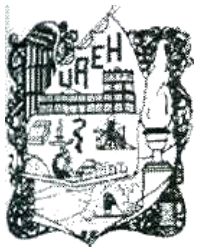


UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO
INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA
INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES
CENTRO DE INVESTIGACIÓN EN TECNOLOGÍAS DE INFORMACIÓN Y SISTEMAS

TESIS

AHIETI 1: MÁQUINA GENERADORA DE CIRCUITOS PCB.

QUE PARA OBTENER EL GRADO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

Presentan:

Christian David Castillo Martínez
Eduardo Sánchez Abad

Asesores
M. en C. Herbert Lara Ordaz
Dr. Ornar Arturo Domínguez Ramírez

Índice general

Índice de figuras

Índice de cuadros

Resumen

El propósito de este mecanismo es el de realizar el descobreado de las pistas y la perforación de los pines que corresponden a los dispositivos que se encuentran en un PCB diseñado previamente mediante software, a n de dejar la placa fenólica lista para soldar los componentes electrónicos. Esta máquina es capaz de trabajar en los tamaños de placas fenólicas comerciales. Para realizar la impresión de un PCB a una placa fenólica con el AHIETI v 2.1.1, es necesario crearlo y exportarlo posteriormente a otro software, éste a su vez envía los datos necesarios al sistema de control del AHIETI v 2.1.1 El descobreado lo efectúa un minidrill con una fresa de carburo de tungsteno. Posteriormente las perfora- ciones se realizan con brocas normales. El prototipo permite ayudar al medio ambiente; ya que en la práctica normal se ocupa un compuesto corrosivo llamado ácido férrico o cloruro férrico, este tiene la propiedad de ser muy agresivo con el cobre y otros metales.

Abstract

The purpose of this mechanism is remove the copper layer of the tracks and the perforation of the pads that correspond to those electronic devices that are in a PCB design, previously made by CAD software, in order to leave the board phenolic ready for the weld of the electronic components. This machine is able to work in the sizes of commercial phenolic boards. The impression of designed circuit on phenolic board is with interpret software that export the file and send the necessary data to the control system device. The device that removes the copper layer is kneed as minidrill and is equipped with a tungsten hss steel instrument. Later the perforations are carried out whit normal high speed steel drills The prototype allows to help to the PCB environment, so the normal practice when is used a compound corrosive call acid ferric or ferric chloride can be changed because this compound is aggressive with the copper and other metals.

Capítulo 1

Introducción.

La tecnología evoluciona rápidamente, inclusive más rápido de lo esperado por los expertos, se ha podido observar sobre todo en la electrónica de consumo, como los semiconductores se hacen más complejos y económicos. Por esta razón los ingenieros deben tratar de seguir este avance, actualizando sus conocimientos, investigando nuevas tecnologías, desarrollando otras técnicas.

El presente trata de emplear una técnica, que si bien no es nueva, si es importarte estudiarla e implementarla. En las aulas se estudian los avances de los componentes y de las tecnologías, tanto de la electrónica, la robótica y las telecomunicaciones, pero se dejan a un lado las técnicas fundamentales para el desarrollo de las mismas. Por ello se abarca el desarrollo de una técnica para elaborar los circuitos impresos de prototipos. Los métodos usados actualmente para el diseño y la elaboración de los PCB por los estudiantes de electrónica no han cambiado desde hace mucho tiempo, se sigue utilizando el marcador indeleble o los tipos para marcar las pistas, esto ya no se puede permitir, porque evita que el ingeniero desarrolle la tecnología que la época demanda.

Es obligación de los actuales profesionistas el improvisar técnicas más efectivas, a precios que los estudiantes puedan cubrir. El prototipo propuesto es un robot que pretende actualizar la vieja escuela, de una manera económica y que suprima los riesgos que representan los químicos para el estudiante y para el medio ambiente.

La solución propuesta es atacar el cobre con un método mecánico (descobreando la placa fenólica, con una fresa), esto ayudado por un dispositivo autómata, basado en elementos mecánicos, electrónicos y de un software especialista instalado en una PC. Se demuestra con base en la experimentación que este prototipo es viable y eficaz, además que puede ser perfeccionado en los bloques que se compone.

Estructura de la tésis

La tésis que se presenta a continuación consta de siete capítulos los cuales se describen brevemente a continuación:

- **Estudio del estado del arte.**

En este capítulo se hace una breve explicación de antecedentes y proyectos similares.

- **Parte mecánica.**

En este capítulo se describe la estructura mecánica del AHIETI, tanto en su diseño y su construcción, respectivamente, así como también la descripción de cada una de sus piezas.

- **Parte electrónica.**

Describe la interfaz que nos permite controlar la parte mecánica con los motores a paso a través de un sistema de cómputo.

- **Parte del sistema de cómputo.**

Se hace una breve descripción de como se logró transformar desde un esquema a órdenes útiles para interpretarlos al sistema electrónico.

- **Conclusiones.**

Describe las conclusiones del trabajo realizado y se mencionan comentarios acerca del desempeño en tiempo real, así como las perspectivas del trabajo futuro.

Capítulo 2

Antecedentes y Estudio del Estado del Arte

2.1. Definiciones

El término robot nace en 1921, en la obra teatral R. U. R. (Rosum's Universal Robot) del novelista y autor dramático checo Karel Capek en cuyo idioma la palabra robota significa fuerza de trabajo o servidumbre. En esta época donde comienza la producción de numerosas fábricas provocando esto una discusión entre el poder de las máquinas y la dominación del hombre en el ámbito industrial. Esto es en cuanto al concepto, si hablamos de un robot ya en forma tendríamos que hablar de tantas personas e inventos que dieron paso a los actuales robots, el trabajo con motores de Tesla, el brazo programable de Devol, el primer robot industrial por Engelberger, etc.

Más adelante, mediante el control automático se pretende realizar ingenios que permitan gobernar cualquier proceso sin la intervención de agentes exteriores.

La Automatización Industrial comienza en el siglo XIX, pero no es hasta después de la Segunda Guerra Mundial cuando comienza a extenderse en forma importante en todos los sectores industriales. La aparición del microprocesador en 1972 suministra un impulso decisivo al control por computadora, (asunto que veremos a detalle posteriormente).

Al pensar en el término robot nos remontamos a la imagen clásica de un humanoide el cual es capaz de realizar el mismo conjunto de actividades físicas que un ser humano. **O en su defecto una máquina capaz de reproducir movimientos y comportamientos de los seres vivos.**

En cuanto a este concepto clásico, es importante hacer la notación que en su momento realizó Leonardo Torres Quevedo: Los antiguos autómatas imitaban la apariencia y movimientos de los seres vivos, lo cual no tiene interés práctico; lo que yo busco es una clase de aparatos que, sin la necesidad de reproducir los gestos más visibles del hombre, intentan obtener los mismos resultados que una persona.

Un robot es una máquina para propósitos generales, en caso contrario la máquina adquiere el nombre del propósito específico, es un dispositivo generalmente mecánico, que desempeña tareas automáticamente, ya sea de acuerdo a supervisión humana directa, a través de un programa predefinido o siguiendo un conjunto de reglas generales, utilizando técnicas de inteligencia artificial. Generalmente estas tareas reemplazan, asemejan o extienden el trabajo humano, como tareas altamente repetitivas, peligrosas o simplemente tareas que para el hombre serían difíciles de realizar por nuestras capacidades físicas limitadas.[1] En la figura 2.1 se muestra el esquema de un robot. En esta se identifican como partes componentes: sistema mecánico, actuadores, fuente de alimentación, sensores internos y externos, herramienta final, además de la computadora que generalmente aloja el sistema de control.[4]

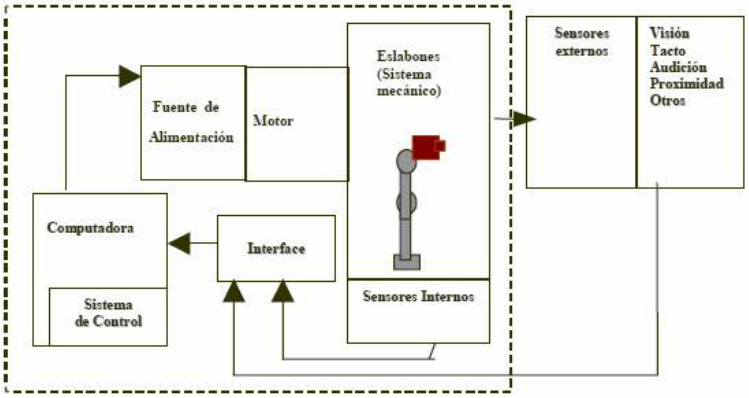


Figura 2.1: Robot y su Interacción con el Entorno.

2.2. Clasificación de los robots

Los robots se clasifican en cuanto a su estructura funcional, dentro de la cual encontramos tres grandes grupos:

- Robots Manipuladores
- Robots Móviles
- Robots Autónomos - Telerrobótica.

2.2.1. Robots manipuladores

Según la definición del Robot Institute of America , un robot industrial es un manipulador programable multifuncional diseñado para mover materiales, piezas, herramientas o dispositivos especiales mediante movimientos variados, programados para la ejecución de distintas tareas.

2.2.2. Robots móviles

Son una extensión de los robots manipuladores restringidos al alcance de una estructura sujetada, su relevancia consiste en buscar una autonomía limitando todo lo posible la intervención humana.

2.2.3. Robots autónomos y telerrobótica

De acuerdo con su grado de autonomía, los robots pueden clasificarse en

- Robots teleoperados
- Robots de funcionamiento repetitivo
- Robots autónomos o inteligentes. En los robots teleoperados las tareas de percepción del entorno, planificación y manipulación compleja son realizadas por humanos. Se utilizan para trabajos en una localización remota (acceso difícil, medios contaminados o peligrosos), en tareas de automatizar y en entornos no estructurados.

Los robots de funcionamiento repetitivo son la mayor parte de los que se emplean en cadenas de producción industrial. Trabajan normalmente en tareas predecibles e invariantes, con una limitada percepción del entorno.

Los robots autónomos o inteligentes son los más evolucionados desde el punto de vista de procesamiento de información, son máquinas capaces de percibir, modelar el entorno, planificar y actuar para alcanzar objetivos sin la intervención de supervisores humanos.[1]

2.2.4. Tipos de articulaciones

Los principales son:

- Rotación (rotación alrededor del eje de la articulación)
- Prismática (traslación a lo largo del eje)
- Cilíndrica (rotación y traslación)
- Planar (desplazamiento en un plano)
- Esférica (tres giros en tres direcciones perpendiculares).

2.2.5. Estructuras básicas

Por la forma en que se especifica la posición, ver figura 2.2:

- Cartesiana (coordenadas cartesianas)
- Cilíndrica (coordenadas cilíndricas)
- Polar (coordenadas polares)
- Angular (coordenadas angulares)

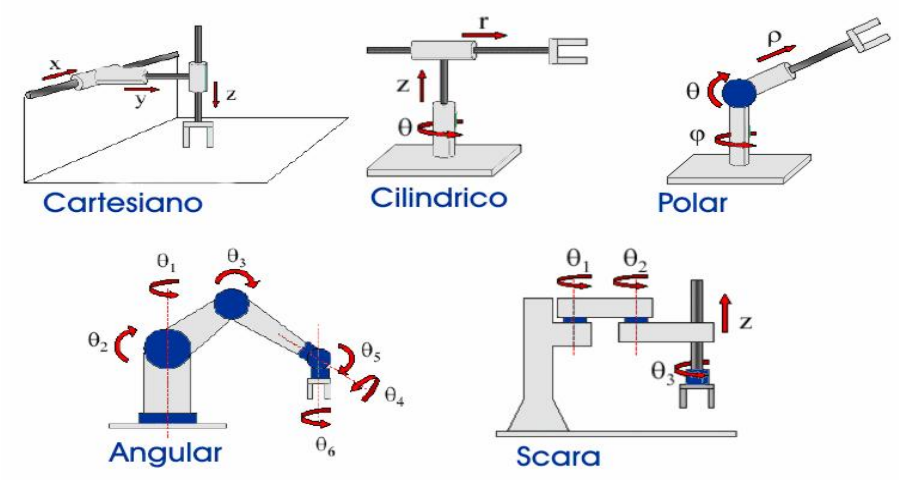


Figura 2.2: Configuraciones básicas de robots manipuladores industriales.

2.3. El control numérico

El control numérico se puede definir de una forma genérica como un dispositivo de automatización de una máquina que, mediante una serie de instrucciones codificadas (el programa), controla su funcionamiento. Cada programa establece un determinado proceso a realizar por la máquina, con lo que una misma máquina puede efectuar automáticamente procesos distintos sin más que sustituir su programa de trabajo. Permite, por tanto, una elevada flexibilidad de funcionamiento con respecto a las máquinas automáticas convencionales en las que los automatismos se conseguían mediante sistemas mecánicos o eléctricos difíciles y a veces casi imposible de modificar.

Los elementos básicos del control numérico son:

1) El programa, que contiene toda la información de las acciones a ejecutar.

2) El control numérico, que interpreta estas instrucciones, las convierte en las señales correspondientes para los órganos de accionamiento de la máquina y comprueba los resultados.

3) La máquina, que ejecuta las operaciones previstas.

Las máquinas y herramientas pueden operarse manualmente o mediante control automático. Las primeras máquinas utilizaban volantes para estabilizar su movimiento y poseían sistemas complejos de engranajes y palancas para controlar la máquina y las piezas en que trabajaba. Poco después de la II Guerra Mundial se desarrollaron los sistemas de control numérico. Las máquinas de control numérico utilizaban una serie de números perforados en una cinta de papel o tarjetas perforadas para controlar su movimiento. En los años 60's se añadieron computadoras para aumentar la flexibilidad del proceso. Tales máquinas se comenzaron a llamar máquinas CNC (Control Numérico por Computadora). Las máquinas de control numérico y CNC pueden repetir secuencias una y otra vez con precisión, y pueden producir piezas mucho más complejas que las que pueda hacer el operario más experimentado.[10]

A medida que el desarrollo de la microelectrónica y la informática se aplica a los controladores numéricos, se potencian extraordinariamente las funciones que permiten desarrollar, simplificándolos a la vez, los procedimientos de programación y operación de las máquinas, de tal manera que las CNC (control numérico con ordenador) que se construyen hoy día sólo conservan de los primitivos CN los principios básicos de funcionamiento.

En función de las capacidades de proceso y de memoria de las CNC han evolucionado también las técnicas y lenguajes de programación. Desde los primeros programas lineales en lenguaje máquina a la programación asistida por ordenador, gráfica e interactiva, existe un amplio espectro de sistemas y lenguajes de programación[2]

Las primeras máquinas de coordenadas en realidad fueron las máquinas de trazos, que son instrumentos con tres ejes mutuamente perpendiculares a n de alcanzar coordenadas volumétricas en un sistema cartesiano para localizar un punto en el espacio sobre una pieza con tres dimensiones. Se conoce que a finales del año 1962, la firma italiana DEA construyó la primera máquina de medición cerca de Turín, Italia.

Posteriormente en 1973 la compañía Carl Zeiss creó una máquina, equipada con un palpador, un ordenador y un control numérico.

Desde entonces han surgido muchas marcas y modelos de máquinas de coordenadas, que se distinguen entre sí por sus materiales de fabricación utilizados, software utilizado, versatilidad, alcances de medición, etc.[5]

El CNC se utiliza para controlar los movimientos de los componentes de una máquina por medio de números. Las máquinas y herramientas con control numérico se clasifican de acuerdo al tipo de operación de corte.

Un nuevo enfoque para optimizar las operaciones de maquinado es el control adaptativo. Mientras el material se esté maquinando, el sistema detecta las condiciones de operaciones como la fuerza, temperatura en la punta de la herramienta, rapidez de desgaste de la herramienta y acabado superficial. Convierte estos datos en control de avance y velocidad

que permita a la máquina cortar en condiciones óptimas para obtener máxima productividad. Se espera que los controles adaptativos, combinados con los controles numéricos y las computadoras, produzcan una mayor e ciencia en las operaciones de trabajos con los metales.

2.4. Proyectos similares

En el principio básico del robot cartesiano se manejan numerosos ejemplos desarrolla- dos formalmente como informalmente, es decir, existen actualmente una gran diversidad de máquinas y dispositivos capaces de realizar movimientos traslacionales sobre los ejes cartesianos X , Y , y Z ; sin embargo también es importante señalar importantes diferencias, en cuanto al principio mecánico de su funcionamiento, el sistema de control (en algunos de estos inexistente), y por supuesto la interfase para el operador humano.

Tomando en cuenta estos aspectos, se investigó el entorno tecnológico y teórico del AHIETI encontrando algunos prototipos con funcionamiento parecido o al menos principio teórico similar al propuesto en el presente trabajo.

2.4.1. Método de planchado

Para poder realizar PCBs con este método no es necesario utilizar ningún tipo de máquina o robot.

Este método consiste en elaborar el circuito PCB deseado en la PC, con el software especializado como el Eagle PCB Express, Express PCB, TraxMaker o cualquier otro, posteriormente se limpia correctamente la placa fenólica, que este libre de grasa y polvo, se calienta la plancha a la más alta temperatura, se imprime el circuito en un acetato e inmediatamente se coloca en la placa fenólica y se comienza con el planchado, hasta ver que las pistas se hallan adherido a la placa fenólica, finalmente se despega el acetato y se sumerge la placa fenólica en cloruro férrico.

2.4.2 Diseño y fabricación de circuitos impresos mediante una PC.

Este es el nombre del trabajo realizado en la Facultad de Ingeniería Eléctrica en la Universidad Michoacana de San Nicolás de Hidalgo.[1] Trabajo en el cual usa el editor de circuitos impresos llamado Eagle[2], el cual consiste en un software capaz de diseñar los circuitos en una vista en la cual se ve la forma en la que el circuito quedará impreso en una placa fenólica. Otro software utilizado en este trabajo es el KCam, el permite a través del puerto paralelo dar instrucciones a la herramienta para la manufactura, se mueve en tres dimensiones para realizar la tarea indicada (perforar, cortar y tallar el cobre).

[1] Víctor Hugo Chávez Urbina y Carlos Manuel Sánchez González en Noviembre del 2004.
[2] Para Windows 95/NT Versión 4.1

En este trabajo usan motores a paso unipolares, los que son controlados desde una PC por medio de una tarjeta de control a través de un puerto paralelo. El driver usado para la tarjeta de control es el L297-298

La herramienta utilizada para el corte y perforación, se le denomina Herramienta de Corte que se muestra en la gura2.3, que es un taladro Dremel, el mismo mini-drill usado en el proyecto AHIETI.

Figura 2.3: Herramienta de corte

El mecanismo que se utilizó para este proyecto está compuesto por una base móvil de cinco travesaños, en los cuales cada uno tiene un motor a paso unipolar, que permite el movimiento de la herramienta de corte a lo largo de los tres ejes, gura 2.4. Para el eje X solamente tenemos un solo travesaño con un motor a paso, el eje Y cuenta con dos travesaños con un motor a paso cada uno, el eje Z, al igual que el eje Y cuenta con dos travesaños, con un motor a paso cada uno.

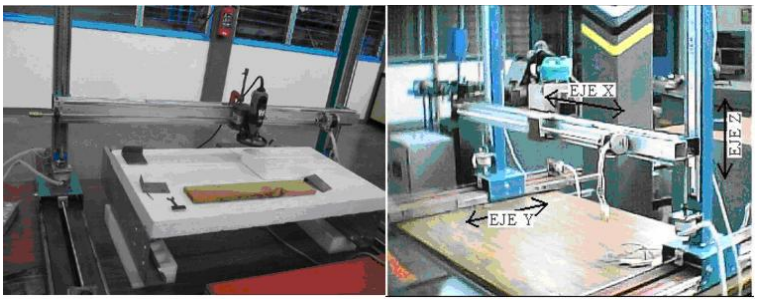


Figura 2.4: Estructura del mecanismo de Víctor Hugo Chávez Urbina y Carlos Manuel Sánchez González, de la Facultad de Ingeniería Eléctrica en la Universidad Michoacana de San Nicolás de Hidalgo

Finalmente con la ayuda del puerto paralelo, la tarjeta de control, la herramienta de

corte y el mecanismo mecánico. En la figura 2.5 podemos observar una placa ya perforada y lista para el montaje de los componentes electrónicos.

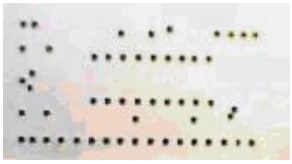


Figura 2.5: Placa Perforada.

Una desventaja que presenta este proyecto es que el mecanismo requiere mayor espacio, comparado con el AHIETI. Por otra parte la combinación de los motores a paso con el tornillo sin n hace que la tarea se ejecute lentamente debido a que el avance depende de el giro del tornillo sin n, también se utilizan demasiados motores a paso, por consiguiente el número de circuitos de potencia se incrementa.

Otro de los proyectos similares es el que a continuación presentamos, no tenemos el nombre de dicho trabajo, tampoco de quien lo fabrico. Pero lo llamaremos mesa móvil.

2.4.3. Mesa móvil

Consiste en un mecanismo similar al anterior, es una estructura igual de robusta, ya que utiliza perfiles de acero como base de todo el sistema, echas de soporte como el usado en el AHIETI, que más adelante detallaremos, y tornillos sin n, como se muestra en la figura 2.6. Las piezas en color verde contienen un par de bujes donde se incrustan las echas de soporte y así se desliza el sistema, se puede notar que en medio de las dos echas hay un tornillo sin n con un motor a paso para mover la base del taladro.

Se aprecia que la mesa en donde se colocan los objetos a trabajar se mueve de izquierda a derecha o Vertical u horizontal (depende del punto de vista). Para la base del taladro se aplicó el mismo sistema, dos echas con un tornillo sin n en medio de ellas, con su respectivo motor a paso.

Figura 2.6: Mesa Móvil autor desconocido

A) Vista General B) Vista Lateral

2.4.4. T-Tech, Inc “QUICK CIRCUIT”.

Modelo S (estándar)

La gura2.7 muestra el modelo estándar de Quick Circuit de la compañía T - Tech provee todo para producir circuitos impresos, realiza un prototipo en unas horas. Con una gran variedad de áreas de trabajo y configuraciones de sistema, usted será capaz de encontrar un sistema compatible con las necesidades de su diseño. El modelo estándar tiene un software controlado, velocidades de 8,000 y 24,000 rpm Con el sistema IsoPro que contiene el software, el cuál permite la conversión de datos automática de archivos CAD. El software también tiene la capacidad de Importar archivos DXF y exportar los datos Gerber.

Es comúnmente usada para: educación, circuitos análogos, circuitos digitales de baja velocidad y fuentes de alimentación. [22]

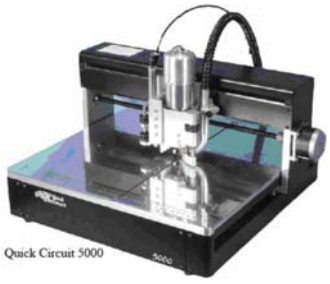


Figura 2.7: Quick Circuit 5000, Modelo S fabricado por T-Tech, Inc

Modelo HS (High Speed)

El modelo HS mostrado en la figura 2.8, de la compañía T-Tech, ha sido en las industrias el punto de referencia en la creación de prototipos. Con un rango de velocidades de 6,000 a 60,000 rpm, permite a usuarios el rendimiento óptimo y la precisión necesitada en el proceso de creación de prototipos. La HS puede realizar pistas de .004 " para RF. Este modelo a diferencia del anterior, es que en el eje Z es neumático e ideal para los laboratorios de RF. El sistema también tiene el IsoPro software, mencionado anteriormente.

Comúnmente usado en circuitos digitales de alta velocidad, R/F, microondas. [22]

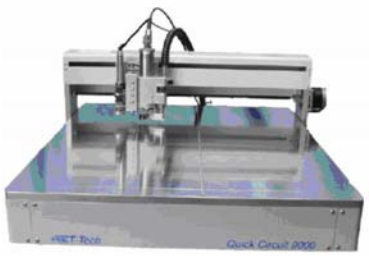


Figura 2.8: Quick Circuit 9000, Modelo HS fabricado por T-Tech, Inc

Modelo HF (High Frecuency)

El modelo HF que se muestra en la figura 2.9, de alta frecuencia, se centra en los circuitos de RF y los de microondas con reducciones más suaves el tiene una velocidad máxima de 100,000 rpm. El El eje de Z neumático también se presenta con este modelo. Este sistema también viene con el software de IsoPro.[22]

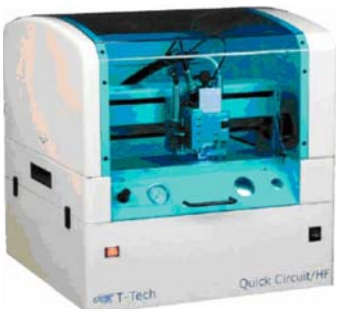


Figura 2.9: Quick Circuit/HF, Modelo HF fabricado por T-Tech, Inc

Las diseños realizados por estos equipos son de buena calidad, en la Universidad Politécnica de Tulancingo cuentan con el modelo Quick Circuit 7000, con este equipo se hizo la

placa que se muestra en la figura 2.10.

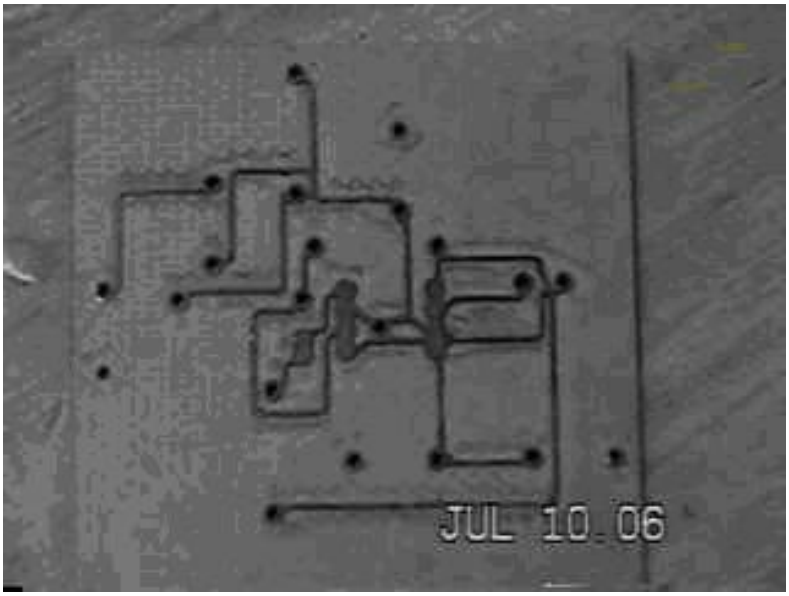


Figura 2.10: Placa Realizada en la Universidad Politécnica de Tulancingo, por los Alumnos de la misma.

2.4.5. Flexicam series

Otro de los proyectos similares al AHIETI son las fresadoras Flexicam Series [3].

Las fresadoras CNC VL, STEALTH, PRO se muestran en la figura 2.11 son ideales para el procesamiento y corte de PVC, aluminio, metales no ferrosos, madera, compuestos sintéticos, otros materiales y que requieran una sólida estructura.

La base de la fresadora está diseñada para soportar un trabajo intenso y libre de vibraciones. En contraste con la fresadora Eco, que utiliza motores paso a paso, la fresadora Stealth está equipada con motores servo AC. Estructura totalmente en acero, mecanizada y libre de tensiones.

El sistema Stealth se conecta a la estación de trabajo, bajo Windows 95/98-/NT/W2000, mediante una conexión en red. El sistema se maneja mediante un cómodo controlador manual.

Figura 2.11: Flexicam Series Fabricadas por Veroe control S.L. en Madrid.

El modelo cobra realiza el grabado y mecanizado de altas prestaciones en 2D y 3D, gura 2.12. [6]

Figura 2.12: Flexicam Cobra Fabricada por Veroe control S.L. en Madrid.

[3]Fabricadas por Veroe control S.L. en Madrid.

Capítulo 3

Diseño Mecánico

En un Robot o en una Máquina CNC, es muy importante el diseño mecánico, ya que sin un buen mecanismo se tiende a no obtener el resultado deseado, por eso es de suma impor- tancia contemplar todos los detalles, inclusive los mínimos, así como también es necesario conocer el material que se va a utilizar, y sus características generales.

3.1. Acero

Los metales y las aleaciones empleados en la industria y en la construcción pueden dividirse en dos grupos principales: ferrosos y no ferrosos. Ferroso viene de la palabra ferrum que los romanos empleaban para el fierro o hierro, por lo tanto, los materiales ferrosos son aquellos que contienen hierro como su ingrediente principal; es decir, las numerosas calidades del hierro y el acero.

Los materiales no ferrosos no contienen hierro. Estos incluyen el aluminio, magnesio, zinc, cobre, plomo y otros elementos metálicos. Las aleaciones, el latón y el bronce, son una combinación de algunos de estos metales no ferrosos y se les denomina aleaciones no ferrosas. Uno de los materiales de fabricación y construcción versátil, adaptable y ampliamente usado es el acero. A un precio relativamente bajo, el acero combina la resistencia y la posibilidad de ser trabajado, lo que se presta para fabricaciones mediante muchos métodos. Además, sus propiedades pueden ser manejadas de acuerdo a las necesidades específicas mediante tratamientos con calor, trabajo mecánico, o mediante aleaciones.

¿Qué es el Acero?

El Acero es básicamente una aleación o combinación de hierro y carbono (alrededor de 0,05 % hasta menos de un 2 %). Algunas veces otros elementos de aleación específicos tales como el Cr (Cromo) o Ni (Níquel) se agregan con propósitos determinados.

Ya que el acero es básicamente hierro altamente re nado (más de un 98 %), su fabricación comienza con la reducción de hierro el cual se convierte más tarde en acero.

El hierro puro es uno de los elementos del acero, por lo tanto consiste solamente de

un tipo de átomos. No se encuentra libre en la naturaleza ya que químicamente reacciona con facilidad con el oxígeno del aire para formar óxido de hierro (herrumbre). El óxido se encuentra en cantidades significativas en el mineral de hierro, el cual es una concentración de óxido de hierro con impurezas y materiales térreos.

3.1.1. Clasificación del acero

Los diferentes tipos de acero se clasifican de acuerdo a los elementos de aleación que producen distintos efectos en el Acero:

Aceros al carbono

Más del 90 % de todos los aceros son aceros al carbono. Estos aceros contienen diversas cantidades de carbono y menos del 1,65 % de manganeso, el 0,60 % de silicio y el 0,60 % de cobre. Entre los productos fabricados con aceros al carbono figuran máquinas, carrocerías de automóvil, la mayor parte de las estructuras de construcción de acero, cascos de buques, somieres y horquillas.

Aceros aleados

Estos aceros contienen un proporción determinada de vanadio, molibdeno y otros elementos, además de cantidades mayores de manganeso, silicio y cobre que los aceros al carbono normales. Estos aceros de aleación se pueden subclasificar en:

Estructurales Son aquellos aceros que se emplean para diversas partes de máquinas, tales como engranajes, ejes y palancas. Además se utilizan en las estructuras de edificios, construcción de chasis de automóviles, puentes, barcos y semejantes. El contenido de la aleación varía desde 0,25 % a un 6 %.

Para herramientas Aceros de alta calidad que se emplean en herramientas para cortar y modelar metales y no-metales. Por lo tanto, son materiales empleados para cortar y construir herramientas tales como taladros, escariadores, fresas, terrajas y machos de roscar.

