App Project 2: Open-Ended Design (Group Project)

Due Date: As per D2L Dropbox

Assignment:

Using the Microcontroller and the driver functions developed so far, you will propose, design, implement, and demonstrate a system of your choice.

Your project implementation **must** satisfy the following requirements:

- You must display something useful on the UART terminal / Python interface
- You must use power-saving features i.e. clock-switching and Idle/Sleep modes
- You must use interrupts/interrupt service routines
- You must use at least one of the reset-related peripherals (e.g., WDT, BOR)
- **In addition** to the peripherals/drivers needed to implement the above, you must make use of at least **three (3) more** peripherals/drivers covered in the course, of which at least one (1) of them **must** be one of the analog-related modules (e.g., the ADC, HLVD…). Note that using multiple of the same "type" of peripheral does not count (i.e., you cannot claim Timer 1 and Timer 2 as two separate peripherals/drivers).

You must document the specification of your system (e.g., the functionality, user inputs, outputs, expected behaviors). You must also document the justification for why you choose to implement a given functionality (for instance, you might want to use the BOR functionality – why is the BOR useful for your app?). You can be creative in your use of the different modules (e.g., a timer can be used to measure the amount of time elapsed, rather than as a way of implementing a given delay). Alongside your specification, you must create a state diagram of how your system should work.

The first step of your project is to submit a Project Proposal – submit details of your proposed project for feedback **to the D2L dropbox for a project proposal.** This should include:

1. A brief explanation of the overall "aim" of your project (e.g., simulated elevator controller, multimeter…)
2. A list of the peripherals/drivers you plan to implement/use
3. **This is due by Friday November 18th**

Additional info:

- Implement your system using the hardware kit and your code, which will be designed using basic ANSI C commands. **Use of polling instead of interrupts will lose points.** You can use some of the code that we have provided as a starting point.
- As a model for how you should specify your design, you can consider the driver/app project specification that we have provided.

- You can choose to use different pins in your microcontroller/breadboard setup if necessary.
- **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.
- You have considerable flexibility to determine what your system should do. For instance, you might consider designing a system that can measure things (e.g., measuring voltages will also let you determine things like current, resistances, you can also measure time, waveform shapes etc.). You can generate different pulses. You also have an ADALM2000 at your disposal, which can be used to generate variable voltage sources, custom waveforms, and so on. Note that you can provide a broader context for your project (e.g., you might say that your app is a controller for an elevator, and that you are "simulating" choosing a floor with the push buttons).

Deliverables:

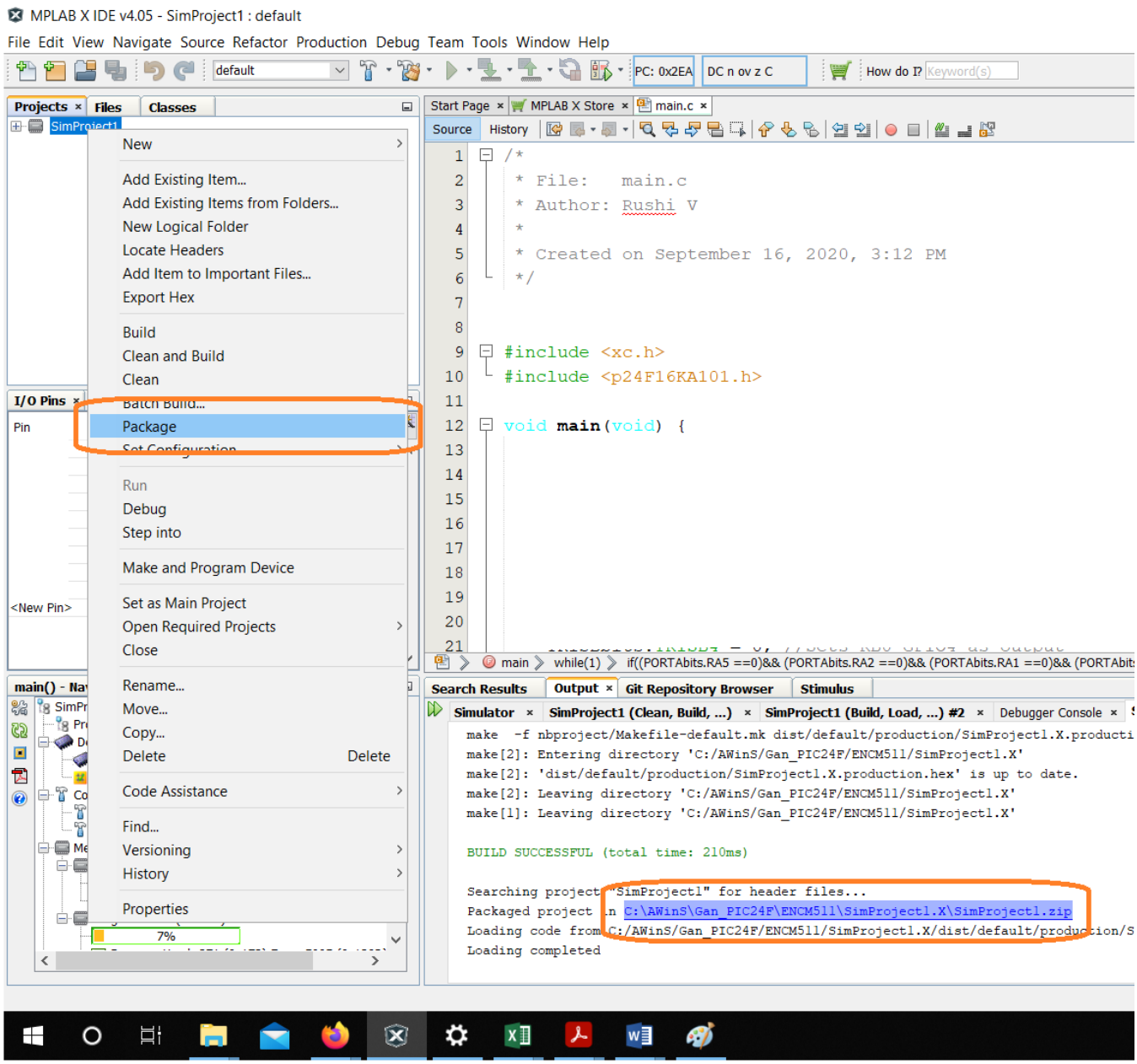
This is a group project. Each group should upload the following onto their respective group D2L-Dropbox folder created:

