IC-Project I-Synthesis

Joachim Rodrigues, LUND UNIVERSITY

Outline

Objective of the Presentation

- Synthesis
- Basic synthesis flow
- Synopsys DesignCompiler
- Synthesis script

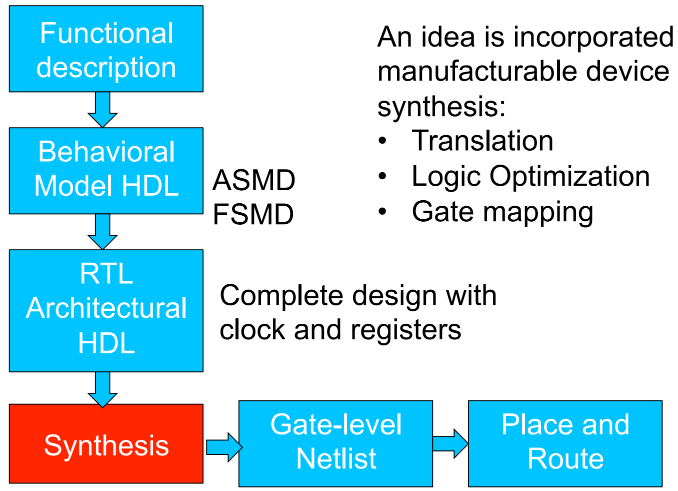
Objective of the Presentation

- Introduce basic synthesis
- Guide that can be used to create a basic synthesis flow
  - Steps
  - Actual commands
- Getting familiar with the synthesis environment
- Your first ASIC synthesis script

What is Synthesis?

- A process which combines two or more pre-existing elements resulting in the formation of something new.
- Synthesis links the conceptual description of the logic functions needed for the design to their actual physical architecture elements in the underlying device.

What is Synthesis?



An idea is incorporated into a manufacturable device by doing synthesis:

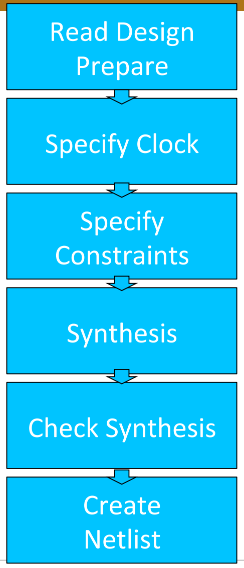
- Translation
- Logic Optimization
- Gate mapping

Synthesis Tool -Design Compiler (DC)

- Common tool provided by Synopsys Well-known in industry and academia
- Online support: https://solvnet.synopsys.com/
- Command help in Synopsys-DC GUI.

- Graphical mode
  - DesignVision
- Shell mode
  - dc_shell

Synthesis Flow



Libraries

- Vendor delivers technology libraries as ASCII file (*.lib_)
- describes parameters and rules for a particular technology(**130nm**,90nm, **65nm**...).
- Every process consists of logic cells that has different functionality.
- full adder, multiplier, flip-flop, XOR, NAND etc
- Compiled for Synopsys DC usage (*.db)
- Various libraries, e.g., low-leakage (LL) or high-speed (HS) are usually available.

Libraries

Target library is used by DC to build the circuit

- DC chooses gates from libraries

- Gate timing information is included in libraries Defined in **.synopsys_dc.setup**
- Copied into the working directory when init scripts are run. specifies the libraries being used and other configurations.
- *.lib information for the memory needs to be read by DC.
- **SYNTAX:** `read_lib memoryX.lib`
  SYNTAX: `write_lib memoryX` (writes the memory in **.db** format)

- If *.db is already available, include them in the *link_library* and *target_library*

Synthesis Flow

Read Design
Prepare

- DC reads both RTL designs and gate-level netlist.

- DC reads design files with **analyze** and **elaborate** Commands
  - **analyze**: analyzes HDL files and stores the intermediate format for the HDL description in the specified library
  - **elaborate**: Builds a design from the intermediate format, a VHDL entity and architecture

- Every instance becomes unique.

Syntax- Analyze

analyze
[**-library** *library_name*]
[**-format** *vhdl* | *verilog* | *sverilog*] **file_list**

-library *library_name*
Maps the work library to *library_name*.
By default, analyze stores all output in the work library.

