2-Antenna 8-bit GPU based CORRELATOR TESTING PROCEDURE

Version 1 , Released Date : 4/11/2011.

Written By : S. Harshavardhan Reddy/I. M. Halagali.

Note : This colour : used for Main titles.
This colour : used for sub/sub-sub titles.
This colour : used for command lines.
This colour : used for the response from the from the PC on terminal.
This colour : used for procedure writeup/information/note.

[CTRL_PC]	: 192.168.4.68
[USER_CTRL_PC]	: gmrt
[CTRL_PC_PASS]	: gmrttifr
[GPU_PC]	: 192.168.4.75
[USER_GPUPC]	: jroy
[GPU_PC_PASS]	: ncra123
[TENGB_IP]	: 192.168.10.11
[TENGB_PORT]	: 10000
[SOFT_DIR]	: /home/jroy/USERS/harsha/psrdada/gmrt_8bit_4k_log_ppm
[PYTHON_DIR]	: /home/gmrt/harsha
[DELAY_DIR]	: /home/jroy/delay_cal
[TAX_DIR]	: /home/jroy/tax
[CONFIG_PY]	: prog_pps_8bit.py

I SETUP :

1. HARDWARE SETUP

A. NOISE SOURCE TEST

The following connections needs to be done for noise source test.

1. Connect the ROACH board to the controlling PC using 100MBPS ethernet.

2. Feed the clock of 400MHZ,0dbm to the clk_i input of the iADC 0 card from the signal generator. Also feed the PPM signal to the Sync input of the iADC card.

3. Connect the input signals to I+ and Q+ of the iADC 0 from the noise generator. The input signal should be of -12 to -17 dbm(with 2 way power splitter)@200MHz bandwidth at the iADC 0 card input.

4. Connect the ROACH board to the GPU PC using 10gbE cable. Connect one end of 10gbE cable's CX4 connector to the CX-4 port 0 of ROACH and the other end to the 10gbE card on the GPU PC.

5. Turn ON the ROACH board

B. ANTENNA TEST

The following connections needs to be done for antenna test.

1. Connect the ROACH board to the controlling PC using 100MBPS ethernet.

2. Feed the clock of 400MHZ,0dbm to the clk_i input of the iADC 0 card from the signal generator. Also feed the PPM signal to the Sync input of the iADC card.

3. Connect the antenna 1 to I+ and antenna 2 to Q+ of the iADC 0. Ensure that the power levels at the iADC inputs are between -15dbm to -20dbm with maximum bandwidth of 200MHz.

4. Connect the ROACH board to the GPU PC using 10gbE cable. Connect one end of 10gbE cable's CX4 connector to the CX-4 port 0 of ROACH and the other end to the 10gbE card on the GPU PC.

5. Turn ON the ROACH board

2. SOFTWARE SETUP

A. NOISE SOURCE TEST

1. Open a terminal and log into the GPU_PC. $ssh -X [USER_GPU_PC]@[GPU_PC]
2. Move into the DELAY_DIR. $cd [DELAY_DIR]
3. Edit the 2 header files for antenna and source information
   a. Antenna used : $vim sampler.hdr

      eg. SMP000 = C00 USB-130

      SMP001 = C00 USB-130

      (keep the same antenna name for noise test)

   b. Source information is not necessary for noise source test.
4. Move into the SOFT_DIR and install it using make and make install.

   $cd [SOFT_DIR]

   $make

   $make install

B. ANTENNA TEST

1. Open a terminal and log into the GPU_PC. $ssh -X [USER_GPU_PC]@[GPU_PC]

2. Move into the DELAY_DIR. $cd [DELAY_DIR]
3. Edit the 2 header files for antenna and source information

   a. Antenna used : $vim sampler.hdr

      eg. SMP000 = C09 USB-130

      SMP001 = E06 USB-130

   b. Source Information : The RA & DEC information required can be obtained from by following these steps ;

      Login: observer@shivneri

      Password: obs@gmrt

      Enter the following at the command prompt

      observer@shivneri:/home/observer 32> work

      observer@shivneri:/home/observer/05aug2006/work 35>./user5.5

      50 # user id number

      gts '3c48' # If your source is '3c48'

   Copy the 4th and 5th Appr. values from the "Process out" entry and paste as the I and II arguments in the line of '3c48' in source.hdr file. Remove the comment (#) for '3c48' and comment all other lines , if any. exit

      # to exit.

4. Move into the SOFT_DIR and install it using make and make install.

   $cd [SOFT_DIR]

   $make

   $make install

II CONFIGURING THE ROACH :

In this step, we will program the ROACH with the [BOF_FILE] and configure parameters like the ip address and port of the destination 10gbE. After running the commands given below, the ROACH starts sampling the input signals given to iADC card and makes UDP packets that can be sent over 10GbE. The size of packet is 8k bytes with 4k bytes of data from each iADC input signal.

