Memoria y Buses

Organización de Computadoras 2019

Universidad Nacional de Quilmes

1. Memoria Principal e instrucciones

Para comprender la ejecución de los programas, es necesario considerar los requisitos que tiene que cumplir el procesador (CPU):

1. Buscar instrucciones desde la memoria o desde un dispositivo de entrada y salida
2. Decodificar instrucciones para determinar que acción es necesaria
3. Buscar los datos que la ejecución de la instrucción puede requerir
4. Procesar los datos que la ejecución puede necesitar a través de una operación lógica o aritmética
5. Almacenar los resultados donde corresponda

Lo anterior se describía en apuntes anteriores mediante la figura 1

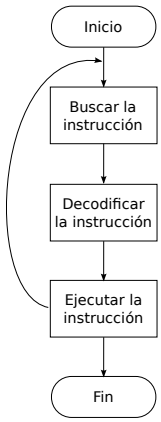


Figura 1: Ciclo de ejecución de instrucciones

En este apunte se describirá la mencionada memoria principal como almacenamiento de las instrucciones pero también de datos u operandos que participan en ellas (pasos 3 y 5 mencionados arriba).

Motivación

La memoria principal es un conjunto homogéneo de **celdas**, esto es: todas de la misma capacidad (es decir, el mismo tamaño en cantidad de bits). Cada celda tiene una **dirección** que la identifica y que le permite ser leída o escrita, es decir que es la *unidad direccionable más pequeña*.

A la memoria principal se la conoce también como memoria RAM (Memoria de acceso aleatorio). Se denominan de acceso aleatorio porque el acceso a una celda tiene un costo en tiempo igual a cualquier otra. Mas detalle sobre cómo esto es posible se da en la sección 2.1.

En la figura 2 se ilustra una memoria con 4 celdas, cuyas direcciones son 00,10,10 y 11.

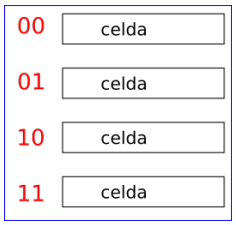


Figura 2: memoria de 4 celdas

Las celdas se agrupan en palabras, porque en algunas arquitecturas esto tiene sentido. En general, el tamaño de palabra coincide con el tamaño de los datos. En nuestra arquitectura, 1 palabra=1 celda pues las variables son de 16 bits (registros y celdas de memoria) y las instrucciones

aritméticas manejan esa cantidad de bits al mismo tiempo.

1.1. Operaciones sobre la memoria

La memoria admite dos operaciones: **lectura** y **escritura**. Para resolver la lectura, la memoria necesita conocer una dirección y recibir una señal de lectura. Luego de esto, pone a disposición de la Unidad de Control (UC) el contenido de la celda correspondiente y activa otra señal de control para indicar su finalización.

Para realizar una escritura, la UC le envía a la memoria una señal de escritura y la dirección de una celda, pero además provee el dato a escribir. A partir de esto, actualiza el contenido de la celda correspondiente con el dato recibido y activa una señal de control para indicar su finalización.

1.2. Interconexión

Como se mencionó, la memoria principal y la unidad de control deben comunicarse para intercambiar datos y direcciones. Estos circuitos se comunican a través de un medio de transmisión compartido entre la CPU y la Memoria Principal que se denomina **Bus del Sistema**.

La información que se necesita transmitir incluye datos desde y hacia la memoria, así como direcciones hacia la memoria y señales de control. Es por esto que el bus de sistema está formado por tres buses para cumplir los diferentes objetivos y que reciben los nombres de **Bus de direcciones**, **Bus de datos** y **Bus de control** (ver figura 3).

