XSTREAM: A Highly Efficient High Speed Real-time Satellite Data Acquisition and Processing System using Heterogeneous Computing

Pramod Kumar K[a], Mahendra P[a], Ramakrishna Reddy V[a], Tirupathi T[a], AkilanA[a], Usha Devi R[a], Anuradha R[a], Ravi N[a],S.S.Solanki[a], AcharyKK[b], AL Sateesha[b] and AnshuChauhan[b]

[a]Advanced Data Processing Research Institute (ADRIN), Department Of Space, Govt. of India,203-Akbar Road, Manovikas Nagar Secunderabad-500009 - (pramod,mahendra, rkreddy, tiru, akilan, usha,anuradha, nooka.ravi, solanki)@adrin.res.in
[b]ISRO Satellite Center (ISAC), Indian Space Research Organization, Dept. of Space, Govt. of India, Old Airport Road,Bangalore-17 - (kkachary, alsatish, anshu)@isac.gov.in

KEY WORDS: Heterogeneous computing, Hybrid Computing, Satellite Data Processing, Real-time Processing, IRS Cartosat, GP/GPU, FPGA, OpenCL

ABSTRACT:

In the last decade, the remote sensing community has observed a significant growth in number of satellites, sensors and their resolutions, thereby increasing the volume of data to be processed each day. Satellite data processing is a complex and time consuming activity. It consists of various tasks, such as decode, decrypt, decompress, radiometric normalization, stagger corrections, ephemeris data processing for geometric corrections etc., and finally writing of the product in the form of an image file. Each task in the processing chain is sequential in nature and has different computing needs. Conventionally the processes are cascaded in a well organized workflow to produce the data products, which are executed on general purpose high-end servers / workstations in an offline mode. Hence, these systems are considered to be ineffective for real-time applications that require quick response and just-in-time decision making such as disaster management, home land security and so on..

This paper discusses anovel approach to processthe data online (as the data is being acquired) using a heterogeneous computing platform namely XSTREAM which has COTS hardware of CPUs, GPUs and FPGA. This paper focuses on the process architecture, re-engineering aspects and mapping of tasks to the right computing devicewithin the XSTREAM system, which makes it an ideal cost-effective platform for acquiring, processing satellite payload data in real-time and displaying the products in original resolution for quick response. The system has been tested for IRS CARTOSAT and RESOURCESAT series of satellites which have maximum data downlink speed of 210Mbps.

1. INTRODUCTION

Remote Sensing (RS) community has observed a significant growth in number of satellites, sensors and their resolutions, thus causing an exponential growth in the volume of data to for processing each day during last decade. On the other hand there has also been a growing demand to cut down the lead time for generating the products that need to be used by different applications, such as disaster management, home land security, etc. that demand quick response; therefore the data products need to be made available for these applications in the quickest possible time. Conventional data processing [1][2]that involves chains, though well proven, time tested and rugged, are considered to be ineffective for real-time applications that require quick response and just-in-time decision making. The examples of processing includes Decoding, Decryption, Decompression, Radiometric Normalization, Geo-coordinate computation and tagging the corner coordinates, and finally writing of the product onto the disk to make a Level-1A product. Additionally, for some of the missions such as Cartosat series of satellites, stagger estimation and stagger correction are also done before writing of the product onto the disk.

IRS Satellites such as IRS Cartosat-1[3],Cartosat-2[4], Resourcesat[5] capture images of ground and transmit the image data (also referred to as video data) after performing various on-board processing such as compression, encryption, error correction coding etc. Along with the image data, data pertaining to various sensors on-board, such as Star Sensor, Gyros, clock, health parameters etc., which are referred to as auxiliary data (ephemeris data or AUX data) are also multiplexed with video data and transmitted to ground stations. Video data is processed to produce viewable image, while aux data is used to estimate and tag the geo-location of the ground features in the final image product. The data is transmitted in the form of fixed frames by attaching a pattern called Frame Sequence Code (FSC) to identify the valid data on ground. Ground systems, acquire the data and process the payload data (video and aux) to generate level-1A product [6]. Cartosat-1 has two streams with each stream having I& Q channels of 52.5Mbps data rate, while Cartosat-2 downlinks the data in one stream having I & Q channels of 52.5Mbps each. The data is transmitted using X-band Carrier Radio Frequency (RF) communication system. The data is sent using X-band Carrier Radio Frequency (RF) communication system. RF data receiving and conversion to Intermediate Frequency (IF) at the ground stations etc., are standard equipments and are always done in real-time hence are not discussed in this paper.

