load_song(...) first.
- void go_charge (void)
  Runs default go charge program; robot will search for dock.
- void oi_byte_tx (unsigned char value)
  Transmit a byte of data over the serial connection to the Create.
- unsigned char oi_byte_rx (void)
  Receive a byte of data from the Create serial connection. Blocks until a byte is received.

4.4.1 Function Documentation

4.4.1.1 void go_charge (void)

Runs default go charge program; robot will search for dock.

Calls in built in demo to send the iRobot to an open home base This will cause the iRobot to enter the Passive state

4.4.1.2 oi_t* oi_alloc ( )

Allocate memory for a the sensor data.

Allocate memory for the oi_sensor_t struct.

4.4.1.3 unsigned char oi_byte_rx (void)

Receive a byte of data from the Create serial connection. Blocks until a byte is received.

Returns

8-bit value returned from the Create

4.4.1.4 void oi_byte_tx (unsigned char *value*)

Transmit a byte of data over the serial connection to the Create.

Parameters

value	8-bit value to transmit to the Create

4.4.1.5 void oi_free (oi_t * *self*)

Free memory from a pointer to the sensor data struct.

4.4.1.6 void oi_init (oi_t * *self*)

Initialize the Create.

Initialize the Create. This must be called first.

4.4.1.7 void oi_load_song (int *song_index*, int *num_notes*, unsigned char * *notes*, unsigned char * *duration*)

Loads a song onto the iRobot Create.

Load song sequence.

4.4.1.8 void oi_play_song (int *index*)

Plays a given song; use oi_load_song(...) first.

Play song.

4.4.1.9 void oi_set_leds (uint8_t *play_led*, uint8_t *advance_led*, uint8_t *power_color*, uint8_t *power_intensity*)

Sets the LEDs on the iRobot.

Set the LEDS on the Create.

Set the state of the three LEDs on the iRobot (Power, Play, Advance). uint8_t either 0 (off) or 1 (on) uint8_t either 0 (off) or 1 (on) uint8_t the color of the power LED; 0 = green, 255 = red uint8_t the intensity of the power LED; 0 = off, 255 = full intensity

4.4.1.10 void oi_set_wheels (int16_t *right_wheel*, int16_t *left_wheel*)

Drive wheels directly; speeds are in mm / sec.

Set direction and speed of the robot's wheels.

4.4.1.11 void oi_update (oi_t * *self*)

Update the Create. This will update all the sensor data and store it in the oi_t struct.

Update the Create. This will update all the sensor data.

4.5 open_interface.h File Reference

```
#include <inttypes.h>
```

```
#include <avr/io.h>
```

Data Structures

- struct oi_t

  iRobot Create Sensor Data

Macros

- #define FOSC 16000000
- #define OI_OPCODE_START 128
- #define OI_OPCODE_BAUD 129
- #define OI_OPCODE_CONTROL 130
- #define OI_OPCODE_SAFE 131
- #define OI_OPCODE_FULL 132
- #define OI_OPCODE_POWER 133
- #define OI_OPCODE_SPOT 134
- #define OI_OPCODE_CLEAN 135
- #define OI_OPCODE_MAX 136
- #define OI_OPCODE_DRIVE 137
- #define OI_OPCODE_MOTORS 138
- #define OI_OPCODE_LEDS 139
- #define OI_OPCODE_SONG 140
- #define OI_OPCODE_PLAY 141
- #define OI_OPCODE_SENSORS 142
- #define OI_OPCODE_FORCEDOCK 143
- #define OI_OPCODE_PWM_MOTORS 144
- #define OI_OPCODE_DRIVE_WHEELS 145
- #define OI_OPCODE_DRIVE_PWM 146
- #define OI_OPCODE_OUTPUTS 147
- #define OI_OPCODE_STREAM 148
- #define OI_OPCODE_QUERY_LIST 149
- #define OI_OPCODE_DO_STREAM 150
- #define OI_OPCODE_SEND_IR_CHAR 151
- #define OI_OPCODE_SCRIPT 152
- #define OI_OPCODE_PLAY_SCRIPT 153
- #define OI_OPCODE_SHOW_SCRIPT 154
- #define OI_OPCODE_WAIT_TIME 155
- #define OI_OPCODE_WAIT_DISTANCE 156
- #define OI_OPCODE_WAIT_ANGLE 157
- #define OI_OPCODE_WAIT_EVENT 158
- #define OI_SENSOR_PACKET_GROUP0 0
- #define OI_SENSOR_PACKET_GROUP1 1
- #define OI_SENSOR_PACKET_GROUP2 2
- #define OI_SENSOR_PACKET_GROUP3 3
- #define OI_SENSOR_PACKET_GROUP4 4
- #define OI_SENSOR_PACKET_GROUP5 5
- #define OI_SENSOR_PACKET_GROUP6 6
- #define MIN(a, b) ((a < b) ? (a) : (b))
- #define MAX(a, b) ((a > b) ? (a) : (b))
- #define PIN_0 0x01
- #define PIN_1 0x02