[Note: We haven't yet started sending packets over 10GbE]

1. Open a terminal and log into CTRL_PC. $ssh -X [USER_CTRL_PC]@[CTRL_PC]
2. Move into the PYTHON_DIR. $ cd [PYTHON_DIR]
3. Run the configuration scrip. $ ./[CONFIG_PY]

III CREATING SHARED MEMORY :

After configuring the ROACH, we need to create shared memory to write the data from the 10GbE. Here, we create two shared memories with 4 buffers each of 256 MB. One shared memory for each of the input signals. The shared memories are represented by the keys 'bada' and 'cada'.

1. Open a terminal and log into GPU_PC. $ssh -X [USER_GPU_PC]@[GPU_PC]

2. Move into the DELAY_DIR. $ cd [DELAY_DIR]

3. Run the script for creating shared memories $ ./ada.csh

This command destroys the shared memories, if any, created before and creates new shared memories. After running this command, there will be message printed which looks as given below

Destroyed DADA data and header blocks
Destroyed DADA data and header blocks
Created DADA data block with nbufs=4 bufsz=268435456
Created DADA header block with nhdrs = 8, hdrsz = 4096 bytes
Created DADA data block with nbufs=4 bufsz=268435456
Created DADA header block with nhdrs = 8, hdrsz = 4096 bytes

IV RUNNING THE CORRELATOR PROGRAM :

After configuring the ROACH and creating the shared memory, we need to run the correlator. By running the commands given below, the CPU-GPU PC starts the correlator program, initialises the variables needed and waits for the shared memory data. After receiving the shared memory data, the data is copied into the GPU and correlation operation is done in GPU. After the correlation operation, the results are written back to CPU and finally written to a file named 'out.txt' in the [DELAY_DIR] folder. Follow the steps given below

1. Open a new terminal and log into GPU_PC. $ ssh -X [USER_GPU_PC]@[GPU_PC]

2. Move into the directory DELAY_DIR. $ cd [DELAY_DIR]

3. Run the correlator command. $ gmrt_correlator -k ~/dada_key_list -g 1024 -q 4 268435456 2048 2 -w

After running this command, the program waits for data from the shared memories. In the command, the arguments give the following information to the process

-k [filename] This argument provides the location and name of the file in which the shared memory keys are present

-g [number] This argumentt gives the number of FFTs that are to be computed parallelly on GPU.The optimised value is 1024

-q [count] This argument gives the number of streams to be run on GPU.The optimised value is 4. The -g and -q options implies 1024 FFTs are parallelly computed using 4 streams on GPU.

The number 268435456 gives the integration time. This number gives the amount of data to be accumulated for one integration. The time to sample this amount of data at 400MHZ comes out to be 0.671088. So, our integration time is 0.671088. For every one integration the program writes the integrated output data to the file 'out.txt'.

The number 2048 gives the FFT size.

The number 2 gives the number of antennas or the number of input signals.

-w This argument informs the program to write the output to file 'out.txt' located in [DELAY_DIR}

More information about the function can be known by command $ gmrt_correlator -h.

The following messages are printed on the screen:

Reading antsys header

Reading sampler header

Reading source header

Initialising cuda events

Initialising data arrays

Allocating CUDA memory

Opening DADA datablocks

V RUNNING THE PACKET CAPTURE PROGRAM :

As the correlator program is running and waiting for the data from shared memory, we need to capture the data over 10gbE cable and write it to the shared memories. By running the following commands, the GPU_PC initialises the 10GbE socket and waits for packets to be sent. On receiving the packets, this program unpacks the packets and writes the data from each signal to corresponding shared memories. As mentioned earlier, each packet is of 8k bytes with 4k bytes from each input signal.

1.Open a terminal and log into GPU_PC. $ ssh -X [USER_GPU_PC]@[GPU_PC]

2. [USER]@[GPU_PC]$ cd [DELAY_DIR]

3. [DELAY_DIR]$ gmrt_udpdb_multi -i [TENGB_IP] -p [TENGB_PORT] -H ~/header 2 bada cada 0 0

The arguments give the following information to the process

-i [ipaddress] ipaddress of the interface from which the data is being received

-p [port] port that is open for UDP packets to be received

-H [filename] location and name of the header file.This header file gives the information regarding the sampling frequency,number of bits per sample etc. The file contents can be viewed using the command $cat ~/header

The number 2 in the command arguments gives the no. of antennas or the input signals

bada and cada are the keys of shared memories to which the data is to be written

The numbers 0 0 are the delays for each input. We will come back to this later. For now just keep them 0 0

-n [secs] time for which the data needs to be received.After the amount of time given using this argument the packet capturing program and the correlator program stops.

