Top file for Q1,Q2

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 03.10.2019 14:10:48
// Design Name:
// Module Name: top
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module top
#(
    parameter integer C_S_AXIS_X_TDATA_WIDTH = 32,
    parameter integer C_S_AXIS_T_TDATA_WIDTH = 32,
    parameter integer C_S_AXIS_N_TDATA_WIDTH = 32,
    parameter integer C_S_AXIS_M_TDATA_WIDTH = 32
)
(
    input s_aclk,
    input s_aresetn,
    input [C_S_AXIS_X_TDATA_WIDTH-1:0] s_x_tdata,
    input s_x_tvalid,
    input [C_S_AXIS_X_TDATA_WIDTH-1:0] s_m_tdata,
    input s_m_tvalid,
    input [C_S_AXIS_T_TDATA_WIDTH-1:0] s_t_tdata,
    input s_t_tvalid,
    input [C_S_AXIS_N_TDATA_WIDTH-1:0] s_n_tdata,
    input s_n_tvalid,
    output s_x_tready,
    output s_t_tready,
    output s_n_tready,
    output s_m_tready,
    output final_q_valid,
    output [31:0] final_q
);


    //1
    wire float_x_valid, float_x_ready;
    wire [31:0] float_x;
    floating_point_0 fx(
        .aclk(s_aclk),
        .aresetn(s_aresetn),
        .s_axis_a_tvalid(s_x_tvalid),
        .s_axis_a_tready(s_x_tready),
        .s_axis_a_tdata(s_x_tdata),
        .m_axis_result_tvalid(float_x_valid),
        .m_axis_result_tready(float_x_ready),
        .m_axis_result_tdata(float_x)
    );

   //2
    wire float_t_valid, float_t_ready;
    wire [31:0] float_t;
    floating_point_0 ft(
        .aclk(s_aclk),
        .aresetn(s_aresetn),
        .s_axis_a_tvalid(s_t_tvalid),
        .s_axis_a_tready(s_t_tready),
        .s_axis_a_tdata(s_t_tdata),
        .m_axis_result_tvalid(float_t_valid),
        .m_axis_result_tready(float_t_ready),
        .m_axis_result_tdata(float_t)
    );

   //3
    wire float_n_valid, float_n_ready;
    wire [31:0] float_n;
```

```
floating_point_0 fn(
    .aclk(s_aclk),
    .aresetn(s_aresetn),
    .s_axis_a_tvalid(s_n_tvalid),
    .s_axis_a_tready(s_n_tready),
    .s_axis_a_tdata(s_n_tdata),
    .m_axis_result_tvalid(float_n_valid),
    .m_axis_result_tready(float_n_ready),
    .m_axis_result_tdata(float_n)
);

//4
wire float_m_valid, float_m_ready;
wire [31:0] float_m;
floating_point_0 fm(
    .aclk(s_aclk),
    .aresetn(s_aresetn),
    .s_axis_a_tvalid(s_m_tvalid),
    .s_axis_a_tready(s_m_tready),
    .s_axis_a_tdata(s_m_tdata),
    .m_axis_result_tvalid(float_m_valid),
    .m_axis_result_tready(float_m_ready),
    .m_axis_result_tdata(float_m)
);




wire sub1_valid,sub1_ready;
wire [31:0] sub1_data;
subtract x_minus_n(
      .aclk(s_aclk),
      .aresetn(s_aresetn),

      .s_axis_a_tvalid(float_x_valid),
      .s_axis_a_tready(float_x_ready),
      .s_axis_a_tdata(float_x),

      .s_axis_b_tvalid(float_n_valid),
      .s_axis_b_tready(float_n_ready),
      .s_axis_b_tdata(float_n),

      .m_axis_result_tvalid(sub1_valid),
        .m_axis_result_tready(sub1_ready),
        .m_axis_result_tdata(sub1_data)

