Journal of Astronomical Instrumentation, Vol. 6, No. 1 (2017) 1641011 (16 pages)
© World Scientific Publishing Company
DOI: 10.1142/S2251171716410117

A Wideband Digital Back-End for the Upgraded GMRT

Suda Harshavardhan Reddy[1,3], Sanjay Kudale[1], Upendra Gokhale[1],
Irappa Halagalli[1], Nilesh Raskar[1], Kishalay De[2], Shelton Gnanaraj[1],
Ajith Kumar B[1] and Yashwant Gupta[1]

[1]Giant Metrewave Radio Telescope, NCRA, TIFR, Pune, India

[2]Indian Institutute of Science, Bangalore, India
[3]reddysh@gmrt.ncra.tifr.res.in

Received 2016 August 19; Accepted 2017 February 3; Published 2017 March 13

Traditionally, back-ends for radio telescopes have been built using a hardware-based approach with ASICs, FPGAs, etc. With advancements in processing power of CPUs, software-based systems have emerged as an alternative option, which has received additional impetus with the advent of GPU-based computing. We present here the design of a hybrid system combining the best of FPGAs, CPUs and GPUs, to implement a next generation back-end for the upgraded GMRT. This back-end can process 400 MHz bandwidth signals from 32 dual-polarized antennas, for both interferometry and beamformer applications, including narrowband spectral line modes for the interferometer, incoherent array and phased array mode of operations for the beamfomer, and also a voltage mode attached to a real-time coherent dedispersion system for the beamformer. We describe in detail the design and architecture of this system, including the novel features and capabilities. We also present sample results from the system that validate its performance in conjunction with the entire receiver chain of the upgraded GMRT.

Keywords: Radio telescope, correlator, beamformer, high performance computing, GPU processing.

1. Introduction

Radio interferometry has revolutionized radio astronomy in the last few decades. The simultaneous drive towards arrays with larger bandwidths and larger number of elements (large N) to increase resolution and sensitivity has led to a manifold increase in the computational requirements for the back-end systems of these telescopes, which carry out the required computations in real time. There are several large interferometers in operation today, such as Very Large Array (VLA; Perley *et al.*, 2011) in USA, the Giant Metrewave Radio Telescope (GMRT; Swarup *et al.*, 1997) in India, the precision Array for Probing the Epoch of Reionization (PAPER; Parsons *et al.*, 2010) in South Africa, and the Long Wavelength Array (LWA; Taylor *et al.*, 2012) in USA.

Historically, interferometer back-ends have been hardware based, using dedicated Application Specific Integrated Circuits (ASICs) and Field Programmable Gate Arrays (FPGAs) (e.g. Perley *et al.*, 2009). Although, these designs are superior in terms of their computational efficiency (per unit of power consumption) compared to software-based back-ends, the rapid growth and increasing availability of high performance computing systems in recent years has allowed a gradual shift towards software-based systems as a viable alternative (e.g. Roy *et al.*, 2010; Deller *et al.*, 2007). These systems have proved useful for a number of reasons, including easier availability of off-the-shelf compute infrastructure, relatively smaller development times compared to dedicated hardware back-ends, higher flexibility and easy upgradability.

Graphics Processing Units (GPUs) have ushered in a new era for radio astronomy digital back-ends, with these massively parallelized, compute-intensive systems proving highly suited for the development of large N, large bandwidth correlators (see for example, Clark *et al.*, 2012). Additionally, hybrid correlators, which divide the FX (or XF; Thompson *et al.*, 2001) correlation process onto different types of compute engines have proved remarkably useful for such instruments (e.g. the Atacama Large Millimeter Array correlator; Escoffier *et al.*, 2007). The Long Wavelength Array (LWA) station at the Owens Valley Radio observatory demonstrated the first use of a hybrid FPGA-GPU-based correlator design capable of supporting a 58 MHz instantaneous bandwidth from 512 input signals (Kocz *et al.*, 2014, 2015), for the Large Aperture Experiment to Detect the Dark Ages (LEDA). Following the FPGA-based modular architecture correlator developed for PAPER by Parsons *et al.* (2008), Kocz *et al.* (2014) demonstrated how such hybrid systems can be used to maximize efficiency by capitalizing on the best of both these architectures for implementation of real-time large bandwidth back-ends. Similarly, Denman *et al.* (2015) presented the implementation of a hybrid FPGA-GPU-based correlator capable of cross-correlating 256 (128 dual polarization) inputs across a bandwidth of 400 MHz, for the Canadian Hydrogen Intensity Mapping Experiment (CHIME) Pathfinder radio telescope.

In this paper, we describe the design of such a hybrid system developed for the upgraded GMRT, capable of supporting 400 MHz instantaneous bandwidth from 64 input signals (dual polarization inputs from 32 antennas) for both interferometry and beamformer modes, and present results from a currently deployed 16-antenna version of this system. Section 2 motivates the need for the new back-end, in the context of the upgrade of the GMRT; Sec. 3 presents top level design of the new back-end; Sec. 4 explains data flow and implementation details; Sec. 5 gives performance results of the prototype; Sec. 6 shows sample results that validate the working of the design; and Sec. 7 gives details about planned future enhancements.

2. Motivation for a Next Generation Wideband Back-End for the upgraded GMRT (uGMRT)

2.1. *Overview of the present GMRT and plans for the upgrade*

The present GMRT consists of an array of 30 antennas, each of 45 m diameter, spread over a region of 25 km extent, and operating at five different wave bands from 150 to 1450 MHz. The maximum instantaneous operating bandwidth at any frequency band for the existing legacy system is 32 MHz. Each antenna provides signals in two orthogonal polarizations, which are processed through a heterodyne receiver chain to intermediate frequencies and transmitted to the central receiver building through optical fiber cables, where they are converted to baseband signals and fed to the digital back-end consisting of correlator and pulsar receiver. The main processing stages in such a digital back-end, are digitization (using an ADC), coarse delay correction, conversion to multi-channel spectra (using FFT or polyphase filter bank), phase correction of the spectra to compensate the fine delay and fringe, channel wise Multiplication and Accumulation (MAC) of spectra of each antenna with every other antenna, as well as incoherent (intensity) or coherent (phased voltage) addition of signals from selected antennas to form a high time resolution beam output (Roy *et al.*, 2010; Gupta *et al.*, 2000). For the present GMRT, the digital back-end, called the GMRT Software Back-end (GSB), is a software-based system, designed using off-the-shelf components — PCI-based ADC cards and a Linux cluster of 48 nodes with dual gigabit inter-node connectivity, hooked together by a highly compute optimized pipeline that uses cache-efficient, multi-threaded parallel code with vectorized processing, to implement a fully real-time correlator and beamformer for 32 antennas, dual polarized signals for 32 MHz bandwidth (Roy *et al.*, 2010).

A major upgrade of the GMRT is currently underway, with the aim to provide near-seamless frequency coverage over 120 to 1500 MHz with improved feeds and receivers, along with a maximum instantaneous bandwidth of 400 MHz (Gupta, 2014). The four sets of feeds and receivers that cover this range, have the RF signals transmitted over optical fiber to the central receiver building, where they are digitized and processed in the new back-end system, called the GMRT Wideband Back-end (GWB). The increase in the maximum processing bandwidth by more than 10 times, as well as additional specifications that the GWB is required to meet (described below), call for a radical new design to handle the increased computing and I/O requirements, as well as the new features and facilities.

2.2. *Target specifications for the new GMRT wideband back-end*

The main specifications for the GWB are as given in Table 1. In addition to increase in the maximum instantaneous bandwidth by a factor of 10, the other major improvements compared to the legacy GMRT system are the larger number of spectral channels (from 256–512 to 2048–16,384), the increase in the number of beams (from two to four), options for real-time RFI filtering, Walsh demodulation capability, and a real-time coherent dedispersion system that supports the much larger bandwidth.

3. Design of the GWB

3.1. *Overview of the design*

Based on our learnings from the design of the GSB, we have opted for a hybrid design that combines a FPGA based front-end with a GPU-based software compute work horse, an Infiniband-based network and optimized software, to achieve the target compute and I/O requirements. The FPGA boards, connected to high speed ADCs, perform the digitization and packetization of the voltage signals from all the antennas, while off-the-shelf compute node servers fitted with GPUs acquire the data, compute visibility products and beamformer outputs, which are then recorded on hard disks for post-processing and analysis. The implementation is a time-slicing model, where each compute node gets a slice of contiguous time series data from all the antennas, on which it performs spectral conversion, correlation and Incoherent Array (IA) and/or Phased Array (PA) beamformation operations. For narrowband mode observations, the data is digitally down-converted and decimated before the spectral conversion stage. Dedicated 10 Gigabit Ethernet links are used for transferring digitized data from the FPGAs to the compute nodes, while a 40 Gbps Infiniband network is used for both redistribution of data for time slicing, and gathering of results onto the host nodes for final recording. Figure 1 shows the top level design of the GWB.

