数字集成电路静态时序分析基础

邸志雄 博士, zxdi@home.swjtu.edu.cn

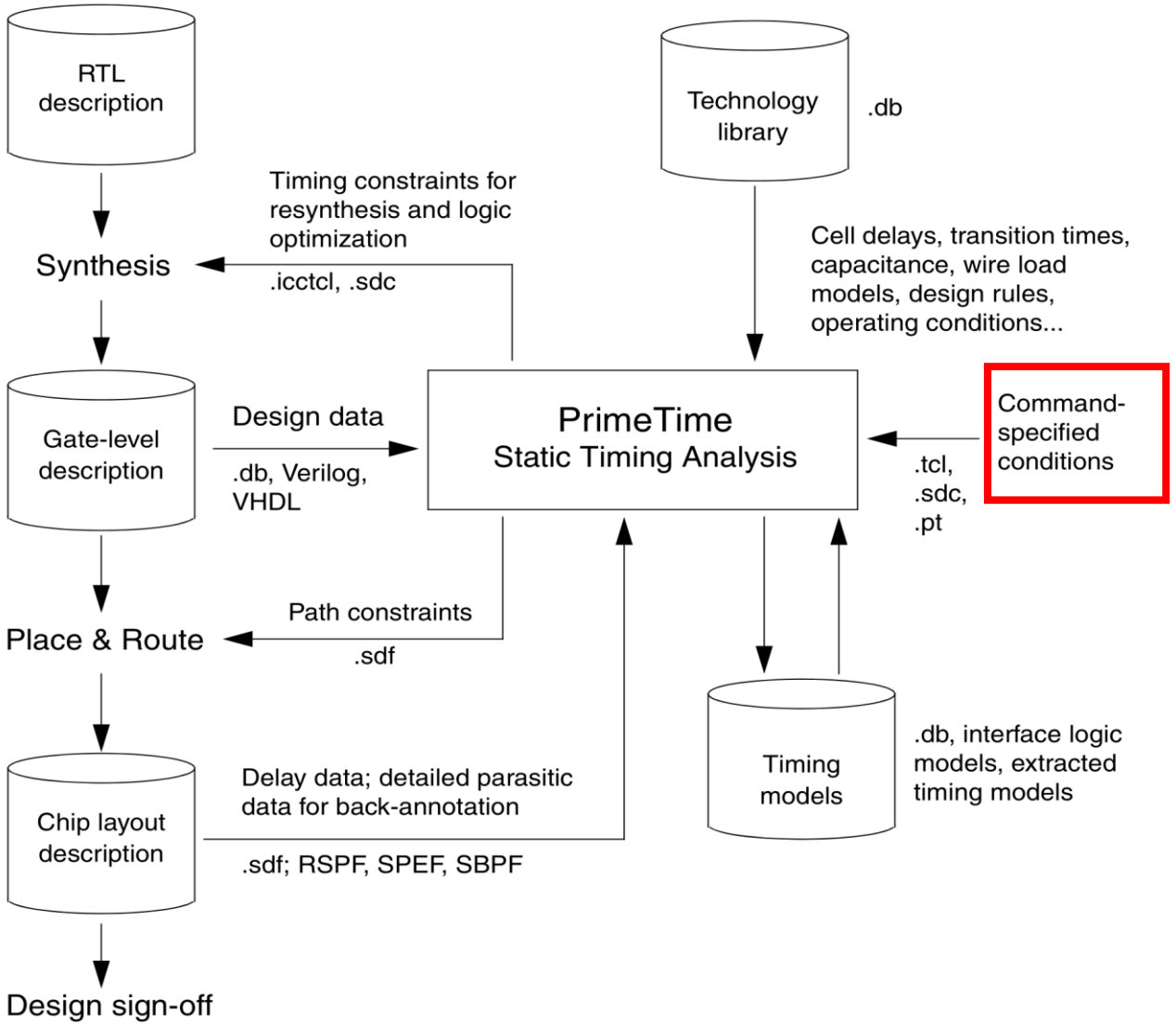
西南交通大学信息科学与技术学院

Part 4: Configuring the STA Environment

CONTENT

1. What is the STA Environment?
2. Specifying Clocks
3. Generated Clocks
4. Constraining Input Paths and Output Path
5. Design Rule Checks
6. Virtual Clocks
7. Refining the Timing Analysis

01-PrimeTime



What is the STA Environment?

- Specification of correct constraints is important in analyzing STA results.
- The design environment should be specified accurately so that STA analysis can identify all the timing issues in the design.
- Preparing for STA involves amongst others, setting up clocks, specifying IO timing characteristics, and specifying false paths and multicycle paths.

What is the STA Environment?

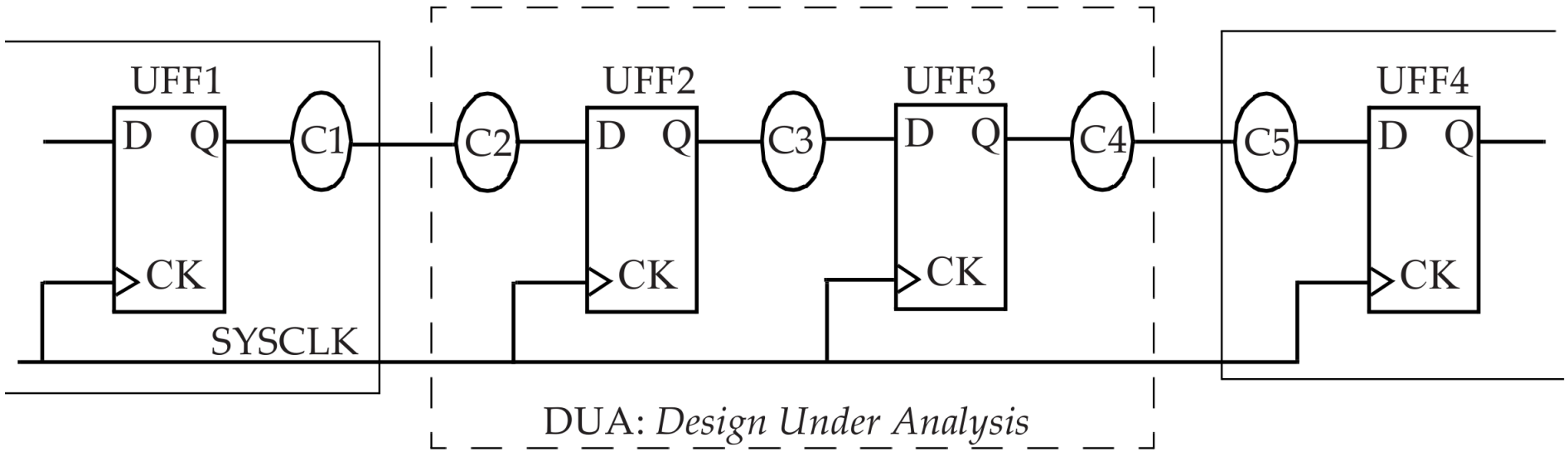


Figure 7-1 *A synchronous design.*

Part 4: Configuring the STA Environment

CONTENT

1. What is the STA Environment?
2. Specifying Clocks
3. Generated Clocks
4. Constraining Input Paths and Output Path
5. Design Rule Checks
6. Virtual Clocks
7. Refining the Timing Analysis

Specifying Clocks

To define a clock, we need to provide the following information:

i. Clock source: it can be a port of the design, or be a pin of a cell inside the design (typically that is part of a clock generation logic).

ii. Period: the time period of the clock.

iii. Duty cycle: the high duration (positive phase) and the low duration (negative phase).

iv. Edge times: the times for the rising edge and the falling edge.

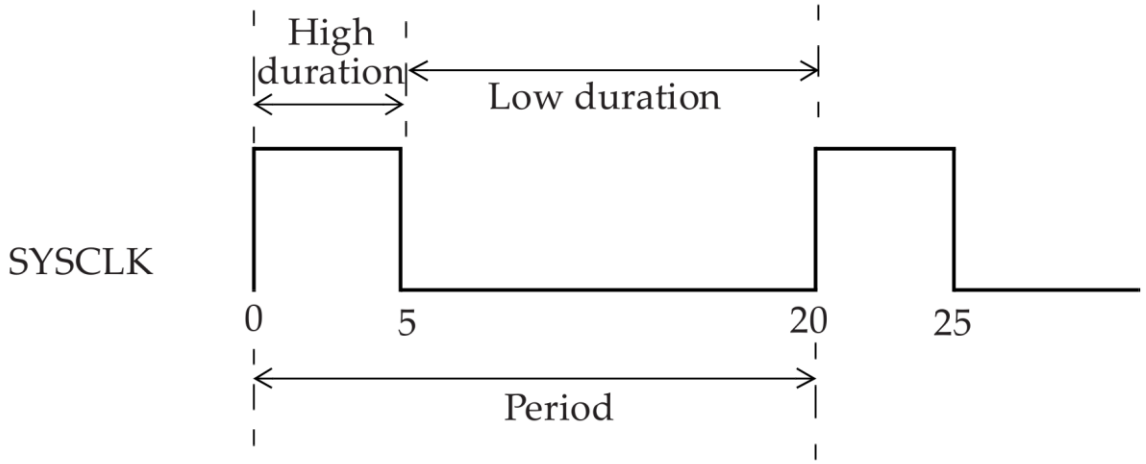


Figure 7-2 *A clock definition.*

Specifying Clocks

To define a clock, we need to provide the following information:

i. Clock source: it can be a port of the design, or be a pin of a cell inside the design (typically that is part of a clock generation logic).

ii. Period: the time period of the clock.

iii. Duty cycle: the high duration (positive phase) and the low duration (negative phase).

iv. Edge times: the times for the rising edge and the falling edge.

```
create_clock -name SYSCLK -period 20 \
-waveform {0 5}  [get_ports 2 SCLK]
```

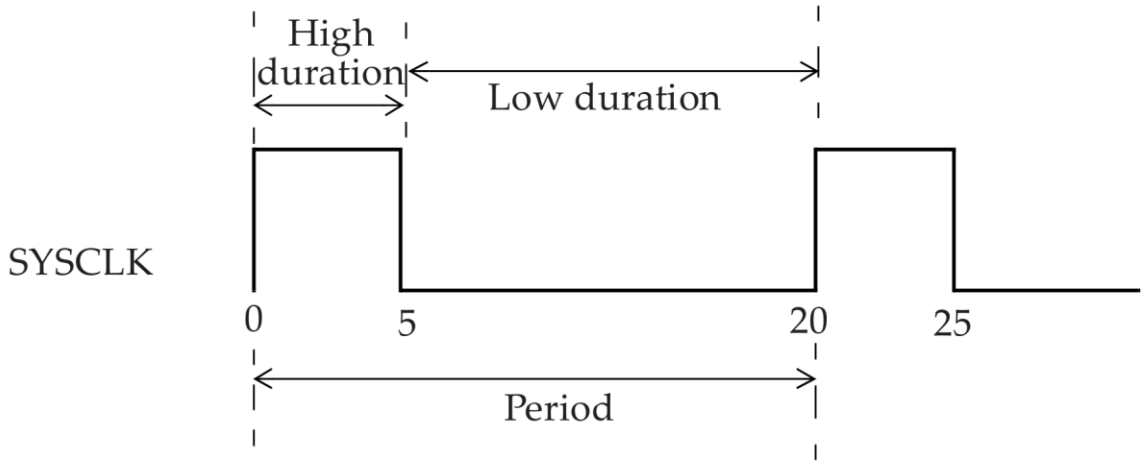


Figure 7-2 *A clock definition.*

Specifying Clocks

```
create_clock -period 5 [get_ports SCAN_CLK]
```

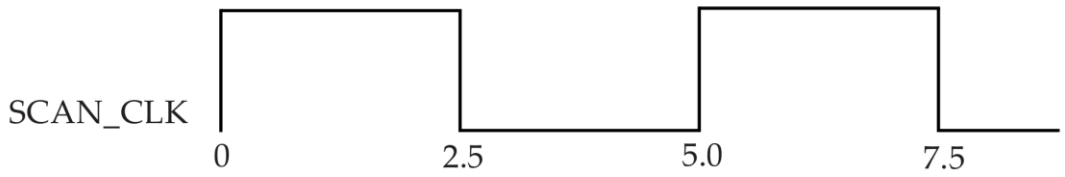


Figure 7-3 *Clock specification example.*

```
create_clock -name BDYCLK -period 15 \
-waveform {5 12} [get_ports GBLCLK]
```

