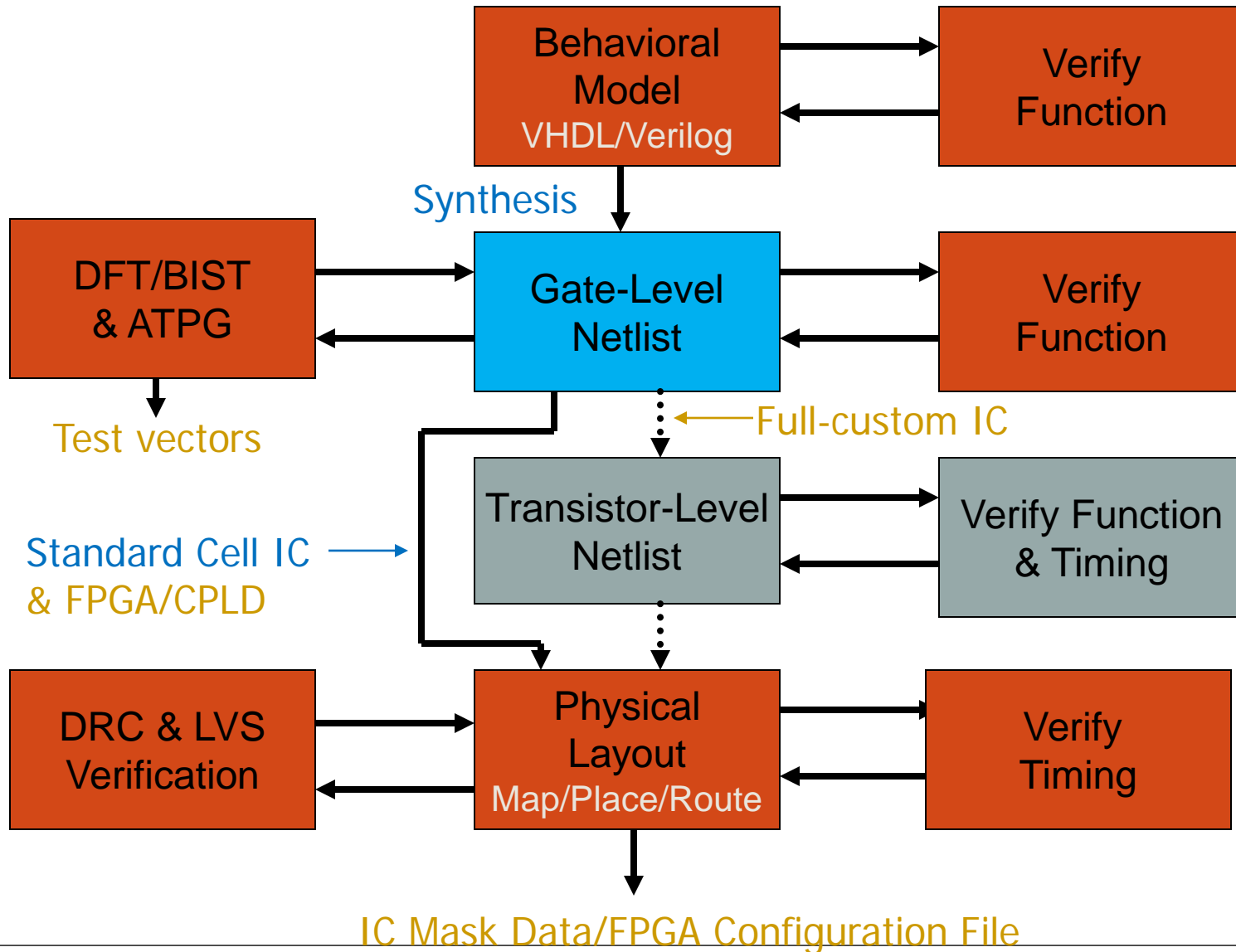


Computer-Aided Design

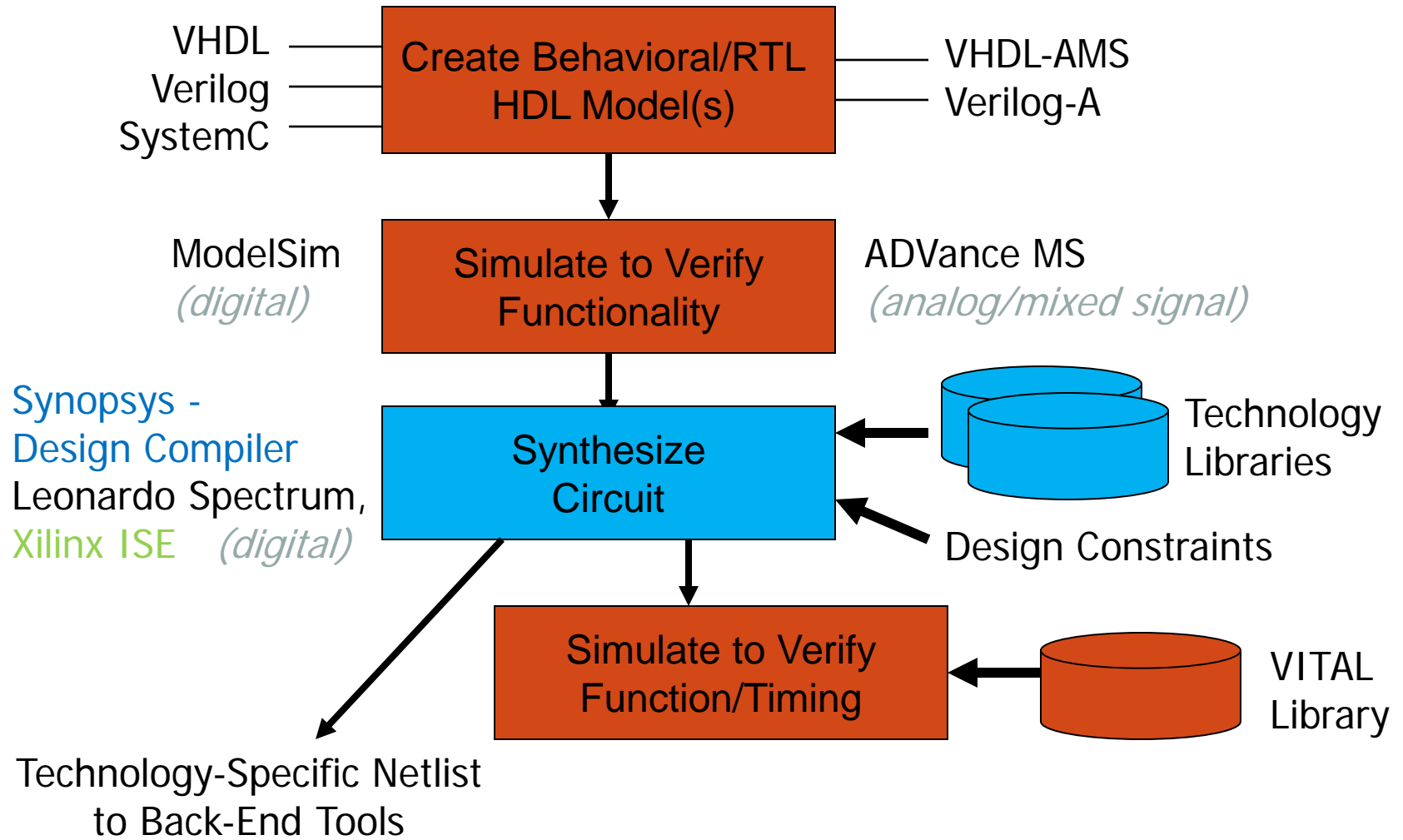
Synthesis with Synopsys Design Compiler

Victor P. Nelson

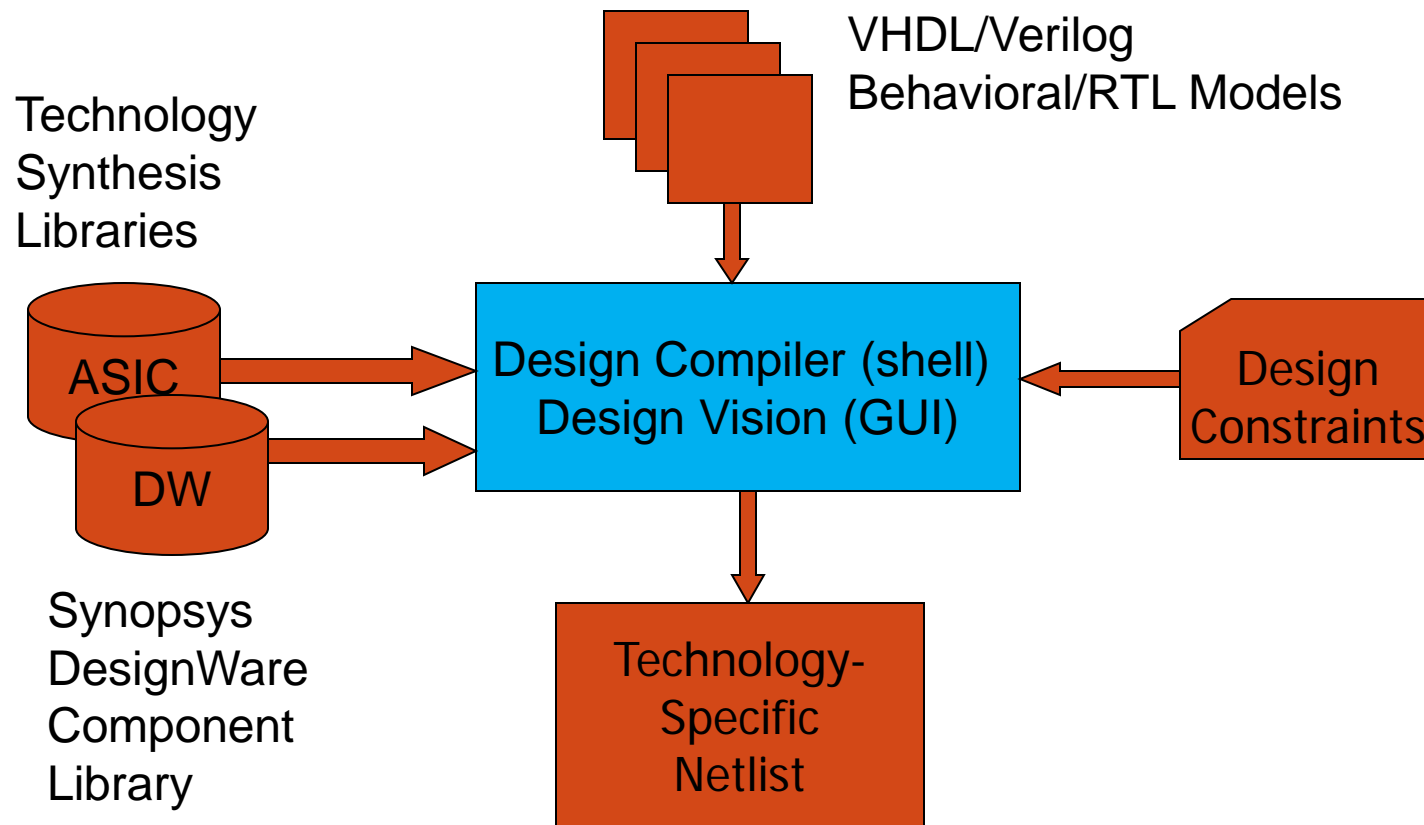
ASIC Design Flow



Behavioral Design & Verification

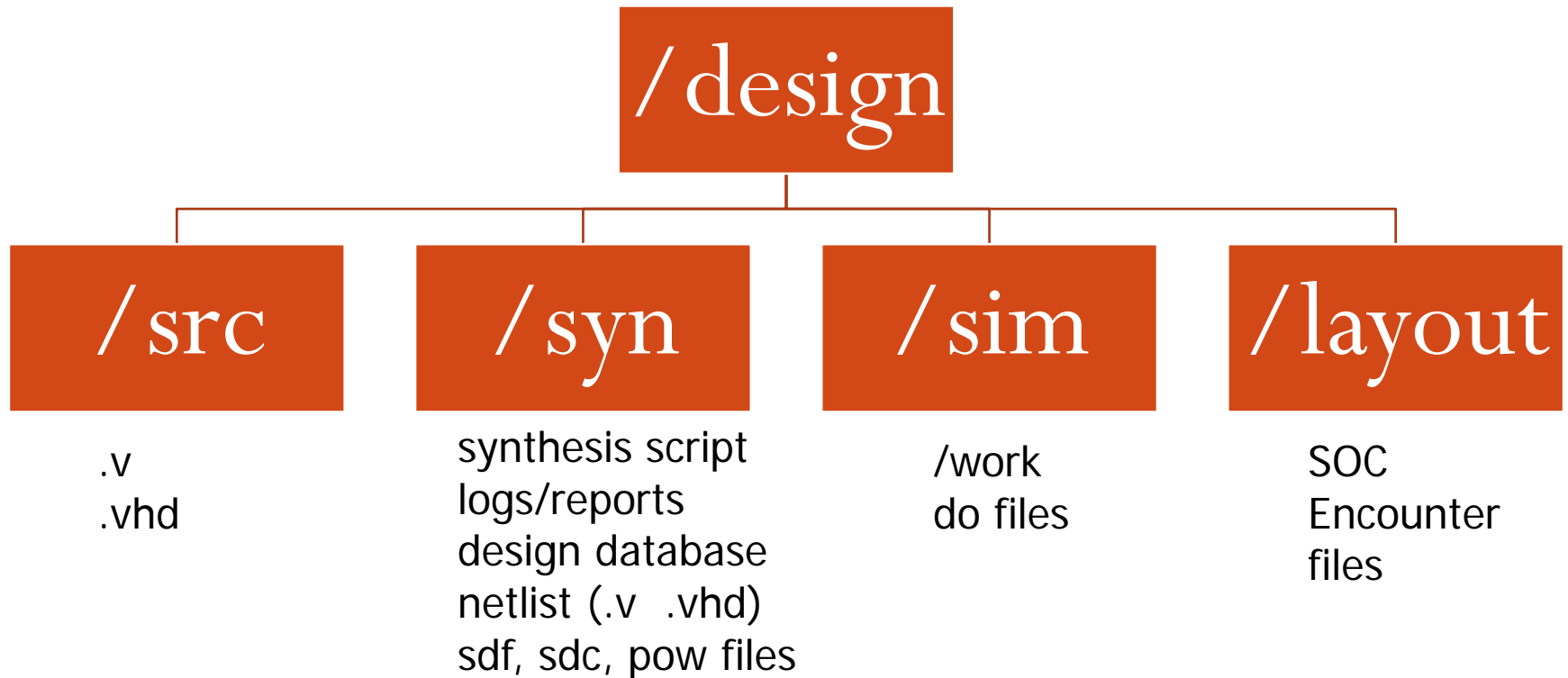


Automated Synthesis with *Synopsys Design Compiler*



Verilog, SDF, DDC database,
SDC constraints, REP report

Project directory structure



Invoking Design Compiler

- Interactive shell version:
 - *dc_shell -f scriptFile*
 - Most efficient and common usage is to put all commands into scriptFile, including “quit” at the end
 - Edit and rerun scriptFile as needed
- GUI version (Design Vision)
 - *design_vision*
 - From dc_shell: *gui_start*
 - Main advantage over dc_shell is to view the synthesized schematic

Design Compiler library files

- **target_library** : standard cell database (binary)
 - cell area/timing data (for synthesis decisions)
 - *set target_library [list vtv_tsmc250.db]*
- **link_library** : use during linking (*=internal data)
 - *set link_library [list "*" vtv_tsmc250.db]*
- **symbol_library** : schematic symbols (for DesignVision)
 - *set symbol_library [list vtv_tsmc250.sdb]*
- **synthetic_library**: DesignWare components
 - defined in main *.synopsys_dc.setup* file
 - *set synthetic_library [list ds_foundation.sldb]*

Setup file `.synopsys_dc.setup` (TSMC 0.25 μ m)

