Máquina de Von Neumann

Lenin G. Falconí

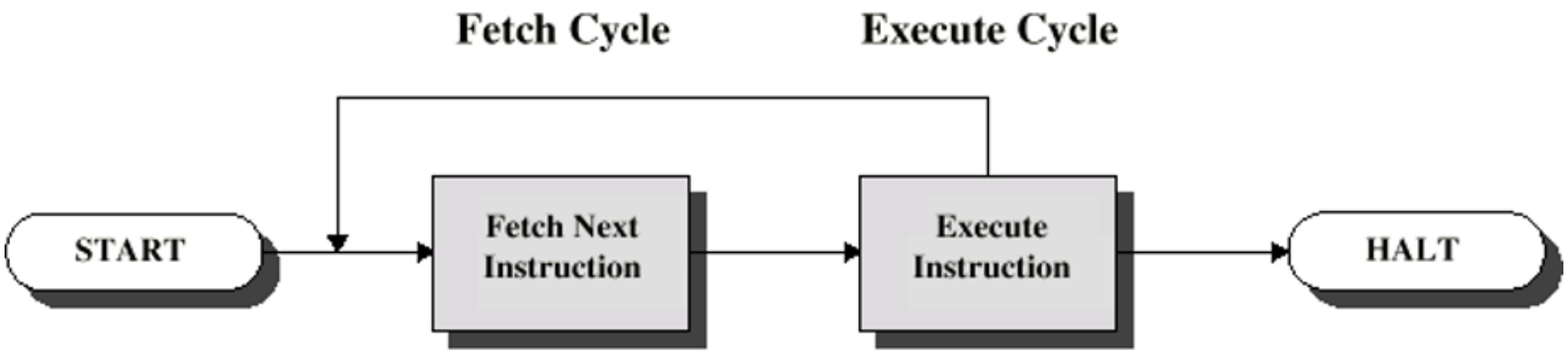
2024-06-04 Tue

Outline

Conceptos de Diseño I

- Los datos y las instrucciones se almacenan en una memoria principal de lectura/escritura
- Los contenidos de la memoria son accesibles por medio de su ubicación y son independientes del contenido o tipo de dato almacenado
- La ejecución ocurre de manera secuencial (aunque puede ser modificado explícitamente e.g. una instrucción de salto)
- El mismo hardware puede desarrollar diferentesfuncines sobre los datos dependiendo de las señales de control aplicadas
- El programa, conjunto de códigos de instrucciones y datos, indica las señales de control requeridas.
- La programación usa el mismo hardware pero diferentes códigos para diferentes propósitos.

Ciclo de Captación, decodificación y ejecución



Proceso de Captación I

- El contador de programa *PC* apunta a la siguiente instrucción
- El contenido del contador de programa se transfiere al *MAR* (Memory Addresds Register)
- *MAR* tiene la dirección de memoria desde donde se leerá datos/instrucciones o hacia donde se escribirá.
- En un ciclo de lectura, el contenido apuntado por *MAR* se transfiere al *MBR* Memory Buffer Register
- La instrucción se pasa del *MBR* al *IR* (instruction register), donde se decodifica en el *opcode* y la dirección del dato.
- La CPU ejecuta la instrucción.

