HTPCP: GNSS-R multi-channel cross-correlation waveforms post-processing solution for GOLD-RTR Instrument

Guo Yi[a,b], David Atienza[c], Antonio Rius[a], Serni Ribó[a], Carles Ferrer[b,d]
[a]Institut de Ciències de l'Espai (IEEC-CSIC), 08193 Bellaterra (Barcelona), Spain
[b] Departament de Microelectrònica i Sistemes Electrònics (IEEC-UAB), 08193 Bellaterra (Barcelona),Spain
[c] Embedded Systems Laboratory (ESL), EPFL, 1015 Lausanne, Switzerland [d] Institut de Microelectrònica de Barcelona (CNM-CSIC), 08193 Bellaterra (Barcelona), Spain

***Abstract*—Global navigation satellite system reflectometry (GNSS-R) remote sensing is a new remote sensing technique of satellite navigation application. Essentially, it entails a method of remote sensing that receives and processes microwave signals reflected from various surfaces. The GPS open-loop differential real-time receiver (GOLD-RTR) instrument have been proposed as GNSS-R instrument, to gather global positioning satellite system signals after they have been reflected from suitable surfaces (e.g. sea, ice and ground); and extract useful information about those surfaces. However to fully benefit from real-time characteristic, the overhead between the stringent real-time parallel processing and the storage of amount of multi-channel cross-correlation waveforms(CC-WAVs) prior to downlink issues have been addressed. Over last years, several embedded solutions for parallel processing are ready available: Symmetric Multiprocessing (SMP), Network-On-Chip (NOC). Indeed, higher performance is achieved, but bus congestion and memory allocation issues have increased with these new embedded solutions. This paper presents a novel architecture, namely, the Heterogeneous Transmission and Parallel Computing Platform (HTPCP); And the mandates imposed by the stringent timing constraints by a Dual-Port RAM (DPRAM). The pros and cons of different approaches are discussed, range from providing parallel computing to analyzing bus busy ratio and memory access time. The numerical results show that HTPCP reaches a speed-up of 8.17x in comparison to the SMP architecture, which enables the highest throughput in the real-time system design of GNSS-R.**

***Index Terms*—Global Navigation Satellite System Reflectometry (GNSS-R), GPS open-loop differential real-time receiver (GOLD-RTR), cross-correlation waveform (CC-WAV), Symmetric Multiprocessing (SMP), Network-on-Chip (NOC), Parallel Processing**

I. INTRODUCTION

THE European Space Agency (ESA) proposed a method for extracting altimeter information from GPS signals after their reflection off the sea surface, so-called the Passive Reflectometry and Interferometry System (PARIS) in 1993 [1]. Conceptually, PARIS is a GNSS-Reflectometry (GNSS-R) scenario shown in Fig.2. Since 2001, PARIS has been tested not only from the ground level, but also in aircrafts and even in spacecraft experiments [2]. Therefore a new instrument was designed by ICE (IEEC-CSIC)[1] to compute and store the cross-correlation waveform (CC-WAV) in real time [3]. The device has been called the GPS open-loop differential real-time receiver (GOLD-RTR) instrument shown in Fig. 1. The processing of the CC-WAV over the ocean can be used to obtain geophysical parameters such as sea level and tides, sea surface mean-square slopes, ice roughness and thickness, soil moisture and biomass etc.

Fig. 1: View of the GOLD-RTR instrument

The authors in [6] focus on the sea surface roughness application with GOLD-RTR instrument, enhancing the current understanding of the geophysical content of scattering at L-band. In [3], the authors discuss the GOLD-RTR instrument and its innovative feature of computation and storage in real-time of complex-valued (I and Q) CC-WAV between GPS L1-C/A signals and the reflection signals. The altimeter data was collected by GOLD-RTR instrument, it emphasized all altimeter data must be post-processed to produce accurate surface elevation measurements in ground level. Since the instrument delivers a huge amount of data in real time, corresponding to dramatic quantity of collected waveforms, this is a clear threat to a spaceborne mission where large amount of data must be sent from space to ground or stored in a mass memory. For the GNSS-R technique, the post-processing design in space level has become the de-facto system in remote sensing application. To overcome this threat, our approach is to process the pre-processing data to a space segment with the aim to reduce significantly the data transmission load on the ground level.

Also, [7] describe a bi-static system for ocean altimetry using the GPS signal and develops the corresponding algo-

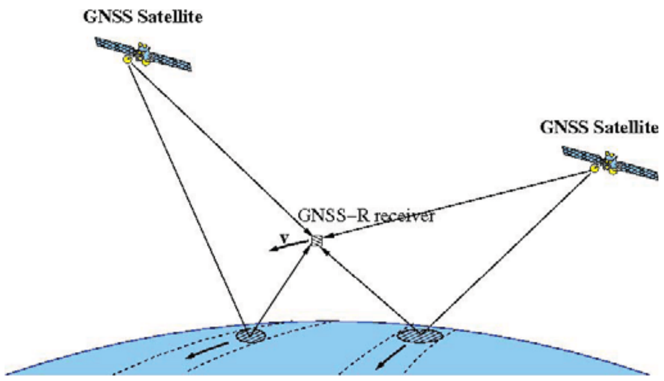

Fig. 2: GNSS-R concept: GNSS signals reflected on the ocean surface are used to gather its properties like roughness or level

rithms to produce real-time altimetric observation. It indicates a method that integrate the waveform for 1s by adding incoherently the different points of the surface from the reflected waveform and compare it with the corresponding direct signal, it could be dramatic reduce satellite to ground transmission link load. However the increasing campaigns of GOLD-RTR has driven the development of CC-WAV post-processing leading to FPGA-based design.

