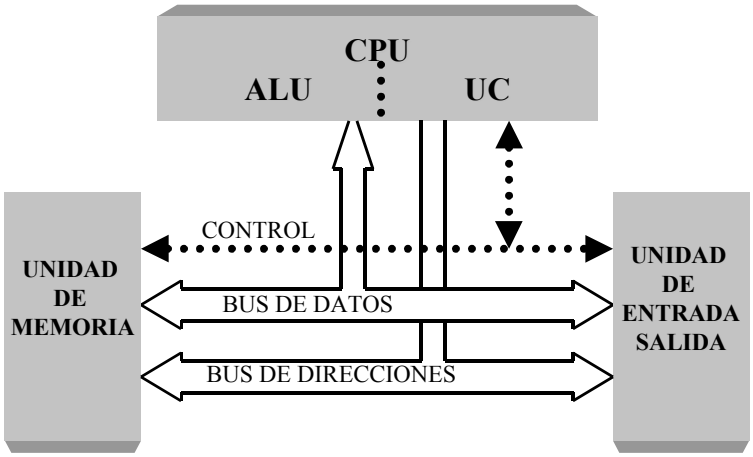
UNIDAD 4

ESTRUCTURA BASICA DE UNA COMPUTADORA

La estructura básica de una computadora está compuesta por cuatro elementos:

- **LA UNIDAD DE PROCESO CENTRAL (CPU)**
- **LA UNIDAD DE MEMORIA**
- **LA UNIDAD DE ENTRADA/SALIDA (PERIFERICOS)**
- **BUSES DE INTERCONECCION ENTRE LAS UNIDADES.**



La **unidad de memoria** es la encargada de almacenar programas y datos.
Su estructura y funcionamiento se desarrolló en la unidad temática anterior.

La **unidad de entrada/salida** se verá en detalle en la unidad temática Nº7. Ahora sólo mencionaremos que dicha unidad se encarga de conectar al cerebro de la computadora (CPU) con los dispositivos externos accesibles al usuario como son el teclado, el monitor, la impresora, etc, por los cuales podemos ingresar o extraer información a o desde la CPU. Estos dispositivos son también llamados Periféricos.

UNIDAD DE PROCESO CENTRAL

La CPU, o centro de control de una computadora, tiene varias funciones, donde las principales son:

- **Ejecutar las instrucciones de los programas almacenados en la memoria del sistema**
- **Controlar la transferencia de datos entre la CPU y los circuitos de memoria y de E/S**
- **Responder a las peticiones de servicio procedentes de los dispositivos de E/S.**

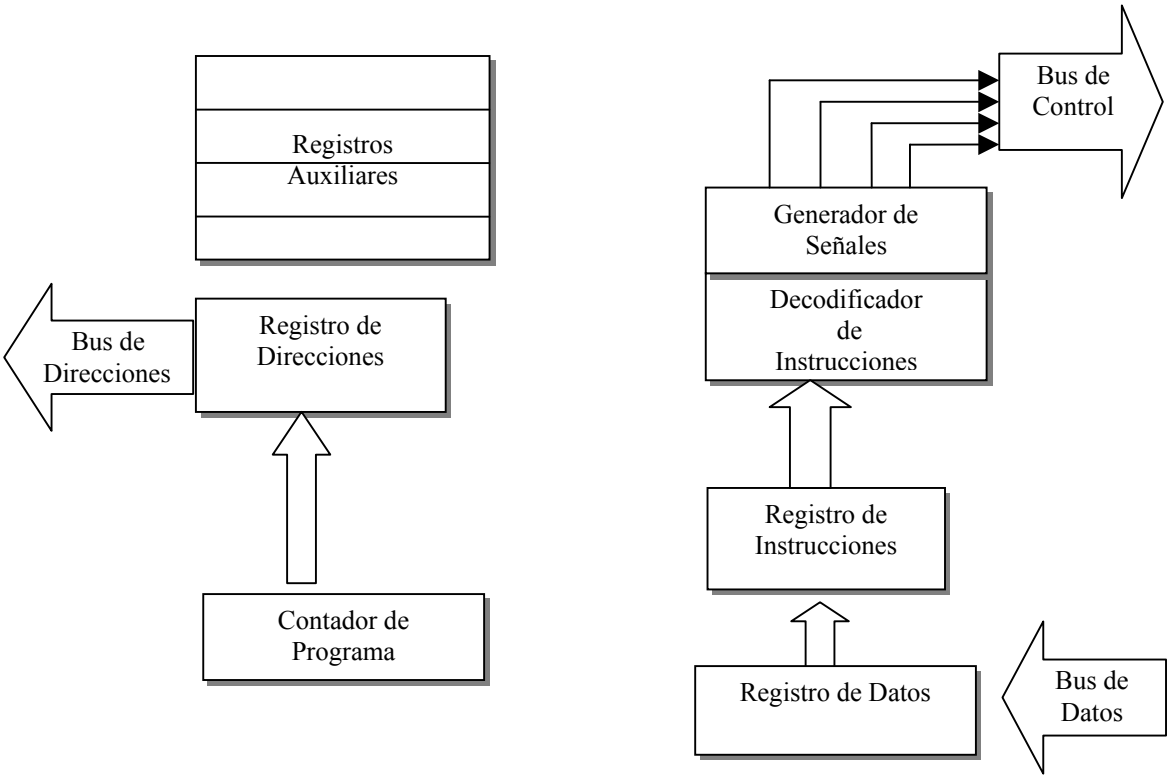
Los programas tienen como objetivo general la realización de diferentes aplicaciones o funciones, limitadas sólo por la imaginación del programador y por la capacidad de la computadora que se dispone.
Para que un programa pueda ser ejecutado por la CPU, debe estar guardado en un determinado lugar de la memoria del sistema y escrito en un lenguaje que la CPU pueda entender. Un programa, básicamente, es una lista de instrucciones que la CPU lee ordenadamente, las interpreta y posteriormente controla su ejecución una tras otra.
La ejecución completa de cada instrucción lleva varios pasos, a saber:

- Leer de la memoria la instrucción correspondiente y guardarla en un registro interno de la CPU
- Identificar (o decodificar) dicha instrucción
- Comprobar si la instrucción necesita datos de la memoria (o de registros internos) y determinar su ubicación
- Buscar los datos correspondientes y traerlos a la CPU
- Ejecutar la instrucción propiamente dicha. En este paso es posible tener que volverse a comunicar con la memoria o con dispositivos de E/S
- Volver al primer paso para ejecutar una nueva instrucción.

