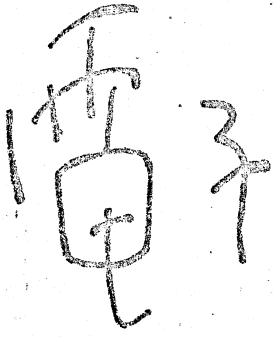


Departamento de  
Electrotecnia

Cátedra de



68

PROYECTO

título

Desarrollo de una Interfase con

Capacidad de Interrupción.

autor

Ricardo César Barrios

tribunal examinador

Jug Santos

presentado el

aprobado el 18/3/74

profesor

Ing. Jorge Santos

## I N D I C E

1) OBJETIVO GENERAL . . . . .	Pág.1
2) INTRODUCCION . . . . .	Pág.2
3) ALTERNATIVAS SUCESIVAS DE LA ARQUITECTURA A ADOPTAR.	Pág.6
4) RESUMEN DEL PROYECTO . . . . .	Pág.8
5) <u>DISEÑO LOGICO</u>	
5-1) TARJETAS DE CONTROL . . . . .	Pág.10
5-2) SISTEMA DE INTERRUPCION . . . . .	Pág.12
5-3) DETENCION DE LA MAQUINA . . . . .	Pág.18
5-4) GENERACION DE PULSOS Y COMPASES . . . . .	Pág.20
5-5) CONEX.ENTRE UC,INTERFASE Y CAJA TARJ.DE CONTROL	Pág.23
5-6) RELOJ . . . . .	Pág.26
5-7) TARJETA INTERFASE EQUIPO LECTOR . . . . .	Pág.27
5-8) TARJETA INTERFASE EQUIPO PERFORADOR . . . . .	Pág.27
6) <u>IMPLEMENTACION ELECTRONICA</u> . . . . .	Pág.30
6-1) TARJETAS DE CONTROL	
6-1-1) TARJETA DE CONTROL EQUIPO LECTOR . . . . .	Pág.31
6-1-2) TARJETA DE CONTROL EQUIPO PERFORADOR . .	Pág.34
6-2) TARJETA II . . . . .	Pág.35
6-3) TARJETA III . . . . .	Pág.35
6-4) CONECTORES . . . . .	Pág.35
6-5) TARJETA INTERFASE EQUIPO LECTOR . . . . .	Pág.38
6-6) TARJETAS INTERFASE EQUIPO PERFORADOR . . . . .	Pág.39
7) <u>APOYO DE PROGRAMACION</u> . . . . .	Pág.40
7-1) SUBROUTINA DE ENTRADA . . . . .	Pág.41
7-2) SUBROUTINA DE SALIDA . . . . .	Pág.44

PROYECTO:

"Desarrollo de una interfase con capacidad de interrupción".

1) OBJETIVO GENERAL:

Cumpliendo con las pautas fijadas por la actual política universitaria tendientes a aumentar la relación entre las / investigaciones realizadas en la Universidad y las necesidades del medio al cual está integrada, el Area de Sistemas Digitales de la Universidad Nacional del Sur "Dr.Miguel López Francés" encara la realización de una minicomputadora de propósito general.

Este trabajo da lugar, por el momento, a tres proyectos que son: desarrollo de la Unidad de Control, de la Unidad Aritmética y de la Unidad de Entrada y Salida.

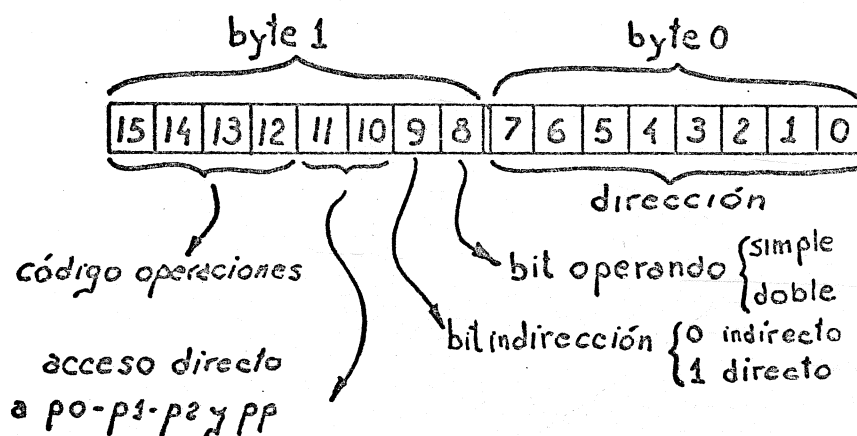
Aunque el trabajo requiere, consecuentemente, la formación de un equipo, cada proyecto en sí está perfectamente individualizado. Este informe se refiere al último de los tres, mencionados, esto es, la Unidad de Entrada y Salida.

Si además agregamos que existen en el país intentos de fabricación de circuitos integrados en pequeña y mediana escala, que son precisamente los que han sido utilizados para implementar los diseños de las distintas unidades, entonces al completarse ambos esfuerzos daríamos al país la capacidad de poseer su propia minicomputadora, tendiendo de esta forma a alcanzar la independencia tecnológica tan deseada por todos.

## 2) INTRODUCCION:

La minicomputadora posee en su totalidad cuatro registros de 16 bits que son: registro C, contador de programa; registro P, instrucción presente; registro A, acumulador; / registro E, extensión acumulador y un registro de ocho / bits de extensión de dirección (reg.D).

La forma de operar es la siguiente: con la dirección del registro C se accede a la memoria de primer nivel, que es lo único que no se ha implementado hasta el momento, pero será el objeto de proyectos posteriores; la instrucción extraída correspondiente a esa instrucción se coloca en el / registro P cuyos bits ordenados en dos bytes responden al siguiente formato:



Cuando el acceso es indirecto o sea el bit de indirección (bit 11) es cero, la dirección que está en el byte cero es la dirección del operando; con ella accedemos a memoria y extraemos la dirección; el byte cero de la misma va al byte cero del registro P y el byte uno de la dirección va al registro D para no perder el byte 1 del registro P que es el que tiene la información de la instrucción.

Las operaciones las realiza la Unidad Aritmética entre el registro acumulador y el registro de salida de memoria que cumple la función de Segundo Operando (SO); además la UA tiene la posibilidad, por la forma en que está implementada, de sumar y restar 1 al acumulador y sumar 1 al S.O..

Los cuatro bits del código de operaciones nos permiten un

código de 16 instrucciones, una de las cuales se utiliza para expandir el código en otras 16 instrucciones que no tienen acceso a memoria; en virtud de que son instrucciones simples no descomponibles en otras mas elementales, las denominamos microinstrucciones.

Debe notarse sin embargo que ellas no responden al otro concepto usual de microinstrucción por no poder ser combinadas varias de ellas de manera de ser obedecidas en un sólo ciclo de máquina.

Las instrucciones son:

- ca: Cargar Acumulador con lo que hay en el contenido de la dirección de memoria presentada por la instrucción M(z)
- ia: Intersectar Acumulador M(z).
- aa: Adicionar el Acumulador M(z).
- ra: Restar al Acumulador M(z).
- sa: Salto condicional según el Acumulador sea mayor o igual a cero o negativo.
- sm: Salto condicional según el contenido en memoria de una dirección cualquiera presentada con la instrucción.
- ta: Transferir el contenido del Acumulador a memoria.
- ss: Salto a subrutina.
- si: Salto incondicional.
- ma-da: Multiplicar Acumulador y Dividir Acumulador (no implementadas por el momento).
- ap: Arrancar periférico.
- te: Transferir salida del periférico.
- ts: Transferir salida (del Acumulador a la entrada del periférico).

Estas últimas tres instrucciones son de 1 byte y no tienen acceso a memoria; con el concepto anterior son microinstrucciones, aunque llamaremos instrucciones de Entrada y Salida.

#### MICROINSTRUCCIONES:

- if: Instrucción fantasma (no hacer nada).
- ka: Complementar Acumulador.
- au: Adicionar 1 al Acumulador.
- ru: Restar 1 al Acumulador.
- di: Desplazar 1 bit a la izquierda.
- dd: Desplazar 1 bit a la derecha.
- ri: Rotar 1 bit a la izquierda.

rd: Rotar 1 bit a la derecha.

1a: Limpiar Acumulador.

sb: Salto por bit menos significativo de E.

pm: Parar máquina.

Los cinco restantes no han sido implementadas.

El flujo de información se realiza siempre a través de un canal único, habilitando las compuertas correspondientes (fig. 1).

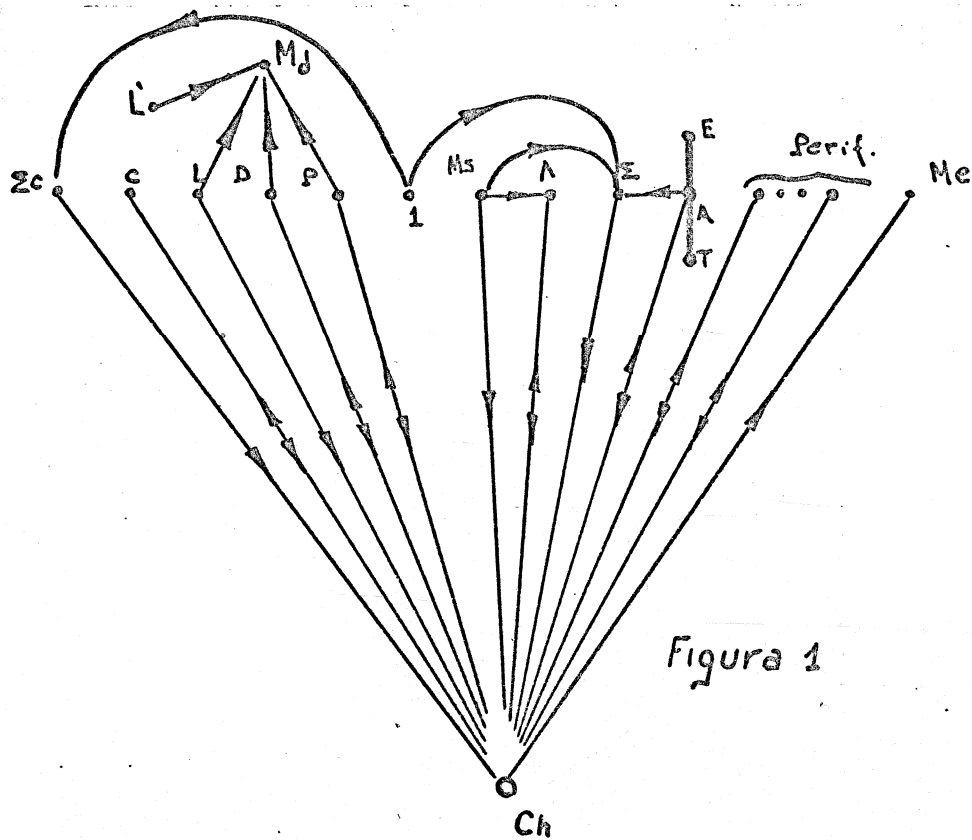


Figura 1

L: Llaves información.

L': Llaves dirección.

T: Transporte.

Me:Entrada memoria.

Ms: Salida memoria.

Md:Entrada de dirección de memoria.

$\wedge$  : Operación de intersección.

$\Sigma$  : Sumador principal.

$\Sigma C$  : Sumador de C.

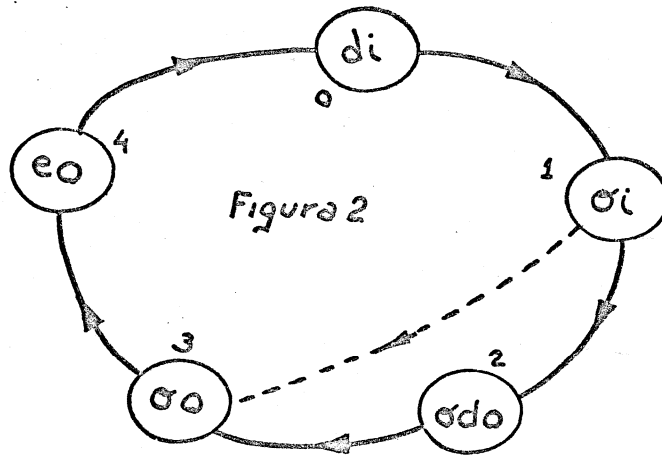
La consola consta de 16 llaves de información y 16 de dirección que permitirán introducir en la máquina el programa cargador.