    );

     wire mul1_valid, mul1_ready;
    wire [31:0] mul1_data;
    multiply first_sq(
        .aclk(s_aclk),
        .aresetn(s_aresetn),

        .s_axis_a_tvalid(sub1_valid),
        .s_axis_a_tready(sub1_ready),
        .s_axis_a_tdata(sub1_data),
        .s_axis_b_tvalid(sub1_valid),
//        .s_axis_b_tready(sub1_ready),
        .s_axis_b_tdata(sub1_data),

        .m_axis_result_tvalid(mul1_valid),
        .m_axis_result_tready(mul1_ready),
        .m_axis_result_tdata(mul1_data)
    );




    wire sub2_valid,sub2_ready;
    wire [31:0] sub2_data;
    subtract t_minus_m(
        .aclk(s_aclk),
        .aresetn(s_aresetn),

        .s_axis_a_tvalid(float_t_valid),
        .s_axis_a_tready(float_t_ready),
        .s_axis_a_tdata(float_t),

        .s_axis_b_tvalid(float_m_valid),
        .s_axis_b_tready(float_m_ready),
        .s_axis_b_tdata(float_m),

        .m_axis_result_tvalid(sub2_valid),
        .m_axis_result_tready(sub2_ready),
        .m_axis_result_tdata(sub2_data)

    );

     wire mul2_valid, mul2_ready;
    wire [31:0] mul2_data;
    multiply second_sq(
        .aclk(s_aclk),
        .aresetn(s_aresetn),

        .s_axis_a_tvalid(sub2_valid),
        .s_axis_a_tready(sub2_ready),
        .s_axis_a_tdata(sub2_data),
        .s_axis_b_tvalid(sub2_valid),
//        .s_axis_b_tready(sub2_ready),
        .s_axis_b_tdata(sub2_data),

        .m_axis_result_tvalid(mul2_valid),
        .m_axis_result_tready(mul2_ready),
        .m_axis_result_tdata(mul2_data)
    );


    wire add1_valid, add1_ready;
    wire [31:0] add1_data;
    add both(
        .aclk(s_aclk),
        .aresetn(s_aresetn),

        .s_axis_a_tvalid(mul1_valid),
        .s_axis_a_tready(mul1_ready),
        .s_axis_a_tdata(mul1_data),

        .s_axis_b_tvalid(mul2_valid),
        .s_axis_b_tready(mul2_ready),
        .s_axis_b_tdata(mul2_data),

        .m_axis_result_tvalid(add1_valid),
        .m_axis_result_tready(add1_ready),
        .m_axis_result_tdata(add1_data)
    );
    sqr_root blah(
        .aclk(s_aclk),
        .aresetn(s_aresetn),

        .s_axis_a_tvalid(add1_valid),
        .s_axis_a_tready(add1_ready),
        .s_axis_a_tdata(add1_data),

        .m_axis_result_tvalid(final_q_valid),
        .m_axis_result_tready(1),
        .m_axis_result_tdata(final_q)

    );





