
# IS-603 Arquitectura de Computadoras

## I Parcial - Instrucciones de control de flujo

IS-UNAH

I PAC 2024

# Contenido



# Recordatorio

# Flags del Registro de estado

Indica el estado actual del procesador, empleando para ellos los indicadores de condición.

- **N** Se activa cuando el resultado de la operación realizada es negativo.
- **Z** Se activa cuando el resultado de la operación realizada es cero.
- **C** Se activa cuando el resultado de la operación produce un acarreo.
- **V** Se activa cuando el resultado de la operación produce un desbordamiento.

¿Cuál es el propósito del PC (Contador de Programa)?

¿Cuál es el propósito del PC (Contador de Programa)?

El contador de programa o PC, en el simulador es el registro 15, almacena la dirección de memoria de la siguiente dirección a ejectuar.

# Saltos incondicionales y condicionales

# Saltos incondicionales [a]

¿Cuál es el valor de r3 al finalizar el programa?

```
1          .text
2  main:   mov r0, #5
3          mov r1, #10
4          mov r2, #100
5          mov r3, #0
6          b salto
7          add r3, r1, r0
8  salto:  add r3, r3, r2
9  stop:   wfi
```

[a] Dibuje el diagrama de flujo del programa

# Saltos condicionales

| Instrucción | Código | Condición de salto |
|---|---|---|
| «**beq**» *(branch if equal)* | 0000 | Igual (`Z`) |
| «**bne**» *(branch if not equal)* | 0001 | Distinto (`z`) |
| «**bcs**» *(branch if carry set)* | 0010 | Mayor o igual sin signo (`C`) |
| «**bcc**» *(branch if carry clear)* | 0011 | Menor sin signo (`c`) |
| «**bmi**» *(branch if minus)* | 0100 | Negativo (`N`) |
| «**bpl**» *(branch if plus)* | 0101 | Positivo o cero (`n`) |
| «**bvs**» *(branch if overflow set)* | 0110 | Desbordamiento (`V`) |
| «**bvc**» *(branch if overflow clear)* | 0111 | No hay desbordamiento (`v`) |
| «**bhi**» *(branch if higher)* | 1000 | Mayor sin signo (`Cz`) |
| «**bls**» *(branch if lower or same)* | 1001 | Menor o igual sin signo (`c` o `Z`) |
| «**bge**» *(branch if greater or equal)* | 1010 | Mayor o igual (`NV` o `nv`) |
| «**blt**» *(branch if less than)* | 1011 | Menor que (`Nv` o `nV`) |
| «**bgt**» *(branch if greater than)* | 1100 | Mayor que (`z` y (`NV` o `nv`)) |
| «**ble**» *(branch if less than or equal)* | 1101 | Menor o igual (`Nv` o `nV` o `Z`) |

# If-then[b]

```
1          .data
2  X:      .word 1
3  Y:      .word 1
4  Z:      .word 0
5
6          .text
7  main:   ldr r0, =X
8          ldr r0, [r0]
9          ldr r1, =Y
10         ldr r1, [r1]
11
12         cmp r0, r1
13         bne finsi
14           add r2, r0, r1
15           ldr r3, =Z
16           str r2, [r3]
17
18 finsi:  wfi
```

[b] ¿Cuál es el valor de r2 al finalizar el programa? Dibuje cómo queda la memoria al finalizar el programa.

Por cada uno de los códigos de las diapositivas siguientes, intente realizar un bosquejo del programa e identifique cuándo se ejecutan los saltos condicionales.

Pregúntese si desconoce alguna instrucción, dibuje un diagrama de flujo o un pseudocódigo, para que antes de ejecutar el programa en el simulador, ya tenga una idea de qué resultado obtendrá.

Un buen ejercicio adicional sería dibujar la memoria, ubicar los datos, y también bosquejar los registros y su contenido. Identifique un método que le permita organizar la información al realizar las pruebas de escritorio.

# If-then-else

```
1          .data
2  X:      .word 1
3  Y:      .word 1
4  Z:      .word 0
5
6          .text
7  main:   ldr r0, =X
8          ldr r0, [r0]
9          ldr r1, =Y
10         ldr r1, [r1]
11
12         cmp r0, r1
13         bne else
14           add r2, r0, r1
15           b finsi
16
17 else:     add r2, r0, #5
18
19 finsi:  ldr r3, =Z
20         str r2, [r3]
21
22 stop:   wfi
```

```
                .data
X:      .word 1
E:      .word 1
LIM:    .word 20

        .text
main:   ldr r0, =X
        ldr r0, [r0]
        ldr r1, =E
        ldr r1, [r1]
        ldr r2, =LIM
        ldr r2, [r2]
bucle:  cmp r0, r2
        bge finbuc
          lsl r3, r1, #1
          add r0, r0, r3
          add r1, r1, #1
          ldr r4, =X
          str r0, [r4]
          ldr r4, =E
          str r1, [r4]
        b   bucle

finbuc: wfi
```

```
1          .data
2  V:      .word 2,4,6,8,10
3  n:      .word 5
4  suma:   .word 0
5
6          .text
7  main:   ldr r0, =V
8          ldr r1, =n
9          ldr r1, [r1]
10         ldr r2, =suma
11         ldr r2, [r2]
12         mov r3, #0
13 bucle:  cmp r3, r1
14         beq finbuc
15           ldr r4, [r0]
16           add r2, r2, r4
17           add r0, r0, #4
18           add r3, r3, #1
19         b bucle
20
21 finbuc: ldr r0, =suma
22         str r2, [r0]
23
24 stop:   wfi
```

# Ejercicios

# Ejercicios

Desarrolle los siguientes ejercicios utilizando el simulador. Trate de bosquejarlos en papel y entender el ejercicio antes de ejecutarlo y comprobar sus resultados.

1. 5.10
2. 5.12
3. 5.14
4. 5.17