M.E. 530.341 Lab 10:
Analog to Digital and Digital to Analog Conversion

Noah J. Cowan*
Department of Mechanical Engineering
The Johns Hopkins University

Reading: Lecture notes
Spec sheet for ADC0831 8-Bit Serial Analog to Digital Converter IC.
Spec Sheet for TMP36GT9 analog temperature sensor IC.
BSII documentation for the SHIFTIN command (in BSII manual).
BSII Application Note 2: Using SHIFTIN and SHIFTOUT (in BSII manual).

Apparatus: ADC0831, TMP37GT9, Basic Stamp, Stamp Motherboard (Board of Education), AC Adapter, Basic Stamp development editor on a PC, RS232 cable, Oscilloscope,

1 Digital to Analog Conversion

Figure 1 shows the results of the circuit analysis we performed in class for the DAC circuit shown in Figure 1.

1) Choose two rows from Figure 1. Independently analyze the DAC circuit for these two cases to verify (or disprove) the results shown in the Figure 2.

2) Wire this simple D/A circuit, Figure 1, to I/O pins P12 through P15 on your breadboard. Use 1% resistors provided by your TA. We stock 1% resistors in 1KΩ, 2KΩ, and 4KΩ. Use two 4KΩ resistors to construct the 8KΩ resistor.

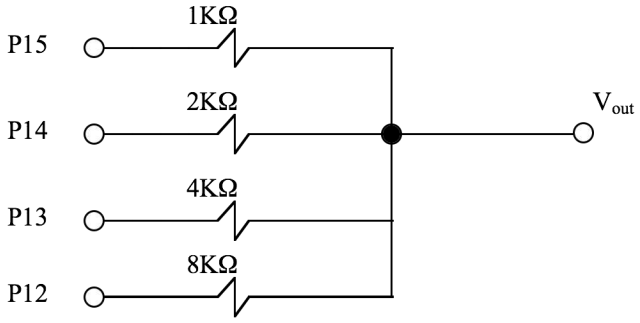


Figure 1: A Simple D/A Converter Circuit

3) Write a Stamp2 basic program to generate an ascending "ramp" of $\mathbf{V_{out}}$ voltage values – the output should begin at 0V and incrementally increase to 5V, then return to 0V.

* This laboratory © Louis L. Whitcomb. Updated with permission by Noah J. Cowan.

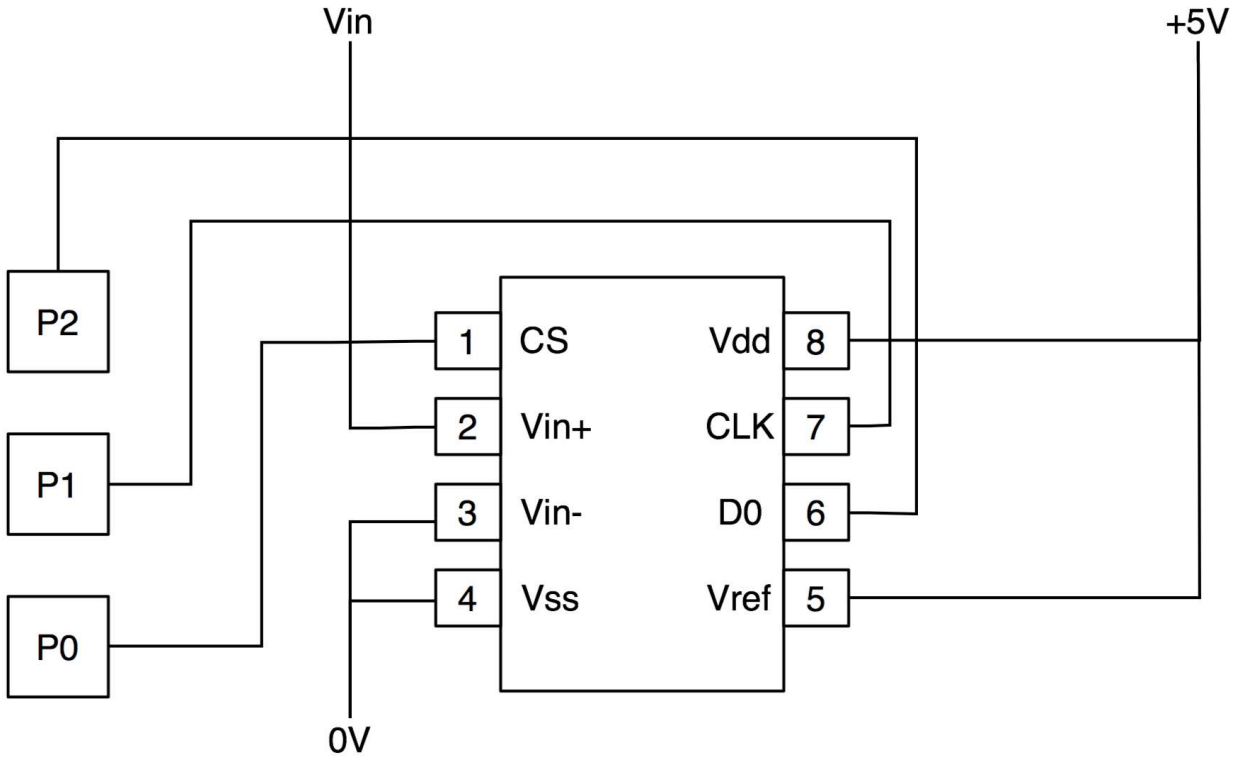
4) Use your scope to measure and print the waveform $\mathbf{V_{out}}$. Title and annotate your plot.

