



CS5120 VLSI System Design, Spring 2025

Walkthrough for Verilog Simulation and Synthesis

黃稚存

Chih-Tsun Huang

cthuang@cs.nthu.edu.tw



國立清華大學
NATIONAL TSING HUA UNIVERSITY

資訊工程學系
Computer Science

Lecture 10



聲明

- ◎ 本課程之內容 (包括但不限於教材、影片、圖片、檔案資料等)，僅供修課學生個人合理使用，非經授課教師同意，不得以任何形式轉載、重製、散布、公開播送、出版或發行本影片內容 (例如將課程內容放置公開平台上，如 Facebook, Instagram, YouTube, Twitter, Google Drive, Dropbox 等等)。如有侵權行為，需自負法律責任。



Outline

⦿ Review CT Verilog Series

- ◆ 01: Fundamental Concepts for Verilog HDL
- ◆ 02: My Very First Verilog Coding
 - ▣ 02-1: Update about Waveform Format
- ◆ You should check the CT Verilog Series before moving on

⦿ Preliminary Synthesis



Preliminary Synthesis



Synopsys Synthesis Tools

⦿ Design Compiler

- ◆ Constraint-driven logic synthesizer

⦿ Design Vision

- ◆ GUI interface

⦿ DesignWare

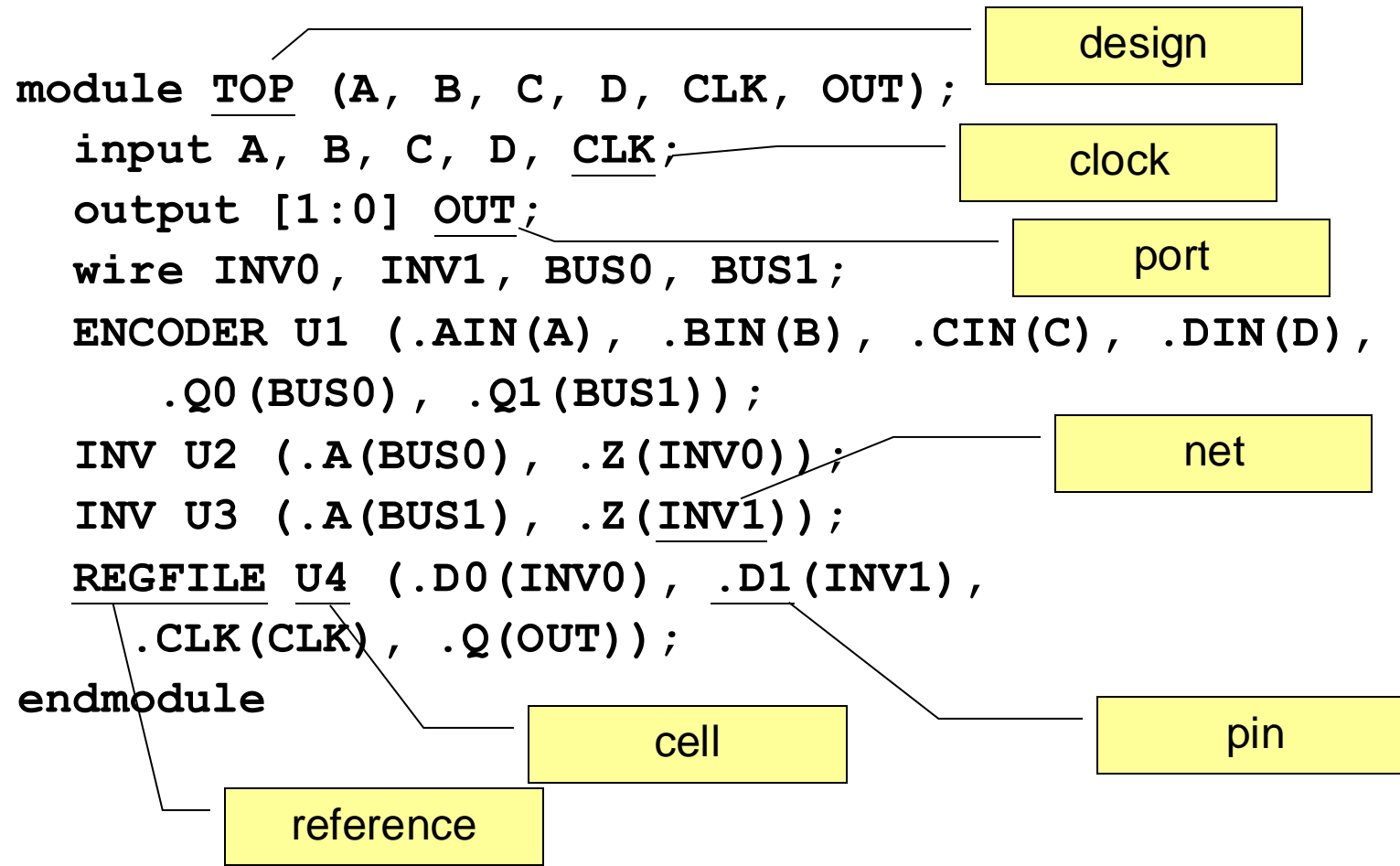
- ◆ DesignWare library
 - ▣ Pre-designed synthesizable logic blocks



