Title:

High Performance Signal and Image Processing for
Remote Sensing Using Reconfigurable Computers

Author(s):

Michael Caffrey
MS D440
Los Alamos NM 87544

J. J. Szymanski
A. Begtrup
J. Layne
T. Nelson
S. Robinson
A. Salazar
J. Theiler

Submitted to:

SPIE '99 Optical Science, Engineering, and
Instrumentation
Denver CO USA
July 18-23, 1999

RECEIVED
SEP 07 1999
OSTI

Los Alamos
NATIONAL LABORATORY

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. The Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

HIGH PERFORMANCE SIGNAL AND IMAGE PROCESSING FOR REMOTE SENSING USING RECONFIGURABLE COMPUTERS

Michael Caffrey*, J. J. Szymanski, A. Begtrup, J. Layne, T. Nelson, S. Robinson, A. Salazar, J. Theiler

Los Alamos National Laboratory, MS D440, Los Alamos NM 87545

ABSTRACT

It is not uncommon for remote sensing systems to produce in excess of 100 Mbytes/sec. Los Alamos National Laboratory designed a reconfigurable computer to tackle the signal and image processing challenges of high bandwidth sensors. Reconfigurable computing, based on field programmable gate arrays, offers ten to one hundred times the performance of traditional microprocessors for certain algorithms. This paper discusses the architecture of the computer and the source of performance gains, as well as an example application. The calculation of multiple matched filters applied to multispectral imagery, showing a performance advantage of forty-five over Pentium II (450 MHz), is presented as an exemplar of algorithms appropriate for this technology.

Keywords: reconfigurable computing, FPGA, matched filter, multispectral analysis

1. INTRODUCTION

Extracting information from sample data can be a huge computational challenge. Remotely sensed data typically has many characteristics that can be exploited with custom computing to increase efficiency and performance. For instance, most digital signal processing can be data flow oriented with a very regular architecture of multiply-accumulate operations. Most applications can tolerate latency and fixed-point arithmetic. In addition, most signal processing functions are iterated rather than employing many different non-repeated operations. Reconfigurable SRAM based Field Programmable Gate Arrays (FPGAs) are well suited to computing problems exhibiting these characteristics. FPGAs allow the designer to tailor the hardware to the application at hand. This adaptability gives two advantages: the first is raw performance because the hardware becomes the algorithm rather than mapping the algorithm into instructions for a processor. The second advantage is cost effectiveness over other custom computing solutions. Reconfigurable signal processors can be used for many different sensors or applications, providing generality in a niche computing market. The design of a Los Alamos National Laboratory (LANL) scalable, general-purpose reconfigurable processor will be discussed.

It is not unusual to have remotely-sensed image cubes that contain hundreds or even thousands of megabytes of data. An image cube is a series of two-dimensional images, each representing a spectral channel. These two-dimensional images are typically stored together in a three-dimensional array, or cube. Large data cubes can be produced by many-channel hyper spectral imagers or large-sized (i.e., many pixel) images from multi spectral imagers. In addition, modern sensors can produce image cubes every few seconds. Such large data volumes require either massive storage or intelligent algorithms to filter unwanted data at the sensor head. Because many sensors are on air or space-borne platforms, where volume, power, weight, and bandwidth to the ground are limited, the only viable alternative is to endow the sensor with intelligent filter algorithms. Furthermore, even if the data could be archived, large compute power is needed to process the data, either at the sensor or on the ground. In either case, high-performance compute engines are necessary for most remote-sensing applications.

The example algorithm chosen to illustrate the potential of reconfigurable computing is a straightforward matched filter for multispectral image applications. The matched filter is typically used where the background can be characterized reasonably well and the spectral signature(s) of interest are known. It exhibits all the characteristics mentioned above and its simplicity and wide application make it a good example. We have chosen data with 16 spectral channels per pixel and 12 filters to operate against each pixel. This application has also been written in the 'C' language and run on an Intel-based workstation for performance comparison.

