Triple DES

Group member: Mengshi Feng, Yiyi Chen, Tian Qiu, Siyi Cai
Lab section: 5

OVERVIEW

What have we designed?

Our design is to implement 3DES algorithm (Data Encryption Standard algorithm) on FPGA.

Why would someone want to use your design?

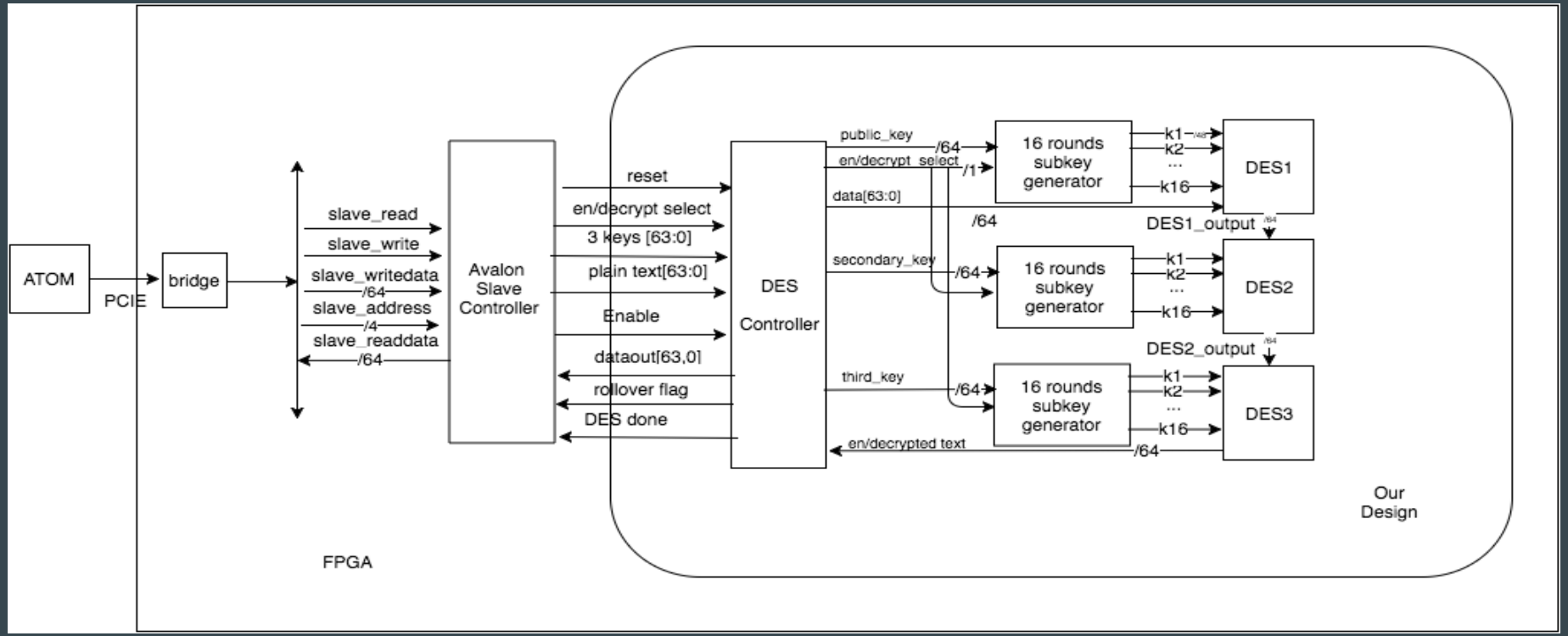
We optimized the speed of current 3DES by pipelining data, which will decrease total number of rounds.

Why is this design appropriate for ASIC?

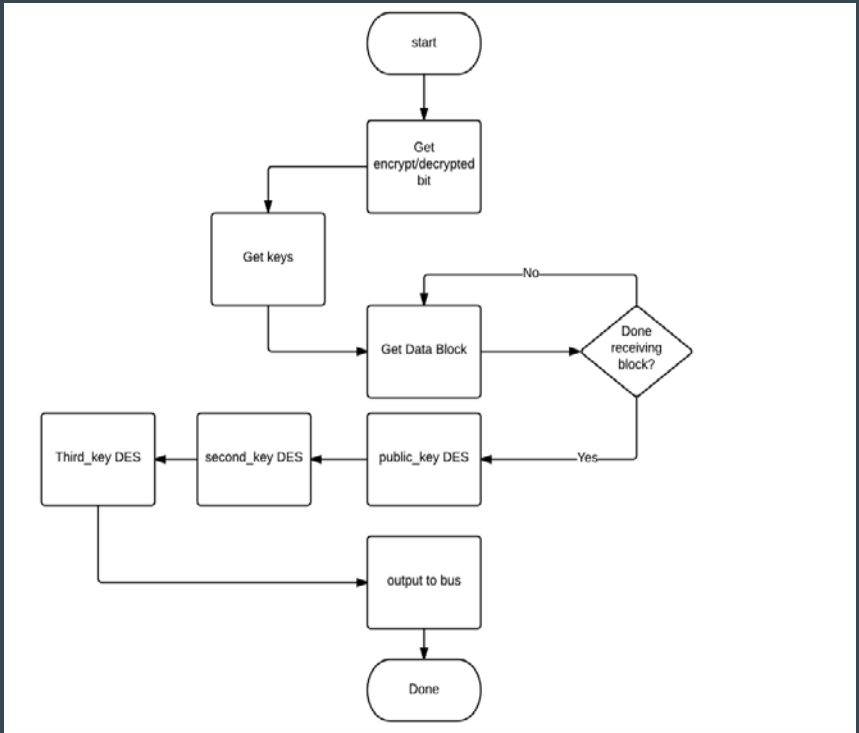
3DES algorithm is ideal for hardware, which use arithmetic and logic operations, 64-bit numbers(plaintext and keys). It is appropriate for ASIC design using system verilog, which allows concurrent statement and combinational logic. Parallel calculation will maximum the efficiency of the implementation.

System Design

Architecture diagram



System Design



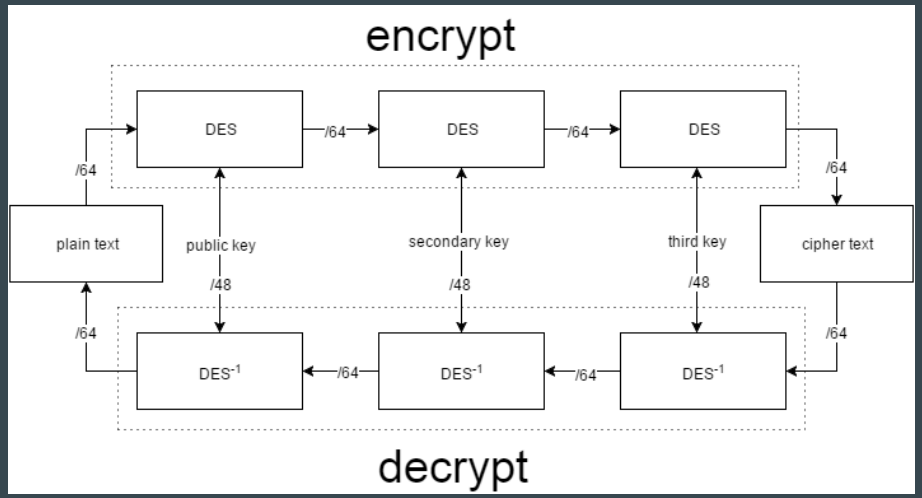
Design flow chart

Main feature:

