

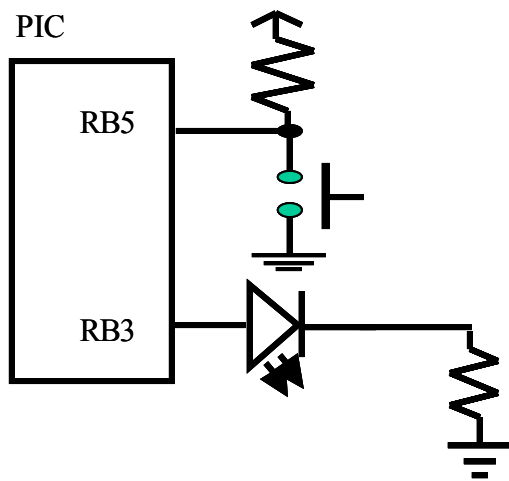
You may use a calculator and the provided reference materials. If a binary result is required, give the value in HEX.

Part I: (55 pts) You must answer all of these questions.

- a. (5 pts) Write C code will configure PORTB as shown in problem (b) – RB5 an input, RB3 an output. I don't care how the other bits of PORTB are configured.

```
TRISB = 0x20;
```

- b. (10 pts) Write C code that after reset, waits for a push and release of the pushbutton, then enters a loop where the LED blinks rapidly as long as the pushbutton is pressed, else it blinks slowly. Assume two subroutines, `long_delay()` and `short_delay()` that are available for use as time delays. You do NOT have to write code to configure PORTB, that is what problem (a) does.



```
while (RB5); //wait for switch to be pressed
while (!RB5); //wait for release
while(1) {
  while(!RB5) { //switch pressed
    RB3 = 1;
    short_delay();
    RB3 = 0;
    short_delay();
  }
  while(RB5) { //switch released
    RB3 = 1;
    long_delay();
    RB3 = 0;
    long_delay();
  }
}
```

- c. (5 pts) Write a C code fragment that prints out a message “WDT Reset” if WDT reset occurs, or prints out “WDT Wake-up” if WDT wakeup occurs.

```
if(!TO) {
    if(PD) printf(“WDT Reset\n”);
    else printf(“WDT Wake-up\n”);
}
```

- d. (10 pts) Assume we are using interrupt-driven IO and a circular buffer for serial data OUTPUT. Write a `putch()` subroutine that checks the TXIF flag; if ‘1’, then write the character to the TXREG and return. If TXIF is ‘0’, then check the circular buffer; if placing data into the buffer would NOT cause buffer overrun, then place the data into the buffer and return. If placing data into the buffer would cause buffer overrun, then wait for this condition to go away (an interrupt service routine is taking data out of the buffer), place the data in the buffer, and return. The definitions for the buffer and the pointers are shown below. Use the same definitions for buffer overrun, placing data into the buffer, and taking data out of the buffer as was used for lab #8.

```
char buf[16]; // circular buffer
char head,tail;
putch(c)
char c;
{
    // you complete this
```

You cannot do head++ here as this will cause the ISR to think the buffer has one extra character in it that it does not have!!!!

```
char tmp_head;

if (TXIF) {
    TXREG = c;
    return;
}else {
    tmp_head = head+1;
    if (tmp_head == 16) tmp_head = 0;
    while(tmp_head == tail); //overflow, wait for ISR to
                            // read buffer

    head = tmp_head;
    buf[head] = c; // put character in buffer
    return;
}
```

- e. (5 pts) Assume a PIC18 is consuming 20 mA at 5V and 40 MHz. If the voltage is lowered to 4.0V and the frequency reduced to 20 MHz, what is the predicted new current consumption based on theory?

$$\begin{aligned} 20 \text{ mA} &\sim 5V * 5V * 40 \text{ Mhz} * C && ; \text{ solve for } C, C \sim 20\text{mA}/(25*40 \text{ MHz}) \\ y \text{ mA} &\sim 4V * 4V * 20 \text{ Mhz} * C && ; \text{ plug in } C \\ y \text{ mA} &\sim 16/25 * 20/40 * 20 \text{ mA} = 6.4 \text{ mA} \end{aligned}$$

- f. (5 pts) Assuming high speed mode, a 12 MHz FOSC, and a desired baud rate of 19200, what value has to be written to the SPBRG register? Do NOT give me a fractional value!!!!

$$\text{SPBRG} = [12\text{e}6/(16*19200)] - 1 = 38$$

- g. (5 pts) What two errors can occur in asynchronous reception of the UART? How can these error bits be cleared? (Writing to them does not clear them!).

Framing Error (FERR), Overrun Error (OERR), clear these bits by setting CREN = 0, then setting CREN back to 1. Using Reset or power-off is not a valid option.

- h. (5 pts) Why is interrupt driven IO superior to polled IO? Be specific.

Polled IO can miss data by not checking data availability often enough, or wastes processor time by checking too often. Interrupt driven IO only does an IO operation when data is available by forcing the processor to jump to an interrupt service routine when the IO requires servicing.

Part II: (45 pts) Answer 15 out of the next 18 questions. Cross out the 3 questions that you do not want graded. Each question is worth 3 pts, there is no partial credit.

1. Assuming even parity, what is the parity bit for the 7-bit value 0x45?

0x45 = 1000101, there is an odd number of '1's, so the parity bit is '1' to make the total number of '1's even.

2. Write a single C statement that will mask (disable) all interrupts.

GIE = 0;

3. What instruction is used to prevent the Watchdog timer from going off?

CLRWDT is the PIC instruction that clears the WDT back to zero, preventing it from generating an interrupt. Setting the SWDTEN bit to zero disables the entire timer, this is not what is used to clear the WDT when the timer is in use.

