UNIVERSIDAD MAYOR DE SAN ANDRÉS

FACULTAD DE TECNOLOGÍA

CARRERA DE ELECTRÓNICA Y TELECOMUNICACIONES

NIVEL TÉCNICO UNIVERSITARIO SUPERIOR

PROYECTO DE GRADO

DISEÑO DE SISTEMA DE TIMBRE AUTOMATICO PARA LA UNIDAD EDUCATIVA “PEDRO POVEDA”

POSTULANTE: UNIV. ISAAC ALBARO APAZA RAMIREZ

TUTOR: LIC. NIXON EMILIANO VARGAS MAMANI

La Paz – Bolivia

2018

DEDICATORIA

A Dios por guiarme siempre por un buen camino y seguir adelante

A mis padres, en especial a la mujer que me dio la vida por su apoyo incondicional en todos estos años ya que a su lado vivimos las alegrías y tristezas que nos da la vida brindándome su confianza y proponiéndome grandes metas a futuro junto con mi padre ya que para ellos aunque sea mayor de edad me siguen cuidando y protegiendo como un gran tesoro

Y mi gran amigo Jaime, mi hermano que siempre confió en mí con sus grandes concejos y enseñanzas

AGRADECIMIENTO

A mi casa de estudios mayor a la UMSA

UNIVERSIDAD MAYOR DE SAN ANDRES

Quiero agradecer al Licenciado NIXON VARGAS docente de la carrera Electrónica y telecomunicaciones por la ayuda por los consejos que me dio día a día para poder realizar este proyecto de grado, también darle las gracias por esa motivación que nos dio para superarnos en nuestra profesión por esa paciencia que tuvo mi persona

ÍNDICE

ÍNDICE III
RESUMEN VII
CAPITULO I 1
MARCO REFERENCIAL 1
1.1 INTRODUCCIÓN 1
1.2 PLANTEAMIENTO DEL PROBLEMA 1
1.3 OBJETIVOS DEL PROYECTO 2
1.3.1 OBJETIVO GENERAL 2
1.3.2 OBJETIVOS ESPECIFICOS 2
1.4 METODOLOGÍA DE LA INVESTIGACIÓN 2
1.5 JUSTIFICACIÓN DEL PROYECTO 2
1.5.1 JUSTIFICACIÓN TECNOLÓGICA 2
1.5.2 JUSTIFICACIÓN ACADÉMICA 3
1.5.3 JUSTIFICACIÓN SOCIAL 3
1.6 LÍMITES DEL PROYECTO 3
1.7 ALCANCE DEL PROYECTO 3
1.8 CRONOGRAMA DE ACTIVIDADES 3
CAPITULO II 4
MARCO TEÓRICO 4
2.1 DEFINICIONES 4
2.1.1 LENGUAJE DE PROGRAMACIÓN 4
2.1.2 ESTÁNDAR 5
2.2 CONCEPTO AUTOMATIZACIÓN 5
2.3 UART 5
2.3.1 ESTRUCTURA DE DATOS 5
2.3.2 RECEPTOR 6

2.3.3 TRANSMISOR 7
2.3.4 APLICACIÓN 7
2.3.5 HISTORIA 8
2.3.6 ESTRUCTURA 9
2.3.7 ERROR DE DESBORDAMIENTO 9
2.3.8 ERROR DE INFRAUTILIZACIÓN 10
2.3.9 ERROR DE TRAMA 10
2.3.10 ERROR DE PARIDAD 10
2.3.11 CONDICIÓN DE DESCANSO 10
2.4 COMANDOS AT 11
2.5 HC-05 BLUETOOTH 11
2.5.1 HC-05 COMO ESCLAVO 12
2.5.2 HC-05 COMO MAESTRO 12
2.6 MICROCONTROLADOR 13
2.6.1 INTRODUCCIÓN 13
2.6.2 DIFERENCIA ENTRE MICROCONTROLADOR Y MICROPROCESADOR 13
2.6.3 RECURSOS DE LOS MICROCONTROLADORES 14
2.6.4 ARQUITECTURA BÁSICA 14
2.6.5 PROCESADOR O CPU 15
2.6.6 MEMORIA 16
2.6.7 PUERTOS DE ENTRADA Y SALIDA 16
2.6.8 RELOJ PRINCIPAL 17
2.6.9 RECURSOS ESPECIALES DE LOS MICROCONTROLADORES 17
2.6.9.1 TEMPORIZADORES 17
2.6.9.2 PERRO GUARDIÁN O WATCHDOG 18
2.6.9.3 PROTECCIÓN ANTE FALLO DE ALIMENTACIÓN O BROWNOUT 18
2.6.9.4 ESTADO DE REPOSO O DE BAJO CONSUMO 18
2.6.9.5 CONVERSOR ANALÓGICO A DIGITAL (ADC) 18
2.6.9.6 CONVERSOR DIGITAL A ANALÓGICO (DAC) 19

2.6.9.7 COMPARADOR ANALÓGICO 19
2.6.9.8 MODULADOR POR ANCHO DE PULSO (PWM) 19
2.6.9.9 PUERTOS DE COMUNICACIÓN 19
2.7 INTERFAZ DE USUARIO 20
2.7.1 DEFINICIÓN 20
2.7.2 FUNCIONES PRINCIPALES 20
2.7.3 TIPOS DE INTERFACES DE USUARIO 20
2.8 VISUAL STUDIO 21
2.8.1 VERSIONES 21
2.8.1.1 VISUAL STUDIO.NET 2002 22
2.8.1.2 VISUAL STUDIO.NET 2003 23
2.8.1.3 VISUAL STUDIO.NET 2005 24
2.8.1.4 VISUAL STUDIO.NET 2008 25
2.8.1.5 VISUAL STUDIO.NET 2010 26
2.8.1.6 OTRAS VERSIONES 27
2.8.2 VISUAL BASIC.NET 28
2.8.3 RELACIÓN CON VISUAL BASIC 29
2.8.4 APLICACIÓN WINDOWS FORM 30
2.9 PIC C CCS 31
2.9.1 INTRODUCCIÓN 31
2.9.2 PROGRAMAS DE UTILIDAD 32
2.9.3 OPERADORES Y EXPRESIONES 33
2.9.4 DIRECTIVAS DEL PROCESADOR 34
2.9.5 CONTROL DE MEMORIA 34
2.9.6 CONTROL DE COMPILADOR 35
2.9.7 IDENTIFICADORES PREDEFINIDOS 35
2.9.8 DIRECTIVAS DEL C ESTÁNDAR 35
2.9.9 CALIFICADORES DE FUNCIÓN 36
2.9.10 LIBRERÍAS INCORPORADAS 37

CAPITULO III 38
DESARROLLO DE LA INVESTIGACIÓN 38
3.1 METODOLOGÍA 38
3.2 DIAGRAMA BÁSICO DEL SISTEMA 38
3.2.1 DESCRIPCIÓN DEL DIAGRAMA DE BLOQUES 39
BIBLIOGRAFÍA 40

RESUMEN

El presente proyecto tiene como objetivo, realizar el diseño de un sistema de timbre automático para la unidad educativa Pedro Poveda. Para este proceso se presentaran controladores y software con interfaz gráfica para un mejor entendimiento del mismo.

En cuanto a puntualidad se refiere, se ve notablemente que las unidades educativas no cuentan con un estricto control, lo que hace que el aprovechamiento de los horarios no sea el correcto, dando lugar a conflictos entre profesores y estudiantes, se nota que la unidades educativas cuentan con personal administrativo el cual se encarga de accionar el timbre de manera manual, siendo esta una solución pero no la adecuada en cuando a precisión se refiere.

Para lo cual se decidió hacer el diseño de un timbre automático para la unidad educativa el cual cuenta con un sistema de microcontrolador e interfaz gráfica para su configuración, lo cual brinda mayor comodidad y precisión en los horarios establecidos.

El sistema nos permitirá agregar los horarios de la unidad educativa contemplando horarios de invierno normal y extendido, también contando con los descansos pedagógicos de invierno y fin de año, dando configuración solo una vez por gestión.

El sistema también contara con activación manual en casos de emergencia deshabilitando por un día el sistema automático.

El software que hará la parte de configuración estará diseñado bajo tendencias actuales, las cuales permitirán al usuario a cargo de la configuración un entorno amigable, fácil de usar y bastante intuitivo.

.

CAPITULO I

MARCO REFERENCIAL

1.1 INTRODUCCIÓN

El continúo avance tecnológico y la necesidad que ahora representa para las personas, dándoles comodidad, ha llevado a que las unidades educativas cuenten con comodidades como cámaras de vigilancia, equipos de laboratorio y otro tipo de equipamiento el cual sea de beneficio a estas instituciones.

Se observa que cuanto a la precisión de los horarios mucha unidades educativas cuentan con timbres manuales controlados por administrativos designados día a día, lo cual hace notable de deficiencia en cuanto a los cambios de hora, teniendo adelantos o retrasos los cuales son de perjuicio tanto para los profesores como para los estudiantes.

Es por ello que con el diseño de un sistema de timbre automático para la unidad educativa “Pedro Poveda” se busca mejorar el rendimiento en cuanto los horarios establecidos, y así dar un ejemplo en cuanto a puntualidad se refiere.

1.2 PLANTEAMIENTO DEL PROBLEMA

La necesidad de automatizar el timbre de la unidad educativa “Pedro Poveda”, surge a causa de la necesidad de tener los horarios establecidos de manera correcta y que no haya conflictos por adelantos o atrasos.

La solución planteada es factible, ya que, el personal administrativo ya no se encargara de accionar el timbre en horas establecidas y podrán realizar otras tareas, los profesores de la unidad educativa cumplirán a cabalidad el horario que les corresponde, el horario de entrada y salida será de manera exacta, con esto se garantiza un mejor rendimiento dentro de la unidad educativa.

