Universidad Autónoma de Querétaro
Facultad de Ingeniería

Maestría en Instrumentación y Control Automático

"Diseño e Implementación de Teclado Industrial Aplicado a una Máquina de Inyección de Plástico"

TESIS

Que como parte de los requisitos para obtener el grado de
Maestro en Ciencias

Línea terminal en Mecatrónica

Presenta:
Rafael Santos Cruz

Dirigido por:
Dr. Roque Alfredo Osornio Rios

Querétaro, Agosto de 2008

Universidad Autónoma de Querétaro
Facultad de Ingeniería
Maestría en Instrumentación y Control Automático

"Diseño e Implementación de Teclado Industrial Aplicado a una Máquina de Inyección de Plástico"

TESIS

Que como parte de los requisitos para obtener el grado de
Maestro en Ciencias
Línea terminal Mecatrónica

Presenta:
Rafael Santos Cruz

Dirigido por:
Dr. Roque Alfredo Osornio Rios

SINODALES

Dr. Roque Alfredo Osornio Rios
Presidente
Firma

Dr. René de Jesús Romero Troncoso
Secretario
Firma

M. C. José Agustín Bravo Curiel
Vocal
Firma

M. C. Luis Morales Velázquez
Suplente
Firma

M. I. Jesús Rooney Rivera Guillén
Suplente
Firma

Dr. Gilberto Herrera Ruiz
Director de la Facultad de Ingeniería

Dr. Luis Gerardo Hernández Sandoval
Director de Investigación y Posgrado

Centro Universitario
Querétaro, Qro.
Agosto de 2008
México

RESUMEN

En el presente trabajo se muestra el diseño e implementación del protocolo de comunicación para un teclado industrial no matricial, mediante el diseño en hardware. Debido a la gran demanda de equipo automatizado, las MPYMEs (Micro, Pequeñas y Medianas Empresas) en la región realizan fuertes gastos en adquirir maquinaria o repuestos del extranjero. Por lo anterior, en esta investigación se propone un módulo digital de comunicación reprogramable y a bajo costo para aplicaciones en máquinas de inyección de plástico. El fundamento teórico del desarrollo se basa en el protocolo de comunicación AT. Para llevar a cabo este protocolo se implementaron módulos digitales de escaneo de teclas, control, recepción y transmisión. Los bloques antes mencionados se sintetizan en FPGA (*Field Programmable Gate Array,* Arreglo de Compuertas Programadas en Campo), utilizando lenguaje descriptivo de hardware. La ventaja que ofrece el prototipo, en relación con la mayor parte de los teclados comerciales, es la programación de una estructura digital que permite la interacción simultánea con otro teclado. La implementación del prototipo se realizó utilizando una tarjeta FPGA Spartan 3, una interfaz, y un teclado propio. Además, se probó la funcionalidad del teclado sobre una máquina de inyección.

(Palabras clave: Protocolo de Comunicación, FPGA, AT, VHDL, teclado)

SUMMARY

This work shows the design and implementation of a communication protocol for a non-array industrial keyboard through its design in hardware. Due to the high demand of automated equipment, local MPYME's (Micro Small and Medium companies) invest great amounts of money in acquiring machinery and devices from other countries. As consequence of the above mentioned, this investigation proposes a low cost digital module of programmable communication to be installed in an injection moulding machine. The theoretical background of the project is based on the AT communication protocol. In order to carry out this protocol, it was necessary to implement digital modules of key scanning, control, reception and transmission. The already mentioned blocks are synthesized in a FPGA (*Field Programmable Gate Array*), using descriptive language of hardware. The advantage that the prototype offers, as compare to most of the commercially existing keyboards, is the programming of a digital structure that allows simultaneous interaction with another keyboard. The implementation of this prototype was made using a Spartan 3 FPGA card, an interface and the keyboard by itself. Additionally, the functionality of the keyboard was also proven in an injection moulding machine.

(Keywords: Communication Protocol, FPGA, AT, VHDL, Keyboard).

A dios.

A mi mamá, gracias por su apoyo y comprensión.

A mis hermanos.

AGRADECIMIENTOS

A mi familia por el apoyo incondicional, a la Universidad Autónoma de Querétaro en especial a la Facultad de Ingeniería. Al Consejo Nacional de Ciencia y Tecnología por el financiamiento otorgado. Al Dr. Roque que me brindo la oportunidad de ser parte del grupo de postrado de la Universidad. A mis maestros y compañeros por su aportación para mi formación académica. A mis asesores Roque, Luis, Agustín, Rooney y René por todo el apoyo en la realización de esta tesis. De igual manera a la maestra Gavina por su apreciable ayuda. Por todo y para todos, ¡Gracias!

ÍNDICE

INDICE DE TABLAS

INDICE DE FIGURAS

I. INTRODUCCIÓN

Actualmente la ciencia y el desarrollo de tecnología son herramientas indispensables para que un país progrese y pueda competir a nivel mundial. La innovación e implementación de nuevas tecnologías en nuestro país es un tema de suma importancia debido a que la mayor parte de tecnología utilizada en las empresas son de procedencia extranjera, lo que ocasiona que inviertan más para adquirir repuestos, en la capacitación de personal, mantenimiento, entre otras cosas.

De acuerdo con los censos económicos del 2006, el sector manufacturero en México es el más importante. Sin embargo, dichas fábricas carecen de tecnología reciente, herramientas de automatización y en ocasiones hasta de personal para dar solución a los problemas a la misma (INEGI, 2006). Es por ello que las industrias no son competitivas comparadas con las grandes corporaciones internacionales, ocasionando en muchos casos la quiebra de las mismas.

Los datos en importación de maquinaria para industrias de plástico marcan un crecimiento de la presencia de países como: Corea del Sur, China, Taiwán, y Hong Kong en el mercado latinoamericano. Considerando que existe un menor costo de la maquinaria asiática frente a las ofertas de América Latina; México es el país más importante en la demanda de esta (Tecnología del Plástico, 2006).

Buscando dar solución a los problemas antes mencionados se han generado planes que permitan solventarlos. El Plan Nacional de Desarrollo 2001 – 2006 estableció que México requiere formar profesionistas, especialistas e investigadores capaces de crear, innovar y aplicar nuevos conocimientos que beneficien a la sociedad en su conjunto. Por lo que se decretó el programa denominado Programa Especial de Ciencia y Tecnología (CONACYT, 2006). Los Fondos Sectoriales y Mixtos constituyen uno de los instrumentos estratégicos para impulsar la inversión en la investigación científica y en el fortalecimiento

tecnológico en áreas como salud, educación, desarrollo económico y social, entre otras (CONACYT, 2006).

Es por ello y por el avance de la ciencia y tecnología que es tiempo de iniciar la creación de herramientas para solucionar los problemas de la industria, sin la necesidad de depender de tecnologías específicas, proporcionando herramientas factibles que estén a la mano de estos sectores.

En este trabajo se presenta el diseño e implementación de un teclado industrial aplicado a una máquina de inyección de plástico, con la intención de generalizar y adaptar el teclado a cualquier máquina herramienta o proceso de automatización con solo una reprogramación del FPGA (*Field Programmable Gate Arrays,* arreglo de puertas programadas en campo), para proporcionar una alternativa a problemas específicos, buscando herramientas adecuadas en costo y funcionalidad en combinación con los conocimientos en el campo de la ingeniería electrónica y control.

El contenido de este capítulo presenta los antecedentes respecto al desarrollo de tecnología así como objetivos, justificación y planteamiento general del trabajo. En el capítulo dos se hace una revisión de literatura existente referente a los protocolos de comunicación, lógica programable y las metodologías de diseño asociadas como es el caso de los lenguajes de programación. Por otra parte el capítulo tres describe la metodología para el desarrollo de la interfaz de comunicación entre el teclado y la PC, así como el proceso del sistema de los módulos de control que permitan implementar el protocolo de comunicación AT, además de un prototipo del proyecto. Los resultados obtenidos de la investigación son presentados en el capítulo cuatro y en el apartado cinco se exponen las conclusiones y la perspectiva del trabajo. Por último se anexa una sección de apéndices referentes al mismo.

1.1 Antecedentes

Hoy día, la automatización de procesos o maquinaria es una parte fundamental de las empresas competitivas, para ello se necesitan sistemas de control flexibles y económicos.

El uso de una PC como controlador en los procesos industriales agrupa una serie de ventajas: la facilidad de mantenimiento, bajo costo de reposición y actualización y la gran variedad de proveedores que hay en el mercado (Bravo, 2004). Algunos trabajos desarrollados en el diseño e implementación de teclados se mencionan a continuación: Varela et al. (2007), propusieron el diseño de un teclado matricial de 70 casillas, en el cual emplearon un sistema de separación mediante material aislante creando campos de activación y por medio del microcontrolador PIC16F873A interactúan con la PC. De Priego (2007) diseñó un teclado matricial de 4 filas y 4 columnas para el aprendizaje y diseño de circuitos digitales, en el cual conectó las filas a tierra y las columnas a un voltaje positivo por medio de una resistencia; para saber qué tecla se activaba utilizó un contador de módulo 16, pasaba por un decodificador y se visualizó en un LCD, se eligió un CPLD (*Complex Programmable Logic Device*, dispositivo lógico programable complejo) para la implementación.

Estévez (1998) propuso la utilización de un teclado de conceptos para el desarrollo de la comunicación no vocal de alumnos, con déficit motores graves, este se conectó directamente a la computadora; consistía de 8 filas y 16 columnas, teniendo 128 celdas que se podían configurar como se deseara. Bravo et al. (2007) utilizó un teclado matricial para la programación de una freidora industrial, se controlaba por medio de un microprocesador PIC16F877 programado en lenguaje C. Dussán et al. (2004), elaboraron 3 teclados para el aprendizaje progresivo de la lectoescritura braille en niños de 3 a 8 años. Las teclas que se activaban eran almacenadas y grabadas en el codificador de voz (CI ISD4004), todo estaba controlado por el microcontrolador Motorola 68HC908GP32.

La Facultad de Ingeniería de la Universidad Autónoma de Querétaro, ha realizado una serie de procesos tecnológicos para dar solución al problema antes mencionado. Parte de ello, es el desarrollo e implementación del control modular aplicado a una máquina de inyección de plástico, que proporciona las ventajas de flexibilidad y

economía en el control de procesos industriales (Bravo, 2004). El control de temperatura en este tipo de máquinas constituye uno de los factores importantes dentro de su funcionamiento integral, para ello se identifican las diferentes zonas de calentamiento en el cañón, utilizando una tarjeta de adquisición con un ADC, para tener el modelo del sistema, sintonizar el controlador y desarrollar las diferentes pruebas y ajuste (Enríquez, 2006). Ésta misma máquina tiene una PC como interfaz con el usuario, se manipula a través de un teclado comercial que se modificó por medio de software; se elaboró un programa en C++ que monitorea, sí se presionó una tecla se procesa la información y continua monitoreando las otras entradas (Bravo, 2004).

Los trabajos antes mencionados su diseño es de tipo matricial y tienen la desventaja de solo activar una tecla. Por ello, en esta investigación se pretende desarrollar un teclado no matricial con aplicación general, que permita la edición y acceso de parámetros en la secuencia del programa; así como la inclusión de teclas que realicen funciones específicas en la secuencia de control, se aplicará a una máquina de inyección de plástico y además dará la pauta para utilizarlo en otros procesos industriales como máquinas-herramienta.

1.2 Objetivos

El crecimiento del país no ha sido satisfactorio en el campo de la creación de tecnología nacional, pues aún se requiere de la importación de equipo, maquinaria, y accesorios para resolver los problemas que tienen las empresas del sector industrial. Con base en lo anterior se fijó el objetivo para la realización de este trabajo.

Diseñar y elaborar un teclado por medio de lógica programable para una máquina de inyección de plástico, que permita la edición y acceso de parámetros en la secuencia del programa; así como la inclusión de teclas que realicen funciones específicas en la secuencia de control y presentar una opción de estandarización. Este constará de 64 teclas y su diseño será no matricial. Subrayando los objetivos particulares:

- Realizar módulos digitales programados en VHDL, así como la simulación y síntesis de los mismos, que permitan la interacción del teclado con la PC.
- Trazar un primer prototipo del teclado industrial que permita la interconexión dinámica con el teclado comercial, sin necesidad de software adicional.

1.3 Justificación

Como se mencionó anteriormente, México es uno de los países con mayor índice de importación en maquinaria debido a las empresas trasnacionales, pero la mayoría de las empresas en el país son MPYMES (micro, pequeñas y medianas empresas), dichas empresas cuentan con maquinaria muchas veces ya obsoleta para los procesos y no cumplen con los requerimientos de eficiencia, por esta razón, los esfuerzos de generar nueva tecnología van encaminados hacia ellas.

La justificación de este trabajo es la necesidad de contar con un teclado industrial que permita tener mayor facilidad para la programación y manipulación en una máquina inyectora de plástico y con ello desarrollar tecnología propia e independencia tecnológica. El teclado ofrecerá una interconexión simultánea con el teclado comercial, su diseño será un arreglo no matricial, tendrá un total de 64 teclas, se podrá aplicar a otros procesos o máquinas herramientas con solo una reprogramación del FPGA y no necesita de software adicional para su instalación. Una característica importante que presentará el proyecto con este arreglo es que se pueden presionar varias teclas al mismo tiempo y la información no se pierde. Con el desarrollo de nuevas alternativas para la automatización de procesos o maquinaria significaría un menor costo para ello, logrando así que las MPYMES tengan la oportunidad de mejorar sus procesos.

1.4 Planteamiento general

Con el desarrollo de los dispositivos lógicos programables (PLDs) y las metodologías asociadas como es el caso de los lenguajes de descripción de hardware, es posible construir sistemas digitales complejos constituidos en un solo circuito integrado con la ventaja de ser configurable.

Una de las características que presenta el proyecto es que cuenta con una interfaz la cual permite la conexión simultánea con otro teclado, sin necesidad de dispositivos o software adicionales y en cualquier instante se puede inhibir el teclado comercial por lo que se recurrió al diseño antes mencionado. La Figura 1.1 muestra el diagrama a bloques de los módulos de control que contiene el teclado diseñado.

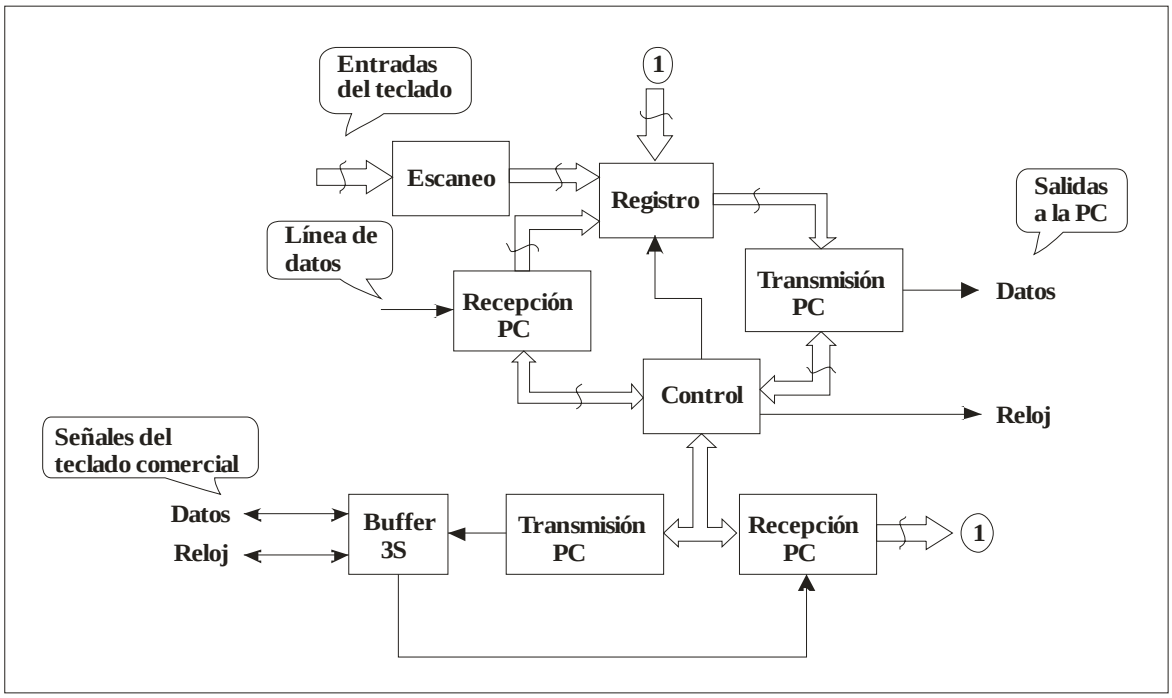

Figura 1.1 Diagrama a bloques de los módulos de control del teclado.

Con estas estructuras digitales será posible adquirir que tecla o teclas se activaron, almacenar la información para después transmitir. Con el módulo de control se proporcionará la sincronización de los diferentes módulos.

Uno de los aspectos importante a considerar cuando se diseña un equipo para uso industrial es el protocolo de comunicación que se utilizará (Bravo, 2004), para el desarrollo del proyecto se empleó el protocolo AT y para su implementación se diseñaron módulos digitales que permitan generar el código, adicionar los bits inicio, stop y paridad, y transmisión del dato, y un sistema de control por medio de una máquina de estados que controle la secuencia de envió y recepción de datos, así como la sincronización de operación de los módulos. De igual manera se generó una estructura digital para proporcionar la señal de reloj empleada en la comunicación entre ambos dispositivos.

II. REVISIÓN DE LITERATURA

En esta sección se presenta la base teórica necesaria para el desarrollo del teclado, mencionando algunos conceptos que son necesarios para la implantación del mismo y dando una breve reseña del desarrollo de los circuitos integrados.

2.1 Teclados

Consisten en un conjunto de botones que son supervisados por un procesador llamado encoder (Chapweske, 1992). Las operaciones internas están controladas por algún procesador, que se programa para realizar un gran número de funciones supervisoras; estas incluyen auto-test, comprobación periódica, el manejo de hasta 20 códigos para la comprobación del mismo, y comunicaciones seriales bidireccionales con la unidad del sistema (Sikonowiz, 1985).

2.1.1 Tipos y funcionamiento

La compañía IBM introdujo teclados para cada uno de sus principales computadoras de escritorio. El IBM PC original y después el XT. En 1984 surge el AT, seguido del PS/2, los cuales son los más utilizados actualmente.

- Teclado IBM. Se conecta a la PC mediante una interfaz similar a un puerto serie. Al presionar una tecla, se envía un código desde el teclado hacia la PC, al levantar la tecla, le manda el código de desactivación al ordenador. La ROM BIOS del ordenador recoge los datos y se encarga de transformarlos en caracteres del código ASCII y de manejar las operaciones relacionadas con las pulsaciones.

- Teclado XT. Surge en el año de 1981, consta de 83 teclas y tiene un sistema de comunicación unidireccional, utiliza un conector DIN. En la Tabla 2.1 se observa las señales y terminales. Cuando el teclado tiene un byte que enviar a la computadora, transmite una trama de 9 bits, sincronizada con 9 pulsos de reloj. La trama está formada por un bit de arranque y ocho bits de datos, y la velocidad del envío es de 2000 bits/s aproximadamente. Cuando el bit de iniciación llega al registro de desplazamiento, se encarga de mantener la línea CTS (clear to send) a nivel bajo. Cada vez que la PC lee el registro de desplazamiento, solo tiene que limpiarlo y habilitar dicha línea.

Terminal	Señal
1	CLK/CTS
2	RxD
3	Reset
4	Tierra
5	+5 Volts

Tabla 2.1 Señales del conector para el teclado XT.

- Teclado AT. Se conectan a la placa base a través de un DIN redondo de 5 terminales macho. Se estandarizó en el año 1986 y tiene entre 101 y 102 teclas. Internamente están formados por una matriz de pulsadores que dan señales directas a la PC identificando la tecla de manera física. Usan un DIN de 5 pines, la comunicación es bidireccional. Cuando la PC tiene necesidad de enviar un byte al teclado, habilita la línea RTS y deshabilita CTS; el teclado tarda en contestarle 10 pulsos de reloj (lo que establece una velocidad de 10 000 baudios). La PC recoge la trama de datos a partir de la línea TxD de forma síncrona, la cadena de información consta de un bit de inicio, 8 de datos, y uno de paridad. La Tabla 2.2 muestra las señales y las terminales correspondientes al tipo de conector empleado (Martín et al, 2005).

Por otro lado y dentro de las características de funcionamiento existen teclados que contienen una lámina de material elastomérico (especie de caucho) entre las teclas y la tarjeta impresa subyacente, al ser oprimidas se cierra el circuito, otros tienen un imán debajo de ellas que pasa por el interior de una bobina, cuando se oprime induce una corriente que puede detectarse (Tanenbaum, 1999).

Terminales	Señal
1	CLK/CTS
2	RxD/TxD/RTS
3	NC
4	Tierra
5	+5 Volts

Tabla 2.2 Señales del conector para el teclado AT.

2.2 Interfaz

En cualquier computadora existe una parte importante llamada subsistema de entrada/salida (E/S), es la que hace posible la comunicación con su entorno. Está formado por varios dispositivos periféricos que proporcionan el medio para intercambiar datos con el exterior y el procesador de la PC; físicamente, es una tarjeta de circuito impreso (Rodríguez, 2006).

2.2.1 Interfaz PS/2 y AT

El puerto PS/2 es uno de los dos tipos de conectores que usan los teclados comerciales: el din de 5 terminales y el mini din de 6. Eléctricamente ambos son similares, la única diferencia es el arreglo de sus conexiones (Chapweske, 1992). La Figura 2.1 muestra el conector min-din para teclados PS/2. La terminal 1 es la línea por la cual se transmite la informacion de éste a la PC y recibe el dato proveniente de ésta última, la

transmisión y recepción de datos se realiza en forma serial. Los pines 3 y 4 se utilizan para energizarlo, tierra y Vcc (+5volts) respectivamente. La terminal 5 es la señal de reloj que sirve para sincronizar la transmisión y la recepción de información (Mclaughlin, 1995).

La Figura 2.2 muestra el conector din para teclados PC estándar. A diferencia del anterior, este tiene menos terminales y cambia su configuración. La terminal 1 es la señal de reloj, la 2 se usa para los datos, 4 y 5 para la alimentación tierra y Vcc respectivamente.

Figura 2.1 Conector mini-din para teclado PS/2.

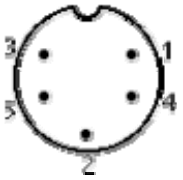

Figura 2.2 Conector din para teclados PC estándar y AT.

2.2.2 Interfaz serie

Para la transmisión con esta interfaz, los bits de información se van enviando uno a uno por medio de una línea de datos y puede ser síncrona y asíncrona. La principal ventaja de la comunicación asíncrona es que el receptor puede funcionar a diferentes frecuencias; por otra parte en la síncrona es necesario que tanto receptor y emisor trabajen a una misma frecuencia (Rodríguez, 2006).

2.3 Protocolos de comunicación

Uno de los aspectos más importantes a considerar al diseñar un equipo para uso industrial es el protocolo de comunicación que se utilizará. Existen varios estándares industriales aprobados por diferentes asociaciones internacionales alrededor del mundo. A continuación se explican las características de algunos de los más utilizados en la industria, tanto en intercambio de información como en buses de campo.

- RS-232. Para enviar datos a grandes distancias, lo factible es aprovechar las estructuras ya disponibles en lugar de realizar nuevos tendidos de líneas. En este aspecto las líneas telefónicas resulta una alternativa, sin embargo, el ancho de banda del canal telefónico va de 300 a 3 300 Hz y las señales digitales tienen un espectro diferente, es decir se deben modular antes de emitirlas (convertirlas en tonos audibles) y remodularlas al recibirlas.

  La EIA (*Electronic Industries Asociation*, Asociación de industrias en electrónica), estableció en la década de los 60's un patrón para la comunicación serie: la norma RS-232C (*Recommended Standard 232, revision C*, estándar recomendado 232 revisión C). Posteriormente, en la década de los noventa se cambió el nombre de RS-232 a EIA-232. Esta describe de forma detallada, las funciones de 25 líneas de señal y protocolo para las comunicaciones serie, los niveles de tensión del 0 y 1 lógico, las impedancias y capacidades de la línea, entre otras. Los conectores habituales son el DB-25 y el DB-9 (Zavala et al, 2007).

  Las ventajas principales que definen a este sistema de comunicación son la velocidad máxima de transmisión de datos que es de 20 Kbits/s y la capacidad de carga máxima. La Tabla 2.3 muestra algunos parámetros importantes de este protocolo (Mateos et al, 2007).

Longitud máxima de la línea	50 ft
Frecuencia	20 Kbaudio/ 50 ft
Modo de comunicación	Simple
Nivel lógico “0”	Menor a 3 Volts
Nivel lógico “1”	3 a 25 Volts
Número de receptores en línea	1
Impedancia de entrada	3 a 7 KΩ

Tabla 2.3 Parámetros para la Norma RS-232

- USB. El puerto USB nació con la finalidad de facilitar el uso de periféricos en equipos de cómputo. Provee una forma fácil y estandarizada para conectar hasta 127 dispositivos a una computadora; permite la conexión y desconexión de los dispositivos en cualquier momento.

  Contiene una línea de alimentación (5 Volts) y otra de tierra; la velocidad máxima de transmisión es de 480 Mb/s con el estándar 2.0 (Piedrahita, 2001). Fue diseñado para conectar dispositivos a la computadora, eliminando la conexión de tarjetas PCI y la reinicialización de la misma. Sin embargo, actualmente también se utiliza para consolas de juegos e incluso en sistemas de audio y video.

  Existen tres tipos de velocidades en la transferencia de información, los cuales se muestran en la Tabla 2.4. La de baja velocidad se emplea en dispositivos de interacción con la computadora como mouse, teclado, impresoras, monitores y joysticks. Los elementos de alta velocidad se emplean para aplicaciones de video y almacenamiento (Hoffman y Szmulewicz, 2006).

Tipo	Velocidad
Baja velocidad	183 Kbytes/s
Velocidad completa	1.4 Mbytes/s
Alta velocidad	57 Mbytes/s

Tabla 2.4 Velocidades de transferencia USB.

