SL-INTERPHASE
Electronic Design Service

Test Bench Analysis

Zong International Ads1258Accumulator

Rev -, Jan 19, 2011

1. BACKGROUND

SL Interphase to be contacted by Zong International to perform VHDL code review, modification, and simulation for a particular problem identified with the MXTX-30 project. The work identified is a problem with a vhdl 'driver' for a 32-channel A/D, to basically grab samples, and do an accumulation on them into a separate ram location for each channel (I then get the accumulator over spi with an arm processor, and divide by numaccum in the arm). The code functions, but it's unstable; clearly have timing issues between grabbing samples from the a/d, accumulating them in a ram, and grabbing them with the arm processor. SLI will make an effort to arrange the code within a stable state machine thereby making the process more stable.

2. ENGINEERYING SUMMARY OF PERFORMANCE

- A VHDL testbench was created to exercise the ads1258accumulator module. This testbench reads in stimulus values from two text files. One text file contains simulated ADC data the other contains timing descriptions of simulated microcontroller reads. While these stimulus files could be used to apply values to any number of the ADC channels to simplify analysis of the results only a single channel is exercised in the current versions of these files.
- It was postulated that the errors that had been observed in actual hardware resulted when a microcontroller read happened in close proximity to a recent ADC update of the same channel. The stimulus files were created to focus on this region of interest and essentially “walk” the microcontroller read through the ADC sample capture state machine.
- During the course of simulation there were six cases where the relative timing of the microcontroller read in relation to the ADC update resulted in an erroneous value being loaded into the accumulator. Screen shots of these error scenarios are included later in this document. While there are subtle differences in the internal signal timing for each of these error conditions the root condition of the failure is the same. The errors seem to result from the microcontroller attempting to reset the contents of the accumulator after the state machine that is capturing the ADC samples has read the current value out of the accumulator and is then going to subsequently add the new sample to this value and then write the updated accumulated value back into memory. When this updated value is written back after an accumulator reset has been performed it essentially has the effect of making it as if the accumulator reset never happened. While this, in and of itself, does not account for the “missing” captured samples that have been observed it is still probably not the desired mode of operation.
- What I think may be even more problematic is what happens in the situation captured in the final error waveform where the final write cycle of the update to Port A overlaps with the clearing cycle of the write occurring on Port B. While it is a dual port RAM, it is my understanding that concurrent writes to the same address on the separate ports results in undefined behavior. While not reflected vary well in this functional simulation which features a direct instantiation of the memory primitive I believe that higher level behavior models typically default to generate warnings for “memory collisions”. I believe the only way to avoid this potential corruption is to architect the module in such a way that ensures that concurrent writes to the same address never happen. This is obviously complicated by the fact that the interfaces on both sides of the accumulator are essentially operating asynchronously. I believe the only options for avoid this is to closely control the timing to both sides

of the interface, or restructure the logic such that writes are only performed on one of the two memory ports.

3. BASIC FUNCTION OF THE “Ads1258Accumulator”

The main signals of interest in the following screen captures are the following:

1. adcsamplelatched - indicates the arrival of new data from the ADC
2. readadcsample - indicates an attempt by the microcontroller to read the accumulated value

The primary difference between each of the subsequent screenshots (SECTION 4) is the delay between the arrival of these two pulses. The delay ranges from 30-80ns (with each 10ns delay representing one clock cycle) and is shown in each screen capture by the delta value between the two cursors which are placed at the rising edge of the aforementioned signals.