Para poder realizar estas tareas la **CPU** normalmente se divide en dos unidades:

- **LA UNIDAD DE CONTROL (UC)**
- **LA UNIDAD ARITMETICA-LOGICA (ALU)**

UNIDAD DE CONTROL
La **UC** controla todas las funciones que realiza una computadora. En la secuencia de ejecución de una instrucción es la encargada de controlar la búsqueda, decodificación y ejecución de la misma.
Los elementos principales que posee dicha unidad están esquematizados en la sig figura:



- REGISTROS
  - Contador de Programa o Puntero de Instrucciones
  - Registro de Instrucciones
  - Registro de datos (o buffer de datos de memoria)
  - Registro de direcciones (de memoria)
  - Registros auxiliares
- SISTEMA DIGITAL DE CONTROL
  - Decodificador de Instrucciones
  - Generador de señales de control.

UNIDAD ARITMETICA Y LOGICA

La ALU es básicamente la calculadora del sistema. Esta unidad se encarga de ejecutar todas las operaciones aritméticas y lógicas necesarias entre los datos que llegan a la CPU.
Los elementos principales que posee dicha unidad son los siguientes:

- REGISTROS
  - Acumulador
  - Registros temporales
  - Registro de estado
- UNIDAD DE CALCULO (O ALU PROPIAMENTE DICHA)

BUSES DE INTERCONECCION ENTRE LAS UNIDADES.

Los buses del sistema son las líneas encargadas de conectar y enviar la información necesaria a los distintos elementos de una computadora. Básicamente existen 3 buses distintos:

- **BUS DE DATOS**
- **BUS DE DIRECCIONES (ADRESS)**
- **BUS DE CONTROL**

El **bus de DATOS** es bidireccional (la información viaja en ambos sentidos) y envía datos o instrucciones entre la CPU, la Memoria y los dispositivos periféricos. El tamaño del bus depende de cada arquitectura, siendo los más comunes de 8,16 ó 32 bits.

El **bus de DIRECCIONES** es unidireccional ya que la información viaja desde la CPU a la memoria o dispositivos periféricos. Su tamaño varía dependiendo de cada CPU y fija la capacidad máxima de memoria que dispone el sistema.

El **bus de CONTROL** está formado por líneas independientes de entrada o salida a la CPU que se utilizan para controlar y sincronizar las operaciones que se realizan entre los distintos dispositivos.
Las señales de control más típicas son :

- READ/WRITE (leer o escribir en memoria)
- RESET (inicializar la CPU)
- INT (manejo de interrupciones)
- Señales de habilitación de registros, buses, memoria, etc.

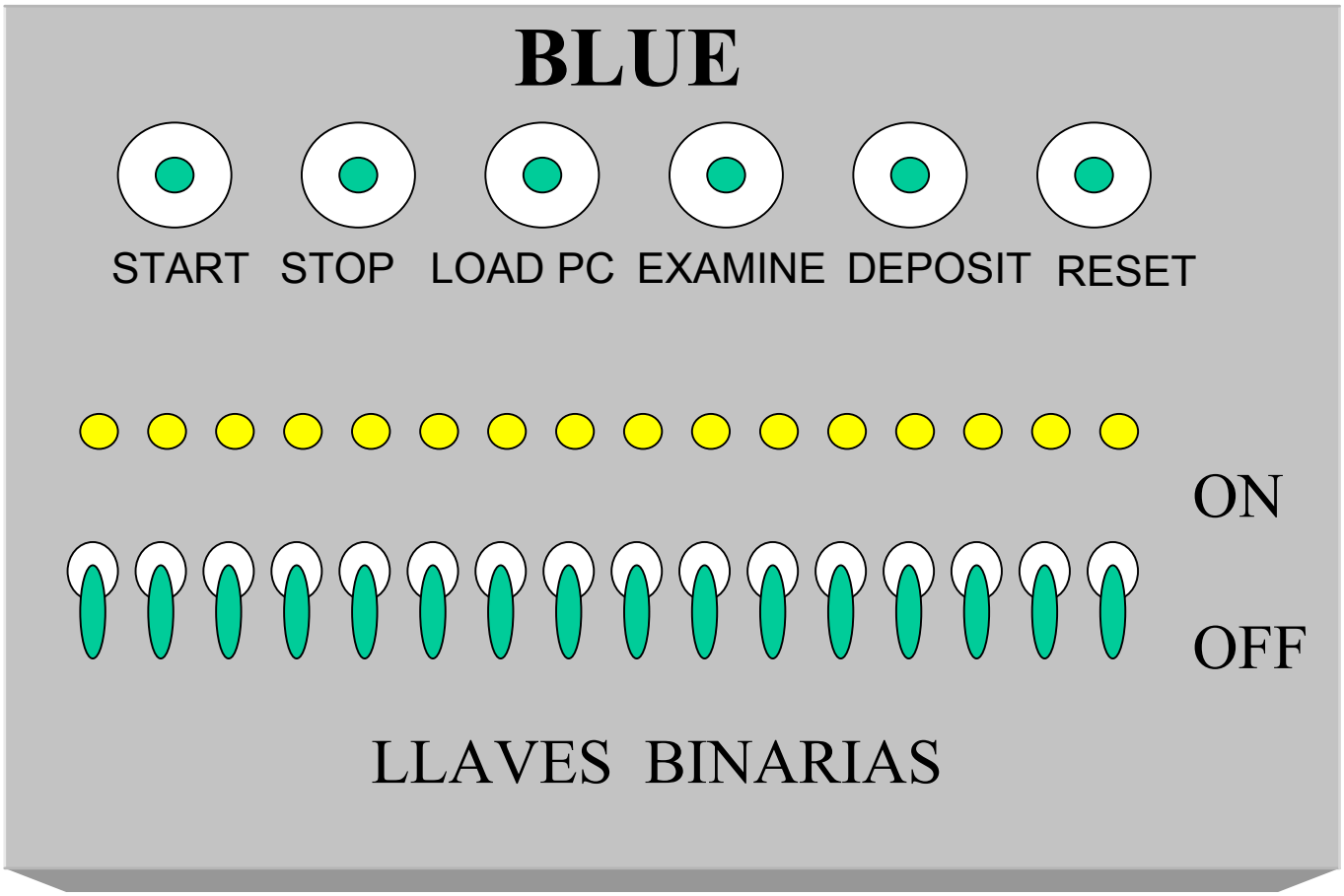
DESCRIPCION DE UNA MAQUINA ELEMENTAL - BLUE

Las características principales de esta computadora son:

