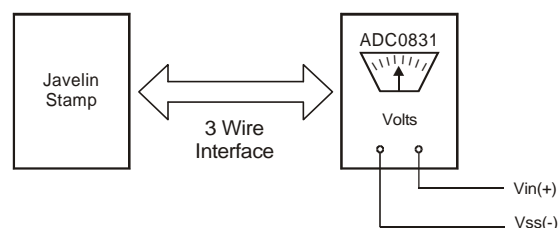


599 Menlo Drive, Suite 100
Rocklin, California 95765, USA
Office/Tech Support: (916) 624-8333
Fax: (916) 624-8003

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 ADC0831 8-bit Analog to Digital Converter	1
How the ADC0831 Works	2
Downloads, Parts, and Equipment for the ADC0831	2
ADC0831 Example Circuit	3
Testing the ADC0831 Circuit	4
Program Listing 1.1 – The ADC0831Test.....	5
Extra Features Built into the ADC0831	6
The ADC0831 Variable Voltage Demonstration	7
Program Listing 1.2 – ADC0831CustomBitValue	9
ADC0831 Demo	10
Published Resources – for More Information	10
Javelin Stamp Discussion Forum – Questions and Answers.....	11

Introduction to the ADC0831 8-bit Analog to Digital Converter

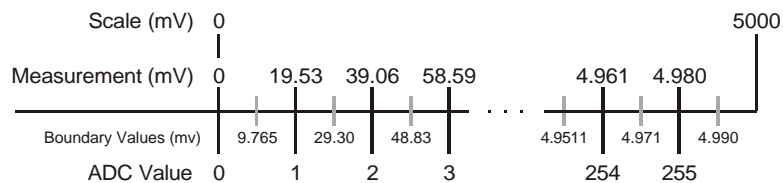
The ADC0831 is an 8-bit analog to digital converter (A/D converter) with synchronous serial output. It will read an analog voltage and give you the digital equivalent. This data that is sent back can be easily converted into the corresponding voltage.

The ADC0831 library class will interact with the ADC0831 chip, and convert the value it receives from the chip into a measurement in millivolts. Converting data into a temperature format is available when reading from temperature devices.

How the ADC0831 Works

The ADC0831 reads a voltage from 0 to 5 volts and will return a *raw* value which corresponds to an interval from 0 to 255 (8-bits). The Javelin will take this *raw* value and calculate the voltage level it represents (see Figure 1.1). The highest voltage the chip can represent is 4.980 volts. Each increment (bit) is approximately 19.53125 millivolts in width. The ADC0831 measures a voltage, determines which increment it is closest to and gives you this result. Therefore, there is a possible error of $\pm 19.53125/2$ or 9.765625 millivolts.

Figure 1.1
Voltage scale for
an 8-bit ADC

**Downloads, Parts, and Equipment for the ADC0831**

This application note (AppNote007-ADC0831.pdf), the ADC0831 library file (adc0831.java), the library's javadoc file (adc0831.pdf), the test program (adc0831Test.java), the demonstration program (adc0831Demo.java) which will demonstrate the methods available to you for the ADC0831, and a program to explore custom bit values (adc0831CustomBitValue.java) are all available to you for free download from:

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

You can use the AppNote007-ADC0831.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.

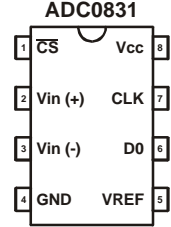
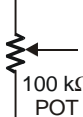
```
<root path>\doc\AppNote007-ADC0831.pdf
<root path>\lib\stamp\peripheral\io\ADC\ADC0831.java
<root path>\doc\ADC0831.pdf
<root path>\Projects\examples\peripheral\io\ADC\ADC0831Demo.java
<root path>\Projects\examples\peripheral\io\ADC\ADC0831Test.java
<root path>\Projects\examples\peripheral\io\ADC\ADC0831CustomBitValue.java
```

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
1	National Semiconductor's ADC0831 Parallax Part #ADC0831	
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.

