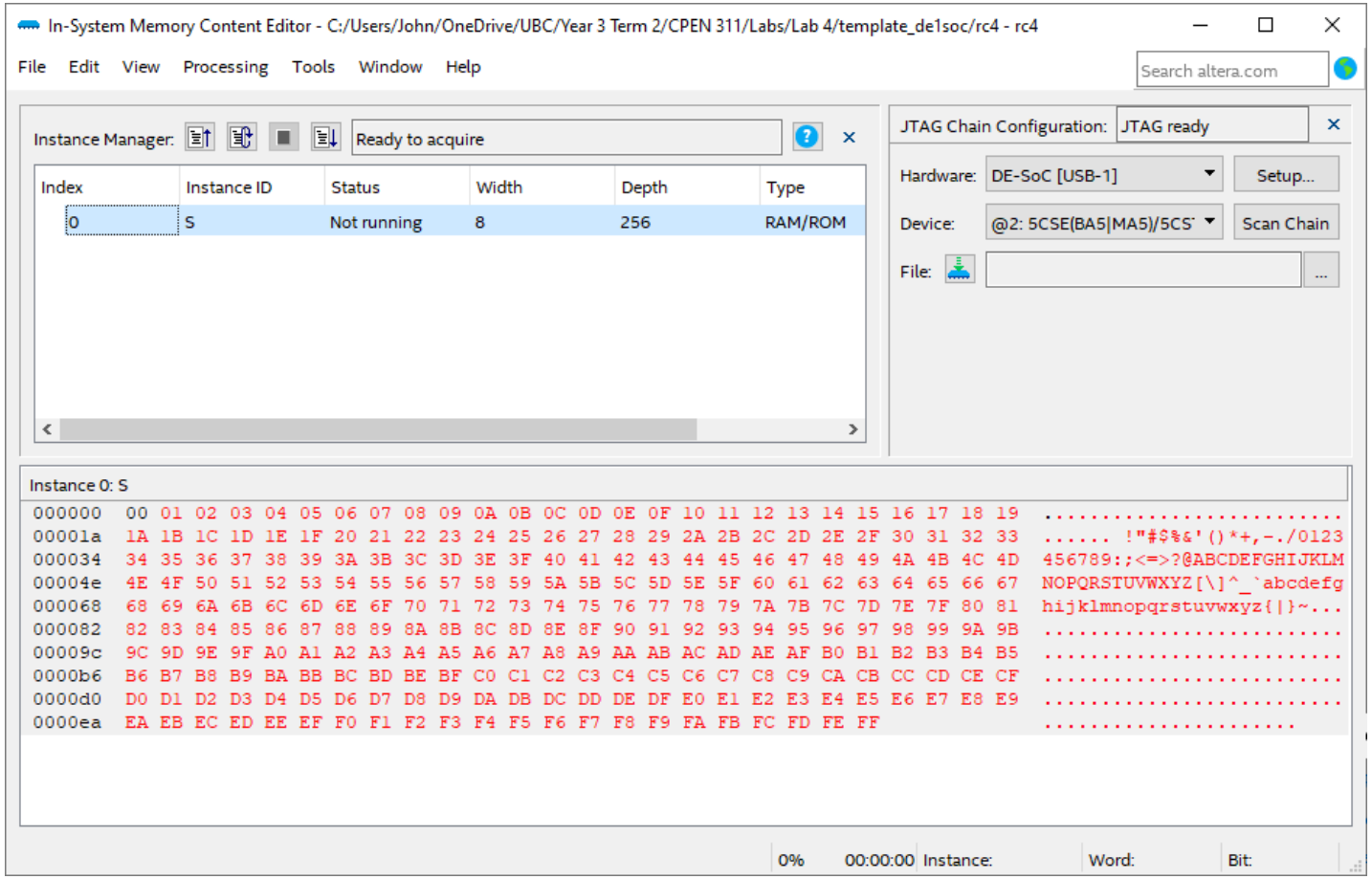
CPEN 311 Lab 4

Zihao Pu

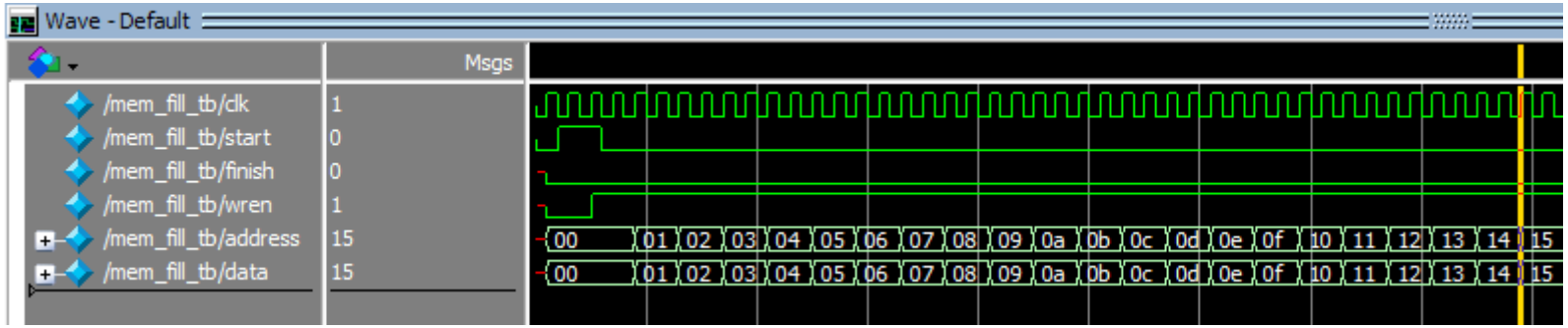
47060645

Task 1: Filling memory

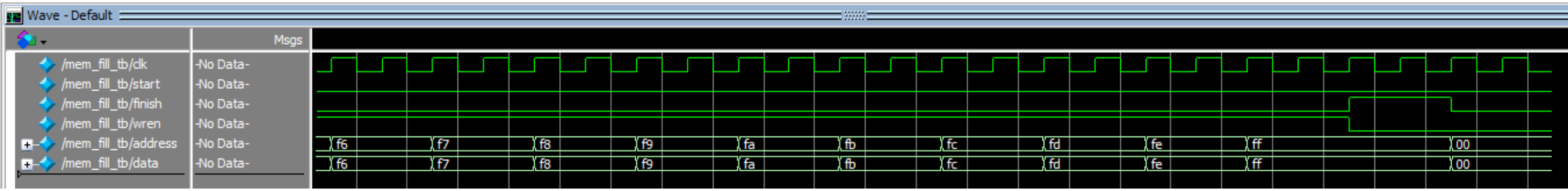


Task 1 Simulation

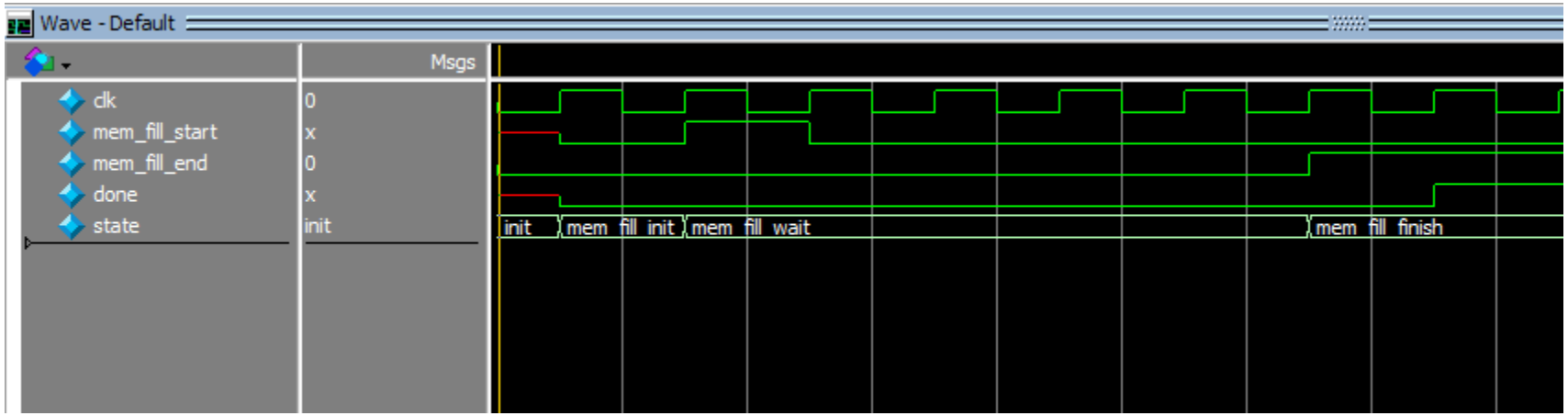
At start:



At end:



