

# ASSIGNMENT 6

SAMAKSH GUPTA

2019200

## PART 2

### 1. Running Yosys

```
samaksh19200@edaserver3:~/Ass6/part1$ yosys
```

```
/-----  
| yosys -- Yosys Open SYnthesis Suite  
|
```

```
| Copyright (C) 2012 - 2020 Claire Xenia Wolf <claire@yosyshq.com>  
|
```

```
| Permission to use, copy, modify, and/or distribute this software for any  
| purpose with or without fee is hereby granted, provided that the above  
| copyright notice and this permission notice appear in all copies.  
|
```

```
| THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES  
| WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF  
| MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR  
| ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES  
| WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN  
| ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF  
| OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.  
|
```

```
|-----  
| Yosys 0.11+10 (git sha1 4871d8f19, clang 6.0.0-1ubuntu2 -fPIC -Os)  
|
```

```
yosys> █
```

## 2. Netlists for the two designs

### Design 1

```
//`include "fast.lib"

module des1(input in, input clk, output out);

//CLKINVSX1
//AND2X4
//XNOR2X1
//SDFFX1
//BUFX2

wire in11; wire in12; wire q1; wire q2; wire d; wire q3;

CLKINVSX1 inv1(in, in11);
BUFX2 buf1(in, in12);

SDFFX1 ff1(.CK(clk), .D(in11), .Q(q1));
SDFFX1 ff2(.CK(clk), .D(in12), .Q(q2));

XNOR2X1 xnor1(q1,q2,d);
SDFFX1 ff3(.CK(clk), .D(d), .Q(q3));

CLKINVSX1 inv2(q3, out);

endmodule
```

### Design 2

```
//`include "fast.lib"

module des2(input in, input clk, output out);

//CLKINVSX1
//AND2X4
//XNOR2X1
//SDFFX1
//BUFX2

wire in11; wire in12; wire q1; wire q2; wire d; wire q3;

CLKINVSX1 inv1(in, in11);
BUFX2 buf2(in, in12);

SDFFX1 ff1(.CK(clk), .D(in11), .Q(q1));
SDFFX1 ff2(.CK(clk), .D(in12), .Q(q2));

AND2X4 and1(q1,q2,d);
SDFFX1 ff3(.CK(clk), .D(d), .Q(q3));

CLKINVSX1 inv2(q3, out);

endmodule
~
```

