Assembly Instructions to be executed, their Machine code in Instruction Memory and the Simulation and Memory from Questa Sim:

Case:1-GCD

The assembly code is as follows:-

```
li x8, 56
li x9, 84
gcd:
    beq x8, x9, stop
    blt x8, x9, less
    sub x8, x8, x9
    j gcd
less:
    sub x9, x9, x8
    j gcd
stop:
    sw x8, 0x08(x0)
    lw x10, 0x08(x0)
end:
    j end
```

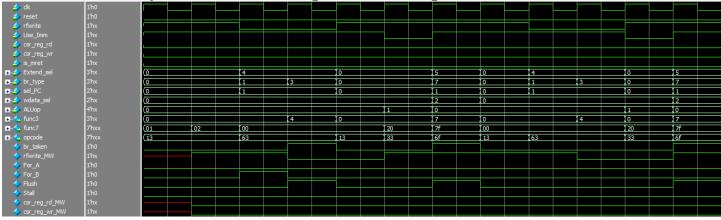
Listing 1. Assembly code to be executed

The generated machine code stored in instruction memory is as follows:-

```
03800413
05400493
00940c63
00944663
40940433
ff5ff06f
408484b3
fedff06f
00802423
00802503
0000006f
```

Listing 2. Instruction memory

The simulation for Top Module for selected signals for above algorithm is as follows:-



Now Cursor 1	0.000215 ms 0.00000 ms		0.00002 ms		0.00004 ms		0.00006 ms		0.00008 ms		0.0001 ms		0.0001
rd_wr	1'hx												
	1'hx												
csr_rdata	32'h00000000	00000000											
epc epc	32'hxxxxxxxxx		333333										
rdata1_MW	32'hxxxxxxxxx	,0000000	(00000000	, 0000000	100000031		00000000	00000054	100000000	00000010	100000038		, 0000000
Forwarded rdata2	32'hxxxxxxxxx	(00000000		100000054	100000054	00000000	00000034	00000000		I 0000001c			0000000
Forwarded rdata1	32'hxxxxxxxx	00000000	(00000000	I 00000038	,00000034	00000000	100000054	00000000	, 00000000	I 00000038	, 0000001c		10000000
rdata2 MW	32'hxxxxxxxxx		00000000	,05400493	100000054	,00944003	100000013	00000038	100000000	7 00000012	10000001c	,00944003	, 4094043
Instruction MW	32'hxxxxxxxxx		03800413	05400493	100000008 100940c63	00944663	00000010	408484b3	fedff06f	100000020	00000008 00940c63	100000000	4094043
PC MW	32'hxxxxxxxx		00000038	I 00000004	100000020	100000018	100000010	100000018	100000008	100000000	100000020	100000018	0000001
Alu result MW	32'hxxxxxxxxx	03600413	00000038	[00940c63 [00000054	00000020	00000013	00000000	0000001c	00000013	100000000	100944663	00000018	0000001
Instruction DE	32hxxxxxxxx	03800413	105400493	100000008	10000000c 100944663	00000010	40848403	fedff06f	00000020	100000008 100940c63	100000000	100000010 140940433	0000001 ff5ff06f
PC_DE	32'hxxxxxxxxx	00000000	10000004	100000000	I 0000000c	00000010	100000018	00000016	100000020	100000000	10000000c	00000010	000000
LSU_wdata	32'hxxxxxxxxx	00000000		00000008	0000000c	00000000	00000054	0000001c	00000000	100000008	1000000c	00000038	0000001
rs1 out	32huuuuuuu 32hxxxxxxxx	0000000		100000000	0000000-	00000000	000000054	0000001-	10000000	100000000	0000000-	00000000	0000001
Alu_result Mem_outData	32'hxxxxxxxx 32'h00000000	00000038	00000054	00000020	00000018	00000000	0000001c	00000008	00000000	100000020	00000018	0000001c	0000000
Imm_out	32'hxxxxxxxxx	00000038	00000054	00000018	0000000c	00000000	00000038	ffffffec	100000000	00000018	(0000000c	0000001c	fffffff4
Extended_Imm	32'hxxxxxxxxx	00000038	00000054	00000018	0000000c	00000000	00000408	ffffffec	00000000	00000018	0000000c	00000409	fffffff4
Instruction	32'h03800413	05400493	00940c63	00944663	40940433	40848463	fedff06f	00802423	00940c63	00944663	40940433	ff5ff06f	408484b
PC	32'h00000000	00000004	00000008	0000000c	00000010	00000018	0000001c	00000020	80000000	0000000c	00000010	00000014	0000001
wdata	32'hxxxxxxxxx		00000038	00000054	00000020	00000018	00000000	0000001c	00000020	00000000	00000020	00000018	0000001
rdata2	32'hxxxxxxxxx	00000000		V	00000054	00000000	00000038	00000000	J	0000001c		V	0000000
rdata1	32'hxxxxxxxxx	00000000		00000038		00000000	000000\$4	00000000		00000038			0000000
JImm	21'hxxxxxX	000038	000054	040808	044808	000000	048408	1fffec	[000000	040808	044808	040c08	1ffff4
UImm	20'hxxxxx	03800	05400	00940	00944	00000	40848	fedff	00000	00940	00944	40940	ff5ff
BImm	13'hxxxX	0028	0848	0018	000c	0000	0c08	17e0	0000	0018	000c	0408	17e0
> SImm	12'hxxx	028	049	018	00c	000	409	fe0	000	018	00c	408	fe0
> Imm	12'hxxx	038	054	009		000	408	fed	000	009		409	ff5
SLImm	5'hxx	18	14	09		00	08	0d	00	09			15
wadd	5'hxx	08	09	18	(0c	(00	09	00		18	Oc	08	00
radd2	5'hxx	18	14	09		(00	08	[Od	00	09			15
radd1	5'hxx	(00		08		00	09	1f	00	08			1f
mask	4'hx												
wdata_sel_MW	2'hx		0						2	10			
is_mret_MW	1'hx												
flush_interrupt	1'h0												
control	1'h0												
intr expc	1'h0												

