High Performance VLSI/IC Systems

MMP (Mobile Multimedia Processor) power reduction with RTL

Adalberto Claudio Quiros
A20294552

INDEX

Objective.

In this homework, the implementation of the Enhanced Clock Gating has been studied as an effective way to reduce the power consumption of a circuit.

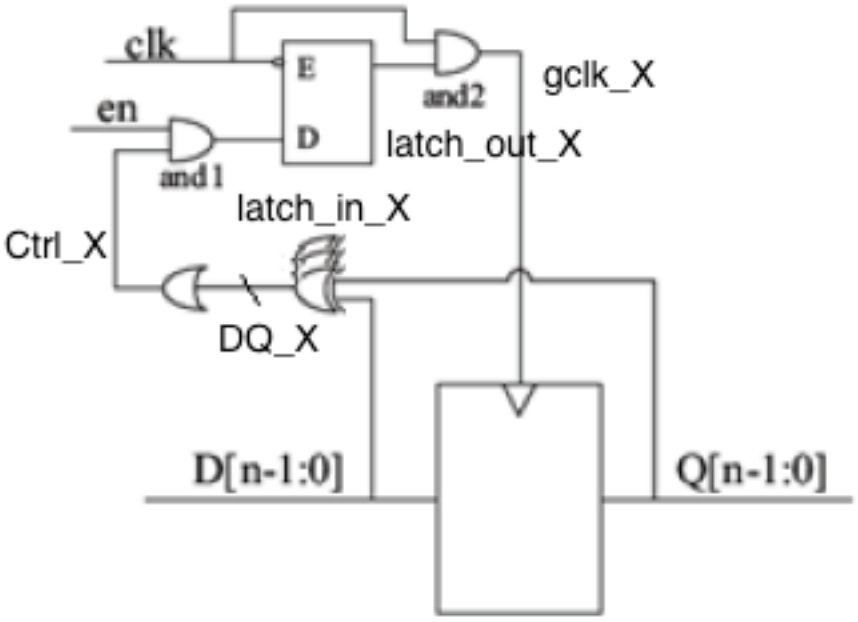
Using the provided example the Mobile Multimedia Processor, modifying it with this technique to prove and study the impact of this implementation on the power consumption. Not leaving apart that the functionality of the circuit still the same, therefore the circuit behaves, as it should.

This will be checked with Synopsys Formality, after this tested and simulated with the modified test bench, and then finally its power consumption will be tested with Synopsys Design Compiler and Power Compiler.

Theory.

Enhanced Clock Gating is a common and effective technique used in circuit design, modifying the RTL design to its final purpose. This implementation takes into account two possible scenarios to reduce power consumption.

- Input of Flip-Flop at this stage is the same as the Output of the Flip-Flop so the next's stage Output will be the same. For this reason there's no need to enable the clock that make the Flip-Flip work.
- Enable input is low, and then the clock input to the Flip-Flop should be disabled so this one does not consume power.



In this figure, as we can see, is the implementation of the Enhanced Clock Gating. Doing a XOR of every bit of Input and Output of the Flip-Flop, InputBit[i]⊕OutputBit[i] will notice if the next output will change, then with all the results bit do an OR so if any of this bits is 1, this way will know if the output will change for the next stage, if not the result will be equal to 0.

After this compare this result signal of the OR, with the enable signal so if any of this signals are 0 then the clock should be disable, doing this with an AND, and to finish the implementation use a latch that will work in the 0 cycle of the clock, the purpose of this is to make sure the system has no glitches so that the change of the clock takes effect in this half cycle and not in the 1 cycle that could lead to timing issues.

And to finish the clock is enable or disable with an AND with this result signal.

The following code corresponds to the Flip-Flops of either 8 or 32 bits.

```
    //Wire 8 or 32bits DQ_X, XORbitwise of the output of the
    //Flip-Flop and the input.
    assign DQ_1 = DC_EX_tr_0_op1_inp ^ DC_EX_tr_0_op1_internal;
    //Or reduction of DQ_X to detect if the signal doesn't change
    assign Ctrl_1 = | DQ_1;
    // And with the enable signal
    and (latch_in_1, Ctrl_1, DC_EX_tr_0_op1_EW);
    // Latch and And, glitches prevented.
    latch l1(latch_in_1, latch_out_1, clk_main_in);
    and (gclk_1, clk_main_in, latch_out_1);

    always @(negedge rst_main_in or posedge gclk_1)

    begin: PROC_DC_EX_tr_0_op1_sync

        if (rst_main_in == 1'b0)
        begin
            DC_EX_tr_0_op1_internal <= 32'd0;
        end
        else
        begin
            //if (DC_EX_tr_0_op1_EW == 1'b1)
            //begin
                  DC_EX_tr_0_op1_internal <= (DC_EX_tr_0_op1_inp);
                  assign QPrev = DC_EX_tr_0_op1_inp;
            //end
        end
    end
```

For the case of 1 Bit data of the Flip-Flop there no need to do the OR gate.

```
    assign Ctrl_33 = to_0_alu_rrr_ex_value_internal^to_0_alu_rrr_ex_value_inp;
    and (latch_in_33, Ctrl_33, to_0_alu_rrr_ex_EW);
    latch l33(latch_in_33, latch_out_33, clk_main_in);
    and (gclk_33, clk_main_in, latch_out_33);

    always @(negedge rst_main_in or posedge gclk_33)

    begin: PROC_pipe_reg_to_0_alu_rrr_ex_sync

        if (rst_main_in == 1'b0)
        begin
            to_0_alu_rrr_ex_value_internal <= 1'b0;
        end
        else
        begin
            //if (to_0_alu_rrr_ex_EW == 1'b1)
            //begin
                to_0_alu_rrr_ex_value_internal <= (to_0_alu_rrr_ex_value_inp);
            //end
        end
    end
```

Implementation.

Using the Enhanced Clock Gating technique, the RTL code of the processor will be modified in order to find the best implementation that has the lowest power consumption.

After modifying the RTL code, to check the functionality of the circuit and compare it to the reference one, the original; to guarantee the functionality of the modified one, Synopsys Formality has been used, and then the simulation of the testbench using ModelSim, to finally get it's power consumption using Synopsys Design Compiler and Power Compiler.

The objective has been to find the Flip-Flop candidates that would lead to a lower power consumption, this way looking for Flip-Flops that depended the main clk, "clk_main_in", there were three types of these; 1, 8 and 32 bits of input data.

The different implementations made are the following:

1. 1 bit Enhanced Clock Gating: ECG technique applied to all 1 bit data Flip-Flops
2. 8 bit Enhanced Clock Gating: ECG technique applied to all 8 bit data Flip-Flops
3. 32 bit Enhanced Clock Gating: ECG technique applied to all 32 bit data Flip-Flops
4. 1 - 32 bit Enhanced Clock Gating: ECG technique applied to all 1 and 32 bit data Flip-Flops
5. 1 – 8 bit Enhanced Clock Gating: ECG technique applied to all 1 and 8 bit data Flip-Flops
6. 8 - 32 bit Enhanced Clock Gating: ECG technique applied to all 8 and 32 bit data Flip-Flops
7. All Enhanced Clock Gating: ECG technique applied to all types of Flip-Flops.

Results.

To compare the result of the modifications, the original and ACG implementations are presented.

Original DC_EX power estimation.

This is the power estimation of the original circuit, the starting point to final the final power optimization.

