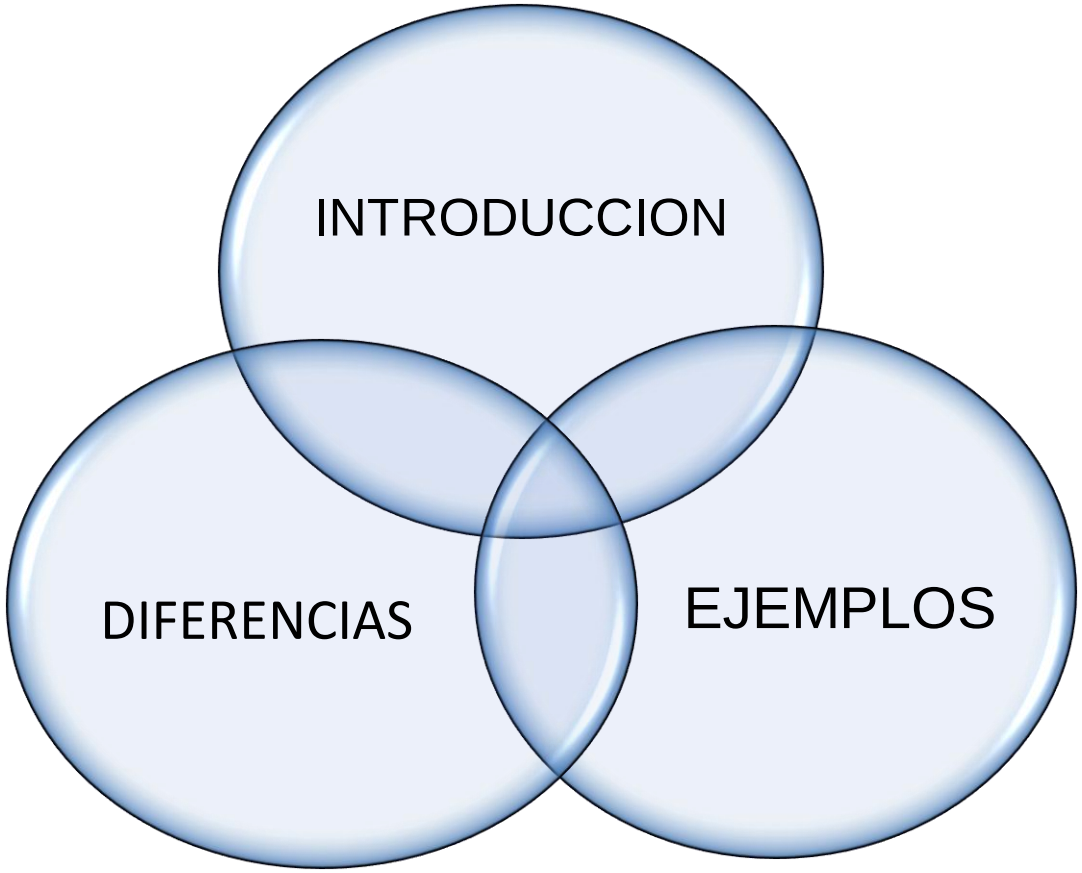
Proyecto de Materia de Graduación

Sensor de Temperatura utilizando el Starter Kit Javelin Stamp

Realizado por:
Bertha Palomeque A.
Rodrigo Barzola J.

CONOCIMIENTO BÁSICO DE JAVA



JAVA

- Orientado a Objetos
- Multiplataforma
- Programar en Java

Programar en Java

Antes de programar en Java tenemos que familiarizarnos con los nombres utilizados:

Objeto	=	Instancia
Funciones	=	Métodos
Características	=	Atributos
Clase	=	Conjunto de objetos

Elementos que deben estar presentes para ejecutar un programa Java :

- El programa debe estar dentro de una definición de clase.
  public class ClassName **{** **}**
- El programa debe contener un método main.
  public static void main**{** **}**
- Los comandos de Java se terminan con punto y coma.
- Al guardar el programa hay que considerar que el fichero tiene que tener el mismo nombre que la clase pública ClassName.

Ejemplo:

```
public class ContadorAlto {
   public static void main{
   i=i+1;
   ----------------------
  }
}
```

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=0;a<10;a++)
   {b+=a ;        // es igual b=b+a
    }
   System.out.println(b)
  }
}
```

Alcance de Objetos y Reciclado de Memoria

- Los objetos tienen un tiempo de vida y consumen recursos durante el mismo.

- // Cierra el canal cuando este objeto es reciclado

