Martin Jaime-Viveros

CPE301 – SPRING 2016

Design Assignment 0

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

The student understands that all required components should be submitted in complete for grading of this assignment.

NO	SUBMISSION ITEM	COMPLETED (Y/N)	MARKS (/MAX)
0.	COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS		
1.	INITIAL CODE OF TASK 1/A		
2.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C		
4.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D		
5.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 5/E		
6.	SCHEMATICS		
7.	SCREENSHOTS OF EACH TASK OUTPUT		
8.	SCREENSHOT OF EACH DEMO		
9.	VIDEO LINKS OF EACH DEMO		
10.	GOOGLECODE LINK OF THE DA		

0.	COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS		

- Atmel Studio 7

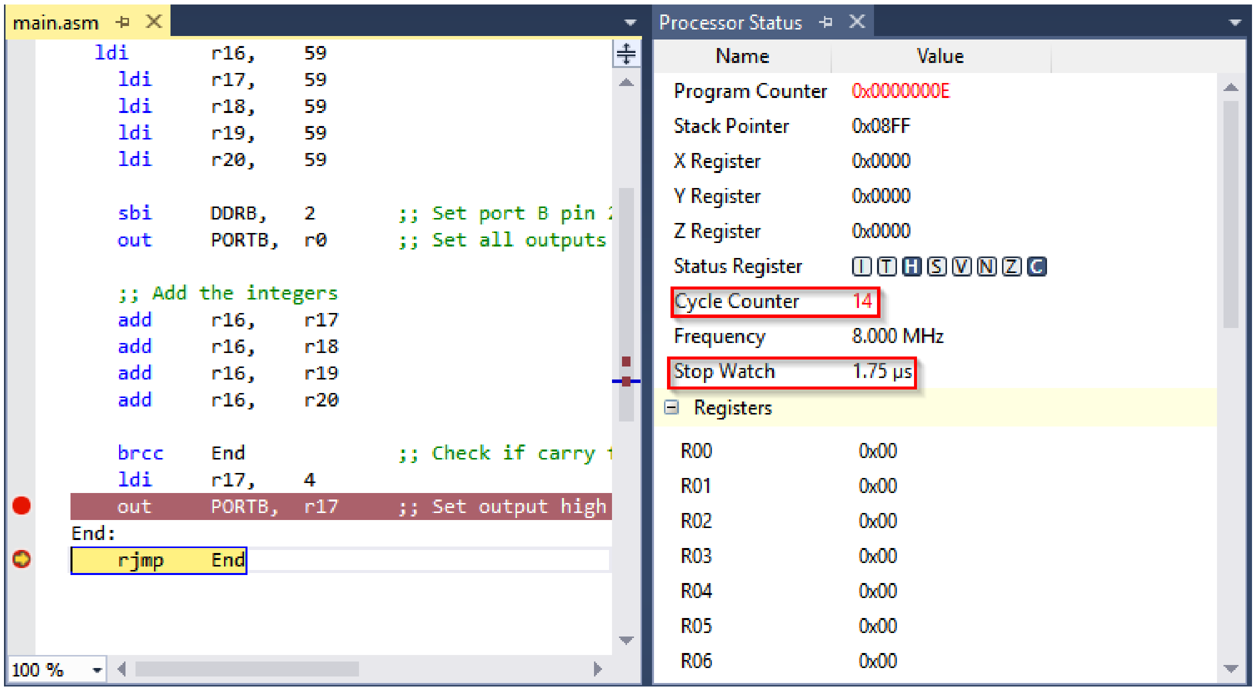
1.	INITIAL CODE OF TASK A		

Task A: Write an assembly code to add five random numbers >30 and <60. If the sum produces an overflow set PORTB.2 pin = HIGH else PORTB.2 pin = LOW.

```
; DA0T1.asm
;
; Created: 2/11/2016 19:06:27
; Author : Martin Jaime-Viveros
;
.cseg

start:
    ;; Load arbitrary immediates into registers 16:20
    ldi     r16,    59
    ldi     r17,    59
    ldi     r18,    59
    ldi     r19,    59
    ldi     r20,    59

    sbi     DDRB,   2       ;; Set port B pin 2 as output
    out     PORTB,  r0      ;; Set all outputs to 0

    ;; Add the integers
    add     r16,    r17
    add     r16,    r18
    add     r16,    r19
    add     r16,    r20

    brcc    End             ;; Check if carry flag set
    ldi     r17,    4
    out     PORTB,  r17     ;; Set output high at pin2 on port B
End:
    rjmp    End
```

