32 Bit RTX Chip Prototype

Philip J. Koopman, Jr.
WISC Technologies, Inc.
5551 Beacon St.
Pittsburgh, PA 15217

This unrefereed paper was presented at the 1988 Rochester Forth Conference
and has been included here at the discretion of the editors.
It does not appear in the Conference Proceedings.

Introduction

WISC Technologies and Harris Semiconductor have jointly developed a 32-bit member of the Harris Real Time Express (RTX) family. The result is the RTX 32P, a prototype implementation of the WISC Technologies CPU/32 processor that was previously implemented using discrete components. The RTX 32P is now fabricated and operational as a 2-chip microprocessor. It is being used as a research tool for a commercially available implementation now under development.

The Real Time Express (RTX) Family

The Real Time Express processor family is a product line of standard and semi-custom microprocessor systems designed especially to meet the needs of real time control processing in embedded systems. By real time control, I mean processing in an environment that must operate with severe limitations on size, weight, cost, and power in hostile environments. In addition, a real-time embedded processor must be fast enough to react to external events in what is often an unpredictable world.

To meet the needs of real time embedded processing, the RTX family combines high performance, a high level of circuit integration, and the compact programs for which stack machines are known. This, along with the goal of compatibility with high level languages, is a achieved by using Forth-like stack architectures which are suitable for most high level languages.

Two other goals of the RTX family are support for application specific optimizations and a rapid time-to-market for companies using the product. These goals are accomplished by using the RTX processors as a component of Harris' standard cell design system.

Why a 32 Bit RTX?

First, I should mention that the 16-bit RTX 2000 is sufficiently powerful to handle a wide variety (in fact, probably most) of today's real time embedded control applications. It can do so faster and with a higher level of integration than other processors currently in use. However, we all know the history of computation. Once we get to the top of a hill to solve today's applications, we discover a whole new mountain of more complex applications that we didn't even dream of before.

While no-one yet knows the full spectrum of real time control problems that will be solved with a 32 bit processor, we can make some generalizations and offer some examples that are known even today. There are two major reasons that system designers turn to 32 bit processors: wider data paths and larger memory address spaces. Applications which require 32 bit arithmetic

obviously are better supported by 32-bit hardware. Applications which require high precision include: high resolution laser graphics (such as a Postscript implementation), robot control equations of motion, and high precision signal processing.

Applications which require a large data memory address space also run more efficiently on a 32-bit processor. Note that I said *data* memory. The RTX stack machines typically use a very small code space, less than 64K bytes. However, some applications, like graphics, are inherently memory-intensive. For example, at 300 dots per inch, a laser printer uses a megabyte of memory to store a single 8 1/2 by 11 inch page of graphics.

RTX 32 Bit Architecture

As I mentioned in the introduction, the RTX 32 bit architecture will be based on work done by WISC Technologies in stack-based machines with writable microcode memory (sometimes called control store). Thus, the current prototype and successor commercial machines will support a 2-stack, 0-operand computation model, very similar to the Forth virtual machine. In addition, they will usually have large data stack and return stack RAMs on-chip, although off-chip stack RAM may be used for very large stacks if necessary on specialized versions. They will also have a combination of ROM and RAM microcode memory, with the commonly used words such as + and **SWAP** in ROM and application specific opcodes loaded into RAM during program execution.

Each fixed-length 32-bit instruction in the RTX-32 machines combines both an opcode in the high 9 bits of the instruction and a branch address field in the low 23 bits of the instruction. This allows executing an opcode as well as a subroutine call, subroutine return, or unconditional branch every memory cycle. This arrangement allows elimination of the wasted cycles spent on calling and returning from subroutines. In other words, the architecture can get subroutine calls "for free." The fact that the machine is microcoded allows the use of small opcodes. This is what leaves room for the large subroutine branch address.

An additional benefit of using microcoded control is that arbitrarily complex application specific instructions can be created. While simple instructions execute in two micro-cycles (the length of one memory cycle), time critical inner loops can be re-written using multi-cycle microcoded primitives to save time over executing a sequence of other opcodes. Often the savings in the inner loop can be two-to-one over normal opcodes. The philosophy behind the design style is to squeeze as much possible out of every memory cycle available to the processor. Since an ALU can be cycled two times in the time it takes a processor to access off-chip memory, why not put those two cycles to work executing compound instructions?

A Prototype Implementation: RTX 32P

My previous comments have been about the RTX-32 in general. We envision having several versions of the RTX-32 built upon a common processor core to meet the needs of different application areas. As a first significant step towards that end, we have designed and fabricated a prototype processor, the RTX 32P.