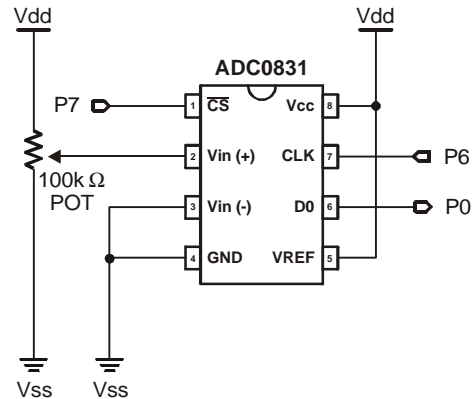
ADC0831 Example Circuit

Figure 1.2 is the circuit that will be used with Program Listing 1.1. Notice that there is only one potentiometer, the other potentiometer will be used for Program Listing 1.2. By turning the potentiometer you will introduce a voltage (0-5 V) to the ADC0831. The ADC0831 will read this voltage and send a numeric number that represents this 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 ADC0831 to the Javelin Stamp (see Figure 1.2).

- Pin 1, the chip select pin (CS) of the ADC0831 is connected to pin 12 (P7) of the Javelin Stamp.
- Pin 2, the positive input voltage pin ($V_{in}^{(+)}$) of the ADC0831 is connected to the potentiometer's *wiper*. The other two leads of the potentiometer are connected to ground (V_{ss}) and +5 V (V_{dd}).
- Pin 3, the negative input voltage pin ($V_{in}^{(-)}$) of the ADC0831 is connected to ground (V_{ss}).
- Pin 4, the ground pin (GND) of the ADC0831 is connected to ground (V_{ss}).
- Pin 5, the voltage reference pin (Vref) is connected to +5 V (V_{dd}).
- Pin 6, the data out pin (DO) of the ADC0831 is connected to pin 5 (P0).
- Pin 7, the clock pin (CLK) of the ADC0831 is connected to pin 11 (P6) of the Javelin Stamp.
- Pin 8, the power pin (Vcc) of the ADC0831 is connected to +5 V (V_{dd}).

Figure 1.2
Wiring diagram for the
ADC0831



Testing the ADC0831 Circuit

Program Listing 1.1 is a short program that will verify that the circuit in Figure 1.2 is working properly. This program will display the current voltage being read from the potentiometer. When the program is executed it will create an ADC0831 object called **adc**. This object contains the methods from both the ADC0831.java and the AtoD.java libraries.

```
ADC0831 adc = new ADC0831(CPU.pin0,CPU.pin6,CPU.pin7); //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. The data that was read from the ADC will be stored within the **adc** object and must be called to be displayed. You will need to pass in a 0 value for this method to work correctly with the ADC0831 chip.

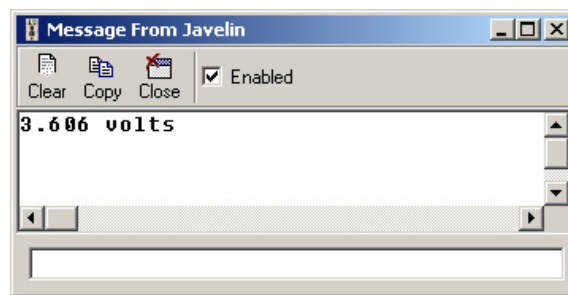
```
adc.read(0); // read and store values
```

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 in decimal.

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

The test program (Program Listing 1.1) will display output, to the Javelin's IDE message window, similar to Figure 1.3.

Figure 1.3
ADC0831 Test
output window



If your output voltage value is zero (or not accurate), try the following:

- ✓ Verify that the potentiometer you are using is rated at 100 K.
- ✓ Carefully verify that each pin on your ADC0831 chip match the circuit shown in Figure 1.2.

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 (Vin⁽⁺⁾). 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 ADC0831Test

```
import stamp.core.*;  
import stamp.peripheral.io.ADC.*;  
  
/*  
 * This class tests the ADC0831 circuit from the Application Note #007.  
 * Version 1.0 - 12/12/02  
 */
```

```
public class ADC0831Test {  
  
    final static char HOME = 0x01;           // Position cursor upper-left  
    final static char CLS = '\u0010';       // Clear Screen  
  
