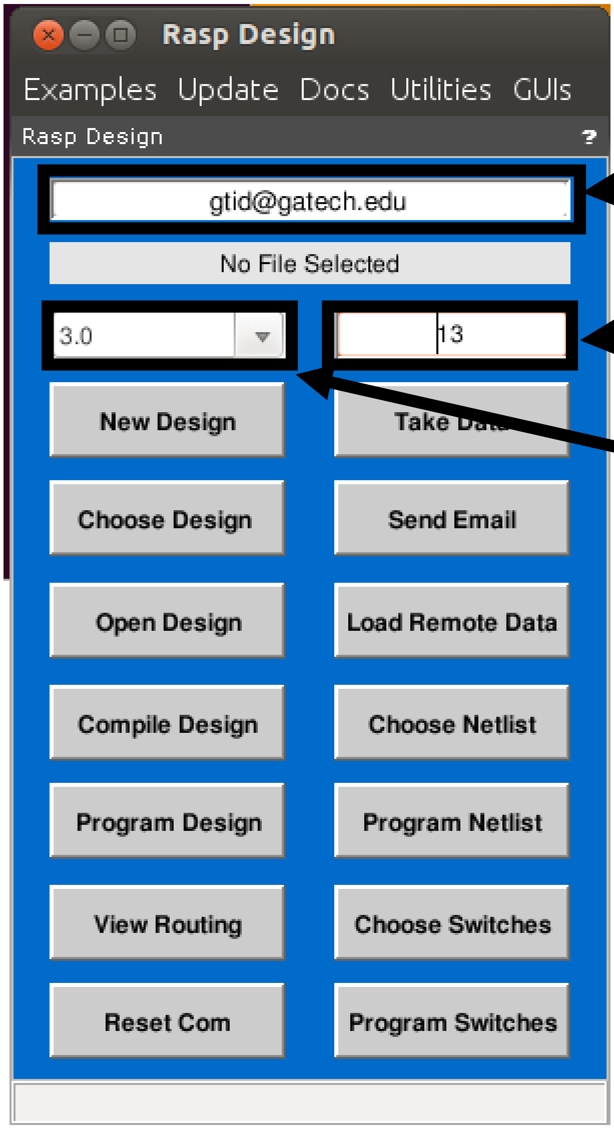
Tutorial on Remote System

- Refer to the VM_FPAA.pdf before this tutorial.

The chip number for remote system is 13 and it is a rasp3.0 board. So the blue GUI should look like this:



gtid@gatech.edu

No File Selected

3.0

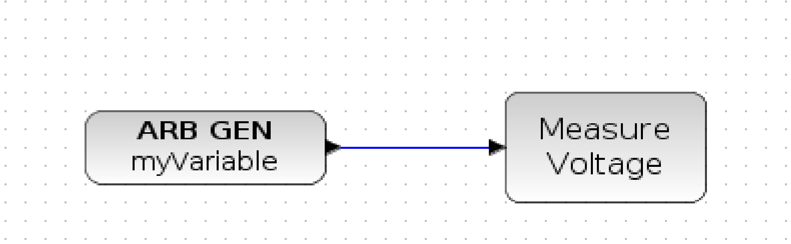
13

Email id where the remote system will send your results.

Chip number. Remote system is 13.

Which board. Remote system is a 3.0 board.

1. Measuring DAC output



Tutorial on Remote System

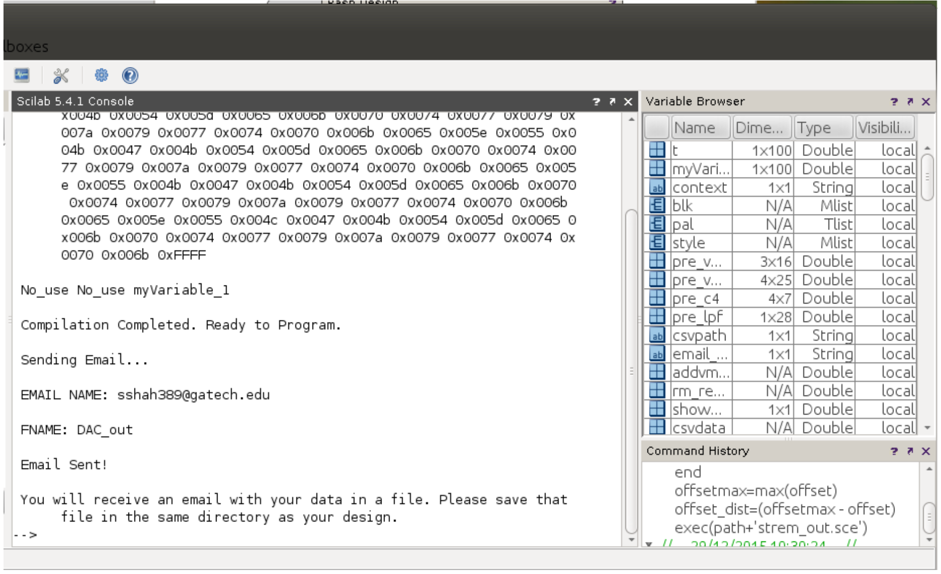
Create a new design using the “New Design” button on the GUI. For the first step we are going to test the output of a DAC. So create an xcos file using an arb gen block and to measure the output of the DAC we will use an ADC here it is called measure voltage block.

