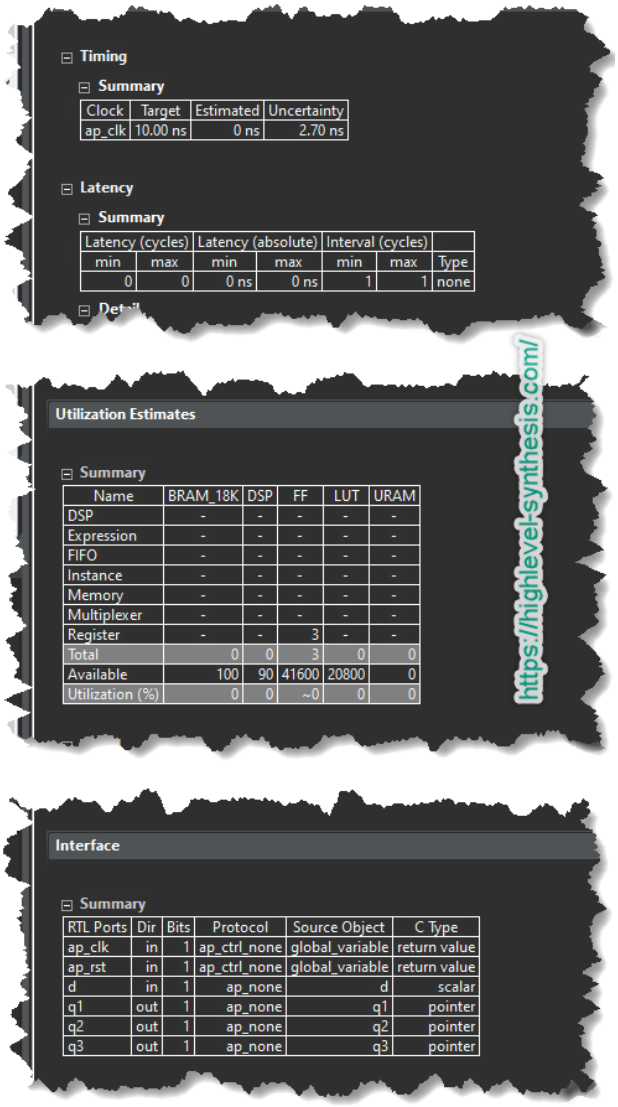
1- This is the design code in HLS. You can find the complete code in the Resources folder attached to this lecture.

```
#include "global_variable.h"

bool dff1 = 0;
bool dff2 = 0;
bool dff3 = 0;

void global_variable(bool d, bool &q1, bool &q2, bool &q3) {
#pragma HLS INTERFACE ap_none port=d
#pragma HLS INTERFACE ap_none port=q1
#pragma HLS INTERFACE ap_none port=q2
#pragma HLS INTERFACE ap_none port=q3
#pragma HLS INTERFACE ap_ctrl_none port=return

        dff3 = dff2;
        dff2 = dff1;
        dff1 = d;

        q1 = dff1;
        q2 = dff2;
        q3 = dff3;
}
```

This figure shows the simulation output

The following figure shows the HLS report after synthesis.



Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00 ns	0 ns	2.70 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
0	0	0 ns	0 ns	1	1	none

Utilization Estimates

Summary

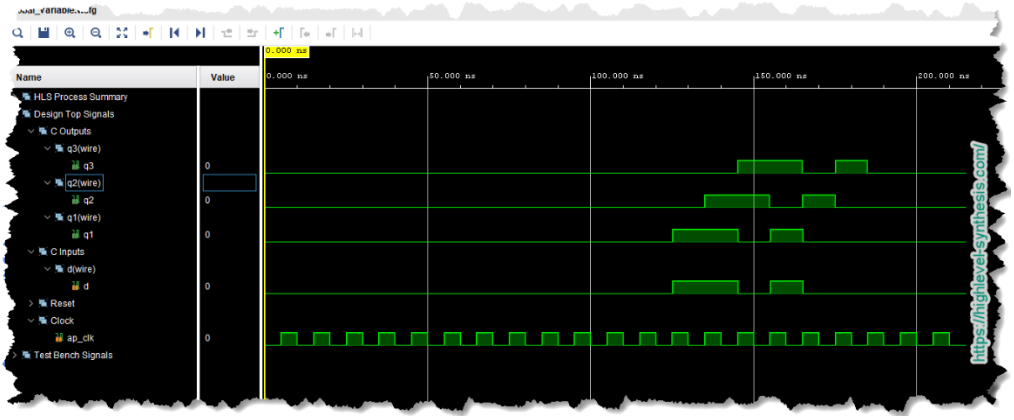
Name	BRAM_18K	DSP	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	-	-	-
FIFO	-	-	-	-	-
Instance	-	-	-	-	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	-	-
Register	-	-	3	-	-
Total	0	0	3	0	0
Available	100	90	41600	20800	0
Utilization (%)	0	0	~0	0	0

Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_none	global_variable	return value
ap_rst	in	1	ap_ctrl_none	global_variable	return value
d	in	1	ap_none	d	scalar
q1	out	1	ap_none	q1	pointer
q2	out	1	ap_none	q2	pointer
q3	out	1	ap_none	q3	pointer

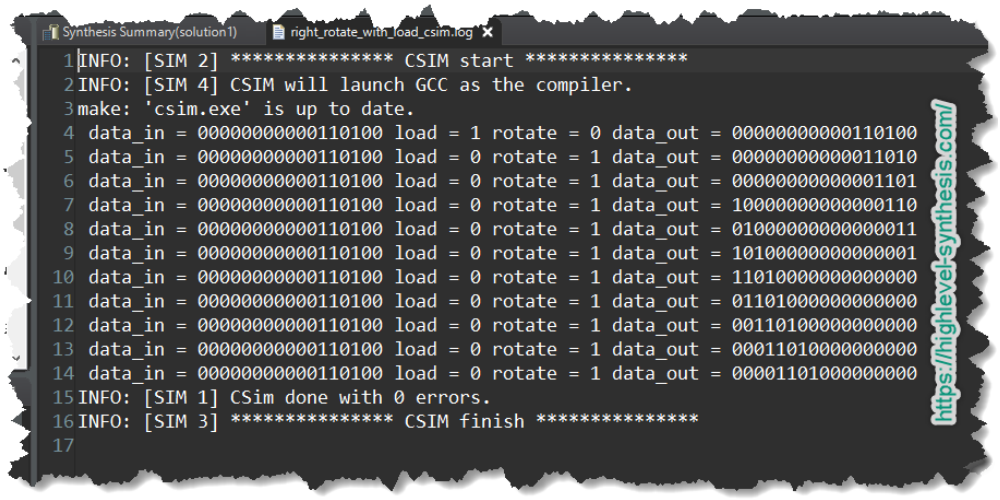
In addition, the following figure shows the signal waveform after RTL/C Co-simulation.



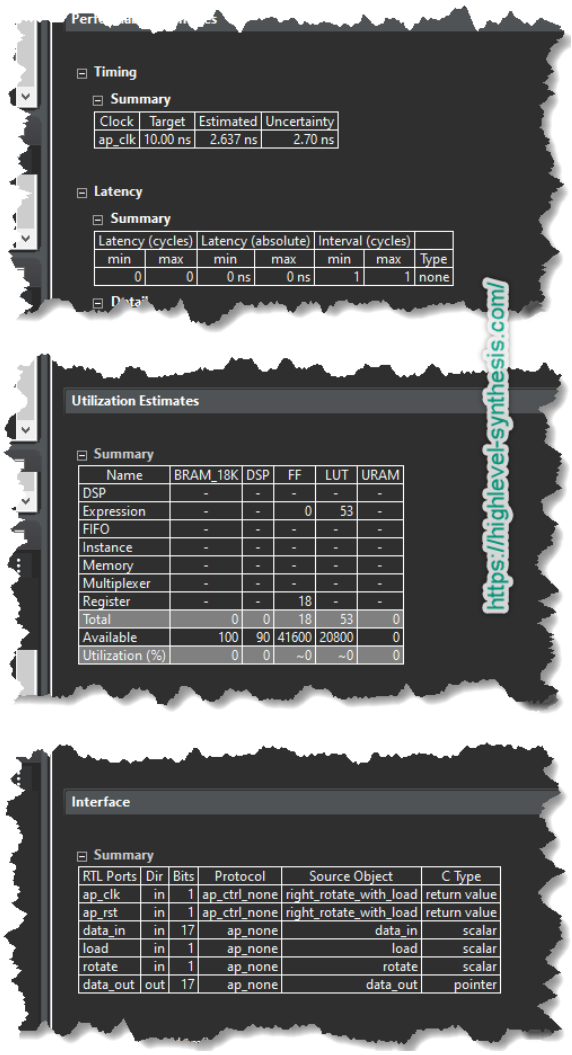
2- This is the design code in HLS. You can find the complete code in the Resources folder attached to this lecture.

```
#include "right_rotate_with_load.h"

void right_rotate_with_load(ap_uint<N> data_in, bool load, bool rotate, ap_uint<N> &data_out) {
#pragma HLS INTERFACE ap_none port=data_in
#pragma HLS INTERFACE ap_none port=load
#pragma HLS INTERFACE ap_none port=rotate
#pragma HLS INTERFACE ap_none port=data_out
#pragma HLS INTERFACE ap_ctrl_none port=return
        static ap_uint<N> rotate_register = 0;
        if (load == 1) {
                rotate_register = data_in;
        }

        if (rotate == 1) {
                rotate_register.rrotate(1);
        }

        data_out = rotate_register;
}
```

