UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

CONTROL DE UN BRAZO ROBÓTICO UTILIZANDO UN FPGA

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO MECÁNICO ELECTRICISTA
ÁREA ELÉCTRICA - ELECTRÓNICA

P R E S E N T A N :

RODOLFO AVIÑA MORALES
JOEL ESQUIVEL VILLAR
EDUARDO ROJANO GARCÍA

DIRECTORA DE TESIS
M. I. NORMA ELVA CHÁVEZ RODRÍGUEZ

Ciudad Universitaria Noviembre del 2005

Agradecimientos

A Guadalupe Villar (q.e.p.d.),
ejemplo indeleble
de amor, trabajo y sacrificio.

A mi esposa Teresa
y a nuestros hijos
Eduardo y Susana,
con amor y gratitud eternos.

A mis hermanos todos,
con cariño, gratitud y admiración.

Agradecimientos

A mis padres, por su confianza y ejemplar educación.

A mi querida Silvia, por su entrega tenaz y comprensión.

A mis hermanos, todos, por su cariño.

A Anita, porque es mi motivo de continuar.

A Esteban, (q.e.p.d.)

Agradecimientos

Con el mayor agradecimiento y todo mi cariño dedico esta tesis a:

A mi padre, Javier Aviña, por mostrarme el camino correcto y el amor a la lectura; dos pilares fundamentales en mi vida

A mi madre, Lilia Morales y Mori, que con su extraordinario creatividad, amor, valor y ejemplo, ha dejado y sigue dejando una profunda huella en todas las personas que tenemos el honor de conocerla, a quien agradezco por enseñarme a encontrar la belleza y felicidad que despierta el hecho de poder descubrir y comprender.

A mi hermana Coppelia, que sin su enorme e incondicional apoyo y sacrificio, no seria posible la conclusión de este paso en mi carrera.

A mi amada esposa Claudia por su apoyo durante todos estos años.

A mi hijo Mauricio, mi mayor fuente de felicidad he inspiración, mi orgullo y motor que me impulsa.

A mis queridas hermanas Laura y Tania que siempre estuvieron a mi lado.

Al señor Arturo Barraza, por su apoyo y tiempo dedicado en la construcción del proyecto del brazo robótico.

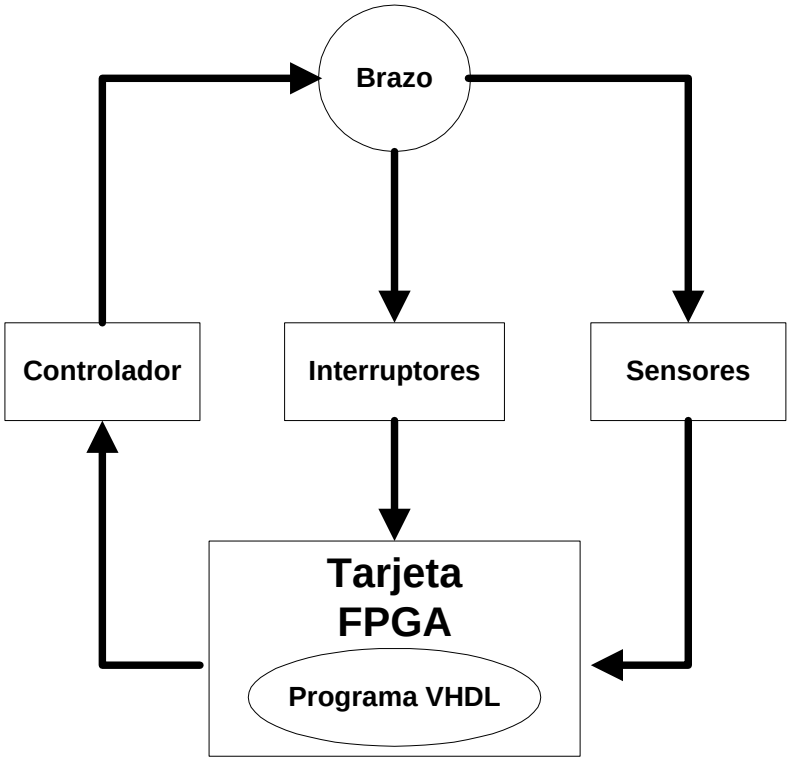
A toda mi familia por sus palabras de aliento.

Indice

Prólogo

Este Proyecto de Tesis muestra la forma en que un FPGA es utilizado para controlar los movimientos de un Brazo Robótico que es capaz de jugar el juego “GATO”.

Este juego se realiza en un tablero con nueve orificios, cada uno de ellos equipado con un par de sensores, estos sensores únicamente envían al FPGA la señal de vacío u ocupado. Por lo que se requiere que el brazo siempre sea el primero en tirar y así almacenar en un lugar las tiradas impares, que siempre serán hechas por el brazo, y las tiradas pares pertenecientes a su oponente. En la siguiente figura se muestra el diagrama de bloques requerido para el control del Brazo Robótico.



Introducción

El objetivo principal de este proyecto radica en mostrar la utilidad del uso de los FPGA dejando el armado del proyecto como prototipo en el laboratorio de DLP de la Facultad de Ingeniería.

Se buscó un tema para el uso de un FPGA y se determinó el uso de los Robots porque cada vez son más importantes por su amplio rango de aplicaciones principalmente para el área de manufactura y cirugía o manejando materiales peligrosos. Este proyecto está centrado en el manejo de un Brazo Robot, donde una de las funciones es moverse a una localidad específica a lo largo de una determinada trayectoria o ruta Hay nueve posibles rutas gracias al algoritmo diseñado especialmente para el juego. Se utiliza un FPGA como control del Brazo Robótico, el cual tiene cinco motores y cinco junturas para la flexibilidad de movimientos. Los movimientos posibles son los siguientes: Base, derecha/izquierda 350 grados; hombro, arriba/abajo 120 grados; codo, arriba/abajo 135 grados; muñeca, derecha/izquierda 340 grados; y pinzas, Abrir/cerrar hasta 50 mm. Las dimensiones; máxima extensión horizontal 360 mm, máxima extensión de altura 510 mm, y máximo peso 130grm.

Los FPGA (Field Programmable Gate Array) son un tipo especial de circuitos ASIC (Applications Specific Integrated Circuits) que pueden ser configurados y reconfigurados por el usuario en lugar del fabricante. Unas de las características importantes de los FPGA es el rápido armado de prototipos, son dispositivos digitales donde pueden implementarse más de miles de compuertas lógicas en días, incluso en horas; además los FPGA son baratos en su elaboración y tienen la propiedad de ser completamente examinables. Los FPGA son típicamente usados para implementar funciones lógicas multi-nivel. Este proyecto muestra cómo un FPGA puede controlar directamente a un Driver lineal conectado a los motores de CD del brazo y además generar una secuencia de comandos.

Para la configuración del FPGA se requiere del uso de un lenguaje de programación para descripción digital, lenguaje VHDL (Very High Speed Integrated Circuit Hardware Description Lenguaje), el lenguaje VHDL que se utilizó para el proyecto, porque viene con la propia tarjeta, es el software de Xilinx con el cual se generó el algoritmo tanto para la lógica del Juego Gato como la lógica del control de movimientos de los motores, incluyendo las entradas de los sensores. Esto quiere decir que aún cuando existen varias formas de controlar un robot y en especial solucionar el juego de Gato, nuestro proyecto se centró específicamente en el algoritmo propio y con comandos sintetizables para la tarjeta FPGA con el software Xilinx.

Esta tesis está organizada de la siguiente manera; primeramente se da una explicación amplia de los temas que abarca el control del Brazo Robótico,

por eso, durante los capítulos del uno al cinco son fundamentos para entender el uso de cada uno de los componentes que se utilizan en el proyecto. En el capítulo uno, su contenido, está constituido como un repaso de cómo y donde los Dispositivos Lógicos Programables son usados, incluimos y centramos nuestra atención a dos de los dispositivos más representativos del diseño digital los cuales son los bloques CPLD y FPGA. En el capitulo dos existe el enfoque de abarcar los procesos de síntesis e implementación para los FPGA, las definiciones y existencias de otros tipos de lenguajes de programación de Hardware, así como algunos comandos principales con el software de lenguaje VHDL de la compañía Xilinx y se indican las características de la tarjeta Spartan-3 utilizada en el proyecto. El capitulo tres es un esbozo de definiciones, así como de los desarrollos y aplicaciones para los diferentes tipos de robots que existen en la actualidad y por consiguiente la explicación de la clasificación en la que se ubica el Brazo Robótico utilizado. El capítulo cuatro muestra la explicación y funcionamiento de los motores de CD, así como el control de éstos utilizando distintos drivers o controladores y el diseño de circuitos de acuerdo a las especificaciones de fábrica de éstos como interfaz entre la etapa de potencia y la lógica de control. En el capitulo cinco se observa la clasificación de los sensores y la amplia aplicación de los mismos dentro de nuestra implementación para lograr obtener elementos de interfaz con el control del Brazo Robótico y a su vez observar que estos dispositivos han sido complementos ideales y los cuales intervienen en forma directa sobre el control lógico

Con las bases mostradas en los capítulos indicados se desarrolla el capítulo 6 donde se explica el proyecto central de la tesis desglosándolo por bloques de acuerdo a los diagramas mostrados, en donde se indica la forma de selección de los distintos dispositivos utilizados tales como sensores, drivers, materiales, etc, y se explican a detalle el diseño de los circuitos armados y se presenta la implementación y explicación completa del control de todos los niveles que intervienen en la realización para todo el conjunto. Para el caso de la configuración de la tarjeta FPGA Spartan-3 se muestra la lógica de funcionamiento del programa y los diagramas de bloques del algoritmo con el cual es suficientemente descriptivo indicando las razones y el uso de las variables involucradas para el proyecto.

Finalmente se muestra el apartado correspondiente a las conclusiones donde se muestra cada uno de los problemas que se encontraron en la construcción del proyecto y los beneficios que se obtuvieron durante el desarrollo y las mejoras que se pueden hacer todavía hacia el control.

Capítulo 1

Dispositivos Lógicos Programables

A mediados de los años setenta se produce una fuerte evolución en los procesos de fabricación de los circuitos integrados, y junto a las tecnologías bipolares, surge la MOS (Metal Oxide Semiconductor), principalmente la NMOS, promoviendo el desarrollo de circuitos digitales hasta la primera mitad de los años ochenta.

En aquellas épocas, el esfuerzo de diseño se concentraba en los niveles eléctricos para establecer características e interconexiones entre los componentes básicos a nivel de transistor. El proceso de diseño era altamente manual y tan solo se empleaban herramientas como el PSPICE para simular esquemas eléctricos con modelos previamente personalizados a las distintas tecnologías.

A medida que pasaban los años, los procesos tecnológicos se hacían más y más complejos. Los problemas de integración iban en aumento y los diseños eran cada vez más difíciles de depurar y de dar mantenimiento. Inicialmente los circuitos MSI (Medium Scale Integration) y LSI (Low Scale Integration) se diseñaron mediante la realización de prototipos basados en módulos más sencillos. Cada uno de estos módulos estaba formado por compuertas ya probadas, este método poco a poco iba quedándose obsoleto. En ese momento (finales de los años setenta) se constata el enorme desfase que existe entre tecnología y diseño.

La considerable complejidad de los chips que se pueden fabricar, implica unos riesgos y costos de diseño desmesurados e imposibles de asumir por las empresas. Es entonces, cuando diversos grupos de investigadores empiezan a crear y desarrollar los llamados "lenguajes de descripción de hardware" cada uno con sus peculiaridades. Empresas tales como IBM con su IDL, el TI - HDL de Texas Instruments, ZEUS de General Electric, etc., así como los primeros prototipos empleados en las universidades, empezaron a desarrollarse buscando una solución a los problemas que presentaba el diseño de los sistemas complejos.

Sin embargo, estos lenguajes nunca alcanzaron el nivel de difusión y consolidación necesario por motivos distintos. Alrededor de 1981 el Departamento de Defensa de los Estados Unidos desarrolla un proyecto llamado VHSIC (Very High Speed Integrated Circuit) su objetivo era rentabilizar las inversiones en hardware haciendo más sencillo su mantenimiento. Se pretendía con ello resolver el problema de modificar el hardware diseñado en un proyecto para utilizarlo en otro, lo que no era posible hasta entonces porque no existía una herramienta adecuada que armonizase y normalizase dicha tarea, era el momento de los HDLs

En 1983, IBM, Intermetrics y Texas Instruments empezaron a trabajar en el desarrollo de un lenguaje de diseño que permitiera la estandarización, facilitando el mantenimiento de los diseños y la depuración de los algoritmos, para ello el IEEE propuso su estándar en 1984.

Tras varias versiones llevadas a cabo con la colaboración de la industria y de las universidades, que constituyeron a posteriori etapas intermedias en el desarrollo del lenguaje, el IEEE publicó en diciembre de 1987 el estándar IEEE std 1076-1987 que constituyó el punto firme de partida de lo que después de cinco años sería ratificado como VHDL.

Esta doble influencia, tanto de la empresa como de la universidad, hizo que el estándar asumido fuera un compromiso intermedio entre los lenguajes que ya habían desarrollado previamente los fabricantes, de manera que éste quedó como ensamblado y por consiguiente un tanto limitado en su facilidad de utilización haciendo dificultosa su total comprensión. Este hecho se ha visto incluso ahondado en su revisión de 1993, para actualizarlo y crear estándar adicional, el IEEE-1164, fue adoptado. Para 1996 el estándar IEEE-1076.3 se convirtió en un estándar de VHDL para síntesis siendo éste el que se utiliza en el diseño de sistemas digitales.

La independencia en la metodología de diseño, su capacidad descriptiva en múltiples dominios y niveles de abstracción, su versatilidad para la descripción de sistemas complejos, su posibilidad de reutilización y en definitiva la independencia de que goza con respecto de los fabricantes, han hecho que VHDL se convierta con el paso del tiempo en el lenguaje de descripción de hardware por excelencia.

1.1 Clasificación de Dispositivos Lógicos Programables (PLD)

Un dispositivo lógico programable, o PLD (Programmable Logic Device), es un dispositivo cuyas características pueden ser modificadas y almacenadas mediante programación. El principio de síntesis de cualquier dispositivo lógico programable se fundamenta en el hecho de que cualquier función booleana puede ser expresada como una suma de productos. Una matriz de conexiones es una red de conductores distribuidos en filas y columnas con un fusible en cada punto de intersección.

La mayoría de los PLD están formados por una matriz de conexiones, una matriz de compuertas AND, y una matriz de compuertas OR y algunos, además, con registros. Con estos recursos se implementan las funciones lógicas deseadas mediante un software especial y un programador. Las matrices pueden ser fijas o programables. El tipo más sencillo de matriz programable, que data de los años 60, era una matriz de diodos con un fusible en cada punto de intersección de la misma. En la figura 1.1, se muestran los circuitos básicos para la mayoría de los PLD.

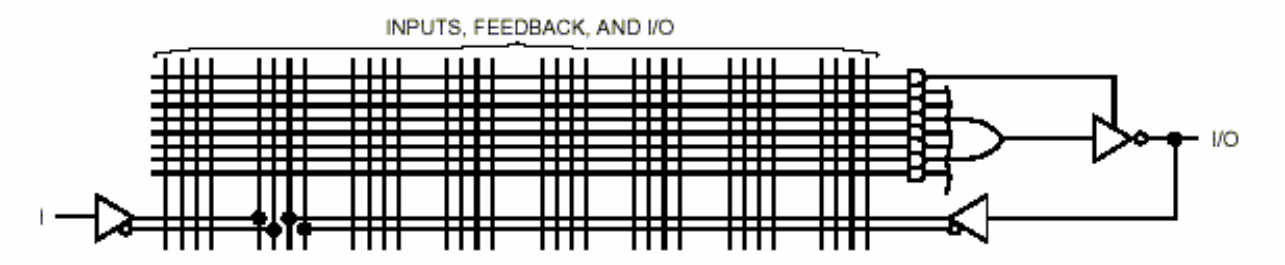


Figura. 1.1 Estructuras básicas de un PLD

Una forma rápida y directa de integrar aplicaciones se logra con la lógica programable, la cual permite independizar el proceso de fabricación del proceso de diseño fuera de la fábrica de semiconductores. El sistema desplaza los errores de alambrado al campo exclusivo de la programación. Los sistemas con estas características se pueden borrar y reprogramar en casos de cambios o revisiones. El resultado es la reducción del espacio físico de la aplicación. El diseño está basado en bibliotecas y mecanismos específicos de mapeado de funciones.

La lógica programable, como el nombre implica, es una familia de componentes que contienen conjuntos de elementos lógicos (AND, OR, NOT, LATCH, FLIP-FLOP) que pueden configurarse en cualquier función lógica que el usuario desee y que el componente soporte. Hay varias clases de dispositivos lógicos programables: ASIC, FPGA, PAL ó PLA, PROM, GAL, y CPLD complejos.

1.1.1 Circuitos Integrados de Aplicación Específica (ASIC)

ASIC significa Circuitos Integrados de Aplicación Específica y son dispositivos definibles por el usuario. Los ASIC, al contrario que otros dispositivos, pueden contener funciones analógicas, digitales, y combinaciones de ambas. En general, son programables mediante máscara y no programables por el usuario. Esto significa que los fabricantes configurarán el dispositivo según las especificaciones del usuario. Se usan para combinar una gran cantidad de funciones lógicas en un dispositivo. Sin embargo, estos dispositivos tienen un costo inicial alto, por lo tanto se usan principalmente cuando es necesaria una gran cantidad.

1.1.2 Arreglos de Compuertas de Campo Programables (FPGA)

La arquitectura de un FPGA (Field Programmable Gate Array) consiste en arreglos de varias celdas lógicas las cuales se comunican unas con otras mediante canales de conexiones verticales y horizontales. Una arquitectura típica se muestra en la figura 1.1.2.

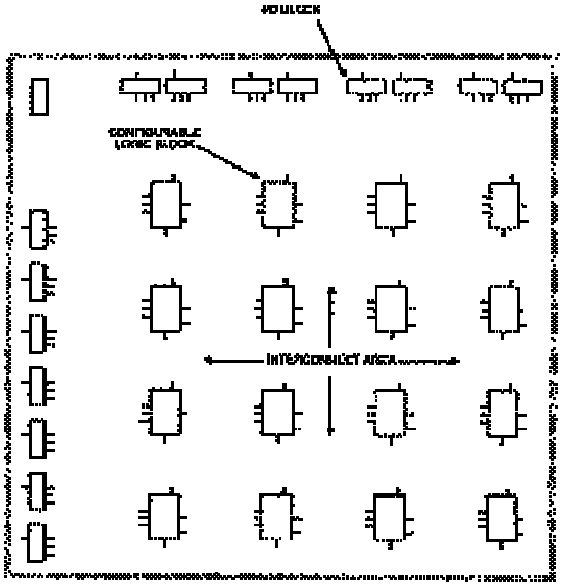


Figura 1.1.2. Representación esquemática de un FPGA.

Cada celda lógica es similar a los bloques lógicos de un CPLD. La estructura de las celdas lógicas y las formas en que estas pueden ser interconectadas, tanto salidas como entradas de la celda, varían de acuerdo al fabricante. En general una celda lógica tiene menos funcionalidad que la combinación de sumas de productos y macroceldas de un CPLD, pero como cada FPGA tienen una gran cantidad de celdas lógicas es posible implementar grandes funciones utilizando varias celdas lógicas en cascada. Además de las celdas lógicas también es importante la tecnología utilizada para crear las conexiones entre los canales, las más importantes son las siguientes:

1.1.2.1 Características de los FPGA

Antifusible (ANTIFUSE)

Son programables una sola vez y utilizan algo similar a un fusible para realizar las conexiones, una vez que es programado éste ya no se puede recuperar. Al contrario que un fusible normal, estos anti - fusibles cuando son programados producen una conexión entre ellos por lo que normalmente se encuentran abiertos. La desventaja es que no son reutilizables pero por el contrario disminuyen considerablemente el tamaño y costo de los dispositivos.

Memoria Estática de Acceso Aleatorio SRAM

Celdas SRAM son implementadas como generadores de funciones para simular lógica combinacional y, además, son usadas para controlar multiplexores e interconectar las celdas lógicas entre si.

La primera FPGA la introdujo Xilinx en el año 1985. La programación de las FPGA de Xilinx basadas en RAM estática es diferente a la programación de los PLD. Cada vez que se aplica la tensión de alimentación, se reprograma con la información que lee desde una PROM de configuración externa a la FPGA. Una FPGA basada en SRAM (RAM estática) admite un número ilimitado de reprogramaciones sin necesidad de borrados previos.

La tecnología SRAM y ANTIFUSE son comúnmente utilizadas por la mayoría de los fabricantes. La tecnología SRAM es utilizada por Altera, Lucent Technologies, Atmel, Xilinx y otros. La tecnología ANTIFUSE es utilizada por Cypress, Actel, QuickLogic, y Xilinx.

Celdas lógicas

La estructura de las celdas lógicas se ve fuertemente influida por la tecnología utilizada para fabricar el FPGA. Un FPGA que tiene una gran cantidad de canales de interconexión tiende a tener pequeñas celdas lógicas con muchas entradas y salidas en comparación con el número de compuertas que tiene la celda, este tipo de FPGA generalmente utilizan tecnología ANTIFUSE.

Un FPGA que tiene una estructura pequeña en canales de interconexión tiende a tener grandes celdas lógicas con pocas entradas y salidas en comparación con el número de compuertas que hay en la celda. Este tipo de FPGA generalmente está hecho con tecnología SRAM.

Una arquitectura con celdas lógicas pequeñas nos permite utilizar totalmente los recursos del dispositivo. Sin embargo, si las celdas lógicas son demasiado pequeñas entonces sucede que tendremos que utilizar un gran numero de estas en cascada para poder implementar funciones booleanas grandes, lo cual afecta porque cada celda lógica en cascada agrega un tiempo de retardo en la función implementada.

Cuando el tamaño de la celda lógica es grande sucede lo contrario. En este tipo de celdas lógicas es posible utilizar un gran número de compuertas por lo que podemos implementar funciones booleanas de varios términos con pocas celdas lógicas. El que el tamaño de la celda sea grande no afecta la frecuencia máxima de trabajo porque estamos hablando de que existe un gran numero de compuertas que pueden ser usadas en la función al mismo tiempo, siendo el mismo tiempo de retardo para todas. En cambio cuando la celda lógica tiene pocas compuertas es necesario utilizar las compuertas de otra celda para poder implementar la misma función y se acumula el tiempo de retardo de las compuertas de la otra celda. Sin embargo, cuando las funciones son pequeñas en comparación con el tamaño de la celda no es necesario utilizar todas las compuertas de la celda, por lo que este tipo de celdas no son precisamente las más indicadas para desempeñar pequeñas funciones.

Bloques lógicos

Las FPGA contienen bloques lógicos relativamente independientes entre sí, con una complejidad similar a un PLD de tamaño medio. Estos bloques lógicos pueden interconectarse, mediante conexiones programables, para formar circuitos mayores. Existen FPGA que utilizan pocos bloques grandes (Pluslogic, Altera y AMD) y otras que utilizan muchos bloques pequeños (Xilinx, AT&T, Plessey, Actel), para la situación de aplicación que presentamos dentro de esta tesis empleamos los bloques lógicos de Xilinx.

A diferencia de los PLD, no utilizan arquitectura de matriz de compuertas AND seguida de la matriz de compuertas OR y necesitan un proceso adicional de enrutamiento (ruteado) del que se encarga un software especializado.

En general la complejidad de una FPGA es muy superior a la de un PLD. Los PLD tienen entre 100 y 2000 compuertas, las FPGA tienen desde 1200 a 20.000 compuertas y la tendencia es hacia un rápido incremento en la densidad de compuertas. El número de flip-flops de las FPGA generalmente supera al de los PLD. Sin embargo, la capacidad de la FPGA para realizar lógica con las entradas suele ser inferior a la de los PLD. Por ello: "los diseños que precisan lógica realizada con muchas patillas de entrada y con pocos flip-flops, pueden realizarse fácilmente en unos pocos PLD, mientras que en los diseños en los que intervienen muchos registros y no se necesita generar combinaciones con un elevado número de entradas, las FPGA pueden ser la solución óptima".

1.1.3 Arreglos Lógicos Programables (PLA o PAL)

Las arquitecturas generales pueden variar pero normalmente consisten en una o más matrices de compuertas AND y OR para implementar funciones lógicas y entre ellas podemos mencionar a las PLA las cuales son matrices lógicas programables. Estos dispositivos contienen ambos términos AND y OR programables lo que permite a cualquier término AND alimentar cualquier término OR. Las PLA probablemente tienen la mayor flexibilidad frente a otros dispositivos con respecto a la lógica funcional. Normalmente poseen realimentación desde la matriz OR hacia la matriz AND que puede usarse para implementar máquinas de estado asíncronas. La mayoría de las máquinas de estado, sin embargo, se implementan como máquinas sincronas. Con esta perspectiva, los fabricantes crearon un tipo de PLA denominado Secuencial (Sequencer) que posee registros de realimentación desde la salida de la matriz OR hacia la matriz AND. (Ver figura 1.1.3)

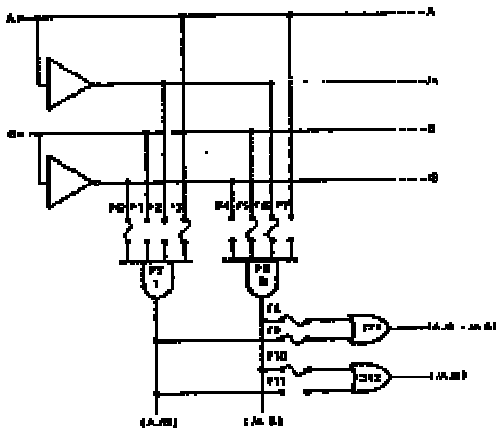


Figura 1.1.3 Representación esquemática de una PLA

Los Arreglos Lógicos Programables (PLA) son un PLD formado por una matriz AND programable y una matriz OR programable. La PLA ha sido desarrollada para superar algunas de las limitaciones de las memorias PROM. (Ver figura 1.1.4)

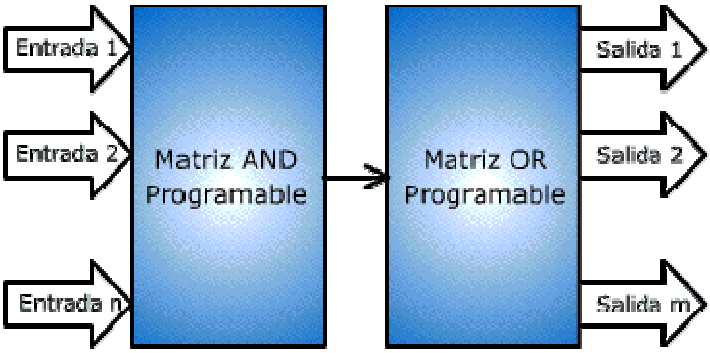


Figura 1.1.4 Diagrama de bloques de una PLA

Un PLA está constituido básicamente por dos submatrices o planos denominados plano AND y OR, respectivamente. Ambos planos están separados entre sí por una pequeña zona divisoria denominada zona de conexión. Tanto el plano AND como el plano OR disponen, a su vez, de dos zonas externas denominadas buffers o separadores de entrada y de salida. Las señales de entrada del PLA ($x_1, \ldots, x_m$) llegan a los buffers de entrada del plano AND y producen las señales invertidas ($\bar{x}_1, \ldots, \bar{x}_m$).Ambos tipos de señales (x_i y $\bar{x}_i$) penetran verticalmente en el plano AND y generan los términos producto pi. Estos últimos discurren horizontalmente por ambos planos, atravesando previamente la zona de conexión, y producen finalmente las salidas del PLA mediante la realización de sumas lógicas entre los términos productos anteriores.

Además de las zonas mencionadas, existen otras dos regiones especiales. Una de ellas está situada a la izquierda del plano AND y la otra en la parte superior del plano OR. Estas regiones están constituidas por transistores del "pull-

up", que actúan como resistencia de carga, a través de los cuales se alimentan las líneas de los términos producto y las líneas de salida del PLA respectivamente. En la figura 1.1.5 se muestra un esquema global de su estructura:

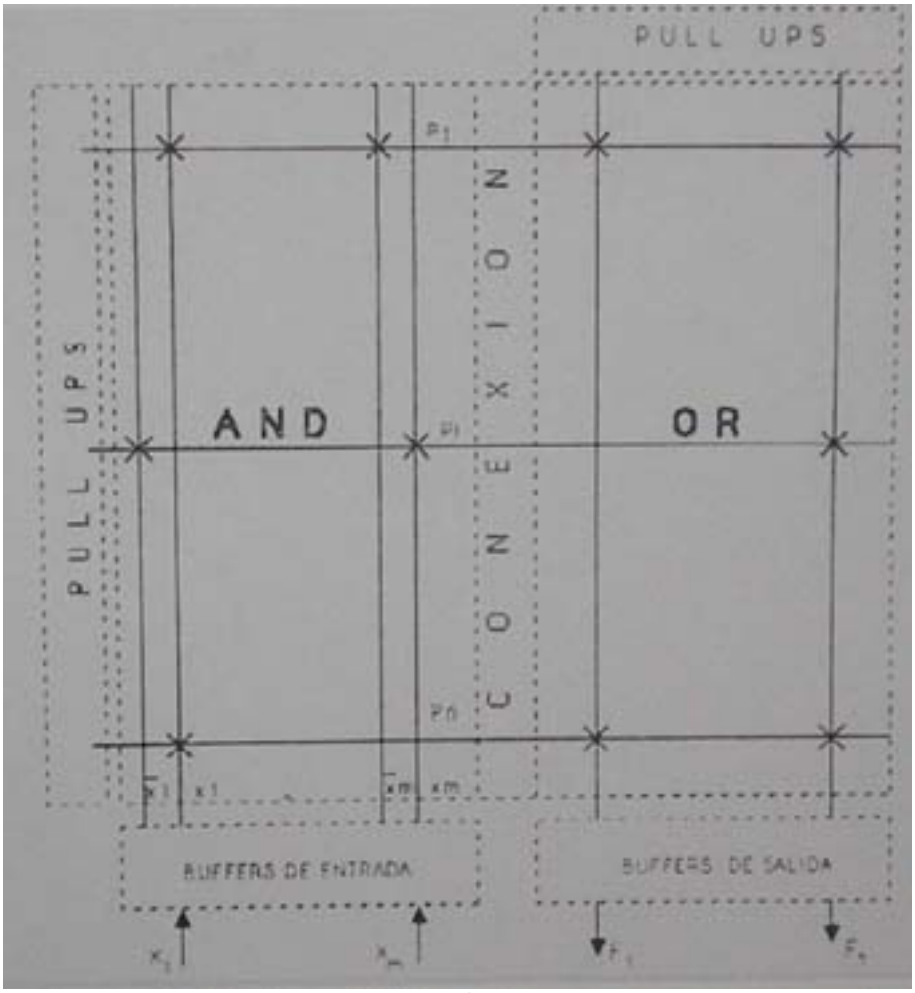


Figura 1.1.5 Estructura básica de un PLA

La realización física de un PLA se lleva a cabo mediante la conexión de cada una de las celdas que pertenecen a las regiones anteriores (buffers de entrada, plano AND, transistores de pull-up del plano AND, conexión AND-OR, plano OR, transistores de pull-up del plano OR y buffers de salida). De todas estas celdas, sólo las de los planos AND y OR están relacionadas con las funciones lógicas que definen el circuito. El resto hace referencia a otros factores externos ajenos a la lógica propiamente dicha. En algunos casos, estos factores han de ser tenidos en cuenta si se quiere hacer una estimación realista del área final ocupada por el PLA. Esto ocurre, por ejemplo, cuando se aplican técnicas de optimización en las que se modifica el número de entradas y/o salidas. (Técnicas de Partición)

1.1.3.1 Características de los PLA

Representación matricial

Los PLA sirven para representar multifunciones booleanas expresadas mediante dos niveles de compuertas. Sea, pues, una multifunción *F* formada por *s* funciones simples f_i cada una de ellas dependiente de *m* variables distintas $(F = Y, f_i(x_1, K, x_m))$. Supongamos también que es necesario desarrollar *n* productos lógicos distintos con las variables dependientes x_i para expresar todas y cada una de las funciones f_i mediante sumas de productos. Entonces, el PLA asociado a la multifunción *F*, puede representarse por una matriz, C, formada por *n* filas y *m*+*s* columnas. Cada una de estas filas C_i se define del modo siguiente:

"*j* : 1..m (plano AND).
C_{ij} = 0 si x_j está complementada en el término producto C_i.
C_{ij} = 1 si x_j no está complementada en el término producto C_i.
C_{ij} = 2 si x_j no aparece en el término producto C_i.
"*j* : *m* + 1... *m* + *s* (plano OR).
C_{ij} = 3 si C_i no forma parte de la función Fj· *m*
C*ij* = 4 si C_i forma parte de la función Fj· *m*

De la definición anterior se deduce que todos los 2's de la matriz C representan elementos vacíos en las *m* primeras columnas pertenecientes al plano AND o submatriz de entradas. Igualmente ocurre con los 3's en las columnas restantes del plano OR o submatriz de salidas. Por elemento vacío se entiende aquella posición del PLA en la que no existen conexiones.

Así, por ejemplo la multifunción $f_1 = \bar{x}_3 x_6 + x_1 \bar{x}_6$, $f_2 = \bar{x}_2 x_4 + \bar{x}_1 x_5 + x_6$, $f_3 = x_1$ se representa mediante la matriz de cobertura de la tabla 1.1.1

X1	X2	X3	X4	X5	X6	F1	F2	F3
2	2	0	2	2	1	4	3	3
2	0	2	1	2	2	3	4	3
1	2	2	2	2	0	4	3	3
0	2	2	2	1	2	3	4	3
1	2	2	2	2	2	3	3	4
2	2	2	2	2	1	3	4	3

Tabla 1.1.1 Matriz de cobertura.

En muchos casos conviene utilizar otra representación matricial más simple del PLA denominada matriz de personalidad. Esta nueva matriz se define a partir de la matriz de cobertura del modo siguiente:

• " j: = 1...m • B_{ij} = 1 si C_{ij} = 0 ó 1. • B_{ij} = 0 si C_{ij} = 2.	• "j := m + 1... m + s • B_{ij} = 1 si C_{ij} = 4. • B_{ij} = 0 si C_{ij} = 3.

Tabla 1.1.2 Matriz de cobertura

Es decir, un 1 en la *j*-ésima columna e *i*-ésima fila del plano AND indica que la columna *j* es un factor del término producto *i*, mientras que un 1 en la *j*-ésima columna e *i*-ésima fila del plano OR indica que el término producto *i* es un término de la salida *j*-*m*. La tabla 1.1.3 corresponde a la matriz de personalidad asociada a la matriz de cobertura de la tabla 1.1.2.

X1	X2	X3	X4	X5	X6	F1	F2	F3
0	0	1	0	0	1	1	0	0
0	1	0	1	0	0	0	1	0
1	0	0	0	0	1	1	0	0
1	0	0	0	1	0	0	1	0
1	0	0	0	0	0	0	0	1
0	0	0	0	0	1	0	1	0

Tabla 1.1.3 Matriz de personalidad.

La representación de un PLA mediante su matriz de personalidad respectiva nos ayudará a resolver los problemas planteados en la optimización lógica y topológica de PLA.

Diversas formas de estructuras PLA

La estructura de los PLA se muestra en la figura 1.1.6, encontrándose las arquitecturas PLA listadas en la tabla 1.1.1 La estructura de los secuenciadores se encuentran en la figura 1.1.7, las arquitecturas de los secuenciadores están listadas en la tabla 1.1.4.

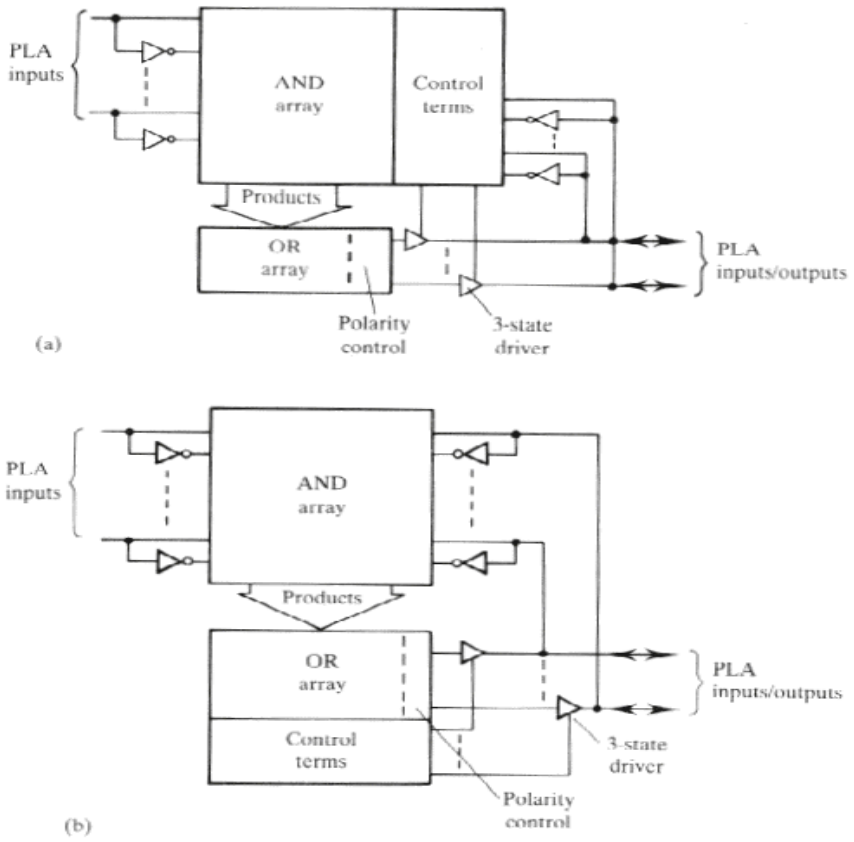


Figura. 1.1.6 (a) PLA con control de salida.
(b) Segunda forma con los términos de control en OR

Nombre	*Nº de entradas*	*Nº de salidas*	*Nº de E/S*	*Maximo nº de entradas*	*Nº del producto*	*Nº de téminos de control*	*Nº de pins*	*Año de revelación*	*Notas*
8575/6	*14*	*8*	*-*	*14*	*96*	*-*	*24*	*1973*	*2,3*
IM5200	*14*	*8*	*-*	*14*	*48*	*-*	*24*	*1975*	
PLS100/1	*16*	*8*	*-*	*16*	*48*	*-*	*28*	*1975*	*2,4*
74S330/1	*12*	*6*	*-*	*12*	*50*	*-*	*20*	*1976*	*2,5,6*
82S106/7	*16*	*8*	*-*	*16*	*48*	*-*	*28*	*1977*	*2,6,7*
PLS152/3	*8*	*-*	*10*	*17*	*32*	*10 AND*	*20*	*1980*	*8*
TIFPLA839/40	*14*	*6*	*-*	*14*	*32*	*-*	*24*	*1981*	*2,9*
100459	*16*		*-*	*16*	*24*	*-*	*64*	*1983*	*10*
PLS161	*12*	*8*	*-*	*12*	*48*	*-*	*24*	*1983*	*4*
PLS173	*12*	*-*	*10*	*21*	*32*	*10 AND*	*24*	*1985*	*8*
PLHS473	*11*	*2*	*9*	*20*	*24*	*11 OR*	*24*	*1985*	*8*
PEEL253	*8*	*-*	*10*	*17*	*42*	*10 OR*	*20*	*1987*	*8*
PEEL273	*12*	*-*	*10*	*21*	*42*	*10 OR*	*24*	*1987*	*8*

Tabla 1.1.4 Arquitecturas PLA

Notas para la tabla 1.1.4:

Todos los dispositivos tienen polaridad de salida programable a menos que se indique lo contrario.

- Versión " open-collector " disponible.
- Dispositivo de máscara programada.
- Incluye un pin de activación de salida.
- Entrada convertible en activador de salida.
- Salida desactivada si no hay producto activo.
- Incluye pin de salida que indica si hay un producto activo.
- Versión borrable disponible.
- Incluye dos salidas con activación de compuertas AND.
- Un dispositivo ECL.

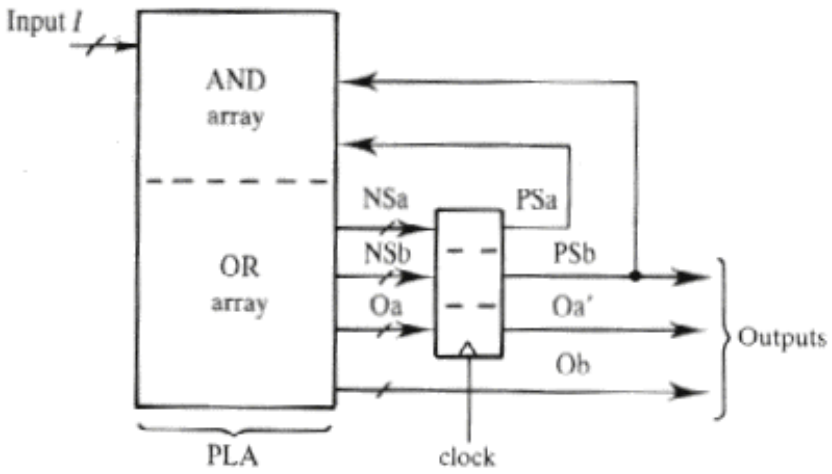


Figura. 1.1.7 Estructura genérica de un secuenciador

Nombre	Nº de entradas	Nº sin registrar	Nº registrados	Nº del producto	Complement array	Téminos de control	Nª de pins	Año de revelación	Notas			
		S	E/S	Buried	Estado+ Salida	Salida						
TMS2000	17	18	-	8J-K	-	-	60	NO	-	40	1970	2
TMS2200	13	10	-	5J-K	-	-	72	NO	-	28	1970	2
HPLA-0174	19	16	-	12D	-	-	70	NO	-	28	1975	2
PLS104/5	16	-	-	6S-R	-	8S-R	48	SI	-	28	1977	3,4
PLS154/5	4	-	8(P)	-	4J-K/ D(P)	-	32	SI	11	20	1980	4,5, 6,7
PLS156/7	4	-	6(P)	-	6J-K/ D(P)	-	32	SI	11	20	1980	4,5, 6,7
PLS158/9	4	-	4(P)	-	8J-K/ D(P)	-	32	SI	11	20	1980	4,5, 6,7
74PLS333/5	12	-	-	4J-K	-	6D	32	NO	-	24	1981	4,5, 8,9
PLS167	14	-	-	6S-R	2S-R	4S-R	48	SI	-	24	1983	3
PLS168	12	-	-	6S-R	4S-R	4S-R	48	SI	-	24	1984	3
PLS179	8	-	4(P)	-	8J-K/ D(L)	-	32	SI	11	24	1986	4,5, 6,7
PSG507	13	ver nota 11	-	8S-R	-	8S-R	80	NO	6	24	1986	10,11, 12,13
21F10	11	-	ver nota 11	-	10D(P)	-	32	NO	12	24	1985	5,11, 14,15, 16,17
GAL6001	11	-	10	8D-E	10D-E(L)	-	64	NO	11	24	1986	18,19, 20,21, 22
PLUS405	16	-	-	8S-R/ J-K	-	8S-R/ J-K	64	2	-	24	1986	23,24
PLS506	13	ver nota 11	-	16S-R	-	8S-R	97	SI	-	28	1987	11,12

Tabla 1.1.5 (a) Arquitecturas de secuenciadores

Nombre	Nº de entradas	Nº sin registrar	Nº registrados	Nº del producto	Complement array	Téminos de control	Nª de pins	Año de revelación	Notas			
		S	E/S	Buried	Estado+ Salida	Salida						
IBMPLA	18	-	-	13J-K	-	16D	70	NO	-	48	1974	2
µPB450	24	-	-	16J-K	-	16D	72	NO	-	48	1985	25

Tabla 1.1.5 (b) Secuenciadores con 2 bits decodificados

Notas para las tablas 1.1.5 (a) y 1.1.5 (b):

- La polaridad se indica allá donde no sea alta (P = programable).
- Dispositivo de máscara programable.

- Pin de activación preset/output.
- Versión "open-collector" disponible.
- Pin de activación de salida.
- Flip-flops en dos grupos para activación de salidas P, R y control de carga.
- Flip-flops que se pueden cargar desde los pines.
- Clock y reset de registro de estados desde la matriz OR.
- Las salidas están retrasadas.
- Contiene contadores de 6 bits controlables por secuenciador.
- Registros de salida con realimentación programable.
- Pin de control de salidas compartido con una entrada.
- Polaridad del reloj programable.
- Registros programables y puenteables.
- Reloj de registro de entrada, 2 pines de reloj de registros de estados/salidas.
- Relojes compartidos con las entradas.
- Términos iniciales síncronos y asíncronos.
- Entradas de registros/retrasos programables y puenteables.
- Registros de entrada y estados programables y puenteables.
- Flip-flops con relojes individuales o comunes.
- Relojes de entrada compartidos con las entradas.
- Un dispositivo borrable.
- Preset/resets programables compartidos con pines de activación de salida.
- Dos relojes, uno compartido con una entrada.
- Estados de flip-flops conectados en trayectoria repasada para fines de prueba

Nomenclatura de una PLA

Los líderes en fabricación de PLD, Texas Instruments y AMD, tienen una notación para identificar los dispositivos. Por ejemplo, la estructura en PLD AMD es, véase la figura 1.1.8

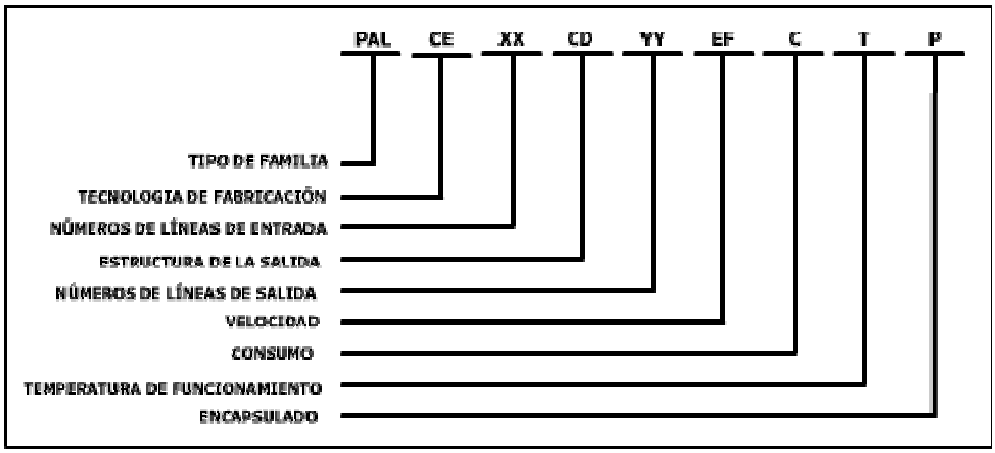


Figura 1.1.8 Ejemplo de Notación para PALs de AMD

Dentro de la estructura de salida se tienen las posibilidades contenidas en la tabla 1.1.6

Códigos	Tipos de Salidas
L	Combinatoria con nivel bajo activo.
H	Combinatoria con nivel alto activo.
R	Registro.
RA	Registro asíncrono.
X	Registro O exclusivo.
V	Vesátil.
M	Macrocélula.

Tabla 1.1.6 Tipos de Salidas de una PLA.

En el mercado se manejan referencias como la PAL16L8, PAL20L8, PAL20V8 y PAL20X8.

Consumo de corriente en los PLD

En la fabricación de PLD se utiliza tecnología bipolar TTL o ECL y tecnología CMOS. Los dispositivos bipolares son más rápidos y consumen más que los dispositivos CMOS. Actualmente los PLD bipolares presentan retardos de propagación inferiores a 7 ns y los consumos típicos rondan los 100-200 mA para un chip con 20-24 patillas.

Mientras los PLD bipolares sólo pueden programarse una vez, la mayoría de los PLD CMOS son reprogramables y permiten una fácil verificación por parte del usuario. A los PLD CMOS borrables por radiación ultravioleta se les denomina

EPLD y a los borrables eléctricamente se les conoce por EEPLD. Los EEPLD con encapsulados de plástico son más baratos que los EPLD provistos de ventanas de cuarzo que obligan a utilizar encapsulados cerámicos.

También existen las PALCE16V8Q (Quarter Power I_{cc} = 55 mA) y las PALCE16V8Z (Zero Power) con un bajísimo consumo estático de potencia.

Acostumbrados a trabajar con dispositivos CMOS con un consumo prácticamente nulo a frecuencia cero, resulta sorprendente una PLA CMOS con un consumo de 90 mA a la máxima frecuencia de operación (15 MHz), pero que todavía tendrá un consumo apreciable a frecuencia cero. En la actualidad, solamente una pequeña fracción de los PLD del mercado se anuncian como Zero Power.

La razón de estos consumos reside en que no existe una célula de memoria EPROM o EEPROM que sea verdaderamente CMOS. La mayoría de los PLD CMOS se construyen con un núcleo programable de transistores N-MOS, y solamente las entradas y las salidas del PLD utilizan drivers CMOS. La matriz de transistores NMOS precisa de una alimentación continua (figura 1.1.9), para poder responder con rapidez.

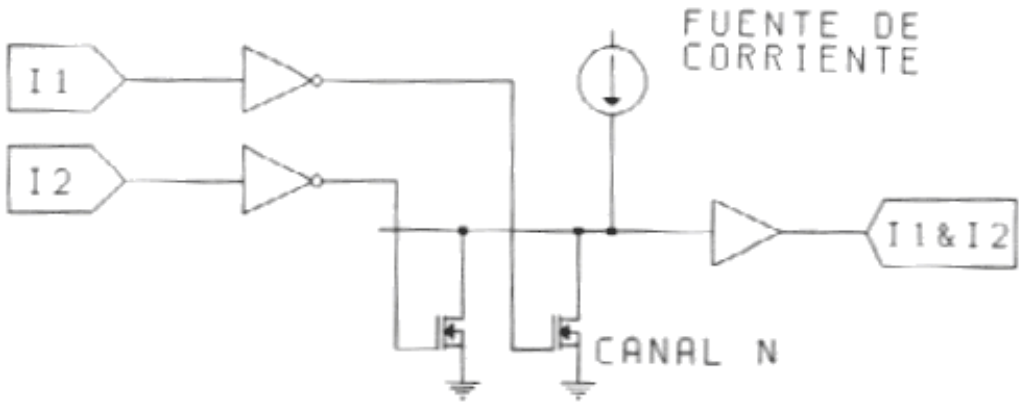


Figura 1.1.9 Término producto en un PLD

Para rebajar los consumos de la matriz de transistores NMOS se utilizan dos técnicas.

La primera de ellas consiste en dotar al PLD de una patilla o fusible de control de consumo de potencia (patilla o fusible Power Down), que quita la alimentación a la matriz de transistores cuando el PLD se encuentra fuera de servicio, proporcionando un menor consumo de potencia. Tiene los inconvenientes de que la puesta en funcionamiento del PLD es más lenta.

La segunda técnica (figura 1.1.10) coloca en las entradas de los PLD unos detectores de transición de estado, que conectan la alimentación a la matriz de

transistores durante un breve instante de tiempo después de que una entrada haya cambiado. Este tiempo deberá permitir el cambio de las salidas y su almacenamiento en latches, tras lo cual se puede quitar de nuevo la alimentación a la matriz de transistores.

El detector de transición de estado de las entradas se obtiene metiendo a las dos entradas de una compuerta OR-exclusiva el estado de una patilla de entrada y el estado de esa misma patilla demorada un tiempo. El tiempo de demora de las patillas de entrada será igual al tiempo durante el cual se mantendrá la alimentación a la matriz de transistores. Los detectores de transición de las entradas y los latches de las salidas se mantienen constantemente alimentados. El consumo de corriente de los PLD que utilizan esta segunda técnica aumenta lógicamente si se incrementa la frecuencia de cambio de las entradas.

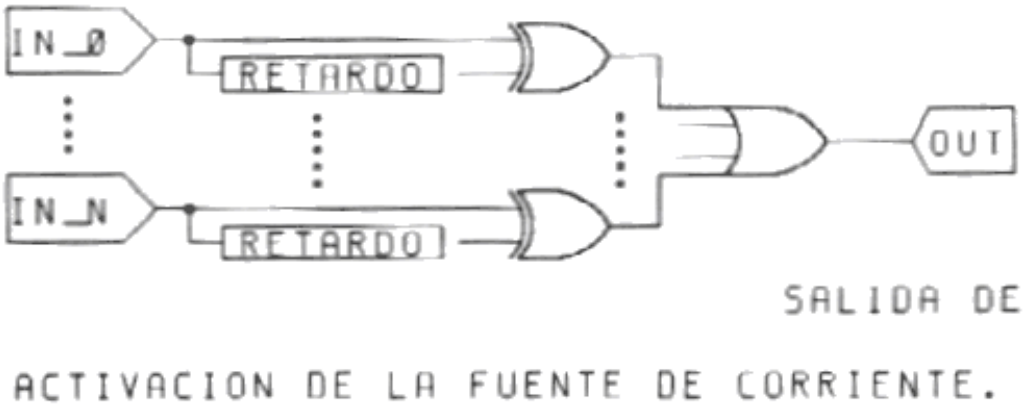


Figura 1.1.10 Circuito detector de transiciones

Cómo se catalogan los PLA

Si consultamos las hojas de datos de una PALCE16V8H-20, encontramos claves que permiten extraer valiosa información del nombre del dispositivo. La información incluida en el nombre nos indica:

PAL	Programmable Array Logic.
CE	C-MOS Electrically Erasable.
16V8	16 Entradas a la matriz de compuertas AND y ocho salidas.
H	Half Power (I_{ec} = 90 mA).
20	Tiempo de propagación = 20 ns.

Estructura de los dispositivos PLA

Los dispositivos PLA son PLD con o sin array OR, pero un conjunto de compuertas OR que suman grupos de productos. La estructura de los dispositivos PLA combinacionales se muestra en la figura 1.1.11 Los dispositivos combinacionales de 20 pines están listados en la tabla 1.1.7 y los de 24 pines están listados en la tabla 1.1.8

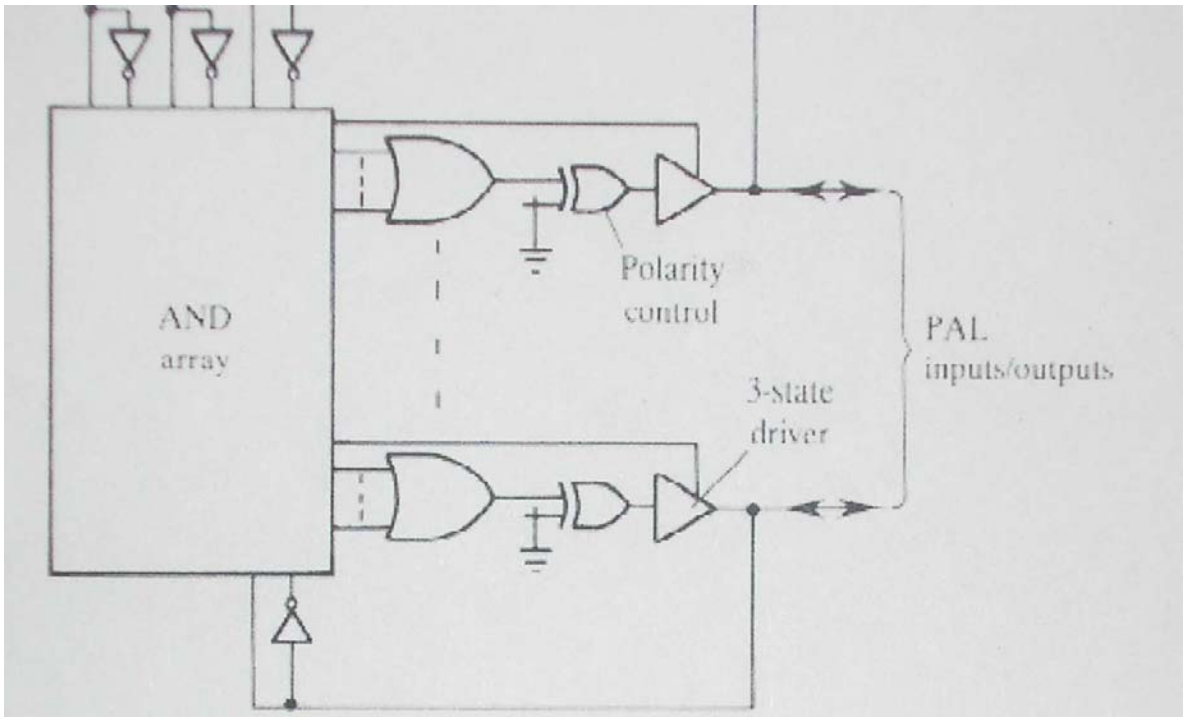


Figura 1.1.11 PLA estructurado con control de salidas

Nombre	*Nº de entradas*	*Nº de salidas*	*Nº de E/S*	*Maximo nº de entradas*	*Product term distribution*	*Nº de productos de control*	*Polaridad de salida*	*Año de revelación*	*Notas*
10H8	*10*	*8*	*-*	*10*	*8(2)*	*-*	*H*	*1977*	*1*
10L8	*10*	*8*	*-*	*10*	*8(2)*	*-*	*L*	*1977*	*1*
10P8	*10*	*8*	*-*	*10*	*8(2)*	*-*	*P*	*1983*	*1,7*
12H6	*12*	*6*	*-*	*12*	*4, 4(2), 4*	*-*	*H*	*1977*	*1*
12L6	*12*	*6*	*-*	*12*	*4, 4(2), 4*	*-*	*L*	*1977*	*1*

12P6	*12*	*6*	*-*	*12*	*4, 4(2), 4*	*-*	*P*	*1983*	*1,7*
14H4	*14*	*4*	*-*	*14*	*4(4)*	*-*	*H*	*1977*	*1*
14L4	*14*	*4*	*-*	*14*	*4(4)*	*-*	*L*	*1977*	*1*
14P4	*14*	*4*	*-*	*14*	*4(4)*	*-*	*P*	*1983*	*1,7*
16H2	*16*	*2*	*-*	*16*	*2(8)*	*-*	*H*	*1977*	*1*
16L2	*16*	*2*	*-*	*16*	*2(8)*	*-*	*L*	*1977*	*1*
16P2	*16*	*2*	*-*	*16*	*2(8)*	*-*	*P*	*1983*	*1,7*
16C1	*16*	*1+T*	*-*	*16*	*16*	*-*	*H and L*	*1977*	*1,2*
16H8	*10*	*2*	*6*	*16*	*8(7)*	*8*	*H*	*1982*	
16HD8	*10*	*8*	*-*	*16*	*8(8)*	*-*	*H*	*1982*	*1,3*
16HE8	*10*	*2*	*6*	*16*	*8(8)*	*-*	*H/P*	*1983*	*4*
16L8	*10*	*2*	*6*	*16*	*8(7)*	*8*	*L*	*1977*	*5,6*
16LD8	*10*	*8*	*-*	*16*	*8(8)*	*-*	*L*	*1982*	*1,3*
16LE8	*10*	*2*	*6*	*16*	*8(8)*	*-*	*L/P*	*1983*	*4*
16P8	*10*	*2*	*6*	*16*	*8(7)*	*8*	*P*	*1983*	
16SP8	*10*	*2*	*6*	*16*	*7, 3(7|7), 7*	*8*	*P*	*1986*	
18P8	*10*	*-*	*8*	*18*	*8(8)*	*8*	*P*	*1984*	
EPL10P8	*10*	*8*	*-*	*10*	*8(2|2)*	*-*	*P*	*1984*	*7,8*
EPL12P6	*10*	*6*	*-*	*12*	*4|4, 4(2|2), 4|4*	*-*	*P*	*1984*	*7,8*
EPL14P4	*10*	*4*	*-*	*14*	*4(4|4)*	*-*	*P*	*1984*	*7,8*
EPL16P2	*1612*	*2*	*-*	*16*	*2(8|8)*	*-*	*P*	*1984*	*7,8*
EPL16P8	*1014*	*2*	*6*	*16*	*4(7|7)*	*8*	*P*	*1984*	*7,9*

Tabla 1.1.7 (a). Dispositivos PLA 20-pin combinacional.

Nombre	*Nº de entradas*	*Nº de salidas*	*Nº de E/S*	*Maximo nº de entradas*	*Product term distribution*	*Nº de productos de control*	*Polaridad de salida*	*Año de revelación*	*Notas*
R29693	*10*	*4*	*-*	*10*	*N/A*	*-*	*L*	*1978*	*10*

Tabla 1.1.7 (b) Multiplexadores programables.

Notas para las tablas 1.1.7 (a) y la tabla 1.1.7 (b):

- Salidas "totem pole".
- Las dos salidas son complementarias.
- Seis salidas tienen realimentación interna a la matriz AND. La `D' significa "salidas dedicadas", o sea, que no pueden ser desconectadas y usadas como entradas externas.
- Salidas con su propia polaridad programable, buffers de la I/O con tres estados programables mediante fusibles.

- Uno de los dispositivos PLA combinacionales con 20 pines más populares.
- Disponible en versión borrable.
- Los dispositivos EPL (Electronically Programmable Logic) son borrables.

Nombre	*Nº de entradas*	*Nº de salidas*	*Nº de E/S*	*Maximo nº de entradas*	*Product term distribution*	*Nº de productos de control*	*Polaridad de salida*	*Año de revelación*	*Notas*
12H10	*12*	*10*	*-*	*12*	*10(2)*	*-*	*H*	*1984*	*1*
12L10	*12*	*10*	*-*	*12*	*10(2)*	*-*	*L*	*1981*	*1*
12P10	*12*	*10*	*-*	*12*	*10(2)*	*-*	*P*	*1983*	*1*
14H8	*14*	*8*	*-*	*14*	*4, 6(2), 2*	*-*	*H*	*1984*	*1*
14L8	*14*	*8*	*-*	*14*	*4, 6(2), 4*	*-*	*L*	*1981*	*1*
14P8	*14*	*8*	*-*	*14*	*4, 6(2), 4*	*-*	*P*	*1985*	*1*
16H6	*16*	*6*	*-*	*16*	*2(4), 2(2), 2(4)*	*-*	*H*	*1984*	*1*
16L6	*16*	*6*	*-*	*16*	*2(4), 2(2), 2(4)*	*-*	*L*	*1981*	*1*
16P6	*16*	*6*	*-*	*16*	*2(4), 2(2), 2(4)*	*-*	*P*	*1985*	*1*
18H4	*18*	*4*	*-*	*18*	*6, 2(4), 6*	*-*	*H*	*1984*	*1*
18L4	*18*	*4*	*-*	*18*	*6, 2(4), 6*	*-*	*L*	*1981*	*1*
18P4	*18*	*4*	*-*	*18*	*6, 2(4), 6*	*-*	*P*	*1985*	*1*
20H2	*20*	*2*	*-*	*20*	*2(8)*	*-*	*H*	*1984*	*1*
20L2	*20*	*2*	*-*	*20*	*2(8)*	*-*	*L*	*1981*	*1*
20P2	*20*	*2*	*-*	*20*	*2(8)*	*-*	*P*	*1985*	*1*
20C1	*20*	*1+T*	*-*	*20*	*16*	*-*	*H and L*	*1981*	*1,2*
20L8	*14*	*2*	*6*	*20*	*8(7)*	*8*	*L*	*1982*	*7*
20P8	*14*	*2*	*6*	*20*	*8(7)*	*8*	*P*	*1985*	
20L10	*12*	*2*	*8*	*20*	*10(3)*	*10*	*L*	*1981*	
20S10	*12*	*2*	*8*	*20*	*7, 4(7|7), 7*	*10*	*P*	*1983*	
20P10	*12*	*-*	*10*	*22*	*10(8)*	*10*	*P*	*1986*	
20XP10	*12*	*-*	*10*	*22*	*10(2|6)*	*10*	*P*	*1986*	*3*

Tabla 1.1.8 (a) Dispositivos PLA combinacional 24-pines.

