

### 3. El ciclo de instrucción de una CPU

El **ciclo de instrucción de una CPU** es el **proceso fundamental** mediante el cual el **procesador ejecuta instrucciones en un programa**. Este ciclo se divide en **tres etapas principales**: **búsqueda, decodificación y ejecución**. A continuación, explicaremos este ciclo utilizando un ejemplo en ensamblador que incluye una instrucción de salto.

#### 3.1. Programa de ejemplo: Control de un contador con salto condicional

Este programa en ensamblador decrece un contador desde un valor inicial, verifica si ha llegado a cero y, si no, salta para repetir el proceso. Cuando el contador alcanza cero, el programa termina.

##### Código en ensamblador:

```
MOV CX, 5      ; Cargar el valor inicial 5 en el registro CX (contador)
LOOP_START:    ; Etiqueta para el inicio del bucle
DEC CX         ; Decrementar el valor en CX
JNZ LOOP_START ; Si CX no es cero, saltar a LOOP_START
HLT           ; Detener la ejecución cuando CX llegue a cero
```

##### Descripción de las instrucciones:

1. MOV CX, 5: Carga el valor 5 en el registro CX.
2. DEC CX: Decrementa el valor en el registro CX en 1.
3. JNZ LOOP\_START: Salta a la etiqueta LOOP\_START si el valor en CX no es cero.
4. HLT: Finaliza el programa cuando el bucle termina (CX = 0).

#### 3.2. Ciclo de instrucción y cambios en el Contador del Programa (PC)

El **Contador del Programa (Program Counter, PC)** es un **registro especial** que **almacena la dirección de la próxima instrucción a ejecutar**. A medida que **avanza** el ciclo de instrucción, el PC se actualiza para apuntar a la instrucción siguiente.

Etapa	Instrucción	PC Inicial	PC Final	Descripción
1. Búsqueda	MOV CX, 5	0x0000	0x0001	Se carga MOV CX, 5 desde la memoria.
2. Decodificación	MOV CX, 5	0x0001	0x0001	El procesador interpreta la instrucción.
3. Ejecución	MOV CX, 5	0x0001	0x0001	El valor 5 se almacena en el registro CX.
4. Búsqueda	DEC CX	0x0001	0x0002	Se carga DEC CX desde la memoria.
5. Decodificación	DEC CX	0x0002	0x0002	El procesador interpreta DEC CX.
6. Ejecución	DEC CX	0x0002	0x0002	Se decrementa CX (ej.: CX = 4).
7. Búsqueda	JNZ LOOP_START	0x0002	0x0003	Se carga JNZ LOOP_START.
8. Decodificación y Ejecución	JNZ	0x0003	0x0000 (salto)	Si CX $\neq$ 0 (verdadero), el PC apunta a LOOP_START.
9. Repetición				El ciclo continúa mientras CX $\neq$ 0.
Final	HLT	0x0003	0x0004	Cuando CX = 0, se ejecuta HLT y se detiene.

### Resultado del programa

El registro **CX** pasa por los valores:  $5 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 0$ . Cada vez que **CX** se decrementa, el salto condicional **JNZ** repite el bucle. Cuando **CX** alcanza 0, el programa se detiene.