Assignment 3A (50 marks) – Lab Week Nine

Due: End of <u>your</u> week Ten's lab period: Week of 9 – 13 Nov 2015

Late submissions will not be accepted and will receive a mark of zero (0).

This Lab has a Pre-Lab Component

I encourage you to start (or complete) this lab by Lab Week Nine and come to the lab prepared to demo no later than the <u>start</u> of Lab Week Ten. In that manner, you will not be "rushed" trying to get things done and competing for a Hardware Board "at the last minute" during Lab Week Ten!

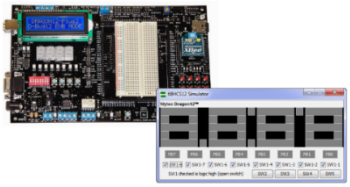
More Assembly Programming – Using the Dragon 12 PLUS Trainer, Completing Code and Tracing a Program

PURPOSE OF LAB:

*The purpose of this lab is to gain experience in Assembly Language using **AsmIDE** using the **Dragon12 & Student Mode Simulator** and the **Dragon 12 PLUS Trainer** hardware board by creating software that will display values on Light Emitting Diodes (LEDs.*

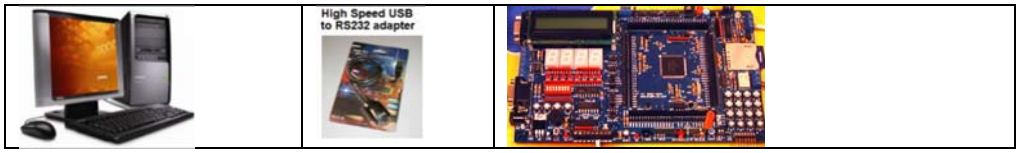
Additionally, we will exercise our knowledge of Flowcharts and Pointers in Assembly Language.

Displaying Values on the Dragon12 Plus LEDs and the Simulator



PRELAB PROCEDURE – Preparing Your Computer for use with the Dragon 12 PLUS Trainer Hardware Board

---



*In order to communicate with the Dragon12-Plus board, a USB to RS232 driver must be installed and configured **PRIOR TO YOUR LAB PERIOD**. The following instructions are provided to accomplish this.*

Resources

Prolific USB to Serial Bridge Chip Family Windows Driver and User Manual

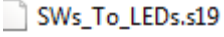
a. *Download the Prolific Driver Package from Blackboard and decompress the file.*
b. *Double click the driver install program, and then follow the prompts to continue with the installation.*
c. *Click Finish once the driver has been installed.*
d. ***Reboot your computer***

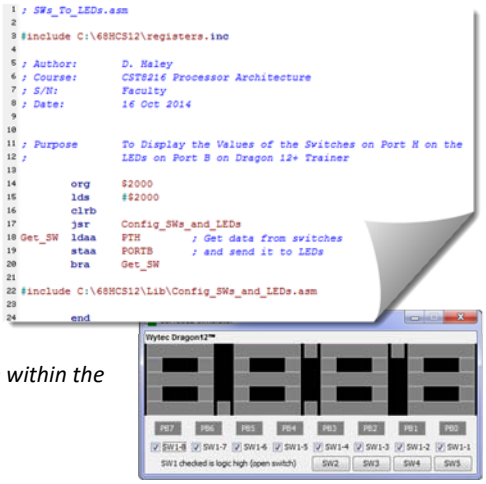
Important Step!

Now you are ready to connect the hardware board in the lab and complete the remainder of the installation.

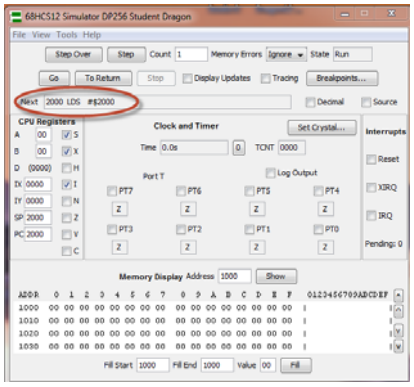
Assembly Language Implementation of Displaying Switch Values on LEDs

1. *Type in the following Assembly Language code listing using ASMIDE.*
2. *Assemble the program and start the simulator.*