2.	TASK B		

Determine the execution time/#cycles of your algorithm using the simulation, set CLOCK speed = 8 MHz.



```
main.asm
  ldi      r16,    59
    ldi      r17,    59
    ldi      r18,    59
    ldi      r19,    59
    ldi      r20,    59

    sbi      DDRB,   2        ;; Set port B pin 2
    out      PORTB,  r0       ;; Set all outputs

    ;; Add the integers
    add      r16,    r17
    add      r16,    r18
    add      r16,    r19
    add      r16,    r20

    brcc     End              ;; Check if carry 
    ldi      r17,    4
    out      PORTB,  r17      ;; Set output high
End:
    rjmp     End
```

Processor Status

Name	Value
Program Counter	0x0000000E
Stack Pointer	0x08FF
X Register	0x0000
Y Register	0x0000
Z Register	0x0000
Status Register	I T H S V N Z C
Cycle Counter	14
Frequency	8.000 MHz
Stop Watch	1.75 µs
Registers	
R00	0x00
R01	0x00
R02	0x00
R03	0x00
R04	0x00
R05	0x00
R06	0x00

The worst case of the algorithm is when the sum is produces a carry as in the case of 59 + 59 + 59 + 59 + 59 = 295 = 0x127 since that would cause the code to execute all lines.

With execution time of 1.75 µs with 14 cycles, the average cycle lasted 125 ns. Which agrees with a clock period of 125 ns.

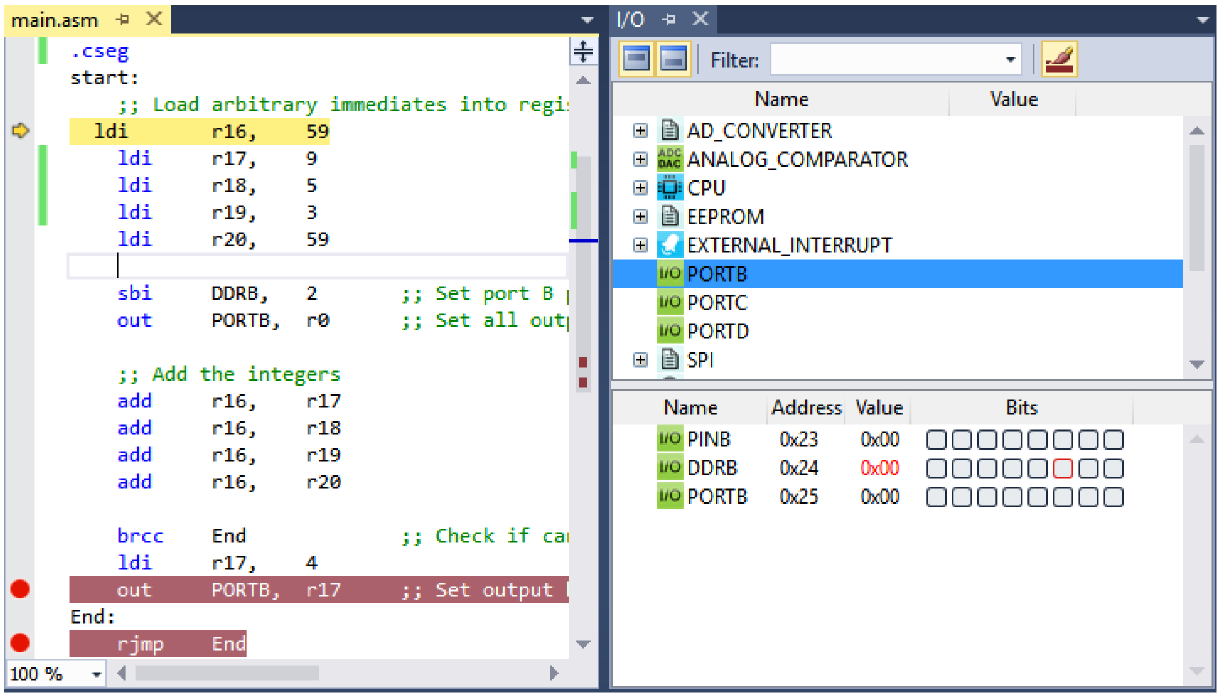
6.	SCHEMATICS		

The project was run on the Atmel Studio 7 simulator.

