ESCUELA SUPERIOR POLITECNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

"Desarrollo de aplicación con sensores de temperatura usando una versión del Lenguaje JAVA llamada JAVELIN adecuada para el uso en Microcontroladores que admiten esta tecnología**"**

TESINA DE SEMINARIO

Previa a la obtención del título de:

INGENIERO EN COMPUTACIÓN ESPECIALIZACIÓN SISTEMAS TECNOLÓGICOS

INGENIERO EN ELECTRICIDAD ESPECIALIZACIÓN ELECTRÓNICA Y AUTOMATIZACIÓN INDUSTRIAL

Presentado por:

Bertha Piedad Palomeque Ávila
Rodrigo Alberto Barzola Jaramillo

GUAYAQUIL - ECUADOR
2010

AGRADECIMIENTO

A la Santísima Virgen y a Dios que me ha iluminado a lo largo de mi carrera universitaria y de mi vida.

Mi eterno agradecimiento y aprecio a mis padres y al Ing. Carlos Valdivieso, Director de Seminario de Graduación por su invaluable ayuda.

B.P.P.A.

Agradezco a Dios por darme la oportunidad de seguir en este mundo y a mis padres por el esfuerzo y el apoyo que siempre me han dado.

R.A.B.J.

DEDICATORIA

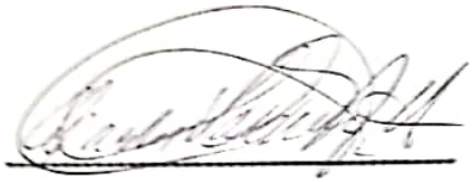
Dedico este trabajo a todas las personas que en forma directa e indirecta me apoyaron para la realización del mismo, y en especial a mis padres Beatriz Ávila y Guillermo Palomeque, mi hija Odalys, mi esposo Jorge Estrada que con su ejemplo de amor y constancia han influido en mi personalidad para poner el entusiasmo, dedicación y esfuerzo necesario a fin de culminar este logro profesional.

B.P.P.A.

Dedico este trabajo a mis padres Modesto Barzola y Elena Jaramillo quienes siempre han estado para apoyarme y me enseñaron que nunca es tarde para salir adelante, a mi hermana Verónica y a mi sobrino Omarcito, quien a pesar de su edad me enseñó que todo es posible mientras te lo propongas, y a mi novia Katty Rodríguez, a quien quiero mucho y con quién compartiré toda mi vida, ella me demuestra que a pesar de los problemas siempre hay que seguir hacia adelante y nunca hay que rendirse; todos son parte de mi vida. Los amo

R.A.B.J

TRIBUNAL DE SUSTENTACIÓN



Ing. Carlos Valdivieso A.

PROFESOR DE SEMINARIO DE GRADUACIÓN

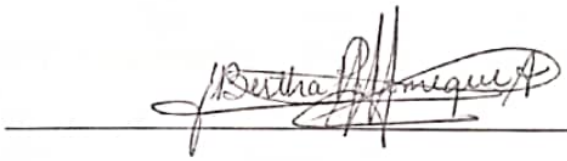
Ing. Hugo Villavicencio

DELEGADO DEL DECANO

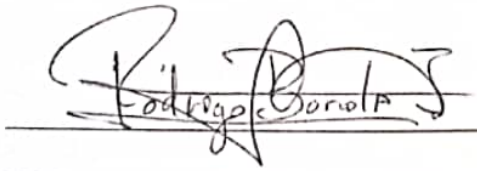
DECLARACION EXPRESA

"La responsabilidad del contenido de esta Tesina de Seminario, me corresponden exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITECNICA DEL LITORAL".

(Reglamento de Graduación de la ESPOL).



BERTHA PIEDAD PALOMEQUE AVILA



RODRIGO ALBERTO BARZOLA JARAMILLO

RESUMEN

El proyecto de tesis se divide en cuatro partes para una mejor comprensión.

Brevemente se presenta el sensor de temperatura y algunos conceptos de Java con el que se desarrolló la programación e implementación del sensor de temperatura.

En el capítulo I se detalla la descripción general del sistema, alcance, limitaciones, descripción específica del sensor de temperatura, ventajas del Javelin Stamp en comparación con otros microcontroladores.

El capítulo II consta de las conexiones que tiene la tarjeta Javelin Stamp, características del módulo Javelin Stamp, componentes, conector RS-232, conector Jack hembra, SP237ACT, descripción del módulo Javelin Stamp, sensor DS1620.

El capítulo III a fin de establecer un conocimiento del código desarrollado, se realiza el análisis del código de programación en Javelin Stamp, el esquemático del hardware del módulo Javelin Stamp, figuras explicativas de

la programación e implementación del sensor de temperatura como parte del desarrollo de nuestro proyecto.

Finalmente de lo desarrollado obtenemos el sensor de temperatura se realizan pruebas y simulaciones con lo cual se obtienen resultados que se presentan a través de ilustraciones, pruebas con el integrado DS1620, y un tutorial de la programación orientada a objetos Java.

INDICE GENERAL

Pág.

INDICE DE FIGURAS

Pág.

INDICE DE TABLAS

Pág.

INDICE DE ANEXOS

Pág.

CAPITULO 1
GENERALIDADES

1.1 Descripción General del Sistema.

El presente proyecto aborda el diseño e implementación de un sensor de temperatura con el módulo Javelin Stamp que viene incluido en el Javelin Stamp Starter Kit, el módulo Javelin Stamp es un sistema con memoria EEPROM, flash, fuente e interfaz, además de ser pequeño pero potente que hace uso del lenguaje de programación Java.

Para la elaboración del sistema, se realiza la codificación en Java mediante un bloque de instrucciones de programa para propósitos específicos (firmware), grabando en memoria tipo no volátil las instrucciones de código que se utilizan para el Javelin Stamp y luego son ejecutados desde una SRAM paralela.

Para escribir el código se utiliza el Entorno de Desarrollo Integrado (IDE) que sirve para la compilación del programa realizado en el lenguaje de programación Java, el Javelin Stamp IDE tiene un terminal virtual que permite enviar y recibir mensajes desde el módulo Javelin Stamp, la misma que viene incorporada con métodos que se utilizan para codificar mediante librerías.

Para el desarrollo del proyecto se utiliza el integrado DS1620 que permite sensar y almacenar temperaturas máxima y mínima leídas, este módulo tiene pines programables para los valores medidos a través de una simple interfaz RS-232. En nuestro proyecto, la tarea principal del módulo es la de regular el tráfico de los datos de temperatura con el sensor.

1.2 Alcance del Sensor de Temperatura.

El trabajo realizado en este proyecto se orienta a la elaboración de un programa que permita simular un sensor de temperatura empleando el módulo Javelin Stamp, además la implementación del esquema de dispositivos para su funcionalidad.

1.3 Limitaciones del proyecto Sensor de Temperatura

El componente DS1620 es un integrado que trabaja con un rango de temperatura, desde -55 a 125 ºC, sin embargo el proyecto no puede ser colocado a temperaturas extremas ya que todos sus componentes y conexiones se encuentran al medio ambiente.

La tarjeta Javelin Stamp se debe manipular con cuidado, debido a que las conexiones de elementos externos a la tarjeta (DS1620, cables puentes, elementos pasivos) no están incorporados en este hardware.

1.4 Descripción específica del Sensor de Temperatura.

Este proyecto tiene como objetivo, la programación e implementación de un sistema para medir temperatura usando el microcontrolador avanzado Javelin Stamp.

El proyecto consta de dos partes, una de software y otra de hardware. En el software se implementarán rutinas donde el Javelin Stamp tiene como base el lenguaje de programación Java.

El lenguaje Java nos da la posibilidad de tener una amplia gama de recursos y librerías para facilitarnos el desarrollo del software.

El hardware vendrá complementado por las diferentes conexiones que tiene el microcontrolador, además admite la comunicación vía serial (RS-232), permitiéndonos transmitir información hardware –PC, PC – hardware.

La arquitectura del hardware facilita las conexiones de los elementos pasivos y sensores para la implementación de proyectos.

1.5 **Ventajas del Javelin Stamp.**

La tarjeta fue diseñada de acuerdo a las necesidades de algunos usuarios que enseñan a conectar y programar microcontroladores, ya que permite de una forma rápida y

cómoda la conexión de la PC con las señales del módulo Javelin Stamp.

El lenguaje de programación Java es una de las razones principales que hace al módulo Javelin Stamp único y diferente a los microcontroladores típicos y lo convierten en un módulo potente.

CAPITULO 2

2. FUNDAMENTACIÓN TEÓRICA

2.1 Starter Kit Javelin Stamp

Para iniciar con una exitosa programación en Java el Starter Kit tiene todo lo que necesita para la realización de proyectos, la tarjeta Javelin Stamp tiene conexiones para puerto RS-232, servomotores, miniprotoboard, bloque de conectores para la comunicación entre el microcontrolador y elementos pasivos e integrados, además un bloque adicional de conectores para realizar la comunicación serial a periféricos externos.

Este proyecto se identifica con el módulo de Javelin Stamp que es un pequeño pero potente sistema que ayuda a la comunicación con el integrado DS1620, mismo que lee los datos, los transforma y envía en valores de temperatura hacia el módulo Javelin.

Para la programación es primordial el uso del software del Javelin Stamp, que interactúa directamente con la tarjeta mediante una conexión serial, reconoce automáticamente el hardware al ejecutar las instrucciones del fichero.

2.2 Características del Módulo Javelin Stamp

- La memoria RAM es de 32 k bytes de espacio de programa (el espacio sobrante puede utilizarse para variables, matrices.
- El buffer del Serial es de 32 bytes de memoria EEPROM no volátil.
- Búfer de comunicación en serie en el fondo hasta siete objetos UART que pueden comunicarse de forma independiente el uno del otro y el programa principal.
- Modulación de ancho de pulso en segundo plano.
- Velocidad de ejecución de instrucciones de 8000 instrucciones por segundo (sin incluir los procesos en segundo plano que se ejecutan independientemente de las tareas de primer plano).

2.3 Componentes de Starter Kit

El Starter Kit del Javelin Stamp consta de varios componentes que se van a presentar a continuación. Cada uno cumple un funcionamiento y acoplados son una herramienta robusta para desarrollar programación Java, adicionalmente tiene dispositivos propios de Parallax.

2.3.1 Starter Kit Javelin Stamp

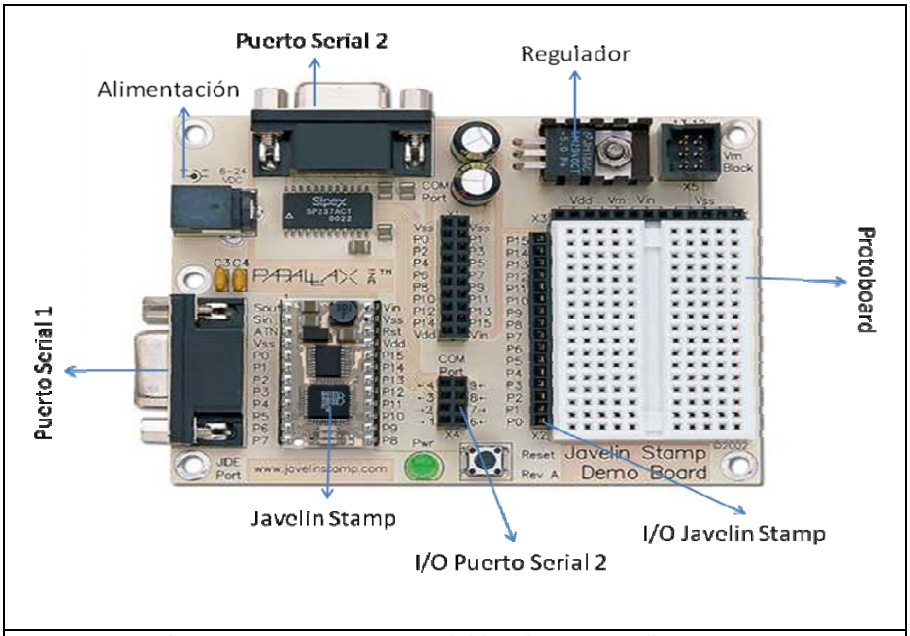


Figura 2.1 Componentes del hardware Javelin Stamp.

2.3.2 Conector RS-232 (comunicación serial)

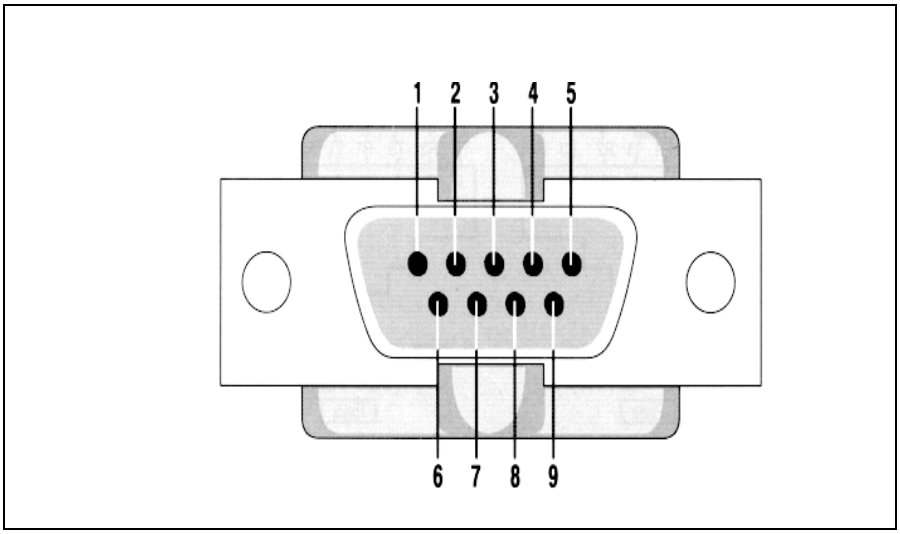


Figura 2.2 Conector RS-232 para comunicación serial.

