STARCON - A Reconfigurable Fieldable Signal Processing System

Scott Brandt
Theseus Research, Inc.

John Budenske
Alliant Techsystems Inc.

Abstract

Conventional fieldable signal processing systems utilize custom hardware manufactured and configured specifically for a single signal processing application. Developing new systems or reconfiguring existing systems involves great expense and time expenditure. We at Alliant Techsystems have developed a signal processing system based on commercially available hardware which is completely software programmable and yet small and fast enough to be used in fieldable multisensor signal processing applications. This paper will discuss Alliant's reconfigurable signal processing system.

1. Introduction

The high computational complexity of signal processing applications typically require them to be extremely hardware intensive. In the past, such complexity was limited by the existing hardware technology and the prohibitive cost of building systems out of that hardware. In recent years, developments in circuit design technology have made dramatic improvements in speed, size, sophistication, and cost of computer circuits. These improvements have made possible ever more complex computer systems, and thus have directly benefited designers of signal processing systems. However, as the systems grow larger they become increasingly more complex, introducing new problems and compounding old ones. Due to the increasing complexity of the systems, development prototypes are becoming ever more important and at the same time extremely expensive and difficult to develop. One solution to this problem is to develop a fast, flexible prototyping system which uses current technology to yield real-time/near-real-time processing speeds and yet retains the simplicity of the best software based systems. Such a prototyping system can be used to quickly and inexpensively develop fully functional signal processing applications.

The first step in a signal processing system development cycle is usually the implementation of a software simulation of the desired algorithms on some general purpose computer (GPC). GPC software simulation is relatively inexpensive, compared to hardware development, and many signal processing labs have libraries of signal processing routines available from which simulations

can be quickly developed. In addition, software systems are extremely flexible allowing the algorithms to be modified and refined as more is learned about their performance. These simulations are usually sufficient to demonstrate the basic feasibility of the algorithms in question, but some considerations exist which greatly limit their overall utility. One major consideration is that such GPC-based simulations typically run very slowly, many orders of magnitude slower than systems using special purpose hardware. This results in prohibitive amounts of time necessary to test and evaluate the simulated system on adequate amounts of data. As a result such simulations are usually tested over a very limited range of data scenarios, often a single set of signals, or sometimes even a single signal. The test results in this situation are rarely representative of the overall system performance, and often result in misguided expectations about the simulated system's future performance. A second consideration arises in that a simulation implemented in software will probably bear little resemblance to the final hardware-based system. Hardware developers are often faced with design constraints which dictate how a particular piece of hardware can be implemented. These constraints are usually not known in advance by the algorithm developer and often affect the implementation of the algorithm significantly.

The second step in a signal processing system development cycle is usually the implementation of a prototype of the system using special-purpose hardware. There are two main advantages of a hardware prototype. The first is that a prototype development proves that the system in question can be implemented using existing technology. Proposed systems often push the state-of-the-art in hardware technology and a successful hardware prototype is essential in proving that the overall concept is workable. The second advantage of a hardware prototype comes in testing the system. Special-purpose hardware systems are generally much faster than software simulations of the same algorithms. This results in greatly increased system throughput allowing the system to be tested and evaluated on a larger, more representative set of signals. Thus, a much more thorough system test is possible than would be with a software simulation. In addition, hardware prototypes are often fieldable, allowing the system to be tested on sequences of real signals/imagery in real data collection scenarios. However, as in the software case there are a few considerations which limit the utility of hardware prototypes in general. The first consideration is cost. In general, the development of signal processing system prototypes is very expensive and time consuming. Such systems almost always require custom hardware resulting in significant design and manufacturing costs. Once built, such systems are also very expensive to maintain and fix should something break. A second consideration is that hardware is somewhat inflexible. In the event that some aspect of the design needs to be changed, significant time and money can be spent attempting to accommodate the changes. Finally, intermediate hardware prototypes usually do not implement all features of the final system. Inasmuch as the prototype is different from the final system, it yields no information about how the final system should be designed. As in the case of a software simulation, the final system can differ significantly from the prototype.

