SISTEMAS DIGITALES

LOGRO DE LA UNIDAD DE APRENDIZAJE

- Al término de la unidad, los alumnos, diseñarán e implementarán sistemas digitales básicos mediante el uso de simuladores que permite describir el funcionamiento interno de los circuitos digitales usados en la computadora.

- Al término de la unidad, los alumnos, describirán el funcionamiento de sistemas digitales básicos usados en la computadora, haciendo uso de sistemas numéricos, voltajes y tiempos.

TEMARIO

- Filp – Flop, Registro y Memoria
- CPU – Unidad Central de Proceso
- Buses de datos, direcciones y control

ACTIVIDADES PROPUESTAS

- Los alumnos comprueban que a través de un conjunto de compuertas se puede formar un dispositivo que almacene información, como punto de partida para las memorias RAM y ROM.

- Los alumnos analizan el comportamiento del CPU al ejecutar las instrucciones de la memoria.

1. FLIP-FLOP, REGISTRO Y MEMORIA

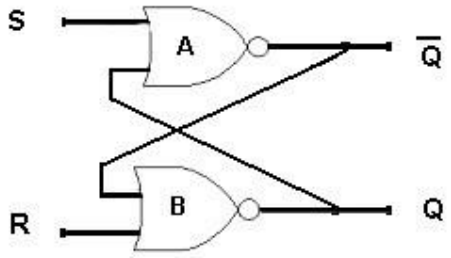
Uno de los sistemas digitales de gran importancia que se han creado es el Flip Flop o FF, el cual fue creado de manera diferente a los otros sistemas, aquí se estableció una conexión extraña entre dos compuertas y luego se determinó su función hallando su tabla de verdad. Al describir su funcionamiento mediante la tabla se encontró un estado especial en el cual la salida no se alteraba, seguía manteniendo su valor anterior. Esto sirvió para que nazca la primera celda que permite guardar un bit, un cero o un uno.

Un flip-flop es un circuito digital, desarrollado con compuertas digitales, que permite mantener un estado binario indefinidamente, siempre y cuando se le esté suministrando energía al circuito. Podemos guardar un "cero" o un "uno". Cambiando las condiciones de la entrada del flip-flop, podemos cambiar su salida.

El primer FF es el FF R-S, como veremos, éste tiene un estado prohibido, porque se tuvo que mejorarlo y de la mejora nace el FF-D. Estos son los dos FF que vamos a estudiar.

1.1. Circuito básico de un FF R-S.

El circuito de FF R-S puede estar formado por dos compuertas NAND o dos compuertas NOR. Analizaremos el caso del FF R-S con compuertas NOR tal como aparece en la siguiente figura. Cada flip-flop tiene dos salidas, Q y Q', y dos entradas S (set) y R (reset).



Para analizar la operación del circuito de la figura anterior se debe recordar que la salida de una compuerta NOR es 0 si cualquier entrada es 1 y que la salida es 1 solamente cuando todas las entradas sean 0.

- Como punto de partida asúmase que la entrada S (set) es 1 y que la entrada R (reset) sea 0. Como la compuerta A tiene una entrada igual a 1, su salida $\bar{Q}$ debe ser 0, lo cual coloca ambas entradas de la compuerta B a 0 para tener la salida Q igual a 1, en este caso hemos seteado al FF, esto se entiendo hacer que la salida sea igual a "uno".

- Si la entrada R (reset) es 1 y S es 0 , en la compuerta B tendremos una entrada igual a 1, si le sumamos la otra entrada, obtendremos 1, lo negamos y obtendremos la salida Q igual a 0 y $\bar{Q}$ igual a 1. hemos reseteado el FF, esto se entiende hacer que la salida sea igual a "cero".

- Cuando se aplica un 1 a ambas entradas R y S, ambas salidas Q y $\bar{Q}$ van a 0. Esta condición viola las leyes, ya que no es posible que sean iguales Q y $\bar{Q}$, éstas deben ser opuestas entre sí. En una operación normal esta condición debe evitarse, este es un estado prohibido.