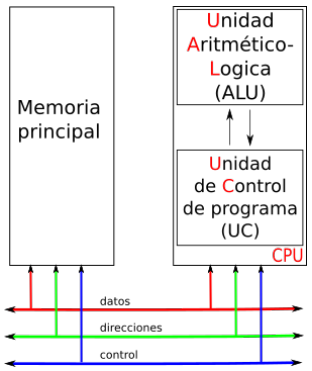


Figura 3: Buses del sistema

Cada bus tiene una determinada cantidad de líneas -esto se denomina ancho del bus- y cada línea transmite un bit a la vez. Entonces el **ancho del bus** determina cuántos bits se pueden transmitir en paralelo.

En el caso del bus de direcciones, la cantidad de líneas determina el conjunto de direcciones de las celdas de memoria, denominado **espacio direccionable**. Por ejemplo, si se cuenta con **m** bits para describir direcciones de la memoria, la cantidad máxima de combinaciones, y por lo tanto de celdas a direccionar, es 2^m.

Por otro lado, el ancho del bus de datos determina el tamaño de las palabras.

1.3. Modos de direccionamiento

Como se presentó en la introducción, las celdas de memoria pueden utilizarse como variables que el programador dispone desde las instrucciones, y entonces se cuenta con 3 mecanismos para hacer referencia a los operandos, denominados modos de direccionamiento.

Direccionamiento Inmediato

Es la forma más sencilla de direccionamiento, el operando está presente en la propia instrucción. Este modo puede utilizarse para definir y utilizar constantes o para fijar valores iniciales (inicializar) de variables.

Este mecanismo es necesario pues todo lenguaje de programación requiere la posibilidad de denotar constantes. En la sección 1.4 se verá porqué el tamaño del número está restringido a la longitud del campo de direcciones.

Direccionamiento Registro

Es similar al direccionamiento Directo, con la diferencia que el operando ahora se encuentra en un registro del sistema en lugar de estar en memoria principal. La ventaja principal de éste modo es que solo es necesario un pequeño campo de direcciones en la instrucción y no se requieren referencias a memoria. Además, el tiempo de acceso a un registro interno de CPU es menor al tiempo de acceso a memoria principal. La desventaja principal es que el espacio de direccionamiento está limitado, pues

se espera que la cantidad de registros visibles al programador no sea muy grande.

Direccionamiento Directo

Una forma también sencilla es el direccionamiento directo, donde el campo de direcciones contiene la dirección efectiva del operando. Ésta técnica fue común en las primeras generaciones de computadoras y se encuentra aún en diversos sistemas. Solo requiere una referencia a memoria y no necesita ningún cálculo especial. La limitación mas evidente es que proporciona un espacio de direcciones restringido.

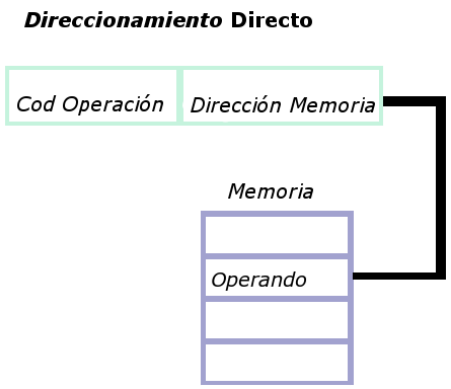


Figura 4:

1.4. Arquitectura Q2

La arquitectura Q2 agrega a la arquitectura Q1 un nuevo modo de direccionamiento: **Directo**, el cual especifica la direccion de memoria donde se encuentra el valor del operando.

Para usar este nuevo modo de direccionamiento la dirección del operando se escribe entre corchetes, por ejemplo: `SUB [0x000A], 0x0001`. El efecto de esta instrucción es el de decrementar en 1 el valor que tenía la celda cuya dirección es `000A`.

Si antes de ejecutar teníamos en memoria:

	...
0x000A	29C8
0x000B	A0A0
	...

Luego de ejecutar se tendrá:

	...
0x000A	29C7
0x000B	A0A0
	...