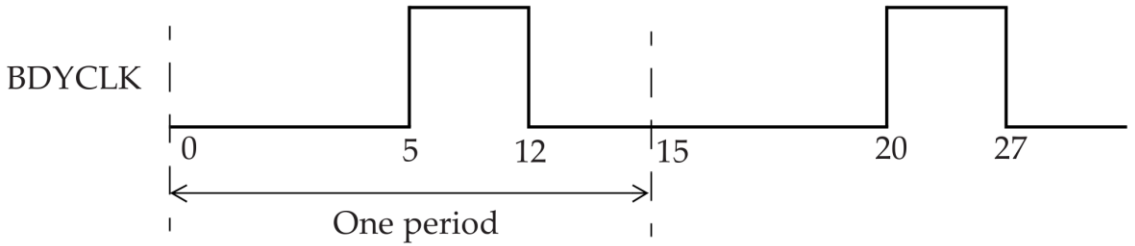


Figure 7-4 *Clock specification with arbitrary edges.*

Specifying Clocks-Clock Uncertainty

- The timing uncertainty of a clock period can be specified using the set_clock_uncertainty specification.
- The uncertainty can be used to model various factors that can ***reduce*** the effective clock period.
- These factors can be *the clock jitter and any other pessimism* that one may want to include for timing analysis.

Specifying Clocks-Clock Uncertainty

```
set_clock_uncertainty -setup 0.2 [get_clocks CLK_CONFIG]
set_clock_uncertainty -hold 0.05 [get_clocks CLK_CONFIG]
```

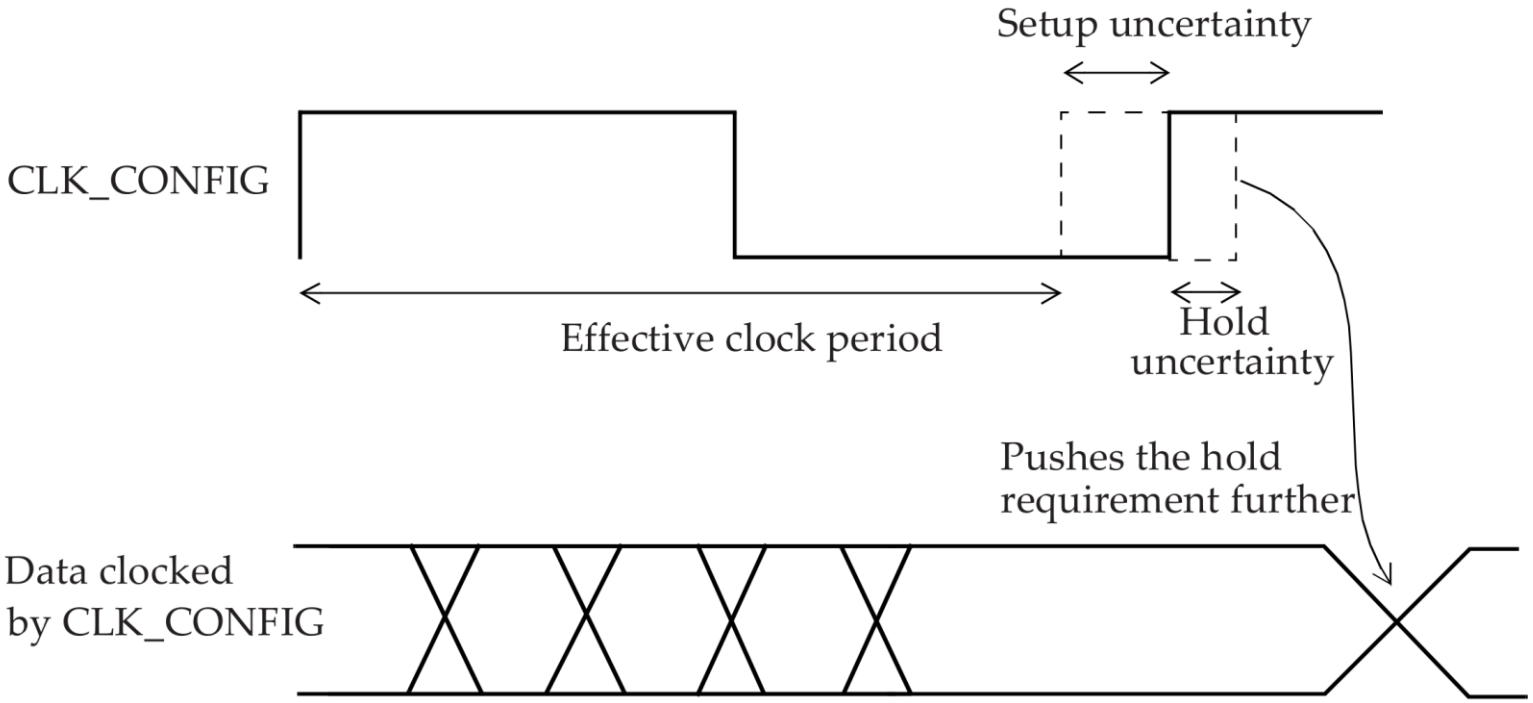


Figure 7-7 *Specifying clock uncertainty.*

Specifying Clocks-Clock Uncertainty

```
set_clock_uncertainty -from VIRTUAL_SYS_CLK -to SYS_CLK -hold 0.05
set_clock_uncertainty -from VIRTUAL_SYS_CLK -to SYS_CLK -setup 0.3
set_clock_uncertainty -from SYS_CLK -to CFG_CLK -hold 0.05
set_clock_uncertainty -from SYS_CLK -to CFG_CLK -setup 0.1
```

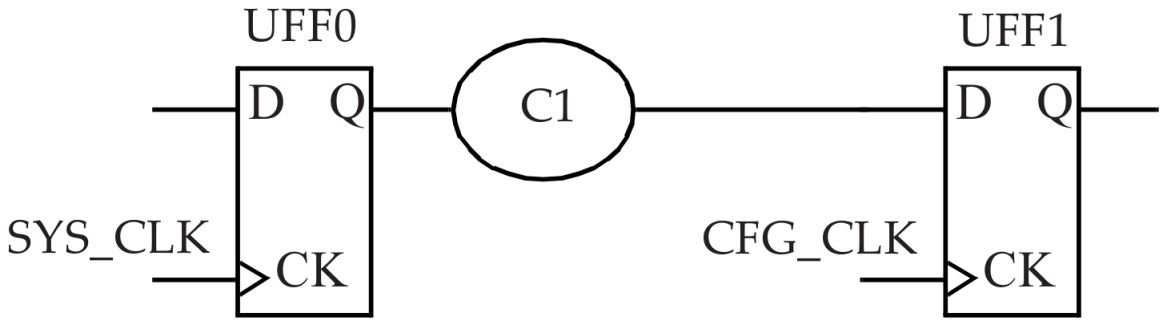
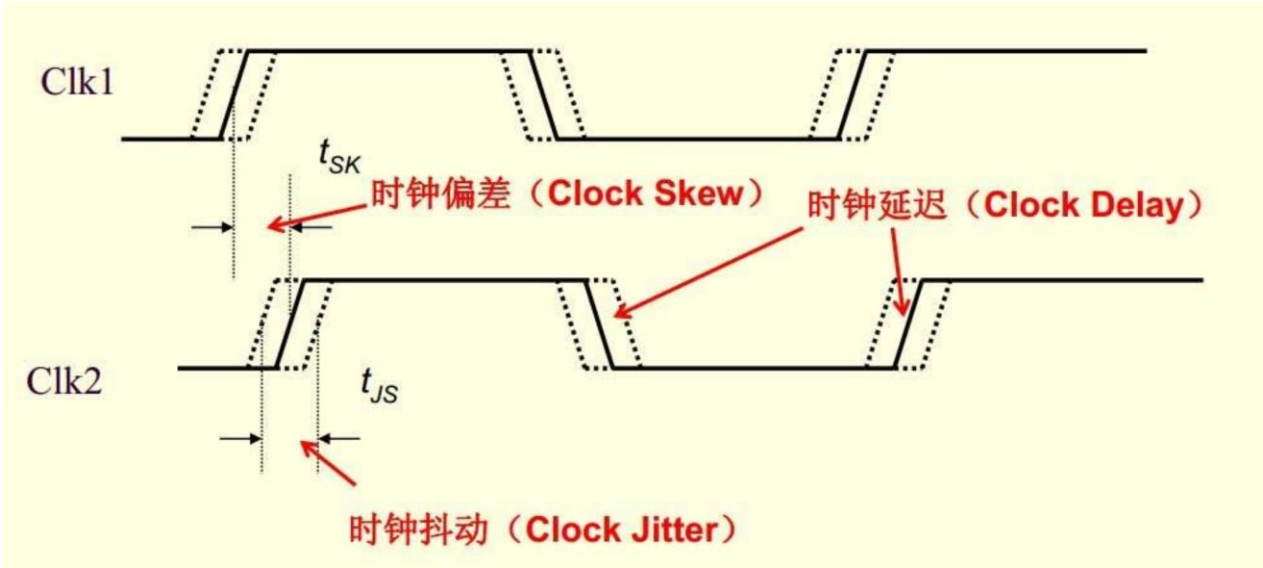
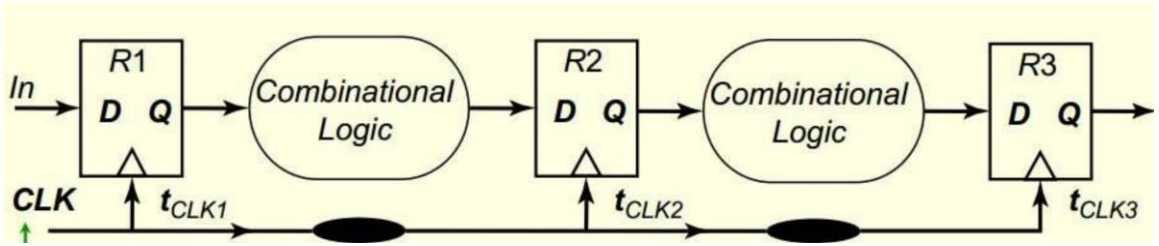


