UNIVERSIDAD LA SALLE

ESCUELA DE INGENIERIA

INCORPORADA A LA U.N.A.M.

ANALISIS DE DISPOSITIVOS LOGICOS PROGRAMABLES PAL'S, Y SU PROGRAMACION

TESIS PROFESIONAL

QUE PARA OBTENER EL TITULO DE INGENIERO MECANICO-ELECTRICISTA

PRESENTA:

MYLENNA DEL CARMEN LOPEZ CASTRO

DIRECTOR DE TESIS:

ING. PATRICIA VASQUEZ AGUILERA

MEXICO,D.F. 1990

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE

INTRODUCCION

EVOLUCION HISTORICA DE LA ELECTRONICA

INTRODUCCION

EVOLUCION HISTORICA

1879, EL BULBO

Desde su invención en 1879, el bulbo ha representado la esencia de la invención en sí. Pero para la industria electrónica, el bulbo es más que un simple símbolo, es el principio de la evolución de la microelectrónica de hoy.

Tomas Edison, mientras desarrollaba las bases para apoyar el filamento en su bulbo, observó que una superficie de metál insertado en lo alto del bulbo producía un corto cuando se apoyaba. En otras palabras, el filamento caliente estaba haciendo algo más que sólo creando la luz. Arrojaba electrones.

Estas pequeñas cargas de energía viajaban hacia el tubo de vacío y se juntaban en la superficie de metál. Experimentos posteriores mostraron que aplicando un voltaje positivo a la superficie de metál, un mayor número de electrones eran atraídos a ella. En otras palabras, drenaría una mayor corriente.

1906 EL TUBO VACIO

En ese año, el Dr. Lee DeForest introdujo un nuevo elemento, el cual él llamó rejilla, entre el filamento y la superficie. Después, aplicó un pequeño voltaje a la rejilla y encontró que variando el voltaje podría crear variaciones en esta gran corriente enviada del filamento a la superficie. También encontró que las variaciones en esta gran corriente era directamente proporcional a las variaciones del pequeño voltaje en la rejilla.

Por lo tanto, el tubo de vacío y el principio de la amplificación nacieron.

A lo largo de la primera mitad del siglo XX, los tubos de vacío eran utilizados para conducir, modular y amplificar señales eléctricas. Estos dispositivos hicieron posible una variedad de nuevos productos, incluyendo la radio y la computadora. Pero los tubos de vacío tenían problemas inherentes; eran voluminosos, delicados y costosos; consumían una gran cantidad de energía, tomaban mucho tiempo para calentarse, se sobrecalentaban y eventualmente se quemaban.

1947 TRANSISTORES

Por los años de 1930, los investigadores de los laboratorios Bell Telephone buscaban un reemplazo del tubo de vacío. Empezaron estudiando las propiedades eléctricas de los semiconductores, substancias no metálicas, como silicón, que no son ni conductores de electricidad, como los metales, ni aislantes, como la madera, pero sus propiedades eléctricas residen en medio de estos límites.

Lo equipos de investigación de los Laboratorios Bell encontraron una forma directa para alterar directamente las propiedades eléctricas del material semiconductor. Se dieron cuenta de que podían cambiar y controlar estas propiedades contaminando el semiconductor, o llenándolo con elementos selectos, calentados a una fase gaseosa.

Cuando el semiconductor también, estaba caliente, los átomos de los gases se introducirían a él y modificarían su pureza, su estructura cristalina desplazando algunos átomos. Debido a esta contaminación los átomos tenían diferentes cantidades de electrones en comparación con lo átomos del semiconductor, formaban caminos conductores.

Si los átomos contaminados tenían más electrones que los átomos del semiconductor, las regiones contaminadas eran llamadas del tipo -n para simbolizar el exceso de carga negativa. Menos electrones, o un exceso de carga positiva creaba las regiónes -p. Permitiendo que esta contaminación se establezca en áreas cuidadosamente delineadas en la superficie del semiconductor, las regiones tipo -p podían ser creadas con regiones tipo -n, y viceversa.

Utilizando esta técnica, llamada difusión, tres científicos de los Laboratorios Bell, Schockley, Brattain y Bardeen, pudieron crear un transistor completamente, y con él la electrónica del estado sólido.

1958 CIRCUITOS INTEGRADOS

El transitor era mucho más pequeño que el tubo de vacío, consumía muy poca energía, no requería de que se calentase, no se sobrecalentaba mucho y no requería de un filamento calentado que eventualmente se quemaba.

A mediados de los 1950's, los primeros transistores comerciales fueron vendidos. Pero la investigación continuaba. Si un transistor podía ser construido en una pieza de material semiconductor, por qué no varios transistores o incluso un circuito completo?

Con el paso de los años esta especulación se volvió una realidad. Dispositivos conteniendo varios componentes de un circuito eran construidos en una sólida pieza de material. Estos circuitos integrados (ICs) redujeron el número de interconexiones eléctricas requeridas en una pieza de equipo eléctrico, por tanto incrementaron la confiabilidad y la velocidad.

Es aquí donde los dispositivos lógicos programabales (PLDs - Programmable Logic Device) empezaron a surgir, incluyendo en sus arrreglos internos circuitos equivalentes conteniendo la mayoría de los componentes de un diseño con tecnología previa. Los PALs (Programmable Array Logic) son dispositivos que pertenecen a la familia de los PLDs y que logran sustituir de 3 a 8 circuitos de tecnología "TTL" en sus primeros diseños.

Al paso de los años la tecnología de los circuitos integrados ha avanzado rapidamente, con marcadas reducciones de tamaño y enormes incrementos en la capacidad. Hoy en día, la mayor industria tecnologica en el mundo, los fabricantes deben tratar con dimensiones de superficie para los circuitos integrados de micrómetros, o una millonésima parte de un metro.

En contraste con la primera computadora electrónica digital (construida con 18,000 tubos de vacío) pesaba 50 toneladas, costo alrededor de 1 millón de dólares, requería 140 kilowatts de energía y ocupaba un cuarto completo. Hoy, una computadora completa, fabricada en una sola pieza de silicón del tamaño de una uña de un niño, cuesta solamente 10 dólares.

Pero los beneficios de los circuitos integrados no son exclusivos de las computadoras. En los 20 años pasados, los circuitos integrados han prevalecido en todo tipo de máquina moderna, mejorando viejos productos y creando una gran variedad de nuevos de ellos. De las computadoras, a los robots, de aplicaciones "inteligentes" caseras a cajas registradoras "parlantes", de tableros luminosos en carros a grandes aeronaves "voladoras", las máquinas continúan mejorandose a través de las innovaciones de la tecnología de los circuitos integrados.

CAPITULO I

ANALISIS DEL "PAL"

CAPITULO I

ANALISIS DEL "PAL"

PRIMERA PARTE

1.1 DEFINICION DE "PAL"

Un "PAL" es un arreglo lógico programable de compuertas que permite implementar en un sólo circuito integrado el número equivalente de paquetes de pequeña y mediana escala de integración. La estructura interna de estos dispositivos es una malla programable de fusibles que interconectan combinaciones de compuertas "AND" y "OR", registros, "flip-flops", y memoria. La configuración lógica final de la malla es determinada por el usuario. Estos dispositivos permiten al usuario diseñar circuitos, tanto lógicos como secuenciales.

El costo total del proceso de un circuito implementado con "PAL's" es frecuentemente menor que aquel con componentes normalizados. Sólo del 25 al 50 por ciento del costo al utilizar circuitos de pequeña o mediana escala de integración es normalmente su precio de compra; el 50 al 75 por ciento restante esta ligado a su costo en el área de la tarjeta, ensamble y pruebas asociadas a ese circuito integrado. Por ello, como el "PAL" reemplaza cada día más circuitos integrados, su uso rapidamente se justifica tan sólo basandose en el costo.

1.1.1 VENTAJAS

Utilizar un dispositivo "PAL" en un diseño origina muchas ventajas, tanto para el diseñador como para el usuario final. A continuación se encuentran algunas de las razones que justifican la utilizacion de los "PAL" en diseños en los que anteriormente se utilizaban una gran cantidad de compuertas lógicas.

EFECTIVIDAD DE COSTO

Todas las justificaciones parten de un fin específico y convergen a un fin económico. Es por ello que estos factores se clasificaron dependiendo de la forma en que logran la reducción en el costo:

- Menor costo en componentes a través de:
  Reducción en el área de tarjetas.
  Reducción en los conectores utilizados.
  Fuentes de energía más pequeñas.

- Menor costo en diseño y desarrollo a través de:
  Facilidad al hacer cambios.
  Compatiblidad en el "software" de diseño.
  "Lay-out simplificado".
  Documentación computarizada.

- Menor costo en manufactura a través de:
  Menor inserción de componentes.
  Menor número de tarjetas para manufacturar.
  Menor número de componentes, tarjetas y sistemas que serán necesario probar.

- Menor costo de servicio a través de:
  Incremento en la confiabilidad.
  Menor número de partes de reposición.
  Compostura de fallas lógicas más rápidas.

Todas estas justificaciones merecen su explicación, es por ello que a continuación se describirán brevemente cada una de ellas.

REDUCCION EN EL ESPACIO DE LA TARJETA

Los "PAL's", generalmente, implementan el equivalente de 4 a 12 paquetes de pequeña y mediana escala de integración en un sólo circuito integrado de 20 terminales. Si el total de espacio en la tarjeta es insuficiente para las necesidades del diseñador, se debe considerar seriamente la posibilidad de utilizar "PAL's" en el diseño.

PEQUEÑO INVENTARIO

La familia de "PAL's" puede ser utilizada para reemplazar un 90% de los componentes "TTL" con sólo 15 diferentes partes. Esto reduce considerablemente los costos de inventarios.

RAPIDO SISTEMA DE DISEÑO

Debido a la facilidad de programación y a la flexibilidad que ofrecen los "PAL's", el tiempo necesario para diseñar e implementar un sistema se puede reducir a la mitad. Los prototipos se pueden construir en "proto-boards" rapidamente para probar nuevas ideas sin necesidad de esperar mucho.

FLEXIBILIDAD EN EL DISEÑO

Los "PAL's" ofrecen al ingeniero de diseño mayor flexibilidad que la normal, sobretodo cuando se trata de partes "off-the-shelf". Si la función deseada no se encuentra disponible con componentes estándar, se requerirá una gran cantidad de componentes para implementar la función. Con los "PAL's", el ingeniero simplemente elige lo que desea en lugar de lo que puede obtener.

FACILIDAD DE CAMBIOS EN EL DISEÑO

Los "PAL's" ofrecen al diseñador la habilidad de reprogramar un circuito integrado en lugar de rediseñar otro "hardware" e implementar otra tarjeta cuando una función del diseño es modificado.

ALTA VELOCIDAD

Los "PAL's" son construidos utilizando la tecnología Schottky; un reloj de 40Mhz, con partes más rápidas en el transcurso.

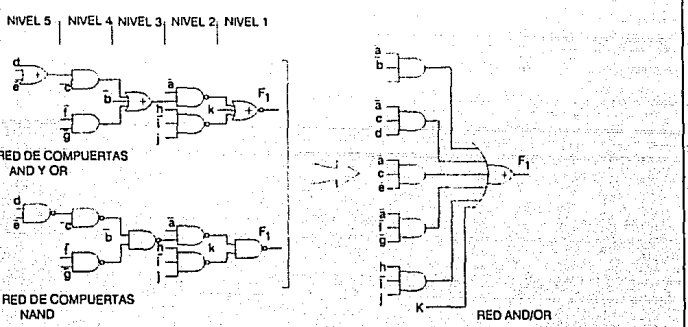
FACILIDAD DE PROGRAMACION EN EL CAMPO

A diferencia de los arreglos de compuertas y otros avances del diseño con componentes lógicos, el "PAL" puede ser programado por el usuario, lo que minimiza el tiempo por completo. El "PAL" puede ser programado rápida y fácilmente utilizando un programador de "PROM's" (por sus siglas en inglés "Programmable Read-Only Memory") convencional con una tarjeta especial para "PAL's". La conversión de las funciones lógicas al formato del "PAL" es fácil y rápido utilizando el "software" correspondiente.

REDUCCION EN LA LOGICA DE MULTINIVEL

El diseñador puede comprimir múltiples niveles de lógica en una estructura de dos niveles con compuertas "AND" y "OR", a través del uso de la lógica programable. Con ello se simplifica el diseño, y en muchos casos se obtiene una ventaja en velocidad y/o consumo de energía. Esto se muestra en la figura 1.

Es importante explicar el impacto de los "PAL" en el diseño lógico ya que los diseñadores de lógica están notando un aparente "nivel de complejidad" entre la tecnología de los "TTL" y la de larga escala de integración. Los productos diseñados utilizando dispositivos "TTL" consumirían cantidades inaceptables de energía eléctrica y de espacio. Los dispositivos de gran escala de integración que son programables mediante "software" (microprocesadores) ofrecen alta densidad y requieren de, relativamente, poca energía para realizar casi todo lo imaginable, pero el diseñador paga un precio muy elevado en el desarrollo del "software" y todavía tiene que utilizar componentes discretos para interconectarlo al exterior. Hasta la fecha, no ha habido un dispositivo que provea una forma efectiva de enlazar estos dos niveles. Es por ello,



Reduccion de la Logica de Multinivel

que surgieron los dispositivos "PAL's ("Programmable Array Logic") para satisfacer este requerimiento. Los "PAL's" ofrecen grandes capacidades para crear nuevos productos los que resultan efectivos y con un costo razonable, o para dar efectividad a la existencia de los diseños basados en componentes lógicos. Los dispositivos "PAL"s ahorran tiempo y dinero resolviendo muchos de los sistemas, dividiendo e interconectando los problemas no resueltos; sino hasta ahora con la efectividad de la tecnología de los dispositivos semiconductores actuales.

Ecuacion Logica $F = a[b+c(d+e)+fg] + hij + k$

figura 1

1.1.2 FAMILIAS

A continuación expondremos brevemente las caracterísiticas de los dispositivos que integran la clasificación de larga escala de integración, los dispositivos "TTL" y los circuitos integrados tradicionales.

