

Tema-4--Unidad-de-Control.pdf



CodeWolf



Arquitectura de Computadores



2º Grado en Ingeniería Informática



**Escuela Politécnica Superior de Córdoba
Universidad de Córdoba**



[Accede al documento original](#)



Escuela de
Organización
Industrial

Contigo que evoluciones.
Contigo que lideras. Contigo que transformas.

**Esto es EOI.
Mismo propósito,
nueva energía.**



Descubre más aquí



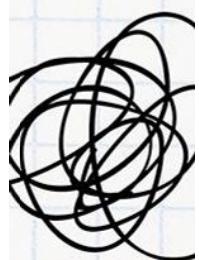
EOI Escuela de
Organización
Industrial

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo
espacio



Necesito
concentración

ali ali ooooh
esto con 1 coin me
lo quito yo...

wuolah

Tema 4 : Unidad de Control

➤ Visión Global

La unidad de control es un bloque de la Arquitectura de Von Neumann. Funciones:

- Asegurar la ejecución . (En M.P)
- Generación de las secuencias de microórdenes para la ejecución de todas y cada una de las instrucciones de la computadora.
- Captar y decodificar cada instrucción.
- Ejecución de la siguiente Instrucción de programa.

➤ Técnicas de diseño de la U.C:

→ Cableada:

- ◆ Mediante Registros de Desplazamientos: (Biestables)
- ◆ Como Sistema Secuencial Síncrono.
- ◆ Mediante Decodificadores de Tiempo e Instrucción.

→ Microprogramada:

- ◆ Mediante ROM de control: (Horizontal o Vertical)

➤ Diseño de la Unidad de Control Microprogramación.

- ROM de Control: Cada palabra está compuesta por todas las señales de microoperación del sistema.
- Las instrucciones estarán compuestas por varios ciclos, tanto de búsqueda - ejecución.
- Tras la ejecución de un paso es necesario saber qué paso será el siguiente que tendrá que realizar.

➤ Codificación de las microinstrucciones

- Las microinstrucciones proporcionan las señales de control a los distintos elementos que existen dentro del sistema.

Las señales de control que gobiernan una misma unidad se suelen agrupar en campos dentro de la codificación de las microinstrucciones.

El formato de las microinstrucciones:

- Formato no codificado.
- Formato completamente codificado.
- Formato codificado por trozos.
- Formato con solapamientos.
- Nanoprogramación.

➤ Codificación de las microinstrucciones

→ Formato no codificado:

Las micropalabras tienen un bit para cada señal de control. Problema que encontramos ROM muy grandes (Espacio), además no puede usarse paralelismo.

→ Formato completamente codificado:

Se codifica la activación de todos sus señales de control con menos bits (aunque en cada codificación sólo se activa 1 única señal de control). (Aumento de ciclos_ejec.)

→ Formato codificado por trozos:

Se codifica la activación de señales de control con menos bits agrupadas en campos. Permitiendo paralelismo. Problema: Aumento de la microinstrucción.

Los campos deben ser consistentes para que la división sea útil.

En todos los decodificadores hay que poner un NOP. (Parar).

→ Formato con solapamientos:

Si hay señales excluyentes entre sí, se pueden solapar campos, reduciendo el tamaño de las microinstrucciones.

➤ Tipos de Microprogramación.

→ Microprogramación horizontal:

- ◆ Formato no codificado.
- ◆ Microinstrucciones muy largas.
- ◆ Alto grado de paralelismo.
- ◆ Alto consumo de memoria de control.
- ◆ Más rápidas.

→ Microprogramación vertical:

- ◆ Formato codificado.
- ◆ Microinstrucciones cortas.
- ◆ Bajo paralelismo.
- ◆ Menor consumo de memoria de control.
- ◆ Menos rápidas.

→ Microprogramación vertical: Nanoprogramación:

- ◆ Objetivo: Reducir el tamaño de la memoria de control.
- ◆ Implica una memoria a 2 niveles.

