Universidad Nacional Autónoma de México
Facultad de Ingeniería
2018-1
Lab. Microcontroladores y Microprocesadores - Grupo 03
Pérez Navarro Maria Yesica - 414039694
dd/mm/yyyy
Práctica 05: Programación en lenguaje
Assembly (Ensamblador) 2

---

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 utilizar las instrucciones de operaciones aritméticas y lógicas, así como emplear saltos condicionales y llamados a subrutinas.

3. Material y equipo.

- Calculadora en modo “Programador”.
- Tarjeta de desarrollo.
- CCS IDE.

4. Actividad previa.

a) ¿Cómo se obtiene la representación binaria de un número negativo en el formato de complemento a 2?
Se niega el número y se le suma un uno.

b) ¿Cuál es la longitud en bytes del resultado de una multiplicación no signada (unsigned multiplication) al multiplicar:

- dos valores de 16 bits cada uno?
  El resultado tiene una longitud de 32 bits.
- dos valores de 32 bits cada uno?
  El resultado tiene una longitud de 64 bits.

c) ¿Qué hace el salto condicional “BEQ etiqueta1” ?
Salta hacia la “etiqueta1” si el resultado de una comparación u operación aritmética es igual a 0 (la bandera Z es igual a 0).

d) ¿Qué hace el salto “BL etiqueta2” ?
Salta hacia una subrutina(“etiqueta2”) y guarda una dirección de retorno en un registro.

e) Para cada uno de los dos saltos anteriores, ¿se debe regresar al lugar del salto?, ¿cómo?
Sólo para el salto BL, para regresar a la direccion de retorno se hace con la instruccion BX.

f) Examine el comportamiento de los segmentos del código proporcionado. Con ayuda de la calculadora, compruebe el resultado de las operaciones.

```
            .data
            .retain         ;mantiene en memoria variables o símbolos
                            ;no referenciados en programa
;******** Seccion de datos no inicializados *****************
            .bss lista1, 10

;******** Seccion de datos inicializados *****************
A           .word 12, 21, 32, 42, 21, 74, 88, 91, 3, 121
            .byte 12, 21, 32, 42, 21, 74, 88, 91, 3, 121

;********** ************************************************
            .global main
                            .text
                            .align 4


main:
MOV R0, #5
; para decrementar un Registro.
SUBS R0, R0,#5
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;   Operación resta
; instruccion substraccion con efecto en banderas
; revisar el estado de la bandera C
; resultados > 0 (C=1)
MOV  R2,#0x4F          ;R2 = 0x4F   =79
MOV  R3,#0x39          ;R3 = 0x39   =57
SUBS R4,R2,R3          ;R4 = R2 - R3  = 22

MOV  R2, #0x8888
MOVT R2, #0x8888
MOV  R3, #0x3333
MOVT R3, #0x3333
SUBS R4,R2,R3          ;R4 = R2 - R3

; resultado < 0 (C=0)
MOV  R1,#0x4C          ;R1 = 0x4C   =76
MOV  R2,#0x6E          ;R2 = 0x6E   =110
SUBS R0,R1,R2          ;R0 = R1 - R2
; si es negativo N=1
; el resultado está en complemento a 2
; hay que negarlo y sumarle 1
; el resultante es el valor negativo

; Reverse Substract. util para obtener el complemento a 2 de un número
; invierte el orden de los operandos en una substraccion
; Sintaxis        RSB Rd,Rn,Op2 ;Rd = Op2 - Rn
MOV R1, #0x1          ;R1=1
RSB R0,R1,#0          ;R0= 0 - R1 = 0 - 1
```

```
MOV R1, #0x3          ;R1=0x03
RSB R0,R1,#0          ;R0= 0 - R1

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Operación Multiplicación

; multiplicacion usando MUL : Half_Word1 * Half_Word2 = 16bits*16bits=32bits
MOVW R0, #0xFFFF
MOVW R1, #2
MUL  R2, R0,R1

MOV R0, #0xFFFF
MOV R1, #0xFFFF
MUL R2, R0,R1

; multiplicacion de valores mayores a 16 bits: operandos deben estar en registros
MOVW R0, #0x86A0   ; R0= 100000
MOVT R0, #0x1
MOVW R1, #0x49F0   ; R1= 150000
MOVT R1, #0x2
MUL R3, R0, R1     ; solo se mantienen los 32 LSB R1*R0= 3 7E11 D600

; multiplicacion para resultados mayores a 32 bits (Unsigned MULtiplication)
UMULL R3,R4, R0,R1  ; RdHI,RdLO, Rn,Op2

; operacion MuLtiplicacion y Acumulación (esencial para procesamiento de señales), operando
; Sintaxis        MLA Rd,Rm,Rs,Rn ;Rd = (Rm × Rs) + Rn
MOV R1, #100                ;R1 = 100
MOV R2, #5                        ;R2 = 5
MOV R3, #40                ;R3 = 40
MLA R4,R1,R2,R3         ;R4 = (R1 × R2) + R3 = (100 × 5) + 40 = 540

MOV R0, #4                        ; inicializa contador
MOV R1, #1
MOV R2, #2
MOV R4, #0
func1                        MLA R4,R1,R2,R4               ; Multiplica y acumula:  R4 = R
ADDS R1,#1
ADDS R2,#2
SUBS R0,#1                        ; decrementa contador
BNE func1                        ; termina ciclo?

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;        Instrucciones lógicas
MOVB R0, #10010000b
MOVB R1, #10000000b
ANDB R2,R0,R1               ; AND

; EOR: Exclusive OR, enciende bits apagados o apaga bits encendidos = "Toggle", especificac
MOVB R3,#80                        ; R3=1000 0000
EOR R3,R3,#0x01            ; enciende bit 0, R3=0000 0001
EOR R3,R3,#0x01               ; apaga bit 0,  R3=0000 0000
EOR R3,R3,#0x11               ; enciede bit 4 y 0
```

```
; BIt Clear
BIC R3, #0x10                ; apaga bit 4,  R3=0000 0001
EOR R3, #0x11                ; enciende bit 4, apaga bit 1, R3=0001 0000

; ORR, permite encender bits de un registro sin modificar los demás
ORRS R3, #0x01


; preguntar si el bit 4 de R0 está encendido
; el valor del bit 4 se guarda en R2
ANDB R2,R0, #0x10        ; R0 && mascara
ASRS R2,#4                ; desplaza R2 4 bits a la derecha, actualiza banderas
CMP R2,#1                 ; el b4='1' ?

MOVB R0, #10010000b
TST  R0, #0x10        ; prueba si todos los bits indicados en mascara son '0', afecta Z
TST  R0, #0x90        ; la operacion es igual a ANDS pero no guarda resultado
TST  R0, #0x6F
TEQ  R0, #0x10        ; prueba si el registro tiene encendidos los bits indicados por la ma
TEQ  R0, #0x90        ; la operacion es igual a EORS pero no guarda resultado

; Saltos condicionales a secciones de programas,
; Llamado a subrutinas y regreso de subrutinas
MOV  R0, #2
loop1                        SUBS R0,#1        ; decrementa R0
CMP  R0,#0         ; es R0=0?
BEQ  func2  ; si, entonces "salta" a func2
NOP                         ; no, entonces sigue con programa
NOP
BL   func3        ; "llama" a func3 y debe regresar
NOP                         ; sigue despues de ejecutar func3
NOP
B loop1               ; repite loop1


func2       NOP                        ; ejecuta func2
NOP
B stop               ; termina programa

func3               NOP                         ; ejecuta func2
NOP
BX LR       ; regresa a donde fué llamada



stop               B stop
.end
```

