

Driver Project: ADC Data Capture using Python programing (Group Project)

Due Date: as per D2L

Assignment:

In this project, you will acquire and plot data from the Microcontroller's ADC on a computer using Python. The ADC will record voltage generated by the potentiometer on IO pin 8 (RA3) of the Microcontroller. Your project should perform the following tasks.

1. Perform Analog to Digital conversion of the analog voltage that the potentiometer induces on pin 8 (RA3) of the Microcontroller
2. Acquire and store the value of ADC's buffer register (ADCBUF) and the resultant voltage on IO pin 8 of the Microcontroller in a data frame on a computer with the help of Python programing over a 10 sec interval.
3. Plot the value of the ADC buffer and ADC's analog voltage in volts vs time over a 10 sec interval on a computer using Python programing. During this 10 sec interval, turn the potentiometer to its lowest and highest resistance value.

Additional info:

- Implement the above controller using the hardware kit and your code, which will be designed using basic ANSI C commands; Microcontroller's ADC, prior driver functions and Python programing.
- Function names: ADC function should be named uint16_t do_ADC(void) and placed in source files ADC.c and ADC.h
- Display instructions: The plots of ADC's buffer and voltage vs time should be plotted on 2 separate graph windows.

Note:

Port RA2 is one of those exceptional ports that is also multiplexed to the input for an external oscillator and an analog input port. To be able to use it as a digital input with a pushbutton, it's multiplexed analog input has to be disabled by including the following line of code in your IOinit() function. However, do ensure that pin 8 (RA3) used in this project is properly configured as an analog input for the data acquisition.

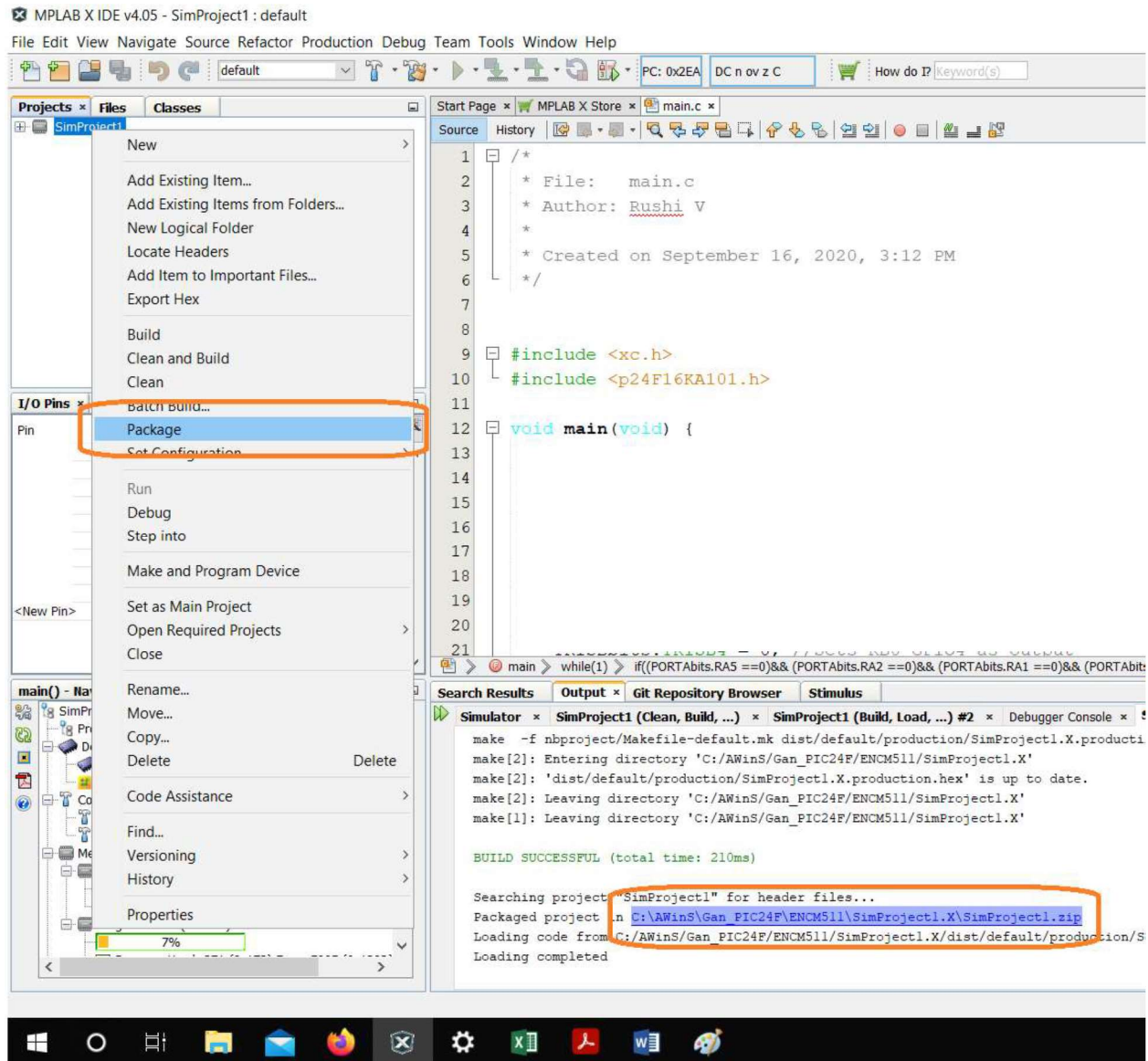
```
AD1PCFG = 0xFFFF; // Turn all analog pins as digital
```

