Understanding Classic, Nut and Saturn CPU Architectures — Threaded Mode

Martin Hepperle
Senior Member

Posts: 475
Threads: 61
Joined: May 2014

07-15-2025, 06:56 AM #1

In order to enhance my understanding of these three CPUs, I am drawing some register maps of the CPUs.
I want to differentiate the three core CPUs - not the complete systems with their peripherals.

Reading through various documents I still have some difficulties with a few details.

Field Selectors

- I understood that in the Nut CPU the field selectors can be used for accessing parts of the A, B, and, C registers (not for M, N).
- Can the various field selectors used for all registers on the Saturn CPU?

RPN Stack
Is it correct to say:

- The Classic CPU has the RPN stack in registers X,Y,Z,T inside the CPU (C=X, D=Y, E=Z, F=T).
- In the Nut these registers do not exist, the RPN stack is in RAM.
- The Saturn does not have them either, the RPL or BASIC stack is in RAM.

Return Stacks (Subroutine calls)
Is it correct to say:

- The Classic processor has no return stack pointer or registers (not programmable, HP-35).
- The Nut processor has 4 dedicated 16-bit registers forming the return stack. Are these part of the CPU as some documents suggest?
- The Saturn processor has one 16-bit register with the TOS address of the 8-level return stack, which is somewhere in RAM (i.e. a "classical" stack pointer).
- Is it possible to manipulate this register directly?
- The size of 8 entries was more or less a design decision, not enforced by the CPU design?

Maybe someone can clarify these items?

Thank you,
Martin

Amphitryon
Member

Posts: 230
Threads: 7
Joined: Jan 2025

07-15-2025, 09:54 AM #2

I may only answer few of your questions.

> **Martin Hepperle Wrote:** (07-15-2025, 06:56 AM)
>
> **Field Selectors**
>
> - I understood that in the Nut CPU the field selectors can be used for accessing parts of the A, B, and, C registers (not for M, N).

See ZENROM Usr Mnl, Chap. 6.2.1 The Accumulators (C, A and B) on p. 77, and Chap. 6.2.2 The Storage Registers (M, N and G), p. 79. See also list of Class-2 commands on p. 94 (only those use TE modifiers).

> **Quote:**
>
> **RPN Stack**
> Is it correct to say:
>
> - The Classic CPU has the RPN stack in registers X,Y,Z,T inside the CPU (C=X, D=Y, E=Z, F=T).

if I trust Tony Nixon's Classic Calculator Emulator (and I do trust it), checking several models with menu function 'For the inquisitive...' confirms C=X, D=Y, E=Z, F=T. IDK if Bernhard's emulators show the same.

> **Quote:**
>
> - In the Nut these registers do not exist, the RPN stack is in RAM.

It depends. The HP-41 keeps X, Y, Z, T, and L in the "first page" (the most important RAM registers defining the system). In all Voyagers the content of the X register is kept in the CPU while the other stack registers are somewhere in RAM. HP-10C: Y, Z, T, L in FC..FF, -11C and -16C: Y, Z, T, L in 00..03, -12C: Y, Z, T in 00..02 and L in EA, -15C: Y, Z, T in 00..02 and L in 13. My *NutEm/PC* offers a RAM-tv (tree view) if you are nosy.

> **Quote:**
>
> **Return Stacks (Subroutine calls)**
> Is it correct to say:
>
> - The Classic processor has no return stack pointer or registers (not programmable, HP-35).

IDK if it's possible to push, pull or drop a RTN address, but there must be two registers to hold the 2-level RTN stack me think.

> **Quote:**

- The Nut processor has 4 dedicated 16-bit registers forming the return stack. Are these part of the CPU as some documents suggest?

Yes, the RTN stack in "first page" registers a and b is for user code, not for MCode.

Find Reply

teenix
Senior Member

Posts: 2,373
Threads: 135
Joined: May 2016

07-15-2025, 11:02 AM #3

> **Martin Hepperle Wrote:** (07-15-2025, 06:56 AM)
>
> [*]The Classic processor has no return stack pointer or registers (not programmable, HP-35).

