Add Instructions and Logic

Lab Goal

Understand the effect of addition instructions on the status bits and results.

Instructions/Questions:

1 Open the Add.doc file to see the add project code.

2 Indicate what the values of the registers **will be** when the code is run:

		Destination (dst)				Source (src)				N	Z	C
		R5	R4	R7	R6	R9	R8	R11	R10	N	Z	C
After line 30 has executed		0x1000	0xFFFC	0x1000	0xFFFC	0x0000	0x0001	0x0000	0x0001			
Loop 1	line 35		0xFFFD				0x0001			1	0	0
	line 36	0x1000				0x0000				0	0	0
	line 39				0xFFFD				0x0001	1	0	0
	line 40			0x1000				0X0000		0	0	0
Loop 2	line 35		0xFFFE				0x0001			1	0	0
	line 36	0x1000				0x0000				0	0	0
	line 39				0xFFFE				0x0001	1	0	0
	line 40			0x1000				0X0000		0	0	0
Loop 3	line 35		0xFFFF				0x0001			1	0	0
	line 36	0x1000				0x0000				0	0	0
	line 39				0xFFFF				0x0001	1	0	1
	line 40			0x1000				0X0000		0	0	0
Loop 4	line 35		0x0000				0x0001			0	1	1
	line 36	0x1000				0x0000				0	0	0
	line 39				0x0000				0x0001	0	1	1
	line 40			0x1000				0X0000		0	0	0
Loop 5	line 35		0x0001				0x0001			0	0	0
	line 36	0x1000				0x0000				0	0	0
	line 39				0x0001				0x0001	0	0	0

	line 40		0x1000			0X0000		0	0	0

Table convert Hexadecimal (base 16) to Binary (base 2)

										10	11	12	13	14	15
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Syntax for ADD instruction in Assembly language
(Double Operand Instructions)

ADD[.B]   src , dst   src + dst => dst

Rule Addition Binary Numbers Using Place Value

if sum of two digits :

= 0 , write a 0

= 1 , write a 1

= 2 , write a 0 , carry 1   (result is 10)

= 3 , write a 1 , carry 1   (result is 11)

(if carried 1)

Conversion Binary to Decimal to Hexadecimal

a 8-bit	Eight	Four	Two	One	
	2^3	2^2	2^1	2^0	
Binary number	1	1	0	1	
	(8 x 1) =	(4 x 1) =	(2 x 0) =	(1 x 1)=	
Decimal number	8 +	4 +	0 +	1	$= 13_{(10)}$
Hexadecimal value	$1101_{(2)}$ = $13_{(10)}$ = $D_{(16)}$				

&☆&

Loop 1   **Hex**   **Binary (base 2)**

line 35

    `ADD.W    R8,R4`   R8 =   0x0001   =   0000 0000 0000 0001

                             R4 =   0xFFFC   =   1111 1111 1111 1100

              =>   R4 =   **0xFFFD**   <=>   1111 1111 1111 1101 =   13 dicemal ≈   **D**   (1101)

line 36

    `ADD.W    R9,R5`   R9 =   0x0000   =   0000 0000 0000 0000

                             R5 =   0x1000   =   0001 0000 0000 0000

              =>   R5 =   **0x1000**   <=>   0001 0000 0000 0000

line 39

    `ADD.W    R10,R6`   R10 =   0x0001   =   0000 0000 0000 0001

                             R6 =   0xFFFC   =   1111 1111 1111 1100

              =>   R6 =   **0xFFFD**   <=>   1111 1111 1111 1101 =   13 dicemal ≈   **D**   (1101)

line 40

    `ADD.W    R11,R7`   R11 =   0x0000   =   0000 0000 0000 0000

                             R7 =   0x1000   =   0001 0000 0000 0000

              =>   R7 =   **0x1000**   <=>   0001 0000 0000 0000

❧☆❧

Loop 2   **Hex**   **Binary (base 2)**

line 35

    `ADD.W`   `R8,R4`   R8 =   0x0001   =   0000 0000 0000 0001

                                 R4 =   0xFFFD   =   1111 1111 1111 1101

                =>   **R4** **=**   **0xFFFE**   <=>   1111 1111 1111 1110 = 14 dicemal ≈   **E**   (1110)

line 36

    `ADD.W`   `R9,R5`   R9 =   0x0000   =   0000 0000 0000 0000

                                 R5 =   0x1000   =   0001 0000 0000 0000

                =>   R5 =   **0x1000**   <=>   0001 0000 0000 0000

line 39

    `ADD.W`   `R10,R6`   R10 =   0x0001   =   0000 0000 0000 0001

                                 R6 =   **0xFFFD**   =   1111 1111 1111 1101

                =>   R6 =   **0xFFFE**   <=>   1111 1111 1111 1110 = 14 dicemal ≈   **E**   (1110)

