1.- Introducción

Introducción a los microprocesadores
2021

Lógica *"cableada"* vs *"programada"*

- Lógica "*cableada*"
  - Circuitos vistos en Diseño Lógico.
    - Lógica combinatoria, secuenciales Modo reloj, RTL,…
  - Función fija al momento de diseñar el circuito por conexiones físicas entre los componentes.
  - Cambiar la función implica nuevo diseño.

Lógica *"cableada"* vs *"programada"*

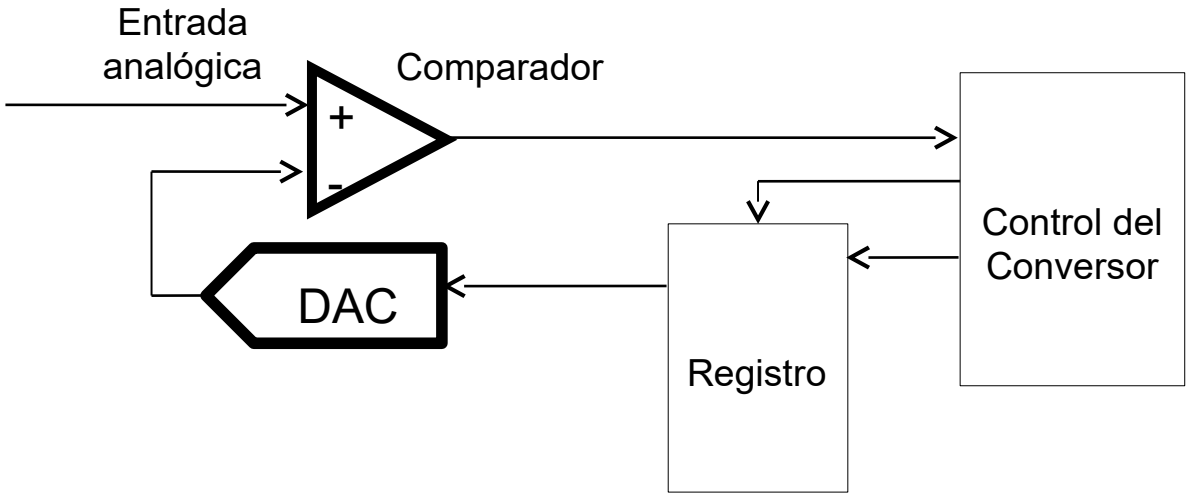
- Lógica *"programada"*
  - Un computador es un dispositivo diseñado para ser capaz de realizar una amplia variedad de tareas.
  - La lógica la define el usuario, no el diseñador del circuito.
  - Cambiar la función lógica no implica cambiar el circuito, sino solamente cambiar el contenido almacenado en una "memoria".

Compromiso
velocidad / flexibilidad / precio

- Un sistema por programa almacenado es más flexible con menores costos de diseño. Para implementar diferentes funciones basta cambiar el contenido de una memoria.
- Entonces, ¿por qué diseñar con lógica cableada?
  - La lógica programada es mucho más lenta (por ej. 100 a 1).
- Y entonces ¿por qué utilizar lógica programada?
  - La lógica programada es mucho más económica. Menor costo de diseño.

