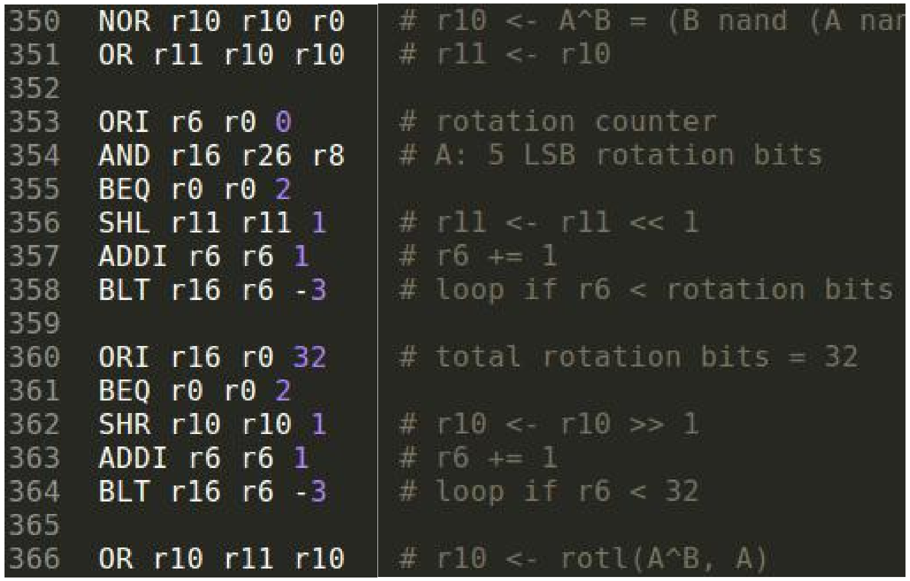
RC5 Assembly to Binary



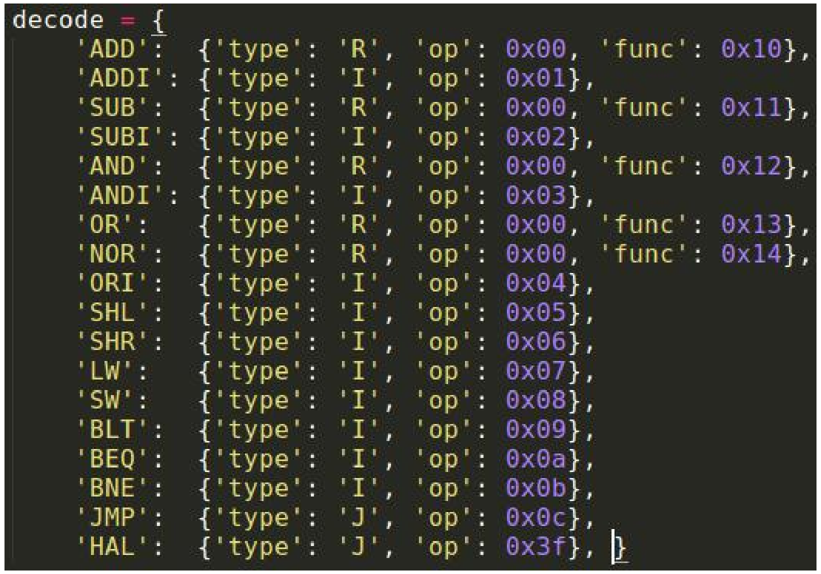
```
NOR r10 r10 r0      # r10 <- A^B = (B nand (A nar
OR r11 r10 r10      # r11 <- r10

ORI r6 r0 0         # rotation counter
AND r16 r26 r8      # A: 5 LSB rotation bits
BEQ r0 r0 2
SHL r11 r11 1       # r11 <- r11 << 1
ADDI r6 r6 1        # r6 += 1
BLT r16 r6 -3       # loop if r6 < rotation bits

ORI r16 r0 32       # total rotation bits = 32
BEQ r0 r0 2
SHR r10 r10 1       # r10 <- r10 >> 1
ADDI r6 r6 1        # r6 += 1
BLT r16 r6 -3       # loop if r6 < 32

OR r10 r11 r10      # r10 <- rotl(A^B, A)
```

