

**MANUAL TEORICO SOBRE EL  
FUNCIONAMIENTO, LÓGICA Y  
ESTRUCTURA DE LA  
ARQUITECTURA DE LOS  
COMPUTADORES**

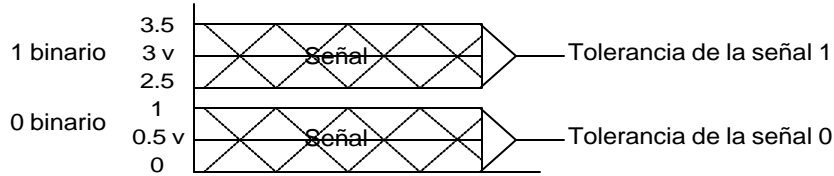
**ELABORADO POR :  
RONALD ALPÍZAR PORRAS**

**1998**

**LÓGICA DIGITAL :**

Computador : sistema digital capaz de proveer información binaria por medio de pulsos llamados señales, una señal está representada por un pulso cuantificable con cierta intensidad. Diferentes cantidades se utilizan para representar los valores boléanos 0(cero) ó 1(uno).

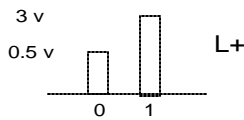
Cada señal debe estar asociada a cierto rango de tolerancia que permite operar los circuitos.



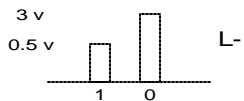
**LÓGICA POSITIVA Y NEGATIVA :**

Estos conceptos tienen que ver con la manera en que los sistemas digitales interpretan el 0 y el 1.

Si un sistema opera con lógica positiva(L+), usará un pulso de corriente más alto para representar el valor binario 1.

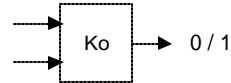


La lógica negativa(L-) usa un pulso mas bajo para representar el valor binario 1 y uno mas alto para el 0.



**COMPUERTAS LÓGICAS(Ko) :**

Son bloques de hardware capaces de responder de diferente forma a la combinación de sus entradas que producen con iguales salidas un único valor el cual será 1 ó 0.



Son usadas para desarrollar diferentes circuitos lógicos capaces de procesar distintos valores boléanos; en la entrada produce los mismos valores según se planteen en una tabla de verdad.

V	V	F	V
V	V	V	V
V	F	V	V

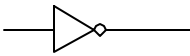
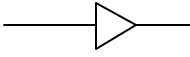






1

V	V	F	F
V	V	V	F
V	F	V	F

0

Cada Ko por sí sola va a ser representada por un símbolo gráfico que responde siempre a una tabla de verdad.

Ejemplo 0 = 1, 1 = 0; y tendrá una función algebraica para facilitar la comprensión de como opera su lógica.

NOMBRE	SIMBOLO	FUNCION ALGEBRAICA	TABLA DE VERDAD
NOT		$X = \overline{A}$ $X = A^1$	0 1
BUFFER (inversor o complemento separador)		$X = A$	1 1
AND		$X = A * B$ $X = AB$	00 0 01 0 10 0 11 1
OR		$X = A + B$	00 0 01 1 10 1 11 1
NAND		$X = \overline{AB}$ $X = (AB)^1$	00 1 01 1 10 1 11 0
NOR		$X = \overline{(A+B)}$ $X = (A+B)^1$	00 1 01 0 10 0 11 0
XOR		$X = A \oplus B$ $X = A^1B + AB^1$	00 0 01 1 10 1 11 0
XNOR exclusivo		$X = A \oplus B$ $X = A^1B^1 + AB$	00 1 01 0 10 0 11 1

**NOTA :**

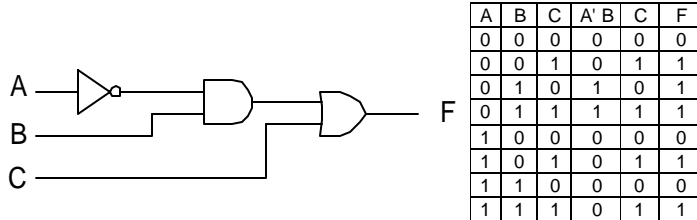
El Buffer es una Ko (Ko = compuerta) que no produce ningún cambio en los valores, si no que se usa con el fin de ampliar o refrescar una señal digital.

**EXPRESIONES ALGEBRAICAS :**

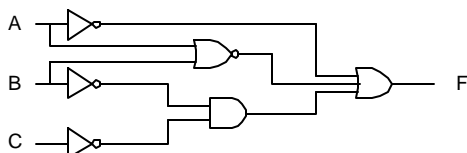
Cada función digital puede ser expresada en una expresión algebraica si conocemos los valores algebraicos de las diferentes Ko de cada expresión algebraica, puedo obtener el diagrama del circuito lógico si conozco la simbología.

1-  $F = \text{NOT } A \text{ AND } B \text{ OR } C$

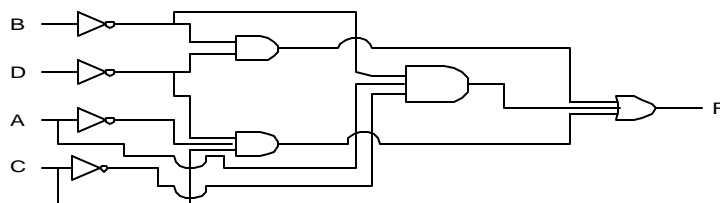
Expresión algebraica  $\Rightarrow F = A^1 B + C$



2-  $F = \text{NOT } (A \text{ OR } B) + \text{NOT } B \text{ AND NOT } C \text{ OR NOT } A$



4-  $F = B^1 D^1 + A^1 D^1 C + AC^1 B^1$



Minitérminos : Cada una de las posibles combinaciones de entrada en un circuito lógico.

[Circuitos integrados (CI's)]

**FAMILIAS DE CIRCUITOS LÓGICOS :**

A-CI's Digitales : Un CI es un pequeño cristal semiconductor de silicio denominado pastilla, que contiene componentes eléctricos tales como diodos, transistores, condensadores y resistencias. Los diversos componentes se conectan dentro de la pastilla para formar un circuito electrónico. La pastilla es montada en un paquete plástico o de metal y las conexiones se hacen por soldadas por afuera para hacer el CI.

Beneficios de usar CI's:

1. menor tamaño
2. menor costo de producción
3. menor requerimiento de potencia
4. más confiabilidad contra fallas
5. más velocidad de ejecución
6. menor ó eliminación de cableado externo

NOTA :

1-Los CI's se presentan en dos formas, 1) chips tipo DIP(double in-line packet) los cuales poseen patas en dos de sus lados y generalmente son rectangulares y 2) los de tipo Reticular ó plano; que tienen patas en sus cuatro lados y normalmente son cuadrados.

**B-Familias :** Los CI's digitales se clasifican por su función, y por pertenecer a una familia de circuitos lógicos específica. Cada familia tiene su propio circuito básico electrónico a partir del cual se desarrollan sus funciones y circuitos más complejos. El circuito básico en una familia es un NOR o un NAND.

Los CI de más uso comercial son :

- TTL(transistor transistor logic)
- ECL(lógica acoplada por emisor)
- MOS(semiconductores de metal oxido)
- CMOS(semiconductores de metal oxido complementario)

**NOTA :** La mayoría de estos circuitos vienen en paquetes de tamaño estándar y un numero de patas que van de los 14 a los 64. Cada uno de los circuitos tiene un numero para su designación numérica el cual es impuesto por el fabricante y luego se publica un catalogo en el cual se da información sobre la composición de cada chip. Los TTL usan 5400 ó 7400 ó algunos 3000 y 9000.

ECL = 10000; CMOS y MOS = 4000, algunos CMOS 54C00 ó 74C00.

**C-Complejidad de los CI's :**

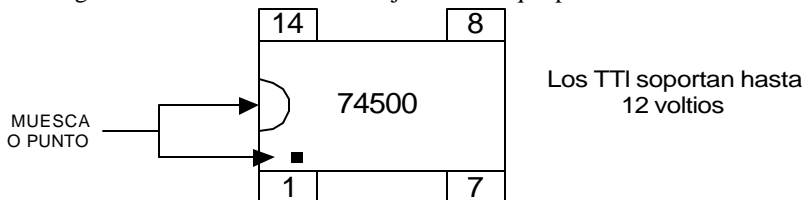
1. S.S.I. (small scale integration) usan varias Ko [menos de 10 Ko].
2. M.S.I. (medium scale integration) usan de 10 a 100 Ko.
3. L.S.I. (large scale integration) microprocesadores, memoria, y pastillas para calculadora.
4. V.L.S.I. (very large scale integration) microprocesadores más complejos y arreglos de memoria.

**COMPARACIÓN ENTRE FAMILIAS DE CIRCUITOS LÓGICOS :**

Existen cuatro características que permiten evaluar y completar las diferentes familias de circuitos lógicos, parámetros que a la vez determinan cuales son los mas convencionales para determinar su uso o función.

El **FAN OUT** : determina el numero de cargas estándar que puede exitar la salida de un Ko sin dañar su operación normal (K = corriente) .

1. Carga estándar se define como : una cantidad de K necesaria para la entrada de una Ko.
2. Disipación de potencia : la potencia consumida por la Ko y que es suministrada por la fuente de poder.
3. Retardo de propagación : es el tiempo de retardo de transición promedio para una señal que se propaga de una a una entrada a una salida, cuando la señal cambia de valor es equivalente a la velocidad de operación de un circuito y la velocidad de un circuito es inversamente proporcional al retardo de propagación; entre más alto sea el retardo de propagación, más rápido es el equipo.
4. Margen de ruido : el mínimo voltaje de ruido que produce un cambio indeseado en la salida.



Si dos circuitos satisfacen idénticamente una tabla de verdad se dice que son equivalentes.

**MAPAS DE KARNAUGH (K) :**

Son herramientas que permiten simplificar el diseño de las funciones digitales.

Se sabe que dos funciones digitales son equivalentes sí y sólo sí satisfacen de igual forma a una tabla de verdad.

Permiten encontrar la función más simple desde el punto de vista que ocupen menos hardware.

Ventajas de utilizar una función más simple.