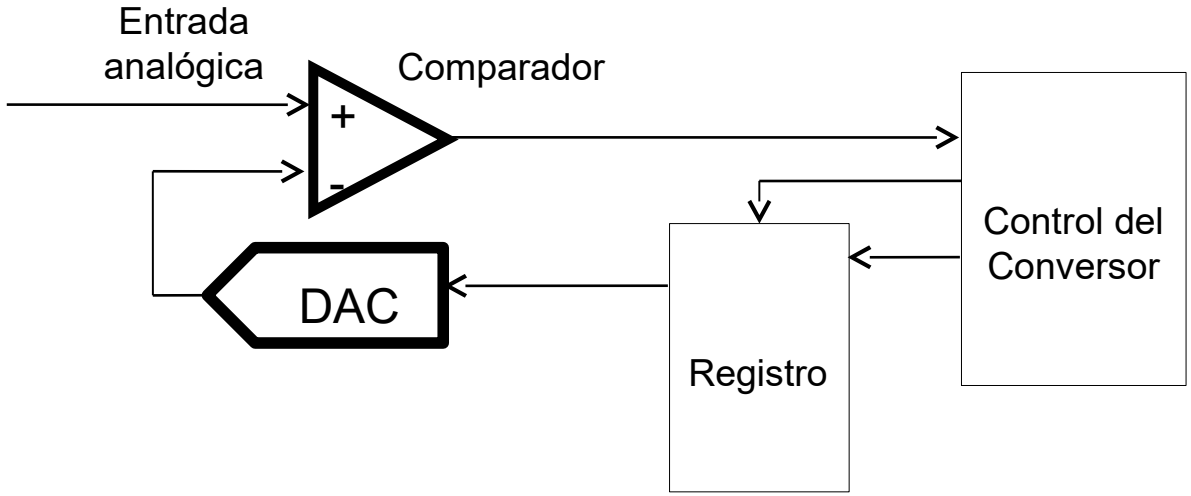
- La lógica cableada reconfigurable (FPGA) es un término medio entre lógica cableada pura y lógica programada. No implica cablear físicamente el nuevo circuito, pero sí diseñarlo.

Ejemplo: Conversor A/D de aproximaciones sucesivas



- más simple un DAC que un ADC
- Método “tonto”
  - Pruebo con reg = 0, 1, 2, …
  - Hasta que cambia el comparador
  - Método muy lento y con duración variable
- Método mejor: aproximaciones sucesivas

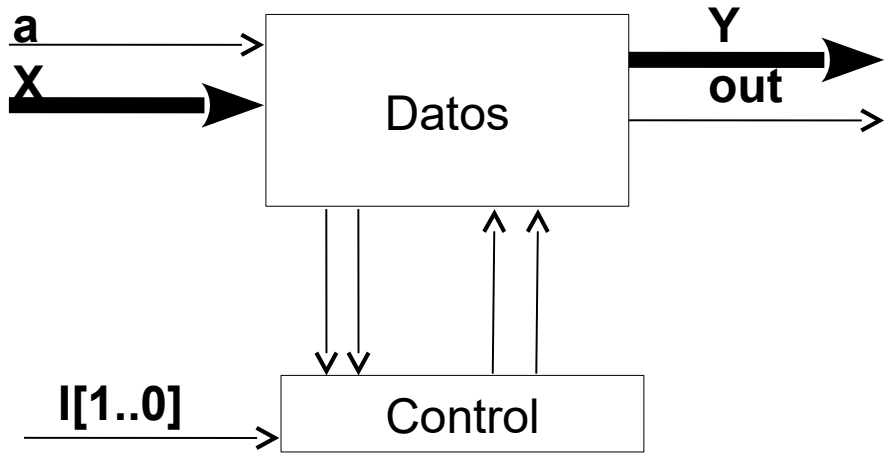
Ejemplo: Conversor A/D de 8 bits de aproximaciones sucesivas



- Reloj de 4Mhz
- Se requieren 8 pasadas por el comparador (una por cada bit)
- ¿Qué tan rápido se puede muestrear? (fs: frecuencia de muestreo)
- Por cada pasada un computador requiere de varias instrucciones y a su vez cada instrucción demora varios períodos de reloj.

