

Korsel ATTiny2313

1

Generated by Doxygen 1.7.1

Fri Feb 4 2011 13:47:45

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	V24Command Struct Reference	5
3.1.1	Detailed Description	5
3.1.2	Field Documentation	5
3.1.2.1	command	5
3.1.2.2	data	5
4	File Documentation	7
4.1	/home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_attiny2313/src/Korsel_BT.c File Reference	7
4.1.1	Define Documentation	9
4.1.1.1	button_left	9
4.1.1.2	button_middle	9
4.1.1.3	button_right	9
4.1.1.4	button_top	9
4.1.1.5	F_CPU	9
4.1.1.6	motor_left_direction	9
4.1.1.7	motor_right_direction	9
4.1.1.8	photo_sensor	9
4.1.1.9	PWM_motor_left	9
4.1.1.10	PWM_motor_right	9
4.1.1.11	RXD	9
4.1.1.12	TXD	9

4.1.1.13	USART_BAUD_RATE	9
4.1.1.14	V24_UBRR	9
4.1.2	Typedef Documentation	10
4.1.2.1	V24COMMAND	10
4.1.3	Function Documentation	10
4.1.3.1	delay	10
4.1.3.2	init_io	10
4.1.3.3	ISR	10
4.1.3.4	ISR	10
4.1.3.5	main	10
4.1.3.6	pwm	10
4.1.3.7	setup_interrupt	10
4.1.3.8	Stop_Motor	10
4.1.3.9	USART_Init	11
4.1.3.10	USART_Transmit	11
4.1.3.11	USART_Transmit_Command	11
4.1.4	Variable Documentation	11
4.1.4.1	LEFT_MOTOR_SPEED_BACKWARD_headerDetected	11
4.1.4.2	LEFT_MOTOR_SPEED_FORWARD_headerDetected	11
4.1.4.3	RIGHT_MOTOR_SPEED_BACKWARD_headerDetected	11
4.1.4.4	RIGHT_MOTOR_SPEED_FORWARD_headerDetected	11
4.2	/home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_attiny2313/src/v24_- commands.h File Reference	11
4.2.1	Define Documentation	12
4.2.1.1	BUTTON_PRESSED	12
4.2.1.2	FRONT_BUTTON_PRESSED	12
4.2.1.3	LEFT_BUTTON_PRESSED	12
4.2.1.4	LEFT_MOTOR_SPEED_BACKWARD	12
4.2.1.5	LEFT_MOTOR_SPEED_FORWARD	12
4.2.1.6	PHOTO_SENSOR	12
4.2.1.7	RIGHT_BUTTON_PRESSED	12
4.2.1.8	RIGHT_MOTOR_SPEED_BACKWARD	12
4.2.1.9	RIGHT_MOTOR_SPEED_FORWARD	12
4.2.1.10	TOP_BUTTON_PRESSED	12

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

[V24Command](#) (Command struct consisting of header and data) 5

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

/home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_attiny2313/src/ Korsel_BT.c . .	7
/home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_attiny2313/src/v24_- commands.h	11

Chapter 3

Data Structure Documentation

3.1 V24Command Struct Reference

Command struct consisting of header and data.

Data Fields

- unsigned char `command`
- char `data`

3.1.1 Detailed Description

Command struct consisting of header and data.

3.1.2 Field Documentation

3.1.2.1 unsigned char V24Command::command

3.1.2.2 char V24Command::data

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

- `/home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_attiny2313/src/Korsel_BT.c`

Chapter 4

File Documentation

4.1 /home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_attiny2313/src/Korsel_BT.c File Reference

