보안 회로 설계

ASIC flow

Dong Kyue Kim
Hanyang University
dqkim@hanyang.ac.kr

ASIC와 FPGA 정의

- ASIC(Application Specific Integrated Circuit)
  - 주문형 반도체
  - 특정한 목적에 의해 가공된 칩
    - 일반적인 표준 셀 방식

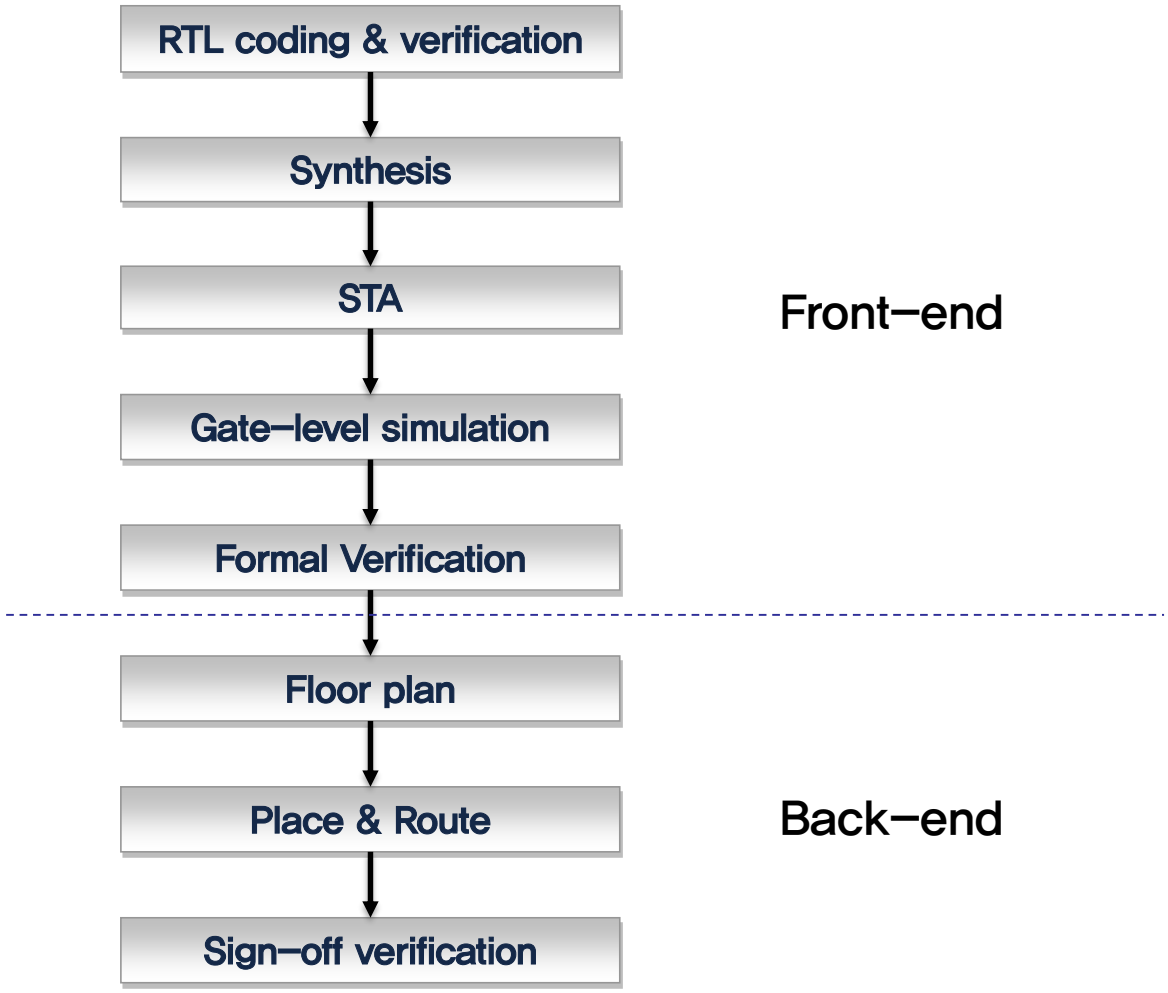
- FPGA(Field Programmable Gate Array)
  - 일반 사용자가 원하는 로직을 프로그램 할 수 있는 gate array
    - 현장에서 재수정 가능

ASIC와 FPGA 장점

- ASIC 장점
  - 큰 규모에서 가격이 효율적이다.
  - 빠른 속도
  - 높은 집적도
  - 낮은 전력소모
  - 특수한 필요성
    - 아날로그와 디지털이 복합된 회로(Full Custom)

- FPGA 장점
  - 작은 규모에서 가격이 효율적
  - 오류발생시 회로 변경이 가능함(수정 가능)
  - 개발 시간이 짧으며 개발 비용이 비교적 적음
    - Time to market : 제품 개발 및 제품을 만들어서 시장에 내놓는 데 걸리는 시간이 짧다.

