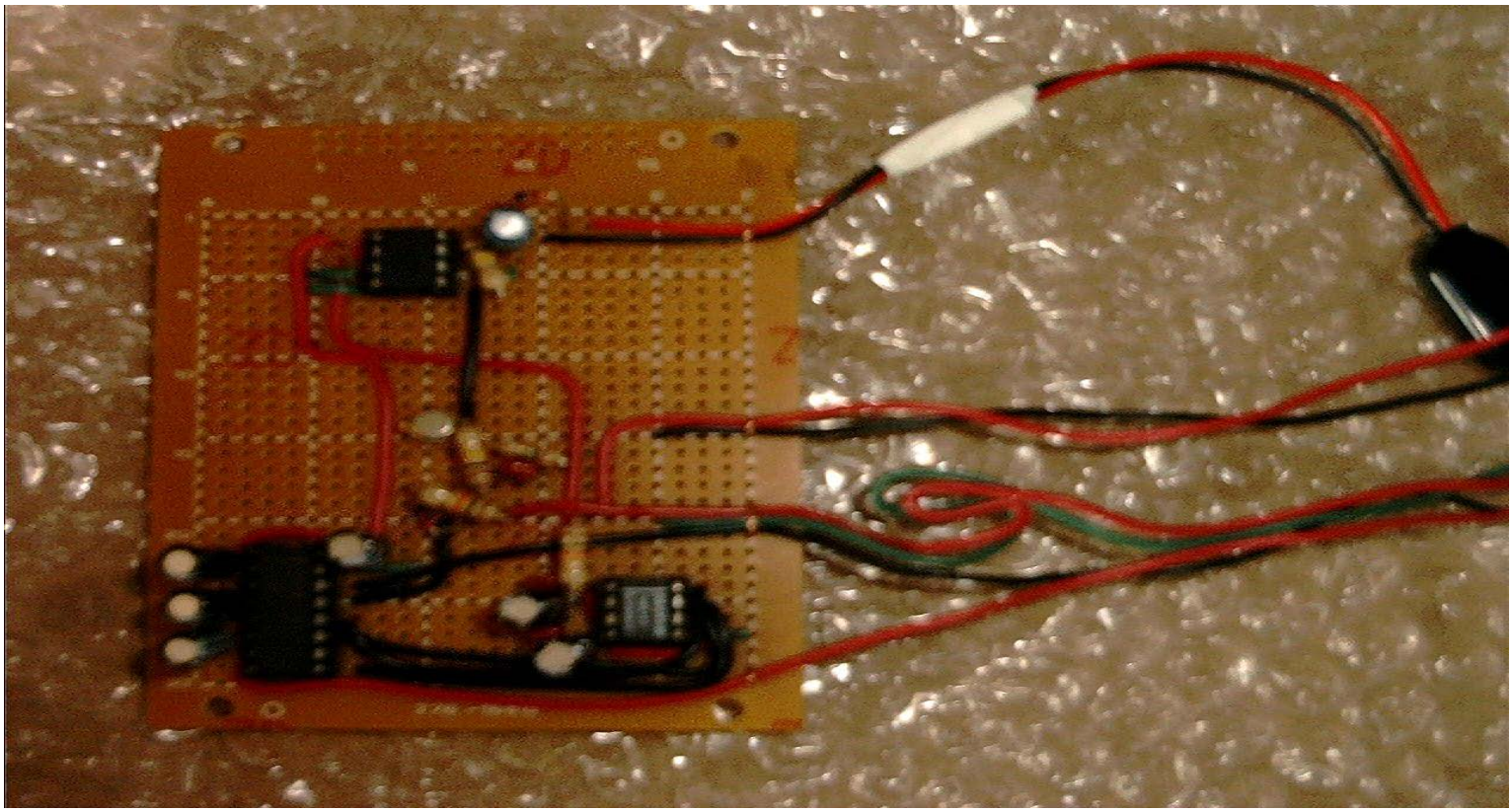
Building An Analog/Digital Converter Circuit (ADC)



What is an ADC, and why do we need one?