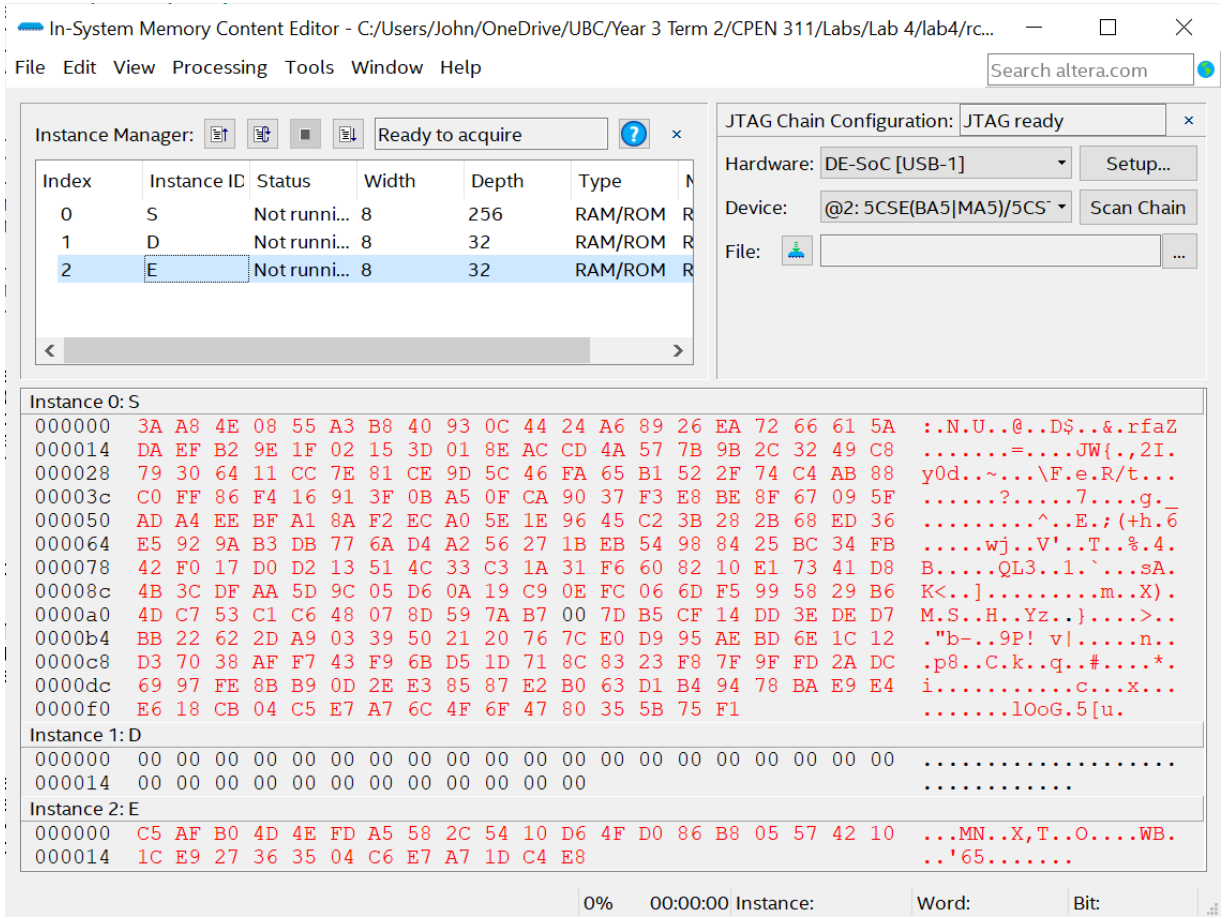
Task 1 – Control FSM



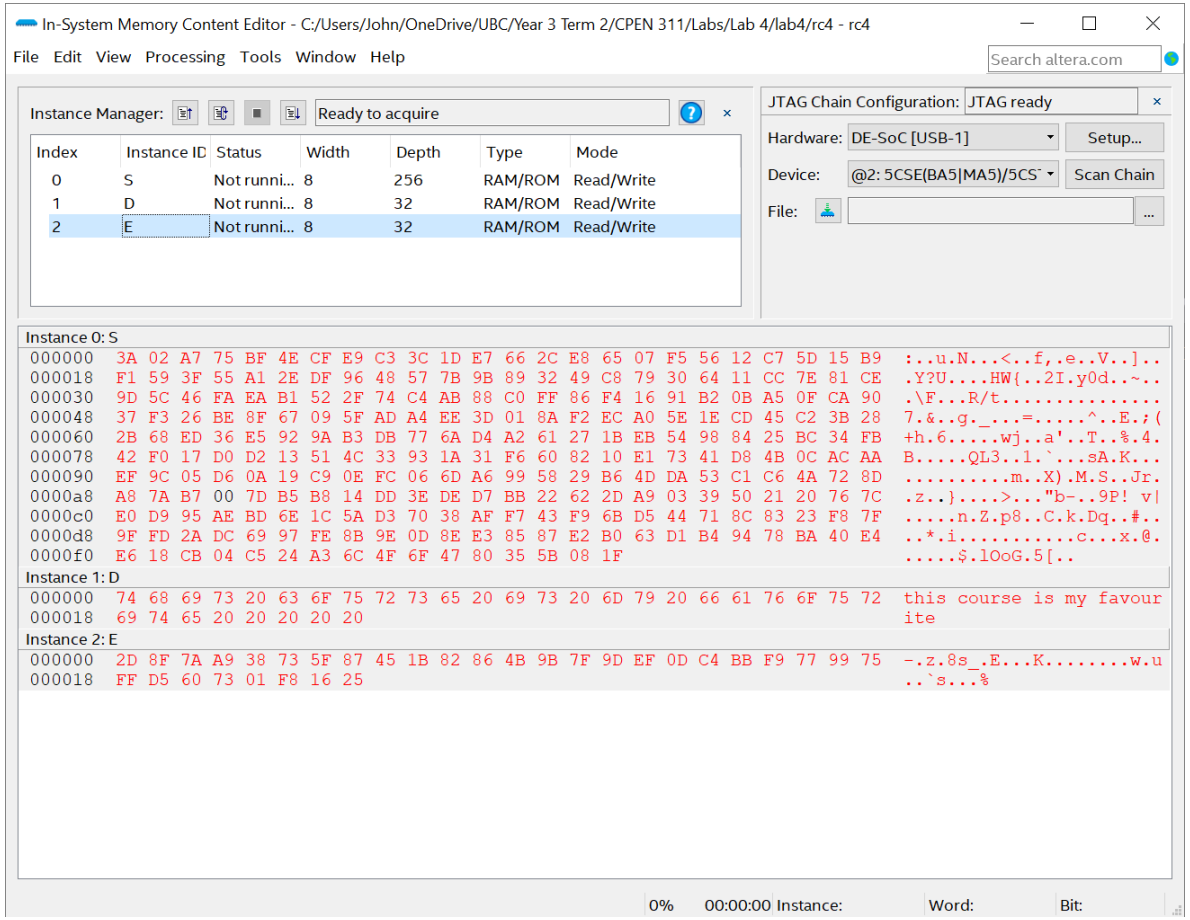
The FSM is done, but the control logic does not work very well. The following tasks are done by 1 massive FSM.

Task 2a

Same as expected

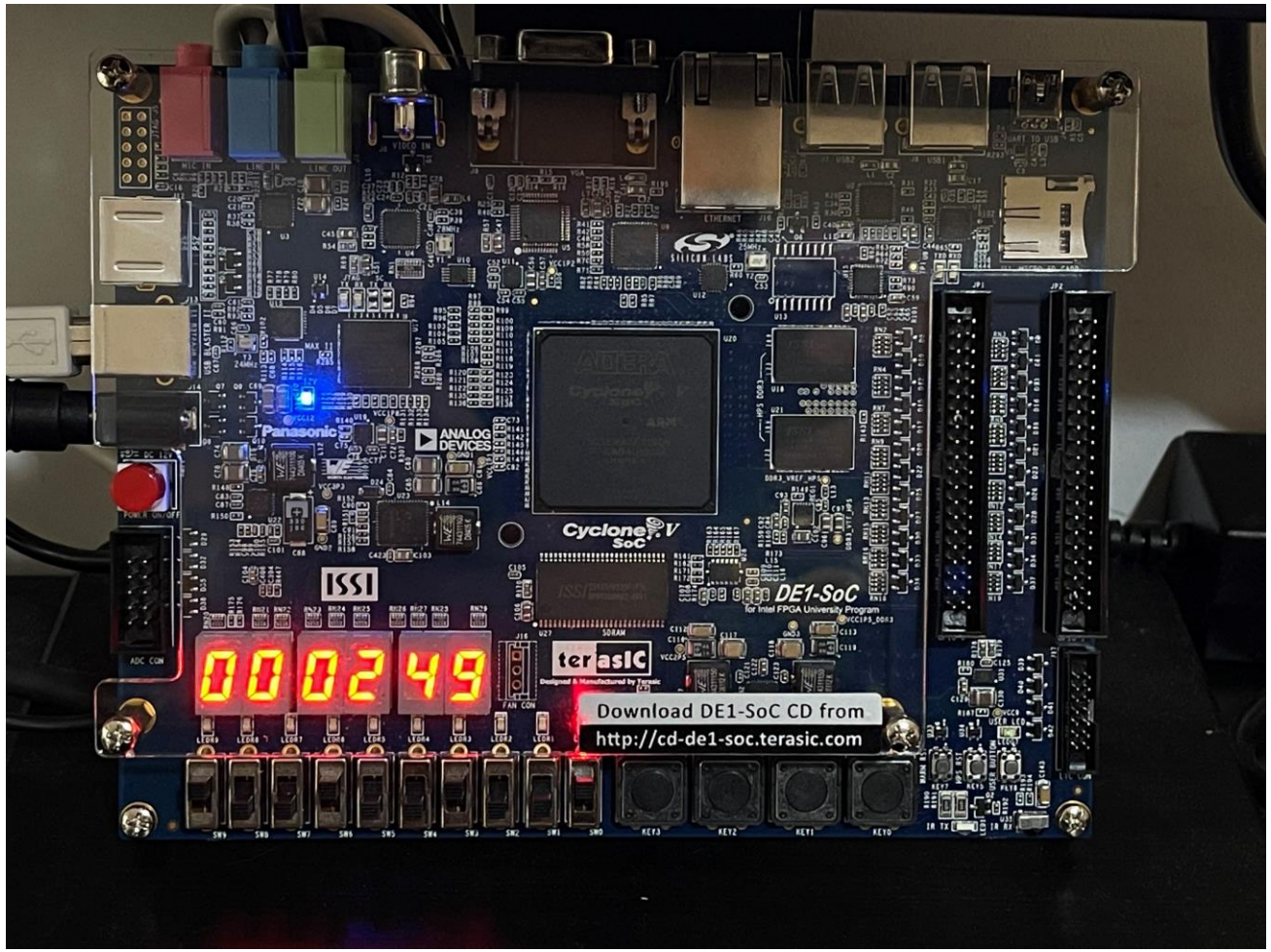


Task 2b


In-System Memory Content Editor - C:/Users/John/OneDrive/UBC/Year 3 Term 2/CPEN 311/Labs/Lab 4/lab4/rc4 - rc4
File Edit View Processing Tools Window Help
Search altera.com
Instance Manager:
Ready to acquire
Index Instance ID Status Width Depth Type Mode
0 S Not runni... 8 256 RAM/ROM Read/Write
1 D Not runni... 8 32 RAM/ROM Read/Write
2 E Not runni... 8 32 RAM/ROM Read/Write
JTAG Chain Configuration: JTAG ready
Hardware: DE-SoC [USB-1]
Setup...
Device: @2: 5CSE(BA5|MA5)/5CS
Scan Chain
File:
Instance 0: S
000000 3A 02 A7 75 BF 4E CF E9 C3 3C 1D E7 66 2C E8 65 07 F5 56 12 C7 5D 15 B9 :..u.N...<..f,.e..V..]..
000018 F1 59 3F 55 A1 2E DF 96 48 57 7B 9B 89 32 49 C8 79 30 64 11 CC 7E 81 CE .Y?U....HW{..2I.y0d..~..
000030 9D 5C 46 FA EA B1 52 2F 74 C4 AB 88 C0 FF 86 F4 16 91 B2 0B A5 0F CA 90 .\F...R/t.....
000048 37 F3 26 BE 8F 67 09 5F AD A4 EE 3D 01 8A F2 EC A0 5E 1E CD 45 C2 3B 28 7.&..g._...=.....^..E.;(
000060 2B 68 ED 36 E5 92 9A B3 DB 77 6A D4 A2 61 27 1B EB 54 98 84 25 BC 34 FB +h.6.....wj..a'..T..%.4.
000078 42 F0 17 D0 D2 13 51 4C 33 93 1A 31 F6 60 82 10 E1 73 41 D8 4B 0C AC AA B.....QL3..1.`...sA.K...
000090 EF 9C 05 D6 0A 19 C9 0E FC 06 6D A6 99 58 29 B6 4D DA 53 C1 C6 4A 72 8D .....m..X).M.S..Jr.
0000a8 A8 7A B7 00 7D B5 B8 14 DD 3E DE D7 BB 22 62 2D A9 03 39 50 21 20 76 7C .z..}....>..."b-..9P! v|
0000c0 E0 D9 95 AE BD 6E 1C 5A D3 70 38 AF F7 43 F9 6B D5 44 71 8C 83 23 F8 7F .....n.Z.p8..C.k.Dq..#..
0000d8 9F FD 2A DC 69 97 FE 8B 9E 0D 8E E3 85 87 E2 B0 63 D1 B4 94 78 BA 40 E4 ..*.i.....c...x.@.
0000f0 E6 18 CB 04 C5 24 A3 6C 4F 6F 47 80 35 5B 08 1F .....$.lOoG.5[..
Instance 1: D
000000 74 68 69 73 20 63 6F 75 72 73 65 20 69 73 20 6D 79 20 66 61 76 6F 75 72 this course is my favour
000018 69 74 65 20 20 20 20 20 ite
Instance 2: E
000000 2D 8F 7A A9 38 73 5F 87 45 1B 82 86 4B 9B 7F 9D EF 0D C4 BB F9 77 99 75 -.z.8s_.E...K.....w.u
000018 FF D5 60 73 01 F8 16 25 ..`s...%
0% 00:00:00 Instance: Word: Bit:

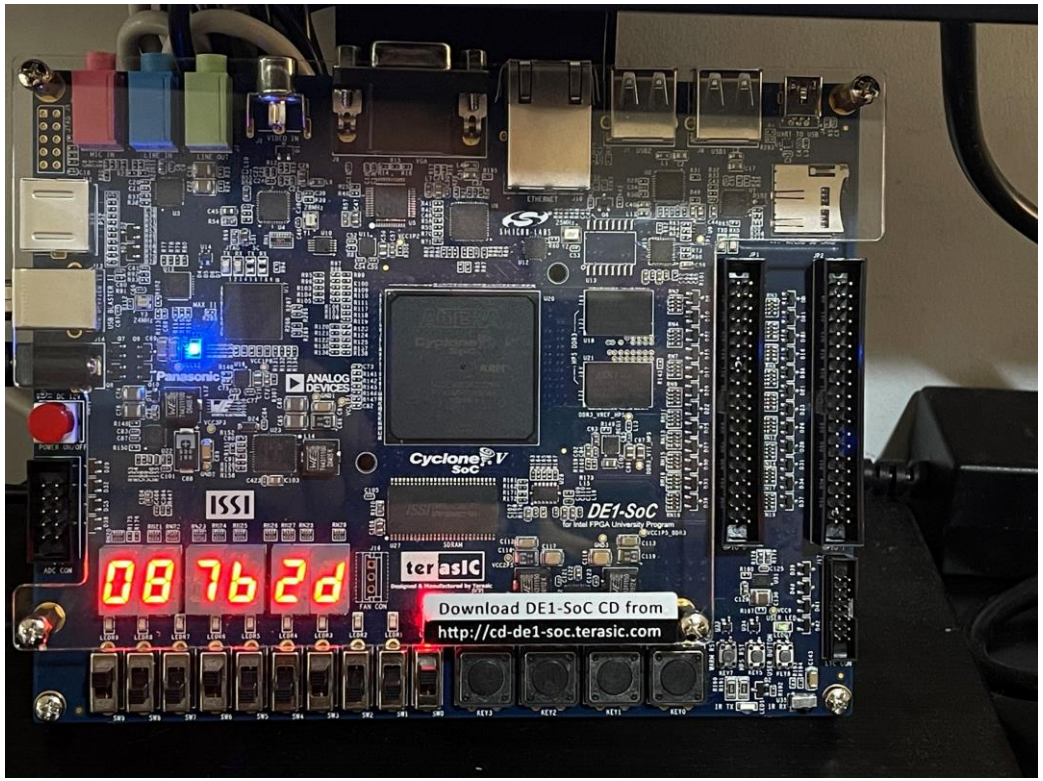
Task 3

A test on message used for task 2a. The cracked code is exactly same as known one.

