ARCHITECTURES FOR HIGH-SPEED PROCESSING

R. K. Bardin
Applied Physics Laboratory
Lockheed Palo Alto Research Laboratories

Abstract

Studies of architectures for high-speed, high-volume signal processing are described on two levels: (1) An experimental system for testing architectures of multi-instruction multi-data processor networks; and (2) an analysis and proposed extension of the Novix NC-4016 architecture. It is proposed that the key design element distinguishing the NC-4016 is not RISC, but simplified instruction decoding, or SID.

Introduction

The high-speed signal processing task is one of the major driving forces in the current wave of microprocessor and computer development. At the Lockheed Palo Alto Research Laboratories (LPARL), we have under way a program in real-time synthetic-aperture radar (SAR) signal processing, a task that requires multiple billions of operations per second. I would like to describe some of the directions we have been exploring in architectures of systems and microprocessors that may be suited to such tasks.

There are two levels at which we have been attacking the problem. The first level is that of parallel processing architectures. Although we expect that single-instruction multiple-data (SIMD) structures will also have a role to play in processing tasks as massive as the SAR problem, the GAPP program described by Dr. Ting at this conference is already under way elsewhere in LPARL. We therefore have chosen to explore multiple-instruction multiple-data (MIMD) architectures.

For our MIMD development, we have been constructing a test system for studying a variety of lattice structures built from small FORTH processor nodes, initially based on the Novix NC4016. The design of the test system will permit us to reconfigure the topology of the lattice at will, so we will be able to compare hypercubes with, for example, hexagonal lattice topology. I will describe below some of the considerations which have gone into this test system.

The second level of exploration is the architecture of the node processor itself. Our past experience with the Novix processors coupled with some preliminary thinking has led us to propose a generalization of the Novix design philosophy and some specific extensions to the Novix chip. The objective of these extensions is to make the extremely fast and flexible Novix-class general-purpose processors competitive with the arithmetic throughput of present dedicated digital signal processor chips. Such a combination should be invaluable in tasks requiring intelligence as well as sheer numeric speed.

SAR Processor Test System

Since SAR processing involves a variety of computational and control tasks, it is not immediately obvious just what parallel processing structures will be best for accomplishing the task. The Test System is therefore designed for maximum flexibility, permitting immediate changes to the network topology by simply reconnecting cable links between nodes.

The design philosophy for the Test System has been oriented toward simplicity and large design margins. The idea is to study model architectures in the quickest and cheapest way; the scaling of speeds to their technological maxima can be carried out later with a well-defined and tested architecture.

Figure 1 shows the overall configuration of the Test System and its peripherals. Since even the initial, small-scale test system is expected to have the throughput of several VAXes, a high-speed streamer tape is necessary to provide a reasonable match to the I/O requirements. We plan to use a high-resolution memory-mapped display both for tracking of parallel node tasks and for output display.

As indicated schematically in Figure 2, the node processors will be individual cards with provision for up to six local internode links per processor, plus a global bus primarily for housekeeping and programming functions. The initial plans for the system include between 10 and 20 nodes, limited by budget. The links we have chosen for the first studies consist of shared blocks of memory between processors. Our past experience with multi-processor systems indicates that shared memory permits closely-coupled high-rate communication between processors, but raises a minimum of critical timing and hand-shaking requirements. This is in contrast to direct bus interconnects, which require cycle-by-cycle cooperation during data transfer.

While normal program loading and booting will take place under control of a host processor via the shared-memory link to the global bus, each node has available a disc interface connector and an RS-232 terminal port. Since each node contains its own copy of the FORTH kernel in PROM, only the connection of a terminal and a 3-1/2-inch disc drive to the card is required to achieve independent access to any node. This facility is expected to be a major aid to the debugging and diagnostic process.

Since the interprocessor links will be standardized, alternative node processor types can easily be substituted for the initial Novix-based design, permitting later upgrade of the system and exploration of architectures at the node level.

Microprocessor Architectures

The SID Concept. There has been a great deal of effort put into the development of microprocessor architectures in the last few years, including recent efforts to simplify designs, and thereby improve speed, chip size and dissipation; this I consider one of the most significant trends of the 1980's. One of the products of this trend to simplification is the Novix processor, which is playing an essential role in making economically and technically feasible the work I have been reporting.