4. What is the advantage of sleep mode?

Reduces power consumption dramatically by stopping the clock.

5. What is the bit time in microseconds for the baud rate 76,800?

$1/76800 = 1.3 \times 10^{-5} \text{ s} = 13 \text{ us}$

6. Write a single C statement that will disable all interrupt priorities. (make all interrupts look like high priority interrupts)

IPEN = 0

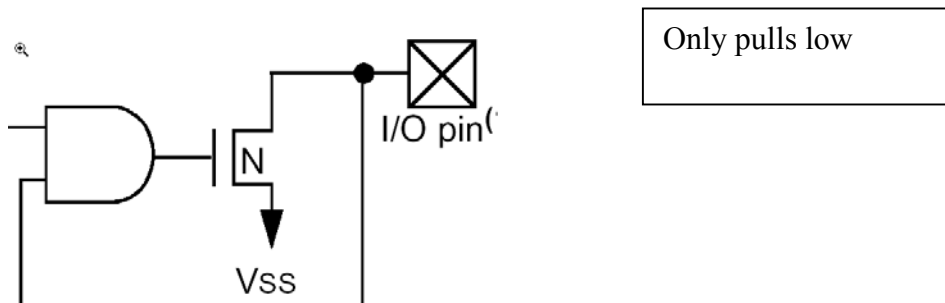
7. If both high and low priority interrupts are being used, why can't the shadow registers be used for low priority interrupts?

A high-priority interrupt that interrupts a low-priority interrupt will overwrite the shadow register contents, losing the values saved by the low-priority interrupt.

8. How do you generate an MCLR reset?

Apply a low voltage to the MCLR pin, use a low-true pushbutton switch with a pullup resistor (see diagram in lab manual).

9. Draw the diagram of an open-drain CMOS output.



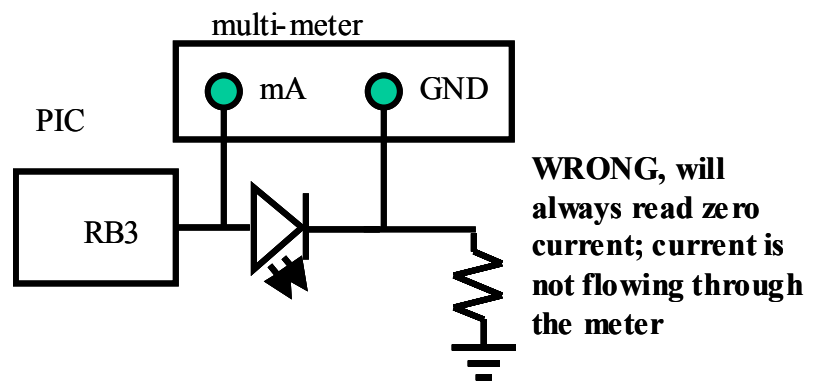
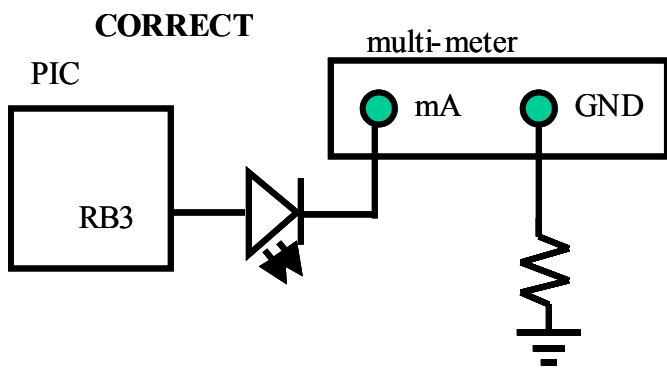
10. Write a single C statement that enables the weak pullups on the PORTB port.

```
RBPU = 0;
```

11. When is the weak pullup on the PORTB port useful?

Eliminates the need for an external pullup on a switch input.

12. Draw a diagram with a multi-meter that measures the current through an LED that is being driven by a PIC output port. The multimeter inputs are labeled as V, mA,  $\Omega$ , and GND. Be sure to indicate the two multimeter inputs that you use.



13. Assume you wanted to use the scope to capture in single-trigger mode a character being received at the RX pin of the PIC (the RX pin is being driven by the MAX202). Would you configure the scope to be falling-edge triggered or rising-edge-triggered? Explain why, if your explanation is wrong you get no credit.

Falling-edge triggered as start bit transition is from high (5 V) to low (0 V).

14. Why is the MAX202 chip needed to implement the serial port function? Why can't I just hook the PIC TX, RX pins up to the TX, RX pins of the DB9 connector?

The MAX202 converts RS232 levels (0 = +3V to +25V, 1 = -3V to -25V) to CMOS logic levels (0 = 0V, 1 = 3V to 5V)

15. What happened to the power consumption of the PIC18 when the HS option was used instead of the HSPLL option? Why did this occur?

The power consumption was reduced in HS mode. In HS mode the clock is running at  $\frac{1}{4}$  the clock frequency of HSPLL mode (this multiplies the crystal frequency by 4). So, lower clock frequency, lower power.

16. What is the basic difference between asynchronous serial IO and synchronous serial IO?

Synchronous serial IO sends the clock with the data, asynchronous does not.

17. Is the RS-232 communication channel full-duplex, half-duplex, or simplex?

Full duplex; the separate TX/RX wires allow communication in both directions simultaneously.

18. Assume that I am using a circular buffer to hold incoming data, but I am getting buffer overrun. What two things can I do to prevent overflow? (and you cannot slow down the incoming data).

- a. Increase buffer size.
- b. Empty the buffer more often.