1.3 OBJETIVOS DEL PROYECTO

1.3.1 OBJETIVO GENERAL

Realizar un sistema electrónico el cual sea capaz de automatizar el timbre de la unidad educativa “Pedro Poveda”, teniendo un software de configuración y establecimiento de horarios.

1.3.2 OBJETIVOS ESPECIFICOS

- Diseñar el firmware para microcontrolador
- Desarrollar una Interfaz Gráfica de Usuario de configuración.
- Desarrollar el circuito esquemático de acuerdo a cálculos y criterios en la selección de componentes.
- Diseñar el PCB del sistema.

1.4 METODOLOGÍA DE LA INVESTIGACIÓN

La investigación estará basada en la “investigación aplicada experimental”, con producto final; un sistema de timbre automático para la unidad educativa “Pedro Poveda”.

Para su implementación se usaran métodos y técnicas de diseño correspondiente a la informática y electrónica de control. Para el accionar del timbre se hará uso de un contactor. Se propone también el diseño de una interfaz gráfica de usuario, la cual servirá para la configuración del sistema.

1.5 JUSTIFICACIÓN DEL PROYECTO

1.5.1 JUSTIFICACIÓN TECNOLÓGICA

Hoy en día los instrumentos o equipos electrónicos son más sofisticados y dan mayor comodidad a la sociedad que los emplea, es por eso que este proyecto es viable tecnológicamente.

1.5.2 JUSTIFICACIÓN ACADÉMICA

En el transcurso de los años se fue adquiriendo conocimientos los cuales ahora se ven aplicados, dando solución a problemas de la sociedad.

1.5.3 JUSTIFICACIÓN SOCIAL

Este proyecto está al alcance de todo bolsillo debido a que no es muy costoso por lo tanto se puede beneficiar a mucha gente es especial al personal educativo y personal de trabajo de diversas instituciones

1.6 LÍMITES DEL PROYECTO

El presente proyecto tendrá la limitante de tener la necesidad de ser configurado una vez al año mínimamente.

1.7 ALCANCE DEL PROYECTO

El presente proyecto por su innovación y aplicación, supone un sistema prototipo aplicado, lo cual a lo largo de su desarrollo de verán las falencias y soluciones a la mismas.

1.8 CRONOGRAMA DE ACTIVIDADES

ACTIVIDAD	AGOSTO				SEPTIEMPBRE		
Recopilación de información							
Diseño de la interfaz gráfica de usuario							
Diseño del firmware del microcontrolador							
Diseño del PCB y Pruebas iniciales							
Revisión y correcciones							
Producto final a ser implementado							

CAPITULO II

MARCO TEÓRICO

2.1 DEFINICIONES

2.1.1 LENGUAJE DE PROGRAMACIÓN

Un lenguaje de programación es un lenguaje formal que especifica una serie de instrucciones para que una computadora produzca diversas clases de datos. Los lenguajes de programación pueden usarse para crear programas que pongan en práctica algoritmos específicos que controlen el comportamiento físico y lógico de una computadora.

Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila (de ser necesario) y se mantiene el código fuente de un programa informático se le llama programación.

También la palabra programación se define como el proceso de creación de un programa de computadora, mediante la aplicación de procedimientos lógicos, a través de los siguientes pasos:

El desarrollo lógico del programa para resolver un problema en particular.

Escritura de la lógica del programa empleando un lenguaje de programación específico (codificación del programa).

Ensamblaje o compilación del programa hasta convertirlo en lenguaje de máquina.

Prueba y depuración del programa.

Desarrollo de la documentación.

2.1.2 ESTÁNDAR

Establece la guía de planeación y construcción, específica los requisitos mínimos de cumplimiento.

2.2 CONCEPTO AUTOMATIZACIÓN

La automatización (del griego antiguo, auto que significa guiado por uno mismo) es el uso de sistemas o elementos computarizados y electromecánicos para fines industriales. Como una disciplina de la ingeniería más amplia que un sistema de control, abarca la instrumentación industrial, que incluye los sensores, o transmisores de campo, los sistemas de control y supervisión, los sistemas de transmisión y recolección de datos y las aplicaciones de software en tiempo real para supervisar y controlar las operaciones de plantas o procesos industriales.

2.3 UART

Un receptor-transmisor asíncrono universal (UART) es un dispositivo de hardware de computadora para comunicación serie asincrónica en el que el formato de datos y las velocidades de transmisión son configurables. Los niveles y métodos de señalización eléctrica son manejados por un circuito de controlador externo al UART.

Un UART es generalmente un circuito integrado (IC) individual (o parte de) utilizado para comunicaciones en serie a través de una computadora o puerto serie de dispositivo periférico. Uno o más periféricos UART se integran comúnmente en chips de microcontroladores. Un dispositivo relacionado, el receptor-transmisor síncrono y asíncrono universal (USART) también admite la operación síncrona.

2.3.1 ESTRUCTURA DE DATOS

El estado inactivo, sin datos, es de alto voltaje o alimentado. Este es un legado histórico de la telegrafía, en el que la línea se mantiene alta para mostrar que la línea y el transmisor no están dañados. Cada carácter está enmarcado como un bit lógico de inicio bajo, bits de datos, posiblemente un bit de paridad, y uno o más bits de parada. En la mayoría de las aplicaciones, el bit de datos menos significativo (el de la izquierda en

este diagrama) se transmite primero, pero hay excepciones (como el terminal de impresión IBM 2741).

El bit de inicio le indica al receptor que viene un nuevo personaje. Los siguientes cinco a nueve bits, dependiendo del código establecido, representan el personaje. Si se usa un bit de paridad, se colocará después de todos los bits de datos. Los siguientes uno o dos bits siempre están en la condición de marca (lógica alta, es decir, '1') y se llaman bit (s) de parada. Le indican al receptor que el personaje está completo. Como el bit de inicio es lógico bajo (0) y el bit de parada es lógico alto (1), siempre hay al menos dos cambios de señal garantizados entre los caracteres.

2.3.2 RECEPTOR

Todas las operaciones del hardware UART están controladas por una señal de reloj interno que se ejecuta en un múltiplo de la velocidad de datos, típicamente 8 o 16 veces la velocidad de bits. El receptor prueba el estado de la señal entrante en cada pulso de reloj, buscando el comienzo del bit de inicio. Si el bit de inicio aparente dura al menos la mitad del tiempo del bit, es válido y señala el inicio de un nuevo carácter. Si no, se considera un pulso falso y se ignora. Después de esperar un poco más de tiempo, el estado de la línea se muestrea nuevamente y el nivel resultante se registra en un registro de desplazamiento. Después de que ha transcurrido el número requerido de periodos de bits para la longitud del carácter (normalmente de 5 a 8 bits), los contenidos del registro de desplazamiento se ponen a disposición (en paralelo) del sistema receptor. El UART establecerá un indicador que indica que hay nuevos datos disponibles, y también puede generar una interrupción del procesador para solicitar que el procesador host transfiera los datos recibidos.

Los UART de comunicación generalmente no tienen un sistema de temporización compartido además de la señal de comunicación. Normalmente, los UART vuelven a sincronizar sus relojes internos en cada cambio de la línea de datos que no se considera un impulso espurio. Al obtener información de tiempo de esta manera, reciben de manera confiable cuando el transmisor está enviando a una velocidad ligeramente diferente de la que debería

Muchos UART tienen una pequeña memoria búfer FIFO de primero en entrar , primero en salir entre el registro de desplazamiento del receptor y la interfaz del sistema host. Esto le permite al procesador host incluso más tiempo para manejar una interrupción desde el UART y evita la pérdida de datos recibidos a altas velocidades.

2.3.3 TRANSMISOR

La operación de transmisión es más simple ya que el tiempo no tiene que determinarse a partir del estado de línea, ni está vinculado a ningún intervalo de temporización fijo. Tan pronto como el sistema emisor deposita un carácter en el registro de desplazamiento (después de completar el carácter anterior), el UART genera un bit de inicio, desplaza el número requerido de bits de datos a la línea, genera y envía el bit de paridad (si se usa), y envía los bits de parada. Como la operación de dúplex completo requiere que los caracteres se envíen y reciban al mismo tiempo, los UART utilizan dos registros de desplazamiento diferentes para los caracteres transmitidos y recibidos. Los UART de alto rendimiento podrían contener un búfer de transmisión FIFO (primero en entrar, primero en salir) para permitir que un controlador de CPU o DMA deposite varios caracteres en una ráfaga en el FIFO en lugar de depositar un carácter a la vez en el FIFO. Como la transmisión de un solo o múltiples caracteres puede llevar mucho tiempo en relación con las velocidades de la CPU, un UART mantiene un indicador que muestra el estado ocupado, de modo que el sistema host sabe si hay al menos un carácter en el búfer de transmisión o registro de desplazamiento; "listo para el / los siguiente (s) personaje (s)" también puede ser señalado con una interrupción.

2.3.4 APLICACIÓN

Los UART transmisores y receptores deben configurarse para la misma velocidad de bits, longitud de caracteres, paridad y bits de parada para un funcionamiento correcto. El UART receptor puede detectar algunos ajustes no coincidentes y establecer un bit de indicador de "error de trama" para el sistema host; en casos excepcionales, el UART receptor producirá una corriente errática de caracteres mutilados y los transferirá al sistema host.

Los puertos serie típicos utilizados con computadoras personales conectadas a módems utilizan ocho bits de datos, sin paridad y un bit de parada; para esta configuración, el número de caracteres ASCII por segundo es igual a la tasa de bits dividida por 10.

Algunas computadoras hogareñas de bajo costo o sistemas integrados prescinden de un UART y usan la CPU para muestrear el estado de un puerto de entrada o manipular directamente un puerto de salida para la transmisión de datos. Si bien requiere mucha CPU (dado que el tiempo de la CPU es crítico), el chip UART se puede omitir, ahorrando dinero y espacio. La técnica se conoce como golpeteo de bits .