La tecnología "LSI" ("LARGE SCALE INTEGRATION") ofrece muchas ventajas, pero los avances se han realizado a expensas, ya sea, de la flexibilidad del dispositivo o de la complejidad del "software". La tecnología "LSI" ha tendido y continuará tendiendo cada vez más, a mayores funciones lógicas convencionales. La tecnología de larga escala de integración ofrece alta densidad funcional y bajos consumos de energía; ahora, un sólo circuito integrado realiza las funciones que antes hubieran requerido de una tarjeta completa. De cualquier manera, muchos dispositivos de larga escala de integración no tienen interfaz con sistemas del usuario sin un gran número de dispositivos de apoyo. Los dispositivos aún estan forzados a girar alrededor de la lógica aleatoria para muchas aplicaciones. LSI es lenta, y está rígidamente dividida. Tomando en cuenta toda su capacidad, para realizar tareas variadas y complejas, el microprocesador representa un método lento y costoso de realizar las operaciones simples y repetitivas; si se toman en cuenta los dispositivos de apoyo que se añaden, como lo son: el tiempo, el dinero y la memoria requerida para desarrollo de "software".

La tecnología "TTL" ("Transistor - Transistor Logic") provee velocidad, y se puede decir que su flexibilidad es infinita. Esto es a cambio de alto consumo de energía, gran número de partes y el espacio utilizado.

Los circuitos integrados tradicionales pueden ser una solución efectiva para el diseño si la complejidad del producto es baja o mediana; su función lógica está bien definida, y tiene un volúmen muy alto en el mercado. Su ciclo de diseño es, tipicamente largo, y su costo puede hacerlo innaccesible. Esto hace que se tienda a la no utilización de estos dispositivos.
Para tratar de superar las desventajas antes mencionadas aparecen los dispositivos lógicos programables. Estos se encuentran en diversas formas. Todos, a excepción del "PAL", requieren de interfaz lógica externa y tienen desventajas que se deben superar y que se presentan a continuacion:

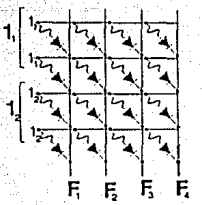
"PROM": Requiere de un diseño cuidadoso para evitar cambios indeseables de información. También,se encuentra limitado el número de variables de entrada que pueden ser acomodadas.

"FPLA": Es muy costoso, difícil de programar y de entender.

"FPGA": Este tipo de dispositivo no se obtiene fácilmente y tiene poca flexibilidad.

"PMUX" : Se encuentra disponible sólo en algunos tipos.

"PAL" - Extensión de la Tecnología de Fusible - Enlace



MATRIZ DE DIODOS OR

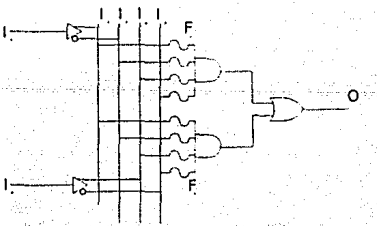
La matriz de diodos fue el primer dispositivo lógico programable en un circuito integrado; fué introducido alrededor de los años 60's. Este dispositivo contenía una sóla matriz implementada con lógica de diodo "OR"; cada cruce que contenía un fusible era una unión. Como se puede observar en la figura 2.

La memoria programable de sólo-lectura "PROM" extendió el concepto de lógica programable considerablemente permitiendo que las variables de entrada fueran codificadas, reduciendo el número de terminales necesarios para cada variable de entrada y proveyendo compatibilidad con la tecnología "TTL". Los "PROM's" son elementos lógicos de compuertas "AND-OR" con una matriz fija "AND" y una matriz "OR" programable.

Figura 2

Una ventaja de utilizar "PROM's" es que son producidas en altos volúmenes ya que son utilizadas en muchas aplicaciones. También, las "PROM's" son una solución universal; en otras palabras, todos los productos de las variables de entrada son generadas, haciendo posible implementar cualquier función "AND-OR" de estas variables.

Los arreglos lógicos programables de campo "FPLA" tienen un segundo fusible (una matriz "AND"), que permite al diseñador seleccionar y programar sólo esos productos utilizados en



NOTACION INTERNA DEL PAL

Figura 3

cada función específica. Estos productos, son entonces, combinadas en el arreglo de fusibles "OR" para formar una ecuación lógica "AND-OR".

Un arreglo lógico programable "PAL" es un dispositivo compuesto por una matriz programable "AND" y una matriz fija "OR", cuya estructura interna final es determinada por el usuario. La estructura interna de estos dispositivos es una malla programable de fusibles que interconectan compuertas "AND", "OR" y registros. Estos dispositivos permiten al usuario diseñar circuitos tanto lógicos como secuenciales.

1.1.3 COMO FUNCIONAN LOS "PAL's"

En el concepto de "PAL", un arreglo de fusibles "AND" permite al diseñador especificar los productos necesarios, y conectarlos a una matriz "OR" escogida para realizar la combinación requerida de funciones lógicas "AND-OR". Los "PAL's" se encuentran disponibles en diferentes tipos lo que depende de la variación de la configuración de compuertas "OR". Especificar la conexión de la compuerta "OR" implica una tarea de selección del dispositivo más que de programación, como es el caso de los dispositivos "FPLA". Con este avance, los "PAL's" eliminan la necesidad de un segundo fusible con una pequeña pérdida en su flexibilidad total. La figura: 1 representa la Notación Interna del "PAL" y muestra como éste circuito procesa un segmento lógico de dos entradas y una salida.

La ecuación general para este segmento es:

Salida O = (I1 . + /f1)(I1 . f2 + /f2)(I2 . f3 + /f3)
(/I2 . f4 + /f4) + (I1 . f5 + /f5)
(/I1 . f6 + /f6)(I2 . f7 + /f7 + /f7)
(I2 . f8 + /f8)

donde los términos "f" representan los estados de la unión del fusible en los arreglos AND de los PAL's. En la ecuación anterior, un fusible intacto es representado por f = 1, y un fusible abierto por f = 0.

A pesar de que las ecuaciones lógicas son convenientes para pequeñas funciones simples, se convierten, progresivamente, menos tratables conforme la función va siendo más compleja.

NOTACION CONVENCIONAL

NOTACION LOGICA DEL PAL

En grandes sistemas, el diagrama lógico o esquemático, y la tabla de verdad son los métodos

más comunmente utilizados para describir las redes lógicas. Para simplificar, la lógica de los "PAL's" se describirá utilizando la notación simbólica que se muestra en la figura 4. El diagrama lógico combinatorio convencional se muestra en la misma figura del lado izquierdo. En la figura, una X representa un fusible intacto que, en conjunto con las series de diodos (no se muestran en el esquema), realiza la función lógica "AND".

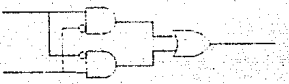


Figura 4

El ejemplo de dos entradas y una salida mostrada en la figura 3, se vuelve a presentar en la figura 5 utilizando, en ella, la nueva notación lógica.
En la figura 6 se muestra el diagrama lógico normal combinacional de un ejemplo cuya función de transferencia es la que se encuentra a continuación:

Salida O = I1 . /I2 + /I1 . I2

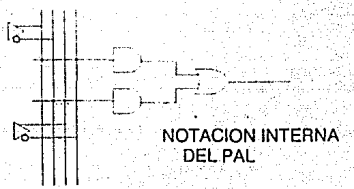


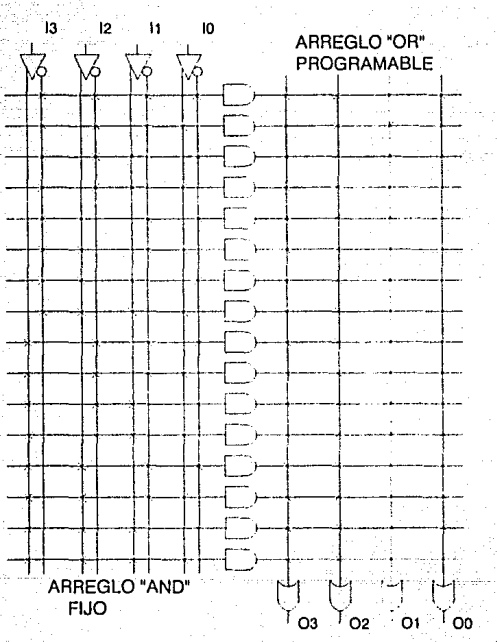
Figura 5

Como se puede observar de la expresión anterior y del diagrama correspondiente, este es un circuito de dos entradas y una salida, por lo tanto, el circuito de la figura 5 podría ser una buena solución para este problema. Quitando las X's de las uniones que no representan términos del ejemplo de la expresión Booleana dará por resultado el diagrama que se muestra en la figura 7, que es el mismo circuito mostrado en la figura 5, excepto que los fusibles apropiados han sido abiertos.

Figura 6

Utilizando esta simbología no sólo se despliegan los atributos del diagrama lógico, sino también los de las tablas de verdad para los mismos ejemplos mostrados. Con esta técnica, es posible comparar la estructura de un "PAL" con los de los "PROM's" y los "FPLA's".

1.1.4 COMPARACION



PROM

16 PALABRAS

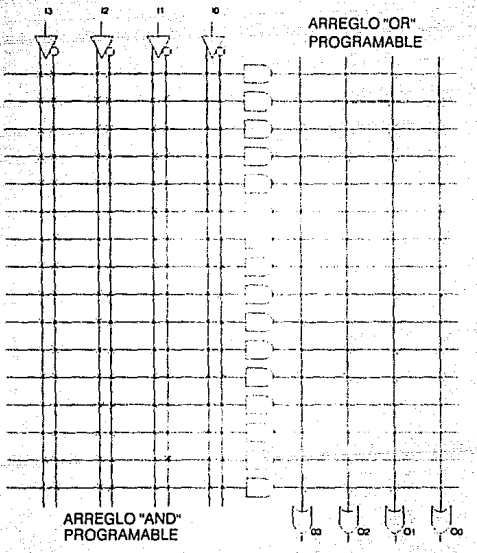
DE 4 BITS

Para ilustrar las diferencias entre estos tres conceptos de lógica programable, cada uno de los logros son mostrados como una matriz "AND", seguida de una matriz "OR". La lógica básica

implementada por una "PROM" es "AND-OR", con las compuertas "AND" todas preconectadas en el circuito integrado, haciéndo esta parte fija, mientras que la matriz "OR" es implementada con interconexiones diodo-fusible, haciéndolas programables. Por ello, una "PROM" es un elemento lógico "AND-OR" con una matriz "AND" fija y una matriz programable "OR". La solución de la "PROM" mostrada en la figura 8 requiere de una matriz de fusibles de 64 uniones.

. Figura 8

Existen muchas ventajas en utilizar "PROM's" como dispositivos lógicos. Una de ellas es que, ya que son utilizadas en muchas aplicaciones, son producidas en altos volúmenes. Además, las "PROM's" son una solución universal, es decir, todos los productos de las variables de entrada son generados, haciéndo posible implementar cualquier función "AND-OR" de estas variables. De cualquier manera, las "PROM's" no pueden acomodar grandes números de variables; el máximo número de variables de entrada que actualmente se están desarrollando es de 11.

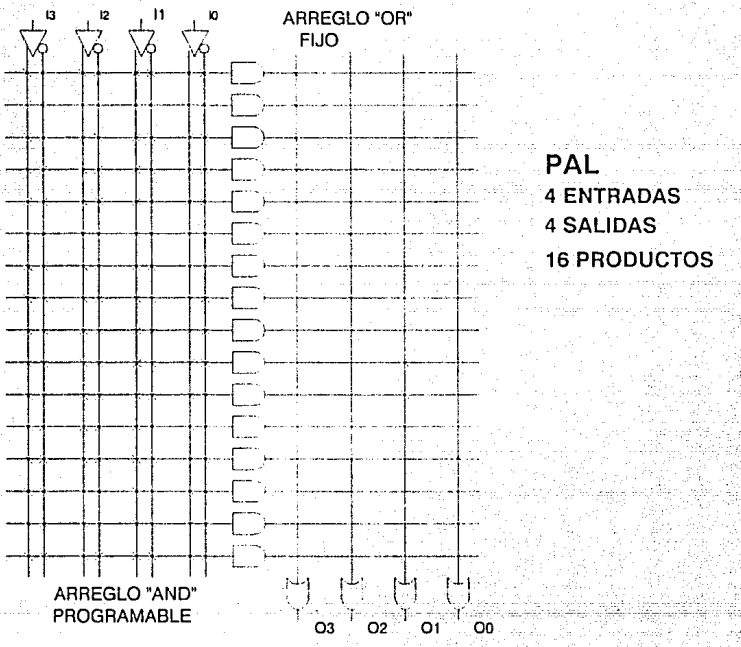


FPLA

4 ENTRADAS
4 SALIDAS
16 PRODUCTOS

Como ya se mencionó anteriormente los "FPLA" tienen un segundo fusible (una matriz "AND") permitiendo al diseñador seleccionar y programar sólo esos productos utilizados en

cada función específica. Estos productos son entonces combinados en el arreglo de fusible "OR" para formar la ecuación lógica "AND-OR". La implementación típica para un "FPLA" tiene menos de 2n términos disponibles (donde n es el número de variables de entrada). Esto permite al "FPLA" integrar grandes valores de variables de entrada (n). En contraste con las "PROM's", donde el número de productos es siempre igual a 2n. A pesar de que los "FPLA" usualmente requieren menos fusibles para implementar una función lógica dada, se requiere



circuitería adicional para seleccionar y programar estos fusibles (circuitería que no es utilizada en la solución final, pero por la que se paga en el área del circuito). Este costo fijo del circuito integrado se vuelve significativo en aquellas aplicaciones donde no se aprovecha al máximo la lógica disponible.

Figura 9

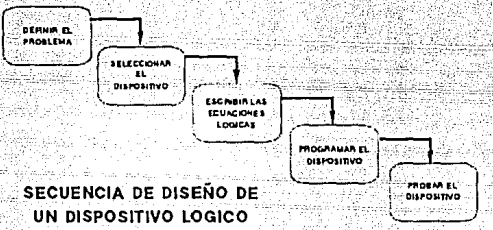
La estructura lógica básica de un "PAL", consistiendo de un arreglo programable "AND" cuyas salidas alimentan a un arreglo "OR" fijo, se muestra en la figura 10. El "PAL" es de menor costo y de más fácil programación, como la "PROM", pero además es más flexible, como el "FPLA".

Figura 10

SEGUNDA PARTE

1.2 SECUENCIA PARA DISEÑAR CON LOGICA PROGRAMABLE

Cada diseño difiere de otros debido a las necesidades de cada uno de ellos en especifico, pero



SECUENCIA DE DISEÑO DE UN DISPOSITIVO LOGICO PROGRAMABLE

el procedimiento es similar para todos ellos:

A continuación se describirán brevemente cada uno de estos pasos:

1.2.1 DEFINICION DEL PROBLEMA

Primero es necesario saber la función que desarrolla el circuito lógico del que se trata. Si se utiliza para generar señales de control combinatorio, decodificar direcciones, códigos de operación, generar diferentes secuencias de control o para implementar un estado de máquina para uso cualquiera.