Especiales Los Aceros de Aleación especiales son los aceros inoxidables y aquellos con un contenido de cromo generalmente superior al 12 %. Estos aceros de gran dureza y alta resistencia a las altas temperaturas y a la corrosión, se emplean en turbinas de vapor, engranajes, ejes y rodamientos.

Aceros de baja aleación ultrarresistentes

Esta familia es la más reciente de las cuatro grandes clases de acero. Los aceros de baja aleación son más baratos que los aceros aleados convencionales ya que contienen cantidades

menores de los costosos elementos de aleación. Sin embargo, reciben un tratamiento especial que les da una resistencia mucho mayor que la del acero al carbono. Por ejemplo, los vagones de mercancías fabricados con aceros de baja aleación pueden transportar cargas más grandes porque sus paredes son más delgadas que lo que sería necesario en caso de emplear acero al carbono. Además, como los vagones de acero de baja aleación pesan menos, las cargas pueden ser más pesadas. En la actualidad se construyen muchos edificios con estructuras de aceros de baja aleación. Las vigas pueden ser más delgadas sin disminuir su resistencia, logrando un mayor espacio interior en los edificios.

Aceros inoxidables

Los aceros inoxidables contienen cromo, níquel y otros elementos de aleación, que los mantienen brillantes y resistentes a la herrumbre y oxidación a pesar de la acción de la humedad o de ácidos y gases corrosivos. Algunos aceros inoxidables son muy duros; otros son muy resistentes y mantienen esa resistencia durante largos periodos a temperaturas extremas. Debido a sus superficies brillantes, en arquitectura se emplean muchas veces con fines decorativos. El acero inoxidable se utiliza para las tuberías y tanques de refinerías de petróleo o plantas químicas, para los fuselajes de los aviones o para cápsulas espaciales. También se usa para fabricar instrumentos y equipos quirúrgicos, o para fijar o sustituir huesos rotos, ya que resiste a la acción de los fluidos corporales. En cocinas y zonas de preparación de alimentos los utensilios son a menudo de acero inoxidable, ya que no oscurece los alimentos y pueden limpiarse con facilidad.[7]

Basándonos en la clasificación de el acero para la fabricación de algunas piezas de el AHIETI v 1.0.0 y AHIETI v 2.1.1 se utilizó el acero al carbón, por la facilidad de conseguirlo y por su costo relativamente bajo a comparación de los demás aceros.

3.2. Aluminio

El aluminio no se encuentra en estado nativo. Abunda mucho en la naturaleza combinado, integrando arcillas y feldespatos. Se obtiene por métodos electrolíticos de la criolita o fluoruro de aluminio y sodio. Es de color blanco azulado, brillante, estructura fibrosa, más duro que el estaño pero menos que el cobre y el zinc.

Es inalterable al aire; expuesto a la humedad forma en su superficie una película protectora a la herrumbre que lo inmuniza contra la acción atmosférica y el agua. Es muy dúctil y maleable, pudiéndose obtener en Oliz y hojas como el oro. Funde a los 660 ° C. En frío no es atacable por los ácidos. Las aplicaciones del metal son múltiples.

Aparte de la fabricación de utensilios domésticos se usa en forma intensa en la fabricación de motores, aviones y piezas para la industria en general. En construcción se lo emplea cuando el factor peso es importante; se obtiene en forma de chapas de 0,02 a 5 mm de espesor, alambres de 1 a 5 mm de diámetro y ángulos de lados iguales o desiguales. El mayor uso es combinado, en especial con el cobre. Como la película protectora tiene la

propiedad de reaccionar con las anilinas y absorberlas, permite el coloreado del metal con tintes atractivos, con lo que se tiene un campo de aplicación muy grande para la decoración.

El empleo del aluminio en construcción es cada vez mayor. Actualmente se está probando al aluminio para sustituir al hierro en aquellos casos en que la duración de éste impone una atención constante. Así, en la construcción del puente Gras River Bridge de Massena, EEUU, se empleó exclusivamente aleación de aluminio Alcoa 14-S.T de la Aluminium Company of America. Con ello se obtuvo un puente muy liviano y muy resistente a la acción altamente corrosiva de una atmósfera industrial. Las aleaciones ofrecidas al comercio, poseen cualidades mecánicas que dependen de la composición.

También se está empleando el aluminio para alivianar puentes existentes, sustituyendo las vigas de acero de los tableros por vigas de aleación de aluminio.

3.2.1. Aplicaciones del aluminio.

La combinación de la ligereza con resistencia y alta conductibilidad eléctrica y térmica es la propiedad que convirtió al aluminio y sus aleaciones en materiales de construcción importantísimos para la construcción de aviones, automóviles, máquinas de transporte, electrotecnia, fabricación de motores de combustión interna, etc.

Un volumen dado de aluminio pesa menos que 1/3 del mismo volumen de acero. Los únicos metales más ligeros son el litio, el berilio y el magnesio. Debido a su elevada proporción resistencia-peso es muy útil para construir aviones, vagones ferroviarios y automóviles, y para otras aplicaciones en las que es importante la movilidad y la conservación de energía.

Por su elevada conductividad térmica, el aluminio se emplea en utensilios de cocina y en pistones de motores de combustión interna. Solamente presenta un 63 % de la conductividad eléctrica del cobre para alambres de un tamaño dado, pero pesa menos de la mitad. Un alambre de aluminio de conductividad comparable a un alambre de cobre es más grueso, pero sigue siendo más ligero que el de cobre. El peso tiene mucha importancia en la transmisión de electricidad de alto voltaje a larga distancia, y actualmente se usan conductores de aluminio para transmitir electricidad a 700.000 voltios o más.

El metal es cada vez más importante en arquitectura, tanto con propósitos estructurales como ornamentales. Las tablas, las contraventanas y las láminas de aluminio constituyen excelentes aislantes. Se utiliza también en reactores nucleares a baja temperatura porque absorbe relativamente pocos neutrones. Con el frío, el aluminio se hace más resistente, por lo que se usa a temperaturas criogénicas. El papel de aluminio de 0,018 cm de espesor, actualmente utilizado en usos domésticos, protege los alimentos y otros productos perecederos.

Debido a su poco peso, a que se moldea fácilmente y a su compatibilidad con comidas y bebidas, el aluminio se usa mucho en contenedores, envoltorios flexibles, botellas y latas de fácil apertura. El reciclado de dichos recipientes es una medida de conservación de la energía cada vez más importante. La resistencia a la corrosión, al agua del mar también lo hace útil para fabricar cascos de barco y otros mecanismos acuáticos. Se puede preparar una amplia gama de aleaciones recubridoras y

aleaciones forjadas que proporcionen al metal más fuerza y resistencia a la corrosión o a las temperaturas elevadas. Algunas de las nuevas aleaciones pueden utilizarse como planchas de blindaje para tanques y otros vehículos militares.[8]

3.2.2. Aleaciones de aluminio.

La evolución técnica continúa experimentando y aplicando nuevas aleaciones de aluminio, entre ellas las más corrientes son las que tienen como componentes principales el cobre y el silicio, cada una de las cuales le incluyen características particulares. Aleado con el cobre, éste le disminuye el inicio del punto de fusión, produciéndose a partir de los 530 ^{o}C, pero aumenta la resistencia a la rotura y su límite elástico, tiene el inconveniente de reducir su resistencia a los agentes atmosféricos aumentando su fragilidad.

El silicio al 12 % forma una aleación eutéctica (homogénea), disminuyendo también el punto de fusión a unos 575 ^{o}C pero con la ventaja sobre el anterior de aumentar su resistencia a los agentes atmosféricos y recibir un buen moldeo. A estas aleaciones se les adiciona, buscando mejorar determinadas condiciones, en porcentajes entre el 0,2 y el 2 %, son éstos el manganeso, el níquel, el titanio, el tungsteno, el zinc y el cobalto.[9]

3.3. AHIETI v 1.0.0

La realización mecánica del AHIETI v 2.0.0 esta basada básicamente en el mecanismo de las máquinas de escribir eléctricas y de algunas impresoras, y en una mejora del diseño ya antes realizado por los autores de este trabajo, las partes de que consta son totalmente diseñadas y elaboradas por los mismos. Para ello se realizaron dos diseños mecánicos, en ambos casos se tuvo que aprender el uso correcto de maquinaria como: Torno, Fresadora, Taladro de Banco, y de otras herramientas.

Para poder llegar al diseño actual del AHIETI v 2.0.0, primero se fabricó un mecanismo similar AHIETI v 1.0.0. constaba de una base cuadrada de aluminio, una Flecha de Soporte de Deslizamiento para Deslizadores (F.S.D.D) de acero inoxidable (en color azul de la figura 3.1), 1 tornillo sin n (color amarillo de la figura 3.1) para el eje X y otra F.S.D.D y un tornillo sin n para el eje Y. . Así como de otras piezas que explicaremos a continuación.

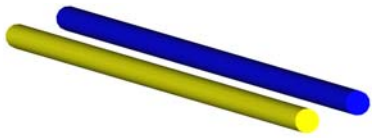


Figura 3.1: Flecha de Soporte de Deslizamiento para Deslizadores (F.S.D.D) y Tornillo sin Fin

3.3.1. Soportes

El soporte del AHIETI v 1.0.0 que ayudaba al funcionamiento del eje X estaba hecho de acero al carbón, se colocaba uno en cada esquina de la base de aluminio en los cuales se introducía una F.S.D.D y el tornillo sin n correspondiente a este eje, solo contaba con un barreno al centro de una de las caras la pieza, como se muestra en la figura 3.2.

Figura 3.2: Soportes AHIETI v1.0.0

3.3.2. Deslizadores

Deslizador izquierdo y derecho

Para poder unir las F.S.D.D y los tornillos sin n del eje X con el eje Y, se fabricaron los deslizadores en acero, dos para la base (izquierdo y derecho), ver figura 3.3. ambos deslizadores constan de tres barrenos que atraviesan al mismo, dos en la parte frontal en los cuales se introducía la F.S.D.D y el tornillo sin n que efectúan el movimiento en el eje Y, y otro barreno lateral, en el caso del deslizador derecho se introduce una F.S.D.D y en el deslizador izquierdo el barreno lateral consta de cuerda para poder introducir el tornillo sin n.

Figura 3.3: Deslizador AHIETI v 1.0.0

Estos deslizadores con ayuda de los soportes trabajan en el eje X y a su vez sirven para soportar las F.S.D.D y el tornillo sin n que conforman al eje Y

Deslizador principal.

El deslizador principal consta básicamente de dos piezas, Carro y Soporte del taladro, las cuales están fabricadas casi en su totalidad de aluminio y una pequeña pieza de Acero.

Cola de milano La cola de milano es un corte que se realiza en madera, acero, aluminio, bronce, etc. Todo depende de la aplicación, se le llama así a este corte, por el milano que es un ave de rapiña diurna, su cola tiene forma de trapecio como se aprecia en la figura 3.4.

Figura 3.4: Ejemplo del ave milano

Para poder ensamblar una pieza con este corte es necesario realizar dos partes (hembra y macho) a 45^o ó 60^o dependiendo del diseño y del diseñador. La gura 3.5 muestra lasdiferentes formas de cortes con cola de milano en madera.

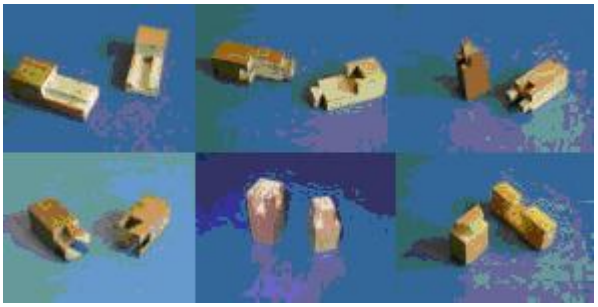


Figura 3.5: Diferentes cortes de Cola de Milano Carro

Como ya se hizo mención, la cola de milano requiere de dos cortes, para nuestro caso a 60^o, para poder ensamblarse, al carro se le hizo el corte hembra, que también pudo haber sido trabajado con el corte macho, para nuestro propósito cualquier forma de corte no afecta, el funcionamiento es el mismo.

El carro se desplaza en la F.S.D.D y en el tornillo sin n, sobre el eje Y.
En la gura 3.6 se muestra el carro, podemos ver los barrenos que atraviesan al carro para poder trasladarse con la ayuda del tornillo sin n, el cual se introduce en el barreno a la derecha de 1/2 pulgada, en el barreno izquierdo de 12 mm se coloca la F.S.D.D, así como en los casos anteriores.

El barreno que se encuentra en la cara superior del carro es para colocar el tornillo para el soporte del taladro.

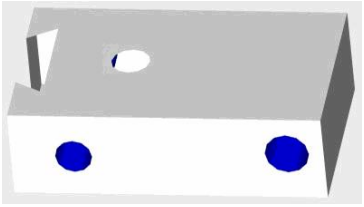


Figura 3.6: Carro AHIETI v 1.0.0

Soporte del taladro El Soporte del taladro trabaja en el eje Z con ayuda del tornillo que se encuentra perpendicular al carro y paralelo al soporte del Taladro, para poder sujetar el taladro al soporte del taladro fue necesario realizarle un medio barreno a la parte frontal del soporte. figura 3.7

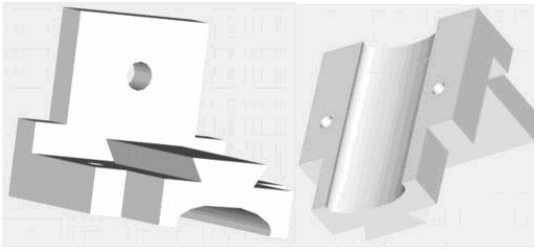


Figura 3.7: Soporte del Taladro AHIETI v 1.0.0

Ensamblando el carro, el soporte del taladro y el tornillo (color rojo de la figura 3.8) se obtiene el deslizador principal, figura 3.8.

Es importante hacer mención que la pieza que trabaja en el eje Y es el carro que el soporte del taladro con ayuda del tornillo sin fin paralelo a este, se encarga de el movimiento en el eje Z

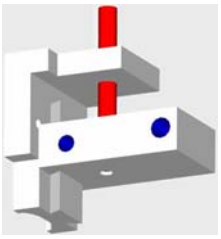


Figura 3.8: Deslizador Principal AHIETI v 1.0.0

Para poder dar movimiento al sistema en los tres ejes X, Y, Z se incorporarían 3 motores a paso, uno en cada tornillo sin n.

3.3.3. Ensamble AHIETI v 1.0.0

Para poder efectuar el movimiento de todo el sistema se pensó en incorporar en los tornillos sin n y a los motores a paso unos engranes, excepto el tornillo del deslizador principal, a este se le incorporaría un sprocket, el motor a paso también llevaría otra sprocket y con ellas su respectiva cadena, pero todo esto no se realizó físicamente.

Finalmente uniendo las piezas en el lugar correspondiente se obtiene el AHIETI v 1.0.0, figura 3.9, pero el funcionamiento no era el correcto, ya que cuando el tornillo sin n del eje X giraba movía al deslizador izquierdo y el deslizador derecho dejaba de avanzar impidiendo el avance de todo el sistema, esta fue la primer razón por la cual se decidió cambiar el diseño.

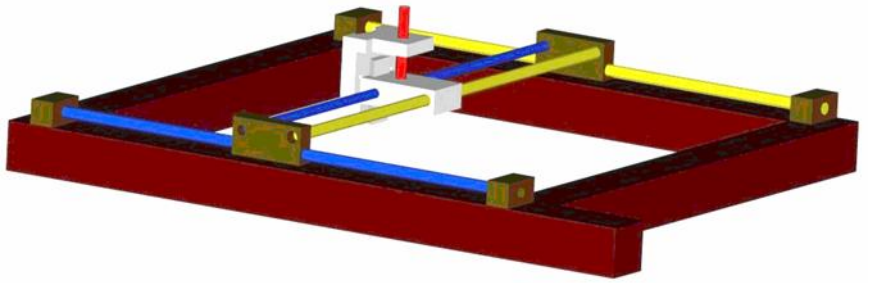
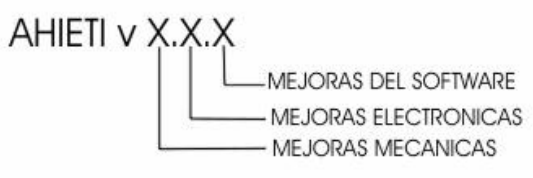


Figura 3.9: AHIETI v 1.0.0

También se observo que el movimiento en ambos casos (eje X y eje Y) era demasiado lento ya que dependía del giro del tornillo sin n y de los motores a paso, para dar solución a este problema se considero en cambiar el tornillo sin n de cuerda estándar por un tornillo de cuerda cuadrada. Pero por su alto costo nunca se incorporó a la estructura.

3.4. AHIETI v 2.0.0

Antes de continuar con el rediseño del AHIETI v 1.0.0, es importante hacer mención de cómo se van actualizando las versiones, es necesario saber que cada dígito va a ir cambiando conforme se concluya el proyecto, a continuación se muestra la descripción:



Así entonces ahora sabemos que el primer diseño corresponde al AHIETI v 1.0.0, hasta este momento no se ha trabajado en el software ni en lo electrónico.

Finalmente se llegó a una decisión, rediseñar el AHIETI v 1.0.0, ya que no era funcional. Para la elaboración del AHIETI v 2.0.0 se pensó en utilizar dos F.S.D.D para evitar el problema del AHIETI v 1.0.0, se continuaban usando los motores a paso. Las demás piezas a excepción del Deslizador principal se fabricarían nuevamente.

3.4.1. Deslizadores

Deslizador principal

La mayoría de las piezas realizadas en el anterior diseño ya no resultaban de utilidad, a excepción del deslizador principal, el carro se modificó ligeramente para que funcionara con el nuevo diseño, tal modificación consistió en quitarle la cuerda y desplazar el barreno aproximadamente unos 10 mm. hacia el centro de la pieza.

Los dos barrenos que se observan en la figura 3.10 son de 12.7 mm. en los cuales se introducen las Flechas de Soporte de Deslizamiento para el Deslizador Principal (F.S.D.D.P) de $\frac{1}{2}$ pulgada, donde el deslizador principal trabaja en el eje Y. Al soporte del taladro no se le hizo modificación alguna, únicamente se le adaptó un cincho para sujetar el taladro con el soporte, se utilizo un cinturón en forma de U de dos pulgadas de diámetro y finalmente el deslizador principal quedó como se aprecia en la figura 3.10.

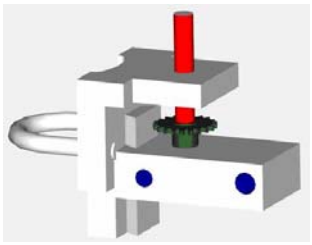


Figura 3.10: Deslizador Principal AHIETI v 2.0.0

Se puede observar que ahora el deslizador principal ya cuenta con su sprocket correspondiente, (color verde) y que efectúa el movimiento en el eje Z en el soporte del taladro.

Deslizador izquierdo

Los deslizadores también son de aluminio, se fabricaron dos uno derecho y uno izquierdo, estos sustituyen a los anteriores, trabajan en el eje X y también sirven para soportar las F.S.D.D del eje Y, ya que el sistema de F.S.D.D así lo requiere, como muestra la figura 3.11, podemos ver que lleva cuatro barrenos, los de color rojo de la figura 3.11 atraviesan todo el deslizador con diámetro de 11 mm, ya que en el se incorporan dos pequeñas echas de soporte de deslizamiento para deslizadores de aproximadamente 10 mm. de diámetro, los barrenos en color azul tienen un diámetro de 1/2 pulgada (aproximadamente 12.7 milímetros) y una profundidad de 15 mm aprox, en donde se incorporan dos F.S.D.D de 1/2 pulgada. La ranura en color amarillo tiene una abertura de 10 mm y una profundidad de 5 mm. Las dimensiones de ambos deslizadores son 100x70x35 mm.

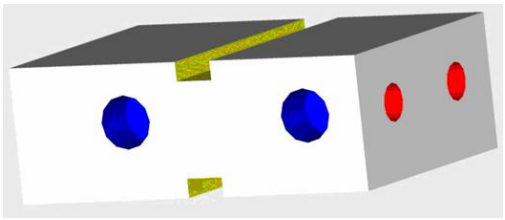


Figura 3.11: Deslizador Izquierdo AHIETI v 2.0.0

Deslizador derecho con sprocket

Este deslizador es idéntico al Deslizador Izquierdo con la única diferencia es que cuenta con un sprocket a un costado de él para que la cadena que jala al deslizador principal de vuelta en el sprocket y pueda completar el movimiento en el eje Y correctamente mostrado en la figura 3.12. El sprocket esta dentro de una base de aluminio con forma de U.

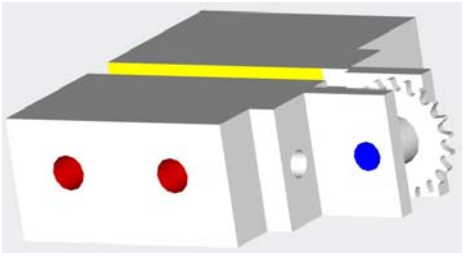


Figura 3.12: Deslizador Derecho Con Sprocket AHIETI v 2.0.0

3.4.2. Soportes

Soporte para Flechas de Soporte de Deslizamiento para Deslizadores con Placa para Motor a Paso (S.F.S.D.D.P.M.P)

El **S.F.S.D.D.P.M.P** se muestra en la figura 3.13, es otra pieza importante para el AHIETI v 2.0.0, ya que en este el motor es sujetado con una placa, cabe mencionar que consta de dos partes, la placa para el motor y el soporte para las F.S.D.D, ambas piezas son de acero, los tres barrenos en los soportes tienen un diámetro de 10 mm., la ranura es de 10 mm. de ancho por 17 mm. de alto, donde pasa la cadena. Las dos piezas (soporte y placa) se soldaron para que se unieran formando un ángulo de 90° y en cuanto se quiera desarmar esto sea más fácil.

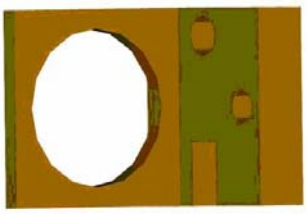


Figura 3.13: Soporte para Flechas de Soporte de Deslizamiento para Deslizadores con Placa para Motor a Paso AHIETI v 2.0.0

Soportes para Flechas de Soporte de Deslizamiento para Deslizadores con Sprocket (S.F.S.D.D.S)

En la figura 3.14 podemos observar que en la parte central de la pieza se encuentra el sprocket, el cual gira si el motor a paso que se encuentra al otro extremo lo hace, es el mismo funcionamiento del deslizador derecho con sprocket, ya que sin esta sería imposible el desplazamiento del sistema, el tamaño y la posición de los barrenos es la misma al del **S.F.S.D.D.P.M.P**

Figura 3.14: Soportes para Flechas de Soporte para Deslizadores con Sprocket AHIETI v 2.0.0

Estos soportes ayudan a realizar el movimiento de los deslizadores izquierdo y derecho con sprocket, en el eje X, con ayuda de las cadenas.

3.4.3. Cadenas y Sprockets

El paso de las cadenas y sprockets es 15, el número de dientes del sprocket es 21, mostrado en la figura 3.15, es obvio que para obtener el largo de las cadenas para nuestro propósito fue necesario unirlas con eslabones llamados candados ya que en el mercado no se encuentran de este tamaño en especifico.

Figura 3.15: Cadenas y Sprockets AHIETI v 2.0.0

3.4.4. Fresas para el descobreado

Las fresas para realizar el descobreado son de carburo de tungsteno como el mostrado en la figura 3.16, estas fresas son utilizadas por los odontólogos para sus intervenciones, existen diversos modelos, son fabricadas con diferentes materiales, el modelo utilizado para los propósitos del AHIETI v 2.0.0 es el FG-56 de MEISINGER.

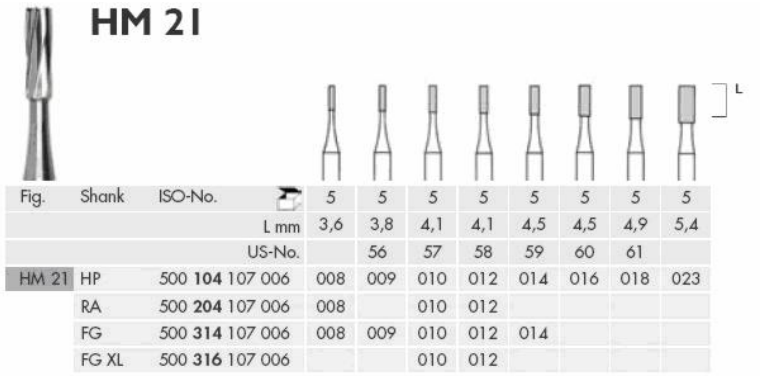


Fig.	Shank	ISO-No.	5	5	5	5	5	5	5	5
		L mm	3,6	3,8	4,1	4,1	4,5	4,5	4,9	5,4
		US-No.		56	57	58	59	60	61	
HM 21	HP	500 **104** 107 006	008	009	010	012	014	016	018	023
	RA	500 **204** 107 006	008		010	012				
	FG	500 **314** 107 006	008	009	010	012	014			
	FG XL	500 **316** 107 006			010	012				

Figura 3.16: Fresas de carburo tungsteno

3.4.5. Ensamble AHIETI v 2.0.0

Finalmente colocando las piezas mecánicas antes mencionadas en su lugar correspondiente obtenemos el AHIETI v 2.0.0, así como se muestra en la figura.3.17

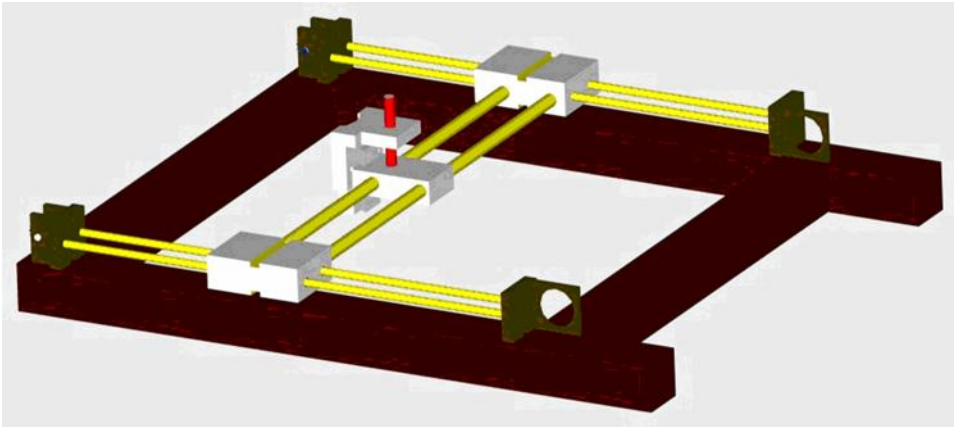


Figura 3.17: AHIETI v 2.0.0

Capítulo 4

Diseño Electrónico

Así como es de suma importancia el diseño mecánico en un robot o en una máquina CNC lo es el electrónico, ya que este último es el control, la potencia, etc. Como en todos los sistemas, incluso en el cuerpo humano, un órgano en menor o mayor proporción, necesita de otro para poder realizar una tarea específica, en este caso el sistema mecánico necesita de la electrónica y viceversa.

4.1. Motores a paso

Los motores a paso son de gran ayuda para mecanismos en donde se requieren movimientos muy precisos.

Estos motores poseen la habilidad de poder quedar enclavados en una posición o bien libres, si una o más de sus bobinas está energizada, el motor estará enclavado en la posición correspondiente y por el contrario quedará libre si no circula corriente por ninguna de sus bobinas.

Generalmente los motores a paso están constituidos por un rotor sobre el que van aplicados distintos imanes permanentes, figura 4.1, y por un cierto número de bobinas excitadoras bobinadas en su estator. Las bobinas son parte del estator y el rotor es un imán permanente. Toda la conmutación (o excitación de las bobinas) deber ser externamente manejada por un controlador.

Figura 4.1: Imán permanente

4.1.1. Tipos de motores a paso

Básicamente existen dos tipos de motor a paso de imán permanente, los bipolares y unipolares. figura 4.2

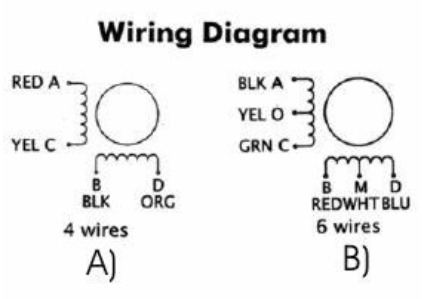


Figura 4.2: Estructura básica de un stepper motor bipolar (A) y unipolar (B)

Bipolar

Estos motores a paso generalmente tienen cuatro cables a la salida así como se muestra en la figura 4.2(a). Necesitan de ciertos controladores para ser usados, debido a que re- quieren del cambio de dirección del ujo de corriente a través de las bobinas en la secuencia apropiada para realizar el movimiento adecuado. En general es recomendable el uso de un Puente H (H-Bridge) integrados como el C.I. L293 mostrado en la gura4.3.

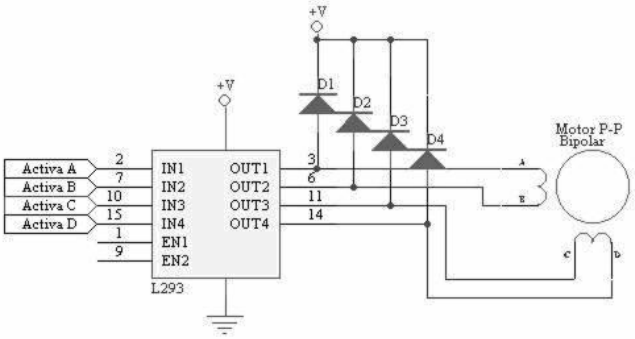


Figura 4.3: Controlador para un Stepper Motor Bipolar

Secuencias para controlar un motor a paso bipolar Como se mencionó anteriormente, estos motores a paso bipolares necesitan la inversión de la corriente que circula en sus bobinas con una secuencia determinada. Cada inversión de polaridad provoca el movimiento del eje en un paso, cuyo sentido de giro está determinado por la secuencia seguida. En la tabla 4.1 se puede ver la secuencia necesaria para controlar estos motores a paso.

PASO	TERMINALES			
	A	B	C	D
1	+V	-V	+V	-V
2	+V	-V	-V	+V
3	-V	+V	-V	+V
4	-V	+V	+V	-V

Cuadro 4.1: Secuencia para controlar un Stepper Motor Bipolar

Unipolar

Estos motores suelen tener 6 ó 5 cables a la salida como se muestra en la figura 4.2(b). Estos se caracterizan por ser más simple de controlar. En la figura 4.4 podemos apreciar un ejemplo del controlador ULN2803 conectado a un motor a paso unipolar, el cual es un array de 8 transistores tipo Darlington capaz de manejar cargas de hasta 500 mA. Las entradas de activación (A, B, C y D) pueden ser directamente activadas por un microcontrolador.

