EAMTA 2019 - Advanced Digital Design

Andrew Parlane

14 de marzo de 2019

1. Objetivo

El objetivo de este curso estuvo implementar un micro básico en verilog, sintesizarle y hacer place and route, para generar las mascaras necesarios para fabricarle.

2. Micro en Verilog

El diseño del MICRO consiste en 5 módulos: Mux4, RegBank, ALU, Control y Top. Elige implementar todos con systemverilog. Todo el código es disponible en mi GitHUB.

Escribe testbenches bastantes completos para todos (excepto del Top) los módulos también usando systemverilog assertions. Usandos estos testbenches encontré varios errores y les arreglé. Por ejemplo al principio mi ALU estuvo equivocando unas operaciones, tuve que marcar las entradas y la salida como "signed".

Para compilar y ejecutar las simulaciones con VCS / SIMV ejecuto los siguientes commandos:

```
vcs -debug -sverilog -assert enable_diag ../../rtl/pkg/*.sv \
        ../../tb/OneBitAdder_tb.sv ../../rtl/*.sv
./simv -assert verbose -assert report +define+ASSERT_ON
vcs -debug -sverilog -assert enable_diag ../../rtl/pkg/*.sv \
        ../../tb/Mux4_tb.sv ../../rtl/*.sv
./simv -assert verbose -assert report +define+ASSERT_ON
vcs -debug -sverilog -assert enable_diag ../../rtl/pkg/*.sv
        ../../tb/RegisterBank_tb.sv ../../rtl/*.sv
./simv -assert verbose -assert report +define+ASSERT_ON
vcs -debug -sverilog -assert enable_diag ../../rtl/pkg/*.sv \
        ../../tb/ALU_tb.sv ../../rtl/*.sv
./simv -assert verbose -assert report +define+ASSERT_ON
```

Los sencillos (Mux4 y RegisterBank) funcionaron bien dando lo siguiente

```
    Chronologic VCS simulator copyright 1991-2017
Contains Synopsys proprietary information.
Compiler version M-2017.03-SP2-11_Full64;
Runtime version M-2017.03-SP2-11_Full64;
Mar 14 13:37 2019
Summary: 1 assertions, 0 with attempts, 0 with failures
          V C S   S i m u l a t i o n   R e p o r t
Time: 4000 ns
CPU Time:      0.200 seconds;       Data structure size:   0.0Mb
Thu Mar 14 13:37:52 2019

    Chronologic VCS simulator copyright 1991-2017
Contains Synopsys proprietary information.
Compiler version M-2017.03-SP2-11_Full64;
Runtime version M-2017.03-SP2-11_Full64;
Mar 14 13:38 2019
$stop at time 460 Scope: RegisterBank_tb File: ../../tb/RegisterBank_tb.sv Line: 71
ucli% q
Summary: 6 assertions, 0 with attempts, 0 with failures
```

```
          V C S   S i m u l a t i o n   R e p o r t
Time: 460 ns
CPU Time:      0.230 seconds;       Data structure size:   0.0Mb
Thu Mar 14 13:38:37 2019
```