In the past ten years, dynamically reconfigurable FPGAs are well suited to form a parallel architecture since it incorporates several soft-cores. Taking into account time-to-market issues and rapid prototype development, a shared memory Symmetric Multiprocessing (SMP) mounted with embedded OS appears as the first option to explore. However bus-based SMP has limited scalability due to bus congestion and shared memory [9]. With the advent of multi-core processor system, Network-on-Chip (NOC) provides separation architecture between computation and communication. Compared with SMP, NOC enhanced the throughput and scalability, but it is impractical to adapt to ever-changing network protocols and memory allocation of Network processor [10]. The memory allocation is amenable to high throughput and stringent timing constrain application. Benefit on the knowledge of the application, this paper outlines a novel on-board parallel processing architecture, heterogeneous transmission and parallel computing platform (HTPCP), to leverage processing capability and tight design constrain. It has been proposed as a promising technique for GNSS-R post-processing systems.

The rest of the paper is organized as follows. Section II discusses the related work and problem statement; Section III describes our target architecture: HTPCP; Section IV realizes the single correlation integration (SCI) algorithm on target platform and memory allocation. Section V reviews previous SMP architecture, presents the simulation results, outlines the bottleneck, demonstrates speedup ratio. Section VI concludes the paper.

II. Related Work and Problem Statement

In this case, the CC-WAV in the GOLD-RTR is generated by cross-correlating local copies of a global positioning system (GPS) signal with an ocean-reflected GPS signal, over a range of carrier frequencies and code delays. The GOLD-RTR schedules consecutive coherent integration time slots of 1ms over ten correlation channels, with 64 lags each, work simultaneously and continuously with the input raw data sampled at 40 MHz. The total throughput is 10 000 waveforms per second, each waveform being 64 lags long. One waveform has 160Bytes (including 32B packet headers). Therefore, the calculation of the aggregate system throughput of one second would be 12.8Mbps.

In order to overcome the storage of the amount of data prior to downlink, the GNSS-R post-processing design was to address two main problems:

1)Parallelizing the inherent serial output of the GOLD-RTR instrument.

2)Post-processing the multi-channel (I and Q) correlator in parallel.

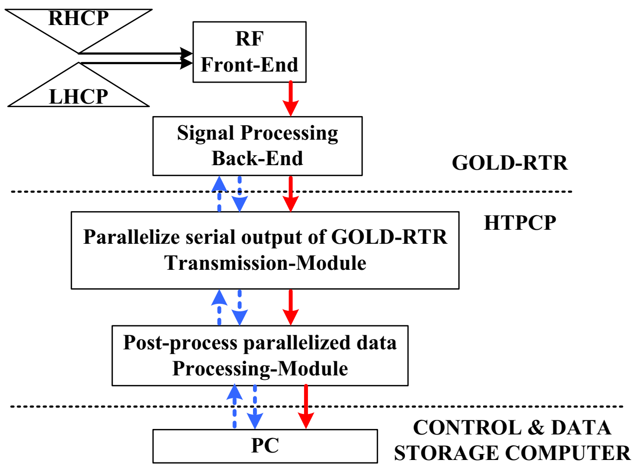
RHCP
LHCP
RF Front-End
Signal Processing Back-End
GOLD-RTR
HTPCP
Parallelize serial output of GOLD-RTR Transmission-Module
Post-process parallelized data Processing-Module
PC
CONTROL & DATA STORAGE COMPUTER

Fig. 3: GOLD-RTR and HTPCP system

As shown in Fig.3, the GOLD-RTR instrument includes two parts: the RF front-end and the signal-processing back-end. As the post-processing system, HTPCP includes two modules: Transmission Module(TM) and Processing Module(PM). The TM is applied to parallelize the inherent serial output of GOLD-RTR; The PM is used to realize multi-channel single correlation integration (SCI) algorithm and hence, to reduce the amount of waveform prior to downlink through the satellite. All these features make it possible to leverage the workload between the communication and computing, and feasibly assign the other applications in the PM in future, for instance, to calculate the sea surface mean-square slopes, ice roughness and thickness, soil moisture and biomass etc.. For a space system, the functions currently performed in the attached PC should be implemented in HTPCP architecture in the future.

