INTRODUCCION

Para describir un sistema digital en términos de funciones tales como sumadores, decodificadores y registros, es necesario emplear una notación matemática de alto nivel. El método de lógica de transferencia entre registros copa esta necesidad. En este método, se seleccionan registros como los componentes primitivos de un sistema digital en vez de las compuertas y los flip−flops como en la lógica secuencial. De esta forma es posible describir en forma precisa y concisa el flujo de información y las tareas de procesamiento entre los datos acumulados entre registros.

Una característica importante de presentación del método lógico de transferencia entre registros es que esta relacionado muy de cerca de la forma en que la gente prefiere especificar las operaciones del sistema digital estas se describen de mejor manera especificando:

- El conjunto de registros en un sistema y sus funciones.
- La información en código binario almacenado en los registros.
- Las operaciones realizadas a partir de la información almacenada en los registros.
- Las funciones de control que inician la secuencia de operaciones.

Estos cuatro componentes forman la base del método de lógica de transferencia entre registros para describir sistemas digitales.

Una afirmación en un lenguaje de transferencia entre registros consiste de una unida de control y una lista de micro operaciones. La función de control (la cual pude ser omitida algunas veces) especifica la función de control y secuencia de tiempos para ejecutar la lista de micro operaciones. Las micro operaciones especifican las operaciones elementales que se realizan con la información almacenada en los registros las cuales se pueden clasificar en cuatro categorías:

- microoperaciones de transferencia entre registros que no cambian el contenido de la información cuando la información binaria se mueve de un registro a otro.
- Las microoperaciones aritméticas realizan aritmética con los números almacenados en los registros.
- Las microoperaciones lógicas realizan operaciones tales como AND y OR con el par de bits individuales almacenados en los registros.
- Las microopraciones de desplazamiento especifican operaciones para los registros de desplazamiento.

TRANSFERENCIA ENTRE REGISTROS

Los registros de un sistema digital son designados por letras mayúsculas (algunas veces seguidas de números) para denotar la función de registro. Por ejemplo, el registro que retiene una dirección para la unidad de memoria se llama comúnmente registro de direcciones de memoria y se designa como MAR (memory address register). Otras serian A, B, R1, R2, e IR.

La forma más común de representar un registro es por medio de un rectángulo con el nombre del registro dentro del de la manera mostrada en las figuras.

Registro A Representación con celdas individuales

Los registros mostrados en un diagrama de bloque pueden convertirse fácilmente en proposiciones de declaración para propósitos de simulación.

La transferencia de un registro a otro se designa en forma simbólica por medio del operador de reemplazo. La

proposición:

A ð B

Denota la transferencia del contenido del registro B al registro A.

Una proposición que especifica una transferencia entre registro implica que los circuitos están conectados entre las salidas de los registros fuente hasta las celdas de entrada de los registros de destino. La condición que determina cuando ocurre la transferencia se llama función de control. Una función de control es una función de Boole que puede ser igual a 1 ó 0.

x'T1: A ð B

La función de control se determina con dos puntos. Esta simboliza las necesidades que la operación de transferencia puede ejecutar por medio de los materiales, solamente cuando la función de Boole x'T1 = 1.

Los símbolos básicos de la lógica de transferencia entre registros se listan en la siguiente tabla. Los registros se denotan por letras mayúsculas y los números pueden estar contiguos a las letras. Los suscritos se usan para distinguir las celdas individuales del registro. Los paréntesis se usan para definir una porción de un registro. La lecha denota una transferencia de información y la dirección de la misma. Dos puntos terminan una función de control y la coma se usa para separar dos o más operaciones que se ejecutan al mismo tiempo.

BUS DE TRANSFERENCIA

Considérese la figura 8−4

4 1 4 1

Habilitar

Uso del multiplexor para transferir información de dos fuentes a un solo destino.

A medida que aumenta él numero de registros, aumenta el numero de multiplexores y el numero de líneas de interconexión.

Un grupo de alambres a través de los cuales se transfiere la información binaria bit a bit, un bit a la ves se llama bus. Para la transferencia en paralelo, el numero de líneas es igual al numero de bit en los registros.

Un sistema de bus comun puede construirse con multiplexores y un registro de destino para que el bus de transferencia pueda seleccionarse por medio de un decodificador. Los multiplexores seleccionan un registro fuente para el bus y el decodificador selecciona un registro de destino para transferir la información desde el bus y el decodificador selecciona un registro de destino para transferir la información desde el bus.