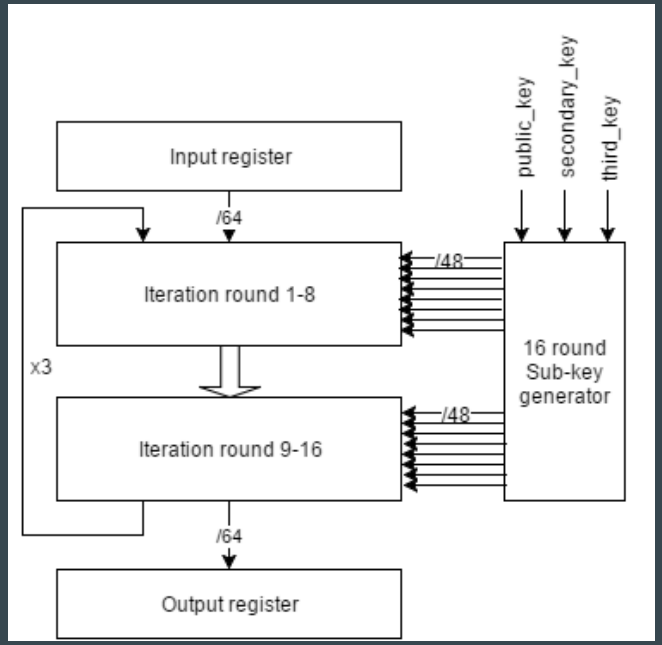
Triple-DES encrypt and decrypt

Avalon bus communicate with FPGA
Pipelining to improve speed

System Design



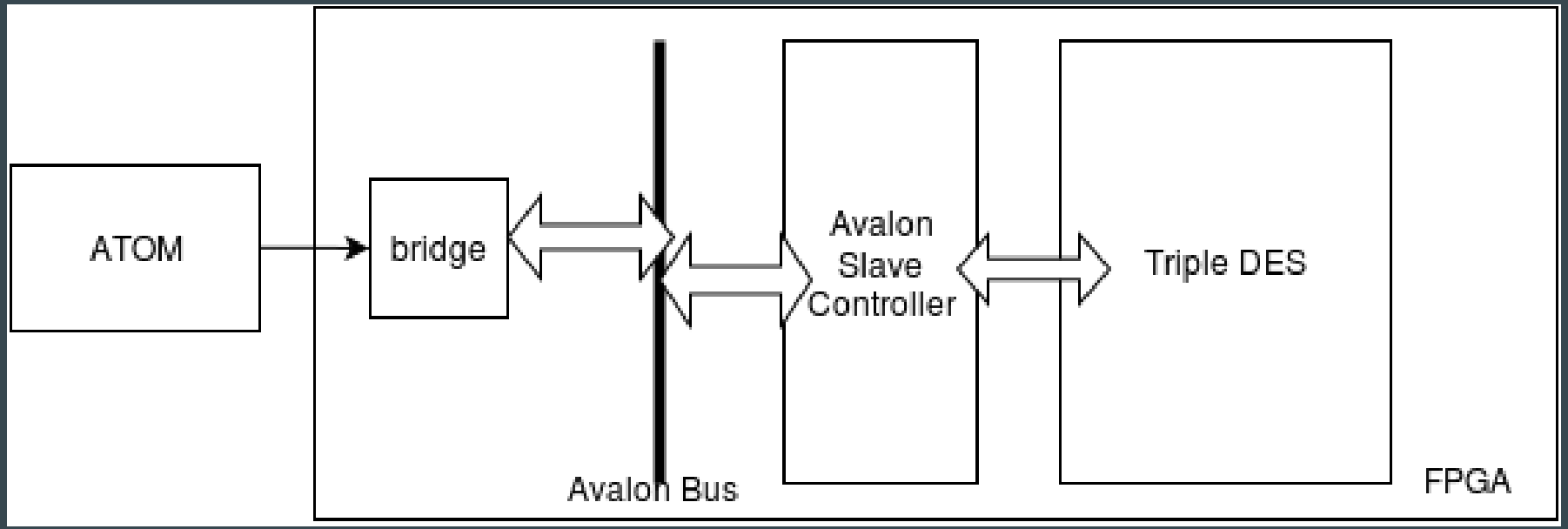
The cipher is symmetric, so decryption uses the same keys with reversed order and the same algorithm as in encryption.



DES pipeline

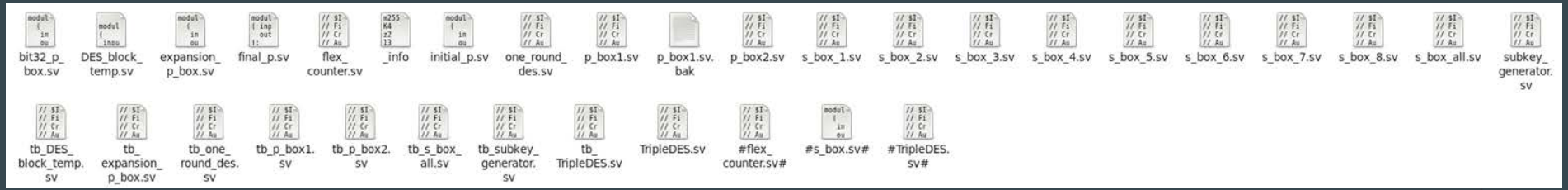
System Design

External interface:



Results – Success Criteria Status and Results

1. Test benches exist for all top level components and the entire design. The test benches for the entire design can be demonstrated or documented to cover all of the functional requirements given in the design specific success criteria. (2 pts)



2. Entire design synthesizes completely, without any inferred latches, timing arcs, and, sensitivity list warnings (4 pts)

```
Point                                                   Incr       Path
-----------------------------------------------------------------------
des1/cnt1_8/rollover_flag_reg/CLK (DFFSR)               0.00       0.00 r
des1/cnt1_8/rollover_flag_reg/Q (DFFSR)                 0.60       0.60 f
des1/cnt1_8/rollover_flag (flex_counter_NUM_CNT_BITS4_5)
                                                        0.00       0.60 f
des1/rollover1 (DES_block_temp_2)                       0.00       0.60 f
rollover1 (out)                                         0.00       0.60 f
data arrival time                                                  0.60
-----------------------------------------------------------------------
(Path is unconstrained)
```

File Edit View Search Tools Documents Help

Open Save Undo

TripleDES.log

```
Inferred memory devices in process
        in routine DES_block_temp line 157 in file
                './source/DES_block_temp.sv'.
===============================================================================
|    Register Name    |   Type    | Width | Bus | MB | AR | AS | SR | SS | ST |
===============================================================================
|     clear1_reg      | Flip-flop |   1   |  N  | N  | Y  | N  | N  | N  | N  |
|  round_signal1_reg  | Flip-flop |   1   |  N  | N  | Y  | N  | N  | N  | N  |
===============================================================================

Inferred memory devices in process
        in routine DES_block_temp line 179 in file
                './source/DES_block_temp.sv'.
===============================================================================
|    Register Name    |   Type    | Width | Bus | MB | AR | AS | SR | SS | ST |
===============================================================================
|   enable_nxt_reg    | Flip-flop |   1   |  N  | N  | Y  | N  | N  | N  | N
|  round_signal2_reg  | Flip-flop |   1   |  N  | N  | Y  | N  | N  | N  | N
|     clear2_reg      | Flip-flop |   1   |  N  | N  | Y  | N  | N  | N  | N
===============================================================================
Presto compilation completed successfully.
Information: Building the design 'flex_counter' instantiated from design 'DES_
        the parameters "4". (HDL-193)

Inferred memory devices in process
        in routine flex_counter_NUM_CNT_BITS4 line 25 in file
                './source/flex_counter.sv'.
===============================================================================
|    Register Name    |   Type    | Width | Bus | MB | AR | AS | SR | SS | ST
===============================================================================
|  rollover_flag_reg  | Flip-flop |   1   |  N  | N  | Y  | N  | N  | N  | N  |
|    count_out_reg    | Flip-flop |   4   |  Y  | N  | Y  | N  | N  | N  | N  |
===============================================================================
Presto compilation completed successfully.
Information: Building the design 'subkey_generator'. (HDL-193)
Presto compilation completed successfully.
Information: Building the design 'one_round_des'. (HDL-193)
Presto compilation completed successfully.
Information: Building the design 'p_box1'. (HDL-193)
Presto compilation completed successfully.
Information: Building the design 'p_box2'. (HDL-193)
Presto compilation completed successfully.
Information: Building the design 'expansion_p_box'. (HDL-193)
Presto compilation completed successfully.
Information: Building the design 's_box_all'. (HDL-193)
Presto compilation completed successfully.
Information: Building the design 'bit32_p_box'. (HDL-193)
Presto compilation completed successfully.
Information: Building the design 's_box_1'. (HDL-193)
```

Find

Search for: latch

- [ ] Match case
- [ ] Match entire word only
- [ ] Search backwards
- [x] Wrap around

