

CICADA Note 10

Documentation of 800Mhz IBOB/ADC based
Spectrometer Design

Dongliang Liu, John Ford, Glen Langston

NRAO Green Bank Jan 10, 2009

ABSTRACT

The 800Mhz IBOB/ADC based Spectrometer has been completed. This document presents an overview of the FPGA design, and give some brief descriptions [1] about configurations which allow the user to take advantage of all the features. We also present tests of the system with IF sources and explains the methods for using this design to send and receive data. My work was modifying the Nancay spectrometer, design to distribute channels to different IP addresses. A major change was adding a corner turner block.

We review features and limitations of the design and point out potential improvements in future work.

[1] The detailed configuration will be given in User's Guide.

Revision	Date	Author	Sections/Pages Affected
			Remarks
1.0	2009-Feb-10	Dongliang Liu	All
			Initial version for the project close-out

Introduction

This document describes the IBOB/ADC based, wideband spectrometer. The design “Guppi Spectrometer” (use GuppSpec for short) is based on Parkes Spectrometer and Nancay Pulsar Machine Spectrometer[1] by CASPER group and based on the IBOB hardware platform, it was built using the CASPER Simulink toolflow and DSP libraries.

The GuppSpec design is a part of Wideband Coherent De-dispersion System. The features of the spectrometer are specified by Scott Ransom and Paul Demorest, it requires to distribute 800Mhz to several PC-clusters(CPU or GPU) and according to Paul’s experience, each computer may handle up to 50Mhz spectrum with 8 bit depth, so we get the design to meet this requirement.

Based on Nancay’s spectrometer which is a dual pol, 400Mhz, 128 channels design with two 10BbE IP out, the major change here was adding a corner turner block and give the multi IP address out.

This note will give a basic description about the whole spectrometer design. Merely enough detail is given to allow the user to take advantage of all the features and configuration options that are available.

An Overview of FPGA design

1. ADC and Sync Pulse Block

Based on Nancay Pulsar Machine Spectrometer design, the GuppiSpec requires a wider band as 800Mhz, so in this mode of sampling, we set XSG core to adc0_clk still, and set the ADC sampling clock rate to 800 in the ADC yellow block and select ADC *interleave mode*. The clock physically connect to the ADC board is a single 800MHz, the interleaved clocking with phase delay will be handled internally by the ADC chip.

The output of data is 8 bits wide, we add 1 clk delay in order to meet the timing constrain during the compiling. To Randy McCullough’s suggestion, we set *outofrange* port to a LED, so that when samples are outside the valid range we may easily change the power range according to the LED’s on/off and brightness.

As Henry Chen’s sync pulse memo[2] describes, we need to setup this pulse to aid in

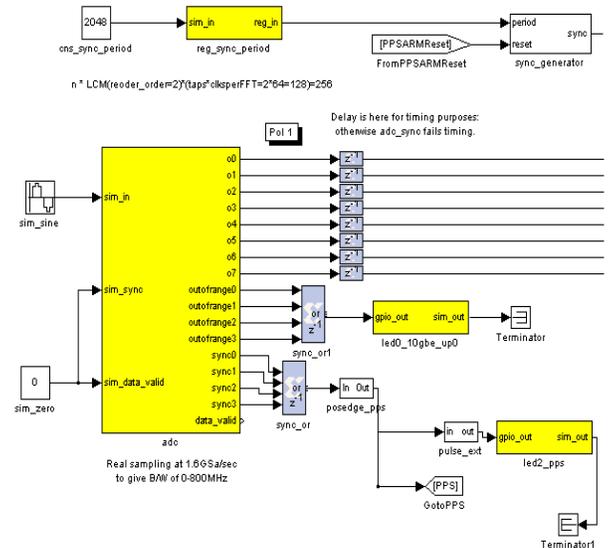


Fig 1. ADC and Sync Pulse block.

[1]. http://casper.berkeley.edu/wiki/index.php/Nancay_CoDeDi_Pulsar_Machine

[2]. http://casper.berkeley.edu/memos/sync_memo_v1.pdf

managing the data stream and get rid of some initialization period and valid data. The formula to calculate this sync period is as follows:

$$\text{Minimum Sync period} = \text{LCM}(\text{reorder orders}) * \text{FFT-Size} / \text{Simultaneous-inputs}$$

Use this formula in the design, reorder orders in FFT is 2, corner turner's reorder order is also 2 (dual buffer mode is active, else the reorder order will up to 11), FFT size is $2^8=256$, taps in PFB is 2, simultaneous-inputs is 8. So the sync period will be $n*2*2*2*256/8=256$, we set 2048 as initialization.

