STM8S: Timer 2 with Overflow Interrupt

Jan 20, 2018 — by Junaid in Electronics

In previous post I showed how we can use Timer 2 as simple counter. In this post I am going to show how we can program to have interrupt when its value overflow. As simple experiment, we will toggle an LED in each second within the interrupt handler (also called ISR, Interrupt Service Routine).

Calculation

As mentioned in previous post, default system clock frequency will be 2MHz and by default Timer 2 will also run with same frequency as system.

Each tick of system clock will be 1/2,000,000 = 0.0000005 seconds or 0.5 micro seconds. If we set 128 as prescaler for Timer 2, then timer 2 counter will increment in each 64µs (0.5µs x 128). So, it will be 1 second when timer counter reaches at 15625 (1,000,000/64).

Program

LED is connected at PB5 pin (5th pin of port B). We create `timer_isr()` as Timer 2 overflow handler.

```
#include <stdint.h>
#include <stm8s.h>

// Default system clock will be 2MHz.
```

```
// We set 128 as prescaler, then each tick of timer 2 will
be in 64 micro seconds.
// So, timer will generate overflow interrupt in each
second,
// when counter reaches at 15625 (1S/64uS) which is set in
auto reload register.
// There are two 8 bit regisgers to hold 16 bit value for
ARR. So, we create a 16
// bit unsigned number.
const uint16_t reload_value = 15625;

void timer_isr() __interrupt(TIM2_OVF_ISR) {
    PB_ODR ^= 1 << PB5; // Toggle PB5 output
    TIM2_SR1 &= ~(1 << TIM2_SR1_UIF); // Clear interrupt
flag
}

void main() {
    enable_interrupts();

    PB_DDR |= 1 << PB5; // 0x00001000 PB5 is now output
    PB_CR1 |= 1 << PB5; // 0x00001000 PB5 is now pushpull

    TIM2_PSCR = 0b00000111; //  Prescaler = 128
    // Fill auto reload registers.
    // We need to put MSB and LSB in separate 8 bit
registers.
    // Also, as per datasheet, we have to put value in
ARRH first, then in ARRL.
    TIM2_ARRH = reload_value >> 8;
    TIM2_ARRL = reload_value & 0x00FF;

    TIM2_IER |= (1 << TIM2_IER_UIE); // Enable Update
Interrupt
    TIM2_CR1 |= (1 << TIM2_CR1_CEN); // Enable TIM2

    while (1) {
        // do nothing
```

```
        }
    }
```

Brief explanation for the program:

- Set PB5 pin as output.
- Set the prescaler value, 128, for Timer 2
- Set required value (here 15625) in Auto Reload Registers (ARR) of Timer 2
- Enable update interrupt on Timer 2. ISR will be called when counter reaches value in reload registers.
- Enable Timer 2
- Within ISR, `timer_isr()`, toggle LED and clear interrupt flag.

8-bit electronics embedded interrupt isr linux microcontroller sdcc stm8 stm8s timer

---

Comments

3 responses to “STM8S: Timer 2 with Overflow Interrupt”

Vasant Pailwan
March 9, 2024

void timer_isr() __interrupt(TIM2_OVF_ISR){

TIM2_SR1 &= ~(1 << TIM2_SR1_UIF);

above statements gives error- missing ;

Reply

Junaid
March 11, 2024

Either you made mistake when copying the code or you are using a different compiler. I was using the SDCC (https://sdcc.sourceforge.net/) for compiling above

program.

Reply

Vasant Pailwan
April 23, 2024

How to create timer_isr() as Timer 2 overflow handler?

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

Post Comment

← Previous: STM8S: Use Timer 2 as Simple Counter to Blink LED (without Interrupt)

Next: Bootloader-less Programming Arduino Pro Mini →

Junaid's Blog