Nombre	*Nº de entradas*	*Nº de salidas*	*Nº de E/S*	*Maximo nº de entradas*	*Product term distribution*	*Nº de productos de control*	*Polaridad de salida*	*Año de revelación*	*Notas*
20P8	*12*	*-*	*8*	*20*	*4(4|4)*	*8*	*P*	*1983*	*4*
16P8	*12*	*4*	*4*	*16*	*8(8)*	*4*	*P*	*1985*	*4,5*
16P4	*16*	*4*	*-*	*16*	*4(8)*	*-*	*P*	*1987*	*4,5*

12C4	*12*		*-*	*12*	*4(8)*	*-*	*H,L*	*1988*	*4,5*
16C4	*16*		*-*	*16*	*4(8)*	*-*	*H,L*	*1989*	*4,5,6*

Tabla 1.1.8 (b). Dispositivos ECL.

Notas para las tablas 1.1.8 (a) y la tabla 1.1.8 (b):

- Salidas "totem pole"
- Las dos salidas son complementarias.
- Compatible con ECL 10KH.
- Compatible con ECL 100KH.
- Dispositivo con 28 pines.
- Versión borrable disponible.

Los decodificadores o los FPGA (Field Programmable Gate Arrays) son PLA como los PLD combinacionales, que generalmente no tienen array OR; por lo que los productos son tomados directamente hacia las salidas. Estos son particularmente útiles para la decodificación de direcciones.

Las PLA conocidos tienen registros de salida alimentados desde el arreglo (array) lógico como el dispositivo mostrado en la figura 1.1.12. La tabla 1.1.9 lista las arquitecturas. Todos estos dispositivos tienen un reloj externo común, siendo estos apropiados para diseños síncronos.

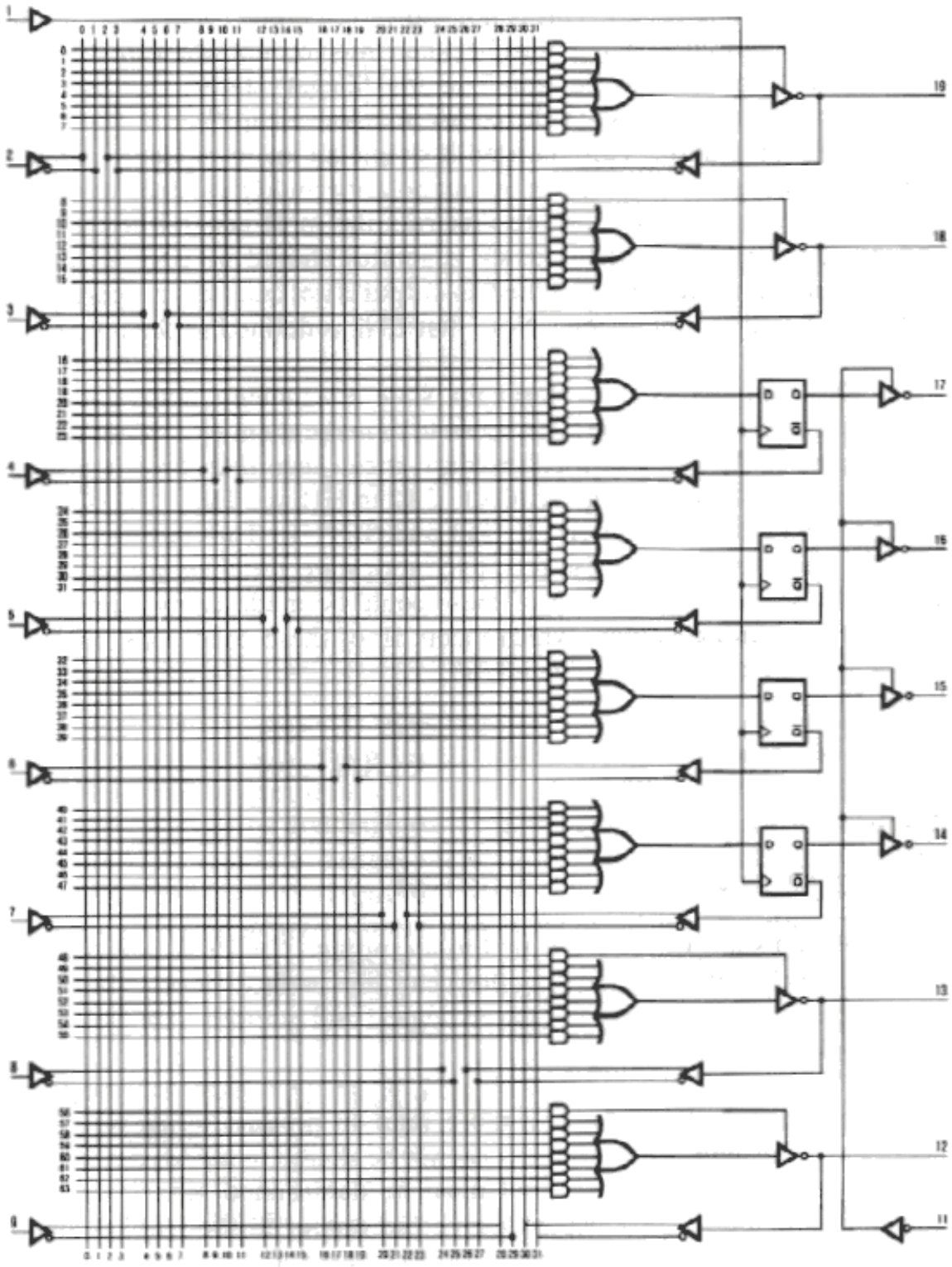


Figura 1.1.12 Dispositivo PAL 16R4

.

Nombre	Nº de entradas	Nº de registros con vuelta atrás (feedback)	Nº de E/S	Maximo nº de entradas	Product term distribution	Nº de productos de control	Polaridad de salida	Año de revelación	Notas
16R8	8	8	-	16	8(8)	-	L	1977	1
16RP8	8	8	-	16	8(8)	-	P	1983	1
16R6	8	6	2	16	7,6(8),7	2	L	1977	1
16RP6	8	6	2	16	7,6(8),7	2	P	1983	1
16R4	8	4	4	16	2(7),4(8), 2(7)	4	L	1977	1
16RP4	8	4	4	16	2(7),4(8), 2(7)	4	P	1983	1
16X4	8	4	4	16	ver nota 3	4	L	1978	1,2,3
16ª4	8	4	4	16	ver nota 4	4	L	1978	1,2,4
HPL-77000	10	8	-	18	8(4)	16	P	1982	5,6,7
EPL16RP8	8	8	-	16	4(8|8)	-	L	1985	1,8,9
EPL16RP6	8	6	2	16	(7|8),2(8|8),(8|7)	2	L	1985	1,8,9
EPL16RP4	8	4	4	16	(7|7),2(8|8),(7|7)	4	L	1985	1,8,9

Tabla 1.1.9 (a).Dispositivos PLA registradas 20-pines síncronos

.

Nombre	Nº de entradas	Nº de celdas de salida	Modos de celdas de salida	Maximo nº de entradas	Distribución de productos	Año de revelación	Notas
EP300*	10	8	i,l,o,r,t,v	18	8(8)	1984	9,11,12
16V8	10	8	p,t or(c), t or a, b	16	8(8)	1984	9,11,13,14
16F8	10	8	o,p,t,u,v,w	18	4(8|8 or 6|10 or 3|13 or 1|15)	1986	11,13,14,15
23S8	9	4 (ver nota 16)	o,r,v,t	17	2(6), 6(8), 4(10), 2(12)	1986	12,17,18
18CV10	8	10	o,t	18	2(8), 2(10), 2(12), 2(14), 2(16)	1986	11,12
18V8	10	8	a,b,p,t	18	8(8)	1987	

Tabla 1.1.9 (b). PLA genericas 20-pines síncronos

1.1.4 Memorias Programables de Sólo Lectura (PROM)

Un dispositivo programable por el usuario es aquel que contiene una arquitectura general pre-definida en la que el usuario puede programar el diseño final del dispositivo empleando un conjunto de herramientas de desarrollo. Las arquitecturas generales pueden variar pero normalmente consisten en una o más matrices de compuertas AND y OR para implementar funciones lógicas. Muchos dispositivos también contienen combinaciones de flip-flops y latches que pueden usarse como elementos de almacenaje para entrada y salida de un dispositivo.

Los dispositivos más complejos contienen macrocélulas. Las macrocélulas permiten al usuario configurar el tipo de entradas y salidas necesarias en el diseño como es el mostrado en la figura 1.1.13 cual hace referencia a un PROM (Programmable Read Only Memory) que tiene como definición lo siguiente:

1.1.4.1 Características de las PROM

Este tipo de dispositivo se basa en la utilización de una matriz AND fija, seguida de una matriz OR programable. La matriz programable esta formada por líneas distribuidas en filas y columnas en las cuales los puntos de cruce quedaran fijos por unos diodos en serie con unos fusibles que serán los encargados de aislar las uniones donde no se requiera la función lógica.

La fase de programación se realiza haciendo circular una corriente capaz de fundir el fusible en aquellas uniones donde no se desee continuidad. Por otra parte, para cada combinación de las señales de entrada, el codificador activa una única fila y a su vez activa aquella columna a las que esta todavía unida a través del diodo.

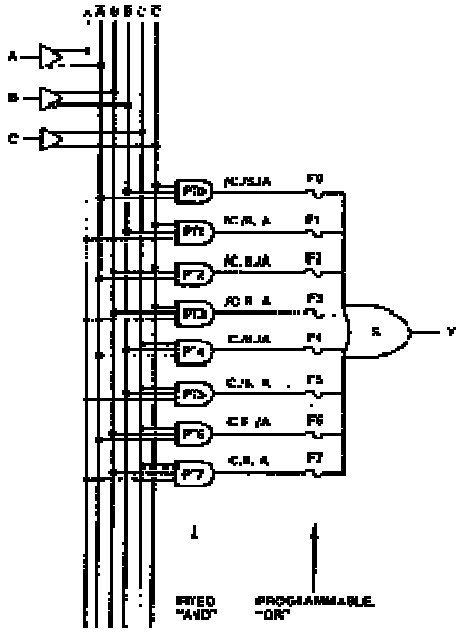


Figura 1.1.13 Ejemplo de un dispositivo que contiene Macrocélulas (PROM)

Las PROM son memorias programables de sólo lectura. Aunque el nombre no implica la lógica programable, las PROM, son de hecho lógicas. La arquitectura de la mayoría de las PROM consiste generalmente en un número fijo de términos AND que alimenta una matriz programable OR. Se usan principalmente para decodificar las combinaciones de entrada en funciones de salida. Otra definición complementaria de una PROM y por consiguiente un poco más entendible es la que a continuación se describe y además tiene una representación gráfica a través de caja negra o de bloque.

La PROM está formada por un conjunto fijo (no programable) de compuertas AND conectadas como decodificador y una matriz programable OR. La PROM se utiliza como una memoria direccionable y no como un dispositivo lógico. (Ver figura 1.1.14)

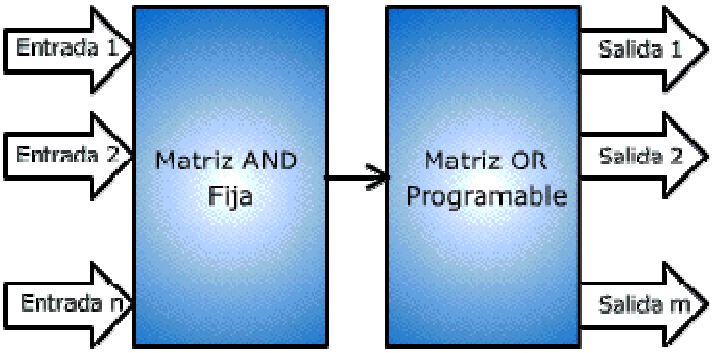


Figura 1.1.14 Diagrama de bloques de una PROM

Definición mínima de una PROM

PROM (Programmable Read Only Memory). Es un PLD en el que las uniones en la matriz de compuertas AND es fija, siendo programables las uniones en la matriz de compuertas OR. (Véase figura 1.1.15) Una PROM es un sistema combinacional completo que permite realizar cualquier función lógica con las n variables de entrada, ya que dispone de 2n términos productos. Están muy bien adaptadas para aplicaciones tales como: tablas, generadores de caracteres, convertidores de códigos, etc. Generalmente las PROM tienen menos entradas que las PLA y FPLA. Se pueden encontrar PROM con capacidades potencia de 2, que van desde las 32 hasta las 8192 palabras de 4, 8 o 16 bits de ancho.

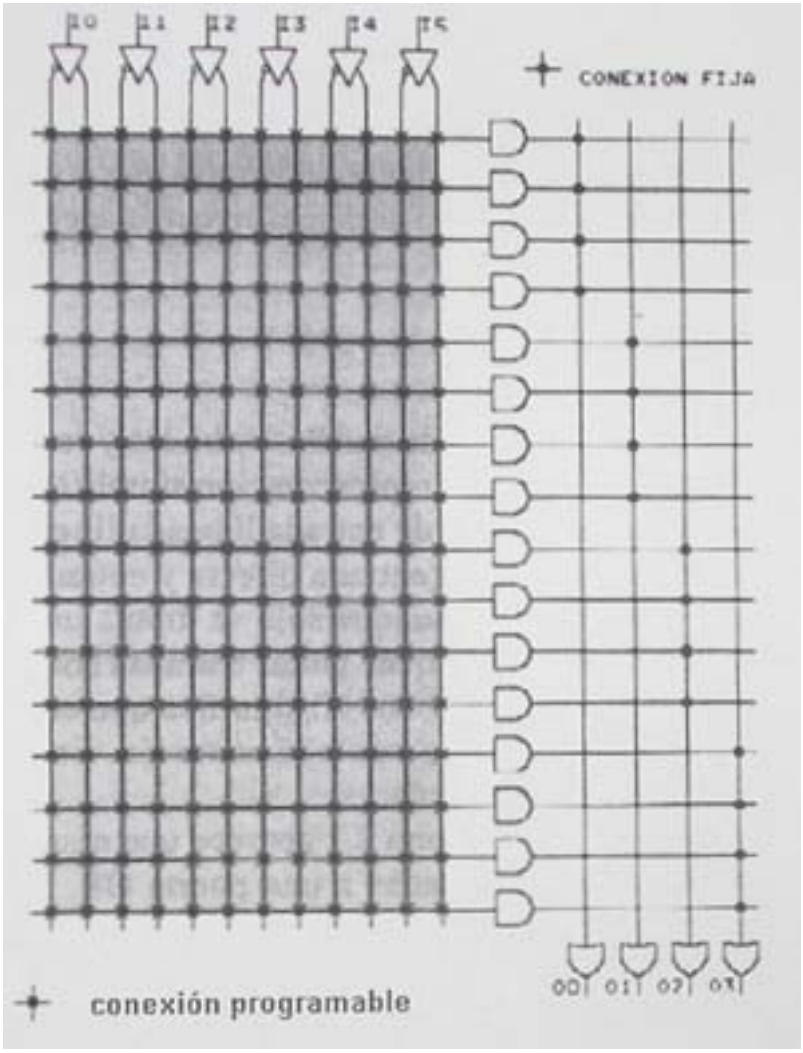


Figura 1.1.15 Estructura de una PROM

Interacción de una PROM sobre un PLD

Aunque las memorias PROM, EPROM y EEPROM son PLD, muchas veces se las excluye de esta denominación debido a que su contenido se define utilizando elementos de desarrollo propios de microprocesadores, tales como; ensambladores, emuladores y lenguajes de programación de alto nivel. Otras veces, cuando estas memorias se usan para realizar una función lógica y no para guardar un programa de un microprocesador, se las incluye dentro del término PLD.

1.1.5 Matriz Lógica Genérica (GAL)

Las GAL (Generic Array Logic) son dispositivos de matriz lógica genérica. Están diseñados para emular muchas PLA pensadas para el uso de macrocélulas. Si un usuario tiene un diseño que se implementa usando varias PLA comunes, puede configurar varias de las mismas GAL para emular cada de uno de los otros dispositivos. Esto reducirá el número de dispositivos diferentes en existencia y aumenta la cantidad comprada. Comúnmente, una cantidad grande del mismo dispositivo debería rebajar el costo individual del dispositivo. Estos dispositivos también son eléctricamente borrables, lo que los hace muy útiles para los ingenieros de diseño.

1.1.5.1 Características de los GAL

La GAL se forma con una matriz AND reprogramable y una matriz OR fija , con una salida lógica programable. La figura 1.1.16 muestra el diagrama de bloques de una GAL. Esta estructura permite implementar cualquier expresión lógica suma de productos con un número de variables limitado.

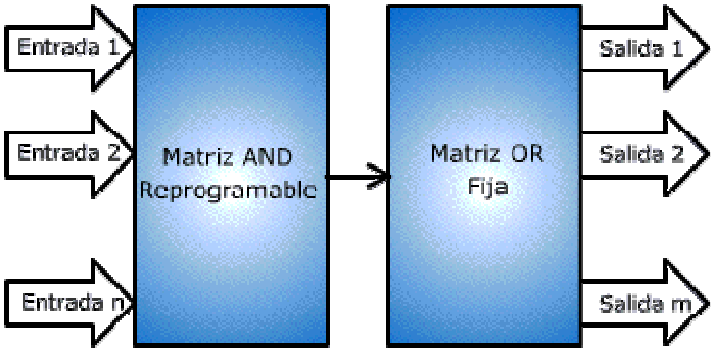


Figura 1.1.16 Diagrama de Bloques de una GAL

Las dos principales diferencias entre los dispositivos GAL y PLA son:

a) La GAL es reprogramable.

b) Tiene configuraciones de salida programables. La GAL se puede programar una y otra vez, ya que usa tecnología ECMOS (Electrically Erasable CMOS, CMOS borrable eléctricamente).

En la figura 1.1.17 se ilustra la estructura básica de una GAL con dos variables de entrada y una de salida. La matriz reprogramable es esencialmente una red de conductores ordenados en filas y columnas, con una celda CMOS eléctricamente borrable (E^2CMOS) en cada punto de intersección, en lugar de un fusible como en el caso de las PLA. Estos PLD son borrables y reprogramables. El transistor CMOS tiene 2 compuertas, una de ellas totalmente aislada, flotante.

Para programar cada celda se aplica o no una tensión mayor a V_{DD} (alta) en la compuerta no flotante. Al aplicar esta tensión el dieléctrico conduce y la compuerta flotante se carga negativamente, dejando en operación normal siempre abierto el transistor.

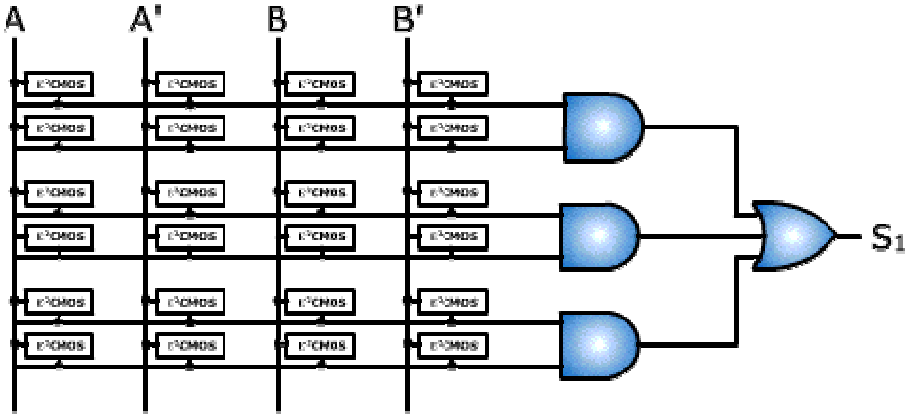


Figura 1.1.17 Estructura Básica de una GAL

En la figura 1.1.18 se muestra un ejemplo de una sencilla matriz GAL programada para obtener la suma de tres productos.

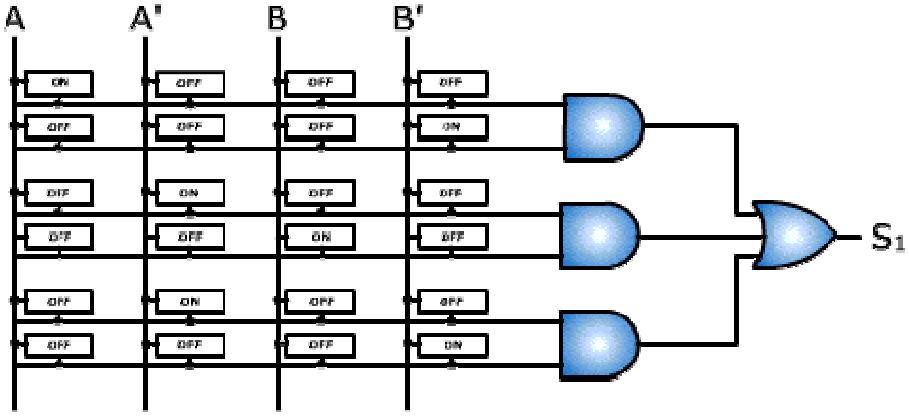


Figura 1.1.18 Programación de una GAL

El borrado se puede hacer de dos formas:

- Con luz ultravioleta (UV): Exponiendo el transistor de 5 a 20 minutos a luz UV, el dieléctrico conduce y permite la descarga de la compuerta flotante. Para este borrado el chip lleva una ventana de cuarzo transparente.
- Borrado eléctrico: Es el más usado hoy en día. La capa que aísla la compuerta flotante es más delgada. Al aplicar una tensión alta con polaridad contraria, la compuerta flotante se descarga porque el dieléctrico

conduce. Las ventajas más importantes de esta técnica son una descarga rápida, no se requiere UV y no se requiere sacar el chip de su base.

Una Matriz GAL, es una denominación que utilizaba originalmente Lattice Semiconductor y que más tarde se licenció a otros fabricantes. Una GAL en su forma básica es un PLD con una matriz AND reprogramable, una matriz OR fija y una lógica de salida programable mediante una macrocelda. Esta estructura permite implementar cualquier función lógica como suma de productos con un número de términos definido.

En los PLD no reprogramables la síntesis de las ecuaciones lógicas se realiza mediante quema de fusibles en cada punto de intersección de los pines de entrada con las compuertas. En el caso de una GAL es básicamente la misma idea pero en vez de estar formada por una red de conductores ordenados en filas y columnas en las que en cada punto de intersección hay un fusible, el fusible se reemplaza por una celda CMOS eléctricamente borrable (EECMOS). Mediante la programación se activa o desactiva cada celda EECMOS y se puede aplicar cualquier combinación de variables de entrada, o sus complementos, a una compuerta AND para generar cualquier operación producto que se desee. Una celda activada conecta su correspondiente intersección de fila y columna, y una celda desactivada desconecta la intersección. Las celdas se pueden borrar y reprogramar eléctricamente.

GAL comerciales

Las diversas GAL tienen el mismo tipo de matriz programable. Se diferencian en el tamaño de la matriz, en el tipo de OLMC (Las macroceldas lógicas de salida que contienen circuitos lógicos programables que se pueden configurar como entrada o salida combinacional y secuencial) y en los parámetros de funcionamiento, tales como velocidad y disipación de potencia (véase tabla 1.1.10).

Referencia	Número de Pines	t_{PD}	I_{CC} (mA)	Características
GAL16V8A	20	10, 15, 25	55, 115	E^2CMOS PLD Genérica
GAL18V10	20	15, 20	115	E^2CMOS PLD Universal
GAL22V8A	24	10, 15, 25	55, 115	E^2CMOS PLD Genérica
GAL22RA10	24	15, 20	115	E^2CMOS PLD Universal
GAL22V10	24	10, 15, 25	130	E^2CMOS PLD Universal
GAL26CV12	28	15, 20	130	E^2CMOS PLD Universal
GAL6001	24	30, 35	150	E^2CMOS FPLA
ispGAL16Z8	24	30, 35	190	E^2CMOS PLD Programable en Circuito

Tabla 1.1.10 Familias GAL del fabricante Lattice

1.1.6 Dispositivos Lógicos Programables Complejos (CPLD)

Los PLD complejos son lo que el nombre implica, dispositivos complejos de lógica programable. Se consideran PLA muy grandes que tienen algunas características de los PLA. La arquitectura básica es muy parecida a la PLA con la capacidad para aumentar la cantidad de términos AND para cualquier término OR fijo. Esto se puede realizar quitando términos AND adyacentes o empleando términos AND desde una matriz expandida. Esto permite que cualquier diseño pueda ser implementado dentro de estos dispositivos.

Un CPLD (Complex Programmable Logic Device) extiende el concepto de un PLD a un mayor nivel de integración ya que permite implementar sistemas con un mejor desempeño porque utilizan menor espacio, mejoran la confiabilidad en el circuito, y reducen costos. Un CPLD se forma con múltiples bloques lógicos, cada uno similar a un PLD. Los bloques lógicos se comunican entre sí utilizando una matriz programable de interconexiones lo cual hace más eficiente el uso del silicio, conduciendo a un mejor desempeño y un menor costo. A continuación se explican brevemente las principales características de la arquitectura de un CPLD.

1.1.6.1 Características de los CPLD

Matriz de Interconexiones Programables

La matriz de interconexiones programables (PIM) permite unir los pines de entrada/salida a las entradas del bloque lógico, o las salidas del bloque lógico a las entradas de otro bloque lógico o inclusive a las entradas del mismo. La mayoría de los CPLD usan una de dos configuraciones para esta matriz: interconexión mediante arreglo o interconexión mediante multiplexores.

El primero se basa en una matriz de filas y columnas con una celda programable de conexión en cada intersección. Al igual que en el GAL esta celda puede ser activada para conectar/desconectar la correspondiente fila y columna. Esta configuración permite una total interconexión entre las entradas y salidas del dispositivo o bloques lógicos. Sin embargo, estas ventajas provocan que disminuya el desempeño del dispositivo además de aumentar el consumo de energía y el tamaño del componente.

En la interconexión mediante multiplexores, existe un multiplexor por cada entrada al bloque lógico. Las vías de interconexión programables son conectadas a las entradas de un número de multiplexores por cada bloque lógico. Las líneas de selección de estos multiplexores son programadas para permitir que sea seleccionada únicamente una vía de la matriz de interconexión por cada multiplexor la cual se propagara hacia el bloque lógico. Cabe mencionar que no todas las vías son conectadas a las entradas de cada multiplexor. La rutabilidad se

incrementa usando multiplexores de mayor tamaño, permitiendo que cualquier combinación de señales de la matriz de interconexión pueda ser enlazada hacia cualquier bloque lógico. Sin embargo, el uso de grandes multiplexores incrementa el tamaño de dispositivo y reduce su desempeño.

Bloques lógicos

Un bloque lógico es similar a un PLD, cada uno posee un arreglo de compuertas AND y OR en forma de suma de productos, una configuración para la distribución de estas sumas de productos, y macroceldas. El tamaño del bloque lógico es una medida de la capacidad del CPLD, ya que de esto depende el tamaño de la función booleana que pueda ser implementada dentro del bloque. Los bloques lógicos usualmente tienen de 4 a 20 macroceldas.

Distribución de productos

Existen pequeñas diferencias en cuanto a las matrices de productos, esto dependerá del CPLD y del fabricante. Obviamente el tamaño de las sumas sigue siendo el factor más importante para la implementación de funciones booleanas.

Cada fabricante distribuye los productos de diferente forma. La familia MAX de CPLD fue desarrollada por Cypress Semiconductor junto con Altera Corporation, siendo los primeros en sacar al mercado una familia de CPLD. Altera la llamó MAX5000 y Cypress por su parte la clasificó como MAX340. En un dispositivo como el 22V10 tenemos que la suma de productos es fija por cada macrocelda - 8, 10, 12, 14 o 16 -, en la familia MAX se colocan 4 productos por macrocelda los cuales pueden ser compartidos con otras macroceldas. Cuando un producto puede ser únicamente utilizado por una macrocelda se le conoce como término-producto dirigido, y cuando estos pueden ser utilizados por otras macroceldas se le llama término-producto compartido. Mediante estos productos compartidos se mejora la utilización del dispositivo, sin embargo, esto produce un retardo adicional al tener que retroalimentar un producto hacia otra macrocelda y con esto disminuye la velocidad de trabajo del circuito. La forma en que son distribuidos los productos repercute en la flexibilidad que proporciona el dispositivo para el diseñador. Además, que estos esquemas proporcionan también flexibilidad para los algoritmos del programa de síntesis que es el que finalmente selecciona la mejor forma en que deben ser distribuidas las funciones booleanas en el dispositivo.

Macroceldas

Las macroceldas de un CPLD son similares a las de un PLD. Estas también están provistas con registros, control de polaridad, y buffers para salidas en alta impedancia.

Por lo general un CPLD tiene macroceldas de entrada/salida, macroceldas de entrada y macroceldas internas u ocultas (buried macrocells), en tanto que un 22V10 tiene solamente macroceldas de entrada/salida. Una macrocelda interna es similar a una macrocelda de entrada/salida, sólo que esta no puede ser conectada directamente a un pin de salida. La salida de una macrocelda interna va directamente a la matriz de interconexión programable.

Las macroceldas de entrada, son utilizadas para proporcionar entradas adicionales para las funciones booleanas. En general las macroceldas de entrada incrementan la eficiencia del dispositivo al ofrecer algunos registros adicionales con los que se pueden almacenar el valor del pin de entrada, lo cual puede ser útil al momento de obtener las funciones booleanas.

Celda de entrada/salida

La función de una celda de entrada/salida es permitir el paso de una señal hacia dentro o hacia afuera del dispositivo. Dependiendo del fabricante y de la arquitectura del CPLD estas celdas pueden ser consideradas o no parte del bloque lógico.

1.2 Otras características y clasificaciones de los PLD

El Incremento de popularidad y de utilización de los dispositivos lógicos programables o PLD está siguiendo un proceso solamente comparable al que hace algunos años acompañó a los microprocesadores. Los PLD se utilizan en casi todos los nuevos equipos electrónicos de control, industriales, de consumo, de oficina, de comunicaciones, etc.

Desde finales de la década de los sesenta, los equipos electrónicos digitales se han construido utilizando circuitos integrados de función lógica fija, realizados en pequeña o mediana escala de integración. Para las realizaciones muy complejas que exigirían un número elevado de Circuitos Integrados (CI) de función fija, se utilizan circuitos diseñados a medida que sólo sirven para una aplicación. Son los llamados CI específicos a una aplicación o ASIC (Application Specific Integrated Circuit). Por regla general, los ASIC los producen los fabricantes de CI con las especificaciones proporcionadas por el usuario.

Los equipos realizados con ASIC ocupan menos espacio, son más fiables, consumen menos energía y en grandes series resultan más baratos que los equipos equivalentes realizados con CI de función fija. Por otro lado, estos circuitos son muy difíciles de copiar.

Diferentes modalidades de ASIC son; los Circuitos a la Medida (Full Custom), las Matrices de Compuertas (Gate Arrays), las Células Normalizadas (Standard Cell) y los FPIC (Field Programmable Integrated Circuits); estos últimos son circuitos programables por el usuario final.

Circuitos integrados a la medida (Full Custom)

Los circuitos integrados a la medida (Full Custom), se diseñan a petición de un cliente para que resuelvan una determinada aplicación. Conllevan un alto costo de desarrollo y su empleo sólo se justifica para volúmenes de producción muy elevados. El tiempo necesario para la construcción de un CI a la medida es considerable ya que puede oscilar de unos meses a unos años.

Matrices de Compuertas (Gate Arrays)

Las matrices de compuertas (Gate Arrays) son pequeños trozos de silicio pendientes de algún proceso de metalización que defina las conexiones entre un importante número de compuertas o transistores que poseen en su interior. Las matrices de compuertas proporcionan densidades superiores a las 100.000 compuertas, con un aprovechamiento del 80 al 90 por 100 para los dispositivos pequeños y del 40 por 100 para los grandes.

Los fabricantes de silicio ponen a disposición de sus potenciales clientes abundante documentación sobre estos Gate Arrays, con una serie de macros que pueden utilizar de forma inmediata y otras que pueden construirse ellos mismos. Los macros son agrupaciones de un número de células básicas que realizan funciones comunes como; sumadores; compuertas NOT, AND, NAND, NOR XOR, etc.; latches y flip-flops S-R, J-K, D; buffer; osciladores; registros, decodificadores, multiplexores, etc.

Junto a esta documentación, los fabricantes aportan un software que contabiliza el número de células básicas utilizadas por todas las macros, sugiere el Gate Array adecuado para la aplicación, calcula la potencia disipada por el Gate Array que alojará el diseño del cliente, proporciona información sobre los tiempos de propagación de las señales y permite verificar el funcionamiento del circuito.

Una vez superadas todas las etapas previas, el cliente envía la documentación generada al fabricante para que éste ultime los procesos de metalización y fabrique un primer prototipo. El diseño con Gate Arrays puede durar semanas o meses. Requiere un volumen alto de circuitos para justificar sus costos.

Células Normalizadas (Standard Cell)

Las células normalizadas (Standard Cell) son, en cierta forma, similares a las Matrices de Compuertas. Su principal ventaja sobre ellas es que en lugar de trabajar con simples compuertas o transistores, se dispone de colecciones de diferentes partes de circuitos que han sido depurados (compuertas lógicas, circuitos MSI, RAM estáticas, ficheros de registro, etcétera). El usuario tiene que ensamblar estos circuitos, verificarlos y finalmente enviar documentación al fabricante de silicio para el desarrollo del primer prototipo. A pesar del concepto de célula normalizada, los períodos y los costos de desarrollo son superiores a los de las matrices de compuertas.

En las matrices de compuertas sólo hay que realizar la máscara final que define las conexiones entre las compuertas, mientras que en las células normalizadas, hay que realizar máscaras para todos los procesos de producción de los CI. Una vez más, el volumen de fabricación deberá ser los suficientemente alto como para amortizar la inversión económica realizada en el desarrollo.

Circuitos Integrados de Campo Programado (FPIC)

Los FPIC (Field Programmable Integrated Circuits): son chips programables por el usuario mediante programadores comerciales. El término FPIC también incluye a los CI no destinados a las aplicaciones lógicas. Son las memorias, los microcontroladores, los PLD (Programmable Logic Device), las FPGA (Field Programmable Gate Array) y los ASPLD (Aplication Specific Programmable Logic Devices).

Los FPIC ofrecen soluciones de bajo costo, de tiempo de desarrollo corto y con menor riesgo que los circuitos a medida, las matrices de compuertas y las células normalizadas.

Dispositivos Lógicos de Aplicación Específica Programable (ASPLD)

Los ASPLD (Application Specific Programmable Logic Devices) son PLD diseñados para realizar funciones específicas como, decodificadores de alta velocidad, secuenciadores, interfaces para buses particulares, periféricos programables para microprocesadores, etc.

Partes del ASPLD son programables permitiendo la adaptación del circuito a una aplicación determinada, pero manteniendo su función básica; así, por ejemplo, un decodificador lo personaliza el usuario, pero sigue siendo un decodificador. Estos circuitos están muy optimizados para la función para la que han sido diseñados. Los decodificadores sólo tienen un término producto, carecen de compuertas OR y resultan por consiguiente muy rápidos; por otro lado, los circuitos de interfase para buses normalmente tienen un fan-out elevado.

1.3 Características del diseño con PLD

Los PLD están situados en una zona intermedia entre los dispositivos a la medida y la lógica de catálogo formada por los CI de función fija. Tienen casi todas las ventajas de los ASIC sin estar penalizados por un costo elevado para pequeñas series. Además el ciclo de diseño con PLD es mucho más rápido que los de las matrices de compuertas o las células normalizadas. En determinadas aplicaciones, un PLD puede sustituir desde unos pocos hasta unas decenas de CI de función fija, mientras que los grandes ASIC pueden sustituir a cientos e incluso miles de CI. En ocasiones, los PLD se utilizan para realizar prototipos que posteriormente se llevarán a un ASIC más complejo.

El trabajo con PLD proporciona: facilidad de diseño, prestaciones, fiabilidad, economía y seguridad.

Facilidad de diseño

Las herramientas de soporte al diseño con PLD facilitan enormemente este proceso. Las hojas de codificación que se utilizaban en 1975 han dejado paso a los ensambladores y compiladores de lógica programable (PALASM, AMAZE, ABEL, CUPL, OrCAD/PLD, etc.). Estas nuevas herramientas permiten expresar la lógica de los circuitos utilizando formas variadas de entrada tales como; ecuaciones, tablas de verdad, procedimientos para máquinas de estados, esquemas, etc. La simulación digital posibilita la depuración de los diseños antes de la programación de los dispositivos. Todo el equipo de diseño se reduce a un software de bajo costo, que corre en una PC, y a un programador.

Prestaciones

Los PLD TTL que hay en el mercado tienen tiempos de conmutación tan rápidos como los circuitos integrados de función fija más veloces. Los PLD ECL son todavía más rápidos. Sin embargo, el incremento de velocidad obtenido con los dispositivos CMOS, que ya han igualado o superado en prestaciones a los dispositivos TTL, está provocando el abandono de la tecnología bipolar por parte de los fabricantes. En cuanto al consumo de potencia, los PLD generalmente consumen menos que el conjunto de chips a los que reemplazan.

Fiabilidad

Cuanto más complejo es un circuito, más probabilidades hay que alguna de sus partes falle. Puesto que los PLD reducen el número de chips en los sistemas, la probabilidad de un fallo disminuye. Los circuitos impresos con menor densidad de CI son más fáciles de construir y más fiables. Las fuentes de ruido también se reducen.

Economía

En este apartado, hay aspectos que resultan difíciles de cuantificar. Por ejemplo, los costos de pérdida de mercado por una introducción tardía de un producto. Otros son más claros, por ejemplo, la reducción del área de las placas de circuito impreso obtenida gracias a que cada PLD sustituye a varios circuitos integrados de función fija. Muchas veces se consigue reducir el número de placas de circuito impreso economizándose en conectores. La reducción de artículos en almacén también aporta ventajas económicas.

De la misma manera que para altos volúmenes de producción las memorias ROM resultan de menor costo que las EPROM, las HAL (Hard Array Logic) o PLD programados por el fabricante proporcionan ahorros adicionales en grandes cantidades.

Seguridad

Los PLD tienen fusibles de seguridad que impiden la lectura de los dispositivos programados, protegiendo los diseños frente a copias.

Además de los puntos mencionados, podemos añadir que los PLD facilitan el ruteado de las placas de circuito impreso debido a la libertad de asignación de patillas que proporcionan. Permiten realizar modificaciones posteriores del diseño y en ocasiones hacen posible la reutilización de circuitos impresos con algunos fallos, mediante una reasignación de los PLD.

Capítulo 2

Síntesis y simulación para componentes programables utilizando diferentes tipos de lenguajes de descripción

La evolución espectacular de la tecnología y del diseño micro-electrónico ha permitido la realización de sistemas electrónicos digitales complejos en un único circuito integrado de escala de integración progresivamente elevada (muy alta (VLSI, Very Large Scale Integrated), ultra alta (ULSI, Ultra Large Scale Integrated), giga (GSI, Giga Scale Integrated)) que llegan a contener en su interior más de 1.000.000 compuertas lógicas. Pero esto sólo se ha podido llevar a cabo mediante un cambio profundo de las técnicas de diseño de circuitos integrados.

Los circuitos de escala de integración media y alta (MSI Médium Scale Integrated y LSI Large Scale Integrated respectivamente) se diseñaron mediante la realización de un prototipo formado por módulos más sencillos y la comprobación de su funcionamiento antes de proceder a la integración. Esta forma de diseño recibe el nombre de abajo a arriba (bottom-up) porque se enlazan diversos módulos para constituir un bloque funcional más complejo. Pero en el caso de los circuitos integrados de complejidad VLSI y superiores no resulta práctica la realización física de un prototipo y por ello es necesario simular y verificar su correcto comportamiento antes de integrarlos. Ello trajo consigo la necesidad del desarrollo de métodos de diseño asistido por computador divididos en varias fases que, a partir de la especificación del funcionamiento, llevan hasta la descripción física del circuito, por lo que reciben el nombre de arriba a abajo (top-down). Así, mediante simulación es posible una rápida detección de errores en fases tempranas del diseño, resulta factible la reutilización del mismo para diferentes tecnologías y se pueden utilizar las herramientas de síntesis actuales para obtener rápidamente un esquema lógico o estructural y, en definitiva, un conjunto de instrucciones que indican las interconexiones entre los componentes de un diseño (netlist) de entrada para el trazado físico (layout) del ASIC, MCM, etc. o la asignación de recursos (mapping) en el caso de la lógica programable (CPLD y FPGA).Todo ello, obviamente, incrementa la productividad y la eficacia del diseño.

2.1 Herramientas computacionales utilizadas en la metodología de diseño (Top – Down)

En el diseño top-down se captura una idea en un nivel de abstracción alto y se implementa a partir de esta descripción, es un proceso hacia abajo incrementando el nivel de detalle según lo requerido. La figura 2.1.1 muestra la forma de diseño top-down. En el primer nivel de la figura se aprecia un sistema

inicial dividido en módulos, los cuales se dividen sucesivamente hasta llegar a los componentes básicos del circuito o elementos primarios. Estos elementos se enmarcan en un cuadrado con la línea más gruesa. Los métodos de diseño se basan en programas computacionales conocidos como herramientas de automatización del diseño electrónico (EDA tools), las cuales sobresalen por ofrecer una reducción significativa en el tiempo del diseño.

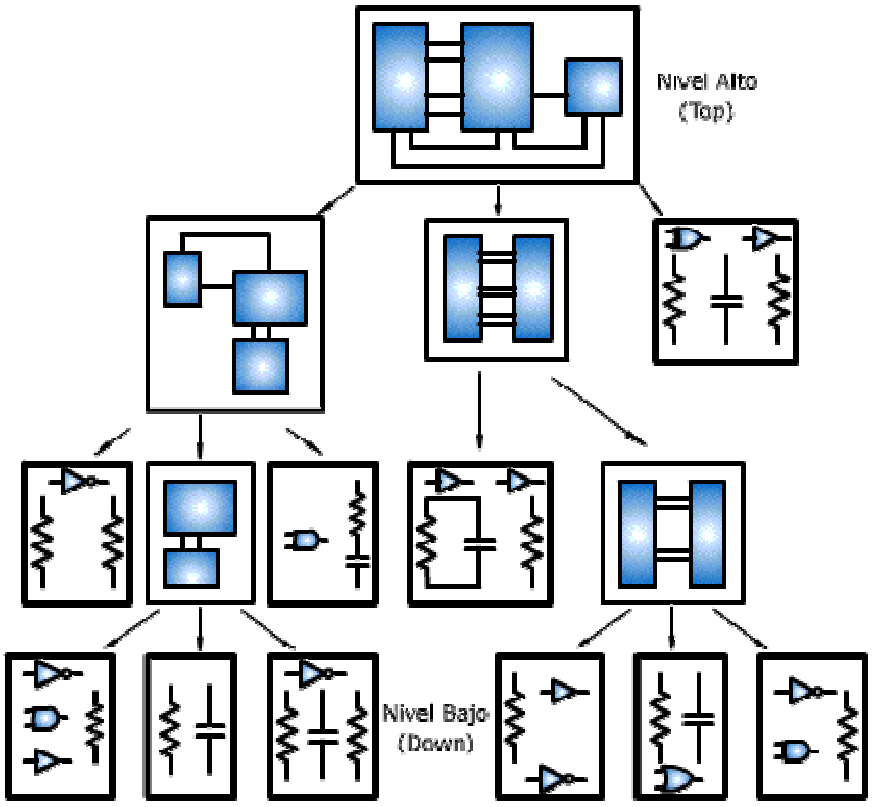


Figura 2.1.1 Metodología de Diseño Top – Down

Las herramientas siguen el diagrama de flujo de la figura 2.1.2

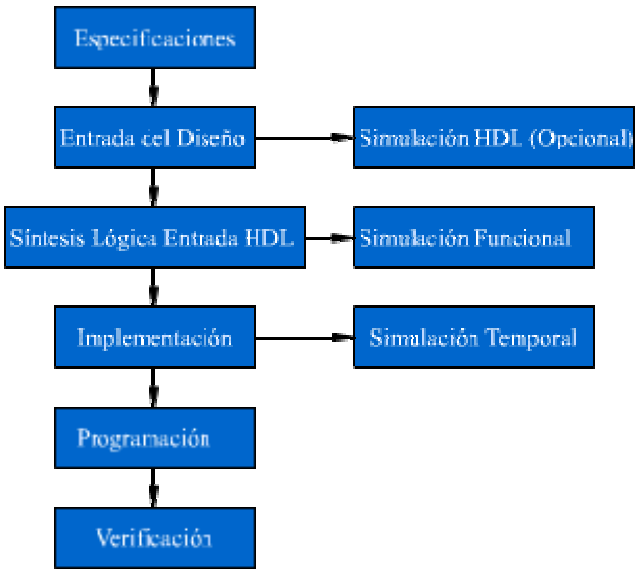


Figura 2.1.2 Diagrama de Flujo con herramientas EDA

Este proceso se resume en los siguientes pasos:

1. Planteamiento de las especificaciones.
2. Entrada del diseño: En esta etapa se realiza una descripción del circuito, para la cual existen varias alternativas,
   a. Captura esquemática: Dibujo del circuito mediante interfaz gráfica, puede ser un diagrama de bloques.
   b. Mediante lenguajes de descripción *HDL* como *VHDL*, *Verilog*, *Abel* y *CUPL*.
   c. Diagramas de transición de estados.
   d. Formas de onda –Tablas de verdad.
3. Simulación *HDL* (opcional): Simula el comportamiento del circuito que se acaba de describir antes de la síntesis.
4. Síntesis lógica: Consiste en tomar la descripción *HDL* y a partir de ella, generar y simplificar las ecuaciones lógicas correspondientes al circuito descrito.
5. Simulación funcional: Simula las ecuaciones lógicas, sin tener en cuenta los retardos.
6. Implementación del diseño: Los pasos a seguir dependen del tipo de PLD que se esté utilizando en el diseño. Trazado del mapa, colocación y enrutamiento, creación del archivo para la programación del dispositivo. Revisa si el circuito se adapta al chip; número de salidas, número de términos productos por salida.

7. Simulación temporal: Después de la implementación ya se conoce como queda programado el circuito y se puede realizar una simulación teniendo en cuenta los retardos.
8. Programación: La implementación genera un archivo JEDEC que indica el estado de las conexiones. Este archivo se usa para programar (o quemar el chip).

2.1.1 Ventajas del diseño Top-Down

La metodología de diseño descendente disminuye el tiempo de diseño. Por medio de los programas CAD para diseño de impresos se ha logrado disminuir el tiempo a 1/10 parte de lo que se gastaba antes, cuando esto se hacia manualmente. En la realización de las simulaciones no es necesario sólo un prototipo, ya que éste generalmente funciona; antes se debía repetir el proceso 2 o 3 veces hasta que el prototipo funcionara.

Las últimas herramientas de diseño electrónico permiten implementar de forma automática la metodología de diseño top-down.

2.1.2 El concepto de herramientas CAD

CAD son las siglas de Computer Aided Design, o Diseño Asistido por Computadora el cual constituye todo un proceso de trabajo utilizando técnicas de análisis apoyadas en gráficos mediante sofisticadas herramientas de software las cuales facilitan el estudio de los problemas asociados con el diseño en cuestión. El concepto CAD se relaciona con el dibujo como parte importante en el proceso de diseño pero, además, el diseño de un circuito debe cumplir con los requerimientos especificados por el equipo de diseño, por las normas de calidad existentes, los costos, etc. por lo que las herramientas CAD intervienen en todas las fases del diseño. Ya que no sólo son importantes por acelerar el desarrollo del sistema al permitir que varias personas puedan trabajar simultáneamente en distintas etapas del diseño sino que, además, es posible verificar el funcionamiento del circuito mediante la simulación del sistema. Todo esto simplifica la tarea del equipo de diseño y conduce a la conclusión del prototipo en menos tiempo.

2.1.3 Herramientas para la Automatización del Diseño Electrónico (EDA Tools)

Las herramientas EDA ("Electronic Design Automation") son las herramientas de hardware y software utilizadas en el diseño de sistemas electrónicos.

El diseño de hardware tiene un inconveniente que no existe en el desarrollo de software. El problema es el alto costo en el ciclo de diseño→ desarrollo del prototipo→ pruebas→ reinicio del ciclo. La etapa de costo más elevado es el prototipo. Por necesidad del mercado, se impone la reducción de costos en esta etapa, con el fin de incluir la fase de desarrollo del prototipo al final del proceso, evitando la repetición de varios prototipos, razón por la cual se encarece el ciclo. La introducción de la fase de simulación y verificación de circuitos utilizando herramientas EDA, hace no necesaria la comprobación del funcionamiento del circuito por medio de la implementación física del prototipo.

Las herramientas EDA están presentes en todas las fases del ciclo de diseño de circuitos. Primero en la fase de generación del sistema que puede representarse en un diagrama esquemático, en bloques o de flujo.

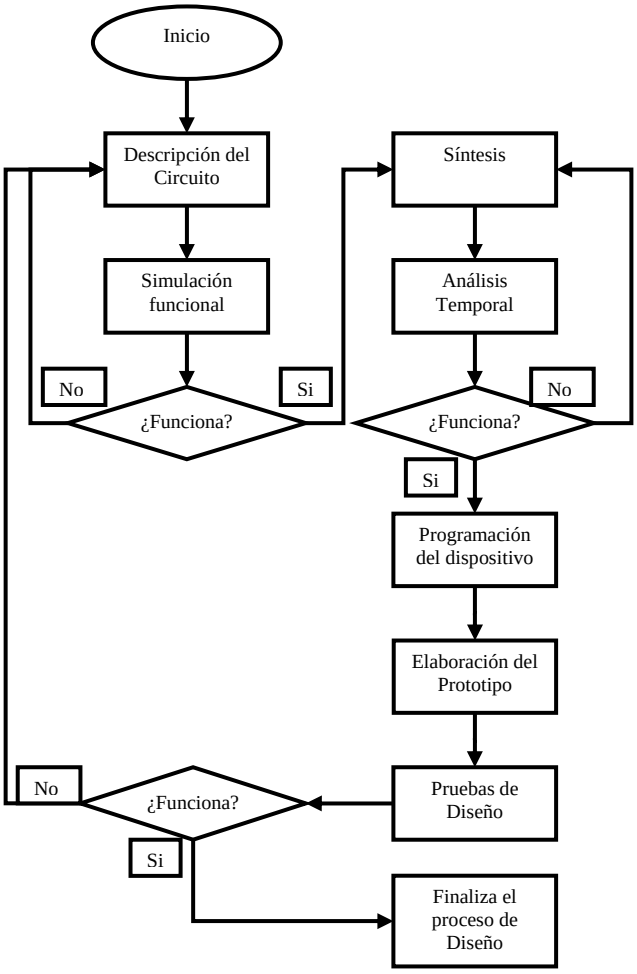


Figura 2.1.3 Flujo de Diseño en el desarrollo de Sistemas electrónicos

Se encuentra también la fase de simulación y comprobación de circuitos, donde diferentes herramientas permiten verificar el funcionamiento del sistema.

Estas simulaciones pueden ser de eventos, funcionales, digitales o eléctricas, de acuerdo al nivel requerido. Después están las herramientas EDA utilizadas en la síntesis y programación de circuitos digitales en dispositivos lógicos programables. Existen, además, las herramientas EDA orientadas a la fabricación de circuitos. En el caso del diseño de hardware estas herramientas sirven para la realización de PCB ("Printed Circuit Boards" o placas de circuito impreso), o para desarrollar circuitos integrados de aplicación especifica como ASIC ("Application Specific Integrated Circuits"). Este ciclo de diseño de hardware se muestra en la figura 2.1.3

Las principales características y finalidad de algunas herramientas EDA que intervienen en el diseño de circuitos son:

- Lenguajes de descripción de circuitos.

Son lenguajes mediante los cuales es posible describir el funcionamiento y estructura de un circuito eléctrico o digital. La descripción puede ser mediante bloques donde se muestra la arquitectura del diseño, o de comportamiento, es decir, se describe el funcionamiento del circuito en vez de especificar los elementos de los que está compuesto.

- Diagramas esquemáticos.

Es la forma clásica de describir un diseño electrónico y la más extendida ya que era la única usada antes de la aparición de las herramientas de EDA. La descripción está basada en un “plano” donde se muestran los diferentes componentes utilizados en el circuito.

- Grafos y diagramas de flujo.

Es posible describir un circuito o sistema mediante diagramas de flujo, redes de Petri, máquinas de estados, etc. En este caso sería una descripción gráfica y además sería comportamental porque no es una descripción mediante componentes.

- Simulación de eventos.

Estas herramientas se usan para la simulación de circuitos a grandes rasgos. En esta simulación los componentes más importantes son los elementos de alto nivel como discos duros, buses de comunicaciones, memorias RAM, etc.

- Simulación funcional.

Bajando al nivel de compuertas digitales se pueden realizar una simulación funcional de las mismas. Este tipo de simulación comprueba la operación de circuitos digitales a partir del comportamiento lógico de sus elementos con el fin de comprobar el funcionamiento en conjunto del circuito mediante unos estímulos dados. Similar a lo que se realiza en un laboratorio.

- Simulación digital.

Esta simulación también exclusiva de los circuitos digitales, es como la anterior con la diferencia de que se tienen en cuenta los retardos de propagación de cada compuerta. Es una simulación muy cercana al comportamiento real del

circuito y prácticamente garantiza el funcionamiento correcto del circuito en cuestión. En las herramientas EDA este tipo de simulación se conoce como análisis temporal o timing.

- Simulación eléctrica.

Es la simulación de más bajo nivel donde las respuestas del sistema se verifican al nivel de transistor. Sirven tanto para circuitos analógicos como digitales y su respuesta es prácticamente idéntica a la realidad ya que se prueban retardos de tiempo, niveles de voltaje, disipación de potencia, etc.

- Diseño de PCB

Con estas herramientas es posible realizar el trazado de pistas para la fabricación de placas de circuitos impresos.

- Diseño de circuitos integrados

Son herramientas EDA que sirven para la realización de circuitos integrados. Las capacidades gráficas de estas herramientas permiten la realización de las diferentes máscaras que intervienen en la realización de éstos.

- Diseño con dispositivos programables.
    - Estas herramientas facilitan la programación de dispositivos, ya sean PAL, PLD, CPLD o FPGA. Lenguajes de Descripción de Circuitos.

Para la automatización del diseño electrónico se utilizan herramientas EDA.

2.1.4 Ventajas de la metodología de diseño que usa herramientas EDA

Entre las ventajas de la metodología de diseño con el empleo de herramientas EDA está la reducción del diseño, la posibilidad de dividir un proyecto en módulos que se desarrollan por separado, la independencia del diseño con respecto a la tecnología, la posibilidad de la reutilización de los diseños, la optimización de los circuitos y las simulaciones posibles con las herramientas.

2.2 Lenguajes de Descripción de Hardware (HDL)

HDL fue diseñado para llenar un número de necesidades en los procesos de diseño. Primeramente, permite la descripción de la estructura de un diseño, es decir como es descompuesto dentro de sub-diseños, y como aquellos sub-diseños son interconectados. Segundo, permite a la especificación de la función del diseño utilizar formas de lenguajes de programación familiar. Tercero, como resultado, permite al diseño ser simulado antes de ser fabricado, así que los diseñadores

pueden rápidamente comparar alternativas y pruebas para corregir sin los retardos y costos del prototipo inherentes del hardware.

2.2.1 Características de los lenguajes de descripción

Los lenguajes HDL (Hardware Description Language) permiten realizar el primer paso de la metodología del diseño descendente. Se describen en un lenguaje de alto nivel el comportamiento requerido del circuito a diseñar. Esta descripción se puede hacer mediante tablas de verdad, lista de transiciones de estados, ecuaciones lógicas. Con base a la descripción, el programa realiza los siguientes pasos:

- Sintetiza y simplifica las ecuaciones lógicas.
- Simula las ecuaciones.
- Sintetiza el circuito lógico.
- Simula el circuito lógico.
- Sintetiza el archivo para programar un PLD.

2.2.2 Ventajas de los lenguajes de descripción

Una metodología de diseño que utiliza un HDL posee varias ventajas sobre la metodología tradicional de diseño a nivel compuerta. Algunas de estas ventajas son listadas a continuación:

- Es posible verificar el funcionamiento del sistema dentro del proceso de diseño sin necesidad de implementar el circuito.
- Las simulaciones del diseño, antes de que éste sea implementado mediante compuertas, permiten probar la arquitectura del sistema para tomar decisiones en cuanto a cambios en el diseño.
- Las herramientas de síntesis tienen la capacidad de convertir una descripción hecha en un HDL, VHDL por ejemplo, a compuertas lógicas y, además, optimizar dicha descripción de acuerdo a la tecnología utilizada.
- Esta metodología elimina el antiguo método tedioso de diseño mediante compuertas, reduce el tiempo de diseño y la cantidad de errores producidos por el armado del circuito.
- Las herramientas de síntesis pueden transformar automáticamente un circuito obtenido mediante la síntesis de un código en algún HDL, a un circuito pequeño y rápido. Además, es posible aplicar ciertas características

al circuito dentro de la descripción para afinar detalles (retardos, simplificación de compuertas, etc.) en la arquitectura del circuito y que estas características se obtengan en la síntesis de la descripción.

- Las descripciones en un HDL proporcionan documentación de la funcionalidad de un diseño independientemente de la tecnología utilizada
- Un circuito hecho mediante una descripción en un HDL puede ser utilizado en cualquier tipo de dispositivo programable capaz de soportar la densidad del diseño. Es decir, no es necesario adecuar el circuito a cada dispositivo porque las herramientas de síntesis se encargan de ello.
- Una descripción realizada en un HDL es más fácil de leer y comprender que los netlist o circuitos esquemáticos.

Entre otras ventajas, se pueden mencionar las siguientes:

- EL programa HDL es el mismo así cambie la tecnología, Ejemplo: FPGA, transistores 2.5 mm., 1.2 mm.
- Facilita la comunicación entre los diseñadores.
- Facilita el uso de las partes de un diseño en otros. (Reutilización)
- Es posible verificar el funcionamiento del sistema dentro del proceso de diseño sin necesidad de implementar el circuito.
- Las simulaciones del diseño, antes de que este sea implementado, permiten probar la arquitectura del sistema para tomar decisiones en cuanto a cambios en el diseño.
- Las herramientas de síntesis tienen la capacidad de convertir una descripción hecha en un HDL, VHDL por ejemplo, a compuertas lógicas y además, optimizar dicha descripción de acuerdo a la tecnología utilizada.
- Las descripciones en un HDL proporcionan documentación de la funcionalidad de un diseño independientemente de la tecnología utilizada.

2.2.3 Lenguajes HDL más populares

En la actualidad existen diversas herramientas de diseño para integrar sistemas de gran complejidad. Los lenguajes de descripción de hardware constituyen una opción de diseño de soluciones de sistemas electrónicos.

2.2.3.1 Lenguaje ABEL

El lenguaje ABEL es el más utilizado en los PLD’s. El lenguaje ABEL facilita la programación de PLD’s combinatorios y secuenciales. Un circuito en ABEL se

puede describir en forma de ecuación lógicas, tabla de verdad o en transición de estados.

El programa ABEL cumple con los siguientes pasos:

1. Verifica si existen errores en la sintaxis del programa fuente.
2. Simplifica o sintetiza las ecuaciones según sea el caso.
3. Simula las ecuaciones.
4. Puede escoger el dispositivo que mejor se adapte, o verificar si el dispositivo especificado sí se adapta a la aplicación.

Genera el archivo JEDEC para la programación del PLD.

2.2.3.2 Lenguaje VHDL

El VHDL (Very High Speed Integrated Circuit Hardware Descrption Lenguaje) es un lenguaje de descripción y modelado diseñado para describir en forma entendible la funcionalidad y la organización del hardware de los sistemas digitales y otros componentes. VHDL maneja una sintaxis amplia y flexible. El lenguaje VHDL permite el diseño top-down o en otras palabras; modelar los bloques de alto nivel, simularlos y adecuar la funcionalidad en alto nivel antes de llegar a los niveles bajos de abstracción en la implementación del diseño.

El lenguaje VHDL está creado específicamente para el diseño de hardware, es decir, podremos implementar con él multitud de circuitos lógicos, tanto combinacionales como secuenciales. Este lenguaje también nos permite describir elementos más complejos, como CPU (Unidad Central de Procesamiento), manejar ficheros, retrasos en el tiempo, etc. pero no siempre se puede implementarlos; tan sólo, y en algunos casos, se llegará a la simulación.

Un programa en VHDL consta de dos partes. La primera, la entidad, nos sirve para relacionar nuestro diseño con el mundo exterior, es decir, analizamos lo que tratamos de crear como una "caja negra", de la que sólo conocemos sus entradas, sus salidas y la disposición de las mismas. La segunda parte, la arquitectura, describe como trata el circuito la información correspondiente a las entradas para obtener las salidas.

No existe unanimidad en lo que se refiere al establecimiento de las distintas fases del diseño de un sistema digital complejo, aunque la necesidad de sistematizar esta metodología ha llevado al establecimiento y progresiva aceptación de algunas propuestas. En la figura 2.2.1 se representa un esquema bastante bien aceptado en el que divide el diseño en siete niveles o formas de caracterización, también denominados niveles de abstracción por cuanto establecen la cantidad de información que se especifica de un circuito o el nivel de detalle en que se encuentra una descripción respecto de su implementación física.

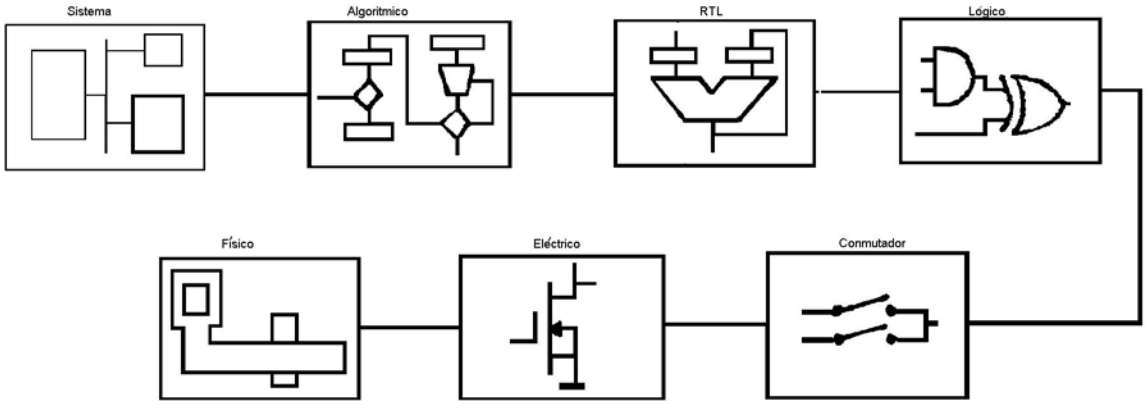


Figura 2.2.1 Esquema de división del diseño o formas de caracterización.