Measure voltage block:
It is a 14 bit ADC. The maximum sampling speed of this ADC is 200 Hz. So make sure your input is much smaller than that. Here we will use the input frequency of the DAC to be 5 Hz.

Now define myVariable = 1.2 + 1*(sin(2*3.14*(0.01:0.01:1)*5));

After this save the xcos design and choose the design from the GUI. After selecting the design compile it using the “Compile Design” button. If there are no errors in the design it should say “Compilation Completed. Ready to Program”.

After the Compilation step click the “Send Email” button to program the remote system. The scilab console should like below if there are no errors. It might take a while depending on the design and number of people using the system.



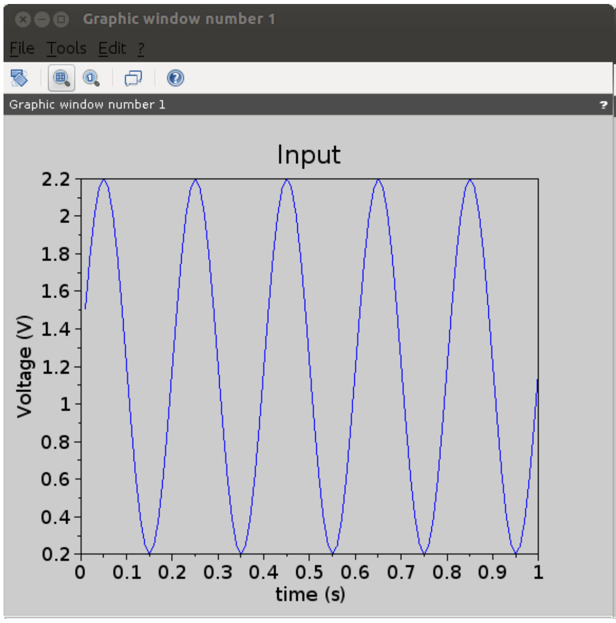
```
      x004b 0x0054 0x005d 0x0065 0x006b 0x0070 0x0074 0x0077 0x0079 0x
      007a 0x0079 0x0077 0x0074 0x0070 0x006b 0x0065 0x005e 0x0055 0x0
      04b 0x0047 0x004b 0x0054 0x005d 0x0065 0x006b 0x0070 0x0074 0x00
      77 0x0079 0x007a 0x0079 0x0077 0x0074 0x0070 0x006b 0x0065 0x005
      e 0x0055 0x004b 0x0047 0x004b 0x0054 0x005d 0x0065 0x006b 0x0070
       0x0074 0x0077 0x0079 0x007a 0x0079 0x0077 0x0074 0x0070 0x006b
      0x0065 0x005e 0x0055 0x004c 0x0047 0x004b 0x0054 0x005d 0x0065 0
      x006b 0x0070 0x0074 0x0077 0x0079 0x007a 0x0079 0x0077 0x0074 0x
      0070 0x006b 0xFFFF

 No_use No_use myVariable_1

 Compilation Completed. Ready to Program.

 Sending Email...

 EMAIL NAME: sshah389@gatech.edu

 FNAME: DAC_out

 Email Sent!

 You will receive an email with your data in a file. Please save that
      file in the same directory as your design.
-->
```

