

Jonathan W. Valvano

October 19, 2007, 1:00pm-1:50pm.

**(5) Question 1.** Breakpoints are highly intrusive

- 1) because the time to service a breakpoint is long compared to the time to execute the software itself
- 2) because the hardware continues to change while the software is halted

**(10) Question 2.** Count the number of times that the ISR is invoked while the system is executing **Stuff1**.

```

unsigned short Count=0; // number of times ISR interrupt fun1
int bEntered=0; // true if main program has entered Stuff1
void fun1(void){
    bEntered = 1; // entering Stuff1
    Stuff1();
    bEntered = 0; // leaving Stuff1
}
interrupt 7 void RTIhandler(void){
    if(bEntered) Count++; // assumes it happens less than 65535 times
    CRGFLG = 0x80; // acknowledge, clear RTIF flag
    Stuff2();
}

```

**(10) Question 3.** No, there is no critical section, because the read-modify-write operation in the main program is atomic. It does not matter where the interrupt occurs in the main program the value of **PTT** will always be valid. This instruction once started will complete before an interrupt is serviced.

```

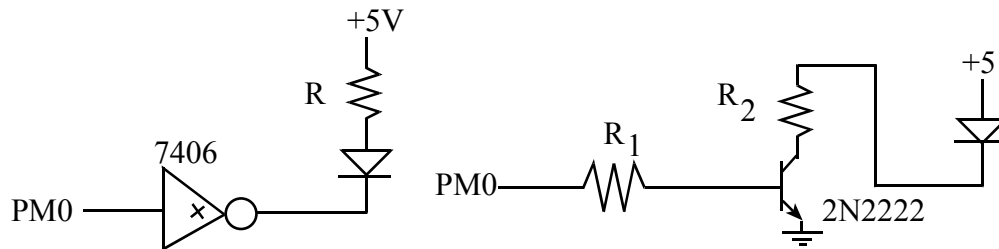
    bset $0240, #01

```

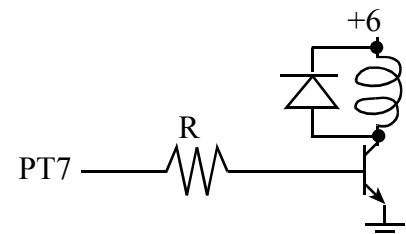
**(10) Question 4.** Interface latency is the time from when **TDRE** is set (by the hardware) to signify the output device is idle until the time the software writes **SCIDRL** (giving it more output data). The *device latency* is the time from writing the **SCIDRL** (the time the software is asking for an output) until the time **TDRE** is set (by the hardware) signifying the I/O request is satisfied.

**(5) Question 5.** The range of 8-bit signed numbers is -128 to +127, thus **-1.28V to +1.27V**

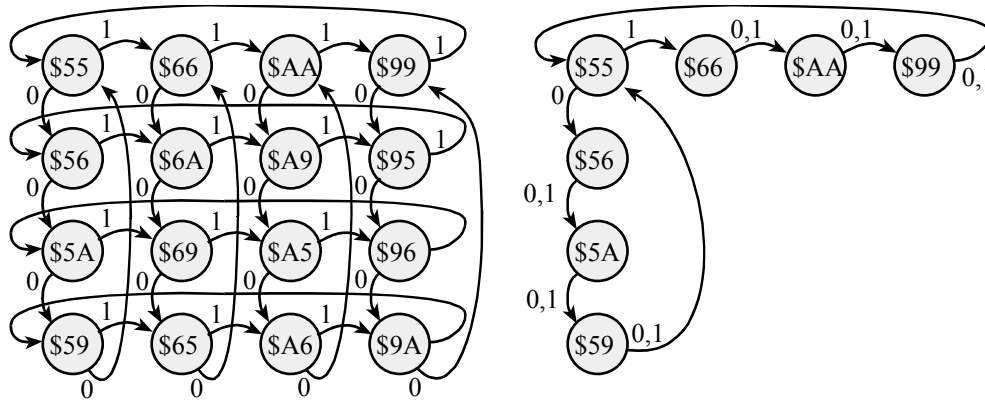
**(10) Question 6.** 40 mA will require either the 2N2222 or the 7406. The output low voltage of the 7406 is 0.5V. The resistor value is calculated as  $R = (5 - 2.5 - 0.5) / 0.04 = 2.0V / 0.040A = 50\Omega$ .



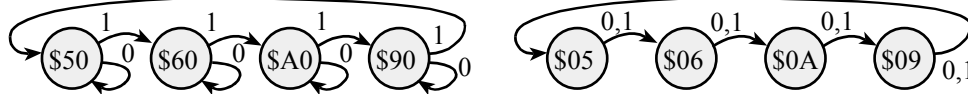
**(10) Question 7.** A 6V power is connected to one side of the solenoid, and the other side is switched to ground. To handle the 100 mA, the 2N2222 can be used (because it can handle up to 500 mA of  $I_{CE}$ ). Because the current gain is 100 ( $h_{fe}$ ) the base current needs to be  $100mA/100 = 1mA$ . The  $I_{OH}$  of the 9S12 can supply this 1mA ( $I_{OH}$  can be up to 10 mA). Because the  $V_{OH}$  of the 9S12 is 4.2V (or greater) and the  $V_{BE}$  of the 2N2222 is 0.6V (or less), the resistor from the 9S12 to the 2N2222 base must be less than  $(4.2 - 0.6V) / 1mA = 3.6 / 0.001 = 3.6\text{ k}\Omega$ . The 1N914 diode provides protection against back EMF.



**(20) Question 8.** For each motor there are 4 valid outputs. Therefore, for the two motors, there are 16 valid outputs. When the input is 1, both motors sequence 5,6,10,9. When the input is 0, just the right motor changes. If you are willing to delay up to 40ms, the FSM on the right could be used.



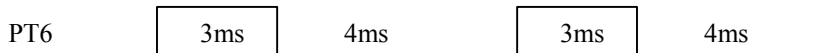
A third alternative is to implement two totally separate machines, and have the controller combine outputs. Again, the right motor always sequences and the left motor sequences only when the input is 1.



**(20) Question 9.** Use output compare 6 interrupts to generate the following periodic output on PT6.

Part a) Show the ritual that is executed once.

```
void OC6_Init(void) {
    DDRT |= 0x40; // PT6 output
    PTT &= ~0x40; // PT6 initially low
    TIOS |= 0x40; // activate TC6 as output compare
    TSCR1 = 0x80; // Enable TCNT 1 MHz in run mode
    TSCR2 = 0x02; // divide by 4 TCNT prescale, 1us
    TIE |= 0x40; // arm OC6
    TC6 = TCNT+4000; // first interrupt in 4ms
    asm cli
}
```



Part b) Show the output compare interrupt service routine that outputs to PT6.

```
interrupt 14 void TC6handler(void) {
    TFLG1 = 0x40; // acknowledge OC6
    if (PTT & 0x40) {
        PTT &= ~0x20; // PT6 used to be high, now low
        TC6 = TC6+4000; // low for the next 4ms
    } else {
        PTT |= 0x40; // PT6 used to be low, now high
        TC6 = TC6+3000; // high for the next 3ms
    }
}
```