2. PFB-FFT, Scaling and Bit-select Block.

The FFT can be viewed as a filter and if given an input to a FFT response, the output will include subband leakage in the spectrum[1], so a PFB is added to change the response, this is the basic part for almost all spectrometer. In the design, ADC has 8 Simultaneous outputs, so we setup both PFB and FFT to this number, we have considered to use two separated PFB-FFT block, each handles 4 outputs data instead of a big PFB-FFT block to handle the whole 8 outputs, the compared result shows that the resources they cost are almost the same but the separated design have the timing constrain errors. In the 400Mhz, dual-pol design, separated design could past time constrains, so we consider this error may be the cause of ADC interleave mode setup.

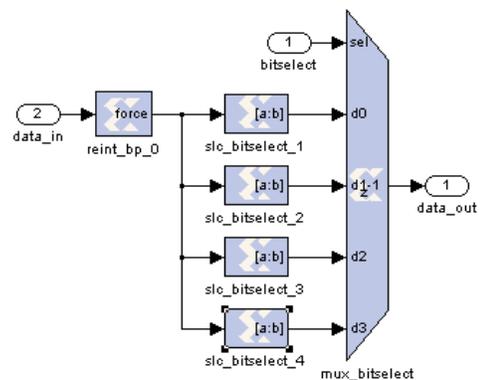


Fig 2. Bit-select Block

The ADC samples data with 8-bits of precision, but in the FPGA design, this bit width is gradually increased to 18, which is the bit width of the data coming out of FFT. However, not all 18-bits are outputted over the 10GbE connection – in the final stage, 8 out of the 18 bits are selected. This selection is user-controlled, but it is not arbitrary: the user must pick one of four bit selection options: bits 0-7, 4-11, 8-15 or 11-18.

The question about which 8 of the 18bits you select relies on several factors. The major consideration is that we typically want to select the most significant bits that are not zero (possibly with some allowance for “room” for RFI), and which collection of 8 bits these most significant.

So the *scope_output* block will help to look into the output of FFT with *bramdump* command

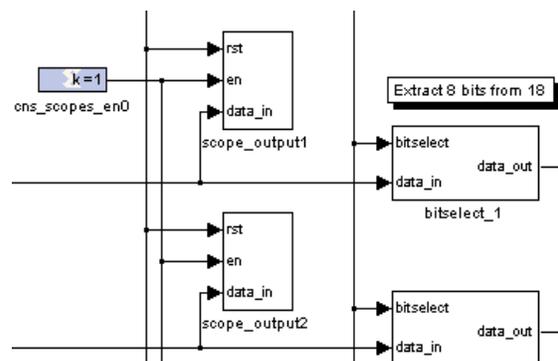


Fig 3. FFT output Scope block

[1]. <http://seti.berkeley.edu/galfa/signalproc/pfb.html>

[2]. http://casper.berkeley.edu/doc/mlib_devel_7_1/doc/html/node43.html

non-zero bits will be in depends primarily on: a. input signal power and b. the scaling parameter. in *TinyShell* of IBOB, one could setup the parameter according this scope output. This feature is not normally used but is very valuable during the development.

3. Corner Turner Block

Corner Turner block is designed to meet the output requirement of distributing the data to 16 IPs. It reorders the data flow, so that each IP could get the same spectrum. The corner turner block will buffer the data and output the same 8 channels to one IP. The data structure in the Corner Turner is as follow table. Each number in the table represent a 64 bit path, included 4 channles.

In order to get larger data packet size which John Ford suggested, we use a 4096*64 bits buffer size[2] (which may be the largest in the design with IBOB, for the larger number neither the Simulink nor the recourses could handle). In the design we setup the dual buffer parameter to 1.