Concept of Synthesis Objects

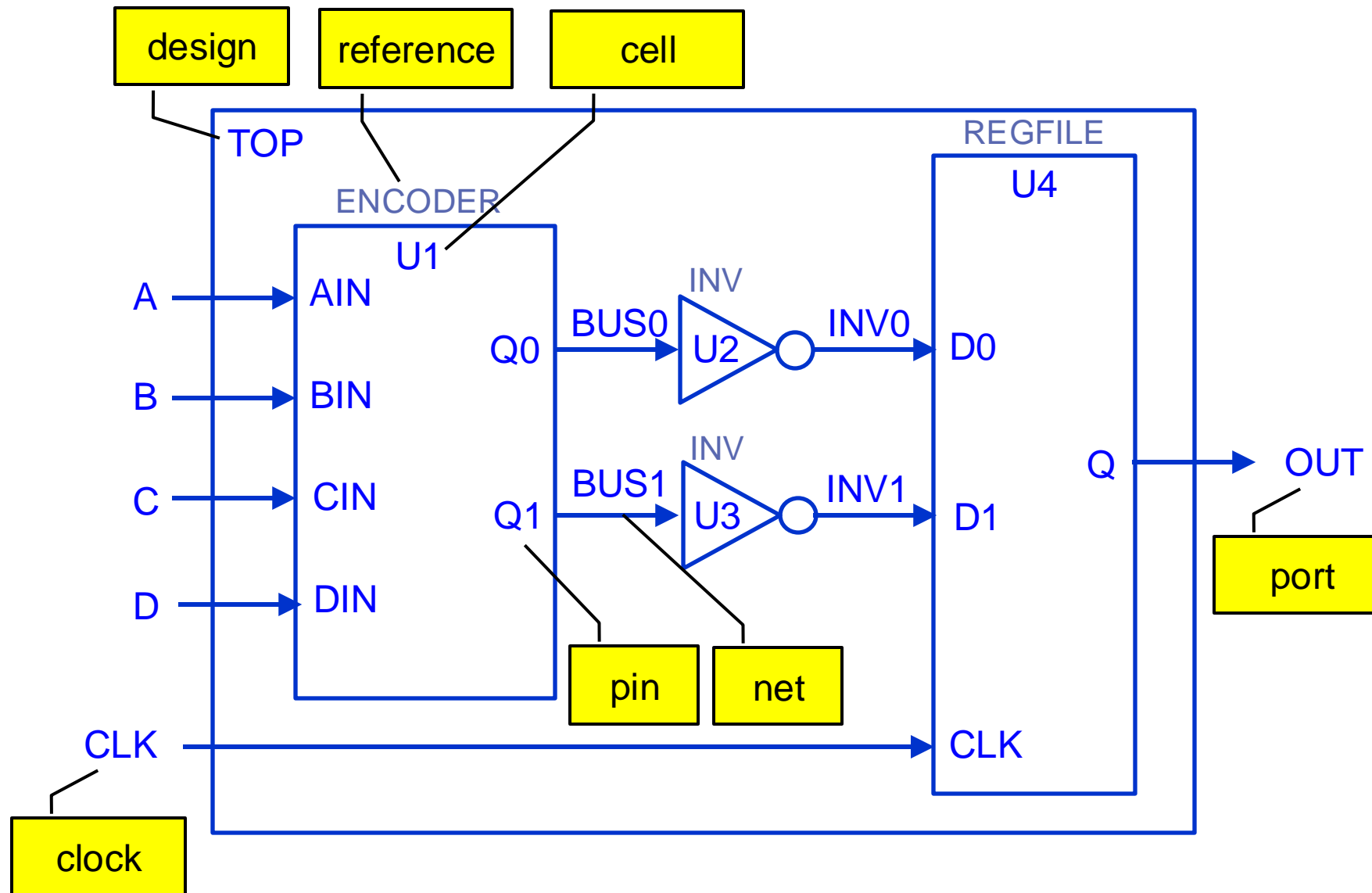


Design Objects in Verilog Perspective





Design Objects in Schematic Perspective





Design Objects

- ⦿ Design: a circuit with one or more logical functions
- ⦿ Cell: an instantiation of a design within another design
- ⦿ Reference: the original module of a cell
- ⦿ Port: an input/output of a design
- ⦿ Pin: an input/output of a cell
- ⦿ Net: a wire connecting port to pin and/or pin to other pin
- ⦿ Clock: clock source to a port or pin



A Quick Synthesis Guide

- » Preliminary Synthesis
- » Post-Synthesis Simulation



Before You Start

- Cell library
- The setting of cell library is defined in `.synopsys_dc.setup`
 - Put this file in your working directory
- Otherwise, you will get this warning:

Warning: Can't read Link_library file 'your_library.db'. (UID-3)



Template of .synopsys_dc.setup (CBDK IC Contest)

Virtual Library

```
# 1. Virtual Library Setup for NTHU VLSI/CAD Lab
# 2. Rename synopsys_dc.setup to .synopsys_dc.setup
#    and put it in the working directory.
set company "NTHU"
set designer "astroboy"
set search_path
    "/theda21_2/CBDK_IC_Contest/cur/SynopsysDC/db $search_path"
set link_library      "slow.db fast.db dw_foundation.sldb"
set target_library    "slow.db fast.db"
set symbol_library    "generic.sdb"
set synthetic_library "dw_foundation.sldb"
```



Template of .synopsys_dc.setup (GPDK045)

```
# 1. Virtual Library Setup for NTHU VLSI/CAD Lab
# 2. Rename synopsys_dc.setup to .synopsys_dc.setup
#    and put it in the working directory.
set company "NTHU"
set designer "Tony Stark"
set search_path
    "/theda21_2/library/GPDK045/cur/gsclib045/db/ $search_path"
set target_library "slow_vdd1v2_basicCells.db"
set link_library "slow_vdd1v2_basicCells.db dw_foundation.sldb"
set symbol_library "generic.sdb"
set synthetic_library "dw_foundation.sldb"
```



Synthesis Tool

◎ Synopsys Design Compiler

```
$ dc_shell
```

◎ Design Vision (GUI, Graphic User Interface)

```
$ design_vision
```

Or

```
$ dc_shell
```

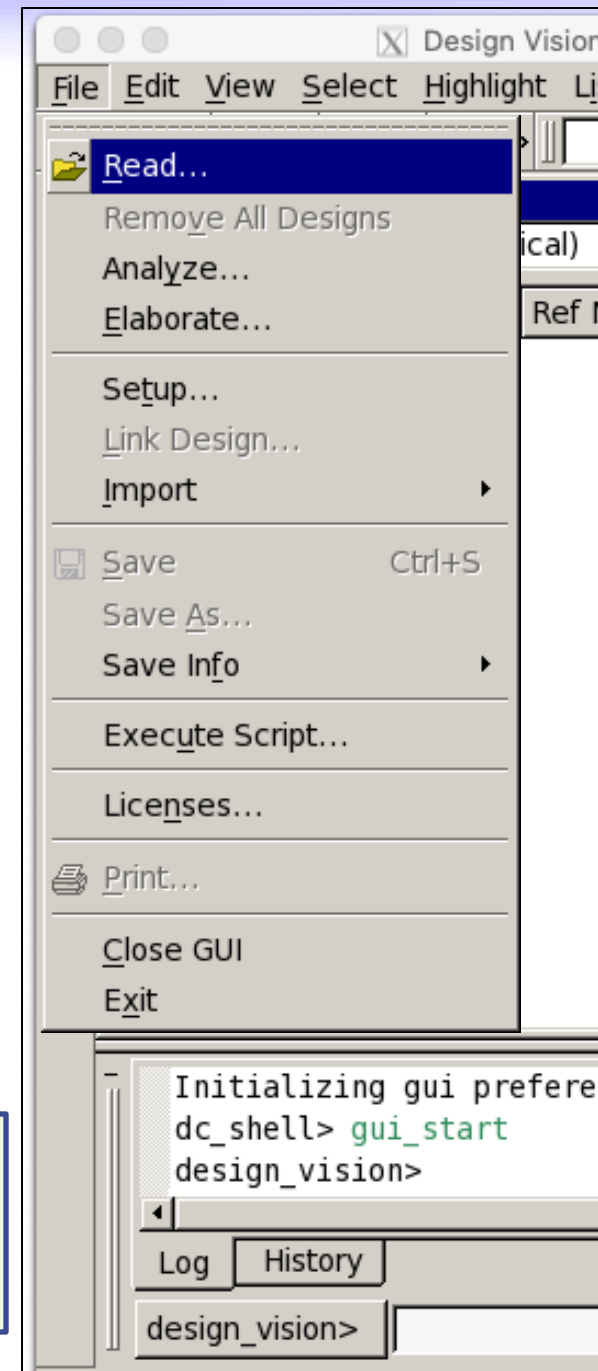
```
dc_shell> gui_start
```

Reading the Source File

- » File » Read
- Read in the design
- Many different formats, popular ones are:
 - ◆ Verilog: *.v
 - ◆ System Verilog: *.sv
 - ◆ VHDL: *.vhd
 - ◆ EDIF
 - ◆ Synopsys internal formats
 - ❑ DB (binary): *.db
 - ❑ Enhanced DB: *.ddc (including netlist and constraints)