- DC reads `.synopsys_dc.setup` files in order:
 1. Synopsys installation directory (all user projects)
 2. User home directory (all projects for this user)
 3. Current project directory (this project only)

```
set MyHome [getenv "HOME"]
```

```
set SynopsysInstall [getenv "STROOT"]
```

```
set search_path [list \
```

```
  [format "%s%s" $MyHome /linux_apps/cadence_libs/vtvt_tsmc250_release_4/Synopsys_Libraries/libs] \
```

```
  [format "%s%s" $SynopsysInstall /libraries/syn] \
```

```
  [format "%s%s" $SynopsysInstall /dw/sim_ver]]
```

```
set target_library [list vtvt_tsmc250.db]
```

```
set link_library [list "*" vtvt_tsmc250.db]
```

```
set symbol_library [list vtvt_tsmc250.sdb basic.sdb US.8ths.sdb NCSU_Analog_Parts.sdb]
```

Synopsys *DesignWare* Package

- Predesigned components (tech-independent)
 - arithmetic, filters, CRC gen's, counters, decoders, FIFOs, flip-flop RAMs, etc.
 - dw_foundation link library
- Let DC choose, or instantiate directly

- Example DW decrementer:

```
module decrementer (in_A, SUM_out;  
    parameter width = 8;  
    input [width-1 : 0] in_A;  
    output [width-1 : 0] SUM_out;  
    DW01_dec #(width) U1( .A(in_A), .SUM(SUM_out));  
endmodule;
```

Read in the design

- Single Verilog/VHDL file:
 - *read_file -f verilog*
- Multiple Verilog/VHDL files:
 - *analyze -format verilog -libWORK <files>*
 - parse & put files into WORK
 - format can be vhdl or verilog
 - *elaborate <top module> -libWORK -update*
 - compile into tech-independent intermediate form
 - discover all inferred memory devices (latch, ff)

Example

```
#Design-specific information — for convenience  
set myFiles [list ./src/top.vhd ./src/Muxbig.vhd ]  
set basename TOP  
set fileFormat vhd  
define _design_libWORK —path ./syn  
# These need not be changed  
analyze —format $fileFormat -libWORK $myFiles  
elaborate $basename —libWORK —update  
current_design $basename  
link                   (link all design parts)  
uniquify               (make unique copies of replicated modules)  
# Now specify design constraints before compiling
```