Conector RS-232 el cual me permite comunicarme de manera serial entre equipos

Tabla I. Pines del conector RS-232

1. Detección de portadora de datos	2. Datos recibidos
3. Datos transmitidos	4. Terminal lectura de datos
5. Señal de tierra	6. Datos listos para lectura
7. Requerimiento para enviar	8. Limpio para enviar
9. Indicador	

Teniendo en cuenta que la comunicación entre la PC y el Javelin Stamp es de conexión punto a punto.

2.3.3 Conector Jack Hembra (Alimentación)

Figura 2.3 Componente Jack Hembra

Este dispositivo permite tener una conexión de voltaje de manera segura, ya que mantiene la línea de voltaje y tierra totalmente aislado.

2.3.4 Regulador de Voltaje (Regulador 5V)

Figura 2.4 Regulador de 5v.

Componente dedicado a establecer un voltaje constante dentro del circuito, con el cual alimento al protoboard, sin tener el peligro de que el voltaje con que está alimentada la tarjeta afecte a otros elementos, también llamado estabilizador de voltaje o acondicionador de voltaje.

Es un equipo eléctrico que acepta una tensión eléctrica de voltaje variable a la entrada, dentro de un parámetro predeterminado y mantiene a la salida una tensión constante (regulada).

Cuando el campo de regulación es insuficiente podemos fabricar un equipo con un rango adecuado a la necesidad.

Para este caso es necesario monitorear o graficar la línea de alimentación para determinar los límites máximo y mínimo de variación de la línea.

2.3.5 SP237ACT (Comunicación Serial)

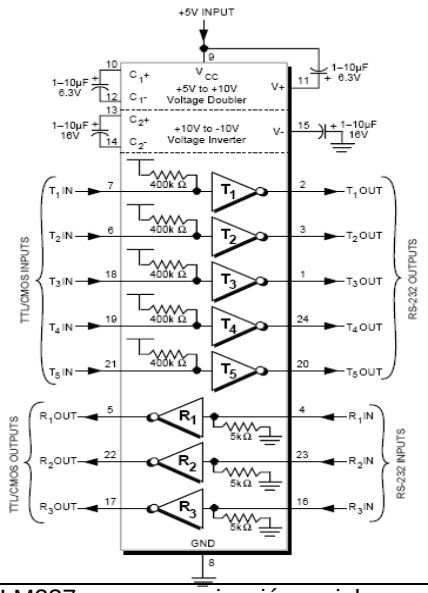


Figura 2.5 Integrado LM237 para comunicación serial

El integrado LM237 me permite comunicar el hardware a cualquier equipo periférico, a través de la comunicación serial con

la ayuda del conector RS-232.

2.4 Descripción del Starter Kit Javelin Stamp

2.4.1 Descripción del Módulo Javelin Stamp.

El Javelin Stamp es uno módulo de 24 pines y este microcontrolador es programado en el lenguaje de programación Java.

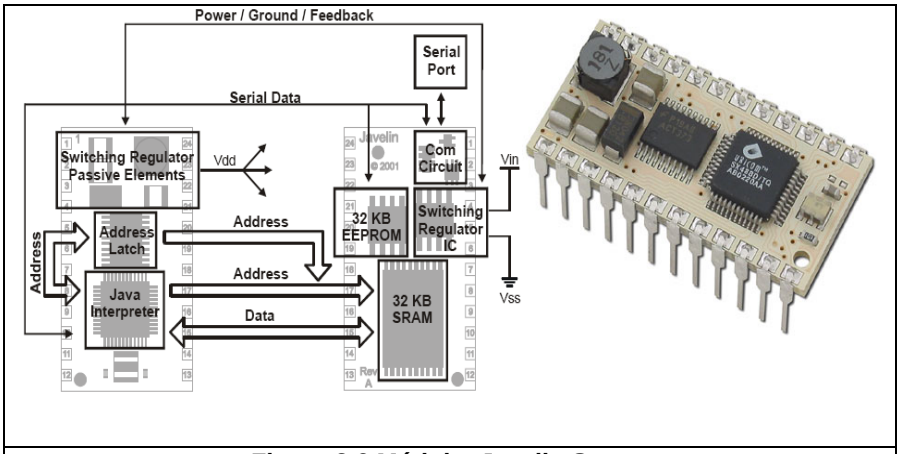


Figura 2.6 Módulo Javelin Stamp

Dentro de la arquitectura podemos encontrar varios componentes que permiten el trabajo del Javelin Stamp, en la Figura 2.6 podemos observar en la parte izquierda la parte superior del Javelin y en la parte derecha la inferior del Javelin.

El conector de comunicación, está en la parte inferior de la tarjeta Javelin Stamp. A través de este conector el Javelin puede comunicarse con la PC por el puerto serial RS-232, y así utilizar este medio para las descargas del programa y depuraciones.

La ventaja de tener dos memorias es que permite utilizar la memoria SRAM para realizar el trabajo con el Javelin de una manera de comunicación paralela, poniendo en segundo plano la comunicación serial mediante la memoria EEPROM.

El bloque del regulador conmutado, alimentados con voltaje de 6 a 24 VDC, y los elementos pasivos dentro del modulo Javelin Stamp, en conjunto me permiten regular el voltaje y mantenerlo constante en 5 VDC.

2.4.2 Descripción del integrado DS1620.

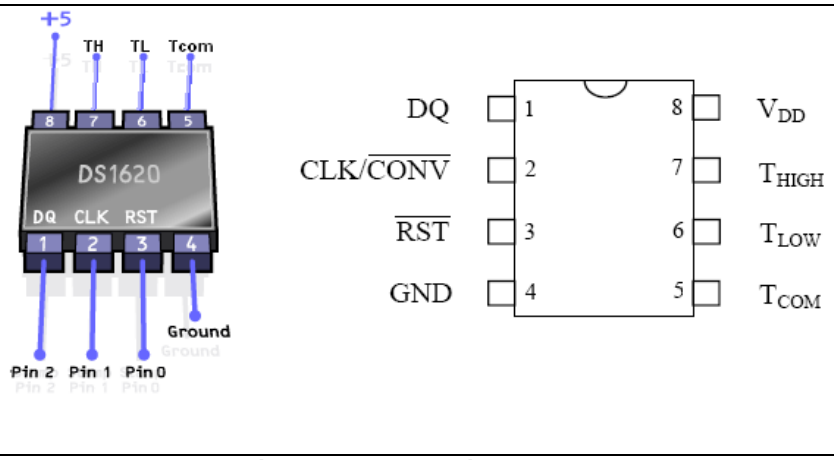


Figura 2.7 Integrado DS1620

El DS1620 termómetro y termostato dispone de 9 bit para la lecturas que indican la temperatura del dispositivo. Con tres salidas de alarma térmica, el DS1620 también puede actuar como un termostato:

T_{ALTA} está impulsado alto si la temperatura del DS1620 es mayor o igual a una temperatura definida por el usuario TH.

T_{BAJA} está impulsado alto si la temperatura del DS1620 es inferior o igual a una temperatura definida por el usuario TL.

Con lo cual TH y TL son utilizadas como alarmas permanentes y solo pueden ser reseteados por software o retirando la alimentación al integrado.

T_{COM} es impulsada alta cuando la temperatura supera los TH y se mantiene alta hasta que la temperatura cae por debajo de la de TL. Con lo cual es muy útil para el uso y funcionamiento del método de termostato.

Ajustes de temperatura definido por el usuario se almacenan en no volátil, así que las piezas pueden ser programados antes de la inserción en un sistema, así como se utiliza en aplicaciones independientes sin una CPU. Los ajustes de temperatura y temperatura son comunicados desde / hasta el DS1620 través de una interfaz sencilla de 3 cables.

Según la figura 2.7 podemos denotar los pines del DS1620 permitiéndonos identificar la conexión para la buena operación del sensor de temperatura.

CAPITULO 3

3. DISEÑO DE SOFTWARE Y HARDWARE

3.1 Análisis del Código de programación en Javelin Stamp

El código del programa escrito en el IDE del Javelin Stamp tiene como objetivo la comunicación entre las instrucciones en Java con el integrado DS1620 que permite enviar y recibir datos de temperatura.

El desarrollo de la codificación tiene como librería principal **core**, esta nos permite la entrada y salida de valores de temperatura entre el módulo Javelin Stamp y el DS1620 usando los métodos shiftIn () y shiftOut ().

El método dsTemp () es importante porque usa las funciones antes mencionados shiftIn () y shiftOut () para la comunicación bidireccional con el DS1620.

Para iniciar la comunicación entre el módulo Javelin y el DS1620 primero se debe setear en true el pin de activación (EnablePin), con el fin de que permita el envío y recepción de los datos. El valor hexadecimal AA reporta la temperatura, y es enviada para el DS1620 usando el método shiftOut (). Lo siguiente que se realiza es que la variable data setea al método shiftIn ().

Aquí también utilizamos el CPU que contiene llamadas específicas para ayudar a manejar los recursos del Javelin Stamp (hardware), en los incluyen pines de entradas y salidas.

Todos los miembros del CPU son estáticos, entonces no se necesita crear una instancia del objeto; simplemente se llama a los miembros como se necesite según la programación.

3.2 Programa realizado en Javelin

```
import stamp.core.*;
public class ProyectoTemp
{
  //----------- declaración de  los pines I/O  para DS1620

final static int dataPin = CPU.pin15;
  final static int clockPin = CPU.pin14;
  final static int enablePin = CPU.pin13;

  //----------- Inicio de caracteres usado para colocar el cursor en la
  ventana de  mensajes de javelin
  final static char HOME = 0x01;

  //---------- códigos para la inicialización DS1620 y la medición de
  temperatura
  final static int WRITE_CONFIG = 0x0C;
  final static int WRITE_TL = 0x02;
  final static int START_CONVERT = 0xEE;
  final static int READ_TEMP = 0xAA;
```

```
static int DSValue, sign, i, data;
static int[] setup = {WRITE_CONFIG,WRITE_TL,START_CONVERT};
  //--------Usando un lazo, el método dsInit (abajo) los valores de los
  accesos //--------de la matriz de configuración
  //-------- El comando SHIFTOUT da el reloj de cada valor en el
DS1620.
  static void dsInit(int config[])
  {
    CPU.writePin(enablePin,false);
    CPU.delay(10);
    for (int i = 0; i < config.length; i++)
    {
      CPU.writePin(enablePin,true);
      CPU.shiftOut(dataPin,clockPin,8,CPU.SHIFT_LSB,config[i]);
      CPU.writePin(enablePin,false);
    }
  }
//-------- El método dsTemp acepta comandos de la rutina principal y
usos
//-------- La SHIFTOUT () para enviar este valor en el DS1620.
Entonces el   //-------- SHIFTIN ()
//-------- Método se utiliza para cambio en los datos de temperatura del
```

```
//-------- DS1620.El Valor positivo o negativo se devuelve a la rutina
principal
  static int dsTemp(int command)
            {
              CPU.writePin(enablePin,true);
             CPU.shiftOut(dataPin,clockPin,8,CPU.SHIFT_LSB,command);
    data =
((CPU.shiftIn(dataPin,clockPin,9,CPU.POST_CLOCK_LSB)>>7));
    CPU.writePin(enablePin,false);
    sign = data >> 8;
  if ( sign == 1 )
  {
    return -data;
  }
  else
  return data;
}
//-------- La rutina principal llama al método dsInit para inicializar el
DS1620,
//-------- Luego se pone el valor de temperatura a partir del método
dsTemp y //--------lo muestra.
```

```
  public static void main()
  {
    dsInit(setup);
    while (true)
    {
      System.out.print(HOME);
      System.out.println ("Celsius temperature: ");
      System.out.println(dsTemp(READ_TEMP)/2); //div para 2 para
grados ºC
      CPU.delay(5000);
    }
  }
}
```

3.3 Esquema del hardware del módulo Javelin Stamp.

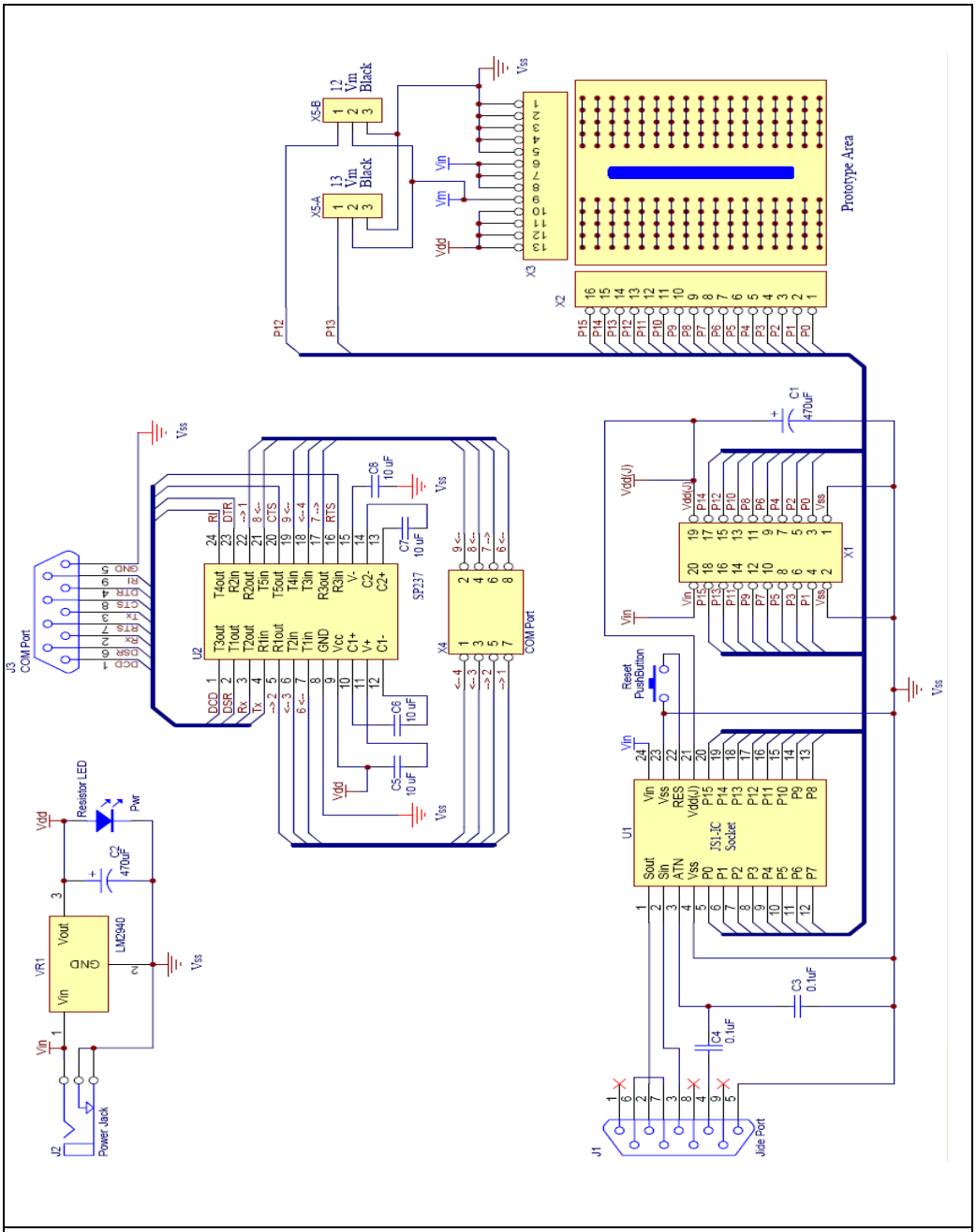


Figura 3.1 Esquemático de conexiones del Javelin Stamp.

