
4.5 ARCHITECTURE OF THE HARRIS RTX 2000

4.5.1 Introduction

The Harris Semiconductor RTX 2000 is a 16-bit stack processor that is a descendent of the Novix NC4016. The RTX 2000 has a very high level of integration. It includes not only the core processor, but also stack memories, a hardware multiplier, and counter/timers on a single chip.

4.5.2 Block diagram

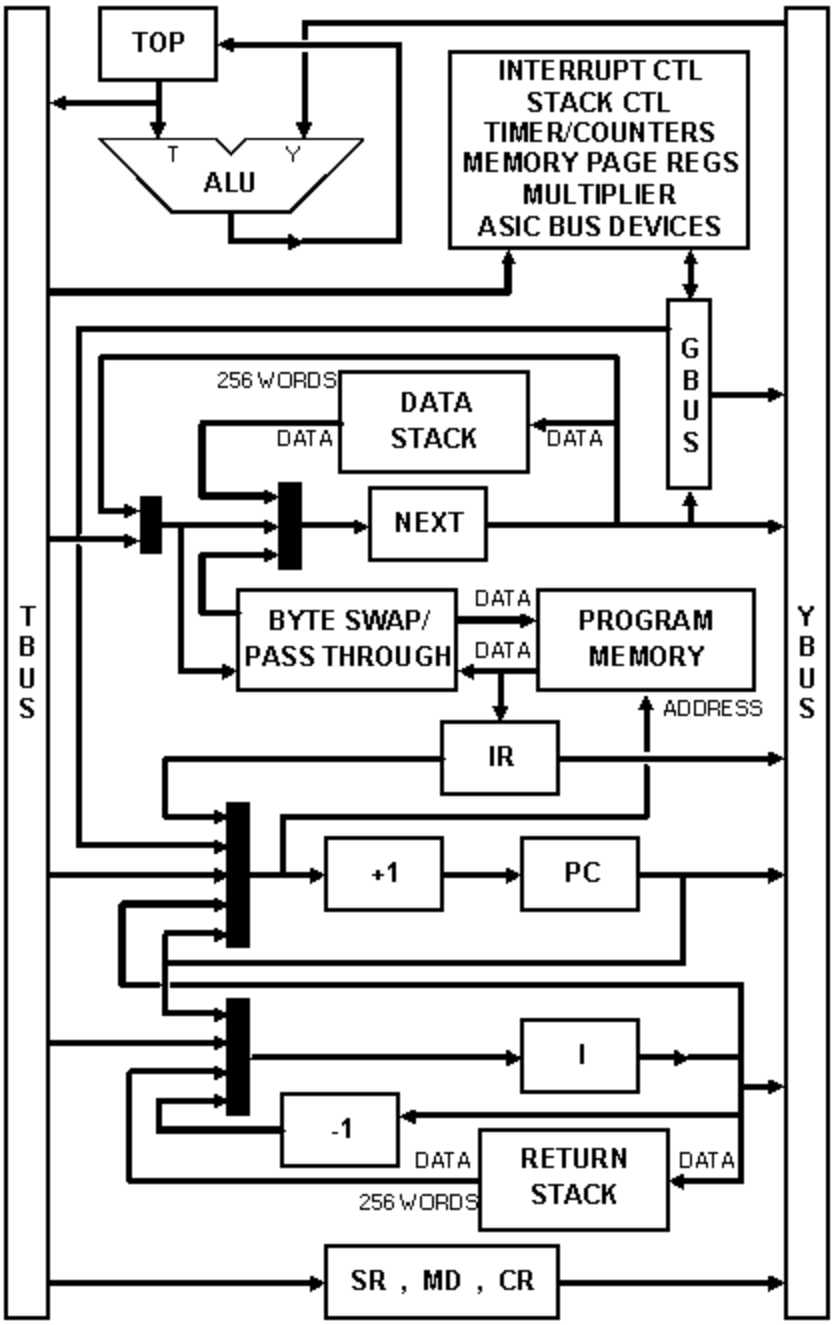


Figure 4.8 -- RTX 2000 block diagram.