Provisoriamente, como aún no se dispone de memoria, las lla-

ves de dirección cumplen la función de segundo operando o sea salida de memoria.

El funcionamiento de la máquina puede hacerse por pulsos, por compases o en forma continua según el modo que se desee.

Las fases de ejecución de la instrucción según un diagrama del tipo Bell y Newel es el indicado en la figura 2.



di: Obtiene dirección instrucción.

oi: Obtiene instrucción.

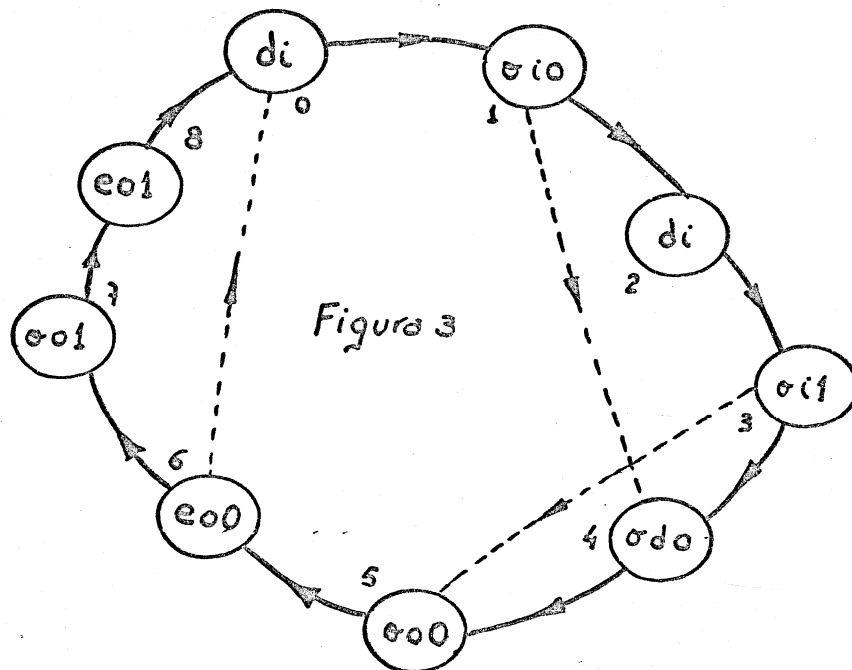
odo: Obtiene dirección operando.

oo : Obtiene operando.

eo: Ejecuta operando.

### 3) ALTERNATIVAS SUCESIVAS DE LA ARQUITECTURA A ADOPTAR:

La idea original de la cual se partió fué el acceso por byte, pensando en obtener de este modo una máquina en su concepción mínima, aunque eficiente y con una memoria mas compacta, permitiendo además operaciones con operandos cortos (de un byte) y largos (de dos bytes).



- di: Obtiene dirección, byte 0 instrucción.
- oi0: Obtiene byte 0 de instrucción.
- di: Obtiene dirección byte 1 instrucción.
- oil: Obtiene byte 1 de instrucción.
- odo: Obtiene dirección operando.
- oo0: Obtiene byte 0 de operando.
- eo0: Ejecuta operación con byte 0.
- ool: Obtiene byte 1 de operando.
- eo1: Ejecuta operación con byte 1.

Sin embargo el ciclo de obediencia de la instrucción (figura 3), se hacía muy largo, sobre todo si se trataba de instrucciones de dos bytes con operandos largos de acceso indirecto, con lo cual se elevaba demasiado el tiempo necesario para la ejecución de cada instrucción. Por lo tanto se modificó la idea primitiva adoptando el esquema de la figura 2, que permite trabajar con operandos cortos u operandos largos. Sin embargo, esto complica la lógica de



/conexión del registro Segundo Operando con el registro de Instrucción Presente, de manera tal que se descartó la posibilidad de tener operandos cortos, por lo menos en la primera versión de esta máquina. Esa posibilidad queda para implementarse en el desarrollo de una futura minicomputadora mas poderosa.

El formato de instrucción sufrió menos cambios durante la evolución del proyecto, los bits 15 a 12 del byte 1 indicarían el código de instrucciones, el bit 9 el tipo de acceso (directo o indirecto); el bit 10 página cero o página / presente; el tipo de operando (corto o largo) estaría determinado por otro bit, quedandonos por lo tanto un último bit en el byte uno cuya función no estaba bien definida.

Los usos podrían ser:

- a) para ampliar el código de instrucciones.
- b) para permitir el acceso directo no sólo a página presente y página cero sino también a página uno y página dos.
- c) para permitir a la máquina trabajar con dos acumuladores indicando es ese caso sobre cual se trabaja.
- d) considerarlo como continuación del byte cero ampliando la parte de dirección.

La más aceptable era la opción b) que hacía a la máquina mas rápida al permitir el acceso directo a mayor cantidad de páginas; además si se tiene en cuenta que esta máquina puede ser la avanzada de proyectos futuros, podría cambiarse el uso de este bit e incorporarlo al código de instrucciones, duplicando de esa manera el mismo.

La posibilidad c) se descarta por economía; la d), que también aumenta las posibilidades de acceso directo, introduce complicaciones en la programación; ellas se debena que una técnica de uso común es modificar por programación el contenido del byte de la parte de dirección de la instrucción de un salto incondicional, permitiendo de esta forma que/ con la misma instrucción podamos saltar a distintos puntos del programa determinados durante su ejecución; si la dirección se completa con un bit en el segundo byte, esta posi-

bilidad, tan útil, se hace de ejecución muy complicada. Por último se fija que los bits 10-11 determinan acceso a / página presente, cero, uno y dos, y el bit 8, operando corto o largo.

#### 4) RESUMEN DEL PROYECTO:

El objetivo fundamental del proyecto es desarrollar un sistema de interfase con capacidad de interrupción, pero se / han elaborado además otros circuitos relacionados con la consola y con los equipos periféricos los cuales serán tra tados posteriormente como así también el apoyo de progr ma ci ón necesario para la entrada y salida (E/S) de informa- / ción.

La arquitectura de la minicomputadora desarrollada es de un único canal de acceso a memoria manejado por la unidad de control UC. Debido a la diferencia de velocidad en el procesamiento de información entre la UC y los equipos de entrada y salida es imprescindible la capacidad de interrupción. Esto significa que la UC una vez que pone en marcha el equipo de Entrada y Salida continúa con la ejecución del programa; cuando el equipo periférico completa la tarea que se le ha encargado avisa a la UC que la información está / lista; al terminar la ejecución de la instrucción, en marcha en el momento de recibir la interrupción, la UC detiene el programa que estaba ejecutando, introduce o saca la información al periférico y continúa con la ejecución normal. De esta manera el manejo de información es siempre a la velocidad de la UC.

El tiempo utilizado en cada interrupción, depende de lo elaborado del sistema. Como puede darse el caso de que varios equipos interrumpan simultáneamente, es necesario un sistema de prioridad para los distintos niveles de interrupción. Además, mientras se introduce o saca la información, es necesario utilizar los registros de la máquina, en especial el registro C y el registro A, para realizar las operacio-

nes asociadas con la interrupción; por lo tanto deben preservarse sus contenidos antes de iniciar la ejecución de la interrupción; para ello existen diferentes técnicas.

Las características del sistema desarrollado son: el tiempo que transcurre desde que se presenta un pedido de interrupción hasta que es atendido por la UC es a lo sumo de / una instrucción; la estructura de prioridades admite interrupción de interrupciones por equipos de mayor prioridad; el contenido de los registros es preservado por programación.

La interconexión de los equipos con el sistema de interrupción se realiza mediante una tarjeta por cada equipo, denominada tarjeta de control, de fácil acceso desde el exterior; la ubicación relativa de la tarjeta en la caja que las contiene determina la prioridad del equipo al cual corresponde; por lo tanto la prioridad de un equipo se varía cambiando la posición de la tarjeta. La tarjeta en la posición cero corresponde al equipo de mayor prioridad; las señales de dichas tarjetas alimentan el sistema de interrupción.

Los equipos periféricos se manejan solamente con tres instrucciones de 1 byte que son: ap, te, y ts.

Esas instrucciones se decodifican según los bits 15-12 y los bits restantes del byte permiten identificar un máximo de 16 periféricos.

## 5) DISEÑO LOGICO:

### 5-1) Tarjetas de Control:

La tarjeta es individual de cada equipo; sin embargo en todas las tarjetas existe un flip-flop C de control, y un / flip-flop IP denominado de interrupción pendiente; además de esto, en la tarjeta deberán estar todos los elementos que permitan el correcto funcionamiento de los equipos.

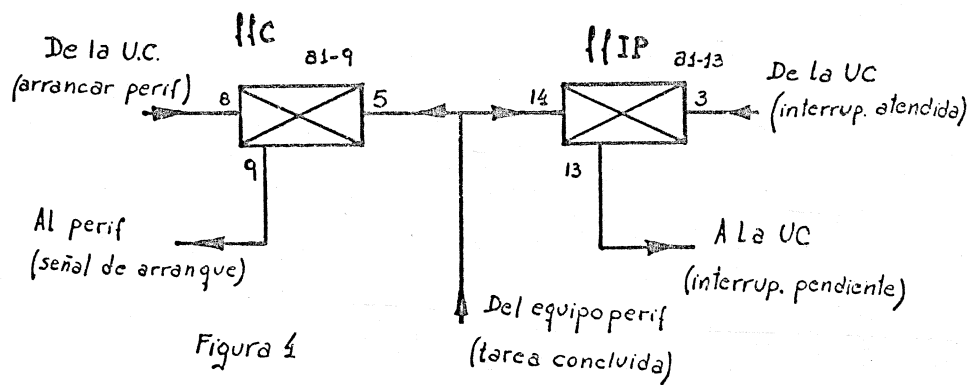
Hasta el momento sólo se manejan dos equipos, uno es el lector Ferranti de cinta de papel (300 caracteres/segundo) que tiene en su interior cinco flip-flops que almacenan la información de la cinta perforada en cada lectura; por lo tanto su tarjeta de control se reduce a los flip-flops de control e IP.

El otro es un perforador de cinta, marca Greed, de 30 caracteres por seg.; su tarjeta de control consta del flip-flop C, el flip-flop IP y además de cinco flip-flops que constituyen el registro del perforador, cuyas salidas alimentan las bobinas de los punzones perforadores.