TRANSFERENCIA DE MEMORIA

Un registro de memoria o palabra se simboliza por medio de la letra M. El registro particular entre los muchos disponibles en una unidad de memoria se selecciona por medio de la dirección de memoria durante la transferencia. Es necesario especificar la dirección de M cuando se escriben proposiciones de transferencias de memorias. En algunas aplicaciones, solamente un registro de direcciones se conecta a las terminales de direcciones de memoria.

Cuando se conecta solamente un registro a la dirección de memoria, se sabe que este registro especifica la

dirección y que se puede adoptar una proposición que especifica la dirección y que se puede adoptar una convención que simplifica la notación.

Definimos lectura como la transferencia de información a partir de un registro de memoria exterior. La escritura se define como la transferencia de información nueva a u registro. Ambas operaciones se especifican por medio de una dirección del registro de memoria seleccionado.

El registro de memoria (M) se selecciona por medio de la dirección de memoria durante la transferencia. Es necesario especificar la dirección de M cuando se escriben proposiciones para transferencia de memorias.

Cuando se coloca solamente un registro a la dirección de memoria se sabe que este registro especifica la dirección. Si la letra M aparece por si sola en una preposición designara siempre un registro de memoria seleccionado por la dirección que esta presente en el MAR (Memory Address Register). De otra manera el registro que especifica la dirección se encierra entre llaves cuadradas después del símbolo M.

Considérese una unidad de memoria con un registro de direcciones MAR y un registro separador de memoria MBR. Existen dos operaciones de transferencia de memoria, la primera la operación de lectura que es una transferencia de un registro M de memoria seleccionando al MBR esto se representa como:

R : MBR M (donde R es la función de control de lectura)

La segunda que es la operación de escritura que básicamente es la transferencia del MBR al registro de memoria seleccionado M representado como:

W : M MBR (donde w es la función de control de lectura)

El tiempo de acceso de una unidad de memoria debe estar sincronizada por pulsos de reloj. En memorias rápidas debe ser menor o igual al pulso de reloj en memorias lentas pueden ser necesario varios pulsos de reloj y en memorias de núcleos magnéticos, los registros del procesador deben esperar para el tiempo de memoria se complete.

MICROOPERACIONES ARITMÉTICAS, LÓGICAS Y DESPLAZAMIENTO

*MICROOPERACIONES ARITMÉTICAS

Las Microoperaciones aritméticas básicas son: suma, resta, complementar y desplazar. Todas las demás operaciones aritméticas se derivan de las Microoperaciones básicas. La suma aritmética es representada como:

F A + B

Donde el contenido del registro A se le va a sumar al contenido del registro B y esto se transfiere al registro F.

La sustracción o resta se representa a menudo por medio de la complementación y suma:

–

F A + B +1

Donde al registro A se le agrega el complemento de 2 al registro B.

Las operaciones de incremento y decremento se simbolizan por la operación de mas 1 o menos 1 respectivamente. La siguiente tabla nos muestra las Microoperaciones aritméticas.

Descripción Simbólica	Descripción
F A + B	Contenido de A mas B se transfiere a F
F A − B	Contenido de A menos B se transfiere a F
B B´	Se complementa el registro B (Complemento de 1)
B B´ + 1	Forma el complemento de 2 del contenido del registro B
F A + B´ + 1	A mas el complemento de 2 de B se transfiere a F
A A + 1	Incrementar el contenido de A en 1 (cuenta creciente)
A A + 1	Decrementar el contenido de A en 1 (cuenta decreciente)

Debe recalcarse que las operaciones aritméticas multiplicación (*) y de división (/) no se listan dentro de las operaciones aritméticas básicas. Ya que la única forma de en que estas operaciones pueden llevarse a cabo es mediante los circuitos combinacionales. En la mayoría de los computadores la operación de multiplicación se ejecuta mediante una secuencia de Microoperaciones de suma y desplazamiento y la división se ejecuta de forma de Microoperaciones de resta y desplazamiento.

*MICROOPERACIONES LÓGICAS

Las Microoperaciones lógicas especifican operaciones binarias para una cadena de bits almacenados en los registros. Estas operaciones consideran cada bit en los registros separadamente y lo tratan como una variable binaria.

Existen 16 operaciones lógicas diferentes posibles que pueden realizarse con dos variables binarias, las cuales se pueden representar en términos de AND, OR y complemento.

Estos mismos términos AND y OR se representan con sus símbolos correspondientes. Tenemos que para representar la operación AND utilizamos (/\) y para representar OR utilizamos (\/) y para complemento usamos (´).Las Microoperaciones lógicas se listan a continuación.

