UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA: INGENIERÍA ELECTRÓNICA

Tesis previa a la obtención del título: INGENIERO ELECTRÓNICO

TEMA:

DISEÑO Y CONSTRUCCIÓN DE UN MÓDULO DE ENTRENAMIENTO BASADO EN EL MICROCONTROLADOR XMEGA DE ATMEL PARA LOS LABORATORIOS DE SISTEMAS MICROPROCESADOS DE LA UNIVERSIDAD POLITÉCNICA SALESIANA.

AUTORES:

NINA GABRIELA CHICAIZA ENRIQUEZ

FRANCISCO JAVIER REYES ALMEIDA

DIRECTOR:

LUIS OÑATE CADENA

Quito, junio de 2013

DECLARACIÓN

Nosotros, Nina Gabriela Chicaiza Enríquez y Francisco Javier Reyes Almeida; autorizamos a la Universidad Politécnica Salesiana la publicación total o parcial de este trabajo de grado y su reproducción sin fines de lucro.

Además declaramos que los conceptos y análisis desarrollados y las conclusiones del presente trabajo son de exclusiva responsabilidad de los autores.

Nina Gabriela Chicaiza Enríquez	Francisco Javier Reyes Almeida
CC: 172097597-6	CC: 172291394-2

DEDICATORIA

Dedicamos este proyecto a nuestros queridos padres, su paciencia, su constancia y su capacidad de enseñar no tan solo con palabras sino con el ejemplo, fue el impulso que nos llevó a culminar esta etapa, por su confianza depositada, por el amor incondicional, este logro a ellos.

A nuestros amados hermanos que día a día encendía una llama tan necesaria para sobrellevar cada adversidad, en este proceso fueron y serán nuestra razón de jamás decaer. Ahora ustedes son quienes nos llevan a terminar este proyecto.

A nuestras abuelitas por todo su hermoso tiempo y cariño.

A todos ellos este proyecto pues es solo la gran unión de todas esas partes la que nos ha permitido llegar hasta el tramo del camino en el que nos encontramos.

Gracias por permitirnos estar a su lado.

Francisco Javier. Nina Gabriela

AGRADECIMIENTOS

A nuestra querida universidad, compañeros, amigos con quienes hicimos la familia educativa, un reconocimiento por ser ese equipo de jóvenes que apoyados unos y otros estamos alcanzando lo que es nuestro ideal, convertirnos en profesionales útiles a la sociedad, estudiantes que llevaremos muy en alto el nombre de nuestra prestigiosa institución universitaria.

A nuestros maestros, parte fundamental en la formación profesional por su dedicación, su profesionalismo, y sobre todo su paciencia, A quien tuvimos como asesor de proyecto de graduación, al Ing. Luis Oñate, por sus enseñanzas en el aula y seguramente en la práctica profesional ya fuera de ellas. Al ing. Rafael Jaya, por su dedicación y sugerencias para culminar con éxito este proyecto de graduación.

Finalmente a quien nos dio la oportunidad de la vida, a Dios por enseñarnos lo valioso de la misma.

Francisco Javier. Nina Gabriela

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ANEXOS

RESUMEN

El presente proyecto de titulación propone la investigación y creación de una tarjeta de entrenamiento, la cual tiene como objetivo central servir de plataforma didáctica en el aprendizaje de la materia sistemas microprocesados, se presenta hacia el docente como una ayuda pedagógica, además en este proyecto se detalla toda la información necesaria para la utilización del microcontrolador XMega como nuevo recurso de enseñanza, se entrega un módulo de pruebas con todos los elementos necesarios a utilizar en las prácticas de laboratorio de sistemas microprocesados, estos materiales se basan en los módulos de enseñanza vigentes para la materia de sistemas microprocesados, también se deja por escrito un grupo de prácticas que ayudarán al estudiante a desenvolverse de manera efectiva en el uso del microcontrolador de Atmel, el cual se utiliza en conjunto con la tarjeta de entrenamiento.

ABSTRACT

This project propose an research and a construction of a training development kit, the main purpose is to serve like a didactic platform that will permit a best teach way in microprocessor systems schedule, at the same time it is present to teachers as a pedagogic help, also in this project its detailed all the needed information to use of a XMega microcontroller as a new technologic resource to teach, it is given a complete training development kit with all the necessary hardware in all microprocessor system laboratories practices, all the hardware were based in the taught modules used at the Salesian Polytechnic University, also it have been build an entire document with lab practices, this document will allow to develop many student's skills about Atmel microcontrollers, this is used altogether with the training development kit.

INTRODUCCIÓN

En este escrito se detallarán las distintas etapas tanto teóricas como prácticas por las cuales está compuesto el desarrollo del presente proyecto de investigación cuyo tema es el "Diseño y construcción de un módulo de entrenamiento basado en el microcontrolador XMega de Atmel para los laboratorios de sistemas microprocesados de la Universidad Politécnica Salesiana" con lo que se busca comprender de una mejor manera cada parte que lo compone:

En el capítulo 1 se deja claro la necesidad de la que surge la elaboración de este proyecto, analizando el problema que se desea resolver al elaborarlo, para lo cual se plantea la respectiva hipótesis y tesis mediante las cuales se llega a proponer el objetivo principal, los objetivos específicos y alcances que se busca cumplir durante la elaboración de este proyecto, el cual se desarrolla según la metodología de la investigación presente en este capítulo.

El capítulo 2 consta del estado del arte en el cual se revisa toda la información con la que se cuenta para empezar con el diseño y construcción de una tarjeta de entrenamiento y el software desarrollado usando un microcontrolador de Atmel y software Bascom, además se realiza una revisión de las técnicas existentes para poder elaborar una tarjeta electrónica.

El capítulo 3 trata sobre el diseño, desarrollo e implementación de la tarjeta de entrenamiento, se detallan las ventajas con las que cuenta al momento de su utilización, también se hace una introducción a cada una de los puertos con los que cuenta el micro y sus prestaciones de manera general, se describen de manera particular cada módulo que compone la tarjeta de entrenamiento, el diseño que tiene, la interconexión de los componentes y los puertos con los que cuentan. En este capítulo se realiza también el análisis de costos de producción, empezando por los componentes, el diseño y la elaboración de los programas a utilizar con la tarjeta.

El capítulo 4 consta del análisis de resultados, en el mismo se realiza pruebas de cada práctica presentada y se toma constancia de su correcto funcionamiento, se realiza una presentación pública para un grupo de estudiantes y se formulan preguntas sobre el uso y funcionamiento del módulo de prácticas.

Finalmente se dan las conclusiones y recomendaciones que se pueden extraer de este proyecto investigativo.

Se sustenta con referencias bibliográficas que sustenta la base teórica del presente proyecto.

En los anexos se pueden encontrar las prácticas desarrolladas en el software que se utilizó para programar el microcontrolador, las características principales de cada práctica, la descripción del uso del programa detallado por líneas de programación.

El microcontrolador que se utiliza en este proyecto es de Atmel, de la familia de los XMega, un microcontrolador de gama alta el cual entrega prestaciones acorde a las necesidades para este proyecto, no presenta mayores conflictos con la polarización ni el tipo de señal de reloj dado que viene en una placa que le provee de lo necesario para su encendido, calibración y programación, la distribución de pines viene detallada en dicha placa.

CAPÍTULO 1

PLANTEAMIENTO DEL PROBLEMA

1.1. Introducción

En este capítulo se analiza y diagnostica el problema a resolver, para lo cual se planteará la respectiva hipótesis y tesis.

Basándose en lo anterior, se plantearán los objetivos, la justificación, los alcances y la metodología de investigación para su diseño e implementación.

1.2. Problema a resolver

En los laboratorios de la carrera de Ingeniería Electrónica de la Universidad Politécnica Salesiana se tiene la necesidad de una herramienta que permita al estudiante realizar prácticas haciendo uso de una marca de microcontroladores distintos a los que actualmente tiene acceso, el uso de una sola marca de microcontroladores reduce el alcance que posibles proyectos podrían tener, la deficiencia de este tipo de herramienta resta conocimiento en los estudiantes de la Universidad Politécnica Salesiana con respecto a los estudiantes de otras entidades de educación superior.

Dado que en la actualidad la automatización industrial tiene un campo de aplicación cada vez más amplio se hace imprescindible contar con nuevas alternativas tecnológicas para realizar control industrial, las marcas más conocidas de microcontroladores entregan buenas prestaciones pero son deficientes cuando se los utiliza en proyectos destinados al ambiente industrial, siendo la robustez una de las principales demandas, y es justamente esta característica la que motiva a trabajar con los microcontroladores XMega de Atmel teniendo en cuenta sus excelentes prestaciones y su extenso campo de aplicación, permitiendo que su uso sea más frecuente y que el estudiante encuentre en este tipo de microcontroladores la mejor alternativa para realizar sus proyectos.

Por lo que aparece la siguiente pregunta:

¿Cómo se puede diseñar y construir un módulo de entrenamiento basado en el microcontrolador XMega de Atmel para los laboratorios de Sistemas Microprocesados de la Universidad Politécnica Salesiana?

1.3. Hipótesis

La hipótesis es la siguiente:
Diseño y construcción de un módulo de entrenamiento basado en el microcontrolador XMega de Atmel para los laboratorios de Sistemas Microprocesados de la Universidad Politécnica Salesiana, desarrollando e implementando un módulo de prácticas de entrenamiento.

1.4. Objetivos

A continuación, se describen el objetivo general y los objetivos específicos de este proyecto.

1.4.1. Objetivo general

Diseño y construcción de un módulo de entrenamiento basado en el microcontrolador XMega de Atmel para los laboratorios de Sistemas Microprocesados de la Universidad Politécnica Salesiana.

1.4.2. Objetivos específicos

- Analizar la arquitectura del microcontrolador XMega
- Comprender la programación en lenguaje de alto nivel del microcontrolador XMega.
- Diseñar aplicaciones con los puertos del microcontrolador XMega
- Diseñar aplicaciones con los temporizadores de los microcontroladores XMega

- Diseñar aplicaciones utilizando el lcd con los microcontroladores XMega
- Diseñar aplicaciones para el teclado con los microcontroladores XMega
- Diseñar aplicaciones para control de motores DC de hasta 12V con los microcontroladores XMega
- Diseñar aplicaciones para el ADC y el DAC con los microcontrolador XMega
- Diseñar aplicaciones para la comunicación Rs-232 con los microcontroladores XMega
- Diseñar aplicaciones para I2C con los microcontroladores XMega
- Diseñar aplicaciones para SPI con los microcontroladores XMega
- Diseñar aplicaciones para transmisión RF
- Construir la tarjeta para las aplicaciones respectivas

1.5. Justificación

Con la implementación de este proyecto para los laboratorios de Ingeniería Electrónica se aportará en el desarrollo académico de los alumnos en la materia Sistemas Microprocesados ya que contarán con un módulo completo de prácticas a realizar y el hardware respectivo para solucionar dichos problemas.

Dando a conocer una alternativa distinta a la que se viene enseñando en dicha materia y por ende capacitando a los alumnos a implementar proyectos con un alcance mayor al que se consigue actualmente.

Esta investigación es importante porque permite mejorar la calidad de enseñanza de la carrera, mediante la construcción de un nuevo equipo de entrenamiento para los laboratorios de la carrera Ingeniería Electrónica.

1.6. Alcances

El módulo de entrenamiento está basado en el microcontrolador XMega de Atmel para los laboratorios de Sistemas Microprocesados de la Universidad Politécnica Salesiana realiza lo siguiente:

- Este proyecto permitirá realizar prácticas usando los puertos y los temporizadores del microcontrolador XMega.
- Se realizarán aplicaciones haciendo uso del LCD, el teclado matricial 4x4, un motor DC de hasta 12 V conectados al microcontrolador XMega.
- Se generarán prácticas utilizando los convertidores ADC y DAC incorporados en el microcontrolador XMega.
- El proyecto planteado permitirá realizar prácticas haciendo uso de módulos de Radio Frecuencia interconectados al microcontrolador XMega.
- Permitirá también realizar aplicaciones haciendo uso del protocolo I2C y del protocolo SPI del microcontrolador XMega.
- Adicionalmente se crearán prácticas usando un chip conversor de RS-232 a USB para los puertos del microcontrolador XMega.

Este proyecto no realiza lo siguiente:

- No implementará comunicación ETHERNET, dado que dentro de las características del microprocesador éste no posee un puerto dedicado a dicha interface, ni pines asignados para tal propósito.

1.7. Metodología de la investigación

En este apartado se detallarán los pasos que se han planteado para la realización del presente proyecto comenzando por la investigación bibliográfica, seguida del desarrollo de hardware y software, y finalmente la construcción final y documentación del proyecto.

1.7.1. Investigación bibliográfica

Se realizará una investigación sobre la arquitectura del microcontrolador y sus registros para aprender a implementarlo de una manera práctica sobre un proceso.

Posteriormente se investigarán las herramientas con las que cuenta el software para la realización de las prácticas.

También se investigarán las características y utilización de la tarjeta XMega Ready para finalmente investigar la conexión y utilización de esta con el software Bascom para AVR

1.7.2. Desarrollo de hardware y software

Se diseñará y construirá una tarjeta de entrenamiento en la que se tendrá el hardware de las prácticas a realizar en el laboratorio.

Se realizarán aplicaciones básicas en el software de programación que permitan probar la correcta comunicación entre este y el microcontrolador así como comprobar que se puede controlar sus entradas y salidas digitales.
Finalmente se desarrollará varias aplicaciones en el software de programación de acuerdo a las prácticas presentadas.

1.7.3. Construcción final

Se harán las conexiones finales desde la tarjeta con el microcontrolador hacia la tarjeta de las prácticas y se realizarán las pruebas correspondientes para cada práctica presentada.

1.7.4. Documentación

Se escribirá el texto de la monografía correspondiente al presente proyecto de investigación, cuyo tema es el diseño y construcción de un módulo de entrenamiento basado en el microcontrolador XMega de Atmel.

CAPÍTULO 2

ESTADO DEL ARTE

2.1. Introducción

En este capítulo se analizará el estado del arte el cual consiste en los proyectos similares al tema del presente proyecto, además se presentarán los fundamentos teóricos en los que se basa el diseño, construcción e implementación del módulo de entrenamiento basado en el microcontrolador XMega de Atmel.

2.2. Estado del arte

Existen programas desarrollados usando el microcontrolador ATXMEGA128A1 con programación en Bascom para AVR sin embargo a continuación, se detallan dos proyectos realizados en otros entornos de programación y con microcontroladores de la serie AVR XMega distintos al que se está usando, con esto se pretende mostrar las ventajas que posee este tipo de microcontrolador, el primero de ellos es el Generador de Ondas Arbitrarias, Boston Android XMega, usando el microcontrolador XMega.

> El proyecto realizado muestra todas las ventajas que tiene el microcontrolador XMega de Atmel y su gran campo de aplicación, además de la versatilidad que presenta en comparación a otros microcontroladores; un generador de forma de onda arbitraria es una pieza útil, pero a menudo costosa, se utiliza para determinar la respuesta de los componentes de frecuencia estos a su vez generan señales portadoras que permiten ajustar circuitos resonantes, reproducir sonidos, o simplemente dibujar gráficos, Boston Android XMega trabaja como un generador de forma de onda y una vez configurado puede generar ondas tipo: sinusoidal, rampa, triángulo, cuadrado, etc. trabaja con frecuencias en el rango de 5 a 20 kHz con una amplitud de hasta 3.3 Vpp y está programado en el software libre AVRDUDE. (Wandererwolf, 2010)

Deja entrever las capacidades que se tienen con un proyecto en el que se use un microcontrolador de esas características. El segundo proyecto es el robot diseñado para jugar damas chinas, por estudiantes de la Universidad Estatal de Oregón para Mecatrónica, Oregón.

> El proyecto se basa en el diseño y construcción de un robot para jugar damas empleando Mecatrónica y Robótica,el robot utiliza el software OpenCV específicamente la biblioteca *visión* para encontrar el tablero de damas y ubicar las fichas, el diseño tiene varias etapas entre ellas, tiene un brazo robótico que trabaja en coordenadas polares y utiliza un sistema de aspiración para recoger y liberar las damas, tiene motores de corriente continua con codificadores y sistemas de control PID para controlar los ejes del brazo, los codificadores ópticos se utilizan para proporcionar información para un sistema de control de servos, el corazón del sistema de control electrónico es un microcontrolador XMega de Atmel que funciona a 32MHz y tiene soluciones completas de hardware para decodificación de cuadratura, la generación de PWM y la comunicación en serie. (Oregon State, 2010)

Los proyectos mostrados ofrecen una perspectiva real del sin número de aplicaciones que ofrece la serie XMega de Atmel y la importancia de crear un módulo que permita al estudiante familiarizarse con este tipo de microcontroladores, para que en un futuro se puedan crear nuevas aplicaciones partiendo de este conocimiento.

2.3. Marco teórico

El microcontrolador AVR XMEGA de la empresa ATMEL es utilizado en el desarrollo de aplicaciones que requieren bajo consumo de potencia, alto rendimiento y periféricos CMOS 8/16-bit, este dispositivo está basado en la arquitectura RISC mejorado. “La familia de microcontroladores AVR se basa en RISC buscando optimizar el tamaño del código, rendimiento y el consumo de energía” (Claus, 1998, pág. 9), ejecuta instrucciones de gran alcance en un solo ciclo de reloj, el microcontrolador XMEGA A logra rendimientos que se

aproximan a 1 millón de instrucciones por segundo (MIPS) por MHz lo que permite al diseñador del sistema optimizar recursos.

La CPU AVR combina un amplio conjunto de instrucciones con 32 registros de propósito general, "con estos 32 registros se elimina el código para la transferencia de datos entre el acumulador y la memoria, ya que cada uno de los registros actúa como acumulador" (Claus, 1998, pág. 9), todos los 32 registros están conectados directamente a la unidad aritmética lógica (ALU). Es la encargada de realizar las operaciones aritméticas básicas (suma, resta, multiplicación y división) y de operaciones lógicas (or, not, and), la ALU además de permitir realizar manipulación de bits como desplazamiento a la izquierda y a la derecha, rotar a la izquierda y a la derecha". Este microcontrolador está basado también en la Arquitectura Harvard, en donde un microcontrolador tiene espacios de memoria distintos para almacenar los datos de programa y memoria, en su procesador, permitiendo que dos registros independientes ingresen en una sola instrucción ejecutándolos en un ciclo de reloj. Los procesadores basados en Arquitectura Harvard pueden acceder a la memoria de datos y programa en un mismo instante, usando direcciones y datos de acceso distintos, la arquitectura resultante permite generar un código más eficiente, mientras que el rendimiento es mucho más rápido que los convencionales de un solo acumulador o microcontroladores basados en CISC, el propósito esencial de una Arquitectura CISC es intentar proporcionar una única instrucción de máquina para cada enunciado.

Los AVR tienen una familia mejorada que son los XMEGA los que incorporan gran funcionalidad y altas velocidades de reloj, entre las características que se encuentran dentro del manual de usuario se tiene:

> Sistema interno programable con capacidades de lectura y escritura, memoria de solo lectura programable y borrable electrónicamente (EEPROM) y memoria estática de acceso aleatorio (SRAM), controlador de cuatro canales para el acceso directo a memoria (DMA), sistema de eventos de ocho canales, controlador programable de interrupciones multinivel de ocho canales, 78 líneas generales de entradas y salidas

(E/S), contador en tiempo real de 16 o 32 bits (RTC), ocho temporizadores/contadores (TC) de 16 bits con modo de comparación y modulación por ancho de pulso (PWM), ocho receptores transmisores síncronos/asíncronos universales (USART), cuatro líneas para manejar circuitos inter integrados (I2C), cuatro interfaces seriales de periféricos (SPI), convertidores analógico digital de 12 bits con entrada diferencial y ganancia programable (ADC), convertidores digital analógico de 12 bits (DAC), perro guardián con un oscilador interno dedicado (WATCHDOG). La interfaz de depuración del programa (PDI) es rápida para la programación y depuración a 2 pines. (Atmel, 2009)

Este dispositivo también tiene un estándar IEEE 1149.1 compatible con la interfaz para pruebas conjuntas del grupo de acción (JTAG) y "se puede utilizar para depuración del chip y la programación". (Lewin, 2006)

Los dispositivos AVR XMega tienen cinco modos seleccionables para ahorro de energía: El modo en ralentí (idle) detiene la CPU mientras permite seguir funcionando a la SRAM, al controlador DMA, al sistema de eventos, las alarmas y todos los periféricos. El modo apagado (power down) mantiene activa la SRAM y el contenido de los registros, pero detiene el oscilador y desactiva todas las demás funciones hasta el próximo reset. El modo ahorro de energía (power save) en el que el contador en tiempo real (RTC) continúa operando mientras los demás dispositivos se desactivan. El modo en espera (standby) desactiva todos los dispositivos excepto el oscilador de cristal. El modo de espera extendido (extended standby) mantiene en funcionamiento al oscilador y al contador asíncrono.

2.3.1 Arquitectura

En este subcapítulo se analizara la arquitectura de los componentes del microcontrolador XMega.

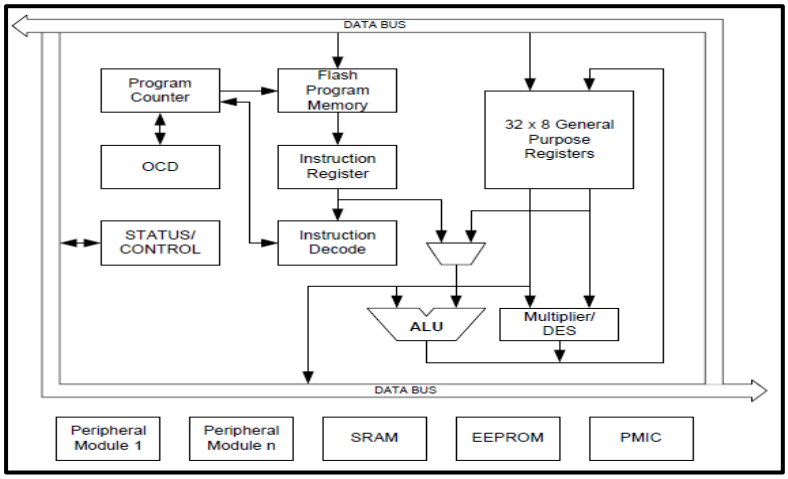
2.3.1.1. Cpu del AVR

En este apartado se tratara la arquitectura del núcleo del AVR XMega, buscando mejorar la multitarea el CPU se basa en arquitectura Harvard la cual usa memorias y buses para datos y programa separados. Las instrucciones en la memoria de programa se ejecutan mediante segmentación de un solo nivel (pipeline de un solo nivel), en la cual mientras una instrucción está siendo ejecutada la siguiente es extraída desde la memoria de programa, permitiendo ejecutar instrucciones en cada ciclo de reloj.

La figura 1, muestra un diagrama de la arquitectura en el AVR. La unidad aritmética lógica (ALU) es compatible con operaciones aritméticas y lógicas entre registros, o entre una constante y un registro; se puede también ejecutar operaciones con un solo registro. Después de una operación aritmética el registro de estado se actualiza para reflejar el resultado de la operación.

La función principal de la CPU es: acceder a memorias, realizar cálculos, controlar periféricos, y manejar interrupciones. La memoria de programa es de tipo flash y es reprogramable. El archivo de registros de acceso rápido contiene 32 registros de trabajo de propósito general de 8 bits con un tiempo de acceso a cada uno de ellos de un ciclo de reloj. Esto permite la operación de la unidad aritmética lógica (ALU) en un ciclo de reloj. En una típica operación de la ALU se toman dos operandos del archivo de registros, se ejecuta la operación y el resultado se almacena nuevamente en el archivo de registros en un ciclo de reloj.

Figura 1. Diagrama de bloques de la CPU del avr



Fuente: (Atmel, 2009)

El espacio de datos y el espacio del programa son dos espacios de memoria diferentes. El espacio de memoria de datos se divide en: los registros de entrada y salida (E/S) y la memoria estática de acceso aleatorio (SRAM). Además el espacio de memoria de la EEPROM puede ser asignado en la memoria de datos. Todos los estados de E/S y registros de control residen en las direcciones más bajas de la memoria de datos, esto se conoce como el espacio de memoria de E/S. A las direcciones de más de 64 bits se puede acceder directamente desde el 0x00 - 0x3F. El resto de la memoria corresponde al espacio de memoria extendido de E/S, que van desde la dirección 0x40 a 0x1FFF, para acceder a las E/S registradas se lo hace utilizando las instrucciones de carga (LD/LDS /LDD) e instrucciones para guardar (ST/STS/STD).

La SRAM contiene datos y código que no se pueden ejecutar desde esta memoria pero se pueden acceder fácilmente a través de un direccionamiento. La primera dirección de la SRAM es 0x2000. Los datos de direcciones de 0x1000 hasta 0x1FFF se reservan para la asignación de memoria de la EEPROM. La memoria de programa se divide en dos secciones: la sección del programa de aplicación y la sección de inicio del programa, en ambas secciones se han dedicado bits de bloqueo para la protección de datos de escritura y lectura. La instrucción de SPM, que se utiliza para la programación de la memoria flash, reside en la sección de inicio del programa. Existe una tercera sección dentro del programa de aplicaciones, esta sección es llamada la sección de la tabla de

aplicaciones, posee bits separados de bloqueo para protección de escritura y lectura, esta sección puede ser utilizada para almacenar datos no volátiles o software de aplicación.

2.3.2. Metodología del diseño electrónico

En la metodología del diseño de circuitos electrónicos, tiene mucha importancia el uso de software como herramienta para la verificación del comportamiento y respuesta de un circuito electrónico antes de la construcción del prototipo final. La metodología que aquí se detalla, es una recopilación de varias fuentes, sin embargo cualquiera sea la metodología que se aplique, es necesario su puesta en práctica en un laboratorio.

El diseño electrónico tiene como objetivo la obtención de un circuito funcionalmente correcto, lo más sencillo y eficiente con una construcción que implique el menor tiempo posible. Por ello la metodología de diseño debe garantizar que se minimicen los errores, y que en caso de que se produzcan, estos puedan ser detectados en las fases iniciales. Para lo cual, es necesario que se implanten puntos de verificación en distintas etapas de diseño antes de empezar con el montaje del circuito. Una de las primeras herramientas que se utiliza y también la más fundamental son los simuladores. No se debería considerar el diseño de un circuito como correcto sin antes comprobar su funcionamiento en algún simulador.

El primer paso consiste en generar unas especificaciones de diseño, las cuales describen la funcionalidad que se espera, así como otras propiedades, como la necesidad de energía para polarizar correctamente todos los componentes que se agreguen, el espacio en el que se van a colocar, etc.

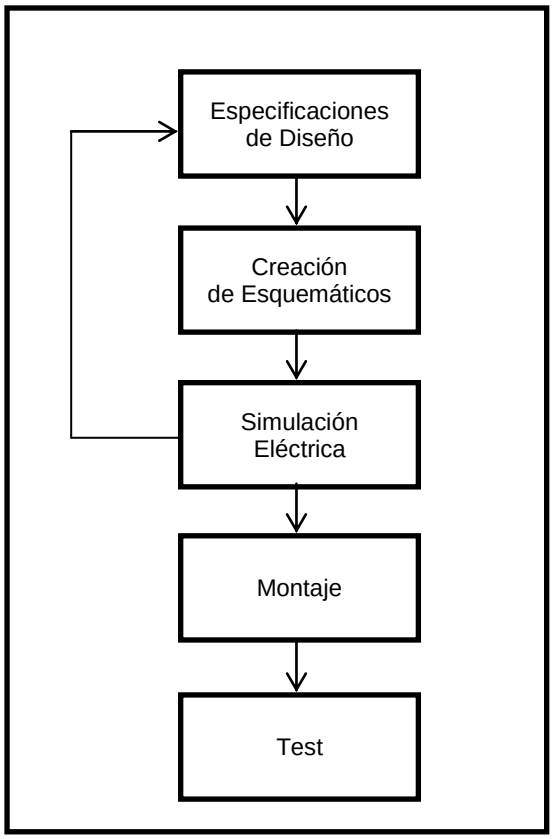
En general las especificaciones de diseño dejan bastante libertad al diseñador en aspectos como la topología del circuito, la distribución de los componentes, el espacio de trabajo. Sin embargo es menester recordar que el circuito debe incluir las fuentes de alimentación, la tierra y los conectores de entrada y salida. Una

vez completa la descripción del circuito utilizando cualquier herramienta afín, el funcionamiento eléctrico y la funcionalidad deben ser verificados utilizando un simulador, con esto no solo se detectan errores respecto a las especificaciones, sino que también se detectan errores en la creación de los esquemáticos. En este paso es extremadamente importante que los resultados de simulación sean los correctos antes de seguir con los siguientes pasos de diseño.

Solamente cuando los resultados de simulación indican que el circuito funciona correctamente se pasa a la etapa de montaje. Terminado el montaje se debe proceder con la verificación del funcionamiento, antes de proceder a su uso definitivo.

La imagen detalla un esquema de la metodología de diseño electrónico

Figura 2. Diagrama de bloque de la metodología del diseño



Fuente: (Jiménez Carlos, 2010)

A continuación se detallan ciertos puntos que ayudaran en el diseño:

- La idea: ¿qué tipo de condiciones quiere que cumpla el circuito electrónico?
- Diseño del circuito: ¿qué tipo de circuito debe usar, para realizar la tarea?
- Partes a seleccionar: costos, tamaño
- Forma del circuito impreso: Distribuya los componentes sobre la placa de circuito impreso, trace las pistas y grabe por el medio deseado.
- Ensamblado: hecho los agujeros para los componentes, monte los componentes verificando la posición adecuada y suéldelos.
- Chequeo: confirme que no haya errores, pistas sobrepuestas, componentes mal soldados
- Ajuste: verifique que el circuito acabado cumple las expectativas
- Instalación: adapte el circuito en la posición o lugar de uso
- Combinación: conecte otros dispositivos al circuito realizado

2.3.2.1. Medios.

Para que un proyecto, sin importar el tamaño del mismo llegue a convertirse en una realidad se necesita de varios componentes, el diseño, la mano de obra y sobre todo las herramientas, a continuación detallamos algunas de las que son necesarias.

2.3.2.1.1. Técnica para crear un circuito impreso

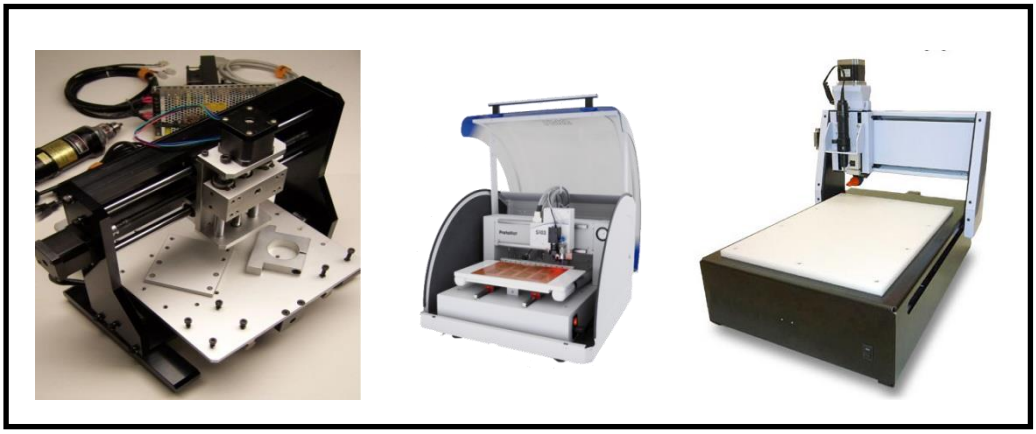
En la actualidad se tiene una amplia gama de técnicas y herramientas para elaborar un circuito impreso, desde los métodos de casa hasta aquellos en que se hace uso de una máquina de control numérico computarizado (CNC). Las técnicas caseras de construcción implican un bajo costo, pero los resultados al emplear control por computador es el grado de precisión en el circuito impreso resultante.

Entre las máquinas de control numérico en el mercado se tiene:

- CNC Sable 2015
- LKF ProtoMat S103
- The ElevenLab

Cada máquina cuenta con el software necesario para el diseño del circuito impreso, Sable 2015 cuenta con Match3, ProtoMat trabaja con Circuit Pro y TheElevenLab con Easy CAD.

Figura 3. Máquinas para crear un circuito impreso



Fuente: CNC, LKFP, Eleven Lab, 2013

En caso de no contar con el presupuesto necesario, se tiene el método alternativo, consiste en:

- Papel Termotransferible
- Marcador de tinta permanente o una impresora laser
- Una planchadora

Este es un método casero de diseño, los resultados dependen de la práctica que el diseñador posea.

Para completar el circuito impreso son necesarias las siguientes herramientas:

- Un soldador de 25 a 30 Vatios
- Estaño con un bajo porcentaje de plomo
- Un de soldador
- Destornilladores
- Alicates

- Corta cables
- Pinzas

Los componentes que se soldaran obedecen al proyecto en el que se trabaje.

2.3.2.1.2. Entornos de diseño

Los siguientes son entornos de diseño que cuentan con un software para simulación de circuitos integrados y para el diseño de circuitos impresos.
Sin la utilización de cualquier entorno de diseño, el proceso para construir un equipo necesita del producto final para realizar pruebas, lo cual implica un gasto de tiempo y dinero al momento de realizar ajustes. Sin garantizar que estos vayan a ser confiables.

Al utilizar un entorno de diseño la depuración de errores queda así:

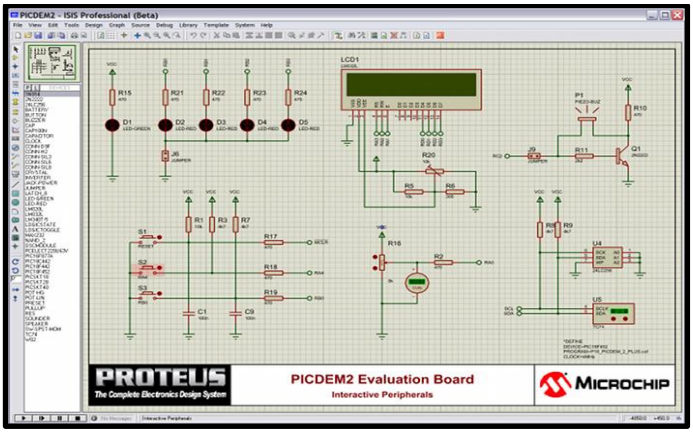
- Diseño del Circuito
- Desarrollo del Software
- Simulación del Circuito y del Software
- Diseño del Circuito Impreso
- Construcción del prototipo

La principal ventaja que se consigue es que la fase de pruebas no supone la necesidad de volver a construir un prototipo, con el ahorro de costos y tiempo que esto implica.

- **Proteus**

Es un entorno de desarrollo integrado para Windows solamente, diseñado para la completa realización de proyectos electrónicos, encargándose de: diseño, simulación, depuración y construcción.
El primer paso consiste en el diseño del esquema electrónico con ISIS

Figura 4. Isis

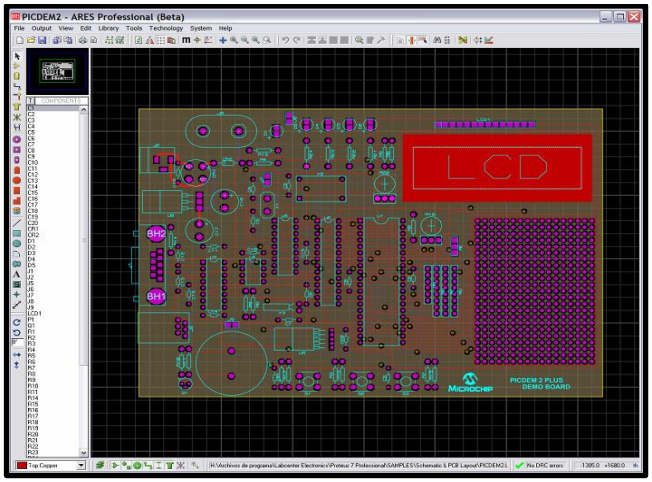


Fuente: (Proteus, 2011)

Se puede realizar simulaciones tanto del circuito electrónico como de la lógica del programa, gracias a dos módulos ProSpice y VSM. Una vez diseñado y depurado en ISIS el esquema electrónico, se generan automáticamente las listas de redes que interconectan pines.

ARES es capaz de recibir las listas generadas, y realizar el diseño de la placa de circuito impreso. Asegurando que los pines estarán unidos de manera idéntica a la del esquema electrónico.

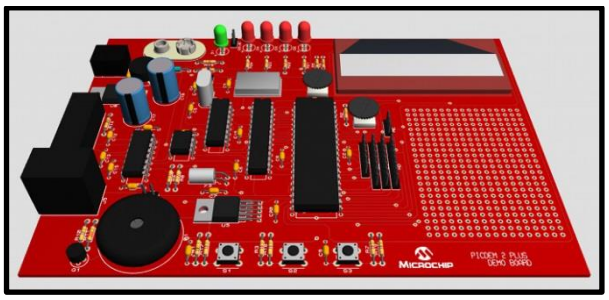
Figura 5. Ares



Fuente: (Proteus, 2011)

Por último ofrece la posibilidad de ver la imagen en tres dimensiones.

Figura 6. Diseño en 3d



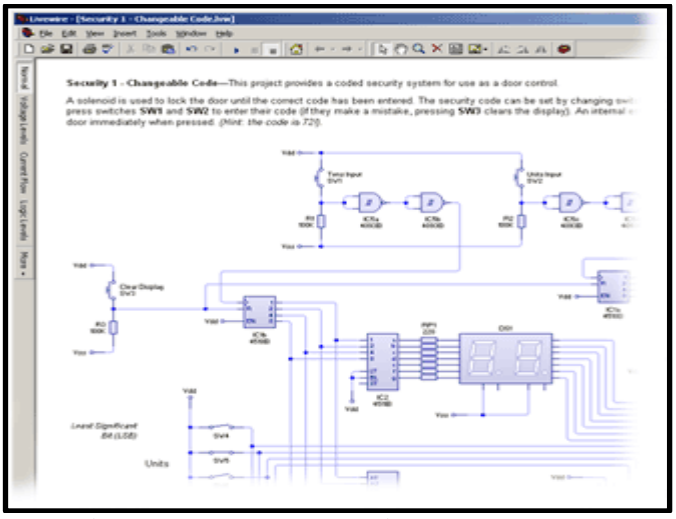
Fuente: (Proteus, 2011)

Cualquier modificación que se genere en el esquema eléctrico de ISIS, será enviados a ARES, y modificara la placa de circuito impreso.

- **Livewire**

Software de diseño orientado a trabajar en plataforma Windows, capaz de diseñar y simular circuitos electrónicos. El diseño consiste en arrastrar componentes hacia el espacio de trabajo, la interconexión será automática, sin embargo el diseñador puede deshabilitar esta opción.

Figura 7. Livewire

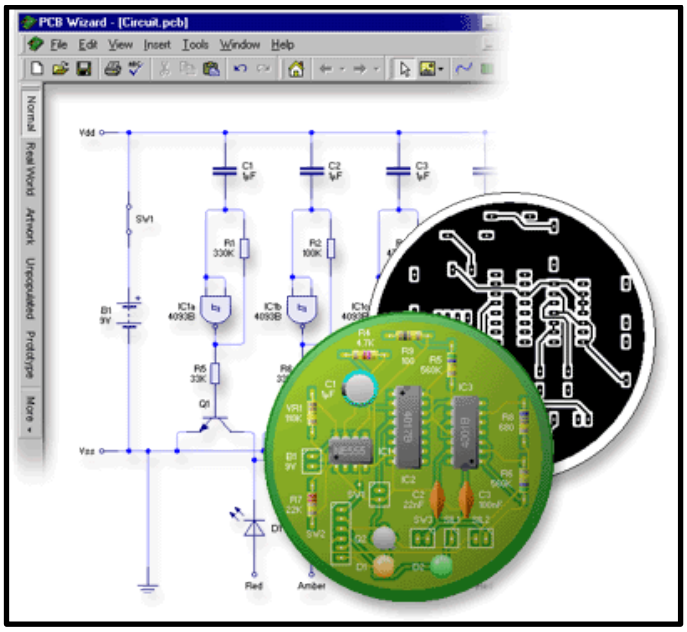


Fuente: (New Wave Concepts, 2012)

Incorpora animaciones en la simulación, así, cuando un circuito presente algún error como un corto o una línea sin conexión, se podrá observar una pequeña explosión.

LiveWire incorpora PCB Wizard, una herramienta diseñada para realizar el circuito impreso, el funcionamiento es similar, los componentes se arrastran al espacio de trabajo y son interconectados automáticamente.

Figura 8. Pcb wizard



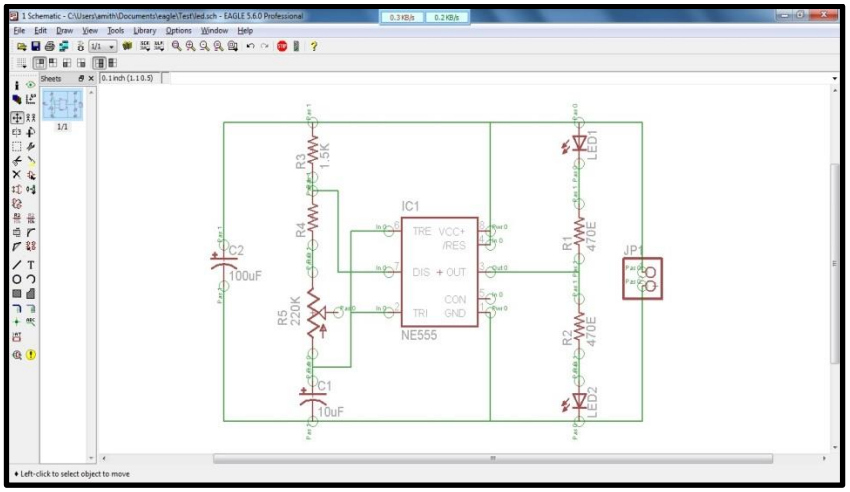
Fuente: (New Wave Concepts, 2012)

- **Eagle**

El nombre EAGLE proviene del acrónimo Easily Applicable Graphical Layout Editor (Editor de Diseño Gráfico de Fácil Aplicación), su diseño le permite funcionar en la plataforma de Windows, MAC o Linux.

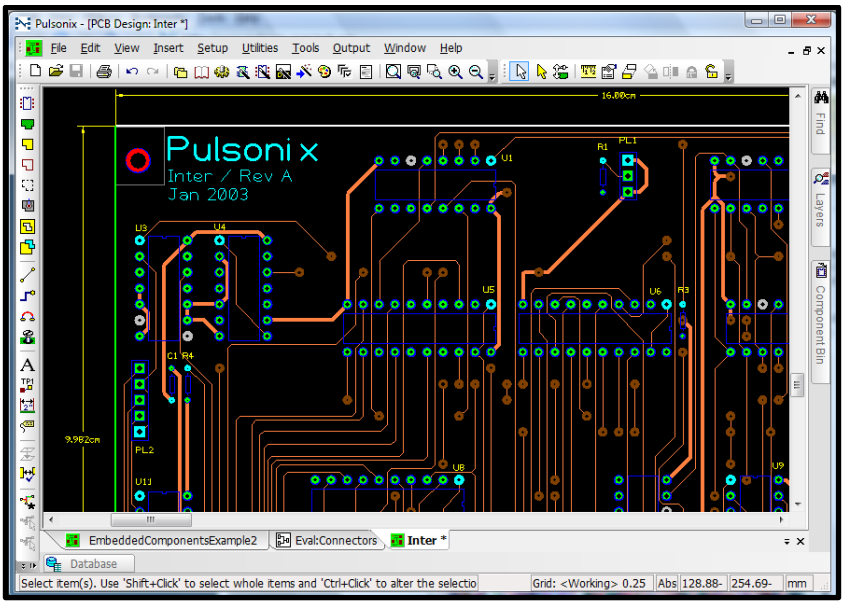
Posee la interfaz para realizar el diseño electrónico, con una librería surtida de elementos, para el diseño de la placa de circuito impreso EAGLE cuenta con un simulador de rutas, las versiones de pago le permiten generar hasta 16 capas de diseño, sin embargo para fines educativos se cuenta con las versiones de prueba, estas permiten manejar 2 capas de diseño.

Figura 9. Eagle diseño del circuito



Fuente: (Cadsoft, 2011)

Figura 10. Eagle diseño de pcb



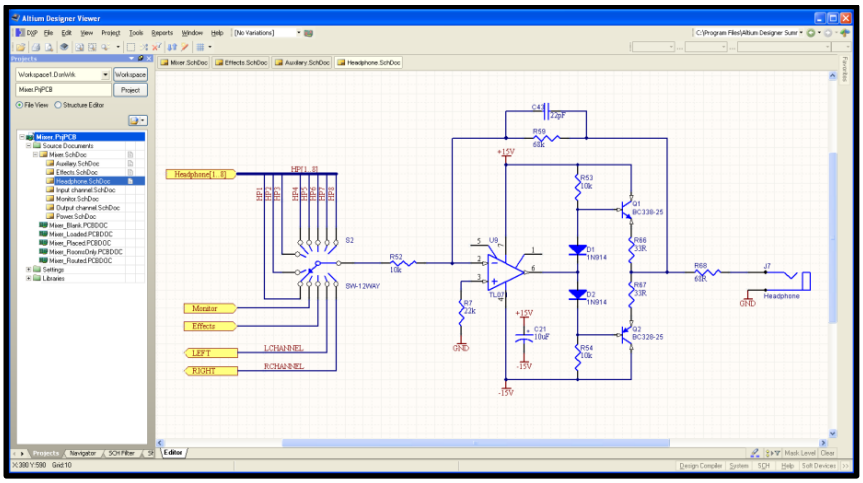
Fuente: (Cadsoft, 2011)

- **Altium designer**

ALTIUM Designer es un conjunto de programas para el diseño electrónico en todas sus fases y para todas las disciplinas, ya sean: esquemas, simulación, diseño de circuitos impresos, FPGA o desarrollo de código para microprocesadores. Es un programa único el que abarca todas estas características, se tiene acceso a aquellas por las cuales se haya pagado la

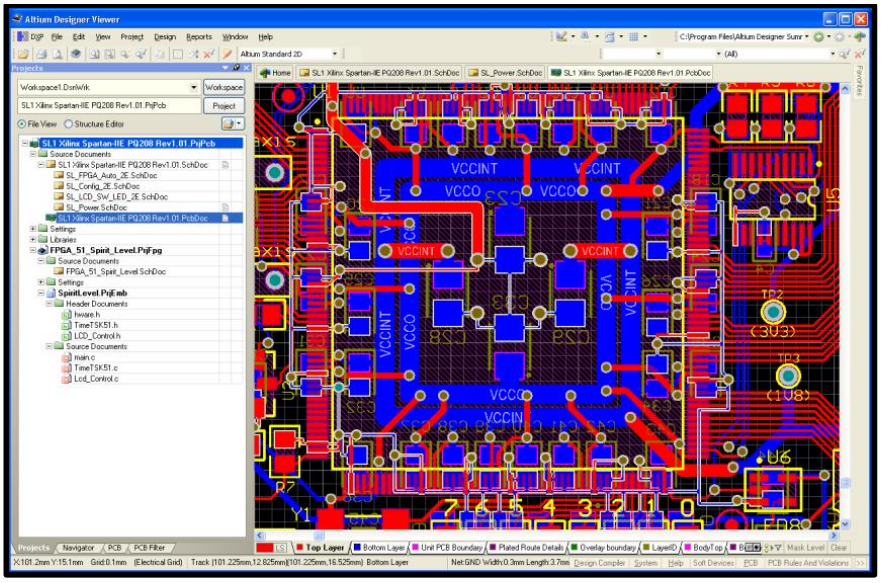
licencia respectiva, es funcional solamente en plataforma Windows según la propia página de ALTIUM designer.

Figura 11. Altium designer diseño del circuito



Fuente: (Altium, 2013)

Figura 12. Altium designer diseño de pcb



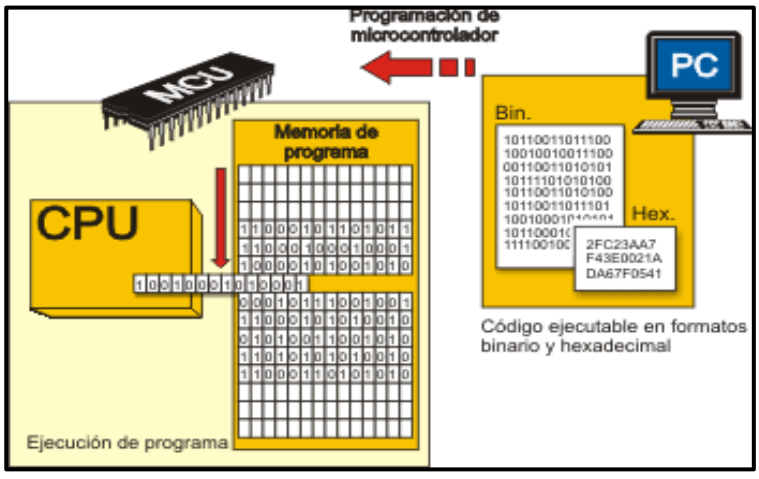
Fuente: (Altium, 2013)

2.3.3. Lenguajes de programación

Los lenguajes de programación han evolucionado a través de los años llevándonos desde lenguajes de bajo nivel, ensambladores; hasta los que hoy en día se ocupan en la escritura de programación de microcontroladores. Uno de ellos es Bascom el cual entrega un entorno de desarrollo integrado que da soporte a varias familias de microcontroladores AVR entre ellas la XMega de ATMEL. Bascom tiene plataforma Basic.

La secuencia para que un programa cargado en un microcontrolador sea ejecutada es: [12] El microcontrolador ejecuta el programa cargado en la memoria Flash. Esto se denomina el código ejecutable y está compuesto por una serie de ceros y unos. Dependiendo de la arquitectura del microcontrolador, el código binario está compuesto por palabras de 12, 14 o 16 bits, cada palabra se interpreta por la CPU como una instrucción a ser ejecutada durante el funcionamiento del microcontrolador. Todas las instrucciones que el microcontrolador puede reconocer y ejecutar se les denominan colectivamente *Conjunto de Instrucciones*. El código ejecutable se representa con frecuencia como una serie de los números hexadecimales denominada código Hex. En la figura 80, se puede observar el proceso de creación de un programa en el microcontrolador.

Figura 13. Programación del microcontrolador



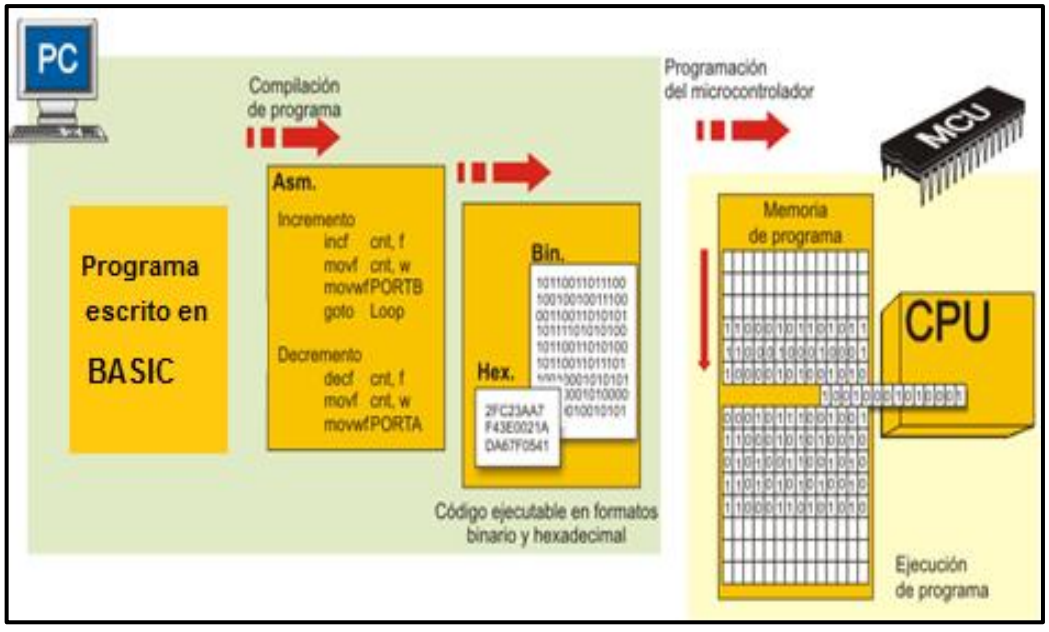
Fuente: (Mikroelectronika, 1998)

2.3.3.1. Lenguaje Basic

El lenguaje Basic dispone de todas las ventajas de un lenguaje de programación de alto nivel y permite realizar algunas operaciones tanto sobre los bytes como sobre los bits. Las características de Basic pueden ser muy útiles al programar los microcontroladores. Además, Basic está estandarizado (el estándar ANSI) así que el mismo código se puede utilizar muchas veces en diferentes proyectos. Lo que lo hace accesible para cualquiera que conozca este lenguaje sin reparar en el propósito de uso del microcontrolador.

Basic es un lenguaje compilado, lo que significa que los archivos fuentes que contienen el código Basic se traducen a lenguaje máquina por el compilador. En la figura 81, se puede observar cómo se realiza la compilación de un programa de un lenguaje de programación de alto nivel a bajo nivel, este lenguaje permite combinar comandos para aplicarlos en funciones de alto nivel, se debe tomar en cuenta que todos los sistemas operativos utilizan este lenguaje por lo que es conocido como lenguaje universal de programación.

Figura 14. Compilación del programa de un lenguaje de programación de alto nivel a bajo nivel

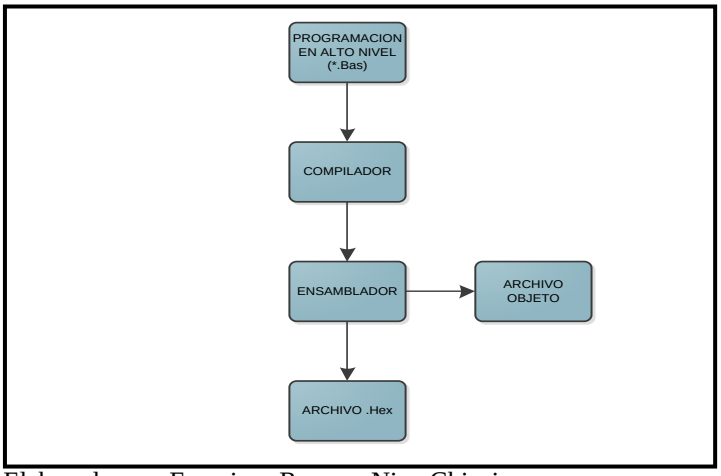


Fuente: (Mikroelectronika, 1998)

La programación de microcontroladores consiste en un conjunto de reglas sintácticas y semánticas, las mismas que definen la estructura y significado de los elementos a utilizarse.

La herramienta Bascom AVR sirve para realizar programas en alto nivel para microcontroladores AVR, el mismo que posee un compilador y un ensamblador que traduce las instrucciones estructuradas en lenguaje de maquina como se muestra en la figura 82:

Figura 15. Diagrama de bloques de programación estructurada



Elaborado por: Francisco Reyes y Nina Chicaiza

Existen algunos tipos de lenguaje de programación para los microcontroladores, en este proyecto se utilizara el lenguaje Basic.

2.3.4. Bascom AVR

BASCOM-AVR es un COMPILADOR BASIC en Windows permite programar todo tipo de microcontroladores de ATMEL AVR. Está diseñado para trabajar en W95/W98/NT/XP/WIN7.

Bascom viene en tres variantes:

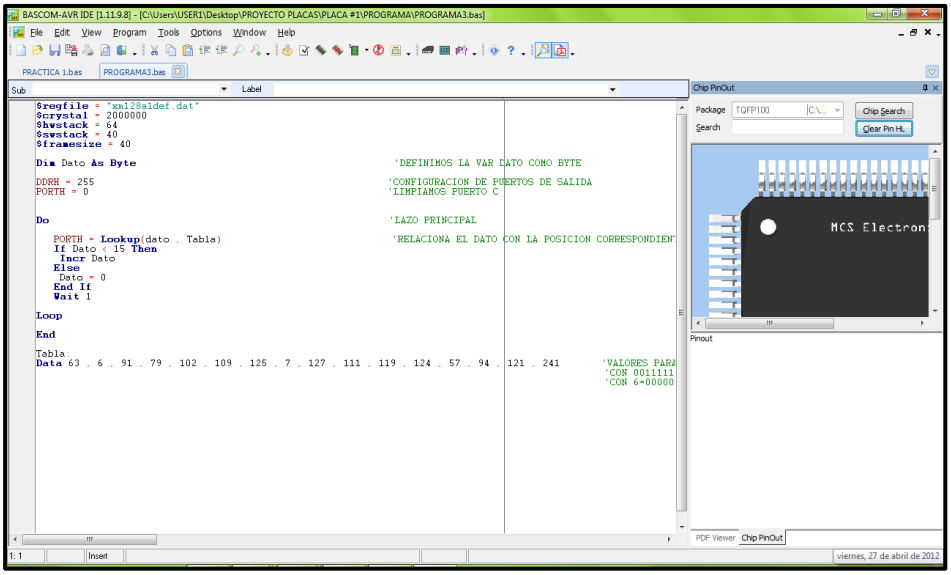
- Bascom-LT para microcontroladores Atmel AT89Cx051
- Bascom-8051 de 8051 microcontroladores.
- Bascom-AVR de microcontroladores Atmel AVR

Bascom es una aplicación para PC que permite:

- Escribir programas en Basic
- Traducir estos programas en el PC para código maquina (un formato que el controlador puede ejecutar AVR).
- Simular el código compilado
- El uso de programas externos compilado en microcontroladores AVR Atmel.

Bascom permite crear rápidamente prototipos porque se ha incorporado soporte para todos los microcontroladores AVR, tiene características tales como: Contadores / temporizadores, UART, ADC, PWM, I2C. Además de que soporta gran cantidad de periféricos, tales como: Botones, LCD de alfanuméricos, LCD Gráficos, PS / 2 para teclado, Control remoto por infrarrojos; es sobre todo este apoyo que hace que el uso de Bascom atractivo en términos de tiempo ahorrado. La figura 83, muestra el entorno de programación Bascom con la versión 1.11.9.8

Figura 16. Entorno de programación Bascom versión 1.11.9.8



Fuente: Bascom
Elaborado por: Francisco Reyes y Nina Chicaiza

CAPÍTULO 3

DISEÑO, DESARROLLO E IMPLEMENTACIÓN

3.1. Introducción

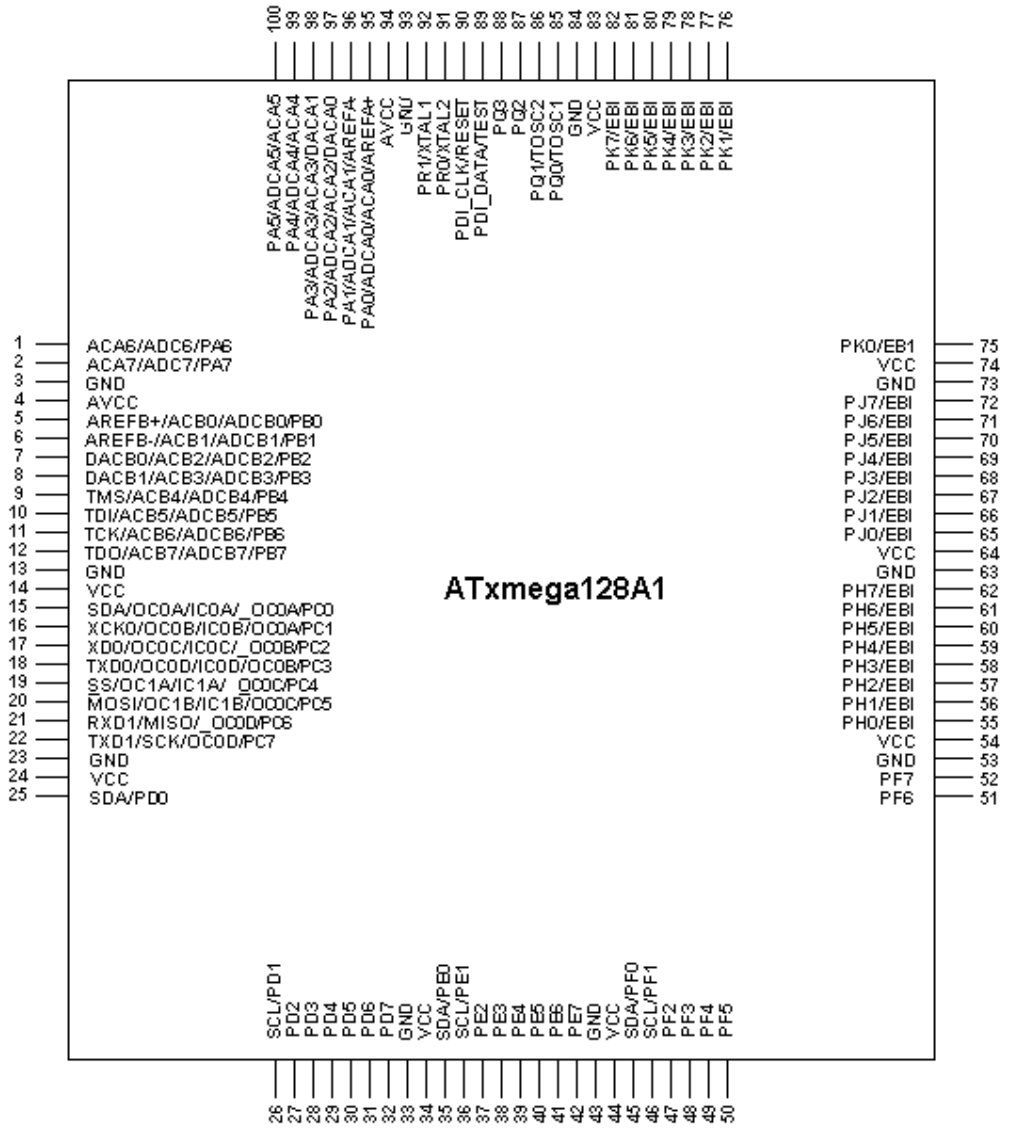
En este capítulo se analizará el diseño del hardware, el desarrollo del software y la construcción de cada uno de los módulos con el microcontrolador XMega, se verificará el funcionamiento de cada uno de los módulos por separado, para lo cual es necesario realizar pruebas prácticas entre el software BASCOM AVR y el hardware con el microcontrolador ATXMEGA128A1.

3.2. Desarrollo de software

Los códigos están programados en Bascom-AVR, el cual es un software de programación ejecutable para plataforma Windows, el microcontrolador AVR tiene precargado un bootloader, lo que permite programar directamente desde el computador, sin necesidad de quitar el microcontrolador de la placa de control. Se debe descargar la aplicación mikrobootloader para cargar código hexadecimal a través del gestor de arranque XMega, el manual de la tarjeta XMega ready se encuentra detallado en el Anexo 1 conjuntamente con el manual para el mikrobootloader que se encuentra en el Anexo 2.

La figura 84, muestra el entorno de distribución de pines generales y de propósito específico en el microcontrolador AVR, sus 100 pines distribuidos en varios puertos entregan un alto rango de funcionalidad.

Figura 17. Distribución de pines del avr xmega128a1



Fuente: (Atmel, 2009)
Elaborado por: Francisco Reyes y Nina Chicaiza

3.2.1. Detalle de los pines en el microcontrolador

Los 100 pines que el microcontrolador Xmega128A1 están distribuidos de la siguiente manera:

Tabla 1. Distribución de los pines en el microcontrolador

Tierra GND	3,13,23,33,43,53,63,73,83,93
VLC	4,14,24,34,44,54,64,74,84,94
PUERTO A	1,2,95-100
PUERTO B	5-12
PUERTO C	15-22
PUERTO D	25-32
PUERTO E	35-42
PUERTO F	45-52
PUERTO H	55-62
PUERTO J	65-72
PUERTO K	75-82
PUERTO Q	85-88
PUERTO R	91-92
DATA	89
CLK	90

Elaborado por: Francisco Reyes y Nina Chicaiza

Solamente los puertos Q y R no cuentan con ocho pines, el primero cuenta con cuatro pines y el segundo solamente con dos; todos los demás puertos tienen una funcionalidad de ocho pines. Cada pin en ciertos puertos comparten funciones, aunque todos se pueden utilizar como entradas y salidas, existen algunos que al habilitarlos sirven como comparadores, realizan adquisición, generan ondas, habilitan canales RX y TX para transmisión de datos seriales, controlan el ancho de pulso, sirven como señal de reloj

3.3. Construcción del módulo de entrenamiento

A continuación, se describe la construcción del módulo en la parte de hardware.