Table 1. Main specifications for the wideband backend for the upgraded GMRT.

Number of antennas	32
Instantaneous bandwidth	100, 200 or 400 MHz
Bits per sample	8 for 100 MHz, 4 for 400 MHz
Number of spectral channels	2048–16384
Number of input polarizations	2
Full Stokes capability	Yes
Minimum visibility integration time	128 μs
Coarse and fine delay tracking range	$+/-128\,\mu$s
Fringe rotation range	up to 5 Hz
Number of sub-arrays	4
Number of beams	4 (Incoherent array/ Phased array)

The time slicing model used here has several features that makes it attractive for a software/hybrid design, as compared to a frequency slicing model (where the F stage is usually implemented on the FPGA). The bulk time delay correction for different antenna signals has to be carried out before the F stage. This is much more conveniently done for the time slicing model, especially for longer baselines (few kilometers) at high sampling rates, as the buffering of digitized data for delay correction is performed on CPU memory, which is generally a few gigabytes, compared to much smaller memory size on the FPGA. In the frequency slicing model, an explicit corner turner (see e.g. Urry *et al.*, 2007) block is required which converts the streams of data having antennas with all spectral channels into streams of data having all antennas for a subset of spectral channels; in the time slicing model, this is implicitly implemented by a swapping of array indices in software, while efficiently utilizing the large RAM of the CPU. Furthermore, for the frequency slicing done on the FPGA, the I/O rate out of the FPGA to the compute node is significantly higher than for the time slicing model (or the number of bits for representing the spectral data have to reduced significantly, which can have adverse consequences). Keeping the F and X stages together on the GPU (as done in our time slicing model) allows for optimal, tightly coupled code, and at the same time, on the FPGA it greatly simplifies the design and frees up precious resources which we have utilized for implementing signal quality enhancing features like RFI filtering and Walsh demodulation. The two main disadvantages of the time slicing approach are that (a) the total I/O rate in and out of the compute node is higher (this is generally not a serious issue as the input from the FPGA is *via* dedicated connections whereas the time slice sharing is over the network switch) and (b) the design requires larger amounts of memory, on both CPU

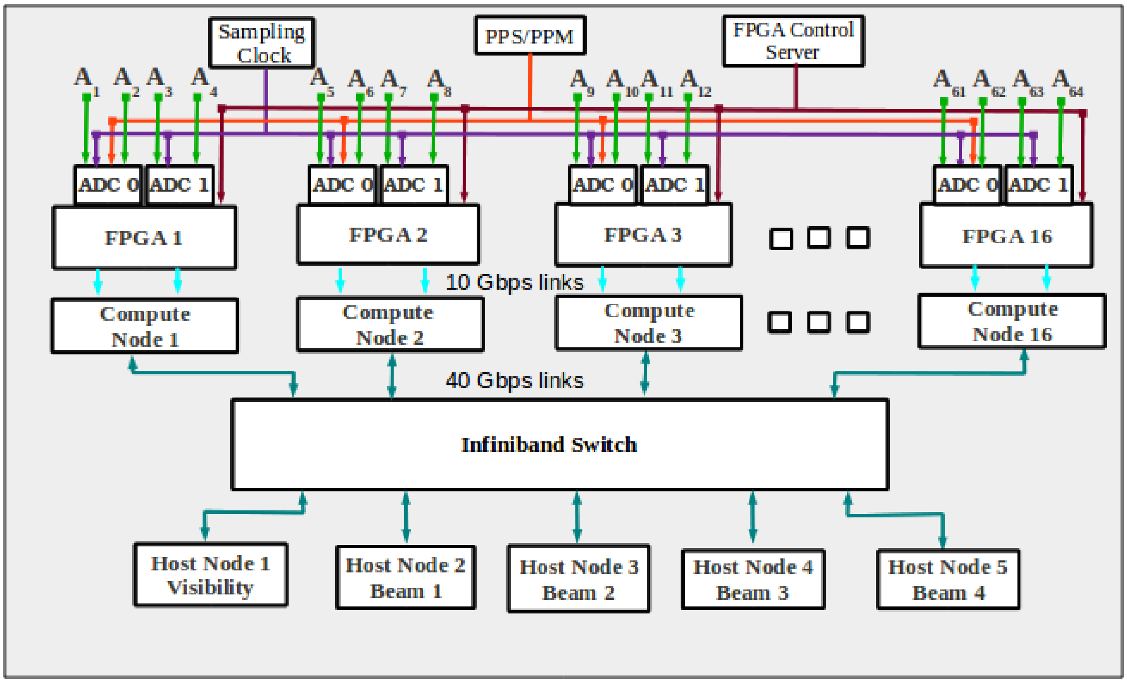

Fig. 1. Illustrating top level design of the GWB A_1, A_2, $A_3, \ldots, A_{64}$ are baseband converted input signals from antenna.

and GPU, as each compute node has to hold the visibility data for all baselines — this can become a restricting factor for scaling the design to very large number of antennas (for a mid-sized array like the GMRT, this is not an issue with the current technology options available).

3.2. *Computation and input-output data rate requirements*

The general considerations for the computational and input-output (I/O) requirements for a correlator and beamformer for the GMRT are discussed in detail in Roy *et al.* (2010), which we summarize here. Assuming N_a antennas, N_p polarizations, N spectral channels, N_b beams, N_s stokes and Δv bandwidth, the computation loads for the correlator are $2\Delta v N_a N_p \log_2 N$ cops for FFT and $\Delta v N_s N_a$ $(N_a{+}1)$ cops for MAC where cops stand for complex operations per second. The computation load for phase shifting is much less compared to FFT and MAC stages and is given by $\Delta v N_a N_p$ cops. The beamformer costs can be calculated as $\Delta v N_a N_p$ (multiplication) cops plus $\Delta v N_a N_p$ (addition) cops for the IA beamformer and $\Delta v N_a N_b N_s$ (multiplication) cops plus $\Delta v(N_a + N_s) N_p N_b$ (addition) cops for the PA beamformer.

For the GWB, going by the target specifications, the computation requirements come to 2.9 Tflops for FFT (for $N = 16,384$), 6.8 Tflops for MAC and 0.1 Tflops for phase shifting operation. For beamformer, assuming two IA beams and two PA beams, the compute load comes out to be 0.45 Tflops. Overall, the computation requirement for the GWB is around 10.2 Tflops.

The maximum overall input data rate that is to be handled is 25.6 GB/s considering 32 antennas (rounding the number of antennas to nearest power of two), 400 MHz bandwidth and 4 bits per digitized sample (or 200 MHz bandwidth, with 8 bits per sample). The output visibility data rate is about 51 MB/s for 2048 spectral channels, 0.67 s integration time and floating point storage data type. The raw output beam data rate is 209 MB/s for IA or PA beamformer considering 2048 spectral channels, integration time of 20 μs and short integer storage data type.

4. Implementation of the GWB Design

The design described above has been implemented, tested and released for a 16-antenna upgraded GMRT system. An expanded version for 32 antennas is under installation, and will be available for users by end of 2017. The design uses Xilinx Virtex-5 FPGA-based Reconfigurable Open Architecture Computing Hardware (ROACH) boards developed by the CASPER group (see https://capser.berkeley.edu for details), 8-bit ATMEL/e2v-based iADC developed by the CASPER group, a combination of DELL T630/T620 servers as compute and host nodes and Nvidia Tesla K20c GPUs. Myricom 10GbE CX4 NIC cards and cables are used for data transfer between FPGAs and compute nodes. A Mellanox QDR Infiniband switch along with Mellanox Infiniband NICs are used for data redistribution between the compute nodes and for gathering visibility and beam data onto the host nodes.

Each iADC can sample two input signals and each ROACH board has provision for handling two iADCs or digitized data from four input signals. The data from each iADC i.e. for two input signals, is bundled into a UDP packet that can be sent through one of four 10GbE CX4 ports on the ROACH board. Thus, four input signals can be sampled, packetized and sent over two 10GbE links, to be received by a single dual port 10GbE NIC on a PCIe x8 or x16 slot of a compute node (Dell T630/620 server). The data thus received is redistributed between the compute nodes over the 40 Gbps Infiniband interconnect. The redistributed data which represents a slice of contiguous time series data from each antenna, is passed onto Nvidia GPUs for further processing. Each compute node is capable of handling two GPUs with each GPU processing half the slice of the contiguous time series data received by the compute node. Figure 2 shows the data flow in one FPGA-compute node combination; the details are explained in the following sub-sections. Overall, the 16-antenna system has been implemented using 16 iADCs, 8 ROACH boards, 8 Dell T630/T620s as compute nodes, 1 Dell R720 as the visibility host node and 4 Dell T620/T630s as the beam host nodes, 16 Nvidia Tesla K20c GPU cards and a 32-port Mellanox Infiniband QDR switch.