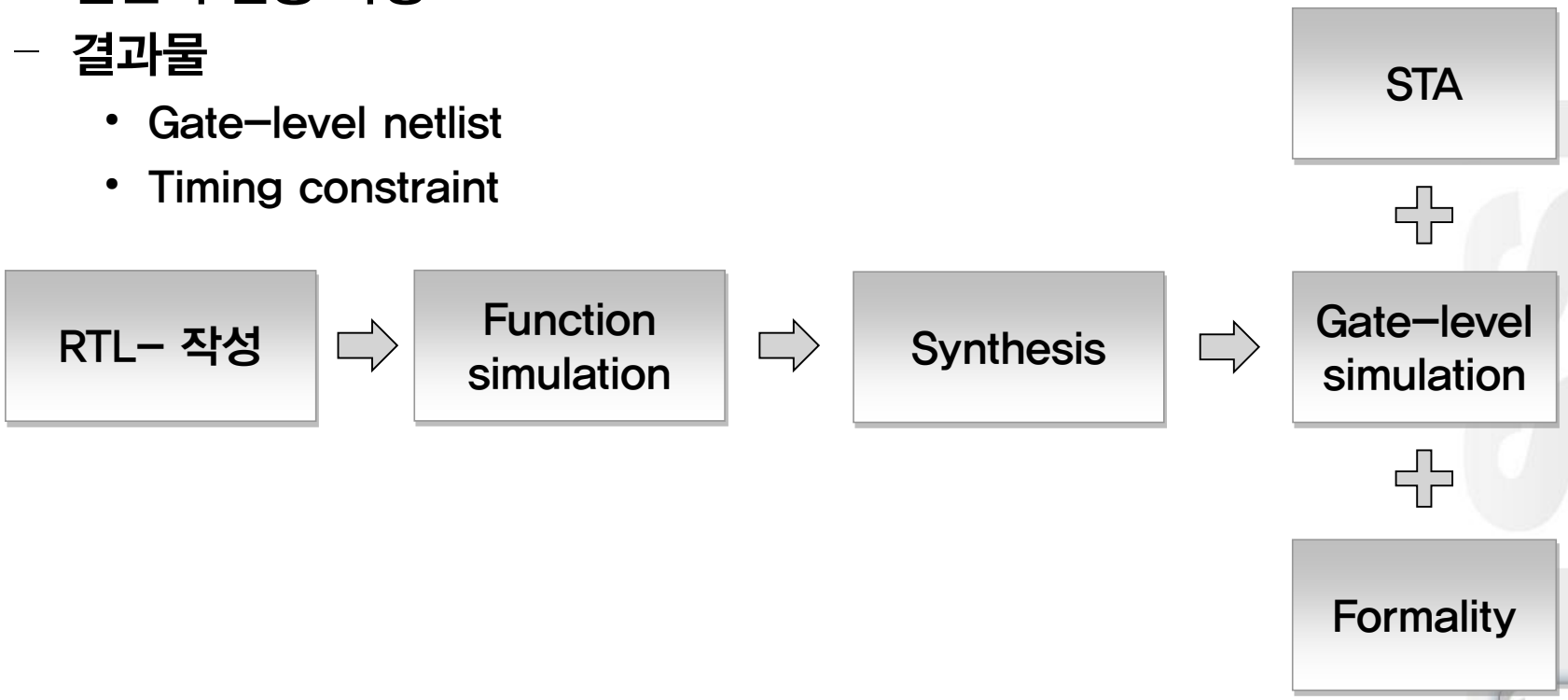
ASIC flow


RTL coding & verification
Synthesis
STA
Gate-level simulation
Formal Verification
Floor plan
Place & Route
Sign-off verification
Front-end
Back-end

Front-end Flow

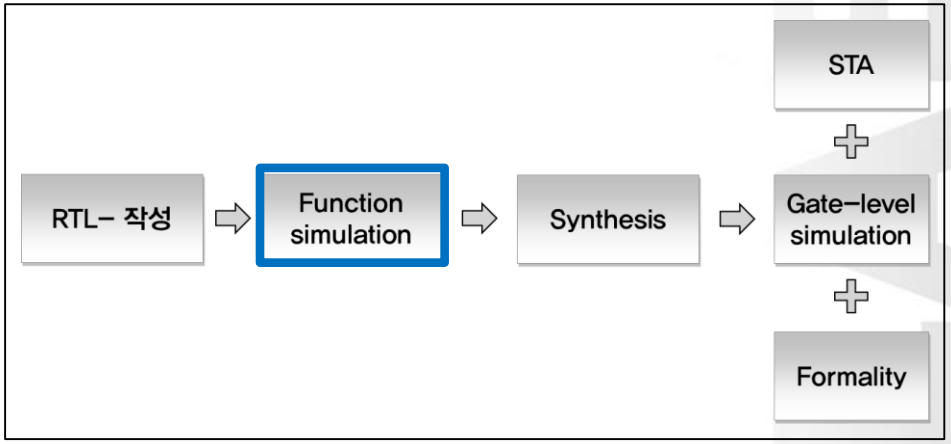
Front-end

- Front-end 란?
  - RTL 작성
  - 게이트 합성
  - 일련의 검증 과정
  - 결과물
    - Gate-level netlist
    - Timing constraint