	Spectrum No.	Input direction (Write direction) 					
Output Direction (Read direction) Forming Packet 	1	0 1	2 3	4 5	6 7	...	30 31
	2	32 33	34 35	36 37	38 39	...	62 63
	3	64 65	66 67	68 69	70 71	...	94 95
	
	256	4064 4065	4066 4067	...			4094 4095
10 GbeE Block Distribution							
Destination IPs		IP1	IP2	...			IP16

Table 1. Corner Turner data structure.

4. 10GbE Block

Following figure shows the design of 10GbE block. Delay the valid and data signals so

that end-of-frame goes high for the last clock that valid is high[1]. This is a requirement for the 10GbE block to work. So we use *tx_valid* and *tx_end_of_frame* port to control the two 10GbE blocks work alternately.

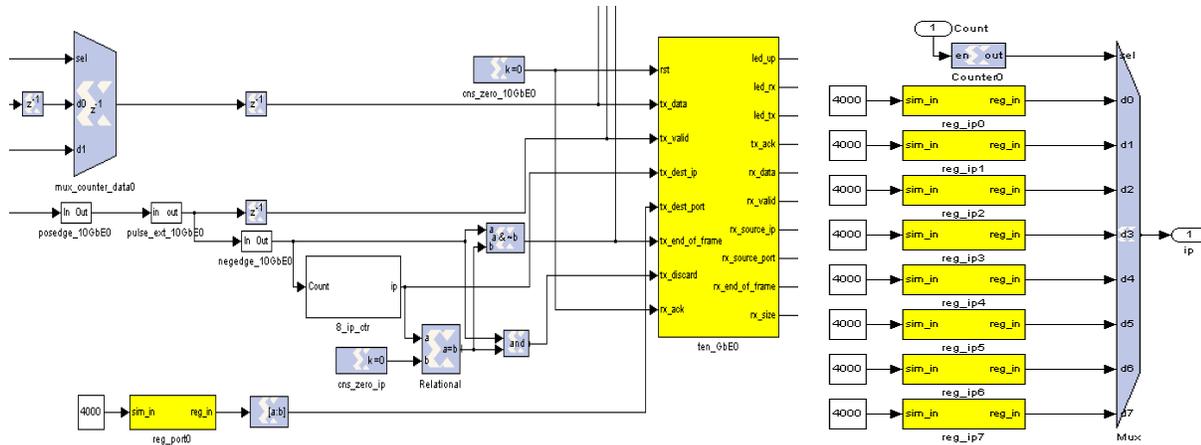


Fig 4. Logic for 10EbE and 8 IP distributor

In the mean time, to cooperate with Corner Turner block to generate the correct packet size, we set a 9-bit counter and every half time one 10GbE block will work to buffer the data and prepare the UDP packet. So every 256 clk we sent a different destination IP to *tx_dest_ip* port with *tx_valid* on high and *tx_end_of_frame* on high at the last clock before *tx_valid* on low. The *tx_dest_port* will be set as UDP port and other port to leave default.

For the packet format, the spectrometer outputs UDP packets whose payloads have the following structure:

Counter							
P ₀ (0)	P ₀ (1)	P ₀ (2)	P ₀ (3)	P ₀ (4)	P ₀ (5)	P ₀ (6)	P ₀ (7)
...							
P ₂₅₆ (0)	P ₂₅₆ (1)	P ₂₅₆ (2)	P ₂₅₆ (3)	P ₂₅₆ (4)	P ₂₅₆ (5)	P ₂₅₆ (6)	P ₂₅₆ (7)

Table 2 Data structure of the packet.

A single packet contains 8-channel spectrum. With the 9-bits wide counter, and all the remaining (data) entries are 8-bits wide. Thus the total size of a single packet payload is 2056 bytes. P_x(y) is the Voltage of y bin/channel of x spectrum.

5. Timing using ARM and 1PPS

In order to get precise time-stamp of each spectrum, we cited this part of design from Nancay Pulsar Machine Spectrometer, the idea here is that it provides a means to reset the counter at a precisely known time, and add the counter number into the packet of the spectrum, so this enables the user to determine the time a spectrum arrived very accurately.

