

ERISC processor Figure /models07/ERISC

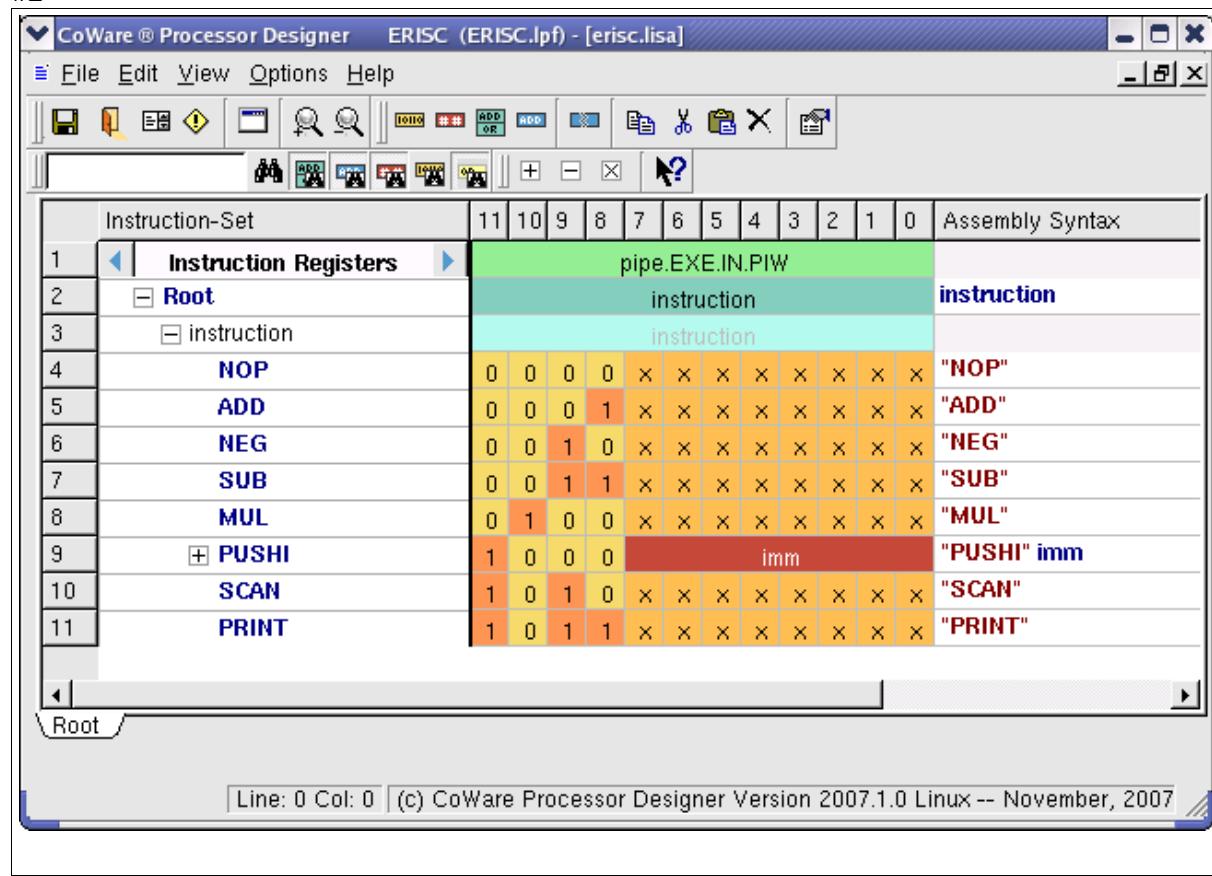
of LISA operations = Proc. No. + 4, i.e., ERISC1->5 Op ... ERISC 13-> 17

Instruction-Set		11	10	9	8	7	6	5	4	3	2	1	0	Assembly Syntax
1	◀ Instruction Registers ▶	IW												
2	└ Root	instruction		instruction										
3	└ instruction	instruction		instruction										
4	NOP	0	0	0	0	x	x	x	x	x	x	x	"NOP"	

Instruction-Set		11	10	9	8	7	6	5	4	3	2	1	0	Assembly Syntax
1	◀ Instruction Registers ▶	pipe.EXE.IN.PIW												
2	└ Root	instruction		instruction										
3	└ instruction	instruction		instruction										
4	NOP	0	0	0	0	x	x	x	x	x	x	x	"NOP"	
5	SCAN	1	0	1	0	x	x	x	x	x	x	x	"SCAN"	
6	PRINT	1	0	1	1	x	x	x	x	x	x	x	"PRINT"	

Instruction-Set		11	10	9	8	7	6	5	4	3	2	1	0	Assembly Syntax
1	◀ Instruction Registers ▶	pipe.EXE.IN.PIW												
2	└ Root	instruction		instruction										
3	└ instruction	instruction		instruction										
4	NOP	0	0	0	0	x	x	x	x	x	x	x	"NOP"	
5	ADD	0	0	0	1	x	x	x	x	x	x	x	"ADD"	
6	NEG	0	0	1	0	x	x	x	x	x	x	x	"NEG"	
7	SUB	0	0	1	1	x	x	x	x	x	x	x	"SUB"	
8	MUL	0	1	0	0	x	x	x	x	x	x	x	"MUL"	
9	+ PUSHI	1	0	0	0	imm		"PUSHI" imm						
10	SCAN	1	0	1	0	x	x	x	x	x	x	x	"SCAN"	
11	PRINT	1	0	1	1	x	x	x	x	x	x	x	"PRINT"	