Front-end Flow : Function Simulation

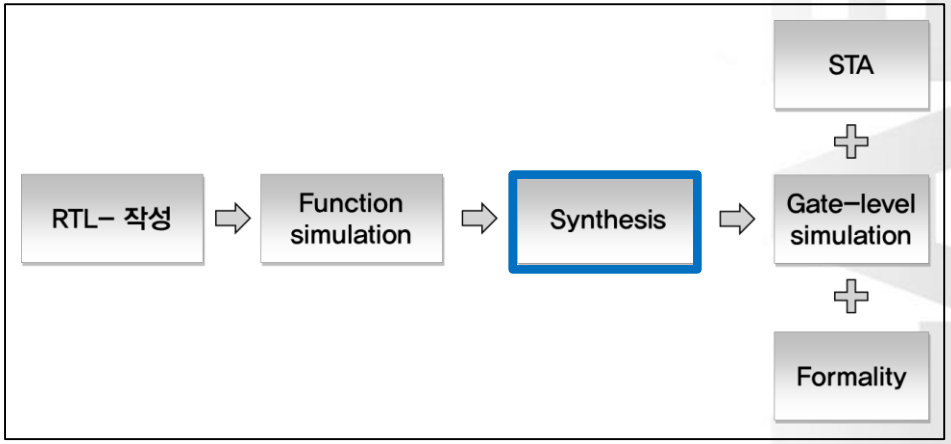
- 목적
  - 모듈의 논리적 동작 확인
- 사용되는 툴
  - Modelsim



- Input
  - 모듈.v
  - 테스트 벤치.v
- Output
  - 시뮬레이션 웨이브 결과

Front-end Flow : synthesis(1/2)

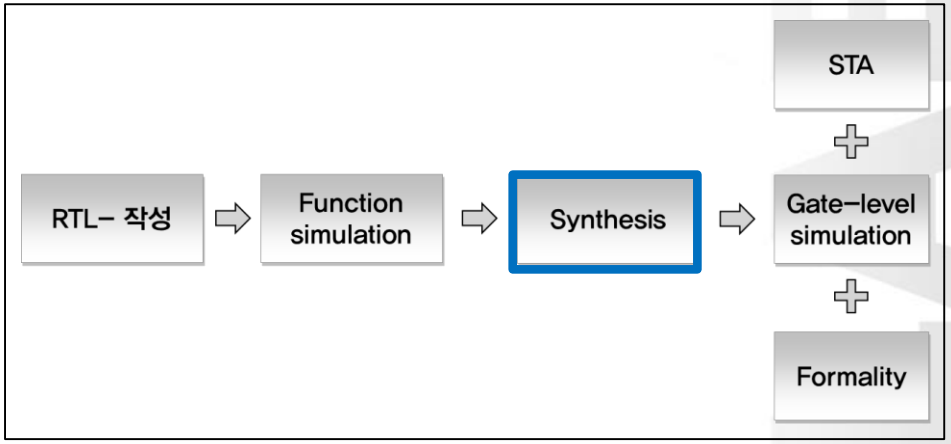
- 목적
  - 합성
    - RTL-level netlist
      => Gate-level netlist
- 사용되는 툴
  - Design Compiler



- Input
  - 모듈.v
  - Library : 셀의 크기와 딜레이
  - WLM(Wire Load Model) : wire에 대한 통계적 net delay정보
  - Constraints : 사용자 요구 설정
    - I/O delay, Clock uncertainty, Max fanout, capacitance, transition 등

Front-end Flow : synthesis(2/2)

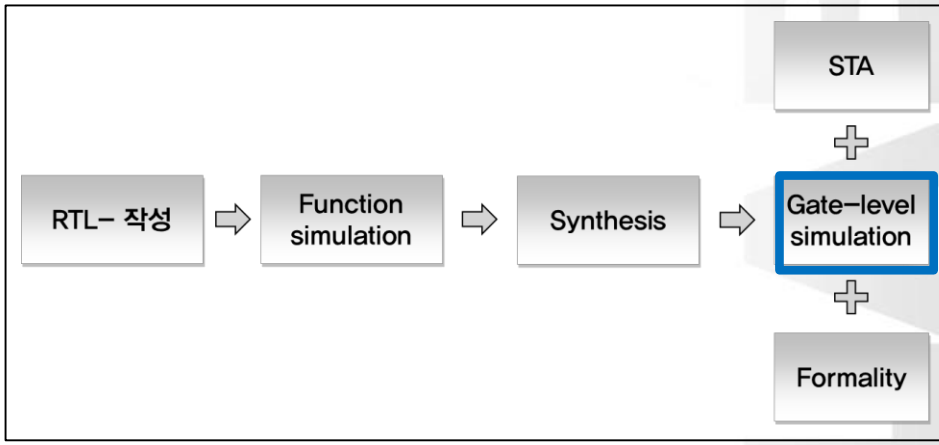
- Output
  - gate-level netlist
    - 모듈.syn.v
    - .sdf : net delay정보
    - .sdc : constraints
      Back-end에서 사용됨
    - .svf : setup verification formality



