**App Project 2: Multimeter (Group Project)**

**Due Date: Wed 8 Dec 2021 at 4pm on D2L Dropbox**

## Assignment:

Using the Microcontroller and the driver functions developed so far (ADCs, IO control, Serial Displays), you will design a multi-meter capable of measuring voltage, resistance, frequency, and capacitance and displayingthe measured values on the Computer terminal. The App will use the push buttons (PBs) connectedto the input ports RA2, RA4 and RB4 as shown in the schematic in the lecture slides and below. PB1, PB2 and PB3 represent push buttons connected to ports RA2, RA4 and RB4 respectively. The app should be able to measure resistance of a resistor connected to port pin16/RB13/AN11, and measure any voltage applied to AN5/pin 8/ RA3 as follows. The app should also be able to measure frequency inputs up to a user defined range, and capacitance up to a user defined range using circuit elements designed by yourself. The frequency measurement will require zero external parts, and the capacitance measurement should just use resistors and the capacitor under test. You will need to get capacitors for this project from your lab TAs or by purchasing them.

| **User input(s)** | **Output(s)** |
|---|---|
| if PB1 is pressed | MCU is in voltmeter mode and should measure any voltage applied to AN5/pin 8/ RA3 in volts. This value should be displayed in volts on a single line of the PC terminal as follows:<br><br>**VOLTMETER Voltage =_____V** |
| if PB2 is pressed | MCU is in ohmmeter mode and should measure the resistance of any resistor connected to pin16/RB13/AN11 in ohms. This value should be displayed in ohms on a single line of the PC terminal as follows:<br><br>**OHMMETER Resistance = _____ Ω**<br><br>Please use the circuit connection shown below for pin16/RB13/AN11 to ensure there is no damage to the programmer and/or MCU. $R_{DUT}$ is the resistance to be measured by your MCU/software |
| if PB3 is pressed | MCU is in capacitor mode and should measure the capacitance of any capacitor connected to the PIC24F on a pin of your choosing. This value should be displayed in uF on a single line of the PC terminal asfollows:<br><br>**Capacitance = _____ uF** |
| Persistently | MCU will read frequency incoming on the pin of your choosing.<br>This value should be displayed in Hz on a single line of the PC terminal asfollows:<br>**Frequency = _____ Hz** |

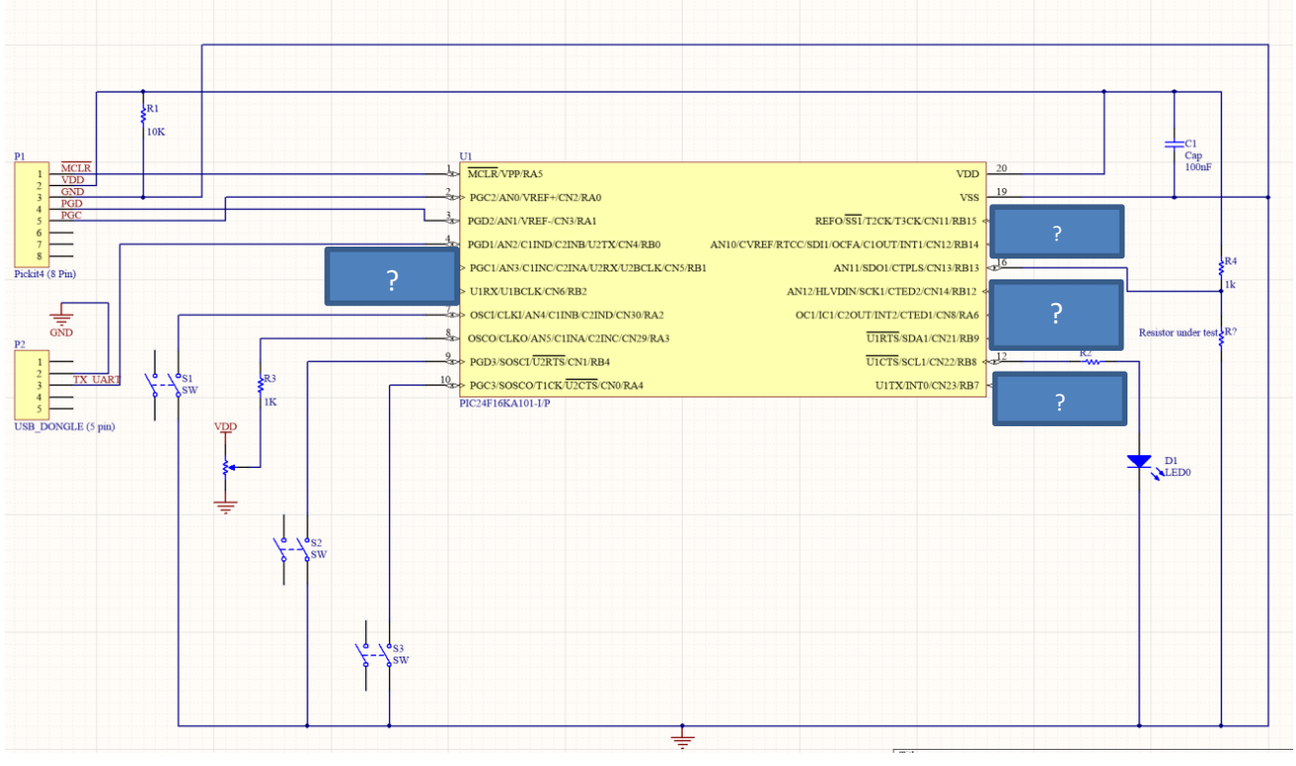
Whenever there are no user inputs selected, and no frequency source connected to your frequency counter, the overall current draw must be less than 1 mA.