```
protected void finalize() {
close();
}
```

Literales

- Java utiliza cinco tipos de elementos:

  Enteros, reales , booleanos, caracteres y cadenas, que se pueden poner en cualquier lugar del código fuente de Java. Cada uno de estos literales tiene un tipo correspondiente asociado con él.

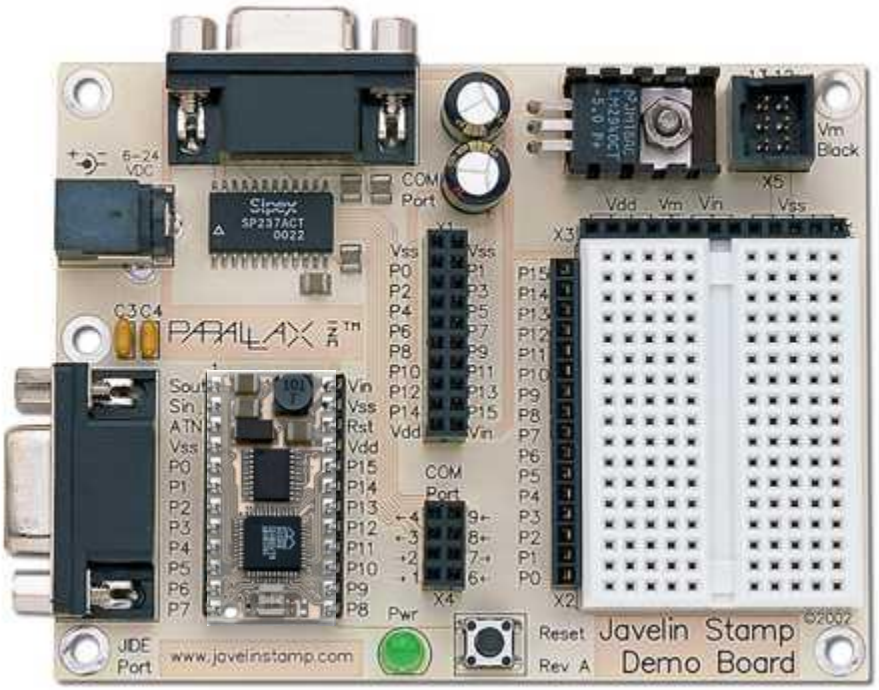
Literales

Arreglos

- Se pueden declarar en Java arreglos de cualquier tipo:
- char s[];
- int iArray[];
- Incluso se pueden construir arreglos de arreglos:

  int tabla[][] = new int[4][5];

JAVELIN



- [Diferencias con Java]
- [Características]
- [Componentes]
- [Esquema de conexión]
- [Ejemplo]
- [Proyecto]

CONCLUSIONES

- ✓ 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.

- ✓ 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.

CONCLUSIONES

- ✓ 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.

