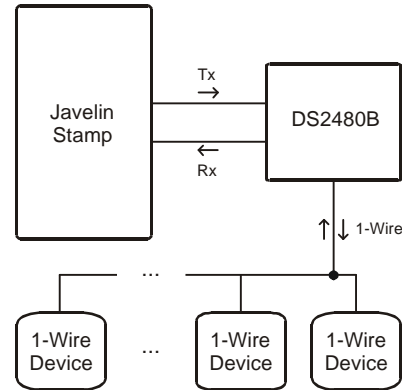


599 Menlo Drive, Suite 100
Rocklin, California 95765, USA
Office/Tech Support: (916) 624-8333
Fax: (916) 624-8003

Web Site: www.javelinstamp.com
Home Page: www.parallaxinc.com

General: info@parallaxinc.com
Sales: sales@parallaxinc.com
Technical: javelintech@parallaxinc.com



Contents

Javelin Stamp to 1-Wire Communication with the DS2480B.....	1
Downloads, Parts, and Equipment	2
1-Wire Example Circuit and How It Works.....	4
Communication Example - Reading a 1-Wire Device ROM	4
Program Listing 1.1 – Read One Device ROM.....	5
Trouble-Shooting	6
Application Example – Collecting 1-Wire Temperature Data	8
Program Listing 1.2 – Read One Device Temperature.....	9
Parasite Power Mode Wiring and Coding Tips.....	12
Published Resources – for More Information	12
Javelin Stamp Discussion Forum – Questions and Answers.....	13

Javelin Stamp to 1-Wire Communication with the DS2480B

True to the name, Maxim 1-Wire® devices require a single wire for communication. You can attach many of these devices to the same wire and then communicate with each device using the 1-Wire communication protocol. Aside from the advantage of being able to communicate with many devices over a single wire, these devices often serve unique functions that you cannot find in other products. Some examples include the useful I-button® security devices, high resolution sensors, and low power data storage devices.

You can use the Javelin Stamp to communicate with 1-Wire devices through the DS2480B Serial Line Driver. The block diagram (top-right of this page) shows how the DS2480B is placed between the Javelin Stamp and the devices on the 1-Wire bus. The DS2480B then translates the Javelin Stamp's Uart serial data into 1-Wire serial data and visa versa.

The DS2480B library class for the Javelin Stamp makes it easy to communicate with the DS2480B chip and exploit its most commonly used features. The examples in this application note will show you how to use the DS2480B class methods to exchange data with devices on a 1-wire bus. These examples will also use a single DS1822 Econo 1-Wire Digital Thermometer to demonstrate the data exchanges.

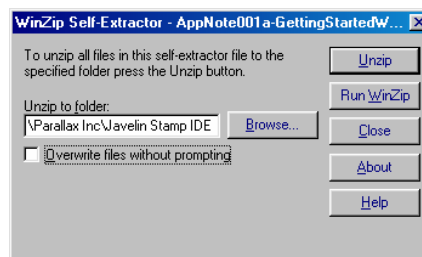
Downloads, Parts, and Equipment

This application note along with all its example program listings, library files, and javadoc html files are available for free download from www.javelinstamp.com.

- ✓ Look for the section entitled “Getting Started with 1-Wire” and download AppNote001a–GettingStartedWithOneWire.exe.

When you run AppNote001a–GettingStartedWithOneWire, it starts with a window that looks similar to Figure 1.1.

Figure 1.1
Unzipping the example
files into your Javelin
Stamp IDE.



- ✓ If you installed the Javelin Stamp IDE to the default directory of C:\Program Files\Parallax Inc\Javelin Stamp IDE, click the Unzip button.

– OR –

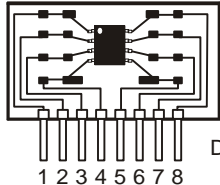



If you installed the Javelin Stamp IDE to a different directory, use the Browse button to find the directory or hand-enter it into the Unzip to folder field before clicking Unzip.

Here is a list of the files that AppNote001a–GettingStartedWithOneWire.exe installs along with their locations relative to the Javelin Stamp IDE’s folder. If you used the default install settings, these paths are inside C:\Program Files\Parallax Inc\Javelin Stamp IDE.

```
\doc\AppNote001a_Getting_Started_with_1-Wire.pdf
\doc\DS2480B.pdf
\lib\stamp\peripheral\onewire\DS2480B.java
\Projects\examples\peripheral\onewire\DS2480BReadOneDS1822ROM.java
\Projects\examples\peripheral\onewire\DS2480BReadOneDS1822Scratchpad.java
```

Table 1.1 lists the parts you will need to try the examples in this application note. The DS2480B and DS1822 are the key components. A Schottky diode is also included to maintain a small voltage difference across pins 5 and 4 specified by the DS2480B's data sheet. For those of you who are wondering if the 4.7 k Ω strong pullup resistor was omitted by accident, the answer is "no". The DS2480B has an internal pullup resistor that can be enabled and disabled by the Javelin Stamp as needed.