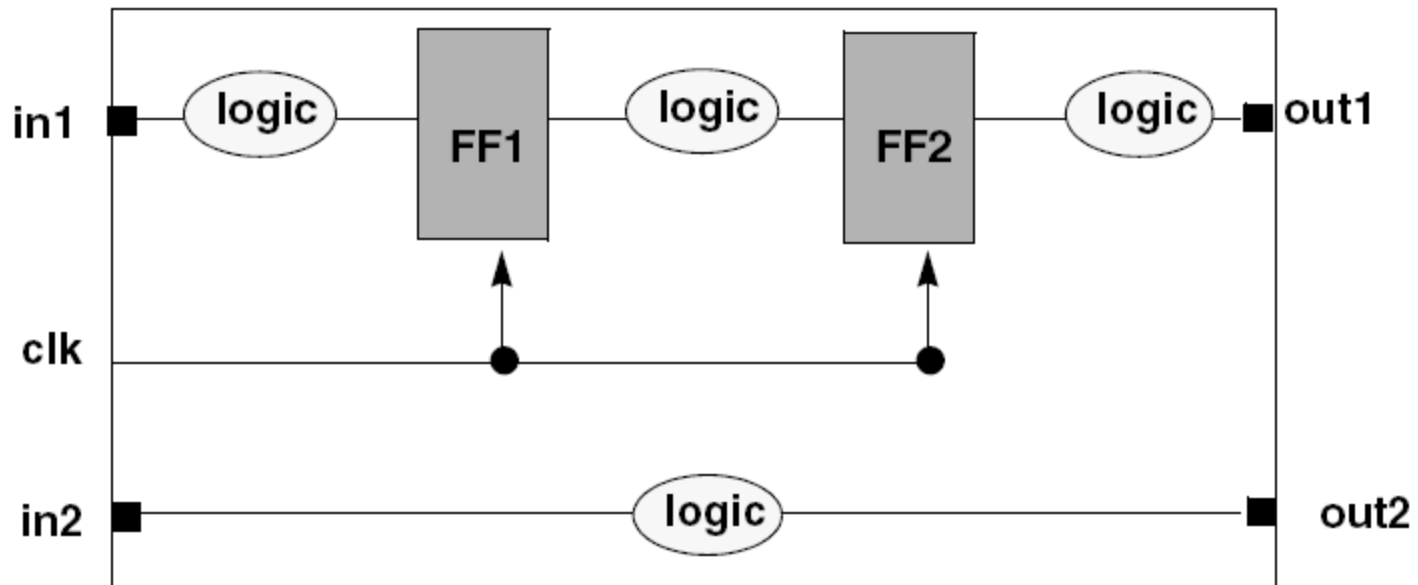
Design constraints

- Speed
 - path delays (min,max)
 - clock specifications (period/frequency/duty)
- Area
 - speed is primary goal, then optimize area if timing constraints met
 - target area 0 forces small as possible
 - *set_max_area 2000*
- Choose realistic constraints (within 1-10%)
 - avoid extra buffers/gates on loaded nets

Timing path types

Path delays of interest

1. Combinational: primary input to primary output (in2 -> out2)
2. Primary input to register input (in1 -> FF1/D1)
3. Clock/register output to primary output (clk -> Q2 -> out1)
4. Clock/register output to register input (clk -> Q1 -> D2)



Path-based commands

- Path startpoint = input pin or register clock pin
- Path endpoint = output pin or register data pin
- Path constraint
 - *set_max_delay* *-from* *from-list* *-to* *to-list* *value*
 - to/from-list = port, pin, clock or cell names
 - Clock in *from-list*: all paths affected by that clock
 - Clock in *to-list*: all related register data pins
 - Register data pin: FF1 or FF1/D
 - Can specify *-rise* and/or *-fall* times
 - Can add *-through P* to capture paths passing through P
 - Can specify [*all_outputs*] or [*all_inputs*]

Path delay examples

- Max delay requirements
 - *set_max_delay 10 -to out1 -from Reset*
 - *set_max_delay 5.1 -from {ff1 ff2} -to {o1 o2}*
 - *set_max_delay 3 -from busA[*] -to u1 / Z*
 - *set_max_delay 6 -to [all_outputs]*
 - If no “from list”, constrain paths from all start points
 - *set_max_delay 8 -from [all_inputs]*
 - If no “to list”, constrain paths to all end points
- Previous also applies to: *set_min_delay*