#2



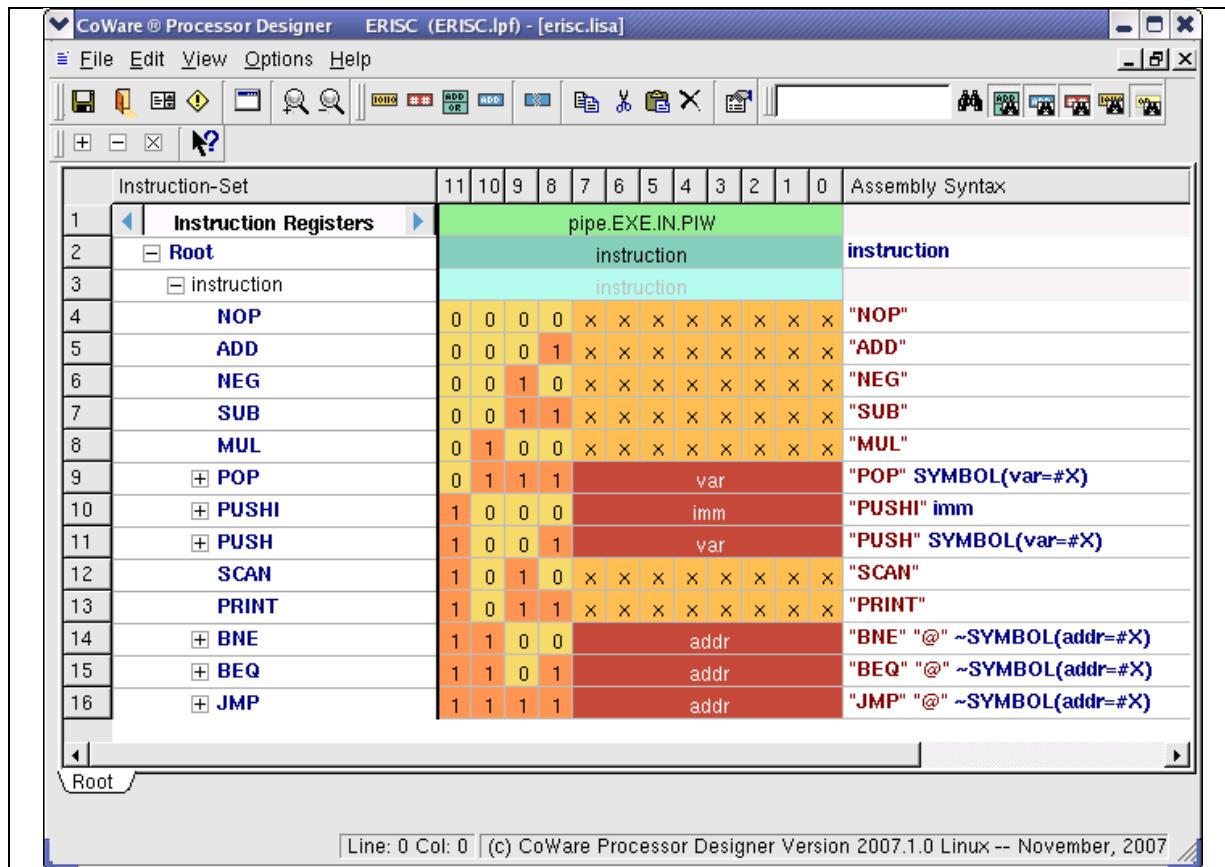
ERISC08isa.bmp

#3

	Instruction-Set	11 10 9 8 7 6 5 4 3 2 1 0	Assembly Syntax
1	Instruction Registers	pipe.EXE.IN.PIW	
2	Root	instruction	instruction
3	instruction	instruction	
4	NOP	0 0 0 0 x x x x x x x	"NOP"
5	ADD	0 0 0 1 x x x x x x x	"ADD"
6	NEG	0 0 1 0 x x x x x x x	"NEG"
7	SUB	0 0 1 1 x x x x x x x	"SUB"
8	MUL	0 1 0 0 x x x x x x x	"MUL"
9	+ PUSHI	1 0 0 0 imm	"PUSHI" imm
10	SCAN	1 0 1 0 x x x x x x x	"SCAN"
11	PRINT	1 0 1 1 x x x x x x x	"PRINT"
12	+ BNE	1 1 0 0 addr	"BNE" "@" ~SYMBOL(addr=#X)
13	+ BEQ	1 1 0 1 addr	"BEQ" "@" ~SYMBOL(addr=#X)
14	+ JMP	1 1 1 1 addr	"JMP" "@" ~SYMBOL(addr=#X)

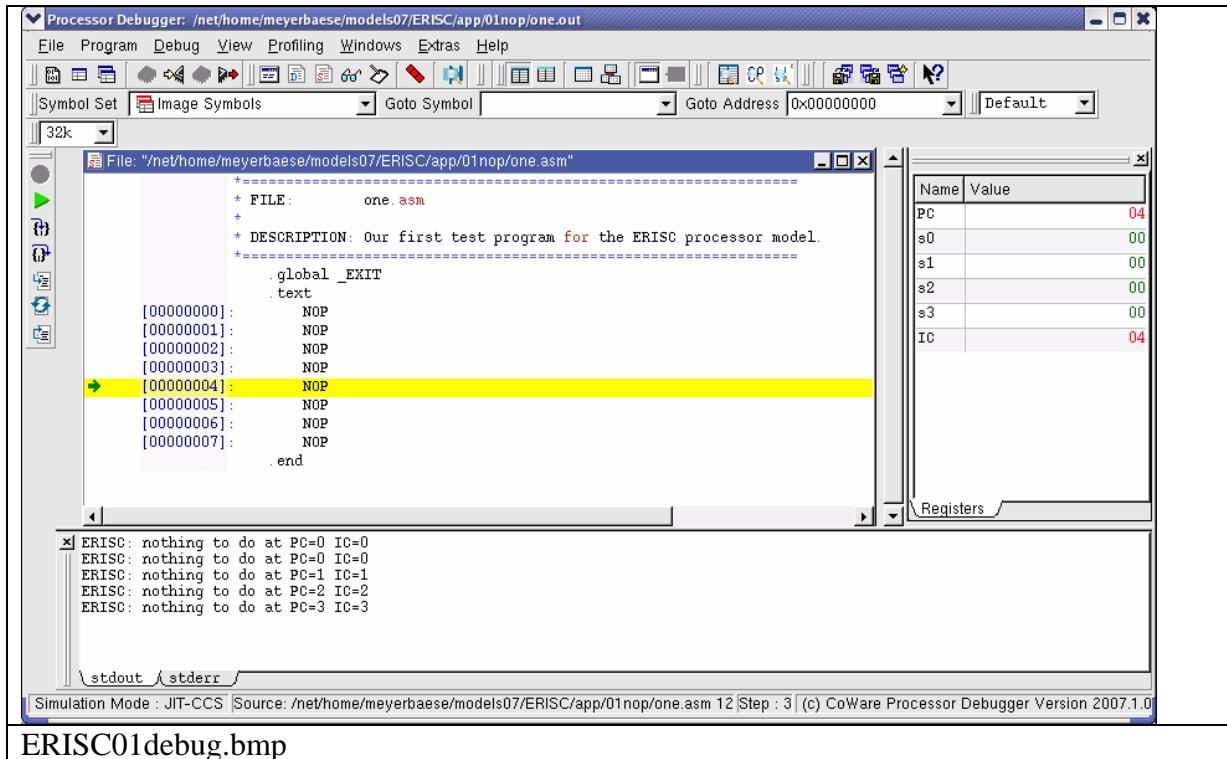
ERISC11isa.bmp

#4

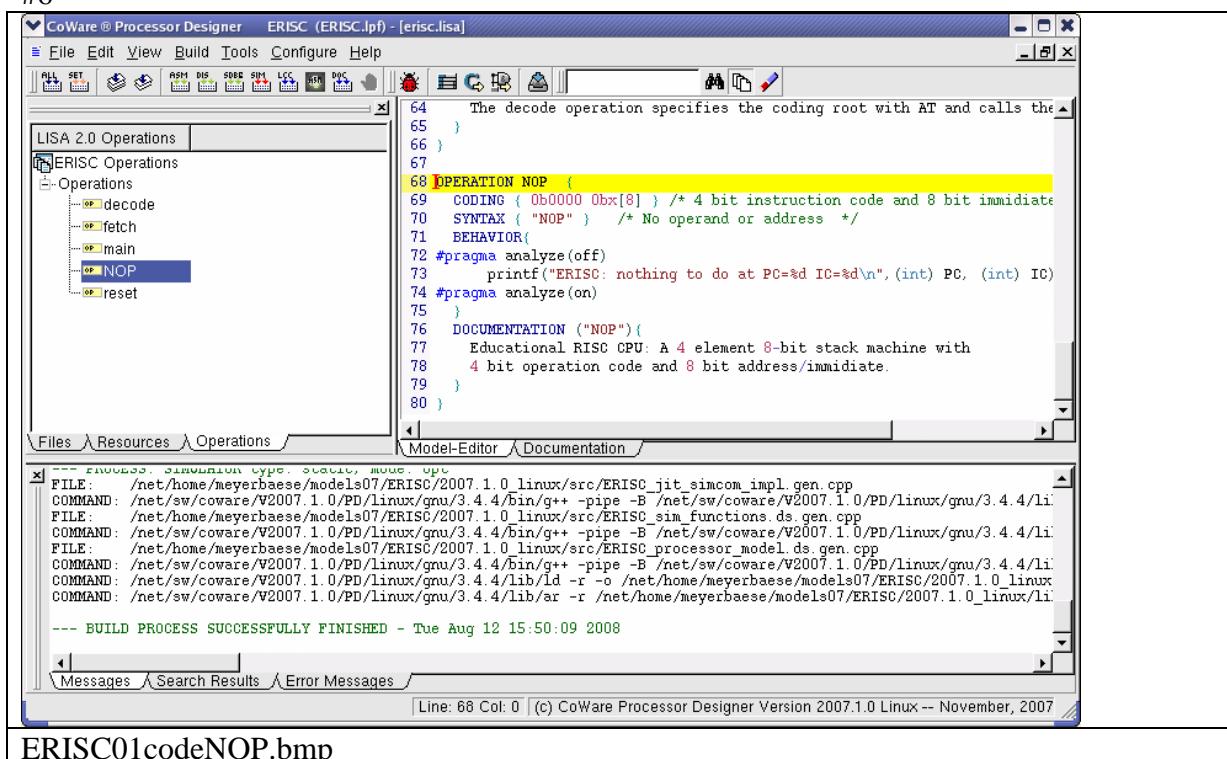


ERISC13isa.bmp

#5



#6



#7

The screenshot shows the CoWare Processor Designer interface. The main window displays the Model Editor with the following code:

```

1 VERSION("ERISC 1, 6/2008 by UMB")
2 // First version with one instructions: NOP
3 // Generate: ASM, DIS, SIM;
4 // Has no pipeline register, no I/O ports, and no data memory
5 RESOURCE {
6
7   RAM uint32 prog_mem {
8     BLOCKSIZE(12),           // Bitwidth is 16
9     SIZE(128),              // Total of 2^7 words
10    FLAGS(R,X);            // Read and execute indicates program memory
11    ENDIANCESS(BIG);       // Use for simulator
12
13  },
14
15
16  PROGRAM COUNTER Tclocked<uint8> PC; // Program counter
17  REGISTER Tclocked<int8> s3,s2,s1,s0; // Register stack, s0 ist top-of
18  unsigned bit[12] IW;                  // The instruction word
19  REGISTER Tclocked<int8> IC;          // Simulator only: Instruction counter
20
21
22 OPERATION reset {
23
24   BEHAVIOR {
25     s0=e1=s2=s3=0; // Set all stack registers to zero
26     IC = 0;
27     PC = LISA_PROGRAM_COUNTER;
28     prog_mem.reset();
29   }
30   DOCUMENTATION "Reset operation"
31
32
33 }
```

