On-board Cloud Contamination Detection with Atmospheric Correction

Kevin Ballou and Jerry Miller
NASA/Goddard Space Flight Center
Greenbelt, MD 20771-0001

Abstract- **The ability to perform on-board science data preprocessing and analysis on a satellite can significantly reduce the amount of bandwidth and storage required in the production of space science products. The implementation of two typical data preprocessing algorithms, cloud contamination detection and atmospheric correction, can provide some valuable insight into the feasibility of implementing more sophisticated processes. Three platforms are considered in this analysis: the microprocessor, the Field Programmable Gate Array (FPGA), and the Application Specific Integrated Circuit (ASIC). By benchmarking the algorithms on a variety of commercial microprocessors, and inferring from the results the expected performance of the available flight-qualified microprocessors, it can be shown that increasing the sophistication of the processing algorithms on this platform would become impractical in a real-time scenario. Hardware implementation of the algorithms, using FPGAs or ASICs, provides the ability to perform much of the processing in parallel, thereby enhancing the performance. The results of benchmarking the hardware implementation of the algorithms can be compared to the results of the microprocessor benchmarks. An additional factor that needs to be addressed with the hardware implementations, especially with the FPGAs, is the utilization of the resources available in the devices. Recent advances in the size and speed of available devices, as well as in the tools used to synthesize, place and route a design, are making this less of a concern. One advantage of the FPGA is that they are relatively quick and easy to program, making them an ideal platform for development and testing, after which the design can be ported to an ASIC.**

I. INTRODUCTION

The ability to preprocess and analyze science data on-board a satellite in real time can significantly reduce the amount of bandwidth and storage required in the production of space science products. Implementing and comparing rudimentary data preprocessing algorithms across various platforms suitable for use on-board satellites can provide valuable insight into the feasibility of implementing more sophisticated algorithms. The two algorithms implemented in this analysis are cloud contamination detection (cloud masking) and atmospheric correction. The platforms under consideration are the microprocessor, the Field Programmable Gate Array (FPGA), and the Application Specific Integrated Circuit (ASIC). The National Oceanic and Atmospheric Administration (NOAA) Advanced Very High Resolution Radiometer (AVHRR) instrument produced the data set used for this analysis, specifically NOAA14 AVHRR Level 1b, 4 kilometer Global Area Coverage (GAC). The feasibility of real time processing is determined by comparing the computational speed of the algorithms on the various platforms to the AVHRR scan rate.

II. ALGORITHMS

A. Cloud Contamination Detection

The presence of cloud contamination can hinder the use of certain satellite data, and these cases require a cloud detection process to mask out cloudy pixels from further processing. The cloud detection tests used in this analysis are based on the work of Saunders and Kriebel [1], except for the Spatial Coherence Test, which came from Thiermann and Ruprecht, as discussed by Cracknell [2]. Simplifications were made to the algorithms so they would be more suitable for hardware implementation.

The image data is processed pixel by pixel through a series of threshold tests, which vary depending on the time of day and underlying surface conditions. For daytime images, a set of background tests is first used to screen pixels that are not suitable for the cloud detection tests. The background tests are comprised of the Sun Glint Test, Snow/Ice Test, and Desert Test. These background tests were taken from the Phillips Laboratory, Automated Satellite Cloud Analysis – Tactical Nephanalysis (TACNEPH) document [3].

For daytime images, the following tests are applied:

1. Infrared Threshold Test
2. Spatial Coherence Test
3. Visible Threshold Test
4. Near-Infrared to Visible Ratio Test
5. Thin Cirrus Test

For nighttime images, the following tests are applied:

1. Infrared Threshold Test
2. Spatial Coherence Test
3. Fog/Low Stratus Test
4. Medium/High Level Cloud Test
5. Thin Cirrus Test

B. Atmospheric Correction

The atmosphere can corrupt the surface target information acquired by a satellite through scattering and absorption. The more dominant of the two is atmospheric scattering. The algorithm used in this analysis addresses this issue, and is based on the basic technique of dark object subtraction, as discussed by Chavez [4]. The algorithm has been simplified and adapted for AVHRR GAC data.