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include "v24_commands.h"
#include <util/delay.h>
```

Data Structures

- struct [V24Command](#)
Command struct consisting of header and data.

Defines

- #define [F_CPU](#) 4000000UL
Define CPU frequency.
- #define [USART_BAUD_RATE](#) 9600
Debug Mode.
- #define [V24_UBRR](#) (F_CPU/(USART_BAUD_RATE*8L)-1)
Calculate and define UBRR.
- #define [PWM_motor_left](#) OCR1A
- #define [PWM_motor_right](#) OCR1B
- #define [motor_right_direction](#) PB1
- #define [motor_left_direction](#) PB2
- #define [RXD](#) PD0
- #define [TXD](#) PD1
- #define [photo_sensor](#) PD2

- #define `button_top` PD3
- #define `button_left` PD4
- #define `button_right` PD5
- #define `button_middle` PD6

Typedefs

- typedef struct `V24Command` `V24COMMAND`
Command struct consisting of header and data.

Functions

- void `init_io` (void)
initialize IO ports
- void `delay` (void)
delay of 100 ms
- void `pwm` (void)
Initialize motor outputs as PWM (pulse width modulation).
- void `setup_interrupt` (void)
Initialize interrupts.
- void `USART_Init` ()
Initialize USART for serial communication.
- void `USART_Transmit` (unsigned char data)
Transmit a char on USART.
- void `USART_Transmit_Command` (`V24COMMAND` *command)
Transmit a package consisting of a header and a value on USART.
- void `Stop_Motor` ()
If a contact button is pressed this method is called Resets motor speed to 0 and sends a "button pressed" command.
- `ISR` (`INT0_vect`)
ISR for photo sensor.
- `ISR` (`USART_RX_vect`)
ISR for USART receive.
- int `main` (void)
Main function.

Variables

- bool `LEFT_MOTOR_SPEED_FORWARD_headerDetected` = false
true if last package received was the protocol header LEFT_MOTOR_SPEED_FORWARD
- bool `LEFT_MOTOR_SPEED_BACKWARD_headerDetected` = false
true if last package received was the protocol header LEFT_MOTOR_SPEED_BACKWARD
- bool `RIGHT_MOTOR_SPEED_FORWARD_headerDetected` = false
true if last package received was the protocol header RIGHT_MOTOR_SPEED_FORWARD
- bool `RIGHT_MOTOR_SPEED_BACKWARD_headerDetected` = false
true if last package received was the protocol header RIGHT_MOTOR_SPEED_BACKWARD

4.1.1 Define Documentation

4.1.1.1 `#define button_left PD4`

4.1.1.2 `#define button_middle PD6`

4.1.1.3 `#define button_right PD5`

4.1.1.4 `#define button_top PD3`

4.1.1.5 `#define F_CPU 4000000UL`

Define CPU frequency.

4.1.1.6 `#define motor_left_direction PB2`

4.1.1.7 `#define motor_right_direction PB1`

4.1.1.8 `#define photo_sensor PD2`

4.1.1.9 `#define PWM_motor_left OCR1A`

4.1.1.10 `#define PWM_motor_right OCR1B`

4.1.1.11 `#define RXD PD0`

4.1.1.12 `#define TXD PD1`

4.1.1.13 `#define USART_BAUD_RATE 9600`

Debug Mode.

4.1.1.14 `#define V24_UBRR (F_CPU/(USART_BAUD_RATE*8L)-1)`

Calculate and define UBRR.

4.1.2 Typedef Documentation

4.1.2.1 typedef struct V24Command V24COMMAND

Command struct consisting of header and data.

4.1.3 Function Documentation

4.1.3.1 void delay (void)