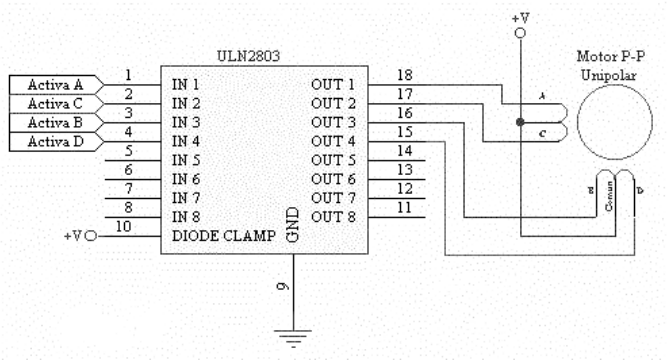
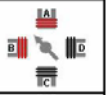
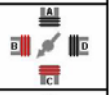
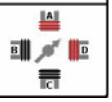


Figura 4.4: Controlador para un Stepper Motor Unipolar Secuencias para controlar un motor a paso unipolar

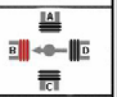
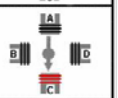
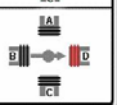
Existen tres secuencias posibles para este tipo de motores, (Secuencia normal, Secuencia wave drive, Secuencia del tipo medio paso). Todas las secuencias comienzan nuevamente con la bobina una vez alcanzado el paso final (4 u 8). Para revertir el sentido de giro, simplemente se deben ejecutar las secuencias en modo inverso.

Secuencia normal Esta es la secuencia más usada y la que generalmente recomienda el fabricante. Con esta secuencia el motor avanza un paso por vez, tabla 4.2, y debido a que siempre hay dos bobinas activas se obtiene un alto torque de paso y de retención.

Paso	Bobina A	Bobina B	Bobina C	Bobina D	Resultado
1	ON	ON	OFF	OFF	
2	OFF	ON	ON	OFF	
3	OFF	OFF	ON	ON	
4	ON	OFF	OFF	ON	

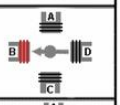
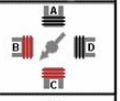
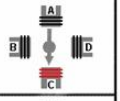
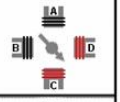
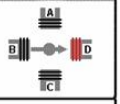
Cuadro 4.2: Secuencia normal para controlar un Steeper Motor unipolar

Secuencia wave drive En esta secuencia se activa solo una bobina a la vez como se aprecia en la tabla 4.3. En algunos motores esto brinda un funcionamiento suave. La desventaja con la secuencia anterior es que al energizar solo una bobina el torque de paso y retención es menor.

Paso	Bobina A	Bobina B	Bobina C	Bobina D	Resultado
1	ON	OFF	OFF	OFF	
2	OFF	ON	OFF	OFF	
3	OFF	OFF	ON	OFF	
4	OFF	OFF	OFF	ON	

Cuadro 4.3: Secuencia wave drive para controlar un Stepper Motor unipolar

Secuencia medio paso En esta secuencia se activan las bobinas de tal forma de brindar un movimiento igual a la mitad del paso real, es decir para los motores del AHIETI v 2.1.0 que son de 1.8º, con esta configuración el paso sería de 0.9º. Para ello se activan primero dos bobinas y luego solo una y así sucesivamente. Como se ve en la tabla 4.4, la secuencia completa consta de ocho movimientos.

Paso	Bobina A	Bobina B	Bobina C	Bobina D	Resultado
1	ON	OFF	OFF	OFF	
2	ON	ON	OFF	OFF	
3	OFF	ON	OFF	OFF	
4	OFF	ON	ON	OFF	
5	OFF	OFF	ON	OFF	
6	OFF	OFF	ON	ON	
7	OFF	OFF	OFF	ON	
8	ON	OFF	OFF	ON	

Cuadro 4.4: Secuencia de medio paso para controlar un Steeper Motor unipolar

Debido a que los motores a paso son dispositivos mecánicos y como tal deben vencer ciertas inercias, el tiempo de duración y la frecuencia de los pulsos aplicados es un punto importante para tener en consideración. Es decir que los motores a paso, en general, están limitados en cuanto a la velocidad que pueden alcanzar, ya que si se excede la frecuencia de pulsos que pueden soportar, estos tienden a reaccionar erróneamente, por ejemplo que comiencen a vibrar sin girar, que no se muevan en absoluto, etc.

Los motores a paso que usa el AHIETI v 2.1.0 son de la compañía ASTROSYN, modelo 23LM-C327-11V, hechos en Tailandia, consumen aproximadamente 1.3 A, cuando se energiza una bobina, el total de la corriente que consume cuando se aplica la secuencia normal al máximo de su velocidad es de 1.06 A, y con una baja velocidad es de 2.6 A.

Para determinar el ángulo de avance del motor a paso, se cuenta el número de pasos que necesita para completar un giro de 360^o, el motor a paso Astrosyn 23LM-C327-11V, debe dar 200 pasos normales para completar un movimiento de 360^o. Aplicando la ecuación 4.1 se obtiene que el ángulo es de 1.8^o.

$$] = \frac{360^o}{\text{totaldepasosen}360^o} \tag{4.1}$$

Es importante saber cual es el avance del motor cuando gira 1.8^o, para poder determinar esto se aplica la ecuación 4.2, del triángulo oblicuángulo, que no es otra cosa que el teorema de los senos.

$$b = \frac{a}{\sin\alpha}\sin\beta \tag{4.2}$$

Tomando como referencia la gura 4.5 y obteniendo los datos necesarios para poder aplicar la ecuación 4.2, conocemos $\beta = 1.8^o$, si sabemos que en cualquier triángulo los ángulos internos deben sumar 180^o y los ángulos opuestos a β son iguales, determinamos que α y γ son iguales a 89.1^o y que a y c tienen valores de 17.5mm, ya que el diámetro del sprocket es de 35mm, obtenemos el valor de b= 0.549mm.

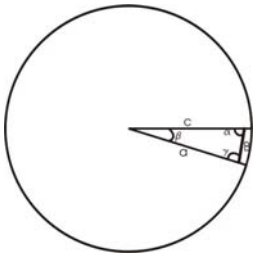


Figura 4.5: Triángulo Oblicuángulo

Así entonces tenemos que por cada paso de 1.8^o del motor a paso el sistema avanza 0.549mm.

4.2. Etapa de potencia y aislamiento

La etapa de potencia mostrado en la gura 4.6 del AHIETI v 2.1.0 esta compuesta esencialmente de dos partes: Transistores Darlington de potencia y Optoaisladores, para proteger al controlador, esta etapa es de suma importancia ya que como se mencionó en la sección 4.1, los motores a paso requieren de 2.6 A como máximo para brindar el torque que requiere el AHIETI v 2.1.0.

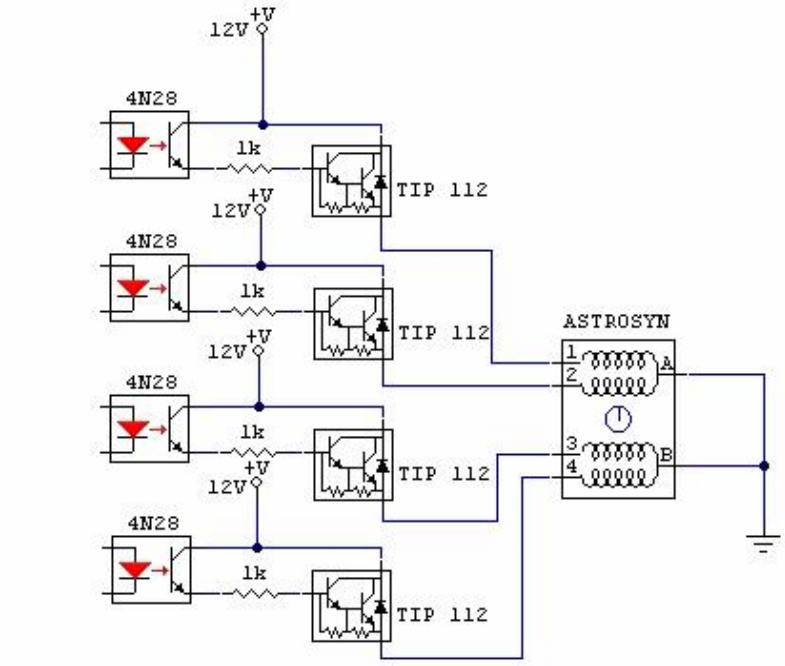


Figura 4.6: Etapa de Potencia

4.2.1. Transistores darlington de potencia

El par Darlington se usa ampliamente en amplificadores de potencia de audio, interruptores de motores de alta corriente y en otras aplicaciones de conmutación de potencia.[3]

Figura 4.7: TIP 112. El transistor utilizado para la potencia del AHIETI v 2.1.0 es el Tip 112[1], es un Transistor de Silicio Darlington NPN expuesto en la figura 4.7. El circuito equivalente de este Transistor se muestra en la figura 4.8.

[1]ver hoja de datos en apéndice D

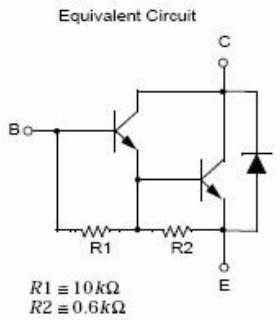


Figura 4.8: Circuito equivalente del TIP 112

4.2.2. Optoacopladores

También se denominan optoaisladores o dispositivos de acoplamiento óptico. Basan su funcionamiento en el empleó de un haz de radiación luminosa para pasar señales de un circuito a otro sin conexión eléctrica. Todos estos elementos se encuentran dentro de un encapsulado que por lo general es del tipo DIP como el de la gura 4.9.

Fundamentalmente este dispositivo está formado por una fuente emisora de luz, y un foto-sensor de silicio, que se adapta a la sensibilidad espectral del emisor luminoso. La gran ventaja de un optoacoplador reside en el aislamiento eléctrico que puede establecerse entre los circuitos de entrada y salida.

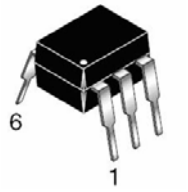


Figura 4.9: Encapsulado Dip

El circuito integrado usado para aislar al controlador del motor a paso es el 4N28 [2] mostrado en la figura 4.10.

[2] ver hoja de datos en apéndice D

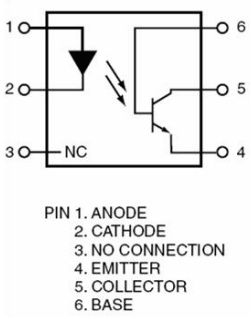


Figura 4.10: OptoAislador 4N28

4.3. Max232

El Max232 es un receptor-emisor dual que incluye un generado de voltaje capacitivo para suministrar a la interfase EIA-232. Cada receptor convierte las entradas de 5 V de los niveles TTL/CMOS a los niveles requeridos por el EIA232. Esos receptores tienen un umbral típico de 1.3 V y una histéresis típica de 0.5 V, y pueden aceptar entradas de ± 30 V y viceversa.

El encapsulado del Max 232 es tipo DIP, figura 4.11, su conexión típica se muestra en la figura 4.12.

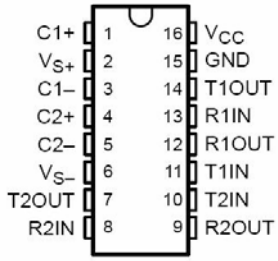


Figura 4.11: Max232

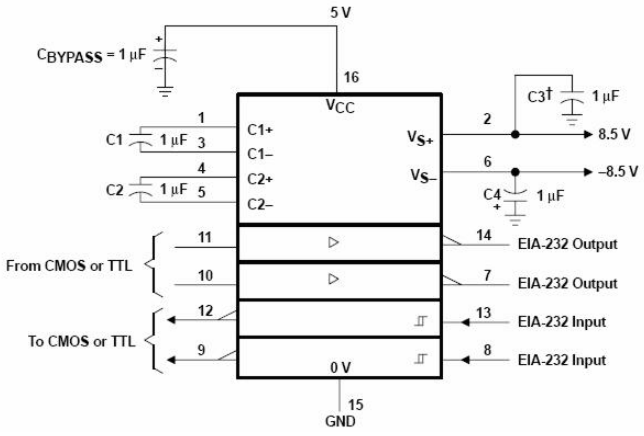


Figura 4.12: Conexión Típica del Max232

4.4. Etapa de control

Está diseñado para interpretar los valores que son enviados desde el AHIETI-SC (AHIETI-Software de Control), que se explicara a detalle en el siguiente capítulo.

4.4.1. Diseño de la etapa de control.

Para poder manejar un motor a paso con la computadora se requiere de cuatros pines del puerto paralelo y de una etapa de potencia como la descrita anteriormente, pero cuando se necesita manipular más motores los pines necesarios se multiplican por el número de motores que se quiera manejar, esto limita el trabajo del puerto paralelo a solo dos motores por puerto usando los ocho pines de entrada-salida.

Los microcontroladores tienen más pines de entrada-salida por lo que tienen la posibilidad de manejar más motores y esto aunado a la comunicación mediante el USART (Universal Synchronous Asynchronous Receiver Transmitter, transmisor receptor sincrónico asincrónico universal) de los microcontroladores podemos manejar un mayor número de motores con una sola PC, inclusive con el puerto serie, de solo dos pines, para comunicar la PC con la etapa de potencia de los motores.

Al estudiar esta posibilidad se diseño una primera versión de la etapa de potencia con la cual se pueden controlar los cuatro motores con el puerto serie y un solo microcontrolador.

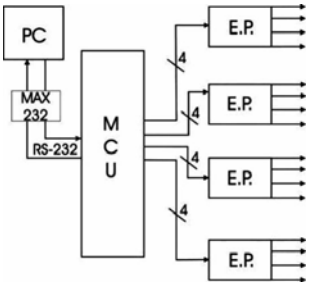


Figura 4.13: Primer Diseño de la Etapa de Control

En este primer diseño el puerto serie de la PC se conecta al USART del microcontrolador, esto posibilita usar dos pines para la comunicación entre MCU y la computadora, y dieciséis pines para manejar los motores, esto resulta en veinte pines.

Una desventaja que presenta esta configuración es que el código se vuelve muy grande y el espacio en memoria es reducido, además se toman pines del microcontrolador que poseen funciones que podrían ser aprovechadas de otra manera. Por esta razón se consideró un segundo diseño que nos permite mejores prestaciones.

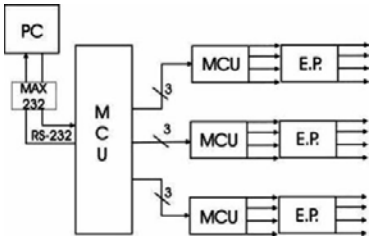


Figura 4.14: Segundo Diseño de la Etapa de Control

Con este diseño se obtiene una mejora tanto en desempeño como en escalabilidad ya que se descentraliza el trabajo en el microcontrolador principal, delegando funciones a los microcontroladores secundarios, además de un ahorro importante de pines, solo dos de comunicación y nueve de control, un total de once pines, se dejan disponibles otros pines de funciones especiales de los microcontroladores, permitiendo el censado y una toma de decisión independiente del procesador principal.

4.4.2. Microcontroladores.

Actualmente en el mercado se encuentran gran cantidad de microprocesadores y microcontroladores disponibles, dependiendo las tareas que se quieran realizar, se debe elegir el adecuado. Los microprocesadores di eren de los microcontroladores, ya que los primeros requieren de periféricos que se encuentran autónomos y los segundos poseen los periféricos

en el mismo encapsulado (sistema Embebido), y se pueden configurar desde el código. El espacio es otra de las diferencias, mientras el microcontrolador ocupa solo su espacio físico dentro del circuito, el microprocesador necesita espacio para si mismo, para los periféricos y para los componentes adicionales que así se necesiten.

El precio del dispositivo dependerá de la cantidad de pines, el espacio en memoria, los periféricos internos y la velocidad de procesamiento que pueda soportar. Por ejemplo un microcontrolador conocido como de gama baja con 10 pines, poca memoria y un par de periféricos, puede costar entre 7 u 8 dolares, mientras un microcontrolador de la gama alta para procesamiento digital con varios pines y periféricos especializados puede costar entre 40 y 50 dolares.

Los microcontroladores que se utilizan en este proyecto son el PIC 16F877A y 16F876A de la empresa Microchip, estos poseen un CPU de 8 bits, memoria de programa de 14.3 Kbytes, SRAM de datos de 368 Bytes y una EEPROM de 256 Bytes, 22 y 33 pines de entrada salida programables respectivamente, 5 y 8 canales de conversión analógica a digital, dos moduladores de ancho de pulso, comunicación SPI, I^2C, USART, 2 timer de 8 bits, uno de 16 bits y dos comparadores.

El microcontrolador principal

Este será el encargado de recibir la información que envía la computadora, interpretarla, activar los controles de los microcontroladores secundarios y enviar la posición actual de regreso a la computadora. figura 4.15

El microcontrolador se posiciona en 0,0 y espera para la instrucción, la computadora envía la primera coordenada de intercepción, el microcontrolador recibe la información, baja el mini-drill y avanza hasta la posición indicada, envía la posición actual y sube el mini-drill, espera la siguiente instrucción, la recibe y avanza hasta el final de la intercepción, envía la posición y baja el mini-drill, repite la operación hasta recorrer la superficie total de la placa en X, después activa un paso en Y repite la operación anterior, así hasta cubrir la superficie de la placa en Y.

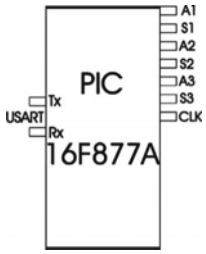


Figura 4.15: Microcontrolador Principal

Para la comunicación entre el microcontrolador y la computadora se usa el módulo de transmisión USART del microcontrolador (pines 25 y 26) para ello hay que configurar los registros internos del microcontrolador. Primero se configura el generador de baud rate, se activa la transmisión asíncrona y full duplex, se elije enviar tramas de 8 bits a una velocidad de 9.6 Kbytes/segundo. Se cargan los valores en el registro TXREG y se envían, para recibir los datos se lee el registro RCREG cuando se activa el registro RCIF, ahora los datos fueron enviados y recibidos.

Además de la comunicación con la computadora el microcontrolador principal tiene la función de controlar los microcontroladores secundarios, esto lo logra con los tres pines A, que activan o desactivan el funcionamiento de los motores, los 3 pines S indican el sentido de giro y el pin CLK sincroniza los pasos en los motores.

Es importante mencionar que se dejaron libres los demás pines del microcontrolador, que son convertidores analógicos-digitales o puertos de entrada y salida programables, esto se debe a que se permite la modificación o mejora del prototipo con base a técnicas de control y censado.

Microcontroladores secundarios

Se encargan de generar las secuencias de movimiento de los motores a paso dependiendo de las órdenes del microcontrolador principal.

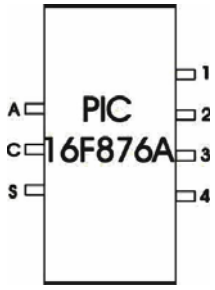


Figura 4.16: Microcontrolador Secundario

Hay tres pines de entrada y cuatro de salida, el pin A es la activación de la secuencia, si esta encendido inicia la secuencia de pasos y la detiene cuando se desactiva, el pin C es el reloj de sincronización, indica cuando el motor debe cambiar de paso mediante un tren de pulsos, el pin S indica si el motor sigue su secuencia para girar en sentido horario o antihorario, los pines de salida están numerados para enviar una secuencia de activación de dos pines por paso, como se explicó en la subsección 4.1.1 , la secuencia para manejar un motor a paso (secuencia normal) ya sea en sentido directo o en sentido inverso.

En el caso de los microcontroladores secundarios también se dejan los pines de conversión analógica digital y varios pines de entrada salida programables para censar los motores

de manera independiente y tomar decisiones descentralizadas que de lo contrario podrían complicar el código del microcontrolador principal.

Interacción entre los microcontroladores

Los microcontroladores secundarios se conectan al principal mediante un bus de tres terminales, aún manejando cuatro motores solo se usan tres controladores ya que dos de los motores comparten la misma secuencia de pasos solo que invertida, por lo cual las conexiones quedan de la siguiente manera.

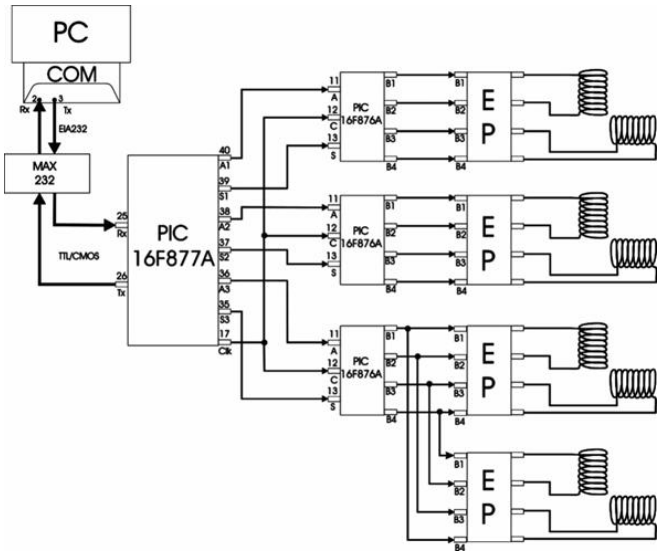


Figura 4.17: Conexión entre microcontroladores

Así queda la conexión entre los diferentes componentes del sistema de control y etapa de potencia, los números indican el número de pin en el componente, y las letras son la nomenclatura que se les asigna a dichos pines, los bloques de la etapa de potencia ya fueron explicados en la sección 4.2.

4.5. Diseño PCB AHIETI v 2.1.1

El diseño del PCB para la etapa de potencia y aislamiento fue desarrollado con el programa Traxmaker y diseñado de tal manera que se pueda implementar en una placa fenólica de 15x10 como se muestra en la figura 4.18.

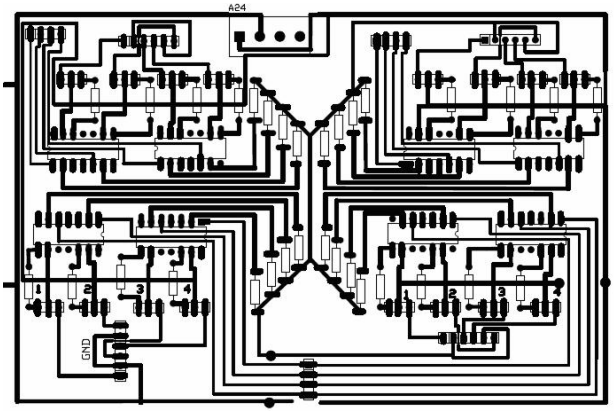


Figura 4.18: Circuito PCB de potencia

La figura 4.19 muestra el diseño de la etapa de control también desarrollado en Traxmaker y diseñada para imprimirse en una placa de 15x10.

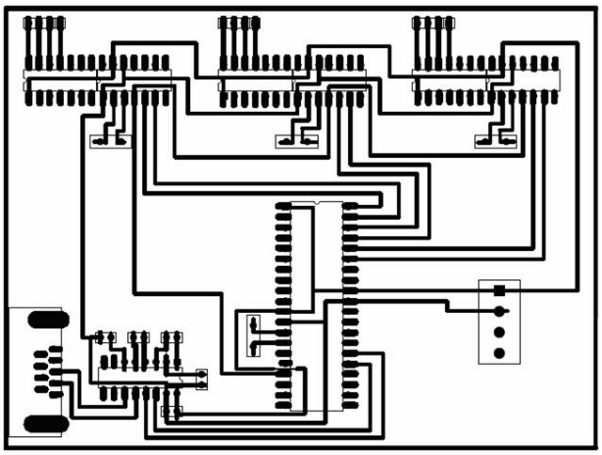


Figura 4.19: Circuito PCB de control

Capítulo 5

AHIETI-SC

El AHIETI-SC es la interfase gráfica que interactúa directamente entre el usuario u operador y el robot, se encarga de interpretar los archivos PCB creados previamente, convertirlos en archivos de soporte, presentar una vista previa, simular la trayectoria que el AHIETI v 2.1.1 deberá seguir para elaborar el circuito PCB y ejecutar dicha trayectoria.

Este proyecto a sido desarrollado en su totalidad con la herramienta de desarrollo Microsoft Visual Studio .NET 2003 con el lenguaje de programación Visual Basic para tener compatibilidad con Microsoft Windows. Los archivos PCB son generados en un programa externo (TraxMaker, que se explicará más adelante), esto es por la mayor cantidad de herramientas y librerías previamente desarrolladas, además sería innecesario desarrollar una herramienta que ya se encuentra disponible.

La posición se determina contando los pasos de los motores a paso ya que cada paso como ya se explicó anteriormente tiene un avance constante, por esto, es posible determinar la posición de los deslizadores simplemente con los pasos del mismo motor.

La trayectoria de los deslizadores es determinada por un algoritmo que permite prever en que posición el AHIETI v 2.1.1 debe descobrear y en que posición solo debe avanzar.

5.1. Lenguaje de programación.

En la actualidad se pueden encontrar infinidad de herramientas que permiten desarrollar diferentes tipos de aplicaciones, en diversos tipos de sistemas operativos, con grados de dificultad muy disímiles y con opciones muy avanzadas, pero se deben conocer estas herramientas muy bien para poder tomar la decisión de cual es la mas adecuada, para el desarrollo de la aplicación que se quiere obtener.

Para conseguir el software que se requiere para esta aplicación se evaluó el conocimiento previo con la herramienta, la facilidad de manejo, además de las capacidades que permitirán manejar los dispositivos que se requieren, la compatibilidad con los sistemas que se

tienen y la dificultad con la que se puede tener acceso a la información acerca de la misma herramienta.

5.1.1. Ensamblador

En un principio la programación del hardware era muy complicada ya que como es lógico cuando el hardware fue desarrollado no existían lenguajes para realizar los programas que lo manejarán, por esta razón los programadores debían entender como funcionaba el hardware y tratar de comunicarse con el mediante un lenguaje que entendiera el hardware, a este lenguaje se le conoció como lenguaje máquina, pero este era demasiado complicado y permitía gran cantidad de errores que eran difíciles de localizar y aún más de corregir.

Por la misma razón fue creado un lenguaje de mayor nivel el cual se conoce como lenguaje Ensamblador, pero aún así este lenguaje depende mucho de las características del hardware por lo cual el programador debe tener aún un gran conocimiento del mismo para saber como se comportaba con las determinadas instrucciones. El ensamblador es considerado como lenguaje de bajo nivel por que aunque usa sentencias parecidas para la mayoría del hardware, estas pueden cambiar dependiendo del procesador en el que se están aplicando.

Aún con la dificultad del lenguaje y su poca compatibilidad de un mismo código para diverso hardware se pueden encontrar ventajas del ensamblador sobre otros lenguajes de alto nivel como son:

Velocidad

A pesar de que las computadoras modernas son cada vez más veloces cabe mencionar que los códigos escritos en ensamblador son más fáciles de interpretar por la computadora, en otros lenguajes el interprete debe descifrar primero el código que escribimos para poder traducirlo a lenguaje máquina, pero en ensamblador esto no lleva demasiado tiempo ya que es lo más parecido a lo que la máquina entiende, si usamos un programa interprete este tendrá que traducir el código cada vez que se ejecuta, pero esto puede ser disminuido si se tiene un programa compilador ya que el código se traduce una sola vez y se obtiene un código en lenguaje máquina que es mucho más rápido de interpretar.

Si por ejemplo, se tiene el mismo proceso y se escribe en ensamblador, un interprete y un compilador en el compilador tardará 2 veces más que en ensamblador y el interprete dieciséis veces mas, esto es suponiendo que los algoritmos están bien hechos en los tres casos.

Eficiencia de tamaño.

Al escribir un ensamblado se ocupan normalmente las líneas de código estrictamente necesarias para realizar una determinada tarea.

Los lenguajes interpretes y aún los compiladores generan líneas adicionales para poder traducir un código, esto genera gran cantidad de código adicional que nos da como resultado

un programa de mayor tamaño (hablando de espacio lógico), esto es importante cuando queremos que nuestra aplicación use una pequeña cantidad de memoria y mas aún cuando queremos que el programa sea más rápido.

Flexibilidad

En muchos programas por cuestión de permisos no es posible explotar todas las capacidades del hardware pero en el lenguaje ensamblador no tenemos limitaciones siempre y cuando se conozca el dispositivo que queremos manejar, esto nos permite realizar tareas que de otra forma seria muy difícil o inclusive imposible de hacer con los interpretes o los compiladores.

Por otra parte por ser un lenguaje muy primitivo nos presenta varias desventajas por ejemplo.

Tiempo de programación

Al ser un lenguaje de bajo nivel, el ensamblador requiere de más instrucciones para realizar una misma tarea en otros lenguajes basta con una sola, por otra parte presenta una mayor posibilidad de cometer errores de lógica que se refleja más en la ejecución y que son los más difíciles de identificar.

Peligro de afectar recursos inesperadamente

Así como tenemos la ventaja de manejar los recursos de la máquina a nuestra disposición, también tenemos la posibilidad de equivocarnos y alterar las configuraciones básicas de los recursos, por ejemplo podemos simplemente bloquear o trabar la máquina por usar una instrucción no valida, pero también podemos hacer daños más graves como sobrescribir el BIOS de la computadora, o en los sectores primarios del disco duro dejando temporalmente inservible la computadora donde estamos trabajando.

Falta de portabilidad

La capacidad del ensamblador de manejar los recursos hasta el fondo se debe a que hay un ensamblador para cada plataforma, por ejemplo, existe un ensamblador solo para procesadores x86, uno para microcontroladores , otro más para microprocesadores SPARC, POWER PC, etc.

Aunque es cierto que pocos lenguajes permite llevar un código de una plataforma a otra, en ensamblador es prácticamente imposible, ya que en los demás lenguajes solo falta seguir los estándares de los lenguajes o de la programación estructurada y hacer unos cuantos cambios correspondientes a algunos recursos, en ensamblador se deberá escribir nuevamente todo el código si se quiere hacer el cambio.

Aunque este lenguaje nos puede permitir inclusive realizar una aplicación que no requiera de un sistema operativo, el tiempo de desarrollo es el más largo de todos los lenguajes de programación, las instrucciones son limitadas a pesar de que se pueden encontrar libre- rías para el manejo de dispositivos, la interfase gráfica es en extremo laboriosa y se debe desarrollar un conocimiento más especializado para el manejo de este lenguaje.[11]

5.1.2. Lenguaje C

El lenguaje C fue desarrollado en los Bell Telephone Laboratories, por Dennis M Ritchie a principios de los años setenta, se considera que su nacimiento fue en el sistema operativo UNIX, pero se pueden encontrar versiones para cualquier sistema operativo de la actualidad. Lo que se buscaba con el era obtener una herramienta eficiente, que pudiera explotar las capacidades de una computadora al máximo y que además de que el código se pudiera compilar en diferentes computadoras. Finalmente este lenguaje obtuvo una gran aceptación al reescribirse el sistema operativo UNIX completo en este lenguaje.

Al lenguaje C se le critica por que los códigos escritos son poco entendibles por terceros y es muy propenso a errores, pero esto no lo demerita ya que es una herramienta muy poderosa.

Este lenguaje de programación no fue tomado ya que el periodo de desarrollo de la aplicación es demasiado largo, por la dificultad de manejar los puertos de comunicación de la computadora, el reducido número de instrucciones que posee y que para poder ampliarlas se requiere de librerías de terceros que en muchas ocasiones son muy difíciles de entender y que tienden a fallar con aplicaciones muy especificas.

Por la parte gráfica este lenguaje presenta una gran dificultad y una inversión considerable de tiempo, la información es demasiado técnica y en algunos temas que nos interesan es inexistente o presenta un costo por el acceso a la misma.

5.1.3. Lenguaje C++

Este lenguaje de programación fue desarrollado por Bjarne Stroustrup a nales de los años 70 en la empresa AT&T, ya que la programación orientada a objetos comenzaba a obtener fuerza, este lenguaje es una mejora al C para permitirle trabajar con la programación orientada a objetos.