[1]. http://casper.berkeley.edu/doc/mlib_devel_7_1/doc/html/node53.html

To let this block work, we need the control computer (the one connected to the IBOB via the 100MbitE port) to set up to use NTP (*Network Time Protocol* [1]), so that its clock is accurate to within a few tens of milliseconds. Then toggle the *reg_arm* register to make IBOB synchronized.[2]

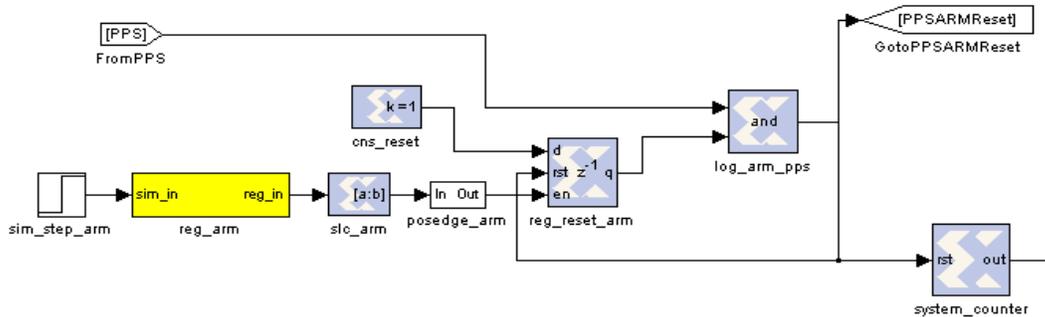


Fig 5. ARM and 1PPS Block

6. Software configuration and Receiving 10GbE/1GbE Packets on the Data Recorder Computer

6.1 IBOB Configuration

There are two main categories of system set up: 1). spectrometer data, 2). connectivity settings. All configuration of the design is done in TinyShell as a simple telnet terminal.

In spectrometer data setting, we need to set Sync Pulse initialization period, FFT bit shift and bit selection.

In connectivity settings, we need to set the sending IP address and port of the 10GbE interface/connection on the IBOB, and the destination IP address and port. We also need to inform the IBOB of the MAC addresses of both the sending and receiving interfaces.

For the detailed configuration, please look into the “User Guide-A 800MHz 128-channel,16 IPs distributed Spectrometer”.

6.2 Receiving 10GbE/1GbE Packets on the Data Recorder Computer

After connecting both the IBOB and PC to the Switch, we use Paul's *udp_recv* program to get the test data from IBOB, and use a Matlab script to transform the binary data into decimal.

[1]. http://en.wikipedia.org/wiki/Network_Time_Protocol. or here in NRAO we use Lazier clock.

[2]. for more information, please visit <http://casper.berkeley.edu/wiki/index.php/Parspec>.

The result can be seen in the following test. Another choice is to capture data with gulp [1] (a network capture program that stores packets in pcap format which Nancay Spectrometer used) and process it.



```
cicada2.gb.nrao.edu - PuTTY
Last login: Wed Feb 11 14:57:55 2009 from cicada2.gb.nrao.edu

*****
National Radio Astronomy Observatory
Authorized use only

NRAO Gold Book at http://www.nrao.edu/gold
*****

[dliu@east ~]$ ~pdemores/bin/udp_rcv -p 5001 -s 2056 -d ./dong_test31 -n 50 192
.168.3.10
sizeof(struct in_addr) = 4
ipaddr=192.168.3.10
Receiving data (packet size=2056) .
Receiving from 192.168.3.10
Packet size 2056 B
Total time 0.060 s
Got 0.1 MB (50 packets)
Recv rate 1.634 MB/s
Send rate 527766798519500800.000 MB/s
Dropped -50 packets
Drop rate -3.096e-18
Avg load 0.000
[dliu@east ~]$
```

Fig 6. Use udp_rcv to get the UDP packets from IBOB.

Limitations of the Design and Future Work

Due to the restriction of numbers be entered in the blank of Simulink and the structure of reorder block, 4096 numbers is the largest one which this design can handle, this limit the packet size of data. We are considering two paths to generate larger packet:

1. Looking into the reorder block, if we could enter the order sequence number directly into the Rom instead of using the Simulink Mask, we may have got larger number.
2. To delay the *tx_valid* and *tx_end_of_frame* signal of 10Gbe block which meet the largest buffer size of this block, so we may send 16 channels means 100Mhz of spectrum or more to one computer. This will need a new reorder sequence and related time delay, but worth trying.