- #define PIN_2 0x04
- #define PIN_3 0x08
- #define PIN_4 0x10
- #define PIN_5 0x20
- #define PIN_6 0x40
- #define PIN_7 0x80

Typedefs

- typedef oi_t oi_sensors_t

Functions

- oi_t * oi_alloc ()
  Allocate memory for the oi_sensor_t struct.
- void oi_init (oi_t *self)
  Initialize the Create. This must be called first.
- void oi_free (oi_t *self)
  Free memory from a pointer to the sensor data struct.
- void oi_update (oi_t *self)
  Update the Create. This will update all the sensor data.
- void oi_set_leds (uint8_t play_led, uint8_t advance_led, uint8_t power_color, uint8_t power_intensity)
  Set the LEDS on the Create.
- void oi_set_wheels (int16_t right_wheel, int16_t left_wheel)
  Set direction and speed of the robot's wheels.
- void oi_byte_tx (unsigned char value)
  Transmit a byte of data over the serial connection to the Create.
- unsigned char oi_byte_rx (void)
  Receive a byte of data from the Create serial connection. Blocks until a byte is received.
- void oi_load_song (int song_index, int num_notes, unsigned char *notes, unsigned char *duration)
  Load song sequence.
- void oi_play_song (int index)
  Play song.
- void go_charge (void)
  Runs default go charge program; robot will search for dock.

4.5.1 Macro Definition Documentation

4.5.1.1 #define FOSC 16000000

Open Interface API - Provides a set of functions for controlling the Create Documentation: http://www.-irobot.com/filelibrary/pdfs/hrd/create/create%20open%20interface_v2.pdf

void main() { oi_sensors_t *robot = oi_alloc(); oi_init(robot);

// ... your code ...

free(robot); }

Author

See "Robotics Primer Workbook" project hosted on SourceForge.Net; Edited for clarity by Chad Nelson

Date

06/26/2012

4.5.1.2 #define MAX(*a*, *b*) ((a > b) ? (a) : (b))

4.5.1.3 #define MIN(*a*, *b*) ((a < b) ? (a) : (b))

