SAE Auto Shifter

Version
3/14/2012 5:41:00 AM

Table of Contents

Main Page

Author:

Rocky Gray Jr.

Introduction

Todo:

Write an introduction This will be the text for the introduction... continued... continued... Second paragraph for intro if needed... cont... cont...

Files used for this project:

- **SAE_AutoShifter.h**
- **delay_rg.h**
- **main.c**

Todo List

page Main Page

Write an introduction This will be the text for the introduction... continued... continued... Second paragraph for intro if needed... cont... cont...

File Index

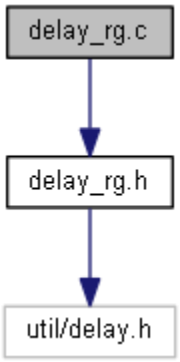
File List

Here is a list of all files with brief descriptions:

File Documentation

delay_rg.c File Reference

This file defines delay functions that extend the usefullness of the avr-gcc delay functions.
`#include "delay_rg.h"`
Include dependency graph for delay_rg.c:



Functions

- void **delay_ms** (uint16_t ms)
  Delays for `ms` *milliseconds.*
- void **delay_us** (uint16_t us)
  Delays for `us` *microseconds.*

Detailed Description

This file defines delay functions that extend the usefullness of the avr-gcc delay functions.

Author:
Rocky Gray Jr.

Date:
2/6/2012

Function Documentation

void delay_ms (uint16_t *ms*)

Delays for `ms` milliseconds.

Allows for longer delays. The longest delay that _delay_ms() can provide is 262.14ms / **F_CPU(in MHz)**.

Parameters:

ms	length of delay in miliseconds

Returns:

none

Here is the caller graph for this function:

void delay_us (uint16_t *us*)

Delays for `us` microseconds.

Allows for longer delays. The longest delay that _delay_us() can provide is 768us / **F_CPU(in MHz)**.

Parameters:

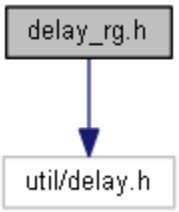
us	length of delay in microseconds

Returns:

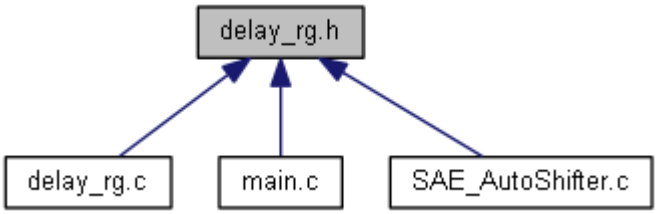
none

delay_rg.h File Reference

This file defines delay functions that extend the usefullness of the avr-gcc delay functions.
`#include <util/delay.h>`
Include dependency graph for delay_rg.h:



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



Defines

- #define **DLAY_MAX_MS** (262.14/**F_CPU**)-1
- #define **DLAY_MAX_US** (768/**F_CPU**)-1

Functions

- void **delay_ms** (uint16_t ms)
  Delays for `ms` *milliseconds.*
- void **delay_us** (uint16_t us)
  Delays for `us` *microseconds.*

Detailed Description

This file defines delay functions that extend the usefullness of the avr-gcc delay functions.

Author:
Rocky Gray Jr.

Date:
2/6/2012

Define Documentation

#define DLAY_MAX_MS (262.14/F_CPU)-1

#define DLAY_MAX_US (768/F_CPU)-1

Function Documentation

void delay_ms (uint16_t *ms*)

Delays for `ms` milliseconds.

Allows for longer delays. The longest delay that _delay_ms() can provide is 262.14ms / **F_CPU(in MHz)**.

Parameters:

ms	length of delay in miliseconds

Returns:

none

Here is the caller graph for this function:

void delay_us (uint16_t *us*)

Delays for `us` microseconds.

Allows for longer delays. The longest delay that _delay_us() can provide is 768us / **F_CPU(in MHz)**.