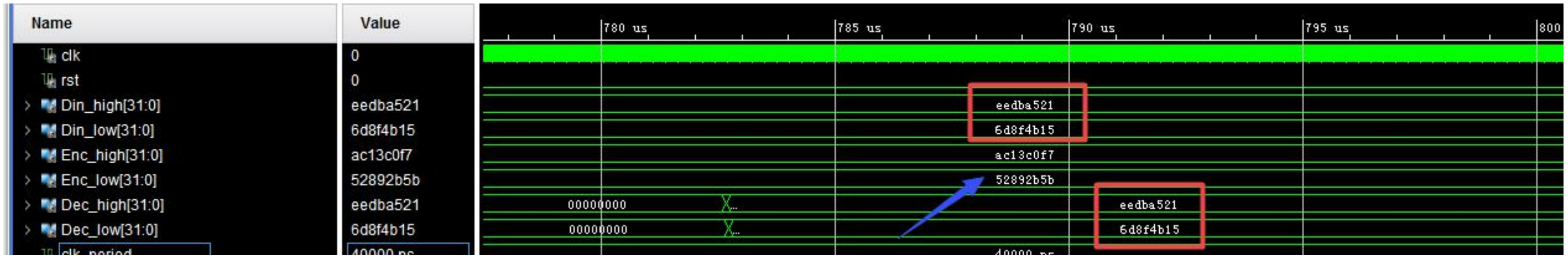
Part of RC5 assembly
(rotate operation)



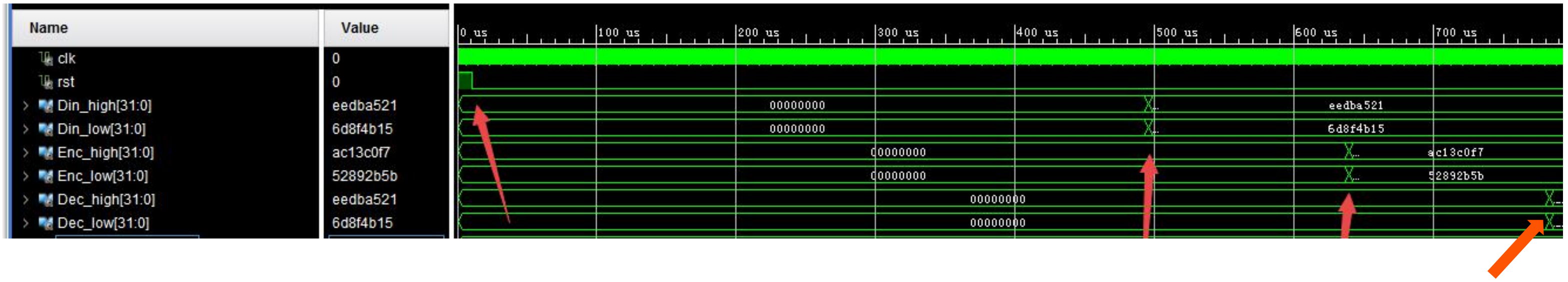
```
decode = {
    'ADD':  {'type': 'R', 'op': 0x00, 'func': 0x10},
    'ADDI': {'type': 'I', 'op': 0x01},
    'SUB':  {'type': 'R', 'op': 0x00, 'func': 0x11},
    'SUBI': {'type': 'I', 'op': 0x02},
    'AND':  {'type': 'R', 'op': 0x00, 'func': 0x12},
    'ANDI': {'type': 'I', 'op': 0x03},
    'OR':   {'type': 'R', 'op': 0x00, 'func': 0x13},
    'NOR':  {'type': 'R', 'op': 0x00, 'func': 0x14},
    'ORI':  {'type': 'I', 'op': 0x04},
    'SHL':  {'type': 'I', 'op': 0x05},
    'SHR':  {'type': 'I', 'op': 0x06},
    'LW':   {'type': 'I', 'op': 0x07},
    'SW':   {'type': 'I', 'op': 0x08},
    'BLT':  {'type': 'I', 'op': 0x09},
    'BEQ':  {'type': 'I', 'op': 0x0a},
    'BNE':  {'type': 'I', 'op': 0x0b},
    'JMP':  {'type': 'J', 'op': 0x0c},
    'HAL':  {'type': 'J', 'op': 0x3f}, }
```