Figure 7-8 *Inter-clock paths.*

Specifying Clocks-Clock Uncertainty



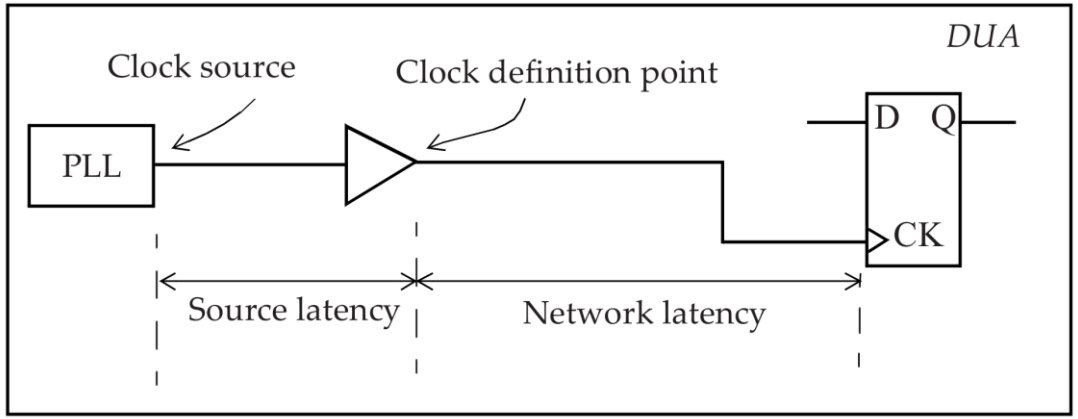


Specifying Clocks-Clock Latency

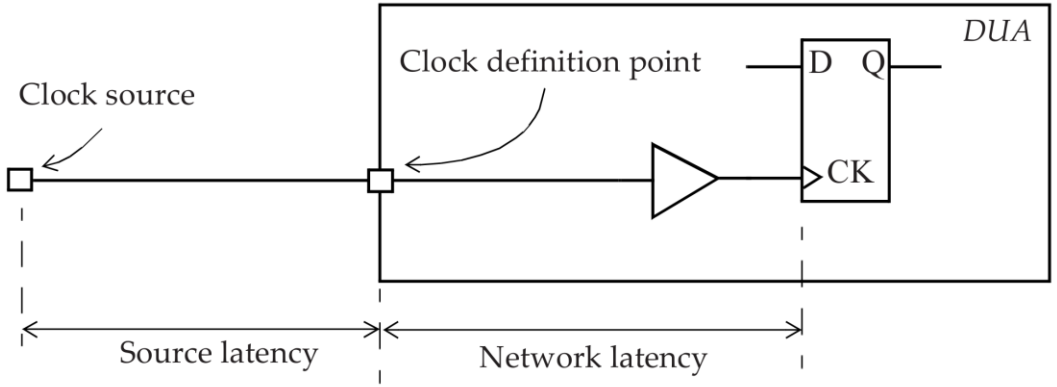
There are two types of clock latencies: **network latency** and **source latency**.

- **Network latency** is the delay from the clock definition point (create_clock) to the clock pin of a flip-flop.
- **Source latency**, also called insertion delay, is the delay from the clock source to the clock definition point.

Source latency could represent either on-chip or off-chip latency. The total clock latency at the clock pin of a flip-flop is the sum of the source and network latencies.



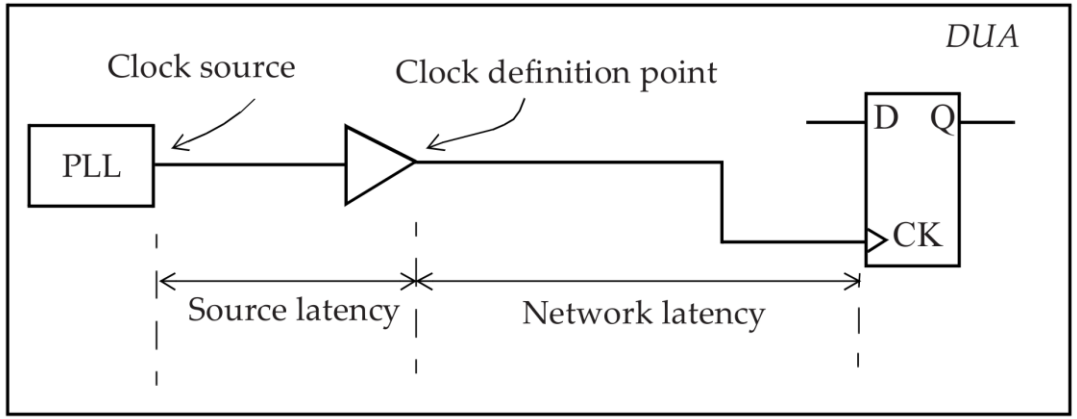
(a) On-chip clock source.



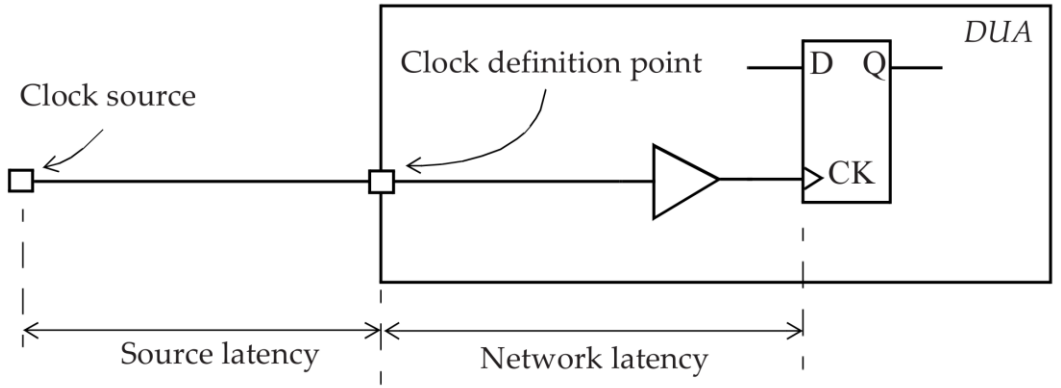
(b) Off-chip clock source.

Specifying Clocks-Clock Latency

 # Specify a network latency (no -source option) of 0.8ns for rise, fall, max and min:
set_clock_latency 0.8 [get_clocks CLK_CONFIG]
 # Specify a source latency:
set_clock_latency 1.9 -source [get_clocks SYS_CLK]
 # Specify a min source latency:
set_clock_latency 0.851 -source -min [get_clocks CFG_CLK]
 # Specify a max source latency:
set_clock_latency 1.322 -source -max [get_clocks CFG_CLK]



(a) On-chip clock source.



(b) Off-chip clock source.

Specifying Clocks-Clock Latency

- One important distinction to observe between source and network latency is that once a clock tree is built for a design, the network latency can be ignored (assuming set_propagated_clock command is specified).
- However, the source latency remains even after the clock tree is built.
- The network latency is an **estimate** of the delay of the clock tree **prior** to clock tree synthesis.
- After clock tree synthesis, the total clock latency from clock source to a clock pin of a flip-flop is the source latency plus the actual delay of the clock tree from the clock definition point to the flip-flop.

Part 4: Configuring the STA Environment

CONTENT

1. What is the STA Environment?
2. Specifying Clocks
3. Generated Clocks
4. Constraining Input Paths and Output Path
5. Design Rule Checks
6. Virtual Clocks
7. Refining the Timing Analysis

Generated Clocks

- A generated clock is a clock derived from a master clock. A master clock is a clock defined using the create_clock specification.
- When a new clock is generated in a design that is based on a master clock, the new clock can be defined as a generated clock.

This definition is needed as STA does not know that the clock period has changed at the output of the divide-by logic, and more importantly what the new clock period is.

Generated Clocks

```
create_clock -name CLKP 10 [get_pins UPLL0/CLKOUT]
  # Create a master clock with name CLKP of period 10ns
  # with 50% duty cycle at the CLKOUT pin of the PLL.
create_generated_clock -name CLKPDIV2 -source UPLL0/CLKOUT \
  -divide_by 2 [get_pins UFF0/Q]
  # Creates a generated clock with name CLKPDIV2 at the Q
  # pin of flip-flop UFF0. The master clock is at the CLKOUT
  # pin of PLL. And the period of the generated clock is double
  # that of the clock CLKP, that is, 20ns.
```

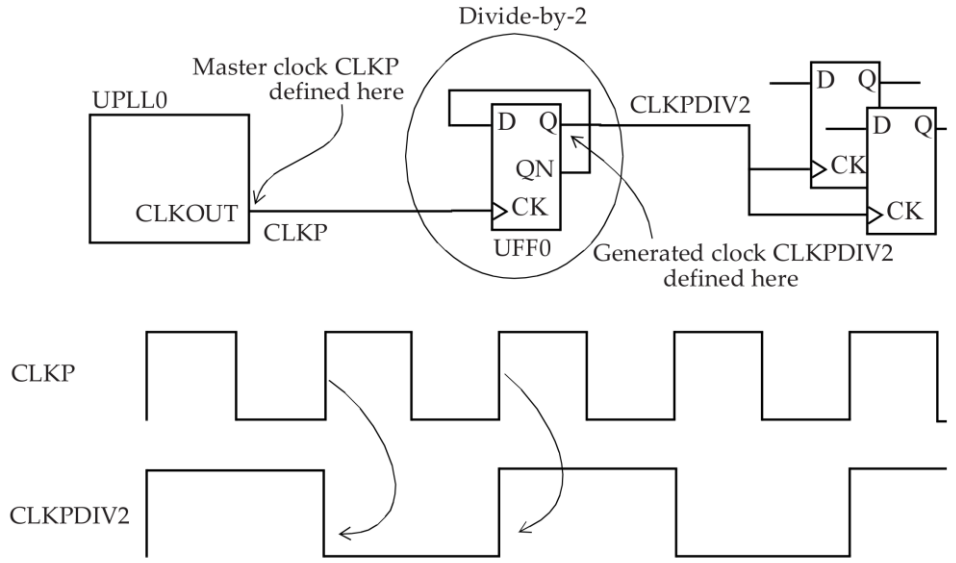


Figure 7-10 *Generated clock at output of divider.*

Generated Clocks

Can a new clock, that is, a master clock, be defined at the output of the flipflop instead of a generated clock? The answer is yes, that it is indeed possible. However, there are some disadvantages. Defining a master clock instead of a generated clock creates a new clock domain.

Defining the new clock as a generated clock does not create a new clock domain, and the generated clock is considered to be in phase with its master clock. The generated clock does not require additional constraints to be developed. Thus, one must attempt to define a new internally generated clock as a generated clock instead of deciding to declare it as another master clock.

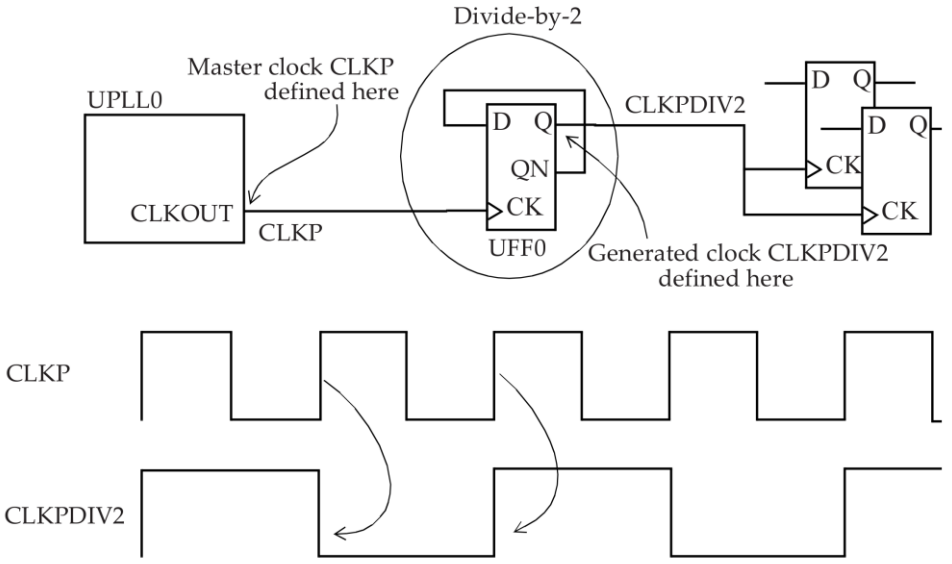


Figure 7-10 *Generated clock at output of divider.*

Generated Clocks

Another important difference between a master clock and a generated clock is the notion of clock origin. In a master clock, the origin of the clock is at the point of definition of the master clock. In a generated clock, the clock origin is that of the master clock and not that of the generated clock.

This implies that in a clock path report, the start point of a clock path is always the master clock definition point. This is a big advantage of a generated clock over defining a new master clock as the source latency is not automatically included for the case of a new master clock.