  - 합성 결과 report
    - Slack : 예상도착 시간과 신호의 실제 도착 시간의 차이
    - Area : gate-level design 면적
      (gate Count/ Gate Equivalent)

Front-end Flow : Gate-level simulation

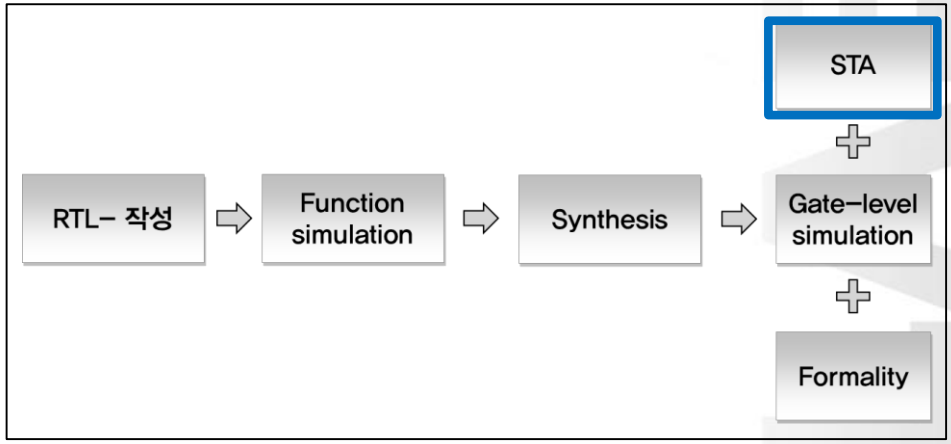
- 목적
  - 모듈 gate들의 논리적 동작 확인
  - cell delay 의한 순차적 동작 확인
- 사용되는 툴
  - Modelsim



- Input
  - 모듈.syn.v
  - sdf : net delay정보 파일
  - 공정 cell .v 파일
  - Testbench.v
- Output
  - 시뮬레이션 웨이브 결과

Front-end Flow : STA(Static Timing Analysis)

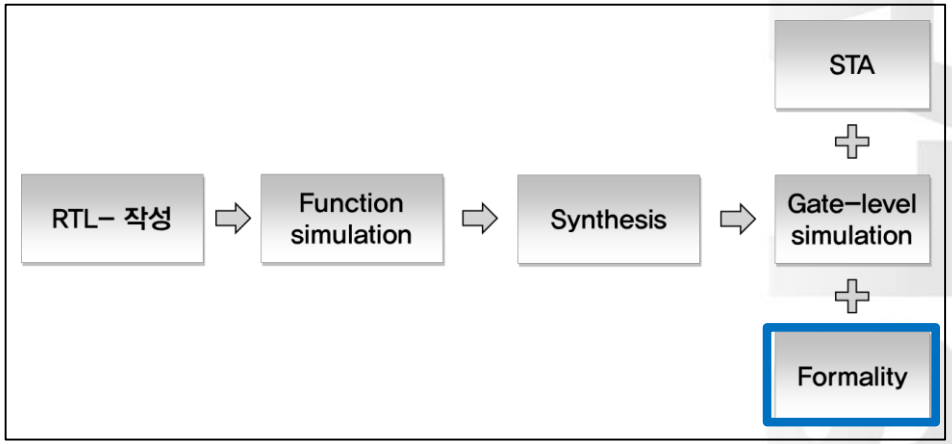
- 목적
  - 합성 결과물의 timing 만족 검증
    - 모든 경로의 delay를 확인
    - Slow path, 글리치 등 문제 감지
- 사용되는 툴
  - Primetime



- Input
  - 모듈.syn.v
  - .sdc : constraints
  - Library : 셀의 크기와 딜레이
- Output
  - .sdf : net delay정보 파일
  - Timing 분석 report