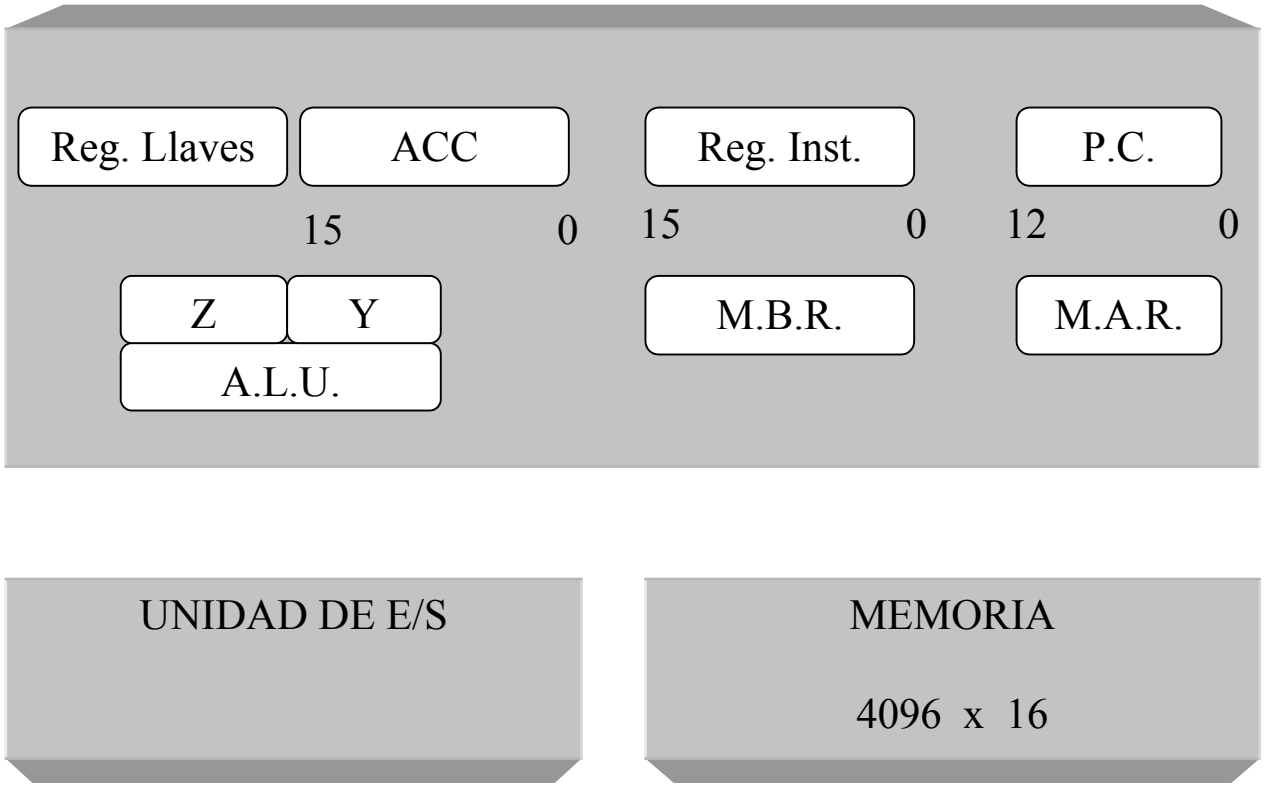
- Usa sistema binario y complemento a dos
- Su memoria es de 4096 x 16 (4096 posiciones de memoria de 16 bits cada una)
- Usa punto fijo por hardware y sus datos son de 16 bits (15 bits de magnitud y un bit de signo)
- Usa instrucciones de longitud fija de 16 bits.
- El bus de datos es de 16 bits y el de direcciones es de 12 bits.

Externamente tiene un frente con las siguientes llaves y luces:

- Pulsador de ARRANQUE (Start)
- Pulsador de PARADA (Stop)
- Pulsador de CARGAR PC (Load PC)
- Pulsador de DEPOSITAR (Deposite)
- Pulsador de EXAMINAR (Examine)
- Pulsador de RESET (Reset)
- 16 Llaves de 3 posiciones
- 16 Leds (luces)

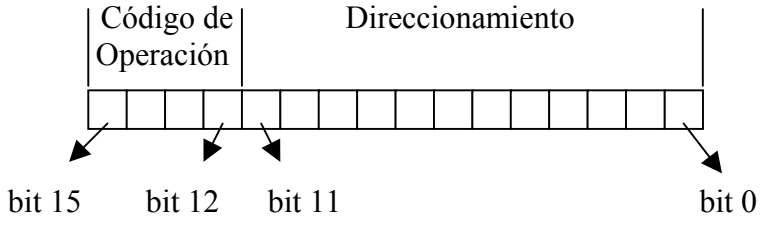


Esta máquina, aunque antigua, posee todas las funciones que realiza una computadora y su arquitectura responde a la máquina elemental de Von Newmann.
Su esquema general es:



EL CONJUNTO DE INSTRUCCIONES

Como anticipamos, esta computadora pose un tipo de instrucción de longitud fija de 16 bits con el siguiente formato:



Debido a que tenemos solamente 4 bits para el código de operación, disponemos de 16 instrucciones que se describen a continuación:

DESCRIPCION DE LAS INSTRUCCIONES:

Cód binario	Cód octal	Nombre	Mnemónico
0000 XXXXXXXXXXXX	00 XXXX	HALT (PARAR)	HLT XXXX
0001 XXXXXXXXXXXX	01 XXXX	ADD (SUMAR)	ADD XXXX
0010 XXXXXXXXXXXX	02 XXXX	OR-EXCLUSIVA	XOR XXXX
0011 XXXXXXXXXXXX	03 XXXX	AND	AND XXXX
0100 XXXXXXXXXXXX	04 XXXX	OR	IOR XXXX
0101 XXXXXXXXXXXX	05 XXXX	NOT (COMPLEM. A 1)	NOT XXXX
0110 XXXXXXXXXXXX	06 XXXX	LOAD AC (CARGAR AC)	LDA XXXX
0111 XXXXXXXXXXXX	07 XXXX	STORE AC (ALMAC. AC)	STA XXXX
1000 XXXXXXXXXXXX	10 XXXX	SALTO SUBRUTINA	SRJ XXXX
1001 XXXXXXXXXXXX	11 XXXX	SALTO CONDICIONAL	JMA XXXX
1010 XXXXXXXXXXXX	12 XXXX	SALTO INCONDICIONAL	JMP XXXX
1011 XXXXXXXXXXXX	13 XXXX	ENTRADA DE DATOS	INP XXXX
1100 XXXXXXXXXXXX	14 XXXX	SALIDA DE DATOS	OUT XXXX
1101 XXXXXXXXXXXX	15 XXXX	ROTACION DE BITS	RAL XXXX
1110 XXXXXXXXXXXX	16 XXXX	COPIAR LLAVES	CSA XXXX
1111 XXXXXXXXXXXX	17 XXXX	NO OPERACION	NOP XXXX

