**Driver Project 3: IOs, Timers, Interrupts and Display drivers (Group Project)**

**Due Date: 31 Oct 2021 at 11:59pm**

## Assignment:

Using the Microcontroller and the driver topics on IOs, Timers, interrupts and display drivers covered so far, you will design a simple IO controller to test out IO and Timer Interrupts with display drivers. Design a state machine to turn on, turn off and blink a LED connected to port RB8 based on the push buttons (PBs) connected to the input ports RA2, RA4 and RB4 as shown in the schematic in the lecture slides. PB1, PB2 and PB3 represent push buttons connected to ports RA2, RA4 and RB4 respectively. The state machine should operate as follows:

| **User input(s)** | **Output(s)** |
|---|---|
| While PB1 is pressed | LED blinks at approx. 0.5 sec intervals (0.5 sec on and 0.5 sec off)<br>***"PB1 is pressed"*** is displayed on PC terminal |
| While PB2 is pressed | LED blinks at approx. 2 sec intervals (2 sec on and 2 sec off)<br>***"PB2 is pressed"*** is displayed on PC terminal |
| While PB3 is pressed | LED blinks at approx. 3 sec intervals (3 sec on and 3 sec off)<br>***"PB3 is pressed"*** is displayed on PC terminal |
| While 2 PBs are pressed together | LED stays on without blinking<br>***"PB_ and PB_ are pressed" is displayed on PC terminal. "*** The blanks __ should display the corresponding PB numbers pressed |
| While all 3 PBs are pressed together | LED stays on without blinking<br>***"All PBs pressed"*** is displayed on PC terminal. |
| No PBs pressed | LED stays off<br>***"Nothing pressed"*** is displayed on the PC terminal |

## Additional info:

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

**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.

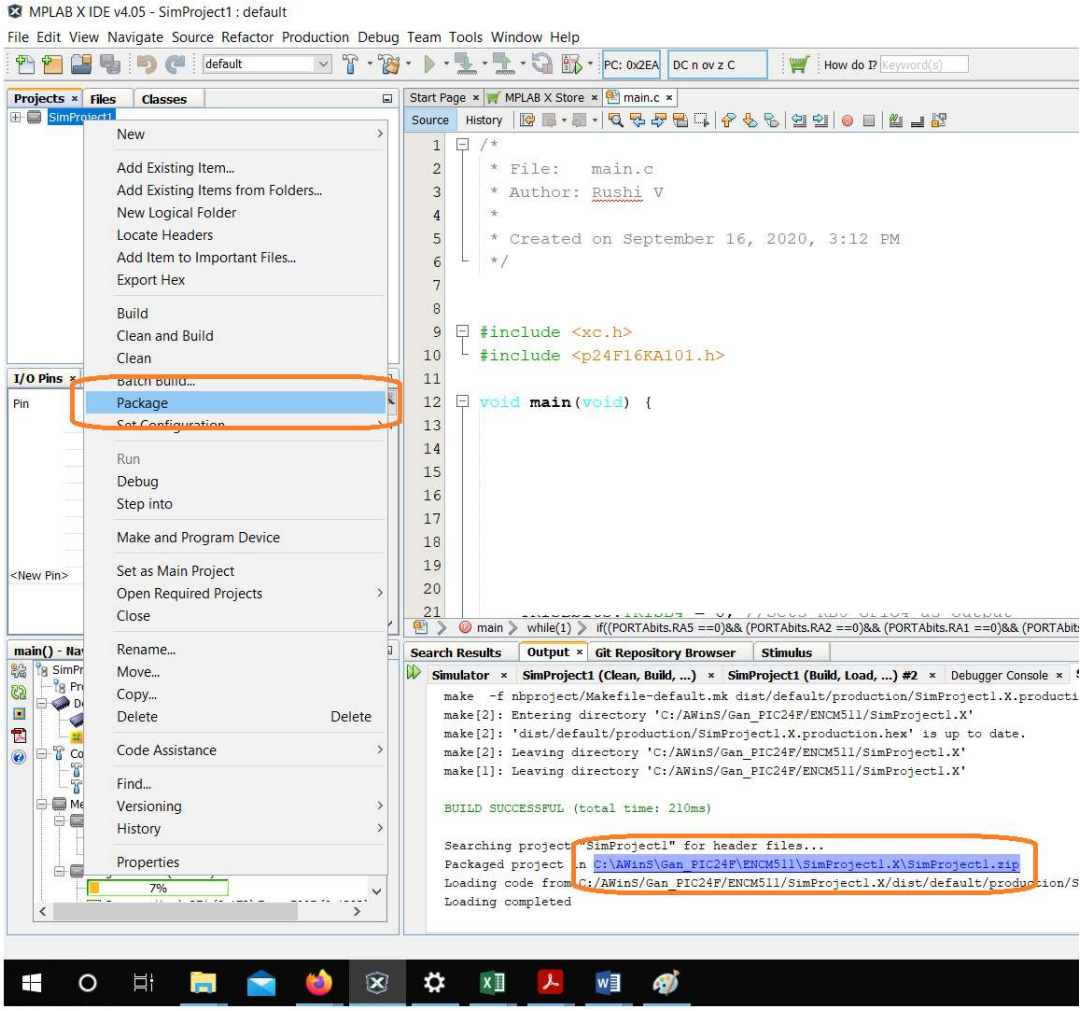
## Deliverables:

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. 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.

## Grading rubric: (Total = 10 points)

- Correct setup and use of timer/IO interrupts, display functions and clock modules – 1 point for each of the states above = 6 points
- Power-efficient operation i.e. optimal use of clock switching, interrupts and idle state = 1 point
- Proper video and code upload format including commenting of all driver lines of code = 2 points
- Group participation = 1


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
Batch Build...
Package
Set Configuration
Run
Debug
Step into
Make and Program Device
Set as Main Project
Open Required Projects
Close
Rename...
Move...
Copy...
Delete
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