- AT. Tuvo mucha popularidad desde la década de los 80's con la fuerte ubicación en el mercado de las computadoras personales compatibles con IBM; es una ampliación a 32 bits del bus ISA, que era solo de 16 bits, y que a su vez fue un desarrollo basado en el bus original de la IBM-PC. Originalmente, el bus de la PC operaba a 4.77 Mb/s.

2.4 Protocolo AT

La comunicación entre el teclado y la PC se lleva acabo por medio del protocolo AT, que es serial, bidireccional, síncrona, la transmisión empieza con LSB (*Less Significant Bit,* bit menos significativo), se lleva acabo en 11 transiciones de reloj y en cualquier instante la PC puede inhabilitar la comunicación y utiliza el conector din (Fig. 2.2). En este caso el teclado tiene la función de generar los pulsos de reloj (10 a 20 KHz) y el tamaño del dato o scan code (en hexadecimal) es de 11 bits los cuales son:

- 1 bit de inicio, normalmente es 0.
- 8 bits de datos, la transmisión inicia con el menos significativo.
- 1 bit de paridad, es el número de bits en "1" (nivel alto) dentro del dato enviado, es 1 si el número es par y 0 si es impar. Se emplea para detectar posibles errores en la información.
- 1 bit de stop, siempre es 1.

Este protocolo usa una interfaz de colector abierto con resistencias de pull up a +5 volts, con ello se tiene dos posibles estados: bajo o alta impedancia. En el estado bajo, una transistor mantiene las líneas (reloj y datos) en 0 volts; el de alta impedancia la interfaz se comporta como un circuito abierto. Sin embargo, como se tiene resistencia de pull up el bus se mantiene en nivel alto, si ninguno de los dispositivos (teclado y PC) se mantienen en nivel bajo.

2.4.1 Comunicación entre el sistema y el teclado

Cuando el sistema quiere enviar información al teclado, primero debe mantener las líneas de reloj y datos en el estado *Reques-to-send*, para ello es necesario prevenir que el teclado transmita datos al mismo tiempo. Es normal mantener la línea de reloj en cero por más de 60 μs, después la de datos se lleva a cero y se libera la de reloj, como se muestra en la Tabla 2.5.

El teclado debe monitorear el bus en intervalos no excedentes a 10 ms, cuando ha detectado el estado *request-to-send*, empieza a generar la señal de reloj. La PC envía los datos en la TN (transición negativa) y son leídos en la TP (transición positiva); después de recibir el bit de stop el teclado mantiene en bajo la línea de datos, y se emite el bit de reconocimiento (ACK) con un último pulso de reloj. Si la PC no libera la línea de datos, se sigue mandando la señal de reloj hasta que la libera cuando esto ocurre se genera un comando de error. La Figura 2.3 muestra el diagrama de tiempos para la transmisión.

Datos	Reloj	Estado
1	1	***Idle*** (Estado inactivo)
1	0	Comunicación inhabilitada
0	1	***Request to send*** (Solicita enviar)

Tabla 2.5 Estados del Bus.

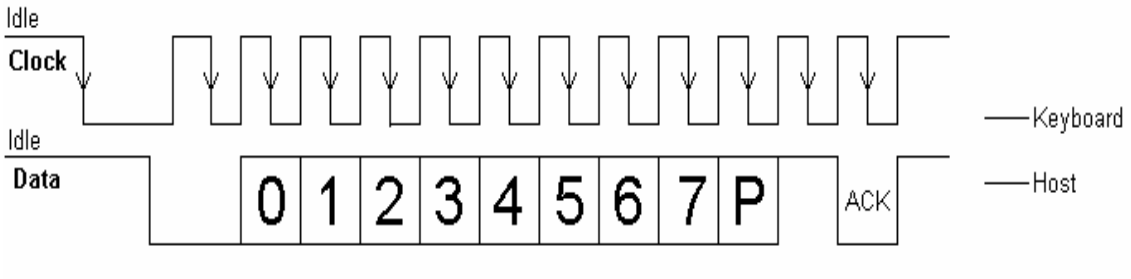

Figura 2.3 Diagrama de tiempos para la comunicación del sistema al teclado.

2.4.2 Comunicación entre el teclado y el sistema

Cuando el teclado requiere enviar información, primero verifica que el estado del bus sea alto (ambas líneas de reloj y datos). Si no lo esta entonces la comunicación esta inhabilitada y debe guardar los datos hasta que la PC libere la línea de reloj; esta debe estar en alto por lo menos 50 μs antes que se empiece la transmisión. En el tiempo alto de la señal de reloj se carga el dato en la línea y esta listo para el sistema en la parte baja del mismo y tienen una duración de por lo menos 5 μs y como máximo 50 μs. La Figura 2.4 muestra el diagrama de tiempos de este caso.

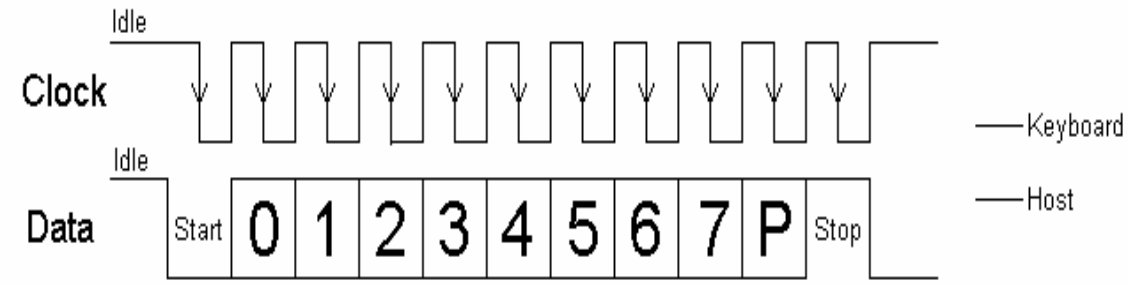

Figura 2.4 Diagrama de tiempos para la comunicación del teclado al sistema.

El sistema puede inhabilitar la comunicación en cualquier instante manteniendo la línea de reloj en bajo por lo menos 100 μs, si esto ocurre antes del onceavo pulso de reloj el teclado aborta la transmisión y se prepara para el reenvío del dato completo; es decir los 11 bits. Si el sistema mantiene la línea de reloj antes de la primera transición alta a baja o después del pulso alto, el teclado no necesita retransmitir la información. Sin embargo, si se crea nueva información que necesita ser enviada, se tendrá que guardar hasta que se libere la línea.

2.4.3 Comandos

Son códigos (hexadecimales) que emplean los dos dispositivos para dar instrucciones y realizar una operación correspondiente a la indicación emitida. En la Tabla

2.6 se muestra algunos comandos que se emplean entre la comunicación del teclado y la PC, los cuales se utilizan para la inicialización del teclado; estos pueden ser enviados en cualquier instante, y deberá responder en un intervalo no mayor de 20 ms, excepto cuando esta efectuando la prueba básica de seguridad (*Basic Assurance Test*, BAT) o que este ejecutando un comando de reset. De igual manera el teclado puede transmitir algunos comandos a la PC, estos se muestran en la Tabla 2.7.

Comando	Significado	Comentarios
ED	Prende o apaga el (los) led (s) de estado, depende del byte que se envíe	Bit 0 es Scroll lock Bit 1 es Num lock Bit 2 Caps lock
F0	Selecciona el modo 1, 2, o 3	
F2	Envía el identificador de teclado	
F3	Establece el valor de repetición y el tiempo de espera	
F4	Borra búfer	
F5	Restablece los valores por defecto y espera una habilitación	
F6	Restablece los valores por defecto	
FE	Error: solicitud de retransmisión	
FF	Reset del teclado	
00	Apaga todos los leds	
02	Enciende el led de Num Lock	

Tabla 2.6 Protocolo para inicialización del teclado AT.

Comando	Valor hexadecimal
Detección de error en el dato, o desbordamiento en el buffer	FF
Prueba de encendido satisfactoria	AA
Reconocimiento (ACK)	FA
Reenvío	FE
Falla en la prueba de encendido	FC
Identificador del teclado	83AB

Tabla 2.7 Comandos del teclado a la PC.

2.5 Lógica programable

Anteriormente existían dos alternativas al momento de implementar un algoritmo digital: su ejecución en un procesador de propósito general o la realización en hardware a medida. La primera es muy barata y flexible, pero generalmente lenta. La segunda es de muy alto desempeño, pero por su costo solo se justifica en aplicaciones de uso masivo (coprocesadores numéricos, chips empleados al procesamiento de señales, entre otros) y en general no permite flexibilidad; debido a que los dispositivos utilizados son de uso específico.

Hoy día, la lógica reconfigurable permite combinar las dos soluciones anteriores y obtener velocidades de hardware adecuadas y flexibilidad de software. La posibilidad de tener hardware reprogramable abarata su costo, ya que puede utilizarse exactamente el mismo hardware para varias aplicaciones cambiando solo su programación interna (Oliver y Pérez, 1999).

Existen una gran variedad de circuitos integrados que pueden tener sus funciones lógicas programadas en su interior después de su fabricación. La mayor parte de estos dispositivos utilizan una tecnología que permite también su reprogramación, lo que significa que si se encuentra un error en el diseño, este se pueda arreglar sin reemplazar físicamente o volver a alambrar el dispositivo.

A continuación se presenta la clasificación de los circuitos integrados digitales, la cual se basa en dos características importantes: la estructura interna del dispositivo y la forma en que se desarrollan.

2.5.1 Circuitos integrados

Un circuito integrado (IC, integrated circuit) es una colección de componentes pasivos y activos que internamente están conectados entre ellos en uno o varios sustratos de

material semiconductor. La característica más notable de un IC es su tamaño ya que es mucho más pequeño que una estructura semiconductora construida con componentes discretos. En la actualidad, existen tres tipos de IC disponibles comercialmente a gran escala; incluyendo los circuitos integrados monolíticos, los de película delgada (o gruesa) y los híbridos.

2.5.2 Clasificación de los circuitos integrados monolíticos

El término monolítico se deriva de palabras griegas, *monos*, y *lithos*, que significa solo y piedra respectivamente, con ello podemos definirlos como circuitos integrados construidos con una sola oblea de material semiconductor (Boylestad y Nashelsky, 1997). Se pueden clasificar de acuerdo con dos conceptos interrelacionados y esta se observa en la Figura 2.5.

a) Conforme a la forma en que se elaboran físicamente, lo que da lugar a:
   - Dependiendo del tipo de semiconductor utilizado (silicio, arseniuro de galio, y recientemente silicio-germanio). Para su desarrollo se emplean transistores MOS (*Metal Oxide Semiconductor,* semiconductor de óxido de metal) de canal N y P y bipolares.

   - De acuerdo al número de dispositivos colocados en su interior. En las primeras integraciones se utilizaron compuertas NAND y NOR, en un número de 1 a 6 en función de la cantidad de entradas, y biestables del tipo J-K, D activado por flancos (edge-triggered) y D cerrojo (latch).

b) Según la forma en que se realiza el diseño desde el punto de vista del ingeniero:
   - Normalizados o estándar. Reciben el nombre de normalizados, estándar o comerciales aquellos dispositivos que tienen una arquitectura establecida por el fabricante y puede ser fija o programable.

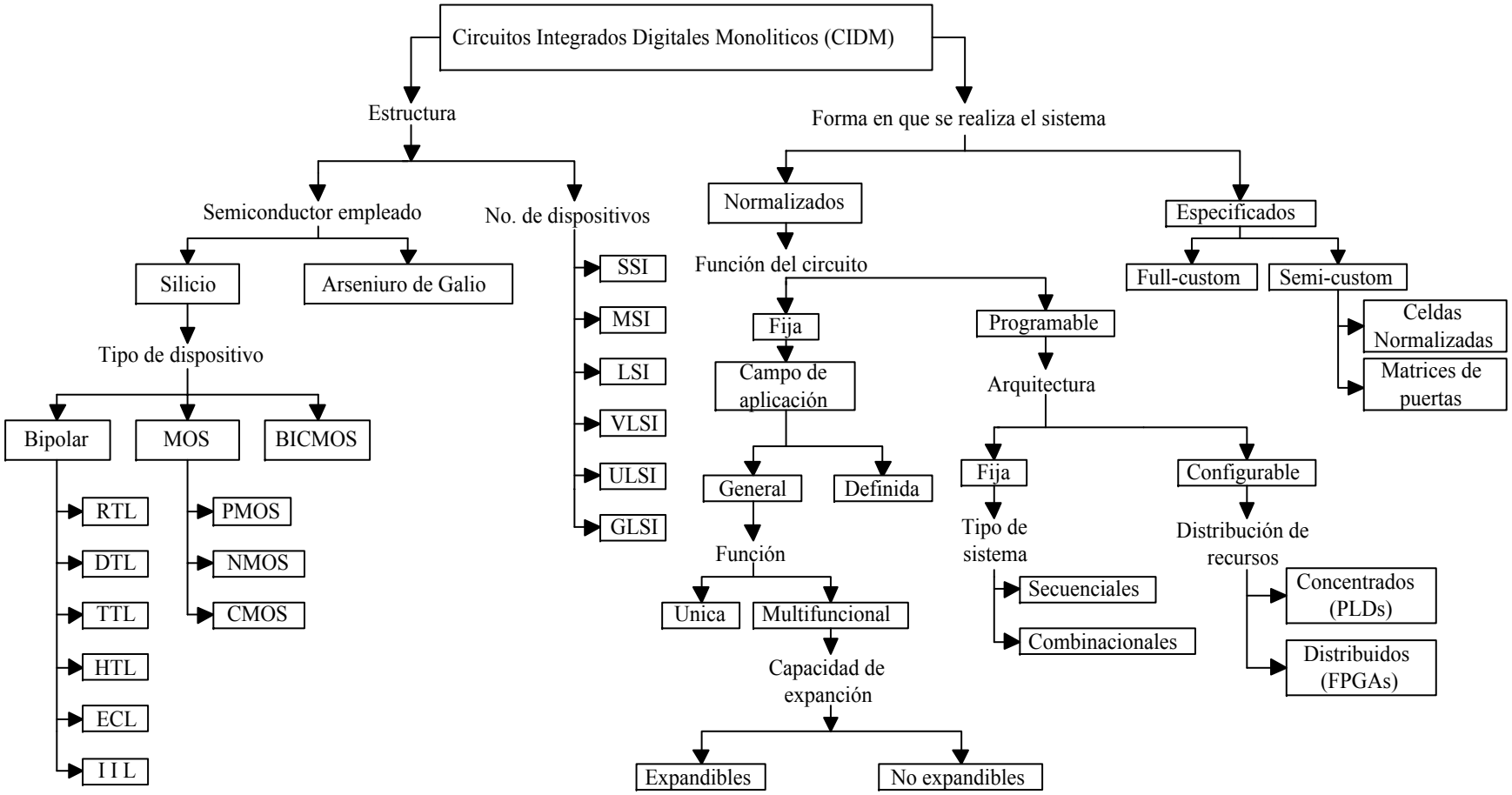

Figura 2.5 Diagrama a bloques de la clasificación de los circuitos integrados digitales monolíticos (CIDM).

- Específicos por el usuario. Estos se desarrollan a la medida del usuario y por ello reciben el nombre de circuitos integrados a la medida (Custom Integrad Circuits).

2.5.3 Circuitos digitales configurables

La configurabilidad es un concepto asociado a los sistemas electrónicos digitales cuya función se puede modificar utilizando solamente una parte de los elementos que los componen y/o cambiando la interconexión entre ellos, dicha modificación se lleva acabo mediante la programación del estado de un conjunto de variables binarias independientes o asociadas entre sí formando una determinada estructura de memoria. Para hacer posible la modificación de la función de estos chips es necesario distribuir especialmente:

- Los recursos lógicos, lo que facilita la realización de circuitos digitales de elevada complejidad.

- Los recursos de interconexión entre los bloques lógicos, para facilitar el enlace entre los mismos.

La Figura 2.6 muestra la clasificación de este tipo de circuitos, basándose en la distribución de los recursos lógicos y los de conexión.

Estos dispositivos surgen siguiendo dos grandes tendencias:

- Dispositivos lógicos programables (PLD, *Programmable Logic Device*). Son circuitos de granularidad reducida o media (la granularidad se refiere a la concentración de recursos lógicos), que surgen a partir de las matrices lógicas programables PLA y PAL, añadiéndoles recursos lógicos cuya conexión se pudiera modificar mediante la programación del estado de las variables binarias.

- Conjuntos configurables de puertas (FPGA). Estos poseen diversos grados de granulidad, que se originan de acuerdo a las arquitecturas utilizadas en la síntesis de

los CIDM semi-medida. Una posible estructura es hacer configurable los bloques funcionales e intercambiables las interconexiones entre ellos, mediante la selección de alta o baja impedancia de determinados dispositivos electrónicos.

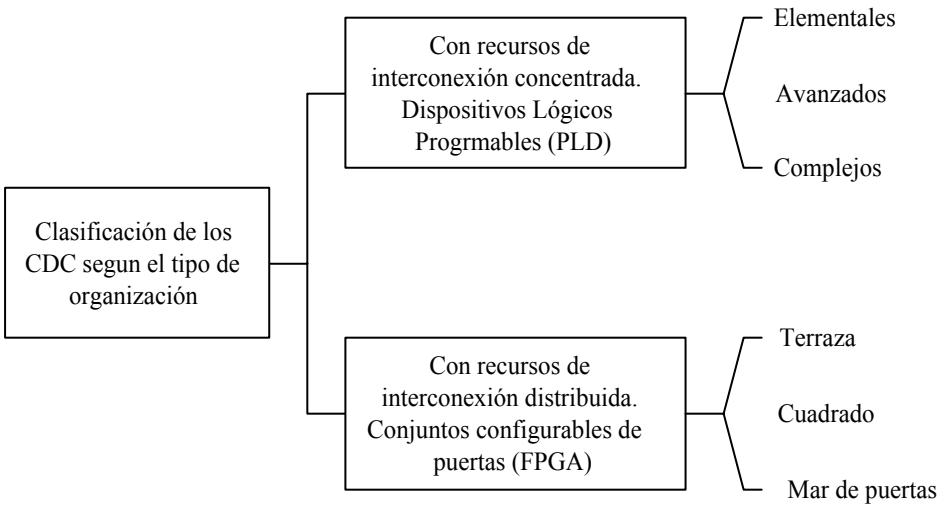

Figura 2.6 Clasificación de los circuitos digitales configurables (CDC) dependiendo del tipo de organización.

Los PLD y FPGA constituyen dos grandes clases de los CDC, que se caracterizan por la forma en que se implementan los recursos de interconexión.

2.6 Dispositivos lógicos programables

Son circuitos digitales configurables que poseen una organización con recursos de interconexión concentrados, la cual tiene sus antecedentes en las matrices lógicas programables y recibe el nombre de matricial (Mandado et al, 2002).

El desarrollo de estos dispositivos se inició mediante la combinación de las matrices lógicas PAL (P*rogrammable Arrays Logic*, lógica de arreglos programables) y PLA (*Programmable Logic Arrays,* arreglos lógicos programables) con un registro

configurable. Estos contenían matrices de compuertas AND y OR con una estructura de dos niveles, con conexiones programables por el usuario. Actualmente, tales dispositivos se conocen de forma genérica como PLD. La Figura 2.7 muestra el diagrama a bloques de la estructura interna de estos dispositivos (Wakerly, 2000).

Se utilizan para reemplazar a los circuitos SSI y MSI ya que ahorran espacio y reducen el número y el costo de los dispositivos en un determinado diseño. Todos los PLD están formados por matrices programables, que son redes de conductores distribuidos en filas y columnas con un fusible en cada punto de interconexión; pueden ser fijas o configurables.

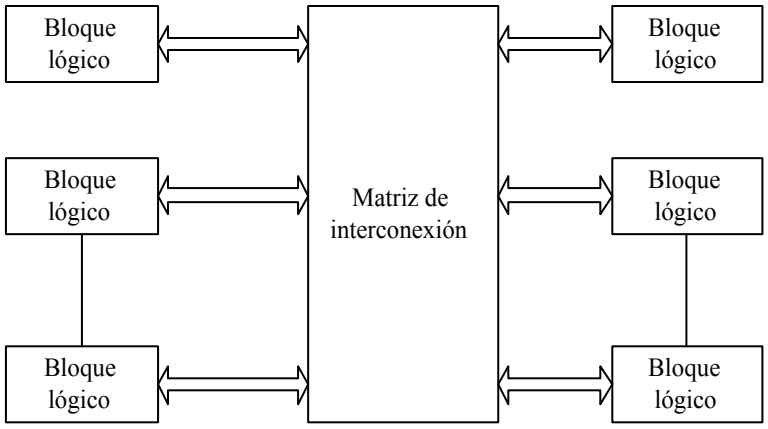

Figura 2.7 Estructura interna de un PLD.

Los PLD se clasifican de acuerdo a su arquitectura, la cual es la ordenación funcional de los elementos internos que proporciona al dispositivo sus características de operación. Dentro de los PLDs se tiene dispositivos: PROM, PLA, PAL, GAL y PLD complejos.

- *PROM (Programmable Read Only Memory,* memoria programable de solo lectura*).* Está formada por un conjunto fijo de compuertas AND conectadas como decodificador y una matriz programable OR, se utiliza como una memoria direccionable y no como dispositivo lógico.

- *PLA (Programmable Logic Array,* arreglo lógico programable*).* Se constituyen por una matriz AND y OR programables.

- *PAL (Programmable Array Logic,* lógica de arreglos programables*).* La arquitectura interna consiste en matrices AND programables que alimentan OR fijas.

- *GAL (Generis Logic Array,* arreglo lógico genérico*).* Las principales diferencias de la GAL y la PAL radican en que es reprogramable y contiene configuraciones de salida programables. Usan la tecnología E^2CMOS (*Electrically Erasable CMOS,* CMOS borrable eléctricamente), en lugar de la tecnología bipolar y fusible.

- *PLD complejos (CPLD).* Consiste en un arreglo múltiple PLD agrupados como bloques en un chip, también conocidos como *EPLD (Enhance PLD,* PLD mejorados*)*. En su estructura contiene varios bloques lógicos conectados por medio de señales canalizadas desde la interconexión programable (PI), esta unidad se encarga de unir los bloques lógicos y los bloques de entrada/salida del dispositivo sobre las redes apropiadas, como se observa en la Figura 2.8.

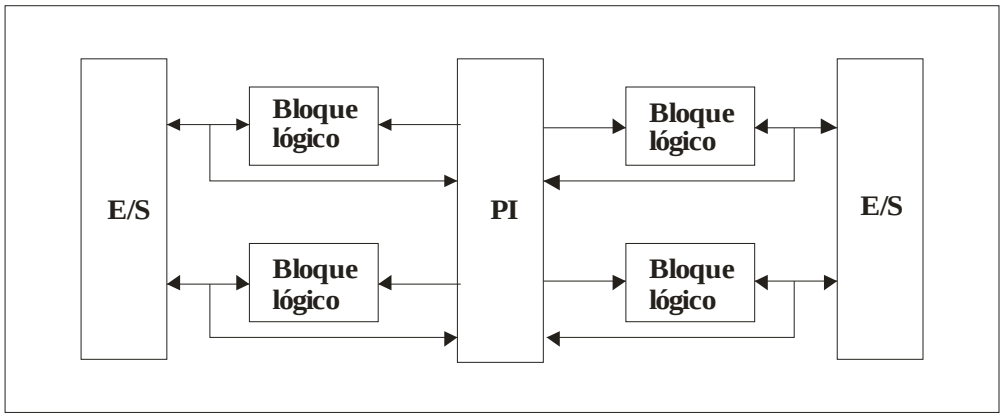

Figura 2.8 Arquitectura básica de un CPLD.

2.7 Conjuntos configurables de compuertas

Son dispositivos que poseen una organización con recursos de interconexión distribuidos, que también suelen denominarse FPGAs. Esta organización tiene sus antecedentes en los circuitos integrados semimedida. Se caracterizan por altas densidades de electrónica combinacional, mayor rendimiento, un gran número de entradas y salidas definidas por el usuario, un esquema de interconexión flexible, y un entorno de diseño similar al de matriz de puertas.

Los FPGA están formados por tres elementos básicos:

- Bloques lógicos internos. Son recursos lógicos que permiten realizar diferentes funciones lógicas, la complejidad puede variar desde un simple par de transistores hasta un conjunto de memorias de acceso aleatorio denominadas tablas de consulta (*look- up tables*), combinadas o no con elementos de memorización.

  Cada uno de estos bloques contiene lógica programable combinacional y registros de almacenamiento. La sección de lógica combinacional es capaz de implementar cualquier función booleana de sus variables de entrada.

- Bloques lógicos de entrada y salida. Establecen el enlace entre los bloques lógicos internos y las terminales de entrada/salida, pueden tener diferentes grados de complejidad. Pueden programarse independientemente para ser una entrada, o salida con control tri-estado o un pin bidireccional. Contiene flip-flops que pueden usarse como buffers de entrada y salida.

- Los recursos de interconexión (*routing channels*). Son un conjunto de líneas e interruptores programables que permiten transmitir las señales entre los bloques lógicos internos y los bloques de entrada/salida y entre ellos mismos.

2.7.1 FPGA organización de terraza

Se caracterizan por tener los bloques lógicos internos colocados en filas, separadas por canales en los que se ubican los recursos de interconexión horizontal, como se muestra en la Figura 2.9. Los bloques lógicos internos pueden tener granularidad fina o gruesa.

La conexión entre ellos y con los bloques de entrada/salida se realiza principalmente mediante recursos de interconexión horizontal, existe también el enlace vertical lo que facilita la comunicación entre los canales horizontales, con este se transmiten señales globales como por ejemplo la de reloj, y se distribuyen por todo el circuito.

2.7.2 FPGA tipo cuadricula

Esta organización está constituida por un conjunto de bloques lógicos internos colocados en forma de filas y columnas delimitadas por los recursos de interconexión que se extienden de forma vertical y horizontal entre los bloques, como se observa en la Figura 2.10.

La complejidad de cada bloque lógico puede variar considerablemente entre los diferentes fabricantes. En estos los recursos de interconexión se intercalan entre los recursos lógicos. Dentro de este tipo se encuentran los FPGA celulares (*Cellular arquitecture* FPGA) que se caracterizan por poseer bloques lógicos simétricos con respecto a sus terminales de entrada y salida.