La descripción de cada una de las instrucciones es la siguiente:

- HLT XXXX: para el funcionamiento de la computadora. Presionando el pulsador START de la consola principal la computadora arranca nuevamente siguiendo con la ejecuciòn de la instrucciòn siguiente al HALT. El campo de direcciones XXXX se ignora.
- ADD XXXX: realiza la suma aritmética en complemento a dos de los operandos ubicados en el acumulador y en la dirección de memoria expresada en el campo XXXX, y deja el resultado en el acumulador. El contenido de la dirección de memoria XXXX no cambia, mientras que el dato que se encontraba en el acumulador sí. Si el resultado de la suma es mayor que 2^{15}-1 ó menor que -2^{15} la computadora se para.
- XOR XXXX: Realiza la OR-Exclusiva bit a bit de los de los operandos ubicados en el acumulador y en la dirección de memoria expresada en el campo XXXX, y deja el resultado en el acumulador. El contenido de la dirección de memoria XXXX no cambia, mientras que el dato que se encontraba en el acumulador sí.

- AND XXXX: Realiza la AND bit a bit de los operandos ubicados en el acumulador y en la dirección de memoria expresada en el campo XXXX, y deja el resultado en el acumulador. El contenido de la dirección de memoria XXXX no cambia, mientras que el dato que se encontraba en el acumulador sí.
- IOR XXXX: Realiza la OR bit a bit de los operandos ubicados en el acumulador y en la dirección de memoria expresada en el campo XXXX, y deja el resultado en el acumulador. El contenido de la dirección de memoria XXXX no cambia, mientras que el dato que se encontraba en el acumulador sí.
- NOT XXXX: Cada bit del dato en el acumulador es reemplazado por su complemento lógico. La dirección de memoria XXXX se ignora.
- LDA XXXX: El contenido de la ubicación de memoria XXXX es copiado en el acumulador. El contenido de la dirección de memoria XXXX no cambia, mientras que el dato que se encontraba en el acumulador sí.
- STA XXXX: El contenido del acumulador es copiado en la dirección de memoria XXXX. El contenido del acumulador no cambia, mientras que el dato que se encontraba en la dirección de memoria sí.
- SRJ XXXX: Sirve para hacer un salto del programa a una subrutina. Para esto realiza una copia del contador del programa en los 12 bits más bajos del acumulador (en los 4 bits más altos del acumulador se ponen ceros). Luego se copia el numero XXXX en el contador del programa para que la próxima instrucción sea tomada de la dicha dirección.
- JMA XXXX: Produce un salto a otra dirección de programa si el bit de signo del acumulador es uno (es decir si el acumulador contiene un número negativo). Si se cumple dicha condición copia el número XXXX en el contador de programa y la próxima instrucción es tomada de esta dirección. Si no cumple la condición (el bit de signo del acumulador es cero, es decir que el dato en el acumulador es positivo o cero), esta instrucción no realiza nada y el programa sigue normalmente.
- JMP XXXX: Produce un salto incondicional a otra parte del programa, por lo que copia el número XXXX en el contador de programa y la próxima instrucción a ejecutar es tomada de la dirección XXXX.
- INP XXYY: Los 8 bits de mayor peso del acumulador se colocan a cero, y el próximo carácter de 8 bits que viene del dispositivo externo YY se coloca en la parte baja del acumulador. La parte XX del campo de dirección se ignora. La próxima instrucción no se ejecuta hasta que la transferencia del dato se haya completado.
- OUT XXYY: Lo 8 bits más significativos del acumulador se envían al dispositivo externo YY. La parte XX del campo de dirección se ignora. Si el dispositivo externo no puede aceptar el dato en ese momento, la computadora espera hasta que se haya podido realizar la trasferencia.
- RAL XXXX: Los bits del acumulador se rotan un lugar hacia la izquierda. El bit AC_{15} se coloca en AC_o de modo que el desplazamiento es cíclico. El campo de direcciones XXXX se ignora.
- CSA XXXX: El número que está en el registro de llaves (introducido por las llaves de la consola) se copia en el acumulador. El campo de direcciones XXXX se ignora.
- NOP XXXX: Esta instrucción no hace nada. El campo de direcciones XXXX se ignora.

EL CICLO DE MAQUINA:

Se llama **"ciclo de máquina"** de una computadora al procedimiento que consta de todas las tareas necesarias para poder ejecutar completamente una instrucción del programa almacenado en memoria, que podemos sintetizar de la siguiente forma:

- Búsqueda de una instrucción a memoria
- Lectura e interpretación de esa instrucción
- Ejecución de la misma.
- Almacenamiento de resultados
- Preparación para leer la próxima instrucción.

Esta computadora elemental (BLUE) tiene un ciclo de máquina básico compuesto por dos partes:

- **CICLO DE BUSQUEDA**
- **CICLO DE EJECUCION**

Durante el **Ciclo de Búsqueda**, la instrucción almacenada en la memoria y apuntada por el **Contador de Programa (P.C.)** es localizada en la memoria y copiada en el **Registro de Instrucciones (R.I.)**. Luego el número almacenado en el **P.C.** es incrementado en uno, logrando así que ahora apunte a la próxima celda de memoria (o sea a la siguiente instrucción).
Al completar el **Ciclo de búsqueda**, la instrucción que está en el **R.I**. es analizada, decodificada y ejecutada. Si la presente instrucción no necesita hacer una nueva búsqueda a memoria (de algún dato u operando) el **ciclo de máquina** termina acá.
Si es necesario buscar un operando a memoria, entonces comienza el **Ciclo de Ejecución** para realizar un nuevo acceso a memoria para traer al operando necesario y completar así la instrucción.

FLUJO DE INFORMACION EN LA BLUE

Sin analizar las instrucciones en detalle todavía, mostraremos los movimientos de información entre registros dentro de la máquina:

- **Flujo de direcciones (addreses)** en la BLUE (son movimientos de 12 bits entre registros):
  - Load PC: envía los 12 bits más bajos del registro de llaves (R.Sw.) al P.C.
  - Saltos (JMP, JMA, SRJ): envía los 12 bits más bajos del R.I. al P.C.
  - SRJ (salto a subrutina): envía los 12 bits del P.C. al Acumulador (ACC).
  - Búsqueda de una instrucción: envía los 12 bits del P.C. al M.A.R.
  - Búsqueda de un Operando: envía los 12 bits más bajos de R.I. al M.A.R.

