Homework 2

(Due date: October 9th @ 5:30 pm)
Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (10 PTS)

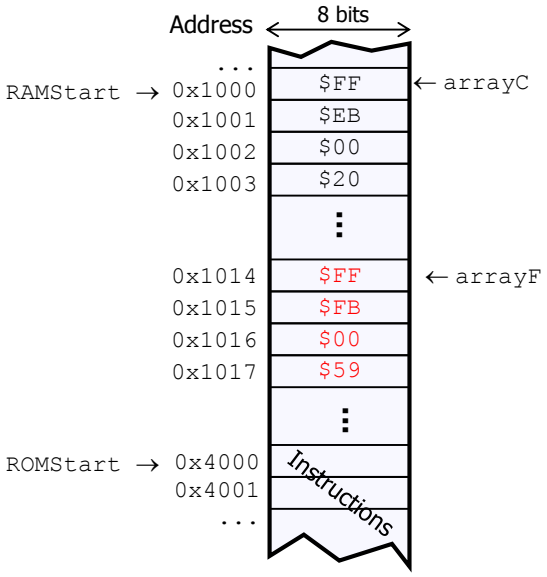
- Given a 24 MHz bus clock, provide a set of instructions to generate:
  - ✓ A time delay of 40 ms.
  - ✓ A time delay of 60 ms.

PROBLEM 2 (30 PTS)

- The following directives store a bunch of numbers in memory that represent degrees in Celsius.
  Complete the program (*provide a printout*) that converts those numbers to Fahrenheit degrees. Use a subroutine for the Celsius to Fahrenheit conversion ($F = \frac{C \times 9}{5} + 32$). Use a loop to convert every number in the array. Store the result in the `arrayF` array.

```
; Include derivative-specific definitions
       INCLUDE 'derivative.inc'
ROMStart    EQU  $4000  ;
N EQU 10

; variable/data section
            ORG  RAMStart  ;  Originate  data  at  address
RAMStart
; variables definition:
arrayC dc.w -21, 32, 45, 1120, 41, 13, -39, 100, 123, 0;
arrayF ds.w N;
```



- ✓ For the division by 5, only consider the integer part of the division. Also, keep in mind that the input array contains 16-bit signed numbers.

PROBLEM 3 (20 PTS)

- For the following code snippets, complete the value of the register when the last instruction is executed:

`ldaa #$8C`
`ldx #$04`
`loop: asra`
`inca`
`dbne X,loop`
`frvr: bra frvr`
	`ldaa #$59`
	`staa $F0`
	`brset $F0,$3A,next`
	`inc $F0`
	`next: asr $F0`
	`ldaa $F0`
		`ldd #$F3FE`
		`cba`
		`bmi next`
		`addd #$10F0`
		`next: deca`
		`decb`
			`movw #$40FF, $F1`
			`ldd #$7122`
			`addd $F1`
			`bvs next`
			`inca`
			`next: decb`

A =	**A =**	**D =**	**D =**

- For the following code snippets, specify a value of `B` that makes the branch instruction branch to '`next`':

`ldab #$___`
`cmpb #$EB`
`bhs next`
	`ldab #$___`
	`stab $FF`
	`dec $FF`
	`brclr $FF,$60,next`
		`ldab #$___`
		`clc`
		`rolb`
		`bcs next`
			`ldab #$___`
			`clc`
			`asrb`
			`cmpb #$EB`
			`bge next`
				`ldab #$___`
				`addb #$F1`
				`bpl next`
					`ldab #$___`
					`incb`
					`cmpb #$FB`
					`blt next`
						`ldab #$___`
						`eorb #$45`
						`bitb #$34`
						`beq next`

B =	**B =**	**B =**	**B =**	**B =**	**B =**	**B =**

PROBLEM 4 (20 PTS)

- Create an Assembly program (*provide a printout*) that reads the DIP Switch of the Dragon12-Light Board and displays the hexadecimal value present on the 4 LSBs. Utilize the 4 MSBs of the DIP Switch to determine which 7-segment displays to display: Bit 7 (MSB) controls display 3 (rightmost), bit 6 controls the display 2, bit 5 controls display 1, bit 4 controls display 0 (leftmost).

  Examples:
  - If DIP Switch: `11001001`, we display the character '9' on the two rightmost 7-segment displays.
  - If DIP Switch: `00011110`, we display the character 'E' on the leftmost 7-segment display.
  - If DIP Switch: `00001001`, no character is displayed.

PROBLEM 5 (20 PTS)

- Given the following Assembly code, specify the `SP` and the Stack Contents at the given times (right after the colored instruction has been executed). `SP` and the Stack Contents (empty) are specified for the first instruction (`LDS #$4000`).