Classics have a single register for a subroutine return address - HP-35, 45, 65 etc
Woodstocks, Topcats Spice have a 2 level return stack so can support a nested subroutine

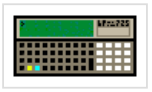
Note that the stacks are not always popped in relation to subroutine calls. Sometimes the stack is pushed when there are two pending returns so the stack can sometimes push values off the top in normal operation.

I'm not sure but I think the 2 level subroutine stack works similar to the 4 level XYZT stack

cheers

Tony

Find Reply



J-F Garnier
Senior Member

Posts: 1,165
Threads: 57
Joined: Dec 2013

07-15-2025, 12:06 PM (This post was last modified: 07-15-2025, 12:07 PM by J-F Garnier.) #4

> **Martin Hepperle Wrote:** (07-15-2025, 06:56 AM)
>
> In order to enhance my understanding of these three CPUs, I am drawing some register maps of the CPUs. I want to differentiate the three core CPUs - not the complete systems with their peripherals.

A good source for an overview is the hpmuseum.org/techcpu page

> **Quote:**
>
> **Field Selectors**
> I understood that in the Nut CPU the field selectors can be used for accessing parts of the A, B, and, C registers (not for M, N).
> Can the various field selectors used for all registers on the Saturn CPU?

As for the Nut CPU, the filed selectors of the original Saturn apply only on the working registers A..D, not on

the scratch registers R0-R4. However, the later Saturn cores (used on the 42, 48S, etc) introduce additional (and longer) opcodes to access the fields of the scratch registers too.
Note that the Saturn CPU introduced new fields: B for byte/character and A for 20-bit address handling.

> **Quote:**
>
> **RPN Stack**
> The Classic CPU has the RPN stack in registers X,Y,Z,T inside the CPU (C=X, D=Y, E=Z, F=T).

Not only the X, Y, Z, T registers are inside the CPU, but there are push, pop and roll opcodes, so this is really an hardware RPN stack.

> **Quote:**
>
> **Return Stacks (Subroutine calls)**
> The Saturn processor has one 16-bit register with the TOS address of the 8-level return stack, which is somewhere in RAM (i.e. a "classical" stack pointer).
> Is it possible to manipulate this register directly?
> The size of 8 entries was more or less a design decision, not enforced by the CPU design?

In the Saturn, the return stack is inside the CPU and there is no accessible stack pointer. It is only possible to push and pop values.
The 8 levels were notorious insufficient for the HP-71B and it was necessary to occasionality save some stack levels (in RAM) to make room for more levels.
It was the source of many bugs in the HP71:1BBBB and especially the HPIL:1A ROM, with a Memory Lost as a consequence of a stack overflow.
Curiously the stack depth was not increased in later Saturn versions, but RPL machines use RAM for the RPL return stack so the hardware stack was (maybe) less critical in RPL machines.

J-F

Website Find Reply

Martin Hepperle
Senior Member

Posts: 475
Threads: 61
Joined: May 2014

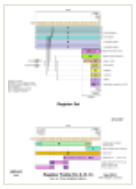
07-15-2025, 12:07 PM (This post was last modified: 07-16-2025, 08:52 AM by Martin Hepperle.) **#5**

The attached pictures show what I have drawn up so far for the Classic, the Nut and the Saturn CPUs (for comparison, I also made similar drawings for 6800, 8080, Z-80, 6502, 8088/86 and Capricorn).

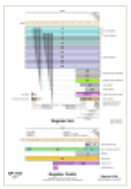
So far, my Classic CPU has neither a PC (program counter) and nor a SP (stack pointer, or set of fixed return stack registers).
For the equivalent Nut drawing I have added 4 return stack registers (4 nibbles wide) and one PC (4 nibbles wide) as well as the G register (1 byte) and the Q register (1 nibble). The field masks at the bottom should demonstrate the versatile access options.

I also tried to indicate by vertical lines with start/end dots which of the main registers can directly interact with others as in the Saturn drawing. It does not make sense to draw ALL possible interactions, but the main limitations connected to the main working registers should become visible.