- **Flujo de instrucciones y operandos** en la BLUE (son movimientos de 16 bits entre registros):
  - C.S.A.: copia los 16 bits del R.Sw. al ACC
  - Deposit: copia los 16 bits del R.Sw. al M.B.R.
  - Instrucciones: se copian del M.B.R. al R.I.
  - LDA: copia los 16 bits del M.B.R. al ACC
  - STA: copia los 16 bits del ACC al M.B.R.
  - Operaciones de la A.L.U. (en el ciclo de ejecución):
    - Copia los 16 bits del ACC al Registro Z de la A.L.U.
    - Copia los 16 bits del M.B.R. al Registro Y de la A.L.U.
    - El Resultado (o salida de la A.L.U.) se copia al ACC

- Flujo de señales de control. (que se verán posteriormente).

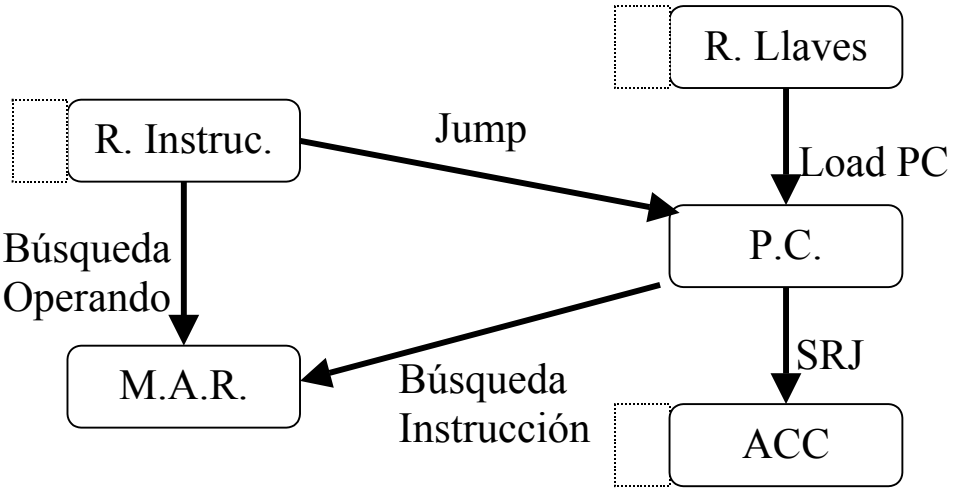


Fig. La transmisión de direcciones en la Blue

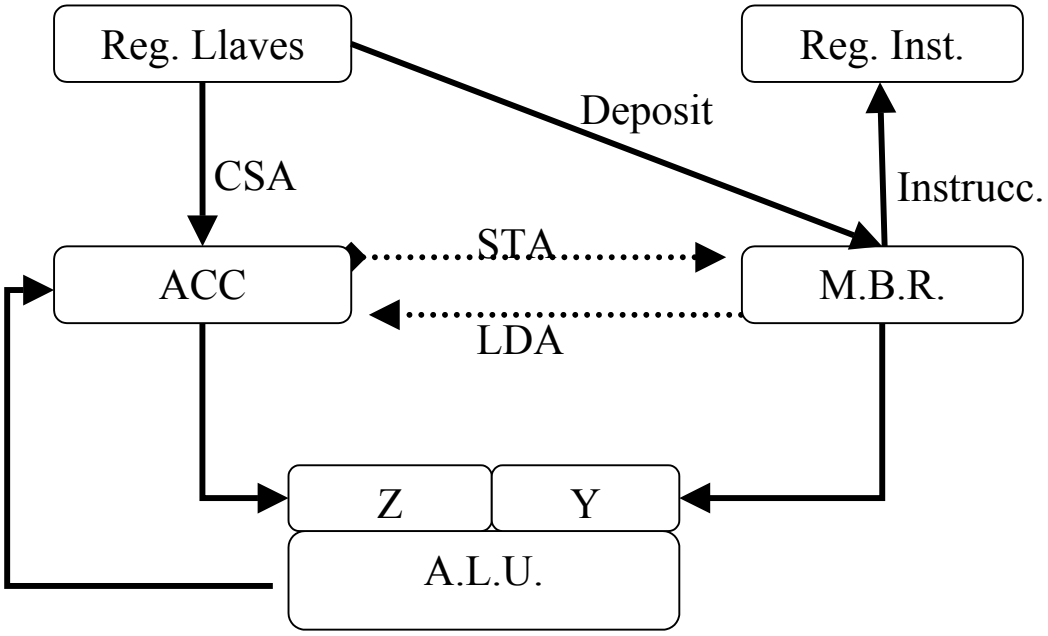


Fig. La transmisión de instrucciones y operandos en la Blue

Para poder realizar las posibles transferencias vistas es necesario implementar algún esquema de relación entre los registros, memoria y unidades de E/S, como por ejemplo:

- BUS COMUN
- BUS MULTIPLE
- PUNTO A PUNTO

UNIDAD DE CONTROL:

La tarea de la unidad de control es coordinar todas las acciones de la máquina. Para este trabajo es necesario una secuencia de pulsos y señales que deben generarse. De acuerdo al esquema utilizado las computadoras se dividen en:

- SINCRONAS

- ASINCRONAS

Las computadoras **asíncronas** son aquellas en que cada unidad recibe una señal del dispositivo anterior para realizar su tarea, y a su vez envía un señal al siguiente cundo termina de realizar la suya. Esto tiene como ventaja de que cada tarea tarda el mínimo de tiempo posible; pero la desventaja de que cada unidad debe ser lo suficiente inteligente y el hardware se complica.

Las computadoras **síncronas** son aquellas que tienen un reloj patrón que emite pulsos en periodos de tiempos fijos (frecuencia de reloj), y en cada pulso hay una o más tareas que se realizan. Esto tiene como ventaja que mantiene las distintas tareas de cada dispositivo en orden y secuencia y con un hardware sencillo La desventaja es que ninguna tarea se puede realizar en menos tiempo que la duración de un ciclo de reloj.

La BLUE es una máquina síncrona, y veremos a continuación el diseño de la unidad de control:

Consta básicamente de 3 (tres) biestables (o flip-flops) de control:

- RUN (arranque): es un biestable Set-Reset que arranca (pone en "1" su salida Q) por el botón deSTART o se para (pone en "0" su salida Q) por el botón de STOP, por la instrucción HALT o por la detección de un overflow aritmético.
- STATE (estado): es un biestable D tiene dos condiciones (sus salidas) para determinar el ciclo de búsqueda (Fetch) o el ciclo de ejecución (Execute)
- CLOCK (reloj) : es un biestable Set-Reset que cumple la función de inicializar el reloj.