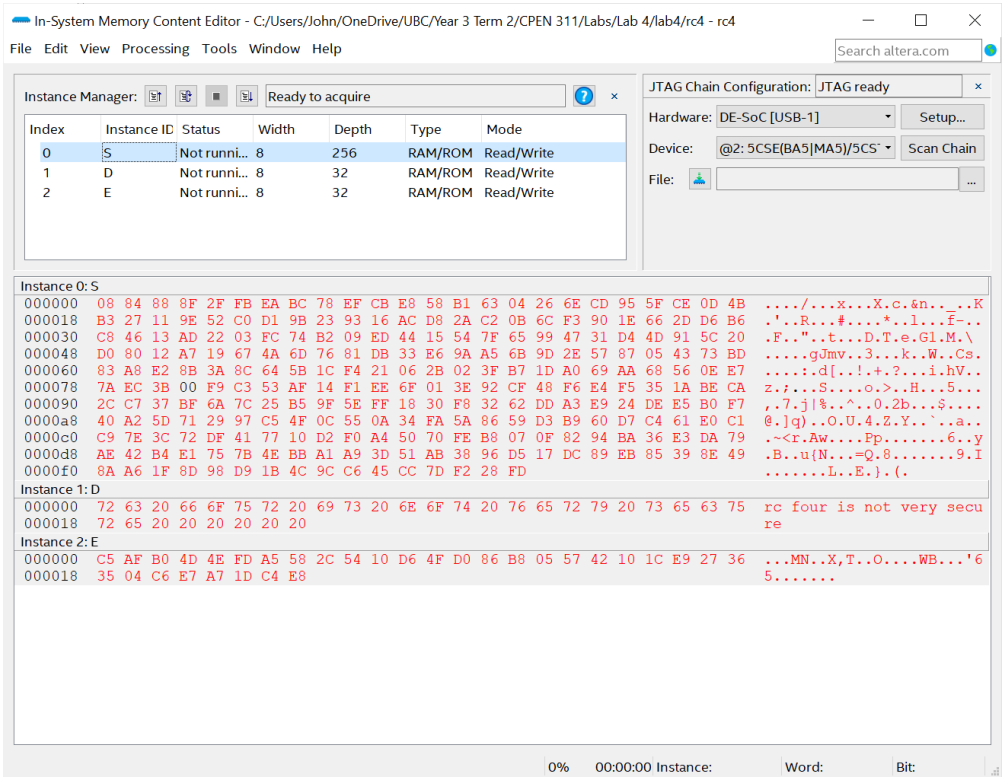


Task 3

Secret Key



Result



Simulation

Full FSM Simulation

Simulation: Construct of Memories

Since the M10K memory is not accessible from ModelSim, hand-written memories are implemented as system memories.

This code here shows the code of s_memory

```
1.  module s_memory(address, clock, data, wren, q);
2.      input wire clock;
3.      input logic [7:0] address;
4.      input logic [7:0] data;
5.      input wire wren;
6.      output logic [7:0] q;
7.
        reg [7:0] mem [256];
8.
        always_ff @( posedge clock ) begin : smem
9.          if(wren) mem[address] <= data;
10.
11.         q <= mem[address];
12.     end
13. endmodule
```

Simulation: Construct of Memories

Since the M10K memory is not accessible from ModelSim, hand-written memories are implemented as system memories.

This code here shows the code of d_memory

```
1.  module d_memory(address, clock, data, wren, q);
2.
    input wire clock;
3.  input logic [4:0] address;
4.  input logic [7:0] data;
5.  input wire wren;
6.  output logic [7:0] q;
7.
    reg [7:0] mem [32];
8.
    always_ff @( posedge clock ) begin : dmem
9.      if(wren) mem[address] <= data;
10.
        q <= mem[address];
11. end
12. endmodule
```

Simulation: Construct of Memories

Since the M10K memory is not accessible from ModelSim, hand-written memories are implemented as system memories.

This code here shows the code of ROM for message

The message is hardcoded from MIF file.

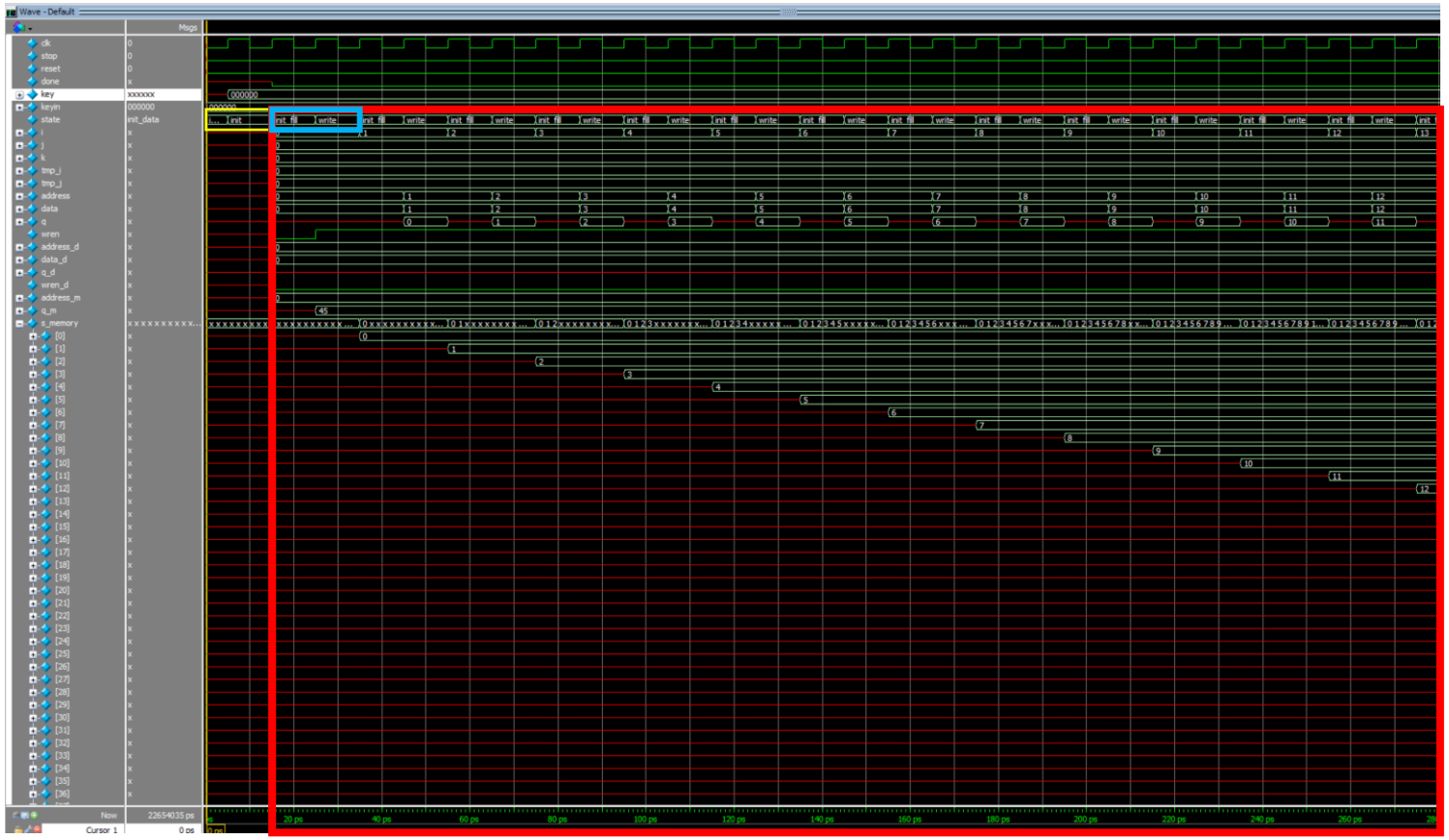
```
module message(address, clock, q);

input wire clock;
input logic [4:0] address;

output logic [7:0] q;

//Message 1
always_ff @( posedge clock ) begin : message
    case(address)
     0 : q<=45;
     1 : q<=143;
     2 : q<=122;
     3 : q<=169;
     4 : q<=56;
     5 : q<=115;
     6 : q<=95;
     7 : q<=135;
     8 : q<=69;
     9 : q<=27;
     10 : q<=130;
     11 : q<=134;
     12 : q<=75;
     13 : q<=155;
     14 : q<=127;
     15 : q<=157;
     16 : q<=239;
     17 : q<=13;
     18 : q<=196;
     19 : q<=187;
     20 : q<=249;
     21 : q<=119;
     22 : q<=153;
     23 : q<=117;
     24 : q<=255;
     25 : q<=213;
     26 : q<=96;
     27 : q<=115;
     28 : q<=1;
     29 : q<=248;
     30 : q<=22;
     31 : q<=37;
     default: q<=8'bx;
    endcase
end
endmodule
```

Simulation: The testbench

Since the IO of FSM is super easy, a simple implementation of testbench is written.

```
1.      module fsm_tb;
2.
            logic clk, stop, reset, done;
3.          logic [23:0] key, keyin;
4.
            fsm DUT(
5.              .clk(clk),
6.              .key(key),
7.              .keyin(keyin),
8.              .stop(stop),
9.              .reset(reset),
10.             .done(done)
11.         );
12.
            always #5 clk = ~clk;
13.
            always @(*) begin
14.             if(done) begin
15.                 #100 $stop;
16.             end
17.         end
18.
            initial begin
19.             #0 clk = 0; reset = 0; stop = 0; keyin <= 0;
20.         end
21.     endmodule
```