We have 4 computers with graphic card in the Lab. So we will configure all 16 IPs to 4 IPs, this implementation will add the complexity of the receiving program such as re-arrange the order of spectrum.

We will work on this program, not only the receive part but also with graphic card FFT program in CUDA. To get the whole 800Mhz Pulsar Coherent de-dispersion system work.

[1]. <http://staff.washington.edu/corey/gulp>

A Simple Tune Test

Generate first 8 channels (0~50Mhz bandwidth with/without 26Mhz pulse in) data from IBOB to EAST. 2056 Bytes/packet with 20 packets, select 16384 (8 bit depth for both real and image) data out, calculate the power of the spectrum and plot in Matlab. The second row of graphics show the 4 packets data, 2056*4=8224 points. The pulse may be caused by the wrong bit-selection in certain part of signal or DC power.

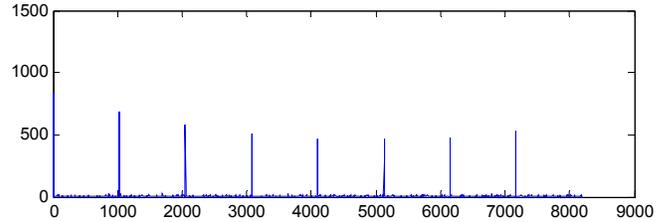
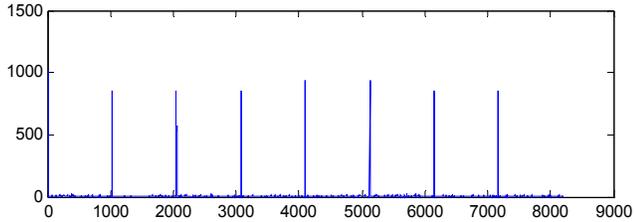
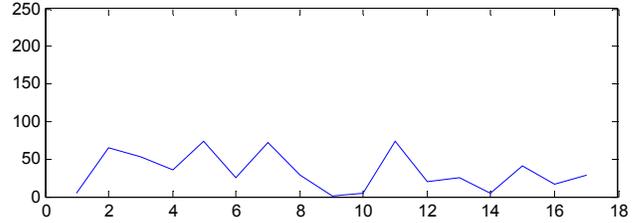
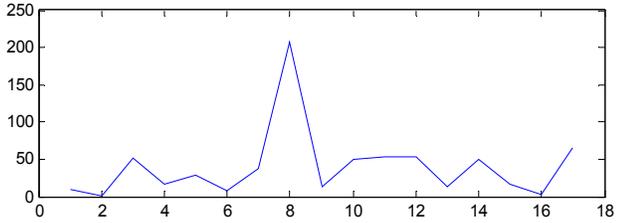


Figure1. With 26Mhz Pulse

Figure2. Without 26Mhz Pulse

%%%%%%%%%

Matlab test code

%%%%%%%%%

clear;

fid = fopen('dong_test');

c= fread(fid,32768);

e=c (16384:32768);

for i=1:8192

f(i) = sqrt(e(2*i-1)^2+e(2*i)^2);

end

for n=1:16

for m=1:512

g(n) = sum(f((m-1)*16+n))/512;

end

end

subplot(211),plot(f/16)

subplot(212),plot(g)

%%%%%%%%%

Appendix A: Design Summary

Design Summary	
Frequency channels:	128 (256 real samples per spectrum)
Signal input:	5MHz -800MHz or 800MHz -1.6GHz (2nd Nyquist zone) or 1.6GHz -2.4GHz (3rd Nyquist zone) -20dBm to -10dBm (-15dBm nominal) 50Ω SMA
Polyphase filter:	2 taps, Hamming window
Output:	Test mode: 100Mbit Ethernet. 32-bits per spectral bin. Observing mode: 10Gbit Ethernet. 8-bits per spectral bin.
Clock input:	800MHz, 0dBm to +4dBm, 50Ω SMA
1PPS input:	0 to 3V pulse nominal (into 50) 2V minimum, 5V maximum. Optional.
Power input:	5V, 7A
Mechanical:	1x IBOB and 1x iADC board on a 6U, 8HP plate.
Control and monitor:	Set up sync period. Set up IP addresses, ports, MAC addresses, and ARP table. Set scaling: 18-bits, binary point at 12. Set output bit selection. Set ARM (optional).