																_
	1'h1									$ldsymbol{ldsymbol{ldsymbol{eta}}}$						
reset	1'h0															
rfwrite	1h1															
<pre>Use_Imm</pre>	1'h1															
<pre>csr_reg_rd</pre>	1'h0															
<pre>csr_reg_wr</pre>	1'h0															
	1'h0															
I-	3'h0	0	4	(0	1	, o	.5	[0	. 5	0		5		0	5	
-	3'h0	0	1	O			7	Į0	17	10		7		0	7	
⊢ 4 ∕> sel_PC	2'h0	0	1	0			1	[0	1	0		1		0	1	
🛶 wdata_sel	2'h0	0				1	2	[0	2	[0		2		0	2	
I -	4'h0	0														
<mark></mark> - func3	3'h0	0			2		[0									
🛶 func7	7'h00	00														
-4 opcode	7h13	13	63	13	23	03	6f	13	, 6f	13		6f		13	6f	
🔷 br_taken	1'h0															
rfwrite_MW	1'h1															
♦ For_A	1'h0															
♦ For_B	1'h0															
Flush	1'h0															
♦ Stall	1'h0															
csr_reg_rd_MW	1'h0															
csr_reg_wr_MW	1'h0															
intr_expc	1'h0															
control	1'h0															
flush_interrupt	1'h0															
is_mret_MW	1'h0															
├ �� wdata_sel_MW	2'h2	2	(O				1	2	Į o	2		Į0		2	(0	
⊢ ♦ mask	4'hx	 				f										
├ - radd1	5'h00	00	08	00												
⊢ ⇔ radd2	5'h00	00	09	00	08		(00									
🛶 wadd	5'h00	00	18	00	08	,0a	(00									
- → SLImm	5'h00	00	09	00	08		00									
⊨ ♦ Imm	12'h000	000	009	000	008		(000									
SImm	12'h000	000	018	000	008	(00a	000									
⊨ √ BImm	13'h0000	0000	0018	0000	8000	(000a	0000									
⊢ ♦ UImm	20'h00000	00000	00940	00000	00802		00000									
🛶 JImm	21'h000000	000000	040808	000000	002008		(000000									
⊢-∲ rdata1	32'h00000000	00000000	0000001c	00000000												
- √ rdata2	32'h00000000	00000000	0000001c	00000000	(0000001c		00000000									
🛶 wdata	32'h00000018	00000018	00000000	00000020	00000000	00000008	0000001c	(0000002	c (000000	00 000	00002c	0000000	00	0000002c	0000000	0
⊢ ∳ PC	32'h00000008	8000000	0000000c	00000020	00000024	00000028	(0000002c	(0000002			000028	000000		00000028	0000002	
	32'h00940c63	00940c63	00944663	00802423	00802503	0000006f		0000006	f	000	00006f	_		0000006f		
	32'h00000000	00000000	00000018	00000000	80000000	00000008	00000000									
	32'h00000000	00000000	00000018	100000000	100000008	00000008	100000000									