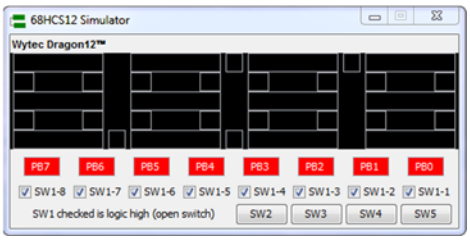
3. *If so prompted, click on the "Don't show this again unless necessary check box" then click on the "Dismiss" button.*
4. *Use File → Load to locate*  *and load the file into the simulator.*
5. *To view the LEDs and Switches in the simulator, click on View → Parallel Ports, which will bring up the display on the right.*
6. *If you have correctly loaded the file, then **2000 LDS #$2000,** which is the first line of code within the program, will be displayed in the simulator as illustrated on the next page.*



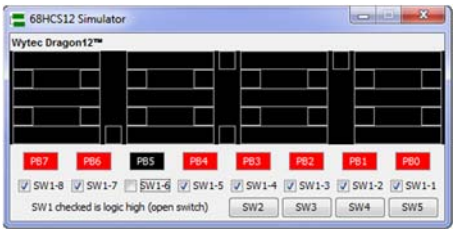
```
; SWs_To_LEDs.asm

#include C:\68HCS12\registers.inc

; Author:       D. Haley
; Course:       CST8216 Processor Architecture
; S/N:          Faculty
; Date:         16 Oct 2014


; Purpose       To Display the Values of the Switches on Port H on the
;               LEDs on Port B on Dragon 12+ Trainer

        org     $2000
        lds     #$2000
        clrb
        jsr     Config_SWs_and_LEDs
Get_SW  ldaa    PTH         ; Get data from switches
        staa    PORTB       ; and send it to LEDs
        bra     Get_SW

#include C:\68HCS12\Lib\Config_SWs_and_LEDs.asm

        end
```

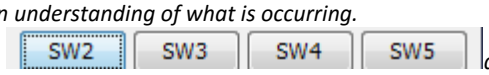


7. *Clicking on "Go" in the simululator should realize the following being displayed.*



8. *The display shown in the previous paragraph indicates that all of the Switches are in the "ON" position. Technically, this means that each of the Port H (PTH) switch values are a logic level "1" or "+5V", turning on the common Cathode LEDs.*
9. *Now, click on any of the SW1-1 to SW1-8 check boxes, which simulate the physical DIP switches on Port H of the hardware board. Observe the resulting changes to the LEDs on Port B. You should observe a direct correlation with the Switches. That is, if SW1-6 is "unchecked", then PB5 is turned off as per the following display, noting that the Switches are numbered 1 – 8 and the LEDs are numbered 0 – 7. Then, if SW1-6 is "checked", PB5 will turn on again. Try this with several of the switches and look at the source code so that you can gain an understanding of what is occurring.*



10. *Clicking on* 

and holding the switch down will also cause the associated LED on PB3, PB2, PB1 and PB0 to momentarily turn off until you release the switch – test this as well to ensure that you obtain the expected results.

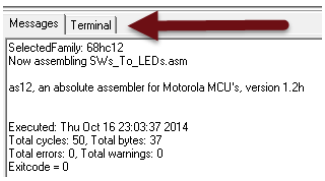
IN LAB CONFIGURATION PROCEDURE – SHOULD BE COMPLETED AT THE BEGINNING OF THE LAB

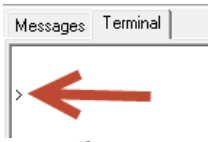
Checking Your Software on the Wytec HCS12 Dragon12-Plus board

In order to communicate with the Dragon12-Plus board, you previously installed a USB to RS232 driver. Now you must complete the installation process by connecting the hardware board using the following instructions:

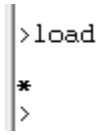
a. *Plug the powered-on Dragon12 Plus Board into a USB port on your computer. You should then notice a Driver Installation dialogue, noting that you may not see it if under Windows 10.*
b. *Once the installation is complete, the hardware will be ready to use as indicated by the following "example."*
c. *Run AsmIDE and Configure it as follows:*

   i. *Load and assemble your program that you have previously tested in the simulator*
   ii. *Select View → Options*
   iii. *Select COM Port → COM 3 (***use Window's Device Manager → Ports to determine the correct Comm Port # if you don't know which one it is***) (see next page)*
   iv. *Click on "Set COM Options" A new dialogue box opens.*
   v. *Click on "OK" in that dialogue box.*
   vi. *Click on "Enable the Terminal Window"*
   vii. *Click on "OK"*
   viii. *You should now see the Terminal Tab in the lower IDE window*