Simulation: Step1

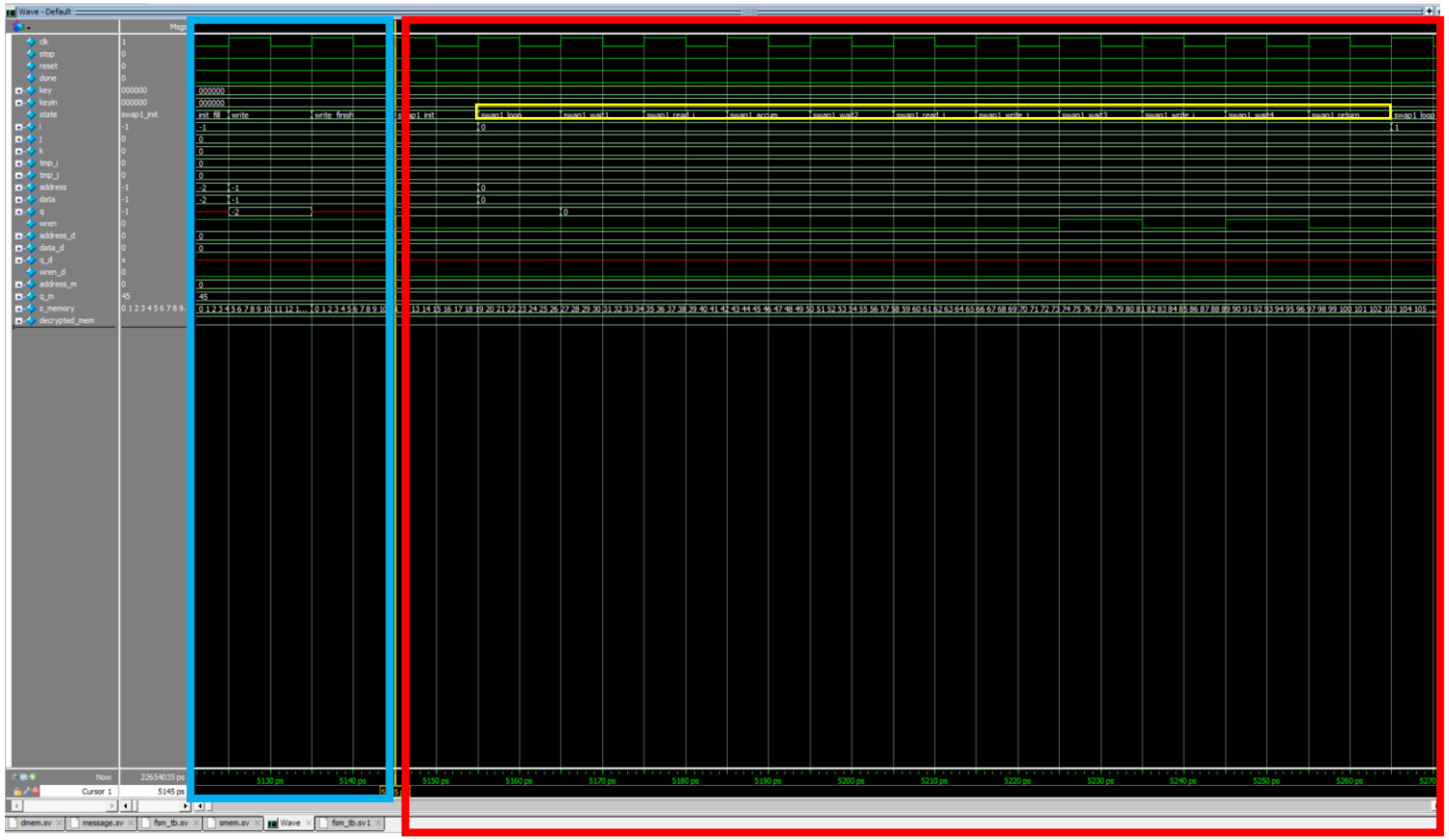


At step 1, the FSM will fill s_memory from 0 to 255.

```
for i = 0 to 255{
    s[i] = i;
}
```

- Yellow box: initialization, to initialize variables
- Red box: filling mechanism
- Blue box: one filling cycle, 2 states
  - -init_fill
  - -write

Simulation: Step1&2

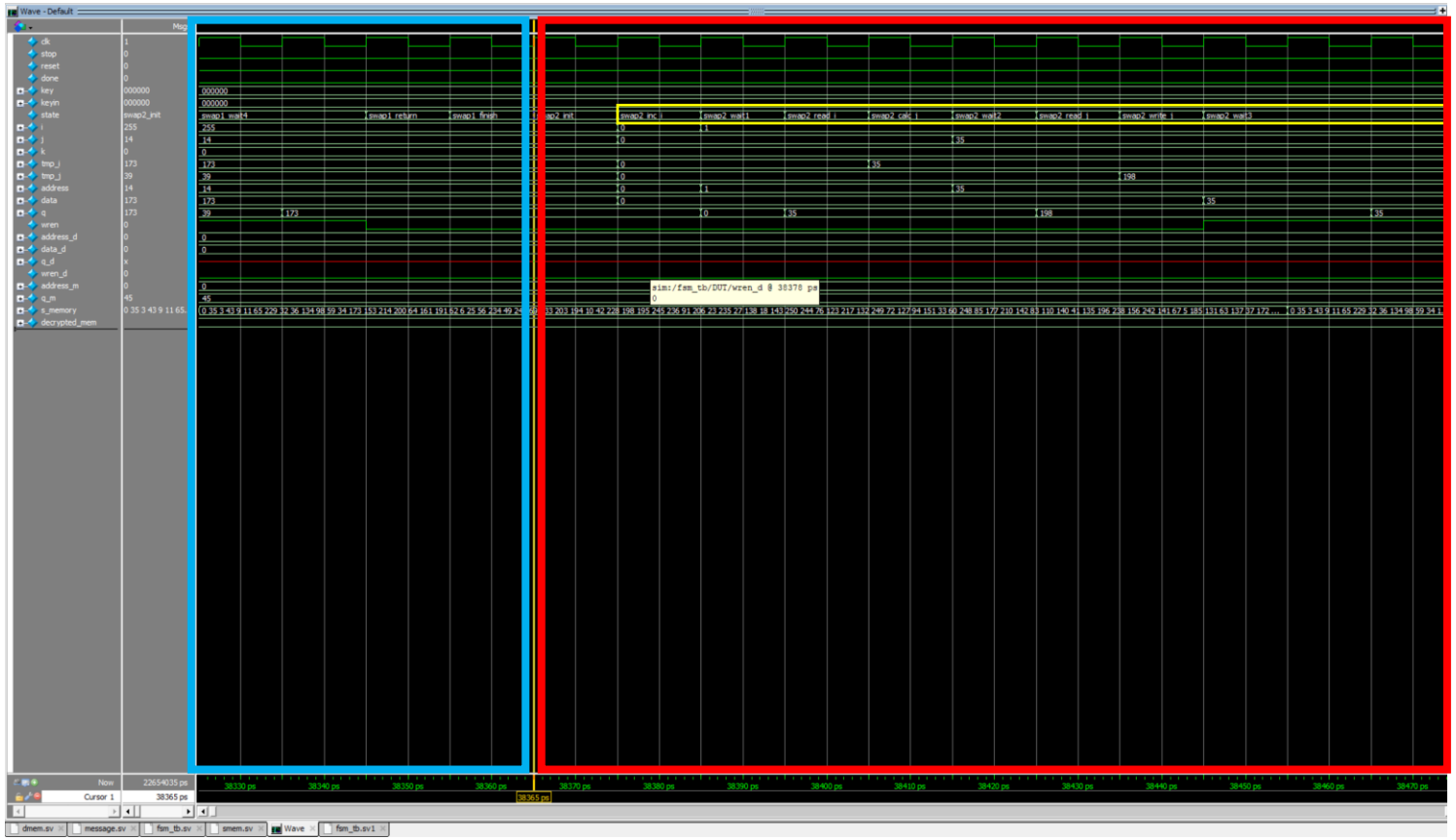


- At step 2, the FSM will swap s_memory from 0 to 255.

```
j = 0
for i = 0 to 255{
    j = (j +s[i]+secretkey[i%3]
    swap s[i] and s[j]
}
```

- Blue box: Filling ends
- Red box: Step 2 FSM, start swaping
- Yellow box: one filling cycle, 11 states

Simulation: Step2a&2b

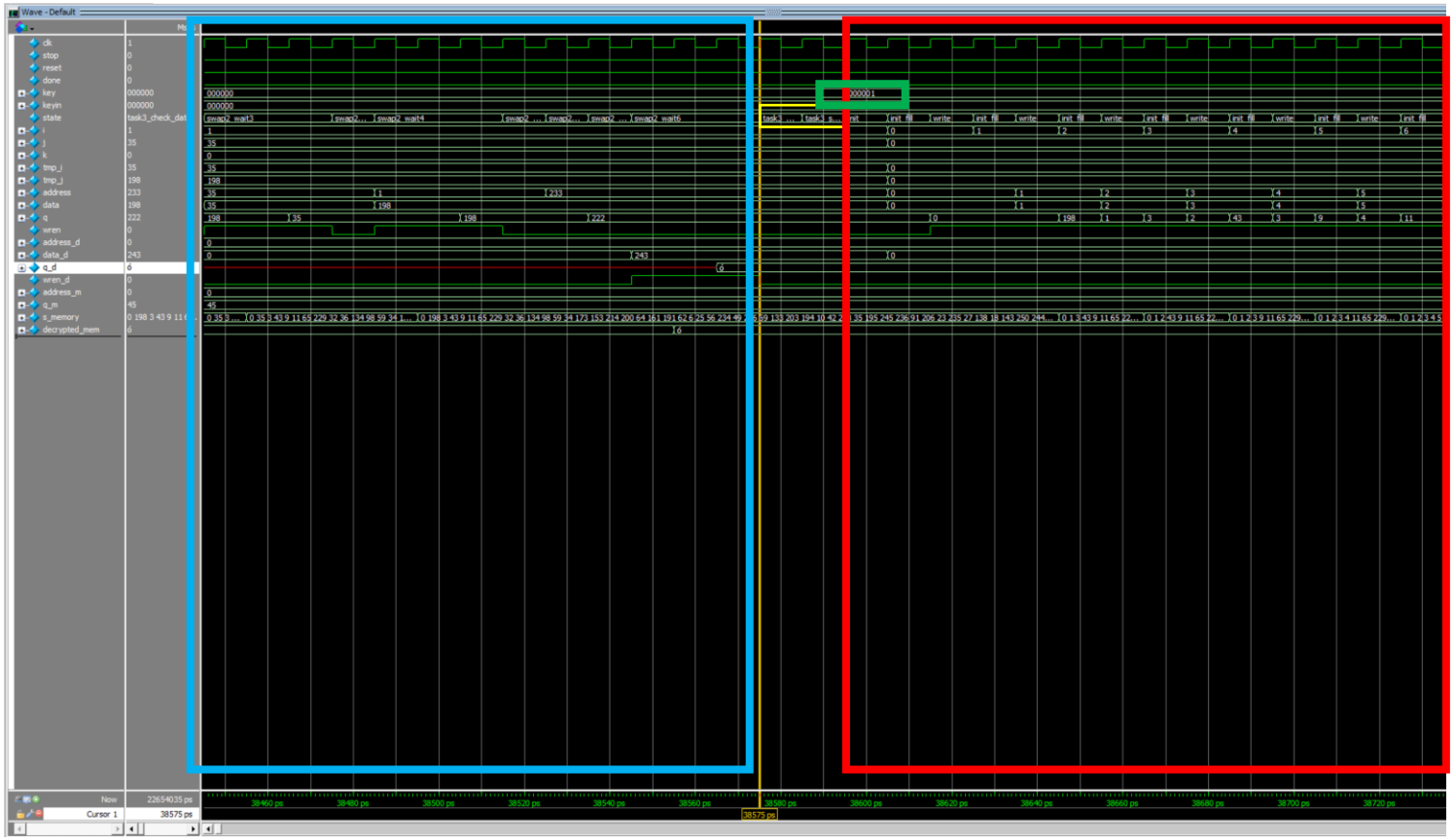


- Step 2a ends at 38365ps.

```
i = 0, j=0
for k = 0 to message_length-1 { // message_length is 32 in our implementation
i = (i+1) mod 256
j = (j+s[i]) mod 256
swap values of s[i] and s[j]
f = s[ (s[i]+s[j]) mod 256 ]
decrypted_output[k] = f xor encrypted_input[k] // 8 bit wide XOR function
}
```

- Blue box: swap1
- Red box: swap 2 FSM, start decrypting
- Yellow box: step 2b states