- Nivel sistema.

Describe el sistema como un conjunto de módulos semi-autónomos y cooperantes, cuya interacción se analiza para obtener un conjunto óptimo. No se especifica la forma de realizar cada uno de los módulos.

- Nivel algorítmico, funcional o de comportamiento.

Cada módulo del nivel superior se define mediante un algoritmo en un lenguaje de alto nivel (HLL). Dada la naturaleza paralela del hardware, en el que varios procesos pueden necesitar un mismo recurso (por ejemplo, un bus de datos) o en el que se deben sincronizar procesos independientes, se utilizan instrucciones muy similares a las de los lenguajes de programación concurrentes, como por ejemplo, semáforos y regiones críticas.

- Nivel RTL (Register Transfer Level) o de flujo de datos (Data-Flow).

En él se describe el sistema mediante diagramas de transferencias entre registros, tablas de verdad o ecuaciones lógicas. Se le concede más importancia a lo que hace el sistema que a como lo hace. Los elementos básicos de este nivel son registros, memorias, lógica combinacional y buses. Se distingue entre elementos con capacidad de almacenamiento y elementos sin ella.

- Nivel lógico.

Consiste en la descripción del sistema mediante la interconexión de bloques básicos, como puertas lógicas y biestables. No se realiza una descripción del comportamiento, sino de la estructura del mismo. Si se incluyen además otro tipo de bloques, en general a este nivel se le suele denominar también estructural.

- Nivel conmutador.

Las compuertas se sustituyen por transistores considerados como conmutadores ideales que toman los valores cero o uno. La descripción es, por lo tanto, básicamente la misma que en el nivel anterior con la única diferencia de que en algunas tecnologías, como por ejemplo CMOS, surgen nuevos tipos de

circuitos porque los conmutadores pueden ser bi-direccionales mientras que las compuertas son unidireccionales.

- Nivel eléctrico.

Se describe el sistema mediante modelos reales del transistor, con sus diferentes parámetros eléctricos.

- Nivel físico.

Está constituido por la descripción geométrica o simbólica de las máscaras que se emplean para la fabricación del circuito.

En la figura 2.2.2 se representa el método de diseño conceptual propuesto por Lemmert y Nebel, que utiliza un lenguaje de descripción del hardware (HDL) a través de los distintos niveles. El diseño del circuito o sistema comienza de la forma más generalizada posible, habitualmente con una especificación algorítmica en un lenguaje de alto nivel. En cada nivel se refina la descripción correspondiente y se compara con la de nivel superior. Este método implica la utilización de herramientas de diseño asistido por computador para simulación y prueba en cada nivel, y por ello los lenguajes deben proporcionar compatibilidad y consistencia entre niveles. De esta manera se trabaja de forma flexible y se logra que en un determinado momento coexistan descripciones de diferentes niveles que se pueden probar conjuntamente.

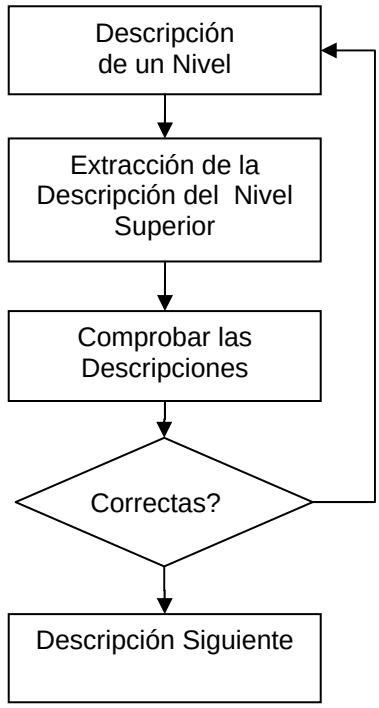


Figura 2.2.2 Método de diseño conceptual.

2.2.4 Sentencias concurrentes y secuenciales

Para iniciar correctamente en el aprendizaje y manejo de VHDL es importante comprender desde un principio la diferencia entre concurrente y secuencial.

El concepto de concurrencia, se ve claramente graficado en los circuitos electrónicos donde los componentes se encuentran siempre activos, existiendo una asociación intrínseca, entre todos los eventos del circuito; ello hace posible el hecho de que si se da algún cambio en una parte del mismo, se produce una variación (en algunos casos casi instantánea) de otras señales.

Este comportamiento de los circuitos reales obliga a que VHDL soporte estructuras específicas para el modelado y diseño de este tipo de especificaciones de tiempos y concurrencias en el cambio de las distintas señales digitales de los diseños.

Por el contrario, las asignaciones secuenciales, son más bien propios de los SDL (Soft Design Language) en los que la programación tiene un flujo natural secuencial, siendo propio de este tipo de eventos las sentencias **case**, **if**, **while**, **loop**, etc más propias de estas sintaxis.

Las construcciones concurrentes del lenguaje son usadas dentro de estructuras concurrentes, por ejemplo una arquitectura tiene una naturaleza eminentemente concurrente (es decir que está activo todo el tiempo), mientras que el cuerpo de un **process** es en principio eminentemente secuencial. La asignación de eventos secuenciales dentro de una estructura concurrente se ejecutará de forma concurrente, es decir, al mismo tiempo que las demás sentencias.

VHDL soporta con este motivo, tres tipos de objetos, las variables, las constantes y las señales . Como las variables y las señales pueden variar su valor mientras ejecutamos un programa, serán éstas las encargadas de almacenar dichos datos, asimismo serán los portadores de la información. Únicamente las señales pueden tener la connotación de globalidad dentro de un programa, es decir, que pueden ser empleadas dentro de cualquier parte del programa a diferencia de las variables que solo tienen sentido en el interior de un **process** .

Los **process** son estructuras concurrentes constituidas por sentencias de ejecución secuencial, esto provocará que dentro de un **process** nos encontremos con sentencias similares a las de los SDL (Lenguajes de Descripción de Software) que nos llevan a emplear VHDL como si de otro lenguaje común se tratara. Dentro de un **process** nos podemos encontrar con la declaración y utilización de las variables como parámetros locales al **process**.

De ejecución secuencial, las variables evalúan su valor dentro del cuerpo del proceso de forma inmediata, sin consumir tiempo de ejecución, pero como están dentro de un **process**, que es una estructura concurrente, este valor no será asumido, sino hasta el final de la ejecución de dicho **process**.

El siguiente ejemplo de asignación concurrente para señales, figura 2.2.3.

```
w<= not a;
x <= a and b;
y <= c and w;
z <= x or y;
```

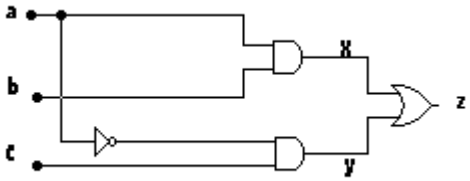


Figura 2.2.3 Ejemplo de Asignación concurrente para señales

Al producirse un cambio en la parte derecha de la estructura de asignación (**<= NOT a**) de alguna de las sentencias, la expresión es evaluada de nuevo en su totalidad, con la siguiente asignación del nuevo valor a la señal de la izquierda. Esto puede provocar que múltiples asignaciones en el cuerpo de una arquitectura se activen simultáneamente, como por ejemplo, en la figura 2.2.3, la cual corresponde al código situado a su izquierda.

2.3. Síntesis lógica

Permite implementar físicamente un diseño gracias a la utilización de herramientas de síntesis.

Usa solamente una parte reducida del juego de instrucciones del lenguaje VHDL.

Numerosas construcciones utilizables en modelos o en simulación, no pueden ser sintetizadas.

Las construcciones soportadas varían según las herramientas.

Portabilidad.

- Entre herramientas de síntesis
- Entre herramientas materiales.

Presentación general.

VHDL permite en particular:

- Definir los puertos de entrada/salida del conjunto lógico descriptivo (Entity). Señales simples o sobre la forma de bus.
- Definir un modelo de comportamiento sintetizable (Architecture) usando el juego de instrucciones soportado por la herramienta de síntesis.

  Estilo de escritura RTL (Register Transfer Logic)

- Unir los diferentes módulos descritos separadamente (gestión de jerarquía, VHDL estructural).

Informaciones previas sobre la sintaxis:

- Minúsculas y mayúsculas no son diferenciados en VHDL. Un objeto puede indiferentemente ser llamado:

  MODULE, Module ó module

  Es lo mismo para las palabras clave.

- Los comentarios pueden ser colocados en cualquier lugar dentro del código. Ellos no afectan en nada los resultados de síntesis o de simulación. Ejemplo:

```
 --  Esto es un comentario
Architecture ARQUI of EJEMPLO is – esto también
begin
```

- Los archivos VHDL llevan la extensión “VHD”.

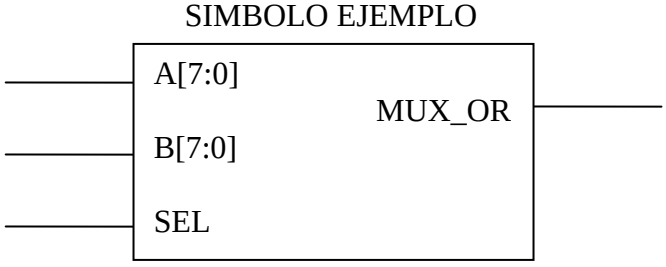
La pareja entidad / arquitectura:

Entidad: Porción del código que permite definir en particular las entradas y salidas.

Ejemplo:

```
entity EJEMPLO is
        port (
        A,B, : in bit_vector(7 downto 0) ;
        SEL : in bit ;
        MUX_OR : out bit_vector(7downto 0) ;
End EJEMPLO ;
```

Equivalencia esquemática



Entidad: Sintaxis de declaración:

```
Entity EJEMPLO is
        Port (
```

```
        Lista de puertos de entrada / salida comprendiendo: nombre_de_señal: modo y tipo.
                ) ;
end [EJEMPLO] ; -- los corchetes "[ ]" indican que es opcional
-- usar de nuevo el nombre de la entidad después de la palabra clave "end".
```

- El nombre dado a la entidad puede ser cualquiera (excepto las palabras reservadas).
- Dar de preferencia el mismo nombre a la entidad y al fichero VHDL (en este caso EJEMPLO.vhd).
- La lista de puertos está comprendida entre dos paréntesis y seguida de un punto y coma.
- Nombre de la entidad opcional después de la palabra "end".

Arquitectura: Porción de descripción del comportamiento del dispositivo a sintetizar.

- Está asociada a una entidad. (dentro de un mismo fichero).
- Posee una parte declaratoria y una parte operatoria.

```
Architecture ARQUI of EJEMPLO is
-- parte declaratoria
signal MUX_OUT : bit_vector(7 downto 0) ; -- señal interna
-- parte operatoria
    begin
-- asignaciones condicionales de señales con "when…else".
-- el signo "<=" es el símbolo de asignación de un valor a una señal
    MUX_OUT <= A when SEL='0'  else B ;
    MUX_OR <= '1' when MUX_OUT /= "00000000" else '0' ;
End [ARCHI] ;
```

Arquitectura: Informaciones complementarias

La parte declaratoria puede contener:

- Declaraciones de señales internas.

  (que no sean I/O)
- Declaraciones de componentes.

  (Gestión de la jerarquía, VHDL estructural).
- Declaraciones de constantes.
- Declaraciones de tipos de objetos.

  (Utilizados en particular para las máquinas de estado).
- Declaraciones de sub-programas.

  (Funciones y procedimientos).

Recapitulación: Archivo “EJEMPLO.vhd”.

```
entity EJEMPLO is
        port (
                A,B : in bit_vector(7 downto 0) ;
                SEL : in bit ;          -- El signo “;” se utiliza como separador entre
                MUX_OR : out bit  -- dos declaraciones de señales y no debe
                                        -- aparecer después de la última señal declarada
end [EJEMPLO] ;
architecture ARQUI of EJEMPLO is
-- parte declaratoria
signal MUX_OUT : bit_vector(7 downto 0) ;   --- señal interna
-- parte operatoria
begin
MUX_OUT <= A when SEL='0'  else B ;
MUX_or <='1' when MUX_OUT /= “00000000”  else ‘0’ ;
```

Otro EJEMPLO: mux2_1.vhd

```
entity MUX2_1 is
        port (
                A_IN, B_IN :    in bit ;   --      La entidad tiene el mismo nombre
                SEL :           in bit ;   --      que el fichero VHDL
                SALIDA :        out bit
                ) ;
end [MUX2_1] ;
architecture ARCHI of MUX2_1 is
-- parte declaratoria vacía en este ejemplo
-- parte operatoria
 begin
 SALIDA <= (A_IN and not (SEL)) or ( B_IN and SEL) ;
 -- “and”, “not”, “or” son operadores lógicos predefinidos del lenguaje VHDL
```

Objetos que se pueden manipular en VHDL y sus tipos:

3 clases principales de objetos:

- SEÑALES: Similares a las señales encontradas en los esquemas. Los puertos declarados dentro de una entidad son señales. Pueden ser declaradas como “bus”.
- CONSTANTES: permiten definir valores permanentes.
- VARIABLES: utilizados solamente dentro de los “PROCESS” (instrucciones secuenciales).

Una declaración objeto comprende:

- CLASE: señal, constante o variable.
- NOMBRE: a elección del creador (excepto palabras reservadas).
- MODO: (solamente señales) : in, out, inout, buffer.

- TIPO: bit, bit_vector, bolean, integer…

Tipos de objetos que se pueden manipular en VHDL

Tipos predefinidos principales:

- BIT: puede tomar el valor ’0’ ó ‘1’.
- BIT VECTOR: grupo de bits (bus).

  El valor binario de un bit_vector está definido entre comillas dobles. EJEMPLO:

```
Signal A : bit_vector(7downto 0) ;
-- El bit de más peso (MSB) está a la izquierda
-- En nuestro caso es el bit 7.
Begin
A <= “01011010” ;   -- equivalente a : A <= X”5A” ;
-- X”valor” indica valor hexadecimal.
```

- BOOLEAN: puede tomar los valores TRUE o FALSE.
- INTEGER: Valores entero codificado sobre 32 bits. (de -2.147.483.648 a + 2.147.483.647)
  Un entero puede estar limitado en su declaración a fin de evitar su codificación sobre 32 bits. EJEMPLO:

```
signal VALEUR : integer range 0 to 255 ;
Begin
VALEUR <= 143 when INIT = ‘1’  else 33 ;
```

2.3.1 Los operadores

Clasificación de los Operadores

- Operadores frecuentemente utilizados en síntesis:
- Operadores lógicos pre-definidos:
  - and, or, nand, nor, xor y not

Operan sobre todos los objetos de todas las clases (señales, constantes, variables) y de tipo:

- bit
- bit_vector
- std_logic, std_ulogic
- std_logic_vector, std_ulogic_vector
- boolean

Los operandos deben ser del mismo tipo y contener el mismo número de bits.

Ejemplos de los operadores

Operadores frecuentemente utilizados en síntesis:

Operadores lógicos pre-definidos:

Ejemplo de utilización (1):

```
entity OPE is
  port ( A, B, C : in bit;

 S :       out bit) ;

end OPE;

architecture ARCHI of OPE is
  begin
    S <= (A and B) and not(C);
end ARCHI;
```

Operadores relacionales: (cont.)

Los operandos deben ser del mismo tipo, pero el número de bits comparados puede ser diferente.

Operadores corrientemente utilizados en síntesis:

Operadores relacionales:

Comparación de vectores: La comparación se hace comparando bit a bit los dos

vectores comenzando por el MSB.

Los resultados pueden sorprender si los vectores no son del mismo número de bits.

Ejemplo:

```
signal REG : std_logic vector(4 downto 0);

signal CNT : std_logic_yector (3 downto 0);

begin                                          ^'^

-- tenemos el derecho de escribir :

     -- REG       CNT
    -- "01111" > "0100" resultado poco sorprendente

--  y   "01111" < "1000" resultado muy sorprendente

--  El MSB de CNT es superior al MSB de REG,
--  el vector CNT es considerado como superior.
```

Operadores relacionales: (cont.)

• La utilización de packages estandarizados permite efectuar operaciones relacionales entre objetos de tipo INTEGER y otro de tipo STD_LOGIC_VECTOR.

• En las herramientas FOUNDATION, estos packages se encuentran en la biblioteca "IEEE".

Estos packages son : STD_LOGIC_UNSIGNED, y STD_LOGIC_ARITH.

Para ser visibles, deben se declarados al inicio del archivo.

Operadores relaciónales: Informaciones complementarias

El resultado de una comparación es de tipo "boolean" puede tomar solo los valores TRUE o FALSE.

Ejemplo :

```
signal IGUAL, VENTANA : bolean;

signal COUNT : integer range O to 31;

begin

IGUAL <= (COÜNT = 27) ;                    ^

VENTANA <= (COUNT >= 13) and (COÜNT <= 25) ;
```

-- operación lógica entre dos “booleans".

Nota: Los operadores lógicos relacionales distintos de “=” y “/=” son frecuentemente implementados en FPGA en forma de funciones aritméticas cableadas.

Operadores relacionales y la implementación de funciones aritméticas;

• La comparación (distinta de " = " o “ /= ") de vectores puede resultar en la utilización de funciones aritméticas cableadas, y por esto no serán optimizadas después de la síntesis. Prudencia en la utilización de los operadores “<”, “<=”, “>”, “>=”. Dos porciones de código pueden tener un comportamiento equivalente, pero los resultados de síntesis pueden ser muy diferentes.

Ejemplo :

```
signal CNT : std_logic_vector (7 downto 0);

begin
TOTO <= “1” when (CNT < "10000000") else 'O';

-- podemos escribir

-- TOTO <= '1’ when CNT(7)='O' else 'O’;

- o mas simple TOTO <= not(CNT(7));
```

Operadores aritméticos :

+ (suma) - (resta)

*(multiplicación) / (división)

** (potencia) mod rem abs

Operan sobre objetos de tipo INTEGER.

Pueden igualmente operar sobre STD_LOGIC_VECTOR utilizando los packages STD_LOGIC_UNSIGNED y STD_LOGIC_ARITH.

• Restricciones: La mayoría de las herramientas de síntesis sólo autorizan las operaciones de multiplicación y división entre CONSTANTES, o una CONSTANTE potencia de 2 y una SEÑAL.

2.4. CUPL

En el medio electrónico hay diferentes herramientas de software para programar PLD. Todos estos tienen semejanzas compartidas y sus diferencias distintivas. Uno de los compiladores disponibles de alto nivel de uso difundido actualmente es CUPL.

CUPL es una herramienta de programación para PLDs y su nombre proviene de la sigla en inglés de Compiler Universal Programmable Logic , la cual traduce Compilador Universal para Lógica Programable. Este compilador ofrece varias características que permiten desarrollos basados en la metodología top-down y puede generar archivos de programación para una gran variedad de dispositivos programables.

La programación en este software se efectúa mediante la creación de un archivo de texto que contiene el código para la programación del dispositivo. Este archivo tiene tres partes básicas: el encabezado, la declaración de los pines de entrada y las definiciones lógicas.

Generalmente se emplea un archivo como el que se muestra en la lista 2.4.1 para dar inicio al diseño lógico y tener una forma estándar para trabajar en CUPL.

```
Name   XXXXX;
Partno  XXXXX;
Date
Revision
Designer
Company
Assembly
Location

/*****************************************************************/

/*Entradas*/
pin 1 = ;       /* */
pin 1 = ;       /* */
pin 1 = ;       /* */
pin 1 = ;       /* */
```

```
/*Salidas*/
pin 1 = ;       /* */
pin 1 = ;       /* */
pin 1 = ;       /* */
pin 1 = ;       /* */

/*Variables Intermedias (Opcionales)*/
pin 1 = ;       /* */
pin 1 = ;       /* */
pin 1 = ;       /* */
pin 1 = ;       /* */

/*****************************************************************/
/* Ecuaciones Lógicas*/
/*****************************************************************/
```

Lista 2.4.1 Archivo de ejemplo de entrada en CUPL

En este archivo los comentarios los grupos de caracteres "/*" y "*/" son empleados para incluir comentarios por parte del usuario, los cuales permiten organizar el archivo de una forma comprensible y especificar la función de ciertos tipos de instrucciones.

En las siguientes secciones se indicarán algunas características a tener en cuenta para programar un PLD en CUPL y la sintaxis que se debe emplear en el archivo fuente para implementar un diseño.

2.4.1 Asignación de pines

La asignación de los pines corresponde al nombramiento de los pines o patas del dispositivo con nombres descriptivos para las entradas y salidas. Los nombres se pueden asignar de forma libre y corresponden a las variables que se emplean para definir las ecuaciones lógicas.

La asignación de pines se puede hacer de forma individual o grupal. En la tabla 2.4.2 se indica la sintaxis que se emplea en CUPL para asignar los pines de un dispositivo programable. Observe que cada asignación finaliza con un punto y coma (;).

Sintaxis General	Sintaxis Abreviada
Pin 1 = Nombre;	Pin [2,3] = [Nombre, Nombre2];
Pin 2 = !Nombre;	Pin [2,3] = ![Nombre, Nombre2];
Pin 3 = !SET;	Pin [2..3] = [Q0..3];

Tabla 2.4.2 Asignación de Terminales

El signo "!" en la asignación de pines indica que la variable se complementa. Este signo se emplea generalmente para declarar variables activas en bajo.

2.4.2 Definición de variables intermedias

Las variables intermedias corresponden a variables asignadas a una ecuación lógica pero que no representan un pin en el dispositivo. Generalmente estas variables se utilizan cuando se requiere manejar varias variables de entradas y salidas.

El objetivo de declarar variables intermedias, consiste básicamente en reducir el tamaño de las ecuaciones lógicas asignadas a los pines de salida y permitir organizar el archivo de entrada de una forma comprensible. Su uso no es obligatorio en el archivo de entrada para CUPL.

2.4.3 Definición de ecuaciones lógicas

Las ecuaciones lógicas corresponden a las expresiones lógicas que relacionan los pines de entrada y salida y, en el archivo fuente se ubican después de la asignación de pines y variables intermedias.

Las ecuaciones lógicas deben tener cierta sintaxis para que el programa interprete las operaciones lógicas. En la tabla 2.4.3 se relacionan los operadores lógicos con lo cuales se construyen las expresiones lógicas junto con la sintaxis que exige CUPL.

Operador	Función	Formato de CUPL	Formato Convencional
&	AND	A&B	A·B
#	OR	A#B	A+B
!	NOT	!A	A'
$	XOR	A$B	A⊕B

Tabla 2.4.3 Sintaxis de CUPL para operaciones lógicas

La sintaxis general de las ecuaciones lógicas en *CUPL* es la siguiente:

[!] var [.ext] = exp;

2.4.4 Extensiones de variables

Las extensiones son atributos que se agregan a las variables en las ecuaciones lógicas y la manera de emplearlas en el archivo fuente es mediante la sintaxis: "V.E", donde V es la variable lógica y E es la extensión.

Las extensiones para las variables son empleadas para definir funciones que dependen de la configuración física de las salidas del PLD. Generalmente el tipo de salidas que se pueden configurar en un PLD son salidas combinacionales, secuenciales y tri-estado, entre otras. En la figura 2.4.1 se muestra un diagrama ilustrativo del tipo de salidas que se pueden configurar en CUPL según la extensión seleccionada, y un ejemplo sobre su utilización.

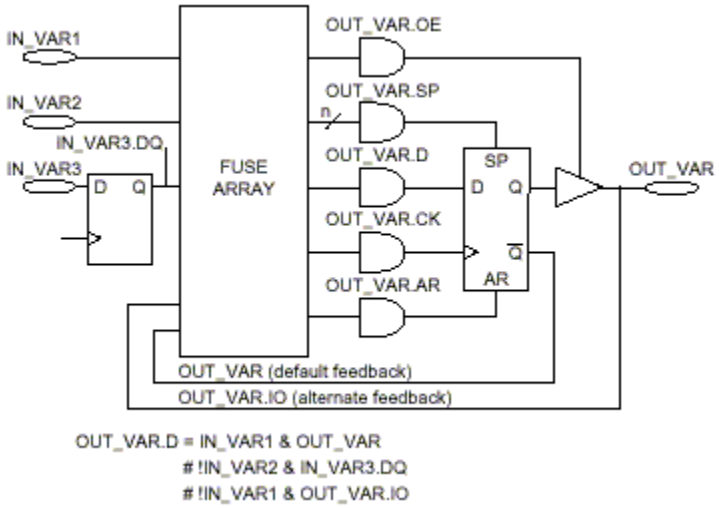


Figura 2.4.1 Circuito Ilustrativo del tipo de salidas configurables en un PLD
(Tomado de ATMEL - WinCUPL User's Manual)

Teniendo en cuenta que las variables lógicas pueden incluir extensiones, la sintaxis que se debe emplear para las ecuaciones lógicas es la siguiente:

[!] var [.ext] = exp;

Donde “var” es la variable lógica, “ext” es la extensión y “exp” es la expresión lógica. Un sencillo ejemplo usando el software para creación de CUPL servirá para ilustrar el proceso.

En CUPL existe otro tipo de sintaxis para simplificar un poco las ecuaciones lógicas. Una de ellas consiste en la utilización de los operadores de forma consecutiva a varias variables. En la tabla 2.4.4 se indica como se pueden representar las expresiones lógicas de forma abreviada.

Forma Convencional	Forma Abreviada
A3 & A2 & A1 & A0	[A3, A2, A1, A0]:&
B3 # B2 # B1 # B0	[B3..B0]:#
C3 $ C2 $ C1 $ C0	[C3, C2, C1, C0]:$

Tabla 2.4.4 Forma abreviada para representar expresiones lógicas con un mismo operador.

Existe otro tipo de sintaxis similar a la anterior que define rangos de datos. Inicialmente se debe definir un campo de la siguiente forma:

FIELD entrada = [A3..A0];

Luego se escribe el rango de la ecuación de la forma:

salida = entrada:[C..F];

Esta ecuación equivale a escribir la siguiente expresión:

salida = entrada:C # entrada:D # entrada:E # entrada:F;

2.4.5 Definición de alternativa de las salidas

En CUPL existen otras formas alternativas de definir las salidas. Entre ellas existen las tablas de verdad, máquinas de estado y sentencias condicionales. A continuación se indica la sintaxis de cada uno de estos tipos de declaración de salidas lógicas.

2.4.5.1 Tablas de verdad

Como su nombre lo indica este tipo de sintaxis agrupa la información sobre la asociación de entradas y salidas en forma de tabla. Para declarar una tabla de verdad inicialmente se declaran las entradas y salidas. Después de ello se asignan los valores uno a uno de las entradas y salidas. En la lista 2.4.5 se indica las sintaxis para un decodificador de hexadecimal a BCD.

```
FIELD Entrada = [Ent3..0];
FIELD Salida = [Sal3..0];
TABLE Ent => Sal {
    0=> 00; 1=>01; 2=>02; 3=>03;
    4=>04; 5=>05; 6=>06; 7=>07;
    8=>08; 9=>09; A=>10; B=>11;
    C=>12; D=>13; E=>14; F=>15;
}
```

Lista 2.4.5 Sintaxis en CUPL para crear Tablas de Verdad

2.4.5.2 Máquinas de estado

Este tipo de definición permite declarar la relación entradas y salidas mediante la definición de máquinas de estado. La sintaxis empleada para este tipo de definición se ilustra en las líneas de código de la lista 2.4.6

```
SEQUENCE lista_vars_estado {
 PRESENT estado_n0
   IF (condición1)NEXT estado_n1;
   IF (condición2) NEXT estado_n2 OUT
sal_n0;
   DEFAULT NEXT estado_n0;
 PRESENT estado_n1
   NEXT estado_n2;
 ...
 ...
 ...
 PRESENT estado_nn estamentos;
}
```

Lista 2.4.6 Sintaxis en CUPL para crear Máquinas de Estado

2.4.5.3 Sentencias condicionales

Las sentencias condicionales es otro tipo de sintaxis que se puede emplear en CUPL para definir el diseño lógico. Básicamente está sintaxis es muy similar a la un lenguaje de programación de alto nivel. La sintaxis que soporta CUPL se relaciona en la lista 2.4.7

```
CONDITION {
   IF expr0 OUT var;
   .
   .
   IF exprn OUT var;
   DEFAULT OUT var;
}
```

Lista 2.4.7 Sintaxis en CUPL para crear Estamentos Condicionales

2.4.5.4 Ejemplo de programación en CUPL

El código que se indica en la lista 2.4.8 corresponde al texto del archivo fuente para programar una GAL16V8, que ilustra como programar funciones lógicas básicas en CUPL.

Una vez que se tiene el archivo fuente de un diseño lógico, el paso a seguir es compilar el archivo para generar el archivo de programación JEDEC, el cual se emplea para programar el dispositivo. Durante el proceso de compilación del archivo fuente CUPL verifica la sintaxis del archivo e indica los posibles errores que puedan existir. Si el programa no detecta errores se genera el archivo .JED.

La CUPL entre sus funciones tiene un simulador con el cual se pueden comprobar las salidas. Este proceso se efectúa mediante la generación de varias entradas que comprueban los estados de las salidas para verificar las ecuaciones lógicas.

La simulación se recomienda para verificar que el diseño es correcto y que no existe ningún error. Después de ello se puede proceder a programar el dispositivo y finalmente verificar su funcionamiento.

```
Name Funciones lógicas;
Partno GAL16V8;
Revision 01;
Date 03/01/03;
Designer J.Beltran;
Company Universidad Nacional;
Location X;
Assembly X;
Device G16V8;

/*****************************************************/
/* Archivo Fuente de ejemplo en CUPL para implementar funciones lógicas */
/*****************************************************/

/* Definición de las entradas */

Pin 1 = a;
Pin 2 = b;

/* Definiciónde las salidas */

Pin 12 = inva;
Pin 13 = invb;
Pin 14 = and;
Pin 15 = nand;
Pin 16 = or;
Pin 17 = nor;
Pin 18 = xor;
Pin 19 = xnor;

/* Definición de Ecuaciones Lógicas*/

inva = !a;              /* Inversión de las entradas a y b*/
invb = !b;
and = a & b;            /* Función AND */
nand = !(a & b);        /* Función NAND */
or = a # b;             /* Función OR*/
nor = !(a # b);         /* Función NOR */
xor = a $ b;            /* Función XOR */
xnor = !(a $ b);        /* Función XOR Negada*/
```

Lista 2.4.8 Archivo fuente de ejemplo en CUPL para implementación de funciones lógicas

2.5 Especificaciones de la tarjeta Spartan-3 de Xilinx

Diagrama de especificaciones de la tarjeta

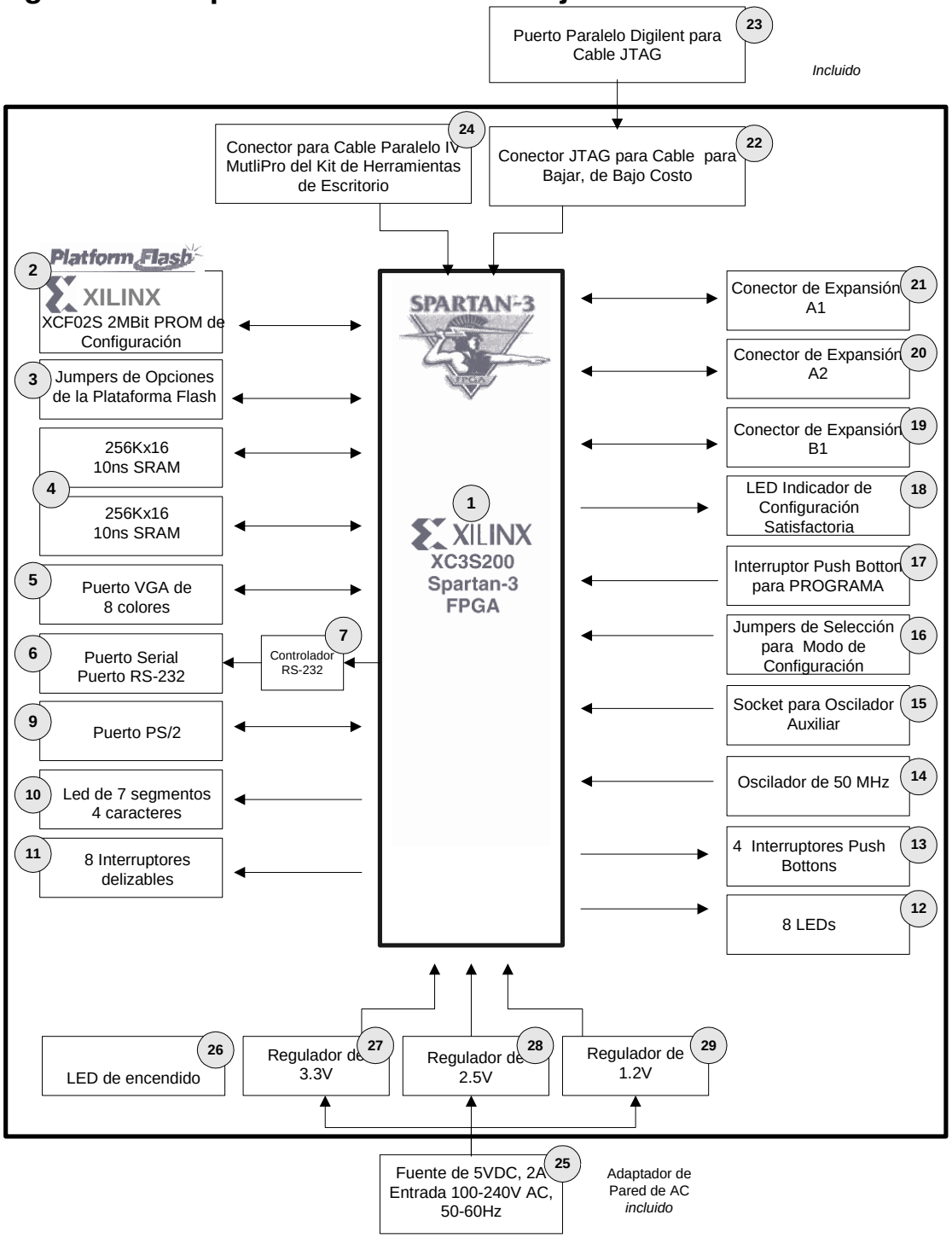


Figura 2.5.1 Diagrama a bloques de la tarjeta Spartan -3

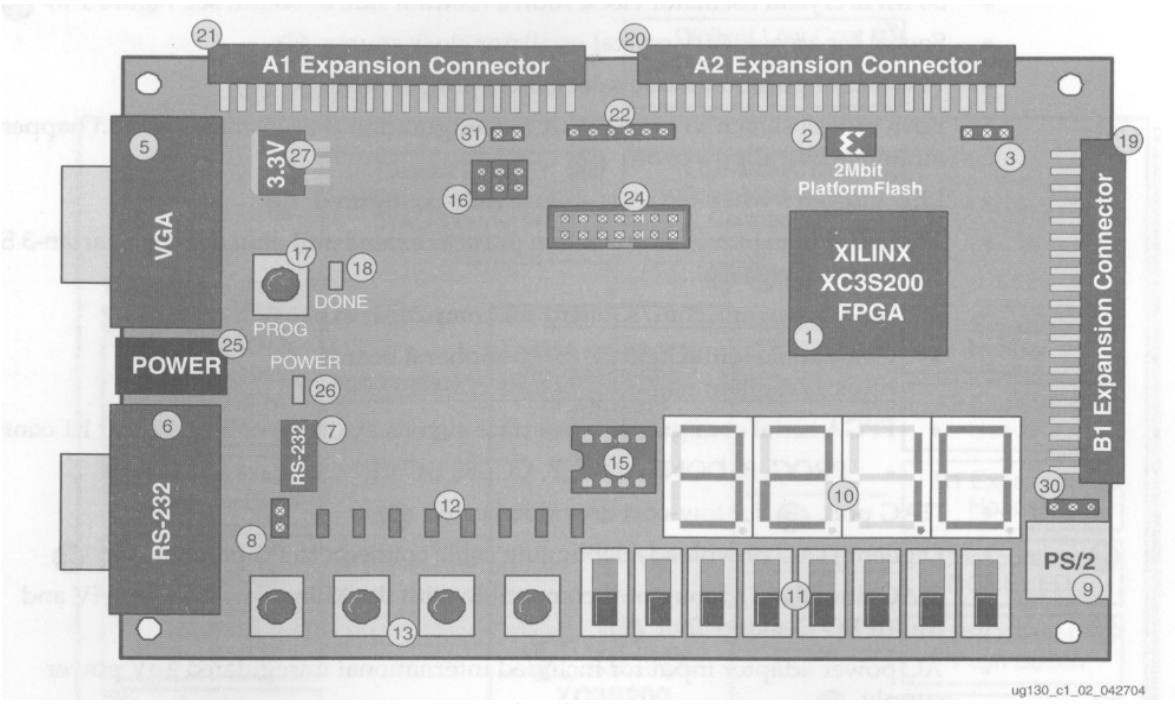


Figura 2.5.2 Distribución Física de la tarjeta Spartan-3 cara frontal

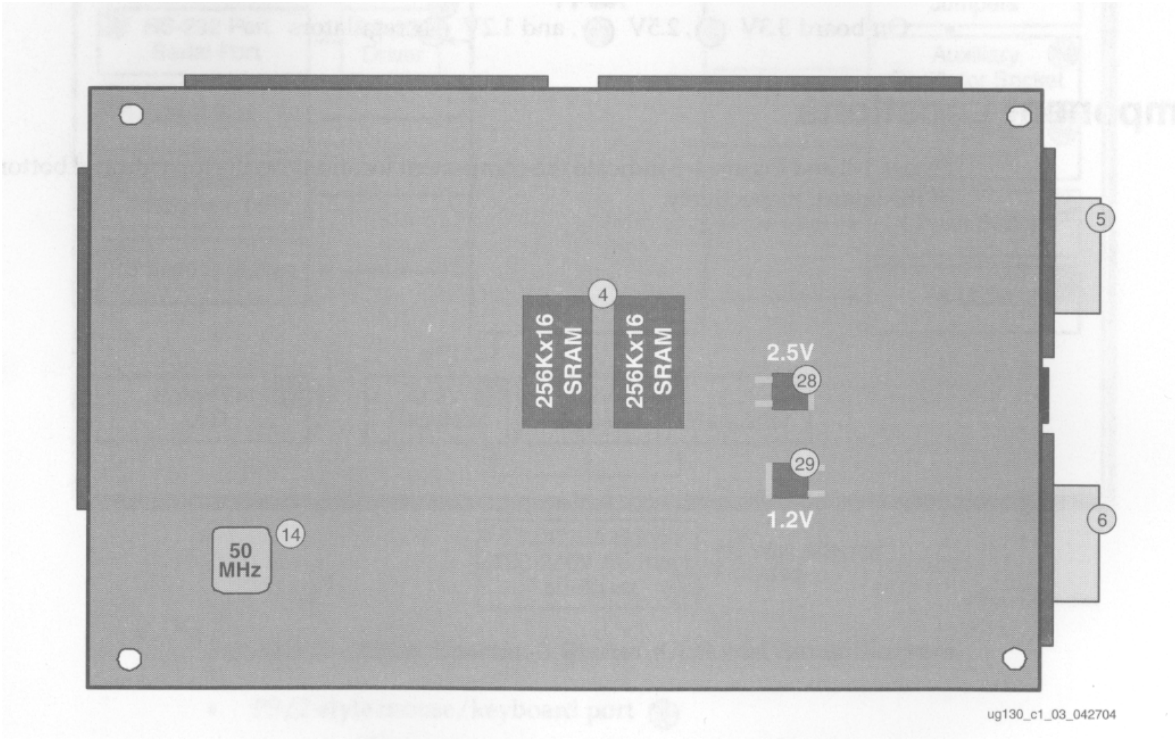


Figura 2.5.3 Distribución Física de la tarjeta Spartan-3 cara trasera

2.5.1 SRAM fast, asíncrona

La tarjeta Spartan-3 tiene un megabyte de memoria SRAM fast asíncrona, montada sobre la superficie de la tarjeta y ubicada en la parte trasera de la misma. El arreglo de memoria incluye dos circuitos 256Kx16 ISSI (IS61LV25616AL-10T de 10ns). Se muestra como (4) , de acuerdo a la figura 2.5.3)

Las formas de arreglo de SRAM pueden utilizar como un solo módulo de 256Kx32 SRAM ó dos independientes. Ambos circuitos comparten las opciones de activación de escritura (Write-Enable - WE#), activación de salida (output-enable – OE#) y direccionamiento de señales (A[17:0]). Sin embargo como cada circuito es independiente se habilitan con control-enable (CE#) independientemente.

La configuración de 256Kx32 es ideal para manejar instrucciones tipo microprocesador. Además provee mayor densidad de almacenamiento de datos útil para varias aplicaciones tales como Procesamiento de Señales Digitales (DSP), datos largos de FIFO y para buffers gráficos.

2.5.2 Cuatro displays de LED de siete segmentos

La tarjeta Spartan-3 viene integrada con 4 display de siete segmentos de un caracter o dígito cada uno ((10) de acuerdo a la figura 2.5.2), controlados y seleccionados por el usuario a través de comandos FPGA. Cada dígito comparte ocho señales de control para iluminar cada segmento de LED. Cada display tiene un ánodo de control para activar la entrada.

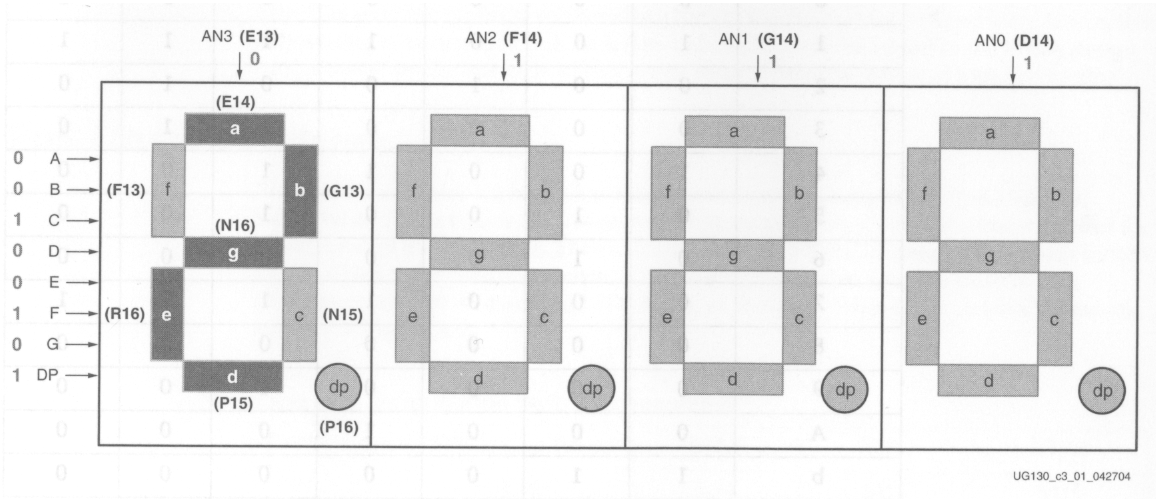


Figura 2.5.4 Control Digital de Display de Siete Segmentos

2.5.3 Interruptores y LEDs

Incluye ocho interruptores o switches deslizables sobre la superficie de la tarjeta, indicados como (11) en la figura 2.5.2. Los interruptores están etiquetados como SW7 al SW0. Los interruptores están conectados a un pin FPGA asociado como se muestra en la siguiente tabla

Interruptor	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
FPGA Pin	K13	K14	J13	J14	H13	H14	G12	F12

Cuando un interruptor se coloca en la posición Up u On, éste conecta el pin del FPGA a Vcc; un uno lógico. Cuando se coloca en la posición Down u Off, éste conecta el pin del FPGA a tierra; un cero lógico. Los interruptores manejan 2 ms de salto mecánico y no hay circuitería de rebote, aunque esta circuitería se puede configurar fácilmente programando el FPGA. Tiene una resistencia nominal en serie que provee una protección en la entrada.

Interruptores de contacto (Push Button)

La tarjeta incluye 4 interruptores de contacto o push button indicados en la figura 2.5.2 como (13). Estos interruptores están etiquetados como BTN3 al BTN0. Cada interruptor se conecta a un pin del FPGA como se indica en la siguiente tabla.

Interruptor de contacto	BTN3(User Reset)	BTN2	BTN1	BTN0
FPGA Pin	L14	L13	M14	M13

Al presionar cada interruptor conecta el pin del FPGA a VCC o genera un uno lógico. Otra vez, éstos interruptores tampoco cuentan con circuitería de rebote. El interruptor más a la izquierda, BTN3, es también el Pin de reset de default del usuario. El BTN3 eléctricamente funciona idénticamente que los otros interruptores. Sin embargo, cuando se configura, BTN3 provee referencia de reset para el diseño.

Diodos Emisores de Luz (LED)

La tarjeta también incluye ocho LED montados arriba de los push buttons sobre la superficie de la tarjeta, en la figura 2.5.2 se pueden localizar como (12) .

Los LED están etiquetados del LED0 al LED7. La siguiente tabla muestra las conexiones de los LED al FPGA.

LED	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0
FPGA Pin	P11	P12	N12	P13	N14	L12	P14	K12

El cátodo de cada LED se conecta a tierra a través de una resistencia de 270 Ω Para iluminar un LED individual, el drive asociado en el FPGA envía una señal alta, un uno lógico.

2.5.4 Puerto VGA

La tarjeta incluye un puerto de display VGA y un conector DB15, indicado en la figura 2.5.2 como 5 . Se conecta éste puerto a través de un cable de monitor estándar a un monitor de PC o una pantalla plana LCD. (Ver figura 2.5.5)

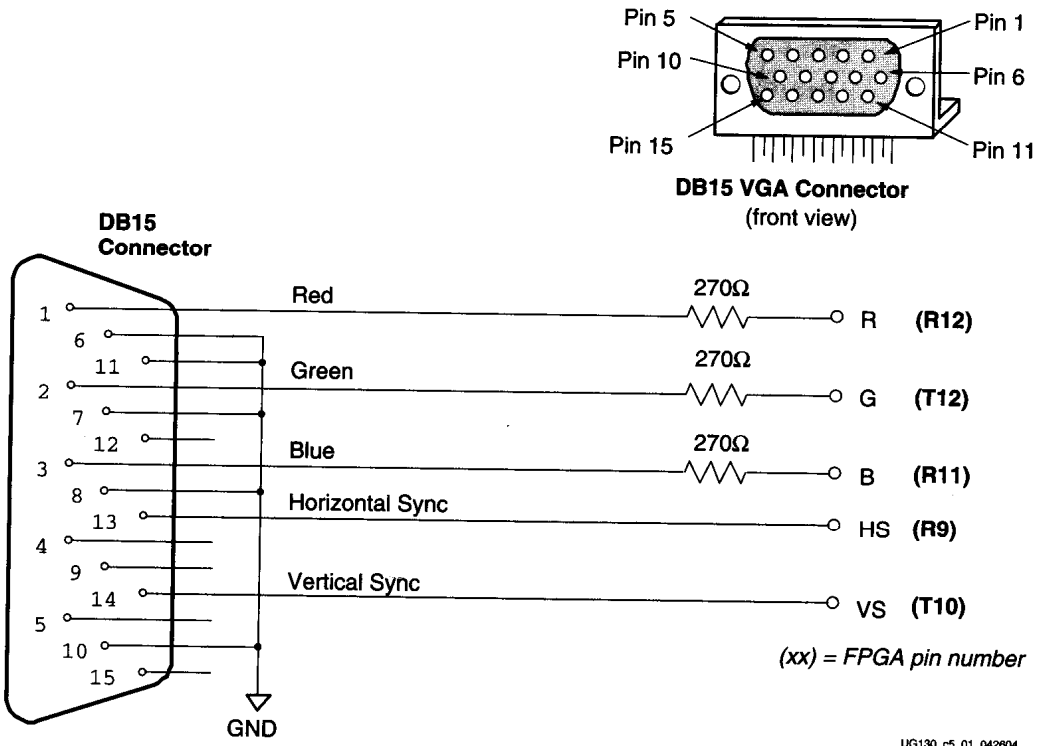


Figura 2.5.5 Conexiones VGA desde el Kit de Tarjeta Spartan-3 Starter

Como se muestra en la figura 2.5.5 la tarjeta Spartan-3 controla 5 señales VGA: Rojo (R), verde (G), azul (B), sincronía horizontal (HS), y sincronía vertical (VS), todas disponibles en el conector VGA. Los pines del FPGA que manejan el puerto VGA aparecen en la siguiente tabla.

Señal	Pin del FPGA
Rojo (R)	R12
Verde (G)	T12
Azul (B)	R11
Sincronía Horizontal (HS)	R9
Sincronía Vertical (VS)	T10

Cada línea de color tiene una resistencia en serie para proveer 3 bits de color, con un bit para cada línea de color R, G y B. La resistencia en serie de la terminación del cable VGA es de 75 Ω para asegurar que las señales de color se mantengan en los rangos específicos de 0V a 0.7 V. Las señales HS y VS están en niveles TTL. Las señales R, G y B manejan niveles Alto y Bajo para generar ocho posibles colores mostrados en la siguiente tabla.

Rojo (R)	Verde (G)	Azul (B)	Color Resultado
0	0	0	**Negro**
0	0	1	**Azul**
0	1	0	**Verde**
0	1	1	**Cian**
1	0	0	**Rojo**
1	0	1	**Magenta**
1	1	0	**Amarillo**
1	1	1	**Blanco**

Tabla Códigos de colores desplegados con 3 bits

Los tiempos de señales de VGA están especificados, publicados, registrados y soportados por Video Electronics Standards Association (VESA). El FPGA puede manejar monitor en modo 640x480 a 60Hz. Para información más precisa o manejo de frecuencias mayores, referirse a documentación disponible en la página de VESA o sitios relacionados.

2.5.5 PS/2 puerto de ratón/teclado

La tarjeta Spartan-3 incluye un puerto de PS/2 ratón/teclado y el conector estándar de 6 pínes mini-DIN, etiquetado en la tarjeta como J3 e indicado en la figura 2.5.2 como (9). En la figura 2.5.6 se muestra el conector PS/2 y en la tabla 2.5.6 las señales de cada conector. Sólo los pines 1 al 5 se conectan al FPGA.

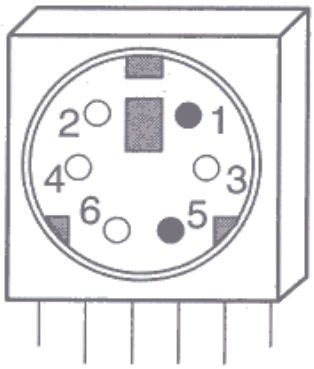


Figura 2.5.6 PS/2 Conector DIN

PS/2	Señal	Pin FPGA
1	Datos (PS2D)	M15
2	Reservado	---
3	Tierra (GND)	Tierra (GND)
4	Alimentación de Voltaje	---
5	CLK (PS/2)	M16
6	Reservado	---

Tabla 2.5.6 Conexiones PS/2 al FPGA de la Spartan-3

El ratón y teclado de una PC utilizan la misma conexión del bus serial PS/2 para comunicarse a los dispositivos, en este caso la tarjeta Spartan -3. El bus PS/2 incluye reloj y datos. Ambos dispositivos, ratón y teclado, utilizan idénticamente las señales de tiempo y usan palabras de 11 bits que incluyen los bits de inicio, paro y paridad.

Los tiempos del bus PS/2 se muestran en la tabla 2.5.7. Las señales de reloj y datos sólo aparecen cuando ocurren transferencias, de otra forma éstas señales se mantienen vacías o en estado lógico Alto. Los tiempos definen los requerimientos de la señal de comunicación del ratón a la tarjeta y bi-direccional para la comunicación al teclado.

Símbolo	Parámetro	Mínimo	Máximo
Tck	Clock high or low time	30 μsec	50 μsec
Tsu	Data-to-clock setup time	5 μsec	25 μsec
Thld	Crack-to-data hold time	5 μsec	25 μsec

Tabla 2.5.7 Tiempos del Bus PS/2

Los teclados y ratones recientes trabajan hoy en día igualmente con alimentación de 3.3 V ó 5 V. El voltaje de alimentación en el puerto PS/2 es seleccionable a través del puente JP2, indicado en la figura 2.5.2 como (30).

2.5.6 Puerto RS-232

La tarjeta Spartan-3 tiene un puerto serial RS-232. Las señales de recepción y transmisión en el puerto RS-232 aparecen en el conector DB9 hembra, indicado en la figura 2.5.2 como (6). El conector es de tipo de puerto

DCE y se conecta a un conector DB9 estilo DTE serial disponible en la mayoría de las computadoras personales y estaciones de trabajo.

En la figura 2.5.7 se muestran las conexiones entre el circuito FPGA y el conector DB9, incluyendo el circuito Maxim convertidor de voltaje MAX3232, indicado en la figura 2.5.2 como (7).

El FPGA alimenta los datos de salida como niveles TTL ó CMOS al circuito Maxim, quien convierte el valor lógico al apropiado nivel de voltaje para el RS-232 y viceversa, también el circuito Maxim convierte la entrada de datos del puerto serial RS-232 a niveles de voltaje para el FPGA. Existe una resistencia entre el pin de salida del circuito Maxim y el pin de RXD del FPGA para proteger contra conflictos lógicos accidentales.

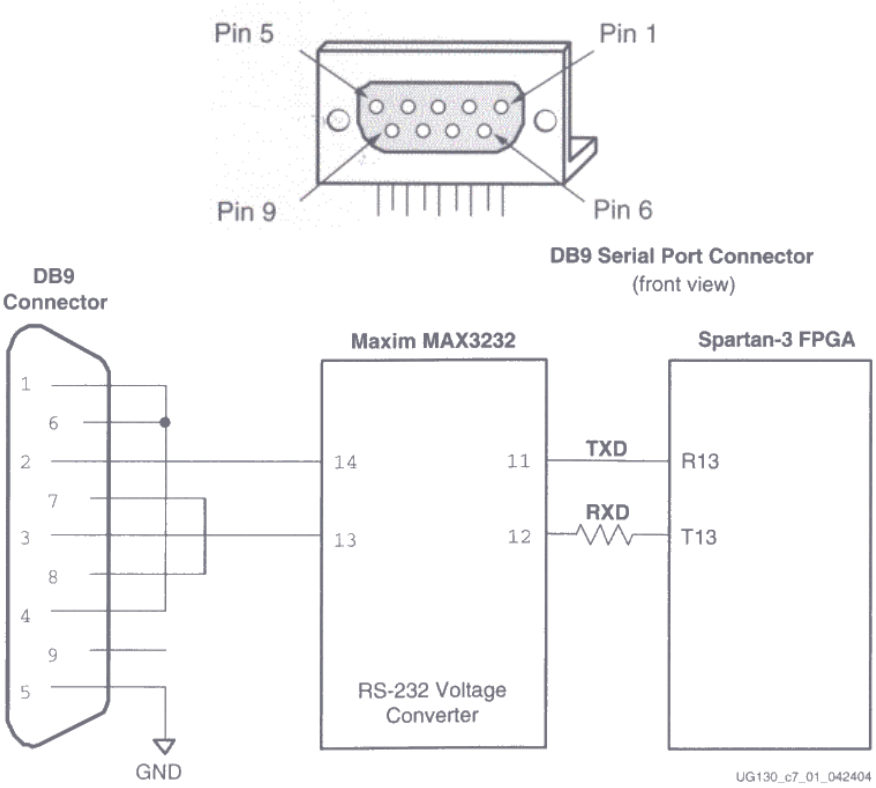


Figura 2.5.7 Puerto Serial RS-232

Las conexiones del FPGA al circuito RS-232 Maxim aparecen en la tabla 2.5.8.

Señal	Pin FPGA
RXD	T13
TXD	R13
RXD-A	N10
TXD-A	T14

Tabla 2.5.8 Conexiones de los accesorios del puerto al FPGA de la tarjeta Spartan-3

Hay también un canal auxiliar serial RS-232 del circuito Maxim disponible en dos pines, indicados en la tarjeta como J1 y se muestra en la figura 2.5.2 como (8). Las conexiones son indicadas en la tabla 2.5.3 con señales RxD-A y TxD-A.

2.5.7 Fuentes de reloj

La tarjeta Spartan-3 tiene una fuente oscilador de reloj dedicado de 50 MHz de la serie Epson SG-8002JF y un socket para otra fuente de oscilación de reloj adicional.

El oscilador de reloj de 50 MHz está montado en la parte trasera de la tarjeta e indicada en la figura 2.5.3 como (14) .

Se puede utilizar el reloj como tal o bien derivar su frecuencia utilizando Manejadores de Reloj Digital del FPGA (DCMs) Digital Clock Managers.

El socket del oscilador opcional está indicado en la figura 2.5.2 como (15) , y acepta osciladores con impreso de 8 patas o pines.

Fuente de Oscilación	Pin FPGA
50 MHz (IC4)	T9
Socket (IC8)	D9

Tabla 2.5.8 Fuentes de Oscilación de Reloj

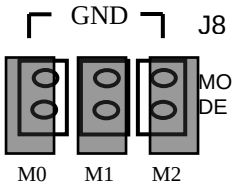
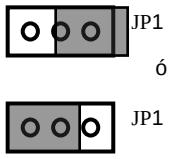
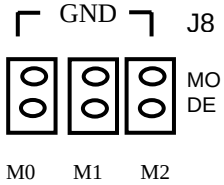
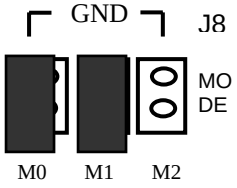
2.5.8 Modos de configuración y funciones del FPGA

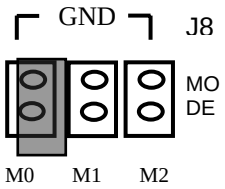
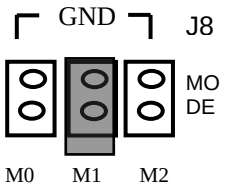
En la mayoría de las aplicaciones para la tarjeta Spartan-3, automáticamente el FPGA inicia de la memoria flash de la tarjeta cuando se alimenta la energía o el boton de “PROG” está presionado. Sin embargo, la tarjeta

soporta todos los modos de configuración disponibles a través del conector etiquetado como J8 e indicado en la figura 2.5.2, como (16). La tabla 2.5.9 indica las opciones disponibles para el conector J8. Adicionalmente, se requiere configurar el puente JP1 cuando se necesita utilizar el modo Configuración Serial Maestra (Master Serial Configuration).

La configuración de default en la tarjeta es la siguiente:

- Todos los puentes en el conector J8 están conectados.
- El puente JP1 está en la posición “default”.

Modo de Configuración <M0:M1:M2>	**Configuración del Conector J8**	**Configuración del puente JP1**	**Descripción**
Master Serial <0:0:0>			DEFAULT. Automáticamente el FPGA inicializa de la Plataforma Flash.
			El FPGA intenta inicializar de una fuente de configuración serial entre el conector A2 ó B1
Slave Serial <1:1:1>			Cualquier otro dispositivo conectado en el A2 ó B1, el conector de expansión provee las señales de reloj y datos para cargar al FPGA
Master Parallel <1:1:0>			El FPGA intenta inicializar de una fuente en configuración paralela conectada al conector de expansión B1.
Modo de Configuración <M0:M1:M2>	**Configuración del Conector J8**	**Configuración del puente JP1**	**Descripción**

Slave Parallel <0:1:1>			Cualquier otro dispositivo conectado en B1 y provee señal de datos y reloj para cargar al FPGA.
JTAG <1:0:1>			La tarjeta FPGA espera la configuración vía la interfase JTAG de 4 hilos.

El LED indicador de Listo y el interruptor Push Button para programar

La tarjeta Spartan-3 incluye dos funciones de configuración del FPGA, localizados cerca del conector VGA y el conector de entrada de alimentación AC, mostrado en la figura 2.5.8. El interruptor push buttom para programar se muestra en la figura 2.5.2 como (17), y maneja el pin o pata de programación PROG_B del FPGA. Cuando se presiona este interruptor se fuerza al FPGA para que se reconfigure y vuelve a cargar los datos de configuración.

El LED de Listo (DONE), se muestra en la figura 2.5.2 como (18), y éste está conectado al pin o pata del FPGA de “DONE” y enciende cuando el FPGA está configurado correctamente.

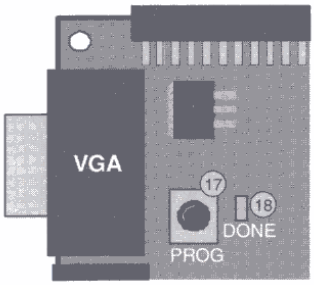


Figura 2.5.8 El Interruptor PROG y el LED DONE.

2.5.9 Almacenamiento de la configuración de la plataforma flash

La tarjeta Spartan-3 tiene un circuito flash PROM de la serie XCF02S para almacenar los datos de configuración del FPGA y agrega potencialidad en los datos no volátiles, incluyendo el código de aplicación MicroBlaze. Para configurar

la memoria del FPGA de Plataforma Flash, los tres puentes deben ser instalados en el conector J8, indicado en la figura 2.5.2 como (16) .

Opciones de la plataforma flash con el puente JP1

La Plataforma Flash tiene tres opcionales configuraciones con el puente JP1. El puente JP1 está indicado como (3) en la figura 2.5.2. En la tabla 2.5.10 se muestra es forma esquemática el detalle.

Opciones	Configuración puente JP1	Descripción
Default		El FPGA inicializa de la Plataforma Flash. No hay datos almacenados adicionales disponibles.
Lectura de la Flash		El FPGA inicializa de la Plataforma Flash, la cual está permanentemente habilitada. El FPGA puede leer datos adicionales de la Plataforma Flash.
Desactivado		Sin puente. Plataforma Flash deshabilitada. Otra fuente de configuración de datos provee los datos de inicio al FPGA.

Tabla 2.5.10 El puente JP1 controla las opciones de la Plataforma Flash

Opción de “Default”

Para la mayoría de las aplicaciones, esta es la configuración por default. Como muestra la figura 2.5.10, la Plataforma Flash está activada sólo durante la configuración cuando en el pin o pata del FPGA “DONE” tiene un cero lógico o nivel bajo. Cuando este pin tiene un uno lógico o un nivel alto se termina la configuración, la Plataforma Flash se desactiva y se mantiene un nivel bajo.