endmodule
```

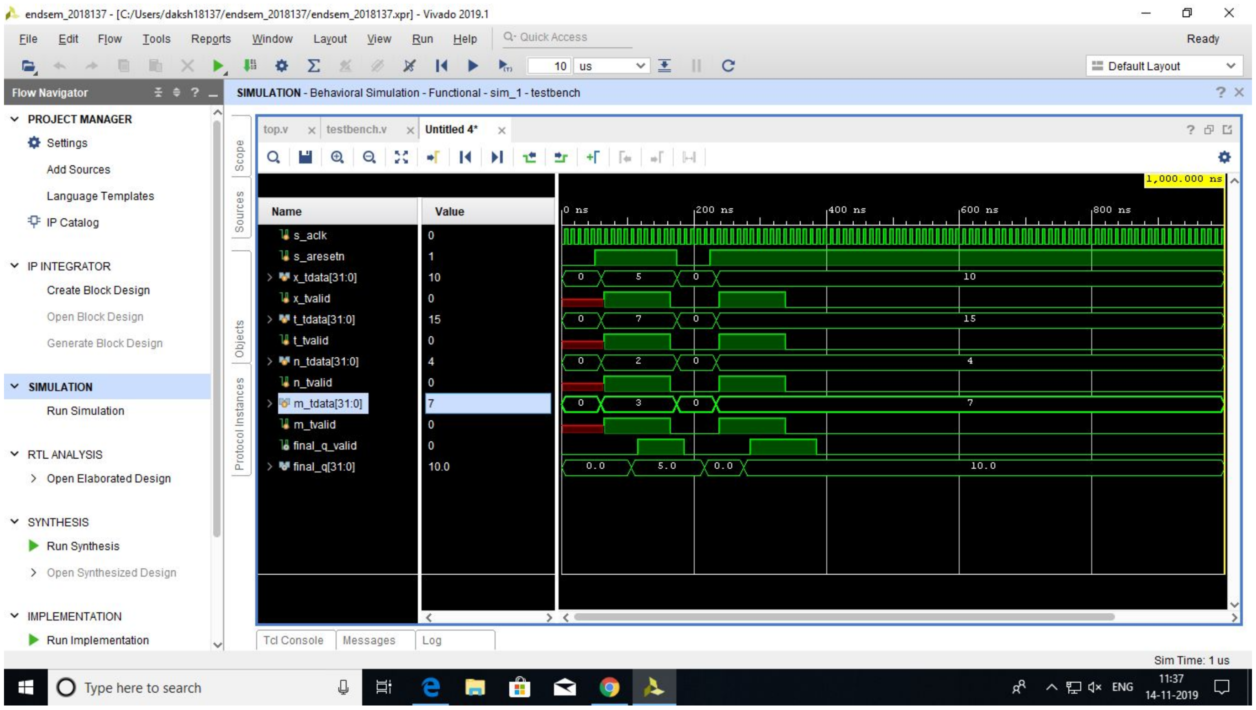
Testbench file for Q1

```
`timescale 1ns / 1ps


module testbench(

    );



    reg s_aclk = 0;
    always #5 s_aclk = ~ s_aclk;

    reg s_aresetn = 0;
    reg [31:0] x_tdata;
    reg x_tvalid;
    reg [31:0] t_tdata;
    reg t_tvalid;
    reg [31:0] n_tdata;
    reg n_tvalid;
    reg [31:0] m_tdata;
    reg m_tvalid;

    wire final_q_valid;
    wire [31:0] final_q;

    top
    #(
        .C_S_AXIS_X_TDATA_WIDTH(32),
        .C_S_AXIS_T_TDATA_WIDTH(32),
        .C_S_AXIS_N_TDATA_WIDTH(32),
        .C_S_AXIS_M_TDATA_WIDTH(32)
    )

    t1(
        .final_q_valid(final_q_valid),
        .final_q(final_q),

        .s_aclk(s_aclk),
        .s_aresetn(s_aresetn),

        .s_x_tvalid(x_tvalid),
        .s_x_tdata(x_tdata),

        .s_t_tvalid(t_tvalid),
        .s_t_tdata(t_tdata),

        .s_n_tvalid(n_tvalid),
        .s_n_tdata(n_tdata),

        .s_m_tvalid(m_tvalid),
        .s_m_tdata(m_tdata)
    );

    initial
    begin
        #0  s_aresetn=0;
            x_tdata = 32'd0;
            t_tdata = 32'd0;
            n_tdata = 32'd0;
            m_tdata = 32'd0;


        #100 s_aresetn = 1;

        #10 x_tdata = 32'd5;
            t_tdata = 32'd7;
            n_tdata = 32'd2;
            m_tdata = 32'd3;


        #4 x_tvalid = 1;
           t_tvalid = 1;
           n_tvalid = 1;
           m_tvalid = 1;

