MICROPROCESSORS A (17.383)

Fall 2010

Lecture Outline

Class # 08

November 02, 2010

Dohn Bowden

Today's Lecture

- Syllabus review
- Programming/Software
  - Useful information
- Lab
  - Discussion on your progress/techniques
  - Continue/complete Lab #3 (Seven Segment LCDs)
  - Start Lab #4
- Homework

Course Admin

Administrative

- Admin for tonight ...
  - Course Project
    - Requirements will be passed out tonight
  - Syllabus Highlights
    - Modified Syllabus ... provided time for Labs tonight
    - Lab report for Lab # 3 is due on November 9th
    - For planning purposes ...
      - Exam #2 is November 30th ... 5 weeks from today

Syllabus Review

Week	Date	Topics	Lab	Lab Report Due
1	09/07/10	Intro, Course & Lab Overview, Microcontroller Basics	1	
2	09/14/10	PIC16F684 Overview and General Input/Output	1 con't	
3	09/21/10	Switches	2	
4	09/28/10	Seven Segment LEDs	2 con't	1
5	10/05/10	**Examination 1**		
X	10/12/10	**No Class – Monday Schedule**		
6	10/19/10	Analog to Digital Conversion	3	2
7	10/26/10	Analog to Digital Conversion con't	3 con't	
8	11/02/10	Lab Work (Finish Lab #3 and start Lab #4)	3, 4	
9	11/09/10	LCD Interface and Assembly Language	4 con't	3
10	11/16/10	Comparators, Timers, Pulse Width Modulation (PWM)	5	
11	11/23/10	Mixed C & Assembly Programming/Course Project	Project	4
12	11/30/10	**Examination 2**		
13	12/07/10	Course Project	Project	5
14	12/14/10	Final Exam/Course Project Brief and Demonstration	Demo	

Microcontroller Hardware and / or Interfaces

Hardware / Interface

- No hardware discussion

Programming / Software

Programming/Software

- Programming/Software
  - Useful information/techniques

Useful Techniques ...

Programming Techniques …

- How do I make the results of the ADC into one variable?
- How can I display data on the 7 segment LED display?

- *Both of these questions will be answered …*

Creating One Variable From Two Registers ...

Variable From Multiple Registers

- PIC16F684 has two ADC associated registers containing the last conversion data ...
    - ADRESH
    - ADRESL
- How would we make one variable from the two registers?

“C” to Combine Two Registers

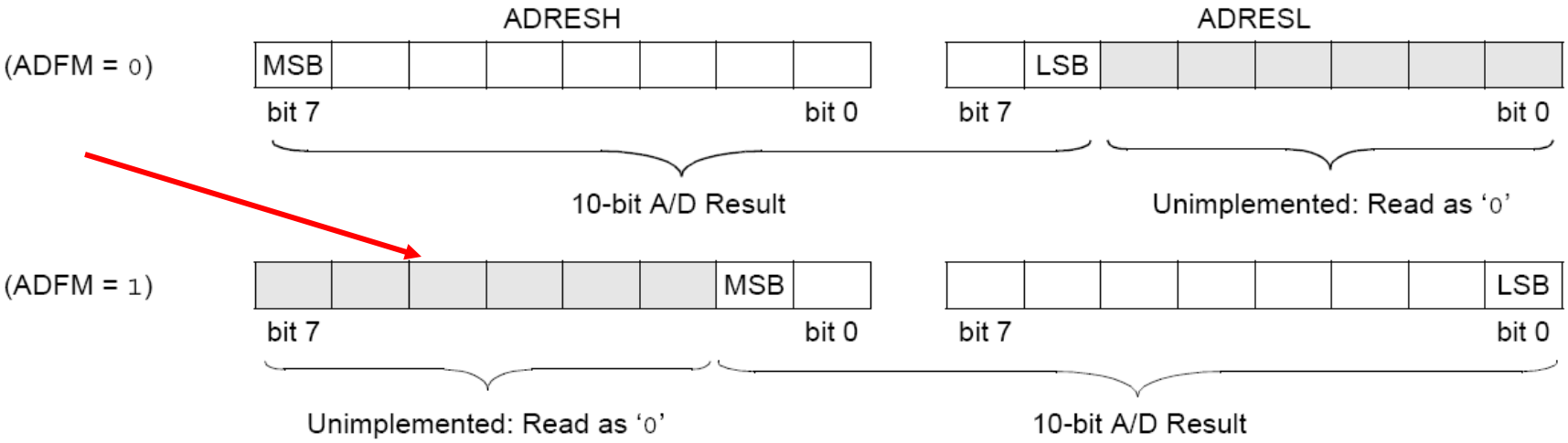
```
int Total_Data;

ADCON0 = 0b10000001;   //   Bit 7    - Right Justified Sample
                       //   Bit 6    - Use VDD
                       //   Bit 5    - Not Used
                       //   Bit 4:2  - Channel 0 (AN0)
                       //   Bit 1    - Do not Start the conversion
                       //   Bit 0    - Turn on ADC



Total_Data=(ADRESH<<8) + ADRESL;
```

