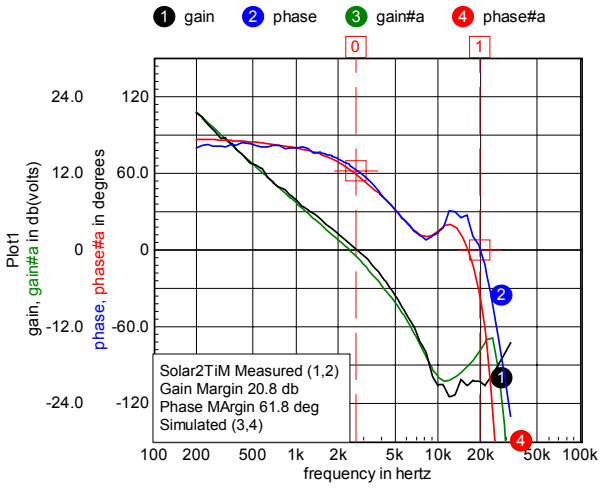
Microchip with DSP Designer

Lesson 1:

Run an AC analysis using the DSP and compare with simulation results:

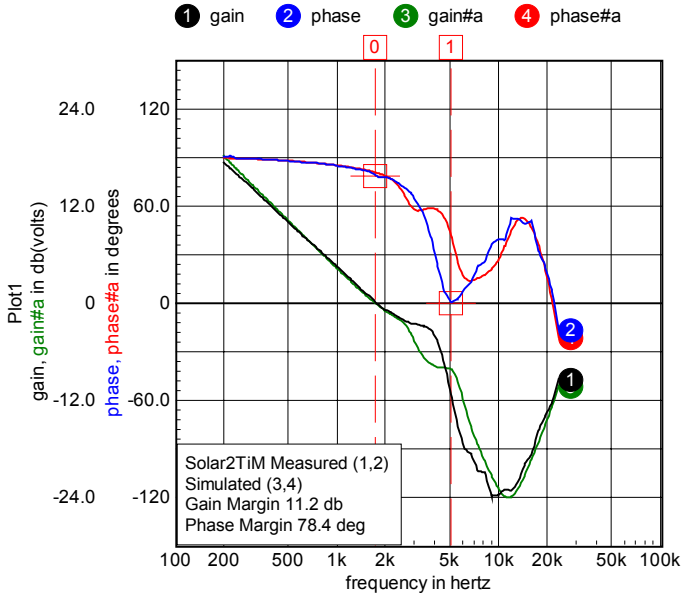
1. Connect the DSP with the solar2TiM board as shown in Getting Started, page 2.
2. Connect TP5 to sync the oscilloscope.
3. Connect the output of channel 1 to the oscilloscope, AC couple; 50mv/div x 50usec/div.
4. Check configure.h and configure.inc for TFA1=9 and TFA2=0, then Compile, program and run.
5. Open IntuScope and Browse in Add Waveforms to <DSP Communications Filter> verify the project and map file locations, then press OK.
6. Open a new Graph using the File menu.
7. Select <DSP/AC> in the calculator menu.
8. Make sure to turn off streaming test points (if snet5 is running) and disconnect the pulsed load.
9. Enter 15u for t if requested. (If you have the vcur.dwg up and have run a simulation, the .par file contains the correct value for t and you won't be asked to enter it).
10. The test signal is synchronized to TP5 so it should be visible on your scope at lower frequencies. At higher frequencies the signal gets smaller and if you are averaging samples, there will be artifacts present.
11. The AC analysis sweeps a test signal from 200Hz to about 22kHz to measure the transfer function using a single injection GFT method that you can read about it in the script syntax help. This takes about 15 seconds to produce the result.
12. Next open vcur.dwg, located in your project, and select vcurac configuration and the ac setup. Press the running man icon.
13. Open a new graph in IntuScope and press 'b' to make a Bode plot.
14. Make sure the <New Trace Only> button is checked in the <Auto Scale> group in the <Add Waveforms dialog>.
15. Then drag the gain waveform onto the DSP AC plot and select waveform 1 at the top and press 'y'. That matches the scales exactly. (Note: waveforms must have an identical x-axis and y-axis units must match. Double click on the round wfm button if they need to be renamed).
16. Drag the phase waveform over, select waveform 2 and press y to match the phase scales.
17. The plots should be pretty close (within 1dB and 10 degs); the difference represents a combination of part tolerance and modeling errors. Running at lower input voltage reduces the PWM gain, resulting in lower bandwidth.



