Middle East Technical University
Department of Computer Engineering

CENG 336

Spring 2006

Term Project - Phase1

Rev 1.0

Basic Up-Down Button Counter with PICos18

Assigned: May 13 2007, Sunday

Due: May 19 2007, Saturday, 23:55

1 Introduction

This part of the term project aims you to get familiar with basic features of PICos18 real time kernel. You will use different tasks for different purposes each of which is executed in a concurrent fashion thanks to the kernel. You will manage a resource (PORTD) to access 7-segment displays and LCD. You also use interrupt features of $PIC18F4520$ to control the buttons.

2 Your Mission

You design a basic $0-99$ up-down "button counter" (decimal) and display the count value in both 7-segment displays and LCD. Since the count value has two digits, you only use two of 7-segment displays as shown in Figure 1. On the LCD, the value will be shown at the last line of the LCD beginning from first character.

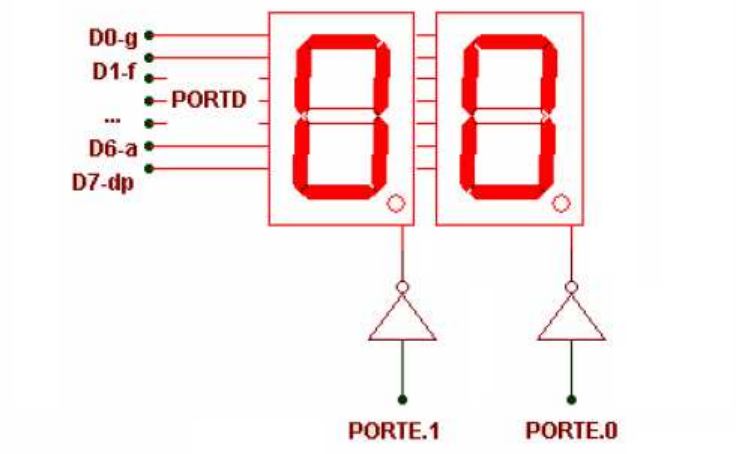

Figure 1: Initial values of displays.

Initially the value is zero. It increases when the user presses $SW04$ button and decreases when $SW08$ is pressed. It is up to your design to continue to increase or to decrease when the corresponding button is hold pressed.

If the count value is 99, next increment will make it 0. Similarly, if the count value is 0, next decrement will make it 99.

Additionally, $SW00$ is used to clear the count value. Whenever it is pressed, the count value will be zero.

3 Hints

- In order to control the buttons you may use *RB Port Change* interrupt or *external* interrupts of $RB0$, $RB1$ and $RB2$ pins.
- If you want to enable PORTB pull-up resistors, be aware of that the kernel code loads $0x80$ to $INTCON2$ by default and disables the pull-ups. This is done in `kernel.asm` file at line 270 and 271. You may edit the code and compile the kernel using `libpicos18.bat` file or write your code considering that your $INTCON2$ setting made in the `init` function of `main.c` file will be overwritten after calling `StartOS` function (The same is applied for other bits of $INTCON2$).
- You are free to use any driver available for $PICos18$ or $MCC18$ library functions.
- For the ones who want to use $LCD\ HD4478$ driver of $PICos18$, consider that the driver is for 2×8 LCDs and that the RW pin of the LCD on PIC Development Tool is connected to GND. Therefore, you need to make some modifications.
- Resource management properties of $PICos18$ can help you while controlling $PORTD$ usage.
- To adjust the configuration settings you may put the below code segment prior to any `#define` or `include` directives in your main file.

```
#pragma config OSC = HS, FCMEN = OFF, IESO = OFF, PWRT = OFF, BOREN = OFF, WDT
= OFF, MCLRE = ON, LPT1OSC = OFF, PBADEN = OFF, LVP = OFF, XINST = OFF, DEBUG
= OFF
```

- We recommend to use 20MHz oscillator.

4 Evaluation

If your code is not compiled or you do not use interrupt features of the PIC while controlling the buttons, you get zero from this part of the term project and will not be able to object your score.

You will **certainly** use PICos18 and MCC18 compiler.

Total of the term project is worth 100 points. The details will be announced later.

5 Hand In Instructions

- You will submit a single file, `phase1.tar.gz`, including all of `.c` and `.h` files in your project folder.
- Make sure that you have included your names and group ID in a comment at the top of `main.c` file.
- You will submit your file(s) using `http://cow.ceng.metu.edu.tr`.
- **No late submission is allowed.**
- For any clarification please follow the newsgroup.