## CEC Script

```
#Source: Google Classroom Assignment-6 (Given)

#gold design
read_verilog des1.v
read_liberty -ignore_miss_func -ignore_miss_data_latch fast.lib
#If we don't use the above method we get miter design instantiation error.
design -save lib
prep -flatten -top des1
splitnets -ports;;
design -stash gold

#####

#gate design
read_verilog des2.v
read_liberty -ignore_miss_func -ignore_miss_data_latch fast.lib
#If we don't use the above method we get miter design instantiation error.
prep -flatten -top des2
splitnets -ports;;
design -stash gate

#####

design -copy-from gold -as gold des1
design -copy-from gate -as gate des2

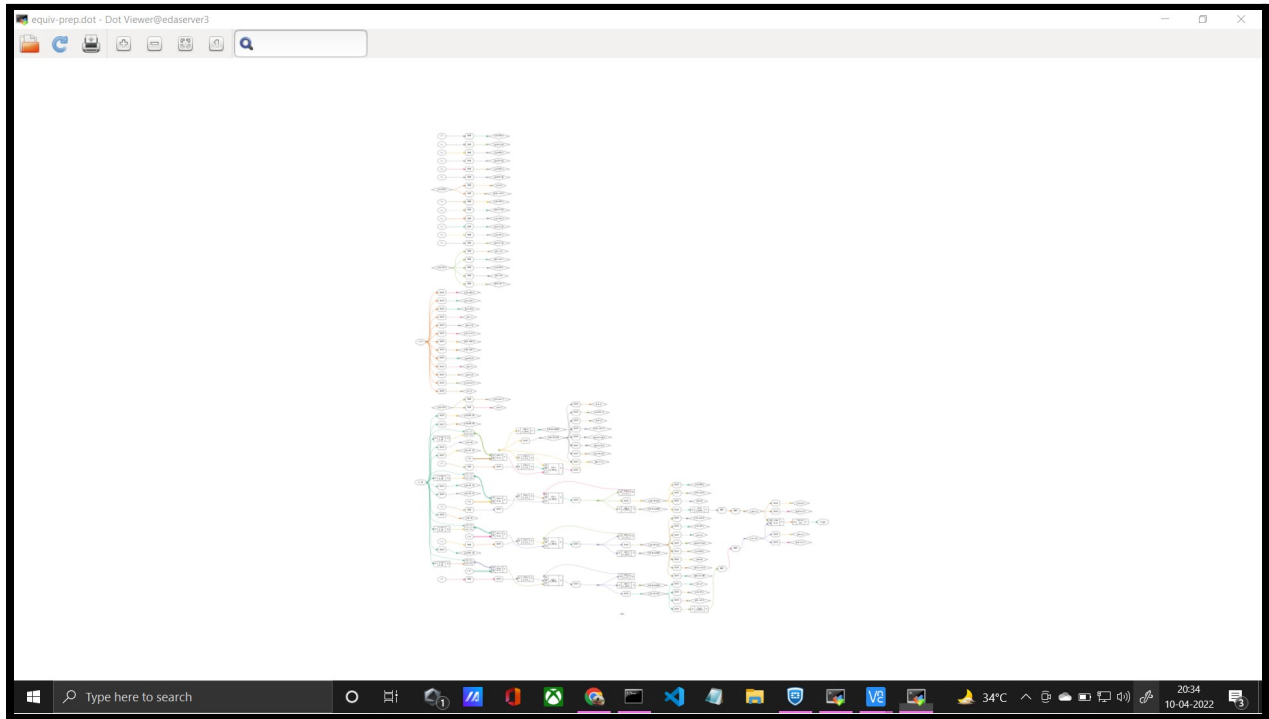
#prove combinational equivalence checking
miter -equiv -flatten gold gate miter
prep -flatten -top miter

#design -save lib
dfflibmap -liberty fast.lib
abc -liberty fast.lib
techmap -map %lib
clk2fflogic

show -prefix equiv-prep -colors 1 -stretch
sat -all -verify -tempinduct -prove trigger 0 -set-at 1 in_in 0

#####
```

# RESULT OF RUNNING THE SEC SCRIPT



```
Final constraint equation: \in_in = 1'0
Imported 28 cells to SAT database.
Import set-constraint from init attribute: $auto$clk2fflogic.cc:185:execute$7391 = 1'1
Import set-constraint from init attribute: $auto$clk2fflogic.cc:185:execute$7381 = 1'1
Import set-constraint from init attribute: $auto$clk2fflogic.cc:185:execute$7371 = 1'1
Import set-constraint from init attribute: $auto$clk2fflogic.cc:185:execute$7361 = 1'1
Final init constraint equation: { $auto$clk2fflogic.cc:185:execute$7361 $auto$clk2fflogic.cc:185:execute$7371 $auto$clk2fflogic.cc:185:execute$7381 $auto$clk2fflogic.cc:185:execute$7391 } = 4'b1111
Import proof-constraint: \trigger = 1'0
Final proof equation: \trigger = 1'0
```

```
[base case 1] Solving problem with 119 variables and 284 clauses..
SAT temporal induction proof finished - model found for base case: FAIL!
```

Proof failed!

Time	Signal Name	Dec	Hex	Bin
init	\$auto\$clk2fflogic.cc:173:execute\$7359	0	0	0
init	\$auto\$clk2fflogic.cc:173:execute\$7369	1	1	1
init	\$auto\$clk2fflogic.cc:173:execute\$7379	1	1	1
init	\$auto\$clk2fflogic.cc:173:execute\$7389	0	0	0
init	\$auto\$clk2fflogic.cc:185:execute\$7361	1	1	1
init	\$auto\$clk2fflogic.cc:185:execute\$7371	1	1	1
init	\$auto\$clk2fflogic.cc:185:execute\$7381	1	1	1
init	\$auto\$clk2fflogic.cc:185:execute\$7391	1	1	1
init	\$auto\$clk2fflogic.cc:205:execute\$7365	0	0	0
init	\$auto\$clk2fflogic.cc:205:execute\$7375	0	0	0
init	\$auto\$clk2fflogic.cc:205:execute\$7385	0	0	0
init	\$auto\$clk2fflogic.cc:205:execute\$7395	0	0	0
1	\in_clk	0	0	0
1	\in_in	0	0	0
1	\trigger	1	1	1

ERROR: Called with -verify and proof did fail!