Decoding table

RC5 Timing



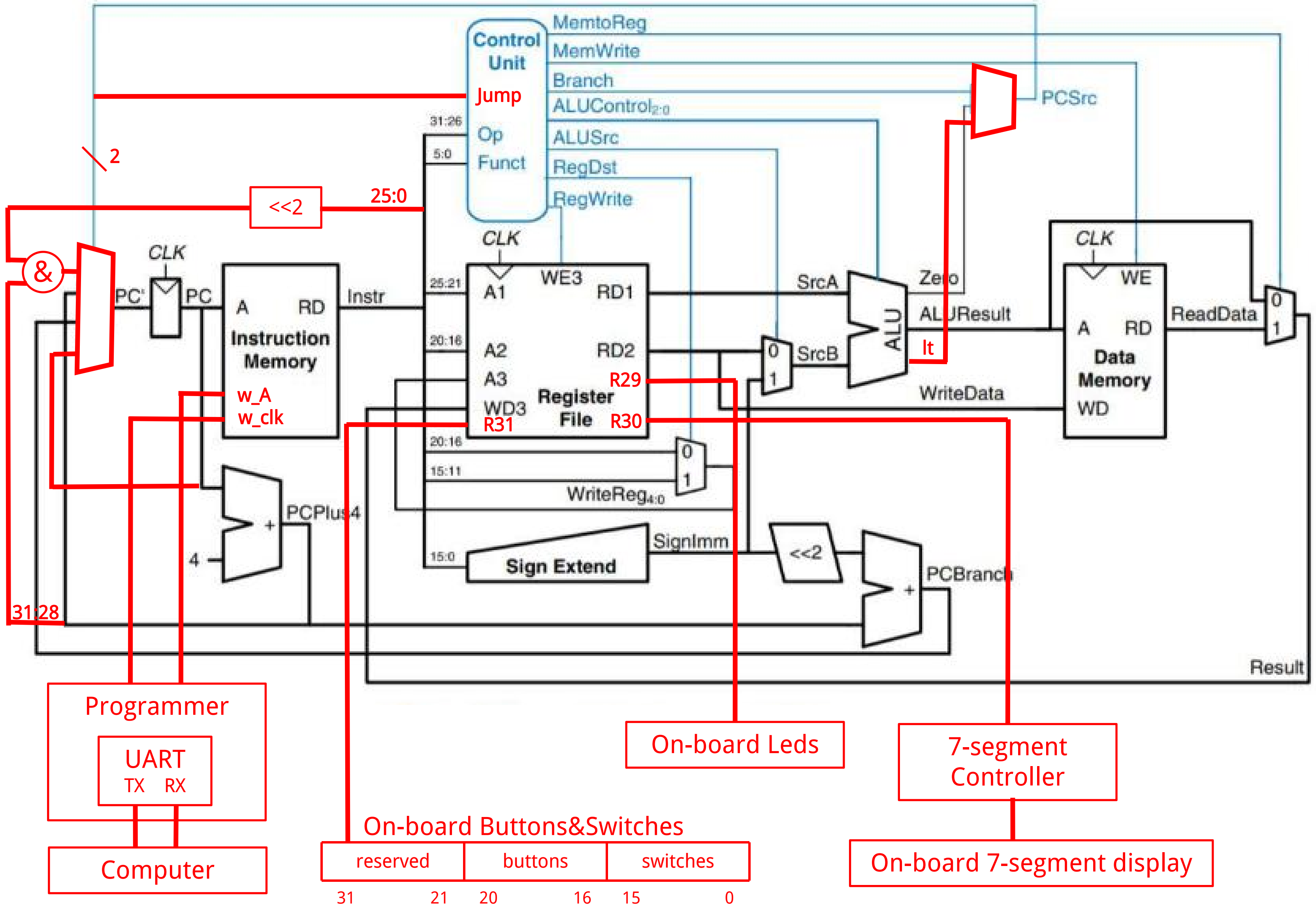
Name	Value
clk	0
rst	0
Din_high[31:0]	eedba521
Din_low[31:0]	6d8f4b15
Enc_high[31:0]	ac13c0f7
Enc_low[31:0]	52892b5b
Dec_high[31:0]	eedba521
Dec_low[31:0]	6d8f4b15