Let's start with a simple example: suppose we had an outdoor greenhouse where we wanted to monitor the temperature on a regular basis. Obviously, we would put a thermometer inside the greenhouse and walk out to read the temperature whenever we wanted to know the value. This could obviously become a hassle, especially during inclement weather. To make it more convenient, we could put a remote readout inside our home. In order for the remote readout to work, though, we need to convert the temperature reading into an electrical signal that can be read by the remote readout. This is where an **Analog/Digital Converter (ADC)** comes in.

Here's a basic overview of analog and digital values. Analog values are things we are all familiar with, such as temperature, time, weight, etc. We measure these quantities with specialized instruments (i.e. scales, thermometers) that give readouts in the proper units. Digital values are different, though. They work purely on a binary system of ones and zeroes. To put it simply, they are either "on" or "off". Even a complicated digital device like your home computer works because of a combination of binary values. So how does a device like the remote readout work?

The answer is actually quite simple. It works on the principle of changing electrical currents and voltages. Some specialized electronics components change the currents or voltages in a circuit with a change in temperature. We simply use these components in a circuit that outputs a voltage that is proportional to a changing temperature. Then we use our ADC to change the voltage (an analog signal) into a digital signal. A device such as the remote readout is programmed to display different values for a given digital signal. Although we've skipped over a lot of the finer details, we can now at least understand the concepts behind analog to digital conversion and why it is necessary.

For our data acquisition device, we are using an ADC along with a personal digital assistant (PDA) and a computer in order to make a portable and versatile instrument that is capable of reading many different types of data. In doing so, we hope to make life easier for people who need to collect data on a regular basis.

Instructions for making an ADC

To build the ADC that we used for our project, it is necessary to have some basic knowledge of circuit diagrams and soldering. If you have no prior experience with these items, you may find a basic "do-it-yourself" manual for electric circuitry to be of some assistance.

The first thing you need to do in order to build the circuit is to obtain the necessary electronics components. Most of these components can be obtained from Radio Shack or a similar electronics supply store, with the exception of the IC chips, which can be found at either Newark Electronics or Digi-Key. The IC chips are manufactured by Maxim, which is also the company that supplied the circuit design. **Be sure to read the notes below the parts list regarding some purchasing issues.**

Here is a list of the required components:

Part Description	Quantity	Supplier	Part Number	Price	Packaging
33 µF capacitor	1	Radio Shack	900-1607	$0.08	each
0.1 µF capacitor	4	Radio Shack	272-109	$2.49	pkg. of 5
10 µF capacitor	4	Radio Shack	272-1025	$0.99	each
4.7 µF capacitor	2	Radio Shack	272-1024	$0.99	each
1 Megohm resistor	1	Radio Shack	271-1356	$0.99	pkg. of 5
100k-Ohm resistor	2	Radio Shack	271-1131	$0.99	pkg. of 5
10k-Ohm resistor	2	Radio Shack	271-1126	$0.99	pkg. of 5
1N4148 diode	2	Radio Shack	276-1122	$1.29	pkg. of 10
VN10KE transistor	1	Radio Shack	900-5529	$1.80	each
MAX187 IC chip	1	Digi-Key	MAX187BCPA-ND	$19.80	each
MAX220 IC chip	1	Newark	34C3822	$4.84	each
MAX666 IC chip	1	Digi-Key	MAX666CPA-ND	$4.88	each
9V battery connector	1	Radio Shack	27-324	$1.99	pkg. of 5
8-pin DIP IC socket	2	Radio Shack	276-1995	$0.59	pkg. of 2
16-pin DIP IC socket	1	Radio Shack	276-1998	$0.99	pkg. of 2
PC board	1	Radio Shack	276-168	$2.79	each
Male serial connector			276-1537		
or	1	Radio Shack		$1.49	each
Female serial connector			276-1538		

Notes on parts list:

1) Although the circuit diagram calls for a 35 µF capacitor, it is extremely difficult to locate one. A 33µF capacitor

can be safely substituted into the circuit in its place. Thus, the part number for the 33µF capacitor we used is listed.

2) The same is the case with the VN10K transistor listed on the circuit diagram. The VN10KE transistor listed above was substituted in its place.

3) The part numbers for both male and female serial connectors are shown above. Only one is needed for the circuit, but you need to select the male or female connector depending on the type of port (male or female) to which you are connecting the circuit. If you are not sure which type of connector you need, or if you are connecting to different types of ports, simply purchase the appropriate gender changer in the event that you need to change the connector type.