- ✓ 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.

CONCLUSIONES

- ✓ 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.

RECOMENDACIONES

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

- ✓ 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.

- ✓ 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.

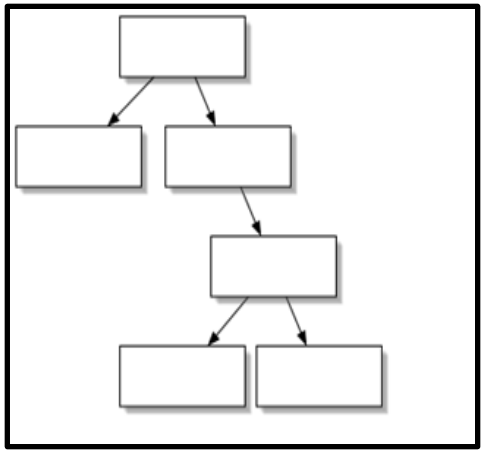
RECOMENDACIONES

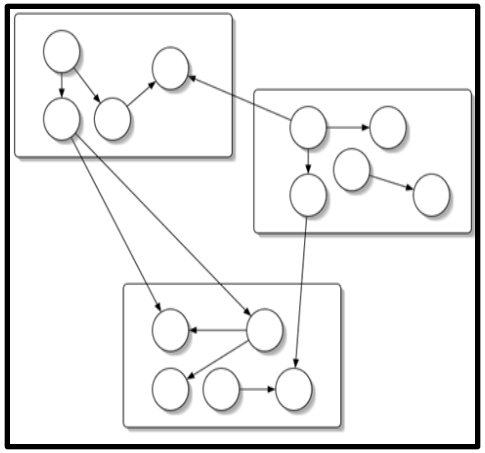
- ✓ Asegurarse de tener conectado el hardware con el cable serial a la PC, para que el software del Javelin me permita trabajar con la tarjeta del Javelin Stamp.

- ✓ 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.

GRACIAS

Programación Orientada a Objetos
vs
Programación Estructurada

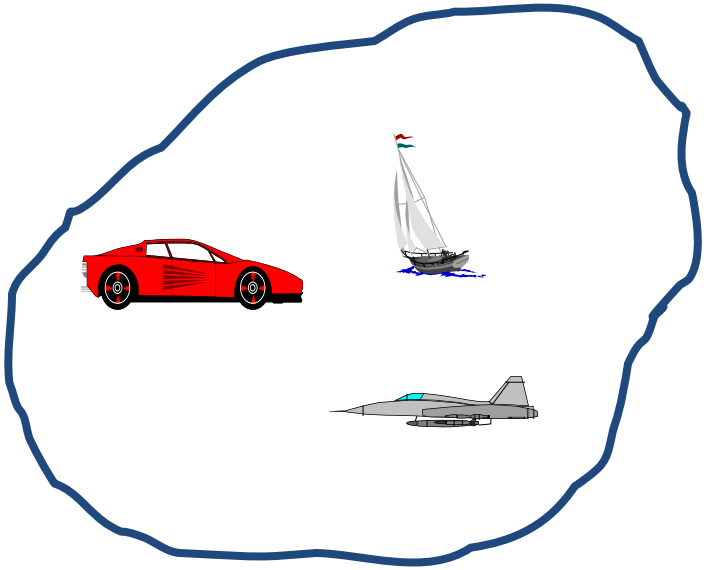




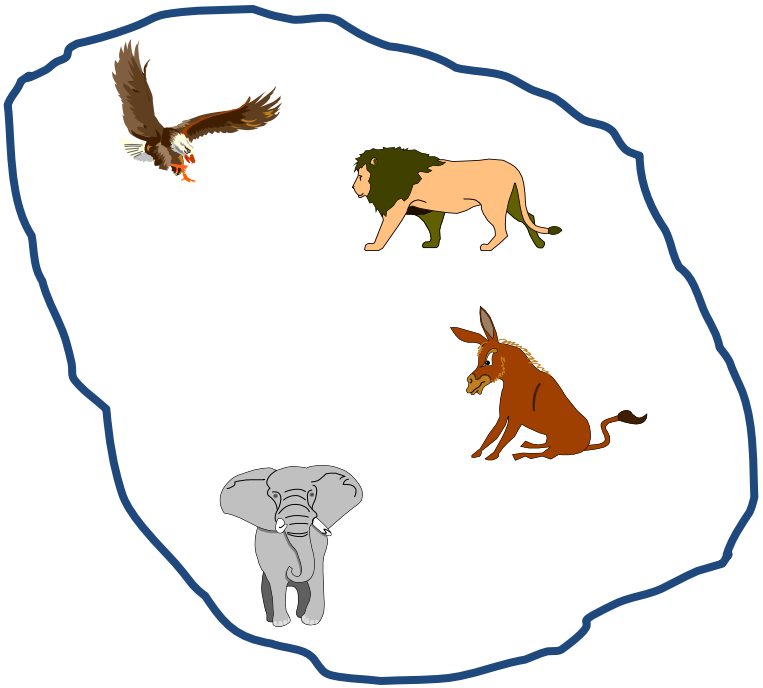
- Un **objeto**, es una abstracción de un conjunto de cosas del mundo real.