Una vez identificada la función que se desea desarrollar es posible decidir el tipo de circuito lógico que se puede utilizar.

1.2.2 SELECCION DEL DISPOSITIVO

El siguiente punto a definir es, que PAL debe ser utilizado para optimizar espacio y por lo tanto el costo.

Por ejemplo: si se tienen 10 señales de entrada y 7 señales de salida y la mayoría de las salidas son bajas - activas, entonces la mejor opción es el 10L8. Si el número de salidas son seis entonces podemos utilizar, ya sea, el 10L8 o el 12L6.

Ya que cada PAL tiene limitaciones de "product terms" es necesario saber cuantas "product terms" utiliza cada salida. Esto se define a partir de ecuaciones lógicas. Por ejemplo si la ecuación lógica es O1 0 = P1 + P2 + P3 + P4 + P5 la salida O1 estará utilizando cinco "product terms".

1.2.3 ESCRIBIR LAS ECUACIONES LOGICAS

El procedimiento que se debe seguir es sencillo y se describe a continuación:

1.- Definir las entradas y salidas.

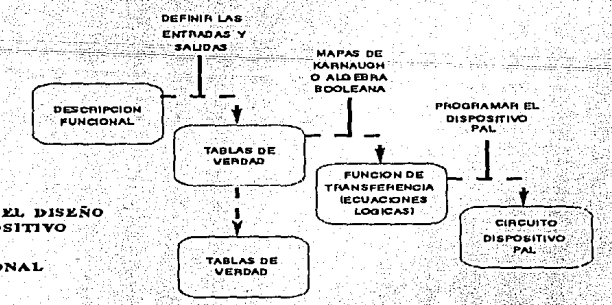
2.- Generar la tabla de verdad.

3.- Obtener la expresión SOP para cada salida.

4.- Utilizar las tecnicas de minimización (Algebra Booleana, Mapa K o Quine - Mc Cluskey) para minimizar las expresiones de SOP.

5.- Siguiendo los puntos anteriores se obtendrán las ecuaciones lògicas.

En la siguiente figura podemos observar estos pasos:



Es mucho más complicado generar ecuaciones lógicas para un circuito secuencial que para un circuito combinatorio.

Generalmente, los procedimientos son como se describe a continuacui

A. Definir las entradas y salidas, diferentes estados y variables.

B. Generar el diagrama de estado.

C. Minimizar la tabla de estado.

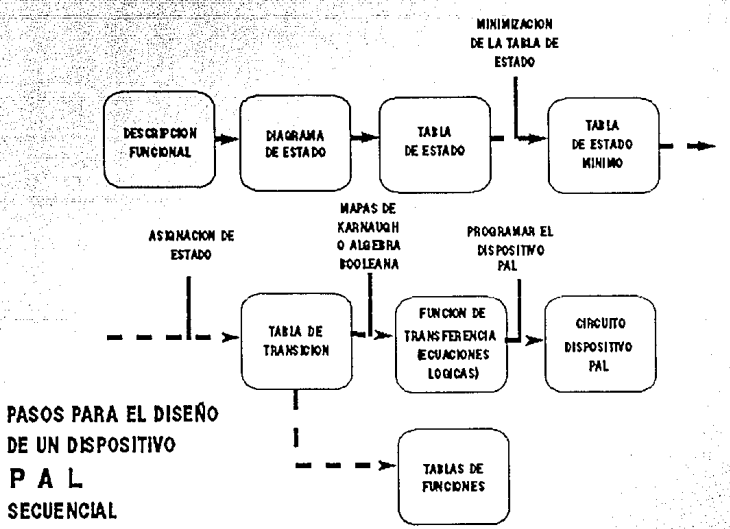
D. Asignar el nuevo estado.

E. Generar la tabla de transición.

F. Minimizar la tabla de transición.

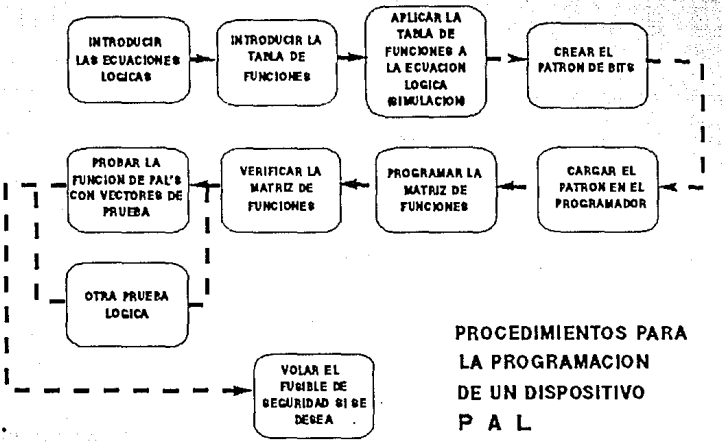
G. De estos seis pasos resultan las ecuaciones lógicas.

La siguiente figura muestra estos pasos:



1.2.4 PROGRAMANDO EL DISPOSITIVO

Dadas las ecuaciones lógicas, el programador del dispositivo PAL realizará la programación. Todo lo que se requiere es introducir las ecuaciones lógicas a la terminal. Los procedimientos para programar se muestran en la tabla que se encuentra a continuación:



CAPITULO II

PROGRAMADORES

CAPITULO II

PROGRAMADORES DE "PAL'S"

2.1 GENERALIDADES

En general un programador de "PAL" "(Programmable Array Logic)" consiste de un tarjeta que puede ser insertada en cualquier micro-computadora, eliminando así, la necesidad de una fuente de poder y una caja. También consiste de una tarjeta :socket: en el cual puede ser insertados los circuitos integrados de 20 y 24 terminales. Esta última tarjeta permite programar con la microcomputadora cerrada ya que se conecta a la tarjeta anteriormente insertada dentro de la micro. Se utiliza un conector comercial norma DB-25. Cada programador cuenta con un programa cuyo trabajo consiste en leer el archivo :JEDEC: e interpretar el mapa de fusibles para manejar la tarjeta y programar el "PAL". Sólo se requerirá proveer el diseño lógico a un compilador que creé el archivo :JEDEC:.

2.1.1FILOSOFIA DE DISEÑO

Enseguida se enuncian algunos puntos que se consideran al diseñar un programador.

* Autocalibración (sin preocuparse por la variacion, sin salirse de tolerancia).
* No se requieren resistencias de precisión
* No se utilizan potenciometros
* Desarrollar el programa en un sólo lenguaje de alto nivel como lo es el lenguaje C.
* No depender de lazos de tiempo en el programa.
* Hasta donde sea posible, no utilizar componentes muy caros.
* Arquitecturas abiertas, que puedan ser expandidas a nuevos dispositivos.
* Bajo costo.

2.1.2 PRINCIPIOS DE PROGRAMACION DE "PALS".

Para comprender el funcionamiento de un programador de "PAL" es necesario entender como está organizado internamente.
Los ejes del PAL están numerados. Las líneas de entrada están numeradas horizontalmente arriba, y los maxi-términos están numerados hacia abajo a la izquierda.
Las siguientes variables serán de utilidad en la comprensión del funcionamiento de un programador de "PAL's".

L = "1" = Estado lógico bajo = VIL
H = "0" Estado lógico alto
Z = Alta impedancia
HH = Super voltaje

El número de fusibles es operado como el maxi-término por 32 más la línea de entrada, es decir:

$$\#\ \text{fusible} = \text{maxi-término} * 32 + \text{línea de entrada}$$

Analizando el número de fusible es posible operar todas las direcciones necesarias para programar cada fusible. Las líneas de entrada están organizadas en grupos,de cuatro, o sea, 0-3, 4-7, 8-11.... 28-31.
Los dos números de línea más baja de cada grupo se conectan a las entradas no inversora e inversora que provienen del lado izquierdo del diagrama del "PAL"; mientras que las líneas de entrada numeradas con mayor rango en cada grupo se conectan ala líneas de entrada no inversora e inversora que provienen del lado derecho del diagrama.
Si el número de fusible es menor que la mitad del total de fusibles del arreglo, el fusible se encuentra en la primera mitad del "PAL" y viceversa.
Se selecciona el grupo de líneas de entrada colocando un valor lógico de z para una terminal Ix. Un valor lógico de z en la terminal I/R selecciona las entradas de baja numeración del grupo, mientras que una valor de HH en la misma terminal selecciona las entradas de alta numeración. La polaridad de la señal de entrada es determinada por la variable línea de entrada. Si línea de entrada es impar la entrada es no inversora, y si la línea de entrada es par, la entrada es inversora. Los maxi-términos son agrupados en grupos de ocho a cada salida.
Para encontrar un fusible se utiliza la siguiente ecuación

$$Ox = \frac{\#\ \text{fusible}}{32 * 8}$$

Para encontrar la dirección (A0, A1 ó A2) del maxi-término de esa salida, se utiliza:

Dirección = (#fusible)8% ó módulo ocho.

Cada "bit" 0 del tercer "bit" de la dirección se ajusta a Z y cada "bit" 1 de la dirección se ajusta a HH.

2.2 PROGRAMADOR

El programador de los "PAL"s, contiene un programa mediante el cual se extraen las características que se deseen tener en un dispositivo. Dichas características se introducen al programa mediante un menú principal, en el que se incluyen funciones como se muestra a continuación.

1. FABRICANTE DEL DISPOSITIVO
Se utiliza para indicar la marca del dispositivo que se va a utilizar.

2. TIPO DE DISPOSITIVO
Conociendo el código del dispositivo, se le indica al programa mediante esta opción.

3. CARGAR EL MAPA DE FUSIBLES
Con esta opción se carga el mapa JEDEC al programa.

4. GUARDAR EL MAPA DE FUSIBLES
Con esta opción se crea un archivo con la información del mapa de fusibles, pero no con formato JEDEC, y este archivo sólo se puede usar en este programador.

5. EDITAR EL MAPA DE FUSIBLES
El mapa cargado en el "buffer" del programador, puede ser desplegado y editado por esta función, para esto se utlizan las teclas de movimiento en ventana de cualquier editor de texto.

Si se tiene un dispositivo nuevo la sintaxis será:
N = No hay fusible
X = Hay fusible

Si se tiene un dispositivo ya programado la sintaxis que se encontrará será:
0 = Fusible intacto
1 = Fusible quemado
N = No importa porque no hay fusible

6. PROGRAMAR
Da la oportunidad de verificar que el dispositivo esté en blanco antes de programar.
Programa el dispositivo.
Realiza una auto-verificación del programa de fusibles después de realizar la programación. Si no encuentra errores lo indica en pantalla.

7. LEER

Lee el mapa de fusibles del "PAL" y lo carga en el "buffer" del programador.

8. COPIAR

Esta función es similar a la función de programar, sin embargo ésta es más rápida ya que no realiza verificación del dispositivo.

9. VERIFICAR

Después de haber copiado varios "PAL"s estos pueden ser verificados con esta función.

10. REVISION DE ESTADO EN BLANCO

Esta función revisa que los dispositivos que van a ser programados estén en blanco. Al realizar ésta función, ésta no altera las características originales del "PAL".

11. QUEMA DE LOS FUSIBLES DE SEGURIDAD

Al operar ésta función se queman los fusibles de seguridad previniendo así que la programación pueda ser leída o alterada.

12. LECTURA DEL ESTADO DE LOS FUSIBLES

Es similar a la función de lectura, la diferencia está en que ésta lee directamente con el voltaje de alimentación del "PAL", según se muestra a continuación.

0 = 0 Volts
1 = 5 Volts

CAPITULO III

INTRODUCCION AL PROGRAMA "PLAN"

CAPITULO III

INTRODUCCION AL PROGRAMA "PLAN"

PRIMERA PARTE

3.1 INTRODUCCION

El programa "PLAN" está constituido principalmente por tres archivos: el "PLAN".EXE, el JED2BEQ.COM y el PAL2GAL.EXE.

Dentro de este programa también se incluyen los archivos de entrada los cuales contienen las ecuaciones booleanas y las etiquetas de los comandos. Estos archivos de entrada son utilizados por el ensamblador del programa "PLAN" para crear los mapas "JEDEC", los cuales contienen la información para programar el dispositivo final, como es: la lista de asignación de terminales, el número de fusibles para determinado producto, la cantidad de compuertas de cada dispositivo, la estructura de la salida y el tipo de retroalimentación (si es que la incluye).

El módulo ensamblador "PLAN".EXE tomará la información del diseño del "PAL" que previamente se introdujo en un archivo de entrada, mediante un editor de textos ASCII. Se escoge el "PAL" que puede implementar el diseño, se construye la lista de terminales y entonces se crea un archivo con el mapa "JEDEC" (con una extensión .JED).

El módulo JED2BEQ realizará la operación inversa del módulo "PLAN".EXE. Tomando un archivo "JEDEC", creará una ecuación booleana basada en un archivo de entrada del "PLAN" (con una extensión .BEQ).

El módulo PAL2GAL tomará un nombre de un dispositivo "PAL" y el archivo "JEDEC" creado para ese dispositivo y lo convertirá en un archivo "JEDEC" de "GAL" (Generic Array Logic).

3.2 INSTRUCCIONES DE OPERACION

PROPOSITO	MODULO	SECCION
+ Crear un diseño original.	ARCHIVO DE ENTRADA	3.3
+ Convertir un archivo de entrada a un mapa.	"JEDEC" "PLAN"	4
+ Convertir un mapa "JEDEC" a un archivo de entrada.	JED2BEQ	5
+ Convertir un mapa de "PAL" a un mapa de "GAL"	PAL2GAL	6

3.3 EL ARCHIVO DE ENTRADA

Es el mínimo requerimiento para programar un "PAL". En él se incluyen las funciones lógicas que desarrollará el "PAL" después de ser programado. El archivo de entrada requiere de una sintaxis que se explicará a continuación. Así mismo, este programa permite realizar otras funciones, éstas serán explicadas en otras secciones posteriores.

3.3.1 EL ARCHIVO "PLAN".EXE

El diseño de un "PAL" comienza con la creación de la ecuación booleana, la cual se obtiene del análisis de las compuertas a implementar.

El archivo de entrada se puede crear con editor de textos, como un archivo ASCII y con extensión ".BEQ". Un archivo de entrada también puede ser creado por el módulo JED2BEQ, aunque esto es, más bien, un proceso inverso al del módulo "PLAN".EXE.