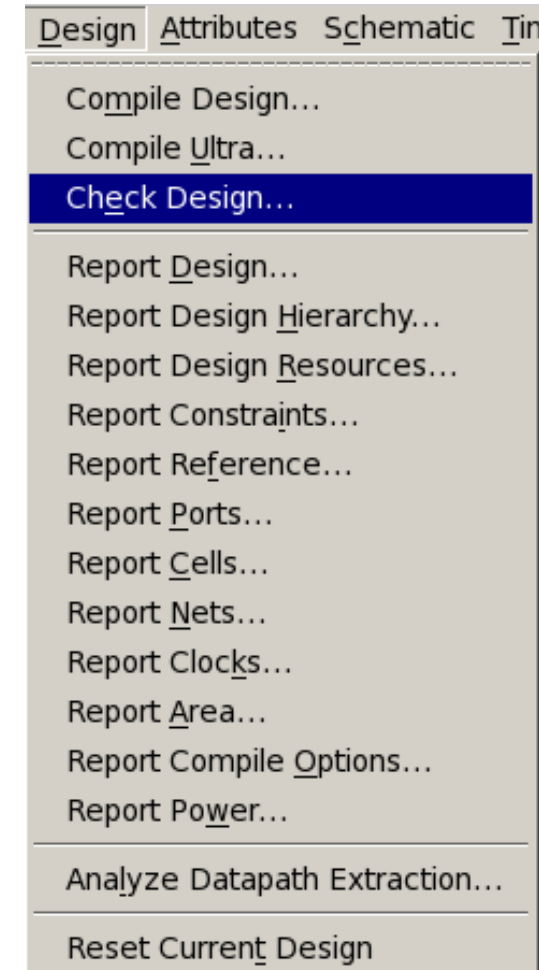
TCL command:

```
> read_file -format verilog gcd.v
```



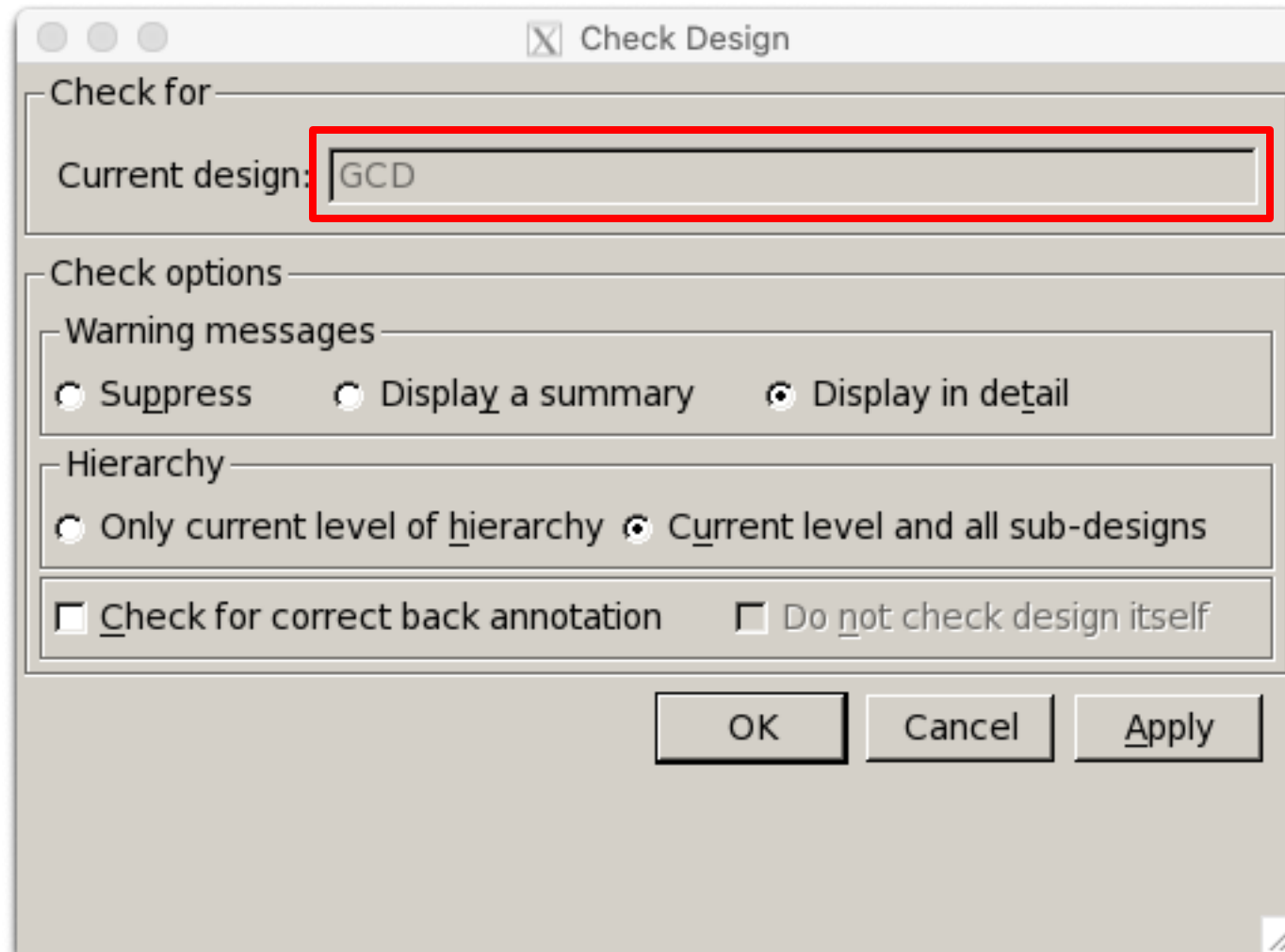
Checking the Design

- ⦿ » Design » Check Design
- ⦿ Check the design before optimization
- ⦿ TCL command
 - > `check_design`
- ⦿ Errors
 - ◆ E.g., syntax error, non-synthesizable code
 - ◆ Errors must be removed
- ⦿ Warning
 - ◆ E.g., floating (unconnected) output
 - ◆ Warnings need to be explained





Checking the Design



A screenshot of a 'Check Design' dialog box. The dialog has a title bar with a close button and the text 'Check Design'. It contains several sections: 'Check for' (empty), 'Current design:' (text box containing 'GCD', highlighted with a red rectangle), 'Check options' (expanded), 'Warning messages' (radio buttons for 'Suppress', 'Display a summary', and 'Display in detail', with 'Display in detail' selected), 'Hierarchy' (radio buttons for 'Only current level of hierarchy' and 'Current level and all sub-designs', with 'Current level and all sub-designs' selected), and two unchecked checkboxes: 'Check for correct back annotation' and 'Do not check design itself'. At the bottom are 'OK', 'Cancel', and 'Apply' buttons.