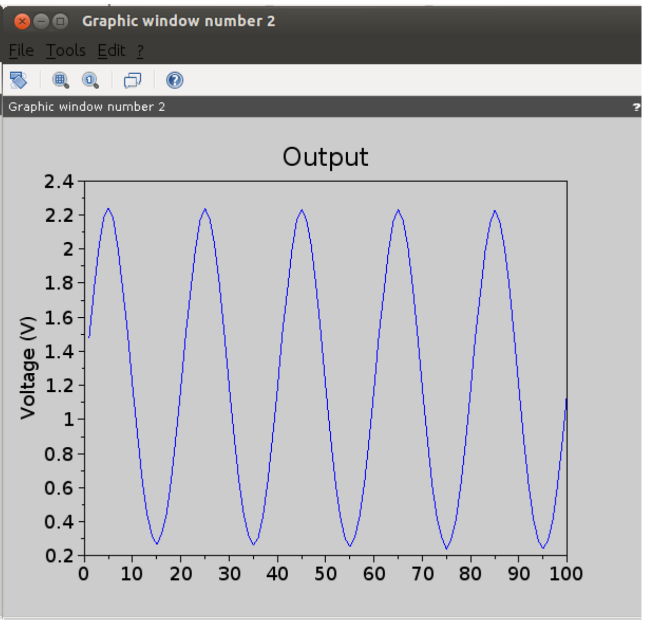
	Name	Dime...	Type	Visibili...
⊞	t	1x100	Double	local
⊞	myVari...	1x100	Double	local
ab	context	1x1	String	local
E	blk	N/A	Mlist	local
E	pal	N/A	Tlist	local
E	style	N/A	Mlist	local
⊞	pre_v...	3x16	Double	local
⊞	pre_v...	4x25	Double	local
⊞	pre_c4	4x7	Double	local
⊞	pre_lpf	1x28	Double	local
ab	csvpath	1x1	String	local
ab	email_...	1x1	String	local
⊞	addvm...	N/A	Double	local
⊞	rm_re...	N/A	Double	local
⊞	show...	1x1	Double	local
⊞	csvdata	N/A	Double	local

```
    end
    offsetmax=max(offset)
    offset_dist=(offsetmax - offset)
    exec(path+'strem_out.sce')
// -- 29/12/2015 10:30:24 -- //
```

You will receive an email from fpaabot.dev@gmail.com with a results.zip file containing the output. Download the file into the same folder as your xcos design and then click “Load Remote Data” on the GUI which should show:
“Loading Data...

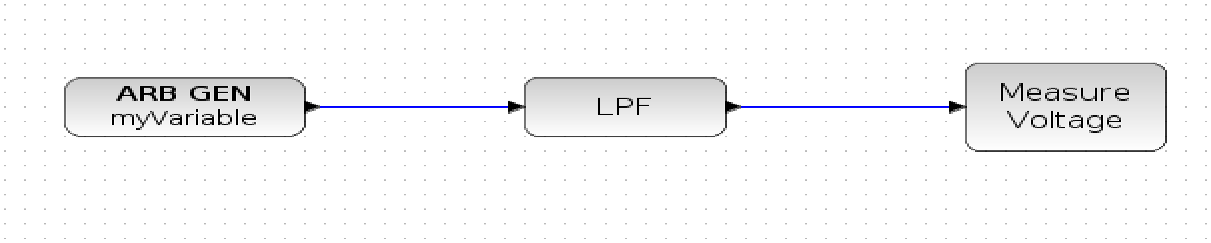
Your data is loaded and saved in the variable rm_results. Your results are saved as “rm_results” in Variable browser. As expected the output should be similar to the input. Below are the two plots.





2. Measuring Low Pass filter in FPAA

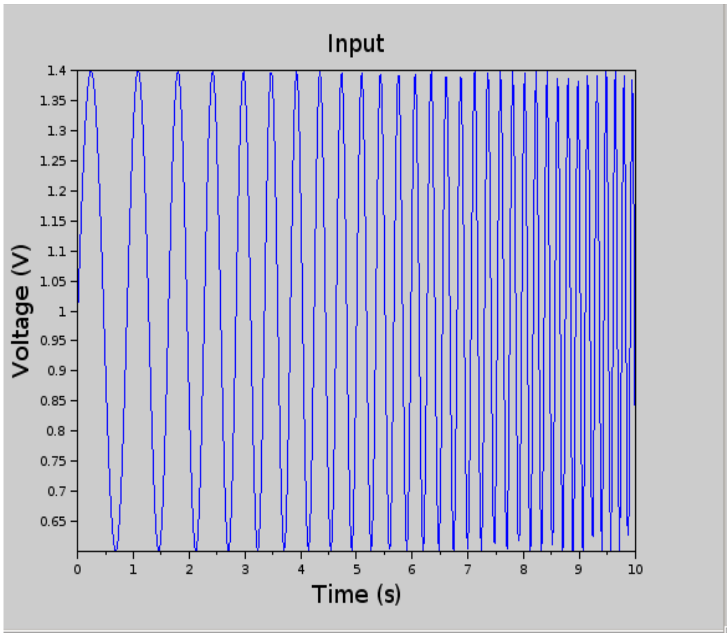
Since the measure ADC block has a sample rate of 200 Hz we will select a corner frequency of the LPF to be smaller than 100 Hz.
To be able to see the cut-off or attenuation in the transient signal we will use a chirp signal as an input signal.

