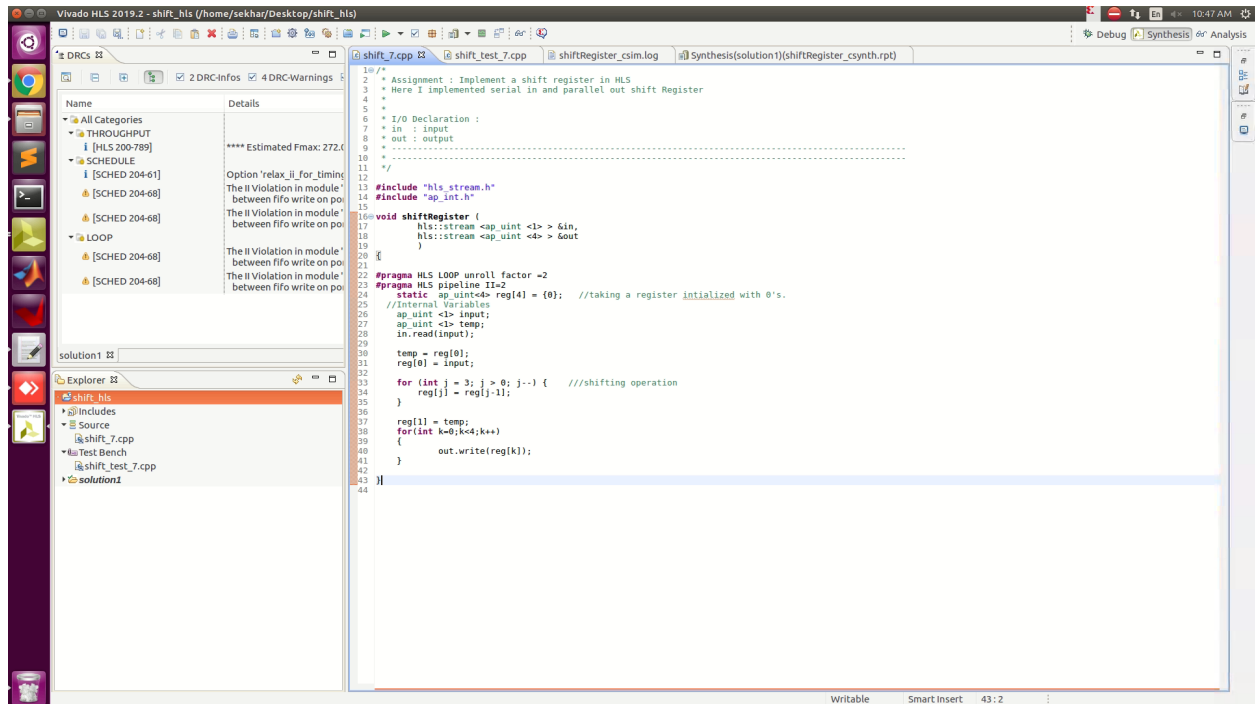


Assignment-7

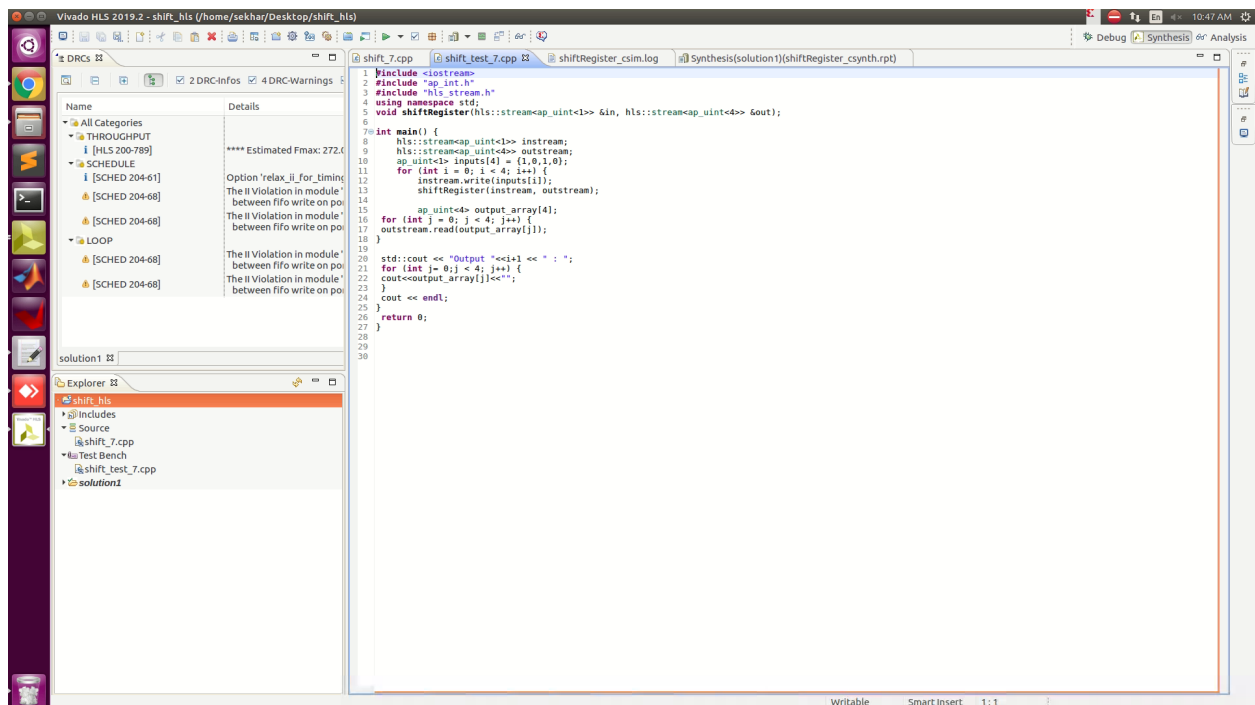
Implement a 10 bit shift register

Simulation :



The screenshot shows the Vivado IDE interface for the 'shift_hls' project. The left pane displays DRCs (Design Rule Checks) and the Explorer. The main editor shows the C++ code for 'shiftRegister.cpp'. The code implements a 10-bit shift register using a loop and a register.