Check Design

Check for

Current design: GCD

Check options

Warning messages

☐ Suppress ☐ Display a summary ☒ Display in detail

Hierarchy

☐ Only current level of hierarchy ☒ Current level and all sub-designs

☐ Check for correct back annotation ☐ Do not check design itself

OK Cancel Apply



Three Different Views

○ Hierarchy view

- » Hierarchy » New Logical Hierarchy View
- Tree-based hierarchical browser

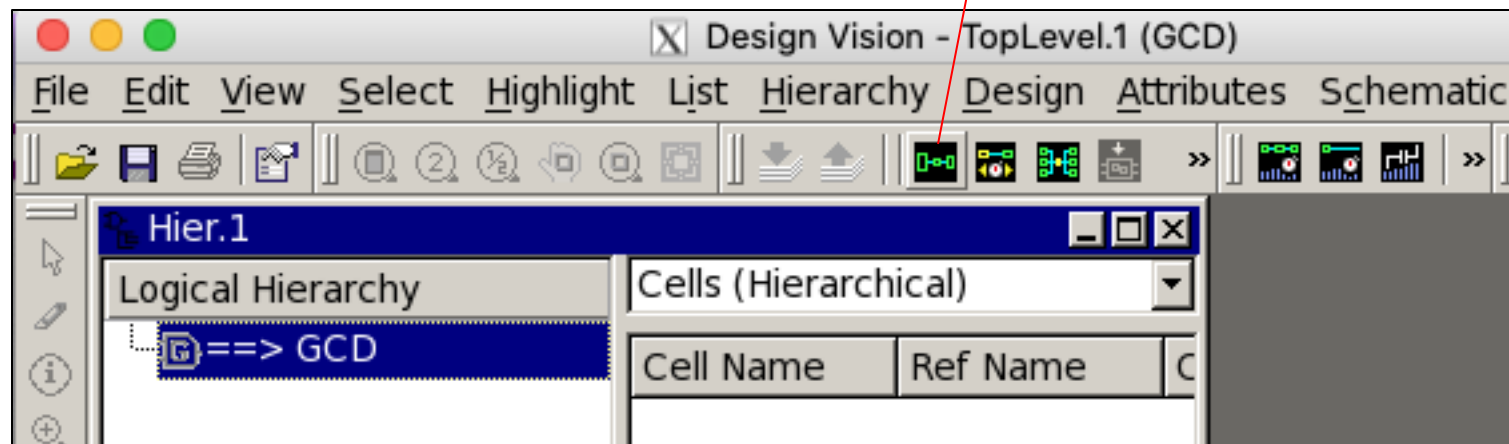
○ Design view

- » List » Design View
- List view

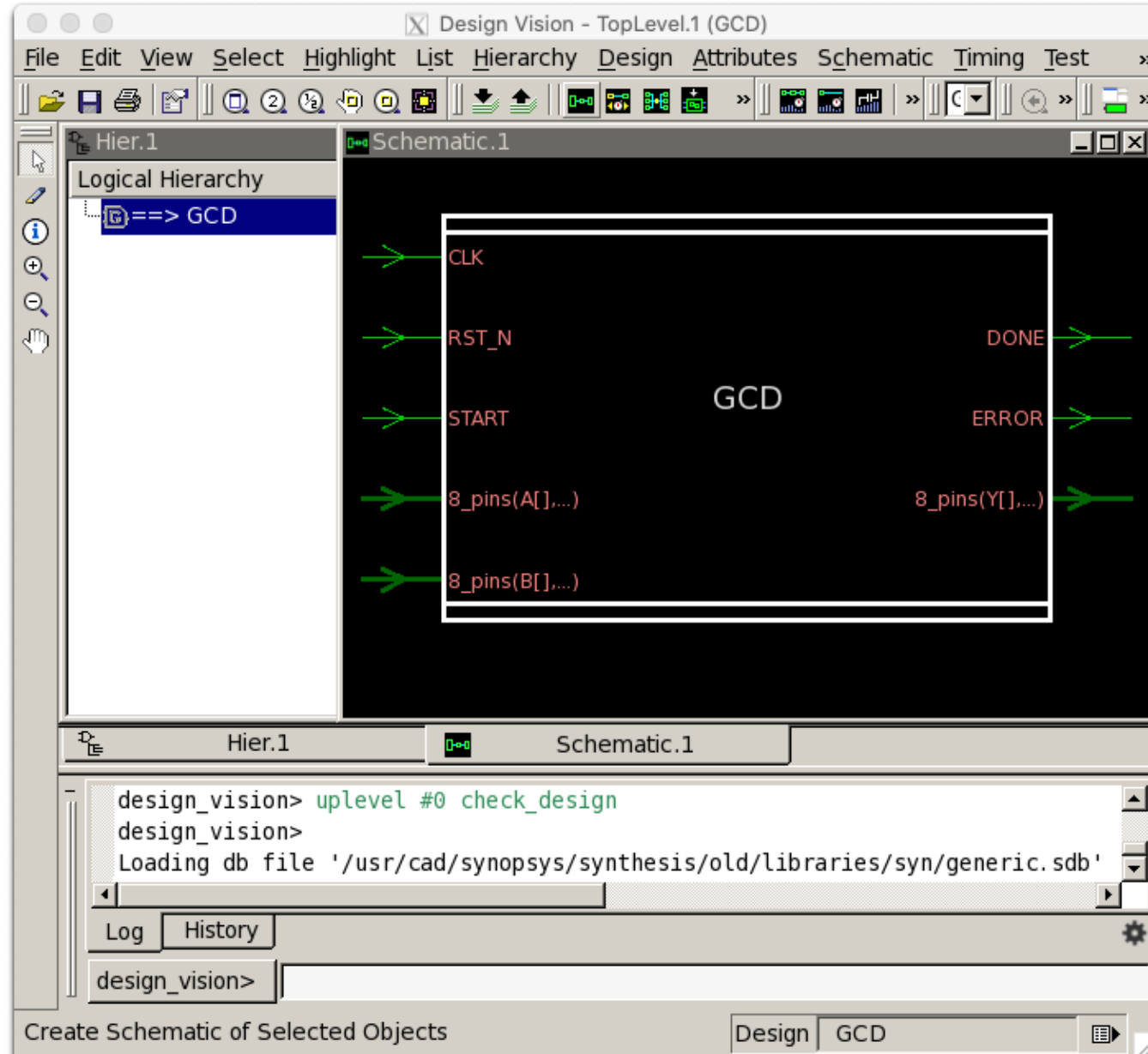
○ Schematic view

- To assign constraints on ports

Create Schematic of Selected Objects



Schematic view



» Attribute » Specify Clock

The image shows a logic simulator interface with a schematic diagram of a GCD (Greatest Common Divisor) block. The schematic has inputs: CLK, RST_N, START, 8_pins(A[],...), and 8_pins(B[],...). It has outputs: ERROR, DONE, and 8_pins(Y[],...). A context menu is open over the schematic, showing options: Specify Clock..., Operating Environment, Optimization Constraints, and Optimization Directives. The 'Specify Clock' dialog box is open, showing the following settings:

- Clock name: CLK
- Port name: CLK
- ☐ Remove clock
- Clock creation:
 - Period: 40
 - Edge table:

Edge	Value
Rising	0
Falling	20
 - Buttons: Add edge pair, Remove edge pair, Invert wave form
- Waveform plot area showing a square wave from 0.00 to 40.00.
- ☒ Don't touch network
- ☒ Fix hold
- Buttons: OK, Cancel, Apply



Timing Setup for Sequential Circuits

⦿ Under the symbol view

- ◆ Select the clock port
- ◆ » Attribute » Specify Clock
- ◆ Clock name: **CLK**
- ◆ Period: **40** (40ns, i.e., 25MHz)
- ◆ Edge value
 - ▣ Rising: **0**
 - ▣ Falling: **20**for 50% duty cycle
- ◆ Select “Don’t touch network”
- ◆ Select “Fix hold”

⦿ TCL Commands

- > **create_clock -name "CLK" -period 40 -waveform {0 20} {CLK}**
- > **set_fix_hold CLK**
- > **set_dont_touch_network CLK**



Timing Setup for Combinational Circuits

- ① Under the schematic (or symbol) view
 - ◆ Select the start and end points of the timing path
 - Usually the inputs and outputs
 - ◆ » Attribute » Optimization Constraints » Timing Constraints
 - ◆ From: the start points
 - ◆ To: the end points
 - ◆ Delays
 - Same rise and fall: use the same value for both settings
 - Max rise: maximum delay constraint (in ns)
 - Min rise: minimum delay constraint (in ns)



Command Prompt and Log Window

```
0:00:03    1913.0      0.00      0.0      0.0
0:00:03    1913.0      0.00      0.0      0.0
Loading db file '/theda21_2/CBDK_IC_Contest/cur/SynopsysDC/db/slow.db'
Loading db file '/theda21_2/CBDK_IC_Contest/cur/SynopsysDC/db/fast.db'

Note: Symbol # after min delay cost means estimated hold TNS across all active scenarios

Optimization Complete
-----
1
Current design is 'GCD'.
design_vision> write_sdf gcd_syn.sdf
```

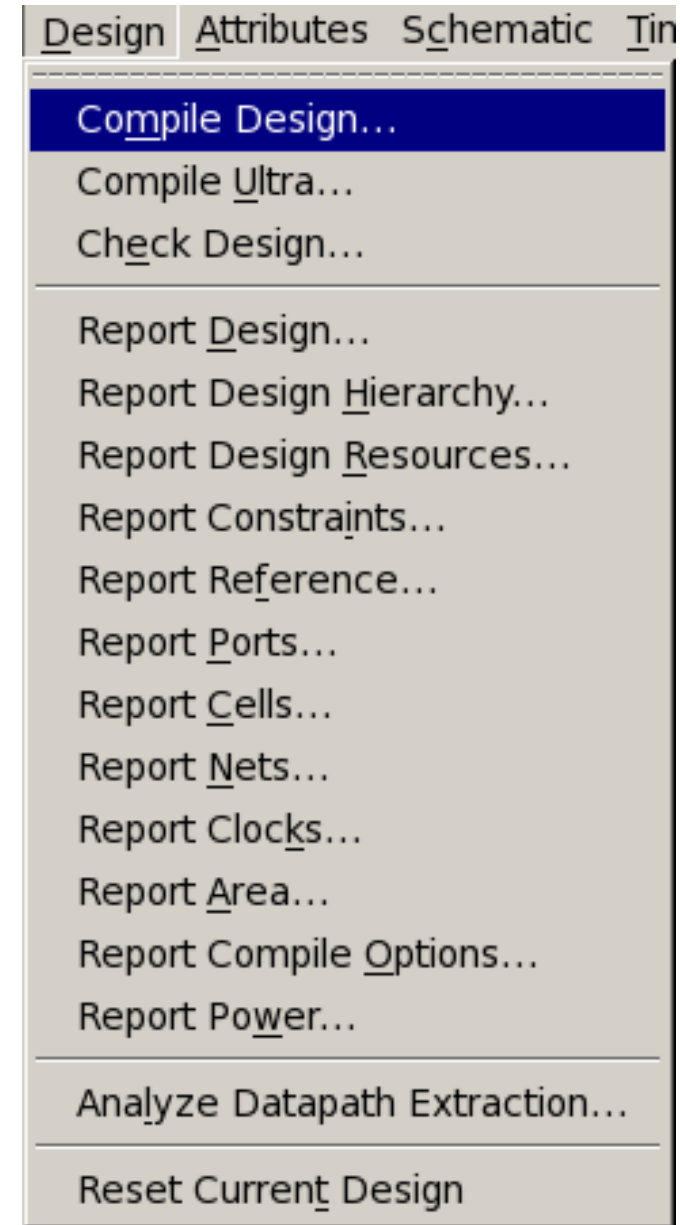
Log History

design_vision> write_sdf gcd_syn.sdf

Ready

Compiling the Design (1/3)

- ① » Design
 - » Compile Design
- ① Mapping options
 - ◆ Map effort: high/**medium**
 - ◆ Area effort: high/**medium**/low/none
 - ◆ Power effort: high/**medium**/low/none





Compiling the Design (2/3)

Compile

Mapping options

- ☒ Map design
 - ☒ Exact map
 - Map effort:
 - Area effort:
 - Power effort:

Compile options

- ☐ Top level
- ☐ Incremental mapping
- ☐ Ungroup
- ☐ Allow boundary condition
- ☐ Scan
- ☐ Auto ungroup
- ☐ Gate Cloc
 - ☒ Area
 - ☐ Delay

Design rule options

- ☒ Fix design rules and optimize mapping
- ☐ Optimize mapping only
- ☐ Fix design rules only
- ☐ Fix hold time only

OK Cancel Apply

Compiling the Design (3/3)

The screenshot shows the Design Vision - TopLevel.1 (GCD) window. The Logical Hierarchy pane on the left shows a tree structure with 'GCD' as the root and 'sub_48' as a child. The Cells (Hierarchical) pane on the right shows a table with the following data:

Cell Name	Ref Name	C
sub_48	GCD_DW01...	st

The Log/History pane at the bottom shows the following text:

```
0:00:03 1913.0 0.00 0.0 0.0
0:00:03 1913.0 0.00 0.0 0.0
0:00:03 1913.0 0.00 0.0 0.0
Loading db file '/theda21_2/CBDK_IC_Constest/cur/SynopsysDC/db/slow.db'
Loading db file '/theda21_2/CBDK_IC_Constest/cur/SynopsysDC/db/fast.db'

Note: Symbol # after min delay cost means estimated hold TNS across all active scenarios

Optimization Complete
-----
1
Current design is 'GCD'.
design_vision>
```

The status bar at the bottom indicates 'Ready'.