Descripción Simbólica	Descripción
A A´	Complementa todos los bits del registro A
F A \/ B	Microoperación lógica OR
F A /\ B	Microoperación lógica AND
F A B	Microoperación lógica OR exclusiva
A shl A	Registro A de desplazamiento a la izquierda
A shr A	Registro A de desplazamiento a la derecha

La razón por la cual se adoptan los signos es que la Microoperación OR puede tener dos significados diferentes ya que él más aritmético (+) en una función de control o de Boole denota OR y en las variables de registro denota suma aritmética. Esto lo podemos constatar en el siguiente ejemplo:

T1 + T2 : A A + B, B D \/ F

Donde (+) entre T1 y T2 es una operación OR entre las variables de control y el (+) entre A y B denota suma y (\/) da a conocer que es la Microoperación OR entre D y F.

*MICROOPERACIONES DE DESPLAZAMIENTO

Las Microoperaciones de desplazamiento transfieren la información binaria entre registros en los computadores en serie y también se usan para operaciones aritméticas, lógicas y de control. No hay símbolos

para esta Microoperaciones pero pueden adoptarse los siguientes símbolos:

A shl A, B shr B

La primera expresión significa un desplazamiento de un bit a la izquierda del registro A y la segunda un desplazamiento a la derecha de un bit del registro B. La información transferida a los flip−flops extremos no se especifica por los símbolos shl y shr, por lo tanto, una proposición de una Microoperación de desplazamiento debe estar acompañada con otra Microoperación que especifica el valor de la entrada en serie del bit transferido al flip−flop extremo. Por ejemplo:

A shl A, A1 An

Es un desplazamiento circular que transfiere el bit del extremo izquierdo desde An hasta el flip−flop de la extrema derecha A1.

PROPOSICIONES CONDICIONALES DE CONTROL

Es conveniente es ocasiones especificar una condición de control por medio de una proposición condicional en vez de con una función condicional de Boole. Una proposición se simboliza de la siguiente manera:

P : *si* (condición) *entonces* [Microoperación(es)]

portanto [Microoperación(es)]

La proposición se interpreta dé manera que si la condición de control, establecida entre paréntesis después de la palabra *si* es verdadera entonces se ejecuta la Microoperación encerrada entre paréntesis después de la palabra *entonces*, si la condición es falsa entonces se ejecuta la Microoperación después de la palabra *porlotanto*. Si la parte de *porlotanto* falta entonces no se ejecuta ninguna operación. Ejemplo.

T2: *si* (C=0) *entonces* (F1) *porlotanto* (F0)

Pero si C es un registro de un bit la afirmación es equivalente a las dos proposiciones siguientes:

C´T2: F 1

CT2: F 0

Esto significa que una sola operación puede ser ejecutada en T2.

DATOS BINARIOS DEL PUNTO FIJO

La información binaria encontrada en los registros representa datos o información de control. La información de control es un bit o un grupo de bits que especifican las operacionesque se van a realizar. Se le llama instrucción a una unidad de iinformación de control en codigo binario almacenada en los registros del computador digital que especifica las operaciones que se van a realizar con los datos acumulados.

*REPRESENTACIÓN DEL SIGNO Y EL PUNTO RADICAL

El signo de un numero es una cantidad discreta de dos valores reprentados por un bit, el mas significativo, por lo regular el signo mas se representa con el cero y el signo menos con un uno.Hay dos maneras posibles de especificar la posición del punto binario es un registro estas son el punto fijo y el punto flotante.

El punto fijo como su nombre lo dice siempre permanece donde mismo. Sus dos posiciones mas usadas son: un punto binario en el extremo izquierdo para hacerlo una fracción y el punto binario en el extremo derecho para hacerlo un entero.

La representación del punto flotante usa un segundo registro para almacenar un numero que designa la posición del punto binario en el primer registro.

*NUMEROS BINARIOS CON SIGNO

Como se menciono anteriormente un numero positivo se reprenta por su magnitud y su signo, peroa diferencia del numero positivo el numero negativo puede puede ser interpretado de las tres maneras siguientes dando como ejemplo el numero 9.