* Correspondence: Email: mpe@lanl.gov; Telephone: 505 667 2422; Fax: 505 665 4197

2. ADVANTAGES OF IN-SITU REAL TIME PROCESSING

One motivation for this work is the problem of processing large data sets acquired by remote sensing instruments, probably at the sensor head or perhaps in the data center. Some examples of the large data volume:

1. The Earth Observing System: The first satellite of this system, Terra, is expected to produce 10s to 100s of Gigabytes/day.
2. A 500-band hyper spectral imager might produce 256x256 (2 byte/pixel) images. Typical imagers require several samples per spectral band, resulting in image cubes of 200 MB or more. Some instruments, when run at high spectral resolution, produce data in several times more bands than this (2000 or more).
3. Two special sessions were held during this symposium to discuss the Multispectral Thermal Imager. This instrument has 15 spectral bands and images that are 2400x2400 pixels (4 bands) and 600x600 pixels (11 bands), resulting in ~55 MB image cubes. Compression can reduce the size of these cubes somewhat (perhaps a 40% reduction), nevertheless, this is a large amount of data for a spacecraft.
4. For further information, there is a DARPA challenge problem outlined at <http://nis-www.lanl.gov/nis-projects/daps/>. This problem gives the data volume, algorithms and total processing time available to do real time calibration of a typical hyperspectral data set.

Another reason for high-performance processing at the sensor is to change the response of the instrument itself. This can be as simple as tuning the operating characteristics of the instrument or the somewhat more sophisticated task of providing the range-to-target. A more complex example is a broad-area search, where images of interest are retained and most images are not. Very carefully designed and tested algorithms are needed to select images to be retained, because an image, once rejected, is lost. These algorithms must be very sophisticated, as they are dealing with complex, changing data sets that required substantial analysis to obtain reliable results. Another application is a hybrid system, where one sensor might cue another sensor, providing pointing information, ranging information and/or information for the data analysis.

Again, even with enough bandwidth and storage capacity to retrieve large data sets from an aircraft or space, processing large volumes of data is difficult at the data center. Many image-processing or remote-sensing algorithms require iteration and/or performing calculations on many regions within an image. Reconfigurable computing offers custom compute engines that can search an archive efficiently, and increase the throughput on these compute intensive tasks.

3. RECONFIGURABLE COMPUTING

3.1. Field Programmable Gate Arrays (FPGAs)

Reconfigurable computing uses an FPGA as the core-processing element. The FPGA has been on the market for more than a decade; its principal use is for discrete logic integration and board level integrated circuit (IC) count reduction. It is itself a digital IC and represents the highest transistor count of all, up to 100 million transistors¹. For comparison, the Intel Pentium III is in the neighborhood of 9.5 million transistors². Recent advances yielding this density and improving speed performance have made these parts appropriate for hardware DSP. A historic concern for DSPs has been technology obsolescence. The

FPGA market is not driven by DSP applications, so this technology will not lose technical support or future product development as many custom DSP ASICs have in the past.

An FPGA is organized into an array of logic blocks and registers surrounded by a configurable routing matrix. The logic blocks are small

Normal Mode

Figure 1: Logic Element of the Altera Flex 10k series device. A Logic Array Block (LAB) is an array of 8 Logic Elements³.

SRAM lookup tables that implement Boolean functions of three to five inputs, depending on the vendor. Figure 1 illustrates the lookup table and register architecture of the logic blocks for an Altera 10k device³. The contents of the lookup table and the setting of the multiplexers is determined by the user's design and can be infinitely reprogrammed; hence, the reconfigurable computing element. For example, the contents of the lookup table might implement a bit adder; several logic blocks can be organized together to implement an eight-bit by eight-bit adder to operate on digital data.

Figure 2: Routing Matrix of an Altera 10k series device⁴.

The configurable routing, Figure 2, connects the arrays of logic blocks to one another and to hundreds of user configurable I/O pins for interfacing the part to the system⁴.

A particular application will have a configuration bitstream that sets the contents of the lookup tables and routing connections to implement a particular algorithm. Depending on the device, this configuration can take a few hundred milliseconds to a second and they can be infinitely reconfigured. FPGA designs are created from either a hardware description language or schematic capture that describes how adders, multipliers, comparators, and other operators are assembled to implement filters, convolutions, and other DSP algorithms. A synthesis tool takes the design input and reduces it into Boolean logic and maps it into the logic blocks of the FPGA architecture of choice. A place and route tool, usually vendor specific, takes the logical decomposition from synthesis and places the logic blocks into the specific FPGA being used and configures the routing as required by the design.

The FPGA architecture can be used to accelerate DSP in several ways. The first and most obvious gain comes

from parallel computing. For example, simply operating tens of multipliers simultaneously. This parallelism puts the FPGA in a unique computational position of being able to maintain throughput while increasing the number of operations. For instance, a microprocessor processes fewer samples in time as filter lengths increase, while an FPGA exhibits increased logic utilization but maintains total data throughput by increasing the number of multiply accumulate units in the design.