III. HTPCP Architecture

The application targeted by this work is the GOLD-RTR instrument depicted in Fig.1. It is composed of two physical devices: a rack that contains the front- and back-end electronics of GOLD-RTR; a PC which provides control and monitoring functions of the rack electronics as well as disk storage for the recorded waveforms. As an intermediate, Fig.4 depicts the interfaces between HTPCP board and these two physical devices. HTPCP board communicates with GOLD-RTR and PC through two full-duplex Ethernet links at 10/100 Mbps. Using two standard unshielded twisted pair (UTP) cables ended with RJ-45 connectors. HTPCP is composed by GR-CPCI-XC4V LEON Compact-PCI Development board plus its extension board GR-CPCI-2ETH-SRAM-8M. The Xilinx Virtex-4 LX200 FPGA on this board is well suited and configurable for LEON3 core implementations.

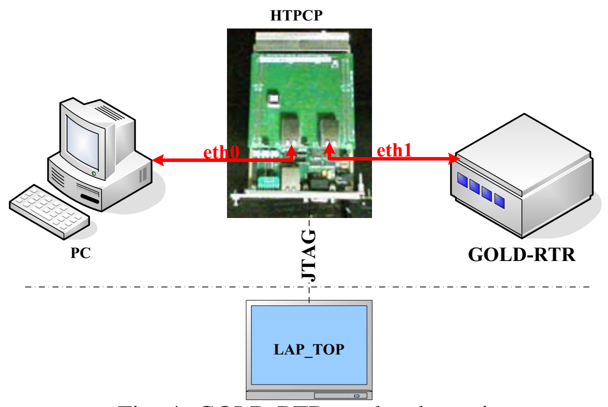

Fig. 4: GOLD-RTR work schematic

In this section we discuss the architecture of HTPCP and the interface with the previous work. Fig.5 illustrates the functional block diagrams of HTPCP, GOLD-RTR and attached PC. As shown, the real-time geophysical parameters transmit from the GOLD-RTR to HTPCP by ethernet interface, and distributed to 8 PM by 2 TM in HTPCP. Each PM processes the consecutive 1 ms waveforms in parallel. Meanwhile each TM accomplishes two tasks: data storage and data distribution. Considered about different function, each module features its private memory and bus, which are accessible from the local processor; the memory hierarchy of TMs and PMs are distinctly; the size of private memory is variant with the size of bandwidth; the software framework can be applied to different stacking approach. The interconnections between TM and PM are done via Fast Simplex Link (FSL), which is composed by a non-latency register-to-memory communication path shown in Fig.12:

- A Dual-Port RAM (DPRAM) connects with 32×32 bits register files by read/write fifos.
- A Finite State Machine (FSM) controls read/write operation of fifos to DPRAM simultaneously.

To fully exploit the potential of FSL, the critical path of TM and PM system doesn't go through the DPRAM and FSM. For instance, the CPU of PM takes 100 clock cycles to calculate a complex result; the FSM control the fifos of FSL to read and write data from DPRAM continuously; the CPU of TM can execute in the meantime a different application code and does not have to wait for FSL to be available. Therefore, the pipeline of reduced instruction set computer (RISC) processor cannot be stalled and the processor frequency will not be decreased.

The interconnections between TM and TM are done via a Dual-Port BRAM and two BRAM controllers, which are connecting to their own buses to realize the interactions between two modules shown in Fig.12:

- The memory controllers are capable of receiving commands coming from remote processor and directing them to the local memory, while sending data coming from local memory and passing them to remote processor.
- The Dual-Port BRAM including two buffers bears the conflicts with the local traffic.

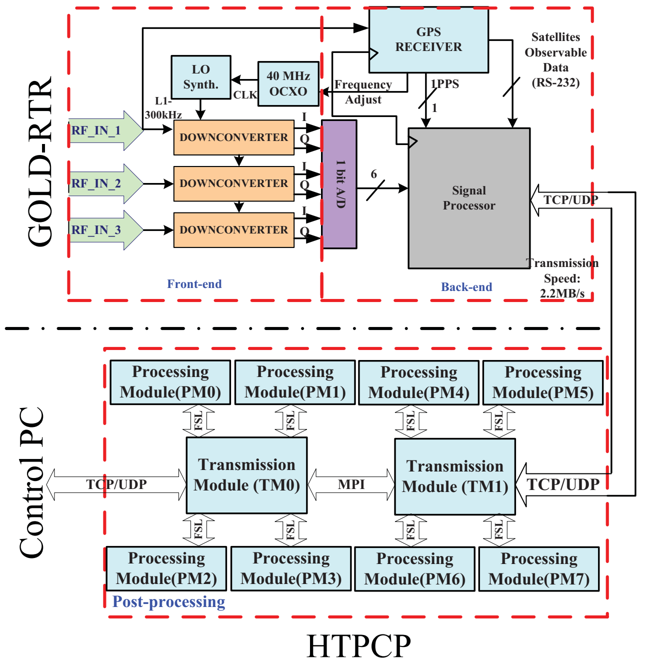

Fig. 5: Functional block diagrams of HTPCP, GOLD-RTR and attached PC

As a sinthesizable 32-bit SPARC V8 processor, LEON3 processor has been developed from 1998 to 2000 as an internal research project by European Space Agency for critical space applications. It is known as the LEON3 open-source processor, and it includes a large range of software development environment(cross compiler,simulator,debugger etc.), as well as support of open-source embedded operating systems (snapgear, uclinux, rtems etc.). Moreover, the fault-tolerant version (LEON3FT) has been designed for operation in the harsh space environment, and includes functionality to detect and correct single-event upset (SEU) errors in all on-chip RAM memories. Therefore, we have chosen the LEON3 to handle the parallel computing in the PM design. Regarding to the interconnect system of the MB and a dedicated FIFO-style connection, called FSL supporting for coprocessors, we propose the Microblaze (MB) processors to sustain the data distribution in TM.