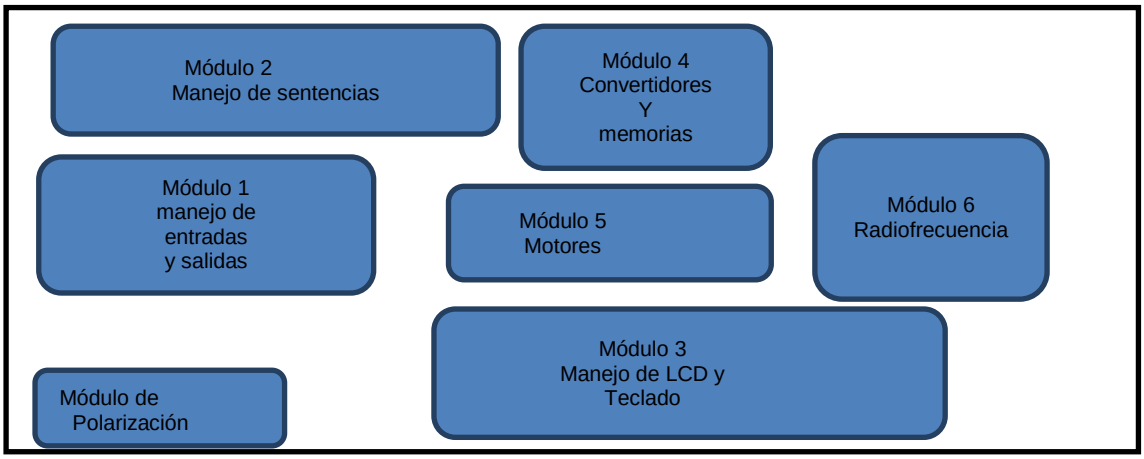
3.3.1. Descripción general

La tarjeta de entrenamiento que se presenta como el objeto de este trabajo de graduación posee varios módulos independientes uno de otro, los módulos poseen pines particulares de polarización, cumpliendo un propósito específico, sin embargo se pueden combinar dos o más módulos para prácticas que lo requieran, las aplicaciones que se presentan en las hojas de anexos, son por el momento aplicaciones que requieren un máximo de dos módulos, lo que se busca es que el lector posea un dominio total de cada módulo de manera singular, en primer lugar; para luego poder realizar combinaciones de los módulos. Un módulo que siempre estará acompañando a cualquier otro, es el módulo de polarización, la energía se obtiene desde el puerto USB, los pines del módulo de polarización nos brindan un alcance suficiente para polarizar todos los módulos a la vez. Los componentes que se utilizan en las prácticas, son remplazables en caso de existir problemas.

Los módulos presentes en la tarjeta de entrenamiento se diseñaron en base a las necesidades de las prácticas para los laboratorios de Sistemas Microprocesados I en la Universidad Politécnica Salesiana.

3.3.2. Diagrama de bloques

Figura 18**.** Diagrama de bloques de la tarjeta de entrenamiento



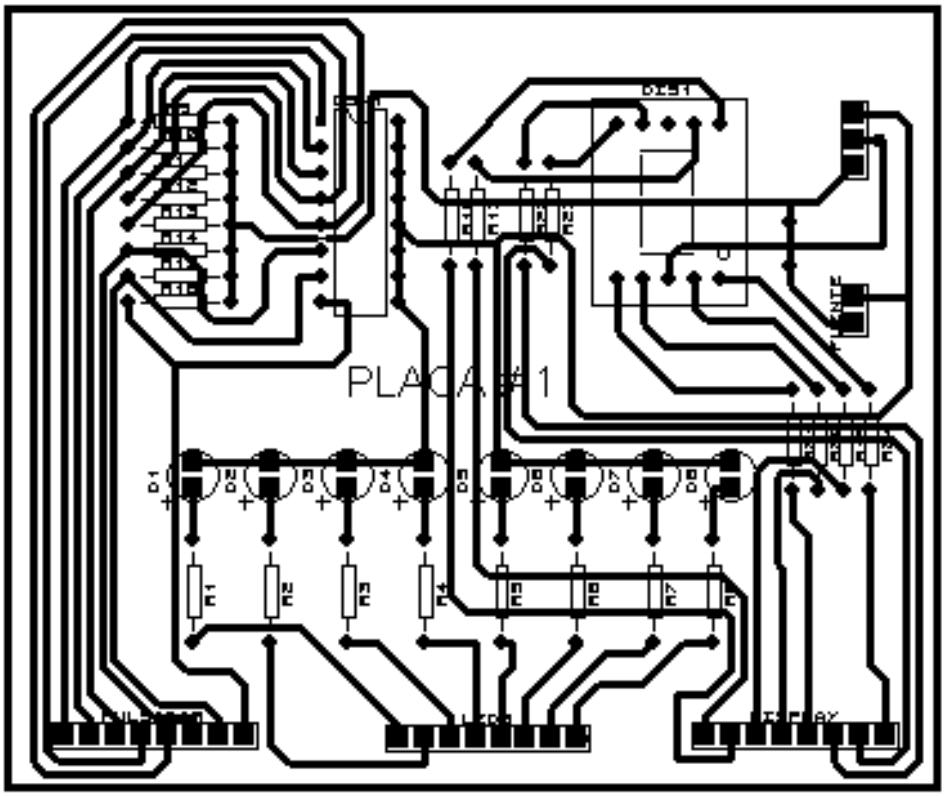
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3. Elaboración de los módulos individuales para la tarjeta de entrenamiento

El diseño de la cada una de los módulos se elaboró utilizando el software Proteus-Ares a partir del circuito diseñado en Proteus-Isis.

3.3.3.1. Diseño del módulo nº 1

Figura 19. Pistas del módulo n° 1



Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

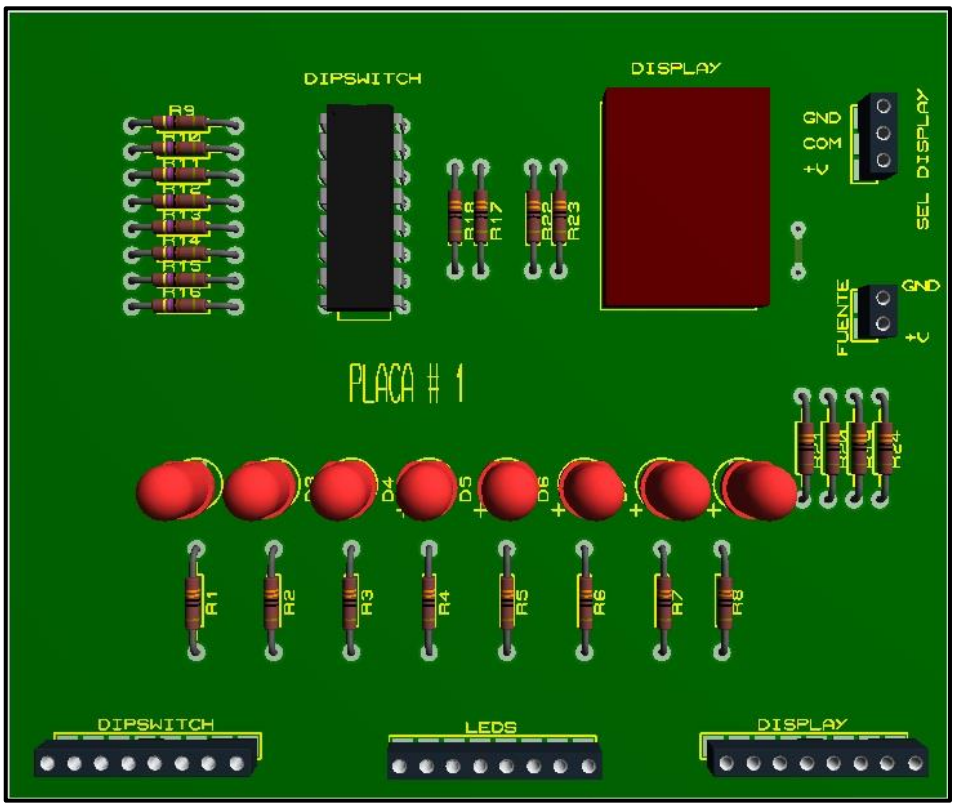
3.3.3.1.1. Descripción del módulo

El propósito de este módulo es realizar prácticas con los distintos puertos del microcontrolador, cuenta con: pines para polarización, puertos para manejo de un display de 7 segmentos con un selector para un display de ánodo o cátodo común según la necesidad, puertos para controlar 8 leds, puertos para enviar

información hacia el microcontrolador mediante un dipswitch y las resistencias necesarias.

3.3.3.1.2. Detalle de pines en el módulo

Figura 20. Detalle de pines del módulo n°1



Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

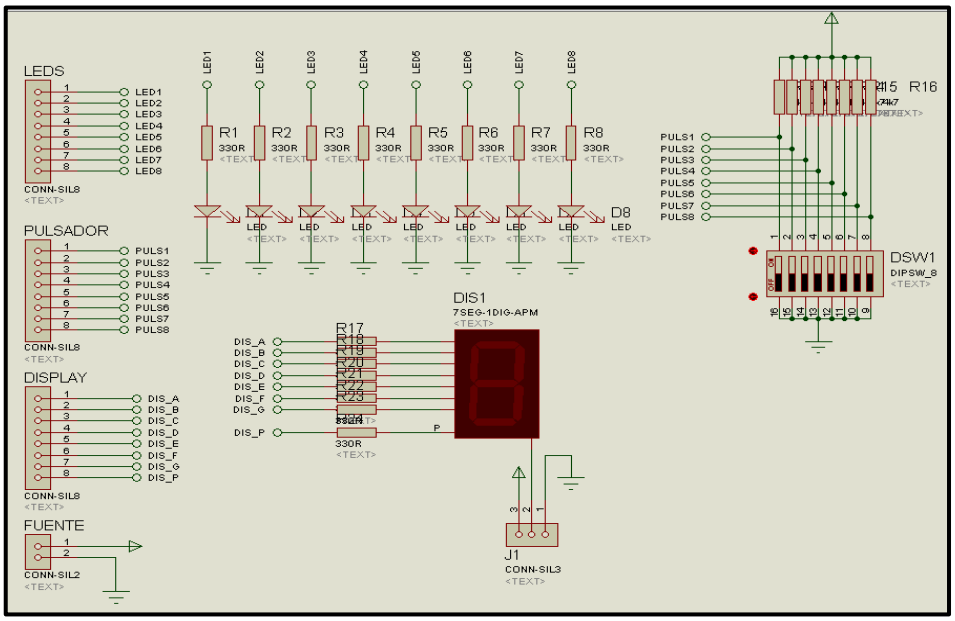
3.3.3.1.3 Uso del módulo nº1

La polarización puede ser de 5v proveniente de cualquier fuente o se puede utilizar 3,3v desde el microcontrolador. Para encender los leds se conectará cualquier puerto del XMega en los pines LEDS. El control del display se realiza mediante un puerto del XMega conectado a los pines display, si es un display de ánodo común se deberá conectar el pin común (COM) en el selector de display con +V, caso de ser tipo Cátodo común se hará la conexión entre el común y tierra. Los puertos de dipswitch sirven para enviar bits hacia el

microcontrolador, se debe controlar que el voltaje con el que se envíen datos hacia el microcontrolador no supere los 3,3V.

3.3.3.1.4 Esquemático de pines en el módulo

Figura 21. Esquemático del módulo n°2

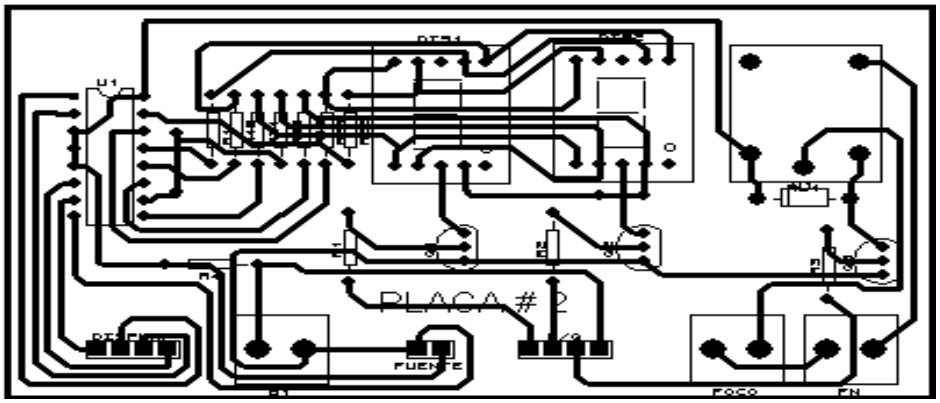


Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.2. Diseño del módulo nº 2

Figura 22. Pistas del módulo n° 2



Fuente: Ares

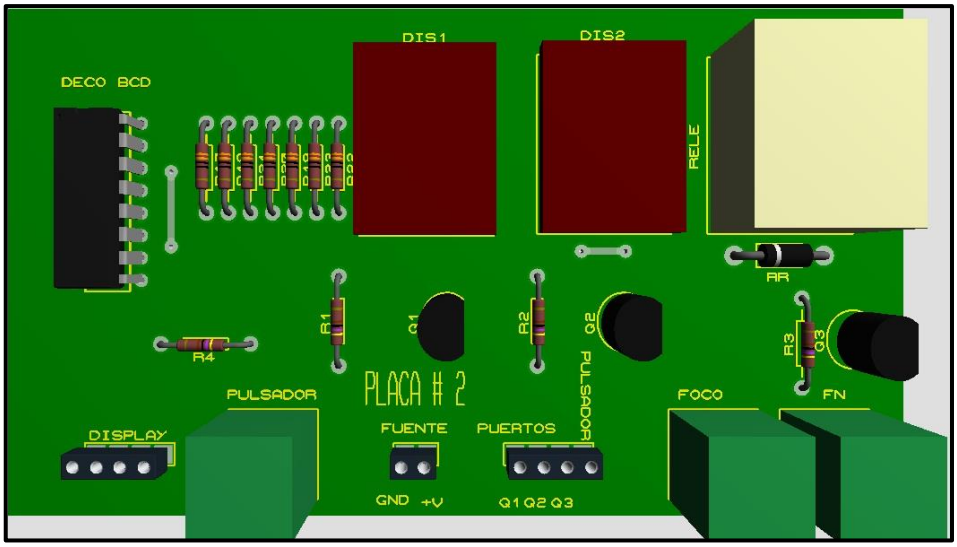
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.2.1. Descripción del módulo

Este módulo permite realizar prácticas utilizando lazos y declaraciones de programación en Bascom, cuenta con: dos display de 7 segmentos controlados por transistores, puertos para controlar dichos transistores, un decodificador BCD, puertos para controlar el decodificador, un pulsador, un relé para manejar practicas a 110v, salida para un foco a 110v; además de sus respectivos pines de polarización.

3.3.3.2.2. Detalle de pines en el módulo

Figura 23. Detalle de pines del módulo n° 2



Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

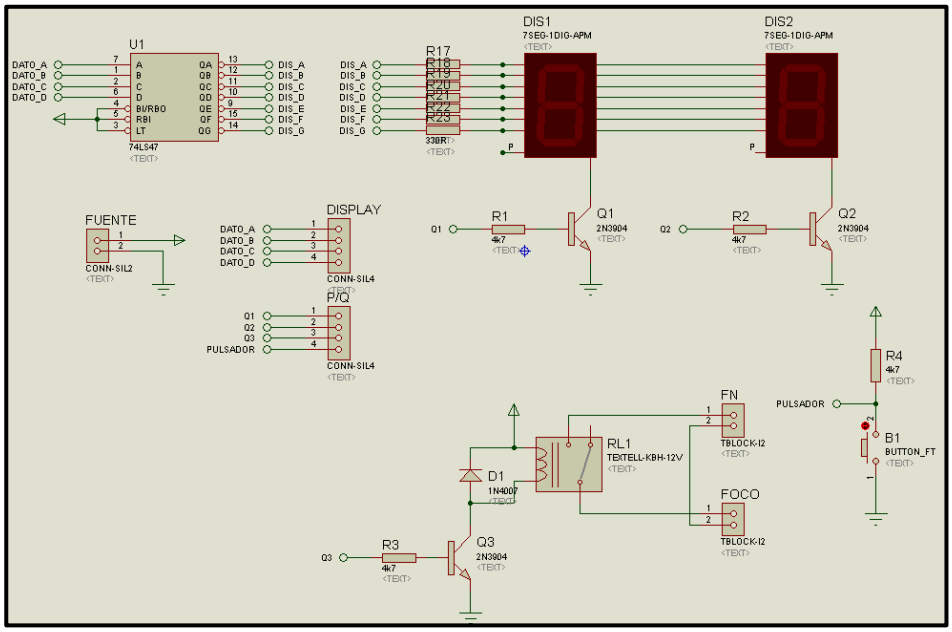
3.3.3.2.1. Uso del módulo nº2

La polarización debe ser de 5v, así el integrado 7448 (decodificador BCD) puede funcionar. El módulo cuenta con un pulsador, el que puede ser utilizado para enviar un bit hacia el microcontrolador. El puerto DISPLAY sirve para encender uno o ambos display presentes, si se requiere encender ambos realizando un contador por ejemplo, se debe utilizar los transistores Q1, Q2. El relé permite realizar prácticas de potencia con el microcontrolador, posee un transistor Q3 que lo activa o desactiva. El puerto PUERTOS permite controlar los tres

transistores presentes y da la salida del pulsador. Para la práctica de potencia se tiene un conector de 110v y un zócalo para conectar las líneas de fase, neutro y retorno.

3.3.3.2.2. Esquemático de pines en el módulo

Figura 24. Detalle de pines del módulo n° 2

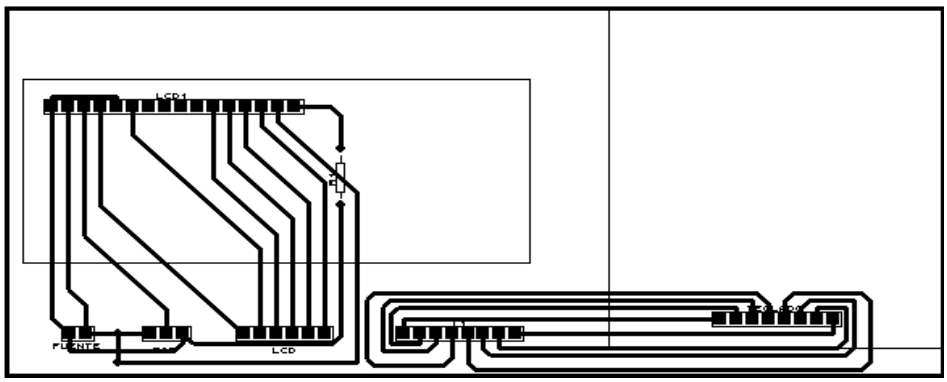


Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.3. Diseño del módulo nº3

Figura 25. Pistas del módulo n° 3



Fuente: Ares

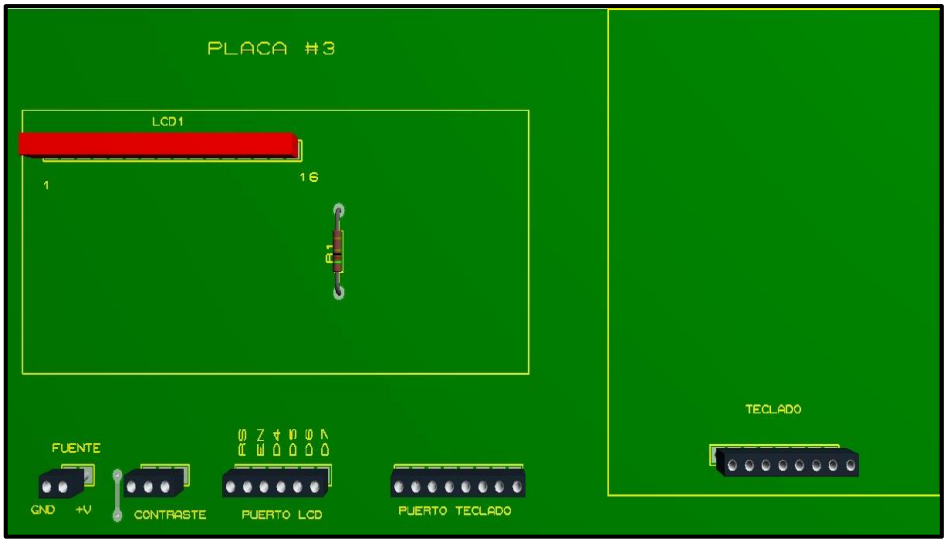
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.3.1. Descripción del módulo

Este módulo tiene un uso muy específico, servirá en casi todas las prácticas ya que controlará el LCD y el TECLADO. Se tiene prácticas exclusivas para LCD y TECLADO. Posee un potenciómetro que controla el contraste en la pantalla.

3.3.3.3.2. Detalle de pines en el módulo

Figura 26. Detalle de pines del módulo n° 3



Fuente: Ares

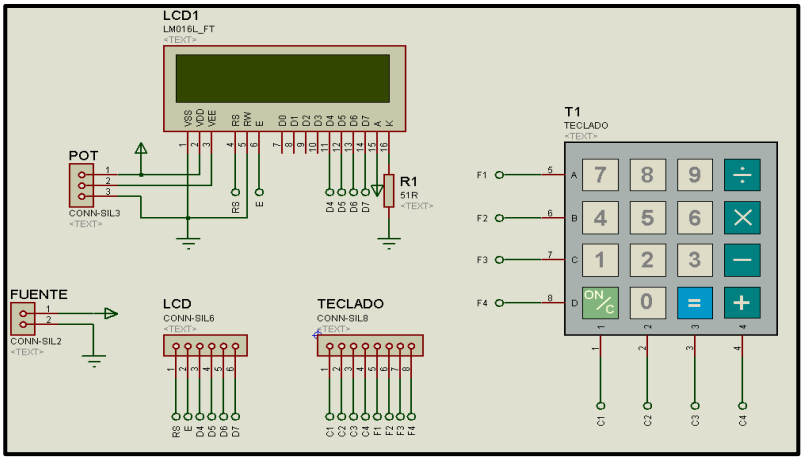
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.3.1. Uso del módulo nº3

Para polarizar el módulo se necesitará 5V. Al puerto CONTRASTE se conecta un potenciómetro. El puerto LCD es en donde se conectarán las líneas para configurar y mostrar mensajes en la pantalla, desde el microcontrolador. El puerto TECLADO permite conectar pines del microcontrolador con pines en el teclado para realizar el barrido (teclado 4x4).

3.3.3.3.4. Esquemático de pines en el módulo

Figura 27. Esquemático del módulo n° 3

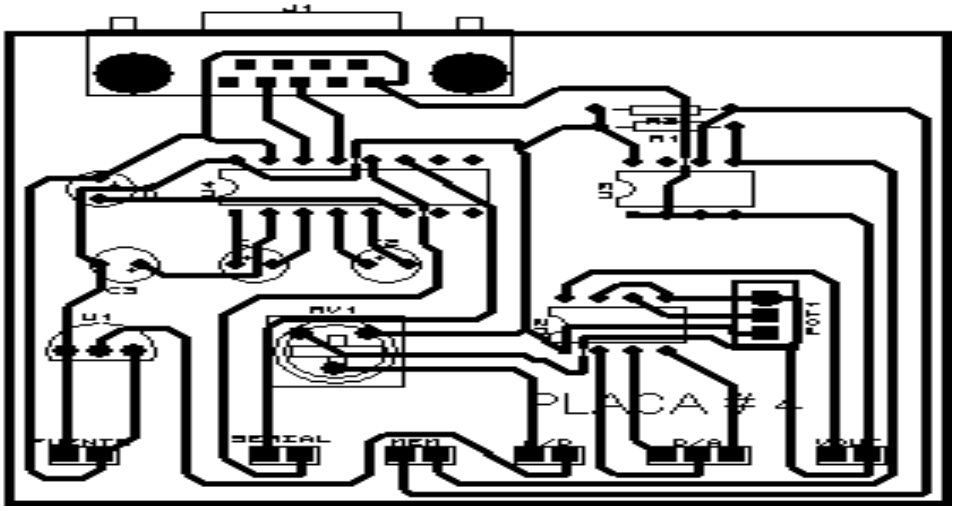


Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.4. Diseño del módulo nº4

Figura 28. Pistas del módulo n° 4



Fuente: Ares

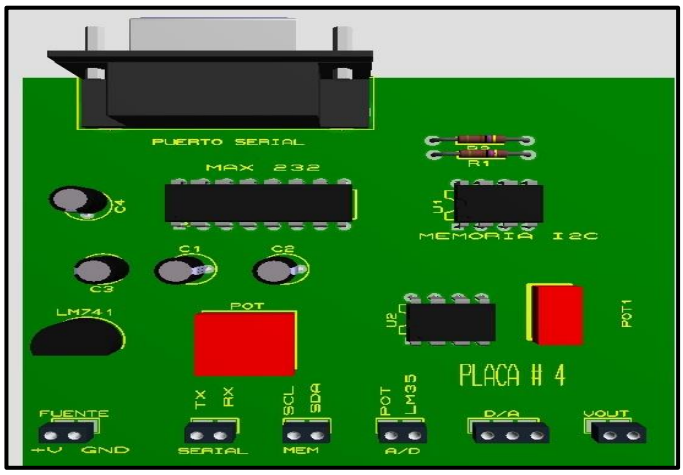
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.4.1. Descripción del módulo

El siguiente módulo provee al practicante de un puerto serial, zócalo para una memoria I2C, zócalo para un DAC (no se utiliza), pines para conexión de las líneas TX, RX, SCL, SDA. Además de un sensor de temperatura y un potenciómetro para prácticas con el ADC

3.3.3.4.2. Detalle de pines en el módulo

Figura 29. Detalle de pines del módulo n°4



Fuente: Ares

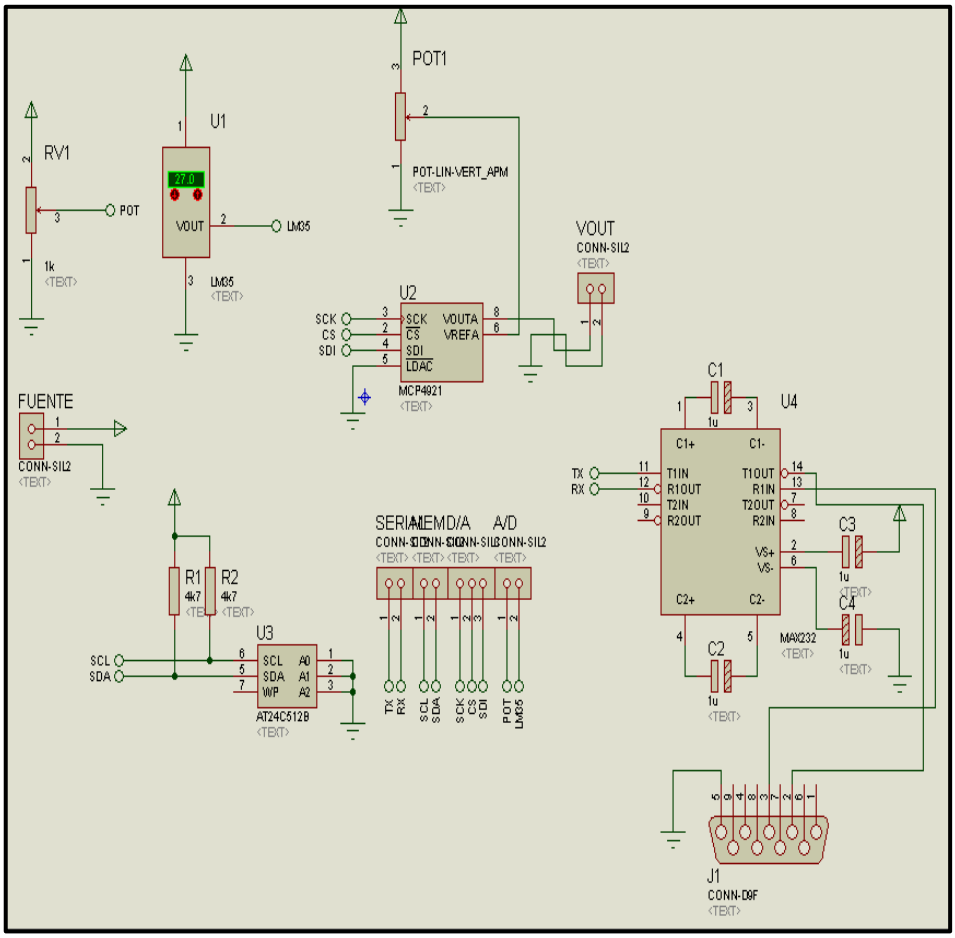
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.4.1. Uso del módulo nº4

La polarización será de 5V necesaria para activar los integrados presentes. El puerto SERIAL permite la conexión de los pines RX, TX tanto en el microcontrolador como en el puerto serial. Con el puerto MEM puedo interconectar las líneas SCL, SDA del micro con la memoria I2C . En el puerto A/D se pueden conectar los pines del ADC al micro, envían señales desde el potenciómetro y otro desde el sensor de temperatura.

3.3.1.4.4. Esquemático de pines en el módulo

Figura 30. Esquemático del módulo n° 4

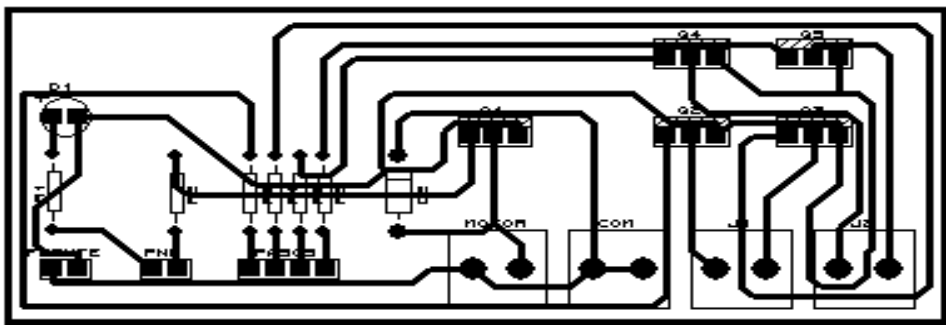


Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.5. Diseño del módulo nº5

Figura 31. Pistas del módulo n° 5



Fuente: Ares

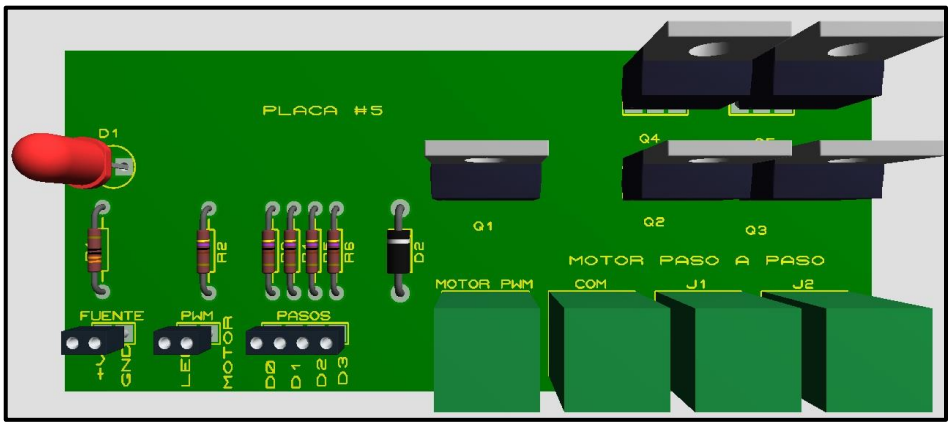
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.1.5.1. Descripción del módulo

Permite realizar prácticas de PWM con un led para mostrar las variaciones de pulso y un puerto donde se puede conectar un motor DC de hasta 12V. Proporciona además conectividad para un motor Paso a Paso de hasta 12V, que es controlado por los puertos J1, J2 y COM.

3.3.1.5.2. Detalle de pines en el módulo

Figura 32. Detalle de pines del módulo n° 5



Fuente: Ares

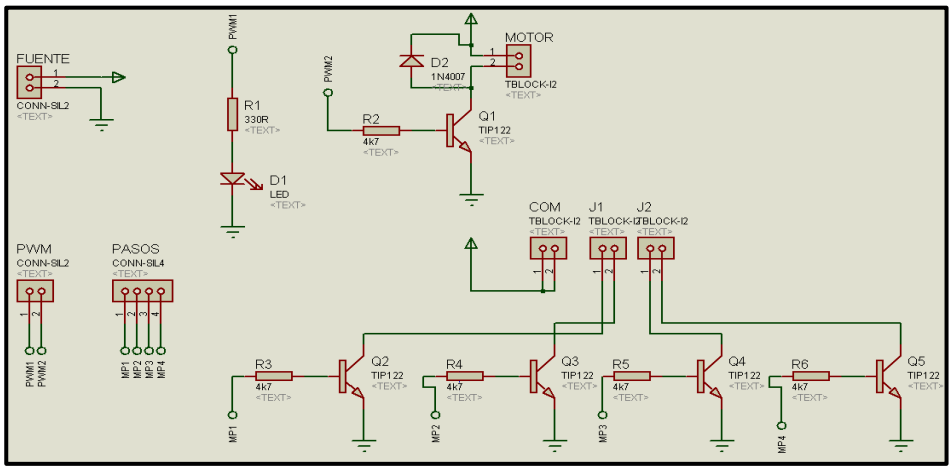
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.5.1. Uso del módulo nº5

La polarización será la que el motor vaya a necesitar (se recomienda un motor de 12v como máximo) ya sea el motor DC para PWM o el motor Paso a Paso. El puerto PWM permite controlar, el LED o el motor. El puerto PASOS controla al motor paso a paso.

3.3.3.5.2. Esquemático de pines en el módulo

Figura 33. Esquemático del módulo n° 5

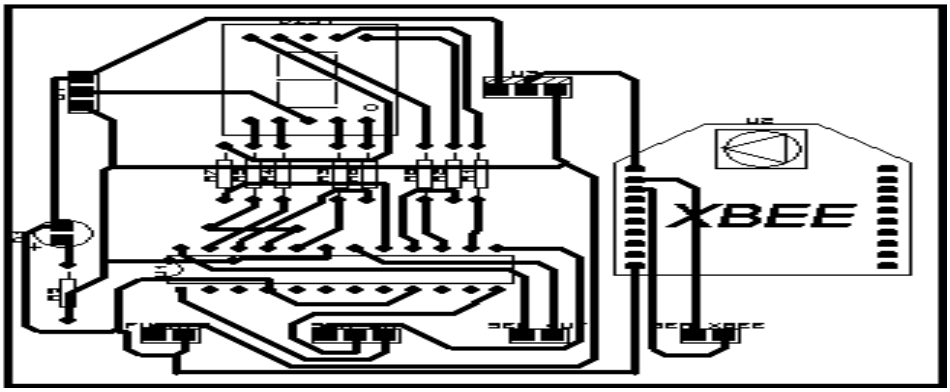


Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.6. Diseño del módulo nº6

Figura 34. Pistas del módulo n° 6



Fuente: Ares

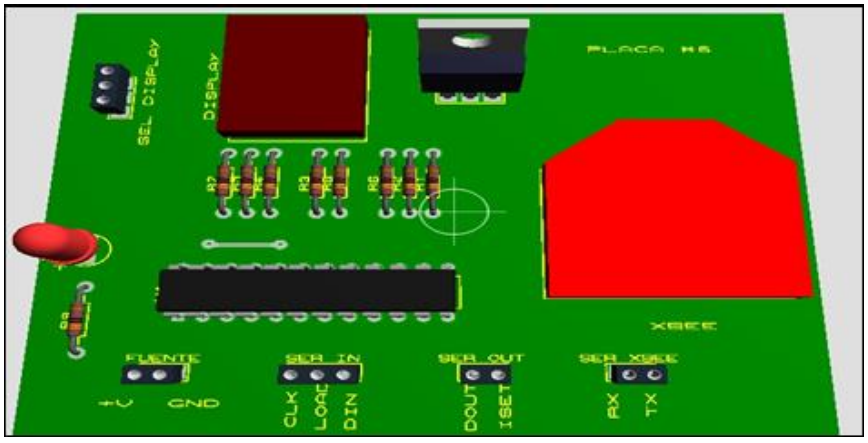
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.1.6.1. Descripción del módulo nº6

El propósito: permitir que el practicante utilice el protocolo de comunicación SPI y además que realizar prácticas con módulos de radio frecuencia.

3.3.1.6.2. Detalle de pines en el módulo nº6

Figura 35. Detalle de pines del módulo n° 6



Fuente: Ares

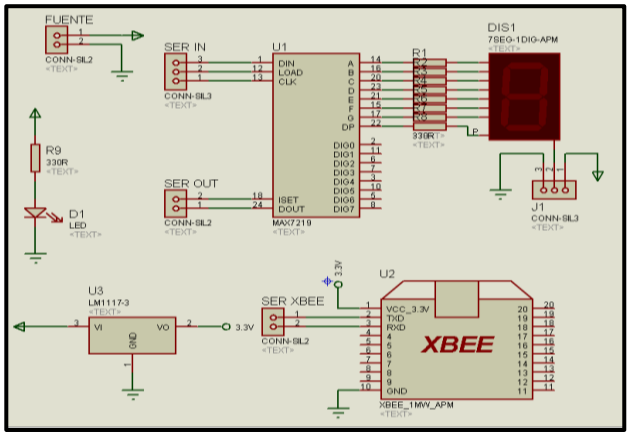
Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.6.1. Uso del módulo nº6

Se necesita 5V para alimentación general de los integrados, los puertos SER IN, SER OUT son para uso del protocolo SPI. El puerto SER XBEE es para comunicación por radio frecuencia.

3.3.3.6.2. Esquemático de pines en el módulo

Figura 36. Esquemático del módulo n° 6

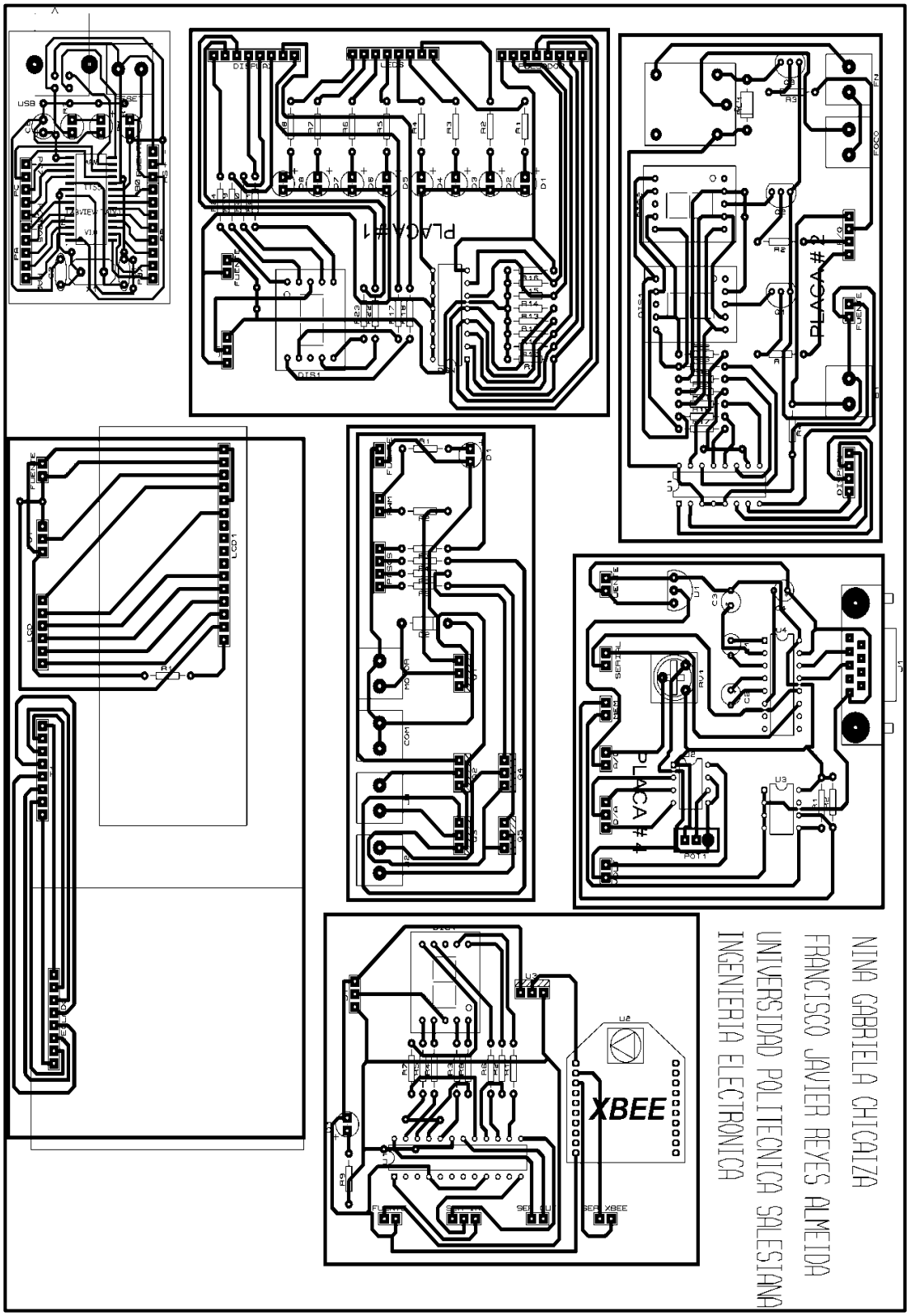


Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.7. Diseño de la tarjeta de entrenamiento

Figura 37. Pistas del módulo

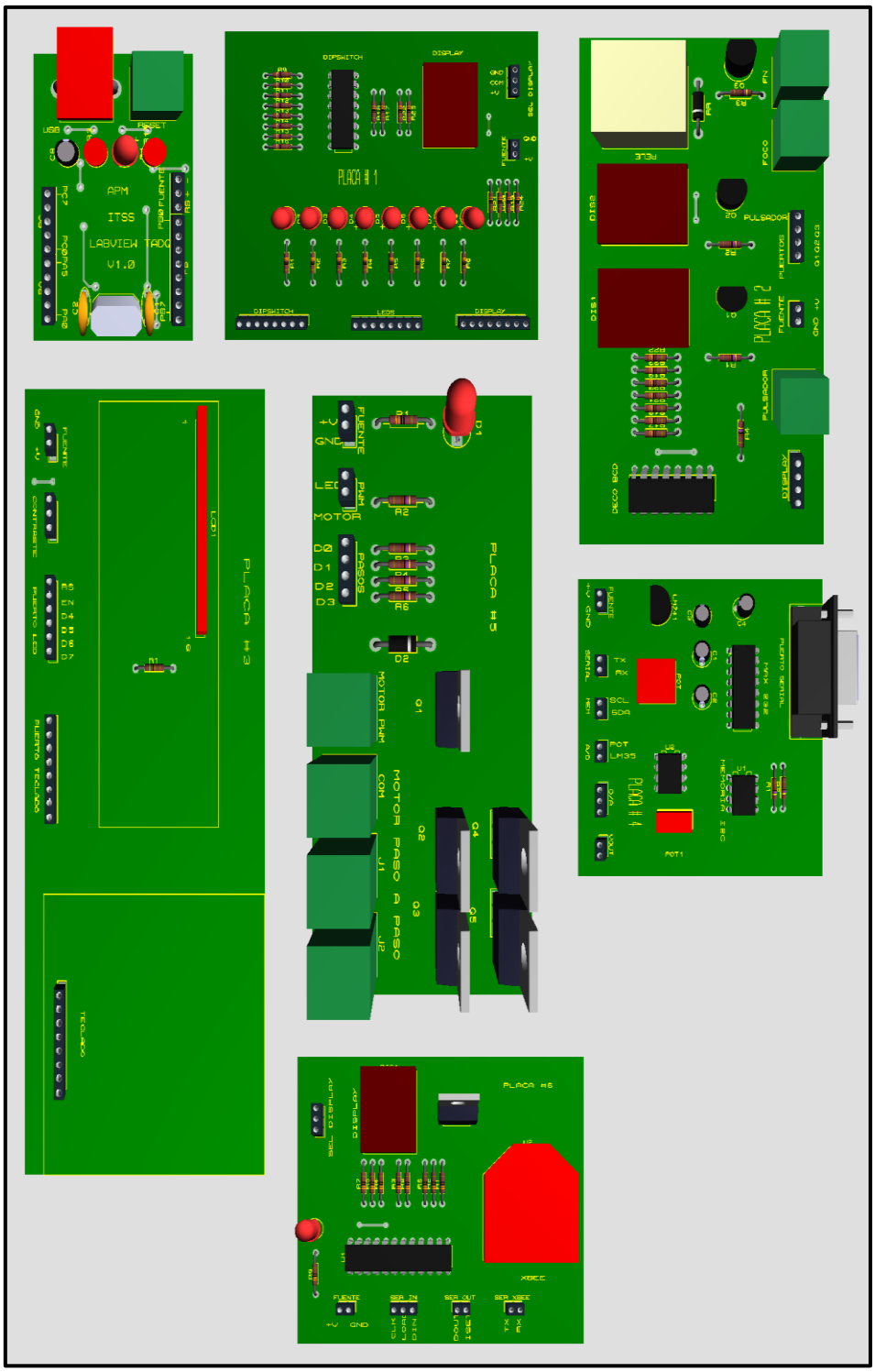


Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.3.7.1. Detalle de pines en la tarjeta:

Figura 38. Diseño completo del módulo

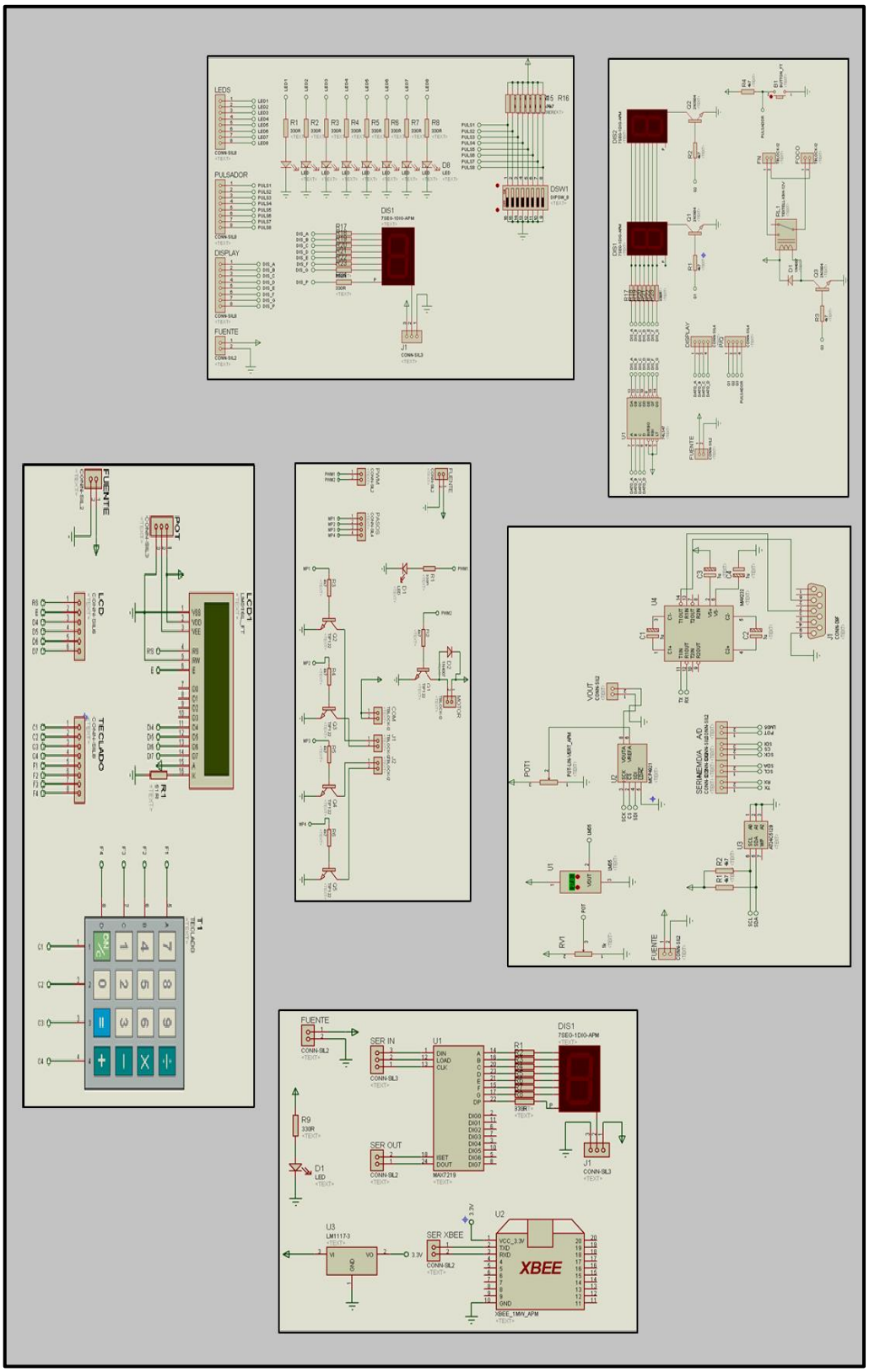


Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

3.3.1.7.2. Esquemático de pines en la tarjeta

Figura 39. Esquemático del módulo n° 7



Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

3.4. Costos del proyecto de investigación

3.4.1 Costos de materiales

En esta sección se desglosa los costos de los materiales que han sido utilizados para la construcción final del módulo. Para esto se analiza el costo de los elementos y diseño de hardware y del desarrollo de software.

3.4.1.1 Hardware

En esta sección se describen los materiales que se utilizaron en la elaboración del hardware del presente proyecto, incluyendo los costos de cada uno.

Tabla 2. Costos de hardware

DESCRIPCIÓN	CANT.	V UNIT.	TOTAL
Resistencia 51 1/4 de vatio	1	0.03	0.03
Resistencia 330 1/4 de vatio	34	0.03	1.02
Resistencia 4.7k 1/4 de vatio	19	0.03	0.57
Regulador 5Vcc 7805	2	0.50	1.00
Regulador 18Vcc 7818	2	0.50	1.00
Transistor NPN TIP122	5	0.50	2.50
Capacitor 10 micro	5	0.35	1.75
Capacitor 20 pico	2	0.35	0.70
Diodo 1n4007	2	0.15	0.30
Diodo LED	11	0.15	1.65
Display 7 segmentos ánodo común	4	0.80	3.20
Dipswitch 8 segmentos	1	0.75	0.75
Compuerta 74LS47	1	0.45	0.45
Transistor 2N3904	3	0.70	2.10
Pantalla LCD	1	8.00	8.00
Teclado Matricial 4*4	1	4.00	4.00
Sensor LM35	1	1.75	1.75
Trimmer 5k	1	0.75	0.75
Potenciómetro de precisión 10k	1	0.75	0.75
Potenciómetro de precisión 5k	1	0.75	0.75
Relé 5v simple	1	0.60	0.60
Regulador LM 1117	1	0.80	0.80

Max 232	1	1.20	1.20
Max 7219	1	15.00	15.00
Cristal 12 MHz	1	0.50	0.50
Baquelita 25cm x 20cm	1	3.25	3.25
Pulsador 2p	2	0.35	0.70
Mcp 4921	1	0.95	0.95
CMOS 24C08	1	1.10	1.10
Xbee	2	38.00	76.00
Xbee explorer	1	7.00	7.00
Par zócalos Xbee	1	0.15	0.15
Zócalo 14 pines	1	0.40	0.40
Zócalo 16 pines	1	0.50	0.50
Zócalo 8 pines	2	0.20	0.20
Zócalo 24 pines	1	0.75	0.75
Conector USB tipo 1	1	0.75	0.75
Motor paso a paso unipolar	1	1.50	1.50
Conector DB9 hembra	1	0.95	0.95
Conector cable 2 pines	3	0.20	0.60
Conector cable 1 pin	5	0.20	1.00
Conector cable 4 pines	3	0.30	0.90
Cable UTP por metros	1	0.45	0.45
Cables para fuente de alimentación	2	0.55	1.10
Bornera 2 pines	6	0.30	1.80
Módulo Microcontrolador AVR XMega	1	65.00	65.00
Impresión de placa en cobre	1	75.00	75.00
		TOTAL	**$ 291.17**

Elaborado por: Francisco Reyes y Nina Chicaiza

3.4.2 Costos de diseño de hardware

Los costos del diseño del hardware del presente proyecto el cual se compone de la circuitería electrónica y el diseño de cada una de las placas para el módulo, incluyendo los recursos necesarios que se han utilizado para la elaboración estos, desde el diseño de los prototipos previos, hasta llegar al diseño final del hardware se estiman en 150 dólares USA.

3.4.3 Costos de desarrollo de software

Los costos del desarrollo del software del presente proyecto en Bascom para AVR incluyendo los recursos necesarios para su elaboración ascienden a 150 dólares USA.

3.4.4 Mano de obra

En esta sección se desglosa los costos de mano de obra tanto del software como del hardware del presente proyecto.

Tabla 3. Costos de mano de obra

DESCRIPCIÓN	**COSTO**
Construcción del Módulo	$ 75.00
Desarrollo de software por los tesistas	$ 100.00
TOTAL	**$ 175.00**

Elaborado por: Francisco Reyes y Nina Chicaiza

3.4.5 Costo total del equipo

En esta sección se recogen los costos totales de los materiales, diseño del hardware, desarrollo de software y mano de obra para saber cuál es el costo total de equipo del presente proyecto de investigación.

Tabla 4. Costo total de la tarjeta

DESCRIPCIÓN	COSTO
Materiales	
Hardware	$ 216.17
Construcción de hardware	$ 75.00
Desarrollo de software	$ 100.00
Mano de obra	$ 75.00
TOTAL	$ 466,17

Elaborado por: Francisco Reyes y Nina Chicaiza

3.4.6 Análisis del valor actual neto (VAN) y tasa interna de retorno (TIR)

Tabla 5. Análisis de VAN y TIR

MES	COSTO	DEPRECIACION DE LOS COSTOS DE LOS ELEMENTOS	VAN	TIR
ENERO	**-466,17**	1,25%	$ 2.208,78	46%
FEBRERO	216,17			
MARZO	216,17			
ABRIL	216,17			
MAYO	216,17			
JUNIO	216,17			
JULIO	216,17			
AGOSTO	216,17			
SEPTIEMBRE	216,17			
OCTUBRE	216,17			
NOVIEMBRE	216,17			
DICIEMBRE	216,17			

Elaborado por: Francisco Reyes y Nina Chicaiza

El costo de la inversión inicial se la detalla en el mes de enero, como la totalidad del costo del proyecto. Lo que se va descontando mes a mes por un año, es el ahorro de costos en dispositivos para las prácticas, dado que la tarjeta de entrenamiento provee todo el hardware necesario, el porcentaje se estima a partir del decremento del costo de los elementos necesarios para la construcción de la tarjeta de entrenamiento. Por tanto se deduce gracias a los indicadores que el proyecto es viable.

CAPÍTULO 4

ANÁLISIS DE RESULTADOS

4.1. Introducción

En este capítulo se analizará la funcionalidad del módulo con el microcontrolador XMega, para ello se realizaron pruebas con el módulo de entrenamiento, de esta manera se pretende identificar si existe algún error y a su vez comprobar que tanto el software como el hardware diseñados para cada aplicación estén realizados de la forma correcta.

4.2. Pruebas en el módulo de entrenamiento

4.2.1. Manejo de puertos

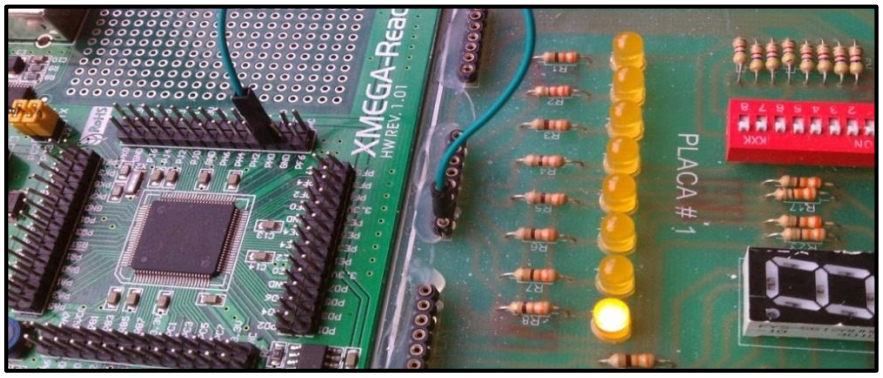
Manejo de Entradas y Salidas en el microcontrolador XMega de Atmel, para ésta práctica se tiene que utilizar la Placa #1 del Módulo de Entrenamiento.

1) Escribir en el puerto H el valor 1h

Se requiere un solo pin de puerto para esta práctica, el pin utilizado es el H.0; es posible configurarlo de varias maneras, la seleccionada es:

Ddrh.0 = 1

Figura 40. Valor 1H escrito en el puerto H



Elaborado por: Francisco Reyes y Nina Chicaiza

Se logra el mismo resultado con las siguientes configuraciones de puerto:

Ddrh = &b00000001 (solo el pin H.0 será salida)

Ddrh = &H01 (solo el pin H.0 será salida)

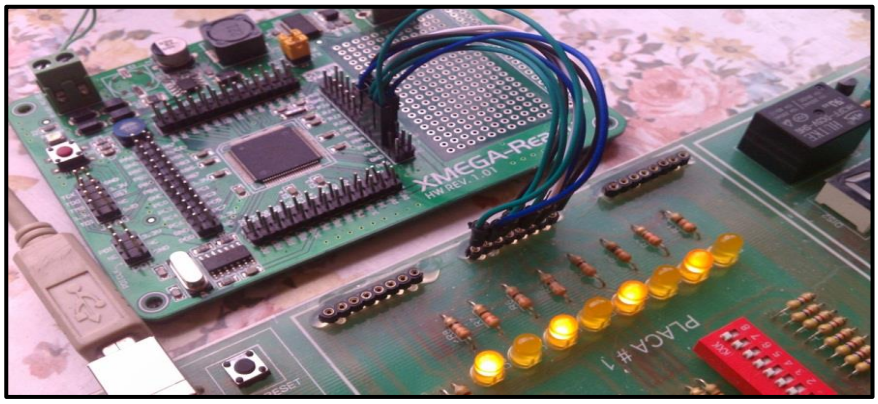
Ddrh = 1 (solo el pin H.0 será salida)

2) Escribir en el puerto H el valor 55h

Se utilizan todos los pines del puerto H, para el efecto se selecciona la siguiente configuración:

Ddrh = 255 (todos los pines en 1)

Figura 41. Valor 55H escrito en el puerto H



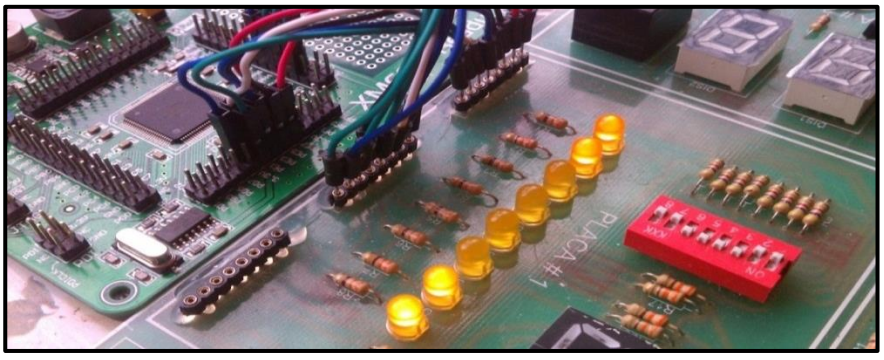
Elaborado por: Francisco Reyes y Nina Chicaiza

3) Leer en el puerto E y escribir en el puerto H.

Para esta práctica se utilizan dos puertos en el microcontrolador, el puerto para manejar las salidas y el puerto para leer las entradas desde el dipswitch.

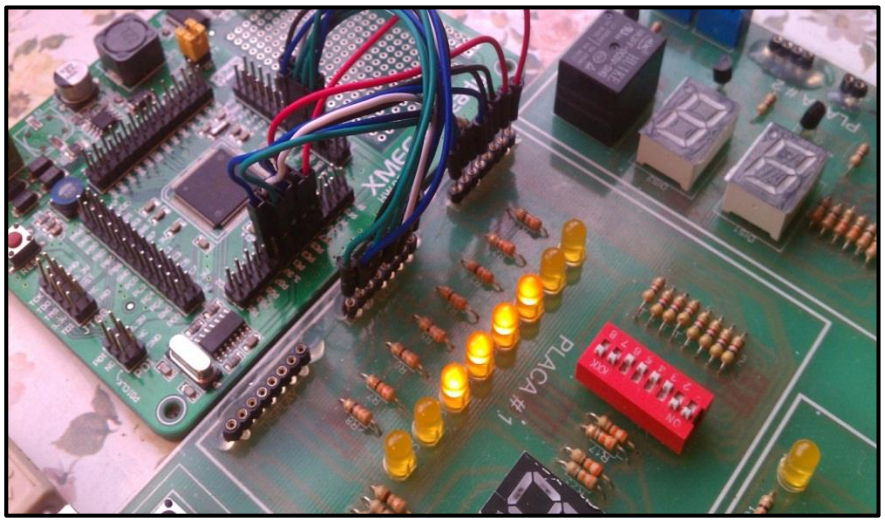
El puerto H sirve para indicar las salidas y el puerto E para leer las entradas

Figura 42. Valor 11000011 escrito en el puerto H y leído en el puerto E



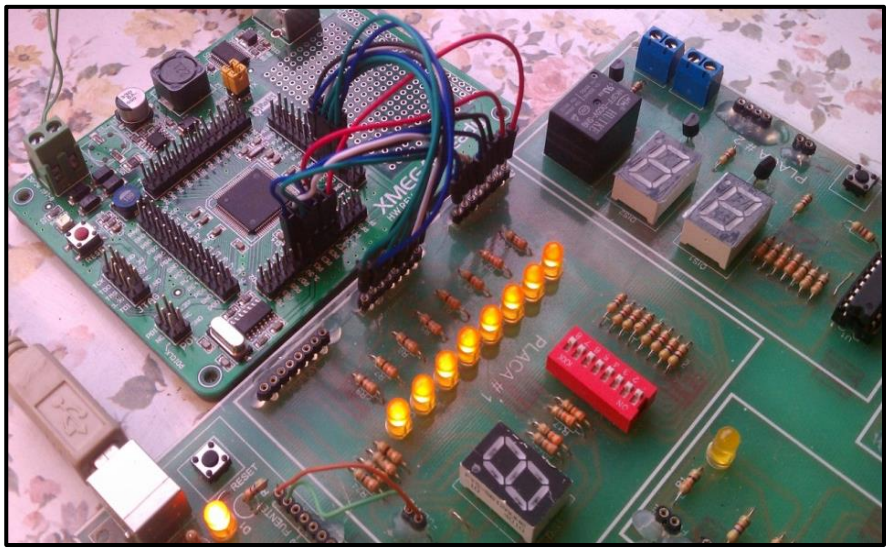
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 43. Valor 00111100 escrito en el puerto H y leído en el puerto E



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 44. Valor 11111111 escrito en el puerto H y leído en el puerto E

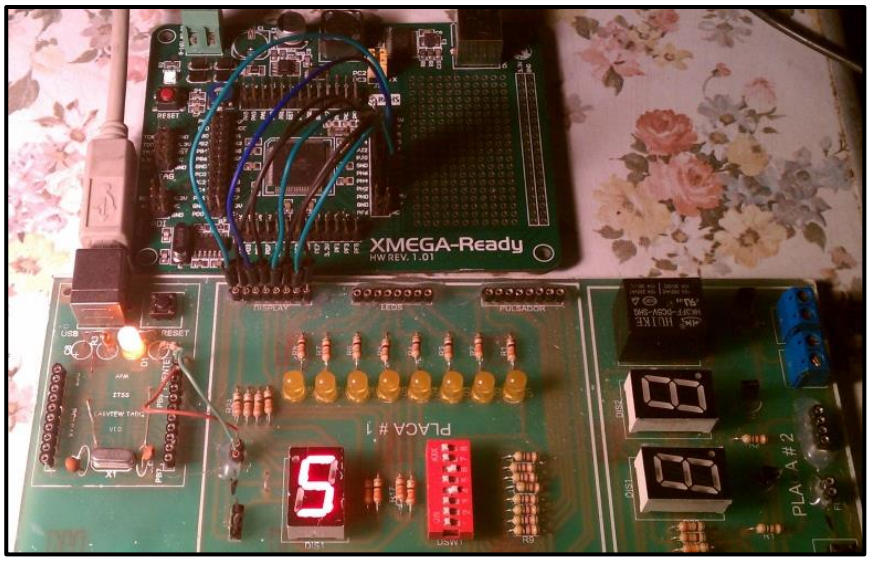


Elaborado por: Francisco Reyes y Nina Chicaiza

4) Escribir en un display de 7 segmentos sin utilizar decodificador, un valor ascendente entre 0 y F

Para esta práctica se utilizan los pines del puerto H, configurándolos como salidas para controlar los leds en el display.