Si la información dependiente, como lo es la asignación de terminales o el dispositivo con el que se requiere trabajar, no se conoce (ya sea en el archivo de entrada o en la línea de comando), entonces "PLAN" la determinará y actualizará el archivo de entrada anexando la información dependiente y con ella creará un archivo para el mapa "JEDEC".

3.3.2 FORMATO DE UN ARCHIVO DE ENTRADA

El formato del archivo de entrada debe tener extensión ".BEQ" y haber sido creado como un no-documento para que el módulo "PLAN".EXE lo reconozca y que con la información ahí contenida cree el mapa "JEDEC".

El formato que se debe respetar es el que se muestra a continuación:

Linea	Información del archivo
1era. línea	Nombre del Dispositivo ; Comentario
2a. línea	1a. mitad de la lista de terminales
3era. línea	2a. mitad de la lista de terminales
Subsecuentes	Información Funcional/Lógica (Ecuaciones Booleanas)

La primera línea contiene el nombre del dispositivo. Este debe estar en formato "PAL"XXXX para "PLD's" bipolares o "GAL"XXXX para "PLD's" "CMOS". Puede ser seguido por un comentario de campo, siempre y cuando los separe un punto y coma ";".

La lista de terminales se escribe en dos líneas en el archivo de entrada. La primera línea incluye la primera mitad de la lista y la segunda línea la otra mitad. La posición de la etiqueta corresponde directamente con el número de la terminal del dispositivo. El signo de las etiquetas de la lista de terminales no tiene correspondencia con el signo de las etiquetas de las ecuaciones.

Todas las terminales del dispositivo deben ser representadas en la lista de terminales. A las terminales no programables como la fuente, el reloj y los habilitadores globales se les puede asignar cualquier etiqueta. Cualquier terminal de entrada o salida no utilizada y programable debe ser asignado por 'nc' o por 'NC'. El caso 'nc' es la única etiqueta reservada en la lista de terminales y no puede ser utilizada en una ecuación.

Las tres primeras líneas son opcionales, es decir, se pueden incluir o no dentro del archivo de entrada.

Las líneas subsecuentes son obligatorias, ya que de ellas se pueden obtener, tanto el dispositivo como la asignación de terminales; y por tanto se puede crear el mapa "JEDEC". Sin embargo, con un archivo de entrada en el que sólo se incluya el dispositivo a utilizar y/o la asignación de terminales no es posible obtener el mapa "JEDEC".

De aquí que se consideren dos tipo de archivos: archivos completos e incompletos.

3.3.2.1 ARCHIVOS

3.3.2.1.1 ARCHIVOS COMPLETOS

Un archivo completo contiene dispositivos con información dependiente, (el nombre del dispositivo y la lista de "terminales").

3.3.2.1.2 ARCHIVOS INCOMPLETOS

Un archivo incompleto no contiene la lista de terminales pero puede incluir el nombre del dispositivo o viceversa. Es opcional (dispositivo dependiente) si no se incluye esta información. El mínimo archivo de entrada incompleto es una sóla ecuación en una sóla línea.

3.3.2.2 FORMATO DE LA ECUACION BOOLEANA

Las ecuaciones booleanas se encuentran especificadas por un archivo de entrada asignando solo un símbolo de salida a una expresión lógica. El operador asignado puede ser "=" o un ":=" dependiendo si la salida es combinacional o registrada, respectivamente. Una expresión lógica consiste de una secuencia de símbolos separados por operadores. Un símbolo puede ser cualquier combinación de caracteres alfanuméricos válidos. Los símbolos pueden tener de 1 a 16 caracteres .

3.3.2.3 OPERADORES

Un archivo de entrada requiere de los siguientes operadores:

.	PUNTO, comando identificador etiqueta/delimitador lista
=	Igualdad
*	"AND", producto
+	"OR", Suma
:=	"CLOCK" (después de la función de reloj. "clock")
:+:	"XOR" ."OR" exclusivo
/	Complemento (precede un símbolo)
;	Comentario delimitador de campo

3.3.3 FUNCIONES LOCALES Y GLOBALES

El lenguaje del programa "PLAN" y el formato del archivo de entrada permiten introducir los comandos tipo que son local/global, local, global y no-funcional.

La diferencia entre funciones globales y locales es que estas afectan sólo una salida del dispositivo y las primeras afectan varias salidas.

3.3.3.1 EJEMPLO DE SELECCION DE DISPOSITIVO

La siguiente lista contiene las descripciones de cinco funciones locales que en conjunto, puede constituir un archivo de entrada incompleto. El PLD que automaticamente será seleccionado por el ensamblador se indica a continuación de cada línea para cada tecnología que incluye el ensamblador "PLAN".

No.	Función	Circuito	TTL	ECL	CMOS
1	/O1 = I1	inversor	PAL16L8	PAL1012C4	GAL16V8
2	O2 = I2*I3	2 inp AND	PAL16P8	PAL1012C4	GAL16V8
3	O3 = O1 + I4	2 inp OR	PAL16P8	PAL1016P8	GAL16V8
4	O3.trst = I5	control	PAL16P8	s/seleccion	GAL16V8
5	O4: = I3 + I4	registrador	PAL16RP4	s/seleccion	GAL16V8

3.3.3.2 COMENTARIOS

Los comentarios pueden aparecer en muchos sitios de un archivo de entrada. Siempre deben ser precedidos de un delimitador de comentarios (;) y no deben contener comas.

El nombre del dispositivo puede ser seguido de un comentario en la misma línea. Si se debe seleccionar un dispositivo diferente para acomodar el diseño, el comentario será acomodado en el nuevo archivo de entrada o en el archivo actualizado. Este comentario es la única entrada directa a la que el usuario tiene acceso, y se ubicará en el campo de comentarios dentro del mapa "JEDEC".

Cualquier comentario que sigue a la información de la lista de terminales (en la misma línea) se perderá si el archivo de entrada se actualiza o se reemplaza. Los comentarios de una línea del archivo, que sigue a la información de la lista de terminales y antes de cualquier ecuación local lógica, será considerado como un error de sintaxis del archivo.

3.3.4 COMANDOS ETIQUETA/TIPO

Para describir aún más la función que realiza el circuito se pueden utilizar los comandos "etiqueta" y "tipo". El formato del archivo de entrada del "PLAN" del dispositivo independiente permite especificar con un lenguaje simple la lógica para que la arquitectura del dispositivo sea transparente para el usuario.
El ejemplo más simple para utilizar un comando etiqueta es '.trst'. Este tipo de lógica de 'ecuación de control local' (tipo 4) siempre sigue a la 'ecuación lógica local' (tipo 3) a la que

se aplica. El tipo n + 1 generalmente sigue a la ecuación/comando del tipo n excepto aquellas ecuaciones de control que siempre siguen la ecuación lógica que controlan.

3.3.4.1 COMANDO SECUENCIA/JERARQUIA

La jerarquía en que la información funcional-lógica se debe dar dentro del archivo de entrada del "PLAN" se detalla a continuación.

Comando	Comando
Tipo	Información tipo ecuación/comando funcional
0	Macros local/global
1	lógicos \ default
2	control /
	Local
3	lógicas \ ecuaciones
4	control /
	Funcional
*5	lógicas \ default
*6	control /
	Global
*7	lógicas \ ecuaciones
*8	control /
*9	listas \ No funcional
A	información /

* no son requeridos en esta versión del programa "PLAN".

Los comandos tipo números sirven solo para indicar la secuencia requerida del comando-ecuación funcional en el archivo de entrada. Los "defaults" son siempre opcionales en un archivo de entrada.

Casi todos los archivos utilizan muy poco de la capacidad total del lenguaje. El archivo de entrada más simple consiste de una sola ecuación [lógica local].

Las macros pueden ser utilizadas para hacer los archivos de entrada más sencillos de leer y editar. Las macros son, tanto, local como global, y son los primeros dispositivos no dependientes comando-ecuación que aparecen en un archivo de entrada. Esta función (tipo 0) requiere del uso del comando etiqueta '.term'.

Las macros son expandidas como sea requerido por el ensamblador para propósitos de selección y generación de mapa.

3.3.5 FUNCIONES GLOBALES

Las funciones globales (tipo 5-8) siempre siguen a las funciones locales (tipo 1-4).

3.3.5.1 UES; FIRMA ELECTRONICA DEL USUARIO

La información puede ser grabada en la porción del UES (Firma Electrónica del Usuario) del mapa "JEDEC". Esto se realiza con el archivo de entrada del ensamblador '.data' (tipo A) comando/ecuación. Un PLD no puede ser aceptado o seleccionado por el ensamblador o por el mapa "JEDEC" si todos los caracteres destinados por el campo del UES no pueden ser acomodados.

El ensamblador procesa la serie de caracteres en la porción del enunciado ".data". El enunciado ".data." incluye caracteres del código ASCII del 33 al 126.

Una línea de comentario es adicionada al mapa "JEDEC" por el ensamblador para desplegar el caracter tipo y utilizar el campo del UES.

3.3.5.2 SINTAXIS DE ECUACION /COMANDO
(Sección 3.3.4.1 y Sección 3.3.6)

La sintaxis de la ecuación/comando para la información función/lógica es:

Sintaxis - ecuación/comando funcional	comando tipo
A - etiqueta macro.etiqueta comando = expresión lógica	0
*B - etiqueta .comando	1,5
C - [modificador] etiqueta .comando = expresión lógica	2,6,8
D - símbolo de etiqueta .comando = expresión lógica	3,4,7,8
E - símbolo .{etiqueta comando} := expresión lógica	3,7
*F - etiqueta . comando = {.símbolo}	9
G - etiqueta . comando = información	A

* no utilizado en esta versión del programa "PLAN".

Las ecuaciones lógicas constan de:

símbolo = expresión lógica
símbolo := expresión lógica

donde:

símbolo	1-16 alfanuméricos/minúsculas/TILDE
etiqueta macro	1-8 alfanuméricos/minúsculas/TILDE
etiqueta comando	4 caracteres alfanuméricos
expresión lógica	secuencia de símbolos y operadores
información	caracteres ASCII o HEXADECIMALES
[]	como se requieran
{}	repetición

3.3.5.3 REGLAS DE ORDEN

Se deben respetar las siguientes reglas básicas de órden y tipo de secuencia en el archivo

1 - Las macros aparecen primero, tanto las locales como las globales.
2 - Las funciones locales se describen antes de cualquier función global.
3 - Las ecuaciones aparecen antes de las funciones controladoras.
4 - Los "defaults " ocurren en el archivo antes que las funciones afectadas.
5 - La información no-funcional es la última, no teniendo efecto local o global.

3.3.5.4 "DEFAULTS" DE REGISTRO DEL PROGRAMA

Los "defults" son asumidos a menos que se indique lo contrario.
Algunos hechos que asume el programa por default son:

Tipo-D, eje positivo disparador, "flip-flop"
* El reloj se asigna con 1, la etiqueta es C1 la salida está disponible a la terminal de salida (si está habilitada).
* El registro se reinicializa cuando se le da reset (lógica 0)
Todas las funciones del registro son asíncronas
* Inicialmente la función 1 es asignada
El arreglo de retroalimentación es desde la salida del registro (ninguna-E/S)
* no se aplica a esta versión del ensamblador "PLAN"

3.3.5.5 EJEMPLOS COMPLICADOS

La lista que se muestra a continuación puede constituir un archivo de entrada incompleto.

#	Función	Comando	Tipo	Opciones PLD
1	t1.term = I1*/I2	macro	0	n/a
2	.clkf = /I3*I5	default	2	PAL16A8,PAL20RA10
3	O1 = /I1	lógica	3	no cambia
4	O3 = O1 + I4	lógica	3	no cambia
5	O3.trst = I5	control	4	no cambia
6	O4: = I3 + I4	lógica	3	no cambia
7	.data = help	UES	A	ninguna

3.3.6 COMANDO ETIQUETAS

A continuación se listan los comandos etiquetas y tipo disponibles. El comando tipo define como se puede utilizar el comando para describir la función lógica de los circuitos. Algunos comandos son utilizados si el "default" del programa no es aceptado. (Sección 3.3.4.1 y 3.3.5.2)

Etiqueta	Tipo	Sintaxis	Función
.CLKF	2,4	D	Definición de reloj local
.CLKN	3	E	Asignación de reloj
.DATA	A	G	Información para UES
.LSCC	1	B	Acarreo de salida del CI
.RSTF	2,4	D	"Reset" del registro local
.RSTF	8	C	"Reset" del registro global
.SETF	2,4	D	"Set" del registro local
.SETF	8	C	"Set" del registro global
.SYNC	8	C	Modificador síncrono
.TERM	0	A	Macro
.TRST	2,4	D	Habilitador local de la salida

NOTA: "Default" = todos los circuitos disponibles, n = 1,2

SEGUNDA PARTE

3.4 EL ENSAMBLADOR

3.4.1 GENERALIDADES

Al ensamblador "PLAN" se le introduce el archivo de entrada y con él crea el archivo "JEDEC". Mediante la sintaxis del programa se logra que los archivos que han sido diseñados con diferentes tecnologías y con diferentes grados de complejidad sean compatibles entre sí.
En un archivo se introducen las especificaciones de diferentes dispositivos y de acuerdo a él, el ensamblador "PLAN" verificará si los dispositivos pueden ser aceptados por la lógica. Si existe alguna razón por la que la lógica no acepte el dispositivo, el ensamblador la dará; al mismo tiempo que escogerá un dispositivo que sí cumpla con los requerimientos del diseño.
Al agotar las posibilidades de escoger un dispositivo "TTL", analizará las posibilidades de escoger un dispositivo "CMOS", para cubrir perfectamente las necesidades del diseño.

3.4.1.1 REFERENCIA RAPIDA

En la línea del comando se deberá teclear:

"PLANXXXXXX"

donde XXXXXX debe ser sustituido por el nombre del archivo en el que se encuentra el diseño lógico y que tiene una extensión .BEQ.
Así como este parámetro puede ser utilizado para controlar el despliegue del procedimiento, es posible realizar otras evasiones tomando en cuenta la siguiente tabla de "interruptores":

1.- Para deshabilitar el despliegue de:

E - el archivo de entrada mientras ensambla
T - el número de términos de producto
M - el mapa "JEDEC"
P - el diagrama del circuito integrado que muestra la asignación de las terminales

2.- Los que añaden:

G - el campo 'G1*' al mapa, programa de celda de seguridad.
Vn - n número de vectores al mapa "JEDEC" (el archivo de vectores debe estar en el disco).

3.- Los que despliegan:

C - el "chip carrier" en el "DIP", si éste está habilitado
X - el diagrama de terminales para encapsulados de 28 terminales "PLCC".