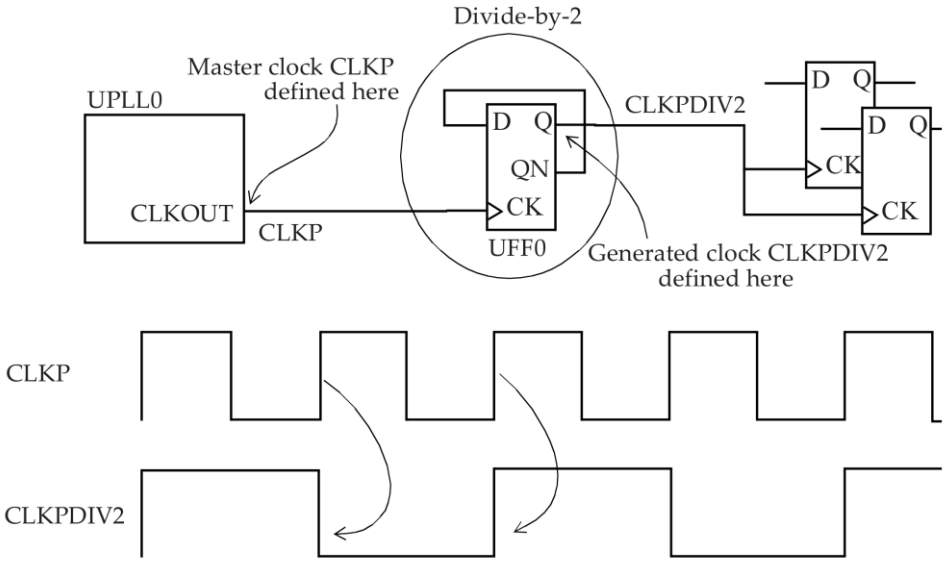
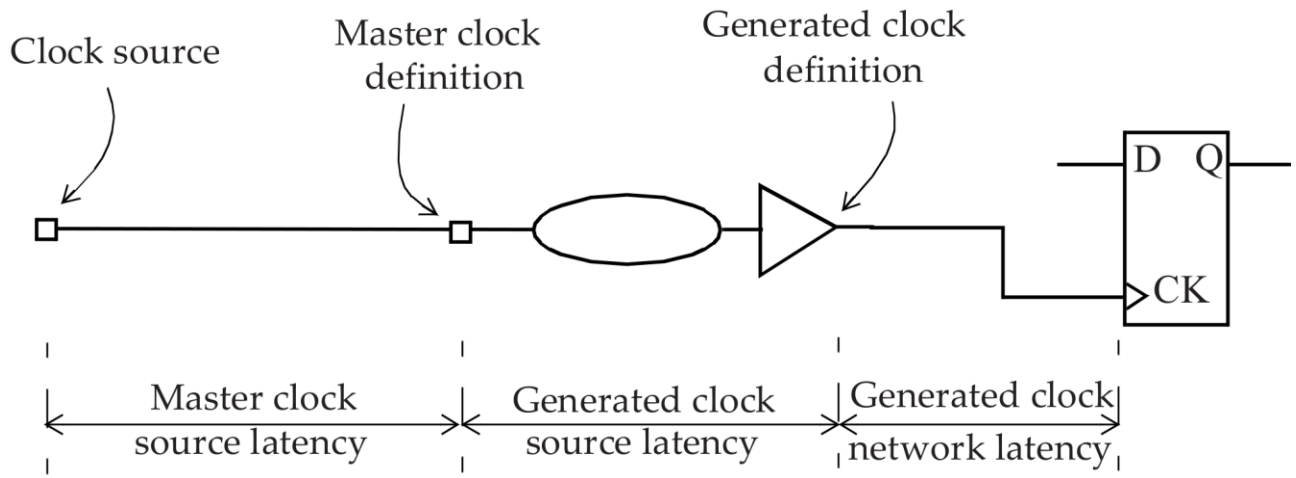


Figure 7-10 *Generated clock at output of divider.*

Generated Clocks

Clock latencies can be specified for generated clocks as well.

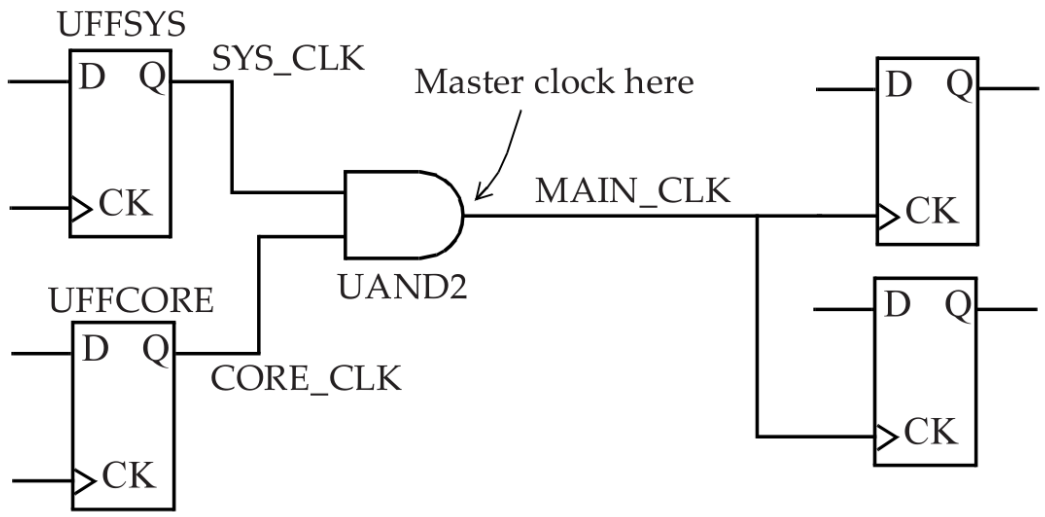
A source latency specified on a generated clock specifies the latency from the definition of the master clock to the definition of the generated clock.

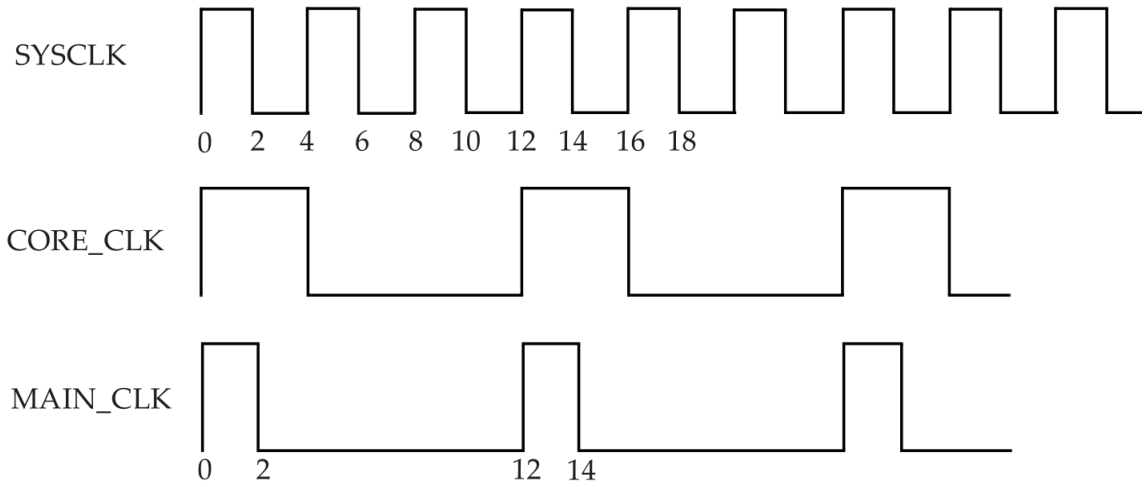


The total clock latency to a clock pin of a flop-flop being driven by a generated clock is thus the sum of the source latency of the master clock, the source latency of the generated clock and the network latency of the generated clock.

Generated Clocks

If the input to the and cell are both clocks, then it is safe to define a new main clock at the output of the and cell, since it is highly unlikely that the output of the cell has any phase relationship with either of the input clocks.

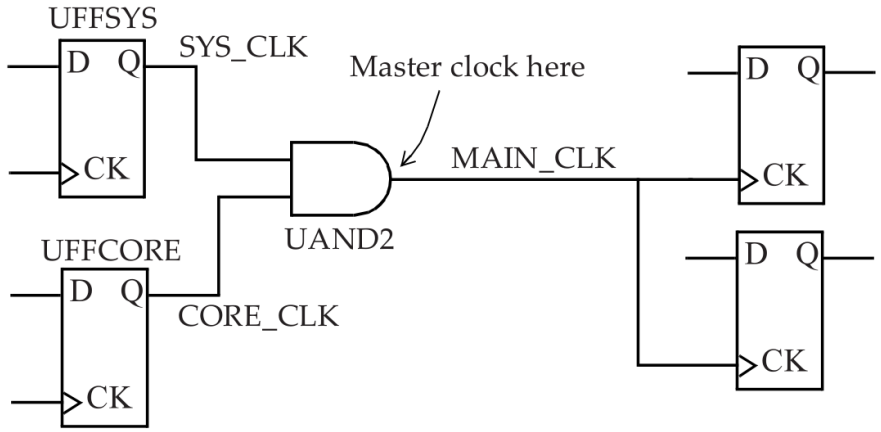




Generated Clocks

If the input to the and cell are both clocks, then it is safe to define a new main clock at the output of the and cell, since it is highly unlikely that the output of the cell has any phase relationship with either of the input clocks.

```
create_clock -name SYS_CLK -period 4 -waveform {0 2} \
  [get_pins UFFSYS/Q]
create_clock -name CORE_CLK -period 12 -waveform {0 4} \
  [get_pins UFFCORE/Q]
create_clock -name MAIN_CLK -period 12 -waveform {0 2} \
  [get_pins UAND2/Z]
  # Create a master clock instead of a generated clock
  # at the output of the and cell.
```