2.3.5 HISTORIA

Algunos esquemas telegráficos tempranos usaban pulsos de longitud variable (como en el código Morse) y mecanismos de relojería giratorios para transmitir caracteres alfabéticos. Los primeros dispositivos de comunicación en serie (con impulsos de longitud fija) eran interruptores mecánicos giratorios (conmutadores). Varios códigos de caracteres utilizando 5, 6, 7 u 8 bits de datos se hicieron comunes en los teleimpresores y más tarde como periféricos de computadora. El teletipo hizo un excelente dispositivo de E / S de propósito general para una computadora pequeña.

Gordon Bell de DEC diseñó el primer UART, que ocupaba toda una placa de circuito llamada unidad de línea, para las computadoras PDP que comenzaba con el PDP-1

Un ejemplo de UART de principios de los 80 fue el National Semiconductor 8250 utilizado en la tarjeta original Adaptador de Comunicaciones Asíncronas de IBM PC .

Según el fabricante, se usan diferentes términos para identificar los dispositivos que realizan las funciones de UART. Intel llamó a su dispositivo 8251 una "Interfaz de comunicación programable". La tecnología MOS 6551 se conocía con el nombre de "Adaptador de interfaz de comunicaciones asíncronas" (ACIA). El término "Interfaz de comunicaciones en serie" (SCI) se utilizó por primera vez en Motorola en 1975 para referirse a su dispositivo de interfaz en serie asíncrono arrítmico, que otros llamaban UART. Zilog fabricó varios controladores de comunicación serie o SCC.

Después de que la mayoría de las computadoras IBM PC compatibles eliminaron el puerto COM RS-232 en la década de 2000, en su lugar podría usar un puente USB-UART externo. FTDI es un proveedor de estos chips.

2.3.6 ESTRUCTURA

Un UART generalmente contiene los siguientes componentes:

- un generador de reloj, generalmente un múltiplo de la velocidad de bits para permitir el muestreo en el medio de un período de bit.
- Registros de desplazamiento de entrada y salida
- transmitir / recibir control
- Lógica de control de lectura / escritura
- Búfer de transmisión / recepción (opcional)
- Búfer del bus de datos del sistema (opcional)
- Memoria intermedia de primer entrada, primera salida (FIFO) (opcional)
- Las señales necesarias para un controlador DMA de terceros (opcional)
- Controlador DMA maestro de bus integrado (opcional)

Condiciones especiales del transceptor

2.3.7 ERROR DE DESBORDAMIENTO

Se produce un "error de desbordamiento" cuando el receptor no puede procesar el carácter que acaba de entrar antes de que llegue el siguiente. Varios dispositivos tienen diferentes cantidades de espacio de búfer para contener los caracteres recibidos. La CPU o el controlador DMA deben dar servicio al UART para eliminar caracteres del búfer de entrada. Si la CPU o el controlador DMA no realizan el UART lo suficientemente rápido y el búfer se llena, se producirá un error de sobrecarga y se perderán los caracteres entrantes.

2.3.8 ERROR DE INFRAUTILIZACIÓN

Se produce un "error de subdesarrollo" cuando el transmisor UART ha completado el envío de un carácter y el búfer de transmisión está vacío. En los modos asíncronos, esto se trata como una indicación de que no quedan datos por transmitir, en lugar de un error, ya que se pueden agregar bits de parada adicionales. Esta indicación de error se encuentra comúnmente en USART

2.3.9 ERROR DE TRAMA

Un UART detectará un error de encuadre cuando no vea un bit de "detención" en el tiempo esperado después de un bit de "parada". Como el bit de "inicio" se utiliza para identificar el comienzo de un carácter entrante, su temporización es una referencia para los bits restantes. Si la línea de datos no está en el estado esperado (alto) cuando se espera el bit de "parada" (de acuerdo con la cantidad de datos y bits de paridad para los que está configurado el UART), el UART señalará un error de trama. Una condición de "interrupción" en la línea también se señala como un error de encuadre.

2.3.10 ERROR DE PARIDAD

Se produce un error de paridad cuando la paridad del número de 1 bits no concuerda con la especificada por el bit de paridad. El uso de un bit de paridad es opcional, por lo que este error solo se producirá si se ha habilitado la verificación de paridad.

2.3.11 CONDICIÓN DE DESCANSO

Se produce una condición de interrupción cuando la entrada del receptor está en el nivel de "espacio" (lógica baja, es decir, '0') durante más tiempo que una cierta duración de tiempo, generalmente, durante más de un tiempo de carácter. Esto no es necesariamente un error, sino que aparece para el receptor como un carácter de todos los bits cero con un error de encuadre. El término "ruptura" se deriva de la señalización de bucle de corriente, que era la señalización tradicional utilizada para teletipos. La condición de "espaciamiento" de una línea de bucle de corriente se indica porque no fluye corriente, y un período muy largo sin flujo de corriente a menudo es causado por un corte u otra falla en la línea.

Algunos equipos transmitirán deliberadamente el nivel de "espacio" durante más tiempo que un personaje como señal de atención. Cuando las tasas de señalización no coinciden, no se pueden enviar caracteres significativos, pero una señal larga de "interrupción" puede ser una forma útil de llamar la atención de un receptor no coincidente para hacer algo (como reiniciarse a sí mismo). Los sistemas informáticos pueden usar el nivel de "interrupción" prolongado como una solicitud para cambiar la velocidad de señalización, para admitir el acceso de marcación a múltiples velocidades de señalización. El protocolo DMX512 usa la condición de corte para señalar el inicio de un nuevo paquete.

2.4 COMANDOS AT

Los comandos AT son instrucciones usadas para controlar las acciones de un modem. AT es una abreviación de la palabra “attention”, que en español significa atención. Cada línea de comando debe comenzar con “AT” o “at”. Algunos de los comandos son: ATD (Dial), ATA (Answer), ATH (Hook control) y ATO (Return to online data estate), estos también son soportados por módems GSM/GPRS y por teléfonos móviles. Además de este conjunto de comandos, los módems GSM/GPRS y teléfonos móviles, también soportan comandos específicos para la tecnología GSM, entre estos se encuentran comandos que nos permiten operar con el servicio SMS.

2.5 HC-05 BLUETOOTH

Es necesario saber que existen diferentes modelos de módulos Bluetooth entre los más populares se encuentran HC-06 y el HC05. El módulo HC-06 funciona como Slave y el HC-05 como Master y Slave (lo que podría confundir a algunos). Físicamente se diferencian por el número de pines. En el HC-06 tiene un conector de 4 pines mientras que el HC-05 trae uno de 6 pines. Existen varios modelos y versiones para el módulo HC-05, el que usaremos es el que se muestra en las siguientes imágenes, que como vemos tiene un pulsador, el que nos servirá para entrar en Modo AT (modo de funcionamiento que permite configurar algunos parámetros del módulo).

EL modulo Bluetooth HC-05 viene configurado de fábrica como esclavo, pero se puede cambiar para que trabaje como maestro, además al igual que el HC-06, se puede cambiar

el nombre, contraseña de vinculación, velocidad y otros parámetros cuando entra en modo AT.

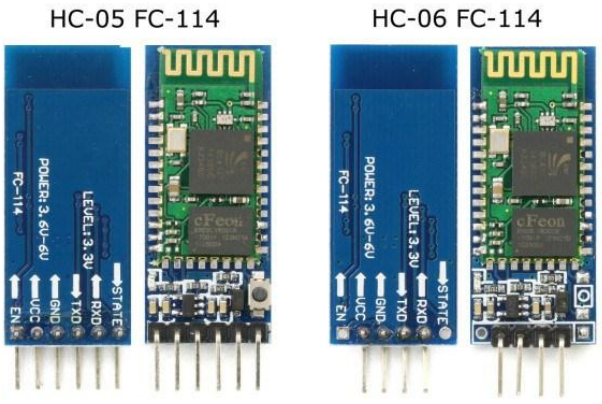

Figura 2.0.0 Modulo HC-05.
https://aprendiendoarduino.wordpress.com/tag/hc-05/

2.5.1 HC-05 COMO ESCLAVO

Cuando está configurado de esta forma espera que un dispositivo bluetooth maestro se conecte a él, generalmente se utiliza cuando se necesita comunicarse con un PC o Smartphone, pues estos se comportan como dispositivos maestros.

2.5.2 HC-05 COMO MAESTRO

En este modo, el HC-05 es el que inicia la conexión. Un dispositivo maestro solo se puede conectarse con un dispositivo esclavo. Generalmente se utiliza este modo para comunicar módulos bluetooth entre ellos.

El módulo HC-05 viene por defecto configurado de la siguiente forma:

- Modo: Esclavo - Nombre por defeco: HC-05 - Contraseña por defecto: 1234 - La velocidad por defecto: 9600 baudios

EL Modulo HC-05 tiene 4 estados:

Estado Desconectado: - Entra a este estado al alimentar el modulo y no se ha establecido una conexión bluetooth con ningún otro dispositivo - EL LED del módulo en este estado parpadea rápidamente - En este estado no puede interpretar los comandos AT

Estado Conectado o de comunicación: - Entra a este estado cuando se establece una conexión con otro dispositivo bluetooth. - El LED hace un doble parpadeo. - Todos los datos que se ingresen al HC-05 por el Pin RX se trasmiten por bluetooth al dispositivo conectado, y los datos recibidos se devuelven por el pin TX.

MODO AT 1

Para entrar a este estado después de conectar el modulo es necesario presionar el botón del HC-05. - En este estado, podemos enviar comandos AT, pero a la misma velocidad con el que está configurado. - EL LED del módulo en este estado parpadea rápidamente igual que en el estado desconectado.

MODO AT 2

Para entrar a este estado es necesario tener presionado el botón en el momento de alimentar el modulo, es decir el modulo debe encender con el botón presionado, después de haber encendido se puede soltar y permanecerá en este estado. - En este estado, para enviar comandos AT es necesario hacerlo a la velocidad de 38400 baudios, esto es muy útil cuando nos olvidamos la velocidad con la que hemos dejado configurado nuestro modulo. - EL LED del módulo en este estado parpadea lentamente.