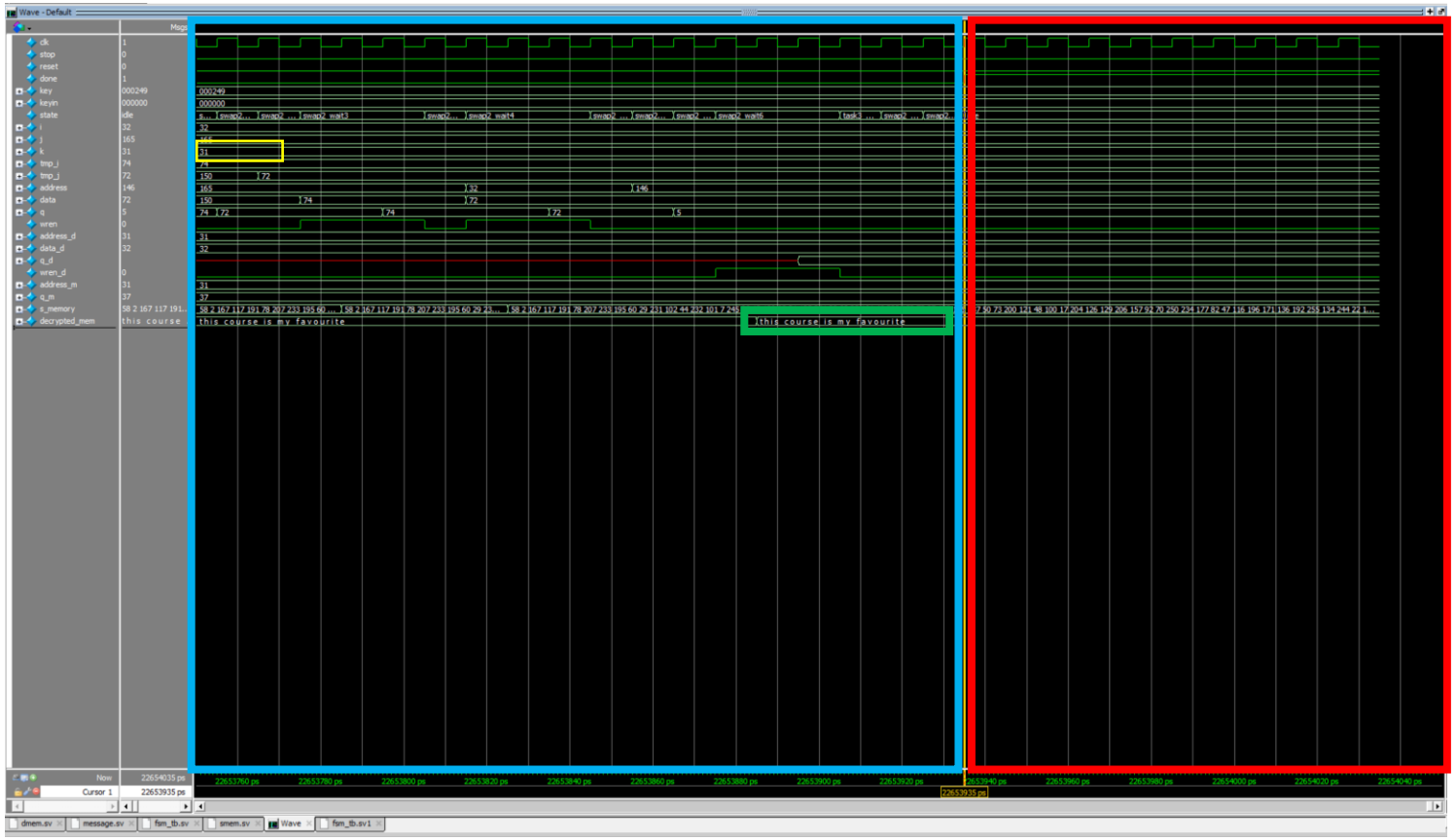
Simulation: Step2b & 3



- First Step 2b decryption ends at 38575ps. The decrypted data is not a letter.

- Blue box: swap2
- Yellow box: Step3 detect FSM
- Red box: Restart with new key
- Green box: New key in new cycle

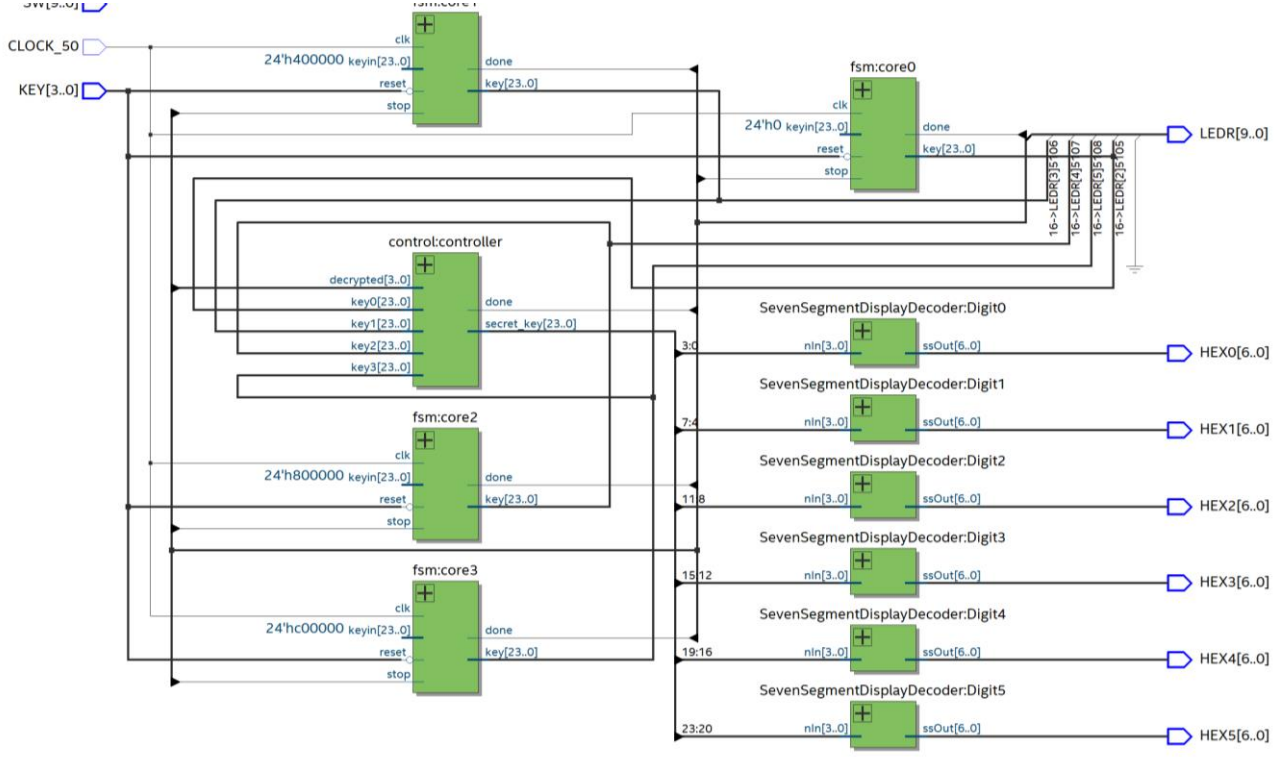
Simulation: Step3



- When message is finally decrypted, k runs out to 31, and decrypted message is stored in decreyted_mem.

- Blue box: swap2
- Yellow box: value of k
- Red box: Idle state
- Green box: Decrypted message

Bonus



- Design: Run 4 cores, each one has it's only memories, and each one runs different decryption range.
  - Core0: Start from 24'h0
  - Core1: Start from 24'h400000
  - Core2: Start from 24'h800000
  - Core3: Start from 24'hC00000
- Once a solution is found, stop all cores and display the result.