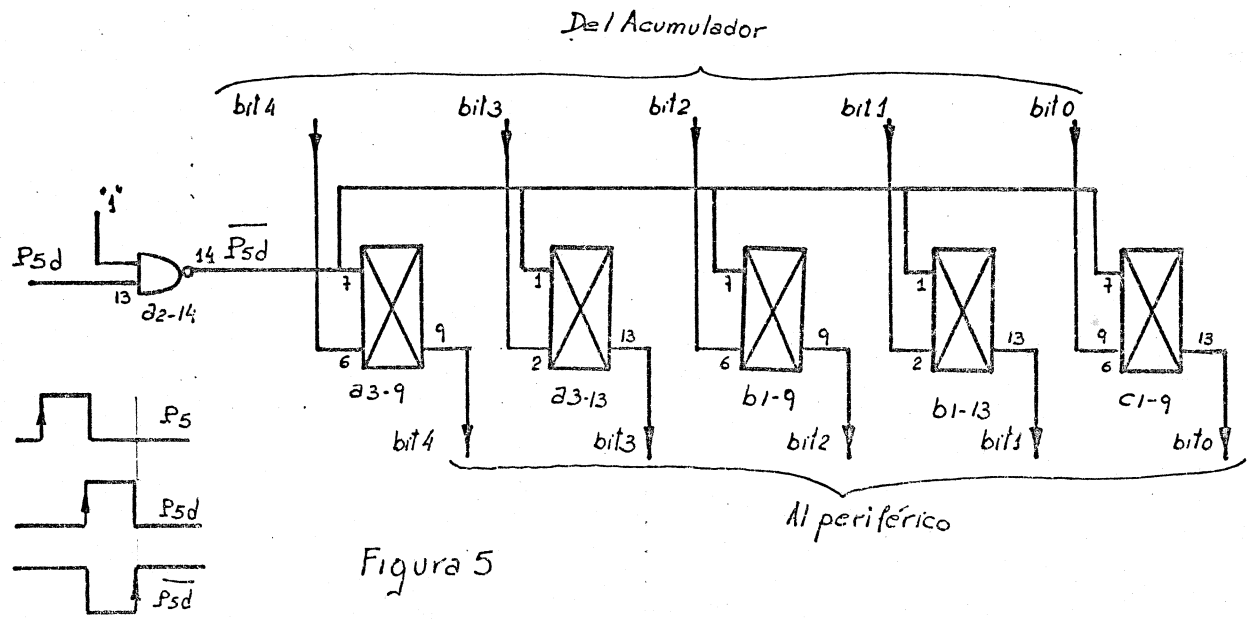
La forma de trabajar es la siguiente: Presentada la instrucción de arrancar periférico en el registro P, con el pulso P5 se vuelca por preset el flip-flop de control del equipo correspondiente poniendo su estado en 1; esa señal debidamente adaptada por circuitos colocados dentro de los equipos y que serán detallados mas adelante, permite arrancar el periférico. Cuando el periférico termina, mediante circuitos simples que también están en su interior, emite una señal (un pulso de cero  $\overline{1}$ ) que vuelva el flip-flop C por CLEAR y simultáneamente al flip-flop IP por PRESET. Es decir que en este momento tenemos al flip-flop C en estado cero y al flip-flop IP en estado uno. El estado del flip-flop IP alimenta el sistema de interrupción, indicando que hay una interrupción pendiente. Al ser atendida por la UC el estado del flip-flop IP retorna a cero como lo indica la siguiente tabla:

FLIP-FLOP C	FLIP-FLOP IP	FUNCION
0	0	El equipo está listo para arrancar.
1	0	El equipo comenzó la lectura o escritura de un caracter.
0	1	La operación está terminada hay un pedido de interrupc.

El estado 1-1 es una combinación prohibida, dado de que el equipo no puede ponerse en funcionamiento si no ha sido atendida la interrupción anterior.  
El diseño entónces es el indicado en la figura 4.



Esta parte es común para ambas tarjetas, pero en la correspondiente al perforador faltan los registros que mantienen la información a ser perforada.



Con la instrucción ts en el registro P y con el pulso P5 desplazado (P5d) cargamos la información del Acumulador en los registros de cinco bits. Figura 5.

Los flip-flops utilizados son del tipo D ( ver aclaración punto 5-4 figura 9), que copian con el frente positivo; de ahí la necesidad de trabajar con P5d e invertirlo; de esta manera estamos seguros que al llegar el frente positivo por la entrada reloj al flip-flop, la información por la entrada D estará perfectamente colocada. Esa información viene del Acumulador a través del canal y fué habilitada con el pulso P5.

La información de la ubicación de los elementos en la tarjeta, como de los pines de salida de cada tarjeta están en el punto 6-1 del informe.

Debe tenerse bien en claro entonces que cada tarjeta es individual para cada equipo, está cableada por el conector de salida de la caja al sistema de interrupción dentro de la máquina y además a una ficha de 8 patas con las cuales se conecta con el periférico. Las fichas están fijadas en un costado de la caja y han sido cableadas de tal manera, que ambos equipos pueden enchufarse en cualquiera de ellas, debiendo cambiarse la posición de la ficha juntamente con la de las tarjetas para intercambiar el orden de prioridad.

#### 5-2) Sistema de Interrupción:

Ahora que tenemos en claro la forma en que se arrancan los periféricos y presentan sus pedidos de interrupción vamos a ampliar el estudio del sistema de interrupción.

Cuando el sistema está ejecutando un programa, tendrá una cierta información necesaria en cada uno de sus registros; ésta deberá ser preservada al presentarse la interrupción. Existen por lo tanto dos problemas a resolver: el primero, que la interrupción es asincrónica respecto de la máquina y el segundo, que de alguna forma debemos modificar la ejecución del programa sin alterar el contenido de sus registros.

Para entender como se sincroniza la interrupción debemos profundizar el conocimiento de la UC en la parte que nos

interesa.

La ejecución de cada instrucción requiere un compás de pulsos; debido a que hay instrucciones cortas y largas, ese compás es variable y puede tener seis o nueve pulsos de reloj respectivamente numerados desde 0 a 5 u 8; por lo tanto P5 es el último pulso de una instrucción corta y P8 el último de una instrucción larga.

El vuelco del flip-flop IP es asincrónico y puede producirse durante la ejecución de una instrucción; la UC sin embargo, no puede dejar por la mitad la ejecución de la instrucción para atender la interrupción; lo que se hace es que con el último pulso de todas las instrucciones (que será P5 ó P8 según la longitud de la misma), inspeccionamos el estado de todos los flip-flops IP copiándolos en otros flip-flops (uno por cada equipo). Las salidas de esos flip-flops van a una gran compuerta intersectora complementadora (NAND) cuya salida le indica a la UC que hay uno o varios equipos pidiendo interrupción. Con esa señal la UC detiene la ejecución del programa y utiliza los compases siguientes para atender la interrupción.

Como dijimos, el pedido puede ser debido a que varios equipos interrumpen simultáneamente; el sistema de interrupción está provisto de un sistema lineal de prioridades que se consigue cortando el estado del flip-flop de un nivel con el complemento de los flip-flops de niveles superiores; las compuertas del sistema de prioridades son inspeccionadas con el primer pulso (P0) del siguiente compás; cuando está atendiendo una interrupción, el pulso no varía el contenido del registro C (en ejecución normal lo incrementa en uno su contenido).

El segundo problema era, como dijimos, presentar la interrupción sin variar el contenido de los registros. Existen por lo menos dos formas de hacerlo. En la primera, debido a que hemos sincronizado la presentación de la interrupción con la finalización de la ejecución de la instrucción, el contenido del registro P puede ser modificado, colocando allí la dirección de la primera instrucción de la interrupción,





to a la subrutina correspondiente a esa interrupción. Hasta ahora, para simplificar el problema, hemos dejado de lado la explicación de la forma en que permitimos la interrupción de interrupciones por pedidos de mayor prioridad. Las subrutinas de interrupción que son distintas para entrar y sacar datos tienen alrededor de 29 instrucciones, siendo la última instrucción de todas las subrutinas un salto incondicional indirecto.

Resumiendo, volcado un flip-flop IP, su estado es cargado a los flip-flops sincronizadores (ff S) de la interrupción con el último pulso de la instrucción en ejecución; con el pulso  $P_0$  (primer pulso del compás siguiente) examinamos la cadena de prioridades, la salida por la compuerta seleccionada vuelca el flip-flop IP y presenta a memoria, por las líneas de C, una dirección trampa que contiene una instrucción de salto a la subrutina de interrupción asociada con el dispositivo que presentó la señal. Las subrutinas son de 29 instrucciones y la última es un salto incondicional indirecto con el cual retornamos al programa original. Todo esto es importante para entender los nidos de interrupción.

La forma de hacerlo es la siguiente: el pulso P5 (ó P8) que sincroniza la interrupción, no entra directamente a la entrada reloj de los flip-flops que cargan el estado de los flip-flops IP, sino que pasa por otra cadena de prioridad la cual impide cargar el estado de un flip-flop IP de menor prioridad que el que corresponde a la subrutina de interrupción en ejecución.

Al ser atendido un pedido de interrupción, hay una salida por una sola de las compuertas que forman parte de la cadena de prioridad principal; esa señal, que hasta ahora cumple dos funciones, la utilizamos para producir el vuelco de otro flip-flop, uno por cada nivel (excepto el último que no debe bloquear a ningún otro por ser el de menor prioridad). Las salidas de estos flip-flops de bloqueo alimentan la segunda cadena de prioridad, formada igual a la anterior, con el estado del flip-flop de un nivel y el complemento de todos los anteriores. La cadena es recorrida por el último /

pulso de todas las instrucciones (P5 ó P8) de tal manera que una vez que se comienza a atender la interrupción se produce automáticamente el vuelco del flip-flop de Bloqueo (ff B) correspondiente a ese nivel. De esta manera se impide que el vuelco de flip-flops IP de equipos de menos prioridad puedan ser cargados por P5 ó P8 en los flip-flops S. Así, la salida de la compuerta intersectora a la cual van los complementos de los estados de los flip-flops S, sigue siendo un cero y por lo tanto la UC no ve ningún pedido de interrupción.

Los flip-flops de bloqueo son restablecidos por el último pulso de la instrucción de salto incond. ind. que es la última instrucción de todas las subrutinas de interrupción. Al sistema, que puede parecer correcto, le quedan sin embargo dos fallas a resolver.

La primera: Puede darse el caso de que dos equipos periféricos interrumpen al mismo tiempo, volcando el flip-flop IP correspondiente a cada nivel antes de que aparezca el último pulso P5 ó P8; por lo tanto, con dicho pulso cargamos los estados de IP en los flip-flops sincronizadores correspondientes a cada nivel; a la salida de la compuerta intersectora habrá en 1 indicando que hay pedido de interrupción a la UC y ésta con el pulso  $P_0$ , determinará el equipo de mayor prioridad, restablecerá el ff IP correspondiente y ejecutará la primera instrucción de esa interrupción. Sin embargo, al terminar la ejecución de esa instrucción, la UC ve un pedido de interrupción del otro equipo de menos prioridad (que no fue atendido), detiene la ejecución de la subrutina y con  $P_0$  explora a que dispositivo corresponde el pedido y coloca por las líneas de C la dirección trampa asociada con el nuevo nivel. Recién cuando complete la ejecución de la subrutina de este pedido de menos prioridad continuará con el anterior que era realmente de los dos el equipo de mayor prioridad. La forma de evitar esta falla es produciendo el restablecimiento de todos los flip-flops de sincronización con el pulso  $\overline{P_1}$ . En consecuencia  $P_0$  limpia el flip-flop IP del nivel elegido y  $\overline{P_1}$  limpia todos los flip-flops de sincronización, de manera que para

la UC es como si no hubiera más pedidos.

Cuando se termina con la ejecución de la subrutina, con el pulso P4 se liberan los flip-flops de bloqueo, permitiéndose el paso de P5 ó P8 de la próxima instrucción, que cargará en el flip-flop S el pedido del equipo de mayor prioridad con interrupción pendiente.

El último problema que queda por corregir es la forma de producir la limpieza de los flip-flops de bloqueo, dado que de dejar las cosas tal cual fueron descritas hasta ahora, podría darse el caso, de que no se cumpliera siempre la estructura de prioridad.