Figura 45. Valor ascendente entre 0 y F



Elaborado por: Francisco Reyes y Nina Chicaiza

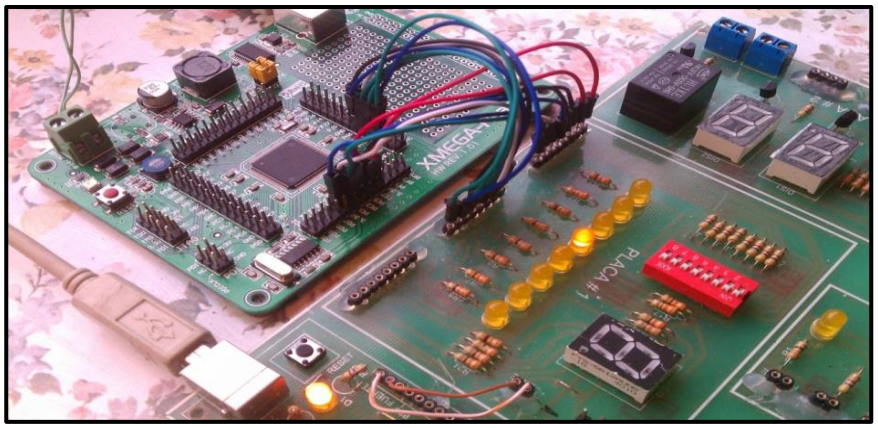
5) Lógica booleana con el microcontrolador.

Para realizar ésta práctica se utilizan los pines del puerto E para las entradas desde el dipswitch hacia el microcontrolador, los pines del puerto H para las salidas hacia los leds. Fueron necesarios 3 leds y 5 puertos del dipswitch

Se simularon las siguientes compuertas:

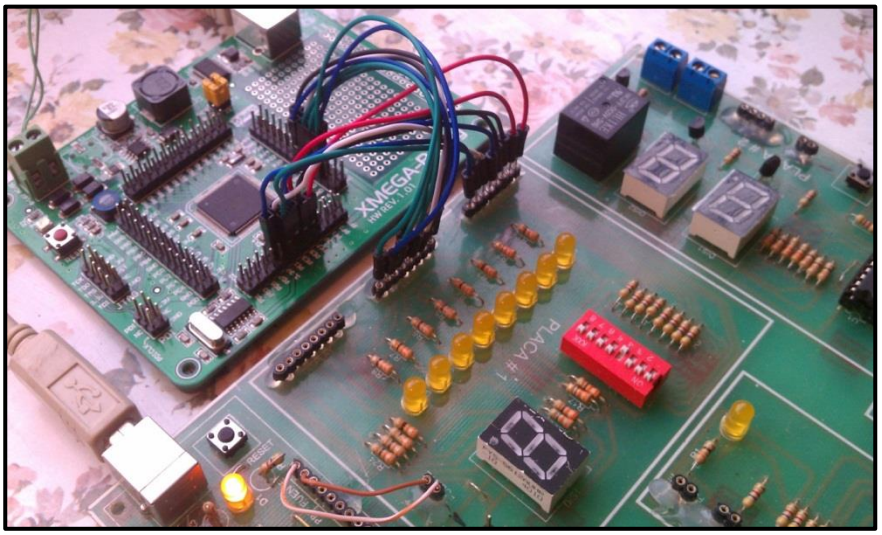
NOT, en el dipswitch el pin 4 se utilizó para simular la compuerta. El led Nº4 muestra el funcionamiento de la compuerta.

Figura 46. Compuerta NOT encendida.



Elaborado por: Francisco Reyes y Nina Chicaiza

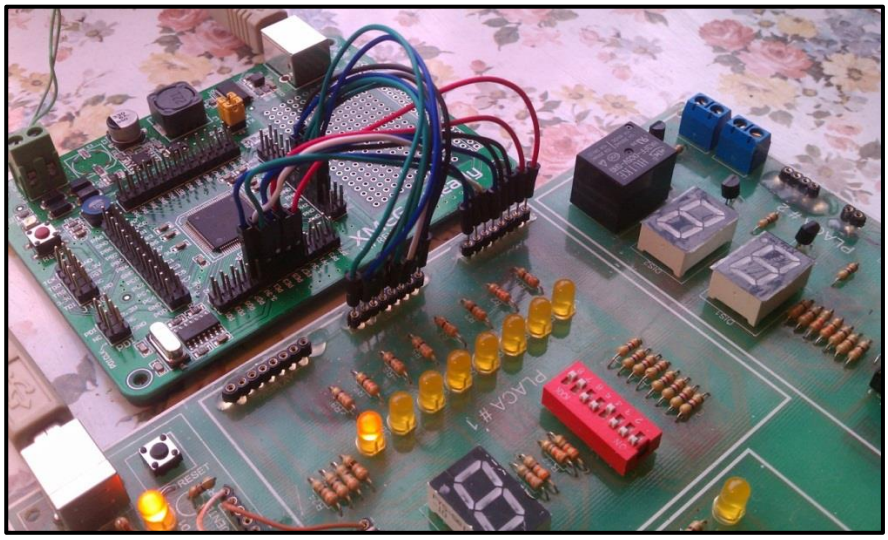
Figura 47. Compuerta NOT apagada



Elaborado por: Francisco Reyes y Nina Chicaiza

AND, en el dipswitch los pines 7 y 8 se utilizaron para simular la compuerta. El led Nº8 se enciende cuando los puertos 7- 8 tengan un nivel de voltaje alto.

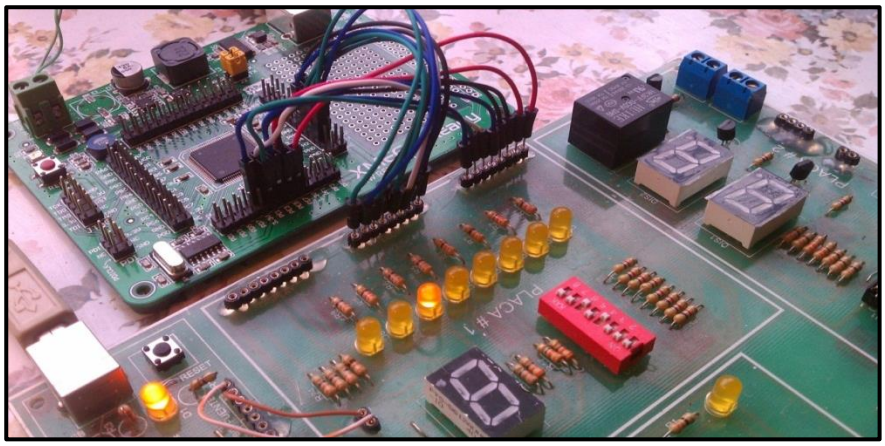
Figura 48. Compuerta AND encendida



Elaborado por: Francisco Reyes y Nina Chicaiza

OR, en el dipswitch los pines 5 y 6 se utilizaron para simular la compuerta. El led Nº6 se activa cuando los pines 5-6 en el dipswitch tengan un voltaje superior a 3.3v

Figura 49. Compuerta OR encendida



Elaborado por: Francisco Reyes y Nina Chicaiza

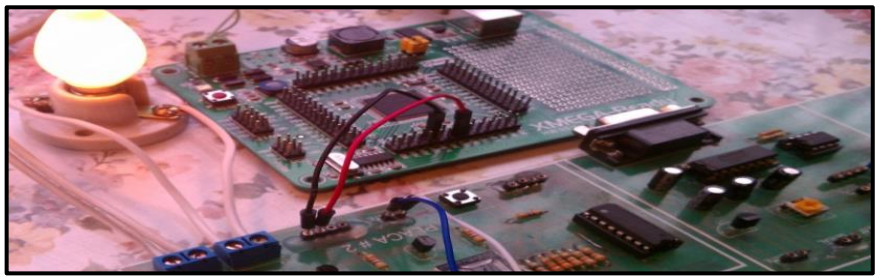
4.2.2. Uso de declaraciones

Utilizar las declaraciones if, select case, do, for, while con el microcontrolador XMega de Atmel, para ésta práctica se tiene que utilizar la Placa #2 del Módulo de Entrenamiento.

1) Si se presiona el botón el foco se enciende por 3 segundos, luego se apaga 1 segundo y se vuelve a encender el foco por 3 segundos. Caso contrario el foco permanece prendido

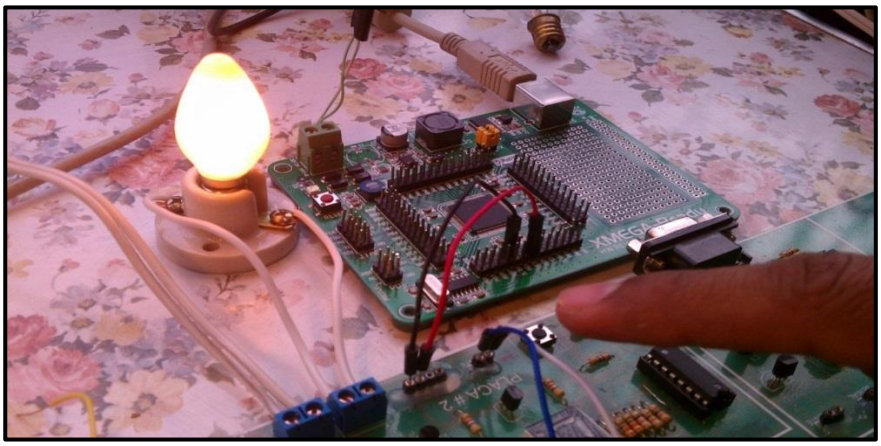
Se utiliza del puerto E el pin E.3 como salida, para controlar el transistor que activa el Relé y el pine E.0 lee la entrada del botón para iniciar la secuencia de encendido y apagado del foco.

Figura 50. Paso 1, conexión de pines para entrada y salida



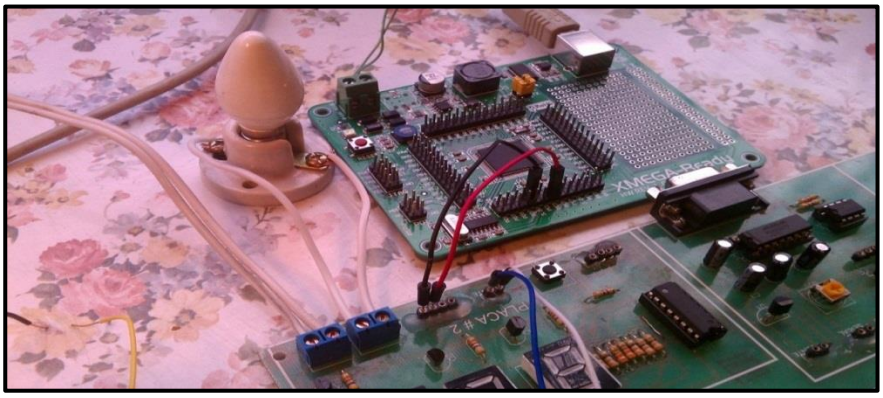
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 51. Paso 2, se pulsa el botón y el foco se encenderá 3 segundos



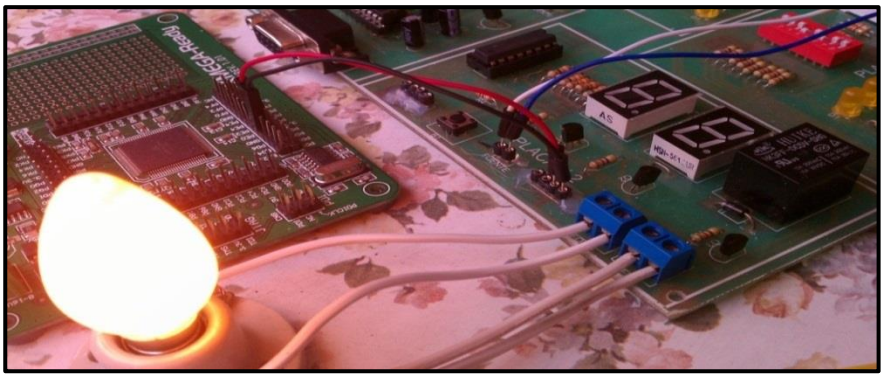
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 52. Paso 3, el foco se apagará por 1 segundo



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 53. Paso 4, el foco se vuelve a encender por 3 segundos

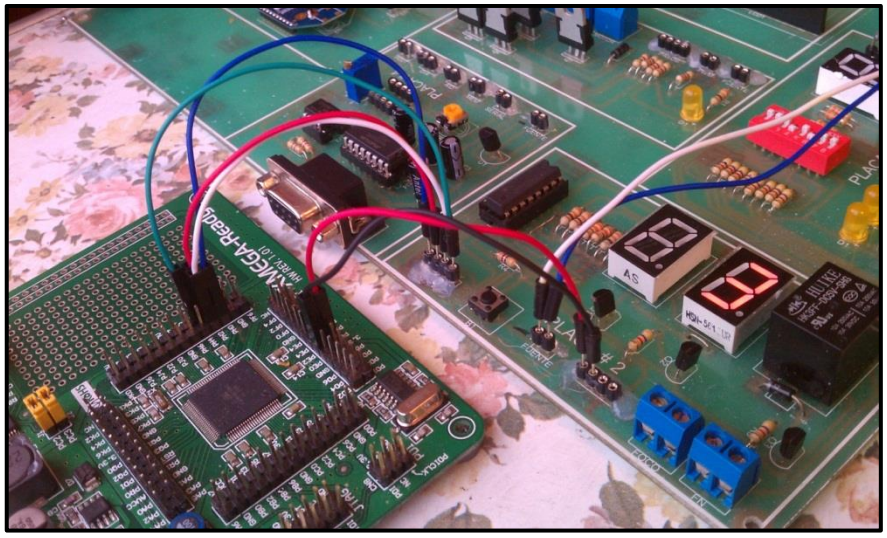


Elaborado por: Francisco Reyes y Nina Chicaiza

2) Utilizando la sentencia For –Next, elaborar un contador ascendente de 0 a 9, que sea repetitivo.

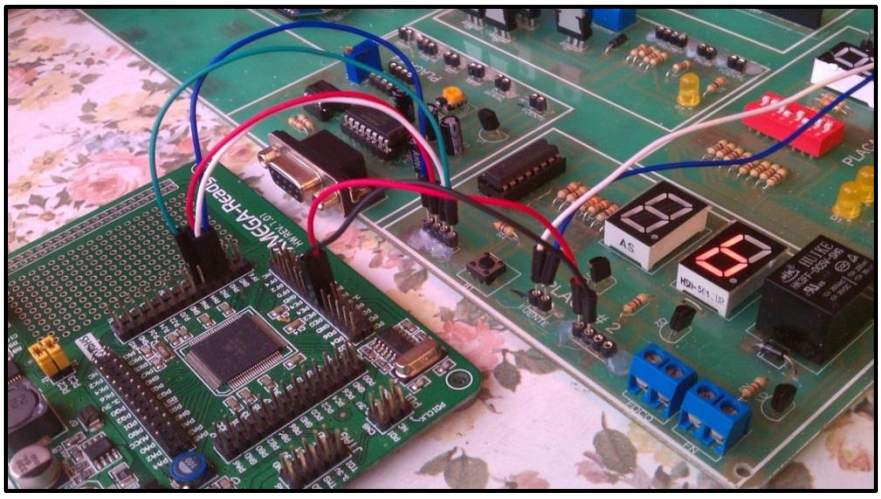
Se utiliza un decodificador BCD por lo que se ocupó únicamente los pines 0-3 del puerto H; para controlar los display se tiene salidas hacia dos transistores uno para las unidades y otro para las decenas para ello se utilizan los pines E.1 para las decenas y E.2 para las unidades.

Figura 54. Contador Ascendente, se muestra el 3



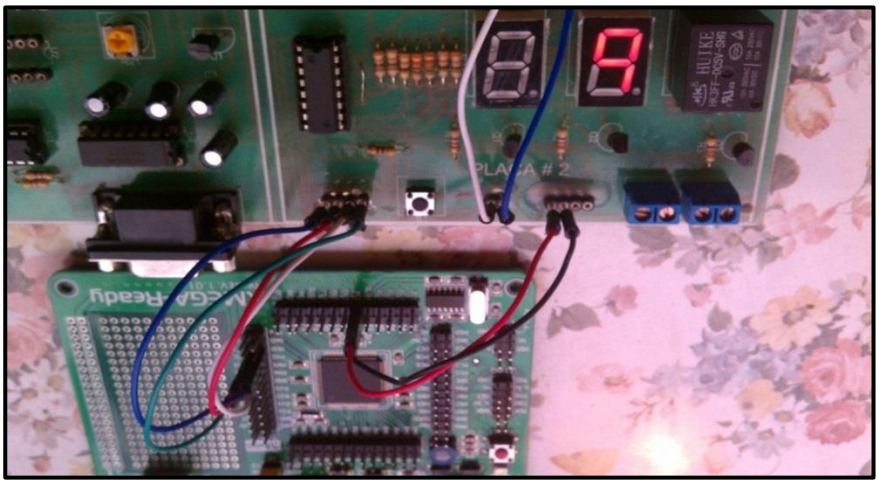
Fuente:
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 55. Contador Ascendente, se muestra el 6



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 56. Contador ascendente, muestra el 9

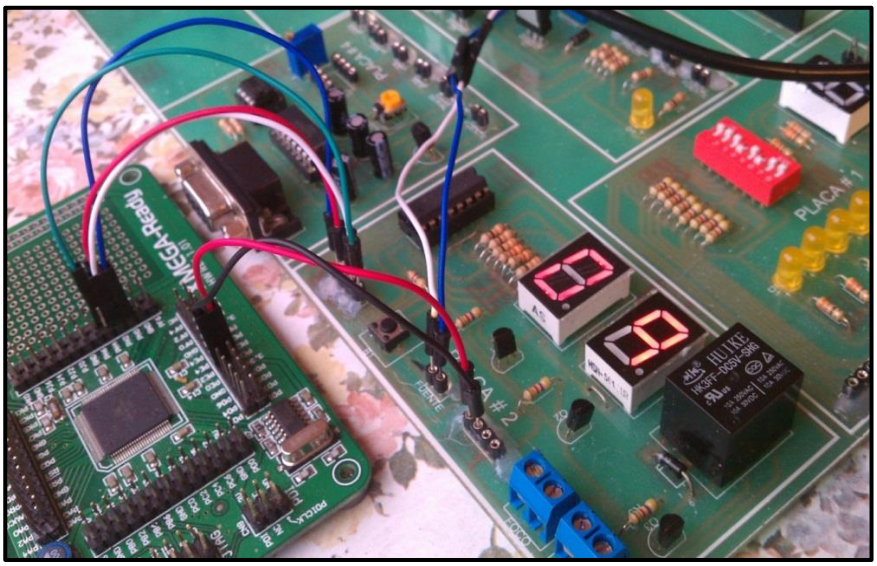


Elaborado por: Francisco Reyes y Nina Chicaiza

3) Utilizar la secuencia While, para elaborar un contador ascendente de dos dígitos.

Se ocupan los mismos puertos que el contador de unidades

Figura 57. Contador ascendente de dos dígitos, se muestra 09



Elaborado por: Francisco Reyes y Nina Chicaiza

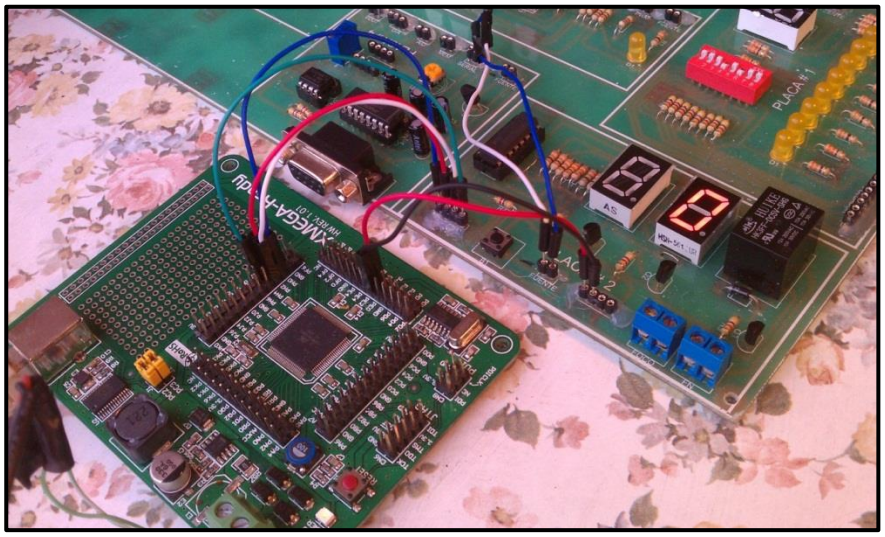
4.2.3. Temporizadores

Utilizar los temporizadores del microcontrolador XMega de Atmel, para ésta práctica se tiene que utilizar la Placa #2 del Módulo de Entrenamiento.

1) Utilizar el temporizador para elaborar un contador de 1 segundo, cuyo valor se observa en un display de 7 segmentos

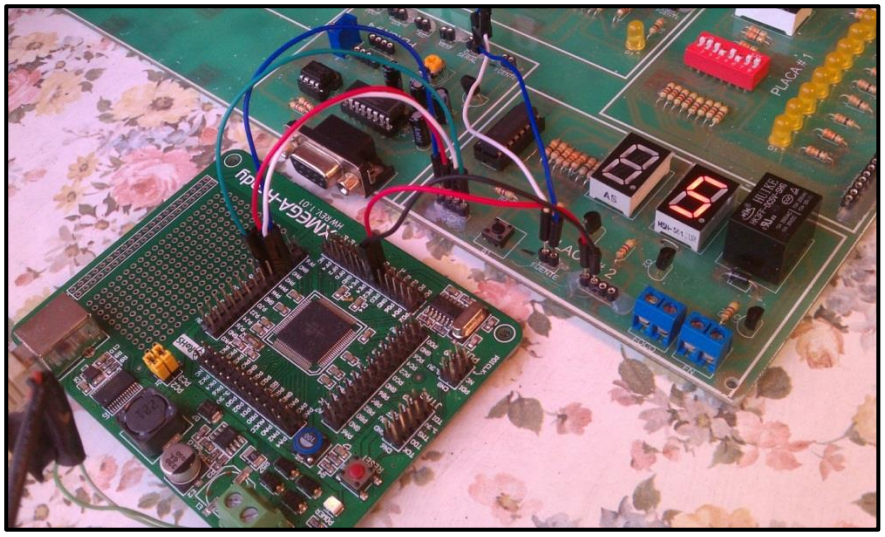
Se utilizó el decodificador BCD, el retardo entre cada dígito en el display está dado por el registro TCC0. El puerto que controla al decodificador BCD es el puerto H, el control de los transistores se realiza con el puerto E.

Figura 58. Contador ascendente de 1 segundo con BCD, se muestra 0



Elaborado por: Francisco Reyes y Nina Chicaiza

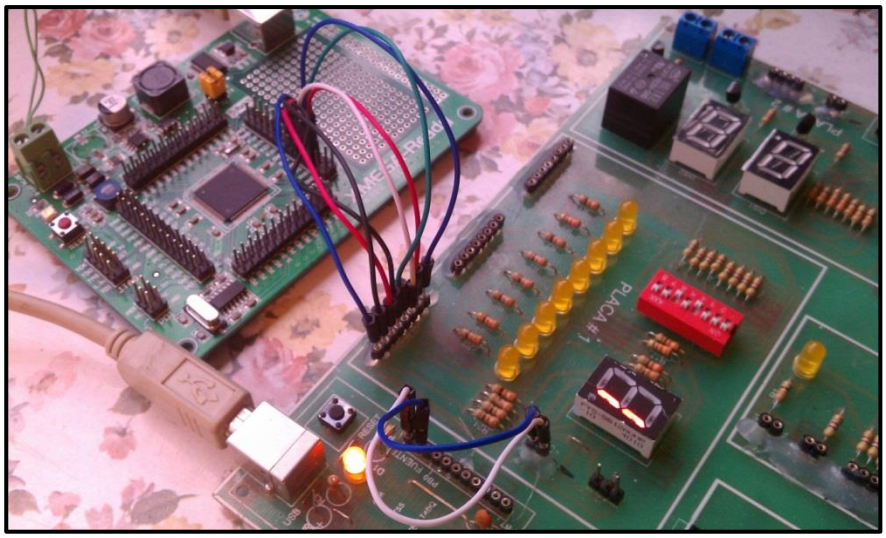
Figura 59. Contador ascendente de 1 segundo sin BCD, se muestra 5



Elaborado por: Francisco Reyes y Nina Chicaiza

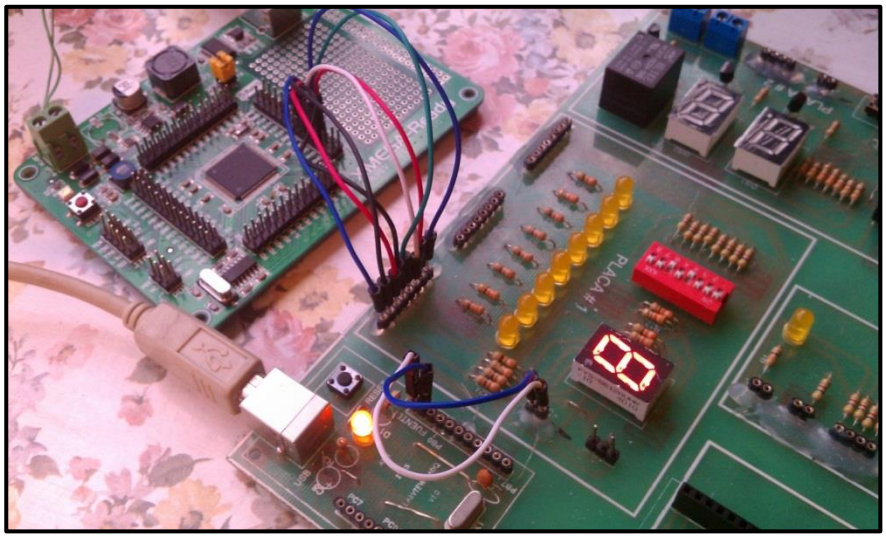
Adicionalmente se realizó una práctica utilizando la Placa #1, ya que ésta sección del módulo no tiene el BCD se ocupa todo el puerto H.

Figura 60. Contador ascendente de 1 segundo sin BCD, se muestra 1



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 61. Contador ascendente de 1 segundo sin BCD, se muestra 8

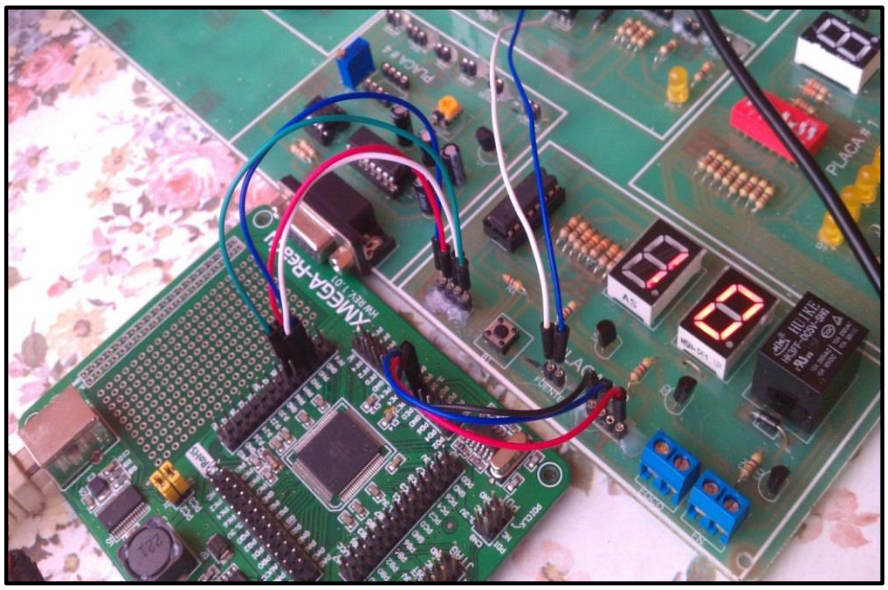


Elaborado por: Francisco Reyes y Nina Chicaiza

2) Elaborar un programa que utilice una entrada, y se visualice el incremento en un display de 7 segmentos.

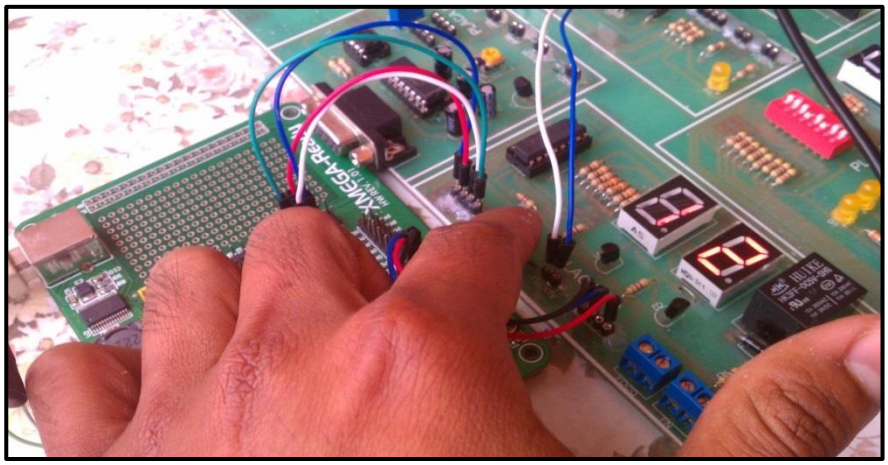
Se utiliza el puerto E.0 para leer la entrada y realizar el incremento en la variable que se muestra en los displays, los transistores fueron controlados con los pines E.1 y E.2. El puerto H se utiliza para controlar las salidas, se ocupó el decodificador BCD por lo tanto solo fueron necesarios 4 pines del puerto H.

Figura 62. Contador ascendente de dos dígitos con interrupción externa



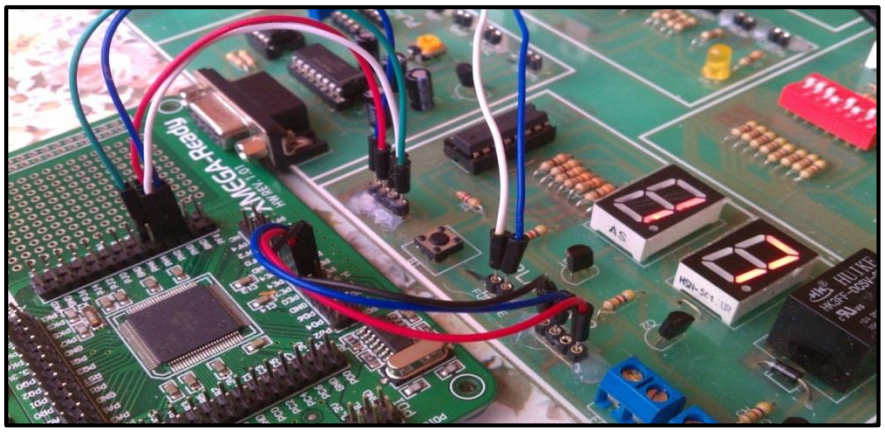
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 63. Contador ascendente de dos dígitos con interrupción externa



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 64. Contador ascendente de dos dígitos con interrupción externa



Elaborado por: Francisco Reyes y Nina Chicaiza

4.2.4. Manejo de lcd

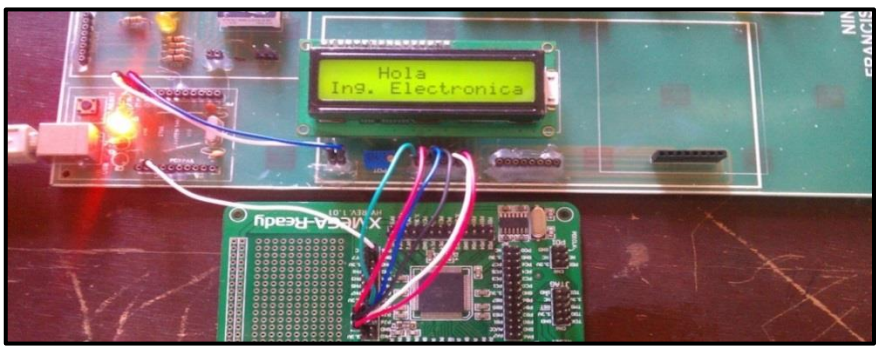
Utilizar con el microcontrolador XMega de Atmel display alfanuméricos, para ésta práctica se tiene que utilizar la Placa #3 del Módulo de Entrenamiento.

1) Escribir la sentencia Hola e Ingeniería Electrónica en un LCD.

Para el manejo de la Pantalla de Cristal Líquido LCD se ocupan en esta práctica los pines 7, 6, 5, 4, 3, 2 del puerto J. Los pines del 7-4 controlan los datos que se imprimirán en la pantalla, los pines 3-2 son pines de control.

NOTA: Es necesario que la línea de tierra del microcontrolador Xmega esté conectada a alguna línea de tierra en la placa de prácticas.

Figura 65. “Hola” e “Ingeniería Electrónica” en el Lcd.

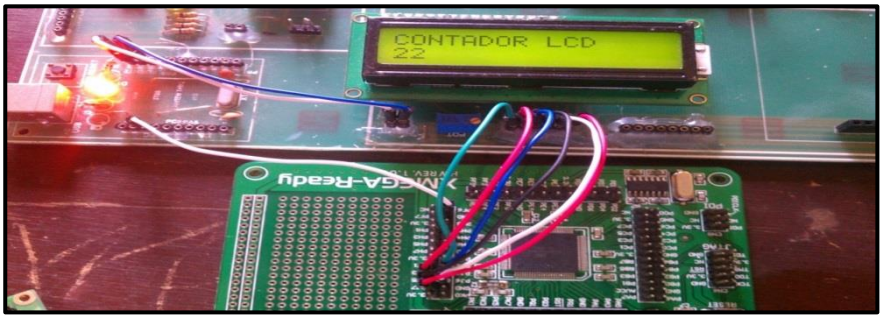


Elaborado por: Francisco Reyes y Nina Chicaiza

2) Elabore un programa para que se observe un contador ascendente, cuyo valor se observa en un display 16x2, con un intervalo de tiempo entre cada valor de 2 segundos.

Se usan nuevamente los pines 7-2 del puerto J, 7-4 pines de datos y 3-2 pines de control.

Figura 66. Contador ascendente en el Lcd



Elaborado por: Francisco Reyes y Nina Chicaiza

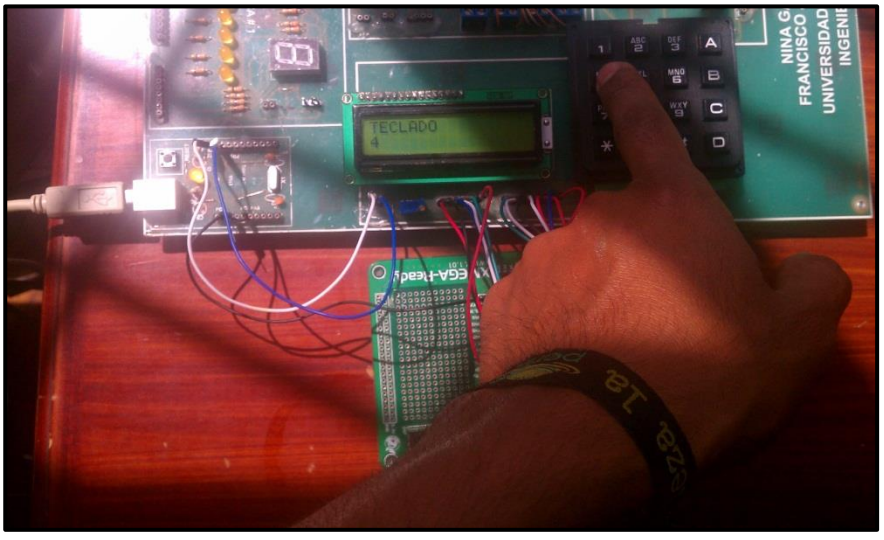
4.2.5. Manejo de lcd y teclado

Utilizar el microcontrolador XMega de Atmel con teclado matricial, para ésta práctica se tiene que utilizar la Placa #3 del Módulo de Entrenamiento.

1) Escribir la sentencia Hola e Ingeniería Electrónica en un LCD.

Utilizar el microcontrolador XMega de Atmel con teclado matricial, para ésta práctica se tiene que utilizar la Placa #3 del Módulo de Entrenamiento.

Figura 67. Lcd y teclado



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 68. Manejo de teclado matricial

Elaborado por: Francisco Reyes y Nina Chicaiza

4.2.6. Adc / Dac

Conversión análoga digital y digital análoga. Utilizar los conversores A/D y D/A del microcontrolador XMega de Atmel, para ésta práctica se tienen que utilizar las Placas #3 y #4 del Módulo de Entrenamiento

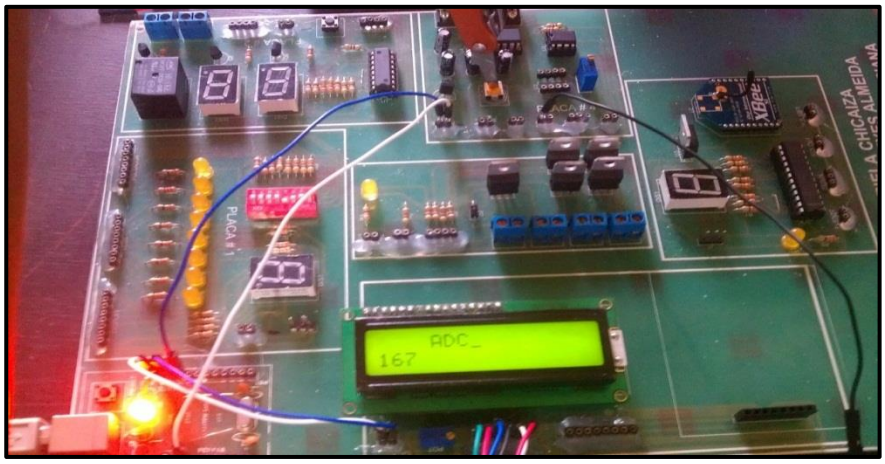
1) Conversión A/D con el microcontrolador XMega de Atmel.

Se adquiere la señal de un potenciómetro entre 0 y 5 voltios, y se observa en el LCD un valor entre 0 y 4095. El puerto J es utilizado para control el LCD; se configuran los registros necesarios para realizar la conversión.

Config Adca inicializa el convertidor en el puerto A.

Getadc (adca, 0), se usa el pin 0 del puerto A para adquirir la señal a convertir.

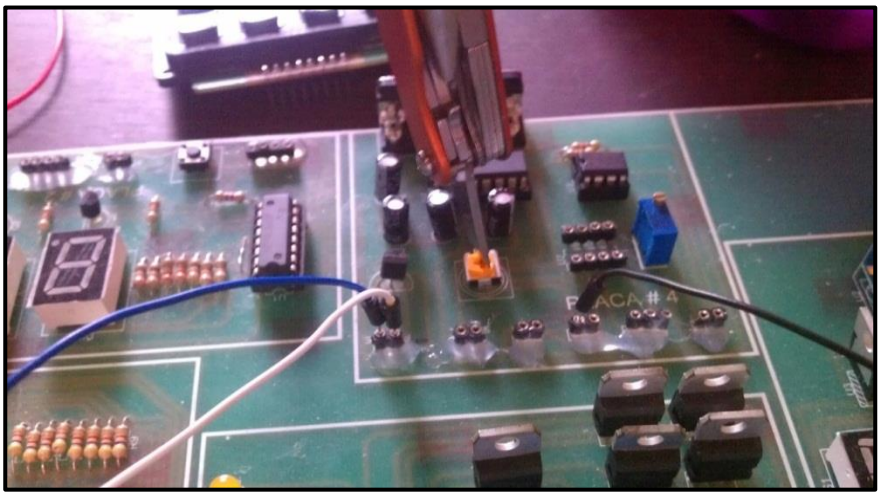
Figura 69. Conversión análoga-digital con valores entre 0 y 4095



Elaborado por: Francisco Reyes y Nina Chicaiza

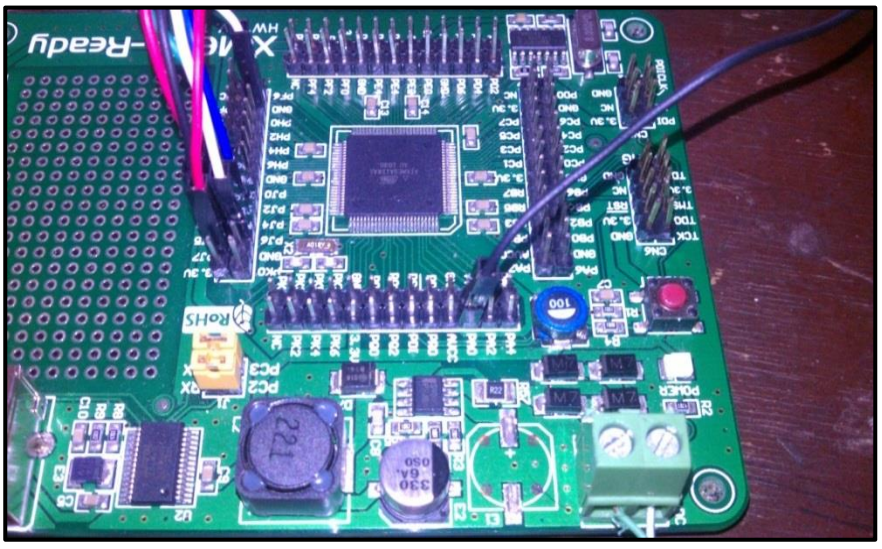
El potenciómetro en la Placa #4 modifica los valores de voltaje que ingresan al pin 0 del puerto A, la conversión es realizada y se muestra en la pantalla los resultados; el rango en el que varía el potenciómetro es de 0-5 V, el microcontrolador posee un conversor de 12 bits, con lo que se consigue un paso de conversión de 0.0012

Figura 70. Potenciómetro de precisión para modificar valores de voltaje



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 71. Conexión en el microcontrolador Avr Xmega



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 72. Conversión A/D con valores entre 0 y 4095

Elaborado por: Francisco Reyes y Nina Chicaiza

2) Conversión A/D con el microcontrolador XMega de Atmel.

Se adquiere la señal de un potenciómetro entre 0 y 5 voltios, y se observa en el LCD un valor entre 0 y 5 voltios. El puerto J es utilizado para controlar el LCD, el pin 0 en el puerto A recibe el voltaje a convertir, se muestra el resultado de procesar el voltaje adquirido entre un rango del 0 al 5.

Figura 73. Conversión A/D con valores entre 0 y 5v

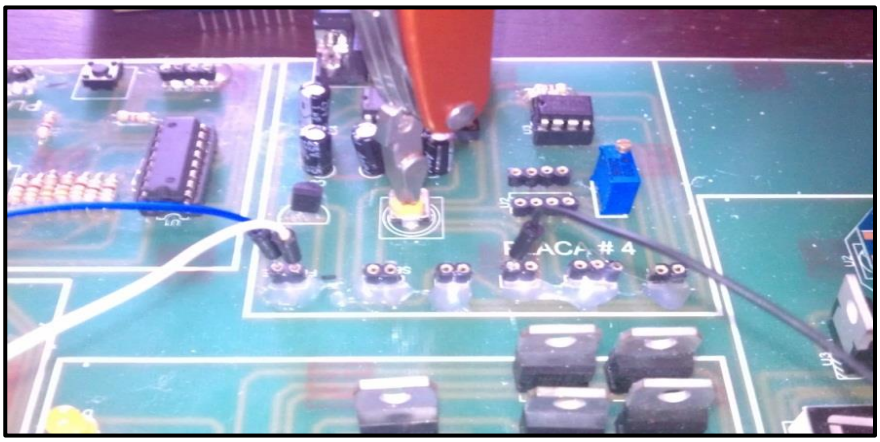
Elaborado por: Francisco Reyes y Nina Chicaiza

El rango de voltaje depende del potenciómetro, usando un algoritmo se ajustan los valores a mostrar en la pantalla; se tiene ahora un rango de 0-5 V para imprimir en el LCD.

Figura 74. ADC con valor máximo de conversión

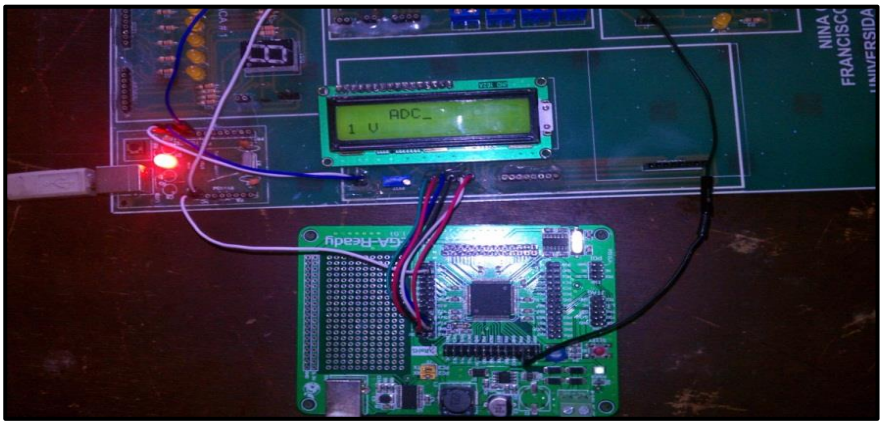
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 75. Potenciómetro de precisión para cambio de valores de voltaje



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 76. Conexión en el microcontrolador Avr XMega



Elaborado por: Francisco Reyes y Nina Chicaiza

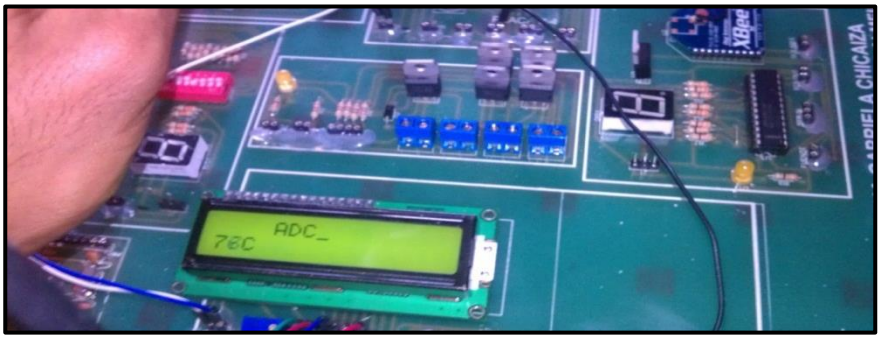
3) Medición de temperatura con Lm35

Las líneas del LCD están conectadas al puerto J, el pin 0 en el puerto A es el encargado de realizar la adquisición de voltaje proveniente del sensor de temperatura.

Figura 77. ADC con el sensor lm35

Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 78. Adquisición de valores de temperatura del sensor



Elaborado por: Francisco Reyes y Nina Chicaiza

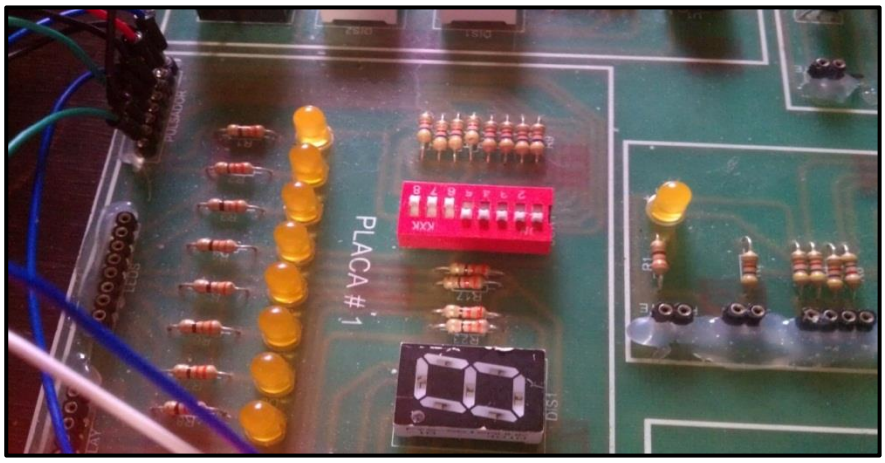
El valor que se muestra en la pantalla varía dependiendo de la fuente de calor que se acerque al sensor, en los archivos adjuntos se detalla el algoritmo utilizado para mostrar el dato adquirido en forma de un valor real de temperatura

4) Conversión Digital Análoga de 12 bits, utilizando las Placas #1 y #3 del módulo.

Se utilizó el Dipswitch para simular las entradas digitales, el puerto E recibe estos datos; el LCD es controlado por el puerto J. La configuración se realiza con:

Config Daca, habilitando el DAC del puerto A, pin 2.

Figura 79. Conversor digital-análogo

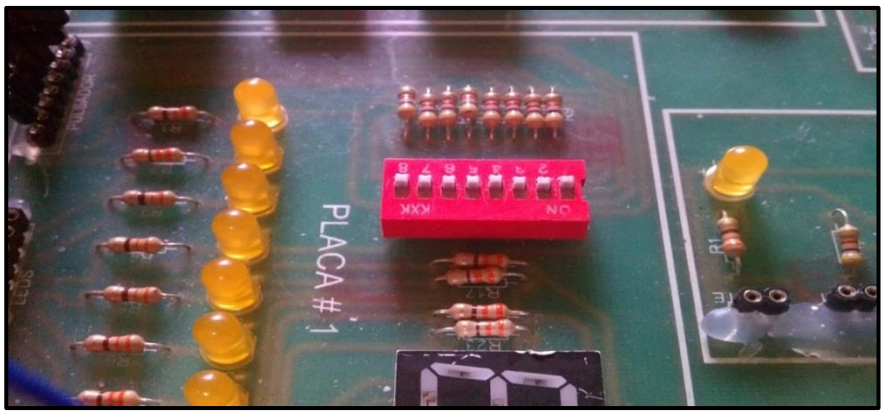


Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 80. Conversor digital-análogo

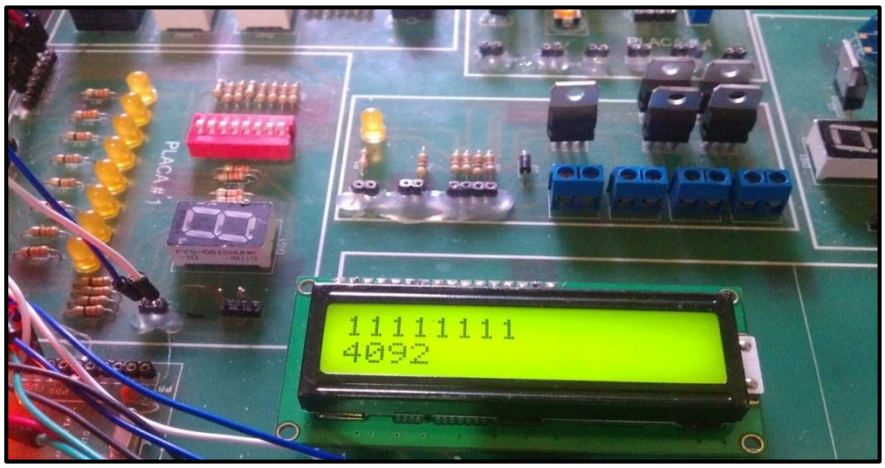
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 81. Entradas digitales en 1



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 82. DAC con todas las entradas digitales en 1



Elaborado por: Francisco Reyes y Nina Chicaiza

Los niveles de voltaje se miden en el pin 2 del puerto A y se comprueba que la transformación se realizó de manera correcta.

4.2.7. Interrupciones

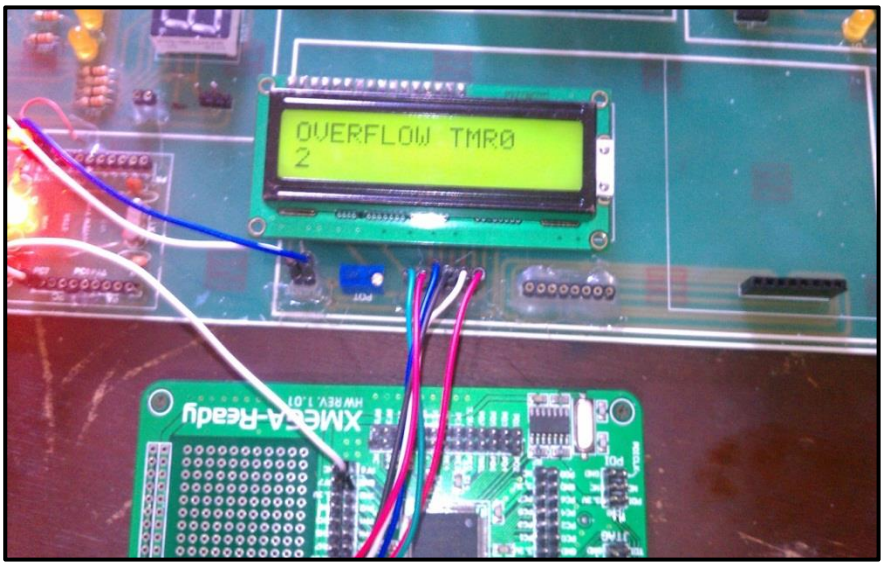
Uso del timer0 para generar interrupciones, para ésta práctica se tienen que utilizar las Placas #1 y #3 del Módulo de Entrenamiento

1) Interrupción, uso del timer0

Para esta práctica solo es necesario el puerto J para controlar la pantalla, la interrupción se realiza por software utilizando el timer 0. La línea de configuración a continuación permite manejar el tiempo en el que se generará una interrupción.

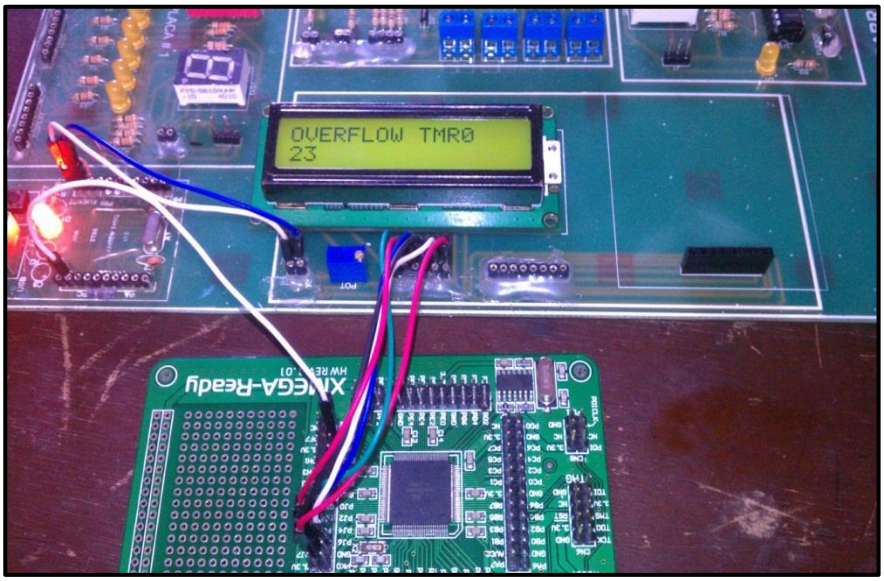
Tcc0_per = 31250 '2MHz/64 = 31250 → One Second Tick

Figura 83. Uso del timer 0



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 84. Uso del timer 0



Elaborado por: Francisco Reyes y Nina Chicaiza

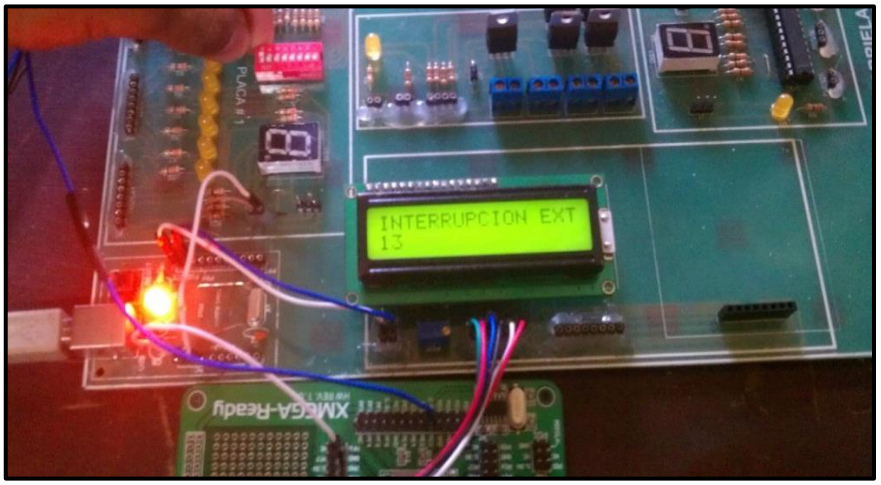
2) Interrupción externa.

Se utilizó el puerto J para controlar la pantalla, y el pin 0 del puerto E para leer la interrupción. Se detalla a continuación el funcionamiento con un pulsador y un dipswitch:

Figura 85. Interrupción externa

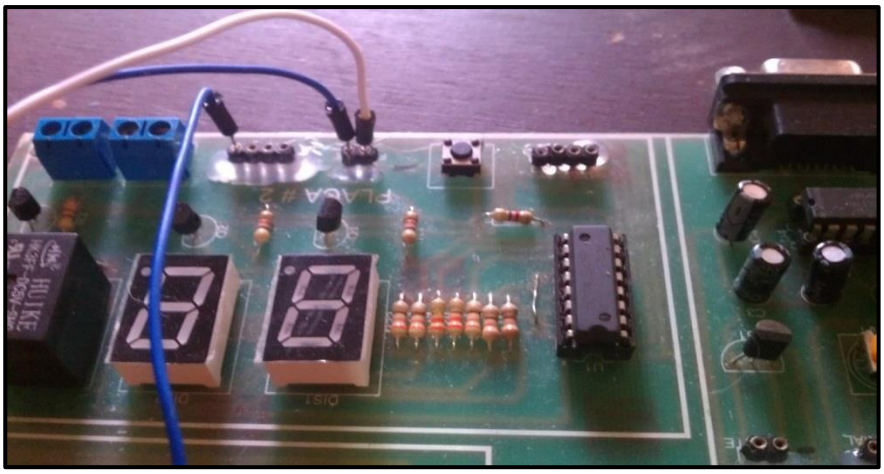
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 86. Se genera la interrupción utilizando un pin del dipswitch



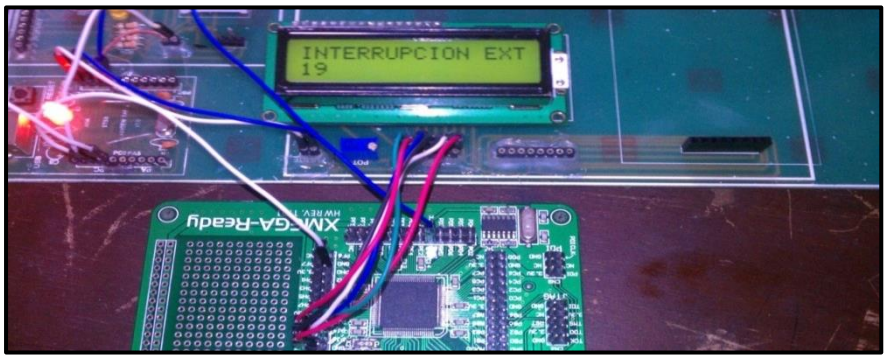
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 87. Interrupción leída por el pin 0 del puerto E



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 88. Se puede generar la interrupción con el botón del módulo n°2



Elaborado por: Francisco Reyes y Nina Chicaiza

Sin importar que placa se utilice para generar la interrupción (placa #1 o placa #2), éstas deben estar correctamente polarizadas.

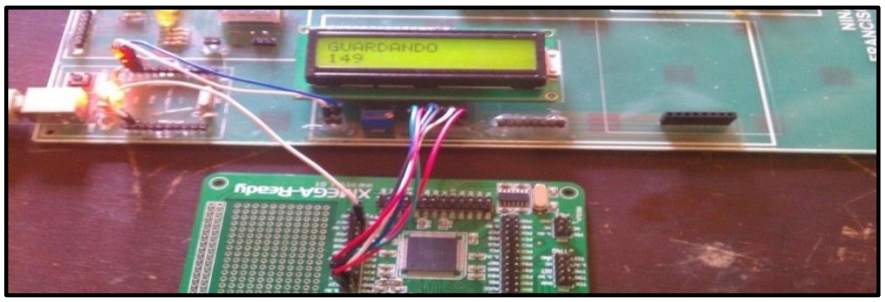
4.2.8. Memorias

Manejar memorias internas y externas con el microcontrolador XMega de Atmel, para ésta práctica se tienen que utilizar las Placas #1 y #3 del Módulo de Entrenamiento

1) Escribir y mostrar una secuencia de números en los sucesivos lugares de EEPROM.

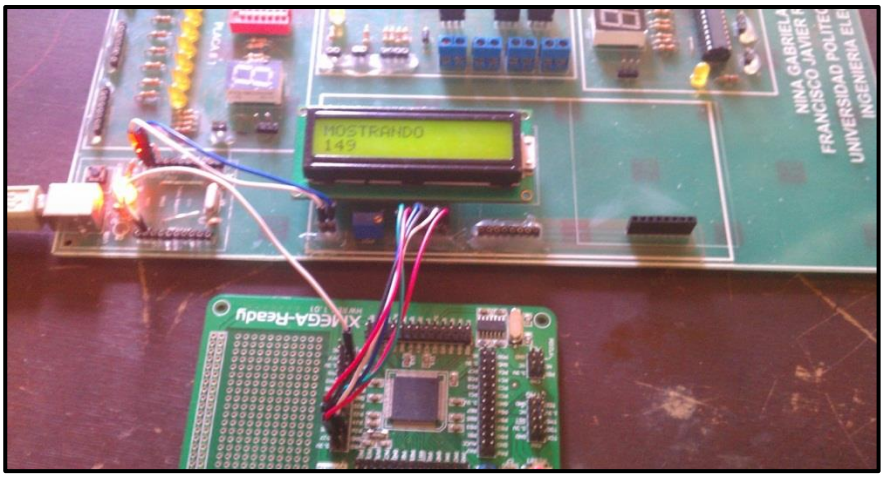
Para esta práctica se utilizó solo el puerto J para mostrar los datos que se guardan y los que se leen en pantalla.

Figura 89. Escribir una secuencia de números en la memoria EEPROM



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 90. Mostrar una secuencia de números en la memoria EEPROM



Elaborado por: Francisco Reyes y Nina Chicaiza

1) Escritura y lectura en una memoria I2C

Se utilizó el puerto J para controlar la pantalla y mostrar los datos que se guarden y los datos que se lean, para controlar la memoria I2C se utilizan 2 líneas de control SDA y SCL, el pin 0 en el puerto C es un pin dedicado que controla la línea SDA mientras el pin 1 en el puerto C controla la línea SCL.

Se guarda una lista de datos que va del 0 al 254, se lee los mismos datos, salvo que al momento de presentar los datos adquiridos, estos son convertidos a sus equivalencias en código ASCII

Figura 91. Escritura y lectura en una memoria I2C

Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 92. Escritura y lectura en una memoria I2C

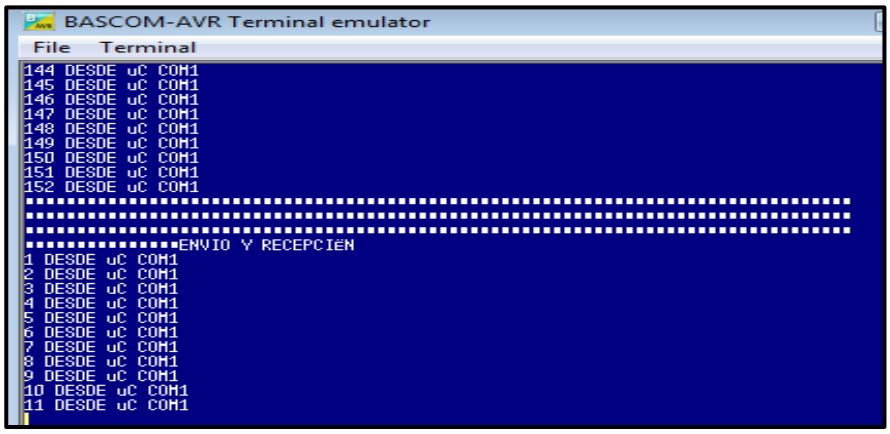
Elaborado por: Francisco Reyes y Nina Chicaiza

4.2.9. Comunicaciones seriales asíncronas rs-232

Utilizar las comunicaciones seriales RS-232, para ésta práctica se tienen que utilizar las Placas #1 y #3 del Módulo de Entrenamiento

1) Transmisión y recepción serial con el microcontrolador XMega de Atmel.
Para observar los datos del envío y recepción se realizara lo siguiente:

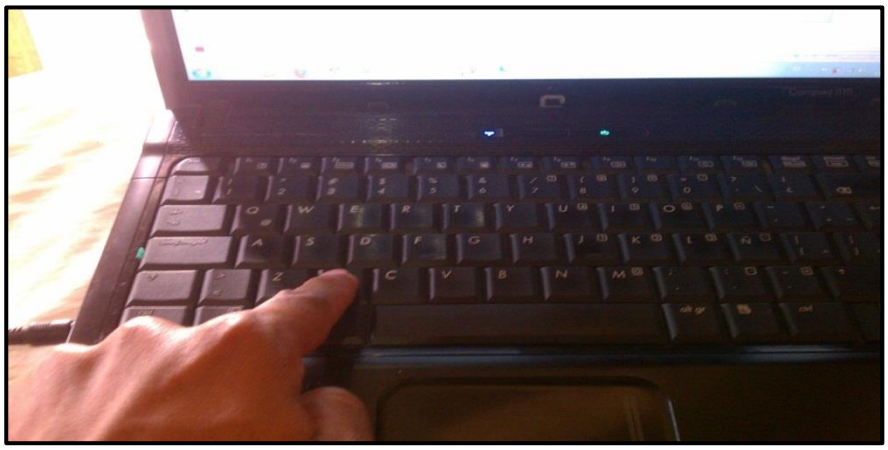
Para abrir el emulador se da un click en Tools > Terminal Emulador
Ya en el terminal se configura el COM que se vaya a utilizar
A continuación, se detalla el envío de datos desde el microcontrolador hacia la pc
Figura 93. Valor de 00111100 escrito en el puerto H y leído en el puerto E



Elaborado por: Francisco Reyes y Nina Chicaiza

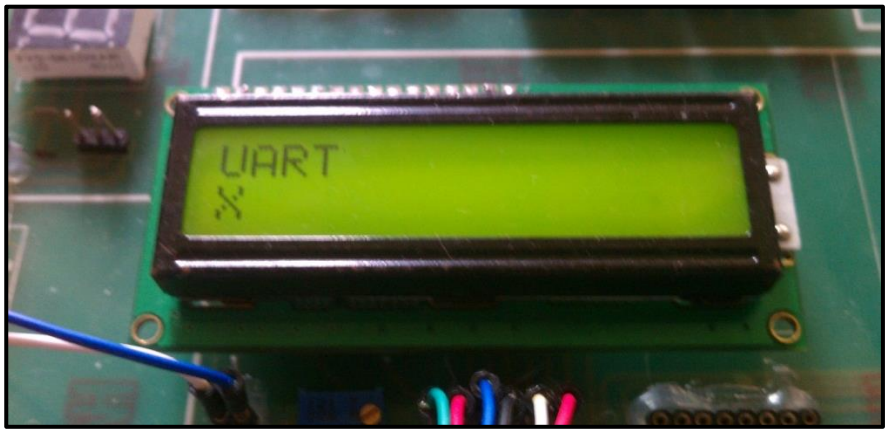
La transmisión de datos desde la pc hacia el microcontrolador se puede observar en la pantalla, se presiona cualquier tecla en la pc e inmediatamente la tecla presionada es visible en el LCD.