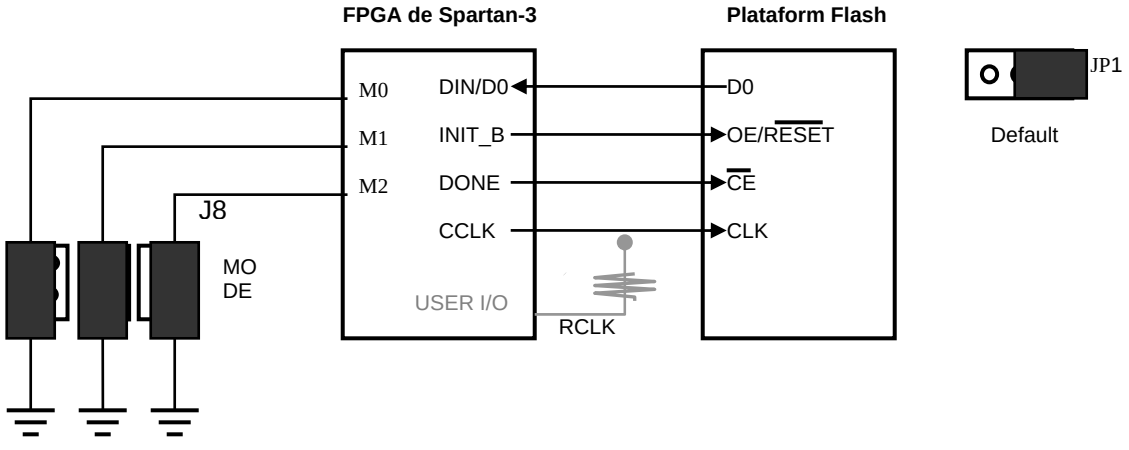


Figura 2.5.10 Opción de la Plataforma Flash de Default.

Opción “Lectura de la flash”

La tarjeta Spartan-3 incluye una Plataforma Flash de 2Mbit de configuración en su PROM. El FPGA XC3S200 de la tarjeta sólo requiere menos de 1Mbit para la configuración de datos. El resto de la Plataforma Flash está disponible para almacenar otros datos no volátiles con el FPGA, tales como revisión de códigos, números seriales, coeficientes, algún ID MAC de tarjeta de red, o código para un procesador embebido, tal como el MicroBlaze.

Para permitir al FPGA que lea de la Plataforma Flash después de la configuración, el puente JP1 debe ser posicionado correctamente, como se indica en la figura 2.5.11. Cuando el Puente JP1 está en posición, la Plataforma Flash está siempre activada. Después que el FPGA completa la configuración, la aplicación del FPGA pone en el Pin “INIT_B” en alto, el Pin “N9” del FPGA. Consecuentemente, el posicionador de datos de la Plataforma Flash no está inicialilzado y apunta al siguiente dato adicional de la configuración del FPGA. Para leer cualquier dato subsecuente, la aplicación del FPGA genera pulsos de reloj adicionales en la señal “RCLK” en el pin “A14” de las señales del FPGA. Después de la configuración, la salida “CCLK” del FPGA es de tres estados con una resistencia conectada en serie al V_{CCAUX} (205V). La Plataforma Flash presenta datos en serie en el pin “DIN” del FPGA, pin “M11”.

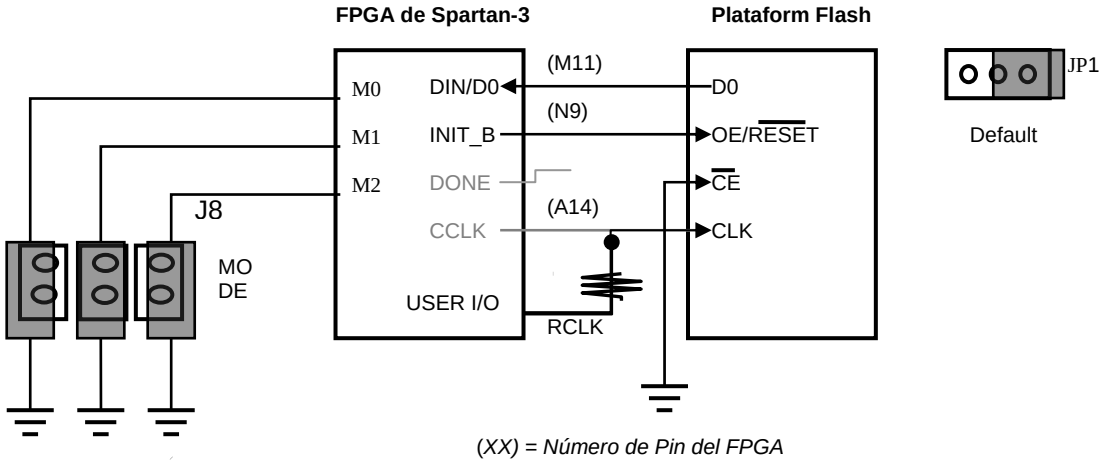


Figura 2.5.11 Datos Adicionales de Lectura de Plataforma Flash por Configuración del Puente JP1

La resistencia entre la salida “CCLK” y el pin A14 del FPGA previene cualquier conflicto entre las dos señales.

Opción “Deshabilitado”

Si el puente JP1 es retirado, entonces la Plataforma Flash está deshabilitada, y permite configuración vía una tarjeta de expansión conectada a uno de los conectores de expansión.

2.5.10 Puertos programación/desvío JTAG

La tarjeta Spartan-3 incluye una cadena de programación y desvío JTAG. Tanto como el FPGA de la tarjeta Spartan-3 como el circuito de la Plataforma Flash son parte de la cadena JTAG, como se muestra en la figura 2.5.12. Adicionalmente hay dos conectores JTAG para manejar las señales JTAG de varios cables para bajar y desviar JTAG. Se incluye un cable de bajo costo paralelo a JTAG como parte del kit de la tarjeta Spartan-3 y se conecta al conector etiquetado J7.

Conector (J7) JTAG

Este conector consiste de un conjunto de pines de estaca de 0.1pulgadas e indicado en la figura en la figura 2.5.2 como (22), y ubicado directamente debajo de los dos conectores de expansión. El cable incluido en el kit se ajusta con el conector J7 en sus pines como se muestra en la figura 2.5.13. Cuando se fija el cable correctamente, el cable debe ser perpendicular a la tarjeta. Hay que asegurarse que las señales de alineación del cable coincidan con las etiquetas mostradas en la tarjeta. El otro extremo del cable, se conecta al puerto paralelo de la PC. El cable es directamente compatible con el software iMPACT de Xilinx

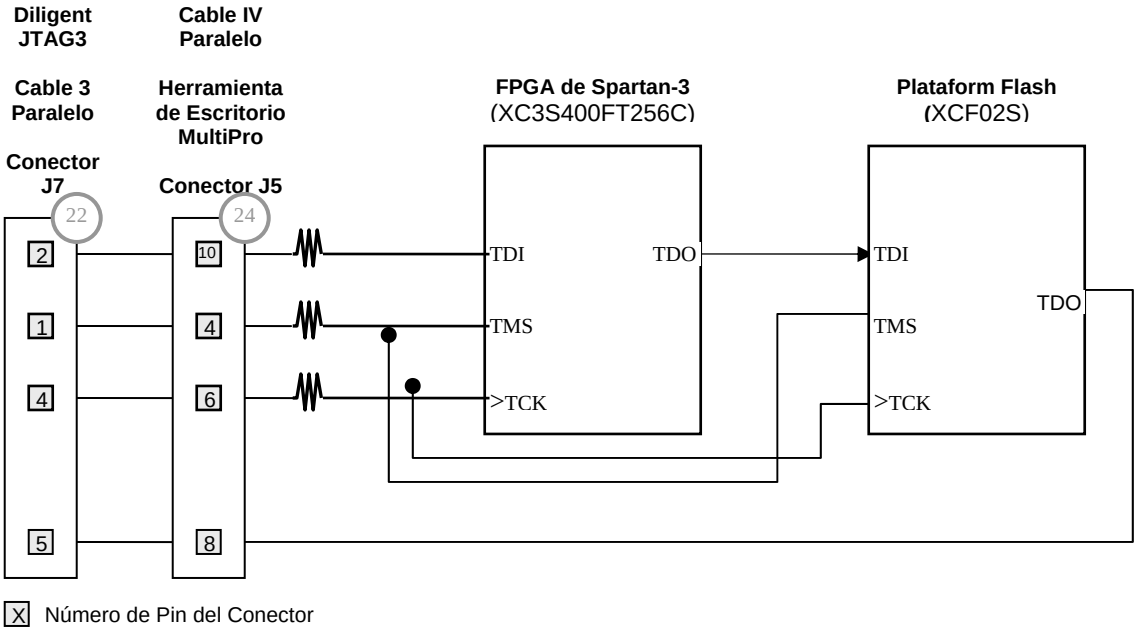


Figura 2.5.12 Cadena JTAG del Kit de la Tarjeta Spartan-3

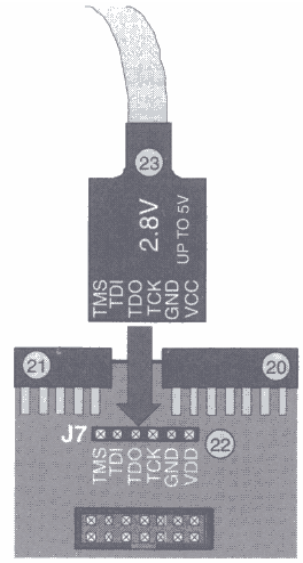


Figura 2.5.13 Cable JTAG Digilent provisto con el kit, se conecta al conector J7

Cable paralelo JTAG de herramienta de escritorio IV/MutiPro, conector (J5)

El conector J5 se muestra en la figura 2.5.2, como (24). Se utiliza un cable plano de 14 hilos que conecta a ambos extremos y uno de ellos se conecta al conector J5. Aunque el kit de herramientas de escritorio MultiPro y el cable paralelo IV utilizan múltiples modos de configuración para FPGAs, la tarjeta Spartan-3 sólo soporta el método de configuración JTAG

2.5.11 Distribución de potencia

Adaptador de CA de pared

La tarjeta Spartan-3 incluye un adaptador internacional de alimentación CA de pared que produce +5VCD. El conector del adaptador de CA se conecta a la tarjeta en el conector indicado en la figura 2.5.2 como (25). No hay un interruptor para deshabilitar la alimentación de la tarjeta. Para desconectarlo de la alimentación se requiere desconectar el cable de alimentación de CD de adaptador.

El indicador de LED de alimentación (POWER) se muestra en la figura 2.5.2 como (26) y enciende cuando el adaptador de CD está conectado correctamente a la tarjeta. Si los Puentes J8 y JP1 está apropiadamente colocados habrá una configuración válida en la memoria de la plataforma flash y el indicador de LED “DONE” se iluminará.

El adaptador es directamente compatible a las localidades de Norte América, Japón y Taiwán. Otras locaciones requerirán un socket adaptador para convertir del estándar de alimentación de Norte América al de otras localidades. El adaptador de corriente alterna opera de 100V hasta 240V de voltaje CA de entrada, a 50 ó 60 Hz.

Reguladores de voltaje

Hay múltiples fuentes de voltaje en la tarjeta Spartan-3, como se indica en la siguiente tabla 2.5.11.

Voltaje	Fuente	Qué Alimenta
+5V CD	Alimentador de CA de Pared, 5V con la fuente de alimentación (25 en la figura 2.5.2)	• Regulador de 3.3V • Opcional, el Puerto PS/2 si está configurado el puente JP2 • El Pin 1(VU) en los conectores de expansión A1, A2, B1
+3.3V CD	Regulador de 3.3V LM1086CS-ADJ de Nacional Semiconductor (27 en la figura 2.5.2)	• Reguladores de 2.5V y 1.2V • Alimentación de entrada V_{CCO} para todos los bancos de I/O del FPGA • La mayoría de los componentes de la Tarjeta • Al Pin 3 de los conectores de expansión A1, A2 B1
+2.5V CD	El regulador de 2.5V LF25CDT de STMMicroelectronics (28 en la figura 2.5.2)	• Alimentación de entrada V_{CCAUX} para el FPGA
+1.2V CD	El regulador de 1.2V FAN1112 de Fairchild Semiconductor (29 en la figura 2.5.2)	• Alimentación V_{CCINT} para el FPGA

Tabla 2.5.11 Fuentes y alimentadores de Voltaje

2.5.12 Tarjetas y conectores de expansión

La tarjeta Spartan-3 tiene tres conectores de 40 pines etiquetados como A1, A2 y B1 y mostrados en la figura 2.5.2 como 21 , 20 y 19 , respectivamente.

La tabla 2.5.12 resume las capacidades para cada puerto de expansión. El puerto A1 soporta hasta un máximo de 32 pines I/O de usuario, mientras que los otros puertos soportan hasta 34 pines I/O de usuario. Algunos pines están

compartidos con otras funciones en la tarjeta, lo cual puede reducir la cantidad de I/O efectivos para aplicaciones específicas

Conector	I/O de usuario	Direccionamiento SRAM	JTAG	Configuración Serial	Configuración paralelo
A1	32	X	X		
A2	34			X	
B1	34			X	X

Tabla 2.5.12 Características de los conectores de expansión.

Cada puerto ofrece la misma habilidad para programar el FPGA en la tarjeta. Los puertos A2 y B1 proveen conexiones para maestro o esclavo en modo configuración serial. Finalmente, el puerto B1 también ofrece esclavo o maestro en modo de configuración paralelo.

Cada conector de 40 pines, como se muestra en la figura 2.5.14, utiliza espaciamiento de 0.1 inch. El pin 1 en cada conector va conectado a Tierra (GND). Similarmente el pin 2 siempre va conectado a +5V CD de la fuente. Y el pin 3 está siempre conectado a la salida de +3.3V CD del regulador.

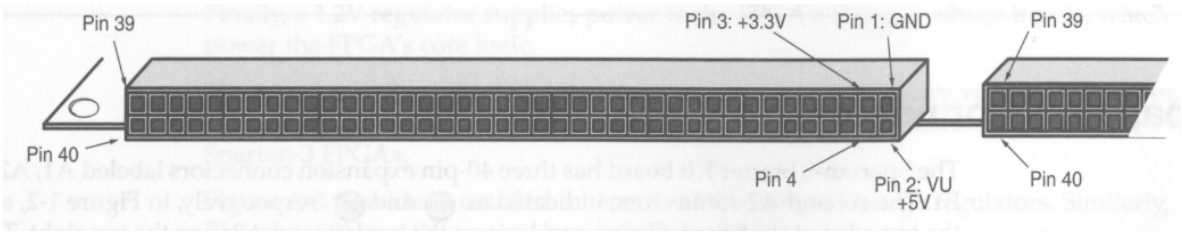


Figura 2.5.14 Conector de Expansión de 40 Pines

En las Tablas 10.5.13, 10.5.14 y 10.5.15 se muestran los nombres de cada señal y la distribución de la salida de cada uno de los pines de los conectores A1, A2 y B1, respectivamente. Las tablas incluyen las conexiones entre el FPGA y los conectores de expansión.

El conector de expansión A1 se muestra en la figura 2.5.2 como (21), y la distribución de sus pines de salida se muestra en la tabla 2.5.13.

Nombre Esquemático	Pata del FPGA	Conector		Pata del FPGA	Nombre Esquemático
GND		1	2		VU (+5V)
V_{cco} (+3.3V)	V_{cco} (Todos los bancos)	3	4	(N8)	ADR0
DB0	(N7)	5	6	(L5) SRAM A0	ADR1
DB1	(T8)	7	8	(N3) SRAM A1	ADR2
DB2	(R6)	9	10	(M4) SRAM A2	ADR3

DB3	(TS)	11	12	(M3) SRAM A3	ADR4
DB4	(R5)	13	14	(L4) SRAM A4	ADR5
DB5	(C2)	15	16	(G3) SRAM WE#	WE
DB6	(C1)	17	18	(K4) SRAM OE#	OE
DB7	(B1)	19	20	(P9) FPGA DOUT/BUSY	CSA
LSBCLK	(M7)	21	22	(M10)	MAI-DB0
MAI-DB1	(F3) SRAM A6	23	24	(G4) SRAM A5	MAI-DB2
MAI-DB3	(E3) SRAM A8	25	26	(F4) SRAM A7	MAI-DB4
MAI-DB5	(G5) SRAM A10	27	28	(E4) SRAM A9	MAI-DB6
MAI-DB7	(H4) SRAM A12	29	30	(H3) SRAM A11	MAI-ASTB
MAI-DSTB	(J3) SRAM A14	31	32	(J4) SRAM A13	MAI-WRITE
MAI-WAIT	(K5) SRAM A16	33	34	(K3) SRAM A15	MAI-RESET
MAI-INT	(L3) SRAM A17	35	36	JTAG Aterrizado	JTAG Aterrizado
TDS	(C13) FPGA JTAG TMS	37	38	(C14) FPGA JTAG TCK	TCK
TDO-ROM	Plataforma Flash JTAG TDO	39	40	Conector J7, pata 3	TDO-A

Tabla 2.5.13 Distribución de Pines de Salida del Conector de Expansión A1

El conector de expansión A2 se muestra en la figura 2.5.2 como (20), y la distribución de sus pines de salida se muestra en la tabla 2.5.14.

Nombre Esquemático	Pata del FPGA	Conector		Pata del FPGA	Nombre Esquemático
GND		1	2		VU (+5V)
V_{cco} (+3.3V)	V_{cco} (Todos los bancos)	3	4	(E6)	PA-I01
PA-IO2	(D5)	5	6	(C5)	PA-I03
PA-IO4	(D6)	7	8	(C6)	PA-I05
PA-IO6	(E7)	9	10	(C7)	PA-I07
PA-IO8	(D7)	11	12	(C8)	PA-I09
PA-IO10	(D8)	13	14	(C9)	PA-I11
PA-IO12	(D10)	15	16	(A3)	PA-I13
PA-IO14	(B4)	17	18	(A4)	PA-I15
PA-IO16	(B5)	19	20	(A5)	PA-I17

PA-IO18	(B6)	21	22	(B7)	MA2-DB0
MA2-DB1	(A7)	23	24	(B8)	MA2-DB2
MA2-DB3	(A8)	25	26	(A9)	MA2-DB4
MA2-DB5	(B10)	27	28	(A10)	MA2-DB6
MA2-DB7	(B11)	29	30	(B12)	MA2-ASTB
MA2-DSTB	(A12)	31	32	(B13)	MA2-WRITE
MA2-WAIT	(A13)	33	34	(B14)	MA2-RESET
MA2-INT/GCK4	(D9) Socket del Oscilador	35	36	(B3) FPGA PROG_B	PROG-B
DONE	(R14) FPGA DONE	37	38	(N9) FPGA INIT_B	INIT
CCLK	(T15) FPGA CCLK Conecta al (A14) via resistencia 390Ω	39	40	(M11)	DIN

Tabla 2.5.14 Distribución de Pines de Salida del Conector de Expansión A2

El conector de expansión B1 se muestra en la figura 2.5.2 como (19), y la distribución de sus pines de salida se muestra en la tabla 2.5.15.

Nombre Esquemático	Pata del FPGA	Conector		Pata del FPGA	Nombre Esquemático
GND		1	2		VU (+5V)
V_{cco} (+3.3V)	V_{cco} (Todos los bancos)	3	4	(C10)	PB-ADR0
PA-IO2	(T3) FPGA RD_WR_B config	5	6	(E10)	PB-ADR1
PA-IO4	(N11) FPGA D1config	7	8	(C11)	PB-ADR2
PA-IO6	(P10) FPGA D2 config	9	10	(D11)	PB-ADR3
PA-IO8	(R10) FPGA D3 config	11	12	(C12)	PB-ADR4
PA-IO10	(T7) FPGA D4 config	13	14	(D12)	PB-ADR5
PA-IO12	(R7) FPGA D5 config	15	16	(E11)	PB-WE
PA-IO14	(N6) FPGA D6 config	17	18	(B16)	PB-OE
PA-IO16	(M6) FPGA D7config	19	20	(R3) FPGA CS_B config	PB-CS
PA-IO18	(C15)	21	22	(C16)	MB1-DB0
MA2-DB1	(D15)	23	24	(D16)	MB1-DB2
MA2-DB3	(E15)	25	26	(E16)	MB1-DB4
MA2-DB5	(F15)	27	28	(G15)	MB1-DB6
MA2-DB7	(G16)	29	30	(H15)	MB1-ASTB
MA2-DSTB	(H16)	31	32	(B13)	MB1-WRITE
MA2-WAIT	(K16)	33	34	(K15)	MB1-RESET

MA2-INT/GCK4	(L15)	35	36	(B3) FPGA PROG_B	PROG-B
DONE	(R14) FPGA DONE	37	38	(N9) FPGA INIT_B	INIT
CCLK	(T15) FPGA CCLK Conecta al (A14) via resistencia 390Ω	39	40	(M11)	DIN

Tabla 2.5.15 Distribución de Pines de Salida del Conector de Expansión B1

Capítulo 3

Brazos robóticos

En el presente capítulo se ve de modo introductorio el origen, clasificación y modos de accionamiento de los robots en general, hasta llegar a la operación, manipulación y aplicaciones, así como las especificaciones del brazo robótico. La construcción del brazo robótico no es motivo de esta tesis.

3.1 Robótica

En el contexto actual, la noción de robótica atiende a una idea de estructura mecánica universal capaz de adaptarse, como el hombre, a muy diversos tipos de acciones y en las que concurran, en mayor o menor medida según los casos, las características de movilidad, gobernabilidad, autonomía y polivalencia. La robótica, tomada en sentido general abarca una amplia gama de dispositivos con muy diversas cualidades físicas y funcionales asociada a la particular estructura mecánica de aquellos, a sus características operativas y al campo de aplicación para el que se ha concebido.

Todos estos factores están íntimamente relacionados, de forma que la configuración y el comportamiento de un robot condicionan su adecuación para un campo de aplicación especifico. La robótica se apoya en gran medida en los progresos de la microelectrónica y la microinformática, así como en nuevas disciplinas como el reconocimiento de formas y la inteligencia artificial; en este sentido la robótica cuenta con valiosos recursos a su alcance: electrónica, servomecanismos, controladores, sensores y equipos de comunicación entre otros. Las investigaciones actuales se orientan especialmente a la construcción de maquinas capaces de trabajar en medios parcialmente desordenados y de responder con eficacia ante situaciones no totalmente previstas o sea que el robot sea capaz de relacionarse con el mundo que le rodea a través de sensores y de tomar decisiones en tiempo real.

3.2 Desarrollo histórico

El comienzo de la revolución industrial proporcionó un medio potencialmente poderoso de mover las maquinas: “el vapor”. Sin embargo también eran potencialmente peligrosas las explosiones de las calderas y máquinas y existía una gran necesidad de alguna forma de regular automáticamente el suministro de vapor según la carga. Por ello el regulador centrífugo de bolas de Sir

James Watt, introducido en 1787, proporciono la solución. Como se muestra si la carga sobre el árbol disminuye bruscamente y el árbol se acelera haciendo que la fuerza centrífuga mueva las bolas giratorias hacia fuera, esto hace que se eleve la pieza deslizante conectada a la válvula de vapor disminuyendo con ello el suministro de vapor y finalmente reduciendo la velocidad de árbol de salida. Este principio fue de hecho utilizado primitivamente en los molinos de viento (Mayr, 1970). El desarrollo clave en la década de los 30 fue el análisis de estabilidad de los nuevos amplificadores electrónicos (Nyquist, 1932), mas tarde mediante la teoría del control han sido reunidos por MacFarlane (1979) que se aplico rápidamente a los dispositivos mecánicos.

El robot industrial incluye el principio de secuenciación y retroalimentación para proporcionar movimientos rápidos y precisos. La razón de por qué la aparición de robots es relativamente reciente reside en la evolución de computadoras rápidas y seguras que forman el corazón del robot y proporcionan tanto el control como la reprogramabilidad. La disponibilidad de computadoras ha significado también un desarrollo paralelo de sensores y procesamiento sensorial, esto junto con la flexibilidad inherente, proporcionada por tener los movimientos del robot, mandados a partir de una computadora de control hacen que los movimientos de un robot pueden modificarse de acuerdo con la información sensorial. La posición se transmite al robot, el cual entonces agarra la pieza correctamente.

El concepto de máquinas automatizadas se remonta a la antigüedad, con mitos de seres mecánicos vivientes. Los autómatas, o maquinas semejantes a personas, ya aparecían en los relojes de las iglesias medievales, y los relojeros del siglo XVIII eran famosos por sus ingeniosas criaturas mecánicas.

Algunos de los primeros robots empleaban mecanismos de realimentación, para corregir errores, mecanismos que siguen empleándose actualmente. Un ejemplo de control por realimentación es una cisterna que emplea un flotador para determinar el nivel del agua. Cuando el agua cae por debajo de un nivel determinado, el flotador baja, abre una válvula y deja entrar mas agua en la cisterna. Al subir el agua, el flotador también sube, y al llegar a cierta altura se cierra la válvula y se corta el paso del agua.

Aunque el primer auténtico controlador realimentado fue el regulador de Watt, inventado en 1788 por el ingeniero mecánico británico James Watt. Este dispositivo constaba de dos bolas metálicas unidas al eje motor de una maquina de vapor y con una válvula que regulaba el flujo del vapor. A medida que aumentaba la velocidad de maquina de vapor, las bolas se alejaban del eje debido a la fuerza centrífuga, con la que cerraban la válvula. Esto hacia que disminuyera el flujo de vapor a la máquina y por tanto la velocidad.

El control por realimentación, el desarrollo de herramientas especializadas y la división del trabajo en las tareas más pequeñas que pudieran realizar obreros o máquinas fueron ingredientes esenciales en la automatización de las fábricas en el siglo XVIII. A medida que mejoraba la tecnología se desarrollaron máquinas especializadas para tareas como poner tapones a las botellas o verter caucho

líquido en moldes para neumáticos. Sin embargo, ninguna de estas máquinas tenia la versatilidad del brazo humano, y no podían alcanzar objetos alejados y colocarlos en la posición deseada.

El desarrollo del brazo artificial multiarticulado, o manipulador, llevó al moderno robot. El inventor estadounidense George Devol desarrolló en 1954 un brazo primitivo que se podía programar para realizar tareas específicas. En 1975, el ingeniero mecánico estadounidense Victor Scheinman, cuando estudiaba la carrera, desarrollo un manipulador polivalente realmenente flexible conocido como Brazo Manipulador Universal Programable (PUMA). El PUMA era capaz de mover un objeto y colocarlo en cualquier orientación en un lugar deseado que estuviera a su alcance. El concepto básico multiarticulado del PUMA es la base de la mayoría de los robots actuales.

3.3 Robots

La necesidad cada vez mayor de aumentar la productividad y conseguir productos acabados de una calidad uniforme lleva a la industria a girar cada vez más hacia una automatización basada en computadoras.

La inflexibilidad y generalmente el alto costo de estas máquinas, llevó a un interés creciente en el uso de robots capaces de efectuar una variedad de funciones de fabricación en un entorno de trabajo más flexible y a un menor costo de producción.

El término robot procede de la palabra checa “robota”, que significa 'trabajo obligatorio', fue empleado por primera vez en la obra teatral R.U.R (Robots Universales de Rossum), estrenada en Enero de 1921 en Praga por el novelista y dramaturgo checo Karel Capek.

Hoy la palabra robot tiene diferentes significados:

"Dispositivos capaces de moverse de modo flexible análogo al que poseen los organismos vivos, con o sin funciones intelectuales, permitiendo operaciones en respuesta a las órdenes humanas". (Asociación Japonesa de Robótica Industrial, JIRA)

"Un manipulador multifuncional y reprogramable diseñado para desplazar materiales, componentes, herramientas o dispositivos especializados por medio de movimientos programados variables con el fin de realizar tareas diversas". (Instituto de Robótica de América, RIA)

Los robots exhiben tres elementos claves según la definición adoptada:

Programabilidad: Lo que significa disponer de capacidades computacionales y de manipulación de símbolos (el robot es un computador).

Capacidad mecánica: Que lo capacita para realizar acciones en su entorno y no ser un mero procesador de datos (el robot es una máquina).

Flexibilidad: Puesto que el robot puede operar según un amplio rango de programas y manipular material de formas distintas.

Por lo que podemos decir que:

"Un robot es un dispositivo de manipulación reprogramable y multifuncional, diseñado para mover material, piezas, herramientas o dispositivos especializados, mediante movimientos programados variables, con el fin de que sea capaz de realizar una cierta variedad de tareas a la vez que interacciona con su entorno. Es una unión de un software y un hardware. El software es la inteligencia que hay detrás del mecanismo, y es esta inteligencia la que diferencia a un robot de otras formas de automatización."

3.4 Clasificación de los robots

Un robot puede estar constituido por cuatro entidades unidas entre sí:

Sistema mecánico articulado dotado de sus motores (eléctricos, hidráulicos o neumáticos) que arrastran a las articulaciones del robot mediante las transmisiones (cables, cintas, correas con muescas). Para conocer en todo instante la posición de las articulaciones se recurre a los captadores (codificadores ópticos) que se denominan propioceptivos. Estos dan el valor a las articulaciones, que no es más que la configuración o el estado del robot.

El entorno es el universo en que está sumergida la primera entidad. Si los robots están sobre un puesto fijo se reduce al espacio alcanzable por el robot. En él, el robot puede encontrar obstáculos que ha de evitar y objetos de interés, o sea los objetos con los que tiene que actuar. Por todo esto existe interacción entre la parte física y el entorno. Mediante los captadores exteroceptivos (cámaras, detectores de fuerzas, detectores de proximidad, captadores táctiles) se toma información sobre el entorno.

Las tareas a realizar es el trabajo que se desea que haga el robot. La descripción de estas tareas se hace mediante lenguajes que pueden ser a través de los gestos, en el que se le enseña al robot lo que se debe hacer; orales, se le habla; por escrito en el que se le escriben las instrucciones en un lenguaje compatible con el robot.

El cerebro del robot es el órgano de tratamiento de la información. Este puede ser desde un autómata programable para los menos avanzados hasta un mini ordenador numérico o microprocesador para los más avanzados.

3.6.1 Arquitectura

Por su arquitectura los robots pueden clasificarse en:

Androides: Los androides son robots que se parecen y actúan como seres humanos. Los robots de hoy en día vienen en todas las formas y tamaños, pero a excepción de los que aparecen en las ferias y espectáculos, no se parecen a las personas y por tanto no son androides. Actualmente, los androides reales sólo existen en las películas de ficción. (Figura 3.5.1)

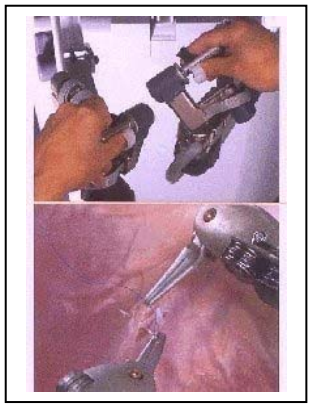
Figura 3.5.1 Androide

Móviles: Los robots móviles están provistos de patas, ruedas u orugas que los capacitan para desplazarse de acuerdo su programación. Elaboran la información que reciben a través de sus propios sistemas de sensores y se emplean en determinado tipo de instalaciones industriales, sobre todo para el transporte de mercancías en cadenas de producción y almacenes. También se utilizan robots de este tipo para la investigación en lugares de difícil acceso o muy distantes, como es el caso de la exploración espacial y las investigaciones.

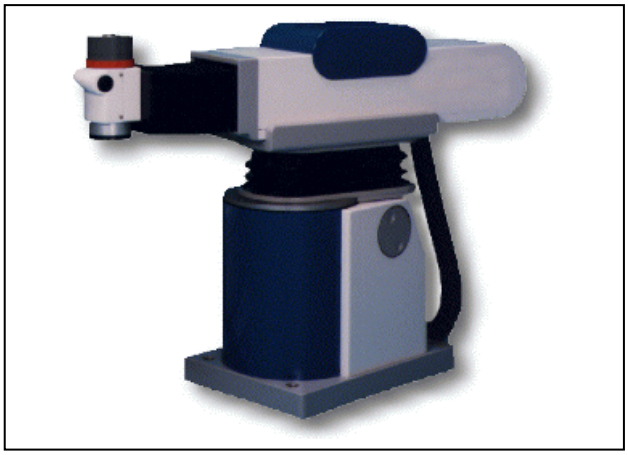
Zoomórficos: Robots caracterizados principalmente por su sistema de locomoción que imita a diversos seres vivos. Los androides también podrían considerarse robots zoomórficos.

Robot atún bajo las profundidades del océano

Médicos: Los robots médicos son, fundamentalmente, prótesis para disminuidos físicos que se adaptan al cuerpo y están dotados de potentes sistemas de mando. Con ellos se logra igualar con precisión los movimientos y funciones de los órganos o extremidades que suplen.



Industriales: Los robots industriales son artilugios mecánicos y electrónicos destinados a realizar de forma automática determinados procesos de fabricación o manipulación. Son en la actualidad los más frecuentes.



Híbridos: Estos robots corresponden a aquellos de difícil clasificación cuya estructura resulta de una combinación de las expuestas anteriormente.

Así mismo en su arquitectura debemos considerar los aspectos con los cuales el robot se va a desarrollar, que son:

Grados de Libertad: es el número de parámetros que es preciso conocer para determinar la posición del robot, es decir, los movimientos básicos independientes que posicionan a los elementos del robot en el espacio. En los robots industriales se consideran 6 grados de libertad: Tres de ellos para definir la posición en el espacio y los otros tres para orientar la herramienta.

Precisión: En la continua repetición del posicionamiento de la mano de sujeción de un robot industrial se establece un mínimo de precisión aceptable de 0,3 mm, aunque es factible alcanzar precisiones de 0,05 mm.

Capacidad de carga: Es el peso en kilogramos (generalmente) que el robot puede manipular. Si son pesos muy elevados se utilizarán mecanismos hidráulicos.

Sistemas de coordenadas para los movimientos del robot: Son los movimientos y posiciones que se pueden especificar en coordenadas cartesianas, cilíndricas y polares.

3.6 Capacidades

En esta clasificación se tiene más en cuenta la potencia del software en el controlador, lo que es determinante de la utilidad y flexibilidad del robot dentro de las limitantes del diseño mecánico y la capacidad de los sensores. De acuerdo a esta posición los robots han sido clasificados de acuerdo a su:

- Nivel de complejidad
- Nivel de inteligencia
- Nivel de control
- Nivel de lenguaje de programación.

3.6.1 Nivel de complejidad

La generación de un robot se determina por el orden histórico de desarrollo en la robótica. Cinco generaciones son normalmente asignadas a los robots.

Robots de primera generación: Dispositivos que actúan como "esclavo" mecánico de un hombre, quien provee mediante su intervención directa el control de los órganos de movimiento. Esta transmisión tiene lugar mediante servomecanismos actuados por las extremidades superiores del hombre, caso típico manipulación de materiales radiactivos, obtención de muestras submarinas, etc.

Robots de segunda generación: El dispositivo actúa automáticamente sin intervención humana frente a posiciones fijas en las que el trabajo ha sido preparado y ubicado de modo adecuado ejecutando movimientos repetitivos en el tiempo, que obedecen a lógicas combinatorias, secuenciales, programadores paso a paso, neumáticos o Controladores Lógicos Programables. Son utilizados en la inyección de termoplásticos y metales no ferrosos, en los procesos de soldadura a punto y continua, en tareas de pintado y reemplazando con ventaja algunas operaciones de máquinas convencionales.

Robots de tercera generación: Son dispositivos que habiendo sido construidos para alcanzar determinados objetivos serán capaces de elegir la mejor forma de hacerlo teniendo en cuenta el ambiente que los circunda. Para obtener estos resultados es necesario que el robot posea algunas condiciones que posibiliten su interacción con el ambiente y los objetos. Las mínimas aptitudes requeridas son: capacidad de reconocer un elemento determinado en el espacio y la capacidad de adoptar propias trayectorias para conseguir el objetivo deseado.

Robots de cuarta generación: se desarrolla en los laboratorios de investigación. Se trata de robots altamente inteligentes con más y mejores extensiones sensoriales

para entender sus acciones y captar el mundo que los rodea. Incorporan conceptos “modélicos” de conducta.

Robots de quinta generación: Actualmente en desarrollo. Esta nueva generación de robots basa su acción principalmente en modelos conductuales establecidos.

3.6.2 Nivel de inteligencia

Robots inteligentes: Son manipuladores o sistemas mecánicos *multifuncionales* controlados por computadoras capaces de relacionarse con su entorno o a través de sensores y tomar decisiones en tiempo real. Concepto de “Inteligencia Artificial”.

Robots con control por computadora**:** Similares a los anteriores pero carecen de la capacidad de relacionarse con el entorno que les rodea.

Robots de aprendizaje**:** Se limitan a repetir una secuencia de movimientos realizada con la intervención de un operador y luego lo memorizan todo.

Robots manipuladores**:** Son sistemas mecánicos multifuncionales cuyo sencillo sistema de control permite gobernar el movimiento de sus elementos de las formas siguientes:

1. Manual: el operador lo controla directamente.
2. De Secuencia Variable: es posible alterar algunas de las características de los ciclos de trabajo.

Nota: los manipuladores son considerados robots en Japón, pero no en Europa y EEUU, sólo algunos de secuencia variable.

3.6.3 Nivel de control

El método más común de programar un robot para que realice una nueva tarea es usar un control de aprendizaje. El control de aprendizaje es un mando de control manual que permite a un operador mover las distintas partes de un robot.

El control de aprendizaje no está unido directamente al robot, sino por medio del control principal de la computadora del robot.

Cuando mueve cada articulación, la computadora graba cada posición. Después de completar el aprendizaje el robot puede realizar el trabajo por si mismo sin necesidad de más ayuda.

Los programas en el controlador del robot pueden ser agrupados de acuerdo al nivel de control que realizan:

Nivel de inteligencia artificial, donde el programa aceptará un comando como "levantar el producto" y descomponerlo dentro de una secuencia de comandos de bajo nivel basados en un modelo estratégico de las tareas.

Nivel de modo de control, donde los movimientos del sistema son modelados, para lo que se incluye la interacción dinámica entre los diferentes mecanismos, trayectorias planeadas, y los puntos de asignación seleccionados.

Niveles de servosistemas, donde los actuadores controlan los parámetros de los mecanismos con el uso de una retroalimentación interna de los datos obtenidos por los sensores, y la ruta es modificada sobre la base de los datos que se obtienen de sensores externos. Todas las detecciones de fallas y mecanismos de corrección son implementados en este nivel.

3.7 Lenguaje de control robótico

Un lenguaje de control robótico es un lenguaje informático diseñado específicamente para controlar un robot. Además de contener las órdenes normales, tales como el control y las condiciones, un lenguaje de control robótico incluye además ordenes para el control de los movimientos del robot. Es justamente este control del movimiento lo que separa el lenguaje de control robótico de todo el resto del lenguaje de programación general. Un lenguaje de control robótico contiene una base de datos incorporada con información espacial sobre cada uno de los movimientos que debe hacer el robot.

Es importante entender que el lenguaje de control robótico no está diseñado para reemplazar al control de aprendizaje, sino para complementarlo. Por tanto, un lenguaje de control robótico debe mantener una relación estrecha con el control de aprendizaje.

En la clasificación final se considera el nivel del lenguaje de programación.

La clave para una aplicación efectiva de los robots para una amplia variedad de tareas, es el desarrollo de lenguajes de alto nivel.

Los sistemas de programación de robots se ubican dentro de tres clases:

1.- Sistemas guiados, en el cual el usuario conduce el robot a través de los movimientos a ser realizados.

2.- Sistemas de programación de nivel-robot, en los cuales el usuario escribe un programa de computadora al especificar el movimiento.

3.- Sistemas de programación de nivel-tarea, en el cual el usuario especifica la operación por sus acciones sobre los objetos que el robot manipula.

3.8 Técnicas generales de programación

Programación explícita del ssistema: El operador es el responsable de las acciones de control y de las instrucciones adecuadas que las implementa. Es el que más se utiliza.

Modelación del mundo exterior: Se basa en cierta dosis de inteligencia. Con una amplia descripción de la tarea y del entorno, es el propio sistema el que lleva a cabo la toma de ciertas decisiones. En esta técnica existen dos subdivisiones: Programación Gestual y Programación Textual. (Ver figura 3.9.1)

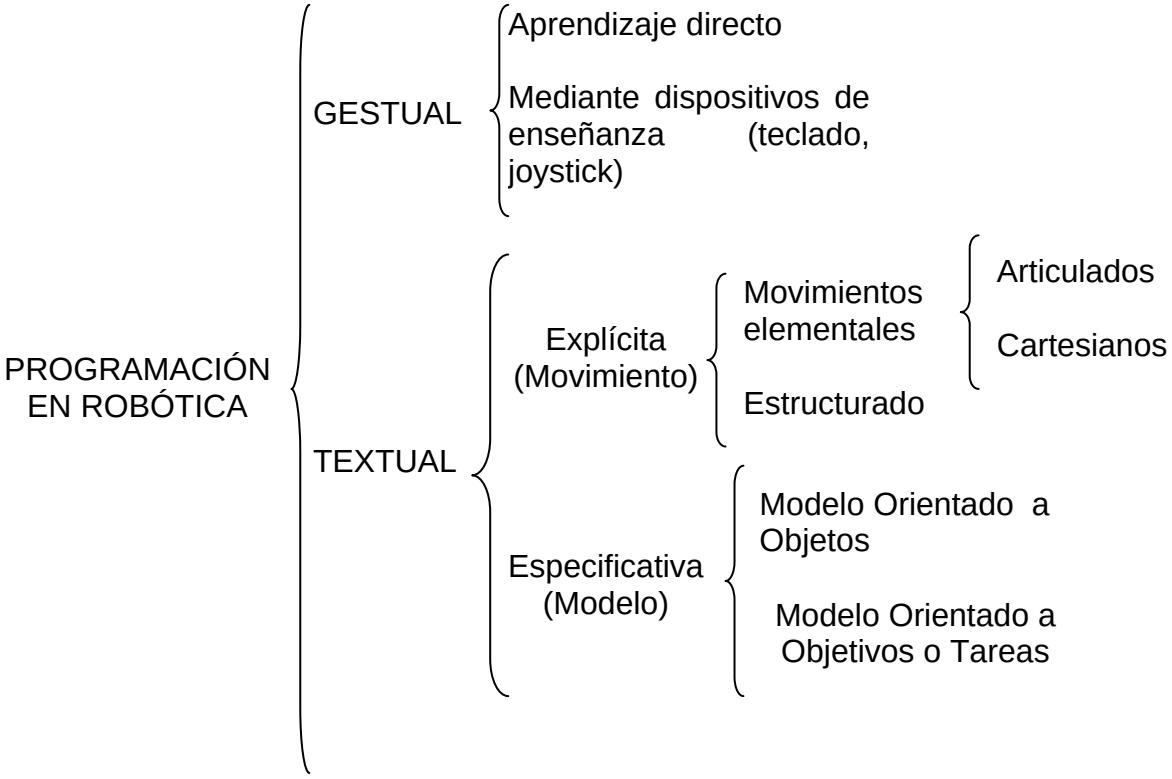


Figura 3.9.1 Clasificación de la Programación

3.8.1 Programación gestual

En este tipo de programación el operador guía el brazo directamente a través del camino que ha de seguir en su trabajo de aplicación. Posteriormente, el sistema repetirá ese camino cíclicamente, se conoce como programación on-line. El software se organiza aquí en forma de intérprete, de tal forma que no se necesita saber programar.

La programación gestual incluye las siguientes funciones:

- Selección de programación
- Generación de retardos
- Programación del estado de los sensores, tanto internos como externos.
- Borrado y modificación de los puntos de trabajo.

- Funciones especiales

3.8.2 Programación textual

El programa queda constituido por un texto de instrucciones o sentencias cuya concepción no requiere de la intervención del robot (se efectúa off-line). Es más exacto para operaciones industriales.

En la programación textual, la posibilidad de reedición es completa, por lo que se puede variar el código a nuestro antojo según las necesidades que se planteen.

Dentro de la programación textual existen dos grandes grupos de características netamente diferentes:

1) Programación textual explícita: El programa consta de una serie de órdenes que van definiendo con rigor las operaciones necesarias para llevar a cabo la aplicación. El programador debe tener en cuenta todos los supuestos.

2) Programación textual Especificativa: Se trata de una programación de tipo no-procesal en la que el usuario describe las especificaciones de los productos mediante una modelización al igual que las tareas que hay que realizar sobre ellos. El sistema informático para la programación textual especificativa ha de disponer de un modelo del *Universo* (el objeto y su entorno y las relaciones con otros objetos de ese entorno) o mundo donde se encuentra el robot.

   Este modelo será normalmente una base de datos más o menos compleja y requerirá de una computación potente. El trabajo de la programación consistirá en la realización de las tareas a realizar, lo que supone llevar a cabo trabajos complicados

3.9 Brazos robóticos

Los robots sin brazos están limitados a moverse sobre ruedas o andar, el robot no puede alcanzar ni tocar algo y no puede por tanto manipular su entorno.

Los robots más sofisticados en la ciencia, industria e investigación y desarrollo tienen al menos un brazo para sujetar, reorientar o mover objetos. Los brazos extienden el alcance de los robots y los hacen más parecidos a los humanos.

En el brazo humano observaremos varios puntos importantes. Primero, sin duda, son mecanismos enormemente adaptables. El brazo es capaz de maniobrar en cualquier posición que se desee, para ello, tienen dos articulaciones principales: el hombro y el codo (la muñeca, hasta donde la robótica trata, se considera parte del mecanismo de la mano). El hombro se puede mover en dos

planos, arriba y abajo, hacia detrás y hacia delante. Si se mueven los músculos del hombro hacia arriba, el brazo entero se levanta separándose del cuerpo. Si se mueven los músculos del hombro hacia delante, el brazo entero se mueve hacia delante. La articulación del codo es capaz de moverse en dos planos: atrás y adelante, arriba y abajo.

Las articulaciones del brazo y su capacidad de moverse se llaman grados de libertad. El hombro ofrece dos grados de libertad por sí mismo: rotación del hombro y flexión del hombro. La articulación del codo añade un tercero y cuarto grados de libertad: la flexión del codo y la rotación del codo. Los brazos robóticos también tienen grados de libertad. No obstante en lugar de músculos, tendones, rótulas y huesos, los brazos robóticos están hechos de metal, plástico, madera, motores, electroimanes, engranajes, poleas y otros componentes mecánicos. Algunos brazos robóticos solo proporcionan un grado de libertad; otros proporcionan tres, cuatro, incluso cinco grados distintos de libertad.

Los brazos robóticos se clasifican por la forma del área que el extremo del brazo (donde se coloca la pinza) puede alcanzar. Esta área accesible se llama "envolvente de trabajo". En beneficio de la simplicidad, la envolvente de trabajo no tiene en consideración el movimiento del cuerpo del robot sino solo los mecanismos de brazo.

3.9.1 Configuraciones de brazos robóticos

Cuando se habla de la configuración de un robot, se habla de la forma física que se le ha dado al brazo del robot.

Frecuentemente es necesario que un brazo robótico extienda su pinza hacia un objeto en línea recta para introducirse por alguna estrechez. Para llevar a cabo esto es necesaria la utilización de una trigonometría bastante compleja.

En un robot, un brazo robótico capaz de tener una envolvente esférica se diría que tiene coordenadas de revolución. Los otros tres tipos importantes de brazos robóticos son coordenadas polares, coordenadas cilíndricas y coordenadas cartesianas o rectangulares. Se observará que hay tres grados de libertad en estos cuatro tipos básicos de brazos robóticos.

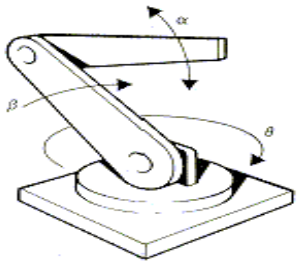
Coordenadas de revolución

Los brazos con coordenadas de revolución se modelan a partir del brazo humano, de modo que tengan muchas de sus capacidades. El diseño típico es algo diferente a causa de la complejidad de la articulación del hombro humano.

La articulación del hombro humano consta realmente de dos mecanismos. La rotación del hombro se consigue mediante el giro del brazo en su base, casi como si el brazo estuviera montado en una plataforma giratoria. La flexión del brazo se consigue moviendo la parte superior del brazo adelante y atrás.

La flexión del codo trabaja justo como en el brazo humano, el antebrazo se mueve arriba y abajo.

Los brazos de coordenadas de revolución son un diseño muy elegido para los robots para aficionados, proporcionan mucha flexibilidad y, además, parecen brazos reales.



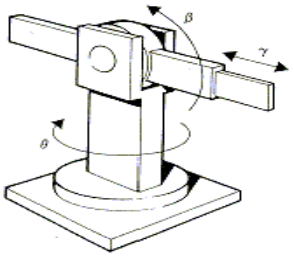
Coordenadas polares

La envolvente de trabajo del brazo de coordenadas polares tiene forma semiesférica. Los brazos de coordenadas polares tienen un diseño cercano al de coordenadas de revolución y son los más flexibles en términos de poder coger una gran variedad de objetos esparcidos alrededor del robot.

Una plataforma giratoria rota al brazo entero, igual que en el brazo de coordenadas de revolución. Esta función es análoga a la rotación del hombro; sin embargo, al brazo de coordenadas polares le falta un modo de flexionar el hombro.

Su segundo grado de libertad es la articulación del codo, que mueve el antebrazo arriba y abajo. El tercer grado de libertad se consigue variando el alcance del antebrazo. Se extiende o se retrae un antebrazo interior para llevar la pinza más o menos lejos del robot. Sin el antebrazo interior el brazo sólo podría alcanzar objetos colocados en un círculo finito bidimensional frente a él, en lugar de en una esfera, lo que no sería muy útil.

El brazo de coordenadas polares se usa a menudo en robots de fabricación, encontrando su mayor aplicación como dispositivo estacionario. No obstante, puede ser montado sobre un robot móvil para incrementar su flexibilidad.

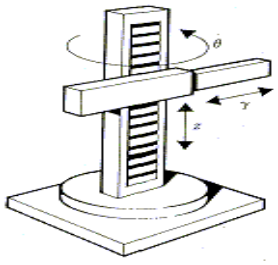


Coordenadas cilíndricas

El brazo de coordenadas cilíndricas se parece un poco a una horquilla elevadora robótica. Su envolvente de trabajo se asemeja a un cilindro grueso, de ahí su nombre. La rotación del hombro se consigue mediante una base que gira, como en los brazos de coordenadas de revolución y de coordenadas polares. El

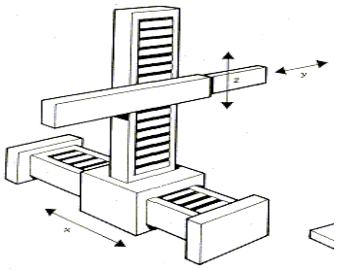
antebrazo se fija a un mecanismo elevador y se mueve arriba y debajo de esta columna para agarrar objetos de varias alturas.

Para permitir al brazo alcanzar objetos en un espacio de tres dimensiones, se dota al antebrazo con un mecanismo de extensión similar al descrito en el brazo de coordenadas polares.



Coordenadas rectangulares

La envolvente de trabajo del brazo de coordenadas cartesianas se parece a una caja, es el brazo más diferente a un brazo humano y a los demás tipos de brazos robóticos, no tiene componentes giratorias. La base posee una cadena que mueve la columna elevadora arriba y abajo, y tiene un brazo interior que extiende el alcance más cerca o más lejos del robot.



3.9.2 Estructura fundamental

La mayoría de los brazos robóticos tienen cinco ejes que les permiten una gran libertad de movimiento. Cada eje funciona por su propio motor o, en el caso de grandes brazos, por un cilindro hidráulico. Un brazo robótico de seis ejes contiene, en realidad, doce estructuras coordinadas.

La base y las dos partes del brazo forman un sistema de coordenadas X, Y, Z, y la pinza permite movimientos delicados dentro del sistema. Por tanto, encontrar cualquier punto específico en el espacio requiere el uso de cinco juntas en el brazo. El movimiento del sexto eje supone la rotación de la pinza lo que afecta solo a la orientación.

La mayor dificultad con la que se encuentra el control de un brazo de este tipo no es hacer el movimiento preciso -de lo que se encarga el hardware-, sino, controlar el movimiento de las seis articulaciones.

Para que un brazo robótico funcione correctamente, todas las juntas deben trabajar al unísono, tal como lo hace un brazo humano.

3.10 Sistemas de impulsión

Hay tres maneras en general de mover las articulaciones de un brazo robótico: Eléctrico, hidráulico y neumático.

Eléctrico

Se le da el nombre de impulsión eléctrica cuando se usa la energía eléctrica para que el robot ejecute sus movimientos. La impulsión eléctrica tiene que ver con el empleo de motores, electroimanes y otros dispositivos electromecánicos, es la más sencilla y común de aplicar.

Es utilizada en robots de tamaño mediano, pues éstos no requieren de tanta velocidad ni potencia. Los robots que usan la energía eléctrica se caracterizan por una mayor exactitud y repetibilidad.

Hidráulico

El sistema de impulsión hidráulica es en la que se utiliza un fluido, generalmente un tipo de aceite, para que el robot pueda movilizar sus mecanismos. La impulsión hidráulica se utiliza para robots grandes, los cuales presentan mayor velocidad, potencia y mayor resistencia mecánica.

Neumática

La actuación neumática es análoga a la hidráulica, excepto que se emplea aire comprimido en lugar de aceite u otro fluido. Tanto los sistemas hidráulicos como los neumáticos proporcionan más potencia que los sistemas eléctricos, pero son más difíciles de usar.

Los robots pequeños están diseñados para funcionar por medio de la impulsión neumática. Los robots que funcionan con impulsión neumática están limitados a operaciones como la de tomar y situar ciertos elementos. Es importante señalar que no todos los elementos que forman el robot pueden tener el mismo tipo de impulsión.

3.11 Aplicaciones

Teóricamente el uso de brazos robóticos podría extenderse a casi todas las áreas imaginables en donde se necesite de la ejecución de tareas mecánicas, tareas hoy ejecutadas por el hombre o imposibles de ejecutar por él (por ej. una exploración sobre el terreno de la superficie marciana). Se entiende, en este contexto, que tarea mecánica es toda actividad que involucra presencia física y movimiento por parte de su ejecutor. Pero al situarnos en el contexto real, en la práctica, nos damos cuenta de que existen factores que limitan el vuelo de nuestra imaginación, los que mencionaremos en el siguiente punto.

Algunos de los campos de aplicación actuales de la robótica son:

Investigación-Exploración

En donde los robots presentan la ventaja de resistir mejor un medioambiente hostil para el ser humano: La colocación de tubos de pruebas dentro de los instrumentos de medición, sistema de preparación de muestras, las suturas automáticas, operar a corazón latiendo, entre otros.

Entretenimiento-Didáctica

Esta industria se favorece del uso de robots para recrear situaciones ficticias o posibles, haciendo uso de los llamados "efectos especiales".

Construcción

Industria en que ya se registran proyectos que incluyen el uso de robots como ejecutores de tareas de dimensionamiento, transporte y montaje, entre otras.

Automatización industrial

Corresponde al uso de robots en la industria a fin de mejorar, agilizar y aumentar la producción en los diferentes procesos: Aplicación de transferencia de material carga y descarga de maquinas, operaciones de procesamiento (Soldadura por puntos, soldadura por arco continua, recubrimiento con spray, corte por chorro de agua, taladro y corte por láser, etc).

3.12 Características técnicas del brazo robótico

Capacidades: Robot de segunda generación, 5 grados de libertad, coordenadas de revolución.

- 5 ejes de movimiento.
- Rotación de la base: 350°.
- Rango de movimiento del hombro: 120°.

- Rango de movimiento del codo: 135°.
- Giro de muñeca: 340°.
- Abertura de mordaza: 50mm.
- Peso máximo a levantar: 130 g.

Inteligencia: Robot controlado por computadora.

Control: Posee nivel de inteligencia artificial.

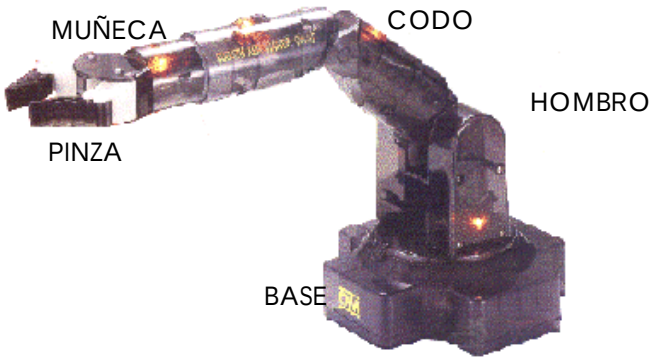
Programación: Sistema de programación guiado, técnica de programación explícita.

Sistema de Impulsión: Eléctrico mediante motores de CD.

Alimentación: 5 Volts.

Aplicaciones: Entretenimiento/Didáctica.

Dimensiones: Largo máximo hacia delante: 360 mm, altura máxima extendido hacia arriba: 510 mm.



Brazo Robótico

Capítulo 4

Control de motores en el brazo robótico

4.1 Definición de motor

Es un sistema basado en las leyes del electromagnetismo básicas, convierte la energía eléctrica en energía mecánica. Además es un sistema reversible, ver figura 4.1.1.

4.1.1 Definición del generador

Es una máquina que, basada en las leyes del electromagnetismo básicas sirve para convertir la energía mecánica en eléctrica.

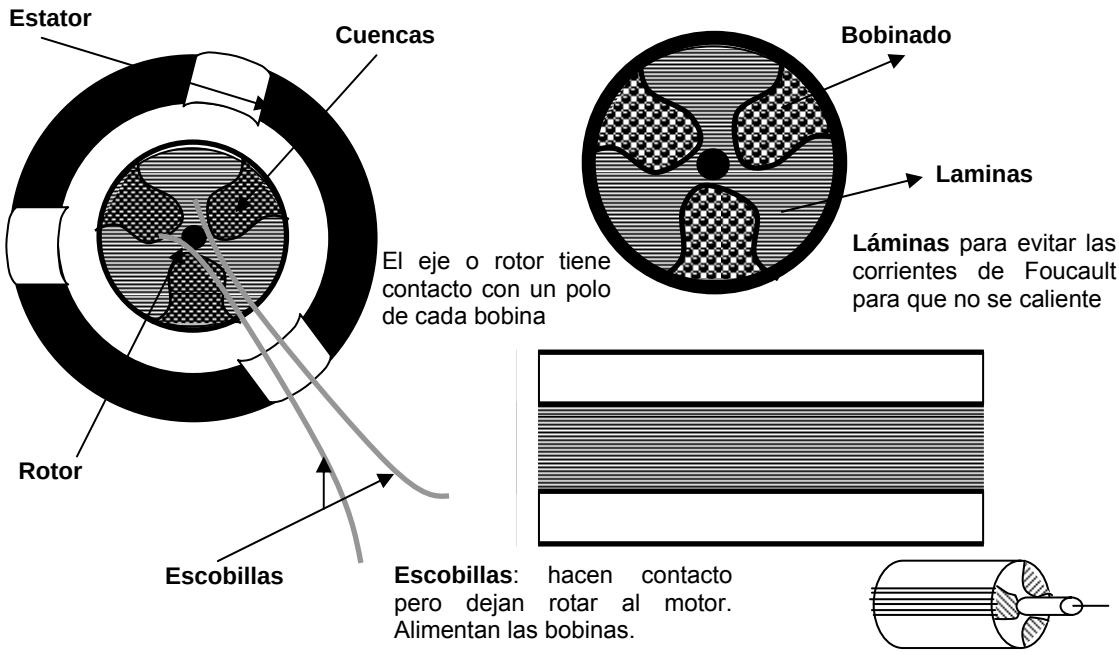
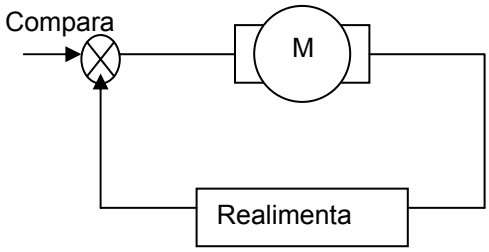


Figura 4.1.1 Sección de un motor eléctrico

Destaca en especial la utilización de los motores de corriente directa por su elevada relación par/velocidad que los hacen más apropiados en muchas aplicaciones, por otra parte también destaca la sencillez de control y su fácil

adaptación a los circuitos electrónicos basados en microprocesadores. Una función de los motores de corriente directa eléctricos es la siguiente:

Si alimentamos el motor, éste toma una muestra de la señal de entrada, al ser un sistema realimentado al final hace una comparación con la señal de entrada y la que le llega después, ver figura 4.1.2.



***** La realimentación determinará la posición en la que se encuentra el motor en ese momento.

Figura 4.1.2 Sistema de motor realimentado

Por su bajo costo y la supresión de la realimentación en la determinación de la posición del eje, los motores paso a paso son muy interesantes.

4.2 Principios generales del electromagnetismo

Se ha comprobado experimentalmente que un conductor recorrido por una corriente eléctrica y colocado en un campo magnético, está sometido a la acción de una fuerza de tipo electromagnética. Se genera un campo magnético en el sentido que muestran las flechas, ver figura 4.2.1.

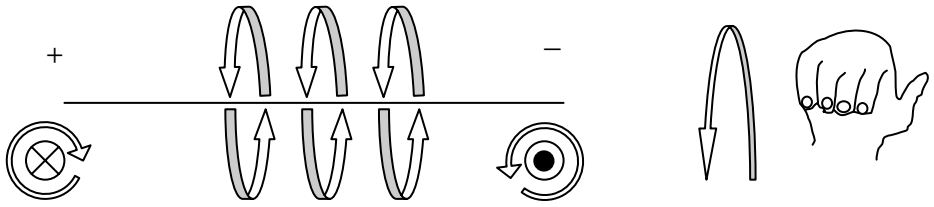


Figura 4.2.1 Conductor sometido a la acción de una fuerza de tipo electromagnética

4.2.1 Principio del motor

Si se toma un elemento diferencial de conductor de una longitud dl sometido a la acción de un campo magnético B que es recorrido por una intensidad I que forma un ángulo θ con la dirección de campo, ver figura 4.2.1.1.