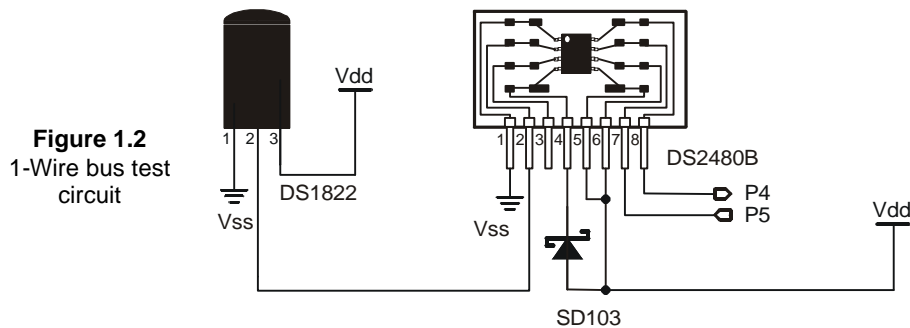
Table 1.1: Parts List

Quantity	Part Ordering Info Part Description	Schematic/Diagram Symbol/Pin Map
1	DS2480B Serial Line Driver <i>Digi-Key #:</i> DS2480B-ND	 DS2480B
1	8-pin SOIC to SIP adaptor <i>Digi-Key #:</i> 9082CA-ND	
1	DS1822 Temperature Sensor <i>Parallax Part:</i> 604-00013 - or - <i>Digi-Key #:</i> DS1822-ND	 DS1822
1	Schottky Diode <i>Digi-Key #:</i> SD103ACT-ND	 SD103  SD103

The equipment used to test this example included a Javelin Stamp, Javelin Stamp Demo Board, serial cable, 7.5 V, 1000 mA DC power supply, serial cable, and PC with the Javelin Stamp IDE v2.01. The module shown in Table 1.1 was built using the two Digi-Key (www.digikey.com) parts listed. The DS2480B IC is a surface mount part that is soldered to the 8-pin SOIC to 8-pin SIP adaptor PCB making it a breadboard-friendly part. The adaptor PCB comes with a very nice set of instructions for soldering surface mount parts. According to these instructions, the soldering job requires a 10 to 25 watt range soldering iron with a fine tip, 10 to 20 mil diameter solder, and water soluble flux.

1-Wire Example Circuit and How It Works

Figure 1.2 shows the test circuit you will use with Program Listings 1.1 and 1.2. The connections labeled P4 and P5 are used for exchanging Uart serial data between the Javelin Stamp and the DS2480B. The connection between pin 2 of the DS2480B and pin 2 of the DS1822 is where the 1-Wire communication occurs.



The DS1822 can also be wired in “parasite power mode” with pins 1 and 3 tied to Vss and power supplied by the 1-wire bus line. For details on using the DS1822 in this mode, see Parasite Power Mode Wiring and Coding Tips on page 12.

Communication Example - Reading a 1-Wire Device ROM

Program Listing 1.1 on page 5 can come in handy for reading each individual 1-Wire device’s ROM address before ganging many devices on a single bus. Some scrutiny of the DS1822’s data sheet reveals that if you only have one device on the bus, you can send the hexadecimal value 0x33 to request a 1-Wire device’s ROM address. Provided there is just one device on the bus, it will automatically reply. The steps required are:

- Send a reset pulse to the 1-wire bus.
- Send the hexadecimal message 0x33, the **READ_ROM** command.
- Read 8 bytes from the 1-wire bus.
- Reset the 1-wire bus.

Once you have a **DS2480B** object at your disposal, this task list is made easy by the **reset**, **message**, and **getData** methods. You can use the DS2480B javadoc (DS2480B.pdf in your “...Doc” folder) as a reference to find out more about the **DS2480B** object (named **oneWireBus** in Program Listing 1.1) and how to use its methods and fields.