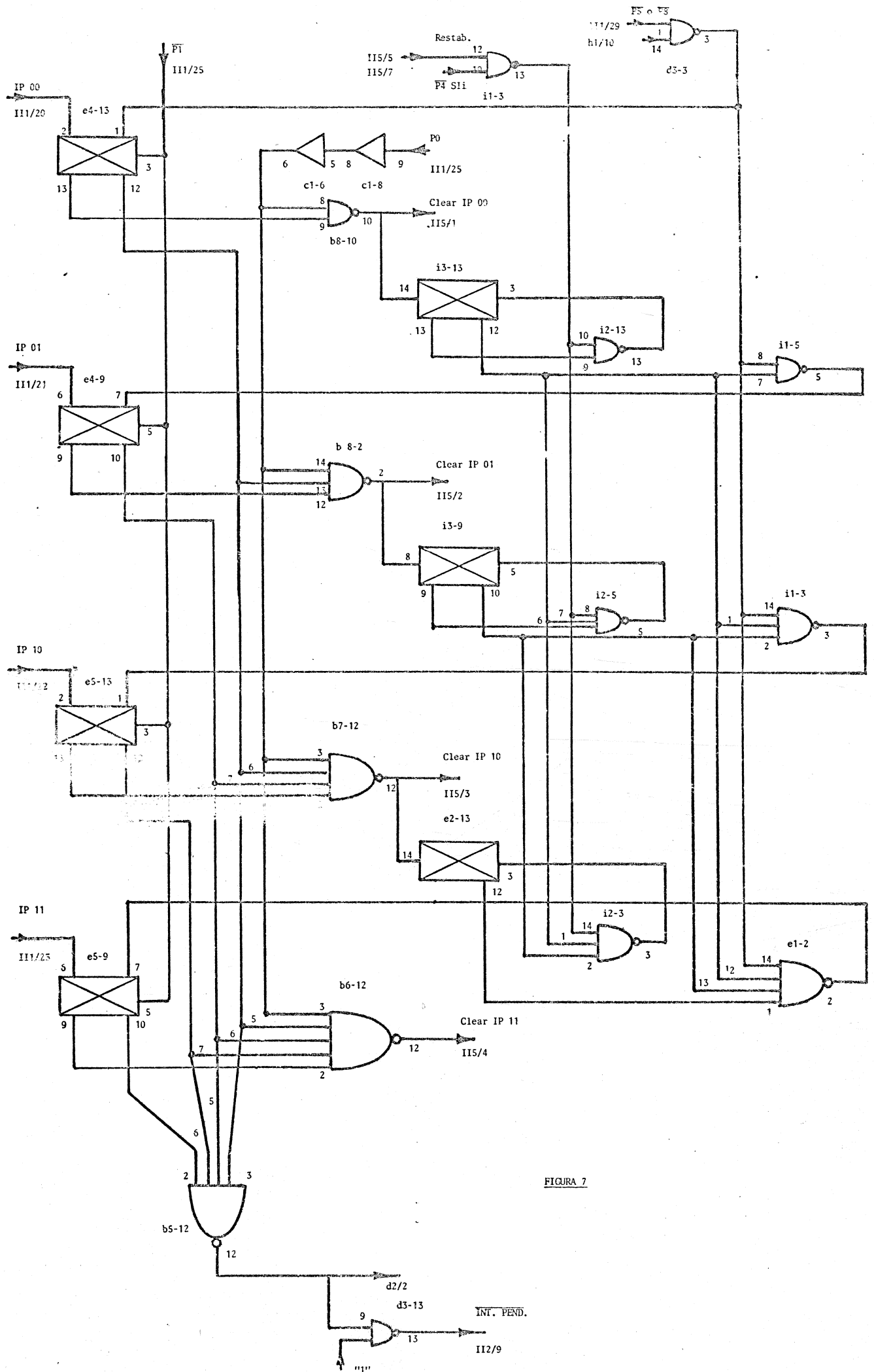
El motivo es el siguiente: Al terminar la ejecución de una subrutina que interrumpió otra de menor prioridad, se liberan los flip-flops de bloqueo de todos los equipos, incluido el correspondiente al equipo que fué interrumpido. En consecuencia, dicho equipo podría ser interrumpido tanto por equipos de mayor prioridad como por equipos de menor prioridad.

Para evitar ésto se incorporarn otra cadena de prioridad, construidas nuevamente con los estados de los flip-flops B y que es recorrida por el pulso ( P4 ) correspondiente a la instrucción de salto incondicional indirecto. La nueva cadena de prioridades hará que al presentarse un caso como el anterior, sólo se limpia el flip-flop IP correspondiente al nivel de mayor prioridad que ya ha sido atendido. El sistema así diseñado es el implementado en la máquina, cumpliendo perfectamente las especificaciones pedidas.

Resumiendo, existen tres tipos diferentes de flip-flops según su uso: flip-flops IP, ubicados uno en cada tarjeta de Control; flip-flops S (de sincronización) y flip-flops B (de bloqueo), uno por cada nivel y ubicados en la máquina.

Existen así mismo tres cadenas de prioridad: una recorrida por Po, que determina realmente la prioridad del equipo a atender; otra recorrida por P5 ó P8, que carga los flip-flops S con la información de los flip-flops IP, y la última recorrida por P4 del salto incondicional indirecto que produce el desbloqueo de los flip-flops B.

Los circuitos referidos anteriormente están en la figura 7 y su información se complementa con las aclaraciones del Pto.6-2.



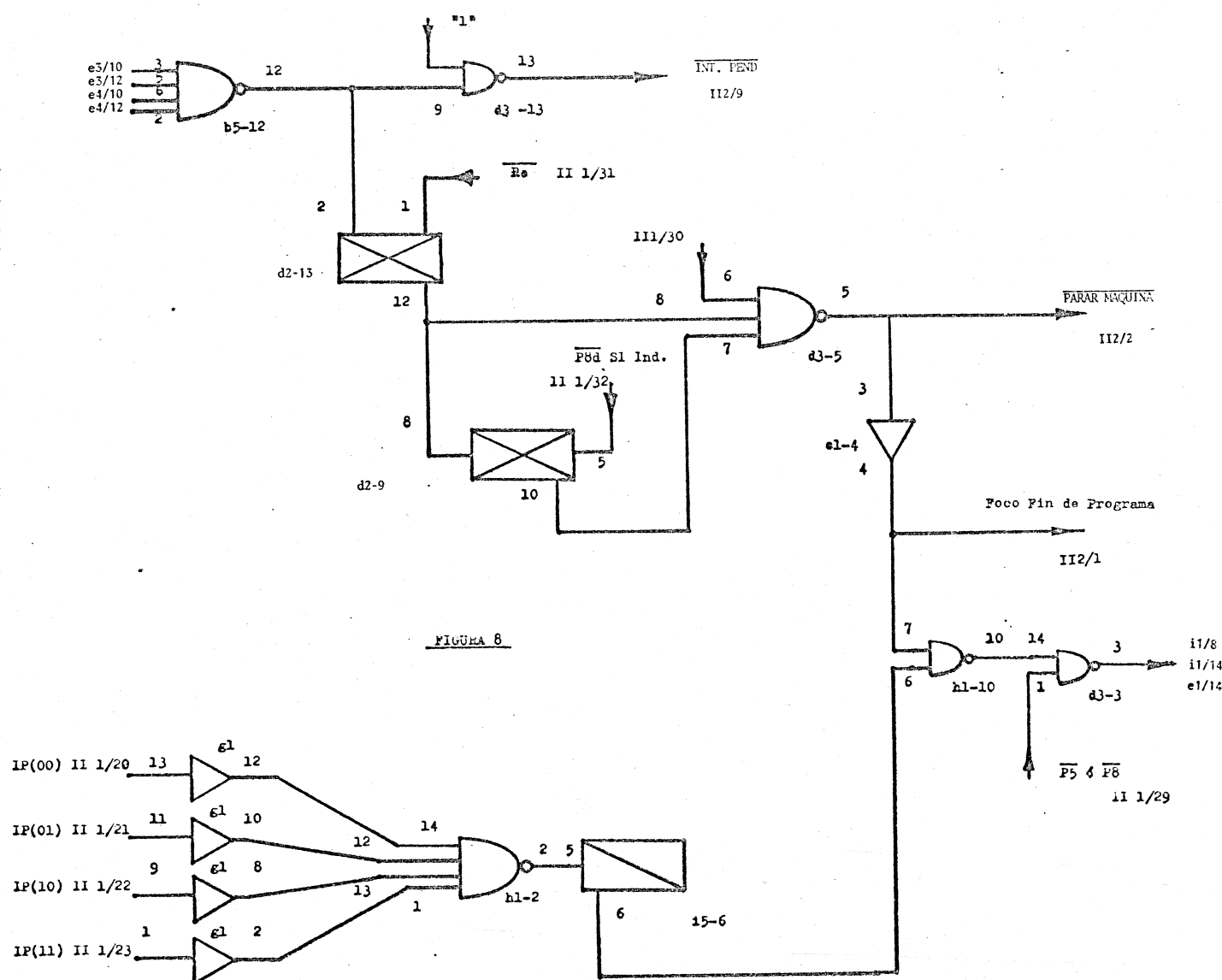
### 5-3) Detención de la máquina:

Una vez que se completa la ejecución del programa aparece la instrucción Parar Máquina (pm) que debería anular el / reloj de la máquina. Sin embargo esto no es correcto dado que lo mas probable es que a pesar de que el programa se ha finalizado faltará imprimir los resultados, es decir / quedará algún equipo periférico en marcha. Esto implica / que la señal de pm será una señal que condicionada por el estado de los periféricos anulará reloj.

Los requisitos que se deben cumplir son dos: 1) Aparecida la instrucción pm, el reloj debe anularse para impedir que aparezca  $P_0$  e incremente uno a C y se comience la ejecución de otro programa. 2) Si había un equipo andando, al terminar volcará un flip-flop IP que no podrá ser cargado en los flip-flops S dado que al estar anulado reloj no habrá P5 ni P8; es decir, necesitamos un pulso que permite cargar los flip-flops de sincronización.

La característica del método es la siguiente: Volcado el flip-flop de parar máquina por la instrucción correspondiente y no habiendo ningún pedido de interrupción pendiente, se anula el reloj no permitiendo la ejecución de la instrucción siguiente. Sin embargo, se le da al reloj la posibilidad de ser habilitado al presentarse un pedido de interrupción por la compuerta NAND de los flip-flops S. La clave del método es cargar el estado de los flip-flops IP en los flip-flops S antes de habilitar nuevamente reloj; sin embargo, por ser una fase de interrupción,  $P_0$  no incrementará en uno el contenido del registro contador de programa.

Mediante un manejo adecuado de las señales, como se indica en la figura 8, hacemos que al volcarse cualquier flip-flop IP se produzca un pulso que circulará por la misma línea de P5 ó P8 de la figura 7, cargando los flip-flops S; las salidas de estos flip-flops a través de la gran compuerta intersectora presentará a la UC un pedido de interrupción y por otra parte habilitará nuevamente reloj; esto significa que  $P_0$  recorrerá la cadena de prioridades sin modificar el contenido del registro C (figura 8).



Hasta ahora las señales son asincrónicas porque cargamos los flip-flops S con una señal proveniente de la unión de todas las salidas de los flip-flops IP que hicimos pasar por la línea de P5 ó P8; sin embargo, como va a habilitar reloj, debe ser sincronizada. La forma de hacerlo es cargar en un flip-flop la salida de la compuerta intersectora de los flip-flops S con reloj complementado y habilitar la compuerta de reloj con la salida de este flip-flop.

Atendida la interrupción con el último pulso de la instrucción de salto incondicional, volcamos este flip-flop anulando nuevamente reloj hasta un nuevo pedido de interrupción.

Existe en la consola una luz de fin de programa que quedará encendida permanentemente sólo cuando el programa esté

concluído, con todos los resultados impresos (en este caso en cinta perforada).

La información de la figura 8 se completa con las aclaraciones del punto 6-2.

#### 5-4) Generación de pulsos y compases:

Para poder ensayar totalmente la máquina y ver la variación de todos los registros en cada pulso, se incluye la posibilidad de correr las instrucciones en forma discontinua ya sea por pulsos o compases, aparte de la forma continua normal. Con tal fin, existen en la consola tres llaves: una selecciona modo continuo o discontinuo, otra por pulso o compás y la tercera da el pulso de arranque.

Existe una cuarta llave denominada de restablecimiento que debe ser apretada al iniciar el funcionamiento de la máquina o cada vez que se termine la ejecución de un programa; esa llave tiene como función colocar algunos flip-flops en estados preestablecidos.

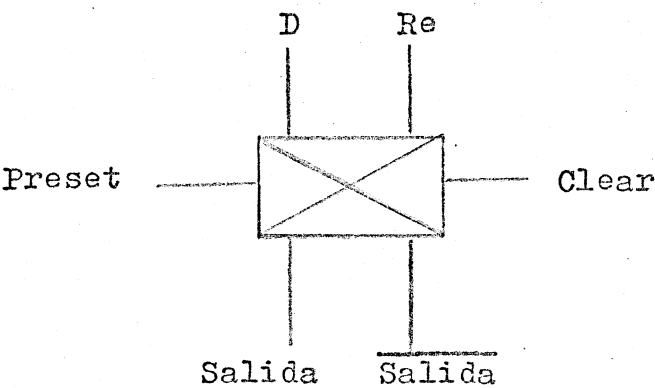
La forma de operar es la siguiente:

- 1) Pulsar restablecimiento.
- 2) Seleccionar el modo continuo o discontinuo.
- 3) De ser este último, seleccionar compás o pulso.
- 4) Pulsar la llave de arranque.

Sea cual fuere el modo, siempre se debe pulsar ARRANQUE para habilitar reloj.