In the following subsection we present the main objective of the proposed module.

A. Transmission Module

The HTPCP has two ethernet ports eth0 and eth1 shown in Fig.8, which are controlled by two Microblaze MB0 and MB1 respectively : eth0 receives the commands sent by PC, and forwards the commands directly to GOLD-RTR by MB1; After receiving the commends, GOLD-RTR transmits the waveforms to HTPCP by eth1, and distributed the waveforms to each PM by MB0 and MB1 in HTPCP. The aim of TM is to solve the following two functions:

1) Forward transparently the control commands and monitoring packets between the attached PC and the GOLD-RTR rack back and forth.

2) Distribute the received waveforms to each PM based on the number of corrector; collect the results from each PM and send them to the attached PC.

Fig.6 depicts the frame of TCP/UDP packets. Each packet contains 5 waveforms. Every packet transmission takes around 45 us. Each waveform contains the number of correlation channels, MB0 and MB1 select each channel's waveforms and transmit them to their own PM. Meanwhile, they collect the results from the PMs and send them to the attached PC every second.

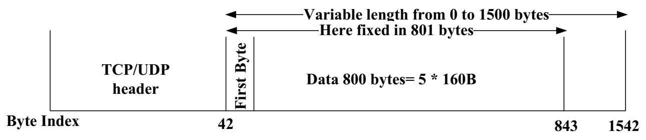

Fig. 6: UDP Transmission Frame

B. Processing Module

As stated in Section II this application has crucial requirements in regard to real-time behavior and performance. As each waveform has the concept space timing parameter, it is neither possible to get the relevant waveform, nor to dump the data. This character is essential for the real-time processing. Unfortunately, timing behavior highly depends on many parameters in hardware/software partitioning, and these parameters are highly mutable during design space exploration make timing behavior is very impracticable. To avoid this, our approach is to apply a DPRAM as Level2 (L2) memory in each PM to fit for different bandwidth; and a seamless design flow is applied, enabling the data entry and data exit. In one hand, it can get rid of the bus busy ratio and reduce the external memory access time; in the other hand, we could process the stream data without external memory, only use the 32KB DPRAM as temporary storage, and send the result to another small memory. The data loss verification could be adjusted by processing the header (32B) of each waveform. The underlying methodology of PM is essential to solve the following problems:

1)Implement the SCI function in seamless design flow.

2)The DPRAM fit for the data bandwidth.

3)The data loss verification could be adjusted by processing the header (32B) of each waveform.

C. SCI Algorithm

In order to exhibit interference of one identical waveform in two continuous time slot. Our approach indicates a method that integrate the waveform for 1s by adding incoherently the different points of the surface from the reflected waveform and compare it with the corresponding direct signal. This method could be dramatic reduce satellite to ground transmission link load. The idea of SCI algorithm for each waveform has two main steps:

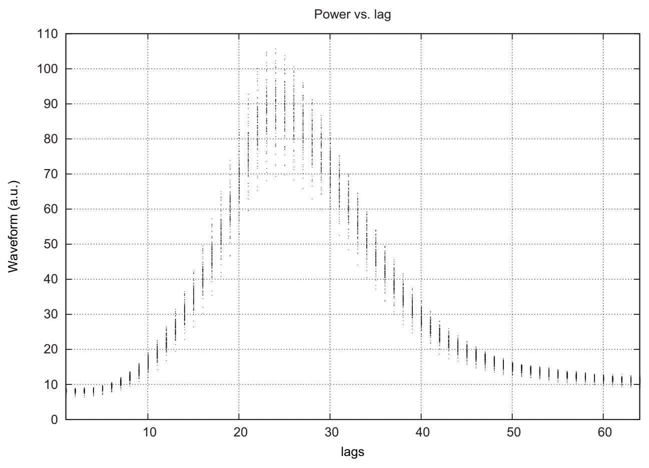

Fig. 7: The coherent integration time is 1 msec; 1 msec include 64 lags; Every 1 sec, we can always get 64 time-varying amplitudes of each lag.

1) To integrate coherently the complex data C quantities during the intervals dt_{coh}

$$C_{\text{coh}}(\text{lag}_i, \text{Cha}_j, t_{\text{coh}}) = \sum C_{\text{coh}}(\text{lag}_i, \text{Cha}_j, T) \qquad (1)$$

where the sum applies to all T within the interval $[t_{\text{coh}} - dt_{\text{coh}}/2$, $t_{\text{coh}} + dt_{\text{coh}}/2]$, if this interval do not contain a possible navigation bit transition.