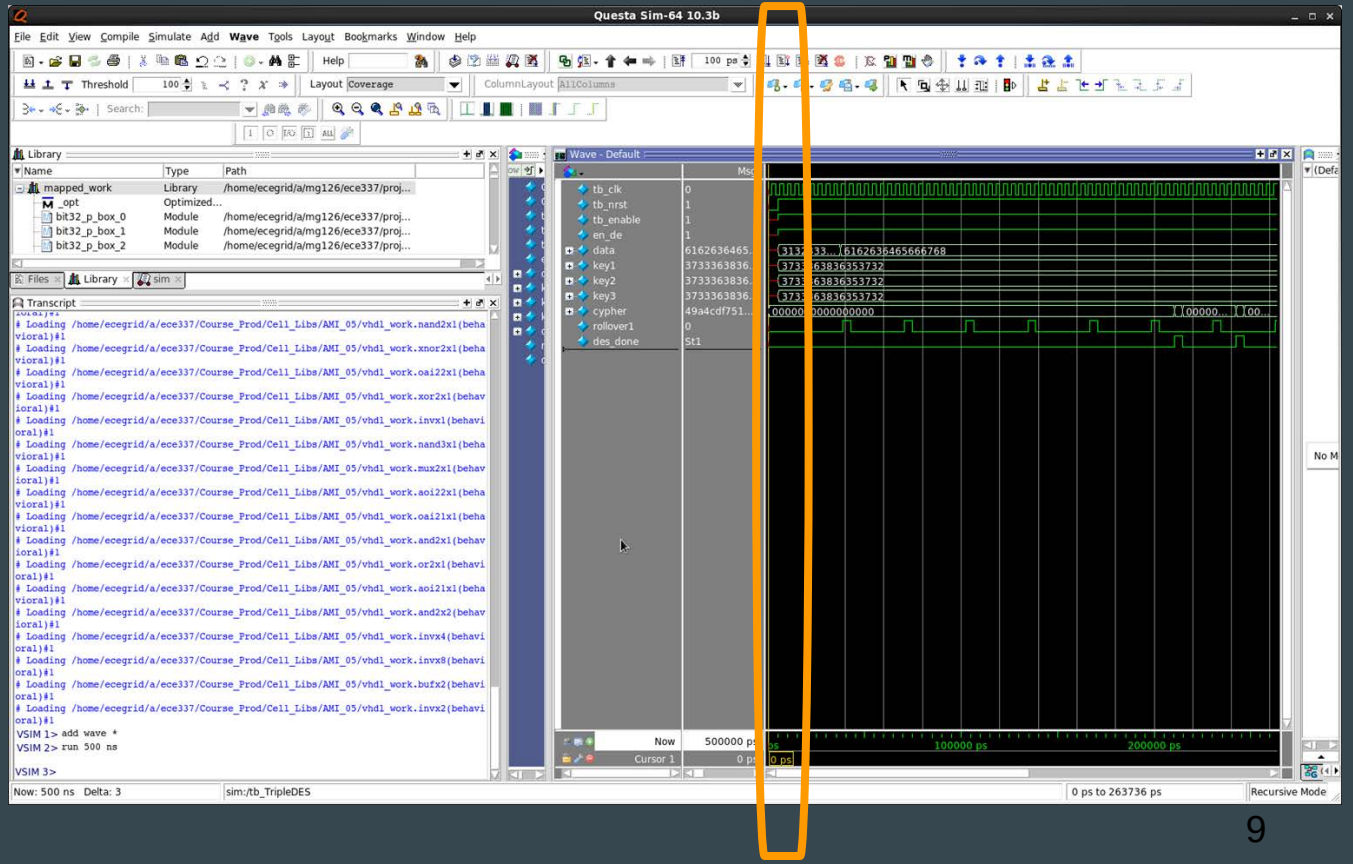
Close Find

"latch" not found

Plain Text Tab Width: 8 Ln 146, Col 63 INS

Results – Success Criteria Status and Results

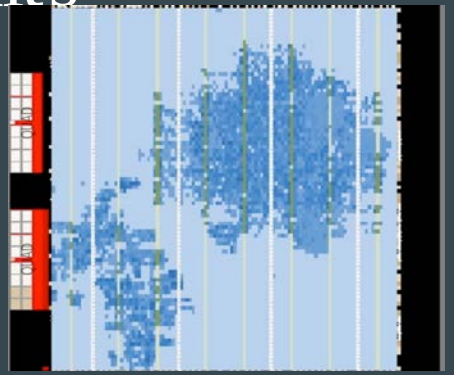
3. Source and mapped version of the complete design behave the same for all test cases. The mapped version simulates without timing errors except at time zero (2 pts)



Result - Success Criteria Status and Results

4. A complete IC layout is produced that passes all geometry and connectivity checks (2 pts)

FPGA



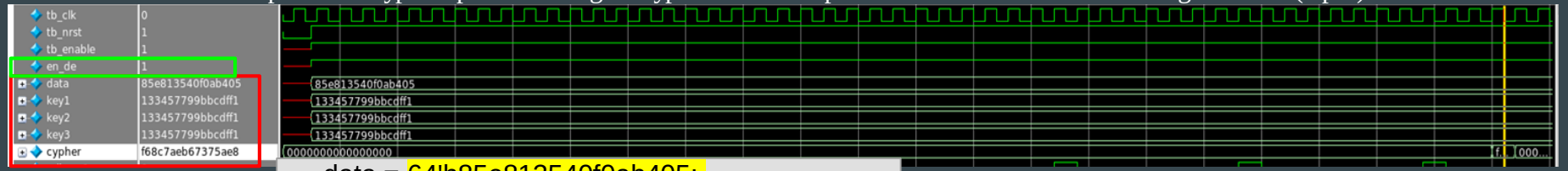
5. The entire design complies with targets for area, pin count, throughput (if applicable), and clock rate. The final targets for these parameters will be determined by course staff based on your design review. Failure to reach any of the targets will result a score of 1 out of 2 provided that you are within 50% on area, 10% on pin count, and 25% on throughput. Doing worse in any category will result in a score of 0. (2 pts)

FPGA

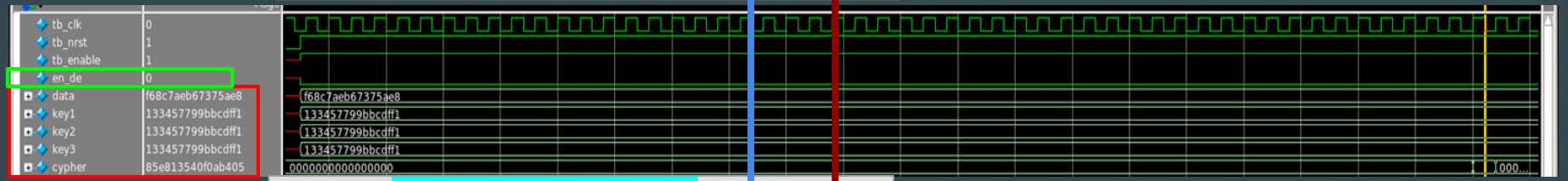
```
****************************************
Report : area
Design : TripleDES
Version: K-2015.06-SP1
Date   : Fri Apr 29 11:53:30 2016
****************************************

Library(s) Used:

    osu05_stdcells (File: /package/eda/cells/OSU/v2.7/synopsys/lib/ami05/osu05_stdcells.db)

Number of ports:                        11985
Number of nets:                         25787
Number of cells:                        14510
Number of combinational cells:          13123
Number of sequential cells:              1250
Number of macros/black boxes:               0
Number of buf/inv:                       3810
Number of references:                       8

Combinational area:            3639231.000000
Buf/Inv area:                   560160.000000
Noncombinational area:          985248.000000
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:               4624479.000000
Total area:                 undefined
```

Results – Key Design Specific Success Criteria Results

1. The output of decryption process using encrypted data as input should be the same as the original data. (2 pts)



data = 64'h85e813540f0ab405;
key1 = 64'h133457799BBCDFF1;
key2 = 64'h133457799BBCDFF1;
key3 = 64'h133457799BBCDFF1;
Cypher = 64'hf68c7aeb67375ae8;

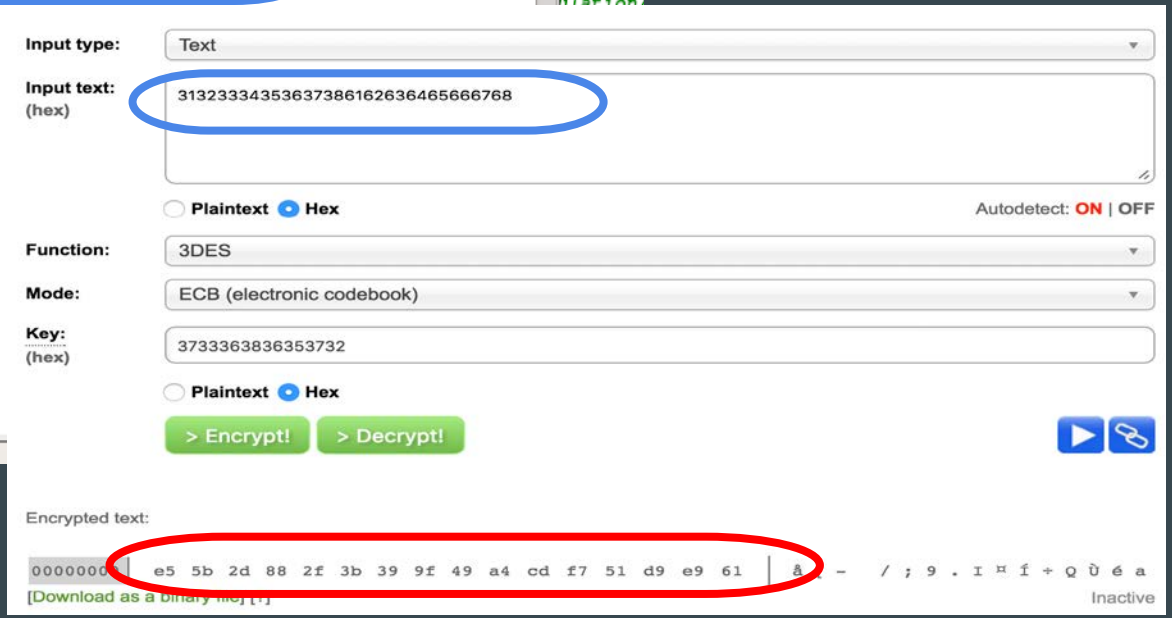


data = 64'hf68c7aeb67375ae8;
key1 = 64'h133457799BBCDFF1;
key2 = 64'h133457799BBCDFF1;
key3 = 64'h133457799BBCDFF1;
Cypher = 64'h85e813540f0ab405;