Satellite payload data processing, to generate level-1A product from raw signal data received from the De-mod output, involves certain complex processes performed sequentially in the reverse order of on-board processing. The processes are similar to that of for most of the remote sensing satellites, with variation in mission specific formats and processes. For example, some missions do not compress or encrypt the data and similarly a different method may be adopted for encoding, encryption, or compression.

Figure-1 explains the conventional ground system data acquisition and processing for IRS Cartosat-1[1]and similar methodology is also adopted for Carosat-2 with mission specific variations. The system employs organized workflow to produce the usable data products, which are executed on general purpose high-end server(s)or workstation(s) in an offline mode (i.e., record the data to files, process them by reading the data from the files and writing them back on to the files on the hard disk).Therefore they consume a lot of time, computing resources and are deemed to be sub-optimal and ineffective for applications that demand quick response.

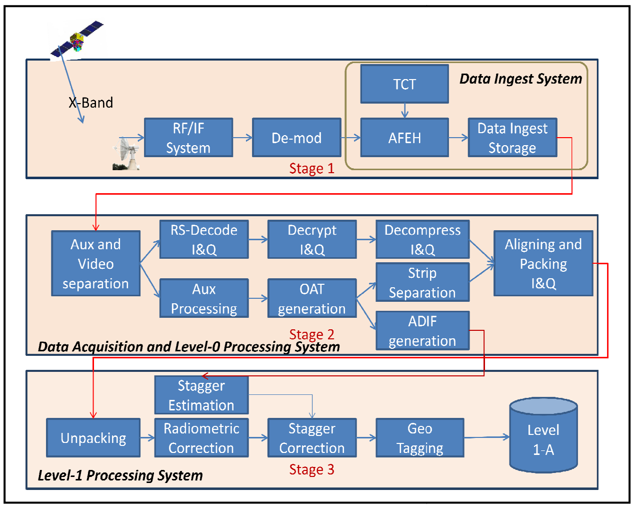


Figure 1. Ground System Processing to generate Level-1A Product for IRS Cartosat-1/2

In the recent past, the computing paradigm has taken a radical shift from usual sequential processing to parallel/multi-core architectures duetothe clock speeds hitting the limits. Today, all the major processor vendors manufacture with around ten cores in a single CPU. GP/GPU (General Purpose Graphics Processing Unit) is the latest addition in the High Performance Computing (HPC) domain. GPU is an accelerator card which contains very large number of cores, specifically designed to perform Single Instruction Multiple Data (SIMD) type of operations. Similarly, Field Programmable Gate Arrays (FPGA) are special electronic chips containing large number of Gates and Common Logic Blocks (CLB) which can be programmed to behave like an electronic circuit, thus can achieve very high performance by consuming less power. Digital Signal Processors (DSPs) which are inherently parallel and are designed for specific processesand perform these tasks very fast.

On the other hand there has also been a significant progress in standardization of interfaces and programming language Interfaces. PCIe[7] has emerged as de-facto interface for many of the add-on accelerator cards, FPGA cards, and GP/GPU cards. OpenCL[8] and OpenMP[9] have emerged as an ideal programming interface which supports all multi-core architectures and are vendor neutral. All these developments have given system engineers greater flexibility to chose the right platform for a given task and integrate them into a single system, single instance of OS and a single program.

Heterogeneous computing or sometimes also referred to as Hybrid Computing [10] platforms integrate various computing platforms such as CPUs, FPGAs and GPUs into a single system which can perform complex tasks/workflows in parallel and achieve higher throughput.

[11]demonstrated the usage of Metacomputing by utilizing multiple resources and high speed network for real-time processing and visualization of existing remote sensing data, but the process does not address the real-time pass acquisition. [12] discusses a requirement of storing the real-time data into non-real-time database using file buffers. It does not discuss elements of payload data processing. [13][14] demonstrated the use of Heterogeneous Computing using GP/GPUs for achieving high throughput for different domains.

