EE 389 Electronic Design Lab-II Project Report, Department of Electrical Engineering., IIT Bombay, Nov'08

Curve Tracer

Group No. D4

Devesh Meena(05D07042)
Hitesh Shakya (05D07005)
Nishant Shreshth (05D07040)

Guides:
Prof. A.Karandikar
Prof. Dipankar
Prof. D.K. Sharma

1. Abstract:

The basic operating principle of the device is to apply a swept (automatically varying) voltage to the main terminals of the DUT while measuring the amount of current that the device permits to flow. This so-called V-I (voltage versus current) graph is displayed on GUI developed in MATLAB. Three-terminal devices require an additional connection; this is usually supplied from a stepped voltage or current source attached to the control terminal of the DUT. By sweeping through the full range of main terminal voltages with each step of the control signal, a family of V-I curves can be generated. Curve tracers usually contain convenient connection arrangements for two- or three-terminal DUTs, often in the form of sockets arranged to allow the plugging-in of the various common packages used for transistors and diodes.

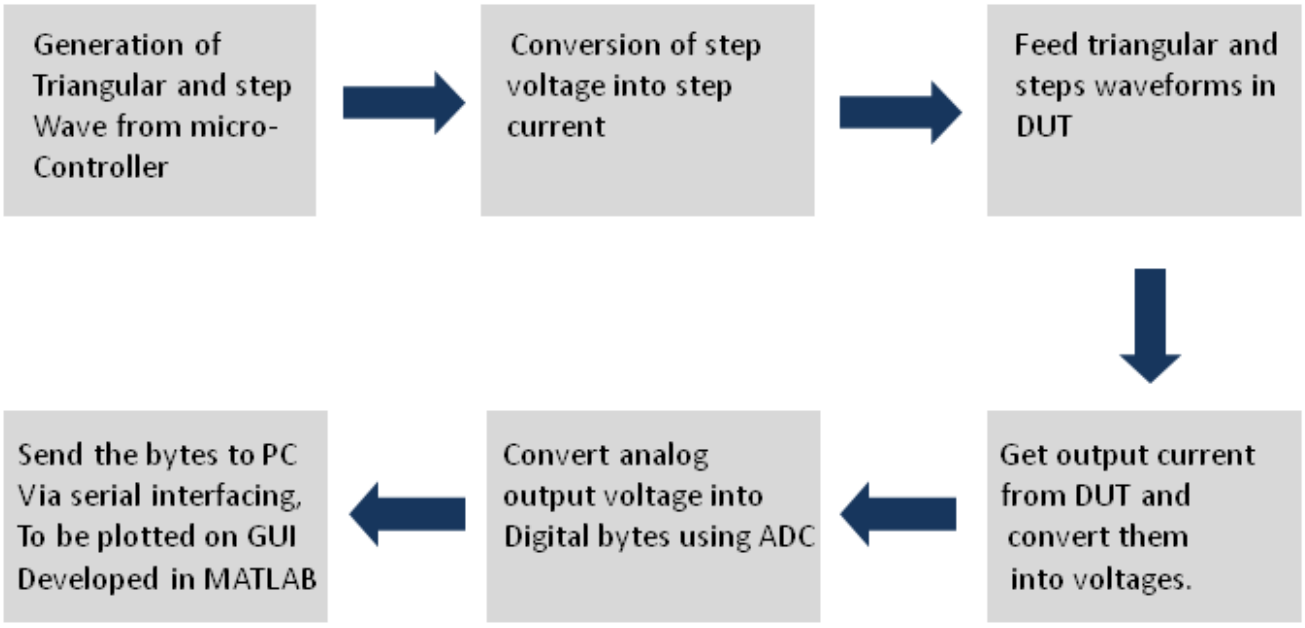
2. Introduction:

The main part of the project deals with :-

Programming microcontroller (AT89c52) for generating triangular and step waveforms which are to be fed in circuitry for DUT.
PC to microcontroller interfacing using inbuilt UART and via serial port.
Developing a software in MATLAB for plotting the characteristics on PC.

3. Design Approach:

The block diagram of our design is as shown in figure 1.



Components used:

AT89c52 Microcontrollller
ADC0808, DAC0800
Opamp LM741
MAX232 Serial Driver
RS232(DB9) Serial Cable
Bipolar Junction Transistors BC107, BC548, BC557

4. Module Description:

4.1 Generation of triangular and step input using microcontroller:

For triangular wave, 8 bit data from 00H to FFH and then back to 00H is sent to a DAC with a certain delay between two consecutive bytes. This bit pattern, after conversion into analog, essentially gives triangular waveform at the output of DAC.