- El **objeto** posee funcionalidades.
- El **objeto** posee características que pueden ser usadas en forma independiente, pero juntas se complementan.

Mundo Real

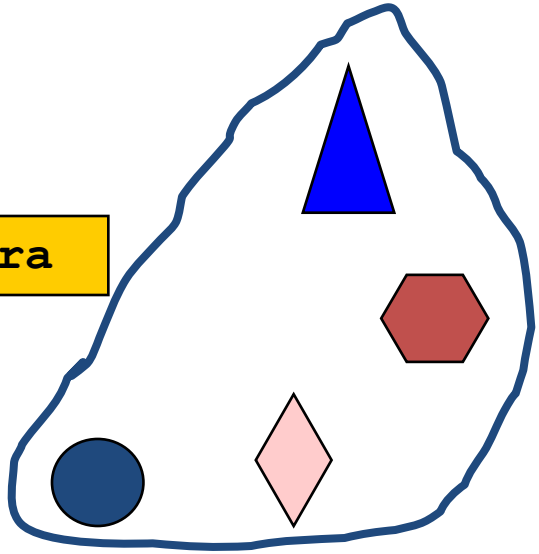
Vehículo

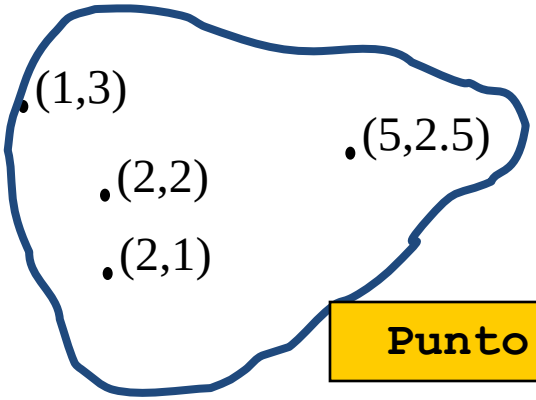


Animal



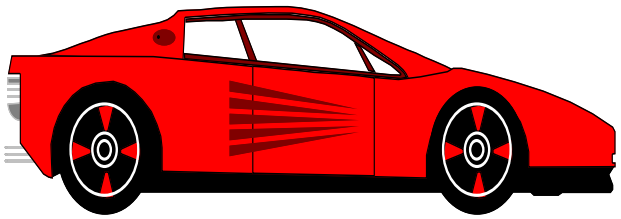
Figura





Punto

Vehículo



Abstraídos en

Características
color
peso
forma
etc.,….

Funcionalidades
encendido del motor
limpia parabrisas
frenar
etc.,……

Regresar

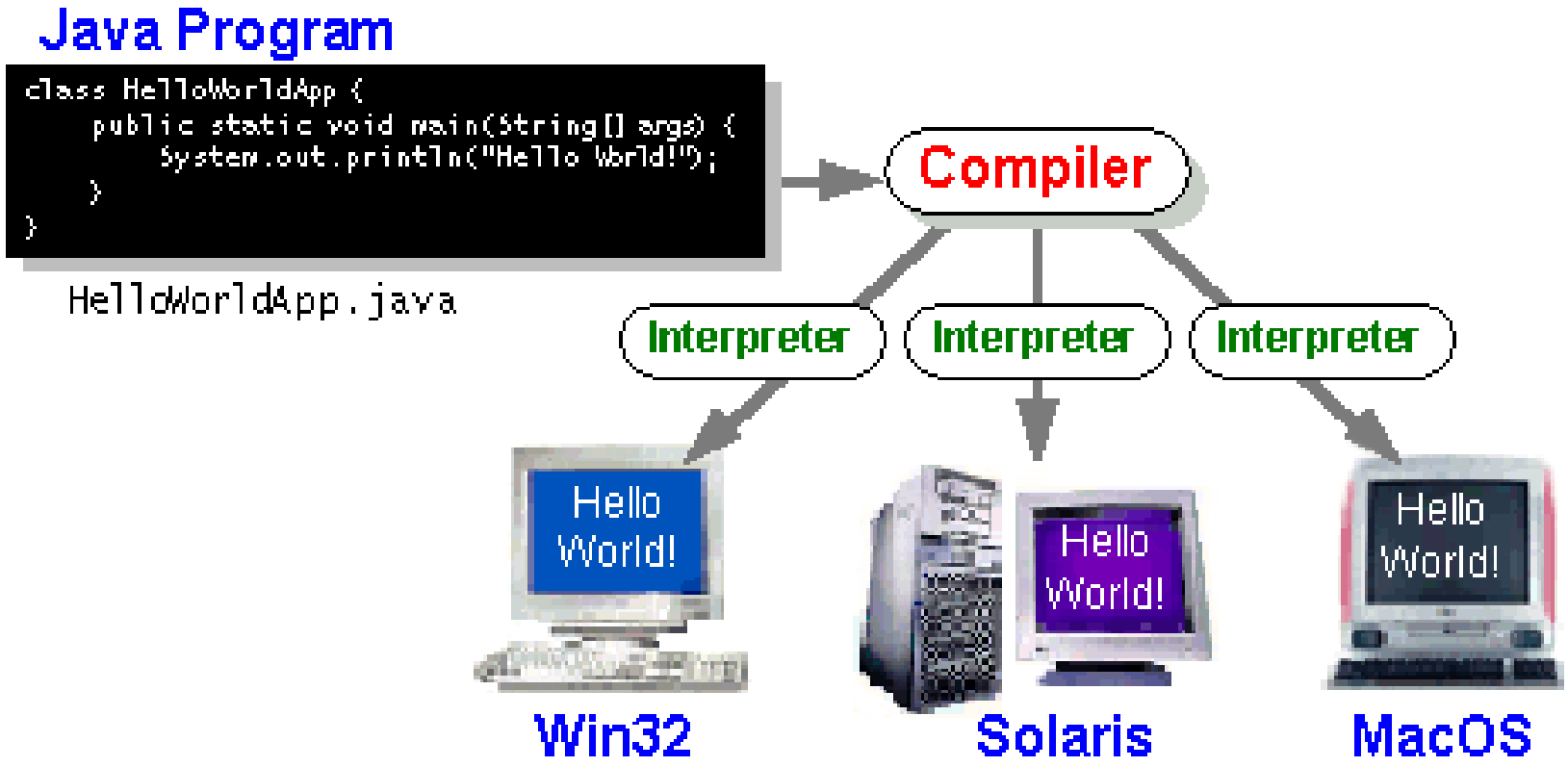
Programación Orientada a Objetos
vs
Programación Estructurado

- Los lenguajes de programación estructurada:
  - Están orientados a acciones.
  - La unidad de programación es la función.

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

MENU INICIO

Multiplataforma



- Interpretador de código (JVM)

MENU INICIO

Diferencia con JAVA

Programa en Java