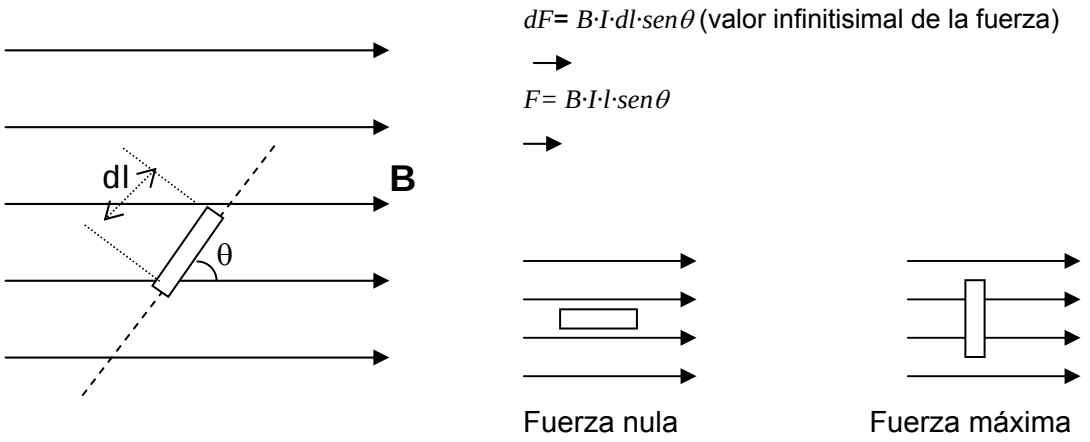


Figura 4.2.1 1 Principio fundamental del Motor

La fuerza electromagnética presenta una dirección perpendicular al plano de la figura 4.2.1 2 desplazando al conductor en un sentido opuesto al observado. En el caso de que el eje del conductor sea perpendicular al campo magnético la expresión quedaría:

$F=B\cdot I\cdot l$

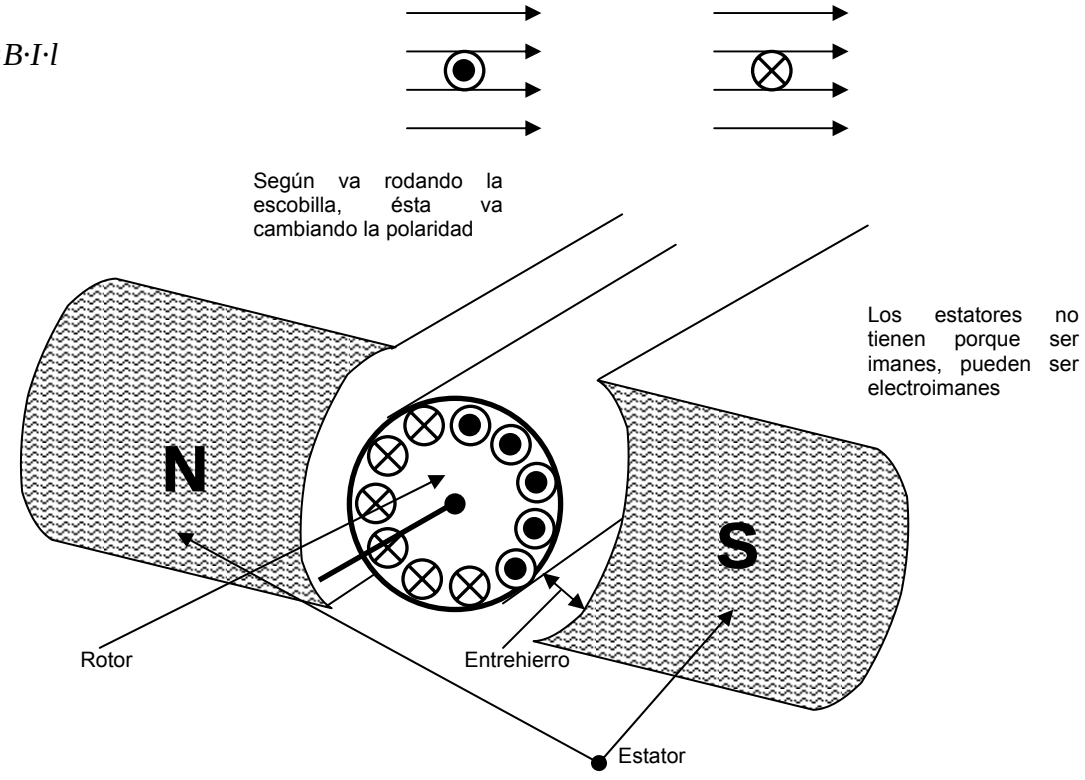


Figura 4.2.1 2 Fuerza electromotriz inducida

El inducido en el que se distribuye un bobinado de Z conductores, alojados en ranuras practicadas en un paquete de chapas magnéticas, se sitúa sobre un cuerpo rotor o eje dispuesto sobre cojinetes que permiten su fácil rotación. Los Z conductores del bobinado del inducido se encuentran bajo la influencia del campo

inductor producidos por los polos norte y sur repartidos regularmente en el interior de la carcasa del motor, ver figura 4.2.1.3.

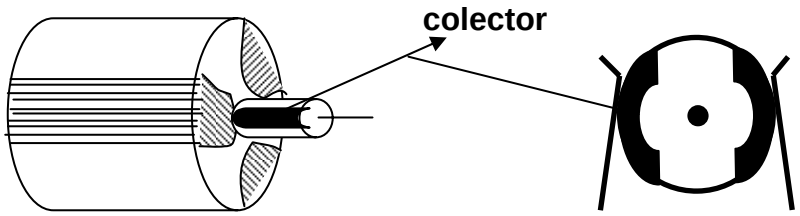


Figura 4.2.1 3 Representación del inducido (rotor) y del inductor (estator)

Los conductores del inducido colocados axialmente en el interior de las ranuras presenta una dirección perpendicular al flujo del entrehierro (lo más pequeño posible teniendo en cuenta la ventilación del motor.

El sistema constructivo del motor con sus dispositivos de portaescobillas y escobillas repartidas sobre la superficie del colector, determina que, con independencia de la posición del inducido sobre su giro, las corrientes absorbidas de la red de alimentación, circulen en una dirección en los conductores situados bajo un polo inductor N (norte) y en una dirección bajo un inductor S (sur), figura 4.2.1.4.

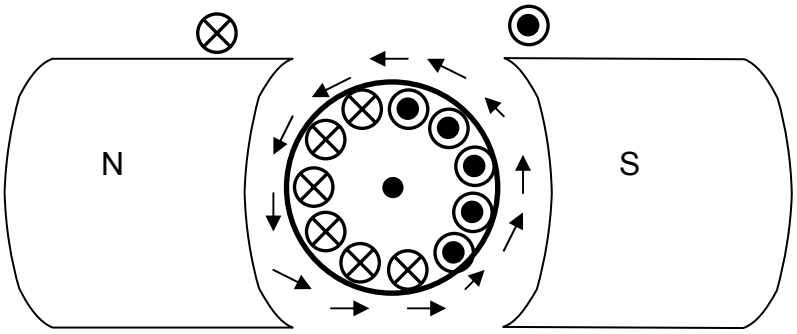


Figura 4.2.1 4 Sistema constructivo del motor

Este hecho da lugar a la aparición de un par motor que hace girar el motor.

4.3 Motores de corriente directa

Los motores de CD son “máquinas de CD” utilizadas como motores. Existen diversos tipos de motores que pueden construirse y cada uno tiene ventajas y desventajas.

Hay varias razones que explican la popularidad de los motores de CD. Una es que los sistemas de potencia todavía son comunes en automóviles, camiones y en la aviación. Una aplicación de los motores de CD es en donde se necesitan amplias variaciones de velocidad, hace tiempo los motores de CD eran insuperables, no obstante, todavía se comercializan y se instalan muchos motores de CD para éstos propósitos.

La estructura física de una máquina de CD consta de dos partes: El estator o parte estacionaria y el rotor o parte giratoria. La parte estacionaria de la máquina consta de carcaza, que provee el soporte físico, y las piezas polares que se proyectan hacia adentro y suministran un trayecto para el flujo magnético de la máquina.

Hay dos embobinados principales en una máquina de CD:

- Los embobinados del inducido
- Los embobinados de los inductores.

Los embobinados del inducido se definen como aquellos en los que se induce un voltaje, y los embobinados de los inductores se definen como los que producen el flujo principal en la máquina. En una máquina normal de CD, los embobinados del inducido se localizan en el rotor y los embobinados del inductor están localizados en el estator. Por esta circunstancia, un rotor de una máquina de CD es llamado armadura y el estator es llamado campo, ver figura 4.3.1.

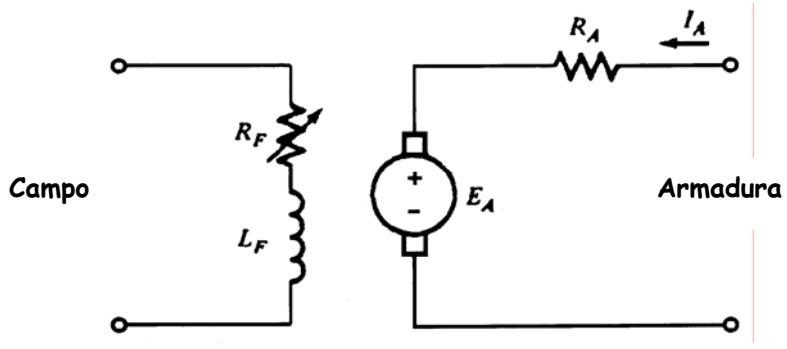
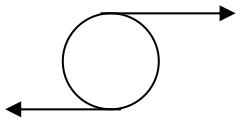


Figura 4.3.1 Circuito equivalente de un motor de CD.

4.3.1 Par motor

Los pares motores son unas fuerzas paralelas sobre un dispositivo rígido que gira creando un momento.

$T = F \cdot R \cdot Z$



$F = B \cdot I \cdot l \cdot sen\,\vartheta$

$\vartheta = 90^o \quad sen\,\vartheta = 1$

$T = Z \cdot B \cdot I \cdot l \cdot R$

$B = \phi / S$

$K = Z + I + R$

S= cte -> k = Z+I+R+S

$T = K \cdot \phi \cdot I$

F = fuerza
R = radio del motor
Z = número de conductores = constante
L = constante
T = Par del Motor
K = Suma e las constantes
S = área total sobre la que K distribuye el flujo magnético

4.3.2 Tensiones generales

En todo conductor que se mueve a través de un campo magnético se genera una tensión.

Si existe un flujo magnético de valor constante y uniforme ϕ (se mide en Webers) y en su interior se mueve un conductor, se genera entre los extremos A y B una tensión de tal forma que cuanto más aumenta la velocidad más aumentará la fuerza electromotriz (fem) entre A y B, ver figura 4.3.2.1.

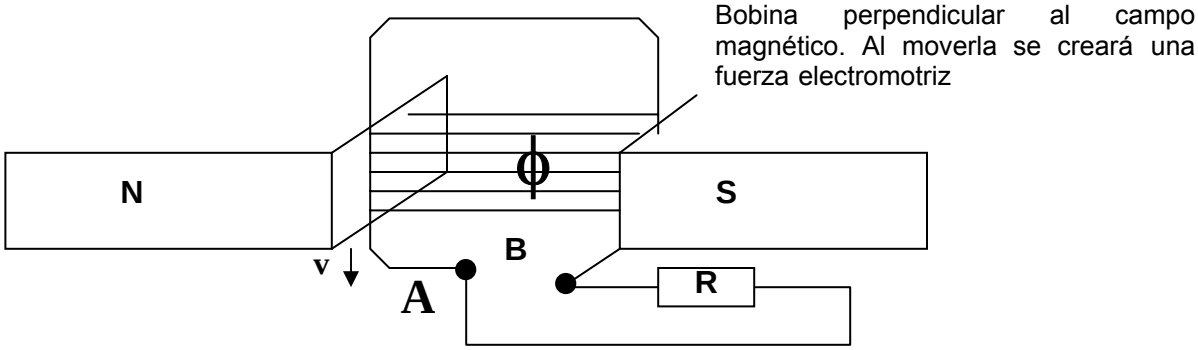
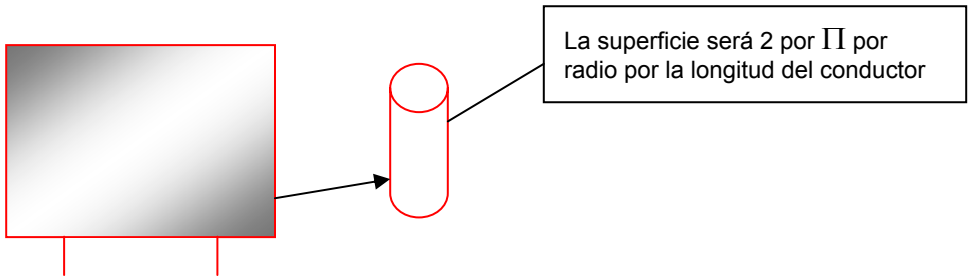


Figura 4.3.2.1 Representación de una bobina perpendicular en un campo magnético.

Cuando la velocidad es constante, la aceleración es nula.

ϕ = cte
v = uniforme
a = 0
t = tiempo en seg. que tarda en recorrer el campo.

Tensión generada = cte = ϕ / t



Si el flujo de ϕ no varía:

Pequeña fuerza Electromotriz = $e = d\phi / dt$

Cuando L, v y B son perpendiculares podemos aplicar las dos fórmulas anteriores: $e = d\phi / dt$ y Tensión generada $= \phi / t$

Teniendo en cuenta que no es el número total de conductores el que contribuye a la tensión interna, sino que hay en cada uno de los circuitos en derivación (Z/Za), se obtiene que:

$$e = B \cdot L \cdot v \cdot Z/Za$$

$$B = \phi / Pp \cdot L$$

$$Pp = \pi \cdot D / 2p$$

$$v = 2 \cdot \pi \cdot r \cdot n$$

Pp = Paso polar (el motor puede tener varios polos y es el graduaje entre los polos)

L = Longitud de la bobina

2p = el número de polos (siempre va de dos en dos)

D =diámetro

n = revoluciones por segundo

$e = B \cdot L \cdot v \cdot Z/Za$ =>

$$e = 2p \cdot \phi \cdot Z/Za \cdot n$$

Cuantos más polos, más energía electromotriz

Por tanto, la tensión interna generada es igual al producto del flujo total (ϕ) por el número de conductores de cada circuito en paralelo del inducido y por el número total de revoluciones por segundo de la máquina.

Teniendo en cuenta que en cada máquina $2p \cdot Z / Za$ es un valor constante que llamaremos *K*'

$$e = K' \cdot \phi \cdot n$$

Sin tener en cuenta lo anterior:

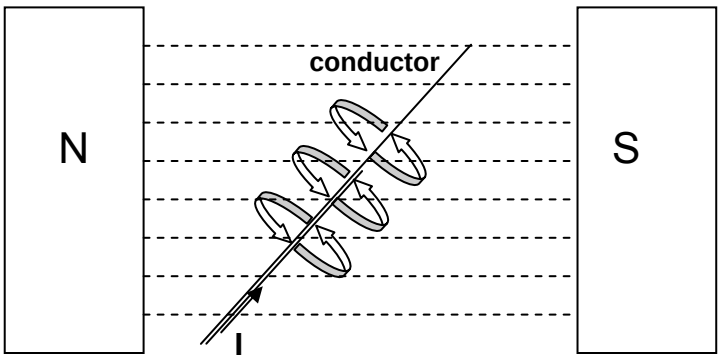
$$d\phi = B \cdot L \cdot dl$$

$$e = d\phi / dt = (B \cdot L \cdot dl) / dt \cdot dv$$

$$\boxed{e = B \cdot l \cdot v}$$

4.3.3 Fuerza contraelectromotriz

Al conectar la máquina en la red de alimentación se desarrolla un par que determina el ciclo del motor. La existencia de un campo inductor portando a Z conductores alojados en las ranuras del rotor que gira a una cierta velocidad, genera fuerzas electromotrices en estos conductores. *Se* cumple siempre que la tensión interna generada en el bobinado del inducido es opuesta a la tensión aplicada en bornes del motor, por eso se llama Fuerza Contraelectromotriz, porque va en sentido contrario al motor, ver figura 4.3.3.1.



Se crea un flujo magnético perpendicular al conductor, lo que significa que interactúa con el estator, esto provoca que el conductor comience a rotar. Al moverse el conductor, se crea una Fuerza Electromotriz, ésta Fuerza Electromotriz crea otros campos magnéticos perpendiculares. Todo esto significa una retroalimentación.

Figura 4.3.3.1 Representación de la fuerza contraelectromotriz.

4.4 Tipos de motores según su excitación

Hay dos tipos de motores, en serie o en paralelo (Shunt)

4.4.1 Motores serie

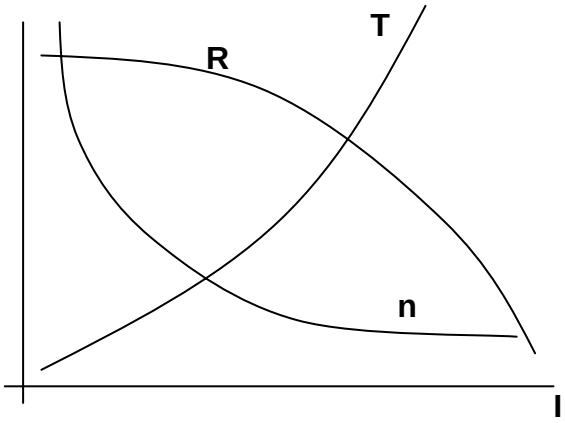
En este tipo de motores, el bobinado de excitación (bobinado del estator) se conecta en serie con el bobinado del inducido. Los motores en serie tienen varios usos:

- Tracción eléctrica: Conseguir arrastrar cosas...
- Trenes de laminación: Para laminar con rodillos...
- Grúas Puente: Se usan en naves para trasladar cosas...
- Cintas transportadoras...

Estos motores se utilizan por su dureza y durabilidad en sus servicios.

4.4.1.1 Características de funcionamiento

Si se expresan con un gráfico, ver figura 4.4.1.1.



I = Intensidad absorbida por la línea.

T = Par

R = Rendimiento

N = Velocidad

Figura 4.4.1.1 Curva característica Par – Intensidad de un Motor Serie.

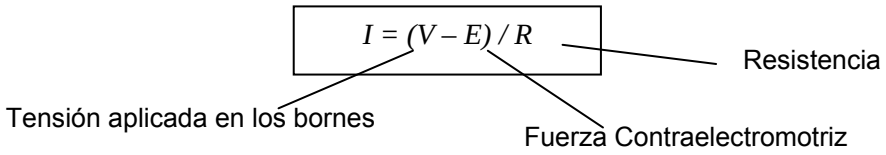
4.4.1.2 Sentido de giro

En un motor serie, la inversión de su sentido de giro se puede conseguir invirtiendo la dirección de la corriente en el bobinado inductor o en el inducido. Para cambiar el sentido de giro sólo tenemos que cambiar la polaridad del motor, saldrá o entrará del bobinado.

4.4.1.3 Fuerza contraelectromotriz e intensidad absorbida

La tensión interna generada, nula en el arranque, aumenta gradualmente de valor, hasta alcanzar un límite impuesto por la corriente necesaria para establecer un par motor igual al resistente solicitado por la máquina accionada.

Resistente solicitado -> lo que solicita la máquina que usa el motor.



En el arranque, el valor de E es nulo y la Intensidad de arranque puede ser muy elevada, de allí que sea necesario colocar resistencias intercaladas entre la línea y los bornes del motor.

El caso anterior hace que la intensidad sea muy alta, por lo cual sumaremos una resistencia R' suplementaria para estabilizar el motor. Cuando conseguimos una fuerza contraelectromotriz, quitamos R'. Estas resistencias suelen estar temporizadas por relés.

$I = V / (R + R')$

Velocidad

$I = (V - E) / R$ $\qquad$ $E = K' \cdot \phi \cdot n$

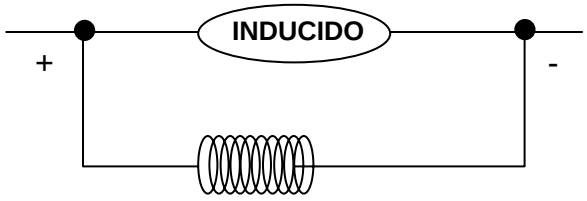
$$n = E / (K' \cdot \phi) = (V - I \cdot R) / (K' \cdot \phi)$$

$$n = (V - I \cdot R) / (K' \cdot \phi)$$

Para un flujo magnético y eléctrico constantes, la velocidad dependerá de la tensión aplicada en los bornes.

4.4.2 Motores Shunt

Los motores en derivación, conocidos como motores Shunt, se caracterizan porque el bobinado de excitación se encuentra en paralelo con el bobinado del inducido ver figura 4.4.2.1.



***** Se suele tomar de 2 a 4 Voltios

Figura 4.4.2.1 Representación de un Motor en Derivación (Motor Shunt)

La velocidad es sensiblemente constante entre el vacío y la plena carga.

Vacío: el motor gira con el eje libre sin carga

Plena Carga: toda la energía es absorbida por la carga ver figura 4.4.2.2.

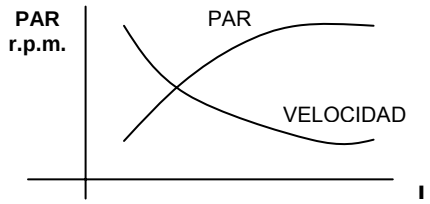


Figura 4.4.2.2 Curva característica Par – Intensidad de un Motor en Derivación.

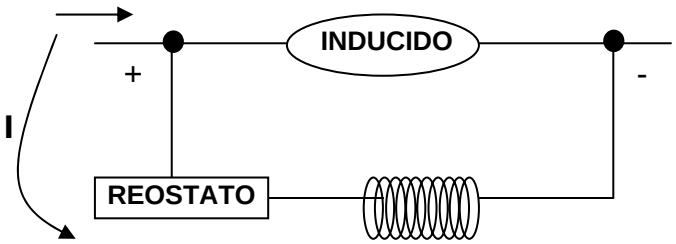
Esta clase de motores se usa para ventiladores industriales, bombas centrífugas y máquinas herramienta.

4.4.2.1 Métodos para el control de velocidad en los motores Shunt

- Control del campo inducido: Cuanto mayor es el campo más disminuye la velocidad.
- Control de la tensión del inducido: Cuanta más tensión caiga en la bobina del motor Shunt, más velocidad adquirirá este.
- Control de combinación: Es una combinación de los dos métodos anteriores.
- Control del inducido mediante resistencias: Intercalar una resistencia para disminuir la velocidad del motor, ya que la tensión en los bornes disminuye.

4.4.2.2 Regulación de la velocidad del motor mediante el campo inducido

Un decremento de la Intensidad de excitación manteniendo constante la tensión V, provoca una disminución del flujo y conlleva un aumento de la velocidad n ver figura 4.4.2.2.1.



El reostato es una resistencia variable. Lo ajustamos según queramos aumentar o disminuir la velocidad

Figura 4.4.2.2.1 Representación de regulación de velocidad por campo inducido.

4.4.3 Motores Compound

También se llaman motores de excitación compuesta. Se distinguen por disponer de un bobinado inductor en serie cuyas fuerzas electromotrices se suman a las del bobinado inductor Shunt, ver figura 4.4.3.1.

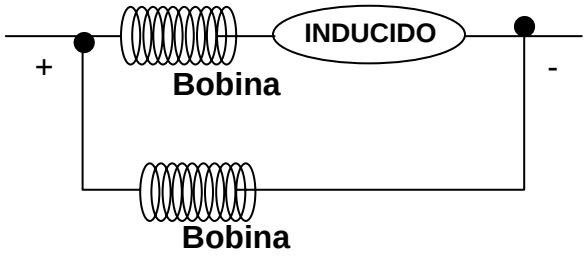


Figura 4.4.3.1 Representación de un Motor Compound.

Estos motores se aplican en trenes de laminación, grúas puente, palas excavadoras, cintas transportadoras, etc., ver figura 4.4.3.2.

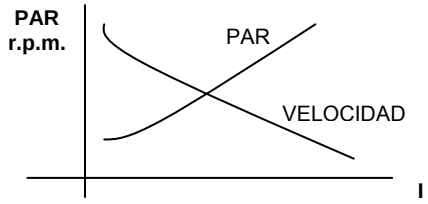


Figura 4.4.3.2 Curva característica Par – Intensidad de un Motor Compound.

Para cambiar el sentido de giro se realiza por el cambio de sentido de la corriente del inducido o del inductor indistintamente.

4.5 Motor de CD de imán permanente (Permanent Magnet, PM)

El motor de imán permanente es en el cual los polos están hechos de imanes permanentes.

Puesto que el campo magnético del estator de un motor de imán permanente (Permanent Magnet, PM) es generado por imanes permanentes, no es usada energía en la estructura del campo. El flujo magnético del estator permanece esencialmente constante en todos los niveles de las corrientes de armadura y, por consiguiente, la curva par motor-velocidad del motor de imán permanente es lineal sobre un rango extendido.

Con modernos imanes de cerámica, el par motor el cual si entra en pérdida de velocidad tenderá a ser alto y la curva par motor velocidad tenderá a ser más lineal que para un comparable motor de campo devanado.

Las razones para las curvas par motor-velocidad no lineales en el caso de un motor en derivación es aquella reacción al flujo de la armadura (el cual siempre es ortogonal al flujo del estator principal en cualquier motor de corriente directa) que tiende a seguir las trayectorias de baja reluctancia a través del patín del polo, y en niveles de corrientes altos causa un efecto de red de desplazamiento angular en la localización del polo y un nivel de flujo inferior efectivo.

Estos motores ofrecen diversos beneficios en comparación con los motores de CD en derivación en algunas aplicaciones. Como no requieren un circuito de campo externo, no tienen las pérdidas de cobre del circuito de campo, además pueden ser más pequeños que los motores de CD en derivación.

En el caso de los imanes de cerámica permanentes, la reacción del flujo de la armadura permanece ortogonal al flujo de imanes permanentes, puesto que la permeabilidad del material de cerámica de imán es muy baja (casi igual que la del aire) En suma la alta fuerza coercitiva de los materiales magnéticos resiste cualquier cambio en flujo cuando entra la reacción del campo de la armadura. El resultado es una característica lineal del par motor –velocidad.

Los motores de imán permanente ofrecen varias ventajas. Quizás la ventaja más obvia es que la fuerza eléctrica necesaria no es suministrada para generar el flujo magnético del estator puesto que la conversión de potencia eléctrica a potencia mecánica toma lugar en el devanado de armadura, la potencia suministrada al devanado de campo resulta la mayoría de las veces en una pérdida RI^2 (Pérdida de calor) en el devanado mismo. El motor de imán permanente así simplifica los requerimientos de fuerza, mientras que al mismo tiempo requiere menos enfriamiento.

Otro beneficio del motor de imán permanente es un tamaño de cuadro reducido para una potencia de salida dada. A causa de la alta fuerza coercitiva de los imanes permanentes, sus dimensiones radiales son típicamente un cuarto de las de un motor de campo devanado para un entre hierro de aire dado.

Las ventajas significativas de los motores de imán permanente sobre los tipos de campo devanado son resumidas como sigue:

- Característica lineal par motor-velocidad
- Entrada en pérdida de alta velocidad (aceleración) par motor.
- No necesita fuerza eléctrica para generar el flujo magnético.

- Un cuadro muy pequeño y motor ligero para una potencia de salida dada.

Sin embargo también tienen desventajas. Los imanes permanentes no pueden producir una densidad de flujo alta, por lo que estos motores tendrán un menor momento inducido (T_{ind}). Además presentan el riesgo de desmagnetización debido a la reacción de la armadura (campo magnético de la armadura), si la corriente se hace muy grande. El flujo magnético de este tipo de máquina es fijo, por lo tanto, no es posible controlar su velocidad variando la corriente de campo ni el flujo. Los únicos métodos disponibles de control de velocidad para este tipo de motor, son el control de voltaje en la armadura y el control de resistencia en la armadura.

4.6 Motores paso a paso

Un motor paso a paso como todo motor, es en esencia un conversor electromecánico, que transforma la energía eléctrica en mecánica; pero de un modo tan peculiar que constituye en la actualidad una categoría aparte.

En efecto, mientras que un motor convencional gira libremente al aplicar una tensión comprendida dentro de ciertos límites (que se corresponden de un lado al par mínimo capaz de vencer su propia inercia mecánica, y de otro a sus propias limitaciones de potencia); el motor paso a paso está concebido de tal manera que gira un determinado ángulo proporcional a la "codificación" de tensiones aplicadas a sus entradas (4, 6, etc.) La posibilidad de controlar en todo momento esta codificación permite realizar desplazamientos angulares lo suficientemente precisos, dependiendo el ángulo de paso (o resolución angular) del tipo de motor (puede ser tan pequeño como 1,8º hasta unos 15º) De este modo, si por ejemplo el número de grados por paso es de 1,8º, para completar una vuelta serán necesarios 200 pasos.

De la misma manera que se puede posicionar el eje del motor, es posible controlar la velocidad del mismo, la cual será función directa de la frecuencia de variación de las codificaciones en las entradas. De ello se deduce que el motor paso a paso presenta una precisión y repetitividad que lo habilita para trabajar en sistemas abiertos sin realimentación.

- Son sencillos y económicos con respecto a los servomotores CCC con realimentación. El problema de estos motores es que tienen una potencia limitada (1CV).
- El motor paso a paso es un elemento capaz de transformar pulsos eléctricos en movimientos mecánicos. El eje del motor gira un determinado ángulo por cada impulso de entrada, con lo que el movimiento es muy preciso y fiable.

- El motor paso a paso puede girar en los dos sentidos, y el ángulo de giro puede variar entre 0,72º (500 pasos/vuelta) y 90º (4 pasos/ vuelta).

4.6.1 Principios de funcionamiento

Ver figuras. 4.6.1 y 4.6.2.

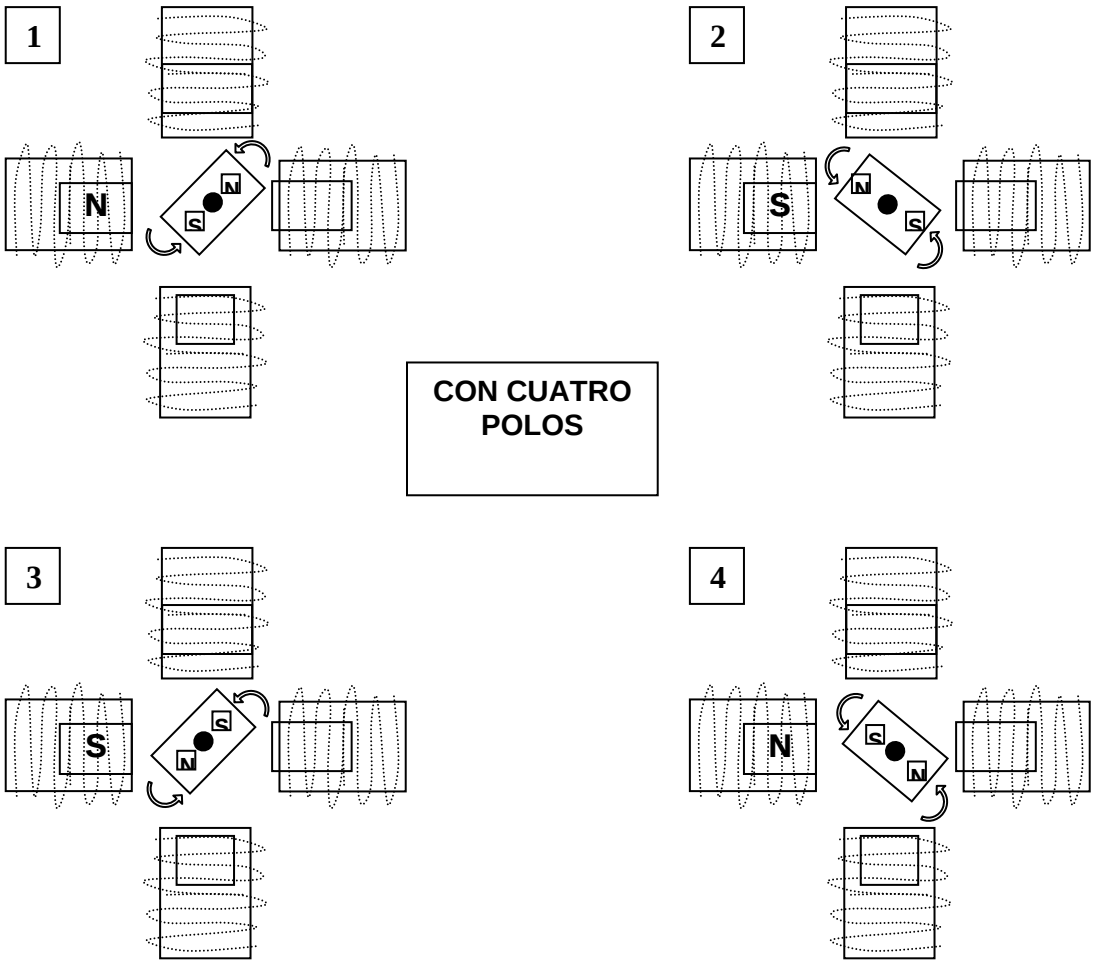


Figura 4.6.1 Motor Paso a Paso con cuatro polos

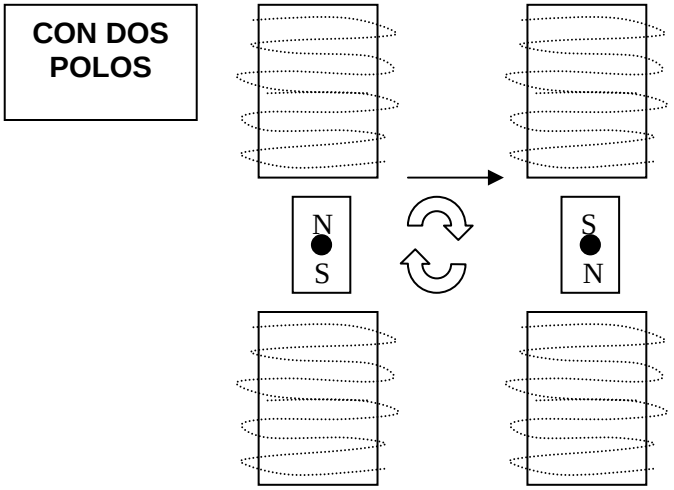


Figura 4.6.2 Motor Paso a Paso con dos polos

- Se modifican los polos para provocar el giro.
- El motor paso a paso perfecto sería el que tuviera polos infinitos, así se obtendrían giros de 0 grados.
- Para permitir una mejor resolución por paso, se añaden más polos al estator; además en dichos polos se mecanizan, ver figura 4.6.3

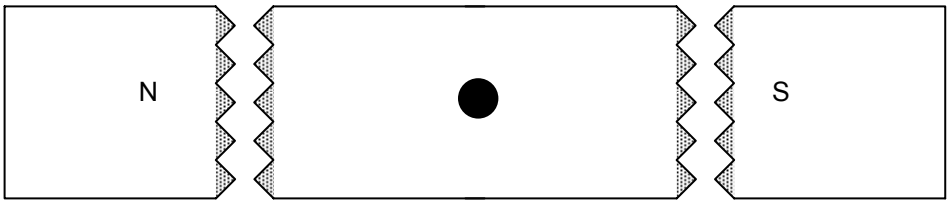


Figura 4.6.3 Mecanización de Motor Paso a Paso

4.6.2 Tipos de motores paso a paso

- De imán permanente: En lugar de electroimanes tienen imanes permanentes.
- De reluctancia variable: Se crea un campo magnético variable con reluctancias variables.
- Híbridos: Son una mezcla de los dos anteriores.

4.6.3 Características principales de los motores paso a paso

Además del giro, también hay otras características. El momento depende de cuatro factores:

- Velocidad de Paso: el tiempo que tarda en recorrer el arco de giro. Los impulsos deben estar sincronizados con la velocidad de paso para no perder tiempo.

- Corriente del devanado: la corriente que atraviesa las bobinas del motor paso a paso. Si el motor se queda quieto después de girar, la corriente usada para el nuevo impulso será muy elevada.

- Diseño del controlador

4.6.4 Motores paso a paso con rotor de imán permanente

En lo que se refiere a la esencia de su funcionamiento, un motor paso a paso clásicamente siempre se ha comparado a un motor síncrono: Un campo magnético rotativo, controlado aquí por un dispositivo electrónico, pone en funcionamiento al rotor, que es un imán permanente

En este tipo de motores, como en todos, cabe destacar dos partes principales (rotor y estator); como se puede ver en la figura 4.6.4.1, estos motores pueden constar de dos o más estatores, oportunamente bobinados.

En todo instante, el campo magnético producido por una de las fases en particular dependerá de la intensidad de corriente de esa fase. Si la intensidad es cero, el campo magnético también será nulo. Si la intensidad es máxima, el campo magnético tendrá una fuerza máxima.

Por otro lado, dado que el rotor es un imán permanente, si se permite el giro de éste dentro de un campo magnético, acabará por orientarse hasta la total alineación con el campo. De otro lado, si el campo magnético giratorio es intenso, se origina un par, capaz de accionar una determinada carga.

Dependiendo del tipo de bobinas que se encuentran devanadas simétricamente sobre los estatores (y, por tanto, del modo de crear el campo giratorio) se pueden clasificar estos tipos de motores en:

- Paso a paso bipolares.
- Paso a paso unipolares.

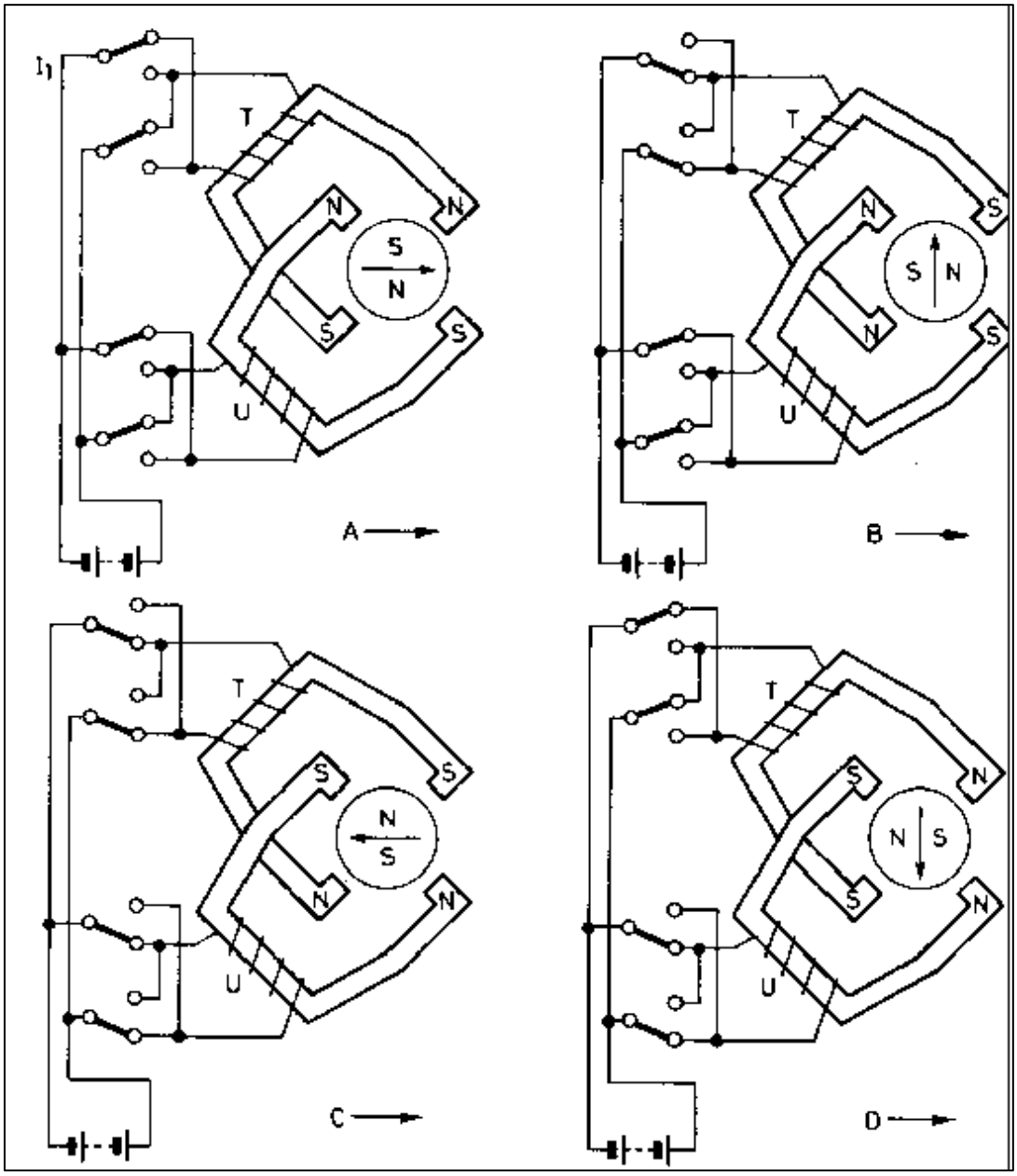


Figura 4.6.4.1 Motor Paso a Paso Bipolar en modo de pasos completos

4.6.4.1 Motores paso a paso bipolares

En el esquema de la figura 4.6.4.1 aparece uno de estos motores con dos estatores, sobre cada uno de los cuales se ha devanado una bobina (1 y U), las cuales se encuentran conectadas directamente a unos conmutadores de control que, como se verá más adelante, podrán ser sustituidos por las líneas de entrada y salida de nuestro ordenador debidamente programadas.

Como las bobinas se encuentran distribuidas simétricamente en torno al estator, el campo magnético creado dependerá en magnitud de la intensidad de corriente por cada fase, y en polaridad magnética, del sentido de la corriente que circule por cada bobina. De este modo el estator adquiere la magnetización correspondiente, orientándose el rotor según ella (figura 4.6.4.1A) Si el interruptor 1.1 se conmuta a su segunda posición (figura 4.6.4.1B), se invierte el sentido de la corriente que circula por T y por tanto la polaridad magnética, volviéndose a reorientar el rotor (el campo ha sufrido una rotación de 90º en sentido antihorario), haciendo girar el rotor 90º en ese mismo sentido. Con esto se llega a la conclusión de que para dar una vuelta completa serían necesarios cuatro pasos de 90º cada uno (el ciclo completo se puede seguir en la figura 4.6.4.1 A, B, C, D.

Ahora bien, este tipo de motores también puede funcionar de un modo menos "ortodoxo", pero que nos va a permitir doblar el número de pasos, si bien a costa de la regularidad del par. Esto se consigue de la siguiente manera: en principio, al igual que en el anterior fondo de funcionamiento, por los devanados T y U se hace circular una corriente, de tal modo que el estator adquiere la magnetización correspondiente y por lo tanto el rotor se orienta según ella. Ahora bien, al contrario que en el caso anterior, antes de conmutar el interruptor I.1 a su segunda posición, se desconectará el devanado T, reorientándose por consiguiente el rotor, pero la mitad de un paso (45º).

4.6.4.2 Motores paso a paso unipolares

Los motores paso a paso unipolares, en cuanto a construcción son muy similares a los anteriormente descritos excepto en el devanado de su estator (figura 4.6.4.2.1) En efecto, cada bobina del estator se encuentra dividida en dos mediante una derivación central conectada a un terminal de alimentación. De este modo, el sentido de la corriente que circula a través de la bobina y por consiguiente la polaridad magnética del estator viene determinada por el terminal al que se conecta la otra línea de la alimentación, a través de un dispositivo de conmutación. Por consiguiente las medias bobinas de conmutación hacen que se inviertan los polos magnéticos del estator, en la forma apropiada. Nótese que en vez de invertir la polaridad de la corriente como se hacía en los M.P.P. bipolares se conmuta la bobina por donde circula dicha corriente.

Al igual que los M.P.P. Bipolares, es posible tener resoluciones de giro correspondientes a un semipaso. Ahora bien, dado que las características constructivas de estos motores unipolares son idénticas a las de los bipolares, se

puede deducir que los devanados tanto en uno como otro caso ocuparán el mismo espacio, y por tanto es evidente que por cada fase tendremos menos vueltas o bien el hilo de cobre será de una sección menor. En cualquiera de los dos casos se deduce la disminución de la relación de A/Vuelta. Por tanto, a igualdad de tamaño los motores bipolares ofrecen un mayor par.

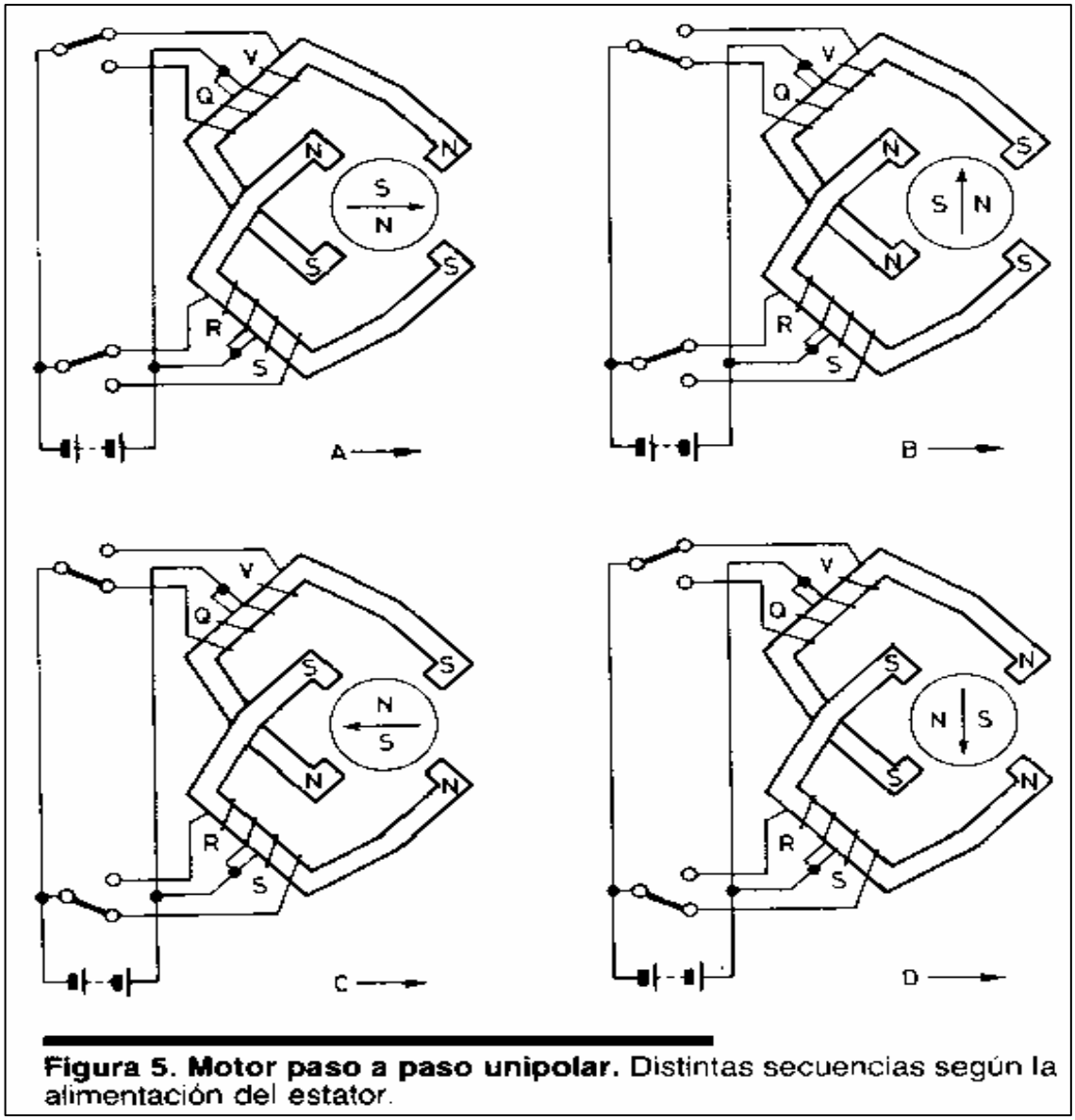


Figura 5. Motor paso a paso unipolar. Distintas secuencias según la alimentación del estator.

Figura 4.6.4.2.1 Motor Paso a Paso Unipolar

Hasta aquí se ha venido describiendo el funcionamiento de los M.P.P. con dos estatores, pero es posible aumentar el número de éstos para mejorar la resolución angular.

4.7 Sistemas de control

Un sistema de control es un conjunto de componentes, dispositivos o señales, que interconectados entre sí realizan una función predeterminada.

El objeto de un sistema de control es gobernar una variable a la salida de determinado proceso de acuerdo a una señal de entrada que le indica a los elementos de un sistema de qué manera deben actuar.

El significado de controlar la velocidad o posición angular de un motor es por un sistema de control de lazo abierto como es mostrado en la figura 4.7.1.1.

En el sistema de lazo abierto, la salida seguirá a la función deseada mientras que todas las variables son constantes. Cualquier cambio en la carga, ganancia del amplificador o cualquier otra variable del sistema causarán, sin embargo, una desviación desde el valor deseado. En categorías para el motor a ajustar a una función deseada independiente de los cambios de las variables, un sistema servo de lazo cerrado, tal como el ilustrado en la figura 4.7.2.1 debe ser construido.

No todos los sistemas pueden controlar satisfactoriamente la variable de salida y esto dependerá de la constitución y configuración de dichos sistemas, por esta razón se han subdividido en 2 categorías:

- Sistemas de control de lazo abierto
- Sistemas de control de lazo cerrado.

4.7.1 Sistemas de control de lazo abierto.

Los sistemas de control de lazo abierto, son aquellos en los que la salida no tiene efecto sobre la acción de control, es por eso que a cada entrada de referencia le corresponde una condición de opción fija ver figura 4.7.1.1, otra definición también empleada es la siguiente: La salida no influye en la entrada, es decir, la salida no tiene efecto sobre la acción de control ver figura 4.7.1.2.

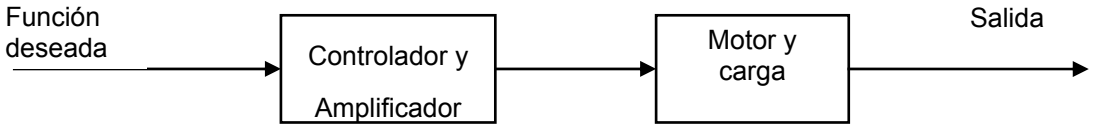


Figura 4.7.1.1 Sistema de Control de Lazo Abierto

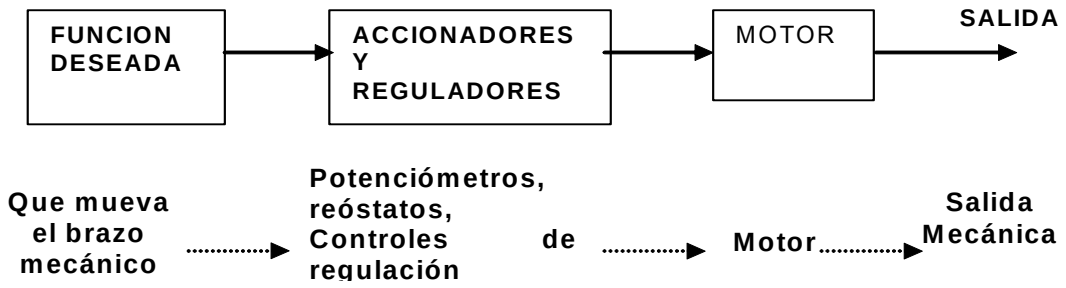


Figura 4.7.1.2 Sistema de Control de Lazo Abierto

Las características principales de un sistema de control de lazo abierto son:

- Sencillos y económicos
- Su exactitud depende del criterio y calibración
- Son sensibles a las variaciones
- Ganancia elevada.

En la práctica solo se puede usar un sistema de control de lazo abierto si la relación entre la entrada y la salida es conocida y si no hay perturbaciones internas ni externas. Debe hacerse notar que cualquier sistema de control que funciona sobre una base de tiempos es de lazo abierto.

4.7.2 Sistemas de control de lazo cerrado.

Las variables de salida son medidas, retroalimentadas y comparadas a la función de entrada deseada. Cualquier diferencia entre las dos es una desviación desde el resultado deseado; la desviación es amplificada y usada para corregir el error. De esta manera, el sistema de lazo cerrado es esencialmente insensible a variaciones en los parámetros y por lo tanto ejecuta correctamente a pesar de cambios en las condiciones de carga y otros parámetros del sistema. Sin embargo, ahora la respuesta del sistema depende en la configuración del lazo cerrado y como tal el sistema puede ser sobreamortiguado, subamortiguado o igual inestable.

Especial cuidado debe ser dado al diseño de lazo cerrado en el orden para obtener la respuesta deseada.

Es aquel en el que la señal de salida tiene efecto directo sobre la acción de control, por eso también son llamados sistemas realimentados. Ver figura 4.7.2.1.

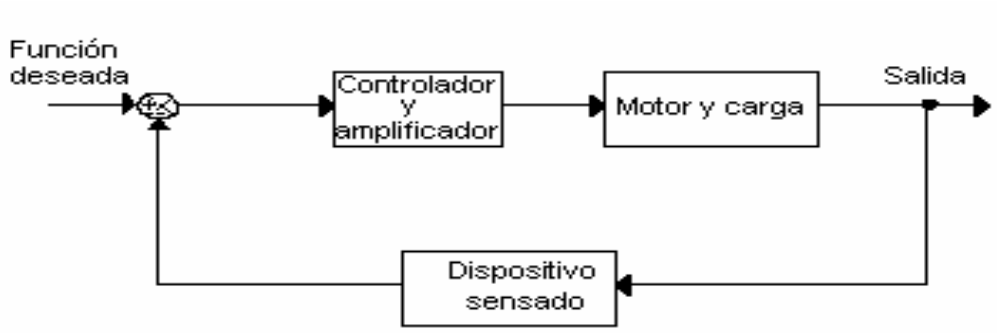


Figura 4.7.2.1 Diagrama general de un sistema de Control de Lazo Cerrado.

Los sistemas de control de lazo cerrado son sistemas realimentados. La señal de error actuante, que es la diferencia entre la señal de entrada y la señal de retroalimentación (que puede ser la señal de salida o una función de ella y sus derivadas), entra al detector o control de manera de reducir el error y llevar la salida del sistema al valor deseado. En otras palabras el termino “lazo cerrado” implica el uso de acción de realimentación para reducir el error del sistema.

Otra definición importante que podemos agregar sobre sistemas de control de lazo cerrado y aplicados al control de motores es la siguiente:

En este tipo de sistemas la salida tiene un efecto directo sobre la señal de control, ver figura 4.7.2.2.

En este tipo de sistemas la salida tiene un efecto directo sobre la señal de control.

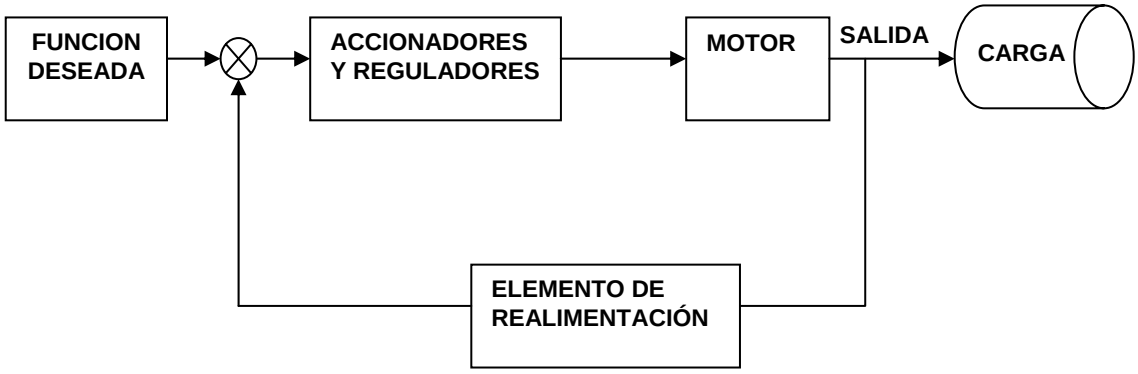


Figura 4.7.2.2 Sistema de Control de Lazo Cerrado o Realimentados.

Se recoge una muestra de la salida eléctrica, el elemento de realimentación la recoge y la lleva a un elemento todavía más complicado que compara la muestra con lo que quiere, después modifica el accionador según los resultados.

Podríamos desear una velocidad determinada para el motor sin saber su carga, por lo que deberá recoger la velocidad a la que va el motor y tomar las medidas pertinentes para luego usar los accionadores. Si por ejemplo quitáramos

la carga del motor, el motor se realimentará y se modificará a través de los accionadores. Este es el tipo de sistema que se utiliza en este proyecto.

Algunas características de estos sistemas son:

- Mayor precisión
- Relativamente sensibles a variaciones internas de los parámetros externos.
- Menor ganancia.
- Mayor flexibilidad.
- Costo elevado, debido al número de componentes necesarios.
- Pueden hacerse inestables por sobrecorrección.
- Facilitan la acción de control.

4.7.3 Sistemas servo

Son divididos por las variables a ser controladas. La mayoría de los sistemas comunes son:

- Sistemas de control de velocidad – La velocidad del motor es seguida de un perfil de velocidad dado.
- Sistemas de control de posición – La posición angular del motor por ser controlada.
- Sistemas de control de par motor – El par motor a ser controlado.
- Sistemas de control híbridos – La conmutación del sistema desde un modo de control a otro. Por ejemplo, nosotros podríamos desear controlar la velocidad por algún tiempo y entonces conmutar al modo de control de posición.

Cuando existe un operador, las maquinas generalmente utilizan sistemas en lazo abierto, en este proyecto no se utilizara un sistema de lazo abierto por requerirse la característica de realimentación.

4.8 Control de motores de CD

El control de los motores de CD se enfoca en dos aspectos: Su velocidad y el sentido de giro o rotación; en muchas aplicaciones sólo es necesario controlar la velocidad, sin embargo en otras es necesario también controlar su sentido de

rotación. Se explican primero los métodos de control de velocidad y posteriormente los de control del sentido de giro.

Mediante la ecuación que determina el voltaje inducido y la Ley de Voltajes de Kirchhoff, podemos encontrar una relación para determinar la velocidad de rotación.

Tenemos:

$$E_A = K\phi\omega$$
$$V_T = E_A + I_A R_A$$

Donde:

E_A : Voltaje inducido en la armadura
K : constante de proporcionalidad
Φ : densidad de flujo de campo
ω : velocidad angular
V_T : Voltaje terminal de armadura
I_A : Corriente de armadura
R_A : Resistencia de armadura

De las ecuaciones anteriores se puede despejar E_A de la segunda ecuación y al sustituir en la primera se obtiene:

$$K\phi\omega = V_T - I_A R_A$$

$$\omega = \frac{V_T - I_A R_A}{K\phi}$$

Esta relación dice todo acerca de la velocidad de un motor de CD. La corriente de armadura *(I_A)* es por lo general dependiente de la carga, por lo que no es comúnmente usada como variable de control.

Observando la última ecuación se pueden obtener varias relaciones:

- Al aumentar el voltaje terminal, aumenta la velocidad si el flujo por polo es constante con el motor de imán permanente.
- Para una corriente I_A dada, la velocidad disminuye si la resistencia del circuito de armadura se incrementa (a costa de una pérdida de I^2R mayor)

- La velocidad es inversamente proporcional al flujo por polo, por lo que el incrementar la corriente de campo hace disminuir la velocidad.

Teniendo en cuenta que la velocidad de un motor de corriente directa es directamente proporcional a la tensión aplicada en los bornes e inversamente proporcional al flujo, se puede controlar la velocidad (n) modificando estos dos factores en la fórmula

$$n = (V - I \cdot R) / (K \cdot \phi)$$

La mayoría de los controladores de velocidad de tipo industrial requieren un par constante para distintas velocidades. Se consigue manteniendo el flujo ϕ constante y la tensión V variable.

Por otra parte, algunas aplicaciones requieren regulación de velocidad manteniendo constante la potencia, para ello habría que mantener el par y la potencia constantes, el producto $V * I$ se debería mantener constante, el flujo ϕ sería variable. Esto se conseguiría variando la excitación del campo de un motor de tipo Shunt que es el que permite variar el flujo ϕ.

4.9 Circuitos de potencia

Dentro de los dispositivos electrónicos de potencia, podemos citar: Los diodos y transistores de potencia, el tiristor, así como otros derivados de éstos, tales como los triac, diac, conmutador unilateral o SUS, transistor uniunión o UJT, el transistor uniunión programable o PUT y el diodo Shockley.

Existen tiristores de características especiales como los fototiristores, los tiristores de doble puerta y el tiristor bloqueable por puerta (GTO).

Lo más importante a considerar de estos dispositivos, es la curva característica que nos relaciona la intensidad que los atraviesa con la caída de tensión entre los electrodos principales.

El componente básico del circuito de potencia debe cumplir los siguientes requisitos:

- Tener dos estados claramente definidos, uno de alta impedancia (bloqueo) y otro de baja impedancia (conducción).
- Poder controlar el paso de un estado a otro con facilidad y pequeña potencia.
- Ser capaces de soportar grandes intensidades y altas tensiones cuando está en estado de bloqueo, con pequeñas caídas de tensión entre sus electrodos, cuando está en estado de conducción. Ambas condiciones lo capacitan para controlar grandes potencias.

- Rapidez de funcionamiento para pasar de un estado a otro.

El último requisito se traduce en que a mayor frecuencia de funcionamiento habrá una mayor disipación de potencia. Por tanto, la potencia disipada depende de la frecuencia, ver figura 4.11.1.

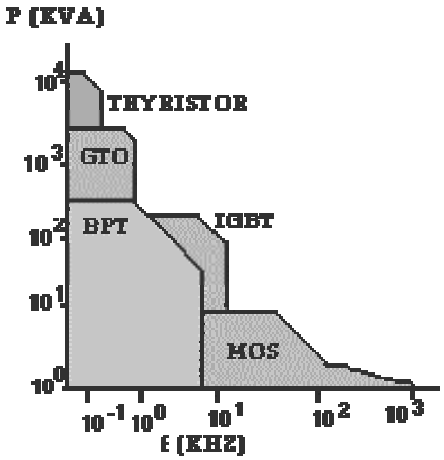


Figura 4.11.1 Gráfica de Potencia disipada vs Frecuencia en los circuitos de Potencia.

4.9.1 Clasificación de los circuitos de potencia

Los tres bloques básicos de semiconductores de potencia (de alta potencia, potencia y baja potencia) y sus aplicaciones fundamentales.

Semiconductores de alta potencia

Dispositivo	**Intensidad máxima**
Rectificadores estándar o rápidos	50 a 4800 Amperios
Transistores de potencia	5 a 400 Amperios
Tiristores estándar o rápidos	40 a 2300 Amperios
GTO	300 a 3000 Amperios

Aplicaciones:

- Tracción eléctrica: Troceadores y convertidores.
- Industria
- Control de motores asíncronos.
- Inversores.
- Caldeo inductivo.
- Rectificadores.
- Etc.

Semiconductores de potencia

Dispositivo	Intensidad máxima
Módulos de transistores	5 a 600 A. 1600 V.
SCR / módulos rectificadores	20 a 300 A. 2400 V.
Módulos GTO	100 a 200 A. 1200 V.
IGBT	50 a 300A. 1400V.

Aplicaciones:

- Soldadura al arco.
- Sistema de Alimentación Ininterrumpida (SAI).
- Control de motores.
- Tracción eléctrica.

Semiconductores de baja potencia

Dispositivo	Intensidad máxima
SCR	0.8 a 40 A. 1200 V.
Triac	0.8 a 40 A. 800 V
Mosfet	2 a 40 A. 900 V.

Aplicaciones:

- Control de motores.
- Aplicaciones domésticas.
- Cargadores de baterías.
- Control de iluminación.
- Control numérico.
- Ordenadores, etc.