The baseband filter choices in the upgraded GMRT receiver are 400, 200 and 100 MHz. This allows the GWB to be operated in either of 3 modes viz., 400 MHz bandwidth mode with 4-bits per sample, 200 MHz bandwidth mode with 8-bits per sample and narrow band mode at 100 MHz or below with 8-bits per sample.

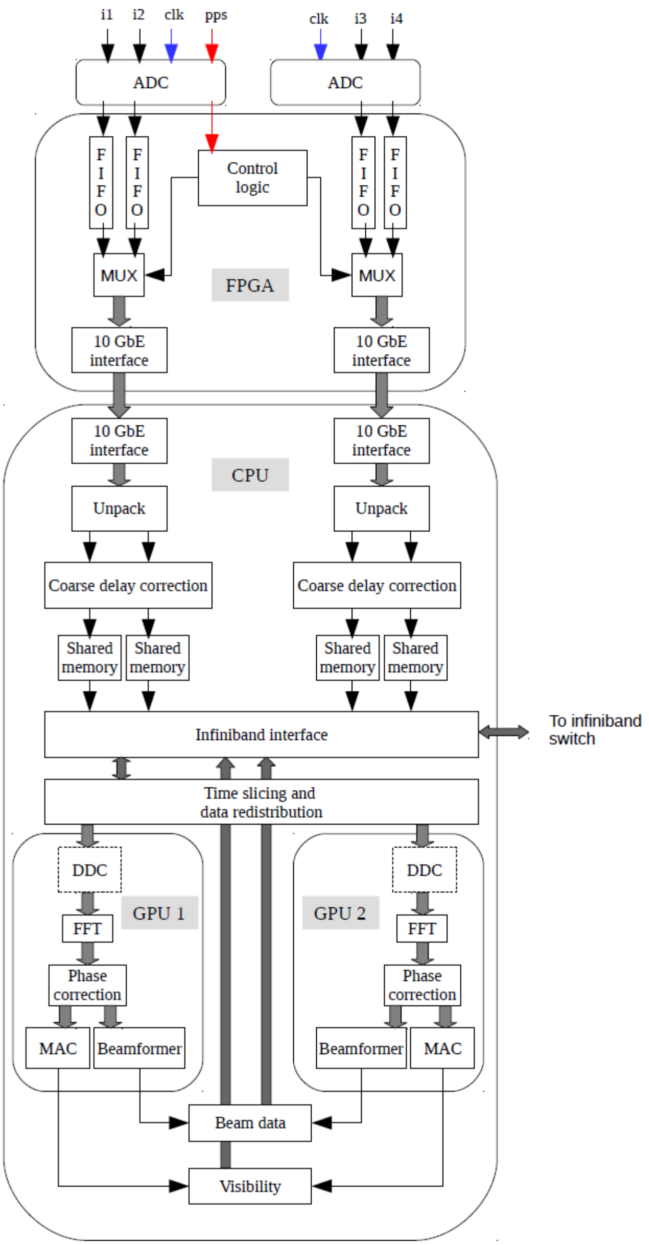

Fig. 2. Data flow in one FPGA-compute node pair. i1, i2, i3, i4 are baseband signals, clk is the ADC sampling clock, pps is the trigger signal.

4.1. *Digitization and packetization of the data on the FPGA boards*

The design on the FPGA has been made using Matlab Simulink blocks, the CASPER BEE-XPS block set and compiled using Casper's MSSGE tool flow. The total payload of each packet is 8200 bytes with 8 bytes of packet counter at the start, which is used to maintain time synchronization, in case any packet is lost. The packet structure is 42 bytes of UDP packet header, 8 bytes of packet counter and 4096 bytes from each input of the iADC. The sampled data is initially buffered into FIFO buffers, dedicated to each input of iADC, to bundle 4096 bytes of contiguous time series data from each input into a packet. For synchronizing the data from all the ROACH boards, a GPS PPS/PPM signal is connected to the sync input of each iADC. The data rate on each 10 GbE link is 6.4 Gbps in the 400 MHz mode (4 bits per sample), as well as the 200 MHz mode (8 bits per sample).

4.2. *Data capture on compute node and coarse delay correction*

The digitized data that is sent over the 10 GbE links as UDP packets is received on the compute nodes through 10GbE NICs. Each compute node has a dual port 10GbE NIC to receive digitized data from two 10GbE links from each ROACH board. Hence, each compute node receives digitized data from four inputs (two over each 10GbE NIC) (see Fig. 2). The packets thus received are unpacked and written into shared memories dedicated for each antenna. Four shared memories are created on each compute node for each antenna input signal. Each shared memory is arranged in ring buffer fashion with four buffers each of 256 MB size (which corresponds to 0.67 s of time, for 8-bit Nyquist sampling of 200 MHz bandwidth or 4-bit Nyquist sampling of 400 MHz bandwidth). The digitized data from each antenna is corrected for corresponding coarse delay (with respect to a fixed reference location in the array) at the start of every 256 MB buffer, before writing into the shared memory buffers.

The maximum delay between the farthest antenna and reference antenna in the GMRT array is nearly 128 μs which corresponds to 51,200 and 102,400 samples at 200 and 400 MHz bandwidth modes, respectively. This maximum delay converts to 51,200 bytes, for both these modes. For correcting coarse delay (in integer clock cycles), the digitized time-series data before being written into shared memory are written into a temporary buffer (current temporary buffer) with a size of 256 MB + $(2 \times 57,344)$ bytes (see Fig. 4). The number 57,344 is chosen as it is the nearest multiple of 8192 (the packet size) that covers the maximum delay. The delay value (in seconds) for the antenna corresponding to the buffer is calculated with respect to a reference antenna. This delay value depends on the geometric location of antenna, the source that is being observed and the time of day. The delay value can be either positive or negative depending on the position of the antenna with respect to the reference antenna and source that is being observed. The delay value, after conversion into bytes, is added (or subtracted) to 57,344 and is taken as starting point from where 256 MB are written from current temporary buffer into shared memory. The last $(2 \times 57,344)$ bytes of current temporary buffer are written into the next temporary buffer. Next temporary buffer and current temporary buffer are interchanged for the next 256 MB of digitized time-series data and the coarse delay correction process mentioned above is repeated.

For the 400 MHz 4 bit mode, each byte has two samples. Hence, coarse delay is corrected to the nearest even number of samples and the residual delay is corrected after FFT. Thus, in the 200 MHz 8 bit mode, the residual (fine) delay correction or the fractional sample time correction (FSTC) range is 1 sample, while in the 400 MHz 4 bit mode, FSTC range is 2 samples.

4.3. *Data redistribution between compute nodes*

The data from the four shared memories in each compute node is read out and shared over the Mellanox Infiniband switch with the other compute nodes. The shared memory buffer, which is a time series buffer of 256 MB is imagined as eight (number of compute nodes) units of 32 MB time series buffers. The data sharing happens such that each compute node ends up with one 32 MB time series buffer, from all the 32 input signals coming to the cluster. After data sharing, compute node 1 will be having the first 32 MB buffer from all the antennas (inputs), which means it has to receive the first 32 MB buffer of input signals from all the other compute nodes. At the same time, it sends second to eighth 32 MB buffers to compute nodes 2 to 8. This happens over all the nodes, using synchronized MPI calls. Figure 3 illustrates this time slicing and data redistribution scheme in a 4-node compute cluster.