Ejemplos de otras instrucciones:

- `MOV R1,[0x000A]`
- `ADD [0x000A], 0x000A`
- `SUB [0x000A], [0x000B]`

2. Funcionamiento a nivel de circuitos

Para llevar a cabo estas tareas es claro que el procesador debe almacenar algunos datos temporalmente: debe recordar la posición de la última instrucción de forma de poder determinar de dónde tomar la siguiente, y necesita almacenar instrucciones y datos temporalmente mientras una instrucción está ejecutándose. En otras palabras, el procesador necesita una pequeña memoria interna.

Dentro del procesador hay un conjunto de registros clasificados en dos tipos:

- Registros visibles al programador: permiten al programador de lenguaje máquina o ensamblador minimizar las referencias a memoria principal por medio de la optimización de uso de registros.
- Registros de control y estado: Son utilizados por la unidad de control para controlar el funcionamiento del procesador y por programas privilegiados del sistema operativo para controlar la ejecución de programas.

Hay diversos registros del procesador que se emplean para controlar su funcionamiento. La mayoría de ellos, en la mayor parte de las máquinas no son visibles por el usuario, pero alguno de ellos puede ser visible por ciertas instrucciones máquina ejecutadas en un modo privilegiado o de sistema operativo.

Naturalmente, diferentes máquinas tendrán distinta organización de registros y usarán distinta terminología, pero esencialmente se necesitan cuatro registros para la ejecución de una instrucción:

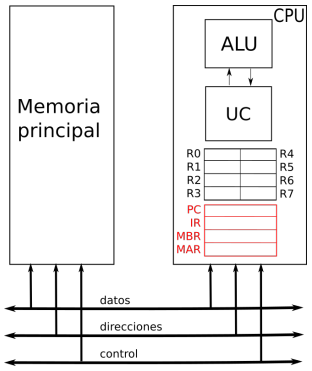


Figura 5: Registros de control y estado

- **Contador de programa (Program Counter- PC)**: Contiene la dirección de la instrucción a buscar
- **Registro de instrucción (Instruction Register- IR)**: Contiene la instrucción buscada mas recientemente. Muchas veces este registro tiene mas capacidad que los demás, a los fines de *almacenar la instrucción completa.*
- **Registro de dirección de memoria (Memory Address Register- MAR)** Contiene la dirección de una posición de memoria
- **Registro amortiguador de memoria (Memory Buffer Register - MBR)** Contiene el dato a escribir en una posición de memoria o el dato contenido en una posición de memoria leído mas recientemente

No todos los procesadores tienen registros internos designados como MAR o MBR pero se necesita algún mecanismo de almacenamiento intermedio equivalente mediante el cual se de salida a los bits que van a ser transferidos por el bus de sistema y se almacenen los bits leídos por el bus de datos. En la figura 5 se ven los registros de control de la arquitectura Q2.

Muchos diseños de procesadores incluyen un registro o un conjunto de registros, conocidos a menudo como palabra del estado de programa o *program status word* (PSW), que contiene información de estado. Típicamente contiene códigos de condición, incluyendo a menudo los siguientes bits de estado:

- El signo del resultado de la última operación
- Indicador de resultado cero o nulo
- Indicador de acarreo en la última operación (suma o resta)
- Indicador de igualdad entre operadores
- Indicador de desbordamiento aritmético
- Habilitación de interrupciones
- Indicador de modo supervisor (para permitir ciertas instrucciones privilegiadas)

En algunos diseños es posible encontrar otros registros relativos a estado y control. Puede existir un puntero a bloque de memoria que contenga información de estado adicional (por ejemplo, bloques de control de procesos). En las máquinas que utilizan interrupciones vectorizadas puede existir un registro de vector de interrupciones. Si se utiliza una pila para llevar a cabo ciertas funciones, se necesita un puntero a pila del sistema. En un sistema de memoria virtual se utiliza un puntero a una tabla de paginas. por último, pueden utilizarse registros para el control de operaciones de E/S.

2.1. Funcionamiento de la memoria principal

Como se dijo en la sección 1.1, la memoria principal es un circuito formado por **celdas** que se acceden a través de dos operaciones: lectura y escritura. En esta sección se revisarán dichas operaciones a partir de los registros y los buses mencionados arriba.