This paper discusses a novel approach of acquiring and processing Remote Sensing Data in real-time (as the data is being received) using a heterogeneous computing platform namely XSTREAM having a combination of CPUs, GPUs and FPGA.

Section 2 discusses the methodology adopted, re-engineering aspects, mapping of the tasks to the right computing device and synchronization of all the modulesin order to generate the data products and display them in original resolution in real time (while the pass is being acquired and processed). Section 3 discusses the test results and comparison with the conventional data processing chain in terms of quality and accuracies. Section 4 discusses the conclusion and future directions.

2. METHODOLOGY

2.1 Level-1A product generation process:

Figure 1 depicts the sequence of steps to be performed for generation of level-1A product (also referred to as basic product for any analysis) from raw signal data and a brief description of nature of the problem is listed below. Please note that due to staggered placement of the CCD arrays in Cartosat missions, odd and even pixels are recorded, processed in separate streams and sent as I & Q channels. Hence the following processes need to be performed for two streams separately (step (i) to step (ix))and finally combined to generate a level-1A product ((step (x) to step (xiii)).

i. *Detection of Valid Frames:* Read the bit stream and search for frame header i.e., FSC code and store the valid frames.
ii. *Time Stamping:* Time stamp each frame with the Ground Reception Time (GRT), which is read from Time Code Translator (TCT)
iii. *Aux Separation:*Separate the Aux data from the video data and process aux data and video data independently.
iv. *Decode:*Decoding is done, which is a reverse process of encoding.In Cartosatmissions standard 255/247 Reed-Solomonencoding method is used, which can correct 4-byte errors in 247 bytes.
v. *Decrypt:* Decrypt the data based on the encryption methodology used in the mission. The methodology used is stream cipher and the details are out of the scope of this paper.
vi. *Decompress:* Most of the high resolution RS satellites use lossy compression techniques to match the data downlink rates vis-a-vis data acquisition rates. These compressions are data dependent and produce variable length bit stream after compression (JPEG, DPCM, JPEG2000 etc). In Cartosat series,JPEG[16] like compression,augmented with a rate control algorithm and Huffman coding is used. This process involves searching of header, followed by decoding the bit-stream, de-quantization and inverse transformation.
vii. *Aux Processing:* The star sensor data, gyro data, on-board time etc (also known as or ephemeris data) are used to construct orbit and attitude information (OAT) with respect to the imaging time. These are used in computing the geo-location (latitude and longitude also known as lat/longs) and tagging the image line with the computed lat/longs.

viii. *Stagger Estimation:* Stagger value is specific to missions having staggered placement of CCD arrays like Cartosat satellites. The stagger value depends on the satellite look direction (Roll bias), and hence it changes for each scene. This requires complex geo-processing using OAT values.
ix. *Radiometric normalization:*Due to the non-linear nature of CCD devices, a gray level normalization process is to be performed. Often this process involves passing through a LookUp Table (LUT) for every pixel. LUT is pre-computed table based on lab settings, and is also routinely updated using a calibration test site.

For steps (i)-(ix), the data is processed in two independent streams I & Q in parallel, which have odd and even images separately without attaching the geo co-ordinates to the data. Further processeslisted through steps (x)-(xiii) are performed to combine and align the data to generate the basic level-1A product.

x. *I&Q channel alignment*: Since the data is processed from two independent chains there is a possibility of data mis-alignment due to any reasons, such as, loss of frames, bit-errors,process exception in one of the channels I/Q etc. Hence the data from both I & Q channels are to be aligned using the time tag and line count extractedfrom the video data.
xi. *Stagger Correction:* As explained in step-(viii), due to staggered placement of odd pixels and even pixels in the focal plane one of the images need to be shifted (resample) by a value computed in step (viii) to generate a stagger freefull swath original resolution image.
xii. *Strip Separation and Geo Tagging:*For missions like Cartosat-2 (which involves manoeuvring between two successive spot acquisitions) the data may contain manoeuvre data during of RT acquisition and in case of PB the strips are streamed continuously without any manoeuvre data. Hence the process involves separating the scenes and writing the separatedstrips as independent images. Aux processed values OAT is processed and stored as Auxiliary Data Interchange Format (ADIF) and are used to compute the geo-coordinates of the scene. The Geo-coordinatesat regular intervals and are stored in a separate file (grd file) and is used in subsequent processing, such as level-2 processing andlat/long readout in display process etc.
xiii. *Geo-Image generation:*The image thus generated in step (xi) and geo coordinates computed at regular intervals in step (xii) are combined together and written in a native format as a level-1A product (the design is flexible and can be written in any other open format like HDF5[15]).