Deliverables:

This is a group project. Each group should upload the following onto their respective group D2LDropbox 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 especially any mathematical computations used.**

2. Properly commented Python code used for Data acquisition
3. A CSV or Excel file containing the data frame of the acquired data. The Data Frame should have properly labeled column names and contain columns for time (sec), ADC Buffer Value and ADC input voltage value.
4. Link to your video demo uploaded on youtube, Vimeo or similar video hosting website along with 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 as follows:
 - a. Single recording no more than 2 mins long
 - b. Show UCID or government issued ID cards of one of the group members placed in front of the computer with MPLAB and/or hardware running
 - c. Demo of the code and hardware operation showing the following:
 - i. The potentiometer being turned from its minimal to its maximal resistance value a few times during the 10 sec measurement interval
 - ii. On the computer, your Python code should generate a data frame containing the time of data capture in seconds, the digital ADC buffer value and ADC's input voltage in volts. You should display the Data frame in Python's terminal window and in its CSV or EXCEL generated using Python commands during the video demo.
 - iii. On the computer, your Python code should generate 2 plots showing the digital ADC buffer value vs. time (sec) and ADC's input voltage (volts) vs time(sec).
 - d. Explanation of the code organization in MPLAB and Python including any special power or time saving features (i.e. interrupts, clock switching, sleep/idle) used, and respective contribution of each group member towards code development and hardware/software testing.



Grading rubric (Total 20 points)

- Correct setup and use of timers, IOs, UART, ADCs, interrupts and clock modules as follows: 16 points
 - Properly displays the graph in Python of the ADC buffer ADCBUF. Graph should show highest and lowest ADCBUF values and ADCBUF values in between (3 points)
 - Properly displays the graph in Python of the ADC's input voltage. Graph should show highest and lowest ADC input voltages and voltages in between (3 points)
 - Properly generates CSV file with column of captured ADCBUF values using Python (3 points)
 - Properly generates same CSV file with column of ADC's input voltage using Python (3 points)
 - Code uses clock switching and Idle() modes to minimize power consumption wherever possible (2 points)
 - Correct setup and use of timers, IOs, UART, ADCs, interrupts and clock modules (2 points)
- Code (including properly uploaded code with comments) = 2 points
- Proper video demo (includes one UCID card displayed, meeting of demo time limit, brief explanation of Hardware and software operation) = 2 points

Driver Project Generic Rubric – Fall 2023

	Fails to meet expectations	Minimally meets expectations	Adequately Meets Expectations	Exceeds Expectations	Score awarded
Peripheral Configuration and Use	None of the peripherals and states working correctly 0 to 2 points	Some of the peripherals and states working correctly 4 points	Most of the peripherals and states working correctly 6 points	All peripherals and states correctly working 8 points	
Program Logic	Does not provide evidence of appropriate program flow 0 to 2 points	Provides evidence of attempting to use C control statements but has some errors or does not cover all scenarios 4 points	Provides evidence of attempting to use C control statements to cover most but not all scenarios or has some errors 6 points	Provides evidence of appropriate C control statements and implements all scenarios correctly 8 points	
Code Quality	No evidence of code commenting or reasonable variable names 0 points	Some evidence of code comments, but infrequent or incomplete 1 point	Evidence that most elements of the code are commented 1.5 points	Evidence that all important elements of the code are commented meaningfully 2 points	
Demonstration and Video Upload	Demo does not appear to work or does not align with project specifications or Video not provided 0 points	Demo does not cover all scenarios or no explanation provided 1 point	Demo shows the design working, covering most states but not all states or proper explanation for all states not provided or Video runs over time limit. 1.5 points	Demo shows the design working, covering all states. Video meets time limit and contains proper explanation of hardware and software operation 2 points	
Total (20):					

Overall

Fails to meet expectations 0 - 8

Minimally meets expectations 9 - 12

Adequately Meets Expectations 13 - 17

Exceeds Expectations 18 - 20