Power (mW)	**Original Design**
Total Dynamic	3.0806
Cell Leakage	0.1266353

Total	3.2072353
Area	

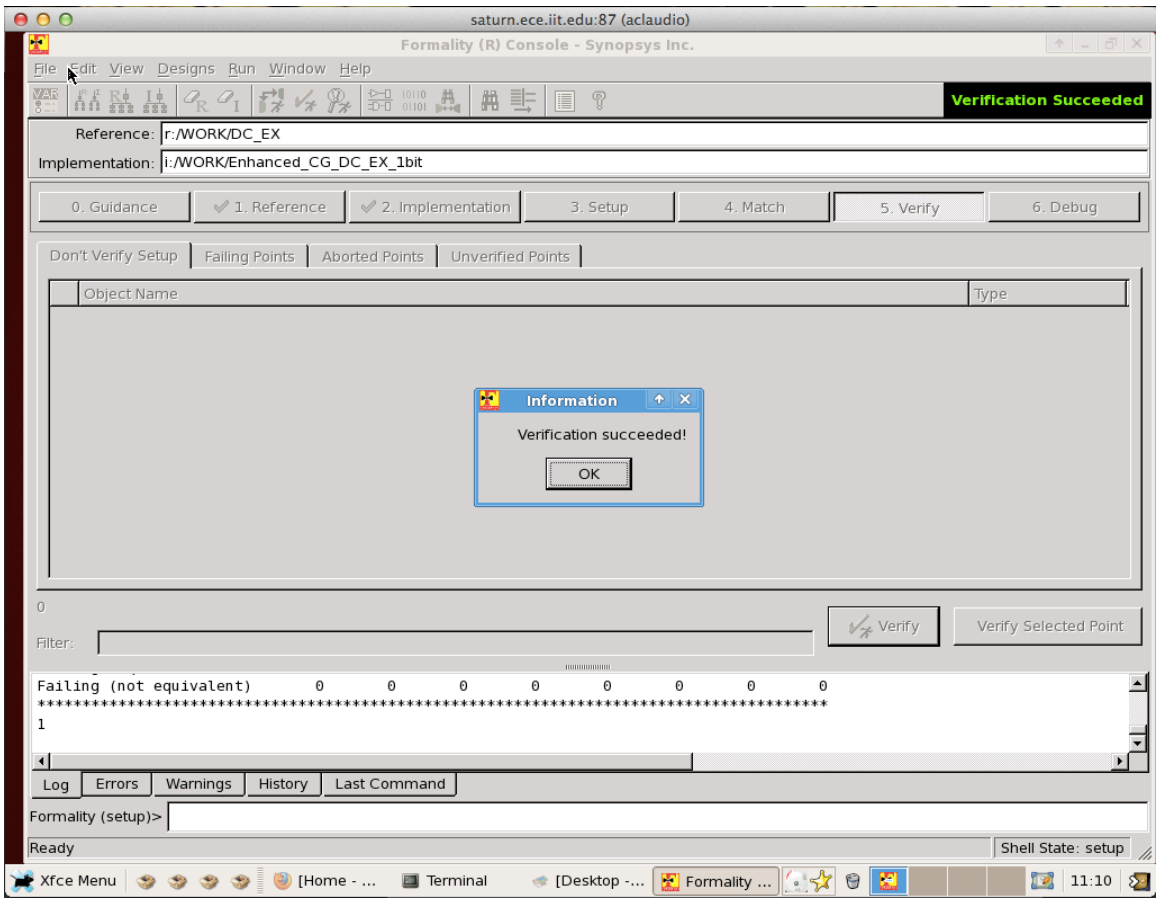
Automatic Clock Gating Insertion.

Using Power Compiler, the insertion of the Clock Gating will be done automatically during the synthesis procedure and as the results shows it is a good implementation, the power consumption have been reduce compared to the original one.

Power (mW)	**Automatic Clock Gating Insertion**
Total Dynamic	2.3668
Cell Leakage	0.1168189
Total	2.4836
Area	

1 bit Enhanced Clock Gating:

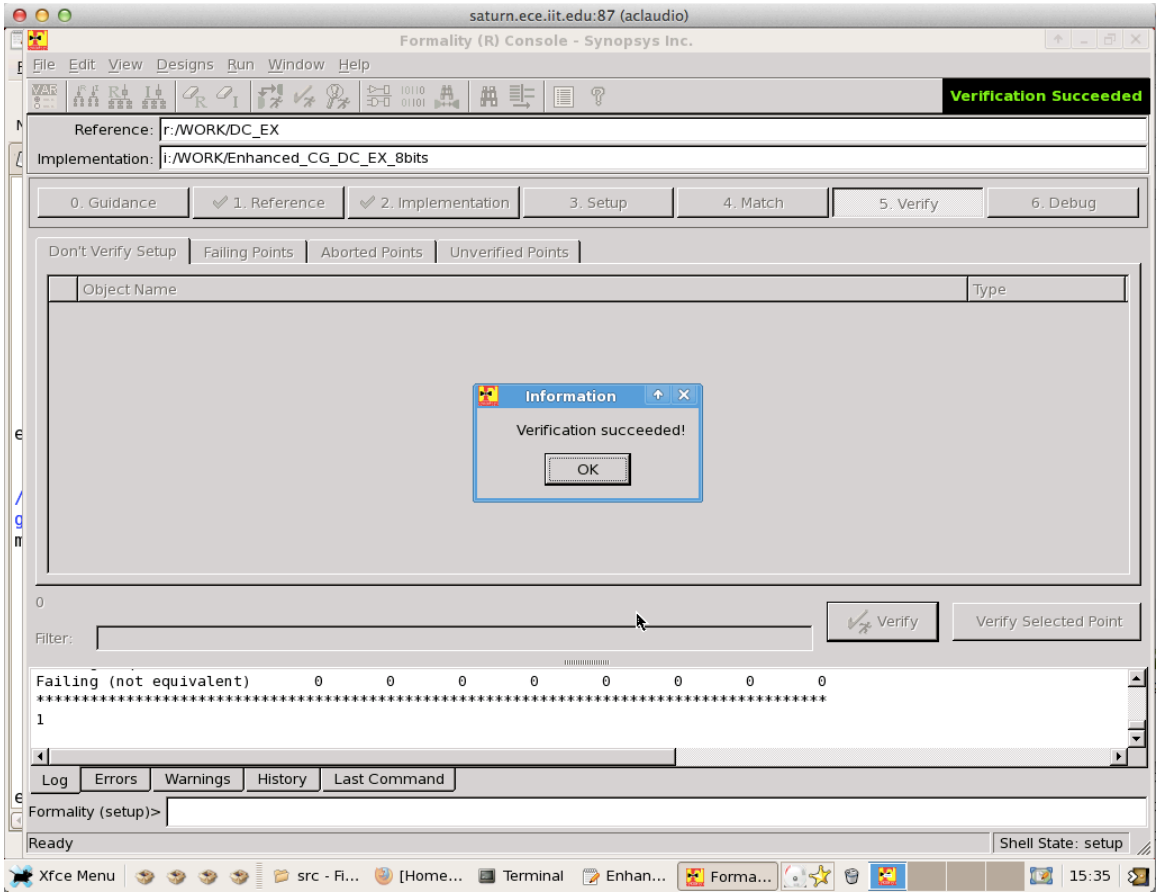
Enhanced_CG_DC_EX_1bit.v



Power (mW)	1 bit Enhanced Clock Gating
Total Dynamic	1.4529
Cell Leakage	0.1285818
Total	1.5814818
Area	18730.2321

8 bit Enhanced Clock Gating:

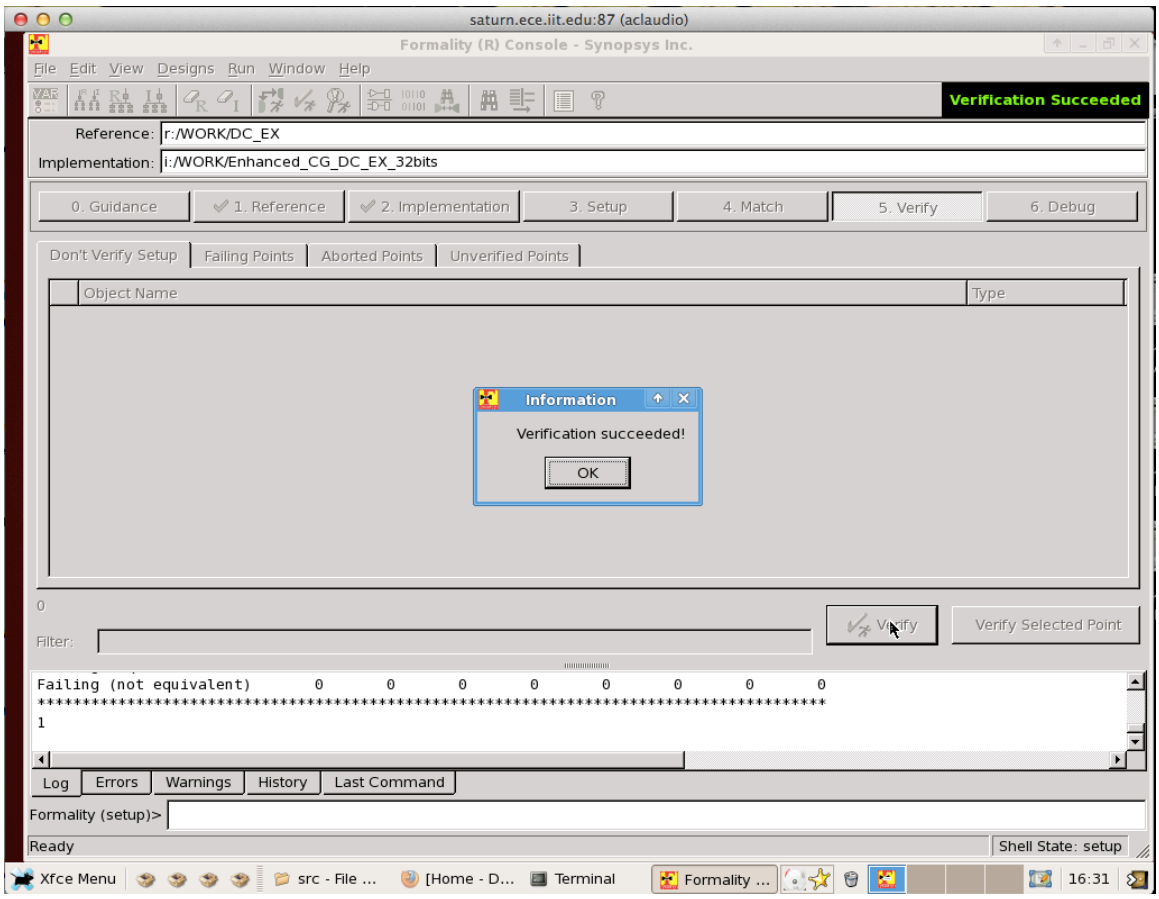
Enhanced_CG_DC_EX_8bits.v



Power (mW)	8 bit Enhanced Clock Gating
Total Dynamic	1.3037
Cell Leakage	0.1292289
Total	1.4329289
Area	18932.0311

32 bit Enhanced Clock Gating:

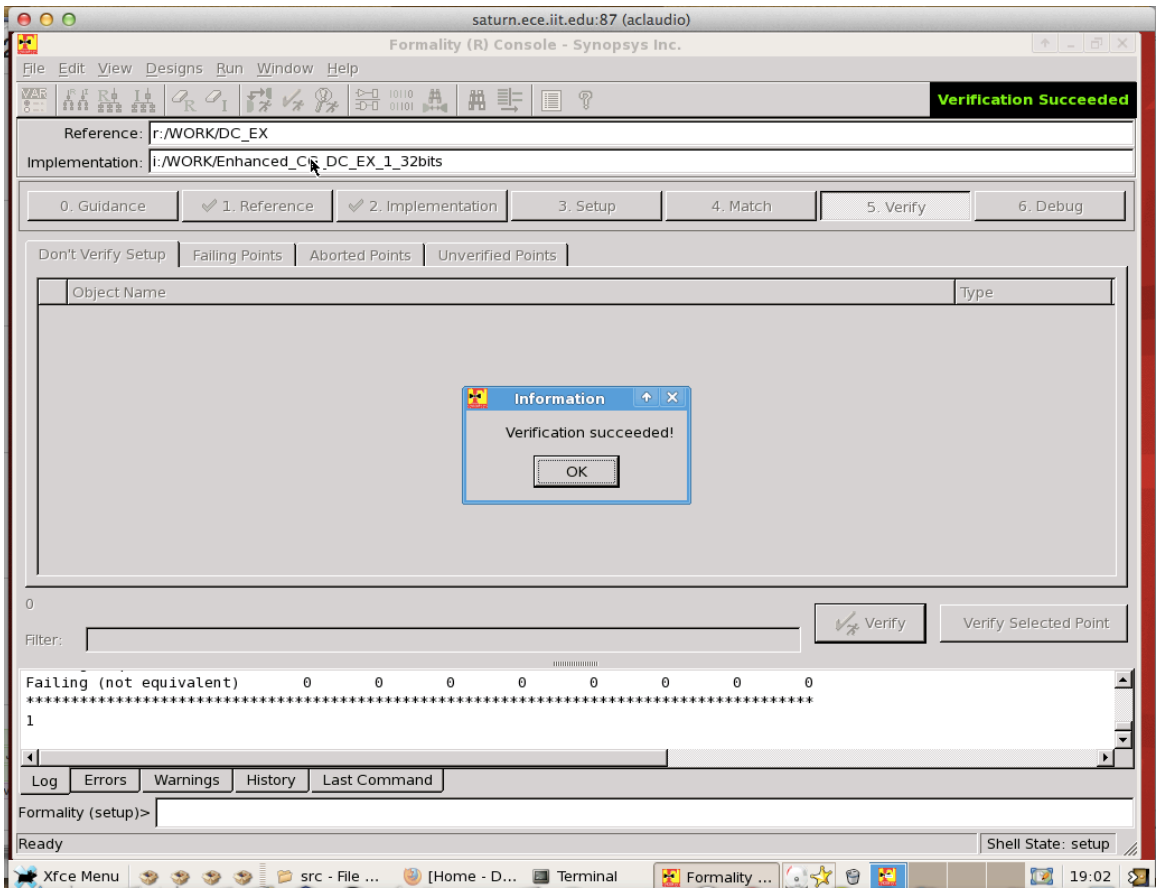
Enhanced_CG_DC_EX_32bits.v



Power (mW)	32 bit Enhanced Clock Gating
Total Dynamic	0.4064414
Cell Leakage	0.1318997
Total	0.5383411
Area	20343.2162

1 - 32 bit Enhanced Clock Gating:

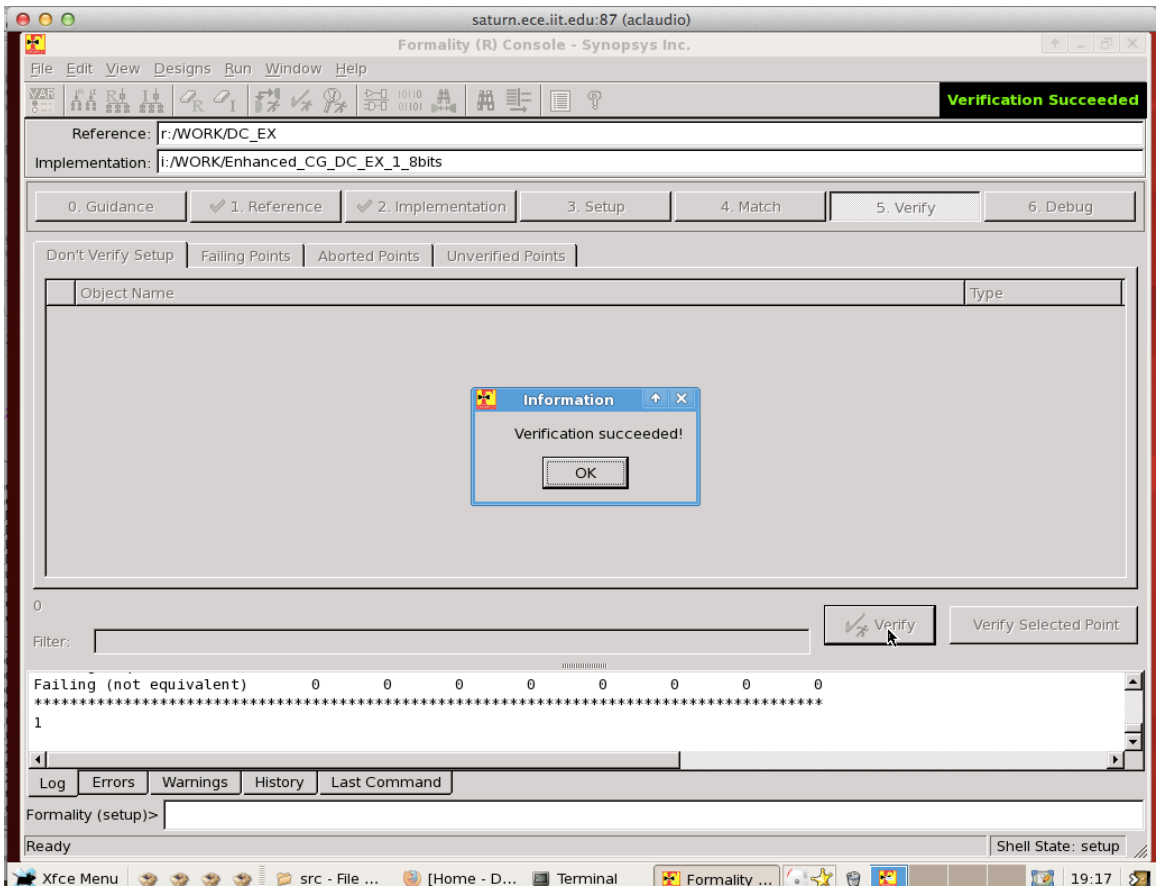
Enhanced_CG_DC_EX_1_32bits.v



Power (mW)	1 - 32 bit Enhanced Clock Gating
Total Dynamic	0.4127558
Cell Leakage	0.1325769
Total	0.5453327
Area	18932.0311

1 - 8 bit Enhanced Clock Gating:

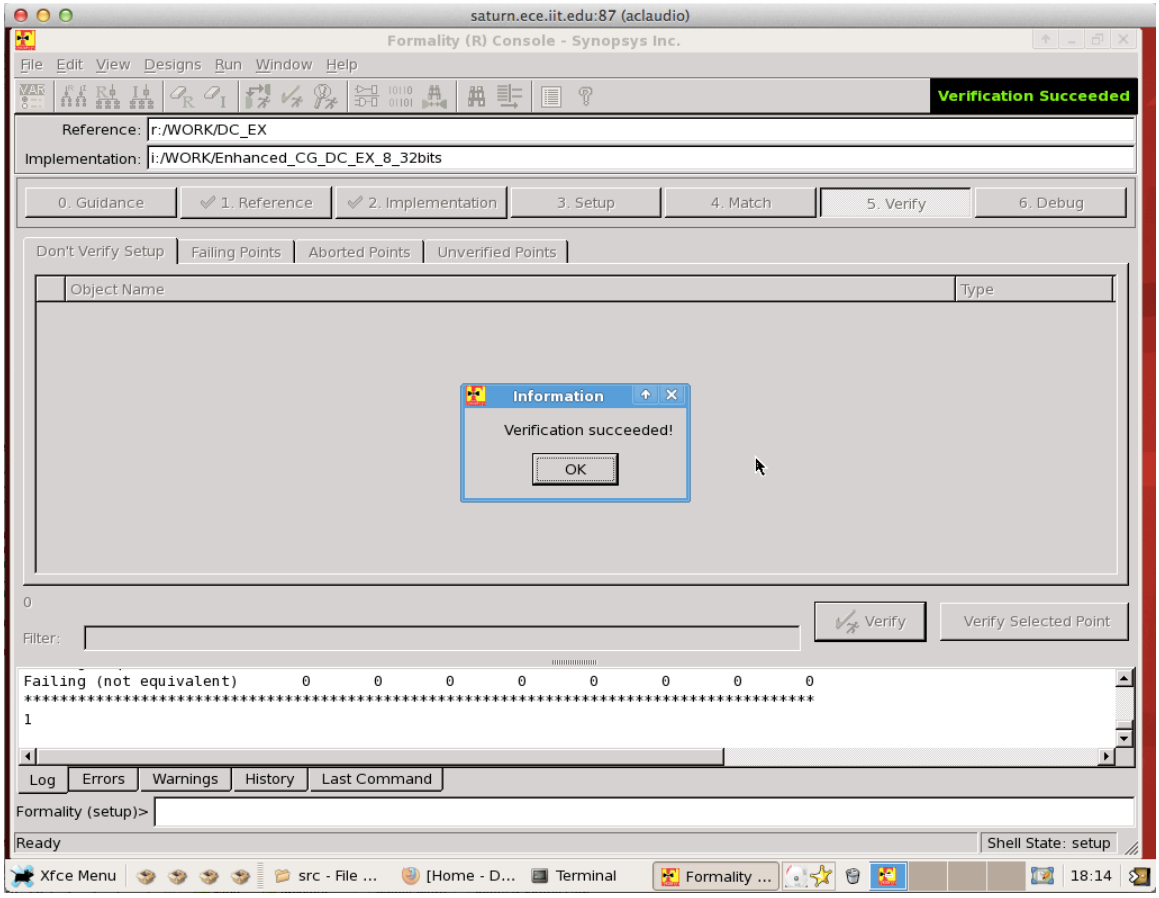
Enhanced_CG_DC_EX_1_8bits.v



Power (mW)	1 - 8 bit Enhanced Clock Gating
Total Dynamic	1.2771
Cell Leakage	0.1290462
Total	1.4061462
Area	19056.39

8 - 32 bit Enhanced Clock Gating:

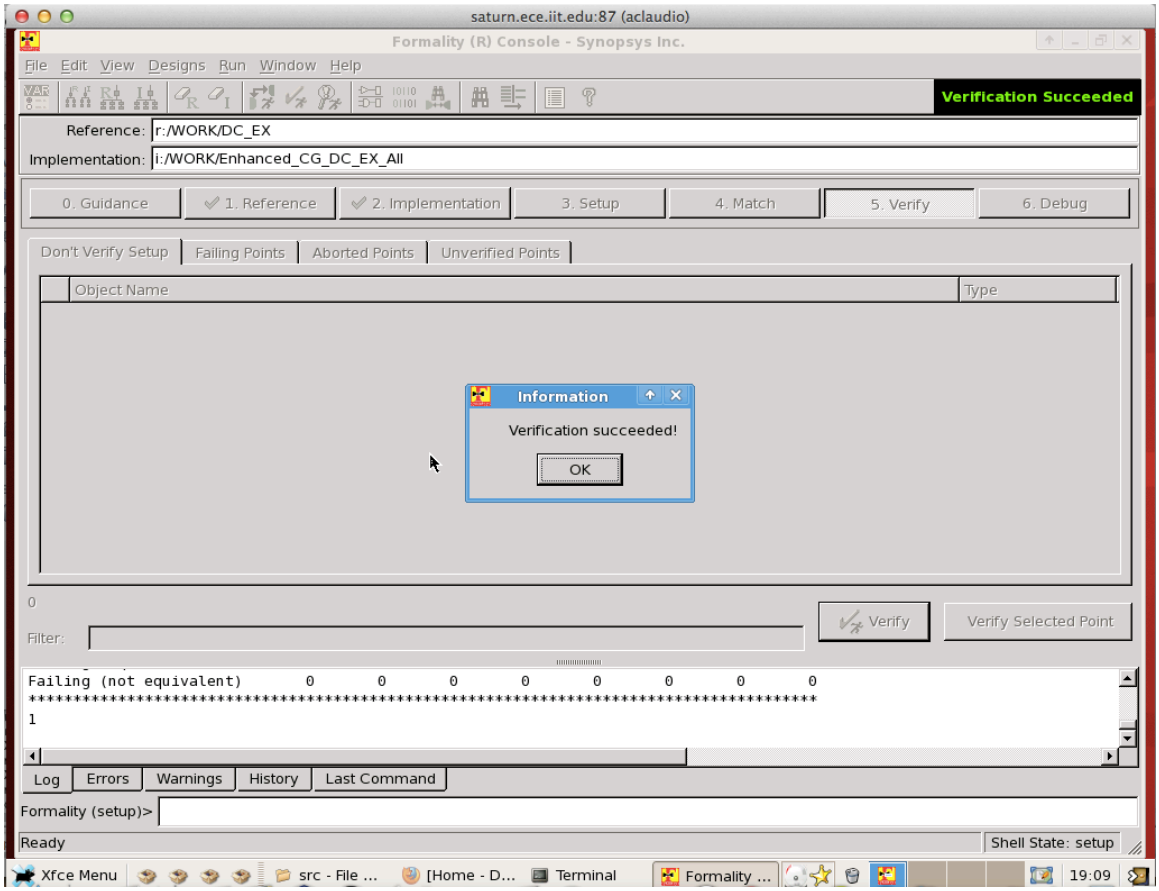
Enhanced_CG_DC_EX_8_32bits.v



Power (mW)	8 - 32 bit Enhanced Clock Gating
Total Dynamic	0.2602367
Cell Leakage	0.1338075
Total	0.3940442
Area	20545.0152

All bit Enhanced Clock Gating:

Enhanced_CG_DC_EX_All.v is the same as ECG_Opti_DC_EX_gen.v



Power (mW)	All bit Enhanced Clock Gating
Total Dynamic	0.2342638
Cell Leakage	0.1341687
Total	0.3684325
Area	20812.9855

Comparative:

Power (mW)	1 bit Enhanced Clock Gating	8 bit Enhanced Clock Gating	32 bit Enhanced Clock Gating	1 - 32 bit Enhanced Clock Gating
Total Dynamic	1.4529	1.3037	0.4064414	0.4127558
Cell Leakage	0.1285818	0.1292289	0.1318997	0.1325769
Total	1.5814818	1.4329289	0.5383411	0.5453327
Area	18730.2321	18932.0311	20343.2162	18932.0311

Power (mW)	1 - 8 bit Enhanced Clock Gating	8 - 32 bit Enhanced Clock Gating	All bit Enhanced Clock Gating
Total Dynamic	1.2771	0.2602367	0.2342638
Cell Leakage	0.1290462	0.1338075	0.1341687
Total	1.4061462	0.3940442	0.3684325
Area	19056.39	20545.0152	20812.9855

In terms of area, as it is logical the implementation with the biggest area is the All bit one, in with the ECG has been applied to all the Flip-Flops, all the AND, OR, XOR gates and the latches increase the area.

In terms of power consumption, in this case the results returned we could came up to the conclusion that in this case because now the power saved not only depends on the enable signal, but also on the times that the signal input of the Flip-Flop changes, the best implementation will be the one in which we apply the ECG to all the Flip-Flops. Also, as we can see the cell leakage increase as we go up in the number of bits, which is logical due to the increase of the number of gates for every bit. But the total dynamic is severally lower.

Comparative with LECG

Power (mW)	32 bit Clock Gating	8 bit Clock Gating	1 bit Clock Gating	Customized Clock Gating
Total Dynamic	2.3885	3.1719	3.0794	2.3901
Cell Leakage	0.1173696	0.124888	0.1273181	0.1164926
Cell Leakage	2.5058	3.29678	3.2067	2.5066

Power (mW)	1 bit Enhanced Clock Gating	8 bit Enhanced Clock Gating	32 bit Enhanced Clock Gating	All bit Enhanced Clock Gating
Total Dynamic	1.4529	1.3037	0.4064414	0.2342638
Cell Leakage	0.1285818	0.1292289	0.1318997	0.1341687
Total	1.5814818	1.4329289	0.5383411	0.3684325
Percentage Saved %	50.68	56.53	78.51	85.30

This results tell us that the ECG implementation produces a several power reduction, if we compare each implementation the decrease is quite important, in the case of the final implementation in both cases the ECG is 10 times lower than the LECG.

The reason because now the most effective implementation is the one in which the ECG is applied to all Flip-Flops, instead of only the 8- 32 bits Flip-Flops in the previous homework, is because now, in this new technique the change of data information for the next stage is taken into account, and for what the results can tell, the changes between data stages in the 1 bit Flip-Flop aren't usual, so it 's worth the ECG being applied.

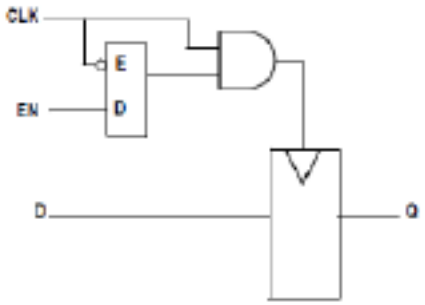
Objective.

In this homework, it has been study how the implementation of the Clock Gating technique can reduce in an effective way the power consumption of a circuit, changing the code at the RTL level.

Using Synopsys Formality the functionality of the modified circuit using Clock Gating will be checked in order to make sure it's final purpose still the same, then compile the code and simulate a test bench on it, to test it's modifications, and finally use Synopsys Design Compiler and Power Compiler.

Theory.

Clock Gating is common and effective technique to reduce power consumption in a circuit design, modifying the RTL to its final purpose. What this implementation does it's to inabilities the clock that controls the flip-flop when the signal Enable is 0, in order to reduce the power consumption. This way there is no rising edge on the input of the flip-flop and the output will not change, then the flip-flop will not work not consuming power.