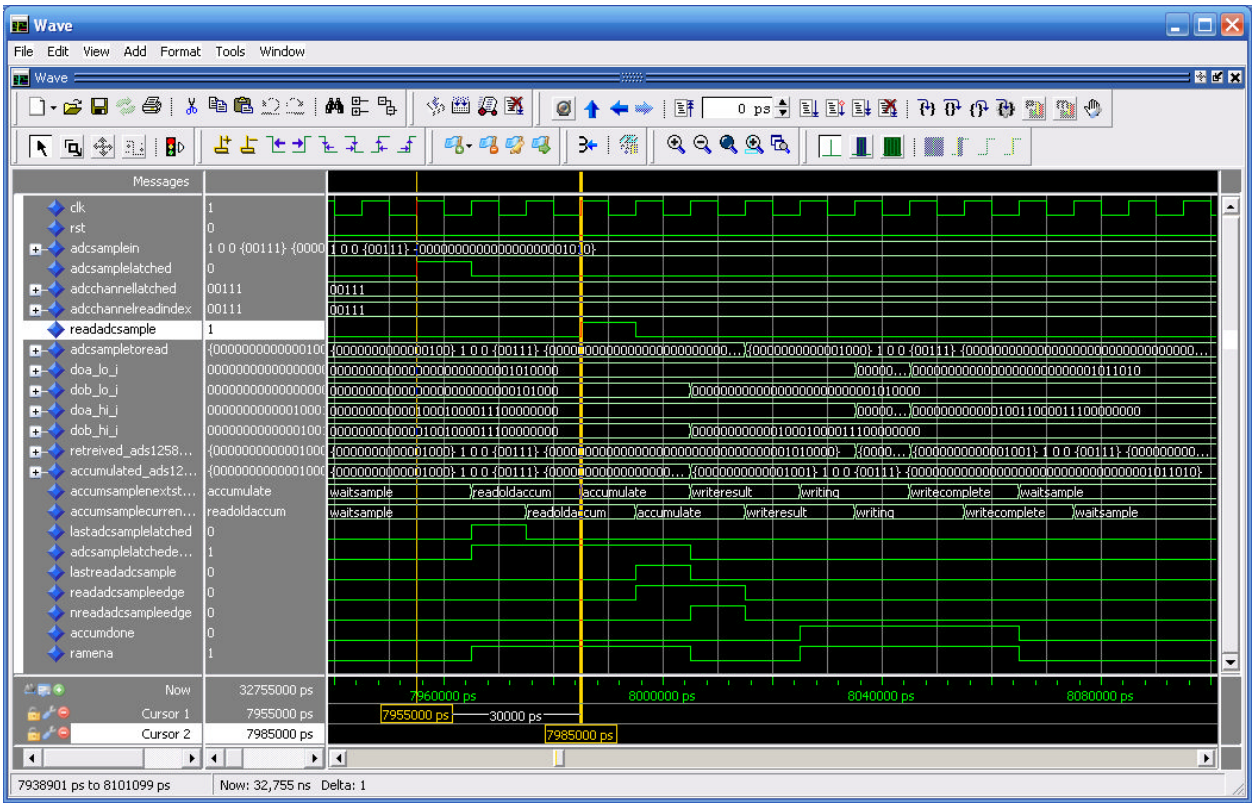
The primary remaining signals of interest are:

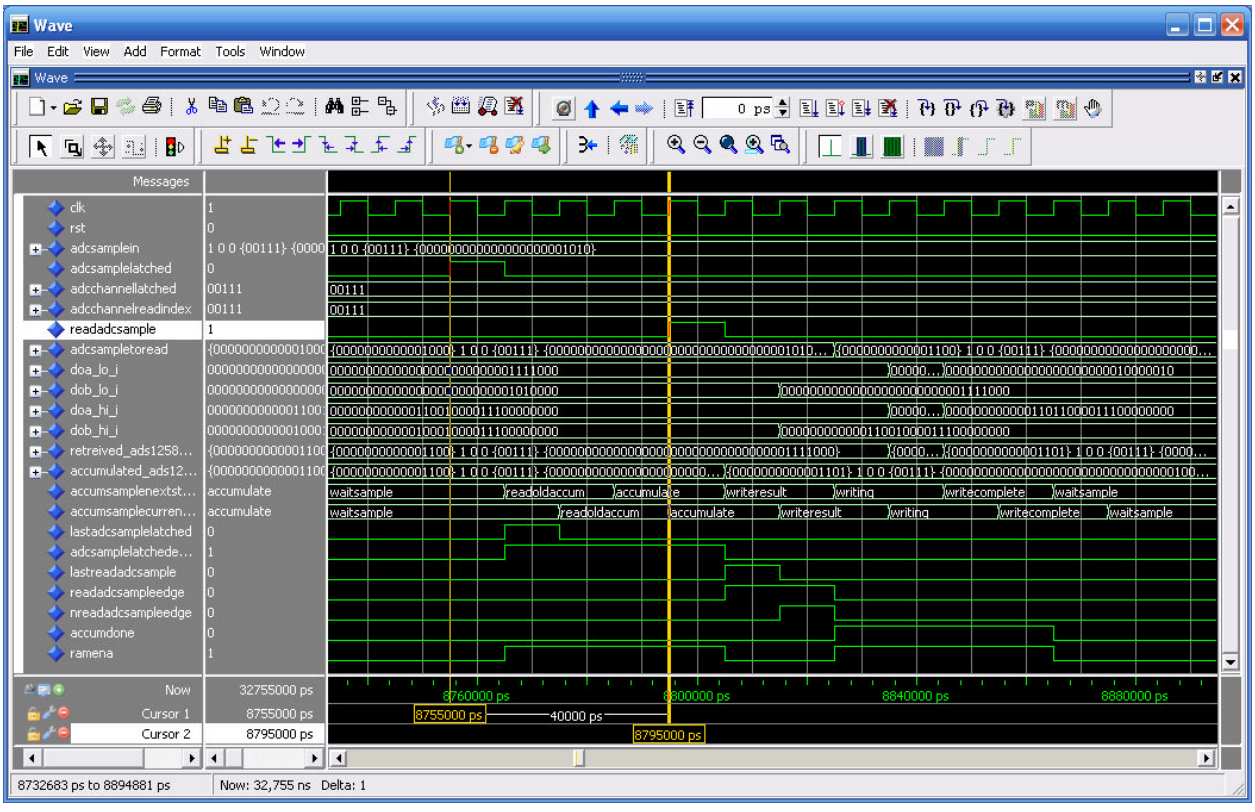
1. ramena - used to enable Port A of the BlockRAM for reading or writing
2. accumdone - used to write the accumulated value into Port A of the BlockRAM
3. readadcsampleedge - used to enable Port B of the BlockRAM for reading or writing
4. nreadadcsampleedge - used to write a value of all 0's to Port B of the BlockRAM and reset the accumulator