4) The 8-pin and 16-pin DIP IC sockets are not required for the circuit, but are strongly recommended. One advantage of using them is that there is no risk of overheating the IC chip as when it is soldered directly into the PC board. Also, if an IC chip should ever need to be replaced for some reason, it is much easier to remove and reinsert a chip into the socket than it is to unsolder and resolder a chip directly.

5) The pricing shown above was based on buying the components in as small of a quantity as possible. Bulk pricing may apply to some of the components, but was not used in the price listings above.

6) Samples of the IC chips can be obtained from Maxim by submitting a written request. Doing this will save some money when purchasing components, but do not "abuse" this method. If you are going to use any quantity of the IC chips, bulk pricing can be obtained from the suppliers. Also, beware that it may take a few weeks to receive samples from Maxim, and we did encounter difficulties with part mix-ups (i.e. having the wrong IC chip sent).

You will need soldering equipment (a soldering iron, wire cutters, solder, desoldering braid) and extra wire in order to make connections across the board. Also, a 9V battery is needed in order for the circuit to operate.

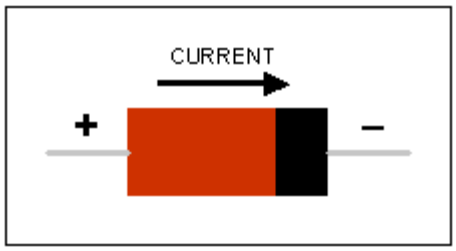
Click here to get the circuit diagram and description from Maxim.

Tips on Circuit Construction

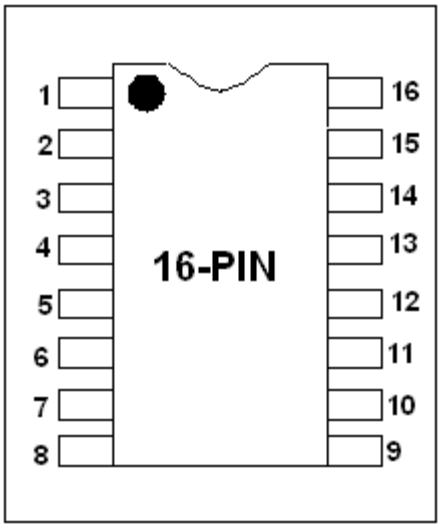
1) First and foremost, try to have a layout of how you want to place all the components on the PC board before you begin soldering. Taking this extra step can eliminate a lot of confusion and unnecessary wiring during assembly.

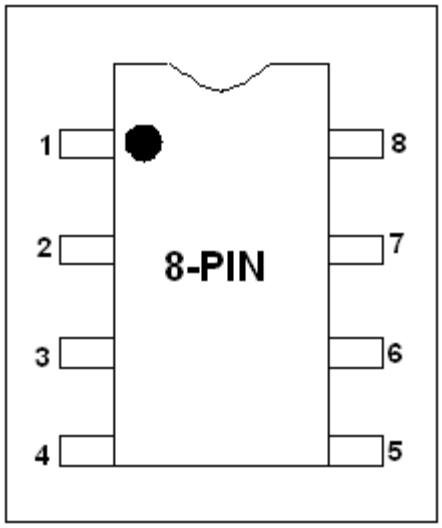
2) If you purchased the PC board from the parts list, use one (or both if you connect them) of the solid strips that runs around the perimeter of the board in order to make all your ground connections. It is much easier to do this than to run an individual ground wire for every ground connection.

3) Make sure to orient the diodes correctly in the circuit. The whole point of a diode is to only let current flow one way (like a check valve), so if you get the two ends switched, the circuit will not function because current will not flow through the diode. There is a **black stripe** on one end of the diode; the end with this stripe is the **negative end**. In other words, current flows towards the end with the stripe. See the diagram below:



4) Also be sure to take note of the proper pin numbering on the IC chips. **The circuit diagram has the connections to the chips laid out by pin number, not by visual position**. The proper IC chip orientation and pin numbering is as follows:





Note that the semicircle at the "top" of the chip indicates the position of Pin 1. If there is no semicircle, then Pin 1 will be indicated by a dot as shown in the diagrams. Regardless of how Pin 1 is indicated, the numbering remains the same.