Parameters:

us	length of delay in microseconds

Returns:

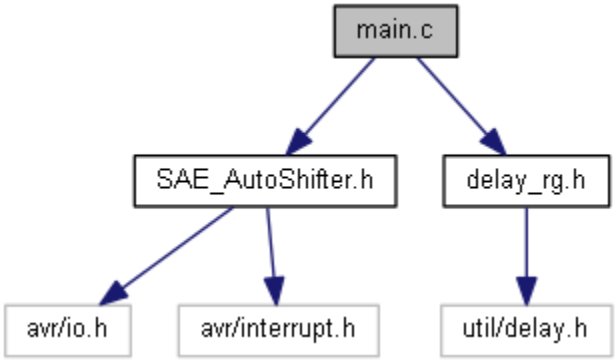
none

main.c File Reference

This file defines the main task.

```
#include "SAE_AutoShifter.h"
#include "delay_rg.h"
```

Include dependency graph for main.c:



Defines

- #define **F_CPU** 16000000UL

Functions

- int **main** (void)
  The main task for the autoshifter.

Detailed Description

This file defines the main task.

Author:
Rocky Gray Jr.

Date:
3/11/2012

Define Documentation

#define F_CPU 16000000UL

Function Documentation

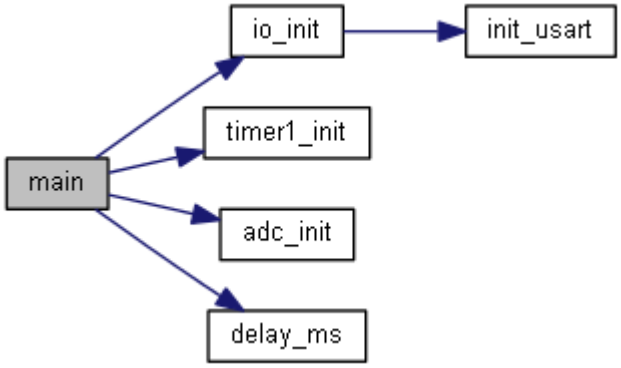
int main (void)

The main task for the autoshifter.

Returns:

This task never returns

Here is the call graph for this function:



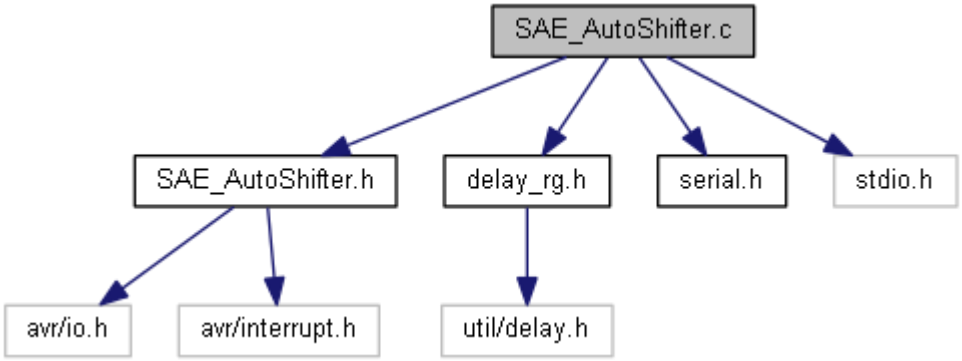
mainpage.h File Reference

SAE_AutoShifter.c File Reference

This file defines the main task and ISRs.

```
#include "SAE_AutoShifter.h"
#include "delay_rg.h"
#include "serial.h"
#include <stdio.h>
```

Include dependency graph for SAE_AutoShifter.c:



Functions

- void **io_init** (void)
  Initialize i/o ports and timer.
- void **timer1_init** (void)
  Initialize Timer 1.
- void **adc_init** (void)
  Initialize the ADC.
- **ISR** (TIMER1_COMPA_vect)
  *Toggle the LED if **USHIFT_PIN** is pressed.*
- **ISR** (ADC_vect)
  ADC Interrupt Service Routine.