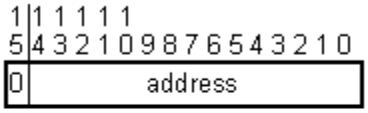
Figure 4.8 shows the block diagram of the RTX 2000. The major difference between the RTX 2000 and the NC4016 are the on-chip resources in addition to the CPU core. These resources include: a 256 element return stack, 256 element data stack, 16 x 16 bit single cycle hardware multiplier, three counter/timers, and a prioritized vectored interrupt controller. Apart from the on-chip stacks, all the RTX 2000's extra features are accessed via the I/O bus (called the G bus in the NC4016 and the ASIC Bus in the RTX 2000).

Other enhancements to the original Novix design available in the RTX 2000 include: a byte swapping capability for manipulation of 8-bit data, the ability to jump between adjacent memory blocks when performing conditional branches, and interrupt on stack overflow/underflow.

Another feature of the RTX 2000 is the on-chip memory page controller logic. This allows extending the 32K word program memory limit by specifying separate page registers for the code segment, the data segment (for fetches and stores), the user memory base address and page registers (for relocating the user variable area), and the index page register (for extending the value of the Return Stack address). Since the Return Stack holds a full 21 bits, subroutine calls can be made anywhere in memory with a special far call instruction sequence that saves the full return address in a single Return Stack location.

4.5.3 Instruction set summary

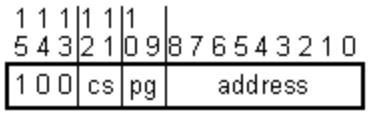
The instructions of the RTX 2000 are quite similar in function to those of the NC4016, but are sufficiently different in format to merit a separate description. Figure 4.9 shows the instruction formats for the RTX 2000.



Bits	Function
15	Constant value of 0
0-14	Subroutine address

Figure 4.9(a) -- RTX instruction formats -- subroutine call.

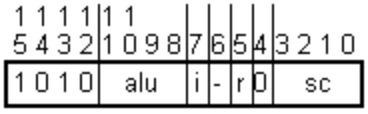
Figure 4.9a shows the instruction format for subroutine calls. In this format, the highest bit of the instruction is set to 0, and the remainder of the instruction is used to hold a 15-bit subroutine address. This limits programs to 32K words of memory.



Bits	Function
13-15	Constant 100 specifies conditional instruction
11-12	Condition select 00 Branch / DROP T if T = 0 01 Branch if T=0 / DROP T 10 Unconditional branch 11 Decrement INDEX & branch if not 0
9-10	Page control for high address bits 00 Same page 01 Next page 10 Page 0 11 Previous page
0-8	Low 9 bits of branch address

Figure 4.9(b) -- RTX instruction formats -- conditional branch.

Figure 4.9b shows the conditional branch instruction format. Bits 11 and 12 select either a branch if T is zero (with either a conditional or unconditional popping of the data stack), an unconditional branch, or a decrement and branch-if-zero using the index register for implementing loops. Bits 0-8 specify the lowest 9 bits of the target address, while bits 9 and 10 control an incrementer/decrementer for the upper 6 bits of the branch address to allow branching within the same 512 byte memory page, to adjacent pages, or to page 0.