2.6 MICROCONTROLADOR

2.6.1 INTRODUCCIÓN

Los microcontroladores en la actualidad son parte de nuestra vida. Se presentan en muchas aéreas de nuestra cotidianidad, se los puede encontrar funcionando en los ratones, teclados de las computadoras personales, televisores, sistemas de iluminación inteligente o donde quera que exista un sistema inteligente.

2.6.2 DIFERENCIA ENTRE MICROCONTROLADOR Y MICROPROCESADOR

El microprocesador es un chip que contiene una unidad central de proceso (CPU).

La CPU está formada por la unidad de control que interpreta las instrucciones, los pines de un microprocesador sacan al exterior buses de direcciones, datos y control, esto para comunicarse con las memorias y dispositivos de entrada y salida de datos, de modo que una computadora está formada por el microprocesador y varios circuitos integrados. Por esta razón el microprocesador es un sistema abierto ya que su configuración es variable y va de acuerdo a la aplicación a la que se destine.

El microcontrolador, por otro lado, es un sistema cerrado. Todas las partes del microcontrolador están contenidas en su interior y solo salen al exterior las líneas que gobiernan los periféricos. En la práctica cada fabricante de microcontroladores oferta un elevado número de modelos diferentes, desde los más sencillos hasta los más poderosos. Es posible seleccionar la capacidad de las memorias, el número de líneas de E/S, la cantidad y potencia de los elementos auxiliares, la velocidad de funcionamiento, etc. Por todo ello, un aspecto muy destacado del diseño es la selección del microcontrolador a utilizar.

2.6.3 RECURSOS DE LOS MICROCONTROLADORES

La estructura fundamental de los microcontroladores y sus características básicas son muy parecidas. Todos deben disponer de los bloques esenciales: Procesador, memoria de datos y de instrucciones, líneas de E/S, oscilador de reloj y módulos controladores de periféricos. Sin embargo, cada fabricante intenta enfatizar los recursos más idóneos para las aplicaciones a las que se destinan preferentemente.

2.6.4 ARQUITECTURA BÁSICA

Aunque inicialmente todos los microcontroladores adoptaron la arquitectura clásica de Von Neumann, en el presente se impone la arquitectura Harvard.

La arquitectura de Von Neumann se caracteriza por disponer de una sola memoria principal donde se almacenan datos e instrucciones de forma indistinta. A dicha memoria se accede a través de un sistema de buses único (direcciones, datos y control). La arquitectura Harvard dispone de dos memorias independientes, una que contiene solo instrucciones y otra, solo datos.

2.6.5 PROCESADOR O CPU

Es el elemento más importante del microcontrolador y determina sus principales características, tanto a nivel hardware como software. Se encarga de direccionar la memoria de instrucciones, recibir el código de la instrucción en curso, su decodificación y la ejecución de la operación que implica la instrucción, así como la búsqueda de los operandos y el almacenamiento del resultado.

Existen tres orientaciones en cuanto a la arquitectura y funcionalidad de los procesadores actuales.

CISC: Un gran número de procesadores usados en los microcontroladores están basados en la filosofía CISC (Computadoras de Juego de Instrucciones Complejo). Disponen de más de 80 instrucciones máquina en su repertorio, algunas de las cuales son muy sofisticadas y potentes, requiriendo muchos ciclos para su ejecución. Una ventaja de los procesadores CISC es que ofrecen al programador instrucciones complejas que actúan como macros.

RISC: Tanto la industria de los computadores comerciales como la de los microcontroladores están inclinadas hacia la filosofía RISC (Computadoras de Juego de Instrucciones Reducido). En estos procesadores el repertorio de instrucciones máquina es muy reducido y las instrucciones son simples y, generalmente, se ejecutan en un ciclo. La sencillez y rapidez de las instrucciones permiten optimizar el hardware y el software del procesador.

SISC: En los microcontroladores destinados a aplicaciones muy concretas, el juego de instrucciones, además de ser reducido, es específico, o sea, las instrucciones se adaptan a las necesidades de la aplicación prevista.

Esta filosofía se ha bautizado con el nombre de SISC (Computadoras de Juego de Instrucciones Especifico).

2.6.6 MEMORIA

En los microcontroladores la memoria de instrucciones y datos está integrada en el propio chip. Una parte debe ser no volátil, tipo ROM, y se destina a contener el programa de instrucciones que gobierna la aplicación.

Otra parte de memoria será tipo RAM, volátil, y se destina a guardar las variables y los datos.

- **ROM:** Es una memoria no volátil de sólo lectura cuyo contenido se graba durante la fabricación del chip.
- **OTP:** El microcontrolador contiene una memoria no volátil de sólo lectura programable una sola vez por el usuario
- **EPROM:** Los microcontroladores que disponen de este tipo de memoria pueden borrarse y grabarse muchas veces. La grabación se realiza mediante un grabador. Si, posteriormente, se desea borrar el contenido, disponen de una ventana de cristal en su superficie por la que se somete a la EPROM a rayos ultravioleta durante varios minutos.
- **EEPROM:** Se trata de memorias de sólo lectura, programables y borrables eléctricamente. Tanto la programación como el borrado, se realizan eléctricamente desde el propio grabador. No disponen de ventana de cristal en la superficie. Este tipo de memoria es relativamente lenta.
- **FLASH:** Se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar. A diferencia de la ROM, la memoria FLASH es programable en el circuito, es más rápida y de mayor densidad que la EEPROM.

Las memorias EEPROM y FLASH son muy útiles al permitir que los microcontroladores que las incorporan puedan ser reprogramados en circuito, es decir, sin tener que sacar el circuito integrado de la tarjeta.

2.6.7 PUERTOS DE ENTRADA Y SALIDA

La principal utilidad de los pines que posee la capsula que contiene un microcontrolador es soportar las líneas de entrada y salida que comunican al computador interno con los

periféricos exteriores. Según los controladores de periféricos que posea cada modelo de microcontrolador, las líneas de entrada y salida se destinan a proporcionar el soporte a las señales de entrada, salida y control

2.6.8 RELOJ PRINCIPAL

Todos los microcontroladores disponen de un circuito oscilador que genera una onda cuadrada de alta frecuencia, que configura los pulsos de reloj usados en la sincronización de todas las operaciones del sistema. Generalmente, el circuito de reloj está incorporado en el microcontrolador y sólo se necesitan unos pocos componentes exteriores para seleccionar y estabilizar la frecuencia de trabajo. Dichos componentes suelen consistir en un cristal de cuarzo junto a elementos pasivos o bien un resonador cerámico o una red RC. Aumentar la frecuencia de reloj supone disminuir el tiempo en que se ejecutan las instrucciones pero lleva aparejado un incremento del consumo de energía.

2.6.9 RECURSOS ESPECIALES DE LOS MICROCONTROLADORES

Cada fabricante oferta numerosas versiones de una arquitectura básica de microcontrolador. En algunas amplia las capacidades de las memorias, en otras incorporan nuevos recursos, en otras reduce las prestaciones al mínimo para aplicaciones muy simples, etc. La labor del diseñador es encontrar el modelo mínimo que satisfaga todos los requerimientos de su aplicación. De esta forma, minimizará el coste, el hardware y el software. Los principales recursos específicos que incorporan los microcontroladores son:

2.6.9.1 TEMPORIZADORES

Se emplean para controlar periodos de tiempo (temporizadores) y para llevar la cuenta de acontecimientos que suceden en el exterior (contadores). Para la medida de tiempos se carga un registro con el valor adecuado y a continuación dicho valor se va incrementando o decrementando al ritmo de los pulsos de reloj o algún múltiplo hasta que se desborde y llegue a 0, momento en el que se produce un aviso. Cuando se desean contar acontecimientos que se materializan por cambios de nivel o flancos en alguna de

los pines del microcontrolador, el mencionado registro se va incrementando o decrementando al ritmo de dichos pulsos.

2.6.9.2 PERRO GUARDIÁN O WATCHDOG

Cuando una computadora personal se bloquea por un fallo del software u otra causa, se pulsa el botón del reset y se reinicializa el sistema. Pero un microcontrolador funciona sin el control de un supervisor y de forma continuada las 24 horas del día. El Perro guardián consiste en un temporizador que, cuando se desborda y pasa por 0 provoca un reset automáticamente en el sistema. Si falla el programa o se bloquea, no se refrescará al perro guardián y al completar su temporización provocará el reset.

2.6.9.3 PROTECCIÓN ANTE FALLO DE ALIMENTACIÓN O BROWNOUT

Se trata de un circuito que resetea al microcontrolador cuando el voltaje de alimentación (VDD) es inferior a un voltaje mínimo ("*brownout*"). Mientras el voltaje de alimentación sea inferior al de *brownout* el dispositivo se mantiene reseteado, comenzando a funcionar normalmente cuando sobrepasa dicho valor.

2.6.9.4 ESTADO DE REPOSO O DE BAJO CONSUMO

Para ahorrar energía, (factor clave en los aparatos portátiles), los microcontroladores disponen de una instrucción especial, que les pasa al estado de reposo o de bajo consumo, en el cual los requerimientos de potencia son mínimos. En dicho estado se detiene el reloj principal y se detienen sus circuitos asociados, quedando sumido en un profundo sueño el microcontrolador. Al activarse una interrupción ocasionada por el acontecimiento esperado, el microcontrolador se despierta y reanuda su trabajo.

2.6.9.5 CONVERSOR ANALÓGICO A DIGITAL (ADC)

Los microcontroladores que incorporan un Conversor A/D (Analógico a Digital) pueden procesar señales analógicas, tan abundantes en las aplicaciones. Suelen disponer de un multiplexor que permite aplicar a la entrada del ADC diversas señales analógicas desde los pines del circuito integrado.

2.6.9.6 CONVERSOR DIGITAL A ANALÓGICO (DAC)

Transforma los datos digitales en su correspondiente señal analógica que saca al exterior por una de sus pines.