Detailed Description

This file defines the main task and ISRs.

Author:

Rocky Gray Jr.

Date:

2/6/2012

Function Documentation

void adc_init (void)

Initialize the ADC.

ADC 0:

- Prescaler set to 128 (16MHz/128 = 125kHz)
- ADC result is left adjusted (Just read ADCH)
- Free-Running mode enabled
- Interrupt enabled

Returns:

none

Here is the caller graph for this function:

void io_init (void)

Initialize i/o ports and timer.

Inputs:

- User Buttons
  - **BTN_IP_DDR** => set **USHIFT_PIN** and **DSHIFT_PIN** as input
  - **BTN_IP_PORT** => set pullups for **USHIFT_PIN** and **DSHIFT_PIN**
- Tachometer Input
  - **TACH_IP_DDR** => set **TACH_PIN** as input
  - **TACH_IP_PORT** => set pullups for **TACH_PIN**
-

Outputs:

- Solenoid Output
  - **SOLEN_OP_DDR** => set **SOLEN_DN** and **SOLEN_UP** as input
  - **SOLEN_OP_PORT** => initialize **SOLEN_DN** and **SOLEN_UP** to 0
-

Returns:

none

Here is the call graph for this function:

Here is the caller graph for this function:

ISR (TIMER1_COMPA_vect)

Toggle the LED if **USHIFT_PIN** is pressed.

Fires at **PIN_CHECK_FREQ** Hz. (1ms)

Returns:

reti

ISR (ADC_vect)

ADC Interrupt Service Routine.

Fires at 16MHz/128 = 125kHz

Returns:

reti

void timer1_init (void)

Initialize Timer 1.

Timer 1:

- Configure Timer 1 for CTC (Clear Timer on Compare)
- Prescaler set to 1 (16MHz/1)
- Fires every **PIN_CHECK_FREQ** Hz (1kHz = 1ms)

Returns:

none

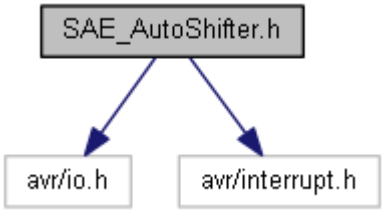
Here is the caller graph for this function:

SAE_AutoShifter.h File Reference

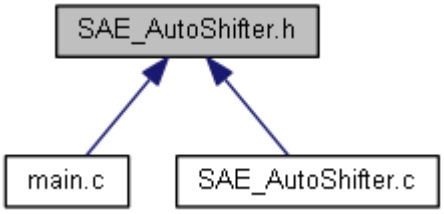
This header file contains constant definitions and function prototypes.

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

Include dependency graph for SAE_AutoShifter.h:



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



Defines

Port Definitions*Mappings for I/O ports*

- #define **BTN_IP_DDR** DDRD
- #define **BTN_IP_PORT** PORTD
- #define **BTN_IP_PIN** PIND
- #define **USHIFT_PIN** PD2
- #define **DSHIFT_PIN** PD3
- #define **TACH_IP_DDR** DDRC
- #define **TACH_IP_PIN** PINC
- #define **TACH_IP_PORT** PORTC
- #define **TACH_PIN** PC0
- #define **SOLEN_OP_DDR** DDRB
- #define **SOLEN_OP_PORT** PORTB
- #define **SOLEN_OP_PIN** PINB
- #define **SOLEN_DN** PB1
- #define **SOLEN_UP** PB2

Timer Definitions*Definitions to make setting up the timer easier*

- #define **PIN_CHECK_FREQ** 1000
- #define **PRESCALER** 1
- #define **TIMER1_COUNT** ((**F_CPU**/(**PIN_CHECK_FREQ***PRESCALER**))-1)

Button States*Defines for the states of the buttons*

- #define **RELEASED** 0
- #define **PRESSED** 1

Functions

- void **io_init** (void)
  Initialize i/o ports and timer.
- void **timer1_init** (void)
  Initialize Timer 1.
- void **adc_init** (void)
  Initialize the ADC.
- **ISR** (TIMER1_COMPA_vect)
  *Toggle the LED if **USHIFT_PIN** is pressed.*
- **ISR** (ADC_vect)
  ADC Interrupt Service Routine.