1. más velocidad
2. menor disipación de potencia
3. menor costo de producción
4. más confiabilidad

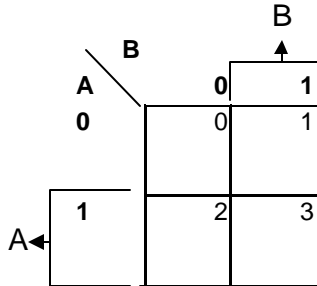
Para hacer un mapa K, requerimos aplicar una serie de reglas y manejar algunos conceptos ya establecidos como una receta<sup>Q</sup> : Minitérmino : Cada una de las posibles combinaciones de las variables de entrada..

<sup>Q</sup>**Reglas :** 1) N variables  $\Rightarrow 2^N$  minitérminos; ABC =  $2^3 = 8$  minitérminos (0 - 7)

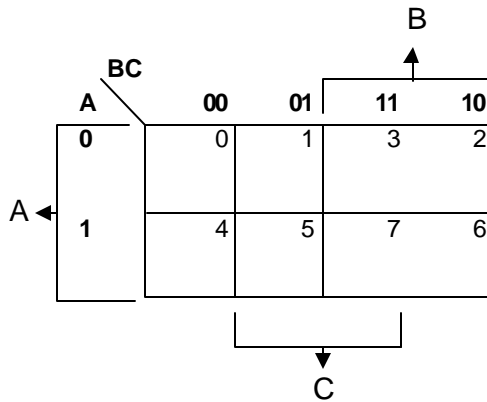
2) Los minitérminos se enumeran de 0 a  $2^N - 1$ .

Para hacer un mapa K se debe desarrollar una matriz o cuadrícula. Un mapa de dos variables produce una matriz de 2 x 2; uno de tres produce una de 4 columnas y 2 filas; y uno de cuatro produce una de 4 x 4; en cada casilla de esa matriz se presentan los minitérminos en un orden preestablecido.

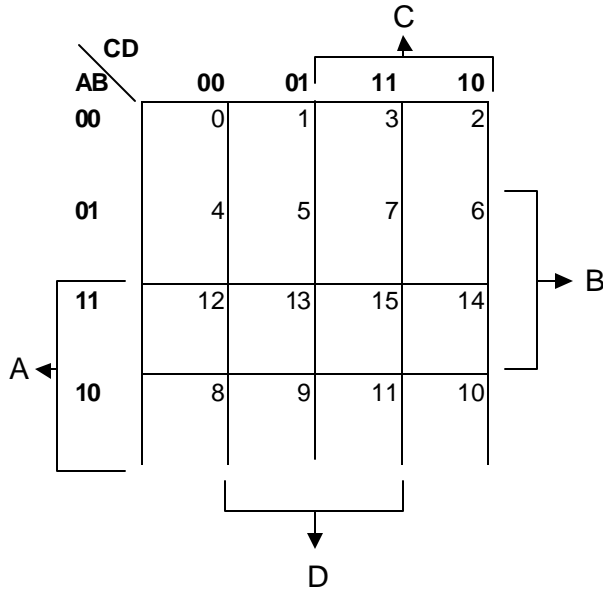
Se dibuja una línea diagonal en la esquina superior izquierda donde se representan los valores de entrada. Sobre cada columna y cada fila se escriben las posibles valores que adoptan esas variables en forma de receta.



A	B	Minitérmino
0	0	0
0	1	1
1	0	2
1	1	3



A	B	C	Minitérmino
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7



A	B	C	D	Minitérmino
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

Por ultimo se dibuja una llave y sobre ella se escribe el nombre de la variable bajo estas se encuentra la media de todos los minitérminos de la cuadrícula y en los cuales esa variable nunca aparece complementada.

Para iniciar la representación de un mapa K se parte de una expresión algebraica  $F(A,B,C) = \Sigma (1,4,5,6,7)[\text{minitérminos}]$ . Donde F = función de salida. A, B, C = lista de variables de entrada, en ese orden y separados por comas después del símbolo  $\Sigma$  y entre paréntesis se listan aquellos minitérminos que producen un valor boleano 1 ó verdadero.

$F(A, B, C) = S(1, 4, 5, 6, 7)$

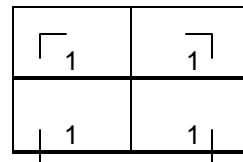
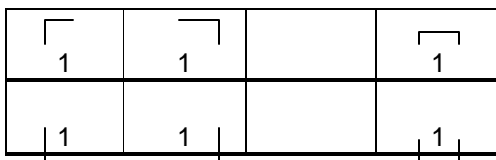
A	B	C	F	Minitérmino
0	0	0	0	0
0	0	1	0	1
0	1	0	0	2
0	1	1	0	3
1	0	0	1	4
1	0	1	1	5
1	1	0	1	6
1	1	1	1	7

$F(X, Y, Z) = S(0, 3, 5, 6)$

X	Y	Z	F	Minitérmino
0	0	0	1	0
0	0	1	0	1
0	1	0	0	2
0	1	1	1	3
1	0	0	0	4
1	0	1	1	5
1	1	0	1	6
1	1	1	0	7

NOTA :

En resumen los cuatro extremos de una cuadrícula son adyacentes. Los grupos de los cuadros adyacentes deben comparar uno o mas cuadros, como uno o mas grupos, deben contener un numero de cuadro que sea una potencia integral de dos. Una adyacencia debe abarcar el mayor numero posible de minitérminos .



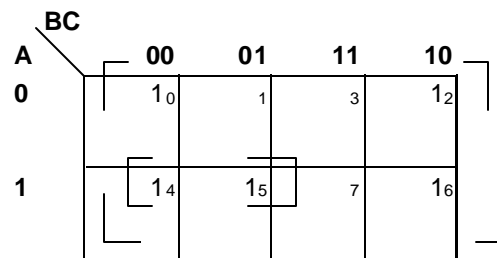
Una vez obtenidas las adyacencias y se evalúan, los valores de las variables, se encontraran los valores que se mantienen iguales y otros que cambian, los que cambian se ignoran y los que se mantienen serán representados por variables primadas cuando su valor sea = 0.

La función simplificada va a estar dada por las evaluaciones de esas adyacencias unidas por un mapa

K.

$F(A,B,C) = S(0, 2, 4, 5, 6)$

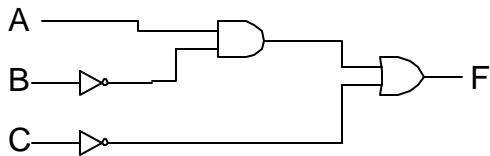
$F = AB' + C$



A	B	C
0	0	0
1	0	0
1	1	0
0	1	0

A	B	C
1	0	0
1	0	1
A	B'	

A	B	C	AB'	C	F	Minitérmino
0	0	0	0	1	1	0
0	0	1	0	0	0	1
0	1	0	0	1	1	2
0	1	1	0	0	0	3
1	0	0	1	1	1	4
1	0	1	1	0	1	5
1	1	0	0	1	1	6
1	1	1	0	0	0	7



**CIRCUITOS COMBINATORIOS :**

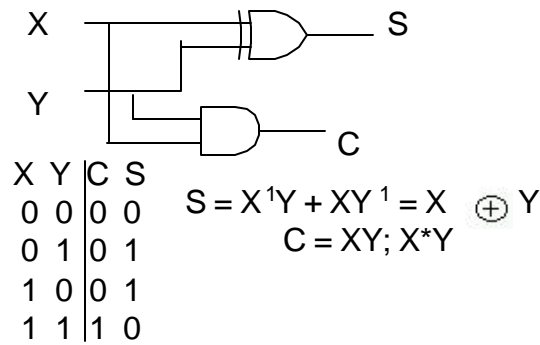
Un circuito combinatorio es un arreglo conectado de compuertas lógicas con un conjunto de entradas de compuertas lógicas con un conjunto de entradas y salidas. En cualquier tiempo dado los valores binarios de las salidas son una función de la combinación de los ceros y unos de las entradas. Los circuitos combinatorios transforman los datos binarios de las entradas en los valores binarios de las salidas que se requieren .



Los circuitos combinatorios se usan para generar decisiones de control binario o para proporcionar funciones digitales que se requieren en el procesamiento de datos.

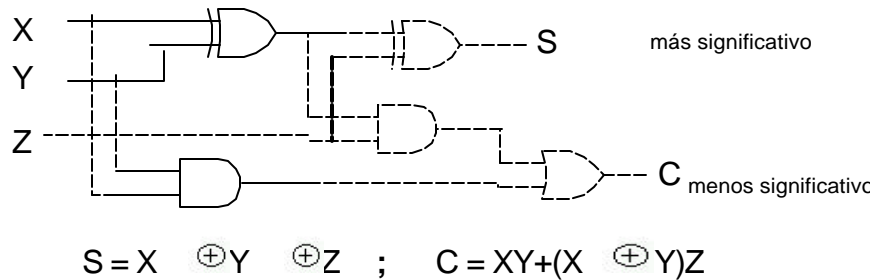
**SEMI SUMADOR (S.S.) :**

Es una función aritmética capaz de realizar la suma de dos dígitos binarios, tiene dos variables de entrada reconocidas : Sumando y Sumador; y dos variables de salida denominadas Suma (S) y Acarreo (C).



**SUMADOR COMPLETO :**

Circuito combinatorio de tipo aritmético capaz de realizar la suma binaria de 3 bits, consta de tres entradas y dos salidas. En sus tres entradas, como "x, y, z"; donde "x, y" representan los dos bits de entrada y "z" representa el bit de acarreo en la posición previa, las salidas serán suma y acarreo.