-format *vhdl* | *verilog* | *sverilog*
Specifies the format of the files that are to be analyzed;

file_list
Specifies a list of files to be analyzed. When specifying more than one file, enclose the files in braces: { }.

Example:
analyze -format vhdl -lib WORK {../vhdl/your_design.vhd}

Syntax- Elaborate

elaborate design_name
[**-library** *library_name* | **-work** *library_name*]
[**-architecture** *arch_name*]
[**-update**]

design_name
Specifies the name of the. Can be a Verilog module, a VHDL entity, or a VHDL configuration.

-library *library_name*
Specifies the library name that work is to be mapped to.
By **Default**: elaborate looks in the *work* library for the design to be built.

-architecture *arch_name*
Specifies the name of the architecture, .e.g., behavioral, structural, rhubarb, ... Example:

```
elaborate fir –lib WORK –arch structural
```

Clock Definition

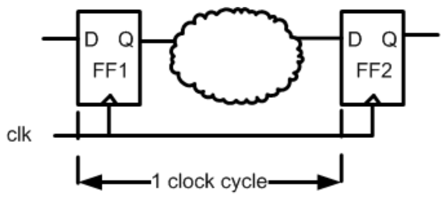
Read Design
Prepare

Specify Clock

In DC, clock is ideal: no buffers, no DRC, no optimization

Required Definitions

- clock period
- clock name
- clock source
- Duty cycle (50% default)
- (Offset/skew)



A clock constrains timing paths between registers.
A design may have several clocks.

Real clock synthesis in *place and route*

Syntax- create_clock

create_clock
  [**-period** *period_value*] [**-name** *clock_name*]
  [**source_objects**]
-period *period_value*
  The period of the clock waveform in library time units.
  default unit is ns
-name *clock_name*
  Specifies the name of the clock being created.
source_objects
  Specifies a list of pins or ports on which to apply this clock.

Example: `create_clock clk –period 20 -name clk`

Clock skew

Worst case clock skew needs to be defined
- technology and design dependent
- not easy to determine
- Around 2% of clock period