Front-end Flow : Formal Verification

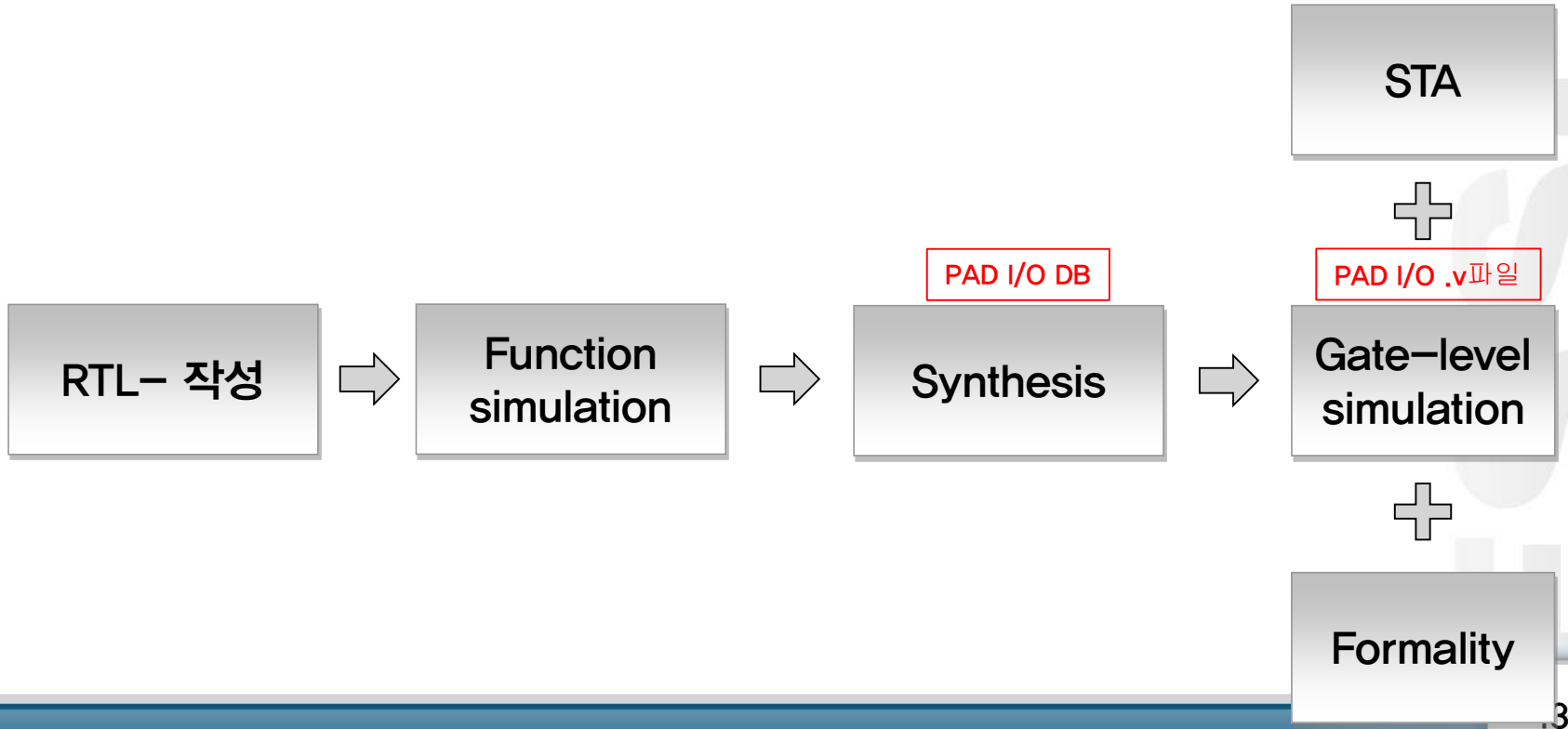
- 목적
  - RTL과 gate의 논리적 동등성 확인
- 사용되는 툴
  - Formality



- Input
  - 모듈.v
  - 모듈.syn.v
  - .svf : setup verification formality
  - LIB
- Output
  - 동등성 확인 레포트