Cualquier combinación de los interruptores puede ser utilizada en cualquier órden mientras se ejecuta el programa de ensamble y/o para definir el número de vectores de prueba disponibles para crear el mapa "JEDEC". Todos estos parámetros deben ser precedidos de una diagonal "/" y esta a su vez del nombre del archivo. En medio de la diagonal y del nombre del archivo no debe existir espacio.

3.4.2 INSERTANDO LA LINEA DE COMANDO

Si el archivo de entrada está completo y correcto, no será necesario incluir mayor información al programa, y el mapa "JEDEC" será creado. Si el archivo no es completo o es incorrecto, el programa pedirá los datos necesarios para completar o corregir el archivo y después, poder crear el mapa "JEDEC".

3.4.2.1 FORMATO DE LA LINEA DE COMANDO

El formato que se debe seguir es el que se muestra a continuación:

"PLAN" (archivo/interruptores) (-oarchivo) (-dnombre del dispositivo)

los parámetros que se encuentran entre paréntesis son opcionales y el órden de aparición es variable.

3.4.2.1 EL DESPLIEGUE

Mientras el archivo de entrada está siendo analizado, en la pantalla se despliega las líneas del archivo. Mediante apuntadores se van indicando las secciones aceptadas por el análisis, y cuando ocurre un error, el último indicador indicará la ubicación del error. Este despliegue se deshabilita con el interruptor "E".
El número de mini-términos se despliega mientras se crea el mapa "JEDEC". Este parámetro se deshabilita con el interruptor "T".
El mapa "JEDEC" se despliega y se escribe en el disco a menos que sea deshabilitado mediante el interruptor "M".

3.4.2.2 NOMBRE DEL ARCHIVO

El archivo de entrada debe tener una extensión ".BEQ", pero esta extensión no debe ser incluida cuando se escribe la línea de comando.

3.4.2.3 BANDERAS

Las banderas con sus correspondientes parámetros, pueden ser añadidas a la línea de comando, en cualquier órden. Las banderas disponibles en esta versión del programa "PLAN" son "-d" (bandera de dispositivo) y "-o" (bandera de nuevo nombre de archivo).

3.4.2.3.1 NUEVO NOMBRE DE ARCHIVO

(-Oarchivo) define un nuevo nombre de archivo. Si un archivo nuevo de entrada está completo y no contiene errores, es superflúo. Si el archivo de entrada requiere actualizarse por alguna razón para justificar la creación del mapa "JEDEC", y un nombre nuevo de archivo no ha sido dado en la línea comando, entonces el ensamblador preguntará por el nombre nuevo. Si el nombre nuevo es el mismo que el original, entonces sobreescribirá el archivo.

3.4.2.3.2 NOMBRE DEL DISPOSITIVO LOGICO PROGRAMABLE Y TECNOLOGIA

La opción (-dpldnombre) ayuda a resolver la selección del dispositivo. El nombre del "PLD" seguirá inmediatamente la bandera -d, y puede especificar únicamente la tecnología o el nombre completo del dispositivo. Si hay un dispositivo aceptable en el archivo de entrada, entonces esta especificación no se toma en cuenta. En caso de no ser aceptado, el programa "PLAN" elegirá la tecnología adecuada al problema. Las opciones válidas para dar esta especificación son:

"CMOS"	palabra clave para esta tecnología.
"TTL"	palabra clave para esta tecnología.
"ECL"	palabra clave para esta tecnología.
PALXXXXX	nombre específico para dispositivos "TTL" o "ECL".
GALXXXXX	nombre específico para dispositivos "CMOS".

Debido a la evolución del mercado de "PLD"s, los "PAL"s de mediana capacidad han mejorado su relación utilidad-costo contra los "PAL"s menores, y esta versión del "PLAN" considera este hecho.

3.4.3 VECTORES DE PRUEBA

Esta versión de ensamblador "PLAN" puede incluir vectores de prueba en el mapa "JEDEC". Los vectores deben estar propiamente formateados y residir en el disco con el nombre 'nombredearchivo.VEC' donde el nombre de archivo debe ser el mismo que el del archivo de entrada '.BEQ'. Los vectores son llamados con el interruptor 'v'. definido en la sección B.2.2.1

3.4.3.1 FORMATO DEL ARCHIVO DE VECTORES

Esta versión de programa "PLAN" no provee la generación de vectores. El archivo de vectores puede ser creado con un editor, siguiendo las siguientes reglas:

1) Todos los campos deben regirse por el estándar "JEDEC" No.3A

2) El campo 'p' debe estar en la primera línea, si es incluido.

3) Los campos 'qv' y 'x' son opcionales en el archivo.

4) Unicamente se permite un vector por línea.

5) El número de vectores opcionales usa un espacio delimitado.

6) Los comentarios siguen a los vectores en la misma línea.

Si es usado el interruptor 'v', debe especificarse un número, si el número excede al número de vectores en el archivo, el ensamblador indicará que todos los vectores disponibles han sido incluidos en el mapa "JEDEC".

3.4.3.2 CARACTERES DE VECTORES

Una referencia completa de vectores se especifica en el estándard "JEDEC" No. 3A. Un vector de prueba es una palabra compuesta por caracteres. Un caracter de un vector es requerido por cada terminal del dispositivo. Y existe correspondencia directa entre la localidad del caracter en la palabra y la terminal del dispositivo, donde el primer caracter en el campo corresponde a la terminal 1 del dispositivo y así sucesivamente. La siguiente tabla lista los caracteres aceptables (condiciones de prueba) para la creación de vectores de prueba.

Caracter	Definición
0	Entrada lógica baja al circuito
1	Entrada lógica alta al circuito
2-9	Entrada alta al dispositivo a un sobrevoltaje
B	Precarga de registro
C	Pulso positivo de reloj (bajo-alto-bajo)
F	Terminal del dispositivo no conectado
H	Salida alta del circuito
K	Pulso negativo del reloj (alto-bajo-alto)
L	Salida baja del circuito
N	Alimentación o terminales de salida sin prueba
P	Precarga de registros
X	Valor de entrada por default o salida no probada
Z	Prueba de terminales del dispositivo en alta impedancia.

CAPITULO IV

PUERTO PARALELO (PRACTICA)

CAPITULO IV

PRACTICA

PUERTO PARALELO

Mediante esta práctica se pretende ilustrar lo analizado en los capitulos anteriores. Siguiendo paso a paso la aplicación aquí presentada se entenderá la evolución tecnológica que esta tesis pretende mostrar. Precediendo la aplicación se proporcionará una breve introducción teórica y posteriormente el desarrollo de la misma.

4.1 Introducción Teorica

Fisicamente se trata de un conector macho de 25 terminales al cual se conecta el cable de comunicación, que puede utilizarse en general, como un puerto de entrada/salida para cualquier dispositivo que cumpla las normas del puerto paralelo. Tiene 12 puntos de salida TTL "buffer", las cuales pueden ser enganchados y leidos o escritos bajo control de programa utilizando las instrucciones del microprocesador "IN" y "OUT".

Cuando se utiliza para conectar una impresora, los comandos son cargados hacia un puerto de salida de o "bits" enganchados y es activada la línea de "strobe", escribiendo así la información en la impresora. El programa podrá, entonces, leer los puertos de entrada para conocer el estado de impresión e indicar cuando se puede escribir el siguiente caracter, o puede ocupar la línea de interrupción para indicar al programa que no está ocupado.

Los puertos de salida pueden ser leidos, también, en la tarjeta de interfaz para funciones de lazo de diagnósticos. Esto permite aislar fallas en el adaptador o en el dispositivo conectado. Este puerto consta, principalmente, de cinco instrucciones de E/S: dos salidas transfieren información a dos "latches" cuyas salidas están presentes en las terminales del caparazón del conector.

Dos de las tres instrucciones de entrada permiten al microprocesador del sistema leer el contenido de los dos "latches". El tercero permite al microprocesador del sistema leer el estado del tiempo-real desde un grupo de terminales del conector.

Enseguida se encuentra la descripción de cada sistema:

La instrucción captura información del canal de datos y está presente en las terminales

SALIDA DE LA DIRECCION HEX 378			
Bit 3 Term 5	Bit 2 Term 4	Bit 1 Term 3	Bit 0 Term 2

respectivas. Cada una de estas terminales tienen la capacidad de mantener una fuente de 2.6mA y drenar 24mA.

Es esencial que el dispositivo externo no trate de mandar estas líneas a tierra.

ADAPTADOR DE LA IMPRESORA SALIDA DE LA DIRECCION HEX 37A			
Bit 3 Term 17	Bit 2 Term 16	Bit 1 Term 14	Bit 0 Term 1

Esta instrucción provoca el enganche para capturar los cinco "bits" menos significativos del canal de datos. Los cuatro "bits" menos significativos presentan sus salidas, o versiones invertidas de sus salidas, a las terminales respectivas como se muestra en figura anterior. Si el "bit" 4 se encuentra en "1" lógico, la tarjeta interrumpirá la unidad del microprocesador del sistema con la condición de que la terminal 10 cambie de estado alto a bajo.

Estas terminales son manejadas por los "drivers" colector-abierto forzados a +5Vdc a través de las resistencias de 4.7k. Cada una puede drenar 7mA aproximadamente y mantener - 0.8volts.

ADAPTADOR DE LA IMPRESORA

ENTRADA DESDE LA DIRECCION HEX 378

Esta instrucción presenta la unidad del sistema del microprocesador con datos presentes en las terminales asociadas con la salida de HEX 3BC. Esto deberá reflejar, normalmente, el último valor exacto que fué escrito en dicha dirección. Si un dispositivo externo manejara datos en estas terminales en el momento de una entrada (violando las reglas de uso de tierra), estos datos serán "ORed" con el contenido del "latch".

ADAPTADOR DE LA IMPRESORA

ENTRADA DESDE LA DIRECCION HEX 379

Esta instrucción presenta el estado del tiempo-real a la unidad del microprocesador del sistema desde las terminales como se muestra a continuación:

"Bit" 7 "Bit" 6 "Bit" 5 "Bit"4 "Bit"3 "Bit"2 "Bit"1 "Bit"0

Term 11 Term 10 Term 12 Term13 Term15 ---- ---- ----

ENTRADA DESDE LA DIRECCION HEX 37A

Esta instrucción provoca que la unidad del micropocesador del sistema lea los datos de las terminales 1, 14, 16, 17 y el "bit" "IRQ". En ausencia de impulso externo aplicado a estas terminales, los datos leidos por la unidad del microprocesador del sistema comparará los últimos datos escritos en HEX 3BE en la posición de estos "bits". Obsérvese que los "bits" de datos 0-2 no se han incluido. Si los "drivers" externos son apuntados a esas mismas terminales, esos datos serán manejados como una compuerta "Or" con datos aplicados a las terminales por el "latch" de la dirección hex3BE.

"Bit" 7"Bit" 6"Bit" 5"Bit"4"Bit"3"Bit"2"Bit"1"Bit"0

IRQ $\overline{\quad\quad}$ $\overline{\quad\quad\quad}$
EnableTerm17Term16Term14Term1
Por = 0 Por = 1 Por = 0 Por = 1 Por = 1

Estas terminales asumen los estados mostrados después de que la unidad del microprocesador del sistema ha enviado una señal de regreso a cero.

4.2 Desarrollo Práctico

Se armó el diseño original en un tableta de pruebas de acuerdo a un diagrama estandar. Posteriormente se analizaron los bloques que se deseaban sustituir por un Arreglo Lógico Programable (PAL). Enseguida se procedió de acuerdo a los cinco pasos para diseñar con lógica programable:

1) Definir el problema

Este circuito puede considerarse como combinatorio, ya que encontramos retroalimentación en una sección de él; aunque la mayoría de él es un circuito secuencial. Este circuito será divido en tres bloques, tomando en cuenta la ubicación de los componentes, así como la función que ellos realizan.

2) Selección del dispositivo

En el primer bloque tenemos diez entradas y tres salidas, según se muestra a continuación:

La salida uno (S1) depende de la salida dos (S2) y de la entrada B14.
La salida dos (S2) depende de las entradas A24, A11, A28, A27, A26, A25, A22 y A23
La salida tres (S3) depende de la entrada B02.

En el segundo bloque tenemos seis entradas y seis salidas. Este bloque se trata de un inversor, basicamente, así es que la salida general del dispositivo lógico programable será logicamente bajo.

La salida cuatro (S4) depende de la entrada A1.
La salida cinco (S5) depende de la entrada A2.
La salida seis (S6) depende de la entrada A3.
La salida siete (S7) depende de la entrada A4.
La salida ocho (S8) depende de la entrada A5.
La salida nueve (S9) depende de la entrada A4.

En el tercer bloque contamos con cuatro entradas y cuatro salidas. Este bloque quedo dividido así, ya que se encuentra después del conector. También se trata de un circuito inversor cuyas salidas serán negadas.

La salida diez (S10) depende de la entrada B1.
La salida once (S11) depende de la entrada B2.
La salida doce (S12) depende de la entrada B3.
La salida trece (S13) depende de la entrada B4.

3) Escribir las ecuaciones lógicas

Al tener definidas el número de entradas y salidas de cada bloque que se analizó, fué posible generar la tabla de verdad para cada salida. Estas se muestran a continuación, clasificadas para cada bloque, y en estas, a su vez, para cada salida.