Similarly step input is achieved by producing bit patterns for #00, #16, #32,...#255 with during for each byte is equal to the period of triangular waveform generated earlier.

Circuit diagram for DAC connections are as shown in fig2.

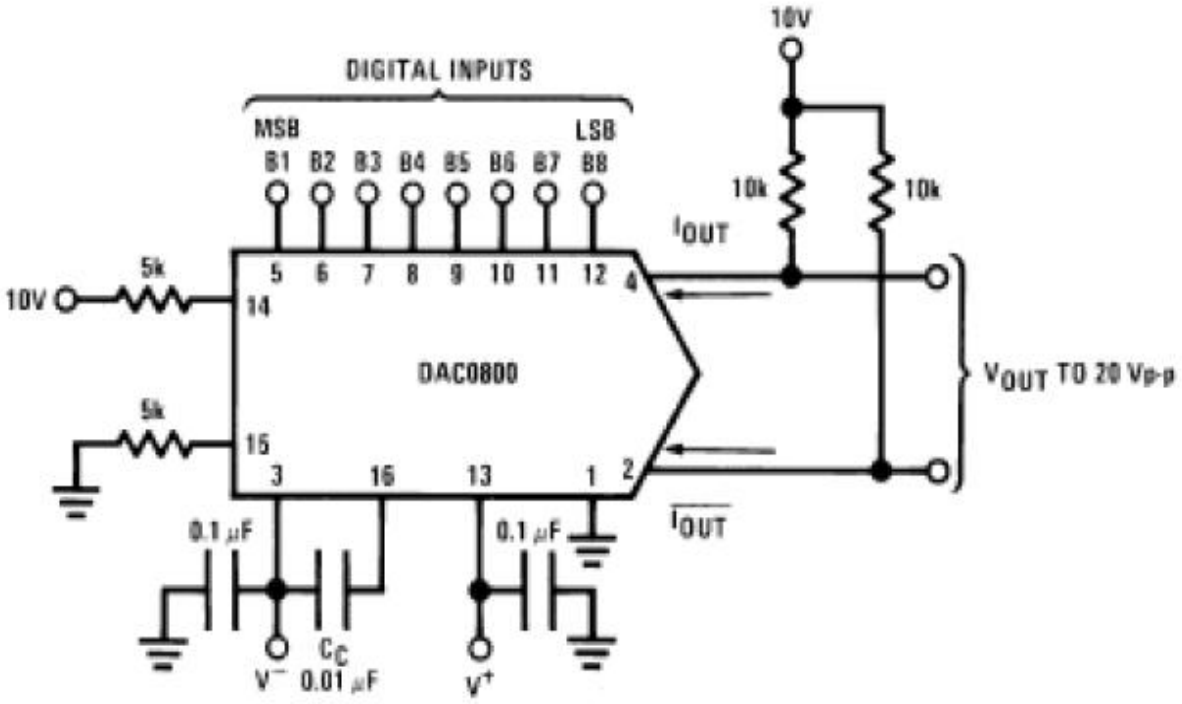


Figure2. Digital to Analog conversion

Now although the digital bytes have been converted into analog waves but DAC gives output as currents not as voltages thus we are now required to convert these output analog current into analog voltages which we did using an Op-amp in inverting configuration

