In Furfi we trust

Final de Orga II resumen

fuente: cubawiki

Versión en LaTeX:
Camila Gallo

Índice

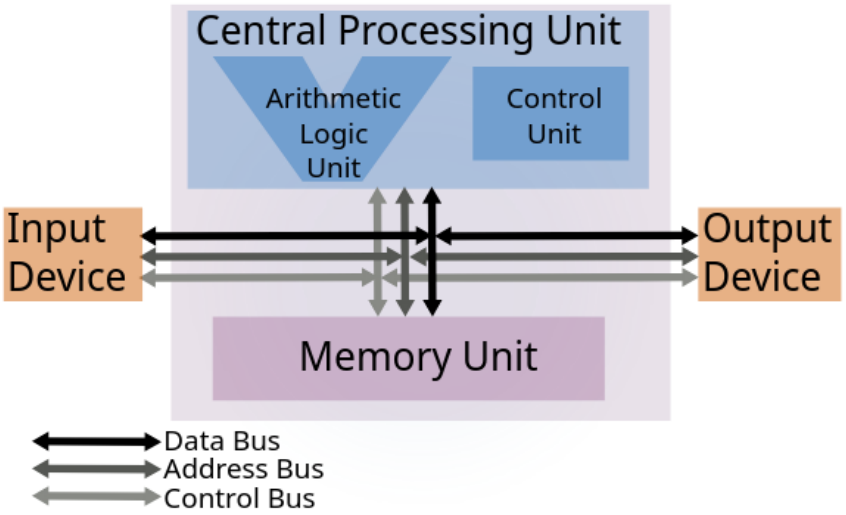
1. Un poco de historia

1.1. Máquina de Turing

En 1936, Turing publica su paper en donde define lo que es una máquina de Turing. Esto es un modelo matemático basado en autómatas que "traduce"las fórmulas matemáticas en programas. Es la primera vez, que se define un modelo de computación y sienta las bases de lo que hoy entendemos como computación.

1.2. Modelo de Von Neumann

Basado en el modelo teórico de Turing, 9 años después, publica un paper en donde define en detalle el modelo de cómputo desarrollado por Turing años antes, y propone este modelo de computadora.



Modelo simplificado de Von Newmann: Tres unidades unidas por tres buses.

Figura 1: Modelo simplificado

Este modelo fue el predominante dentro de la 1era y 2da generación de computadoras. Los componentes más importantes eran:

- Central aritmética (CA): Una unidad de procesamiento aritmético, de lógica y registros.
- Central de Control (CC): Una implementación de la máquina de estados, con registro de instrucciones y registro Contador de Programa.

- Memoria: Para guardar los datos y el código que se ejecutaban en ese momento. No había espacio para otro programa. Ambos se almacenaban en el mismo lugar, esto daría problemas a un cuello de botella.
- I/O: Mecanismos tanto de entrada como de salida para que se pudieron ingresar datos y comandos a la computadora y para que esta puediera enviar resultados al exterior.
- Memoria Externa: Era otro medio de almacenamiento con mayor capacidad que la Unidad de Memoria. Se guardaban otros programas y datos que luego se copiaban a la Unidad de Memoria. Por ejemplo, cintas perforadas.

Tenemos entonces un modelo en donde hay un programa almacenado, los datos y el programa están en el mismo y único banco de memoria. El cpu es una máquina de estados perpétua.

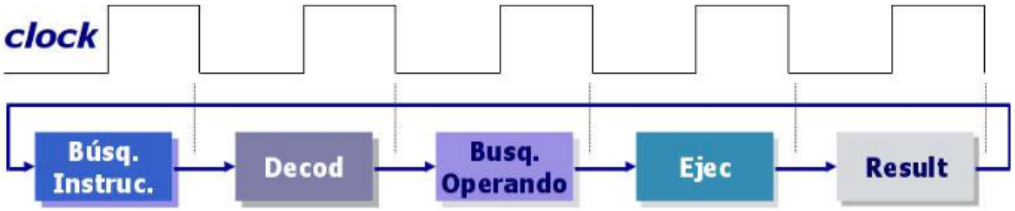


Figura 2: Máquina de estados de la máquina de Von Neumann

1.3. Modelo de Hardvard

Con el pasar de los años, en menos de dos décadas, se necesitó leer **al mismo tiempo**(o al menos que se comporte como así lo fuese) tanto los datos de la operación a realizar como sus operandos. Entonces, claramente había un problema con el modelo de Von Neumann porque al haber un único banco de memoria con buses compartidos, no se podía hacer ambas cosas .ª la vez". Esto se conoce como el cuello de botella de Von Neumann.

Para solucionar el problema del cuello de botella, revisaron la arquitectura de la computadora Mark I, un computador electromecánico desarrollado por IBM y la universidad de Harvard entre 1939 y 1944. Es decir, era anterior a la ENDVAC de Von Neumann.

La Mark I tenía datapaths diferentes para los datos que ingresaban y para las instrucciones que se ejecutaban. Por eso era interesante a pesar de ser un computador en decimal y que tarda entre 0.3 y 10 segundos para cada cálculo.

1.4. Computadoras de 2da Generación

Son computadoras con transistores. Aparecen en la década del 50. Las ventajas que tienen es que tienen menor consumo, menor espacio y mejor desempeño a largo plazo.

También aparecen los lenguajes de alto nivel. En 1957 se escribe el primer compilador de FORTRAN.

Se consolida IBM, nacen XEROX y DEC.