Figura 94. Se presiona la tecla x en el computador.



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 95. El Lcd muestra el carácter presionado: “x”



Elaborado por: Francisco Reyes y Nina Chicaiza

Se ocupó el puerto J para mostrar los mensajes en la pantalla, y el puerto C para la conexión de las líneas de control del UART.

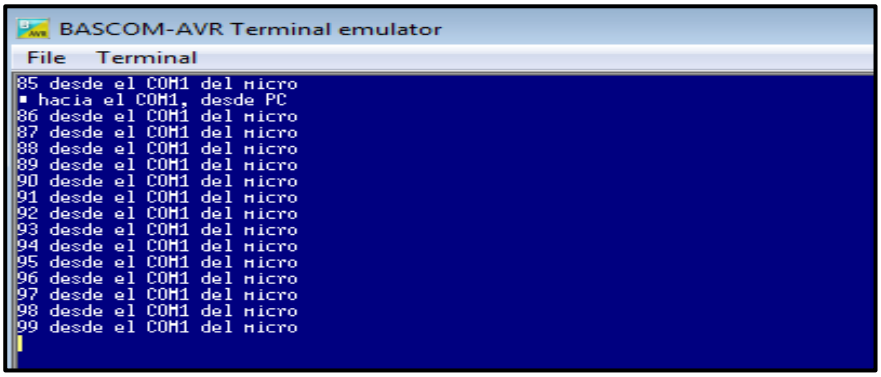
4.2.10. Comunicaciones seriales con usb

Realizar comunicaciones USB del microcontrolador XMega con USB del PC.

1) El siguiente ejemplo permite enviar datos desde el microcontrolador XMega al PC

Se utiliza el puerto USB del microcontrolador, por defecto es el COM1 el que se utilizará al momento de conectar el cable USB. El puerto J maneja la pantalla, para enviar o recibir datos desde el pc o hacia la pc, se debe habilitar el emulador de terminal de BASCOM.

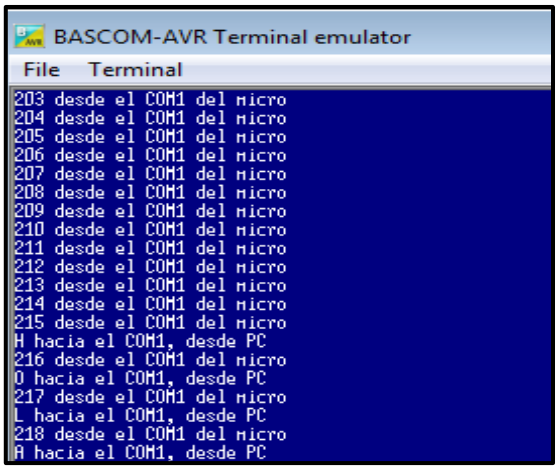
Figura 96. Emulador de terminal de Bascom utilizando el com 1



Elaborado por: Francisco Reyes y Nina Chicaiza

Para enviar datos desde la pc, solo utilizamos el teclado, el simulador de terminal muestra que caracteres hemos enviado:

Figura 97. Envío de datos desde el pc

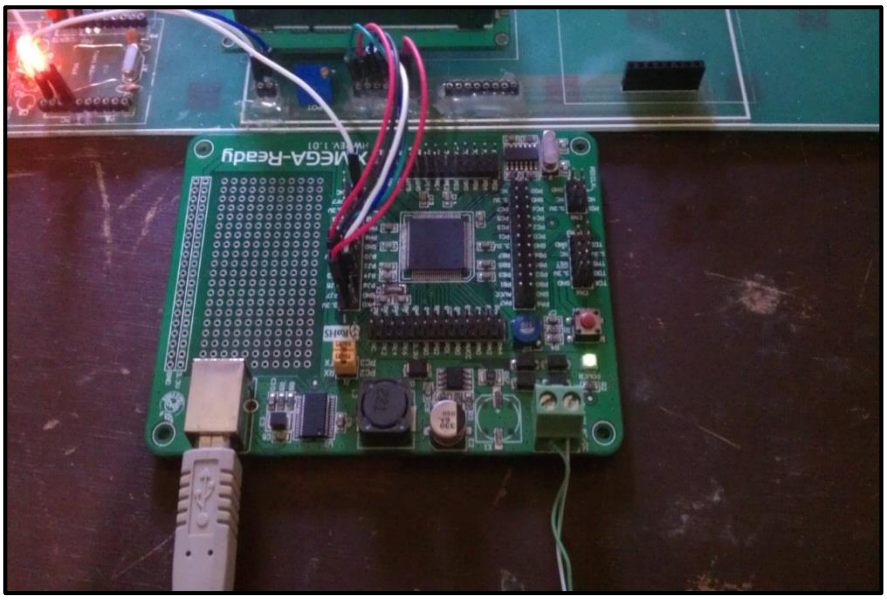


Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 98. Caracteres escritos en la pc y mostrados en el lcd.

Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 99. Conexión realizada en el Avr XMega

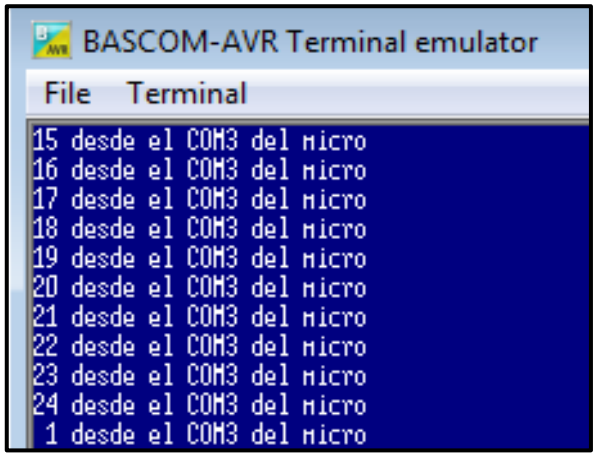


Elaborado por: Francisco Reyes y Nina Chicaiza

No se tiene ninguna otra línea conectada para el control del USB dado que el cable por defecto interconecta al microcontrolador con el puerto C, COM1. Si quisiéramos ocupar otra COM distinto al COM1 solo debemos interconectar los pines TX, RX en la placa del microcontrolador y los pines relacionados con esas líneas de control.

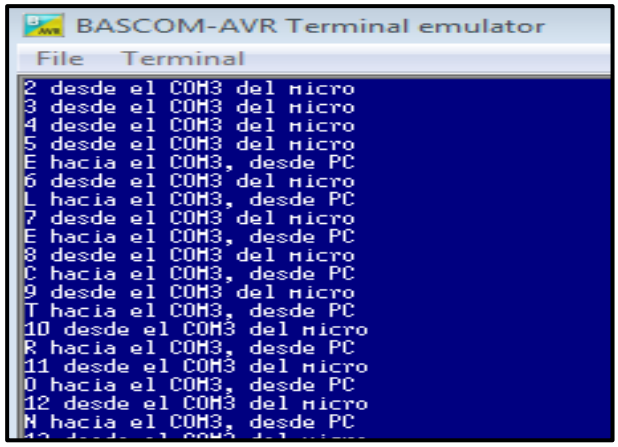
Ocupando el COM3:

Figura 100. Emulador de terminal de Bascom, utilizando el com 3



Elaborado por: Francisco Reyes y Nina Chicaiza

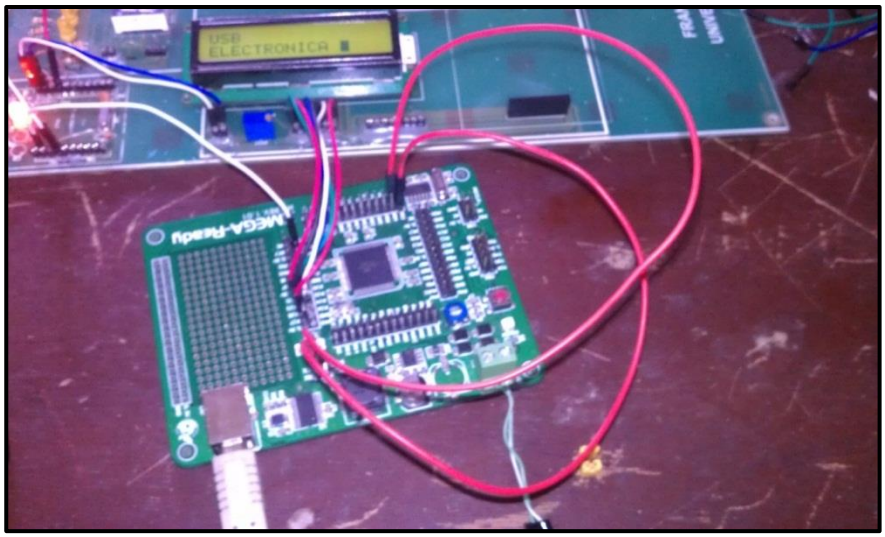
Figura 101. Envío de datos desde la pc.



Elaborado por: Francisco Reyes y Nina Chicaiza

Como fue el COM3 el que se utilizó, son los pines D2, D3 los que controlan las líneas TX, RX en el microcontrolador.

Figura 102. Conexión para activar el com 3



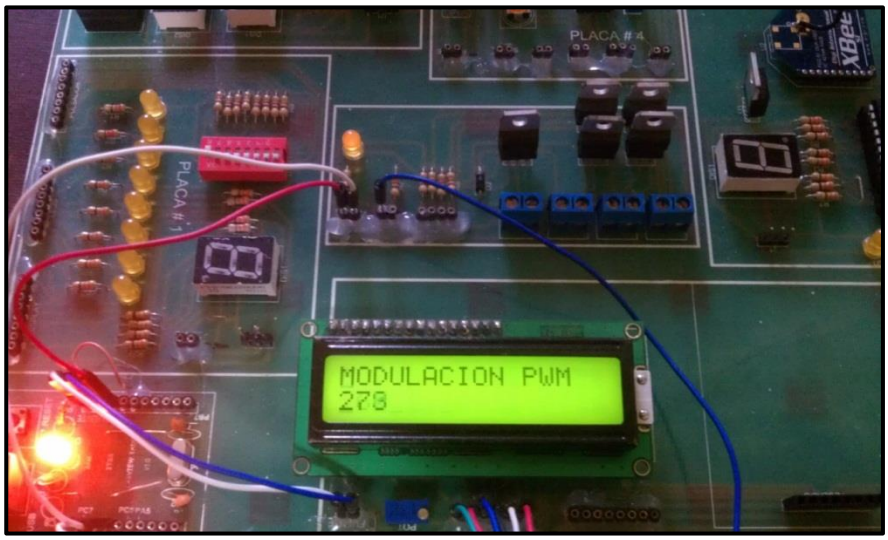
Elaborado por: Francisco Reyes y Nina Chicaiza

4.2.11. Comunicaciones seriales síncronas spi

Utilizar las comunicaciones seriales sincrónicas SPI, entre el microcontrolador XMega y periféricos; para ésta práctica se tiene que utilizar la Placa #6 Módulo de Entrenamiento

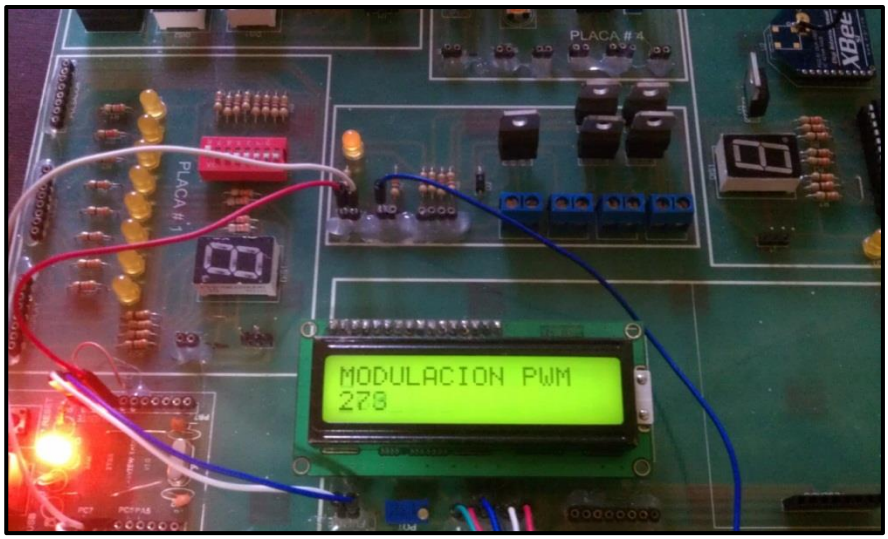
1) El siguiente programa es utilizado para controlar el max7219.

Figura 103. Conexión para activar el com 3



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 104. Conexión para activar el com 3



Elaborado por: Francisco Reyes y Nina Chicaiza

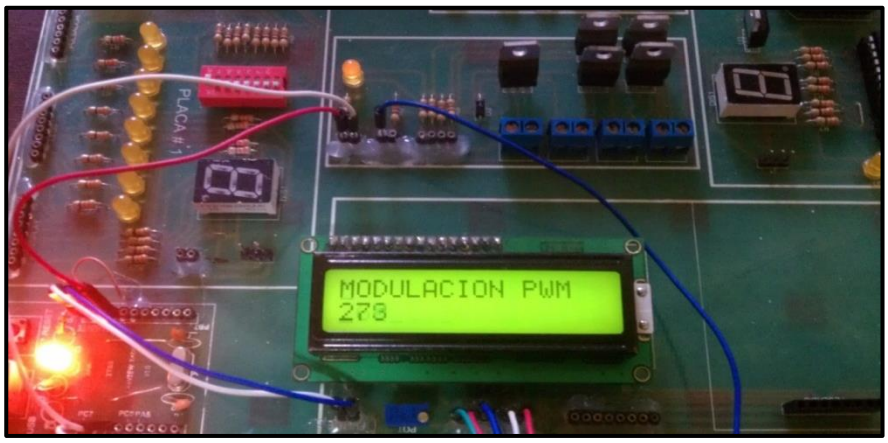
4.2.12. Control de motores

Para la práctica de Control de Motores se tienen que utilizar las Placas #3 y #5 del Módulo de Entrenamiento

1) Encendido de un led mediante el uso de PWM utilizando el microcontrolador XMega.

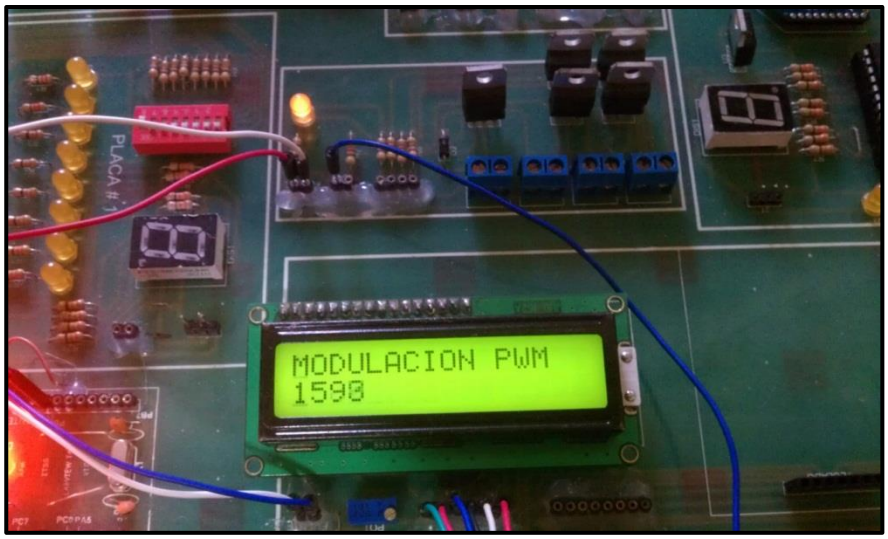
Se controló las líneas de la pantalla utilizando el puerto J; la línea de control de PWM se conecta mediante el pin D0 hacia el microcontrolador.

Figura 105. Encendido de un led usando pwm



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 106. Encendido de un led usando pwm



Elaborado por: Francisco Reyes y Nina Chicaiza

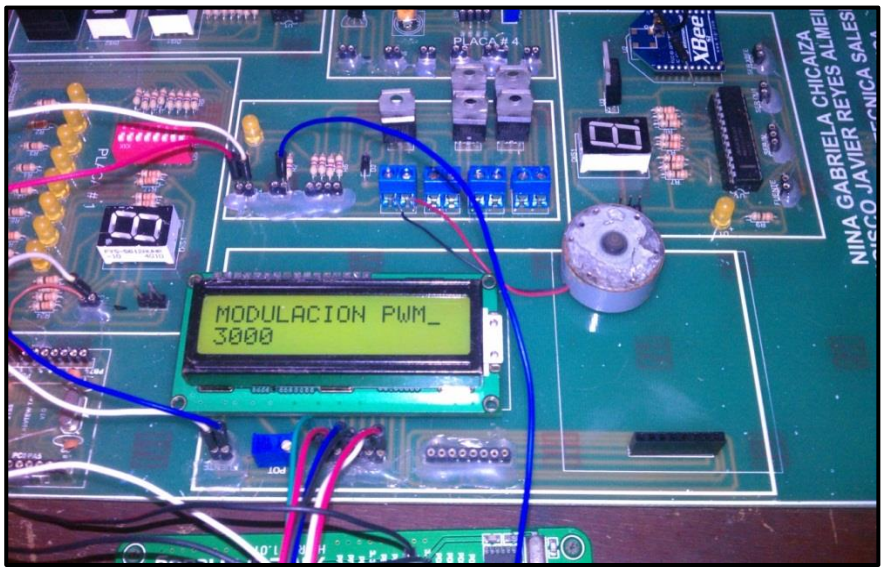
2) Mediante 2 pulsadores controle la velocidad de un motor DC, los pulsadores incrementarán la velocidad en pasos de 10%.

Se utilizan dos puertos del dipswitch para controlar el registro que variará el valor del ancho de pulso, los pines E.0 y E.1

El ancho de pulso es modificado y se envía a través del pin D0.

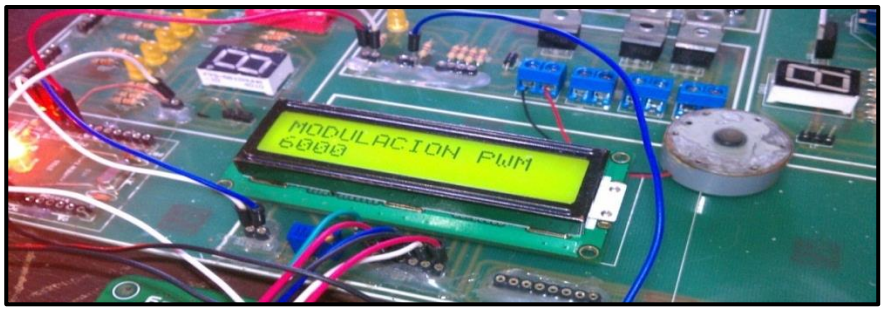
El puerto para controlar la pantalla fue el puerto J

Figura 107. Controlar la velocidad de un motor DC con dos pulsadores



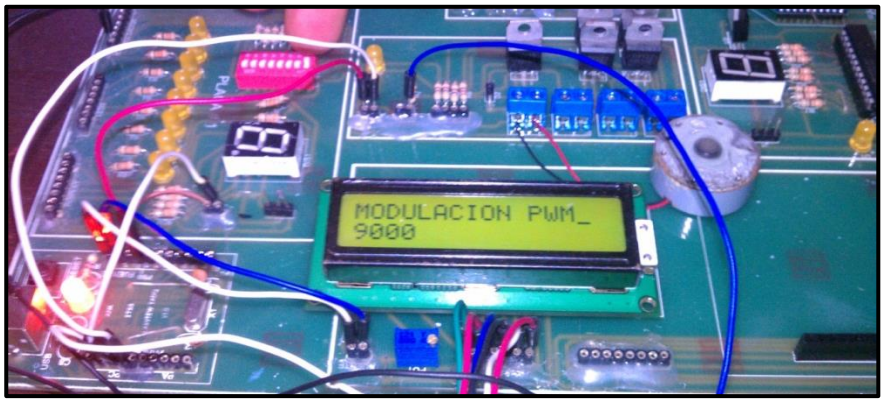
Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 108. Controlar la velocidad de un motor DC con dos pulsadores



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 109. Controlar la velocidad de un motor DC con dos pulsadores



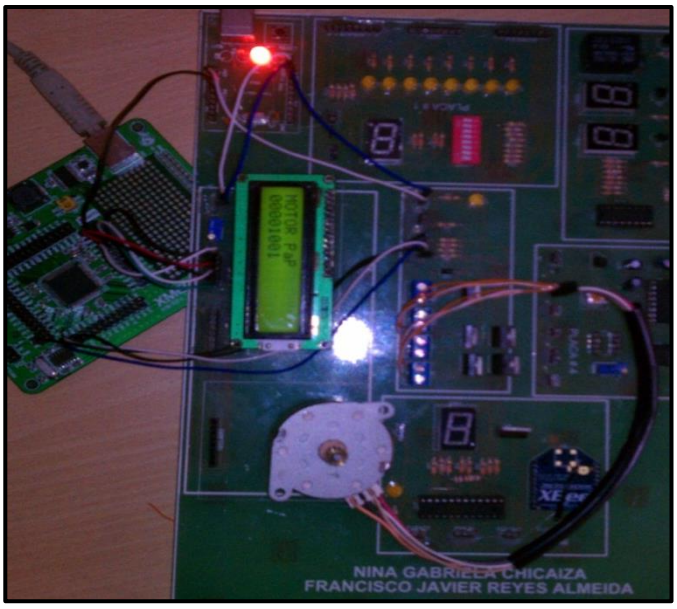
Elaborado por: Francisco Reyes y Nina Chicaiza

3) Control de un motor paso a paso unipolar en secuencia normal, utilizando el microcontrolador XMega.

Figura 110. Controlar la velocidad de un motor paso a paso

Elaborado por: Francisco Reyes y Nina Chicaiza

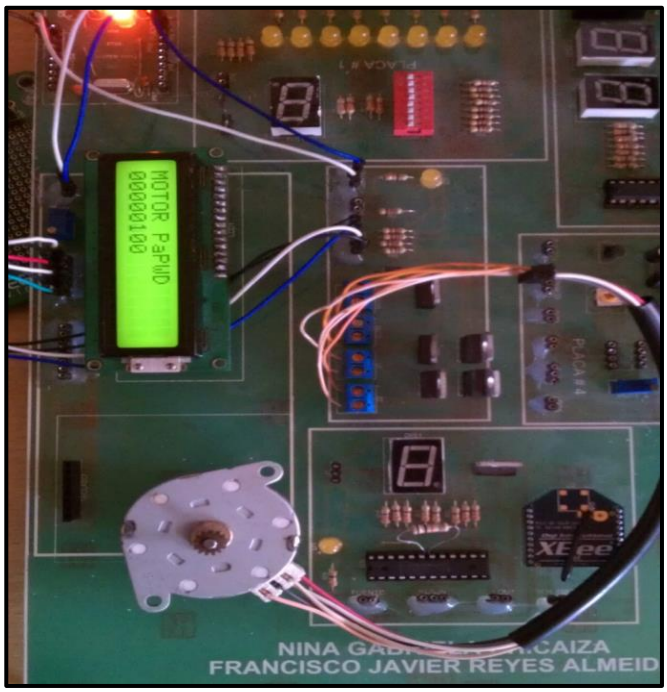
Figura 111. Controlar la velocidad de un motor paso a paso en secuencia normal.



Elaborado por: Francisco Reyes y Nina Chicaiza

4) Control de un motor paso a paso unipolar en secuencia wave drive, utilizando el microcontrolador XMega.

Figura 112. Controlar la velocidad de un motor paso a paso en secuencia wave drive



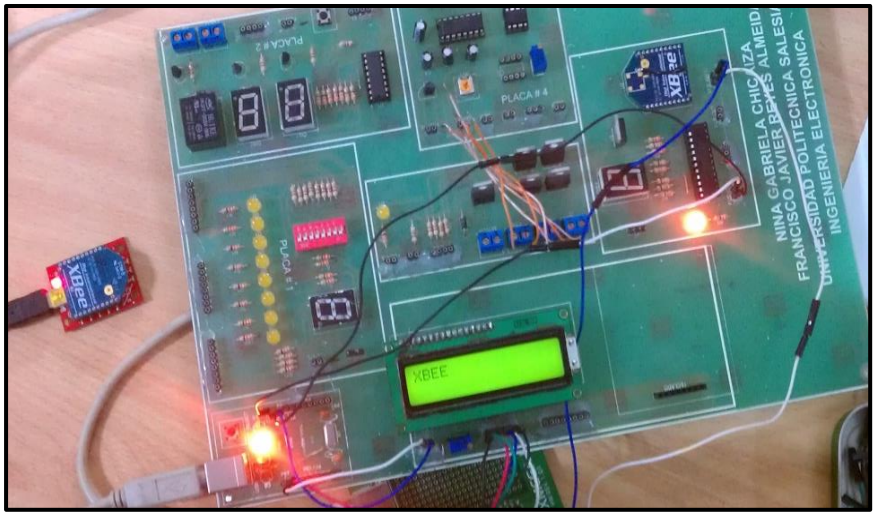
Elaborado por: Francisco Reyes y Nina Chicaiza

4.2.13. Transmisión radiofrecuencia

Utilizar los puertos del microcontrolador XMega de Atmel para tener comunicación RF, para ésta práctica se tiene que utilizar la Placa #6 Módulo de Entrenamiento

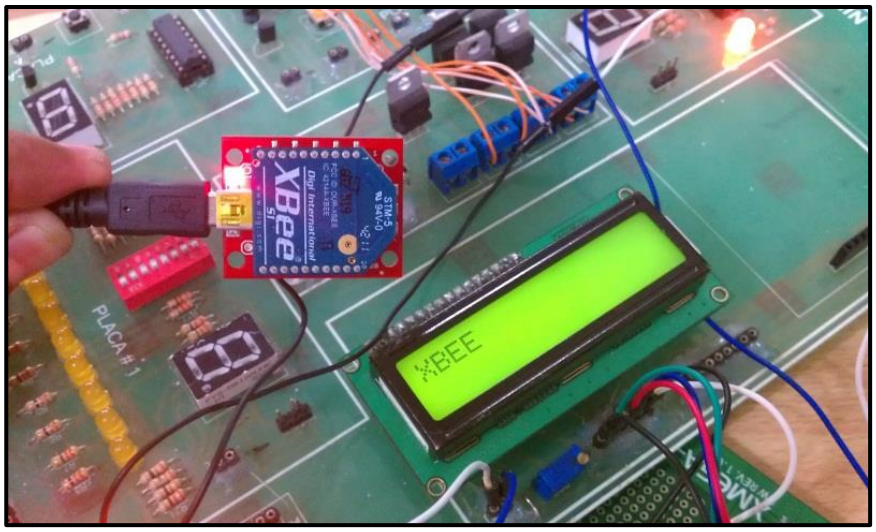
1) Transmisión y recepción de datos con el módulo de radiofrecuencia XBEE.

Figura 113. Envió y recepción vía XBee



Elaborado por: Francisco Reyes y Nina Chicaiza

Figura 114. Envió y recepción vía XBee



Elaborado por: Francisco Reyes y Nina Chicaiza

4.2.13. Presentación del módulo funcional

Con el módulo y el folleto de prácticas se realiza una presentación para el personal que labora en el laboratorio de electrónica en la Universidad Politécnica Salesiana periodo Febrero – Julio 2013

La presentación se la realizo en el laboratorio de electrónica con fecha Viernes 8 de Marzo del 2013, la lista de participantes se adjunta a continuación

Tabla 6. Datos de los asistentes en la presentación del módulo.

Apellidos y Nombre	Institución	Cédula	Celular	Casa
Negrete Ayala Milton Stalin	Colegio Técnico Sucre	1724169733	0979020583	3066409
Mena Jara David Josué	Colegio Técnico Sucre	1751291996	0999261103	2693719
Víctor Daniel Proaño Gavilanes	Universidad Politécnica Salesiana	1718686692	0992521849	2695868
Erick Alexander Peñaherrera Aguilar	Universidad Politécnica Salesiana	1717986861	0984297864	2962399

Gustavo Javier Caiza Guanochanga	Universidad Politécnica Salesiana	1721192919	0998595749	2962399
William Paul Oñate	Universidad Politécnica Salesiana	1715580500		2654860
José Luis Bucheli Naranjo	Universidad Politécnica Salesiana	1718399578	0984358528	2695164

Elaborado por: Francisco Reyes y Nina Chicaiza

Los **Tema**s que se trataron en la presentación son:

1. Manejo de puertos, entradas y salidas
2. Diseño de un contador haciendo uso de subrutinas e interrupciones
3. Manejo de periféricos de entrada y salida de datos
4. Diseño de un convertidor analógico digital
5. Manejo de memorias mediante la interfaz I2C
6. Manejo de la interfaz USB
7. Manejo de motores DC y PaP
8. Diseño de una aplicación con interfaz ZigBee

Concluida la presentación de las prácticas, se plantean 10 preguntas para calificar el grado de satisfacción de los usuarios.

Se tabulan los datos y se resuelve:

El porcentaje de satisfacción tiende a más del 80% en la mayoría de preguntas planteadas, salvo dos en donde la aceptación se encuentra entre el 60% y 80%. Con esto se deja claro que manipular el módulo AVR XMega y sus componentes es “Satisfactorio”.

En la pregunta 1, se tienen el 86% de aceptación, queda constancia de que la mayoría considera que le resulta muy satisfactorio el manejo del módulo y sus componentes.

En la pregunta 2, la aceptación fue del 93%, siendo el módulo una herramienta muy satisfactoria para aprender el manejo de microcontroladores.

En la pregunta 3, se tiene una aceptación del 93% considerando el material que se entrega al momento de realizar las prácticas como muy satisfactorio.

En la pregunta 4, la tendencia cambia y presenta una aceptación del 66% considerando que es satisfactoria la robustez de los componentes, y su maniobrabilidad se tiene que realizar con el cuidado respectivo en este tipo de prácticas. Sin embargo la mayoría de los componentes presentes en el módulo son de tecnología TTL resistente a la estática presente en el cuerpo humano.
En la pregunta 5, se tiene una aceptación del 93%, se consideró que la polarización para todas las prácticas fue muy satisfactoria.

En la pregunta 6, la aceptación es del 96%, los encuestados consideraron que es muy satisfactorio el orden de las prácticas, pues cada práctica lo prepara para el desenvolvimiento de las siguientes.

En la pregunta 7, la aceptación es del 86%, considerando muy satisfactoria la información que se entrega sobre la interconexión entre periféricos, sin embargo los encuestados consideran que se debería agregar dicha información en el módulo de prácticas.

En la pregunta 8, la aceptación es del 73%, los encuestados consideraron que los componentes que lleguen a fallar no se podrían cambiar fácilmente. Recomendaron como solución el uso de sócalos soldados a la placa.

En la pregunta 9, la aceptación para las conexiones a 110v es del 86%, se considera muy satisfactoria.

Para la pregunta 10, que son lo que se recomiendan agregar se tiene:

Información detallada de los periféricos y puertos

Practicas con GLCD y el TOUCH resistivo

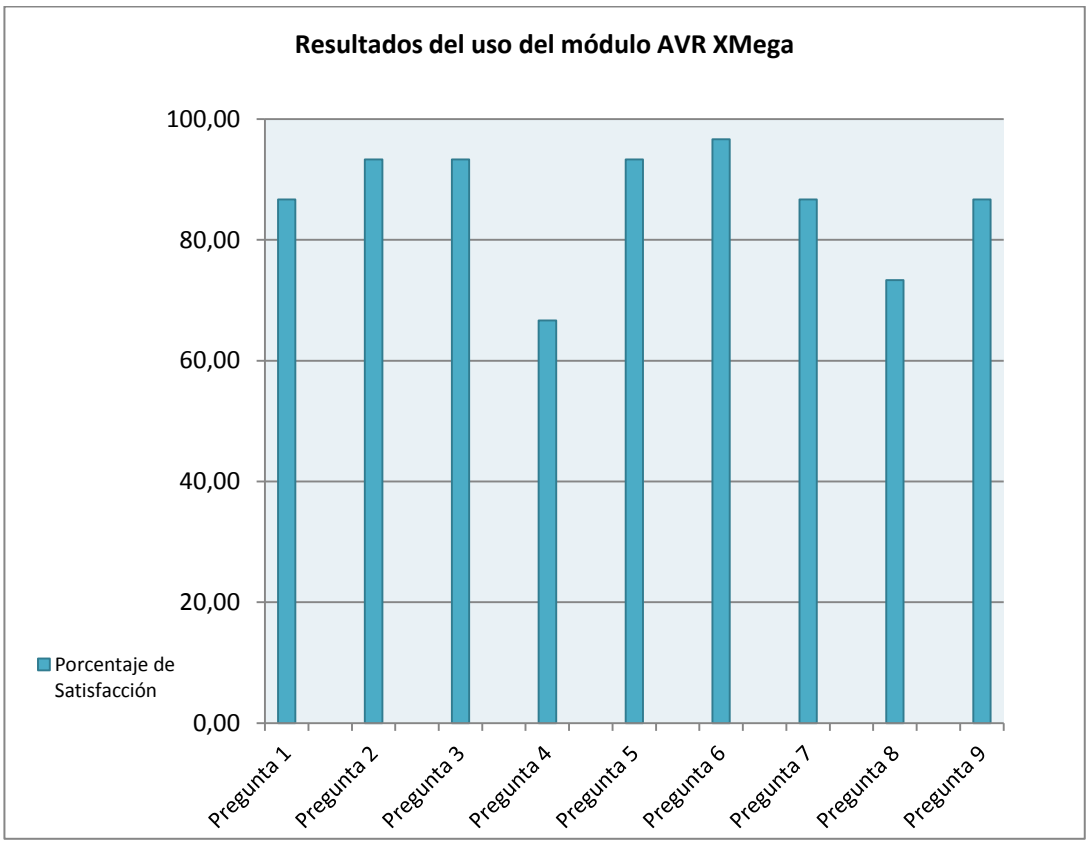
Practicas implementando un módulo GSM

La implementación en una práctica de una tarjeta SD

Soldar sócalos para los componentes.

A continuación, la gráfica detalla los resultados de la encuesta:

Figura 115. Presentación de datos encuestados



Elaborado por: Francisco Reyes y Nina Chicaiza

Las preguntas que se plantearon son:

1. ¿Le resulto sencillo el manejo del módulo y sus componentes?
2. ¿Cree usted que este módulo le sirve para aprender manejo de microcontroladores?
3. ¿El material que se entrega es el apropiado para realizar las practicas?
4. ¿Los componentes en el módulo son robustos, es decir, se los puede maniobrar sin mucho cuidado?
5. ¿La polarización en cada práctica se la realizo sin inconvenientes?
6. ¿Las prácticas que se proponen al comienzo del módulo lo preparan para las que continúan?
7. ¿La interconexión de los periféricos se encuentra correctamente detallada?
8. ¿Se puede cambiar componentes del módulo que lleguen a fallar?
9. ¿La conexión a 110v AC se realiza de manera adecuada?
10. ¿Qué sugiere añadir al módulo?

Conclusiones

- A la finalización del desarrollo, y tras comprobar los resultados se puede concluir que se han cumplido con los objetivos propuestos. El resultado obtenido ha sido una tarjeta de prácticas. Las tarjetas que se utilizan para la materia de Sistemas Microprocesados en la Universidad Politécnica Salesiana, son específicas para una sola marca de microcontroladores, inhibiendo el desarrollo completo de los estudiantes que la cursan. El dispositivo resultado de este proyecto de titulación, no restringe su uso a una determinada marca de microcontroladores, su uso es universal. Tal característica es la principal ventaja, frente a cualquier otra tarjeta comercial, el único requisito para su utilización es el poseer un microcontrolador e interconectar sus entradas y salidas con los pines de los periféricos en la tarjeta, independientemente de marcas de microcontrolador, la tarjeta es capaz de trabajar con marcas como Microchip o Atmel.

- Uno de los principales inconvenientes al momento de manejar la tarjeta fue el hecho de que está diseñada con partes electrónicas que trabajan a más de 3.3v, que es voltaje máximo que entrega el microcontrolador actual, se resolvió al implementar una fuente de alimentación de 5v, los que se consiguen desde un puerto USB. Las prácticas que se realizan en los laboratorios de la materia de Sistemas Microprocesados uno incorporan el manejo de convertidores digitales y análogos, para suplir esta necesidad fue necesario diseñar un espacio para el manejo de señales análogas, el cual está compuesto por sensores y potenciómetros, con los cuales se consiguen señales análogas de hasta 5v, este voltaje no representa ningún problema para las entradas del microcontrolador. El intermediario para la conexión entre la tarjeta y la PC es el micro, tal ventaja elimina la necesidad de cualquier driver. La gran mayoría de prácticas no poseen una interfaz propia para mostrar peticiones o resultados, por lo tanto se incorpora en la tarjeta una pantalla de cristal líquido, la misma que previamente se enseña a conectar y configurar, se tuvo la presencia de ruido en la pantalla al momento de imprimir resultados del uso con el módulo de motores, se mejora el rendimiento al conectar el módulo de polarización en la tarjeta a una fuente con un mejor suministro de corriente que el que da el conector USB.

- El auge de tecnología para transmisión de datos de manera inalámbrica presento la necesidad de un módulo para el desarrollo de prácticas con el estándar de transmisión XBee, que cumplirá de emisor o receptor; para iniciar una transmisión de datos se precisa de otro módulo, el cual se entrega junto con el cable y el terminal que permite la conexión con una pc. Se complementa el diseño electrónico de bajo voltaje con prácticas que permiten la conexión y control de dispositivos que operen a 110v, se utiliza relés como protección debido a la relación costo – eficiencia. La documentación con la que cuenta este proyecto detalla minuciosamente los distintos módulos que conforman la tarjeta, sus puertos y periféricos, el respaldo de prácticas le garantizan al estudiante un uso eficiente de todos los recursos.

- Al término de este proyecto se entrega una herramienta desarrollada completamente para estudiantes, que les permitirá desarrollar proyectos de investigación futuros, a partir de una base teórico-práctica. El presupuesto es superior al que se requiere para conseguir una tarjeta de prácticas de cualquier marca comercial, pero se debe tener en cuenta que tarjetas comerciales tienen una producción en serie y al por mayor, abaratando costos. Se entrega un material de consulta para el estudio de un microprocesador de Atmel, con características internas, detalle de registros, etc. Además de un folleto completo acerca de cómo programar dicho microcontrolador haciendo uso de Bascom.

Cualquier proyecto de investigación futuro puede hacer uso de este documento y sus recursos.

Recomendaciones

- Desarrollar un proyecto utilizando el módulo DMA del microcontrolador.
- El microcontrolador posee pines dedicados para conectar una batería, se recomienda diseñar un proyecto que utilice una fuente alterna de voltaje.
- Se tiene en el dispositivo XMega un contador a 32 bits, se recomienda realizar un proyecto para un contador con dichas características.
- Diseñar un proyecto para transmisión de datos utilizando los módulos de transferencia infrarrojo.
- Elaborar un proyecto que conecte el microcontrolador con una memoria SDram, utilizando la interfaz de bus externo EBI.
- Plantear un proyecto para utilizar la interfaz ZIGBEE y realizar procesamiento de datos.
- Realizar Procesamiento Digital de Señales con el microcontrolador XMega.
- Es necesario habilitar una librería externa para poder controlar comandos que se envían a dispositivos periféricos como la pantalla de cristal líquido al momento de mostrar mensajes.
- Se tendrá desconectados los jumpers en la placa del microcontrolador cuando se esté realizando una práctica de UART - RS232, al no realizar este cambio en las conexiones, la interfaz que por defecto envía y recibe datos será la interfaz UART – USB, sin que se pueda apreciar el cambio al momento de seleccionar un puerto COM distinto al por defecto.

Lista de referencias

Altium. (2013). *Altium Design*. Recuperado el 24 de Abril de 2013, de http://altium.com/en

Atmel. (2009). *Atmel Corporation*. Recuperado el 15 de Noviembre de 2012, de www.atmel.com/images/doc8077.pdf

Cadsoft. (2011). *Cadsoft Eagle*. Recuperado el 23 de Abril de 2013, de http://www.cadsoftusa.com/

Claus, K. (1998). *Avr risc microcontrolleer handbook.* (Newnes, Ed.) Estados Unidos de Norteamérica: Butterworth - Heinemann.

Jiménez Carlos, L. A. (Julio de 2010). *Pixel Bit.* Recuperado el 15 de Abril de 2013, de http://www.sav.us.es/pixelbit/pixelbit/articulos/n37/2.pdf

Lewin, E. (2006). *So, you wanna be an enbedded engineer.* Reino Unido: Elsevier Inc.

Lpkf. (2013). *Laser & Electronics*. Recuperado el 23 de Abril de 2013, de http://www.lpkf.com/products/rapid-pcb-prototyping/circuit-board-plotter/protomat-s103.htm

Mikroelectronika. (1998). *Mikroe*. Recuperado el 10 de Enero de 2012, de http://www.mikroe.com/

Mikroelektronica. (2008). *Mikrobootloader Avr XMega*. Recuperado el 12 de Abril de 2013, de http://www.mikroe.com/products/view/579/xmega-ready-board/

Mits. (2012). *Mits Electronics*. Recuperado el 26 de Abril de 2013, de http://www.mitspcb.com/edoc/11lab.htm

National Instruments. (2010). *Labview.* Recuperado el 29 de Julio de 2012

New Wave Concepts. (2012). *New Wave Concepts*. Recuperado el 24 de Abril de 2013, de http://www.new-wave-concepts.com/pr/pw_tour1.html

Oregon State. (2010). *Oregon State*. Recuperado el 12 de Noviembre de 2012, de www.oregonstate.edu/mime/spring2010

Proteus. (2011). *Proteus Hubor*. Recuperado el 24 de Abril de 2013, de http://proteus.hubor.es/proteus-pcb/proteus-pcb.html

Ramiro, V. (2008). *Aplicaciones Electrónicas con Microcontroladores.* Quito, Pichincha, Ecuador: Graficolor.

Sable. (2012). *Cnc Sable*. Recuperado el 24 de Abril de 2013, de http://www.cnc-sable.nl/sable-2015-p-21.html?language=en

Wandererwolf. (2010). *Instructables*. Recuperado el 16 de Noviembre de 2012, de http://www.instructables.com/id/Atmel-Xmega-USBSerial-Arbitrary-Waveform-Generato/

XMega Ready. (1998). *XMega Ready*. Recuperado el 12 de Abril de 2013, de http://www.mikroe.com/products/view/579/xmega-ready-board/

ANEXO 1

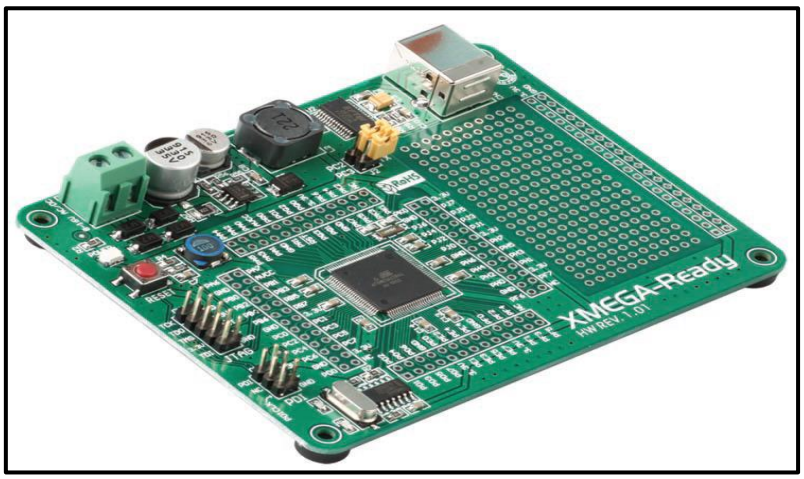
MANUAL TARJETA XMEGA READY

La Tarjeta XMEGA READY es una solución completa para el desarrollo rápido y sencillo de aplicaciones en las que se utiliza el microcontrolador ATMEL ATxmega 128A1, dispositivo conectado a un oscilador de 8 MHz.

Características

- Bootloader cargado en el microcontrolador ATxmega 128A1
- Comunicación USB-UART
- Alimentación de 7-23V AC ó 9-32V DC

Figura 1. Tarjeta xmega ready



Fuente: (XMega Ready, 1998)

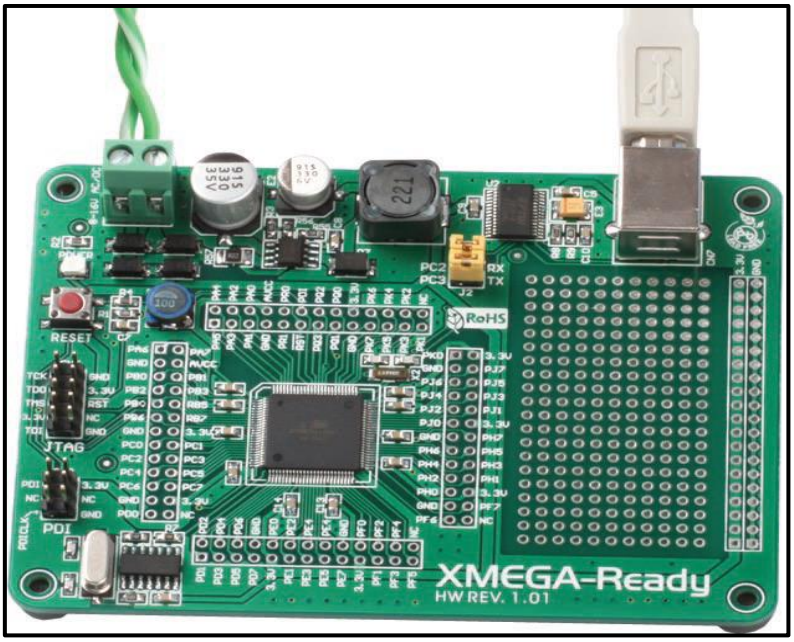
Aplicación

Representa un sistema de desarrollo el cual puede ser utilizado en combinación con otras tarjetas o de forma autónoma. Debido al bootloader precargado y a su MCU de 8 bits es ideal para experimentos de bajo costo y diseño de productos finales.

Fuente de poder

Para conectar una fuente de poder a la tarjeta XMega Ready se utiliza un terminal CN5. El voltaje puede variar en AC desde 7 a 23V y en DC de 9-32V. Cuando se programa el MCU haciendo uso del bootloader es necesario conectar la tarjeta con el PC usando el cable USB y la tarjeta debe estar energizada.

Figura 2. Tarjeta xmega ready conectada a la fuente de poder y a la pc vía usb



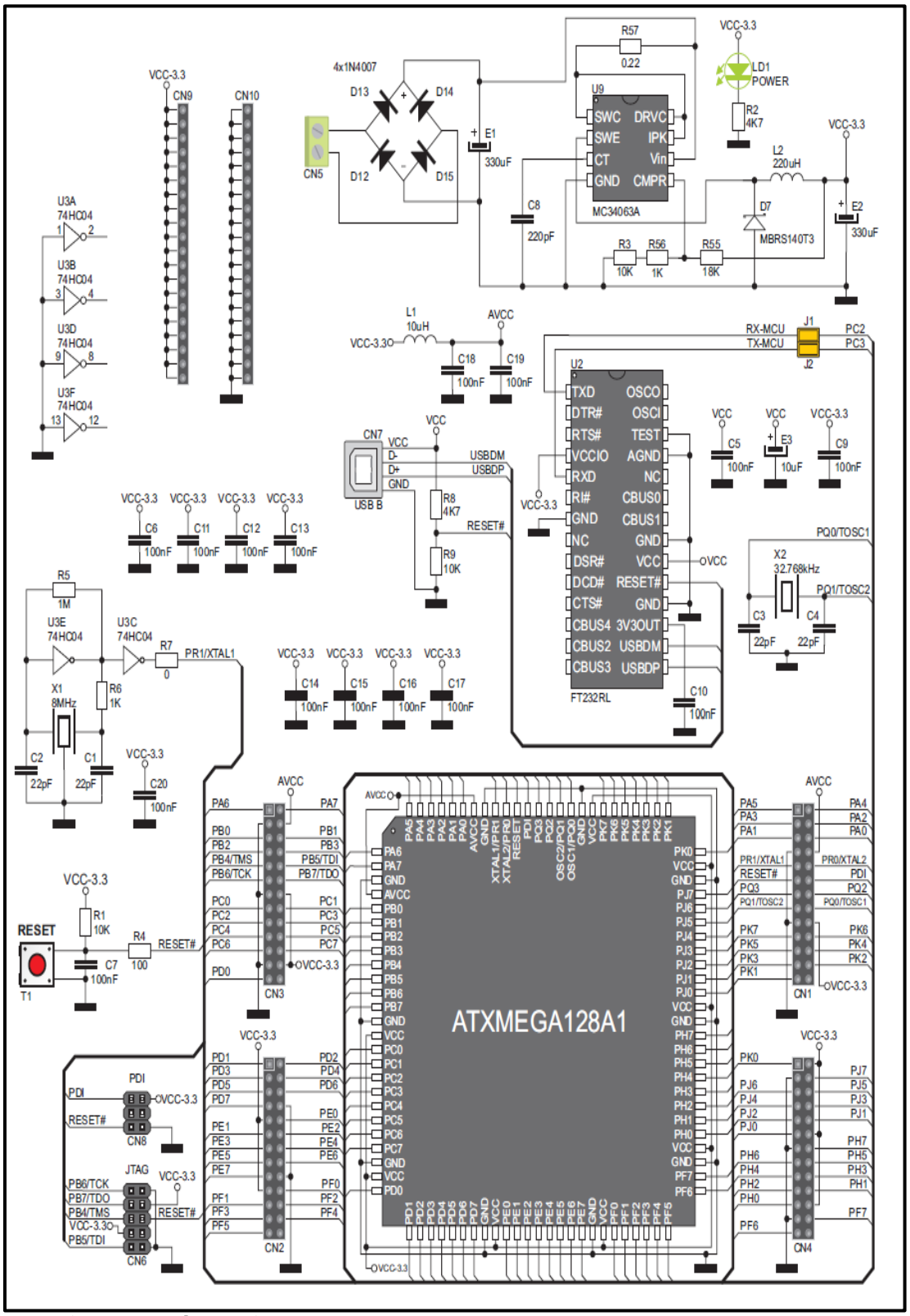
Fuente: (XMega Ready, 1998)

Para un fácil acceso a los pines del microcontrolador la tarjeta incorpora terminales. Cada terminal viene marcado con el nombre de pin al que está conectado. Se tiene un espacio libre donde se pueden colocar elementos para prácticas adicionales.

Para conectar la tarjeta y la PC se necesita un cable USB con terminal para conector tipo CN7, cuando la conexión se establece, la PC se comunica con el chip FTDI el cual está conectado a los pines PC2 Y PC3 en el MCU, estos pines se utilizan para comunicación serial.

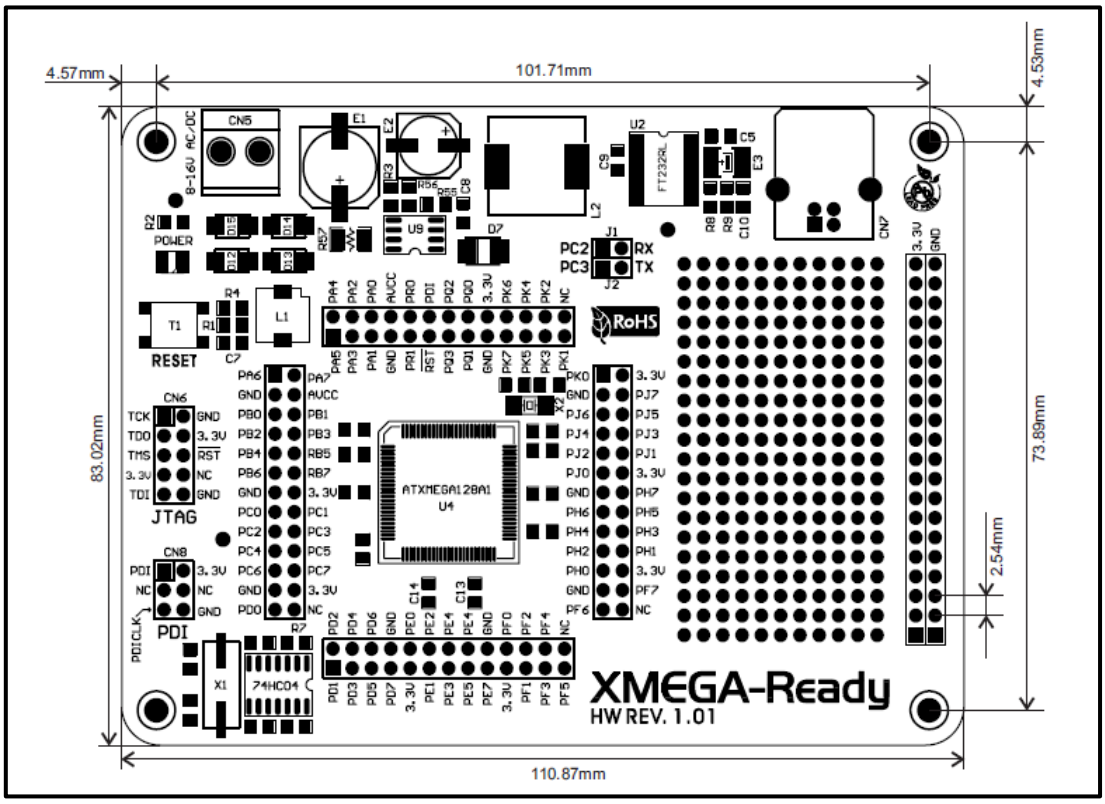
El conector CN3 (PDI) se utiliza para programar y/o depurar usando la interface PDI. El conector CN6 (JTAG) se utiliza para programar y/o depurar usando la interface JTAG.

Figura 3. Diagrama de dimensiones de la tarjeta xmega ready



Fuente:(XMega Ready, 1998)

Figura 4. Diagrama de dimensiones de la tarjeta xmega ready



Fuente: (XMega Ready, 1998)

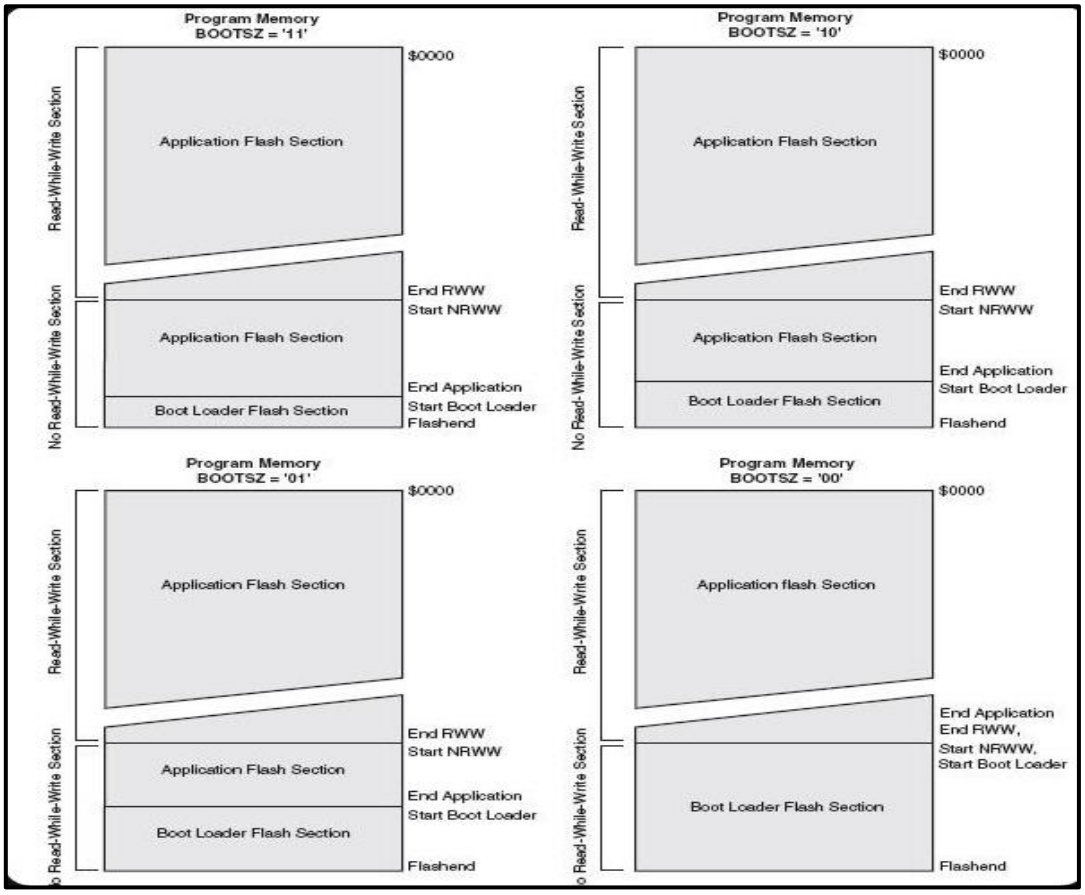
ANEXO 2

MANUAL PARA EL USO DEL MIKROBOOTLOADER

¿Qué es un bootloader?

Un bootloader es un pequeño programa que grabado previamente en un área especial de la flash (la zona o área de booteo), permitirá la actualización de la flash. Es decir que una vez que el microcontrolador tiene el bootloader ya no se necesita un programador para volver a actualizar las aplicaciones. Esta zona de booteo se encuentra siempre al final de la flash, y su tamaño puede variar entre 4 valores (que dependerán del tamaño de la flash). La elección del tamaño de ésta área se debe realizar en los fuses como se muestra en la siguiente figura.

Figura 1. Zona de booteo



Fuente: (Mikroelektronica, 2008)

Se puede ver que se definen dos áreas, la de Aplicación y la de Booteo. En la sección de aplicación es donde normalmente se coloca el código (inicia en la dirección 0x0000 de la flash).

Todos los periféricos funcionan igual en las dos áreas, pero en el área de booteo sí se puede usar la instrucción SPM (SPM - Store Program Memory), ésta instrucción es la que permite modificar la memoria flash. La memoria flash está dividida en páginas (el tamaño de la página dependerá del microcontrolador). El bootloader puede recibir los datos para la programación por cualquier medio que le sea posible (UART, SPI, I2C, USB, ETHERNET…etc.), éste bootloader lo realiza por el USB.

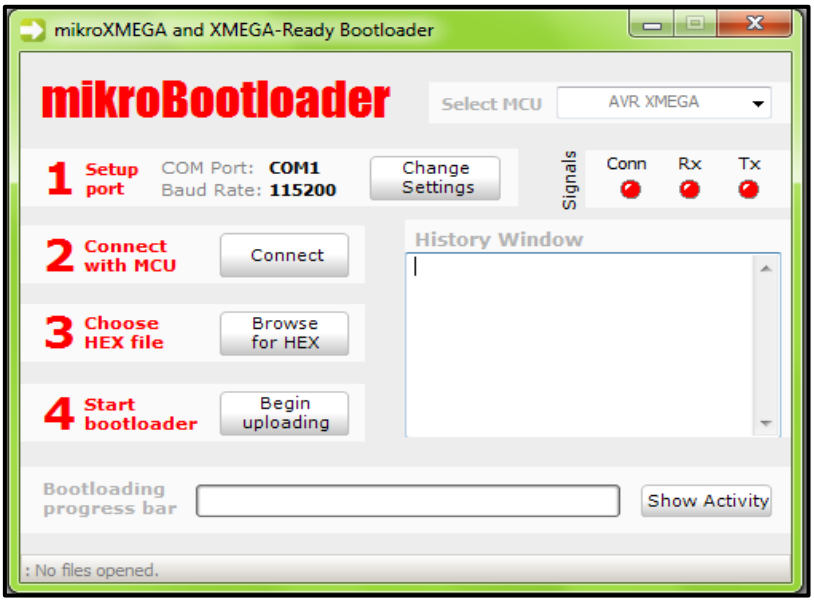
Mikrobootloader para avr xmega

El mikrobootloader se utiliza únicamente para el microcontrolador ATxmega128A1. Para habilitar el funcionamiento apropiado del chip es necesario instalar el controlador para el sistema operativo utilizado.

Los controladores se descargan desde la siguiente página web:

- http://www.ftdichip.com/drivers/VCP.htm

Figura 2. Entorno del mikrobootloader



Fuente: •(Mikroelektronica, 2008)

Pasos para usar el gestor de arranque

1) Configurar el puerto

Para configurar el puerto se debe pulsar sobre el botón *Change Settings* y se selecciona el puerto COM apropiado en la ventana *Setup.* Los otros parámetros están configurados por defecto y no se deben cambiar. Una vez realizado esto pulsamos OK para continuar.

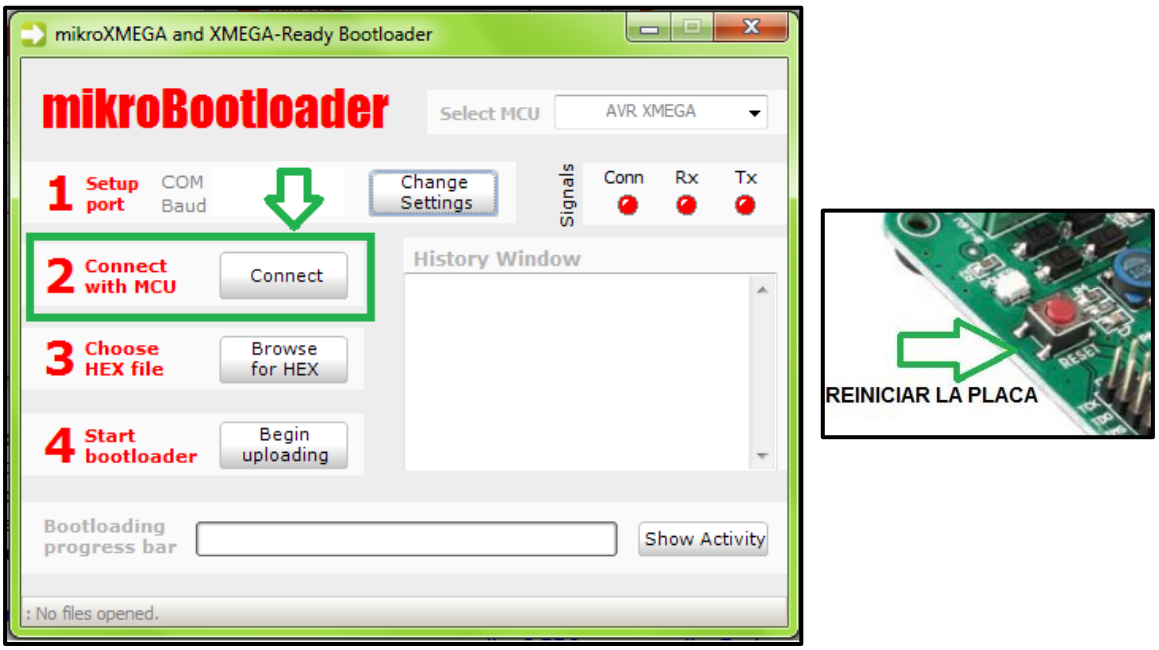
Figura 3. Configurar el puerto

Fuente: •(Mikroelektronica, 2008)

2) Conectar con el MCU

Para configurar el MCU se debe reiniciar la placa, posteriormente de pulsa sobre el botón *Connect* por 5 segundos, una vez realizados estos pasos el microcontrolador entra automáticamente en el modo de cargador de arranque.