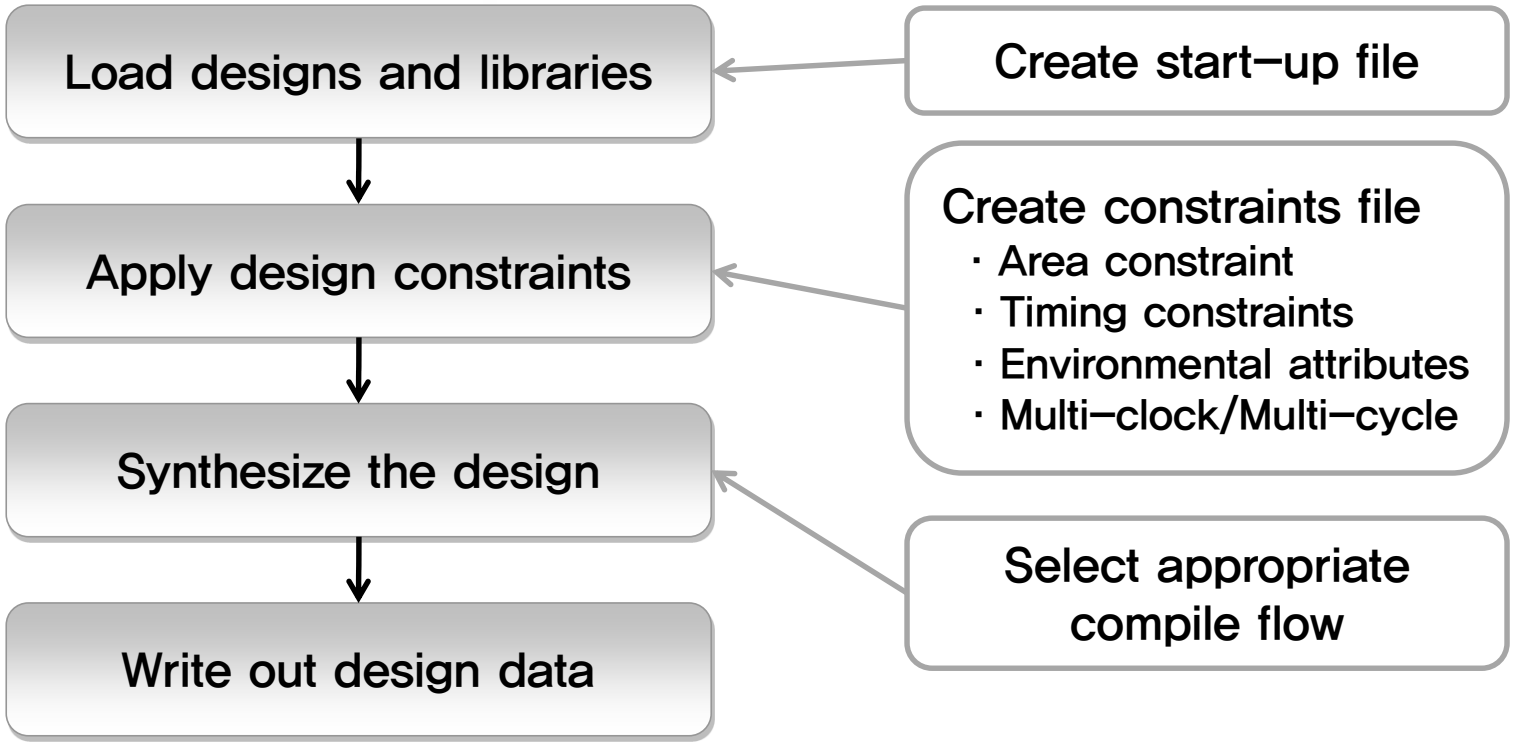
Front-end Flow : Pad 추가 (Optional)

- Pad
  - 외부 H/W에서 들어오는 전압을 cell에 맞게 맞춰주는 역할
  - 목적에 맞춰 Pad선정 : I/O pad, clk pad, power/ground pad…
  - RTL level에서 포함 시 synthesis 단계에서 같이 optimization 적용



Synthesis

Design Compiler Flow



Constraint 소개

- Constraint 정의
  - 설계하고 싶은 목표에 따라 제한을 두는 것

- Design rule constraints vs Optimization constraints
  - Design rule constraints : transition time, fanout load, capacitance 와 같이 chip의 원활한 동작을 위해 foundry에서 제공하는 minimum requirement.
  - Optimization constraints : timing, area와 같이 user가 optimized chip을 만들기위해 지정해야 하는 constraints. Clock의 정보, port의 timing, combinational path, maximum area 등을 지정.

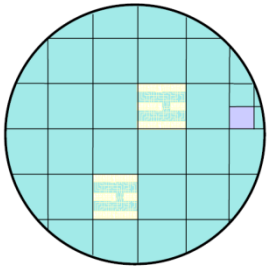
Constraint 소개

- Constraint 종류
  - Area constraint
  - Timing constraints
  - Environmental attributes
    - PVT, transition time, fanout, capacitance