→ Nanoprogramación:

- ◆ Se construye una memoria de m palabras de W bits, que contendrá las m microinstrucciones “únicas”.
- ◆ Se construye la memoria de control sustituyendo los W bits por la dirección única de la microinstrucción asociada en cada paso del microprograma.

➤ Secuencia en Microprogramación

→ Secuencia dentro de un microprograma:

- ◆ Tras la ejecución de cada microinstrucción se tiene que determinar qué nueva microinstrucción se ha de ejecutar.
- ◆ Hay 3 posibilidades:
 - Pasar a ejecutar la siguiente: (Incremento)
 - Pasar a ejecutar la que se encuentra en una dirección CROM (Bifurcación).
 - Pasar a ejecutar la primera asociada al ciclo de ejecución (Carga de Rutina).

Y esto puede ser Condicional o Incondicional.

El secuenciamiento se consigue variando la dirección contenida en el registro de dirección de acceso a la CROM.

Incremento : Pasar a la microinstrucción que se encuentra en la dirección CROM consecutiva ==> Incrementar el contenido de CMAR.

- **Bifurcación** : Saltar a una microinstrucción que se encuentra en una dirección CROM particular==> Dicha dirección debe ser indicada ==> Carga paralela de dicha dirección.
- **Bifurcación condicional** :Depende de los bits de estado del sistema se querrá llevar un rumbo u otro en casos determinados. (Z / F)
- **Carga de rutina**: Saltar a la microinstrucción de una de las rutinas de las instrucciones ==> La dirección de inicio de cada instrucción debe conocerse => Carga paralela de dicha dirección.

➤ Diseño de la Unidad de Control Cableado por Decodificadores Tiempo / Instrucción .

El microprogramado sigue existiendo => Codificación implícita mediante la existencia de cableado específico.Para cada micro operación:

- Se activa exclusivamente en un determinado ciclo de tiempo de una instrucción.
- La activación puede ser incondicional o estar condicionada a un valor de estado.

Una microoperación se ejecuta : $M_n = I_i * T_j * C_k$.

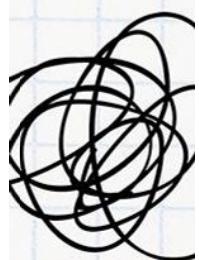
La suma de todas las veces , que se puede activar individualmente una microoperación , nos dará la lógica de activación completa.

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo
espacio



Necesito
concentración

ali ali ooooh
esto con 1 coin me
lo quito yo...

wuolah

> Diseño de la Unidad de Control Cableado por Decodificadores Tiempo / Instrucción

→ Material necesario :

- ◆ Contador de tiempo
- ◆ Señales :
 - Incremento
 - Carga paralelo
- ◆ Decodificador de tiempo.
- ◆ Decodificador de instrucciones.
- ◆ Lógica combinacional. (Puerta NOT , AND , OR)

→ Comparación de las técnicas de diseño

◆ Cableado mediante registros de Desplazamiento.

- También llamada mediante "trenes de biestables"
- Implementación sencilla.
- Alta flexibilidad.
- Facilidad de análisis.
- Bastante rápida.
- Minimiza la necesidad de lógica adicional.
- Alto uso de biestables (Muy por encima del óptimo).

◆ Cableado mediante Circuito Secuencial Síncronos

- Tedioso y rígido.
- Difícil de analizar.
- Difícil de detectar errores.
- Garantiza la ejecución más rápida de todos los métodos.
- Minimiza el número de biestables.

◆ Microprogramado mediante ROM de Control.

- Muy fácil de diseñar.
- Extremadamente flexible.
- Fácil de analizar.
- Es el mecanismo más lento.
- Costosa en circuitería (ROM, Registros , Lógica combinacional, etc).

◆ Cableado mediante Decodificadores de Tiempo e Instrucción.

- Sencillo de diseñar.
- Bastante flexible.
- Fácil de analizar.
- Tiene una ejecución bastante rápida.
- Bastante lógica combinacional.

wuolah