4.4. *Data processing on GPUs on compute nodes*

After data sharing, each compute node has a slice of time series data from all antennas. Compute node 1 will have the first slice, compute node 2 will have the second slice and so on. Each slice of 32 MB of time series data is processed by two GPUs, each processing half of the slice i.e. GPU 1 will process first 16 MB of the slice from all antennas and GPU 2 will process second 16 MB of the slice from all antennas. This processing involves the following steps: converting 8/4 bit samples into floating point, along

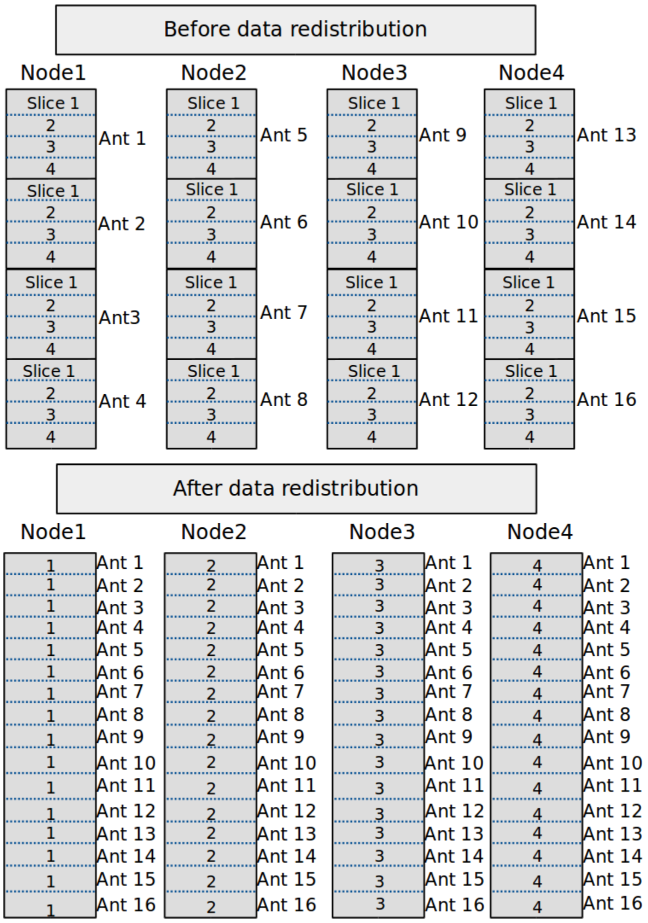

Fig. 3. GWB data redistribution in a 4 node compute cluster. "Ant X" indicates time series data from antenna X.

with digital down conversion (only for the narrow band modes), FFT for converting time series data to spectral channels, phase correction of spectral data for residual delay and fringe compensation, MAC and beamformer operations to obtain the visibility and beam data products of final interest.

4.4.1. *FFT and phase correction*

The FFT for converting time series data to spectra is performed using the Nvidia CUDA CUFFT library. Real to Complex FFT is performed efficiently by performing Complex to Complex CUFFT and reconstructing the output taking advantage of the symmetry properties (Press *et al.*, 1992). While reconstructing, parallelism is achieved over number of channels at the thread level and number of FFTs at the block level.

The coarse delay correction (explained in Sec. 4.2) compensates the delay up to the nearest integer clock sample or two. This leaves a residual delay of up to 1 sample in the case of the 200 MHz 8-bit mode and up to two samples in the case of the 400 MHz 4-bit mode. This residual delay correction (or FSTC) is achieved by applying an appropriate gradient in phase across the spectral channels. The FSTC values for each antenna are pre-calculated at the start of processing of each 256 MB buffer and loaded into the constant memory of the GPU. Along with the FSTC values, additional broadband phase compensation for each antenna (arises because of the delay being corrected after baseband conversion instead of being corrected at sky frequency) known as fringe de-rotation are pre-calculated and loaded into constant memory of GPU. Apart from FSTC and fringe compensation values, to keep up with change of delay within the buffer, rate of change of delay is also pre-calculated and extrapolated for each FFT spectrum. Finally, any antenna-based phase offsets are also corrected here.

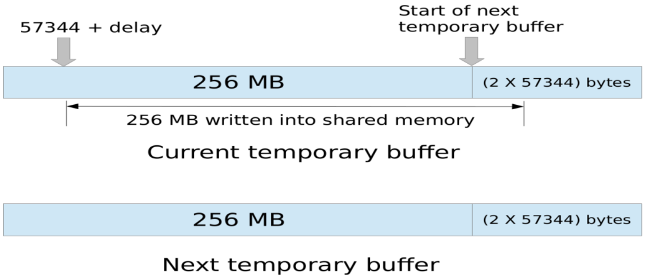

Fig. 4. Illustrating coars delay correction. Current temporary buffer and next temporary buffer are interchange for every 256 MB of digitized data received for each antenna input.

Each spectral channel is accessed by a thread in the GPU. The same thread calculates the total amount of phase from the FSTC, fringe and rate of change of delay values and applies the net value to each spectral channel. Parallelism over antennas is achieved at the block level.

4.4.2. *MAC operations*

The MAC part is the most compute intensive of all the processing. For an array of N antennas, there are $N(N+1)/2$ baselines (including self- and cross-correlations of all antennas) and corresponding visibility data. The MAC stage was implemented following Group Parallel approach (GPMAC from here) presented by Harris *et al.* (2008). A similar approach was examined on GeForce 8800GTS, GeForce 8800GTX and Tesla C1060 Nvidia GPUs in the MWA 32 antenna prototype (Wayth *et al.*, 2009), with group size of two or four. Group size limitation is decided by number of registers and shared memory availability per block in a GPU. In GWB, GPMAC was implemented with group size 16 on Tesla K20c and K40c Nvidia GPUs and achieved a performance of 433 GFlops on K20c and 530 GFlops on K40c, respectively. The performance achieved is only 12% of the peak performance possible with these GPUs because of non-coalesced global memory access and low arithmetic intensity i.e. number of floating point operations performed per byte of global memory transfer (Clark *et al.*, 2012).

There was an attempt to use the xGPU library (Clark *et al.*, 2012) for the MAC in the GWB. Though some performance improvement was observed with the xGPU library over GPMAC on Tesla K20c, implementing the xGPU library for the time slicing model used in the GWB, necessitated a 3-dimensional transpose of data post FFT stage (similar to a corner turner) and another data re-ordering of the visibility output to be compatible with visibility output recording stage of the GWB. The post FFT 3-dimensional transpose (a global memory intensive operation) reduced the performance improvement of xGPU over GPMAC. Output side data re-ordering (another global memory intensive operation) nullified the performance improvement of xGPU library for spectral channels below 8192, and in cases of spectral channels above 8192, the performance was significantly degraded. Consequently, the final choice was to go with the GPMAC approach for the GWB design.

4.4.3. *Beamformer operations*

The design provides for four beams simultaneously, each of which can be configured as an IA or PA beam with any combination of antennas, allowing for up to 4 sub-arrays to be supported in the beamformer. Different sub-arrays can be configured to track a given phase center with individual sub-array set to different frequency of observation. This feature is useful for simultaneous multiple frequency observation of same pulsar. In all the sub-arrays, the PA beam formed is only at phase center. For the IA beam, spectral data from selected antennas is first squared to convert to intensity and then added separately for each spectral channel, whereas for the PA mode, the complex spectral data are first added for all the selected antennas and then converted to intensity samples. In both the cases, the raw beamformer output is integrated in time to the desired resolution. The minimum sampling period achieved is 20 μs for the IA beam and 40 μs for the PA beam. While computing beam data, parallelism is achieved over number of spectral channels at the thread level; and over number of sampling period blocks in the 16 MB buffer processed by each GPU, at the block level. The number of sampling period blocks is calculated by dividing the number of FFTs in a 16 MB buffer by the number of FFTs in one sampling period block. Each thread carries out addition of one spectral channel for all the FFTs in a sampling period block. The design was optimized to read the global memory of the GPU only once for the processing of all beams. The beam data from all the compute nodes are gathered onto a host node for recording, for each beam separately.

The GWB supports a voltage beam mode to facilitate special signal processing techniques (like coherent dedispersion) on the phased voltage signal from the PA beamformer. In this mode, the phased voltage signal (in spectral domain with a specified number of frequency channels) from the two polarizations of a phased array beam is transferred to a single host machine, and written in a ring buffer in shared memory. This allows an external process on the host node to attach to this memory segment and perform real-time coherent dedispersion, or if required, record the data on a disk (limited by disk writing speed).

4.4.4. *Real-time coherent dedispersion for the voltage beamformer*

Residual smearing in incoherent dedispersion at low frequencies can severely limit high time resolution pulsar studies. While the dispersive effects of the interstellar medium can be completely corrected using coherent dedispersion (Hankins and Rickett, 1975), the use of this technique is limited due to the high computational requirements for real-time implementation for large bandwidths or alternatively, the large disk I/O requirements for offline processing. A real-time GPU-based coherent dedispersion system was previously developed for the GSB by De & Gupta (2016), capable of processing in once single chunk the entire 32 MHz bandwidth of the legacy back-end, producing dedispersed data at the Nyquist time resolution. However, such a technique is not feasible for the larger bandwidths of the GWB, owing to the manifold increase in computational and memory requirements.

Since dedispersed output at the Nyquist time resolution is rarely required for most pulsar studies, one can instead perform coherent dedispersion by phase correcting smaller parts of the entire bandwidth to produce coherently dedispersed sub-bands with lower time resolution which are aligned offline in incoherent dedispersion. This process significantly reduces the computational cost of the technique (allowing for real-time processing), and allows a much larger range of Dispersion Measure (DMs) to be processed on the limited memory of a GPU. The number of sub-bands that can be used for a given bandwidth is a trade-off between requirements of higher time resolution (lower number of sub-bands) and capability of real-time performance (higher number of sub-bands). The technique discussed here (and designed for the GWB) is similar to the Convolving Filterbank approach described in van Straten and Bailes (2010). Several radio telescopes today have real-time (or quasi real-time) coherent dedispersion systems which use similar techniques (typically at higher frequencies of observation), such as the PuMa-II back-end for the Westerbrok Synthesis Radio Telescope (Karuppusamy *et al.*, 2008), the Green Bank Ultimate Pulsar Processing Instrument (GUPPI; Ford *et al.*, 2010) and the Nancay radio telescope correlator (Ait allal *et al.*, 2009).