Actualmente es la versión de C más utilizada por la mayoría de los ingenieros del software alrededor del mundo, se pueden encontrar diversos compiladores de una amplia lista de desarrolladores como Microsoft, Borland, entre otros.

Presenta mejoras en las librerías con un código más amigable y prestaciones con los puertos (gracias a librerías externas), las instrucciones se extienden considerablemente, pero aún es muy complicado desarrollar una interfase gráfica y en las versiones más recientes de este lenguaje, se presenta la oportunidad para trabajar con ventanas y controles, pero el enlace de estos controles a eventos establecidos deben programarse manualmente, esto hace

que la programación sea más lenta y que se generen errores no tanto sintácticos si no de lógica por parte del programador que al tratar de corregirlos pueden conllevar a tiempos de depuración muy grandes.[14]

5.1.4. Delphi

Delphi es un lenguaje de programación visual basado en el lenguaje PASCAL desarrollado por Niklaus With en 1971, para facilitar la enseñanza de la programación estructurada.

Con esta herramienta se pueden desarrollar aplicaciones gráficas con diseñadores de ventanas, las instrucciones son muchas y permite desarrollar aplicaciones muy elaboradas.

Desafortunadamente como ya se había mencionado esta herramienta esta basada en el lenguaje de programación PASCAL, el cual es un lenguaje muy viejo, y las nuevas herramientas están basadas a programadores que conocían previamente este lenguaje, es muy difícil aprender la sintaxis para nuevos programadores. Esto hace de Delphi un lenguaje apto para aquellas personas que ya trabajaron con versiones anteriores de PASCAL.

5.1.5. Visual C++

Esta versión del C++ por parte de Microsoft proporciona un entorno de desarrollo dinámico para crear aplicaciones en Windows. Proporciona a los programadores un lenguaje orientado a objetos.

Permite manejar casi la totalidad de los recursos del sistema. Pero el aprendizaje es muy complicado para personas que no tienen grandes bases o al menos las suficientes de la programación orientada a objetos, por lo cual a ingenieros que no obtuvieron las bases necesarias les es difícil entender este lenguaje.[15]

5.1.6. Java

Java fue diseñado en 1990 por James Gosling, de Sun Microsystems, como software para dispositivos electrónicos de consumo. Curiosamente, todo este lenguaje fue diseñado antes de que diese comienzo la era World Wide Web, puesto que fue pensado para dispositivos electrónicos como calculadoras, microondas y televisión interactiva.

Entre las características que hacen de java un lenguaje puramente orientado a objetos es la eliminación de punteros que en muchos de los casos provocaba errores de colisión de memoria, que son muy difíciles de detectar, las variables globales también fueron eliminadas para evitar que funciones pudieran producir efectos laterales o fallos importantes, las sentencias de escape como el goto son eliminadas, la conversión de tipo es más segura en java ya que se puede hacer una comprobación en tiempo de ejecución de la compatibilidad de tipos de variables.

Aunque es el lenguaje más universal que podemos obtener, es muy complicado y requiere de un gran tiempo de aprendizaje para aplicaciones más especializadas.

5.1.7. Basic

Lenguaje desarrollado en la década de los 80 por John George Kemeny y Thomas Eugene Kurtz, fue originalmente ideado como una herramienta de enseñanza, por su acrónimo en ingles de Beginners All-purpose Symbolic Instruction Code algo así como código de instrucciones simbólicas de propósito general para principiantes.

Fue diseñado para permitir a los estudiantes escribir programas usando terminales de computadora de tiempo compartido, respetando ocho principios básicos de diseño. Ser fácil de usar por los principiantes, ser un lenguaje de propósito general, permitir que los expertos puedan añadir características avanzadas, ser interactivo, proveer mensajes de errores claros y amigables, responder a los programas pequeños, no requerir de conocimientos de hardware, proteger al usuario del sistema operativo.

Tal vez el mas sencillo de los lenguajes de programación por su gran parecido al Idioma Ingles, la información acerca de este es muy amplio y su tiempo de aprendizaje es muy corto, sin embargo la programación de interfases gráficas es muy laboriosa y las posibilidades de manejar los recursos son muy limitadas.

5.1.8. Gambas

Gambas es un lenguaje de programación libre similar a BASIC. Es similar al producto de Microsoft Visual Basic y se distribuye con licencia GNU GPL. También desde otro punto de vista, está muy inspirado también por Java. Gambas nace como respuesta a la necesidad de tener un entorno de desarrollo rápido de aplicaciones o RAD por sus siglas en ingles, además cumple la necesidad de muchos programadores del lenguaje de Microsoft de un lenguaje conocido en plataforma abierta.

Permite crear formularios, botones de comandos, cuadros de texto o enlazar bases de datos como MySQL o PostgreSQL.

Gracias a estar basado en BASIC es igualmente sencillo y la información es muy similar a la del BASIC normal, el compilador es gratuito y hay grandes foros de donde se pueden obtener fragmentos de código muy útiles, la programación gráfica es posible sin invertir gran tiempo en el desarrollo en la misma, las posibilidades de manejar los recursos es muy amplia pero complicada.

La compatibilidad de este lenguaje esta restringidas a el sistema operativo LINUX por lo cual aún cuando se desarrollara el sistema en este lenguaje no se podría hacer muy sencillo ya que se requiere de conocimientos básicos en UNIX o en LINUX y más avanzados en computación por parte de los usuarios, ya que hay que configurar los servicios necesarios y saber manejar los servidores de los puertos que queremos ocupar.

5.1.9. Visual Basic

Lenguaje de programación desarrollado por Microsoft en 1991 para el desarrollo de aplicaciones visuales en Windows.

Inicialmente, Visual Basic fue pensado para ser un producto muy táctico. Microsoft tenía varias iniciativas en el desarrollo que lideraba Visual Basic 1.0, todas fueron pensadas para convertirse en las herramientas de programación a largo plazo, estratégicas, gráficas y orientadas a objetos. Como siempre ocurre con los productos en su versión 1.0, el equipo de Visual Basic 1.0 fue forzado a cortar características de su larga lista de ideas para entregar realmente el producto al mercado. Consecuentemente, la primera versión incluyó poco más que la tecnología Embedded Basic que había sido desarrollada originalmente en Microsoft QuickBasic 4.0 (el código "p" y compilador de Microsoft) y una herramienta compiladora de diseño simple originalmente diseñada para Windows 3.0 pero que nunca fue utilizada para tal n.

A través de los años, Visual Basic ha evolucionado del juguete de un aficionado a una herramienta imprescindible que continúa cambiando el mundo. Ha revolucionado la manera que se trabaja con la información, la manera de comunicarse, y la forma en la que se construyen aplicaciones.

Con todos los cambios en la configuración de la tecnología y de la aplicación en los últimos 10 años, la comunidad de Visual Basic ha continuado innovando como líderes en el desarrollo y la producción de software. Mientras, es interesante mirar al pasado y observar los éxitos alcanzados por este producto, es importante remarcar que Visual Basic está sólo en una fase bastante inicial.

Como su antecesor el BASIC, es muy sencillo de aprender, la comprensión del mismo es igualmente sencilla, el tiempo de aprendizaje es muy corto, la información del mismo es muy extensa en libros, paginas WEB, Foros de Internet, entre otros.

El manejo de los controles es intuitivo, el enlace de los controles con los eventos es muy sencillo ya que el mismo editor se encarga del manejo de los mismos, con esto el programador no se preocupa de programarlos y no puede cometer errores de lógica con los eventos.

El manejo de los recursos se vuelve muy sencilla gracias a que estos están incluidos en los controles, son muy completos, aún así esta limitado en muchos aspectos y aún más en sistemas Windows más modernos.

5.1.10. Visual Studio .NET

Visual Studio .NET es un conjunto completo de herramientas de desarrollo para la construcción de aplicaciones Web ASP, servicios Web XML, aplicaciones para escritorio y aplicaciones móviles. Visual Basic .NET, Visual C++ .NET, Visual C# .NET y Visual J# .NET utilizan el mismo entorno de desarrollo integrado (IDE), que les permite compartir herramientas y facilita la creación de soluciones en varios lenguajes. Asimismo, dichos lenguajes aprovechan las funciones de .NET Framework, que ofrece acceso a tecnologías clave para simpli car el desarrollo de aplicaciones Web ASP y servicios Web XML.

NET Framework es un entorno multilenguaje que permite generar, implantar y ejecutar Servicios Web y aplicaciones XML.

El uso de Frameworks permite poder manejar varios lenguajes de programación en un solo sistema de desarrollo y que el programador pueda elegir el lenguaje que más conozca sin tener que sacri car funcionalidad ya que el Frameworks funciona como interprete entre el lenguaje y el compilador, por lo cual la programación es muy parecida con cualquier lenguaje que se quiera usar.

El lenguaje que se uso para el desarrollo del AHIETI-SC es el Visual Basic incluido en la herramienta de desarrollo Visual Studio .NET 2003, esta decisión esta basada a que ya se tienen conocimientos previos del visual Basic, la migración entre versiones no es muy complicada, la información que se tienen en la misma ayuda del programa es muy completa, nos permite manejar los recursos necesarios, la interfase grá ca se desarrolla con un diseñador de ventanas, los eventos se enlazan muy fácilmente con controles y permite la programación orientada a objetos, además, la compatibilidad de las aplicaciones desarrolladas con los sistemas Windows existentes es muy buena.[16]

5.2. Estudio de los archivos PCB

Como mencionamos anteriormente estos archivos son creados por otro programa llamado TraxMaker el cual es independiente al sistema desarrollado en este trabajo.

5.2.1. TraxMaker

Aplicación grá ca desarrollada por la empresa PROTEL dentro de la Suite de simulación y diseño de circuitos electrónicos CircuitMaker 2000 (La presentación se muestra en la figura 5.1), especialmente para el desarrollo de prototipos PCB, es muy sencilla de usar, básicamente el usuario puede arrastra los componentes hasta el área de trabajo, dibujar las pistas como desee y usar librerías de componentes que cumplen con los estándares más comunes de la industria de componentes electrónicos y semiconductores.

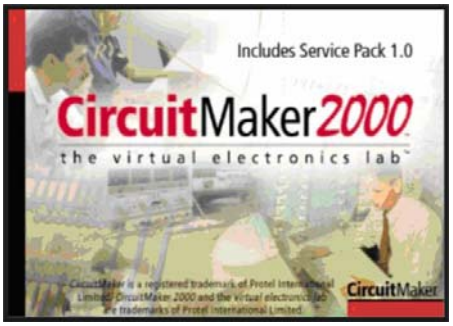


Figura 5.1: Ventana de presentación de CircuitMaker

El programa muestra la ventana de la figura 5.2, como se puede ver, contiene un espacio de trabajo, una barra de menú en donde se muestran los comandos comunes a las ventanas de Windows como archivo, edición, ver y opciones; además de otras que son de funciones exclusivas del programa, una barra sencilla de herramientas en donde encontramos los comandos más comunes para el desarrollo de los circuitos impresos, y una barra de estado que nos muestra la capa donde se trabaja y la posición del cursor dentro del área de trabajo.

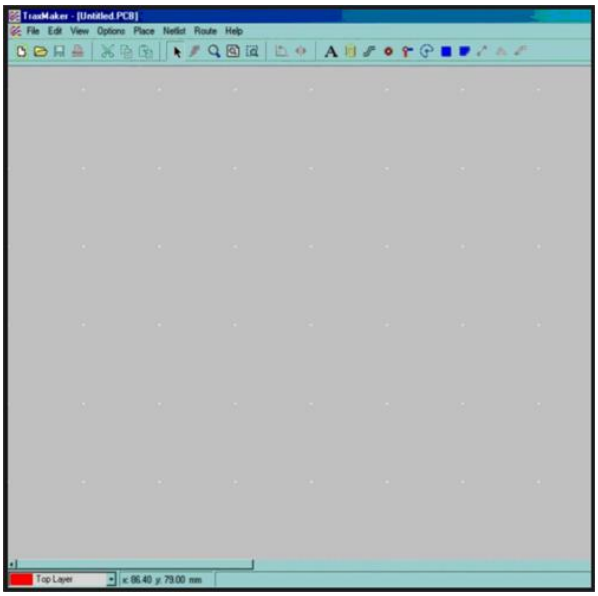


Figura 5.2: Ventana principal TraxMaker

Al terminar el diseño de un circuito impreso podemos guardarlo en un archivo propio del programa, este archivo tiene una extensión PCB y guarda una lista de coordenadas y propiedades precedida de una cabecera que identifica el tipo de elemento que se encuentra en el área de trabajo.

5.2.2. Archivos PCB

Los archivos generados por el programa TraxMaker tienen la extensión PCB (Printable Circuit Board, Placa de Circuito Impreso), a continuación se explicara la estructura de un archivo PCB.

Cabe mencionar que el programa esta diseñado para trabajar en el sistema ingles, por lo cual las medidas esta dadas en MILS (milesimas de pulgadas), no en milímetros, esto puede causar confusiones con personas que están acostumbradas al uso del sistema internacional.

Cabecera y Fin del archivo

Al inicio de un archivo PCB se encuentra una cabecera que lo identifica como PCB está cabecera contenida en una sola línea dice que es un archivo, que es PCB y el número de capas del archivo, generalmente son cuatro, dos son de las capas de cobre una inferior y una superior, y dos más de los gráficos que identifican los componentes en la placa fenólica igual que la anterior una inferior y una superior. El formato es el siguiente:

FILE PCB 4

En el final del archivo se encuentra una etiqueta que indica que hasta ese punto termina el archivo, sin esta etiqueta la lectura del archivo sigue de manera indeterminada y esto puede ocasionar errores. La etiqueta es:

ENDPCB

Cuerpo del archivo

Como ya habíamos mencionado dentro del cuerpo del archivo se encuentran cabeceras que indican que tipo de elemento se coloca en el área de trabajo, estos elementos pueden ser:

- **Tracks (lineas)**
- **Arcs (Arcos)**
- **Fill (Figuras solidas ó rellenas)**
- **Pads (Perforaciones)**
- **Text (Texto)**

Los elementos Básicos

Arc

Son líneas curvas, que de hecho son secciones de círculos que son seccionadas en cuatro partes de 90 grados cada una. La etiqueta que se encuentra en el archivo e identifica este elemento es FA seguido de una línea de números separados por espacios cada número representa una coordenada u otro elemento característica de un arco necesaria para poder grafi carla. El formato es el siguiente:

484 588 308 15 10 1

Las primeras dos cifras son las coordenadas del punto de origen en (X,Y) de un circulo

La tercera cifra es el radio del círculo medido a partir del punto de origen que se obtiene con las primeras dos.

La cuarta es el cuadrante en el cual esta dibujada la curva para el primer cuadrante esta denominada con el número uno, el segundo con el número dos, el tercero por el número cuatro y el cuarto por el número ocho; si se quiere que el arco se dibuje en más de un cuadrante estos números se suman, por ejemplo si se quiere que este dibujado en el primero y segundo cuadrante el número que se obtiene es tres.

La quinta cifra es el ancho de la línea igualmente representado. La sexta y última cifra es la capa donde se dibujara el elemento.

Tracks

Son las líneas rectas que representan las pistas que unen los pines de los componentes, la etiqueta que caracteriza al elemento es FT seguido de una línea similar a la del elemento anterior. El formato es el siguiente:

820 128 1000 500 12 1 1

Las primeras dos cifras representan las coordenadas en (X,Y) del punto de origen de la recta

Las siguientes dos cifras representan las coordenadas en (X,Y) del punto de destino de la recta.

La quinta cifra es el ancho de la línea tomada La sexta es la cifra que indica la capa donde deberá ser colocada. La séptima y última cifra no tiene importancia, solo es una cifra de referencia para el
programa, pero si se retira esta cifra la línea no es dibujada, este elemento es el único que tiene esta última cifra.

Fill

Son rectángulos sólidos usados generalmente par colocar indicaciones o para facilitar la conexión entre componentes, su etiqueta es FF y la línea que caracteriza su forma es la siguiente:

380 260 936 620 1

Las primeras dos cifras son el punto de inicio de la esquina superior izquierda del rectángulo.

Las siguientes dos son el punto final de la esquina inferior derecha. La quinta cifra es la capa donde esta colocado.

Pads

Son los ori cios o perforaciones que se tienen que hacer en la placa fenólica para insertar los componentes, por lo regular los Pads tienen cobre alrededor de los mismos para poder soldarlos a las placas asegurando una correcta conexión. La etiqueta característica es FP y la descripciones la siguiente.

144 140 125 125 1 50 1 13 0

A diferencia de los elementos anteriores este posee dos líneas con sus características, las primeras dos cifras indican la posición del centro del Pad en el área de trabajo.

Las siguientes dos son el tamaño total del Pad contando el orificio así como el cobre que se encuentra alrededor, la primera es el ancho y el segundo es el alto.

La quinta es la forma que tendrá el Pad, este puede ser redondo, cuadrado, hexagonal, rectangular con bordes redondos y sin cobre alrededor.

La sexta es el tamaño de la broca que se debe usar para perforar la placa. La séptima es el plano de conexión este puede ir sin plano de conexión, liberado del plano de energía, directo al plano de energía, liberado del plano de tierra o directo al plano
de tierra. La ultima cifra de la primera línea es la capa en donde se encontrara el Pad por lo general el pad se dibuja en ambos planos aunque esto depende del diseño. En la segunda línea solo se tiene una cifra la cual representa cual es el número del Pad dentro del circuito.

Componente

Hay que aclarar que se tienen librerías con componentes previamente elaborados estos están construidos por los elementos que se mencionan anteriormente, con la diferencia que su etiqueta inicia con C en lugar de F y se encuentran agrupados en una estructura como la que se muestra a continuación

COMP
IC2
DIP6
PIC
77 370 60 0 10 7
77 440 60 0 10 7
108 292 1 1 2
CA
258 342 25 12 8 7
ENDCOMP La primera línea indica que las siguientes pertenecen a un componente, la segunda línea
es el nombre del componente, la tercera es el tipo de encapsulado, la cuarta es el valor que tiene el componente, las siguientes 2 líneas indican la posición del nombre y del valor en donde se van a colocar dentro de la placa.

Las siguientes líneas dependerán de la cantidad de elementos que conformen al componente en el PCB, terminando en una etiqueta que indica que es el n del componente.

Otras consideraciones

Hay que tener en cuenta que en la elaboración de los circuitos hay que tomar en cuenta, además de las características de los componentes también as características de la placa fenólica. Las siguientes son las características más importantes a considerar.

La placa fenólica tiene dos caras es decir, que tiene dos lado por los cuales se pueden trabajar uno superior y otro inferior, para el diseño es importante decidir en que cara se van a colocar los componentes, si el componente es de montaje superficial el componente y las conexiones estarán en la misma cara si el montaje es normal las conexiones pueden ir en la cara opuesta o en ambas caras.

La placa también puede tener varias capas en una sola cara, por ejemplo en las placas madre de las computadoras hay dos o más capas de cobre separadas pos una capa de esmalte, también hay que considerar que se pueden ocupar capas superiores o inferiores en una sola cara, algunos programas para el diseño de PCB manejan múltiples capas en una o en las dos caras.

El AHIETI v 2.1.1 maneja únicamente una capa por cara ya que para la elaboración de placas con mayor numero de capas se usan técnicas más complejas para fabricar las mismas.

5.3. Propuesta del intérprete

El AHIETI v 2.1.1 ha sido diseñado para seguir trayectorias lineales y decidir en base a los archivos PCB generados por el TraxMaker en donde se debe descobrear y en donde se debe mantener el cobre para la interconexión de los componentes del circuito electrónico, pero estos archivos contienen parámetros que solo son para el uso del programa, por ejemplo el color de las pistas, por eso se deben interpretar para utilizar los parámetros necesarios.

Los parámetros que se pueden tomar en cuenta son los de posición, tamaño, forma, cara y capa. Hay que tomar en cuenta que los valores de posición y tamaño están dados en mils y para poder trabajar con ellos más fácilmente hay que convertirlos en milímetros; el valor que indica la forma es un número entero y por ultimo la cara y la capa esta indicado por números enteros que representan colores, por ejemplo uno es el color rojo que indica que el elemento esta en la cara inferior en la primera capa.

Para interpretar el archivo se lee línea por línea, al leer la línea se debe inspeccionar si es una etiqueta o es una línea de parámetros, como en la mayoría de las veces una etiqueta precede a una línea de parámetros; al leer una etiqueta se deduce que la siguiente línea debe ser de parámetros.

Cuando se lee una línea de parámetros se debe elegir que parámetros son útiles y cuales son innecesarios para ello se debe separa la línea en todas sus componentes valorar las componentes y convertir las que deban ser convertidas ya que se encuentran mezclados los parámetros de posición y tamaño con otros de los cuales se debe tomar el valor así como es mostrado.

Para no hacer un intérprete tan complicado que haga todo el trabajo para cada función se hace un interprete que también traduzca y genere archivos de soporte que sean fácilmente entendibles por cada una de las funciones del programa.

Para la primera de las funciones que es la de graficar el archivo se hace un archivo que nos permita usar los parámetros de posición, tamaño capa y cara, después de esto que convierta

los valores de posición y tamaño de mils a la unidad que maneja la computadora para representar gráficos es decir, a píxeles para esto primero se convierte el valor a milímetros y posteriormente a un equivalente en píxeles. Ya con los valores que se tienen se crea un archivo con una extensión APP (AHIETI Píxel PCB).

La segunda función es la de generación de la placa de circuito impreso, para esta necesitamos en primera las posiciones de los elementos y el ancho d las líneas, por otra parte se requieren de las posiciones que s van a perforar, primero tomamos los valores que están en mils y los convertimos en milímetros y creamos dos archivos uno para posteriormente calcular la trayectoria y otro para posicionar el taladro en el lugar correcto. El primer archivo tiene una extensión AMP (AHIETI Milimetric PCB) y en segundo una extensión ADP (AHIETI Drill Position).

5.4. Sistema de trayectoria

Para facilitar el cálculo de la trayectoria se considera el movimiento del AHIETI v 2.1.1 parecido al de una impresora, en estas los cabezales de impresión se encuentra en un deslizador que viaja a través de una echa, a su vez un eje arrastra una hoja de papel, la acción combinada de estos mecanismos hacen que se cubra por completo el espacio de la hoja, el cabezal de impresión inyecta tinta o martilla una cinta entintada en puntos especí cos para formar las imágenes.

En el AHIETI v 2.1.1 el mecanismo es muy similar, por lo que el trayecto puede calcularse como en el de la impresora, el deslizador principal viaja a través de las echas de deslizamiento para el deslizador principal subiendo el soporte del taladro del deslizador principal en los lugares donde queremos que el cobre se mantenga y bajándolo en los lugares donde queremos remover el cobre, a su vez cuando el deslizador principal termina su camino regresa a su posición inicial y los deslizadores derecho con sprocket e izquierdo avanzan una posición, el deslizador principal realiza su viaje nuevamente, así sucesivamente hasta cubrir toda la superficie de la placa.

De acuerdo con las posiciones de las líneas que describen las pistas en el archivo AMP se calculan las ecuaciones de cada una de ellas, y se determina en que punto estas ecuaciones interceptan la trayectoria, con el ancho de la línea se calcula no solo el punto en donde la ecuación intercepta la trayectoria, sino también los puntos donde inicia y finaliza la pista. Así el programa determina si el taladro descobrea o solo se traslada.

5.5. Interfase

Esta fue desarrollada en el lenguaje de programación que elegimos Visual Studio .NET 2003, es una interfase gráfica muy amigable con el usuario, no requiere de experiencia para ser usado.

Consta de una ventana de introducción o presentación del programa, un menú de opciones, una ventana del interprete, una ventana para graficar el archivo y comprobar que se encuentra el circuito completo y correcto, y otra ventana en la cual se simula la trayectoria y posteriormente la ejecuta.

5.5.1. Introducción

Es una ventana e presentación, en ella se encuentra el nombre de los autores, el nombre y versión del proyecto.

Figura 5.3: Presentación AHIETI-SC

En la ventana de la gura5.3 estará abierta por 30 segundos y se cerrara automáticamente, también se cerrará si se hace click sobre ella.

5.5.2. Menú

Al cerrarse la ventana de presentación aparece el menú principal que se muestra en la figura 5.4, que posee cuatro opciones:

Abrir archivo PCB y generar archivos de soporte. Abre otra ventana desde la cual se elije el archivo PCB con el que se va atrabajar y genera los archivos AMP, APP y ADP que se usan en los siguientes módulos. Si no se generan los archivos de soporte, las demás opciones no se activan.

Graficar PCB. Pide una confirmación y abre una ventana en donde se puede graficar el archivo.

Generar pistas PCB. Pide una confirmación y abre una ventana parecida a la anterior pero aquí se tienen las opciones de simular la trayectoria que llevará el mini-drill o de ejecutarla.

Salir. Termina la ejecución del programa.

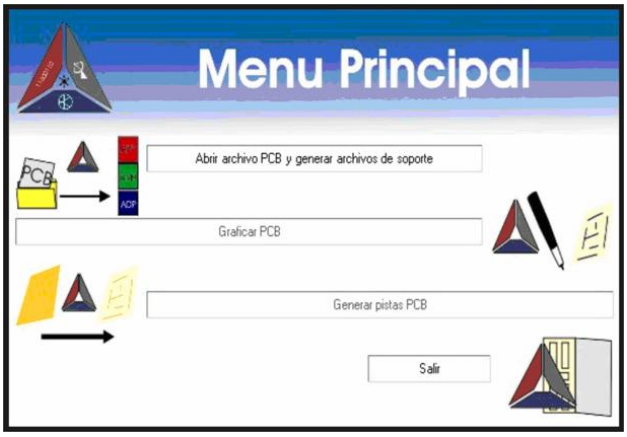


Figura 5.4: Menú AHIETI-SC

5.5.3. Abrir archivo PCB y generar archivo de soporte

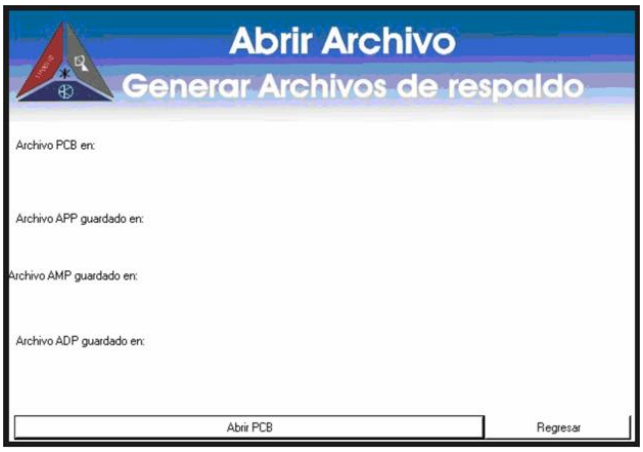


Figura 5.5: Ventana Abrir Archivo PCB y generar archivo de soporte

La ventana abrir archivo PCB solo tiene 2 botones una para abrir el archivo PCB y otro para salir como se muestra en la figura 5.5, al presionar el botón abrir archivo PCB se abre el cuadro de dialogo abrir archivo, posteriormente abre una serie de 3 cuadros de diálogo guardar como se puede ver en la figura 5.6.

Figura 5.6: Cuadro de dialogo Abrir

Lee el archivo PCB indicado genera las cabeceras de los archivos AMP, APP y ADP, si el archivo ya existe lo sobrescribe, comienza a leer línea por línea, si la línea es una etiqueta lee la siguiente línea, la descompone y la guarda en variables, los valores posteriormente son convertidos a milímetros multiplicándolos por 2.54, esto es para manejar solo números enteros y no crear confusión al guardar los archivos de soporte, después se divide entre 0.5 para convertirlos en un equivalente de píxeles.

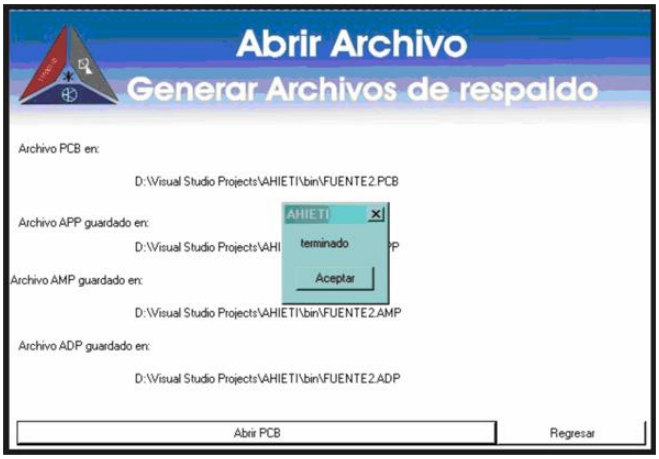


Figura 5.7: Proceso Terminado

Como las operaciones son internas y transparentes para el usuario, al finalizar el trabajo se muestra un mensaje como el de la figura 5.7 que indica que el programa termino las operaciones y que se puede continuar.

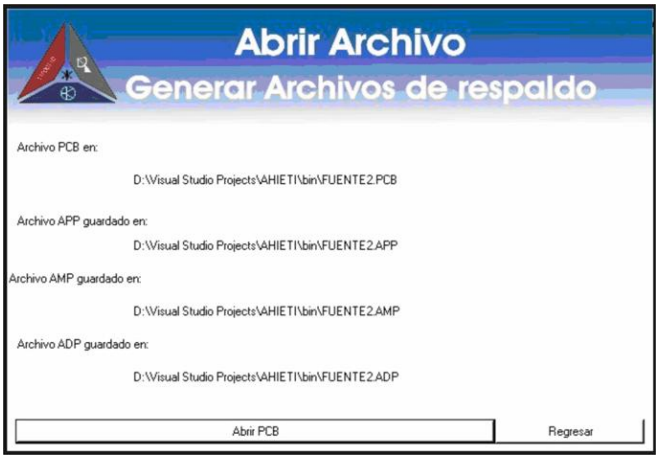


Figura 5.8: Lista de archivos generados

Al terminar los procesos nos aparece una lista de los archivos generados con la dirección en la que se están guardando asi como en la figura 5.8.

Los archivos se han generado correctamente, las direcciones y nombres de los archivos están guardados en variables globales dentro del programa para poder usarlos en los siguientes pasos sin que se requiera buscar los nombres o los directorios.

5.5.4. Graficar PCB

Cuando los archivos de soporte ya han sido generados el menú activa el acceso a la ventana de la gura5.9.

Figura 5.9: Ventana Graficar PCB

Al iniciar la ventana se carga el archivo APP, se lee y se calcula el tamaño de la placa a usar, el tamaño calculado de la placa fenólica es en base al circuito que se va a usar y a las placas fenólicas comerciales (figura 5.10).

Figura 5.10: Placa fenólica calculada

Al presionar el botón Graficar se leen las líneas del archivo PCB se identifican los elementos que se encuentran en la placa y se procede a graficar cada una de las ecuaciones con base a las coordenadas iniciales y finales de cada una de las líneas (figura 5.11).

Figura 5.11: Placa gra cada

5.5.5. Generar pistas PCB

Al igual que la anterior solo se activa cuando se han generado previamente los archivos de soporte, la diferencia entre las ventanas es que la anterior gráfica las pistas y esta gráfica alrededor de ellas siguiendo el sistema de trayectoria. La ventana que se muestra es la siguiente.

Figura 5.12: Ventana de simulación

En la gura5.12 se muestra la placa de fenólica completamente cubierta de cobre, el simulador leerá el archivo AMP y APP para terminar la trayectoria y la posición de las perforaciones.

