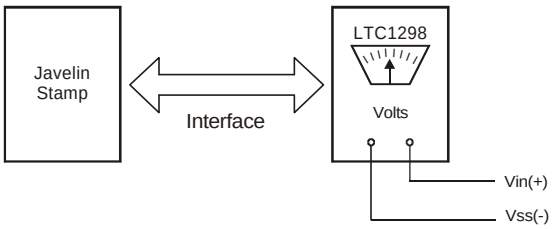
599 Menlo Drive, Suite 100
Rocklin, California 95765, USA
Office/Tech Support: (916) 624-8333
Fax: (916) 624-8003

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

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



Contents

Introduction to the LTC1298 12-bit Analog to Digital Converter

The LTC1298 is a 2 channel 12-bit analog to digital converter (A/D converter). An A/D converter will read an analog voltage and give you a corresponding digital number. This number can be easily converted into the equivalent voltage measurement.

How the LTC1298 Works

The LTC1298 reads a voltage from 0 to 5 volts and will return a *raw* value which corresponds to an interval from 0 to 4095 (12-bits). The Javelin will take this *raw* value and calculate the voltage level it represents. The

highest voltage the chip can read is 4.998 volts. Each increment (bit) is 1.220703125 millivolts in width. There are 4,095 increments from 0 to 5 volts. The LTC1298 measures a voltage, determines which increment it is closest to and gives you this result. Therefore, there is an error of ±1.220703125/2 or 0.6103515625 millivolts.

The LTC1298 has 2 channels, so it can read two separate voltages. If you read each of these voltages and compared them manually there is a possibility of a 1.220703125 millivolt error. It is best when comparing the two channels to allow the LTC1298 to do the comparison itself.

The LTC1298 returns a *raw* value of 0 to 4095. The Javelin will take this *raw* value and calculate the correct voltage measurement.

Downloads, Parts, and Equipment for the LTC1298

This application note (AppNote008-LTC1298.java), the LTC1298 library file (LTC1298.java), the library's javadoc file (LTC1298.pdf), the test program (LTC1298Test.java), the demonstration program (LTC1298Demo.java) which will demonstrate the methods available to you for the LTC1298, and an application example (LTC1298VoltageCompare.java) are all available to you for free download from:

http://www.javelinstamp.com/Applications.htm

You can use the AppNote008-LTC1298.exe, to install the files listed below. These files must be located in specific paths within the Javelin Stamp IDE directory. Although the path to this directory can be different, the default root path is: C:\Program Files\Parallax Inc\Javelin Stamp IDE

The file list below is organized by directory, then by filename; please verify that your file list is organized in the same way.

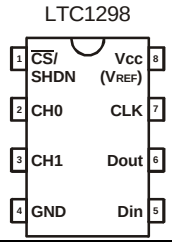
<root path>\doc\AppNote008-LTC1298.pdf
<root path>\lib\stamp\peripheral\io\ADC\LTC1298.java
<root path>\doc\LTC1298.pdf
<root path>\Projects\examples\peripheral\io\ADC\LTC1298Test.java
<root path>\Projects\examples\peripheral\io\ADC\LTC1298Demo.java
<root path>\Projects\examples\peripheral\io\ADC\LTC1298VoltageCompare.java

In addition to the files above, the AtoD abstract library class file is also required. The AtoD.java file can be found in AppNote006 and must be installed in the following directory:

<root path>\lib\stamp\peripheral\io\ADC\AtoD.java

Table 1.1 lists the parts you will need for this application note.

Table 1.1: Parts List

Quantity	Part Ordering Info and Part Description	Schematic Symbol/ Pin Map
1	Linear Technology's LTC1298 Parallax Part #604-00001	
1	10 µF tantalum Capacitor Parallax Part #202-01060	
1	1 kΩ Resistor Parallax Part #150-01020	
2	100 kΩ Potentiometer Parallax Part #152-01040	

The equipment used to test this example includes a Javelin Stamp, Javelin Stamp Demo Board, 7.5 V, 1000 mA DC power supply, serial cable, and PC with the Javelin Stamp IDE v2.01.

LTC1298 Example Circuit

Figure 1.1 is the circuit that will be used with Program Listing 1.1. Each potentiometer is connected to a separate channel of the LTC1298 chip. By turning a potentiometer you will introduce a voltage (0-5 V) to one of the channels the LTC1298 is measuring. The channels are software selectable; once the channel is selected, the LTC1298 will read the voltage then send a numeric number that represents this voltage value to the Javelin Stamp. The Javelin Stamp will transmit this message to the IDE's *messages from the Javelin window*.