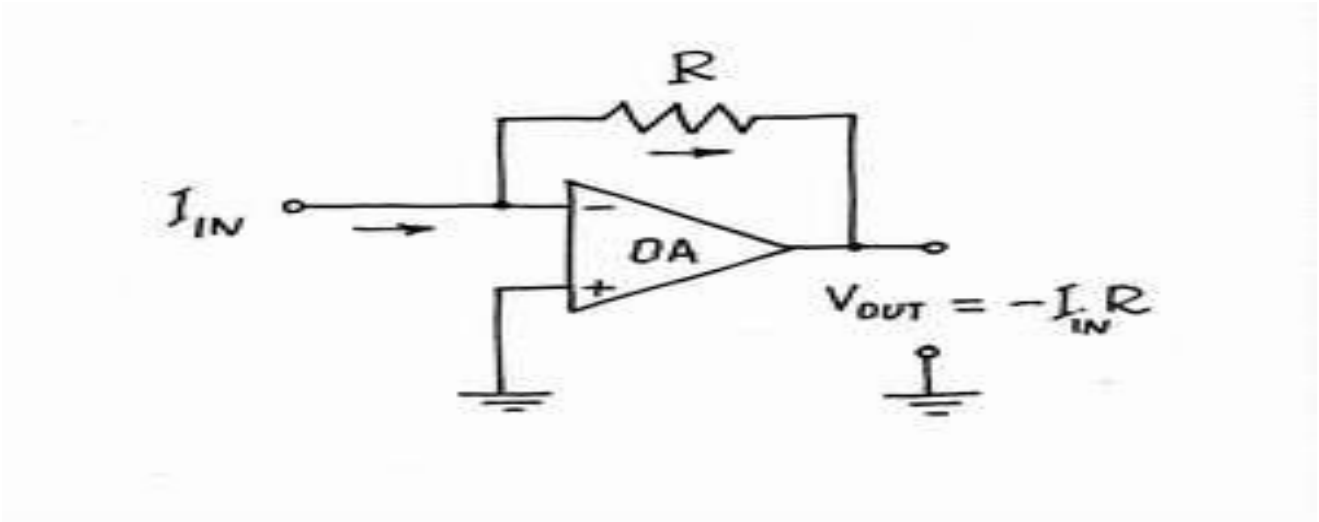


Fig.3 Op-amp current-to-voltage converter

Resulting waveforms are