2) To integrate incoherently the complex data C_{coh} quantities during intervals dt_{incoh}

$$C_{\text{incoh}}(\text{lag}_i, \text{Cha}_j, t_{\text{incoh}}) = \sum |C_{\text{coh}}(\text{lag}_i, \text{Cha}_j, T)|^2 \qquad (2)$$

where the sum applies to all T within the interval $[t_{\text{incoh}} - dt_{\text{incoh}}/2$, $t_{\text{incoh}} + dt_{\text{incoh}}/2]$. In our application dt_{coh} will take one of the values (1 msec, 2 msec, 4 msec, 10 msec, 20 msec) and dt_{incoh}=1 second.

The consecutive coherent integration time slot is 1 msec. Therefore, every 1 sec, we can always receive 64 time-varying amplitude of each time slot shown in Fig.7. The character of each waveform has (every msec) a header of 32 bytes and 128 bytes of complex-valued samples. The 64 complex signal pairs [I(i),Q(i)] are processed and accumulated over 1 sec. The 64 W(i) correspond to 64 lag's amplitude of the complex signal pairs. The SCI function for each channel is the following:

$$\mathrm{W(i)} = \sum_{\mathrm{j=0}}^{\mathrm{N_w-1}} \sqrt{\mathrm{I_i(j)^2 + Q_i(j)^2}} \Big/ \mathrm{N_w} \qquad (3)$$

Where i represents the lag index (0 to 63), j represents a waveform index (0 to 999), and Nw represents number of waveform for each lag (Nw=1000).

IV. SCI ALGORITHM TARGET ON HTPCP PLATFORM

In Section III, we have discussed HTPCP architecture and the interconnects between TM and PM, as well as between

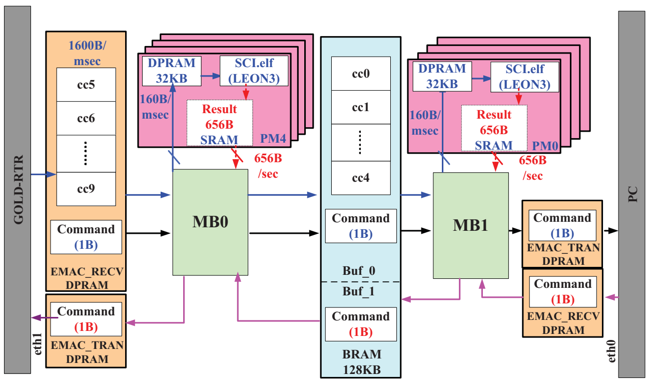

Fig. 8: Data Path in HTPCP

TM and TM. In this Section, we review the bandwidth of the memory system and memory allocation issues that have been addressed in PM, as the main contribution of this work. The parallel architectures, especially multi-processors, have the potential to satisfy the timing performance. However, in most cases, adding more cores does not increase throughput because workloads cannot always take benefit from multiple cores, due to the shared resources (and bandwidth) in the system. Indeed, by into account the characteristic of the final parallel system, the time spent on actual data processing is mostly predictable but the communication and data transfer times are rather dynamic by nature [8]. Thus, we have divided the workload into several PMs and TMs, and reallocated the memory resource for each module to exploit maximize efficiency of parallel computing, without saturating the available bus bandwidth. Thus, the parallel workload solution derived for the target satellite remote sensing application avoids data dependencies from each correlation channels by selective (pre-ordered) transmission. To this end, for each correlation channel, we fix the size of each waveform to 160B in each time slot (1msec), such that each waveform contains space timing parameters (e.g., a fixed 32B header).

A. *System bandwidth optimized by DPRAM*

Fig.8 depicts in detail the data path in HTPCP. The blue arrow displays the data path of the waveforms. The waveforms received by eth1 transmits through the FSL directly to the DPRAM. The throughput degrades from 1600B/msec to 160B/msec, after distributing by MB0. The function of SCI.elf is trigged by the entry address pointer, and generate the result by the end address pointer. The red arrow represents the data path of the back out results. The throughput of results is only 656B/sec per PM. These two pointers of the platform binary file (SCI.elf) are the physical address of the DPRAM. The DPRAM works as a safe threshold to restrict the data bandwidth of the system. Finally, the black and pink arrows correspond to the interactive control command between GOLD-RTR and attached PC.

B. *Memory Hierarchy in PM*

Memory allocation issues are also a design challenge to minimize system latencies. Fig.9 depicts the memory allocation diagram of the LEON3 configuration platform. The

TABLE I: Processing Module design summery

LEON3 memory hierarchy	Memory Size (KB)	No.LUT	No.RAMB16	No. Dual Port RAMs	Max app size
d/iCACHE+DPRAM+ ON-CHIP-MEMORY	2*4+2*16+32 +(128+8)	2370	56	0	128KB
d/iCAHCE+DPRAM+ OFF-CHIP-MEMORY	2*4+2*16+32 +8000	4370	36	392	8MB

Level 1 (L1) memory is the i/dcache to achieve a fast computation; the Level 2 (L2) memory is the DPRAM, which aims at optimizing the system bandwidth and non-latency communication between MB and LEON3; the Level 3 (L3) memory represents an on-chip memory (Block RAM) or off-chip memory (Static ram), which keeps the binary file of the application. Considering the limited available resources on the target FPGA families, Table III shows the processing module design summary, and details the overhead between FPGA resource (RAMB16) and the maximum (MAX) size of the binary file of the application.