Here's how you connect the LTC1298 to the Javelin Stamp (see Figure 1.1).

- Pin 1, the chip select pin (CS) of the LTC1298 is connected to pin 9 (P4) of the Javelin Stamp.
- Pin 2, the channel 0 pin (CH0) of the LTC1298 is connected to the *wiper* of a potentiometer. The other two leads of the potentiometer are connected to ground (Vss) and +5 V (Vdd).
- Pin 3, the channel 1 pin (CH1) of the LTC1298 is connected to the *wiper* of a potentiometer. The other two leads of the potentiometer are connected to ground (Vss) and +5 V (Vdd).
- Pin 4, the ground pin (GND) of the LTC1298 is connected to ground (Vss).
- Pin 5, the data in pin (Din) is connected to pin 6 (P1) of the Javelin Stamp.
- Pin 6, the data out pin (Dout) of the LTC1298 is connected to a 1 kΩ resistor. The resistor is then connected to pin 5, the data in pin (Din), of the LTC1298.
- Pin 7, the clock pin (CLK) of the LTC1298 is connected to pin 7 (P2) of the Javelin Stamp.
- Pin 8, the power pin (Vcc) of the LTC1298 is connected to +5 V (Vdd). This pin is also connected to the positive lead of a 10 uF tantalum capacitor. The negative lead of this capacitor is connected to ground (Vss).

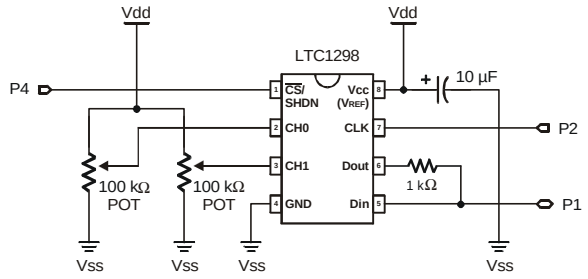


Figure 1.1
Wiring diagram
for the LTC1298

Testing the LTC1298 Circuit

Program Listing 1.1 is a short program that will verify that the circuit in Figure 1.1 is working properly. This program will display the current voltage being read from each potentiometer. When the program is executed it will create an LTC1298 object called **adc**. This object contains the methods from both the LTC1298.java and the AtoD.java libraries.

```
LTC1298 adc = new LTC1298(CPU.pin1,CPU.pin2,CPU.pin4);// Create ADC object
```

Next, the program will clear the Javelin's message window by printing the value **CLS**.

```
System.out.print(CLS);                  // Clear the display
```

Next, the program will enter a **do** loop, and then by printing the **HOME** value it will position the cursor to the top-left corner of the screen. This is done so the output data from the Javelin will remain in one area of the screen.

```
System.out.print(HOME);              // position cursor
```

Now the program will read the ADC on channel 0. The data that was read from the ADC will be stored within the **adc** object and must be called to be displayed.

```
adc.read(0);                         // read and store values
```

To recall the values from the **adc** object and display them in the message window, the program executes the **lastVf()** method within a print statement. This method will display the last read voltage with the proper formatting for the decimal.

```
System.out.print("Channel 0: ");
System.out.print(adc.lastVf());      // display formatted volts
System.out.println(" volts ");
```

Next the program will read the ADC on channel 1. Notice the only change was the parameter that was passed into the read method.

```
adc.read(1);                         // read and store values
```

The values from the **adc** object are recalled the same way as with channel 0.

```
System.out.print("Channel 1: ");
System.out.print(adc.lastVf());      // display formatted volts
System.out.print(" volts ");
```

The test program (Program Listing 1.1) will display output, to the Javelin's IDE message window, similar to Figure 1.2.

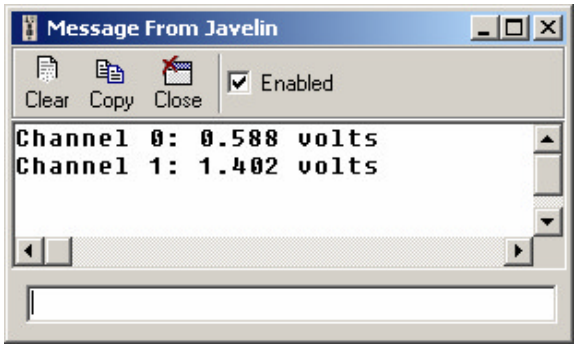


Figure 1.2
LTC1298 Test output window

The values will be different, but your screen should look similar as above. If not, try the following:

- ✓ Verify that the potentiometer you are using is rated at 100 KΩ.
- ✓ Carefully verify that each pin on your LTC1298 chip match the circuit shown in Figure 1.1.