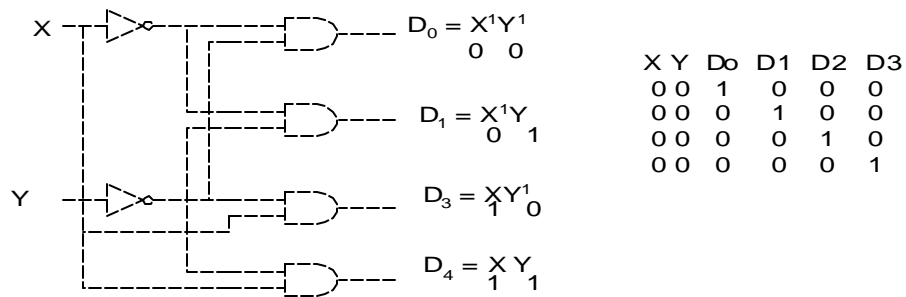


**DECODIFICADOR :**

Es una función digital que convierte información binaria de una forma codificada a otra; por ejemplo : un decodificador de código binario decimal(BCD), de 7 segmentos, convierte un dígito decimal en código binario decimal en 7 salidas para la selección de un conjunto de segmentos necesarios para exhibir un dígito decimal.

Normalmente los circuitos digitales forman circuitos de N variables de entrada y de dos variables de salida.

Un decodificador tiene tantas salidas como combinaciones tengan sus entradas binarias, las variables de las salidas son mutuamente suspendidas, lo que significa que solo una salida a la vez puede optar por un valor binario 1, esto va a quedar determinado por las variables de entrada.

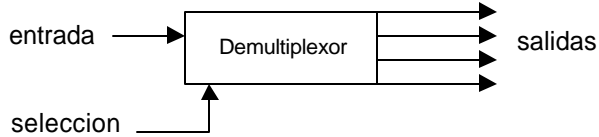


**DEMÚLTIPLEXOR :**

Función digital que recibe información de una sola línea y transmite esta información a  $2^N$  posibles líneas de salida.

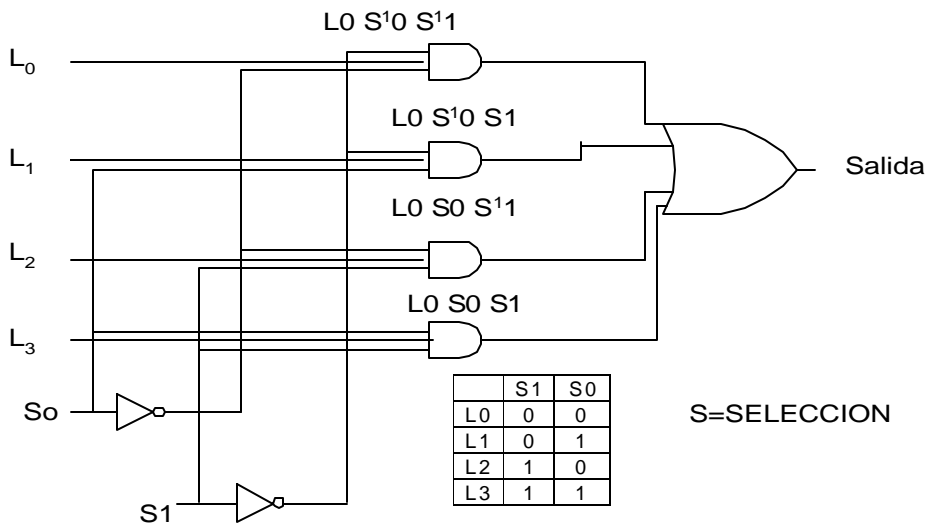
La línea de salida es seleccionada según la combinación de bits de las N líneas de selección.

Funcionan de la misma forma que el decodificador, de ahí su nombre circuito decodificador/demultiplexor.



**MÚLTIPLEXOR :**

Función digital que recibe información binaria de  $2^N$  líneas y la transmite a una línea de salida, la línea de salida que queda habilitada depende de la combinación de los valores binarios de las líneas de selección.



**INTRODUCCIÓN AL TEMA DE FLIPS FLOPS :**

Existen dos tipos de circuitos secuenciales sincrónicos y asincrónicos. Los circuitos secuenciales sincrónicos usan elementos de almacenamiento llamados flips flops que permiten cambiar los valores binarios solamente en determinados instantes discretos de tiempo. Un circuito secuencial asincrónico es aquel cuyas salidas van a depender de la combinación de las variables de entrada y por lo tanto puede afectarse en cualquier instante.

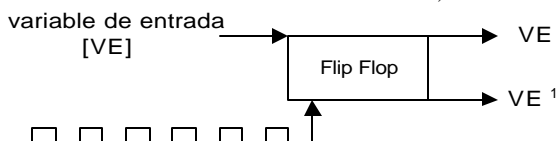
Los sistemas asincrónicos normalmente usan realimentación en sus rutas de datos, mientras que los otros sincronizan su operación mediante un dispositivo llamado generador de pulsos de reloj (CP).

Los cuales son distribuidos a través de todo el sistema al mismo instante y afectando a cada Flip Flop (FF) a la vez.

**DEFINICIÓN DE FLIP FLOP :**

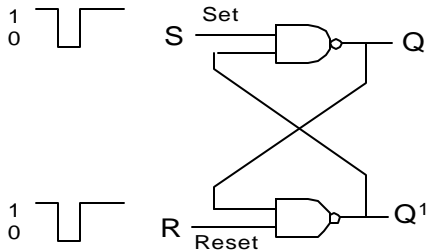
Celda binaria, capaz de guardar un bit de información y contienen dos salidas, una para el valor normal del bit guardado en él y la otra para el valor complementado del mismo.

Un FF mantiene su estado binario, hasta tanto no lo alcance un CP que conmute su estado.



**FLIP FLOP BÁSICO O DE PESTILLO (FFP):**

Son asincrónicos.



Se construye a partir de dos NAND o dos NOR conectados frente a frente. Las conexiones cruzadas de la salida de una compuerta con la entrada de la otra construye el camino de realimentacion.

Cada FFP tiene dos salidas Q y Q¹ y dos entradas S y R.

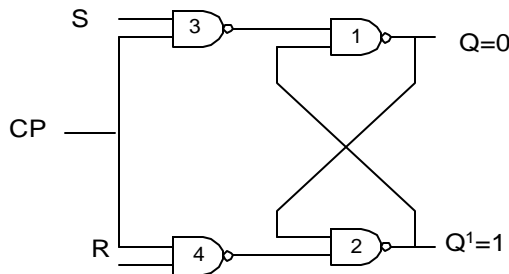
El pestillo opera normalmente con ambas entradas en 1 a no ser que el circuito tenga que cambiarse. Si se aplica un 0 en Set hace que Q sea 1 y Q¹ sea 0, las salidas del circuito no cambian cuando Set regresa a 1.

Un 0 aplicado momentáneamente en Reset hace que Q sea 0 y Q¹ sea 1. El estado del 0 se toma siempre del valor normal de Q, cuando Q sea 1 el FF esta Set y cuando Q sea 0, se dice que Q guarda 0 y el FF esta Reset.

El circuito de pestillo presenta una condición indeseable si las dos entradas se ponen en 0. Ya que ambas salidas Q y Q¹ son = 1, esta condición no tiene ningún significado lógico para la operación del FF, si ambas entradas regresan a 1 el estado del FF es impredecible.

**FLIP FLOP SINCRONICO (FF R/S):**

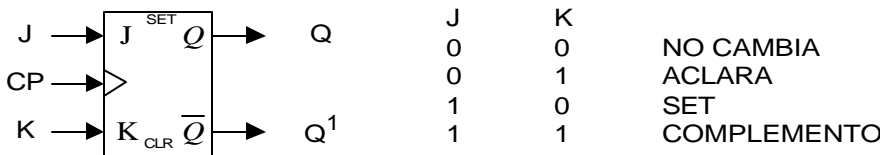
Es un FF construido básicamente con un FFP, añadiendo dos compuertas NAND a sus entradas haciendo que estas respondan a la ocurrencia de un CP.



Las salidas de las compuertas 3 y 4 están siempre en 1 cuando el CP es 0, independientemente que tengan S y R. Cuando el CP va a 1 permite que el valor de S y R alcancen las compuertas 1 y 2.

**S = S = 1 y R = 0; CP = 1.      R = R = 1 y S = 0; CP = 0.**

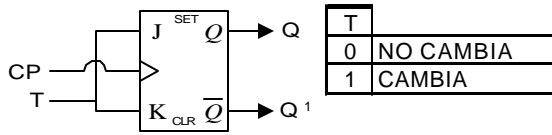
**FLIP FLOP J K:**



Es una mejora del FF R/S; donde J = R y K = S. Cuando J y K = 1 el CP conmuta las salidas a su estado complementario, J = S y K = R queda resuelto que si ambas entradas son = 0 el resultado no cambia.

**FLIP FLOP T (TOGGLE) :**

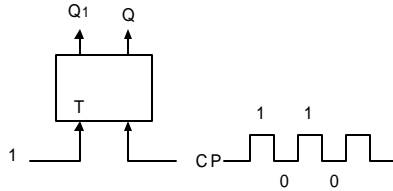
Se comporta igual al FFJK, excepto por que sus dos entradas están juntas.



T	Q	Q'
0	NO CAMBIA	
1	CAMBIA	

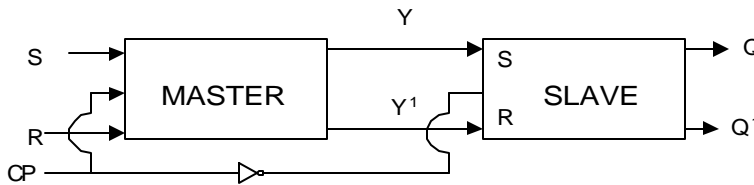
T = 0 => J = K = 0  
 NO CAMBIA  
 T = 1 => J = K = 1 => CP  
 CONMUTA

Ejemplo :



CP	T	Q	Q'
1	1	1	0
1	1	0	1
1	1	1	0

**FLIP FLOP MASTER SLAVE :**

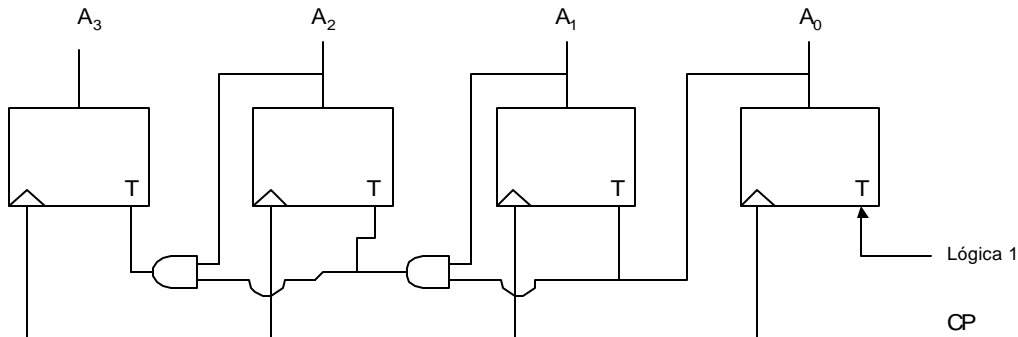


Es un paquete constituido por dos FF independientes y un inversor. El primero actúa como Master y el otro como Slave. Cuando el CP es 0 la salida es 1, lo que habilita al Slave y en ese momento las salidas Y y Y', alcanzan a Q y Q'. Mientras que Master esta deshabilitado.