2.2 Analysis& Design:

This section discusses the nature of processing involved for each of the task/step mentioned in section 2.1 and provides the reasoning for mapping them to the chosen device. Summary of the same is shown in Table 1.

The de-mod provides digital data to the system.The satellite transmission starts slightlyahead of the actual imaging data transmission.Hence to detect valid data a fixed code i.e., FSC is attached to every valid frame. The FSC code needs to be searched, for obtaining the valid data in the ground processing systems. Time stamping on each frame is also required to detect the frame losses during the transmission (if any) and replace them with interpolated values if possible. Since the nature of these tasks is search and replace in the bit stream obtained from de-mod, it is ideal to perform these tasks on an FPGA. Hence a COTS FPGA based card is used which takes the Low Voltage Differential Signal (LVDS) input data and writes the valid frames onto main memory of the system. The card sits on a PCI-X bus and also provides a logic to stamp the time through IRIG-B interface,which can be used to correct the frame losses, wherever possible.

In Cartosat series of missions, the aux data is added to video data by the data formatter on-board. Hence the aux data in each valid frame needs to be separated from video data. Aux separationinvolvesextraction of fixed number of bytes from valid frame andcan be easily done on FPGA as well as on CPU. In some of the missions aux data is also encoded. Hence to make the process generic and flexible,the aux separation task is mapped to CPU in the proposed method.

The next step in the process chain is RS-Decoding. RS-Encoding[17] is a common procedure adopted in satellite communications for correcting communication bit errors.RS-Encoding is a fixed block 255/247 encoding, the decoding process is iterative in nature and also to balance the load the on GPU and CPU, CPU implementation is chosen using multiple threads mapped to multi-core.

After decoding process, the next step is decryption. In Cartosat series of satellites stream cipher encryption is used. Decryption of stream cipher text involves simple XOR operations, which can be done on FPGA or GPU. Since the data is already available in the main memory of the host, noting that the process is a light weight process,further the next step needs to be done on CPU, the task is mapped to CPU to avoid multiple data transfers.

As explained in previous section that the compression results in a variable length code, a marker is attached to every compressed block. The decompression process involves searching of marker on CPU, and sent to GPU. Additionally, to address frame loss or irrecoverable errors in one of the channels (I or Q) the blocks are aligned and sent to GPU for decompression. The decompression is executed in block parallel way on GPU which is more optimal than that of CPU.

Aux processing involves complex modelling,which uses the ephemeris (AUX data), and iterative computations. This process is independent of video processing, hence, can easily be spawned as a separate dedicated thread on CPU. This module also performs stagger estimation and identifies the strip based on the time tag.

The next step in the process chain after the decompression is radiometric normalization,which is a pixel based operations.Pixel based operations falls in the category of SIMD, for which GPU architecture is ideally suited.After the radiometric normalization the images from I & Q (odd and even) need to be combined to generate full swath image. The stagger correction process is also applied to optimize the process flow while combining the data. Since stagger correctioninvolves re-sampling, which is again a pixel based operation,it is also mapped to GPU.

Design

Since all the steps need to be performed in sequential mode a ‘pipelined architecture’ as shown in the figure-2 is adopted to achieve real-time processing. Three way pipelineapproach is adopted, in which each stage is performed on a different computing device. Parallel pipeline architecture requires buffer based processing. Hence in XSTREAM, based on constraints of all the processes involved, block size is fixed to three seconds of input data.