Reporting the Synthesis Result

- ① » Design » Report Area
 - ◆ You can report to a file
- ① » Design » Report Power
- ① » Timing » Report Timing Paths
 - ◆ You can simply press “OK” or “Apply”
 - ◆ Slack
 - MET (positive)
 - VIOLATED (negative)





Timing Path Example

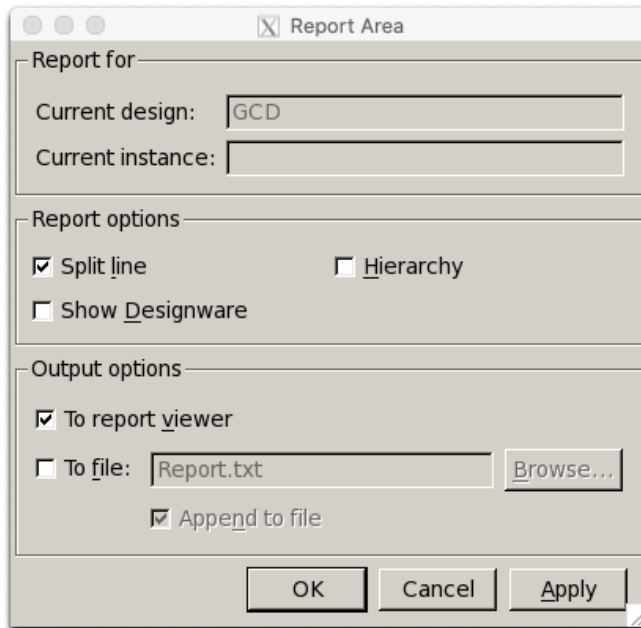
Point	Incr	Path

clock CLK (rise edge)	0.00	0.00
clock network delay (ideal)	0.00	0.00
reg_a_reg[0]/CK (DFFRX1)	0.00	0.00 r
reg_a_reg[0]/Q (DFFRX1)	0.51	0.51 f
U160/Y (NOR2BX1)	0.12	0.63 r
U161/Y (A021X1)	0.18	0.81 r
	:	:
	:	:
	:	:
U112/Y (A022X1)	0.31	5.50 f
reg_a_reg[7]/D (DFFRX1)	0.00	5.50 f
data arrival time		5.50
clock CLK (rise edge)	40.00	40.00
clock network delay (ideal)	0.00	40.00
reg_a_reg[7]/CK (DFFRX1)	0.00	40.00 r
library setup time	-0.21	39.79
data required time		39.79

data required time		39.79
data arrival time		-5.50

slack (MET)		34.29

Report Area



For Virtual Library,
cell area of NAND2 is
5.092200

1912.969779/5.0922
~ 376 gates
(equivalent gates)

```
design_vision> uplevel #0 { report_area }

*****
Report : area
Design : GCD
Version: K-2015.06-SP1
Date   : Tue May 26 17:35:08 2020
*****

Library(s) Used:

    slow (File: /theda21_2/CBDK_IC_Constest/cur/SynopsysDC/db/slow.db)

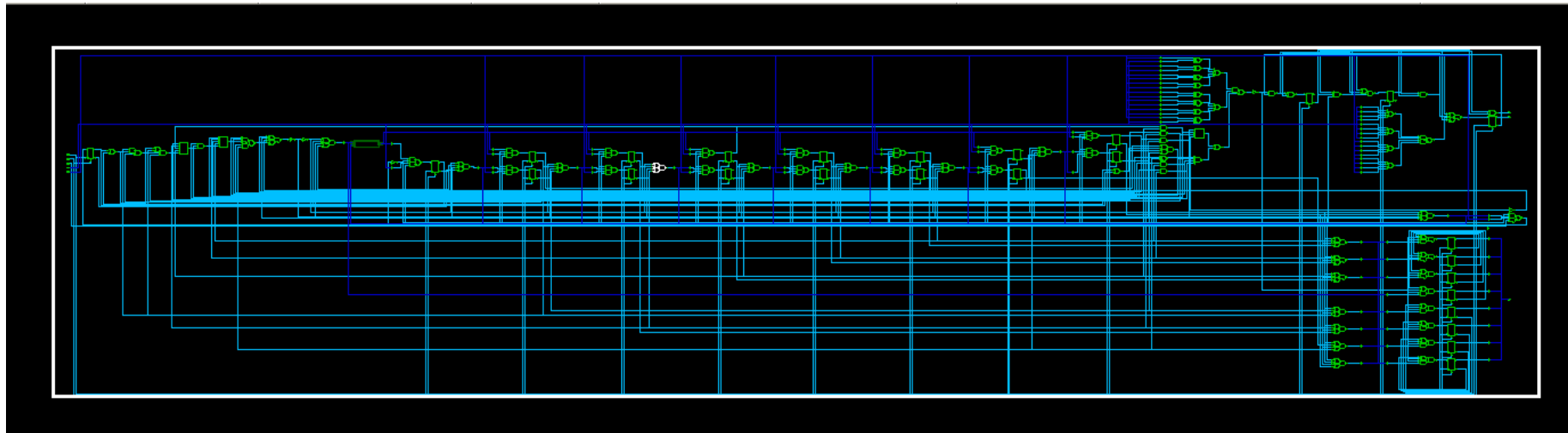
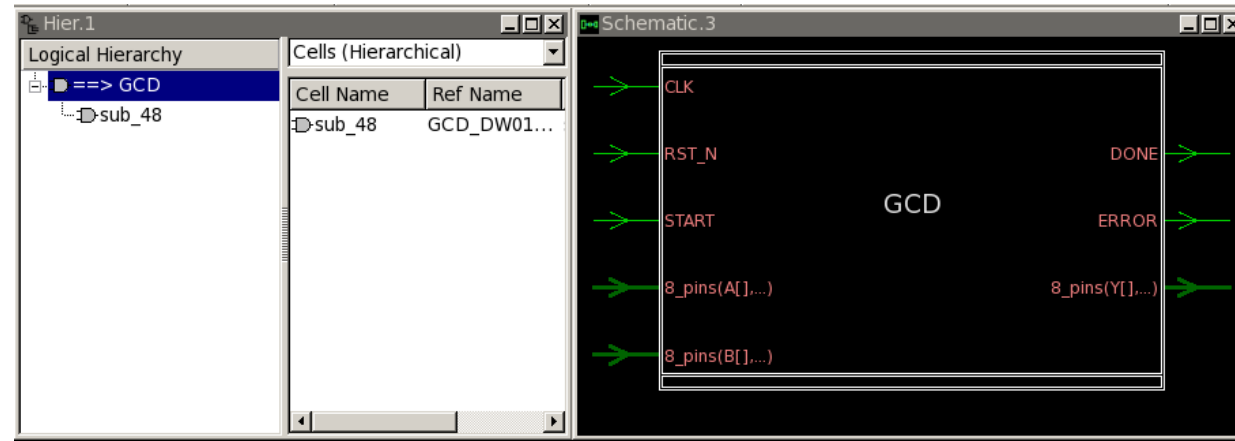
Number of ports:                53
Number of nets:                 200
Number of cells:               134
Number of combinational cells: 105
Number of sequential cells:    28
Number of macros/black boxes:  0
Number of buf/inv:            14
Number of references:          24

Combinational area:             1042.203609
Buf/Inv area:                   54.316799
Noncombinational area:         870.766171
Macro/Black Box area:          0.000000
Net Interconnect area:         undefined (No wire load specified)

Total cell area:                1912.969779
Total area:                    undefined
design_vision>
```