Al presionar el botón simular el archivo AMP es leído línea por línea, se calculan las ecuaciones de la recta con la forma dada en la expresión 5.1.

$$y = mx + b \tag{5.1}$$

Cada una de estas ecuaciones representa una línea por lo cual se genera un archivo temporal para guardar las ecuaciones obtenidas, inmediatamente se calculan los valores de x cuando y esta en la posición n (para n = 0, hasta n igual al tamaño de la placa), se toman los valores que arrojan las ecuaciones se comparan con los limites de las ecuaciones y se decide si para x= n estas ecuaciones interceptan la recta de la trayectoria, si la ecuación obstruye la trayectoria se obtiene la coordenada de la intersección, pero como el ancho de la línea no vale solo un píxel se le suma y se le resta la mitad del ancho de la línea y se obtiene una coordenada del principio y otra del final de la intercepción. Obtenidas las coordenadas de intercepción se gráfica una recta desde el principio de la superficie de la placa hasta el principio de la intercepción, la recta se interrumpe y sigue graficando a partir del final de la intercepción hasta el siguiente inicio de la intercepción continua y así sucesivamente para el valor de n, posteriormente se incrementa n en uno y se repite el proceso hasta que n sea igual al valor máximo de la placa fenólica.

Figura 5.13: Simulación en curso

El proceso de descobreado en la generación es el mismo que en la simulación solo di ere en que no se gráfica si no que se mandan las coordenadas de inicio y final de la intercepción por el puerto serial al microcontrolador y el microcontrolador se encargara de realizar los avances con respecto a las coordenadas que le son enviadas (gura 5.13).

Al finalizar tanto en la simulación como en la generación las trayectorias, e inicia la fase de la perforación de las placas en ambos casos el AHIETI v 2.1.1 regresa a su posición inicial, después tona las primeras coordenadas se coloca primero en la posición adecuada de la coordenadas en X y posteriormente en la coordenada en Y, entonces procede a perforar el primer orificio, compara las siguientes coordenadas y se posiciona en ellas y perfora, así sucesivamente hasta terminar con las perforaciones, finalmente muestra un mensaje para indicar que ha terminado y se coloca nuevamente en la posición final .

Al regresar al menú principal el usuario puede decidir si quiere iniciar un nuevo trabajo, realizar nuevamente el mismo trabajo o simplemente salir de la aplicación.

Capítulo 6

Conclusiones

El deslizamiento de la F.S.D.D del AHIETI v 1.0.0 debe tomarse con precaución, ya que presentó problemas de deslizamiento, este detalle pudo haberse corregido reemplazando el tornillo de cuerda estándar por un tornillo de cuerda cuadrada, esto no fue posible debido al alto costo y la dificultad de fabricación.

Aún así se corregía el deslizamiento con un motor de corriente continua en la echa opuesta al tornillo, pero presenta problemas de desgaste en el deslizador correspondiente y puede alterar la trayectoria de desplazamiento.

Por consiguiente con el AHIETI v 2.0.0 se corrigen estos problemas haciendo uso de dos echas para cada deslizador y un motor a paso por cada uno de ellos. No se usaron los drivers L297 y L298 por la dificultad de conseguirlos, sobre todo en la ciudad de Pachuca. Finalmente los PICs secundarios pueden llevar a cabo la misma función, además de estar apto para proveer mayores prestaciones con un diseño futuro. El sistema de trayectoria estaba dispuesto a contornear las pistas una por una, los problemas con esta técnica es que al terminar el contorno de cada elemento, este quedaba aislado de los demás. En un segundo sistema de trayectoria se ideaba calcular el punto medio entre las líneas y componentes, pero este método solo funciona con circuitos muy simples. En circuitos más complejos o de transmisión las pistas más amplias tienden a generar un ruido capacitivo que interviene con el correcto funcionamiento del circuito diseñado.

Se decide hacer una interfase sencilla usando varias ventanas, una para cada paso o función que se quiere realizar, en lugar de una sola ventana con exceso de controles que hace más difícil el manejo del programa.

El graficado y simulación del circuito se hace a escala y no a tamaño real, debido a que se quiere una mejor compatibilidad entre computadoras, si se hace en tamaño real la resolución de diferentes monitores y tarjetas gráficas pueden alterar la presentación del circuito.

El uso del puerto serial en lugar del puerto paralelo es para evitar confusiones con las conexiones, esto también nos aporta una interfase más rápida con transmisión bidireccional en modo full-duplex.

El programa TraxMaker tiene la capacidad de generar archivos para el manejo de dispositivos de producción (archivos gerber y N/C drill), por otro lado se utilizaron los archivos que el programa genera para comprender la forma en que se pueden convertir los trazos de un dibujo a movimientos en el actuador electromecánico.

Aunque el dispositivo puede comportarse como una impresora, no es nuestra meta fabri- car una, en el mercado se encuentran estos dispositivos y son demasiado costosos por com- portarse como tal.

Apéndice A

Glosario

Descobreado: Cobrear. Dar o cubrir de cobre algo. Des. (De las preps. lats. de y ex). 1. prep. desus. desde.

Des-. (Con uencia de los prefs. lats. de-, ex-, dis- y a veces e-). 1. pref. Denota negación o inversión del significado del simple. Desconfiar, deshacer. 2. pref. Indica privación. Desabejar. 3. pref. Indica exceso o demasía. Deslenguado. 4. pref. Significa 'fuera de'. Descamino, deshora. 5. pref. A veces indica afirmación. Despavorido. [19]

Mini-Drill: Moto Herramienta de alta velocidad y bajo par de propósito general, parecido a un taladro.

Sprocket: Is a toothed wheel upon which a chain rides. Contrary to popular opinion, a sprocket is not a gear. [20]

Somier. (Del fr. sommier). 1. m. Soporte de tela metálica, láminas de madera, etc., sobre el que se coloca el colchón. [19]

Horquilla. (Del dim. de horca). 1. f. horca (II de labrador). 2. f. Herramienta en forma de horca de labrador para diversos usos. 3. f. horca (II palo para sostener las ramas de los árboles). 4. f. Pie para apoyar las armas de fuego. 5. f. Palo terminado en uno de sus extremos por dos puntas. 6. f. Pieza de un mecanismo con forma de Y, que suele servir para sujetar otras piezas o hacerlas girar. 7. f. horqueta (II parte del árbol). 8. f. Bifurcación que se produce en el extremo de algo. 9. f. Pieza metálica o de otro material, que se emplea para sujetar el pelo. 10. f. Pieza que en las bicicletas, motocicletas y vehículos de similares características va desde la rueda delantera hasta el manillar. 11. f. Distancia o espacio entre dos magnitudes dadas. 12. f. Mar. Pieza en forma de V sobre la que se apoya el remo en determinadas embarcaciones. 13. f. p. us. Deterioro del cabello que hiende en dos sus puntas. 14. f. desus. fúrcula. [19]

Escariador. (De escariar). 1. m. Herramienta para escariar. [19]

Escariar. 1. tr. Agrandar o redondear un agujero abierto en metal, o el diámetro de un tubo, por medio de herramientas adecuadas. [19]

Terraja. 1. f. Tabla guarnecida con una chapa de metal recortada con arreglo al perfil

de una moldura, y que sirve para hacer las de yeso, estuco o mortero, corriéndola cuando la pasta está blanda. 2. f. Herramienta formada por una barra de acero con una caja rectangular en el medio, donde se ajustan las piezas que sirven para labrar las roscas de los tornillos. 3. f. despect. Ur. Objeto de mala calidad. U. t. c. adj. 4. com. despect. Ur. En lenguaje juvenil, persona de condición social baja. 5. com. despect. Ur. En lenguaje juvenil, persona mal vestida. [19]

Vanadio. (De Vanadis, diosa de la mitología escandinava). 1. m. Elemento químico de núm. atóm. 23. Metal escaso en la corteza terrestre, se encuentra disperso en minerales de hierro, titanio y fósforo, y en forma de óxido, asociado al plomo. De color gris claro, dúctil y resistente a la corrosión, se usa como catalizador, y, aleado con aluminio o con hierro, mejora las propiedades mecánicas del hierro, el acero y el titanio. (Símb. V). [19]

Herrumbre. (Del lat. ferrumen, -inis). 1. f. Óxido del hierro. 2. f. Gusto o sabor que algunas cosas, como las aguas, toman del hierro. 3. f. roya (II hongo parásito). [19]

Feldespato. (Del al. Feldspat). 1. m. Nombre común de diversas especies minerales, de color blanco, amarillento o rojizo, brillo resinoso o nacarado y gran dureza, que forman parte de rocas ígneas, como el granito. Químicamente son silicatos complejos de aluminio con sodio, potasio o calcio, y cantidades pequeñas de óxidos de magnesio y hierro. Entre los feldespatos más importantes están la ortosa, la albita y la labradorita. [19]

Anilina. (Del al. Anilin, y este del port. anil, añil, índigo). 1. f. Quím. Amina aromática, oleosa, incolora, tóxica por ingestión, inhalación o absorción a través de la piel, que tiene muchas aplicaciones industriales, especialmente en la fabricación de colorantes. 2. f. U. para referirse popularmente a diversos productos utilizados como colorantes.[19]

SQL. Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Aúna características del álgebra y el cálculo relacional permitiendo lanzar consultas con el n de recuperar información de interés de una base de datos, de una forma sencilla.

Molibdeno. (Del lat. molybdaena, y este del gr. trocito de plomo). 1. m. Elemento químico de núm. atóm. 42. Metal escaso en la corteza terrestre, se encuentra generalmente en forma de sulfuro. De color gris o negro y brillo plateado, pesado y con un elevado punto de fusión, es blando y dúctil en estado puro, pero quebradizo si presenta impurezas. Se usa en la fabricación de aceros y lamentos resistentes a altas temperaturas. (Símb. Mo). [19]

MIL: Es la mínima unidad de longitud, en el sistema inglés de medidas, que equivale a la milésima parte de una pulgada.

Apéndice B

Lista de acrónimos y abreviaturas

F.S.D.D: Flecha de Soporte de Deslizamiento para Deslizadores.

F.S.D.D.P: Flecha de Soporte de Deslizamiento para Deslizador Principal.

S.F.D.D.P.M.P: Soporte para Flechas de Soporte de Deslizamiento para Deslizador con Placa para Motor a Paso

S.F.S.D.D.S: Soporte para Flechas de Soporte de Deslizamiento para Deslizador con Sprocket

PCB: Printed Circuit Board

MCU: Microcontrolador

GNU GPL: (General Public License o licencia pública general) es una licencia creada por la Free Software Foundation a mediados de los 80, y esta orientada principalmente a los términos de distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre. [21]

RAD: Rapid application development, Desarrollo Rapido de Aplicaciones.

SQL: Structured Query Lenguaje, Lenguaje de consulta estructurada.

Apéndice C

Optoacopladores

Funcionamiento del optoacoplador

Los foto-emisores que se emplean en los optoacopladores de potencia son diodos que emiten rayos infrarrojos (IRED) y los fotorreceptores pueden ser tiristores o transistores.

Cuando aparece una tensión sobre los terminales del diodo IRED, este emite un haz de rayos infrarrojo que transmite a través de una pequeña guia-ondas de plástico o cristal hacia el fotorreceptor. La energía luminosa que incide sobre el fotorreceptor hace que este genere una tensión eléctrica a su salida. Este responde a las señales de entrada, que podrían ser pulsos de tensión.

En general pueden sustituir a relés ya que tienen una velocidad de conmutación mayor, así como, la ausencia de rebotes.

Diferentes tipos de optoacopladores

Fototransistor

Se compone de un optoacoplador con una etapa de salida formada por un transistor BJT o de un arreglo Darlington figura C.1.

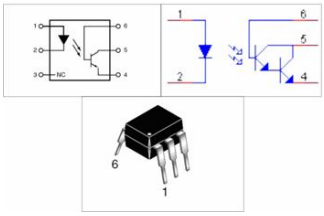


Figura C.1: Fototransistor BJT y Darlington

Fototriac

Dispone de un optoacoplador con una etapa de salida formada por un triac figura C.2.

Fototriac de paso por cero

Optoacoplador en cuya etapa de salida se encuentra un triac de cruce por cero figura C.2. El circuito interno de cruce por cero conmuta al triac sólo en los cruce por cero de la corriente alterna.

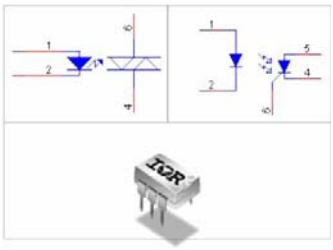


Figura C.2: Fototriac's

Apéndice D

Hojas de datos

FAIRCHILD
SEMICONDUCTOR®

TIP110/111/112

Monolithic Construction With Built In Base-Emitter Shunt Resistors

- Complementary to TIP115/116/117
- High DC Current Gain : h_{FE}=1000 @ V_{CE}=4V, I_C=1A(Min.)
- Low Collector-Emitter Saturation Voltage
- Industrial Use

TO-220

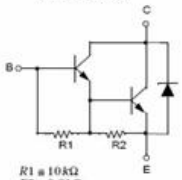
1.Base 2.Collector 3.Emitter

NPN Epitaxial Silicon Darlington Transistor

Absolute Maximum Ratings T_C=25°C unless otherwise noted

Symbol	Parameter		Value	Units
V_{CBO}	Collector-Base Voltage	: TIP110	60	V
		: TIP111	80	V
		: TIP112	100	V
V_{CEO}	Collector-Emitter Voltage	: TIP110	60	V
		: TIP111	80	V
		: TIP112	100	V
V_{EBO}	Emitter-Base Voltage		5	V
I_C	Collector Current (DC)		2	A
I_{CP}	Collector Current (Pulse)		4	A
I_B	Base Current (DC)		50	mA
P_C	Collector Dissipation (T_a=25°C)		2	W
	Collector Dissipation (T_C=25°C)		50	W
T_J	Junction Temperature		150	°C
T_{STG}	Storage Temperature		- 65 ~ 150	°C

Equivalent Circuit



$R1 \approx 10k\Omega$
$R2 \approx 0.6k\Omega$

Electrical Characteristics T_C=25°C unless otherwise noted

Symbol	Parameter	Test Condition	Min.	Max.	Units
V_{CEO}(sus)	Collector-Emitter Sustaining Voltage				
	: TIP110	I_C = 30mA, I_B = 0	60		V
	: TIP111		80		V
	: TIP112		100		V
I_{CEO}	Collector Cut-off Current				
	: TIP110	V_{CE} = 30V, I_B = 0		2	mA
	: TIP111	V_{CE} = 40V, I_B = 0		2	mA
	: TIP112	V_{CE} = 50V, I_B = 0		2	mA
I_{CBO}	Collector Cut-off Current				
	: TIP110	V_{CB} = 60V, I_E = 0		1	mA
	: TIP111	V_{CB} = 80V, I_E = 0		1	mA
	: TIP112	V_{CB} = 100V, I_E = 0		1	mA
I_{EBO}	Emitter Cut-off Current	V_{BE} = 5V, I_C = 0		2	mA
h_{FE}	DC Current Gain	V_{CE} = 4V, I_C = 1A	1000		
		V_{CE} = 4V, I_C = 2A	500		
V_{CE}(sat)	Collector-Emitter Saturation Voltage	I_C = 2A, I_B = 8mA		2.5	V
V_{BE}(on)	Base-Emitter ON Voltage	V_{CE} = 4V, I_C = 2A		2.8	V
C_{ob}	Output Capacitance	V_{CB} = 10V, I_E = 0, f = 0.1MHz		100	pF

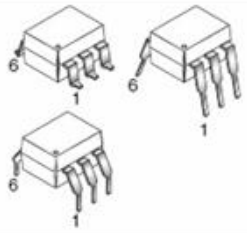
Rev. A1, June 2001

FAIRCHILD
SEMICONDUCTOR®

GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

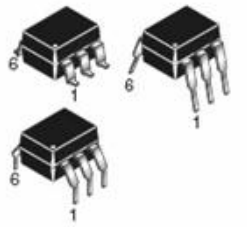
WHITE PACKAGE (-M SUFFIX)



SCHEMATIC

PIN 1. ANODE
2. CATHODE
3. NO CONNECTION
4. EMITTER
5. COLLECTOR
6. BASE

BLACK PACKAGE (NO -M SUFFIX)



DESCRIPTION

The general purpose optocouplers consist of a gallium arsenide infrared emitting diode driving a silicon phototransistor in a 6-pin dual in-line package.

FEATURES

- Also available in white package by specifying -M suffix, eg. 4N25-M
- UL recognized (File # E90700)
- VDE recognized (File # 94766)
  - Add option V for white package (e.g., 4N25V-M)
  - Add option 300 for black package (e.g., 4N25.300)

APPLICATIONS

- Power supply regulators
- Digital logic inputs
- Microprocessor inputs

FAIRCHILD
SEMICONDUCTOR®

GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

ABSOLUTE MAXIMUM RATINGS (T_A = 25°C unless otherwise specified)

Parameter	Symbol	Value	Units
TOTAL DEVICE			
Storage Temperature	T_{STG}	-55 to +150	°C
Operating Temperature	T_{OPR}	-55 to +100	°C
Lead Solder Temperature	T_{SOL}	260 for 10 sec	°C
Total Device Power Dissipation @ T_A = 25°C	P_D	250	mW
Derate above 25°C		3.3 (non-M), 2.94 (-M)	
EMITTER			
DC/Average Forward Input Current	I_F	100 (non-M), 60 (-M)	mA
Reverse Input Voltage	V_R	6	V
Forward Current - Peak (300µs, 2% Duty Cycle)	I_F(pk)	3	A
LED Power Dissipation @ T_A = 25°C	P_D	150 (non-M), 120 (-M)	mW
Derate above 25°C		2.0 (non-M), 1.41 (-M)	mW/°C
DETECTOR			
Collector-Emitter Voltage	V_{CEO}	30	V
Collector-Base Voltage	V_{CBO}	70	V
Emitter-Collector Voltage	V_{ECO}	7	V
Detector Power Dissipation @ T_A = 25°C	P_D	150	mW
Derate above 25°C		2.0 (non-M), 1.76 (-M)	mW/°C

FAIRCHILD
SEMICONDUCTOR®

GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

ELECTRICAL CHARACTERISTICS (T_A = 25°C unless otherwise specified)

INDIVIDUAL COMPONENT CHARACTERISTICS

Parameter	Test Conditions	Symbol	Min	Typ*	Max	Unit
EMITTER						
Input Forward Voltage	(I_F = 10 mA)	V_F		1.18	1.50	V
Reverse Leakage Current	(V_R = 6.0 V)	I_R		0.001	10	µA
DETECTOR						
Collector-Emitter Breakdown Voltage	(I_C = 1.0 mA, I_F = 0)	BV_{CEO}	30	100		V
Collector-Base Breakdown Voltage	(I_C = 100 µA, I_F = 0)	BV_{CBO}	70	120		V
Emitter-Collector Breakdown Voltage	(I_E = 100 µA, I_F = 0)	BV_{ECO}	7	10		V
Collector-Emitter Dark Current	(V_{CE} = 10 V, I_F = 0)	I_{CEO}		1	50	nA
Collector-Base Dark Current	(V_{CB} = 10 V)	I_{CBO}			20	nA
Capacitance	(V_{CE} = 0 V, f = 1 MHz)	C_{CE}		8		pF

ISOLATION CHARACTERISTICS

Characteristic	Test Conditions	Symbol	Min	Typ*	Max	Units
Input-Output Isolation Voltage	(Non '-M', Black Package) (f = 60 Hz, t = 1 min)	V_{ISO}	5300			Vac(rms)
	('-M', White Package) (f = 60 Hz, t = 1 sec)		7500			Vac(pk)
Isolation Resistance	(V_{I-O} = 500 VDC)	R_{ISO}	10^{11}			Ω
Isolation Capacitance	(V_{I-O} = &, f = 1 MHz)	C_{ISO}		0.5		pF
	('-M' White Package)			0.2	2	pF

Note
* Typical values at T_A = 25°C

FAIRCHILD SEMICONDUCTOR®

GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25 4N26 4N27 4N28 4N35 4N36 4N37 H11A1 H11A2 H11A3 H11A4 H11A5

TRANSFER CHARACTERISTICS (T_A = 25°C Unless otherwise specified.)

DC Characteristic	Test Conditions	Symbol	Device	Min	Typ*	Max	Unit
Current Transfer Ratio, Collector to Emitter	(I_F = 10 mA, V_{CE} = 10 V)	CTR	4N35 4N36 4N37	100			%
			H11A1	50			
			H11A5	30			
			4N25 4N26 H11A2 H11A3	20			
			4N27 4N28 H11A4	10			
	(I_F = 10 mA, V_{CE} = 10 V, T_A = -55°C)		4N35 4N36 4N37	40			
	(I_F = 10 mA, V_{CE} = 10 V, T_A = +100°C)		4N35 4N36 4N37	40			
Collector-Emitter Saturation Voltage	(I_C = 2 mA, I_F = 50 mA)	$V_{CE\ (SAT)}$	4N25 4N26 4N27 4N28			0.5	V
	(I_C = 0.5 mA, I_F = 10 mA)		4N35 4N36 4N37			0.3	
			H11A1 H11A2 H11A3 H11A4 H11A5			0.4	
AC Characteristic Non-Saturated Turn-on Time	(I_F = 10 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{ON}	4N25 4N26 4N27 4N28 H11A1 H11A2 H11A3 H11A4 H11A5		2		µs
Non Saturated Turn-on Time	(I_C = 2 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{ON}	4N35 4N36 4N37		2	10	µs

FAIRCHILD
SEMICONDUCTOR®

GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25 4N26 4N27 4N28 4N35 4N36
4N37 H11A1 H11A2 H11A3 H11A4 H11A5

TRANSFER CHARACTERISTICS (T_A = 25°C Unless otherwise specified.) (Continued)

AC Characteristic	Test Conditions	Symbol	Device	Min	Typ*	Max	Unit
Turn-off Time	(I_F = 10 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{OFF}	4N25 4N26 4N27 4N28 H11A1 H11A2 H11A3 H11A4 H11A5		2		µs
	(I_C = 2 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)		4N35 4N36 4N37		2	10	

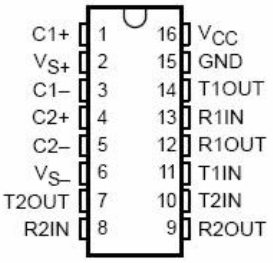
* Typical values at T_A = 25°C

MAX232, MAX232I
DUAL EIA-232 DRIVERS/RECEIVERS

SLLS047I – FEBRUARY 1989 – REVISED OCTOBER 2002

- **Meet or Exceed TIA/EIA-232-F and ITU Recommendation V.28**
- **Operate With Single 5-V Power Supply**
- **Operate Up to 120 kbit/s**
- **Two Drivers and Two Receivers**
- **±30-V Input Levels**
- **Low Supply Current . . . 8 mA Typical**
- **Designed to be Interchangeable With Maxim MAX232**
- **ESD Protection Exceeds JESD 22 – 2000-V Human-Body Model (A114-A)**
- **Applications**
  - **TIA/EIA-232-F**
  - **Battery-Powered Systems**
  - **Terminals**
  - **Modems**
  - **Computers**

MAX232 . . . D, DW, N, OR NS PACKAGE
MAX232I . . . D, DW, OR N PACKAGE
(TOP VIEW)



description/ordering information

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply EIA-232 voltage levels from a single 5-V supply. Each receiver converts EIA-232 inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V and a typical hysteresis of 0.5 V, and can accept ±30-V inputs. Each driver converts TTL/CMOS input levels into EIA-232 levels. The driver, receiver, and voltage-generator functions are available as cells in the Texas Instruments LinASIC™ library.

ORDERING INFORMATION

T_A	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	PDIP (N)	Tube	MAX232N	MAX232N
	SOIC (D)	Tube	MAX232D	MAX232
		Tape and reel	MAX232DR	
	SOIC (DW)	Tube	MAX232DW	MAX232
		Tape and reel	MAX232DWR	
	SOP (NS)	Tape and reel	MAX232NSR	MAX232
–40°C to 85°C	PDIP (N)	Tube	MAX232IN	MAX232IN
	SOIC (D)	Tube	MAX232ID	MAX232I
		Tape and reel	MAX232IDR	
	SOIC (DW)	Tube	MAX232IDW	MAX232I
		Tape and reel	MAX232IDWR	

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

Function Tables

EACH DRIVER

INPUT TIN	OUTPUT TOUT
L	H
H	L

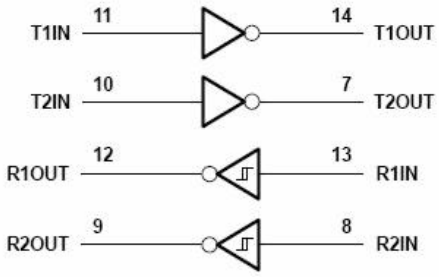
H = high level, L = low level

EACH RECEIVER

INPUT RIN	OUTPUT ROUT
L	H
H	L

H = high level, L = low level

logic diagram (positive logic)



absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Input supply voltage range, V_{CC} (see Note 1) . . . . . –0.3 V to 6 V
Positive output supply voltage range, V_{S+} . . . . . V_{CC} – 0.3 V to 15 V
Negative output supply voltage range, V_{S-} . . . . . –0.3 V to –15 V
Input voltage range, V_I: Driver . . . . . –0.3 V to V_{CC} + 0.3 V
Receiver . . . . . ±30 V
Output voltage range, V_O: T1OUT, T2OUT . . . . . V_{S-} – 0.3 V to V_{S+} + 0.3 V
R1OUT, R2OUT . . . . . –0.3 V to V_{CC} + 0.3 V
Short-circuit duration: T1OUT, T2OUT . . . . . Unlimited
Package thermal impedance, θ_{JA} (see Note 2): D package . . . . . 73°C/W
DW package . . . . . 57°C/W
N package . . . . . 67°C/W
NS package . . . . . 64°C/W
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds . . . . . 260°C
Storage temperature range, T_{stg} . . . . . –65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: All voltage values are with respect to network ground terminal.
2. The package thermal impedance is calculated in accordance with JESD 51-7.

recommended operating conditions

			MIN	NOM	MAX	UNIT
V_{CC}	Supply voltage		4.5	5	5.5	V
V_{IH}	High-level input voltage (T1IN,T2IN)		2			V
V_{IL}	Low-level input voltage (T1IN, T2IN)				0.8	V
R1IN, R2IN	Receiver input voltage				±30	V
T_A	Operating free-air temperature	MAX232	0		70	°C
		MAX232I	–40		85	

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted) (see Note 3 and Figure 4)

PARAMETER		TEST CONDITIONS	MIN	TYP‡	MAX	UNIT
I_{CC}	Supply current	V_{CC} = 5.5 V, All outputs open, T_A = 25°C		8	10	mA

‡ All typical values are at V_{CC} = 5 V and T_A = 25°C.
NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

MAX232, MAX232I
DUAL EIA-232 DRIVERS/RECEIVERS

SLLS047I – FEBRUARY 1989 – REVISED OCTOBER 2002

DRIVER SECTION

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature range (see Note 3)

PARAMETER			TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	T1OUT, T2OUT	R_L = 3 kΩ to GND		5	7		V
V_{OL}	Low-level output voltage‡	T1OUT, T2OUT	R_L = 3 kΩ to GND			–7	–5	V
r_o	Output resistance	T1OUT, T2OUT	V_{S+} = V_{S-} = 0,	V_O = ±2 V	300			Ω
I_{OS}§	Short-circuit output current	T1OUT, T2OUT	V_{CC} = 5.5 V,	V_O = 0		±10		mA
I_{IS}	Short-circuit input current	T1IN, T2IN	V_I = 0				200	µA

† All typical values are at V_{CC} = 5 V, T_A = 25°C.
‡ The algebraic convention, in which the least positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.
§ Not more than one output should be shorted at a time.
NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 3)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
SR	Driver slew rate	R_L = 3 kΩ to 7 kΩ, See Figure 2			30	V/µs
SR(t)	Driver transition region slew rate	See Figure 3		3		V/µs
	Data rate	One TOUT switching		120		kbit/s

NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

RECEIVER SECTION

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature range (see Note 3)

PARAMETER			TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	R1OUT, R2OUT	I_{OH} = –1 mA		3.5			V
V_{OL}	Low-level output voltage‡	R1OUT, R2OUT	I_{OL} = 3.2 mA				0.4	V
V_{IT+}	Receiver positive-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V,	T_A = 25°C		1.7	2.4	V
V_{IT-}	Receiver negative-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V,	T_A = 25°C	0.8	1.2		V
V_{hys}	Input hysteresis voltage	R1IN, R2IN	V_{CC} = 5 V		0.2	0.5	1	V
r_i	Receiver input resistance	R1IN, R2IN	V_{CC} = 5,	T_A = 25°C	3	5	7	kΩ

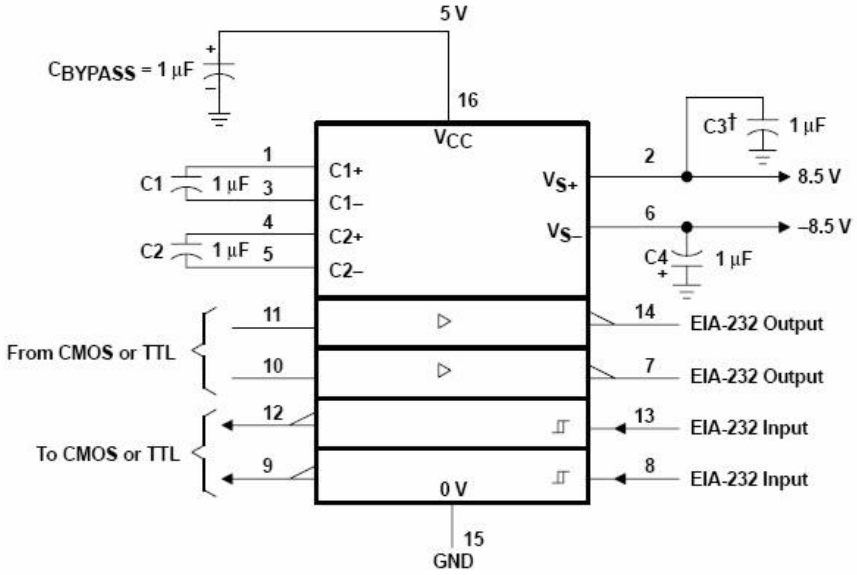
† All typical values are at V_{CC} = 5 V, T_A = 25°C.
‡ The algebraic convention, in which the least positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.
NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 3 and Figure 1)

PARAMETER		TYP	UNIT
$t_{PLH(R)}$	Receiver propagation delay time, low- to high-level output	500	ns
$t_{PHL(R)}$	Receiver propagation delay time, high- to low-level output	500	ns

NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

APPLICATION INFORMATION



† C3 can be connected to V_{CC} or GND.