As this image shows, the enable will be the input data of a latch an the enable of the latch will work with the 0 cycle of the clock, this is done to make sure no glitch occurs so the change of the enable, takes effect in this half cycle and not in the 1 cycle that could lead to synchronous problems.

```
latch l6(DC_EX_tr_0_op2_poll_EW, latch_out_6, clk_main_in);
and (gclk_6, clk_main_in, latch_out_6);

always @(negedge rst_main_in or posedge gclk_6)

begin: PROC_DC_EX_tr_0_op2_poll_sync

    if (rst_main_in == 1'b0)
    begin
        DC_EX_tr_0_op2_poll <= 32'd0;
    end
    else
    begin
        //if (DC_EX_tr_0_op2_poll_EW == 1'b1)
        //begin
            DC_EX_tr_0_op2_poll <= (DC_EX_tr_0_op2_poll_inp);
        //end
    end
end
```

This is an example of how the clock gating is implemented in Verilog, as we can see in this case the enable input of the latch will be DC_EX_tr_0_op2_poll_EW, then using an and gate with the clock and the output of the latch as in previous figure.

Implementation.

Using this technique, the RTL code of the processor given will be changed to find the optimal power consumption. The objective is to find the flip-flops that could be modified using this technique and then different implementations will be compared.

- Automatic Clack Gating Insertion: The power compiler supports an automatic clock gating insertion during the synthesis procedure.
- Customized Clock Gating: All possible flip-flops will be modified.
- 1bit Clock Gating: Flip-flop of 1 bit data will be modified.
- 8bit Clock Gating: Flip-flop of 8 bit data will be modified.
- 32bit Clock Gating: Flip-flop of 32 bit data will be modified.
- Final Clock Gating: Using all the previous information a custom clock gating implementation will be done to find the lowest power consumption of the processor.

In all this cases, after modifying the RTL code, the first step is to use the Synopsys Formality to do a formal verification of the modified circuit, comparing the modified one with DC_EX, the original one, to guarantee that the functionality of the modified one.

If this test is passed, next step is to do the Synthesis and Power Consumption. Compile the Verilog RTL Code and simulate it, to finally get the power results.

Results.

Original DC_EX power estimation.

This is the power estimation of the original circuit, the starting point to final the final power optimization.

Power (mW)	Original Design
Total Dynamic	3.0806
Cell Leakage	0.1266353
Total	3.2072353

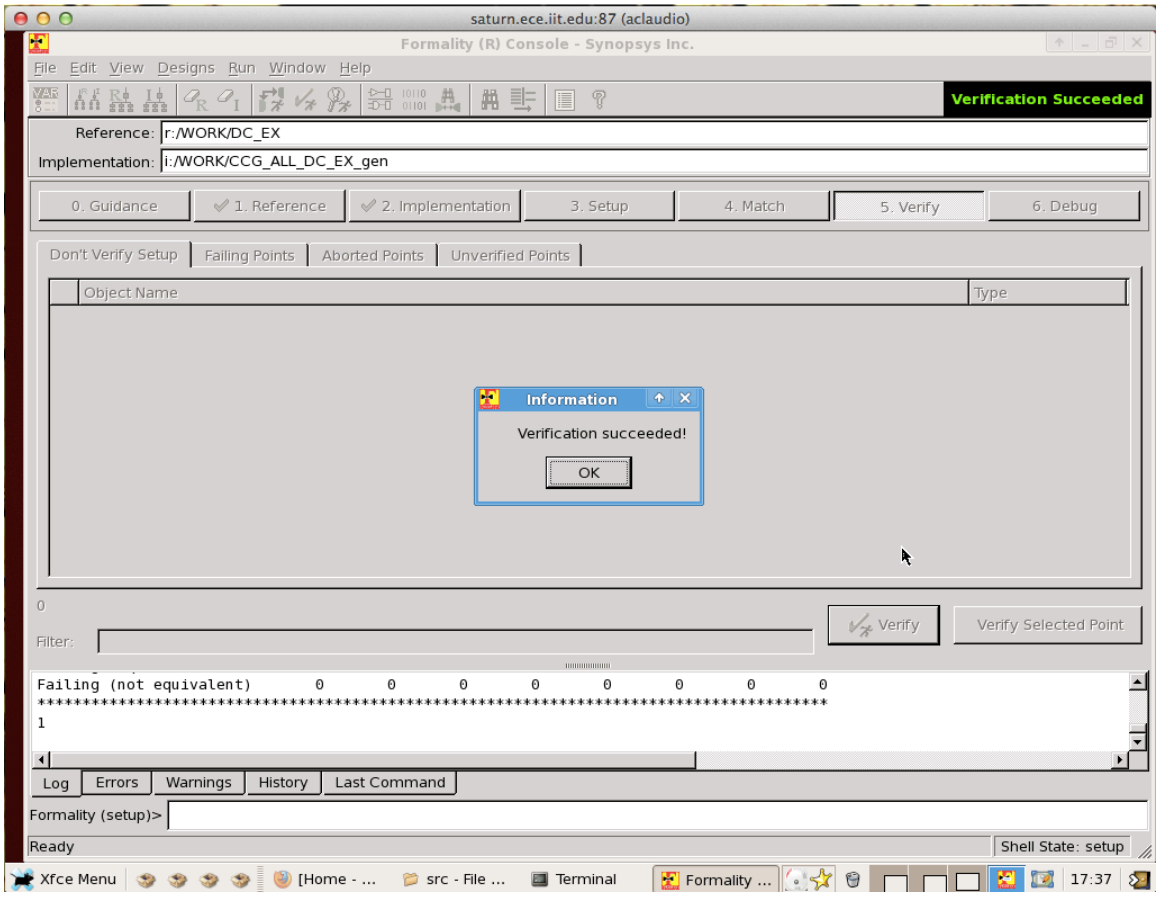
Automatic Clock Gating Insertion.

Using Power Compiler, the insertion of the Clock Gating will be done automatically during the synthesis procedure and as the results shows it is a good implementation, the power consumption have been reduce severally compared to the original one.

Power (mW)	Automatic Clock Gating Insertion
Total Dynamic	2.3668
Cell Leakage	0.1168189
Total	2.4836

Customized Clock Gating

In this implementation every possible flip-flop has been modified using the clock gating technique, always guaranteeing the functionality of the circuit using Formality Synopsys, as the following image shows this implementation still has the same purpose.