Stage / Step No.	Function	Mapped on to	Sequential /Parallel
Stage 1 i ii	FSC Logic Valid Frame Detection Time Stamping	FPGA FPGA	Sequential Sequential
Stage 2 iii iv	Decode Aux Separation RS-Decoding	 CPU CPU	 Sequential Block level Data Parallel
Stage 2 vi	Decompression Marker Search	 CPU	 Sequential
Stage 3 vi	Code block decoding	GPU	Block level Data Parallel
Stage 2 vii	Aux Processing	CPU	Sequential and iterative
Stage 2 viii	Stagger Estimation OAT based / Image Based	 CPU / GPU	 Sequential
Stage 3 ix	Radiometric Correction	GPU	Data Parallel
Stage 2 x	I and Q Channel Misalignment due to frame loss	CPU	Sequential
Stage 3 xi	Stagger Correction	GPU	Data Parallel
Stage 2 xii, xiii	Geo Tagging and File creation	CPU	Sequential

Table 1. Mapping of tasks to computing devices

The first stage, which is mapped to a FPGA acquires the data, does FSC validation and time stamping. The result is passed to the second stage, which is mapped to CPU, where decoding, decompression and block identification tasks are performed. Additionally, second stage also performs aux processing, which is completely an independent task and can be executed on CPU. The result of second stage viz., decompression blocks and discrete stagger values, are passed to GPU, which is the third stage of the pipelined architecture, proposed in XSTREAM software. The GPU performs decompression, radiometric normalization and stagger corrections as three independent kernels. The result of the third stage is a full swath and full resolution image and is sent back to the host system. The GPU uses a high speed PCIex16 bus for communicating with the host. Finally, the geo-tagging and writing to a file is done in a separate dedicated thread with asynchronous IO. Figure 2 depicts the three way pipeline architecture of XSTREAM.

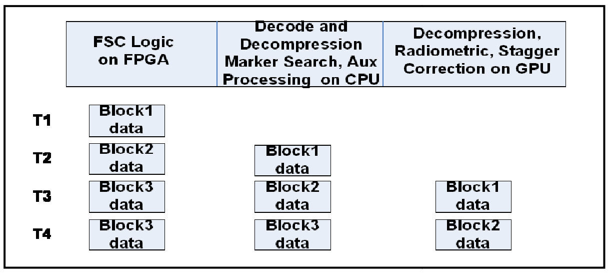
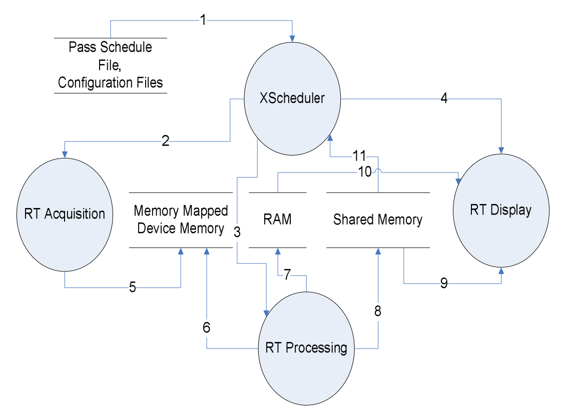


Figure 2. Three way pipeline processing adopted in XSTREAM

2.3 XSTREAM System and Process architecture

Based on the analysis the XSTREAM softwareis designed as*four*separate processes, namely ‘Real-Time(RT) stream Acquisition’, ‘RT Data processing’, ‘RT Full resolution scroll display’ and ‘XScheduler’.This is to modularize and arrest the propagation of failure of one module to others. Every process is engineered such that it can work independently.
Real-Time Acquisition module is responsible for acquiring the data and writing the valid frames into host memory area.The data can also be flushed to disk in real-time or flushed after the pass for analysis and playback support. This procedure is adopted to providefail-safe option in case of failure in RT Processing.
The ‘RT Data processing’ module is responsible for processing the data in blocks and in real-time. Processing includes all the steps as discussed in section 2.2.
‘RT Full resolution Display’ displays the processed Level-1A products on a multi-screen display. The display also provides minimal navigational aids such as scroll speed adjustments, jump to arbitrary location within the strip and minimal enhancements, such as contrast stretch.This process can also be used in offline mode to view already processed strips.
The ‘XScheduler’ controls all the processes and displays the status information, error messages, alerts requiring immediate action, logs etc., on a GUI. All the real-time processes are controlledby XScheduler. One of the main tasks of scheduler is to handle clash scenario by providing pre-emptive priority based scheduling as well as manual override options.XScheduler polls for the pass schedules of each mission and updates the process schedule accordingly.