Cuando el CP es 1 entonces las entradas externas (R y S) se transmiten al Master y Slave esta deshabilitado. Mientras el FF Master esta aislado, se previene que las entradas lo afecten y el FF Slave se pone = al FF Master.

**CONTADORES BINARIOS (CB):**

Diagrama de un contador binario de cuatro bits.



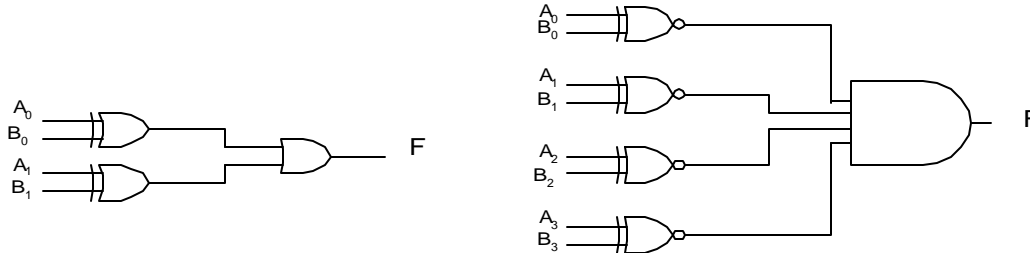
A3	A2	A1	A0	TIEMPO
0	0	0	0	T0
0	0	0	1	T1
0	0	1	0	T2
0	1	0	0	T3
1	0	0	0	T4
1	1	1	1	T5

Ronald Alpizar Porras #950114

Es un arreglo con n FF y compuertas que siguen una secuencia de estados de acuerdo a la cuenta binaria desde 0 hasta  $2^n - 1$ . Un contador se construye con FF que presentan condición de complementación, ej: JK y T, estos FF son contadores facultados para contar de 0 a 1, ordenados secuencialmente y con el uso de AND estos FF permitirán una secuencia de conteo binario que el FF anterior (o sea el de orden mayor) haya sido complementado.

**COMPARADORES :**

Un comparador se compone de n compuertas XOR(OR) ó XNOR(AND) y comparan n bits.



**UNIDAD ARITMÉTICA LÓGICA (ALU) :**

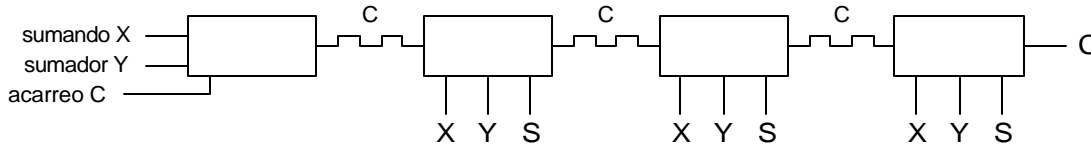
Un ALU es un circuito digital que hace un conjunto de microoperaciones aritméticas y lógicas. La ALU tiene un conjunto de líneas de selección de cada una de las microoperaciones. Un ALU se compone de dos circuitos independientes, aritméticos y lógicos.

**NOTA 1 :**

complemento a 1 = 1 0 1 0 1 1  
 c a 1 = 0 1 0 1 0 0  
 complemento a 2 = 0 1 1 0 1  
 c a 2 = 1 1 1 1 1

**NOTA 2 :**

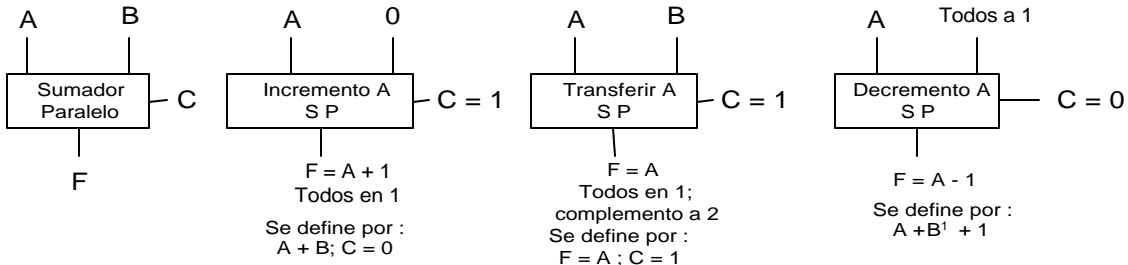
Sumador paralelo : Un sumador paralelo de n bits es un arreglo de n sumadores completos concatenados entre sí, como n sea el numero de bits.



**CIRCUITO ARITMÉTICO :**

Se construye con sumadores paralelos binarios; donde un conjunto de entradas externas reciben el numero binario A, otra el numero binario B y el acarreo en la tercera entrada llamada C.

Algunas funciones aritméticas son ejecutadas a nivel de microoperación.



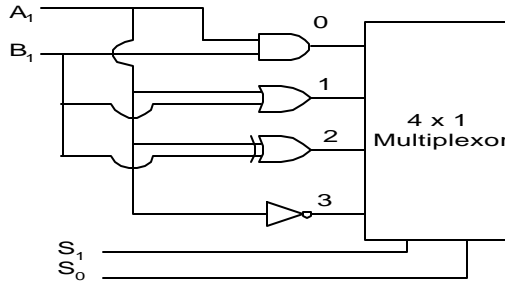
5	A =	101	101
-2	B =	010 = B' = 101	101
3 ^	C =	1	010
		^ 011	<u>1</u> C
			^ 011

A= 14	1110
B=+-6	110
8	<u>1000</u>

**CIRCUITOS LÓGICOS (CL):**

Las microoperaciones lógicas manipulan los bits de los operandos tratando cada uno de los bits como un valor binario.

Esencialmente el CL desarrolla 16 microoperaciones lógicas a partir de 4 compuertas básicas AND, OR, XOR e INVERSORES, además usa un multiplexor al cual se conectan las salidas de las compuertas y por medio de dos líneas de selección se garantiza a única salida a la vez.

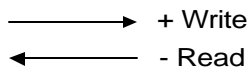


S1	S0	SALIDA	MICROOPERADOR
0	0	$F = A \wedge B$	AND
0	1	$F = A \vee B$	OR
1	0	$F = A \oplus B$	XOR
1	1	$F = A'$	INVERSOR

**MEMORIAS :**

ROM (Read Only Memory) : Son en general un tipo de memoria que solo se graba una vez y que solo puede ser leída.

RAM (Random Acces Memory) : Se leen y modifican tantas veces como se requiera. Desde el punto de vista de suministro y requerimiento de energía, se clasifican en Volátiles. y No Volátiles. Las RAM son volátiles y requieren constante suministro de energía. Desde el punto de vista de preservación de la información almacenada encontramos memorias con la propiedad de lectura destructiva, como eran las de núcleo magnético donde al almacenar información se usaba una dirección de magnetización y la lectura se hacía invirtiendo esta posición.



Las memorias ROM modernas no tienen esta propiedad.

PROM (Programable ROM) y EPROM (Eraseable Programable ROM).

Las memorias RAM modernas se clasifican en dos tipos DRAM (Dinámica RAM) y RAM (Estáticas).

Las DRAM son más caras y más rápidas, pero requieren ser refrescadas en determinados ciclos de reloj.

La Estática no ocupa ser refrescada una vez que la energía es guardada y son volátiles.

**NOTAS :**

Caché : intermedio, entre el disco duro y RAM; de alta velocidad, cuya finalidad es acelerar el proceso de lectura del disco duro. Hay cache de disco, instrucciones y datos. El de instrucciones es el que tiene instrucciones de programa ya establecidas en él y que están listas para ser utilizadas en cualquier momento.

Virtual : Utilizan un dispositivo de almacenamiento secundario para simular una extensión de memoria RAM.

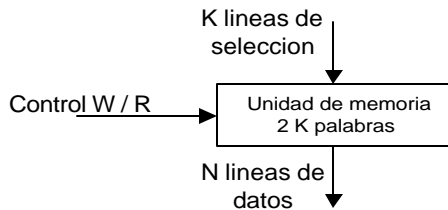
Memoria RAM de vídeo (VRAM) : Hace tratamiento directo de imagen sin necesitar el procesador.

**UNIDADES DE MEMORIA :**

Una unidad de memoria es compuesta por palabras y cada palabra almacena un registro de memoria. El concepto de longitud de palabra depende del numero de líneas para datos.

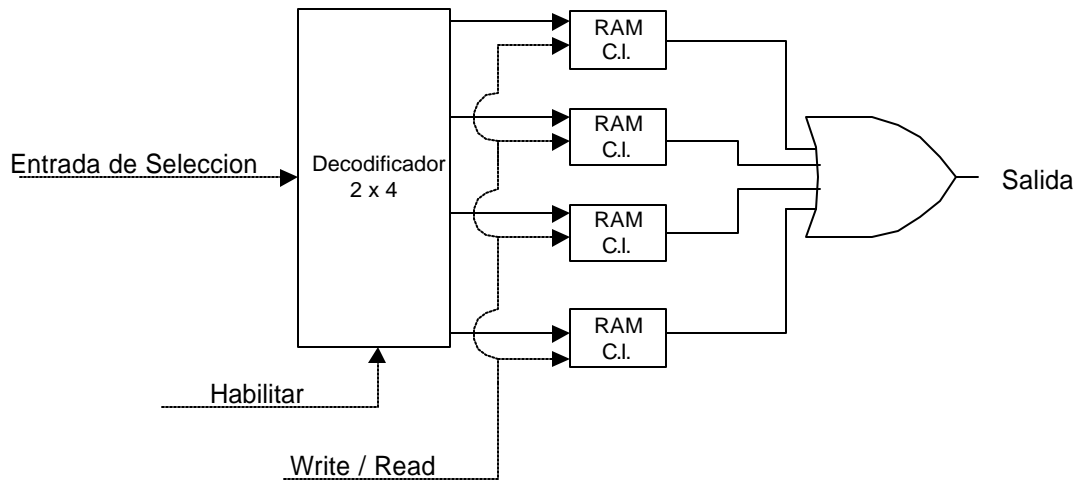
Las pentium manejan palabras de 64 bits. Pueden llevar o traer una hilera de 64 bits desde y hacia la menor. La comunicación de una menor con su entorno, se hace a través de las líneas de control, dirección, entrada y salida de datos, las líneas de control pueden leer o escribir en memoria. Las líneas de dirección indican el numero de palabras a leer o escribir.