4. **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.
5. **A single pdf document** showing the following:
    a. Names and UCIDs of all students in the group at the top of the document
    b. The specification document of your project, outlining, but not limited to:
        i. A description of the app project's functionality
        ii. A detailed specification of the input/output behavior
        iii. A description of the drivers/modules that you chose to use, and why you are using them
        iv. An explanation of where you used power saving functionality
    c. A State diagram showing the working of your code. Use microcontroller-specific register and bit names wherever application in the state diagram.
    d. List of tasks performed by each group member

6. **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 **all group members** placed in front of the computer with MPLAB and/or hardware running
    b. An overview of what your project does, what modules you picked to implement, and why
    c. Demo of the code and hardware operation showing all states. You **must** demonstrate how all the different modules in your design have been used (for instance, you might want to demonstrate how your design safely restarts after a brownout, so you can "emulate" this by varying the supply voltage as part of your video demo).

Grading rubric: (Total = 25 points)

- Correct setup and use of drivers/modules – 4 points for implementation and justification per driver (the fixed set of prescribed drivers is considered jointly as 1 driver). Must have at least one analog-related module otherwise 0 overall = up to 16 points
- Power-efficient and seamless operation i.e., optimal use of clock switching, interrupts and idle state, including justification = 3 points
- Proper specification of the design (specification document and state diagrams) = 2 points
- Proper video and code upload format including commenting of all driver lines of code = 2 points
- Group member participation/justification in report = 2 points
- Note that the requirements must be satisfied to score points in this project.


MPLAB X IDE v4.05 - SimProject1 : default
File Edit View Navigate Source Refactor Production Debug Team Tools Window Help
Projects
Files
Classes
SimProject1
New
Add Existing Item...
Add Existing Items from Folders...
New Logical Folder
Locate Headers
Add Item to Important Files...
Export Hex
Build
Clean and Build
Clean
Package
Run
Debug
Step into
Make and Program Device
Set as Main Project
Open Required Projects
Close
Rename...
Move...
Copy...
Delete
Code Assistance
Find...
Versioning
History
Properties
I/O Pins
Pin
<New Pin>
Start Page
MPLAB X Store
main.c
Source
History
/*
* File: main.c
* Author: Rushi V
*
* Created on September 16, 2020, 3:12 PM
*/
#include <xc.h>
#include <p24F16KA101.h>
void main(void) {
Search Results
Output
Git Repository Browser
Stimulus
Simulator
SimProject1 (Clean, Build, ...)
SimProject1 (Build, Load, ...) #2
Debugger Console
make -f nbproject/Makefile-default.mk dist/default/production/SimProject1.X.producti
make[2]: Entering directory 'C:/AWinS/Gan_PIC24F/ENCM511/SimProject1.X'
make[2]: 'dist/default/production/SimProject1.X.production.hex' is up to date.
make[2]: Leaving directory 'C:/AWinS/Gan_PIC24F/ENCM511/SimProject1.X'
make[1]: Leaving directory 'C:/AWinS/Gan_PIC24F/ENCM511/SimProject1.X'
BUILD SUCCESSFUL (total time: 210ms)
Searching project "SimProject1" for header files...
Packaged project in C:\AWinS\Gan_PIC24F\ENCM511\SimProject1.X\SimProject1.zip
Loading code from C:/AWinS/Gan_PIC24F/ENCM511/SimProject1.X/dist/default/production/S
Loading completed