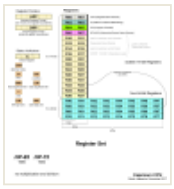
Classic CPU



Nut CPU



Saturn CPU



Capricorn CPU

Motorola 6800 CPU

Feel free to criticize - after all errors and omissions are repaired, I will make the final drawings available, of course.

Martin

[Edit: graphics updated: 16-JUL-2025]

Find Reply

Martin Hepperle
Senior Member

Posts: 475
Threads: 61
Joined: May 2014

07-15-2025, 12:22 PM (This post was last modified: 07-15-2025, 12:32 PM by Martin Hepperle.) #6

J-F Garnier Wrote: (07-15-2025, 12:06 PM)

[... Wrote:

pid='206894' dateline='1752562610']

Quote:

Return Stacks (Subroutine calls)
The Saturn processor has one 16-bit register with the TOS address of the 8-level return stack, which is somewhere in RAM (i.e. a "classical" stack pointer).

> Is it possible to manipulate this register directly?
> The size of 8 entries was more or less a design decision, not enforced by the CPU design?

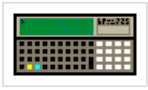
In the Saturn, the return stack is inside the CPU and there is no accessible stack pointer. It is only possible to push and pop values.
The 8 levels were notorious insufficient for the HP-71B and it was necessary to occasionality save some stack levels (in RAM) to make room for more levels.
It was the source of many bugs in the HP71:1BBBB and especially the HPIL:1A ROM, with a Memory Lost as a consequence of a stack overflow.
Curiously the stack depth was not increased in later Saturn versions, but RPL machines use RAM for the RPL return stack so the hardware stack was (maybe) less critical in RPL machines.

J-F

ah ... so I redrew the RSTK register for the Saturn and the Nut graphics as a set of 8 resp. 4 registers - in a different form (a set of staggered rectangles) as the visible part would be just the top level of a 8 resp. 4 register LIFO buffer/stack.

Thank you, Martin

Find Reply



J-F Garnier
Senior Member

Posts: 1,165
Threads: 57
Joined: Dec 2013

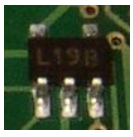
07-15-2025, 02:01 PM #7

> **Martin Hepperle Wrote:** (07-15-2025, 12:07 PM)
>
> Feel free to criticize - after all errors and omissions are repaired, I will make the final drawings available, of course.

In the Nut CPU (not sure about the "Classic" or Woodstock CPU), there is no "mantissa & sign" field, just the mantissa field (nib. 3-12) and the mantissa sign field (nib. 13).
And of course, there is a "Pointer" 1-nib field.

J-F

Website Find Reply



MeindertKuipers
Senior Member

Posts: 483
Threads: 45
Joined: Dec 2013

07-15-2025, 03:42 PM #8

The NUT status register is usually referred to as ST, and most documents also list the T register as the beeper output and the FI for the Flag Input. One could also count the BCD or HEX mode as a 1-bit register and there is a Carry bit.

Regards, Meindert

Find Reply

Christoph Giesselink
Member

Posts: 280
Threads: 72
Joined: Dec 2013

A good Saturn CPU overview is in

Jim Donnelly
An Introduction to HP 48 System RPL and Assembly Language Programming
https://literature.hpcalc.org/items/1475
p.200

> **Martin Hepperle Wrote:**
>
> Can the various field selectors used for all registers on the Saturn CPU?

No, the field selector is limited to the working registers A, B, C and D on the Saturn 1LF2 (level 0 instruction set) and 1LK7 (level 1 instruction set). Both CPU's are used in the HP71B. The 1LK7 CPU was used also in the HP18C and HP28C.

On later Saturn CPU's 1LU7, 1LR2, 1LR3, 1LT8, ... (with level 2 instruction set) the scratch registers R0-R4 can also be used with the field selector. The data pointers D0 and D1 have no field selector.

> **Quote:**
>
> RPN Stack
> The Saturn does not have them either, the RPL or BASIC stack is in RAM.

The Saturn CPU has no RPN Stack inside the CPU. Dependent on the implementation RPL/RPN you have the RPL stack in RAM which contains pointers to the stack objects somewhere in RAM or on RPN machines you have fixed addresses for the X,Y,Z,T registers in RAM.