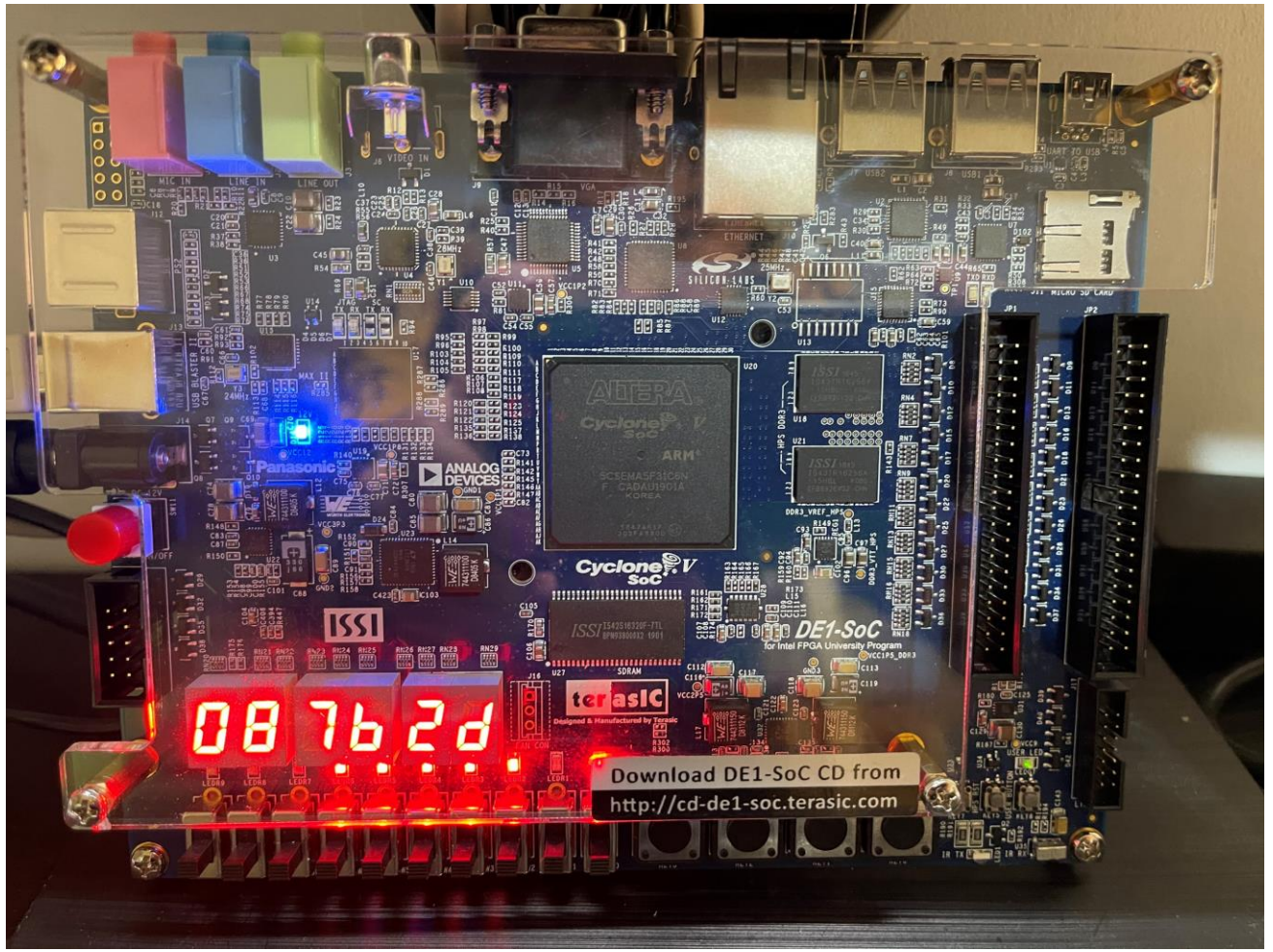
Interface design

LED[9:6]: which core decrypted data first

LED[5:2]: toggle if running

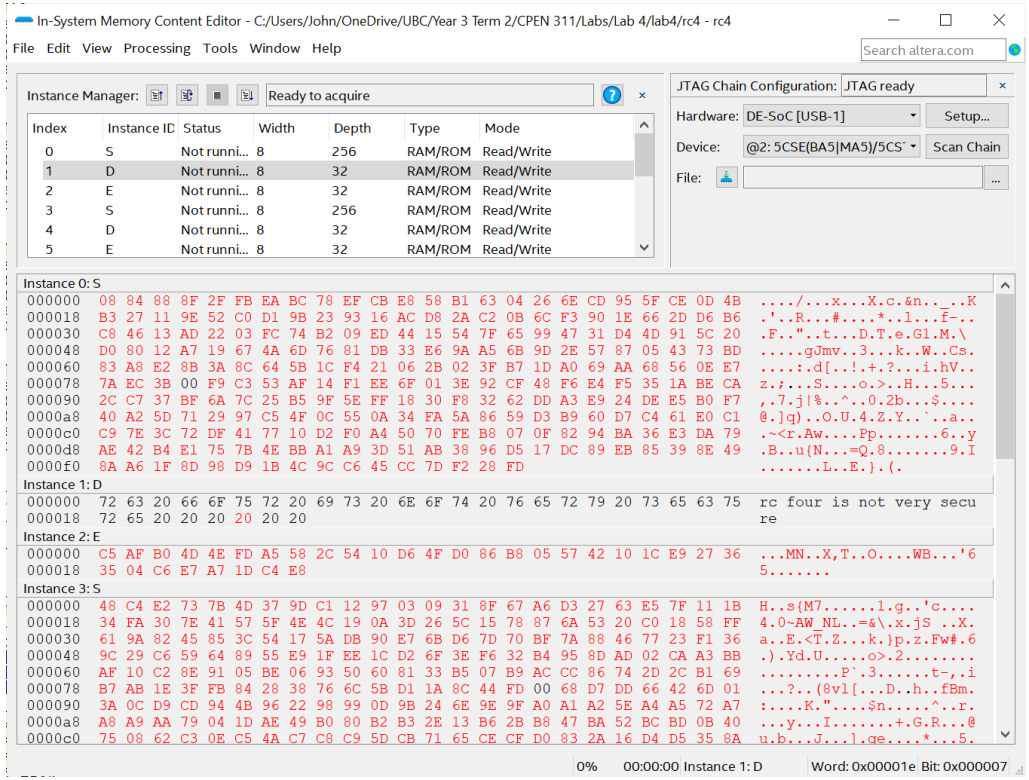
LED[0]: on if done

HEX: decrypted key

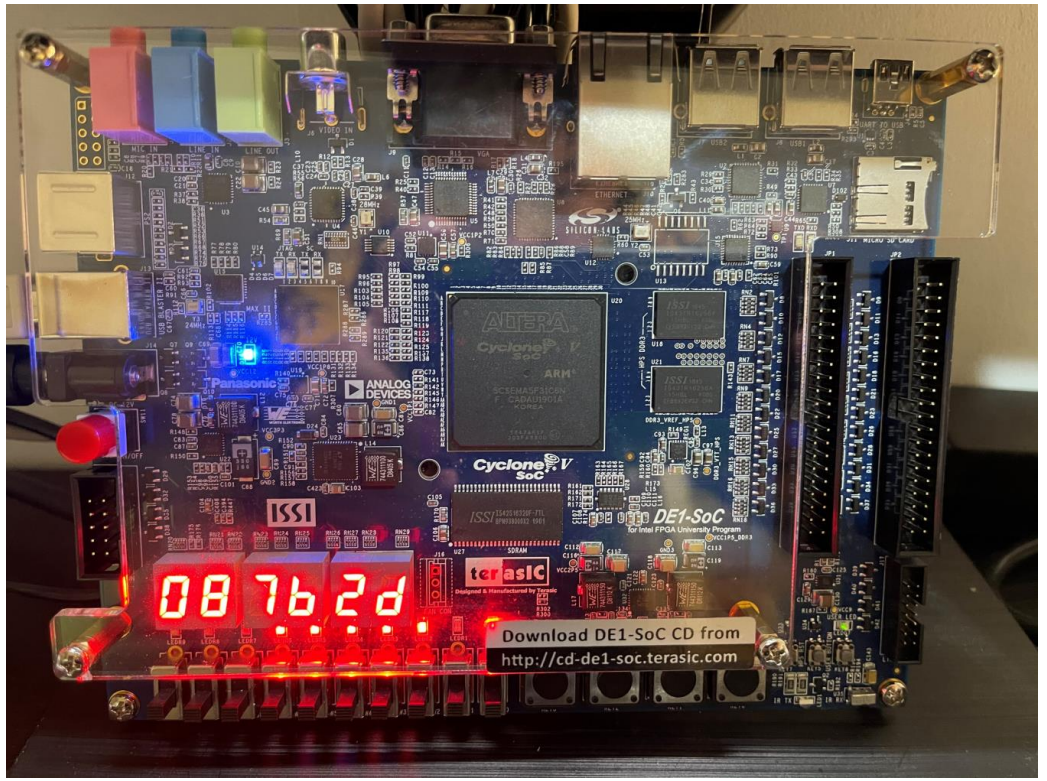


Bonus-Example Decrypted Messages 4

Message



Secret Key



LED[9:6] indicate core[3:0] separately.
LED on means the message is decrypted by which core.

Bonus-Example Decrypted Messages 5

Message

In-System Memory Content Editor - C:/Users/John/OneDrive/UBC/Year 3 Term 2/CPEN 311/Labs/Lab 4/lab4/rc4 - rc4

File Edit View Processing Tools Window Help

Search altera.com

Instance Manager: Ready to acquire

JTAG Chain Configuration: JTAG ready

Hardware: DE-SoC [USB-1] Setup...

Device: @2: 5CSE(BA5|MA5)/5CS Scan Chain

File:

Index	Instance ID	Status	Width	Depth	Type	Mode
1	D	Not runni...	8	32	RAM/ROM	Read/Write
2	E	Not runni...	8	32	RAM/ROM	Read/Write
3	S	Not runni...	8	256	RAM/ROM	Read/Write
4	D	Not runni...	8	32	RAM/ROM	Read/Write
5	E	Not runni...	8	32	RAM/ROM	Read/Write
6	S	Not runni...	8	256	RAM/ROM	Read/Write

Instance 0: S

```
000000  23 7A B0 34 33 44 36 D2 28 CD 43 72 E7 A8 D8 C2 BE 7C 6B 15 87 4D 99 B3  #z.43D6.(.Cr.....|k..M..
000018  82 B4 F5 92 55 0E 21 4C 75 52 D1 E5 E1 CE 2A 20 9D C8 F6 F9 09 06 AC FE  ....U.!LuR....* ........
000030  26 90 79 9A 46 DB 70 88 9B A7 57 2C 3F 86 E4 0A F4 D0 3A 12 A5 EE 22 29  &.y.F.p...W,?.....:..")
000048  47 85 4A DA 73 FA C9 8E 25 1D 83 B5 38 7D E3 FC 1C 18 BC 16 2E 13 77 AD  G.J.s...%...8}........w.
000060  94 24 48 58 89 32 C4 5F 98 B2 1B FF 8B DC DF C6 51 35 8F E8 2F 9E 50 1A  .$HX.2._........Q5../.P.
000078  D5 C5 03 BD B1 37 63 CA 59 EF C3 67 7F AB 45 A9 CF A0 95 8A 39 F7 4F 05  .....7c.Y..g..E.....9.O.
000090  31 3B 7B CC D6 3C A2 0F EA 6A 96 D7 B8 CB 5D 10 71 27 DE 97 17 19 5E F1  1;{..<...j....].q'....^.
0000a8  BB 14 C0 8D 5A AE 53 00 E0 B7 54 E6 6F 3D D4 30 C7 C1 02 EC 5B F8 42 93  ....Z.S...T.o=.0....[.B.
0000c0  DD 2D 76 6C 65 1E 41 0B 7E E2 84 40 F0 04 FD D3 9F 60 49 2B 0D F2 08 11  .-vle.A.~..@.....`I+....
0000d8  5C A1 78 4B 07 F3 BA A4 64 61 66 6E 0C 1F 8C 3E 74 EB 69 ED 91 81 A6 BF  \.xK....dafn...>t.i.....
0000f0  E9 6D B9 4E A3 56 01 62 9C AF 80 68 FB B6 AA D9                          .m.N.V.b...h....
```

Instance 1: D

```
000000  79 6F 75 20 6E 6F 77 20 68 61 76 65 20 73 6F 6C 69 64 20 76 68 64 6C 20  you now have solid vhdl
000018  73 6B 69 6C 6C 73 20 20                                                  skills
```

Instance 2: E

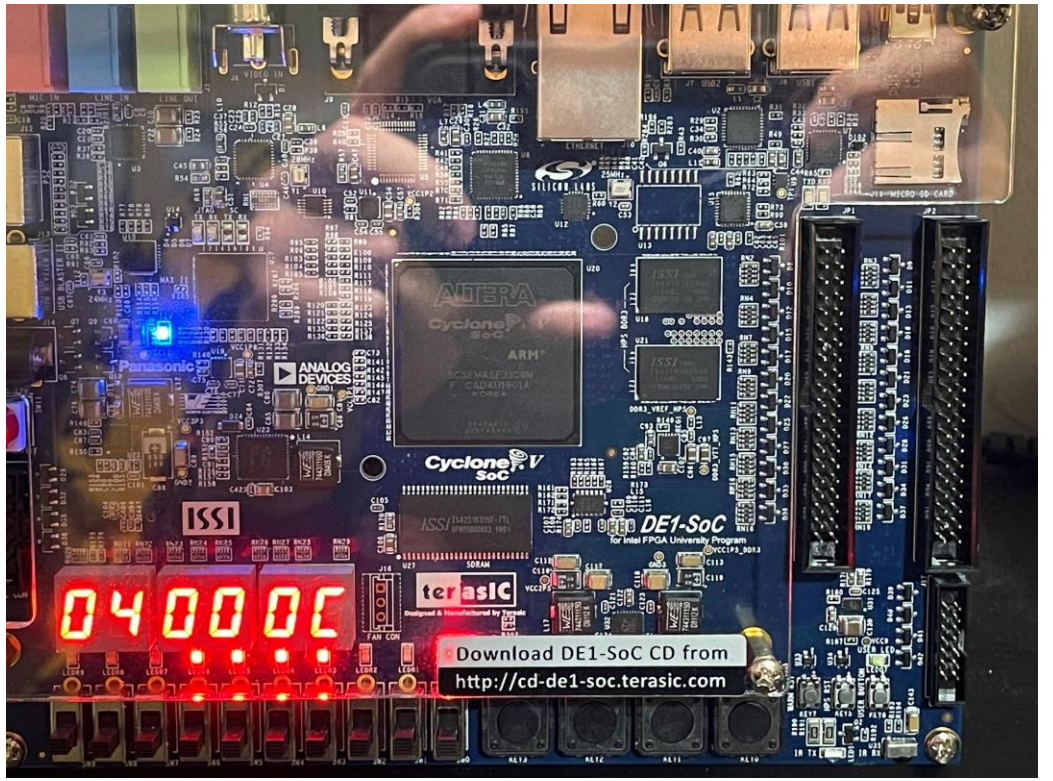
```
000000  22 03 BC 50 91 2E 28 9A 2B F0 E7 D6 54 D3 F2 17 B5 C4 FF 96 5C 38 18 A3  "..P..(.+...T.......\8..
000018  0F DC B5 62 C0 70 8E 03                                                  ...b.p..
```

Instance 3: S

```
000000  2E 00 42 39 8B 10 D6 DC E1 2D AE 3E 7C 1A 97 28 8D 55 68 78 89 08 26 36  ..B9.....-.>|..(.Uhx..&6
000018  65 AD 94 F2 0A 27 03 5D C3 84 19 12 61 F3 53 2A AA 77 34 5E 85 3D 43 87  e....'.]....a.S*.w4^.=C.
000030  22 8E FE 74 41 0B 99 52 2C 7F 3C 33 66 D3 31 6B 75 9D 45 02 BA 79 38 CB  "..tA..R,.<3f.1ku.E..y8.
000048  5A 11 1F 4D 98 E0 71 4A 73 A0 06 0F BD 8C 64 4C 13 AB 62 A3 FB 9B C5 1D  Z..M..qJs.....dL..b.....
000060  C4 CF 83 C7 0C B0 F1 5B C2 6E D7 B7 EE 9F C0 AC 15 2F DE 05 1E 16 B5 63  .......[.n......./.....c
000078  96 49 81 91 50 48 BB 88 B6 A9 5C 6D 5F A5 A1 B1 EB FC C9 CC 58 0E 24 54  .I..PH....\m_.......X.$T
000090  90 17 0D F4 1C 30 76 A4 32 4E 95 A6 57 7A 23 B2 EA F6 B9 A2 7E 92 BF 6C  .....0v.2N..Wz#.....~..l
0000a8  D9 09 86 60 72 7D 7B B8 93 A7 2B B3 B4 01 80 3F AF 67 44 56 BC 8F BE 4F  ...`r}{...+....?.gDV...O
0000c0  69 C1 9E 20 59 9A C6 29 C8 8A CA 47 04 CD CE 35 D0 D1 D2 21 D4 D5 37 6A  i.. Y..)...G...5...!..7j
```