Cada ciclo de memoria se divide en 8 intervalos de tiempo iguales (o ciclos menores) de 125 nanosegundos cada uno generados por el CLOCK; y en cada intervalo se genera un pulso de salida. Estos pulsos se van generando al pasar por líneas de retardo (o temporizadores) (las marcadas por T en la figura A).

El funcionamiento es el siguiente:

Cuando se apreta el botón de START, el flip-flop RUN genera un uno a su salida y después de pasar por unos circuitos lógicos llega un pulso al flip-flop CLOCK. Este pulso se transmite a través de la cadena de temporizadores generando así una secuencia de pulsos como la de la figura B.

EL CICLO DE BUSQUEDA:

Si el flip-flop RUN se setea en ON, arranca el reloj, y si el flip-flop de STATE está en Fetch se inicia el ciclo de búsqueda de la máquina, en el cual la máquina carga la instrucción en el registro de instrucciones que está almacenada en la posición de memoria apuntada por el contador de programa.
La secuencia es la siguiente:

Pulso Reloj	Acción	Comentario
1.	PC ⇨ MAR PC ⇨ Z	Copia el PC al MAR y a Z y comienza a leer la próxima instrucción.
2.	+1 ⇨ Y	Coloca +1 en Y
3.		Tiempo de espera
4.	ALU ⇨ PC	Hace la suma de PC+1 y coloca el resultado en el PC
5.	M ⇨ MBR	Coloca el dato de la memoria en el MBR.
6.	MBR ⇨ IR	Copia el contenido del MBR en el IR y comienza la decodificación de la instrucción.
7.		Disponible para decodificación y ejecución.
8.		

En el pulso de reloj 1 se copia el contenido del contador de programa (PC) al registro de direcciones de memoria (MAR) y al registro Z de la ALU, y se inicia el ciclo de lectura de la memoria. En el pulso de reloj 2 se coloca el número +1 en el registro Y de la ALU. En el pulso de reloj 3 esperamos que se realice la suma PC + 1. En el pulso 4 se copia el resultado de la suma (la salida de la ALU) al contador de Programa (esto incrementa el PC y está listo para indicar la próxima instrucción). En el pulso 5 se copia el dato de la memoria al registro buffer de memoria (MBR). En el pulso 6 se copia el contenido del MBR al registro de instrucción (IR) y comienza la decodificación de la instrucción extraída de la memoria.
La memoria no estará disponible hasta finalizar el pulso de reloj 8, por lo que este tiempo disponible (entre los pulsos 7 y 8) se puede utilizar para finalizar algunas instrucciones de la BLUE, que son HALT, NOP, JMP, JMA, SRJ, CSA, NOT, RAL como indica la figura C. Estas instrucciones finalizan en el ciclo de Búsqueda (no necesitan otro ciclo de memoria) y al finalizar empieza una nueva búsqueda para la próxima instrucción.

Instrucciones de dos ciclos:

Todas las instrucciones que requieran la búsqueda de un dato a memoria para realizar una operación lógica o matemática, o realizar una transferencia de datos hacia o desde la memoria, requieren un segundo ciclo de memoria llamado ciclo de ejecución. Las instrucciones que requieren este segundo ciclo son: LDA, STA, ADD, XOR, AND, IOR.
La descripción paso a paso del ciclo de ejecución de cada instrucción se explica en la figura D.

Instrucciones de Entrada/Salida:

Existen dos instrucciones que no requieren referencia a memoria, pero sí necesitan más tiempo que un simple ciclo. Estas son las instrucciones INPUT y OUTPUT que su ciclo de ejecución se explica a continuación:

Pulso Reloj	INP	OUT	Comentarios
7.	1 ⇨ TRA	1 ⇨TRA	Los bits IR_{5-0} seleccionan el dispositivo de E/S
8.	E⇨STATE	E ⇨ STATE	Fin ciclo de búsqueda y empieza el de ejecución
1.	-----	-----	
2.	-----	-----	
3.	-----	-----	
4.	-----	-----	
5.	-----	-----	
6.	-----	-----	
7.	Si R = 1 Inp⇨ A; 0 ⇨TRA	Si R = 1 0 ⇨ TRA	Si el Flag R=1 en INP se copian los datos del periférico a A, y en ambos se pone TRA=0.
8.	Si TRA = 0 F ⇨STATE Si TRA = 1 E⇨STATE	Si TRA = 0 F ⇨STATE Si TRA = 1 E ⇨STATE	Si se completó la transferencia se inicia un nuevo ciclo de búsqueda. Si no se completó la transferencia se inicia un nuevo ciclo de ejecución.

Estas instrucciones se utilizan para intercambiar datos entre la CPU de la BLUE y dispositivos periféricos de entrada/salida.

Debido a que los pulsos de reloj son internos a la BLUE, la sincronización entre la CPU y los dispositivos externos se hace con dos señales de control especiales, que son:

- **TRA** que la envía la CPU para indicar que está lista para enviar o recibir datos (TRA=1).
- **R** que la envía los periféricos para indicar que están listos para la transferencia.

Cuando se utiliza alguna de estas instrucciones (INP para enviar datos desde algún periférico al computador, o OUT para enviar datos desde el computador a los periféricos), la BLUE en el pulso de reloj 7 coloca la señal TRA en 1 y envía los 6 bits más bajos del IR al bus de datos, los cuales identifican a uno de los 64 posibles dispositivos de E/S, con el cual vamos a intercambiar los datos; e inmediatamente comienza la transferencia de información

Para INPUT, con TRA=1, se espera hasta el pulso 7 del ciclo de ejecución, y si R=1 se copian los datos del bus al acumulador (recordando que se copian sólo 8 bits) y se cambia TRA=0; y en el pulso 8 se termina la transferencia. Si en el pulso 7 R=0 es porque el periférico no terminó de colocar el dato en el bus y se necesita otro ciclo de ejecución.

Para OUTPUT, con TRA=1, se colocan los 8 bits más altos del acumulador en el bus y se espera hasta el pulso 7 del ciclo de ejecución, y si R=1 se terminó la transferencia de datos al periférico seleccionado y se cambia TRA=0; y en el pulso 8 se termina la transferencia. Si en el pulso 7 R=0 es porque el periférico no terminó de recibir el dato y se necesita otro ciclo de ejecución.