By setting a breakpoint and then running Program Listing 1.1, you can view the ROM address in the Debugger as well as in the Messages from Javelin window. Your results should be similar to the screen captures in Figure

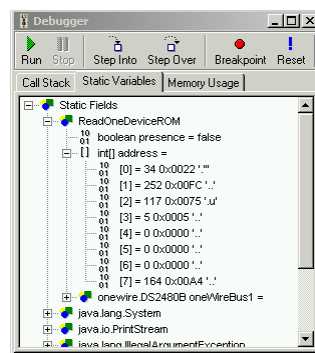
1.3 (a) and (b). Keep in mind that each device's ROM address is unique by design, so it is very unlikely that your numbers will match the ones in the screen captures.

- ✓ Use the Javelin Stamp IDE to Open DS2480BReadOneDS1822ROM from the "...Projects\examples\peripheral\onewirecom" directory.
- ✓ Click the Javelin Stamp IDE's Debug button.
- ✓ Click the gray margin to the left of the second **reset** method call to set a breakpoint.
- ✓ Click the Debugger's Run button.
- ✓ When the Javelin Stamp IDE stops at the breakpoint, use the Debugger to view the address array's contents as shown in Figure 1.3 (a).

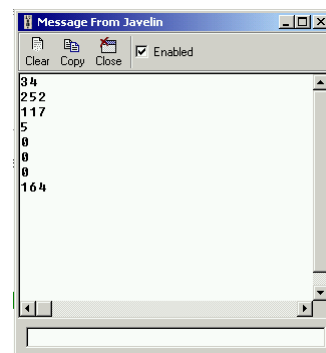
The Debugger window displays the decimal values to the left of their hexadecimal equivalents. The hexadecimal values are useful, especially since that's what Maxim uses in their documentation to explain transactions between a controller and their 1-Wire devices.

- ✓ Click the Debugger's Run button a second time to see the decimal version of the ROM address in the messages from Javelin window as shown in Figure 1.3 (b).

Figure 1.3
ROM address in
the (a) Debugger
and (b)
Messages from
Javelin window



(a)



(b)

Program Listing 1.1 – Read One Device ROM

```
import stamp.peripheral.onewire.*;
import stamp.core.*;

/**
 * Class demonstrates using the DS2480B object to read the
 * ROM of a single device on a 1-Wire bus.
 * <p>
 * @version 1.2 8/28/02
 * @author Parallax, Inc.
```

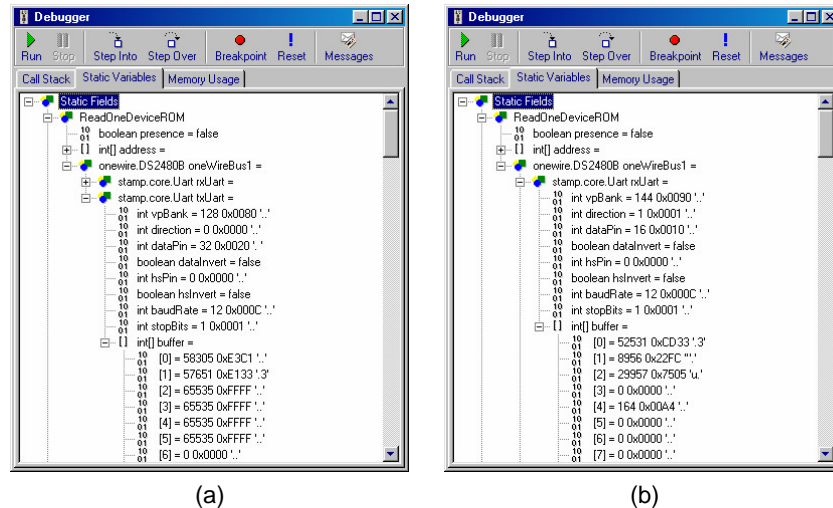
```
*/  
  
public class DS2480BReadOneDS1822ROM {  
  
    // 0x33 = READ_ROM command.  
    final static int READ_ROM = 0x33;  
  
    // Field for storing the presence response from the reset() method.  
    static boolean presence;  
  
    // Array for storing the 1-Wire device's address.  
    static int [] address = new int[8];  
  
    // Declare I/O pins to be used by the DS2480B's Uart transmit & receive lines.  
    public final static int SERIAL_TX_PIN = CPU.pin5;  
    public final static int SERIAL_RX_PIN = CPU.pin4;  
  
    // Constructor that declares a new DS2480B object and initializes it with  
    // the I/O pins used for the transmit and receive Uarts.  
    private static DS2480B oneWireBus = new DS2480B(SERIAL_TX_PIN, SERIAL_RX_PIN);  
  
    public static void main() {  
        oneWireBus.reset();  
        oneWireBus.message(READ_ROM);  
        oneWireBus.getData(8,address,READ_ROM);  
        oneWireBus.reset();  
        for(int i = 0; i < 8; i++){  
            System.out.println(address[i]);  
        }  
    }  
}
```

Trouble-Shooting

You can re-run the program and examine the contents of the Uart transmit and receive buffers to see what's going on under the hood. This is especially useful for verifying the information in the DS2480B and DS1822 data sheets and for trouble shooting.