This method has been implemented to run on a single GWB host machine housing two NVIDIA Tesla K40c GPUs, each processing a single polarization of a phased array voltage beam. We use batched FFT routines provided by the cuFFT library to initially perform IFFT of multiple consecutive voltage spectra, to recover the phased voltage time series. This is followed by multiple forward FFTs (in batch mode) with overlapping samples (for overlap-save method) using in-built cuFFT overlap options. Application of the inverse ISM phase function combined with a smooth filter multiplication for individual sub-bands (each with a typical user defined bandwidth ∼0.2 to 2 MHz) is performed using GPU kernels, where the complex multiplication function is initialized only in the beginning and subsequently accessed from global memory. Finally, multiple IFFTs (one for each sub-band) are computed to recover the corrected voltage time series for each sub-band. The final rejection of overlap-save samples for the individual sub-bands is performed in a GPU kernel along with squaring of the voltages. The power signals for the two polarizations are sent out of the GPU, where they are added and integrated further in time as needed, before writing to disk.

The aforementioned system is currently capable of processing 200 MHz bandwidth in real-time, using 256 or 512 sub-bands for a large range of DMs (∼100 pc/cc at 300–500 MHz; higher DMs possible with larger number of sub-bands), achieving a total computation rate of ∼220 GFlops on two Tesla K40c GPUs. The smaller number of (dedispersed) channels allows one to achieve higher time resolutions than the standard IA/PA beam modes. This implementation has resulted in one of the largest bandwidth real-time coherent dedispersion systems in operation at these low frequencies (below 0.5 GHz).

4.4.5. *Narrowband modes*

Narrowband modes are used in spectral line observations where the signal of interest is present in a small portion of the full baseband signal. In the GWB, this mode is implemented using the Digital Down Conversion (DDC) technique, as illustrated in Fig. 5. DDC is a technique that takes a band limited high sample rate digitized signal, mixes the

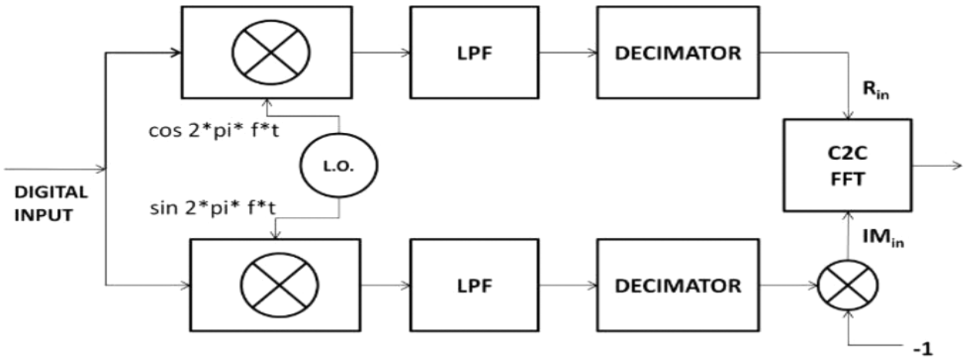

Fig. 5. Block diagram for the DDC scheme for the narrow bands of the GWB.

signal to a lower frequency and reduces the sample rate while retaining all the information in the band of interest. The digital signal is multiplied by sine and cosine of the DDC LO frequency. The DDC LO value chosen is the starting frequency of the narrowband that is selected for down-conversion. This mixing operation generates In-phase (I) and Quadrature (Q) signals. The I and Q signals are low pass filtered (anti-aliasing filter) to remove the higher frequency signals above the reduced bandwidth. FFT is performed on I and Q signals treating Q signal as imaginary component of complex FFT. A phase shift of 180° is given to Q signal so that positive frequency components after the mixing operation are eliminated (Lyons, 2004).

The digital synthesizer (cosine and sine values generation), mixing operation, FIR filtering and decimation are performed on the GPU. The windowed sinc FIR filter coefficients are generated using MATLAB FDAtool using Kaiser window and 64 (for decimation factors 2 and 4), 128 (for decimation factors 8, 16 and 32) or 256 (for decimation factor 64) taps. As decimation factor increases, the data to be processed in MAC stage becomes less by decimation factor and this allows in performing filtering operation with higher number of taps. The filtering operation is the most time taking part of the DDC, as it requires reading number of taps data samples from global memory and reading FIR filter coefficients for generating one FIR filtered data sample. For better performance, filter coefficients are loaded onto constant memory (to reduce global memory access) of GPU at the start of observation.

After DDC, correlation operation is performed on the de-sampled or decimated signals, in the usual fashion by the MAC block. In the GWB, this narrowband mode has been implemented for the 100 MHz, 8-bit mode. The decimation range is from 1 (100 MHz Bandwidth) to 64 (1.5625 MHz Bandwidth). In narrowband modes, the usable signal is less than the bandwidth that is obtained after decimation because of limitations of the anti-aliasing filter. There is flexibility in choosing the narrowband anywhere within the 100 MHz baseband signal by changing the DDC LO value. The flexibility extends to sub-arrays too, where each sub-array can observe any part of the spectrum. The decimation value is kept same over all the sub-arrays. Another feature is that two polarizations of same antenna can view different parts of the spectrum using different DDC LO values.

4.5. *Time keeping in the GWB*

In the GWB, the start of an observation is synchronized with a 1-PPS signal derived from the observatory GPS receiver. Initially, all the ROACH boards are programmed, such that they wait for a “master set” signal in the FPGA to go high for starting the digitization and packetization. Then, an initialization (“init”) command is given to the GWB through an online control server by the operator. 5 s after this “init” command, the “sync set” signal in the FPGA is made high through a telnet port (Fig. 6). After the “sync set” signal is made high, the “master set” signal goes high at the start of the next 1-PPS pulse. Once the “master set” becomes high, packets are formed with the sampled data and sent to compute nodes through 10GbE link. Along with the sampled data, packet number information is sent for every packet.

Packets are received by the compute nodes through the10GbE links. Upon receiving the first packet, the system time is stored in a timeval structure, as the start time, rounded off to the previous second boundary. For subsequent buffers, the timestamp is calculated by incrementing the time value by the buffer duration, which is 0.67108864 s, as explained in Sec 4.2. This timestamp is propagated down the chain to the visibility and beamformer data streams.

5. Software Model, Optimization Techniques and Performance Achieved

5.1. *Software model for the GWB*

Figure 7 illustrates the top level software model for the design of the GWB. OpenMP sections are used to perform various stages of processing in parallel on

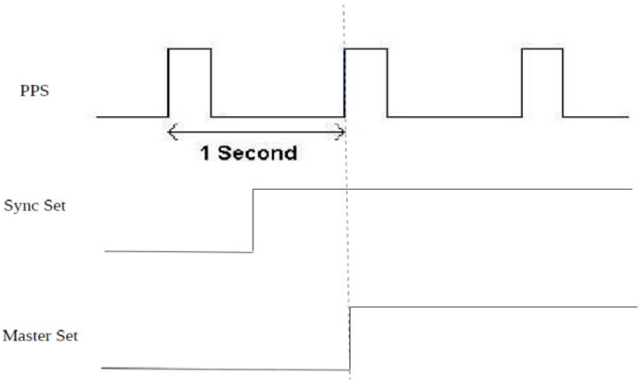

Fig. 6. Synchronizing in the GWB using the GPS 1-PPS signal.