Area Constraint

- Area constraint

  set_max_area 넓이



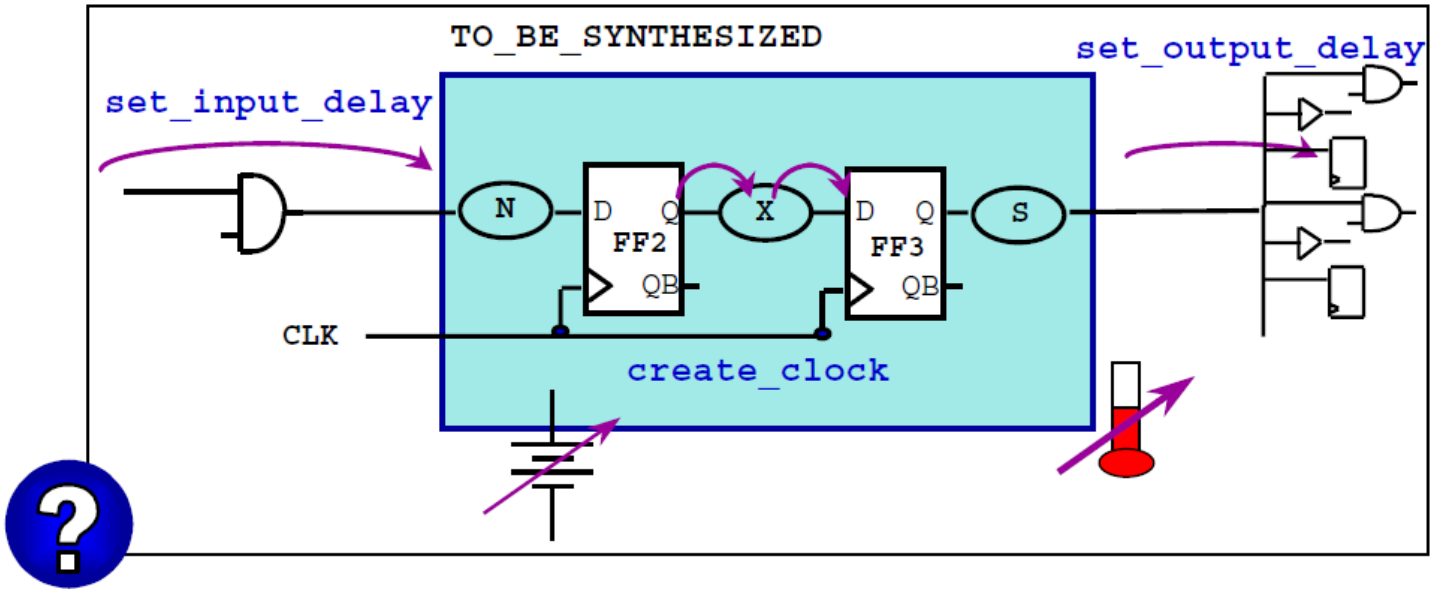
  - 면적의 단위는 target_library 의해 정의된다.
  - 기본 단위는 μ^2
  - set_max_area 0
    - 가능한 최소의 size로 area optimization 수행하라는 의미의 명령어.

Constraint 설정

- Design 내의 Timing Constraint
  - Design Compiler는 synchronously-clocked 환경이라 가정(default)
  - Sequential design 내의 모든 path에 대해 setup timing constraint 지정
    - 모든 input ports
    - Internal (register to register) paths
    - 모든 output ports

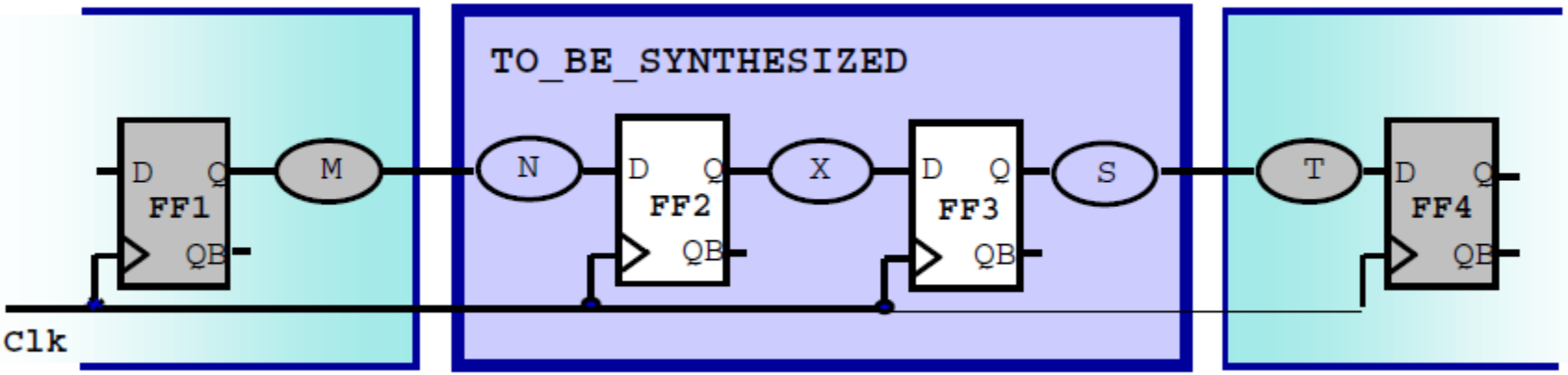
Constraint 설정

- Timing에 영향을 주는 요소들
  - Input_delay, output_delay, register to register delay



  - 위의 constraint들은 synthesis시 꼭 필요
  - 추가적으로 더 고려해야 할 사항
    - Input drivers/transition times, output loading, PVT, parasitic RC