> **Quote:**
>
> - The Saturn processor has one 16-bit register with the TOS address of the 8-level return stack, which is somewhere in RAM (i.e. a "classical" stack pointer).
> - Is it possible to manipulate this register directly?
> - The size of 8 entries was more or less a design decision, not enforced by the CPU design?

That's wrong. The Saturn CPU has only two 16bit registers, the Status and the In register. The 8-level return stack with a width of 20bit each (to hold the content of the 20bit PC counter) is implemented in hardware inside the CPU. This is the assembler code return stack for subroutines. So you're limited nesting assembler subroutines. Jim Donellys document shows that on the HP48 two Return Stack entries must be reserved for the interrupt system. When I remember correctly this is wrong. Under special circumstances, at an Alarm during the interrupt handling, the interrupt service routine needs three Return Stack registers. Never checked this, if this information is correct. So what happens when you push or pop more than the eight available registers? The Return Stack has a wraparound, when you push more than eight return addresses, the oldest one is overwritten with the last pushed address.

It's not possible to access the stack directly. The GOSUB/RTN and C=RSTK and RSTK=C opcodes are manipulate the Return Stack.

With a trick using the C=RSTK opcode multiple times you would be able to select and then read (C=RSTK) or write (RSTK=C) a special Return Stack element. With additional C=RSTK opcodes using the wraparound it would be possible to restore the old stack pointer position without other modifications. Never seen such an implementation in the wild.

Are the 8 entries a design decision? Yes and No, with 8 entries it's just simple to implement the registers wraparound in hardware, just do an AND operation on the stack register pointer with 7. On others sizes, which are not a power of 2, you have to implement a modulo operation in hardware.

What else?

Watch also the different behavior of the Carry flag between NUT and Saturn CPU. On the Saturn CPU only special opcodes modify the content of the Carry flag. On the NUT CPU, the Carry flag is regular cleared after every instruction, with the exception of some opcodes setting the Carry flag for the next instruction.

There's also a different behavior between NUT and Saturn CPU on arithmetic operations in decimal mode on a register containing pseudo-tetrades (very special, but here already seen in the wild on a HP41 and a HP48).

Website Find Reply

Martin Hepperle
Senior Member

Posts: 475
Threads: 61
Joined: May 2014

07-16-2025, 08:49 AM (This post was last modified: 07-16-2025, 09:40 AM by Martin Hepperle.) **#10**

Thank you all very much for your corrections and clarifications!

It is difficult to wade through this field of mines. I For the Classic CPU I also liked the graph in the US Patent 4,001,569 which showed the possible interactions between the registers. That's what I tried to indicated with the vertical lines between registers, but I did not include the connections to the flag registers to avoid cluttering the images too much.

I have updated the drawings accordingly.

When reading more about the Nut CPU family, I also stumbled about the term "Time Enable Fields" for the register field select masks - does anyone know the background why the word "Time" appears here? Maybe it refers to the clock cycles == times when the bits are read serially from the register? I also noticed this term in the Fairchild PPS 25 Chip Set Manuals.

Find Reply

brouhaha
Senior Member

Posts: 751
Threads: 64
Joined: Dec 2013

07-16-2025, 09:47 AM **#11**

> **Martin Hepperle Wrote:** (07-16-2025, 08:49 AM)
>
> .When reading more about the Nut CPU family, I also stumbled about the term "Time Enable Fields" for the register field select masks - does anyone know the background why the word "Time" appears here? Maybe

it refers to the clock cycles == times when the bits are read serially from the register? I also noticed this term in the Fairchild PPS 25 Chip Set Manuals.

Yes, that is exactly what it refers to. Which digit TIMES within the overall word time the arithmetic operation is enabled (active), thus performing the operation on a field (range) of digits within the word.