Figure 3 shows the DFD and process architecture of XSTREAM software. The hardware block diagram of the host system is shown in Figure 4.



1) Pass Information form Pass Schedule file, Pass Configurations including pass priority etc to XScheduler
2) Configuration Parameters from Scheduler to Acquisition Process to start acquisition
3) Pass Information from Scheduler to Preprocessing
4) Display configuration from Scheduler to RT Display
5) Acquired data from acquisition cards to Dedicated Memory
6) Raw data from Dedicated memory to RT Preprocessing module for preprocessing
7) Processed data to RAM, Memory mapped file
8) No lines processed, No of strips processed to Shared Memory
9) No lines to display from shared memory to RT Display
10) Processed data from RAM to Display
11) No of lines processed from shared memory to X Scheduler for status updation

Figure 3. DFD-0 of XSTREAM Software modules

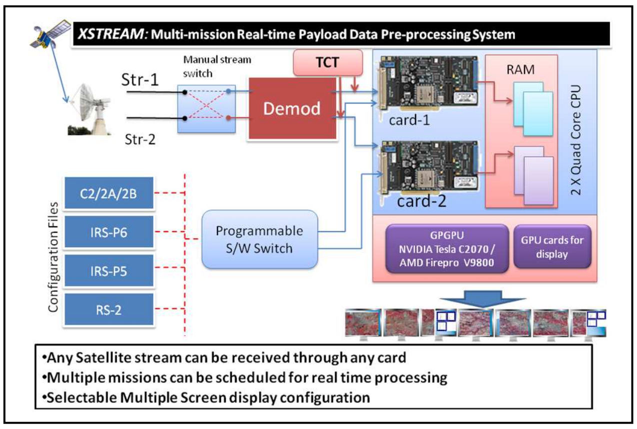

XSTREAM: Multi-mission Real-time Payload Data Pre-processing System
Manual stream switch
Str-1
Str-2
TCT
Demod
card-1
card-2
RAM
2 X Quad Core CPU
Configuration Files
C2/2A/2B
IRS-P6
IRS-P5
RS-2
Programmable S/W Switch
GPGPU NVIDIA Tesla C2070 / AMD Firepro V9800
GPU cards for display
•Any Satellite stream can be received through any card
•Multiple missions can be scheduled for real time processing
•Selectable Multiple Screen display configuration

Figure 4. Hardware block diagram of XSTREAM system

2.4 XSTREAM Software Implementation Specifications

The entire software is built on COTS hardware and Open platforms. Standard Linux Operating system with ANSI C/C++ language and gcc [18] compiler is used. On host side POSIX threads are used for exploiting all the cores in CPU, while OpenCL API is used for GP/GPU programming. OpenCL implementation is chosen because it is independent of the GPU provider and provides portability across all the computing devices viz., CPU, GPU, etc. OpenSceneGraph[19] is used for Real-time rendering of processed images on multi-screen display. The Scheduler module is developed using Qt [20] User Interface (UI) framework. Qt toolkit is lightweight and is an Open source project. For synchronize between all the modules within XSTREAM, standard POSIX inter process communication (IPC) mechanisms are adopted.

3. TEST RESULTS

The XSTREAM host systemhaving the configuration as listed in Table-2 is used for testing. The system was connected to COTS demodulator of M/S Cortex make at NRSC-Shadnagar. The COTS High Speed Data Acquisition Card supplied by M/S Apollo Microsystems[21], Hyderabad, was used to acquire data from the de-mod and was connected to PCI-X bus within the host system. The high speed acquisition card also had IRIG-B interface to stamp time on the valid frames.

CPU	2 x 12-core
Main Memory	32 GB DDR3
Operating System	RHEL 6.2
GP-GPU for Compute	NvidiaTesla-C2070 / AMDv9800
COTS Hardware	Altera Stratix II FPGA
GPU for Graphics Display	2XnVidia Quadro6000 / 1xAMD v9800

Table-2: Hardware configuration of XSTREAM Host

The system was initially tested with Cartosat-2/2A/2B passes for all modes, such as Real-Time (RT), Solid State Recorder (SSR) Playback (PB) and mixed modes i.e., RT and PB. Later the software, with mission specific modifications, was also tested for Cartosat-1 and Resourcesat-2 missions. The system was continuously tested for more than a year and specific evaluations for the following data sets were performed to study the radiometric quality and geometric accuracies.