PIC16F87XA
Data Sheet

28/40/44-Pin Enhanced Flash Microcontrollers

MICROCHIP

PIC16F87XA

28/40/44-Pin Enhanced Flash Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
  DC – 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I^2C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) – 8 bits wide with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (V_{REF}) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

CMOS Technology:

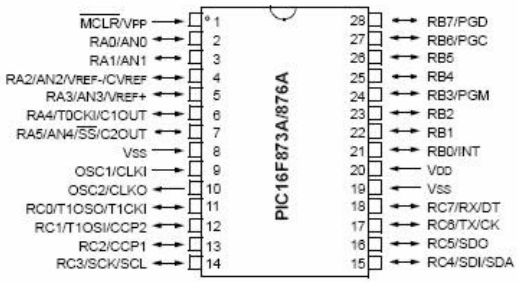
- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I^2C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

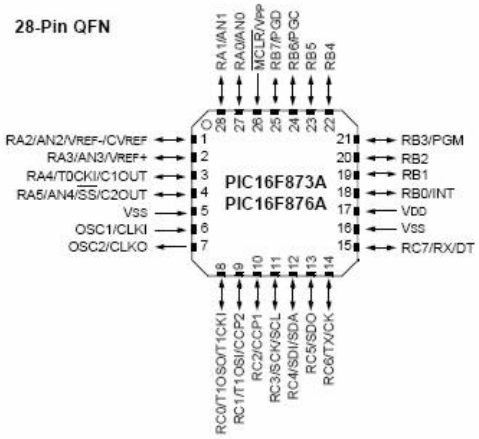
PIC16F87XA

Pin Diagrams

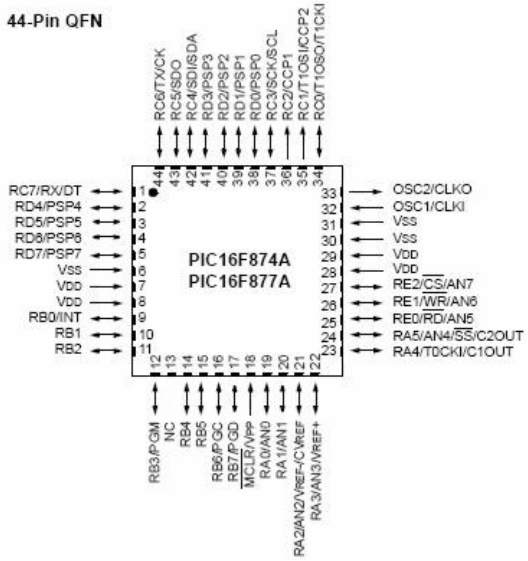
28-Pin PDIP, SOIC, SSOP



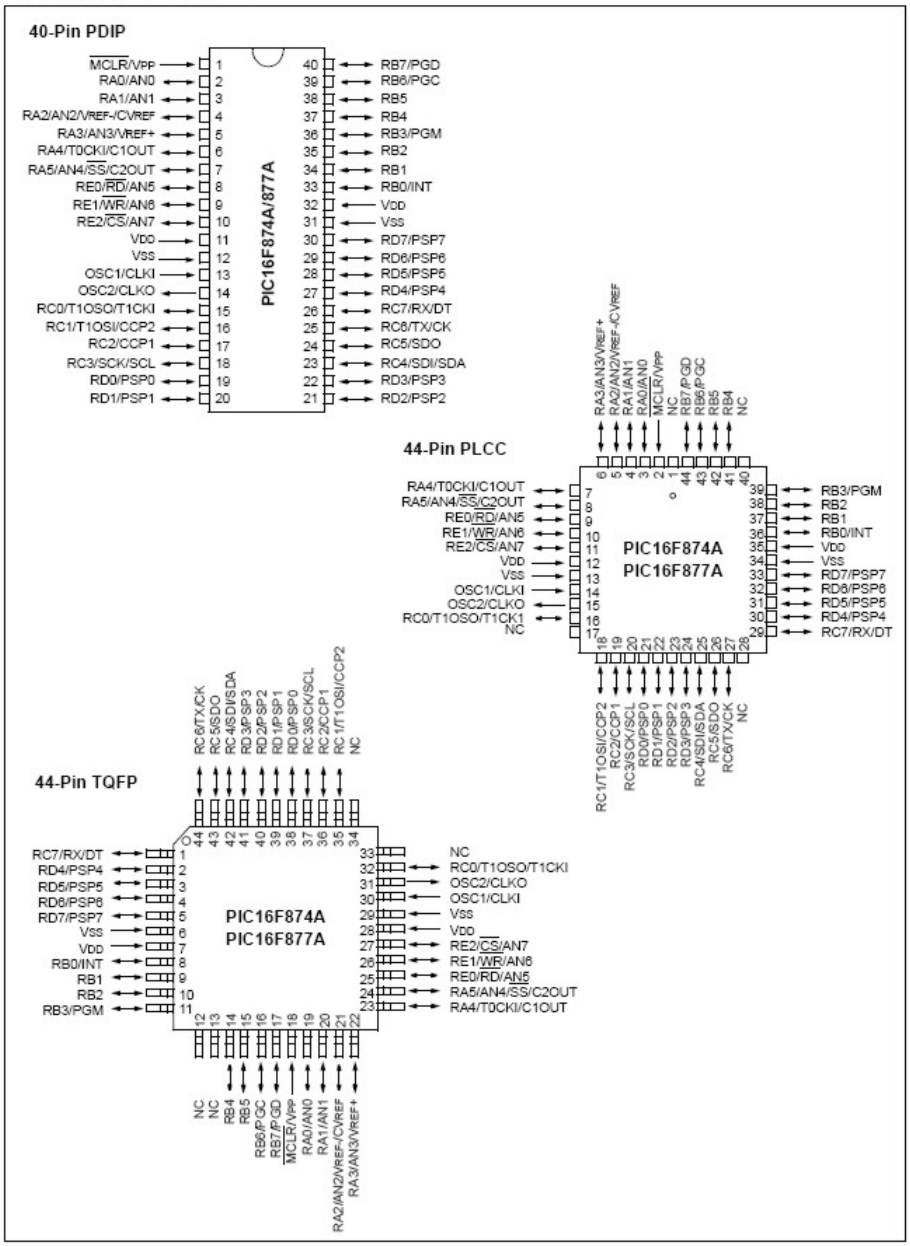
28-Pin QFN



44-Pin QFN



Pin Diagrams (Continued)



1.0 DEVICE OVERVIEW

This document contains device specific information about the following devices:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

PIC16F873A/876A devices are available only in 28-pin packages, while PIC16F874A/877A devices are available in 40-pin and 44-pin packages. All devices in the PIC16F87XA family share common architecture with the following differences:

- The PIC16F873A and PIC16F874A have one-half of the total on-chip memory of the PIC16F876A and PIC16F877A
- The 28-pin devices have three I/O ports, while the 40/44-pin devices have five
- The 28-pin devices have fourteen interrupts, while the 40/44-pin devices have fifteen
- The 28-pin devices have five A/D input channels, while the 40/44-pin devices have eight
- The Parallel Slave Port is implemented only on the 40/44-pin devices

The available features are summarized in Table 1-1. Block diagrams of the PIC16F873A/876A and PIC16F874A/877A devices are provided in Figure 1-1 and Figure 1-2, respectively. The pinouts for these device families are listed in Table 1-2 and Table 1-3.

Additional information may be found in the PICmicro® Mid-Range Reference Manual (DS33023), which may be obtained from your local Microchip Sales Representative or downloaded from the Microchip web site. The Reference Manual should be considered a complementary document to this data sheet and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

TABLE 1-1: PIC16F87XA DEVICE FEATURES

Key Features	PIC16F873A	PIC16F874A	PIC16F876A	PIC16F877A
Operating Frequency	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz
Resets (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
Flash Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM Data Memory (bytes)	128	128	256	256
Interrupts	14	15	14	15
I/O Ports	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C	Ports A, B, C, D, E
Timers	3	3	3	3
Capture/Compare/PWM modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	—	PSP	—	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Analog Comparators	2	2	2	2
Instruction Set	35 Instructions	35 Instructions	35 Instructions	35 Instructions
Packages	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN

FIGURE 2-3: PIC16F876A/877A REGISTER FILE MAP

Bank 0	File Address	Bank 1	File Address	Bank 2	File Address	Bank 3	File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD(1)	08h	TRISD(1)	88h		108h		188h
PORTE(1)	09h	TRISE(1)	89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(2)	18Eh
TMR1H	0Fh		8Fh	EEADRH	10Fh	Reserved(2)	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h	General Purpose Register 16 Bytes	117h	General Purpose Register 16 Bytes	197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch	CMCON	9Ch		11Ch		19Ch
CCP2CON	1Dh	CVRCON	9Dh		11Dh		19Dh
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
General Purpose Register 96 Bytes	20h	General Purpose Register 80 Bytes	A0h	General Purpose Register 80 Bytes	120h	General Purpose Register 80 Bytes	1A0h
	7Fh		EFh		16Fh		1EFh
		accesses 70h-7Fh	F0h	accesses 70h-7Fh	170h	accesses 70h - 7Fh	1F0h
			FFh		17Fh		1FFh

Unimplemented data memory locations, read as '0'.
* Not a physical register.

Note 1: These registers are not implemented on the PIC16F876A.
2: These registers are reserved; maintain these registers clear.

PIC16F87XA

FIGURE 2-4: PIC16F873A/874A REGISTER FILE MAP

Bank 0	File Address	Bank 1	File Address	Bank 2	File Address	Bank 3	File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD(1)	08h	TRISD(1)	88h		108h		188h
PORTE(1)	09h	TRISE(1)	89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(2)	18Eh
TMR1H	0Fh		8Fh	EEADRH	10Fh	Reserved(2)	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h				
T2CON	12h	PR2	92h				
SSPBUF	13h	SSPADD	93h				
SSPCON	14h	SSPSTAT	94h				
CCPR1L	15h		95h				
CCPR1H	16h		96h				
CCP1CON	17h		97h				
RCSTA	18h	TXSTA	98h				
TXREG	19h	SPBRG	99h				
RCREG	1Ah		9Ah				
CCPR2L	1Bh		9Bh				
CCPR2H	1Ch	CMCON	9Ch				
CCP2CON	1Dh	CVRCON	9Dh				
ADRESH	1Eh	ADRESL	9Eh				
ADCON0	1Fh	ADCON1	9Fh				
	20h		A0h		120h		1A0h
General Purpose Register 96 Bytes	7Fh	General Purpose Register 96 Bytes	FFh	accesses 20h-7Fh	16Fh 170h 17Fh	accesses A0h - FFh	1EFh 1F0h 1FFh

Unimplemented data memory locations, read as '0'.
* Not a physical register.

Note 1: These registers are not implemented on the PIC16F873A.
2: These registers are reserved; maintain these registers clear.

6.0 TIMER1 MODULE

The Timer1 module is a 16-bit timer/counter consisting of two 8-bit registers (TMR1H and TMR1L) which are readable and writable. The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The TMR1 interrupt, if enabled, is generated on overflow which is latched in interrupt flag bit, TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing TMR1 interrupt enable bit, TMR1IE (PIE1<0>).

Timer1 can operate in one of two modes:

- As a Timer
- As a Counter

The operating mode is determined by the clock select bit, TMR1CS (T1CON<1>).

In Timer mode, Timer1 increments every instruction cycle. In Counter mode, it increments on every rising edge of the external clock input.

Timer1 can be enabled/disabled by setting/clearing control bit, TMR1ON (T1CON<0>).

Timer1 also has an internal "Reset input". This Reset can be generated by either of the two CCP modules (**Section 8.0 "Capture/Compare/PWM Modules"**). Register 6-1 shows the Timer1 Control register.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI/CCP2 and RC0/T1OSO/T1CKI pins become inputs. That is, the TRISC<1:0> value is ignored and these pins read as '0'.

Additional information on timer modules is available in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023).

REGISTER 6-1: T1CON: TIMER1 CONTROL REGISTER (ADDRESS 10h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **T1CKPS1:T1CKPS0:** Timer1 Input Clock Prescale Select bits
11 = 1:8 prescale value
10 = 1:4 prescale value
01 = 1:2 prescale value
00 = 1:1 prescale value

bit 3 **T1OSCEN**: Timer1 Oscillator Enable Control bit
1 = Oscillator is enabled
0 = Oscillator is shut-off (the oscillator inverter is turned off to eliminate power drain)

bit 2 **$\overline{\text{T1SYNC}}$**: Timer1 External Clock Input Synchronization Control bit
When TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS**: Timer1 Clock Source Select bit
1 = External clock from pin RC0/T1OSO/T1CKI (on the rising edge)
0 = Internal clock (Fosc/4)

bit 0 **TMR1ON**: Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

6.1 Timer1 Operation in Timer Mode

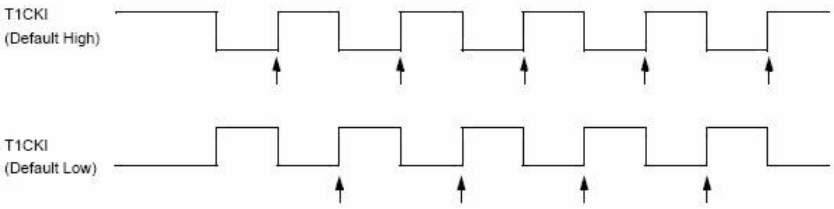
Timer mode is selected by clearing the TMR1CS (T1CON<1>) bit. In this mode, the input clock to the timer is Fosc/4. The synchronize control bit, $\overline{\text{T1SYNC}}$ (T1CON<2>), has no effect since the internal clock is always in sync.

6.2 Timer1 Counter Operation

Timer1 may operate in either a Synchronous, or an Asynchronous mode, depending on the setting of the TMR1CS bit.

When Timer1 is being incremented via an external source, increments occur on a rising edge. After Timer1 is enabled in Counter mode, the module must first have a falling edge before the counter begins to increment.

FIGURE 6-1: TIMER1 INCREMENTING EDGE



Note: Arrows indicate counter increments.

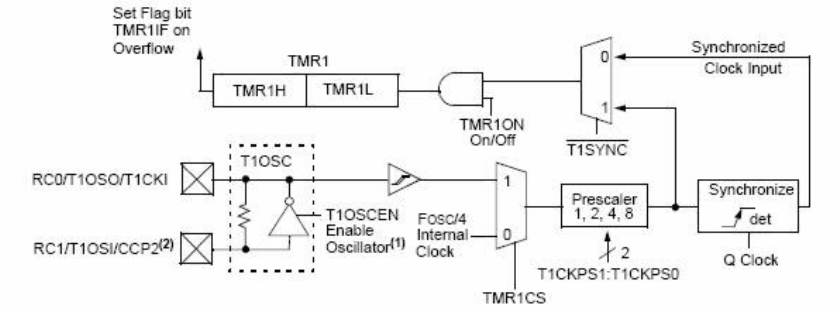
6.3 Timer1 Operation in Synchronized Counter Mode

Counter mode is selected by setting bit TMR1CS. In this mode, the timer increments on every rising edge of clock input on pin RC1/T1OSI/CCP2 when bit T1OSCEN is set, or on pin RC0/T1OSO/T1CKI when bit T1OSCEN is cleared.

If $\overline{\text{T1SYNC}}$ is cleared, then the external clock input is synchronized with internal phase clocks. The synchronization is done after the prescaler stage. The prescaler stage is an asynchronous ripple counter.

In this configuration, during Sleep mode, Timer1 will not increment even if the external clock is present since the synchronization circuit is shut-off. The prescaler, however, will continue to increment.

FIGURE 6-2: TIMER1 BLOCK DIAGRAM



Note 1: When the T1OSCEN bit is cleared, the inverter is turned off. This eliminates power drain.

6.4 Timer1 Operation in Asynchronous Counter Mode

If control bit $\overline{\text{T1SYNC}}$ (T1CON<2>) is set, the external clock input is not synchronized. The timer continues to increment asynchronous to the internal phase clocks. The timer will continue to run during Sleep and can generate an interrupt-on-overflow which will wake-up the processor. However, special precautions in software are needed to read/write the timer.

In Asynchronous Counter mode, Timer1 cannot be used as a time base for capture or compare operations.

6.4.1 READING AND WRITING TIMER1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers while the register is incrementing. This may produce an unpredictable value in the timer register.

Reading the 16-bit value requires some care. Examples 12-2 and 12-3 in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023) show how to read and write Timer1 when it is running in Asynchronous mode.

6.5 Timer1 Oscillator

A crystal oscillator circuit is built-in between pins T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit, T1OSCEN (T1CON<3>). The oscillator is a low-power oscillator, rated up to 200 kHz. It will continue to run during Sleep. It is primarily intended for use with a 32 kHz crystal. Table 6-1 shows the capacitor selection for the Timer1 oscillator.

The Timer1 oscillator is identical to the LP oscillator. The user must provide a software time delay to ensure proper oscillator start-up.

TABLE 6-1: CAPACITOR SELECTION FOR THE TIMER1 OSCILLATOR

Osc Type	Freq.	C1	C2
LP	32 kHz	33 pF	33 pF
	100 kHz	15 pF	15 pF
	200 kHz	15 pF	15 pF
These values are for design guidance only.			
Crystals Tested:			
32.768 kHz	Epson C-001R32.768K-A		± 20 PPM
100 kHz	Epson C-2 100.00 KC-P		± 20 PPM
200 kHz	STD XTL 200.000 kHz		± 20 PPM

Note 1: Higher capacitance increases the stability of oscillator but also increases the start-up time.

2: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

6.6 Resetting Timer1 Using a CCP Trigger Output

If the CCP1 or CCP2 module is configured in Compare mode to generate a "special event trigger" (CCP1M3:CCP1M0 = 1011), this signal will reset Timer1.

Note:	The special event triggers from the CCP1 and CCP2 modules will not set interrupt flag bit, TMR1IF (PIR1<0>).

Timer1 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. If Timer1 is running in Asynchronous Counter mode, this Reset operation may not work.

In the event that a write to Timer1 coincides with a special event trigger from CCP1 or CCP2, the write will take precedence.

In this mode of operation, the CCPRxH:CCPRxL register pair effectively becomes the period register for Timer1.

6.7 Resetting of Timer1 Register Pair (TMR1H, TMR1L)

TMR1H and TMR1L registers are not reset to 00h on a POR, or any other Reset, except by the CCP1 and CCP2 special event triggers.

T1CON register is reset to 00h on a Power-on Reset, or a Brown-out Reset, which shuts off the timer and leaves a 1:1 prescale. In all other Resets, the register is unaffected.

6.8 Timer1 Prescaler

The prescaler counter is cleared on writes to the TMR1H or TMR1L registers.

TABLE 6-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh,8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	--00 0000	--uu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.

Note 1: Bits PSPIE and PSPIF are reserved on the 28-pin devices; always maintain these bits clear.

8.0 CAPTURE/COMPARE/PWM MODULES

Each Capture/Compare/PWM (CCP) module contains a 16-bit register which can operate as a:

- 16-bit Capture register
- 16-bit Compare register
- PWM Master/Slave Duty Cycle register

Both the CCP1 and CCP2 modules are identical in operation, with the exception being the operation of the special event trigger. Table 8-1 and Table 8-2 show the resources and interactions of the CCP module(s). In the following sections, the operation of a CCP module is described with respect to CCP1. CCP2 operates the same as CCP1 except where noted.

CCP1 Module:

Capture/Compare/PWM Register 1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. The special event trigger is generated by a compare match and will reset Timer1.

CCP2 Module:

Capture/Compare/PWM Register 2 (CCPR2) is comprised of two 8-bit registers: CCPR2L (low byte) and CCPR2H (high byte). The CCP2CON register controls the operation of CCP2. The special event trigger is generated by a compare match and will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

Additional information on CCP modules is available in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023) and in application note *AN594, "Using the CCP Module(s)"* (DS00594).

TABLE 8-1: CCP MODE – TIMER RESOURCES REQUIRED

CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

TABLE 8-2: INTERACTION OF TWO CCP MODULES

CCPx Mode	CCPy Mode	Interaction
Capture	Capture	Same TMR1 time base
Capture	Compare	The compare should be configured for the special event trigger which clears TMR1
Compare	Compare	The compare(s) should be configured for the special event trigger which clears TMR1
PWM	PWM	The PWMs will have the same frequency and update rate (TMR2 interrupt)
PWM	Capture	None
PWM	Compare	None

8.3 PWM Mode (PWM)

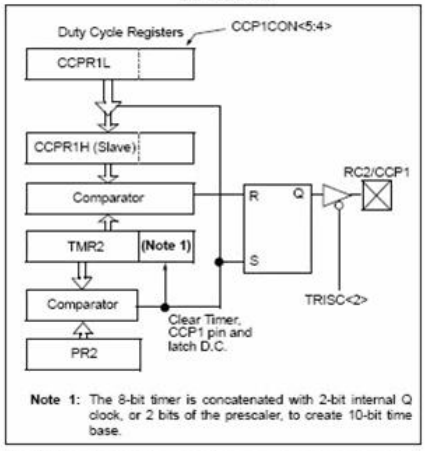
In Pulse Width Modulation mode, the CCPx pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output.

Note:	Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTC I/O data latch.

Figure 8-3 shows a simplified block diagram of the CCP module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 8.3.3 "Setup for PWM Operation"**.

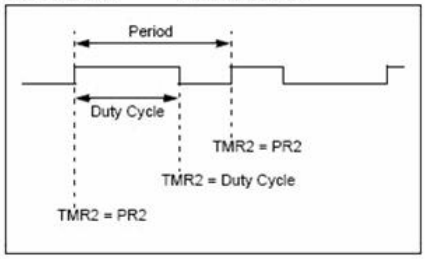
FIGURE 8-3: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: The 8-bit timer is concatenated with 2-bit internal Q clock, or 2 bits of the prescaler, to create 10-bit time base.

A PWM output (Figure 8-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 8-4: PWM OUTPUT



8.3.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from CCPR1L into CCPR1H

Note:	The Timer2 postscaler (see **Section 7.1 "Timer2 Prescaler and Postscaler"**) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

8.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSbs and the CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

$$\text{PWM Duty Cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read-only register.

The CCPR1H register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitch-free PWM operation.

When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the following formula.

EQUATION 8-1:

$$\text{Resolution} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

Note:	If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

8.3.3 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISC<2> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

TABLE 8-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 20 MHz

PWM Frequency	1.22 kHz	4.88 kHz	19.53 kHz	78.12kHz	156.3 kHz	208.3 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	0xFFh	0xFFh	0xFFh	0x3Fh	0x1Fh	0x17h
Maximum Resolution (bits)	10	10	10	8	7	5.5

TABLE 8-4: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE AND TIMER1

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh,8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
0Dh	PIR2	—	—	—	—	—	—	—	CCP2IF	---- ---0	---- ---0
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
8Dh	PIE2	—	—	—	—	—	—	—	CCP2IE	---- ---0	---- ---0
87h	TRISC	PORTC Data Direction Register								1111 1111	1111 1111
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	--00 0000	--uu uuuu
15h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	uuuu uuuu
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000
1Bh	CCPR2L	Capture/Compare/PWM Register 2 (LSB)								xxxx xxxx	uuuu uuuu
1Ch	CCPR2H	Capture/Compare/PWM Register 2 (MSB)								xxxx xxxx	uuuu uuuu
1Dh	CCP2CON	—	—	CCP2X	CCP2Y	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by Capture and Timer1.
Note 1: The PSP is not implemented on 28-pin devices; always maintain these bits clear.

TABLE 8-5: REGISTERS ASSOCIATED WITH PWM AND TIMER2

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh,8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
0Dh	PIR2	—	—	—	—	—	—	—	CCP2IF	---- ---0	---- ---0
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
8Dh	PIE2	—	—	—	—	—	—	—	CCP2IE	---- ---0	---- ---0
87h	TRISC	PORTC Data Direction Register								1111 1111	1111 1111
11h	TMR2	Timer2 Module's Register								0000 0000	0000 0000
92h	PR2	Timer2 Module's Period Register								1111 1111	1111 1111
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
15h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	uuuu uuuu
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000
1Bh	CCPR2L	Capture/Compare/PWM Register 2 (LSB)								xxxx xxxx	uuuu uuuu
1Ch	CCPR2H	Capture/Compare/PWM Register 2 (MSB)								xxxx xxxx	uuuu uuuu
1Dh	CCP2CON	—	—	CCP2X	CCP2Y	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by PWM and Timer2.
Note 1: Bits PSPIE and PSPIF are reserved on 28-pin devices; always maintain these bits clear.

10.0 ADDRESSABLE UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART)

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module is one of the two serial I/O modules. (USART is also known as a Serial Communications Interface or SCI.) The USART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers, or it can be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The USART can be configured in the following modes:

- Asynchronous (full-duplex)
- Synchronous – Master (half-duplex)
- Synchronous – Slave (half-duplex)

Bit SPEN (RCSTA<7>) and bits TRISC<7:6> have to be set in order to configure pins RC6/TX/CK and RC7/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter.

The USART module also has a multi-processor communication capability using 9-bit address detection.

REGISTER 10-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER (ADDRESS 98h)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC:** Clock Source Select bit
Asynchronous mode:
Don't care.
Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9**: 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN**: Transmit Enable bit
1 = Transmit enabled
0 = Transmit disabled
Note: SREN/CREN overrides TXEN in Sync mode.

bit 4 **SYNC**: USART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

bit 3 **Unimplemented:** Read as '0'

bit 2 **BRGH**: High Baud Rate Select bit
Asynchronous mode:
1 = High speed
0 = Low speed
Synchronous mode:
Unused in this mode.

bit 1 **TRMT**: Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 **TX9D:** 9th bit of Transmit Data, can be Parity bit

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 10-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER (ADDRESS 18h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN:** Serial Port Enable bit
1 = Serial port enabled (configures RC7/RX/DT and RC6/TX/CK pins as serial port pins)
0 = Serial port disabled

bit 6 **RX9**: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN**: Single Receive Enable bit
Asynchronous mode:
Don't care.
Synchronous mode – Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode – Slave:
Don't care.

bit 4 **CREN**: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive
0 = Disables continuous receive
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and load of the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit

bit 2 **FERR**: Framing Error bit
1 = Framing error (can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

bit 1 **OERR**: Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 **RX9D:** 9th bit of Received Data (can be parity bit but must be calculated by user firmware)

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

10.1 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USART. It is a dedicated 8-bit baud rate generator. The SPBRG register controls the period of a free running 8-bit timer. In Asynchronous mode, bit BRGH (TXSTA<2>) also controls the baud rate. In Synchronous mode, bit BRGH is ignored. Table 10-1 shows the formula for computation of the baud rate for different USART modes which only apply in Master mode (internal clock).

Given the desired baud rate and Fosc, the nearest integer value for the SPBRG register can be calculated using the formula in Table 10-1. From this, the error in baud rate can be determined.

It may be advantageous to use the high baud rate (BRGH = 1) even for slower baud clocks. This is because the Fosc/(16 (X + 1)) equation can reduce the baud rate error in some cases.

Writing a new value to the SPBRG register causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

10.1.1 SAMPLING

The data on the RC7/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

TABLE 10-1: BAUD RATE FORMULA

SYNC	BRGH = 0 (Low Speed)	BRGH = 1 (High Speed)
0	(Asynchronous) Baud Rate = Fosc/(64 (X + 1))	Baud Rate = Fosc/(16 (X + 1))
1	(Synchronous) Baud Rate = Fosc/(4 (X + 1))	N/A

Legend: X = value in SPBRG (0 to 255)

TABLE 10-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used by the BRG.

TABLE 10-3: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 0)

BAUD RATE (K)	Fosc = 20 MHz			Fosc = 16 MHz			Fosc = 10 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	-	-	-	-	-	-	-	-	-
1.2	1.221	1.75	255	1.202	0.17	207	1.202	0.17	129
2.4	2.404	0.17	129	2.404	0.17	103	2.404	0.17	64
9.6	9.766	1.73	31	9.615	0.16	25	9.766	1.73	15
19.2	19.531	1.72	15	19.231	0.16	12	19.531	1.72	7
28.8	31.250	8.51	9	27.778	3.55	8	31.250	8.51	4
33.6	34.722	3.34	8	35.714	6.29	6	31.250	6.99	4
57.6	62.500	8.51	4	62.500	8.51	3	52.083	9.58	2
HIGH	1.221	-	255	0.977	-	255	0.610	-	255
LOW	312.500	-	0	250.000	-	0	156.250	-	0

BAUD RATE (K)	Fosc = 4 MHz			Fosc = 3.6864 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	0.300	0	207	0.3	0	191
1.2	1.202	0.17	51	1.2	0	47
2.4	2.404	0.17	25	2.4	0	23
9.6	8.929	6.99	6	9.6	0	5
19.2	20.833	8.51	2	19.2	0	2
28.8	31.250	8.51	1	28.8	0	1
33.6	-	-	-	-	-	-
57.6	62.500	8.51	0	57.6	0	0
HIGH	0.244	-	255	0.225	-	255
LOW	62.500	-	0	57.6	-	0

TABLE 10-4: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 1)

BAUD RATE (K)	Fosc = 20 MHz			Fosc = 16 MHz			Fosc = 10 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	-	-	-	-	-	-	-	-	-
1.2	-	-	-	-	-	-	-	-	-
2.4	-	-	-	-	-	-	2.441	1.71	255
9.6	9.615	0.16	129	9.615	0.16	103	9.615	0.16	64
19.2	19.231	0.16	64	19.231	0.16	51	19.531	1.72	31
28.8	29.070	0.94	42	29.412	2.13	33	28.409	1.36	21
33.6	33.784	0.55	36	33.333	0.79	29	32.895	2.10	18
57.6	59.524	3.34	20	58.824	2.13	16	56.818	1.36	10
HIGH	4.883	-	255	3.906	-	255	2.441	-	255
LOW	1250.000	-	0	1000.000		0	625.000	-	0

BAUD RATE (K)	Fosc = 4 MHz			Fosc = 3.6864 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	-	-	-	-	-	-
1.2	1.202	0.17	207	1.2	0	191
2.4	2.404	0.17	103	2.4	0	95
9.6	9.615	0.16	25	9.6	0	23
19.2	19.231	0.16	12	19.2	0	11
28.8	27.798	3.55	8	28.8	0	7
33.6	35.714	6.29	6	32.9	2.04	6
57.6	62.500	8.51	3	57.6	0	3
HIGH	0.977	-	255	0.9	-	255
LOW	250.000	-	0	230.4	-	0

10.2 USART Asynchronous Mode

In this mode, the USART uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip, dedicated, 8-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator. The USART transmits and receives the LSb first. The transmitter and receiver are functionally independent but use the same data format and baud rate. The baud rate generator produces a clock, either x16 or x64 of the bit shift rate, depending on bit BRGH (TXSTA<2>). Parity is not supported by the hardware but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during Sleep.

Asynchronous mode is selected by clearing bit SYNC (TXSTA<4>).

The USART Asynchronous module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

10.2.1 USART ASYNCHRONOUS TRANSMITTER

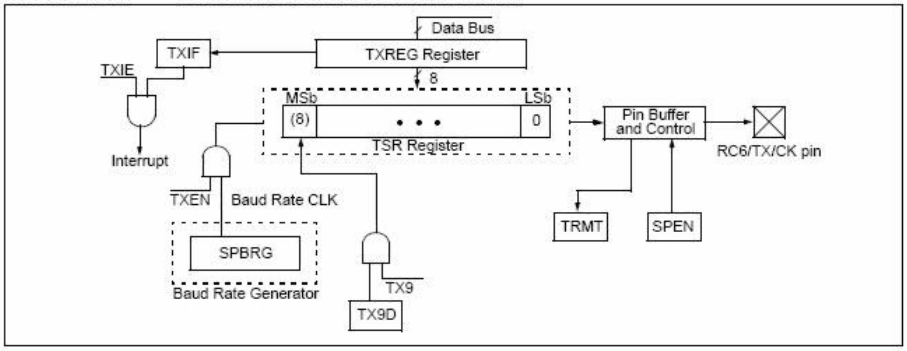
The USART transmitter block diagram is shown in Figure 10-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the Read/Write Transmit Buffer, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the Stop bit has been transmitted from the previous load. As soon as the Stop bit is transmitted, the TSR is loaded with new data from the TXREG register (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCY), the TXREG register is empty and flag bit, TXIF (PIR1<4>), is set. This interrupt can be enabled/disabled by setting/clearing enable bit, TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. Status bit TRMT is a read-only bit which is set when the TSR register is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register is empty.