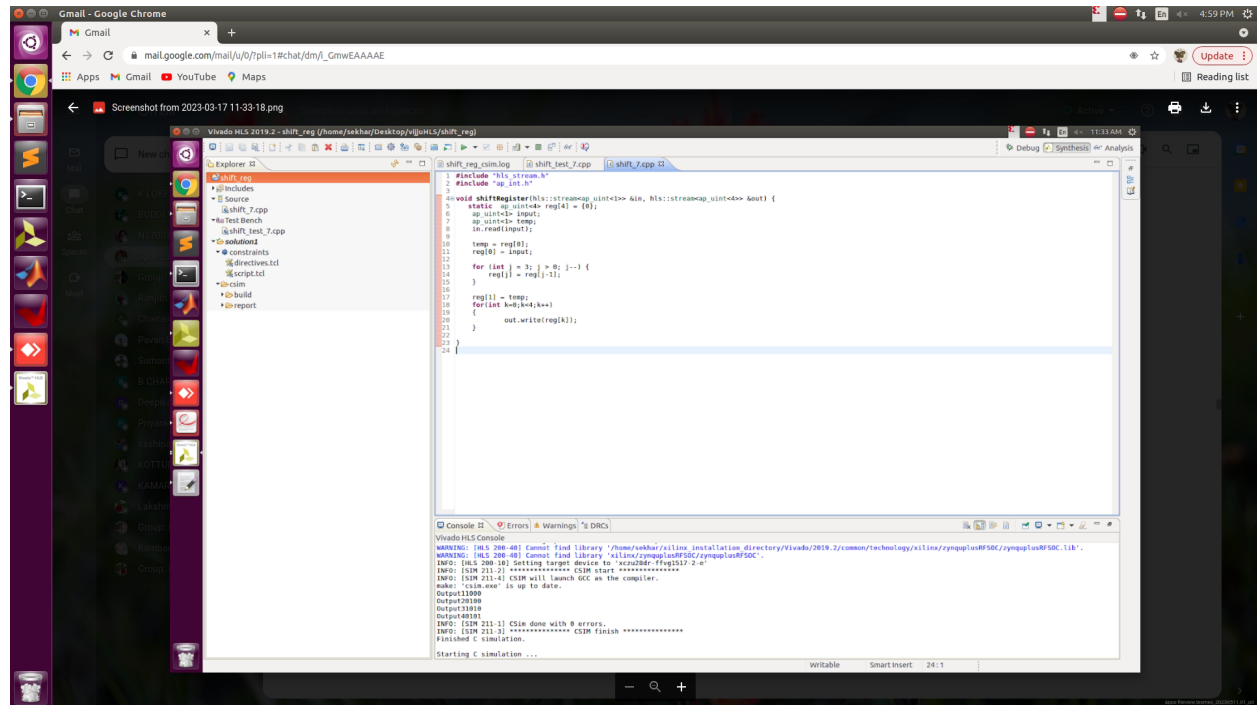
```
10 /*
11  * Assignment : Implement a shift register in HLS
12  * Here I implemented serial in and parallel out shift Register
13  *
14  * I/O Declaration :
15  * in : input
16  * out : output
17  *
18  *
19  */
20 #include "hls_stream.h"
21 #include "ap_int.h"
22
23 void shiftRegister (
24     hls::stream<ap_uint<1>> &in,
25     hls::stream<ap_uint<4>> &out)
26 {
27     #pragma HLS LOOP UNROLL FACTOR = 2
28     #pragma HLS PIPELINE II=2
29     static ap_uint<4> reg[4] = {0}; //taking a register initialized with 0's.
30     //Internal Variables
31     ap_uint<1> input;
32     ap_uint<1> temp;
33     in.read(input);
34     temp = reg[0];
35     reg[0] = input;
36     for (int j = 3; j > 0; j--) { //shifting operation
37         reg[j] = reg[j-1];
38     }
39     reg[1] = temp;
40     for (int k=0;k<4;k++)
41     {
42         out.write(reg[k]);
43     }
44 }
```



The screenshot shows the Vivado IDE interface for the 'shift_hls' project. The left pane displays DRCs (Design Rule Checks) and the Explorer. The main editor shows the C++ code for 'shiftRegister.cpp'. The code implements a 10-bit shift register using a loop and a register.

```
1 #include <iostream>
2 #include "ap_int.h"
3 #include "hls_stream.h"
4 using namespace std;
5 void shiftRegister(hls::stream<ap_uint<1>> &in, hls::stream<ap_uint<4>> &out);
6
7 int main() {
8     hls::stream<ap_uint<1>> instream;
9     hls::stream<ap_uint<4>> outstream;
10     ap_uint<1> inputs[4] = {1,0,1,0};
11     for (int i = 0; i < 4; i++) {
12         instream.write(inputs[i]);
13         shiftRegister(instream, outstream);
14     }
15     ap_uint<4> output_array[4];
16     for (int j = 0; j < 4; j++) {
17         outstream.read(output_array[j]);
18     }
19
20     std::cout << "Output " << endl << " : ";
21     for (int j=0;j<4;j++) {
22         cout<<output_array[j]<<" ";
23     }
24     cout << endl;
25     return 0;
26 }
27
28
29
30
```

Result :



Synthesis Result :