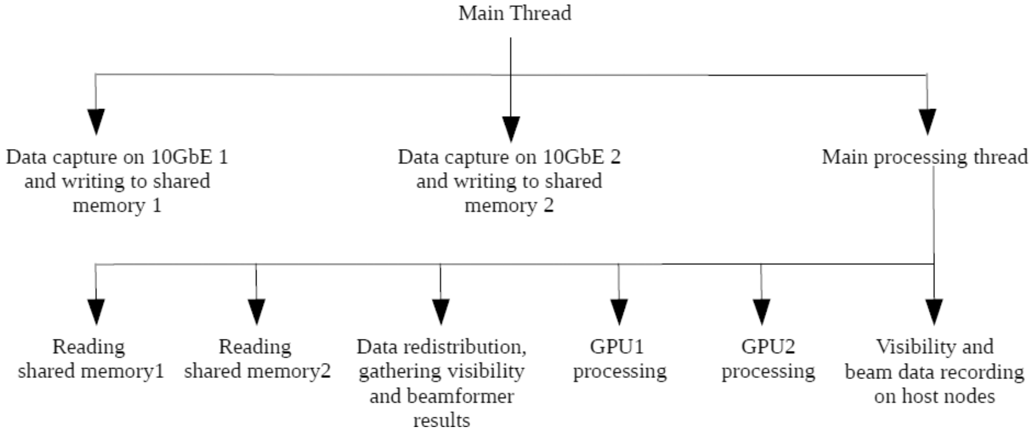
Main Thread
Data capture on 10GbE 1 and writing to shared memory 1
Data capture on 10GbE 2 and writing to shared memory 2
Main processing thread
Reading shared memory1
Reading shared memory2
Data redistribution, gathering visibility and beamformer results
GPU1 processing
GPU2 processing
Visibility and beam data recording on host nodes

Fig. 7. Parallel OpenMP sections model in the GWB.

the compute nodes, with individual threads devoted to specific tasks. The CPU affinity feature of OpenMP has been used to assign threads to a particular CPU which hosts the peripheral that is used in the process, to exploit the advantage of Non-Unified Memory Access (NUMA) architecture and to efficiently handle interrupt requests. MPI programming is used for running the program over the cluster, for data redistribution and for gathering the results on the host nodes. Data capturing on compute node and coarse delay correction is done using PSRDADA software, written in C, developed in collaboration with Swinburne University, Australia. GPU processing code for FFT and MAC uses the CUDA programming framework.

5.2. *Performance achieved*

Computational performance: This was measured on two different GPUs — Telsa K20c and Tesla K40c. Time taken for computation (correlation + beamformer) on Tesla K40c is nearly 64% of real-time for 16,384 spectral channels and 400 MHz 4-bit mode with 2 PA beams. For Tesla K20c, time taken is nearly 80% of real-time. In terms of flops, overall computation performance achieved on Tesla K40c is 357 Gflops and on Tesla K20c is 281 Gflops. The performance achieved is significantly less than the peak performance as the data samples, after being loaded onto the GPU global memory, are read multiple times i.e. for each processing stage — FFT, Phase shifting, MAC and Beamformer.

I/O performance: The data rate achieved between compute nodes for time slicing and data redistribution over infiniband was 25.2 Gbps (bi-directional).

6. Testing of the GWB and Sample Results

The GWB system has gone through various stages of design verification and performance testing, before being released for user level trials and science observations. The initial tests were done by running the basic GWB correlator on noise source inputs. After that, the performance of the GWB was benchmarked against the GSB — the working correlator for the legacy GMRT system — which provided a trusted system to compare against. The first main goal was to see that the GWB outputs should match those of the GSB, for low level tests such as values of visibility amplitude and phase on the same calibrator sources observed simultaneously; after the GWB passed these basic tests, it was checked to see if it provides the enhanced performance features such as improved sensitivity when the signal over the entire wideband is combined into a scientific result. We present here sample results from such tests of the different modes of operation of the GWB, that demonstrate the proper working of this new wideband back-end, and also offer a glimpse of the potential of the enhanced performance that the user can look forward to with the uGMRT and the GWB. Where ever possible, we have compared the performance of the new system with that of the legacy system, as it offers one of the best ways to verify the improved performance that has been the target of the upgrade.

6.1. *Interferometry results*

Figure 8 shows one of the sample results from interferometric imaging of a target test source, 3C129, observed simultaneously with both the existing legacy GMRT system and the upgraded GMRT

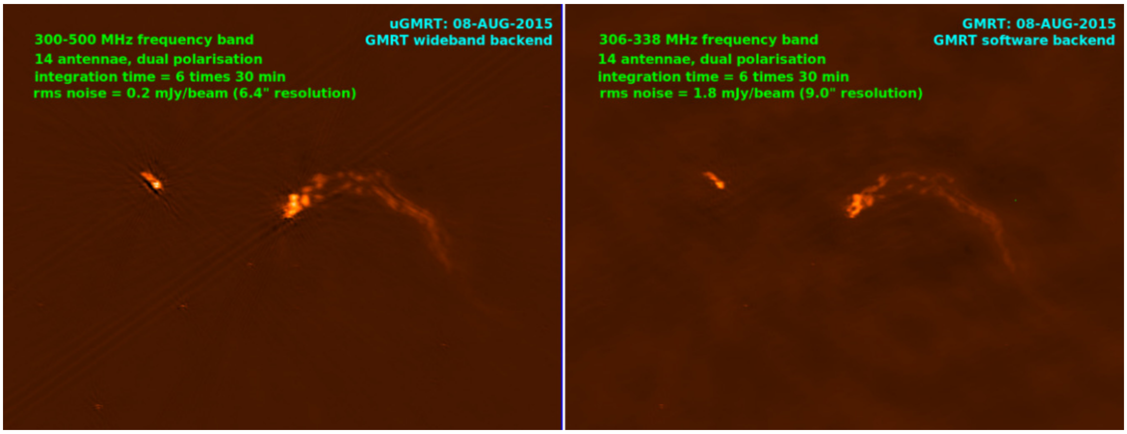
uGMRT: 08-AUG-2015
GMRT wideband backend
300-500 MHz frequency band
14 antennae, dual polarisation
integration time = 6 times 30 min
rms noise = 0.2 mJy/beam (6.4" resolution)
GMRT: 08-AUG-2015
GMRT software backend
306-338 MHz frequency band
14 antennae, dual polarisation
integration time = 6 times 30 min
rms noise = 1.8 mJy/beam (9.0" resolution)

Fig. 8. Sample results from imaging of 3C129 with the uGMRT (left panel), compared with the legacy GMRT (right panel).

(using the wideband receivers and the GWB). A total of 14 antennas were used for these observations, which spanned a frequency range of 306 to 338 MHz for the legacy system and 300 to 500 MHz for the upgraded system. Not only does the uGMRT system reproduce the features of the source faithfully, the rms noise in the image is 0.2 mJy per beam for the uGMRT system, compared to 1.8 mJy per beam for the legacy GMRT system, clearly bringing out the improvement achieved with the wideband receiver system and the GWB.

Figure 9 shows a sample result from test observations of a spectral line using the narrow band mode of the GWB. Here, a sub-band of 3.125 MHz is extracted using the DDC, from the input band of 100 MHz, to zoom in on a red-shifted HI absorption line, centered at 1048.65 MHz. The line is detected at the expected location, with the expected properties, verifying the proper functioning of the narrow band mode.

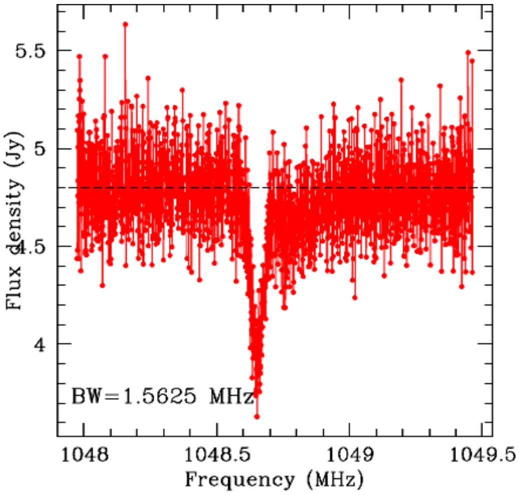

Fig. 9. Sample spectral line result from the GWB narrowband mode.

6.2. *Beamformer results*

Several systematic tests have been carried out of the IA and PA beamformer of the GWB, and the system has been found to work as per the specifications. Figure 10 shows a sample data set from PA mode observations over the frequency range of 1260 to 1460 MHz (using a millisecond pulsar as a test source). The signal is faithfully reproduced across the 200 MHz band. Figure 11 shows a sample result from comparison of the average profile of another millisecond pulsar obtained from simultaneous L-band observations with the uGMRT (200 MHz bandwidth) and the existing legacy system (32 MHz bandwidth), showing that the uGMRT result delivers a measurably higher signal-to-noise ratio than the legacy system.