Primer bloque

Entradas		Salida
B14	S2	S1
0	0	1
0	1	0
1	0	0
1	1	0

Entradas								Salidas
A28	A27	A26	A25	A22	A23	A24	A11	S2
0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	1	0	1
0	0	0	0	0	0	1	1	1
0	0	0	0	0	1	0	0	1
0	0	0	0	0	1	0	1	1
0	0	0	0	0	1	1	0	1
0	0	0	0	0	1	1	1	1
0	0	0	0	1	0	0	0	1
0	0	0	0	1	0	0	1	1
0	0	0	0	1	0	1	0	1
0	0	0	0	1	0	1	1	1
0	0	0	0	1	1	0	0	1
0	0	0	0	1	1	0	1	1
0	0	0	0	1	1	1	0	1
0	0	0	0	1	1	1	1	1
0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	0	1	1
0	0	0	1	0	0	1	0	1
0	0	0	1	0	0	1	1	1
0	0	0	1	0	1	0	0	1
0	0	0	1	0	1	0	1	1
0	0	0	1	0	1	1	0	1
0	0	0	1	0	1	1	1	1
0	0	0	1	1	0	0	0	1
0	0	0	1	1	0	0	1	1
0	0	0	1	1	0	1	0	1
0	0	0	1	1	0	1	1	1

0	0	0	1	1	1	0	0		1
0	0	0	1	1	1	0	1		1
0	0	0	1	1	1	1	0		1
0	0	0	1	1	1	1	1		1
0	0	1	0	0	0	0	0		1
0	0	1	0	0	0	0	1		1
0	0	1	0	0	0	1	0		1
0	0	1	0	0	0	1	1		1
0	0	1	0	0	1	0	0		1
0	0	1	0	0	1	0	1		1
0	0	1	0	0	1	1	0		1
0	0	1	0	0	1	1	1		1
0	0	1	0	1	0	0	0		1
0	0	1	0	1	0	0	1		1
0	0	1	0	1	0	1	0		1
0	0	1	0	1	0	1	1		1
0	0	1	0	1	1	0	0		1
0	0	1	0	1	1	0	1		1
0	0	1	0	1	1	1	0		1
0	0	1	0	1	1	1	1		1
0	0	1	1	0	0	0	0		1
0	0	1	1	0	0	0	1		1
0	0	1	1	0	0	1	0		1
0	0	1	1	0	0	1	1		1
0	0	1	1	0	1	0	0		1
0	0	1	1	0	1	0	1		1
0	0	1	1	0	1	1	0		1
0	0	1	1	0	1	1	1		1
0	0	1	1	1	0	0	0		1
0	0	1	1	1	0	0	1		1
0	0	1	1	1	0	1	0		1
0	0	1	1	1	0	1	1		1
0	0	1	1	1	1	0	0		1
0	0	1	1	1	1	0	1		1
0	0	1	1	1	1	1	0		1
0	0	1	1	1	1	1	1		1
0	1	0	0	0	0	0	0		1
0	1	0	0	0	0	0	1		1
0	1	0	0	0	0	1	0		1
0	1	0	0	0	0	1	1		1
0	1	0	0	0	1	0	0		1
0	1	0	0	0	1	0	1		1
0	1	0	0	0	1	1	0		1
0	1	0	0	0	1	1	1		1
0	1	0	0	1	0	0	0		1
0	1	0	0	1	0	0	1		1

0	1	0	0	1	0	1	0	1
0	1	0	0	1	0	1	1	1
0	1	0	0	1	1	0	0	1
0	1	0	0	1	1	0	1	1
0	1	0	0	1	1	1	0	1
0	1	0	0	1	1	1	1	1
0	1	0	1	0	0	0	0	1
0	1	0	1	0	0	0	1	1
0	1	0	1	0	0	1	0	1
0	1	0	1	0	0	1	1	1
0	1	0	1	0	1	0	0	1
0	1	0	1	0	1	0	1	1
0	1	0	1	0	1	1	0	1
0	1	0	1	0	1	1	1	1
0	1	0	1	1	0	0	0	1
0	1	0	1	1	0	0	1	1
0	1	0	1	1	0	1	0	1
0	1	0	1	1	0	1	1	1
0	1	0	1	1	1	0	0	1
0	1	0	1	1	1	0	1	1
0	1	0	1	1	1	1	0	1
0	1	0	1	1	1	1	1	1
0	1	1	0	0	0	0	0	1
0	1	1	0	0	0	0	1	1
0	1	1	0	0	0	1	0	1
0	1	1	0	0	0	1	1	1
0	1	1	0	0	1	0	0	1
0	1	1	0	0	1	0	1	1
0	1	1	0	0	1	1	0	1
0	1	1	0	0	1	1	1	1
0	1	1	0	1	0	0	0	1
0	1	1	0	1	0	0	1	1
0	1	1	0	1	0	1	0	1
0	1	1	0	1	0	1	1	1
0	1	1	0	1	1	0	0	1
0	1	1	0	1	1	0	1	1
0	1	1	0	1	1	1	0	1
0	1	1	0	1	1	1	1	1
0	1	1	1	0	0	0	0	1
0	1	1	1	0	0	0	1	1
0	1	1	1	0	0	1	0	1
0	1	1	1	0	0	1	1	1
0	1	1	1	0	1	0	0	1
0	1	1	1	0	1	0	1	1
0	1	1	1	0	1	1	0	1
0	1	1	1	0	1	1	1	1

0	1	1	1	1	0	0	0	1
0	1	1	1	1	0	0	1	1
0	1	1	1	1	0	1	0	1
0	1	1	1	1	0	1	1	1
0	1	1	1	1	1	0	0	1
0	1	1	1	1	1	0	1	1
0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	1	1
1	0	0	0	0	0	1	0	1
1	0	0	0	0	0	1	1	1
1	0	0	0	0	1	0	0	1
1	0	0	0	0	1	0	1	1
1	0	0	0	0	1	1	0	1
1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	0	1
1	0	0	0	1	0	0	1	1
1	0	0	0	1	0	1	0	1
1	0	0	0	1	0	1	1	1
1	0	0	0	1	1	0	0	1
1	0	0	0	1	1	0	1	1
1	0	0	0	1	1	1	0	1
1	0	0	0	1	1	1	1	1
1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	1	1
1	0	0	1	0	0	1	0	1
1	0	0	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	0	1	0	1	0	1	1
1	0	0	1	0	1	1	0	1
1	0	0	1	0	1	1	1	1
1	0	0	1	1	0	0	0	1
1	0	0	1	1	0	0	1	1
1	0	0	1	1	0	1	0	1
1	0	0	1	1	0	1	1	1
1	0	0	1	1	1	0	0	1
1	0	0	1	1	1	0	1	1
1	0	0	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1
1	0	1	0	0	0	0	0	1
1	0	1	0	0	0	0	1	1
1	0	1	0	0	0	1	0	1
1	0	1	0	0	0	1	1	1
1	0	1	0	0	1	0	0	1
1	0	1	0	0	1	0	1	1

1	0	1	0	0	1	1	0		1
1	0	1	0	0	1	1	1		1
1	0	1	0	1	0	0	0		1
1	0	1	0	1	0	0	1		1
1	0	1	0	1	0	1	0		1
1	0	1	0	1	0	1	1		1
1	0	1	0	1	1	0	0		1
1	0	1	0	1	1	0	1		1
1	0	1	0	1	1	1	0		1
1	0	1	0	1	1	1	1		1
1	0	1	1	0	0	0	0		1
1	0	1	1	0	0	0	1		1
1	0	1	1	0	0	1	0		1
1	0	1	1	0	0	1	1		1
1	0	1	1	0	1	0	0		1
1	0	1	1	0	1	0	1		1
1	0	1	1	0	1	1	0		1
1	0	1	1	0	1	1	1		1
1	0	1	1	1	0	0	0		1
1	0	1	1	1	0	0	1		1
1	0	1	1	1	0	1	0		1
1	0	1	1	1	0	1	1		1
1	0	1	1	1	1	0	0		1
1	0	1	1	1	1	0	1		1
1	0	1	1	1	1	1	0		1
1	0	1	1	1	1	1	1		1
1	1	0	0	0	0	0	0		1
1	1	0	0	0	0	0	1		1
1	1	0	0	0	0	1	0		1
1	1	0	0	0	0	1	1		1
1	1	0	0	0	1	0	0		1
1	1	0	0	0	1	0	1		1
1	1	0	0	0	1	1	0		1
1	1	0	0	0	1	1	1		1
1	1	0	0	1	0	0	0		1
1	1	0	0	1	0	0	1		1
1	1	0	0	1	0	1	0		1
1	1	0	0	1	0	1	1		1
1	1	0	0	1	1	0	0		1
1	1	0	0	1	1	0	1		1
1	1	0	0	1	1	1	0		1
1	1	0	0	1	1	1	1		1
1	1	0	1	0	0	0	0		1
1	1	0	1	0	0	0	1		1
1	1	0	1	0	0	1	0		1
1	1	0	1	0	0	1	1		1

1	1	0	1	0	1	0	0		1
1	1	0	1	0	1	0	1		1
1	1	0	1	0	1	1	0		1
1	1	0	1	0	1	1	1		1
1	1	0	1	1	0	0	0		1
1	1	0	1	1	0	0	1		1
1	1	0	1	1	0	1	0		1
1	1	0	1	1	0	1	1		1
1	1	0	1	1	1	0	0		1
1	1	0	1	1	1	0	1		1
1	1	0	1	1	1	1	0		1
1	1	0	1	1	1	1	1		1
1	1	1	0	0	0	0	0		1
1	1	1	0	0	0	0	1		1
1	1	1	0	0	0	1	0		1
1	1	1	0	0	0	1	1		1
1	1	1	0	0	1	0	0		1
1	1	1	0	0	1	0	1		1
1	1	1	0	0	1	1	0		1
1	1	1	0	0	1	1	1		1
1	1	1	0	1	0	0	0		1
1	1	1	0	1	0	0	1		1
1	1	1	0	1	0	1	0		1
1	1	1	0	1	0	1	1		1
1	1	1	0	1	1	0	0		1
1	1	1	0	1	1	0	1		1
1	1	1	0	1	1	1	0		1
1	1	1	0	1	1	1	1		1
1	1	1	1	0	0	0	0		1
1	1	1	1	0	0	0	1		1
1	1	1	1	0	0	1	0		1
1	1	1	1	0	0	1	1		1
1	1	1	1	0	1	0	0		1
1	1	1	1	0	1	0	1		1
1	1	1	1	0	1	1	0		1
1	1	1	1	0	1	1	1		1
1	1	1	1	1	0	0	0		1
1	1	1	1	1	0	0	1		1
1	1	1	1	1	0	1	0		1
1	1	1	1	1	0	1	1		1
1	1	1	1	1	1	0	0		0
1	1	1	1	1	1	0	1		1
1	1	1	1	1	1	1	0		1
1	1	1	1	1	1	1	1		1

Entradas	Salida
B02	S3
0	1
1	0

Para ayudar al desarrollo de la reducción de esta salida se tomaron en un sólo bloque las variables A28, A27, A26, A25, A22 asignándole al nuevo bloque el nombre de A. Así tenemos ahora la siguiente tabla de verdad.

Entradas				Salida
A	A23	A24	A11	S2
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Segundo Bloque

Entradas	Salida
A1	S4
0	1
1	0

Entradas	Salida
A2	S5
0	1
1	0

Entradas	Salida
A3	S6
0	1
1	0

Entradas	Salida
A4	S7
0	1
1	0

Entradas	Salida
A5	S8
0	1
1	0

Entradas	Salida
A4	S9
0	0
1	1

En el tercer bloque

Entradas	Salida
B1	S10
0	1
1	0

Entradas	Salida
B2	S11
0	1
1	0

Entradas	Salida
B3	S12
0	1
1	0

Entradas	Salida
B4	S13
0	1
1	0

A partir de estas tablas se pasó a trabajar con los mapas de Karnaugh para lograr reducir las ecuaciones lo más posible y obtenerlas en suma de productos. Esto es por que el programa que seleccionará el dispositivo correspondiente, así lo requiere. La reducción de ecuaciones sólo fué necesaria en el primer bloque, ya que los otros dos bloques no lo requirieron así.

Enseguida podemos observar los mapas para las salidas S1 y S2:

Mapa de Karnaugh para la salida S1

S2\B14	0	1
0	1	0
1	0	0

De aquí que la ecuación minimizada para la salida S1 sea:

$$S1 = /B14 * /S2 \qquad \text{Ecuación 1}$$

Mapa de Karnaugh para la salida S2

A24 A11\A A23	00	01	11	10
00	1	1	0	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

De este mapa obtenemos la ecuación minimizada para la salida S2, la cual se muestra a continuación:

$$S2 = A24 + A11 + /A + /A23 \qquad \text{Ecuación 2}$$

Sustituyendo en esta ec. 2 las variables que corresponden al bloque de la variable A, tenemos que:

$$S2 = A24 + A11 + /A28 + /A27 + /A26 + /A25 + /A22 + A23 \qquad \text{Ecuación 3}$$

Y ahora, sustituyendo la ecuación 3 en la ecuación 1, obtenemos:

$$S1 = /B14(A24 + A11 + /A28 + /A27 + /A26 + /A25 + /A22 + \overline{/A23}) \quad \text{Ecuación 4}$$

Utilizando la siguiente Ley de Morgan par Algebra Boolena en la ecuación 4:

$$\overline{(/A + /B)} = A * B$$

tenemos que finalmente la salida S1 es:

$$S1 = /B14(/A24*/A11*A28*A27*A26*A25*A22*A23) \quad \text{Ecuación 5}$$

Y para la salida S3, tenemos que:

$$S3 = /B02 \quad \text{Ecuación 6}$$

Las ecuaciones para las salidas del bloque 2 y 3 se muestran a continuación:

Bloque 2

S4 = /A1
S5 = /A2
S6 = /A3
S7 = /A4
S8 = /A5
S9 = A4
S10 = /B1
S11 = /B2
S12 = /B3
S13 = /B4

4) Programacion del dispositivo.

Las ecuaciones 2,5 y 6 se introdujeron en un editor de textos con extensión .BEQ y en un archivo ASCII (formato no-documento); con esto se obtuvo el archivo de entrada para el primer bloque del circuito. Después se corrió el programa "PLAN" y éste escogió el dispositivo PAL que correspondió. La elección no fué inmediata ya que la ecuación que corresponde a la salida S2 contenía demasiadas sumas y el programa eligió un dispositivo lógico programable que al ser de gran capacidad su disponibilidad no es tan inmediata, éste circuito fué el PAL22V10. Sobretodo porque con un dispositivo lógico programable con polaridad programada de menor capacidad es posible implementar dicha función. Por lo tanto, se separó la ecuación de la salida S2 en dos y al introducirse en el programa, éste eligió el PAL16P8.

Para el segundo y tercer bloque también se procedió a introducir las ecuaciones en un editor de textos con extensión .BEQ y formato no-documento, creando así los archivos de entrada para cada bloque. Se le dió esta información al programa "PLAN" y éste eligió al dispositivo PAL16P8, para ambos bloques. Con esta elección nuestro diseño queda uniforme con el Arreglo Lógico Programable PAL16P8.

Sin embargo, este código no satisfacía los requerimientos de facilidad de implementación, ya que no es tan comercial, como lo que se requería. Así que se decidió forzar al programa a escoger otro dispositivo que cumpliera con más características del diseño.

De esta manera se observó que en su mayoría se trataba de circuitos inversores y que las salidas dentro del archivo de entrada habían sido indicadas como positivas, así que se procedió a obtener salidas negadas, por medio de las Leyes de Morgan para el Algebra Booleana.