Interpretación	**9 Positivo**	**9 Negativo**
Signo−Magnitud	Signo 0 001001	Signo 1 001001
Signo−Complemento a 1	Signo 0 001001	Signo 1 110110
Signo−Complemento a 2	Signo 0 001001	Signo 1 110111

SUMA ARITMETICA

El proceso de sumar dos números con signo, cuando los números negativos estás representados en la forma de signo−magnitud , requiere que se comparen estos signos. Si los dos signos no son iguales, se comparan las magnitudes relativas de los números y luego se resta el menor del mayor; es necesario determinar el signo del resultado; ejemplo: +25 + (−35) = − (35−25) = −10.

Suma representada por signo−complemento de 2. La suma de dos números binarios con signo y los números negativos representados por sus complementos de 2 se obtiene de la suma de dos números con sus bits de signo incluidos. Se descarta el arrrastre en el bit más significativo (signo). Ejemplos:

+6 0 000110 −6 1 000110

\+ +

+9 0 001001 +9 0 001001

+15 0 001111 +3 0 000011

Suma representada por signo−complemento de 1. La suma de dos números binarios con números negativos representados por sus complementos de 1, se obtiene de la suma de dos números, con sus bits de signo incluidos. Si hay arrastre del bit más significativo (signo), le resultado se incrementa en 1 y el arrastre se descarta. Ejemplos:

+6 0 000110 −6 1 111001

\+ +

+9 0 001001 +9 0 001001

+15 0 001111 10 000010

\+

1

+3 0 000011

La ventaja de la representación en la forma signo−complemento de 2 sobre la forma signo−complemento de 1 (y la forma signo−magnitud) es que la primera contiene solamente un tipo de cero. las otras dos representaciones tienen ambas un cero positivo y cero negativo.

El rango de los números enteros binarios que pueden ser acomodados en un registro de n = a+1 bit es $\pm$ (2^a − 1), donde se reservan a bits para el número y un bit para el signo.

La representación de signo−complemento de 2 puede acomodar números en el rango +(2^a − 1) a −2^a, donde a = n −1 y n es el número de bits en el registro.

SUSTRACCION ARITMÉTICA

La sustracción de dos números binarios con signo, cuando los números negativos están en la forma de complemento de 2, es muy simple y puede exponerse como signo: obtengase el complemento de 2 del sustraendo (incluyendo el signo del bit) y sumese al minuendo (incluyendo el bit de signo).

La sustracción con números en complemento de 1 es similar, excepto por el arrastre final o lleva final de reinicio. La sustracción con signo−magnitud requiere que solamente el bit signo del sustraendo se complementa.

La razón por la cual el complemento de 2 se escoge, en vez del complemento de 1, es para evitar el arrastre final o lleva final de reinicio y la ocurrencia de un cero negativo.

SOBRECAPACIDAD

Una sobrecapacidad es un problema en un computador digital ya que las longitudes de todos los registros, incluyendo todos los registros de memoria son de longitud finita. Un resultado de n+1 bits no puede acomodarse en un registro de longitud normalizada n**.**

Un sobrecapacidad puede ocurrir si los dos números se suman y ambos son positivos o ambos son negativos. Cuando se suman dos números representados en signo−magnitud, se puede detectar fácilmente una sobrecapacidad por el arrastre o el número de bits.

El algoritmo para sumar dos números representados por signo−complemento de 2, produce un resultado incorrecto cuando sucede una sobrecapacidad. Esto debido a que una sobrecapacidad de los bits del número cambian siempre el signo del resultado y se causa un respuesta errónea de n bits. Si el arrastre que se emana de la posición del bit del signo se toma como del signo del resultado entonces la respuesta será correcta.

DESPLAZAMIENTOS ARITMETICOS

Un desplazamiento aritmético es una microoperación que mueve un número binario con signo a la izquierda o a la derecha.

El bit de la extrema izquierda de un registro almacena el bit del signo y los bits restantes almacenan el número. La figura muestra un registro de n bits. El bit An de la extrema izquierda mantiene el bit del signo y se designa como A (S). Los bits del número se almacenan en la parte del registro designada por A (N). A1 se refiere al bit menos significativo, An−1 se refiere a la posición mas significativa de los bits del número, y A se refiere al registro entero.

n n−1 1 n 1

Bit del Bit del número

Signo

Un desplazamiento aritmético a la derecha que divide el número por 2, puede simbolizarse de las siguientes proposiciones:

A(N) ! shr A(N), An−1 ! 0 para signo−magnitud

A!shr A, A(S) !A(S) para signo−complemento de 1 ó de 2.