7.	SCREENSHOTS OF EACH TASK OUTPUT		

TASK 1a: Write an assembly code to add five random numbers >30 and <60. If the sum produces an overflow set PORTB.2 pin = HIGH else PORTB.2 pin = LOW. screenshots of the AVRStudio6 during debugging at the beginning and end of Task a.

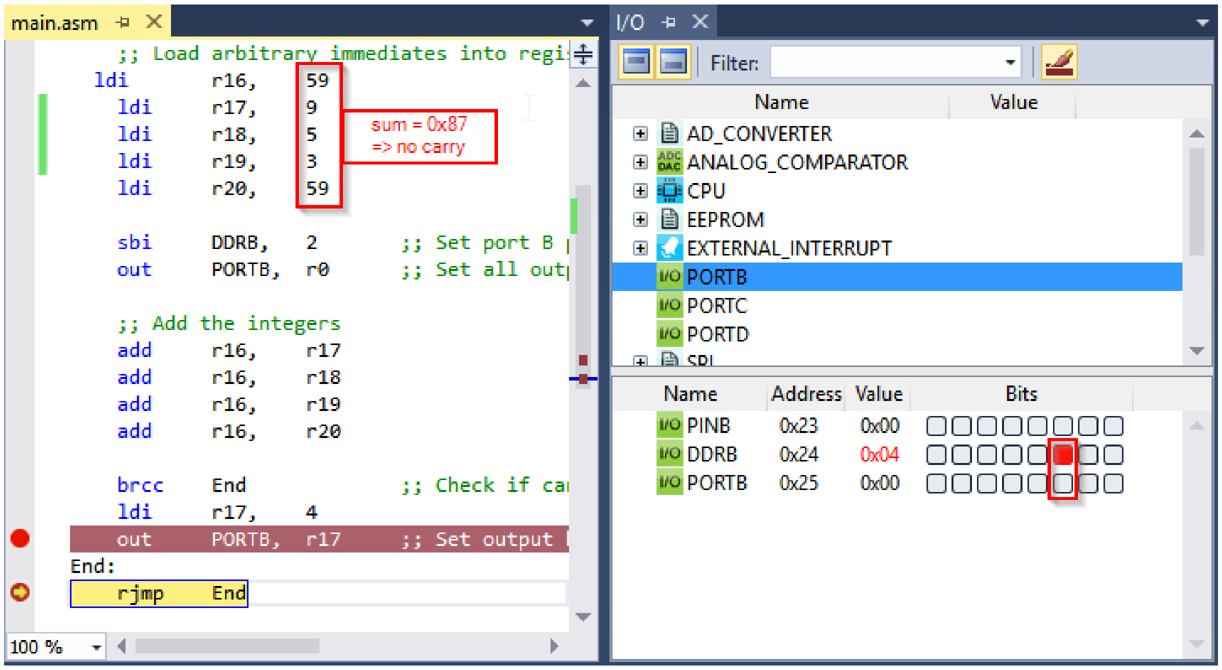
Before with no overflow



```
.cseg
start:
    ;; Load arbitrary immediates into regi
  ldi      r16,    59
    ldi      r17,    9
    ldi      r18,    5
    ldi      r19,    3
    ldi      r20,    59

    sbi      DDRB,   2        ;; Set port B
    out      PORTB,  r0       ;; Set all out

    ;; Add the integers
    add      r16,    r17
    add      r16,    r18
    add      r16,    r19
    add      r16,    r20

    brcc     End              ;; Check if ca
    ldi      r17,    4
    out      PORTB,  r17      ;; Set output
End:
    rjmp     End
```

Name	Address	Value	Bits
PINB	0x23	0x00	☐☐☐☐☐☐☐☐
DDRB	0x24	0x00	☐☐☐☐☐☐☐☐
PORTB	0x25	0x00	☐☐☐☐☐☐☐☐