Results – Key Design Specific Success Criteria Results

2. The result of FPGA shows that the design is capable to implement triple DES encryption. (2 pts)

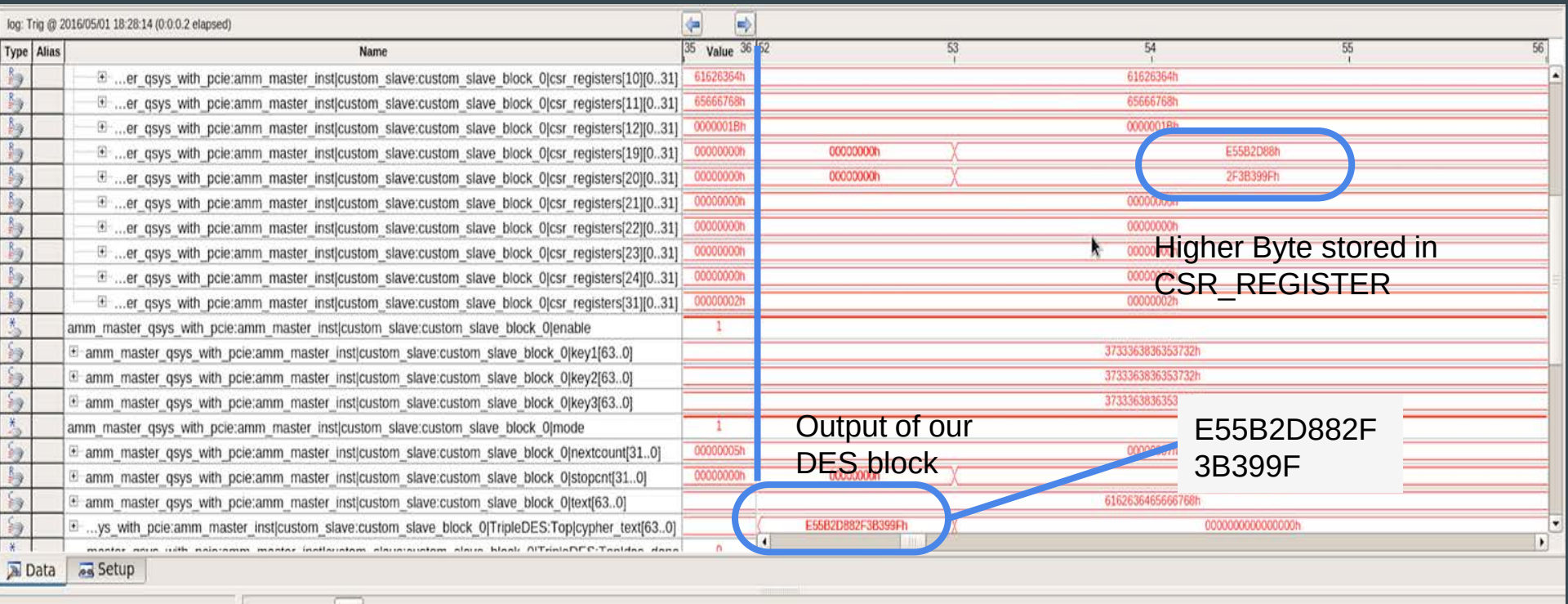
```
The value at register 0 is: 00000000
size=16
The plain text written in csr_register in hex is31323334353637386162636465666768
Plain text written.
Key input written.
The key written in csr_register in hex is 3733363836353732
Key input written.
The key written in csr_register in hex is 3733363836353732
Key input written.
The key written in csr_register in hex is 3733363836353732
Selection of encryption or decryption written.
Start byte written.
Stop byte received, demo finished
Process done!!!!!!
The result is E55B2D882F3B399F49A4CDF751D9E961
root@cedartrail:~/linux_app_sample#
```



http://tripledes.online-domain-tools.com/

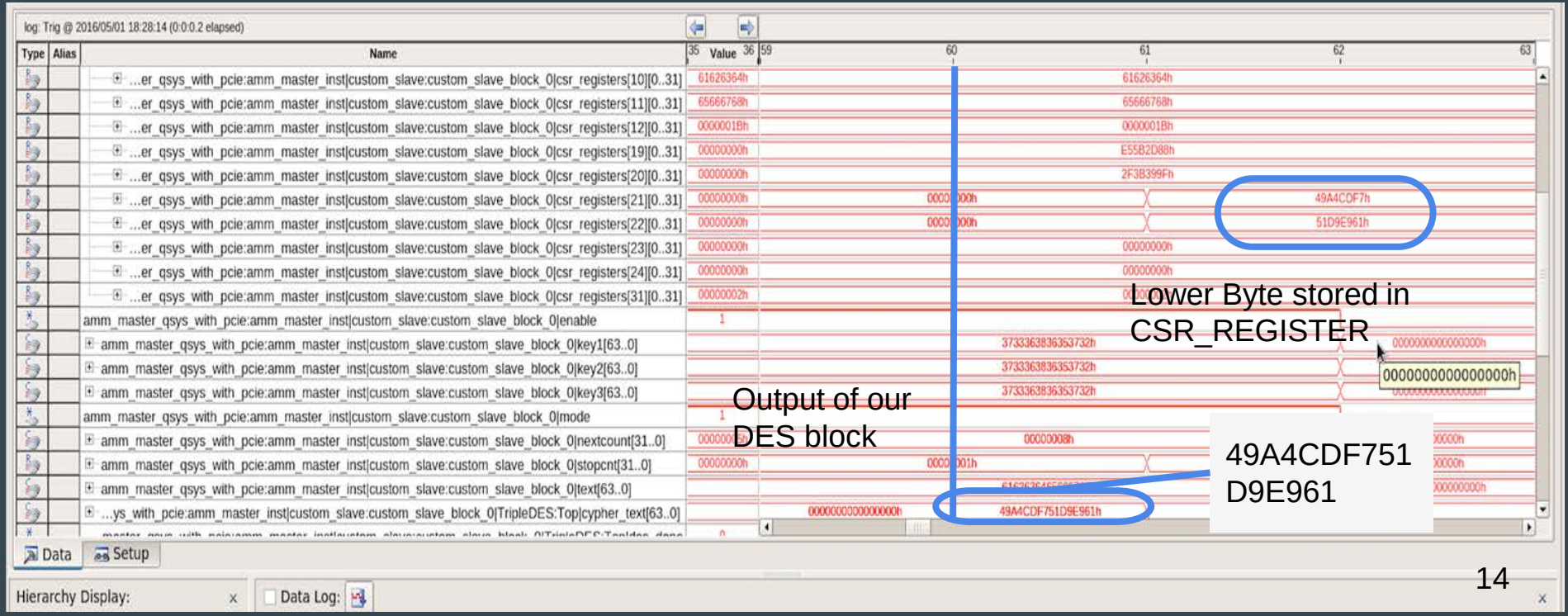
Results – Key Design Specific Success Criteria Results

Evidence that our FPGA works



Results – Key Design Specific Success Criteria Results

Evidence that our FPGA works



Results – Key Design Specific Success Criteria Results

3. The result of FPGA shows that the design is capable to implement triple DES decryption. (2 pts)

```
The value at register 0 is: 00000012
size=16
The plain text written in csr_register in hex is31323334353637386162636465666768
Plain text written.
Key input written.
The key written in csr_register in hex is 37333638
Key input written.
The key written in csr_register in hex is 37333638
Key input written.
The key written in csr_register in hex is 37333638
Selection of encryption or decryption written.
Start byte written.
Stop byte received, demo finished
Process done!!!!!!
The result is 4A9A31B5B95CC73F029D6BCBD7C837C5
root@cedartrail: /linux_app_sample# cd ..
```