    public static void main() {  
        ADC0831 adc = new ADC0831(CPU.pin0,CPU.pin6,CPU.pin7); // create ADC object  
        System.out.print(CLS);               // Clear the display  
        do {  
            System.out.print(HOME);           // position cursor  
            adc.read(0);                       // read and store values  
            System.out.print(adc.lastVf());    // display formatted volts  
            System.out.println(" volts ");  
        } while (true);                       // do forever  
    }  
}  
// end method: main  
// end class: ADC0831Test
```

Extra Features Built into the ADC0831

You can customize the range the ADC0831 measures by setting voltages at the $V_{in}^{(-)}$ and V_{ref} pins. *Doing this will require you to re-calculate the bit value.*

- $V_{in}^{(-)}$ determines the lowest voltage the ADC0831 can measure.
- V_{ref} determines the range (above $V_{in}^{(-)}$) of valid measurements.

For example, if you set $V_{in}^{(-)}$ to 1 volt, and set V_{ref} to 2 volts then the *voltage range* that the chip measures is from 1 to 3 volts. Narrowing the *voltage range* increases the precision of the ADC0831. The ADC0831 always looks at the voltage range as 256 different voltage levels regardless of whether you set your voltage range to a maximum of 5 volts, or 2 volts.

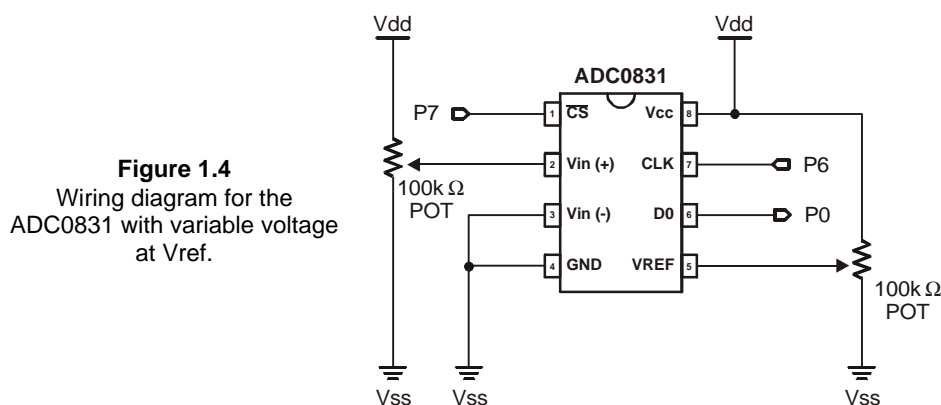
For an example of setting the ADC0831 to its maximum range see Figure 1.2. This circuit has V_{ref} connected to +5 V and $V_{in}^{(-)}$ connected to ground.

Program Listing 1.2 explores this feature of the ADC0831 library. It helps you to custom create your own bit value. The easiest way to change the input voltage to V_{ref} on your circuit is to connect V_{ref} to a 100 K potentiometer, see Figure 1.4 for circuit schematic. You will need to adjust the potentiometer until the voltmeter reads the voltage you want as close as possible. If you do not have access to a voltmeter, you could set the voltage at V_{ref} by using two resistors as a voltage divider. For more information on how to do this you can refer to Parallax book [Basic Analog and Digital's](#) "Build your own DC digital voltmeter" experiment.

Tip

The ADC0831 library cannot determine the voltage at Vref automatically. Unless this voltage is tested with another ADC0831 chip. Therefore, once you set Vref to a particular voltage, you will also need to pass this information to the library class for the correct calculations.

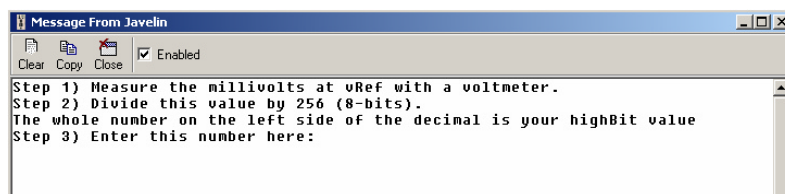
The circuit shown below in Figure 1.4 is very similar to the circuit in Figure 1.2, the only difference is that Vref is not connected to +5 (Vdd) and is instead connected to a 100 K potentiometer.

**The ADC0831 Variable Voltage Demonstration**

Program Listing 1.2 uses the circuit described in Figure 1.4 to change the voltage range of the ADC0831 chip and increase or decrease its precision. Before executing this program, please read the prior section titled, [Extra Features Built into the ADC0831](#).

The output shown in Figure 1.5 should appear in the Javelin window from the IDE. If it does not, please verify the circuit shown in Figure 1.4.

Figure 1.5
IDE Javelin
window



The program will display instructions to help you calculate a new bit value to match the voltage set at Vref. Verify the voltage at Vref with a voltmeter before calculating the bit value.

Once you have calculated and entered the bit value, new output will appear as in Figure 1.6. This will show you the measured voltage at Vref. The program will read 100 measurements from the ADC0831; you can test the accuracy with a voltmeter connected to Vref.

Figure 1.6
IDE Javelin
window



This program uses the methods below from the ADC0831 and AtoD libraries.

- **ADC0831(CPU.pin0,CPU.pin6,CPU.pin7)**
Constructor to create the adc object and bus.
 - CPU.pin0 = data pin, I/O pin P0 on the Javelin
 - CPU.pin6 = clock pin, I/O pin P6 on the Javelin
 - CPU.pin7 = chip select pin, I/O pin P7 on the Javelin
- **setBitValue(highBit,lowBit,0)**
Set the current state of Vref.
 - highBit = High portion of the bit value.