Data dependencies known a priori can be exploited to maintain high bandwidth data throughput in an FPGA because the logic architecture is specifically adapted to the dependency structure. The microprocessor has been designed to accomplish all computing tasks, meaning that information about data dependencies for any particular algorithm cannot be heavily leveraged to accelerate the application. For DSP, the location of data is typically known and can be designed into the FPGA. For instance, circular buffers that are commonly used for delay functions in DSP can be optimized for performance. A microprocessor has many cache coherency problems as well as memory bus contention problems. These advantages give the FPGA 10 to 100 times the performance of a programmable DSP or microprocessor for applications that exhibit the regularity often found in DSP.

The FPGA has different design optimization criteria than the microprocessor. In particular, the operand precision has a huge impact on the performance of the operation in an FPGA. It requires far fewer logic blocks to implement eight-bit arithmetic than 16-bit arithmetic. This requires algorithm designers to truly understand the precision needs of the algorithm and how fixed-point arithmetic will impact the stability and precision of results. This challenge is frequently overlooked when designing for microprocessors that deliver 32-bit floating-point performance, but in an FPGA 32-bit floating-point is exceedingly inefficient. Computing techniques also have a large impact on efficiency and performance for FPGAs. For instance, when taking the magnitude of a complex number, the obvious approach uses the square root operation, but the square root operation is inefficient in the hardware implementation. The CORDIC (COordinate Rotation DIgital Computer) algorithm for coordinate transformation is far more efficient in hardware for performing this computation. Designing DSP in an FPGA often requires creative solutions and understanding of algorithms dating back 30 years to the time before the

microprocessor. It is well worth the effort, though, because the computation available for given weight, space and power is so much greater that it is feasible to do substantial in situ processing of remotely sensed data.

The reconfigurability of the FPGA can be used in two ways for computing. The first, of course, is that it provides a cost-effective general-purpose approach to custom computing. The computer can adapt to many different sensors and applications without redesigning the hardware of the printed circuit board assembly. The algorithm implemented in the FPGA can also evolve as science refines information extraction techniques; this saves costly redesigns of custom DSP lacking the programmability of FPGAs. The second, and more interesting from the research point of view, is the idea of switching configurations at runtime to achieve a greater apparent logic density. This dynamic reconfiguration can dedicate more of the FPGA to the processing being done at the instant, then swap a configuration that performs the next step of the calculation. This means that no two steps share the same resources, hence each could use more resources individually and potentially achieve higher throughput. While an important research topic, many remote sensing applications need to perform the same calculations over and over and hence do not utilize runtime reconfiguration because of the performance penalty of actually reconfiguring the FPGA. It is an important resource to keep in mind as it can be used for test purposes, pre-loading memory before operation, or for processing different modes (search vs. tracking, for instance) of a sensor.

3.2. Computing Architectures and Techniques

There are several important architectural considerations for computing with FPGAs including I/O, memory, data buses and control interface. Critical to remote sensing is the architecture of the data I/O. Microprocessor bus architectures like VME or PCI are arbitrated bi-directional buses with several transmitters and receivers. Contention on the bus between these devices quickly reduces the theoretical throughput and becomes a bottleneck that starves the processors. The solution for FPGA computers is to have several dedicated data buses in addition to the traditional arbitrated bus. These additional data buses are point-to-point and have only one transmitter with extremely simple arbitration. Such data buses can support in excess of one hundred MBytes/sec sustained to overcome the I/O bottlenecks.

Effective FPGA based signal processing also requires huge memory bandwidth. This means multiple separately addressable memory banks on separate data buses. Only the number of available pins on the FPGA limits the number of memory banks. This helps the FPGA perform multiple fetch and store operations each clock cycle to overcome the limitation of the microprocessor and its cache hierarchy, which limits memory bandwidth.

In applications that require multiprocessing to achieve the required performance there must be a mechanism for communicating between the processors. With a microprocessor architecture there is typically a bottleneck between a microprocessor and access to a peripheral communication port as well as substantial packaging overhead. The FPGA-based computers can hardwire the buses between processors to resemble the point-to-point data paths discussed above. If a large number of FPGAs are needed, then crossport switches can configure the communication ports between processors on an application by application basis.

The control interface is typically done over the arbitrated interface bus by a microprocessor. Responding to interrupts or monitoring status as well as providing configurations and control input is a function best performed by the microprocessor because of its flexibility. For this reason, most reconfigurable processors are in fact heterogeneous systems incorporating a microprocessor that interfaces to an operator. The microprocessor is often referred to as the host; it retrieves the final processing results from the reconfigurable system after the processing has reduced the volume of information to manageable levels.

These architectural techniques were incorporated by LANL into the reconfigurable computer architecture II for processing remotely sensed data. Several sources additional sources^{5,6,7} can provide more depth for those interested in configurable computing.