```
public  class Ejemplo {
  public  static  void  main(String args[])
 {
   System.out.println ("Hola Mundo");
  }
}
```

Programa en Javelin

```
public  class Ejemplo {
  public  static  void  main()
  {
   System.out.println ("Hola Mundo");
  }
}
```

- El tipo int es de 16 bits de ancho, en lugar de 32-bits.
- El tipo long no es compatible.

Diferencia con JAVA

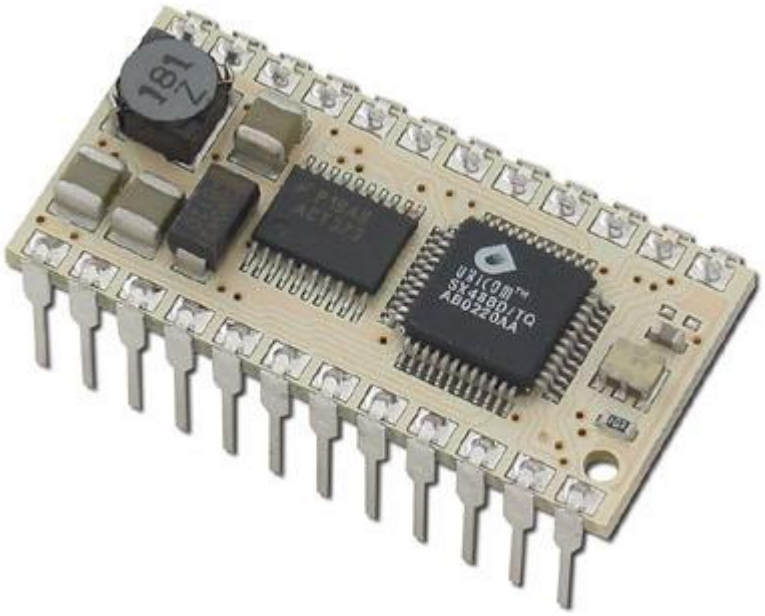
- Con el tipo byte de 8-bit de datos, los valores oscilan entre - 128 y 127.
- Si necesita tipo byte sin signo, el uso del char puede ir desde 0 hasta 255.
- Tipos de punto flotante (float y double) no son compatibles.
- No hay recolección de basura.