as shown in figure 4.

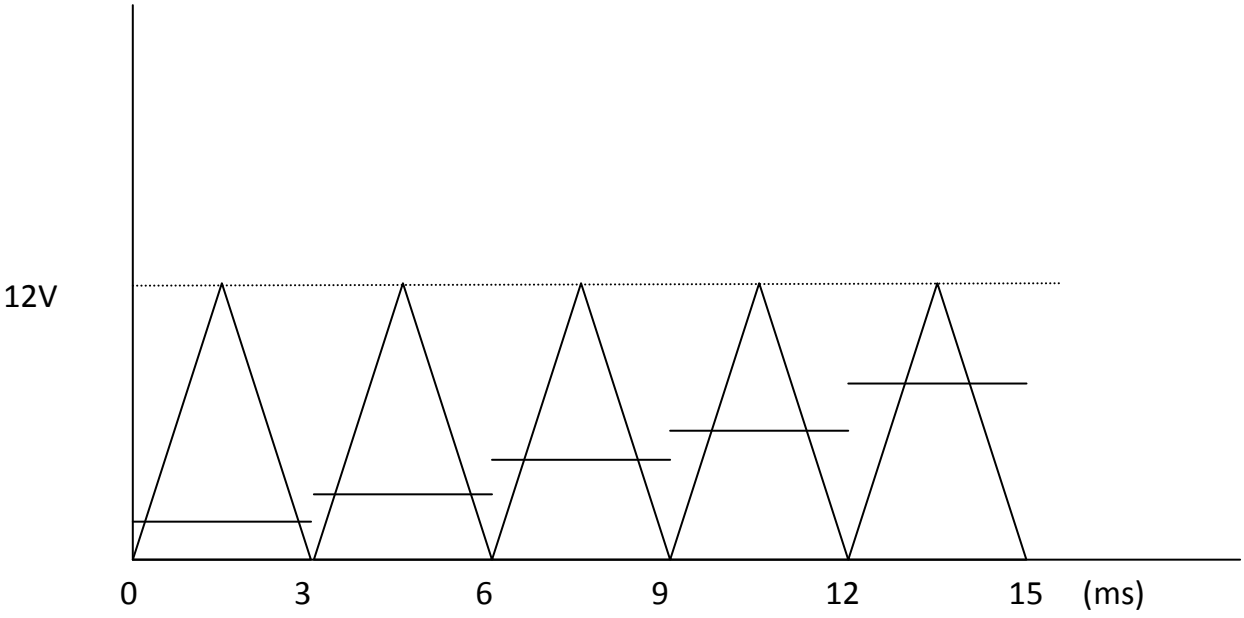
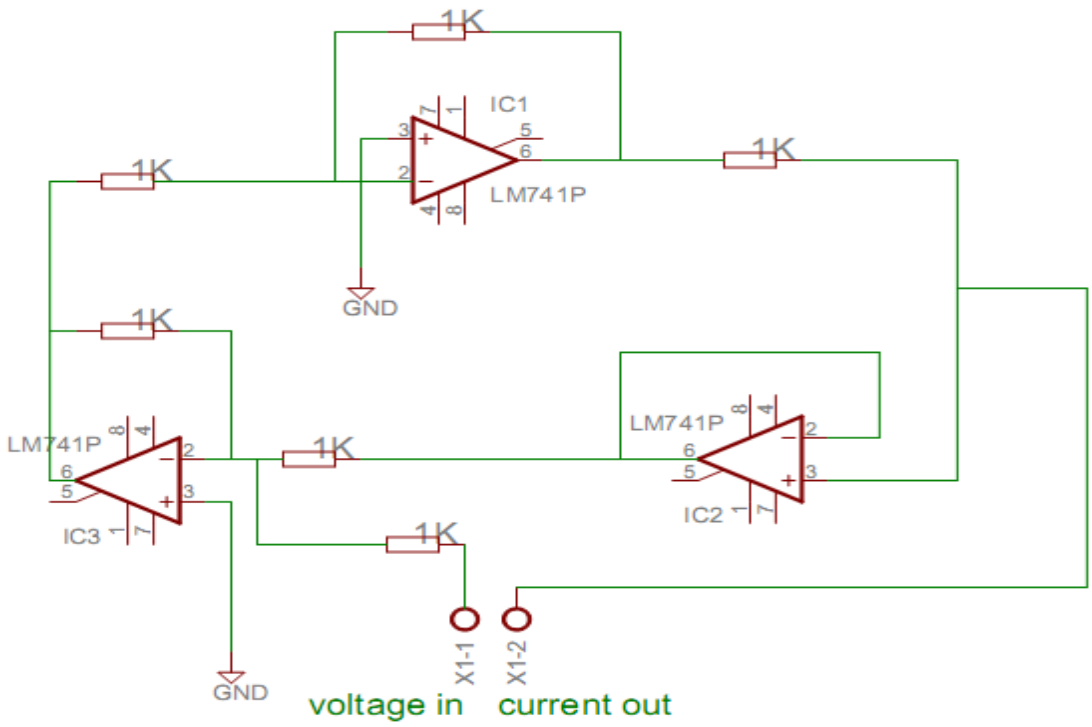