The terminal window below shows build logs:

```

ALL RIGHTS RESERVED
This program is proprietary and confidential information of Coware, Inc.
and may be used and disclosed only as authorized in a license agreement
controlling such use and disclosure.

++ Using Project File: /net/home/meyerbaese/models07/ERISC/ERISC.lpf
-- PROCESS: MODEL
-- Processed 5 LISA 2.0 operations.

-- BUILD PROCESS SUCCESSFULLY FINISHED - Tue Aug 12 15:46:25 2008
```

Line: 9 Col: 0 (c) CoWare Processor Designer Version 2007.1 Linux -- November, 2007

ERISC01code.bmp

#8

The screenshot shows the Processor Debugger interface. The assembly code window displays the following code:

```

=====
* FILE:          io.asm
* DESCRIPTION:  Short I/O test program for the ERISC processor model.
=====

.global _EXIT
.text
[00000000]: NOP
[00000001]: NOP
[00000002]: SCAN
[00000003]: NOP
[00000004]: NOP
[00000005]: PRINT
[00000006]: NOP
[00000007]: NOP
.end
```

The Registers window shows the following values:

Name	Value
iport	03
oport	03

The terminal window shows the following logs:

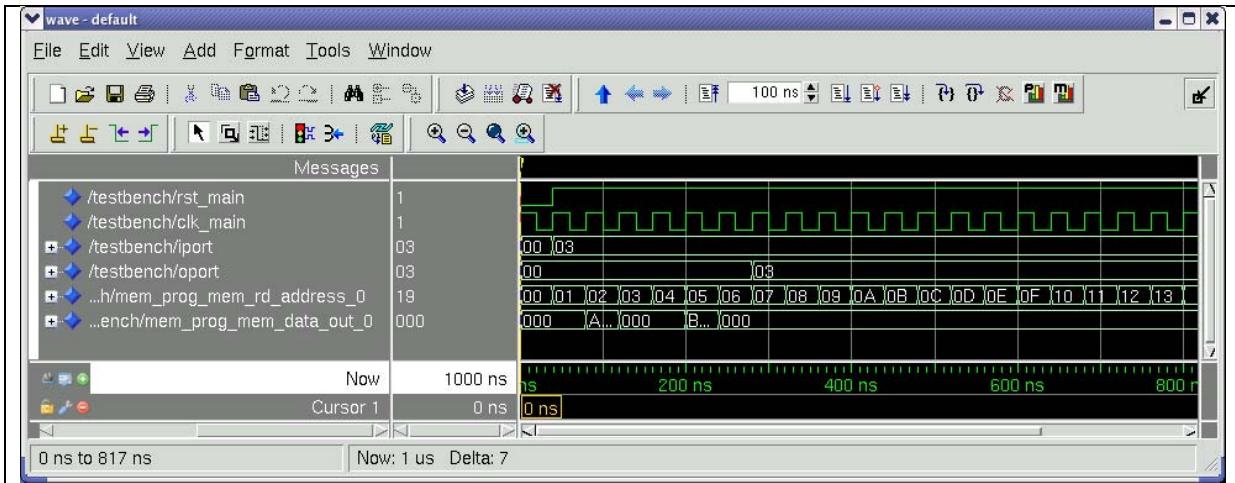
```

ERISC: nothing to do at PC=1 IC=1
ERISC: nothing to do at PC=2 IC=2
ERISC: nothing to do at PC=4 IC=4
ERISC: nothing to do at PC=5 IC=5
```

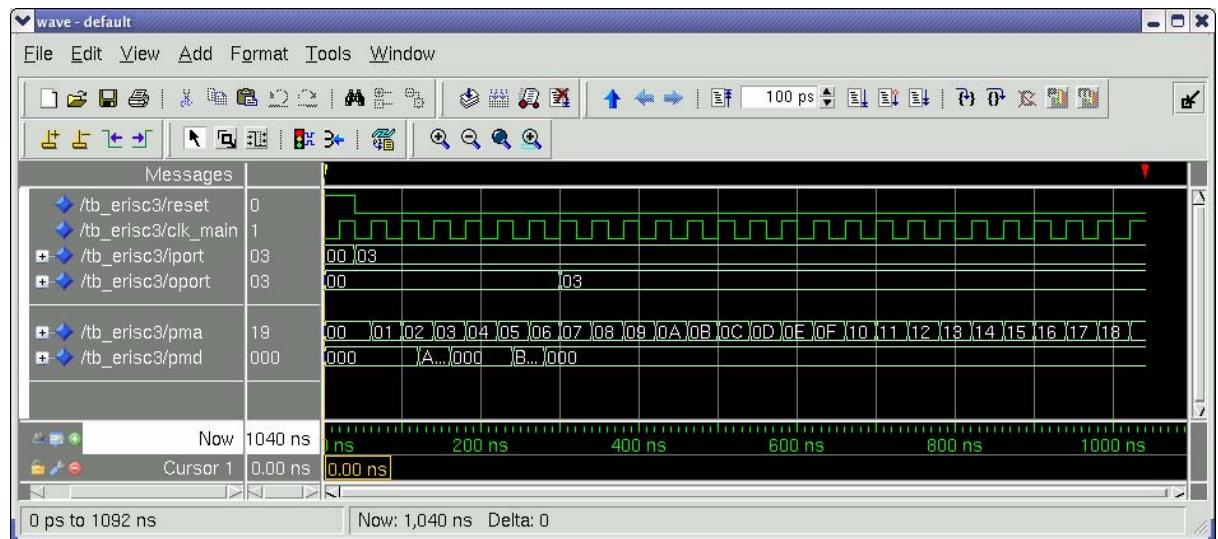
Simulation Mode: JIT-CCS Source: /net/home/meyerbaese/models07/ERISC/app/03io/io.asm Step : 7 (c) CoWare Processor Designer Version 2007.1