## SEC Script

```
##Source: Google Classroom Assignment-6 (Given)
```

```
#gold design
read_verilog des1.v
read_liberty -lib fast.lib
read_liberty -ignore_miss_func -ignore_miss_data_latch fast.lib
prep -flatten -top des1
splitnets -ports;;
design -stash gold
```

```
#####
```

```
#gate design
read_verilog des2.v
read_liberty -lib fast.lib
read_liberty -ignore_miss_func -ignore_miss_data_latch fast.lib
prep -flatten -top des2
splitnets -ports;;
design -stash gate
```

```
#####
```

```
design -copy-from gold -as gold des1
design -copy-from gate -as gate des2
```

```
#####
```

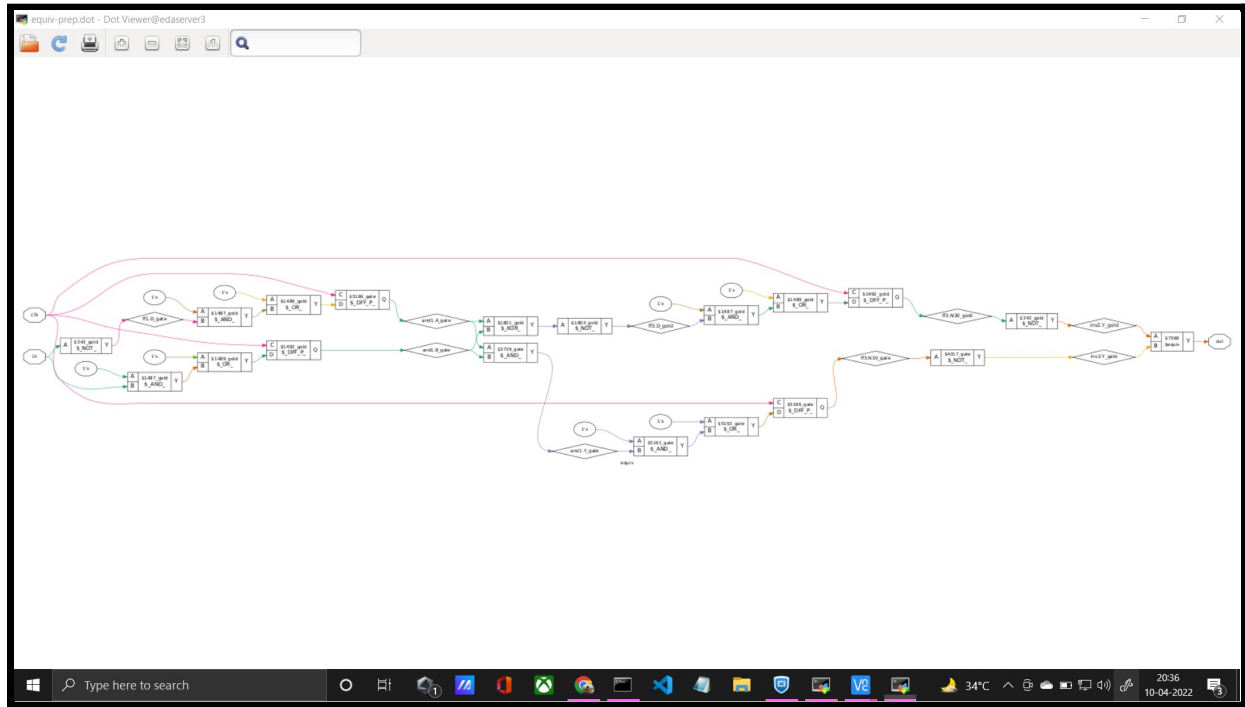
```
#prove sequential equivalence checking
equiv_make gold gate equiv
prep -flatten -top equiv
opt_clean -purge

show -prefix equiv-prep -colors 1 -stretch
equiv_induct -seq 5
equiv_status -assert
```

```
#####
```

```
ms
```