P15	P14	P13	P12	R_L	R_H	R_L+R_H	$R_L/(R_L+R_H)$	V_{out}
0	0	0	0	533	∞	∞	0.000	0.00
0	0	0	1	571	8000	8571	0.067	0.33
0	0	1	0	615	4000	4615	0.133	0.67
0	0	1	1	666	2666	3333	0.200	1.00
0	1	0	0	727	2000	2727	0.267	1.33
0	1	0	1	800	1600	2400	0.333	1.67
0	1	1	0	888	1333	2222	0.400	2.00
0	1	1	1	1000	1142	2142	0.467	2.33
1	0	0	0	1142	1000	2142	0.533	2.67
1	0	0	1	1333	888	2222	0.600	3.00
1	0	1	0	1600	800	2400	0.667	3.33
1	0	1	1	2000	727	2727	0.733	3.67
1	1	0	0	2666	666	3333	0.800	4.00
1	1	0	1	4000	615	4615	0.867	4.33
1	1	1	0	8000	571	8571	0.933	4.67
1	1	1	1	∞	533	∞	1.000	5.00

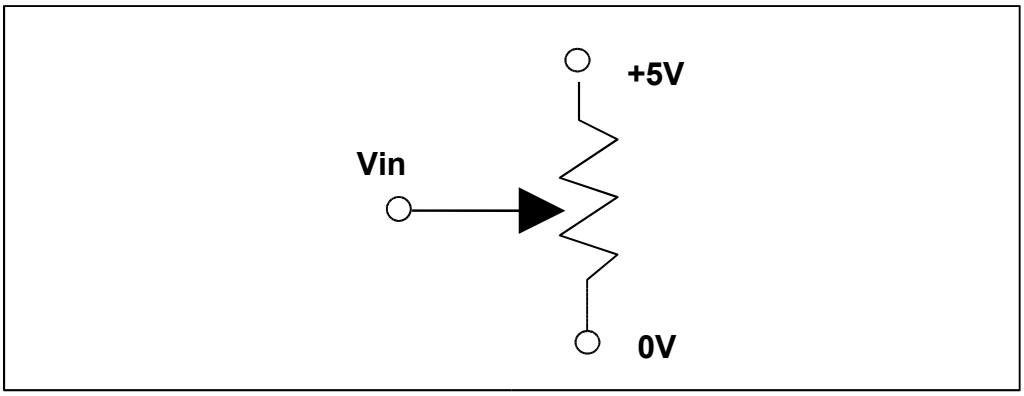
Figure 2: DAC Output Table showing D/A output voltage as a function of P11-P15 logical (TTL) output.

2 Analog to Digital Conversion

5) Read the specification sheet for the ADC0831 Analog to Digital converter integrated circuit. Construct the following circuit:



6) Hook a potentiometer to the V_{in} signal to provide an adjustable voltage as shown below.



7) Write a program to use the ADC0831 to perform an analog-to-digital conversion of the voltage $V_{in,}$ and to transfer the 8-bit digital value to the basic stamp. Your program should:
   a) Initialize by setting P2 (D0) to be an input,
   b) Initialize P0 (CS) and P1 (CLK) to be outputs, with default values of HIGH and LOW.
   c) To perform a conversion your program should:
      i) Bring CS low to initiate A/D conversion
      ii) Read the A/D serial data with the SERIN command
      iii) Bring CS high to complete the conversion
      iv) Print put the A/D value

A sample program to read the A/D is as follows:

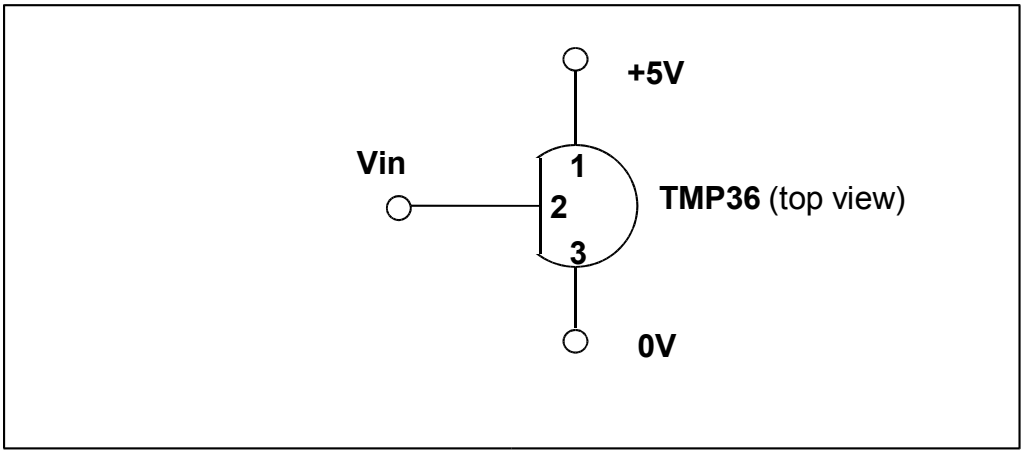
```
'{STAMP BS2}
' A/D Conversion with the ADC0831 Nov 3, 2003 LLW

' define variables
ad_byte var byte

' define constant labels for the 3 I/O pins
CS  con 0
CLK con 1
D0  con 2

' configure CS line as output, quiescent state is high
OUTPUT CS
HIGH CS

' configure CLK line as output, quiescent state is low
OUTPUT CLK
LOW CLK

' top of sampling loop
TOP:

' a falling edge on CS initiates the A/D conversion
' and activates the D0 pin of the AD0831
low CS

' shift in the leading dummy bit and the 8 bits of A/D data
' data  input on pin D0
' clock output on pin CLK
' A/D data is MSB first, loaded into variable AD_BYTE
shiftin D0, CLK, MSBPOST, [ad_byte\9]

' complete the conversion cycle and deactivate the D0 output
high cs

' print out the value of the A/D counts
debug "A/D count=", DEC ad_byte, CR

goto top
```

8) Hook a voltmeter to Vin. Hoop up your scope to display CLK on CH1 and D0 on CH2. Experiment with various settings of your potentiometer.
   a) What is the A/D count value for Vin = 0V?
   b) Repeat the above for Vin = 2.5V or so.
   c) Repeat the above for Vin = 5.0V.
9) Modify your program to display the A/D value in VOLTS or MILLIVOLTS.

3 Using the A/D with an Analog Temperature Sensor

This section is optional, it is not required.

10) Remove the potentiometer from your circuit and replace it with the TMP36 analog temperature sensor:



11) Read the spec sheet. What is the mathematical relationship between the temperature (in Celsius) and the output voltage (in volts)?

12) Write a program to read the output voltage of the TMP36 with the A/D converter, compute the temperature of in degrees Celsius, and print the temperature once a second. A sample program is given on the next page.

13) What happens when you warm up the temperature sensor by holding it in your fingers? When you let go?

14) Print a screen shot of the debug output while you are warming up the temperature sensor by holding it in your fingers.

- Hand in a floppy disk containing your programs with your lab. Floppy disks are available in the lab.
- Remember to note your lab partner's name on your lab report.
- Remember to show your work.
- Typed or hand-written lab reports are OK. Messy or ambiguous lab reports will be rejected.
- Please clean up your workstation to perfection when you are done.

```
'{$STAMP BS2}
' A/D Conversion with the ADC0831 April 21, 2004 LLW
' define variables
ad_byte    VAR Byte
centivolts VAR Word
temp_C     VAR Word

' define constants as labels for the 3 I/O pins
CS  CON 0
CLK CON 1
D0  CON 2

' configure CS line as output, quiescent state is high
OUTPUT CS
HIGH   CS

' configure CLK line as output, quiescent state is low
OUTPUT CLK
LOW    CLK

' top of sampling loop

TOP:
' a falling edge on CS initiates the A/D conversion
' and activates the D0 pin of the AD0831

LOW CS
' shift in the leading dummy bit and the 8 bits of A/D data
' data input on pin D0
' clock output on pin CLK
' A/D data is MSB first, loaded into variable AD_BYTE

SHIFTIN D0, CLK, MSBPOST, [ad_byte\9]
' complete the conversion cycle and deactivate the D0 output
HIGH cs

' convert the A/D count to centivolts
centivolts = ad_byte * 500 /256

' compute temperature
' temp C = (volts - 0.5V) * 1 degree C / 1 centivolt
temp_C  = (centivolts - 50)

' print out the value of the A/D counts

DEBUG "A/D Counts= ",   DEC ad_byte
DEBUG "  centivolts= ", DEC centivolts
DEBUG "  Temp= ",       DEC temp_C, " Centigrade"
DEBUG CR

PAUSE 1000  ' pause 1000 ms

GOTO top
```