Tuve unos problemas con mis assertions en los testbenches más complicados y no tuve tiempo depurarles, pero funcionaron bien en modelsim dando me lo siguiente:

```
# ----------------------------------------------------------------------
# Name                           File(Line)                Failure Pass
#
# ----------------------------------------------------------------------
# /OneBitAdder_tb/.../immed__21 OneBitAdder_tb.sv(21)       0     8

# ----------------------------------------------------------------------
# Name                            File(Line)                  Failure Pass
#
# ----------------------------------------------------------------------
# /RegisterBank_tb/.../immed__42 RegisterBank_tb.sv(42)       0     1
# /RegisterBank_tb/.../immed__48 RegisterBank_tb.sv(48)       0     1
# /RegisterBank_tb/.../immed__54 RegisterBank_tb.sv(54)       0     1
# /RegisterBank_tb/.../immed__60 RegisterBank_tb.sv(60)       0     1
# /RegisterBank_tb/.../immed__66 RegisterBank_tb.sv(66)       0     1
# /RegisterBank_tb/.../immed__69 RegisterBank_tb.sv(69)       0     1

# ----------------------------------------------------------------
# Name                            File(Line)          Failure Pass
#
# ----------------------------------------------------------------
# /ALU_tb/errorMeansOutIsMinus1  ALU_tb.sv(41)        0  8160
# /ALU_tb/nvalidError            ALU_tb.sv(49)        0  4990
# /ALU_tb/invalidOpError         ALU_tb.sv(56)        0  3685
# /ALU_tb/div0Error              ALU_tb.sv(64)        0  1285
# /ALU_tb/errorOnlyIfActualError ALU_tb.sv(74)        0  8160
# /ALU_tb/zeroTest               ALU_tb.sv(82)        0 204841
# /ALU_tb/.../addAssert          ALU_tb.sv(94)        0 65536
# /ALU_tb/.../subAssert          ALU_tb.sv(105)       0 65536
# /ALU_tb/.../mulAssert          ALU_tb.sv(116)       0 65536
# /ALU_tb/.../divAssert          ALU_tb.sv(128)       0 65280

# ----------------------------------------------------------------
# Name                        File(Line)              Failure Pass
#
# ----------------------------------------------------------------
# /Control_tb/muxSelA        Control_tb.sv(71)        0 984303
# /Control_tb/muxSelB        Control_tb.sv(78)        0 984303
# /Control_tb/opDecode       Control_tb.sv(85)        0 984303
# /Control_tb/dataValidChec  Control_tb.sv(94)        0 968674
# /Control_tb/dataInOneTick  Control_tb.sv(101)       0 328054
# /Control_tb/aluInOneTick   Control_tb.sv(108)       0 328054
```

```
# /Control_tb/aluoutOneTick  Control_tb.sv(115)       0 322868
# /Control_tb/dataInToAluIn  Control_tb.sv(122)       0 328054
# /Control_tb/AluInToAluOut  Control_tb.sv(129)       0 322869
# /Control_tb/AluOutToDataIn Control_tb.sv(136)       0 317722
# /Control_tb/resetCheck     Control_tb.sv(143)       0 15561
# /Control_tb/onlyOneRegEn1  Control_tb.sv(150)       0 333283
# /Control_tb/onlyOneRegEn2  Control_tb.sv(157)       0 328054
# /Control_tb/onlyOneRegEn3  Control_tb.sv(164)       0 322869
```

Más tarde en la etapa de sintesís estuve fallando timing con la ruta critical desde una de las entradas DIN del ALU hasta la salida ZERO. Al principio estuve generando este señal con: .ªssign zero = (out == 0);". Para arreglar el timing cambié el código para usar las entradas en el caso de MUL y DIV. Eliminando la dependencia en el resultado. Con esto pude aprobar timing.

3. Sintesís

Mi scripts final están en la carpeta synopsys/synth en mi GitHUB.
Hice lo siguiente:

- Configurar los variables del entorno.
- Analizar el fuente código.
- Elaborate, link y check_design.
- Configurar los constraints.
- compile_ultra.
- Clock gating.
- Optimización de registros.
- Incremental compilación.
- Optimize netlist por area.
- Change_names.
- Write_ddc.

El clock gating reduce la potencia un un poco, en el orden de 4 %. La optimización de registros se reduce otra 2 %. La compilcaión incremental se reduce potencia hasta 70 % del inicial.

Etapa	Area	Potencia (mW)
Inicial	19682	52.97
Gate Clocks	18360	50.69
Optimiza Registros	18398	49.66
Compilación Incremental	18398	36.72
Optimiza Netlist por Area	17819	36.72

4. Place and Route

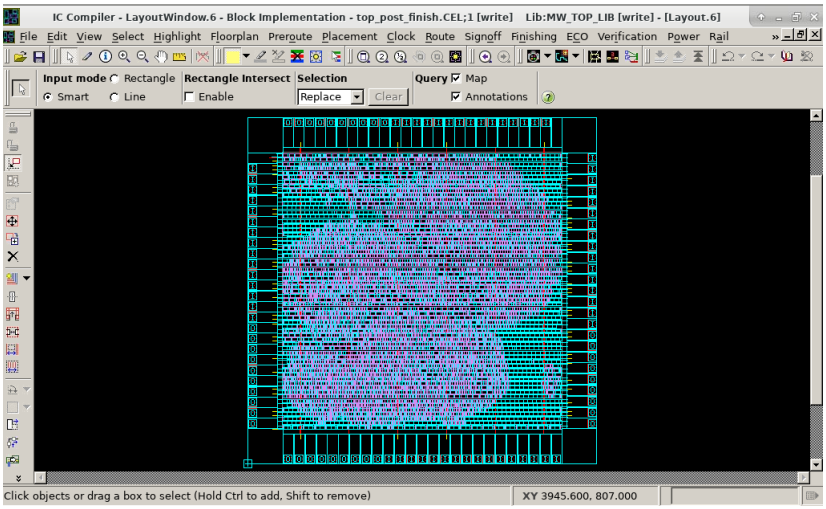
Ejecuté los commandos como dado en el LAB, y experiminté con el GUI un poco para ver que hace cada commando.