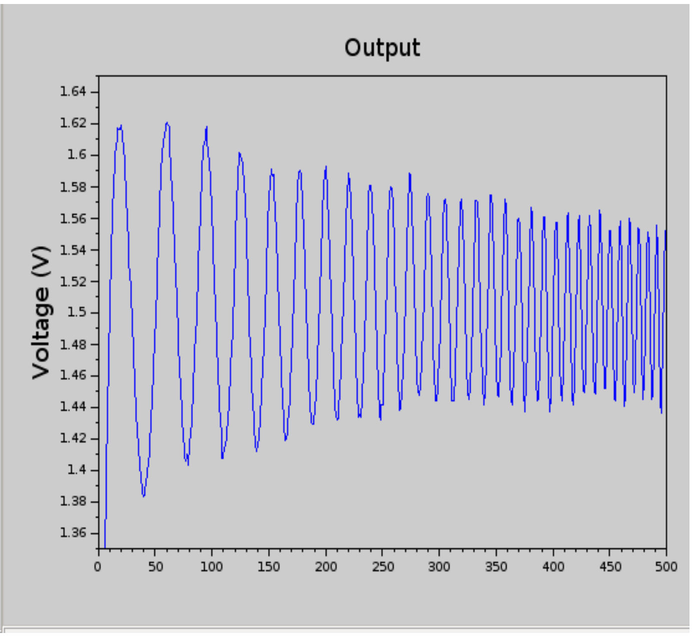


The low pass filter is just an OTA in a source follower configuration. The cut-off frequency of this first order filter can be changed by changing the bias current of the OTA. Here to look at the cut-off the biasing is kept at 30e-12 A of current. The chirp signal used as an input could be generated using the following code:

```
t=0:0.02:10;
f0=logspace(0.01,0.5,501)
myVariable=1+ 0.4*(sin(2*3.14*f0.*t));
figure();
plot(t,myVariable)
```

The output signal gets attenuated as the frequency increases. The voltage difference is probably due to the offset in the amplifier.

