Soen 422

Dr Michel de Champlain

Lab # 4

Jason Klimock
26322298

1 Discussion

1.1 Task 1

Task 1 asks us to write an ISR (interrupt service routine) that turns off an LED for 5 seconds if PD2 is connected or disconnected. In completing this task we need to enable interrupts in the EIMSK (external interrupt mask) register by setting bit 0 (INT0) to 1. This bit vecotr it then passed into the ISR function using INT0_Vect. We also need to set the 0 bit(ISC00) on the EICRA(External control register A) to 1. With the EICRA set and when the global interrupt bit is set, interrupts from INT) will be allowed. In the ISR funtion we first disable interrupts with cli() and the turn off the LED and ten reenable interrupts. The five second timer is handled in the main loop, which turns the LED back on after the 5 second wait.

1.2 Task 2

Task 2 asks us to wirte a program that toggles pin D2 as out put with a 1 second wait. We are then to use pin D8 as a switch to turn the led off or on, when a jumper cable from 8 is plugged into or unplugged from pin D2. First we need to set bit 0 (PCINT0) PCMSK0 (pinchange mask) register to 1 inorder to enable interrupts on I/O PB0. The we set bit 0 (PCIE) to 1 on PCICR (Pinchange interrupt control) register inorder to enable pinchange interrupts. When we enbale the global interrupt bit interrupts will be enabled. We pass PCINT0_vect to the ISR because that provides the information for which i/o pin has the interrupt. the ISR then checks if PINB0 is on or off (plugged into D2 or not) and turns the LEd off or on accordingly.

1.3 task 3

Task 3 asks us to wite a 8 bit timer to generate an interrupt every 4 ms flash the led every 0.5 seconds. First we need to set CTC (clear timer on Compare) mode, which we do by setting bit 1 (WGM01) of the TCCR0A(Timer/counter control register 0A) to 1. The we need to select a clock set a clock source for the timer counter with the correct prescaler of 256, which we do by setting but 2(CS02) of TCCR0B to 1. The we enable a compare match register for the timer/counter0 by setting bit 1 (OCIE0A) to 1. Finally we want to set the compare value in OCR0A by calculting cpu clock speed hz divided by 256 (prescaler value) and wultiplying this time the time interval, 4 ms in this case. Passing the TIMERO_COMPA_vect to the ISR function will create an interrupt every 4 ms. Creating a global volatile count variable allows the isr to cound every interrupt cycle up to a certain value. In this case we want 500 ms, so I increment the timer by 4 (for 4 ms) every time there is an interrupt. On the 125th interrupt the count will reach 500 and will toggle the led on or off and will set the count variable to 0. This will cause the LED to blink every 0.5 seconds

2 Conclusion

Interrupt service rountines provide a fast, hardware based solutuion ot triggering certain events. Since we are relying on the hardware timer rather than a software solutuion, events can be written as an ISR instead of creating a loop with a condition to check a condition every loop iteration. This can reduce th eover all overhead of the program. In th ecase of timers, we can create a much more precise time by using the timer registers becuase time is directly calculated from the cpu cycle. This offers more precision becuase it is directly tied to hardware, rather than having a software based solution, whihc could be less reliable. Reading the Official documentation is key to understanding what the registers do and ehihc bits need to be set in order to acomplish each task.