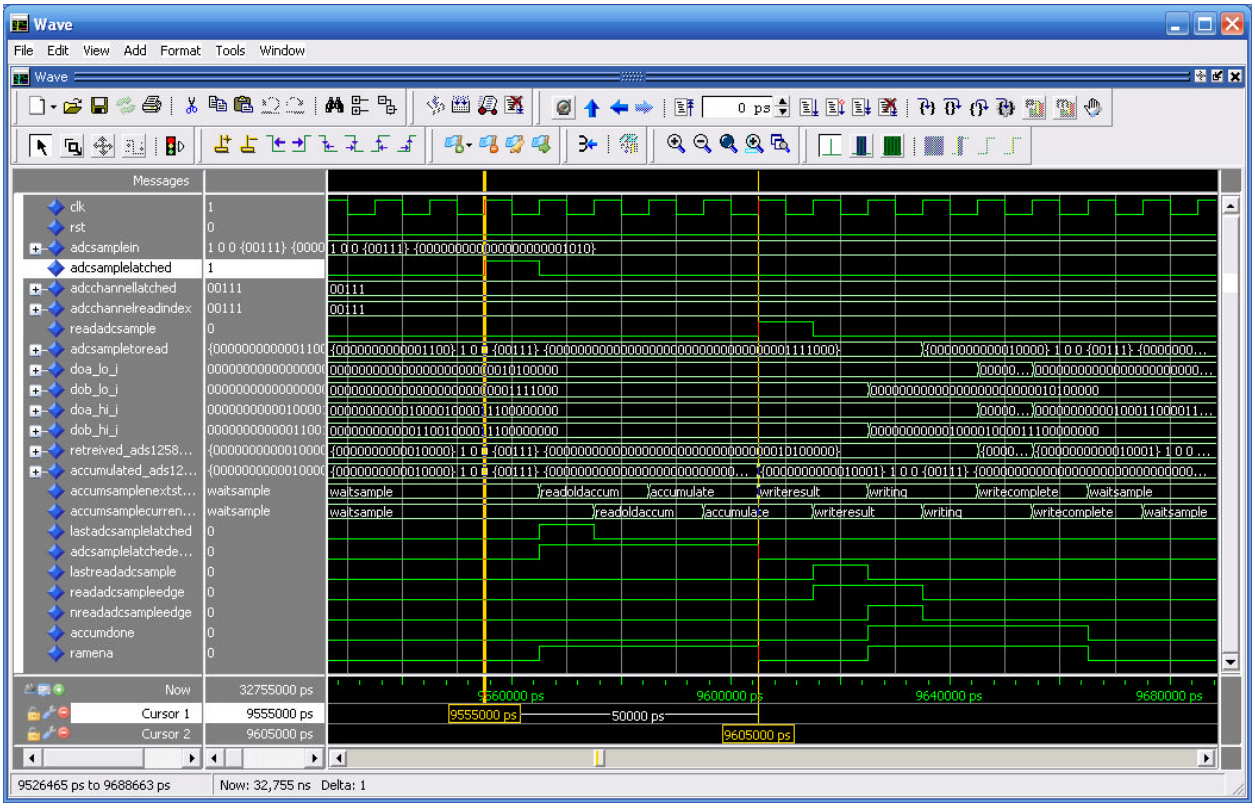
3.1 HYPOTHESIS FOR LOSS OF DATA

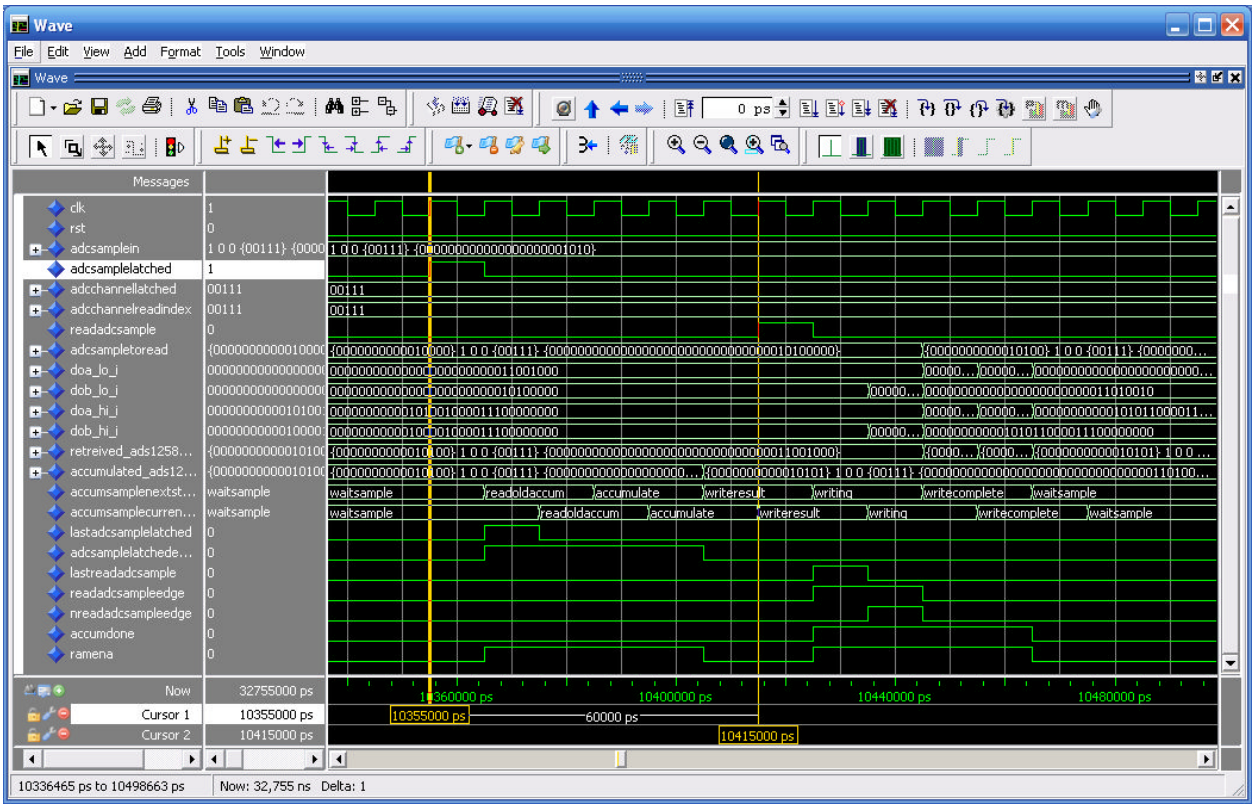
As the delta between the first two signals (adcsamplelatched and readadcsample) increases by one clock cycle in each screenshot, the relative timing of the last two signals (readadcsampleedge and nreadadcsampleedge) shift to the right by by the same amount. In all cases though the position of this read and subsequent write to Port B occurs after the accumulated value has been already been read out of Port A which will subsequently be added to the current sample and stored back in the accumulator RAM. In the last four screenshots the write to Port B overlaps with the write to Port A. Of those four, in all but the last screen shot since the update to Port A actually spans four clock cycles and continues to be applied to the BlockRAM even after the clearing write to Port B the end value stored at that particular location in memory is it's old value added to the current sample from the ADC. In the last screenshot I believe the actual contents of the RAM are unknown because a write is being attempted and concluded on Port A and Port B at the same time.