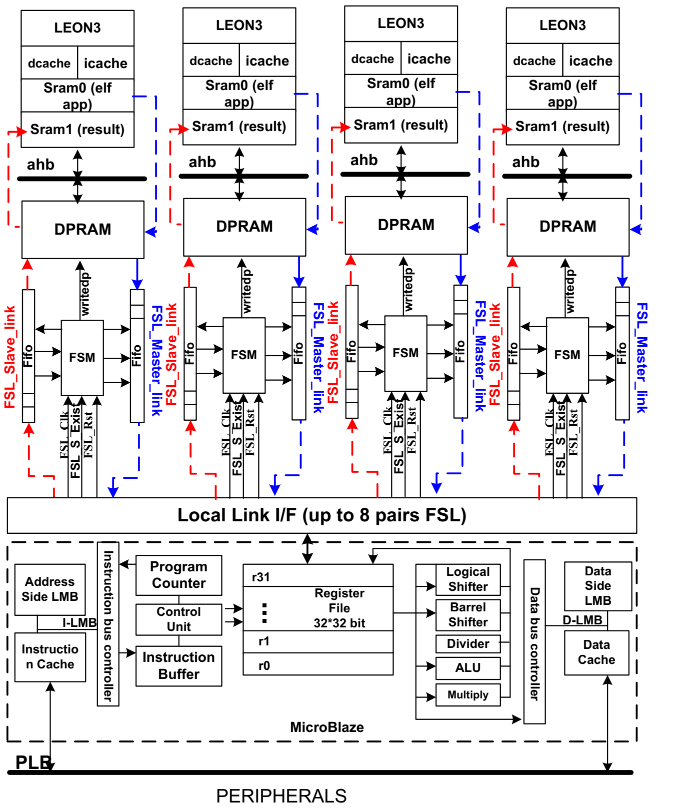

Fig. 9: 4PM + 1TM memory allocation diagram details in Leon3 configuration platform and Microblaze interface

C. *SCI seamless processing flow.*

Fig.10 depicts the SCI seamless processing flow. After resetting the system and initialing the structure of the output waveform, the first waveform is read at the entry address pointer. Then, the system decides if the waveform has received the right correlator number and the valid status. Otherwise, the system moves the pointer up to the stack of DPRAM and

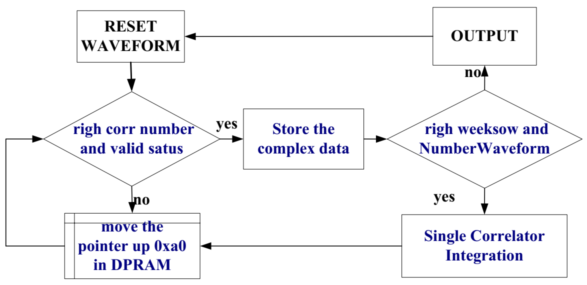
RESET WAVEFORM
OUTPUT
no
righ corr number and valid satus
yes
Store the complex data
righ weeksow and NumberWaveform
no
yes
move the pointer up 0xa0 in DPRAM
Single Correlator Integration

Fig. 10: Single Correlator Integration (SCI) design flow

TABLE II: the speedup parameter for multi-processors.

No. of processors	1	2	3	4	5	6
speed up ratio	1	1.78x	2.23x	2.58x	2.71x	2.7x

checks again. In case the correlator and status are valid, the system stores the complex waveform data of 1 msec, and then decides if this waveform got the right second of week (SOW) and a valid waveform number. In this case, the stored complex data is included in the integration function and the pointer is moved back to the stack of DPRAM in order to decide the next waveform. In case the waveform did not receive a right SOW and a valid waveform number, the system generates the output result and resets the waveform structure to restart.

V. Review SMP Architecture, Simulation tackle on bottleneck , Comparing testing results

In this section we present the evaluation methodology followed by a comparion of timing performance in an SMP architectures versus the proposed HTPCP. We provide the experimental results to analyze the bottlenecks on the proposed architecture. Finally, the numerical simulations are intended to illustrate the parallel computing improvements that the proposed HTPCP can achieve.

A. Review SMP Architecture

Fig.8 depicts a SMP architecture which we proposed in our previous work [9]. The system integrated with multi-LEON3 processor with its own caches and Memory Management Unit (MMU), but shared the main memory. It is easy to build up the higher OS level and SW programming. Task level parallel computing is supported in SMP, the OS scheduler leverage the processes to each processor, and access to the independent data file on the shared memory. SMP has the potential to avoid the data dependency and supply workload fairness principle on each processor. However, adding more CPUs to an SMP does not increase throughput because workloads cannot always take advantage efficiently of multiple CPUs due to the shared resource and scheduling. TableII shows the speedup parameter with the Number of processors.