En modo continuo, el reloj sólo se detiene al aparecer la instrucción pm y cumplirse las condiciones del punto 5-3, lo cual haría las pruebas muy engorrosas, especialmente no habiendo memoria; por lo tanto, provisoriamente, pulsando la llave que selecciona el modo continuo-discontinuo hacia abajo (tercera posición de la llave) anulamos reloj; al parar la máquina de esta forma el generador de pulsos de la UC puede quedar en cualquier estado y nada se consigue pulsando la llave de restablecimiento dado que, como dijimos, esto es sólo provisorio; lo que debe hacerse es completar el compás mediante el uso adecuado de las llaves, quedando, ahora sí, la máquina lista para funcionar.

Para implementar los circuitos se utilizaron flip-flops D, (SN 7474) la convención sobre la forma de graficar las entradas como así también su tabla de verdad, es la indicada en la figura 9.



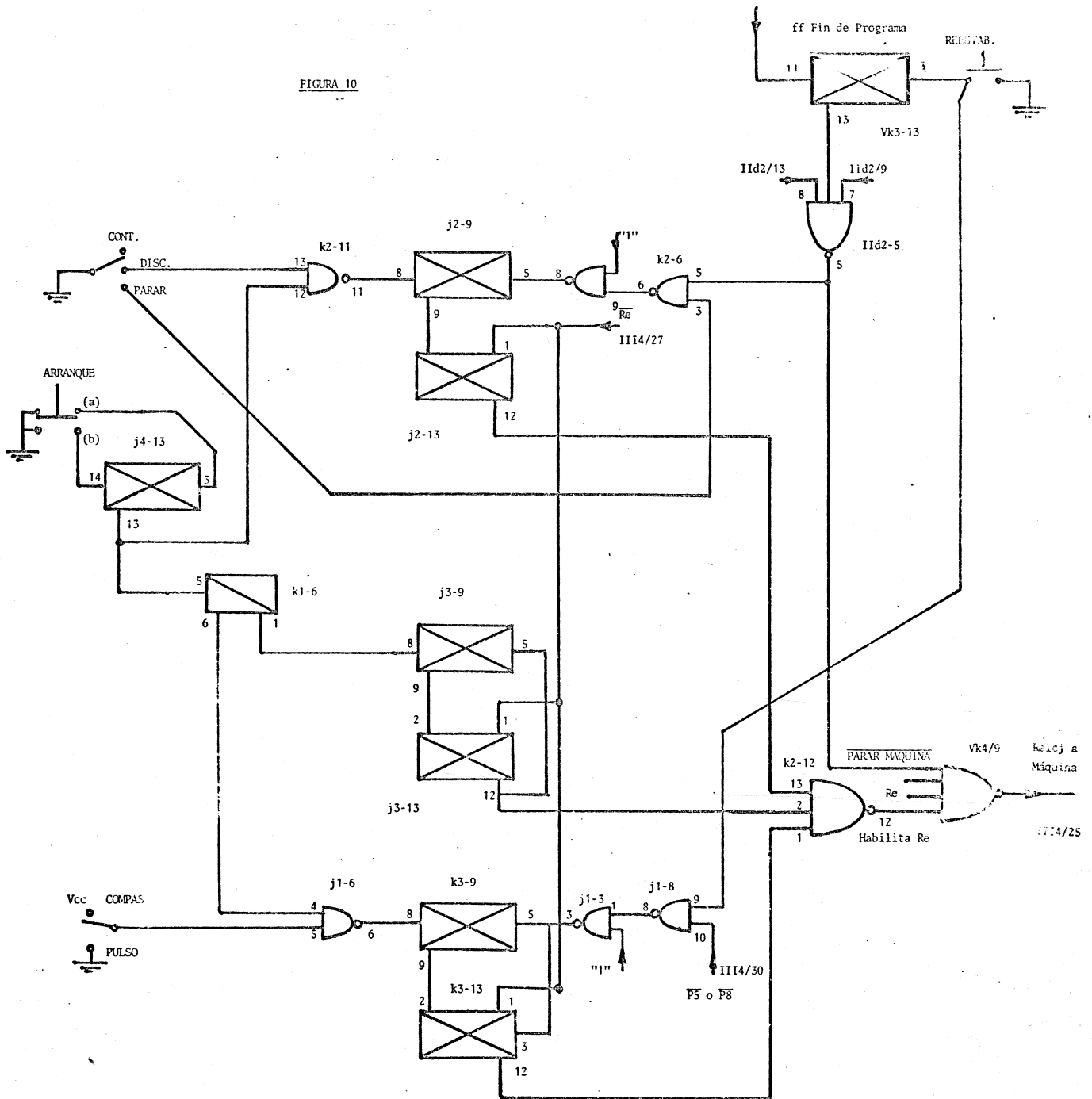
$t_n$	$t_{n+1}$	
Entrada D	Salida	Salida
0	0	1
1	1	0

Una entrada "baja" en Preset coloca en la Salida un 1 lógico.  
Una entrada "baja" en Clear coloca en la Salida un 0 lógico.

La forma de operar de los circuitos de comando es la siguiente (todos pertenecen a la tarjeta III, símbolo que en adelante se omite). El pulsador de arranque trabaja sobre el flip-flop j4-13 y no directamente sobre el monoestable kl-6 para evitar los posibles rebotes de la llave y su disparo por ruidos externos. Esto último se logra porque la entrada de CLEAR del flip-flop j4-13 está constantemente anclada a cero / mientras no se pulse ARRANQUE. (Figura 10)



FIGURA 10



Del monoestable utilizamos la SALIDA (normalmente en 0) y la SALIDA ; con la SALIDA (pata 6) volcamos a 1 (establecemos) el flip-flop k3-9, y con la SALIDA establecemos el el flip-flop j3-9; como siempre, el estado del primer flip-flop es transferido al segundo (k3-13 y j3-13 respectivamente) con reloj complementado ( o retardado).

Lo expresado anteriormente sucederá sólo si estamos en modo Discontinuo-Compás; si estamos en modo Discontinuo-Pulso el monoestable volcará sólo los flip-flops del módulo j3. El restablecimiento (puesto en 0) de los flip-flops de / compás (k3) se hace con el último pulso de la instrucción que se ejecutó (P5 ó P8); el restablecimiento de los flip-flops de pulso (j3) es automático como se ve en la figura anterior .

Al operar en forma continua, el flip-flop j4-13, que es / manejado por el pulsador de Arranque, produce el vuelco del flip-flop j2-9. Su estado se carga con reloj complementado al flip-flop j2-13. El restablecimiento de estos flip-flops se efectúa con la señal Parar Máquina y además, por ahora sólo en forma provisoria, pulsando hacia abajo la llave que selecciona modo Continuo-Discontinuo (Posición Parar).

Las salidas de los flip-flops j2/12, j3/12 y k3/12, van a la compuerta k2-12; su salida (k3-12) ha sido bautizada / HABILITA-RELOJ y actúa sobre la compuerta por la que entra el reloj de la máquina.

Falta aclarar la necesidad de la entrada de restablecimiento al flip-flop k3-9; Puede darse el caso de intentar la ejecución de una instrucción por un compás, sin fijarse que estaban prendidas las luces de Fin de Programa y/o Desborde, estando de esta forma el reloj anulado. Por el hecho de haber seleccionado el Modo Compás y haber pulsado ARRANQUE, quedan volcados los flip-flops del módulo k3, que no fueron restablecidos dado que P5 ó P8 no fueron generados por no estar habilitado el reloj.

De no estar conectado el Restablecimiento a esos flip-flops y pulsarse esa llave para apagar las luces y en consecuencia habilitar reloj, se producirá un arranque asincrónico, introduciendo posibilidades de error.

La ubicación de los elementos en la tarjeta está en el punto 6-3.

#### 5-5) Conexionado entre UC, Interfase y Caja de tarjetas de Control:

La UC envía siete salidas al equipo de interfase. Ellas co-

responden a la decodificación de las instrucciones pertinentes y además a los pulsos en que éstas se deben ejecutar. Las siete señales son:

- 1) te, transferir entrada.
- 2) ts, transferir salida.
- 3) ap, arrancar periférico.

4) a 7) los estados correspondientes a la decodificación de cuatro periféricos (00,01,10,11), que son los únicos implementados hasta ahora. Con esto y además el pulso P5, que es el pulso con que se ejecutan las instrucciones ap y te, y el pulso P5d (P5 desplazado o retardado) usado en la instrucción ts, debemos obtener las doce combinaciones correspondientes a las tres instrucciones y los cuatro periféricos. (figura 11).

Entre el control de interfase (que está en la UPC) y la caja de tarjetas de control, existen diversas conexiones, algunas de ellas propias de cada tarjeta y otras comunes a todas las tarjetas. Las propias son doce líneas que comprenden las señales: Pedido de Información, Limpieza de Interrupción, Transferencia de Información, Arranque y 8 líneas de Información. Las comunes a todas las tarjetas son 8 líneas de información que provienen de la salida del árbol. Las líneas Pedido de Interrupción, Limpieza de Interrupción y Arranque son independientes de la función del periférico. La línea de Transferencia de Información se usa sólo cuando se trata de un periférico de salida con registro intermedio (buffer) de la tarjeta.

Las 8 líneas propias de información se usan cuando el periférico es de entrada. Las 8 líneas comunes se usan cuando el periférico es de salida.

La información sobre la ubicación de los módulos en la tarjeta está en el punto 6-2.

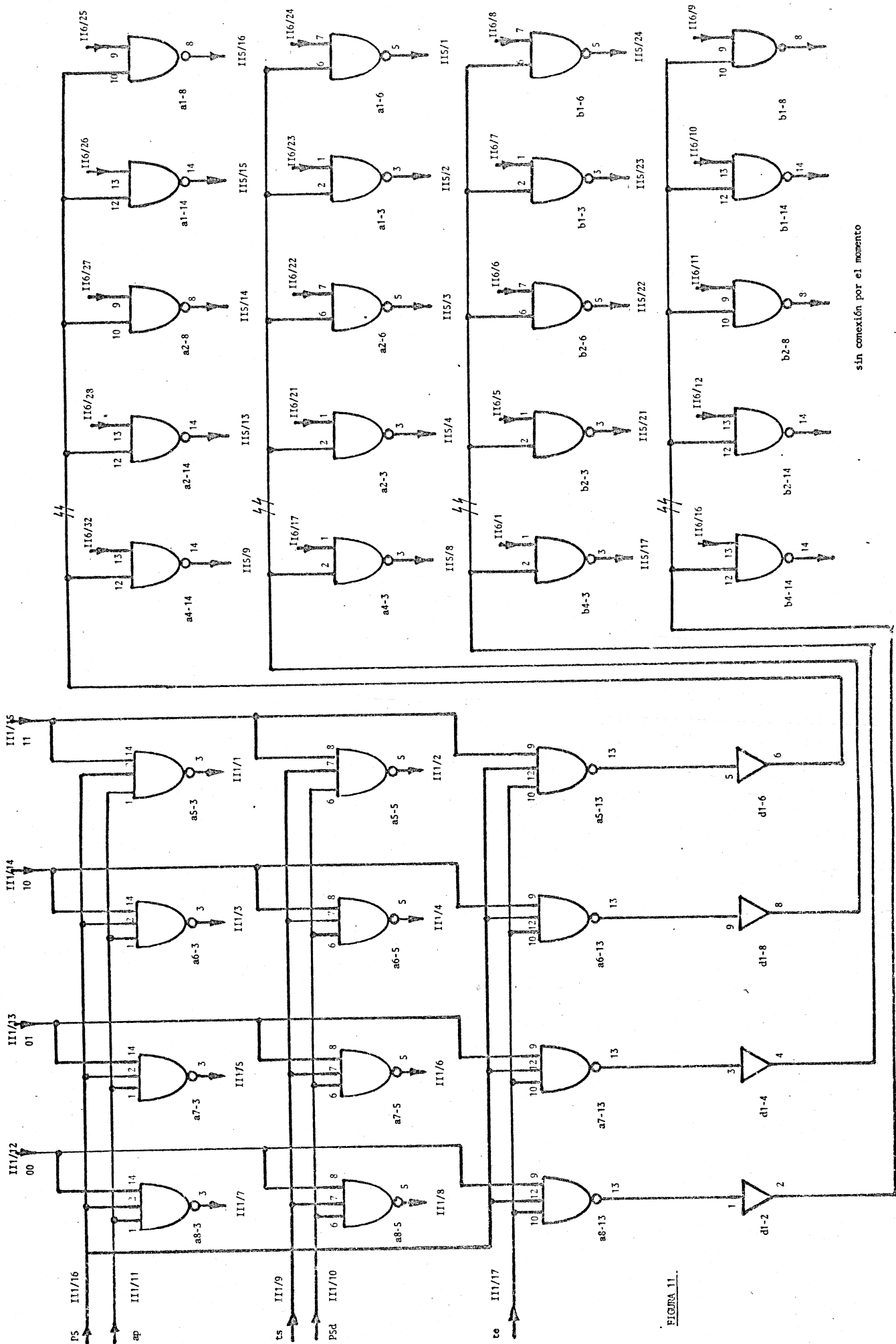


FIGURA 11.