Generated Clocks

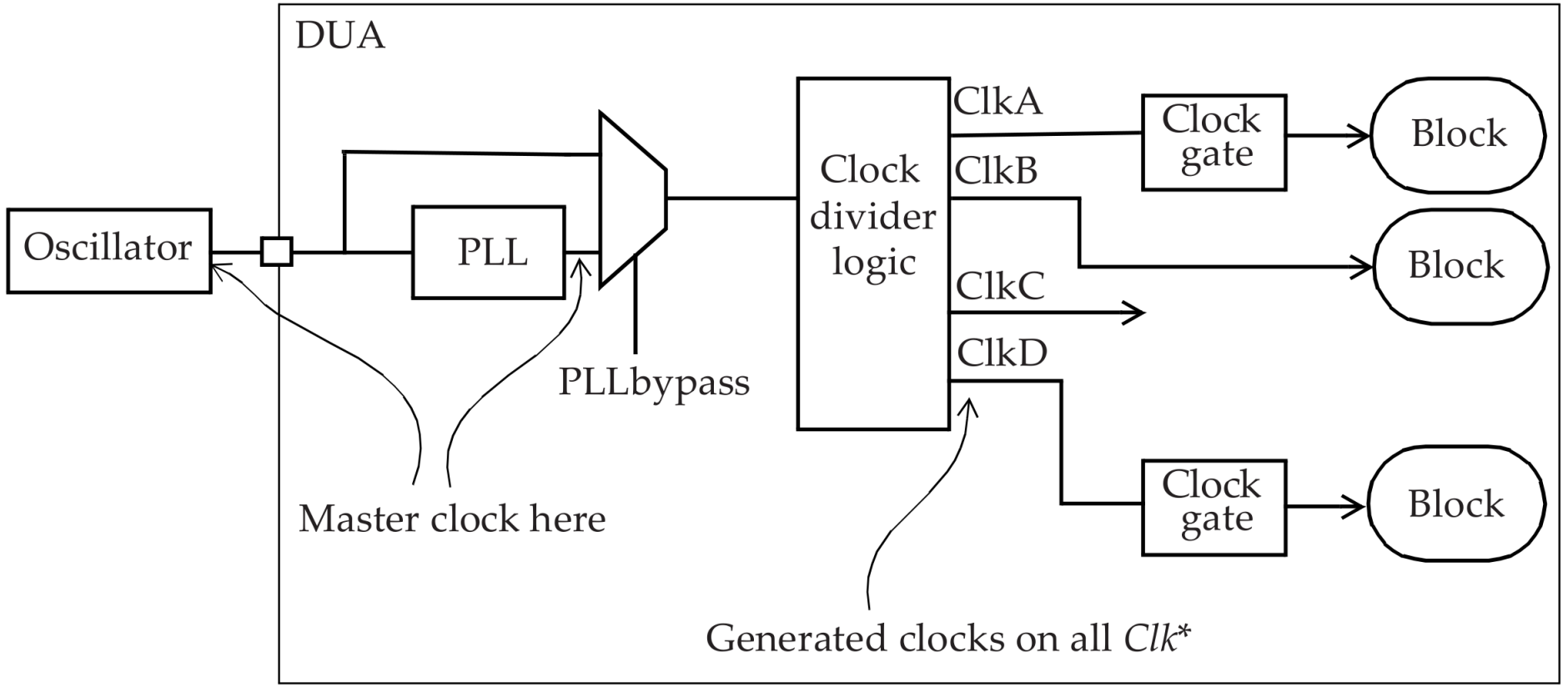


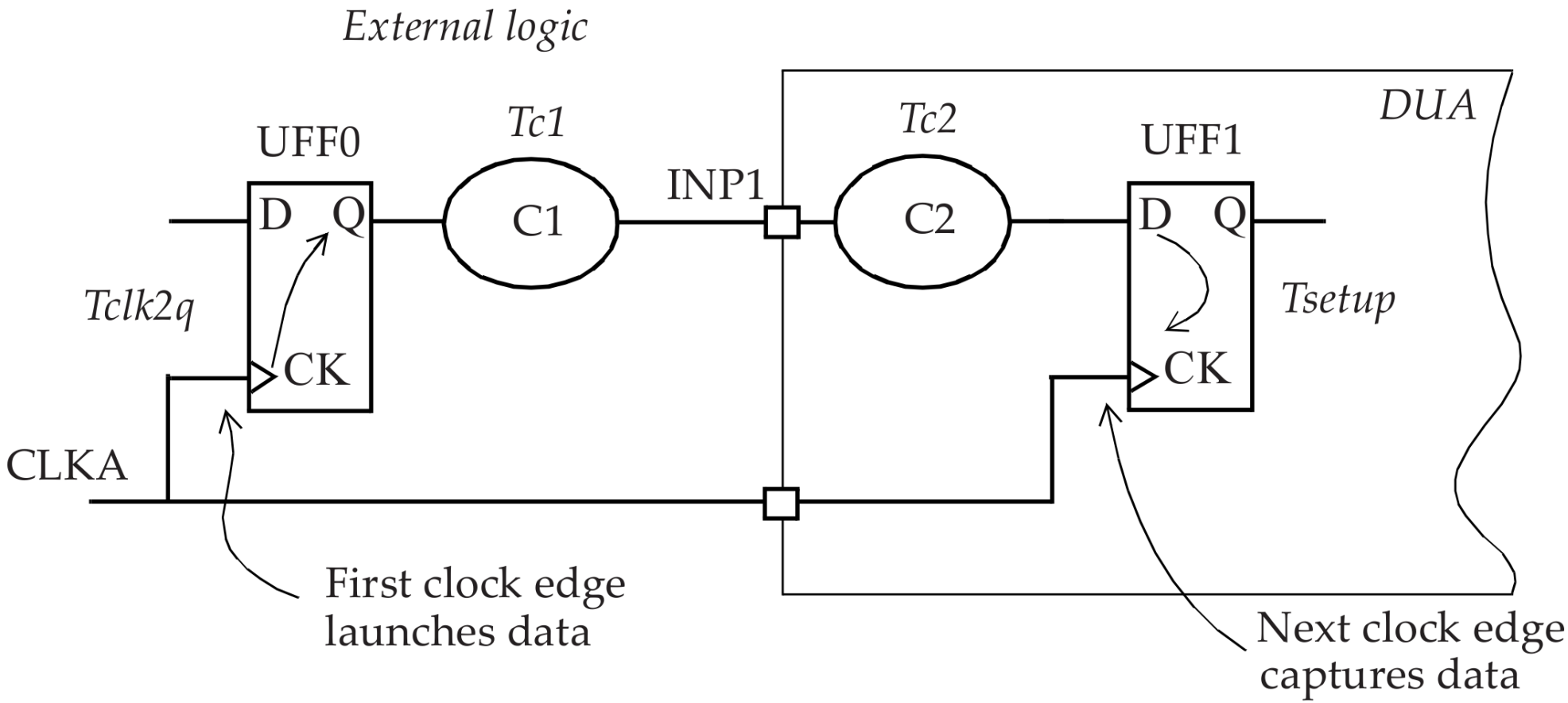
Figure 7-20 *Clock distribution in a typical ASIC.*

Part 4: Configuring the STA Environment

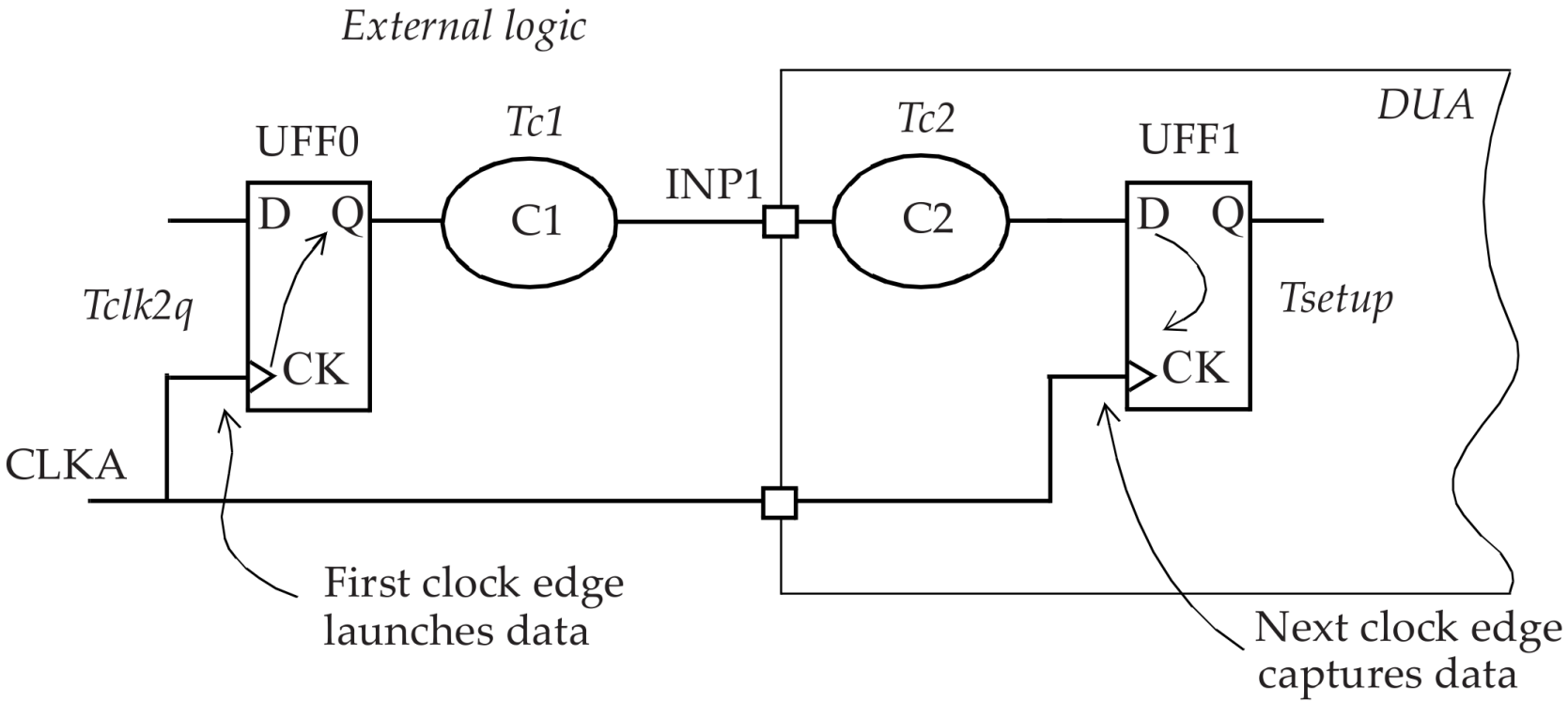
CONTENT

1. What is the STA Environment?
2. Specifying Clocks
3. Generated Clocks
4. Constraining Input Paths and Output Path
5. Design Rule Checks
6. Virtual Clocks
7. Refining the Timing Analysis

Constraining Input Paths



Constraining Input Paths




```
set Tclk2q 0.9

set Tc1 0.6

set_input_delay -clock CLKA -max [expr Tclk2q + Tc1] [get_ports INP1]
```

Constraining Input Paths

```
create_clock -period 15 -waveform {5 12}
[get_ports CLKP]
set_input_delay -clock CLKP -max 6.7
[get_ports INPA]
set_input_delay -clock CLKP -min 3.0
[get_ports INPA]
```