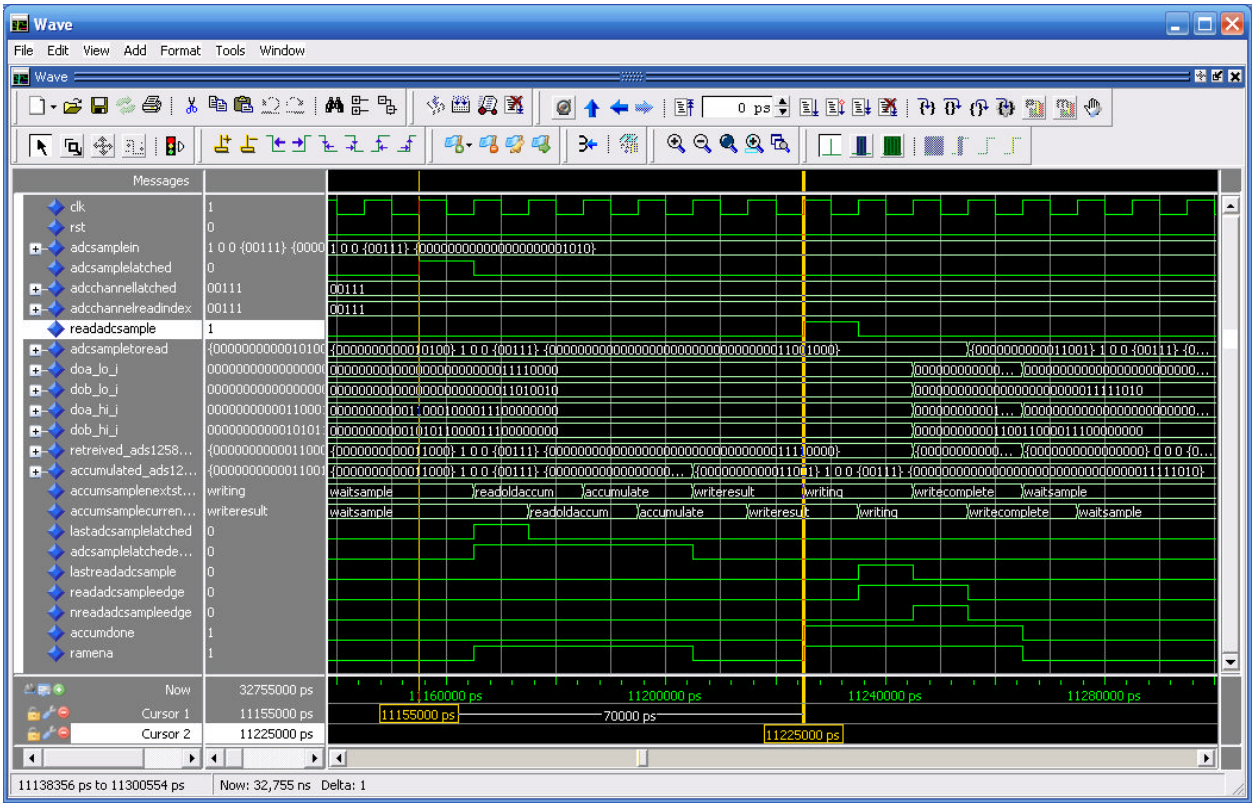
4.0 TEST BENCH SCREEN SHOTS

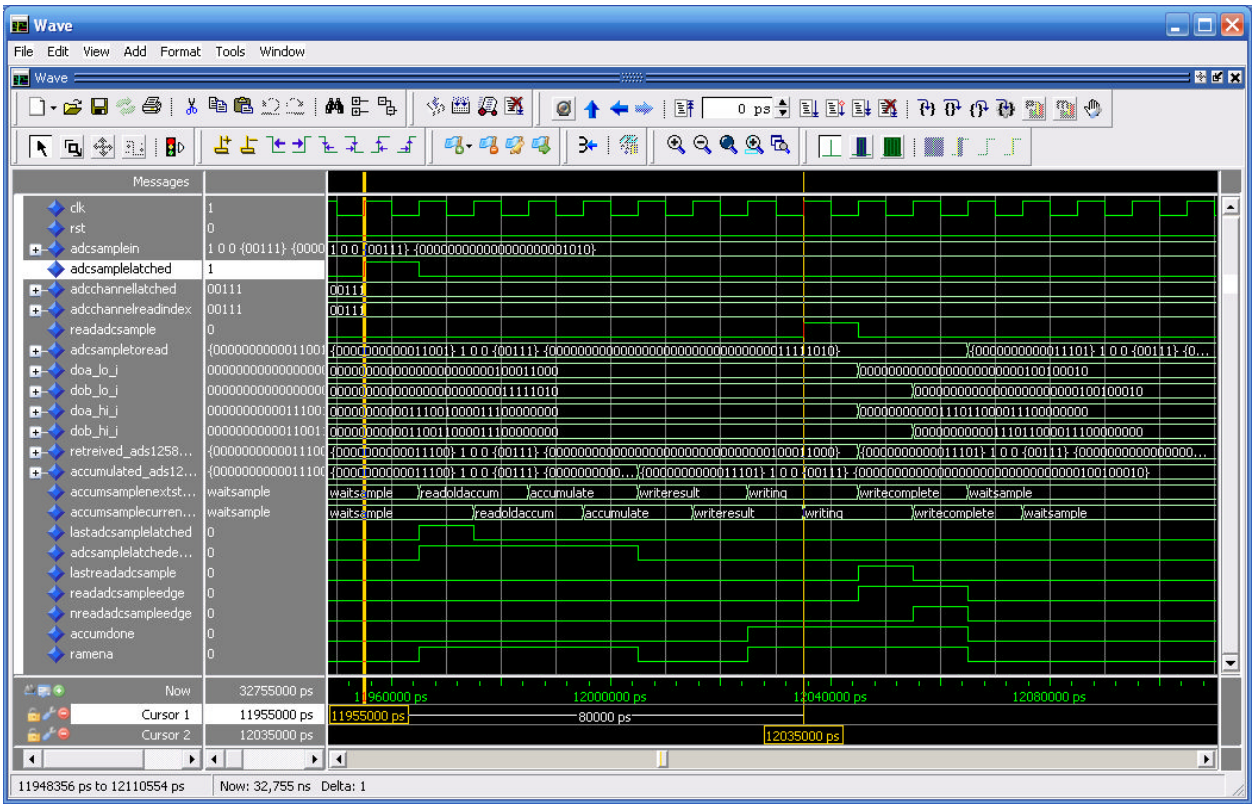





Wave
File Edit View Add Format Tools Window
Messages
clk
rst
adcsamplein
adcsamplelatched
adcchannellatched
adcchannelreadindex
readadcsample
adcsampletoread
doa_lo_i
dob_lo_i
doa_hi_i
dob_hi_i
retreived_ads1258...
accumulated_ads12...
accumsamplenextst...
accumsamplecurren...
lastadcsamplelatched
adcsamplelatchede...
lastreadadcsample
readadcsampleedge
nreadadcsampleedge
accumdone
ramena
Now
Cursor 1
Cursor 2
9555000 ps
50000 ps
9605000 ps
9560000 ps
9600000 ps
9640000 ps
9680000 ps
waitsample
readoldaccum
accumulate
writeresult
writing
writecomplete
9526465 ps to 9688663 ps
Now: 32,755 ns Delta: 1


Wave
File Edit View Add Format Tools Window
Messages
clk
rst
adcsamplein
adcsamplelatched
adcchannellatched
adcchannelreadindex
readadcsample
adcsampletoread
doa_lo_i
dob_lo_i
doa_hi_i
dob_hi_i
retreived_ads1258...
accumulated_ads12...
accumsamplenextst...
accumsamplecurren...
lastadcsamplelatched
adcsamplelatchede...
lastreadadcsample
readadcsampleedge
nreadadcsampleedge
accumdone
ramena
Now
Cursor 1
Cursor 2
10355000 ps
60000 ps
10415000 ps
10360000 ps
10400000 ps
10440000 ps
10480000 ps
waitsample
readoldaccum
accumulate
writeresult
writing