- Cuando las entradas S y R son iguales a cero, las salidas permanecerán iguales, mantendrán sus estados anteriores, por ejemplo, si hemos seteado con S=1 y R=0 haciendo que la salida Q tome el valor igual a 1, y cambiamos S a cero, haciendo que estemos en el estado 00, el valor de Q no se altera, mantiene el valor igual a 1. Por el contrario, si hemos reseteado con S=0 y R=1, la salida Q será cero, si luego pasamos R a cero regresando al estado 00, el valor de Q mantiene su valor, sigue siendo Q=0.

El estado más importante de este sistema digital es el estado "00" (R = 0 y S= 0), en este estado la salida no se altera mantiene el mismo estado, si Q era "1" sigue siendo "1", si Q era un "0" sigue siendo "0" cuando vamos al estado "00". Este estado "recuerda" que valor tenía Q y sigue en el mismo estado. Esta interpretación permitió usar a los FF como el medio para almacenar un bit, almacenar un "0" o un "1".

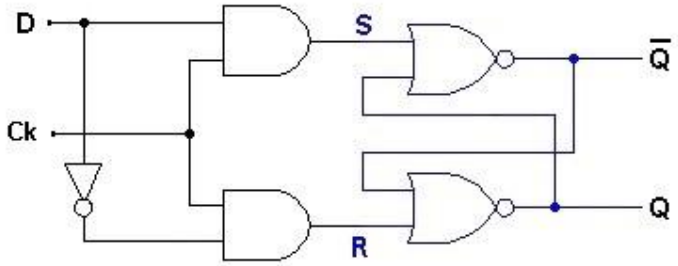
Esto que hemos explicado podemos expresarlo mediante la tabla de estado del FF R-S.

Tabla de estado del FF R-S

	S	R	Q	$\bar{Q}$
Con S=0 y R=0 la salida no se altera, mantiene su valor anterior Q →	0	0	Q	$\bar{Q}$
	0	1	0	1
	1	0	1	0
Estado prohibido →	1	1	0	0

1.2. Flip-flop D

El flip-flop D es una modificación del flip-flop R-S al cual se le ha agregado un inversor y dos compuertas AND. La entrada D va directamente a la entrada S y su complemento se aplica a la entrada R a través del inversor, con ello se evita el estado 11 al parte que corresponde la FF R-S, solo es posible el ingreso de los siguientes estados: 00, 01 y 10.

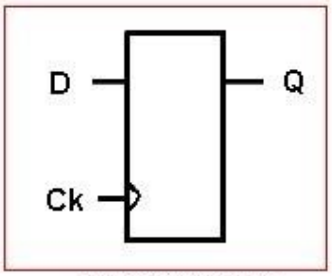


Mientras que la entrada Ck sea "0", no se produce ningún cambio en la salida Q, por más que cambie los estados de la entrada D. Por otro lado, si Ck es "1", la salida Q cambia de estado al cambiar la entrada D, por ejemplo si D es iguala uno la salida Q será también "1", pero si D es igual a cero, entonces, la salida Q será también "0". Esto se pude resumir en la siguiente tabla de estado.

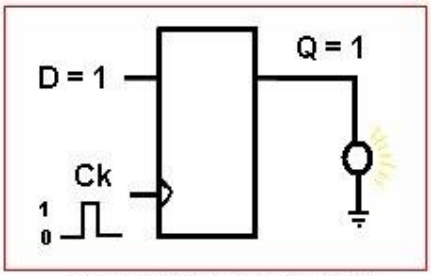
Tabla de estado del FF D

	Ck	D	Q	$\bar{Q}$
Con Ck=0 la salida no se altera, mantiene su valor anterior Q →	0	0	Q	$\bar{Q}$
	0	1	Q	$\bar{Q}$
Con Ck=1 la salida cambia con la entrada D, Q=D →	1	0	0	1
	1	1	1	0