Tip Verify with a Voltmeter	You can also use a voltmeter to verify that the output window matches the actual voltage. To do this you would connect the ground lead of your voltmeter (COM) to the ground side of the potentiometer (Vss). Connect the positive lead of your voltmeter to the wiper leg of the potentiometer (CH0/CH1). Select the proper setting on the voltmeter. If the voltage shown is negative then the connection is backwards. To fix this, simply switch the two leads.

Program Listing 1.1 – The LTC1298Test

```
import stamp.core.*;
import stamp.peripheral.io.ADC.*;

/*
 * Tests the LTC1298 circuit from the Application Note #008.
 * Version 1.0 - 11/29/02
 */

public class LTC1298Test {

  final static char HOME = 0x01;            // Position cursor upper-left
  final static char CLS = '\u0010';         // Clear Screen

  public static void main() {
    LTC1298 adc = new LTC1298(CPU.pin1,CPU.pin2,CPU.pin4);  // Create ADC object
    System.out.print(CLS);                                  // Clear the display

    do {
      System.out.print(HOME);               // position cursor
```

```
      // Test channel 0
      adc.read(0);                          // read channel 0, store values
      System.out.print("Channel 0: ");
      System.out.print(adc.lastVf());       // display formatted volts
      System.out.print(" volts ");

      // Test channel 1
      adc.read(1);                         // read chennel 1, store values
      System.out.print("\nChannel 1: ");
      System.out.print(adc.lastVf());      // display formatted volts
      System.out.print(" volts ");

    } while (true);                        // do forever

  }// end method: main
}// end class: LTC1298Test
```

Extra Features Built into the LTC1298

The LTC1298 can measure the difference between the voltages on each channel (CH0 and CH1). Normally to read from channel 0 (CH0), pass the value 0 into the read method **`read(0)`**. To read from channel 1 (CH1), pass the value 1 into the read method **`read(1)`**. But if you would like to compare the difference between the two channels, pass a value 2 or 3 into the read method **`read(2)`** or **`read(3)`**.

Passing a 2 will subtract channel 1 (CH1) from channel 0 (CH0). Channel 0 (CH0) must be of a higher voltage than channel 1 (CH1); if not, the ADC object will return zero.

Passing a 3 will subtract channel 0 (CH0) from channel 1 (CH1). Channel 1 (CH1) must be of a higher voltage than channel 0 (CH0); if not, the ADC object will return zero.

The LTC1298 Library

The LTC1298 library extends methods from the AtoD abstract library. The AtoD abstract library contains methods that are general in nature and can work with a variety of ADC chips. The LTC1298 library has specific information for the LTC1298 chip, the AtoD abstract library uses this information and data within its methods.

The LTC1298 has one constructor and two methods. The constructor will create a LTC1298 communication bus using the three pins from the Javelin that you indicate. The method **`setOffset()`** will allow you to adjust the bit value to obtain a more accurate reading if needed. The method **`read()`** will pass the correct command to the LTC1298 chip and obtain the *raw* value from the chip and convert it into the calculated voltage.

The AtoD Abstract Library

The AtoD abstract library, sometimes called a super class, is a collection of methods common to ADC chips. It receives specific information for the LTC1298 chip from the LTC1298 library. When an LTC1298 object is created all methods within the AtoD library will be included for your use.

Compare the LTC1298 Voltages

Program Listing 1.2 uses the same circuit described in Figure 1.1 and will read the voltages on channel 0, channel 1 and measure the difference between both channels as described in the section above in *Extra Features Built into the LTC1298.*

The output shown in Figure 1.3 should appear in the Javelin window from the IDE. If it does not, please verify your circuit matches the circuit shown in Figure 1.1.