One overriding problem in the whole signal processing system development cycle is the great difficulty and large expense involved in transitioning the algorithms from initial concept to GPC simulation to hardware prototype to real-time fieldable system. Our objective in developing the system described in this paper was to incorporate the advantages of both software and hardware based systems and at the same time create a well-defined path from initial system concept to simulation to fieldable system. We wanted to develop a system which was completely software programmable requiring the algorithm developer to make no hardware adjustments, settings etc. and

which would process at or near real-time rates. In addition, we wanted to realize a fully integrated prototype system/demonstration testbed which would provide a clear path to the final system.

2. System Description

The STARCON system consists of a platform of commercial hardware and software upon which we have implemented several layers of functional software. The basic hardware consists of a VME chassis containing a general purpose 680XO-based CPU card with an ethernet interface and several Datacube image processing cards. Datacube image processing cards are real-time register-programmable image processing modules. Each module provides a range of related image processing functions. The Datacube cards are custom cabled to support the STARCON system. The cabling configuration is designed such that all of the capabilities of each of the Datacube cards in the system can be utilized by the software without changing any of the physical connections. The CPU card provides VME access for register configuration of the cards. The VME chassis is connected via ethernet to the local Sun network allowing communication with and logins from any of the Suns on our network. The system can be connected to any of several inputs and outputs including laserdisk, VCR, and video monitors and cameras. The entire architecture of the system is open such that any new cards compatible with the Datacube image transfer format can be easily integrated into the system.

The operating system we chose for the STARCON systems is VxWorks. VxWorks provides a nice development environment compatible with the Sun 680XO series C compiler in a library format which provides processing speed adequate for real-time and near-real-time image processing applications. VxWorks provides many high-level library functions for ethernet communication, NFS, semaphores, and many other high- and low-level functions. All of the STARCON code is written in C on a Sun and downloaded onto the CPU running VxWorks.

The STARCON software consists of two libraries: the System Functions and the Algorithmic Functions. The System Functions provide a software interface to all of the hardware-specific capabilities in the system. In particular, the System Functions do all of the general hardware-specific functionality required for the low-level Algorithmic Functions as well as all of the UO related functions. The System Functions also provide a structured software methodology for programming and using the system.

The Algorithmic Functions provide all of the image processing functionality in the STARCON system. Each of the lowest level Algorithmic Functions is based upon a single Datacube card, thus providing the algorithmic primitives. Examples are convolutions, histograms, averages, and mathematical operators. These low-level functions are used to create higher level Algorithmic Functions by filling in parameters or by combining together several low-level functions. For instance, by filling in the kernel of a convolution in different ways, many specific functions can be created. Through the creation and combination of low-level functions we have created a fairly extensive library of Algorithmic Functions. in cases where a function cannot be created using one of the existing primitives, new primitives can be created easily using existing functions as templates. In addition, if it is found that a function cannot be implemented using one of the boards in the system, a software function running on the general purpose CPU can be added to the library using the

same software interfaces. To any higher level function, this software function will be indistinguishable from any other hardware-based function in the system.

3. Design Philosophy

There are many aspects of the STARCON system as described above which make it an ideal laboratory tool for algorithm development. The use of off-the-shelf components allowed us to take advantage of many pre-existing standard products including compilers, editors, and others which provide a natural and flexible environment. In addition, commercially available products are cheaper to obtain and cheaper/easier to fix than similar custom hardware. Datacube image processing cards also provide a degree of flexibility combined with speed which would be prohibitively expensive to achieve using any other hardware, custom or commercial.

On top of this base of commercial products we have implemented a flexible, hierarchical software system which makes the STARCON system extremely powerful. There are seven distinct software levels in STARCON which provide a modular, layered interface to the system. The seven layers incorporate the commercial base of software, the STARCON system software, the various levels of algorithmic software, and a high-level programming interface. Each of the software layers provides a distinct functional purpose and allows the system to be programmed at many different levels depending upon the needs of the algorithm developer. We have incorporated strict standards defining the structure of functions at the various levels and through their use have developed a uniform, straightforward library of functions. By using these distinct software layers, programs written for STARCON are easy to understand and modify.

