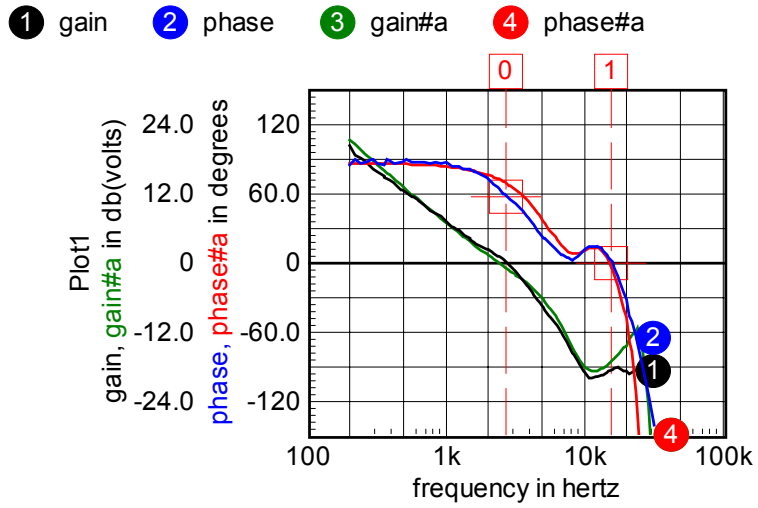
Please note the following input/output voltage requirements for the Solar2TiM board:

Startup power (barrel connector)	12volts
Input power to each of the 2 inputs:	48 volts
Regulated output at each of the 2 outputs:	12 volts

Texas Instruments
Lesson 1:

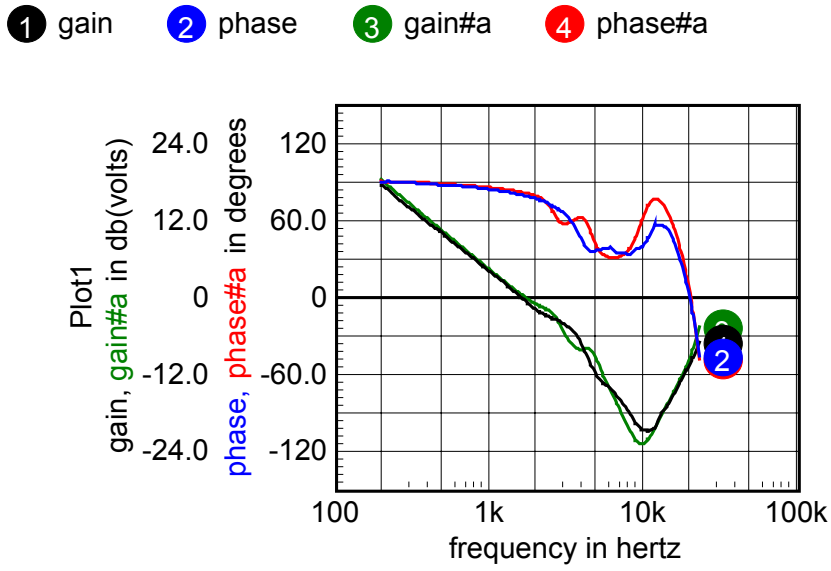
Run an AC analysis using the DSP and compare with simulation results

1. Connect the DSP with the solar1TiM 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 for TFA1=9 and TFA2=0,Compile (ReBuild All), program, load program, go main and run
5. Open IntuScope and Brows in Add Waveforms to <DSP Communications Filter> verify the project and map file locations and 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 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 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 scope 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 the waveform 1 at the top and press 'y'. That matches the scales exactly.
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.



