

# EXPLICACIÓN GENERAL

## Ejercicio 2:

### SUB dir

**GPR(AD) -> MAR:** (Preparar dirección: Carga la dirección del dato (que estaba en GPR) en el registro de direcciones (MAR) para acceder a la memoria.)

**M -> GPR:** (Leer dato: Lee el valor guardado en esa dirección de memoria y lo guarda temporalmente en el registro GPR.)

**Acc - GPR -> Acc:** (Restar: Resta el valor leído (GPR) al valor que ya tenías en el Acumulador. El resultado se guarda en el Acumulador. (Fin de instrucción).)

### SUBI dir

**GPR(AD) -> MAR:** (Buscar puntero: Coloca la dirección inicial en el MAR.)

**M -> GPR:** (Leer puntero: Lee el contenido de memoria. Este valor no es el dato a restar, sino una nueva dirección (puntero) donde está el dato real.)

**GPR(AD) -> MAR:** (Ir al dato real: Carga esa nueva dirección (que acabamos de leer) en el MAR.)

**M -> GPR:** (Leer dato real: Ahora sí, lee el valor numérico final de la memoria y lo guarda en GPR.)

**Acc - GPR -> Acc:** (Restar: Realiza la resta final: Acumulador menos el dato encontrado.)

## Ejercicio 3:

### ADDV11 val (el truco es reducir el número de bits de 16 a 11)

**GPR -> PC ; PC -> GPR:** (Intercambio (Swap): Guarda el PC actual en GPR y carga el valor de GPR en el PC.)

**GPR -> PC ; PC -> GPR:** (Re-intercambio: Vuelve a hacer lo mismo. Al hacerlo dos veces seguidas, los valores vuelven a su sitio original.)

**Acc + GPR -> Acc:** (Suma: Suma el valor que hay en GPR al Acumulador. Esta es la operación útil de la instrucción.)

### SUBV11 val (el truco es reducir el número de bits de 16 a 11)

**GPR -> PC ; PC -> GPR:** (Intercambio (Swap): Guarda el PC actual en GPR y carga el valor de GPR en el PC.)

**GPR -> PC ; PC -> GPR:** (Re-intercambio: Vuelve a hacer lo mismo. Al hacerlo dos veces seguidas, los valores vuelven a su sitio original.)

**Acc - GPR -> Acc:** (Resta: Resta el valor de GPR al Acumulador.)

## Ejercicio 4:

### DJNZ dir

**Acc complementado + 1 -> Acc:** (Inicio de la resta: Convierte el valor del Acc en su negativo (-Acc).)

**Acc + 1 -> Acc:** (Suma 1 al negativo (-Acc + 1).)

**Acc complementado + 1 -> Acc:** (Fin de la resta: Niega el resultado anterior. Matemáticamente esto resulta en  $\text{Acc} - 1$ .)

**GPR -> QR:** (Salvar la dirección: Guardamos la dirección de salto (que está en GPR) en el registro auxiliar QR para no perderla si necesitamos usar el bus.)

**Acc -> GPR:** (Mover resultado: Pasamos el valor decrementado al GPR para activar las banderas (Flags) o prepararlo.)

### (Condicional)

**La Decisión:** Verifica la bandera Z (Cero).

- Si **Z=0 (No es cero)**: ¡BIFURCA! (Salta las siguientes líneas y va a ejecutar el salto).
- Si **Z=1 (Es cero)**: INCREMENTA (Sigue a la línea de abajo, o sea, no salta).

**(Bifurca) Caso "Es Cero":** Como el contador llegó a 0, **terminamos aquí**. Esta instrucción devuelve el control al ciclo de captación (Fetch). No se produce el salto.

**QR -> GPR:** (Recuperar dirección: (Solo llegamos aquí si NO era cero). Recuperamos la dirección de salto que guardamos en QR al principio.)

**GPR -> PC:** (Ejecutar Salto: Cargamos esa dirección en el Contador de Programa. El bucle se repite.)

## Ejercicio 5:

### JMPR n

**0 -> Acc:** (Limpieza: Pone el Acumulador a 0 para empezar a operar.)

**Acc + GPR -> Acc:** (Carga: Pasa el valor del salto ( $n$ ), que estaba en GPR, al Acumulador.)

**A complementado + 1 -> A:** (Negación: Convierte  $n$  en  $-n$  (Complemento a 2).)

**Acc -> GPR:** (Transferencia: Mueve  $-n$  al registro GPR.)

**GPR complementado -> GPR:** (El Truco Matemático: Aplica NOT a GPR (Complemento a 1). El resultado es  $n-1$ .)

**0 -> Acc:** (Limpieza: Vuelve a poner el Acc a 0.)

**Acc + GPR -> Acc:** (Recarga: Pasa el valor calculado ( $n-1$ ) al Acumulador.)

**PC -> GPR:** (Capturar PC: Guarda el valor actual del Contador de Programa en GPR.)

**Acc + GPR -> Acc:** (Suma Final: Suma el PC con el desplazamiento calculado.  $Acc = PC + (n-1).$ )

**Acc -> GPR:** (Preparar Salto: Mueve la dirección final calculada al GPR.)

**GPR -> PC:** (Ejecutar Salto: Actualiza el PC con la nueva dirección. ¡Salto relativo completado!)