1.- Introducción

Introducción a los microprocesadores
2015

Lógica “cableada” vs Lógica “programada”

Lógica cableada:

- Circuitos vistos en Diseño Lógico (Combinatoria, Modo reloj, RTL,…)
- **Función fija** determinada en el momento del diseño por las conexiones físicas entre los componentes del circuito
- Para cambiar la función es necesario diseñar otro dispositivo.
- En el caso de FPGA, sigue siendo Lógica Cableada aunque la tecnología permite redefinir el conexionado sin necesidad de soldar.

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 cambia el circuito, solo implica cambiar el contenido almacenado en una “memoria”.

Lógica "cableada" vs Lógica "programada"

Compromiso velocidad / flexibilidad / precio

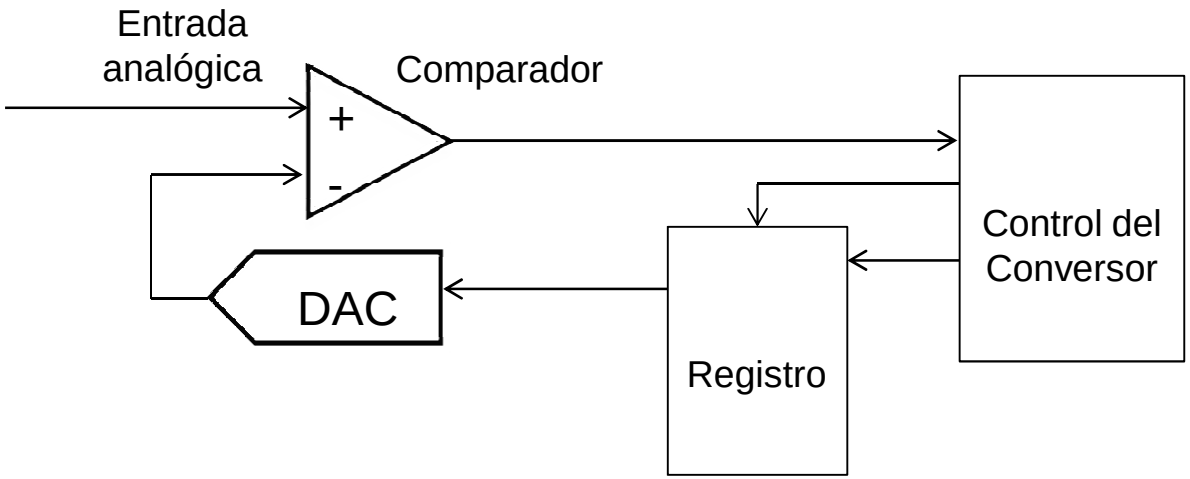
- Un sistema por programa es más flexible para implementar diferentes funciones, (menor costo de diseño) . Basta cambiar el contenido de una memoria.

- ¿por qué diseñar con lógica cableada?
  - **La lógica programada es mucho más lenta (por ej. 100 a 1)**

- ¿Porque 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.

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



-Reloj de 4Mhz

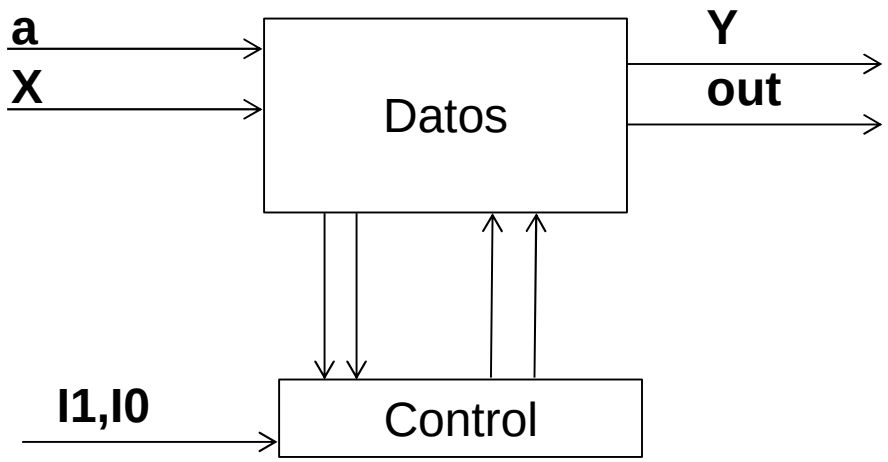
- Se requieren 8 pasadas por el comparador (una por cada bit)