Bits	Function
12-15,4	Constant 1010,0 specifies normal ALU operation
8-11	ALU function select 0000 T 0001 0010 T and Y 0011 T nor Y 0100 Y -T 0101 Y -T -borrow 0110 T or Y 0111 T nand Y 1000 T + Y 1001 T + Y + carry 1010 T xor Y 1011 T xnor Y 1100 T -Y 1101 T -Y -borrow 1110 1111 Y
7	Logical inverse bit (1's complement)
5	Subroutine return
0-3	Shift control 0000 none 0001 0 0010 Shift left 0011 Shift left with carry 0100 Logical shift right/carry 0101 Arithmetic shift right/carry 0110 Logical shift right 0111 Arithmetic shift right 1000 Shift N left 1001 Shift N left/carry 1010 32 bit shift left 1011 32 bit shift left/carry 1100 32 bit logical shift right/cy 1101 32 bit arithmetic shift right/cy 1110 32 bit logical shift right 1111 32 bit arithmetic shift right

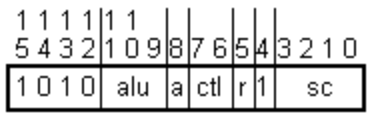
Figure 4.9(c) -- RTX instruction formats -- ALU operation.

Figure 4.9c shows the format of the ALU instruction. Bits 0-3 control the

operation of the shifter that shifts the output of the ALU.

Bit 5 of the ALU instruction indicates a subroutine return operation. This allows subroutine returns to be combined with preceding arithmetic operations to obtain "free" subroutine returns in many cases.

Bits 8-11 select an ALU function, with bit 7 controlling whether the output of the ALU is inverted.

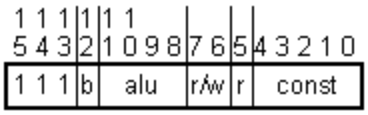


Bits	Function	
12-15,4	Constant 1010,1 specifies multi-step ALU operation	
9-11	ALU function select	
	000 T	100 T + Y
	001 T and Y	101 T xor Y
	010 T -Y	110 Y - T
	011 T or Y	111 Y
8	Register access	
	0 MD	1 MD*2 or SR
6-7	Multi-step special control field	
5	Subroutine return	
0-3	Shift control	
	0000 none	1000 Shift N left
	0001 0	1001 Shift N left/carry
	0010 Shift left	1010 32 bit shift left
	0011 Shift left with carry	1011 32 bit shift left/carry
	0100 Logical shift right/carry	1100 32 bit logical shift right/cy
	0101 Arithmetic shift right/carry	1101 32 bit arithmetic shift right/cy
	0110 Logical shift right	1110 32 bit logical shift right
	0111 Arithmetic shift right	1111 32 bit arithmetic shift right

Figure 4.9(d) -- RTX instruction formats -- ALU operation (multistep mode).

Figure 4.9d shows the format of the ALU instruction in multi-step mode. This format is quite similar to the ALU instruction format. Bits 0-3 select a shift control function, bit 5 controls the subroutine return function, and bits 9-11 selects the ALU operation.

In multi-step mode, bit 8 selects either the Multiply/Divide register or the Square Root register for special operations, while bits 6-7 select special multi-step control functions. A primary use of the multi-step mode is for repeated multiplication and division step operations.

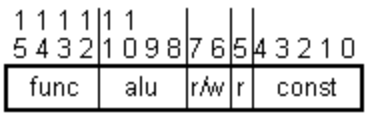


Bits	Function	
13-15	Constant 111 specifies memory reference	
12	Byte access control	
	0 Word access	1 Byte access
8-11	ALU function select	
	0000 T	1000 T + Y
	0001	1001 T + Y + carry
	0010 T and Y	1010 T xor Y
	0011 T nor Y	1011 T xnor Y
	0100 Y - T	1100 T - Y
	0101 Y - T - borrow	1101 T - Y - borrow
	0110 T or Y	1110
	0111 T nand Y	1111 Y
6-7	Read/write and routing control	
5	Subroutine return	
0-4	Auto-increment/decrement constant	

Figure 4.9(e) -- RTX instruction formats -- memory reference.

Figure 4.9e shows the format of a memory reference instruction. These instructions take two clock cycles: one for the instruction fetch, and one clock cycle for the actual reading or writing of the operand. The address for the memory access is always taken from the T register. Bit 12 selects either a byte or word memory access. Since the RTX 2000 uses word memory addresses, this bit selects a "low half/high half" or "full word" operation at the selected memory word.