3.4 Ilustraciones de Java y Javelin Stamp.

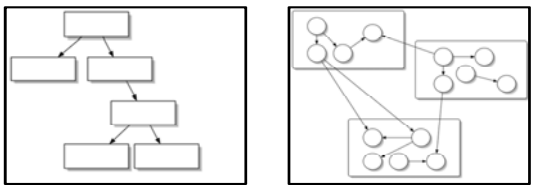


Figura 3.2 Programación Orientada a Objetos vs Estructurado

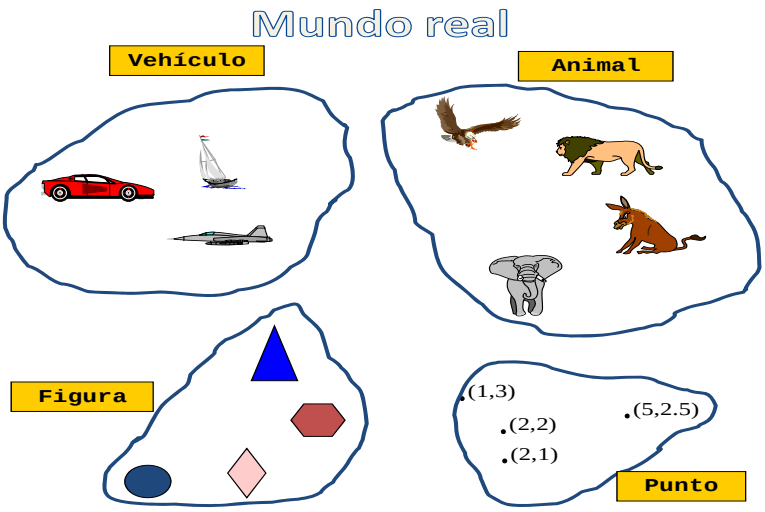


Figura 3.3 Abstracción de Objetos del Mundo Real

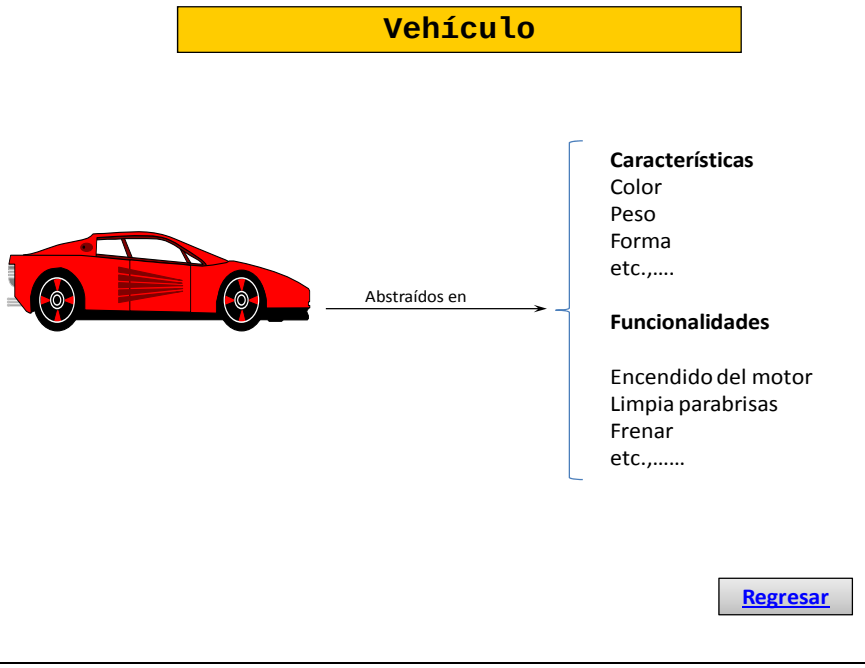


Figura 3.4 Ejemplo de Abstracción de un Objeto

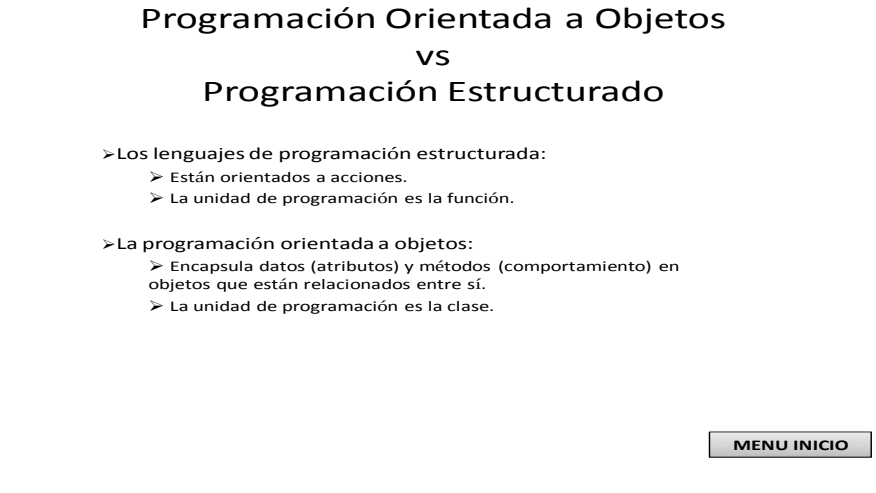
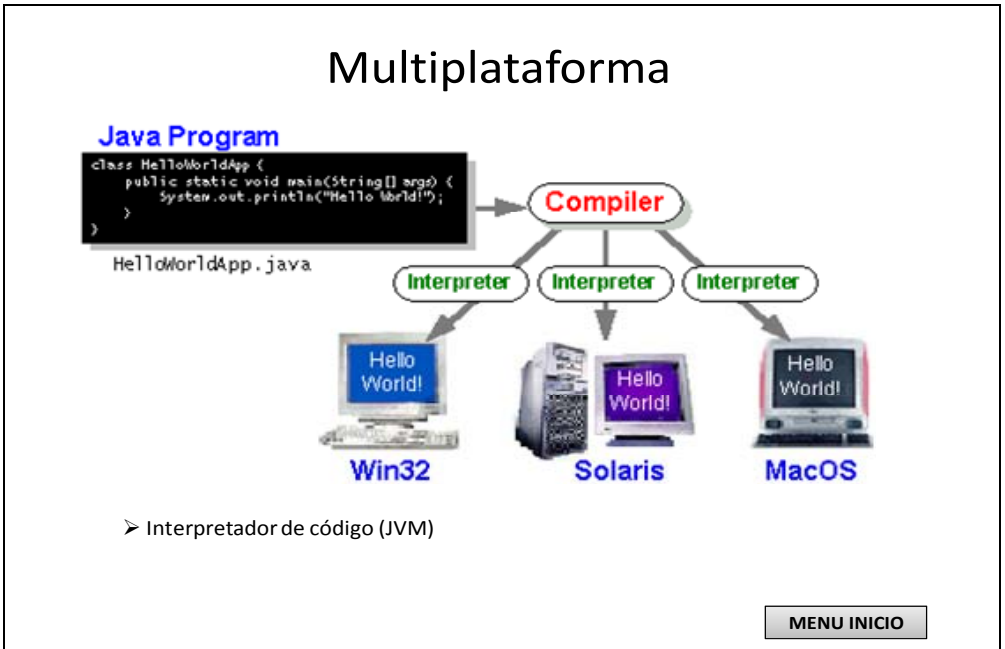


Figura 3.5 Diferencias entre Lenguaje Estructurado y Orientado a Objetos



Fi gura 3.6 Plataformas en las que puede trabajar Java.

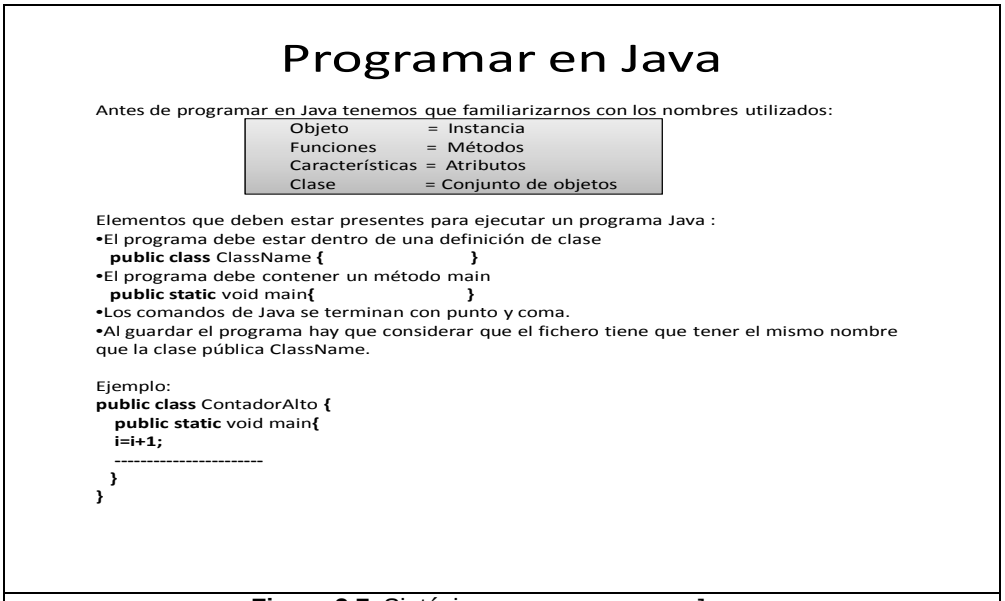


Figura 3.7 Sintáxis para programar en Java.

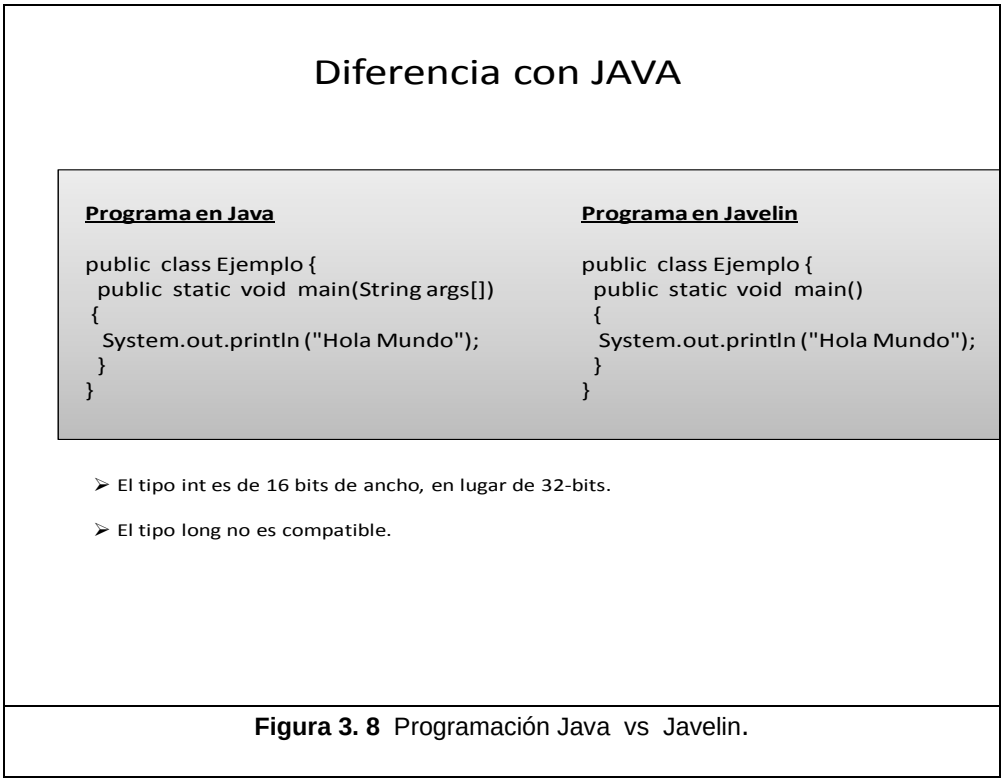


Figura 3. 8 Programación Java vs Javelin.