Aplicaciones generales y evolución práctica

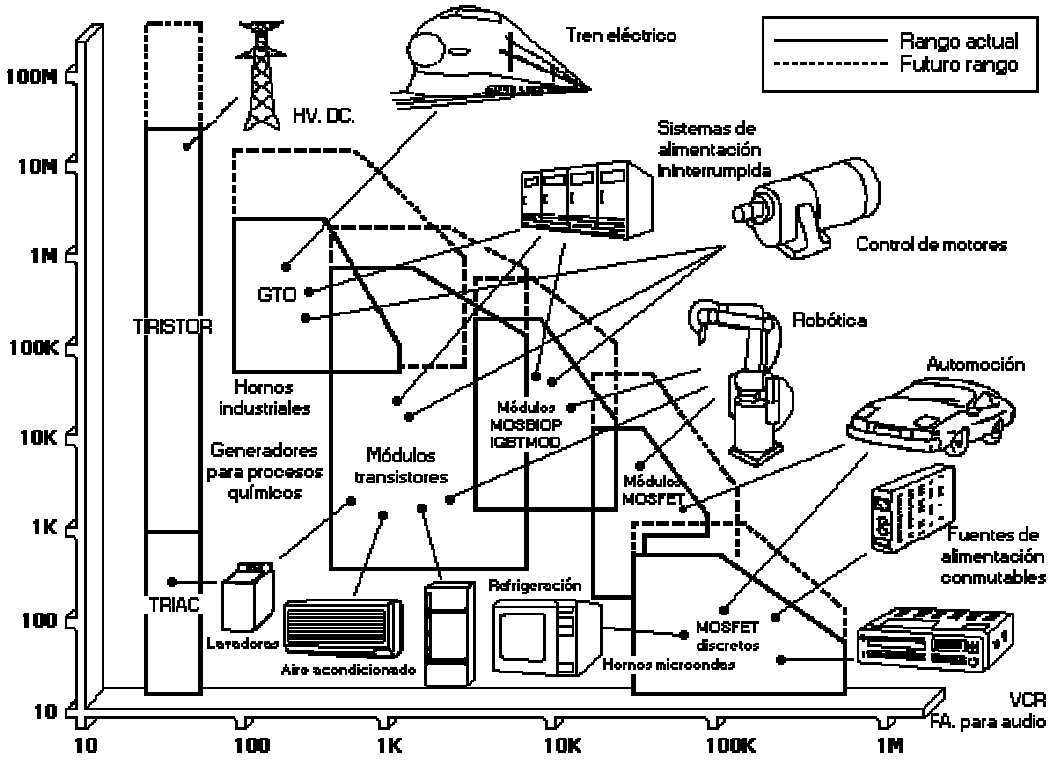


Figura 4.11.2 Aplicaciones de Semiconductores de acuerdo a la potencia manejada.

En la figura 4.11.2 se muestran las aplicaciones de cada tipo de semiconductor indicado anteriormente de acuerdo a la potencia que manejan y el rango de crecimiento para el futuro.

4.9.2 El transistor de potencia

El funcionamiento y utilización de los transistores de potencia son idénticos a los de los transistores normales, teniendo como características especiales las altas tensiones e intensidades que tienen que soportar y, por tanto, las altas potencias a disipar.

Existen tres tipos de transistores de potencia:

- Bipolar.
- Unipolar o FET (Transistor de Efecto de Campo).
- IGBT.

Parámetros	MOS	Bipolar
Impedancia de entrada	Alta (1010 ohmios)	Media (104 ohmios)
Ganancia en corriente	Alta (107)	Media (10-100)
Resistencia ON (saturación)	Media / alta	Baja
Resistencia OFF (corte)	Alta	Alta
Voltaje aplicable	Alto (1000 V)	Alto (1200 V)
Máxima temperatura de operación	Alta (200ºC)	Media (150ºC)
Frecuencia de trabajo	Alta (100-500 Khz)	Baja (10-80 Khz)
Costo	Alto	Medio

El IGBT ofrece a los usuarios las ventajas de entrada MOS, más la capacidad de carga en corriente de los transistores bipolares:

- Trabaja con tensión.
- Tiempos de conmutación bajos.
- Disipación mucho mayor (como los bipolares).

Nos interesa que el transistor se parezca, lo más posible, a un elemento ideal:

- Pequeñas fugas.
- Alta potencia.
- Bajos tiempos de respuesta (t_{on}, t_{off}), para conseguir una alta frecuencia de funcionamiento.
- Alta concentración de intensidad por unidad de superficie del semiconductor.
- Que el efecto avalancha se produzca a un valor elevado (VCE máxima elevada).
- Que no se produzcan puntos calientes (grandes di/dt).

Una limitación importante de todos los dispositivos de potencia y concretamente de los transistores bipolares, es que el paso de bloqueo a conducción y viceversa no se hace instantáneamente, sino que siempre hay un retardo (t_{on}, t_{off}) Las causas fundamentales de estos retardos son las capacidades asociadas a las uniones colector-base y base-emisor y los tiempos de difusión y recombinación de los portadores.

4.9.2.1 Características de operación

Principios básicos de funcionamiento

La diferencia entre un transistor bipolar y un transistor unipolar o FET es el modo de actuación sobre la terminal de control. En el transistor bipolar hay que inyectar una corriente de base para regular la corriente de colector, mientras que en el FET el control se hace mediante la aplicación de una tensión entre puerta y fuente. Esta diferencia viene determinada por la estructura interna de ambos dispositivos, que son substancialmente distintas.

Es una característica común, sin embargo, el hecho de que la potencia que consume la terminal de control (base o puerta) es siempre más pequeña que la potencia manejada en los otros dos terminales.

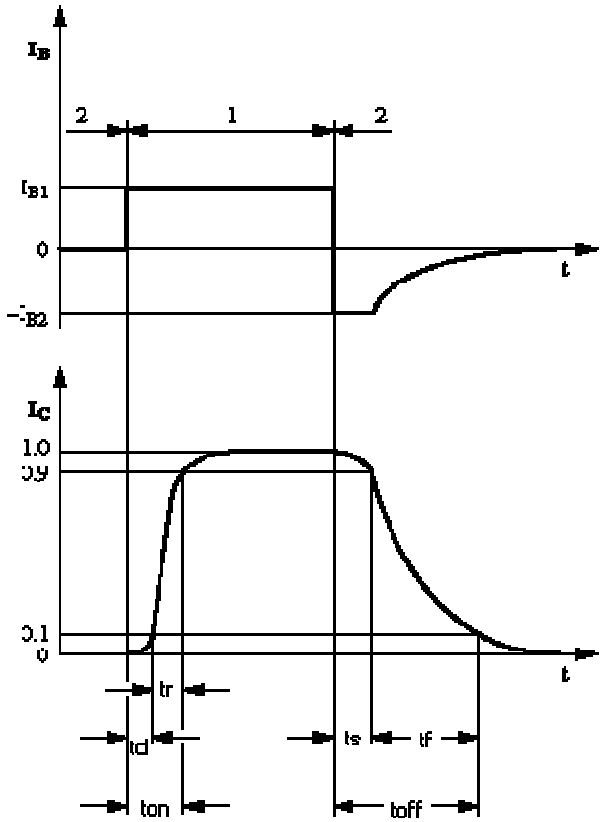


Figura 4.9.1 Gráfica característica de corriente de Colector y Base de un transistor BTJ.

En resumen, destacamos tres cosas fundamentales:

- En un transistor bipolar I_B controla la magnitud de I_C.
- En un FET, la tensión V_{GS} controla la corriente I_D.
- En ambos casos, con una potencia pequeña puede controlarse otra bastante mayor.

Tiempos de conmutación

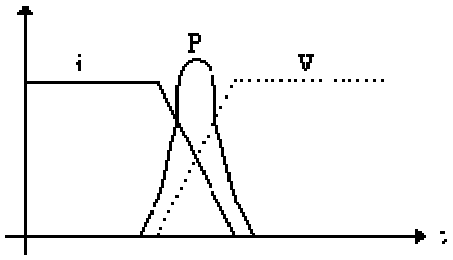


Figura 4.9.2 Gráfica que indica el tiempo o retardo de conmutación entre un estado y otro

Cuando el transistor está en saturación o en corte las pérdidas son despreciables pero si tenemos en cuenta los efectos de retardo de conmutación, al cambiar de un estado a otro se produce un pico de potencia disipada, ya que en esos instantes el producto $I_C * V_{CE}$ va a tener un valor apreciable, por lo que la potencia media de pérdidas en el transistor va a ser mayor. Estas pérdidas aumentan con la frecuencia de trabajo, debido a que al aumentar ésta, también lo hace el número de veces que se produce el paso de un estado a otro.

Podremos distinguir entre tiempo de excitación o encendido (t_{on}) y tiempo de apagado (t_{off}) A su vez, cada uno de estos tiempos se puede dividir en otros dos.

Tiempo de retardo (Delay time, t_d): Es el tiempo que transcurre desde el instante en que se aplica la señal de entrada en el dispositivo conmutador, hasta que la señal de salida alcanza el 10% de su valor final.

Tiempo de subida (Rise time, t_r): Tiempo que emplea la señal de salida en evolucionar entre el 10% y el 90% de su valor final.

Tiempo de almacenamiento (Storage time, t_s): Tiempo que transcurre desde que se quita la excitación de entrada y el instante en que la señal de salida baja al 90% de su valor final.

Tiempo de caída (Fall time, t_f): Tiempo que emplea la señal de salida en evolucionar entre el 90% y el 10% de su valor final.

Por tanto, se pueden definir las siguientes relaciones:

$$t_{on} = t_d + t_r$$
$$t_{off} = t_s + t_f$$

Es de hacer notar el hecho de que el tiempo de apagado (t_{off}) será siempre mayor que el tiempo de encendido (t_{on}).

Los tiempos de encendido (t_{on}) y apagado (t_{off}) limitan la frecuencia máxima a la cual puede conmutar el transistor:

$$F_{max} = \frac{1}{t_{on} + t_{off}}$$

Otros parámetros importantes

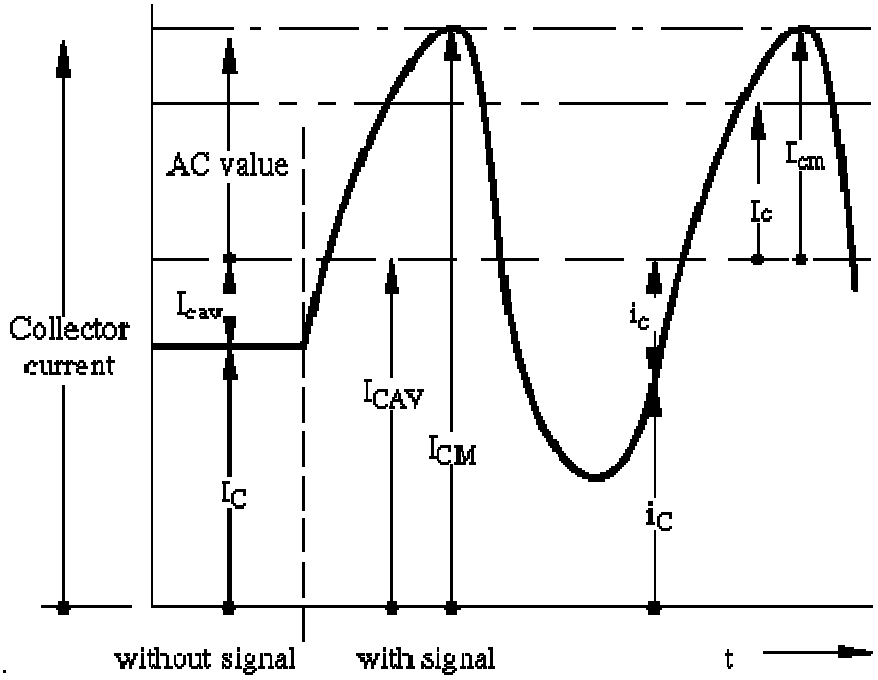


Figura 4.9.3 Análisis de la gráfica de la Corriente en el Colector de un BTJ

Corriente media: Es el valor medio de la corriente que puede circular por un terminal (ej. I_{CAV}, corriente media por el colector).

Corriente máxima: Es la máxima corriente admisible de colector (I_{CM}) o de drenador (I_{DM}). Con este valor se determina la máxima disipación de potencia del dispositivo.

V_{CBO}: Tensión entre los terminales colector y base cuando el emisor está en circuito abierto.

V_{EBO}: Tensión entre los terminales emisor y base con el colector en circuito abierto.

Tensión máxima: Es la máxima tensión aplicable entre dos terminales del dispositivo (colector y emisor con la base abierta en los bipolares, drenador y fuente en los FET).

Estado de saturación: queda determinado por una caída de tensión prácticamente constante. V_{CEsat} entre colector y emisor en el bipolar y resistencia de conducción R_{DSon} en el FET. Este valor, junto con el de corriente máxima, determina la potencia máxima de disipación en saturación.

Relación corriente de salida-control de entrada: h_{FE} para el transistor bipolar (ganancia estática de corriente) y g_{ds} para el FET (transconductancia en directa).

Modos de trabajo

Existen cuatro condiciones de polarización posibles y se indican en forma gráfica en la figura 4.9.4 Dependiendo del sentido o signo de los voltajes de polarización en cada una de las uniones del transistor.

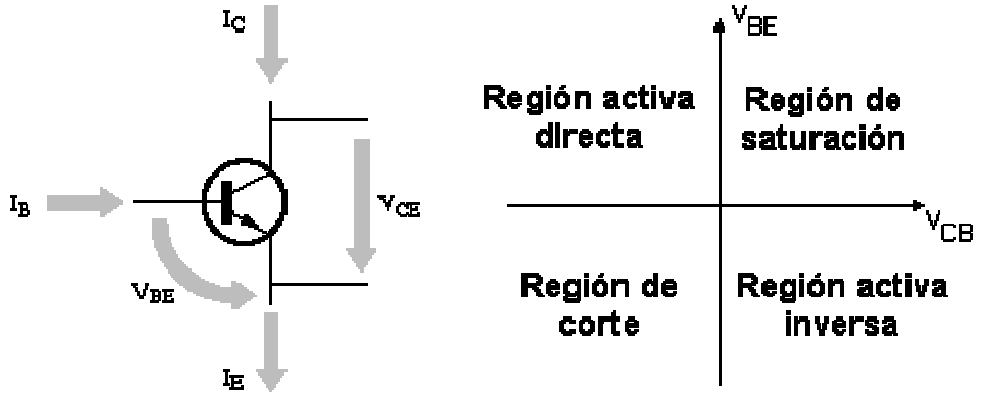


Figura 4.9.4 Representación de las regiones de BJT y dirección y sentido de la corriente que circula

- *Región activa directa*: Corresponde a una polarización directa de la unión emisor-base y a una polarización inversa de la unión colector-base. Esta es la región de operación normal del transistor para amplificación.
- *Región activa inversa*: Corresponde a una polarización inversa de la unión emisor-base y a una polarización directa de la unión colector-base. Esta región es usada raramente.
- *Región de corte*: Corresponde a una polarización inversa de ambas uniones. La operación en ésta región corresponde a aplicaciones de conmutación en el modo apagado, pues el transistor actúa como un interruptor abierto (I_{C0}).
- *Región de saturación*: Corresponde a una polarización directa de ambas uniones. La operación en esta región corresponde a aplicaciones de conmutación en el modo encendido, pues el transistor actúa como un interruptor cerrado (V_{CE0}).

Avalancha secundaria. Curvas SOA

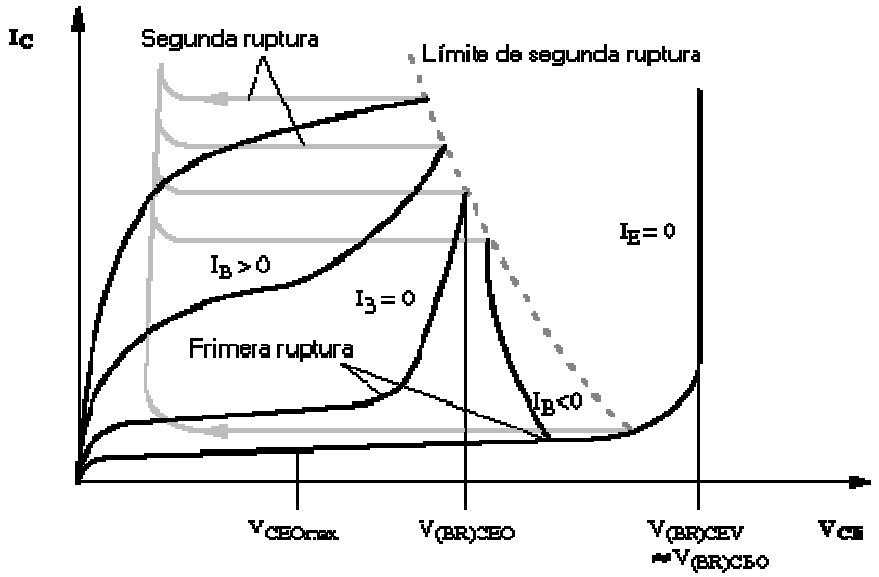


Figura 4.9.5 Curva característica SOA en un Transistor, correspondiente a la primera y segunda ruptura.

Si se sobrepasa la máxima tensión permitida entre colector y base con el emisor abierto (V_{CBO}), o la tensión máxima permitida entre colector y emisor con la base abierta (V_{CEO}), la unión colector-base polarizada en inverso entra en un proceso de ruptura similar al de cualquier diodo, denominado avalancha primaria.

Sin embargo, puede darse un caso de avalancha cuando estemos trabajando con tensiones por debajo de los límites anteriores debido a la aparición de puntos calientes (focalización de la intensidad de base), que se produce cuando tenemos polarizada la unión base-emisor en directo. En efecto, con dicha polarización se crea un campo magnético transversal en la zona de base que reduce el paso de portadores minoritarios a una pequeña zona del dispositivo (anillo circular)La densidad de potencia que se concentra en dicha zona es proporcional al grado de polarización de la base, a la corriente de colector y a la V_{CE}, y alcanzando cierto valor, se produce en los puntos calientes un fenómeno degenerativo con el consiguiente aumento de las pérdidas y de la temperatura. A este fenómeno, con efectos catastróficos en la mayor parte de los casos, se le conoce con el nombre de avalancha secundaria (o también segunda ruptura).

El efecto que produce la avalancha secundaria sobre las curvas de salida del transistor es producir unos codos bruscos que desvían la curva de la situación prevista (ver gráfica de figura 4.9.5).

El transistor puede funcionar por encima de la zona límite de la avalancha secundaria durante cortos intervalos de tiempo sin que se destruya. Para ello el fabricante suministra unas curvas límites en la zona activa con los tiempos límites de trabajo, conocidas como curvas FBSOA.

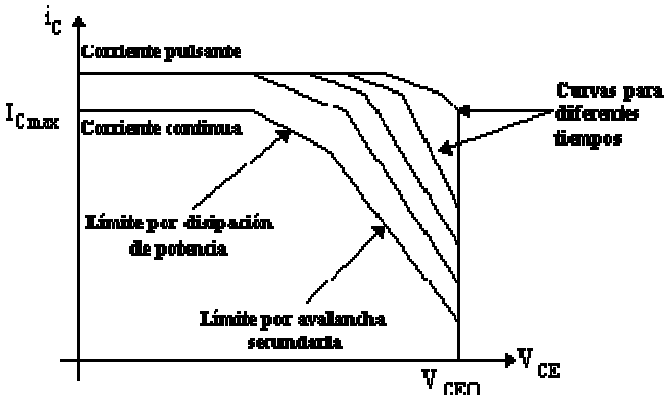


Figura 4.9.6 Gráfica del fabricante FBSOA para la segunda ruptura en un transistor.

Se puede ver como existe una curva para corriente directa y una serie de curvas para corriente pulsante, cada una de las cuales es para un ciclo concreto.

Todo lo descrito anteriormente se produce para el t_{on} del dispositivo. Durante el t_{off}, con polarización inversa de la unión base-emisor se produce la focalización de la corriente en el centro de la pastilla de silicio (Si), en un área más pequeña que en polarización directa, por lo que la avalancha puede producirse con niveles más bajos de energía. Los límites de I_C y V_{CE} durante el t_{off} vienen reflejado en las curvas RBSOA dadas por el fabricante.

Efecto producido por carga inductiva. Protecciones

Las cargas inductivas someten a los transistores a las condiciones de trabajo más desfavorables dentro de la zona activa.

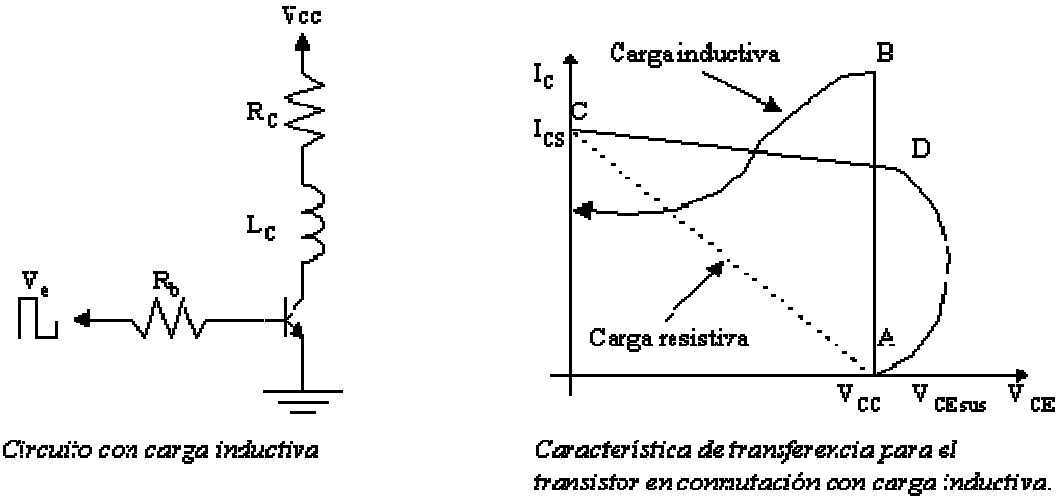


Figura 4.9.7 Gráfica de operación de un Transistor con carga Inductiva

En el diagrama de la figura 4.9.7 se han representado los diferentes puntos idealizados de funcionamiento del transistor en corte y saturación. Para una carga resistiva, el transistor pasará de corte a saturación. Para una carga resistiva, el transistor pasará de corte a saturación por la recta que va desde A hasta C, y de saturación a corte desde C a A. Sin embargo, con una carga inductiva como en el circuito anterior el transistor pasa a saturación recorriendo la curva ABC, mientras que el paso a corte lo hace por el tramo CDA. Puede verse que este último paso lo hace después de una profunda incursión en la zona activa que podría fácilmente sobrepasar el límite de avalancha secundaria, con valor V_{CE} muy superior al valor de la fuente (V_{CC}).

Para proteger al transistor y evitar su degradación se utilizan en la práctica varios circuitos, que se muestran en la figura 4.9.8, en donde se explica su funcionamiento.

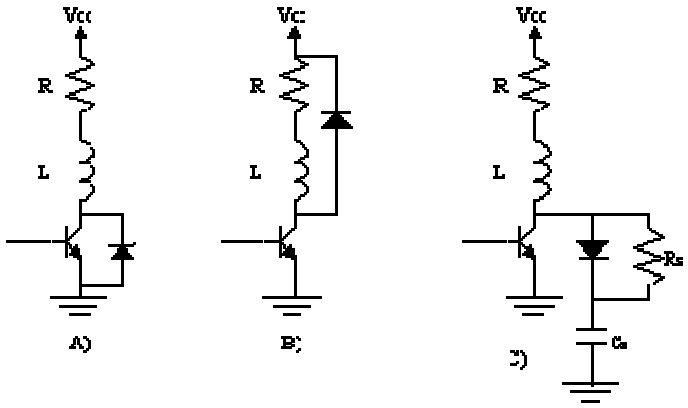


Figura 4.9.8 Diagramas de Protección para el Transistor con cargas Inductivas

A) Diodo Zéner en paralelo con el transistor (la tensión nominal zéner ha de ser superior a la tensión de la fuente V_{CC}).

b) Diodo en antiparalelo con la carga R_L.

c) Red RC polarizada en paralelo con el transistor (red snubber).

Las dos primeras limitan la tensión en el transistor durante el paso de saturación a corte, proporcionando a través de los diodos un camino para la circulación de la intensidad inductiva de la carga.

En la tercera protección, al cortarse el transistor la intensidad inductiva sigue pasando por el diodo y por el condensador C_S, el cual tiende a cargarse a una tensión V_{CC}. Diseñando adecuadamente la red RC se consigue que la tensión en el transistor durante la conmutación sea inferior a la de la fuente, alejándose su funcionamiento de los límites por disipación y por avalancha secundaria. Cuando el transistor pasa a saturación el condensador se descarga a través de R_S.

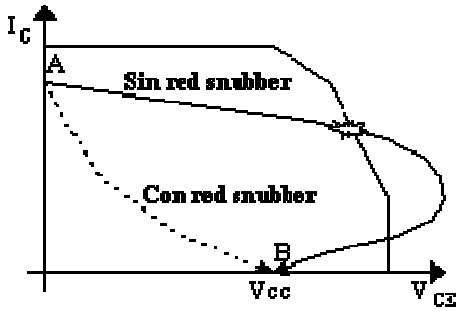


Figura 4.9.9 Diagrama de funcionamiento del transistor con red snubber para protección.

El efecto producido al incorporar la red snubber es la que se puede apreciar en la figura 4.9.9, donde vemos que con esta red, el paso de saturación (punto A) a corte (punto B) se produce de forma más directa y sin alcanzar valores de V_{CE} superiores a la fuente V_{CC}.

Para el cálculo del valor del capacitor C_S podemos suponer, despreciando las pérdidas, que la energía almacenada en la bobina L antes del bloqueo debe haberse transferido a C_S cuando la intensidad de colector se anule. Por tanto:

$$\frac{1}{2} \times L \times I_{C(sat)}^2 = \frac{1}{2} \times C_S \times V_{cc}^2$$

De donde:

$$C_S = \frac{L \times I_{C(sat)}^2}{V_{cc}^2}$$

Para calcular el valor de la resistencia R_S hay que de tener en cuenta que el condensador debe de estar descargado totalmente en el siguiente proceso de bloqueo, por lo que la constante de tiempo de R_S y C_S debe ser menor (por ejemplo una quinta parte) que el tiempo que permanece en saturación el transistor:

$$\tau_S = R_S \times C_S \leq \frac{\text{tiempo con BJT saturado}}{5}$$

Cálculo de potencias disipadas en conmutación con carga resistiva

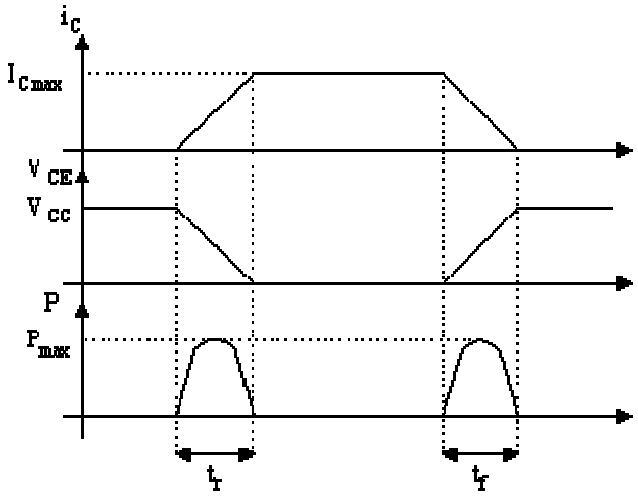


Figura 4.9.10 Gráfica de Tiempos de encendido y apagado contra Voltaje, corriente y Potencia en el colector de un Transistor

La gráfica superior muestra las señales idealizadas de los tiempos de conmutación (t_{on} y t_{off}) para el caso de una carga resistiva.

Suponiendo el momento origen en el comienzo del tiempo de subida (*tr*) de la corriente de colector. En estas condiciones (0 t tr) tendremos:

$$i_C = I_{Cmax} \times \left(\frac{t}{t_r} \right)$$

donde I_C más vale:

$$I_{Cmax} - \frac{V_{cc}}{R}$$

También tenemos que la tensión colector - emisor viene dada como:

$$V_{ce} = V_{cc} - R \times i_C$$

Sustituyendo, tendremos que:

$$V_{ce} = V_{cc} - R \times \frac{V_{cc}}{R} \times \left(\frac{t}{t_r} \right) = V_{cc} \times \left(1 - \frac{t}{t_r} \right)$$

Asumiendo que la V_{CE} en saturación es despreciable en comparación con Vcc. Así, la potencia instantánea por el transistor durante este intervalo viene dada por:

$$p = V_{ce} \times i_C = V_{cc} \times I_{Cmax} \times \left(\frac{t}{t_r} \right) \times \left(1 - \frac{t}{t_r} \right)$$

La energía, Wr, disipada en el transistor durante el tiempo de subida está dada por la integral de la potencia durante el intervalo del tiempo de caída, con el resultado:

$$W_r = \left(\frac{V_{cc} \times I_{Cmax}}{4}\right) \times \left(\frac{2 \times t_r}{3}\right)$$

De forma similar, la energía (Wf) disipada en el transistor durante el tiempo de caída, viene dado como:

$$W_f = \left(\frac{V_{cc} \times I_{Cmax}}{4}\right) \times \left(\frac{2 \times t_f}{3}\right)$$

La potencia media resultante dependerá de la frecuencia con que se efectúe la conmutación:

$$P_{AV} = f \times \left(W_r + W_f\right)$$

Un último paso es considerar t_r despreciable frente a t_f, con lo que no cometeríamos un error apreciable si finalmente se deja la potencia media, tras sustituir, como:

$$P_{C(AV)} = \frac{V_{CC} \times I_{Cmax}}{6} \times t_f \times f$$

Cálculo de potencias disipadas en conmutación con carga inductiva

En la figura 4.9.11 podemos ver la gráfica de la $i_C(t)$, $V_{CE}(t)$ y p(t) para carga inductiva. La energía perdida durante en ton viene dada por la ecuación:

$$W_{t_{on}} = \frac{1}{2} \times V \times i_{C(sat)} \times (t_1 + t_2)$$

Durante el tiempo de conducción (t_5) la energía perdida es despreciable, puesto que V_{CE} es de un valor ínfimo durante este tramo.

Durante el t_{off}, la energía de pérdidas en el transistor vendrá dada por la ecuación:

$$W_{t_{off}} = \frac{1}{2} \times V \times i_{C(sat)} \times (t_3 + t_4)$$

La potencia media de pérdidas durante la conmutación será por tanto:

$$P_{TOT(AV)} = \frac{W_{t_{on}} + W_{t_{off}}}{T} = f \times (W_{t_{on}} + W_{t_{off}})$$

Si lo que queremos es la potencia media total disipada por el transistor en todo el periodo debemos multiplicar la frecuencia con la sumatoria de pérdidas a lo largo del periodo (conmutación + conducción) La energía de pérdidas en conducción viene como:

$$W_{cond} = V_{C(sat)} \times I_{C(sat)} \times t_2$$

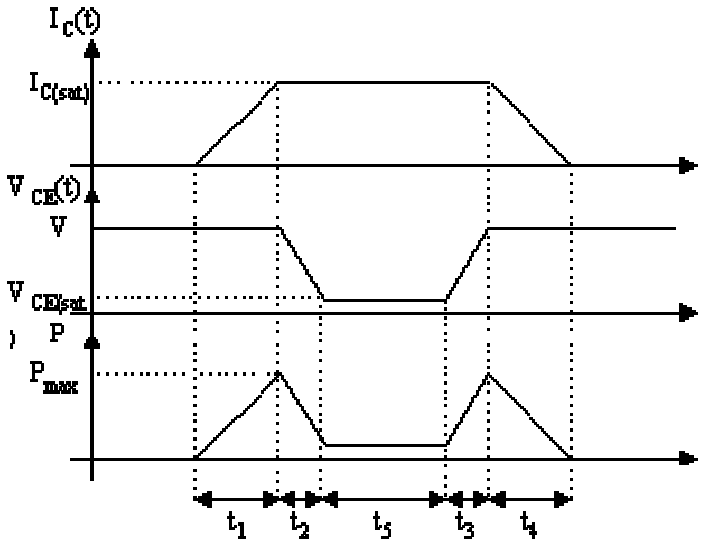


Figura 4.9.11 Gráfica de Tiempos de encendido y apagado contra Voltaje, corriente y Potencia en el colector de un Transistor con carga Inductiva

Ataque y protección del transistor de potencia

Como hemos visto anteriormente, los tiempos de conmutación limitan el funcionamiento del transistor, por lo que nos interesaría reducir su efecto en la medida de lo posible.

Los tiempos de conmutación pueden ser reducidos mediante una modificación en la señal de base, tal y como se muestra en la figura 4.9.12.

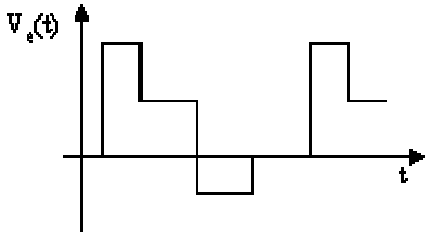


Figura 4.9.12 Gráfica de encendido para un transistor de acuerdo a los tiempos de conmutación

Puede verse como el semiciclo positivo está formado por un tramo de mayor amplitud que ayude al transistor a pasar a saturación (y por tanto reduce el t_{on}) y uno de amplitud suficiente para mantener saturado el transistor (de este modo la potencia disipada no será excesiva y el tiempo de almacenamiento no aumentará) El otro semiciclo comienza con un valor negativo que disminuye el t_{off}, y una vez que el transistor está en corte, se hace cero para evitar pérdidas de potencia.

En consecuencia, si queremos que un transistor que actúa en conmutación lo haga lo más rápidamente posible y con menores pérdidas, lo ideal sería atacar la base del dispositivo con una señal como el de la figura anterior. Para esto se puede emplear el circuito como el mostrado en la figura 4.9.13.

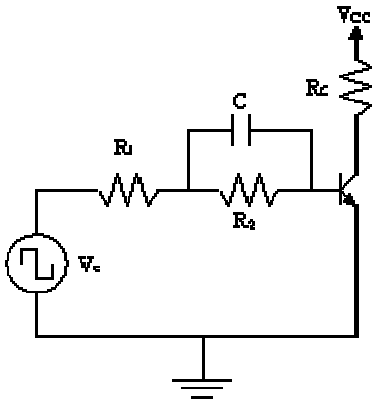


Figura 4.9.13 Circuito ideal para alimentar el encendido del transistor para conmutación rápida

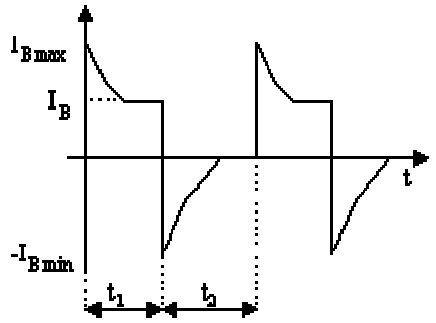


Figura 4.9.14 Gráfica resultante por conectar circuito ideal de la figura 4.9.13

En estas condiciones, la intensidad de base aplicada tendrá la forma indicada en la figura 4.9.14.

Durante el semiperiodo t_1, la tensión de entrada (Ve) se mantiene a un valor $V_{e\ (máx)}$ En estas condiciones la V_{BE} es de unos 0.7 V y el condensador C se carga a una tensión V_C de valor:

$$V_C = R_2 \times \frac{V_{e(max)} - 0.7}{R_1 + R_2}$$

Debido a que las resistencias R_1 y R_2 actúan como un divisor de tensión.

La constante de tiempo con que se cargará el condensador será aproximadamente de:

$$\tau_1 = C \times \frac{R_1 \times R_2}{R_1 + R_2}$$

Con el condensador ya cargado a V_C, la intensidad de base se estabiliza a un valor I_B que vale:

$$I_B = \frac{V_{e(max)} - 0.7}{R_1 + R_2}$$

En el instante en que la tensión de entrada pasa a valer $-V_{e(min)}$, tenemos el condensador cargado a V_C, y la V_{BE}=0.7 V. Ambos valores se suman a la tensión de entrada, lo que produce el pico negativo de intensidad $I_{B(mín)}$:

$$I_{B(min)} = \frac{V_{e(min)} + V_C + 0.7}{R_1 + R_2}$$

A partir de ese instante el condensador se descarga a través de R_2 con una constante de tiempo de valor R_2C.

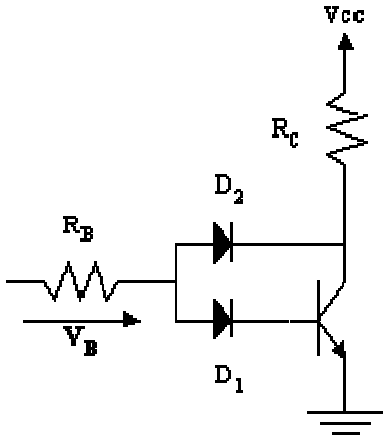
Para que todo lo anterior sea realmente efectivo, debe cumplirse que:

$$5 \times \tau_1 \leq t_1$$
$$5 \times \tau_2 \leq t_2$$

Con esto nos aseguramos que el condensador está cargado cuando apliquemos la señal negativa. Así, obtendremos finalmente una frecuencia máxima de funcionamiento:

$$f_{max} = \frac{1}{t_1 + t_2} = \frac{1}{5 \times \tau_1 + 5 \times \tau_2} = \frac{0.2}{t_1 + t_2}$$

Un circuito más serio es el de control antisaturación:



El tiempo de saturación (t_S)será proporcional a la intensidad de base, y mediante una suave saturación lograremos reducir t_S :

$$I_C = \frac{V_{CC} - V_{CE}}{R_C}$$

Inicialmente tenemos que:

$$I_B = \frac{V_B - V_{D1} - V_{BE}}{R_B}$$

En estas condiciones conduce D_2, con lo que la intensidad de colector pasa a tener un valor:

$$I_L = \frac{V_{CC} - V_{BE} - V_{d1} + V_{d2}}{R_C}$$

Si imponemos como condición que la tensión de codo del diodo D_1 sea mayor que la del diodo D_2, obtendremos que I_C sea mayor que I_L:

$$I_C = \beta \times I_B$$
$$\beta \times I_B \times R_C > V_{CE} - V_{BE} - V_{d1} + V_{d2}$$

En lo que respecta a la protección por red snubber, ya se ha visto anteriormente.

4.10 Tiristores

El tiristor es uno de los tipos más importantes de los dispositivos semiconductores de potencia. Los tiristores se utilizan en forma extensa en los circuitos electrónicos de potencia. Se operan como conmutadores biestables, pasando de un estado no conductor a un estado conductor. Para muchas aplicaciones se puede suponer que los tiristores son interruptores o conmutadores ideales, aunque los tiristores prácticos exhiben ciertas características y limitaciones.

4.10.1 Características

Un tiristor es un dispositivo semiconductor de cuatro capas de estructura pnpn con tres uniones pn tiene tres terminales: ánodo cátodo y compuerta. La figura 4.13.1.1 muestra el símbolo del tiristor y una sección recta de tres uniones pn. Los tiristores se fabrican por difusión.

Cuando el voltaje del ánodo se hace positivo con respecto al cátodo, las uniones J1 y J3 tienen polarización directa o positiva. La unión J2 tiene polarización inversa, y solo fluirá una pequeña corriente de fuga del ánodo al cátodo. Se dice entonces que el tiristor está en condición de bloqueo directo o en estado desactivado llamándose a la corriente fuga corriente de estado inactivo I_D. Si el voltaje ánodo a cátodo (V_{AK}) se incrementa a un valor lo suficientemente grande la unión J2 polarizada inversamente entrará en ruptura. Esto se conoce como ruptura por avalancha y el voltaje correspondiente se llama voltaje de ruptura directa (V_{BO}) Dado que las uniones J1 y J3 ya tienen polarización directa, habrá un movimiento libre de portadores a través de las tres uniones que

provocará una gran corriente directa del ánodo. Se dice entonces que el dispositivo está en estado de conducción o activado.

Figura 4.13.1.1 Símbolo del tiristor y tres uniones pn

La caída de voltaje se deberá a la caída óhmica de las cuatro capas y será pequeña, por lo común 1 V. En el estado activo, la corriente del ánodo está limitada por una impedancia o una resistencia externa, R_L, tal y como se muestra en la figura 4.13.12. La corriente del ánodo debe ser mayor que un valor conocido como corriente de enganche I_L, a fin de mantener la cantidad requerida de flujo de portadores a través de la unión; de lo contrario, al reducirse el voltaje del ánodo al cátodo, el dispositivo regresará a la condición de bloqueo. La corriente de enganche, I_L, es la corriente del ánodo mínima requerida para mantener el tiristor en estado de conducción inmediatamente después de que ha sido activado y se ha retirado la señal de la compuerta. En la figura 4.13.1.2 aparece una gráfica característica *v-i* común de un tiristor.

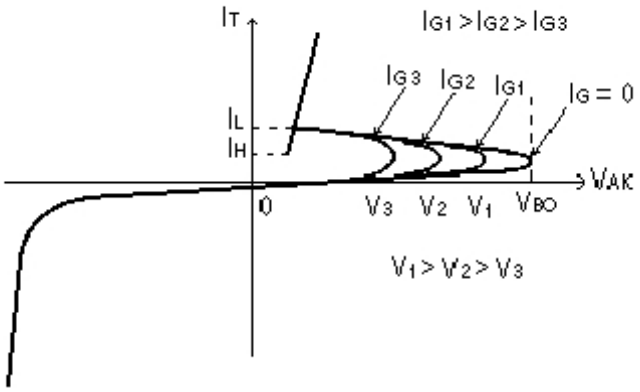


Figura 4.13.1.2 Efectos de la corriente de compuerta sobre el voltaje de bloqueo directo.

4.10.2 Tipos de tiristores

Los tiristores se fabrican casi exclusivamente por difusión. La corriente del ánodo requiere de un tiempo finito para propagarse por toda el área de la unión, desde el punto cercano a la compuerta cuando inicia la señal de la compuerta para activar el tiristor. Para controlar el *di/dt*, (la corriente en función del tiempo) el tiempo de activación y el tiempo de desactivación, los fabricantes utilizan varias estructuras de compuerta.

Dependiendo de la construcción física y del comportamiento de activación y desactivación, en general los tiristores pueden clasificarse en nueve categorías:

1. Tiristores de control de fase o de conmutación rápida (SCR).
2. Tiristores de desactivación por compuerta (GTO).
3. Tiristores de triodo bidireccional (TRIAC).
4. Tiristores de conducción inversa (RTC).
5. Tiristores de inducción estática (SITH).
6. Rectificadores controlados por silicio activados por luz (LASCR).
7. Tiristores controlados por FET (FET-CTH).
8. Tiristores controlados por MOS (MCT).

4.10.3 Tiristores de control de fase o de conmutación rápida (SCR)

El miembro más importante de la familia de los tiristores es el tiristor de tres terminales, conocido también como el Rectificador Controlado de Silicio o SCR. Este dispositivo lo desarrolló la General Electric en 1958 y lo denominó SCR. El nombre de tiristor lo adoptó posteriormente la Comisión Electrotécnica Internacional (CEI) En la figura 4.10.1 se muestra el símbolo de un tiristor de tres terminales o SCR.

Tal como su nombre lo sugiere, el SCR es un rectificador controlado o diodo. Su característica voltaje-corriente, con la compuerta de entrada en circuito abierto, es la misma que la del diodo PNPN.

Lo que hace al SCR especialmente útil para el control de motores en sus aplicaciones es que el voltaje de ruptura o de encendido puede ajustarse por medio de una corriente que fluye hacia su compuerta de entrada. Cuanto mayor sea la corriente de la compuerta, tanto menor se vuelve V_{BO}. Si se escoge un SCR de tal manera que su voltaje de ruptura, sin señal de compuerta, sea mayor que el mayor voltaje en el circuito, entonces, solamente puede activarse mediante la aplicación de una corriente a la compuerta. Una vez activado, el dispositivo permanece así hasta que su corriente caiga por debajo de I_H. Además, una vez que se dispare el SCR, su corriente de compuerta puede retirarse, sin que afecte

su estado activo. En este estado, la caída de voltaje directo a través del SCR es cerca de 1.2 a 1.5 veces mayor que la caída de voltaje a través de un diodo directo-oblicuo común.

Figura 4.10.1 Tiristor de tres terminales o SCR.

Los tiristores de tres terminales o SCR son, sin lugar a dudas, los dispositivos de uso más común en los circuitos de control de potencia. Se utilizan ampliamente para cambiar o rectificar aplicaciones y actualmente se encuentran en clasificaciones que van desde unos pocos A hasta un máximo de 3,000 A.

Un SCR:

- Se activa cuando el voltaje V_D que lo alimenta excede V_{BO}
- Tiene un voltaje de ruptura V_{BO}, cuyo nivel se controla por la cantidad de corriente I_G, presente en el SCR
- Se desactiva cuando la corriente I_D que fluye por él cae por debajo de I_H
- Detiene todo flujo de corriente en dirección inversa, hasta que se supere el voltaje máximo inverso.

4.10.4 Tiristores de Apagado por Compuerta (GTO)

Entre las mejoras más recientes que se le han hecho al tiristor está el apagado por compuerta (GTO) Un tiristor GTO es un SCR que puede apagarse por una pulsación suficientemente grande en su compuerta de entrada, aun si la corriente I_D excede I_H. Aunque los tiristores GTO se han venido usando desde 1960, solamente se volvieron prácticos para las aplicaciones de control de motores, al final de los años setenta. Estos dispositivos se han vuelto más y más comunes en las unidades de control de motores, puesto que ellos eliminaron la necesidad de componentes externos para apagar los SCR en circuitos de CD, ver figura 4.10.2

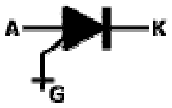


Figura 4.10.2 Tiristor apagado por compuerta (GTO).

La típica forma de onda de la corriente de compuerta de un tiristor GTO de alta potencia se muestra a continuación. Un tiristor GTO requiere una mayor corriente de compuerta para encendido que un SCR común. Para grandes aparatos de alta potencia se necesitan corrientes de compuerta del orden de 10 A o más. Para apagarlos se necesita una gran pulsación de corriente negativa de entre 20 y 30 ms de duración. La magnitud de la pulsación de corriente negativa debe ser de un cuarto a un sexto de la corriente que pasa por el aparato, ver figura 4.10.3

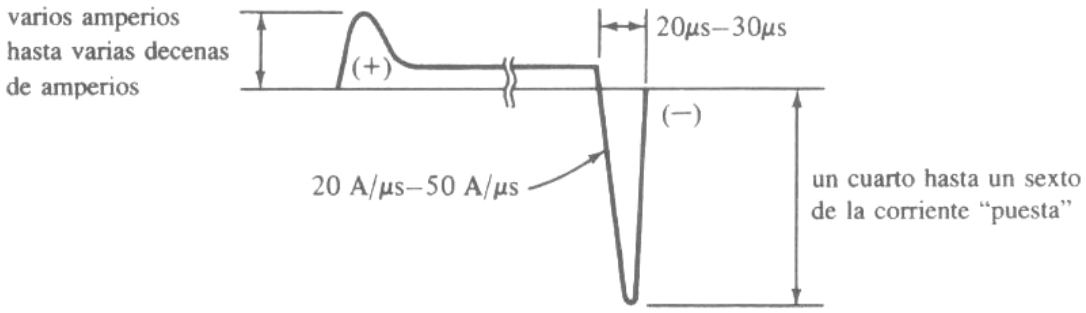
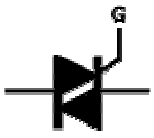


Figura 4.10.3 Forma de onda de corriente de compuerta de un tiristor (GTO).

4.10.5 Tiristores de triodo bidireccional (TRIAC)

Es un dispositivo que se comporta como dos SCR conectados en contraposición, con una compuerta de paso común; puede ir en cualquier dirección desde el momento en que el voltaje de ruptura se sobrepasa. El símbolo del TRIAC y su característica Corriente - Voltaje se ilustra en la figura 4.10.4. El voltaje de ruptura en un TRIAC disminuye si se aumenta la corriente de compuerta, en la misma forma que lo hace en un SCR, con la diferencia que un TRIAC responde tanto a los impulsos positivos como a los negativos de su compuerta. Una vez encendido, un TRIAC permanece así hasta que su corriente cae por debajo de I_H.



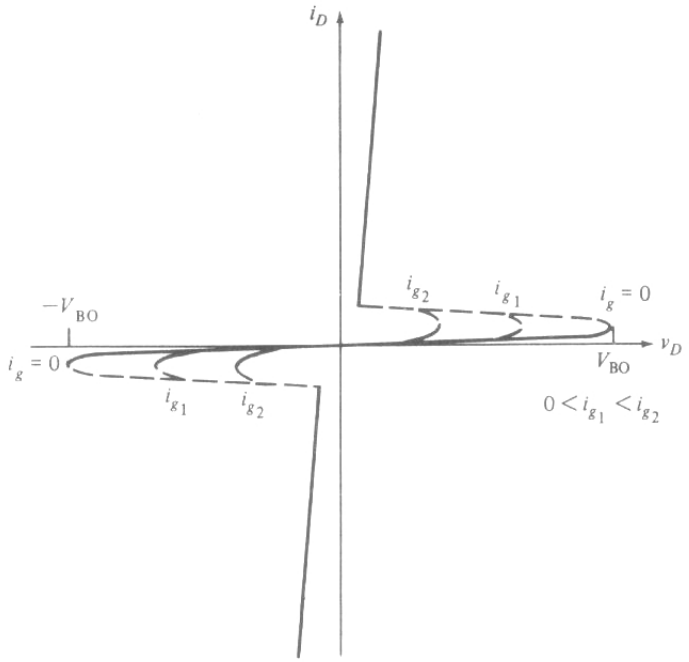


Figura 4.10.4 Símbolo del TRIAC y su característica Corriente - Voltaje.

4.10.6 Tiristores de Conducción Inversa (RTC)

En muchos circuitos pulsadores e inversores, se conecta un diodo antiparalelo a través de un SCR, con la finalidad de permitir un flujo de corriente inversa debido a una carga inductiva, y para mejorar el requisito de desactivación de un circuito de conmutación. El diodo fija el voltaje de bloqueo inverso del SCR a 1 ó 2 V por debajo de las condiciones de régimen permanente. Sin embargo, bajo condiciones transitorias, el voltaje inverso puede elevarse hasta 30 V debido al voltaje inducido en la inductancia dispersa del circuito dentro del dispositivo.

Un RTC es un intercambio entre características del dispositivo y requisitos del circuito; puede considerarse como un tiristor con un diodo antiparalelo incorporado, tal y como se muestra en la figura 4.10.5 Un RTC se conoce también como tiristor asimétrico (ASCR) El voltaje de bloqueo directo varía de 400 a 2000 V y la especificación de corriente llega hasta 500 A. El voltaje de bloqueo inverso es típicamente 30 a 40 V. Dado que para un dispositivo determinado está preestablecida la relación entre la corriente directa a través de un tiristor y la corriente inversa del diodo, sus aplicaciones se limitarán a diseños de circuitos específicos.

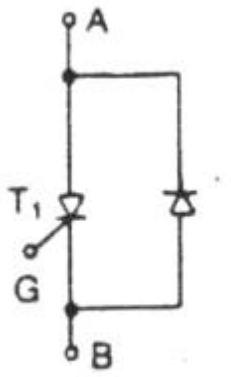


Figura 4.10.5 Tiristor de conducción inversa

4.10.7 Tiristores de Inducción Estática (SITH)

Por lo general, un SITH es activado al aplicársele un voltaje positivo de compuerta, como los tiristores normales, y desactivado al aplicársele un voltaje negativo a su compuerta. Un SITH es un dispositivo de portadores minoritarios. Como consecuencia, el SITH tiene una baja resistencia en estado activo así como una baja caída de potencial, y se puede fabricar con especificaciones de voltaje y corriente más altas.

Un SITH tiene velocidades de conmutación muy rápidas y capacidades altas de *dv/dt* y *di/dt*. El tiempo de conmutación es del orden de 1 a 6 ms. La especificación de voltaje puede alcanzar hasta 2500 V y la de corriente está limitada a 500 A. Este dispositivo es extremadamente sensible a su proceso de fabricación, por lo que pequeñas variaciones en el proceso de manufactura pueden producir cambios de importancia en sus características.

4.10.8 Rectificadores Controlados de Silicio Activados por Luz (LASCR)

Este dispositivo se activa mediante radiación directa sobre el disco de silicio provocado con luz. Los pares electrón-hueco que se crean debido a la radiación producen la corriente de disparo bajo la influencia de un campo eléctrico. La estructura de compuerta se diseña a fin de proporcionar la suficiente sensibilidad para el disparo, a partir de fuentes luminosas prácticas (por ejemplo, LED y para cumplir con altas capacidades de *di/dt* y *dv/dt*).

Los LASRC se utilizan en aplicaciones de alto voltaje y corriente [por ejemplo, transmisión de CD de alto voltaje (HVDC) y compensación de potencia reactiva estática o de Volt-Amperes reactivos (VAR)] Un LASCR ofrece total

aislamiento eléctrico entre la fuente de disparo luminoso y el dispositivo de conmutación de un convertidor de potencia, que flota a un potencial tan alto como unos cuantos cientos de kilovoltios. La especificación de voltaje de un LASCR puede llegar tan alto como 4 KV a 1500 A, con una potencia de disparo luminoso de menos de 100 mW. El *di/dt* típico es 250 A/ms y el *dv/dt* puede ser tan alto como 2000 V/ms.

4.10.9 Tiristores Controlados por FET (FET-CTH)

Un dispositivo FET-CTH combina un MOSFET y un tiristor en paralelo, tal y como se muestra en la figura 4.10.6. Si a la compuerta del MOSFET se le aplica un voltaje suficiente, típicamente 3 V, se genera internamente una corriente de disparo para el tiristor. Tiene una alta velocidad de conmutación, un *di/dt* alto y un *dv/dt* alto.

Este dispositivo se puede activar como los tiristores convencionales, pero no se puede desactivar mediante control de compuerta. Esto serviría en aplicaciones en las que un disparo óptico debe utilizarse con el fin de proporcionar un aislamiento eléctrico entre la señal de entrada o de control y el dispositivo de conmutación del convertidor de potencia.

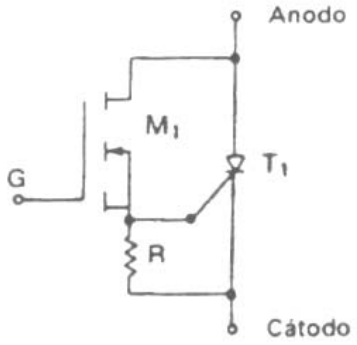


Figura 4.10.6 Tiristor FET-CTH

4.10.10 Tiristores Controlados por MOS (MCT)

Un tiristor controlado por MOS (MCT) combina las características de un tiristor regenerativo de cuatro capas y una estructura de compuerta MOS. El circuito equivalente se muestra en la figura 4.10.7 (b) y el símbolo correspondiente en la (a) La estructura NPNP se puede representar por un transistor NPN Q_1 y con un transistor Q_2. La estructura de compuerta MOS se puede representar por un MOSFET de canal p M_1 y un MOSFET de canal n M_2.

Debido a que se trata de una estructura NPNP, en vez de la estructura PNPN de un SCR normal, el ánodo sirve como la terminal de referencia con respecto a la cual se aplican todas las señales de compuerta. Supongamos que el MCT está en estado de bloqueo directo y se aplica un voltaje negativo VGA. Un

canal, p (o una capa de inversión) se forma en el material dopado n, haciendo que los huecos fluyan lateralmente del emisor p E_2 de Q_2 (fuente S_1 del MOSFET M_1 del canal p) a través del canal p hacia la base p B_1 de Q_1 (que es drenaje D_1 del MOSFET M_1, del canal p). Este flujo de huecos forma la corriente de base correspondiente al transistor npn Q_1. A continuación el emisor n+ E_1 de Q_1, inyecta electrones, que son recogidos en la base n B_2 (y en el colector n C_1) que hace que el emisor p E_2 inyecte huecos en la base n B_2, de tal forma que se active el transistor PNP Q_2 y engancha al MCT. En breve, un VGA de compuerta negativa activa al MOSFET M_1 canal p, proporcionando así la corriente de base del transistor Q_2.

Supongamos que el MCT está en estado de conducción, y se aplica un voltaje positivo VGA. Se forma entonces un canal n en el material contaminado p, haciendo que fluyan lateralmente electrones de la base n B_2 de Q_2 (fuente S_2 del MOSFET M_2 del canal n) a través del canal n del emisor n+ fuertemente contaminado de Q_l (drenaje D_2 del MOSFET M_2 del canal n+). Este flujo de electrones desvía la corriente de base del transistor PNP Q_2 de tal forma que su unión base-emisor se desactiva, y ya no habrá huecos disponibles para recolección por la base p B_1 de Q_1 (y el colector p C_2 de Q_2) La eliminación de esta corriente de huecos en la base p B_1, hace que se desactive el transistor NPN Q_1, y el MCT regresa a su estado de bloqueo. En breve, un pulso positivo de compuerta VGA, desvía la corriente que excita la base de Q_l, desactivando por lo tanto el MCT.

El MCT se puede operar como dispositivo controlado por compuerta, si su corriente es menor que la corriente controlable pico. Intentar desactivar el MCT a corrientes mayores que su corriente controlable pico de especificación, puede provocar la destrucción del dispositivo. Para valores más altos de corriente, el MCT debe ser conmutado como un SCR estándar. Los anchos de pulso de la compuerta no son críticos para dispositivos de corrientes pequeñas. Para corrientes mayores, el ancho del pulso de desactivación debe ser mayor. Además, durante la desactivación, la compuerta utiliza una corriente pico. En muchas aplicaciones, incluyendo inversores y pulsadores, se requiere, de un pulso continuo de compuerta sobre la totalidad del periodo de encendido/apagado a fin de evitar ambigüedad en el estado.

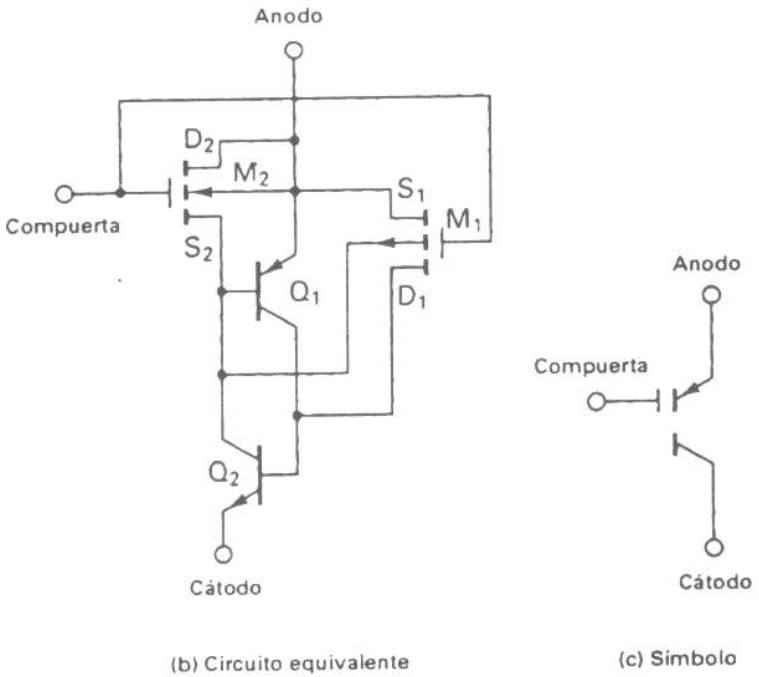


Figura 4.10.7 Tiristor controlado por MOS (MCT)

Un MCT tiene (1) una baja caída de voltaje directo durante la conducción; (2) un tiempo de activado rápido, típicamente 0.4 ms, y un tiempo de desactivado rápido, típicamente 1.25 ms, para un MCT de 300 A, 500 V; (3) bajas perdidas de conmutación; (4) una baja capacidad de bloqueo voltaje inverso y (5) una alta impedancia de entrada de compuerta, lo que simplifica mucho los circuitos de excitación. Es posible ponerlo efectivamente en paralelo, para interrumpir corrientes altas, con sólo modestas reducciones en la especificación de corriente del dispositivo. No se puede excitar fácilmente a partir de un transformador de pulso, si se requiere de una polarización continua a fin de evitar ambigüedad de estado.

4.10.11 Comparación de potencia y velocidad de los componentes electrónicos

La figura 4.10.8 enseña una comparación de las velocidades relativas y las capacidades de manejo de potencia de los tiristores SCR y GTO y los transistores de potencia. Los SCR sirven, sin duda, para operar con mayor potencia que cualquiera de los otros dispositivos. Los tiristores GTO pueden operar a una potencia casi tan alta que la de los SCR, pero mucho más rápido que éstos. Finalmente, los transistores de potencia pueden manejar menos potencia que cualquier tipo de tiristor, pero pueden accionar diez veces más rápido que éstos o aun más.

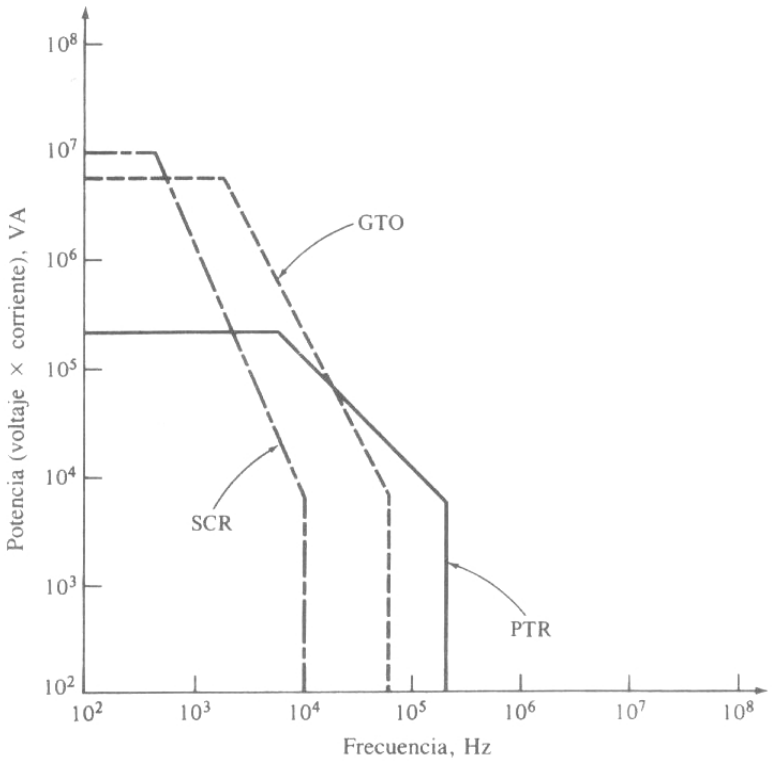


Figura 4.10.8 Comparación de las velocidades relativas y las capacidades de manejo de Potencia de los tiristores SCR y GTO y los transistores de Potencia

4.11 Amplificación de señales

La necesidad de amplificación se presenta en virtud de las señales consideradas débiles o pequeñas que poseen poca amplitud (μV o mV) y poca energía. Estas señales son demasiado reducidas para tener un procesamiento confiable, y este proceso es mucho más fácil si la magnitud de la señal se hace mayor. El bloque funcional que lleva a cabo esta tarea es el amplificador de señal.