Lesson 2:
Repeat Lesson 1, but useMPID2.dwg and in configure.h
#define TFA1 0 // for mpidv1
#definee TFA2 9 // for mpid2

Then build all, load program, go main and run; these changes switches signal injection to channel 2. Run as before

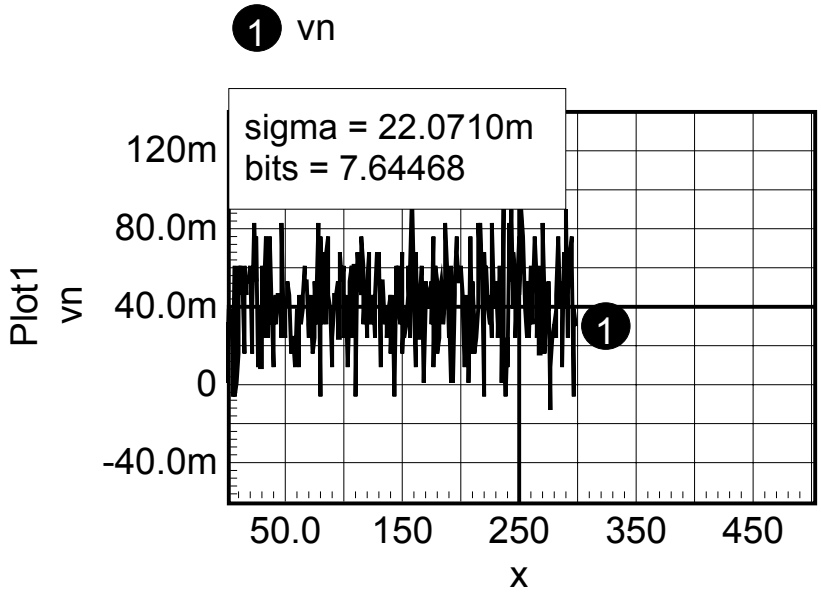


Notice that the MPID pole-zero cancellation is affected by the input filter, the output filter and the failure to match the Laplace transfer function with the z-transform PID equations. The constant, Kc, in parameters uses a lager value of filter capacitance to calculate the PID terms. That allows you to select an average of first and second stage filter values to get the best result. The virtual current controller smoothes out the

resonance by using inductor current feedback in the inner control loop. Decreasing the input voltage lowers the PWM gain.

Lesson 3:
Measure inductor current noise.