5. Desarrollo.

Escriba un programa que funcione como una calculadora de números enteros de hasta 32 bits, con operaciones de Suma, Resta, Multiplicación, AND, OR, XOR.

Para elegir la operación a realizar, desde el depurador se escribirá un valor en R0 del 1 al 6 (1:suma, 2:resta, etc.). El programa debe hacer llamados a dos subrutinas: una para carga de operandos y otra para guardar resultado.

Los dos operandos se encuentran definidos en memoria RAM y se deben colocar en R1 y R2. El resultado se escribe también en memoria RAM, usando R3.

Por tanto se debe hacer acceso de lectura y escritura (LDR, STR).

Los operandos se pueden modificar en cualquier sobre-escribiendo su valor en memoria RAM.

6. Cuestionario

1. Después de ejecutar la instrucción SUBS, ¿qué signo tiene el resultado cuando las bandera C = 0, y cuando C=1? En cada caso, qué valor debe tener la bandera N?

   Para un valor en un Registro, en lenguaje ensamblador.

2. ¿Cómo se enciende un bit específico sin afectar a los demás? (2 maneras).

3. ¿Cómo se apaga un bit específico sin afectar a los demás? (2 maneras).

4. ¿Cómo se invierte el estado de un bit específico?.

5. Al emplear instrucción de comparación CMP, ¿Qué nos dice la bandera Z?.

6. Se quiere saber si los 4 LSB de un byte son '0', ¿cómo emplearía la instrucción TST y qué condición/bandera se debe cumplir/establecer?.

7. Se quiere saber si los 4 MSB de un byte son '1', ¿cómo emplearía la instrucción TEQ y qué condición/bandera se debe cumplir/establecer?

8. Para qué sirve la directiva .retain

7. Conclusiones.

Referencias

[1] Como citar: `http://www.cva.itesm.mx/biblioteca/pagina_con_formato_version_oct/apa.htm`

[2] Autor, (Fecha de publicacion), Titulo, paginas, Fecha de recuperacion, Sitio web: `http://www.google.com`

[3] Repositorio del proyecto `https://github.com/penserbjorne`