5) Be sure to orient the electrolytic capacitors in the proper direction, as they are polar (unless otherwise noted). The negative lead is always the shorter lead and will be indicated by a series of minus symbols on the corresponding side of the capacitor. Follow the same principle from note number (3): current flows from positive to negative. Any non-polar capacitors (such as the 0.1 µF capacitors from the parts list above) can have their leads placed at either position.

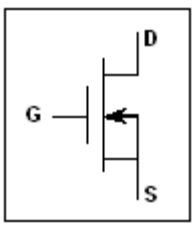
6) The numbers for the serial port terminals are written on the connector itself right next to the pins. The numbers are rather small, so be careful not to confuse them, as the connections to the serial connector are indicated by pin number on the circuit diagram.

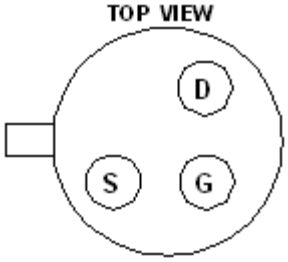
7) The ground connections on the circuit diagram are indicated by a symbol that looks like this: ⏚.

8) Any IC pins marked "N.C." are not connected to anything else in the circuit.

9) For the input wires (shown in the lower right-hand corner of the diagram), the positive wire is connected to the resistor "R5", and the negative wire is connected to ground.

10) The MOSFET transistor is perhaps the most difficult component to install in the circuit. You must keep very careful track of the leads. The diagrams below show which lead is which.





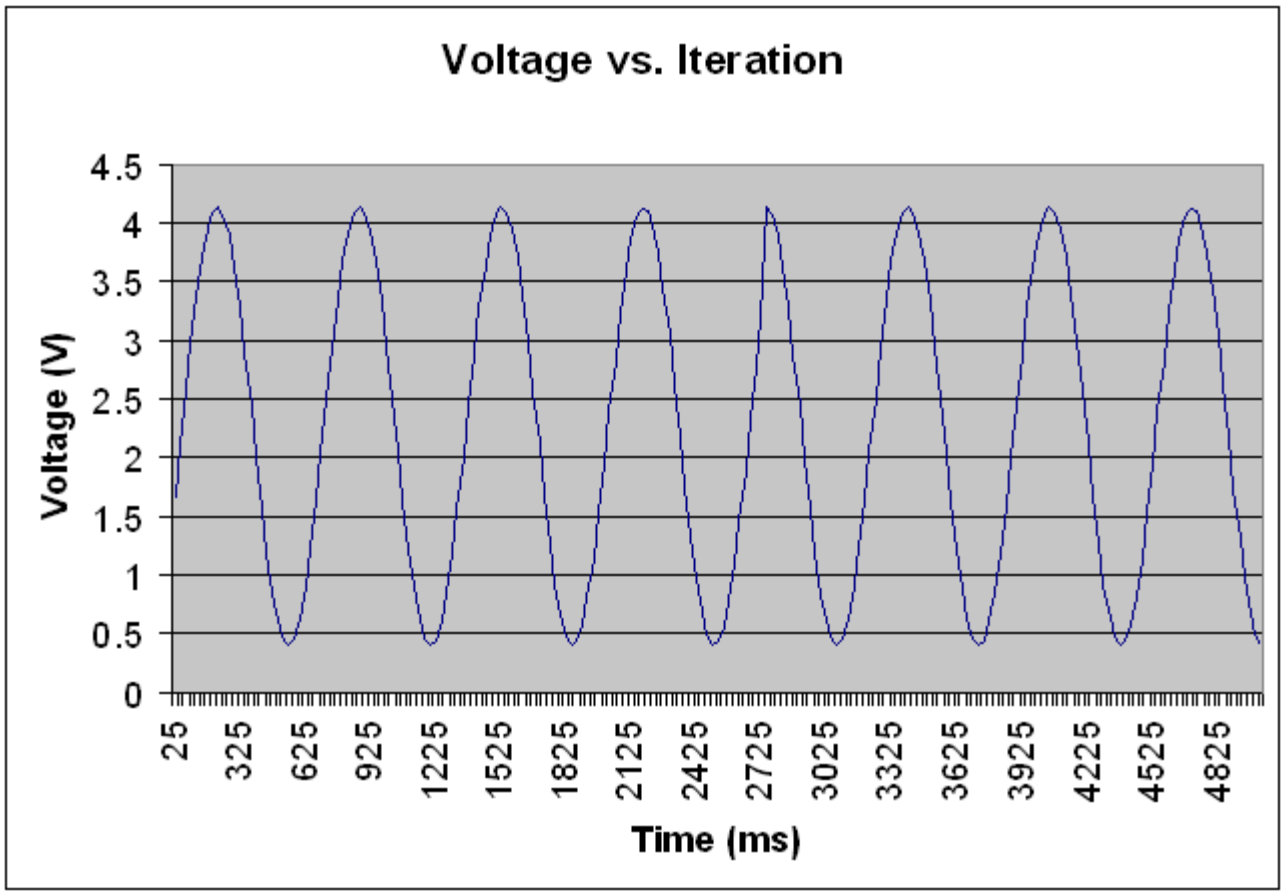
The "D" stands for "drain", "S" corresponds to "source", and "G" is for "gate". Those are three different terminals of a transistor. **Pay special attention to the lead layout diagram on the right above. The leads are displayed in their positions as they are seen from the *TOP* of the transistor.**