line 40

    `ADD.W`   `R11,R7`   R11 =   0x0000   =   0000 0000 0000 0000

                                 R7 =   0x1000   =   0001 0000 0000 0000

                =>   R7 =   **0x1000**   <=>   **0001 0000 0000 0000**

ೋ☆ೋ

Loop 3   **Hex**   **Binary (base 2)**

   line 35

      `ADD.W    R8,R4`   R8 =   0x0001   =   0000 0000 0000 0001

                               R4 =   0xFFFE   =   1111 1111 1111 1110

                =>   **R4 =   0xFFFF**   <=>   1111 1111 1111 1111 =   15 dicemal ≈   **F**   (1111)

   line 36

      `ADD.W    R9,R5`   R9 =   0x0000   =   0000 0000 0000 0000

                               R5 =   0x1000   =   0001 0000 0000 0000

                =>   R5 =   **0x1000**   <=>   0001 0000 0000 0000

   line 39

      `ADD.W    R10,R6`   R10 =   0x0001   =   0000 0000 0000 0001

                               R6 =   0xFFFE   =   1111 1111 1111 1110

                =>   **R6 =   0xFFFF**   <=>   1111 1111 1111 1111 =   15 dicemal ≈   **F**   (1111)

   line 40

      `ADD.W    R11,R7`   R11 =   0x0000   =   0000 0000 0000 0000

		Hex		Binary (base 2)
R7 =	0x1000	=	0001 0000 0000 0000
=> R7 =	0x1000	<=>	0001 0000 0000 0000

ری ☆ ری

Loop 4

                                         Hex   **Binary (base 2)**

line 35

```
ADD.W    R8,R4
```

R8 =	0x0001	=	0000 0000 0000 0001
R4 =	**0xFFFF**	**=**	**1111 1111 1111 1111**
=> R4 =	0x0000	<=>	**1** 0000 0000 0000 0000 (C = 1 , Z = 1)
			(Carry 1)

line 36

```
ADD.W    R9,R5
```

R9 =	0x0000	=	0000 0000 0000 0000
R5 =	**0x1000**	**=**	**0001 0000 0000 0000**
=> R5 =	0x1000	<=>	0001 0000 0000 0000

line 39

```
ADD.W    R10,R6
```

R10 =	0x0001	=	0000 0000 0000 0001
R6 =	**0xFFFF**	**=**	**1111 1111 1111 1111**
=> R6 =	0x0000	<=>	**1** 0000 0000 0000 0000 (C = 1 , Z = 1)
			(Carry 1)

line 40

```
ADD.W    R11,R7
```
R11 =   0x0000   =   0000 0000 0000 0000

R7 =   0x1000   =   0001 0000 0000 0000

=>   R7   =   0x1000   <=>   0001 0000 0000 0000

&☆&

Loop 5		**Hex**		**Binary (base 2)**

line 35

```
ADD.W    R8,R4
```
R8   =   0x0001   =   0000 0000 0000 0001

R4   =   0x0000   =   0000 0000 0000 0000

=>   **R4   =   0x0001   <=>   0000 0000 0000 0001**

line 36

```
ADD.W    R9,R5
```
R9   =   0x0000   =   0000 0000 0000 0000

R5 =   0x1000   =   0001 0000 0000 0000

=>   R5   =   0x1000   <=>   0001 0000 0000 0000

line 39

```
ADD.W    R10,R6
```
R10 =   0x0001   =   0000 0000 0000 0001

R6   =   0xFFFF   =   1111 1111 1111 1111

=>   **R6   =   0x0001   <=>   0000 0000 0000 0001**

line 40

```
ADD.W    R11,R7        R11  =     0x0000      =      0000 0000 0000 0000
                       R7  =      0x1000      =      0001 0000 0000 0000
                  =>   R7   =     0x1000     <=>     0001 0000 0000 0000
```

ॐ📖ॐ

Add Instructions and Logic

Lab Goal

Understand the effect of addition instructions on the status bits and results.

3 Open Add.eww workspace

4 Show me your work on instruction 3 then debug the "add" project. Select "add" tab in Workspace window.

5 Indicate what the values of the registers **are** when the code is stepped through:

After completing Lab_6a_Add. Download Attached File: Lab File - Add.zip (33.612 KB) in the Lab 6b (Add) folder.

and Unzip the entire "Add.zip" file. Save it in the LAB_CET 453 folder.

Then open the   ☑️ 🌀 add   file IAR IDE Workbench workspace (1 KB)

Click   **Project =>** Download and Debug (Ctrl + D)   for run

           Project => Clean

           Use key F11 for run Loop

           The values of the registers are the same as those implemented in Lab_6a_add. When the code is executed step by step.

And the Professor confirmed that it is correct.

❧☆❧

The values of the registers in the ams.s43 file when the code is executed step by step.