Proceso de Captación II

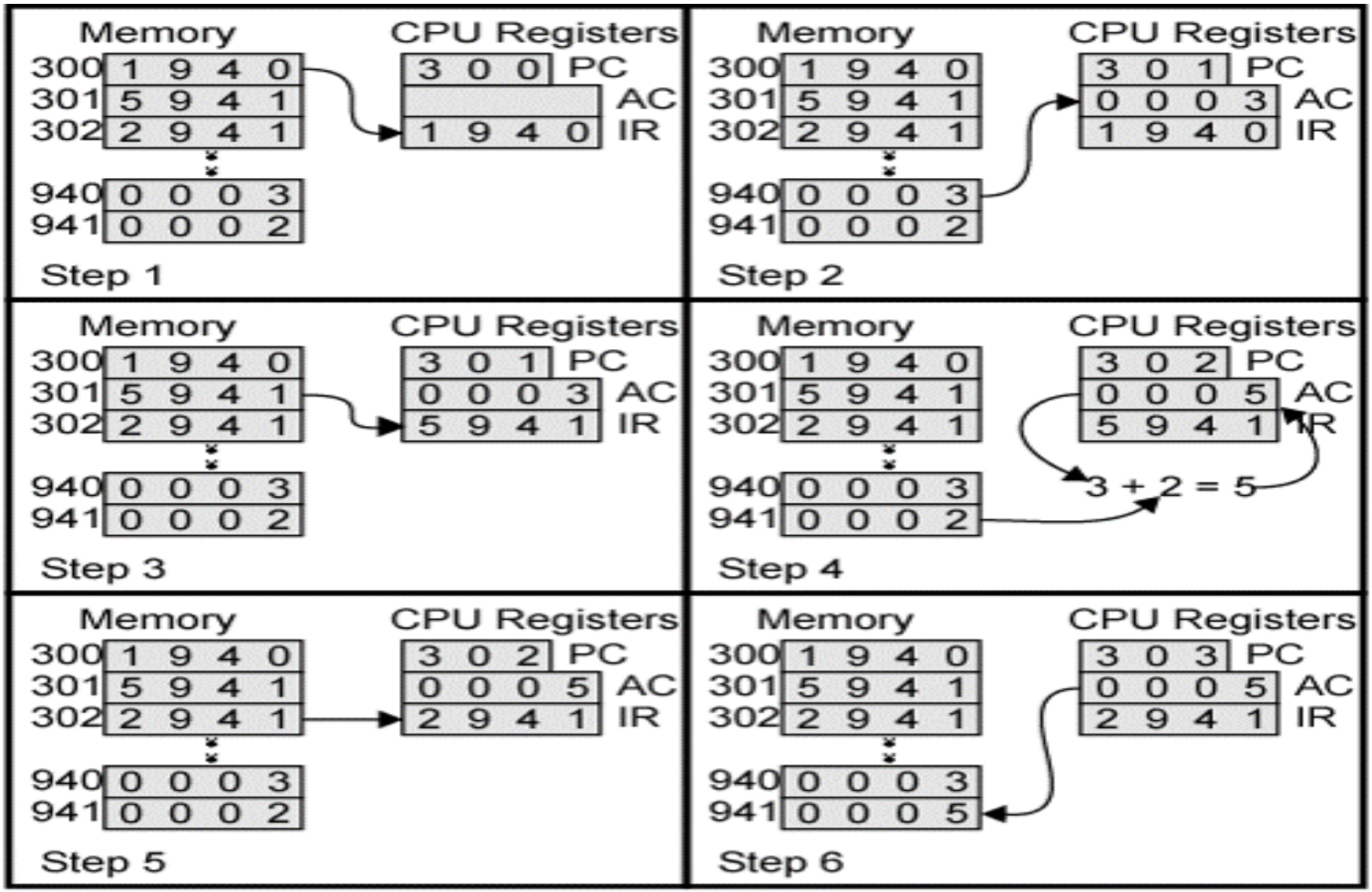
- El proceso de captación se puede escribir en notación RTL como:
  1. $[MAR] \leftarrow [PC]$
  2. $[PC] \leftarrow [PC] + 1$
  3. $[MBR] \leftarrow [[MAR]]$
  4. $[IR] \leftarrow [MBR]$
  5. $CPU \leftarrow [IR_{opcode}]$

Ejemplo I

Considere un computador de las siguientes características:

- Un único registro de acumulación AC
- La memoria es de 16 bits con 4 bits para opcode y 12 bits para direcciones de la memoria.
- ¿Cuántos Opcodes son posibles de almacenar?
- ¿Cuántas direcciones se puede alcanzar?
- El computador tiene el siguiente juego de instrucciones:
  - 0001 Cargar AC desde memoria
  - 0010 Almacenar AC en la memoria
  - 0101 Sumar al AC un dato de memoria

Ejemplo II


Memory
CPU Registers
300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2
3 0 0 PC
AC
1 9 4 0 IR
Step 1
3 0 1 PC
0 0 0 3 AC
1 9 4 0 IR
Step 2
3 0 1 PC
0 0 0 3 AC
5 9 4 1 IR
Step 3
3 0 2 PC
0 0 0 5 AC
5 9 4 1 IR
3 + 2 = 5
Step 4
3 0 2 PC
0 0 0 5 AC
2 9 4 1 IR
Step 5
3 0 3 PC
0 0 0 5 AC
2 9 4 1 IR
Step 6

Programa Fetch en Python

```
pc = 0
mem = [0]*16
def fetch(memory):
    global pc
    mar = pc
    pc = pc +1
    mbr = memory[mar]
    ir = mbr
    cu = ir >> 8
    address = ir & 0xFF
    return (cu, address)
```