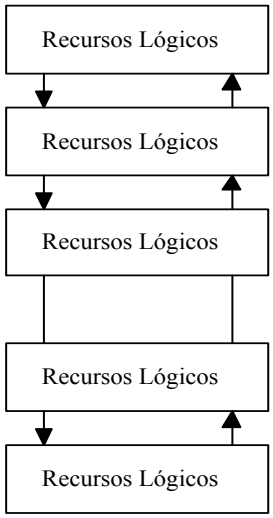

Figura 2.9 FPGA organización terraza.

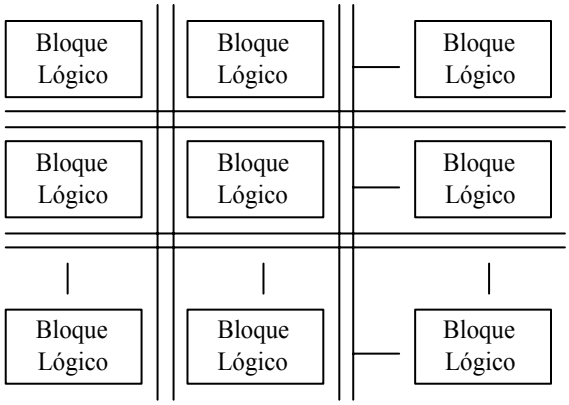

Figura 2.10 FPGA organización cuadrícula.

2.7.3 FPGA organización mar de puertas

En este tipo, los recursos lógicos están formados en filas y columnas, la diferencia con la anterior es que los recursos de interconexión se ocultan con los recursos lógicos. Para ello los recursos de interconexión se sitúan en capas diferentes de las de los bloques lógicos internos, tal como se muestra en la Figura 2.11.

En una primera capa se localizan los bloques lógicos, en general, son elementos lógicos muy simples que permiten implementar puertas lógicas y funciones sencillas. La ausencia de recurso de interconexión en esta capa permite un mejor aprovechamiento del área, por lo que este tipo de distribución posee una gran densidad de recursos lógicos y una granularidad fina.

En la segunda y tercera capa se ubican los recursos de interconexión vertical y horizontal respectivamente, el elevado número de recursos de conexión disponibles en esta estructura permite que las salidas de un bloque lógico puedan conectarse a las entradas de cualquier otro. De esta forma se obtiene un aprovechamiento de recursos cercano al 100 %.

Además las facilidades de enlace de esta organización permiten disminuir los retardos, por lo que su velocidad de operación es menos sensible a las decisiones de partición y de ubicación de los bloques lógicos (Mandado et al, 20002).

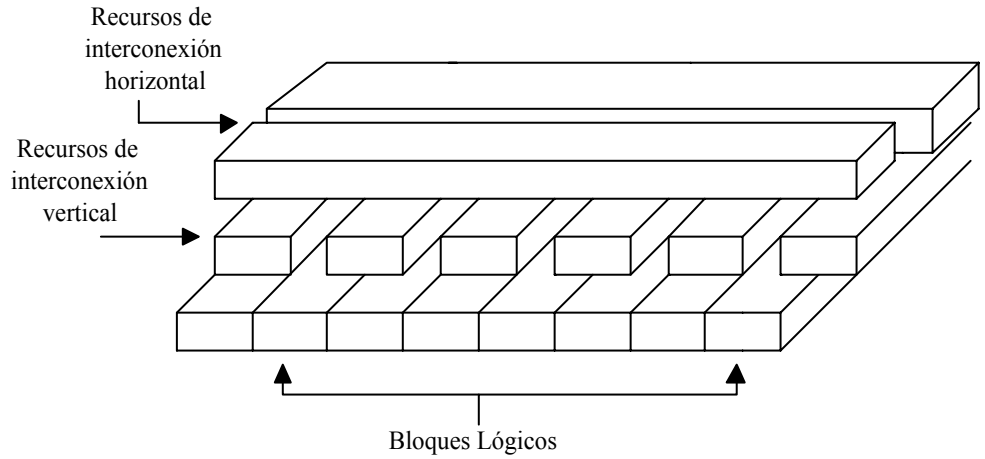

Figura 2.11 FPGA de organización mar de puertas.

2.7.4 Aplicaciones de FPGAs

La lógica programable como parte del desarrollo y evolución de la electrónica digital ha facilitado la implementación de sistemas complejos, reduciendo

considerablemente el espacio y costo en las aplicaciones. Las implementaciones de la lógica programable iniciaron con los PLDs, en la actualidad los FPGA's debido a sus características eléctricas que ofrecen han ido reemplazando componentes como microcontroladores y microprocesadores. Se han desarrollado aplicaciones con FPGA en diversos campos de la automatización y control, procesamiento de señales, CNC, robots, entre otros.

Algunos trabajos desarrollados en robótica se mencionan a continuación: González et al. (2003), desarrollaron un sistema de locomoción aplicado a un robot apodo, donde emplearon un FPGA para diseño de un contador, una memoria ROM y las unidades PWM necesarias para mover al robot.

Por otro lado Palacios (2004), realizó un algoritmo capaz de efectuar los cálculos necesarios para posicionar las articulaciones de dos manipuladores, el cual fue sintetizado en una tarjeta XESS-AS40 de Xilinx que contiene un FPGA de la familia virtex II, se basó en la cinemática inversa conociendo el punto inicial y final de posicionamiento. Ayala et al. (2006), desarrollaron un programa en VHDL que contiene varios módulos (que son estructuras de multiplexores, codificadores y registros) para el control de un robot móvil por medio de una tarjeta Spartan 3 de Xilinx.

De igual manera en el campo de procesamiento de imágenes se tienen varios trabajos con FPGA's como: Vélez et al. (2007), que realizaron un sistema de captura y procesamiento de imágenes a alta velocidad, mediante un control para la adquisición y edición de video, empleando un FPGA de Xilinx XC4010XL. Pérez et al. (2007), emplean un FPGA como procesador de imágenes, lo que permite el procesamiento de secuencia de imágenes en tiempo real; utilizan una interfaz CameraLink para enviar la información de video al dispositivo y emplean un reloj de 80 MHz para sincronizar las señales de entrada. Iborra et al. (2007), emplean una tarjeta Aristotle de Mirotech que contiene un FPGA XC4036 de Xilinx, en la cual programan los algoritmos de procesado como: binarización, erosionados, transformada de Hough y etiquetado, para la inspección visual de cigüeñales de automóviles.

2.8 Lenguajes de descripción de hardware

En las décadas pasadas, la mayor parte del diseño lógico se efectuaba de manera gráfica, mediante el uso de diagramas a bloques y esquemáticos. Sin embargo, con el surgimiento de los lenguajes de descripción de hardware (HDL, *Hardware Description Language*), ligado a los dispositivos de lógica programable y la tecnología ASIC a gran escala, hicieron posible que en la década de los 90's cambiara radicalmente la forma en que se realizaban los diseños digitales complejos (Wakerly, 2000).

Los circuitos de MSI (*Médium Scale Integration,* mediana escala de integración) y LSI (*Large Scale Integration,* gran escala de integración) se diseñaron mediante un prototipo formado por módulos sencillos y la comprobación de su funcionamiento antes de integrarlos. En el caso de los VLSI (*Very Large Scale Integration,* muy gran escala de integración) no resultó practicó realizar un prototipo y por ello fue necesario simular y verificar su correcto funcionamiento antes de constituirlos; lo que trajo consigo la necesidad del desarrollo de métodos de diseño asistido por computadora divididos en varias fases.

La Figura 2.12 representa las distintas fases del diseño de un sistema digital complejo, que a continuación se describe las mismas.

- Nivel sistema. Describe el sistema como un conjunto de módulos semiautónomos y cooperantes, cuya interacción se analiza para obtener un conjunto óptimo.
- Nivel programa, funcional o de comportamiento. Cada módulo del nivel superior se define mediante un algoritmo en un lenguaje de alto nivel (HLL, *High Level Language*). Dada la naturaleza paralela del hardware, en el que varios procesos pueden necesitar un mismo recurso o en el que se deben sincronizar procesos independientes, se utilizan instrucciones similares a las de los lenguajes de programación concurrentes.

- Nivel RTL (*Register Transfer Level*) o de flujo de datos (*Data-flow*). En se describe el sistema mediante diagramas de transferencias entre registros, tablas de verdad o ecuaciones lógicas. Los elementos básicos de este nivel son registros, memorias, lógica combinacional y buses.

- Nivel lógico. Consiste en la descripción del sistema mediante la interconexión de bloques básicos, como compuertas lógicas y biestables.

- Nivel conmutador. Las compuertas se sustituyen por transistores considerados como conmutadores ideales, y dependiendo de la tecnología pueden se bidirecionales.

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

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

Los primeros lenguajes de transferencia fueron entre registros, tenían la capacidad de expresar algoritmos mediante un lenguaje de alto nivel y de definir una estructura realizada con compuertas. Se integraron con simuladores que permitían comprobar el funcionamiento mezclando diferentes niveles; un ejemplo de ellos es el CAP/DSDL. Actualmente los HDL más usados son Verilog y VHDL.

2.8.1 HDL no estructurados

Son HDL sencillos que están orientados a la realización de un único módulo por fichero. Se caracterizan por el hecho de que las herramientas de CAD asociadas con ellos imponen la obligación de diseñar un esquema en el nivel superior de la jerarquía e incluir en él un símbolo que haga referencia a cada uno de los módulos. Uno de ellos es el PALASM (PAL *Assembler*) desarrollado en 1978 por Monolithic Memories para los

dispositivos: PAL 16R4, 16L8 16R8. Este permite describir un circuito mediante sus ecuaciones lógicas o su tabla de verdad y posteriormente convierte de forma automática esa descripción en el patrón de programación del sistema. Solo soportaba las tablas de verdad como forma de descripción de diseño. La descripción de los sistemas digitales complejos mediante este tipo de lenguajes se realiza en un fichero.

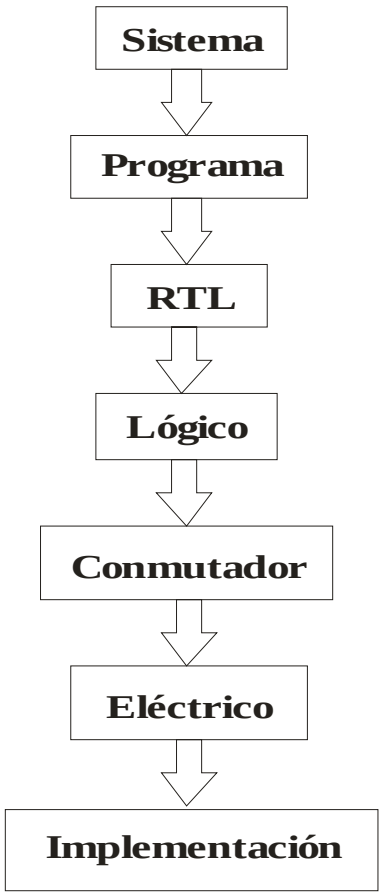

Figura 2.12 Fases de diseño.

2.8.2 HDL estructurados

Además de poseer las características de los no estructurados, permiten llevar a cabo, la descripción estructural jerárquica de los sistemas digitales complejos en un único fichero. Su desarrollo se inició en el año de 1970. Parten del concepto de componente, que

es un circuito digital especificado mediante su descripción funcional o de comportamiento y/o estructural. Cada componente se identifica mediante un nombre y tiene asociado un conjunto de terminales de entrada/salida que permiten establecer conexiones externas (Deschamps, 2002). Uno de estos es VHDL.

2.9 VHDL

En 1980 el Departamento de Defensa de los Estados Unidos inició un proyecto denominado *Very High Speed Integrated Circuit* (VHSIC), con el principal objetivo de desarrollar circuitos integrados en tecnología de 0.5 micras con muy altas prestaciones y resistencia a la radiación.

En julio de 1983 se inició formalmente el proyecto de desarrollo del lenguaje VHDL (*Very High Speed Integrated Circuits Hardware Description Language*, lenguaje de descripción de hardware para dispositivos de alta velocidad), con la participación de tres compañías: Intermeetrics, Texas Instrument e IBM; debería de ser un lenguaje para diseño y descripción de hardware, y en concreto para ser empleado en (Pérez et al, 2002):

- Documentación del diseño. En principio, se estandarizó para la descripción de hardware.

- Descripción genérica de modelos, de tal forma que resultara sencillo configurar un componente en cuanto a tamaño, características físicas, temporales, fan-out, entre otras.

- Subprogramas. Se permite la declaración y definición de funciones y procedimientos para conversiones de tipos, redefinición de operadores, creación de otros nuevos, entrada y salida de datos.

VHDL es un lenguaje de descripción de hardware de gran generalidad derivado del lenguaje de alto nivel ADA. Dispone de tres tipos abstractos para definir el formato y

valores de señales, variables y constantes y proporciona amplias facilidades para la realización de algoritmos. Esta orientado a la descripción o modelado de sistemas digitales, en el cual se puede describir, analizar y evaluar el comportamiento de un sistema electrónico digital; permite la integración de sistemas digitales sencillos, elaborados o ambos en un dispositivo lógico programable de baja capacidad de integración como una GAL o de mayor capacidad como los CPLD y FPGA.

La estructura general de un programa en VHDL está formado por módulos o unidades de diseño, compuesto por un conjunto de declaraciones e instrucciones que definen, describen, estructuran, analizan y evalúan el comportamiento de un sistema digital. Existen 5 tipos de unidades de diseño (Maxinez y Alcalá, 2002):

- Declaración de entidad: consiste en definir las terminales de entrada y salida.
- Arquitectura: describe el funcionamiento del sistema.
- Configuración: se asigna la dirección del flujo del dato (entrada, salida, entrada/salida y buffer).
- Declaración del paquete: son las librerías que se incluyen en el programa.
- Cuerpo del paquete: procedimientos efectuados para llegar al resultado deseado.

Permite modelar y simular un sistema desde un alto nivel de abstracción hasta el nivel lógico más elemental con compuertas y biestables. Permite tres estilos de descripción: algorítmico o de comportamiento, RTL o flujo de datos y estructural.

Los modelos creados pueden ser usados para diferentes tecnologías, las cuales se pueden cambiar con facilidad haciendo uso de herramientas EDA. Se puede especificar características tecnológicas mediante la definición de nuevos tipos, componentes, atributos y parámetros genéricos. Es compatible con las herramientas de diseño disponibles en el mercado; no es un lenguaje de propietario, por lo tanto cualquier usuario puede desarrollar una herramienta para VHDL y comercializarla (Deschamps, 2002).

Admite diseño jerárquico, es decir, un sistema digital puede se modelado como un conjunto de componentes interconectados y cada componente a su vez ser como otro

conjunto de subcomponentes. Los bancos de prueba para simulación pueden escribirse en el propio lenguaje y ser usados para comprobar diversos modelos.

No obstante, existen aspectos del lenguaje que hacen difícil la portabiliad de las descripciones entre los distintos simuladores. Para dar solución a este problema el IEEE. Para solventar el problema, desarrollaron un nuevo estándar, el 1164-1993 o *std_logic_1164*, que declara tipos de datos con 9 valores lógicos.

2.10 Diseño de tarjetas

Normalmente un CI se instala en una tarjeta de circuito impreso (PCB, *printed circuit board*) [o tarjeta de alambrado impreso (PWB, *printed wiring board*)] que lo conectan con otros componentes dentro de un sistema.

Las PCB de capas múltiples (utilizadas en los sistemas digitales típicos) tienen pistas de cobre que son grabadas sobre varias capas delgadas de fibra de vidrio, las cuales están laminadas formando una sola tarjeta que tiene un espesor de 1/16″ aproximadamente. En general las pistas son bastante angostas, normalmente tienen un espesor que va de 10 a 25 mil (milésimas de pulgadas). En la tecnología de PCB de líneas finas, las pistas miden 4 mil de ancho y la separación entre las pistas adyacentes es de 4 mil.

La mayor parte de los componentes que se instalan en las tarjetas utilizan tecnología de montaje superficial (SMT, *surface mount technology*), en lugar de emplear dispositivos con encapsulado DIP que tienen terminales largas, las cuales se hacen pasar por las perforaciones de la tarjeta y se sueldan en el lado inferior (lado de pistas) de la misma, las terminales de los dispositivos SMT vienen dobladas para que hagan contacto con el lado superior de esta.

Antes de montar estos componentes a la tarjeta, se aplica una pasta especial para soldar, para esto se utiliza una plantilla cuyo patrón de orificios coincide con los puntos de

contacto donde se va a aplicar la soldadura y la terminal del componente. Después se colocan los dispositivos y por último, el ensamble completo se pasa por un horno especial que funde la pasta de soldar.

Con el uso de tecnología de dispositivos SMT aunado a la tecnología de PCB de línea fina, permite un empaquetamiento denso de CI y otros componentes en una tarjeta. Permitiendo con ello un ahorro en espacio, entre otras cosas. Para satisfacer los requerimientos rigurosos de velocidad y densidad, se han desarrollado módulos de chips múltiples (MCM, *multichip modules*) (Wakerly, 2000).

Existe gran variedad de software con los cuales podemos simular el comportamiento de un circuito, cambiarle componentes y valores, hay otros que se emplean para el diseño de tarjetas, en él se dibujan o interconectan componentes de forma gráfica, tal es el caso de Protel. La ventaja que presentan estos programas de diseño, es que dentro de sus funciones, tiene la facilidad de generar un archivo Gerber o formato RS-274; que es un estándar internacional avalado por la EIA, el cual es un formato de datos en donde se especifica toda la información pertinente a un diseño, como son: el diámetro de la perforación, tamaño del componente, grosor de un track, entre otra (Haro et al, 2003).

El circuito impreso cumple una doble función: se usa para interconectar los componentes de un circuito y para sostenerlos físicamente de una manera estable. Los materiales más utilizados para elaborar plaquetas de circuitos impresos son: baquelita y fibra de vidrio. Existen circuitos impresos de una cara, de doble cara y de capas múltiples. A continuación se presenta alternativas o métodos para la elaboración de tarjetas.

2.10.1 Método experimental

Los pasos para la fabricación de un circuito impreso, en forma experimental, son los siguientes:

- Elaborar el diagrama eléctrico del circuito.

- Hacer los trazos del circuito en una hoja de papel para que los componentes queden conectados como lo indica el diagrama.
- Dibujar las trayectorias del diseño, sobre la cara de cobre de la placa, con tinta especial para que sea resistente al ácido o solución rebajadora del cobre.
- Eliminar el cobre sobrante por medio de un baño químico.
- Perforar los orificios para las terminales de los componentes.

2.10.2 Diseño de circuitos impresos

Para cada circuito electrónico en particular, existe un diseño de circuito impreso diferente. Este diseño es un dibujo preciso y ordenado que sirve para trazar los caminos o conductores de cobre sobre el lado del circuito que está recubierto de cobre y para establecer la ubicación de los componentes en el mismo. La elaboración de este diseño es una especie de rompecabezas, donde se deben unir las terminales de los componentes de tal manera que se forme el circuito deseado, utilizando el menor espacio posible, y sin que se crucen las conexiones entre sí.

2.10.3 Reglas para el diseño de circuitos impresos

- Tener a la mano todos los componentes que se necesitan para el proyecto y ubicarlos en forma perpendicular o paralela a los bordes de la placa.
- Determinar cuáles van montados o no en la placa con el fin de asignar terminales que permitan la conexión externa.
- Se deben distribuir uniformemente todos los componentes sobre la superficie de la placa, para evitar aglomeraciones o espacios vacíos.
- Buscar la trayectoria más corta para unir las terminales de los elementos que se unen entre sí, teniendo en cuenta todas las pistas que puedan pasar cerca de este punto, y no impedir que un trazo interfiera con el paso de otro.

2.10.4 Recomendaciones técnicas

- Tener en cuenta que el dibujo del circuito siempre está por debajo de los componentes.
- El ancho de los conductores determina la corriente que puede circular por ellos. Una línea con un ancho de un milímetro soporta aproximadamente un amper.
- La separación mínima entre dos líneas adyacentes debe ser de un milímetro, lo que garantiza un buen aislamiento eléctrico hasta de unos 180 volts, en condiciones normales.
- Los círculos de cobre para la conexión de los componentes deben tener un mínimo de 3 ó 4 milímetros. Entre mayor sea el área de cobre en el punto de conexión, más difícil será su desprendimiento por acción del calor al momento de efectuar la soldadura.
- Los orificios para la inserción de los componentes deben quedar en el centro de las conexiones.
- Se debe disponer de un punto de conexión para cada terminal de los componentes. No se debe conectar dos o más componentes a través de un mismo agujero.
- Procurar no fijar en la placa de circuito impreso elementos que se calienten o que pesen demasiado (resistencias de alto voltaje, transistores de potencia, transformadores de alimentación, entre otros).
- Dejar el espacio para los agujeros por donde habrán de pasar los tornillos para fijar la placa al chasis, si es necesario. La fijación puede ser vertical u horizontal.
- Cuando en el trazado de pistas sea imposible hacer dos puntos para hacer un puente eléctrico mediante un alambre por el lado de componentes.
- Se debe evitar formar ángulos rectos, por problemas de estática.
- Para los componentes que van montados en forma horizontal, se deben colocar los círculos para los terminales a una distancia mayor que el largo total del cuerpo del elemento. Se recomienda dejar aproximadamente 1,5 mm entre el extremo del cuerpo y el punto donde habrá de colocarse la soldadura, con el fin de evitar alteraciones por calentamiento y esfuerzos mecánicos en los terminales de conexión.

Una vez que se tiene el diseño definitivo de la tarjeta, impreso sobre la superficie de cobre, se debe introducir la placa en una preparación de alguna sustancia que retire el

cobre sobrante. A este proceso se le llama “grabado”. Para ensamblar el circuito, se debe perforar los orificios en el centro de los círculos para introducir las terminales de los componentes y soldar por el lado del cobre.

Hay otros procesos o técnicas que nos permiten eliminar el cobre sobrante de la placa, como es el uso de una máquina herramienta semiautomática o automática que permita hacer barrenos de 0.3 mm y una separación mínima de 0.1 mm.

2.10.5 Ensamble de componentes

- Se deben instalar los componentes con el fin de soldar sus terminales por el lado de cobre y así avanzar con seguridad hacia la terminación del proyecto.

- Primero, se instalan las fuentes de alambre, si las hay, con el fin de apoyar la plaqueta sobre la superficie de la mesa de trabajo y que éstos queden bien ajustados sobre el circuito.

- Después se colocan los componentes como los diodos pequeños, las resistencias de 1/4 de watt o elementos similares.

- Luego, los diodos más grandes o las resistencias de ½ y 1 watt, los condensadores electrolíticos acostados o de *tipo axial* y los sockets o bases para circuitos integrados. Después, los condensadores de cerámica, los diodos LED, los condensadores electrolíticos parados o *radiales* y los transistores.

Para soldar tradicionalmente los componentes, se deben seguir cuatro pasos simples:

1. Coloque la punta del cautín en la terminal del componente.
2. Aplique la soldadura.
3. Remueva la soldadura.

4. Remueva el cautín.

Una vez finalizados los pasos anteriores, se debe realizar una prueba del circuito y revisar cada componente por separado para detectar y corregir las posibles fallas que se encuentren.

III. METODOLOGÍA

Las estructuras digitales diseñadas en esta tesis tienen el propósito de llevar a cabo la implementación en hardware de un teclado no matricial que permita efectuar la comunicación con el teclado comercial y la PC. El diseño consiste en el desarrollo de un sistema digital que sea capaz de llevar a cabo el protocolo de comunicación AT. La Figura 3.1 muestra el prototipo del proyecto que consta de 64 teclas y tiene un diseño no matricial para la aplicación de una máquina de inyección de plástico.

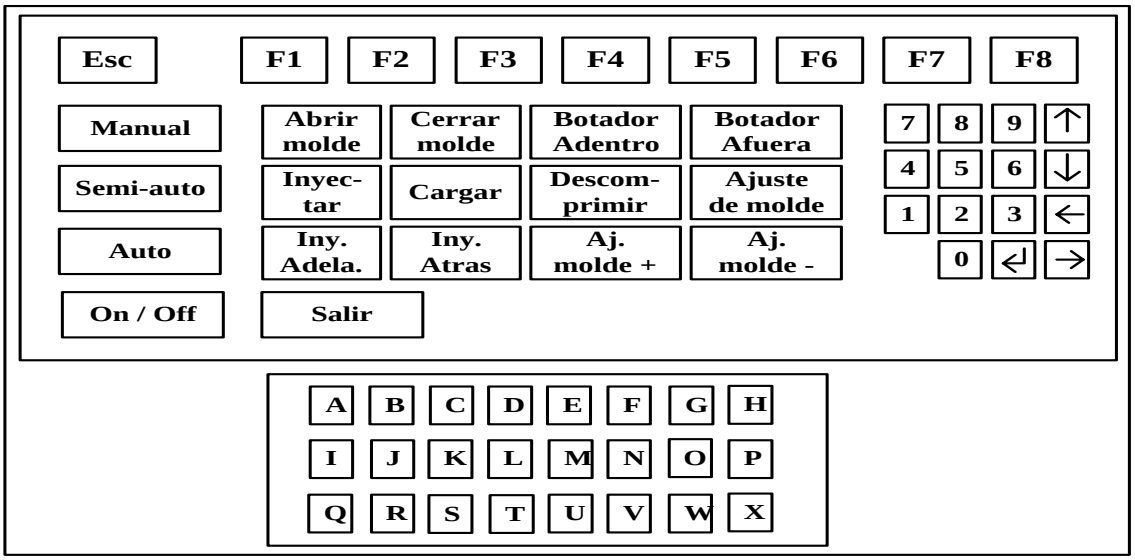

Figura 3.1 Prototipo del teclado.

Para elaborar este prototipo se basó en la configuración que tiene actualmente el teclado de la máquina de inyección con la que cuenta la UAQ (Universidad Autónoma de Querétaro) y esta dividido en cuatro partes fundamentales:

1. *Teclas de función* (F1…F8): Se emplean para acceder a las diferentes pantallas o submenús que se tienen en el programa.
2. *Modos de operación*: Son las formas en las que puede operar la máquina y se tienen tres tipos; manual, semiautomático y automático.

3. *Teclas de operación manual*: Estas teclas permiten hacer ajustes del molde o movimientos de la unidad de inyección.
4. *Teclas alfanuméricas*: Con ellas se puede editar o accesar nuevos parámetros para la secuencia del programa.