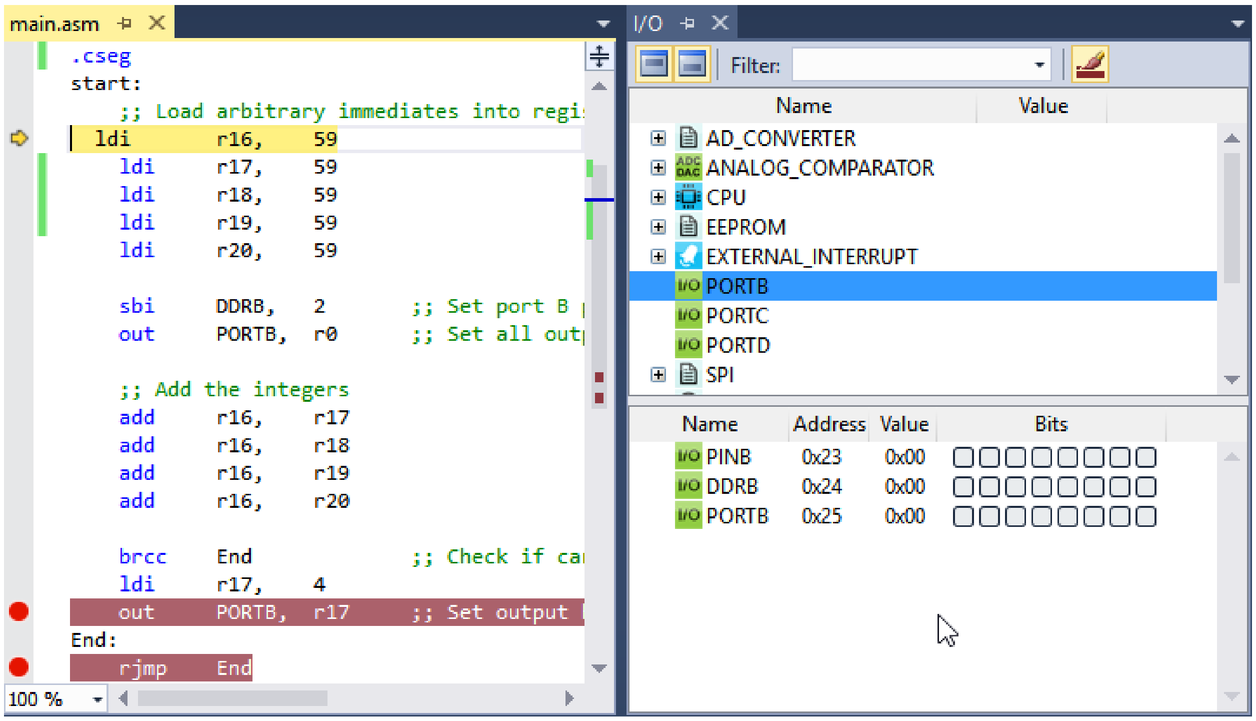
After with no overflow



```
    ;; Load arbitrary immediates into regi
  ldi      r16,    59
    ldi      r17,    9
    ldi      r18,    5
    ldi      r19,    3
    ldi      r20,    59

    sbi      DDRB,   2        ;; Set port B
    out      PORTB,  r0       ;; Set all out

    ;; Add the integers
    add      r16,    r17
    add      r16,    r18
    add      r16,    r19
    add      r16,    r20

    brcc     End              ;; Check if ca
    ldi      r17,    4
    out      PORTB,  r17      ;; Set output
End:
    rjmp     End
```

Name	Address	Value	Bits
PINB	0x23	0x00	☐☐☐☐☐☐☐☐
DDRB	0x24	0x04	☐☐☐☐☐🟥☐☐
PORTB	0x25	0x00	☐☐☐☐☐☐☐☐