Si una unidad de memoria cuenta con K líneas de selección y N líneas de datos, tiene  $2^K$  palabras de N bits cada uno y numerados hasta  $2^{K-1}$ .



Las memorias del computador están en rango de 1024 palabras = 1 K por lo que ocupan almacenar al menos 10 bits de dirección.

Las unidades de memoria RAM pueden crecer concatenando paquetes de RAM en circuitos integrados, todos concatenados y con solo una línea para el control, para las operaciones leer o escribir, además utilizan líneas de habilitación para seleccionar el bloque de memoria.



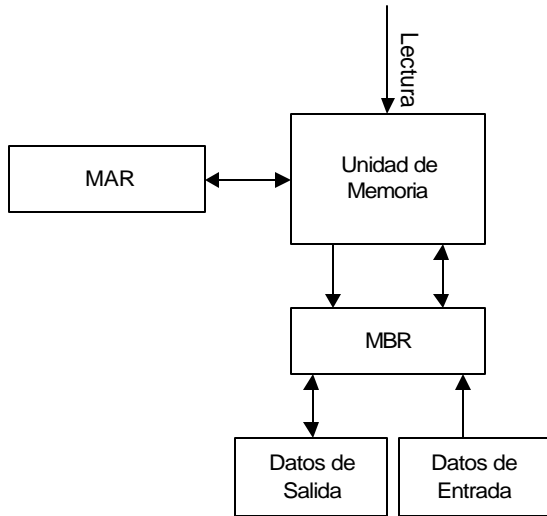
Las unidades de memoria no pueden comunicarse directamente con el procesador por lo que para realizar operaciones se apoyan el uso de dos registros.

**1-REGISTRO DE DIRECCION DE MEMORIA(MAR) :**

El MAR contiene la dirección de memoria de la palabra a leer o en la cual se va a escribir.

## 2-REGISTRO SEPARADOR DE MEMORIA(MBR) :

El MBR deberá contener los datos a escribir en la memoria o los que se traen de aquella dirección, especificada por el MAR.



### OPERACIONES DE LECTURA :

MBR maneja los datos de entrada y salida de memoria. Para hacer una operación de lectura; consiste en sacar una palabra de una unidad de memoria fuera del entorno de la memoria.

1. Se transfieren los bits de dirección al MAR.
2. Se activa la señal de control de lectura "read".
3. El contenido de la información binaria contiene la palabra que especifica el MAR, luego se pasa al MBR y el contenido de la dirección no cambia.
4. MBR se puede comunicar con el registro procesador (AC).

### OPERACIONES DE ESCRITURA :

Este es un mecanismo por medio del cual se pueden pasar los datos hacia el interior de la unidad de memoria.

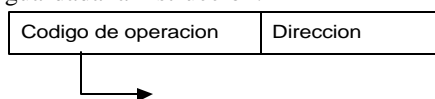
1. Se transfiere la dirección al MAR(donde se lee o escribe).
2. Se activa la señal de control de escritura "write".
3. Se pasan los bits de datos al MBR.
4. La información binaria almacenada en MBR se guarda en el numero de palabras especificadas por el MAR.
5. El contenido de esa palabra se actualiza.

### ORGANIZACIÓN DEL COMPUTADOR :

Un computador es un sistema digital de propósito general el es capaz de ejecutar un numero determinado de diversas microoperaciones y además de ser instruido sobre la secuencia de operaciones especificas que debe ejecutar.

Un programa es un conjunto de instrucciones que especifican operaciones, operandos y la secuencia con que debe desarrollarse el procesamiento.

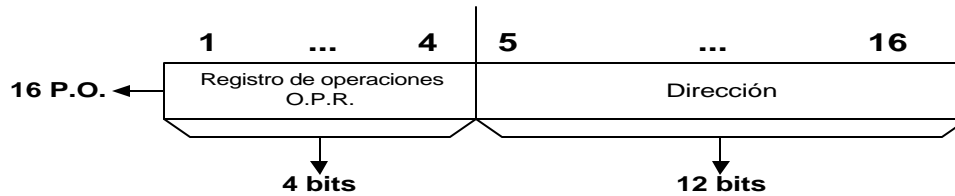
Un código de instrucción es un conjunto de bits que le dice al computador que debe hacer, una operación especifica y se compone de dos partes : el código de la operación y la dirección donde esta guardada la instrucción.



Un código de operación es un grupo de bits que le dice al computador que suma, resta, desplaza, complementa, multiplica o divide.

Una operación es la parte de una instrucción almacenada en la memoria ROM y que contiene un código que la unidad de control interna interpreta y hace una serie de microoperaciones a nivel de los registros internos del computador, por eso suele llamársele Macrooperación.

Son el conjunto de código de instrucciones que apuntan a los registros y/o palabras donde están los operandos o donde se deben almacenar ciertos resultados.



Usa 12 bits para especificar una dirección por lo tanto es capaz de manejar bloques de unidades de memoria de 4 K, ( $12^{12} = 4096$ ), los restantes cuatro bits se usan para especificar una de las 16 posibles operaciones.

Las operaciones se ejecutan entre el operando y el registro procesador (AC).

No siempre los 12 bits van a tener ese uso, ya que hay instrucciones que no necesitan de una operación de memoria. Aclarar AC, conmutar AC, sumar AC, y que actúan directamente sobre él.

De las instrucciones que hacen referencia a operandos en memoria; hay dos tipos :

1. Instrucciones de Dirección Directa(I.D.D.) : son aquellos que indican en los 12 bits de dirección la dirección donde esta guardado el operando.
2. Instrucciones de Dirección Indirecta(I.D.I.) : son aquellos que usan los 12 bits de dirección para indicar otra dirección que apunta a donde esta el operando.

Para que el computador reconozca esta diferencia, divide los cuatro bits del código de operación en dos partes; un primer bit de MODO y tres bits de Código de Operación, el bit de MODO es I, y puede optar por dos valores; si es igual a 0 es I.D.D., pero si es igual a 1 es I.D.I..

### INSTRUCCIONES DEL COMPUTADOR :

Son normalmente guardadas en direcciones de consecutivas de memoria.

La memoria lee una instrucción de una dirección específica, la decodifica, la ejecuta y continúan en secuencia.

Se necesita un contador para calcular cada vez la dirección siguiente, para ello usa un registro de 12 bits(se hace con 12 FF) que se denomina Program Counter (P.C.) y que es un contador de 12 bits.

Las palabras de memoria no pueden comunicarse directamente con el registro acumulador por lo que usa un registro de 12 bits llamado MAR y otro de 16 llamado MBR.

Además del AC de 16 bits se usa un FF llamado 'E' que es una extensión del registro AC, y se encarga de recibir el acarreo final durante el acarreo de suma y durante el desplazamiento, intervienen I y O.P.R. y cuando hay desbordamiento entra 'E'.

### INSTRUCCIONES DE RAMIFICACIÓN O BIFURCACIÓN :

Un computador ejecuta las instrucciones secuencialmente a no ser que encuentre una instrucción de bifurcación, una instrucción de este tipo contiene una parte de operación que exige una transferencia a una instrucción no consecutiva en la memoria, esta nueva dirección es la que se carga en P.C. para que sea la siguiente instrucción en ejecutarse.

Un computador tiene tres tipos diferentes de formatos de instrucciones :

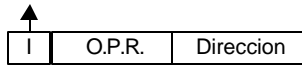
1. Instrucción de referencia a memoria.
2. Instrucción de referencia a registro.
3. Instrucción de referencia a E / S.

Cada instrucción de las 25 básicas que ejecuta un computador se codifica agrupando los 16 bits en 4 direcciones Hexadecimales y se usa un símbolo de 3 letras para abreviar su significado.

**INSTRUCCIONES DE REFERENCIA A MEMORIA :**

Son instrucciones que usan los últimos 12 bits para especificar una DIRECCION donde esta el operando, por lo tanto su primer bit va a ser representado por I, que es el bit de MODO.

Bit 1 / 0

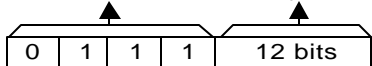


**INSTRUCCIONES DE REFERENCIA A REGISTRO :**

Especifican una operación o prueba de los registros AC ó E. No usa un operando de memoria por lo que los últimos 12 bits se usan para codificar la operación a ejecutar.

Este código se ensamblara con un 1 y once 0's, se reconocen además por que sus primeros 4 bits ensamblan la cadena 0 1 1 1 lo que representa 7(el primer bit ya no funciona como I, sino que se ensambla una cadena de 4 bits ).

7 hexadecimal Un 1 y once 0's

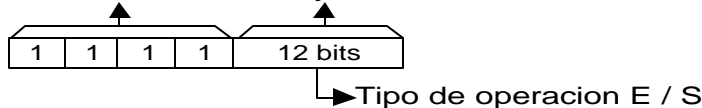


**INSTRUCCIONES DE ENTRADA Y SALIDA (E / S) :**

No requieren una DIRECCION de memoria por lo que los últimos 12 bits se usan para codificar una operación de E / S. Codifican también con un 1 y once 0's y especifica el tipo de operación de E / S.

Los primeros 4 bits se ensamblan como una cadena de bits 1 1 1 1 = F = 15 hexadecimal.