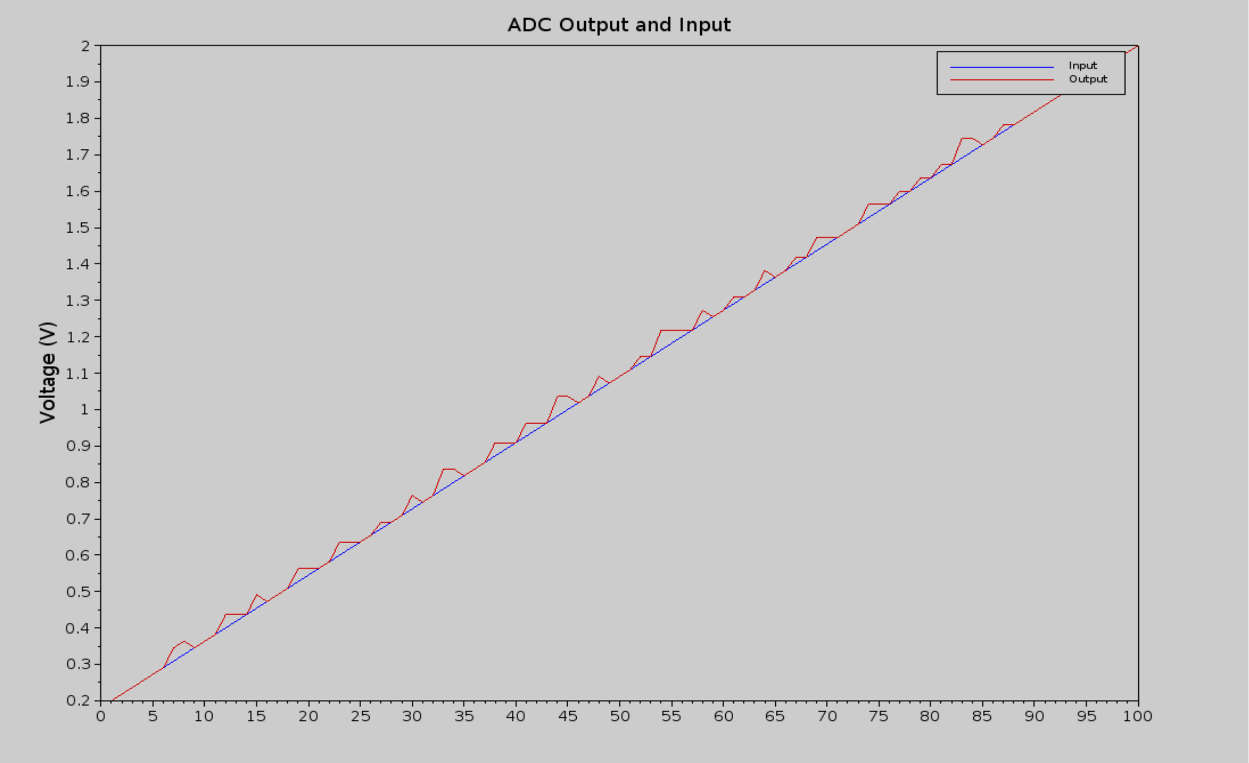

Input
Voltage (V)
1.4
1.35
1.3
1.25
1.2
1.15
1.1
1.05
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0
1
2
3
4
5
6
7
8
9
10
Time (s)


Output
Voltage (V)
1.64
1.62
1.6
1.58
1.56
1.54
1.52
1.5
1.48
1.46
1.44
1.42
1.4
1.38
1.36
0
50
100
150
200
250
300
350
400
450
500

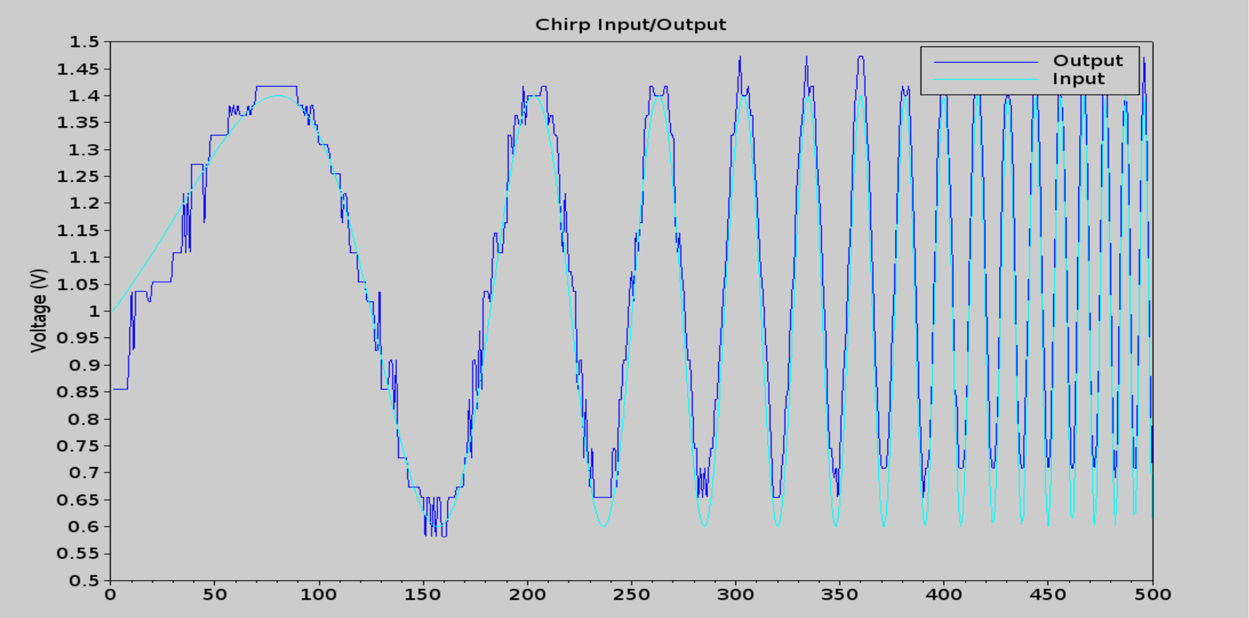
Tutorial on Remote System

3. C4 using a ramp ADC

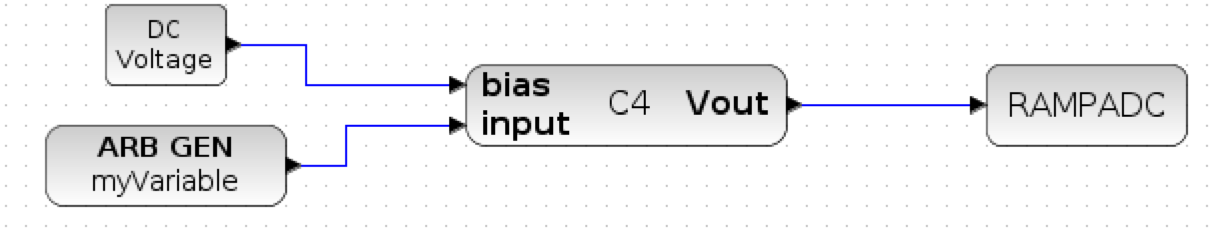
To sample faster than 200 Hz an ADC with faster sampling rates needs to be used. Here we will use a 8 bit ramp ADC. If you give an input as we did in the first step and just look at the output it should look like this:



Also the above graph gives a range of the ADC which is from 0.2 volts to 2 volts. Sometimes due to variation in programming the outputs might be off a little bit.

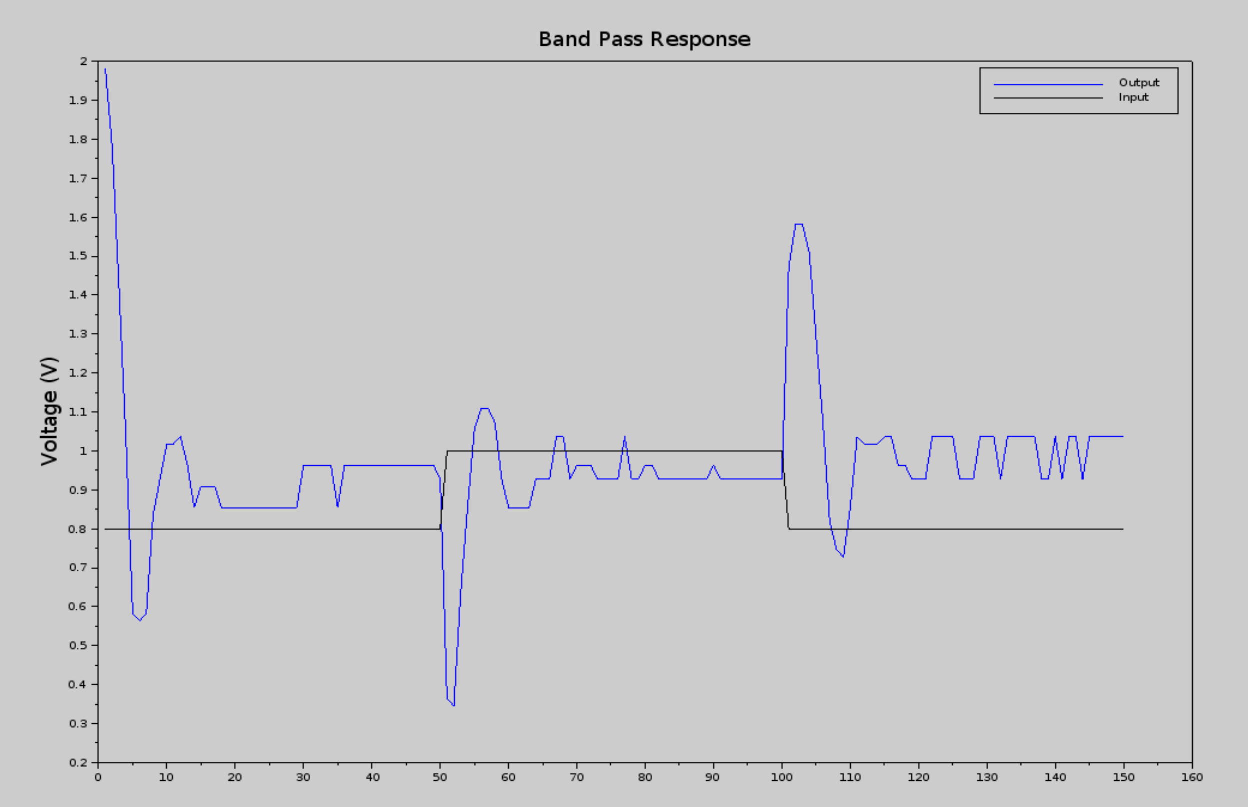


The chirp signal would look like the above figure when measured through the ADC (which will quantize it to certain levels).