Finally, Fig. 12 shows a sample result validating the working of the voltage mode and coherent dedispersion system for the GWB. The left panel shows the results of observing the well-known millisecond pulsar B1937+21 with the PA beam mode of the GWB in the 300 to 500 MHz band of the uGMRT: on the top is the average folded profile of the pulsar integrated over the entire band, and on the bottom is the gray scale plot showing the profile over different sub-bands over the 200 MHz bandwidth. The right panel shows the same quantities for the pulsar signal after it has been processed through the coherent dedispersion pipeline of the GWB. As can be seen, the dispersion correction is much more accurate for the latter, resulting in a dramatically sharper and clearer pulsar profile that

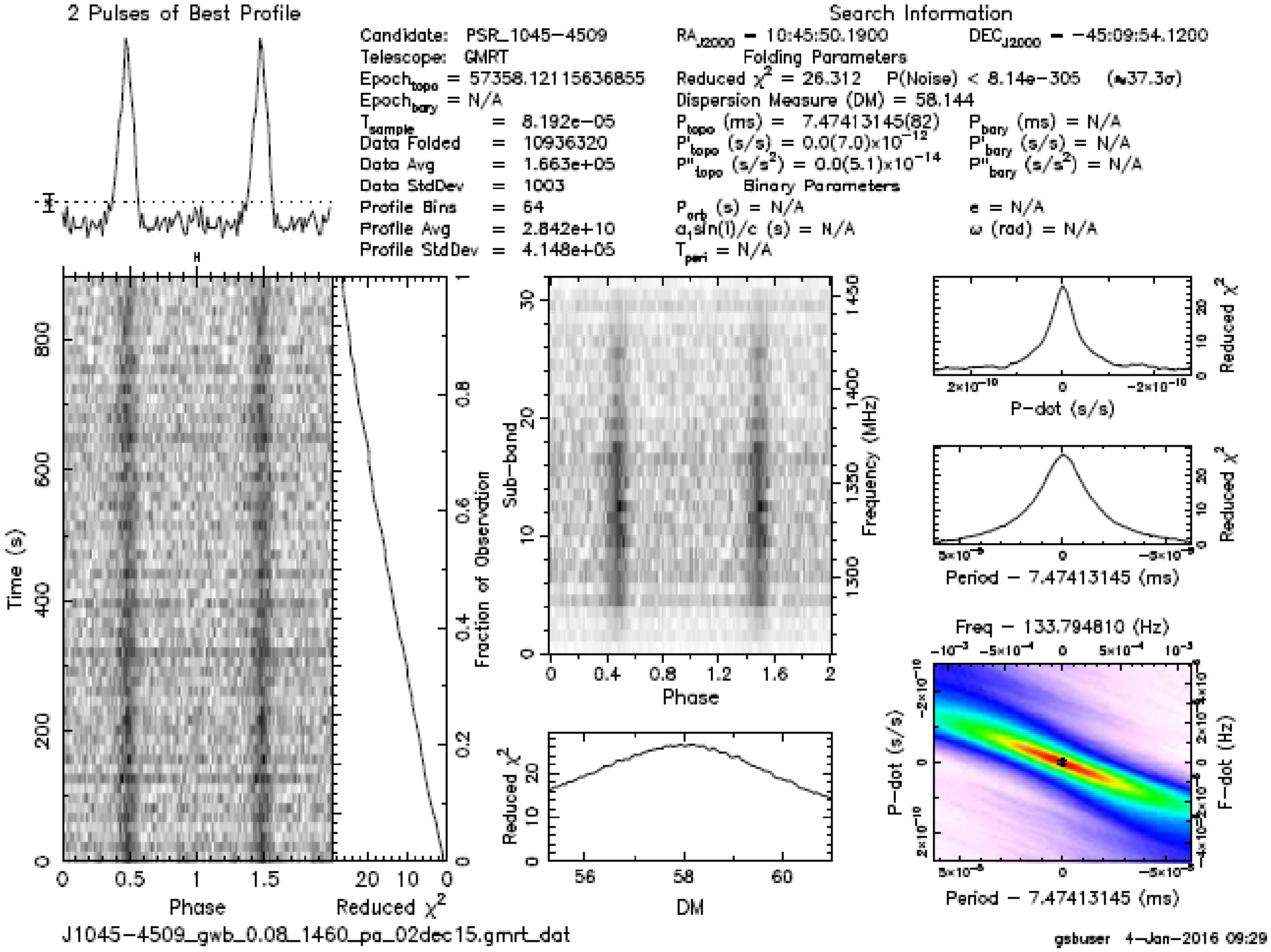

Fig. 10. Folded profiles for PSR J1713+0747 at L-band with 200 MHz bandwidth with the uGMRT (top) and 32 MHz bandwidth with the legacy GMRT systemv (bottom).

maintains its shape and sharpness uniformly across the entire the 200 MHz band, unlike the PA mode signal which smears out completely at the lower frequencies of the band. The GWB will produce such high quality pulsar signals routinely with the real-time coherent dedispersion system over such large bandwidths.

7. Ongoing Developments and Future Plans

Even as the 16-antenna version of the GWB has been shown to be working well and has been released to users for science observations, work is underway to expand this to a full 32-antennas system for the uGMRT — this is expected to be released to the users by the end of 2017. Furthermore, work is on to add new modes to the GWB design, as well as enhance the performance of the existing modes.

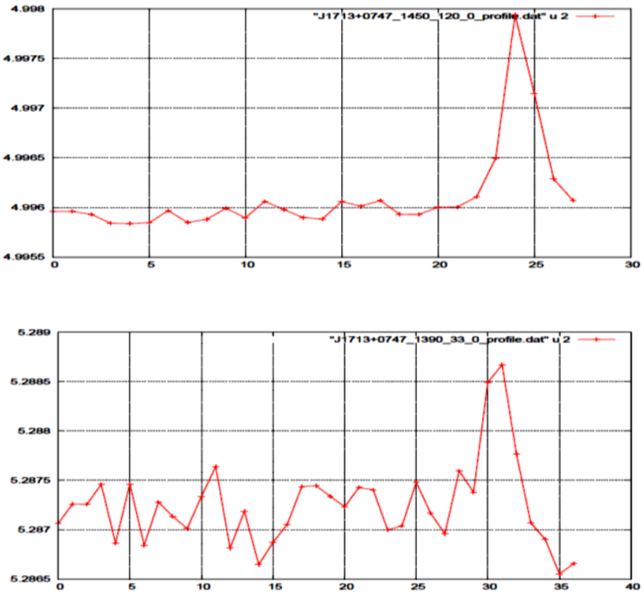

Fig. 11. Sample beamformer result using the PA beam of the GWB at L-band with 200 MHz bandwidth (1260 to 1460 MHz) for the millisecond pulsar J1045-4509.

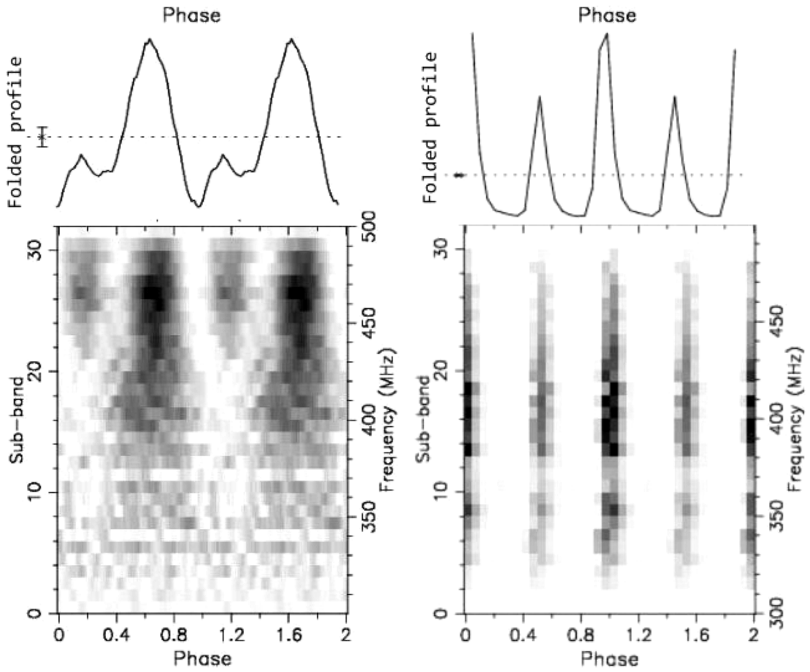

Fig. 12. Comparison of signal from the millisecond pulsar B1937+21, using the GWB with (right panel) and without (left panel) coherent dedispersion, for observation in the 300 to 500 MHz range of the uGMRT.

7.1. *Pulsar gating*

Pulsars can prove to be the ideal point sources for calibration of an interferometric array, provided the signal from the on and off pulse regions is integrated separately in the correlator (as has been demonstrated for the GSB by Pen *et al.*, 2009). This technique of pulsar gating has other applications, such as localizing the sky position of newly discovered pulsars, studying emission properties of pulsars in more detail. If one knows the period and starting phase of the pulsar signal, then it is possible to record visibility data separately for on-pulse and off-pulse regions. From the difference of these, it is possible to solve for the antenna gains, thereby using the pulsar as an in-field phase calibrator. It is planned to implement such a gated correlator mode for the GWB in the near future.