Input type: Text

Input text: (hex) 31323334353637386162636465666768

Plaintext Hex Autodetect: ON | OFF

Function: 3DES

Mode: ECB (electronic codebook)

Key: (hex) 3733363836353732

Plaintext Hex

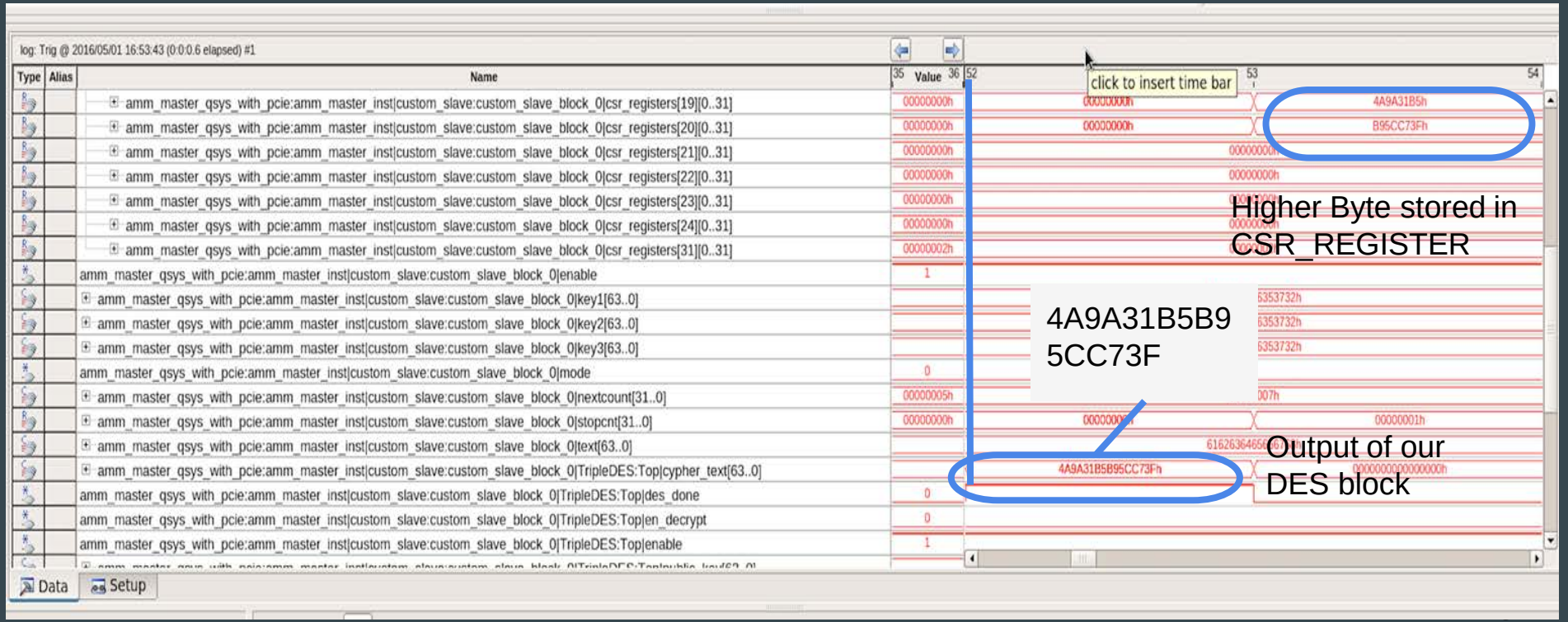
> Encrypt! > Decrypt!

Decrypted text:

00000000 4a 9a 31 b5 b9 5c c7 3f 02 9d 6b cb d7 c8 37 c5 J . 1 µ ¹ \ Ç ? . k Ë × È 7 Å

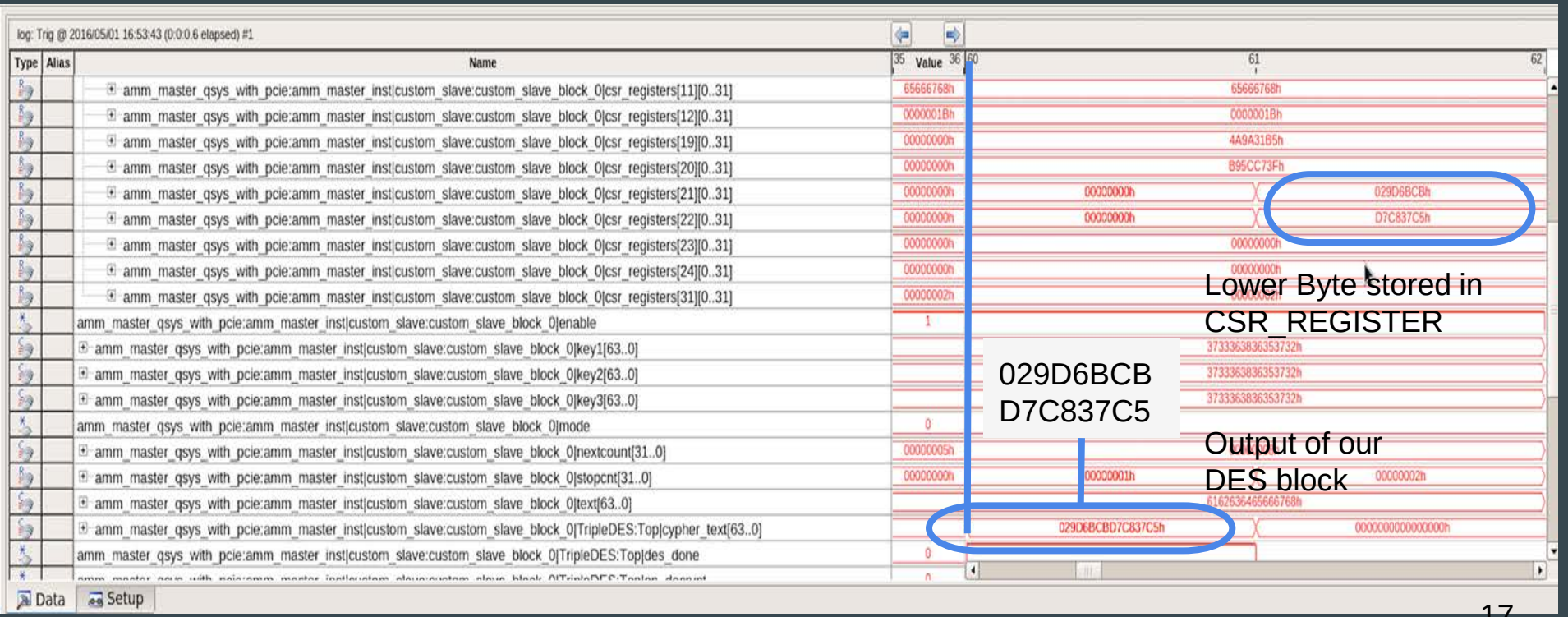
Results – Key Design Specific Success Criteria Results