0% 00:00:00 Instance: Word: Bit:

Secret Key

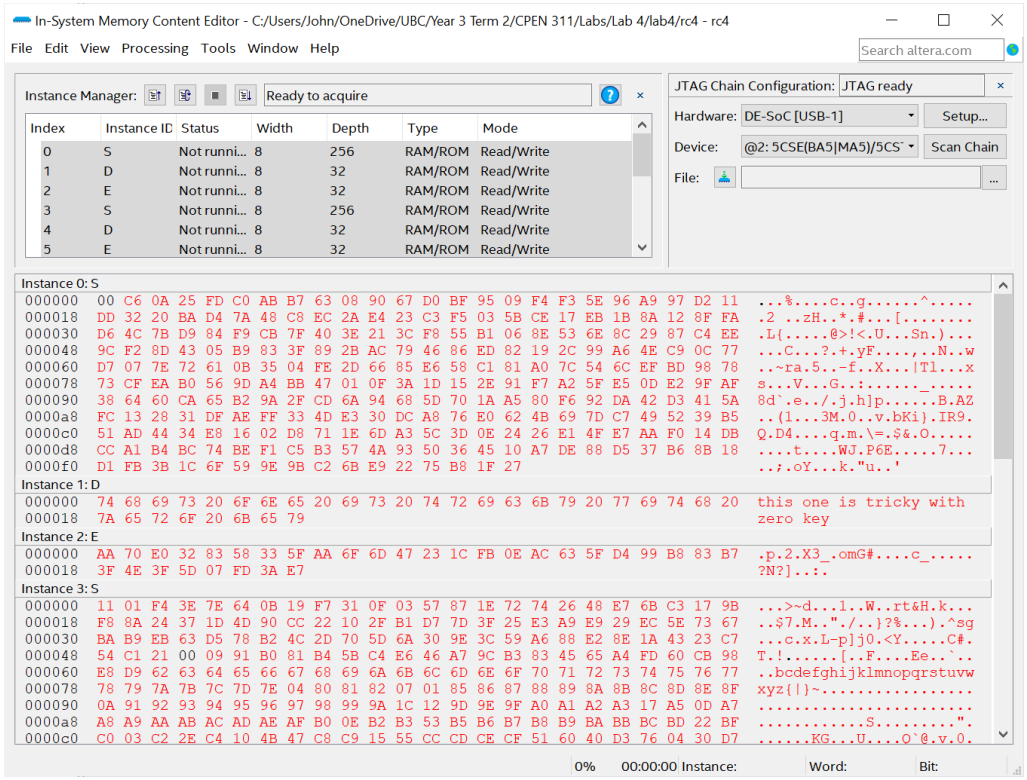


LED[9:6] indicate core[3:0] separately.

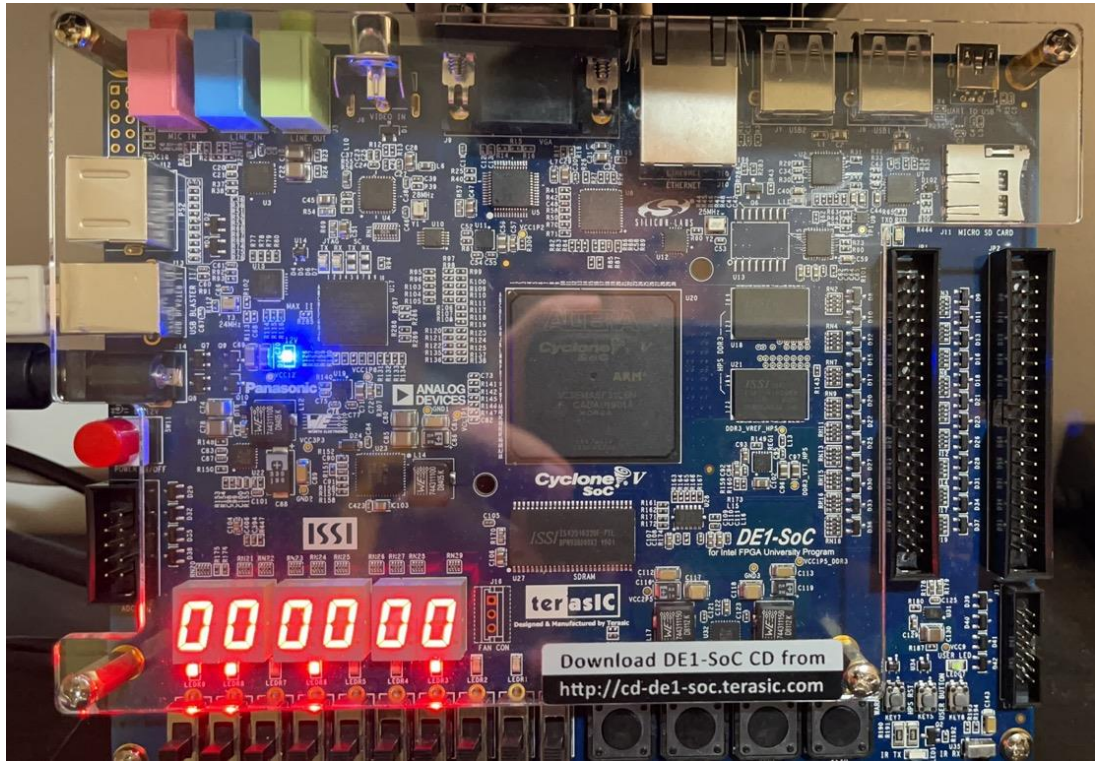
LED on means the message is decrypted by which core.

Bonus-Example Decrypted Messages 6

Message



Secret Key



LED[9:6] indicate core[3:0] separately.
LED on means the message is decrypted by which core.

Bonus-Example Decrypted Messages 7

Message

In-System Memory Content Editor - C:/Users/John/OneDrive/UBC/Year 3 Term 2/CPEN 311/Labs/Lab 4/lab4/rc4 - rc4

File Edit View Processing Tools Window Help

Search altera.com

Instance Manager: Ready to acquire

JTAG Chain Configuration: JTAG ready

Hardware: DE-SoC [USB-1] Setup...

Device: @2: 5CSE(BA5|MA5)/5CS Scan Chain

File:

Index	Instance ID	Status	Width	Depth	Type	Mode
0	S	Not runni...	8	256	RAM/ROM	Read/Write
1	D	Not runni...	8	32	RAM/ROM	Read/Write
2	E	Not runni...	8	32	RAM/ROM	Read/Write
3	S	Not runni...	8	256	RAM/ROM	Read/Write
4	D	Not runni...	8	32	RAM/ROM	Read/Write
5	E	Not runni...	8	32	RAM/ROM	Read/Write

Instance 0: S

```
000000  4A 2D 1C 25 B0 51 62 FF D9 E2 AA C2 3A 1B 93 16 A7 EF EA E0 2F F3 D6 27  J-.%.Qb.....:......./..'
000018  42 A1 57 13 DC C7 97 5A 95 69 9F D1 05 09 3B DE 07 82 CE B2 CA 46 59 AB  B.W....Z.i....;......FY.
000030  BB 50 91 A8 FD 6C 01 54 60 7B 53 9D 6F 40 C9 AC AE FE CB 08 39 5C 87 E4  .P...l.T`{S.o@......9\..
000048  9E 4B A4 4E 0C 0B 85 41 AF 38 74 79 CF 28 A9 F6 BE 4D 8E C3 83 72 D4 68  .K.N...A.8ty.(...M...r.h
000060  5B 49 35 90 7C A5 2B 67 F5 8C B3 EC 10 0E E7 44 EB D2 48 64 C0 80 88 B1  [I5.|.+g.......D..Hd....
000078  B9 F1 7A D3 8D 9B 19 47 1A E9 34 C4 03 E6 F7 9C F8 6A 0D 58 D8 B4 C6 23  ..z....G..4......j.X...#
000090  A6 F9 9A BC B7 37 14 29 1F 56 1D F2 20 A0 26 00 84 06 8A 70 02 A3 30 17  .....7.).V.. .&....p..0.
0000a8  CD 6E 32 89 FA 2E 73 DB B8 55 D7 96 22 8B 3C DD 15 99 E8 C8 B5 FB 71 E3  .n2...s..U.."..<.......q.
0000c0  BD C5 ED DA 86 52 36 7E 65 2A 98 3E 33 77 E1 31 C1 12 11 FC 3F AD 6B 5D  .....R6~e*.>3w.1....?.k]
0000d8  63 4F 94 F0 1E 6D CC 24 BF 81 2C 4C 45 75 EE 92 F4 D5 0A 5F D0 3D 8F 04  cO...m.$..,LEu....._.=..
0000f0  66 18 43 E5 DF 78 76 5E 21 A2 B6 7D 7F BA 61 0F                          f.C..xv^!..}..a.
```

Instance 1: D

```
000000  76 68 64 6C 20 6E 69 6E 6A 61 20 73 61 76 65 73 20 74 68 65 20 64 61 79  vhdl ninja saves the day
000018  20 20 20 20 20 20 20 20                                                  
```

Instance 2: E

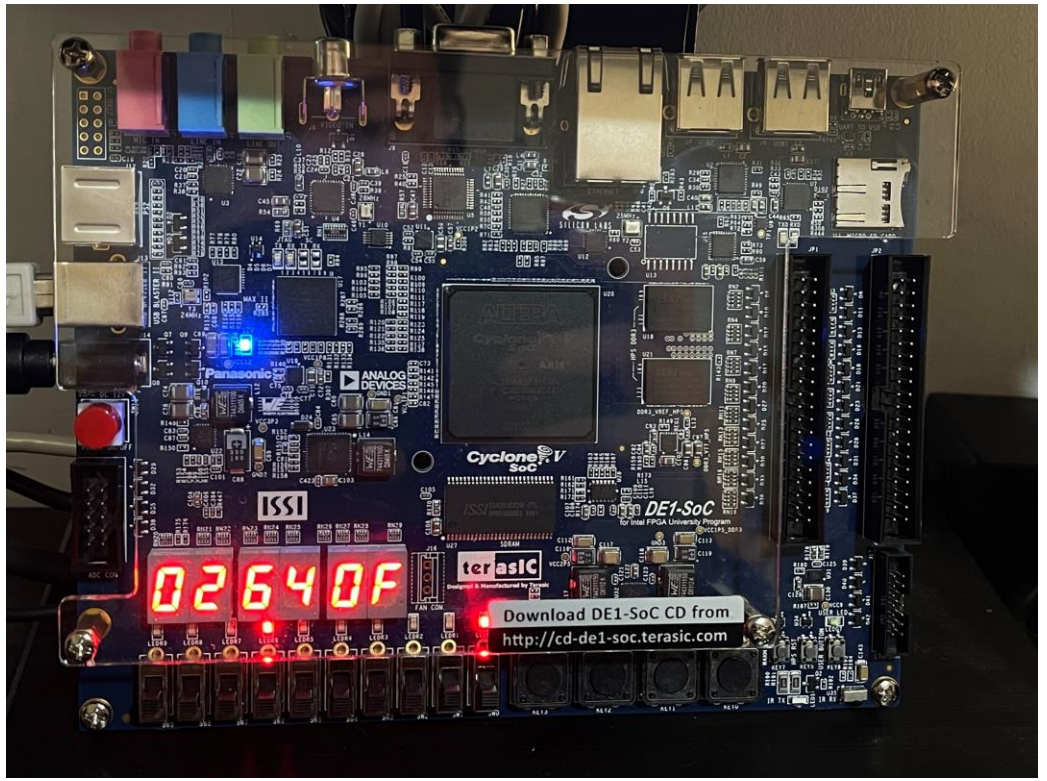
```
000000  C1 0C 35 D0 8D 27 66 4B 0D AD 4B B6 31 29 32 C2 90 0D 68 1E 76 5E 02 62  ..5..'fK..K.1)2...h.v^.b
000018  DA 45 70 C3 A5 92 74 BE                                                  .Ep...t.
```