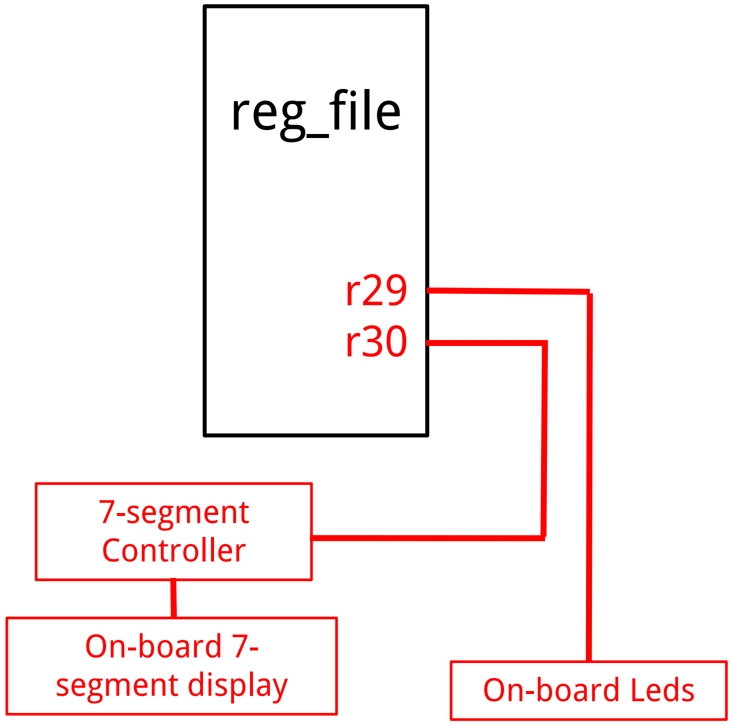
Name	Value
clk	0
rst	0
Din_high[31:0]	eedba521
Din_low[31:0]	6d8f4b15
Enc_high[31:0]	ac13c0f7
Enc_low[31:0]	52892b5b
Dec_high[31:0]	eedba521
Dec_low[31:0]	6d8f4b15

Critial path delay: **38 ns**
Clock frequency: **25 MHz**
Latency (cycles used in key-gen, encryption and decryption):

	Start	End	Time-Diff	Latency	clk period
Key-Gen	10460	495860	485400	12135	40
Encryption	495860	639260	143400	3585	
Decryption	639260	782700	143440	3586	