        #50 x_tvalid = 0;
           t_tvalid = 0;
           n_tvalid = 0;
           m_tvalid = 0;
    end


endmodule
```

Testbench file for Q2

```
`timescale 1ns / 1ps


module testbench(

    );



    reg s_aclk = 0;
    always #5 s_aclk = ~ s_aclk;

    reg s_aresetn = 0;
    reg [31:0] x_tdata;
    reg x_tvalid;
    reg [31:0] t_tdata;
    reg t_tvalid;
    reg [31:0] n_tdata;
    reg n_tvalid;
    reg [31:0] m_tdata;
    reg m_tvalid;

    wire final_q_valid;
    wire [31:0] final_q;

    top
    #(
        .C_S_AXIS_X_TDATA_WIDTH(32),
        .C_S_AXIS_T_TDATA_WIDTH(32),
        .C_S_AXIS_N_TDATA_WIDTH(32),
        .C_S_AXIS_M_TDATA_WIDTH(32)
    )

    t1(
        .final_q_valid(final_q_valid),
        .final_q(final_q),

        .s_aclk(s_aclk),
        .s_aresetn(s_aresetn),

        .s_x_tvalid(x_tvalid),
        .s_x_tdata(x_tdata),

        .s_t_tvalid(t_tvalid),
        .s_t_tdata(t_tdata),

        .s_n_tvalid(n_tvalid),
        .s_n_tdata(n_tdata),

        .s_m_tvalid(m_tvalid),
        .s_m_tdata(m_tdata)
    );

    initial
    begin
        #0  s_aresetn=0;
            x_tdata = 32'd0;
            t_tdata = 32'd0;
            n_tdata = 32'd0;
            m_tdata = 32'd0;


        #50 s_aresetn = 1;

        #10 x_tdata = 32'd5;
            t_tdata = 32'd7;
            n_tdata = 32'd2;
            m_tdata = 32'd3;


        #4 x_tvalid = 1;
           t_tvalid = 1;
           n_tvalid = 1;
           m_tvalid = 1;

        #100 x_tvalid = 0;
           t_tvalid = 0;
           n_tvalid = 0;
           m_tvalid = 0;
```

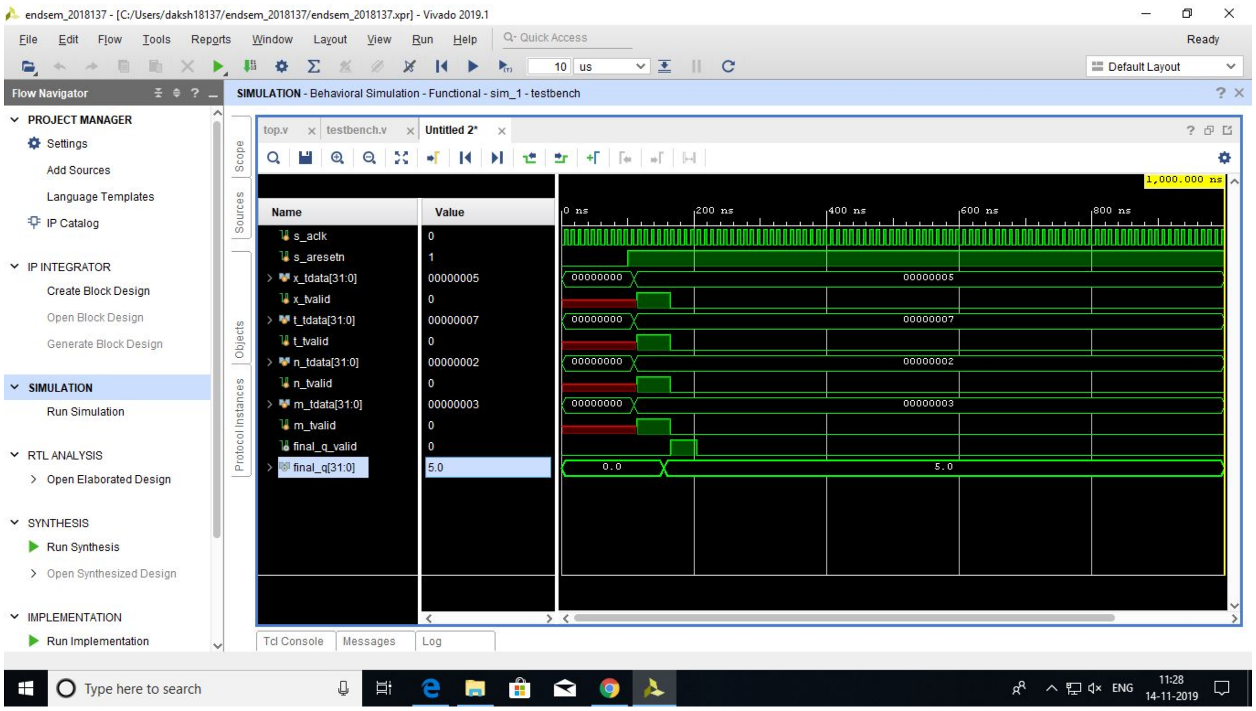
```
        #10  s_aresetn=0;
         x_tdata = 32'd0;
         t_tdata = 32'd0;
         n_tdata = 32'd0;
         m_tdata = 32'd0;