The RTX 32P is a 2-chip version of the RTX-32 architecture. It is built using the 2.5 micron Harris CMOS standard cell library directly from the schematics used for the discrete system implementation. As an example of how powerful the standard cell approach is (and how easy it will be for customers desiring to combine their TTL circuits onto RTX microcontrollers), I spent 31 calendar days at Harris from the time we started with a blank slate until the time the schematic was entered, simulated, and verified. And, I even managed a weekend off to visit Disney World.

All chip sets run at over 8 MHz under typical operating conditions. This gives up to four million Forth opcode/subroutine call pairs per second. There are 512 words of 32 bits each on the

data and return stacks on-chip, as well as 2K words of 30 bits each of microcode RAM, supporting 256 possible opcodes. An all-RAM microcode memory was chosen to allow maximum flexibility in using the chip set for research and development. With all that RAM, you can see why we used two chips.

The commercial version of the chip, which will probably be called the RTX 4000, will have all the processor logic, both stacks, and a combination of microcode RAM and ROM on a single chip.

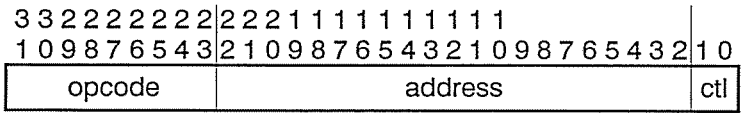
Integrated Software Development Environment

An important aspect of product design that is often left out of talks such as this is the issue of software. A unified software development environment for the RTX family is a cornerstone of the Harris product support philosophy. Despite their different implementations, the RTX 2000 and the RTX 4000 will enjoy a software environment that permits easy porting of code between the two systems. A key component of the RTX family will be a software development environment attuned to the needs of engineers solving problems in Forth, C, and other languages.

Harris and WISC have already made significant progress in complementary areas of software environments independently. Together, we will make an environment that provides quick and accurate program development while making non-Forth programmers feel comfortable with the transition from conventional programming environments. The idea is to preserve those vital things of the Forth environment that are good: interactivity, simplicity, and conciseness – without alienating new users with the quirks found in many Forth systems.

Conclusions

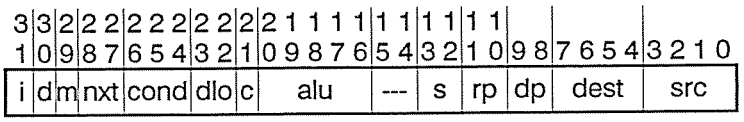
To answer the most obvious question that some of you have: a limited number of RTX 32P chip sets will be available to interested third party developers in the coming months. A commercial version of the design is currently in development with some announcement of further details expected in 1989.



Bits	Function
23-31	Opcode
2-22	Address for jump or call (word aligned)
0-1	Program flow control