Before with overflow

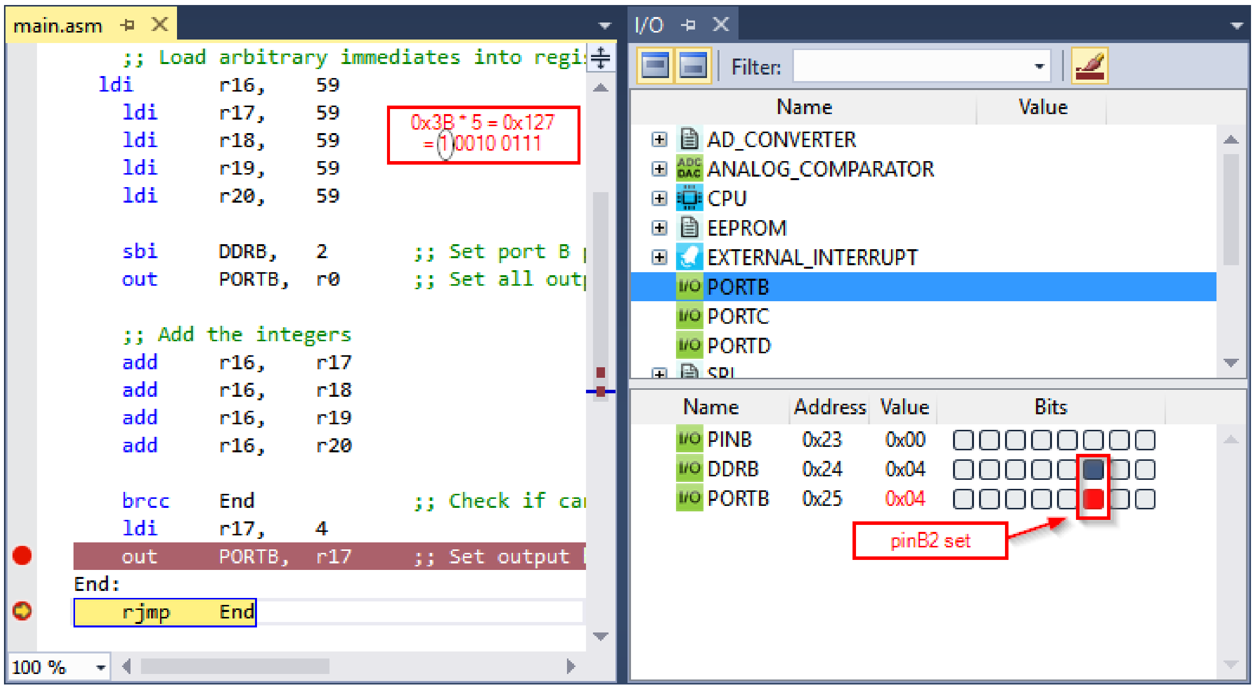


```
.cseg
start:
    ;; Load arbitrary immediates into regi
  ldi       r16,    59
    ldi     r17,    59
    ldi     r18,    59
    ldi     r19,    59
    ldi     r20,    59

    sbi     DDRB,   2           ;; Set port B
    out     PORTB,  r0          ;; Set all out

    ;; Add the integers
    add     r16,    r17
    add     r16,    r18
    add     r16,    r19
    add     r16,    r20

    brcc    End                 ;; Check if ca
    ldi     r17,    4
    out     PORTB,  r17         ;; Set output
End:
    rjmp    End
```

Name	Value
AD_CONVERTER	
ANALOG_COMPARATOR	
CPU	
EEPROM	
EXTERNAL_INTERRUPT	
PORTB	
PORTC	
PORTD	
SPI	

Name	Address	Value	Bits
PINB	0x23	0x00	☐☐☐☐☐☐☐☐
DDRB	0x24	0x00	☐☐☐☐☐☐☐☐
PORTB	0x25	0x00	☐☐☐☐☐☐☐☐

After with overflow



```
    ;; Load arbitrary immediates into regi
  ldi       r16,    59
    ldi     r17,    59
    ldi     r18,    59
    ldi     r19,    59
    ldi     r20,    59

    sbi     DDRB,   2           ;; Set port B
    out     PORTB,  r0          ;; Set all out

    ;; Add the integers
    add     r16,    r17
    add     r16,    r18
    add     r16,    r19
    add     r16,    r20

    brcc    End                 ;; Check if ca
    ldi     r17,    4
    out     PORTB,  r17         ;; Set output
End:
    rjmp    End
```

Name	Value
AD_CONVERTER	
ANALOG_COMPARATOR	
CPU	
EEPROM	
EXTERNAL_INTERRUPT	
PORTB	
PORTC	
PORTD	
SPI	

Name	Address	Value	Bits
PINB	0x23	0x00	☐☐☐☐☐☐☐☐
DDRB	0x24	0x04	☐☐☐☐☐■☐☐
PORTB	0x25	0x04	☐☐☐☐☐🟥☐☐

8.	SCREENSHOT OF EACH DEMO		

See simulation output on previous section.

9.	VIDEO LINKS OF EACH DEMO		
Videos were not requested			
10.	Github repository		
https://github.com/martinjaime/CpE301_Assignments2016S.git			

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"This assignment submission is my own, original work".
Martin Jaime-Viveros