7.2. *Polyphase filter bank*

Performing the spectral channelization in a correlator using the FFT suffers from two drawbacks — spectral leakage and scalloping loss (Lyons, 2004). Performing a Polyphase Filter Bank (PFB) before the DFT helps in overcoming these and gives better results (Harris and Haines, 2011). The PFB has been successfully implemented and tested on a 2-antenna version of the GWB, using the mechanism followed by Chennamangalam (2011), for implementing a PFB spectrometer on GPU using CUDA C at the Green Bank Telescope. It is planned to scale this to the full GWB with some extra computation cost, which depends on the number of taps (or order) used for filtering.

7.3. *Development of raw voltage recording option*

In the GWB design, there is a provision for attaching a pipeline for recording raw voltages for off-line processing for specialized applications. Each ROACH boards has two spare 10GbE interfaces available through which sampled data can be packetized and sent to additional server nodes for recording. On the Dell T630 class of machine, the sustainable writing speed onto disk has been tested to be around 100 MB/s. This will straightaway allow for 100 MHz bandwidth signals from individual antennas to be recorded with 4-bit precision, and other combinations and future growth in technology will allow larger bandwidths to be achieved.

7.4. *Development of multi-beaming option*

As an extension of the raw voltage recording system above, it is planned to build a compute platform that will allow for raw voltages from multiple antennas to be combined with appropriate phase terms to generate a very large number of phased array beams covering a significant area of the primary beam of the antenna. This will allow for high sensitivity, large scale pulsar surveys to be carried out more effectively. Each beam output will have a pulsar search or transient detection pipeline attached to it. The application is highly scalable, and one can begin with a limited number of beams and keep expanding as the better compute technology becomes available.

8. Conclusions

A real-time back-end for a multi-antenna radio telescope has been implemented on a hybrid platform with FPGA and CPU-GPU systems. Several new techniques have been developed as part of this design, which is quite modular and can be easily scaled to meet next generation radio telescope requirements. It can also be easily reprogrammed to add additional features. A 400 MHz bandwidth, 16-antenna dual polarization implementation at the upgraded GMRT has been shown to be working well, and is being expanded to a 32-antenna system.

Acknowledgments

We would like to thank the following persons and groups for their contributions:

- Andrew Jameson, Swinburne University, Australia for his help in PSRDADA software programming;
- Benjamin Barsdell, Swinburne University, Australia (now at Nvidia, Santa Clara, CA, USA) for his help in initial stages of GPU programming;
- Sandeep Chaudhari and Mekhala Muley, Digital Backend Group, GMRT for their help in FPGA programming and python programming for controlling ROACH boards;
- Dharam Vir Lal, NCRA for sample results from the normal imaging mode of the GWB;
- Nissim Kanekar, NCRA for sample results from tests of the GWB in narrowband modes;
- CASPER group, Berkeley for their technical support for ROACH boards;
- Nvidia, India for their technical support for GPU cards and help in GPU programming;
- Back-end group, GMRT for their help in installation of hardware;
- Control room group, GMRT for their help in testing the GWB;
- Computer group, GMRT for their help in installation of software on GWB compute and host nodes.

References

Ait allal, D., Weber, R., Cognard, I. *et al.* [2009], "RFI mitigation in the context of pulsar coherent dedispersion at the Nancay radio astronomical observatory," *17th European Signal Processing Conf.* Glasgow, Scotland.

Chennamangalam J., [2011] *The Polyphase Filter Bank Technique*, http://casper.berkeley.edu/wiki/The_Polyphase_Filter_Bank_Technique.

Clark, M. A., La Plante, P. C. & Greenhill, L. J. [2012] "Accelerating radio astronomy cross-correlation with graphics processing units," *IJHPCA*, doi: 10.1177/1094342012444794.

De, K. & Gupta, Y. [2016] "A real-time coherent dedispersion pipeline for the giant metrewave radio telescope, experimental astronomy," **41** (1–2), 67–93.

Deller, A. T., Tingay, S. J., Bailes, M. & West, C. [2007] "DiFX: A software correlator for very long baseline interferometry using multiprocessor computing environments," *PASP* **119**(853) 318–336.

Denman, N., Amiri, M., Bandura, K. *et al.* [2015] *A GPU-Based Correlator X-Engine Implemented on the CHIME Pathfinder*, arXiv:1503.06202.

Escoffier, R. P., Comoretto, G., Webber, J. C. *et al.*, [2007] The ALMA correlator, *A&A*, **462**(2) 801–810.

Ford, J. M., Demorest, P. & Ransom, S. [2010] "Heterogeneous real-time computing in radio astronomy," in *Proc. SPIE, Software and Cyberinfrastructure for Astronomy*, Vol. 7740, 77400A.

Gupta, Y. *et al*, [2000] *IAU Colloq. 177, Pulsar Astronomy: 2000 and Beyond*, p. 277

Gupta, Y. [2014] *The Metrewavelength Sky*, ASI Conf. Series, Vol. 13.

Hankins, T. H. & Rickett, B. J. [1975] "Pulsar signal processing," *Methods Comput. Phys.* **14**, 55.

Harris, C., Haines, K. & Staveley-Smith, L. [2008]. "GPU accelerated radio astronomy signal convolution," *Exp. Astron.* **22**, 129–141.

Harris, C. & Haines, K. [2011] "A Mathematical review of polyphase filterbank implementations for radio astronomy," *Publ. Astron. Soc. Aust.* **28**(4), 317–322.

Karuppusamy, R., Stappers, M. & van Straten, W. [2008] PuMa-II: A wide band pulsar machine for the westerbork synthesis radio telescope, *Publ. Astron. Soc. Pacific* **120** 191–202.

Kocz, T. *et al.* [2014] *JAI* **3**, doi: 10.1142/S2251171714500020.

Kocz, J., Greenhill, L. J., Barsdell, B. R. *et al.* [2015] "Digital signal processing using stream high performance computing: A 512-input broadband correlator for radio astronomy," *J. Astron. Instrum.* **4**, 1550003.

Parsons, A., Backer, D., Siemion, A. *et al.* [2008] "A scalable correlator architecture based on modular FPGA hardware, reuseable gateware, and data packetization," *PASP*, **120** (873), 1207–1221.

Parsons, A. R., Backer, D. C., Foster, G. S. *et al.* [2010] "The precision array for probing the epoch of re-ionization: Eight station results," *AJ* **139**(4), 1468–1480.

Perley, R., Napier, P., Jackson, J. *et al.* [2009] "The expanded very large array," *Proc. IEEE* **97**(8), 1448–1462.

Perley, R. A., Chandler, C. J., Butler, B. J. *et al.* [2011] "The expanded very large array: A new telescope for new science," *ApJL* **739**(1) L1.

Pen, U.-L., Chang, T.-C., Hirata, C. M. *et al.* [2009] The GMRT EoR experiment: Limits on polarized sky brightness at 150 MHz," *Mon. Not. R. Astron. Soc.* **399**, 181–194 (2009).

Lyons, R. G. [2004] "The discrete fourier transform," Chapter 3; "Quadrature Signals," Chapter 8, in *Understanding Digital Signal Processing*, 2nd edition, Prentice Hall PTR, New Jersey, pp. 81–98; pp. 439–478.

Roy, J., Gupta, Y. *et al.* [2010] "A real-time software backend for the GMRT," *Exptl. Astron.* **28**, 25–60.

Swarup, G., Ananthakrishnan, S., Subrahmanya, C. R. *et al.* [1997] "The giant metrewave radiotelescope, high-sensitivity radio astronomy," in *Proc. Meeting held at Jodrell Bank, University of Manchester*, January 22–26, 1996. p. 217.

Taylor, G. B., Ellingson, S. W., Kassim, N. E. *et al.* [2012] "First light for the first station of the long wavelength array," *J. Astron. Instrum.* **1**(1) 1250004–284.

Thompson, A. R., Moran, J. M. & Swenson, G. W. Jr. [2001] *Interferometry and Synthesis in Radio Astronomy*, 2nd edition (Wiley, New York) p. 692.

Urry, W. L., Wright, M., Dexter, M. & Macmahon, D. [2007] "ATA memo 73," The ATA Correlator, 16 Feb.

van Straten W. & Bailes M. [2010] DSPSR: Digital signal processing software for pulsar astronomy," *Publ. Astron. Soc. Aust.* **28**, 1–14.

Wayth, R. B., Greenhill, L. J. & Briggs, F. H. [2009] "A GPU-based real-time software correlation system for the murchison widefield array prototype," *PASP* **121**, 857–865.

Press, W. H., Teukolsky, S. A., Vetterling W. T. & Flannery, B. P. [1992] "Fast fourier transforms" in *Numerical Recipes in C: The Art of Scientific Computing*, 2nd edition, Cambridge University Press, Cambridge, Chapter 12, pp. 496–536.