1.5. 3era generación

Consiste de las computadores de circuitos integrados (CI). Es decir, chips. Aparecen en la época de los 60.

Se produjeron los minicomputadores y nace la programación estructurada. Se libera el primer compilador de C y nace UNIX.

Nota:	En 1971, Intel presenta el CI 4004, el primer microprocesaor de la historia.

1.6. Problemas a resolver

- Los usuarios querían disminuir el costo de actualización de los equipos.
- Los equipos eran incompatibles con otros modelos de la misma marca. Cambiaba la ISA, las toolchains, los SO, etc. Cada mercado era distinto.
- Cada actualización te hacía cambiar todo.
- Empieza a hacer más ruido la palabra compatibilidad. Al menos para IBM.

2. La era del silicio

Hay dos aspectos importantes cuando se diseña un computador. El diseño del:

- El Datapath. Es el conjunto de recursos que se utilizan para que los números se almacenen en registros y las operacion entre ellos se puedan ejecutar.

- El Control. Es la lógica necesaria para controlar el correcto envío por el datapath de las operaciones que se van a realizar.

El control es el item más complejo. Y en IBM, a Maurice Wilkes, se le ocurrió el concepto de **microcódigo**, una lógica de control para un microprocesador. En ese momento como la ROM era más barata, se propuso implementar el microcódigo en una memoria de estas y cada palabra era una microinstrucción(pequeños programas). A mitad de la década del 60, IBM anuncia que sus modelos de mainframes IBM 360 se basarán en microcódigo.

Este control por microcódigo dio a luz al diseño de una ISA con instrucciones de complejidad creciente. La complejidad se cargaba en el microcódigo en la CPU y ocupaba la menor memoria posible porque no se podía minimizar el tamaño de los datos, era lo que era.

Nota: Para el caso del código, menos instrucciones implica menos memoria.

Acá elijo no anotar varias cosas.

2.1. Nuevo paradigma de arquitectura: RISC

Antes de las computadoras personales, lo que predominaban era los mainframes y quienes dominaban el mercado eran IBM, DEC, Amdahl, entre otros.

En los 80, Intel lanza los primeros microprocesadores de 16 bits y arrasa con todo. No olvidemos que el se de instrucciones tiene una complejidad creciente con cada lanzamiento.

Durante la época de los años 60 y 70, IBM, Control Data Corporation y Data General, lanzan lines de investigación para el diseño de chips con pocas pocas instrucciones simples. En los años 80, en la universidad de Berkley y en la Stanford, se publican los primeros papers donde se presenta una arquitectura antagonista a la que dominaba en ese momento llamada RISC. Por sus siglas en inglés: Reduced Instruction Set Computer.

Nota: Lo que motiva a llamar a los procesadores de ese entonces como CISC: Complex Instruction Set Computer.

En RISC lo que importa es lo siguiente

- Hay 32 registros de propósito general(O sea, una banda).

- Las instrucciones se ejecutan en un sólo ciclo de clocl.
- Las instrucciones derivan en códigos de operación de igual tamaño y formato.
- Las instrucciones deben ser sencillas de decodificar.
- No se utiliza microcodigo. O sea, no existen instrucciones complejas. Ni mul ni div.
- Los datos en memoria se accenden mediante LOAD y STORE.

Nota:	Por estas razones, verificar una máquina RISC es más simple y es menor la probabilidad de tener un bug en el procesador.

Nota:	Es esperable tener programas de mayor tamaño.

2.2. Resumen del capítulo

Arquitectura Harvard

- Las instrucciones y los datos se leen desde memorias físicamente diferentes y por caminos de señal diferentes.
- Se pueden tener tecnologías y organizaciones diferentes para la memoria usada para instrucciones y para datos, además de caminos de señal (buses) independientes.
- Si tenés buses diferentes, tenés dos direcciones `0x0000_0000` de memoria.
- La memoria de instrucciones puede estar organizada en palabras de 4 bytes (32 bits) y la de datos en 1 byte (8 bits), sin que nada se rompa.

Arquitectura Von Neumann

- Las instrucciones y los datos se guardan y leen de la misma unidad de memoria.
- Comparten la misma organización y tecnología.

- No hay separación entre memoria de instrucciones y memoria de datos.

Los microprocesadores actuales presentan una organización siguiendo el modelo de Von Neumann a los usuarios. Pero, desde los 90 que se usa una tecnología **split cache**. Se dividen los primeros niveles de memoria cache en una de datos y otra de instrucciones con dos buses independientes para accederlas en paralelo. Es decir, un modelo Hardvard. Sin embargo, los niveles de memoria más externos almacenan datos e instrucciones juntos, o sea, Von Neumann. El subsistema de memoria cache está diseñado y dimensionado para que se resuelvan en el él la mayoría de los accesos de memoria de parte de las CPUs. Un valor típico de eficiencia de un cache de Nivel 1 es 80 %. O sea, se resuelve en una memoria organizada según el modelo de Hardvard.

Nota:	El nivel 1 de cache no es estrictamente Hardvard porque quien programa ve un solo espacio de direccionamiento en lugar de dos. Esta división la realiza el sistema Cache.

Nota:	Sólo los procesador de señales (DSP) son puramente Hardvard. Los procesadores de propósito general actuales y algún que otro microcontrolador CORTEX-M trabajan con split cache en nivel 1 de su jerarquía de memoria y después juntan código y datos en los niveles restantes.

Nota:	Como todos presentan un mapa de direcciones de memoria único al programador, se denominan çasi Von Neumann".

3. Arquitecura vs. Microarquitectura