Control Unit
MemtoReg
MemWrite
Branch
ALUControl2:0
ALUSrc
RegDst
RegWrite
Jump
Op
Funct
31:26
5:0
PCSrc
2
<<2
25:0
&
CLK
PC'
PC
A
RD
Instruction Memory
w_A
w_clk
Instr
25:21
20:16
20:16
15:11
15:0
A1
A2
A3
WD3
WE3
RD1
RD2
R29
R30
R31
Register File
SrcA
SrcB
Zero
ALUResult
lt
ALU
WriteData
WriteReg4:0
Sign Extend
SignImm
PCPlus4
4
PCBranch
31:28
WE
A
RD
Data Memory
WD
ReadData
Result
Programmer
UART
TX
RX
Computer
On-board Leds
7-segment Controller
On-board 7-segment display
On-board Buttons&Switches
reserved
buttons
switches
31
21
20
16
15
0

Leds and 7-segment display



Example code:

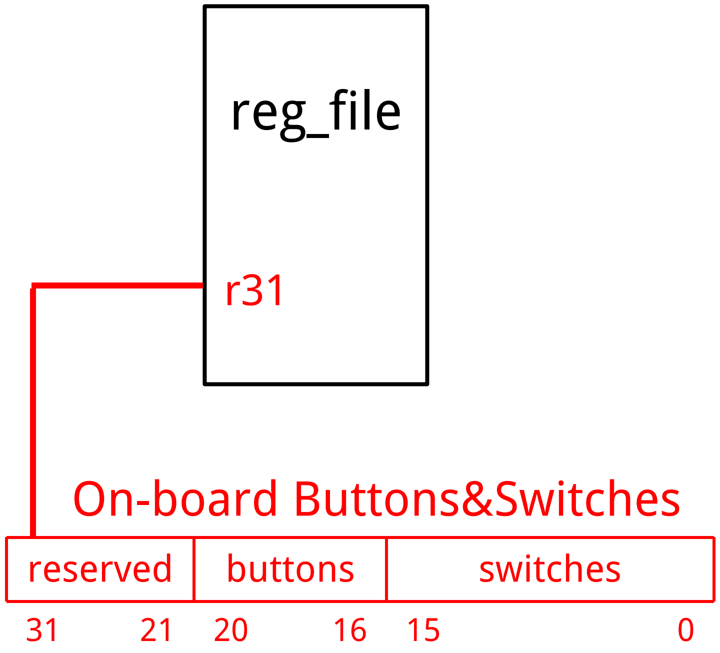
```
# Turn on led(7)
# r0 = 0, r1 = 1
SHL r28, r1, 7      # r28(7) = 1
OR  r29, r29, r28

# display data[30]
LW  r30 r0 30
```

VHD pseudocode:

```
led <= r29(15 downto 0);
U_7seg_ctrl(r30, seg, an);
```

Buttons & Switches



VHD pseudocode:

```
r31(20 downto 0) <= btn&sw;
```

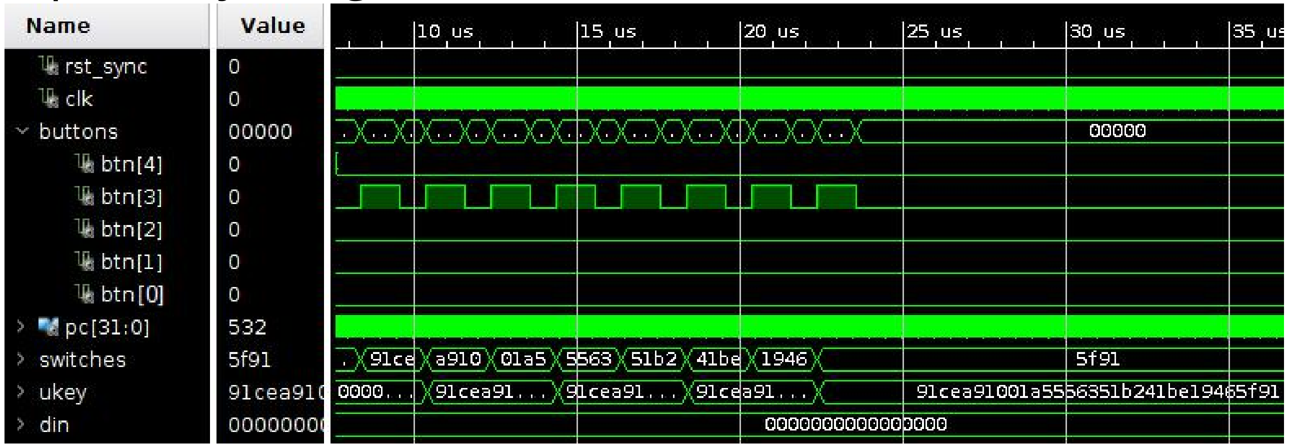
Example code:

```
# Wait for btn0
# r22 = 1<<16 for picking up r31(16)
LOOP:
AND r28, r22, r31   # pick r31(16)
BNE r28, r22, LOOP
# Do something here...
```