This figure shows the simulation output



The following figure shows the HLS report after synthesis.



Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00 ns	2.637 ns	2.70 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
0	0	0 ns	0 ns	1	1	none

Utilization Estimates

Summary

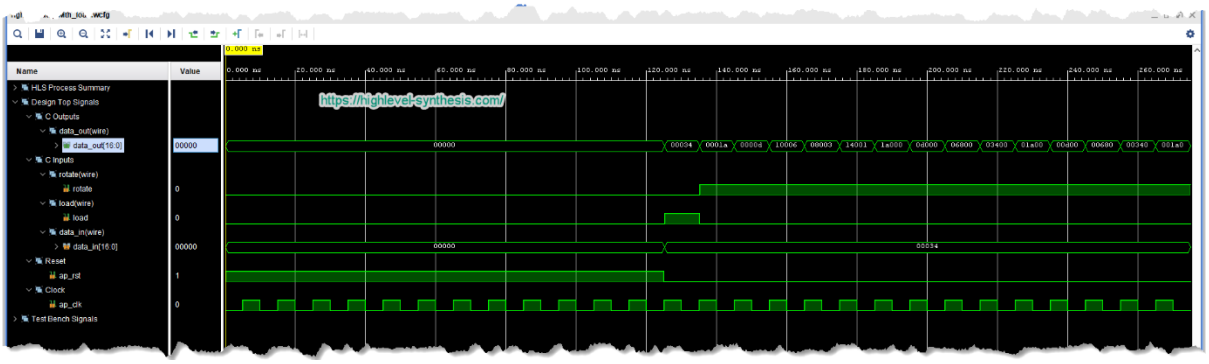
Name	BRAM_18K	DSP	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	0	53	-
FIFO	-	-	-	-	-
Instance	-	-	-	-	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	-	-
Register	-	-	18	-	-
Total	0	0	18	53	0
Available	100	90	41600	20800	0
Utilization (%)	0	0	~0	~0	0

Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_none	right_rotate_with_load	return value
ap_rst	in	1	ap_ctrl_none	right_rotate_with_load	return value
data_in	in	17	ap_none	data_in	scalar
load	in	1	ap_none	load	scalar
rotate	in	1	ap_none	rotate	scalar
data_out	out	17	ap_none	data_out	pointer

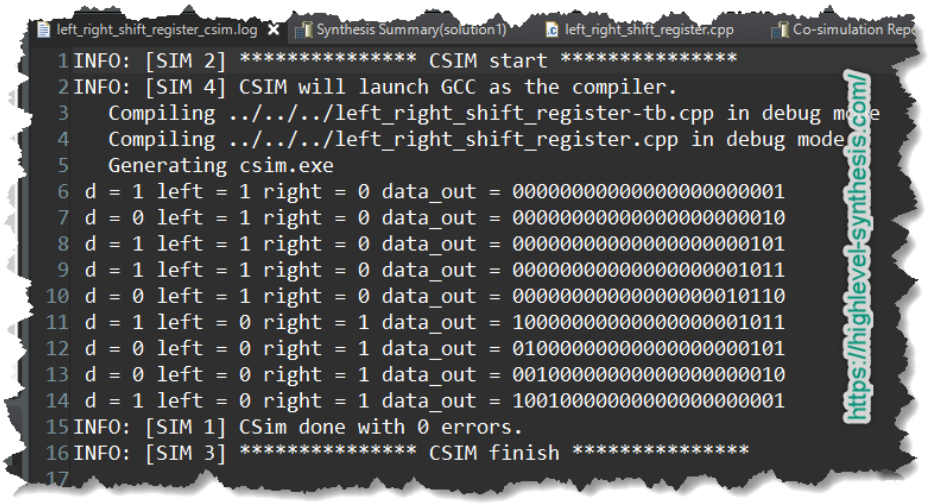
In addition, the following figure shows the signal waveform after RTL/C Co-simulation.