El protocolo de comunicación AT es bidireccional, serie y síncrono por lo tanto, para realizar la comunicación entre los dispositivos (Teclado y PC) se emplean solamente dos líneas:

a) Línea de datos: por este medio el teclado recibe y envía información.
b) Línea de reloj: en esta se transmiten los pulsos de reloj para sincronizar el envío o recepción de información.

La Figura 3.2 muestra el diagrama a bloques de manera general empleado para llevar a cabo el protocolo de comunicación entre la PC y el teclado, así como la adquisición de información del teclado comercial.

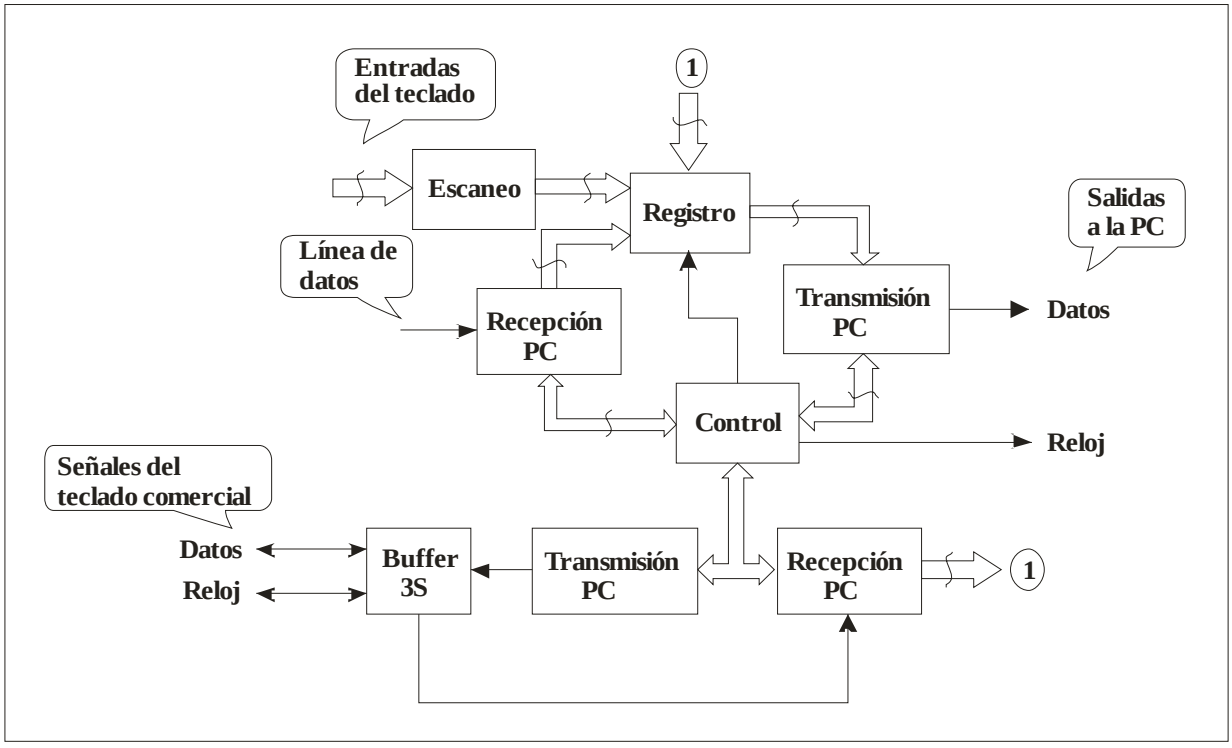

Figura 3.2 Diagrama general para llevar acabo el protocolo AT.

La estructura digital que se muestra en la Figura 3.2 tiene como función principal de llevar a cabo el protocolo de comunicación AT entre el teclado y la PC, tomando en consideración los requerimientos pare este protocolo y debe de ser capaz de adquirir información del teclado comercial o enviarle los comando necesarios dependiendo del proceso de comunicación (transmisión o recepción). Está integrado por diferentes módulos digitales como escaneo, registro, control, transmisión y recepción de la PC, transmisión y recepción del teclado comercial, y un búfer tri-estado.

El bloque de escaneo recopila la información del teclado, dependiendo de la tecla que se activa proporciona un código en hexadecimal y es almacenado en el registro cuando el bloque de control proporciona la señal requerida para este proceso, posteriormente es adquirida por módulo de transmisión PC, el cual coloca bit a bit la información en la terminal *DO* cada vez que el bloque de control le indique.

La estructura de recepción PC obtiene los datos enviados por la PC y de igual manera el código se guarda en el registro. Por otra parte el sistema de control provee de la señal de reloj empleada para llevar a cabo el protocolo y sincronizar los procesos anteriormente mencionados.

Los módulos de búfer tri-estado, recepción TC y transmisión TC se encargan para adquirir o transmitir información al teclado comercial y de igual manera son manipulados por el bloque de control; el búfer tiene la función de conmutar las señales tanto de reloj como datos, mientras la estructura de recepción de TC se encarga de guardar los datos del teclado comercial a través de la línea de datos y los envía al registro o al bloque de transmisión, este último transfiere información correspondiente al teclado.

Para llevar acabo la adquisición de datos del teclado comercial es necesario monitorear las líneas de reloj y de datos, de igual forma para transmitir información al teclado se requiere colocar ambas líneas en un cierto nivel. A continuación se describen de forma más detallada las estructuras digitales empleadas para llevar acabo el protocolo de comunicación AT.

3.1 Escaneo

La Figura 3.3 muestra el módulo general para el escaneo del teclado que tiene como entradas las siguientes terminales que se muestran en la tabla 3.1.

Señal	Descripción
RST	Reset maestro asíncrono activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
En	Habilitador asíncrono activo en alto.
RDF	Señal de entrada síncrona.
KI	Vector de entrada que contiene las señales de las teclas, el tamaño puede variar dependiendo de los requerimientos.
FF	Señal que indica cuando la memoria FIFO se ha llenado, se activa en alto.
EF	Bandera que da la pauta para saber cuando la FIFO ha recibido un dato y se activa en bajo.
Code	Es un vector de 8 bits que conforman el código de la tecla presionada.

Tabla 3.1 Señales del módulo de escaneo.

Con esta estructura digital se monitorea el teclado en un intervalo de 20 ms aproximadamente, proporcionado por el bloque de frecuencia de escaneo, cuando hay una tecla presionada la máquina de estados detiene el incremento de direcciones y habilita el proceso de escritura de la memoria FIFO. Para asignar el código a la tecla activada se tiene un contador que proporciona la posición de la misma, con ello se lee la memoria ROM, la cual contiene los códigos de todas las teclas. Cuando se almacena un código en la memoria FIFO la bandera *EF* se activa y permanecerá así hasta que la señal *RDF* cambia del nivel bajo al nivel alto, al ocurrir esto se activa el proceso de lectura de la memoria FIFO.

Con este arreglo se tiene la capacidad de almacenar información cuando se presiona más de una tecla simultáneamente. El módulo de escaneo a su vez esta compuesto por varios módulos, estos se muestran en el diagrama a bloques de la Figura 3.4.

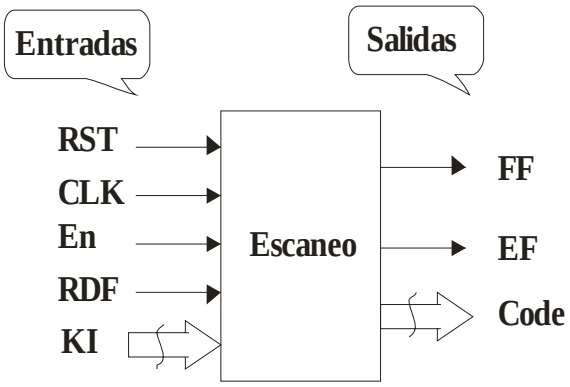

Figura 3.3 Módulo general de escaneo.

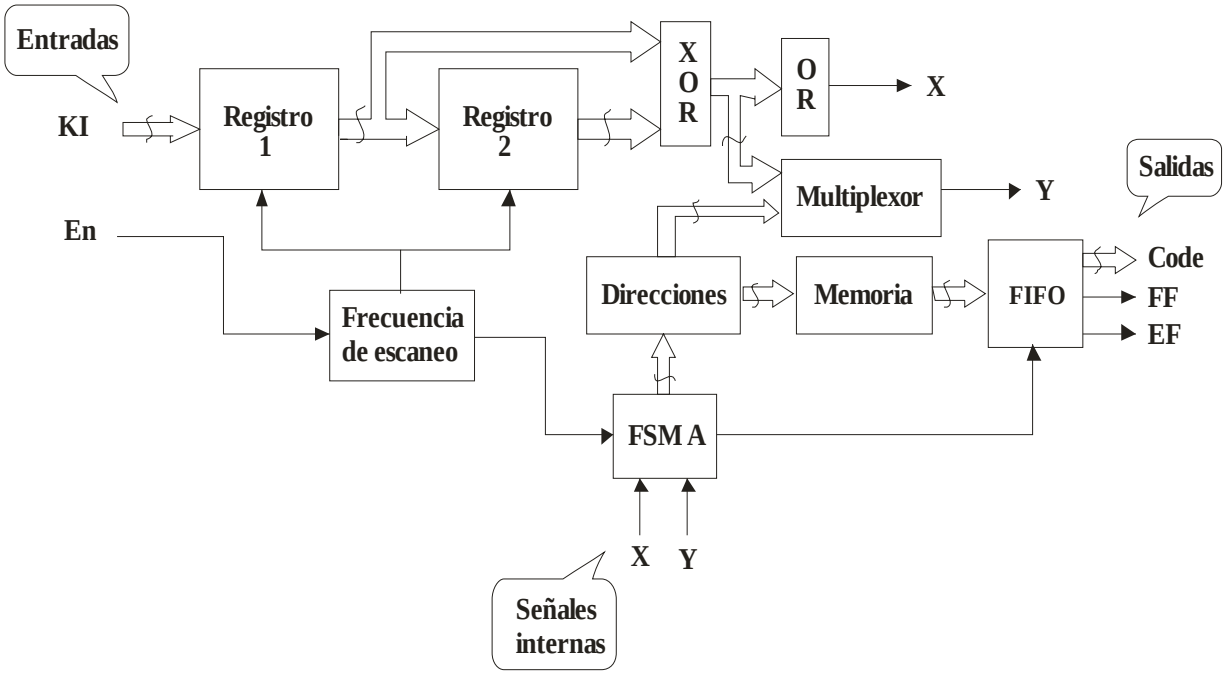

Figura 3.4 Diagrama a bloques para el monitoreo del teclado.

3.1.1 Registro

Es un banco de almacenamiento, pueden ser candados o flip-flops, trabajan en forma síncrona, y son activados por nivel o borde de disparo. Hay dos tipos principales:

a) Paralelos. Se nombran de esta forma debido a que la información se transfiere de forma conjunta.
b) Seriales. Envían los datos bit a bit.

Tomando en cuenta las características que debía tener el proyecto se implemento un registro paralelo, el cual es activado por borde de disparo y lo constituyen las siguientes terminales que se muestran en la Tabla 3.2.

Señal	Descripción
N	Variable con la cual se define la cantidad de entradas necesarias.
RST	Reset maestro asíncrono activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
FS	Habilitador para que cargue el registro el vector KI.
KI	Vector que contiene el estado de las teclas, el tamaño depende de N.
KO	Contiene la adquisición que se realizó al activarse

Tabla 3.2 Terminales del bloque de registro.

Al activarse la bandera *FS* se carga los 2 registros con la información que tienen en la entrada de cada uno, y se almacenan en el vector *KO,* hasta que se vuelva a activar *FS*.

3.1.2 Frecuencia de escaneo

Los circuitos secuenciales son una estructura combinacional que poseen elementos de memoria para almacenar información, entre los cuales se tiene; flip-flops, registros, contadores, entre otros. Los contadores permiten dar una secuencia de datos cíclica, son ascendentes o descendentes y pueden ser síncronos o asíncronos. Para obtener la frecuencia para escanear las terminales del teclado, se utilizó un contador ascendente síncrono, que proporciona un pulso cada 20.97152 ms con una duración de 20 ns, empleando una frecuencia base de 50 MHz. Este bloque tiene las siguientes terminales mencionadas en la Tabla 3.3.

Señal	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
En	Habilitador síncrono activo en alto.
FS	Frecuencia a la cual se desea escanear las líneas del teclado.

Tabla 3.3 Señales empleadas en el módulo de Frecuencia.

3.1.3 FIFO

De las memorias de acceso aleatorio RAM (*Random Access Memory*), las más comunes son las memorias de un solo puerto de acceso lo que implica que la lectura y la escritura se realice a través de una misma línea, necesitando un bus bidireccional. Otro tipo son las memorias que contienen dos puertos, con lo cual se tiene un acceso de entrada o salida independiente.

Existe un segundo grupo de memorias de lectura/escritura y son las memorias del tipo secuencial, se caracterizan de las anteriores porque tiene un control de escritura o lectura; de acuerdo al uso y aplicación se clasifican en:

a) *LIFO* (*Last In First Out,* último que entra es el primero en salir), permite el acceso como si fuera una pila de objetos donde se van colocando uno sobre otro y solo se puede leer el que está arriba.

b) *FIFO* (*Firs In First Out,* primero que entra es el primero que sale), permite almacenar información a una velocidad y extraerla a otra velocidad. Tiene puertos individuales y diversos indicadores de estado.

De acuerdo con las características que ofrecen estas memorias se eligió una memoria secuencial FIFO, debido a que se requiere transmitir los datos conforme se van almacenando.

La estructura digital implementada tiene la capacidad de almacenar 16 bytes, con la opción de incrementar su capacidad al cambiar el valor de la variable interna *Z*. Este módulo consta de las siguientes terminales listadas en la Tabla 3.4.

Señal	Descripción
N	Variable que indica el tamaño del vector de entrada; es decir de cuantos bits es el dato a almacenar.
W	Con esta entrada se define la cantidad de líneas de dirección.
Z	Se emplea para determinar las localidades posibles.
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
WR	Por medio de esta señal se habilita a la memoria para escribir.
RD	Habilitador que permite el proceso de lectura de la memoria.
Din	Dato que se desea guardad.
Dout	Vector que contiene la información cuando se lee la memoria.
EF	Bandera que indica cuando la memoria tiene información almacenadad y se activa en bajo.
FF	Bandera para señalar el momento cuando la memoria se ha llenado y se activa en alto.

Tabla 3.4 Terminales de la memoria FIFO.

3.1.4 Multiplexor

Es un bloque funcional con gran relevancia en electrónica digital. El principio básico es el de permitir el paso selectivo de dos señales hacia una sola salida; tienen mucha importancia por el hecho de que son fáciles de realizar y sus retardos son iguales al de las compuertas genéricas. Son diseñados en escala binaria, es decir; el número de entradas es una potencia exacta de dos.

El multiplexor digital que se implemento es un *MUX 64-1*, se emplea para revisar todas las entradas que se tienen del teclado y está constituido por las siguientes terminales mostradas en la Tabla 3.5.

Señal	Descripción
KI	Vector que contiene la información obtenida al escanear las teclas.
Sel	Selector que permite elegir una de las entradas.
Y	Bandera que toma el valor de la entrada seleccionada.

Tabla 3.5 Terminales del multiplexor de escaneo.

3.1.5 Compuerta xor

Los dispositivos digitales más elementales se conocen con el nombre de compuertas, deben su calificativo por su capacidad de permitir o retardar el flujo de la información. En general, una compuerta tiene una o más entradas y produce una salida que es función del (los) valor(es) de la corriente de entrada, tomando dos valores discretos, 0 y 1. También se conocen como circuitos combinacionales debido a que su salida depende única y exclusivamente de las combinaciones de sus entradas.

Una compuerta AND produce una salida de 1 si sus entradas son iguales a 1; de otro modo genera una señal de 0, mientras que las OR proporcionan un nivel alto si una o más de sus entradas son 1 y cuando están en 0 suministra un 0. La compuerta NOT, más comúnmente conocida como inversora, genera un valor de salida opuesto al valor de entrada. Estas compuertas se pueden combinar para formar otras más complejas, tal es el caso de la compuerta XOR que es un arreglo de las tres compuertas esenciales, produce una salida en 1 si sus entradas son diferentes, en la Figura 3.5 a) se observa la equivalencia en compuertas y en la Figura 3.5 b) se muestra el símbolo.

Se emplea un bloque digital XOR para verificar que tecla(s) se ha(n) activado y es asíncrono. Para llevar a cabo este proceso se toman los vectores que contienen los estados actuales y anteriores de las teclas proporcionados por los registros y se le aplica la operación XOR, y en otro vector se almacena la información obtenida de todas las teclas.

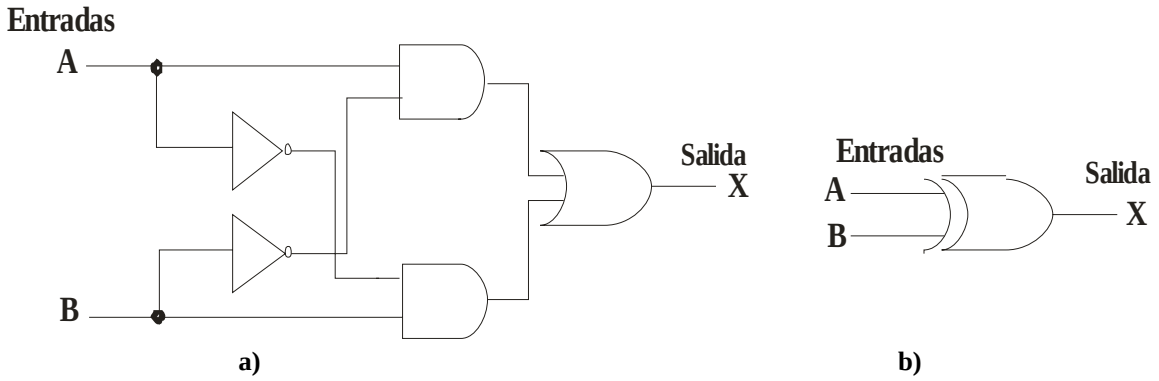

Figura 3.5 Compuerta XOR

3.1.6 Compuerta or

Como se citó anteriormente existen tres compuertas básicas y una de ellas es la compuerta OR. Se empleo un módulo digital de este tipo para verificar si se ha activado una tecla. Como este módulo es asíncrono la señal generada se mantiene hasta que los registros se borren.

3.1.7 Direcciones

Se emplea un contador ascendente síncrono, y la secuencia que proporciona es empleada para seleccionar la entrada en el Muxtiplexor e indicar que dirección de la memoria se desea leer, la cantidad de secuencias depende del número de teclas.

Dependiendo de la combinación binaria de la señal OPC el contador bebe de efectuar varias acciones, cuando es “00” el contador esta en la cuenta cero, con “01” la cuenta permanece igual y en “11” hay un incremento en la cuenta, la Tabla 3.6 muestra el comportamiento de este módulo y en la Tabla 3.7 se observa las señales requeridas del mismo.

Entrada	Estado	Salida
OPC		ADD
00	Cn = “000…0”	“000...0”
01	Cn = Cp	No cambia
11	Cn = Cp +1	Incrementa

Tabla 3.6 Proceso que realiza el módulo de direcciones.

Señal	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
OPC	Señal de entrada síncrona que permite la secuencia del contador. Esta permite borrar la secuencia, mantener el conteo mientras se procesa la información o incrementar.
EOC	Bandera que indica cuando se ha finalizado toda la secuencia.
ADD	Vector que contiene la secuencia generada, el tamaño es igual al número de teclas.

Tabla 3.7 Señales para el bloque de direcciones.

3.1.8 Memoria

Los primeros dispositivos lógicos programables que estuvieron disponibles en el mercado fueron las memorias de solo lectura (*Read Only Memory*, ROM), que son circuitos combinacionales masivos donde cada uno de los minitérminos de una función lógica de n variables es incluido en su estructura. La capacidad esta determinada por el número de variables *n* y la cantidad de funciones lógicas que puede sintetizar en forma simultanea, m. Se determina de acuerdo a la siguiente formula.

$C = 2^n * m$	Ec. 3.1

Para este bloque se empleo una memoria *ROM* digital, la cual contiene los *scan codes*, es decir la información correspondiente de cada una de las teclas, es un arreglo de 8

bits, que posteriormente se almacena en la *FIFO*. Los códigos empleados son los mismos que utilizan los teclados comerciales y sus terminales se listan en la Tabla 3.8.

Señal	Descripción
ADDR	Vector que contiene las direcciones de la memoria ROM.
Code	Almacena el código hexadecimal de la tecla presionada.

Tabla 3.8 Terminales del módulo de memoria.

3.1.9 Máquina de estados A

La Figura 3.6 muestra la secuencia que se requiere para realizar la adquisición de información de manera síncrona y llevar a cabo el proceso de escritura de la memoria FIFO; la combinación binaria entre comillas y la de debajo de esta representan los estados y las salidas de la máquina de estados (*WRF* y *OPC*) respectivamente, mientras que las variables de afuera (*X, Y, EOC,* y *RDY*) muestran las condiciones necesarias para efectuar un cambio de estado o que permanezcan en el mismo; con respecto a las salidas los dos primeros bits son asignados a la señal *OPC* y el tercer bit a la señal *WRF*. En la Tabla 3.9 se describen las terminales de la máquina de estados.

Esta estructura se encarga de controlar la secuencia de lectura de la memoria *ROM* y el almacenamiento del *scan code* en la *FIFO*, se auxilia de la señal proporcionada por el multiplexor y de la compuerta *OR*. Proporciona la señal *OPC* empleada por el bloque antes mencionado y el habilitador para el proceso de escritura en la memoria *FIFO.* Para llevar a cabo el proceso de lectura y escritura de la FIFO es necesario que ocurra un pulso de muestreo.

Señal	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
FS	Habilitador, se activa en alto.
EOC	Bandera para indicar cuando el bloque de direcciones ha terminado.
X	Señal interna proporcionada por la compuerta OR.
Y	Señal interna generada por el multiplexor.
RDY	Bandera que indica cuando se ha terminado de escribir el dato.
WRF	Habilitador para el proceso de escritura de la memoria FIFO.
OPC	Permisivo para el módulo de direcciones.

Tabla 3.9 Terminales de la máquina de estados A.

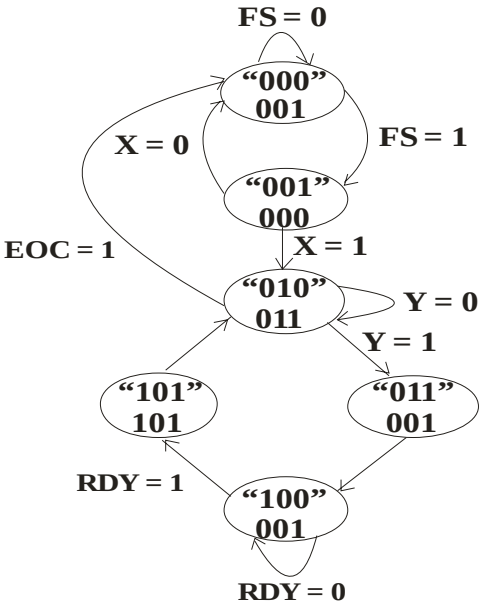

Figura 3.6 Secuencia de la máquina de estados A.

3.2 Control

El protocolo de comunicación que se empleó es el AT, se lleva a cabo en dos partes; una es Teclado-PC y la otra PC-Teclado para realizar este proceso se empleó la

estructura, que se muestra en la Figura 3.7. Al igual que los demás módulos esta estructura esta compuesta por otros componentes, la Figura 3.8 plasma los elementos de este bloque.

Este módulo controla la secuencia de operaciones como recepción, transmisión, e inicialización del teclado, proporciona la señal de reloj, el habilitador para leer la información que se almacena en el módulo digital escaneo, además genera la señal del registro, el selector del multiplexor y el habilitador del acumulador y del buffer tri-estado.

Al momento de energizar el equipo, el bloque de control es capaz de llevar acabo la comunicación con la PC, es decir; debe de llevar la secuencia de comandos para las instrucciones que son enviadas por la PC hacia el teclado, de igual forma genera las señales que permite al teclado comercial poder enviar o recibir información.

En el protocolo de comunicación empleado el teclado es responsable de generar la señal de reloj para la comunicación entre los dispositivos, por lo tanto la estructura digital de control proporciona dicha señal empleando un contador incremental. Para lleva a cabo la comunicación con el teclado comercial, se toma la señal de reloj proporcionada por este último y el módulo de control manipula dicha señal para llevar a cabo el proceso de transmisión o recepción. Las señales que conforman este sistema se describen en la Tabla 3.10

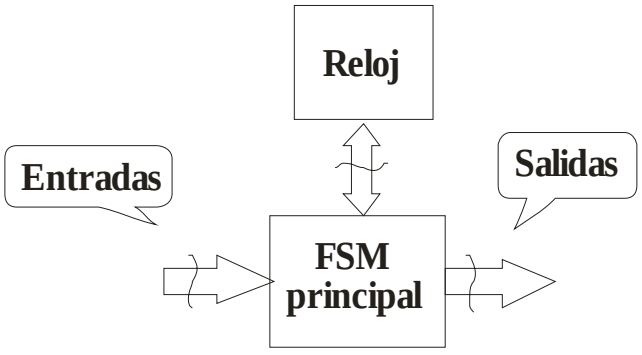

Figura 3.8 Diagrama a bloques del módulo de control.