2.6.9.7 COMPARADOR ANALÓGICO

Algunos modelos de microcontroladores disponen internamente de un comparador.

La salida del comparador proporciona un nivel lógico 1 ó 0 según una señal sea mayor o menor que la otra. También hay modelos de microcontroladores con un módulo de tensión de referencia que proporciona diversas tensiones de referencia que se pueden aplicar en los comparadores.

2.6.9.8 MODULADOR POR ANCHO DE PULSO (PWM)

Son circuitos que proporcionan en su salida pulsos de anchura variable, que se ofrecen al exterior a través de los pines del encapsulado.

2.6.9.9 PUERTOS DE COMUNICACIÓN

Con objeto de dotar al microcontrolador de la posibilidad de comunicarse con otros dispositivos externos, otros buses de microprocesadores, buses de sistemas, buses de redes y poder adaptarlos con otros elementos bajo otras normas y protocolos. Algunos modelos disponen de recursos que permiten directamente esta tarea, entre los que destacan:

- **UART:** adaptador de comunicación serie asíncrona.
- **USART:** adaptador de comunicación serie síncrona y asíncrona Puerta paralela esclava para poder conectarse con los buses de otros microprocesadores.
- **USB:** (Universal Serial Bus), que es un moderno bus serie para las PC.
- **BUS I2C:** que es un interfaz serie de dos hilos desarrollado por Philips.
- **CAN** (*Controller Area Network*), para permitir la adaptación con redes de conexionado multiplexado desarrollado conjuntamente por Bosch e Intel para el cableado de dispositivos en automóviles.

2.7 INTERFAZ DE USUARIO

La interfaz de usuario es el medio con que el usuario puede comunicarse con una máquina, equipo, computadora o dispositivo, y comprende todos los puntos de contacto entre el usuario y el equipo. Normalmente suelen ser fáciles de entender y fáciles de accionar, aunque en el ámbito de la informática es preferible referir sea que suelen ser "amigables e intuitivos" por qué es complejo y subjetivo decir "fácil".

2.7.1 DEFINICIÓN

Las interfaces básicas de usuario son aquellas que incluyen elementos como menús, ventanas, contenido gráfico, cursor, los beeps y algunos otros sonidos que la computadora hace, y en general, todos aquellos canales por los cuales se permite la comunicación entre el ser humano y la computadora. La mejor interacción humano-máquina a través de una adecuada interfaz (de usuario), que le brinde tanto comodidad, como eficiencia.

2.7.2 FUNCIONES PRINCIPALES

Las funciones principales son las siguientes:

- Puesta en marcha y apagado.
- Control de las funciones manipulables del equipo.
- Manipulación de archivos y directorios.
- Herramientas de desarrollo de aplicaciones.
- Comunicación con otros sistemas.
- Configuración de la propia interfaz y entorno.
- Intercambio de datos entre aplicaciones.
- Control de acceso.
- Sistema de ayuda interactivo.

2.7.3 TIPOS DE INTERFACES DE USUARIO

Las interfaces de usuario se pueden distinguir básicamente tres tipos:

- **CLI:** Command-line, interface o interfaz de línea de comandos codificada estrictita
- **GUI:** GRAPHICAL User, interface o interfaz gráfica de usuario. Metaforica. Explorativa
- **NUI:** Natural User, Interface o interfaz natural del usuario. Diestra. Intuitiva.

Figura 2.0.1 evolución de las interfaces de usuario.
Fuente: https://es.wikipedia.org/wiki/archivo:CLI-GUI-NUI,_evoluci%C3%B3n_de_interfaces_de_usuario.png

2.8 VISUAL STUDIO

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta múltiples lenguajes de programación, tales como C++, C#, Visual Basic .NET, F#, Java, Python, Ruby y PHP, al igual que entornos de desarrollo web, como ASP.NET MVC, Django, etc., a lo cual hay que sumarle las nuevas capacidades online bajo Windows Azure en forma del editor Monaco.

Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así, se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos y consolas, entre otros.

2.8.1 VERSIONES

A partir de la versión 2005, Microsoft ofrece gratuitamente las Ediciones Express, que son versiones básicas separadas por lenguajes de programación o plataforma enfocadas a estudiantes y programación amateur. Dichas ediciones son:

- Visual Basic Express Edition
- Visual C# Express Edition
- Visual C++ Express Edition
- Visual Web Developer Express Edition (para programar en ASP.NET)
- Visual F# (Apareció en Visual Studio 2010, es parecido al J#)*
- Windows Azure SDK

2.8.1.1 VISUAL STUDIO.NET 2002

En esta versión se produjo un cambio sustancial, puesto que supuso la introducción de la plataforma .NET de Microsoft. .NET es una plataforma de ejecución intermedia multilenguaje, de forma que los programas desarrollados en .NET no se compilan en lenguaje máquina, sino en un lenguaje intermedio.

Visual Studio .NET 2002 supuso también la introducción del lenguaje C#, un lenguaje nuevo diseñado específicamente para la plataforma .NET, basado en C++ y Java. Se presentó también el lenguaje J# (sucesor de J++), el cual, en lugar de ejecutarse en una máquina virtual Java, se ejecuta únicamente en el framework .NET. El lenguaje Visual Basic fue remodelado completamente y evolucionó para adaptarse a las nuevas características de la plataforma .NET, haciéndolo mucho más versátil y dotándolo con muchas características de las que carecía. Todos los lenguajes se unifican en un único entorno. La interfaz se mejora notablemente en esta versión, siendo más limpia y personalizable.

Visual Studio .NET puede usarse para crear programas basados en Windows (usando Windows Forms en vez de COM), aplicaciones y sitios web (ASP.NET y servicios web), y dispositivos móviles (usando el .NET Compact Framework).

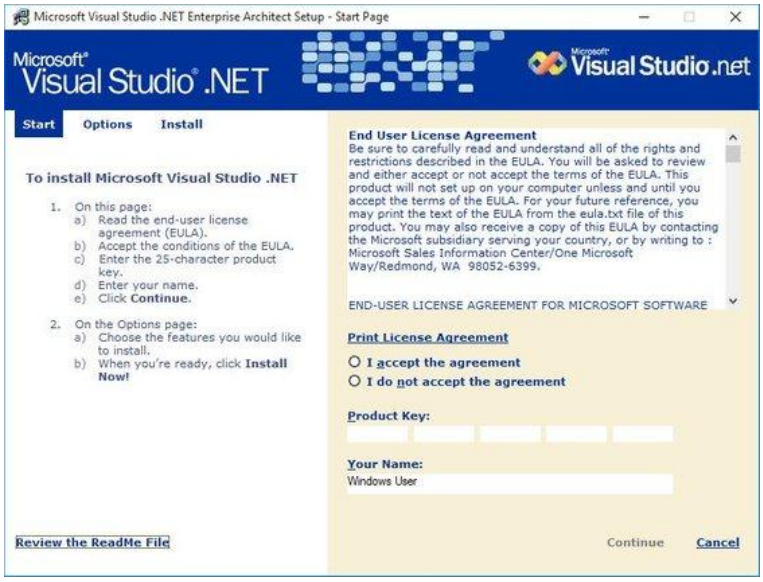

Figura 2.0.2 Visual Studio.NET 2002.
https://www.instructables.com/id/Install-Visual-Studio-NET-2002-in-64-bit-Windows/

2.8.1.2 VISUAL STUDIO.NET 2003

Visual Studio .NET 2003 supone una actualización menor de Visual Studio .NET. Se actualiza el .NET Framework a la versión 1.1

Visual Studio 2003 se lanza en cuatro ediciones: Academic, Professional, Enterprise Developer y Enterprise Architect.

Microsoft lanzó el Service Pack 1 para Visual Studio 2003 el 13 de septiembre de 2006.

La versión interna de Visual Studio .NET 2003 es la 7.1, aunque el formato del archivo que emplea es el de la 8.0.

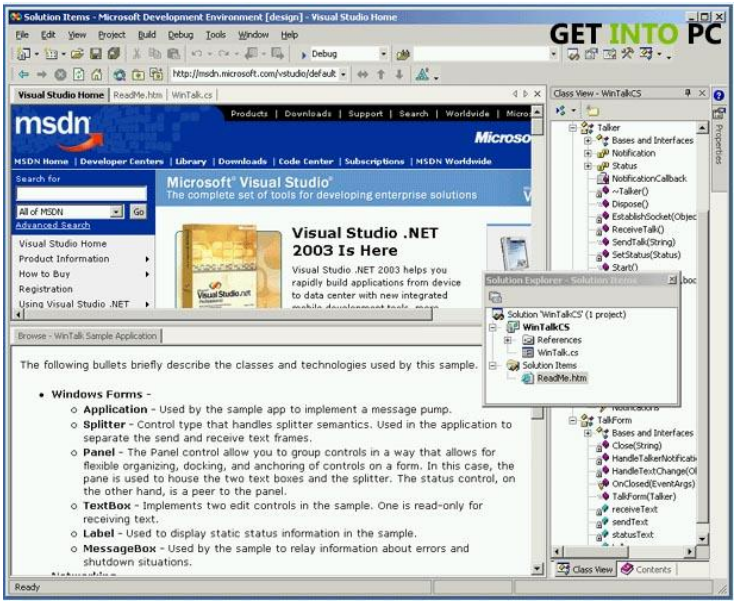

Figura 2.0.3 Visual Studio.NET 2003.
http://getintopc.com/softwares/development/visual-studio-net-2003-free-download/

Es compatible solo con Windows XP, Windows Server 2003 o anteriores

2.8.1.3 VISUAL STUDIO.NET 2005

Visual Studio 2005 se empezó a comercializar a través de Internet a partir del 4 de octubre de 2005, y la versión en inglés llegó a los comercios a finales del mes de octubre. En castellano no salió hasta el 4 de febrero de 2006. Microsoft eliminó la coletilla .NET de su nombre, pero eso no indica que se alejara de la plataforma .NET, de la cual se incluyó la versión 2.0.

