VEGAS Pulsar Mode Integration – Initial Specifications

Richard Prestage

V4: 7th March 2014

Abstract

This document provides an initial overview of the work which will be required to implement pulsar modes in the VEGAS spectrometer.

Contents

1. Introduction

This document provides an initial overview of the work which will need to be done to implement pulsar modes in the VEGAS spectrometer. Naively, this work consists of porting the "DIBAS pulsar modes" to VEGAS. However, there are additional activities (e.g. implementing additional pulsar modes desired but not delivered with DIBAS).

The purpose of this document is to attempt to summarize ***all*** of the project work required to support current and medium-term planned spectrometer and pulsar modes in VEGAS. The intent of that project is to deliver a well-documented, easily maintainable instrument which has all the facilities and ancillary capabilities (e.g. status displays) necessary to become the workhorse GBT pulsar/spectral line backend.

This document simply provides an initial outline of the work that will be required. Most areas described here will need more careful and detailed requirements development before implementation can begin.

2. Project Scope

The following items are in scope:

- Development of additional FPGA personalities
- Rationalization of the HPC software
- Config_tool and Astrid
- CLEO screens and status displays
- The Proposal Submission Tool
- Documentation
- Commissioning of new (mainly pulsar) modes in VEGAS
- Running old GUPPI and new VEGAS in parallel

The following items are considered out of scope, and are not covered in this document:

- Work required to deliver potential "expert user" modes for either spectral line or pulsar observing. An example is being able to configure different banks to have different integration times.
- Commissioning of the already-delivered VEGAS spectral line modes (an ongoing activity)
- Configuring new VEGAS to have some banks set up for pulsar observing, and some banks set up for spectral line observing.
- Real-time spectral line displays
- Data streaming, data reduction pipelines and data archiving.

3. Hardware / Firmware

The desired pulsar modes for VEGAS are shown in Table 1. These comprise existing DIBAS modes which can be used unchanged, some which should simply require to be recompiled for a different FPGA clock frequency, and some which will require additional development.

Incoherent Modes					
Chans \ BW	100	200	400	800	1500
32	Development	Development	Development	Development	Development
64	Development	Development	Rebuild	Tested	Tested
128	Development	Development	Rebuild	Tested	Tested
256	Development	Development	Rebuild	Tested	Tested
512	Development	Development	Rebuild	Tested	Tested
1024	Development	Development	Rebuild	Tested	Tested
2048	Development	Development	Rebuild	Tested	Tested
4096	Development	Development	Rebuild	Tested	Tested
8192	Development	Development	Rebuild	Tested	Tested
Fast4k	Development	Development	Rebuild	Tested	Tested
Fast8k	Development	Development	Development	Development	Development
Raw Time Domain	Development	Development	Development	N/A	N/A
Coherent Modes					
Chans \ BW	100	200	400	800	1500
32	Development	Development	Development	Development	Development
64	Development	Development	Rebuild	Tested	Tested (1)
128	Development	Development	Rebuild	Tested	Tested (1)
256	Development	Development	Rebuild	Tested	Tested (1)
512	Development	Development	Rebuild	Tested	Tested (1)
1024	Development	Development	Rebuild	Tested	Tested (1)
2048	Development	Development	Rebuild	Tested	Tested (1)
4096	Development	Development	Rebuild	Tested	Tested (1)
8192	Development	Development	Rebuild	Tested	Tested (1)

Table 1: Desired pulsar modes for VEGAS

Note 1: FPGA personality tested, but CPU cannot keep up with the data rate.

3.1.Existing "DIBAS" Pulsar Modes

The following, existing pulsar modes should be ported to VEGAS:

- Incoherent Modes, 800 MHz bandwidth:
  - 64, 128, 256, 1024, 2048, 4096, 8192 channels
- Coherent Modes, 800 MHz bandwidth:
  - 64, 128, 256, 1024, 2048, 4096, 8192 channels
- Fast4K mode that is "I-only" for fast searching
  - Does any work need to be done for this mode?

3.2.New Pulsar Modes

The following, new pulsar modes need to be developed for VEGAS. Many of them are simply variants of the existing modes, but some development is required.