4.5.1.4 #define OI_OPCODE_BAUD 129

4.5.1.5 #define OI_OPCODE_CLEAN 135

4.5.1.6 #define OI_OPCODE_CONTROL 130

4.5.1.7 #define OI_OPCODE_DO_STREAM 150

4.5.1.8 #define OI_OPCODE_DRIVE 137

4.5.1.9 #define OI_OPCODE_DRIVE_PWM 146

4.5.1.10 #define OI_OPCODE_DRIVE_WHEELS 145

4.5.1.11 #define OI_OPCODE_FORCEDOCK 143

4.5.1.12 #define OI_OPCODE_FULL 132

4.5.1.13 #define OI_OPCODE_LEDS 139

4.5.1.14 #define OI_OPCODE_MAX 136

4.5.1.15 #define OI_OPCODE_MOTORS 138

4.5.1.16 #define OI_OPCODE_OUTPUTS 147

4.5.1.17 #define OI_OPCODE_PLAY 141

4.5.1.18 #define OI_OPCODE_PLAY_SCRIPT 153

4.5.1.19 #define OI_OPCODE_POWER 133

4.5.1.20 #define OI_OPCODE_PWM_MOTORS 144

4.5.1.21 #define OI_OPCODE_QUERY_LIST 149

4.5.1.22 #define OI_OPCODE_SAFE 131

4.5.1.23 #define OI_OPCODE_SCRIPT 152

4.5.1.24 #define OI_OPCODE_SEND_IR_CHAR 151

4.5.1.25 #define OI_OPCODE_SENSORS 142

4.5.1.26 #define OI_OPCODE_SHOW_SCRIPT 154

4.5.1.27 #define OI_OPCODE_SONG 140

4.5.1.28 #define OI_OPCODE_SPOT 134

4.5.1.29 #define OI_OPCODE_START 128

4.5.1.30 #define OI_OPCODE_STREAM 148

4.5.1.31 #define OI_OPCODE_WAIT_ANGLE 157

4.5.1.32 #define OI_OPCODE_WAIT_DISTANCE 156

4.5.1.33 #define OI_OPCODE_WAIT_EVENT 158

4.5.1.34 #define OI_OPCODE_WAIT_TIME 155

4.5.1.35 #define OI_SENSOR_PACKET_GROUP0 0

4.5.1.36 #define OI_SENSOR_PACKET_GROUP1 1

4.5.1.37 #define OI_SENSOR_PACKET_GROUP2 2

4.5.1.38 #define OI_SENSOR_PACKET_GROUP3 3

4.5.1.39 #define OI_SENSOR_PACKET_GROUP4 4

4.5.1.40 #define OI_SENSOR_PACKET_GROUP5 5

4.5.1.41 #define OI_SENSOR_PACKET_GROUP6 6

4.5.1.42 #define PIN_0 0x01

4.5.1.43 #define PIN_1 0x02

4.5.1.44 #define PIN_2 0x04

4.5.1.45 #define PIN_3 0x08

4.5.1.46 #define PIN_4 0x10

4.5.1.47 #define PIN_5 0x20

4.5.1.48 #define PIN_6 0x40

4.5.1.49 #define PIN_7 0x80

4.5.2 Typedef Documentation

4.5.2.1 typedef oi_t oi_sensors_t

4.5.3 Function Documentation

4.5.3.1 void go_charge (void)

Runs default go charge program; robot will search for dock.

Calls in built in demo to send the iRobot to an open home base This will cause the iRobot to enter the Passive state

4.5.3.2 oi_t* oi_alloc ( )

Allocate memory for the oi_sensor_t struct.

4.5.3.3 unsigned char oi_byte_rx (void)

Receive a byte of data from the Create serial connection. Blocks until a byte is received.

Returns

8-bit value returned from the Create

4.5.3.4 void oi_byte_tx (unsigned char *value*)

Transmit a byte of data over the serial connection to the Create.

Parameters

value	8-bit value to transmit to the Create

4.5.3.5 void oi_free (oi_t * *self*)

Free memory from a pointer to the sensor data struct.

4.5.3.6 void oi_init (oi_t * *self*)

Initialize the Create. This must be called first.

4.5.3.7 void oi_load_song (int *song_index,* int *num_notes,* unsigned char * *notes,* unsigned char * *duration*)

Load song sequence.

Parameters

An	integer value from 0 - 15 that acts as a label for note sequence
An	integer value from 1 - 16 indicating the number of notes in the sequence
A	pointer to a sequence of notes stored as integer values
A	pointer to a sequence of durations that correspond to the notes

4.5.3.8 void oi_play_song (int *index*)

Play song.

Parameters

An	integer value from 0 - 15 that is a previously establish song index

4.5.3.9 void oi_set_leds (uint8_t *play_led,* uint8_t *advance_led,* uint8_t *power_color,* uint8_t *power_intensity*)

Set the LEDS on the Create.

Parameters

play_led	0=off, 1=on

advance_led	0=off, 1=on
power_color	(0-255), 0=green, 255=red
power_intensity	(0-255) 0=off, 255=full intensity

Set the LEDS on the Create.

Set the state of the three LEDs on the iRobot (Power, Play, Advance). uint8_t either 0 (off) or 1 (on) uint8_t either 0 (off) or 1 (on) uint8_t the color of the power LED; 0 = green, 255 = red uint8_t the intensity of the power LED; 0 = off, 255 = full intensity

4.5.3.10 void oi_set_wheels (int16_t *right_wheel,* int16_t *left_wheel*)

Set direction and speed of the robot's wheels.