Variables

Port Values*Used to get the values from the I/O ports*

- uint8_t **btn_pins**
  *Only the **USHIFT_PIN** and **DSHIFT_PIN** pins are active.*
- uint8_t **tach_pin**
  *Only the **TACH_PIN** pin is active.*
- uint8_t **solen_port**
  *Only the **SOLEN_DN** and **SOLEN_UP** pins are active.*

Button Variables*The states of the buttons and debouncer values.*

- uint8_t **up_shift_btn**
  The state of the Up Shift Button.
- uint8_t **dn_shift_btn**
  The state of the Down Shift Button.
- uint16_t **uPressCount**
  Up button debouncer value.
- uint16_t **dPressCount**
  Down button debouncer value.

Detailed Description

This header file contains constant definitions and function prototypes.

This project uses the Arduino Uno board. It uses the ATMEGA 328P microcontroller with a 16MHz crystal oscillator. Pin mappings described below should only be used for this board or ones with the same pin mappings

Author:

Rocky Gray Jr.

Date:

3/7/2012

Define Documentation

#define BTN_IP_DDR DDRD

Maps to DDRD

#define BTN_IP_PIN PIND

Maps to PIND

#define BTN_IP_PORT PORTD

Maps to PORTD

#define DSHIFT_PIN PD3

Downshift

- Connect to Digital Pin 3 (PIND3)

#define PIN_CHECK_FREQ 1000

Button pin check frequency in Hz

#define PRESCALER 1

Prescaler needed for longer timer.

Valid values:

- 1 (no prescaling)
- 8
- 64
- 256
- 1024

#define PRESSED 1

Button is pressed (1)

#define RELEASED 0

Button is released (0)

#define SOLEN_DN PB1

Solenoid Down

- Connect to Digital Pin 9(PWM) (PINB1)
  - Shifts down, pushes solenoid out
-

#define SOLEN_OP_DDR DDRB

Maps to DDRB

#define SOLEN_OP_PIN PINB

Maps to PINB

#define SOLEN_OP_PORT PORTB

Maps to PORTB

#define SOLEN_UP PB2

Solenoid Up

- Connect to Digital Pin 10(PWM) (PINB2)
  - Shifts up, pulls solenoid in
-

#define TACH_IP_DDR DDRC

Maps to DDRC

#define TACH_IP_PIN PINC

Maps to PINC

#define TACH_IP_PORT PORTC

Maps to PORTC

#define TACH_PIN PC0

Tachometer

- Connect to Analog Input 0 (PINC0)

#define TIMER1_COUNT ((F_CPU/(PIN_CHECK_FREQ*PRESCALER))-1)

Computes the value the timer should count to.

#define USHIFT_PIN PD2

Upshift

- Connect to Digital Pin 2 (PIND2)

Function Documentation

void adc_init (void)

Initialize the ADC.

ADC 0:

- Prescaler set to 128 (16MHz/128 = 125kHz)
- ADC result is left adjusted (Just read ADCH)
- Free-Running mode enabled
- Interrupt enabled

Returns:

none

Here is the caller graph for this function:

void io_init (void)

Initialize i/o ports and timer.