F hexadecimal Un 1 y once 0's



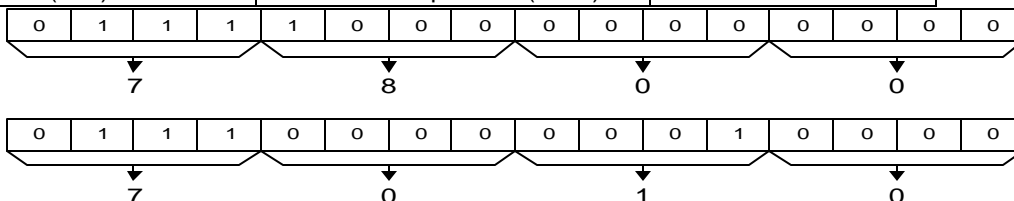
**Ejemplo :**

Referencia a memoria :

- AND ⇒ contiene a la palabra en memoria ⇒ AC
- ADD ⇒ suma palabras de memoria a AC

**REFERENCIA A REGISTRO :**

Nombre mnemónico	Descripción	Código Hexadecimal
CLA ⇒	Clear AC	7800
CLE ⇒	Clear E	7400
CMA ⇒	A'C' ó AC	7200
CME ⇒	combina E E' = E	7100
CIR ⇒	circular a la derecha (right)	7080
CIL ⇒	circular a la izquierda (left)	7040
INC ⇒	incrementar AC	7020
SPA ⇒	salta si AC es positivo	7010
SNA ⇒	salta si AC es negativo	7008
SZA ⇒	salta si AC = 0	7004
SZE ⇒	salta si E = 0	7002
HLT(Halt) ⇒	detiene el computador (reset)	7001

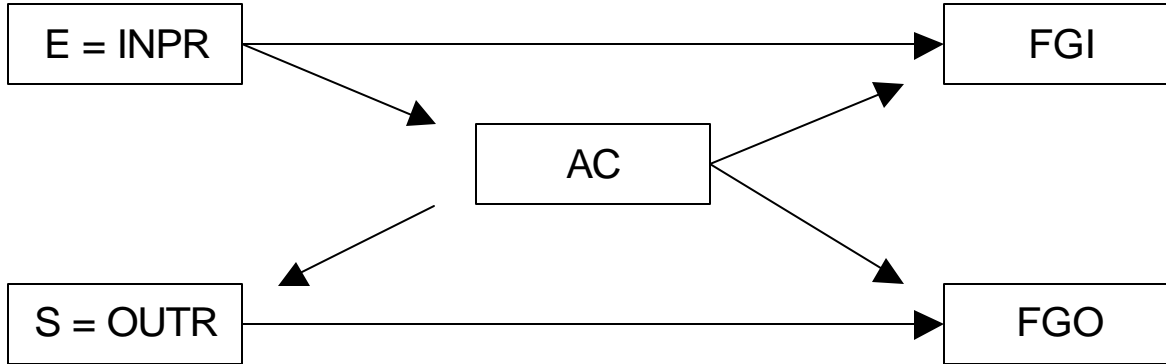


**INSTRUCCIONES E/S :**

Tienen formato FXXX

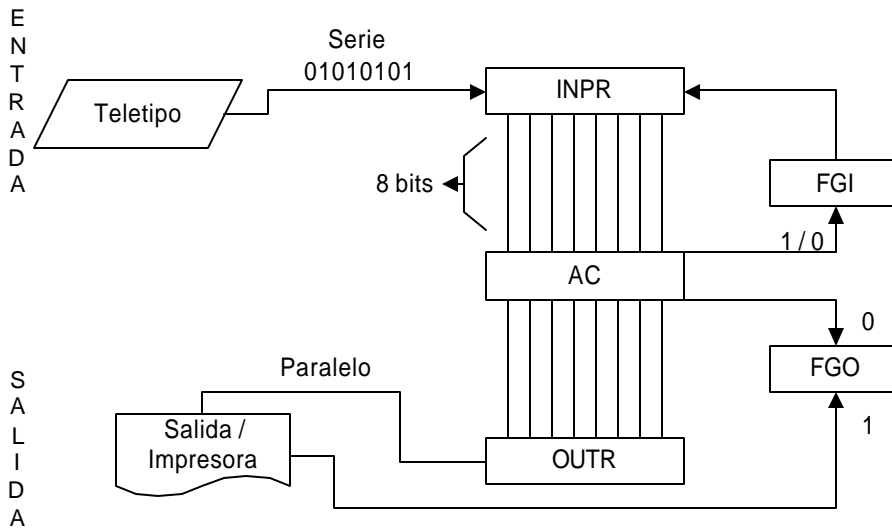
**REGISTRO E/S :**

Los registros de E / S constan de 8 bits cada uno y son los siguientes : E = INPR y S = OUTR, se usan para administrar los datos desde y hacia AC, cuando se produzca alguna instruccion de entrada y salida.



Cuando hay una información en el registro INPR que ha sido aceptada por AC se activa una bandera ó FF llamada FGI, cuando AC tiene disponible una salida para el registro OUTR se activa FGO.

**CONTROL DE E/S :**



La información que proviene de un teclado se reciben en series de 8 bits y es guardada en registros de entrada. La información en serie que va a la impresora se guarda en un registro de salida, estos registros se comunican entre sí en forma serial pero con el registro procesador (AC) se comunica en paralelo.

El registro de entrada se denomina INPR y consta de 8 bits y apoya en su función en un FF ó bandera de control que se denomina FGI. Esta bandera se activa cuando la información está disponible en el dispositivo de entrada y se aclara cuando la información es aceptada por el computador.

Inicialmente FGI esta de aclarado, cuando se pulsa una tecla el código de 8 bits es transferido a INPR y la bandera pasa a 1. Siempre que este estado se mantenga (1) la información contenida en INPR no se puede modificar, pulsando otra tecla.

El registro procesador(AC) determina que FGI esta en 1 y procede a pasar en forma paralela los datos de INPR a AC e inmediatamente aclara a FGI de manera que INPR puede recibir nueva información.

La salida funciona en similar usando el registro OUTR y una bandera llamada FGO, solamente que al inicio  $FGO = 1$ .

AC verifica que esta condición se mantenga y se transfiera en forma paralela al registro OUTR en ese momento FGO es aclarado(0). Terminada la tarea de impresión FGO vuelve a 1.

Un elemento adicional que interviene en el proceso de E / S es el FF llamado IEN o de Interrupción ó Habilidadación el cual puede aclararse o habilitarse por medio de dos interrupciones. Cuando este FF esta aclarado FGI y FGO no pueden interrumpir el computador. Esto permite al programador decidir que programa pueda o no ser interrumpido.

#### INSTRUCCIONES DE E / S :

Permiten pasar información desde y hacia AC y verificar las banderas y controlar el FF de Interrupción/Habilidadación (IEN).

Además inicia su código con una cadena binaria 1 1 1 1 y codificada con un 1 y once 0's.

SÍMBOLO	CÓDIGO HEXADECIMAL	DESCRIPCION
INP ⇒	F800	Entra al carácter AC; AC + INPR; FGI = 0
OUT ⇒	F400	Saque al carácter AC; OUTR + AC; FGO = 0
SKI ⇒	F200	Salta en la bandera de entrada; si(FGI=1) entonces (PC ← PC+1)
SKO ⇒	F100	Salta en la bandera de salida; si(FGO=1) entonces (PC ← PC+1)
ION ⇒	F080	Interrupción ON; IEN ← 1
IOF ⇒	F040	Interrupción OFF; IEN ← 0

#### NOTA :

SKI y SKO verifican el estado de la bandera respectiva y produce un salto si la bandera es igual a 1, esta instruccion es normalmente, una instruccion de ramificación para retornar y verificar las banderas de nuevo, si la bandera es igual a 0 no se salta, en caso contrario la instruccion de ramificación se salta y se ejecuta una instruccion de entrada / salida.

#### COMPONENTES BÁSICOS DEL DISEÑO DEL COMPUTADOR :

Un computador consta de :

- Una unidad de memoria
- Un teclado, maquina de escribir ó teletipo.
- Un generador de pulsos de reloj maestro(C.P.M.)
- 8 registros
- 8 FF
- 3 Decodificadores
- Un numero de compuertas de control lógico

La memoria y el teclado son unidades estándar que se pueden comprar como productos terminados. El C.P.M. es una fuente común de pulsos de reloj generados en forma periódica, los cuales se distribuyen a todo el sistema afectando a cada registro y FF al mismo tiempo.

Los decodificadores y los registros están disponibles en CI's de M.S.I y los FF están disponibles en CI's de S.S.I.

#### REGISTROS QUE TIENE EL COMPUTADOR :

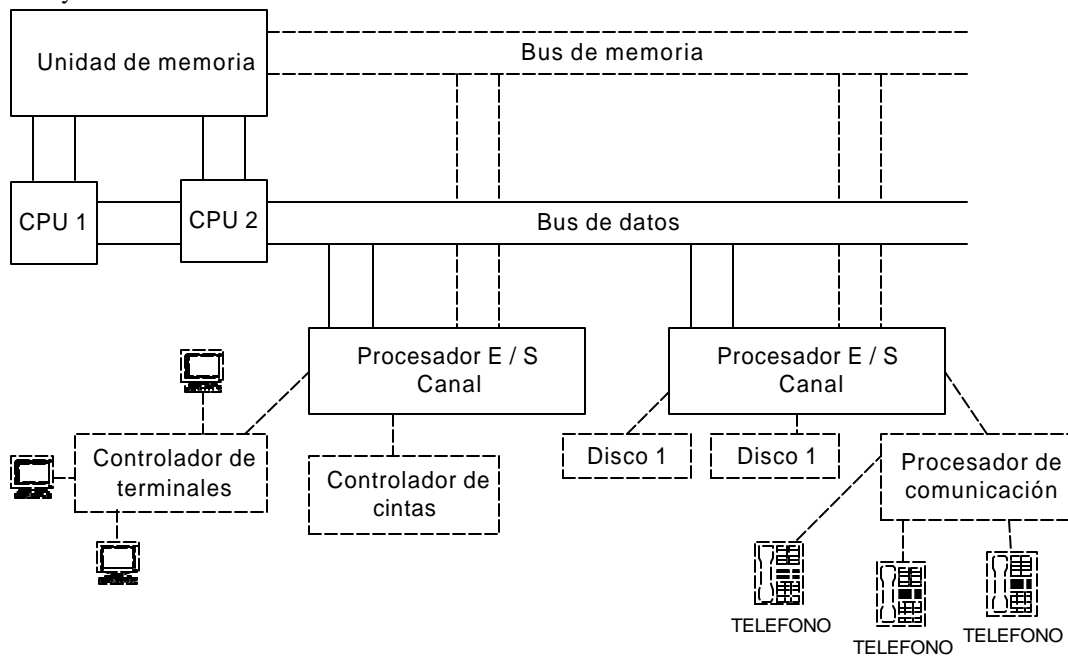
1. AC ⇒ Registro procesador / Acumulador
2. PC ⇒ Program Counter,  $PC \leftarrow PC + 1$
3. MAR ⇒ Registro de dirección de memoria
4. MBR ⇒ Registro separador de memoria
5. OUTR ⇒ Registro de salida
6. INPR ⇒ Registro de entrada
7. OPR ⇒ Registro de código de operación
8. SC ⇒ 2 bits ⇒ Contador de secuencia (de sincronización y control)