Note 1:	The TSR register is not mapped in data memory so it is not available to the user.
2:	Flag bit TXIF is set when enable bit TXEN is set. TXIF is cleared by loading TXREG.

Transmission is enabled by setting enable bit, TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data and the Baud Rate Generator (BRG) has produced a shift clock (Figure 10-2). The transmission can also be started by first loading the TXREG register and then setting enable bit TXEN. Normally, when transmission is first started, the TSR register is empty. At that point, transfer to the TXREG register will result in an immediate transfer to TSR, resulting in an empty TXREG. A back-to-back transfer is thus possible (Figure 10-3). Clearing enable bit TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. As a result, the RC6/TX/CK pin will revert to high-impedance.

In order to select 9-bit transmission, transmit bit TX9 (TXSTA<6>) should be set and the ninth bit should be written to TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG register can result in an immediate transfer of the data to the TSR register (if the TSR is empty). In such a case, an incorrect ninth data bit may be loaded in the TSR register.

FIGURE 10-1: USART TRANSMIT BLOCK DIAGRAM



When setting up an Asynchronous Transmission, follow these steps:

1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (**Section 10.1 "USART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set transmit bit TX9.
5. Enable the transmission by setting bit TXEN, which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).
8. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

FIGURE 10-2: ASYNCHRONOUS MASTER TRANSMISSION

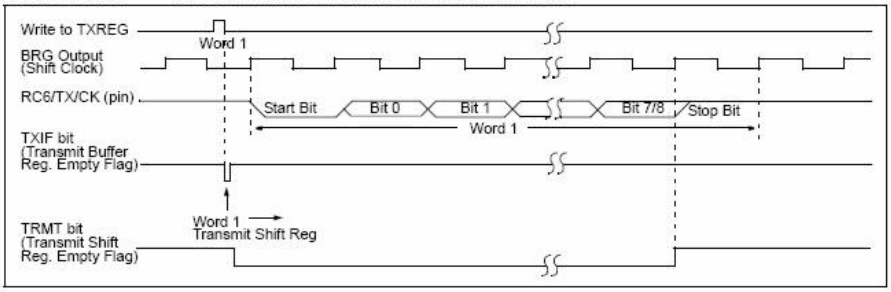
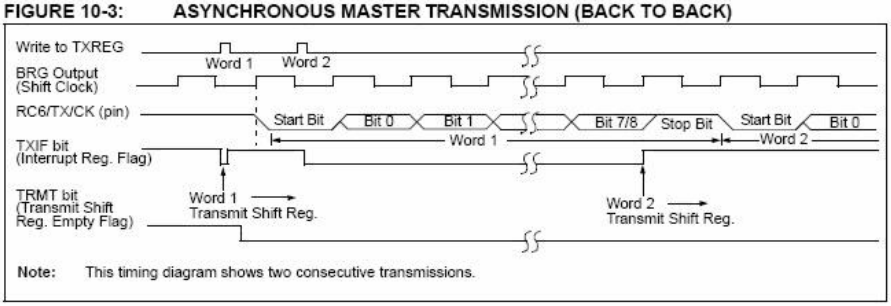


FIGURE 10-3: ASYNCHRONOUS MASTER TRANSMISSION (BACK TO BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 10-5: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	R0IF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
19h	TXREG	USART Transmit Register								0000 0000	0000 0000
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.
Note 1: Bits PSPIE and PSPIF are reserved on 28-pin devices; always maintain these bits clear.

10.2.2 USART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 10-4. The data is received on the RC7/RX/DT pin and drives the data recovery block. The data recovery block is actually a high-speed shifter, operating at x16 times the baud rate; whereas the main receive serial shifter operates at the bit rate or at Fosc.

Once Asynchronous mode is selected, reception is enabled by setting bit CREN (RCSTA<4>).

The heart of the receiver is the Receive (Serial) Shift Register (RSR). After sampling the Stop bit, the received data in the RSR is transferred to the RCREG register (if it is empty). If the transfer is complete, flag bit, RCIF (PIR1<5>), is set. The actual interrupt can be enabled/disabled by setting/clearing enable bit, RCIE (PIE1<5>). Flag bit RCIF is a read-only bit which is cleared by the hardware. It is cleared when the RCREG register has been read and is empty. The RCREG is a double-buffered register (i.e., it is a two-deep FIFO). It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte to begin shifting to the RSR register. On the detection of the Stop bit of the third byte, if the RCREG register is still full, the Overrun Error bit, OERR (RCSTA<1>), will be set. The word in the RSR will be lost. The RCREG register can be read twice to retrieve the two bytes in the FIFO. Overrun bit OERR has to be cleared in software. This is done by resetting the receive logic (CREN is cleared and then set). If bit OERR is set, transfers from the RSR register to the RCREG register are inhibited and no further data will be received. It is, therefore, essential to clear error bit OERR if it is set. Framing error bit, FERR (RCSTA<2>), is set if a Stop bit is detected as clear. Bit FERR and the 9th receive bit are buffered the same way as the receive data. Reading the RCREG will load bits RX9D and FERR with new values, therefore, it is essential for the user to read the RCSTA register before reading the RCREG register in order not to lose the old FERR and RX9D information.

FIGURE 10-4: USART RECEIVE BLOCK DIAGRAM

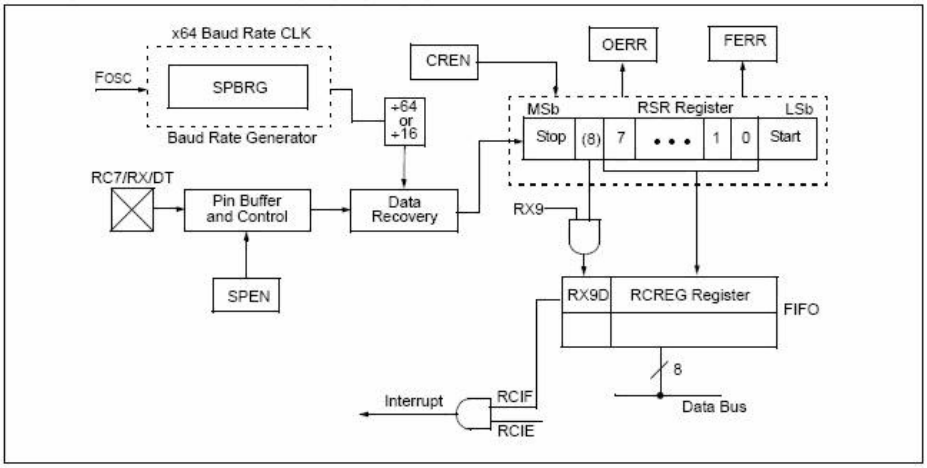
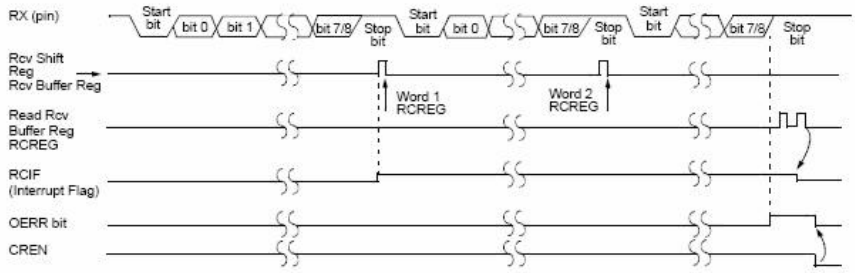


FIGURE 10-5: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX input. The RCREG (Receive Buffer) is read after the third word, causing the OERR (Overrun Error) bit to be set.

When setting up an Asynchronous Reception, follow these steps:

1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (**Section 10.1 "USART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit RCIE.
4. If 9-bit reception is desired, then set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE is set.
7. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit CREN.
10. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

TABLE 10-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	R0IF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
1Ah	RCREG	USART Receive Register								0000 0000	0000 0000
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

Note 1: Bits PSPIE and PSPIF are reserved on 28-pin devices; always maintain these bits clear.

Apéndice E Publicaciones Nacionales e Internacionales

Artículo publicado en el Primer Congreso Nacional de Mecatrónica y Tecnologias Inteligentes CONAMTI 2006 con sede en el Instituto Tecnológico Superior de Huichapan, Hidalgo.

AHIETI 1 (Máquina Generadora de Circuitos PCB)

Christian David Castillo Martínez y Eduardo Sánchez Abad

Universidad Autónoma del Estado de Hidalgo

Instituto de Ciencias Básicas e Ingeniería

Ciudad Universitaria Carretera Pachuca - Tulancingo Km 4.5 C.P. 42184

ccastillo@uaeh.edu.mx, se107575@uaeh.reduaeh.mx

Resumen

Este mecanismo realiza el descobrizado de placas fenólicas y la perforación de los pines correspondientes a los dispositivos de un diagrama electrónico diseñado previamente mediante un software, dejar la placa lista para soldar los componentes electrónicos en las perforaciones hechas por el AHIETI v 2.1.1.

Para realizar la impresión de un PCB a una placa fenólica con el AHIETI v 2.1.1, es necesario crearlo y exportarlo posteriormente a otro software, este a su vez envía los datos necesarios al sistema de control del AHIETI v 2.1.1

El descobrizado lo efectúa un mini-drill con una fresa de carburo de tungsteno, las perforaciones se realizan con brocas normales.

Este se encuentra adaptado en un deslizador principal que a su vez esta ensamblado a

un par de deslizadores. El software para dibujar los circuitos es muy sencillo, sin embargo nos permite diseñar circuitos complejos, sin usar herramientas mas complicadas como los basados en CAD.

El software de control es una aplicación que concuerda con la dinámica del software de diseño, muy sencillo pero, cumple con las expectativas buscadas para este trabajo.

1. Introducción

La tecnología evoluciona rápidamente, inclusive mas rápido de lo esperado por los expertos, se ha podido observar sobre todo en la electrónica de consumo, como los semiconductores se hacen más complejos y económicos. Por esta razón

los ingenieros deben tratar de seguir este avance, actualizando sus conocimientos, investigando nuevas tecnologías, desarrollando nuevas técnicas.

Este trabajo se trata de eso exactamente, investigar una técnica que si bien no es nueva si es importarte estudiarla. En las aulas se estudian los avances de los componentes y de las tecnologías tanto de la electrónica, la robótica y las telecomunicaciones, pero se dejan de lado las técnicas fundamentales para el desarrollo de las mismas.

Por ello se abarca el desarrollo de una técnica para elaborar los circuitos impresos de prototipos, que si bien no es nueva, se ha dejado de lado. Los métodos usados actualmente para el diseño y la elaboración de los PCB por los estudiantes de electrónica no han cambiado desde hace mucho tiempo, se sigue ocupando el marcador indeleble o los tipos para marcar las pistas, esto ya no se puede permitir, porque evita que el ingeniero desarrolle la tecnología que esta época demanda.

Es la obligación de los actuales profesionistas el improvisar técnicas más efectivas y a precios que los estudiantes puedan cubrir. El prototipo propuesto es un robot que pretende actualizar la vieja escuela, de una manera económica y que además elimina los riesgos que representan los químicos para el estudiante y para el medio ambiente.

La solución que proponemos es atacar el cobre con un método mecánico (descobrizando la placa fenólica, con una fresa), esto ayudado por un dispositivo autómata, basado en elementos mecánicos, electrónicos, electromecánicos y de un software especialista instalado en una PC.

Demostramos en base a la experimentación que este prototipo es viable y eficaz, además que puede ser perfeccionado en las etapas de las que se compone, esto tomando las bases que se explican en este trabajo.

2. Proyectos similares

En el principio básico del robot cartesiano se manejan numerosos ejemplos desarrollados formalmente como informalmente, es decir, existen actualmente una gran diversidad del maquinas y dispositivos capaces de realizar movimientos traslacionales sobre los ejes cartesianos X, Y, y Z; sin embargo también es importante señalar importantes diferencias, en cuanto al principio mecánico de su funcionamiento, el sistema de control (en algunos de estos inexistente), y por supuesto la interfase para el operador humano.

Tomando en cuenta estos aspectos, nos dimos a la tarea de investigar el entorno tecnológico y teórico del AHIETI encontrando algunos prototipos con funcionamiento parecido o al menos principio teórico similar al propuesto en el presente trabajo.

2.1. Diseño y fabricación de circuitos impresos mediante una PC.

Este es el nombre del trabajo realizado en la Facultad de Ingeniería Eléctrica en la Universidad Michoacana de San Nicolás de Hidalgo.[1] Trabajo en el cual usa el editor de circuitos impresos llamado Eagle[2], el cual consiste en un software capaz de diseñar los circuitos en una vista en la cual se ve la forma en la que el circuito quedará impreso en una placa fenolica. Otro software utilizado en este trabajo es el KCam, el permite a través del puerto paralelo dar instrucciones a la herramienta para la manufactura, se mueve en tres dimensiones para realizar la tarea indicada (perforar, cortar y tallar el cobre).

En este trabajo se usan motores de pasos bipolares, los que son controlados desde una PC por medio de una tarjeta de control a través de un puerto paralelo. El driver usado para la tarjeta de control es el L297-298

La herramienta utilizada para el corte y perforación, se le denomina Herramienta de Corte que se muestra en la figura1, que es un taladro Dremel, igual al usado en el AHIETI.

Figura 1: Herramienta de corte

[1] Víctor Hugo Chávez Urbina y Carlos Manuel Sánchez González en Noviembre del 2004.
[2] Para Windows 95/NT Versión 4.1

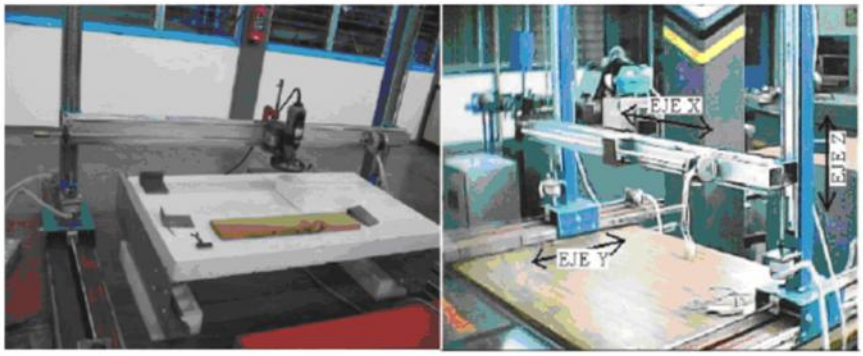


Figura 2: Estructura del mecanismo

2.2. Mesa móvil

Otro de los proyectos similares es el que a continuación presentamos, no tenemos el nombre de dicho trabajo, tampoco de quien lo fabrico. Pero lo llamaremos **mesa móvil.**

Consiste en un mecanismo similar al anterior, es una estructura igual de robusta, ya que utiliza perfiles de acero como base de todo el sistema, flechas de soporte como el usado en el AHIETI, que mas adelante detallaremos, y tornillos sin fin, como se muestra en la figura 3. Las piezas en color verde contienen unos bujes donde se incrustan las flechas de soporte y así se desliza el sistema, se puede notar que en medio de las dos flechas hay un tornillo sin fin con un stepper motor para mover la base del taladro.

También se nota que la mesa donde se colocan los objetos a trabajar se mueve de izquierda a derecha o Vertical u horizontal (depende del punto de vista). Para la base del taladro se aplico el mismo sistema, dos flechas con un tornillo sin fin en medio de ellas, con su respectivo stepper motor.

Figura 3: Mesa Móvil

2.3. Flexicam series

Otro de los proyectos similares al Ahieti son las fresadoras Flexicam Series[3].

Las fresadoras CNC VL, STEALTH, PRO se muestran en la figura 4 son ideales para el procesamiento y corte de PVC, aluminio, metales no ferrosos, madera, compuestos sintéticos, otros materiales y que requieran una sólida estructura.

La base de la fresadora está diseñada para soportar un trabajo intensivo y libre de vibraciones. En contraste con la fresadora Eco, que utilíza motores paso a paso, la fresadora Stealth está equipada con motores servo AC. Estructura totalmente en acero, mecanizada y libre de tensiones.

El sistema Stealth se conecta a la estación de trabajo, bajo Windows 95/98-/NT/W2000, mediante una conexión en red. El sistema se maneja mediante un cómodo controlador manual.

Figura 4: Flexicam Series

El modelo cobra realiza el grabado y mecanizado de altas prestaciones en 2D y 3D, figura 5. [2]

Figura 5: Flexicam Cobra

[3]Fabricadas por Veroe control S.L. en Madrid.

3. AHIETI v 1.0.0

La realización mecánica del AHIETI v 1.0.0 (maquina generadora de PCBs) esta basada básicamente en el mecanismo de las maquinas de escribir eléctricas y algunas impresoras, por el movimiento horizontal del inyector de tinta, y en una mejora del diseño ya antes realizado por los autores de este trabajo, las partes de que consta son totalmente diseñadas y elaboradas por nosotros. Para ello realizamos dos diseños mecánicos, en ambos casos tuvimos que aprender el uso correcto de maquinaria como: Torno, Fresadora, Taladro de Banco, y de otras herramientas.

Para poder llegar al diseño actual del AHIETI v 2.0.0, primero se fabricó un mecanismo similar AHIETI v 1.0.0. constaba de una base cuadrada de aluminio, dos Flechas de Soporte de Deslizamiento para Deslizadores (F.S.D.D) de acero inoxidable (en color azul de la figura 6), y de dos tornillos sin fin (en color amarillo de la figura 6). Así como de otras piezas que explicaremos a continuación.

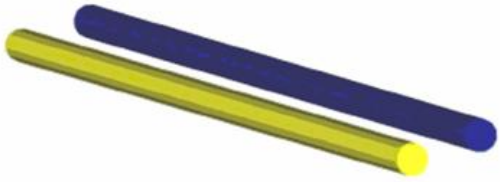


Figura 6: Flecha de Soporte de Deslisamiento para Deslizadores (F.S.D.D) y Tornillo sin Fin

3.1. Soportes

Otra pieza que contaba el AHIETI v 1.0.0, son los soportes, hechos de acero, en las esquinas de la base de aluminio en los cuales se introducían la F.S.D.D y el tornillo sin fin, estos pequeños soportes se fabricaron con acero, y solo tenían un barreno a la mitad de la pieza. figura7.

Figura 7: Soportes AHIETI v1.0.0

3.2. Deslizadores

3.2.1. Deslizador izquierdo y derecho

Para poder unir las F.S.D.D y los espárragos del eje *X* con el eje *Y*, se fabricaron los deslizadores en acero, dos para la base (izquierdo y derecho), ver figura8. ambos deslizadores constan de tres barrenos que atraviesan al mismo, dos en la parte frontal en los cuales se introducía la F.S.D.D y el tornillo sin fin que efectúan el movimiento en el eje *X*, y otro barreno lateral, en caso del deslizador derecho introduce una F.S.D.D.

El deslizador izquierdo, a diferencia del derecho es que en el barreno lateral consta de cuerda para que pudiera introducirse el tronillo sin fin.

Figura 8: Deslizador AHIETI v 1.0.0

3.2.2. Deslizador principal.

El deslizador principal consta básicamente de dos piezas, Carro y Soporte del taladro, las cuales están fabricadas casi en su totalidad de aluminio y una pequeña pieza de Acero.

Cola de milano

La cola de milano es un corte que se realiza en madera, acero, aluminio, bronce, etc. Todo depende de la aplicación, se le llama así por el milano, el milano es un ave de rapiña diurna, su cola tiene forma de trapecio como se aprecia en la figura 9.

Figura 9: Ejemplo del ave milano

Para poder ensamblar una pieza con este corte es necesario realizar dos partes (hembra y macho) a 45° ó 60° dependiendo del diseño y del diseñador. La figura 10 muestra las diferentes formas de cortes con cola de milano.

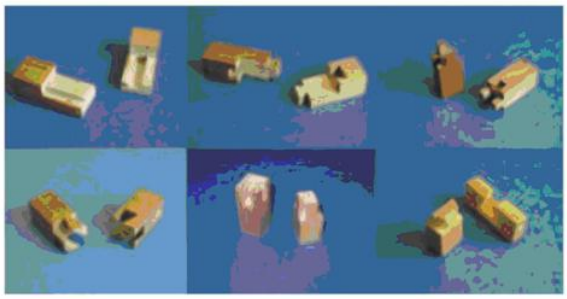


Figura 10: Diferentes cortes de Cola de Milano

Carro

Como ya se hizo mención, la cola de milano necesita de dos cortes y para nuestro caso a 45°, para poder ensamblarse, al carro se le hizo el corte hembra, que también pudo haber sido trabajado con el corte macho, para nuestro propósito cualquier forma de corte no afecta, el funcionamiento es el mismo.

El carro se desplaza en la F.S.D.D y en el tornillo sin fin, sobre el eje *Y*.

En la figura 11 se muestra el carro, podemos ver los barrenos que atraviesan al carro para poder trasladarse con la ayuda del tornillo sin fin, el cual se introduce en el barreno a la derecha de $\frac{1}{2}$ pulgada, en el barreno izquierdo de 12 mm se coloca la F.S.D.D, así como en los casos anteriores. El barreno que se encuentra en la cara superior del carro es para colocar el tornillo para el soporte del taladro.

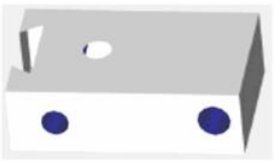


Figura 11: Carro AHIETI v 1.0.0

Soporte del taladro

El Soporte del taladro trabaja en el eje *Z* con ayuda del tornillo que se encuentra perpendicular al carro y paralelo al soporte del Taladro, para poder sujetar el taladro al soporte del taladro fue necesario realizarle un medio barreno ala parte frontal del soporte. figura 12

Figura 12: Soporte del Taladro AHIETI v 1.0.0

Ensamblando el carro, el soporte del taladro y el tornillo se obtiene el deslizador principal, figura 13

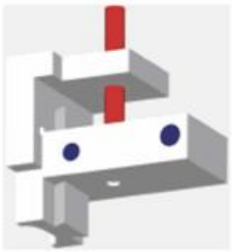


Figura 13: Deslizador Principal AHIETI v 1.0.0

3.3. Ensamble AHIETI v 1.0.0

Para poder efectuar el movimiento de todo el sistema se pensó en incorporar en los tornillos sin fin y a los stepper motors unos engranes, excepto el tornillo del deslizador principal, a este se le incorporaría una rueda dentada (sprocket) obviamente el stepper motor también llevaría otra sprocket y con ellas su respectiva cadena, pero todo esto no se realizó físicamente.

Finalmente uniendo las piezas en el lugar correspondiente obtenemos al AHIETI v 1.0.0, figura 14, pero el funcionamiento no era el correcto, ya que cuando el tornillo sin fin giraba movía al deslizador y el deslizador de la F.S.D.D dejaba de desplazarse impidiendo el avance de todo el sistema, esto es que dejaba de estar en escuadra, esta fue la primer razón por la cual decidimos cambiar de diseño.

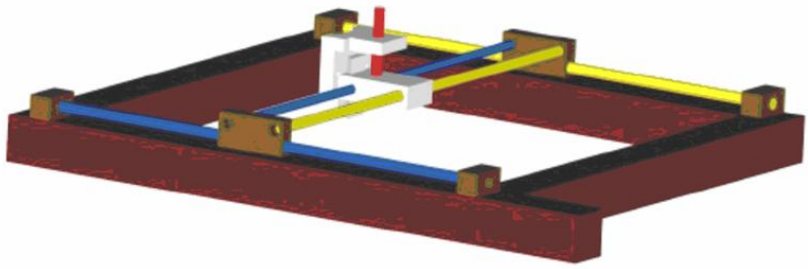


Figura 14: AHIETI v 1.0.0

También se observo que el movimiento en ambos casos (eje X y eje Y) era demasiado lento ya que dependía del giro del esparrago y de los stepper motors, para dar solución a este problema se considero en cambiar el tornillo sin fin de cuerda estándar por un tornillo de cuerda cuadrada. Pero por su alto costo nunca se incorporo a la estructura.

4. AHIETI v 2.0.0

Antes de continuar con el rediseño del AHIETI v 1.0.0, es importante hacer mención de cómo se van actualizando las versiones, es necesario saber que cada digito va a ir cambiando conforme se concluya el proyecto, a continuación se muestra la descripción:

Así entonces ahora sabemos que nuestro primer diseño corresponde al AHIETI v 1.0.0, hasta este momento no se ha trabajado en el software y en la electrónica.

Finalmente se llegó a una decisión, rediseñar el AHIETI v 1.0.0, ya que no era funcional. Para la elaboración del AHIETI v 2.0.0 se pensó en utilizar dos F.S.D.D para evitar el problema del AHIETI v 1.0.0, se continuaban usando los stepper motors. Las demás piezas a excepción del Deslizador principal se fabricarían nueva mente.

4.1. Deslizadores

4.1.1. Deslizador principal

Casi todas las piezas realizadas en el anterior diseño ya no resultaban de utilidad, a excepción del deslizador principal, el carro se modifico ligeramente para que funcionara con el nuevo diseño, tal modificación consistió en quitarle la cuerda y desplazar el barreno aproximadamente unos 10 mm. hacia el centro de la pieza.

Los dos barrenos que se observan en la figura 15 son de 12.7 mm. en los cuales se introducen las flechas de soporte de deslizamiento para el deslizador principal (F.S.D.D.P) de $\frac{1}{2}$ pulgada, donde trabaja el deslizador principal en eje Y. Al soporte del taladro no se le hizo modificación alguna, se le adaptó el cincho para sujetar el taladro con el soporte, se utilizo un cinturón en forma de U de dos pulgadas de diámetro y finalmente el deslizador principal quedó como se apreciaen la figura 15.

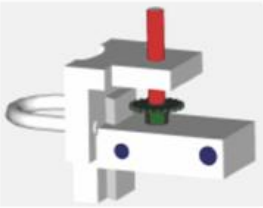


Figura 15: Deslizador Principal AHIETI v 2.0.0

Se puede observar que ahora el deslizador principal ya cuenta con su sprocket correspondiente, (color verde) y que efectúa el movimiento en el eje Z en el soporte del taladro.

4.1.2. Deslizador izquierdo

Los deslizadores también son de aluminio se fabricarón dos, uno derecho y el otro izquierdo y estos sustituyen a los anteriores, trabajan en el eje Y, ya que el sistema de F.S.D.D así lo requiere, como muestra la figura 16, podemos ver que lleva cuatro barrenos, los de color rojo atraviesan todo el deslizador con diámetro de 11 mm, ya que en el se incorporan dos pequeñas flechas de soporte de deslizamiento para deslizadores de aproximadamente 10 mm. de diámetro, los barrenos en color azul tienen un diámetro de $\frac{1}{2}$ pulgada (aproximadamente 12.7 milímetros) y una profundidad de aproximadamente 15 mm, en donde se incorporan dos F.S.D.D de $\frac{1}{2}$ pulgada. La ranura en color amarillo tiene una abertura de 10 mm. Las dimensiones de ambos deslizadores son 100x70x35 mm.

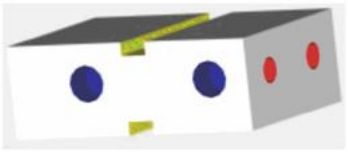


Figura 16: Deslizador Izquierdo AHIETI v 2.0.0

4.1.3. Deslizador derecho con sprocket

Este deslizador es idéntico al Deslizador Izquierdo con la única diferencia es que cuenta con un sprocket a un costado de él para que la cadena que jala al deslizador principal de vuelta en el sprocket y pueda completar el movimiento correctamente mostrado en la figura 17. El sprocket esta dentro de una base de aluminio con forma de U.

Figura 17: Deslizador Derecho Con Sprocket AHIETI v 2.0.0

4.2. Soportes

4.2.1. Soporte para flechas de soporte de deslizamiento para deslizadores con placa para stepper motor (S.F.S.D.D.P.S.M)

El **S.F.S.D.D.P.S.M** se muestra en la figura 18, es otra pieza importante para el AHIETI v 2.0.0, ya que en ellos el motor es sujetado con una placa, cabe mencionar que consta de dos partes, la placa para el motor y el soporte para las F.S.D.D, ambas piezas son de acero, los tres barrenos en los soportes tienen un diámetro de 10 mm., la ranura es de 10 mm. de ancho por 17 mm. de alto, donde pasa la cadena. Las dos piezas (soporte y placa) se soldaron para que se unieran y en cuanto se quiera desarmar esto sea mas facil.

Figura 18: SoporteSoporte Para Flechas De Soporte De Deslizamiento Para Deslizadores Con Placa Para Stepper Motor AHIETI v 2.0.0

4.2.2. Soportes para flechas de soporte para deslizadores con sprocket (S.F.S.D.D.S)

En la figura 19 podemos observar que en la parte del centro se encuentra el sprocket el cual gira si el stepper motor que se encuentra al otro extremo lo hace, es el mismo funcionamiento del deslizador derecho con sprocket, ya que sin esta sería imposible el desplazamiento del sistema, el tamaño y la posición de los barrenos es la misma al del **S.F.S.D.D.P.S.M.**

Figura 19: Soportes Para Flechas De Soporte Para Deslizadores Con Sprocket AHIETI v 2.0.0

Estos soportes ayudan a realizar el movimiento de los deslizadores izquierdo y derecho con sprocket, en el eje X, con ayuda cadenas.

4.3. Cadenas y sprockets

El paso de las cadenas y de los sprockets es 15, el número de dientes del sprocket es de 21, mostrado en la figura 20, es obvio que para obtener el largo de las cadenas para nuestro propósito fue necesario unirlas con eslabones llamados candados.

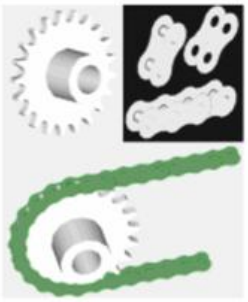


Figura 20: Cadenas y Sprockets AHIETI v 2.0.0

4.4. Fresas para el descobrizado

Las fresas para realizar el descobrizado son de carburo de tungsteno como el mostrado en la figura , estas fresas son utilizadas por los odontólogos, para sus intervenciones, existen diversos modelos, son fabricadas con diferentes materiales, el modelo utilizado para los propósitos del AHIETI v 2.0.0 es el FG-56 de MEISINGER.