Estas nuevas ecuaciones fueron introducidas nuevamente en el programa ""PLAN"", el cual escogió el PAL16L8 para implementar los tres diseños. Entonces creó el mapa de fusibles de acuerdo al estandar JEDEC; éste fue guardado en el archivo correspondiente a cada bloque, con extensión .JED. Los mapas de fusibles creados por el programa para cada una de las tres posibilidades de implementación pueden consultarse en el apéndice.

Una vez que se obtuvieron los mapas de fusibles se procedió a realizar la programación física del dispositivo. Para ello se utilizó el programa que corresponde al programador.

Mediante el menú principal a indicarle al programa la marca y el código del dispositivo que se va a utilizar. En este caso NSC y PAL16L8 respectivamente. También se le indicó el archivo en el que debe buscar la información para cada bloque, según se muestra a continuación:

Bloque 1 TESISPA7.JED
Bloque 2 TESISA3.JED
Bloque 3 TESISB3.JED

Se verificó que el dispositivo estuviera en blanco y se procedió a realizar la programación física de cada uno de ellos.

Se colocó el dispositivo en una base según las instrucciones gráficas del programador. Se aseguró el dispositivo y eligiendo la opción P del menú principal, se procedió a verificar y programar el dispositivo. Después con la opción R se leyó el estado lógico del dispositivo, después con la opción 5 se editó dicho estado verificando visualmente que el archivo JEDEC creado había sido cargado correctamente en el circuito.

Enseguida se procedió a abrir el fusible de seguridad. Quedando así la programación concluida.

CAPITULO V

LA NUEVA TECNOLOGIA DE "PAL'S"

CAPITULO V

TECNOLOGIA DE FUSIBLE VERTICAL

Esta tecnología es utilizada en los dispositivos lógicos programables "TTL" de mayor velocidad. El fusible vertical utiliza una tecnología de programación conocida como "Avalanche Induced Migration" "(AIM)". A continuación se describirán las características de la tecnología de fusible vertical comparada con la tecnología de enlace lateral de fusible para dispositivos lógicos programables bipolares.

5.1 FUSIBLE LATERAL

El "PLD" convencional está basado en un arreglo de celdas de dos dimensiones. Una célula consiste de un diodo, o un seguidor emisor, en serie con una angosta película de nicrome, titanio, tugsteno, polisilicón o silicio platino electricamente conductor. Cada material tiene ventajas y desventajas, sin embargo, todos tienen la característica de que el fusible descanza en la superficie del silicón, de ahí que se conozca como fusible lateral. Es por ello que la célula de enlace de fusible tiene un enorme significado en el estado de silicón real.

El área de célda ha incrementado su importancia conforme los adelantos de tecnología de proceso han reducido notablemente el tamaño de todos los componentes del circuito. Previamente, se lograron reducciones en el área de célda a través de adelantos en técnicas fotolitográficas, pero estas técnicas han llegado a su límite. La manufactura, los costos, el espacio y el desarrollo están relacionados directamente con el área de celda y el tamaño del "die". Esta tendencia continua a reducir el tamaño de la celda ha puesto una severa resistencia en la capacidad de manufactura.

La tecnología de unión de fusible lateral no se permite a sí misma completar las pruebas de la programación y una característica de AC del producto. La programabilidad de los fusibles debe ser deducida por la programación de fusibles "sacrificial" (los cuales no son parte del arreglo funcional) y por la funcionalidad completa y la prueba de AC no es posible porque todas las uniones laterales de fusibles son conectadas en estado virgen. Las limitantes de esta tecnología de fusible se han vuelto obvias conforme el manejo de un producto de cada vez mayor calidad, se vaya intensificando. Se requiere una tecnología de fusible que direccionen estas tipos de manufactura , de desarrollo y calidad para realizar el nivel cuántico hacia la siguiente generación de dispositivos lógicos programables bipolares.

5.2 FUSIBLE VERTICAL - LA SIGUIENTE GENERACION

La tecnología de fusible vertical de "AIM" ("Avalanche Induced Migration") está surgiendo como la solución a problemas inherentes por enlaces fusibles mediante pequeñas películas. Un simple transistor con una base flotante forma la célula de memoria completa. El fusible se desarrolla con el silicón en la juntura base-emisor. Ya que se trata de un transistor vertical y sencillamente integrado (de ahí el nombre de fusible vertical), cada transistor ocupa un mínimo de estado real de silicón. Formado con un proceso bipolar aislado mediante oxidación, el tamaño actual de la célda es menor a un tercio de una película delgada de célda convencional.

El fusible vertical está debajo de la superficie del "die" predominantemente. Solamente el área de contacto de la superficie del emisor de alrededor de 2 micras cuadradas, está cubierta por la interconexión de aluminio de la línea de producto. Por comparación, la unión del fusible reside en la superficie del "die" y mediante alta magnificación se muestra el estado de cada unión de fusibles. Ya que, también, el fusible vertical es menor que un tercio del tamaño de la unión del fusible y la juntura actual en corto ocurre debajo de la superficie, es virtualmente imposible observar el estado de las celdas incrementando así la seguridad de la información programada. Los tamaños de célda más pequeños del fusible vertical también se transforman en dispositivos con ambas características, más pequeños y de mayor desarrollo.

5.3 CIRCUITOS EQUIVALENTES

Unión de Fusible y Fusible Vertical

Una celda de unión de fusible lateral conecta una línea de entrada a una línea de maxi-término. En el estado de no-programado existe una conexión de baja impedancia. Después de programar, se ve un circuito abierto donde no se desea conexión entre la línea de entrada y la línea del maxi-término. Contrariamente, un fusible vertical está compuesto por un transistor base-abierta conectando una línea de entrada a una línea de maxi-término. Esta representación, también, puede ser mostrada como dos diodos unidos de forma inversa uno al otro. Cuando no ha sido programado, existe una alta impedancia entre la línea de entrada y la línea del maxi-término. En otras palabras, la línea de entrada no está logicamente

conectada a la línea delo maxi-término. Después de programar la conexión de baja impedancia se observa donde se desea conexión entre la línea de entrada y el maxi-término.

La tecnología del proceso de unión de fusible lateral no está tan integrada como la tecnología de fusible vertical. La consistencia de su programación es una función de muchas variables adicionales como una composición de películas, del espesor y dimensiones de una película, y de las cualidades aisladoras del material que rodea al fusible. La célda vértical es integrada en la tecnología del proceso estándar del silicón, y las caracterísitcas de su programación es función de menos variables.

Una diferencia importante entre ambas tecnologías se explica a continuación. Para los dispositivos "PAL" de unión fusible. un fusible no-programado establece una conexión entre la línea de entrada y el maxi-término. Programándolo se remueve esta conexión.

En los dispositivos de fusible vertical una celda de fusible vertical establece la conexión entre la línea de entrada y el maxi-término.

5.4 MECANISMO DE "FUSING" DE CELDA VERTICAL

La celda vertical cuando el dispositivo no ha sido programado puede ser representado como diodos unidos de forma invertida, o como un transistor con la "lead" de la base abierta.

Para programarlo se forza una corriente controlada hacia el emisor del corte ("breakdown) de la avalancha de la inducción del transistor de la juntura emisor-base. El calor, generado localmente por el voltaje y la corriente de corte, causa que la interfaz aluminio - silicón alcance la temperatura eutéctica (Al-Si "eutectic Soldius) aproximadamente 575C.

La conductividad eléctrica del aluminio eutéctica se difunde, entonces hacia el emisor y a través de la juntura emisor-base, formando un corto permanente, conectando la línea de entrada con la línea del maxi-término apropiado. La disipación de energía causa que el aluminio se forme en aleacion con el silicón en el instante en que se acorta la juntura emisor-base.

5.5 CONFIABILIDAD Y CALIDAD

Ventajas del Fusible Vertical

El fusible vertical es un transistor NPN base abierta, creado dentro de la superficie del silicón por los mismos procedimientos de manufactura utilizados para crear los otros elementos activos. Por ello, el fusible vertical programado no muestra ninguno de los problemas clásicos de la post-programación asociados con varias tecnologías de fusible lateral, apertura inadecuada de los fusibles, sobreapertura de los mismos, etc.

La tecnología de fusible vertical permite probar tecnologías las cuales caracterizan cada uno de los elementos programables antes de embarcarlos. Esto se logra forzando las corrientes de

magnitudes significativamente menores que las necesarias para programar, y midiendo las características de corriente y voltaje de cada elemento que sea un arreglo programable. Debido a esta modalidad de prueba, el fusible vertical ofrece los rendimientos de programación más altos en la industria.

El desarrollo en AC de los dispositivos lógicos programables pueden ser probados, también, a un grado mucho mayor, utilizando el fusible vertical, ya que el estado inicial del elemento fusible es "abierto" permitiendo que el arreglo sea objeto de pruebas en AC, lo que no es posible utilizando la tecnología de unión de fusible lateral.

5.6 VENTAJAS DEL FUSIBLE VERTICAL

+ Mayor al 99.9% de rendimiento programable
+ 100% pruebas paramétricas y funcionales del 100% AC en dispositivos vírgenes eliminan fallas futuras en tarjetas.
+ Tamaño de celda menor = = tamaño de "die" menor = = dispositivos más rápidos
+ Utiliza equipo y "software" de programación estándar en la industria.
+ Alto nivel de seguridad de información
+ Mayor confiabilidad

CONCLUSIONES

CONCLUSIONES

La electronica ha evolucionado cada día con mayor velocidad. Hoy en día el grado de integración de semiconductores es altísimo. Dentro de estos, se tienen disponibles los Dispositivos Lógicos Programables, cuya versatilidad los hace accesibles a los requerimientos del México de hoy. Incluidos en esta familia de dispositivos, encontramos a los PAL's (Arreglos Lógicos Programables, por sus siglas en inglés), entre cuyas ventajas principales que ofrecen tanto a los ingenieros de diseño como a los usarios finales, podemos mencionar :

- Flexibilidad en el diseño.
- Rapidez en el diseño.
- Menor espacio en la tarjeta.
- Menor número de terminales por soldar.
- Mejor y más sencillo control de calidad.
- Mayor confiabilidad.
- Mayor confidencialidad.

Todas estas ventajas convergen a:

+ Menor costo

através del uso de una nueva tecnología.

Este resultado se puede apreciar más claramente en los resultados obtenidos de la práctica desarrollada en el capítulo IV.

Es posible observar que la diferencia en la operacion de soldar y su consecuente control de calidad es significativa entre un diseño y otro.

Enseguida se muestra un cuadro comparativo entre el diseño original y el obtenido a partir del diseño con "PAL's", en el que se tomaron en cuentra varios factores que resultan de interés.

	DISEÑO ORIGINAL		DISEÑO CON PAL'S
No. de Codigos	7	promedio	1
No. de Circuitos Integrados	9	promedio	3
No. de Terminales po IC.	14	promedio	20
Total de Terminales	126		60
Costo Unitario de IC's	0.18 USD		0.50 USD
Costo Total de IC's	1.62 USD		1.50 USD
Ahorro en costo directo de semiconductores		0.12 USD	
Area por IC *	1.08 cm^2		1.74cm^2
Area Total	9.72cm^2		5.22cm^2
Ahorro en el Area		4.5cm^2	

* Sin considerar el area de seguridad entre codigos

Tomando en cuenta que el costo por cm^2 del circuito impreso es de 2.25 US dollar, se tiene:

Ahorro en semiconductores = 0.12 USD
Ahorro en circuito impreso = 10.13 USD

AHORRO TOTAL = 10.25 USD
por cada tarjeta
implementada.

Las caracteristicas anteriormente presentadas son solo las que afectan directamente el costo de un diseño. Sin embargo, es importante considerar tambien, los costos de inventario por codigo, el seguimiento por cada orden de compra, ademas del costo por tiempo de desarrollo.

Este tiempo de desarrollo puede optimizarse si el diseño se lleva a cabo siguiendo los pasos que a continuación se presentan :

Definir el problema
Seleccionar el dispositivo
Escribir las ecuaciones lógicas
Programar el dispositivo
Probar el dispositivo

Estos lineamientos pueden ser aplicados tambien a los dispositivos GALs (Generic Array Logic), cuya facilidad de borrar y reprogramar agregadas a las ventajas de de los dispositivos PALs, los hace un campo interesante en el area de desarrollo.

Con todo lo mencionado en este trabajo, se pretende mostrar una tecnología de vanguardia en la electrónica que es aplicable a las necesidades y accesible a los recursos de un país en pleno desarrollo como lo es el nuestro: MEXICO.

GLOSARIO

AIM .- Migración Inducida por Avalancha (Por sus siglas en inglés). Es una solución a problemas inherentes por enlaces de fusibles mediante pequeñas películas. Se trata de un transistor con una base flotante que forma la célula de memoria completa. Se basa en una juntura de silicón base-emisor.

AND.- Operación de lógica booleana.

BIT.- Dígito binario. Mínima unidad en lógica booleana

BUFFER.- Area almacenamiento temporal, también se utiliza como un dispositivo de refuerzo de corriente en etapas de salida de circuitos electrónicos.

CHIP CARRIER.- Encapsulado del circuito integrado.

CLOCK.- Señal que sincroniza la operación y demás señales del circuito.

CMOS.- Tecnología de fabricación de semiconductores con lógica metal-oxido complementaria.

DIE.- Pastilla de silicio con la que se construye el circuito integrado.

DIP.- Empaquetado de semiconductores que tiene dos líneas de pins en cada lado del circuito.

ECL.- Tecnología de fabricación de semiconductores con lógica de emisor acoplado.

EUTECTIC SOLiDUS.- Es un sólido en estado eutéctito. Estado de Cristalización (Metalurgia)

FLIP-FLOP.- Circuito característico de lógica secuencial cuya función primordial es almacenar un bit de información.

FPGA.- Arreglo genérico de campo programable

FPLA.- Arreglo lógico programable de campo.

FUSING.- Fundimiento de fusibles.

GAL.- Arreglo lógico genérico.

HARDWARE.- Elementos físicos en todo circuito de cómputo.

JEDEC.- Estandar para mapas de fusibles.

LAY-OUT.- Estructura general o arquitectura.

LEAD.- Terminal de salida de los circuitos integrados.

LSI.- Tecnolgía de fabricación de semiconductores de alta escala de integración.-

OFF-THE-SHELF.- Dispositivos que se encuentran fuera del ensamble.

OR.- Operación de lógica booleana.