-¿Qué tan rápido se puede muestrear? (fs: frecuencia de muestreo)

- Por cada pasada un computador requiere de varias instrucciones que a su vez cada instrucción demora varios períodos de reloj.

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

Ejemplo: Sistema RTL



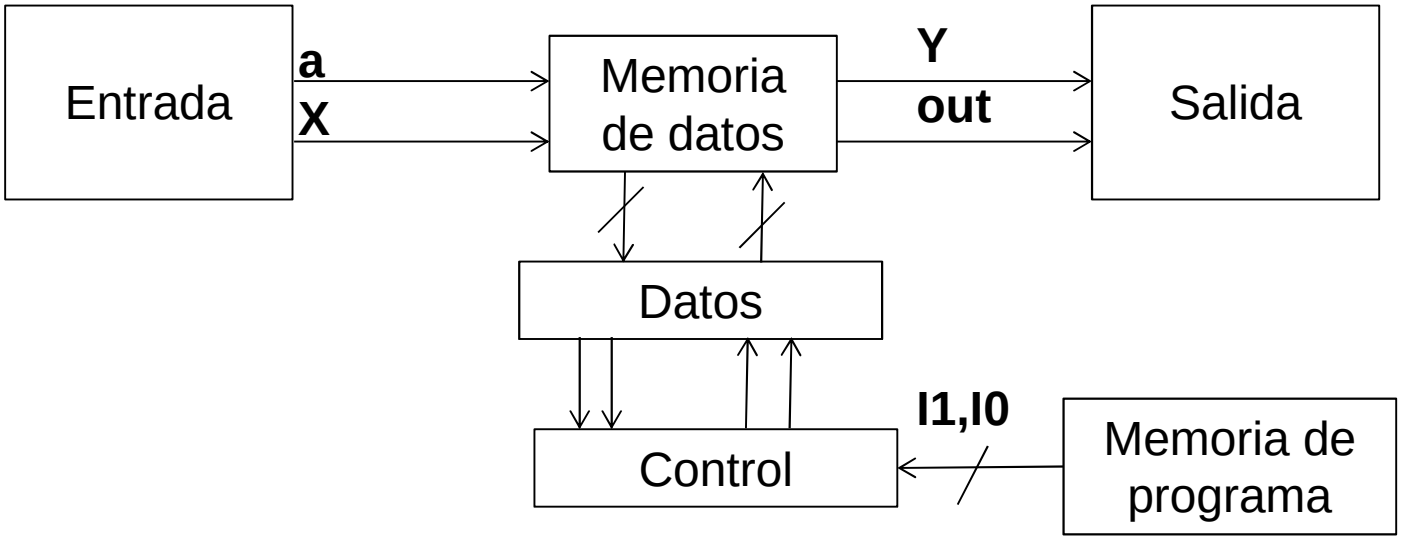
I1,I0	Función
00	Suma
01	Resta
10	AND bit a bit
11	OR bit a bit

Sea el siguiente sistema que procesa datos:

- **a** va a 1 para indicar que el primer dato está pronto en **X**
- Un tiempo después a vuelve a 1 para indicar que el segundo dato está pronto en **X**
- El sistema debe después poner en **Y** el resultado de la operación idicada por **i1,i0** 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.