❧☆❧

The values of the registers in the ams.s43 file when the code is executed step by step. are the same as the values implemented in Lab_6a_add.

		Destination (dst)				Source (src)				N	Z	C
		R5	R4	R7	R6	R9	R8	R11	R10	N	Z	C
After line 30 has executed		0x1000	0xFFFC	0x1000	0xFFFC	0x0000	0x0001	0x0000	0x0001			
Loop 1	line 35		0xFFFD				0x0001			1	0	0
	line 36	0x1000				0x0000				0	0	0
	line 39				0xFFFD				0x0001	1	0	0
	line 40			0x1000				0X0000		0	0	0
Loop 2	line 35		0xFFFE				0x0001			1	0	0
	line 36	0x1000				0x0000				0	0	0
	line 39				0xFFFE				0x0001	1	0	0
	line 40			0x1000				0X0000		0	0	0
Loop 3	line 35		0xFFFF				0x0001			1	0	0
	line 36	0x1000				0x0000				0	0	0
	line 39				0xFFFF				0x0001	1	0	1
	line 40			0x1000				0X0000		0	0	0
Loop 4	line 35		0x0000				0x0001			0	1	1
	line 36	0x1000				0x0000				0	0	0
	line 39				0x0000				0x0001	0	1	1
	line 40			0x1000				0X0000		0	0	0
Loop 5	line 35		0x0001				0x0001			0	0	0
	line 36	0x1000				0x0000				0	0	0
	line 39				0x0001				0x0001	0	0	0

| | line 40 | | | 0x1000 | | | | 0X0000 | | 0 | 0 | 0 |

❧ 📖 ❦

		R5	R4	R7	R6	R9	R8	R11
After line 30 has e		0x1000	0xFFFC	0x1000	0xFFFC	0x0000	0x0001	0x0000
Loop 1	line 35		0xFFFD				0x0001	
	line 36	0x1000				0x0000		
	line 39				0xFFFD			
	line 40			0x1000				0X0000
Loop 2	line 35		0xFFFE				0x0001	
	line 36	0x1000				0x0000		
	line 39				0xFFFE			
	line 40			0x1000				0X0000
Loop 3	line 35		0xFFFF				0x0001	
	line 36	0x1000				0x0000		
	line 39				0xFFFF			
	line 40			0x1000				0X0000
Loop 4	line 35		0x0000				0x0001	
	line 36	0x1000				0x0000		
	line 39				0x0000			
	line 40			0x1000				0X0000
Loop 5	line 35		0x0001				0x0001	
	line 36	0x1000				0x0000		
	line 39				0x0000			
	line 40			0x1000				0X0000

Study the assembly Code in the attached word document and answer question 3 i

3 Open Add.eww workspace

4 Show me your work on instruction 3 then debug the "add" project. Select "add" tab in Workspace

5 Indicate what the values of the registers **are** when the code is stepped through:

		R5	R4	R7	R6	R9	R8	R11
After line 30 has e		0x1000	0xFFFC	0x1000	0xFFFC	0x0000	0x0001	0x0000

Loop 1	line 35		0xFFFD				0x0001	
	line 36	0x1000				0x0000		
	line 39				0xFFFD			
	line 40			0x1000				0X0000
Loop 2	line 35		0xFFFE				0x0001	
	line 36	0x1000				0x0000		
	line 39				0xFFFE			
	line 40			0x1000				0X0000
Loop 3	line 35		0xFFFF				0x0001	
	line 36	0x1000				0x0000		
	line 39				0xFFFF			
	line 40			0x1000				0X0000
Loop 4	line 35		0x0000				0x0001	
	line 36	0x1000				0x0000		
	line 39				0x0000			
	line 40			0x1000				0X0000
Loop 5	line 35		0x0001				0x0001	
	line 36	0x1000				0x0000		
	line 39				0x0000			
	line 40			0x1000				0X0000

Study the assembly Code in the attached word document and answer question 3 i

R10	N	Z	C
0x0001	0	0	0
	0	0	1
	0	0	0
0x0001	0	0	1
	0	0	0
	0	0	1
	0	0	0
0x0001	0	0	1
	0	0	0
	0	0	1
	0	0	0
0x0001	0	0	1
	0	0	0
	0	0	1
	0	0	0
0x0001	1	0	1
	0	0	0
	0	0	1
	0	0	0
0x0001	0	0	1
	0	0	0

in the the Excel file.

window.

R10	N	Z	C
0x0001	0	0	0

	0	0	1
	0	0	0
0x0001	0	0	1
	0	0	0
	0	0	1
	0	0	0
0x0001	0	0	1
	0	0	0
	0	0	1
	0	0	0
0x0001	0	0	1
	0	0	0
	0	0	1
	0	0	0
0x0001	1	0	1
	0	0	0
	0	0	1
	0	0	0
0x0001	0	0	1
	0	0	0

in the the Excel file.