- ✓ Click the Debugger's Run button a third time.
- ✓ When the program reaches the breakpoint, use the Debugger window to look into the txUart and rxUart serial buffers shown in Figure 1.4 (a) and (b).

Figure 1.4
(a) txUart buffer and
(b) rxUart buffer



The **txUart** and **rxUart** buffers show a history of the characters sent and received by the Javelin Stamp. Here's a narrative of the transmit and receive sequence:

- The first byte transmitted by the **txUart** is 0xE3, which places the DS2480B in COMMAND mode.
- 0xC1 causes the DS2480B to send a reset pulse to the 1-wire bus.
- If a presence pulse is received in reply, the DS2480B sends back a 0xCD byte (shown in the **rxUart** buffer).
- Sending 0xE1 puts the DS2480B back into DATA mode.
- The Character 0x33 (**READ_ROM**) is sent to the 1-Wire bus. This value is also echoed back by the DS2480B and received and stored in the **rxUart** buffer.
- The **txUart** issues Eight 0xFF characters. These characters are sent by the DS2480B object's **getData** method.
- Each time a 0xFF is sent by the Javelin Stamp, the device on the 1-Wire bus responds with the next byte's worth of its 64-bit ROM address. Each value is relayed by the DS2480B to the Javelin Stamp on the **rxUart** line and stored in the Javelin Stamp's **rxUart** buffer.

For trouble-shooting, there are a few different sequences that you may see in the **rxUart** buffer if there are wiring or pin definition problems.

- If the **rxUart** buffer contents shows all 0xFF values where the address should be, it indicates that the DS1822 is not responding. If this is the case, you will probably also see a 0xCF response to the reset pulse instead of 0xCD.

- If the `rxUart` and `txUart` buffers contain the same characters, check to make sure your `SERIAL_TX_PIN` and `SERIAL_RX_PIN` connections are not reversed.
- If the `rxUart` buffer contains all zeros, the DS2480B may not be functioning properly due to a wiring problem.

Application Example – Collecting 1-Wire Temperature Data

Program Listing 1.2 can be used to read and display the temperature measured by a DS1822 on the 1-Wire bus. Remember that the DS2480B class methods that are called in this program are explained in the DS2480B.pdf javadoc. Here is how the `getTempFromOneDs1822` method in the example program reads the temperature from the DS1822's scratchpad RAM:

- *Deliver a reset pulse to the 1-Wire bus.*
- *Arm the DS2480B's strong pullup resistor.*
- *Send the **SKIP_ROM** message to the 1-Wire bus.*
- *Send the **CONVRERT_T** message to the 1-Wire bus.*
- *Sample the state of the bus periodically by using the `getBit` method. The DS1822 holds the line low while it's busy measuring temperature, and the `getBit` method returns false. When the measurement is done, the DS1822 lets go of the 1-Wire bus, and the `getBit` method returns true.*
- *Disarm the strong pullup resistor.*
- *Deliver another reset pulse to the 1-Wire bus.*
- *Send the **SKIP_ROM** message to the 1-Wire bus.*
- *Send the **READ_SCRATCHPAD** message to the 1-Wire bus.*
- *Collect the first two bytes of scratchpad RAM from the DS1822 on the 1-Wire bus using the `getData` method.*
- *Reset pulse to the 1-Wire bus.*

Keep in mind that the only time it's appropriate to issue the **SKIP_ROM** command is if there is a single device on the bus; otherwise, you have to call out the device's unique address. You can do this by adding an array that contains the DS1822's address. A good place to insert this statement is right before or after the `temperature` array declaration:

```
// REMEMBER to use your own DS1822's unique address, not this one.
final static int [] address = {0x22, 0xFC, 0x75, 0x05,
                               0x00, 0x00, 0x00, 0xA4};
```