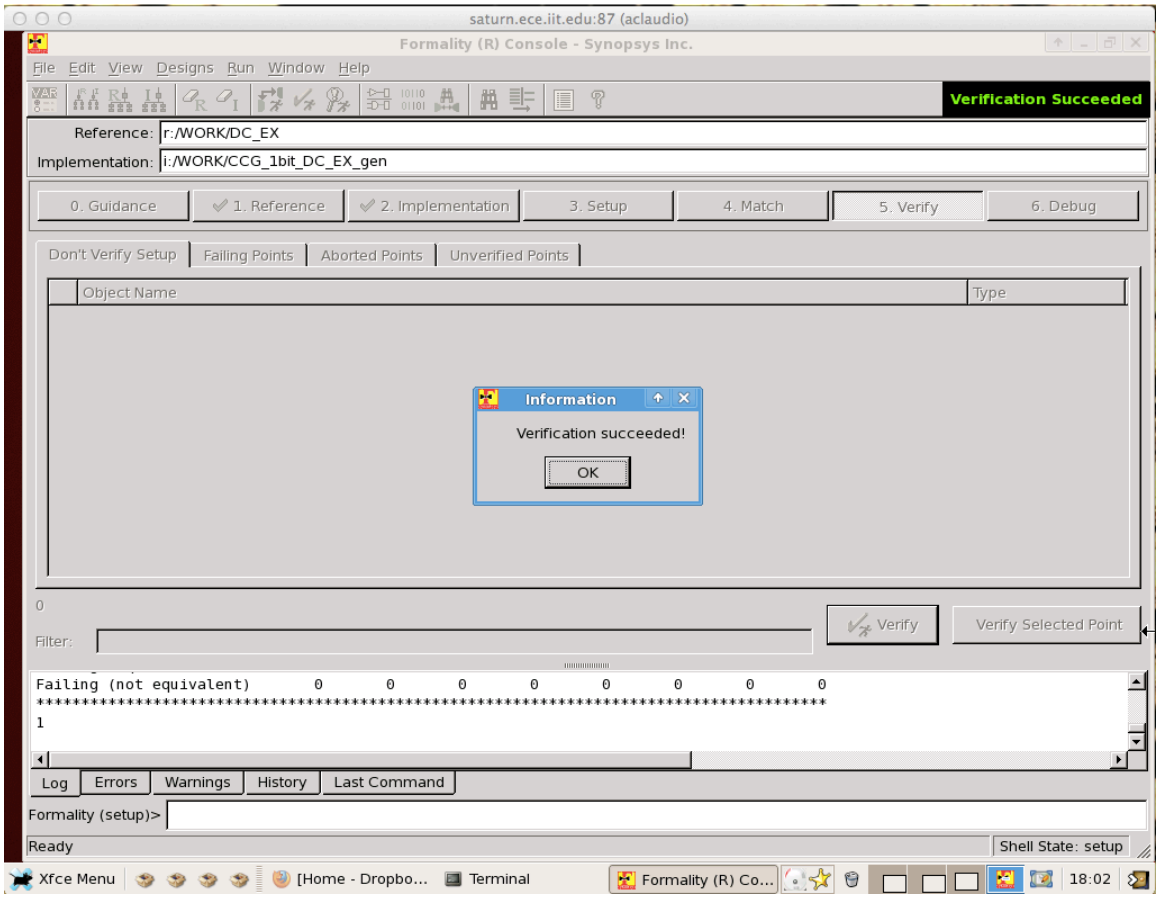
After guaranteeing this, it's time to find the power consumption. The results returned show that although it could seem that this will assure the lowest power consumption, another possible implementation could lead to even lower power consumption.

Power (mW)	**Customized Clock Gating**
Total Dynamic	2.3901
Cell Leakage	0.1164926
Total	2.5066

To this point is the best power consumption that have been found, but as future results will show, not always is the best implementation to do the clock gating technique to every flip-flop of the design because this technique works depending of the enable signal, as it will be explain after.

1bit Clock Gating.

This time only the 1bit input data flip-flop would be modified to see what the change of the consumption is.

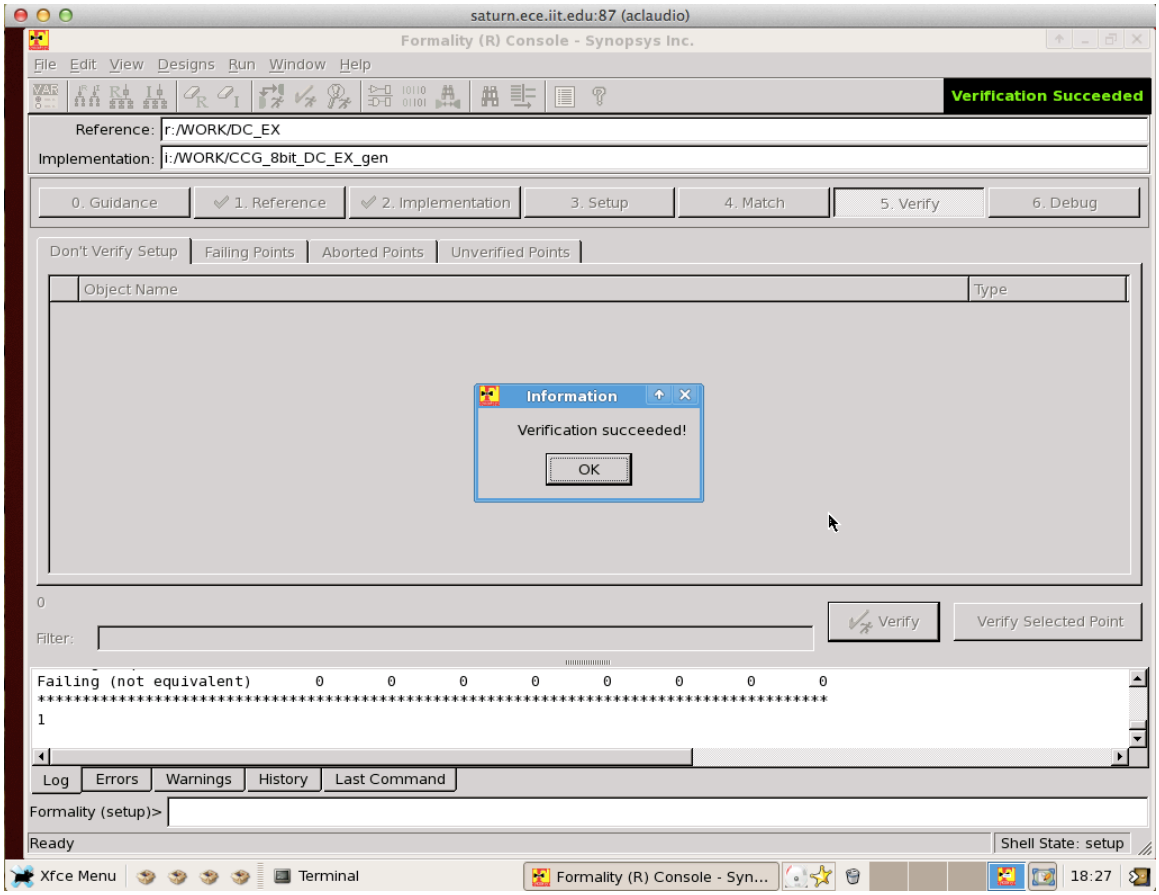


Power (mW)	**1 bit Clock Gating**
Total Dynamic	3.0794
Cell Leakage	0.1273181
Total	3.2067

This results returned tell us that although the change still being too small.

8bit Clock Gating.

The same will be done to the 8 bit input data of the flip-flops.