## RESULT OF RUNNING THE SEC SCRIPT



```
Number of wires:          90
Number of wire bits:      90
Number of public wires:   81
Number of public wire bits: 81
Number of memories:       0
Number of memory bits:    0
Number of processes:      0
Number of cells:          19
$ AND_                     5
$ DFF_P                    4
$ NOT_                     4
$ OR_                      4
$ XOR_                     1
$equiv                    1

12.13. Executing CHECK pass (checking for obvious problems).
Checking module equiv...
Found and reported 0 problems.

13. Executing OPT CLEAN pass (remove unused cells and wires).
Finding unused cells or wires in module \equiv..
Removed 0 unused cells and 69 unused wires.
<suppressed ~69 debug messages>

14. Generating Graphviz representation of design.
Writing dot description to 'equiv-prep.dot'.
Dumping module equiv to page 1.
Exec: { test -f 'equiv-prep.dot.pid' && fuser -s 'equiv-prep.dot.pid' 2> /dev/null; } || ( echo $$ >&3; exec xdot 'equiv-prep.dot'; ) 3> 'equiv-prep.dot.pid' &

15. Executing EQUIV_INDUCT pass.
Found 1 unproven $equiv cells in module equiv:
  Proving existence of base case for step 1. (186 clauses over 78 variables)
  Proving induction step 1. (409 clauses over 166 variables)
  Proof for induction step holds. Entire workset of 1 cells proven!
Proved 1 previously unproven $equiv cells.

16. Executing EQUIV_STATUS pass.
Found 1 $equiv cells in equiv:
  Of those cells 1 are proven and 0 are unproven.
  Equivalence successfully proven!

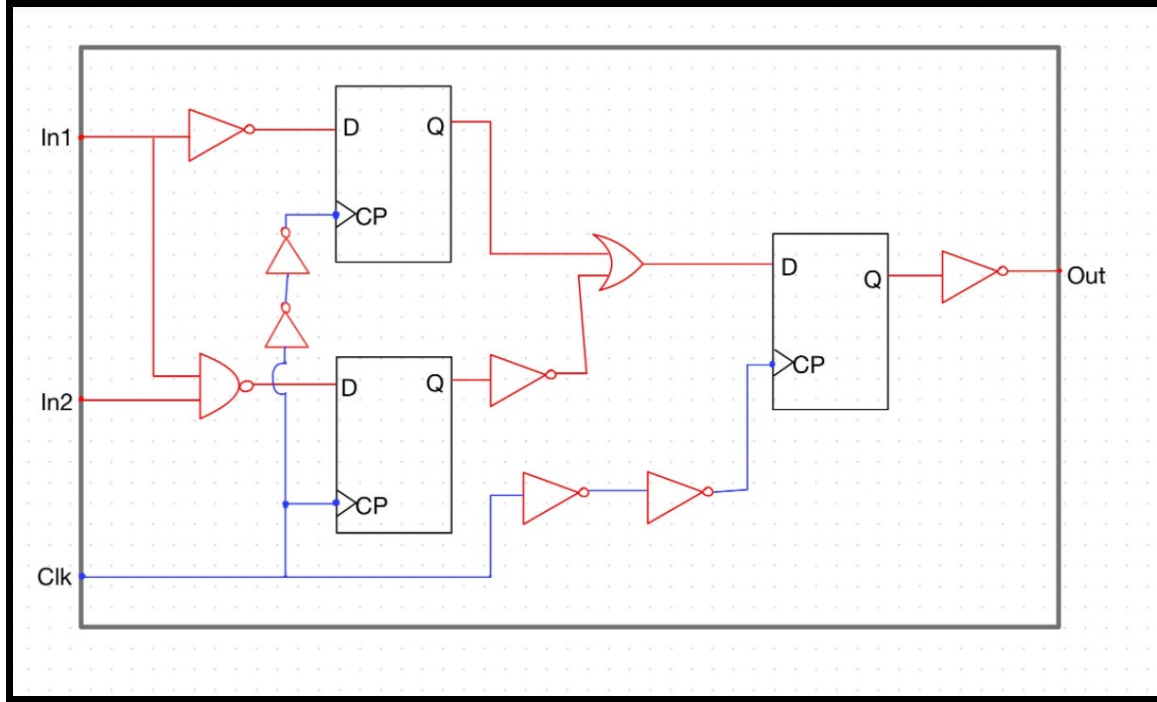
yosys>
```