## Additional info:

Implement the above controller using the hardware kit and your code, which will be designed using basic ANSI C commands; and driver functions provided. **Use of polling instead of interrupts will lose points.**

**Circuit connections:**

**Use the following the circuit connections to implement the voltmeter and ohmmeter. Ensure the use of the current-limiting 1 k-ohms resistor between the 3V supply (from the PICKIT) and port RB13/16 of the MCU to protect the programmer and MCU. R-DUT is resistance to be measured by your software. Use any area denoted with "?" on the PIC to implement the capacitor value calculator and the frequency counter.**



**Pre-processor Directives:**

Pin 8 is one of those exceptional pins that are multiplexed to the external clock oscillator port in addition to an analog and digital IO. Using the below preprocessor commands will turn off any clock-input related feature on pin 8 thereby avoiding any conflicts on pin 8 while it is being used with the ADC.

#pragma config FCKSM = CSECMD // Clock switching is enabled, clock monitor disabled

#pragma config OSCIOFNC = ON //CLKO output disabled on pin 8, use as IO.

**Function names:** Students can use any convention when naming functions or organizing code. A state diagram is required as part of your submission. Use microcontroller-specific register and bit names wherever applicable in the state diagram.

**Display instructions:** All displays on the PC terminal window should be on a single line. Note that display functions carried out at 32 kHz (300 Baud) can affect timer delays. Your code should account for such delays when producing delays specified in the table above.

**Interrupts:** Interrupt ISR names are provided in the lecture slides. As specified in lecture, IO (CN interrupts) are triggered on rising and falling edges and due to any debounce effects of the push buttons. A debounced switch will result in several hi to lo and lo to hi fluctuations at the Microcontroller input before stabilizing to a steady and fixed voltage when the switch is pressed. Your code should filter out any such effects.

## Deliverables:

This is a group project. Each group should upload the following onto their respective group D2L-Dropbox folder created:

1. **Zipped up file of the MPLAB project**. MPLAB projects can be zipped up by right clicking on the project and selecting package (See screenshot below). The zipped project is saved in the same project folder created by user. Make sure your driver code is commented properly.
2. **A single pdf document** showing the following:
    a. Names and UCIDs of all students in the group at the top of the document
    b. A State diagram showing the working of your code. Use microcontroller-specific register and bit names wherever application in the state diagram.
    c. The mathematical formulas used in determining the voltage, resistance, and capacitance being measured and displayed on the PC terminal.
    d. The maximum and minimum limits of the resistance, voltage, capacitance, and frequency that your app project can measure
    e. List of tasks performed by each group member
3. **Link to your video demo** uploaded on youtube, Vimeo or similar video hosting website along with the zipped up project. Include the link under description while uploading the zipped up project. Dropbox or Google or OneDrive links are allowed as well but ensure that videos are in .mp4 or .mov format. Videos uploaded in any other format will lose points. Video demo should be a single recording and show the following
    a. UCID card of one group member placed in front of the computer with MPLAB and/or hardware running
    b. Demo of the code and hardware operation showing all states. Use the

potentiometer to generate different voltages and resistances on pins 8 and 16 of the MCU in voltmeter and ohmmeter modes. Show your app measuring the voltage and resistance on the PC terminal in real time as the potentiometer knob is being turned. Use the REF out as a frequency source in the demo and show two RODIV settings being correctly counted. If you do not have a multimeter, show the portion of the code where you are putting the processor into an idle state using the debugger.

## Grading rubric: (Total = 25 points)

- Correct setup and use of all modes - 4 point for voltmeter and ohmmeter each, and 4 points each for frequency and capacitance, 3 points for idle current <1 mA = 19 points
- PDF report = 2 points
- Proper video and code upload format including commenting of code = 2 points
- Group participation = 2