ERISC03debug.bmp

#9

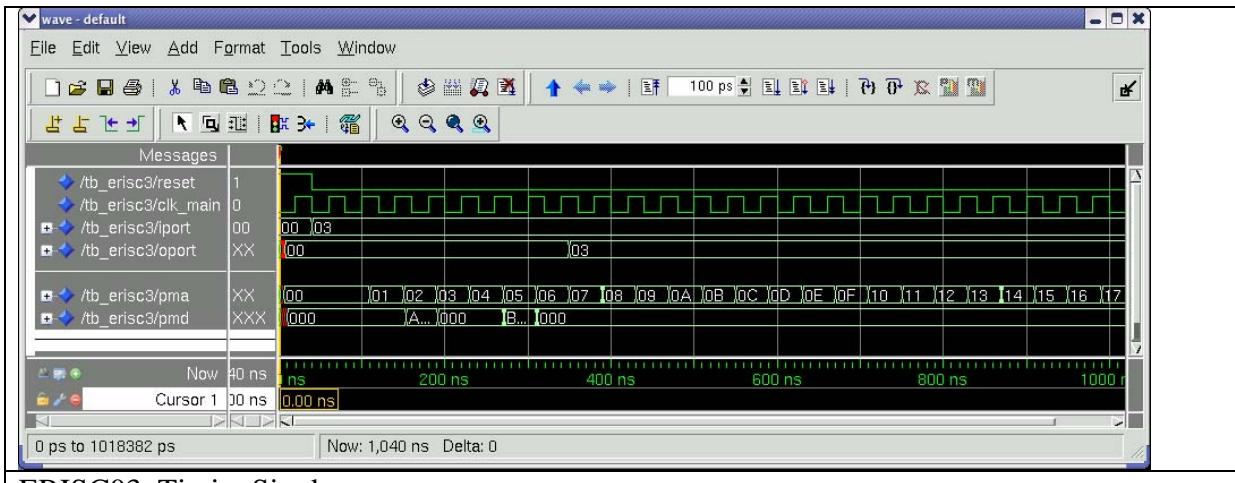


ERISC03funSim.bmp



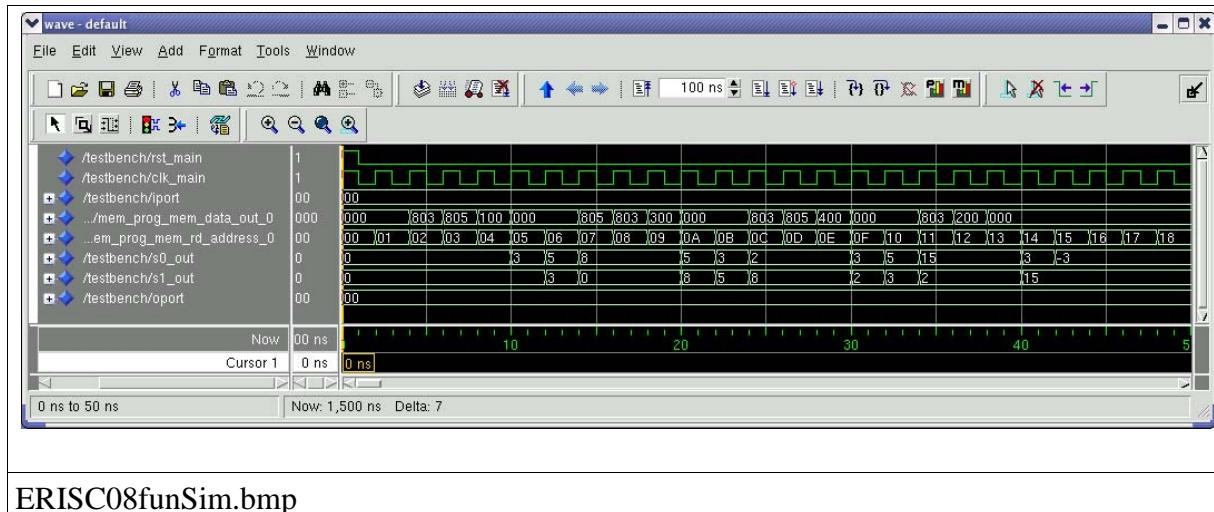
ERISC03xfunSim.bmp

#11



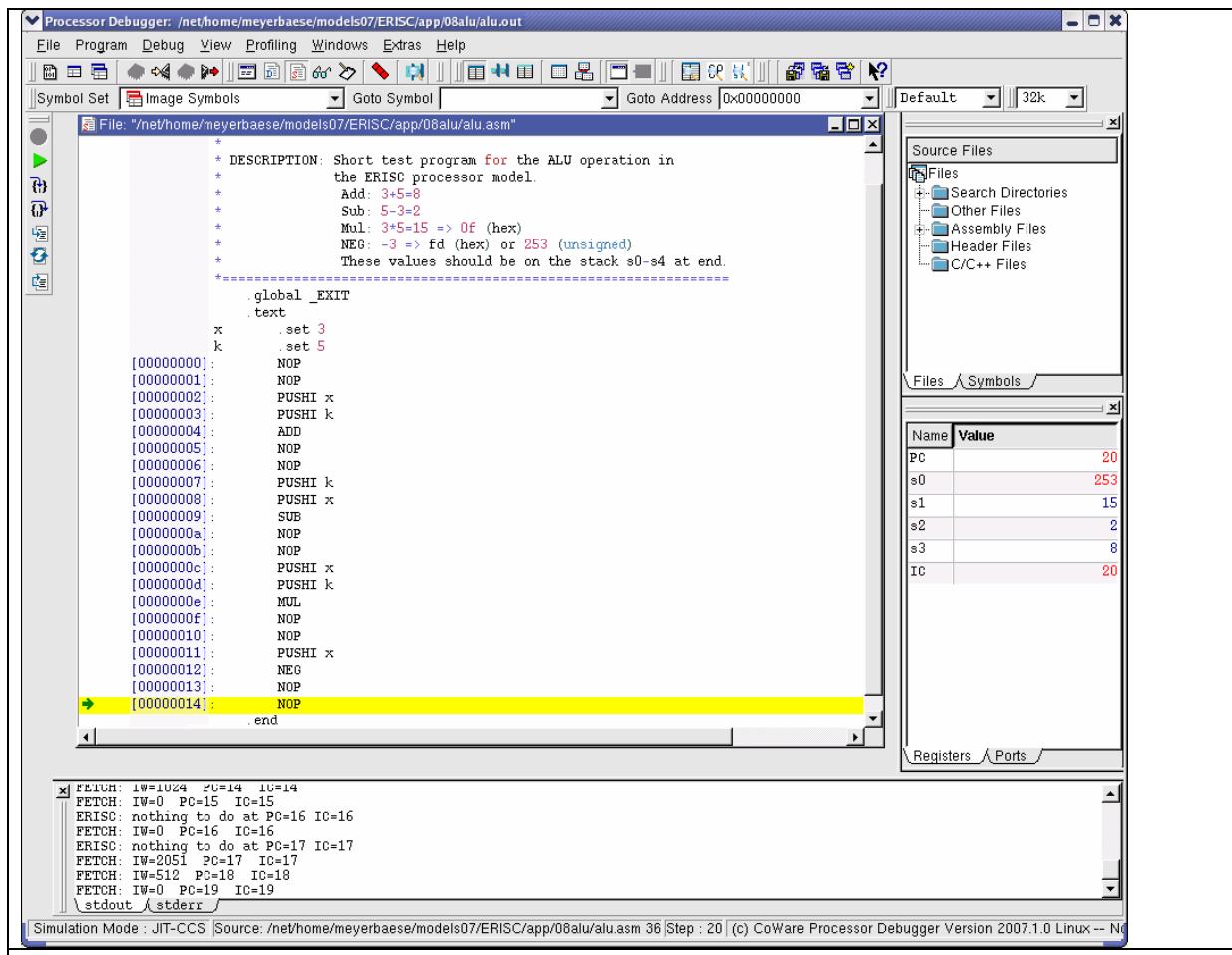
ERISC03xTimingSim.bmp

#12



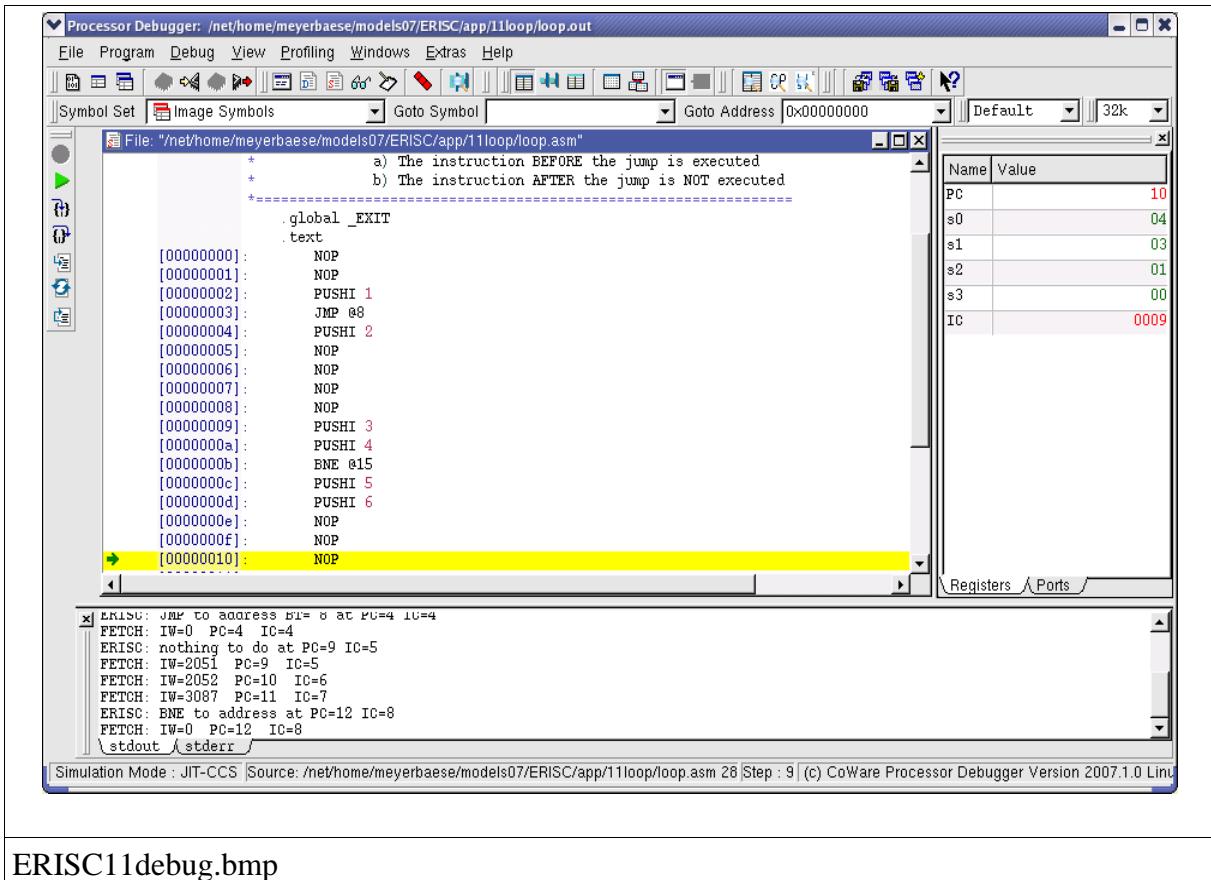
ERISC08funSim.bmp

#13



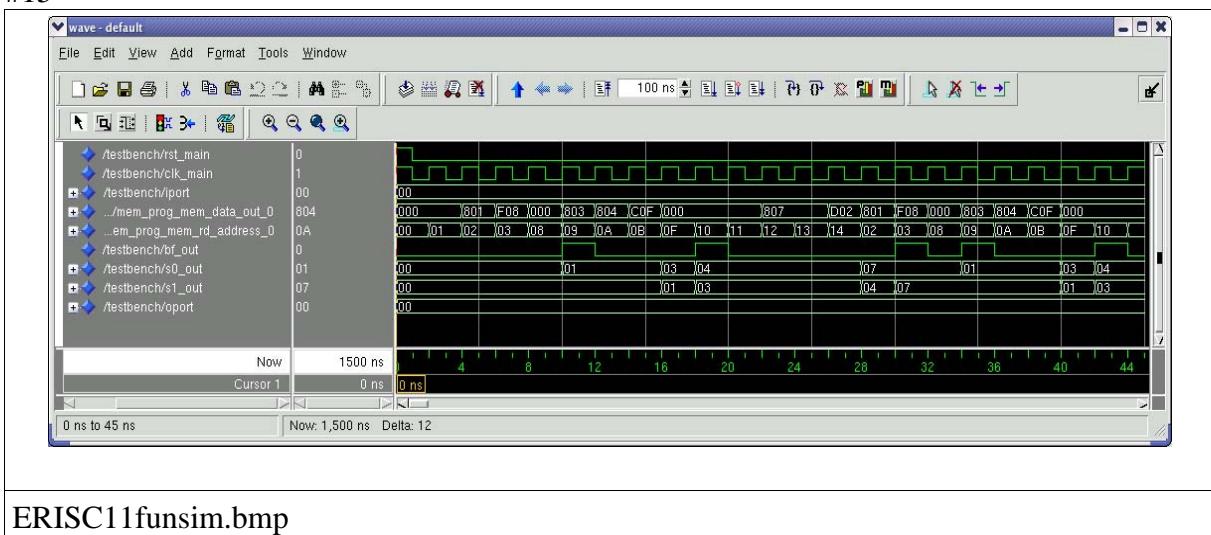
ERISC08debug.bmp

#14



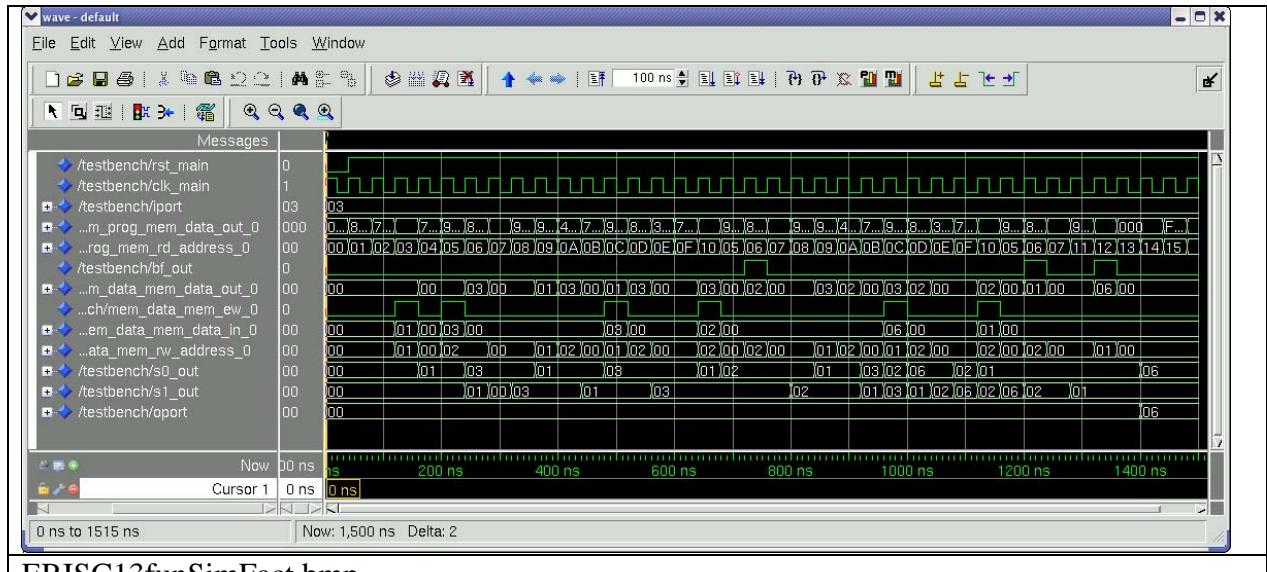
ERISC11debug.bmp

#15



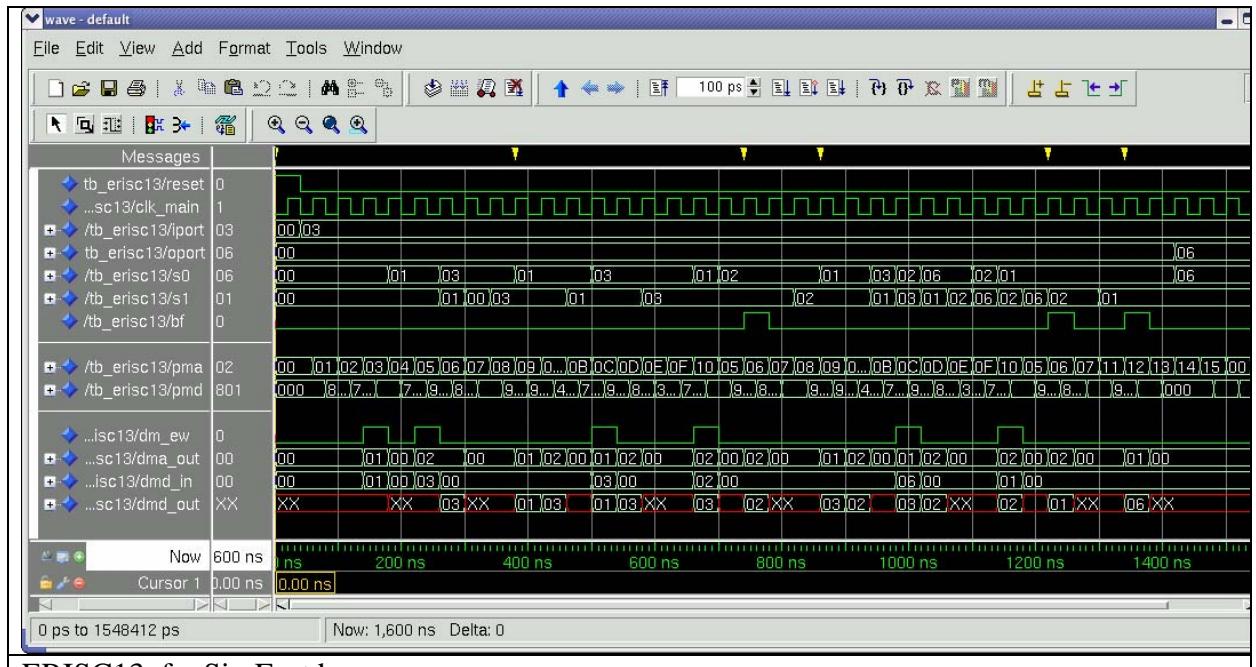
ERISC11funsim.bmp

#16