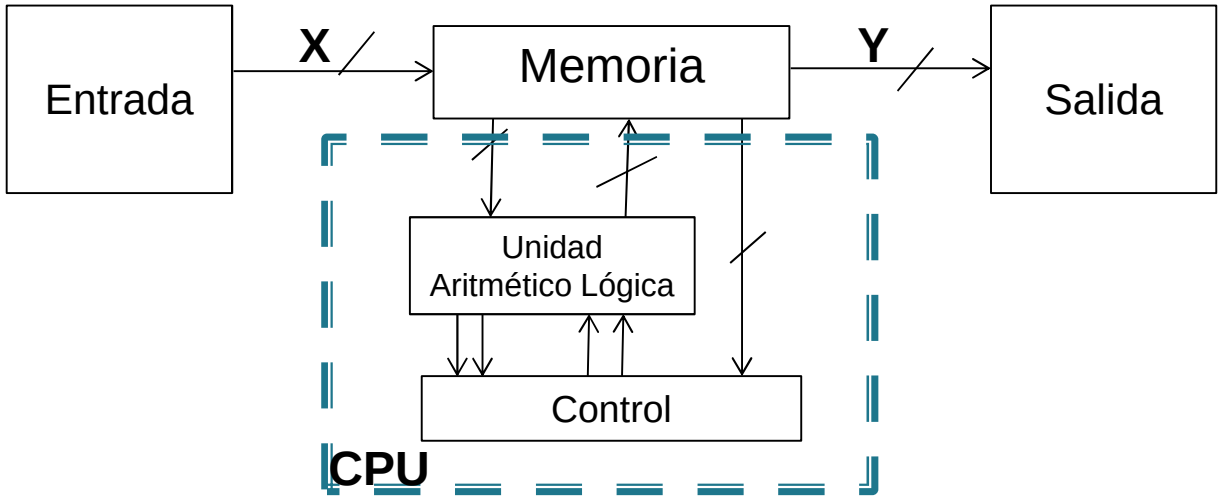
Ejemplo: Sistema RTL



- 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



- Según Von Neumann un computador debe tener:

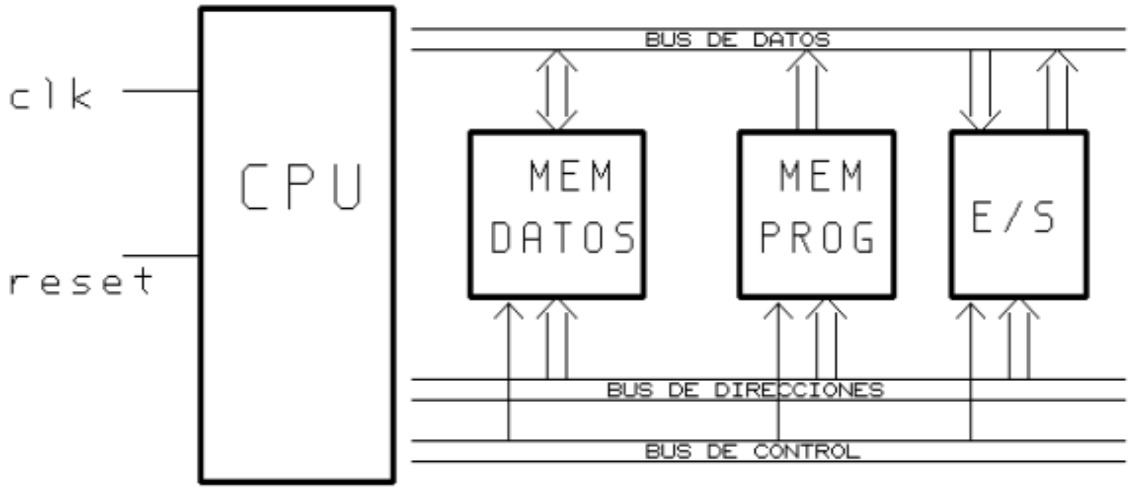
1. *Dispositivo de entrada*: instrucciones y operandos
2. *Medio de almacenamiento* (memoria): instrucciones, operandos, resultados
3. *Unidad Aritmético Lógica* – ALU
4. *Dispositivo de salida*
5. *Unidad de control*: señales de control

ALU + Control = Unidad Central de Procesamiento (CPU)