Figura 4. Conectar con el MCU

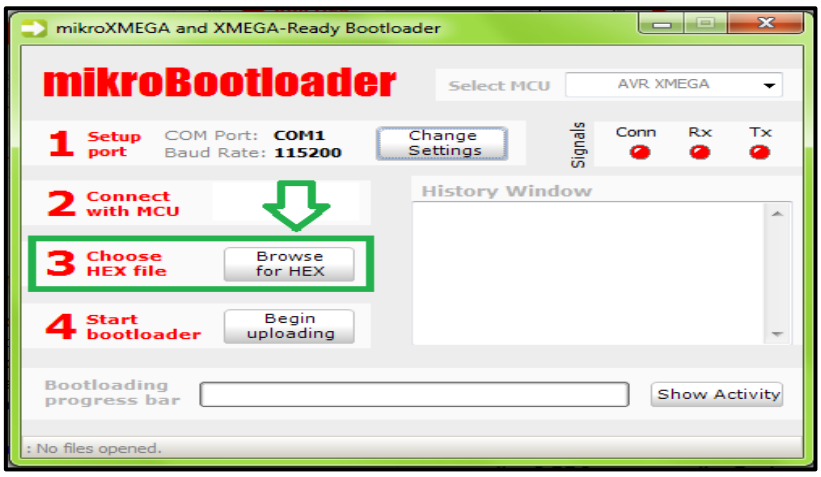


Fuente:(Mikroelektronica, 2008)

3) Elegir el archivo hex

Al pulsar el botón *Browse for Ex* (o simplemente arrastrar y soltar el código HEX en la ventana del MikroBootloader) se elige el programa que se quiere cargar en el microcontrolador; la ventana *History Window* informa sobre el archivo que está abierto.

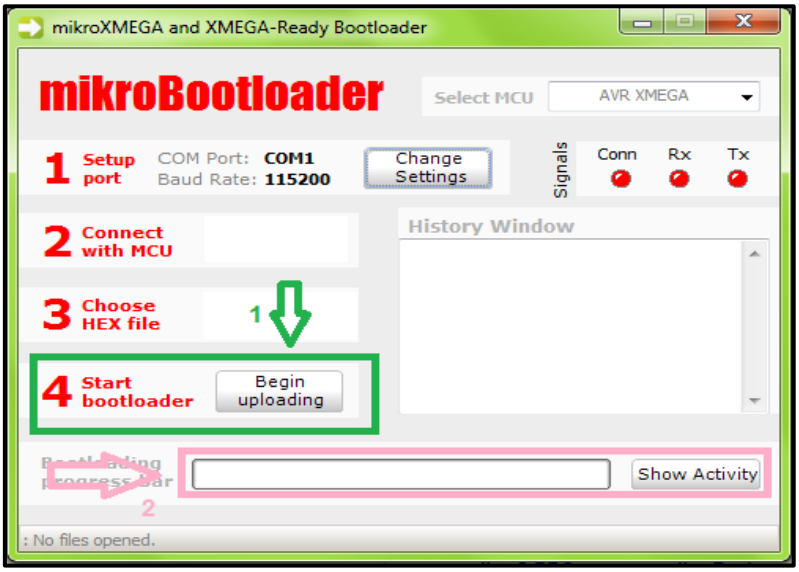
Figura 5. Elegir el archivo .hex



Fuente: (Mikroelektronica, 2008)

Cuando se pulsa el botón *Begin uploading* se inicia el cargador de arranque. Todo el proceso se realiza en dos etapas, al pulsar sobre el botón *Show Activity* se visualiza el funcionamiento del cargador de arranque.

Figura 6. Iniciar el cargador de arranque

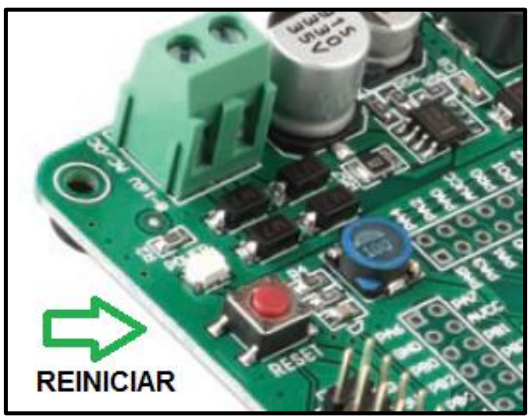


Fuente: (Mikroelektronica, 2008)

4) Reiniciar el chip

Una vez completado el proceso de carga, se debe reiniciar el chip para iniciar el nuevo programa.

Figura 7. Reiniciar el chip



Fuente: (Mikroelektronica, 2008)

ANEXO 3

MANEJO DE PUERTOS

PRÁCTICA 1

- **Tema**

  Entradas y Salidas en el microcontrolador XMega de Atmel

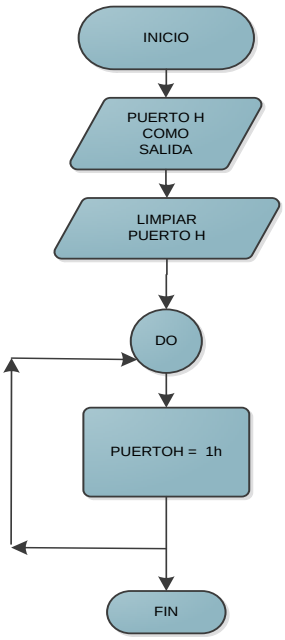
- **Objetivo**

  Utilizar los puertos del microcontrolador XMega de Atmel, como interface digital al mundo exterior.

- **Desarrollo**

1) Escribir en el puerto H el valor 1h

Diagrama de flujo para escribir 1h



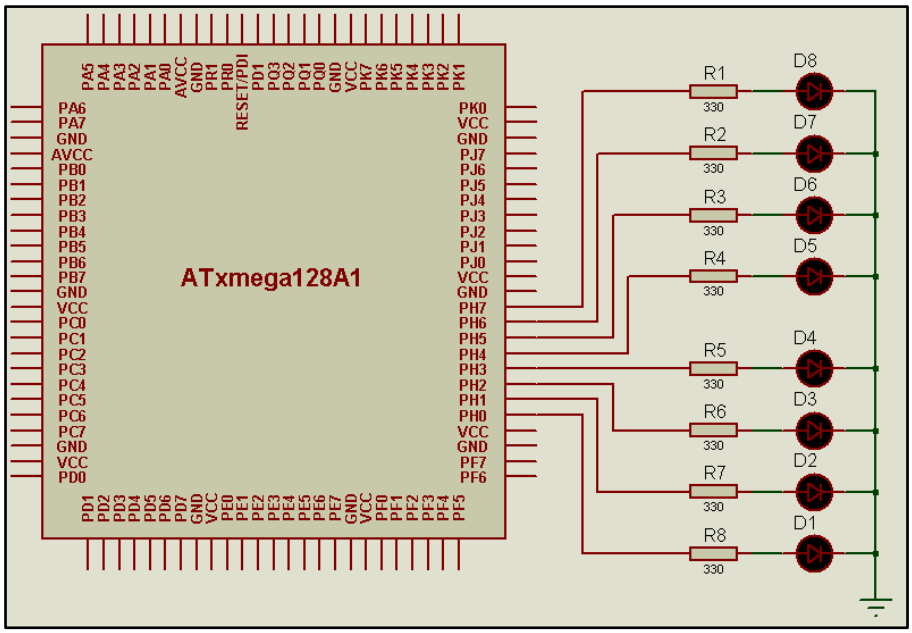
Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- Se configuran los pines del puerto H como salida
- *Limpiar puerto H*
- Inicio del lazo
- *Puerto H= 1h*
- *Cerrar lazo*
- *Fin*

CÓDIGO

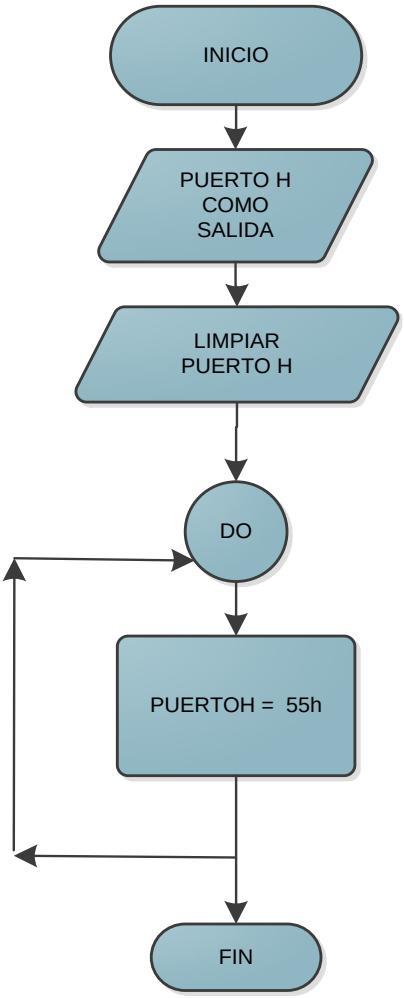
```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

Ddrh = 1                          'PUERTO H COMO SALIDA
Porth = 0                         'LIMPIAR PUERTO H

Do                                'INICIO DE LAZO
  Porth.0 = 1                     'PUERTO H.0=1
  Wait 1                          'TIEMPO DE ESPERA 1 seg
Loop                              'FIN DE LAZO
```

ESQUEMÁTICO



2) Escribir en el puerto H el valor 55h, utilizar el diagrama esquemático del ejercicio 1.

Diagrama de flujo para escribir 55h



Elaborado por: Francisco Reyes y Nina Chicaiza

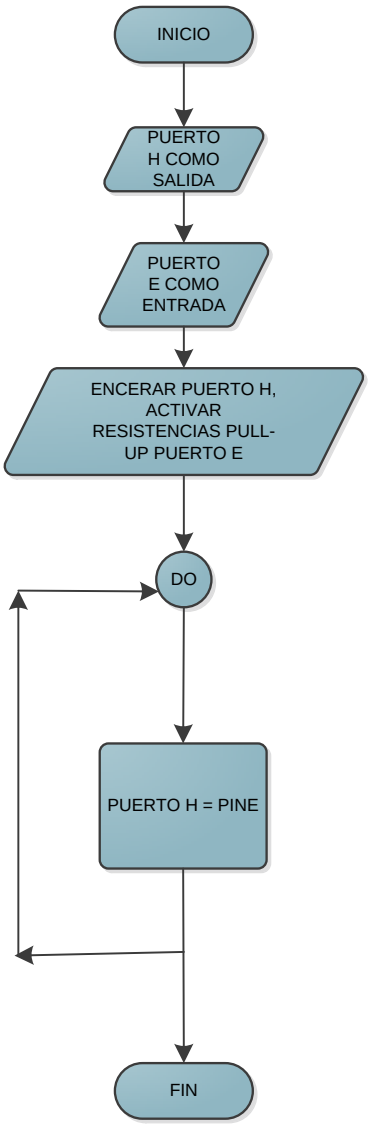
- *Inicio*
- *Puerto H como salida*
- *Limpiar puerto H*
- Inicio del lazo
- *Puerto H= 55h*
- *Cerrar lazo*
- *Fin*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

Ddrh = 1                          'PUERTO H COMO SALIDA
Porth = 0                         'LIMPIAR PUERTO H
Do
                                  'INICIO DE LAZO
  Porth = &H55                    'PUERTO H =55Hex
  Wait 1                          'TIEMPO DE ESPERA 1 SEGUNDO
Loop                              'FIN DE LAZO
```

3) Leer en el puerto E y escribir en el puerto H.

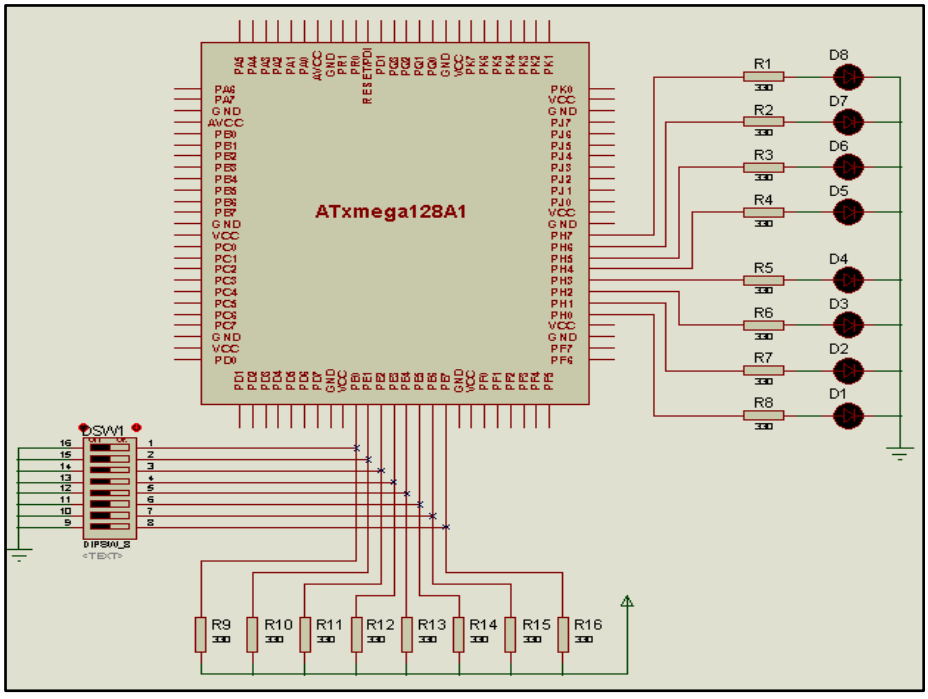
Diagrama de flujo para leer en el puerto y escribir en el puerto h



Elaborado por: Francisco Reyes y Nina Chicaiza

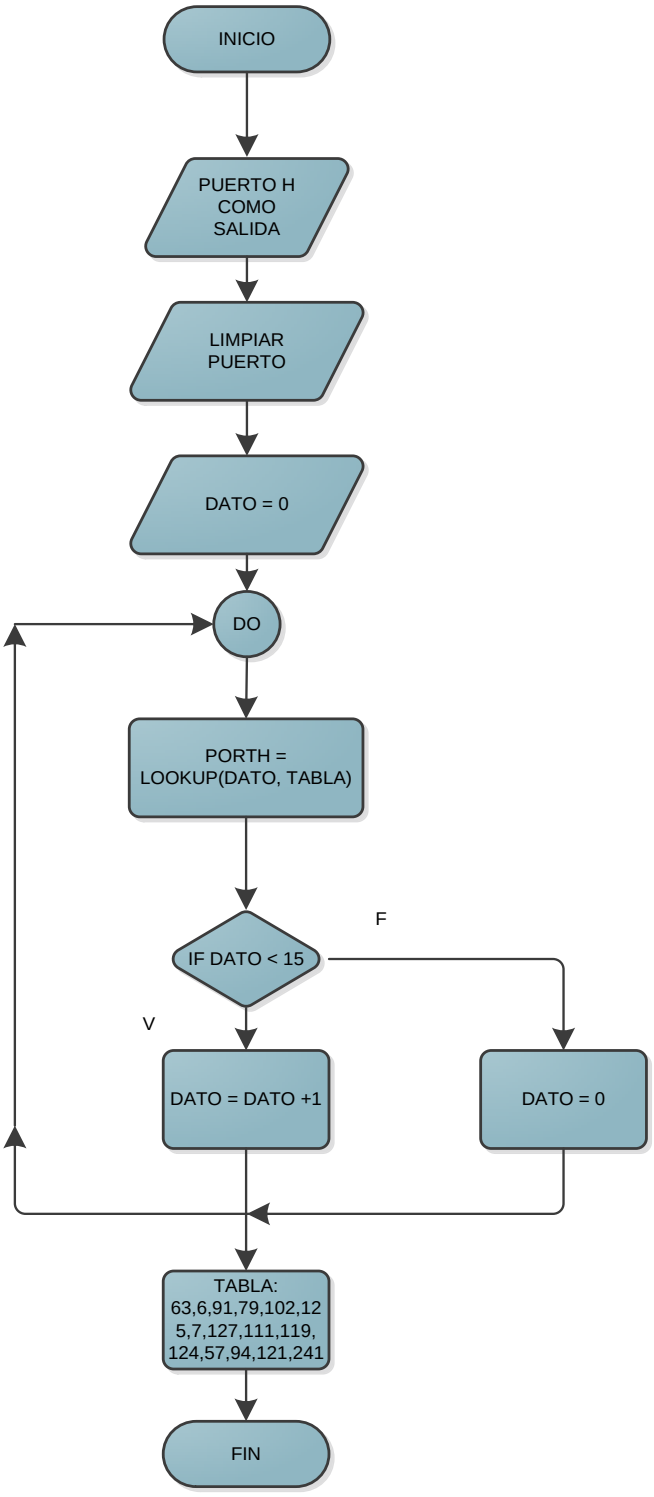
- *Inicio*
- *Puerto H como salida*
- *Puerto E como entrada*
- *Encerar puerto H*
- *Activar resistencias pull-up en puerto E*

- Inicio del lazo
- *Puerto H= PINE:*
- *Cerrar lazo*
- *Fin*

CÓDIGO
$regfile = "xm128a1def.dat" $crystal = 2000000 $hwstack = 64 $swstack = 40 $framesize = 40 Ddrh = 255 ‘PUERTO H COMO SALIDA Porth = 0 'LIMPIAR EL PUERTO H Ddre = 0 'PUERTO E COMO ENTRADA Porte = 0 'ACTIVAR RESISTENCIAS DE PULL-UP Do 'INICIO DE LAZO   Porth = Pine 'LEER EL PUERTO E Y MOSTRAR EN PUERTO H Loop 'FIN DE LAZO
ESQUEMÁTICO


4) Escribir en un display de 7 segmentos sin utilizar decodificador, un valor ascendente entre 0 y F

Diagrama de flujo para escribir en un display de 7 segmentos un valor ascendente entre 0 y f.



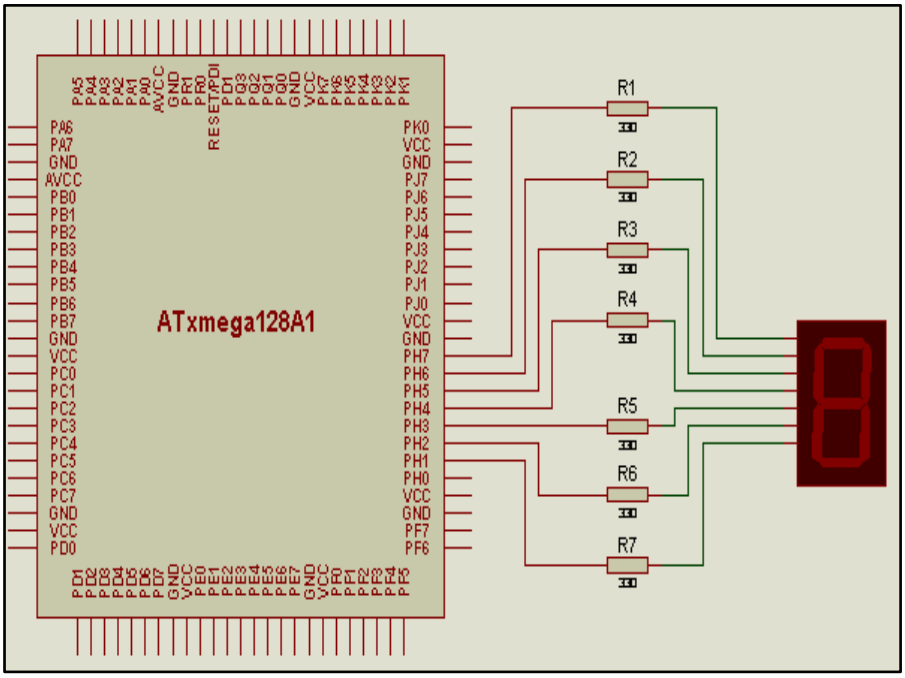
Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto H como salida*
- *Limpiar puerto*
- *Variable byte Dato=0*
- Inicio del lazo
- *PortH = lookup(dato, tabla)*
- *Si dato<15*
- *Por verdad, Dato=dato+1*
- *Por falso, Dato=0*
- *Finalizar condición*
- *Fin del lazo*
- *Fin*

CÓDIGO

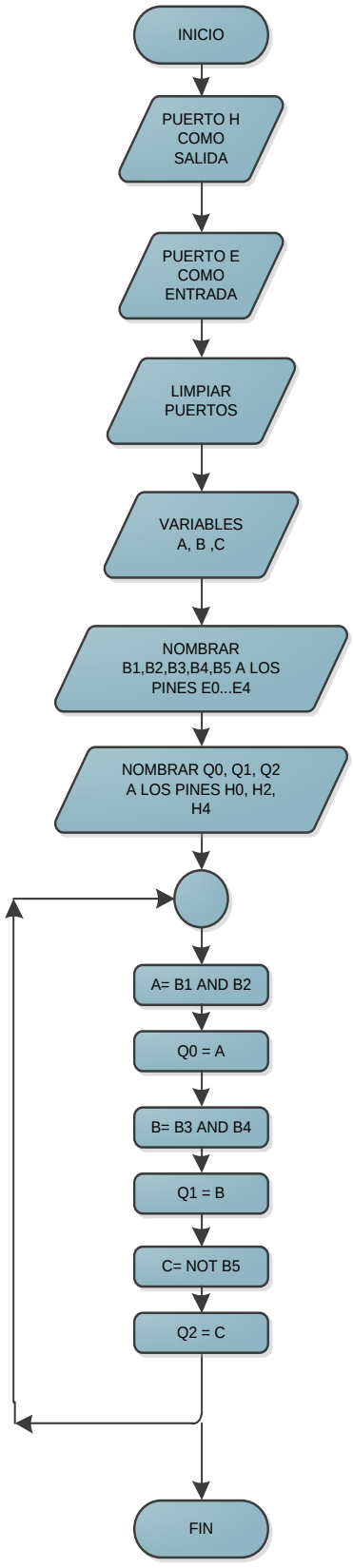
```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

Dim Dato As Byte                'DEFINIR LA VARIABLE Dato COMO BYTE

Ddrh = 255                      'CONFIGURACION DE PUERTOS DE SALIDA
Porth = 0                           'LIMPIAR PUERTOS

Do                                  'LAZO PRINCIPAL

  Porth = Lookup(dato , Tabla)  'RELACIONA EL DATO CON LA POSICION
                                    'CORRESPONDIENTE EN LA TABLA
   If Dato < 15 Then
   Incr Dato
  Else
  Dato = 0
  End If
  Wait 1
Loop
End

Tabla:
Data 63 , 6 , 91 , 79 , 102 , 109 , 125 , 7 , 127 , 111 , 119 , 124 , 57 , 94 , 121 , 241

'VALORES PARA ENCENDER EL DISPLAY
'CON 63 = 00111111 MUESTRA EL 0
'CON 6 = 00000110 MUSTRA EL 1
‘ETC, ETC
```

ESQUEMÁTICO



5) Lógica booleana con el microcontrolador, utilizar el diagrama esquemático del ejercicio 3.

Diagrama de flujo para realizar lógica booleana



- *Inicio*
- *Puerto H como salida*
- *Puerto E como entrada*
- *Limpiar puertos*
- *Variables A, B, C tipo byte*
- *Nombrar B1, B2, B3, B4, B5*
- *Alias PIN E0…E4*
- *Nombrar Q0, Q1, Q2*
- *Alias PORT H0, H2, H4*
- *Abrir lazo*
- *A= B1 AND B2*
- *Q0=A*
- *B= B3 AND B4*
- *Q1= B*
- *C= NOT B5*
- *Q2= C*
- *Fin del lazo*
- *Fin*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

'---OPERACIONES BOOLEANAS---'

Ddrh = 255              'CONFIGURACION DE PUERTOS DE SALIDA
Porth = 0               'LIMPIAR PUERTOS

Ddre = 0                'CONFIGURAR POUERTOS DE ENTRADA
Porte = 0               'ACTIVAR RESISTENCIAS DE PULL-UP

Dim A As Byte           ‘DECLARACION DE VARIABLES A, B, C TIPO BYTE
Dim B As Byte
Dim C As Byte
```

```
'CUANDO SE NECESITE HACER USO DE NOMBRES QUE REPRESENTEN UN
‘PUERTO ESPECIFICO SE USA LA PALABRA ALIAS

B1 Alias Pine.0          'SE USA LA PALABRA PIN CUANDO VA A SER
B2 Alias Pine.1           ‘UN PUERTO POR  EL QUE ENTRARAN DATOS
B3 Alias Pine.2
B4 Alias Pine.3
B5 Alias Pine.4

Q0 Alias Porth.0         'PORT CUANDO VA A SER UN PUERTO
Q1 Alias Porth.2         ‘POR EL QUE SACAREMOS DATOS
Q2 Alias Porth.4

Do
  A = B1 AND B2            ‘OPERACION AND
  Q0 = A                   ‘MOSTRAR POR EL PORT Q0 EL RESULTADO

  B = B3 OR B4             ‘OPERACIÓN OR
  Q1 = B                   ‘MOSTRAR POR EL PORT Q1 EL RESULTADO

  C = NOT B5               ‘OPERACIÓN NOT
  Q2 = C                   ‘MOSTRAR POR EL PORT Q2 EL RESULTADO

Loop
```

6) Ejercicio de aplicación: Escribir en el puerto F el valor AA

7) Ejercicio de aplicación: Escribir en el puerto B el valor F0

8) Ejercicio de aplicación: Leer en el puerto H y escribir en el puerto E

9) Ejercicio de aplicación: Mediante un display alfanumérico mostrar en orden descendente todas las letras del alfabeto de la Z a la A

ANEXO 4

USO DE DECLARACIONES

PRÁCTICA 2

- **Tema**

  Utilización de declaraciones con el microcontrolador XMega de Atmel

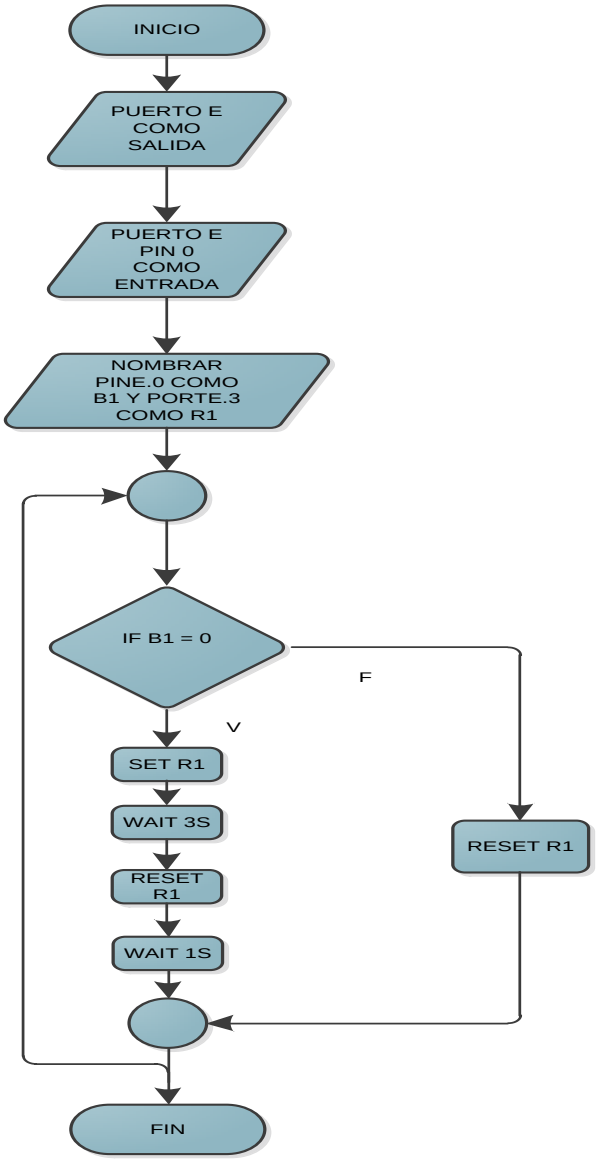
- **Objetivo**

  Utilizar las declaraciones if, select case, do, for, while con el microcontrolador XMega de Atmel.

- **Desarrollo**

1) Si se presiona el botón el foco se enciende por 3 segundos, luego se apaga 1 segundo y se vuelve a encender el foco por 3 segundos. Caso contrario el foco permanece prendido.

Diagrama de flujo para uso de declaración if



Elaborado por: Francisco Reyes y Nina Chicaiza

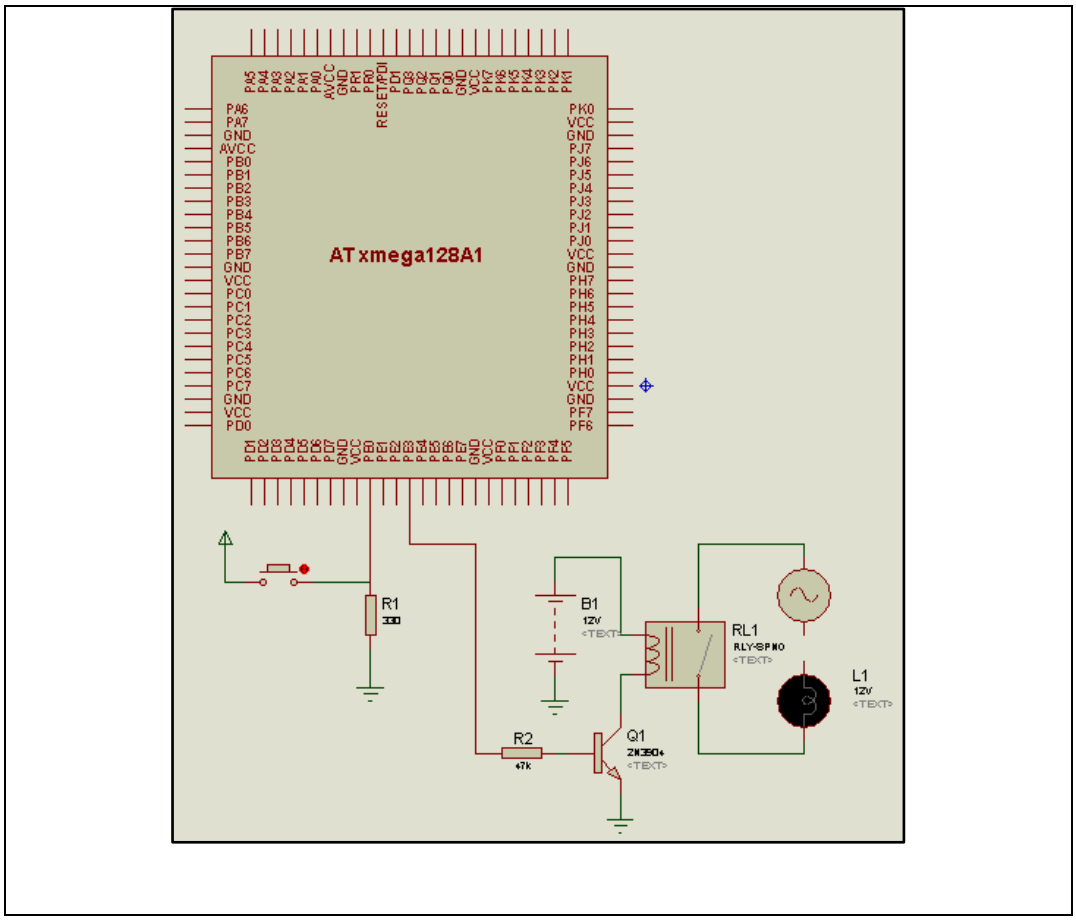
- *Inicio*
- *Puerto E* = 11111110
- *Nombrar PINE.0 = B1*
- *Nombrar PORTE.3 = R1*
- *Abrir un lazo*
- *IF B1=0*
- POR VERDAD R1 = 1
- POR FALSO R1 = 0

- *Finalizo Condición*
- *Cierro lazo*
- *Fin de programa*

CÓDIGO

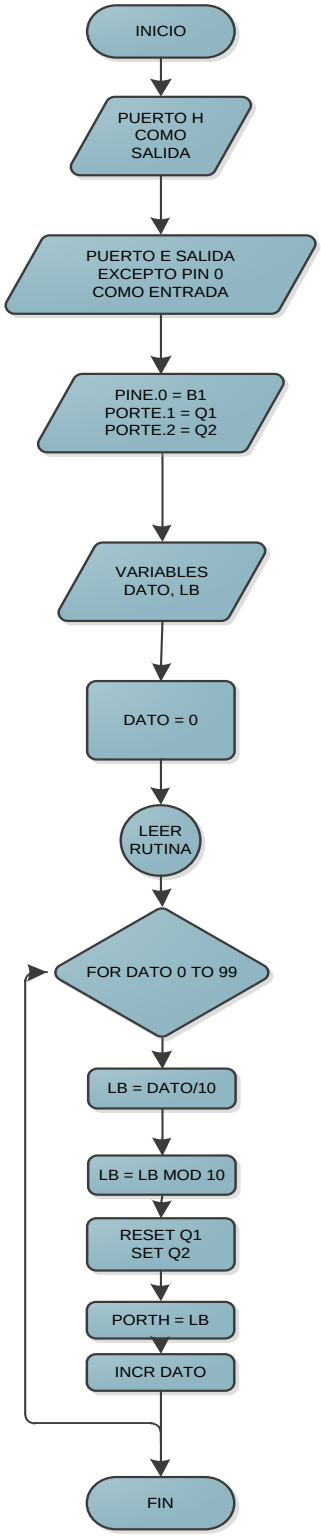
```
$regfile = "xm128a1def.dat"
'$regfile = "m16def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

'---ACTIVACION RELE TRES SEG, APGD----'

Ddrh = 255          'CONFIGURACION PUERTOS DE SALIDA
Porth = 0           'LIMPIAR EL PUERTO

Ddre = 254          'CONFIGURACION DE PUERTO COMO SALIDA
                    ‘ EXCEPTO PIN E.0 COMO ENTRADA, (&B11111110)
Porte = &H01        'ACTIVAR PULL-UP SOLO PARA E.0

B1 Alias Pine.0                 'ENTRADA PARA BOTON
Q1 Alias Porte.1                'SALIDA TRANSISTOR Q1
Q2 Alias Porte.2                'SALIDA TRANSISTOR Q2
Rl Alias Porte.3                'SALIDA TRANSISTOR RELE

Do                              'INICIO DE LAZO
  If B1 = 0 Then                'INICIO DE CONDICION
  Reset Rl                      ‘POR VERDAD
  Wait 1                        ‘RESET asigna un 0
  Set Rl                        ‘SET asigna un 1
  Wait 3                        ‘WAIT retardo en segundos
  Reset Rl
  Wait1
  Set Rl
  Wait 3
  Reset Rl
  Wait 1
  Else                          'POR FALSO
  Reset Rl
  Wait 1
  End If                        'FIN DE CONDICION
Loop                            'FIN DEL LAZO
```

ESQUEMÁTICO



2) Utilizando la sentencia For –Next, elaborar un contador ascendente de 0 a 9, que sea repetitivo.

Diagrama de flujo para uso de declaración for-next



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto H como salida*
- *Puerto E* = 11111110
- *Nombrar PinE.0=B1*

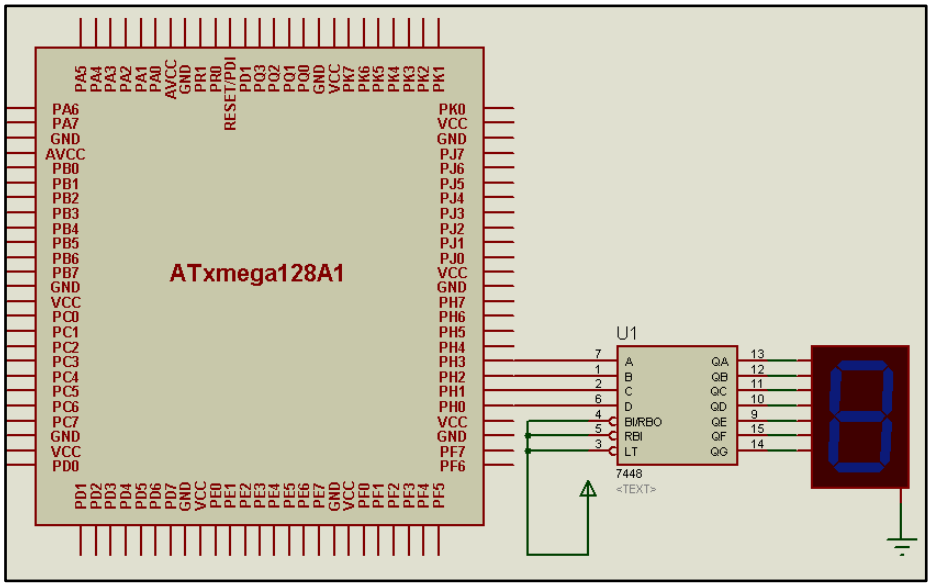
- *Nombrar PortE.1=Q1*
- *Nombrar PortE.2=Q2*
- Declarar dos variables, DATO y LB
- *Dato=0*
- *Abrir rutina*
- *For Dato de 0 a 99*
- *LB= Dato/10*
- *LB= LB MOD 10*
- *Reset Q1*
- *Set Q2*
- *PortH = LB*
- *Incr Dato*
- *Fin de rutina*
- *Fin de programa*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

'---CONTADOR CON FOR DISPLAY---'

Dim Dato As Byte      ' VAR Dato ACUMULARA LOS VALORES DEL 1 AL 99
Dim Lb As Byte        ‘VAR Lb GUARDARÁ EL DIG MENOS SIGNIFICATIVO
Ddrh = 255           'CONFIGURACION PUERTO H COMO SALIDA
Porth = 0            'LIMPIAR PUERTO H
Ddre = 255           'CONFIGURACION PUERTO E COMO ENTRADA
Porte = 0            'LIMPIAR PUERTO
Q1 Alias Porte.1      'SALIDAS PARA CONTROLAR LOS TRANSISTORES
Q2 Alias Porte.2      'SALIDAS PARA CONTROLAR LOS TRANSISTORES
Dato = 0
Programa:                'INICIO DE SUB-PROGRAMA
For Dato = 1 To 99        'INICIO LAZO FOR
                          'SERA UNA REPETICION DE 99 VECES
  Lb = Dato / 10
  Lb = Lb Mod 10          'SE TOMA EL RESIDUO

  Reset Q1 : Set Q2      'APAGAR TRANSISTOR Q1, ACTIVAR Q2
  Porth = Lb              ‘IMPRIMIR Lb EN EL PUERTO H
  Wait ms 100             'RETARDO EN milisegundos (100 ms)

  Dato = Dato + 1         'INCREMENTO DEL DATO
```

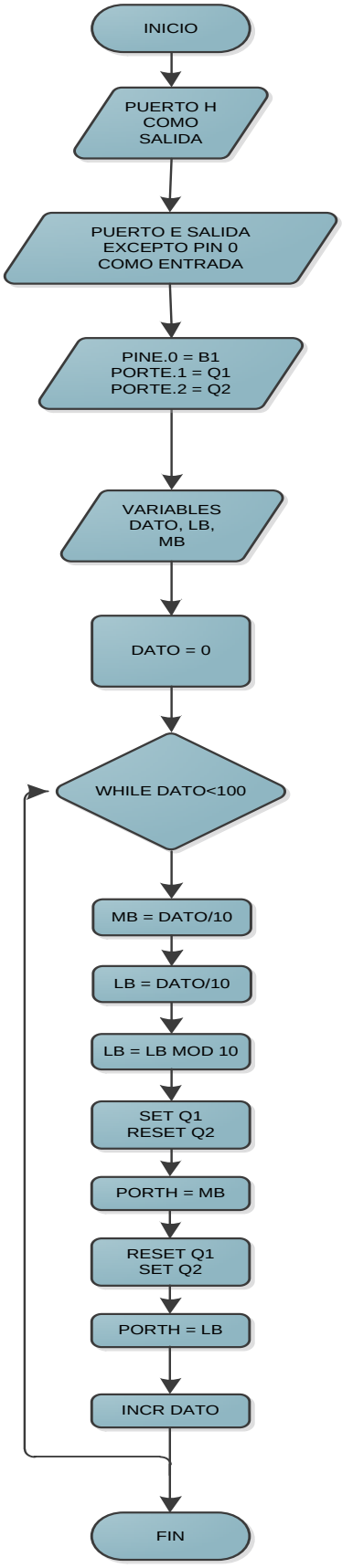
Next	'FIN DEL LAZO FOR
Goto Programa	'RETORNO A SUB-PROGRAMA
End	

ESQUEMÁTICO



3) Utilizar la secuencia While, para elaborar un contador ascendente de dos dígitos.

Diagrama de flujo para uso de secuencia while



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*

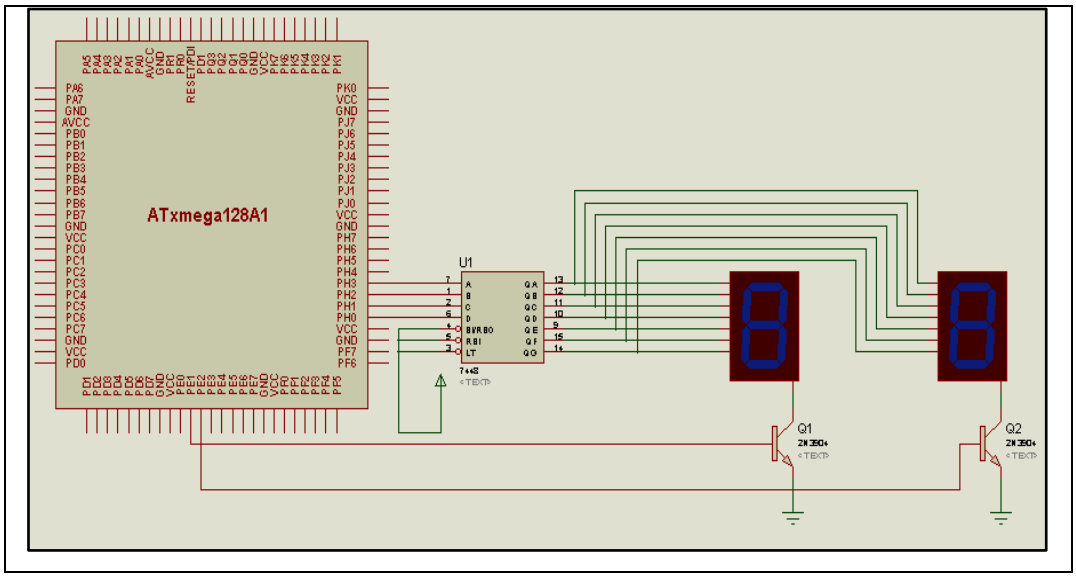
- *Puerto H como salida*
- *Puerto E* = 11111110
- *Nombrar PinE.0=B1*
- *Nombrar PortE.1=Q1*
- *Nombrar PortE.2=Q2*
- Declarar tres variables, DATO, MB, LB
- *While dato <100*
- *Mb= dato/10*
- *Lb= Lb MOD 10*
- *Set Q1, Reset Q2:* Activar Q1 y apagar Q2.
- *PortH= Mb*
- *Reset Q1, Set Q2*
- *PortH=Lb*
- *Incr Dato*
- *Wend*
- *Fin*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

Dim Dato As Byte          ‘DECLARAR 3 VARIABLES Dato, Mb, Lb
Dim Mb As Byte
Dim Lb As Byte

Ddrh = 255                'CONFIGURACION DE PUERTOS DE SALIDA
Porth = 0
Ddre = 255                'CONFIGURACION DE PUERTOS DE SALIDA
Porte = 0

Q1 Alias Porte.1          ‘TRANSISTOR 1
```

```
Q2 Alias Porte.2          ‘TRANSISTO 2

Dato = 0

While Dato < 100          'CONDICION WHILE

  Mb = Dato Mod 10        ‘RESIDUO DE LA DIVISION Dato/10
  Lb = Dato / 10
  Lb = Lb Mod 10          ‘RESIDUO DE LA DIVISION Dato/100

  Incr Dato               'INCREMENTO EN UNO A LA VARIABLE DATO

  Set Q1 : Reset Q2       'ENCENDER EL TRANSISTOR DEL DISPLAY
                          ‘MAS SIGNIFICATIVO Q1

  Porth = Mb              'MOSTRAR EL BIT MAS SIGNIFICATIVO
  Waitms 18               'RETARDO EN MILISEGUNDOS

  Reset Q1 : Set Q2       'ENCENDER EL TRANSISTOR DEL DISPLAY
                          ‘MENOS SIGNIFICATIVO

  Porth = Lb              'MOSTRAR EL VALOR MENOS SIGNIFICATIVO
  Waitms 200              'RETARDO EN MILISEGUNDOS
Wend                      'FIN DE CONDICION
```

ESQUEMÁTICO



4) Ejercicio de aplicación: Mediante la sentencia For-netx, elaborar un contador descendente entre 0 y9.

ANEXO 5

TEMPORIZADORES

PRÁCTICA 3

- **Tema**

  Temporizadores

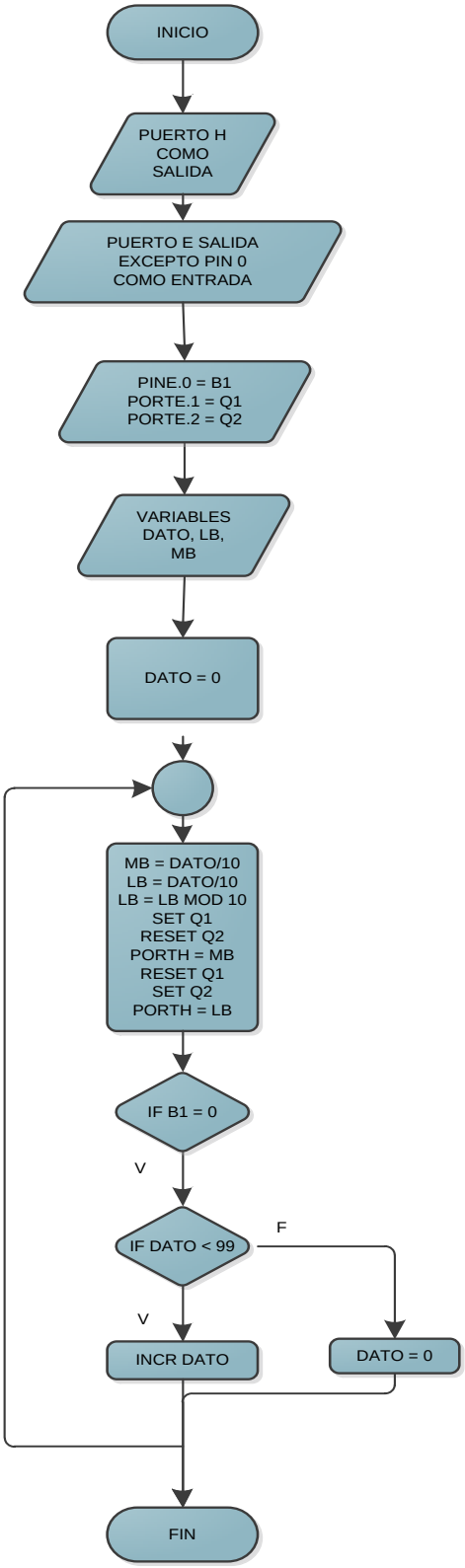
- **Objetivo**

  Utilizar los temporizadores del microcontrolador XMega de Atmel.

- **Desarrollo**

1) Utilizar el temporizador para elaborar un contador de 1 segundo, cuyo valor se observa en un display de 7 segmentos

Diagrama de flujo para uso del temporizador



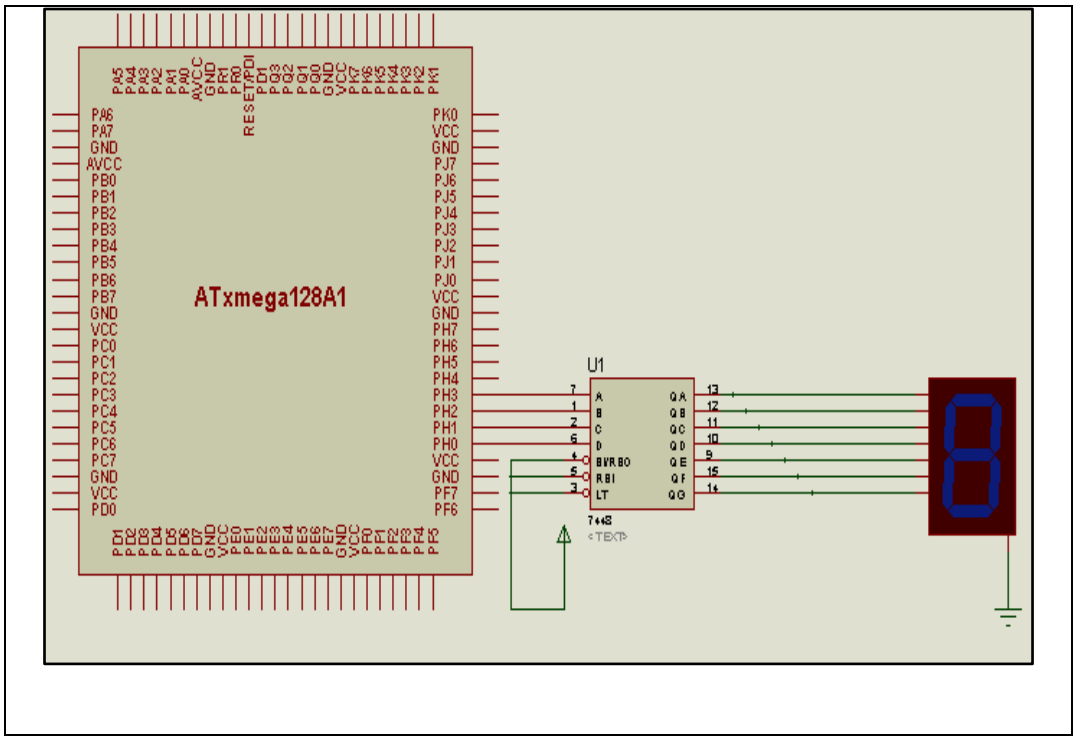
Elaborado por: Francisco Reyes y Nina Chicaiza

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

'CONFIGURACION DEL TEMPORIZADOR

Config Priority = Static , Vector = Application , Lo = Enabled
On Tcc0_ovf Tiempo          'SUBRUTINA Tiempo TEMPORIZADA
Enable Tcc0_ovf , Lo        'HABILITAMOS TEMPORIZADOR
Enable Interrupts

'SELECCIONAMOS EL RELOJ PARA EL TIMER   0101 = Preescales=clk/64

  Tcc0_ctrla = &B00000101

'--------CONTADOR CON FOR DISPLAY--------'

Dim Dato As Byte            ‘ES DONDE SE ACUMULARAN LOS
                            ‘VALORES DEL 1 AL 99
Dim Lb As Byte

Ddrh = 255                  'CONFIGURACION PUERTO DE SALIDA
Porth = 0                   'LIMPIO PUERTO H
Ddre = 254                  ‘CONFIGURACION PUERTO E COMO
                            ‘ENTRADA  11111110 TODOS EXCEPTO PINE.0
Porte = &H01                'ACTIVAMOS RESISTENCIAS DE PULL UP
                            ‘SOLO PARA E.0

Q1 Alias Porte.1            'SALIDAS PARA CONTROLAR LOS TRANSISTORES
Q2 Alias Porte.2            ‘SALIDAS PARA CONTROLAR LOS TRANSISTORES

Dato = 0
```

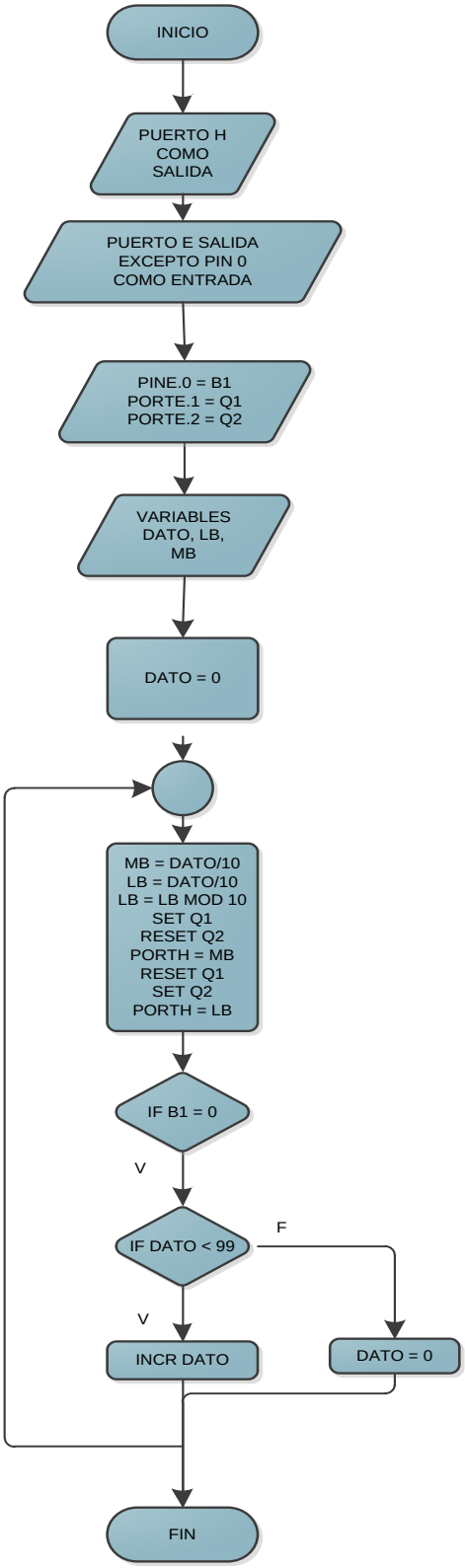
```
Do                          'LAZO PRINCIPAL

  Porth = Lb                'IMPRIMIR EN EL DISPLAY

Loop

Tiempo:
      Tcc0_per = 1000       'MAS ALTO EL PERIODO MAYOR EL RETARDO
  If Dato < 99 Then
    Dato = Dato + 1
    Lb = Dato / 10
    Lb = Lb Mod 10          'TOMAMOS RESIDUO DE DIVIDIR LB PARA 10
    Reset Q1 : Set Q2
    Else
    Dato = 0
  End If

Return
```

ESQUEMÁTICO



2) Elabore un programa que utilice una entrada, y se visualice el incremento en un display de 7 segmentos.

Diagrama de flujo para uso del temporizador



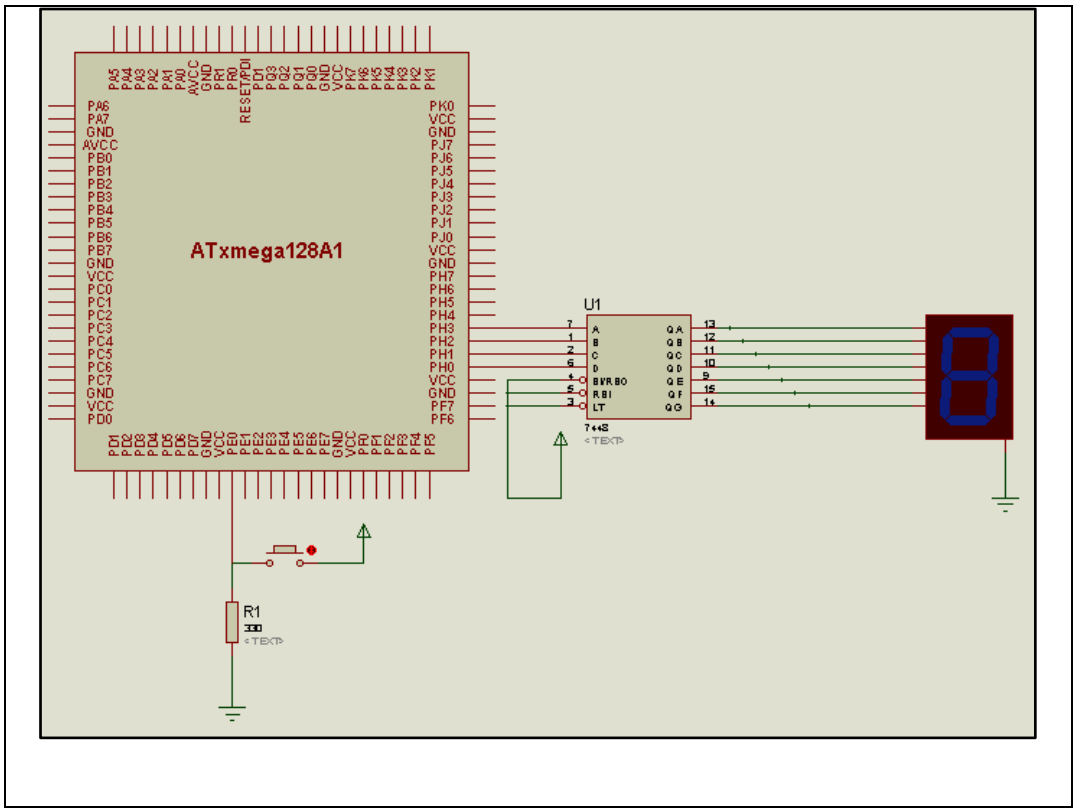
Elaborado por: Francisco Reyes y Nina Chicaiza

CÓDIGO

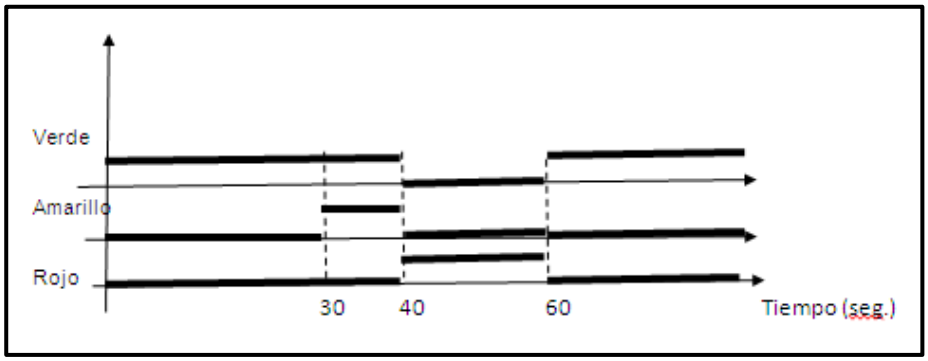
```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

'--------------'
'CONFIGURACION DEL TEMPORIZADOR
Config Priority = Static , Vector = Application , Lo = Enabled
On Tcc0_ovf Tiempo                'SUBRUTINA TIEMPO TEMPORIZADA
Enable Tcc0_ovf , Lo              'HABILITAMOS TEMPORIZADOR
Enable Interrupts

Tcc0_ctrla = &B00000100
'SELECCIONAMOS EL RELOJ PARA EL TIMER   0100 = Preescales=clk/8
'------------'

Dim Dato As Byte
Dim Mb As Byte
Dim Lb As Byte
Dim Aux As Byte

Ddrh = 255              'CONFIGURACION DE PUERTOS
Porth = 0
Ddre = 254              'CONFIGURACION DE PUERTOS
Porte = &H01

B1 Alias Pine.0         'DEF COMO ENTRADA DE PULSADOR
Q1 Alias Porte.1
Q2 Alias Porte.2

Dato = 0
Do
  Mb = Dato Mod 10
  Lb = Dato / 10
```

```
  Lb = Lb Mod 10


  Set Q2 : Reset Q1
  Porth = Mb
  Waitms 10
  Reset Q2 : Set Q1
  Porth = Lb
  Waitms 10

Loop

Tiempo:
  Tcc0_per = 50000
  If B1 = 0 Then
   If Dato < 99 Then
    Incr Dato
   Else
    Dato = 0
   End If
  End If
Return
```

ESQUEMÁTICO



3) EJERCICIO DE APLICACIÓN: Diseñar un semáforo según el siguiente diagrama secuencial. El sistema es cíclico, es decir luego de 60 segundos vuelve a repetirse la secuencia con tiempos idénticos, utilizar 3 focos de 110 voltios (verde, amarillo, rojo), optoacopladores, triacs



ANEXO 6

MANEJO DE LCD

PRÁCTICA 4

- **Tema**

  Manejo de Lcd

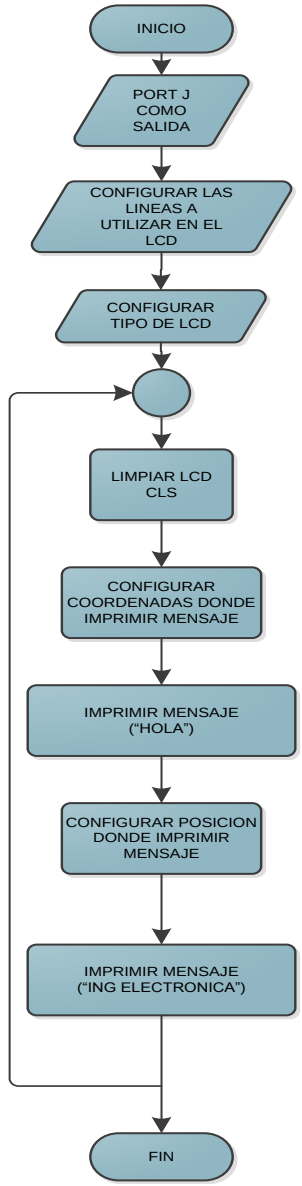
- **Objetivo**

  Utilizar con el microcontrolador XMega de Atmel display alfanuméricos.

- **Desarrollo**

1) Escribir la sentencia Hola e Ingeniería Electrónica en un LCD.

Diagrama de flujo para uso de lcd



Elaborado por: Francisco Reyes y Nina Chicaiza

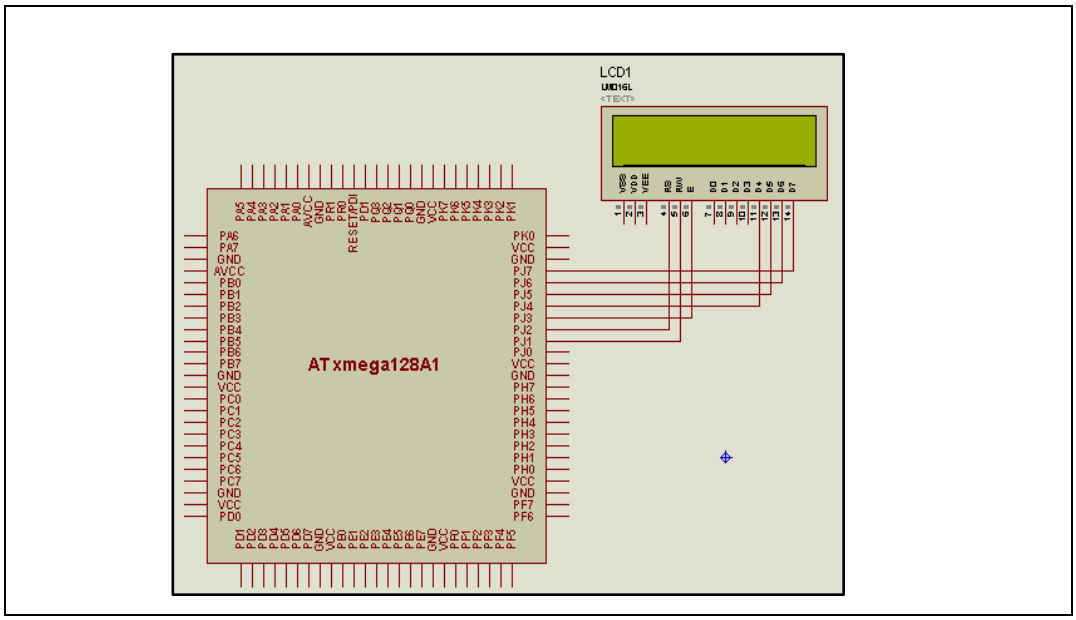
- *Inicio*
- *PortJ como salida*
- *Configurar las líneas del lcd*
- *Configurar tipo de lcd*
- *Inicio de lazo*
- *Limpiar LCD*
- *Configurar coordenadas donde se va a imprimir mensaje*
- *Imprimir mensaje ("HOLA")*
- *Imprimir mensaje ("ING ELECTRONICA")*

- *Fin de lazo*
- *Fin*

CÓDIGO

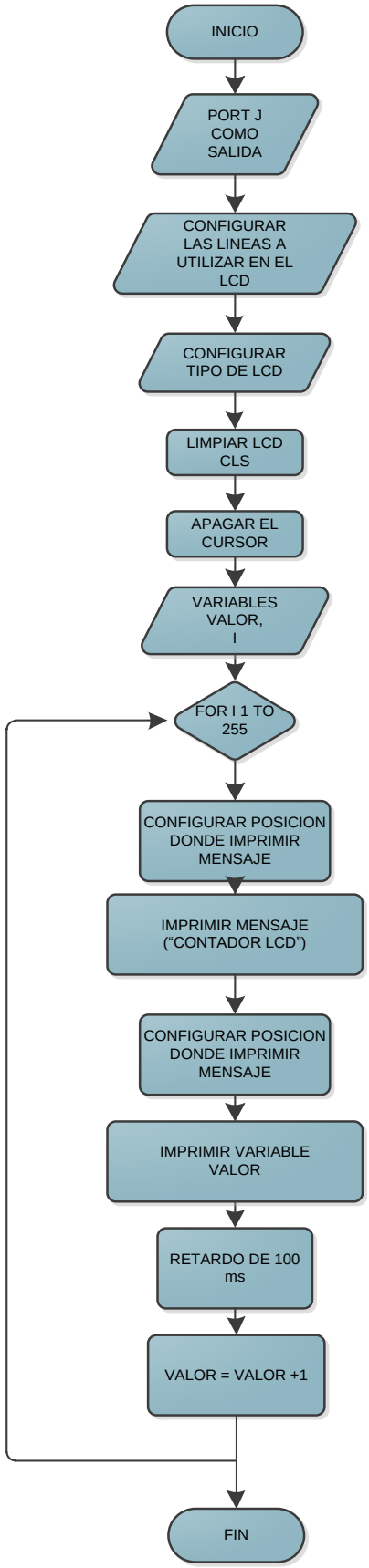
```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

Ddrj = 255                    'CONFIGURACION PUERTO J COMO SALIDA
Portj = 0                     'LIMPIAR PUERTO J

Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2      'CONFIGURAR LAS LINEAS DEL LCD

Config Lcd = 16 * 2           'CONFIGURAR TIPO DE LCD
Cls                           'LIMPIAR LA PANTALLA DEL LCD
Locate 1 , 5                  'IMPRIMIR UN MENSAJE EN FILA 1, COLUMNA 5
Lcd "Hola"
Home L                        'IMPRIMIR UN MENSAJE EN LA FILA L
                              'L= LOW FILA BAJA; H = HIGH FILA ALTA

Lcd "Ing. Electrónica"
```

ESQUEMÁTICO



2) Elabore un programa para que se observe un contador ascendente, cuyo valor se observa en un display 16x2, con un intervalo de tiempo entre cada valor de 2 segundos. Utilice el esquema del numeral uno.

Diagrama de flujo para realizar un contador en el lcd



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*

- *PortJ como salida*
- *Configurar las líneas a utilizar en el lcd*
- *Configurar tipo de lcd*
- *Inicio de lazo*
- *Limpiar LCD*
- Apagar el cursor intermitente.
- Crear dos variables Valor, i
- *FOR i de 1 a 255*
- *Configurar posición donde imprimir mensaje*
- *Imprimir variable VALOR*
- *Retardo de 100 ms*
- *Valor= Valor+1*
- *Fin de lazo*
- *Fin*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

'---LIBRERIAS PARA CONVERSION---'
$lib "XMEGA.LIB"
$external _xmegafix_clear
$external _xmegafix_rol_r1014
'---------------------------------'
Ddrj = 255                    'CONFIGURACION DE PUERTOS PARA LCD
Portj = 0                     'LIMPIAR EL PUERTO

Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2       'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2           'TIPO DE LCD
Cursor Off                    'APAGAR EL CURSOR
Cls                           'LIMPIAR LA PANTALLA
```

```
Dim Valor As Byte
Dim I As Byte

Valor = 0
Cls
For I = 1 To 255                    'LAZO PARA CUENTA DE 1 A 255
  Home U                            'IMPRIMIR EN LA FILA 1 COLUMNA 0
  Lcd "CONTADOR LCD"                 'TEXTO A IMPRIMIR
  Incr Valor                        'INCREMENTAR VARIABLE VALOR
  Home L                            'IMPRIMIR EN LA FILA 2 COLUMNA 0
  Lcd Valor ; "              "       'IMPRIMIR VARIABLE VALOR
  Wait 2                            'RETARDO DE 2 SEGUNDOS

Next
```

3) Ejercicio de aplicación: Elaborar un programa para que aparezcan los nombres de cada uno de los integrantes del grupo por 3 segundos.