### 5-7) Tarjeta Interfase Equipo Lector:

Debido a que el equipo lector trabaja con lógica negativa, fué necesario diseñar los circuitos para adaptar sus salidas a la lógica TTL con la cual se implementaron las tarjetas de control. Esos circuitos están ubicados en una tarjeta de 24 pines que fué colocada en el exterior del lector debidamente blindada al ruido. La alimentación de las tensiones necesarias se hace desde el exterior a través de una manguera con blindaje para reducir al mínimo los problemas de ruido.

Las señales que debieron ser adaptadas fueron siete:

- 1) La señal de arranque, dado que el flip-flop de control en la máquina da un escalón de 0 a +5 V mientras que el equipo lector necesita un escalón de 0 a -5 V.
- 2) La señal que vuelca por CLEAR al flip-flop C y por PRESET el flip-flop IP, que debe ser un pulso de cero, mientras que el equipo genera una señal de -5 V por un tiempo de 4,5m seg.
- 3) a 7) Las señales de salida, correspondientes a los estados de los flip-flops que vienen con el equipo son 1,2 V y -3,5 V correspondientes a un uno y un cero respectivamente, mientras que el sistema de interfase necesita +5 V y 0 para distinguir los estados.

La información sobre la ubicación de los elementos (discretos en este caso) en la tarjeta, y de los pines de salida está en el punto 6-5.

### 5-8) Tarjetas Interfase Equipo Perforador:

Para adaptar las señales de salida y de entrada a la tarjeta de control, se diseñaron los circuitos que se explican a continuación.

Los circuitos implementados fueron ubicados dentro mismo del equipo perforador, para lo cual fué modificada su estructura física.

Se utilizaron tres tarjetas de 32 pines las cuales fueron numeradas de 1 a 3 estando el número adherido a cada tarje-

ta y grabado en el bastidor que la contiene.

Las señales adaptadas fueron siete:

- 1) El flip-flop de control da un escalón de 0 a 5 V. mientras que el equipo usa señal de arranque de 0 a 45V.
- 2) Para generar la señal de cero (L) que vuelca los flip-flops C e I P el perforador tiene dos contactos normalmente abiertos que luego del arranque se cierran durante 6ms; además es necesario esperar 10ms. para iniciar la perforación de un segundo caracter desde que se cierran esos contactos.

La forma de generar esa señal es la siguiente: Los contactos son alimentados con una tensión de -5 Volts a través de una resistencia ubicada en la tarjeta 3 como indica la fig. 13 a.

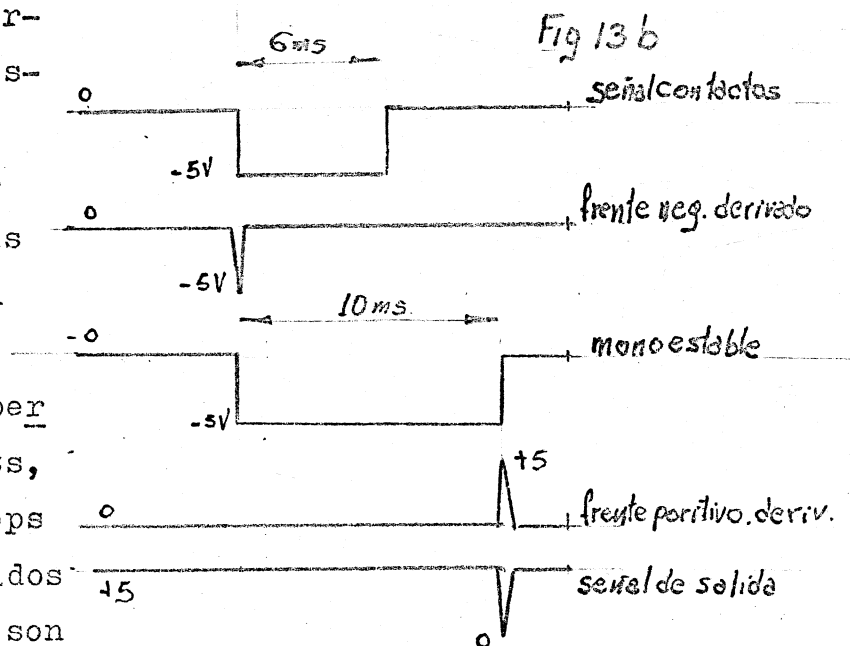
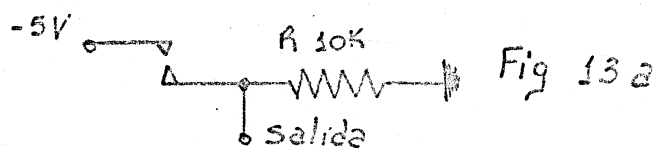
Por la forma en que se conectó, la señal que se obtiene con los contactos abiertos (posición normal) es un cero; cuando se cierran se produce un escalón de -5 Volts por un tiempo de 6 ms.

Un monoestable de 10 ms.

se vuelca con el frente negativo que producen los contactos al cerrarse; al terminar el tiempo del monoestable su salida derivada por un circuito pasador e invertida para adaptar las señales, produce la señal deseada (figura 13b).

3) a 7) Las bobinas del perforador necesitan 45 Volts, mientras que los flip-flops del Registro BUFFER ubicados en la tarjeta de control son

de lógica TTL (5 Volts). La adaptación de los niveles se consigue con los transistores AF 115 y ASY 77 ubicados en las tarjetas 2 y 1 respectivamente. Los 5 Volts de salida de los flip-flops son transformados a una señal de -5 Volts por la primera etapa hecha con transistores AF 115; esa señal, ahora



con la polaridad adecuada, alimenta los transistores de potencia ASY 77, cuya salida da los 45 Volts que necesitan las bobinas.

En la tarjeta 2, hay seis transistores AF 115, uno para cada bit y otro para la señal de arranque, que trabajan en la misma forma.

Los seis transistores ASY 77 de la tarjeta 1 son guiados por los AF 115 correspondientes y alimentan la bobina de arranque y las 5 bobinas que manejan los punzones perforadores.

La información sobre los circuitos está en el punto 6-6.



## 6) IMPLEMENTACION ELECTRONICA:

Con el objeto de poder seguir todos los circuitos explicados anteriormente en la máquina, este inciso incluye la ubicación de los elementos en la tarjeta, la información sobre qué señal o estado llega a cada pata de cada módulo y la relación de éstos con los pines de salida.

La máquina está organizada, por el momento, en cinco tarjetas, las cuales están marcadas con números romanos; cada tarjeta posee seis conectores de 32 pines cada uno; tres conectores son fijos al gabinete y los otros tres móviles para permitir la extracción de la tarjeta; los conectores están numerados de 1 a 6.

El gabinete está construido con dos puertas. En la puerta frontal va el tablero en la parte inferior y la tarjeta I, en la parte superior; en la puerta posterior están ubicadas las tarjetas IV en la parte superior y la tarjeta V en la inferior. Las otras dos tarjetas están ubicadas en forma similar a las anteriores pero en el interior del gabinete, la II en la parte superior y la III en la inferior; queda lugar para otras dos tarjetas que es donde será ubicada la memoria.

Se ha tenido el cuidado de que todos los circuitos sean de cómodo acceso desde el exterior haciendo muy fácil la detección de fallas. Cada módulo puede ser identificado en la tarjeta, por una letra minúscula y un número; la letra indica la columna a la cual pertenecen y el número determina su fila. La cantidad de columnas es particular de cada tarjeta y además la cantidad de filas de cada columna es variable dentro de la misma tarjeta.

Como en un mismo módulo puede haber uno o más flip-flops, o varias compuertas etc., los circuitos serán identificados por el número de la pata de salida, el cual va después de un guión que sigue a la letra y el número anterior; ejemplo: a 12-6 indica el módulo ubicado en la columna a fila 12 y al elemento dentro del módulo cuya salida es por la pata 6. La identificación total sería, ejemplo: II a 12-6 con lo cual decimos que el elemento anterior está en la tarjeta II.

Si quisiéramos referirnos a alguna pata del módulo pondríamos: II a 12/1 y de ésta forma hacemos referencia a la pata 1 del módulo ubicado en la columna a fila 12 de la tarjeta II.

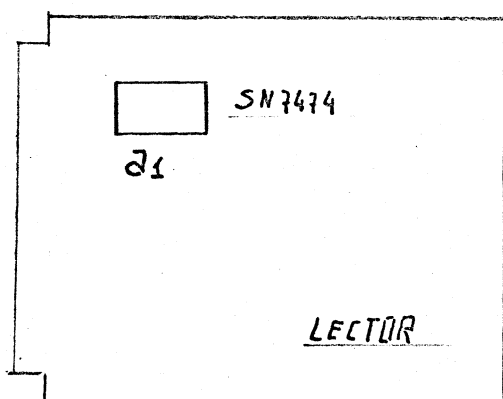
Los circuitos que se describieron en el punto 5 están ubicados algunos en las tarjetas II y III y otros en las tarjetas de control, ubicadas en el exterior de la máquina y los restantes en las tarjetas colocadas dentro de los equipos periféricos.

La distribución de todos los elementos en las tarjetas II y III se muestra a continuación en la figura 14a y b respectivamente.

#### 6-1) Tarjetas de Control:

Las tarjetas de control no se pueden identificar con un número ya que éstas toman el número de la posición en que estén ubicadas en la caja. Por el momento sólo son dos y están identificadas por el nombre del equipo al cual pertenecen, ( LECTOR, PERFORADOR).

##### 6-1-1) Tarjeta de Control Equipo Lector:



a 1-9 ffC

a 1-13 ffIP

Pines de salida(en el conector)

- 1) Clear IP
- 2) Clear IP-Preset C
- 3) +5V.
- 4) Salida flip-flop IP
- 5) Salida flip-flop  $\overline{IP}$  (no se usa por modificaciones posteriores).
- 6) Tierra.
- 7) Salida flip-flop C
- 8) Preset C
- 9) a 32) Sin conexión.