CPU contenida en un CHIP = Microprocesador

El Computador

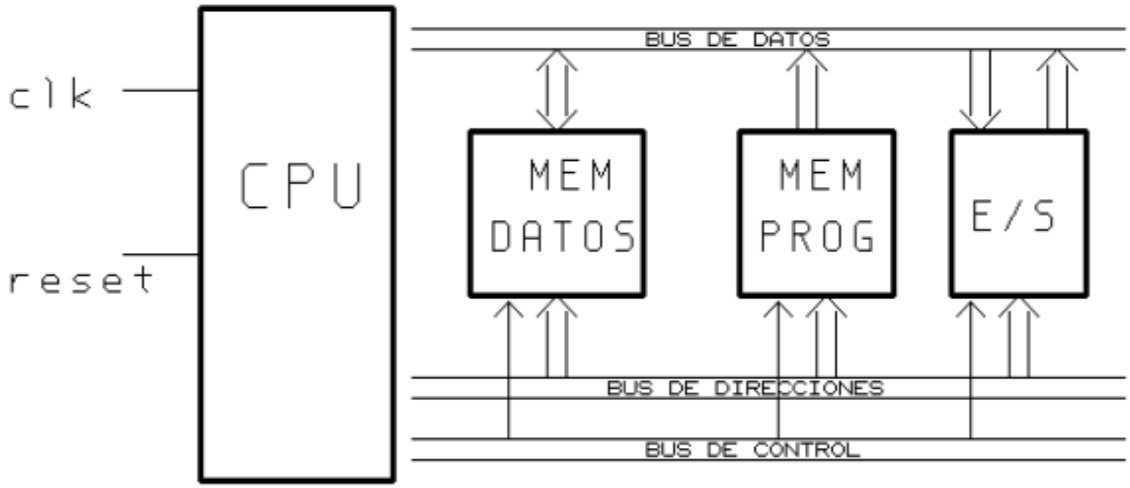
Arquitectura de 3 Buses



- **CPU (Unidad Central de Proceso)**
  - Es el corazón del computador
  - Tareas:
    - Lee las instrucciones en forma ordenada de la memoria de programa
    - Interpreta su significado
    - Ejecuta la operación
    - Guarda el resultado (si corresponde)

El Computador

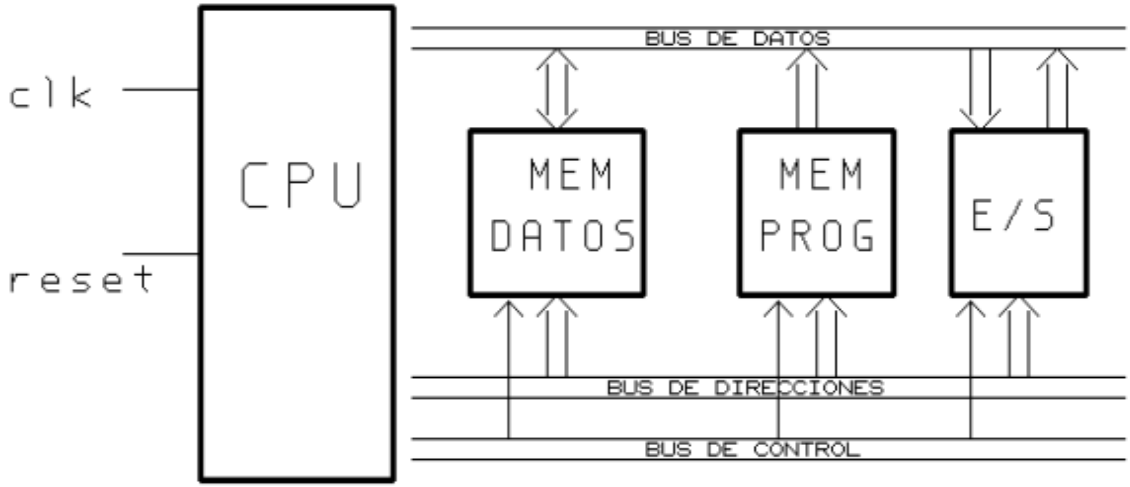
Arquitectura de 3 Buses



- **Reloj**
  - Se trata de un sistema secuencial
  - El reloj es el mismo para todo el sistema.
- **Reset**
  - Fuerza al sistema a ir a un estado inicial conocido.