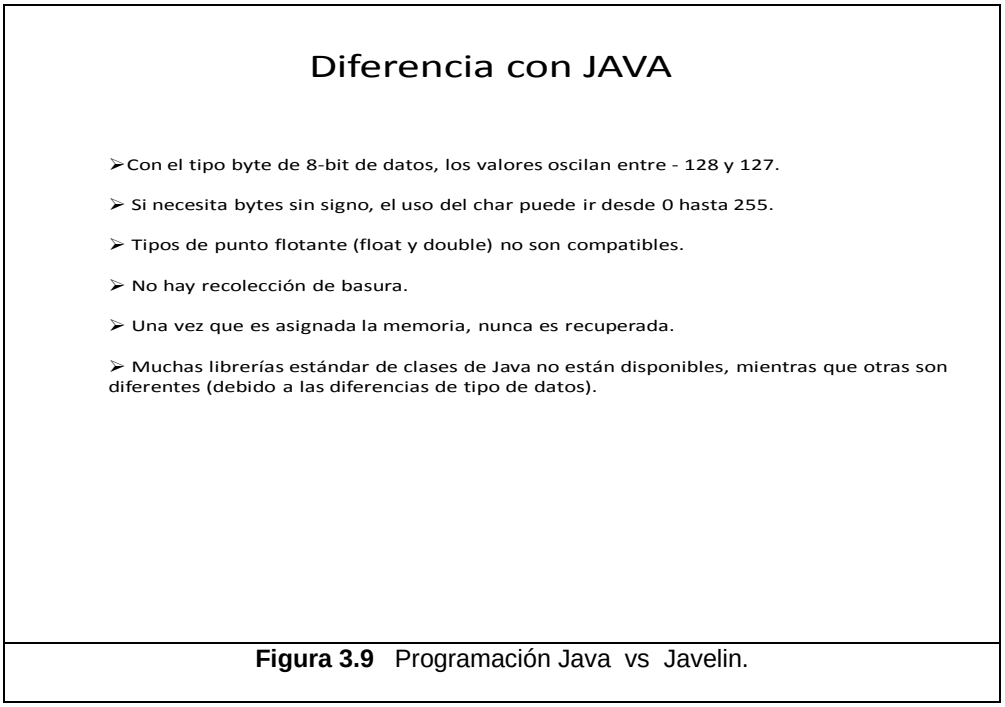


Figura 3.9 Programación Java vs Javelin.

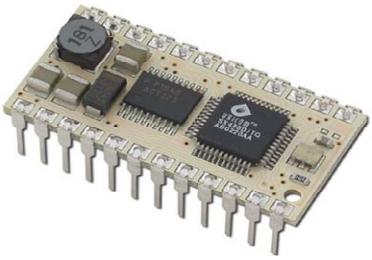
Diferencia con JAVA

- El módulo de Javelin Stamp tiene muchas bibliotecas que no figuran en el estándar de Java que permiten controlar el hardware y los dispositivos periféricos.
- los tipos de datos strings y char están compuestos de caracteres ASCII 1-bytes.
- El microcontrolador de Javelin Stamp solamente admite una matriz.

MENU INICIO

Figura 3.10 Diferencias en la programación Java vs Javelin

Características de Javelin



El Javelin puede ser programado y re-programado hasta un millón de veces.

Los códigos de instrucciones del Javelin se buscan y se ejecuta desde una SRAM paralelo en lugar de una EEPROM serie.

El Javelin tiene 32k de memoria RAM / memoria de programa con una arquitectura plana.

El Javelin ha construido en el Periférico Virtual (VPS) que se ocupan de la comunicación serial,

Comunicación serie se almacenan como un proceso en segundo plano.

Figura 3.11 Características del módulo Javelin Stamp.

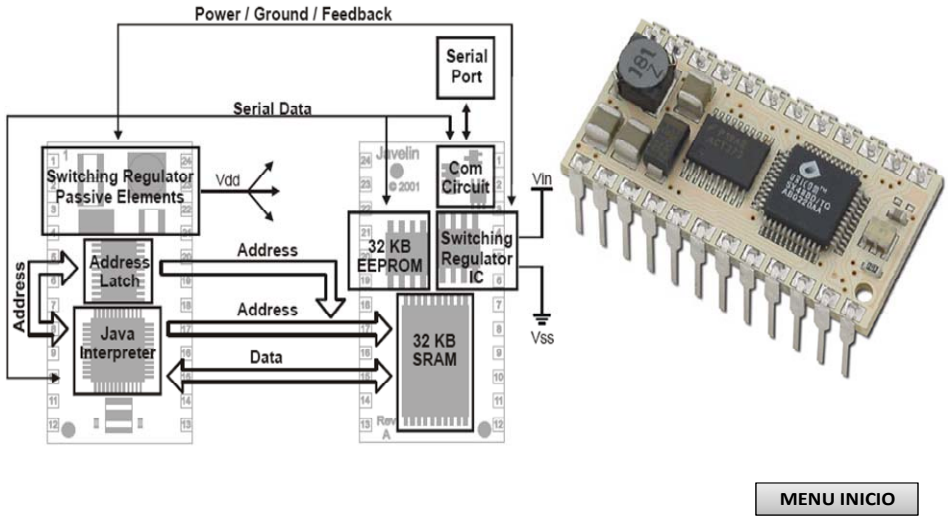


Figura 3.12 Diagrama de Bloque del módulo Javelin Stamp

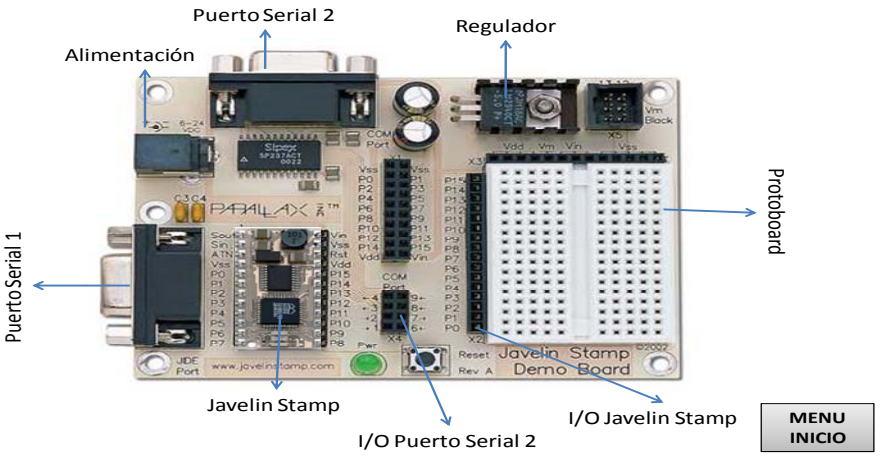
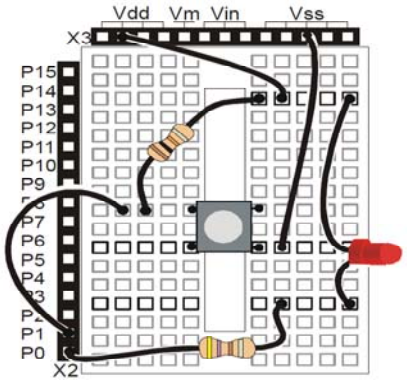


Figura 3. 13 Componentes de Starter Kit Javelin Stamp

Ejemplo

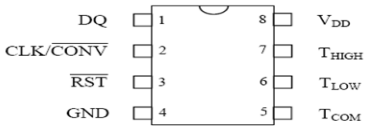
```
import stamp.core.*; // Para ser capaz de utilizar métodos de la clase de CPU
public class BotonLed // Nombre de archivo es igual que el nombre de la clase
{
 static boolean P0 = true;

 public static void main()
  { while (true)
    { if (CPU.readPin(CPU.pins[1])== false)
      { P0= !P0;
        CPU.writePin(CPU.pins[0],P0);
        CPU.delay(1000);
      }
      else
      { CPU.writePin(CPU.pins[0],true);
      }
    }
  }
}
```



MENU INICIO

Figura 3. 14 Programación en Javelin Stamp

SENSOR DE TEMPERATURA



DS1620 8-Pin DIP (300-mil)

PIN DESCRIPTION

DQ	- 3-Wire Input/Output
CLK/$\overline{\text{CONV}}$	- 3-Wire Clock Input and Stand-alone Convert Input
$\overline{\text{RST}}$	- 3-Wire Reset Input
GND	- Ground
T_{HIGH}	- High Temperature Trigger
T_{LOW}	- Low Temperature Trigger
T_{COM}	- High/Low Combination Trigger
V_{DD}	- Power Supply Voltage (3V - 5V)

Figura 3.15 Descripción de pines de conexión del DS1620

DS1620 COMMAND SET Table 4

INSTRUCTION	DESCRIPTION	PROTOCOL	3-WIRE BUS DATA AFTER ISSUING PROTOCOL	NOTES
TEMPERATURE CONVERSION COMMANDS				
Read Temperature	Reads last converted temperature value from temperature register.	AAh	<read data>	
Read Counter	Reads value of count remaining from counter.	A0h	<read data>	
Read Slope	Reads value of the slope accumulator.	A9h	<read data>	
Start Convert T	Initiates temperature conversion.	EEh	Idle	1
Stop Convert T	Halts temperature conversion.	22h	Idle	1
THERMOSTAT COMMANDS				
Write TH	Writes high temperature limit value into TH register.	01h	<write data>	2
Write TL	Writes low temperature limit value into TL register.	02h	<write data>	2
Read TH	Reads stored value of high temperature limit from TH register.	A1h	<read data>	2
Read TL	Reads stored value of low temperature limit from TL register.	A2h	<read data>	2
Write Config	Writes configuration data to configuration register.	0Ch	<write data>	2
Read Config	Reads configuration data from configuration register.	ACh	<read data>	2

MENU INICIO

Figura 3. 16 Comandos para el Integrado DS1620

CAPITULO 4

4. PRUEBAS Y SIMULACIÓN

4.1 Simulación del Javelin Stamp con DS1620.

La medición de temperatura se la realiza con el integrado DS1620 como parte sensora y el hardware del Javelin Stamp como base de control y comunicación con la PC.

El módulo Javelin Stamp permite comunicarse con el integrado DS1620 para así realizar la interacción y control de la temperatura.

El Javelin Stamp a través de la programación en Java y con el IDE que es propio del Javelin muestra los valores de temperatura en la PC luego de ser obtenidos y convertido a valores en centígrados o Farenheit.

4.1.1 Pruebas con el Integrado DS1620

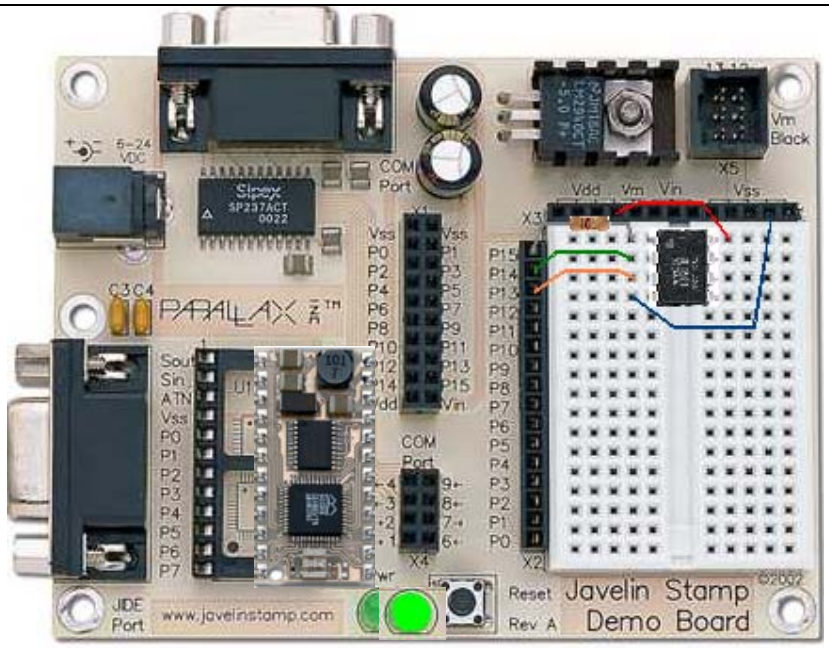


Figura 4.1 Conexión del DS1620 en tarjeta Javelin Stamp.

La conexión del Integrado DS1620 se realizó en el protoboard del Javelin Stamp.

Todas las conexiones de alimentación se realizaron a través de pequeños cables desde la fuente interna obtenida por el Starter Kit Javelin Stamp.

4.2 Pruebas y simulación con el módulo Javelin Stamp.

Luego de programar el microcontrolador Javelin Stamp se compila el programa e inmediatamente se abre una pantalla de tipo windows donde se muestran los valores de temperatura ambiente que se está sensando, con el formato que se colocó en el programa fuente.