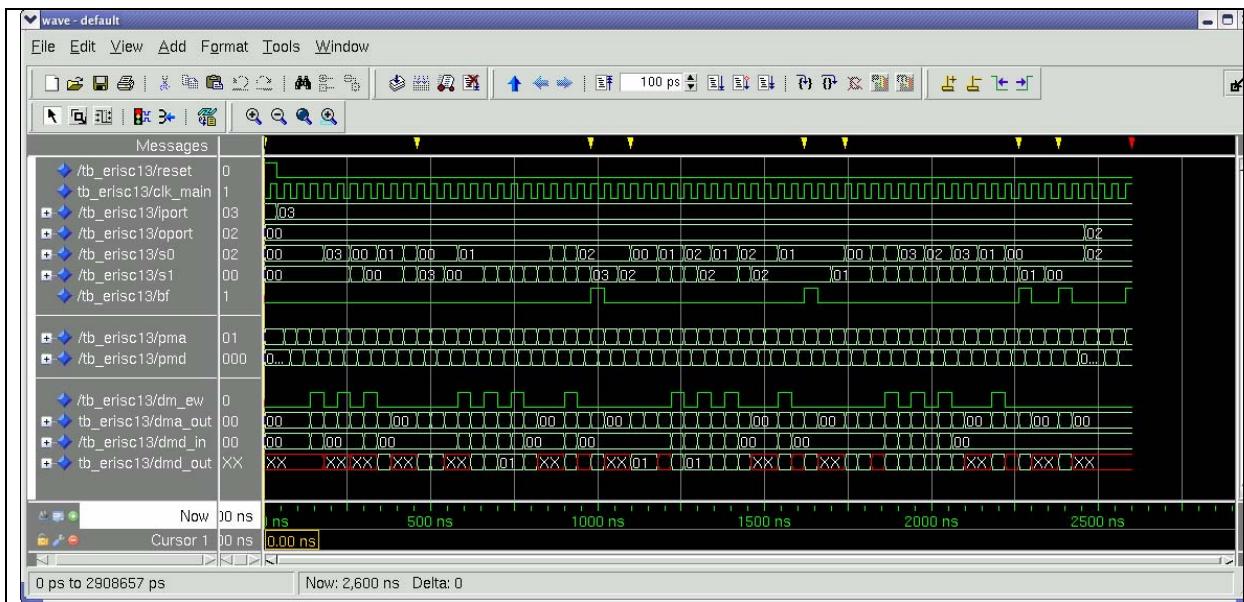
ERISC13funSimFact.bmp

#17



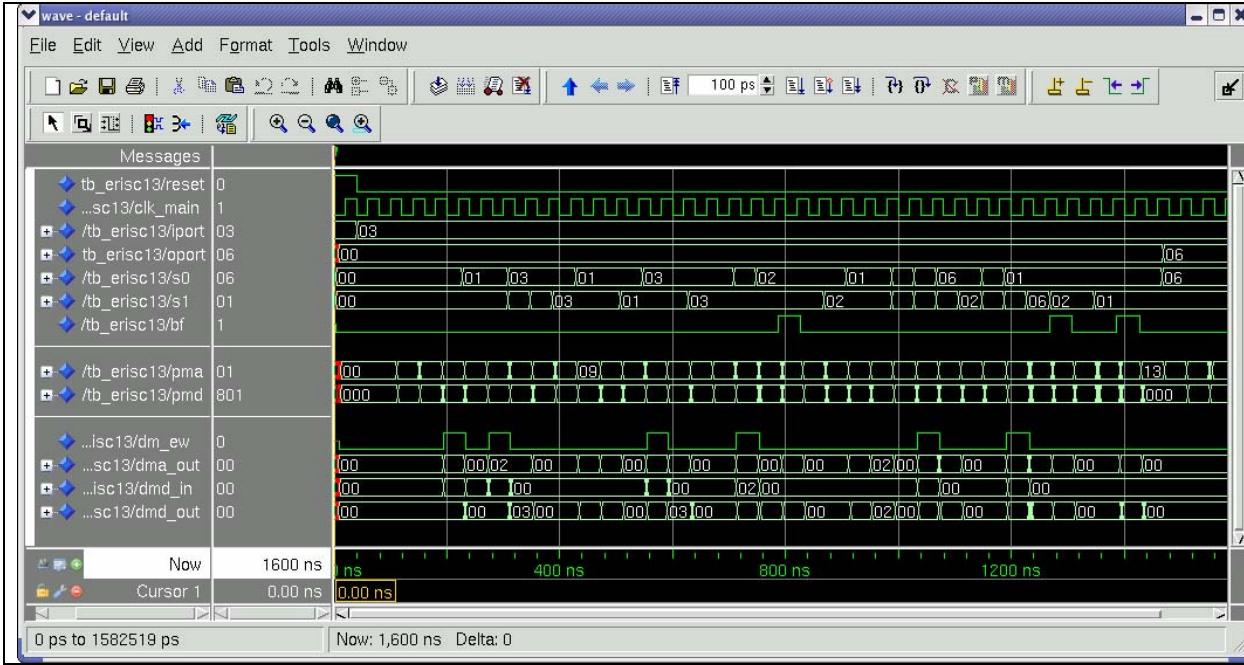
ERISC13xfunSimFact.bmp

#18



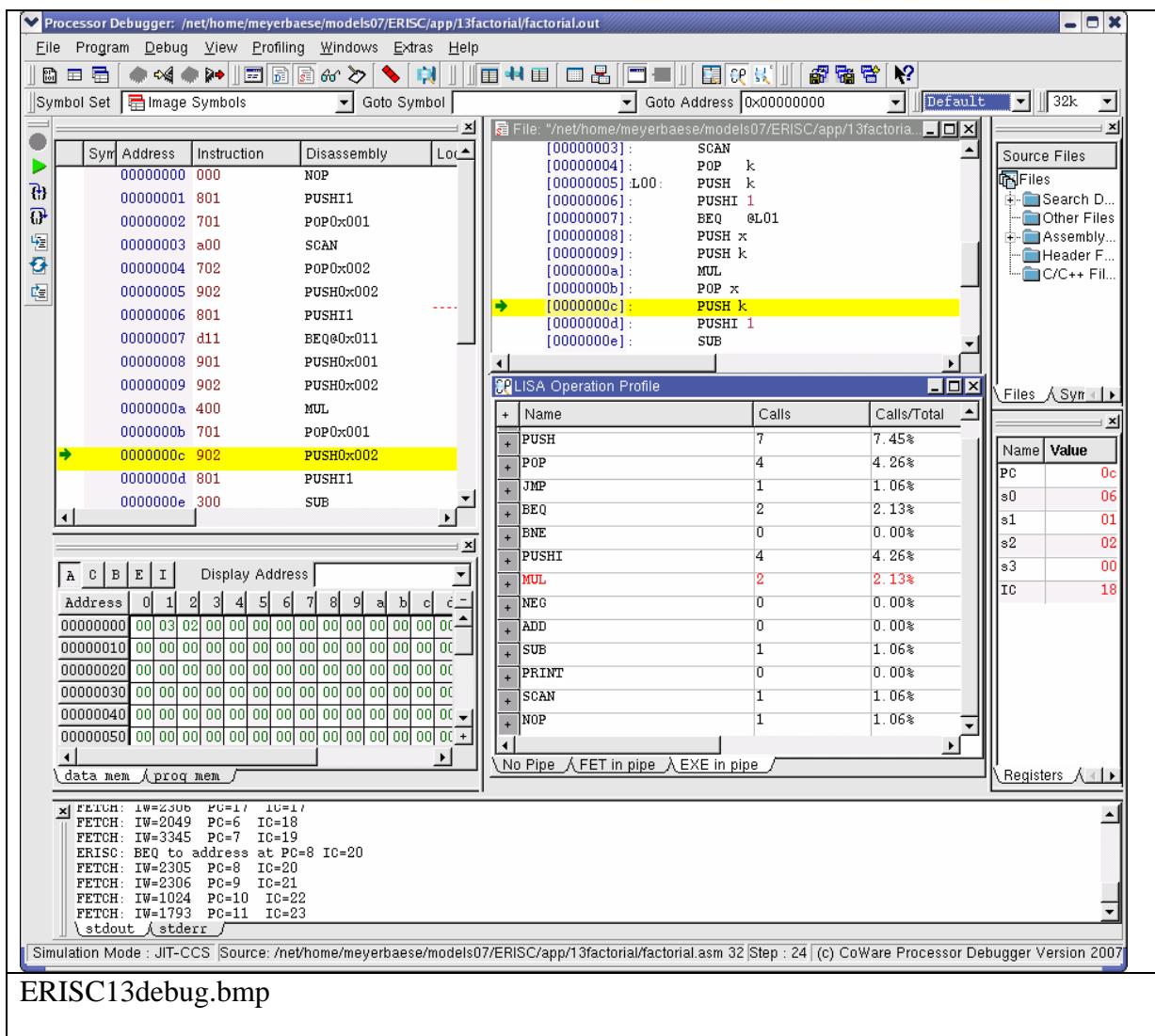
ERISC13xfunSimFibo.bmp

#19

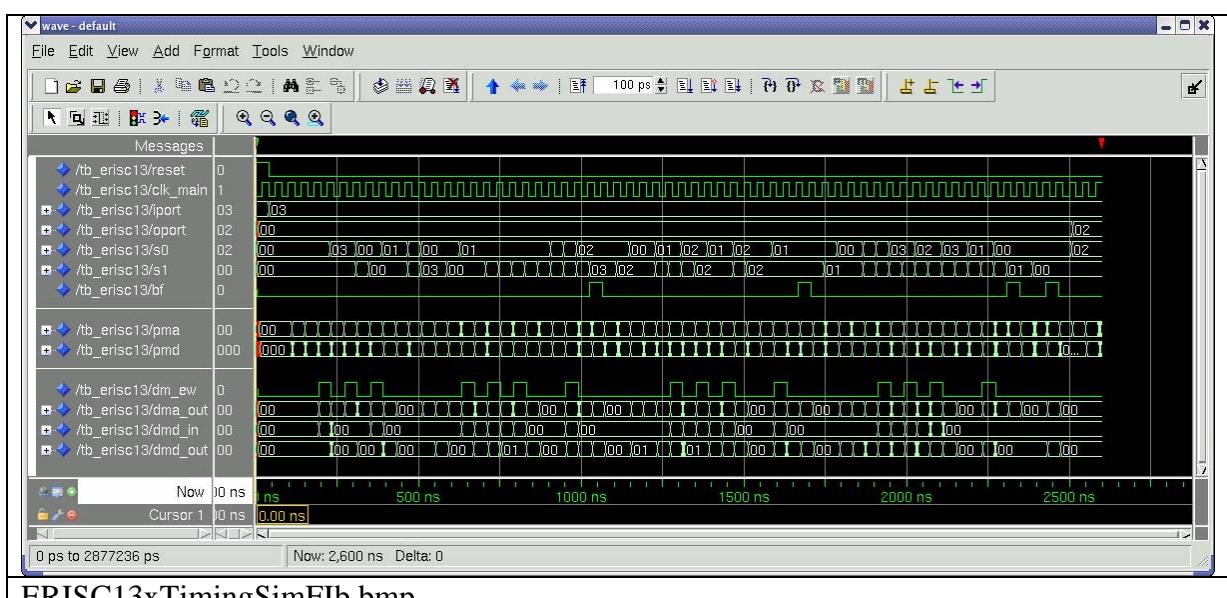


ERISC13xTimingSimFact.bmp

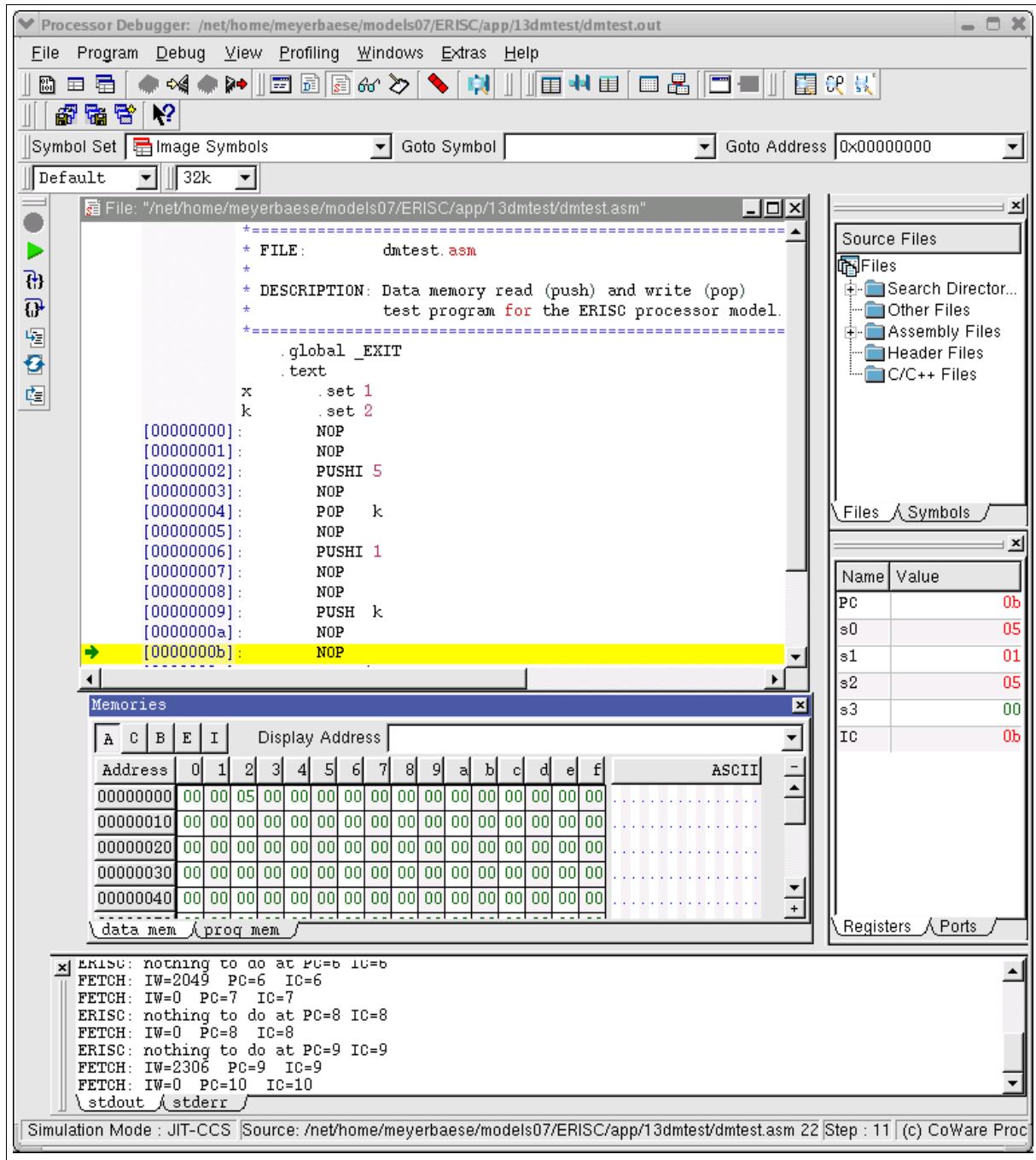
#20



#21

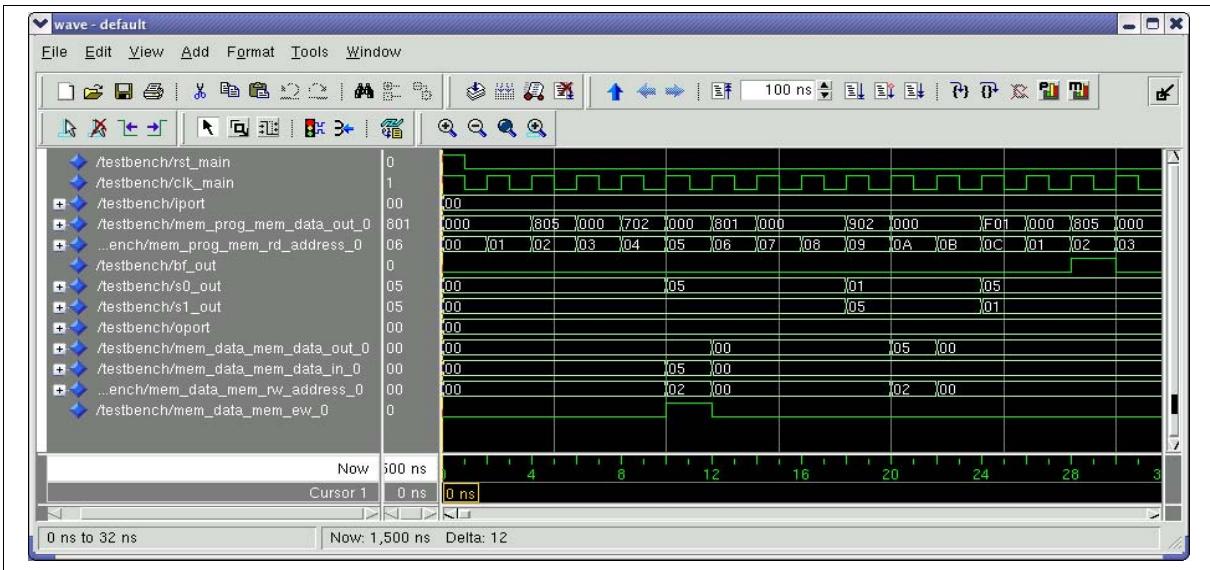


#22



ERISC13debugDMtest.bmp

#23



ERISC13funSimDMtest.bmp

#24

```

14 // * CONDITIONAL_BRANCH (NOT EQUAL: BEQ, BNE