III. Microprocessor Results

Three NOAA AVHRR GAC data sets containing land and water bodies were used for testing: a nighttime image, a daytime image with minimal sun glint, and a daytime image with extensive sun glint. To provide a fair comparison of the various platforms, the Input/Output (I/O) interfaces were not considered, and only the computational portions of the algorithms were timed for comparison. A single scan line of low resolution GAC data represents approximately one fifth the number of samples of a scan line of High Resolution Picture Transmission (HRPT) AVHRR data. Therefore, the execution times of the algorithms are scaled up by a factor of five before being divided into the number of scan lines for the GAC data set under test. This provides a value that represents the execution speed in terms of scan lines of HRPT processed per second. This is easily compared with the AVHRR scan rate of six scans per second.

Three microprocessors were tested, but only the PowerPC 750, with an estimated performance rating of 400+ Million Instructions Per Second (MIPS), could provide a realistic comparison with the currently available flight-qualified computers. The PowerPC 750 was able to process the cloud detection algorithm 6-9 times faster than the AVHRR scan rate, depending on the data set, and the atmospheric correction algorithm 80-120 times faster than the AVHRR scan rate.

The accuracy of the simplified cloud detection algorithms was measured by comparing the output data to a cloud mask obtained from the NOAA Clouds from AVHRR (CLAVR) experimental algorithm. The overall accuracy of the simplified algorithm varied by as much as 68-89 percent across the three data sets.

IV. Hardware Platform

The hardware platform used for implementing the algorithms on an FPGA is the Annapolis Microsystems FIREBIRD^{TM}/PCI reconfigurable computing board. This board utilizes the Xilinx XCV2000E FPGA, contains 36 Mbytes of on-board memory, and provides processing clocks up to 150MHz. The Xilinx XCV2000E FPGA contains over 2.5 million system gates, as well as Block Random Access Memory (RAM), which can be used to store arrays of constants. The FIREBIRD^{TM}/PCI board resides in a host computer, and data is passed between the host and FPGA using the on-board memory. The host configures the board by loading an image into the FPGA, and the board can be reconfigured on the fly. Control of the FPGA is handled with 64-bit bi-directional registers.

V. Hardware Design

The implementation of the cloud detection algorithms in the FPGA involved the development of both the host code, written in C, and the FPGA code, written in Very High Speed Integrated Circuit (VHSIC) Hardware Description Language (VHDL). The VHDL code can be simulated, and there are Annapolis Microsystems tools that allow the FPGA design to be simulated in conjunction with the host code. A Floating Point Math Library was also available from Annapolis Microsystems, which saved the time and effort that would have been required to develop the floating point math routines. Once the design has been functionally simulated, it is synthesized, or translated to a Xilinx specific gate level implementation. Xilinx tools are then used to place and route the synthesized design into an image that can be loaded into the FPGA.

At this point, the accuracy of the algorithms was not as important as achieving a good comparison of processing speed across the various platforms. As long as the output of the FPGA implementation was identical to that of the microprocessor, a good comparison could be made. Therefore, the overall design approach was to implement the cloud detection algorithm one test at a time. Once all of the tests were implemented, and the outputs were successfully compared to the corresponding outputs of the microprocessor, the various test modules could be integrated to form the final design. Before any of the tests could be implemented, the interface between the host software and the FPGA design had to be developed, which proved to be a large part of the design effort.

Because the pixel data for the entire image could not fit into the on-board memory, it was necessary to develop a method of breaking the data set into manageable blocks. In this way, the host could transfer a data block to the on-board memory and signal the FPGA to begin processing. While the FPGA was processing the data, it would place the results in a separate bank of on-board memory, which was large enough to store the entire cloud mask. The FPGA would signal the host when processing was finished. This would repeat until the entire data set was processed. This method also made it convenient for the host to accumulate the amount of time that was consumed by the FPGA in performing the cloud detection algorithm.

There were several things to consider when breaking the data set into blocks. There were five possible results for the cloud mask: CLEAR, CLOUDY, CLEAR_SNOW, CLEAR_DESERT, and UNCLASSIFIED. Therefore, the resulting cloud mask for each image pixel was three bits wide and could fit in a single byte. The memory bank used to store the resulting cloud mask was 32 bits wide, so on-board memory could be conserved and the cloud mask could be stored in its entirety if four byte-wide pixel masks were packed into each output location. In order to remain aligned with the input scan line boundary as well as the output word boundary, the data set was broken into the largest possible blocks that contained an even multiple of four scan lines.