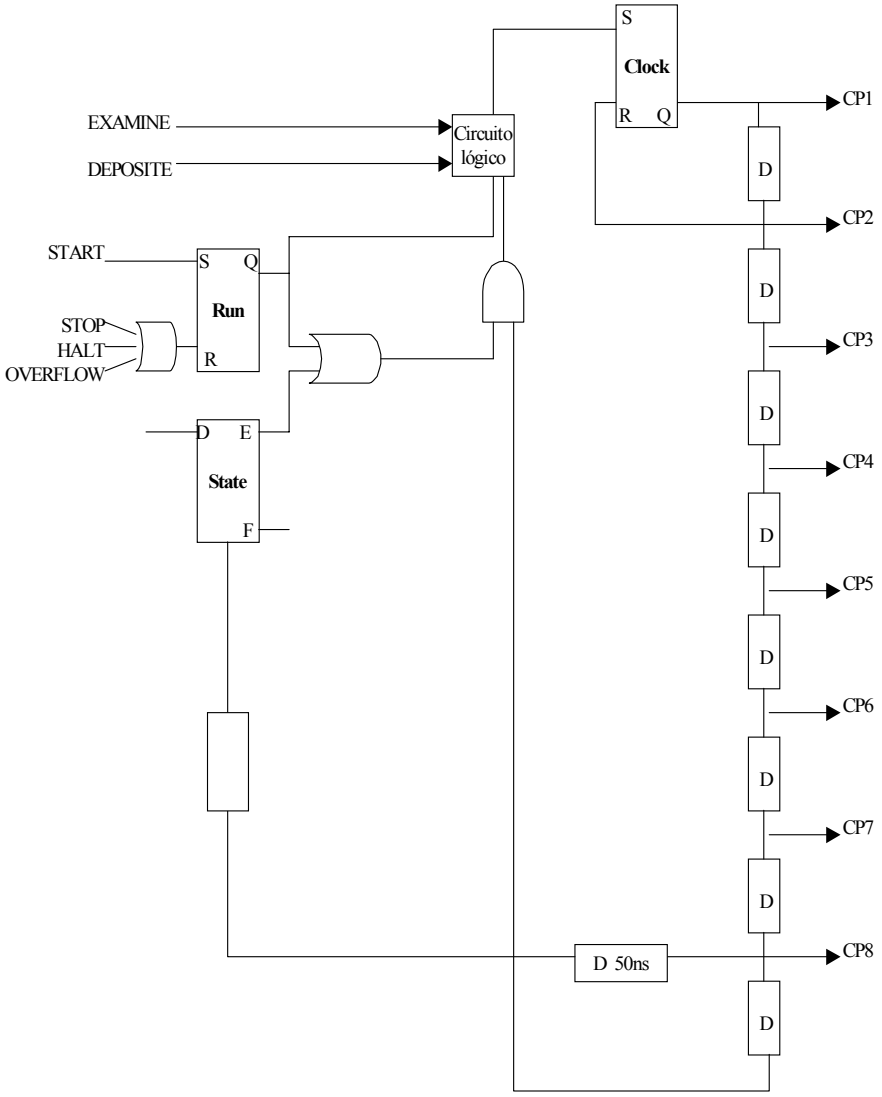

S
Clock
R Q
CP1
EXAMINE
DEPOSITE
Circuito lógico
D
CP2
START
S Q
Run
R
STOP
HALT
OVERFLOW
D
CP3
D E
State
F
D
CP4
D
CP5
D
CP6
D
CP7
D
CP8
D 50ns
D

Figura A:

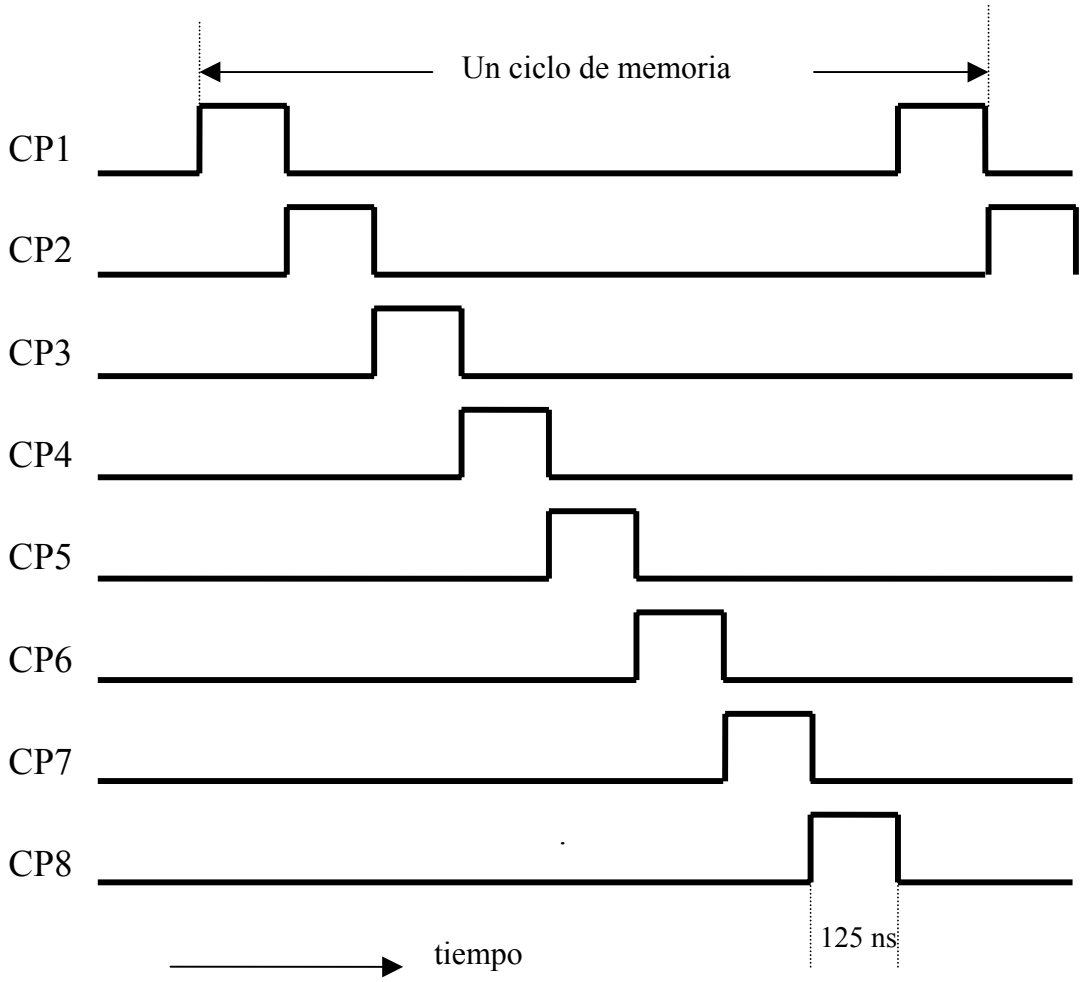
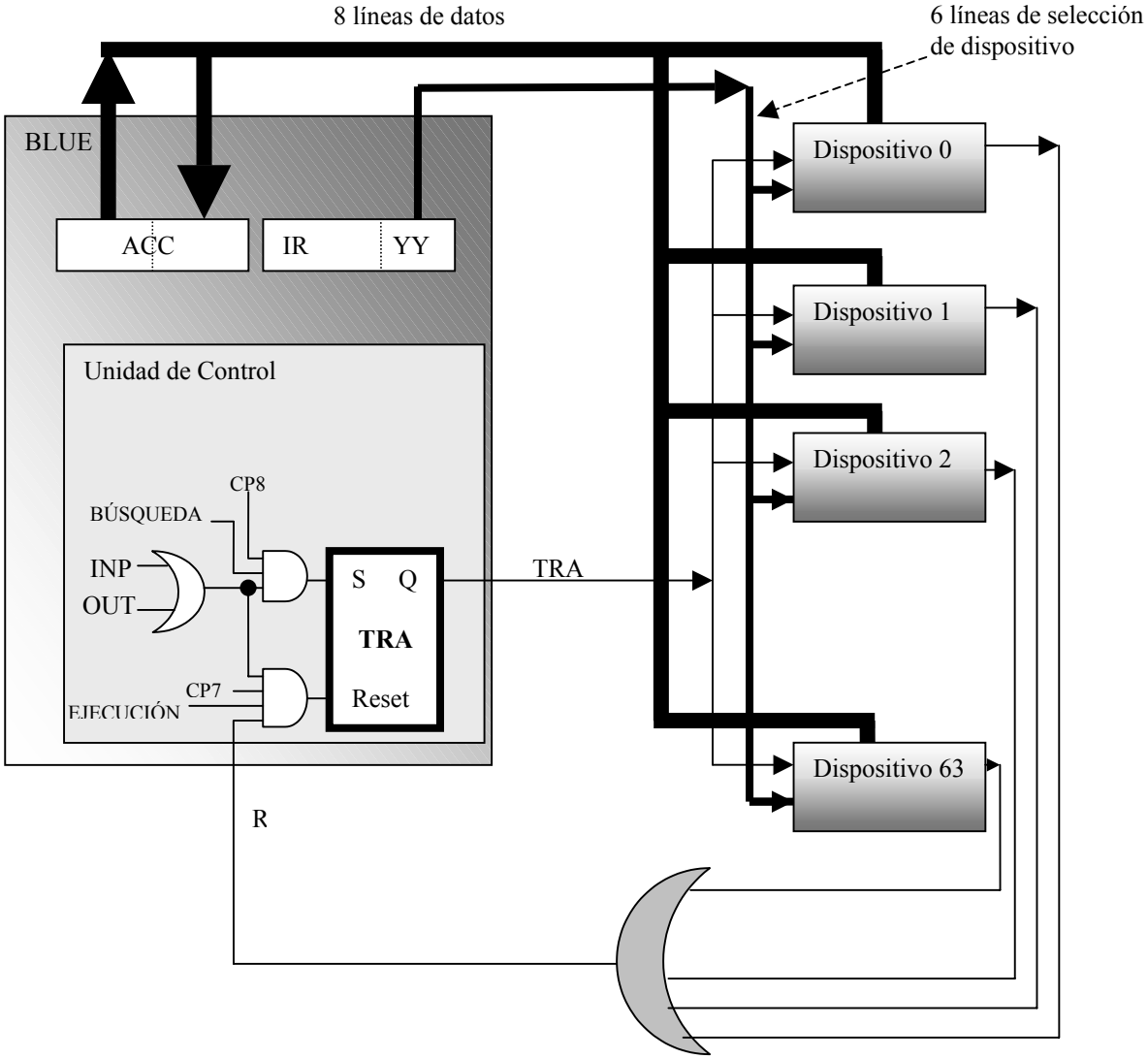


Figura B: Diagrama de tiempo del reloj

Entrada - Salida en BLUE

Como se vió anteriormente las transferencias de entrada - salida en Blue son realizadas bajo control de programa y a través del acumulador. Cuando se ejecuta una instrucción INP o OUT la *Unidad de Control* genera la señal Transferencia (TRA) que deberá conectarse a todos los dispositivos externos, sólo el dispositivo seleccionado en el campo YY de la instrucción OUT o INP se dará por "aludido" y generará una señal Ready (R) cuando haya completado su tarea. Esta señal R es sensada por la unidad de Control a fin de proseguir con la próxima instrucción. Si R nunca pasa a uno (por ejemplo si el dispositivo externo se dañó) la *Unidad de Control* permanecerá indefinidamente en el estado de Ejecución de la instrucción de entrada o salida.

El siguiente esquema aclara lo explicado. Se observa un biestable encargado de generar la señal TRA y sensar la señal R proveniente de los dispositivos externos.



Pulso Reloj	HALT	NOP	JMP	JMA	SRJ	CSA	NOT	RAL
7.	---	---	---	----	PC ⇨ A	----	A ⇨ Z	A ⇨ Z
8.	Off ⇨ Run	---	IR ⇨ PC	If A_{15}=1 IR ⇨ PC	IR ⇨ PC	$\overline{SR}$ ⇨ A	Z ⇨ A	2*Z ⇨ A

Figura C: Fin del ciclo de búsqueda para las 8 instrucciones de un solo ciclo.

Pulso Reloj	LDA	STA	ADD	XOR	AND	IOR
8.	al finalizar este pulso del ciclo de búsqueda, el biestable de estado cambia a Ejecución					
1.	IR ⇨ MAR	IR ⇨ MAR	IR ⇨ MAR	IR ⇨ MAR	IR ⇨ MAR	IR ⇨ MAR
2.	-----	-----	A ⇨ Z	A ⇨ Z	A ⇨ Z	A ⇨ Z
3.	-----	-----	-----	-----	-----	-----
4.	-----	-----	-----	-----	-----	-----
5.	Mem ⇨ MBR	A ⇨ MBR	Mem ⇨ MBR	Mem ⇨ MBR	Mem ⇨ MBR	Mem ⇨ MBR
6.	MBR ⇨ A	-----	MBR ⇨ Y	MBR ⇨ Y	MBR ⇨ Y	MBR ⇨ Y
7.	-----	-----	-----	-----	-----	-----
8.	STATE ⇨ F	STATE ⇨ F	SUM ⇨ A STATE ⇨ F	XOR ⇨ A STATE ⇨ F	AND ⇨ A STATE ⇨ F	OR ⇨ A STATE ⇨ F

Figura D: Fin del ciclo de búsqueda para las instrucciones de dos ciclos