Lógica cableada: fs = 500khz
Lógica programada: fs = 5khz

Ejemplo: Sistema RTL



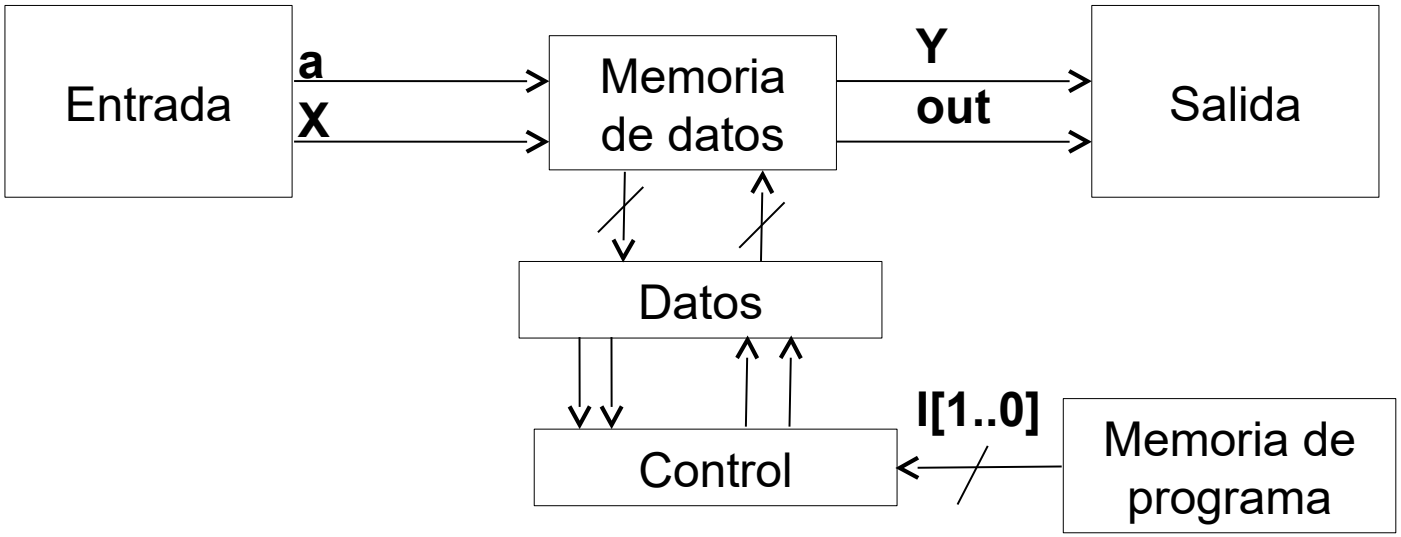
I[1..0]	Función
00	Suma
01	Resta
10	AND bit a bit
11	OR bit a bit

Sea el siguiente sistema que procesa datos:

- **a** va a 1 para indicar que el primer dato está pronto en **X**
- Más tarde **a** vuelve a 1 para indicar un segundo dato pronto en **X**
- El sistema debe después poner en **Y** el resultado de la operación indicada por **i[1..0]** de acuerdo a la tabla y avisar con **out**=1 que el resultado está listo.

Se puede diseñar muy fácilmente con las técnicas del curso de Diseño Lógico.
Manipulando en forma adecuada las señales I1,I0,a y X[] se puede realizar una secuencia de operaciones.

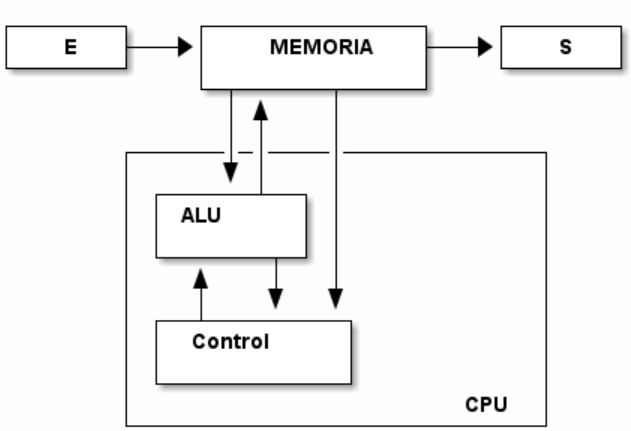
Ejemplo: Sistema RTL



- Si en lugar de manejar la secuencia de entrada desde el exterior, la almacenamos en memorias y dotamos al sistema de inteligencia necesaria para ir leyendo esta secuencia
  ➔ nos aproximamos a un computador.

- El bloque de control extrae INSTRUCCIONES de la MEMORIA DE PROGRAMA.
Los datos de entrada pueden cargarse previamente en la MEMORIA DE DATOS, donde también se almacenan los resultados.