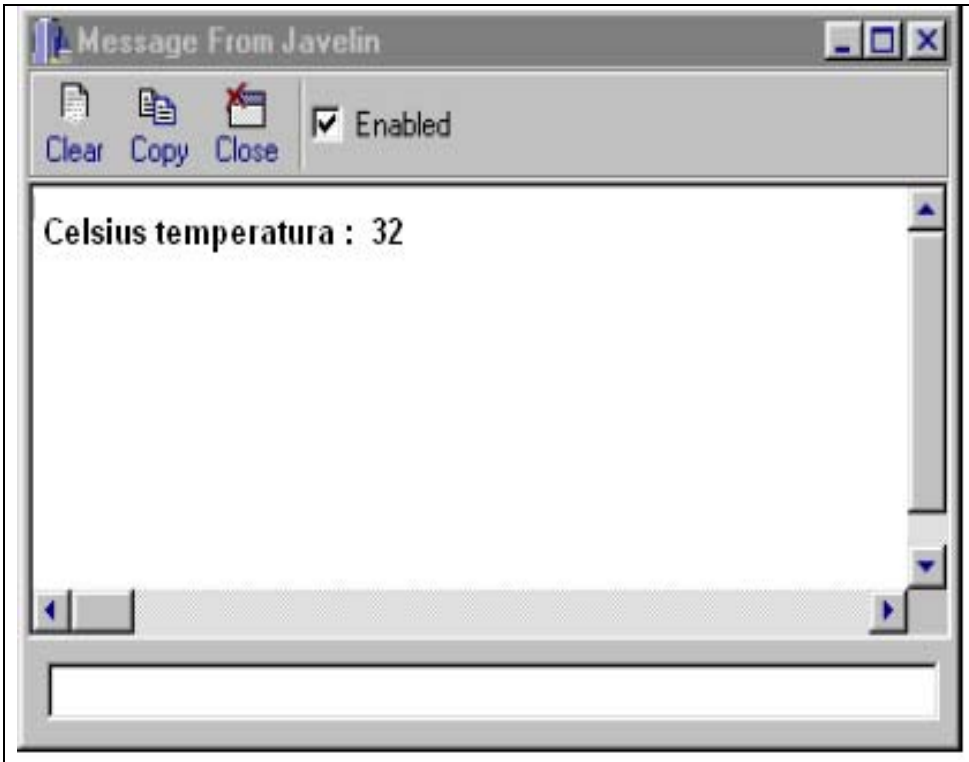


Figura 4.2 Pantalla de respuesta de simulación con Javelin 32°

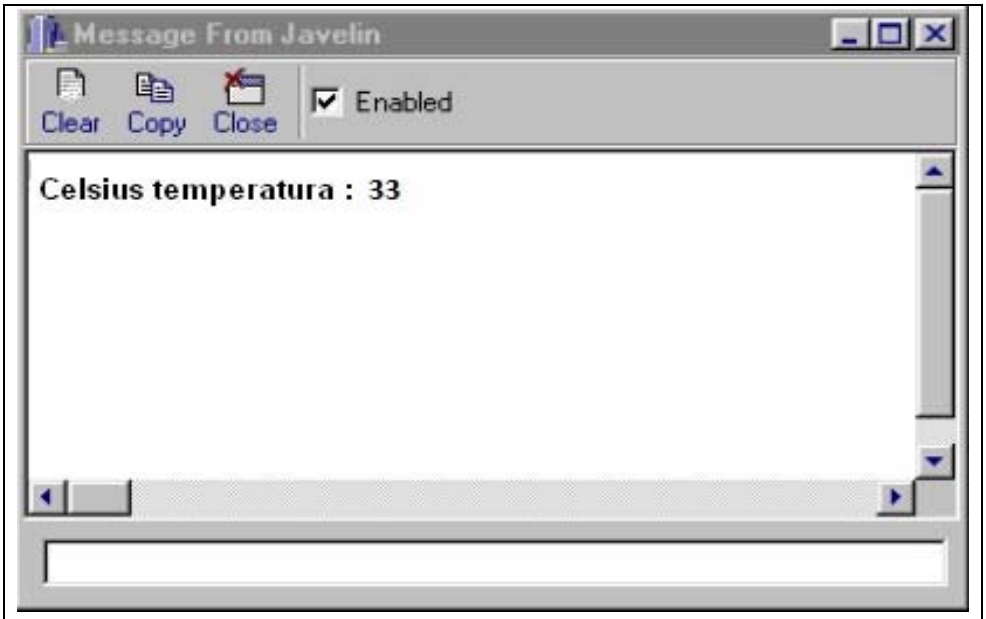


Figura 4.3 Simulación con Javelin Stamp 33˚

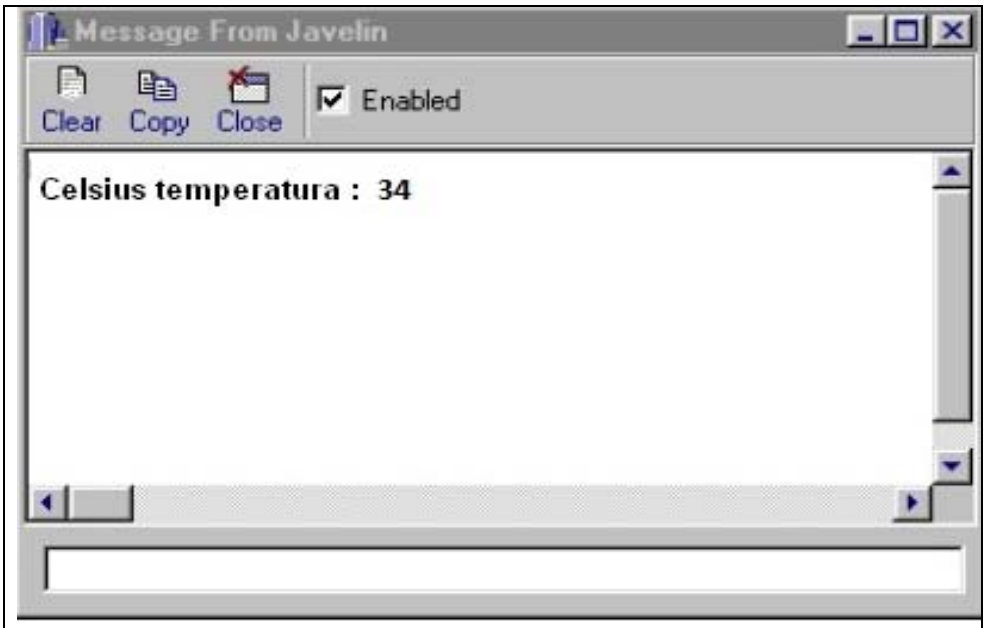


Figura 4.4 Simulación con Javelin 34˚

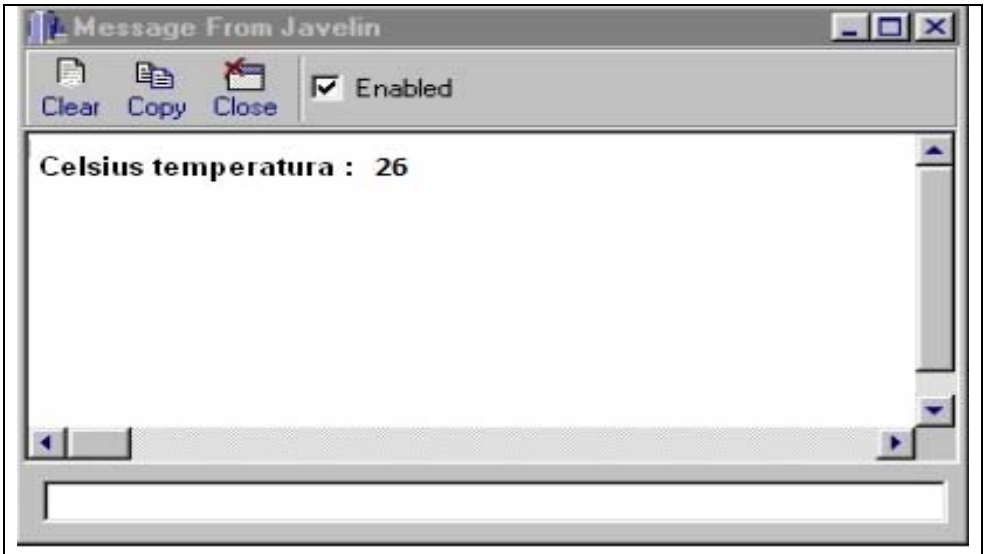


Figura 4.5 Pantalla de respuesta de simulación con Javelin 26°

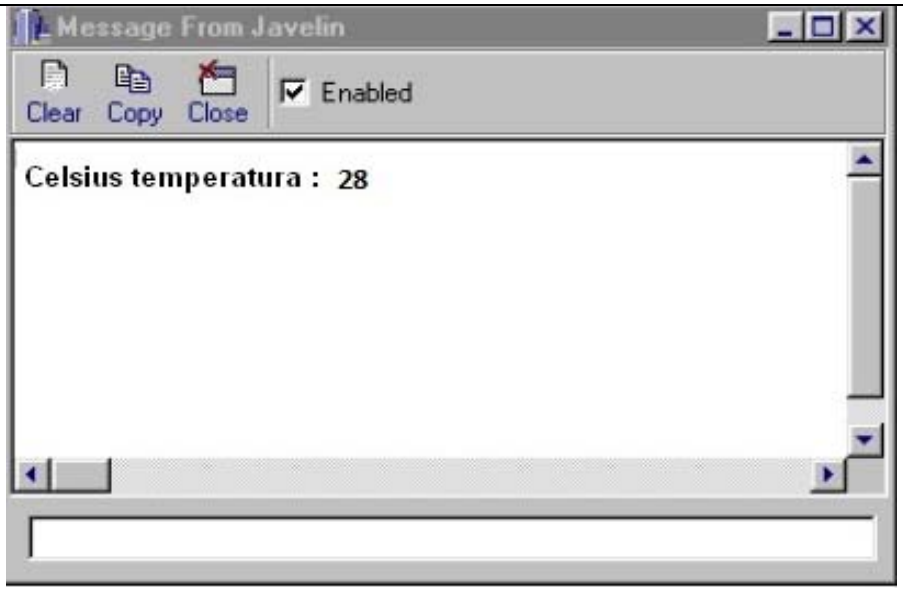


Figura 4.6 Pantalla de respuesta de simulación con Javelin 28°

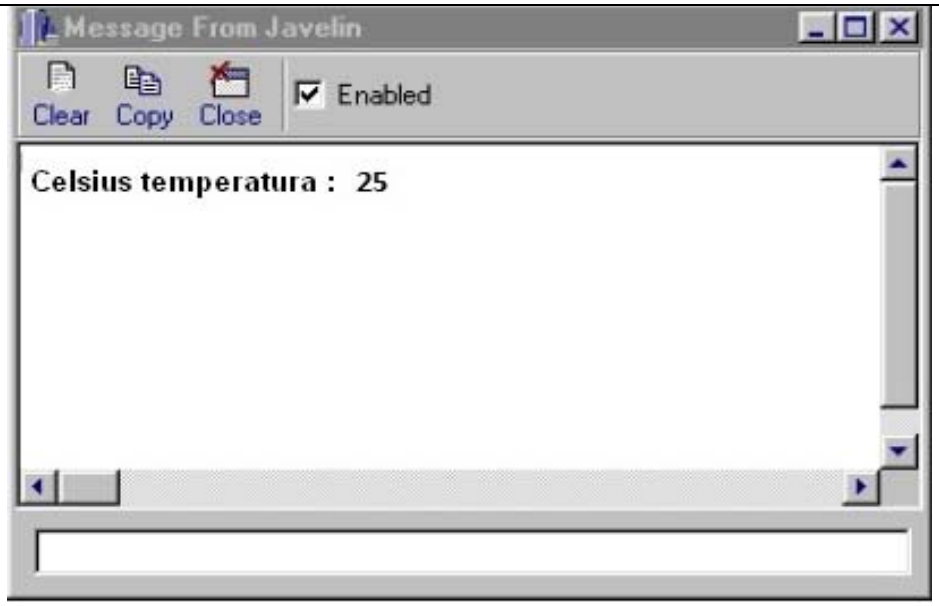


Figura 4.7 Pantalla de respuesta de simulación con Javelin 25˚

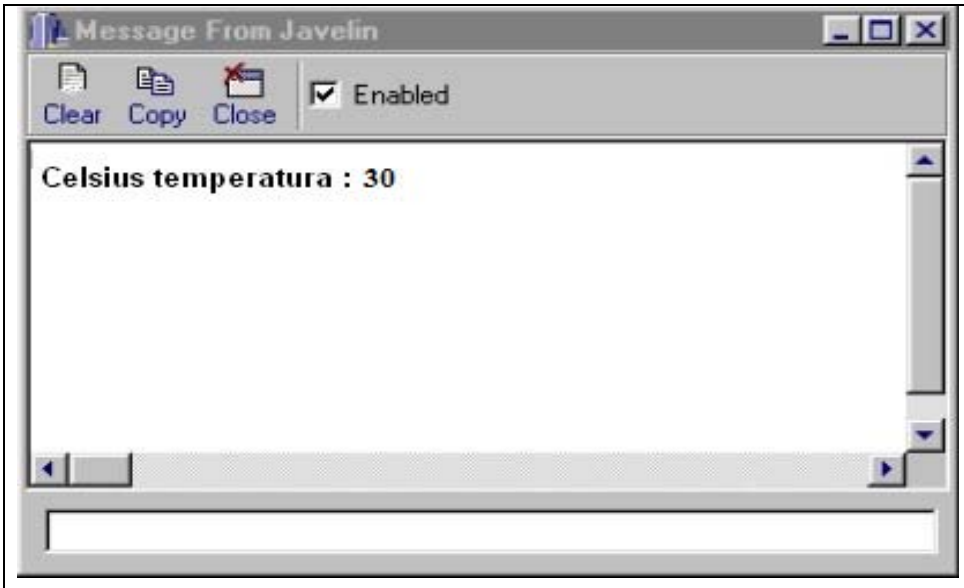


Figura 4.8 Pantalla de respuesta de simulación con Javelin 30˚

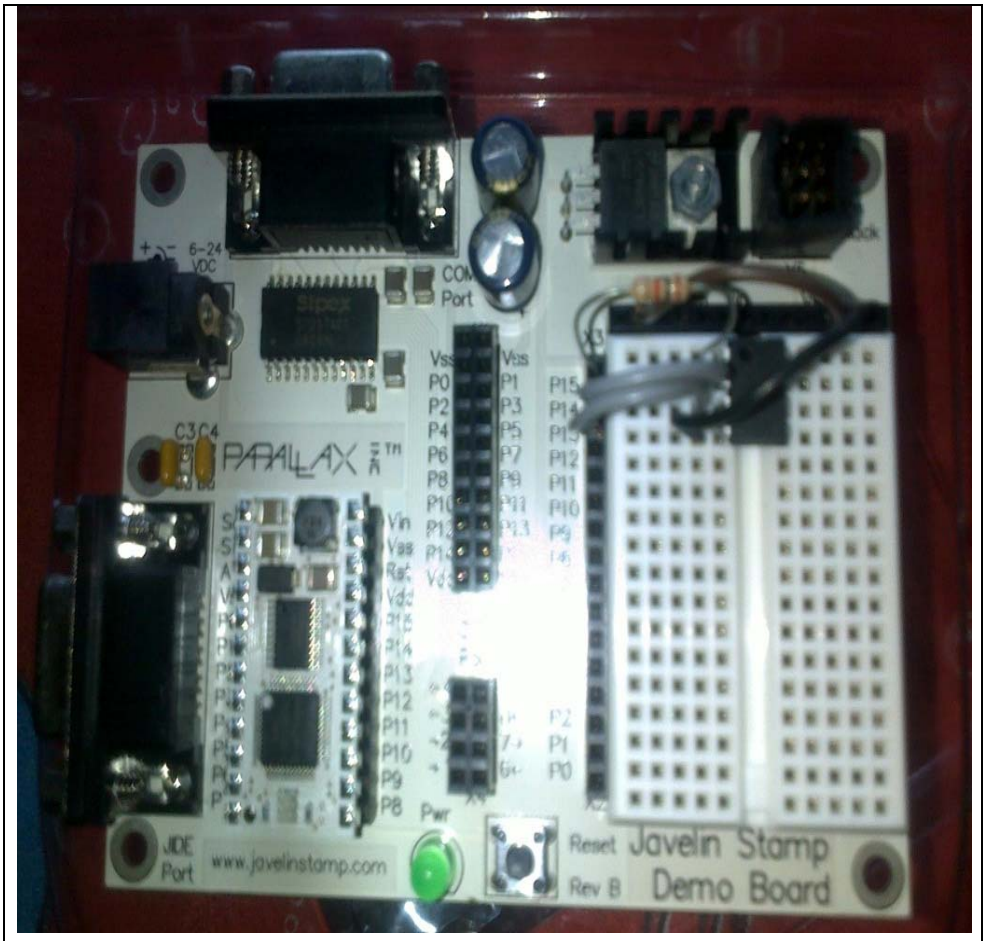


Figura 4.9 Ilustración de maqueta con módulo Javelin Stamp 1

Para las mediciones de temperatura se utilizó el Starter Kit del Javelin Stamp junto con el dispositivo DS1620, y el módulo Javelin Stamp que es programado con el lenguaje orientado a objetos Java.