Instance 3: S

```
000000  20 A6 A7 EC 43 57 9F 33 10 59 23 D0 1E 8E 9B 03 05 28 BB 31 16 0E 4C 14  ...CW.3.Y#.....(.1..L.
000018  84 1A 42 38 41 26 5F E1 2B 63 E8 3A 70 F8 67 86 1D 39 92 08 4B BA 2C 79  ..B8A&_.+c.:p.g..9..K.,y
000030  EB 61 17 09 9E D2 4A E4 76 3F C6 52 DB 7B 04 27 DC F5 5E 53 2A 34 47 8C  .a....J.v?.R.{.'..^S*4G.
000048  44 C2 F7 11 48 62 F2 A4 F3 29 5D B8 D4 46 1F 19 58 07 5A 5B 5C 3B 00 0C  D...Hb...)]..F..X.Z[\;..
000060  60 13 4D 21 64 65 66 2E 68 69 6A 6B 6C 6D 6E 6F 24 71 72 73 74 75 1B 77  `.M!def.hijklmno$qrstu.w
000078  78 2F 7A 3D 7C 7D 7E 7F 80 81 82 83 18 85 51 87 88 89 8A 8B 55 8D 0D 8F  x/z=|}~.......Q.....U...
000090  90 91 32 93 94 95 96 97 98 99 9A 15 9C 9D 45 06 A0 A1 A2 A3 4F A5 01 02  ..2.........E.....O...
0000a8  A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 3E B9 2D 12 BC BD BE BF  ..............>.-.....
0000c0  C0 C1 49 C3 C4 C5 0A C7 C8 C9 CA CB CC CD CE CF 0B D1 35 D3 54 D5 D6 D7  ..I.............5.T...
```

0% 00:00:00 Instance: Word: Bit:

Secret Key

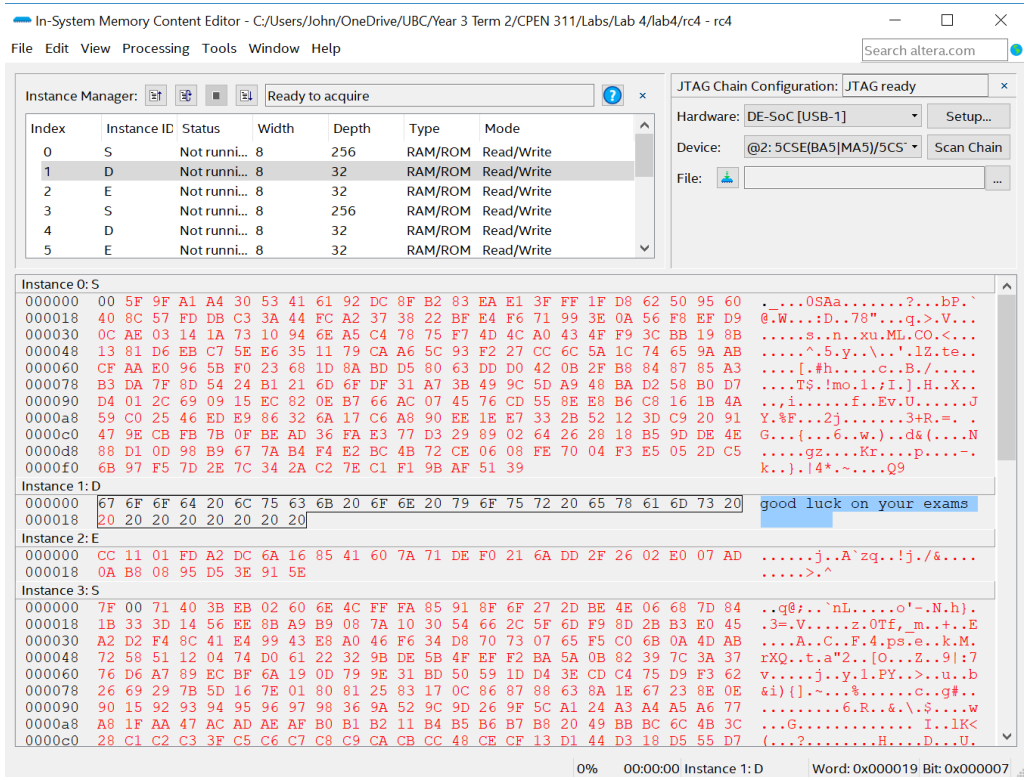


LED[9:6] indicate core[3:0] separately.

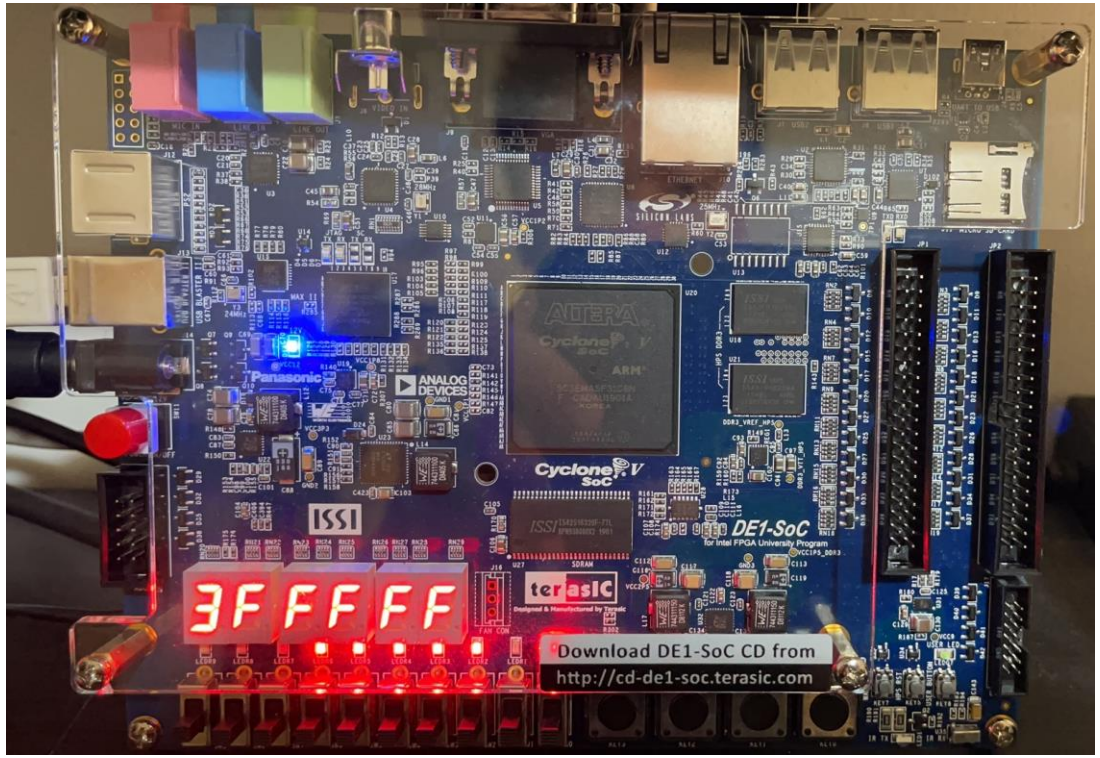
LED on means the message is decrypted by which core.

Bonus - Example Decrypted Messages 8

Message



Secret Key： 3FFFFF



LED[9:6] indicate core[3:0] separately.
LED on means the message is decrypted by which core.

Bonus - Example Decrypted Messages 8

Secret Key： 3FFFFF

Setting:

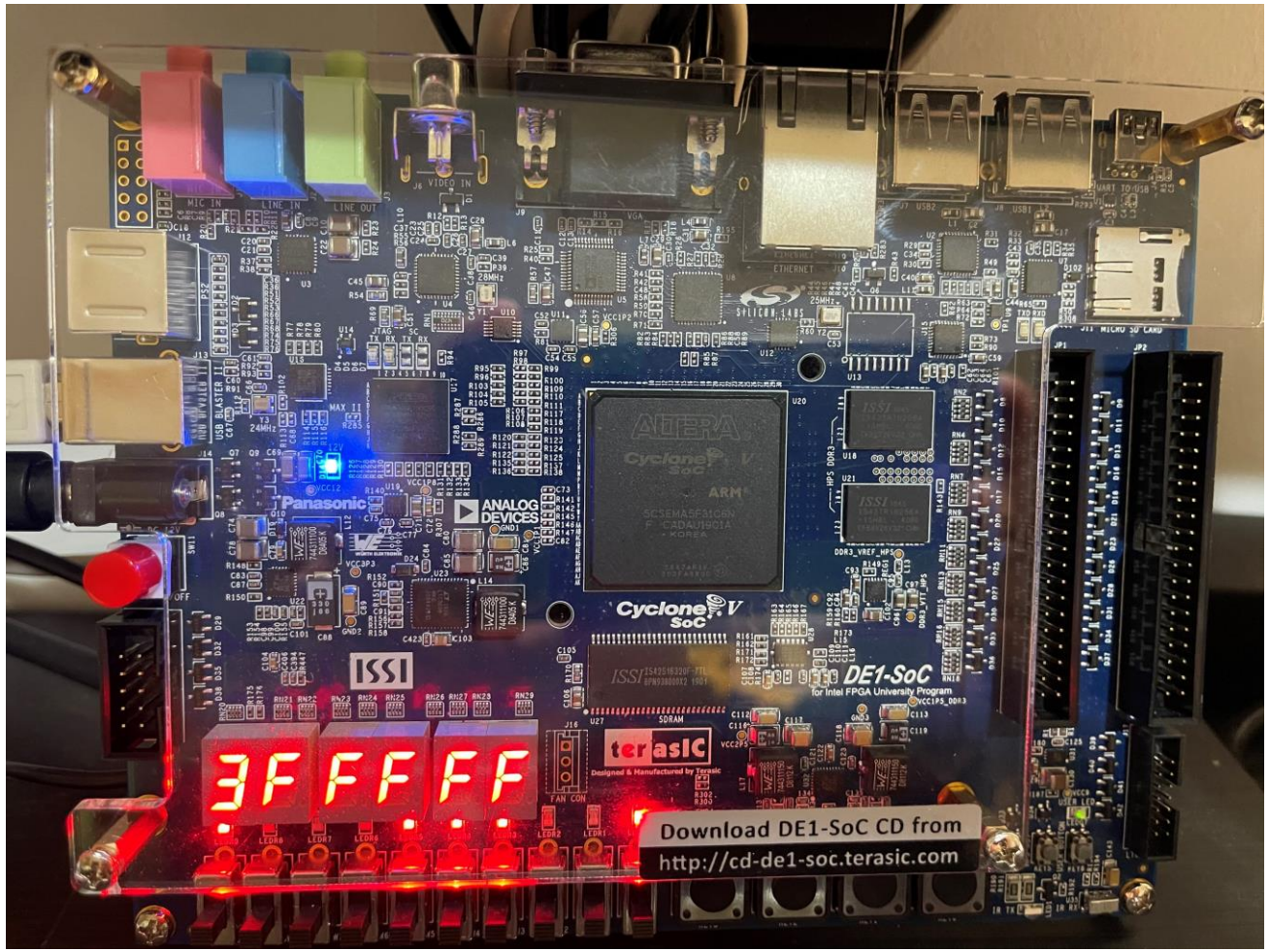
Core0: Start from 24’h0

Core1: Start from 24’h010000

Core2: Start from 24’h020000

Core3: Start from 24’h030000

The result is as expected: Multicore decryption works fine.



Design Integrity

No latch found