More information can be obtained by using the command $ gmrt_udpdb_multi -h

While running this program, the following messages will be printed on the screen. [Note: The numbers may vary]

[2012-09-04-15:02:02 077088] Using current time for UTC_START = 2012-09-04-09:33:00

[2012-09-04-15:02:02 077188] B/sample=2.000000, TSAMP=0.002500, B/second=800000000.000000

[2012-09-04-15:02:02 077738] header marked filled

start of udpdb_multi : 1346751122

[2012-09-04-15:02:02 077865] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=0 dropped=0

timestamps 1346751180 0

[2012-09-04-15:02:02 082848] stream[0] delay offset -0.0000008540631597 s -> -341 bytes -> 57685 bytes

[2012-09-04-15:02:02 082903] stream[1] delay offset -0.0000008540631597 s -> -341 bytes -> 57685 bytes

[2012-09-04-15:02:03 078791] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2722325 dropped=0

[2012-09-04-15:02:04 079797] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2988382 dropped=0

[2012-09-04-15:02:05 080806] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2988729 dropped=0

[2012-09-04-15:02:06 081813] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2988428 dropped=0

[2012-09-04-15:02:07 082809] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2988449 dropped=0

[2012-09-04-15:02:08 083824] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2989368 dropped=0

[2012-09-04-15:02:09 084833] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2988609 dropped=0

[2012-09-04-15:02:10 085833] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2988643 dropped=0

[2012-09-04-15:02:11 086845] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2988902 dropped=0

[2012-09-04-15:02:12 087846] T=0.0, R=0.0, D=0.0 MiB/s bsleeps=2988972 dropped=0

VI START SENDING PACKETS:

1. Go to the terminal corresponding to the step 2 which is "**CONFIGURING THE ROACH**"
2. Run the script that starts sending packets. $ ./set.py

   After running this script wait for the next minute boundary to start the transmission of packets.

   At the start of the next minute, the "**PACKET CAPTURE PROGRAM**" starts receiving the packets and writes the data into shared memories. We can see the following messages in that window:

[2012-09-04-15:04:14 497856] START : received packet 1

[2012-09-04-15:04:14 634595] T=109.5, R=109.5, D=0.0 MiB/s bsleeps=2509364 dropped=0

timestamps 1346751300 671088

[2012-09-04-15:04:15 504848] stream[0] delay offset -0.0000008532218203 s -> -341 bytes -> 57685 bytes

[2012-09-04-15:04:15 504909] stream[1] delay offset -0.0000008532218203 s -> -341 bytes -> 57685 bytes

[2012-09-04-15:04:15 635613] T=525.1, R=525.1, D=0.0 MiB/s bsleeps=839540 dropped=0

timestamps 1346751301 342177

[2012-09-04-15:04:16 229774] stream[0] delay offset -0.0000008532172160 s -> -341 bytes -> 57685 bytes

[2012-09-04-15:04:16 229838] stream[1] delay offset -0.0000008532172160 s -> -341 bytes -> 57685 bytes

[2012-09-04-15:04:16 636699] T=976.1, R=976.1, D=0.0 MiB/s bsleeps=0 dropped=0

timestamps 1346751302 13265

At the same time, the "**CORRELATOR PROGRAM**" starts reading the data from shared memories, performs correlation process and writes the visibility data onto disk(out.txt file). We can see the following messages in that window:

gmrt_corr_get_header: ipcbuf_get_next_read()

gmrt_corr_get_header: ipcbuf_mark_cleared 2

gmrt_corr_get_header: ipcbuf_get_next_read()

gmrt_corr_get_header: ipcbuf_mark_cleared 2

Runnning correlator

bytes_per_sec = 800000000

SAMPLES_PER_WORD = 1

freq_count = 2048

time_update = 1.024e-05

timestamps 1346751300 0

C06 2012-09-04 15:05:00.000000 -0.290592581034 -341.290592587029 0.000000000007 186.17219543 3.47117496

C06 2012-09-04 15:05:00.000000 -0.290592581034 -341.290592587029 0.000000000007 186.17219543 3.47117496

timestamps 1346751300 671088

C06 2012-09-04 15:05:00.671088 -0.288728117943 -341.288728122890 0.000000000007 183.82281494 3.47081113

C06 2012-09-04 15:05:00.671088 -0.288728117943 -341.288728122890 0.000000000007 183.82281494 3.47081113

timestamps 1346751301 342177

C06 2012-09-04 15:05:01.342177 -0.286886394024 -341.286886396119 0.000000000007 181.50207520 3.47044730

C06 2012-09-04 15:05:01.342177 -0.286886394024 -341.286886396119 0.000000000007 181.50207520 3.47044730

timestamps 1346751302 13265

To stop the "CORRELATOR PROGRAM" and "PACKET CAPTURE PROGRAM" use CTRL+C in the corresponding windows.