Parameters

linear	velocity in mm/s values range from -500 -> 500 of right wheel
linear	velocity in mm/s values range from -500 -> 500 of left wheel

4.5.3.11 void oi_update (oi_t * *self*)

Update the Create. This will update all the sensor data.

4.6 usart.c File Reference

```
#include <avr/io.h>
#include "usart.h"
```

Macros

- #define F_CPU 16000000
- #define BAUD 38400

Functions

- void serial_init (void)
- void USART_Transmit (char data)
- unsigned char USART_Receive (void)
- char * USART_RecieveString ()
- void USART_SendString (char SentString[])
- void USART_Flush (void)

4.6.1 Macro Definition Documentation

4.6.1.1 #define BAUD 38400

4.6.1.2 #define F_CPU 16000000

4.6.2 Function Documentation

4.6.2.1 void serial_init (void)

4.6.2.2 void USART_Flush (void)

4.6.2.3 unsigned char USART_Receive (void)

4.6.2.4 char* USART_RecieveString ( )

4.6.2.5 void USART_SendString (char *SentString[]*)

4.6.2.6 void USART_Transmit (char *data*)

4.7 usart.h File Reference

Functions

- void serial_init (void)
- void USART_Transmit (char data)
- unsigned char USART_Receive (void)
- void USART_SendString (char SentString[])
- char * USART_RecieveString ()
- void USART_Flush (void)

4.7.1 Function Documentation

4.7.1.1 void serial_init (void)

4.7.1.2 void USART_Flush (void)

4.7.1.3 unsigned char USART_Receive (void)

4.7.1.4 char* USART_RecieveString ( )

4.7.1.5 void USART_SendString (char *SentString[]*)

4.7.1.6 void USART_Transmit (char *data*)

4.8 util.c File Reference

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

Macros

- #define FAST_SPEED 200
- #define MEDIUM_SPEED 150
- #define SLOW_SPEED 100

Functions

- void timer2_start (char unit)

  Start timer2.

- void timer2_stop ()

  Stop timer2.
- void ADC_init (void)

  Initialize the IR distance sensor.
- unsigned int ADC_read (char channel)

  Read the output from the ADC, currently using channel 2.
- float calcCm (unsigned int DigitalOutput)

  Calculate the distance from ADC result in centimeters.
- void wait_ms (unsigned int time_val)

  Blocks for a specified number of milliseconds.
- ISR (TIMER2_COMP_vect)

  Interrupt handler (runs every 1 ms)
- void init_push_buttons (void)

  Initialize PORTC to accept push buttons as input.
- void shaft_encoder_init (void)

  Initialize PORTC for input from the shaft encoder.
- char read_shaft_encoder (void)

  Read the shaft encoder.
- void stepper_init (void)

  Initialize PORTE to control the stepper motor.
- void move_stepper_motor_by_step (int num_steps, int direction)

  Turn the Stepper Motor.
- void ping_init (void)

  initializes ping sensor
- float ping_read (void)

  Reads the current distance on the sonar.
- ISR (TIMER1_CAPT_vect)

  Gets called when we have a rising or falling edge depending on TCCR1B bit 6.
- void send_pulse (void)

  send a pulse on PD4
- float time2dist (unsigned int time)

  Converts the time that we measure into a distance in centimeters, takes in count from read.
- void servo_turn (int degrees)

  turns the servo motor a specific degrees
- void timer3_init (void)
- char read_push_buttons (void)

  Return the position of button being pushed.
- char move_forward (oi_t *sensor, int centimeters)

  Move forward contains all other functions, allows for error checking while moving.
- void move_backward (oi_t *sensor, int centimeters)

  Moves iRobot Create platform backwards a specified value of centimeters.
- void turn_clockwise (oi_t *sensor, int degrees)

  turns clockwise by a given degree
- void turn_counterclockwise (oi_t *sensor, int degrees)

  turns clockwise by a given degree
- void reportData (oi_t *sensor)

  sends all sensor data back through USART
- void song_init ()

  Plays song 'Take on me'.