writecomplete
10336465 ps to 10498663 ps
Now: 32,755 ns Delta: 1


Wave
File Edit View Add Format Tools Window
Messages
clk
rst
adcsamplein
adcsamplelatched
adcchannellatched
adcchannelreadindex
readadcsample
adcsampletoread
doa_lo_i
dob_lo_i
doa_hi_i
dob_hi_i
retreived_ads1258...
accumulated_ads12...
accumsamplenextst...
accumsamplecurren...
lastadcsamplelatched
adcsamplelatchede...
lastreadadcsample
readadcsampleedge
nreadadcsampleedge
accumdone
ramena
Now
Cursor 1
Cursor 2
11155000 ps
70000 ps
11225000 ps
11138356 ps to 11300554 ps
Now: 32,755 ns Delta: 1


Wave
File Edit View Add Format Tools Window
Messages
clk
rst
adcsamplein
adcsamplelatched
adcchannellatched
adcchannelreadindex
readadcsample
adcsampletoread
doa_lo_i
dob_lo_i
doa_hi_i
dob_hi_i
retreived_ads1258...
accumulated_ads12...
accumsamplenextst...
accumsamplecurren...
lastadcsamplelatched
adcsamplelatchede...
lastreadadcsample
readadcsampleedge
nreadadcsampleedge
accumdone
ramena
Now
Cursor 1
Cursor 2
11955000 ps
80000 ps
12035000 ps
11948356 ps to 12110554 ps
Now: 32,755 ns Delta: 1

5.0 HOURS FOR ANALYSIS AND SIMS

Subcontractor		
Items	**Hours**	**Sub Total**
Inspection of Code and inter-related blocks	**6**	
Test Bench Creation	**8**	
Wave Form Generation and Analysis	**4**	
TOTAL HOURS		**18hr**

5.