Evidence that our FPGA works



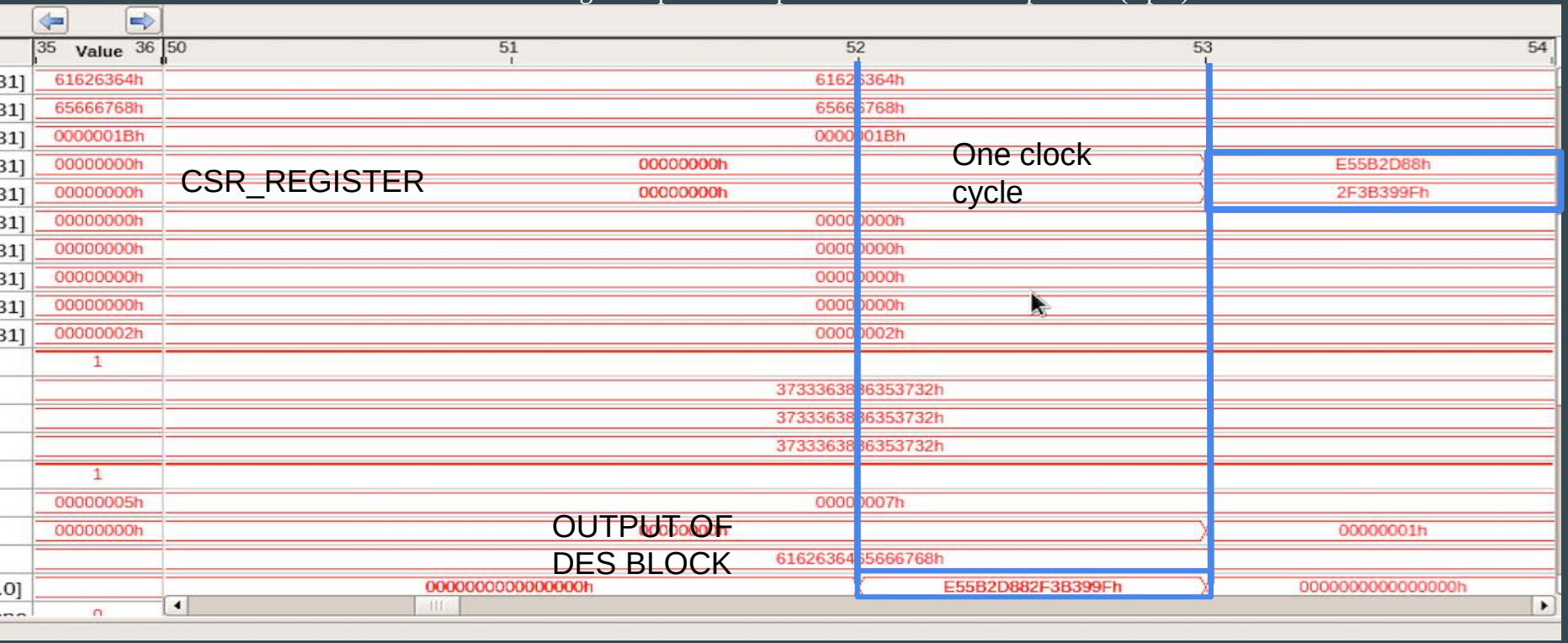
Results – Key Design Specific Success Criteria Results

Evidence that our FPGA works



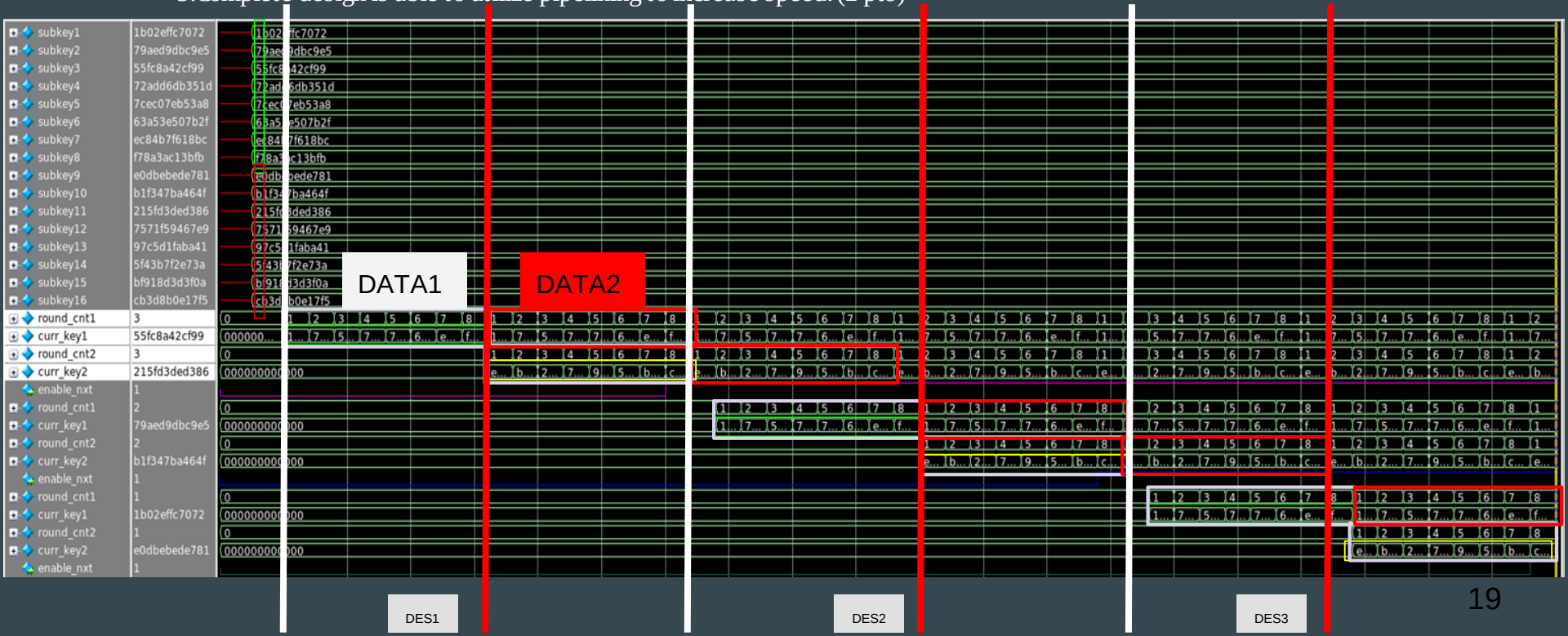
Results – Key Design Specific Success Criteria Results

4. The result of FPGA shows that the design is capable to implement an Avalon bus protocol.(2 pts)



Results – Key Design Specific Success Criteria Results

5.Complete design is able to utilize pipelining to increase speed. (2 pts)



Results – Layout

Quartus report

Slow 1200mV 85C Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	59.1 MHz	59.1 MHz	altera_reserved_tck	
2	61.19 MHz	61.19 MHz	clock_50_1	
3	94.85 MHz	94.85 MHz	amm_master_inst|pcie_ip|pcie_interna...i.cycloneiv_hssi_pcie_hip|corec|kout	

Flow Summary

Flow Status	Successful - Sun May 1 16:35:00 2016
Quartus II 64-Bit Version	14.0.0 Build 200 06/17/2014 SJ Full Version
Revision Name	master_example
Top-level Entity Name	master_example
Family	Cyclone IV GX
Device	EP4CGX150DF31C7
Timing Models	Final
Total logic elements	36,303 / 149,760 (24 %)
Total combinational functions	19,478 / 149,760 (13 %)
Dedicated logic registers	27,765 / 149,760 (19 %)
Total registers	27883
Total pins	171 / 508 (34 %)
Total virtual pins	0
Total memory bits	1,706,728 / 6,635,520 (26 %)
Embedded Multiplier 9-bit elements	0 / 720 (0 %)
Total GXB Receiver Channel PCS	1 / 8 (13 %)
Total GXB Receiver Channel PMA	1 / 8 (13 %)
Total GXB Transmitter Channel PCS	1 / 8 (13 %)
Total GXB Transmitter Channel PMA	1 / 8 (13 %)
Total PLLs	2 / 8 (25 %)

Fitter Resource Usage Summary

	Resource	Usage
1	Total logic elements	36,264 / 149,760 (24 %)
1	-- Combinational with no register	8498
2	-- Register only	16780
3	-- Combinational with a register	10986
2		
3	Logic element usage by number of LUT inputs	
1	-- 4 input functions	12625
2	-- 3 input functions	4630
3	-- <=2 input functions	2229
4	-- Register only	16780
4		
5	Logic elements by mode	
1	-- normal mode	18097
2	-- arithmetic mode	1387
6		
7	Total registers*	27,884 / 152,165 (18 %)
1	-- Dedicated logic registers	27,766 / 149,760 (19 %)
2	-- I/O registers	118 / 2,405 (5 %)
8		
9	Total LABs: partially or completely used	3,090 / 9,360 (33 %)
10	Virtual pins	0
11	I/O pins	171 / 508 (34 %)
1	-- Clock pins	2 / 10 (20 %)

Conclusion

- The most challenge part of our design is to get familiar with Quartus and using Signaltap to debug while doing FPGA part. It also takes a while to think out how Atom communicate with FPGA through software. For the hardware part, pipeline takes some time to debug.
- Utilize Avalon bus master controller and put all the input data in the SDRAM instead of CSR register, so that there's no input data size limit can process more data input.
- Have more complicated pipeline structure, send input data every four clock cycles, instead of eight.