Máquina de Von Neumann

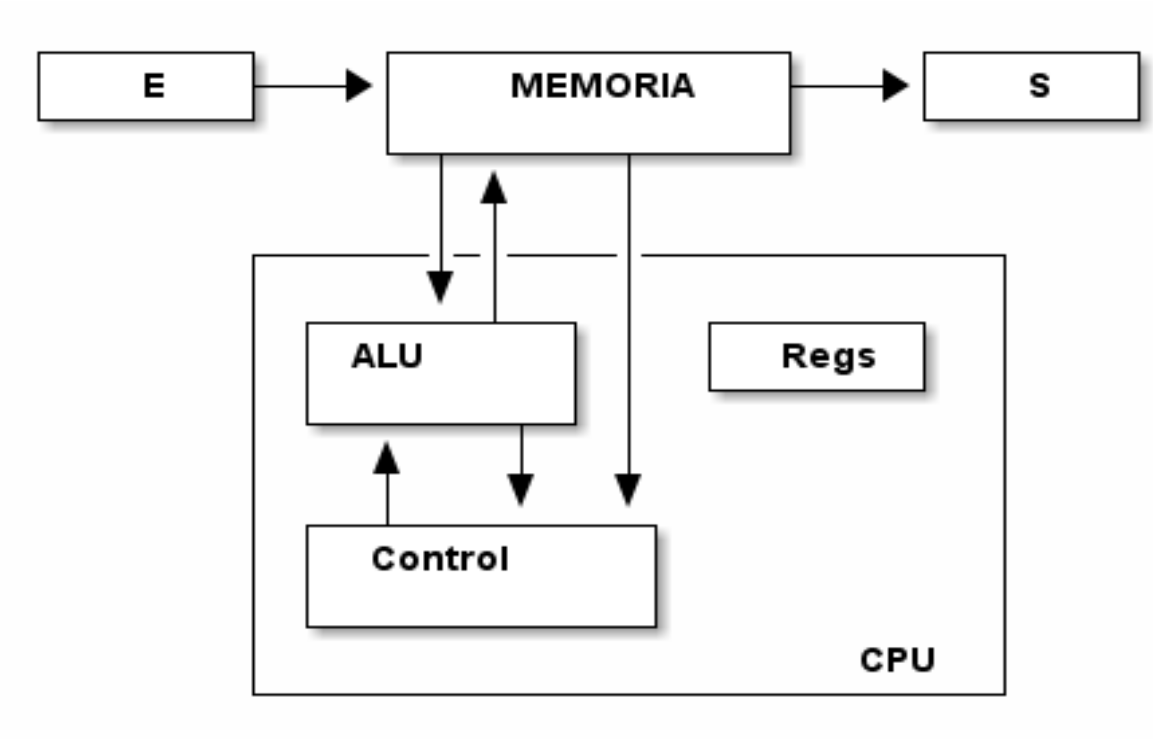


ALU + Control = CPU
(Unidad Central de Procesamiento)