PAL.- Arreglo lógico programable.
PLAN.- Programa para PC para diseño de dispositivos lógicos programables de National Semiconductor Corp.
PLCC.- Empaquetamiento de semiconductores sin pins de salida, únicamente terminales en los cuatro lados del circuito integrado.
PLD.- Dispositivo lógico programable.
PROM.-Memoria de sólo lectura programable.
PROTO-BOARD.- Tarjeta de pruebas utilizada para ensamblar prototipos.
RESET.- Establecer un estado lógico "0". Se usa como término para poner en condiciones iniciales al circuito.
SET.- Establecer un estado lógico "1".
SOFTWARE.- Programas.
TTL.- Lógica Transistor -Transitor (Por sus siglas en inglés)
XOR.- Abreviatura de transistor.

APENDICE I

MAPAS JEDEC

APENDICE I

Mapa Jedec PAL16L8 @ Bloque 1

```
PLAN v3.14  02-16-1990  19:42 Source filename: TESISA3.BEQ  Device: PAL16L8 *
QP20*  QF2048*  F0*  L0000  11111111111111111111111111111111
11110111111111111111111111111111* L0256 11111111111111111111111111111111
11111111101111111111111111111111* L0512 11111111111111111111111111111111
11111111011111111111111111111111* L0768 11111111111111111111111111111111
01111111111111111111111111111111* L1024 11111111111111111111111111111111
11011111111111111111111111111111* L1792 11111111111111111111111111111111
11111111111011111111111111111111* C2FA8* 0000
```

Mapa Jedec PAL16L8 @ Bloque 2

```
PLAN v3.14  02-16-1990  19:43 Source filename: tesisb3.BEQ  Device: PAL16L8 *QP20*
QF2048*  F0*  L0000  11111111111111111111111111111111
11011111111111111111111111111111* L0512 11111111111111111111111111111111
11111110111111111111111111111111* L0768 11111111111111111111111111111111
01111111111111111111111111111111* L1792 11111111111111111111111111111111
11110111111111111111111111111111* C1FCA* 0000
```

Mapa Jedec PAL16L8 @ Bloque 3

```
PLAN v3.14  02-14-1990  20:59 Source filename: TESISPA7.BEQ  Device: PAL16L8 *
QP20*  QF2048*  F0*  L0000  11111111111111111111111111111111
01010111011111110111011011110111* L1024 11111111111111111111111111111111
01010111011111111111111111111111* L1280 11111111111111111111111111111111
11111111111011111111111111111111 11111111111111101111111111111111
11111111111111111110111111111111 11111111111111111111111011111111
11111111111111111111111111111011 11111111111111111011111111111111* L1792
11111111111111111111111111111111 11111111111111111111111111111101* C32E2* 0000
```

APENDICE II

DIAGRAMAS

DIAGRAMA II.1

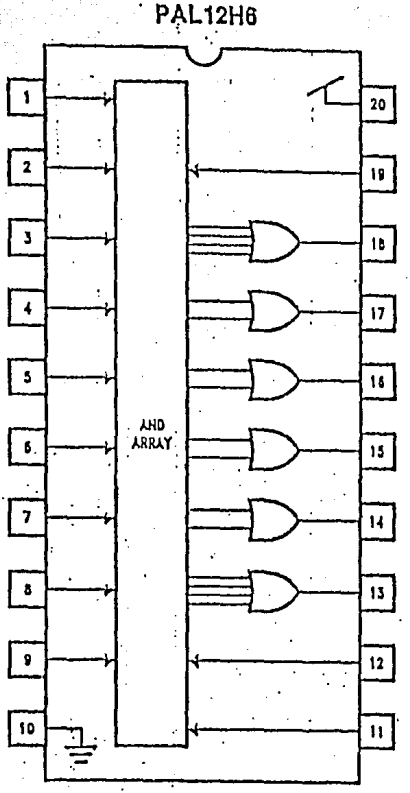
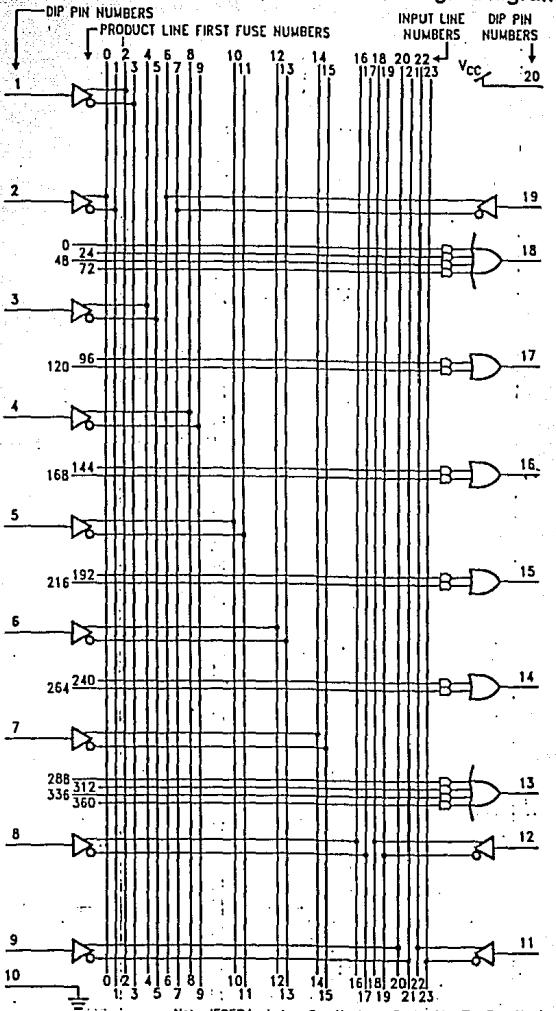


DIAGRAMA II.2 Logic Diagram PAL12H6



Note: JEDEC Logic Array Fuse Number = Product Line First Fuse Number + Input Line Number.

DIAGRAMA II.3

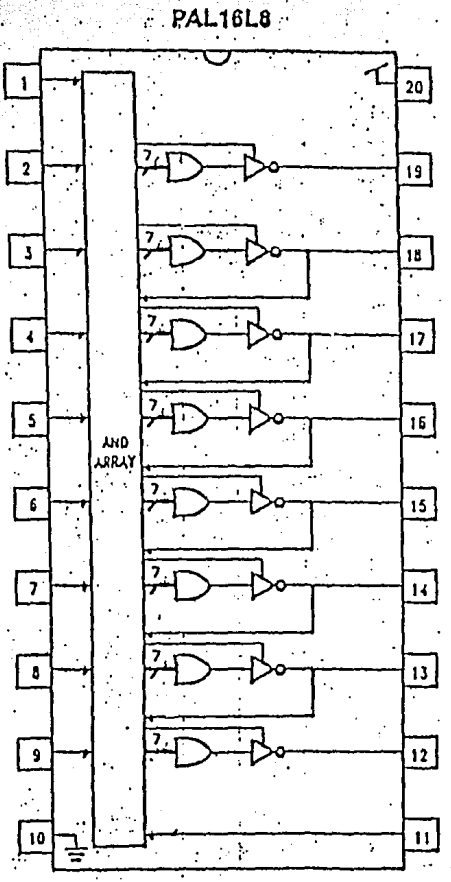


DIAGRAMA II.4 Logic Diagram—PAL16L8

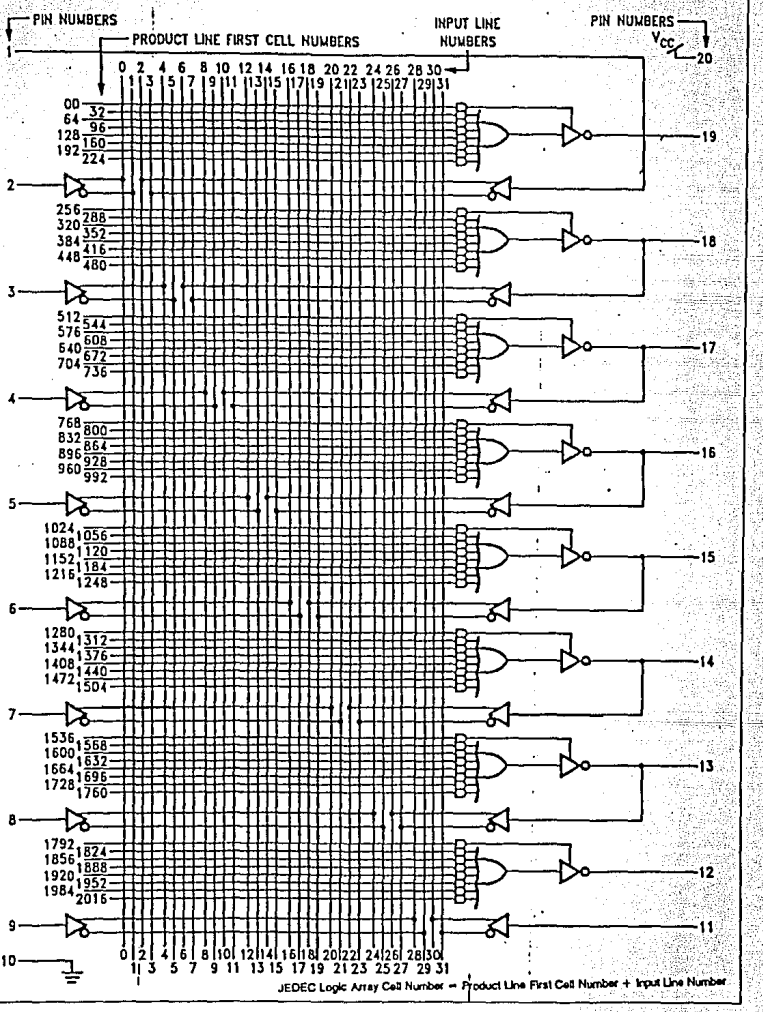


DIAGRAMA II.5

PAL16R8

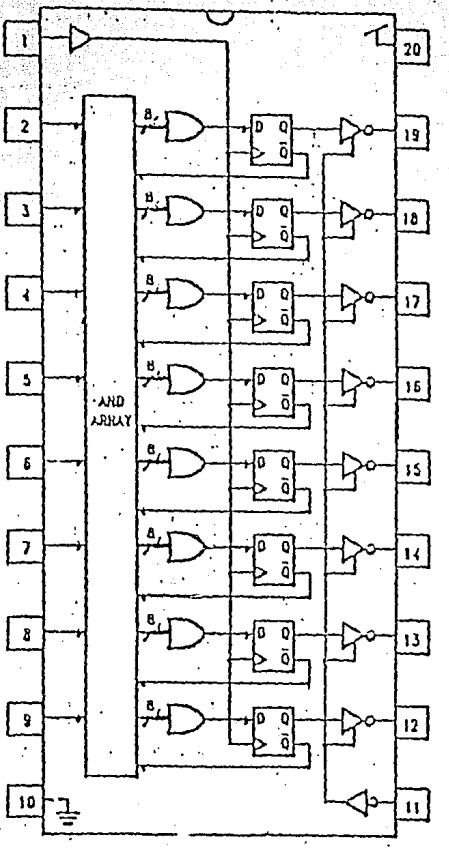
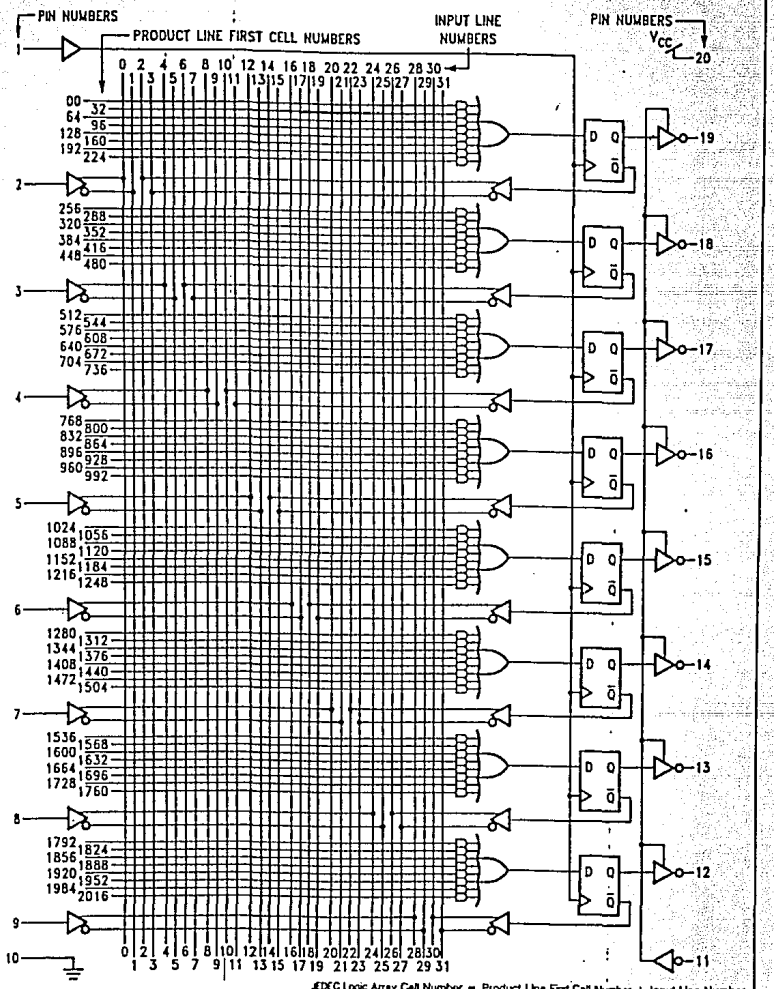


DIAGRAMA II.6 Logic Diagram—PAL16R8



BIBLIOGRAFIA

+ PROGRAMMMABLE LOGIC DEVICES
DATABOOK AND DESIGN GUIDE
NATIONAL SEMICONDUCTOR
EDICION 1989.

+ IBM TECHNICAL REFERENCE
OPTIONS AND ADAPTERS
VOLUMEN I
EDICION 1984

+ REVISTA BYTE
VOLUMEN 12
NO. 1
EDICION 1987

+ INTERFACE BIPOLAR LSI
BIPOLAR MEMORY
PROGRAMMABLE LOGIC
NATIONAL SEMICONDUCTOR
EDICION 1983

+ PROGRAMMABLE LOGIC DEVICE
ALTHERA
EDICION

+ PROGRAMA "PLAN"
"PROGRAMMABLE LOGIC ANALISYS BY NATIONAL"
VERSION 3.14
NATIONAL SEMICONDUCTOR

+ PROGRAMA DEL
PROGRAMADOR DATA I/O

+ "VERTICAL FUSE TECHNOLOGY FOR PROGRAMMABLE LOGIC DEVICES"
APPLICATION NOTE
K.C. SHEKAR
NATIONAL SEMICONDUCTOR
EDICION 1988

+ "IC'S MADE SIMPLE"
NATIONAL SEMICONDUCTOR