- If btn0 is not pressed, r31 = '0'. BNE becomes True. The code will keep looping.
- If btn0 is pressed, r31 = '1'. BNE becomes False. The code will continue.

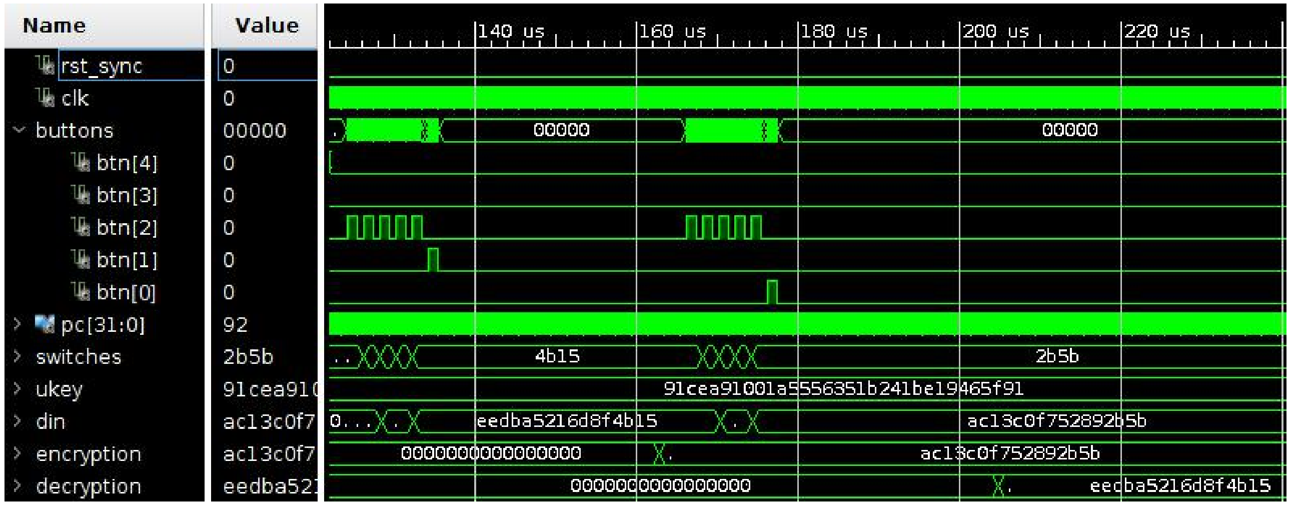
Buttons & Switches

Input Ukey using btn3:



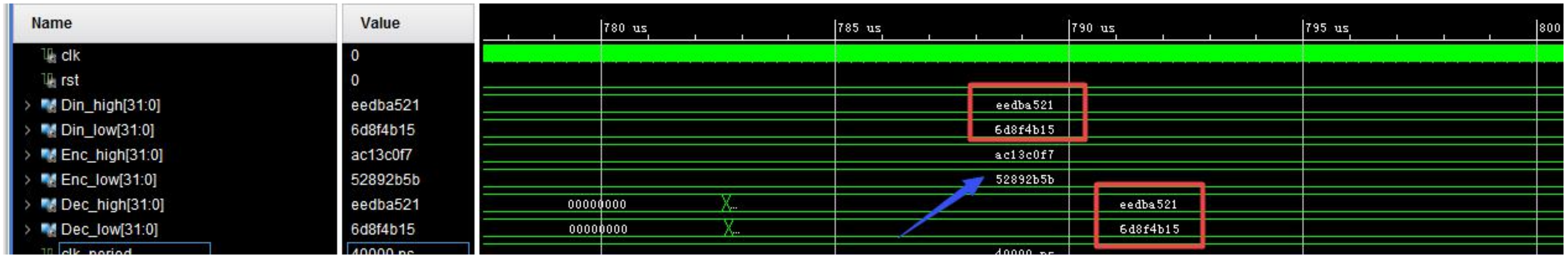
Name	Value
rst_sync	0
clk	0
buttons	00000
btn[4]	0
btn[3]	0
btn[2]	0
btn[1]	0
btn[0]	0
pc[31:0]	532
switches	5f91
ukey	91cea91(
din	0000000(

Input Din (btn3); start encryption/decryption (btn1/btn0)

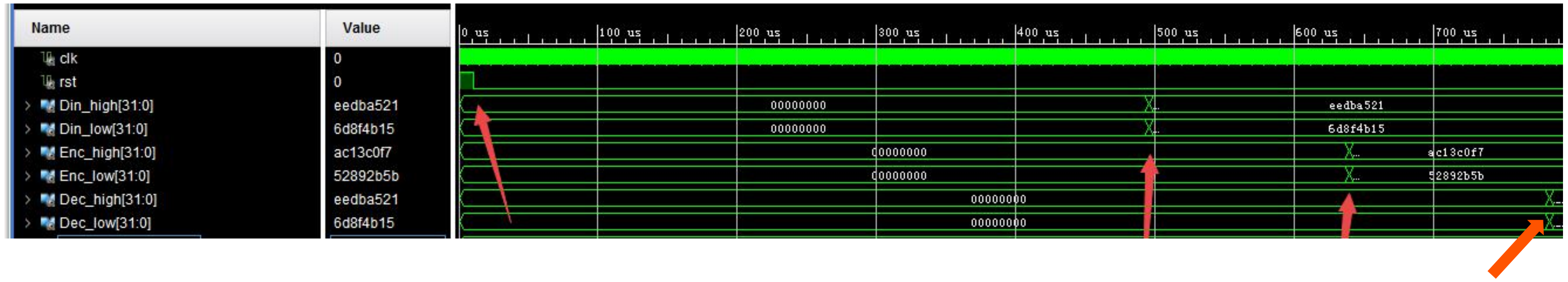


Name	Value
rst_sync	0
clk	0
buttons	00000
btn[4]	0
btn[3]	0
btn[2]	0
btn[1]	0
btn[0]	0
pc[31:0]	92
switches	2b5b
ukey	91cea91(
din	ac13c0f7
encryption	ac13c0f7
decryption	eedba52!

RC5 Timing



Name	Value
clk	0
rst	0
Din_high[31:0]	eedba521
Din_low[31:0]	6d8f4b15
Enc_high[31:0]	ac13c0f7
Enc_low[31:0]	52892b5b
Dec_high[31:0]	eedba521
Dec_low[31:0]	6d8f4b15



Name	Value
clk	0
rst	0
Din_high[31:0]	eedba521
Din_low[31:0]	6d8f4b15
Enc_high[31:0]	ac13c0f7
Enc_low[31:0]	52892b5b
Dec_high[31:0]	eedba521
Dec_low[31:0]	6d8f4b15

Critial path delay: **38 ns**
Clock frequency: **25 MHz**
Latency (cycles used in key-gen, encryption and decryption):

	Start	End	Time-Diff	Latency	clk period
Key-Gen	10460	495860	485400	12135	40
Encryption	495860	639260	143400	3585	
Decryption	639260	782700	143440	3586	