4) Ejercicio de aplicación: Elaborar un programa para crear un contador por señal externa y mostrar el valor en un display, la señal del reloj se genera mediante un pulsante externo.

ANEXO 7

MANEJO DE LCD Y TECLADO

PRÁCTICA 5

- **Tema**

  Teclados y Lcd

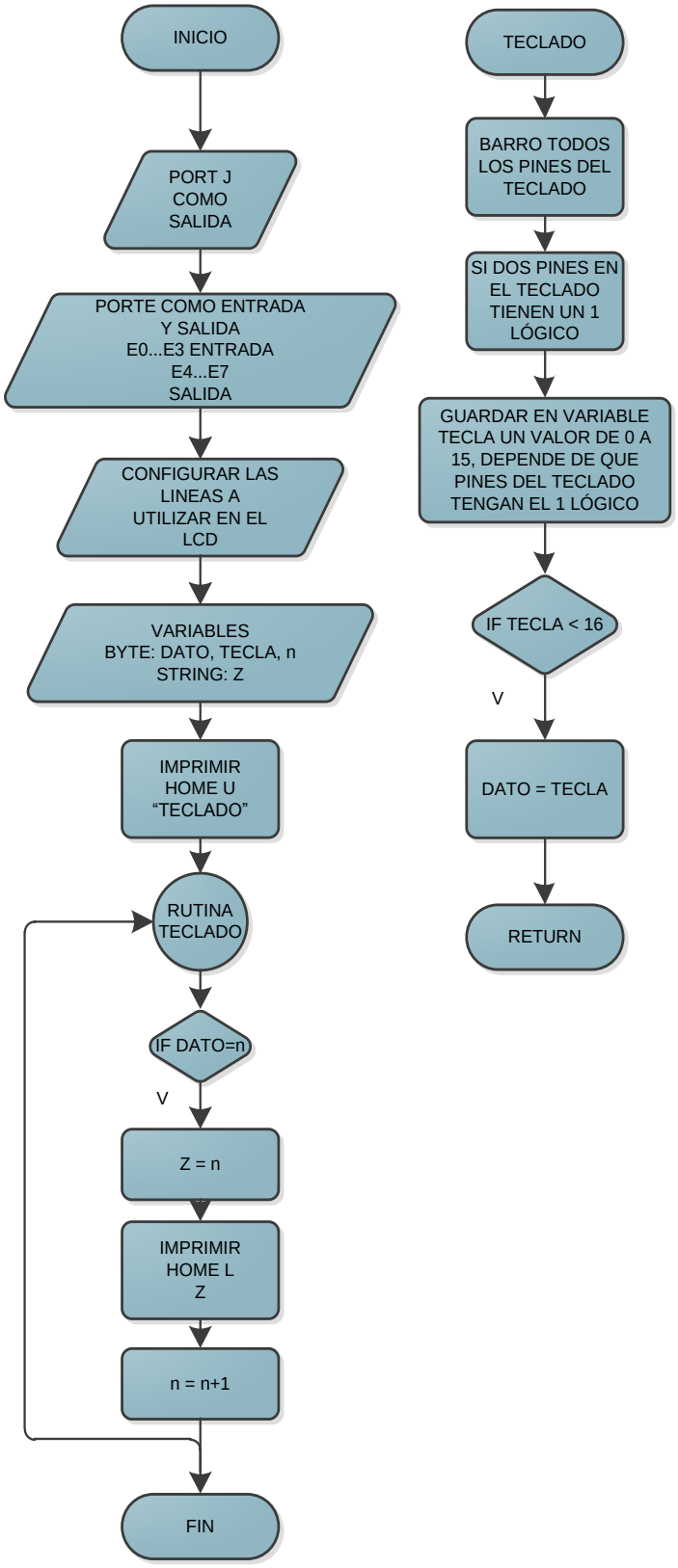
- **Objetivo**

  Utilizar el microcontrolador XMega de Atmel con teclado matricial

- **Desarrollo**

1) Visualizar en un display LCD el valor ingresado por un teclado matricial.

Diagrama de flujo para uso de lcd y teclado



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *PortJ como salida*
- *PortE = 00001111*
- *Configurar las líneas del lcd*
- *Configurar tipo de lcd*
- *Limpiar lcd*
- Variables tipo byte: Dato, tecla, n
- Variable tipo String: Z.
- *Imprimo mensaje "TECLADO"*
- *Ejecutar rutina teclado*
- Guardar valor en variable TECLA
- Si TECLA >= 0 y TECLA <= 15
- Se devuelve la variable DATO
- SI DATO = n
- VERDAD Z = n
- Imprimir Z
- *N=n+1*
- *Fin de condición*
- *Fin de lazo*
- *Fin*

CÓDIGO

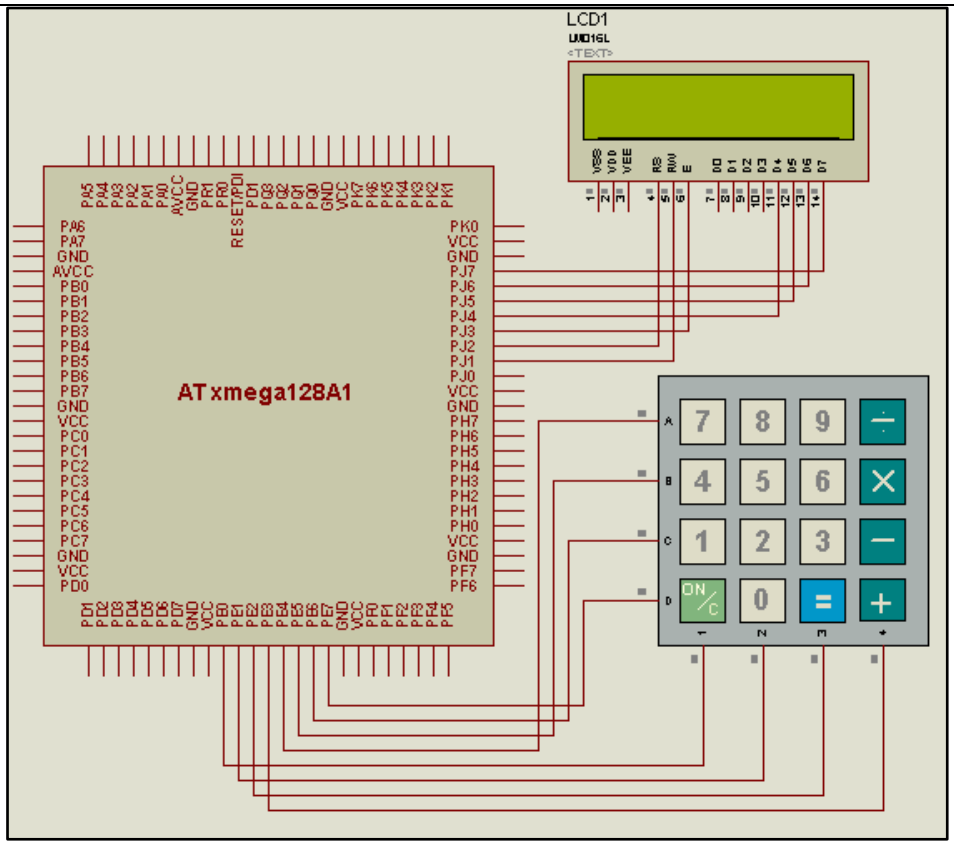
```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

Ddrj = 255                    'CONFIGURACION DE PUERTOS PARA LCD
Portj = 0
Ddre = &HF0                   'CONFIGURACION DE PUERTOS PARA TECLADO
Porte = &HFF                  'ACTIVAR RESISTENCIAS DE PULL-UP
```

```
 Y1 Alias Pine.0
 Y2 Alias Pine.1
 Y3 Alias Pine.2
 Y4 Alias Pine.3

 X1 Alias Porte.4
 X2 Alias Porte.5
 X3 Alias Porte.6
 X4 Alias Porte.7

Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2        'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2             'TIPO DE LCD

Dim Dato As Byte
Dim Tecla As Byte
Dim A As Byte
Dim Z As String * 10            'DEFINIR UNA VARIABLE STRING
                                'DE 10 POSICIONES
Cursor Off
Cls
Home U
Lcd "TECLADO"

Do

  Gosub Teclado                 'SE LLAMA A LA SUBRUTINA TECLADO

 '---SE TOMA EL VALOR DATO DE LA SUBRUTINA---'

  If Dato = 0 Then Z = "0"
  If Dato = 1 Then Z = "1"
  If Dato = 2 Then Z = "2"
  If Dato = 3 Then Z = "3"
  If Dato = 4 Then Z = "4"
  If Dato = 5 Then Z = "5"
```

```
 If Dato = 6 Then Z = "6"
 If Dato = 7 Then Z = "7"
 If Dato = 8 Then Z = "8"
 If Dato = 9 Then Z = "9"
 If Dato = 10 Then Z = "10"
 If Dato = 11 Then Z = "11"
 If Dato = 12 Then Z = "12"
 If Dato = 13 Then Z = "13"
 If Dato = 14 Then Z = "14"
 If Dato = 15 Then Z = "15"

 '---IMPRIMIMOS EL CARACTER CORRESPONDIENTE AL VALOR DE LA SUBRUTINA---'

 Home L
 Lcd Z ; "   "

Loop

'---INICIO SUBRUTINA TECLADO---'

Teclado:
 Tecla = 16

 '----BARRIDO----'

Reset X4
 If Y1 = 0 Then Tecla = 10
 If Y2 = 0 Then Tecla = 0
 If Y3 = 0 Then Tecla = 11
 If Y4 = 0 Then Tecla = 15
Set X4
Reset X3
 If Y1 = 0 Then Tecla = 7
 If Y2 = 0 Then Tecla = 8
 If Y3 = 0 Then Tecla = 9
```

```
 If Y4 = 0 Then Tecla = 14
Set X3
Reset X2
 If Y1 = 0 Then Tecla = 4
 If Y2 = 0 Then Tecla = 5
 If Y3 = 0 Then Tecla = 6
 If Y4 = 0 Then Tecla = 13
Set X2
Reset X1
 If Y1 = 0 Then Tecla = 1
 If Y2 = 0 Then Tecla = 2
 If Y3 = 0 Then Tecla = 3
 If Y4 = 0 Then Tecla = 12
Set X1

'---FIN DE BARRIDO---'

  If Tecla < 16 Then
     Dato = Tecla
     Waitms 300
  End If
Return

'---FIN DE SUBRUTINA---'
```

ESQUEMÁTICO



2) Generar un programa que permita al usuario ingresar una clave si es correcta se muestra en el LCD la palabra “CONTRASEÑA CORRECTA”, caso contrario, se muestra “CONTRASEÑA INCORRECTA”, la contraseña se valida presionando la tecla #, utilizar el diagrama esquemático del ejercicio 1.

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

Ddrj = 255                  'CONFIGURACION DE PUERTOS DEL LCD
Portj = 0                   'TECLADO
Ddre = &HF0                 'CONFIGURACION DE PUERTOS
Porte = &HFF
```

```
X1 Alias Porte.4
X2 Alias Porte.5
X3 Alias Porte.6
X4 Alias Porte.7
Y1 Alias Pine.0
Y2 Alias Pine.1
Y3 Alias Pine.2
Y4 Alias Pine.3

Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2       'CONFIGURAR LAS LINEAS DEL LCD

Dim Dato As Byte
Dim Tecla As Byte
Dim A As Byte
Dim Z As String * 10
Config Lcd = 16 * 2
Cursor Off
Cls
Home U
Lcd "TECLADO"

Dim Num As Byte
Dim Valor As String * 20
Dim Clave As String * 20
Num = 0
Dato = 16
Clave = "1234"

Do

  Gosub Teclado                    'LEER SUBRUTINA
  Gosub Ingreso                    'LEER SUBRUTINA
  Home L
  Lcd Valor ; "   "                 'IMPRIMIR VARIABLE VALOR
```

```
Loop

'---INICIO SUBRUTINA INGRESO---'

Ingreso:

'--- SOLO NUMEROS DE CERO A NUEVE---'

If Dato < 10 Then
   Gosub Asigna
   Incr Num

   If Num = 1 Then          'Z DEPENDE DE QUE TECLA SE HAYA PRESIONADO
    Valor = Z               'EL CARACTER EN Z SE DESPLAZA A LA
                            ‘VARIABLE VALOR, Z=1>>> VALOR =1
   End If

   If Num = 2 Then
    Valor = Valor + Z       'EL CARACTER EN Z SE ANADE A
                           ‘LA VARIABLE VALOR, Z=2 >>> VALOR =12
   End If

   If Num = 3 Then
    Valor = Valor + Z       'EL CARACTER EN Z SE ANADE A
                           ‘LA VARIABLE VALOR, Z=2 >>> VALOR =122
   End If

   If Num = 4 Then
    Valor = Valor + Z       ' EL CARACTER EN Z SE ANADE A
                           ‘LA VARIABLE VALOR, Z=6 >>> VALOR =1226

   Num = 0
   End If
   Dato = 16
  End If
```

```
  '---VALIDAR CONTRASENA---'

  If Dato = 11 Then            'LA CLAVE SE COMPRUEBA CON LA TECLA #
   If Valor = Clave Then       'SI VALOR ES IGUAL A CLAVE
                               ‘(CLAVE DEFINIDA AL INICIO)

     Cls
     Home U : Lcd "CLAVE CORRECTA"
     Else
     Cls
     Home U : Lcd "CLAVE INCORRECTA"
   End If

   Wait 1

   Cls
   Home U
   Lcd "TECLADO"

   '----RESETEAR VARIABLES---'
   Valor = " "
   Num = 0
   Dato = 16
  End If
Return
'-------FIN DE SUBRUTINA-----------'


'------ INICIO SUBRUTINA ASIGNA----'
Asigna:
  If Dato = 0 Then Z = "0"
  If Dato = 1 Then Z = "1"
  If Dato = 2 Then Z = "2"
  If Dato = 3 Then Z = "3"
  If Dato = 4 Then Z = "4"
  If Dato = 5 Then Z = "5"
```

```
  If Dato = 6 Then Z = "6"
  If Dato = 7 Then Z = "7"
  If Dato = 8 Then Z = "8"
  If Dato = 9 Then Z = "9"
  If Dato = 10 Then Z = "10"
  If Dato = 11 Then Z = "11"
  If Dato = 12 Then Z = "12"
  If Dato = 13 Then Z = "13"
  If Dato = 14 Then Z = "14"
  If Dato = 15 Then Z = "15"

Return
'---- Fin De Subrutina----'
              '

'--- INICIO SUBRUTINA TECLADO---'
Teclado:
Tecla = 16
Reset X4
 If Y1 = 0 Then Tecla = 10
 If Y2 = 0 Then Tecla = 0
 If Y3 = 0 Then Tecla = 11
 If Y4 = 0 Then Tecla = 15
Set X4
Reset X3
 If Y1 = 0 Then Tecla = 7
 If Y2 = 0 Then Tecla = 8
 If Y3 = 0 Then Tecla = 9
 If Y4 = 0 Then Tecla = 14
Set X3
Reset X2
 If Y1 = 0 Then Tecla = 4
 If Y2 = 0 Then Tecla = 5
 If Y3 = 0 Then Tecla = 6
 If Y4 = 0 Then Tecla = 13
Set X2
```

```
Reset X1
 If Y1 = 0 Then Tecla = 1
 If Y2 = 0 Then Tecla = 2
 If Y3 = 0 Then Tecla = 3
 If Y4 = 0 Then Tecla = 12
Set X1

  If Tecla < 16 Then
     Dato = Tecla
     Waitms 300
  End If

Return
'---FIN SUBRUTINA---'
```

3) EJERCICIO DE APLICACIÓN: Escribir un programa para realizar las siguientes operaciones: suma, resta, multiplicación, división. Para escoger al operación utilizar la opción case-select.

OPERACIÓN	SÍMBOLO PARA OPERACIÓN	NÚMEROS DE LA OPERACIÓN
Suma	1	2
Resta	2	2
Multiplicación	3	2
División	4	2
Log10	5	1

El LCD debe mostrar el siguiente mensaje: ingrese el número de la operación a realizar

Al ingresar el número 1 por ejemplo:

- En el LCD aparece el mensaje ingrese N1=2
- Luego de ingresar el número aparece el mensaje ingrese el N2=3
- Luego aparece el mensaje 3+2=5

ANEXO 8

ADC / DA

PRÁCTICA 6

- **Tema**

  Conversión análoga digital y digital análoga

- **Objetivo**

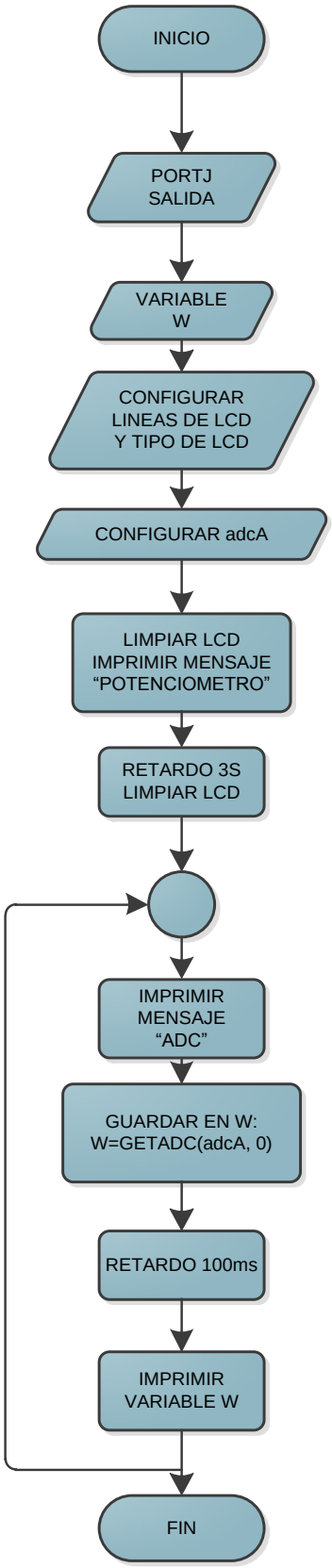
  Utilizar los conversores A/D y D/A del microcontrolador XMega de Atmel.

- **Desarrollo**

1) Conversión A/D con el microcontrolador XMega de Atmel.

Se adquiere la señal de un potenciómetro entre 0 y 5 voltios, y se observa en el LCD un valor entre 0 y 4095

Diagrama de flujo para la conversión a/d



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *PortJ Salida*

- *Variable tipo word W*
- *Configurar líneas de LCD*
- *Configurar tipo de LCD*
- *Configurar adcA:* se configura ADC

  - **CONFIG ADCA | ADCB** poner registros configurados

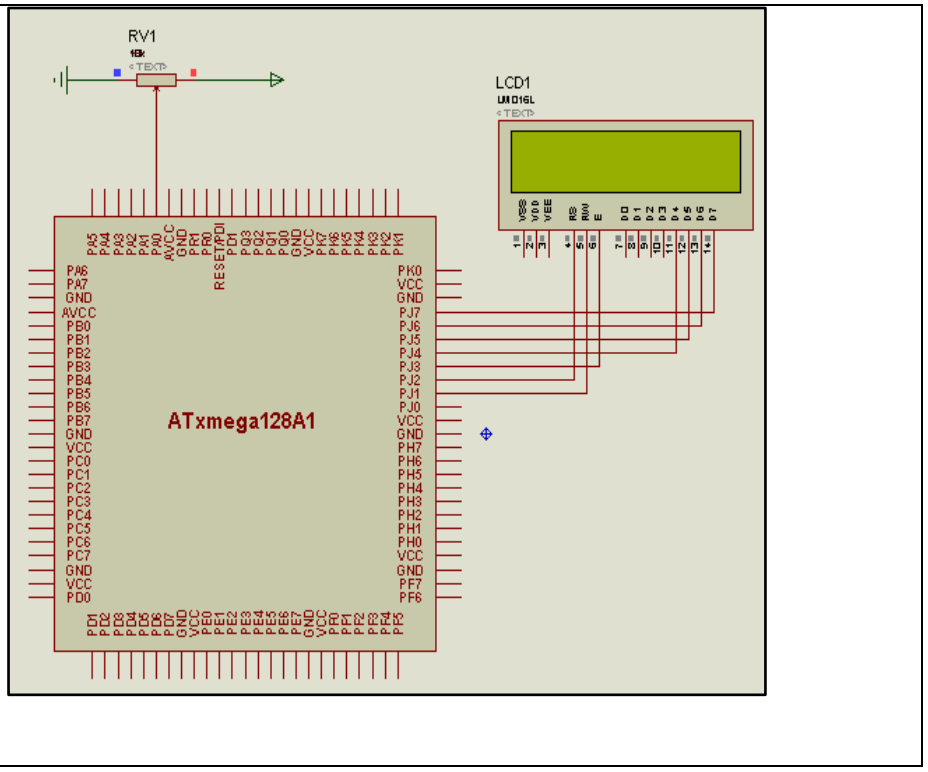
- *Limpiar pantalla*
- *Imprimir mensaje "POTENCIOMETRO"*
- Retardo de 3 segundos
- *Inicio lazo*
- *Imprimir mensaje "ADC"*
- *Adquirir W= GETAD(adcA,0)*
- *Retardo de 100 ms*
- Imprimir variable W
- *Fin de lazo*
- *Fin*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "XMEGA.LIB"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Ddrj = 255                    'CONFIGURACION PUERTO COMO SALIDA
Portj = 0

Dim W As Word
```

```
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2       'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2              'CONFIGURAR TIPO DE LCD

'---CONFIGURACION DEL ADC-A ---'
Config Adca = Single , Convmode = Unsigned , Resolution = 12bit , Dma = Off ,
Reference = Int1v , Event_mode = None , Prescaler = 32 , Ch0_gain = 1 , Ch0_inp =
Single_ended , Mux0 = 0

Cls                              'LIMPIAR LA PANTALLA DEL LCD
Home U                           'IMPRIMIR UN MENSAJE EN LA FILA L
Lcd "POTENCIOMETRO"
Wait 3
Cls

Do
  Locate 1 , 5                   'IMPRIMIR UN MENSAJE EN FILA 1, COLUMNA 5
  Lcd "ADC"
  W = Getadc(adca , 0)      'GUARDAR EN LA VARIABLE W LA LECTURA
                                 'DEL ADC DEL PUERTO A CON OFFSET DE CERO
  Waitms 100
  Locate 2 , 1
  Lcd W ; "    "                 'IMPRIMIR VARIABLE W
Loop
```

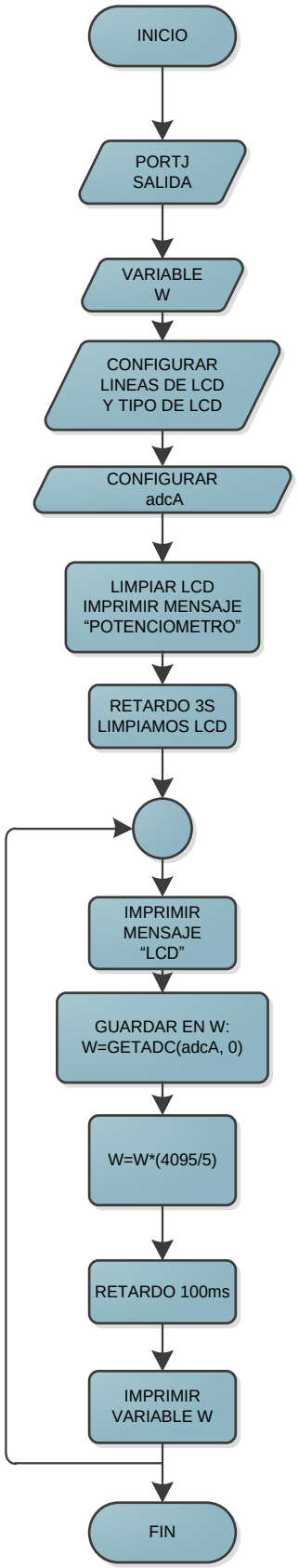
ESQUEMÁTICO



2) Conversión A/D con el microcontrolador XMega de Atmel. Utilizar el diagrama esquemático del ejercicio 1.

Se adquiere la señal de un potenciómetro entre 0 y 5 voltios, y se observa en el LCD un valor entre 0 y 5 voltios.

Diagrama de flujo para la conversión a/d usando la señal de un potenciómetro



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *PortJ Salida*
- *Variable tipo word W*
- *Configurar líneas de LCD*
- Configurar ADC
- *Limpiar la pantalla*
- *Imprimir mensaje "POTENCIOMETRO"*
- *Retardo 3s*
- *Inicio lazo*
- *Imprimir mensaje "ADC"*
- *W=W(4095/5)*
- *Retardo de 100 ms*
- Imprimir variable W
- *Fin de lazo*
- *Fin de programa*

CÓDIGO

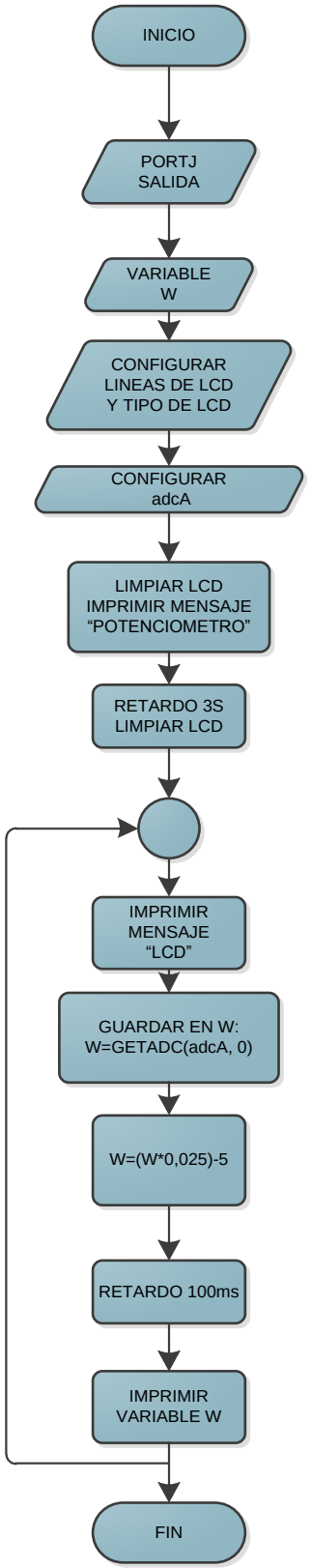
```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "XMEGA.LIB"                    'LIB PARA REALIZAR ADQUISICION DE
ADC
$external _xmegafix_clear           'LIB PARA CONVERTIR DE WORD A STRING
$external _xmegafix_rol_r1014       'LIB PARA CONVERSIONES WORD A STRING

Ddrj = 255                          'CONFIGURACION PUERTO COMO SALIDA
Portj = 0

Dim W As Word
Dim A As Single
```

```
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2            'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2                'CONFIGURAR TIPO DE LCD



'---CONFIGURACION DEL ADC-A---'
Config Adca = Single , Convmode = Unsigned , Resolution = 12bit , Dma = Off ,
Reference = Int1v , Event_mode = None , Prescaler = 32 , Ch0_gain = 1 , Ch0_inp =
Single_ended , Mux0 = 0       'you can setup other channels as well

Cls                              'LIMPIAR LA PANTALLA DEL LCD
Home U                           'IMPRIMIR UN MENSAJE EN LA FILA L
Lcd "POTENCIOMETRO"
Wait 3
Cls

Do
  Locate 1 , 5                  'IMPRIMIR UN MSN EN FILA 1, COLUMNA 5
  Lcd "ADC"
  W = Getadc(adca , 0)          'GUARDAR EN LA VARIABLE W LA LECTURA
                                'DEL ADC DEL PUERTO A CON OFFSET DE CERO

  '---CONVERSION---'
  A = W
  A = A / 4095
  A = A * 5
  W = A
  Waitms 100
  Locate 2 , 1
  Lcd W ; " V   "
Loop
```

3) Medición de temperatura con lm35, el LM35 es un sensor analógico que devuelve la temperatura en forma de tensión, esta tensión devuelta es

proporcional a la temperatura. Su rango comprende desde -55º hasta 150ºC y el valor devuelto es el equivalente a la temperatura dividida por 10, entonces en su salida se obtienen valores como estos

Diagrama de flujo para la conversión a/d usando un lm35



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *PortJ Salida*

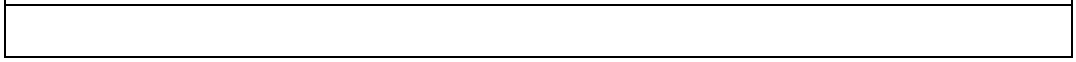
- *Variable tipo word W*
- *Configurar líneas de LCD*
- Configurar ADC
- *Limpiar pantalla*
- *Imprimir mensaje "TEMPERATURA"*
- *Retardo 3s*
- *Inicio lazo*
- *Imprimir mensaje "LCD"*
- *W= GETAD(adcA,0)*
- *W=(W*0.025)-5*
- *Retardo de 100 ms*
- Imprimir variable W
- *Fin de lazo*
- *Fin de programa*

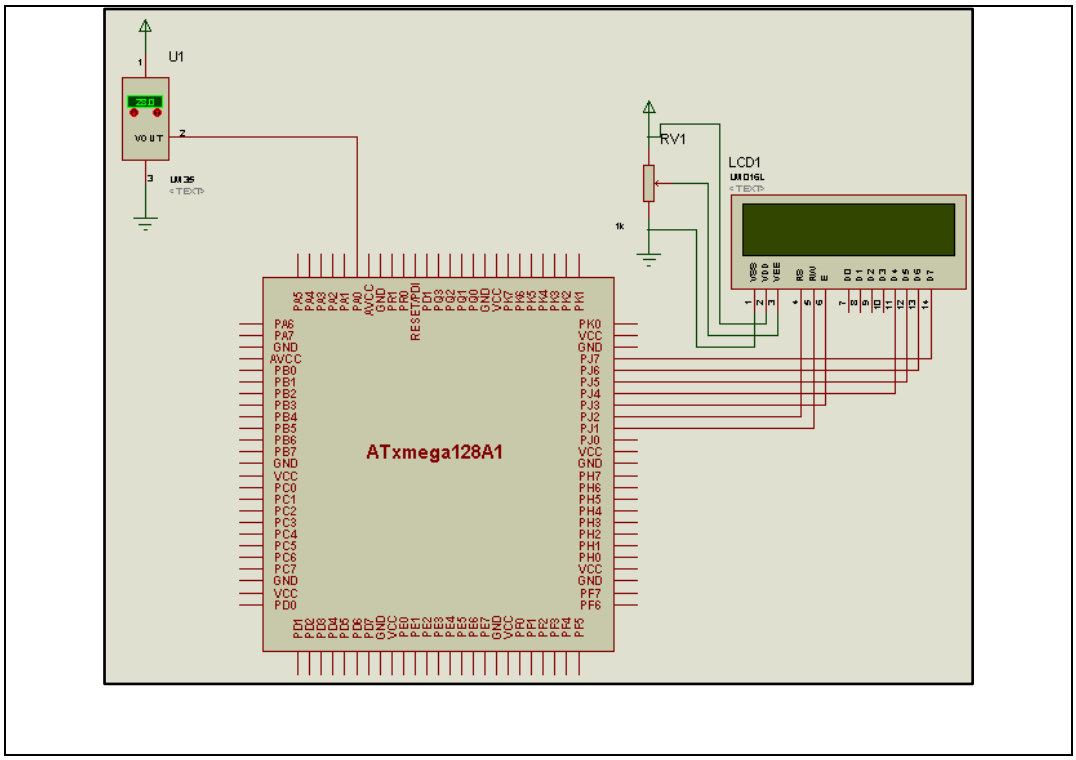
CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "XMEGA.LIB"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Ddrj = 255                    'CONFIGURACION PUERTO COMO SALIDA
Portj = 0

Dim W As Word
Dim A As Single



Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
```

```
Portj.3 , Rs = Portj.2          'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2             'CONFIGURAR TIPO DE LCD



'---CONFIGURACION DEL ADC-A---'

Config Adca = Single , Convmode = Unsigned , Resolution = 12bit , Dma = Off ,
Reference = Int1v , Event_mode = None , Prescaler = 32 , Ch0_gain = 1 , Ch0_inp =
Single_ended , Mux0 = 0       'you can setup other channels as well

Cls                                     'LIMPIAR LA PANTALLA DEL LCD
Home U                                  'IMPRIMIR UN MENSAJE
Lcd "LM35 SIGNAL"
Wait 3
Cls

Do
  Locate 1 , 5                  'IMPRIMIR UN MSN EN FILA 1, COLUMNA 5
  Lcd "ADC"
  W = Getadc(adca , 0)          'GUARDAR EN LA VARIABLE W LA LECTURA DEL
                                'ADC DEL PUERTO A CON OFFSET DE CERO
  '----ESCALAMIENTO----'
  A = W
  A = A * 0.025
  A = A - 5
  W = A

  Waitms 100
  Locate 2 , 1
  Lcd W ; "C      "
Loop
End
```

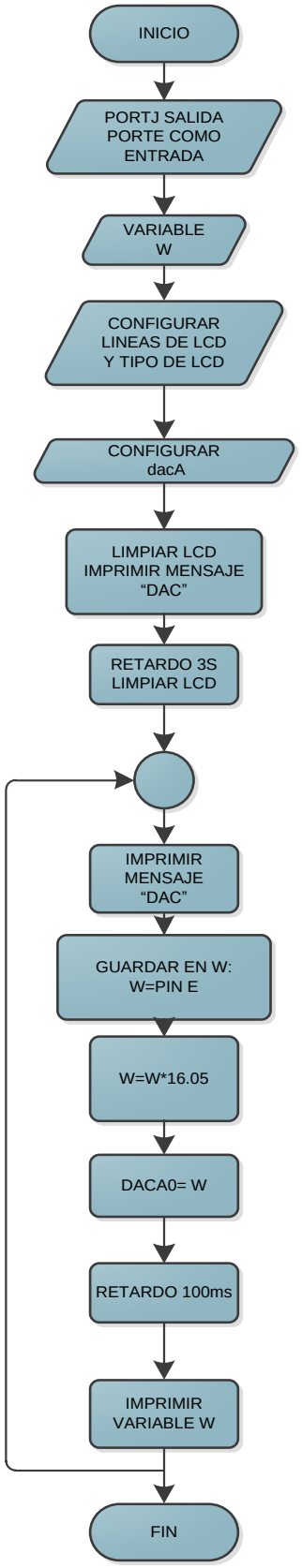
ESQUEMÁTICO





4) Conversión Digital Análoga de 12 bits

Diagrama de flujo para la conversión d/a de 12 bits



Elaborado por: Francisco Reyes y Nina Chicaiza

-
- *Inicio*

- *Configurar PortJ como salida*
- *PortE como entrada*
- *Variable tipo word W*
- *Configurar líneas de LCD*
- Configurar DAC
  - CONFIG DACx escribir los registros

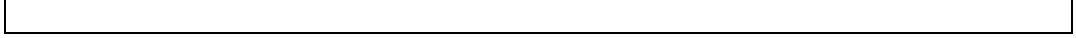
- *Limpiar pantalla*
- *Imprimir mensaje "DAC"*
- *W = PIN E*
- *W = W*16,05*
- *DAC0 = W*
- *Retardo 100ms*
- Imprimir variable W
- *Fin de lazo*
- *Fin de programa*

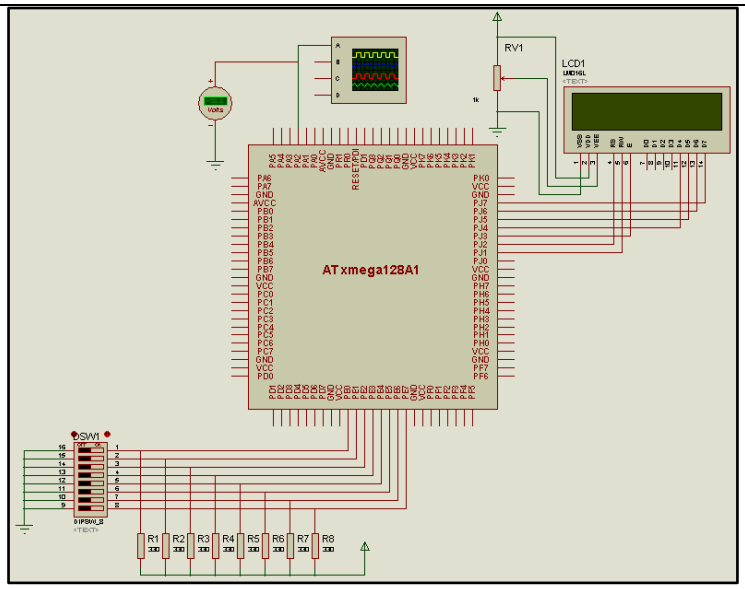
CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "XMEGA.LIB"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Ddre = 0                    'PUERTO DE ENTRADA, PARA DIPSWITCH
Porte = &HFF                'ACTIVAR RESISTENCIAS DE PULL UP

Ddrj = 255                  'PUERTO DE SALIDA
Portj = 0                   'LIMPIAR PUERTO
```

```
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2            'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2                  'CONFIGURAR TIPO DE LCD

'---CONFIGURACION DEL DAC, PUERTO A PIN2---'
Config Daca = Enabled , Io0 = Enabled , Channel = Single , Reference = Int1v ,
Interval = 64 , Refresh = 64

Dim W As Word
Dim A As Single

Cls                                  'LIMPIAR LA PANTALLA DEL LCD
Locate 1 , 5                         'IMPRIMIR UN MSN EN FILA 1, COLUMNA 5
Lcd "DAC"
Home L                               'IMPRIMIR UN MENSAJE EN LA FILA L
Lcd "MEDIR VOLTAJE EN PUERTO A"
Cls
Do

  W = Pine                           'LEER EL DIPSWITCH
  A = W

  '---ESCALAMIENTO---'
  A = A * 16.05
  W = A
  Daca0 = W            'CONVERSION DE BITS A VALORES DE VOLTAJE

Locate 1 , 1 : Lcd Bin(pine)
Locate 2 , 1 : Lcd W ; "   "
Loop
```

ESQUEMÁTICO





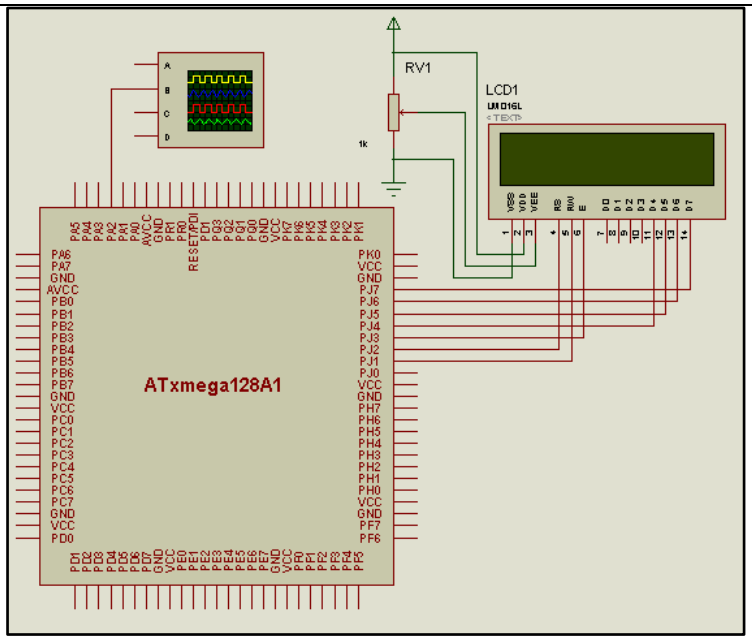
5) Generar una onda diente de sierra con el Conversor Digital- Análogo de 12 bits

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "XMEGA.LIB"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Ddrj = 255                    'CONFIGURACION PUERTO COMO SALIDA
Portj = 0

Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2       'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2

'---CONFIGURACION DEL DAC, PUERTO A PIN2---'
```

```
Config Daca = Enabled , Io0 = Enabled , Channel = Single , Reference = Int1v , Interval = 64 , Refresh = 64

Cls

Dim A As Word
Dim I As Word

A = 0

 Home U
 Lcd "DIENTE DE SIERRA"

Do
For I = 0 To 4095
    Incr A
    Waitus 10
    Daca0 = A
Next
I = 0
Loop
```

ESQUEMÁTICO



6) Generar una onda triangular con el Conversor Digital- Análogo de 12 bits, usar el diagrama esquemático del ejercicio 4.

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "XMEGA.LIB"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Ddrj = 255                    'CONFIGURACION PUERTO COMO SALIDA
Portj = 0

Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2      'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2
```

```
'----CONFIGURACION DEL DAC, PUERTO A PIN2----'
Config Daca = Enabled , Io0 = Enabled , Channel = Single , Reference = Int1v , Interval = 64 , Refresh = 64

Cls

Dim A As Word
Dim I As Word

A = 0

 Home U
 Lcd "ONDA TRIANGULAR "

Do
For I = 0 To 4095
    Incr A
    Waitus 10
    Daca = A
Next
I = 0

For I = 0 To 4095
    Decr A
    Waitus 10
    Daca = A
Next
I = 0
A = 0
Loop
```

ANEXO 9

INTERRUPCIONES

PRÁCTICA 7

- **Tema**

Interrupciones

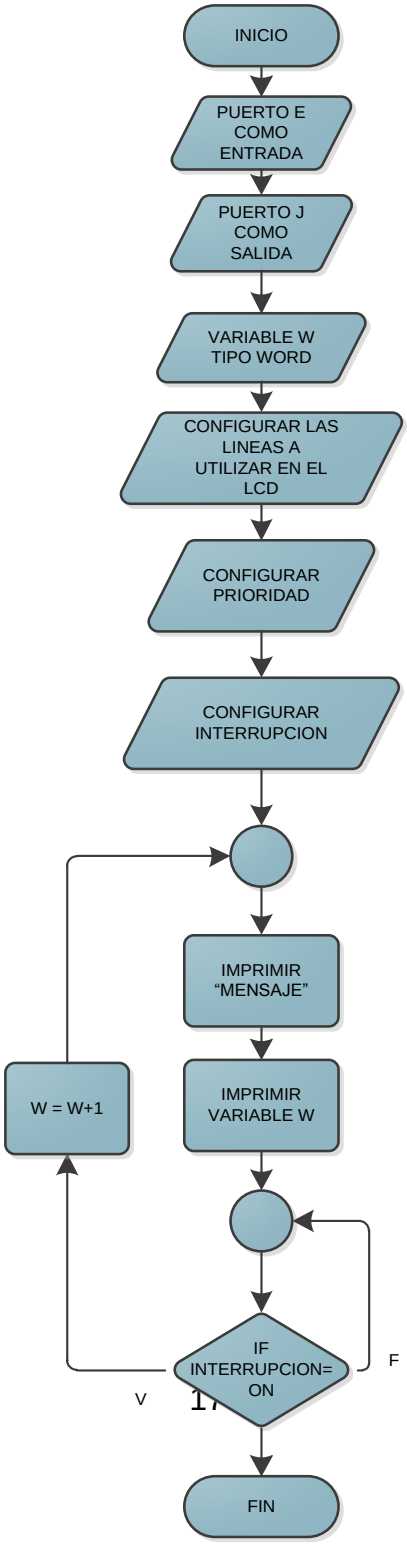
- **Objetivo**

Elaborar programas para comprender el uso de interrupciones.

- **Desarrollo**

1) Interrupción temporizadores, uso de timer0

Diagrama de flujo para uso de interrupciones



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *PortE como entrada*
- *Puerto J como salida*
- *Variable W tipo Word*
- *Configurar las líneas a utilizar en el LDC*
- Configurar la prioridad (PRIORITY)

- CONFIG PRIORITY ver en programación como esta
- *Configurar la interrupción*
  - ENABLE interrupt [, prio]
- *Asignar un nombre para la interrupción*
- *Encender la interrupción*
  - On Porte_int0 Port_e_int0_interrupcionext
  - Nombre: Port_e_int0_interrupcionext
- *Iniciar lazo*
- *Imprimir "Mensaje"*
- *Imprimir variable W*
- *Fin de lazo*
- *Fin de programa*
- W = W+1Return
- *Fin de interrupción*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000                  '2MHz
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

'---CONFIGURACION DE PRIORIDAD DE INTERRUPCION---'
Config Priority = Static , Vector = Application , Lo = Enabled
'PRIORIDAD BAJA

'---CONFIGURACION DE TIMER C0---'
Config Tcc0 = Normal , Prescale = 64
Tcc0_per = 31250
'CRISTAL = 2MHz, PREESCALER =64, 2MHz/64 = 31250
' --> OVERFLOW CADA SEGUNDO
```

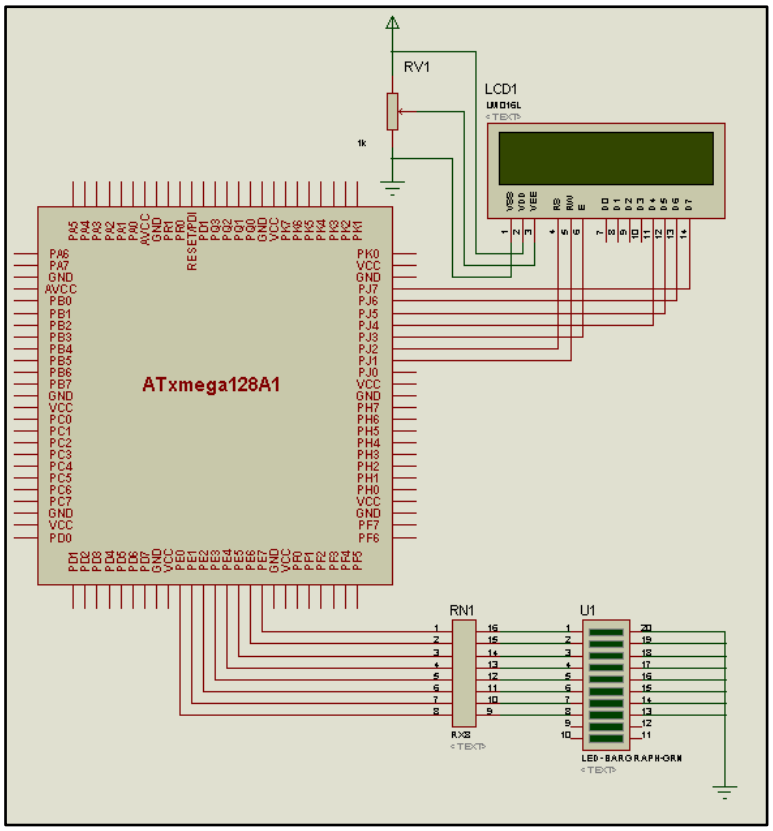
```
On Tcc0_ovf Tc0_interrupcion
'NOMBRAR A LA INTERRUPCION COMO Tc0_interrupcion

Enable Tcc0_ovf , Lo
'BAJA PRIORIDAD PARA EL OVERFLOW

Enable Interrupts
'ACTIVAR INTERRUPCIONES
'---------------------------------'
Dim W As Byte                  'VARIABLE A INCREMENTAR
Config Porte = Output         'PUERTO QUE SE ENCENDERÁ Y APAGARÁ
                               ‘CADA VEZ QUE HAYA UN OVERFLOW
                               ‘Y LA INTERRUPCIÓN SEA EJECUTADA
'---LCD CONFIG---'
Config Portj = Output
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2       'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2
Cursor Off
Cls

'---[LAZO PRINCIPAL]---‘
Do
  Home U
  Lcd "OVERFLOW TMR0"
  Home L
  Lcd W ; "   "                 'IMPRIMIR LA VARIABLE QUE SE
                               ‘INCREMENTA EN CADA INTERRUPCIÓN
Loop
'---[LAZO PRINCIPAL]---‘
End                            'FIN DEL PROGRAMA PRINCIPAL

'---[RUTINAS DE INTERRUPCIÓN]---‘
Tc0_interrupcion:
Tcc0_per = 31250
```

Incr W

Porte = Not Porte

Return

ESQUEMÁTICO



2) Interrupción externa.

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "XMEGA.LIB"
$external _xmegafix_clear
$external _xmegafix_rol_r1014
```

```
Ddre = 0              'PUERTO DE ENTRADA PARA ACTIVAR INTERRUPCIONES
Porte = &HFF

Ddrj = 255            'PUERTO DE SALIDA PARA CONTROLAR LCD
Portj = 0

Dim W As Word
Dim A As Single

'---CONFIGURACIÓN PARA HABILITAR INTERRUPCIONES EXTERNAS---'

Config Priority = Static , Vector = Application , Lo = Enabled
Enable Porte_int0 , Lo
'ACTIVAR LA INTERRUPCIÓN COMO DE BAJO NIVEL

On Porte_int0 Port_e_int0_interrupcionext  'NOMBRAR A LA INTERRUPCIÓN
                                           ‘COMO Port_e_int0_interrupcionext

Enable Interrupts

Porte_pin0ctrl = &B00_011_010              'PULL UP
Porte_int0mask = &B0000_0001               'MASCARA DE PIN0 E INT 0 DEF INT

'---------LCD CONFIG-------'
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2      'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2
Cursor Off
Cls

Do
  Home U
  Lcd "INTERRUPCION EXT"
  Locate 2 , 1
  Lcd W ; "   "
```

Loop

'---RUTINA DE INTERRUPCIÓN---'

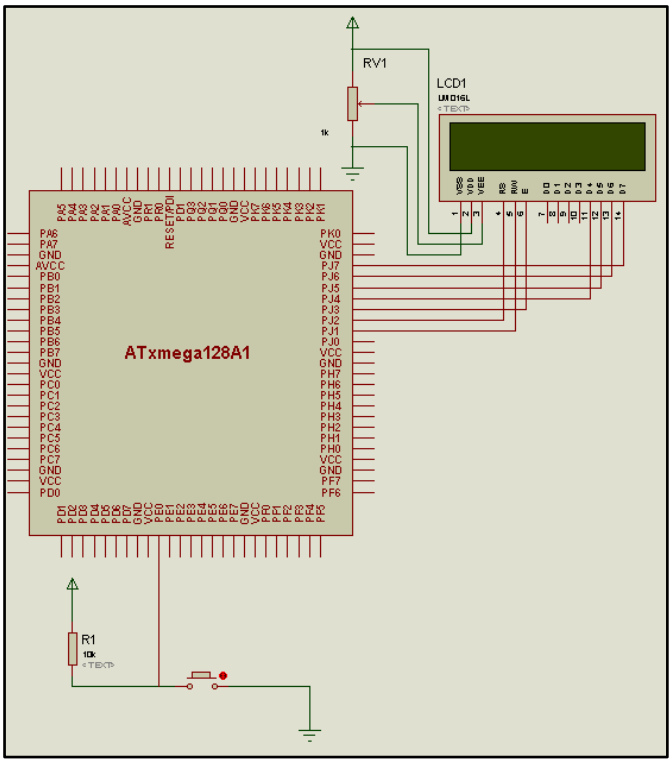
Port_e_int0_interrupcionext:

Incr W

Waitms 100

Return

ESQUEMÁTICO



3) Ejercicio de aplicación: Mediante el uso de la interrupción del timer realizar un generador de onda cuadrada de periodo de 1 ms.

ANEXO 10

MEMORIAS

PRÁCTICA 8

- **Tema**

  Memorias

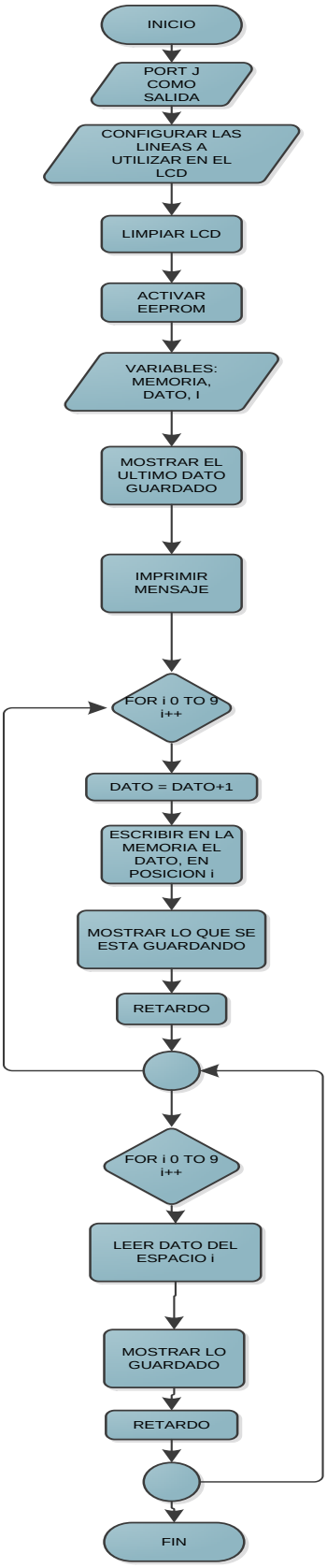
- **Objetivo**

  Manejar memorias internas y externas con el microcontrolador XMega de Atmel.

- **Desarrollo**

2) Escribir una secuencia de números a los sucesivos lugares de EEPROM.

Diagrama de flujo para uso de la memoria eeprom



Elaborado por: Francisco Reyes y Nina Chicaiza

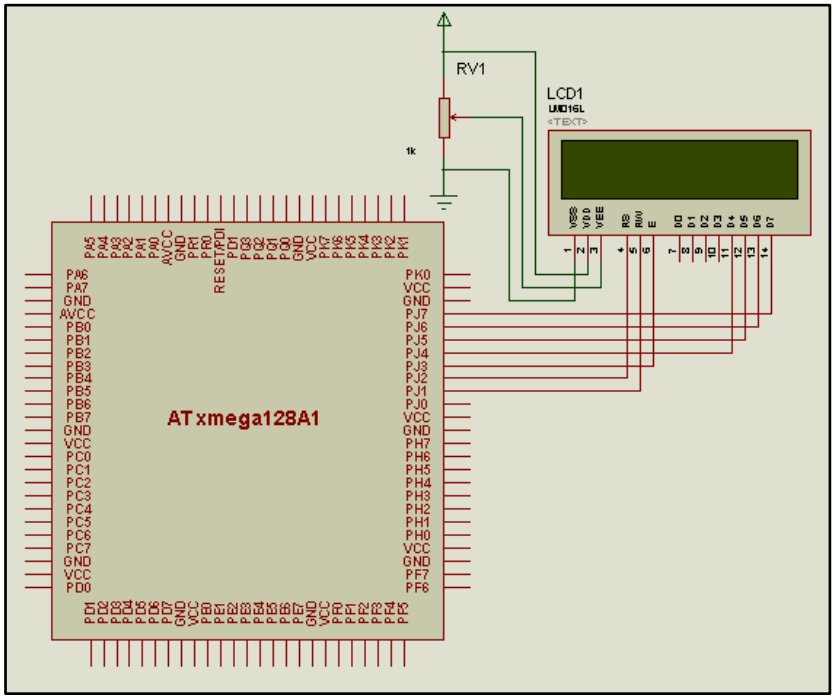
- *Inicio*

- *Puerto J como salida*
- *Configurar líneas de salida*
- Activar la memoria EEPROM:
    - Config Eeprom = Mapped
- *Variables Memoria, Dato, i* tipo ERAM BYTE
- *Mostrar el último dato guardado:* Leer el último dato guardado
- *For i 0 TO 9*
- *Dato= Dato+1*
- *Escribir en la memoria el dato en la posición i*
- *Mostrar lo que se está guardando*
- *For i 0 TO 9*
- *Leer dato del espacio i*
- Imprimir lo que se está leyendo
- *Fin de programa*

CÓDIGO

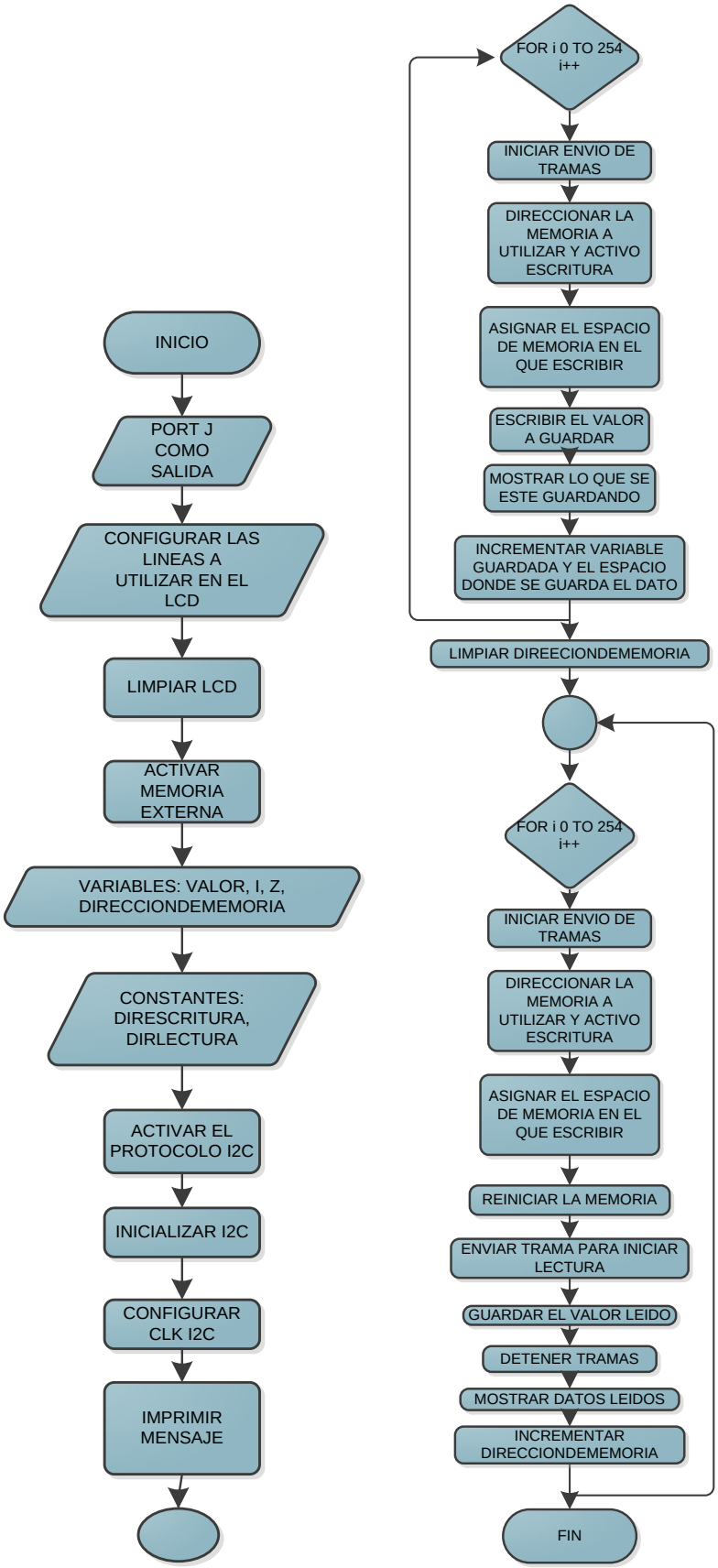
```
$regfile = "xm128a1def.dat"
$crystal = 2000000                          '2MHz
$hwstack = 64
$swstack = 40
$framesize = 40
$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Config Portj = Output
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2        'CONFIGURO LAS LINEAS DEL LCD
Config Lcd = 16 * 2
Cursor Off
Cls

Config Eeprom = Mapped        'CONFIG DE MEMORIA EEPROM
Dim Memoria As Eram Byte
Dim Dato As Byte
```

```
Dim I As Byte
Dim A As Byte

Dato = Memoria                    'LEEMOS EL VALOR
                                  ‘GUARDADO E INICIAMOS DESDE AHI
Home U
Lcd "ULTIMODATO GUARDADO"
Readeeprom Dato , A
Wait 2
Cls
'escribiendo 10 datos
  Home U
  Lcd "GUARDANDO"
  For I = 1 To 10
    Incr Dato
    Writeeeprom Dato , I
    Home L
    Lcd Dato ; "  "
    Wait 3
  Next
  A = I
  I = 0
  Cls
'leer los 10 datos
  Home U
  Lcd "MOSTRANDO"
  For I = 1 To 10
    Readeeprom Dato , I
    Home L
    Lcd Dato ; "  "
    Wait 3
  Next

  Memoria = Dato
  Wait 1
```

Do Loop
ESQUEMÁTICO


3) Escritura y lectura en una memoria I2C (lectura y escritura de 1 dato)

Diagrama de flujo para uso de la memoria i2c



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*

- *Puerto J como salida*
- *Configurar líneas de control de lcd*
- *Limpiar la pantalla*
- *Activar memoria externa*
- *Variables:* Valor, I, Z, DIRECCIONDEMEMORIA
- *Establecer constantes*
- *Activar el protocolo I2C*
- *Inicializar I2C*
- Configurar el reloj
- *For i 0 TO 254*
- *Iniciar envió de tramas*
- *Imprimir variable guardada*
- *Incrementar la variable que guarda el dato*
- *Fin de lazo*
- *Limpiar DIRECCIONDEMEMORIA*
- *For i 0 TO 254*
- *Imprimir lo guardado*
- Incrementar el valor de la variable DIRECCIONDEMEMORIA
- *Fin de lazo*
- *Fin de programa*

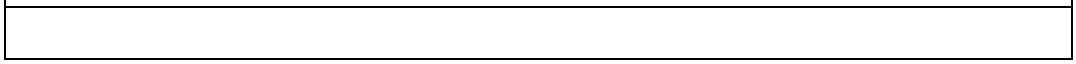
CÓDIGO

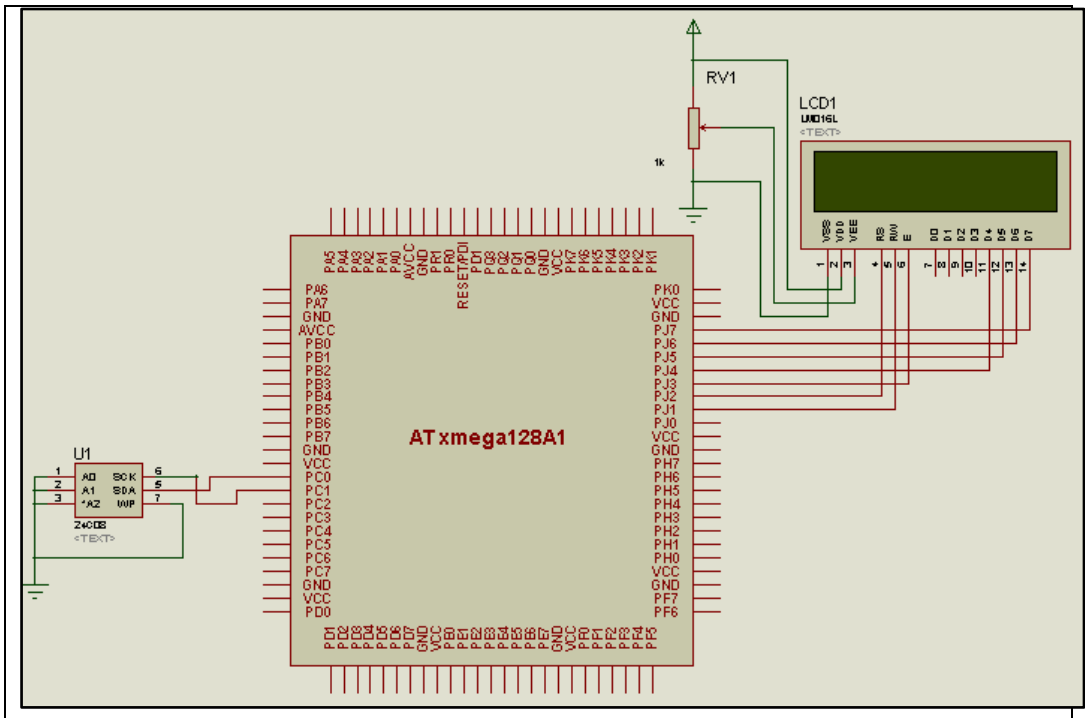
```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Ddrj = 255                    'PARA LCD
Portj = 0
```

```
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2        'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2             'CONFIGURAR TIPO DE LCD



Dim Twi_start As Byte           'VARIABLE UTILIZADA PARA LAS FUNCIONES
I2C

Dim Valor As Byte
Dim Direccionmemoria As Byte

Const Direscritura = &B10100000
'1010>>> del chip; 000>>> A0,A1,A2; 0 ESCRIBIR

Const Dirlectura = &B10100001
'1010>>> del chip; 000>>> A0,A1,A2; 1 LEER

Valor = 254                     'UN SOLO DATO A GUARDAR
Direccionmemoria = 2            'DIRECCION DONDE SE GUARDA O SE LEE

Open "twic" For Binary As #4
'Portc.0 'SDA PIN DEL PUERTO C (ATXMEGA128A1)
'Portc.1 'SCL PIN DEL PUERTO C (ATXMEGA128A1)

I2cinit #4                      'CONFIGURAR I2C ESTADO ADECUADO
                                ‘COLECTOR ABIERTO, PULL UP ACTIVADO
Config Twic = 100000            '100KHZ - CONFIGURAR TWI BAUD RATE Y
                                ‘ACTIVAR TWI EN MODO MAESTRO
Cls
Cursor Off
Home U
Lcd "  MEMORIA I2C "
'---ESCRIBIENDO EN LA MEMORIA---'
I2cstart #4
I2cwbyte Direscritura , #4      '1er BYTE, CONSTA TODA LA INFO DE
                                ‘LA MEMORIA Y COMANDO PARA ESCRIBIR
```

```
I2cwbyte Direccionmemoria , #4     '2do BYTE,  SEASIGNA EL ESPACIO DE
                                   ‘MEMORIA EN EL QUE ESCRIBIREMOS
  I2cwbyte Valor , #4              '3er BYTE, ESCRIBIMOS
  I2cstop #4
  Home L
  Lcd "WR EN REG: " ; Direccionmemoria ; "  "

Wait 2

'---LEYENDO DE LA MEMORIA---'

 I2cstart #4
 I2cwbyte Direscritura , #4          '1er BYTE, CONSTA TODA LA INFO DE LA
                                    ‘MEMORIA Y CMD PARA ESCRIBIR

 I2cwbyte Direccionmemoria , #4      '2do BYTE, A QUE ESPACIO DE MEMORIA
                                    ‘QUEREMOS ACCEDER
 I2crepstart #4
 I2cwbyte Dirlectura , #4        '3er BYTE, TODA LA INFO DE LA MEMORIA
                                   ‘Y CMD PARA REALIZAR UNA LECTURA
 I2crbyte Valor , Nack , #4          '4to BYTE, RETORNA CON EL
                                    ‘DATO GUARDADO
 I2cstop
 Home L
 Lcd "REG: " ; Direccionmemoria ; " " ; "Val: " ; Valor ; "    "
Do
Loop
```

ESQUEMÁTICO





4) Escritura y lectura en una memoria I2C (lectura y escritura de varios datos), utilizar el diagrama esquemático del ejercicio 2.