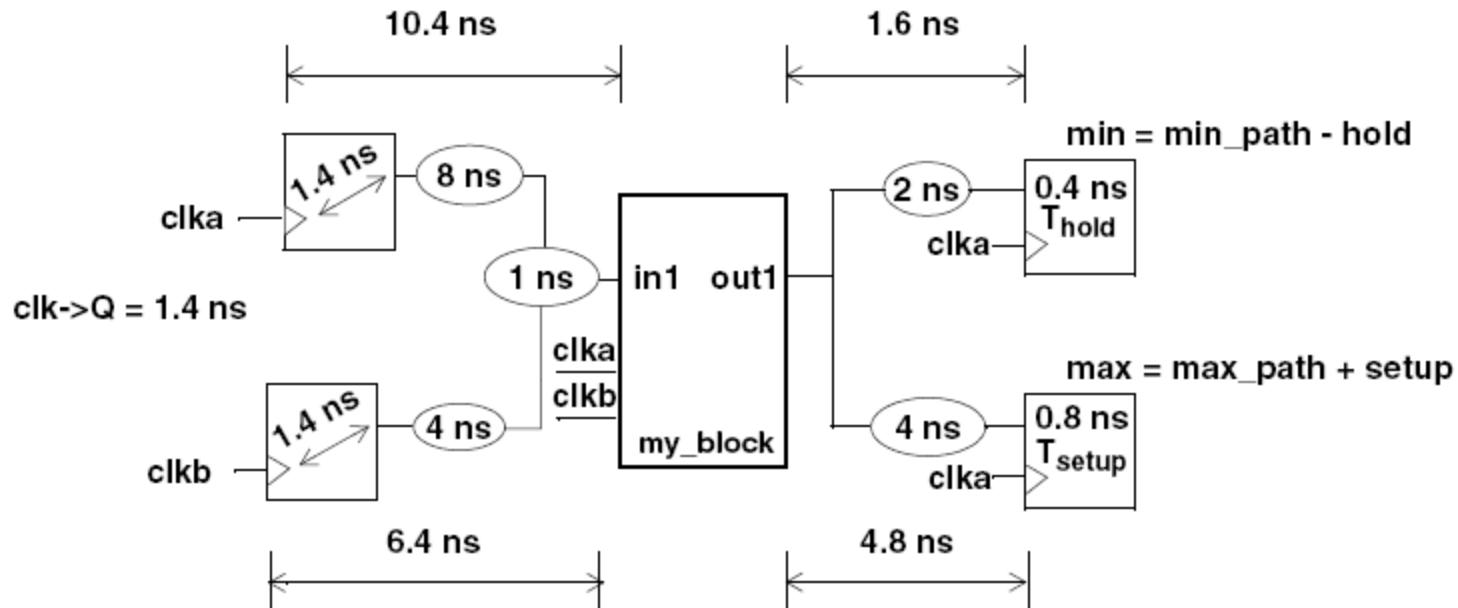
Clock specifications

- DC does not automatically imply clock signals
- Define required period/waveform for each clock
 - *create_clock ckname -period 5*
 - *create_clock ckname -period 5 -waveform {2 4}*
 - *period=5, rise at 2, fall at 4*
 - *create_clock -name ckname -period 5*
 - *creates a “virtual clock” associated with a port/pin*
- Clock latency = delay through clock network
 - *set_clock_latency 2.1 -rise CLK1*
 - *set_clock_latency 0.7 -source CLK1*
 - *from clock origin to clock pin*

Input and output delays

- Delay from clock edge through “external” logic to an input port or internal pin.
 - *set_input_delay 2.3 {in1 in2}*
 - default input delay = 0
- Time a signal is required at output port by external destination before a clock edge
 - external circuit logic delay + external ff setup time
 - *set_output_delay 7 -clock CLK1 [all_outputs]*
 - default output delay = 0

Sequential circuit example



```

create_clock -period 20 -waveform {5 15} clka
create_clock -period 30 -waveform {10 25} clkb
set_input_delay 10.4 -clock clka in1
set_input_delay 6.4 -clock clkb -add_delay in1
set_output_delay 1.6 -clock clka -min out1
set_output_delay 4.8 -clock clka -max out1
    
```

Required clock periods

Arrival at input pin from previous clock edge

Setup time of output pin from next clock edge

Compiling the design

- Compile (and optimize) the design
 - *compile -map_effort \$mapEffort1*
 - *design hierarchy preserved*
 - *map_effort = medium(default) or high*
 - *compile -ungroup_all -map_effort \$mapEffort1*
 - *design “flattened” (ungrouped – all levels collapsed)*
 - *compile -incremental_mapping -map_effort \$mapEffort2*
 - *work on it some more – incremental improvements*
- High-effort compile
 - *compile_ultra*
 - use on high-performance designs, tight time constraints
 - specify *-no_automgroup* to preserve hierarchy

Synthesis Output Files

- **Design.v** – verilog structural netlist
 - *change_names -rules verilog*
 - *write -format verilog -output file.v*
- **Design.sdf** – std delay file for timing simulation
 - *write_sdf -version 1.0 file.sdf*
- **Design.rep** – synthesis report (timing, area, etc)
 - *redirect file.rep {report_timing}*
 - *redirect -append file.rep {report_area -hier }*
- **Design.ddc** – Synopsys database format (to view in DV)
 - *write -format ddc -hierarchy -o file.ddc*
- **Design.sdc** – constraints for Encounter place/route
 - *write_sdc file.sdc*
- **Design.pow** – power estimate
 - *redirect file.pow { report_power }*

Modulo7 counter in *DesignVision*