The fashionable approach to architecture simplification has been RISC, the Reduced Instruction Set Computer, which simplifies the microcode structure used on chip by reducing the size and complexity of the instruction set. The resulting gain in speed per instruction must then more than offset the loss of flexibility in the instructions. The burden of any needed additional instruction flexibility is passed on to the compilers of the higher-level application language.

The Novix is usually described as a RISC machine. However, I claim that RISC is a misnomer here, and that the term is in fact concealing a significant new concept in the Novix design.

I find two key properties characteristic of the Novix architecture, and strangely enough, FORTH is not one of them. It is true that the FORTH language motivated the Novix design, and also true that a FORTH representation of the Novix instruction set is the primary software interface to the machine, but neither of these factors is intrinsic to the resulting design.

The first key characteristic of the Novix architecture is its greatly simplified approach to the implementation of instruction decoding and execution on the chip. A single static decoding array carries out all of these functions, with no multi-cycle microcode processing required other than for two or three intrinsically multi-cycle ALU functions. Nearly all instructions, therefore, are a single clock cycle long. The very important consequence of a single-cycle instruction set is that the Novix saturates its main memory port: the chip uses essentially every available clock cycle to access memory for either instructions or data. There is neither need nor available clock cycles for pipelines or instruction prefetch systems. The Novix is pushing the intrinsic limits of its Von Neumann architectural heritage.

Notice that the emphasis is on simplifying the chip structure here rather than on reducing the instruction set. In fact, by making the instruction decoding as simple and transparent as possible, more of the intrinsic capabilities of the ALU and data-flow architecture can be made available, leading to a natural increase in instruction-set size, not a decrease.

The Novix instruction set illustrates these concepts very well, being simple in content but moderately large. One must, of course, look under the FORTH overlay, which tends to conceal this fact by restricting itself to a mere 40 to 50 primitives in present Novix compiler implementations. However, many more instructions are accessed by combinations of primitives, and it is clear that present compilers by no means exhaust the available instruction set of the machine. As the number of instruction bits per word in similar microprocessor designs increases from 16 to 32, this idea will become increasingly practical and easy to implement, and will lead to even larger instruction sets.

Because of the importance I see in the simplification of instruction decoding, and to resolve the paradox of tagging designs with enlarged instruction sets with the RISC label, I would like to call this new approach a Simplified Instruction-Decoding, or SID, architecture.

SID and Software. The software effects of the SID architecture are interesting: the large instruction set permits the programmer a great many of the options previously available only in microcode. With reasonable care in the design of the residual decoding structure on the chip, this new, external microcode instruction set (a microinstruction set, if you will) can allow enormous flexibility in coupling the machine instruction set to language compilers.

In effect, the necessary mapping of microcode onto a usable instruction set has been removed from the hardware functional requirements of the chip, and is now carried out in software; the mapping may therefore be made directly to a chosen high-level language, bypassing the restrictions of a fixed assembly language. Clearly, there are many opportunities for intelligent, optimizing compilers here, yielding, with a little care in design, high-level languages of very high performance at the cost of slightly greater compilation time.

There appear to be no intrinsic limitations of the SID architecture to particular languages, although languages which are extensible at the primitive level as FORTH is are strongly favored for writing compilers for these machines.

Data-Path Architectures. The second key descriptor of the Novix processor design is its highly parallel internal data-path architecture, coupled with fully simultaneous access to both stack memories in parallel with main memory access. Such parallelism is clearly essential to the support of the Novix single-cycle instruction philosophy and the resultant speed of the processor.

We have shown that the instruction decoding of the Novix has reached the limit of the Von Neumann bottle-neck, the bandwidth of the main memory port; my next question is what are the corresponding limits for the data paths and ALU, and how close is the Novix design to optimum here? Clearly, the ALU cannot be kept completely busy over the long term just through the main memory port, since full saturation at one ALU operation per clock cycle implies the transfer of three data words per clock cycle, two ALU input words and one output. In practice, of course, outputs are frequently reused as inputs to succeeding operations, and the stacks serve as data caches to allow flexibility in that reuse. Nevertheless, a cache is of only limited use if there are no spare bus cycles with which to fill it, and this is precisely the situation with the Novix design. One would like to achieve an architecture where the instruction stream runs at one instruction per clock cycle, while the full capabilities of the ALU are saturated by separate data streams in the fashion typical of the Harvard architecture.