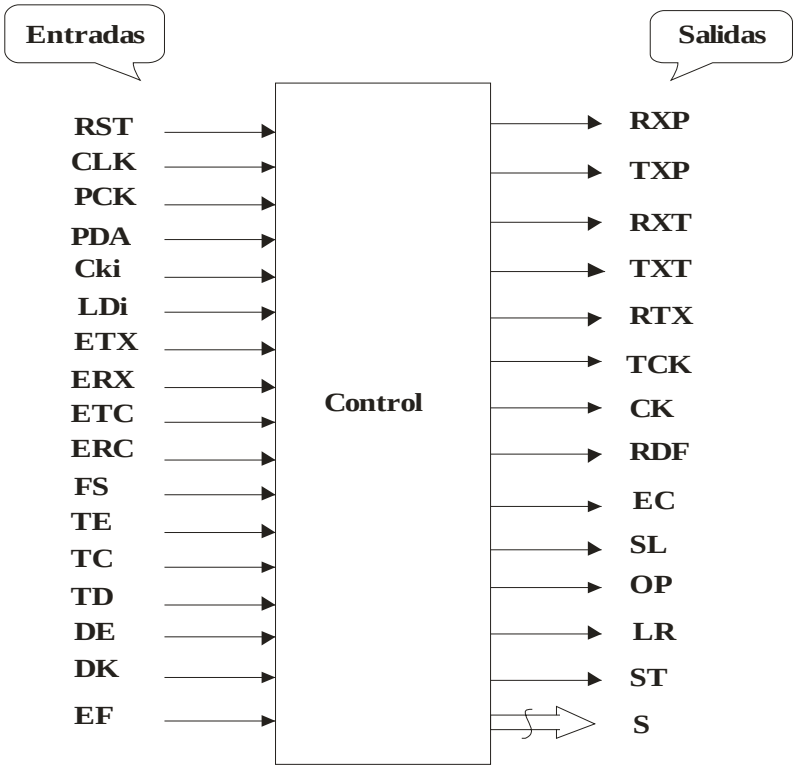

Figura 3.7 Módulo digital de control.

3.2.1 Reloj

De acuerdo con la frecuencia empleada en el protocolo que es de 20 a 50 KHz, se estableció un bloque para proporcionar el tiempo necesario para llevar a cabo la comunicación entre los dispositivos. Consiste en un contador incremental síncrono, genera un pulso en un intervalo de tiempo de 20.48 μs y otro cada 40.96 μs ambos con una duración de 20 ns.

3.2.2 Máquina de estados principal

Este módulo se encarga de realizar todas las secuencias de control para llevar a cabo la inicialización del teclado, monitoreo de líneas, transmisión y la recepción. En la

inicialización, el teclado empieza la comunicación enviando HXAA y espera que la PC responda, la Tabla 3.11 se muestra la secuencia que se realiza en este proceso.

Señal	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
FS	Señal síncrona que habilita la máquina de estados.
EF	Bandera activa en alto cuando hay un dato en la memoria FIFO.
ETX	Proceso de transmisión a la PC ha finalizado.
ERX	Indica que el proceso de recepción ha finalizado entre la PC.
ETC	Señala cuando se ha terminado el proceso de transmisión hacia el teclado comercial.
ERC	Indica el fin de la recepción entre el teclado comercial.
PCK	Línea de reloj que comunica al teclado con la PC.
PDA	Línea de datos que comunita al teclado con la PC.
CKi	Línea de reloj del teclado comercial.
LDi	Línea de datos del teclado comercial.
DK	Indica que se recibió un comando por parte de la PC y se tiene que responder con HXFA (código en hexadecimal) o el correspondiente al comando recibido.
DE	Se activa cuando hay una instrucción de error debido a una mala transmisión.
TC	Cuando se tiene que contestar al teclado comercial al comando enviado esta bandera se activa.
TE	Si se realizó una mala transmisión al teclado comercial esta se activa.
TD	Indica que hay información para enviar a la PC.
TCK	Señal que permite inhibir la comunicación del teclado comercial.
CK	Señal de reloj necesaria para llevar acabo el proceso de transmisión o recepción.
TXP	Habilitador para el proceso de transmisión entre el teclado y la PC.
TXT	Bandera que permite realizar el proceso de transmisión hacia el teclado comercial.
RXP	Habilita el proceso de recepción entre el teclado y la PC.
RXT	Autoriza la recepción de información del teclado comercial.
ST y SL	Selectores para el multiplexor.
OP / LR	Habilitadores para los acumuladores.
RDF	Bandera que permite leer la memoria FIFO, se activa en alto.
OE/OE1	Habilitadores para los buffer tri-estado, se activan en alto.
EC	Habilitador para el módulo de reloj.
S	Vector de 2 bits, empleado para el registro.

Tabla 3.10 Señales que conforman el módulo de control.

Dispositivo	Comando	Acción
Teclado	HXAA	Prueba correcta de la verificación del procesador.
PC	HXED	Prende y/o apaga los indicadores de estado.
Teclado	HXFA	Reconocimiento.
PC	HX00	Apaga todos los leds.
Teclado	HXFA	Reconocimiento.
PC	HXF2	Identificador del teclado.
Teclado	HXFA	Reconocimiento.
Teclado	HXAB	Primer byte de la identificación.
Teclado	HX83	Segundo byte.
PC	HXED	Prende y/o apaga los indicadores de estado.
Teclado	HXFA	Reconocimiento.
PC	HX02	Enciende el led de Num Lock.
Teclado	HXFA	Reconocimiento.

Tabla 3.11 Secuencia de inicialización del teclado.

Al momento de comenzar la recepción la línea *PCK* permanece en nivel bajo por lo menos 60 μs, para evitar que el teclado quiera transmitir al mismo tiempo después de este intervalo se baja la línea de datos y se libera *PCK*, en este momento el teclado genera la señal de reloj y la PC coloca los bits en la línea *PDA*. La PC coloca los bits en *PDA* en el pulso alto de la señal de reloj, y estos son leídos en el pulso bajo por el teclado. Después del bit de paridad, la PC libera la *PDA* y espera a que el teclado le envíe otro ciclo de reloj y bajar la *PDA* para reconocer la recepción.

Para el proceso de transmisión, primero se revisa que *PCK* este en nivel alto por lo menos 50 μs, antes de empezar este evento. El teclado coloca el bit en *PDA* a la mitad del pulso alto es por eso que el módulo anterior genera dos señales y es leído por la PC en el pulso bajo. La PC puede inhibir la transmisión en cualquier momento manteniendo *PCK* en nivel bajo por lo menos 100 μs.

En la secuencia de recepción de información del teclado comercial se inspecciona que la línea *LDi* este en nivel bajo y se toma el dato cuando la señal de reloj este en bajo, se

continua adquiriendo los datos enviados hasta completar la trama de la palabra. Para la transmisión al teclado comercial, es necesario mantener en bajo la señal interna *TCK* por lo menos 60 μs, después de este intervalo se coloca el bit de inicio en la terminal *TLD* y se activa *TCK*. Se van colocando los bits del dato a enviar cuando la señal de reloj este en bajo.

La Figura 3.9 muestra los módulos digitales que se implementaron para realizar los procesos antes mencionados. Este también genera la señal de reloj que se emplea para realizar el envío y recepción de información, se auxilia de las señales que proporciona el bloque antes mencionado.

En el estado de monitoreo se concentran las señales de reloj (PC y teclado comercial), datos (PC y teclado comercial), y las banderas que indican cuando hay un comando o un dato a procesar (*DE, DK, TC, TE,* y *TD*). Dependiendo de la señal o bandera que se activa se realiza el proceso de escritura o lectura, dando prioridad a las líneas de la PC.

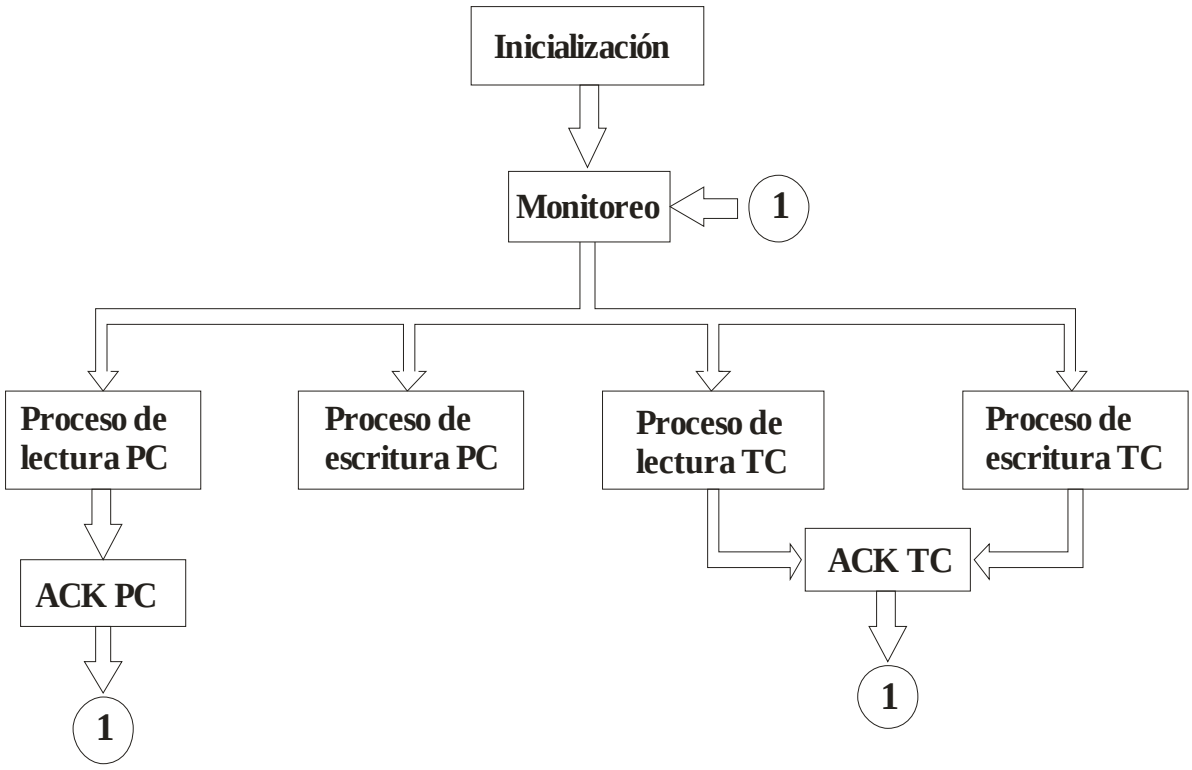

Figura 3.9 Diagrama a bloques de la máquina de estados principal.

3.3 Recepción PC

Como se mencionó en el capítulo II, la comunicación entre la PC y el teclado es de manera síncrona, para ello se generó una estructura digital para llevar a cabo la recepción de información, en el diagrama a bloques de la Figura 3.10 se observa el módulo digital principal, las terminales que conforman este módulo se muestran en la Tabla 3.12.

El papel que desempeña este módulo es la de recibir y almacenar temporalmente la información que es enviada de la PC al teclado, verificar si es correcto el envío, proporcionar el comando adquirido e indicar por medio de sus banderas el proceso que se requiere hacer con el comando.

Para efectuar el proceso de lectura, el módulo de control monitorea la línea de datos y reloj (*PDA, PCK* respectivamente), la computadora primero coloca en nivel bajo la *PCK*, con ello evita que el teclado envíe al mismo tiempo, después de 60 μs libera *PCK* y bajo la *PDA*, cuando ocurre esto el módulo de control debe de generar la señal de reloj hasta que termine recibir todos los bits de la palabra. Para realizar este módulo se emplearon otros elementos los cuales se muestran en la Figura 3.11.

Señal	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
RX	Habilitador para iniciar el proceso de recepción.
PDA	Línea de datos que comunica con la PC.
ETX	Bandera que se activa cuando se ha finalizado el proceso de transmisión y con ello permite adquirir más información.
ERX	Señala el fin de transmisión.
DK	Esta señal se activa cuando se ha recibido un comando y se debe de responder a la PC.
DE	Se activa cuando se tiene un comando de error.
DPC	Vector que contiene la información recibida.

Tabla 3.12 Señales principales para el módulo de recepción.

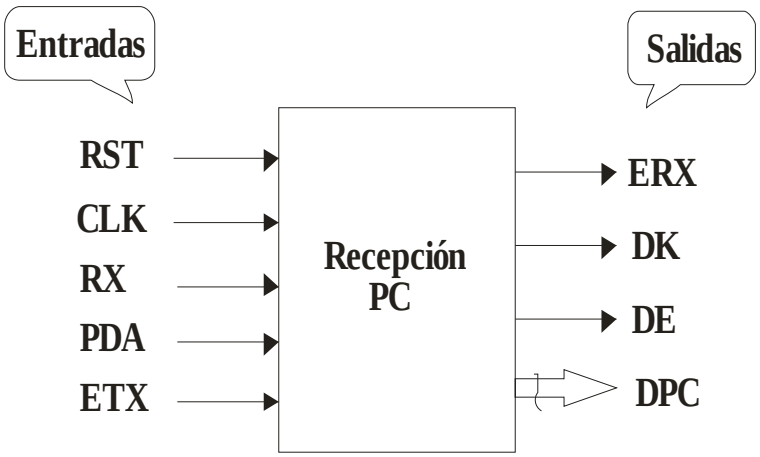

Figura 3.10 Módulo general para la recepción.

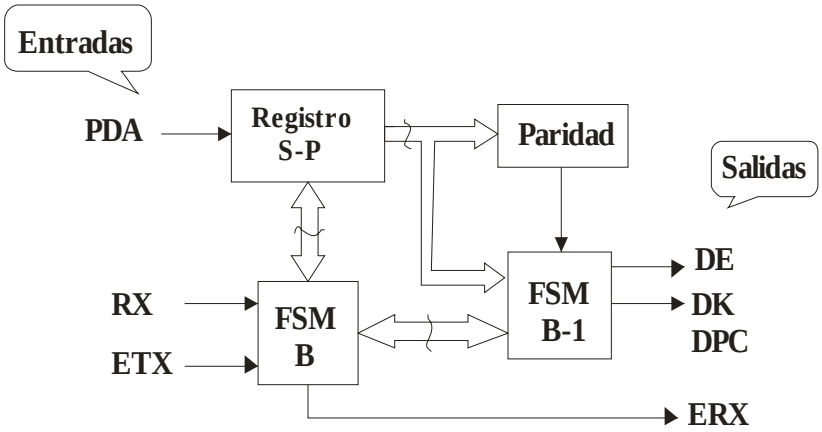

Figura 3.11 Diagrama a bloques para la recepción de información.

3.3.1 Paridad PC

La paridad de una palabra binaria se define como la cantidad unos o ceros que contiene. Si el número de unos es par entonces la paridad es par positiva, si es impar se tiene una impar positiva. Por otra parte, cuando la cantidad de ceros es par negativa y se es impar se tiene impar negativa.

Los circuitos verificadores de paridad se emplean en sistemas de comunicación y transmisión de datos para revisar el correcto envío de la información. De manera

simultánea se envía el dato con su respectiva paridad, si en el receptor la paridad del dato no concuerda con la paridad enviada, entonces se genera un error de paridad.

Para el protocolo empleado es necesario utilizar la paridad positiva, cuando la cantidad de unos es par el bit de paridad es 1 y cuando es impar es 0. Por lo tanto a la salida del circuito de paridad se aplica una compuerta NOT.

De acuerdo a lo antes mencionado se realizó un módulo digital que estructura XOR para determinar el bit de paridad, contiene las siguientes terminales mostradas en la Tabla 3.13. El proceso consiste en leer la información del vector *DPC* y en una señal auxiliar se guarda los 8 bits de la palabra (del bit 8 al 1), posteriormente se aplica la operación XOR a los datos y por último se compara el bit obtenido con el bit de paridad enviado (bit 9), si son iguales se activa la bandera *OKP,* de lo contrario permanece en nivel bajo.

Señal	Descripción
DPC	Vector que contiene la información recibida, consiste en 11 bits.
OKP	Bandera que indica si la recepción de datos fue la correcta.

Tabla 3.13 Señales para el módulo de paridad.

3.3.2 Registro serie-paralelo

Los registros pueden tener una combinación de modos de operación de acuerdo con la forma de entrada de datos y su salida. Así se tiene las siguientes combinaciones:

a) Paralelo-paralelo.
b) Paralelo-serie.
c) Serie-paralelo.
d) Serie-serie.
e) Universales.

Para el proceso de recepción se empleó un registro serie-paralelo debido al tipo de protocolo empleado. El objetivo que tiene este bloque es adquirir la información que envía la PC al teclado, y la almacena en un vector de 11 bits. Las terminales de este módulo se listan en la Tabla 3.14.

Como la comunicación empieza con un bit de inicio que siempre es cero y el teclado lee la información cuando la señal de reloj esta en el semiciclo negativo entonces se requiere inicializar el vector *DPC* con unos y dependiendo de la combinación binaria de la señal *ST* realizar determinado proceso. Cuando se tiene la combinación "00" el vector de salida *DPC* se carga con unos, con "10" la salida no cambia y en "11" se adquiere el bit de la línea *LDi* y se almacena en *DPCK*, en la Tabla 3.15 se muestra esta secuencia.

Señales	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
PDA	Línea de datos.
ST	Señal que permite adquirir información de la línea de datos y consiste en un vector de dos bits.
CRX	Bandera empleada para indicar cuando se ha terminado de recibir la información.
DPC	Vector que almacena la información.

Tabla 3.14 Terminales del módulo de registro serie-paralelo.

Entrada	Estado	Salida
ST		DPC
00	Qn = "000…0"	"000...0"
10	Qn = Qp	No cambia
11	Qn (10) = LDi Qn (i) = Qp (i+1)	Adquiere el bit de la línea LDi y hace un corrimiento a la izquierda

Tabla 3.15 Proceso que realiza el módulo de registro serie-paralelo.

3.3.3 Máquina de estados B

Esta estructura digital se encarga de sincronizar el momento adecuado para leer la información de la línea de datos de la computadora, para ello es necesario que se active la bandera *RX,* cuando se termina la adquisición de datos genera la señal *ERX* que indica el fin de la recepción; en ese momento se activa la máquina de estado B-1.

3.3.4 Máquina de estados B-1

Esta máquina de estados es la encargada de verificar el dato enviado por la PC, de acuerdo con el código leído se proporciona el comando para contestarle a la computadora, el cual se coloca en la terminal de *DPC* y lo almacena temporalmente hasta que se halla procesado. Se auxilia de la bandera generada por el módulo de parida para poder determinar una acción correspondiente.

3.4 Transmisión

La Figura 3.12 muestra el esquema general del módulo transmisión, y las terminales que conforman este bloque se muestran en la Tabla 3.16.

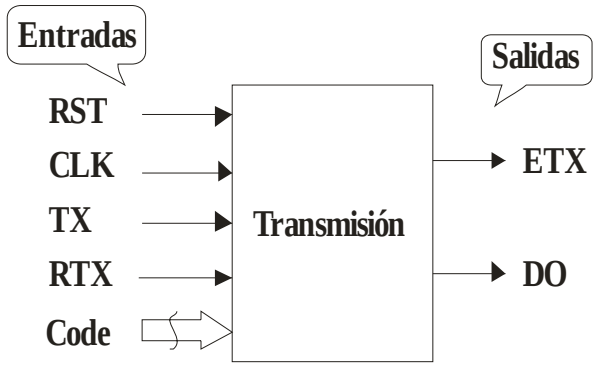

Figura 3.12 Bloque general para la transmisión.

Señal	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
TX	Habilita inicia el proceso de transmisición.
RTX	Resetea el proceso de transmisión.
DI	Vector que contiene el código a enviar.
ETX	Bandera que indica cuando ha terminado el envío.
DO	Terminal en la cual se colocan los bits del dato.

Tabla 3.16 Terminales del módulo de transmisión.

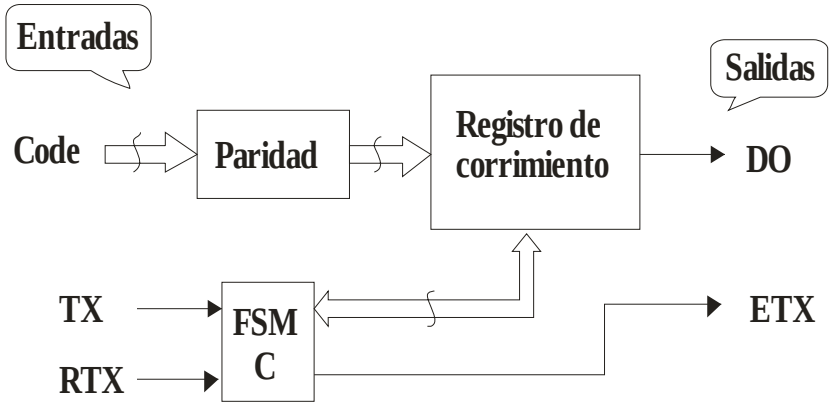

Figura 3.13 Estructuras digitales para la transmisión.

En el proceso de transferencia de información a la PC, el bloque de transmisión la toma del vector *Code* y concatena los bits de inicio, paridad y stop, después por medio de un registro de corrimiento coloca en la terminal *DO* cada uno de los bits hasta terminal con la trama. La etapa de desplazamiento de bits es controlada por una máquina de estados que depende de la activación de la bandera *TX* y una vez que haya terminado el envío de información prende la bandera *ETX*. Se emplea de igual forma para realizar el envío de datos al teclado comercial, con la diferencia de que la información se adquiere del módulo de Recepción TC.

3.4.1 Paridad

Este módulo consiste en una estructura XOR, para determinar el bit de paridad. Toma la información del vector *Code* que consta de 8 bits e internamente se concatena a la información el bit de inicio siempre es 0, bit de paridad, y el de stop que es 1. Proporcionando a la salida un arreglo de 11 bits.

3.4.2 Registro de corrimiento

Es un registro de *n* bits con una disposición para recorrer los datos almacenados por una posición de bit en cada pulso de reloj. Esta estructura toma la señal proporcionada por el módulo anterior, el corrimiento es controlado por medio de la señal *OP*. Las terminales de este bloque se describen en la Tabla 3.17.

Señal	Función
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
OP	Habilitador que permite realizar el recorrido u otra secuencia, y consta de dos bits.
Data	Vector de entrada que contiene la información para ser enviada, consta en un arreglo de 11 bits.
EOC	Bandera que indica el fin del corrimiento.
DO	Salida donde se colocan los bits.

Tabla 3.17 Señales del registro de corrimiento.

3.5 Registro Paralelo

Este módulo tiene la función de almacenar el *scan code* proporcionado por la salida de escaneo o guardar el comando proveniente de la recepción. Cambia la información hasta que se finalice la transmisión del código previamente recopilado. Cuando no hay actividad de ninguno de los dos módulos el vector de salida se carga con unos. Las terminales que constituyen este bloque se listan en la Tabla 3.18.

Señal	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
DPC	Vector que proporciona el comando recibido.
DTC	Vector que almacena el *scan code* de la tecla presionada.
DTC1	Contiene el *scan code* del teclado comercial.
S	Vector de dos bits, de acuerdo a la combinación se guarda cierta información para ser procesada.
Data	Almacena la información seleccionada.

Tabla 3.18 Terminales del bloque de registro.

3.6 Acumulador

La función principal que tiene esta bloque digital es la de almacenar la información por si hay un error en la transmisión. Recoge el código de la salida del registro y la mantiene hasta que la PC mande el comando correspondiente y procesarlo. A continuación se menciona sus terminales en la Tabla 3.19.

Señal	Descripción
RST	Reset asíncrono general activo en bajo.
CLK	Reloj maestro, flanco positivo con una frecuencia de 50 MHz.
Data	Contiene la información del teclado comercial, la PC o el teclado diseñado.
OP	Vector que permite al acumulador cargar, mantener o borrar información.
Dac	Almacena la información.

Tabla 3.19 Señales del Acumulador.

3.7 Multiplexor

Este bloque selecciona la información que se va ha enviar, la Tabla 3.20 se describe las terminales que lo conforman, y en la Figura 3.14 representa la estructura digital del multiplexor. Cuando la bandera *Sel* se activa el vector de salida toma el valor de *DAc* de lo contrario se le asigna lo que tiene *Code_in*.

Señal	Descripción
Code_in	Vector que contiene la información de la tecla presionada o del comando adquirido.
DAc	Contiene la información acumulada y consta de 8 bits.
Sel	Bandera que permite la selección de una de las dos entradas.
Code_out	Vector que toma el valor de una de las dos entradas.

Tabla 3.20 Terminales del multiplexor.

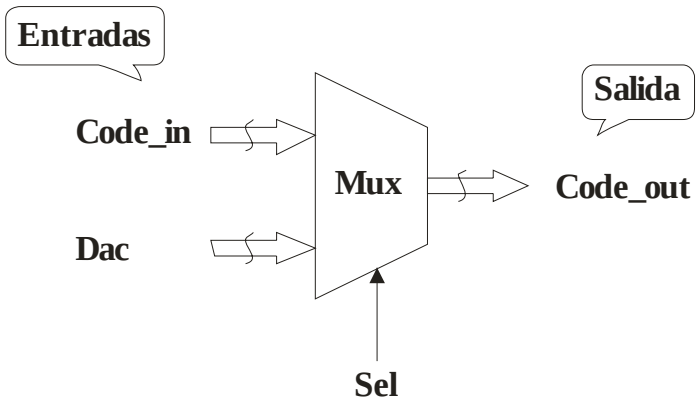

Figura 3.14 Bloque del multiplexor.

3.8 Recepción del teclado comercial

La estructura digital de recepción del teclado comercial se encarga de adquirir la información que envía este dispositivo y proporciona el código de la tecla o del comando a transmitir. La Figura 3.15 se muestra el bloque general, en tanto la Figura 3.16 se detalla los componentes que lo integran.

Para llevar a cabo la adquisición de información el módulo de recepción toma los bits de la terminal *LDi*, los almacena en un vector por medio de un registro serie-paralelo, posteriormente se guardar por un mayor tiempo en un registro paralelo-paralelo, se determina el bit de paridad del código y dependiendo de este bit y del código se activa una de las banderas (*TE, TD, o TC*), el dato o comando correspondiente al código adquirido se coloca en el vector *DTC*.

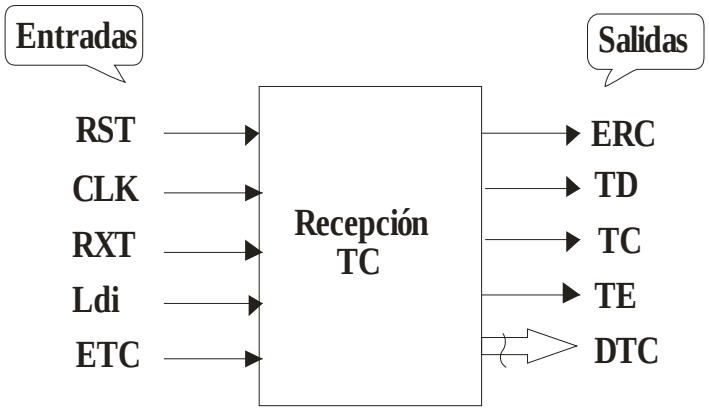

Figura 3.15 Módulo general para la recepción de información del teclado comercial.