         #50 s_aresetn = 1;

         #10 x_tdata = 32'd10;
             t_tdata = 32'd15;
             n_tdata = 32'd4;
             m_tdata = 32'd7;


         #4 x_tvalid = 1;
            t_tvalid = 1;
            n_tvalid = 1;
            m_tvalid = 1;

         #100 x_tvalid = 0;
            t_tvalid = 0;
            n_tvalid = 0;
            m_tvalid = 0;



    end


endmodule
```


endsem_2018137 - [C:/Users/daksh18137/endsem_2018137/endsem_2018137.xpr] - Vivado 2019.1

SIMULATION - Behavioral Simulation - Functional - sim_1 - testbench

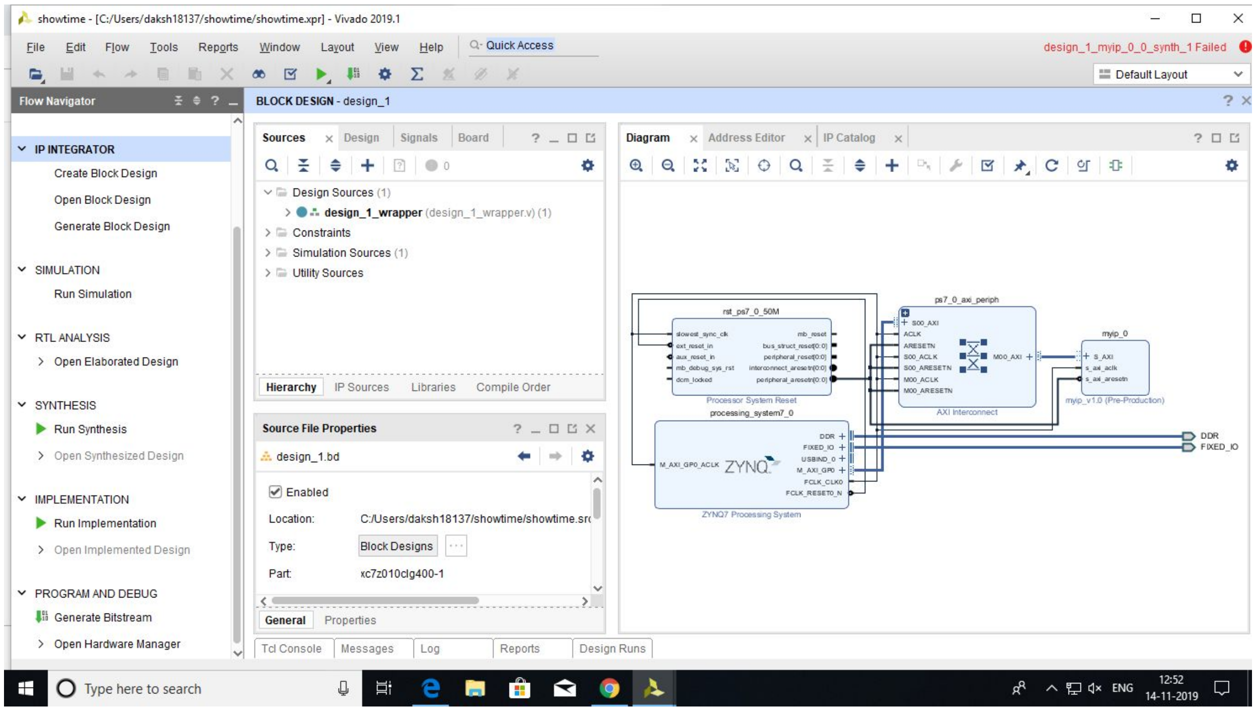
Name	Value
s_aclk	0
s_aresetn	1
x_tdata[31:0]	10
x_tvalid	0