B. Simulation tackle on bottleneck

Tackle on the bottleneck in Hardware(HW), we implement a series of simulation based on the MPARM and uClinux emulator [4], [5]. Through simulation, we can optimize the

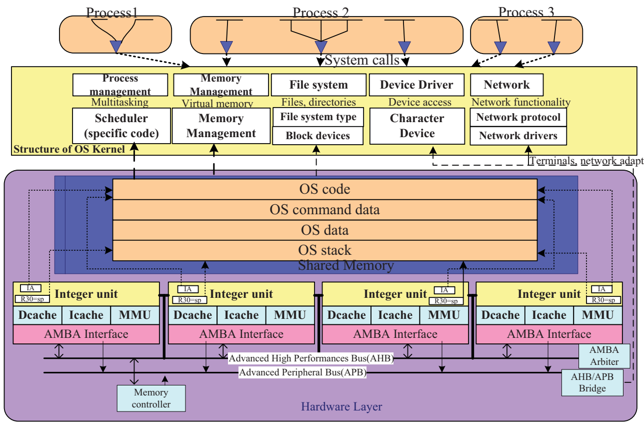

Fig. 11: SMP Architecture

memory hierarchy and get the HW parameters (bus busy ratio, cache size, bus access time and cache hit and miss ratio, computing clock cycle, memory latency etc.). Also accurately measure the overall executive time for our SCI application. TableIII describes the simulation results of different platforms parameters. The simulation are carried out with the following assumptions:

- Cache-memory gap: the gap between the cycle time of processors and data exchange time of cache-memory has continuously increased in SMP, due to the overhead involved in the sharing resource.

- Memory access time and Bus busy ratio: According to the parallel workload analysis [8][9], applying the L2 memory configuration and enables getting rid of the bus busy ratio and memory access time.

The objective of the first simulation is to analyze the L2 memory configuration affects the bus utilization ratio. In this regard, having P1 with L2 memory can imply a lower bus utilization but with a higher execution time. In P2 it is experienced the contrary effect, as we consider in this case that the L3 memory is a private memory, and the bus utilization ratio and memory access time will be critical. Therefore, the final solution needs to have the L2 memory to get rid of the burden of bus transmission.

The objective of the second simulation is to compare how the additional cores affects the bus access time. In this case, the P4 with 4 cores will experience more than four times the bus access time with respect P3 having only 1 core. However, the overall execution time is almost the same. Thus, the multi-core solution can be an efficient solution to enhance the timing features of the final system.

The objective of the third simulation is to compare how the memory access time affects the bus utilization ratio. In this regard, the bus of P5 will be four times busier than P6 because it has a slow external memory. In this case we can use the L2 memory to solve the slow memory access time, if this becomes a problem in the future real-life setup, but it is not required for the time being.

Finally, the objective of the fourth simulation is to compare how the I-cache association affects the I-cache miss rate. After we increase the I-cache association from P7-P9, we found out

TABLE III: Platform parameters

Number of Platform	P1	P2	P3	P4	P5	P6	P7	P8	P9
Number of cores	1	1	1	4	1	1	1	1	1
D-cache	4k*4	4k*4	4k*4	4k*4	4k*4	4k*4	4k*4	4k*4	4k*4
I-cache	8k*2	8k*2	8k*2	8k*2	8k*2	8k*2	8k*1	8k*2	8k*4
D-cache miss (%)	1.10	2.42	1.10	1.10	2.29	2.29	2.29	1.74	2.29
I-cache miss (%)	1.92	0.91	1.92	1.92	0.95	0.95	1.49	0.91	0.95
L2 memory	2MB	NON	2MB	2MB	NON	NON	NON	NON	NON
L3 memory	1MB	1MB	1MB	1MB	1MB	1MB	1MB	1MB	1MB
Overall time(s)	0.407	0.133	0.207	0.24	0.066	0.134	0.143	0.133	0.134
MP ratio	13	13	13	13	2	13	13	13	13
bus access	76128	4482493	38064	159377	4490505	4490562	4645920	4454921	4490562
bus busy(%)	4.34	61.8	2.17	8.50	22.5	61.9	63.9	61.4	61.9

that a 2-set I-cache is the optimal solution for this design. Indeed, we observe that I-cache miss ratio can be an important effect in the overall execution time of our final platform and further exploration about the optimal assignment of data to the multi-layered memory hierarchy is an interesting topic for furture research in this area.

After a series simulations with a multiprocessor emulation platform for the LEON3 architecture [4], [5], we discover the bottleneck is the bus busy ratio and external memory access time. There is also the risk of conflicts between transmission and processing.