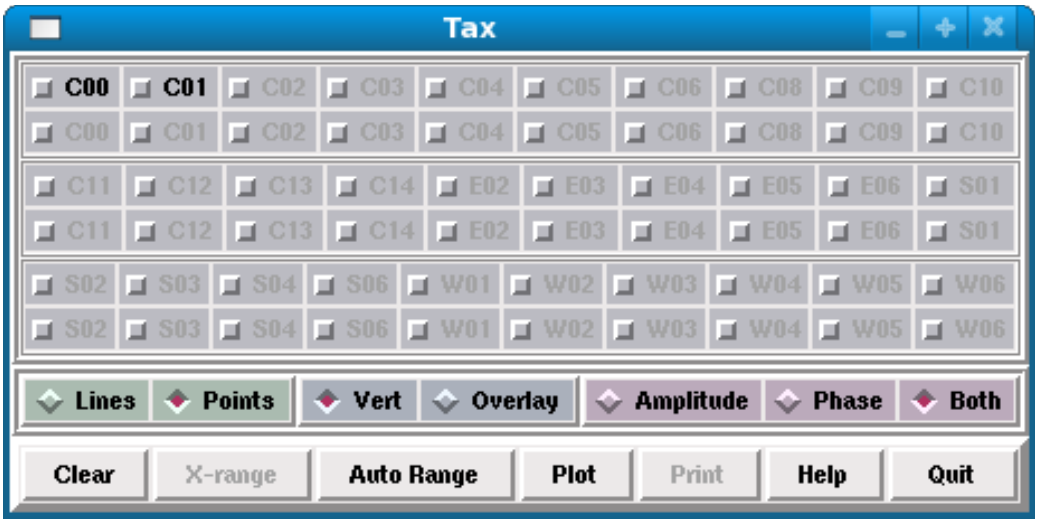
VII ANALYSING THE RESULT USING TAX PROGRAM :

The correlator program writes the results to output file "out.txt" located in [DELAY_DIR] folder. The results are analysed using the software Tax(built in GMRT). Follow the steps given

below

1. Open a new terminal and log into GPU_PC. $ ssh -X [USER_GPU_PC]@[GPU_PC]
2. Move into the directory TAX_DIR. $ cd [TAX_DIR]
3. Run the command $ ./xtrgsb32_rawsamp -v [DELAY_DIR]/out.txt -c 1,2047 -t 0

   This command will open a window which looks as below



The arguments given to this command represents

-v [filename] the output filename to which the correlation output results are written

-c [start,stop] channel selection format ***[start_chan,stop_chan,chan_step]*** default chan_step is 1

-t [start,stop] timestamp[TS] selection format ***[start_TS, stop_TS, TS_step]*** default TS_step is 1. To select only one timestamp give the corresponding timestamp number.

4. In the window opened click on 'C00','C01','Amplitude' and then click 'Plot'. A window opens. Which shows the self amplitudes of both the signals. The plot corresponding to 'C00-USB-130' is of ADC input 'i' and the plot corresponding to 'C01-USB-130' is of ADC input 'q'.

The x-axis is frequency channels and the y-axis is power. Bandwidth is 200MHZ as sampling frequency is 400MHZ.

5. Close the plot window and click 'quit' on tax window.

6. Now to plot the cross amplitude and phase use the command given below

$ ./xtrgsb32_rawsamp -v [DELAY_DIR]/out.txt -c 1,2047 -t 0 -r C00 -n 1

This command is similar to the previous command with extra arguments.

-r [antenna] this argument gives the reference antenna with which cross is to be plotted

-n 1 this agrument gives information that the cross amplitude to be normalised

After running this command a window similar to tax window opens with only 'C01' option.Click on 'C01' and click on 'Plot'. A window opens wich shows the cross amplitude(normalised) and cross phase.

7. To plot cross amplitude and cross phase of a single channel over time use the command given below

$ ./xtrgsb32_rawsamp -v [DELAY_DIR]/out.txt -c 1000 -t 0,100000 -r C00 -n 1

After running this command a window similar to tax window opens with only 'C01' option.Click on 'C01' and click on 'Plot'. A window opens wich shows the cross amplitude(normalised) and cross phase over time.