t_tdata[31:0]	15
t_tvalid	0
n_tdata[31:0]	4
n_tvalid	0
m_tdata[31:0]	7
m_tvalid	0
final_q_valid	0
final_q[31:0]	10.0

Sim Time: 1 us


endsem_2018137 - [C:/Users/daksh18137/endsem_2018137/endsem_2018137.xpr] - Vivado 2019.1

SIMULATION - Behavioral Simulation - Functional - sim_1 - testbench

Name	Value
s_aclk	0
s_aresetn	1
x_tdata[31:0]	00000005
x_tvalid	0
t_tdata[31:0]	00000007
t_tvalid	0
n_tdata[31:0]	00000002
n_tvalid	0
m_tdata[31:0]	00000003
m_tvalid	0
final_q_valid	0
final_q[31:0]	5.0



Enabled

Location: C:/Users/daksh18137/showtime/showtime.src

Type: Block Designs

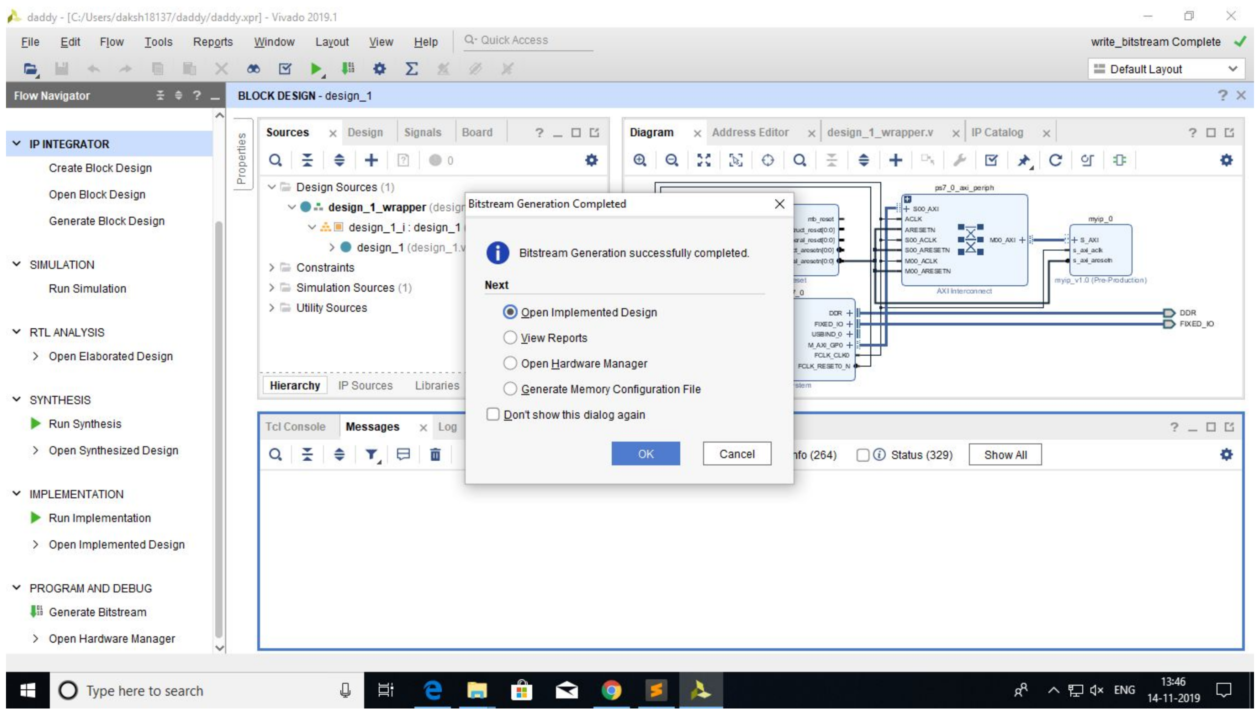
Part: xc7z010clg400-1

Q3,4