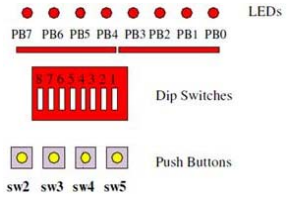




ix. *Click on the "Terminal" Tab*
x. *Click any place in the Terminal window to get focus for that window.*
xi. *Press "Enter" on your keyboard – you should observe the display to the right.*

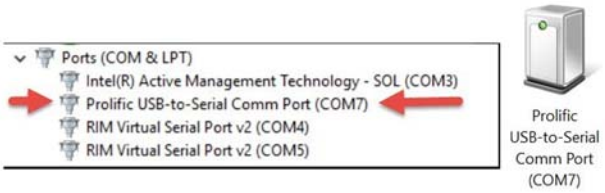


xii. *Type "load" (without the quotes) and press "Enter", then press the lightning bolt (or use Build →Assemble) and observe the display to the right:*
xiii. *In the Terminal window, type "g 2000" (without the quotes) and press "Enter." Observe the LEDs on the Dragon12 Plus board. Note that "g 2000" executes the program code starting at address $2000 and that there is a <space> between 'g' and '2000'*



xiv. *Experiment with the Switches on the board to confirm that the LEDs turn on/off as per what you observed in the simulator.*
xv. *Press the Blue Reset Button located on the lower-middle portion of the board to end the program on the board.*

<u>Note</u>: If you missed the Comm Port display or you are using Windows 10, then you can always determine what Comm Port via the Window's Device Manager or Printers and Devices as per the following "example."



Assessment Portion of Assignment 3A (50 Marks) – Lab Exercise Week Nine

Task One – "Flash_Even_Odd.asm" (10 marks)

To complete this task, modify the Flash_PB0.asm file from Lab Week Eight so that instead of just flashing one LED, your solution has the following ***Program Behaviour****:*

a) *Sends the appropriate values to* ***portb*** *to turn off all LEDS, then delays for 250 ms;*
b) *Sends the appropriate values to* ***portb*** *to turn on b6, b4, b2 and b0 (Even numbered) LEDS, then delays for 250 ms;*
c) *Sends the appropriate values to* ***portb*** *to turn on b7, b5, b3 and b1 (Odd numbered) LEDs, then delays for 250 ms;*
d) *Infinitely loops through a), b) and c) above*

*View the video "****Flash_Even_Odd.asm.mp4****" on Blackboard to see the required behaviour of the software.*

First, observe the correct functionality in the simulator*, and then demonstrate your software on the Wytec HCS12 Dragon12-Plus board (see hand-in sheet), as per the previous instructions in this assignment – Checking Your Software on the Wytec HCS12 Dragon12-Plus board.*

Modifications Required to Lab Week Seven Assembly Language Program

Modify your existing Lab Week Eight code (Flash_PB0.asm) as follows:

a. *change the source file name to* ***Flash_Even_Odd.asm***
b. *include your student information, the date and the purpose of the program in the header portion of your code*
c. *noting that all equate (****equ****) statements appear before the first org statement, write the following equ statements with their given labels, values and comments:*

```
; Equate Statements
STACK       equ     $2000
PB_EVEN     equ     %01010101       ; b6, b4, b2, b0 on – remainder of bits off
PB_ODD      equ     %10101010       ; b7, b5, b3, b1 on – remainder of bits off
PB_OFF      equ     %00000000       ; all LEDs off
DTIME       equ     250             ; 250 ms delay value
```