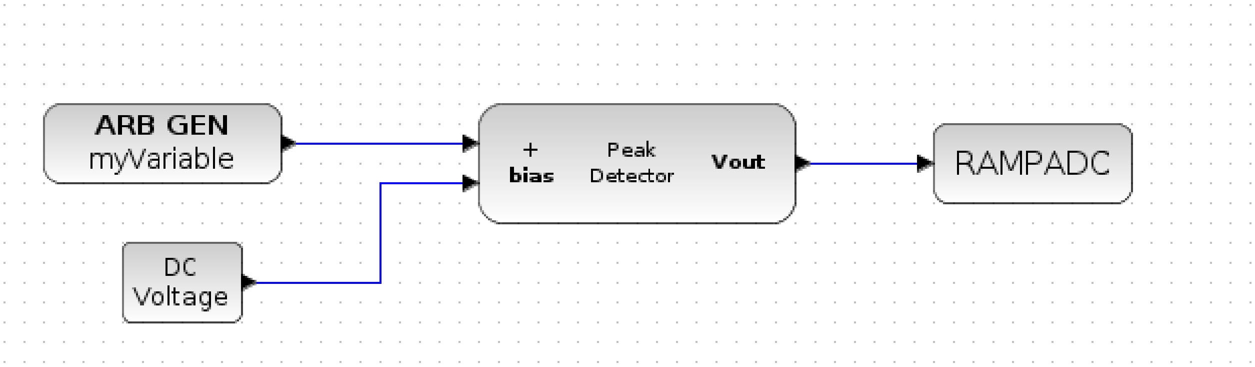
The xcos design for the using a C4(band pass filter) is shown above. For input we will use a step input to see the response of the band pass filter. Alternatively you could use a chirp as the one shown above to characterize the C4.
Here current used are as follows.
GAIN BIAS: 9.56e-9
Feedback Bias: 9.56e-11
And the floating gate biases are kept at 100e-9
Also the cap used is 512fF which corresponds to 6x in scilab multiple value request.

The response of the band pass filter is show below.