La operación de **lectura** ocurre cuando la Unidad de Control (UC) pide un dato almacenado en una determinada celda de la memoria principal. Para esto, la UC pone la dirección en el registro **MAR** y luego activa una **señal de lectura** en el **bus de control**. Luego, la Memoria Principal activa un circuito decodificador (ver figura 7) que permite elegir la celda específica y copiar su contenido en el **bus de datos**. Finalmente, el dato se encontrará disponible en el registro **MBR** de la CPU y mediante el bus de control activará una señal que indica que el dato está disponible

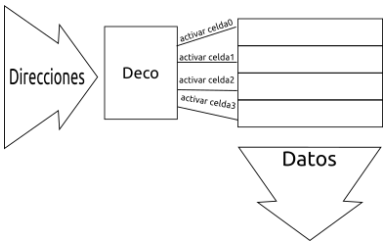


Figura 6: Memoria conectada a través de los buses

La operación de **escritura** ocurre cuando la UC necesita almacenar un dato en una determinada celda de memoria. Para hacerlo, la UC pone el dato en el registro **MBR** y una dirección en el registro **MAR**. Luego activa una **señal de escritura** en el bus de control para indicar la operación de escritura. También en este caso, la Memoria activa un circuito decodificador que permite elegir la celda específica, pero además toma el dato del **bus de datos** y lo copia en dicha celda. Finalmente, mediante el bus de control se indica a la UC que la operación ha finalizado.

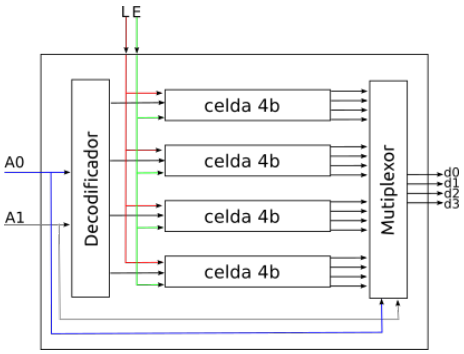


Figura 7: Circuito de una memoria de 4 celdas

2.2. Ciclo de ejecución de instrucción

En esta sección se revisará el ciclo de ejecución de instrucciones que recorre el procesador durante la ejecución de un programa, detallando los dispositivos digitales que entran en juego.

Típicamente, el procesador actualiza el **registro PC** luego de cada búsqueda de instrucción, de manera que siempre quede preparado para la siguiente instrucción a ejecutar. La instrucción buscada se carga en **registro IR**, donde son analizados las distintas partes de la instrucción, en particular el código de operación y los **modos de direccionamiento** de los operandos. A continuación se intercambian datos con la memoria por medio del MAR y el MBR, tal como se describió en la sección anterior.

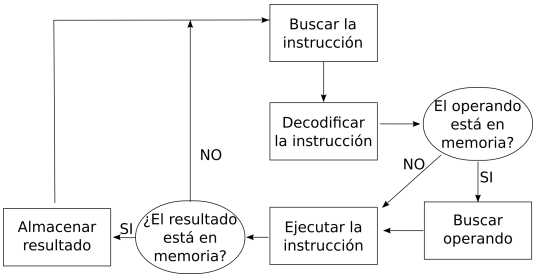


Figura 8: Ciclo de ejecución de instrucción

Los registros mencionados se usan para la transferencia de datos entre el procesador y la memoria principal. Dentro del procesador, los datos tienen que ofrecersele a la ALU para su procesamiento. La ALU puede tener acceso directo al MBR y a los registros visibles por el programador, pero alternativamente puede haber registros intermedios en torno a la ALU, que le sirven como entrada y salida e intercambian datos con el MBR y los registros visibles al programador.