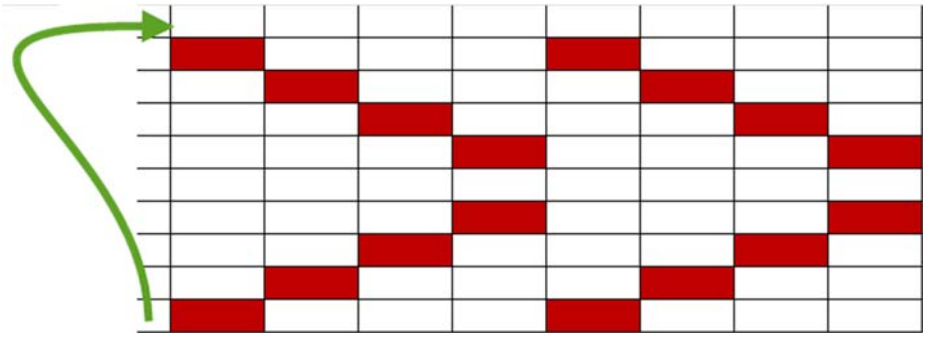
d. *modify the original code where appropriate to incorporate the* ***Program Behaviour*** *listed above*
e. *modify the original code where appropriate to incorporate these* ***CONSTANTS*** *and modify the existing comments appropriately to explain their use*
f. *remove any old code/comments that are no longer required*

Note: *Since this program interacts with hardware, and uses a delay routine, you must ensure that you retain all three of the* ***include*** *statements from Lab Week Eight's code listing in this program.*

Once you write your software, test it using the Simulator – Dragon12 & Student Mode and ensure that it works in the simulator BEFORE demonstrating it on the Hardware Board (see Hand-In Sheet)

Task Two – "Walk A Bit" (15 marks)

*In this task you will write a complete the Assembly Language program "**Walk_A_Bit.asm**" that implements the following bit pattern (using iteration) that continually repeats itself on the LEDs using a **180** ms delay between patterns. Note that an empty entry in the table indicates the LED is "OFF", while a filled in square indicates it is "ON."*



***Hint**: Use an Array, pointers and iteration to "walk through" and display the values on the LEDs.*

Note: A "complete" program includes header information, Labels, CONSTANTS, and comments in the program that clearly explain what the code is doing (versus explaining the instruction set).

*View the video "**Walk_A_Bit.mp4**" on Blackboard to see the required behaviour of the software.*

***First, observe the correct functionality in the simulator**, and then demonstrate your software on the Wytec HCS12 Dragon12-Plus board (see hand-in sheet), as per the previous instructions in this assignment – Checking Your Software on the Wytec HCS12 Dragon12-Plus board.*

Task Three – BCD Counter Flowchart (15 marks)

Using Microsoft Visio, create a flowchart that implements a counter that counts in BCD from $00 (BCD) - $99 (BCD), then continually starts over at $00 (BCD) - $99 (BCD). Each count will be displayed on PORT B LEDs as a binary number.

i.e. BCD $09 is displayed as: 0000 1001, BCD $10 is displayed as: 0001 0000, BCD $44 is displayed as: 0100 0100, where a 0 signifies that the LED is OFF and a 1 signifies that the LED is ON. There are no "special display" features that must be employed here. For example, if Accumulator A contains the value $55, then sending that value to PORT B will result in 0101 0101 being displayed on the LEDs.

Ensure that your flowchart initializes the count to $00 (BCD) (all LEDs will be OFF for this value) and that it checks to see if you have reached the upper count of $99 (BCD) (the LED pattern will be 1001 1001 for this value). Once this value has been reached, your flowchart should indicate that the count starts over at $00 (BCD).

*You must also ensure that you do not have **any** invalid counts (e.g. in BCD the sequence is 0, 1, 2, … 9, 10, 11, … **not** 0, 1, 2, … 9, $A, $B, $C, $D, $E, $F, $10, $11…). Note that you **cannot** READ the values on the LEDs. Don't forget that your flowchart should incorporate a "delay" between each of the displayed values; otherwise, the count will occur too fast. Recall that the "delay" routine is in a subprogram and as such must be correctly annotated as such on the Flowchart.*

*Additionally, **do not** include any registers or Assembly Language instructions in your flowchart, as flowcharts should be able to "solve the problem" in any programming language.*

When creating a new Visio document, chose "Blank Drawing" over "Basic Diagram", which employs blue-coloured diagrams. After choosing "Blank Drawing", select More Shapes → Flowchart → Basic Flowchart Shapes. They should all be in Black and White versus Blue and White, which is unreadable when printed.

A READABLE hardcopy of your Visio flowchart must be submitted with this lab assignment. Ensure your name and student number are on your submission.

Task Four – Program Tracing (10 Marks)

Complete the entries of Trace1.asm on the Hand-In sheet.