Figura 4.10 Ilustración de maqueta con módulo Javelin Stamp 3

Se tomó en cuenta que el cableado del componente DS1620 al igual que todos los dispositivos, se encuentren en un área fija por lo que es muy sensible al movimiento.

Software utilizado (Java).

Las características de orientación a objetos fueron agregadas a muchos lenguajes existentes, la adición de estas características a los lenguajes que no fueron diseñados inicialmente para ellas condujo a menudo a problemas de compatibilidad y en la capacidad de mantenimiento del código.

La programación orientada a objetos es una forma de programar que trata de encontrar una solución a estos problemas. Introduce nuevos conceptos, que superan y amplían conceptos antiguos ya conocidos.

Java nació con el deseo por parte de Sun MicroSystems de buscar un lenguaje de programación enfocado a electrodomésticos pero este no tuvo la suficiente acogida por lo cual fracasó.

Java en 1995 dió un salto al mundo de Internet con lo que representó un paso muy importante dentro de este lenguaje de la programación exclusivamente enfocado a trabajar con objetos muy similar a c++,

pero tomando los beneficios de este y quitando las partes que lo hacen complejo.

Java es un lenguaje de programación que tiene múltiples características para describir sus ventajas en comparación a otros programas que se han creado con anterioridad, principalmente con los programas que son de tipo estructurado como por ejemplo c, c++, etc. Estos lenguajes fueron creados con un propósito predefinido al igual que JAVA, pero que por sus principales características lo convierten en un programa muy difícil de superar.

4.2.1 Características del Lenguaje de Programación

Existe un acuerdo acerca de qué características contempla la "orientación a objetos", las características siguientes son las más importantes:

- **Simple**

Java ofrece toda la funcionalidad de un lenguaje potente, pero sin las características menos usadas y más confusas de éstos. C++ es un lenguaje que adolece de falta de seguridad, por ello

Java se diseñó para ser parecido a C++ y así facilitar un rápido y fácil aprendizaje.

Es un lenguaje sencillo de aprender. Los creadores de Java partieron de la sintáxis de C++ “simplificada” y trataron de eliminar de este lenguaje todo lo que resultase complicado o fuente de errores, tiene incluidas librerías que facilitan el desarrollo de los programas.

- **Orientado a Objetos**

Es un lenguaje que utiliza objetos como elementos fundamentales en la construcción de la solución. Un objeto es una abstracción de algún hecho o cosa del mundo real que tiene atributos que representan sus características o propiedades y; métodos que figuran su comportamiento o acciones que realizan. Todas las propiedades y métodos comunes a los objetos se encapsulan o se agrupan en clases. Una clase es una plantilla o un prototipo para crear objetos, por eso se dice que los objetos son instancias de clases.

Un objeto posee funcionalidades, características que pueden ser usadas en forma independiente, pero juntas se complementan.

En la actualidad posiblemente Java sea el lenguaje más orientado a objetos de todos los existentes; en Java todo, a excepción de los tipos fundamentales de variables (int, char, long...) es un objeto.

- **Robusto**

Java realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. Maneja la memoria para eliminar las preocupaciones por parte del programador de la liberación o corrupción de memoria.

El compilador Java detecta muchos errores que otros compiladores solo detectarían en tiempo de ejecución o incluso nunca.

- **Seguro**

  En general Java no permite realizar cualquier acción que pudiera dañar la máquina o violar la intimidad del que visita las páginas web.

  Java resulta extremadamente seguro, ya que no permite el acceso a zonas delicadas de memoria o de sistema, con lo cual evitan la interacción de ciertos virus. Es decir, la seguridad se integra en el momento de compilación, con el nivel de detalle y de privilegio que sea necesario.

  Dada, pues la concepción del lenguaje y si todos los elementos se mantienen dentro del estándar marcado por Sun Microsystems, no hay peligro.

- **Arquitectura Neutral (Multiplataforma)**

  Para establecer Java como parte integral de la red, el compilador Java compila su código a un fichero objeto de formato independiente de la arquitectura de la máquina en que

se ejecutará. Cualquier máquina que tenga el sistema de ejecución (run-time) puede ejecutar ese código objeto, sin importar en modo alguno la máquina en que ha sido generado.

El código generado por el compilador Java es independiente de la arquitectura: podría ejecutarse en un entorno Linux, Mac o Windows.

El motivo de esto es que el que realmente ejecuta el código generado por el compilador no es el procesador del ordenador directamente, sino que este se ejecuta mediante una máquina virtual (JVM).

En una representación en que tuviésemos que indicar los elementos que forman parte de la arquitectura de Java sobre una plataforma, obtendríamos una figura como la siguiente:

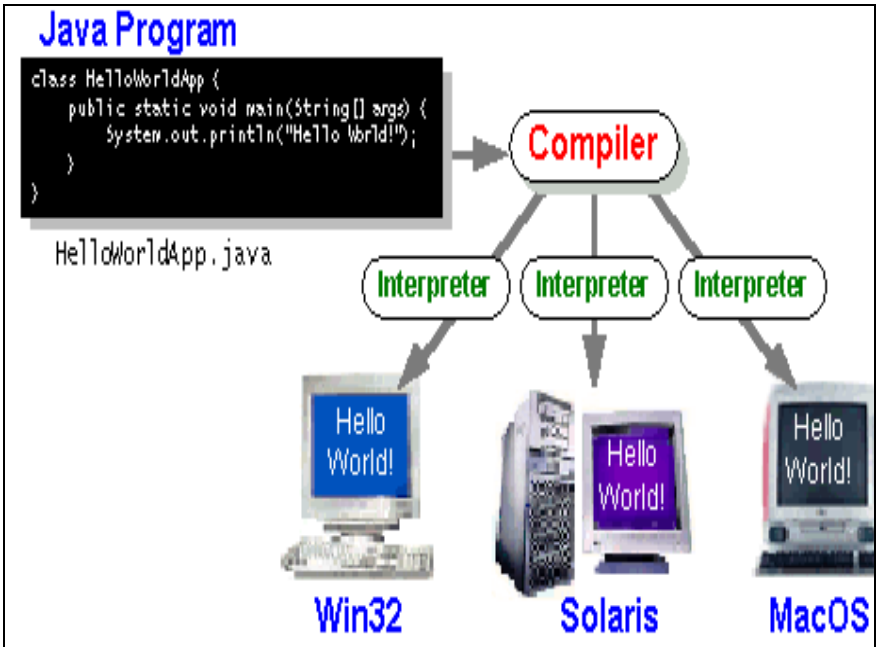


Figura 4.11 Java lenguaje de programación multiplataforma

4.3 Diferencias con Lenguaje Estructurado

Las diferencias entre los lenguajes estructurados y Java son varias pero las que hemos considerado primordiales son:

Los lenguajes de programacion estructurada están orientados a acciones, y la unidad de programación es la función.

La programacion orientada a objetos encapsula datos (atributos) y métodos (comportamiento), en objetos que están relacionados entre sí, la unidad de programación es la clase.

4.4 Principios Básicos para Programar en Java

Antes de empezar a programar en Java se tiene que familiarizar con los nombres utilizados; el lenguaje se basa en pensar que hay en el mundo real objetos y esos objetos tienen un tipo o clase. Por ello el lenguaje se basa en clases, que describen como son los objetos.

El objeto es igual que decir la instancia de la clase, las funciones son llamadas métodos, las características son los atributos y el conjunto de objetos es la clase.

Hay elementos que deben estar presentes para ejecutar un programa en Java: El programa debe estar dentro de una definición de clase

```
public class ClassName{
    }
```

La clase se debe de llamar exactamente igual que el fichero que la contiene.

La clase que se llama igual que el fichero debe de estar precedida de la palabra public.

Cuando se intenta ejecutar una clase java, la máquina virtual lo que hace es llamar a un método especial llamado main que debe estar dentro de la clase a ejecutar:

```
public static void main{
        }
```

Los comandos de Java se terminan con punto y coma; Java utiliza cinco tipos de elementos: enteros, reales, booleanos caracteres que se pueden poner en cualquier lugar del código fuente de Java, cada uno de estos literales tiene un tipo asociado con él.

En este lenguaje se pueden declarar arreglos de cualquier tipo:

```
char s[];
int iArray[];
int tabla[][]=new int[4][5]
```

4.5 Ejemplos de Programación

Ejemplo1

```
public class Ejemplo1 {
public static void main(String []args){
System.out.println(“Hola a todos”)
}
}
```

Ejemplo2

```
Public class Ejemplo2 {
Public static void main (String []args){
int a,b=0;
for(a=o;a<10;a++)
{b+=a;          // es igual b=b+a;
}
System.out.println(b)
}
}
```

4.6 Diferencia entre Java y Javelin Stamp

- La declaración del Programa principal no requiere del (String args[]) es decir : Java: public static void main (String args[])
  Javelin: public static void main ()

- El tipo int es de 16 bits de ancho, en lugar de 32-bits como es en el lenguaje de programación en Java.

- El tipo long no es compatible.

- Con el tipo de byte de 8-bit de datos, los valores oscilan entre -128 y 127.

- Si necesita tipo byte sin signo, el uso del char puede ir desde 0 hasta 255.

- Tipos de punto flotante (float y double) no son compatibles.

- No hay recolección de basura.

- Una vez que asignada memoria, nunca es recuperado.

- Muchas librerías estándar de clases de Java no están disponibles, mientras que otras son diferentes (debido a las diferencias de tipo de datos).

- El módulo de Javelin Stamp tiene muchas librerías que no figuran en el estándar de Java que permiten controlar el hardware y los dispositivos periféricos.

- Los tipos de datos string y char están compuestos de caracteres ASCII 1-byte.
- El microcontrolador de Javelin Stamp sólo admite una sola matriz de diferentes dimensiones a diferencia de Java que admite varias matrices.

4.7 Ejemplos en Javelin Stamp.

1. **Programa en Javelin**

```
public class Ejemplo {
public  static  void  main()
```

```
{
System.out.println ("Hola Mundo");
}
}
```

2. Programa en Javelin

```
import stamp.core.*;  // Para ser capaz de utilizar métodos de la clase de CPU
public class BotonLed // Nombre de archivo es igual que el nombre de la clase
{
static boolean P0 = true;
public static void main()
        { while (true)
          { if (CPU.readPin(CPU.pins[1])== false)
            { P0= !P0;
              CPU.writePin(CPU.pins[0],P0);
              CPU.delay(1000);
            }
            else
            { CPU.writePin(CPU.pins[0],true);
            }
```

```
    }
  }
}
```

INTRODUCCION

El Starter Kit Javelin Stamp consiste básicamente en un módulo que se programa con instrucciones del lenguaje Java de Sun Microsystems. El lenguaje Java es un lenguaje orientado a objetos.

El objetivo de este proyecto es el desarrollo de la programación e implementación para un sistema que permite sensar temperatura, con un paquete de Java usado en el Javelin Stamp.

El proyecto consta de dos partes, una de software que implementa el código con el lenguaje de programación Java; y otra de hardware que proporciona un control inteligente de un dispositivo de temperatura DS1620 que puede trabajar en forma autónoma con función de termostato.

El hardware de Javelin Stamp es una herramienta que permite el desarrollo del proyecto basado en el módulo Javelin Stamp y la programación en Java utilizando librerías propias.

En el capítulo III se presentan figuras que demuestran como se realiza la integración de ambas partes, es decir parte electrónica hardware con el programa software, lo necesario para iniciarse en la programación JAVA y consta con ejemplos para la identificación de la sintáxis y términos que se utilizan; que tiene como punto el poder lograr utilizar el lenguaje de programación para el proyecto.

CONCLUSIONES

1. La combinación del software que es el lenguaje de programación JAVA y el hardware, hacen que el módulo Javelin Stamp sea una poderosa herramienta dentro de la implementación de circuitos con microcontroladores, permitiendo de esta manera alcanzar uno de los objetivos de nuestro proyecto, la elaboración de un sensor de temperatura.

2. Tomando en cuenta que la idea inicial de incursionar en la elaboración y simulación de módulos a través de Java se puede considerar que los resultados que se obtuvieron en la simulación del sensor de temperatura son satisfactorios con los que se podría extender a una mayor investigación para casos particulares en otros controles.

3. Dependiendo de la programación del microcontrolador, podemos disponer de una gran cantidad de funciones y aplicaciones. En nuestro caso, la tarea principal del microcontrolador es la de regular el tráfico de los datos con el integrado DS1620; las funciones proporcionadas por el programa del microcontrolador establecen sobre el circuito los umbrales de conmutación y el almacenamiento de la temperatura máxima y mínima leídas.

4. En base a nuestra experiencia en el desarrollo de nuestro proyecto se pudo observar que se pueden obtener iguales o mejores aplicaciones gracias a las ventajas que proporciona las librerías del módulo de Javelin Stamp, tales como core diseñada para facilitar el uso al Javelin Stamp en el momento de leer sensores, controles de salidas de circuitos, comunicación con periféricos y más.