1. Open a new scope graph and with the board running as before, 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.

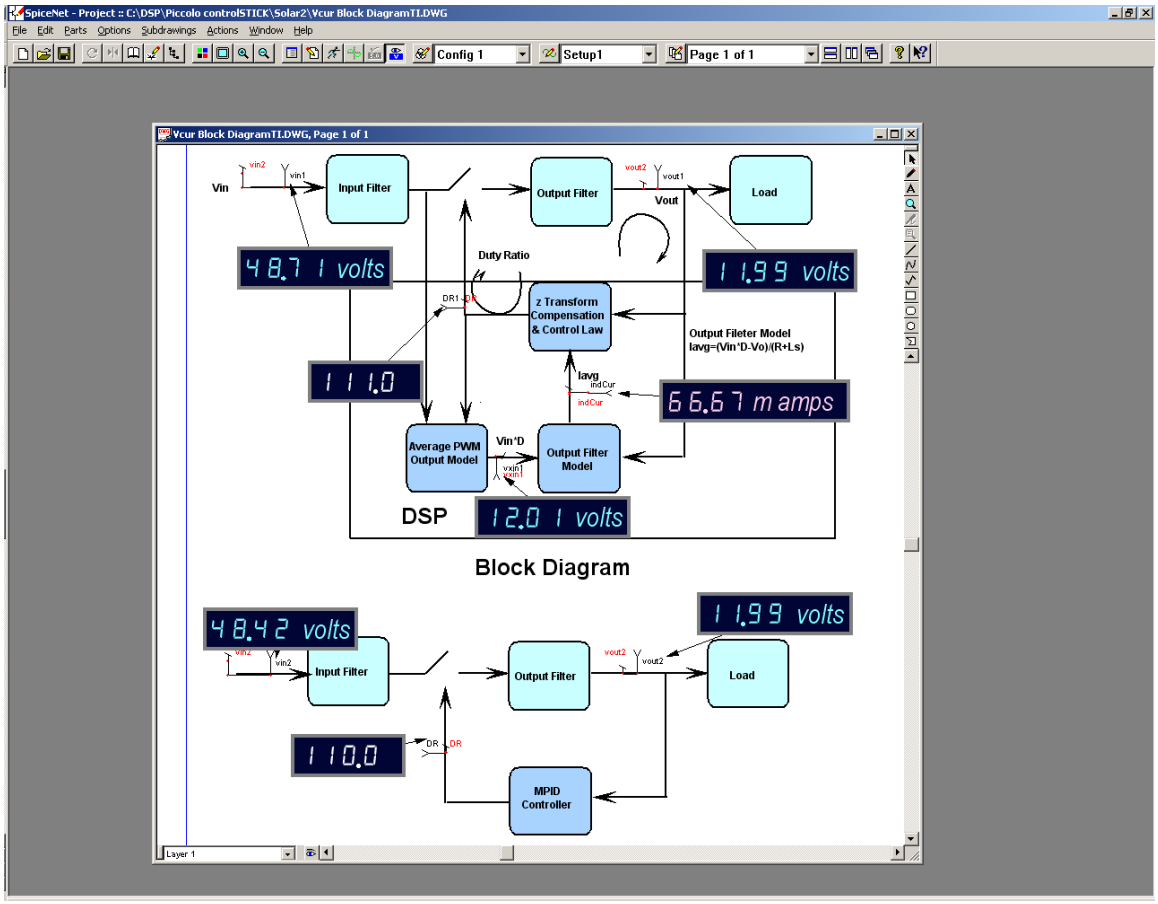


Lesson 4:
Streaming Data points

1. Make sure the DSP is running as before and the scope <DSP Communication Filter> is active.
2. Open "Vcur Block DiagramTI.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. Its always a good idea to re-connect to the choosing <DSP Communications Filter> any time you re-compile because the compiler may adding 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 struct 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.
4.

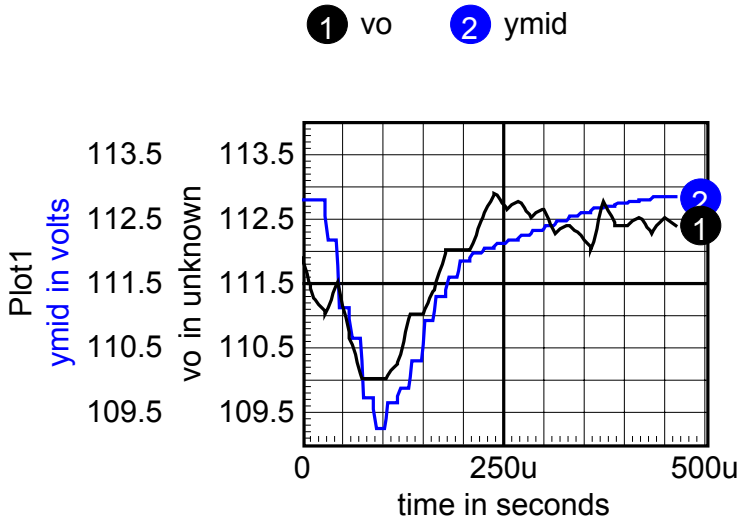


Lesson 5:

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 tfo 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>.
8. The ymid vector is Ton/PERIOD and DR1 is in Ton counts. To make a comparison enter "ymid=ymid*451" in the command window and execute using (Ctrl+r)



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 A/D converters

Background: The TI ADC's are automatically calibrated, however the input R-C network causes an offset to be introduced and resistor tolerances alter the overall scale factor. Moreover there exists cross talk from the previous sample measurement. Average virtual current is measured as (Vin*D-Vout)/R so that ADC errors need to be removed to get reasonable accuracy