En la representación de signo−magnitud, el desplazamiento aritmético a la derecha requiere un movimiento de los bits del número con un 0 colocado en la posición mas significativa. en la representación de signo−complemento de 2 o de 1, todo el registro se desplaza mientras que el bit del signo permanece inalterado esto se debe que para un signo positivo se debe colocar un 0 en la posición mas significativa y para un número negativo se debe colocar un 1.

El desplazamiento aritmético a la izquierda que multiplica el número por 2. Puede simboliza por cualquiera de las siguientes proposiciones:

A(N) ! shl A(N), A1 ! 0 para signo−magnitud

A ! shl A, A1 ! A(S) para signo−complemento de 1

A ! shl A, A1 ! 0 para signo−complemento de 2

En la representación de signo magnitud, los bits del numero se desplazan a la izquierda con un 0 colocado en la posición menos significativa. En la de signo−complemento de 1 todo el registro se desplaza y el bit del signo se coloca en la posición menos significativa. El signo−complemento de 2 es similar, excepto que un 0 es desplazado a la posición menos significativo.

Un número desplazado a la izquierda puede causar que ocurra un desbordamiento por sobre capacidad; si existe la siguiente condición antes del desplazamiento:

An−1 = 1 para signo−magnitud

An OREX An−1 = 1 para signo complemento de 1 ó de 2.

En al caso de signo magnitud, se desplaza y desaparece un 1 de la posición mas significativa. En el caso de signo−complemento, ocurrirá la sobrecapacidad si el bit de signo An = A(S), no es igual al bit mas significativo.

Si el bit de signo después del desplazamiento no es el mismo que el bit de signo después de él, ocurrirá una sobrecapacidad. El resultado correcto será un número de *n*+1 bits, con el bit de posición (*n*+1) contenido en el signo original del número el cual desapareció después del desplazamiento.

DATOS DECIMALES

La representación de números decimales en los registros es una función del código binario usado para representar un digito decimal. Un código decimal de cuatro bits, requiere 4 flip−flops para cada digito

decimal.

Al representar los números en decimal, se desperdicia una cantidad considerable de espacio de almacenamiento ya que el numero de flip−flops necesarios para almacenar un numero decimal en código binario es mayor que para representación binaria equivalente.

Hay 3 maneras de representar números decimales negativos de punto fijo. Estas son similares a las representaciones de un número binario negativo, excepto por el cambio del radical:

1.−Signo−magnitud

2.−Signo−complemento de 9

3.−Signo−complemento de 10

Un número decimal positivo se representa por un 0 seguido por la magnitud del número para las tres representaciones. Es con respecto a los números negativos que difieren las representaciones. El signo de un número negativo se representa por un 1 y la magnitud del número es positiva en la representación de signo−magnitud. En las otras dos representaciones la magnitud se representa por el complemento de 9 y de 10.

Es costumbre representar un mas con cuatro ceros y un menos con el equivalente BDC de 9; en esta forma todos los procedimientos desarrollados por los números de signo−complementos de 2 se aplican también a los números de signo−complemento de 10.

Las operaciones aritméticas decimales pueden usar los mismos símbolos que las operaciones binarias siempre y cuando la base de los números se entienda cómo 10 en vez de 2. Los desplazamientos aritméticos son aplicables también a los números decimales excepto que un desplazamiento a la izquierda corresponde a la multiplicación por 10 y un desplazamiento a la derecha a una división por diez. El signo−complemento de 9 es similar al signo−complemento de 1 y la representación de signo−magnitud en ambas representaciones de radicales tienen procedimientos aritméticos similares.

DATOS DEL PUNTO−FLOTANTE

La representación del punto flotante de los números necesita dos registros el primero representa un número con signo de punto fijo y el segundo la posición del punto del radical. Ejemplo, la representación del número decimal +6132.789 es de la siguiente manera:

signo punto decimal inicial signo

! ! !

primer registro segundo registro

(coeficiente) (exponente)

El primer registro tiene un 0 en la posición del flip−flop mas significativo para denotar un mas. la magnitud del número se almacena en un código binario de 28 flip−flops, con cada digito decimal ocupando 4 flip−flops.

El segundo registro contiene el número decimal 4 para indicar que la posición actual del punto decimal es 4 posiciones decimales a la izquierda.

otra posición usada para el exponente es quitar del todo su bit de signos y considerar el exponente como polarizado. Un número binario de punto flotante se representa de manera similar con dos registros, uno para almacenar el coeficiente y el otro para el exponente. Ejemplo el número más 1001.110 puede representarse de la siguiente manera.

signo punto binario inicial signo

! ! !

coeficiente exponente

El registro del coeficiente tiene 10 flip−flops: uno para el signo y 9 para la magnitud. Asumiendo que el coeficiente es una fracción de punto fijo, el punto binario actual es cuatro posiciones a la derecha, el exponente tiene el valor binario de +4.