Most of the tests could be performed on a single independent pixel. However, the Spatial Coherence Test involved inspection of a sliding 3x3 pixel window, which meant that each data block, except the first, would need to include the last scan line from the previous data block. The 3x3 pixel window complicated the Spatial Coherence Test algorithm, and necessitated many additional memory accesses for each pixel.

VI. PRELIMINARY RESULTS

At the time of this writing, all of the individual tests were successfully implemented and tested using the nighttime image. The resulting cloud masks were identical to those generated by the corresponding microprocessor algorithms. Unfortunately, when all of the tests were integrated, the design exceeded the available resources of the FPGA. This was a first run at the design, and there are still ways of optimizing the space utilization of the design. One method, which would demonstrate the advantage of the reconfigurable architecture, would be to split the design into two parts. One would process nighttime images, and the other would process daytime images. Depending on the time of day, the host would load the appropriate FPGA image into the FIREBIRD^{TM}/PCI board.

Since the individual tests will be run in parallel in an integrated design, computational speed performance information can be obtained by looking at the execution times for the individual tests. Generally, the integrated design can only execute as fast as the slowest test. Exceptions to this would be when the results from a faster test supersede those of the slower test, in which case the slower test would not need to be carried through to completion. Also, the execution times of the various tests could vary depending on the pixel data being tested. For instance, the Spatial Coherence Test requires all members of the 3x3 pixel window to have the same underlying surface conditions, i.e. Land or Sea. If this is determined not to be the case, the test is aborted.

A preliminary examination of the execution times indicates that most of the tests could process the entire nighttime image in well under half a second. The two exceptions were the Spatial Coherence Test, which took just over a second, and the Thin Cirrus Test, which took just under a second. There most likely are ways to optimize the time performance of the test implementations, but the preliminary results are still able to offer some valuable insight. For instance, if we assume worst case an execution time of 1.5 seconds, scale it up by the factor of 5, and divide it into the 628 scan lines of nighttime image data, we get a result of approximately 84 scan lines per second, or 14 times faster than the AVHRR scan rate. This is about twice as fast as the PowerPC 750. A similar, if not greater, performance improvement is expected for the atmospheric correction algorithm, as well as the ASIC platform.

VII. CONCLUSIONS

While there is still much work to be done, this analysis has already demonstrated several things. Although the PowerPC 750 was able to process the cloud detection algorithm 6-9 times faster than data was being scanned in, this commercial processor is at least 100 MIPS more powerful than its radiation hardened counterpart. Although this design ran into problems with FPGA resource constraints, there are FPGA devices available now with three times the number of system gates available. Since these parts are much larger than their radiation hardened counterparts, radiation tolerance requirements again become a factor and must be considered when implementing a system design. This analysis has shown that the reconfigurable computing platform can be a valuable tool in the development cycle of a hardware design. As the FPGA devices grow in size and complexity, it will become increasingly feasible to implement even more complex real time data processing algorithms. In turn, FPGA devices and reconfigurable computing platforms can play an important role in reducing the amount of bandwidth and storage required for producing space science products.

REFERENCES

[1] R.W. Saunders and K.T. Kriebel, "An improved method for detecting clear sky and radiances from AVHRR data," *International Journal of Remote Sensing*, vol. 9, no. 8, pp. 123-149, August 1988.

[2] A.P. Cracknell, *The Advanced Very High Resolution Radiometer*, 1997.

[3] G.B. Gustafson, R.G. Isaacs, J.M. Sparrow, D.C. Peduzzi, and J.S. Belfiore, "Automated satellite cloud analysis – tactical nephanalysis (TACNEPH)," Phillips Laboratory Directorate of Geophysics, Air Force Materiel Cmd, Hanscom AFB MA, PL-TR-94-2160, November 1988.

[4] P.S. Chavez, Jr., "An improved dark-object subtraction technique for atmospheric scattering correction of multispectral data," *Remote Sensing of Environment*, vol. 24, no. 3, pp. 459-479, April 1988.