Then, replace both instances of

```
oneWireBus.message(SKIP_ROM);
```

with

```
oneWireBus.message(MATCH_ROM);  
oneWireBus.message(8,address);
```

The two bytes read from the DS1822's scratchpad RAM contain the temperature, but it's stored in a binary format that's not all that friendly for displays. The example program's **formatAndDisplay** method converts the binary value into a useful degrees-Celsius measurement. This method examines the upper 12 bits of the two temperature bytes and displays them as the integer value to the left of the decimal point. The lower four bits are then converted to the value that goes to the right of the decimal point. To view these measurements in the Messages from Javelin window:

- ✓ Load DS2480BReadOneDS1822Scratchpad.java into the Javelin Stamp IDE.
- ✓ Click the Javelin Stamp IDE's Run button.

Program Listing 1.2 – Read One Device Temperature

```
import stamp.peripheral.onewire.*;  
import stamp.core.*;  
  
/**  
 * Class demonstrates using the DS2480B object to acquire temperature  
 * information from the scratchpad RAM of a single DS1822 device on  
 * a 1-Wire bus.  
 *  
 * @version 1.2 11-20-02  
 * @author Parallax, Inc.  
 */  
  
public class DS2480BReadOneDS1822Scratchpad {  
  
    // Constants used for communication with the DS1822 on the 1-Wire bus.  
    final static int SKIP_ROM = 0x00CC;  
    final static int CONVERT_T = 0x0044;  
    final static int READ_SCRATCHPAD = 0x00BE;  
    final static int MATCH_ROM = 0x55;  
  
    // Field for storing the presence response from the reset() method.  
    static boolean presence;  
  
    // Field for storing the two temperature bytes sent by the DS1822.
```

```
static int [] temperature = new int[2];

// Declare I/O pins to be used by the DS2480B's Uart transmit and
// receive lines.
public final static int SERIAL_TX_PIN = CPU.pin5;
public final static int SERIAL_RX_PIN = CPU.pin4;

// Declare a new DS2480B object and initialize with connections
// for Uart serial communication with a DS2480B chip.
private static DS2480B oneWireBus = new DS2480B(SERIAL_TX_PIN,
                                                SERIAL_RX_PIN);

/**
 * Measures temperature of a single DS1822 device on a 1-Wire bus.
 * IMPORTANT: This method will not work if there is more than one
 * 1-Wire device on the bus.
 */
public static void getTempFromOneDs1822(int [] value){
    // Send reset pulse to 1-wire bus.
    oneWireBus.reset();

    // Arm strong pullup resistor.
    oneWireBus.strongPullup(oneWireBus.INFINITE_DURATION);

    // Send two characters to 1-wire bus using message method.
    oneWireBus.message(SKIP_ROM);
    oneWireBus.message(CONVERT_T);

    // Wait for a bit reading to go from false to true indicating that
    // the temperature measurement is ready.
    while(!oneWireBus.getBit()){
        CPU.delay(500);
    }

    // Disarm strong pullup resistor.
    oneWireBus.strongPullup(oneWireBus.TERMINATE_DURATION);

    // Issue another reset pulse followed by a skip ROM command.
    oneWireBus.reset();
    oneWireBus.message(SKIP_ROM);

    // Issue read scratchpad command and load the first two bytes from the DS1822's
    // scratchpad into the temperature array. Then, deliver a final reset pulse.
    oneWireBus.message(READ_SCRATCHPAD);
    oneWireBus.getData(2,temperature,READ_SCRATCHPAD);
    oneWireBus.reset();
}
```

```
/**
 * Formats and displays raw temperature data that was read from
 * the DS1822's scratchpad RAM.
 */
public static void formatAndDisplay(int [] a){
    int tempLo, tempHi, tempInt, tempDec;
    tempLo = a[0];
    tempHi = a[1];

    // Move the bits of the integer value of the measurement into
    // a single field.
    tempInt = tempLo + (tempHi << 8);

    // Convert the lowest four bits into the fractional decimal value that goes
    // to the right of the decimal point and the upper eight bits into the integer
    // decimal value to the left of the decimal point.
    tempDec = (((tempInt & 0x000F) * 100)/16);
    tempInt = tempInt >> 4;

    System.out.print(tempInt);
    System.out.print(".");

    // Maintain a leading zero if the fractional decimal measurement is less
    // than 10.
    if(tempDec < 10){
        System.out.print("0");
    }
    System.out.print(tempDec);
    System.out.println(" degrees-Celsius");
}

public static void main() {
    while(true){
        getTempFromOneDs1822(temperature);
        // Clear Messages from Javelin display.
        System.out.print('\u0010');
        formatAndDisplay(temperature);
    }
}
}
```