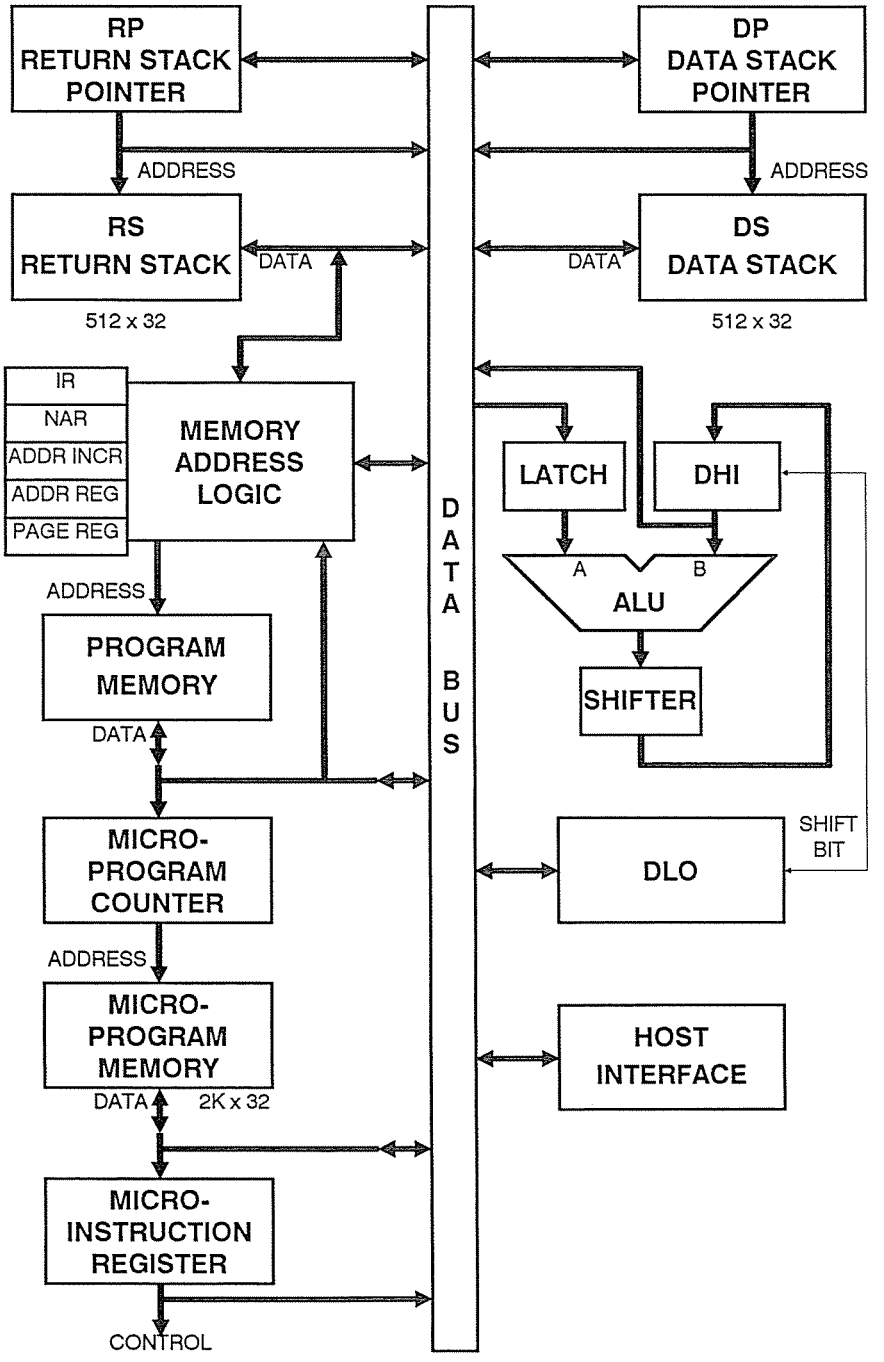
00 Jump
01 subroutine exit
10 subroutine call
11 unused

RTX 32P instruction format.



Bits	Function		
31	Increment return address register		
30	Begin decoding next instruction		
29	Increment MPC control		
27-28	Next microaddress constant bits		
24-26	Condition code select bits		
	000 0	011 Sign	110 Unused
	001 Not(Carry-out)	100 DLO lowest bit	111 1
	010 Not(Zero)	101 Unused	
22-23	DLO shift/destination control		
	00 Nop	10 Shift left	11 Load from bus
	01 Shift right		
21	ALU carry-in/shift-in bit		
16-20	ALU mode & function select		
	00000 A+not(CIN)	10001 A nor B	11010 B
	00010 A-B-CIN	10011 0	11011 A and B
	00110 A-B	10100 A nand B	11100 -1
	01001 A+B+CIN	10101 Not(B)	11110 A or B
	01100 A+A+CIN	10110 A xor B	11111 A
	10000 Not(A)	11001 A xnor B	
14-15	Unused		
12-13	ALU shifter control		
	00 Nop	10 Shift left 1 bit	11 Rotate right 8 bits
	01 Shift right 1 bit		
10-11	RP count control		
	00 Nop	10 Increment RP	11 Decrement RP
	01 Nop		
8-9	DP count control		
	00 Decrement DP	10 Nop	11 Nop
	01 Increment DP		
4-7	Bus destination select		
	0000 None	0101 Addr Latch	1010 Return Addr Incr
	0001 DP	0110 Status Reg	1011 RAM
	0010 DS	0111 Flag Reg	1100 RAM-BYTE
	0011 Unused	1000 RP	1101 Instruction Reg
	0100 RAM Page Reg	1001 RS	1110 Microcode RAM
0-3	Bus source select		
	0000 Host	0101 Multiply-select	1010 Return Addr Incr
	0001 DP	0110 Divide-select	1011 RAM
	0010 DS	0111 FLAGS	1100 RAM-BYTE
	0011 DLO	1000 RP	1101 Micro-PC
	0100 DHI	1001 RS	1110 Microcode RAM

RTX 32P microinstruction format.



RTX 32P block diagram.