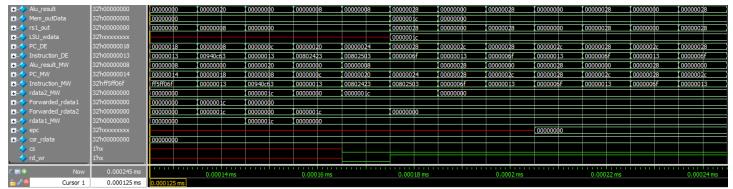


Figure 6. Top Module Simulation

The register memory after simulation is as follows:-

Figure 7. Register memory

The data memory after simulation is as follows:-

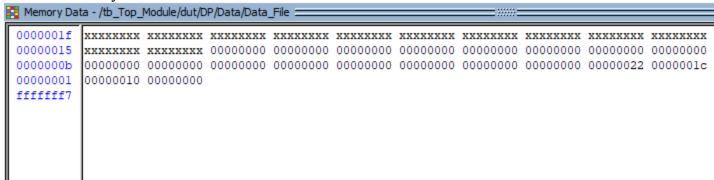


Figure 8. Data memory

The instruction memory after simulation is as follows:-

```
Memory Data - /tb_Top_Module/dut/DP/Inst/Instruction_Files - Default =
000001a2
 0000002e xx xx xx 6f 00 00 00 03 25 80 00 23 24 80 00 6f f0 df fe b3 84 84 40 6f f0 5f ff 33 04 94 40
0000000f 63 46 94 00 63 0c 94 00 93 04 40 05 13 04 80 03
fffffff0
```

Figure 9. Instruction memory

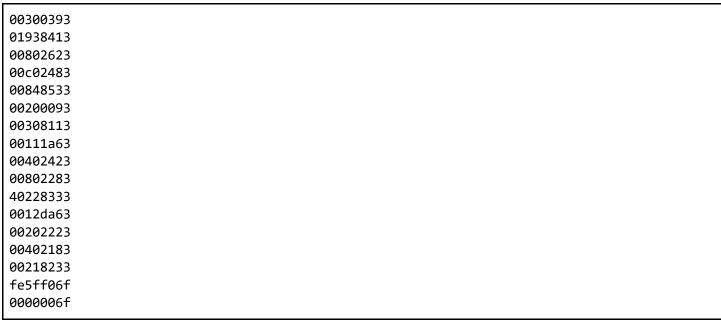
Case: 2- Stalling, Forwarding and Flushing

Here is the assembly code to be tested:-

```
addi x7, x0, 3
    addi x8, x7, 25
    sw x8, 0x0c(x0)
    1w \times 9, 0 \times 0 c(\times 0)
    add x10, x9, x8
main:
    addi x1, x0, 2
    addi x2, x1, 3
    bne x2, x1, jump
back:
    x4, 0x08(x0)
    1w x5, 0x08(x0)
    sub x6, x5, x2
    bge x5, x1, stop
jump:
    x^{2}, 0x^{04}(x^{0})
    1w x3, 0x04(x0)
    add x4, x3, x2
    j back
stop:
    j stop
```