Variables

- volatile unsigned int timer2_tick
- volatile unsigned current_time = 0
- volatile int update_flag = 0

4.8.1 Macro Definition Documentation

4.8.1.1 #define FAST_SPEED 200

util.c: utility functions for the Atmel platform

For an overview of how timer based interrupts work, see page 111 and 133-137 of the Atmel Mega128 User Guide

Author

Zhao Zhang, Chad Nelson, Nick Montelibano, Alex Lende, Chris Chafe, Nic Dubois, Ben Williams

4.8.1.2 #define MEDIUM_SPEED 150

4.8.1.3 #define SLOW_SPEED 100

4.8.2 Function Documentation

4.8.2.1 void ADC_init (void)

Initialize the IR distance sensor.

4.8.2.2 unsigned int ADC_read (char *channel*)

Read the output from the ADC, currently using channel 2.

4.8.2.3 float calcCm (unsigned int *DigitalOutput*)

Calculate the distance from ADC result in centimeters.

4.8.2.4 void init_push_buttons (void)

Initialize PORTC to accept push buttons as input.

Initialize PORTC, which is used by the push buttons.

4.8.2.5 ISR (TIMER2_COMP_vect)

Interrupt handler (runs every 1 ms)

4.8.2.6 ISR (TIMER1_CAPT_vect)

Gets called when we have a rising or falling edge depending on TCCR1B bit 6.

4.8.2.7 void move_backward (oi_t * *sensor,* int *centimeters*)

Moves iRobot Create platform backwards a specified value of centimeters.

4.8.2.8 char move_forward (oi_t * *sensor,* int *centimeters*)

Move forward contains all other functions, allows for error checking while moving.

4.8.2.9 void move_stepper_motor_by_step (int *num_steps,* int *direction*)

Turn the Stepper Motor.

Stepper motor move function.

Turn the stepper motor a given number of steps.

Parameters

num_steps	A value between 1 and 200 steps (1.8 to 360 degrees)
direction	Indication of direction: 1 for CW and -1 for CCW

4.8.2.10 void ping_init (void)

initializes ping sensor

4.8.2.11 float ping_read (void)

Reads the current distance on the sonar.

4.8.2.12 char read_push_buttons (void)

Return the position of button being pushed.

Return the position of button being pushed.

Returns

the position of the button being pushed. A 1 is the rightmost button. 0 indicates no button being pressed

4.8.2.13 char read_shaft_encoder (void)

Read the shaft encoder.

Shaft encoder read function.

Reads the two switches of the shaft encoder and compares the values to the previous read. This function should be called very frequently for the best results.

Returns

a value indicating the shaft encoder has moved: 0 = no rotation (switches did not change) 1 = CW rotation -1 = CCW rotation

4.8.2.14 void reportData (oi_t * *sensor*)

sends all sensor data back through USART

4.8.2.15 void send_pulse (void)

send a pulse on PD4

4.8.2.16 void servo_turn (int *degrees*)

turns the servo motor a specific degrees

move to the position

4.8.2.17 void shaft_encoder_init (void)

Initialize PORTC for input from the shaft encoder.

Shaft encoder initialization.

4.8.2.18 void song_init ( )

Plays song 'Take on me'.

4.8.2.19 void stepper_init (void)

Initialize PORTE to control the stepper motor.

Initialize Stepper Motor.

4.8.2.20 float time2dist (unsigned int *time*)

Converts the time that we measure into a distance in centimeters, takes in count from read.

4.8.2.21 void timer2_start (char *unit*)

Start timer2.

4.8.2.22 void timer2_stop ( )

Stop timer2.

4.8.2.23 void timer3_init (void)

4.8.2.24 void turn_clockwise (oi_t * *sensor,* int *degrees*)

turns clockwise by a given degree

4.8.2.25 void turn_counterclockwise (oi_t * *sensor,* int *degrees*)

turns clockwise by a given degree

4.8.2.26 void wait_ms (unsigned int *time_val*)

Blocks for a specified number of milliseconds.