Cuando una señal se amplifica, se debe tener cuidado para que la información contenida en la señal no cambie ni se introduzca nueva información. Es decir las variaciones en la forma de la onda de salida deben ser idénticas a las de la onda de entrada. Cualquier cambio en la forma de onda se considera distorsión y obviamente es indeseable.

Los amplificadores que hacen más grande la amplitud de la señal, se consideran como amplificadores de voltaje o de tensión. Sin embargo existe el amplificador de potencia. Este amplificador proporciona poca o ninguna ganancia de voltaje, pero si una sustancial ganancia de corriente. Así mientras se absorbe poca potencia de la fuente de señal de entrada a la cuál este se conecta, el amplificador entrega grandes cantidades de potencia a su carga.

Los amplificadores usados en sistemas de control de velocidad caen dentro de dos amplias categorías; amplificadores lineales (también amplificadores llamados Clase A) y amplificadores de conmutación. Los amplificadores lineales son casi exclusivamente dispositivos a transistor, mientras que los amplificadores de conmutación pueden ser diseñados usando transistores o SCR's (Rectificadores Controlados de Silicio, también conocidos como Tiristores).

4.11.1 Amplificadores lineales

El amplificador lineal es muy deseable desde un punto de vista de análisis de control, a partir de que tiene características lineales de control y no retardos significativos de control dentro del ancho de banda de operación. Esto es fácilmente compensado, y es capaz de proveer control de velocidad sobre un amplio rango. Es el menos problemático en términos de disturbios transitorios en circuitos adyacentes.

Los amplificadores servo a transistor son caracterizados por dos diseños principales de etapa de salida, las configuraciones tipo básicas "H" y "T", como se muestra en las Figuras 4.10.9 y 4.10.10.

El "H" o etapa de salida en puente consiste de cuatro transistores usando una simple fuente de poder de CD. Esta etapa de salida tiene la ventaja de una simple, unipolar fuente de poder y algunas divisiones de protección de voltaje entre los transistores. Sin embargo, no es fácil de manejar en casos lineales, y los voltajes y corrientes retroalimentados no son fáciles de lograr, puesto que el motor esta "flotando".

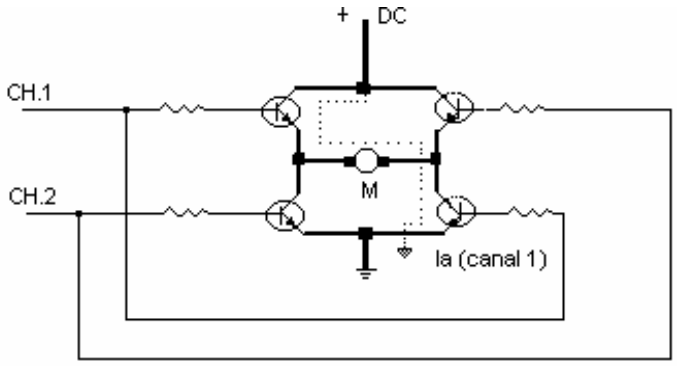


Figura 4.10.9 Etapa de salida tipo "H" (una fuente de poder es requerida)

La etapa "T" necesita dos fuentes de poder y transistores complementarios, pero es fácil de manejar y puede suministrar señales de retroalimentación en voltaje o corriente en un modo simple. Por esta razón los "T" son usados más a menudo en amplificadores lineales. La polarización de los transistores de salida requiere atención cuidadosa, puesto que una conducción simultánea de ambos transistores resultaría en corto circuito entre las dos fuentes de poder.

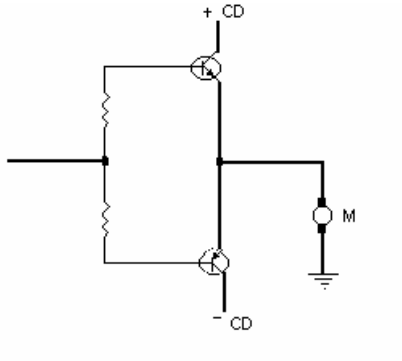


Figura 4.10.10 Etapa de salida tipo “T” (dos fuentes de poder son requeridas)

Las características *entrada–salida* de la etapa de salida generalmente tendrán algunas zonas muertas; un rasgo indeseable desde el punto de vista de retroalimentación lineal. Alrededor de la retroalimentación negativa el amplificador reducirá tal no linealidad a una cantidad insignificante.

En caso donde el amplificador lineal será usado en modo “strike strike”, la zona muerta es de consecuencias no principales, puesto que el amplificador será usado solamente en el estado de conducción o no conducción.

Las principales limitaciones en la capacidad de manejo de la potencia de salida del amplificador lineal son las características térmicas de su estado de salida. Puesto que la disipación de potencia de la etapa de salida es el producto del voltaje y la corriente a través de los transistores, el transistor y sus ensambles (disipador) asociados de calor deben ser capaces de disipar este calentamiento. En suma, se tendrá que considerar la característica secundaria de rompimiento hacia abajo del transistor, lo cual limitará la duración del tiempo del pico de corriente en niveles de voltaje dados, en el orden para asegurar que tales niveles de voltaje no son excedidos.

El más simple sistema de control de lazo cerrado es tipificado por el sencillo amplificador a transistor ver figura 4.10.11. Ha sido usado ampliamente en aplicaciones donde las cargas son esencialmente constantes y un rango limitado de velocidad fue aceptable en vista de la simplicidad y consecuentemente el bajo costo. El circuito consiste simplemente de un potenciómetro de 5 W y un transistor de poder, montado en un disipador de calor.

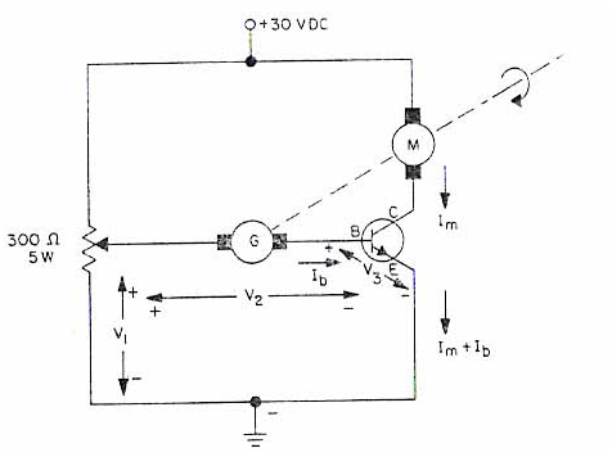


Figura 4.10.11.Un típico sistema de control de velocidad de un sencillo transistor.

La operación del circuito puede ser explicado en el modo siguiente: El potenciómetro es establecido en un punto tal que un voltaje V_1 aparece entre el contacto deslizante y la terminal negativa (tierra) del circuito. Este voltaje V_1 causará una corriente I_b para fluir a través de la base (B) y el emisor (E) del transistor. La corriente de base causa que el transistor conduzca, y la corriente I_m fluirá desde la terminal positiva a través del devanado del motor (M) al colector (C) y el emisor (E) del transistor a la terminal negativa. La corriente del motor I_m causará al motor girar y, si el tacómetro es conectado apropiadamente con respecto a su polaridad, un voltaje contrario V_2 aparecerá en las terminales del tacómetro. Ya que cualquier transistor en conducción tiene un voltaje dado base emisor asociado para cada nivel de conducción, un voltaje V_3 aparecerá a través del transistor con la polaridad mostrada en la figura anterior.

Equilibrio (control de velocidad) aparecerá cuando el voltaje V_2 alcance su valor de estado estable. El motor no puede correr más rápido en la cual lo produce V_2 ya que si lo hizo así, I_b dejaría de fluir, así haciendo al transistor no conductor, por lo tanto más lento el motor bajando a la velocidad de “equilibrio”. A la inversa, si un cambio en la carga causara que el motor tendiera a bajar lentamente, el voltaje V_2 sería pequeño, permitiendo un incremento en I_b. Esto tendería a hacer que el transistor incremente su conductividad, así permitiendo un incremento en I_m, lo cual en cambio superaría la carga incrementada. Podemos observar que este circuito tiene una cualidad auto – regulable, y por lo tanto califica como un sistema regulador de lazo cerrado.

El sencillo amplificador a transistor tiene la limitación básica de baja ganancia. En otras palabras, requiere de un error significativo (ΔI_b) para corregir una perturbación de velocidad. A condición de una mayor ganancia es una tarea simple, sin embargo, y en la figura 4.10.12, se muestra un amplificador de cuatro transistores, empleando un amplificador de voltaje a dos etapas y un amplificador de corriente emisor – seguidor doble.

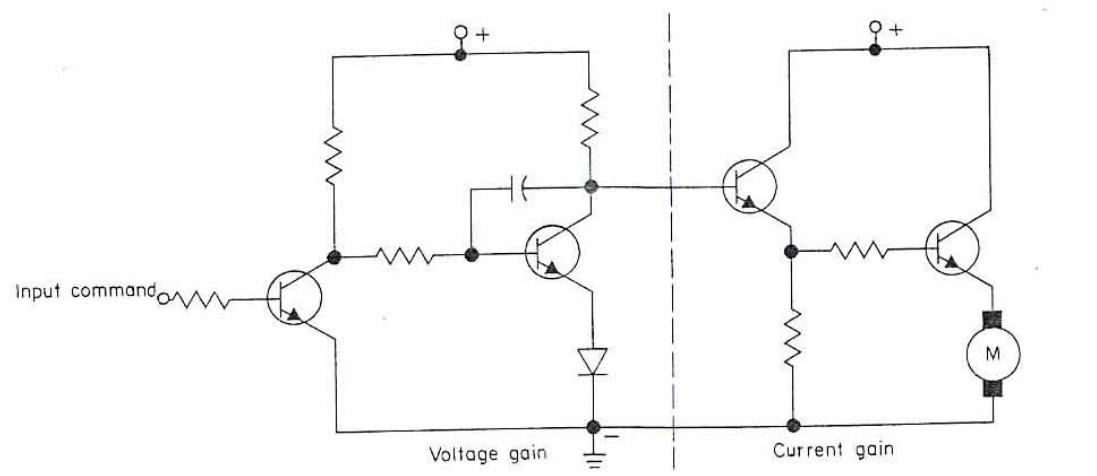


Figura 4.10.12 Amplificador elemental multi - transistor

4.11.2 Amplificadores de conmutación (Switching amplifiers)

Un amplificador conmutado es quizá el más versátil y popular amplificador servo hoy en día. Con transistores de potencia disponibles los cuales pueden conmutar en la región de MHz, es ahora fácil de diseñar amplificadores conmutados de potencia capaces de manejar índices de conmutación de ondas cuadradas de 50 Hz. Estos altos índices de conmutación son obtenibles a través del uso de técnicas de retroalimentación de corrientes positivas.

En general, ambos la frecuencia y el ciclo de rendimiento de un amplificador conmutado variará con la carga. Mientras que la variación en el ciclo de rendimiento es requerida, el cambio de frecuencia es indeseable ya que podría excitar modos de resonancia en el sistema. Otra desventaja de la variación de frecuencia es que produce un ruido audible, lo cual en la mayoría de los casos es no deseable.

En el orden para impedir variaciones de frecuencia el amplificador puede ser diseñado para tener una frecuencia de conmutación constante. Tal amplificador conmutado es llamado amplificador modulado de ancho de pulso (“Pulse-Width-Modulated” PWM), ya que solamente el ancho de los pulsos variara con la carga.

Mientras que los amplificadores lineales se ejecutan excelentes en controles de velocidad de alto desempeño, ellos tienen el problema de generación de calor en la etapa de salida, requiriendo enfriamiento por aire forzado en amplificadores sobre 100 a 200 W (dependiendo de la temperatura ambiente y del diseño del disipador de calor) Los amplificadores de conmutación superan este problema por permitir su etapa de salida conmutar rápidamente desde un estado no conductor a un estado completamente conductor, por eso la operación es minimizada de la etapa de salida en la región de alta disipación.

Tres métodos básicos son usados para controlar potencia en amplificadores de conmutación: Modulación de Ancho de Pulso (Pulse Width Modulation PWM),

Modulación de Frecuencia de Pulso (Pulse Frecuency Modulation PFM), y Rectificador Controlado de Silicio (Silicon Controlled Rectifier SCR), sus principales diferencias son mostradas en la figura 4.10.13.

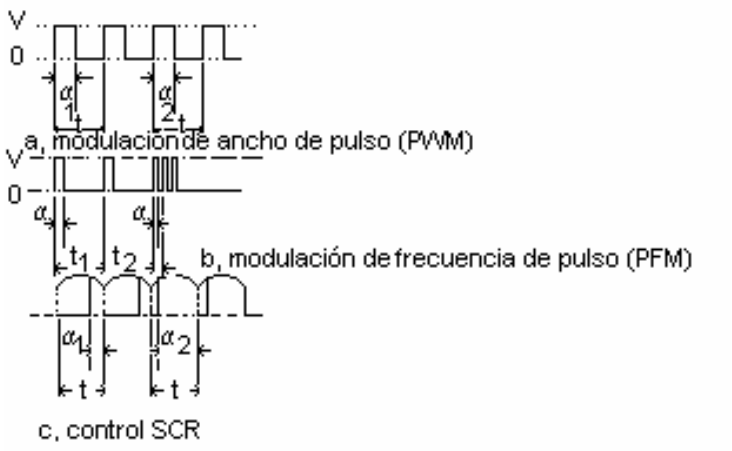


Figura 4.10.13. Formas de onda de voltaje en amplificadores de conmutación.

El sistema PWM usualmente utiliza una fuente de CD, y el amplificador conmuta el voltaje de la fuente a encendido o apagado en una frecuencia determinada y en una variable "ángulo de disparo" α (ver figura 4.9.4.1.a) así que el promedio ajustable de voltaje a través de la carga es establecido. La cantidad de potencia transferida a la carga (motor) dependerá de la proporción de conmutación de encendido y de la inductancia de carga.

El sistema PFM tiene un ángulo de disparo determinado y una proporción de repetición variable (ver figura 4.10.13.b), logrando esencialmente el mismo resultado como el PWM, pero cuando es usado en circuitos de control, mucha de la frecuencia de pulso variable requerida causa problemas de disipación lo cual hace al PWM más atractivo. Dicho sea de paso, un circuito PWM con una proporción de pulso variable es realmente un híbrido entre los dos.

El circuito SCR para control de CD es usualmente usado con un suministro de corriente alterna CA rectificado, aunque el circuito de rectificación puede ser localizado antes o después de la sección de control del amplificador. La figura 4.10.13.c muestra una fuente de voltaje rectificada de onda completa de una determinada frecuencia.

El ángulo de disparo puede ser variado para cubrir una porción desde 0 a 180º α de media onda.

El voltaje de salida promedio no es proporcional al ángulo de disparo, y esta parte requiere atención especial en diseño de control.

De la discusión de los amplificadores en las líneas arriba descritas centradas alrededor de los sistemas de control de velocidad operando en el "primer cuadrante". Ahora describiremos los verdaderos servos amplificadores capaces de proveer voltajes o corrientes de salida positivas y negativas, y así poder operar en los cuatro cuadrantes.

Otra vez, son clasificados los amplificadores dentro de tres categorías: los amplificadores de transistor lineal, los amplificadores SCR, y los amplificadores de conmutación.

4.11.3 Conversión de CD a CD (Chopper)

Los convertidores de onda continua son dispositivos que transforman un nivel continuo a su entrada, entregando otro valor a la salida. Se pueden clasificar en elevadores, reductores y mixtos (elevadores-reductores) El funcionamiento básico de los reductores de tensión consiste en “trocear”, seccionar o muestrear el nivel de continua a la entrada mediante un interruptor, y entregar a la salida, previo filtrado, el valor medio de la corriente troceada y filtrada. En el caso de los elevadores de tensión se añaden elementos almacenadores de energía (bobinas condensadores) que entregan a la carga una tensión añadida a la de la entrada. Los sistemas mixtos pueden funcionar en cualquiera de las configuraciones mencionadas.

En la lista siguiente figuran las principales aplicaciones de los convertidores de corriente directa:

- Alimentación y control de corriente directa.
- Alimentación de equipos de electrónica a partir de baterías, fuentes autónomas de corriente directa.
- Automóviles y demás vehículos eléctricos

En aplicaciones en las que la fuente de energía es una batería y la eficiencia es importante, varios seccionadores proporcionan voltaje terminal variable de armadura a los motores de CD como medio de control de la velocidad. Ejemplos de este tipo son los automóviles alimentados por batería. Dichos seccionadores pueden tener tiristores o transistores de potencia. El Seccionador, Troceador o Chopper es esencialmente un interruptor que conecta la batería durante breves lapsos y que puede variar el valor promedio o de CD del voltaje terminal variando el ancho del pulso o la frecuencia del pulso o ambos. Ver figura 4.10.14.

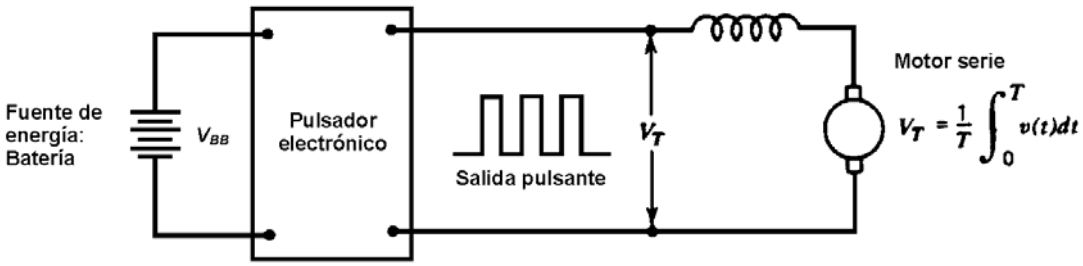


Figura 4.10.14 Control de velocidad mediante V_T variable, el pulsador modifica el ancho y/o frecuencia de los pulsos

En varias aplicaciones es requerido convertir una fuente de voltaje fijo de CD en una fuente de voltaje variable de CD. Un chopper de CD convierte directamente de CD a CD. Los choppers son ampliamente usados para control de motores de tracción en automóviles eléctricos, trolebuses, grúas marinas, montacargas, etc. Proporcionan un control de aceleración suave, alta eficiencia, y rápida respuesta dinámica. Pueden ser usados en frenado regenerativo de motores de CD para regresar energía al sistema, lo cual representa ahorro de energía en sistemas de transporte con paradas frecuentes.

Los seccionadores o choppers se clasifican dependiendo de la dirección del flujo de la corriente y voltaje como:

- Clase A.- La corriente de carga fluye hacia la carga. La corriente y el voltaje son positivos.
- Clase B.- La corriente de carga fluye fuera de la carga. El voltaje es positivo y la corriente es negativa.
- *Clase C.*- La corriente de carga puede ser positiva o negativa y el voltaje es siempre positivo.
- *Clase D.*- La corriente de carga es siempre positiva. El voltaje de la carga puede ser positivo o negativo.
- *Clase E.*- Tanto la corriente de carga como el voltaje de la carga pueden ser positivos o negativos en sus 4 combinaciones.

El analizando como ejemplo el seccionador de Clase E con el cual teniendo el voltaje y corriente positivos en la carga (motor) se puede controlar para que gire en un sentido y si se mantienen negativos girará en el otro sentido. El esquema básico de un seccionador Clase E se muestra en la figura 4.10.15, los interruptores determinan la polaridad del voltaje y el sentido de la corriente en el motor o carga, siendo que los interruptores pueden ser implementados de muchas formas.

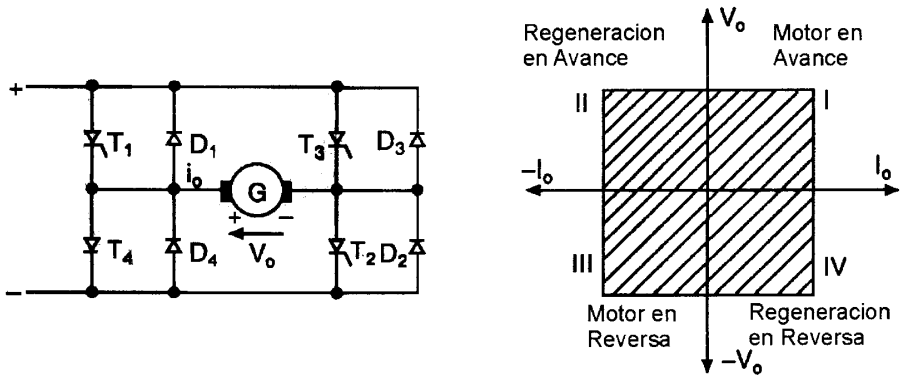


Figura 4.10.15 Seccionador Clase E

Este seccionador es llamado de 4 cuadrantes debido a que opera en 4 modos, como se aprecia en la grafica de la figura 4.10.15. En el 1er. cuadrante T_1 y T_4 conducen. El voltaje y la corriente son positivos, por lo que la maquina opera como motor en avance. Si la fem (fuerza electromotriz) del motor llega a ser mayor que el voltaje de entrada con la polaridad positiva entonces D_1 y D_2 entran en conducción y la máquina (motor) entrega corriente a la fuente. El voltaje es positivo y la corriente es negativa por lo que cae en el 2do. cuadrante y la maquina opera en el modo de generación de avance. Si T_3 y T_4 se cierran la corriente y el voltaje son opuestos a las direcciones de referencia, siendo negativos. La máquina opera en motor en reversa, y se encuentra en el tercer cuadrante. Si la polaridad de la fem generada polariza directamente a los diodos D_3 y D_4, la maquina entrega corriente a la fuente y esta en modo de generación de reversa.

4.11.4 Smart power

La expresión smart power se refiere a la tecnología de integración en un dispositivo monolítico de uno o varios componentes de potencia y de componentes lógicos o analógicos de tratamiento de señal.

Campos de aplicación:

- Sistemas basados en microprocesador.
- Motores (CD, CA y paso a paso).
- Pantallas planas.
- Telecomunicaciones.
- Cabezales de impresora.
- Fuentes de alimentación.
- Lámparas (automóvil).

Estos circuitos integrados disipan una potencia apreciable (2-4 A) Algunos pueden incluso llevar la etapa de control (Circuitos Integrados Inteligentes).

Para integrar en una pastilla la parte de potencia y la parte de control, se han usado dos tecnologías: *la bipolar y la mixta*.

La tecnología bipolar consiste en la utilización de soluciones bipolares para cada uno de los elementos de potencia y de control. *La tecnología mixta* se basa en la realización de la parte de potencia y de la parte de control mediante procesos diferentes.

Según el tipo de uso que necesitemos escogeremos una tecnología de fabricación, optando por CMOS cuando la parte de control del circuito smart power ha de incluir funciones digitales. Ello se debe a un menor consumo de potencia y al hecho de no presentar dependencias entre la ganancia y la corriente.

Con estos circuitos obtenemos un mayor rendimiento y una mayor facilidad de implementación, ya que los circuitos de control no hay que diseñarlos, los tenemos hechos. Esto hace que su implantación en el mercado vaya creciendo con el paso de los años, al proporcionar soluciones a múltiples necesidades, con un bajo costo y sencillez.

Capítulo 5

Sensores

La función de los sensores del robot puede dividirse en dos categorías principales: Estado interno y estado externo. Los sensores de estado interno operan con la detección de variables, tales como la posición de la articulación del brazo, que se utilizan para el control del robot. Por el contrario, los sensores de estado externo operan con la detección de variables tales como el alcance, la proximidad y el contacto. La detección externa, se utiliza para el guiado del robot. así como para la manipulación e identificación de objetos.

Los sensores de estado externo pueden clasificarse también como sensores de contacto o no contacto. Como su nombre indica, la primera clase de sensores responde al contacto físico, tal como el tacto, deslizamiento y torsión.

Los sensores de no contacto se basan en la respuesta de un detector a las variaciones en la radiación electromagnética o acústica. Los ejemplos más destacados de los sensores de no contacto miden el alcance, la proximidad y las propiedades visuales de un objeto.

Los sensores de fuerza y de torsión se utilizan como dispositivos de realimentación para controlar la manipulación de un objeto una vez que se haya agarrado (por ejemplo, para evitar el aplastamiento del objeto o para impedir su deslizamiento).

5.1 Sensores de proximidad ópticos

Los sensores de proximidad ópticos son similares a los sensores ultrasónicos en el sentido de que detectan la proximidad de un objeto por su influencia sobre una onda propagadora que se desplaza desde un transmisor hasta un receptor. Uno de los métodos más utilizados para detectar la proximidad por medios ópticos se ilustra en la figura 5.1.1. Este sensor está constituido por un diodo emisor de luz de estado sólido (LED), que actúa como un transmisor de luz infrarroja y un fotodiodo de estado sólido que actúa como el receptor. Los conos de luz formados enfocando la fuente y el detector en el mismo plano se intersectan en un volumen largo en forma de lápiz. Este volumen define el campo de operación del sensor, puesto que una superficie reflectora que intersecta el volumen se ilumina por la fuente y es «vista» simultáneamente por el receptor.

Es importante destacar que el volumen de detección mostrado en la figura 5.1.1 no proporciona una medición del punto. Dicho de otro modo, una superficie localizada en cualquier lugar en el volumen producirá una lectura. Aunque es

posible calibrar la intensidad de estas lecturas como una función de la distancia para características reflectoras y orientaciones del objeto conocidas, la aplicación típica de la disposición mostrada en la figura 5.1.1 está en un modo en donde una señal binaria es generada cuando se recibe una intensidad de luz superior a un valor umbral.

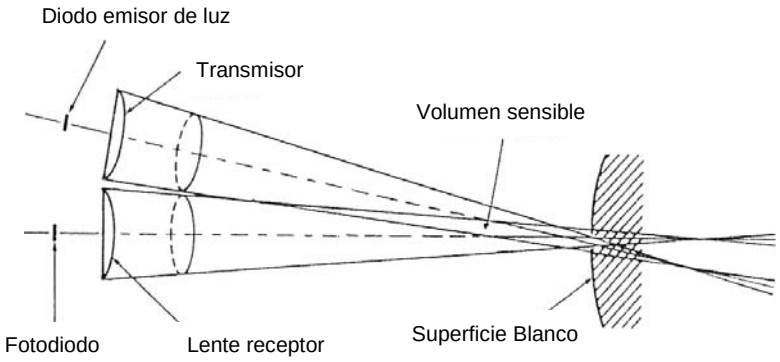


Figura 5.1.1 Sensor de proximidad óptico. (De Rosen y Nitzan [1977]). (© IEEE.)

5.2 Sensores de contacto

Estos sensores se utilizan en robótica para obtener información asociada con el contacto entre una mano manipuladora y objetos en el espacio de trabajo.

Cualquier información puede utilizarse, por ejemplo. para la localización, y el reconocimiento del objeto, así como para controlar la fuerza ejercida por un manipulador sobre un objeto dado. Los sensores de contacto pueden subdividirse en dos categorías principales: binarios y analógicos. Los sensores binarios son esencialmente conmutadores que responden a la presencia o ausencia de un objeto. Por el contrario, los sensores analógicos proporcionan a la salida una señal proporcional a una fuerza local. Estos dispositivos se examinan con más detalle en las secciones siguientes:

Sensores binarios

Como se indicó anteriormente. Los sensores binarios son dispositívos de contacto, tales corno microinterruptores. En la disposición más simple un conmutador está situado en la superficie interior de cada dedo de una mano de manipulación. Este tipo de detección es de utilidad para determinar si una pieza está presente entre los dedos. Desplazando la mano sobre un objeto y estableciendo secuencialmente contacto con su superficie, también es posible centrar la mano sobre el objeto para su agarre y manipulación.

Sensores de contacto binarios múltiples pueden emplearse, en la superficie

interior de cada dedo, para proporcionar información táctil. Además, suelen estar montados en las superficies exteriores de una mano de manipulación para proporcionar señales de control de utilidad para guiar la mano a través de todo el espacio de trabajo. Este último empleo de detección por contacto es análogo al que los seres humanos sienten cuando se desplazan a través de un recinto completamente oscuro.

Sensores analógicos

Un sensor de contacto analógico es un dispositivo manejable cuya salida es proporcional a una fuerza local. El más simple de estos dispositivos está constituido por una varilla accionada por resorte (figura 5.2.1) que está mecánicamente enlazada con un eje giratorio, de tal manera que el desplazamiento de la varilla debido a una fuerza lateral da lugar a una rotación proporcional del eje. La rotación se mide luego, de manera continua, utilizando un potenciómetro o de forma digital con el empleo de una rueda de código. El conocimiento de la constante del resorte proporciona la fuerza que corresponde a un desplazamiento dado.

En los últimos años se dedicó un esfuerzo considerable al desarrollo de conjuntos de detección táctil, capaces de proporcionar una información de contacto sobre un área más amplia que la proporcionada por un sensor único. El empleo de estos dispositivos se ilustra en la figura 5.2.2, que muestra una mano de robot en la que la superficie interior de cada dedo ha sido recubierta con un arreglo táctil de detección. Las placas detectoras exteriores suelen ser dispositivos binarios y tienen la función descrita al final de la sección de sensores binarios.

Aunque pueden formarse arreglos de detección utilizando sensores individuales múltiples, una de las soluciones más prometedoras a este problema consiste en utilizar un arreglo de electrodos en contacto eléctrico con un material conductor dúctil (por ejemplo, sustancias basadas en grafito) cuya resistencia varia como una función de la compresión. En estos dispositivos, que suelen denominarse *pieles artificiales,* un objeto que presiona contra la superficie produce deformaciones locales que se miden corno variaciones continuas de la resistencia. Estas últimas se transforman con facilidad en señales eléctricas, cuya amplitud es proporcional a la fuerza que se aplica en cualquier punto dado sobre la superficie del material.

Algunos métodos básicos utilizados en la construcción de pieles artificiales se muestran en la figura 5.2.3. El esquema mostrado en la figura 5.2.3a está basado en un concepto de «ventana», caracterizado por un material conductor dispuesto en «sandwich» entre una masa común y un arreglo de electrodos grabados en una placa de circuito impreso de fibra de vidrio. Cada electrodo está constituido por un área rectangular (con la denominación de *ventana)* que define un punto de contacto

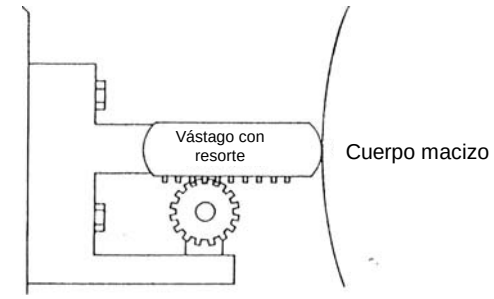


Figura 5.2.1 Un sensor de contacto analógico básico

La corriente circula desde la masa común hasta los electrodos individuales como una función de compresión del material conductor.

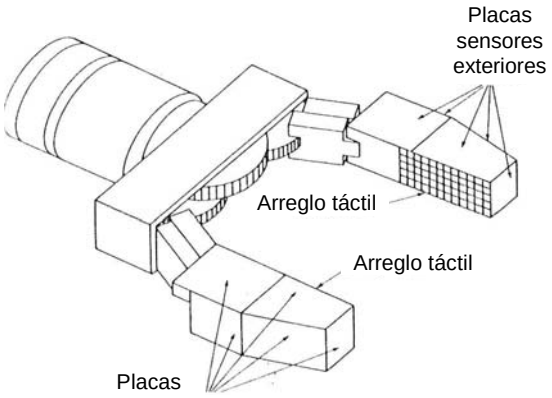


Figura 5.2.2 Un robot provisto de arreglo de sensores táctiles

En el método mostrado en la figura 5.2.3b, pares de electrodos estrechos y largos están situados en el mismo plano de sustrato con circuitos electrónicos activos que utilizan tecnología de integración a baja escala (LSI). El material conductor está situado por encima de este plano y aislado del plano del sustrato,

excepto en los electrodos. Los cambios de la resistencia resultantes de la compresión del material se miden e interpretan por los circuitos activos localizados entre los pares de electrodos.

Otra técnica posible se ilustra en la figura 5.2.3c En este método, el material conductor se sitúa entre dos arreglos de electrodos delgados, planos y flexibles que se intersectan en sentido perpendicular. Cada intersección, y el material conductor intermedio, constituyen un punto de detección. Los cambios en la resistencia, como una función de la compresión del material, se miden excitando eléctricamente los electrodos de un arreglo (uno a uno) y midiendo la corriente que circula por los elementos del otro arreglo. La magnitud de la corriente en cada uno de estos elementos es proporcional a la compresión del material entre ese elemento y el que se excita desde el interior.

Por último, la disposición mostrada en la figura 5.2.3b exige el empleo de

un material anisotrópicamente conductor. Dichos materiales tienen la propiedad de ser eléctricamente conductores en una sola dirección. El sensor se construye utilizando un arreglo lineal de electrodos delgados y planos en la base. El material conductor se coloca sobre la parte superior de la base, con el eje de conducción perpendicular a los electrodos y separados de ellos por una malla, de modo que no exista ningún contacto entre el material y los electrodos en la ausencia de una fuerza.

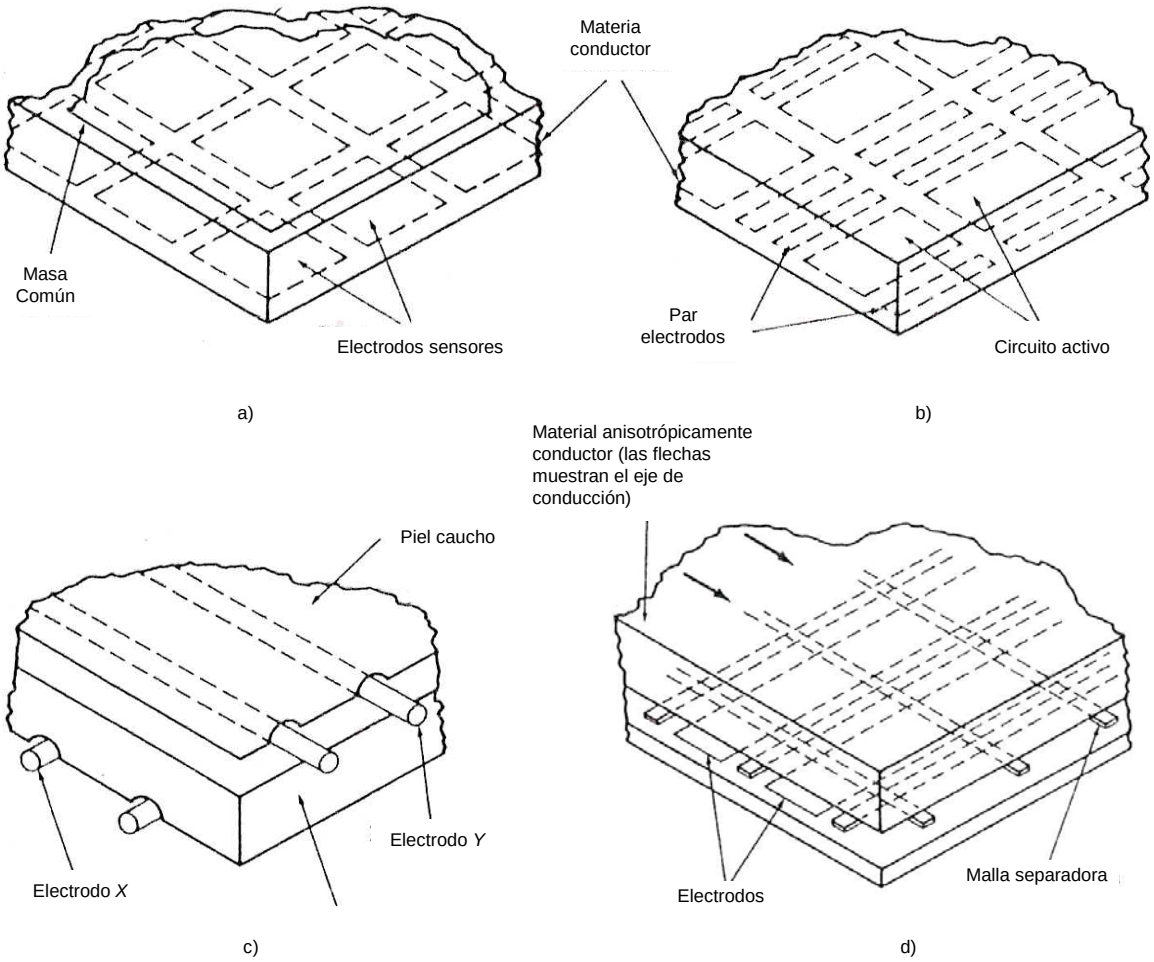


Figura 5.2.3 Cuatro métodos para construir pieles artificiales

La aplicación de fuerza suficiente produce el contacto entre el material y los electrodos. Cuando aumenta la fuerza también lo hace el área de contacto, lo que da lugar a una resistencia más baja. Como ocurría con el método de la figura 5.2.3c, un arreglo se excita desde el exterior y la corriente resultante se mide en el otro. Hay que destacar el hecho de que la sensibilidad de contacto depende del espesor del separador.

Los métodos de la figura 5.2.3c y d están basados en la excitación secuencial de los elementos de uno de los arreglos. Esto suele dar lugar a

dificultades al interpretar señales que resulten de configuraciones de contacto complejas debido a las inducciones de «punto de cruce» producidas por caminos eléctricos alternativos. Una solución consiste en situar un elemento de diodo en cada intersección para eliminar la circulación de corriente a través de los caminos alternativos. Otro método es poner a masa todos los caminos, con la excepción de que está excitado. Explorando el arreglo receptor camino a camino, seremos capaces de «ver» la contribución de las intersecciones de los elementos individuales.

5.3 Introducción a la toma de medidas en sistemas físicos

5.3.1 Introducción

Sensor

Elemento encargado de medir una magnitud cualquiera en un sistema físico. (Véase figura 5.3.1)

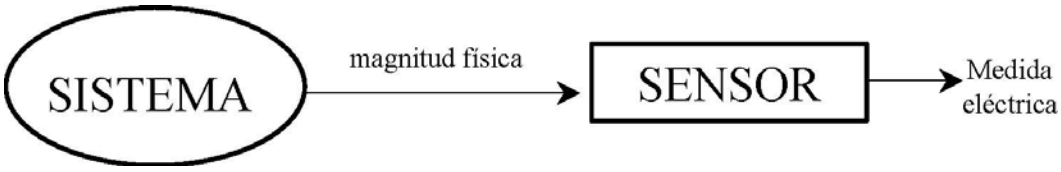


Figura 5.3.1 Módulo de un Sensor

Ejemplos de magnitudes a medir:

- Temperatura, caudal, nivel de un depósito, velocidad de un giro de un motor,

En general, sensor = transductor

Transductor: dispositivo que convierte una señal física en otra señal física de otro tipo.

Sensor: dispositivo que convierte una señal física en una señal eléctrica.

En robótica

Sensor = transductor de entrada (obtención de la información)

Actuador o accionamiento = transductor de salida (conversión de la energía)

5.3.2 Finalidad de la toma de medidas de un sistema físico

Presentación: Las medidas se muestran mediante un indicador analógico o digital

Registro: Los valores que toma un determinado parámetro son almacenados.

Control: La medida de un parámetro es utilizada en un bucle de control

Ejemplo: Servomotor. (Ver figura 5.3.2)

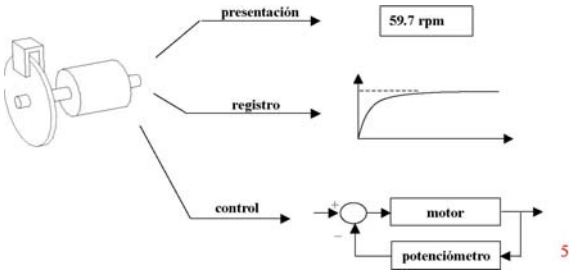


Figura 5.3.2 Servo Motor.

5.3.3 Tipo de señales a obtener con un sensor transductor

Transductor: Convierte una señal física de cualquier tipo en otra señal de un tipo distinto. (Véase figura 5.3.3)

Figura 5.3.3 Representación de un elemento transductor

Los transductores normalmente utilizados para la toma de medidas son aquellos cuya señal de salida *y(t)* es de tipo eléctrico: tensión, intensidad, frecuencia, ...

Las razones son 2:

- Las señales eléctricas se pueden usar fácilmente para registro, visualización y control (son prácticas)
- La variación de cualquier magnitud física de un objeto siempre produce una variación en alguna característica eléctrica (son fáciles de obtener)

Un transductor ideal debería ofrecer una señal eléctrica de salida *y(t)* proporcional (lineal) a la magnitud que se desea medir *x(t)*:

$$y(t) = K\,x(t)$$

Normalmente esto no se consigue y la relación entre la magnitud a medir y la señal eléctrica generada no es proporcional sino que obedece a una expresión más complicada:

$$y(t) = f[x(t)]$$

Esto complica la obtención de la medida a partir del valor de la señal eléctrica. A veces es necesario utilizar tablas de conversión de mediciones proporcionadas por el fabricante del dispositivo a medir.

5.3.4 Etapas en la obtención de una medida

La mayoría de las veces, la obtención de una medida no se puede lograr en una sola etapa.

Procesos anteriores: Sensores primarios

Procesos posteriores: Acondicionamiento

Procesos anteriores: Sensores primarios

A veces, la magnitud a medir no es fácilmente traducible a una señal eléctrica y es más sencillo realizar un proceso previo. (Ver figura 5.3.4)

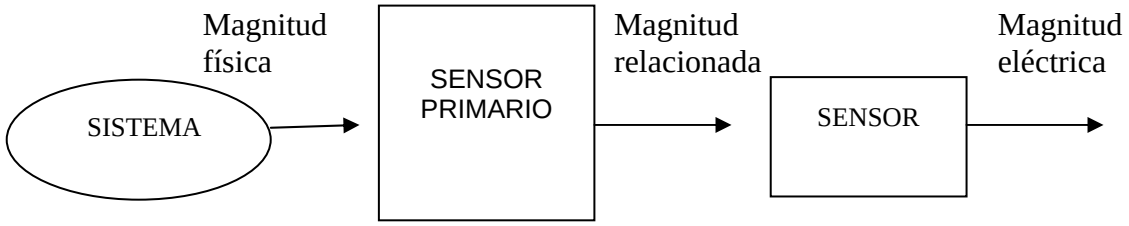


Figura 5.3.4 Traducción de una señal física.

Procesos posteriores: Circuitos acondicionadores. (Ver figura 5.3.5)

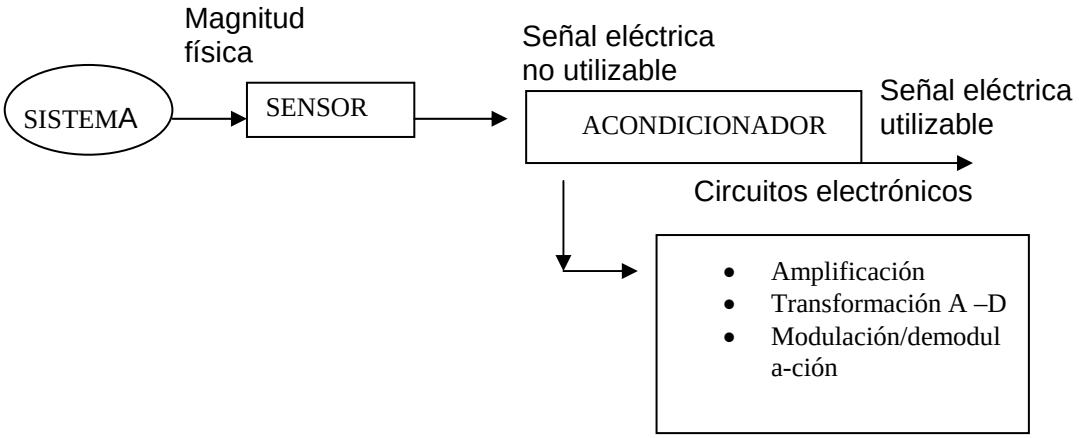


Figura 5.3.5 Circuito acondicionador

5.3.5 Clasificación de los sensores

Existen diversas formas de clasificar los sensores: (ver tabla 5.3.1)

- Según la magnitud a medir:

  Posición, velocidad, aceleración, temperatura, fuerza, presión, caudal, nivel, humedad, ...

- Según el tipo de salida ofrecida:

  Analógicos, digitales, cuasidigitales

- Según el principio de funcionamiento:

  Resistivos, capacitivos, inductivos, electromagnéticos, generadores, digitales, básicos en semiconductores, básicos en ultrasonidos, …

	Magnitudes								
Sensores	**Posición Distancia Desplazamiento**	**Velocidad**	**Aceleración Vibración**	**Temperatura**	**Presión**	**Caudal Flujo**	**Nivel**	**Fuerza**	**Humedad**
Resistivos	Potenciómetros Galgas Magnetorresis - tencias		Galgas + masa - resorte	RTD Termistores	Potenciómetros + tubo Bourdon	Anemómetros de hilo caliente Galgas. voladizo Termistores	Potenciómetro + flotador Termistores LDR	Galgas	Humistor

Capacitivos	Condensador di ferencial				Condensador variable + diafragma		Condensador variable	Galgas capacitivas	Dieléctrico variable
Inductivos y electro magnéticos	LVDT Corrientes Foucault Resolver Inductosyn Efecto Hall	Ley Faraday LVT Efecto Hall Corrientes Foucault	LVDT + masa-Resorte		LVDT + diafragma Reluctancia Variable + diafragma	LVDT + rotámetro Ley Faraday	LVDT + flotador Corrientes foucault	Magneto-elástico LVDT + célula carga	
Generadores			Piezo eléctricos + masa-resorte	Termopares Piroeléctricos	Plezoeléctricos			Piezo eléctricos	
Digitales	Codificadores Incrementales y absolutos	Codificadores Incrementales		Osciladores de cuarzo	Codificador + tubo Bourdon	Vórtices			SAW
Uniones p-n	Fotoeléctricos			Diodo Transistor Convertidores T/I			Fotoeléctricos		
Ultrasonidos	Reflexión	Efecto Doppler				Efecto Doppler Tiempo tránsito Vórtices	Reflexión Absorción		

Tabla 5.3.1 Clasificación de sensores

Hay que tomar en cuenta las siguientes características para considerar al momento de elegir el sensor más adecuado para una aplicación.

Características:

Estáticas: Errores que aparecen en las medidas como diferencia entre los valores reales y los valores indicados por el sensor

Dinámicas: Relativas a la velocidad de respuesta del sensor

Características estáticas:

- Exactitud
- Fidelidad
- Sensibilidad
- Linealidad
- Resolución
- Derivas

5.3.6 Características estáticas

- Exactitud de un sensor: Se refiere a la diferencia entre la medida ofrecida por el instrumento y el valor real de la magnitud que se mide. Esta diferencia es el error del sensor y puede expresarse de distintas formas:
  - Error absoluto

    $$error\ absoluto = medida\_sensor - valor\_real$$

  - Error relativo

    $$error\ relativo = error\ absoluto\ /valor\_real$$

  - Clase de precisión de un sensor: Indica el máximo error porcentual que puede presentar un sensor, en general con respecto al valor de fondo de escala o al rengo de medida

    $$error\ máximo = [índice\_clase - fondo\_escala]/100$$

- Fidelidad de un sensor: No tiene en cuenta la magnitud del error con respecto al valor real sino la capacidad del instrumento de dar el mismo resultado al realizar varias medidas, siempre en las mismas condiciones.
-

Ensayos de fidelidad: Repetibilidad y reproducibilidad

Fidelidad: Máxima diferencia que existe entre 2 valores obtenidos por el mismo sensor en medidas distintas de una misma magnitud, con una probabilidad del 95%

- Exactitud y fidelidad de un sensor: La figura 5.3.6 muestra dos sensores, uno con gran exactitud y baja fidelidad y otro con menor exactitud pero mayor fidelidad.

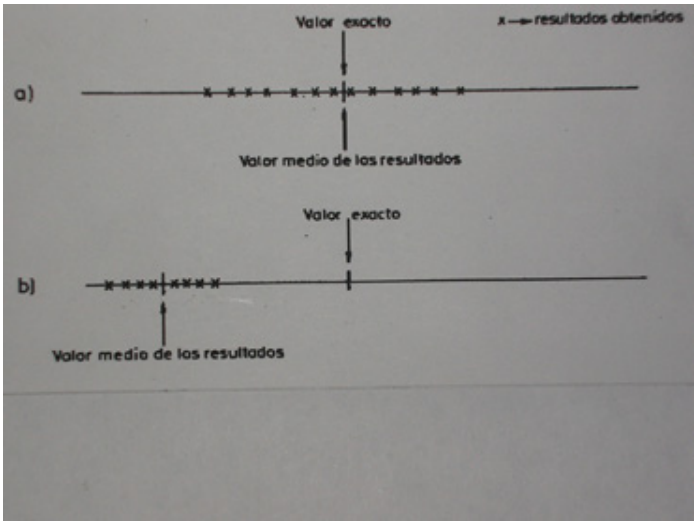
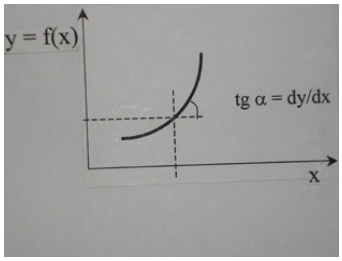
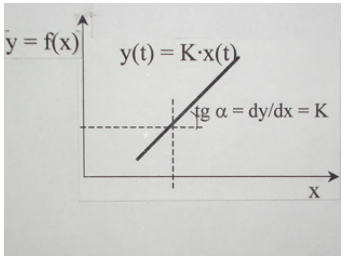


Figura 5.3.6 a) Gran exactitud, baja fidelidad; b) Menor exactitud, mayor fidelidad

Sensibilidad o factor de escala: Pendiente de la curva de calibración o curva que relaciona la medida devuelta por el sensor y la magnitud real medida.	
Linealidad de un sensor: El sensor ideal, tiene una sensibilidad constante, es un sensor lineal.	

igura 5.3.7 Curvas de Sensibilidad y Linealidad.

- Linealidad de un sensor

Para valorar lo cercano o alejado que se encuentra un sensor determinado del comportamiento ideal, se utiliza la linealidad, que se define como la divergencia entre la curva de calibración y la recta que mejor se aproxima a ella ajustada por mínimos cuadrados. (Ver figura 5.3.7)

- Resolución de un sensor

Mínima variación que es preciso que se produzca en la magnitud a medir para que se produzca una variación en la respuesta del sensor.

- Derivas de un sensor

Variaciones de la medida ofrecida por un sensor a lo largo del tiempo. Su origen es térmico, debido al calentamiento de los componentes.

- Deriva de cero

Variación de salida con entrada nula.

- Deriva de factor de escala

Variación de la sensibilidad

5.3.7 Características dinámicas

Estudian la respuesta del sensor ante señales a medir que varían en el tiempo. Para ello, el sensor se representa por su relación entrada-salida mediante su función de transferencia. (Ver figura 5.3.8)

Figura 5.3.8 Representación de un sensor por su transferencia

Se analiza el comportamiento dinámico de los sensores

- En el dominio del tiempo (respuesta en régimen transitorio)
- En el dominio de la frecuencia (respuesta en régimen permanente)

La mayor parte de los sensores se pueden aproximar como sistemas de orden 0 (no tienen dinámica), de orden 1 ó de orden 2. En la tabla 5.3.2 se muestra la función de cada orden, y en las imágenes de la figura 5.3.9 se representan las gráficas en respuesta a tres señales de excitación distinta, la figura 5.3.10 muestra las gráficas de la función orden 2 en respuesta a dos tipos de respuesta de excitación.

orden 0	$y(t) = k \cdot x(t) \rightarrow Y(s) = k \cdot X(s)$
orden 1	$dy(t)/dt + a \cdot y(t) = k.x(t) \rightarrow Y(s) = (k/(s-a)) \cdot X(s)$
orden 2	$\mathrm{d2y(t)/dt\ 2} + \mathrm{dy(t)}/dt + b \cdot y(t) = K \cdot x(t) \rightarrow Y(s) = (k/(s2 + a \cdot s + b)) \cdot X(s)$

Tabla 5.3.2 Funciones de orden dinámico para sensores

Orden 1 $y(t) = k \cdot x(t) \rightarrow Y(s) = k \cdot X(s)$

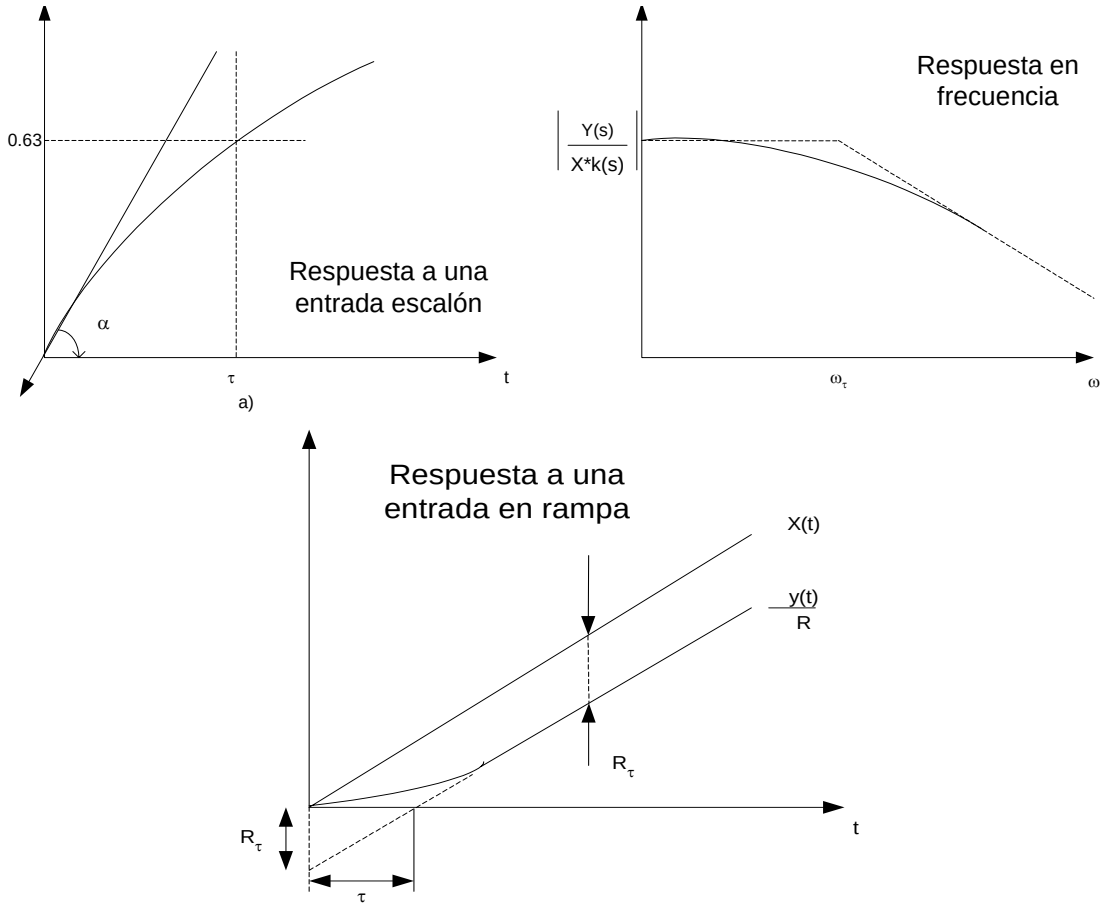


Figura 5.3.9 Gráficas representan respuesta a varias señales de entrada para función de orden 0.

Orden 2 $$\frac{d^2y(t)}{dt^2} + \frac{dy(t)}{dt} + by(t) = Kx(t) \rightarrow Y(s) = \frac{k}{s^2 + as + b} X(s)$$

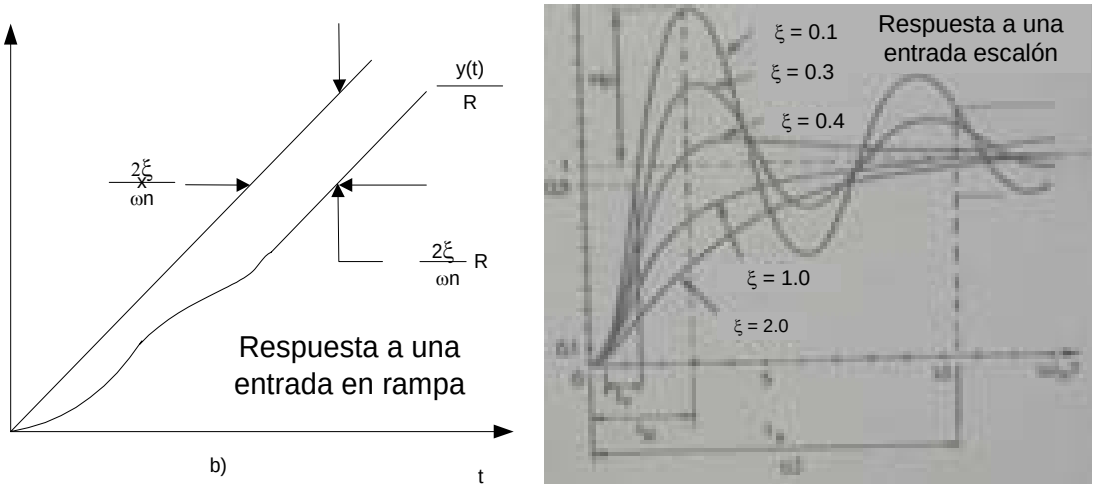


Figura 5.3.10 Gráfica en respuesta a dos tipos de señales de entrada distinta para la función de orden 2.

5.3.8 Influencia del sensor sobre la magnitud a medir

Idealmente, al introducir un sensor en un sistema cualquiera no se altera la magnitud a medir. Esto nunca es completamente cierto en la realidad, y lo que se pretende es que la influencia del sensor sobre la magnitud a medir sea lo menor posible.

En general, la influencia será tanto más grande cuanto mayor sea la cantidad de energía que el sensor extrae del sistema, o cuanto mayor sea la "carga" que el sensor supone para el sistema.

Error por carga: El que se comete en la medida debido a la energía que el sensor extrae del sistema durante el proceso de medición.

Capítulo 6

Construcción física del control del brazo robótico y pruebas de su funcionamiento

En este capítulo se explica la construcción física del control del brazo robótico al igual que las pruebas de funcionamiento realizadas en el brazo, problemas y arreglos finales. Teniendo en consideración la existencia de un brazo robótico con 5 grados de libertad y motores de CD de baja potencia, además de kit de Xilinx con una tarjeta Spartan-3. La aplicación de jugar Gato se refiere al juego de dominio general y conocido, donde se tiene un tablero dividido con el signo de gato de (#) donde cada cuadro representa una posición para jugar. Se planteó el proyecto para que el brazo robótico sea el primer jugador y cualquier gente fuera el contrincante. El proyecto requirió del diseño, planeación, programación (rutas a seguir e inteligencia del juego) y armado de todos los componentes involucrados para esta aplicación incluyendo los siguientes:

- El armado del tablero de juego donde están las casillas o posiciones para jugar,
- El diseño del mismo de tal forma que incluyera sensores para detectar el juego del contrincante y las casillas seleccionadas por él mismo,
- El espacio necesario para que el brazo robótico se mueva con libertad e incluyera los PBC (Print Board Circuits) lo más discretamente posible,
- La selección e implantación de sensores sobre el brazo para ubicar la posición de cada motor que mueve el brazo,
- La programación de la tarjeta Spartan-3 para el movimiento de cada motor de corriente directa del brazo robótico, (pinza, codo, hombro y base) para ubicar una pelotita sobre las casillas del tablero,
- La programación de la lógica del funcionamiento del juego “GATO”, por lenguaje VHDL,
- El diseño de los componentes electrónicos como drivers, sensores, elementos eléctricos, y adaptadores que se utilizarían para conectar la interfase Spartan-3 a los motores del brazo robótico a través de lenguaje VHDL,
- Selección de fuente de alimentación para todos los componentes,
- Selección del tipo de pelotas para utilizarse como fichas.

De forma general, el diagrama a bloques del proyecto propuesto se muestra en la figura 6.1, incluyendo la dirección y sentido de la comunicación entre módulos o entrega de señales entre ellos. Para cada módulo se indica el voltaje de CD con que se alimenta dicho módulo, las otras señales indicadas sin etiqueta son señales de control.

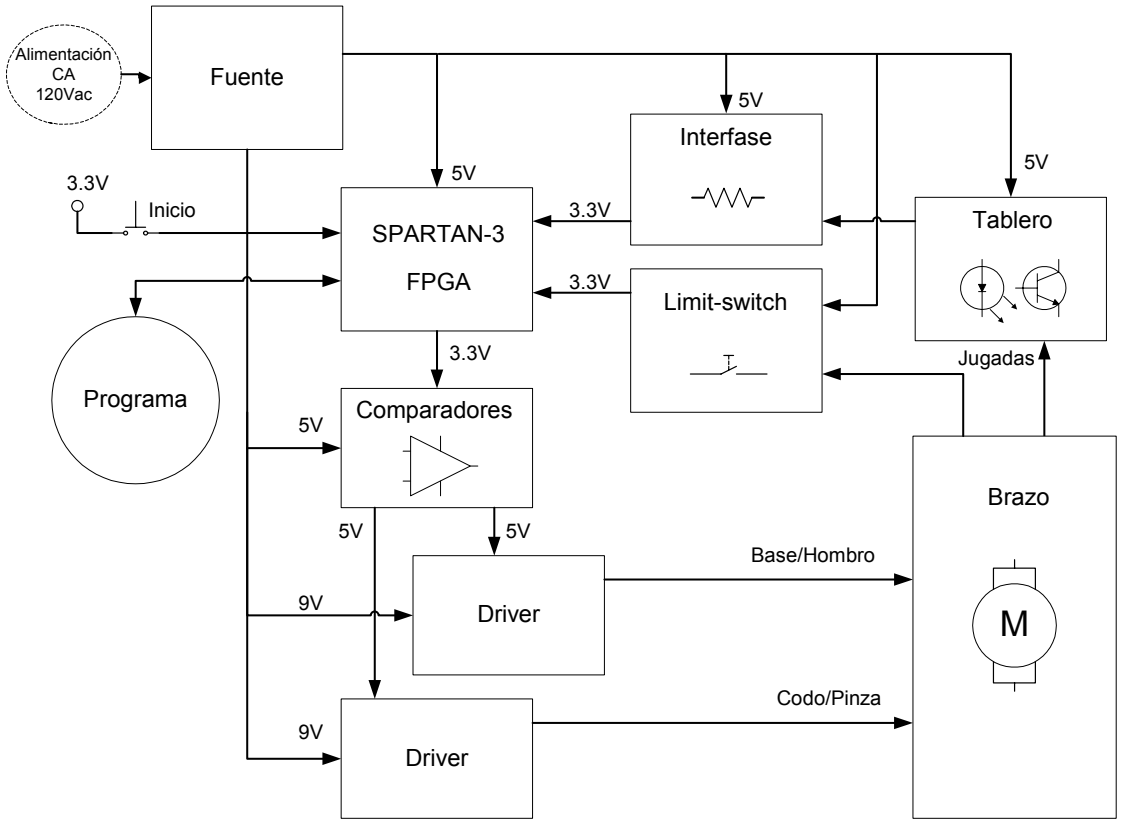


Figura 6.1 Diagrama General de Bloques del Proyecto

6.1 Selección de fuente de alimentación

Se seleccionó una fuente de alimentación de corriente directa (CD) conmutada de una computadora, por la facilidad de que ya está armada de una sola pieza, con alimentación de corriente suficiente para el proyecto y entrega dos voltajes fijos que nos interesan para el proyecto +5V CD y +12V CD. Con éstos voltajes alimentados al proyecto los derivamos y obtenemos los voltajes necesarios para los motores (+9V CD) y la lógica de la tarjeta Spartan-3 (+3.3V CD) Otra razón por la que se decidió el uso de una fuente ya armada es porque el tema de interés del proyecto no incluía la fabricación de la fuente de alimentación.