Find Reply

Jonathan Busby
Senior Member

Posts: 320
Threads: 39
Joined: Nov 2014

08-30-2025, 07:14 PM (This post was last modified: 09-11-2025, 08:42 PM by Jonathan Busby. *Edit Reason: Image URLs updated*) **#12**

Martin Hepperle Wrote: (07-15-2025, 12:07 PM)

The attached pictures show what I have drawn up so far for the Classic, the Nut and the Saturn CPUs (for comparison, I also made similar drawings for 6800, 8080, Z-80, 6502, 8088/86 and Capricorn).

So far, my Classic CPU has neither a PC (program counter) and nor a SP (stack pointer, or set of fixed return stack registers).
For the equivalent Nut drawing I have added 4 return stack registers (4 nibbles wide) and one PC (4 nibbles wide) as well as the G register (1 byte) and the Q register (1 nibble). The field masks at the bottom should demonstrate the versatile access options.

I also tried to indicate by vertical lines with start/end dots which of the main registers can directly interact with others as in the Saturn drawing. It does not make sense to draw ALL possible interactions, but the main limitations connected to the main working registers should become visible.

Classic CPU
Nut CPU
Saturn CPU
Capricorn CPU
Motorola 6800 CPU

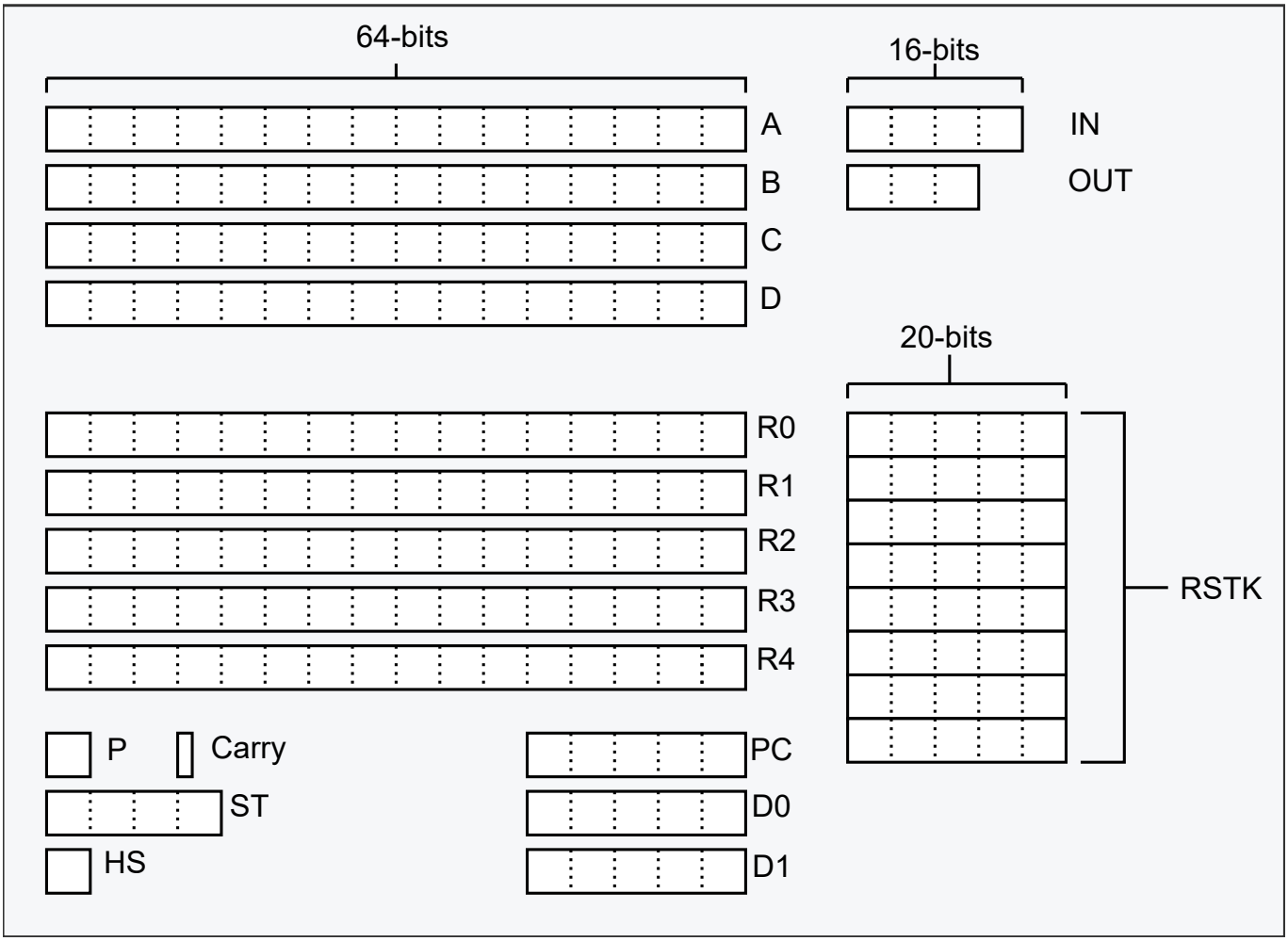
Feel free to criticize - after all errors and omissions are repaired, I will make the final drawings available, of course.