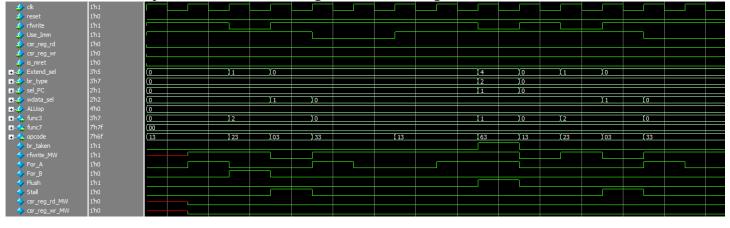
Listing 3. Assembly code to be executed

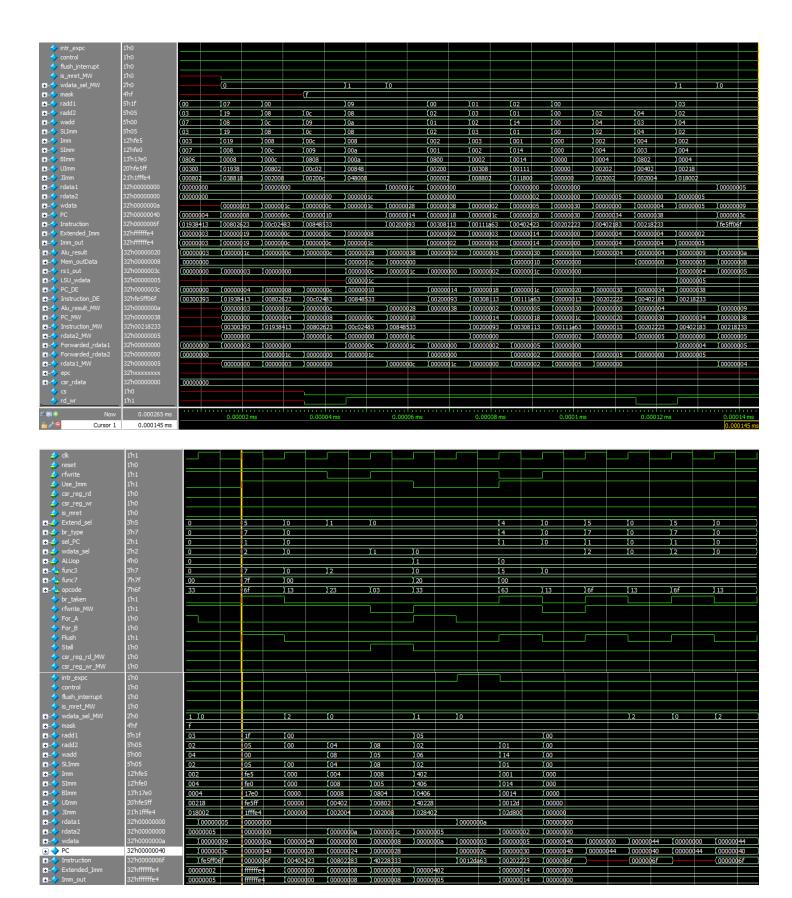
The generated machine code stored in instruction memory is as follows:-



Listing 4. Instruction memory

The simulation for Top Module for selected signals for above algorithm is as follows:-