A possible direction to a solution to this problem is provided by one of the internode coupling schemes we are investigating on the SAR Test System, where we are arranging to share data-stack memory between adjacent node processors. This provides an additional route for data to and from the ALU, providing a partial alleviation of the bottleneck problem, but not sufficient extra bandwidth to fully saturate the ALU under the most general conditions.

To achieve full access to the ALU bandwidth, additional ports are required. My proposal is that we extend the Novix architecture to include three data-stack ports in addition to the existing return stack, all capable of parallel activity on the same clock cycle. Figure 3 sketches the basic configuration. The combination of this extension with suitable external access to the stacks should be extremely powerful, providing full access to both ALU and main memory, in the spirit of the Harvard architecture, but maintaining the full programming flexibility of the SID-based Von Neumann design.

There are many further questions and comments that arise on examining this proposal, more than can be covered in this brief discussion. To set the tone, however, let me comment that the unadorned 32-bit version of the proposed design has a natural pin count of more than 320. Obviously, we must not be discouraged if the development path seems a little prickly.

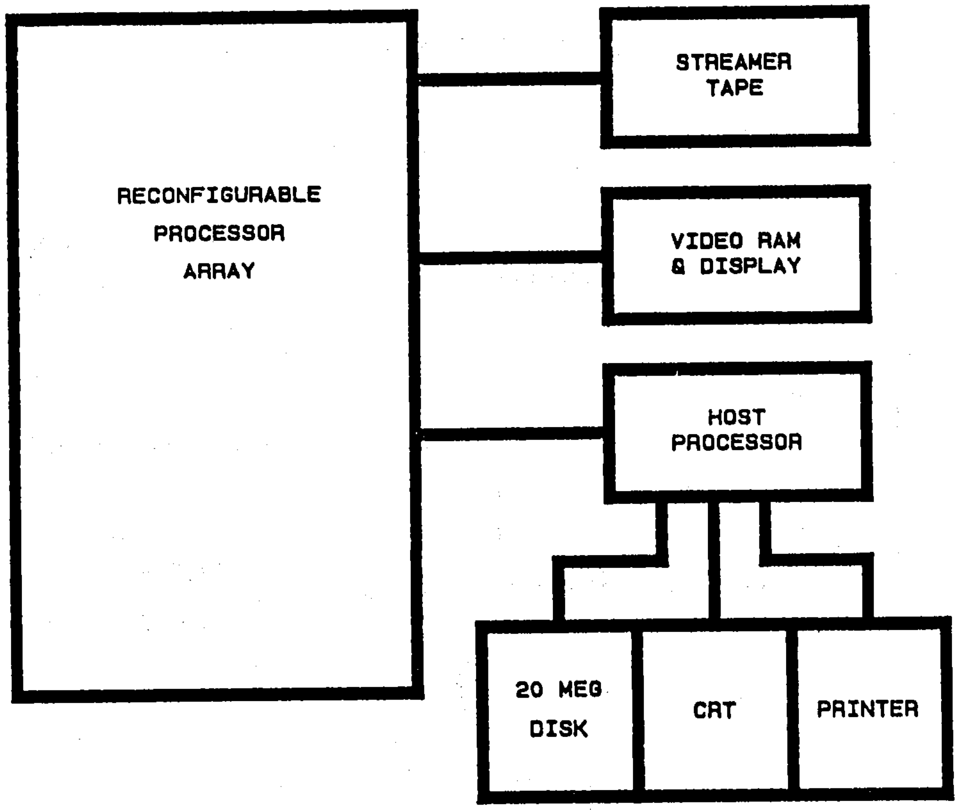

Figure 1. SAR Processor Test System block diagram.