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014



Ddrj = 255                    'PARA LCD
Portj = 0
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2        'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2           'CONFIGURAR TIPO DE LCD
```

```
Dim Twi_start As Byte            'VARIABLE UTILIZADA PARA LAS FUNCIONES
I2C

Dim Valor As Byte
Dim I As Byte
Dim Z As String * 5
Dim Direccionmemoria As Byte

Const Direscritura = &B10100000
'1010>>> del chip; 000>>> A0,A1,A2; 0 ESCRIBIR

Const Dirlectura = &B10100001
'1010>>> del chip; 000>>> A0,A1,A2; 1 LEER

Valor = 0                        'UN SOLO DATO A GUARDAR
Direccionmemoria = 0             'DIRECCION DONDE SE GUARDA O SE LEE

Open "twic" For Binary As #4
  'Portc.0 'SDA PIN DEL PUERTO C (ATXMEGA128A1)
  'Portc.1 'SCL PIN DEL PUERTO C (ATXMEGA128A1)

I2cinit #4                       'CONFIGURAR I2C ESTADO ADECUADO
                                 ‘COLECTOR ABIERTO ,PULL UP ACTIVADO

Config Twic = 100000             '100KHZ - CONFIGURAR TWI BAUD RATE Y
                                 ‘ACTIVAR TWI EN MODO MAESTRO
Cls
Cursor Off
Home U
Lcd " I2C "

'---ESCRIBIENDO EN LA MEMORIA---'

For I = 0 To 254
```

```
  I2cstart #4
  I2cwbyte Direscritura , #4                '1er BYTE, CONSTA TODA LA INFO DE
                                            ‘LA MEMORIA Y CMD PARA ESCRIBIR

  I2cwbyte Direccionmemoria , #4            '2do BYTE, ASIGNAMOS EL ESPACIO
                                            ‘DE MEMORIA EN EL QUE SE ESCRIBE

  I2cwbyte Valor , #4                       '3er BYTE, ESCRIBIMOS
  I2cstop #4
  Gosub Transascii
  Locate 1 , 8
  Lcd "ASCII CODE"
  Home L
  Lcd "WR EN REG: " ; Direccionmemoria ; "  "
  Waitms 100
  Incr Valor
  Incr Direccionmemoria
Next

Wait 2
I = 0
Direccionmemoria = 0
Cls
Home U
Lcd " I2C "

'---LEYENDO DE MEMORIA---'

For I = 0 To 254
  I2cstart #4
  I2cwbyte Direscritura , #4              '1er BYTE, CONSTA TODA LA INFO DE LA
                                          ‘MEMORIA Y CMD PARA ESCRIBIR

  I2cwbyte Direccionmemoria , #4          '2do BYTE, A QUE ESPACIO DE MEMORIA
                                          ‘QUEREMOS ACCEDER
  I2crepstart #4
```

```
  I2cwbyte Dirlectura , #4              '3er BYTE, TODA LA INFO DE LA MEMORIA
                                       ‘Y CMD PARA REALIZAR UNA LECTURA

  I2crbyte Valor , Nack , #4           '4to BYTE, RETORNA CON EL
                                       ‘DATO GUARDADO
  I2cstop
  Gosub Transascii
  Home L
  Lcd "REG: " ; Direccionmemoria ; " " ; "Val: " ; Z ; "    "
  Waitms 200
  Incr Direccionmemoria
Next

Do
Loop
Transascii:
Z = Chr(valor)
Return
```

4) EJERCICIO DE APLICACIÓN: Utilizando una memoria EEPROM I2C de mayor capacidad guardar las letras del alfabeto en la memoria EEPROM I2C, y luego muéstrela en una matriz de Leds 8 x5.

ANEXO 11

PROGRAMACIÓN EN LABVIEW PARA COMUNICACIÓN RS-232/USB/XBEE

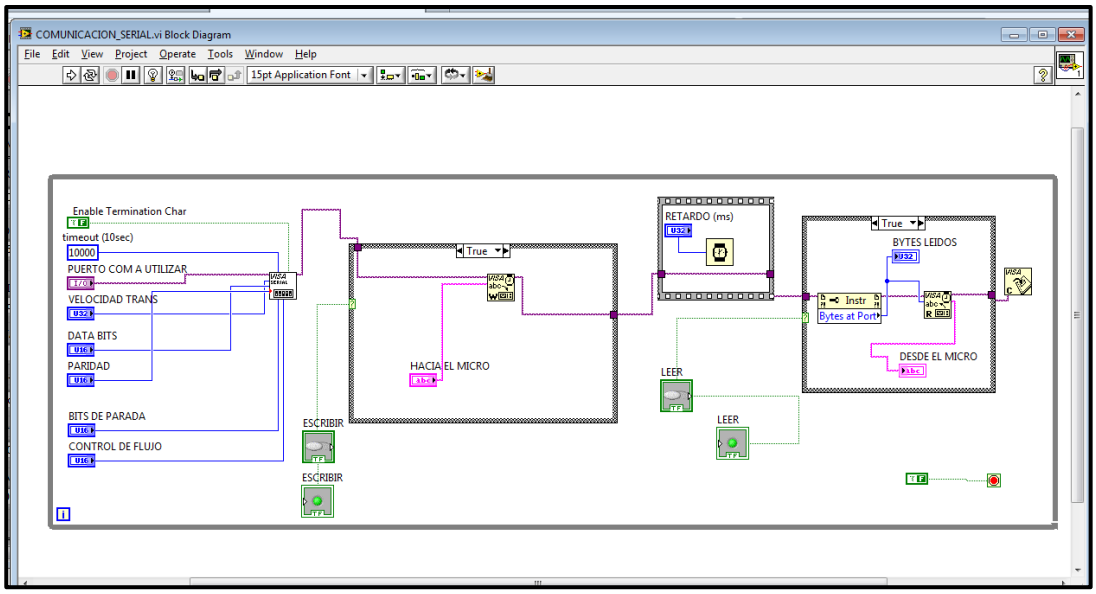
PROGRAMACIÓN EN LABVIEW PARA COMUNICACIÓN SERIAL.

I. Pantalla frontal

Figura 1. Pantalla ejecutable de labview

Fuente: (National Instruments, 2010)

Figura 2. Pantalla de programación de labview



Fuente: (National Instruments, 2010)

II. Detalle:

Los parámetros que se configurara son:

1. Puerto COM a utilizar, si no se cuenta con un puerto físico, se puede utilizar un adaptador, siendo cuidadosos con el COM virtual que se active, para saber que COM utilizar debemos realizar lo siguiente:

   - Ir a PANEL DE CONTROL
   - Buscar el ADMINISTRADOR DE DISPOSITIVOS
   - Se identifica el COM que se genera al momento de conectar el adaptador USB-RS232 o USB

2. La velocidad de transmisión depende de la configuración que se le haya dado al microcontrolador.
3. La paridad en la transmisión es un parámetro que se define también al configurar el microcontrolador
4. Los bits que se enviaran en cada trama deben ser los mismos en la configuración del microcontrolador
5. No poseemos control de flujo en la configuración de parámetros del microcontrolador

6. Bits de parada se asigna al configurar el puerto COM en el microcontrolador, deberá tener el mismo valor
7. Retardo, es el tiempo en el que se recibirá un dato, para configurar este valor se tiene que tomar en cuenta cuanto el tiempo que se demore el microcontrolador.

III. Funcionamiento:

Una vez conectado el microcontrolador, con la programación que se adjunta en los anexos, se inicia una transferencia de datos desde el microcontrolador, solo debemos activar la lectura en el panel de labview y se visualizará un contador que incrementa su valor en uno más un mensaje.

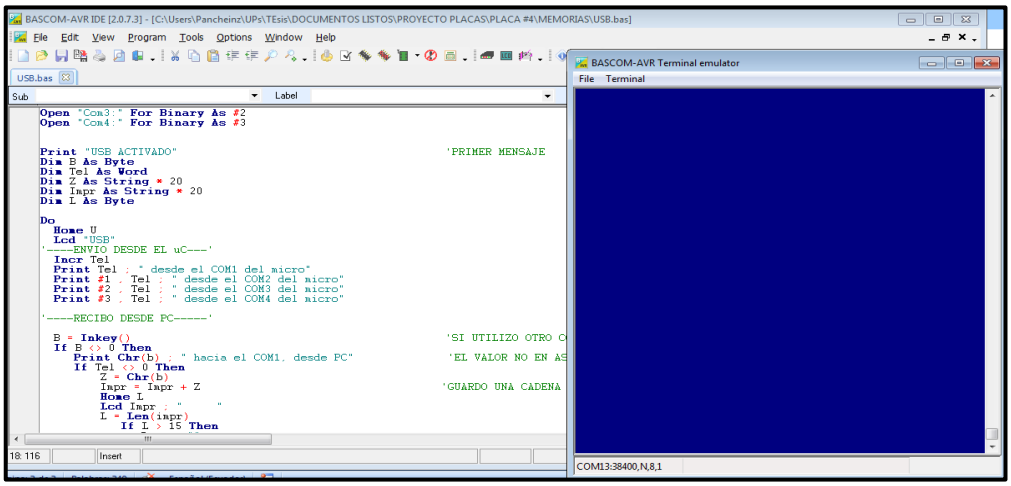
Para enviar una trama hacia el microcontrolador se debe activar el panel de escritura en labview y escribir un mensaje en la pantalla, cada vez que se ingrese un nuevo carácter se enviara una trama nueva.

- Este software sirve tanto para la práctica de comunicación vía RS-232 como para comunicación USB.
- Se debe ser cuidadoso con los retardos.
- Si no se desea utilizar el software aquí presentado, se puede ejecutar un emulador de terminal propio del software BASCOM AVR

Pasos para ejecutar el terminal en BASCOM:

1. Tener la conexión entre la pc y el microcontrolador
2. Conocer cuál es el COM asignado a dicha conexión
3. Abrir el emulador de terminal

Figura 3. Simulador de terminal

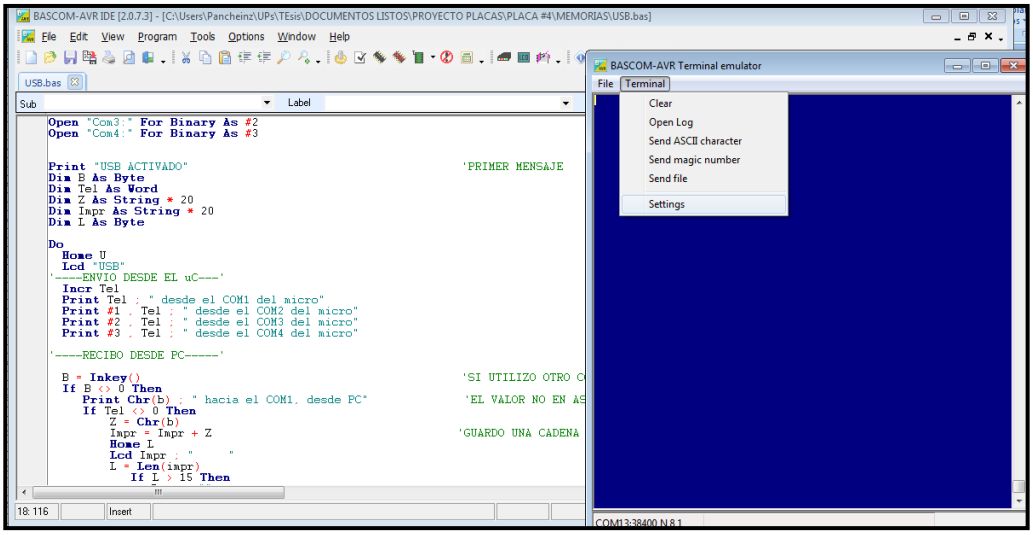


Fuente: (National Instruments, 2010)

Atajo con teclas CTRL+T

4. En el terminal se configuran los parámetros necesarios para establecer la conexión

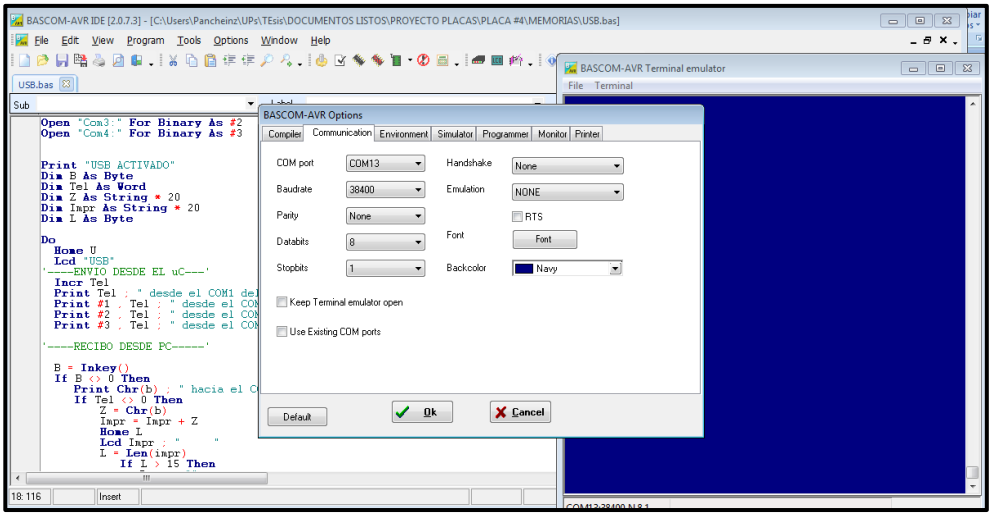
Figura 4. Simulador de terminal



Fuente: (National Instruments, 2010)

5. Configurar con los mismos parámetros con los que se configuro el microcontrolador.

Figura 5. Simulador de terminal



Fuente: (National Instruments, 2010)

6. Aceptar la configuración e inmediatamente se recibirá datos desde el microcontrolador debido a la programación cargada. Se puede enviar datos hacia el microcontrolador utilizando el teclado de la pc, dichos datos se visualizaran en el lcd de la placa

ANEXO 12

COMUNICACIONES SERIALES ASÍNCRONAS RS-232

PRÁCTICA 9

- **Tema**

  Comunicaciones seriales asíncronas

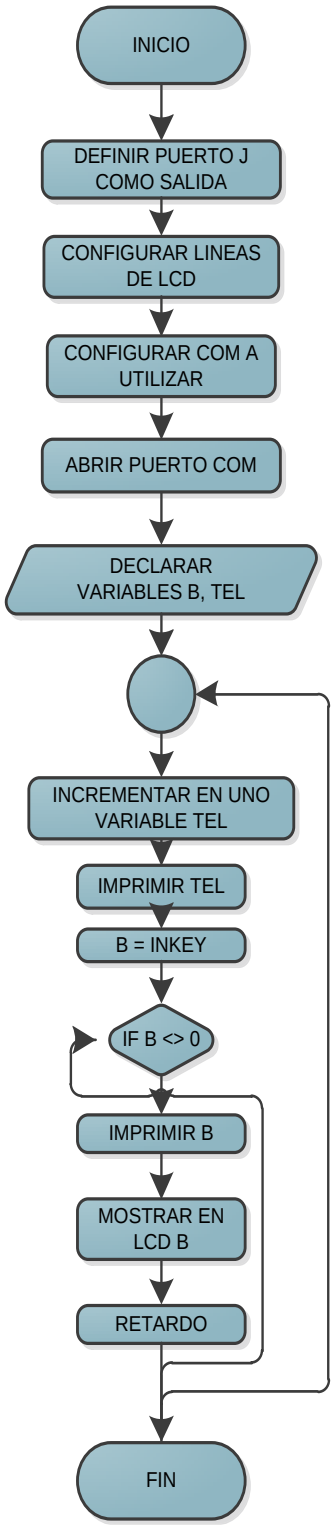
- **Objetivo**

  Utilizar las comunicaciones seriales RS-232

- **Desarrollo**

1) Transmisión y recepción serial con el microcontrolador XMega de Atmel.

Diagrama de flujo para hacer transmisión serial rs-232



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto J como salida*

- *Configurar las líneas del LCD*
- *Configurar COM a utilizar*
  - CONFIG COMx = ver en programación

- *Abrir puerto COM*
- Declarar variables a utilizar B, TEL
- *Inicio de lazo*
- Mostrar algún mensaje.
- *Se incrementa en uno la variable TEL*
- *Imprimir B*
- *Si B<> 0*
- Imprimir B en el terminal
- *Imprimir B*
- Fin de lazo
- Fin de programa

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Config Portj = Output
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2                'CONFIGURO LAS LINEAS DEL LCD
Config Lcd = 16 * 2
Cursor Off
Cls
```

```
Config Com1 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8
Config Com2 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8
Config Com3 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8
Config Com4 = 38400 , Mode = Asy
nchroneous , Parity = None , Stopbits = 1 , Databits = 8

Open "com2:" For Binary As #1
Open "Com3:" For Binary As #2
Open "Com4:" For Binary As #3

Print "ENVIO Y RECEPCIÓN"                    'PRIMER MENSAJE
Dim B As Byte
Dim Tel As Word

Do
  Home U
  Lcd "UART"
'----ENVIO DESDE EL uC---'
  Incr Tel
  Print Tel ; " desde el Uc COM1"
  Print #1 , Tel ; " desde el Uc COM2"
  Print #2 , Tel ; " desde el Uc COM3"
  Print #3 , Tel ; " desde el Uc COM4"

'----RECIBO DESDE PC-----'
  B = Inkey(#1)
  If B <> 0 Then
     Print #1 , Chr(b) ; "FROM COM2"          'EL VALOR NO EN ASCII,NORMAL
     Home L
     Lcd Chr(b) ; "     "
  End If
  Waitms 500
Loop
```

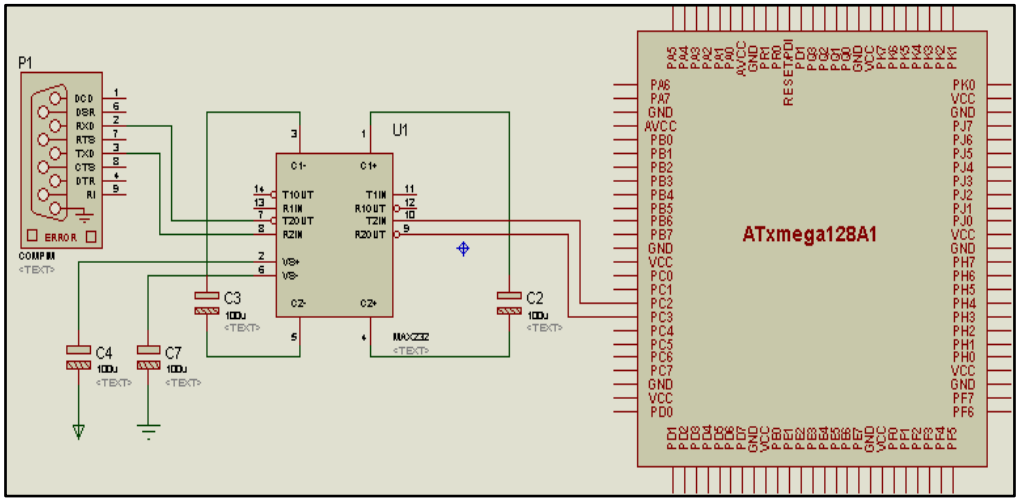
Close #1

Close #2

Close #3

End

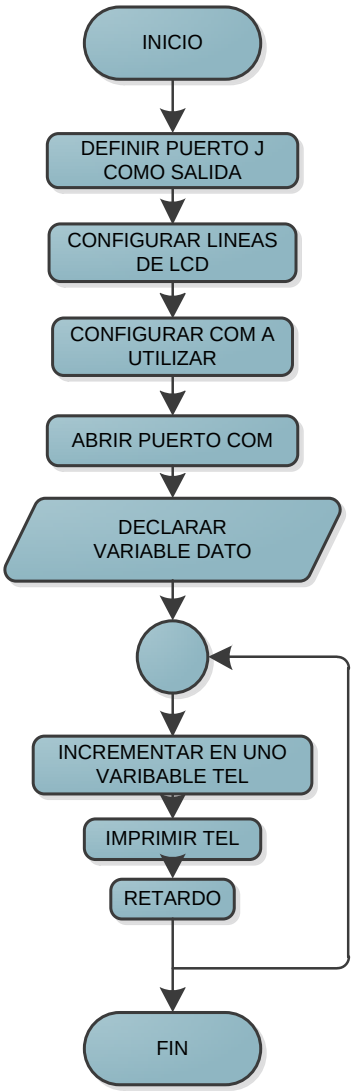
ESQUEMÁTICO



Conectar al puerto J el LCD.

2) Transmisión de datos desde el microcontrolador XMega de Atmel al PC. Utilizar el diagrama esquemático del ejercicio 1.

Diagrama de flujo para hacer transmisión desde el microcontrolador al pc



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto J como salida*
- *Configurar las líneas de la pantalla*
- *Abrir puerto COM*
- Declarar variables a utilizar DATO
- *Inicio de lazo*
- Se imprimirá algún mensaje
- Se incrementa en uno la variable TEL
- Imprimir TEL por algún COM
- *Fin de lazo*
- *Fin de programa*

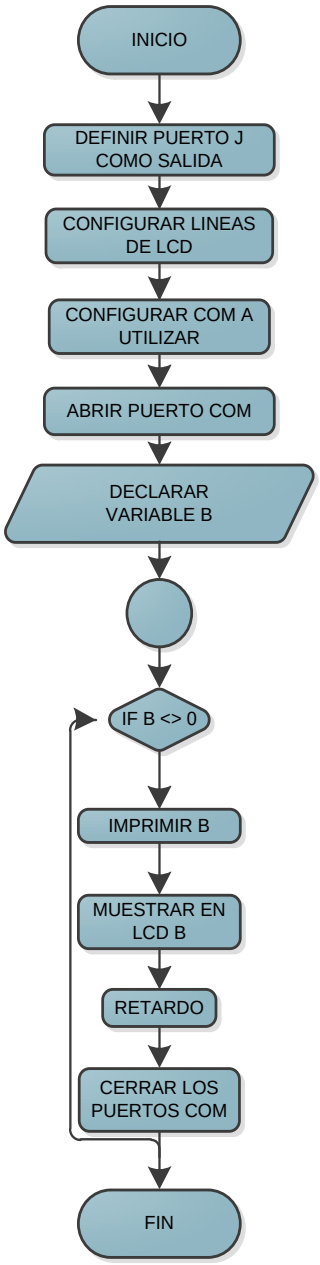
CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Config Portj = Output
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2       'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2
Cursor Off
Cls

'---CONFIGURANDO LOS COM DEL MICRO---'

'---CADA COM TIENE Rx, Tx EN UN PIN PARTICULAR---'
Config Com1 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM1=c2>Rx,c3>Tx
Config Com2 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM2=c6>Rx,c7>Tx
Config Com3 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM3=d2>Rx,d3>Tx
Config Com4 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM4=d6>Rx,d7>Tx

'Open all UARTS
'COM1 ES ACTIVO POR DEFAULT

Open "Com2:" For Binary As #1    'NOMBRAR AL COM2 COMO #1
Open "Com3:" For Binary As #2    'NOMBRAR AL COM2 COMO #2
```

```
Open "Com4:" For Binary As #3    'NOMBRAR AL COM2 COMO #3

Print "UART, DATOS ENVIADOS DESDE EL uC"
'PRIMER MENSAJE A MOSTRAR EN EL TERMINAL

Dim Dato As Word                 'PRINT>>> Enviar salida hacia el puerto RS-232

Do
  Home U
  Lcd "UART"

'---ENVIO DESDE EL uC---'
  Incr Dato
  Print Dato ; " dato enviado desde COM1 del uC"
  Print #1 , Dato ; " dato enviado desde COM2 del uC"
  Print #2 , Dato ; " dato enviado desde COM3 del uC"
  Print #3 , Dato ; " dato enviado desde COM4 del uC"
Loop

'-----CERRAR LOS COM ABIERTOS----'
Close #1
Close #2
Close #3

End
```

3) Envió de datos desde el PC al microcontrolador XMega. Utilizar el diagrama esquemático del ejercicio 1.

Diagrama de flujo para hacer envío desde el pc al microcontrolador



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto J como salida*
- *Configurar COM a utilizar*
- Abrir puerto COM a utilizar
- Declarar variables B

- Inicio de lazo
- Guardar el valor de INKEY en la variable B
- Comparar B <> 0
- Imprimir B en el terminal

- *Fin de lazo*
- *Fin de programa*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Config Portj = Output
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2       'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2
Cursor Off
Cls

'---CONFIGURANDO LOS COM DEL MICRO---'

'---CADA COM TIENE Rx, Tx EN UN PIN PARTICULAR---'
Config Com1 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM1=c2>Rx,c3>Tx
Config Com2 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM2=c6>Rx,c7>Tx
Config Com3 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM3=d2>Rx,d3>Tx
Config Com4 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM4=d6>Rx,d7>Tx



'Open all UARTS
```

```
'COM1 ES ACTIVO POR DEFAULT

Open "Com2:" For Binary As #1          'NOMBRAR AL COM2 COMO #1
Open "Com3:" For Binary As #2          'NOMBRAR AL COM2 COMO #2
Open "Com4:" For Binary As #3          'NOMBRAR AL COM2 COMO #3



Print "UART, DATOS ENVIADOS DESDE EL PC"
'PRIMER MENSAJE A MOSTRAR EN EL TERMINAL

Dim B As Byte

Do

'---RECIBIR DESDE PC---'
  B = Inkey(#1)                    'RECIBIR POR EL COM2>>>#1, PINES C6,C7
  If B <> 0 Then                   'PARA HACER USO DEL COM1 SOLO
                                   'SE DEBE ESCRBIR B=INKEY()

  Print #1 , Chr(b) ; "  DESDE EL PC AL COM2"
 'TRANSFORMAR EL VALOR ASCCI A CARACTER
    Home L
    Lcd Chr(b) ; "     "           'ESCRIBIR EN EL LCD CADA DATO RECIBIDO
  End If
  Waitms 500
Loop

'---CERRAR LOS COM ABIERTOS---'
Close #1
Close #2
Close #3

End
```

4) Ejercicio de aplicación: Realice la adquisición de datos de un potenciómetro y envíelos para que se observe en el hyperterminal de Windows.

5) Ejercicio de aplicación: Realice al comunicación entre el microcontrolador XMega de Atmel y Labview por el puerto serie

ANEXO 13

COMUNICACIONES SERIALES CON USB

PRÁCTICA 10

- **Tema**

  Comunicaciones seriales con USB

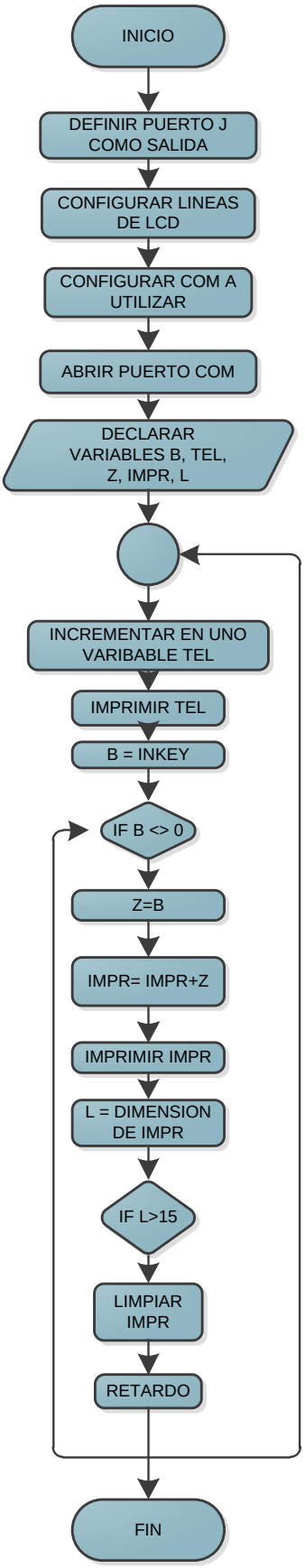
- **Objetivo**

Realizar comunicaciones USB del microcontrolador XMega con USB del PC.

- **Desarrollo**

1) El siguiente ejemplo permite enviar y recibir datos desde el microcontrolador XMega al PC

Diagrama de flujo para hacer envio y recepción de datos vía usb



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*

- *Puerto J como salida*
- *Configurar las líneas del LCD*
- *Configurar COM a utilizar*
- Abrir puerto COM a utilizar

- Declarar variables a utilizar B, TEL
- *Inicio de lazo*
- Mostrar algún mensaje en el LCD
- Incrementar en uno la variable TEL
- Imprimir TEL por vía COM
- *B= KINKEY*
- *Si B<> 0*
- *B = Z*
- *IMPR= IMPR+Z*
- *L = DIMENSION DE IMP*
- *Fin de programa*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Config Portj = Output
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2                'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2
Cursor Off
```

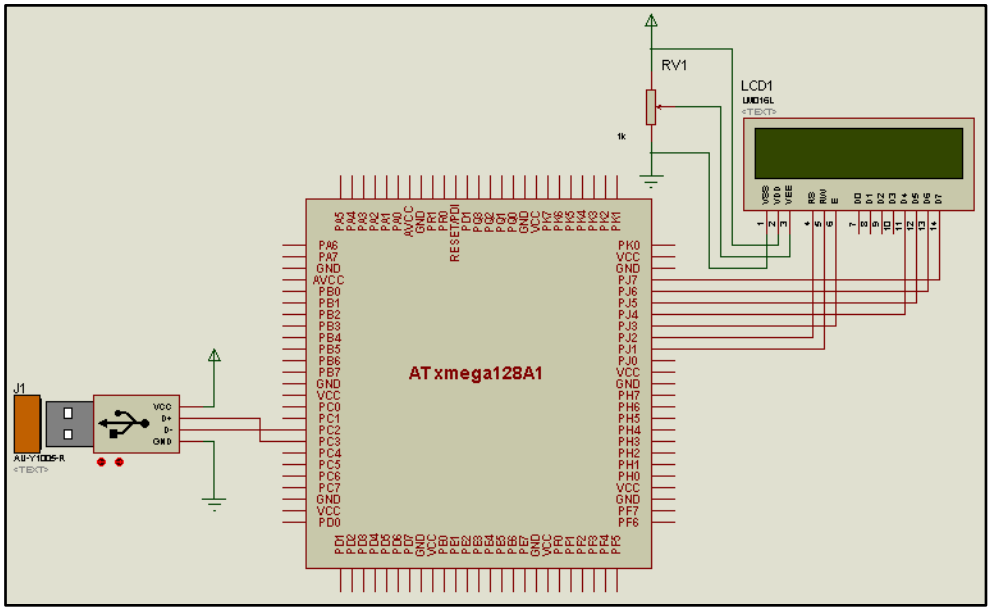
```
Cls



'---------CONFIGURANDO LOS COM DEL MICRO---------'

'---------CADA COM TIENE Rx, Tx EN UN PIN PARTICULAR-----'
Config Com1 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM1=c2>Rx,c3>Tx
Config Com2 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM2=c6>Rx,c7>Tx
Config Com3 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM3=d2>Rx,d3>Tx
Config Com4 = 38400 , Mode = Asynchroneous , Parity = None , Stopbits = 1 ,
Databits = 8      'COM4=d6>Rx,d7>Tx



'Open all UARTS
'COM1 ES ACTIVO POR DEFAULT

Open "com2:" For Binary As #1
Open "Com3:" For Binary As #2
Open "Com4:" For Binary As #3



Print "USB ACTIVADO"                  'PRIMER MENSAJE
Dim B As Byte
Dim L As Byte
Dim Tel As Word
Dim Z As String * 20
Dim Impr As String * 16

Do
  Home U
  Lcd "USB"

'---ENVIO DESDE EL uC---'
```

```
  Incr Tel
  Print Tel ; " desde el COM1 del micro"
  Print #1 , Tel ; " desde el COM2 del micro"
  Print #2 , Tel ; " desde el COM3 del micro"
  Print #3 , Tel ; " desde el COM4 del micro"

'---RECIBO DESDE PC---'

  B = Inkey()
 'SI SE UTILIZA OTRO COM SE DEBERÁ ESCRIBIR
‘Inkey (#1)>>>Com2; Ó inkey(#2)>>>Com3; ETC.

  If B <> 0 Then
    Print Chr(b) ; " hacia el COM1, desde PC"
    'EL VALOR NO EN ASCII,NORMAL, PARA IMPRIMIR
    ‘HACIA OTRO COM(2,3,etc) print #1, char (b)>>>COM2

     If Tel <> 0 Then
       Z = Chr(b)
       Impr = Impr + Z
       'GUARDAR UNA CADENA DE TEXTO EN VAR=IMPR

        Home L
        Lcd Impr ; "    "

        L = Len (impr)
             If L > 15 then
               Impr = “  ”
            End If
     End If
  End If
  Waitms 500
Loop

Close #1
Close #2
```

Close #3

End

ESQUEMÁTICO



2) Ejercicio de aplicación: Escriba un programa que reciba datos del PC.

ANEXO 14

COMUNICACIONES SERIALES SINCRÓNICAS SPI.

PRÁCTICA 11

- **TEMA**

Comunicaciones seriales sincrónicas SPI.

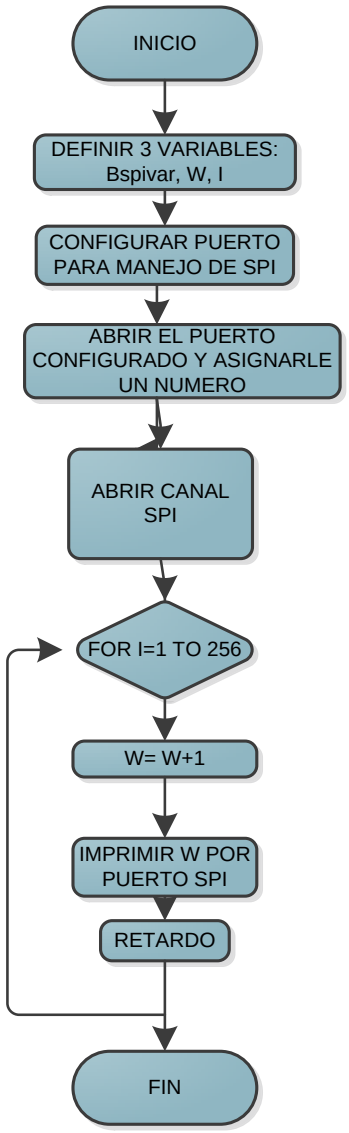
- **OBJETIVO**

  Utilizar las comunicaciones seriales sincrónicas SPI, entre el microcontrolador XMega y periféricos

- **DESARROLLO**

  1) El siguiente programa es utilizado para controlar el max7219.

Diagrama de flujo para transmisión con comunicación spi



Elaborado por: Francisco Reyes y Nina Chicaiza

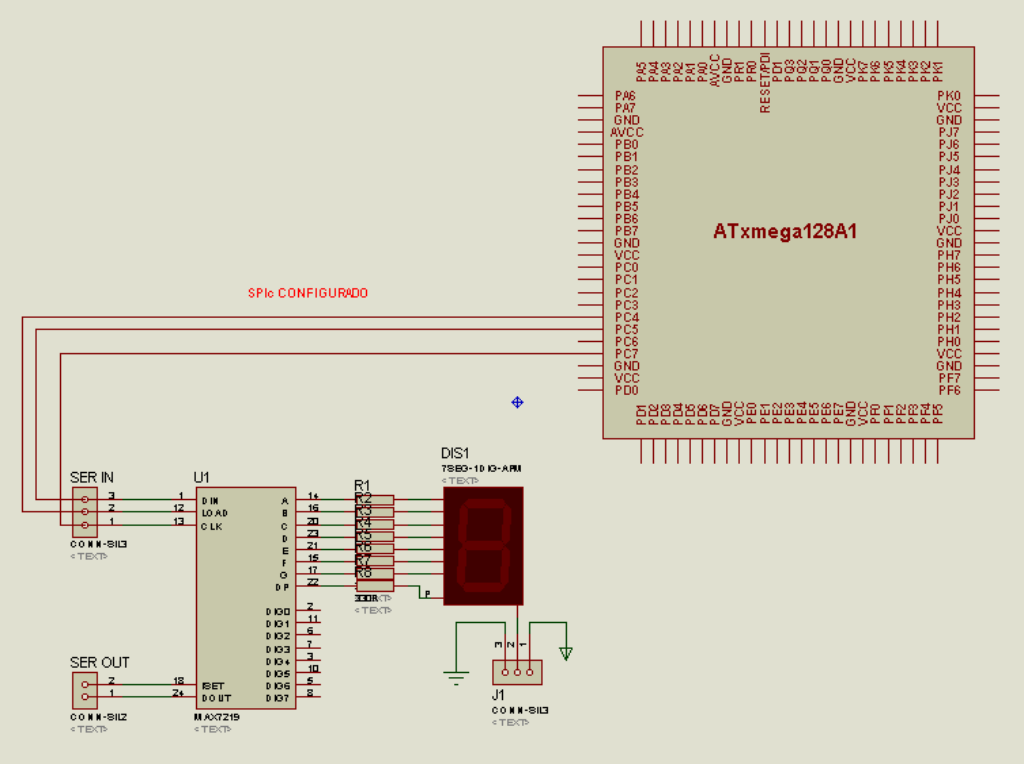
- *Inicio.*
- *Definir tres variables*
- *Configurar puerto para manejo de SPI*
- *Abrir el puerto configurado y asignarle un número*
- *Abrir el canal SPI*
- *For i= 1 To 256*
- *W=W+1*
- *Imprimir W por puerto SPI*
- Se asigna un tiempo de espera
- *Fin del lazo*
- *Fin de programa*

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000                          '2MHz
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014



Dim Bspivar As Byte , Ar(4) As Byte , W As Word

Dim L As Byte
Dim I As Word
L = 0
I = 0
Bspivar = 1

Config Spic = Hard , Master = Yes , Mode = 0 , Clockdiv = Clk2 , Data_order = Msb ,
Ss = Auto       '_SS = portC.4  MOSI = portC.5   SCK = portC.7
Config Spid = Hard , Master = Yes , Mode = 1 , Clockdiv = Clk8 , Data_order = Lsb
Config Spie = Hard , Master = Yes , Mode = 2 , Clockdiv = Clk4 , Data_order = Msb
Config Spif = Hard , Master = Yes , Mode = 3 , Clockdiv = Clk32 , Data_order = Msb



Open "SPIC" For Binary As #10
Open "SPID" For Binary As #11
Open "SPIE" For Binary As #12
Open "SPIF" For Binary As #13
Open "SPI" For Binary As #bspivar                    ' use a dynamic channel
'SPI channel only suppor PRINT and INPUT
```

```
Do

For I = 1 To 256
  Incr W
  Print #10 , W
  Waitms 50
Next
Loop
```

ESQUEMÁTICO



ANEXO 15

CONTROL DE MOTORES

PRÁCTICA 12

- **Tema**

  Control de motores PWM

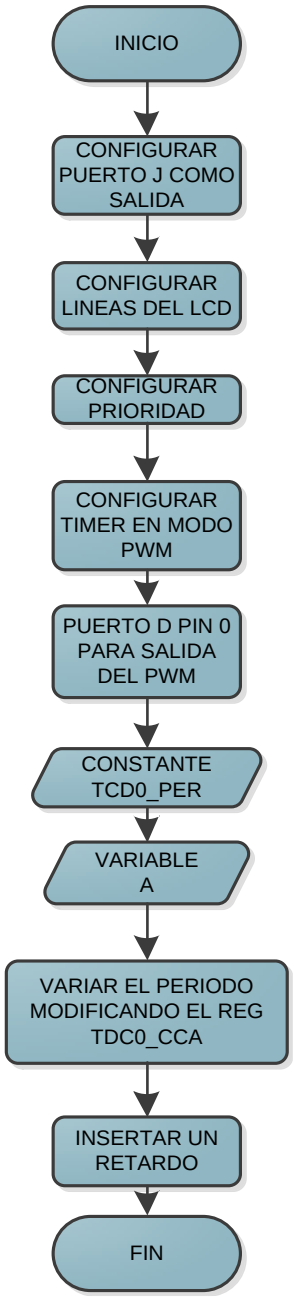
- **Objetivo**

  Utilizar el microcontrolador para operar motores con configuraciones distintas, entre ellas está el PWM.

- **Desarrollo:**

  5) Encendido de un led mediante el uso de PWM utilizando el microcontrolador XMega.

Diagrama de flujo para realizar el encendido de un led mediante el uso de pwm



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto J como salida*
- *Configurar las líneas del LCD*
- *Configurar prioridad*
- *Configurar TIMER esta vez para PWM*

- *Definir un periodo*
- *Puerto D Pin 0 para salida PWM*

- Definir el período
- *Variar el periodo modificando*
- *Fin de programa*

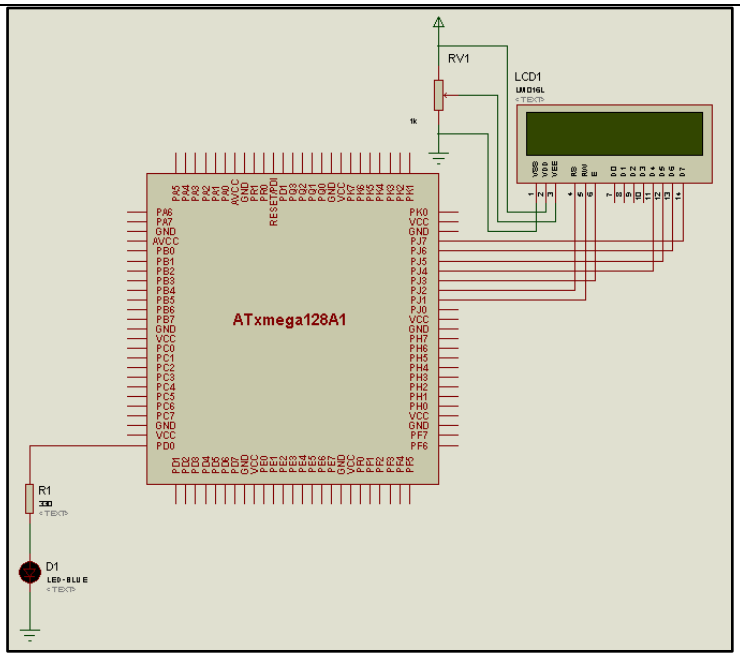
CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000                              '2MHz
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Ddrj = 255
Portj = 0
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2      'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2             'CONFIGURAR TIPO DE LCD

'---USANDO PWM SALIDA PIND.0---'

Config Priority = Static , Vector = Application , Lo = Enabled
'ACTIVAR INT DE NIVEL BAJO



Config Tcd0 = Pwm , Prescale = 2 , Comparea = Enabled , Resolution = 16
'CONFIGURAR EL TIMER d0 PARA GENERACION PWM



' Tcd0 --> pwm --> PWM single slope
' Prescale = 2 --> 2MHz/2 = 1MHz        ---2MHz del crystal---
' Comparea = enabled --> ACTIVADO COMPARE o CAPTURE A
' Resolution = 16 --> 16-Bit Resolución
```

```
' <--TCD0_CCA->  VARIABLE A VARIAR
'      +--------+         +
'      |         |         |
'------+          +--------+
'
'     <---------------------->
'       Periodo = 65.54mSec
'SET Resolución PWM (min. = &H0003 ..... max. = &HFFFF)

Config Portd.0 = Output              'PIN DE SALIDA PA PWM



Tcd0_per = &HFFFF                    'CONF PERIODO = FFFF = 65535 -->
                                     ‘65535/1MHz =65.54mSec

Dim A As Word
Cls
Do

  Home U
  Lcd "MODULACION PWM"
  Waitms 500
  Tcd0_cca = 20000                   '20000/1MHz = 20ms CAMBIAR TCD0_CCA
  Home L
  Lcd Tcd0_cca ; "   "

  Waitms 500
  Tcd0_cca = 10000                   '10ms 'CAMBIAR TCD0_CCA
  Home L
  Lcd Tcd0_cca ; "   "
  Wait 2



  'CAMBIAR EL DUTY CYCLE DESDE 0 HASTA 30000 = 30ms
```

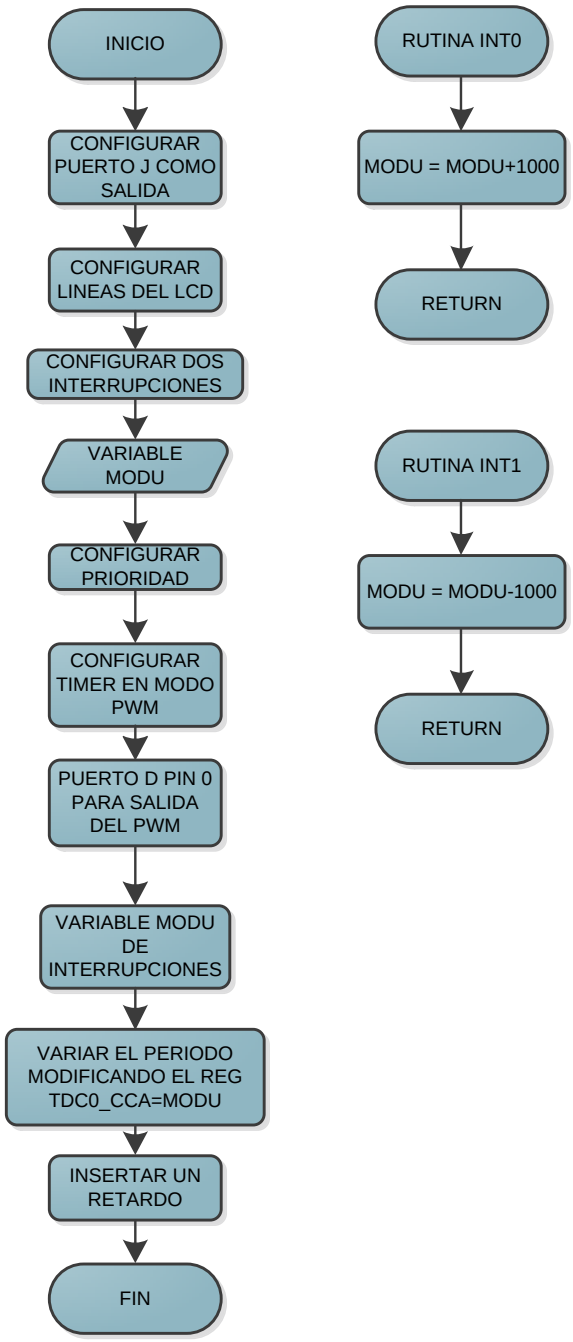
```
Do
        Incr A
        Tcd0_cca = A
        Home L
        Lcd Tcd0_cca ; "    "
        Waitus 100
        Loop Until A = 30000

     A = 0
   Wait 2
Loop

End
```

ESQUEMÁTICO



6) Mediante 2 pulsadores controle la velocidad de un motor DC, los pulsadores incrementaran la velocidad en pasos de 10%.

Diagrama de flujo para controlar la velocidad de un motor dc



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto J como salida*
- *Configurar las líneas del LCD*
- *Definir variable MODU*
- Configurar prioridad
- Configurar interrupción
- Activar las interrupciones

- *Configurar timer en modo PWM*
- *Puerto D PIN 0 para salida PWM*
- *Rutina INTO e INT1*
- Se devuelve la el valor en la variable MODU
- *Variar el periodo modificando*
- Fin de programa

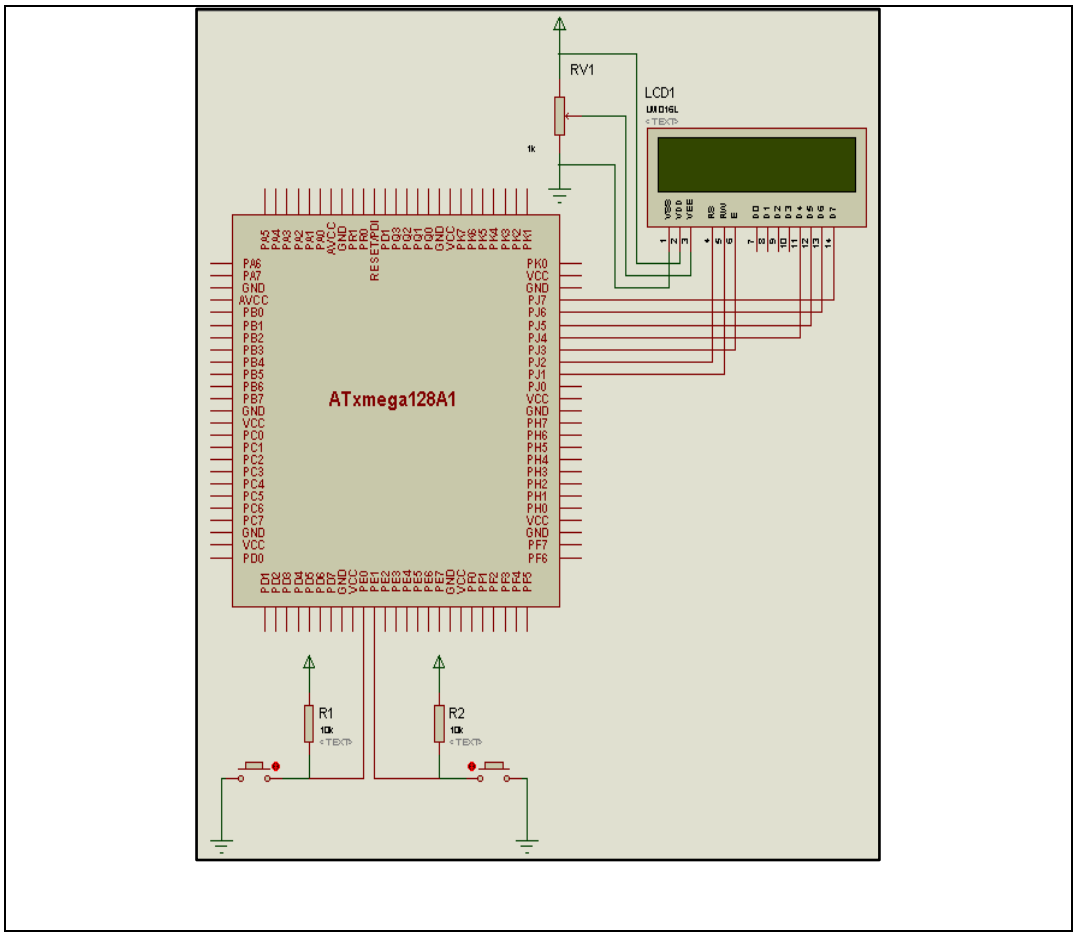
CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Config Priority = Static , Vector = Application , Lo = Enabled
'ACTIVAR INT DE NIVEL BAJO

Ddrj = 255
Portj = 0
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2              'CONFIGURAR LAS LINEAS DEL LCD
Config Lcd = 16 * 2                 'CONFIGURAR TIPO DE LCD

'---CONFIGURACIÓN PARA HABILITAR INTERRUPCIONES EXTERNAS---'

Enable Porte_int0 , Lo
Enable Porte_int1 , Lo              'ACTIVANDO LA INTERRUPCIÓN
                                    ‘COMO DE BAJO NIVEL

Config Pine.0 = Input               'PIN PARA ACTIVAR LA INTERRUPCIÓN
Config Pine.1 = Input
```

```
On Porte_int0 Port_e_int0_interrupcionext
'NOMBRAR A LA INTERRUPCIÓN COMO PORT_E_INT0_INTERRUPCION EXT

On Porte_int1 Port_e_int1_interrupcionext1
Enable Interrupts

Porte_pin0ctrl = &B00_011_010
'ENABLE PULLUP AND REACTION ON FALLING EDGE  PIN0 PORTC

Porte_pin1ctrl = &B00_011_010
'ENABLE PULLUP AND REACTION ON FALLING EDGE  PIN1 PORTC

Porte_int0mask = &B0000_0001        'EL PIN0 PARA LA INT0
Porte_int1mask = &B0000_0010        'EL PIN1 PARA LA INT1

'---USANDO PWM SALIDA PIND.0---'

Config Tcd0 = Pwm , Prescale = 2 , Comparea = Enabled , Resolution = 16
'CONFIGURAR EL TIMER d0 PARA GENERACION PWM

Config Portd.0 = Output                  'PIN DE SALIDA PA PWM
Tcd0_per = &HFFFF                        'CONF PERIODO = FFFF = 65535 -->
                                         ‘65535/1MHz =65.54mSec

Tcd0_cca = 10000                         'REGISTRO A VARIAR var TCD0_CCA
                                         ‘10000/1MHz = 10mSec

Dim A As Word
Dim Modu As Word

Cls
Cursor Off
Do

  Home U
```

```
  Lcd "MODULACION PWM"
  Waitms 500
  Tcd0_cca = Modu
  Home L
  Lcd Tcd0_cca ; "    "

Loop

'---RUTINA DE INTERRUPCIÓN---'
Port_e_int0_interrupcionext:

 Modu = Modu + 1000

Return

'---RUTINA DE INTERRUPCIÓN---'
Port_e_int1_interrupcionext1:

 Modu = Modu - 1000

Return
```

ESQUEMÁTICO

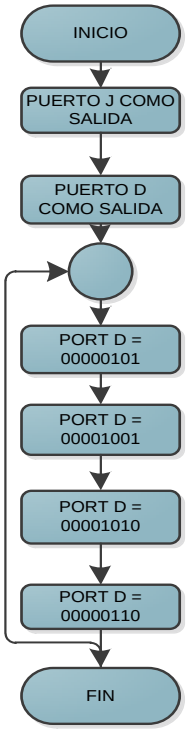


7) Control de un motor pasó a paso en secuencia normal, utilizando el microcontrolador XMega.

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 y debido a que siempre hay al menos dos bobinas activadas, se obtiene un alto torque de paso y de retención.

PASO	BOBINA A	BOBINA B	BOBINA C	BOBINA D
1	ON	ON	OFF	OFF
2	OFF	ON	ON	OFF
3	OFF	OFF	ON	ON
4	ON	OFF	OFF	ON

Diagrama de flujo para control de un motor pasó a paso



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto J como salida*
- *Puerto D como salida*
- Inicio de lazo
- Escribir en el puerto de salida la siguiente serie:
  - 5,9,10,6 (00000101, 00001001, 00001010, 00000110; respectivamente)
- *Fin de lazo*
- *Fin de programa*

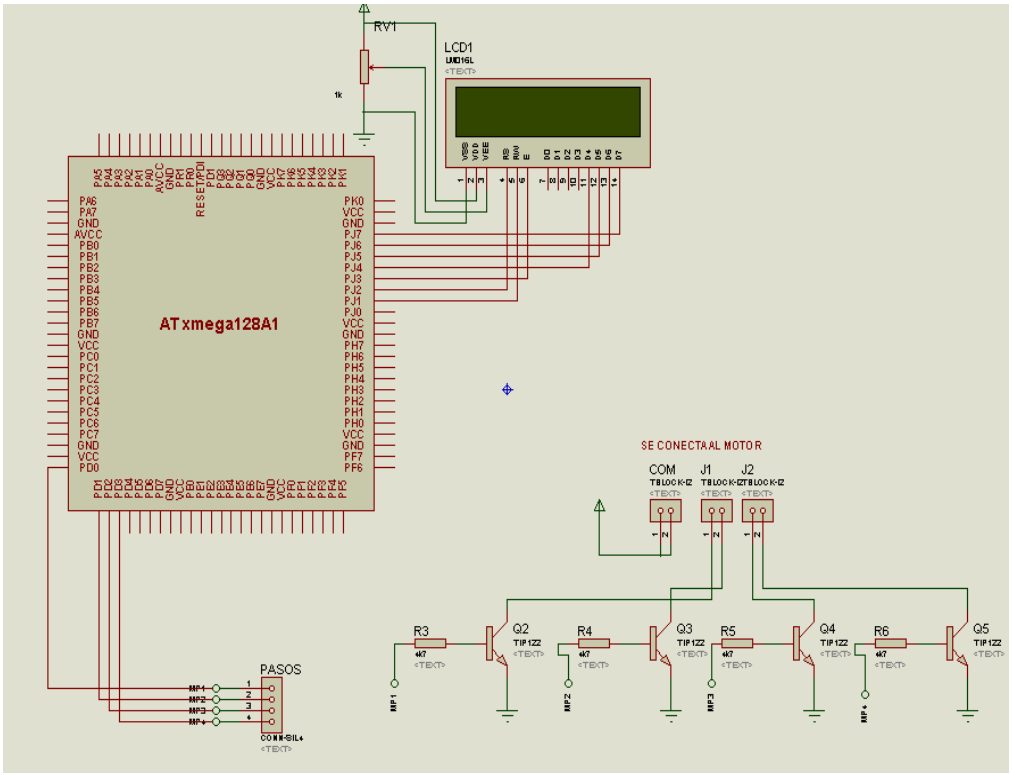
CÓDIGO
$regfile = "xm128a1def.dat" $crystal = 2000000 '2MHz $hwstack = 64 $swstack = 40 $framesize = 40 $lib "xmega.lib"

```
$external _xmegafix_clear
$external _xmegafix_rol_r1014



'CONFIGURO LAS LINEAS DEL LCD
Ddrj = 255
Portj = 0
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2
Config Lcd = 16 * 2                    'CONFIGURO TIPO DE LCD

'-----PINES PARA CONTROLAR LAS BOBINAS----'
Config Portd.0 = Output                'A
Config Portd.1 = Output                'B
Config Portd.2 = Output                'C
Config Portd.3 = Output                'D

Cls
Cursor Off
Do
  Home U : Lcd "MOTOR PaP"

  Portd = &B00000101
  Waitms 200
  Home L : Lcd Bin(portd)

  Portd = &B00001001
  Waitms 200
  Home L : Lcd Bin(portd)

  Portd = &B00001010
  Waitms 200
  Home L : Lcd Bin(portd)

  Portd = &B00000110
  Waitms 200
```

Home L : Lcd Bin(portd)

Loop

ESQUEMÁTICO



8) Control de un motor pasó a paso en secuencia tipo wave drive, utilizando el microcontrolador XMega. Utilizar el diagrama esquemático del ejercicio 3.

Secuencia del tipo wave drive: En esta secuencia se activa solo una bobina a la vez. En algunos motores esto brinda un funcionamiento más suave. La contrapartida es que al estar solo una bobina activada, el torque de paso y retención es menor.