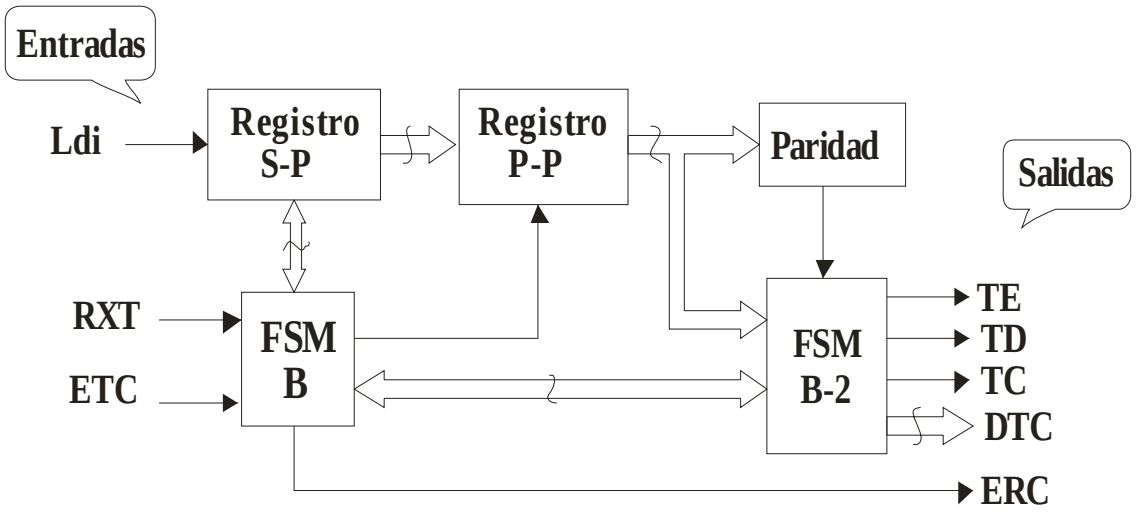

Figura 3.16 Diagrama a bloques del módulo de recepción del teclado comercial.

3.9 Búfer tri-estado

Esta estructura se utiliza en el proceso de transmisión o recepción de información del teclado comercial, dependiendo del proceso su función es conmutar la línea de reloj o datos por medio de los habilitadores *OE* y *OE_1*. La Figura 3.19 muestra el bloque digital del buffer tri-estado y en la Tabla 3.21 se describen sus terminales que lo conforman.

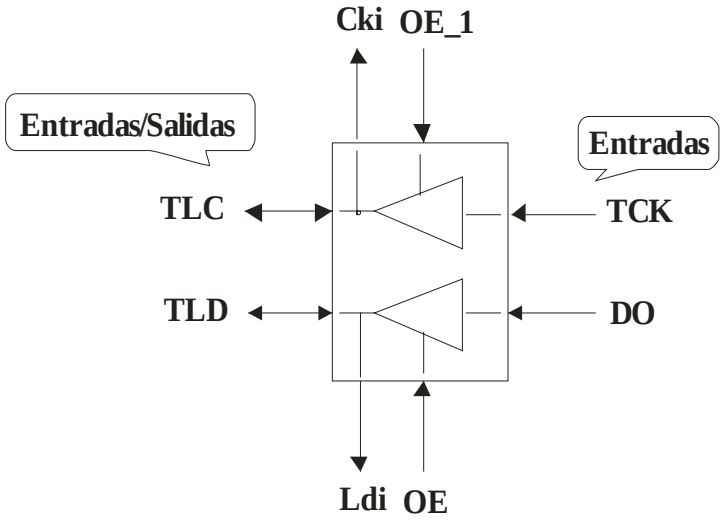

Figura 3.17 Búfer tri-estado.

Señal	Descripción
TLC	Línea bidireccional de reloj del teclado comercial.
TLD	Señal bidireccional de dataos del teclado comercial.
TCK	Señal que permite bloquear la línea de reloj.
DO	Contiene la información del comando a enviar.
OE	Habilitador que permite pasar el bit de DO cuando esta en nivel alto.
OE_1	Cuando se activa la información que se encuentra en TCK pasa a TLC.
LDi	Variable que adquiere el valor de TLD en todo momento.
CKi	Contiene la señal de reloj.

Tabla 3.21 Terminales que conforman el buffer tri-estado.

IV. RESULTADOS

En esta sección se presentan los resultados obtenidos para el teclado industrial desarrollado en este trabajo, el cuál se probó en tres diferentes computadoras. Por lo tanto, se presentan las simulaciones de los diferentes módulos digitales que se programador bajo el software de Active HDL 3.6, este tiene herramientas para realizar simulaciones, lo que ayuda a comprobar las estructuras diseñadas.

4.1 Comunicación entre el teclado comercial y la PC

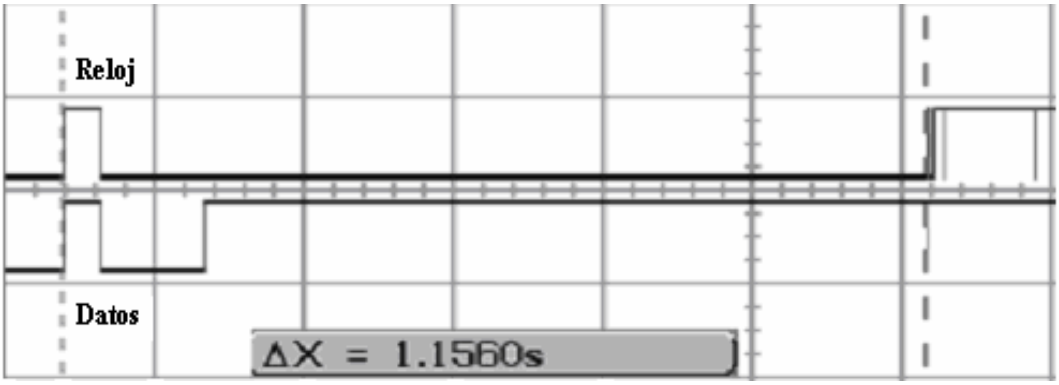

Figura 4.1 Comunicación entre el teclado comercial y la PC.

La Figura 4.1 se muestran las señales generadas al momento de prender la computadora, inhibe la comunicación con el teclado durante 1.1560 s aproximadamente, después de ese tiempo, libera la línea de reloj entonces, el teclado debe de enviar el comando AA, para indicar que la prueba de auto-test fue correcta.

La Figura 4.2 se observa la frecuencia empleada para la comunicación entre los dispositivos, cabe mencionar que está la proporciona el teclado y se usa tanto para la transmisión o recepción de datos.

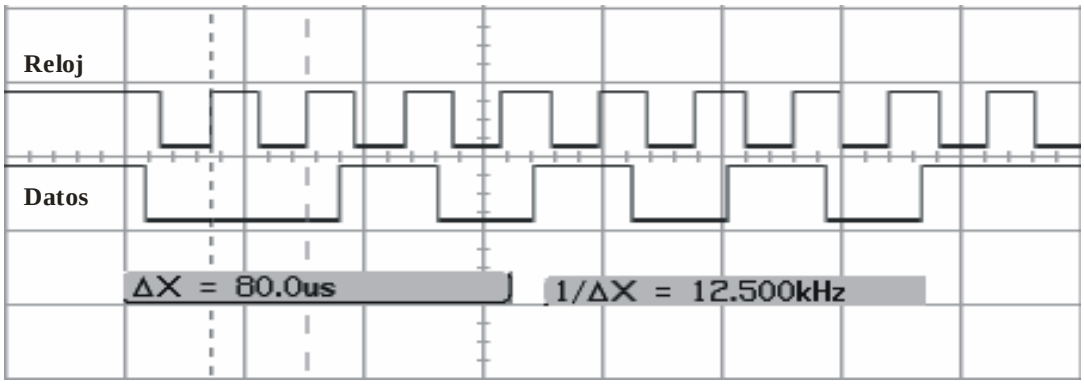

Figura 4.2 Frecuencia de la señal de reloj.

4.2 Módulo de escaneo

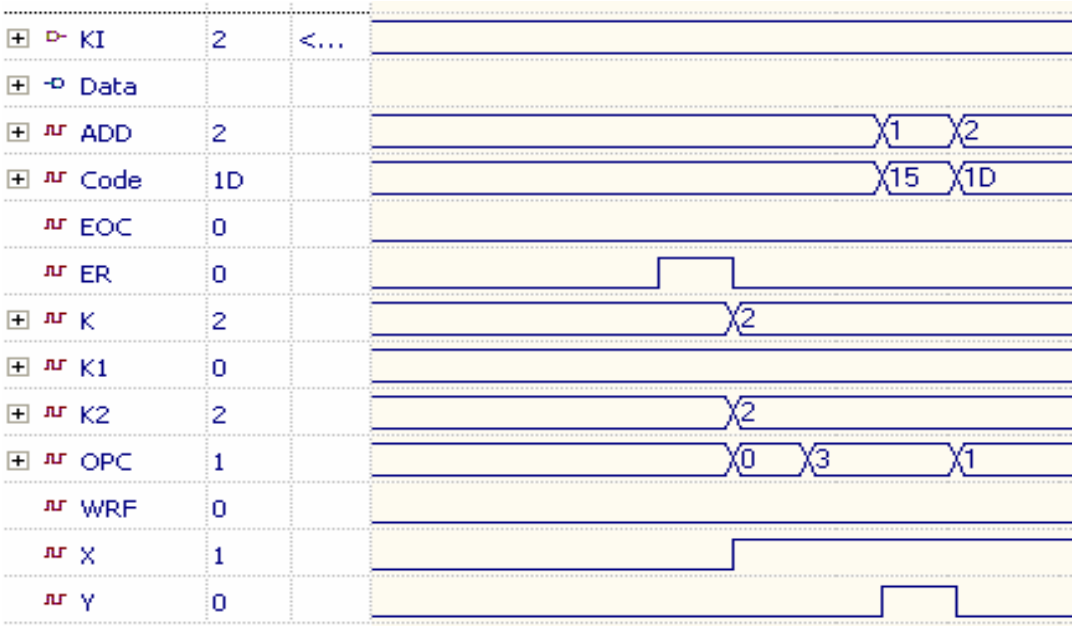

Figura 4.3 Señales internas del módulo de Escaneo.

Cuando se presiona una tecla y además ha transcurrido el tiempo de muestreo (*ER*), se adquiere la información de las teclas activadas que se guardan en un vector (*K2*) en el mismo instante la compuerta OR genera la señal indicando un cambio en el vector de entradas con ello se habilita el contador. Al momento de cambiar *Y* el conteo se detiene y se

almacena el código respectivo. Para efectuar esta operación es necesario subir *RDYTX*, lo que indica que se ha finalizado la lectura de la memoria, posteriormente cambia *WRF*, habilitando la escritura de la *FIFO* al ocurrir esto cambia de nivel *E* (señala que hay un dato guardado).

Los datos adquiridos se van almacenando en la *FIFO* y pasan al registro al momento de que Control proporcione la señal de *RDF*. Al vaciar toda la información guardada se activa de nuevo *E*. La Figura 4.4 se observa el proceso de escritura y lectura de la *FIFO*.

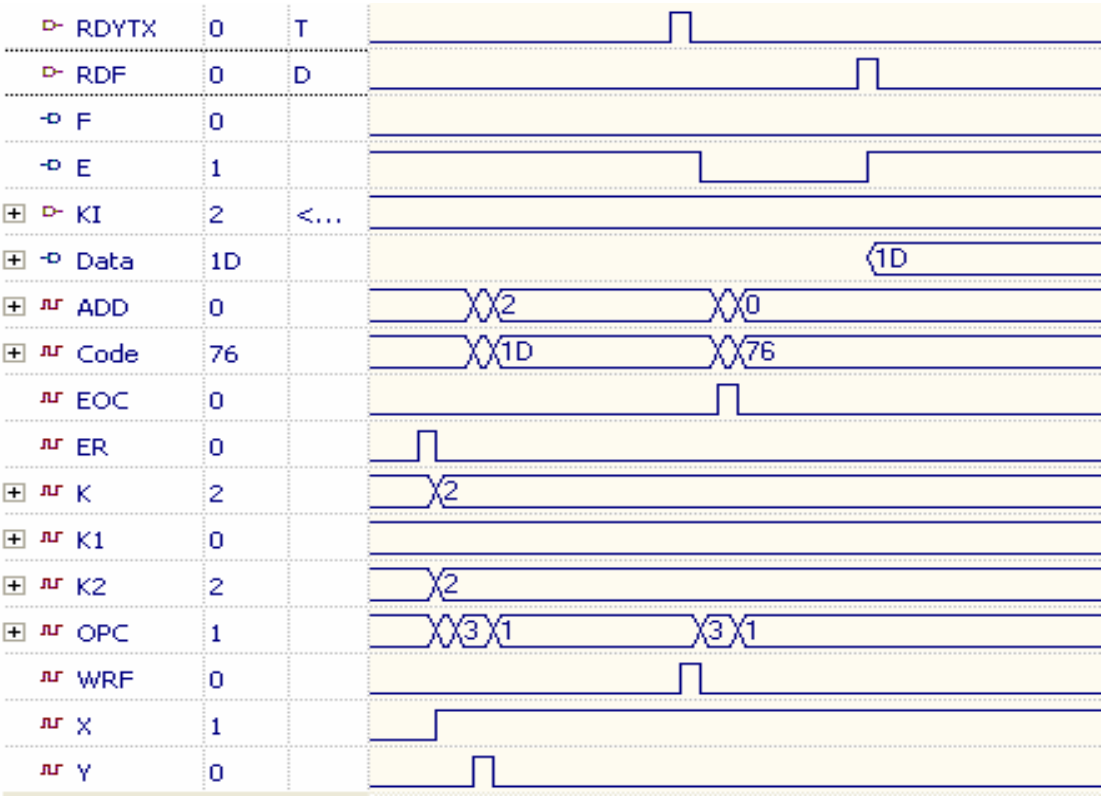

Figura 4.4 Secuencia de escritura y lectura de la FIFO.

4.3 Transmisión

Una vez que se tenga el código de una tecla o la respuesta al comando enviado por la PC es necesario transmitir, en la Figura 4.5 se observa las señales internas y externas requeridas en el envío de información.

Como se mencionó en el capítulo anterior, para procesar la información es necesario añadir 3 bits más (bit de inicio, paridad y stop) y se almacenan en un vector (*iDO*). La señal que habilita el proceso de transmisión es *TX*, está dura 20ns, y cada vez que se activa se coloca un bit en la salida *DO*.

Al momento de activarse *ETX* (fin de transmisión) el proceso se detiene y espera a que se habilite de nuevo. La Figura 4.6 muestra en que instante se activa dicha señal. La bandera *RTX* se emplea solo cuando se recibe información y su función es interrumpir el envío de datos.

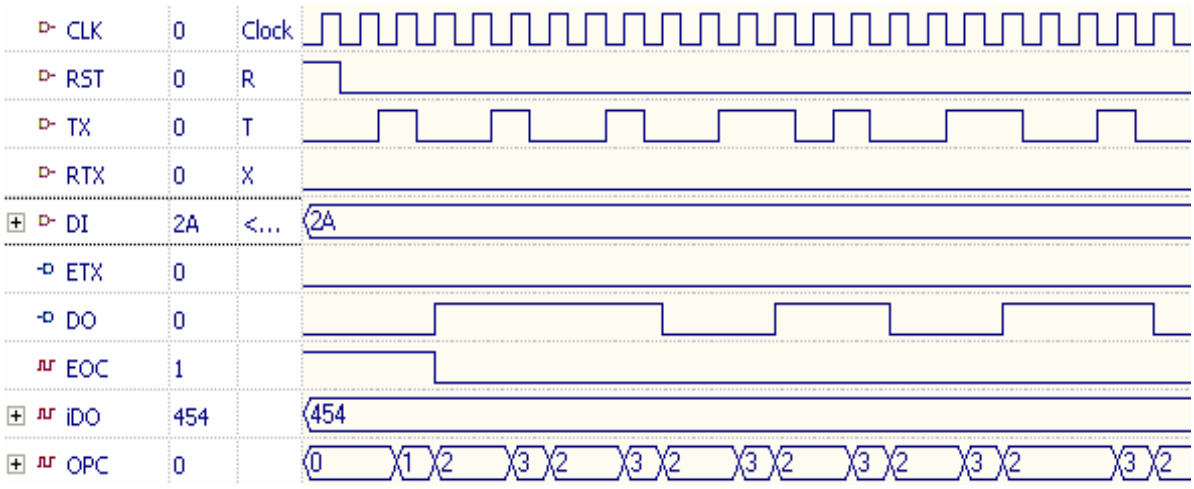

Figura 4.5 Proceso de Transmisión

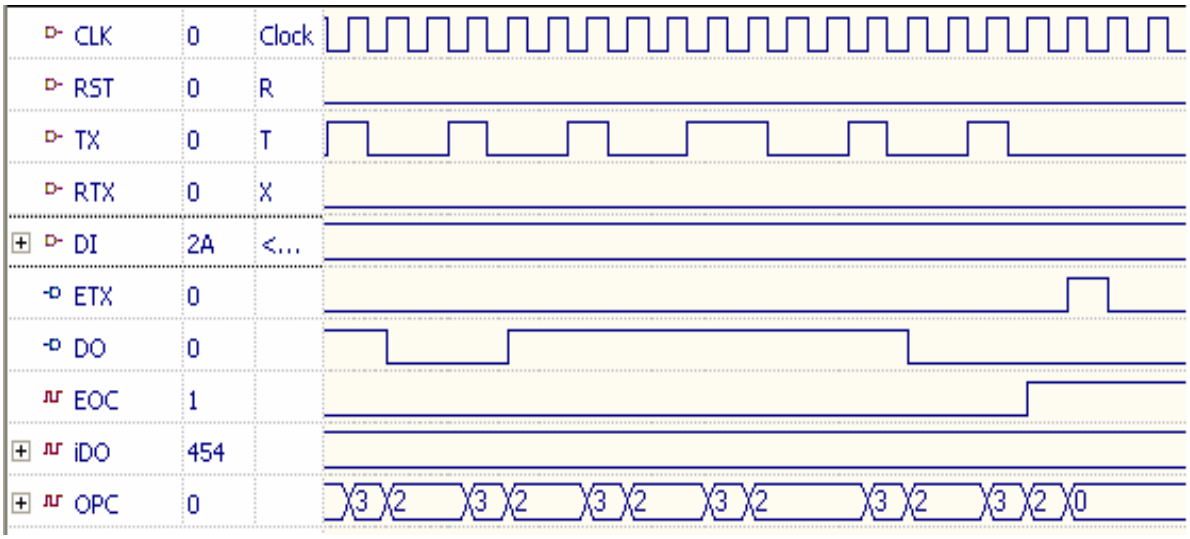

Figura 4.6 Fin de la Transmisión.

4.4 Recepción

Este módulo es el encargado de recibir la información enviada por la computadora, al momento de que la PC solicita al teclado que va ha enviar datos, se genera la señal RX, habilitando este proceso.

Se reciben los datos a través *LDA* y se almacenan en el vector *iD* por medio de un registro de corrimiento serie-paralelo que se habilita con la señal OP. En el mismo instante de recibir la información se va determinando el bit de paridad, para determinar el correcto envío. Este proceso finaliza al activarse la señal de *ERX*. Al principio el vector *CMD* contiene el código de *AA* que sirve para la inicialización del teclado. En la Figura 4.7 se observa algunas de estas señales.

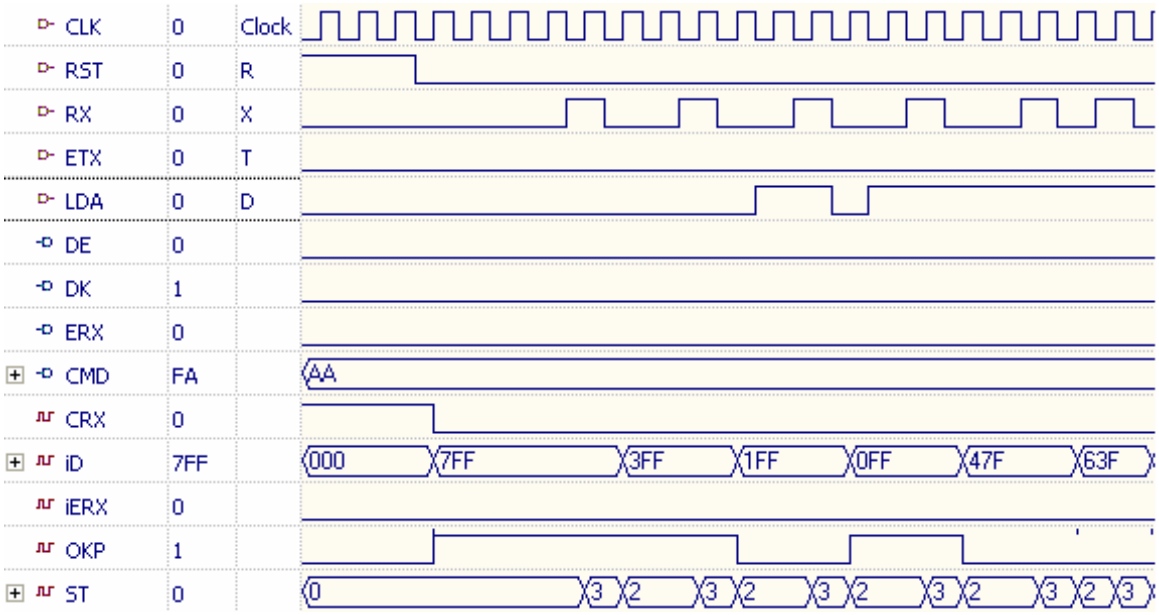

Figura 4.7 Secuencia para la recepción de información

4.5 Máquina de estados principal

Este módulo es el cerebro de todas las estructuras diseñadas, se encarga de proporcionar a cada bloque las señales necesarias, en la Figura 4.8 se muestran las banderas que deben de activarse cuando se inicia el proceso de transmisión. Al momento de energizar el dispositivo (teclado), la primera función es el proceso de transmisión, antes de iniciar con esta tarea monitorea las líneas *LDA y LCK,* cuando las ambas líneas estén en alta impedancia empieza con el envío de información. Al finalizar con esto de nuevo monitorea las líneas para recibir la respuesta al comando enviado.

En la Figura 4.9 se ilustra el proceso de recepción y las señales que intervienen para que el módulo de recepción realice su trabajo. En este proceso se requiere que la línea de reloj baje por lo menos 60 µs, después se libere y baje la de datos, seguido de ello el teclado debe proporcionar la señal de reloj.

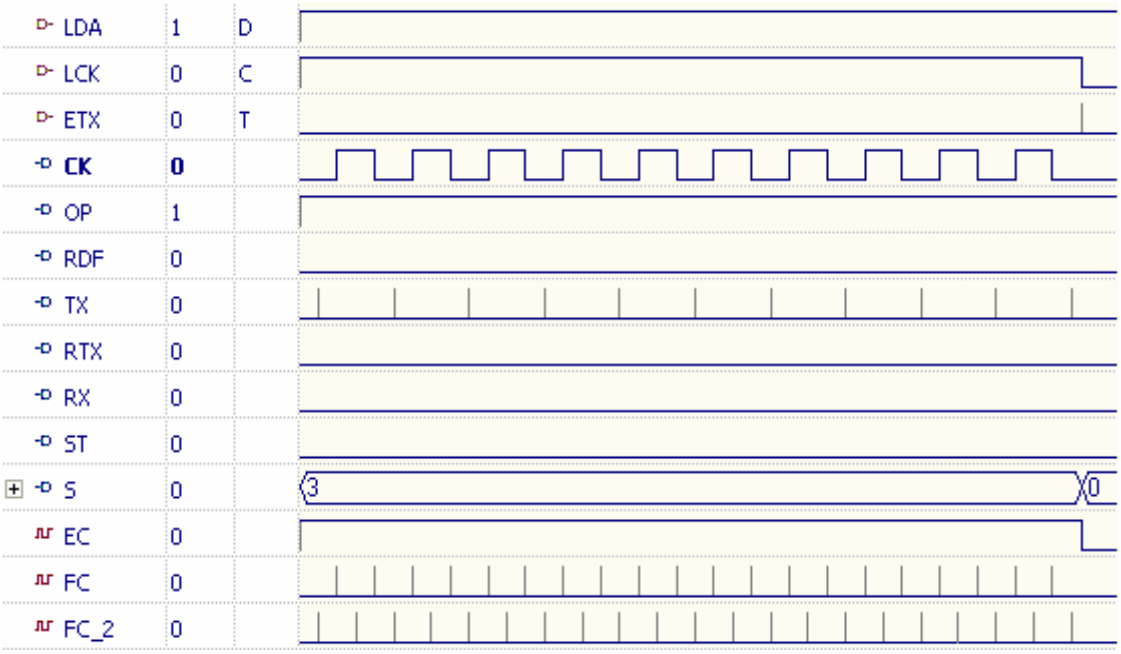

Figura 4.8 Señales generadas para la transmisión

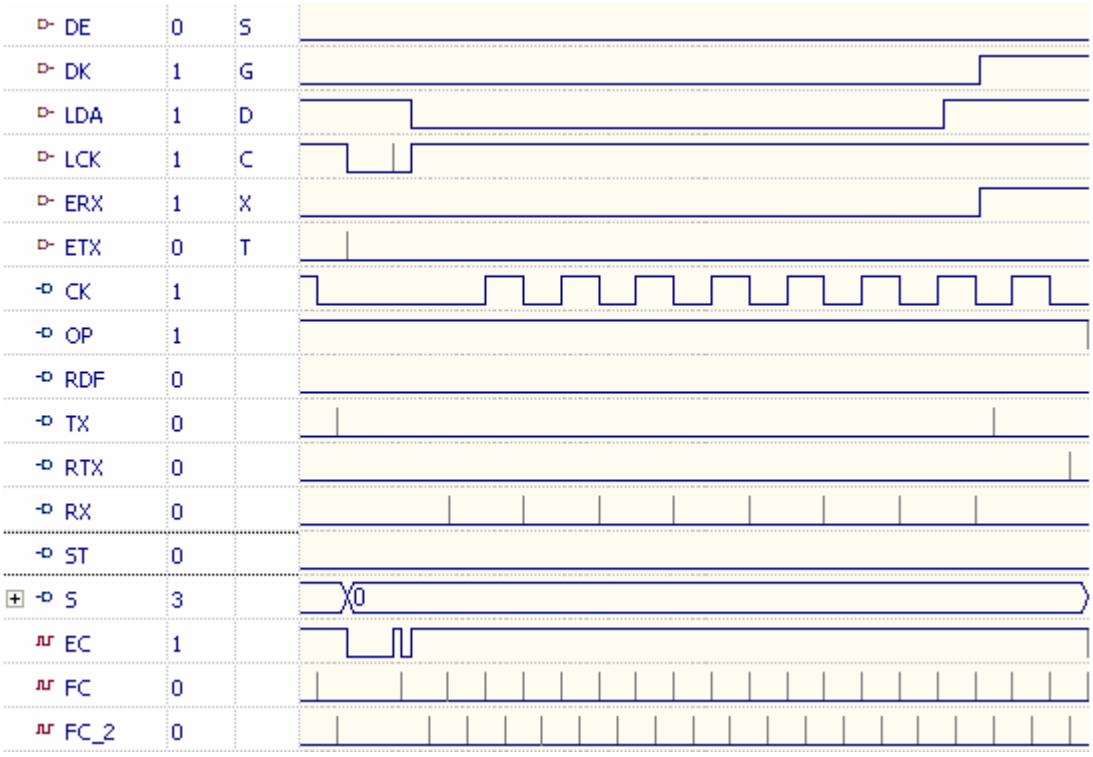

Figura 4.9 Señales producidas para la recepción

4.6 Registro

OP	Dato
00	"0000000"
01	Toma el valor de DTC
10	Toma el valor de DPC
11	No cambia

Tabla 4.1 Valores posibles del vector Dato.