15 // * Jump (JMP) unconditional
16 // Version 13: Add a data memory interface
17 // * PUSH on TOS and POP TOS
18 RESOURCE {
19   MEMORY_MAP (PAGE(0), RANGE(0x00000, 0x007F) -> prog_mem((11..0));
20   PAGE(1), RANGE(0x00000, 0x00FF) -> data_mem((7..0));
21 }
22
23 RAM uint32 prog_mem {
24   BLOCKSIZE(12); // Bitwidth is 12
25   SIZE(128); // Total of 2^7 words
26   FLAGS(R|X); // Read and execute indicates program memory
27   ENDIANESS(BIG); // Use for simulator
28 };
29
30 RAM uint32 data_mem {
31   BLOCKSIZE(8); // Bitwidth is 8
32   SIZE(256); // Total of 2^8 words
33   FLAGS(R|W); // Read and write indicates data memory
34   ENDIANESS(BIG); // Use for simulator
35 };
36
37 PROGRAM_COUNTER TClocked(uint8> PC; // Program counter
REGISTER TClocked:int8> s3,s2,s1,s0; // Register stack; s0 ist top-of-stack (TOS)
PIN IN uint8 iport; // Input PINs of the core
PIN OUT TClocked:uint8> oport, s0_out, s1_out; // Output PINs of the core
41 uint16 IW; // The instruction word