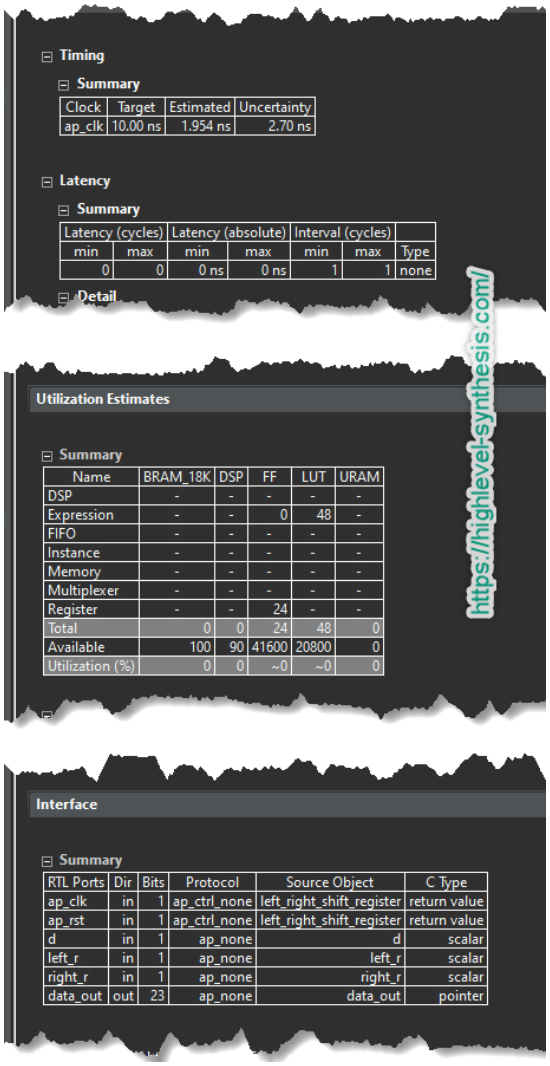
3- This is the design code in HLS. You can find the complete code in the Resources folder attached to this lecture.

```
#include "left_right_shift_register.h"


ap_uint<N> rl_shift_reg = 0;

void left_right_shift_register(bool d, bool left, bool right, ap_uint<N> &data_out) {
#pragma HLS INTERFACE ap_none port=d
#pragma HLS INTERFACE ap_none port=left
#pragma HLS INTERFACE ap_none port=right
#pragma HLS INTERFACE ap_none port=data_out
#pragma HLS INTERFACE ap_ctrl_none port=return

        if (left == 1) {
                rl_shift_reg = rl_shift_reg << 1;
                rl_shift_reg[0] = d;

        }

        if (right == 1) {
                rl_shift_reg = rl_shift_reg >> 1;
                rl_shift_reg[N-1] = d;
        }

        data_out = rl_shift_reg;

}
```

This figure shows the simulation output



The following figure shows the HLS report after synthesis.



Timing

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00 ns	1.954 ns	2.70 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
0	0	0 ns	0 ns	1	1	none

Detail

Utilization Estimates

Summary

Name	BRAM_18K	DSP	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	0	48	-
FIFO	-	-	-	-	-
Instance	-	-	-	-	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	-	-
Register	-	-	24	-	-
Total	0	0	24	48	0
Available	100	90	41600	20800	0
Utilization (%)	0	0	~0	~0	0

Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_none	left_right_shift_register	return value
ap_rst	in	1	ap_ctrl_none	left_right_shift_register	return value
d	in	1	ap_none	d	scalar
left_r	in	1	ap_none	left_r	scalar
right_r	in	1	ap_none	right_r	scalar
data_out	out	23	ap_none	data_out	pointer

In addition, the following figure shows the signal waveform after RTL/C Co-simulation.