Manual de Prácticas Microprocesadores

División: Ingeniería Eléctrica	Departamento: Ingeniería Electrónica

Programación en lenguaje Assembly (Ensamblador)

N.° de práctica: 04

Nombre completo del alumno		Firma
nombre		
N.° de brigada:	Fecha de elaboración	Grupo:

Manual de Prácticas Microprocesadores

División: Ingeniería Eléctrica	Departamento: Ingeniería Electrónica

1. Seguridad en la Ejecución

	Peligro o fuente de energía	Riesgo asociado
1	Manejo de Corriente Alterna	Electrochoque
2	Manejo de corriente Continua	Daño al equipo

2. Objetivos de aprendizaje.

El alumno aprenderá a realizar ciclos condicionales empleando lenguaje ensamblador. Diseñará la estructura de un programa que resuelva la implementación de un algoritmo iterativo como el cálculo de los primeros elementos de la serie de fibonacci.

3. Material y equipo.

Tarjeta de desarrollo y CCS IDE.

4. Actividad de investigación.

a) Examine el comportamiento de los siguientes segmentos de código. Identifique el resultado de las instrucciones ADD(S) y el estado de las banderas C,Z,N del APSR.

```
                .global main
                .text
main
                MOVW R0, #0xFFFF    ; carga solo 16 bits lsb
                MOVW R1, #0x1       ; carga solo 16 bits lsb
                ADD  R0, R1         ; R0 = 0xFFFF + 0x1,  no afecta banderas
```

```
                MOVW R0, #0xFFFF    ;
                ADDS R0, R1         ; R0 = 0xFFFF + 0x1, si afecta banderas
                ADDS R2, R1, #-1    ; suma con signo, destino especificado
```

```
                MOVW R1, #5
                MOVW R2, #0x0
LOOP1           ADD  R2, #10
                SUBS R1, #1        ; subs afecta banderas
                BNE LOOP1  ;"Branch if Not Equal" va a LOOP1 si la instrucción anterior no resulta 0
```

```
                MOVW R2, #0XABCD  ; carga un Registro con 32 Bits R2 = 0x1234ABCD
                MOVT R2, #0X1234
                B main
```

b) En un programa, ¿qué hace la siguiente instrucción?: `B etiqueta1`

c) Las instrucciones derivadas de `B` tienen un sufijo, que es una condición que se debe cumplir para realizar la instrucción `B`. Explique las siguientes sintaxis de la instrucción B con los diferentes sufijos.

Sufijo/cond	B*{cond} etiqueta*	BX*{cond} Rm*
EQ		
NE		
CS or HS		
CC or LO		
MI		
PL		

d) Explique la operación que realizan las siguientes instrucciones.

CBNZ *Rn, etiqueta*	CBZ *Rn, etiqueta*

e) Explique la función del siguiente segmento de código, después con un diagrama de flujo ilustre la secuencia de pasos realizada.

```
                MOV R3, #0x64       ;comentarios
                B Test
Loop            .                   ;aquí hace cualquier otra operación
                .
                .
                SUB R3, R3, #1
Test            BNE Loop
```

5. Desarrollo.

Escriba y depure un programa que:

1) Emplee ciclos, para llenar un arreglo (memoria reservada en RAM) de 100 localidades de 1 Byte con la secuencia 0,1,2,…,99.
2) En otro ciclo, se realiza la suma (acumulada) de todos los valores de esta lista y la suma parcial se almacena en otra sección de datos.

lista	0	1	2	3	4	5	6	7	8	9	…
suma	0	1	3	6	10	15	21	28	36	…	…

Para auxiliarse en el direccionamiento de la memoria, defina dos variables donde en una guarde la dirección inicio y en otra la dirección final de cada lista.

Utilice las instrucciones de carga LDR y almacenamiento STR. Puede auxiliarse de la siguiente plantilla. **Agregue las etiquetas y saltos condicionales necesarios**.

```
N       .equ    100
                            ; variables no inicializadas
        .bss    lista, N    ; reserva 100 localidades de 8 bits para lista
        .bss    data, N     ; reserva 100 localidades de 8 bits para data

        .global main
        .text               ; programa principal
                            ; estos punteros son constantes colocados en ROM
ptLista .field  lista, 32   ; ptXXXX = dirección de 32 bits
ptData  .field  data,  32

main:                       ; inicio del programa

        LDR   R4, ptLista   ; R4 apunta a inicio de lista

; Llenar las 100 localidades de 'lista' con la secuencia 0-99
; usar R2 como la fuente del dato a guardar
; escribir su código aqui


        STR   R2, [R4]      ; guarda R2 en dirección apuntada por R4
;_____________________________________________________________________________
;sumas parciales

        LDR   R3, ptData    ; R3 apunta a inicio de data

; LLenar las 100 localidades de ‘data’ con la suma parcial de las localidades
; usar R2 como la fuente del dato a guardar
; escribir su código aqui

        STR   R2, [R3]      ; guarda R2 en dirección apuntada por R3

stop    B   stop
```

2) Escriba un programa para escribir en 10 localidades en RAM la serie de Fibonacci. La serie es 0, 1, 1, 2, 3, 5, 8, 13… Asuma los primeros dos números de la serie: 0,1; calcule el resto.

6. Cuestionario.

¿Para qué sirve la directiva .equ?

¿De qué tipo debe ser una instrucción aritmética, para que tenga efecto en las banderas de condición de programa?

¿Qué instrucciones de salto condicional se puede emplear para realizar ciclos?

Presente en lenguaje ensamblador, la estructura de los ciclos de control: while-do, do-while, if A<B else, if A>B else, if A= = 0 else.

7. Conclusiones.

De cada estudiante y de carácter obligatorio en el informe correspondiente.

8. Bibliografía.