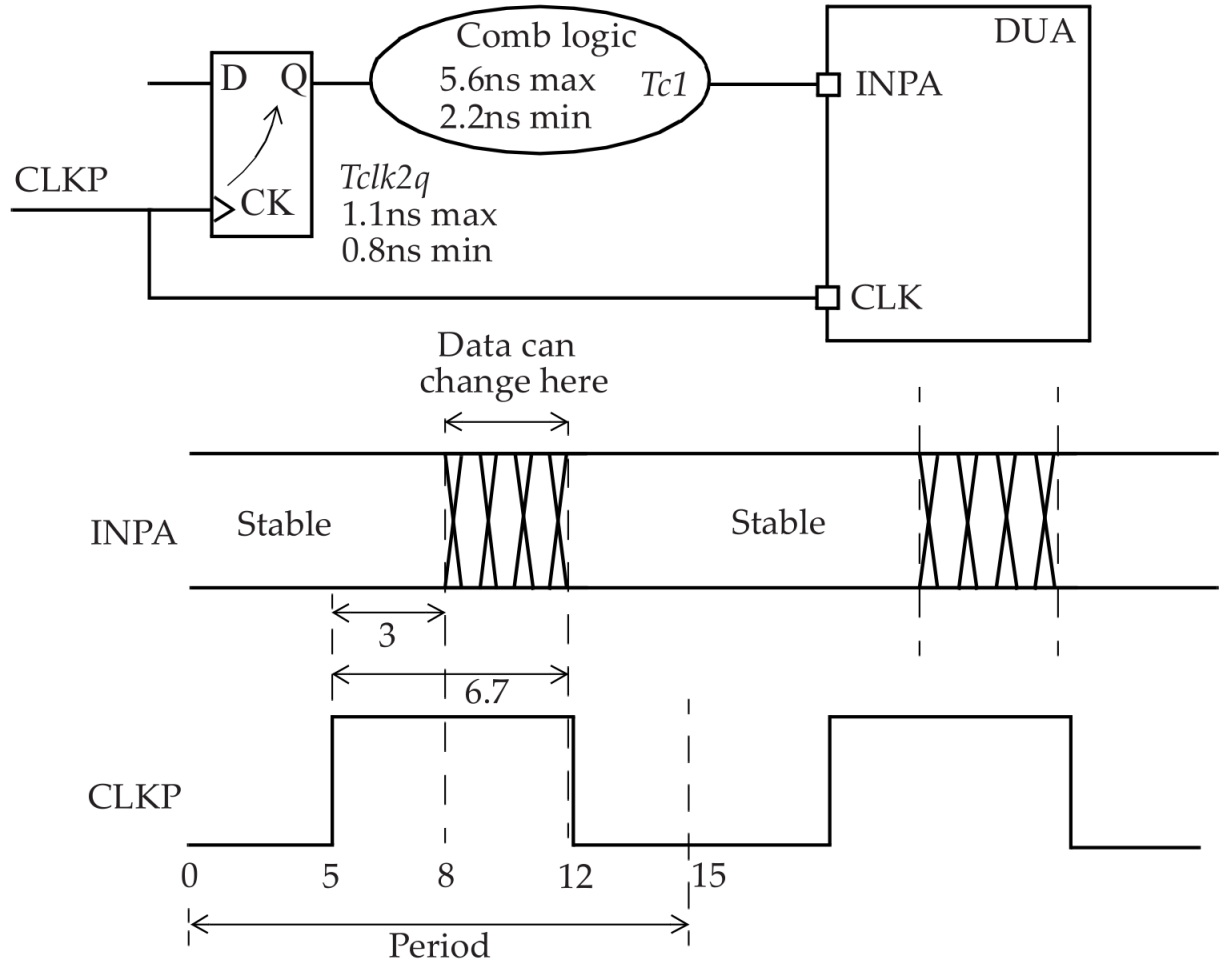


Figure 7-22 *Max and min delays on input port.*

Constraining Output Paths

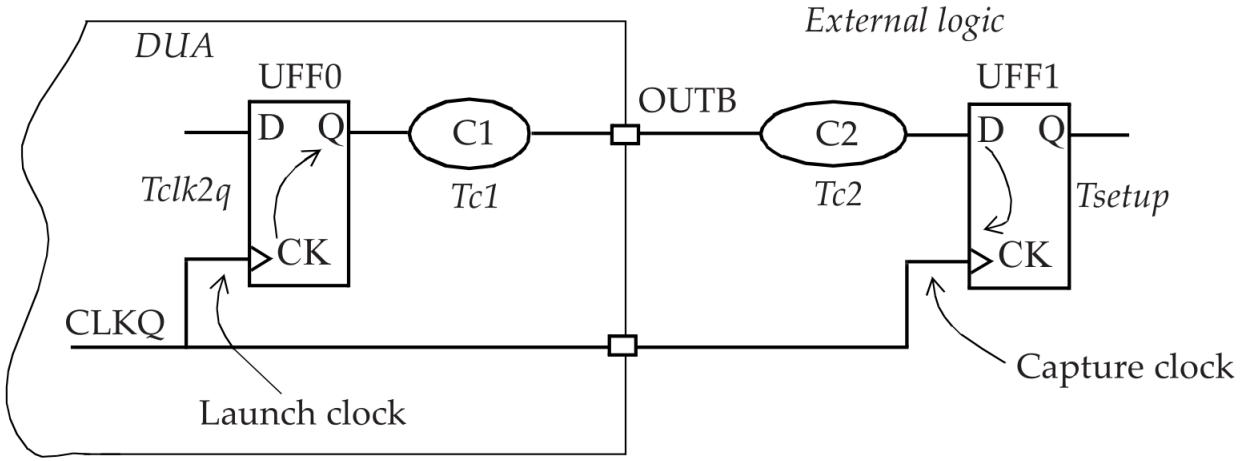


Figure 7-23 *Output port timing path for example A.*

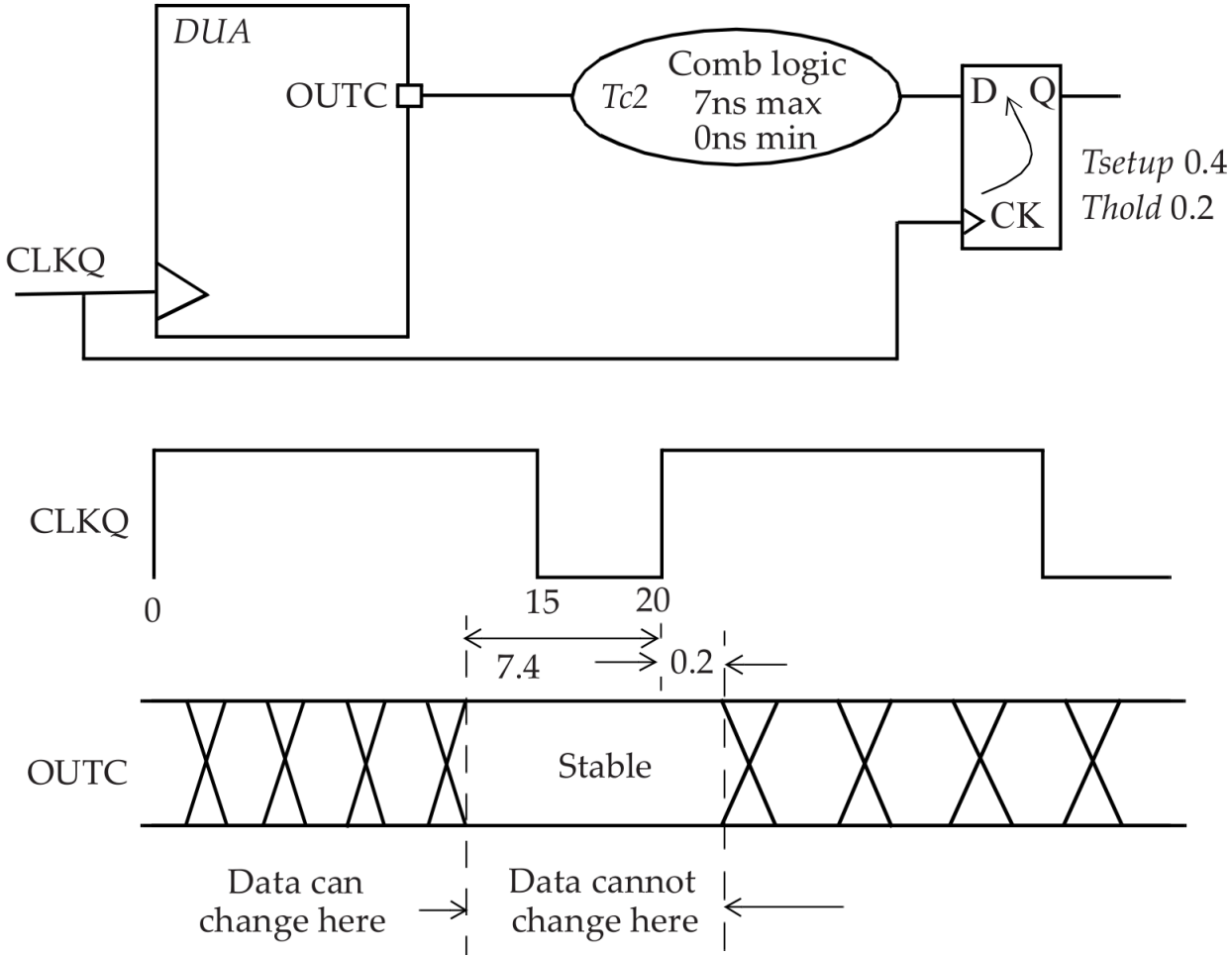
```
set Tc2 3.9

set Tsetup 1.1

set_output_delay -clock CLKQ -max [expr Tc2 + Tsetup] [get_ports OUTB]
```

Constraining Output Paths

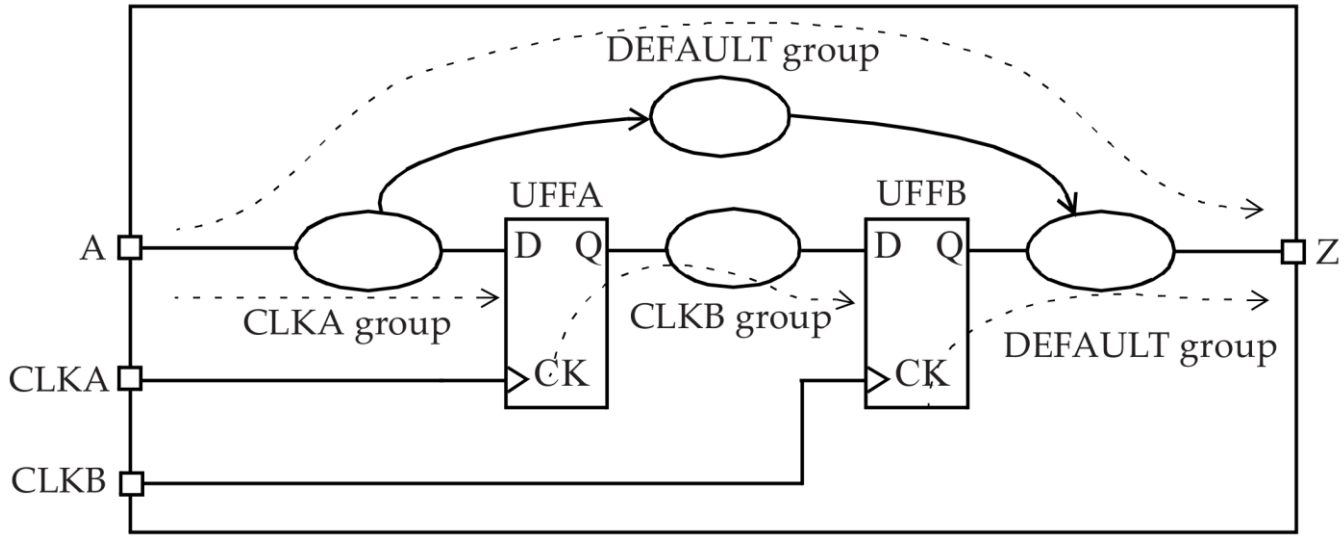
```
create_clock -period 100 -waveform {5 55}
[get_ports MCLK]
set_input_delay 25 -max -clock MCLK
[get_ports DATAIN]
set_input_delay 5 -min -clock MCLK
[get_ports DATAIN]
set_output_delay 20 -max -clock MCLK
[get_ports DATAOUT]
set_output_delay -5 -min -clock MCLK
[get_ports DATAOUT]
```



Timing Path Groups

Timing paths are sorted into path groups by the clock associated with the **endpoint** of the path. Thus, each clock has a set of paths associated with it.

There is also a default path group that includes all non-clocked (asynchronous) paths.



Modeling of External Attributes

While create_clock, set_input_delay and set_output_delay are enough to constrain all paths in a design for performing timing analysis, these are not enough to obtain accurate timing for the IO pins of the block.

The following attributes are also required to accurately model the environment of a design.

Modeling of External Attributes

For inputs, one needs to specify the slew at the input. This information can be provided using:

- set_drive
- set_driving_cell
- set_input_transition

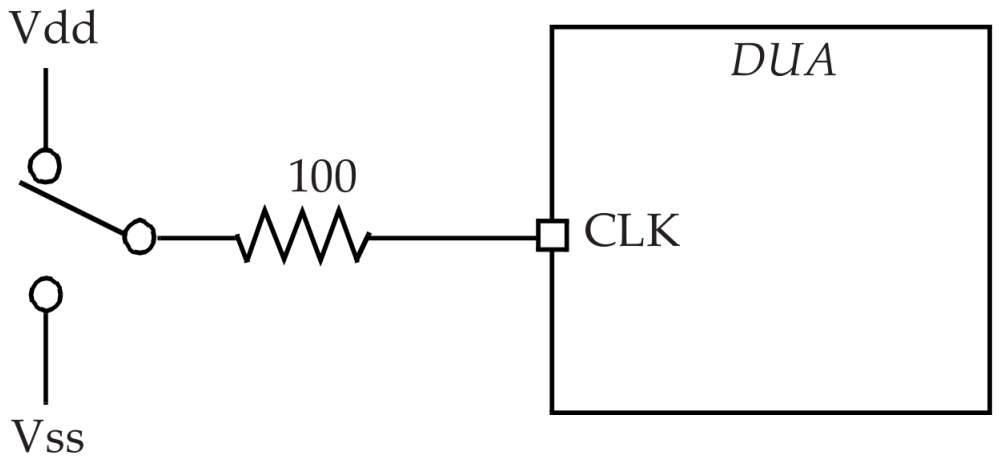
For outputs, one needs to specify the capacitive load seen by the output pin. This is specified by using the following specification:

- set_load

Modeling of External Attributes

The set_drive explicitly specifies a value for the drive resistance at the input pin of the DUA. The smaller the drive value, the higher the drive strength.