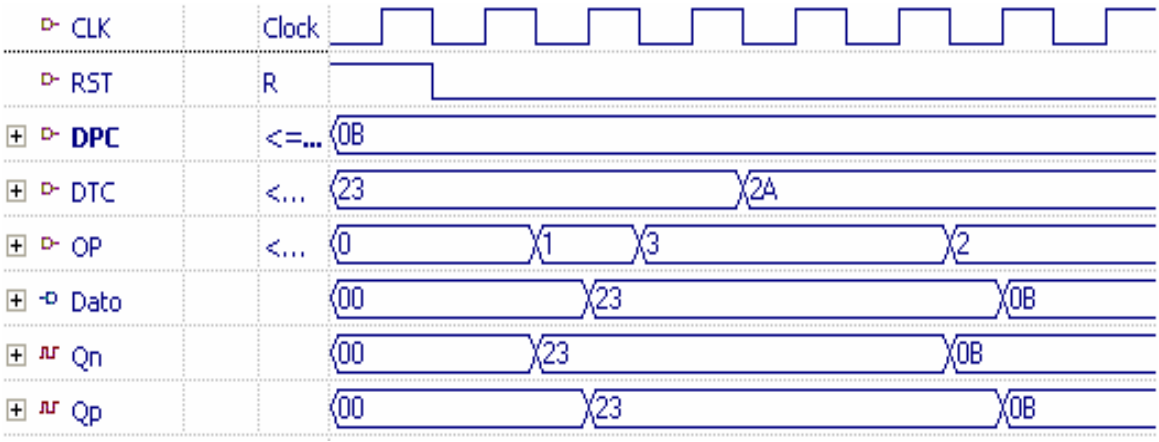

Figura 4.10 Señales del Registro.

En la Tabla 4.1 se observa los valores que puede tomar el vector *Dato*, dependiendo de las combinaciones de *OP*, mientras que en la Figura 4.10 se muestra de igual forma el procedimiento para esta asignación. El valor de *OP* es proporcionado por el módulo de control.

4.7 Protocolo AT

Este bloque es la unión de todas las estructuras anteriores y tiene la función de realizar el protocolo de comunicación entre el teclado y la PC y viceversa. El primer proceso que desempeña es el de transmisión que se muestra en la Figura 4.11, se toma el comando *AA* proporcionado por el módulo de recepción, se almacena en el acumulador y se

pasa a la transmisión, cuando ya se tiene el dato listo el bloque de control debe de generar la señal de reloj, y *TX*. Al finalizar el envío monitorea *LDA y LCK* para recibir la respuesta al código enviado.

De igual forma para la recepción se monitorean *LDA, EF, DE, DK; LDA* es la línea con la se comunica la PC y el teclado, en tanto las otras son banderas que indican que hay información para procesar. En la Figura 4.12 se observa este proceso; al momento de activarse una tecla y se guarda la información correspondiente de la misma, cambia de estado *EF,* entonces Control produce el habilitador de lectura de la *FIFO,* la información pasa por los diferentes módulos hasta llegar el de transmisión.

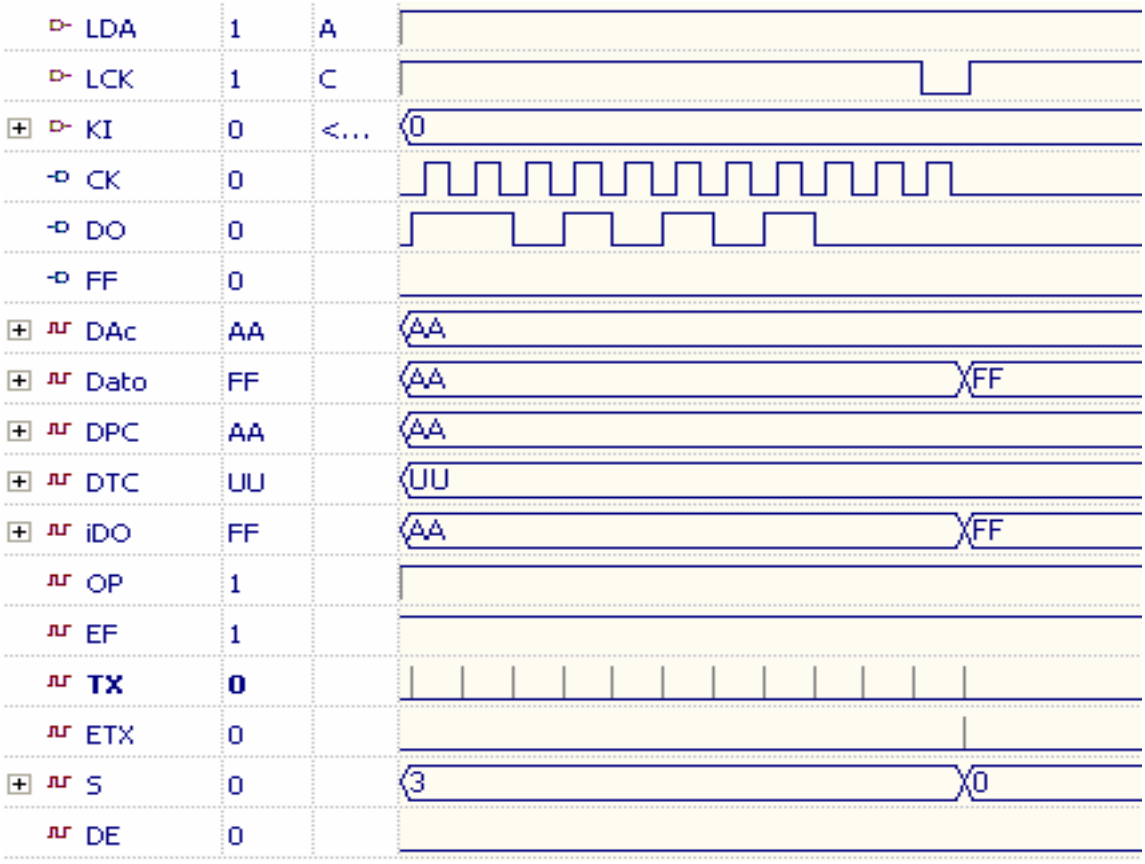

Figura 4.11 Protocolo de transmisión.

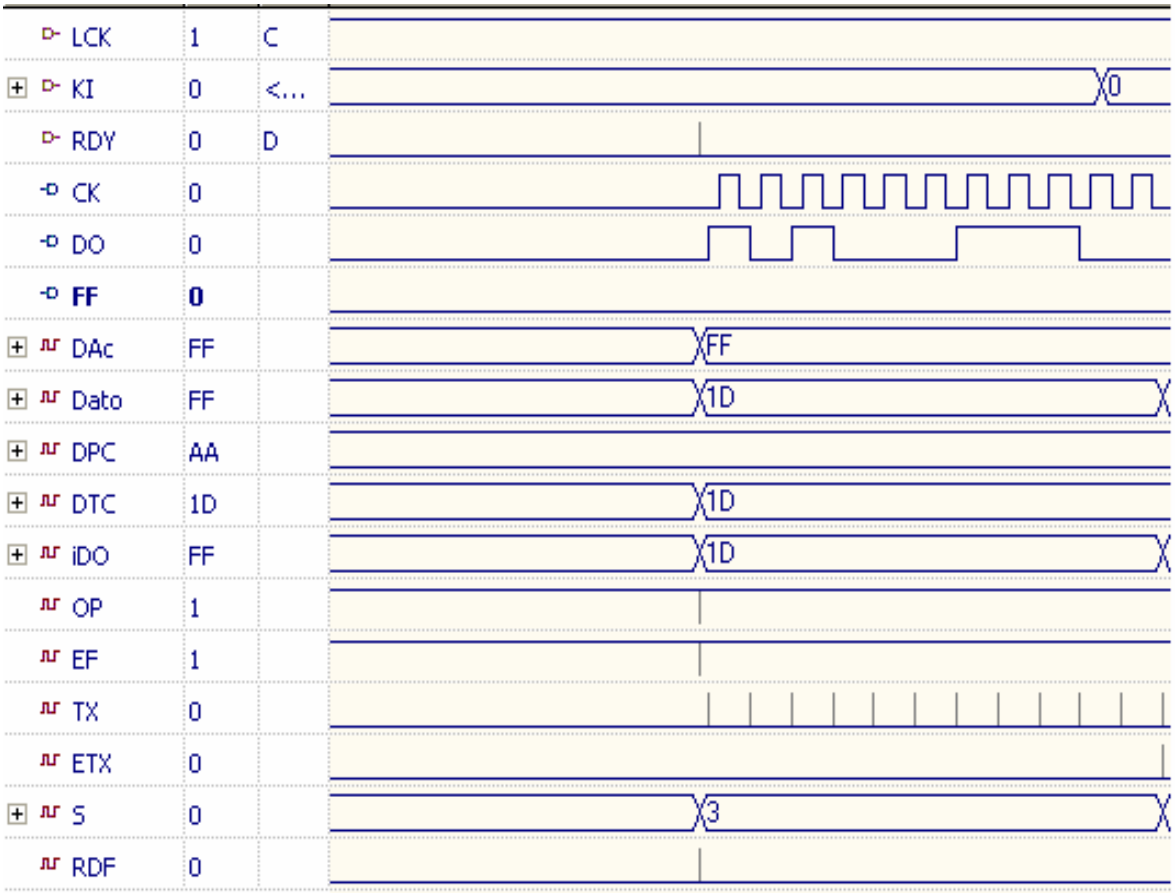

Figura 4.12 Protocolo de recepción.

4.8 PCB del teclado

La Figura 4.13 muestra el circuito impreso del prototipo, se emplearon switchs push-button normalmente abiertos para representar las teclas, consta de 32 switchs conectados a un común que es VCC. Al momento de cerrar el interruptor genera el 1 lógico, para producir el cero lógico se conectan a tierra a través de una resistencia.

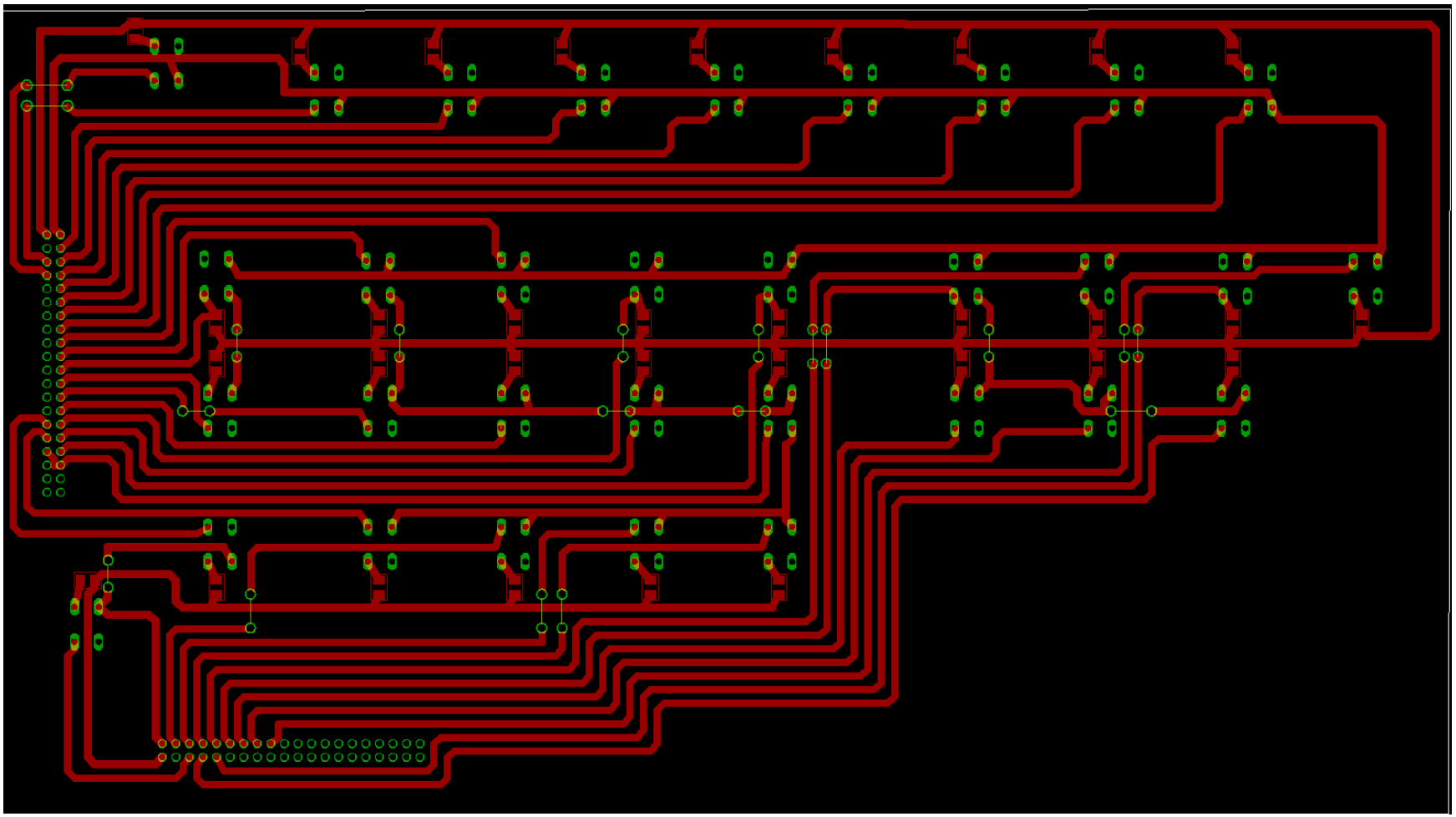

Figura 4.13 PCB del teclado.

V. CONCLUSIONES

La tecnología FPGA permite desarrollar diseños a la medida de las necesidades, bajo costo e incluso para la producción de pocas unidades. Con la ventaja que se puede reutilizar o modificar los diseño sin afectar el diagrama eléctrico.

Debido a las características que ofrecen los FPGA permite que el trabajo realizado tenga mayor flexibilidad, es decir; es posible implementarlo a diferentes procesos de automatización o maquinaria que requiera de algún teclado especial, logrando con ello se una opción más para las MPYMEs.

Con los módulos digitales que se implementaron permiten guardar la información de la tecla activada aún cuando se hayan presionado más de una. El tipo de diseño implementado permite que en un futuro se adicionen otros bloques sin afectar los anteriores.

BIBLIOGRAFÍA

Ayala Guzmán Oscar Mauricio, Gonzáles Hernández Javier. 2006. Desarrollo de un Robot Móvil con Control Inteligente usando Algoritmos Evolutivos y un FPGA.

Boylestad Robert L., Nashelsky Louis. 1997. Pearson Education. Sexta Edición.

Bravo Curiel José Agustín. 2004. Desarrollo de un Control Modular para Maquinaria Aplicado a una Máquina de Inyección de Plástico.

Bravo Muñoz Ignacio, Boquete Vázquez Luciano, Marrón Romera Marta, Palazuelos Cagigas Sira E. 2007. Sistema de Control de una Freidora Industrial.

Chapweske Adam. 1992. PS/2 Keyboard Protocol.

CONACYT. 2002. Decreto por el que se Aprueba y se Expide el Programa Especial de Ciencia y Tecnología 2001-2006.

Consejo Nacional de Ciencia y Tecnología (CONACYT). 2006. Informe General del Estado de la Ciencia y la Tecnología en México.

De Priego Arturo Miguel. 2007. Una Metodología Integral de Diseño Digital con CPLD's, VHDL y C.

Deschemps Jean Pierre. 2002. Síntesis de Circuitos Digitales; un Enfoque Algorítmico. Thomson.

Dussán Álvarez María Antonieta, Jiménez Hernández Luis Alexander, Hernández Suárez Cesar Augusto, Giraldo Peñaranda Leonel, Acosta Villamizar Felipe. 2004. Sistema Electromecánico para el Aprendizaje de la Lectoescritura del Braille.

Enríquez González Edgar. 2006. Sistema de Control de Temperatura Aplicado a Máquina de Inyección de Plástico.

Estévez Mesa Manuel. 1998. Teclado de Conceptos y Aplicaciones Informáticas como Ayudas para la Comunicación no Vocal.

González J., González I., y Boemo E, 2003, Alternativas Hardware para la Locomoción de un Robot Apodo.

Haro Orozco Luis Eduardo, Hernández Muñoz Luis Gerardo, Trujillo Hernández Juan Paulo, Varela Pérez Alonso. 2003. Diseño de PCB's de Alto Nivel utilizando Interfaz Gráfica.

Hoffman Pablo, Szmulewicz Martín, 2006. Osciloscopio USB

Iborra Andrés, Álvarez Bárbara, Jiménez Concha, Navarro Pedro J., Fernández Carlos. 2007. Sistema de Inspección de Visión Automatizado (SIVA) de Cigüeñales, sobre la Base de una Arquitectura Hardware Reconfigurable.

Mandado Enrique, Luis Jacobo Álvarez, Valdés Ma. Dolores. 2002. Dispositivos Lógicos Programable. Thomson.

Martín José María, Palazuelo Martin. 2005. Harware Microinformático. Alfaomega Grupo Editorial. Cuarta Edición.

Mateos Suárez Juan Gilberto, Mateos Ortega Héctor. 2007. Instrumentación Virtual en la Medición de Nivel de Líquidos.

Maxinez David G., Alcalá Jessica. 2002. El Arte para Programar Sistemas Digitales. CECSA. Primera Edición. México.

Mclaughlin Dave. 1995. Pinout for PS/2 Keyboard.

Oliver Juan P., Pérez Acle Julio. 1999. Utilización de FPGAs como Aceleradores de Cálculo.

Palacios Escamilla Victor Alberto. 2004. Arquitectura Basada en FPGA para la Gestión de Movimiento de dos Robots con 3 Grados de Libertad.

Pérez Madaín, Cabestaing Francois, Postaire Jack Gérard. 2007. STREAM, un Procesador para Tratamiento de Secuencia de Imágenes: Aplicación a la Estéreo Visión Densa.

Pérez Serafín Alfonso, Soto Enrique, Fernández Santiago. 2002. Diseño de Sistemas Digitales con VHDL. Thomson.

Piedrahita M. R.. 2001. Ingeniería de la Automatización Industrial. Alfaomega Grupo Editor. México D. F.

Rodríguez Zaragoza Adela M. 2006. Desarrollo de una Estación Meteorológica USB.

Rudolph Eric. 1991. IBM Keyboard Interfact Project.

Schultz Mark. 1995. AT Keyboard Interface.

Tecnología del Plástico. 2002. "Asia en América Latina: Crece la Presencia Tecnológica". Información Técnica y de Negocios para la Industria Plástica en América Latina.

Varela Álvaro, Bohórquez Juan Carlos, De Salazar Nahir, Pinilla Mario, García Antonio, Gonzales Mónica, Macías Hugo, Rivera Jaime. 2007. Tecnologías de Apoyo Basadas en la Comunicación Aumentativa y Alternativa.

Vélez Juan Carlos, Echevarría Alex, Gallón Alejandra, 2007, Chip para Adquisición y Edición de Video.

Wakerly John F. 2001. Diseño Digital: Principios y Prácticas. Pearson Prentice Hall. Tercera Edición.

Zavala R., Anzurez J., Lázaro I.I. 2007. Instrumentación Virtual Aplicada a la Comunicación con Instrumentos Vía RS-232.

ANEXOS

Apéndice A

Tarjeta Spartan III

Para realizar las pruebas de los módulos digitales se emplea la tarjeta FPGA Spartan 3 de Xilinx de la Figura A1, que entre sus principales características presenta una capacidad de 200,000 compuertas, 216 Kb RAM (Random Access Memory), doce multiplicadores 18x18 bits, 173 entradas/salidas definidas por el usuario y un oscilador de 50MHz..

a) **b)**

Figura A.Tarjeta FPGA Spartan 3 de Xilinx. a) Vista Superior, b) Vista inferior

Apéndice B

Artículo Publicado

Servoamplificador aplicado a maquinados de alta velocidad.

Servo-amplifier applied to high speed machinery.

Ing. Rafael Santos Cruz[1] y M. en C. Roque Alfredo Osornio Ríos[2]

Estudiante de Posgrado en Ingeniería de la Universidad Autónoma de Querétaro, Campus San Juan del Rio, Qro.[1], Profesor de la Facultad de Ingeniería de la Universidad Autónoma de Querétaro, Campus San Juan del Río, Qro.[2].

rscruz_13@yahoo.com.mx[1], raor@uaq.mx[2].

RESUMEN. Los maquinados de alta velocidad son una tecnología reciente que han ido desplazando a los maquinados convencionales. Consiste en varias pasadas rápidas de poca profundidad de corte, para ello es necesario contar con equipo eficiente que permita controlar los actuadores.

Este trabajo presenta la solución en cuanto al requerimiento de servoamplificadores aplicados en máquinas de alta velocidad, en el que se destaca como principal aportación el uso de herramientas digitales programables y otras, accesibles en costo y funcionalidad; que en combinación con los conocimientos adquiridos en los campos de Ingeniería Electrónica y Control constituyen la base de éste desarrollo. La fundamentación del diseño consiste en el desarrollo de un módulo PWM programable (*Pulse Width Modulation*, Modulación por Ancho de Pulso), el cual se elaboró usando el lenguaje de descripción de hardware VHDL (*Very High Speed Integrated Circuits Hardware Description Language*, Lenguaje de Descripción de Hardware para circuitos de alta velocidad), que permite la descripción, análisis y evaluación del diseño.

La ventaja que ofrece el PWM respecto a los circuitos comerciales es la programabilidad, pues permite variar el ciclo de trabajo y periodo de muestreo, requeridos en los maquinados de alta velocidad para adecuarse al control sin necesidad de cambiar los dispositivos que interactúan con él. La implementación de este módulo se realizó en una tarjeta FPGA (*Field Programmable Gate Array*, Arreglo de Compuertas Programables en Campo) Spartan III de la marca Xilinx. Además del modulo PWM se desarrolló la lógica necesaria para controlar la etapa de potencia. El circuito utilizado fué el módulo IGBT CPV364M4F (*Insulated Gate Bipolar Transistor*, transistor bipolar de puerta aislada) de la compañía International Rectifier, también se usó el circuito controlador IR2130 para el controlar el disparo de los IGBT, de la misma marca. Algunas de las aplicaciones en las cuáles puede emplearse éste diseño son: centros de maquinados, robots, para la automatización de máquinas herramienta (tornos, fresadoras), entre otras. Los resultados que arrojaron las pruebas fueron satisfactorios, se controló el giro de un servomotor (actuador) en vació para ambos sentidos.

Palabras Clave: Servoamplificador, FPGA, PWM, Maquinados de Alta Velocidad.

1. INTRODUCCIÓN.

En la actualidad las investigaciones enfocadas a la automatización de procesos han ido tomando cada vez más importancia, pues se necesita tener un control más efectivo sobre las máquinas herramienta, para lograr la reducción de los tiempos de producción sin que se pierda la calidad en las piezas. En los maquinados de alta velocidad se tiene la capacidad de aumentar la velocidad de arranque de viruta, teniendo la reducción de tiempo de maquinado y el aumento de vida de la herramienta de corte (Sivak et al, 2001). Por otro lado, el empleo de mayores velocidades conlleva a que el control para una máquina CNC de alta velocidad exige un mejor diseño en comparación con un CNC convencional, como son los periodos de muestreo y el tiempo accionamiento del control y de actuadores.

A medida que los sistemas de desarrollo son elaborados por compañías específicas o sean innovaciones, el costo es considerable, ocasionando que algunos sectores no puedan tener acceso, como es el caso de las PyMES (Pequeñas y Medianas Empresas). Algunas compañías que fabrican servoamplificadores son Copley Control, Delta Tau, Advanced Motion Control, Bayside Motion Group, Galil, Cinfranor Group,

PML, Numatics. La Facultad de Ingeniería de la Universidad Autónoma de Querétaro, ha empleado en diversos proyectos servoamplificadores de la compañía Copley Control, la cual fabrica dos tipos de servoamplificadores: Analógicos y Digitales, cuyos costos oscilan entre $ 6000 y $8000 (2006). Por ello que es necesario implementar nuevas alternativas para dar solución a los problemas de la industria, sin necesidad de depender de tecnologías específicas.