El FF – D me permite poder guardar un bit de información digital, por lo tanto, en él podría guardar un 0 o un 1, si digo "puedo guardar", implica que se está comportando como un medio de almacenamiento, como una memoria, aunque por ahora de muy poca capacidad. Para simplificar el esquema del FF D se lo representa como un bloque con dos entradas D y Ck y una salida Q, tal como lo vemos en la siguiente figura:



Esquema del FF D



El FF D alamcenando un "1"

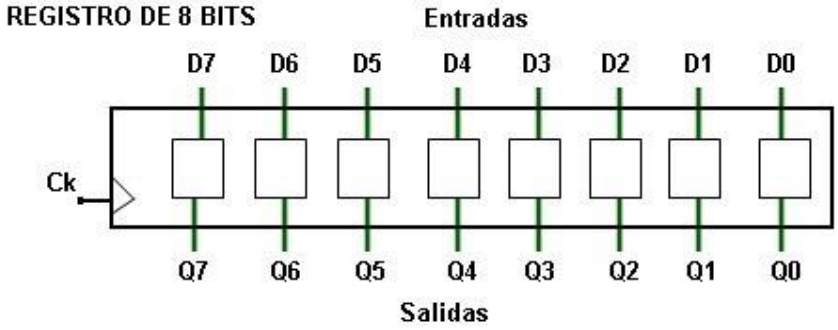
Haciendo uso de cinco compuertas hemos logrado desarrollar un sistema digital que almacena 1 bit, este es el FF D. De forma sencilla podemos pensar que el FF D se comporta como un cajón donde se puede guardar un cero o un uno. Por ejemplo, si queremos guardar un "1", debemos hacer que D sea igual a "1" y para que ingrese el uno al cajón, lo que debemos hacer es cambiar por un pequeño instante el estado de Ck, de cero lo pasamos a uno e inmediatamente lo regresamos a cero, así de esta forma, la salida toma el valor de "1" y no cambiará mientras Ck se mantenga en cero.

Si ya tenemos un sistema digital que almacena un bit, nuestro siguiente desafío es hacer un sistema que pueda guardar un grupo de bits, más específicamente 8 bits o un byte. La respuesta a este desafío es mediante 8 FF D, y a este sistema le llamamos Registro.

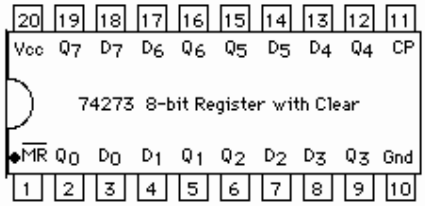
1.3. Registro

Es un conjunto de FF D, el cual permite guardar información en una cantidad limitada de bits, el registro de 8 bits estará formado por 8 FF, tal cual como lo muestra la figura. En ella se denota las variables de entrada, desde D0 hasta D7, y de las salidas sólo hemos indicado las salidas Q, pero también estarán allí sus complementos $\overline{Q}$. Con respecto a la variable de control Ck, ahora ésta está conectada a todos los FF, para que actúen simultáneamente. Mientras Ck permanezca en 0, las salidas no se verán afectadas, si queremos cambiar su valor deberemos poner a la entrada el valor deseado (los 8 bits) y luego haciendo Ck = 1, las salidas tomarán este nuevo valor, para que no cambien debemos retornar

Ck a cero. Recuerden que Ck solo debe recibir pulsos en el momento que se quiera ingresar un dato, y mantenerlo siempre en "0".



Los registros se usan como medios de almacenamiento de datos en los distintos controladores, siendo el microprocesador, el que más los utiliza, tomando nombres particulares de acuerdo con la función que desempeñan internamente. Por ejemplo, en los microprocesadores Intel, existe los registros AX, BX, CX, DX, CS, IP, SS, etc.
La cantidad de bits varía siempre en cantidades de 8 bits, por ejemplo, tenemos de 8, 16, 32 y 64 bits, un ejemplo de registro lo mostramos en el siguiente chip, utilizado en algunos sistemas digitales para guardar un byte.