Lesson 2:
Repeat Lesson 1, but used the MPID2.dwg and in configure.h
#define TFA1 0 // for mpidv1
#definee TFA2 9 // for mpid2
and in configure.inc

```
.equ TFA1, 0 ; ######### mpidv1 sig gen
.equ TFA2, 9 ; ########## mpid2 sig gen
```

Then build all, program and run; these changes switch signal injection to channel 2. Run as before.

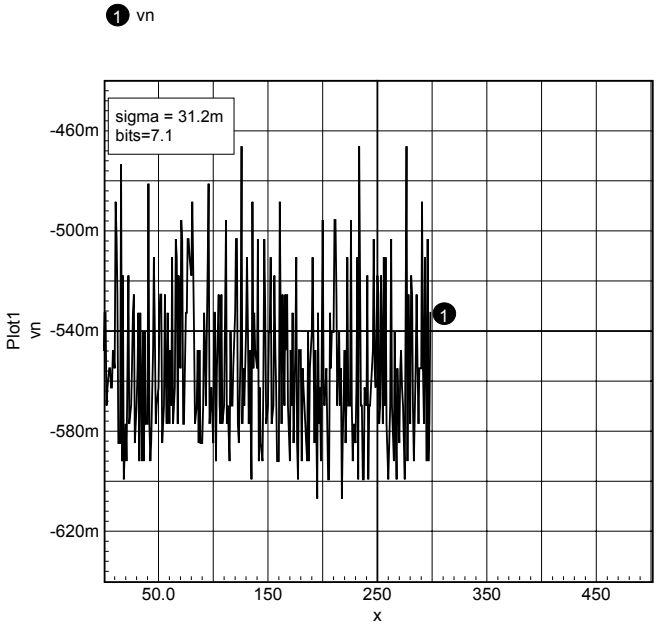


Notice that the MPID pole-zero cancellation is affected by the input filter, the output filter, and failure to match the Laplace transfer function with the z-transform PID

equations. The virtual current controller smoothes out the resonance by using inductor current feedback in the inner control loop.

Lesson 3:
Measure inductor current noise:

1. With the board running as before, open a new graph and select <DSP/indCurNoise> from the scope calculator menu. Each measurement is the average of 1024 samples. 256 of these data are collected and plotted. The standard deviation is measured and used to calculate the effective bits. You can change the number of samples used for the average using "#define AVGRADIX 10" in configure.h. The number of samples is 2 << AVGRADIX = 2^ AVGRADIX. The standard deviation should be inversely proportional to the square root of the number of sample for a Gaussian distribution.