Coeficiente Exponente

representa el nùmero .2601000 X 10−4 = .0000261000, los cuales producen cuatro ceros de màs a la izquierda.

Otro formato usado para el exponente es quitar el bit de signo y considerara al exponente como polarizado.

El punto decimal se interpreta en la representacion de un numero de la siguiente manera:

c.re

donde c representa el contenido del registro del coeficiente y e el contenido del registro exponente, el radical r y la posicion del punto flotante se asumen siempre. El radical r posee la magnitud de la base del sistema numerico a usar. Por ejemplo r=2 para el binario, r=8 para el octal, etc.

Un numero binario de punto flotante se representa de manera similar con dos registros, uno para almacenar el coeficiente y el otro para el exponente. El registro del coeficiente tiene 9 flip−flops para la magnitud y un flip−flop para el signo.

Un numero octal se representa con cuatro flip−flops para la magnitud y uno para el signo, el registro exponente tiene dos flip−flops para la magnitud del exponente y uno para el signo.

OPERACIONES EN PUNTO FLOTANTE

Para realizar operaciones en punto flotante, primero se establece un número fijo de dígitos para la mantisa y el exponente (no tiene que ser el mismo) a esto se le conoce como forma normalizada.

Suma y Resta

Para relizar sumas o restas se necesita que los dos operandos posean el mismo exponente, en caso de que esto no suceda, lo que se hace es desplazar hacia la derecha la mantisa del número con el menor exponente, incrementándo en uno el valor del exponente por cada uno de los corrimientos.

Luego se suman o restan las mantisas de los operandos, el exponente del resultado es el de cualquiera de los operandos, si la operación genera un acarreo, se desplaza una posición el resultado a la derecha y se incrementa el exponente. Recuérdese que se trata de números con signo por lo que se debe tener cuidado a la

hora de sumar y restar número negativos.

Multiplicación y División

En este caso, no hace falta que los exponentes sean iguales, en el caso de la multiplicación basta con multiplicar las mantisas y sumar los exponentes y para la división se dividen las mantisa y se restan los exponentes. Como en el caso de la suma y la resta, los números ngativos reciben un tratamiento especial, por ejemplo negativo por positivo da negativo, y negativo por negativo da positivo.

DATOS NO NUMERICOS

La mayoria de los programas escritos para los usuarios estan en forma de caracteres, la computadora es capaz de aceptar caracteres (en codigo binario), almacenarlos en la memoria y realizar operaciones.

Los caracteres se representan en los registros de la computadora por medio del codigo binario. Cada componente del codigo representa un carácter y consiste de seis, siete u ocho bits dependiendo del código. El número de caracteres que pueden almacenarse en un registro depende de la longitud del registro y del número de bits usados en el código.

Las cadenas de caracteres se almacenan en la memoria en lugares consecutivos. El primer carácter de la cadena puede ser especificado a partir de la dirección de la primera palabra. El último carácter de la cadena puede encontrarse a partir de la dirección de la última palabra.

Un código binario puede adoptarse para representar diferentes símbolos y así ser almacenados dentro de la memoria.

Una de las operaciones hechas en datos numericos es la de transferencia, en esta se puede preparar la información binaria codificada en algún orden requerido por la memoria .

Las operaciones de lógica y de desplazamiento en datos numéricos ayudan en el proceso de toma de decisiones. Las operaciones lógicas pueden cambiar valores de bits, eliminando un grupo de bits, o adicionar otros valores de bits en los registros.

La operación OR puede ser utilizada para poner a uno un bit o un grupo seleccionado de bits en un registro.

La operación AND puede ser usada para borrar un bit o un grupo seleccionado de bits de un registro.

La operación AND seguida de una operación OR puede usarse para cambiar un bit de un grupo de bits de un valor dado a un nuevo valor deseado. Esto se hace para enmascarar primero los bits y luego aplicar a una compuerta OR el nuevo valor. La operación de máscara es la operación AND y la operación de inserción es la microoperación OR.

La microoperación XOR puede usarse para complementar un bit o un grupo de bits seleccionados de un registro.

Las operaciones de desplazamiento son útiles para agrupar o dispersar información binaria codificada. Agrupar información binaria tal como caracteres en una operación que une dos o más caracteres en una palabra. Dispersar es la operación inversa que separa dos o más caracteres alamacenados en una palabra o caracteres individuales.