3.3. The Reconfigurable Computer Architecture II

The LANL Reconfigurable Computer Architecture II (RCA2)⁸, Figure 3, uses three Altera 10k250 Complex Programmable Logic Devices (the CPLD is an industry alternate name of the FPGA) processing elements organized in a ring architecture. Each of these processors has 40 Kbits of on-chip memory and provides a nominal 250,000 configurable logic gates⁹. There are three high-speed data ports on the front panel of the processor module. This allows for computing on two input data streams or producing two sets of output results from one input data stream; this is a common data artifact of operations like cross-correlation. The physical form factor is a 'C' sized VMEbus single width card. Each of the data ports is implemented as a daughtercard so the interface can be configured for input or output and the port can be customized for different sensors. The middle daughter card broadcasts to all 3 processor nodes for additional flexibility in data distribution, see Figure 4. The host communicates with the module over the VME bus and provides device configuration, clock rate adjustment, application control input, monitors status, and retrieves results. First In First Out (FIFO) buffers on the daughtercards interface the computing system and its processing clock to a sensor with an independent sampling clock. To prevent overflow, the processor must, on average, maintain the rate of the sensor. Each processor element has 3 independent banks of high performance Zero Bus Turnaround (ZBT) synchronous SRAM that can store intermediate calculations, results, lookup tables, etc. This memory is extremely fast and can perform back-to-back read and write operations, making it ideal for DSP. It is also simple to use from an

Figure 3: The LANL RCA2 Processor

Figure 4: Functional Diagram of the RCA2 Processor

application because it is synchronous and does not require wait states or complex state machines to control. In addition to the individual banks of SRAM, there is a bank of shared SRAM between each of the processors. This memory can be switched between the processors in a single clock cycle. One powerful feature of this memory is the 'free' image transpose. One processor can write data into its shared memory, then change the switch. Then the next processor can simply read the memory with flipped address lines to automatically transpose the image for the next calculation.

4. EXAMPLE: MATCHED FILTERS FOR MULTISPECTRAL IMAGERY

The matched filter algorithm is used to find known (spectral) signatures within an estimated background. For this discussion, we will think of the spectra that describe a signature as vectors in N-dimensional space, where N is the number of spectral channels. Many matched filters can be defined – the example used in this paper assumes white noise so that the preferred direction of the filter is along the direction defined by the signature's spectral vector. This "classical" matched filter is expressed as

$$\text{Filter vector} = \frac{b}{\sigma \sqrt{b^T b}}, \quad (1)$$

where b is the signature vector, and σ is the strength of the noise.

It is easy to modify the matched filter to take into account the background clutter of the scene. Using the covariance matrix computed from the spectral vectors describing each pixel typically does this. The matched filter with background clutter derived from the covariance matrix is¹⁰

$$\text{Filter vector} = \frac{K^{-1}b}{\sqrt{b^T K^{-1}b}}, \quad (2)$$

where b is again the signature vector, and K is the covariance matrix. The matched filters shown are normalized in the sense that the output is the size of the signal compared to the noise.

The data used in this study come from the MODIS Airborne Simulator (MAS). MODIS in turn is the MODerate Resolution Imaging Spectrometer to be flown as part of NASA's Earth Observing System program. MAS data have 50 spectral bands. For this study, 16 of the spectral bands were taken that correspond roughly to the visible, near-, short-, and mid-wave infrared. Figure 5 shows spectral band two from the 16 in the MAS data cube and the result of the matched filter with one signature vector. The output is identical from both the Pentium II and RCA2 processors that were used in the experiments.

Figure 5 A: Band 2 of the 16 bands in the MAS data cube.

B: Output result from a single signature vector.

We implemented a search for 12 spectral signatures against the 16 spectral bands of the MAS imagery. The example used 512 x 512 images, though any size can be accommodated. The data was organized in a pixel-interleaved format, meaning the first 16 samples input to the reconfigurable processor were the 16 bands of spatial coordinate 0,0. The next 16 were for 0,1 and so on. The input data and filter coefficients were set to 8 bits of precision, signed 2s-complement integer, zero mean. The results were 16-bit precision, signed 2s-complement integer to accommodate for the growth in the accumulator. The design in the Altera, Figure 6, required 45% logic utilization of a single 10k130 (nominal 130k gates) processor on the RCA2. This design maintained a sustained data rate of 35 MegaSamples/sec. Additional signatures can be added to the design in a single part or the other 2 processors without affecting throughput. Other post-processing such as thresholding can also be performed without slowing the data rate. In fact, the architecture can be scaled at the module level; several modules can be assembled together to search for an arbitrary number of signatures while never reducing the data rate. This is the real power of scalable processing with FPGAs.