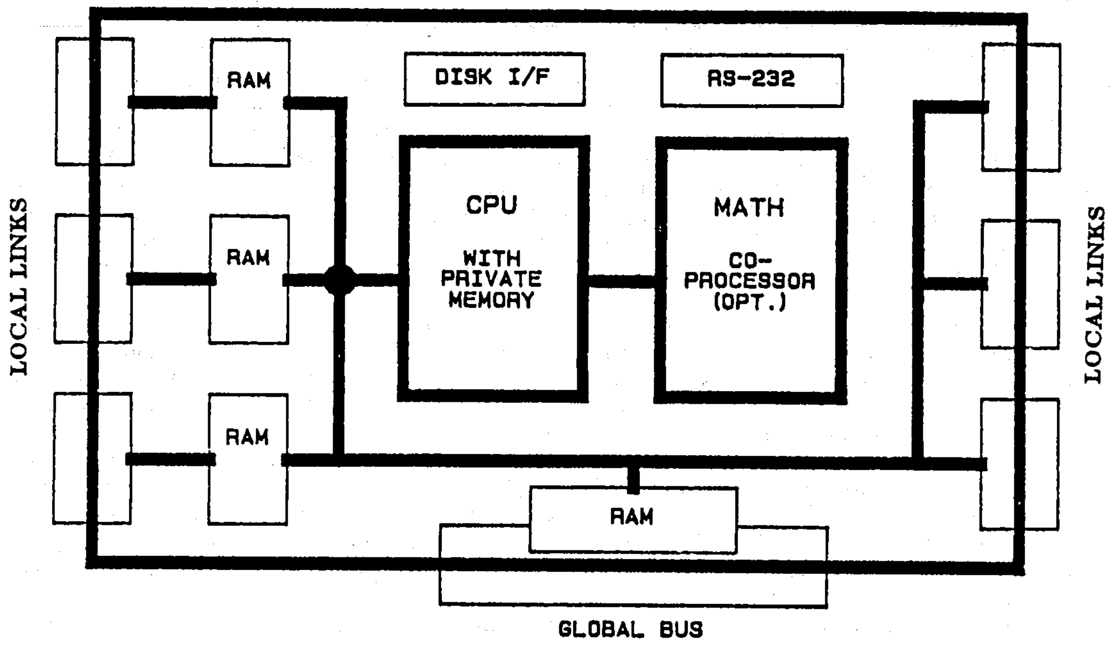

Figure 2. Node Processor block diagram.

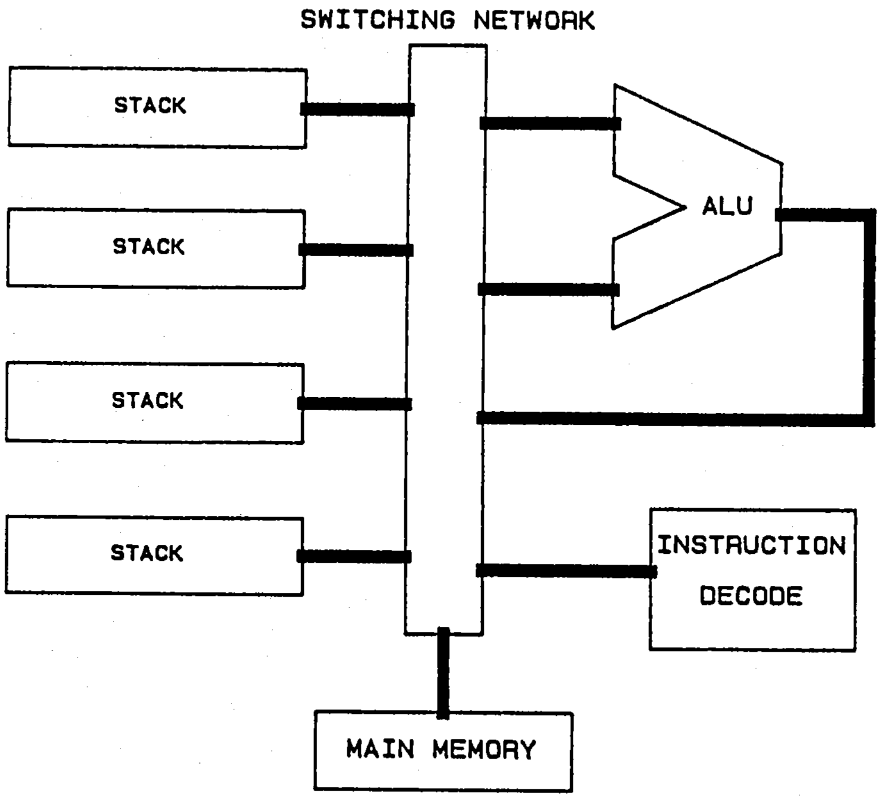

Figure 3. Proposed SID Processor architecture.