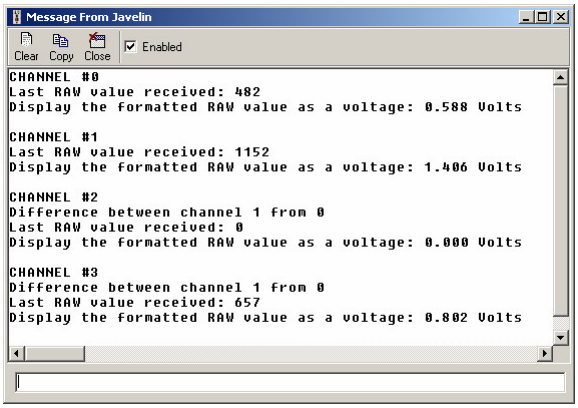


Figure 1.3
IDE Javelin window

The program will display the *raw* value received from the LTC1298 chip, as well as the calculated voltage for each channel 0-3. Channels 2 and 3 are not individual channels, they are used to compare the voltage between the channels. For more information on this see the section *Extra Features Built into the LTC1298.*

This program uses the methods below from the LTC1298 and AtoD libraries.

- **`LTC1298(CPU.pin1,CPU.pin2,CPU.pin4)`**
  Constructor to create the adc object and bus.
  - CPU.pin1 = data pin, I/O pin P1 on the Javelin
  - CPU.pin2 = clock pin, I/O pin P2 on the Javelin
  - CPU.pin4 = chip select pin, I/O pin P4 on the Javelin

- **`read(channel)`**
  Reads *raw* value from the LTC1298 and stores it for later retrieval (see **`lastRaw()`**, **`lastMV()`** and **`lastVf()`**).
  - channel = Specify channel:
    Use 0 to read from channel 0.
    Use 1 to read from channel 1.
    Use 2 to compare between channel 1 from 0.
    Use 3 to compare between channel 0 from 1.

- **`lastRaw()`**
  Will display the last read *raw* value from the LTC1298 chip.

- **`lastVf()`**
  Will display last known measurement as a formatted voltage.

Tip	Use the javadoc help files, in your <"Javelin Stamp IDE\doc"> folder, LTC1298.pdf and the AtoD.pdf *(from AppNote006)* as a reference to find out more about using the methods available to you.

Program Listing 1.2 – LTC1298VoltageCompare

```
import stamp.core.*;
import stamp.peripheral.io.ADC.*;

/*
 * This class will fully demonstrate the LTC1298 chips abilities.
 * It will read the individual channels 0 and 1 as well as compare the voltage
 * difference between them.
 *
 * Version 1.0 - 11/29/02
 */

public class LTC1298VoltageCompare {

  final static char HOME = 0x01;                // Position cursor upper-left
  final static char CLS = '\u0010';             // Clear Screen

  public static void main() {
    System.out.print(CLS);                      // position cursor

    // Create a new LTC1298 object: pin1=data, pin2=clock, pin4=Chip Select
    LTC1298 adc = new LTC1298(CPU.pin1,CPU.pin2,CPU.pin4);

    while(true) {                               // do forever
```

```
      System.out.print(HOME);                    // position cursor

      // Loop through all channels
      for (int x=0;x<4;x++){

        adc.read(x);                             // read channel 'x'

        System.out.print("CHANNEL #");
        System.out.print(x);

        if (x==2)System.out.print("\nDifference between channel 1 from 0");
        else if (x==3)System.out.print("\nDifference between channel 1 from 0");

        // Will display the last read RAW value from the LTC1298 chip
        System.out.print("\nLast RAW value received: ");
        System.out.print(adc.lastRaw());
        System.out.println("   ");

        // Will display last known RAW value as a formatted voltage
        System.out.print("Display the formatted RAW value as a voltage: ");
        System.out.print(adc.lastVf());          // display volts
        System.out.print(" Volts");
        System.out.println("   \n");
      }// end for

    }// end while
  }// end method: main
}// end class: LTC1298VoltageCompare
```

LTC1298 Demo

LTC1298Demo.java is a detailed, step-by-step, fully comprehensive demonstration of the LTC1298 and AtoD library classes which is included with this application note.

Published Resources – for More Information

This class was developed to allow the Javelin to interact with the LTC1298 chip. Much care has been taken in creating this class. Below you will find useful information that was used in creating this document.

"LTC1286/LTC1298", Data Sheet, Linear Technology Corporation, 1994

This document will give you detailed information about the internal working of the LTC1298.

"AN006 – Analog To Digital", Application Note, Parallax Inc., Dec. 2002

This application note will explain how you can use the provided abstract class for your own specific ADC by using the ADC0831 as an example.

Javelin Stamp Discussion Forum – Questions and Answers

The Parallax, Inc. Javelin Stamp Discussion Forum is a searchable repository of design questions and answers for the Javelin Stamp. 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. The Parallax technical staff, and Javelin Stamp users who monitor the list, will see your questions and reply with helpful tips, part numbers, pointers to useful web pages, etc.