La operación binaria disponible en un durante operaciones lógicas se llama una palabra lógica, esta se interpreta como una cadena de bits en oposición a una cadena de caracteres o datos numéricos. Cada bit en

una palabra lógica funciona exactamente de la misma manera que otro bit cualquiera.

CODIGOS DE INSTRUCCIÓN

En un sistema digital para propósitos especiales, la secuencia de microoperaciones se fija y el sistema ejecuta la misma tarea especifica repetidas veces. El usuario de una computadora puede controlar el proceso por medio de un programa. Los códigos de instrucción, conjuntamente con los datos, se almacenan en la memoria. El control interpreta entonces la instrucción y procede a ejecutarla emitiendo, una secuencia de funciones de control. La habilidad de almacenar y ejecutar instrucciones, el concepto de programa almacenado, es la propiedad más importante de una computadora para propósito general.

Un código de instrucción es un grupo de bits que le dice a la computadora como realizar una operación especifica. La parte más básica de un código de instrucción es su parte operativa. El código de operación es un grupo de bits que define una operación. El número de bits requeridos para la parte de operación del código de instrucción es una función del número total de operaciones usadas. Debe de consistir de por lo menos n bits para 2n operaciones dadas diferentes.

FORMATOS DE CÓDIGOS DE INSTRUCCIÓN.

Los bits de instrucción por lo general se dividen en grupos que subdividen la instrucción en `partes. A cada grupo de le crea un nombre simbólico tal como parte del código de operación o parte de una dirección. Las diferentes partes determinan diferentes funciones para la instrucción y cuando se muestran juntas constituyen el formato de código de instrucción.

En la siguiente figura, considérese, los tres formatos del código de instrucción en (a) consiste de un código de operación que implica un registro en la unidad procesadora. El formato de instrucción (b) tiene un código de operación seguido de un operando. Este se llama una instrucción de operando inmediato, porque el operando sigue inmediatamente después de la parte de código de operación de la instrucción. El formato de instrucción en (c) es similar al de (b) excepto que el operando debe extraerse de la memoria al lugar especificado por la parte de dirección de la instrucción.

(a) Implicito

(b) Operando Inmediato

DISEÑO DE UN COMPUTADOR SENCILLO.

Este sistema consiste básicamente de una unidad de memoria, siete registros y dos decodificadores. La unidad de memoria contiene 256 palabras de 8 bits con esto se puede demostrar las operaciones básicas que se encuentran en los computadores.

Una breve descripción de los registros que contiene un computador sencillo el cuál, estos se encargan del proceso de información se muestra en esta tabla.

Símbolo	*Número de bits*	*Nombre del registro*	*Función*
MAR	8	Registro de dirección de memoria	Almacena direcciones de memoria
MBR	8	Registro separador de memoria	Almacena contenidos de palabras de memoria
A	8	Registro A	Registro procesador
R	8	Registro R	Registro procesador

PC	8	Contador de programa	Almacena la dirección de instrucción
IR	8	Registro de Instrucción	Almacena códigos de operación corrientes
T	8	Contador de tiempo	Generador de secuencias

Si queremos leer una instrucción, el contenido que hay en PC se transfiere al MAR y así se inicia un ciclo de lectura de lectura de memoria, así pues el PC se incrementa a 1 y esto hace que almacene la siguiente dirección en la secuencia de instrucciones, después un código de operación leído de la memoria al MBR, se transfiere al IR. Si la parte de dirección de memoria de una instrucción se le al MBR, esta dirección se transfiere al MAR para leer él operando. Entonces el MAR puede recibir direcciones del PC o del MBR.

Este computador esta compuesto de 8 bits en el código de operación por lo que tenemos hasta 256 operaciones diferentes. Pero a manera de simplificar tenemos tres instrucciones para un computador sencillo.

Código de operación	*Mnemónico*	*Descripción*	*Función*
00000001	MOV R	Mover R a A	A R
00000010	LDI OPRD	Cargar OPRD a A	A OPRD
00000011	LDA ADRS	Cargar el operando especificado por ADRS a A	A M(ADRS)

La mnemotécnica asociada con cada instrucción puede usarse por los programadores para especificar las instrucciones con nombres simbólicos.