Luego de la ejecución de la instrucción, y en función del modo de direccionamiento indicado en la instrucción para el **operando destino**, se almacena el resultado. Esto puede resultar en una escritura a memoria (si el modo correspondiente es directo)

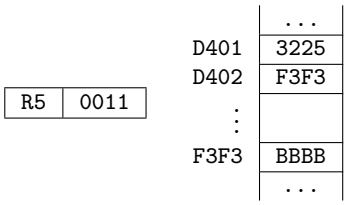
3. Ejecución de programas en la arquitectura Q

En esta sección, se desarrollará el ciclo de ejecución de una instrucción, poniendo atención a las etapas y a los accesos a la memoria.

La ejecución de un programa no es ni más ni menos que la ejecución en secuencia de sus

instrucciones. Para llevar cuenta de la instrucción actual, o mejor dicho, la siguiente a ejecutar, se utiliza el registro *Program Counter*.

Consideremos como ejemplo la instrucción que está ensamblada en la celda `0xD401`. Por lo tanto `PC=D401`. El estado inicial de las celdas de memoria y el registro R5 es como sigue:



1. **Búsqueda de la instrucción**: Se lee la celda indicadas por el PC y se almacenan en **IR**. La lectura a memoria se describirá en término del estado de los buses, en el siguiente cuadro:

Bus de ctrl	Bus de dir.	Bus de datos
L=1	D401	3225

   Nota: en el bus de control se debe indicar L=1 cuando es una lectura o E=1 cuando es una escritura

   Por lo tanto `IR=3225`. Además, se actualiza el PC para que quede preparado para la siguiente lectura `PC=D402`

2. **Decodificación de la instrucción**: A partir de la interpretación de la cadena en **IR**, la Unidad de Control entiende lo siguiente:

codop(4b)	M.D.(6b)	M.O.(6b)
0011	001000	100101

   Dado que el modo de direccionamiento del operando destino es **directo**, la UC entiende que hace falta la lectura de otra celda

3. **Completar la búsqueda de la instrucción**: Se lee la celda indicada por el PC (`D402`)y se almacena también en **IR**. Por lo tanto `IR=3225F3F3`

Bus de ctrl	Bus de dir.	Bus de datos
L=1	D402	F3F3

4. **Completar la decodificación de la instrucción**: Con esta nueva lectura puede concluirse que:

codop	M.D.	M.O.	Dest(16b)
0011	001000	100101	1111001111110011

   A los fines de la comprensión de nuestro ejemplo, es importante notar que dicho código máquina corresponde a la instrucción
   `SUB [0xF3F3], R5`

5. **Búsqueda de operandos**: La Unidad de control solicita una lectura de la celda `F3F3` para obtener el operando destino. Para esto realiza los siguientes pasos:

   a) Copia la dirección `F3F3` del registro IR al registro **MAR**

   b) Indica a la Memoria Principal una lectura a través de las lineas correspondientes en el **Bus de control**.

   c) Cuando la Memoria Principal indica que el dato está disponible (también a través del bus de control), la UC tiene disponible el dato en el registro **MBR**.

Bus de ctrl	Bus de dir.	Bus de datos
L=1	F3F3	BBBB

6. **Ejecución de la instrucción**: La UC le indica a la **ALU** la operación `SUB` y le suministra los dos operandos:

   a) Operando destino: desde el registro **MBR** (`BBBB`)

   b) Operando origen: desde el registro **R5** (`0011`)

7. **Almacenamiento de resultado**: La Unidad de Control solicita una **escritura** en Memoria Principal para almacenar el resultado en el operando destino. Para esto realiza los pasos:

   a) Copia la dirección `F3F3` del registro IR al registro **MAR**

   b) Copia el resultado arrojado por la ALU en el regitro **MBR**

   c) Indica a la Memoria Principal una **escritura** a través de las lineas correspondientes en el **Bus de control**.

Bus de ctrl	Bus de dir.	Bus de datos
E=1	F3F3	BBAA

8. Comienza un nuevo ciclo con la siguiente instrucción

Fuente

Extraído de *Organización y Arquitectura de Computadores*, William Stallings - 7ta edición. Capítulo 11.