Testing the ADC

Once the circuit is built, it can be tested using the program found in Maxim's circuit description. Follow the link entitled "download EJ22 Listing Zip File". You simply connect the circuit to your PC using the serial port, apply an input voltage to the input wires, and the test program will produce a value for the given voltage. A good test for the circuit is to connect the input wires to a function generator running a sine wave, and verify that the output values

reproduce a sine wave when graphed. In order to do this, however, the test source program must be modified. A more detailed description of the test program modifications and other relevant programs for our application of the ADC can be found in the Programming Tutorial. A more detailed explanation of the test procedure follows here:

Shown below is the graph that we obtained from the ADC test values:



Readings are taken by the circuit every 25 ms, as set by the test program. The function generator should be set to a frequency of about 1 Hz for the purpose of sample accuracy, and the peak-to-peak voltage should be no greater than 4.096 V (as shown for the input voltage on the circuit diagram). The wave should be positioned such that the lower peak is at exactly 0 Volts so that the signal produced by the function generator is entirely a positive voltage. Note that it is necessary to have an oscilloscope to assure the accuracy of the wave. The The output values that are saved by the test program will be 12-bit signal values. In order to make the graph of voltage vs. time as shown above, the 12-bit values must be converted into voltage readings. To do this, a simple formula must be used:

$$\frac{(4.5)*(\text{Digital Value})}{4095}$$

The digital values are divided by 4095 because the signal is a 12-bit signal. Since the signal is binary, 12-bits translates to 2^12, or 4096. 1 is subtracted from that value for conversion purposes, making it 4095. The values are also multiplied by 4.5 to obtain an approximate magnitude for the signal readings. The signal we used was slightly over 4 Volts, so we used 4.5 just to be safe. Once the digital values are converted using the above formula, they should produce a graph something like the one above when graphed vs. time.

The modified version of the test program that we used to test the circuit with the function generator can be found here, and the C-source code is also displayed below. Note that the output values will be placed in a text file that is created in the same directory as the "Palm1_1.exe" file. The data file will be named "MYDATA.txt".