A Closer Look at ... "`int Total_Data`" ?

- Data Type "int" means ...
  - 16 bits
  - Signed integer
  - Representing integral values ranging from ...

    -32,768 ... to ... +32,767
- So ... int Total_Data would look like this in binary ...
  - 0b0000000000000000 (16 bits)

Recall … Right Justified Sample – (ADFM=1)
ADRESH and ADRESL



How would we make one variable from the two registers?

- Total_Data=(ADRESH<<8) + ADRESL;
- What is it doing?
  1. First ... set Total_Data to equal ADRESH
  2. Then ... shift Total_Data left eight bits
  3. Next ... add ADRESL to Total_Data (the result of the shift)
  4. Resulting in one variable representing the entire ADC results

Example …

- For Left justified ADC Results (maximum possible result) …
  - If … ADRESH = 0b00000011
  - And … ADRESL = 0b11111111

```
Total_Data = (ADRESH << 8) + ADRESL

Total_Data = 0b0000000000000011 << 8 =

Total_Data = 0b0000001100000000 + 0b11111111 =

Total_Data = 0b0000001111111111 = 1023
```

Obtaining separate digits for 7 - Segment Displays ...

The Problem …

- We have a variable that we want to display on multiple 7 segment displays.
  - How do we break up the variable to display the ...
    - » One’s
    - » Tenth’s
    - » Hundredth’s
    - » Thousandth’s
    - » Ten thousandth’s
    - » One hundred thousandth’s
    - » etc?

“C” to Break up a Variable into its Components …

```
int data;

data = 641077;

hundred_thousand = data/100000;
data = data % 100000;
ten_thousand = data/10000;
data = data % 10000;
thousand = data / 1000;
data = data % 1000;
hundred = data / 100;
data = data % 100;
tenth = data / 10;
data = data % 10;
ones= data;
```

A Closer Look at what the Code is doing …

- Our variable that we want to display is *641077*
- First ... divide 641077 by 100000
  - » Equals 6
- Why 6 and not 6.41077?
  - – Because this is integer division and not floating point
    - » Hence hundred_thousand = 6

Next ...

```
data = data % 100000;
```

- The *%* is the *modulus operator* ...
  - The *modulus operator %* is to give the remainder of the first number divided by the second
- So ...

```
hundred_thousand = data/100000;
data = data % 100000;
```

- We have the remainder of data equal to 641077 divided by 100000 which is 41077

To get the ten_thousandth's digit …

- Our variable is now *41077*
- We now divide 41077 by 10000
  - » Equals 4, therefore ten_thousand = 4
- data = data % 10000;
  - » Equals 1077

To get the thousandth's digit …

- Our variable is now *1077*
- We now divide 1077 by 1000
  - » Equals 1, therefore thousand = 1
- data = data % 1000;
  - » Equals 77

The remaining digits ...

- Are found the same way ...

- Now we can display each digit!

If you need ASCII

- To display ASCII data ...
  - We obtain each digit ... then ...
    - Apply an offset of 0x30 to each digit of the input data

Lab

Peer Code Review ...

Peer Code Review

- For Lab #3 ...
  - How did you approach counting from 0x00 to 0xff?
  - How did you approach integrating the switches to ...
    - Count by one?
    - Reset?
- How are you making out?
  - Problems?

Labs Tonight ...

Lab

- Continue/finish Lab #3
- Start Lab #4

Next Class …

Next Class Topics

- The class of 11/09/10 ...
    - LCD Interface
    - Assembly Language

HomeWork

Homework

1. Complete and submit Lab Report #3 ...
    - Due NLT November 09, 2010
2. Work on Lab #4 ...
    - Report is due NLT November 23, 2010

Time To ...

Start the Lab ...

References

1. PIC16F684 Data Sheet 41202F