- 1.5GHz bandwidth
    - The FPGA personalities can be run at bandwidths up to 2 GHz.
    - In 1.5 GHz coherent mode, the CPUs cannot keep up.
- 400 MHz bandwidth, incoherent and coherent.
    - We believe the designs can be simply rebuilt for 400 MHz. They will need to be retested.
- 200 and 100 MHz bandwidth, incoherent and coherent.
    - The Valon Synthesizer will not go below 137.5 MHz, and the waveform looks unsatisfactory at 200 MHz. So, these bandwidths will have to be implemented by running the clock at 400 MHz, and decimating the ADC samples into the DSP chain. The FPGA clock would still be running at a fast rate, but we would downsample the datastream thus reducing the bandwidth. Implementing decimation would require fairly substantial engineering time.
- 32 channel modes
    - These modes are low priority
    - Incoherent and coherent
    - Same choice of bandwidths as for e.g. 64 channels
    - These are problematic in two regards: First the CASPER FFT block can't be configured for less than 64 channels since we're driving 16 inputs into it. Second, in the CoDD modes, less than 64 channels becomes a data rate problem in that there aren't enough clock cycles available during data readout operations to keep up with the data production rates from the ADCs.
        - We think this can also be cured by decimating the samples. This will allow us to use a smaller number of inputs to the PFB & FFT blocks which solves the problem of not being able to configure the block for less than 64 channels.
        - Some thought does need to be given for the CoDD mode problem (number of clock cycles available during readout operations) to make sure decimation relieves the problem.
- Fast4k mode.
    - This has been developed /tested on the FPGA side at 800 MHz and up for DIBAS. This mode is actually only relevant at low bandwidth.
- Raw time domain mode. Exists for GUPPI but not officially supported.
    - This is a mode that does not have a DSP chain, so raw ADC samples are passed through, arranged, and transmitted out the 10GbE ports. This has been sparsely used in the past, but has been used a little bit more as of late. Implementing this would also require some engineering time on the firmware, and software modifications.
    - There was a bug in the current GUPPI implementation (data from nodes 1-4 is replicated on nodes 5-8), this has recently been fixed.

    - Only required at 100 MHz bandwidth and above (max bandwidth will be constrained by disk write speed). Narrower bandwidths are provided by the spectrometer modes.
- Ultra-wideband Mode.
    - The requirement is 2x1500 MHz (or at least, 2x 1000 GHz). The main problem is the GPU machines in coherent mode. The limitation is in memory before the data gets into the GPUs. Joe Brandt has done some experimentation moving the pre and post dispersion transposes onto the GPU the result was a significant reduction in the net thread load. Preliminary tests indicated GUPPI could handle around 1000 MHZ reliably with the updated code.
    - We note it would be possible to add the GUPPI GPU nodes (with updated GPUs) to the VEGAS GPUs.

3.3.IF System / Analog Modules / Digital Filters

The IF modules in VEGAS have bandwidths of 950, 1150 and 1400 MHz, and are connected directly to the Converter Rack. For GUPPI, the narrower bandpass filters are provided by the Analog Filter Rack. We would therefore have two options: connect VEGAS via the Analog Filter Rack, or sample at 3GSPS, and then filter digitally. One possibility would be 1600 MHz converted to 400, 200 or 100 MHz. This approach would allow us to decommission the analog filter rack once VEGAS became the primary backend. This approach would need further prototyping.

4. Software

4.1.VEGAS Coordinator / Managers

This comprises adding extensions to the Coordinator / Managers to implement the pulsar (and other) modes. The use of a “strategy pattern” implementing various types of backend will be retained. A number of new backends will be required:

- GuppiBackend (or perhaps better, IncoherentBackend) for all incoherent pulsar modes
- GuppiCODDBackend (or perhaps better, CoherentBackend) for all coherent pulsar modes
- XXX Backend for the raw timing mode?
- YYY Backend for the fast4k mode? May not be necessary since this is nearly identical to the incoherent modes.

4.2.Config_Tool / Astrid

The config_tool framework will need to be extended to include the pulsar modes. We propose that this would be indicated by selecting:

- Backend = VEGAS
- Obstype = Pulsar

Naively, GUPPI specific keywords would be moved to the VEGAS class in the config_tool. This would force users to alter their GUPPI scripts slightly, but would clearly delineate the use of the VEGAS backend versus the GUPPI backend.