Si mediante compuertas se ha creado un FF D para almacenar un bit, agrupando FF´s podemos generar los registros, pero que pasa si queremos almacenar miles de bytes o millones o miles de millones de bytes. ¿Se puede crear con compuertas un sistema que almacene por ejemplo 256 MB?

La respuesta lo tenemos en el siguiente sistema digital, la memoria la cual me va a permitir guardar millones de bytes, para ello se deberá usar millones de registros.

1.4. Memoria

Para poder almacenar una gran cantidad de información se requiere ahora lo que llamamos memoria, pero ¿Cómo podríamos desarrollarlo mediante compuertas?

La respuesta es similar al caso del registro, pero aquí utilizaremos a los registros como elementos base; por ejemplo, si queremos implementar una memoria de 256 bytes, necesitaríamos 256 registros de 8 bits, si queremos una memoria de 32MBytes, necesitaremos más de 32 millones de registros.

Lo complicado de la memoria es que cada registro debe estar identificado con un número para poder ubicarlo, este número o código es su dirección. Cada registro debe tener su propia dirección, esta dirección deberá estar escrita en el sistema binario usando n bits, dependiendo de la cantidad de registros que tiene la memoria. Por ejemplo, si la memoria es de 16 bytes, significa que tiene 16 registros y se necesita 16 direcciones, estas direcciones lo podremos generar con 4 bits, porque con 4 bits puedo generar 2^4 direcciones igual a 16 direcciones.

Si la memoria fuese de 1024 bytes o sea 1 Kbytes, se necesita 1024 direcciones, esta cantidad de direcciones lo conseguiremos con n bits siendo 2^n igual a 1024, por lo tanto n es igual a 10. Las direcciones iniciarán con 0000000000 hasta 1111111111, para expresar mejor estas direcciones usemos el sistema hexadecimal, de esta forma las direcciones van desde 000H hasta 3FFH.

Para poder determinar la cantidad de bits usados en las direcciones debemos tener en cuenta lo siguiente:

Cuando se trabaja con bits tenemos que tener en cuenta las potencias de 2, saber calcular $2^{n.}$

- Con n bits se puede generar una tabla con 2^n estados.
- Considerar la secuencia

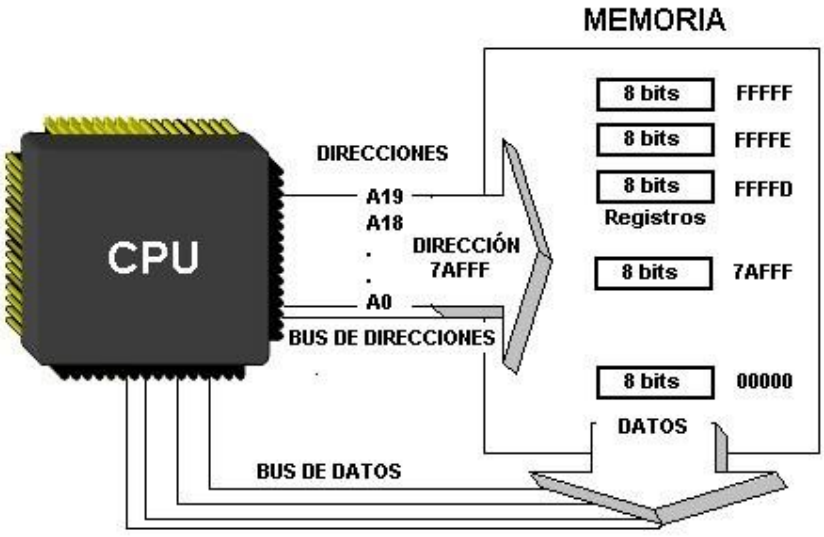
1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 ó 1K
1K = 1024 = 2^{10}
1K, 2K, 4K, 8K, 16K, 32K, 64K, 128K, 256K, 512K, 1024K ó 1M
1M = 1024K = 2^{20}
1M, 2M, 4M, 8M, 16M, 32M, 64M, 128M, 256M, 512M, 1024M ó
1G = 1024M = 2^{30}