delay of 100 ms

4.1.3.2 void init_io (void)

initialize IO ports

4.1.3.3 ISR (INT0_vect)

ISR for photo sensor.

this ISR is called if the photo sensor state changes from white to black or from black to white

4.1.3.4 ISR (USART_RX_vect)

ISR for USART receive.

A command consists of two chars. The first one is the header and the second one the speed value. The first time the ISR is called the char is read from the UDR. If a header was detected the headerDetected Flag is set. The next time the ISR is called the char read from UDR is a value for motor speed.

4.1.3.5 int main (void)

Main function.

Initializations and main loop

4.1.3.6 void pwm (void)

Initialize motor outputs as PWM (pulse width modulation).

4.1.3.7 void setup_interrupt (void)

Initialize interrupts.

4.1.3.8 void Stop_Motor ()

If a contact button is pressed this method is called Resets motor speed to 0 and sends a "button pressed" command.

4.2

/home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_attiny2313/src/v24_commands.h

File Reference

11

4.1.3.9 void USART_Init ()

Initialize USART for serial communication.

no parity, 1 stop bit, char size 8

4.1.3.10 void USART_Transmit (unsigned char *data*)

Transmit a char on USART.

4.1.3.11 void USART_Transmit_Command (V24COMMAND * *command*)

Transmit a package consisting of a header and a value on USART.

4.1.4 Variable Documentation

4.1.4.1 bool LEFT_MOTOR_SPEED_BACKWARD_headerDetected = false

true if last package received was the protocol header LEFT_MOTOR_SPEED_BACKWARD

4.1.4.2 bool LEFT_MOTOR_SPEED_FORWARD_headerDetected = false

true if last package received was the protocol header LEFT_MOTOR_SPEED_FORWARD

4.1.4.3 bool RIGHT_MOTOR_SPEED_BACKWARD_headerDetected = false

true if last package received was the protocol header RIGHT_MOTOR_SPEED_BACKWARD

4.1.4.4 bool RIGHT_MOTOR_SPEED_FORWARD_headerDetected = false

true if last package received was the protocol header RIGHT_MOTOR_SPEED_FORWARD

4.2 /home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_attiny2313/src/v24_commands.h File Reference

Defines

- #define LEFT_MOTOR_SPEED_FORWARD 0x01
- #define LEFT_MOTOR_SPEED_BACKWARD 0x11
- #define RIGHT_MOTOR_SPEED_FORWARD 0x02
- #define RIGHT_MOTOR_SPEED_BACKWARD 0x12
- #define PHOTO_SENSOR 0x22
- #define BUTTON_PRESSED 0x33
- #define TOP_BUTTON_PRESSED 0x03
- #define LEFT_BUTTON_PRESSED 0x04
- #define RIGHT_BUTTON_PRESSED 0x05
- #define FRONT_BUTTON_PRESSED 0x06

4.2.1 Define Documentation

4.2.1.1 #define BUTTON_PRESSED 0x33

4.2.1.2 #define FRONT_BUTTON_PRESSED 0x06

4.2.1.3 #define LEFT_BUTTON_PRESSED 0x04

4.2.1.4 #define LEFT_MOTOR_SPEED_BACKWARD 0x11

4.2.1.5 #define LEFT_MOTOR_SPEED_FORWARD 0x01

4.2.1.6 #define PHOTO_SENSOR 0x22

4.2.1.7 #define RIGHT_BUTTON_PRESSED 0x05

4.2.1.8 #define RIGHT_MOTOR_SPEED_BACKWARD 0x12

4.2.1.9 #define RIGHT_MOTOR_SPEED_FORWARD 0x02

4.2.1.10 #define TOP_BUTTON_PRESSED 0x03

Index