- Una vez que es asignada la memoria, nunca es recuperada.
- Muchas librerías estándar de clases de Java no están disponibles, mientras que otras son diferentes (debido a las diferencias de tipo de datos).

Diferencia con JAVA

- 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 Javelin Stamp admite solamente una matriz.

MENU INICIO

Características de Javelin



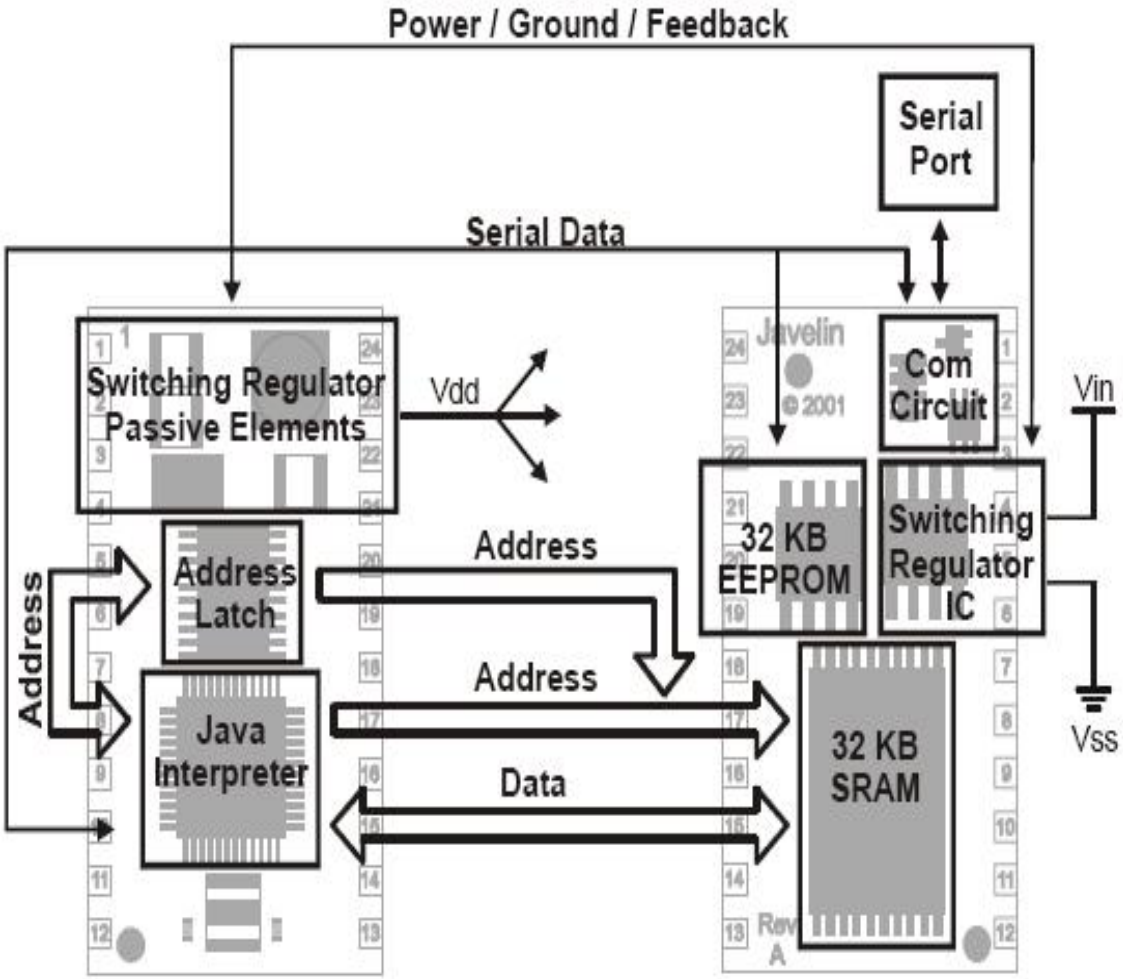
El Javelin puede ser programado y re-programado hasta un millón de veces.

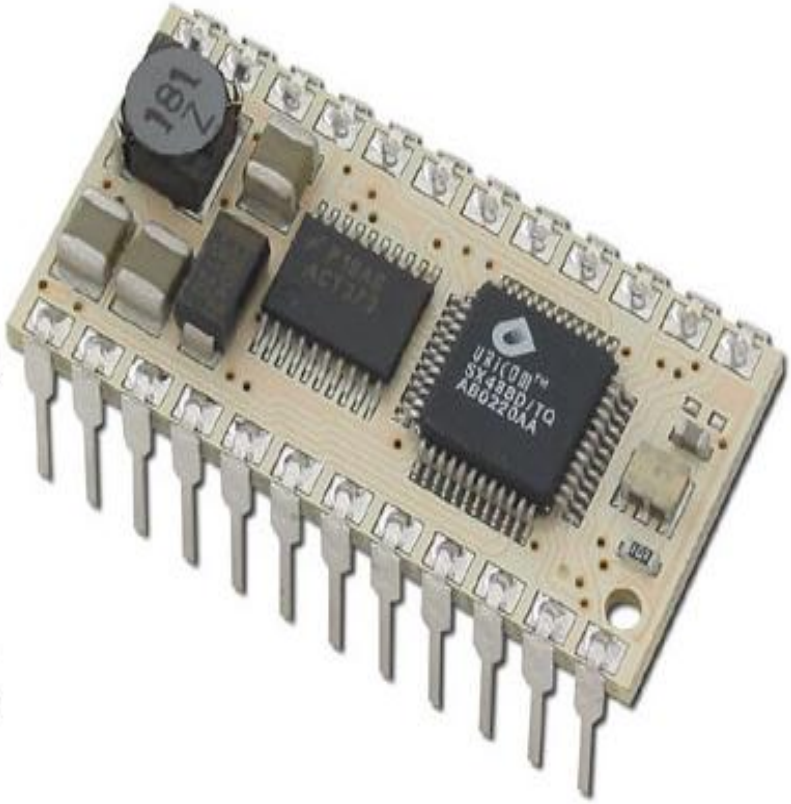
Los códigos de instrucciones del Javelin se buscan y se ejecuta desde una SRAM paralela en lugar de una EEPROM serie.