Fig.4 Triangular and step waveforms generated using MuC and DAC

For plotting I-V characteristics for a BJT transistors we'll be feeding triangular voltage to its collector while step currents (not voltages!) to the base. Thus we need to convert step voltage waveform into current wave form; a proposed block diagram for this is shown in fig.



Thus the resulting step current is fed into the Base and triangular voltage wave is fed into the Collector of the BJT.

4.2 Measurement of output currents from DUT:

The output current from the DUT are analog values, so first they are converted into voltages using the configuration shown in fig3 and then converted into digital using ADC0808. A typical circuit used here is as shown in fig5.

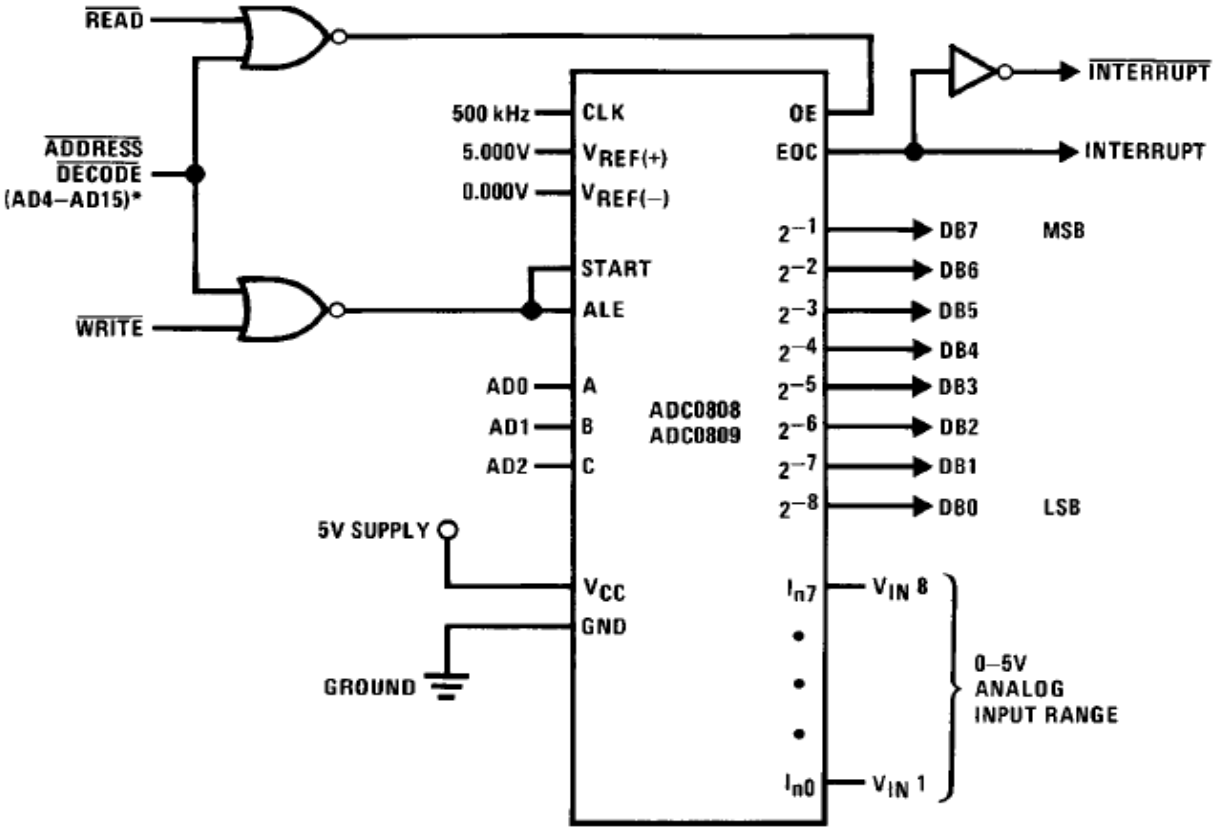


Fig5. A typical circuit for A-D conversion

4.3: Reading output current from DUT on PC

The analog input is continuously fed into one of the analog input channels (IN0 in our case) of ADC0808. We feed the ADC clock from the ALE signal of the controller. This provides a sampling rate of 11.0592/6 MHz = 1843 KHz, which is sufficient for the required serial transfer (considering the upper bound on the rate of the serial transfer).

Analog to digital conversion starts only when the controller receives an instruction from PC. Having received this signal, the controller sends control signals (ALE, START) to ADC to start the conversion. Once a byte is received from the ADC, it is immediately uploaded in the SBUF register of the controller. The controller waits until the serial transfer is done and then it sends control signals to ADC for the next byte.

The serial transfer is done at a baud rate of 57600 which is the highest possible with our crystal frequency using Timer 1 in auto reload mode (Mode 2, SMOD = 1). The other serial port specifications are:
1. Data Bits = 8
2. Start Bit = 1
3. Stop Bit = 1
4. Parity = None

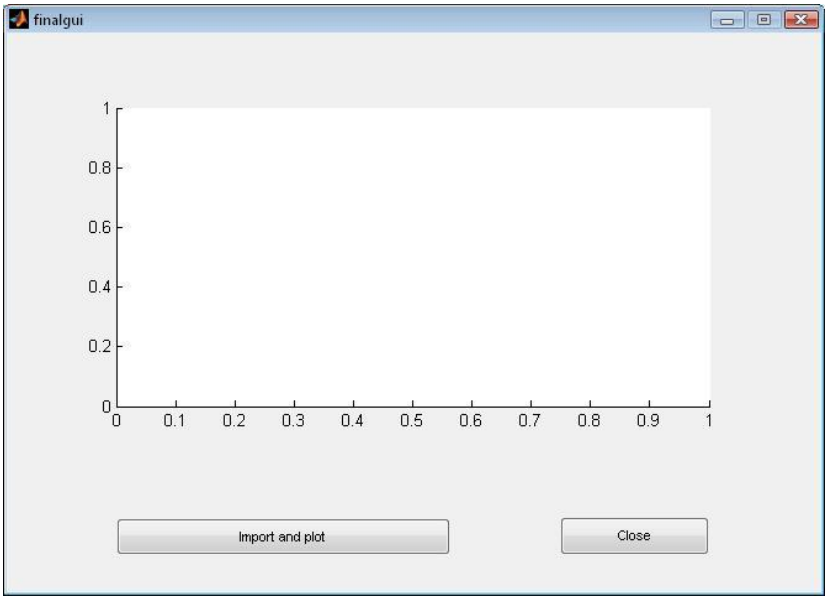
A total of 100 bytes are transferred for every instruction received from user. Increasing the number of bytes only slows the overall process with only slight improvement in the appearance of final characteristic in the form of resolution.

Received bytes are stored in the input buffer of MATLAB serial port and are read only when all the 100 bytes have been received.

The transfer of one byte is timed as follows (not in actual time order):
1. Serial: 1/5760sec = 0.1736ms
2. ADC: 0.110ms approx.
3. Controller: 20micro seconds approx.

Hence, the total delay accounted in transferring a single byte is approximately 0.28ms.

4.4: MATLAB Graphical User Interface:



The ‘Import and plot’ pushbutton when pressed, sends an instruction to the microcontroller to start the analog to digital conversion. The received points are plotted automatically after all the bytes are picked up. The ‘Close’ pushbutton closes the interface. We did not use any stop function here as there is no need to read the data continuously.