Si la memoria fuese de 1MB, necesitaría 1M direcciones y esto lo logramos con 20 bits ya que 2^{20} es igual a 1M, el CPU deberá generar las direcciones desde 00000 hasta FFFFF. Si la memoria fuese de 1GB, como las que venden ahora, necesitaría 230 direcciones, por lo tanto, se necesitan 30 bits para direccionar desde 00000000 hasta 3FFFFFFF.

Mediante estas direcciones el CPU podrá seleccionar un registro y poder en el grabar un byte o leer el contenido de cada uno de los registros direccionados. Estos 8 bits que salen del registro los podemos identificar con D7 a D0, cada uno de estos bits viajan en un cable, por lo que se necesitan 8 cables para trasladar los 8 bits, a este conjunto de cables se le llama bus de datos.

Por otro lado, el CPU debe indicar la dirección mediante n bits, cada bit debe llegar a la memoria en un cable, por lo tanto, llegan n cables y a este conjunto de cables se le llama bus de direcciones, si la memoria es de 1MB las direcciones llegarían en 20 cables denominados desde A19 hasta A0.

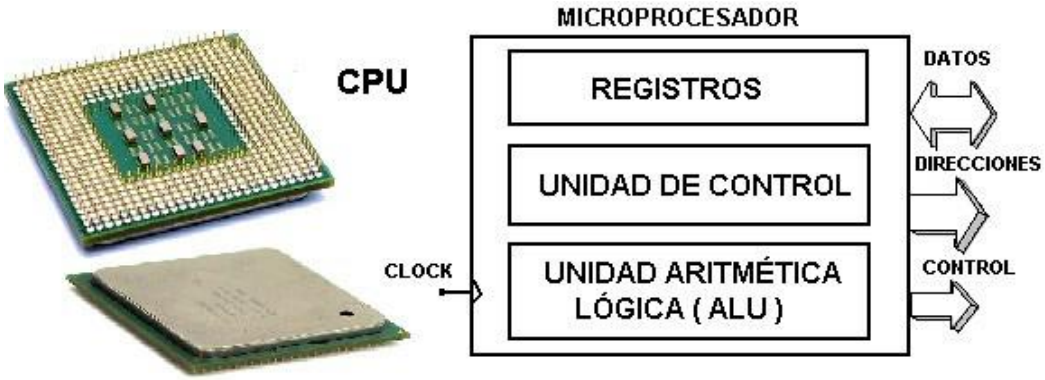
De manera sencilla podemos ver que una memoria tiene un bus de datos, bus de direcciones y en la parte interna millones de registros, tal como aparece en la siguiente figura.



2. CPU - UNIDAD CENTRAL DE PROCESAMIENTO

El CPU es la Unidad Central de Procesamiento, que como su nombre lo indica se encarga del procesamiento, procesamiento de datos, procesamiento de información. El procesamiento lo hace gracias a que puede ejecutar un programa el cual ha sido desarrollado para ese fin. Un programa es un conjunto de instrucciones, mediante el cual el CPU puede procesar. Es interesante conocer a este CPU, este es el sistema digital más complejo, por lo que nos interesa saber que componentes lo conforman.