Bits 6 and 7 indicate whether the operation is a memory read or write as well as other control information. Bits 0-4 specify a small constant that can be added or subtracted to the T value to perform and autoincrement or autodecrement addressing function. Bits 8-11 of this instruction specify the same ALU functions as the ALU instruction format.



Bits	Function
12-15	Function selection 1011 I/O, Register, Short literal 1100 User memory access 1101 Long literal access
8-11	ALU function select
	0000 T 0001 0010 T and Y 0011 T nor Y 0100 Y - T 0101 Y - T - borrow 0110 T or Y 0111 T nand Y
	1000 T + Y 1001 T + Y + carry 1010 T xor Y 1011 T xnor Y 1100 T - Y 1101 T - Y - borrow 1110 1111 Y
6-7	Read/write and routing control
5	Subroutine return
0-4	Literal/ASIC address/user memory offset

Figure 4.9(f) -- RTX instruction formats -- miscellaneous instructions.

Figure 4.9f shows the miscellaneous instruction format. This instruction can be used to read or write a word in the 32 word relocatable user space, saving the time taken to push a memory address on the stack before performing the fetch or store. It can also be used to transfer registers within the chip, or push either a 5 bit literal (in a single clock cycle) or a 16 bit literal (in two clock cycles) onto the stack. Bits 8-11 of this instruction specify the same ALU functions as the ALU instruction format.

The RTX 2000 is specifically designed to execute the Forth language. Because of the unencoded format of many of the instructions, machine operations that correspond to a sequence of Forth operations can be encoded in a single instruction. Table 4.3 shows the Forth primitives and instruction sequences that can be executed by the RTX 2000.

: (subroutine call)	AND
; (subroutine exit)	BRANCH
!	DROP
+	DUP
-	I
0	LIT
0<	NOP
0BRANCH	OR
1+	OVER
1-	R>
2*	R@
>R	SWAP
@	XOR

Table 4.3(a) RTX 2000 Instruction Set Summary -- Forth Primitives. (see

Appendix B)

```
inv shift                DUP @ SWAP
lit inv                  DUP nn G! inv
lit SWAP inv             DUP U! inv
lit SWAP op              DUP U@ op
nn inv (short literal)   nn G! inv
nn OVER op               nn G@ inv
nn SWAP op               nn G@ DROP inv
op shift                 nn G@ OVER op
! inv                    nn G@ SWAP op
! nn                     OVER inv shift
@ inv                    OVER SWAP op shift
@ nn                     OVER SWAP ! inv
@ SWAP inv               OVER SWAP ! nn
@ SWAP op                OVER SWAP @ op
?DUP ?BRANCH             SWAP inv shift
DDUP inv shift           SWAP DROP inv shift
DDUP nn SWAP op          SWAP DROP @ nn
DDUP !                   SWAP DROP DUP inv shift
DROP inv shift           SWAP DROP DUP @ nn ROT op
DROP lit inv             SWAP DROP DUP @ SWAP
DROP nn inv              SWAP OVER op shift
DROP DUP inv shift       SWAP OVER !
DUP inv shift            U! inv
DUP lit op               U@ op
DUP @ nn ROT op          U@ SWAP inv

Notation:      inv - 1's complement or no-op
               lit - long literal value
               nn - short literal value
               op - ALU operation
               shift - shift select or no-op
```