42 unsigned bit[1] BF; // Branch flag without register
43 PIN OUT TClocked<bit[1]> BF_OUT; // Branch flag display
44 uint8 BT; // The branch target
45
46 PIPELINE pipe = (PET, EXE);
47 PIPELINE_REGISTER IN pipe {
48   uint8 PPC;
49   uint16 PIV;

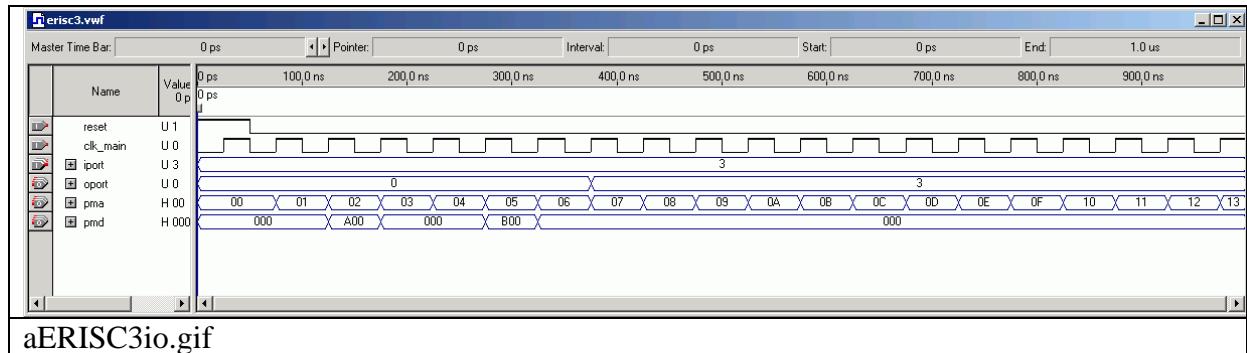
```

ERISC13code.bmp

#25

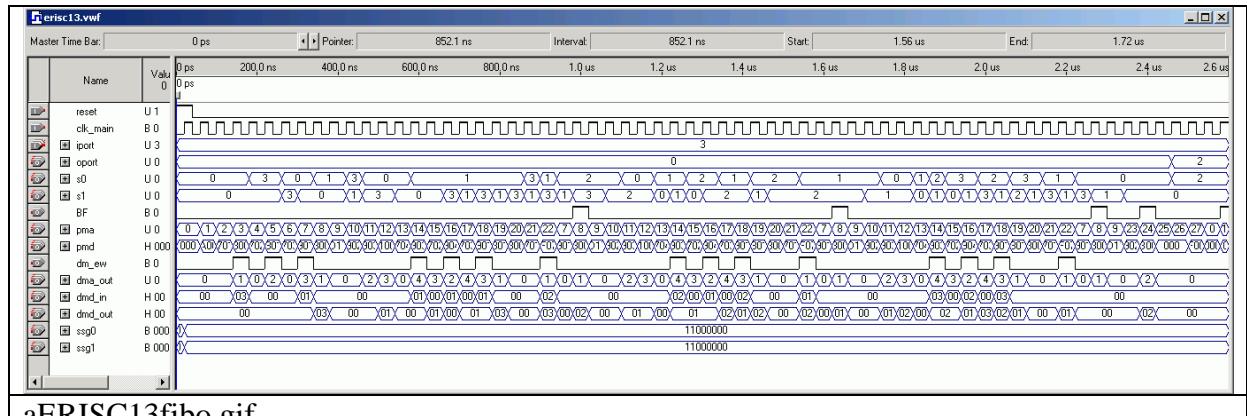
August 2008

Altera Synthesis Results



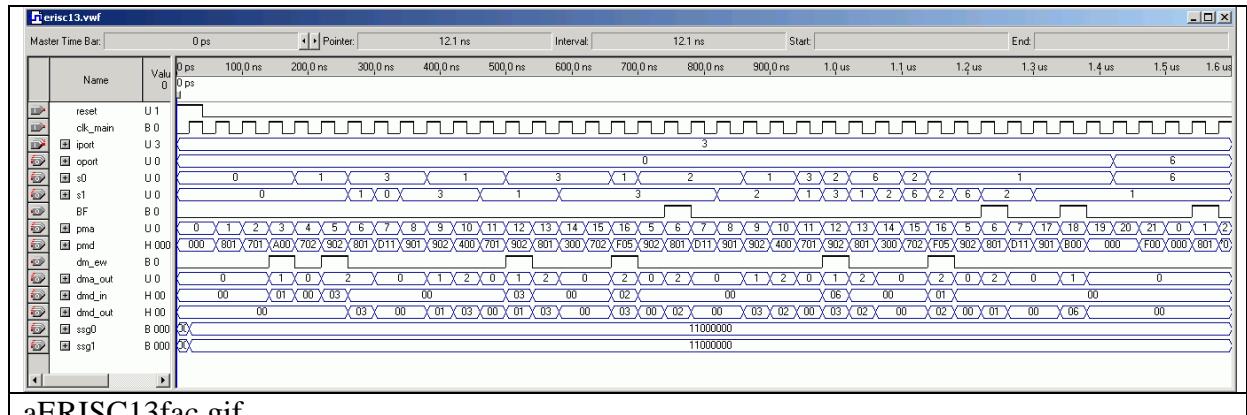
aERISC3io.gif

#26



aERISC13fibo.gif

#27



aERISC13fac.gif

#28