Fig.	Shank	ISO-No.	5	5	5	5	5	5	5	5
		L mm	3,6	3,8	4,1	4,1	4,5	4,5	4,9	5,4
		US-No.		56	57	58	59	60	61	
HM 21	HP	500 **104** 107 006	008	009	010	012	014	016	018	023
	RA	500 **204** 107 006	008		010	012				
	FG	500 **314** 107 006	008	009	010	012	014			
	FG XL	500 **316** 107 006			010	012				

Figura 21: Fresas de carburo tungsteno

4.5. Ensamble AHIETI v 2.0.0

Finalmente colocando las piezas mecánicas antes mencionadas en su lugar correspondiente obtenemos el AHIETI v 2.0.0, así como se muestra en la figura.22

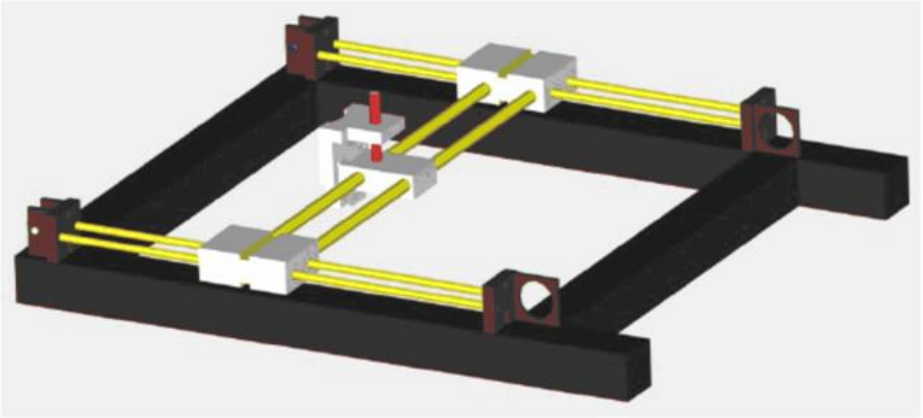


Figura 22: AHIETI v 2.0.0

5. Electrónica

Así como es de suma importancia el diseño mecánico en un robot o en una máquina CNC lo es el electrónico, ya que es ultimo es del control, potencia, etc. Como en todos los sistemas, incluso en el cuerpo humano, un órgano, en menor o mayor proporción, necesita de otro para poder realizar una tarea especifica, en este caso el sistema mecánico necesita de la electrónica y viceversa.

5.1. Stepper motors

Los stepper motors son de gran ayuda para mecanismos en donde se requieren movimientos muy precisos.

Estos motores poseen la habilidad de poder quedar enclavados en una posición o bien libres, si una o más de sus bobinas está energizada, el motor estará enclavado en la posición correspondiente y por el contrario quedará libre si no circula corriente por ninguna de sus bobinas.

Generalmente los stepper motors están constituidos por un rotor sobre el que van aplicados distintos imanes permanentes, y por un cierto número de bobinas excitadoras bobinadas en su estator. Las bobinas son parte del estator y el rotor es un imán permanente. Toda la conmutación (o excitación de las bobinas) deber ser externamente manejada por un controlador.

Los stepper motors que usa el AHIETI v 2.1.0 son de la compañía ASTROSYN, modelo 23LM-C327-11V, hechos en Tailandia, consumen aproximadamente 1.3 amp, cuando se energiza una bobina, el total de la corriente que consume cuando se aplica la secuencia normal al máximo de su velocidad es de 1.06 amp, y con

se multiplican por el numero de motores que se quiera manejar, esto limita el trabajo del puerto paralelo a solo dos motores por puerto usando los 8 pines de entrada-salida.

Los microcontroladores tienen mas pines de entrada-salida por lo que tienen la posibilidad de manejar mas motores y esto aunado a la comunicación mediante el USART (Universal Synchronous Asynchronous Receiver Transmitter, transmisor receptor sincrónico asincrónico universal) de los microcontroladores podemos manejar un mayor numero de motores con una sola PC, inclusive usando el puerto serie, de solo dos pines, para comunicar la PC con la etapa de potencia de los motores.

Al estudiar esta posibilidad se diseño una primera versión de la etapa de potencia con la cual se pueden controlar los cuatro motores con el puerto serie y un solo microcontrolador.

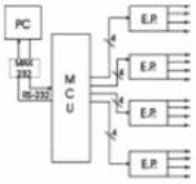


Figura 27: Primer Diseño

En este primer diseño el puerto serie de la PC se conecta al USART del microcontrolador, esto posibilita usar dos pines para la comunicación entre MCU y la computadora, y dieciséis pines para manejar los motores, esto resulta en veinte pines.

Pero el problema en esta configuración es que el código se vuelve muy grande y el espacio en memoria es reducido, además que se toman pines del microcontrolador que poseen funciones que podrían ser aprovechadas de una mejor manera. Por esta razón se considero un segundo diseño que nos permite mejores prestaciones.

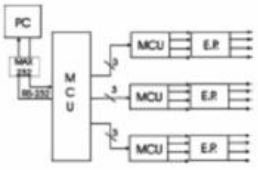


Figura 28: Segundo Diseño

Con este diseño se obtiene una mejora tanto en desempeño como el escalabilidad ya que se descentraliza el trabajo en el microcontrolador principal, delegando

funciones a los microcontroladores secundarios, además de un ahorro importante de pines, solo dos de comunicación y nueve de control, un total de once pines, se dejan disponibles otros pines de funciones especiales de los microcontroladores, permitiendo el sensado y una toma de decisión independiente del procesador principal.

Los microcontroladores que se utilizan en este proyecto son el PIC 16F877A y 16F876A de la empresa Microchip, estos poseen un CPU de 8 bits, memoria de programa de 14.3Kbytes, SRAM de datos de 368 Bytes y una EEPROM de 256 Bytes, 22 y 33 pines de entrada salida programables respectivamente, 5 y 8 canales de conversión analógica a digital, dos moduladores de ancho de pulso, comunicación SPI, $I2C$ y USART, 2 timer de 8 bits y uno de 16 bits y dos comparadores.

5.6.1. El microcontrolador principal

Este será el encargado de recibir la información que envía la computadora, interpretarlo, activar los controles de los microcontroladores secundarios y enviar la posición actual de regreso a la computadora.

El microcontrolador se posiciona en 0,0 y espera para la instrucción, la computadora envía la primera coordenada de intercepción, el microcontrolador recibe la información, baja el minidrill y avanza hasta la posición indicada, envía la posición actual y sube el minidrill, espera la siguiente instrucción, la recibe y avanza hasta el final de la intercepción, envía la posición y baja el minidrill, repite la operación hasta recorrer la superficie total de la placa en X, después activa un paso en Y repite la operación anterior, así hasta cubrir la superficie de la placa en Y.

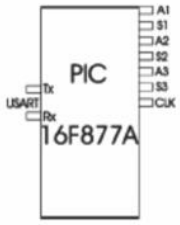


Figura 29: Microcontrolador Principal

Para la comunicación entre el microcontrolador y la computadora se usa el modulo de transmisión USART del microcontrolador (pines 25 y 26) para ello hay que configurar los registros internos del microcontrolador. Primero se configura el generador de baud rate, se activa la transmisión asíncrona y full duplex, se elije enviar tramas de 8 bits a una velocidad de 9.6 Kbytes/segundo. Se cargan los valores en el registro TXREG y se envían, para recibir los datos se lee el

registro RCREG cuando se activa el registro RCIF, ahora los datos fueron enviados y recibidos.

Además de la comunicación con la computadora el microcontrolador principal tiene la función de controlar los microcontroladores secundarios, esto lo logra con los tres pines A, que activan o desactivan el funcionamiento de los motores, los 3 pines S indican el sentido de giro y el pin CLK sincroniza los pasos en los motores.

Cabe mencionar que se dejaron libres los demás pines del microcontrolador, que son convertidores analógicos-digitales o puertos de entrada y salida programables, esto se debe a que se permite la modificación o mejora del prototipo con base a técnicas de control y sensado.

5.6.2. Microcontroladores secundarios

Se encargaran de generar las secuencias de movimiento de los motores a pasos dependiendo de las órdenes del microcontrolador principal.

Figura 30: Microcontrolador Secundario

Hay tres pines de entrada y cuatro de salida, el pin A es la activación de la secuencia, si esta encendido inicia la secuencia de pasos y la detiene cuando se desactiva, el pin C es el reloj de sincronización, indica cuando el motor debe cambiar de paso mediante un tren de pulsos, el pin S indica si el motor sigue su secuencia para girar en sentido horario o antihorario, los pines de salida están numerados para enviar una secuencia de activación de dos pines por paso, como se explica secuencia para manejar un steeper motor (secuencia normal) ya sea en sentido directo o en sentido inverso.

En el caso de los microcontroladores secundarios también se dejan los pines de conversión analogía digital y varios pines de entrada salida programables para sensar los motores de manera independiente y tomar dediciones descentralizadas que de lo contrario podrían complicar el código del microcontrolador principal.

5.6.3. Interacción entre los microcontroladores

Los microcontroladores secundarios se conectan al principal mediante un bus de tres terminales, aun manejando cuatro motores solo se usan tres controladores ya que dos de los motores comparten la misma secuencia de pasos solo que invertida, por lo cual las conexiones quedan de la siguiente manera.

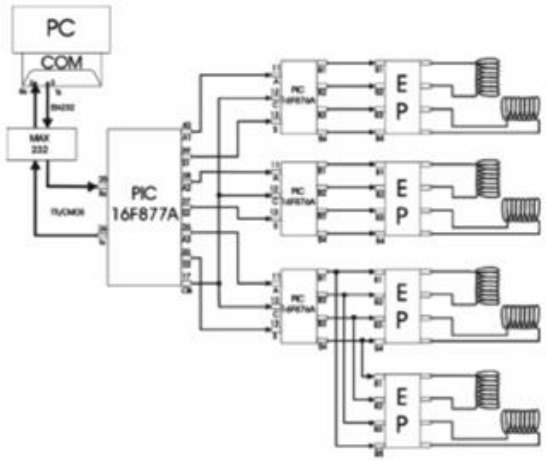


Figura 31: Conexión entre microcontroladores

Así es como queda la conexión entre los diferentes componentes del sistema de control y etapa de potencia, los números indican el numero de pin en el componente, y las letras son la nomenclatura que se les asigna a dichos pines, los bloque de la etapa de potencia ya fueron explicados en una sección de etapa de potencia y aislamiento.

6. AHIETI-SC

El AHIETI-SC es la interfase gráfica que interactúa directamente entre el usuario u operador y el robot, se encarga de interpretar los archivos PCB creados previamente, convertirlos en archivos de soporte, presentar una vista previa, simular la trayectoria que el AHIETI v2.1.1 debera seguir para elaborar el circuito PCB y ejecutar dicha trayectoria.

Este proyecto a sido desarrollado en su totalidad con la herramienta de desarrollo Microsoft Visual Studio .NET 2003 con el lenguaje de programación Visual Basic para tener compatibilidad con Microsoft Windows.

Los archivos PCB son generados en un programa externo (TRAXMAKER), esto es por la mayor cantidad de herramientas y librerías previamente desarrol-

ladas, además sería innecesario desarrollar una herramienta que ya se encuentra disponible.

La posición se determina contando los pasos de los stepper motors ya que cada paso como ya se explico anteriormente tiene un avance constante, por esto, es posible determinar la posición de los deslizadores simplemente con los pasos del mismo motor.

La trayectoria de los deslizadores es determinada por un algoritmo que permite prever en que posición el AHIETI v2.1.1 debe descobrizar y en que posición solo debe avanzar.

6.1. Estudio de los archivos PCB

Como mencionamos anteriormente estos archivos son creados por otro programa llamado TRAXMAKER el cual es independiente al sistema desarrollado en este trabajo.

6.2. Archivos PCB

Los archivos generados por el programa TRAXMAKER tienen la extensión PCB (Printable Circuit Board, Placa de Circuito Impreso), a continuación se explicara la estructura de un archivo PCB.

Cabe mencionar que el programa esta diseñado para trabajar en el sistema ingles, por lo cual las medidas esta dadas en MILS, no en milímetros, esto puede cuasar confusiones con personas que están acostumbradas al uso del sistema internacional.

6.3. Propuesta del intérprete

El AHIETI v2.1.1 ha sido diseñado para seguir trayectorias lineales y decidir en base a los archivos PCB generados por el TRAXMAKER en donde se debe descobrizar y en donde se debe mantener el cobre para la interconexión de los componentes del circuito electrónico, pero estos archivos contienen parámetros que solo son para el uso del programa, por ejemplo el color de las pistas, por eso se deben interpretar para utilizar los parámetros necesarios.

Los parámetros que se pueden tomar en cuenta son los de posición, tamaño, forma, cara y capa. Hay que tomar en cuenta que los valores de posición y tamaño están dados en mils y para poder trabajar con ellos mas fácilmente hay que convertirlos en milímetros; el valor que indica la forma es un numero entero y por ultimo la cara y la capa esta indicado por números enteros que representan colores, por ejemplo uno es el color rojo que indica que el elemento esta en la cara inferior en la primera capa.

ladas, además sería innecesario desarrollar una herramienta que ya se encuentra disponible.

La posición se determina contando los pasos de los stepper motors ya que cada paso como ya se explico anteriormente tiene un avance constante, por esto, es posible determinar la posición de los deslizadores simplemente con los pasos del mismo motor.

La trayectoria de los deslizadores es determinada por un algoritmo que permite prever en que posición el AHIETI v2.1.1 debe descobrizar y en que posición solo debe avanzar.

6.1. Estudio de los archivos PCB

Como mencionamos anteriormente estos archivos son creados por otro programa llamado TRAXMAKER el cual es independiente al sistema desarrollado en este trabajo.

6.2. Archivos PCB

Los archivos generados por el programa TRAXMAKER tienen la extensión PCB (Printable Circuit Board, Placa de Circuito Impreso), a continuación se explicara la estructura de un archivo PCB.

Cabe mencionar que el programa esta diseñado para trabajar en el sistema ingles, por lo cual las medidas esta dadas en MILS, no en milímetros, esto puede cuasar confusiones con personas que están acostumbradas al uso del sistema internacional.

6.3. Propuesta del intérprete

El AHIETI v2.1.1 ha sido diseñado para seguir trayectorias lineales y decidir en base a los archivos PCB generados por el TRAXMAKER en donde se debe descobrizar y en donde se debe mantener el cobre para la interconexión de los componentes del circuito electrónico, pero estos archivos contienen parámetros que solo son para el uso del programa, por ejemplo el color de las pistas, por eso se deben interpretar para utilizar los parámetros necesarios.

Los parámetros que se pueden tomar en cuenta son los de posición, tamaño, forma, cara y capa. Hay que tomar en cuenta que los valores de posición y tamaño están dados en mils y para poder trabajar con ellos mas fácilmente hay que convertirlos en milímetros; el valor que indica la forma es un numero entero y por ultimo la cara y la capa esta indicado por números enteros que representan colores, por ejemplo uno es el color rojo que indica que el elemento esta en la cara inferior en la primera capa.

Para interpretar el archivo se lee línea por línea, al leer la línea se debe inspeccionar si es una etiqueta o es una línea de parámetros, como en la mayoría de las veces una etiqueta precede a una línea de parámetros; al leer una etiqueta se deduce que la siguiente línea debe ser de parámetros.

Cuando se lee una línea de parámetros se debe elegir que parámetros son útiles y cuales son innecesarios para ello se debe separa la línea en todas sus componentes valorar las componentes y convertir las que deban ser convertidas ya que se encuentran mezclados los parámetros de posición y tamaño con otros de los cuales se debe tomar el valor así como es mostrado.

Para no hacer un intérprete tan complicado que haga todo el trabajo para cada función se hace un interprete que también traduzca y genere archivos de soporte que sean fácilmente entendibles por cada una de las funciones del programa.

Para la primera de las funciones que es la de graficar el archivo se hace un archivo que nos permita usar los parámetros de posición, tamaño capa y cara, después de esto que convierta los valores de posición y tamaño de mils a la unidad que maneja la computadora para representar gráficos es decir, a píxeles para esto primero se convierte el valor a milímetros y posteriormente a un equivalente en píxeles. Ya con los valores que se tienen se crea un archivo con una extensión APP (AHIETI Píxel PCB).

La segunda función es la de generación de la placa de circuito impreso, para esta necesitamos en primera las posiciones de los elementos y el ancho d las líneas, por otra parte se requieren de las posiciones que s van a perforar, primero tomamos los valores que están en mils y los convertimos en milímetros y creamos dos archivos uno para posteriormente calcular la trayectoria y otro para posicionar el taladro en el lugar correcto. El primer archivo tiene una extensión AMP (AHIETI Milimetric PCB) y en segundo una extensión ADP (AHIETI Drill Position).

7. Sistema de trayectoria

Para facilitar el calculo de la trayectoria se considera el movimiento del AHIETI v2.1.1 parecido al de una impresora, en estas los cabezales de impresión se encuentra en un deslizador que viaja a través de una flecha, a su vez un eje arrastra una hoja de papel, la acción combinada de estos mecanismos hacen que se cubra por completo el espacio de la hoja, el cabezal de impresión inyecta tinta o martilla una cinta entintada en puntos específicos para formar las imágenes.

En el AHIETI v2.1.1 el mecanismo es muy similar, por lo que el trayecto puede calcularse como en el de la impresora, el deslizador principal viaja a través de las flechas de deslizamiento para el deslizador principal subiendo el soporte del taladro del deslizador principal en los lugares donde queremos que el cobre se mantenga y bajándolo en los lugares donde queremos remover el cobre, a su

vez cuando el deslizador principal termina su camino regresa a su posición inicial y los deslizadores derecho con sprocket e izquierdo avanzan una posición, el deslizador principal realiza su viaje nuevamente, así sucesivamente hasta cubrir toda la superficie de la placa.

De acuerdo con las posiciones de las líneas que describen las pistas en el archivo AMP se calculan las ecuaciones de cada una de ellas, y se determina en que punto estas ecuaciones interceptan la trayectoria, con el ancho de la línea se calcula no solo el punto en donde la ecuación intercepta la trayectoria, sino también los puntos donde inicia y finaliza la pista. Así el programa determina si el taladro descobriza o solo se traslada.

Conclusiones

El deslizamiento de la FSDD del AHIETI v 1.0.0 debe tomarse con precaución, ya que presento problemas de deslizamiento, este detalle pudo haberse corregido reemplazando el tornillo de cuerda estándar por un tornillo de cuerda cuadrada, esto no fue posible debido al alto costo y la dificultad de fabricación.

Aun así se corregía el deslizamiento con un motor de corriente continua en la flecha opuesta al tornillo, pero presenta problemas de desgaste en el deslizador correspondiente y puede alterar la trayectoria de desplazamiento.

Por consiguiente con el AHIETI v 2.0.0 se corrigen estos problemas haciendo uso de dos flechas para cada deslizador y un steeper motor por cada uno de ellos.

No se usaron los drivers L297 y L298 por la dificultad de conseguirlos, sobre todo en la ciudad de Pachuca. Finalmente los PICs secundarios pueden llevar a cabo la misma función, además de estar apto para proveer mayores prestaciones con un diseño futuro.

El sistema de trayectoria estaba dispuesto a contornear las pistas una por una, los problemas con esta técnica es que al terminar el contorno de cada elemento, este quedaba aislado de los demás.

En un segundo sistema de trayectoria se ideaba calcular el punto medio entre las líneas y componentes, pero este método solo funciona con circuitos muy simples. En circuitos más complejos o de transmisión las pistas mas amplias tienden a generar un ruido capacitivo que interviene con el correcto funcionamiento del circuito diseñado.

Por esta razón se emplea el sistema de trayectoria explicado en el capitulo 5.5, ya que es mas parecido a las impresoras y evita la necesidad del calculo de colisiones de trayectorias o de hacerlas complejas.

En el AHIETI-SC se modifico el interprete, en un primer programa se hacia uso de controles (Texbox) para leer la totalidad de los archivos, posteriormente se realizaban las operaciones necesarias. En la versión final las líneas son procesadas

conforme se están leyendo esto hace que el proceso sea mas veloz y eficiente.

Se decide hacer una interfase sencilla usando varias ventanas, una para cada paso o función que se quiere realizar, en lugar de una sola ventana con exceso de controles que hace más difícil el manejo del programa.

El graficado y simulación del circuito se hace a escala y no a tamaño real, debido a que se quiere una mejor compatibilidad entre computadoras, si se hace en tamaño real la resolución de diferentes monitores y tarjetas graficas pueden alterar la presentación del circuito.

El uso del puerto serial en lugar del puerto paralelo es para evitar confusiones con las conexiones, esto también nos aporta una interfase más rápida con transmisión bidireccional en modo full-duplex.

No se uso un motor de corriente continua en lugar del mini-drill, ya que se hubiera tenido que adaptar un mandrill a la flecha del motor y cabía la posibilidad de que no lograra la fuerza que un motor de corriente alterna nos proporciona, además se uso por que en funcionamiento normal el taladro hace este tipo de trabajo sin ningún problema.

El programa TRAXMAKER tiene la capacidad de generar archivos para el manejo de dispositivos de producción (archivos gerber y N/C drill), por otro lado se utilizaron los archivos que el programa genera para comprender la forma en que se pueden convertir los trazos de un dibujo a movimientos en el actuador electromecánico.

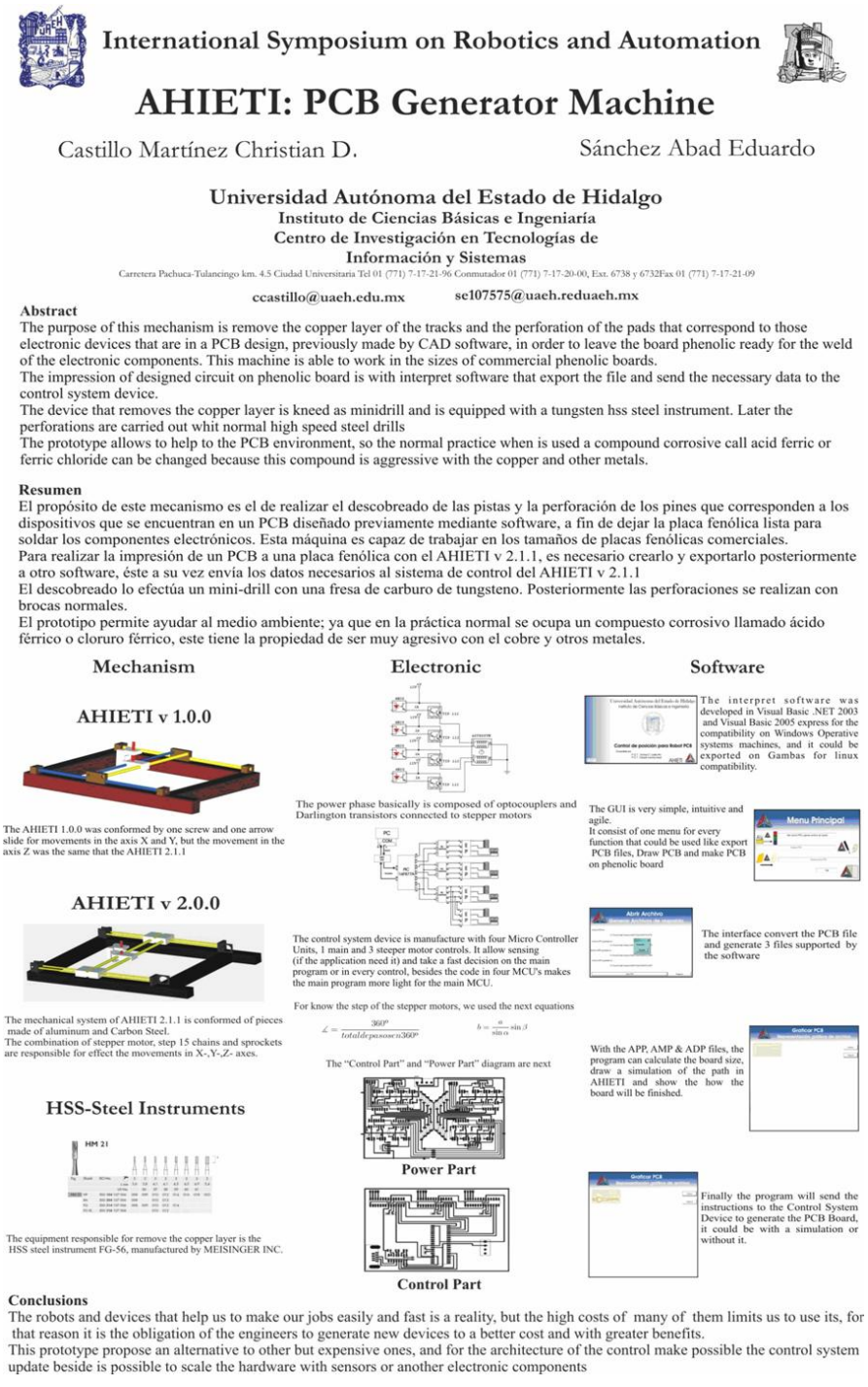
Aunque el dispositivo puede comportarse como una impresora, no es nuestra meta fabricar una, en el mercado se encuentran estos dispositivos y son demasiado costosos por comportarse como tal.

Referencias

[1] Floyd Thomas L. "**Dispositivos Electrónicos**". Ed. Limusa., México, 2001, pp. 295

[2] http://www.tendu.com/flexicam.htm (Marzo-Abril 2006)

Participation in 5th International Symposium on Robotics and Automation with the contribution in Poster, August 25-28, 2006. San Miguel Regla, México



International Symposium on Robotics and Automation

AHIETI: PCB Generator Machine

Castillo Martínez Christian D. Sánchez Abad Eduardo

Universidad Autónoma del Estado de Hidalgo
Instituto de Ciencias Básicas e Ingeniaría
Centro de Investigación en Tecnologías de Información y Sistemas
Carretera Pachuca-Tulancingo km. 4.5 Ciudad Universitaria Tel 01 (771) 7-17-21-96 Conmutador 01 (771) 7-17-20-00, Ext. 6738 y 6732Fax 01 (771) 7-17-21-09

ccastillo@uaeh.edu.mx **se107575@uaeh.reduaeh.mx**

Abstract
The purpose of this mechanism is remove the copper layer of the tracks and the perforation of the pads that correspond to those electronic devices that are in a PCB design, previously made by CAD software, in order to leave the board phenolic ready for the weld of the electronic components. This machine is able to work in the sizes of commercial phenolic boards.
The impression of designed circuit on phenolic board is with interpret software that export the file and send the necessary data to the control system device.
The device that removes the copper layer is kneed as minidrill and is equipped with a tungsten hss steel instrument. Later the perforations are carried out whit normal high speed steel drills
The prototype allows to help to the PCB environment, so the normal practice when is used a compound corrosive call acid ferric or ferric chloride can be changed because this compound is aggressive with the copper and other metals.

Resumen
El propósito de este mecanismo es el de realizar el descobreado de las pistas y la perforación de los pines que corresponden a los dispositivos que se encuentran en un PCB diseñado previamente mediante software, a fin de dejar la placa fenólica lista para soldar los componentes electrónicos. Esta máquina es capaz de trabajar en los tamaños de placas fenólicas comerciales.
Para realizar la impresión de un PCB a una placa fenólica con el AHIETI v 2.1.1, es necesario crearlo y exportarlo posteriormente a otro software, éste a su vez envía los datos necesarios al sistema de control del AHIETI v 2.1.1
El descobreado lo efectúa un mini-drill con una fresa de carburo de tungsteno. Posteriormente las perforaciones se realizan con brocas normales.
El prototipo permite ayudar al medio ambiente; ya que en la práctica normal se ocupa un compuesto corrosivo llamado ácido férrico o cloruro férrico, este tiene la propiedad de ser muy agresivo con el cobre y otros metales.

Mechanism

AHIETI v 1.0.0

The AHIETI 1.0.0 was conformed by one screw and one arrow slide for movements in the axis X and Y, but the movement in the axis Z was the same that the AHIETI 2.1.1

AHIETI v 2.0.0

The mechanical system of AHIETI 2.1.1 is conformed of pieces made of aluminum and Carbon Steel.
The combination of stepper motor, step 15 chains and sprockets are responsible for effect the movements in X-,Y-,Z- axes.

HSS-Steel Instruments

The equipment responsible for remove the copper layer is the HSS steel instrument FG-56, manufactured by MEISINGER INC.

Electronic

The power phase basically is composed of optocouplers and Darlington transistors connected to stepper motors

The control system device is manufacture with four Micro Controller Units, 1 main and 3 steeper motor controls. It allow sensing (if the application need it) and take a fast decision on the main program or in every control, besides the code in four MCU's makes the main program more light for the main MCU.

For know the step of the stepper motors, we used the next equations

$$\angle = \frac{360^{\circ}}{totaldepasosen360^{\circ}} \qquad b = \frac{a}{\sin\alpha}\sin\beta$$

The "Control Part" and "Power Part" diagram are next

Power Part

Control Part

Software

The interpret software was developed in Visual Basic .NET 2003 and Visual Basic 2005 express for the compatibility on Windows Operative systems machines, and it could be exported on Gambas for linux compatibility.

The GUI is very simple, intuitive and agile.
It consist of one menu for every function that could be used like export PCB files, Draw PCB and make PCB on phenolic board

The interface convert the PCB file and generate 3 files supported by the software

With the APP, AMP & ADP files, the program can calculate the board size, draw a simulation of the path in AHIETI and show the how the board will be finished.

Finally the program will send the instructions to the Control System Device to generate the PCB Board, it could be with a simulation or without it.

Conclusions

The robots and devices that help us to make our jobs easily and fast is a reality, but the high costs of many of them limits us to use its, for that reason it is the obligation of the engineers to generate new devices to a better cost and with greater benefits.
This prototype propose an alternative to other but expensive ones, and for the architecture of the control make possible the control system update beside is possible to scale the hardware with sensors or another electronic components

Bibliografía

[1] Ollero B., Anibal. Robótica, Manipuladores y Robots Móviles . Ed. Alfa Omega Marcombo. España. 2001, pp. 1-13.

[2] Universidad Tecnológica Nacional Facultad Regional La Plata Departamento de Ingeniería Mecánica Laboratorio de Control Numérico de M.H. EL CONTROL NUMÉRICO DE MÁQUINAS HERRAMIENTAS

[3] Floyd Thomas L. Dispositivos Electrónicos . Ed. Limusa México 2001, pp. 295

[4] Mendoza V., José Rafael. Diseño Del Control De Un Robot De Dos Grados De Libertad Para Aplicaciones De Seguimiento De Objetos . tésis para obtener el grado de Maestro en Ciencias en la especialidad de Electrónica en el Instituto Nacional de Astrofísica, Óptica y Electrónica. Tonantzintla, Puebla México. 2003 pp. 2-5.

[5] http://www.cenam.mx/cmu-mmc/Historia.htm (Marzo-Abril 2006)

[6] http://www.tendu.com/ exicam.htm (Marzo-Abril 2006)

[7] http://www.infoacero.cl/acero/que_es.htm (Marzo-Abril 2006)

[8] http://www.arqhys.com/arquitectura/aluminio-aplicaciones.html (Marzo-Abril 2006)

[9] http://www.arqhys.com/construccion/aluminio-aleaciones.html (Marzo-Abril 2006)

[10] http://es.wikipedia.org/wiki/M%C3%A1quina-herramienta (Marzo-Abril 2006)

[11] http://www.sinergia-web.com.mx/ clases/asm9708/Temas/clase02.htm (Marzo 2006)

[12] http://cm.bell-labs.com/cm/cs/who/dmr/chist.html (Marzo 2006)

[13] http://www.research.att.com/~bs/C++.html (Febrero 2006)

[14] http://leo.worldonline.es/acanudas/pdf/dp5intro.pdf (Marzo 2006)

[15] http://pisuerga.inf.ubu.es/lsi/Invest/Java/Tuto/I_2.htm (Marzo 2006)

[16] http://www.microsoft.com (Marzo 2006)

[17] Microsoft Corporation, "MSDN Library Visual Studio 6.0", Ed. Microsoft Press. 1991.

[18] Microsoft Corporation, "MSDN Library de Visual Studio .NET 2003", Ed Microsoft Press. 2002.

[19] www.rae.es (mayo 2006)

[20] http://www.gizmology.net/sprockets.htm (mayo 2006)

[21] http://es.wikipedia.org/wiki/GPL (mayo 2006)

[22] http://www.t-tech.com/products/upload/T-Tech%20Brochure.pdf (Agosto 2006)