Figure 6: Architecture of Filters in the Altera 10K

The performance of the reconfigurable processor can be difficult to compare against a microprocessor because it is so scalable. The performance increase comes from parallel processing, so the results appear better when the search is for more signatures (e.g. more parallelism). For this specific example, we coded the algorithm in 'C' and executed it on a 450 MHz Intel PII. The data was loaded into memory before the beginning of the timed calculation. To process one image cube 512x512 by 16 bands, for 12 signatures, required 2.04 seconds, averaged over multiple trials. Only .12 seconds were required for the same data on the RCA2. This represents a performance advantage of 17:1 for this particular algorithm. If using the entire RCA2 to search for 32 signatures, there is an acceleration of 45:1 because the microprocessor requires more time while the RCA2 maintains throughput. Realize that the microprocessor has one hidden advantage, the ability to do floating-point arithmetic with little or no performance penalty. While convenient, floating-point arithmetic is not usually necessary for signal processing and in fact has only been widely available for the last decade.

5. CONCLUSION

Reconfigurable computing, based on the FPGA, solves many of the data processing challenges posed by remote sensing applications. The ability to perform application dependent parallel processing and to exploit data dependencies known a priori enables a 10 – 100x performance gain over microprocessor based DSP. The ability to generalize the reconfigurable processor to different sensors and algorithms as well as evolve the algorithms as science progresses makes the FPGA solution very much more cost effective than other custom computing. In addition, there is a strong and broad market driving FPGA product development that will provide similar performance evolution to that seen in the microprocessor arena. The FPGA brings a different set of optimization challenges for the algorithm designer, including regular data flow operations and fixed-point arithmetic. In addition, familiarity with alternative algorithms like the CORDIC helps improve efficiency in reconfigurable computers.

The architecture of an FPGA-based reconfigurable computer relies heavily on the ability to deliver data to the FPGA compute nodes. This drives the need for multiple dedicated data paths that are shown in the RCA2 design we discussed. These multiple data paths each deliver up to three streams of data sustained at over 100 Mbytes/sec. The front panel daughtercard design allows the RCA2 the flexibility of interfacing to different sensors or to one another for scalability. The organization of memory is also critical design input. With many banks of fast SRAM, the FPGA will offer multiple store and fetch operations in a single clock cycle, which boosts parallel processing performance. The shared memory of the RCA2 also offers convenient data formatting when passing data from one compute node to another.

The matched filter example showing the search for spectral matches in the MAS data set demonstrates the utility of reconfigurable computing for remote sensing. The algorithm has wide application in both signal and image processing. A speedup of 17x was shown for 45% of a single Altera 10k130 part on the RCA2. Full use of the RCA2 offers an advantage of 45 over a 450 MHz Pentium II. The example demonstrates the parallel processing and scalability of reconfigurable processing that allow a constant data throughput.

ACKNOWLEDGMENTS

This work was performed under the auspices of the United States Department of Energy.

REFERENCES

-
- 1 Approximate for Altera 20K1000, Hotline 1800800EPLD.
 - 2 <http://www.intel.com/pressroom/kits/processors/quickref.htm>
 - 3 Altera Data Book, p. 40, 1998.
 - 4 Altera Data Book, p. 48, 1998.
 - 5 Villasenor, J., Schoner, B., Chia, K., Zapata, C., Kim, H., Jones, C., Lansing, S., and Mangione-Smith, B. "Configurable computing solutions for automatic target recognition," in 4th IEEE Symposium on FPGAs for Custom Computing Machines (FCCM '96) (1996).
 - 6 Villasenor, J. and B. Hutchings, "The Flexibility of Configurable Computing," IEEE Signal Processing Magazine, pgs 67-84, 9/98.
 - 7 Villasenor, J. and W. Mangione-Smith, "Configurable Computing," Scientific American, (June 1997).
 - 8 Robinson, S. H., Caffrey, M. P., and Dunham, M. E., "Reconfigurable Computer Array: The Bridge between High Speed Sensors and Low Speed Computing". Proceedings of the Eighth International Workshop on Field-Programmable Logic and Applications, FPL '98, pages 159-168, Tallinn, Estonia, August/September 1998. Springer, ISBN 3-540-64948-4.
 - 9 Altera Data Book, p. 22, 1998.
 - 10 A.D. Stocker, I.S. Reed and X. Yu, "Multi-dimensional signal processing for electro-optical target detection," Proc. SPIE 1305, 218 - 231 (1990).