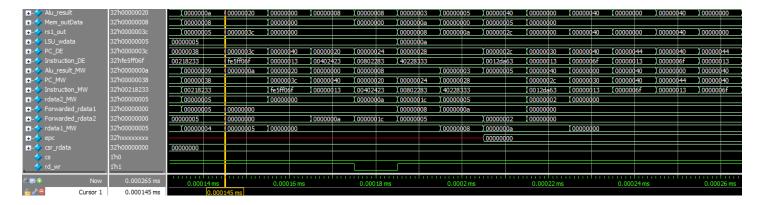


Figure 10. Top Module Simulation

The register memory after simulation is as follows:-

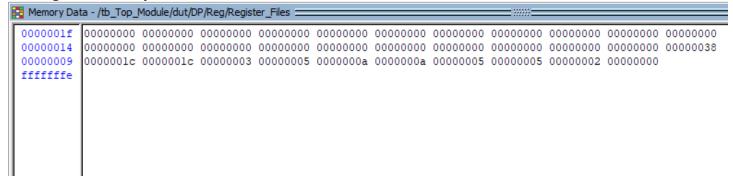


Figure 11. Register memory

The data memory after simulation is as follows:-

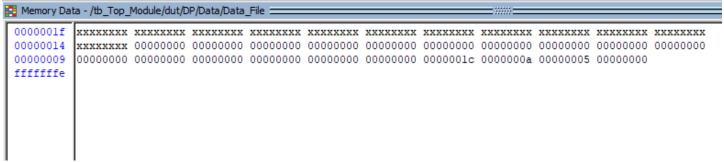


Figure 12. Data memory

The instruction memory after simulation is as follows:-

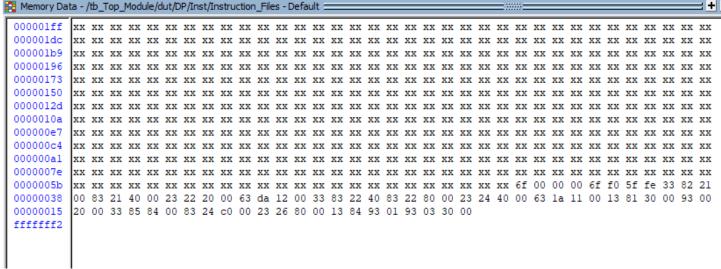


Figure 13. Instruction memory

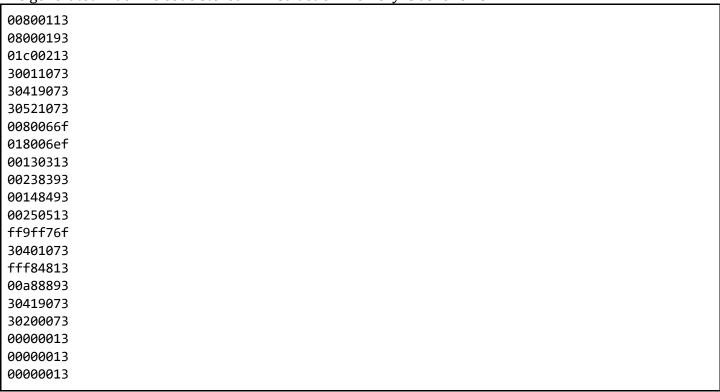
Case:3- CSR Timer Interrupt Handler

Here is the assembly code to be tested:-

```
li x2, 8
li x3, 128
li x4, 28
csrrw x0, mstatus, x2
csrrw x0, mie, x3
csrrw x0, mtvec, x4
jal x12, main
jal x13, handler
main:
    addi x6, x6, 1
    addi x7, x7, 2
stop:
    addi x9, x9, 1
    addi x10, x10, 2
    jal x14, stop
handler:
    csrrw x0, mie, x0
    xori x16, x16, 0xFFFFFFFF
    addi x17, x17, 10
    csrrw x0, mie, x3
    mret
    nop
    nop
```

Listing 5. Assembly code to be executed

The generated machine code stored in instruction memory is as follows:-



Listing 6. Instruction memory

When interrupt occurs first time, register memory (handler executed) is as follows:-

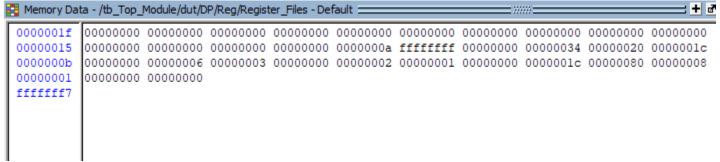


Figure 10. Register Memory

When interrupt occurs second time, register memory (handler executed) is as follows:-

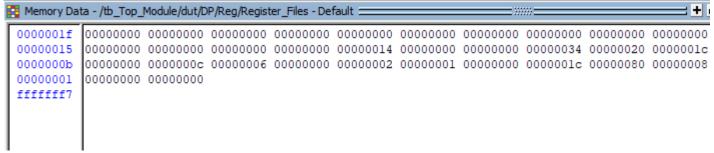
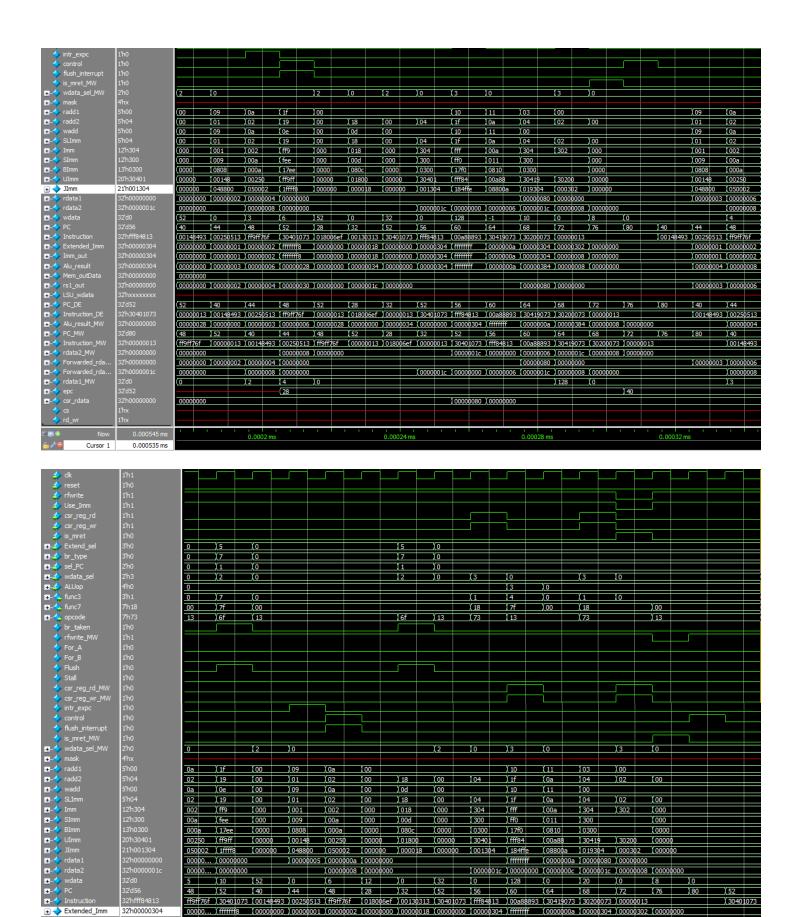