C. Analysis testing results

The following testing results are intended to illustrate that the improvement in execution time and the system throughput of HTPCP. We compare these two performances with the convectional SMP. These testings are carried out on the proposed HTPCP (Fig.5) and the convectional SMP (Fig.11), based on the actual clock cycle count system. The benchmark shown in Fig.10 is used for all the testings in this paper. The corresponding throughput equation is shown below:

$$Throughput = \frac{W_s \times N_w}{T_w} \qquad (4)$$

Where W_s defines the size of each waveform; T_w determines the time duration for processing one waveform; N_w defines the number of waveforms that process in parallel. Fig.12

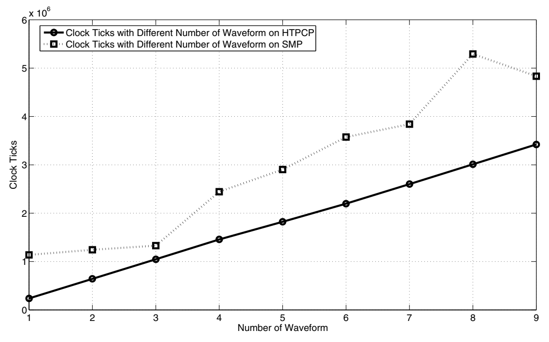

Fig. 12: lag of execution time for each waveform in SMP and HTPCP

shows that the average execution time for each waveform can improve 8.17 times comparing with the conventional SMP, and it overcomes the shortcomings of fluctuation in SMP which is effected by the shared resource. The throughput of HTPCP has improved by 2 orders of magnitude compared to SMP.

VI. Conclusions

In this paper, a novel parallel platform, named as HTPCP, is presented in order to realize the realtime post-processing for GNSS-R. Moreover, two problems for HTPCP are proposed and solved, 1) Parallelize the inherent serial output of the GOLD-RTR instrument and 2)Post-processing the multi-channel (I and Q) correlators in parallel. In order to evaluate the performance of this novel platform, we compare the testing results with SMP platform. The results show that the parallel computing speed of HTPCP outperforms SMP. Specially, the numerical results indicate that the system throughput can reach up to 12.8Mbps. The the average execution time for each waveform can improve 8.17 times comparing with the conventional SMP. Finally, we conclude that the proposed platform is very effective in reducing memory access time and bus busy ratio and hence, is compatible for GOLD-RTR instrument.

Acknowledgment

The research in this paper is partly supported by the Spain National Space research Project ESP2005-03310, with participants ICE (IEEC-CSIC). Also, this research has been partly supported by the Swiss SNF Grant number 200021-127282. Finally, the authors would like to thank Lei Jiang for the interesting discussions to improve the paper.

References

[1] Martín-Neria,M. ”A Passive Reflectometry and Interferometry System(PARIS):application to ocean altimetry”, *ESA Journal*, 1993,VOL. 17, pp.331-355.

[2] Martín-Neira M., Caparrini M., Font-RossellóJ., Lannelongue S., Vallmitjana C. S. ”The PARIS concept: an experimental demonstrationof sea surface altimetry using GPS reflected signals”, *IEEE Transactions on Geoscience and Remote Sensing*, 2001 Jan, VOL. 39, pp.142-149.

[3] Nogués-Correig O., Cardellach-GalíE., Sanz-Camderrós J., Rius A., ”A GPS-Reflections Receiver That Computes Doppler-Delay Maps in Real Time”, *IEEE Transactions on Geoscience and Remote sensing*, VOL. 45, Issue 1, January 2007, pp.156-174.

[4] Atienza D., De Micheli G., Benini L., Ayala J.L., Garcia Del Valle P., Debole M., Narayanan V., ”Reliability-Aware Design for Nanometer-Scale Devices”, *Proc. of ASP-DAC*, January 2008, pp. 549554.

[5] Mulas F., Atienza D., Acquaviva A., Carta S., Benini L., De Micheli G., ”Thermal Balancing Policy for Multiprocessor Stream Computing Platforms”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, VOL. 28, Issue 12, December 2009, pp. 1870-1882.

[6] Estel Cardellach, Antonio Rius, ”A new technique to sense non-Gaussian features of the sea surface from L-band bi-static GNSS reflecitons”, *Journal of Remote Sensing of Environment 112(2008)* pp.2927 - 2937

[7] Antonio Rius, Estel Cardellach, Manuel Martin Neria, ”Altimetric Analysis of the Sea-Surface GPS-Reflected Signals”, *IEEE Transactions on Geoscience and Remote Sensing* pp.2927 - 2937

[8] Kreku J., PenttiJ., Kangas J., Juha-Pekka S. ”Workload simulation method for evaluation of application feasibility in a mobile multiprocessor platform”, *Proceedings of the EUROMICRO Systems on Digital System Design (DSD’04)*, September 2004, pp. 532-539

[9] Guo Yi, Lena Kanellou, Luis Andrés Cardona, Antonio Rius, Carles Ferrer ”Parallel workload analysis in SMP platform: a new modelling approach to infer the HW efficiency for remote sensing application”, *Proceedings of SPIE 2009*, VLSI Circuits and Systems IV, 28 May 2009, Proceedings Vol. 7363.

[10] Shuai Mu, Xinya Zhang, Nairen Zhang, Jiaxin Lu, Yangdong Steve Deng, Shu Zhang ”IP Routing Processing with Graphic Processors”, *Proceedings of DATE 2010*, 8 March, 2010, pp. 93-99