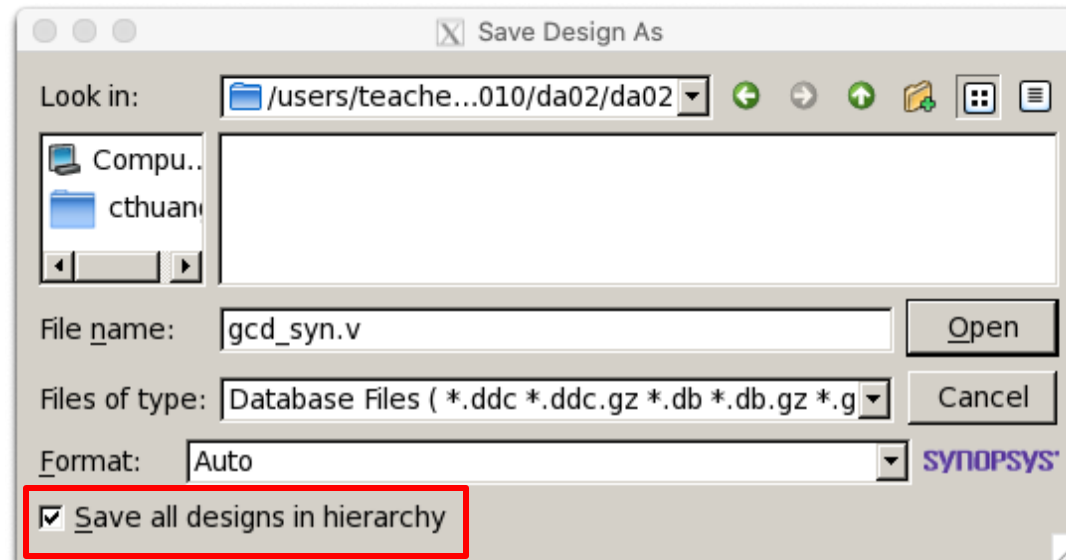
For GPDK045
Cell area of NAND2x1 is
1.026 μm^2

Schematic View



Saving the Design

- » File » Save As
- Select the top design module first
- Select “Save all design in hierarchy”
- TCL command
 - > `write_file -format verilog -hierarchy -output gcd_syn.v`





Example: gcd_syn.v (1/2)

```
module GCD_DW01_sub_0 ( A, B, CI, DIFF, CO );
  input [7:0] A;
  input [7:0] B;
  output [7:0] DIFF;
  input CI;
  output CO;
  wire      n17, n18, n19, n20, n21, n22, n23, n24;
  wire      [8:0] carry;

  XOR3X2 U2_7 ( .A(A[7]), .B(n17), .C(carry[7]), .Y(DIFF[7]) );
  ADDFX2 U2_1 ( .A(A[1]), .B(n23), .CI(carry[1]), .CO(carry[2]), .S(DIFF[1])
    );
  ADDFX2 U2_5 ( .A(A[5]), .B(n19), .CI(carry[5]), .CO(carry[6]), .S(DIFF[5])
    );
  ADDFX2 U2_4 ( .A(A[4]), .B(n20), .CI(carry[4]), .CO(carry[5]), .S(DIFF[4])
    );
  ...
```




Example: gcd_syn.v (2/2)

```
module GCD ( CLK, RST_N, A, B, START, Y, DONE, ERROR );
  input [7:0] A;
  input [7:0] B;
  output [7:0] Y;
  input CLK, RST_N, START;
  output DONE, ERROR;
  wire  N22, N23, N24, N25, N26, N27, N28, N29, n68, n69, n70, n71, n72, n73,
        n74, n75, n76, n77, n78, n79, n80, n81, n82, n83, n84, n85, n386,
        n387, n388, n389, n390, n391, n392, n393, n394, n395, n396, n397,
        n398, n399, n400, n401, n402, n403, n404, n405, n406, n407, n408,
  ...
  GCD_DW01_sub_0 sub_48 ( .A({n481, n451, n453, n455, n457, n459, n461, n463}),
    .B({n465, n466, n467, n468, n469, n470, n471, n472}), .CI(1'b0),
    .DIFF(diff) );
  DFFRX1 \Y_reg[6] ( .D(n85), .CK(CLK), .RN(RST_N), .Q(Y[6]), .QN(n75) );
  DFFRX1 \Y_reg[5] ( .D(n84), .CK(CLK), .RN(RST_N), .Q(Y[5]), .QN(n74) );
  ...
```



SDF Timing Information

- ⦿ SDF: Standard Delay Format
- ⦿ Wire delays in addition to gate delays
- ⦿ Select the top design module, use the TCL command to save the information
 - > `write_sdf -version 1.0 gcd_syn.sdf`
 - Or
 - > `uplevel #0 write_sdf -version 1.0 gcd_syn.sdf`



Design Setup Information

⦿ » File » Save Info » Design Setup

- ◆ Design setup file (e.g., gcd.tcl)

⦿ Setup script:

- ◆ Timing constraints
- ◆ Area constraints
- ◆ IO constraints
- ◆ Etc.

⦿ TCL command

> `write_script -output gcd.tcl`

⦿ Synthesis commands can also be found in command.log



Example of gcd.tcl

```
read_file -format verilog {./gcd.v}
create_clock -name "CLK" -period 40 -waveform { 0 20 } { CLK }
set_fix_hold CLK
set_dont_touch_network CLK
compile -exact_map
remove_unconnected_ports -blast_buses [find -hierarchy cell "*"]
write_file -format verilog -hierarchy -output gcd_syn.v
write_sdf -version 1.0 gcd_syn.sdf
uplevel #0 { report_timing -path full -delay max -nworst 1 -max_paths 1
              -significant_digits 2 -sort_by group }
report_area
exit
```



Synthesis without GUI

⦿ dc_shell is a command-line shell

```
dc_shell> source gcd.tcl
```

Or

```
$ dc_shell -f gcd.tcl
```



What's Next?

- ⦿ After synthesis, you need to perform post-synthesis simulation
- ⦿ Post-syn simulation should match RTL simulation
- ⦿ If not matched?
 - ◆ RTL design style
 - ◆ Improper setup in the simulation
 - ◆ Test patterns with improper timing



Gate-Level Simulation with Timing Information

- In the stimulus, add the following task
- `$sdf_annotate(sdf_filename, top_instance_name);`
- E.g.,
`$sdf_annotate("gcd_syn.sdf", gcd01);`
- Verilog simulation
`ncverilog gcd_t.v \
gcd_syn.v \
-v /theda21_2/CBDK_IC_Contest/cur/Verilog/tsmc13.v \
+access+r`

for Virtual Cell Library

Modified Test Stimulus with Compiler Directive

```
initial begin
  `ifdef SYNTHESIS
    $sdf_annotate("gcd_syn.sdf", gcd01);
    $fsdbDumpfile("gcd_syn.fsdb");
  `else
    $fsdbDumpfile("gcd.fsdb");
  `endif
  $fsdbDumpvars;
  ...
end
```