We will need to be able to configure Astrid for simultaneous GUPPI / VEGAS observations. This mode will need to exist for 6-12 months, so should be implemented cleanly rather than as a "hack". We will need to think about the high-level config_tool keywords – for example we will need to have a way to configure GUPPI to be 800 MHz and VEGAS to be 1500 MHz.

4.3. High Performance Computer Software

Joe Brandt's "VEGAS Pulsar Mode Integration" document, dated February 21, 2014, provides more details of the HPC software. We would like to refactor this code for a variety of reasons. One is that the VEGAS and GUPPI code bases share a substantial amount of code, with many symbols differing only by name. A second is to separate the application-specific processing stages from the general infrastructure (e.g. thread management).

The "GUPPI_DAQ" code is now in use in a number of places / machines; for example PUPPI, WUPPI, Nancay, SHAO, PAPER, VLA pulsar work. Ideally, we would do the code refactoring in such a way as to make this useful for other groups. Related work is being done by Willem van Straten (Swinburne), David McMahon (CASPER) and the MeerKAT group (SKA SA). We should coordinate with these individuals/groups before tackling this work.

Note that to some extent, the work which will be performed in this area will depend upon how much effort is made available.

4.4. Real Time Displays

Paul notes "the existing uppi/dibas displays get the job done, but are rather clunky". We might want to have some extra discussions and come up with a good plan for this from the start. I'm not sure explicitly which displays he is referring to, but presumably one of the following. Need to expand what is required here.

4.4.1. Shared Memory Status Display

4.4.2. GBT Status Display

4.4.3. VEGASDM Data Monitor

4.5. CLEO Application

5. Testing and Commissioning

5.1. Short – Medium Term Commissioning

Paul notes that testing the pulsar modes could/should involve a fairly large amount of work. We should expect to spend at least a couple of months on it, doing things like long overnight runs on test signals in the various modes. See https://staff.nrao.edu/wiki/bin/view/PMD/ShaoBackendPulsarTesting for more details.

5.2.Long-term Commissioning

We will likely want to do ~6-12 months of real astronomical observations using both GUPPI and VEGAS simultaneously for certain projects. This is to ensure continuity for the sensitive long-term pulsar monitoring projects.

6. Proposal Submission Tool

We need to update the proposal submission tool to handle all of the new modes. We need to fix a bug in the PST where the proposer is allowed to change the bandwidth.

7. Documentation

- Update the Proposers' guide
- Update the Observers' guide
- Update the trouble-shooting page

8. Other Issues

8.1. I/O Performance

One issue to think about is the capability of the lustre filesystem to handle the data rates for the various modes. As things stood ~ 6 months ago, it maxed out around 150 MB/s, which is not enough to replicate the existing GUPPI search modes.

We understand that writing to the same file with multiple threads is supposed to be much faster than with a single writer. We'll need to do some testing, and then decide how to modify the HPC code.

8.2.CPU Performance

Under some circumstances the network thread will saturate. Doing some proling it was found that the transpose performed in the net thread was a major load contributor. This pre-transpose and the post dispersion transpose were moved onto the GPU. The result was a signicant reduction in the net thread load. Preliminary tests indicated GUPPI could handle around 1000MHz reliably with the updated code. (This version of guppi daq is currently in a unreleased 'performance' branch.)

8.3.Register Name Conventions

Although a sort of 'water under the bridge' point is the various register naming conventions used throughout GUPPI and VEGAS. Even for blocks which seem to be common to various BOF's, the exposed registers have inconsistent names. As an example, for the x BOF, the register for network configuration is "tGv20", whereas the name for coherent BOFs is "tGX8 tGv2". VEGAS BOF files use "ge0".   Both

VEGAS and GUPPI BOF's have a hardware 'accumulation' register. In VEGAS the register is acc len, in GUPPI it is 'ACC LEN'. The arming strategies are also quite different. The VEGAS BOF's require a number of register writes, delays, and more writes for the arming sequence. In contrast, GUPPI BOF's only require a single register write. The same ADC block is used for both spectral line and pulsar BOF's, however, the registers which expose signal levels are not named the same. This impedes the direct use of vegasdm for pulsar modes

9. Other Items

Anything else we want now, while we are at it?