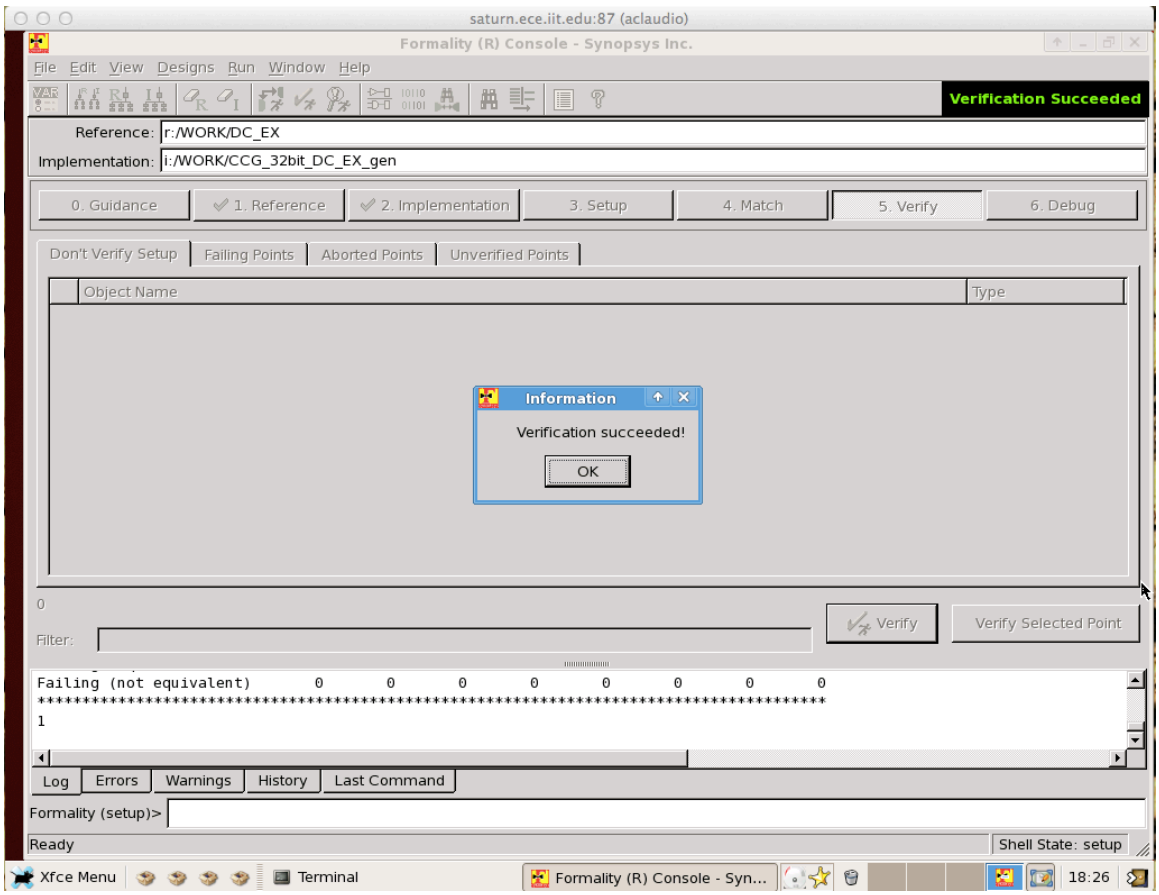


Power (mW)	**8 bit Clock Gating**
Total Dynamic	3.1719
Cell Leakage	0.124888
Total	3.29678

This results will be very important to make a final decision on which flip-flops will be modified, because as we can see the power consumption is greater than the 1 bit implementation what could be seem contradictory.

32bit Clock Gating.

The same procedure has been done with the 32 bit flip-flops.



Power (mW)	**32 bit Clock Gating**
Total Dynamic	2.3885
Cell Leakage	0.1173696
Total	2.5058

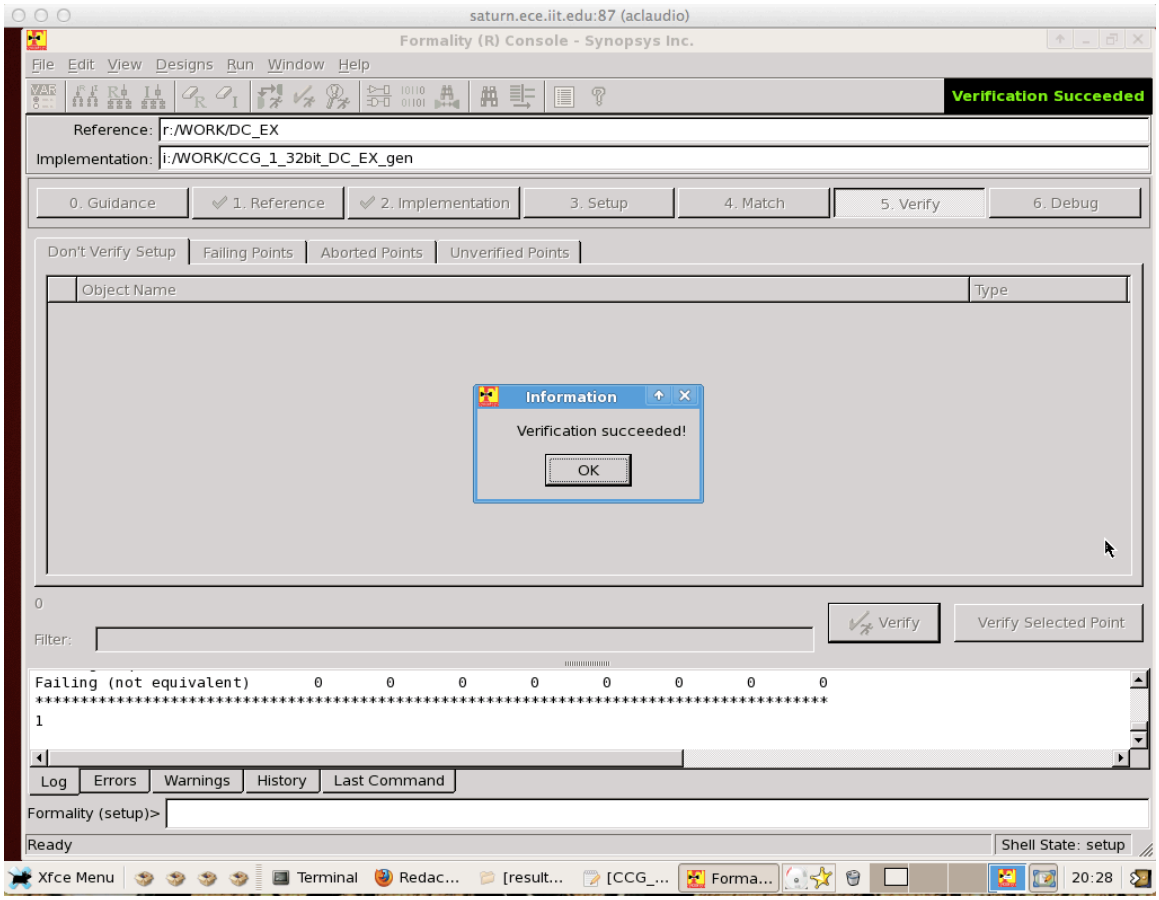
This other result is important because the power consumption is smaller than the one in which all the flip-flop have been modified.

Customized Final Clock Gating.

Power (mW)	**Customized Clock Gating**	**32 bit Clock Gating**	**8 bit Clock Gating**	**1 bit Clock Gating**
Total Dynamic	2.3901	2.3885	3.1719	3.0794
Cell Leakage	0.1164926	0.1173696	0.124888	0.1273181
Cell Leakage	2.5066	2.5058	3.29678	3.2067

If we compare the results of every implementation we can come up to this conclusions, that not only the number of flip-flops is important to do this implementation but also that the enable signal change enough to make it worth it, because the latch and the and gate also have power consumption, and with this table

we can notice that the flip-flop of the 8 bit input data does not worth it's modification because the consumption is very close to the original one and comparing the customized and the 32 bit we can have the final confirmation to this. Because of this the final customized processor will have modified only the 1 and 32 bit flip-flops modified.



Power (mW)	Customized Final Clock Gating
Total Dynamic	2.3595
Cell Leakage	0.1183295
Total	2.4778295

With this implementation we came up with the best design in terms of power consumption.