A resistance value of 0 implies an infinite drive strength.



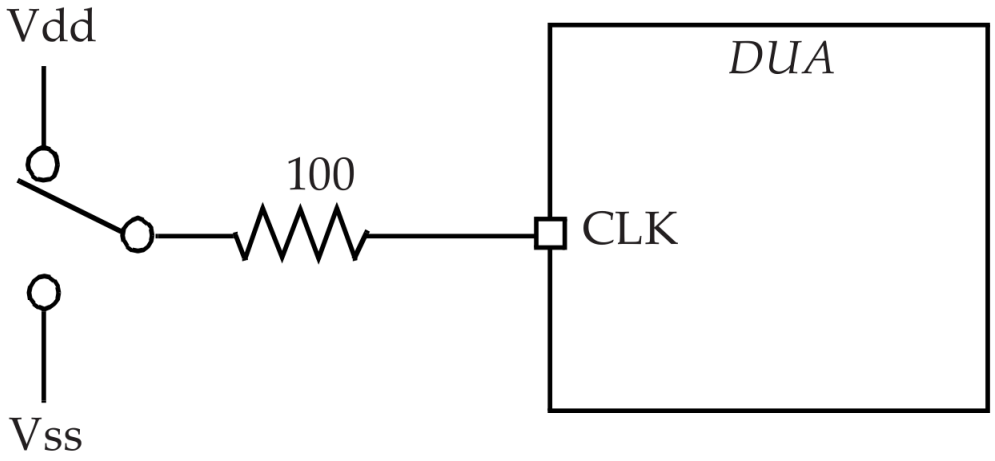

```
set_drive 100 UCLK
# Specifies a drive resistance of 100 on input UCLK.
```

```
# Rise drive is different from fall drive:
set_drive -rise 3 [all_inputs]
set_drive -fall 2 [all_inputs]
```

Modeling of External Attributes

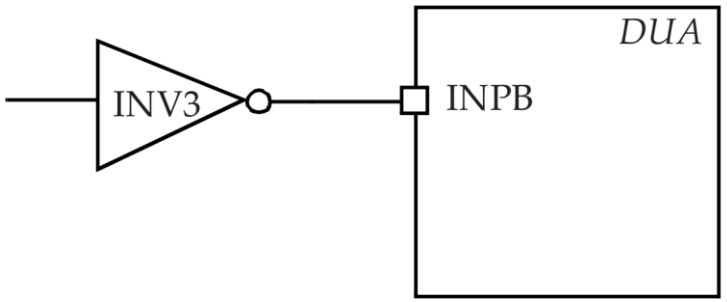
The drive of an input port is used to calculate the transition time at the first cell. The drive value specified is also used to compute the delay from the input port to the first cell in the presence of any RC interconnect.

Delay_to_first_gate = (drive * load_on_net) + interconnect_delay



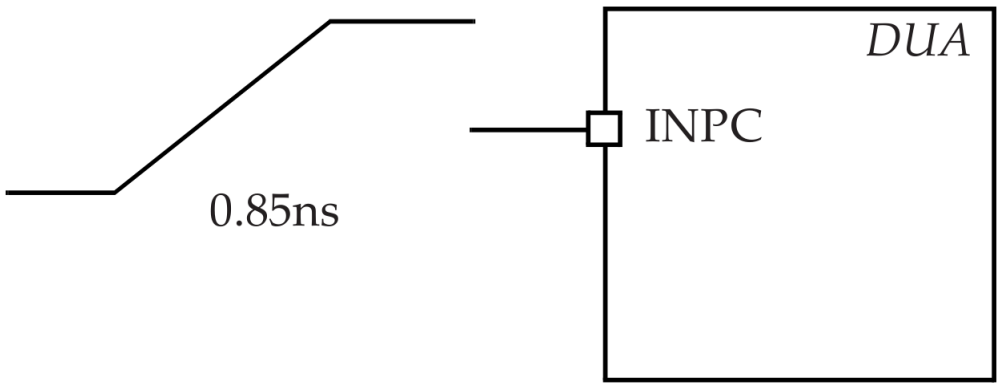
Modeling of External Attributes

The set_driving_cell specification offers a more convenient and accurate approach in describing the drive capability of a port.




```
set_driving_cell -lib_cell INV3  -library slow [get_ports INPB]
# The input INPB is driven by an INV3 cell from library slow.
set_driving_cell -lib_cell INV2 -library tech13g [all_inputs]
# Specifies that the cell INV2 from a library tech13g is the driving cell for all inputs.
set_driving_cell -lib_cell BUFFD4 -library tech90gwc [get_ports {testmode[3]}]
# The input testmode[3] is driven by a BUFFD4 cell from library tech90gwc.
```

Modeling of External Attributes




```
set_input_transition 0.85 [get_ports INPC]
# Specifies an input transition of 850ps on port INPC.
set_input_transition 0.6 [all_inputs]
# Specifies a transition of 600ps on all input ports.
set_input_transition 0.25 [get_ports SD_DIN*]
# Specifies a transition of 250ps on all ports with pattern SD_DIN*.
# Min and max values can optionally be specified using the -min and -max options.
```

Modeling of External Attributes

- In summary, a slew value at an input is needed to determine the delay of the first cell in the input path.
- In the absence of this specification, an ideal transition value of 0 is assumed, which may not be realistic.

Modeling of External Attributes

The set_load specification places a capacitive load on output ports to model the external load being driven by the output port.

By default, the capacitive load on ports is 0. The load can be specified as an explicit capacitance value or as an input pin capacitance of a cell.

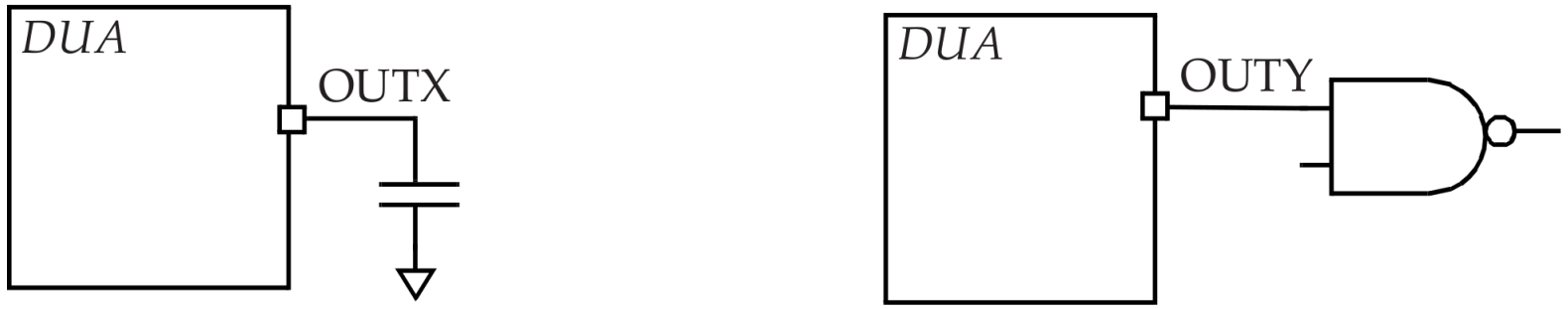


Figure 7-31 *Capacitive load on output port.*

Modeling of External Attributes

```
set_load 5 [get_ports OUTX]
# Places a 5pF load on output port OUTX.
set_load 25 [all_outputs]
# Sets 25pF load capacitance on all outputs.
set_load -pin_load 0.007 [get_ports {shift_write[31]}]
# Place 7fF pin load on the specified output port.
# A load on the net connected to the port can be specified using the -wire_load option.
# If neither -pin_load nor -wire_load option is used, # the default is the -pin_load option.
```

It is important to specify the load on outputs since this value impacts the delay of the cell driving the output. In the absence of such a specification, a load of 0 is assumed which may not be realistic.

```
set_load [get_attribute [get_lib_pins tech_lib/NAND2/A] pin_capacitance] [all_outputs]
```

Part 4: Configuring the STA Environment

CONTENT

1. What is the STA Environment?
2. Specifying Clocks
3. Generated Clocks
4. Constraining Input Paths and Output Path
5. Design Rule Checks
6. Virtual Clocks
7. Refining the Timing Analysis

Design Rule Checks

Two of the frequently used design rules for STA are max transition and max capacitance. These rules check that all ports and pins in the design meet the specified limits for transition time 1 and capacitance.

These limits can be specified using:

- set_max_transition
- set_max_capacitance

```
set_max_transition 0.6 IOBANK
# Sets a limit of 600ps on IOBANK.
set_max_capacitance 0.5 [current_design]
# Max capacitance is set to 0.5pf on all nets in current design.
```

Design Rule Checks

There are other design rule checks that can also be specified for a design.

These are: set_max_fanout (specifies a fanout limit on all pins in design), set_max_area (for a design);

however these checks apply for synthesis and not for STA.

Part 4: Configuring the STA Environment

CONTENT

1. What is the STA Environment?
2. Specifying Clocks
3. Generated Clocks
4. Constraining Input Paths and Output Path
5. Design Rule Checks
6. Virtual Clocks
7. Refining the Timing Analysis

Virtual Clocks

A virtual clock is a clock that exists but is not associated with any pin or port of the design. It is used as a reference in STA analysis to specify input and output delays relative to a clock.

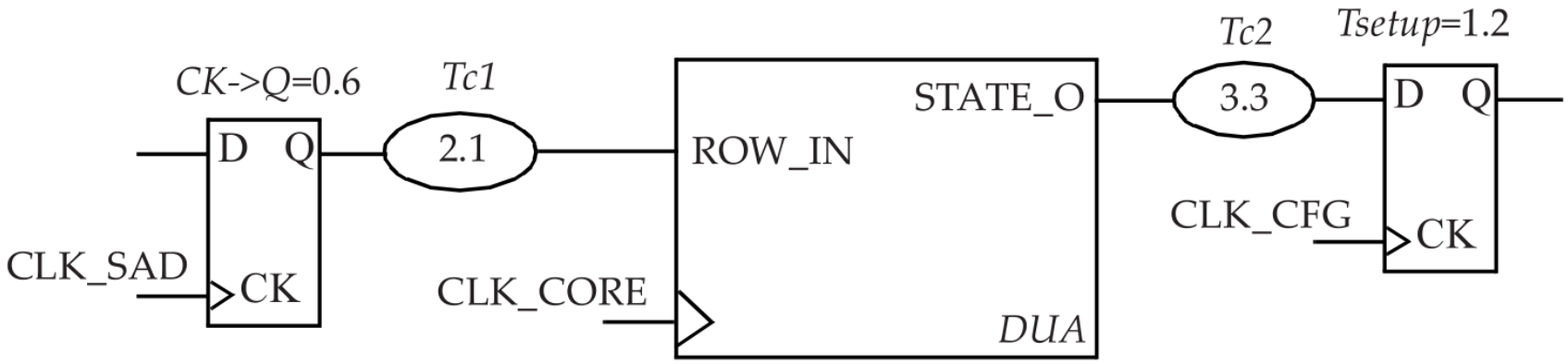


Figure 7-33 *Virtual clocks for CLK_SAD and CLK_CFG.*

Virtual Clocks

```
create_clock -name VIRTUAL_CLK_SAD -period 10 -waveform {2 8}
create_clock -name VIRTUAL_CLK_CFG -period 8 -waveform {0 4}
create_clock -period 10 [get_ports CLK_CORE]
set_input_delay -clock VIRTUAL_CLK_SAD -max 2.7 [get_ports ROW_IN]
set_output_delay -clock VIRTUAL_CLK_CFG -max 4.5 [get_ports STATE_O]
```