Assignment 3A (50 marks) – Lab Week Nine (Page 1 of 2)

Please staple the pages together.

Due: End of <u>your</u> week Ten's lab period: Week of 9 – 13 Nov 2015
Late submissions will not be accepted and will receive a mark of zero (0).

I encourage you to start (or complete) this lab by Lab Week Nine and come to the lab prepared to demo no later than the <u>start</u> of Lab Week Ten. In that manner, you will not be "rushed" trying to get things done and competing for a Hardware Board "at the last minute" during Lab Week Ten!

Name: _____

Student Number: _____

Circle Your Lab Period/Time

Tue: 10 – 12 Wed: 1 – 3 Wed 3 – 5 Fri 2 – 4

For all Tasks with demos, there is only ONE Demo permitted on the hardware boards. If you have problems with your solutions in the simulator, discuss them with me BEFORE you demo on the hardware board**. This means being prepared to demo your solutions during the early stages of the lab period; not "at the last minute!"**

Task One – "Flash_Even_Odd.asm" (10 marks)

A. Demo of your solution on the Hardware Board (Must work 100% for credit) Professors Initials: _____ /3

B. Post-Lab Code Inspection Check-Off List

Item	*Marks*
File name: Flash_Even_Odd.asm	*/1*
Your student information, the date and the purpose of the program are in the header portion of your code	*/1*
`; Equate Statements` `STACK  equ  $2000` `PB_EVEN  equ  %01010101  ; b6, b4, b2, b0 on – remainder of bits off` `PB_ODD  equ  %10101010  ; b7, b5, b3, b1 on – remainder of bits off` `PB_OFF  equ  %00000000  ; all LEDs off` `DTIME  equ  250  ; 250 ms delay value`	*/2*
Comments in the program that clearly explain what the code is doing (versus explaining the instruction set)	*/2*
Remove any old code/comments that are no longer required	*/1*

Post-Lab Code Inspection – hand in a hard copy of your code – it will be evaluated Post-Lab */7*

Task Two – "Walk A Bit" (10 marks)

Demo of your solution on the Hardware Board (Must work 100% for credit) Professor's Initials: _____ /7

Post-Lab Code Inspection – hand in a hard copy of your code – it will be evaluated Post-Lab */8*

Task Three – BCD Counter Flowchart (15 marks)

A READABLE hardcopy of your Visio flowchart must be submitted with this lab assignment. <u>Ensure your name and student number are on your submission</u>.

*Post-Lab Code Evaluation of Visio Flowchart (**must be completed in Visio for credit**)* */15*

Assignment 3A (50 marks) – Lab Week Nine (Page 2 of 2)

Please staple the pages together.

Due: End of your week Ten's lab period: Week of 9 – 13 Nov 2015
Late submissions will not be accepted and will receive a mark of zero (0).

Name: _____

Student Number: _____

Circle Your Lab Period/Time

Tue: 10 – 12 Wed: 1 – 3 Wed 3 – 5 Fri 2 – 4

Task Four – Programming Tracing (10 Marks)

*(Given the following program listing, trace the results (hexadecimal value of registers being asked) of each program step **after** the line of code has been executed, as per the example answer for the Stack Pointer (SP).*

Hint: You may use the Instruction Set handout on Blackboard, which will be included with the Term Test).

```
1  ; Trace1.asm
2
3          org     $1000
4  Data1   db       $22, $40, $48
5  Data2   db       $88, $13, $46, $65, $53
6
7          org     $2000
8          lds     #$2000          ; SP =?  $2000 <= example answer
9
10         ldy     #Data1          ; Y = ?  _____ (1 mark)
11
12         ldaa    0,y             ; A = ?  _____ (1 mark)
13
14                                 ; Y = ?  _____ (1 mark)
15
16         ldab    1,y+            ; B = ?  _____ (1 mark)
17
18                                 ; Y = ?  _____ (1 mark)
19
20         ldx     #Data2          ; X = ?  _____ (1 mark)
21
22         ldaa    4,x             ; A = ?  _____ (1 mark)
23
24         ldd     2,x+            ; A = ?  _____ (1 mark)
25
26                                 ; B = ?  _____ (1 mark)
27
28                                 ; X = ?  _____ (1 mark)
29
30         swi
31         end
```

Post-Lab Code Evaluation */10*