Figure 11. Register memory

The simulation for Top Module for selected signals for above algorithm is as follows:-(0 (0 (0 (O 71 (O (3 0 () o ΪO ίο 17 (04 13 (6f 6f 0 (3 (2 2 (0 χo (1f 02 03 04 (00 (06 (07 09 (0a (00 09 (1f ľ 1c) oo 00 ((00 04 05 08 01 02 01 02 19 (00 01 (02 19 (03 (04 (00 (Oc (00 06 (07 09 (0a 0e (00 (09 (0a (0e (00 (08 (080 (01c 300 304 (008 000 002 001 ff9 (000 002 ff9 1083 004 300 100c 000 1006 1007 009 . 00a) fee (000 009 100a) fee 0882 (0004) 000c (0000 (0806 000a . 17ee (0000 0808 17ee 0300 (0006 0808 (000a 30419 00130 00148 00148 (011300 (019304 (021b04 (000008 (000000 (030800 038002 048800 050002 (00000008 (00000080 (0000001c (00000000 (00000001 (00000002 (00000000 (00000008 (00000000 (00000008 (00000000 (0000001c)(00000000 00000000 16 20 (24) (28) (32 36 (40 (44 (48 (8) 12 152 (40 (00000000 (0000001c (00000000 (00000008 (00000000 (00000008 (00000000 (00000008 (00000080 (0000001c (00000000 Y0000001 Y0000002 Y000000 00000000 X00000008 X00000000 X00000008 X00000000 0000001c 00000000 (00000008 (0000000) 0 (8 128 1 00000000 Cursor 1 0.000535 ms) 5 χo Ϊ5 χo (0 (0 13 χo χo χo Ϊ3 χo 7f (00 18 18 00 13) 6f 13 (6f 13 73 13 73 13



X00000000 X00000001 X00000002 X00000000 X00000018 X00000000 X00000304 Xffffffff

1,0000000a 1,00000304 1,00000008 1,00000000

<u>+</u> → Extended_Imm

32'h00000304

00000... \ fffffff8

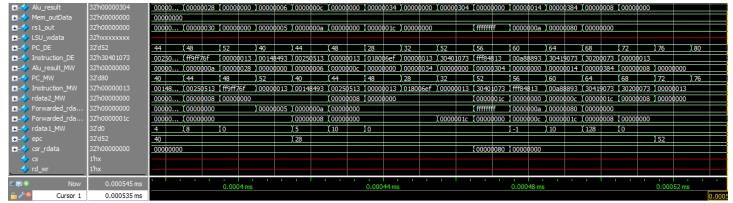


Figure 12. Top Module Simulation

The instruction memory after simulation is as follows:-

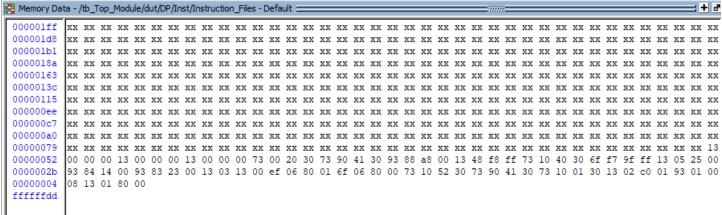


Figure 13. Instruction memory