**FLIPS FLOPS QUE TIENE EL COMPUTADOR :**

1. FGO ⇒ Bandera de salida
2. FGI ⇒ Bandera de entrada
3. I ⇒ Bit de modo, 1 / 0
4. E ⇒ Extensión del AC
5. IEN ⇒ Interrupción / Habilitación
6. S ⇒ Start / Stop ⇒ si S = 0 ⇒ No CP
7. " F " y " R " ⇒ F y R son los encargados de controlar los ciclos de estado del computador, éstos pueden tener tres estados : Fetch(decodificando la información), Ejecución, e Indirecto(acceso a nuevas instrucciones)

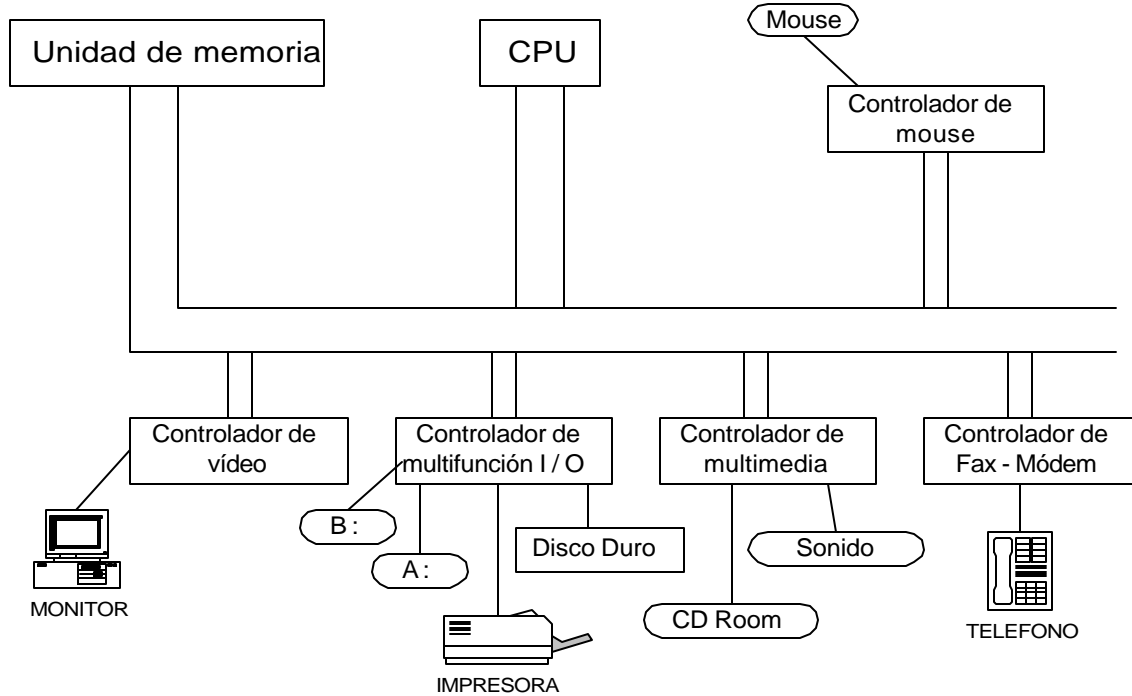
**ARQUITECTURA E / S :**

Los computadores modernos usan básicamente dos modelos para representar la E / S de los datos :  
1- Para macrocomputadores donde podemos encontrar la unidad de memoria, uno ó tal vez múltiples CPU's, uno ó más procesadores especializados en E / S, denominados canales, todos interconectados por los buses de memoria y E / S.



2-Computadores personales :

Tienen un único bus sobre el cual van los datos, señales de control y señales de dirección. Tiene una unidad de memoria, un CPU y un dispositivo de E / S.(Un controlador específico más el dispositivo como tal)



**MAQUINAS RISC :**

RISC (Computador Reducido Conjunto de Instrucciones); en contraposición con la arquitectura CISC (Computador con Complejo Conjunto de Instrucciones).

La tendencia a desarrollar maquinas con arquitectura RISC, se da a partir del análisis de muchos programas y diversos lenguajes de alto nivel donde se encontraron las siguientes instrucciones :

TIPO DE INSTRUCCION	%
Asignación	47
Condicionales	23
Llamadas a procedimientos	15
Interacciones	6
GOTO	3
Otras	7

Se notó además que la lentitud del CISC obedece a un gran numero de instrucciones ensambladas a nivel de microcódigo y que hacen tareas / instrucciones del tipo : Registro - Registro, Registro - Memoria, Memoria - Memoria.

Las RISC reducen al mínimo el numero de microinstrucciones y pretende que únicamente haga operaciones de Registro - Registro.

Solo las instrucciones LOAD y STORE, hacen Referencia a Memoria.

La filosofía del RISC se fundamenta en completar una instruccion por cada ciclo de la trayectoria, dado que esto no siempre es posible, el proceso se ejecuta iniciando una instruccion la cual se hace en un procesador paralelo.

	<b>CICLOS</b>										
	1	2	3	4	5	6	7	8	9	10	11
<b>Extracción de instrucciones</b>	1	2	L	4	5	6	S	8	9	10	
<b>Ejecución de instrucciones</b>	1	2	2	L	⓪4	5	6	S	⓪8	9	10
<b>Referencia a Memoria</b>		1			L				S		

El circulo significa que: El compilador determina la posibilidad de error e inserta un NO OP ó NULL para que no se haga un despelote. NO OP y NULL saltan a la siguiente dirección o instruccion para darle tiempo a la actual de que tome algún valor o instrucción.

# **Proyecto Número 1 del curso de arquitectura de computadores.**

**Tema :  
La granja**

### **Planteamiento del Problema**

Un Granjero tiene junto con él, un perro, una cabra y unos repollos. El posee además en su finca, dos graneros, uno norte y otro sur. El granjero, el perro, la cabra y los repollos están todos en el granjero sur. Pero él debe atender otras tareas en el granero norte o en ambos.

El problema es que si se ausenta y deja solo al perro con la cabra, el perro morderá a la cabra . Si deja solos los repollos y la cabra, esta se los comerá. Estas son situaciones de peligro.

**Tabla de Verdad**

<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>Resultado</b>	<b>Minitérmino</b>
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	0	2
0	0	1	1	1	3
0	1	0	0	0	4
0	1	0	1	0	5
0	1	1	0	1	6
0	1	1	1	1	7
1	0	0	0	1	8
1	0	0	1	1	9
1	0	1	0	0	10
1	0	1	1	0	11
1	1	0	0	1	12
1	1	0	1	0	13
1	1	1	0	0	14
1	1	1	1	0	15

A: GRANJERO	B : PERRO	C : CABRA	D : REPOLLO
-------------	-----------	-----------	-------------

**Mapas K**

		CD			
		00	01	11	10
AB	00	0	1	1 <sub>3</sub>	2
	01	4	5	1 <sub>7</sub>	1 <sub>6</sub>
	11	1 <sub>12</sub>	13	15	14
	10	1 <sub>8</sub>	1 <sub>9</sub>	11	10

Tablas de adyacencias :

A	B	C	D
1	0	0	0
1	0	0	1
A	B'	C'	

A	B	C	D
1	0	0	0
1	1	0	0
A		C'	D'

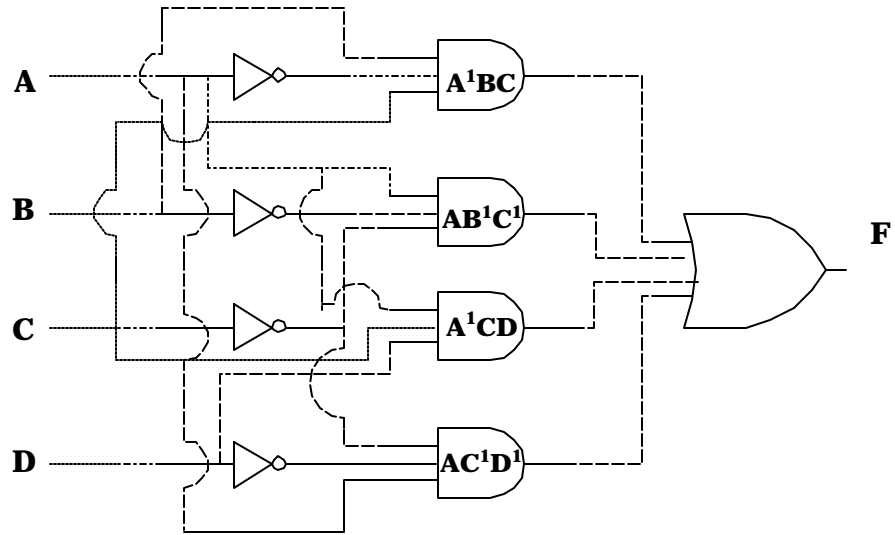
A	B	C	D
0	1	1	1
0	1	1	0
A'	B	C	

A	B	C	D
0	1	1	1
0	0	1	1
A'		C	D

**Situaciones de Peligro**

Minitérmino	Situación
3	La cabra se queda con los repollos
6	El perro se queda con la cabra
7	El perro se queda con la cabra, y ésta con los repollos
8	El perro se queda con la cabra, y ésta con los repollos
9	El perro se queda con la cabra