La sigla MOVE simboliza una instrucción de movimiento, el símbolo R indica el contenido de R que se mueve al registro A, la sigla LDI (load inmediate) simboliza una instrucción de carga inmediata, el OPRD se establece para un operando actual que el programador debe especificar con esta instrucción, LDA(lad into A) es una abreviatura para cargar a A y ADRS establece un número de dirección que el programador debe especificar con esta instrucción.

Un computador con tres funciones no es muy útil. Se debe asumir que este computador tiene muchas más instrucciones pero se consideran tres de ellas.

CICLO DE ENVÍO DE INSTRUCCIONES.

Este ciclo de envío de instrucciones comienza cuando un código de operación cuya dirección esta en PC se lee de la memoria al MBR. El PC se incrementa a 1 para prepararla para la siguiente dirección en secuencia. El código de operación se transfiere del MBR al IR donde es decodificado por el control y a todo esto sele llama ciclo de envío de instrucción, ya que ésta saca el código de operación de la memoria y lo coloca en un registro de control.

Un diagrama de esta operación podría ser:

t0= MAR PC Transferir dirección del cod. de operación

t1= MBR M, PC PC + 1 Leer el cod. de operación, incremento PC

t2= IR MBR Transferir el cod. de operación al IR

Las microoperaciones y funciones de control que preceden al ciclo de envío se determinan en la sección de control a partir del código de operación decodificado.

EJECUCIÓN DE LAS INSTRUCCIONES.

El control usa las variables qi para determinar las siguientes microoperaciones es secuencia. La instrucción tiene MOV R tiene un código de operación que hace q1= 1.

La ejecución de esta instrucción requiere la microoperación:

q1t3: A R,T 0

Así cuando q10 1 en el tiempo t3=, el contenido de R se transfiere al registro A y el registro T se borra.

Cuando PC se incrementa a 1 la instrucción LDI OPRD tiene un código de operación que hace q2=1. Las microoperaciones que ejecutan esta instrucción son:

q2t3: MAR PC transferir dirección del operando

q2t4: MBR M, PC PC + 1 leer el operando incrementar PC

q2t5: A MBR, T 0 transferir él operando, pasar al ciclo de envío

La instrucción LDA ADRS tiene un código de operación que hace q3=1. Las micooperaciones necesarias para ejecutar esta instrucción son:

q3t3: MAR PC transferir la siguiente operación de instrucción

q3t4: MBR M, PC PC + 1 leer DIRECCIÓN, (ADRS) incrementar PC

q3t5: MAR MBR transferir dirección del operando.

q3t6: MBR M Leer él operando.

q3t7: A MBR, T 0 Transferir él operando de A, pasar al ciclo de envío.

DISEÑO DEL COMPUTADOR.

El primer paso en el diseño es repasar las proposiciones de transferencia entre registros, listadas anteriormente y escoger aquellas que realicen la misma macrooperación en el mismo registro, por ejemplo:

t0+ q2t3 + q3t3: MAR PC

Tenemos que recordar que una función de control es una función de Boole. El + entre las funciones de control se refiere a una operación de Boole OR y la secuencia de un operador entre q2 y t3 denota una operación de Boole AND.

Las funciones de control obtenidas para cada micooperación se forman en una ecuación de la variable binaria xi, i=1,2,.....,8.

El diseño de un computador sencillo se puede obtener de la información de transferencia entre registros dada a continuación:

X1 = t0 + q2t3 + q3t3:	MAR ! PC
X2 = q3t5:	MAR ! MBR
X3 = t1 + q2t4 + q3t4:	PC ! PC + 1
X4 = x3 + q3t6:	MBR ! M
X5 = q2t5 + q3t7:	A ! MBR
X6= q1t3:	A ! R
X7 = x5 + x6:	T ! 0
X8 = t2:	IR ! MBR

A8 A7 A6 A5 A4 A3 A2 A1

A

Descripción

Denota un registro

Denota un bit de un registro

Denota una porción de un registro

Denota una transferencia de información

Termina una función de control

Separa dos microoperaciones

Especifica una dirección para una transferencia de memoria

Símbolo

Letras (y numerales)

Suscrito

Paréntesis( )

Flecha ð

Dos puntos :

Coma ,

Llaves cuadradas[ ]

- 0

Multiplexor cuádruple

2x1

Registro A

Registro B

OR

Registro de C

Ts t1

Registro A

A(N)

A(S)

0 0 4

0 6 1 3 7 8 9

0 0 1 0 0

0 1 0 0 1 1 1 0 0 0

104

02601000

Código−operación

Operando

Código−operación

Código−operación

Dirección del operando

(c) Dirección directa