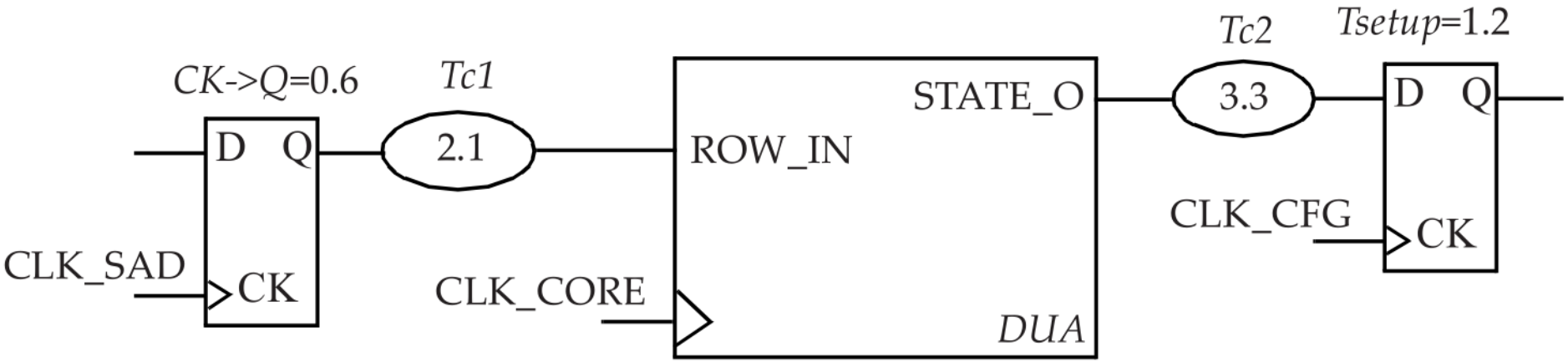


Figure 7-33 *Virtual clocks for CLK_SAD and CLK_CFG.*

Part 4: Configuring the STA Environment

CONTENT

1. What is the STA Environment?
2. Specifying Clocks
3. Generated Clocks
4. Constraining Input Paths and Output Path
5. Design Rule Checks
6. Virtual Clocks
7. Refining the Timing Analysis

Refining the Timing Analysis

Four common commands that are used to constrain the analysis space are:

i. set_case_analysis: Specifies constant value on a pin of a cell, or on an input port.

ii. set_disable_timing: Breaks a timing arc of a cell.

iii. set_false_path: Specifies paths that are not real which implies that these paths are not checked in STA.

iv. set_multicycle_path: Specifies paths that can take longer than one clock cycle

Refining the Timing Analysis

In a design, certain signals have a constant value in a specific mode of the chip.

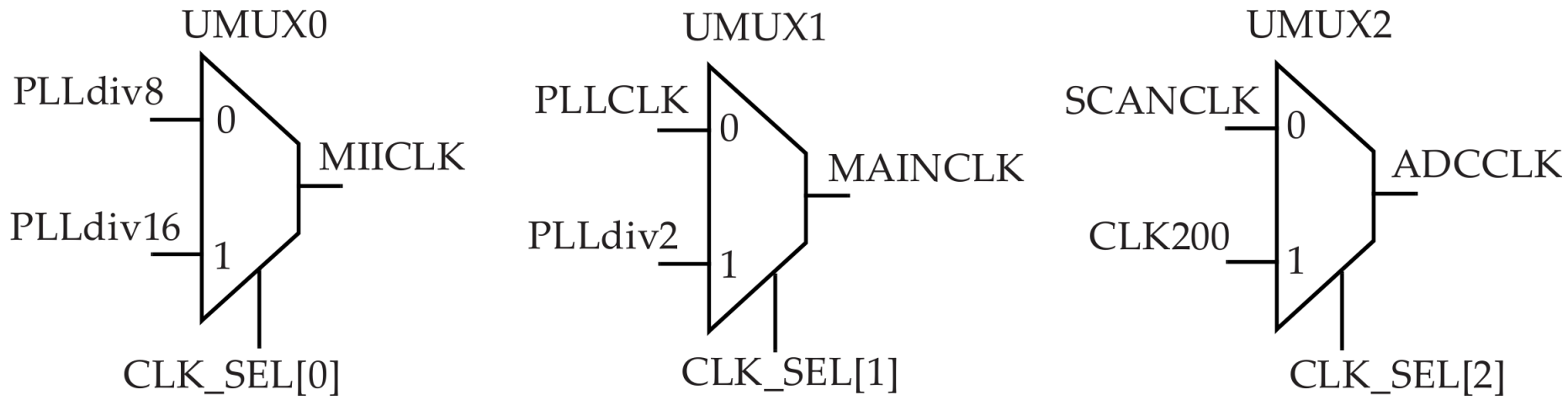
For example, if a chip has DFT logic in it, then the TEST pin of the chip should be at 0 in normal functional mode. It is often useful to specify such constant values to STA

```
set_case_analysis 0 TEST

set_case_analysis 0 [get_ports {testmode[3]}]
set_case_analysis 0 [get_ports {testmode[2]}]
set_case_analysis 0 [get_ports {testmode[1]}]
set_case_analysis 0 [get_ports {testmode[0]}]
```

Refining the Timing Analysis

Another common application of case analysis is when the design can run on multiple clocks, and the selection of the appropriate clock is controlled by multiplexers. To make STA analysis easier and reduce CPU run time, it is beneficial to do STA for each clock selection separately.

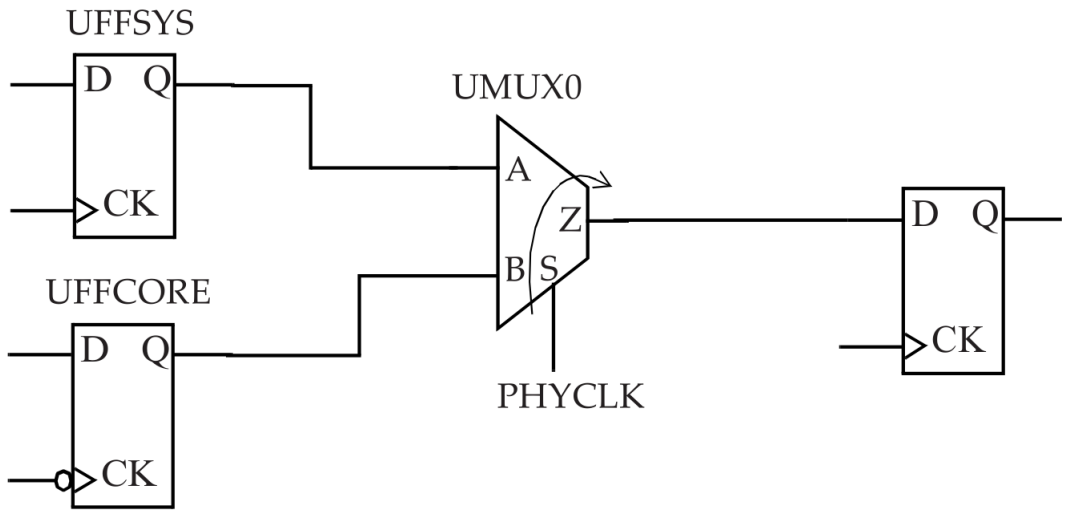



```
set_case_analysis 1 UCORE/UMUX0/CLK_SEL[0]
set_case_analysis 1 UCORE/UMUX1/CLK_SEL[1]
set_case_analysis 0 UCORE/UMUX2/CLK_SEL[2]
```

Refining the Timing Analysis

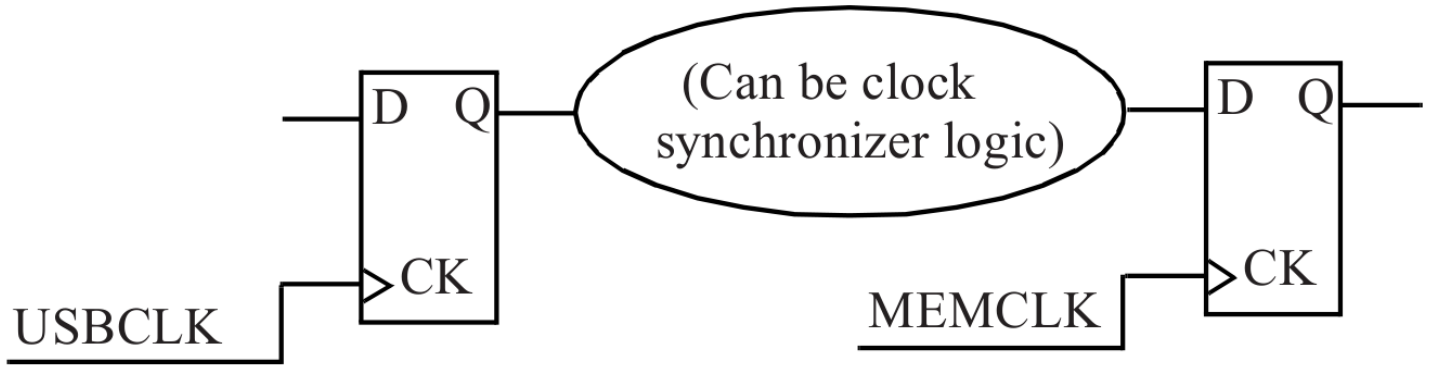
In some situations, it is possible that a certain path through a cell cannot occur.

Such a timing arc can be broken by using the set_disable_timing SDC command.



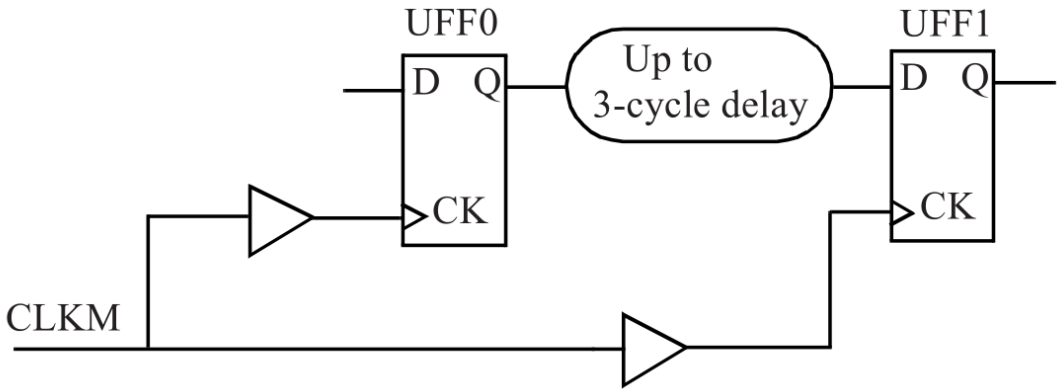
set_disable_timing -from S -to Z [get_cells UMUX0]

Refining the Timing Analysis



set_false_path -from [get_clocks USBCLK] -to [get_clocks MEMCLK]

Refining the Timing Analysis



create_clock -name CLKM -period 10 [get_ports CLKM]

set_multicycle_path 3 -setup -from [get_pins UFF0/Q] -to [get_pins UFF1/D]

参考书目

- Static Timing Analysis for Nanometer Designs: A Practical Approach. J. Bhasker, Rakesh Chadha. Springer Science Business Media, LLC 2009. Chaper 7.
- 集成电路静态时序分析与建模. 刘峰，机械工业出版社.出版时间：2016-07-01. 第六章.

谢谢聆听！

个人教学工作主页https://customizablecomputinglab.github.io/