**Función Simplificada Utilizando las Compuertas Lógicas Respectivas**



**Materiales Requeridos**

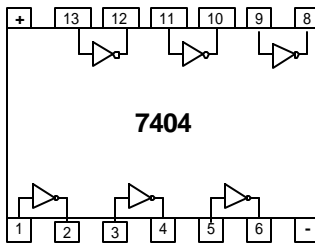
1. 2 Chips de 3 compuertas AND de 3 entradas; serie 74LS11N
2. 1 Chip de 6 INVERSORES de una entrada; serie 7404
3. 1 Chip de 4 compuertas OR de 2 entradas; serie 7432
4. 1 Metro de cable UTP; del tipo par trenzado
5. 4 Switchs; del tipo ON/OFF
6. 1 Led de 3 voltios
7. 2 Baterías de 1.5 voltios
8. 1 Proto-Board

**Costos**

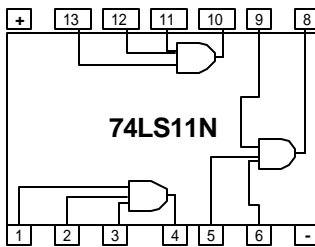
<b>Artículo</b>	<b>Precio</b>
2 chips 74LS11N	¢ 450,00
1 chip 7404	¢ 265,00
1 chip 7432	¢ 315,00
1 led de 3 voltios	¢ 79,65
1 metro de cable UTP	¢ 150,00
4 switchs	¢ 148,60
1 par de baterías	¢ 120,00
<b>Subtotal</b>	<b>¢ 1.528,25</b>
<b>Impuesto de ventas</b>	<b>¢ 231,95</b>
<b>Total</b>	<b>¢ 1.760,20</b>

### Compuertas Lógicas Utilizadas

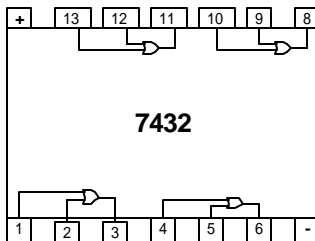
#### Un Chip serie 7404 de 6 inversores

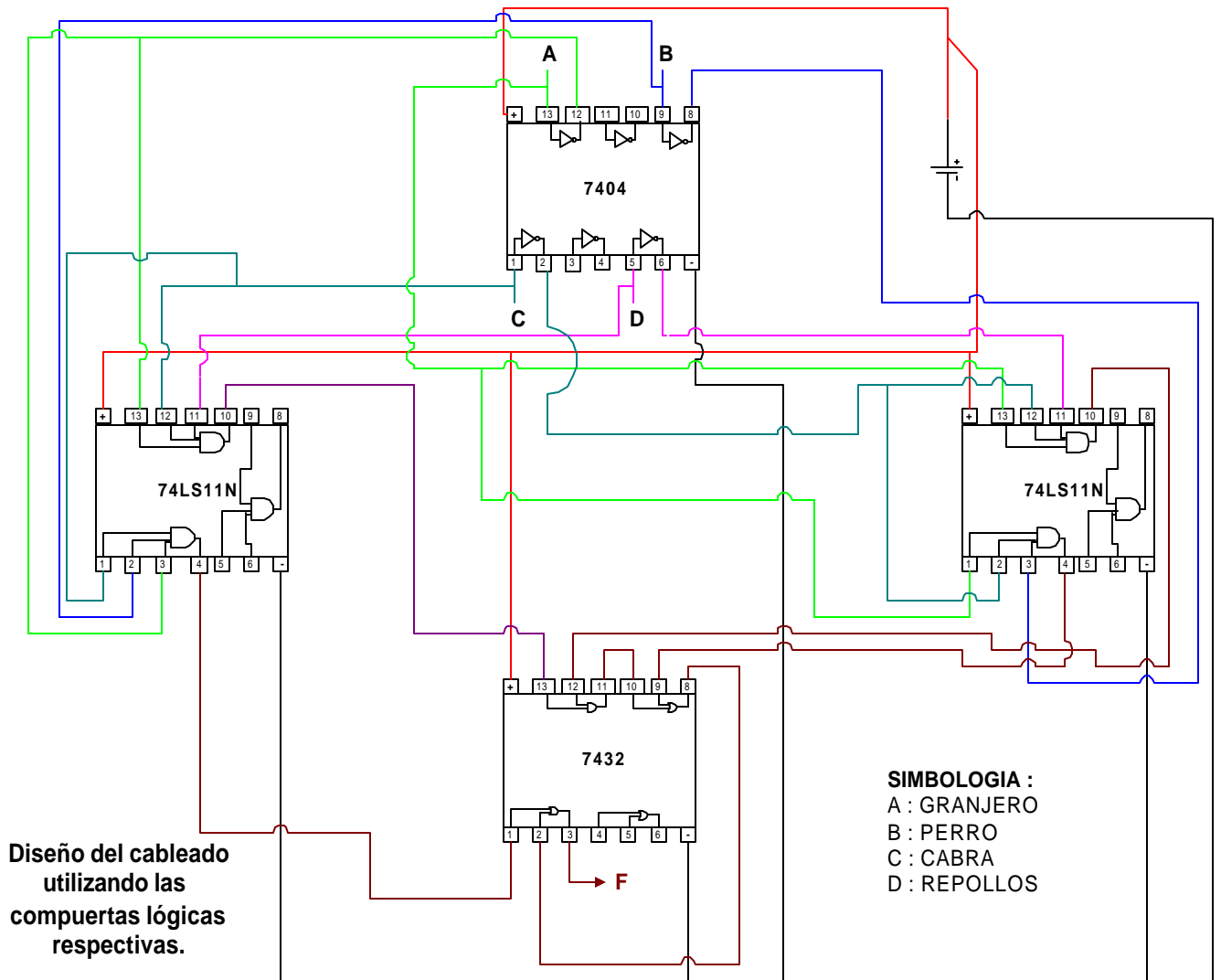


#### Dos Chips serie 74LS11N de 3 compuertas AND de 3 entradas



#### Un Chip serie 7432 de 4 compuertas OR de 2 entradas





La función algebraica utilizada en este cableado es :  
 $F = A'CD + A'BC + AC'D' + AB'C'$

# **Proyecto Número 2 del curso de arquitectura de computadores.**

## **Tema : El Contador Binario de 4 bits**

**Problema :**

Se requiere construir un circuito que sea capaz de funcionar como un contador cíclico, que sea manejado por pulsos de controlados por el usuario. Deberá estar en capacidad de contar como máximo, las combinaciones que permiten 4 bits.(0000H hasta 1111H)

El circuito debe estar diseñado de manera que exhiba una pequeña pantalla o display de 7 segmentos, con el dígito hexadecimal que corresponde a la cuenta binaria. Una vez completado el ciclo con 1111(F hexadecimal) deberá regresar a 0.

Por último, el sistema deberá estar provisto de un mecanismo que funcione como “RESET”, de modo que en el momento que se desee, la cuenta puede regresar a 0.

**Requerimientos :**

Construya el circuito requerido, utilizando componentes electrónicos del grupo TTL ya estudiados en clase, ya que se ajustan a rangos de voltaje de 5 v.

Presente el trabajo sobre un “Proto-Board”, además de los diagramas lógicos de su respuesta.

Indique la lógica de interconexión de los diferentes componentes utilizados en su trabajo.

Para la presentación, se deberá entregar toda la documentación posible, así como desarrollar una corrida de prueba del circuito.

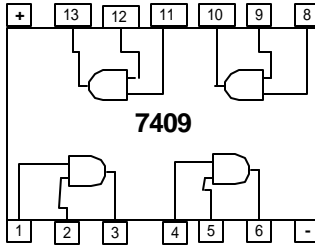
Se recomienda que se presenten informes de avance que permitan orientar el trabajo del estudiante.

### **Materiales Requeridos para la elaboración de este proyecto**

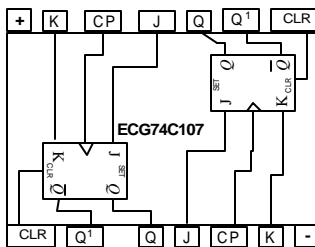
1. 2 Chips Flip-Flop JK; serie ECG74C107
2. 1 Chip decodificador BCD de 7 segmentos; serie ECG7448
3. 1 Chip de 4 compuertas AND de 2 entradas; serie 7409
4. 1 Metro de cable UTP; del tipo par trenzado
5. 2 Pulsadores de corriente
6. 1 Led display de 5 voltios, serie ECG3054
7. 1 Fuente de poder de 5 voltios
8. 1 Proto-Board

### Compuertas Lógicas Utilizadas

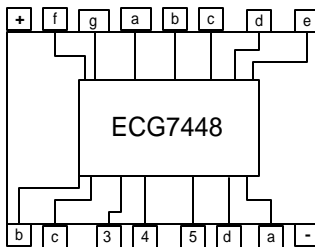
#### Un Chip de cuatro AND de dos entradas; serie 7409



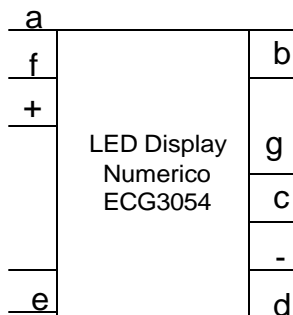
#### Dos Chips Flip Flop JK; serie ECG74C107



#### Un Chip Decodificador BCD de 7 segmentos; serie ECG7448



#### Un Led display numérico; serie ECG3054



## **RESULTADOS**

**El contador binario, que físicamente está bien diagramado y cableado, presenta algunas fallas en su funcionamiento (los números desplegados en el Led aparecen incompletos). Creemos que el fallo se encuentra en el Led o en los pulsadores, no pudimos corregir estas deficiencias por falta de tiempo e investigación para poder averiguar cual es la verdadera causa del problema y así lograr su funcionamiento óptimo.**