1. Continue with the same setup used previously.
2. Turn off the 12 volt supply
3. Reduce the input voltage to zero and short out both input and outputs
4. Use streaming data to record the A/D converter offset values
5. Calculate hardware.h coefficients

Note:69.1 and 135 are the ADC gains for input and output respectively

#define VIN1OFFSET (Vin1 offset)*69.1
#define VIN2OFFSET (Vin2 offset)*69.1
#define VOUT1OFFSET (Vout1 offset)*135
#define VOUT2OFFSET (Vout2 offset)*135

6. Apply 48 VDC to the inputs and record vin1,vin2, vout1 and vout2 using streaming data
7. Calculate hardware.h coessifients

define VIN1SF 48/(vin1-vin1offset)*2048
#define VIN2SF 48/(vin2-vin2offset)*2048

#define bVOUT1_VIN1 (vout1(vin1=48)-vout1(vin1=0))*69.1
#define mVOUT1_VIN1 (vout1(vin1=48)-vout1(vin1=0))*69.1/(12*135)*2048

8. Apply 1.3 volts directly to V12AnaB on the Solar1TiM PCB, record the value ov vin2

#define bVIN2_VOUT2 (vin2(vout2=48)-vin2(vout2=12))*135
#define mVIN2_VOUT2 (vin2(vout2=48)-vin2(vout2=12))*135/(48*69.1)*2048

9. Rebuild the software and run it.

Lesson 6 part 2 Calibrating offset current:

Background: The ripple voltage at the output sense point can introduce and error in the output voltage. The errors cause an apparent current error as a function of duty ratio. It can be observed as a small change in output voltage vs input voltage. The following procedure eliminates that error

1. Continue with the same setup used previously, turn the 12 volt pwm power back on
2. Comment out the ioffset calculation in uart.c, build, program and run.
3. Remove the transient load
4. Record the inductor current and duty ratio for input voltage from 30 volts to 56 volts
5 fill out a scope script as follows

for the following inputs 28 32 36 40 44 48 52 56
compose DR values DR1 DR2 … DR8
compose IL values indcur1, … indur8 8
plot IL DR
IR = IL*135

Execute the script in scope to plot the offset vs duty ratio, the do a 2nd order (poly2)
Script using calculator\functions\poly2
Paste the coefficients into uart.c and replace them in the ioffset equation.

6. Build, program and run. Get the DSP Communicator Filter running and open "Vcur Block Diagram.DWG". Then select options\streaming test points. The offset current inductor current should be nearly zero as you sweep the input voltage

Lesson 6 part3

Background: The input voltage is sampled near the beginning of the PWM ON state. Its current will increase during that interval so that the input voltage will be above the average value. The input voltage can be corrected using mINDCUR_VIN1 in hardware.h.

1. With no load an 48 volt input, the offset current should be small.
2. Apply a 2.35 anp load by loading the output with a 5.1 ohm resistor
3. record the vin and indcur
   calculate
   #define mINDCUR_VIN1 dvin1/(2.35*135)*69.1*2048
4. Program and run, incdur should be approximately 2.35 amps

Note: instead of using a 5.1 ohm 50 watt resistor you can connect the 2 outputs through a .25 ohm 5 watt resistor and in ISR.c initialize dv = -95 for a 2.35 amp load. Later we will use this technique to simulate a solar panel using a .25 ohm 20 watt resistor

1. Increase the input voltage and press <Ctrl+R>
2. Repeat 7 for input voltage in 1 volt increments up to 19 volts.
3. Copy and paste the header and the measurements to ised and save in the cal.txt file in your project folder.
4. Use the <add waveforms> dilog to Browse for Text Files, cal.txt and plot indcur in a new graph.
5. Then do <calculator\DSP\offsetCal> copy and paste the polynomial coefficients into the uart.c file. Uncomment the ioffset calculation and paste in the new coefficients.
6. 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 Solar1TiM 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.