4.8.3 Variable Documentation

4.8.3.1 volatile unsigned current_time = 0

4.8.3.2 volatile unsigned int timer2_tick

4.8.3.3 volatile int update_flag = 0

4.9 util.h File Reference

```
#include "open_interface.h"
```

Functions

- void wait_ms (unsigned int time_val)

  Blocks for a specified number of milliseconds.
- void shaft_encoder_init (void)

  Shaft encoder initialization.
- char read_shaft_encoder (void)

  Shaft encoder read function.
- void stepper_init (void)

  Initialize Stepper Motor.
- void move_stepper_motor_by_step (int num_steps, int direction)

  Stepper motor move function.
- void init_push_buttons (void)

  Initialize PORTC, which is used by the push buttons.
- char read_push_buttons (void)

  Return the position of button being pushed.
- void send_pulse (void)

  send a pulse on PD4
- float time2dist (unsigned int Cycles)

  Converts the time that we measure into a distance in centimeters, takes in count from read.
- void ping_init (void)

  initializes ping sensor
- float ping_read (void)

  Reads the current distance on the sonar.
- void timer3_init (void)
- void servo_turn (int degrees)

  turns the servo motor a specific degrees
- void ADC_init (void)

  Initialize the IR distance sensor.
- unsigned int ADC_read (char channel)

  Read the output from the ADC, currently using channel 2.
- float calcCm (unsigned int DigitalOutput)

  Calculate the distance from ADC result in centimeters.
- char move_forward (oi_t *sensor, int centimeters)

  Move forward contains all other functions, allows for error checking while moving.
- void move_backward (oi_t *sensor, int centimeters)

  Moves iRobot Create platform backwards a specified value of centimeters.
- void turn_counterclockwise (oi_t *sensor, int degrees)

turns clockwise by a given degree

- void turn_clockwise (oi_t *sensor, int degrees)

  turns clockwise by a given degree

- void song_init ()

  Plays song 'Take on me'.

- void reportData (oi_t *sensor)

  sends all sensor data back through USART

4.9.1 Function Documentation

4.9.1.1 void ADC_init (void)

Initialize the IR distance sensor.

4.9.1.2 unsigned int ADC_read (char *channel*)

Read the output from the ADC, currently using channel 2.

4.9.1.3 float calcCm (unsigned int *DigitalOutput*)

Calculate the distance from ADC result in centimeters.

4.9.1.4 void init_push_buttons (void)

Initialize PORTC, which is used by the push buttons.

4.9.1.5 void move_backward (oi_t * *sensor,* int *centimeters*)

Moves iRobot Create platform backwards a specified value of centimeters.

4.9.1.6 char move_forward (oi_t * *sensor,* int *centimeters*)

Move forward contains all other functions, allows for error checking while moving.

4.9.1.7 void move_stepper_motor_by_step (int *num_steps,* int *direction*)

Stepper motor move function.

Stepper motor move function.

Turn the stepper motor a given number of steps.

Parameters

num_steps	A value between 1 and 200 steps (1.8 to 360 degrees)
direction	Indication of direction: 1 for CW and -1 for CCW

4.9.1.8 void ping_init (void)

initializes ping sensor

4.9.1.9 float ping_read (void)

Reads the current distance on the sonar.

4.9.1.10 char read_push_buttons (void)

Return the position of button being pushed.

Return the position of button being pushed.

Returns

the position of the button being pushed. A 1 is the rightmost button. 0 indicates no button being pressed

4.9.1.11 char read_shaft_encoder (void)

Shaft encoder read function.

Shaft encoder read function.

Reads the two switches of the shaft encoder and compares the values to the previous read. This function should be called very frequently for the best results.

Returns

a value indicating the shaft encoder has moved: 0 = no rotation (switches did not change) 1 = CW rotation -1 = CCW rotation

4.9.1.12 void reportData (oi_t * *sensor*)

sends all sensor data back through USART

4.9.1.13 void send_pulse (void)

send a pulse on PD4

4.9.1.14 void servo_turn (int *degrees*)

turns the servo motor a specific degrees

move to the position

4.9.1.15 void shaft_encoder_init (void)

Shaft encoder initialization.

4.9.1.16 void song_init ( )

Plays song 'Take on me'.

4.9.1.17 void stepper_init (void)

Initialize Stepper Motor.

4.9.1.18 float time2dist (unsigned int *Cycles*)

Converts the time that we measure into a distance in centimeters, takes in count from read.

4.9.1.19 void timer3_init (void)

4.9.1.20 void turn_clockwise (oi_t * *sensor,* int *degrees*)

turns clockwise by a given degree

4.9.1.21 void turn_counterclockwise (oi_t * *sensor,* int *degrees*)

turns clockwise by a given degree

4.9.1.22 void wait_ms (unsigned int *time_val*)

Blocks for a specified number of milliseconds.

Index