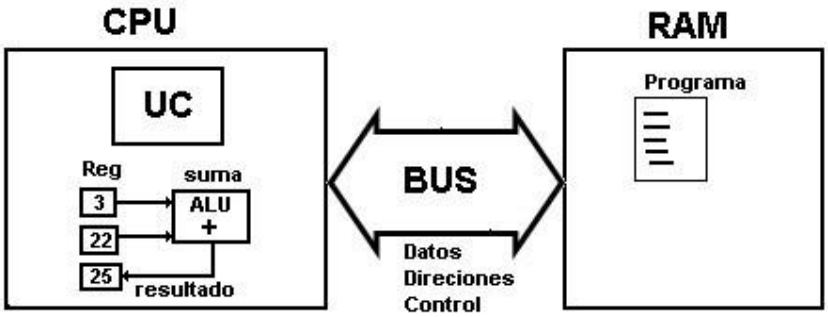
Como el tipo de computadora que usamos es la microcomputadora, a su CPU se le llama microprocesador, este CPU tiene tres componentes básicos: unidad de control, unidad aritmética lógica y registros, tal como se puede apreciar en la siguiente figura.



2.1. Unidad de control

La función de la unidad de control es coordinar todas las actividades del computador. Controla los buses a través de los cuales se comunica con el resto de componentes del computador, en especial con la memoria RAM, porque allí deben estar los programas a ser ejecutados.

Esta unidad direcciona a la RAM, buscando su instrucción u orden, lo hace a través de su bus de direcciones, una vez ubicada la instrucción, esta es leída, ello significa que es trasladada, de la RAM a la unidad de control, a través del bus de datos. La instrucción luego es interpretada o decodificada y a continuación ejecutada. Para la ejecución, a veces, necesita guardar datos temporalmente, para ello usa los registros, otras veces, necesita realizar cálculos, ya sea aritméticos o lógicos, para ello usa el ALU.

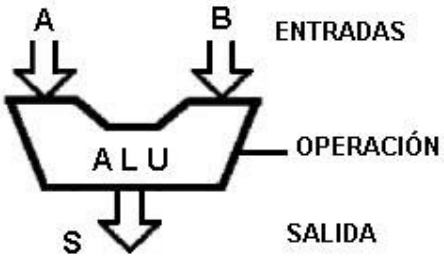


Todos los recursos del CPU son administrados por la unidad de control. Esta unidad interpreta las instrucciones de los programas que ejecuta, para ello dispone de un decodificador de instrucciones, a través de un microprograma. La unidad de control contiene una lista de todas las operaciones que puede realizar el CPU, ósea, un conjunto de instrucciones. Cada instrucción del conjunto de instrucciones está acompañada por un código. Estos códigos son instrucciones básicas o micro instrucciones, que le dicen a la CPU cómo ejecutar las instrucciones.

En resumen, la unidad de control es la que supervisa, controla las demás partes del computador y regula el trabajo que debe realizar, para la ejecución de los programas debe buscar la instrucción y para ello direcciona mediante su bus, luego lee la instrucción ubicada, la interpreta o decodifica, para saber que tiene que hacer, y finalmente la ejecuta. Una vez que termina de ejecutar la instrucción, debe direccionar la siguiente instrucción, luego lee, interpreta y ejecuta, luego, direcciona, lee, interpreta y ejecuta, y así sucesivamente. Note que la tarea de un CPU solo se resume en ejecutar instrucciones.

2.2. Unidad de Aritmética Lógica – ALU (Arithmetic Logic Unit)

En el ALU se realizan operaciones aritméticas (como adición, substracción, etc.) y operaciones lógicas (como OR, NOT, XOR, etc.), entre dos números. Como vemos es un sistema digital complejo diseñado para que pueda recibir como entrada dos números y dar como resultado una salida, el cual es el resultado de realizar alguna operación aritmética o lógica.