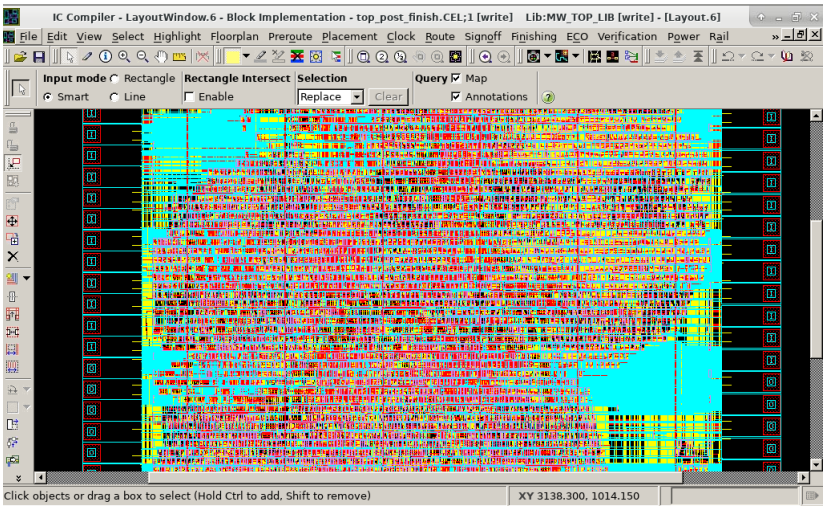
Al final tengo unos errores con verify_lvs que debería depurar, pero desafortunadamente no tuve más tiempo.

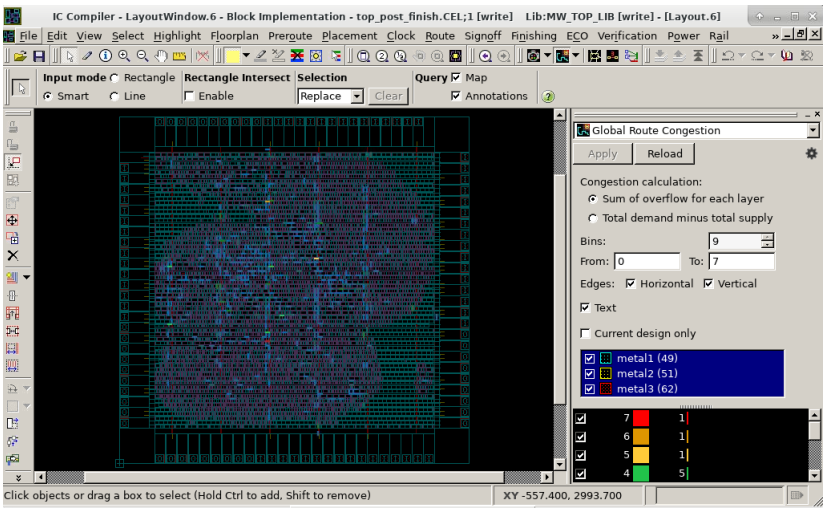
```
ERROR : OUTPUT PortInst u_dout_low[5]_pad DataIn doesn't connect to any net.
```

```
ERROR : OUTPUT PortInst u_dout_low[5]_pad DataInB doesn't connect to any net.

...

ERROR : There are 2 nets short together.
n8116 (205576).
vdd! (149031).
```

Aquí hay unas captura de pantalla del diseño final:






IC Compiler - LayoutWindow.6 - Block Implementation - top_post_finish.CEL;1 [write] Lib:MW_TOP_LIB [write] - [Layout.6]
File Edit View Select Highlight Floorplan Preroute Placement Clock Route Signoff Finishing ECO Verification Power Rail
Input mode Rectangle Smart Line
Rectangle Intersect Enable
Selection Replace Clear
Query Map Annotations
Global Route Congestion
Apply Reload
Congestion calculation:
Sum of overflow for each layer
Total demand minus total supply
Bins: 9
From: 0 To: 7
Edges: Horizontal Vertical
Text
Current design only
metal1 (49)
metal2 (51)
metal3 (62)
7 1
6 1
5 1
4 5
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove)
XY -557.400, 2993.700