Syntax:
set_clock_uncertainty ***1 name_of_your_clock***

also

set_fix_hold ***name_of_your_clock***

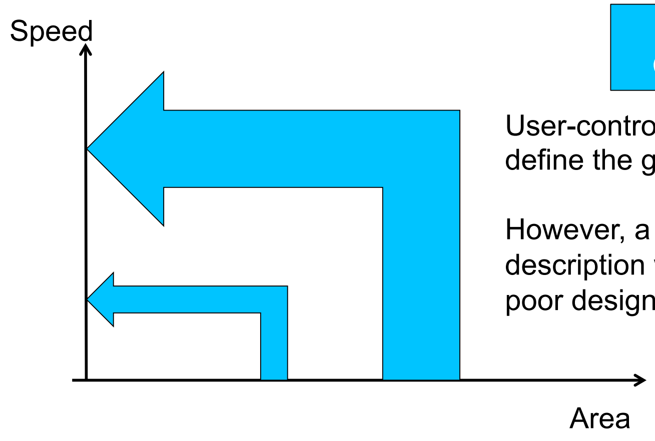
Synthesis Constraints

Read Design
Prepare

Specify Clock

Specify
Constraints

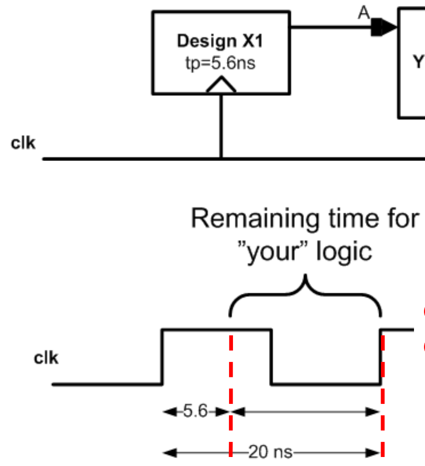
High speed vs low-area



User-controlled constraints define the goal

However, a poor RTL description will result in a poor design

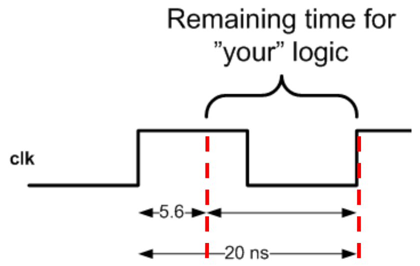
Constraining Input Paths



Logic is triggered by the same clock

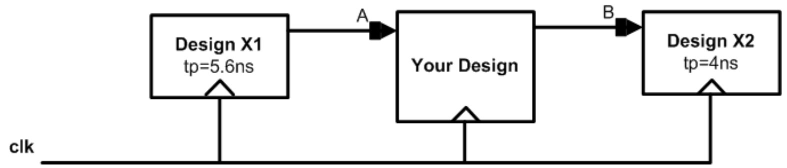
Need to Specify propagation delay of external logic that drives your logic

Constraining Input Paths



```
set_input_delay –max 5.6 –clock clk [get_ports A]
```

Constraining Output Paths



Need to Specify propagation delay of external logic that is driven by your logic

```
set_output_delay –max 4 –clock clk [get_ports B]
```

This command could be useful in the project part if you need to connect several designs.

Constraining Area

Area is expensive and needs to be constrained

`set_max_area`

sets the max_area attribute to a specified value on the current design.

The max_area attribute represents the target area of the design and is used by the compile command to calculate area cost of the design.

SYNTAX

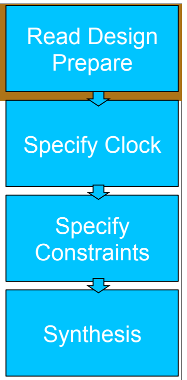
`set_max_area area_value` e.g. **`set_max_area 0`**

- Synthesis tool prioritizes total negative slack over area.
- A design that does not meet timing will not work.
- Compile does not create new delay violations or worsen existing delay violations on a path that has negative delay slack in order to improve area.

Area vs Speed

- For a high-speed circuit do not set any area constraint **but** specify a high clock frequency.
- For an area optimized circuit set area to 0 and specify a low clock frequency.
- Assignment:
  - Two synthesis runs are necessary.
    - Highest speed
    - Smallest area.

Synthesis and Optimization



- The command ***compile*** performs logic and gate-level synthesis and optimization on the current design.
- Optimization is controlled by user-specified constraints
  - to obtain smallest possible circuit
  - or fastest design
  - or any other design requirement.
- The constraints describe
  - goals for the optimization process (area).
  - try to make specified outputs arrive by a specified time.
- Values for components' area and speed used during synthesis and optimization are obtained from user-specified libraries.

Syntax: *compile*

compile [-map_effort low | medium | high] -map_effort

Relative amount of CPU time spent during mapping phase of compile. Default : Medium effort.

Example: **compile -map_effort high**

More switches for compile are available but not scope of this presentation!!

Netlist and Timing Information

All possible violations need to be checked by executing:
report_constraint -all_violators

Other commands to check design:
report_design
report_area –hierarchy report_timing –max_paths *no_of_paths*

Thereafter, a netlist can be written in several formats
- VHDL
- Verilog
- db or ddc (Synopsys specific format)

Netlist and Timing Information

The names of nets, buses etc., need to be changed to the desired netlist format

change_names -rules [vhdl | verilog] -hierarchy

A netlist is generated with
write -format [vhdl | verilog] -hierarchy -output ./netlists/your_design.v

A file that contains timing information for gate-level simulation is generated
write_sdf ./netlists/your_design.sdf
(.sdf required for post-synthesis simulation)

write_sdc ./netlists/your_design.sdc
(.sdc required for Place and Route)

Getting Started

Change to the folder where you want to do synthesis, and execute ***inittde*** **dig130x17**
initializes the environment and copies some setup files (if required)

For synthesis **.synopsys_dc.setup** is the initialization file
CAD tools initialization script creates several directories (retaining directory structure STRONGLY recommended)

- vhdl *(copy your VHDL design files into this directory)*
- netlists *(save your netlist, sdf and sdc files in this directory)*
- WORK *(for Synopsys)*
- work *(for ModelSim)*
- soc

Execute ***design_vision*** in the same terminal as inittde was executed and graphical user interface of the synthesis tool pops-up.

A tcl script is available in “comp.dv” file for the dummy design, go through it !!