```
#include &lt;stdio.h&gt;
#include &quot;platform.h&quot;
#include &quot;xil_printf.h&quot;
#include &quot;xparameters.h&quot;
#include &quot;xil_io.h&quot;
#include &quot;math.h&quot;
#include &lt;time.h&gt;
#include &lt;stdlib.h&gt;
#include &quot;xtime_l.h&quot;
#include &quot;practiceIp.h&quot;

long double q_basic(double x, double t, double n, double m){
double q;


q=sqrt(pow(x-n,2)+ pow(t-m,2))
```

```
return q;
}




int main()
{
init_platform();
double q = 0; double q_machine = 0;
int x = 0;
int t = 0;
int n = 0;
int m = 0;

int inform = 0;
XTime tprocessorStart, tprocessorEnd, tplStart,tplEnd;
//XTime_GetTime(&amp;tStart);
if(inform == 0){
x = 1; t = 1;
n = 4;
m = 4;
}
if(inform == 9){
t = t + 1;
n = n+1;
m=m+1;
}
if(inform == 11){
t = t+1;
m=m+1;
x = x+1;
n = n+1;
}


XTime_GetTime(&amp;tprocessorStart);
q = (double) q_basic(x, t, n, m);
XTime_GetTime(&amp;tprocessorEnd);
printf(&quot;X: %d, T: %d, N: %d,M: %d \r\n&quot;, x, t, n,m);
printf(&quot;Q: %f \r\n&quot;, q);
////////////////////////////////////////////////////////////////////////
////////////
```

```
MYIP_mWriteReg(XPAR_MYIP_0_S_AXI_BASEADDR,0, 0x00000000);
XTime_GetTime(&amp;tplStart);
q_machine = MYIP_mReadReg(XPAR_MYIP_0_S_AXI_BASEADDR, 16);
XTime_GetTime(&amp;tplEnd);
printf(&quot;Q machine: %f \r\n&quot;, q_machine);
//XTime_GetTime(&amp;tEnd);
// printf(&quot;Output took %llu clock cycles.\n&quot;, 2*(tEnd - tStart));
printf(&quot;PI took %.2f us.\n&quot;, 1.0 * (tplEnd - tplStart) /
(COUNTS_PER_SECOND/1000000));
printf(&quot;PS took %.2f us.\n&quot;, 1.0 * (tprocessorEnd - tprocessorStart)
/ (COUNTS_PER_SECOND/1000000));

printf( 1.0 * (tplEnd - tplStart) /
(COUNTS_PER_SECOND/1000000)/1.0 * (tprocessorEnd - tprocessorStart)
/ (COUNTS_PER_SECOND/1000000));
cleanup_platform();
return 0;
}
```



Bitstream Generation Completed

Bitstream Generation successfully completed.

Next

- [✓ Open Implemented Design]
- View Reports
- Open Hardware Manager
- Generate Memory Configuration File

Don't show this dialog again

OK Cancel