6.2 Adaptación de sensores del proyecto

6.2.1 Adaptación de sensores en el Brazo Robótico

El brazo robótico utilizado para el proyecto se muestra en la figura 6.2.1, el cual es un brazo robótico comercial con cinco grados de libertad. Para funcionamiento de nuestro proyecto requerimos únicamente cuatro grados de libertad y, por tanto, no se utilizó uno de ellos, el correspondiente a la muñeca del brazo. Los otros cuatro motores de CD, correspondientes a los cuatro grados de libertad se ajustaron lo más posible limpiándolos, engrasándolos y atornillándolos correctamente para que tuvieran el menor desajuste posible y minimizar los

errores por la tolerancia del movimiento provocado por el juego en los engranes en cada motor o la inercia del movimiento, y además tratando de reducir los problemas por el uso de este tipo de motores pues considerando que los motores del brazo son de CD del tipo de imán permanente y no son de exactitud y para nuestro proyecto requerimos un poco de precisión porque los movimientos principalmente se basan por intervalos de tiempos. Los movimientos de los motores son los siguientes:

Base (giro a la derecha o a la izquierda),

Hombro (sube y baja),

Codo (sube y baja) y

Pinza (abre y cierra).

Después de ajustar el brazo en la posición final intentado que ejecutara el menor número de errores posibles, se detectó que sólo se requería fijar tres sensores para los movimientos del motor del hombro, codo y base del robot, porque el movimiento de las pinzas está limitado al abrir a un tope máximo y para cerrar al choque de las mismas pinzas o el volumen de la pelota tomada. Por tanto, los motores del hombro, codo y base se requirió limitar con un interruptor cada uno, con interruptores de los llamados micro-switch del tipo Interruptores de fin de carrera (limit-switch). El ajuste final de los switches se realizó a mano hasta ajustar la distancia necesaria para el movimiento de cada motor.

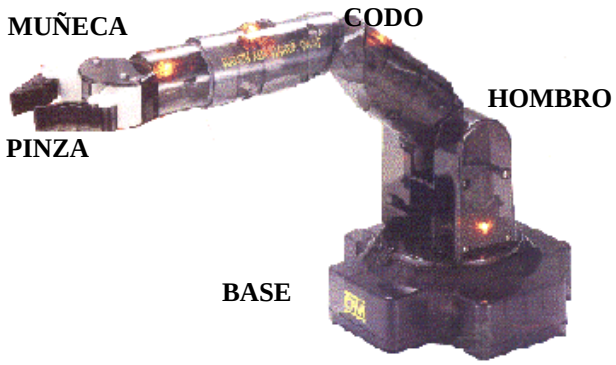


Figura 6.2.1 Grados de Libertad del Brazo Robótico

De esta forma, las consideraciones para definir la conexión de los Interruptores de carrera para cada motor fueron los siguientes. La fuente de alimentación para la lógica del motor es +5V CD y los valores de entrada para los puertos de la tarjeta Spartan-3 es de +3.3V CD para el FPGA, (de acuerdo a las especificaciones de la tarjeta Spartan-3 mostradas en el capítulo 2) por lo que nuestro diagrama quedaría como se indica en la figura 6.2.3.

La forma en que se diseñaron los valores de las resistencias fue la siguiente, considerando una sólo malla de la figura 6.2.3

Si:

$$Vin=5VDC$$

$Vo=3.3VCD$

$R1\ y\ R2 = ¿?$

Por la ley de kirchoff, de suma de voltajes:

$$Vin= Iin*R1 +Vo$$

$$(Vin-Vo)/R1=Iin$$

Y como Vo = 3.3 VCD , entonces el valor de la resistencia R_1 depende directamente de la cantidad de corriente, y como se pretende que éste circuito consuma lo menos posible de corriente, entonces se le da un valor a R_1 comercial, de tal forma que la corriente no sea grande, más bien del rango de menos de 1mA, por lo tanto:

Si $R_1 = 2.2K\Omega$

Entonces; $I_{in} =(Vin-Vo)/R_1$ y

$I_{in} = 0.77$ mA, lo cual es correcto.

Y entonces para obtener el valor de R_2 tenemos por ley de Ohm

$$Iin*(R_1+R_2)=Vin$$

Y; $R_2 = (Vin /Iin) –R_1$

Por tanto; $R_2 = 4.3\ K\Omega$

Pero comercialmente el valor más cercano es R_2 = 4.7 KΩ

$$Iin*R_2=Vo$$

Y por tanto, el diagrama con una sólo malla queda como se indica en la figura 6.2.2

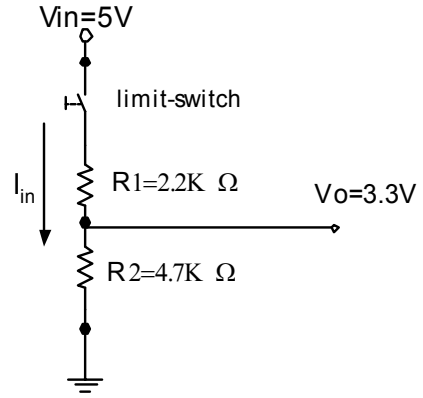


Figura 6.2.2 Diagrama del interruptor Limit-switch para conectarlo a la Spartan

La fuente de alimentación es de +5V CD por lo que se requiere de un divisor de voltaje para obtener los +3.3V CD los cuales son solicitados para mandar las señales a la tarjeta SPARTAN-3. Cuando el interruptor está abierto, la señal de salida está conectada a tierra a través de la resistencia de R_2 y tenemos un 0 lógico; cuando el interruptor se cierra, se genera la malla de la figura 6.2.2 y conecta la fuente de 5 V CD al divisor de voltaje para entregar un 1 lógico.

Los tres interruptores limit-switch se fijaron a la superficie de policarbonato del brazo robótico con pegamento de contacto y se ajustó una manija metálica en la medida que cumplía los requerimientos para apagar el motor del brazo para

limitar el movimiento. Este ajuste se hizo manualmente en ambos motores a través de prueba y error hasta dejarlo fijo.

Las mismas consideraciones de voltaje de alimentación se tomaron para diseñar los componentes del interruptor de inicio, pues en ambos casos sólo se requiere de realizar un divisor de voltaje para la entrada de la señal a los puertos de la tarjeta Spartan-3, sólo que para este último se utilizó un interruptor de contacto o del tipo push-buttom. El diagrama de conexión incluyendo los 3 interruptores que se ilustran en la figura 6.2.3

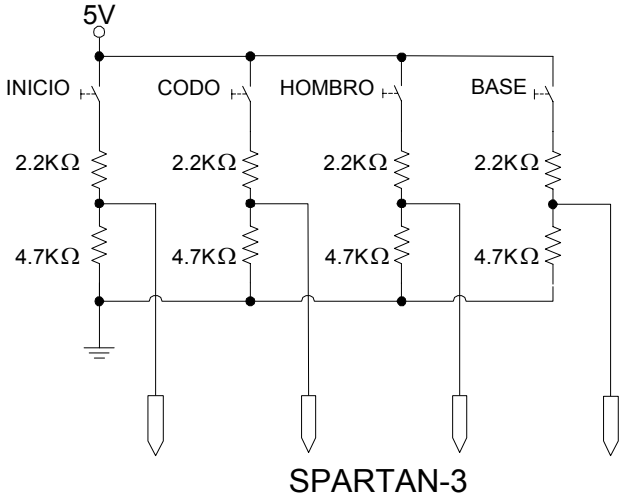


Figura 6.2.3 Diagrama de conexiones para los interruptores limit-switch y push button de “inicio”.

6.2.2 Adaptación de sensores en el tablero

El tablero construido en este proyecto se ilustra en la figura 6.2.4. El tablero es de madera y sólo la superficie de las casillas está forrada con corcho para tratar de que las pelotas no reboten al golpear con ella. Se utilizan pelotas de goma de tamaño aproximado de una pulgada y media de diámetro considerando su peso y la textura de ellas, para hacer las tiradas tanto del brazo robótico como de su oponente dentro del tablero, los orificios dentro del tablero cuentan con dos sensores, estos últimos sirven para mandar la información al control (“FPGA”) del estado en que se encuentran los orificios. (vacíos u ocupados).

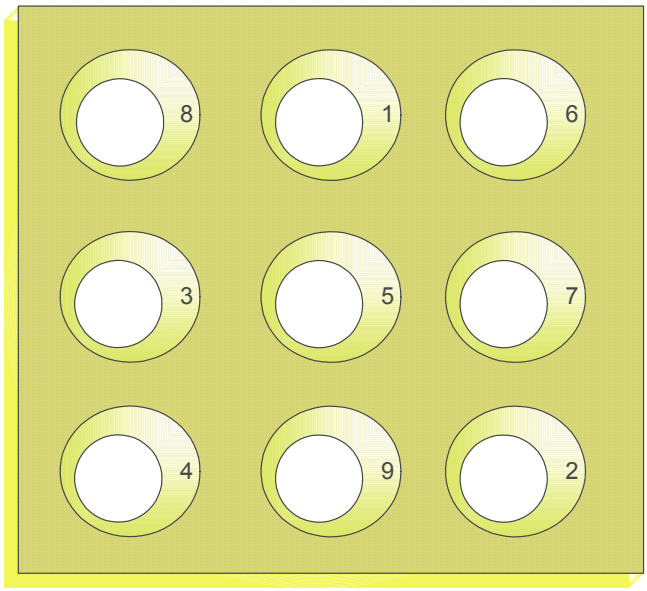


Figura 6.2.4 Tablero de juego hecho con madera y superficie de corcho.

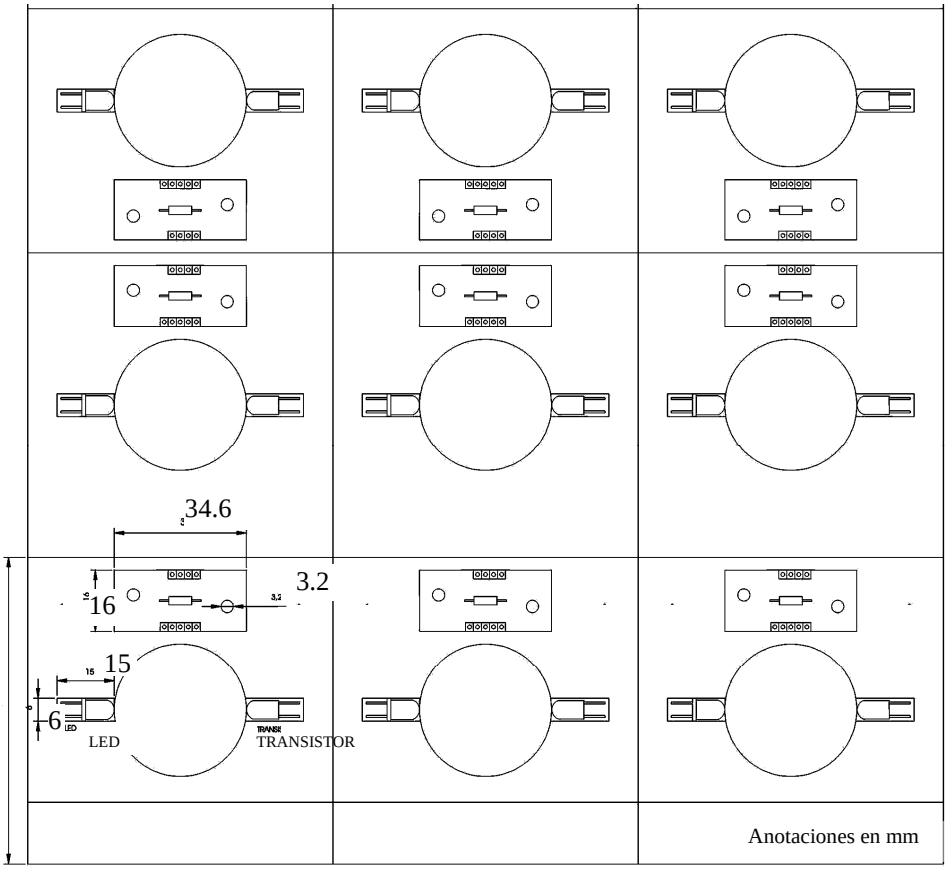


Figura 6.2.5 Diagrama esquemático interno de la disposición de los orificios e incorporación de los elementos sensores ópticos.

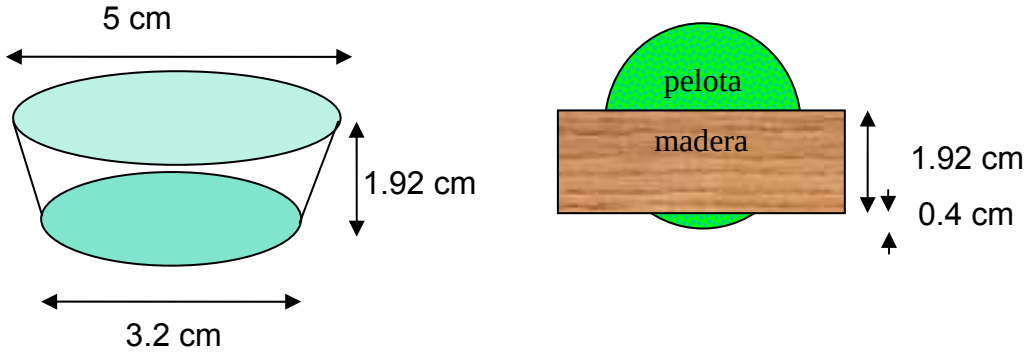


Figura 6.2.6 Dimensiones de los Orificios para cada posición del Tablero.

La información obtenida por el "FPGA", enviada por los sensores de cada uno de los orificios también sirve para identificar cuando le toca al brazo tirar. Con el mismo movimiento de colocación de la pelota debe de indicarle al brazo cuando ya tiró el contrincante para que él mismo inicie su siguiente rutina de movimiento. Para llevarlo a cabo, se hicieron pruebas con interruptores de contacto mecánicos (micro switch), pero como la pelota de goma no pesa suficiente no alcanzaba a presionar bien el interruptor aún poniéndole peso, entonces se pensó que se debía utilizar interruptores o sensores ópticos, de los cuales descartamos los sensores de intensidad luminosa porque con alguna sombra, incluso del mismo brazo, podría obstaculizar la operación. Por tanto, se seleccionaron sensores ópticos del tipo de luz infrarroja donde al interrumpir el haz de luz infrarrojo con la pelota se abre el circuito que conecta.

Para el proyecto se colocaron en cada orificio del tablero (9 orificios) por la parte de abajo un sensor óptico de luz infrarroja con su respectivo emisor y receptor, como se indica la figura 6.2.7; y se armaron sus respectivas interfases que conectan a la tarjeta Spartan-3.

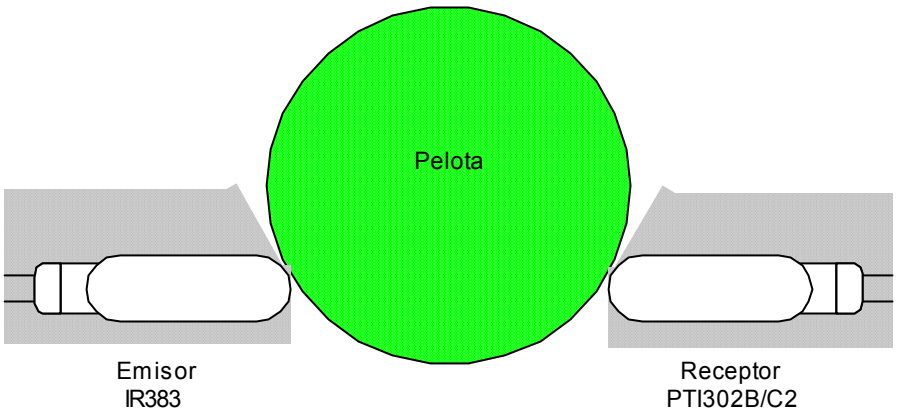


Figura 6.2.7 Ubicación de los sensores ópticos de cada orificio.

De las especificaciones de los fabricantes de sensores se consideró un circuito que no consumiera mucho voltaje y corriente, y que la distancia del orificio la pudiera alcanzar sin problemas. Entonces se seleccionó el circuito del emisor basado en el LED infrarrojo IR383, el cual tiene un voltaje de operación (V_F) de

1.2 VCD con una corriente (I_F) de 20 mA, de acuerdo a las especificaciones del fabricante.

Con estos valores se calcula la resistencia que limita la corriente al diodo, de acuerdo al circuito mostrado en la figura 6.2.8.

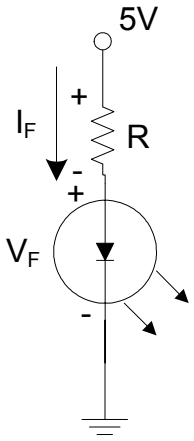


Figura 6.2.8 Circuito resultante para diodo emisor

Sumando los voltajes de nodo:

$$V_{CC} = V_R + V_F$$

En términos de sus elementos:

$$V_{CC} = RI_R + V_F$$

Despejando:

$$R = \frac{V_{CC} - V_F}{I_R}$$

Sustituyendo:

$$R = \frac{5 - 1.2}{20 * 10^{-3}}$$

Por lo que:

$$R = 220\Omega$$

Por otro lado, el circuito del receptor es el foto-transistor PPT1302/C2, por ser el receptor del emisor, por especificaciones del producto, cuyos valores de saturación son V_{CESAT}=0.4 V CD e I_{CSAT}=2 mA. Con estos valores, podemos calcular la resistencia de colector para la zona de saturación, de acuerdo al siguiente circuito de la figura 6.2.9.

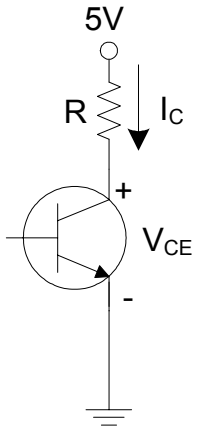


Figura 6.2.9 Circuito resultante en zona de saturación para el receptor

Sumando los voltajes de nodo:

$$V_{CC} = V_R + V_{CESAT}$$

En términos de sus elementos:

$$V_{CC} = RI_C + V_{CESAT}$$

Despejando:

$$R = \frac{V_{CC} - V_{CESAT}}{I_C}$$

Sustituyendo:

$$R = \frac{5 - 0.4}{2 * 10^{-3}}$$

Por lo que al sustituir valores tenemos que el valor comercial más cercano al resultado es:

$$R = 2.2K\Omega$$

Para la zona de corte, el transistor actúa como un circuito abierto, de tal manera que el voltaje de salida que será la señal de entrada a la tarjeta SPARTAN-3 lo tomamos del Voltaje en el colector, sin embargo, no podemos alimentar con +5 VCD a la tarjeta, puesto que sólo acepta señales de +3.3 VCD. Con esta información, el circuito para la región de corte queda de la misma manera como se consideró el divisor de voltaje para los interruptores de fin de carrera de los motores (Ver figura 6.2.2), por lo que el análisis y el valor de la resistencia Ro se obtiene de la misma forma y es:

$$Ro = 4.7K\Omega$$

De esta forma, se tiene al fototransistor trabajando como un switch electrónico. Mientras no haya pelota en la casilla el transistor está saturado y la señal de salida es un 0 lógico, cuando una pelota es colocada en la casilla, el haz de luz infrarroja se interrumpe y el transistor se corta, mandando un 1 lógico como señal de salida. El circuito completo incluyendo el dispositivo emisor y receptor para cada unos de los orificios se muestra en la figura 6.2.10

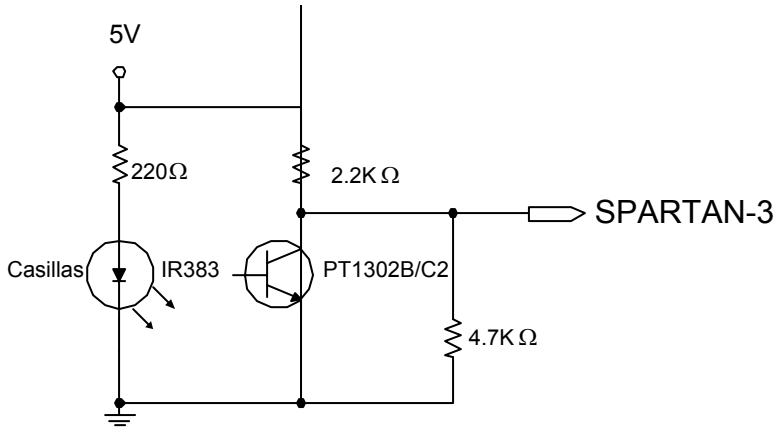


Figura 6.2.10 Circuito resultante para los sensores de cada orificio del tablero

6.3 Adaptación de tarjeta de potencia en el Brazo Robótico.

Para el diseño de la etapa que controla a los motores se requirió hacer pruebas con varios controladores o drivers de motores de CD y se llegó a la conclusión de que el driver L298 por la corriente que maneja, (3 A) es el adecuado para este proyecto.

Se consideró la alimentación hacia los motores con +9 VCD, porque con 5 V la velocidad es muy lenta y con +12 V giran muy rápido, entonces seleccionando una velocidad intermedia entre ese rango fue +9 VCD.

Entonces,

V_{motor} = 9 VCD

Y se midió físicamente la impedancia resistiva en las puntas de cada motor, resultando lo siguiente:

R_{motor1} = 32 Ω (base)
R_{motor2} = R_{motor3} = 14 Ω (hombro y codo)
R_{motor4} = 2 Ω (pinzas)

Considerando la resistencia para el peor escenario R_{motor2} =14Ω porque es de los motores que más se estarán moviendo, se descarta la consideración del motor de las pinzas (el peor escenario de consumo de corriente) porque no se usará mucho y además en ese motor no es importante la precisión. Por tanto, el cálculo para la selección del driver de los motores es el siguiente:

$$\mathrm{Im} = \frac{Vmotor}{Rmotor}$$

$$\mathrm{Im} = \frac{9}{14}$$

$$\mathrm{Im} = 642mA + 25\%(arranque)$$

$$\mathrm{Im} \approx 800mA$$

Se decidió utilizar para la etapa de potencia de los motores de los amplificadores integrados o drivers, y no con dispositivos semiconductores como transistores, SCR, o TRIAC por la facilidad de que ya son circuitos probados y nos ahorran tiempo y espacio en el diseño. Nuestra tarea consistió en seleccionar el driver más adecuado para el control de los motores, considerando además de la corriente, la velocidad de los motores, que se consideró fija para nuestro control; y con ello nos reduce el tipo de controlador de motor a los amplificadores de tipo lineal.

6.3.1 Driver o controlador de motores de CD

Se seleccionó el driver L298 para el manejo de los motores del brazo robótico por sus características del fabricante que maneja, y además porque el circuito integrado tiene la capacidad de manejar dos cargas inductivas a la vez como solenoides, motores de corriente directa o de pasos; así como de utilizar voltajes separados tanto para la carga (hasta 45 V CD) como para la lógica de control (5 V CD) con una corriente máxima de 2 A para la carga.

Cada driver o circuito integrado L298 controla dos motores a través de tres pines de entrada: Enable (Ena), giro a la derecha (C) y giro a la izquierda (D). En la tabla 6.3.1, se indica la lógica de operación del driver L298.

Enable	C	D	Giro
0	X	X	No (giro libre)
1	0	1	Izquierda
	1	0	Derecha
	C=D		Frenado

X = No importa

Tabla 6.3.1 Tabla de verdad del Driver L298

Para el proyecto se utilizaron dos drivers L298 para el control total de los cuatro motores del brazo.

Para el manejo de cargas inductivas es necesario colocar un Puente H de diodos de recuperación rápida y, con esto, proteger el circuito integrado contra las corrientes de retorno generadas por el motor.

El circuito que se armó para un driver se muestra en la figura 6.3.1, en él se muestra el circuito H de diodos en las salidas del driver que alimentan al motor. Los componentes eléctricos se armaron de acuerdo a las especificaciones del fabricante y los diodos rectificadores utilizados fueron los IN4937 de propósito general.

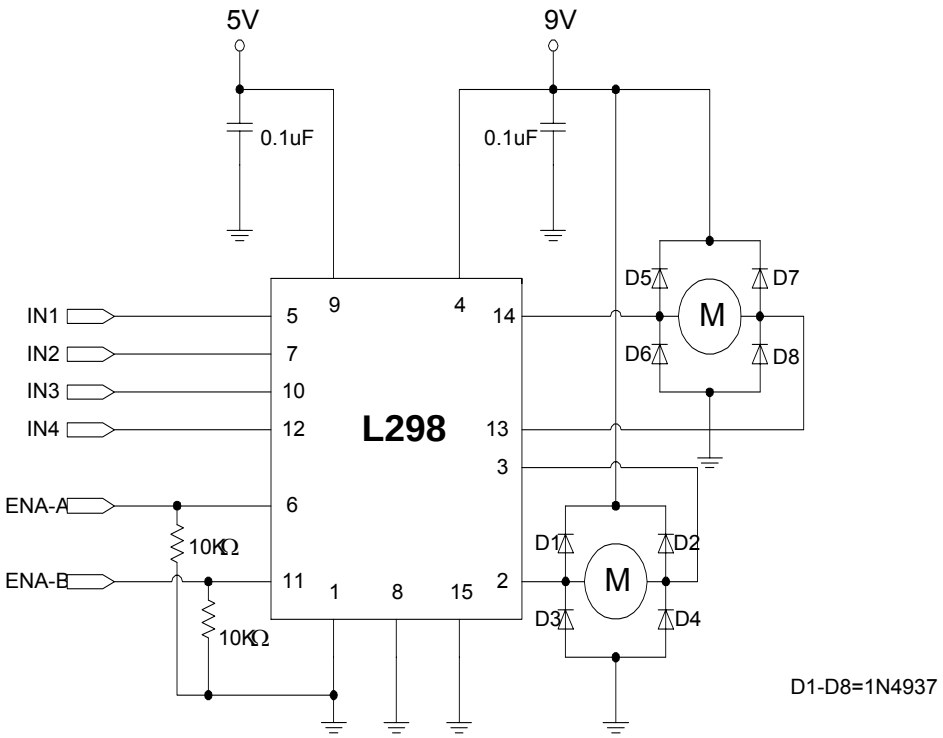


Figura 6.3.1 Diagrama del Driver L298 para control de dos motores.

6.3.2 Comparadores de voltaje

Las señales de control de entrada a los drivers L298 provienen del puerto de la tarjeta del FPGA de la tarjeta Spartan-3, con los que se controla, como indica la tabla 6.3.1, el encendido, frenado y sentido de giro de cada motor. El voltaje entregado por la tarjeta electrónica Spartan-3 es de +3.3 VCD y para el manejo de las señales lógicas del driver se necesitan +5 VCD. Aunque la lógica TTL maneja desde +2.4 VCD para reconocerlo como un 1 lógico, el ruido eléctrico generado por la fuente conmutada de alimentación así como el generado por el arranque y paro de los motores obliga a manejar en la entrada a los drivers los +5 VCD estrictamente.

Para hacer el cambio de niveles en la alimentación de cada señal de control al driver, se recurrió al comparador de voltaje LM339, por su amplio rango de operación de voltaje y encapsulado con varios comparadores, a los cuales se le introduce un voltaje de referencia en la terminal negativa a través de un potenciómetro para compararlo con los 3.3 VCD entregados por la tarjeta de Xilinx. De forma que el circuito resultante comparador de voltaje se muestra en la figura 6.3.2.

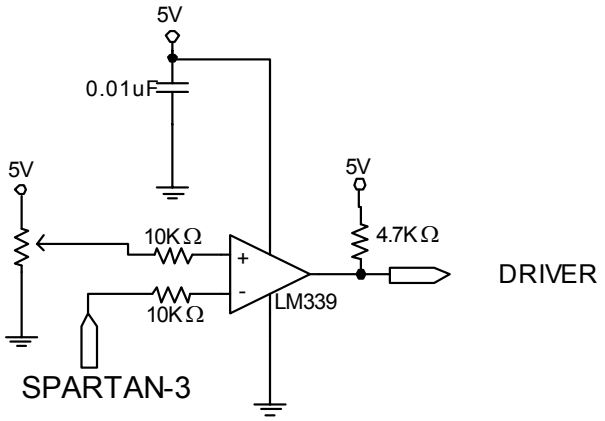


Figura 6.3.2 Comparadores de Voltaje para la alimentar el Driver L298

El voltaje de referencia se utiliza para que no haya cambios inesperados provocados por ruido. La idea de introducir un potenciómetro para poder variar el voltaje en un amplio rango, y no un voltaje fijo, es para poder tener el voltaje necesario para que el ruido eléctrico no afecte el funcionamiento del circuito.

Los valores de las resistencias colocadas en las entradas del comparador se seleccionaron altas (del rango Kilo Ω) para limitar la corriente que entra al CI LM339, la resistencia a la salida del comparador es necesaria para cerrar el circuito de salida, pues el comparador es de Colector abierto, y se seleccionó del rango de los KΩ de igual forma, para limitar la corriente a la entrada el driver L298. Se requirieron 12 comparadores, una para cada entrada a los 2 drivers L298. Ver figura 6.3.1.

Este bloque indicado en el figura 6.1, con los comparadores de voltaje en un principio no se estaba considerando, hasta que en las pruebas reales y funcionales de toda la operación de los cuatro motores se determinó que se requería porque se comportaban inestables los movimientos de los motores sin razón aparente más que el ruido provocado por el movimiento de los motores y sensores.

6.4 Adaptación del Starter Kit Spartan-3 al Brazo Robótico

La adaptación del brazo robótico a la tarjeta Spartan-3 físicamente se lleva a cabo con los bloques indicados en la figura 6.1, pero el control se realiza a través de programación de la misma tarjeta Spartan-3. A esto se refiere el bloque indicado como programa en la misma figura 6.1.

El programa consta de dos partes principales, el correspondiente al control de los movimientos de los motores y el otro a la lógica de operación del juego de Gato, ambos algoritmos se realizaron en lenguaje VHDL de la tarjeta Xilinx. Como primera parte se analiza el programa correspondiente al Control de los movimientos de los motores de CD del brazo robótico. Esta parte es la que controla cada uno de los bloques mostrados en la figura 6.1.

6.4.1 Programa para la lógica de control de los motores

De una forma más descriptiva, se presenta un diagrama de funcionamiento del programa que se configuró en el FPGA de la tarjeta Spartan-3 para control de los motores de CD, sin indicar los procedimientos a seguir en el método de programación. En la figura 6.4.1 se indica con bloques cada movimiento requerido para llevar a cabo el juego de Gato, pero no es todavía la lógica del juego sino la plataforma de movimientos en la que el juego se va a desarrollar. Se puede ver en la misma figura 6.4.1 que los bloques principales del programa son los correspondientes a cada ruta.

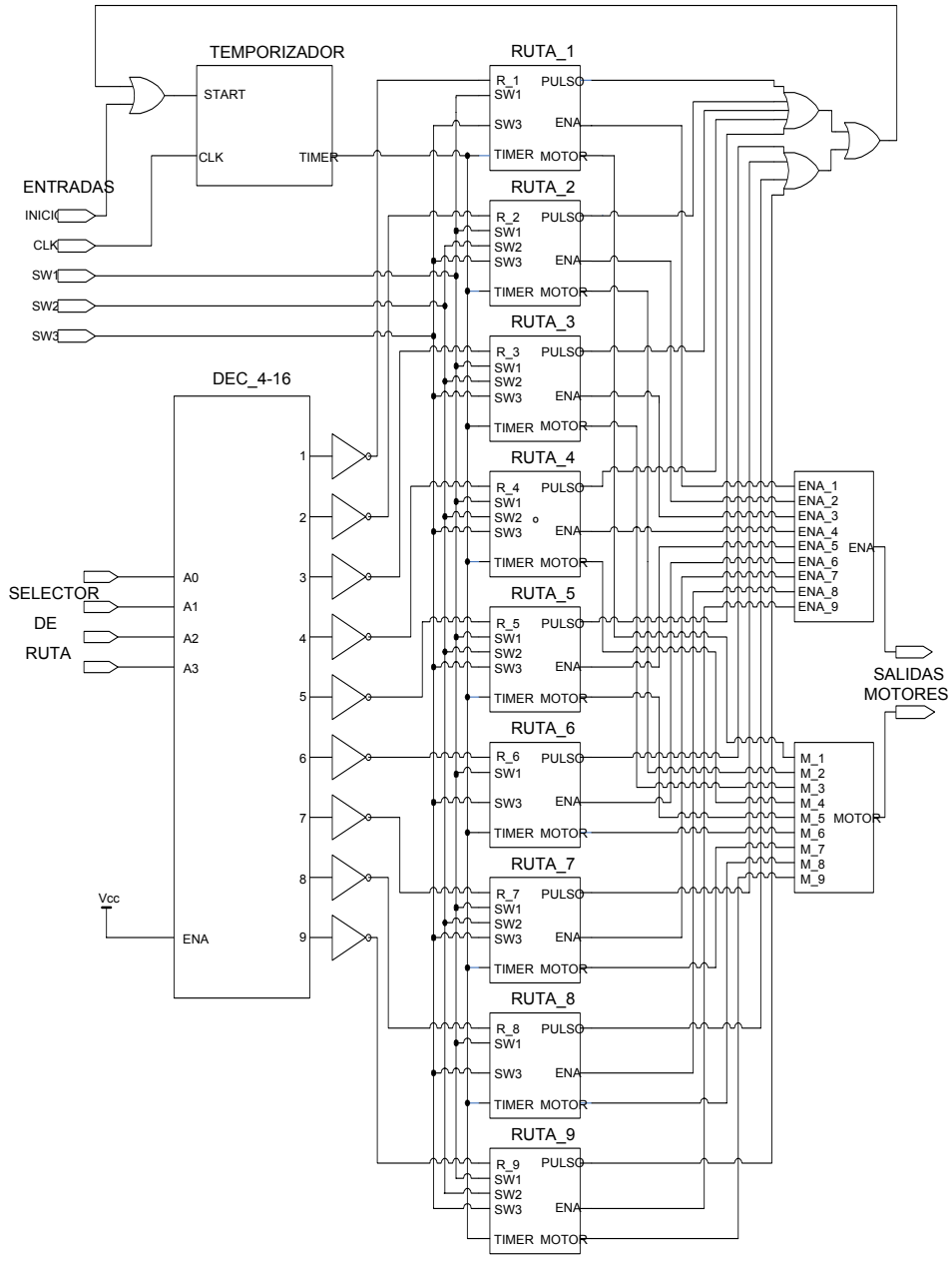


Figura 6.4.1 Diagrama de las rutas de operación de los movimientos del Brazo Robótico de acuerdo al Programa del FPGA.

La descripción de una ruta para el programa se refiere a la rutina de movimientos que tiene que hacer el brazo robótico para colocarse en cada posición del tablero, de acuerdo a la numeración mostrada en la figura 6.2.4. Para cada posición del tablero le corresponde una ruta indicada en la figura 6.4.1. La selección de la ruta a seguir se hace a través de un decodificador de 4-16 líneas. El programa selecciona la siguiente jugada y deshabilita la señal de Reset de la ruta correspondiente dejando las demás entradas de Reset en alto.

La colocación de la pelota en la casilla correspondiente, por parte del brazo robótico, se logra energizando los motores por un tiempo determinado; es decir, cada motor se energiza con la polaridad específica para avanzar o retroceder, según sea necesario. La programación de las rutas se realizó a través de diagramas de estados, considerando cada estado como un movimiento del motor. Como el reloj con el que trabaja la tarjeta Spartan-3 está a 50 MHz, fue necesario el desarrollo de un reloj más lento para la operación del proyecto. Esto se puede ver en la figura 6.4.1 como el bloque temporizador al cual la entrada de "CLK" corresponde a la frecuencia de reloj proporcionada por la tarjeta Spartan-3. El cambio entre estados se hace tomando el reloj como condicionante. Este reloj se modificó para dar pulsos cada 0.1 segundos de tal manera que el cambio entre estados y, por lo tanto, el tiempo que permanecen energizados los motores se puede hacer con una precisión de hasta una décima de segundo. Se muestra en la figura 6.4.1 el intervalo de tiempo del proyecto como el correspondiente a la señal "Timer".

El problema principal que se genera al trabajar los motores por tiempo es la inercia. Cada vez que se detiene un motor, éste presenta cierto avance después de que la energía fue interrumpida, tal vez sólo unos cuantos milímetros, sin embargo, si se suma la inercia de cada movimiento, al final se tendría una ruta totalmente diferente de la que se programó.

Para eliminar la mayor cantidad de errores debida a la inercia de los motores, se programó el avance de varios motores a la vez y aprovechando la capacidad de freno con que cuenta el driver L298 y los sensores de fin de carrera colocados en cada motor.

Sin embargo, la inercia no se puede eliminar por completo, por lo que se colocaron los interruptores de final de carrera en el codo, hombro y base para determinar la posición de inicio. Con esto reducimos al máximo los errores debidos a la inercia de los motores, al partir siempre de la misma posición de inicio, y aseguramos que el brazo se auto ajuste cada vez que coloca una pelota en alguna casilla.

Los interruptores de final de carrera no están regulados por el tiempo del reloj y no todos los movimientos de los miembros del brazo se implementaron con interruptores pues, cada ruta debería de contar con un número de interruptores, lo cual hace inviable la colocación de interruptores de final de carrera para cada ruta. De esta forma como se dejó la configuración es que los interruptores de carrera detienen el movimiento para ciertos movimientos estratégicos, como posición de inicio, y con ellos se reinicia el contador de tiempo para los demás movimientos los cuales son controlados por tiempo, con intervalos de tiempo de 0.1 segundos, de acuerdo a la programación previa para cada movimiento.

Cada ruta tiene como salida las señales de habilitación de "Enable", "C" y "D" de cada motor. Sin embargo para poder juntar todas estas señales en una

sola, se implementaron compuertas OR que aceptan señales de entrada tipo buffer de 4 entradas para la señal de “Enable” y 8 entradas para las señales “C” y “D” de los motores, de tal manera que sólo tengamos al final 12 salidas que van conectadas a los drivers de los motores, ver figura 6.4.1.

6.4.2 Programa para la lógica del juego de Gato

De forma operativa y a nivel de descripción se muestra en la figura 6.4.2 un diagrama a bloques del funcionamiento del algoritmo del juego de Gato. Hay que considerar que cada uno de los bloques que se presentan están desarrollados con el programa de configuración del FPGA o en lenguaje VHDL de Xilinx, igual que el control de movimientos de motores sólo que este algoritmo es el que realmente controla el programa de rutas de movimiento de los motores del brazo robótico. El código VHDL de construcción se omite por cuestiones de espacio.

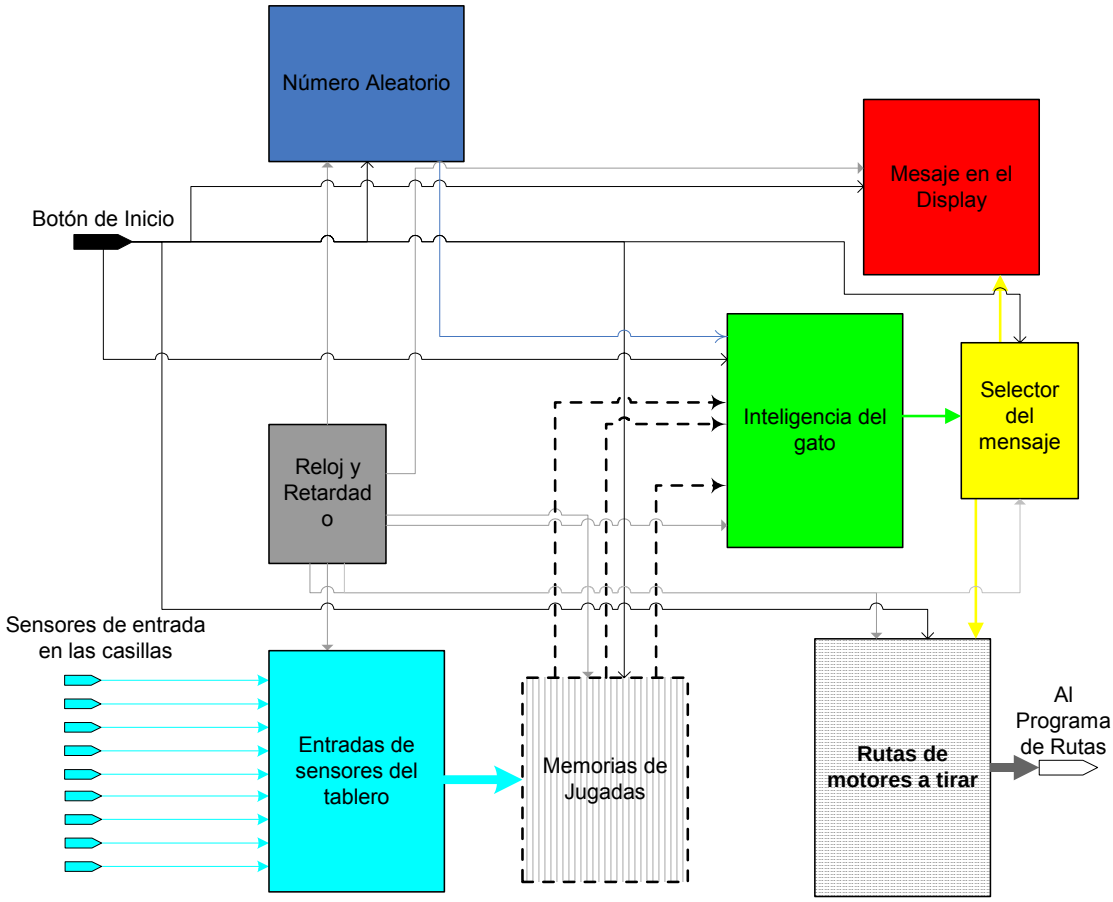


Figura 6.4.2 Diagrama a bloques de la lógica del Juego de Gato

Reloj y retardo de tiempo

Este bloque se refiere a la entrada para la sincronía de la programación. La fuente de reloj utilizado es el mismo que trae la tarjeta Spartan-3, de 50MHz, sólo para el caso de algunos bloques se entregan retardos de tiempo para

sincronizar con otros módulos del mismo algoritmo. Pero en general se obtiene de múltiple retardos de o divisores de frecuencia de éste reloj fuente.

Número aleatorio

Este bloque se refiere a las primeras dos tiradas por parte del brazo robótico. Una forma de reducir el número de líneas de programación de jugadas para el brazo robótico fue el definir desde un inicio del proyecto que éste siempre va a ser el que tire la primera jugada, de tal forma y de acuerdo a las estadísticas que enseguida se comentan, el brazo siempre va a Ganar o Empatar, nunca va a Perder, porque él siempre va a realizar la primera jugada, en forma aleatoria, en las casillas de las esquinas o bien en la del centro como primera tirada.

Basándose en el algoritmo del cuadro mágico es cómo se determinaron las posibles soluciones para el juego de Gato, de tal forma que con la combinación de 3 números del tablero sumados entre si resulta 15, estos números forman una línea (vertical, horizontal o inclinada), no importando la posición de los dígitos para estas combinaciones. La numeración dentro del tablero es como se muestra en la figura 6.4.3.

8	1	6
3	5	7
4	9	2

Figura 6.4.3 Asignación numérica a las casillas u orificios del tablero

Las soluciones para este algoritmo son 8 posibles líneas:

Combinación de Casillas
159
258
357
456
816
276
294
834

Tabla 6.4.1 Líneas o combinación de dígitos para solucionar el cubo mágico

Existen tres tipos de casillas, las pares (2, 4, 6 y 8), que están ubicadas en las esquinas del tablero; el cinco (5), que está en la casilla centro, y las impares (1, 3, 7 y 9) que están ubicadas en los cuatro puntos cardinales y en medio de dos casillas pares.

Como se puede ver en la tabla 6.4.1, la casilla 5 está presente en cuatro las ocho posibles soluciones ó líneas (líneas 1, 2, 3 y 4). Cualquier número par tiene 3 posibilidades o aparece en 3 líneas distintas cada número; por ejemplo la casilla 2 está en las líneas 2, 6 y 7. Y cualquier casilla impar (excepto el 5) tiene dos posibilidades o aparece cada uno en dos líneas; por ejemplo la casilla 3 está en las líneas 3 y 8.

De esta forma, para que tenga mayores posibilidades de ganar el brazo robótico, su primera tirada tiene que ser en la casilla 5, con 4 posibilidades de ganar, o en una de las casillas pares, con 3 posibilidades de ganar; y entonces obligar al oponente a tirar en las casillas impares con dos posibilidades de ganar.

El algoritmo programado es el siguiente.

Se asignaron 3 variables para almacenar los dígitos de las casillas ocupadas en el tablero durante el juego.

Variable	Descripción
RAM1	Almacena las tiradas del brazo
RAM2	Almacena las tiradas del oponente
RAM3	Almacena las casillas ocupadas en el tablero
numeroDeCasillasOcupadas	Contador, número total de casillas ocupadas

Tabla 6.4 Variables o Memorias guardas para el Juego de Gato

La lógica del algoritmo del programa ejecuta rutinas del tipo Case para determinar el valor que tiene la variable numeroDeCasillasOcupadas en un momento dado, y se va incrementando cada vez que se coloca una ficha sobre cualquier casilla del tablero. La lógica de programación en forma descriptiva sería de la forma siguiente:

```
Si numeroDeCasillaOcupadas = 0,  entonces (se esta iniciando la partida)
                                 La tirada es aleatoria en la casilla 5 o en cualquier casilla par

Si numeroDeCasillaOcupadas = 2 entonces
                                 Si brazo tiro en casilla 5, que tire aleatorio en cualquier  casilla
                                        par Desocupada
                                 Si oponente tiro en casilla 5, que tire aleatorio en cualquier
                                        casilla par Desocupada
                                 Si ninguno tiro en casilla 5, el brazo tiene que tirar una casilla
                                        par, de tal modo que obligue al oponente a tirar en
                                        casilla impar.

Si numeroDeCasillaOcupadas = 4 entonces
                                 Si el brazo tiene posibilidades de completar cualquier
                                        solución de la tabla 6.4.1 Que tire
                                 sino
                                        Si el oponente tiene posibilidades de completar
                                               cualquier solución de la figura 6.4.3 Que tape
                                 sino
                                        Que tire en la casilla par desocupada

Si numeroDeCasillaOcupadas = 6 entonces
                                 Si el brazo tiene posibilidades de completar cualquier solución
                                        de la tabla 6.4.1 Que tire
```

sino
Si el oponente tiene posibilidades de completar
cualquier solución de la tabla 6.4.1 Que tape
sino
Que tire aleatorio

Si numeroDeCasillaOcupadas = 8 entonces
Que tire en la casilla desocupada.

Si numeroDeCasillaOcupadas = 1, 3, 5, 7,9 entonces
Espero porque le toca tirar al oponente…

Esta es la lógica que el brazo utiliza para saber las jugadas que va a realizar. Para el caso de la primera jugada hace uso del módulo de Número Aleatorio, el cual funciona de la siguiente manera:

El módulo Número Aleatorio es un contador configurado en el programa VHDL que está siempre operando, cuenta del 0 al 15 a la frecuencia del reloj de 50MHz, en el instante en que se presiona el Botón de Inicio, Ver figura 6.4.2, el número en que se encuentra el contador lo registra el programa, lo codifica de acuerdo a la siguiente tabla 6.4.2 y le asigna el valor a una variable con esta primera jugada.

Numero decimal aleatorio	Número en Binario Aleatorio	Número casilla para jugar
0	0000	2
1	0001	4
2	0010	5
3	0011	6
4	0100	8
5	0101	2
6	0110	4
7	0111	5
8	1000	6
9	1001	8
10	1010	8
11	1011	8
12	1100	8
13	1101	8
14	1110	8
15	1111	8

Tabla 6.4.2 Codificación del contador para primera jugada del Brazo Robótico

La tabla está realizada ex profesamente para que, aunque sea seleccionada en modo aleatorio la casilla, el brazo sólo seleccione casillas con dígito par (2, 4, 6 u 8) o la casilla de en medio (5) y, además, para que tenga mayor probabilidad de jugar la casilla 8 con la primera jugada que en las otras casillas.

Entrada de sensores del tablero

Este módulo se refiere a un bloque decodificador donde entran las señales de las nueve posibles casillas pero sólo se van a ir procesando de una en una, porque es la entrada para la jugada de cada jugador. Este módulo tiene nueve

distintos bloques de seguimiento para cada casilla el cual lo que hace es asegurarnos que no se va a atenuar la señal de entrada, así como un bloque para evitar posible ruido al colocar las fichas en las casillas. Este módulo sólo es de paso y la señal que detecte la entrega al módulo de memoria para registrar la tirada jugada y procesar la siguiente jugada para el brazo.

Memorias de jugadas

Este módulo lo que hace es almacenar en un circuito configurado a través de lenguaje VHDL, no en los dispositivos de memoria propias de la tarjeta Spartan-3, cada una de las señales de entrada tanto de los sensores en las casillas, guardadas como variables, como se indica en la tabla 6.4., así como las señales de Botón de Inicio y grabación de número aleatorio.

Con los valores de las variables actualizadas ya sea en forma secuencial o por evento, es que se logra la lógica del juego. El nombre de las variables es más o menos identificable por sí sola.

Se evitaron la grabación de algunas memorias conociendo la lógica del juego, por ejemplo, si al brazo robótico siempre será el primero que tire, entonces, a él le corresponden todas las jugadas nones, y las jugadas pares son las del oponente y ésas es necesario contabilizar. De esta forma se redujeron las variables a grabar.

Mensaje en el display

Para cada uno de los eventos principales dentro de la programación se despliega en el display de la propia tarjeta Spartan-3 un mensaje correspondiente al evento que prosigue o que se está ejecutando, esto se logra programando un decodificador para cada uno de los cuatro display de 7 segmentos localizado en la tarjeta, como se puede ver en la figura 2.5.2. Las señales que se presentan en ellos son enviadas por el bloque de Inteligencia del Gato al bloque de Selector de Mensaje y éste de acuerdo al valor y número de variables es que determina qué mensaje se desplegará sobre el display.

Los mensajes en el display se configuraron como apoyo para realizar las pruebas de movimiento del robot, pues muchos de los bloques correspondientes al algoritmo del juego del Gato se generaron sin estar conectado el brazo robótico y la referencia era el display, pues únicamente se requería confirmar si la lógica del juego era correcta de acuerdo a las variables conocidas que se debían activar, tales como el botón de inicio, las variables de las casillas del tablero y las de los sensores de límite de carrera en cada motor del brazo. Al final del proyecto, cuando se integraron todos los bloques junto con el brazo robótico y el tablero ya no eran tan necesarios los mensajes del display porque se podían ver los movimientos del brazo, pero se decidió dejar los mensajes del display como referencia y además nos sirven como confirmación de los movimientos que va a realizar el brazo.

Los mensajes identificables en el display son los que se muestran en la tabla 6.4.4.

Mensaje en el display	Descripción del mensaje
VOyn	Cuando le toca jugar al gato aparece este mensaje.
VAS	Cuando le corresponde jugar al contrincante.
ErrOr	Cuando se presenta un error no identificable por las reglas del juego.
n	n indica el número del 1 al 9 de la casilla donde va a tirar el brazo robótico de acuerdo a su lógica y mientras se desplaza a la casilla correspondiente.
gAnE	Cuando el brazo termina su última tirada, no hay posibilidad de que el contrincante tire más y además completó una línea.
EmPAtE	Cuando ya no hay más posibilidades de tirar y ninguno de los dos jugadores completó una línea.
LIStO	Cuando se inicia el juego y se coloca en la posición de Inicio el brazo, también cuando está en modo de calibración y termina. Despliega este mensaje.
PErdI	En caso que se llegara a presentar tal situación, aunque por el algoritmo configurado nunca aparecerá, pero por concepto en la lógica del juego está como opción.
Sin Desplegar	Cuando se conecta de inicio y no se ha presionado el botón de inicio se queda en este estado, sin mostrar ningún mensaje.

Tabla 6.4.4 Mensajes sobre el Display de la Tarjeta Spartan-3

Selector del mensaje

Este bloque en el algoritmo del juego de Gato es útil como una interfase entre la Inteligencia y los bloques de salida del algoritmo. Como interfase se utiliza porque está de acuerdo a una codificación que se tiene y también de acuerdo al valor de las variables grabadas en memoria y es el que determina el número de mensaje que va a desplegar. Este bloque nos sirve para sincronizar el mensaje del display con los movimientos del robot, pues se utilizó de cierta forma el mismo codificador de salidas para mandar llamar en forma automática dos salidas del bloque de inteligencias, uno para que presente el mensaje en el display y otro para que se mande llamar la ruta correspondiente a mover en el programa de Rutas de Movimientos. Hay ciertos valores en que con el desplegado del display se determina qué movimiento es el que ejecuta el programa de Rutas.

Inteligencia del Gato

Este módulo corresponde al módulo principal de la lógica de programación del juego de Gato. En este algoritmo es donde se decide qué debe hacer con cada una de las señales de entrada y cómo debe tirar para ganar, así como es el que le indica al programa de Rutas de Movimientos, mostrado en la figura 6.1, correspondiente al programa mueve los motores del brazo de acuerdo al programa mostrado en la figura 6.1

En sí este módulo está diseñado con sentencias del tipo Case donde se introdujeron cada una de las posibilidades que puede llevar a cabo el brazo de acuerdo a cada uno de los valores de las variables asignadas en los módulos anteriores. En forma general este módulo controla y organiza todas las variables de entrada para codificarlas y asignar una tarea por cada estado de las variables. El problema mayor que se presentó en este módulo fue la asignación de algunos comandos que se encontraron en el algoritmo para la solución del cuadro mágico, no eran sintetizables, por lo que se tuvo que introducir en forma manual y cada una de las opciones para el Case que había que analizar, o sea, se tuvo que generar y analizar por completo el algoritmo de la lógica de juego de gato por sentencias del lenguaje VHDL que fueran sintetizables por la tarjeta Xilinx.

El algoritmo del proyecto, consta de alrededor de 1000 sentencias introducidas en lenguaje VHDL y programadas principalmente a través de Diagramas de Estado.

Conclusiones

El desarrollo de este proyecto nos permitió, principalmente, conocer el uso y aplicaciones de los dispositivos FPGA así como del lenguaje de programación VHDL. Nos percatamos de la gran capacidad de configuración y amplia utilidad y, aunque se trata de una tecnología relativamente nueva, su manejo y aprendizaje es realmente sencillo. La complicación estribó un poco en conocer la operación de la plataforma de desarrollo, la tarjeta Spartan-3 de la empresa Xilinx y la operación del software del lenguaje VHDL pues gran parte de tiempo del proyecto lo dedicamos a la revisión de la forma de trabajar, principalmente del software más que a la tarjeta Spartan-3. Pero es importante resaltar que gracias a la tecnología de dicha tarjeta se redujo en gran medida el hardware que se hubiera utilizado de haberse hecho el proyecto con algún otro tipo de lógica programable.

Es importante mencionar que el éxito de este proyecto, principalmente se debió a la experiencia y conocimiento en lo que a definiciones y principios básicos para la programación por rutinas en cualquier lenguaje de programación se refiere, por parte de los que participamos en él, pues ayudó bastante para poder definir y entender desde el inicio qué era necesario desarrollar para que fuera realmente funcional, esto muy independientemente de la revisión que realiza el mismo software VHDL de Xilinx respecto a la compilación, “reindexación” y sintetización del programa. Un problema encontrado en la programación fue que varias funciones conocidas para lenguaje VHDL no eran sintetizables por el software de Xilinx, esto significa que no son comandos reconocidos por el hardware, aunque por programación sí era aceptado. Al querer convertirse a lenguaje HDL no se podía y nos afectó porque algunas ocasiones hubo que reprogramar utilizando otro comando y sustituir el algoritmo que no funcionó. De esta forma, se generaron rutinas que son utilizadas para varios bloques dentro del algoritmo de programación, tanto del Juego de Gato como del control de movimientos de los motores de CD.

Fue complicado el hecho de usar los motores de corriente directa que venían incluidos en el Brazo Robótico. El juego no sólo requiere de cierta precisión y exactitud de movimientos sino además de repetitibilidad y con este tipo de motores (motores de CD de imán permanente) se convirtió un problema controlarlos. Para corregir el problema se incluyeron los interruptores de final de carrera, pues al determinar con precisión la posición de inicio del brazo, se redujeron notablemente los errores de posicionamiento de las pelotas en las cavidades lo cual nos permitió programar los movimientos de los motores en base a tiempo. Sin embargo, el brazo tardaba mucho en completar una partida ya que los motores de cada articulación se movían en secuencia. Para reducir ese tiempo se desarrollo el algoritmo para que operen varios motores a la vez y aún así presenta algo de lentitud al realizar los movimientos pero ésta ya es provocada por el juego de engranes que vienen incluidos en cada motor del Brazo.

El hecho de haber aceptado trabajar con los motores de corriente directa incluidos en el brazo y no cambiarlos por otros, ya fueran de pasos o servos, nos obligó a aplicar ingeniería para resolver el problema: Encontrar la solución económicamente más adecuada. Haber cambiado los motores, hubiera significado, tal vez, cambiar el brazo y ese no era el objetivo principal de la tesis. Tuvimos que encontrarle solución a todos los problemas que se nos presentaron, algunos de los cuales no predijimos, como el ruido eléctrico o el hecho de que algunas instrucciones del lenguaje VHDL no fueran sintetizables, pero de igual manera hicimos uso de las herramientas estudiadas en nuestras materias de carrera para salir adelante y no alterar la construcción del brazo.

Para la elaboración de la plataforma, diseño del tablero y lugares donde alojar toda la electrónica para controlar el Brazo, fue necesario invertir algunas cantidades de tiempo a fin de determinar cuales eran los mejores materiales, dimensiones y el diseño para mostrar todo el Brazo como una unidad y sin cables externos, este objetivo fue pensado mas que nada por mejorar la presentación, además de buscar la mejor selección e integración de algunos sensores sobre el mismo tablero y que estuvieran dispuestos en forma intrínseca. La única parte que no se integró al diseño fue el despachador de las pelotas por falta de tiempo, pues el proyecto desde un principio se definió por cierto lapso por convención, y como no afecta el objetivo principal se decidió dejarlo así.

Algunas personas ajenas al proyecto nos hicieron el comentario que es demasiado lento y que se podía agilizar utilizando otro tipo de algoritmo para solucionar el juego de gato, pero el proyecto siempre estuvo definido desde el comienzo para utilizar lógica de diseño a través de la FPGA, por lo que la solución está limitada, por una parte, al lenguaje de programación VHDL y, por otro lado, a la habilidad del programador para administrar la lógica de control.

Podemos agregar, que el FPGA utilizado para este prototipo es exactamente el mismo PLD que el utilizado en la producción de un producto final, como puede ser, un ruteador de redes, un modem, un reproductor de DVD’s, o un sistema de navegación para un automóvil. Los costos de desarrollo se reducen considerablemente y el diseño final es terminado mucho más rápido que con el uso de dispositivos de lógica predeterminada.

En general se terminó el proyecto con éxito, porque se cubrió el objetivo especificado desde el inicio. Así, se puede resumir como resultado final que el control del Brazo funciona correctamente y de forma autosuficiente al encender y presionar el botón de inicio y, principalmente, el proyecto se quedará en el laboratorio de DLP de la Facultad de Ingeniería, como un prototipo de lo que se puede realizar con el uso de la lógica por medio de un FPGA y, como un valor agregado, puede servir para despertar la curiosidad en los alumnos, lo cual seguramente activará la sinergia para el desarrollo de proyectos más ambiciosos en la aplicación de los FPGA’s a niveles industriales y de desarrollo.

Bibliografía

- Sánchez, Antonio. *Principios básicos de la Robótica.*. Ed. DISA. UPV. (s.l)
- Angulo, José Ma., Sánchez, José N. *Guía Fácil de Robótica.*, Ed. Paraninfo. (s.l)
- Groover, Mikell P.; Weiss, Mitchell; Nagel, Roger N.; Odrey, Nicholas G. *Robótica Industrial: Tecnología Programación y Aplicaciones*. Ed. Mc. Graw Hill. México D.F. 1990
- Angulo, José Ma.; Avilés, Rafael. *Curso de Robótica*. Ed. Paraninfo
- Muhammad, H. Rashid. *Power electronics, circuits, devices and applications*, Ed. Prentice Hall. Third Edition, cop 2004.
- Valeriano Assem, Jorge; Chávez Rodríguez, Norma Elva. *Lenguaje de Descripción de Hardware VHDL*, Ed. Facultad de Ingeniería UNAM. México, D.F., 2002.
- *Spartan-3 Started Kit Borrad User Guide*, Ed. Xilinx., U.S.A., jun 2004
- Perry, Douglas L. *VHDL: programming by Example,* Ed. McGraw-Hill, Fourd Edition, U.S.A., 2002.
- *Xilinx Desing Reuse Methodology for ASIC and FPGA Designers,* Ed. Xilinx., U.S.A., 2001.
- *ModelSim Command Reference v5.5e*, Ed. Xilinx, U.S.A., 2001.
- *ModelSim User`s Manual v5.5e*, Ed. Modelsim, U.S.A., 2001.
- *Xilinx HDL coding hints*, Ed. Xilinx. U.S.A., 2001
- Ashenden, Peter J. *The VHDL CookBook*, Ed. Dept. Computer Science University of Adelaide South Australia, first edition. Australia, Julio 1990
- Synario Desing Automation, Division of Data I/O. *VHDL Reference Manual*. Ed. Synario, U.S.A., 1997
- *1076 IEEE Standard VHDL Language reference Manual*, IEEE., New York, U.S.A., may 2002
- Pedroni, Volnei A. *Circuit Design with VHDL*, Ed. MIT Press. U.S.A., 2004
- Parnell, Karen; Mehta, Nick. *Programmable Logic Desing Quick Start Hand Book*, Ed. Xilinx. U.S.A., 2001
- Cooper, William D.; Helfrick, Albert D., *Instrumentación Electrónica Moderna y Técnicas de Medición,* Ed. Prentice Hall, México, 1991
- *Xilinx Documentation&Literature*, [U.S.A.] : < http://www.xilinx.com > [Consulta: junio y Julio 2005]
- *Productos PLC,* [U.S.A.]: < http://www.efalcom.com > [Consulta: junio 2005]
- *Manuales de Contrucción Paso a Paso, [*Argentina]: Club de Robotica - Mecatrónica (UAM). < http://roboticajoven.mendoza.edu.ar > [Consulta: junio 2005]