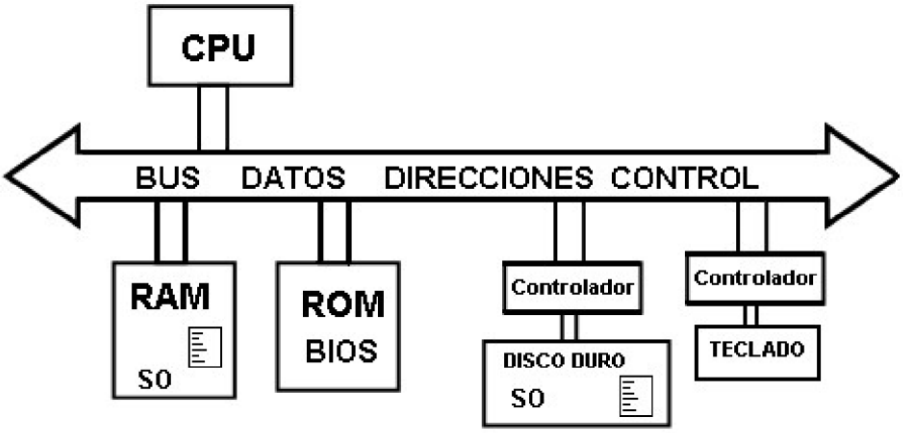
2.3. Registros

Es el medio de almacenamiento temporal para los datos de las instrucciones, así como la ALU necesita el uso de registros para poner los datos a operar. También en otras partes de la CPU se necesita de registros para poner los datos y las direcciones. Lo importante de reconocer en estos registros es su capacidad, lo medimos en bits, por ejemplo: 16, 32 y 64 bits; y para poder ubicarlos, tienen asignado cada uno de ellos su nombre, así, por ejemplo, en los microprocesadores de nuestra computadora hay nombres como: AX, BX, CS, etc.

2.4. Bus de Datos, de Direcciones y de Control.

Como vimos en un gráfico anterior, es necesario que el CPU tenga contacto con la memoria RAM, esto es posible a través de un conjunto de cables a los cuales le llamamos buses, para identificarlos mejor, se los ha clasificado, de acuerdo a su aplicación, en buses de datos, de direcciones y de control, los cuales van permitir la comunicación entre los diferentes componentes de la computadora.

Para entender mejor el funcionamiento del CPU y su interrelación con el resto de componentes, usaremos el siguiente diagrama en bloques.



En este diagrama vemos al CPU, el cual tiene como función, ejecutar instrucciones, que, de acuerdo al diseño del computador, estas instrucciones deben estar en la memoria RAM, tal como se ve en el gráfico. Para que el CPU acceda a la RAM, se dispone de los buses de datos direcciones y control.

La memoria RAM tiene que tener almacenado los programas a ser ejecutados por el CPU, además de ellos, debe tener también un conjunto de programas, los cuales llevan el control del funcionamiento de la computadora, a este conjunto de programas le llamamos Sistema Operativo (SO). A manera de ejemplo, la mayoría de nuestras computadoras tiene que tener en la memoria RAM el SO Windows, para que pueda funcionar la computadora.

La memoria RAM tiene una característica especial, se dice que es volátil, esto significa que, si se apaga el computador, toda la información y programas desaparecen, por lo que al encender nuevamente el computador en la memoria RAM no hay nada, por lo que el CPU no podría hacer nada. Para solucionar este problema se debe trabajar con otra memoria, llamada memoria ROM, la cual, a diferencia de la RAM, es una memora no volátil.

En la memoria ROM están los programas que le van a permitir al CPU, tener las instrucciones iniciales en el momento de encendido de la computadora, pero no puede contener al sistema operativo, tiene si, un pequeño sistema básico llamado BIOS (Sistema básico de entrada / salida) con el cual, el CPU va poder cargar el sistema operativo en la RAM.