An Example of Header File: header.v

```
`define SYNTHESIS
```



An Example for Makefile (for CBDK IC Contest)

```
VLOG      = ncverilog
SRC       = gcd_t.v \
           gcd.v
SYNSRC    = header.v \
           gcd_t.v \
           gcd_syn.v \
           -v /theda21_2/CBDK_IC_Contest/cur/Verilog/tsmc13.v

VLOGARG   = +access+r

all :: sim

sim :
    $(VLOG) $(SRC) $(VLOGARG)
syn :
    $(VLOG) $(SYNSRC) $(VLOGARG)
```



Makefile (for GPDK045)

```
DEBUG = 3
# add your source code
SRC = testbench.v input_sram.v weight_sram.v output_sram.v top.v
BAK = *.bak
LOG = *.log *.history *.key *.fsdb out_log.txt
INCA_libs = INCA_libs
SYNSRC = header.v testbench.v input_sram.v weight_sram.v output_sram.v top_syn.v -v
        /theda21_2/library/GPDK045/cur/gsclib045/verilog/slow_vdd1v0_basicCells.v

all :: sim
sim :
    ncverilog +debug=${DEBUG} ${SRC} +access+r
syn :
    ncverilog +debug=${DEBUG} ${SYNSRC} +access+r
clean:
    -rm -f ${BAK} ${LOG}
    -rm -rf ${INCA_libs}
```



Check up Output Log Carefully (1/2)

ncverilog.log

```
file: gcd_syn.v
  module worklib.GCD:v
    errors: 0, warnings: 0
    Caching library 'tsmc18' ..... Done
    Caching library 'worklib' ..... Done
    Elaborating the design hierarchy:
    GCD_DW01_sub_0 sub_48 ( .A({n481, n451, n453, n455, n457, n459, n461, n463})),
      |
ncelab: *W,CUVWSP (./gcd_syn.v,61|22): 1 output port was not connected:
ncelab: (./gcd_syn.v,2): C0
...
```



Check Log Carefully (2/2)

ncverilog.log

```
Reading SDF file from location "gcd.sdf"
```

```
  Annotating SDF timing data:
```

```
    Compiled SDF file:      gcd.sdf.X
```

```
    Log file:
```

```
    Backannotation scope:  stimulus.gcd01
```

```
    Configuration file:
```

```
    MTM control:
```

```
    Scale factors:
```

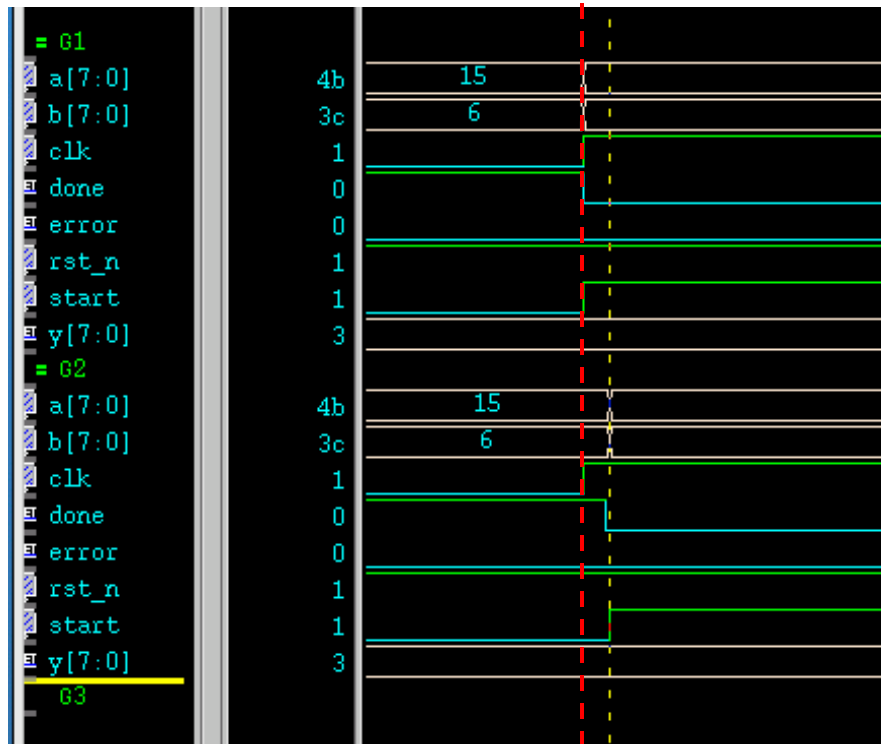
```
    Scale type:
```

```
Annotation completed successfully...
```

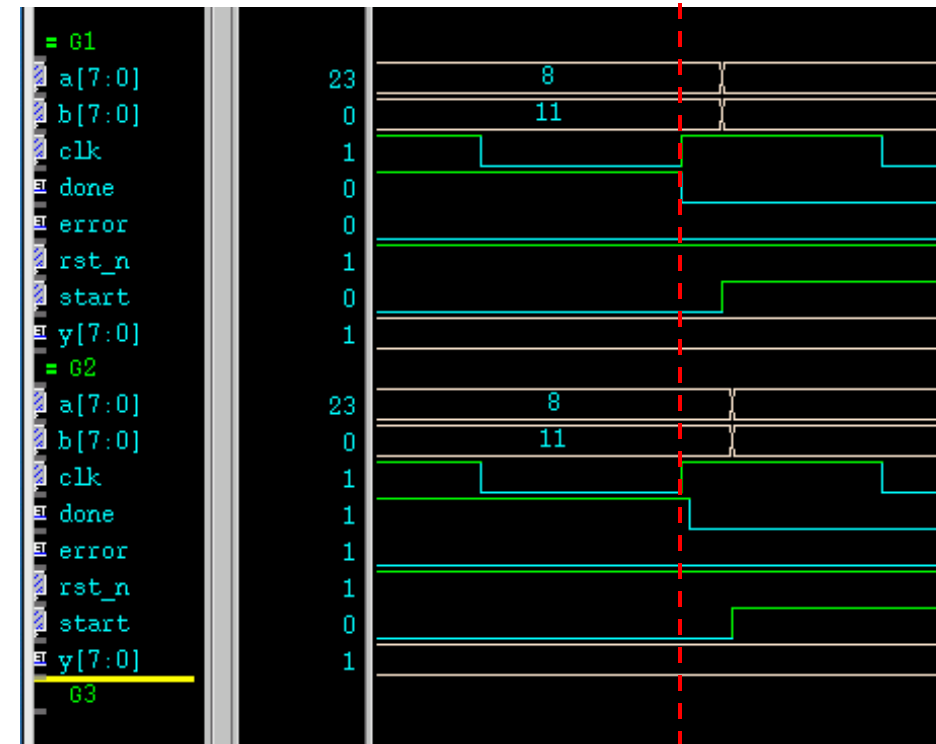


RTL vs. Gate-Level Simulation

Zero-delay in RTL



Non-Zero delay in Gate-Level





Summary

- ⦿ Preliminary synthesis is a handy tool to verify your RTL coding
- ⦿ (Static) lint tool and (dynamic) code coverage tool are also helpful
 - ◆ Synthesis may be costly
- ⦿ Remember that a good design plan is necessary
 - ◆ Plan the design before coding
 - ◆ Verify the coding with the design plan
 - ◆ Modify the design plan (not the code) whenever the (simulation) result do not meet your expectation