La actualización más importante que recibieron los lenguajes de programación fue la inclusión de tipos genéricos, similares en muchos aspectos a las plantillas de C++. Con esto se consigue encontrar muchos más errores en la compilación en vez de en tiempo de ejecución, incitando a usar comprobaciones estrictas en áreas donde antes no era posible. C++ tiene una actualización similar con la adición de C++/CLI como sustituto de C# manejado.

Visual Studio 2005 tiene varias ediciones radicalmente distintas entre sí: Express, Standard, Professional, Tools for Office y cinco ediciones Visual Studio Team System. Estas últimas se proporcionaban conjuntamente con suscripciones a MSDN cubriendo

los cuatro principales roles de la programación: Architects, Software Developers, Testers y Database Professionals. La funcionalidad combinada de las cuatro ediciones Team System se ofrecía como la edición Team Suite. Por otra parte, Tools for the Microsoft Office System está diseñada para extender la funcionalidad a Microsoft Office.

Se lanzó el Service Pack 1 para Visual Studio 2005 el 14 de diciembre de 2006.

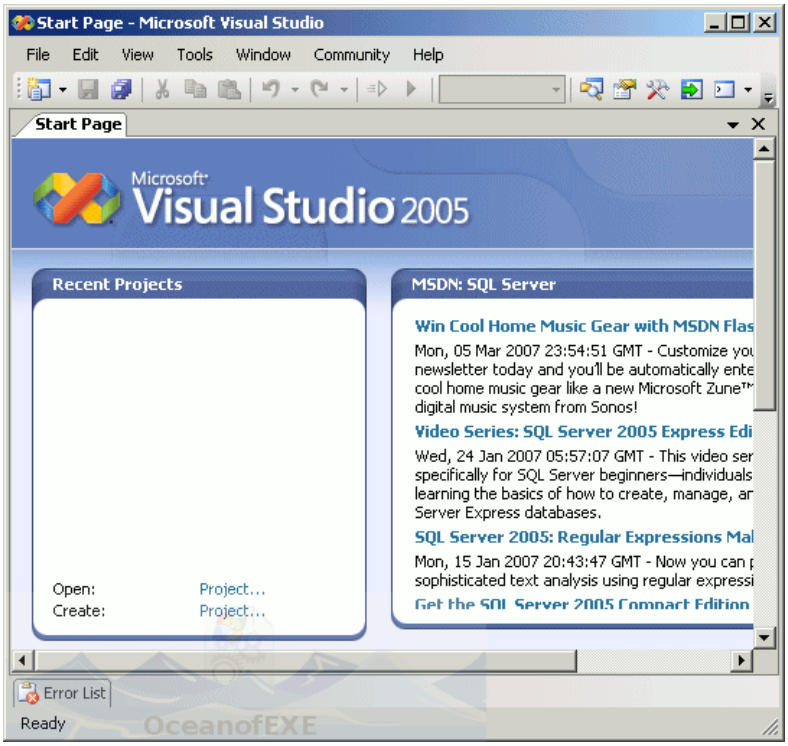

Figura 2.0.4 Visual Studio.NET 2005.
http://oceanofexe.com/visual-studio-2005-download-free/

La versión interna de Visual Studio 2005 es la 8.0, mientras que el formato del archivo que emplea es el de la 9.0.

2.8.1.4 VISUAL STUDIO.NET 2008

Permite trabajar con los Frameworks:

- .NET Framework 2.0
- .NET Framework 3.0

- .NET Framework 3.5

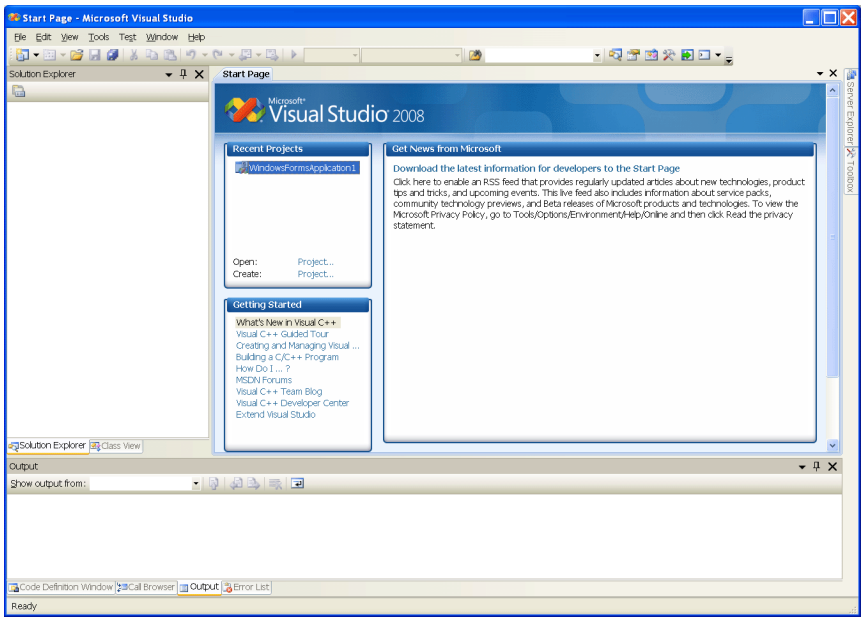

Figura 2.0.5 Visual Studio.NET 2008.
http://forum.codecall.net/topic/39727-visual-studio-2008-c-hello-world-tutorial/

2.8.1.5 VISUAL STUDIO.NET 2010

El IDE se rediseña para una mejor legibilidad. Se han eliminado gradientes y líneas innecesarias para hacer más simple su uso.

Las ventanas de documentos tales como el editor de código y la ventana de la vista diseño ahora pueden colocarse fuera de la ventana IDE. Por ejemplo, puede arrastrar el editor de código en el IDE de modo que se puede ver la ventana de la vista de diseño al lado.

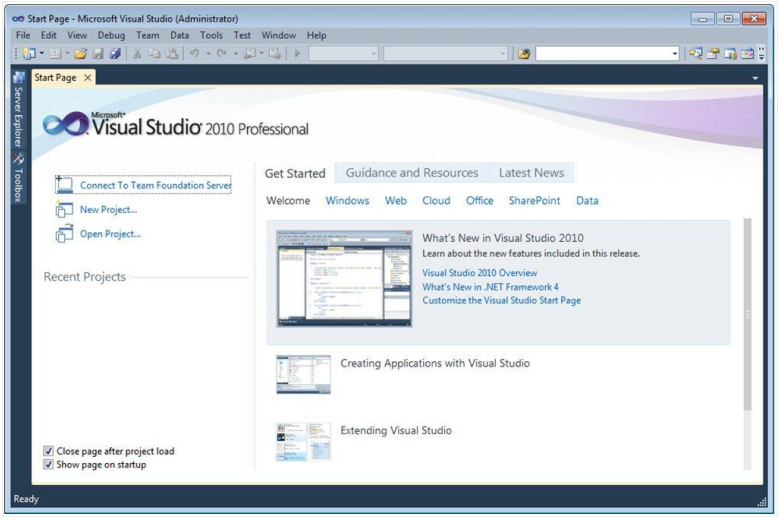

Figura 2.0.6 Visual Studio.NET 2010.
https://www.malavida.com/es/soft/visual-studio-2010/#gref

Permite trabajar con los Frameworks:

- .NET Framework 2.0
- .NET Framework 3.0
- .NET Framework 3.5
- .NET Framework 4.0

2.8.1.6 OTRAS VERSIONES

Con los años se potenciando más con la versión VS2012 se comenzó a mostrar en una interfaz metro, en la versión VS2015 se cuentan con la incorporación de Xamarin, el cual permite diseñar aplicaciones móviles en Windows 10 Mobile, Android y IOS.

En la versión VS2017 se notan muchas más herramientas como la compilación de programas para sistemas basados en Linux y la integración de más lenguajes de programación como Python.

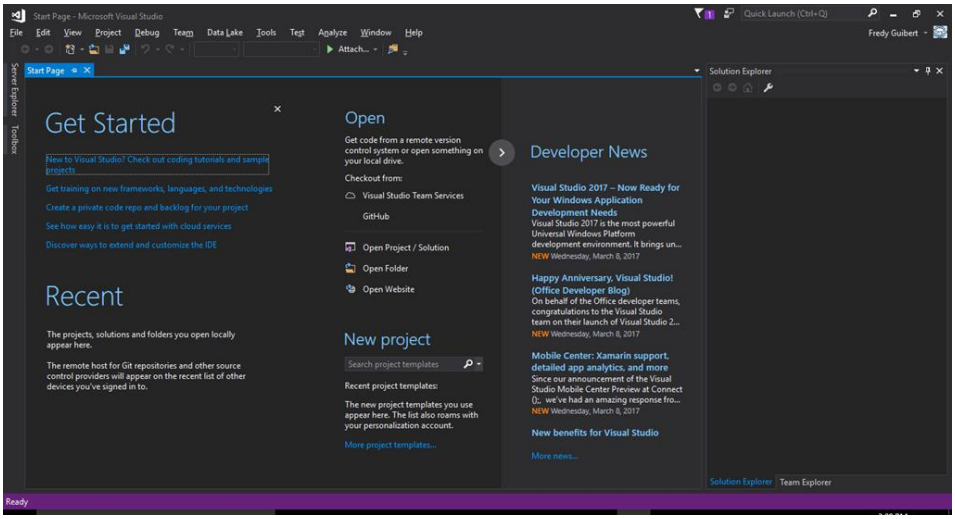

Figura 2.0.7 Visual Studio.NET 2017.
http://fredyfx.com/post/instalar-visual-studio-2017-iso-offline/

2.8.2 VISUAL BASIC.NET

Visual Basic .NET (VB.NET) es un lenguaje de programación orientado a objetos que se puede considerar una evolución de Visual Basic implementada sobre el framework .NET.