Figure 14a

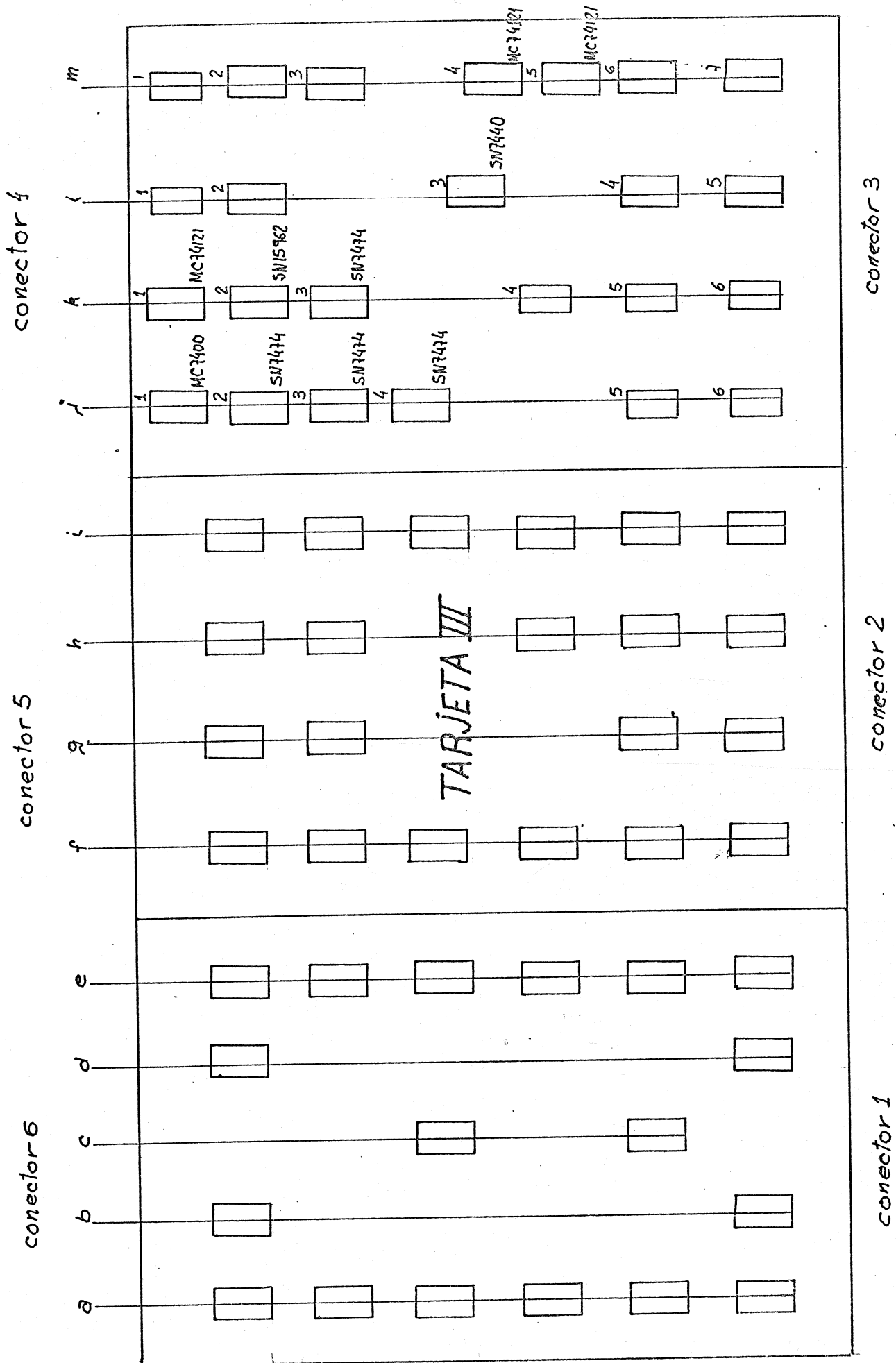
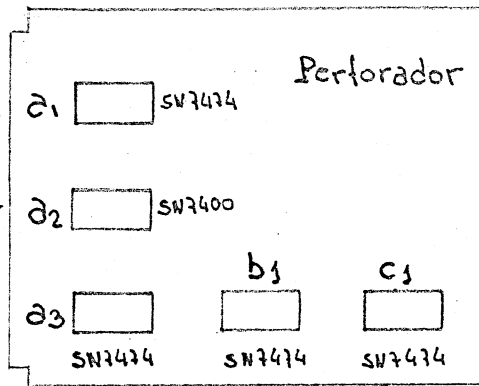


Figura 14b

6-1-2) Tarjeta Control Equipo Perforador:

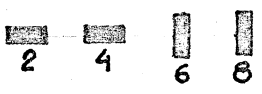
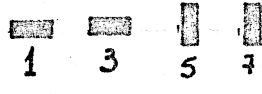


- a1-9 ffC
- a1-13 ffIP
- a2-14 Inversor (produce  $\overline{P5d}$  de ap)
- a3-9 ff bit 4
- a3-13 ff bit 3
- b1-9 ff bit 2
- b1-13 ff bit 1
- c1-9 ff bit 0

Pines de Salida (en el conector)

- 1) a 8) Idem.punto 6-1-1
  - 9) Sin conexión.
  - 10) P5d de ap      a2/13
  - 11) bit 0              c1/6
  - 12) bit 1              b1/2
  - 13) bit 2              b1/6
  - 14) bit 3              a3/2
  - 15) bit 4              a3/6
  - 16) a27) Sin conexión
  - 28) bit 4              a3/9
  - 29) bit 3              a3/13
  - 30) bit 2              b1/9
  - 31) bit 1              b1/13
  - 32) bit 0              c1/9
- del Acumulador
- a la ficha de salida

Observación: las fichas de salida de 8 patas con las cuales se conectan los equipos a la caja de control son idénticas y las señales correspondientes a cada pata son las siguientes:

	Pata	Señal
	1	-
	2	Arranque periférico
	3	Clear C -Preset IP
	4	bit 0
	5	bit 3
	6	bit 1
	7	bit 4
	8	bit 2

## 6-2) Tarjeta II

En la tarjeta II están totalmente ubicados:

- 1) Sistema de Interrupción, explicado en el punto 5-2, fig.7
- 2) Detención de la máquina, explicado en el punto 5-3, fig.8
- 3) Instrucciones de periférico, explicadas en el punto 5-5, fig.11

Existen además otros circuitos en esa tarjeta, pero su información corresponde al informe de la Unidad de Control.

## 6-3) Tarjeta III

En la tarjeta III están ubicados los circuitos que permiten la generación de pulsos y compases, explicados en el punto 5-4, figura 10, como así también el reloj (punto 5-6, figura 12) y otros circuitos de la Unidad de Control.

## 6-4) Conectores:

Como dijimos son seis, numerados de 1 a 6 y de 32 pines c/u. Los pines que corresponden a este informe son:

### Tarjeta II

#### Conector 1

- 1) ap 11 amarillo a5/3
- 2) ts 11 marrón a5/5
- 3) ap 10 amarillo a6/3
- 4) ts 10 marrón a6/5
- 5) ap 01 amarillo a7/3
- 6) ts 01 marrón a7/5
- 7) ap 00 amarillo a8/3
- 8) ts 00 marrón a8/5
- 9) ts rojo a8/7, a7/7, a6/7, a5/7
- 10) P5d verde a8/6, a7/6, a6/6, a5/6
- 11) ap blanco a8/1, a7/1, a6/1, a5/1
- 12) 00 azul a8/14, /5, /13
- 13) 01 azul a7/14, /5, /13
- 14) 10 azul a6/14, /5, /13,
- 15) 11 azul a5/14, /5, /13
- 16) P5 naranja a8/2, /12, a7/2, /12, a6/2, /12, a5/2, /12
- 17) te blanco a8/10, a7/10, a6/10, a5/10
- 18) Vcc blanco

- 19) tierra negro
- 20) IP 00 blanco e4/2
- 21) IP 01 blanco e4/6
- 22) IP 10 blanco e5/2
- 23) IP 11 blanco e5/6
- 24)  $\overline{PI}$  blanco e4/3, /5, e5/3, /5,
- 25) Po amarillo c1/9
- 26) a 28) sin conexión.
- 29)  $\overline{P5}$  ó  $\overline{P8}$  gris d3/1,
- 30)  $\overline{Parar}$  máq. pend. blanco d3/6
- 31)  $\overline{Re}$  naranja d2/1
- 32)  $\overline{P8d}$  de Salto incond. ind. verde d2/5

Conector 5:

- |                 |   |
|-----------------|---|
| 1) bit 7 a1/5   | 25) Sin conexión                          |
| 2) bit 6 a1/3   | 26) $\overline{P4}$ de Salto incond. ind. |
| 3) bit 5 a2/5   | 27) Restablecimiento                      |
| 4) bit 4 a2/3   | 28) Idem                                  |
| 5) bit 3 a3/5   | 29) Clear ff IP perif.11                  |
| 6) bit 2 a3/3   | 30) Clear ff IP perif.10                  |
| 7) bit 1 a4/5   | 31) Clear ff IP perif.01                  |
| 8) bit 0 a4/3   | 32) Clear ff IP perif.00                  |
| 9) bit 0 a4/14  |   |
| 10) bit 1 a4/8  |   |
| 11) bit 2 a3/14 |   |
| 12) bit 3 a3/8  | Perif.00                                  |
| 13) bit 4 a2/14 |   |
| 14) bit 5 a2/8  |   |
| 15) bit 6 a1/14 |   |
| 16) bit 7 a1/8  |   |
| 17) bit 0 b4/5  |   |
| 18) bit 1 b4/3  |   |
| 19) bit 2 b3/5  |   |
| 20) bit 3 b3/3  | Perif.10                                  |
| 21) bit 4 b2/5  |   |
| 22) bit 5 b2/3  |   |
| 23) bit 6 b1/5  |   |
| 24) bit 7 b1/3  |   |

Conector 6:

1) bit 0	b4/1	} Periférico 10 (morado)
2) bit 1	b4/7	
3) bit 2	b3/1	
4) bit 3	b3/7	
5) bit 4	b2/1	
6) bit 5	b2/7	
7) bit 6	b1/1	
8) bit 7	b1/7	
9) bit 7	b1/9	} Periférico 11 (naranja)
10) bit 6	b1/13	
11) bit 5	b2/9	
12) bit 4	b2/13	
13) bit 3	b3/9	
14) bit 2	b3/13	
15) bit 1	b4/9	
16) bit 0	b4/13	
17) bit 0	a4/1	} Periférico 01 (blanco)
18) bit 1	a4/7	
19) bit 2	a3/1	
20) bit 3	a3/7	
21) bit 4	a2/1	
22) bit 5	a2/7	
23) bit 6	a1/1	
24) bit 7	a1/7	
25) bit 7	a1/9	} Periférico 00 (amarillo)
26) bit 6	a1/13	
27) bit 5	a2/9	
28) bit 4	a2/13	
29) bit 3	a3/9	
30) bit 2	a3/13	
31) bit 1	a4/9	
32) bit 0	a4/13	

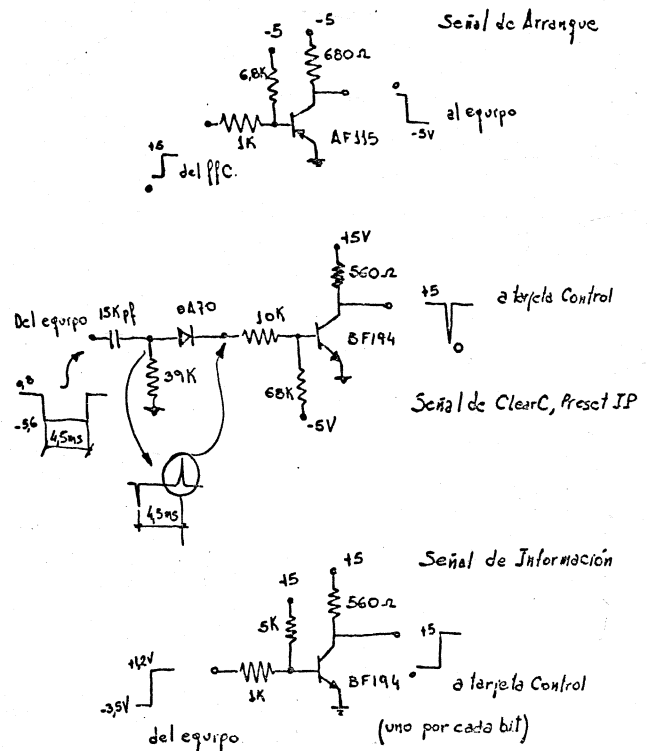


Tarjeta IIIConector 4:

- 32) Cont-Disc. k2/13
- 31) Arranque(b) j4/14
- 30)  $\overline{P5}$  ó  $\overline{P8}$  j1/10
- 29) Restab. j1/9
- 28) Parar Máquina k2/3
- 27) Re j2/1, j3/1, k3/1
- 26) Compás-Pulso k2/5
- 25) Habilita Re. k2/12
- 24) Vcc
- 23) Tierra
- 22) Parar k2/3
- 21) Arranque (a) j4/3

6-5) Tarjeta Interfase equipo lector:

- A) bit 0
  - B) bit 1
  - C) bit 2
  - D) bit 3
  - E) Tierra
  - F) bit 4
  - H) bit 4
  - J) Sin Conexión
  - K) bit 3
  - L) bit 2
  - M) bit 1
  - N) bit 0
  - P) +5 V.
  - R) -5 V.
  - S) Salida señal arranque  
(al equipo)
  - T) Entrada señal de fin  
de lectura(del equipo)
  - U) Sin conexión
- Del Equipo
- Al Acum--  
lador





## 7) APOYO DE PROGRAMACION:

La atención de una interrupción se comienza presentando a memoria la dirección trampa asociada con el dispositivo. En esa locación de memoria se encuentra una instrucción de salto a la subrutina asociada con la naturaleza de la interrupción; es evidente que el contenido de la dirección trampa puede ser cambiado por programación, pudiendo entonces saltar a distintas subrutinas según las necesidades.