The libraries of functions we have developed for STARCON provide an easy-to-use interface for algorithm developers using the system. One of our goals was for an algorithm developer to be able to use the STARCON system without any knowledge about the underlying hardware and software. The System and Algorithm libraries accomplish this goal. The programming interface is completely functional, requiring no knowledge beyond that used to develop the algorithm and a familiarity with the libraries of functions, and dependent only upon the existence of the specific low-level functions to accomplish the desired algorithm. The extensive library of Algorithm Functions insures that this will occur only rarely. If a particular function is not available, it can usually be created from existing lower-level functions. If this is not possible, the algorithm developer need only learn enough to program the particular low-level function, using the System Functions to accommodate all aspects of interfacing the function to the rest of the STARCON software.

Finally, the Datacube image processing hardware is extremely useful in developing real-time pipelined systems. By incorporating the Datacube hardware into the STARCON system we have virtually eliminated the development time necessary to convert the desired algorithm to a real-time prototype system. Once an algorithm is developed on STARCON, all of the requirements for the real-time prototype are known, the board-level software is written, and the timing requirements are available. If desired, a real-time pipelined system can easily be assembled in a very short period of time.

There are many advantages to the approach we took in developing the STARCON system. We achieved a system which is fast, easy-to-use, easy-to-support, and provides a simple path to the final real-time prototype system.

4. Future Plans

Due to the initial success of STARCON, additional enhancements to the system are already being made. These enhancements will provide an easy-to-use conceptually oriented programming interface to the STARCON system. They are designed to extend the user's ability to program the system based on the concept of a “Multi-Component User-Interface” running on a Sun workstation in the XWindows environment. The User-Interface will provide several tools for programming the STARCON system including tools for creating Algorithmic Functions, tracing algorithm execution, tuning applications and Algorithmic Functions, and iconic algorithm programming.

The main component is a “Chassis Controller” interface which will allow the user to control STARCON through the use of mouse/menu selections. The second component is an “Iconic Programmer” which includes a hierarchical graphic dataflow-oriented display/programming capability. Icons representing algorithm pieces are manipulated dynamically via the mouse to design/program/view STARCON applications. The Iconic Programmer allows the user to select algorithm pieces, data paths between algorithm pieces, and parameter values. Thus, users with little knowledge of the hardware and C programming language can develop a complete prototype system from a conceptual dataflow design. A menu-based “Librarian” is another component of the User-Interface. The Librarian will be a series of “access menus” for accessing the Algorithmic Functions and System Functions, as well as other CPU-based algorithm libraries. Finally, additional User-Interface tools to aid the programming of Datacube/VME-based systems and the conversion of STARCON programs into a final real-time pipelined system are being developed.

Each of the above components of the User-Interface provide additional capabilities to the user. This is accomplished by applying user-friendly methodologies to reduce the amount of in-depth knowledge required of the programmer. Special care is being used in the design of the User-Interface to insure that those users who wish to continue programming on the lower levels of the system are not limited or restricted by the design and to insure that any work performed on those levels will still be of use to those who interact through the User-Interface.

5. Conclusions

STARCON is a real-time system development environment. It has been found to be useful in rapid algorithm prototyping and development, as well as allowing developed prototypes to be tested in the field. Though only a year and a half since the inception of STARCON, it has been successfully used in two large research programs, one of which required that the system be tested in the field. STARCON will be used in many upcoming programs, most of which would not be possible if not for STARCON's low cost in developing and fielding systems.

Many attributes can be used to describe the STARCON system. It is a software reconfigurable system; a development environment; a recirculating architecture; an image and signal processing library; a cost-effective path to real-time system implementation; a conglomeration of specialized

off-the-shelf hardware; a prototype toolkit; a fieldable system; a fully functional demonstration environment; etc. However, the most impressive attribute of all is “successful IR&D program”. The STARCON system is a successful attempt to combine the speed of specialized image processing hardware with the flexibility of software control. STARCON's unique and innovative design allows advanced signal and image processing research to proceed in a quick and cost-effective manner. New ideas can be quickly implemented and explored. Fully functional systems can be brought to the field and evaluated long before costly custom hardware implementation and long before costly final design decisions are made. Development on STARCON continues. As the Multi-Component User-Interface implementation moves forward, and the next stage of development is being designed, the STARCON systems is progressing towards a complete image and signal processing research environment.