/home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_BT.c, [7](#)
attiny2313/src/Korsel_BT.c, [7](#)
/home/steffen/Dropbox/Bachelorarbeit/Programmierung/Korsel_BT.c, [10](#)
attiny2313/src/v24_commands.h, [11](#)

button_left
 Korsel_BT.c, [9](#)
button_middle
 Korsel_BT.c, [9](#)
BUTTON_PRESSED
 v24_commands.h, [12](#)
button_right
 Korsel_BT.c, [9](#)
button_top
 Korsel_BT.c, [9](#)

command
 V24Command, [5](#)

data
 V24Command, [5](#)

delay
 Korsel_BT.c, [10](#)

F_CPU
 Korsel_BT.c, [9](#)
FRONT_BUTTON_PRESSED
 v24_commands.h, [12](#)

init_io
 Korsel_BT.c, [10](#)
ISR
 Korsel_BT.c, [10](#)

Korsel_BT.c
 button_left, [9](#)
 button_middle, [9](#)
 button_right, [9](#)
 button_top, [9](#)
 delay, [10](#)
 F_CPU, [9](#)
 init_io, [10](#)
 ISR, [10](#)
 LEFT_MOTOR_SPEED_BACKWARD_
 headerDetected, [11](#)
 LEFT_MOTOR_SPEED_FORWARD_
 headerDetected, [11](#)
 motor_left_direction, [9](#)
 motor_right_direction, [9](#)
 photo_sensor, [9](#)
 pwm, [10](#)
 PWM_motor_left, [9](#)
 PWM_motor_right, [9](#)
 RIGHT_MOTOR_SPEED_BACKWARD_
 headerDetected, [11](#)
 RIGHT_MOTOR_SPEED_FORWARD_
 headerDetected, [11](#)
 RXD, [9](#)
 setup_interrupt, [10](#)
 Stop_Motor, [10](#)
 TXD, [9](#)
 USART_BAUD_RATE, [9](#)
 USART_Init, [10](#)
 USART_Transmit, [11](#)
 USART_Transmit_Command, [11](#)
 V24_UBRR, [9](#)
 V24COMMAND, [10](#)

LEFT_BUTTON_PRESSED
 v24_commands.h, [12](#)
LEFT_MOTOR_SPEED_BACKWARD
 v24_commands.h, [12](#)
LEFT_MOTOR_SPEED_BACKWARD_
 headerDetected
 Korsel_BT.c, [11](#)
LEFT_MOTOR_SPEED_FORWARD
 v24_commands.h, [12](#)
LEFT_MOTOR_SPEED_FORWARD_
 headerDetected
 Korsel_BT.c, [11](#)

main
 Korsel_BT.c, [10](#)
motor_left_direction
 Korsel_BT.c, [9](#)
motor_right_direction
 Korsel_BT.c, [9](#)

PHOTO_SENSOR

v24_commands.h, [12](#)
 photo_sensor
 Korsel_BT.c, [9](#)
 pwm
 Korsel_BT.c, [10](#)
 PWM_motor_left
 Korsel_BT.c, [9](#)
 PWM_motor_right
 Korsel_BT.c, [9](#)

 RIGHT_BUTTON_PRESSED
 v24_commands.h, [12](#)
 RIGHT_MOTOR_SPEED_BACKWARD
 v24_commands.h, [12](#)
 RIGHT_MOTOR_SPEED_BACKWARD_-
 headerDetected
 Korsel_BT.c, [11](#)
 RIGHT_MOTOR_SPEED_FORWARD
 v24_commands.h, [12](#)
 RIGHT_MOTOR_SPEED_FORWARD_-
 headerDetected
 Korsel_BT.c, [11](#)
 RXD
 Korsel_BT.c, [9](#)

 setup_interrupt
 Korsel_BT.c, [10](#)
 Stop_Motor
 Korsel_BT.c, [10](#)

 TOP_BUTTON_PRESSED
 v24_commands.h, [12](#)
 TXD
 Korsel_BT.c, [9](#)

 USART_BAUD_RATE
 Korsel_BT.c, [9](#)
 USART_Init
 Korsel_BT.c, [10](#)
 USART_Transmit
 Korsel_BT.c, [11](#)
 USART_Transmit_Command
 Korsel_BT.c, [11](#)

 v24_commands.h
 BUTTON_PRESSED, [12](#)
 FRONT_BUTTON_PRESSED, [12](#)
 LEFT_BUTTON_PRESSED, [12](#)
 LEFT_MOTOR_SPEED_BACKWARD, [12](#)
 LEFT_MOTOR_SPEED_FORWARD, [12](#)
 PHOTO_SENSOR, [12](#)
 RIGHT_BUTTON_PRESSED, [12](#)
 RIGHT_MOTOR_SPEED_BACKWARD, [12](#)
 RIGHT_MOTOR_SPEED_FORWARD, [12](#)
 TOP_BUTTON_PRESSED, [12](#)
 V24_UBRR
 Korsel_BT.c, [9](#)
 V24COMMAND
 Korsel_BT.c, [10](#)
 V24Command, [5](#)
 command, [5](#)
 data, [5](#)