Microprocesador
CPU contenida en un CHIP

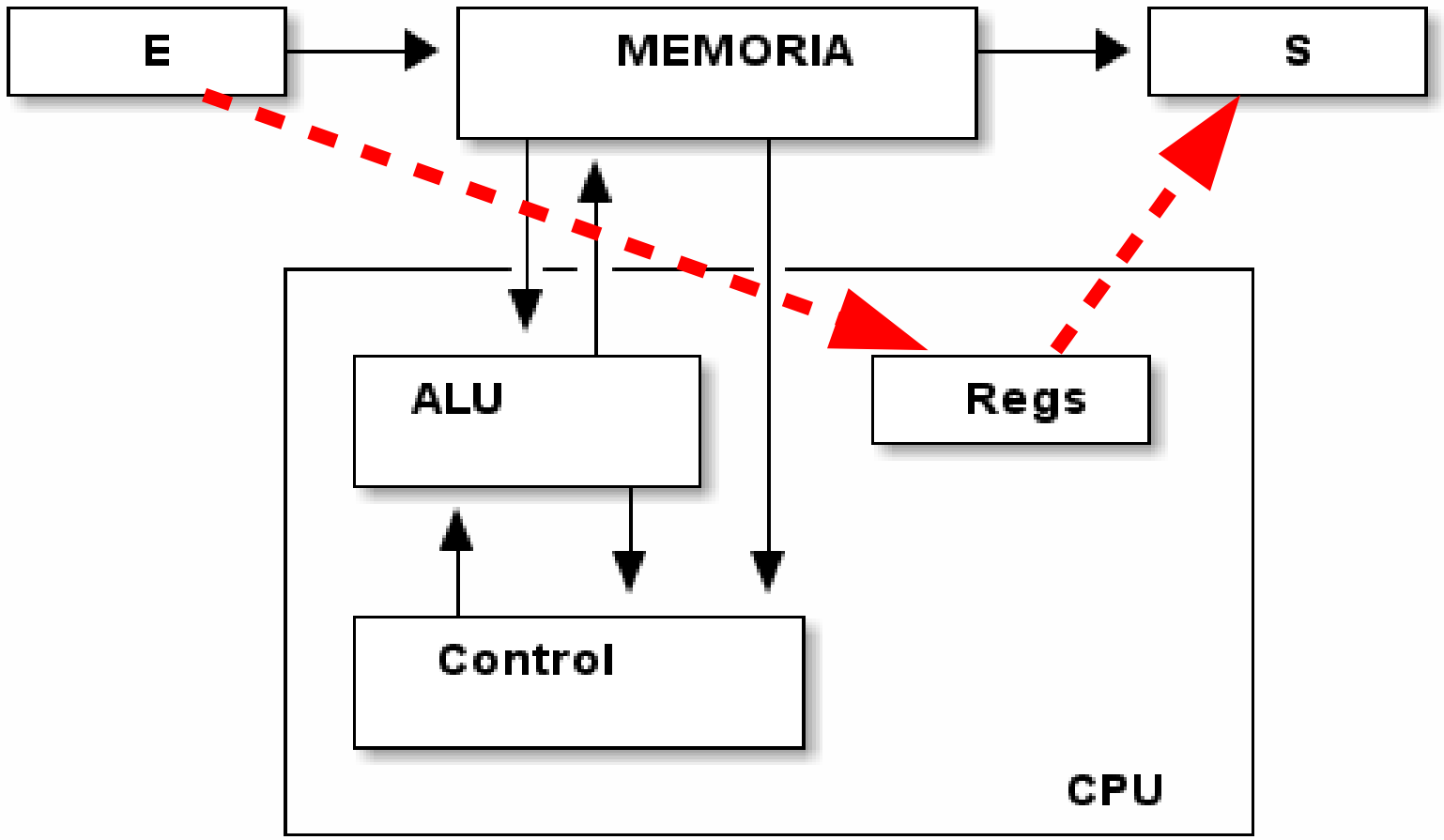
- Dispositivos de entrada
  - Ingresar instrucciones y operandos
- Almacenamiento (memoria)
  - Obtener instrucciones y operandos
  - Depositar resultados
- Unidad aritmético lógica (ALU)
  - realizar operaciones aritméticas y lógicas sobre datos obtenidos de la memoria
- Dispositivos de salida
  - Suministrar resultados
- Unidad de control
  - Generar señales de control para las transferencias entre todos los bloques en el orden y en los tiempos adecuados

Máquina de Von Neumann



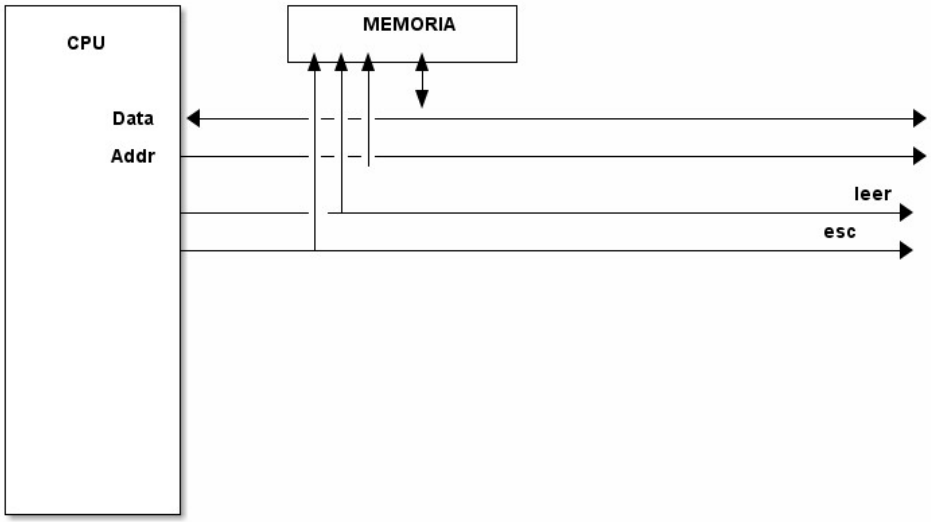
- Registros
  - A menudo se incluye una pequeña memoria dentro de la CPU
  - Mayor velocidad de acceso