5. El DS1620 tiene un conjunto de grupos funcionales que nos permiten realizar un gran número de aplicaciones, es un elemento que puede trabajar como un termostato sin necesidad de una circuitería periférica demasiado amplia y compleja, con lo cual no necesita la conexión a elementos externos como microcontroladores para poder realizar un control de tipo ON – OFF (relés), convirtiéndose de esta manera en un pequeño hito para innovar con nuevas tecnologías de simulación y que se puedan desarrollar a gran escala.

RECOMENDACIONES

1. Tener conocimiento básico en microcontroladores y lenguaje de programación Java facilita el entendimiento y programación de las sentencias dentro del Javelin Stamp.

2. Al conectar la fuente de voltaje al hardware del Javelin Stamp hay que poner atención en la polaridad y el nivel de voltaje que esta envía para no dañar el microcontrolador.

3. Si se utiliza un cable de comunicación serial diferente al que trae el hardware, cerciorarse de que sea una conexión de punto a punto, de no ser así no se podrá comunicar la PC con el Javelin Stamp.

4. Asegurarse de tener conectado el hardware con el cable serial a la PC, para que el software del Javelin me permita trabajar con la tarjeta del Javelin Stamp.

5. El trabajo se lo realizó con un kit con cable de comunicación serial por lo cual sería recomendable obtener un adaptador o seleccionar un kit con comunicación USB para que sea más accesible la conexión del hardware a todas las máquinas.

ANEXO I

COMANDOS Y CARACTERÍSTICAS DEL DS1620

ORDERING INFORMATION

PART	PACKAGE MARKING	DESCRIPTION
DS1620	DS1620	8-Pin DIP (300 mil)
DS1620+	DS1620 (See Note)	Lead-Free 8-Pin DIP (300 mil)
DS1620S	DS1620	8-Pin SOIC (208 mil)
DS1620S+	DS1620 (See Note)	Lead-Free 8-Pin SOIC (208 mil)
DS1620S/T&R	DS1620	8-Pin SOIC (208 mil), 2000-Piece Tape-and-Reel
DS1620S+T&R	DS1620 (See Note)	Lead-Free 8-Pin SOIC (208 mil), 2000-Piece Tape-and-Reel

Note: A "+" symbol will also be marked on the package near the Pin 1 indicator

Tabla 1. DETAILED PIN DESCRIPTION

PIN	SYMBOL	DESCRIPTION
1	DQ	**Data Input/Output pin** for 3-wire communication port.
2	CLK/$\overline{\text{CONV}}$	**Clock input pin** for 3-wire communication port. When the DS1620 is used in a stand-alone application with no 3–wire port, this pin can be used as a convert pin. Temperature conversion will begin on the falling edge of $\overline{\text{CONV}}$.
3	$\overline{\text{RST}}$	**Reset input pin** for 3-wire communication port.
4	GND	**Ground pin.**
5	T_{COM}	**High/Low Combination Trigger**. Goes high when temperature exceeds TH; will reset to low when temperature falls below TL.
6	T_{LOW}	**Low Temperature Trigger.** Goes high when temperature falls below TL.
7	T_{HIGH}	**High Temperature Trigger.** Goes high when temperature exceeds TH.
8	V_{DD}	**Supply Voltage.** 2.7V – 5.5V input power pin.

Table 2. DS1620 REGISTER SUMMARY

REGISTER NAME (USER ACCESS)	SIZE	MEMORY TYPE	REGISTER CONTENTS AND POWER-UP/POR STATE
Temperature (Read Only)	9 Bits	SRAM	Measured Temperature (Two's Complement) Power-Up/POR State: -60°C (1 1000 1000)
T_H (Read/Write)	9 Bits	EEPROM	Upper Alarm Trip Point (Two's Complement) Power-Up/POR State: User-Defined. Initial State from Factory: +15°C (0 0001 1110)
T_L (Read/Write)	9 Bits	EEPROM	Lower Alarm Trip Point (Two's Complement) Power-Up/POR State: User-Defined. Initial State from Factory: +10°C (0 0001 0100)

Figura 1. DS1620 FUNCTIONAL BLOCK DIAGRAM

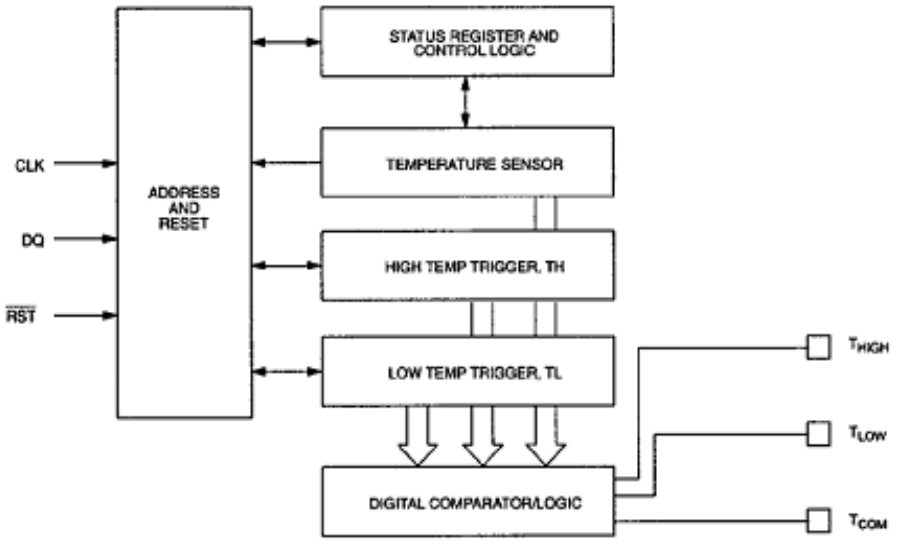


Tabla 3. TEMPERATURE / DATA RELATIONSHIPS

TEMP	DIGITAL OUTPUT (Binary)	DIGITAL OUTPUT (Hex)
+125˚C	0 11111010	00FA
+25˚C	0 00110010	0032h
+½˚C	0 00000001	0001h
+0˚C	0 00000000	0000h
-½˚C	1 11111111	01FFh
-25˚C	1 11001110	01CEh
-55˚C	1 10010010	0192h

Figura 3. THERMOSTAT OUTPUT OPERATION

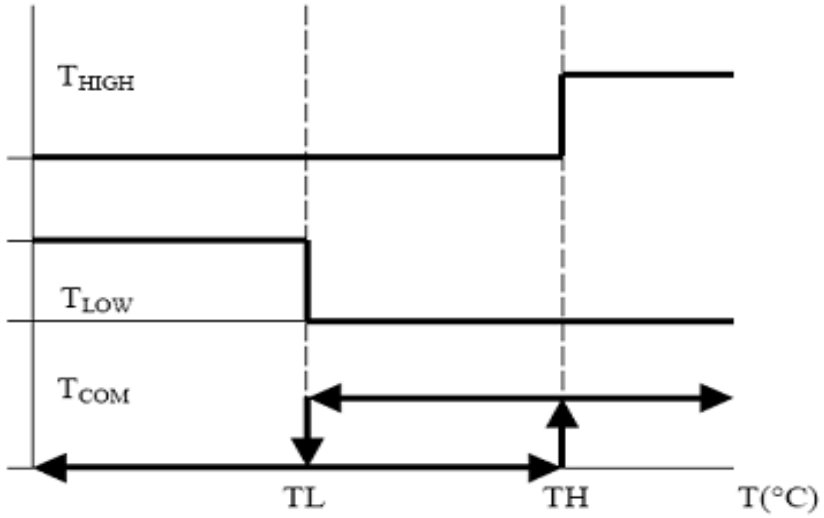


Tabla 4. DS1620 COMMAND SET

INSTRUCTION	DESCRIPTION	PROTOCOL	3-WIRE BUS DATA AFTER ISSUING PROTOCOL	NOTES
TEMPERATURE CONVERSION COMMANDS				
Read Temperature	Reads last converted temperature value from temperature register.	AAh	<read data>	
Read Counter	Reads value of count remaining from counter.	A0h	<read data>	
Read Slope	Reads value of the slope accumulator.	A9h	<read data>	
Start Convert T	Initiates temperature conversion.	EEh	Idle	1
Stop Convert T	Halts temperature conversion.	22h	Idle	1
THERMOSTAT COMMANDS				
Write TH	Writes high temperature limit value into TH register.	01h	<write data>	2
Write TL	Writes low temperature limit value into TL register.	02h	<write data>	2
Read TH	Reads stored value of high temperature limit from TH register.	A1h	<read data>	2
Read TL	Reads stored value of low temperature limit from TL register.	A2h	<read data>	2
Write Config	Writes configuration data to configuration register.	0Ch	<write data>	2
Read Config	Reads configuration data from configuration register.	ACh	<read data>	2

DC ELECTRICAL CHARACTERISTICS (-55°C to +125°C; V_{DD}=2.7V to 5.5V)

PARAMETER	SYMBOL	CONDITION	MIN	MAX	UNITS	NOTES
Thermometer Error	T_{ERR}	0°C to +70°C 3.0V ≤ V_{DD} ≤ 5.5V		±0.5	°C	2
		0°C to +70°C 2.7V ≤ V_{DD} < 3.0V		±1.25		
		-55°C to +125°C		±2.0		
Thermometer Resolution				12	Bits	
Logic 0 Output	V_{OL}			0.4	V	4
Logic 1 Output	V_{OH}		2.4		V	5
Input Resistance	R_I	$\overline{RST}$ to GND DQ, CLK to V_{DD}	1 1		MΩ MΩ	
Active Supply Current	I_{CC}	0°C to +70°C		1	mA	6
Standby Supply Current	I_{STBY}	0°C to +70°C		1.5	µA	6
Input Current on Each Pin		0.4 < $V_{I/O}$ < 0.9 x V_{DD}	-10	+10	µA	
Thermal Drift				±0.2	°C	7

AC ELECTRICAL CHARACTERISTICS (-55°C to +125°C; V_{DD}=2.7V to 5.5V)

PARAMETERS	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Temperature Conversion Time	T_{TC}			750	ms	
Data to CLK Setup	t_{DC}	35			ns	8
CLK to Data Hold	t_{CDH}	40			ns	8
CLK to Data Delay	t_{CDD}			150	ns	8, 9, 10
CLK Low Time	t_{CL}	285			ns	8
CLK High Time	t_{CH}	285			ns	8
CLK Frequency	f_{CLK}	DC		1.75	MHz	8
CLK Rise and Fall	t_R, t_F			500	ns	
$\overline{RST}$ to CLK Setup	t_{CC}	100			ns	8
CLK to $\overline{RST}$ Hold	t_{CCH}	40			ns	8
$\overline{RST}$ Inactive Time	t_{CWH}	125			ns	8, 11
CLK High to I/O High-Z	t_{CDZ}			50	ns	8
$\overline{RST}$ Low to I/O High-Z	t_{RDZ}			50	ns	8
Convert Pulse Width	t_{CNV}	250 ns		500 ms		12

AC ELECTRICAL CHARACTERISTICS (-55°C to +125°C; V_{DD}=2.7V to 5.5V)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Capacitance	C_I		5		pF	
I/O Capacitance	$C_{I/O}$		10		pF	

EEPROM AC ELECTRICAL CHARACTERISTICS

(-55°C to +125°C; V_{DD}=2.7V to 5.5V)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
EEPROM Write Cycle Time			4	10	Ms
EEPROM Writes	-55°C to +55°C	50k			Writes
EEPROM Data Retention	-55°C to +55°C	10			Years

NOTES:

1. All voltages are referenced to ground.
2. Valid for design revisions D1 and above. The supply range for Rev. C2 and below is 4.5V $\leq$ 5.5V.
3. Thermometer error reflects temperature accuracy as tested during calibration.
4. Logic 0 voltages are specified at a sink current of 4 mA
5. Logic 1 voltages are specified at a source current of 1 mA.
6. I_{STBY}, I_{CC} specified with DQ, CLK/$\overline{CONV}$ = V_{DD}, and $\overline{RST}$ = GND.
7. Drift data is based on a 1000hr stress test at +125°C with V_{DD} = 5.5V
8. Measured at V_{IH} = 0.7 x V_{DD} or V_{IL} = 0.3 x V_{DD}.
9. Measured at V_{OH} = 2.4V or V_{OL} = 0.4V.
10. Load capacitance = 50 pF.
11. t_{CWH} must be 10 ms minimum following any write command that involves the E^2 memory.
12. 250ns is the guaranteed minimum pulse width for a conversion to start; however, a smaller pulse width may start a conversion.