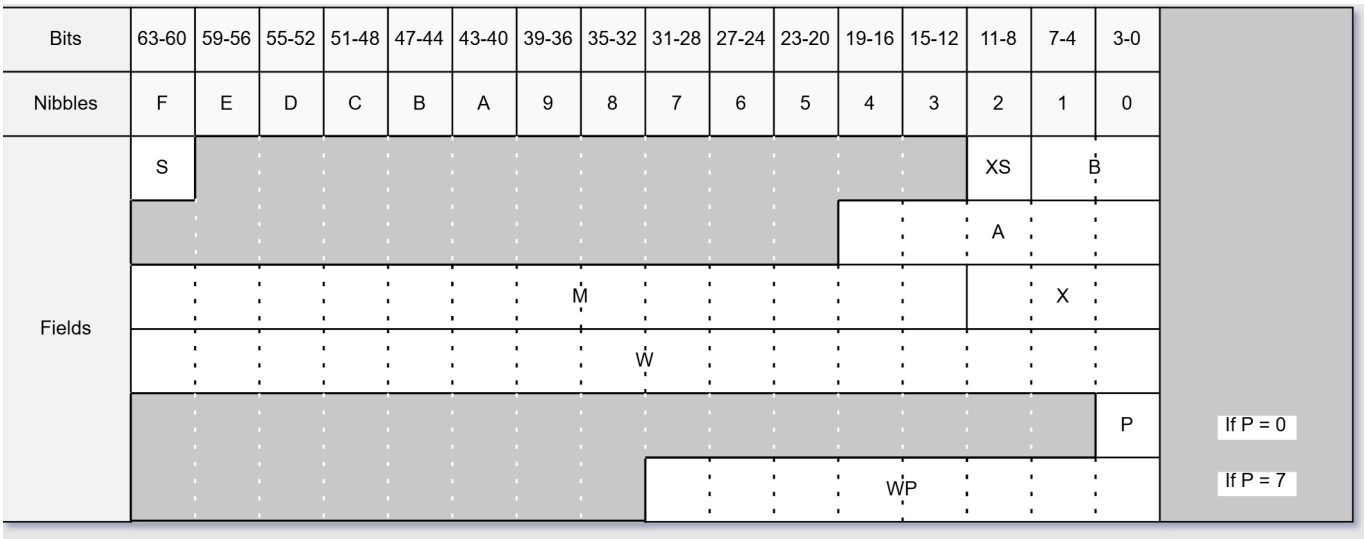
Martin

[Edit: graphics updated: 16-JUL-2025]

Those are some really nicely drawn and formatted register and register field diagrams 😃

The following are the register and register field diagrams that I created way back in 2019 for the Wikipedia HP Saturn article (Note that they're vector images, specifically SVGs—I've attached the originals) :





(Don't know why the above register fields diagram looks blurry—Possibly due to the MyBB SVG to raster rendering code?)

The register fields diagram isn't in the current version of the article : instead there's a crappy looking wikitable, due to the other main editor being obdurate.

Anyways, the article is a mess as it's a collision of my edits, another editor's edits, and random edits by a few other editors who apparently have no understanding of Saturn internals and who removed content that has made the article factually inaccurate.