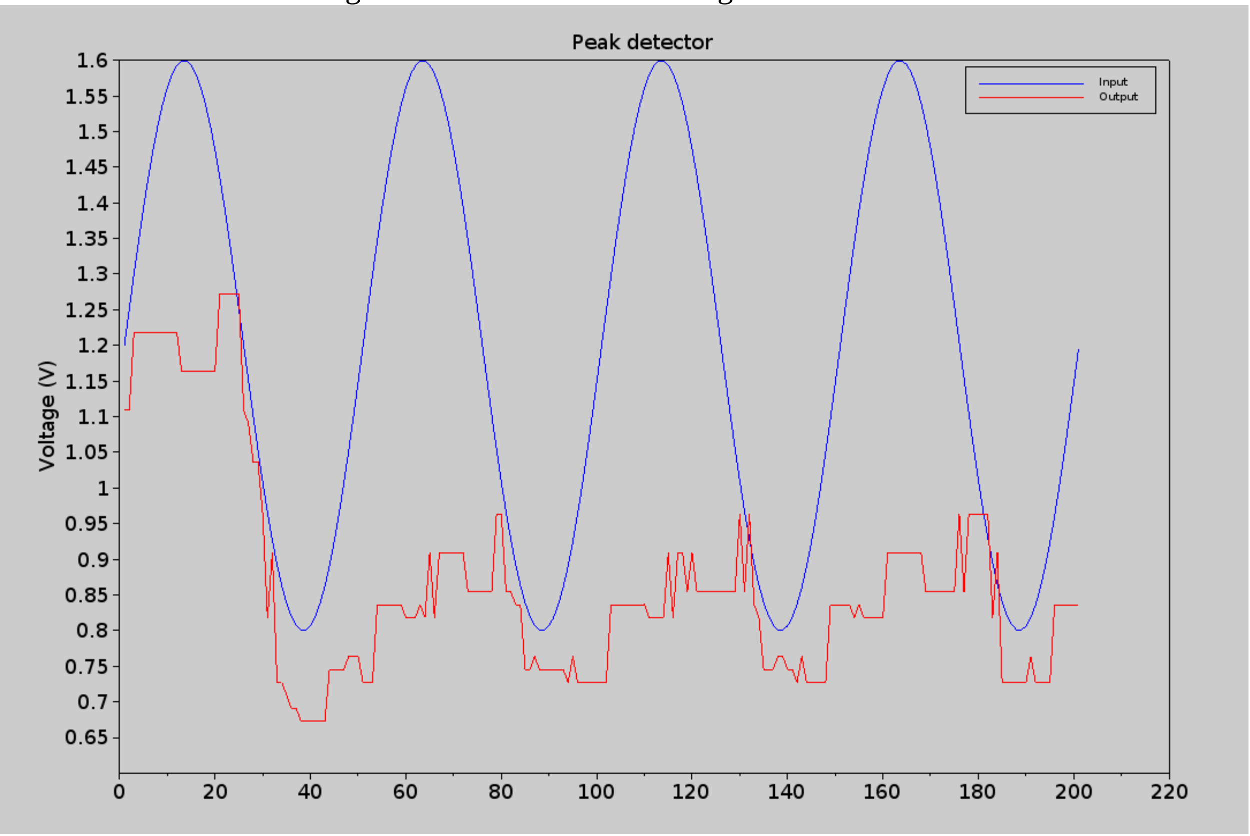


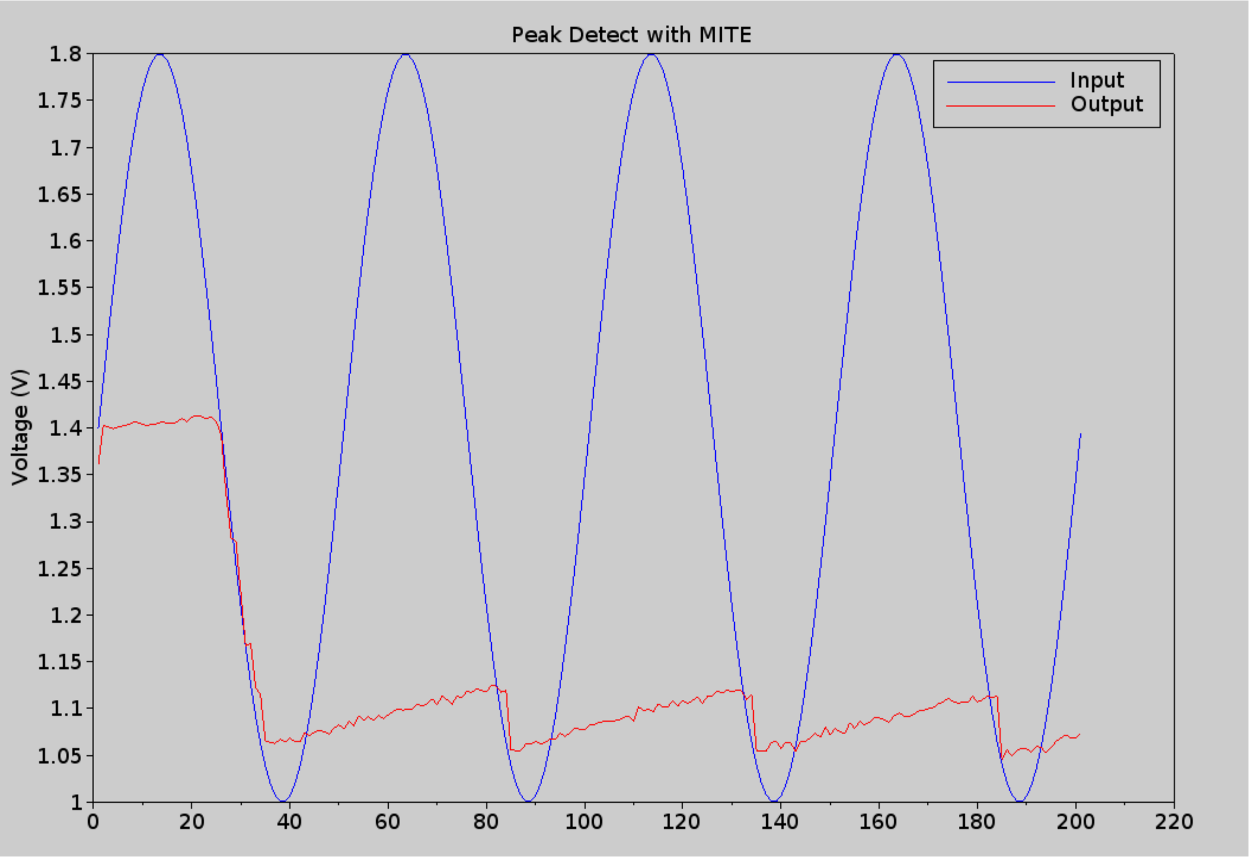
4. Minimum detector and RAMP ADC

The xocs design for this would look like as shown below. The peak detector is actually a minimum detector. For characterizing this we will use a sin signal as an input and the output would detect the minimum of the signal.



When the bias voltage is high the rise is slow and hence it follows the input in one direction but is starved in other. The output is shown below the output is quantized because of the lower resolution of the ADC. If we use mite block which uses a 14 bit ADC we will see something like the one shown in the figure called with the mite.





Both of these are ramp ADC but one of them has a higher speed the one which has lesser resolution and which is used for measuring the first peak detect figure.