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 ocupa de la comunicación serial.

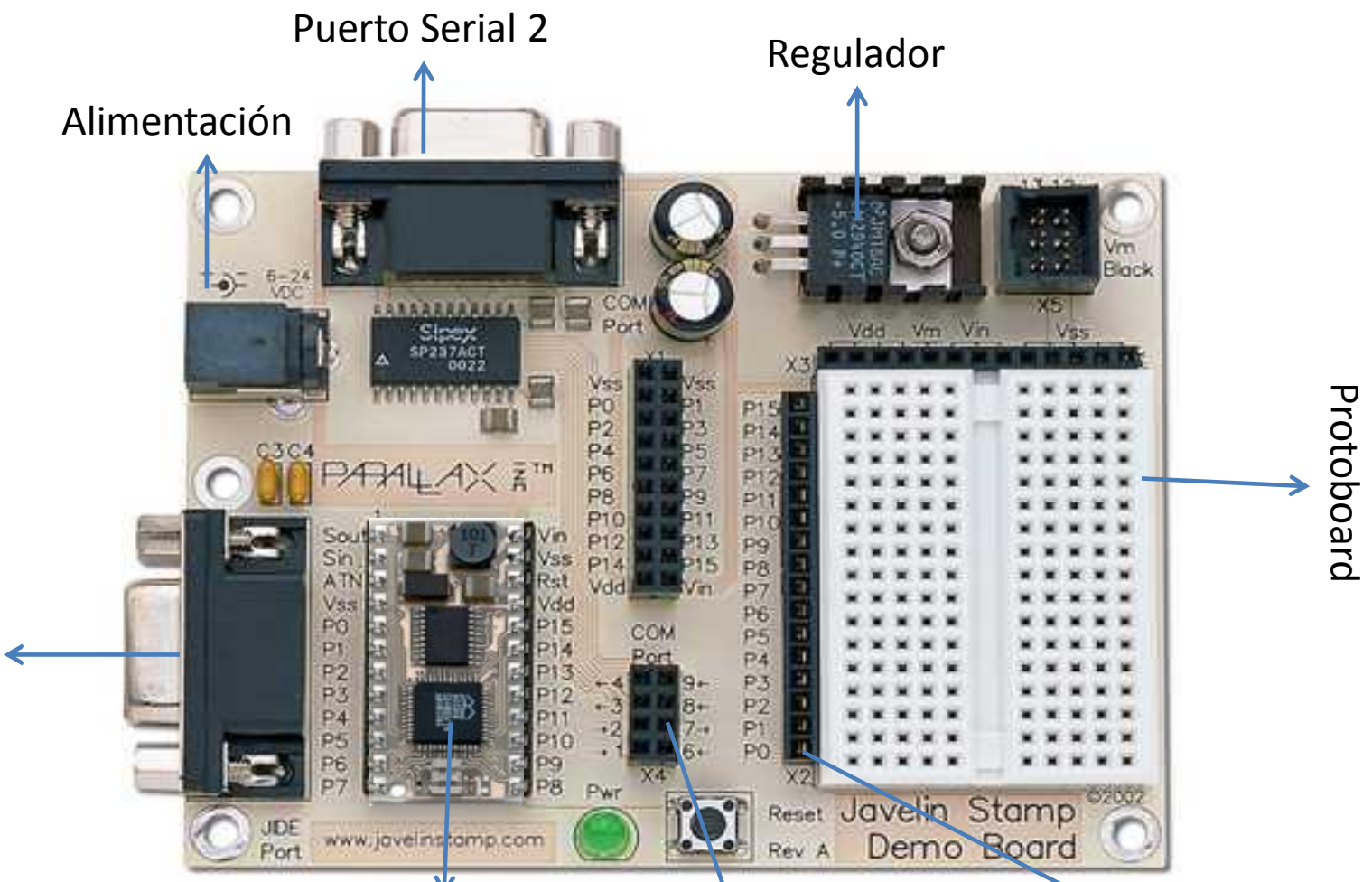
La comunicación serie se almacena como un proceso en segundo plano.


Power / Ground / Feedback
Serial Port
Serial Data
Switching Regulator Passive Elements
Vdd
Javelin
© 2001
Com Circuit
Vin
32 KB EEPROM
Switching Regulator IC
Address
Address Latch
Address
Address
Java Interpreter
32 KB SRAM
Data
Vss
Rev A



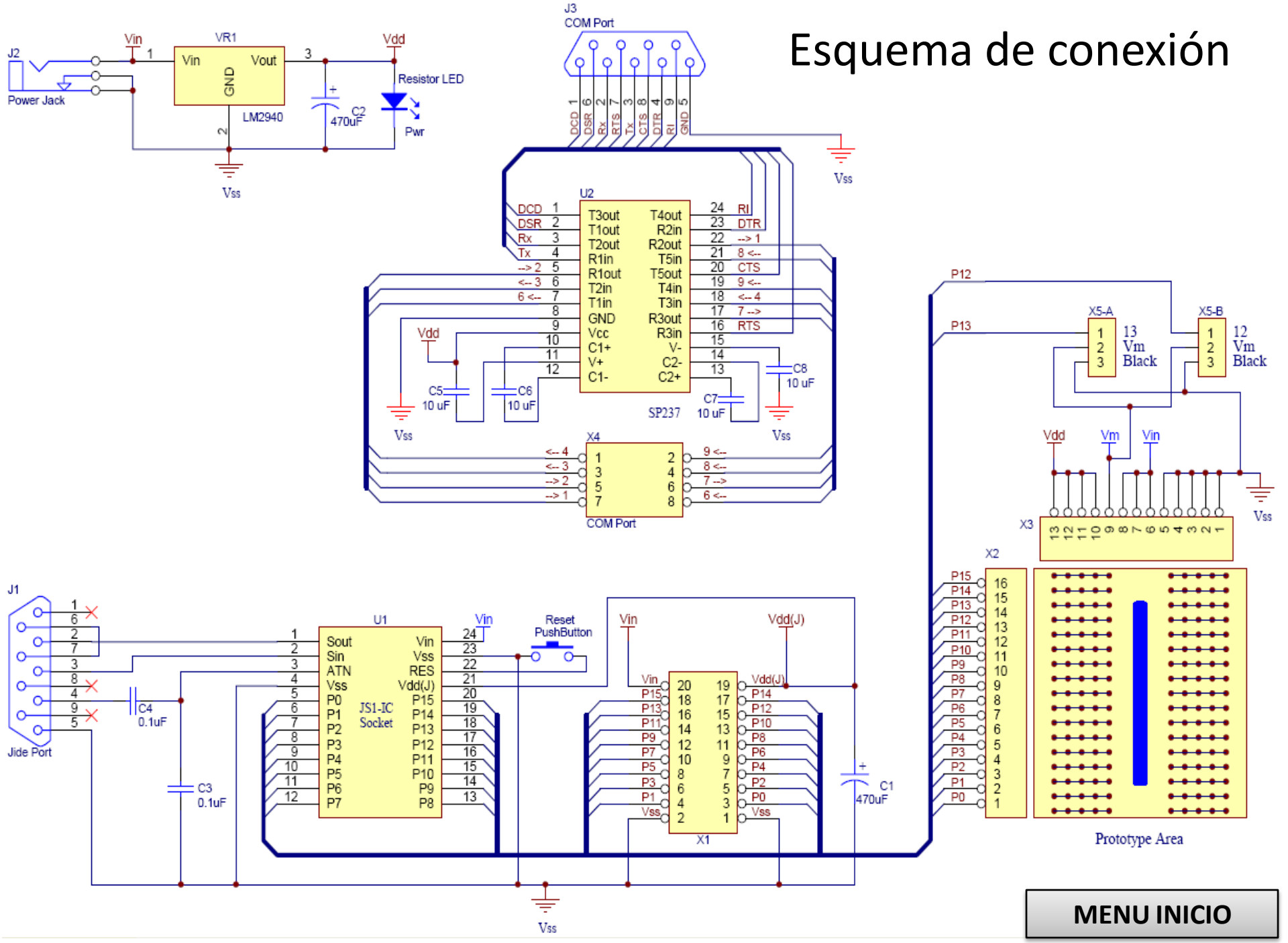
MENU INICIO

Componentes



MENU INICIO

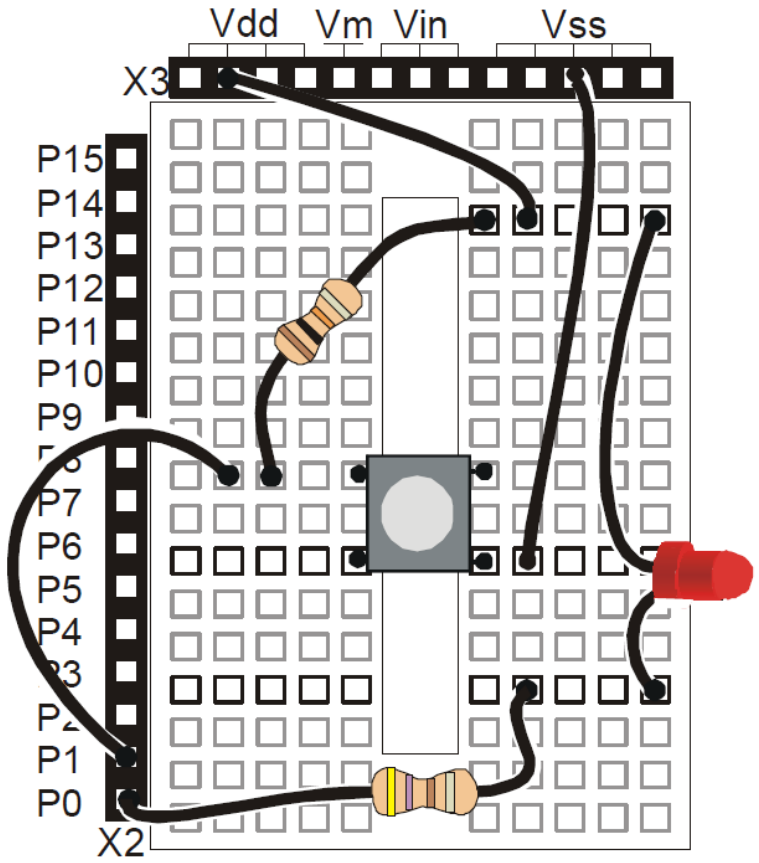
Esquema de conexión



MENU INICIO

Ejemplo

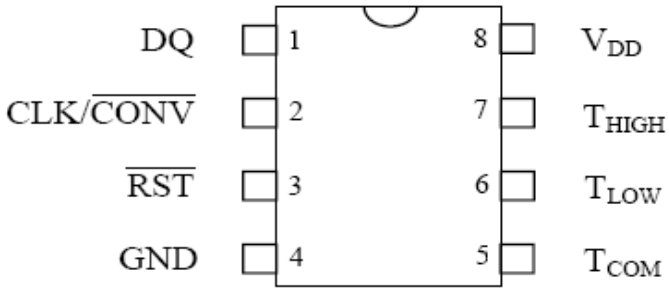
```
import stamp.core.*;  // Para ser capaz de utilizar métodos de la clase de CPU
public class BotonLed // Nombre de archivo es igual que el nombre de la clase
{
 static boolean P0 = true;

 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

Sensar Temperatura

SENSOR DE TEMPERATURA



DS1620 8-Pin DIP (300-mil)

PIN DESCRIPTION

DQ	- 3-Wire Input/Output
CLK/$\overline{CONV}$	- 3-Wire Clock Input and Stand-alone Convert Input
$\overline{RST}$	- 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)

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

```
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);
      }
    }
  }
}
```