PASO	BOBINA A	BOBINA B	BOBINA C	BOBINA D
1	ON	OFF	OFF	OFF

2	OFF	ON	OFF	OFF
3	OFF	OFF	ON	OFF
4	OFF	OFF	OFF	ON

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000                          '2MHz
$hwstack = 64
$swstack = 40
$framesize = 40

$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

'CONFIGURO LAS LINEAS DEL LCD
Ddrj = 255
Portj = 0
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2
Config Lcd = 16 * 2                         'CONFIGURO TIPO DE LCD

'-----PINES PARA CONTROLAR LAS BOBINAS----'
Config Portd = Output

Cls
Cursor Off
Do
  Home U : Lcd "MOTOR PaP"

  Portd = &B00000001
  Waitms 200
  Home L : Lcd Bin(portd)

  Portd = &B00001000
  Waitms 200
```

```
  Home L : Lcd Bin(portd)

  Portd = &B00000010
  Waitms 200
  Home L : Lcd Bin(portd)

  Portd = &B00000100
  Waitms 200
  Home L : Lcd Bin(portd)
Loop
```

5) EJERCICIO DE APLICACIÓN: Utilizar un motor DC que se controle mediante un mosfet

ANEXO 16

RADIOFRECUENCIA

PRÁCTICA 13

- **Tema**

Transmisión Radiofrecuencia

- **Objetivo**

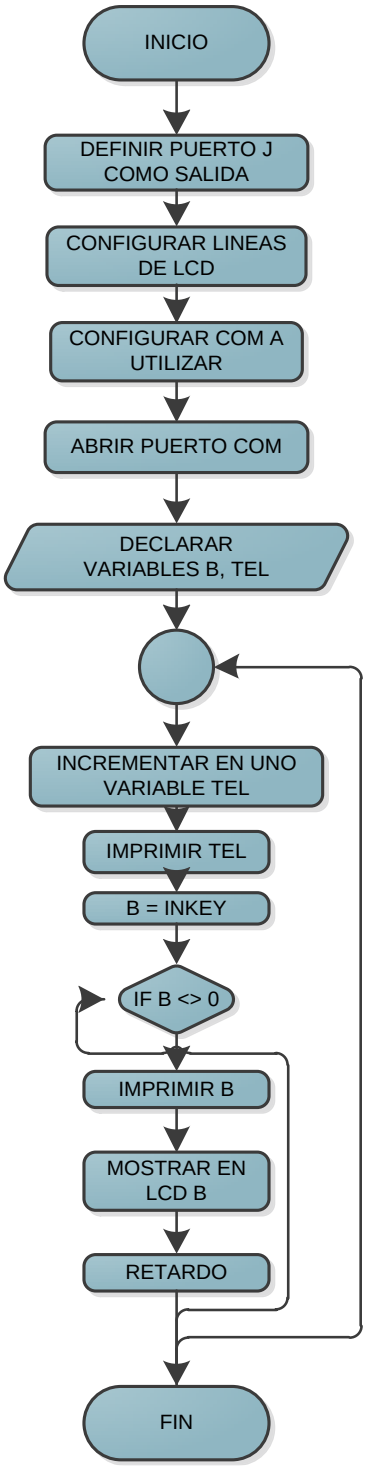
  Utilizar los puertos del microcontrolador XMega de Atmel para tener comunicación RF.

- **Desarrollo**

1) Envío de datos desde el microcontrolador XMega con el módulo de radiofrecuencia XBEE.

Diagrama de radiofrecuencia

.



Elaborado por: Francisco Reyes y Nina Chicaiza

- *Inicio*
- *Puerto J como salida*
- *Configurar las líneas del LCD*
- *Configurar COM a utilizar*
- *Abrir puerto COM*

- Declarar variables a utilizar B, TEL
- *Inicio de lazo*
- Mostrar algún mensaje en el lcd
- *Se incrementa en uno la variable TEL*
- *Imprimir B*
- Enviar un cero si no hay carácter esperando ser enviado
- Guardar el valor de INKEY en la variable B
- *Si B<> 0*
- Imprimir B en el terminal
- Mostrar B en el lcd
- Fin de programa

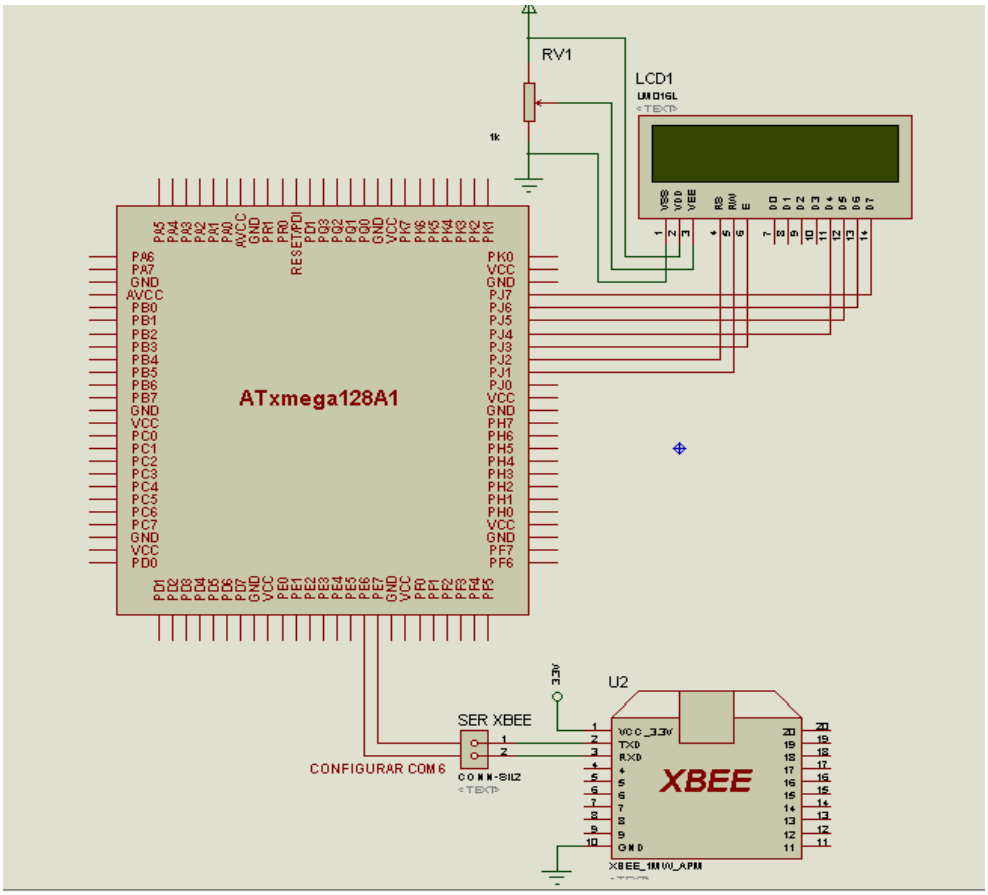
CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000                          '2MHz
$hwstack = 64
$swstack = 40
$framesize = 40
$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

Config Portj = Output
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E =
Portj.3 , Rs = Portj.2       'CONFIGURO LAS LINEAS DEL LCD
Config Lcd = 16 * 2
Cursor Off
Cls
'---------CONFIGURANDO LOS COM DEL MICRO---------'
'---------CADA COM TIENE Rx, Tx EN UN PIN PARTICULAR-----'
Config Com1 = 9600 , Mode = Asynchroneous , Parity = None , Stopbits = 1 , Databits
= 8      'COM1=c2>Rx,c3>Tx

Config Com2 = 9600 , Mode = Asynchroneous , Parity = None , Stopbits = 1 , Databits
```

```
= 8       'COM2=c6>Rx,c7>Tx

Config Com3 = 9600 , Mode = Asynchroneous , Parity = None , Stopbits = 1 , Databits
= 8       'COM3=d2>Rx,d3>Tx

Config Com4 = 9600 , Mode = Asynchroneous , Parity = None , Stopbits = 1 , Databits
= 8       'COM4=d6>Rx,d7>Tx

'COM1 ES ACTIVO POR DEFAULT

Open "Com2:" For Binary As #1                    'NOMBRAMOS AL COM2
COMO #1
Open "Com3:" For Binary As #2                    'NOMBRAMOS AL COM3
COMO #2
Open "Com4:" For Binary As #3                    'NOMBRAMOS AL COM4
COMO #3

Print "XBEE, DATOS ENVIADOS DESDE EL uC"          'PRIMER MENSAJE
A MOSTRAR EN EL TERMINAL
Dim Dato As Word

Do
  Home U
  Lcd "XBEE"
'----ENVIO DESDE EL uC---'
  Incr Dato
  Print Dato ; " XBEE COM1 del uC"
  Print #1 , Dato ; " XBEE COM2 del uC"
  Print #2 , Dato ; " XBEE COM3 del uC"
  Print #3 , Dato ; " XBEE COM4 del uC"
Loop

'-----CIERRO LOS COM ABIERTOS----'
Close #1
Close #2
Close #3
```

End

ESQUEMÁTICO



9) Transmisión y recepción de datos con el módulo de radiofrecuencia XBEE. Utilizar el diagrama esquemático del ejercicio 1.

CÓDIGO

```
$regfile = "xm128a1def.dat"
$crystal = 2000000                    '2MHz
$hwstack = 64
$swstack = 40
$framesize = 40
```

```
$lib "xmega.lib"
$external _xmegafix_clear
$external _xmegafix_rol_r1014

'CONFIGURO LAS LINEAS DEL LCD
Config Portj = Output
Config Lcdpin = Pin , Db4 = Portj.4 , Db5 = Portj.5 , Db6 = Portj.6 , Db7 = Portj.7 , E = Portj.3 , Rs = Portj.2
Config Lcd = 16 * 2
Cursor Off
Cls



Config Com1 = 9600 , Mode = Asynchroneous , Parity = None , Stopbits = 1 , Databits = 8
'CONFIGURACION DEL PUERTO COM1
Config Com2 = 9600 , Mode = Asynchroneous , Parity = None , Stopbits = 1 , Databits = 8
'CONFIGURACION DEL PUERTO COM2
Config Com3 = 9600 , Mode = Asynchroneous , Parity = None , Stopbits = 1 , Databits = 8
'CONFIGURACION DEL PUERTO COM3
Config Com4 = 9600 , Mode = Asynchroneous , Parity = None , Stopbits = 1 , Databits = 8
'CONFIGURACION DEL PUERTO COM 4



                                    'EL COM1 NO NECESITA NOMBRE DE INTERFACE
Open "Com2:" For Binary As #1                    'ASIGNAR UN NOMBRE DE INTERFACE PARA EL COM2
Open "Com3:" For Binary As #2                    'ASIGNAR UN NOMBRE DE INTERFACE PARA EL COM3
Open "Com4:" For Binary As #3                    'ASIGNAR UN NOMBRE DE INTERFACE PARA EL COM4
```

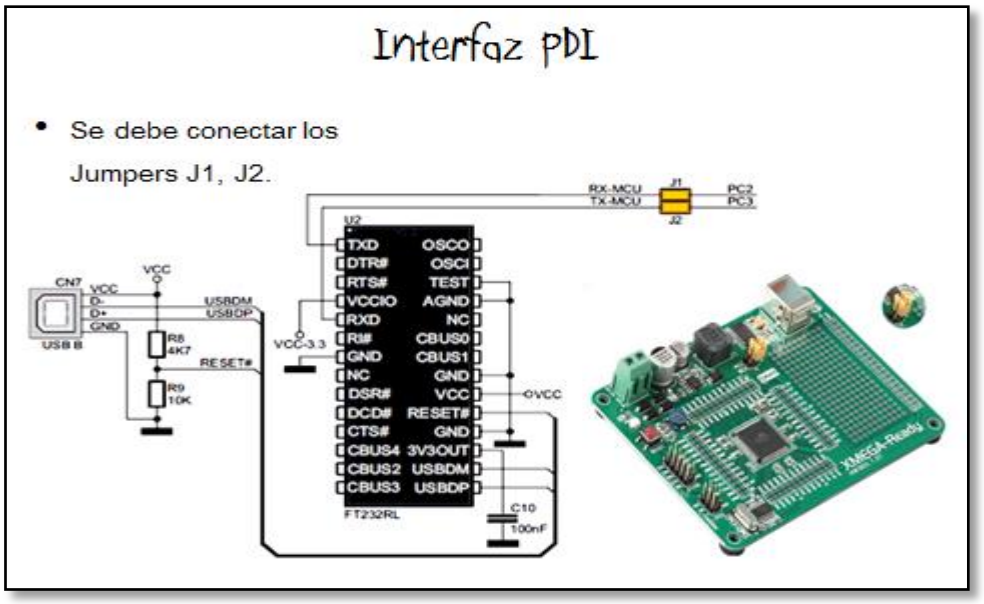
```
Print "ENVIO Y RECEPCIÓN"                    'IMPRIMIR MENSAJE
Dim B As Byte
Dim Tel As Word

Do                                          'INICIO DE LAZO
  Home U
  Lcd "XBEE"                                'IMPRIMO MENSAJE
'----ENVIO DESDE EL uC---'
  Incr Tel                                  'INCREMENTO LA VARIABLE A ENVIAR
EN UNO
  Print Tel ; " DESDE uC COM1"                          'IMPRIMO EN LA PC EL
VALOR DE LA VARIABLE Tel MAS UN MENSAJE
  Print #1 , Tel ; " DESDE uC COM2"                     'IMPRIMO EN LA PC EL
VALOR DE LA VARIABLE Tel MAS UN MENSAJE
  Print #2 , Tel ; " DESDE uC COM3"                     'IMPRIMO EN LA PC EL
VALOR DE LA VARIABLE Tel MAS UN MENSAJE
  Print #3 , Tel ; " DESDE uC COM4"                     'IMPRIMO EN LA PC EL
VALOR DE LA VARIABLE Tel MAS UN MENSAJE

  'EN HARDWARE SE DEFINE QUE COM UTILIZAR

'----RECIBO DESDE PC-----'
  B = Inkey(#1)                             'ESPERAR ALGUN DATO DESDE LA
PC, UTILIZANDO EL COM2, tiene el alias #1
  If B <> 0 Then                            'CONDICION DE ESPERA DEL DATO
    Print #1 , Chr(b) ; "DESDE PC COM2"             'IMPRIMIR EL VALOR EN
NUMERO DE LA TECLA QUE SE PRESIONE EN LA PC
    Home L
    Lcd Chr(b) ; "     "                    'MOSTRAR EN LCD
  End If
  Waitms 500
Loop



Close #1                                    'CERRAR EL PUERTO
```

```
Close #2                    'CERRAR EL PUERTO
Close #3                    'CERRAR EL PUERTO

End
```

ANEXO 17

DETALLE DE LOS PINES EN LA PLACA DE PRÁCTICAS

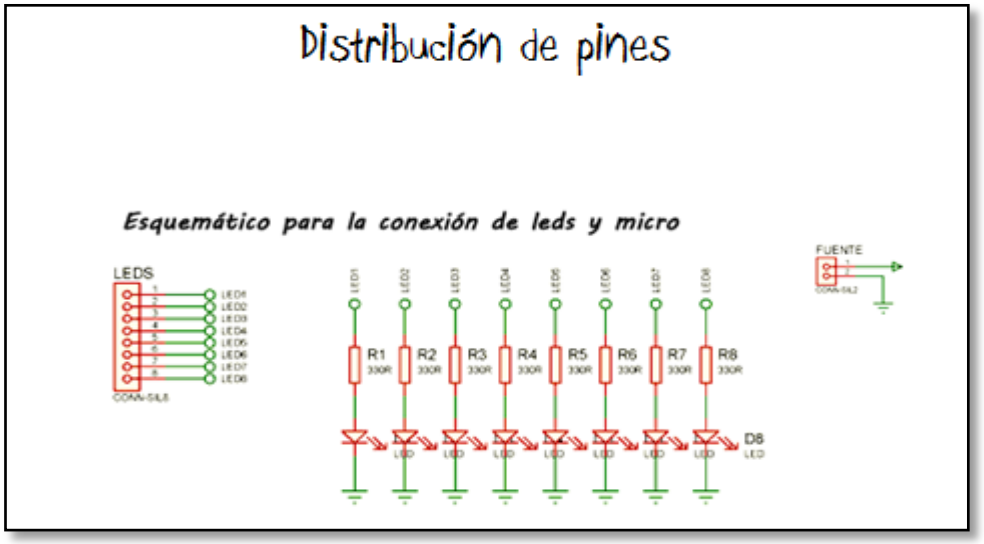
Conexión de la interfaz para programar el microcontrolador



Fuente: (XMega Ready, 1998)

Elaborado por: Francisco Reyes y Nina Chicaiza

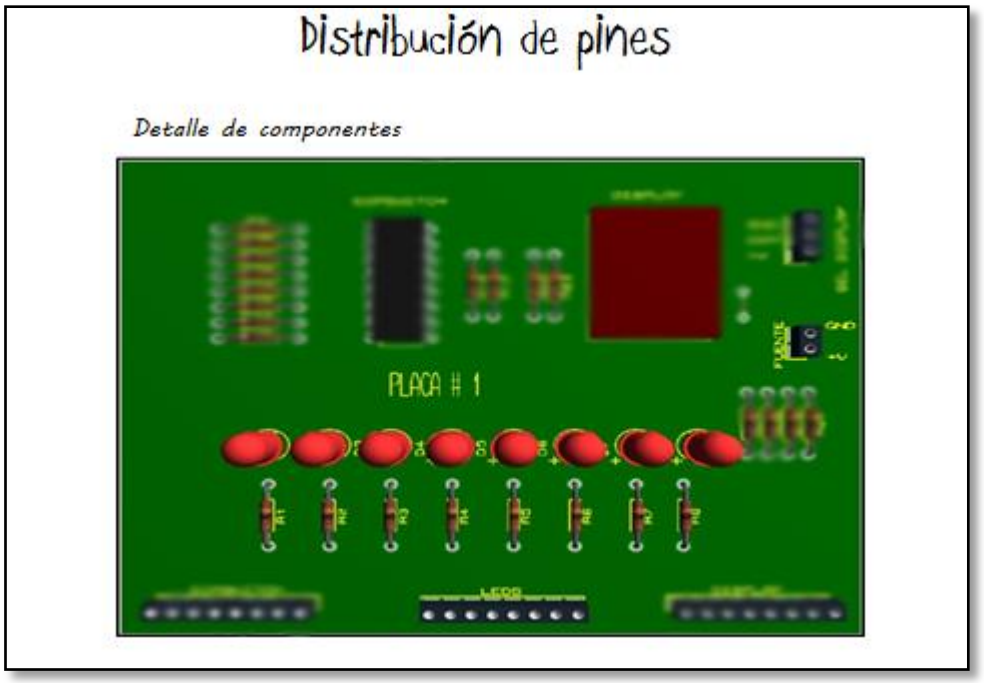
- Detalle de pines para las prácticas con los leds



Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

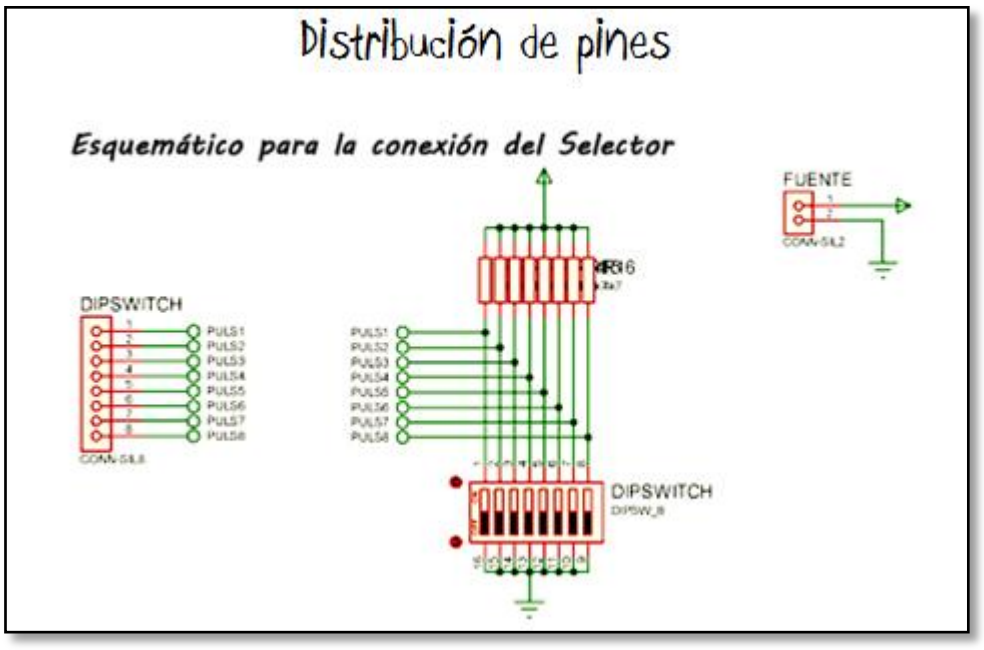
- Ubicación de los sócalos para el control de leds



Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

- Detalle de pines para el control del dipswitch



Fuente:Isis

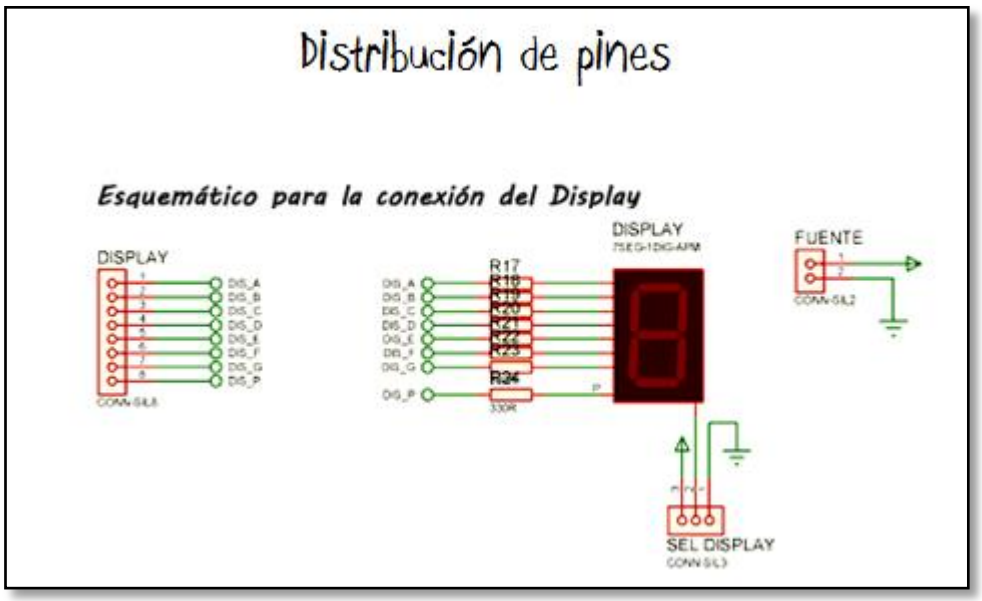
Elaborado por: Francisco Reyes y Nina Chicaiza

- Ubicación de los sócalos para controlar las entradas del dipswitch

Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

- Detalle de pines para la conexión del display



Fuente: Isis

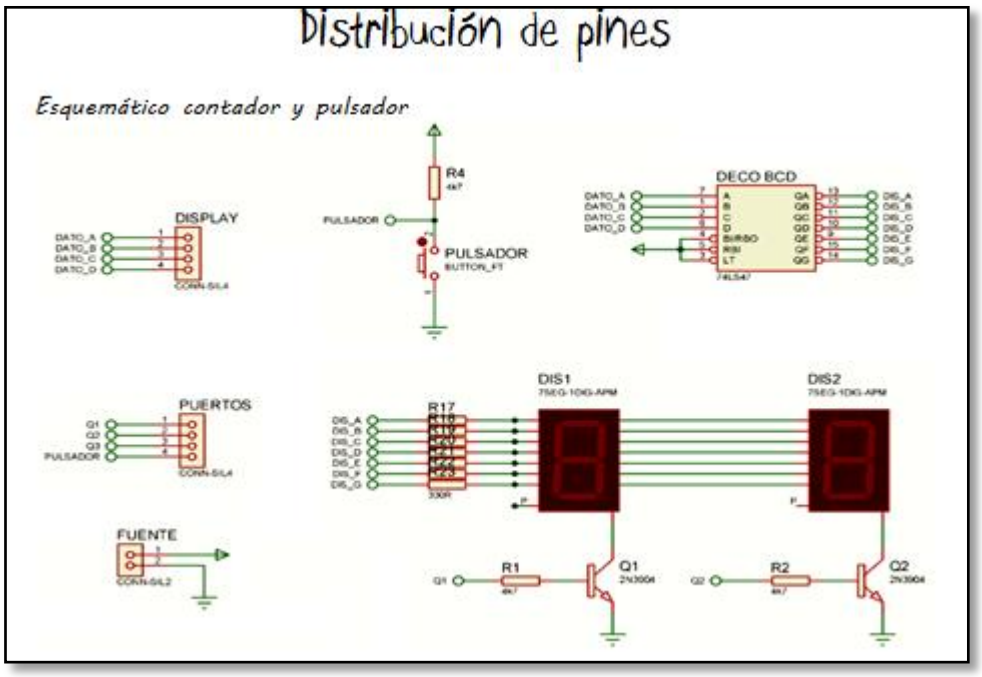
Elaborado por: Francisco Reyes y Nina Chicaiza

- Ubicación de los sócalos para conectar el display

Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

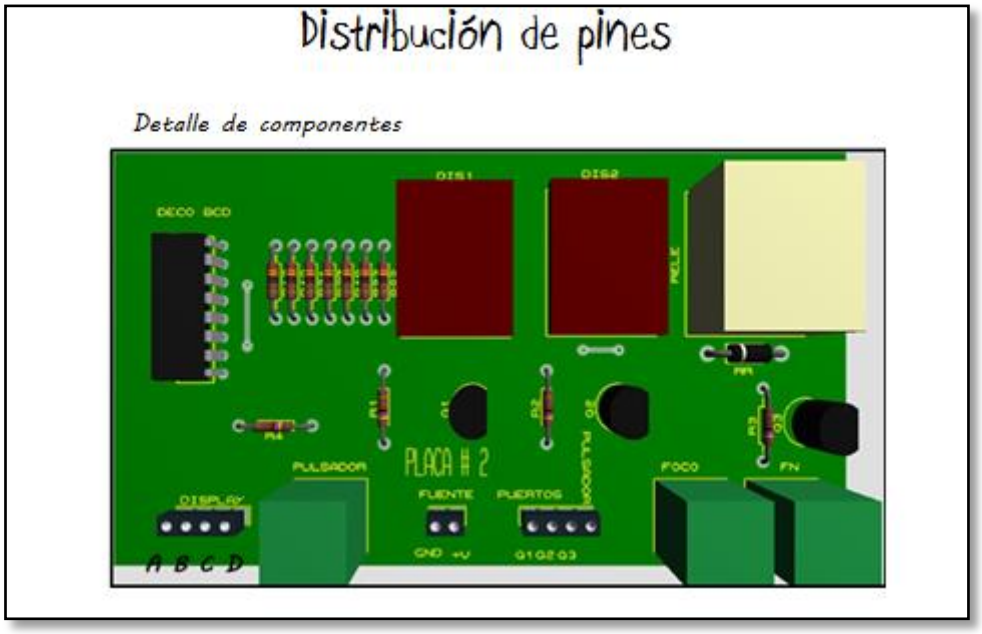
- Detalle de pines para el uso de la placa 2



Fuente: Isis

Elaborado por: Francisco Reyes y Nina Chicaiza

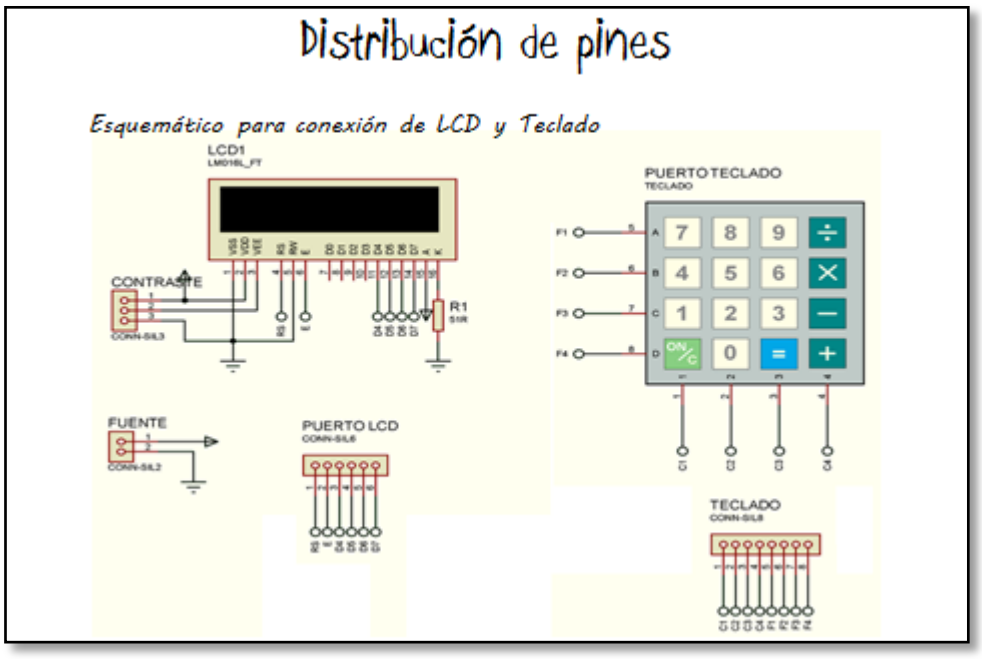
- Ubicación de los pines en la placa 2



Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

- Detalle de pines para el control de la pantalla de cristal líquido y el teclado



Fuente: Isis

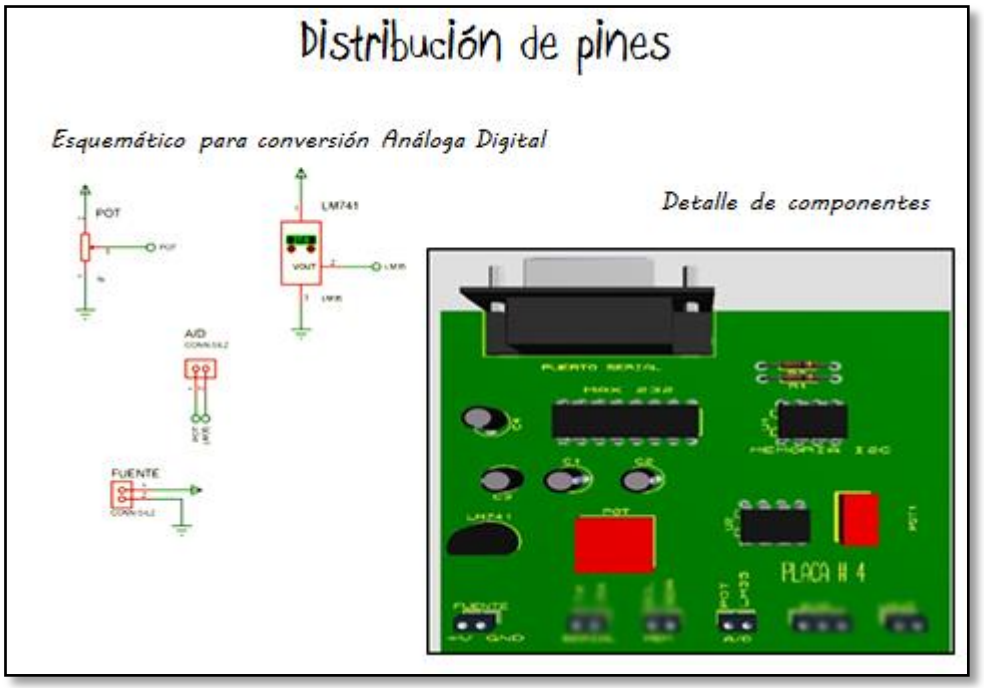
Elaborado por: Francisco Reyes y Nina Chicaiza

- Ubicación de los sócalos para controlar el lcd y el teclado

Fuente: Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

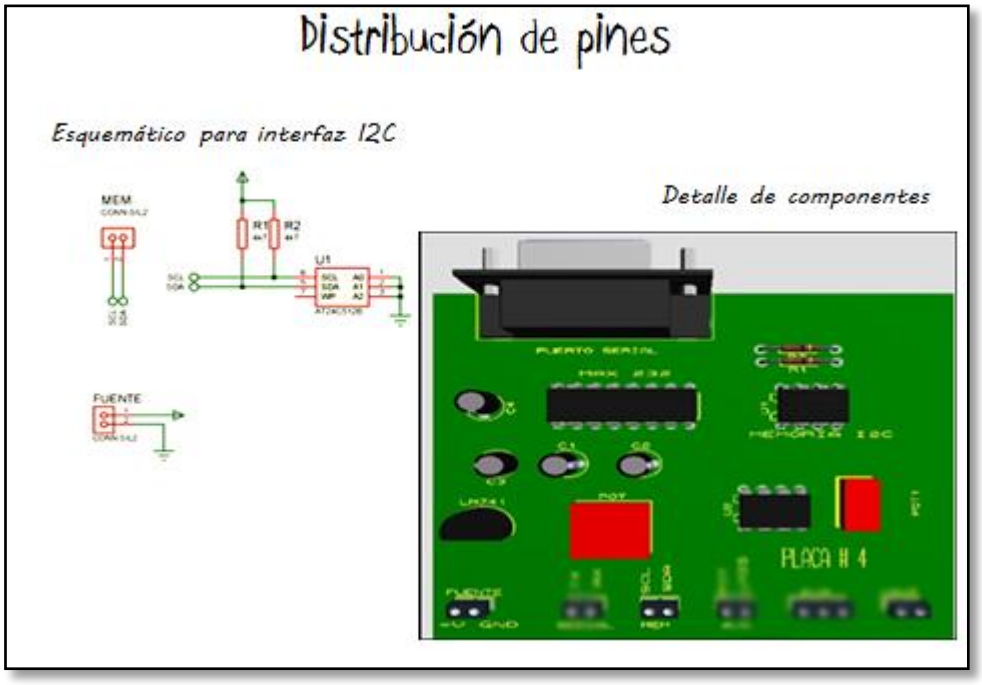
- Detalle de pines y ubicación en la placa para la práctica con el ADC



Fuentes: Isis y Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

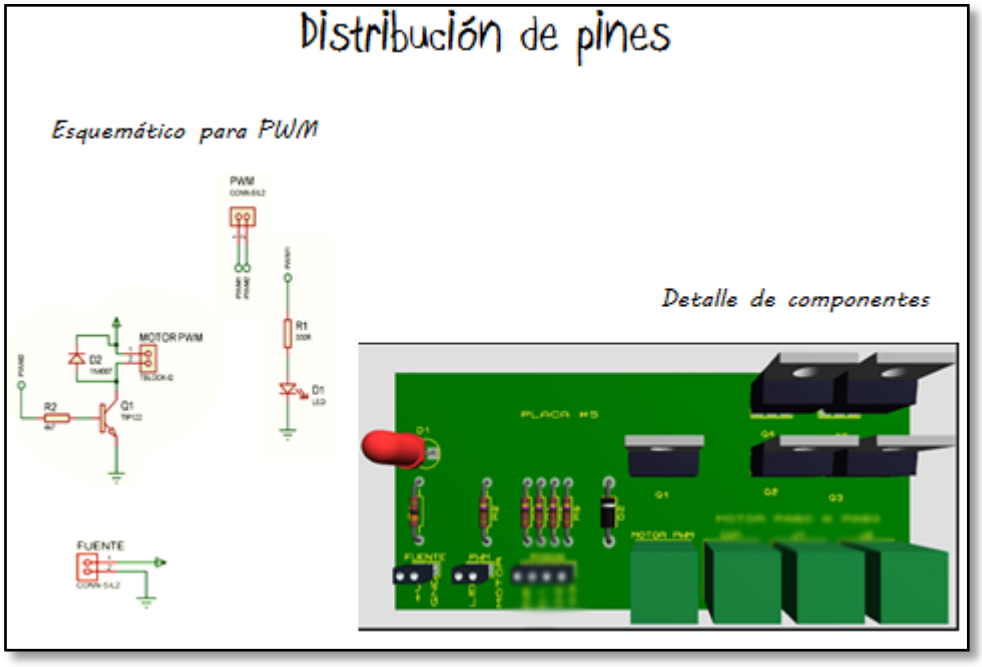
- Detalle de pines y ubicación en la placa para la práctica de memorias



Fuentes: Isis y Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

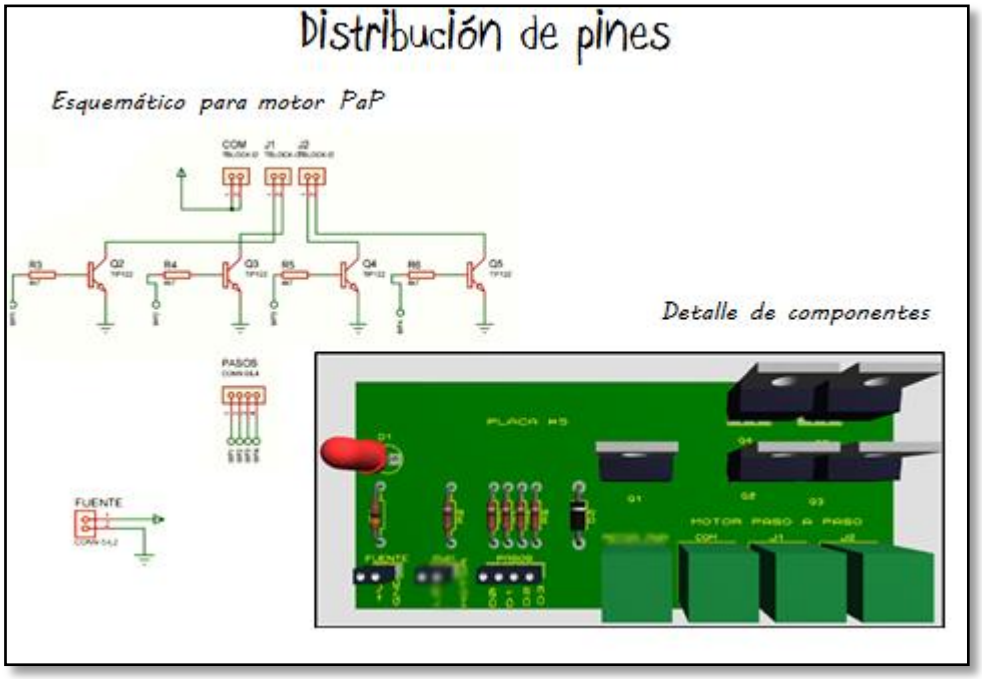
- Detalle de pines y ubicación en la placa para la práctica de PWM



Fuentes: Isis y Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

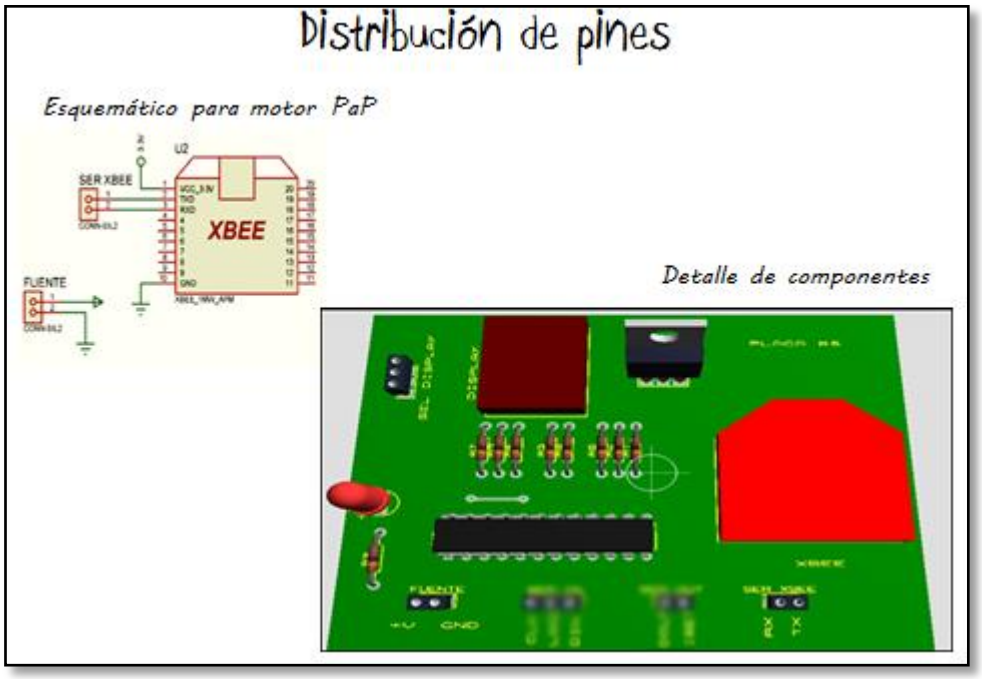
- Detalle de pines y ubicación en la placa para el control de motores PaP



Fuentes: Isis y Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

- Detalle de pines y ubicación en la placa para las prácticas de SPI e XBEE



Fuentes: Isis y Ares

Elaborado por: Francisco Reyes y Nina Chicaiza

ANEXO 18

ENCUESTAS

PREGUNTAS PLANTEADAS

1. Le resulto sencillo el manejo del módulo y sus componentes?

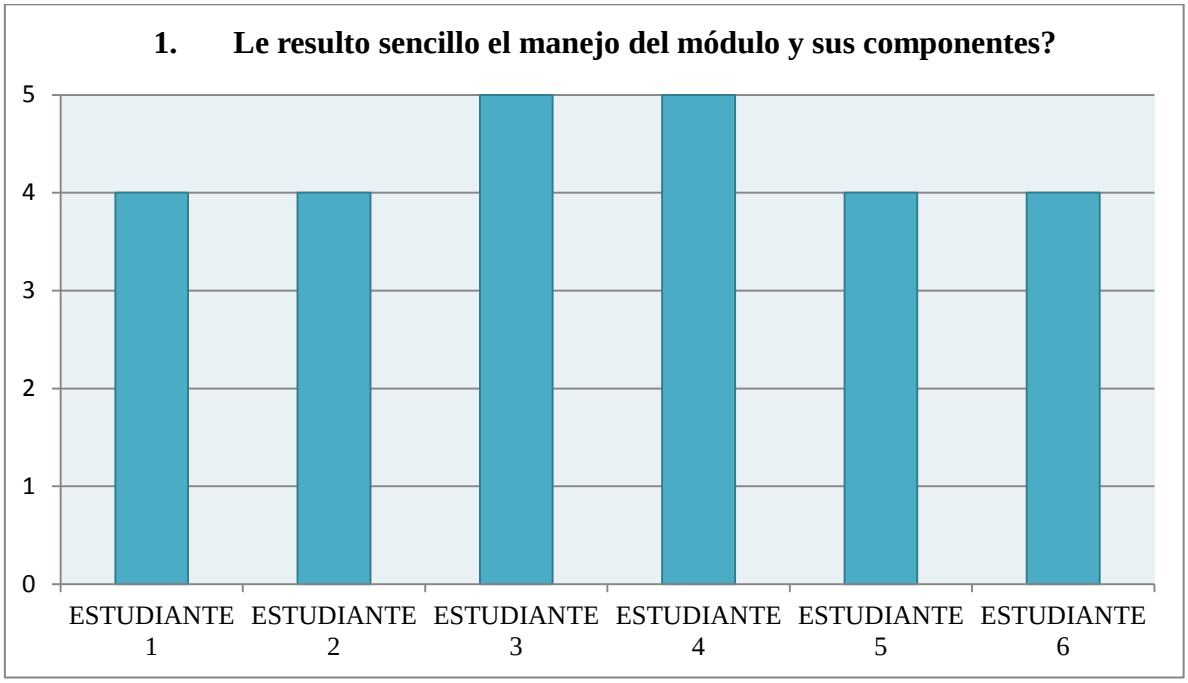
    5 = Muy satisfactorio

    4 = Satisfactorio

    3= Aceptable

    2= Deficiente

    1= Muy deficiente



2. Cree usted que este módulo le sirve para aprender manejo de microcontroladores

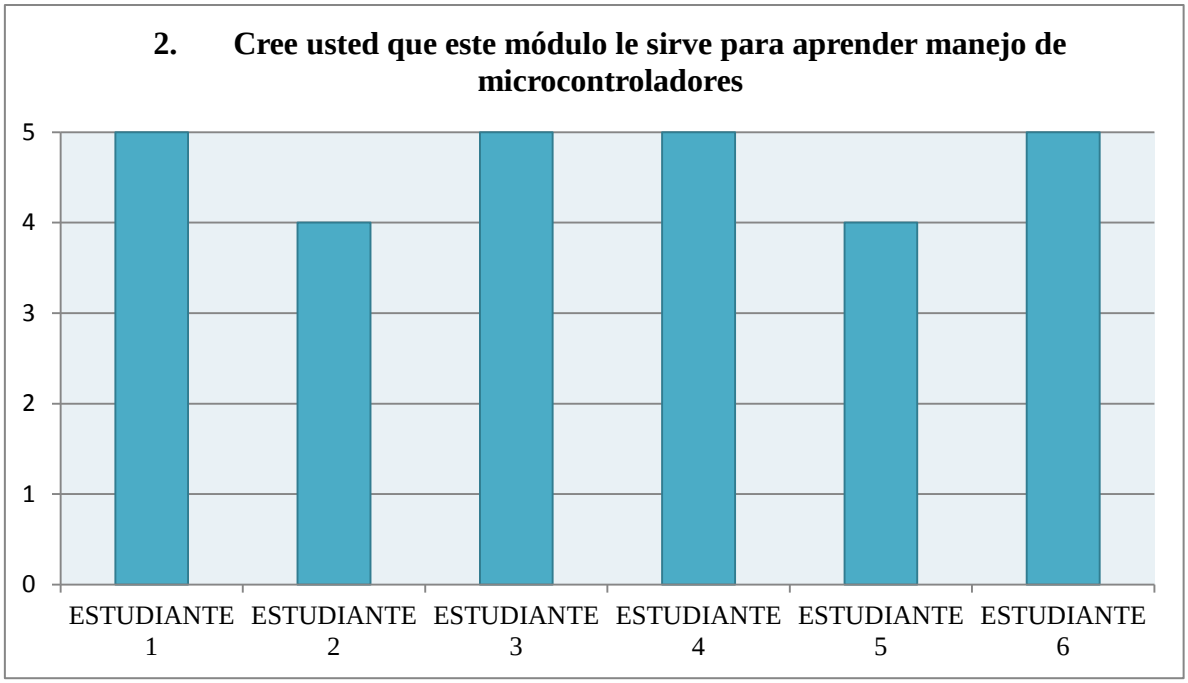
    5 = Muy satisfactorio

    4 = Satisfactorio

    3= Aceptable

    2= Deficiente

    1= Muy deficiente



3. El material que se entrega es el apropiado para realizar las practicas

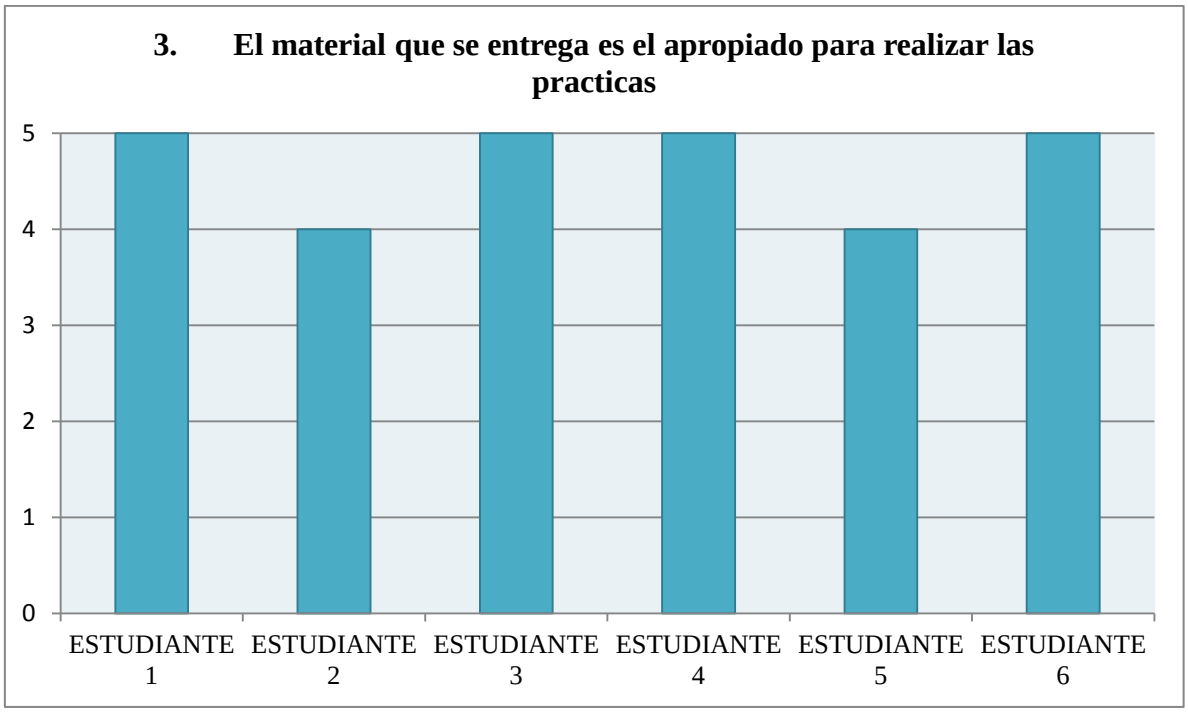
5 = Muy satisfactorio

4 = Satisfactorio

3= Aceptable

2= Deficiente

1= Muy deficiente



4. Los componentes en el módulo son robustos, es decir, se los puede maniobrar sin mucho cuidado?

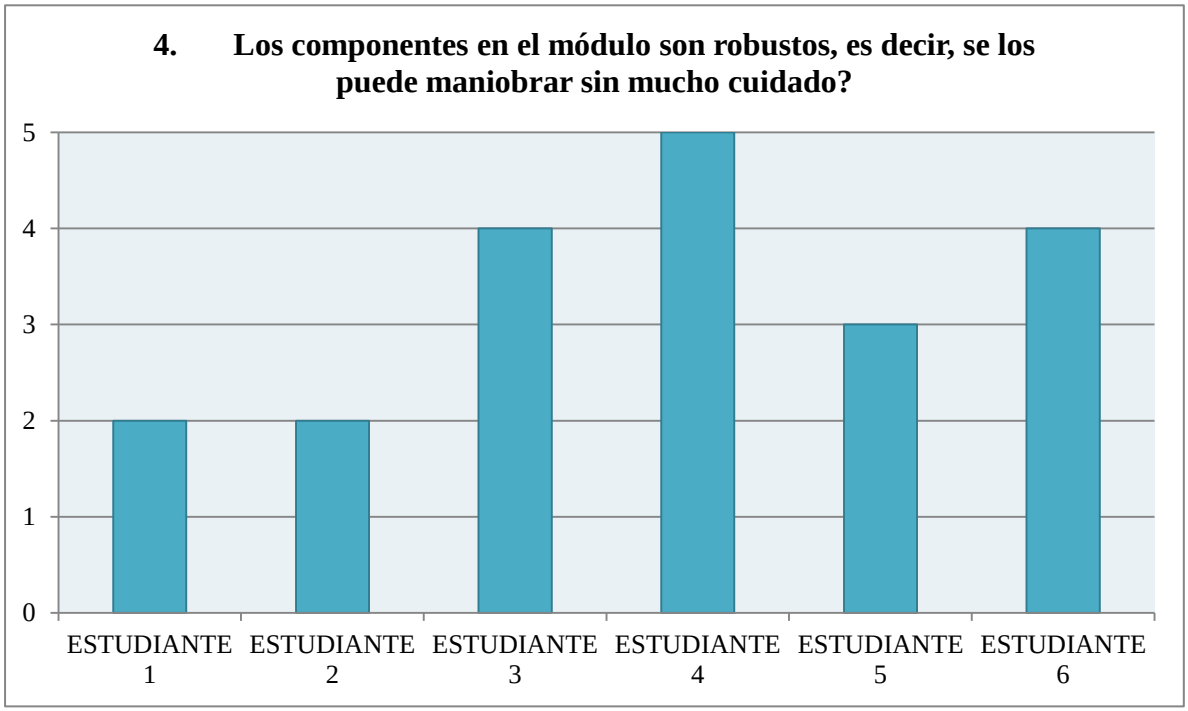
    5 = Muy satisfactorio

    4 = Satisfactorio

    3= Aceptable

    2= Deficiente

    1= Muy deficiente



5. La polarización en cada practica se la realizo sin inconvenientes

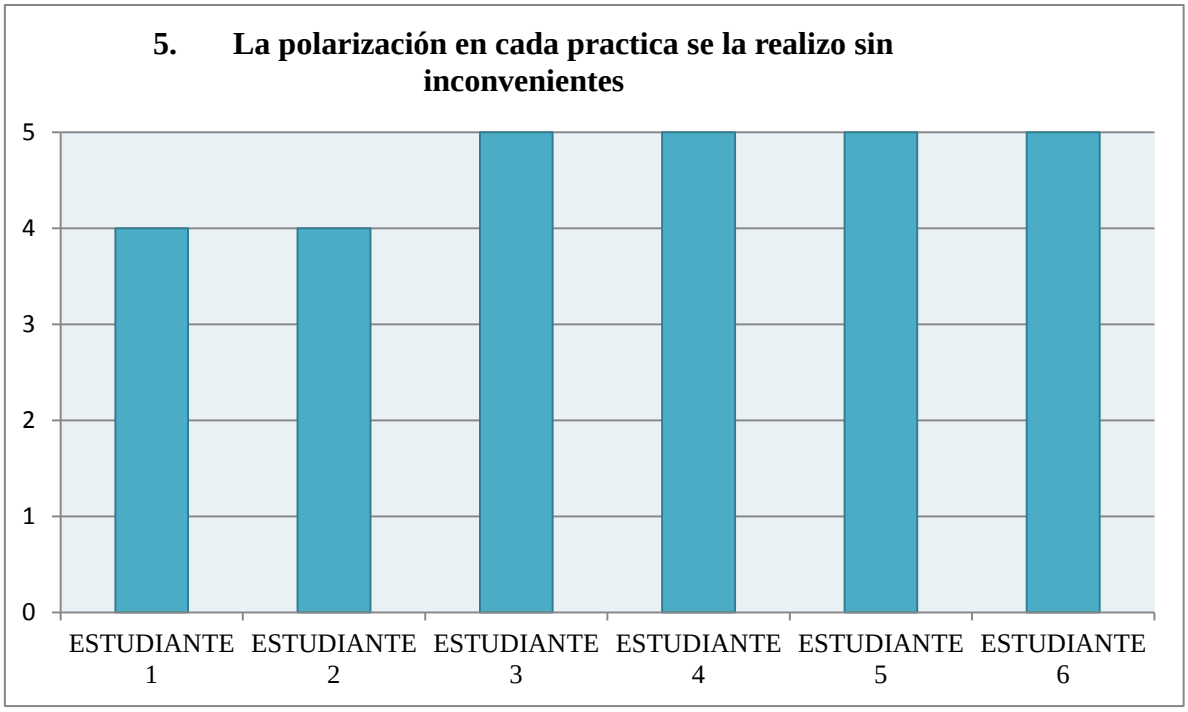
    5 = Muy satisfactorio

    4 = Satisfactorio

    3= Aceptable

    2= Deficiente

    1= Muy deficiente



6. Las prácticas que se proponen al comienzo del módulo lo preparan para las que continúan

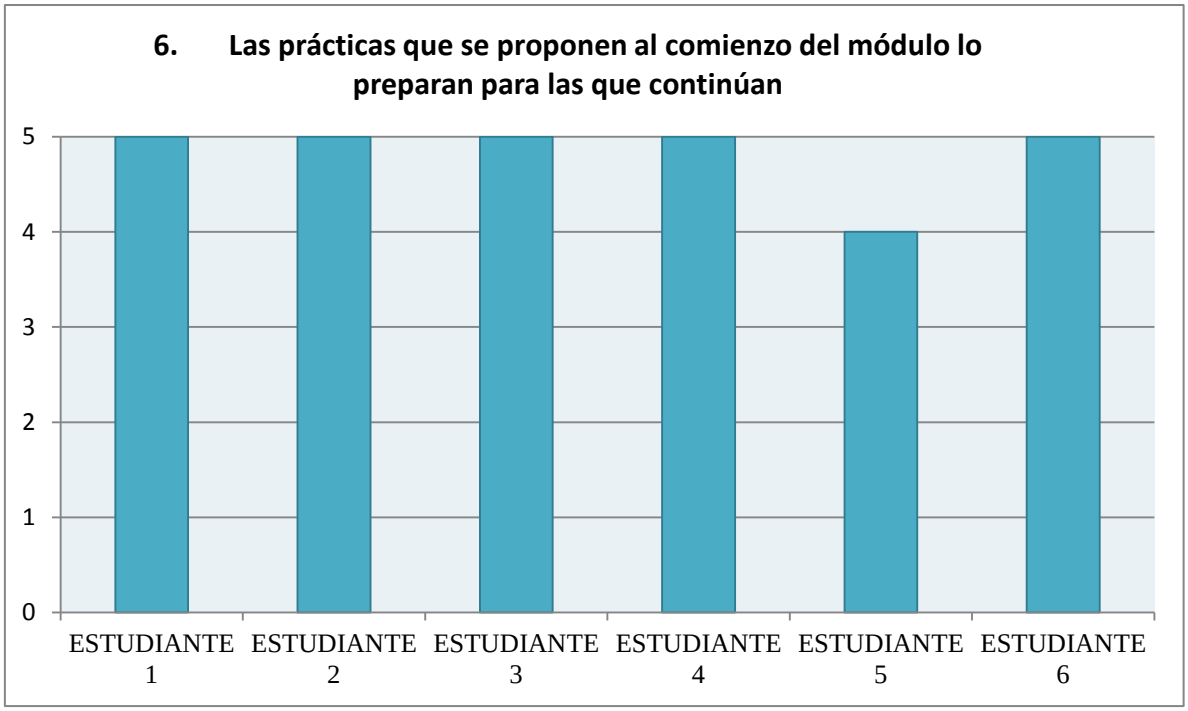
5 = Muy satisfactorio

4 = Satisfactorio

3= Aceptable

2= Deficiente

1= Muy deficiente



7. La interconexión de los periféricos se encuentra correctamente detallada

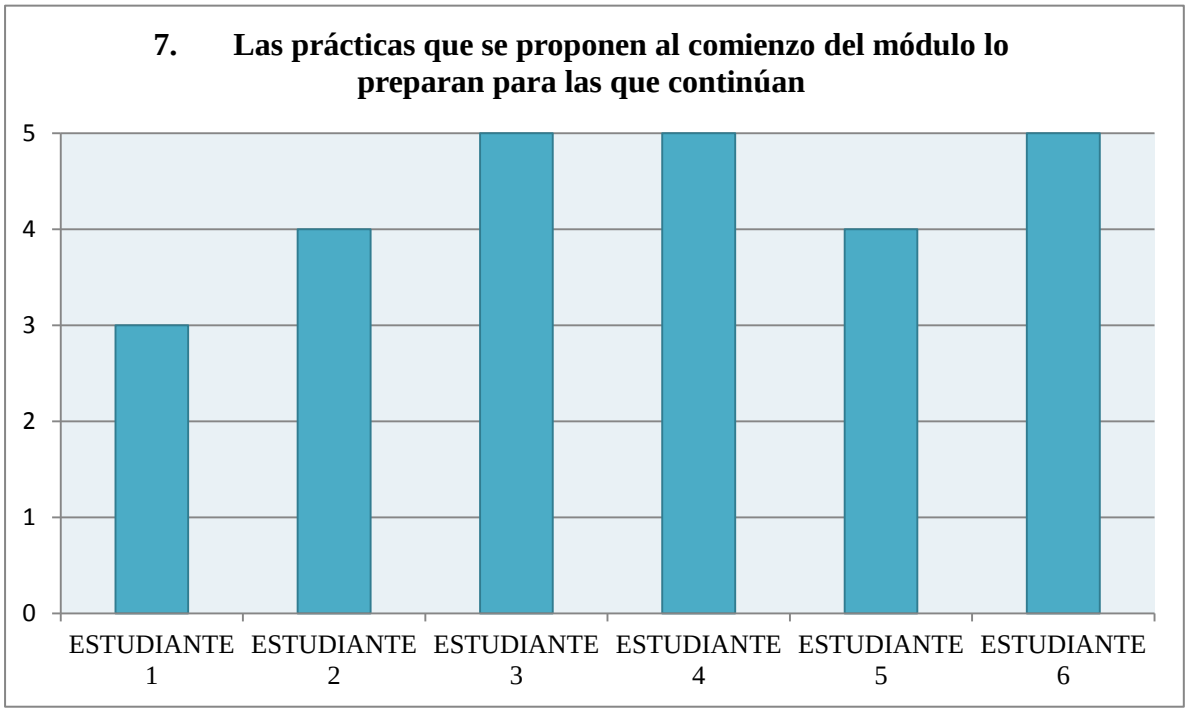
5 = Muy satisfactorio

4 = Satisfactorio

3= Aceptable

2= Deficiente

1= Muy deficiente



8. Se puede cambiar componentes del módulo que lleguen a fallar

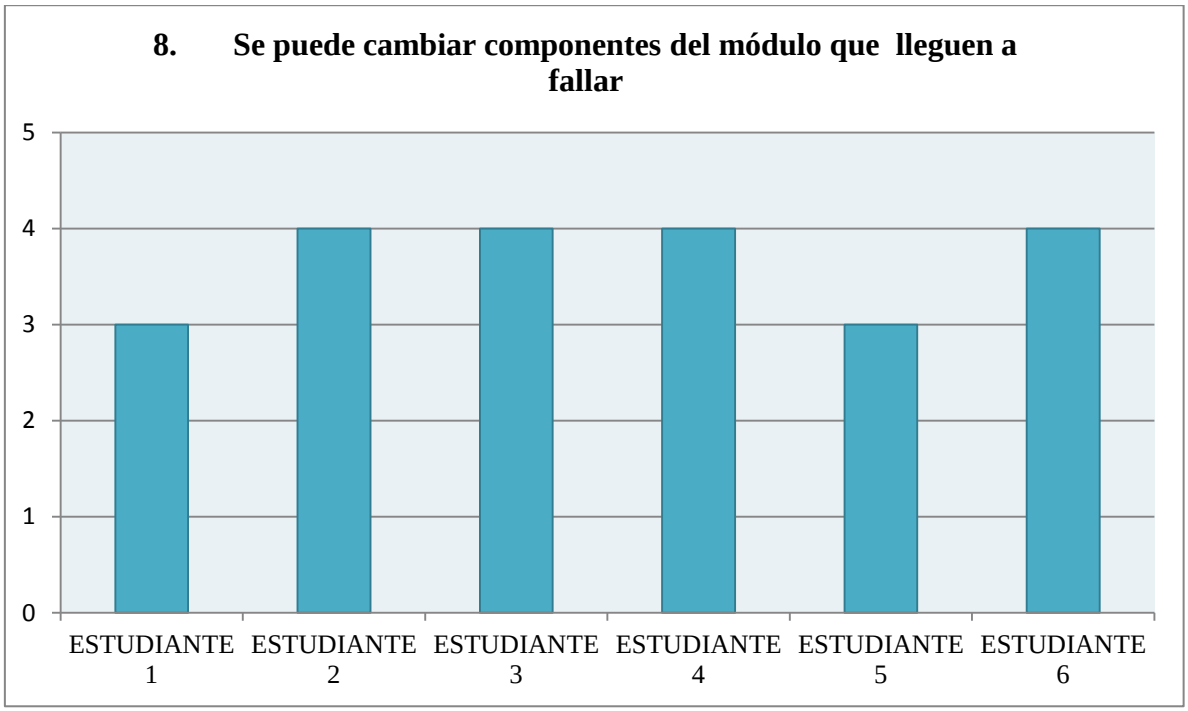
5 = Muy satisfactorio

4 = Satisfactorio

3= Aceptable

2= Deficiente

1= Muy deficiente



9. La conexión a 110v AC se realiza de manera adecuada

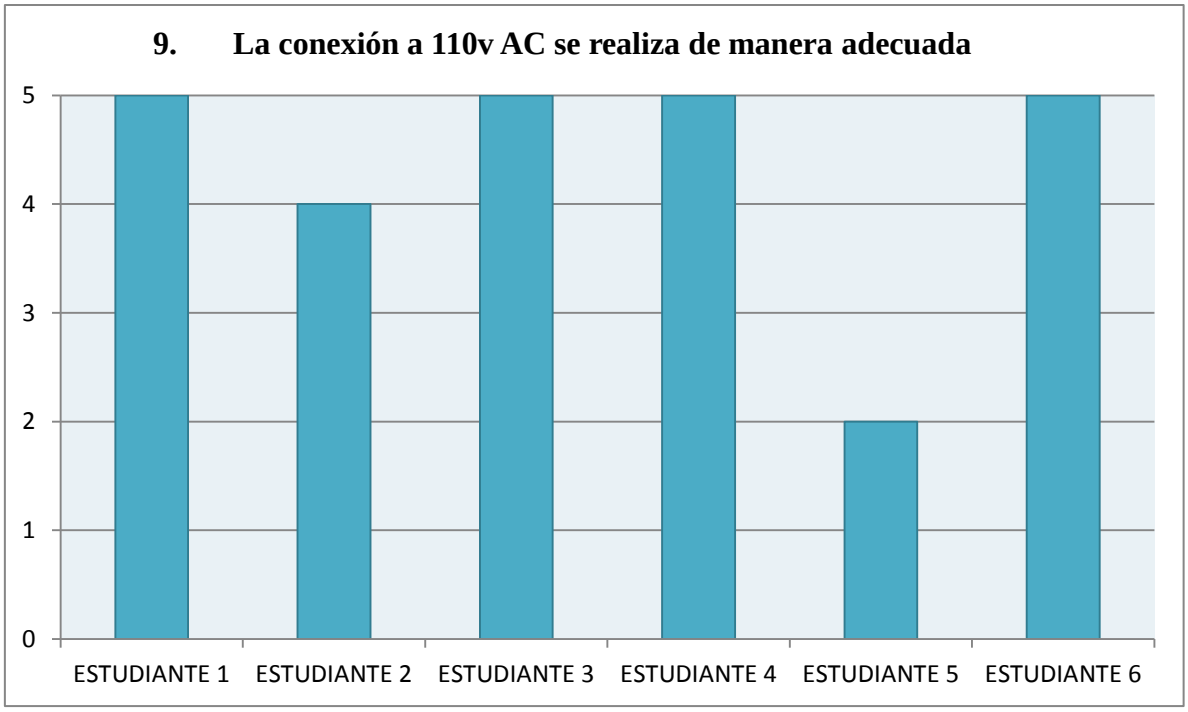
5 = Muy satisfactorio

4 = Satisfactorio

3= Aceptable

2= Deficiente

1= Muy deficiente



10. Que sugiere añadir al módulo