Parasite Power Mode Wiring and Coding Tips

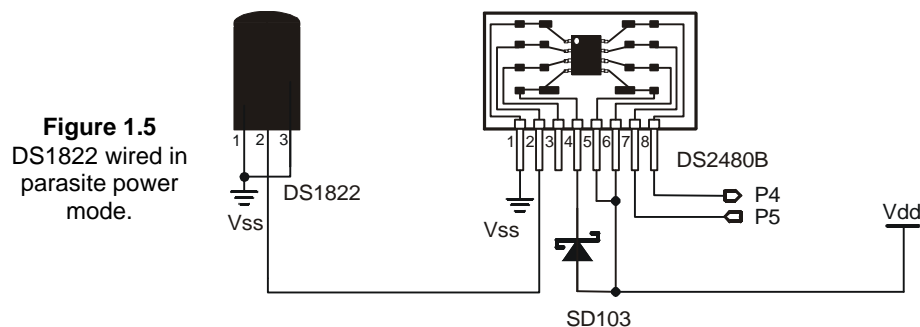
You can also run the DS1822 in “parasite power mode” as shown in Figure 1.5, and the DS1822 will draw its supply current from the 1-Wire bus’ signal line. This wiring scheme cannot be used in conjunction with the technique of polling the bus to find out when the temperature measurement is complete because it causes the `getBit` method to return true, even while the DS1822 is processing the temperature measurement. Since the DS1822’s data sheet states that the longest a 12-bit temperature measurement will take is 750 ms, you can replace

```
while(!oneWireBus.getBit()){  
    CPU.delay(500);  
}
```

with

```
CPU.delay(7500);
```

to guarantee that the temperature measurement will be complete before your Javelin Stamp attempts to read it from the DS1822’s scratchpad.



Published Resources – for More Information

“SM-EZ-101”, Bulletin, Capital Advanced Technologies Inc., 1994.

The Capital Advanced Surface Mount EZ-101 document shows how to solder surface mount parts onto their surface mount adaptor PCBs. This bulletin is available for download from the Capital Advanced web site: www.capitaladvanced.com

“DS2480B Serial 1-Wire Line Driver with Load Sensor”, Datasheet, Maxim/Dallas Semiconductor, October, 1999.

The DS2480B data sheet supplied the basic information for developing the DS2480B class and this application note. It also has some useful pseudo-code examples for communicating with other 1-Wire devices. This data sheet is available from www.maximic.com.

***“DS1822 Econo 1-Wire Digital Thermometer”*, Datasheet, Maxim/Dallas Semiconductor, June, 2002.**

The DS1822 data sheet was used to develop the circuits and code samples in this application note. This datasheet is also available from www.maximic.com.

***“Javelin Stamp Manual”*, Users Manual, Version 1.0, Parallax, Inc., 2002**

The Javelin Stamp Manual available from www.javelinstamp.com contains more information about the Uart object and other techniques used in this application note’s example programs.

***“BASIC Stamp Manual”*, Users Manual, Version 2.0c, Parallax, Inc., 2000**

The BASIC Stamp Manual available from www.parallaxinc.com contains more information about the 1-Wire protocol with PBASIC examples for the BASIC Stamp 2p.

Javelin Stamp Discussion Forum – Questions and Answers

The Parallax, Inc. Javelin Stamp Discussion Forum is a searchable repository of questions and answers about the Javelin Stamp and Javelin Stamp Applications. The DS2480B class and this application note came about because of a question posted to this list that asked if the Javelin Stamp supported the 1-Wire protocol. To view the Javelin Stamp Forum, go to www.javelinstamp.com and follow the Discussion link. You can also join this forum and post your own questions. Other Javelin Stamp users who monitor the list will see your questions and reply with helpful tips, part numbers, pointers to useful web pages, etc.

Copyright © 2002 by Parallax, Inc. All rights reserved. Javelin, Stamp, and PBASIC are trademarks of Parallax, Inc., and BASIC Stamp is a registered trademark of Parallax, Inc. Windows is a registered trademark of Microsoft Corporation. Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. 1-Wire is a registered trademark of Maxim Integrated Products, Inc. Other brand and product names are trademarks or registered trademarks of their respective holders.

Parallax, Inc. is not responsible for special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, goodwill, damage to or replacement of equipment or property, and any costs of recovering, reprogramming, or reproducing any data stored in or used with Parallax products.