Table 4.3(b) RTX 2000 Instruction Set Summary -- Compound Forth Primitives.

```
INSTRUCTION      DATA STACK   RETURN STACK
nn G@                 -> N         ->
   Fetch the value from internal register or ASIC bus device
     nn (stored as a 5 bit literal in the instruction).

nn G!               N ->           ->
   Store N into the internal register or ASIC bus device nn
     (stored as a 5 bit literal in the instruction)

*               N1 N2 -> D3        ->
   Single clock cycle hardware multiply

*'                 D1 -> D2        ->
   Unsigned Multiply step (takes two 16 bit numbers and
     produces a 32 bit product).

*-                 D1 -> D2        ->
```

```
   Signed Multiply step (takes two 16 bit numbers and produces
     a 32 bit product).

*F                 D1 -> D2        ->
   Fractional Multiply step (takes two 16 bit fractions and
     produces a 32 bit product).

*/'                D1 -> D2        ->
   Divide step (takes a 16 bit dividend and divisor and
     produces 16 bit remainder and quotients).

*/''               D1 -> D2        ->
   Last Divide step (to perform non-restoring division fixup
     step).

2/                 N1 -> N2        ->
   Arithmetic shift right (same as division by two for
   non-negative integers.

D2/                D1 -> D2        ->
   32 bit arithmetic shift right (same as division by two for
   non-negative integers.

S'                 D1 -> D2        ->
   Square Root step.

NEXT                  ->        N1 -> N2
   Count-down loop using top of return stack as a counter.
```

Table 4.3(c) RTX 2000 Instruction Set Summary -- Special Purpose Words.

4.5.4 Architectural features

Like the NC4016, the internal structure of the RTX 2000 is optimized to provide single clock cycle instruction execution. All primitive operations except memory fetch, memory store, and long literal execute in a single clock cycle.

The RTX 2000 also allows some sequences of Forth instructions to be combined into a single instruction. A key capability is provided by the return bit in some formats that allows combining subroutine returns with ALU operations.

4.5.5 Implementation and featured application areas

The RTX 2000 is implemented on 2.0 micron CMOS standard cell technology, packaged in an 84 pin Pin Grid Array (PGA). The RTX 2000 runs at up to 10 MHz. A large advantage of standard cell technology is that RAM and logic may be mixed on the same chip, allowing both the return stacks and data stacks to

be placed on-chip.

Because the RTX 2000 executes most instructions, including conditional branches and subroutine calls, in a single cycle, there is a significant amount of time between the beginning of the clock cycle and the time that the memory address is valid for fetching the next instruction. This time is approximately half the clock cycle, meaning that program memory must be approximately twice as fast as the clock rate.

While the RTX 2000 was originally based on the NC4016 design, it has been substantially improved and does not have the hardware anomalies found on the NC4016.

The RTX 2000 is aimed at the high end 16 bit microcontroller market. Because it is implemented with semicustom technology, specialized versions of the processor can be made for specific design applications. Some possible applications include laser printer control, Graphics CRT display control, telecommunications control, optical character recognition, signal processing, and military control applications.

The information in this section is derived from the RTX 2000 Data Sheet (Harris 1988a) and the RTX 2000 Instruction Set Manual (Harris 1988b).

4.5.6 Standard cell designs

The Harris RTX 2000 derives many of its benefits from the fact that it is built using standard cell technology instead of a gate array. The difference is that in a gate array, the designer is customizing a regular pattern of preplaced logic gates on the silicon. In standard cell design, the designer is working with a library of logic functions that can be arbitrarily arranged on the silicon, as no predetermined gate arrangement scheme is used. While gate arrays with predefined memory areas are coming into use, the flexibility afforded by standard cell design techniques is not be equalled by a gate array approach.

Thus, the major differences between the NC4016 and the RTX 2000 become apparent: the RTX 2000 is able to take advantage of the flexibility of standard cells to include stack RAM cells on-chip. Because of this flexibility, a family of RTX 2000 processors with differing capabilities is planned using the same core processor as a large standard cell in the design process.

In addition to standard product versions of the RTX family, users that can benefit from application specific hardware. Examples of special-purpose hardware include serial communication ports, FFT address generators, data compression circuitry, or any other hardware that might otherwise have to be

built off-chip. With standard cell technology, users can have tailored versions of the chip made for their own use. This tailoring can include, as the process technology gets denser than 2.0 microns, a significant amount of program RAM and ROM on-chip.





NEXT SECTION


Phil Koopman -- koopman@cmu.edu