C-source code:

```
/*
FILE: palm1_1.c
```

```
DATE: 1.1 - 04/12/02 16:00
VERS: 1.1 : Version to convert decimal digital values to voltages
DESC: Serially interfaced ADC circuit based on MAX187
*/

/* ***************************************************************
// DESIGN IDEA SUBMISSION:
// "12 BIT A/D ON PC SERIAL PORT AND DRAWS MICROPOWER"
// Wettroth 12/5/94, Maxim Integrated Products- Sunnyvale
// ***************************************************************
// *** DESCRIPTION ***
// demonstration and test code for interfacing a MAX187 12 bit A/D
// converter to com2 of a pc. circuit consists of MAX187, MAX220
// and MAX666 low Iq regulator. "bit banging" interface using DTR
// line for sclk and RTS line for chip select (start conversion).
// power is provided by 9 volt battery. current drain is 2 mA in
// run mode and can be powered down by making DTR low- circuit
// powers down after DTR low for more than 100 ms. circuit
// power goes down to MAX666 shutdown current levels of 1 uA.
// power up by writing DTR high- wait 10 mS for Vcc and reference
// stabilization.
// ***************************************************************
// compiled under qc v2.0 (microsoft MSC V6 compatible)
// ***************************************************************
*/

#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <conio.h>
#include <time.h>

/* *** function prototypes */
int main( void ); /* main */
void wake_ad(void); /* wake a/d and wait 10 mS */
int get_ad(void); /* get a/d value- bang bits */
void myDelay(clock_t wait); /* delay by milliseconds */


/* *** global variable declarations */
/* com2 port values (2f8 base add) */

static unsigned int MSR= 0x2FE; /* modem status */
static unsigned int MCR= 0x2FC; /* modem control */
static clock_t delts = 25; /* 25 ms- delay wake */

/* *** define literals for bits */
#define SCLK 1 /* bit 1 of mcr is sclk */
#define CS 2 /* bit 2 of mcr is cs */

/* ******************* begin code ********************** */
int main(void)
{
int ad_value;
long v_value;
int k;

FILE *output_file;

printf("Creating data file on drive c\n");
```

```
output_file = fopen("myData.txt", "wt"); /* ASCII text will save ADC data */

/* Wake up the ADC before acquiring samples */
wake_ad();

printf("Acquiring samples...\n");
for (k=0; k<=200; k++) { /* Perform data acquisition 101 times */
myDelay(delts); /* Delay for 200 msec */
ad_value= get_ad(); /* get_ad serially acquires 12-bit data */

v_value=ad_value*4.5/4095; /*convert decimal digital value to voltage*/

/* ad_value will range from 0 to 2^12 - 1 = 4095 */
/* v_value will range from 0V to 4.5V */
fprintf(output_file, "%d\n", v_value);
/* myData.txt will have numbers ranging from 0 to 4095 */

}
fclose(output_file);
} /* end of main */


/* *** functions ********************** */

/* *** wake up a/d- turn on 666 regulator */
void wake_ad()
{
outportb(MCR,SCLK);
return;
}

/* get a/d value */
int get_ad()
{
int ad_result,i,j;
void myDelay( clock_t wait );
static clock_t d1ms=0x1;
outportb(MCR,SCLK+CS);

while ((inportb(MSR) & 32) !=0 ); /* wait convert complete */

/* clock in 12 bits to ad_result */
ad_result=0;
for(i=1;i<=12 ; i++) {
myDelay(d1ms);
outportb(MCR,CS); /* clock high */
myDelay(d1ms);
outportb(MCR,CS+SCLK); /* clock low */
myDelay(d1ms);

/* get a bit and put in place */
if ((inportb(MSR) & 32) ==0)
ad_result += 1;
/* shift left one */
ad_result = ad_result << 1;
}
outportb(MCR,SCLK); /* deselect */
ad_result=ad_result >> 1; /* fixup */

return(ad_result);
```

```
}

/* *** delay routine */
void myDelay( clock_t wait )
{
clock_t t1, t2,i;

if( !wait ) return;

/* special logic for short waits- granularity normally 50 ms) */
/* empirical..*/

if(wait < 100) {
do {
for(i=0 ; i<=50 ; i++) {
/* do nothing */
}
wait -= 1;
} while (wait >=0);
return;

}

t1 = wait + clock();
do {
t2 = clock();
} while( t2 < t1 );
}
/* the end */
```

Once you have saved the program (including all the files in the ZIP file) to a directory on your computer and have connected the circuit, you should be able to run the "Palm1_1.exe" file and begin to take values. As previously noted, the values will be placed in a text file called "MYDATA.txt" in the same directory as the program.

The circuit should be connected such that the positive and negative wires from the function generator are spliced to the positive and negative input wires on the circuit, respectively. The serial port on the circuit should be placed into the serial port on your PC, and a 9V battery must be connected to the circuit as well.

Important note about test program: The program is set up to look for the circuit on the COM2 serial port. If you do not have two serial ports, or your PC's serial port is assigned as COM1, you may have to change the resource settings in your device manager to assign the port as COM2.

Questions or comments? E-mail us at palmtek18@yahoo.com.

Return to Homepage