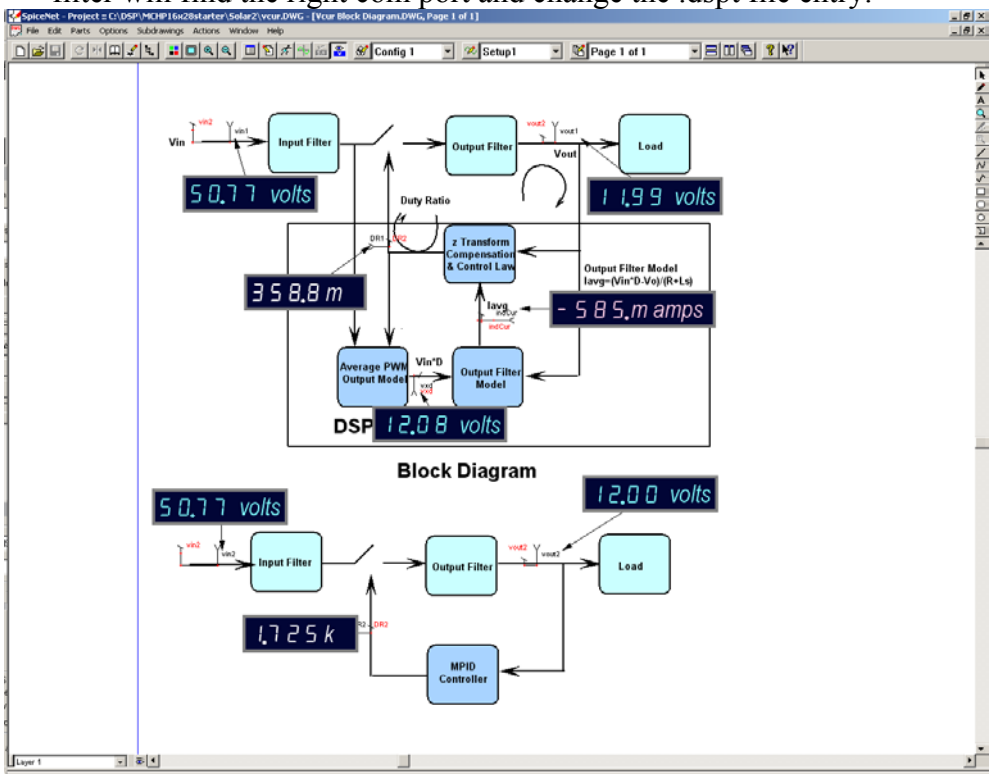


Lesson 4:
Streaming Data points:

1. Make sure the DSP is running as before and the scope <DSP Communication Filter> is active.
2. Open "Vcur Block Diagram.DWG" and select <Stream DSP Testpoint Values> in the Options menu. Each testpoint in the drawing has a name that corresponds to the vector names in the <Y Axis> list box. You can add to or modify the list and scale factors using the Browse button and choosing <DSP Communications Filter>. Values are saved in the ".dspt" file. The memory locations are read from the DSP ".map" file. It's always a good idea to re-connect to the choosing <DSP Communications Filter> any time you re-compile because the compiler may add a different address to the variable. Currently, variables can't be accessed inside of data structures so that you

must add variables and assign the members of the structure to the variables, as is done for DR1 in main..c; DR1 = MPIDv1_Statei.pwm1;

3. Note: the .dspt and .map file contain the information needed to access the DSP memory. The test point names in the .dwg file must agree with the .dspt names. There is a backup copy of the .dspt file shipped with DSP Designer in the .zip file in your project so you can recover the original .dspt file if you made inadvertent changes. Building the project creates the .map file. When you first connect the DSP communications filter, the com port may have a different number than is recorded in the .dspt file. The DSP communication filter will find the right com port and change the .dspt file entry.