La subrutina de Entrada de datos es distinta de la subrutina de Salida; esto implica que a cada dirección trampa asociada con un equipo, le corresponde una subrutina de Entrada y una de Salida, ya que se desconoce la naturaleza del equipo con que va a trabajar esa dirección. Si tenemos 16 "trampas" necesitaremos 32 subrutinas (16 de Entrada y 16 de Salida).

Con esa previsión, desde el punto de vista de E/S estaríamos en un diseño aceptable; sin embargo, la cantidad de memoria necesaria para ubicar las subrutinas sería elevada, mas aún en el caso de una minicomputadora.

El otro caso extremo sería poseer una única subrutina de Entrada y una única de Salida, las cuales serían compartidas por todos los equipos. De adoptar esta posibilidad, que es la que ocupa menos memoria, a lo sumo podríamos tener funcionando, al mismo tiempo, un equipo de Entrada y uno de Salida.

Una solución aceptable sería entonces poseer un término medio entre las dos alternativas mencionadas anteriormente, por ejemplo 8 subrutinas de Entrada y 8 de Salida. Con ello utilizamos una cantidad no muy grande de memoria y el rendimiento bajará recién en el caso que esté funcionando consecuentemente 8 equipos (por ejemplo de Entrada) y se trate de arrancar un noveno equipo.

Las tareas que debe cumplir la subrutina de interrupción son:

- 1) Preservar el contenido de los registros C y A.
- 2) Por medio de una "traba", evitar la posibilidad de que una subrutina que está atendiendo un pedido sea utilizada por otro equipo antes de que termine con el primero.

- 3) Llevar la cuenta de la cantidad de datos que debe entrar o sacar.
- 4) Llevar la cuenta de las direcciones de memoria donde aquéllos deben ubicarse.
- 5) Arrancar de nuevo el equipo en el caso de que todavía no esté completa la tarea.

#### 7-1) Subrutina de Entrada:

Antes de explicar la subrutina de Lectura, aclararemos el contenido de algunos lugares de memoria nominados de Z1 a Z7:

<u>Dirección de Memoria</u>	<u>Contenido</u>
Z1	vacío (lugar donde se preservará el contenido de A en la primera parte de la subrutina).
Z2	-1
Z3	vacío ( lugar donde se preservará el contenido de A en la segunda parte de la subrutina).
Z4	Dirección del primer dato.
Z5	-(Cantidad de datos -1)
Z6	vacío ( lugar donde se ubicará -(cantidad de datos -1) después de colocar la "traba".
Z7	vacío ( lugar donde se colocará la Dirección a ubicar el dato después de colocar la "traba").

Los valores de las direcciones Z1 a Z7 son fijos y dependen de la subrutina con la que se trabaje; el contenido de Z4 y Z5 será cambiado por programación antes de arrancar el equipo que será atendido por esa subrutina, colocando los valores apropiados al número de datos que se desea leer y a la dirección donde será almacenado el primero.

La subrutina está dividida en dos partes. La primera, manejada por la UC, verifica si la subrutina está en uso; de no ser así, pasa a usarse y se coloca la "traba" que impide un uso posterior por cualquier equipo hasta que no se "destrabe"; luego coloca los datos correctos en las direcciones Z6 y Z7 y finalmente arranca el equipo.

La necesidad de repetir Z4 y Z5 (en Z7 y Z6, respectivamente)

es evidente si se considera que los datos se colocan en Z4 y Z5, antes de saber si la subrutina estaba ocupada. De no existir esa providencia podría darse el caso de que el equipo estuviera en marcha, interrumpiera después de haber cambiado los contenidos de Z4 y Z5, como consecuencia de lo cual los datos de la primera operación se guardarían en las direcciones reservadas para los datos de la segunda.

La primera parte de la subrutina es la siguiente:

<u>Dirección</u>	<u>Instrucción</u>	<u>Comentario</u>
50	—	En esta direc. se preserva el contenido del registro C.
51	$M(z_1) \leftarrow A$ ta	Preserva el contenido del Acumulador.
52	$A \leftarrow M(z_2)$ ca	Verifican el estado de la "traba", que está dado por el contenido de la direc. $z_2$ .
53	$A \geq 0, C \leftarrow 53$ sa	
54	$A \leftarrow 0$ la	Si la subrutina estaba libre se coloca la "traba" ( $M(z_2)=0$ ) para que no pueda ser utilizada por otro equipo.
55	$M(z_2) \leftarrow A$ ta	
56	$A \leftarrow M(z_5)$ ca	Colocan -(cantidad de datos -1) en $z_5$ para que no sean alterados durante la ejecución de la subrutina.
57	$M(z_6) \leftarrow A$ ta	
58	$A \leftarrow M(z_4)$ ta	Colocan la dirección del primer dato después de colocar la "traba" con el mismo fin anterior.
59	$M(z_7) \leftarrow A$ ta	
60	$A \leftarrow M(z_1)$ ca	Restablece el cont. del Acumulador.
61		ap Arrancar periférico.
62	$C \leftarrow M(50)$ si	Retorna al programa original por el contenido de la direc. 50.

Las direcciones pueden ser cualquiera; se tomaron esos valores sólo por dar un ejemplo.

La forma de operar es la siguiente: Cuando en el programa principal, se desea leer un dato, con el equipo manejado por la subrutina cuya primera instrucción es la 50, se incluirá en

el programa principal una instrucción de Salto a Subrutina con la direc. 50. Luego de ejecutar las 13 instrucciones de la primera parte de la subrutina, la UC continúa con la ejecución del programa principal, pero ya el equipo está en marcha.

La segunda parte de la subrutina se comienza a ejecutar cuando la UC atiende el pedido de interrupción presentado por el equipo. Al ser atendido éste presenta, por su dirección trampa, una instrucción de ss (con la dirección 63 en el caso del ejemplo).

Segunda parte de la subrutina:

<u>Direc.</u>	<u>Instruc.</u>	<u>Comentario</u>
63	—	En esta direc. se preserva el cont. del registro C en el momento que se atendió la int.
64	$M(z_3) \leftarrow A$	ta Preserva el contenido del A.
65	$A \leftarrow P_e$	te Coloca en el A. el cont. del reg. del periférico.
66	$M(M(z_7)) \leftarrow A$	ta Guarda el cont. del A. en la direcc. contenida en $M(z_7)$ .
67	$M(z_6) < 0, M(z_6) \leftarrow Mz_6 + 1$ $M(z_6) \geq 0, C \leftarrow C + 2$	sm Verifica si se han leído todos los datos; en caso afirmativo saltea una inst.; en caso contrario obtiene la cant. de datos que falta leer y <u>cont.</u> con la inst. siguiente.
68	$C \leftarrow 73$	si
69	$A \leftarrow M(z_2)$	ca
70	$A \leftarrow A - 1$	ru
71	$M(z_2) \leftarrow A$	ta
72	$A \leftarrow M(z_3)$	ca Restablece el cont. del A.
73	$C \leftarrow M(63)$	si Retorna al programa principal sin mas datos a leer.
74	$A \leftarrow M(z_7)$	ca
75	$A \leftarrow A + 1$	au
76	$M(z_7) \leftarrow A$	ta
77	$A \leftarrow M(z_3)$	ca Restablece el contenido del A.
78		ap
79	$C \leftarrow M(63)$	si Retorna al programa principal.

La subrutina tiene 30 instrucciones en total. Debido a los saltos, sin embargo, el número máximo de instrucciones que pueden obedecerse en una pasada es de 25 (al leer el último dato).

Debe tenerse bien en claro que cualquier equipo puede ser manejado por cualquier subrutina, pero debe prestarse especial atención a:

- 1) Colocar, por programación, en la dirección trampa del equipo elegido la instrucción ss con la dirección de la primera instrucción de la subrutina deseada.
- 2) En las instrucciones de ap de la subrutina elegida, (en el caso del ejemplo, las instrucciones 61 y 78), colocar por programación la instrucción que arranque el número de equipo deseado.
- 3) Idem para la instrucción te (número 65 del ejemplo).

#### 7-2) Subrutina de Salida:

La subrutina de salida del perforador es muy parecida a la anterior. La diferencia fundamental está en que en la subrutina de lectura, primero se arranca el equipo (o sea se lee), y luego se transfiere el contenido del Buffer al Acumulador, en ésta, el proceso es a la inversa, o sea primero se transfiere el contenido del Acumulador al registro del periférico, y luego se arranca el equipo.

Como en el caso anterior, debemos aclarar el contenido de algunos lugares de memoria que denominamos de  $z_1$  a  $z_7$  y que tienen la misma información que en el ejemplo de la subrutina de entrada.

- $z_1$  vacío (lugar donde se preservará el contenido del A. durante la ejecución de la primera parte de la subrutina)
- $z_2$  -1
- $z_3$  vacío (lugar donde se preservará el contenido del A. durante la ejecución de la segunda parte de la subrutina).
- $z_4$  dirección del primer dato.
- $z_5$  - (cantidad de datos -1)
- $z_6$  vacío (lugar donde se ubicará -(cantidad de datos-1)- después de la "traba")

$z_7$  vacío (lugar donde se colocará la dirección a ubicar el primer dato después de la traba)

Estando completa esa información y cuando en el programa principal se desea perforar algún dato, se comienza la ejecución de la primera parte de la subrutina de entrada, por medio de una instrucción de ss a la subrutina correspondiente.

Primera parte de la subrutina del perforador:

<u>Direc.</u>	<u>Inst.</u>	<u>Comentario</u>
100		En esta dirección se preserva el contenido del registro C al producirse el ss.
101	$M(z_1) \leftarrow A$	ta Preserva el contenido del acumulador.
102	$A \leftarrow M(z_2)$	ca } Verifican el estado de la "traba" que esta dado por el cont.de la direcc. $z_2$ . sa } Si $M(z_2)$ es $\geq 0$ , la subrutina está ocupada.
103	$A \geq 0, C \leftarrow 103$	
104	$A \leftarrow 0$	la } Si la subrutina estaba libre se colocan la "traba" ( $M(z_2) = 0$ ) para que no pueda ser utilizada por otro equipo
105	$M(z_2) \leftarrow A$	ta }
106	$A \leftarrow M(z_5)$	ca } Colocan -(cantidad de datos -1) en $z_6$ para evitar que sean alterados durante la ejecución de la subrutina.
107	$M(z_6) \leftarrow A$	ta }
108	$A \leftarrow M(z_4)$	ca } Colocan la dirección del primer dato después de colocar la "traba" con el mismo fin anterior.
109	$M(z_7) \leftarrow A$	ta }
110	$A \leftarrow M(M(z_7))$	ca Carga el A. con el contenido de la dirección de $M(z_7)$ (Información a ser perforada).
111	$Pe \leftarrow A$	ts Transfiere el contenido del A. al Buffer del periférico elegido.
112	$A \leftarrow M(z_1)$	ca Restablece el contenido del A.
113		ap Arranca el equipo elegido.
114	$C \leftarrow M(100)$	si Retorna al programa principal por el contenido de la dirección 100.

La segunda parte de la subrutina se inicia cuando la UC atiende el pedido de interrupción presentado por un equipo que en su dirección trampa tiene colocada una instrucción de ss con la dirección de la primera instrucción de la segunda parte de la subrutina.





R E F E R E N C I A S :

1) Digital Equipment Corporation.

" Small Computer Handbook 1972 "

2) Hewlett Packard.

" A Pocket Guide to Hewlett Packard Computers "

3) General Automati6n

" The Value of Power "

4) Hewlett Packard

" An introduction to H P Computers "

5) Antonio D. Osterman

" Unidad de Salida de una Computadora Electr6nica "

Trabajo de Proyecto Final, UNS, 1969.

6) Ferranti

" High Speed Tape Reader Type TRS "

7) Hewlett Packard

" Driver Manual "