Inputs:

- User Buttons
  - **BTN_IP_DDR** => set **USHIFT_PIN** and **DSHIFT_PIN** as input
  - **BTN_IP_PORT** => set pullups for **USHIFT_PIN** and **DSHIFT_PIN**
- Tachometer Input
  - **TACH_IP_DDR** => set **TACH_PIN** as input
  - **TACH_IP_PORT** => set pullups for **TACH_PIN**
-

Outputs:

- Solenoid Output
  - **SOLEN_OP_DDR** => set **SOLEN_DN** and **SOLEN_UP** as input
  - **SOLEN_OP_PORT** => initialize **SOLEN_DN** and **SOLEN_UP** to 0
-

Returns:

none

Here is the call graph for this function:

Here is the caller graph for this function:

ISR (TIMER1_COMPA_vect)

Toggle the LED if **USHIFT_PIN** is pressed.

Fires at **PIN_CHECK_FREQ** Hz. (1ms)

Returns:

reti

ISR (ADC_vect)

ADC Interrupt Service Routine.

Fires at 16MHz/128 = 125kHz

Returns:

reti

void timer1_init (void)

Initialize Timer 1.

Timer 1:

- Configure Timer 1 for CTC (Clear Timer on Compare)
- Prescaler set to 1 (16MHz/1)
- Fires every **PIN_CHECK_FREQ** Hz (1kHz = 1ms)

Returns:

none

Here is the caller graph for this function:

Variable Documentation

uint8_t btn_pins

Only the **USHIFT_PIN** and **DSHIFT_PIN** pins are active.

uint8_t dn_shift_btn

The state of the Down Shift Button.

uint16_t dPressCount

Down button debouncer value.

uint8_t solen_port

Only the **SOLEN_DN** and **SOLEN_UP** pins are active.

uint8_t tach_pin

Only the **TACH_PIN** pin is active.

uint8_t up_shift_btn

The state of the Up Shift Button.

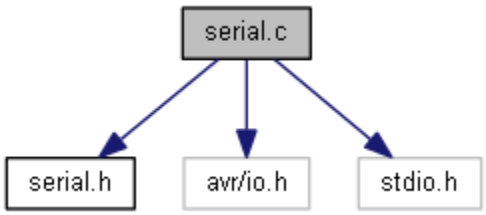
uint16_t uPressCount

Up button debouncer value.

serial.c File Reference

```
#include "serial.h"
#include "avr/io.h"
#include <stdio.h>
```

Include dependency graph for serial.c:



Functions

- void **init_usart** (unsigned int baudrate)

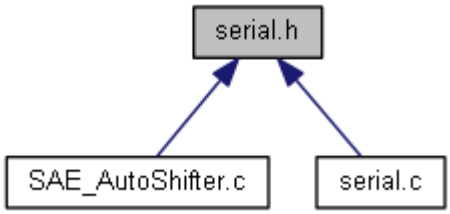
Function Documentation

void init_usart (unsigned int *baudrate*)

Here is the caller graph for this function:

serial.h File Reference

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



Defines

- #define **LF** '\n' /* line feed */
- #define **CR** '\r' /* carriage return */
- #define **UBRR_V**(Baud) ((**F_CPU**/16/Baud) - 1)
- #define **Baud115200** (**UBRR_V**(115200))
- #define **Baud57600** (**UBRR_V**(57600))
- #define **Baud38400** (**UBRR_V**(38400))
- #define **Baud19200** (**UBRR_V**(19200))
- #define **Baud9600** (**UBRR_V**(9600))

Functions

- void **init_usart** (unsigned int baudrate)

Define Documentation

#define Baud115200 (UBRR_V(115200))

#define Baud19200 (UBRR_V(19200))

#define Baud38400 (UBRR_V(38400))

#define Baud57600 (UBRR_V(57600))

#define Baud9600 (UBRR_V(9600))

#define CR '\r' /* carriage return */

#define LF '\n' /* line feed */

#define UBRR_V(Baud) ((F_CPU/16/Baud) - 1)

Function Documentation

void init_usart (unsigned int *baudrate*)

Here is the caller graph for this function:

Index