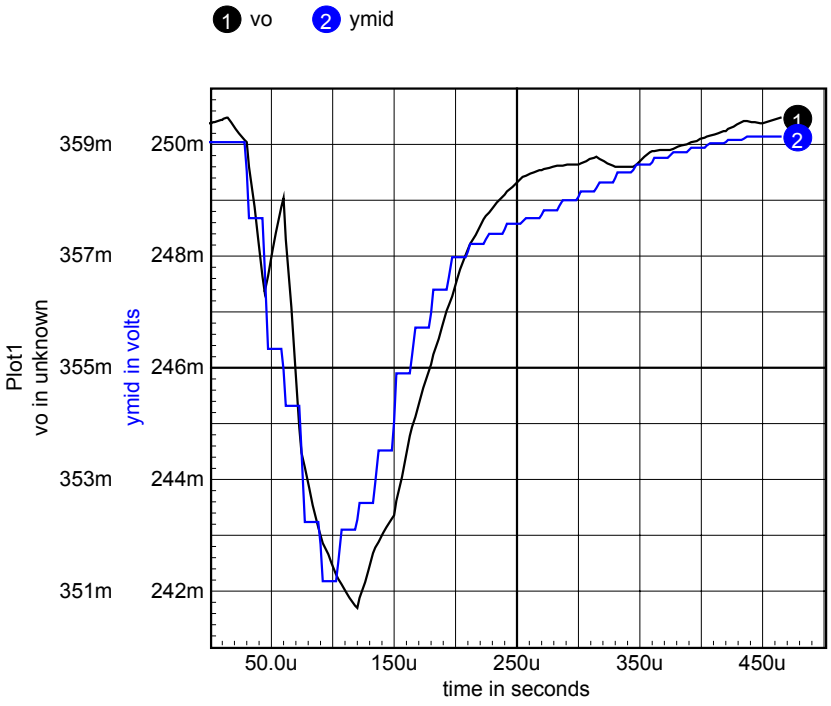


Lesson 5:

Run a TRAN test and compare with simulation results:

1. Using the same test set up used previously, connect the Q5 test point to one end of a 12 ohm resistor and the other end to the output of channel 1.
2. Sync your scope to TP5 and look at the output waveform.
3. In IntuScope open a new graph and select <DSP/TRAN> from the calculator menu. Then enter DR1 for the vector you wish to view. That will be the duty ratio of the channel 1 PWM. You can sort of view this on you scope, but you would have to convert the width of each pulse to a numerical value to plot it.
4. Open vcur.dwg and select the <start> configuration and the <tran 10m> setup and run the simulation.
5. Open a new graph and plot v(dr) in IntuScope or cross probe node DR from the schematic. Use the <[]> button and edit the waveform to start at 5m and end at 5.452m, then press the <←> button to make it start at zero. So far you have matched the time scales.

6. Next, double click on the ymid button and change xmid to time and select seconds for it units. This is needed to satisfy scope when you copy this plot to the DSP measured plot.
7. Drag the new plot to the previously plot made from <DSP/TRAN>.



Notice, the offset value is different but the transient is in reasonable agreement. You can go ahead and look at other vectors, for example, inductor current. The indcur vector exists only in the DSP Designer model.

Lesson 6

Calibrating offset current:

1. Continue with the same setup used previously.
2. Comment out the ioffset calculation in isr.c, build, program and run.
3. Remove the transient load and open a new scope window.
4. Witt the DSP Communications filter active, open a new window and run <DSP/Header> from the calculator menu.
5. Then adjust the input voltage to about 24 volts.
6. Load the line script and press <Ctrl+R> to execute it. It measures the input voltage and inductor current.
7. Increase the input voltage and press <Ctrl+R>.
8. Repeat 7 for input voltage in 6 volt increments up to 54 volts.
9. Copy and past the header and the measurements to ised and save in the cal.txt file in your project folder.
10. Use “add waveforms” to Browse for Test Files, cal.txt and plot indcur in a new graph.

11. Then do <calculator\DSP\offsetCal> copy and paste the polynomial coefficients into the isr.c file. Uncomment the ioffset calculation and paste in the new coefficients.
12. Build, program and run. Get the DSP Communicator Filter running and open "Vcur Block Diagram.DWG". Then select options\streaming test points. Then vary the input voltage to confirm that the offset current has been trimmed out.

Advanced projects:

Project 1: find the offset current sensitivity to temperature (it should be between the conductivity of Copper and the conductivity of Silicon) and account for it in the ioffset calculation.

Project 2: Correct for nonlinearity of the measured current.

Project 3: make a current limit control loop, stable of course.

Project 4. Make a peak power point tracking regulator using the Solar2TiM board connected in an "auto transformer" configuration.

Project 5. Make a sweep analysis similar to SPICE where you pass the vector address, the start values, stop value and step value. Base this analysis on the AC method in which a new value will be given every 50msec.