Su introducción resultó muy controvertida, ya que debido a cambios significativos en el lenguaje VB.NET no es retro compatible con Visual Basic, pero el manejo de las instrucciones es similar a versiones anteriores de Visual Basic, facilitando así el desarrollo de aplicaciones más avanzadas con herramientas modernas. Para mantener eficacia en el desarrollo de las aplicaciones. La gran mayoría de programadores de VB.NET utilizan el entorno de desarrollo integrado Microsoft Visual Studio en alguna de sus versiones (desde el primer Visual Studio .NET hasta Visual Studio .NET 2017, que es la última versión de Visual Studio para la plataforma .NET), aunque existen otras alternativas, como SharpDevelop (que además es libre).

Al igual que con todos los lenguajes de programación basados en .NET, los programas escritos en VB .NET requieren el Framework .NET o Mono para ejecutarse.

2.8.3 RELACIÓN CON VISUAL BASIC

Si Visual Basic .NET debe considerarse una mera versión de Visual Basic, o si debe considerarse como un nuevo lenguaje de programación es un tema que ha traído mucha discusión, y que aún la trae.

La sintaxis básica es prácticamente la misma entre VB y VB.NET, con la excepción de los añadidos para soportar nuevas características como el control estructurado de excepciones, la programación orientada a objetos, o los Genéricos.

Figura 2.0.8 Visual Basic 6.0.
http://www.beaugauge.com/en/gauge-activex-control/support/use-beaugauge-activex-control-in-visual-basic-6-project.htm

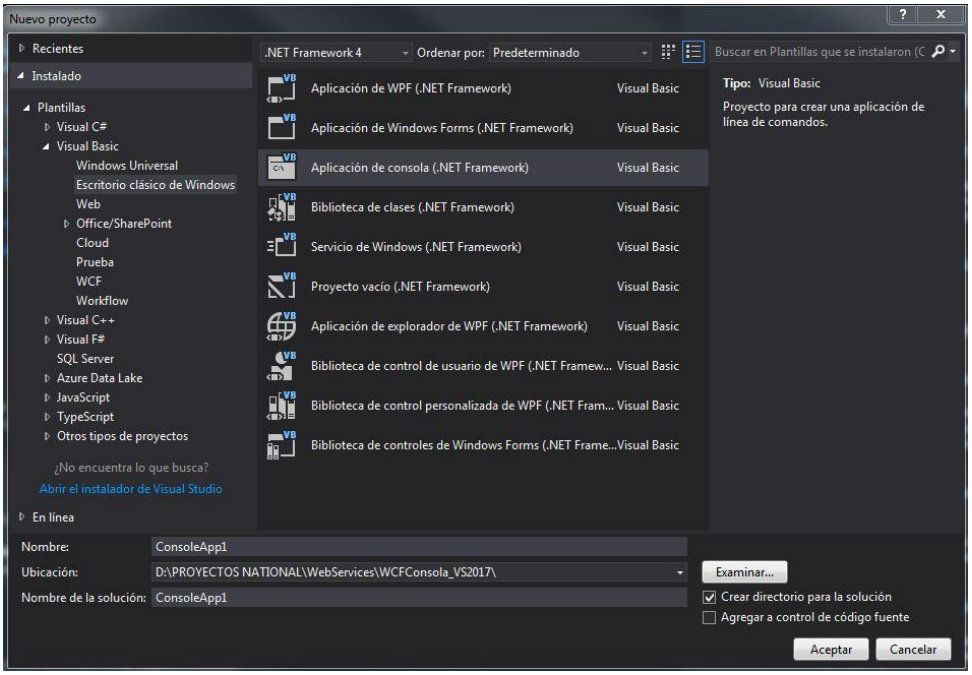

Figura 2.0.9 Visual Basic.NET.
https://www.codeproject.com/Articles/1184324/HOW-CREATE-A-WCF-WebService-in-VB-NET

Las diferencias entre VB y VB.NET son profundas, sobre todo en cuanto a metodología de programación y bibliotecas, pero ambos lenguajes siguen manteniendo un gran parecido, cosa que facilita notablemente el paso de VB a VB.NET.

2.8.4 APLICACIÓN WINDOWS FORM

Windows Forms (o **formularios Windows**) es el nombre dado a la interfaz de programación de aplicación gráfica (API) que se incluye como parte de Microsoft .NET Framework, que proporciona acceso a los elementos de la interfaz de Microsoft Windows nativas envolviendo la API de Windows existente en código administrado.

Al igual que Abstract Windows Toolkit (AWT), la API de Java equivalente, Windows Forms era una forma temprana y fácil de proporcionar componentes de la interfaz gráfica de usuario para el .NET Framework. Windows Forms, está construido sobre la API de Windows existente y algunos controles sólo envuelven componentes subyacentes de Windows.

2.9 PIC C CCS

2.9.1 INTRODUCCIÓN

Si queremos realizar la programación de los microcontroladores PIC en un lenguaje como el C, es preciso utilizar un compilador de C.

Dicho compilador nos genera ficheros en formato Intel-hexadedimal, que es el necesario para programar (utilizando un programador de PIC) un microcontrolador de 6, 8, 18 ó 40 patillas.

El compilador de C que vamos a utilizar es el PCW de la casa CCS Inc. A su vez, el compilador lo integraremos en un entorno de desarrollo integrado (IDE) que nos va a permitir desarrollar todas y cada una de las fases que se compone un proyecto, desde la edición hasta la compilación pasando por la depuración de errores. La última fase, a excepción de la depuración y retoques hardware finales, será programar el PIC.

Al igual que el compilador de Turbo C, éste "traduce" el código C del archivo fuente (.C) a lenguaje máquina para los microcontroladores PIC, generando así un archivo en formato hexadecimal (.HEX)

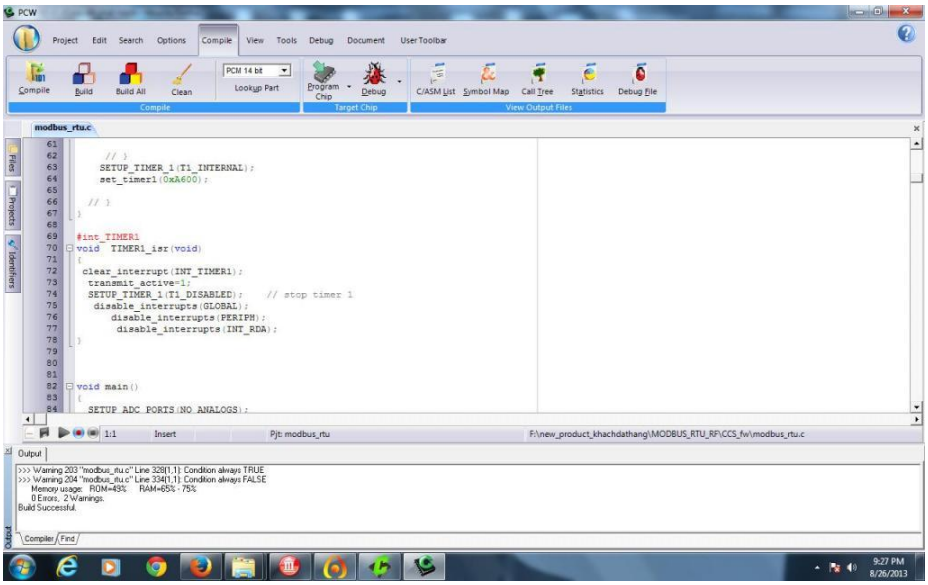

Figura 2.0.10 PIC C Entorno de desarrollo.
http://www.dientuvietnam.net/forums/forum/

2.9.2 PROGRAMAS DE UTILIDAD

• SIO

SIO (Serial Input Output) es un simple programa "terminal no inteligente" que puede ejecutarse desde el DOS para realizar entradas y salidas sobre un puerto serie. SIO es útil ya que muestra todos los caracteres entrantes, excepto los no imprimibles que mostrará su código hexadecimal en rojo.

• PICCHIPS

PICCHIPS es un programa de utilidad que lee la base de datos de un dispositivo. El compilador utiliza esta base de datos para determinar las características específicas del dispositivo durante la compilación. Al ejecutar el programa sin ningún parámetro, listará todos los dispositivos (PIC) disponibles.

Si especificamos un dispositivo como parámetro por ejemplo pic16c84, es decir, escribimos picchips pic16c84, obtenemos información detallada sobre este dispositivo

• CHIPEDIT

ChipEdit es una utilidad de Windows (sólo para PCW) que permite editar la base de datos de un dispositivo. Con esta utilidad podemos agregar dispositivos, modificarlos o eliminarlos de la base de datos. Para agregar un dispositivo, seleccionar de la lista otro equivalente, de características similares, y pulsar el botón ADD. Para editar o borrar un dispositivo, seleccionarlo y pulsar el botón EDIT o DELETE.

• CONVERT

Convert es una utilidad de Windows (PCW sólo) que permite realizar conversiones de un tipo de datos a otros tipos. Por ejemplo, de decimal en Punto Flotante a Hexadecimal de 4 byte. La utilidad abre una ventana pequeña para realizar las conversiones y puede permanecer activa durante una sesión con PCW o con MPLAB. Esto puede ser útil durante el proceso de depuración de un programa.

2.9.3 OPERADORES Y EXPRESIONES

• Operadores de asignación.

Una expresión de asignación tradicional es de la forma

expr1 = expr1 operador expr2 , es decir, i = i + 5. Esta expresión se puede representar por otra forma más corta: expr1 operador= expr2 siguiendo con el mismo ejemplo i += 5.

Es en las expresiones complejas, y no en una tan simple como la del ejemplo, donde se puede apreciar la conveniencia de usar esta notación.

• Operadores relacionales

Su misión es comparar dos operando y dar un resultado entero: 1 (verdadero); 0 (falso).

• Operadores lógicos

Al igual que los operadores relacionales, éstos devuelven 1 (verdadero), 0 (falso) tras la evaluación de sus operandos.