El Computador

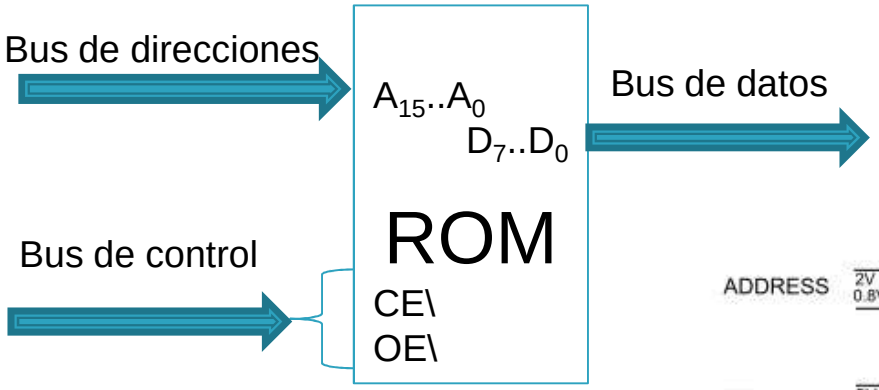
Arquitectura de 3 Buses



- **Bus**: Conjunto de señales (conductores eléctricos) que funcionalmente es una unidad.
- **Bus de direcciones:** Dirección para acceder a “una posición de memoria o E/S”.
- **Bus de datos:** Transporta los datos de un elemento del sistema a otro.
  - **Bus de datos bidireccional** ➔ un único bus de datos triestado
  - **Bus de datos unidireccional** ➔ 2 buses de datos, uno entrada y otro salida.
- **Bus de control:** Señales para controlar el funcionamiento del sistema

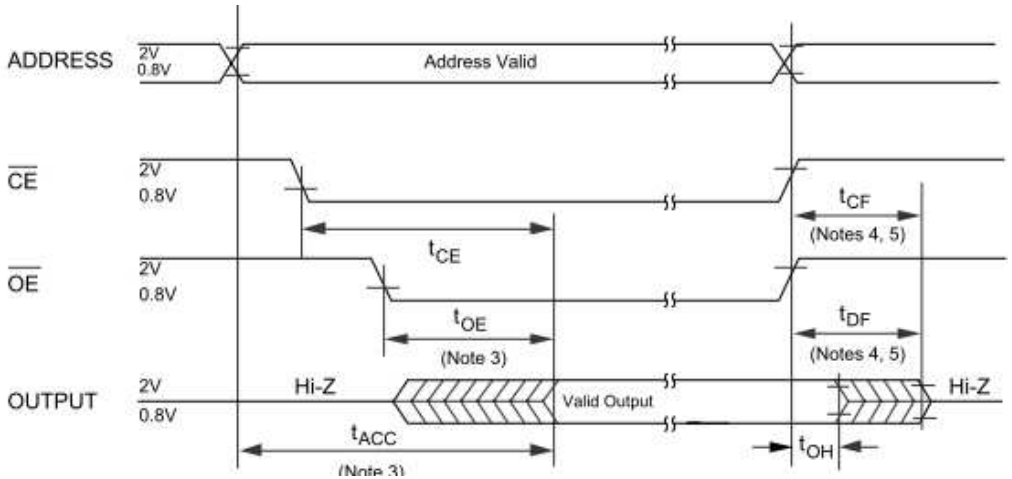
Repaso – Memoria ROM

- ROM: (Read Only Memory) Memoria de solo lectura (o lectura preferente).
- Si se interrumpe la energía se conservan los datos.
- **Contiene el primer programa que se ejecuta** al inicializar el sistema.
- En general es la memoria de programa.