ANEXO II

Javelin Stamp Starter Kit

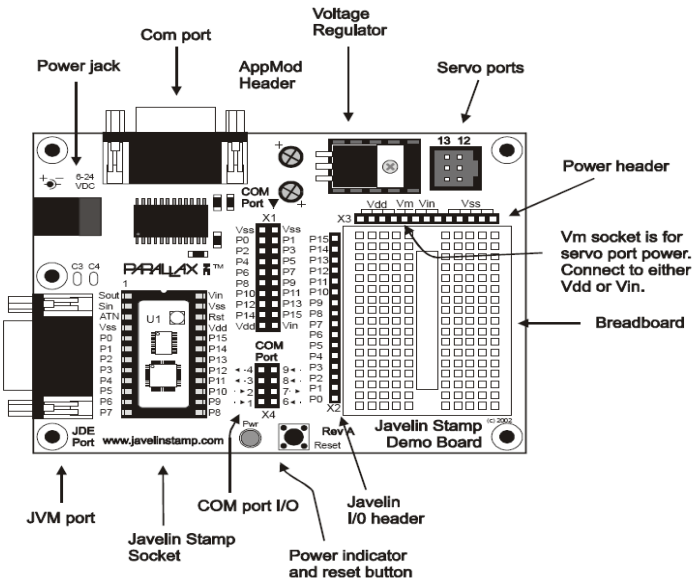


Figure 1.3
Javelin Stamp Demo Board Features

Javelin Quick Start

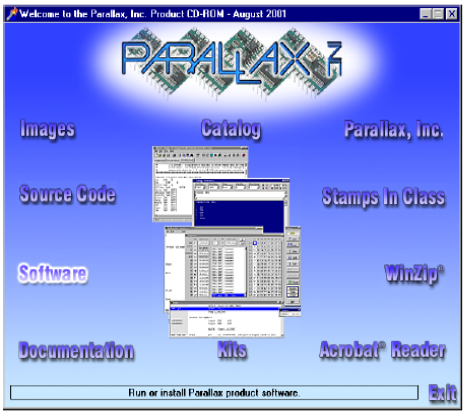


Figure 2.5
Parallax CD Browser

Quantity	Part Number	Part Description
1	550-00019	Javelin Stamp Demo Board Rev A
1	JS1-IC	Javelin Stamp Module Rev B
1	27957	Javelin Stamp Manual
1	800-00003	Serial Cable
1	800-00002	DB9 Null Modem Adapter Male to Male
1	604-00002	DS1620 Digital Thermometer
1	350-00009	Photoresistor
1	900-00001	Piezo Speaker
1	602-00009	74HC595 Output Shift Register
1	602-00010	74HC165 Input Shift Regster
3	400-00002	Tact Switch (Pushbutton)
2	350-00006	LED - Red - T1 3/4
8	350-00001	LED - Green - T 3/4
1	150-02210	RED - 220 - ¼ W - 5%
8	150-04710	RES - 470 - ¼ W - 5%
1	150-01020	RES - 1 k - ¼ W - 5%
3	150-01030	RES - 10 k - ¼ W - 5%
2	150-02230	RES - 22 k - ¼ W - 5%
2	200-01040	CAP - 0.1 µF - MonRad
2	201-01050	CAP - 1 µF - Elect.
1	201-01061	CAP - 10 µF - 16V - Elect.
1	800-00016	3" Jumper Wires (1 Bag of 10)
1	27000	Parallax CD
1	750-00009	7.5 V_{DC} DC Power Supply
1	900-00005	Parallax Standard Servo

Javelin Quick Start

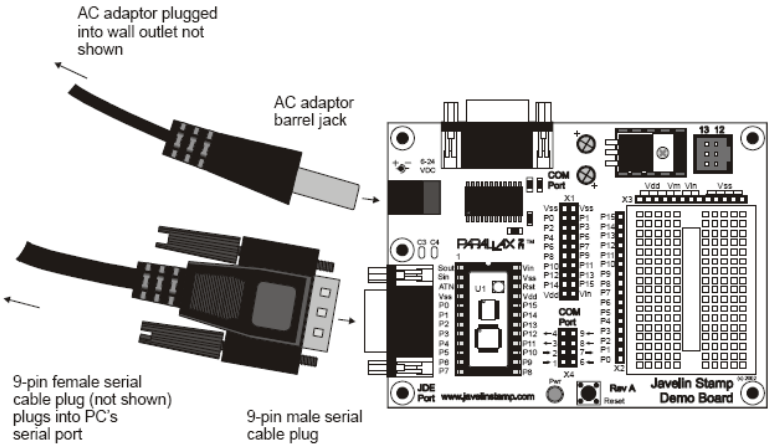
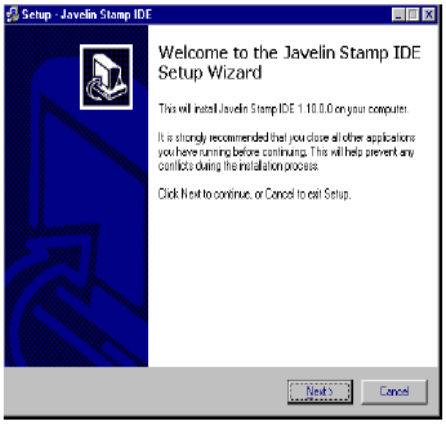
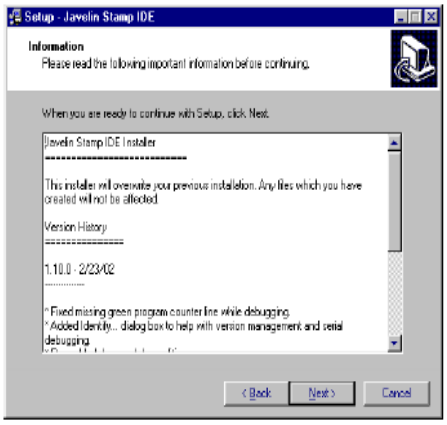


Figure 2.1 Connecting Power and Serial Cable to Javelin Stamp Demo Board

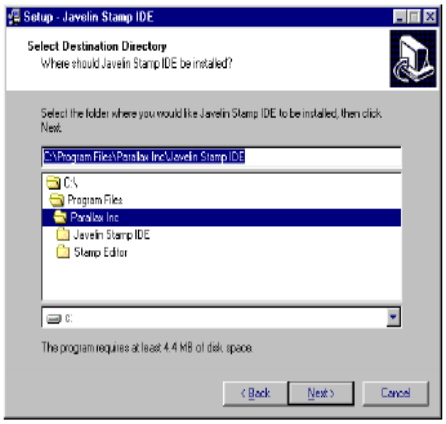
NOTE: Serial cable is a "straight-through" cable. **Do not use a null-modem cable!**



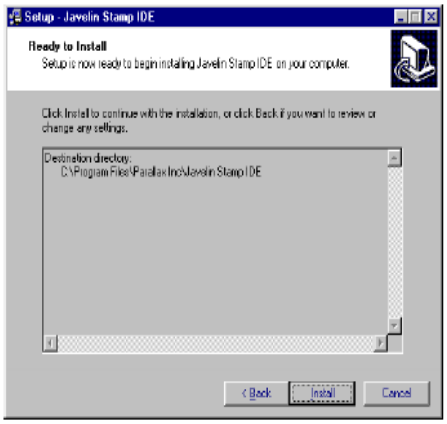
(a) Introducing the setup wizard



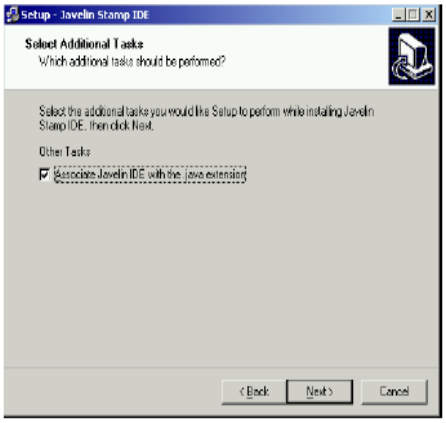
(b) IDE version information



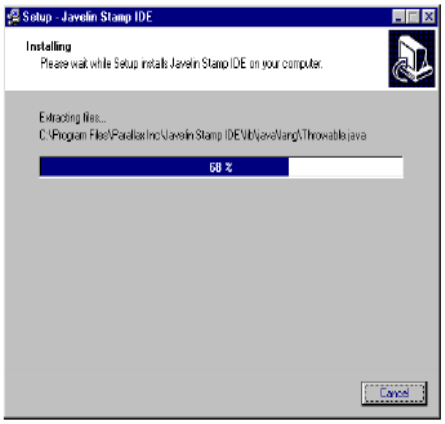
(c) Destination directory



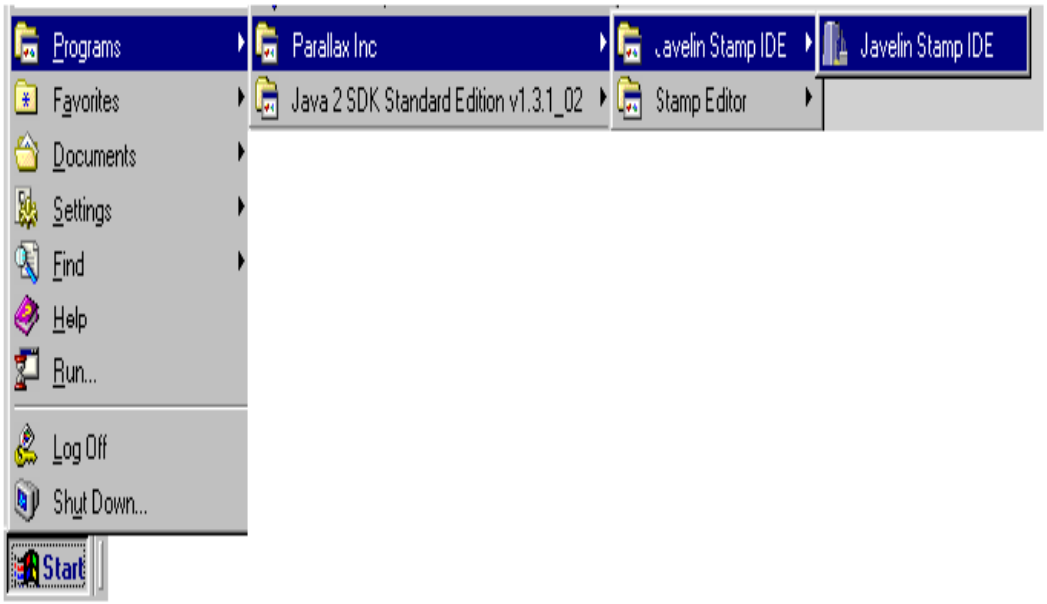
(d) Confirm destination directory

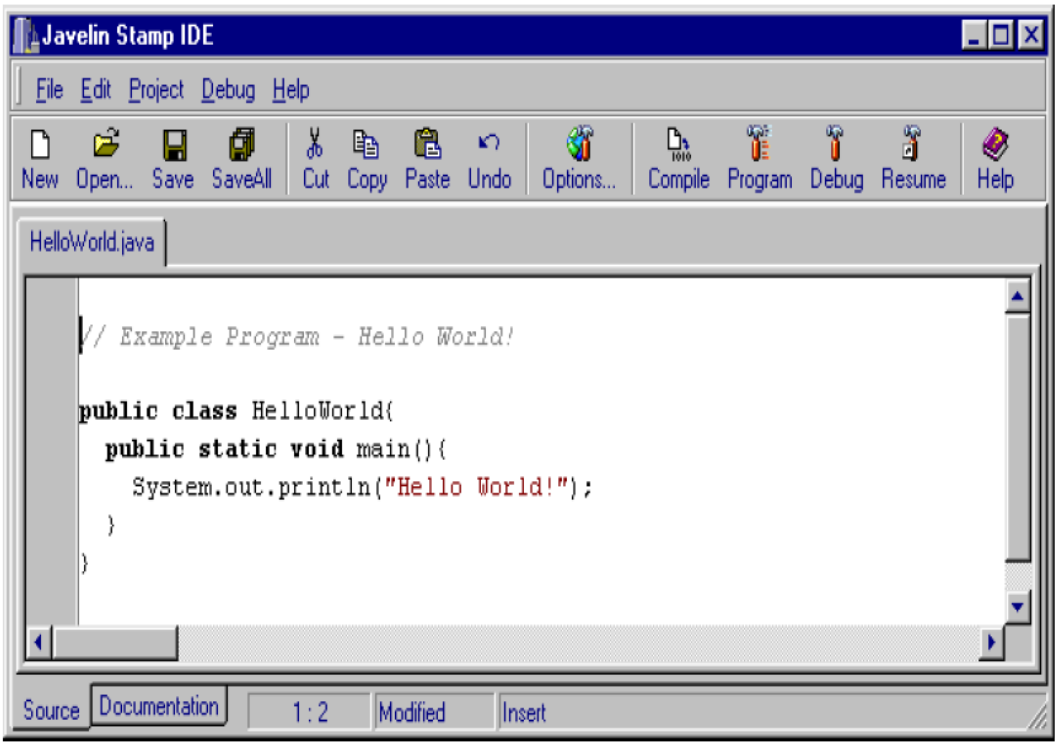


(e) Confirm file association



(f) Watch the pretty blue bar get longer


Programs
Favorites
Documents
Settings
Find
Help
Run...
Log Off
Shut Down...
Start
Parallax Inc
Java 2 SDK Standard Edition v1.3.1_02
Javelin Stamp IDE
Stamp Editor
Javelin Stamp IDE


Javelin Stamp IDE
File Edit Project Debug Help
New Open... Save SaveAll Cut Copy Paste Undo Options... Compile Program Debug Resume Help
HelloWorld.java
// Example Program - Hello World!
public class HelloWorld{
public static void main(){
System.out.println("Hello World!");
}
}
Source Documentation 1 : 2 Modified Insert

BIBLIOGRAFÍA

[1] Deitel & Deitel, Cómo Programar en Java,Primera Edición, Prentice Hall, 20/03/2010.

[2] Dennis M Ritchie & Brian W. Kernighan, El Lenguaje de Programación Java, Segunda Edición, Prentice Hall, 22/03/2010.

[3] Wikipedia, Lenguaje de Programación Java, 20 / 03 / 2010, http://es.wikipedia.org/wiki/Java_(lenguaje_de_programación).

[4] Desarrolloweb, Descripción y Características de Java, 25 / 03 / 2010, http://www.desarrolloweb.com/articulos/497.php.

[5] Profesores, Literales en Java, 25 / 03 / 2010, http://profesores.fi-b.unam.mx/carlos/java/java_basico2_4.html.

[6] error500, Recolector de basura en Java, 30 / 03 / 2010,

http://www.error500.net/garbagecollector/java/recolector_de_basura.html

[7] Parallax, Especificaciones del Javelin Stamp, 4 / 04 /2010,

http://www.parallax.com/Store/Microcontrollers/JavelinStamp/tabid/517/CategoryID/13/List/0/Level/a/ProductID/5/Default.aspx?SortField=ProductName%2cProductName

[8] Parallax, Información general sobre Javelin Stamp, 4 /04/ 2010,

http://www.parallax.com/ProductInfo/Microcontrollers/JavelinStampGeneralInformation/tabid/255/Default.aspx

[9] Parallax, JavelinDownloads, 06 / 04 / 2010,

http://www.parallax.com/ProductInfo/Microcontrollers/JavelinDownlads/tabid/443/Default.aspx

[10] Parallax, portals, 08 / 04 / 2010,

http://www.parallax.com/Portals/0/Downloads/sw/Javelin/Java.zip

[11] Datasheets, Especificaciones del DS1620 (página PDF), 10 /04/2010, http://datasheets.maxim-ic.com/en/ds/DS1620.pdf

[12] datasheetcatalog, Datasheet para el DS1620 (página PDF), 10/04/2010 http://www.datasheetcatalog.net/es/datasheets_pdf/D/S/1/6/DS1620.shtml