• Operadores de manejo de bits

Estos operadores permiten actuar sobre los operandos a nivel de bits y sólo pueden ser de tipo entero (incluyendo el tipo char)

• Operadores de dirección (&) e indirección (*)

Los operadores se utilizan para trabajar con punteros. El lenguaje C está muy influenciado por el uso de punteros. Un puntero es una variable que contiene la dirección de una variable o de una función, es decir, es una variable que apunta a otra variable. Los punteros permiten la manipulación indirecta de datos y códigos. Disponemos de dos operadores

2.9.4 DIRECTIVAS DEL PROCESADOR

Todas las directivas del pre-procesador comienzan con el caracter # seguido por un comando específico. Algunas de estas directivas son extensiones del C estandar. El C proporciona una directiva del preprocesador, que los compiladores aceptan, y que permite ignorar o actuar sobre los datos que siguen. Nuestro compilador admite cualquier directiva del pre-procesador que comience con PRAGMA, lo que nos asegura la compatibilidad con otros compiladores.

2.9.5 CONTROL DE MEMORIA

• #ASM #ENDASM

Las líneas entre #ASM y #ENDASM se tratan como código ensamblador. La variable predefinida _RETURN_ puede utilizarse para asignar un valor de retorno a la función desde el código en ensamblador. Tener presente que cualquier código C después de ^I^B #ENDASM ^i^b y antes del final de la función puede falsear el valor de retorno

• #BYTE Identificador = X

Esta directiva creará un identificador "id" que puede utilizarse como cualquier NT (un byte). El identificador referenciará un objeto en la posición de memoria x, donde x puede ser una constante u otro identificador. Si x es otro identificador, entonces éste estará localizado en la misma dirección que el identificador "id".

• #RESERVE

Permite reservar posiciones de la RAM para uso del compilador.

#RESERVE debe aparecer después de la directiva #DEVICE, de lo contrario no tendrá efecto.

• #ROM

Esta directiva permite insertar datos en el archivo .HEX. En particular, se puede usar para programar la EEPROM de datos de la serie 84 de PIC**.**

2.9.6 CONTROL DE COMPILADOR

• #CASE

Hace que el compilador diferencie entre mayúsculas y minúsculas. Por defecto el compilador hace esta distinción.

• #OPT n

Esta directiva sólo se usa con el paquete PCW y, establece el nivel de optimización. Se aplica al programa entero y puede aparecer en cualquier parte del archivo fuente. El nivel de optimización 5 es el nivel para los compiladores DOS. El valor por defecto para el compilador PCW es 9 que proporciona una optimización total.

• #PRIORITY

Esta directiva se usa para establecer la prioridad de las interrupciones.

Los elementos de mayor prioridad van primero.

2.9.7 IDENTIFICADORES PREDEFINIDOS

• _ _DATE_ _

Este identificador del pre-procesador contiene la fecha actual (en tiempo de compilación) en el formato siguiente: "30-SEP-98"

• _ _PCM_ _

Se utiliza para determinar si es el compilador PCM el que está haciendo la compilación.

2.9.8 DIRECTIVAS DEL C ESTÁNDAR

• #DEFINE Identificador CADENA

Se utiliza simplemente para reemplazar el IDentificador (ID) con CADENA

• #IF expresión_constante #ELSE #ENDIF

El pre-procesador evalúa la expresión_constante y si es distinta de cero procesará las líneas hasta el #ELSE -que es opcional- o en su defecto hasta el #ENDIF.

• #IFDEF #ELSE #ENDIF

Esta directiva actúa como el #IF sólo que aquí el pre-procesador simplemente comprueba que reconoce el id especificado (creado con un #DEFINE). Nótese que #IFDEF verifica si se definió un id pero #IFNDEF comprueba que no está definido el id.

• #ERROR

Esta directiva para el compilador y emite el mensaje que se incluye a continuación (en la misma línea)de la propia directiva. El mensaje puede incluir macros. También puede utilizarse para alertar al usuario de una situación anómala en tiempo de compilación.

2.9.9 CALIFICADORES DE FUNCIÓN

• #INLINE

Esta directiva le dice al compilador que el procedimiento que sigue a la directiva será llevado a cabo EN LÍNEA. Esto causará una copia del código que será puesto en cualquier parte donde se llame al procedimiento. Esto es útil para ahorrar espacio de la pila (stack) y aumentar la velocidad.

Sin esta directiva es el compilador quien decidirá cuándo es mejor hacer los procedimientos EN LÍNEA.

• #INT_GLOBAL función

La función que sigue a esta directiva reemplaza al distribuidor de interrupciones del compilador; dicha función toma el control de las interrupciones y el compilador no salva ningún registro. Normalmente no es necesario usar esto y debe tratarse con gran prudencia.

• #INT_xxx función_de_interrupción

Estas directivas especifican que la función que le sigue es una función de interrupción. Las funciones de interrupción no pueden tener ningún parámetro. Como es natural, no todas las directivas pueden usarse con todos los dispositivos.

2.9.10 LIBRERÍAS INCORPORADAS

• #USE DELAY (CLOCK=frecuencia)

Esta directiva indica al compilador la frecuencia del procesador, en ciclos por segundo, a la vez que habilita el uso de las funciones DELAY_MS() y DELAY_US(). Opcionalmente podemos usar la función restart_WDT() para que el compilador reinicie el WDT durante el retardo.

• #USE FAST_IO (puerto)

Esta directiva afecta al código que el compilador generará para las instrucciones de entrada y salida. Este método rápido de hacer I/O ocasiona que el compilador realice I/O sin programar el registro de dirección. El puerto puede ser A-G.

CAPITULO III

DESARROLLO DE LA INVESTIGACIÓN

3.1 METODOLOGÍA

En esta etapa del proyecto se hará el análisis específico del diseño dando una solución concreta a los problemas planteados, con el objetivo de tener un sistema de timbre automático para la unidad educativa "Pedo Poveda". Para responder a los problemas se usarán recursos como ser: un microcontrolador, módulo Bluetooth HC-05, un display de cristal líquido y una PC par la configuración final del sistema.

3.2 DIAGRAMA BÁSICO DEL SISTEMA

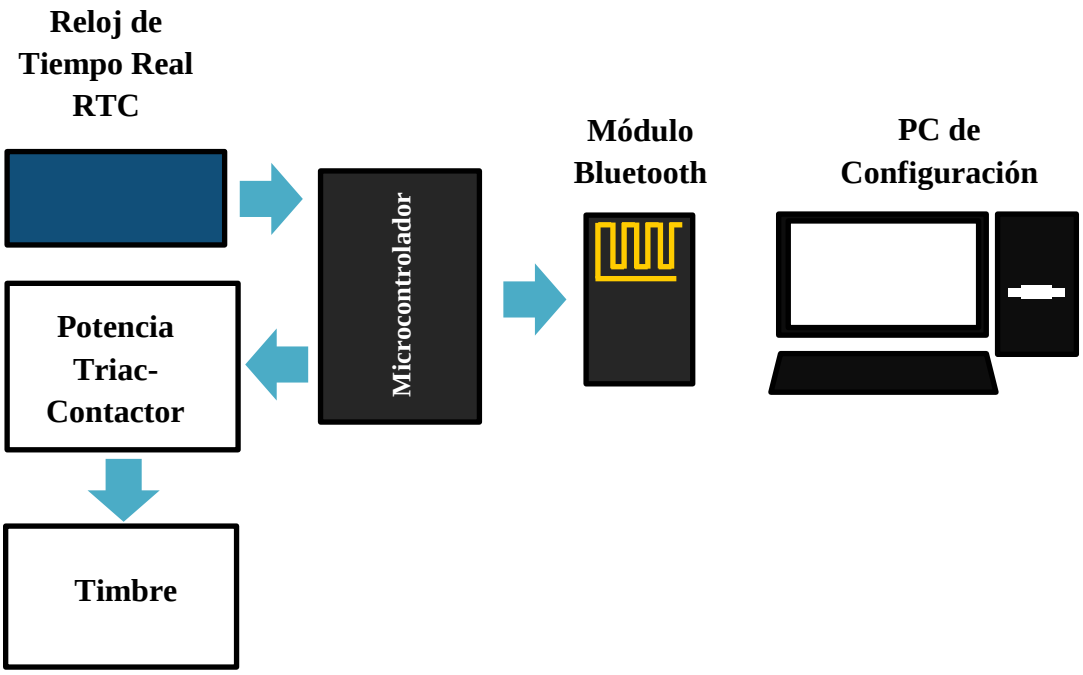

Figura 3.0.1 Diagrama de bloques básico del sistema de monitoreo.
Fuente: Diseño propio

3.2.1 DESCRIPCIÓN DEL DIAGRAMA DE BLOQUES

Se cuenta con un rtc el cual enviara la fecha y la hora al microcontrolador, este a su vez interpretara los días de lunes a sábado teniendo almacenada en su memoria el registro para cada alarma el cual se enviaran por la terminal grafica para establecer los horarios, una vez que se dispare la alarma el microcontrolador enviara un pulso a un triac el cual estará haciendo de llave al contactor que estará conectado a timbre del colegio.

BIBLIOGRAFÍA

Eduardo García Breijo (2008). "Compilador C CCS y Simulador Proteus para microcontroladores PIC". Editorial: Alfaomega.

Andrés Cánovas López (2003). "Manual de usuario del Compilador PCW de CCS". Edición en formato digital: Víctor Dorado.

DIRECCIONES WEB

Estándar de comunicaciones RS-232C (O cómo funciona un puerto serie). 3, Agosto, 2018. De euskalnet.com Sitio Web: www.euskalnet.com/shizuka/underc.html.

Universal Asynchronous Receiver-Transmitter. 27, Agosto 2018 https://es.wikipedia.org/wiki/Universal_Asynchronous_Receiver-Transmitter.

Conjunto de comandos Hayes. 27, Agosto,2018. https://es.wikipedia.org/wiki/Conjunto_de_comandos_Hayes.

HC-05. 27, Agosto, 2018. http://wiki.jmoon.co/hc-05-bluetooth/