2. REVISIÓN DE LITERATURA.

2.1 Estado del arte.

La lógica programable como parte del desarrollo y evolución de la electrónica digital ha facilitado la implementación de sistemas complejos, reduciendo considerablemente el espacio y costo en las aplicaciones. Las aplicaciones de la lógica programable iniciaron con los PLD's (*Programmable Logic Device*, Dispositivo Lógico Programable), en la actualidad los FPGA's debido a sus características eléctricas que ofrecen han ido reemplazando componentes como microcontroladores y microprocesadores. Se han desarrollado aplicaciones con FPGA en diversos campos de la automatización. González et al. (2003), desarrollaron un sistema de locomoción aplicado a un robot apodo, donde emplean un FPGA, para diseño de un contador, una memoria ROM (*Read Only Memory*, memoria de solo lectura) y las unidades PWM necesarias para mover al robot. Por otro lado Sánchez et al. (2003) utilizaron un FPGA XC4000XL de Xilinx en el desarrollo de una placa prototipo para test automatizado de circuitos, que consiste en la verificación funcional y medición de algunos parámetros importantes para el diseño. Ramos et al. (2001), usando VHDL programaron un FPGA A1010B de Actel para expandir el puerto paralelo de una computadora. La implementación de una red neuronal artificial en un FPGA de la familia XC4000 de Xilinx, fue realizada por Tosini et al. (2007), mediante la cual pretende efectuar predicciones climáticas en un medioambiente acotado. Vélez et al. (2007), realizaron un sistema de captura y procesamiento de imágenes a alta velocidad, mediante un control para la adquisición y edición de video, empleando un FPGA de Xilinx XC4010XL.

La Modulación de Ancho de Pulso es un método empleado para limitar la corriente que circula por las bobinas de un motor a pasos utilizado en sistemas, donde se requiera grandes precisiones en el movimiento (Berti et al, 2007). Espinosa et al. (2005), utilizaron una señal PWM bipolar con ciclo de trabajo variable, la emplearon para conmutar dispositivos de potencia, implementados en la regulación de voltajes en inversores UPS (*Uninterruptible Power Supply*, fuente de poder ininterrumpida), mediante un controlador basado en redes neuronales.

En los inicios del diseño de servoamplificadores, Maezawa (1975), realizó un diseño empleando circuitos analógicos (amplificadores operacionales principalmente), para controlar un servomotor de 2 fases de corriente alterna. Hoy día con la evolución de la tecnología, es posible utilizar componentes que nos permitan tener un mejor diseño, reducción de espacio y un control eficaz sobre el actuador. Bossa et al. (2006), utilizaron el integrado IR2130 para generar las señales de excitación de un modulo IGBT, por medio de la tecnología Bootstrap, y proporciona los tiempos de retardo, en el desarrollo de banco didáctico enfocado a la electrónica de potencia (Bossa et al, 2006). Lamitch et al. (2004), utilizaron pulsos TTL de un DSP (*Digital Signal Processor*, procesador digital de señales) para controlar al drive IR2132, empleado en la etapa de potencia de un inversor de tensión para el control de motores.

2.2 Fundamentación teórica.

2.2.1 Dispositivos Lógicos Programables.

El desarrollo de la microelectrónica ha hecho posible desarrollar sistemas completos dentro de un solo circuito integrado. La combinación de sistemas ha ido evolucionando con el surgimiento de nuevas tecnologías de fabricación, permitiendo la obtención de componentes estándares con mayor complejidad y beneficios como los dispositivos lógicos programables. Se trata de circuitos fabricados que se pueden personalizar desde el exterior mediante técnicas de programación. Su diseño se basa en bibliotecas y

mecanismos específicos de mapeo de funciones. Los PLD's han ido reemplazando circuitos de pequeña, mediana e incluso alta escala de integración (SSI, MSI, VSI respectivamente). La estructura básica de un PLD esta formada por un arreglo de compuertas AND y OR conectadas a las entradas y salidas del dispositivo. Algunos PLD's y su descripción se muestran en la Tabla 1.

2.2.2 Lenguaje de Descripción de Hardware.

Con la creciente necesidad de integrar un mayor número de componentes en un solo circuito integrado, se fueron desarrollando nuevas herramientas de diseño que permitieron integrar sistemas de mayor complejidad. Ésto generó que en la década de los cincuentas surgieran los lenguajes de descripción de hardware (HDL); algunos de ellos fueron IDL, TI-HDL, ZEUS, entre otros; los cuales estaban destinados al uso industrial. Mas tarde en la década de los ochentas surgen lenguajes como VHDL, Verilog, ABEL 5.0, AHDL, entre otros, que permitieron tomar un problema lógico a nivel funcional, facilitando la evaluación de soluciones antes de iniciar un diseño detallado (Maxinez et al, 2002). VHDL es un lenguaje no necesariamente conectado con síntesis de hardware, pero viene asociado a la implementación de sistemas digitales (Rosado et al, 2007).

TABLA 1. Dispositivos lógicos programables.

Dispositivo	Descripción
PROM (*Programmable Read-Only Memory*, memoria programable de sólo lectura)	Tiene un arreglo de compuertas AND fijo, OR programable, se utiliza como memoria bidireccionable.
PLA (*Programmable Logic Array*, arreglo lógico programable).	Arreglo AND y OR programables, el usuario lo programa, poseen retroalimentación de la matriz OR a la AND.
PAL (*Programmable Array Logic*, lógica de arreglos programables).	Arreglo AND programable y OR fijo, dispositivo programable por el usuario.
GAL (*Generic Array Logic*, arreglo lógico genérico).	AND programable y OR fijo, es reprogramable y contiene configuraciones de salida programables.
CPLD (*Complex Programmable Logic Device*, dispositivo lógico programable completo).	Son de alto nivel de integración, mayor velocidad y frecuencia de operación, tienen una gran capacidad equivalente a unos 50 PLD sencillos.
FPGA (*Field Programmable Gate Array*, arreglos de compuertas programables en campo).	Se basan en arreglos de compuertas, los cuales contienen elementos configurables: bloques lógicos configurables, bloques de entra y salida y canales de comunicación.

2.2.3 Transistor Bipolar de Puerta Aislada.

Los transistores de potencia tienen características controladas de corte y conducción; además se utilizan como elementos conmutadores y trabajan en la región de saturación, dando como resultado una caída de voltaje baja en estado de conducción. La velocidad de conmutación de los transistores modernos es mucho mayor que la de los tiristores, por lo que se utilizan en forma amplia en convertidores de corriente alterna a corriente directa y viceversa. Sin embargo, las especificaciones de voltaje y de corriente son menores en los tiristores, es por ello que los transistores se utilizan en aplicaciones de baja y media potencia. Se clasifican de manera general en cuatro categorías: BJT (*Bipolar Juntion Transistor*, transistor bipolar de juntura), MOSFET (*Metal Oxide Semiconductor Field Effect Transistor*, transistor semiconductor de metal de óxido de efecto de campo), IGBT y SIT (*Static Induction Transistor*, transistor de inducción estática).

Un IGBT combina las ventajas de los BJT y MOSFET. La impedancia de entrada en la terminal de control es muy alta, de modo que la cantidad de corriente que fluye en ella es pequeña. La especificación de corriente para un solo IGBT puede llegar hasta 400 amperes, 1200 volts, y una frecuencia de conmutación

hasta 20 KHz (Kilo Hertz) (Muhammad, 1995). Existen varios dispositivos que pueden emplearse como interruptores de potencia: SCR (*Silicon Controlled Rectifier*, rectificador controlado de silicio), SBS (*Switch Bilateral Silicon*, interruptor bilateral de silicio), GTO (*Gate Turn-Off Switch,* interruptor abierto por puerta), pero los IGBT's tienen mayor auge en aplicaciones de media y alta potencia; en configuración tipo puente el IGBT MGW20N60D que al ser activados adecuadamente permiten invertir el flujo de corriente a través de la carga (Sandoval et al, 2007).

3. METODOLOGÍA.

La Figura 1 muestra el sistema de control de una máquina herramienta de alta velocidad; la cual consta de dos tipos de actuadores: motor de corriente alterna en el husillo y servomotor en las bancadas o ejes, éste trabajo se aplica en los ejes. En la computadora se programa la rutina que la máquina debe desempeñar, manda la posición por medio de una interfaz a la tarjeta controladora, ésta a su vez, envía la señal de control al servoamplificador, para proporcionar la corriente necesaria a los servomotores, generando así el movimiento de los ejes. También la tarjeta controladora manda una señal al variador para aumentar o disminuir la velocidad del husillo.

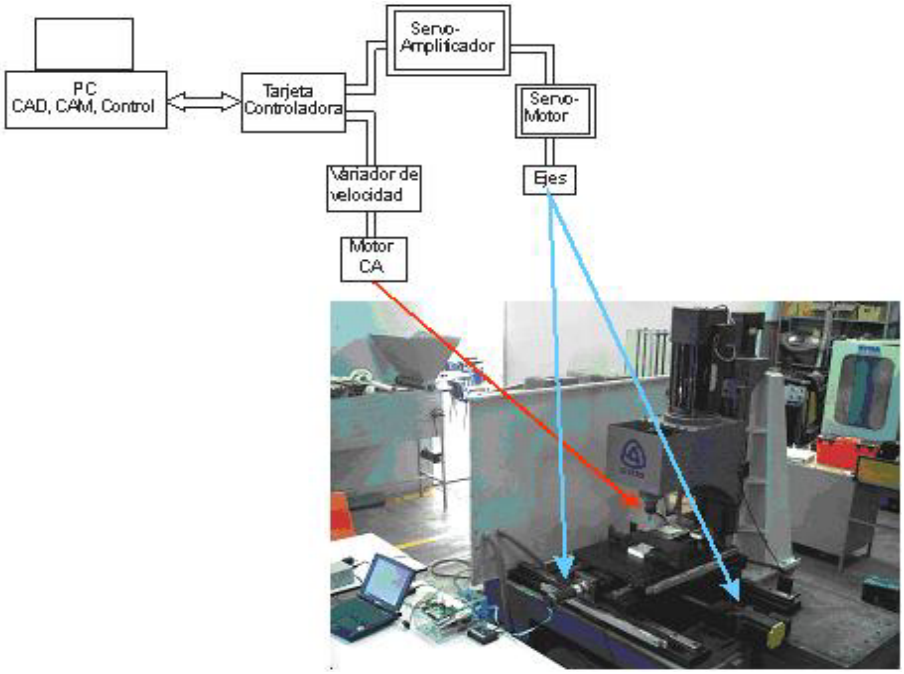

FIGURA 1. Diagrama a bloques del sistema de control para una máquina de alta velocidad.

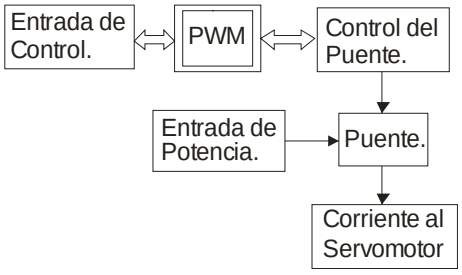

FIGURA 2. Diagrama a bloques del servoamplificador

En la figura 2 se muestra la estructura de diseño del servoamplificador; la señal de entrada de control proviene de un controlador digital, el PWM toma la señal de control para generar las señales para activar el

circuito controlador IR2130, el cual genera las señales de disparo para el puente, que es un módulo de IGBT's CPV364M4F. Al activarse los IGBT's proporcionan la corriente necesaria al servomotor. La Figura 3 muestra el diagrama a bloques de los módulos digitales que conforman al PWM programable, los cuales fueron programados y se simularon en VHDL. El módulo de selección está constituido por una estructura multiplexora con entrada de 3 bits y una salida de 32 bits. Se requiere un contador digital de 32 bits, tipo ascendente/descendente; un reloj auxiliar genera el tiempo de la duración del ciclo de trabajo. El divisor, cuya función es la de convertir la frecuencia de trabajo 50 MHz (FPGA) a una frecuencia de muestreo acorde con los maquinados de alta velocidad; se emplea un módulo de muestreo para generar las señales de sentido de giro; un sincronizador empleando un flip-flop tipo D. Todo el modulo PWM programable se implementó en una tarjeta FPGA de la marca Xilinx, que presenta las siguientes características: tiene un reloj de 50 MHz, un socket para un oscilador auxiliar, 3 puertos de expansión con 40 pines de entrada/salida cada uno y 200000 compuertas.

En la etapa de potencia se utilizó el circuito controlador IR2130, que es un dispositivo de alto voltaje, de hasta 600 volts, alta velocidad, tiene 3 salidas de canales altos y bajos, sus entradas son compatibles con tecnología TTL (*Transistor Transistor Logic,* lógica de transistor transistor) y CMOS (*Complementary Metal Oxide Semiconductor,* semiconductor de metal de oxido complementario), cuenta con un circuito de protección por sobrecorriente y bajo voltaje, tolerancia para voltaje transitorio negativo. El IR2130 toma los pulsos proporcionados por la tarjeta FPGA para generar las señales requeridas en la activación de las puertas de los transistores IGBT. Éste circuito integrado, contiene 6 IGBT's conectados internamente en tipo puente, cada IGBT tiene conectado un diodo para la protección de marcha inversa, el rango de frecuencia para la conmutación es de 1 a 10 KHz y un voltaje de suministro de hasta 360 Vdc.

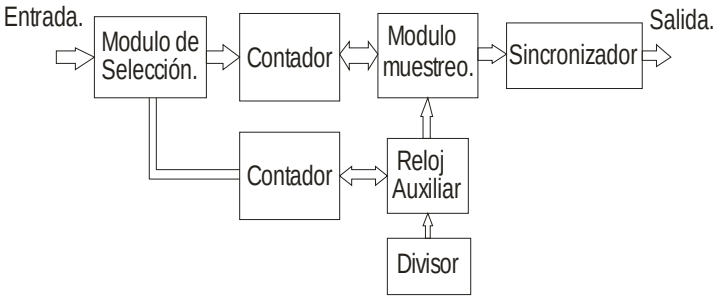

FIGURA 3. Estructura interna del PWM programable.

4. RESULTADOS.

Para comprobar la funcionalidad de los módulos diseñados, se realizaron simulaciones en VHDL de los módulos individuales y del PWM completo. Posteriormente se sintetizaron e implementaron en el FPGA. La Figura 4 muestra las señales adquiridas del FPGA, donde presenta dos señales de salida complementarias, mediante las cuales es posible tener el giro en ambos sentidos del servomotor.

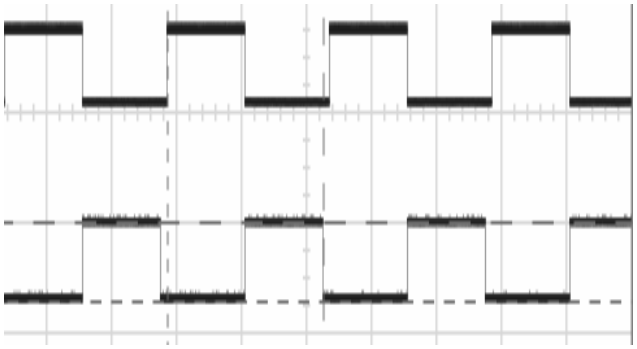

FIGURA 4. Señales de salida del PWM.

Al alimentar la etapa de potencia y conectar las señales generadas en el FPGA al circuito controlador, los transistores conmutan obteniendo a la salida del puente una señal cuadrada como se muestra en la Figura 5, cuya amplitud pico es 2 veces la amplitud de la señal de entrada menos la caída de tensión en los dos transistores que conmutan en cada semiciclo.

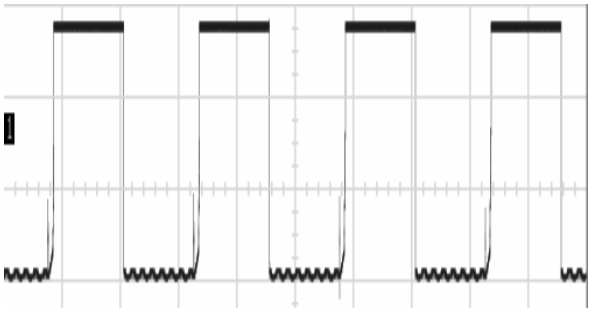

FIGURA 5. Señales de salida del puente.

La Figura 6 muestra el prototipo que se realizó para el servoamplificador.

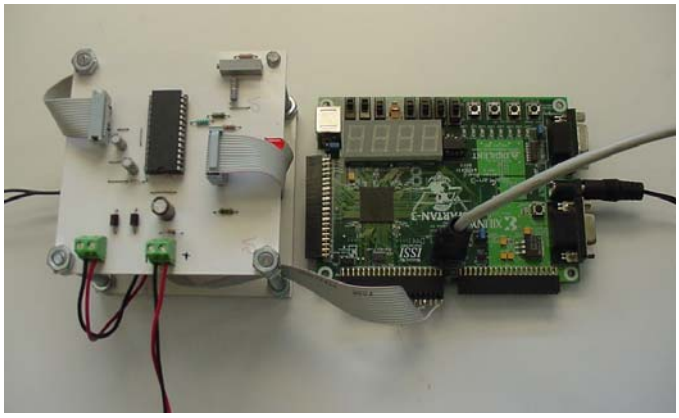

FIGURA 6. Prototipo del servoamplificador.

5. CONCLUSIONES

La creación de un PWM programable mediante diseño digital, empleando FPGA es posible gracias a la capacidad de reconfigurabilidad, diseño compacto y bajo costo. Se ha desarrollado un servoamplificador para maquinados de alta velocidad, en máquinas CNN y que además se puede aplicar en robots. Es posible generar PWM a bajo costo y con las características adecuadas, en cuanto al ciclo de trabajo y frecuencia, mediante tecnología propia. Los resultados obtenidos fueron validados con simulaciones y haciendo pruebas de funcionabilidad sobre un servomotor. Frente a otras alternativas de aplicación que utilizan servoamplificadores comerciales, éste diseño ofrece la misma capacidad de efectividad como lo demostraron los resultados. Como propuesta de trabajo futuro se pretende probar el servoamplificador diseñado en un sistema de control de una máquina herramienta, para validación del mismo.

6. REFERENCIAS

Berti Sergio, Roitman Javier, Verrastro Claudio, 2007, Controlador de Motores Paso a Paso Mediante Técnicas de Micropasos por Modulación de Ancho de Pulso.

Bossa José Luis, Serra Federico Martín, 2006, Banco Didáctico Dedicado a la Electrónica de Potencia.

Copley Controls Corp., 2006, Advanced Motion Control & Power Amplifiers.

Espinosa Martínez Israel, 2005, Regulación de Voltaje en Inversores UPS Mediante un Controlador Basado en Redes Neuronales.

González J., González I., y Boemo E, 2003, Alternativas Hardware para la Locomoción de un Robot Apodo.

Lamitch M., Pérez D., Arias A., Sala V., Jean C., Aldabas E., 2004, Inversor de Tensión Controlado Mediante DSP de Aplicación Docente en Electrónica de Potencia y Control.

Maezawa, Susumu Olita, 1975, Servoamplifier Device.

Maloney Timothy J., 1983, Electrónica Industrial. Dispositivos y Sistemas, Ed. Prentice-Hall Hispanoamericana, S.A.

Maxinez David G. y Alcalá Jara Jessica, 2002, VHDL. El Arte de Programar Sistemas Digitales, Editorial Continental, Primera edición.

Muhammad H. Rashid, 1995, Electrónica de Potencia. Circuito, Dispositivos y Aplicaciones, Ed. Person Educación, Segunda edición.

Ramos Arreguin Juan Manuel, 2001, Descripción VHDL de Interfaz para el Puerto Paralelo de una PC.

Rosado A., Bataller M., Guerrero J., Muñoz J., Vila J., 2007, Un Laboratorio de Diseño Digital en VHDL: Aprendizaje por Proyectos.

Sánchez M, Jiménez R., 2007, Placa Prototipo para el Desarrollo de Test Automatizados de Circuitos en FPGAs.

Sandoval J., Salamanca W., Cardozo V., Duarte J., Fernández F., 2007, Desarrollo de Inversor Monofásico Didáctico.

Serra Moisés, Xavier Rafael, Ordeix Juli, Marti Pere y Carabina Jordi, 2007, Prototipo Demostrador de OFDM: Transmisor.

Sivak M., Martínez Krahmer D., Maceira G., 2007, El Mecanizado de Alta Velocidad (MAV).

Tosini Marcelo Alejandro, Acosta Gerardo, Boemo Eduardo Ivan, 2007, Ungen: Un Sistema de Generación Automática de Redes Neuronales Digitales para FPGA.

Vélez Juan Carlos, Echevarría Alex, Gallón Alejandra, 2007, Chip para Adquisición y Edición de Video.

Apéndice C

Código en VHDL del módulo general para el protocolo AT

```
Library IEEE;
Use IEEE.std_logic_1164.all;

entity ProtoAT is
        port (
             RST : in std_logic;
             CLK : in std_logic;
             En : in std_logic;
             PCK : in std_logic;
             PDA : in std_logic;
             RDY : in std_logic;
             KI : in std_logic_vector(3 downto 0);
            TLC : inout std_logic;
            TLD : inout std_logic;
            FF : out std_logic;
            CK : out std_logic;
            DO : out std_logic
            );
end ProtoAT;

architecture ProtoAT of ProtoAT is
component SCAN_AT
        port (
             RST : in std_logic;
             CLK : in std_logic;
             En : in std_logic;
             RDF : in std_logic;
             RDYTX : in std_logic;
             KI : in std_logic_vector(3 downto 0);
             E : out std_logic;
             F : out std_logic;
             Data : out std_logic_vector(7 downto 0)
             );
end component;

component TX_AT
        port (
             RST : in std_logic;
             CLK : in std_logic;
             TX : in std_logic;
             RTX : in std_logic;
             DI : in std_logic_vector(7 downto 0);
             ETX : out std_logic;
```

```
            DO : out std_logic
            );
end component;

component RXAT
        port (
            RST : in std_logic;
            CLK : in std_logic;
            RX : in std_logic;
            LDA : in std_logic;
            ETX : in std_logic;
            ERX : out std_logic;
            DK : out std_logic;
            DE : out std_logic;
            CMD : out std_logic_vector(7 downto 0)
            );
end component;

component RegPP
        port (
            RST : in std_logic;
            CLK : in std_logic;
            OP : in std_logic_vector(1 downto 0);
            DTC_1 : in std_logic_vector(7 downto 0);
            DTC_2 : in std_logic_vector(7 downto 0);
            DPC : in std_logic_vector(7 downto 0);
            Dato : out std_logic_vector(7 downto 0)
            );
end component;

component RXTC
        port (
            RST : in std_logic;
            CLK : in std_logic;
            RXT : in std_logic;
            LDi : in std_logic;
            ETC : in std_logic;
            ERC : out std_logic;
            TD : out std_logic;
            TC : out std_logic;
            TDE : out std_logic;
            TDat : out std_logic_vector(7 downto 0)
            );
end component;
```

```
component Acumulador
        port (
            RST : in std_logic;
            CLK : in std_logic;
            OP : in std_logic;
            Di : in std_logic_vector(7 downto 0);
            Do : out std_logic_vector(7 downto 0)
            );
end component;
component TXAT_1 is
        port (
            RST : in std_logic;
            CLK : in std_logic;
            TX : in std_logic;
            RTX : in std_logic;
            DI : in std_logic_vector(7 downto 0);
            ETX : out std_logic;
            DO : out std_logic
            );
end component;


component MUXATP
        port (
            ST : in std_logic;
            Dn : in std_logic_vector(7 downto 0);
            DAc : in std_logic_vector(7 downto 0);
            DO : out std_logic_vector(7 downto 0)
            );
end component;

component TSBuffer
        port (
            OE : in std_logic;
            DI : in std_logic;
            DO : out std_logic;
            IO : inout std_logic
            );
end component;

component ControlAT
        port (
            RST : in std_logic;
            CLK : in std_logic;
            En : in std_logic;
            EF : in std_logic;
            ETX : in std_logic;
```

```
            ERX : in std_logic;
            ETC : in std_logic;
            ERC : in std_logic;
            PCK : in std_logic;
            PDA : in std_logic;
            CKi : in std_logic;
            LDi : in std_logic;
            DK : in std_logic;
            DE : in std_logic;
            TC : in std_logic;
            TD : in std_logic;
            TE : in std_logic;
            TCK : out std_logic;
            CK : out std_logic;
            TXP : out std_logic;
            TXT : out std_logic;
            RXP : out std_logic;
            RXT : out std_logic;
            RTX : out std_logic;
            ST : out std_logic;
            SL : out std_logic;
            LR : out std_logic;
            OP : out std_logic;
            RDF : out std_logic;
            OE : out std_logic;
            OE1 : out std_logic;
            S : out std_logic_vector(1 downto 0)
            );
end component;

signal DT, DK, DE, EF, ETX, RDF, ERX, OP, RXP, TXP, RTX, ST, TCK: std_logic;
signal FS, ERC, ETC, TXT, RXT, TC, TD, TE, LR, SL, CKi, LDi, OE, OE1: std_logic;
signal DTC, DPC, Dato, iDO, DAc, iD, Ac, Data : std_logic_vector(7 downto 0);
signal S : std_logic_vector(1 downto 0);

begin
        Control: ControlAT   port map (RST, CLK, FS, EF, ETX, ERX, ETC, ERC, PCK,
PDA, CKi, LDi, DK, DE, TC, TD, TE, TCK, CK, TXP, TXT, RXP, RXT, RTX, ST, SL,
LR, OP, RDF, OE, OE1, S);
        Escaneo: SCAN_AT port map (RST, CLK, En, RDF, RDY, KI, EF, FF, FS, DTC);
        Registro: RegPP port map (RST, CLK, S, Data, DTC, DPC, Dato);
        Regis: Acumulador port map (RST, CLK, OP, Dato, DAc);
        Multiplexor: MUXATP port map (ST, Dato, DAc, iDO);
        Recepcion_PC: RXAT port map (RST, CLK, RXP, PDA, ETX, ERX, DK, DE,
DPC);
        Transmision_PC: TX_AT port map (RST, CLK, TXP, RTX, iDO, ETX, DO);
```

```
        Recepcion_TC: RXTC port map (RST, CLK, RXT, LDi, ETC, ERC, TD, TC, TE,
Data);
        Regis1: Acumulador port map (RST, CLK, LR, Data, Ac);
        Multiplexor_1: MUXATP port map (SL, Data, Ac, iD);
        Transmisin_TC: TXAT_1 port map (RST, CLK, TXT, RTX, iD, ETC, DT);
        Buf1: TSBuffer port map (OE, DT, LDi, TLD);
        Buf2: TSBuffer port map (OE1, TCK, CKi, TLC);
end ProtoAT;
```