Reference

[1] Understanding Static RAM Operation, Application notes by IBM, http://www.engr.uconn.edu/~omer.khan/courses/ece4401_f12/sramop.pdf

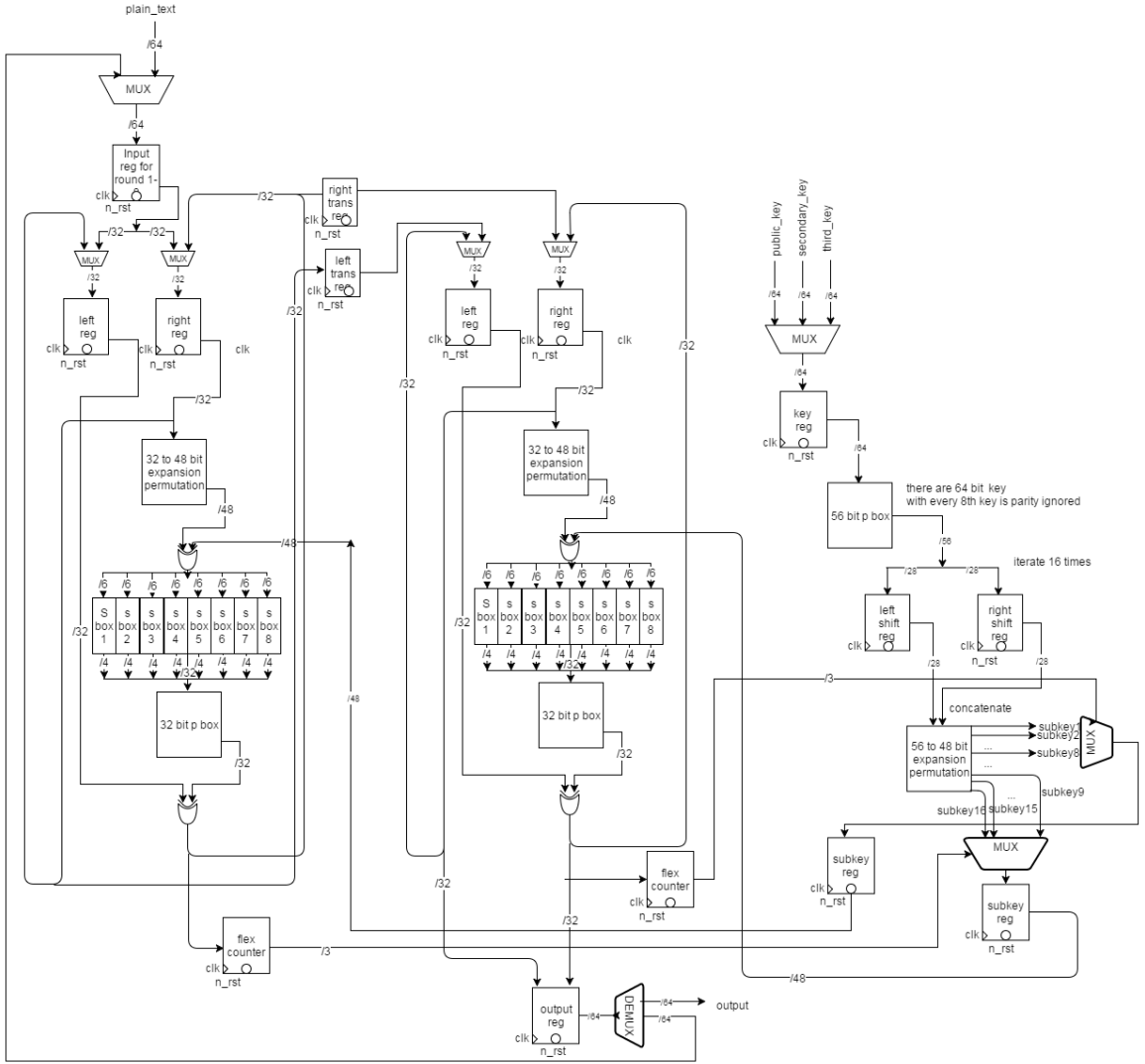
[2] Beuchat, R. (2011). Avalon Interconnect and SOPC Component. Chu/Embedded Embedded SoPC Design with Nios II Processor and VHDL Examples, 305-339.

http://page.math.tu-berlin.de/~kant/teaching/hess/krypto-ws2006/des.htm

http://tripledes.online-domain-tools.com/

Question?

Reference Diagram



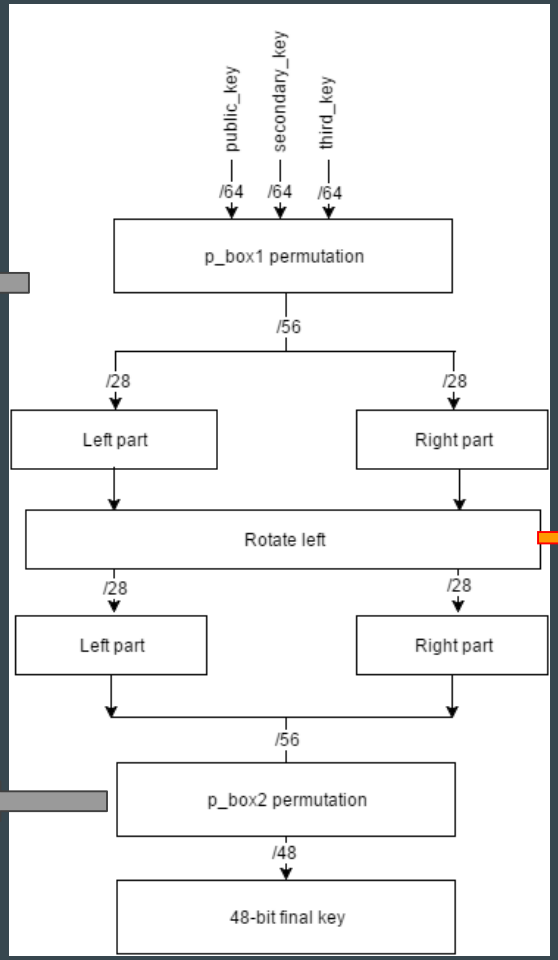
3DES RTL

Reference Diagram

P-box1 Permutation table						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

P-box2 Permutation table							
13	16	10	23	0	4	2	27
14	5	20	9	22	18	11	3
25	7	15	6	26	19	12	1
40	51	30	36	46	54	29	39
50	44	32	47	43	48	38	55
33	52	45	41	49	35	28	31

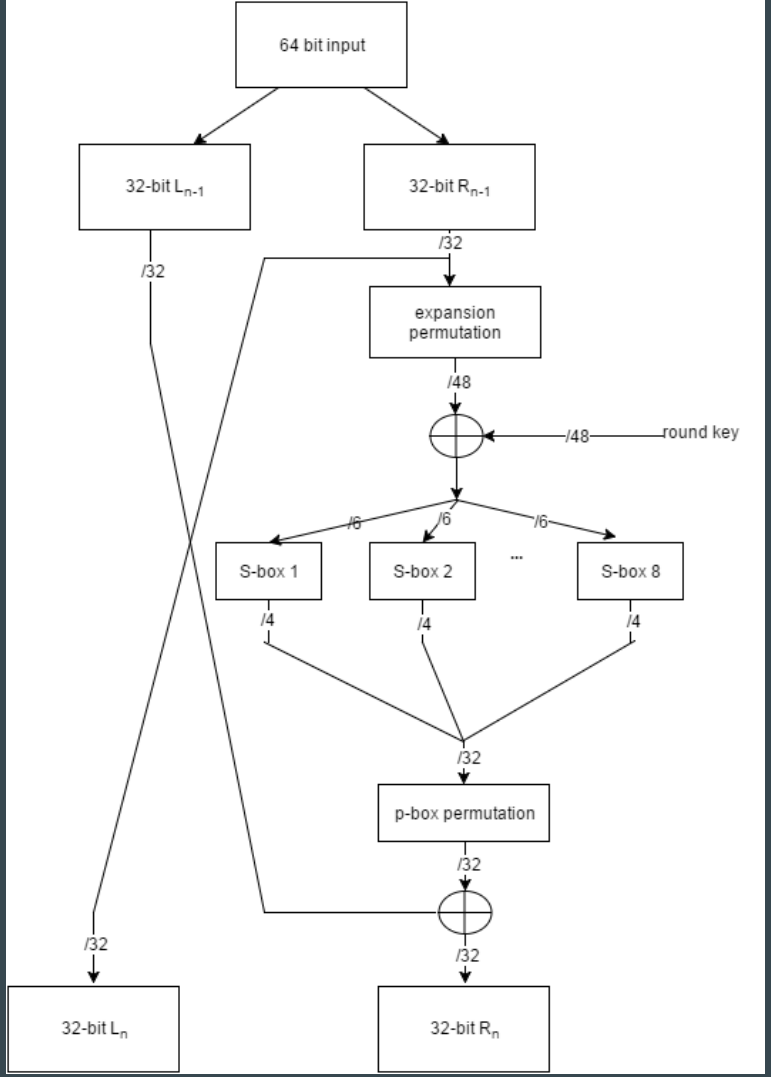
16 round subkey generator



Keys are 64-bit numbers & Every 8th bit is unused, So 8, 16, 24, 32, 40, 48, 56, 64 bit are ignored, and p_box1
Only permute the rest bits.