Test– 1:
Dataset: Cartosat-2A data, of orbit 18946 with a pass duration of 9 minutes 13 seconds.

The results of both processesi.e., conventional processing and XSTREAM processingare compared below.

No of Strips			Latency		Geometric Accuracy	
Intended	Sys-I	Sys-II	Sys-I	Sys-II	Sys-I	Sys-II
6	6	6	~2 hrs	~6 secs	100 mts	150 mts

Test - 2:
Dataset: Cartosat-2B data, of orbit number 68 with a pass duration of 10 minutes.

No of Strips within the pass			Latency		Geometric Accuracy	
Intended	Sys-I	Sys-II	Sys-I	Sys-II	Sys-I	Sys-II
16	16	16	~2 hrs	~6 secs	100 mts	130 mts

Sys-I as mentioned in the above table, is the conventional system, which consists of two Itanium based servers one performs data acquisition and Level-0 processing, while the second server performs the Level-1A processing in offline mode. The Sys-II is the XSTREAM host system having the configuration as shown in Table 2.

The radiometric quality was visually verified and found to be similar for both the products. Similarlythe Level-2 products generated from both chains showed similar accuracies.

Sample Outputs:

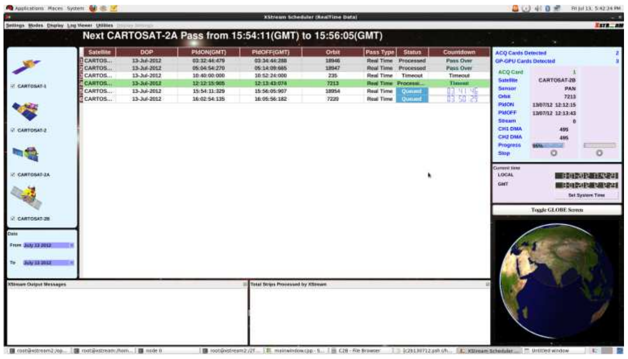

Xstream Scheduler (RealTime Data)
Next CARTOSAT-2A Pass from 15:54:11(GMT) to 15:56:05(GMT)
CARTOSAT-1
CARTOSAT-2
CARTOSAT-2A
CARTOSAT-2B
XStream Output Messages
Total Strips Processed by XStream
Toggle GLOBE Screen

Figure 5.XScheduler displaying the list of passes scheduled and RT Processing stage

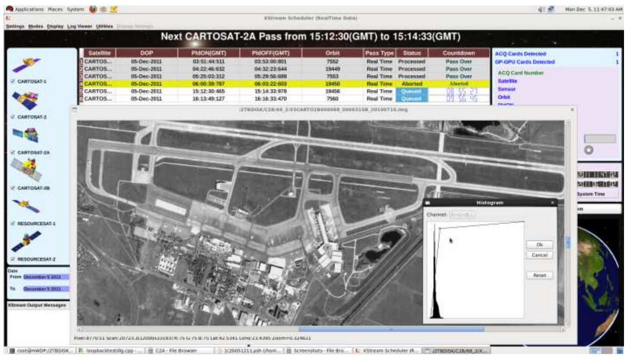

Xstream Scheduler (RealTime Data)
Next CARTOSAT-2A Pass from 15:12:30(GMT) to 15:14:33(GMT)
CARTOSAT-1
CARTOSAT-2
CARTOSAT-2A
CARTOSAT-2B
RESOURCESAT-1
RESOURCESAT-2
Histogram
Ok
Cancel
Reset

Figure 6.XScheduler displaying the processes output

4. CONCLUSIONS

Satellite payload data processing to generate level-1A product involves sequence of complex multiple tasks that need to be executed one after the other. Conventionally they are done on high server(s) in a sequential mode which is not only sub-optimal but also introduces substantial delays in actual usage of the data, thereby is not suitable for applications needing quick response. This work presents generation of level-1A products in real-time using a single heterogeneous computing platform namely XSTREAM having a combination of FPGA, GP/GPU and multi-core CPUs. The approach adopted involves re-engineering of all the software modules and careful orchestration among all computing entities within a single system and a single process.