Equivalence Successfully Proved



# PART 1

## Design



## Netlist

```
top.v
module top(input clk, input in1, input in2, output out);

wire clk1; wire clk2; wire clk3; wire clk4;
wire d1; wire d2; wire d3;
wire q1; wire q2; wire q3; wire q4;

CLKINVTX1 Inv1(clk, clk1);
CLKINVTX1 Inv2(clk1, clk2);
CLKINVTX1 Inv3(clk, clk3);
CLKINVTX1 Inv4(clk3, clk4);

CLKINVTX1 Inv5(in1, d1);
NAND2TX1 Nand1(in1, in2, d2);

SDFFX1 Ff1(.CK(clk2), .D(d1), .Q(q1));
SDFFX1 Ff2(.CK(clk), .D(d2), .Q(q2));

CLKINVTX1 Inv6(q2, q3);
NOR2TX1 Nor1(q1,q3,d3);

SDFFX1 Ff3(.CK(clk4), .D(d3), .Q(q4));
CLKINVTX1 Inv7(q4, out);

endmodule
```

```
RC-Corner PostRoute Cap Factor      : 1
RC-Corner PostRoute XCap Factor      : 1
Current (total cpu=0:00:08.8, real=0:00:09.0, peak res=856.6M, current mem=816.7M)
INFO (CTE): Constraints read successfully.
Ending "Constraint file reading stats" (total cpu=0:00:00.0, real=0:00:00.0, peak res=835.9M, current mem=835.9M)
Current (total cpu=0:00:08.9, real=0:00:09.0, peak res=856.6M, current mem=835.9M)
**WARN: (IMPESI-3468): User needs to specify -equivalent_waveform_model propagation first before specifying -waveform_compression mode accurate|clock_detailed .
AAE DB initialization (MEM=1094.76 CPU=0:00:00.0 REAL=0:00:00.0)
#####
# Design Name: rtl_module
# Design Mode: 65nm
# Analysis Mode: MMMC OCV
# Parasitics Mode: No SPEF/RCDB
# Signoff Settings: SI Off
#####
Start delay calculation (fullDC) (1 T). (MEM=950.77)
End delay calculation. (MEM=987.062 CPU=0:00:00.0 REAL=0:00:00.0)
End delay calculation (fullDC). (MEM=979.062 CPU=0:00:00.1 REAL=0:00:00.0)
INFO: Path Based Analysis (PBA) performed on total '30' paths
INFO: Path Based Analysis (PBA) performed on total '30' paths
INFO: Path Based Analysis (PBA) performed on total '30' paths
tempus l>
```

## Unable to get the license ... :(

```
samaksh19200@edaserver4:q1_1
File Edit View Search Terminal Help
Check your Tempus Timing Signoff Solution license status.

[samaksh19200@edaserver4 q1_1]$ /cadence/SSV201/bin/tempus -nowin -file sta_1.tcl

Cadence Tempus(TM) Timing Signoff Solution.
Copyright 2020 Cadence Design Systems, Inc. All rights reserved worldwide.

Version:      v20.10-p003_1, built Wed Apr 29 15:56:37 PDT 2020
Options:      -nowin -file sta_1.tcl
Date:         Mon Apr 11 00:24:32 2022
Host:         edaserver4 (x86_64 w/Linux 2.6.32-754.35.1.el6.x86_64) (18cores*
72cpus*Intel(R) Xeon(R) Gold 6354 CPU @ 3.00GHz 39936KB)
OS:           Red Hat Enterprise Linux Server release 6.10 (Santiago)

License:
Initialization status was 'ERROR (LMC-01902): License call failed. The license server search path is defined as <none>. Can't find license file.

Run 'lic_error LMC-01902' for more information.' (-902)
Fail to find any Tempus Timing Signoff Solution license...
Check your Tempus Timing Signoff Solution license status.

[samaksh19200@edaserver4 q1_1]$
```