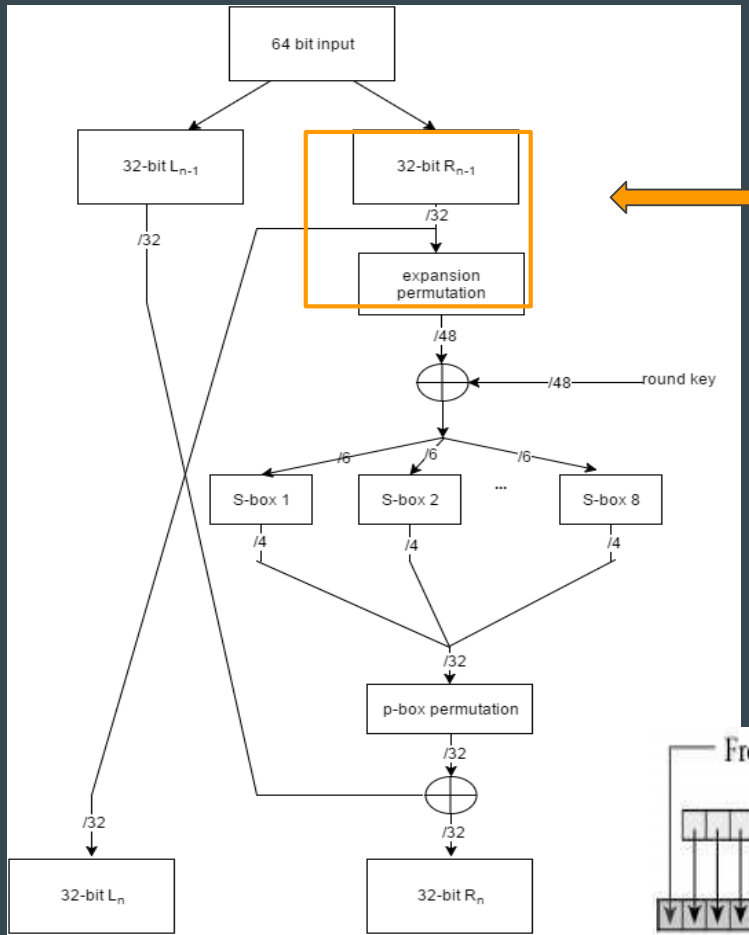
Iteration #	# of shifts
1	1
2	2
3	4
4	6
5	8
6	10
7	12
8	14
9	15
10	17
11	19
12	21
13	23
14	25
15	28
16	0

Reference Diagram



Single DES Iteration Round

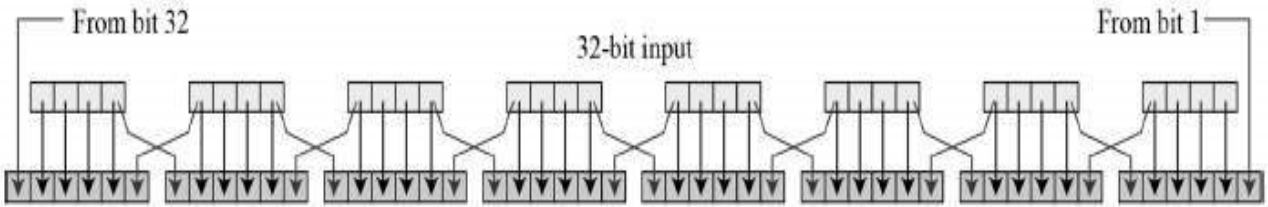
Reference Diagram



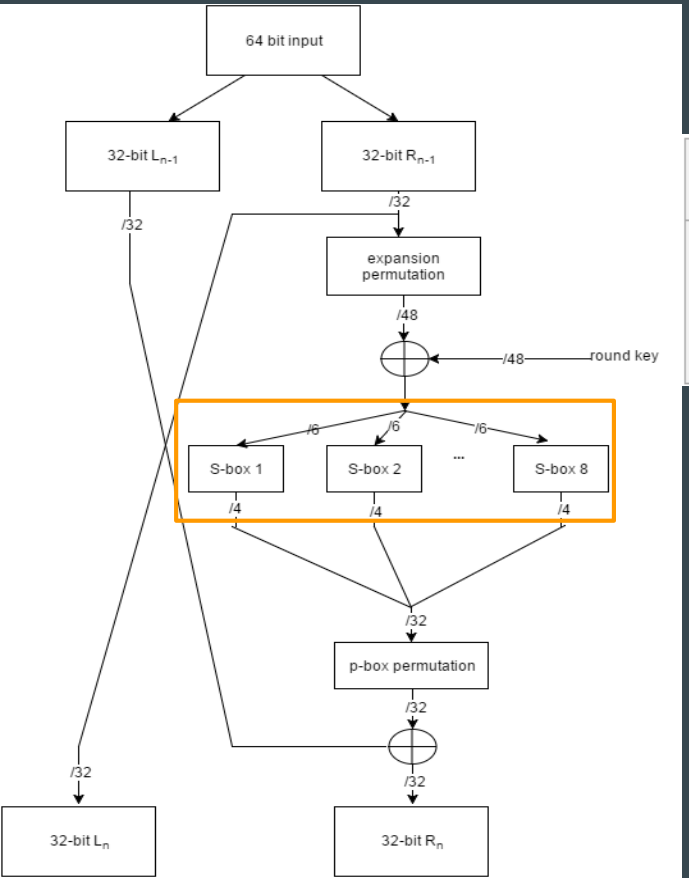
Expansion Permutation

Expansion Permutation table					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

- Expansion permutation expands 32 bit to 48 bit by duplicate some bits.
- e.g. the fifth bit of the input is duplicated in both the sixth and eighth bit of the output



Reference Diagram



Substitution Box (S-box)

S_5		Middle 4 bits of input															
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Outer bits	00	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

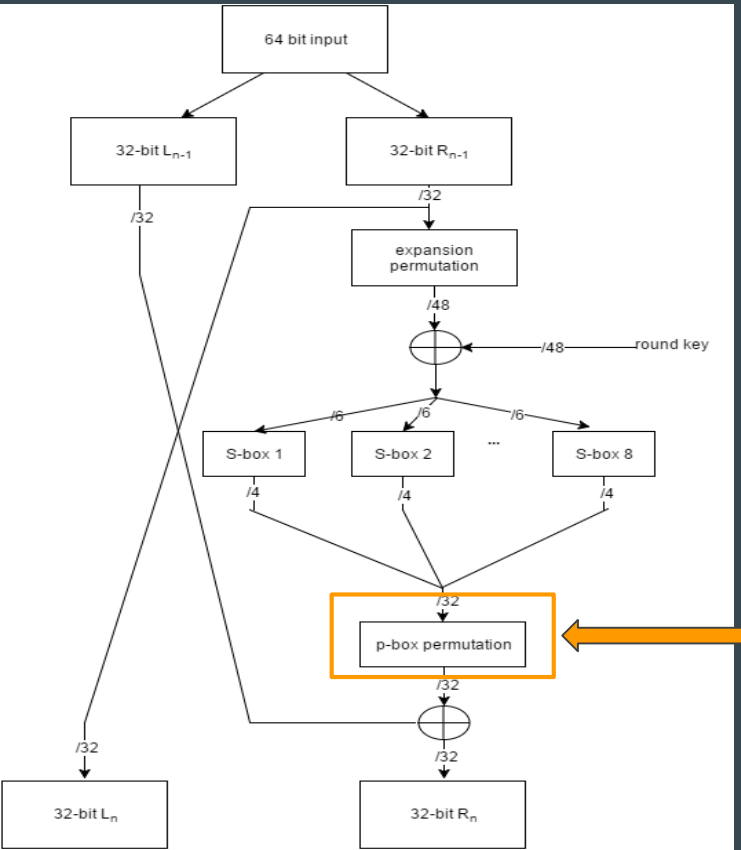
- Each S-box replaces a 6-bit input with a 4-bit output.
- Given a 6-bit input, the 4-bit output is found by selecting the row using the outer two bits, and the column using the inner four bits.

Reference Diagram

S_1	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0yyyy1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
1yyyy0	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
1yyyy1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
0yyyy1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
1yyyy0	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
1yyyy1	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
0yyyy1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
1yyyy0	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1yyyy1	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
0yyyy1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
1yyyy0	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
1yyyy1	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
0yyyy1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
1yyyy0	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
1yyyy1	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
0yyyy1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
1yyyy0	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
1yyyy1	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
0yyyy1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1yyyy0	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
1yyyy1	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
0yyyy1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
1yyyy0	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8

Substitution Box (S-box)

Reference Diagram



P box permutation

P-box permutation							
15	6	19	20	28	11	27	16
0	14	22	25	4	17	30	9
1	7	23	13	31	26	2	8
18	12	29	5	21	10	3	24