Driver Project 3: 16 bit Timers with Interrupts (Group Project) – Fall 2023

Due Date: As per D2L

Assignment:

Using the PIC 24F16KA101 and Timer interrupts, you will design a simple IO controller to test out Timers and Interrupts. 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 slide 'HW and IO Control.pdf'. 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. 1 sec intervals (1 sec on and 1 sec off)
While PB2 is pressed	LED blinks at approx. 2 sec intervals (2 sec on and 2 sec off)
While PB3 is pressed	LED blinks at approx. 3 sec intervals (3 sec on and 3 sec off)
While PB1 and PB2 are pressed	LED blinks for 1 msec or smallest delay that your delay function can produce. Show and explain the value of Clock speed, Prescaler and PRx value used
No PBs pressed	LED stays off

Additional info:

Implement the above controller using the hardware kit and your code, which will be designed using basic ANSI C commands and Timer interrupts. *Use of polling and built in __delay () functions instead of timers and interrupts will lose points.*

IOinit() – initializes the IO ports and placed in IOs.c

IOcheck() – implements the IO checks and LED blinking functions and placed in IOs.c

delay_ms(time_ms) – implements the delay functions used to time the LED blinks. time_ms is the user specified time delay in milliseconds. Your function is only expected to handle delays of whole numbers (e.g. 1 ms, 5 ms, 2504 ms) and not floating point numbers (e.g. 1.2 ms, 5.5 ms, 2504.8ms) . Place timer related functions in source file TimeDelay.c.

main() – Used to call IOinit() and IOcheck() and placed in main.c

Hint on generating delay cycles: Use timers, interrupts and if applicable clock switching for the delay function.

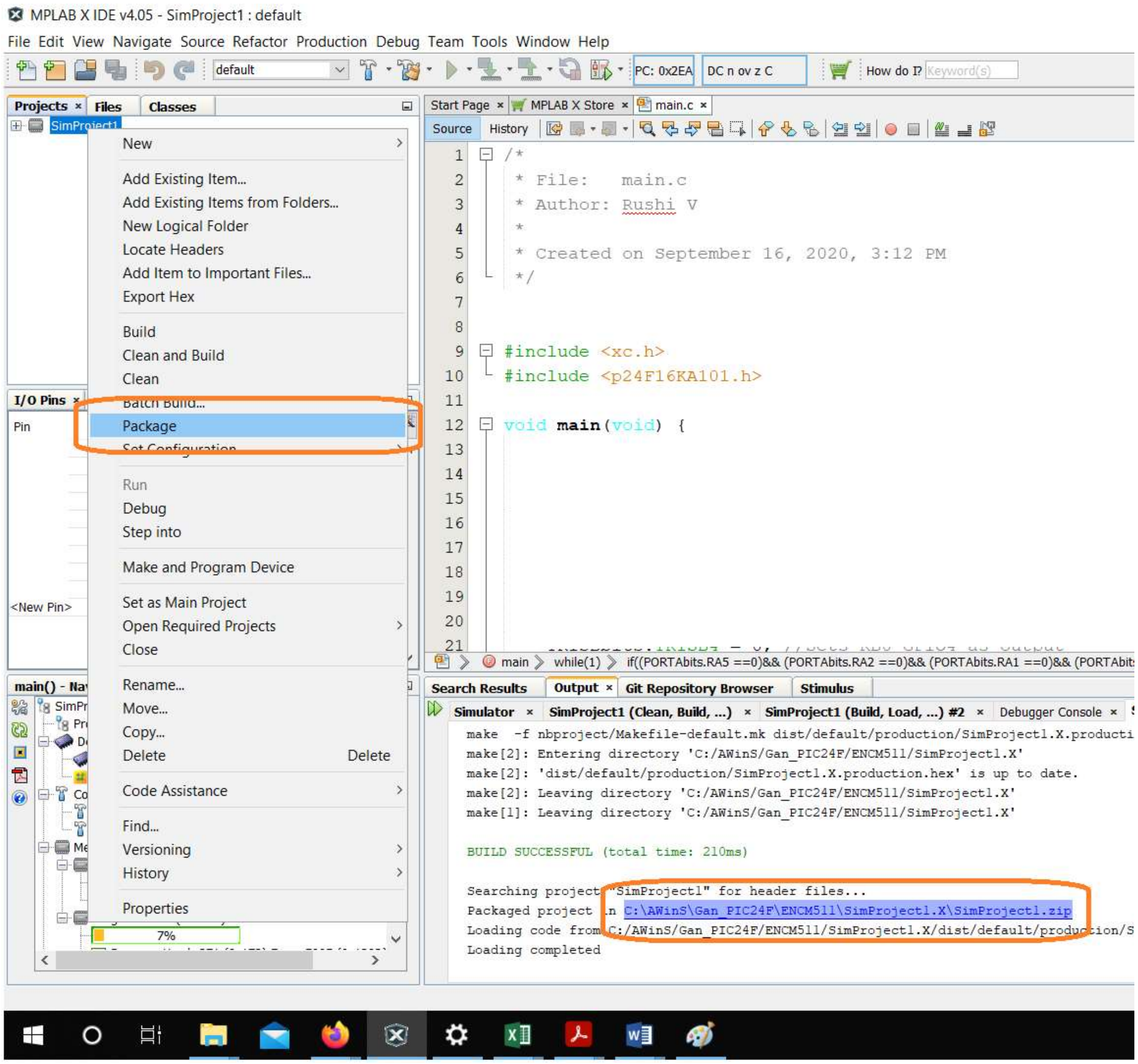
Note: Port RA2 is one of those exceptional ports that is also multiplexed to the input for an external oscillator and an analog input port. To be able to use it as a digital input with a pushbutton, it's multiplexed analog input has to be disabled by including the following line of code in your IOinit() function. We will revisit this multiplexing when we look at ADC converters in a couple of weeks.

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

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 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. **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 1 group member placed in front of the computer with MPLAB and/or hardware running**
    c. **Demo of the code and hardware operation showing the following:**
        i. **Each of the PBs pressed individually - showing LEDs blinking**
        ii. **2 or more PBs pressed simultaneously – showing ADALM screen shot of LED or RB8 output**
        iii. **No PBs pressed**
    d. **Explanation of the code organization in MPLAB 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.**



```c
/*
 * File:   main.c
 * Author: Rushi V
 *
 * Created on September 16, 2020, 3:12 PM
 */


#include <xc.h>
#include <p24F16KA101.h>

void main(void) {
```

```
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
```

Grading rubric
-Correct Peripheral setup, Program logic and working of all PB IO states = 16 points (4 points per state)
-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):	

<u>Overall</u>

Fails to meet expectations 0 - 8

Minimally meets expectations 9 - 12

Adequately Meets Expectations 13 - 17

Exceeds Expectations 18 - 20