I do plan on performing a total rewrite of the article when I have the time—perhaps you might want to contribute your Saturn register diagrams when that happens?

Regards,

Jonathan

Attached Files

HP_Saturn_register_diagrams.zip (Size: 12.39 KB / Downloads: 7)

There are only 1T types of people : Those who understand balanced ternary and those who don't

Find Reply

Jonathan Busby
Senior Member

Posts: 320
Threads: 39
Joined: Nov 2014

08-30-2025, 08:29 PM (This post was last modified: 08-31-2025, 06:45 PM by Jonathan Busby. *Edit Reason: Fixed two accidental ommisions*) **#13**

> **Quote:**
>
> **Christoph Giesselink Wrote:**
>
> Jim Donellys document shows that on the HP48 two Return Stack entries must be reserved for the interrupt system. When I remember correctly this is wrong. Under special circumstances, at an Alarm during the interrupt handling, the interrupt service routine needs three Return Stack registers. Never checked this, if this information is correct.

You are correct about alarm processing requiring three return stack levels, but, this requirement doesn't violate the rule that user code must provide two return stack levels for the interrupt system. AFAIK, when the interrupt system services timer 2, one return stack level is used by a GOSBVL call to the timer 2 service code—that uses up one return stack level making only one available. But, when the timer 2 service code starts processing alarms, it first saves two return stack levels to temporary locations, making three levels available. This is because there are calls to SAFESKIPOB which, in addition to saving the return address on the stack, also uses two return stack levels to save register contents. After alarm service is finished, the timer 2 service code restores the two saved return stack levels, and eventually returns so that two return stack levels are available for the rest of the interrupt processing code.

> **Quote:**
>
> With a trick using the C=RSTK opcode multiple times you would be able to select and then read (C=RSTK) or write (RSTK=C) a special Return Stack element. With additional C=RSTK opcodes using the wraparound it would be possible to restore the old stack pointer position without other modifications. Never seen such an implementation in the wild.

Are you sure about this? From all the documentation I've read, there is no "special return stack element." I've

tested popping more than eight return stack levels on real hardware and everything past the eighth pop (assuming the return stack is completely occupied by non-zero values) just returns zero. In fact, I don't see how one could go about determining how many return stack levels are currently occupied, because one would need to have a special non-zero sentinel value to achieve this and one would in addition need to guarantee that there is no contiguous sequence of one or more zero values starting at the bottom of the stack. But, and assuming arbitrary user machine code can manipulate the stack in any way it wants, one cannot guarantee the aforementioned conditions. In fact, if there are one or more contiguous zero values starting at the bottom of the stack, one cannot differentiate this from the same stack state wherein there is no contiguous sequence of zero values at the bottom of the stack. Indeed, arbitrary pushing to or popping from the return stack generates the same behavior as a stack with the previously mentioned sequence of contiguous zero values starting at its bottom.

Quote:

with 8 entries it's just simple to implement the registers wraparound in hardware, just do an AND operation on the stack register pointer with 7.

Actually, the AND operation is not needed : all one needs is a 3-bit register—I cannot imagine that the Saturn designers would have wasted logic resources on a superfluous logic operation.

In (System)Verilog, one would just need to declare a 3-bit register

Code:

```
logic [2:0] sp;
```

and perform

Code:

```
sp <= sp + 1;
```

or

Code:

```
sp <= sp - 1;
```

to push to or pop from the stack, respectively.

The Saturn hardware return stack is probably implemented (albeit likely with schematic capture) in a very similar way to what is described by the following SystemVerilog code :

Code:

```
module HW_RSTK(
  input CLK,
  input RST,
  input [19:0] data_in,
  output logic [19:0] data_out,
  input push, pop
);
```

```
logic [19:0] rstk [7:0];
```

Regards,

Jonathan

There are only 1T types of people : Those who understand balanced ternary and those who don't

Find | Reply

« Next Oldest | Next Newest »

Enter Keywords | Search Thread

View a Printable Version

Users browsing this thread: 1 Guest(s)

Powered By **MyBB**, © 2002-2025 **MyBB Group**.

Current time: 12-12-2025, 12:52 PM