The XSTREAM system is tested for Cartosat series of satellites having data rates of 105Mbps (2 (I&Q) X 52.5Mbps) and 210Mbps (4 X 52.5Mbps) downlink. All operational modes such as RT, PB, and SSR were tested and the products were generated in real-time (during the live-pass) with latencies between 6-10 seconds, by using 2 Socket CPU with 8 core each, along with one GP/GPU card per channel (I&Q) of data. The radiometric quality of both the products i.e., the product generated through normal offline mode and through XSTREAM appeared similar, while the geometric accuracies for generation level-1B was slightly inferior as compared to that of the offline processing. However, the level-2 product accuracy again showed similar results for the conventional offline processing as well as for on-line processing through XSTREAM.

The system is mission neutral, highly modular and independently scalable to support any future missions such as Cartosat-3 which has higher data rates. Future work is also oriented towards processing of SAR missions, such as RISAT-1 in which the challenge would be not only the high data rates but also signal processing. Another direction of work is to generate level-2 products in real-time with automatic identification of GCPs.

5. ACKNOWLEDGEMENTS

Authors would like to thank Director ADRIN, Director ISAC and Group Director-SEG-ADRIN for their continuous support and valuable guidance during the entire process of development of XSTREAM system.

The authors would also like to acknowledge and place on record the kind cooperation and support extended by Director NRSC, Scientist/Engineers, scientific staff and administration of NRSC for providing the necessary support to carryout developments and testing at NRSC Earth station Shadnagar.

REFERENCES

[1]http://www.nrsc.gov.in/pdf/hcartosat1.pdf : Cartosat-1 User Handbook

[2]http://www.nrsc.gov.in/pdf/hresourcesat1.pdf : Resourcesat User Handbook

[3] http://www.isro.org/satellites/cartosat-1.aspx

[4] http://www.isro.org/satellites/cartosat-2.aspx

[5] http://www.isro.org/satellites/irs-p6resourcesat-1.aspx

[6] http://modaps.nascom.nasa.gov/services/about/nomenclature.html

[7] http://en.wikipedia.org/wiki/PCI_Express

[8] https://www.khronos.org/opencl/

[9] http://openmp.org/wp/openmp-specifications/

[10] http://en.wikipedia.org/wiki/Heterogeneous_computing

[11] John E. Stone, David Gohara, andGuochun Shi. A Parallel Programming Standard for Heterogeneous Computing Systems.Computing in Science and Engineering. May 2010; 12(3): 66-72. doi: 10.1109/MCSE.2010.69

[11] Craig Lee, Carl Kesselman, Stephen Schwab. Near-real-time Satellite Image Processing: Metacomputing in CC++. In CC++. Computer Graphics and Applications, 1996, pp 79-84.

[12]Yang Zhang, Dan Yu, Shilong Ma. Researches on real-time satellite data flow processing based on file buffer. Seventh International Conference on Natural Computation, ICNC 2011, DOI: 10.1109/ICNC.2011.6022348.

[13] Paolo Bientinesi, Jos R.Herrero, Enrique S. Quintana-Ort and Robert Strzodka.Parallel computing on graphics processing units and heterogeneous platforms. Concurrency and Computation: Practice and Experience 2014. DOI: 10.1002/cpe.3411.

[14] DominikGrewe, Michael F.P.O'Boyle.A Static Task Partitioning Approach for Heterogeneous Systems Using OpenCL. Springer, Lecture Notes in Computer Science Volume 6601, 2011, pp 286-305.

[15]http://www.hdfgroup.org/HDF5/

[16] http://en.wikipedia.org/wiki/JPEG

[17]https://www.princeton.edu/~achaney/tmve/wiki100k/docs/Redd-solmon_error_correction.html

[18] https://gcc.gnu.org

[19] http://www.openscenegraph.org

[20] http://qt-project.org

[21] High speed data acquisition and playback cards: www.apollo-micro.com/space.php