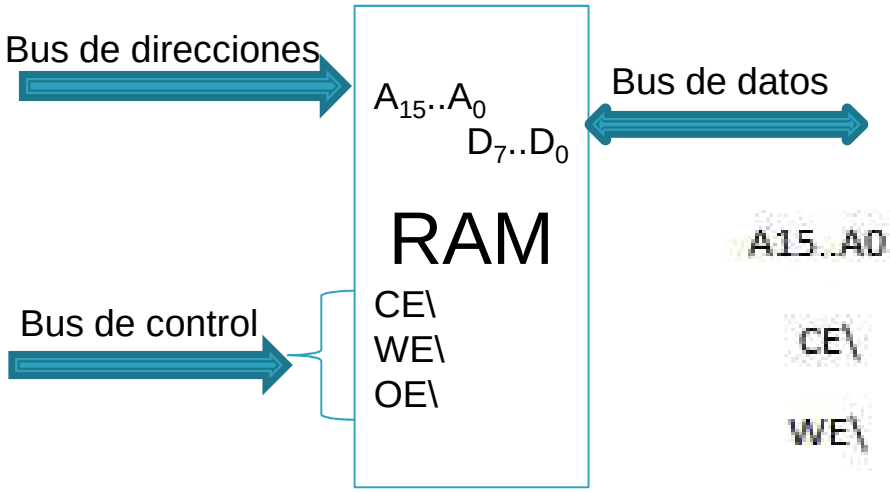
CE\ :Chip enable => Enciende (consumo)

OE\: Output enable => Habilita salidas



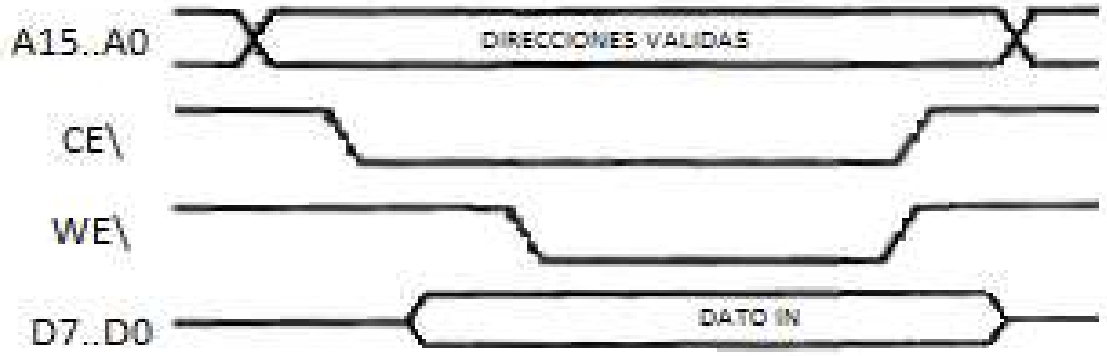
Repaso – Memoria RAM

- RAM: Memoria de lectura / escritura.
- Si se interrumpe la energía se pierden los datos.
- **Se utiliza para guardar variables, datos temporales del programa y también programas temporarios.**.
- En general se asocia a la memoria de datos.



CE\ :Chip enable => Enciende (consumo)

OE\: Output enable =>. Habilita salidas



Ciclo escritura

Nota: El ciclo de lectura es igual al de la ROM

El Microprocesador (uP)

- **Microprocesador =** CPU en un chip.
- **Especificación desde el punto de vista de usuario:**
  - Ancho de palabra: Tamaño del bus de datos, registros internos, ALU
  - Capacidad de direccionamiento :
    - o Tamaño de “memoria” que puede direccionar.
    - o Si bus de **n bits** ➔ capacidad de direccionamiento de **2^n palabras**.
    - o El direccionamiento de E/S puede estar separado o junto con el de memoria.
- **Otros datos que especifican a un uP:**
  - Registros internos: es la memoria interna de un uP.
  - Repertorio de instrucciones: El conjunto de instrucciones que es capaz de ejecutar.
  - Velocidad del reloj: Ej 4MHz, 400 MHz, …..
  - ……..