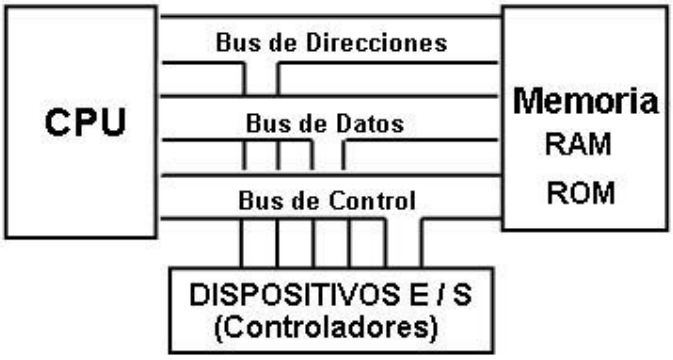
El SO está almacenado en el disco duro, el cual es otro elemento importante de la computadora, que permite almacenar grandes cantidades de información y también no es volátil, pero el CPU no toma las instrucciones directamente del disco duro, porque su acceso es lento y por ello debe ser trasladado a la RAM. Esta transferencia lo realiza el CPU, pero para ello necesita de las instrucciones que le indiquen como hacerlo, estas instrucciones están en el BIOS, por ello cuando encendemos la computadora, el CPU va directo al ROM BIOS.

Como vemos los buses le permiten al CPU comunicarse con los componentes de la computadora ya mencionados, note que, en el caso del disco duro, este no se conecta directamente al bus del microprocesador, vemos que el disco tiene también su bus, pero este bus es diferente al bus principal, por lo que debe

haber una etapa intermedia que le sirva de nexo, esta etapa debe servir de adaptador o interfaz y se le llama controlador, este controlador le llamaremos controlador del disco duro.

Para ingresar datos disponemos de diferentes medios, como e teclado, este también tiene su bus diferente al principal, por lo que el teclado, para su conexión debe usar un controlador, al cual llamamos controlador de teclado. De igual manera el resto de componentes de la computadora necesitan de controladores para su conexión con el bus principal y tener contacto con el CPU.

Mediante los buses de datos, direcciones y control, el CPU puede acceder a los programas que se encuentran en las memorias, como la RAM y la ROM y también comunicarse con los controladores, tal como aparece en la siguiente figura. El CPU podrá direccionar a la memoria para ubicar un registro de un byte, para ello usa todas sus líneas de direcciones, también direcciona a los dispositivos de entrada y salida (E/S) y en estos controladores hay registros de un byte, pero para direccionar estos registros solo utiliza 16 bits.



- **Bus de Direcciones.**

Este bus permite transportar la dirección de origen o destino de la información que se transmite sobre el bus de datos. El CPU genera una dirección en su bus para ubicar, por ejemplo, una instrucción que se encuentra en la RAM. Este bus es controlado por el CPU, pero no es el único, lo puede también controlar un Bus Master o Controlador de DMA con el consentimiento del CPU.

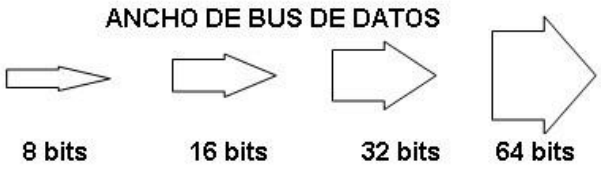
En los diferentes microprocesadores que se han desarrollado, se ve que el bus de direcciones ha sido incrementado para logra una mayor capacidad de

direccionamiento, dicho de otra forma, podrá manejar mayor cantidad de RAM.

- **Bus de Datos.**

Este bus permite transportar los datos a o desde el dispositivo direccionado, es un bus bidireccional, ingresan y salen datos del CPU. Determina la cantidad de información transferida en cada instante, para ello dependemos de la cantidad de líneas que utiliza, por ejemplo, si tiene 64 cables, transporta 64 bits en cada instante.

El ancho del bus de datos, en los microprocesadores, han sido incrementados, como se aprecia en el gráfico siguiente, el objetivo es aumentar la velocidad de transferencia.



- **Bus de Control.**

Mediante este bus el CPU transportar las señales de reloj y controla la ejecución de los ciclos completos. Determina cuando un dispositivo puede ser leído o escrito, ya sea a memoria o a los dispositivos de E/S y principalmente a través de este bus controla la comunicación con los controladores.