Máquina de Von Neumann

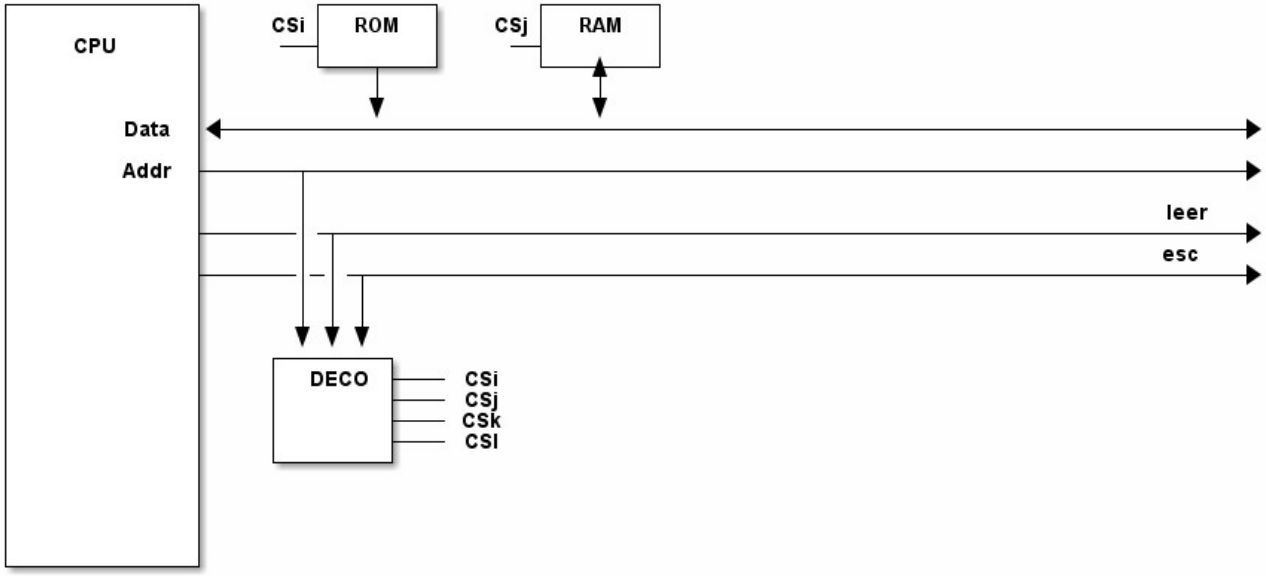


Comunicación memoria-CPU

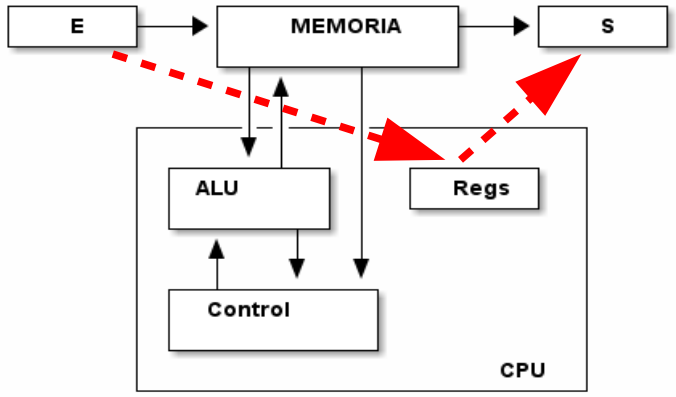
- Memoria: colección numerada de registros
- Tres buses
  - Datos: desde o hacia la CPU
  - Direcciones: indica cuál registro
  - Control: indica qué transferencia quiero hacer



Comunicación memoria-CPU varias memorias



Comunicación memoria-CPU dispositivos E/S



Comunicación memoria-CPU
Acceso directo a memoria