 - lowBit = Low portion of the bit value.
- **read(channel)**
Read *raw* value from the ADC0831 and stores it for later retrieval (see **lastRaw()**, **lastMV()** and **lastVf()**).
 - channel = Specify channel, use zero for the ADC0831 chip.
- **lastVf()**
Will display last known measurement as a formatted voltage.

Tip

Use the javadoc help files, in your <"Javelin Stamp IDE\doc"> folder, ADC0831.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 – ADC0831CustomBitValue

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

/*
 * This class will allow you to set the bit value to match the input voltage
 * on vRef. The voltage on vRef can be set by using a potentiometer connected
 * to the vRef pin. A voltmeter will be needed to watch/verify the volts at vRef.
 * For complete instructions on using this class, refer to Application Note #007.
 *
 * Version 1.0 - 12/12/02
 */

public class ADC0831CustomBitValue {

    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 ADC0831 object: pin5=data, pin6=clock, pin7=Chip Select
        ADC0831 adc = new ADC0831(CPU.pin0,CPU.pin6,CPU.pin7);

        while(true) {                        // do forever
            System.out.println("Step 1) Measure the millivolts at vRef with a voltmeter.");
            System.out.println("Step 2) Divide this value by 256 (8-bits).");
            System.out.println("The whole number on the left side of the decimal is your highBit value");
            System.out.print("Step 3) Enter this number here: ");
            int highBit=getInt();
            System.out.println("\nStep 4) Take the decimal portion and multiply by 65536.");
            System.out.println("If your answer from above is greater than 32767 then,");
            System.out.println("Step 5) subtract 65536.");
            System.out.println("This is your lowBit value, it might be negative");
            System.out.print("Step 6) Enter this value including the negative: ");
            int lowBit=getInt();
            adc.setBitValue(highBit,lowBit,0);

            System.out.print(CLS);
            for (int loop=100;loop>0;loop--){

                System.out.print(HOME);        //Clear display
                System.out.print("Reads remaining: ");
                System.out.print(loop);
                System.out.println(" ");

                // Read the current voltage
                System.out.print("\nMeasured Value: ");
                adc.read(0);                    // 0=ADC chip
                System.out.print(adc.lastVf()); // display formatted volts
                System.out.print(" volts");
            }
        }
    }
}
```

```
        System.out.println("    ");

    } //end for
    System.out.println(CLS);

} // end while
} // end method: main

// Terminal Helper method which reads an integer from the keyboard.
static char keyboardChar;
static StringBuffer keyboardMsg = new StringBuffer(30);

public static int getInt() {
    keyboardMsg.clear();
    while ( (keyboardChar = Terminal.getChar()) != '\r' ) {
        keyboardMsg.append(keyboardChar);
    } // end while
    return Integer.parseInt(keyboardMsg);
} // end method: getInt

} // end class: ADC0831CustomBitValue
```

ADC0831 Demo

ADC0831Demo.java is a detailed, step-by-step, fully comprehensive demonstration of the ADC0831 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 ADC0831 chip. Much care has been taken in creating this class. Below you will find useful information that was used in creating this document.

“ADC0831/ADC0832/ADC0834/ADC0838 8-Bit Serial I/O A/D Converters with Multiplexer Options”, Data Sheet, National Semiconductor, 2002

This document will give you detailed information about the internal working of the ADC0831.

“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.

“Basic Analog and Digital”, Student workbook, Parallax Inc., 1999

This book provides information on how to create a voltage divider using resistors. Valuable if you do not have a voltmeter.

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.

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. 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.