Constraint 설정

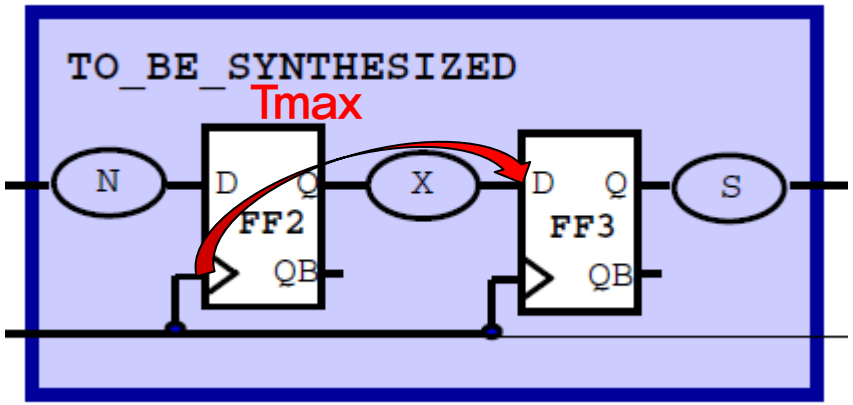


- Data arrives from a clocked device
- Data goes to a clocked device

- DC에서의 Timing path
  - Startpoints : all input ports, clock pins of FF or registers
  - Endpoints : all output port, all input pins of sequential devices(except clock pins)

Constraint 설정

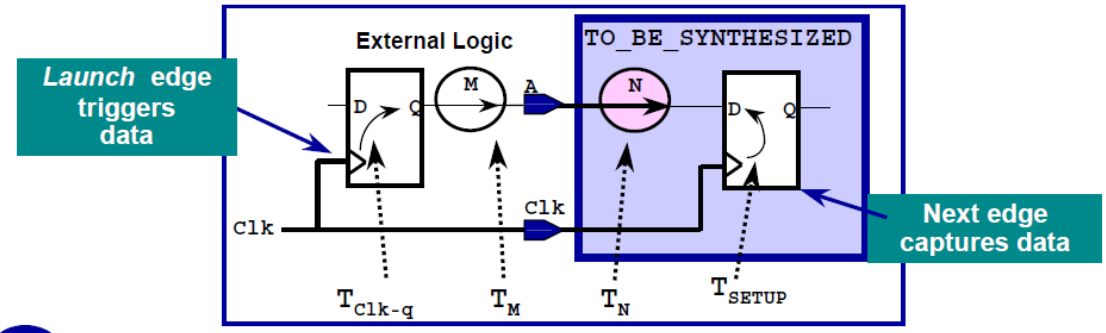
- Register to Register path



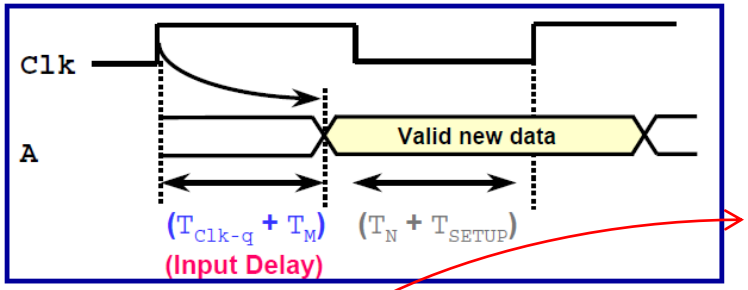
  - DC는 FF2의 clock pin부터 FF3의 D pin까지의 maximum delay 계산
  - Tmax = Clock Period – FF3의 Setup_time
  - FF3의 setup_time은 technology library로부터 가져온다.

Constraint 설정

- Input delay



What information must you provide to constrain the input paths?



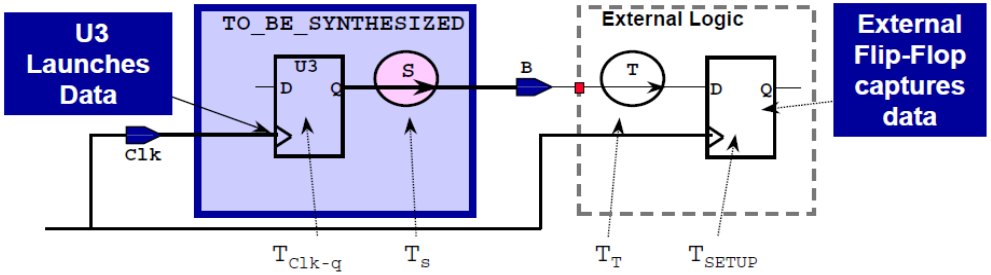
Time Budgeting : 입력단의 delay를 알 수 없을 경우 clock 주기의 40% 할당

- 입력단이 주기의 40%를 가지고 있다고 생각. 즉 주기의 60% 안으로 input logic이 합성되어야 함.

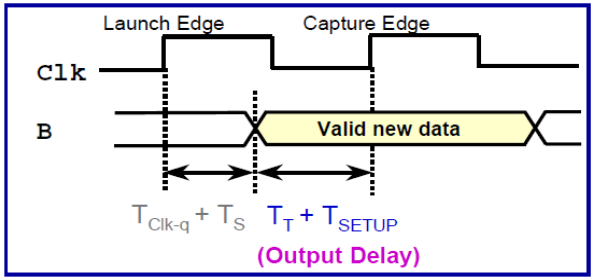
set_input_delay –max 시간 –clock 클럭명 [get_ports 포트명] [all_inputs]

Constraint 설정

- Output delay



What information must you provide to constrain the output paths?

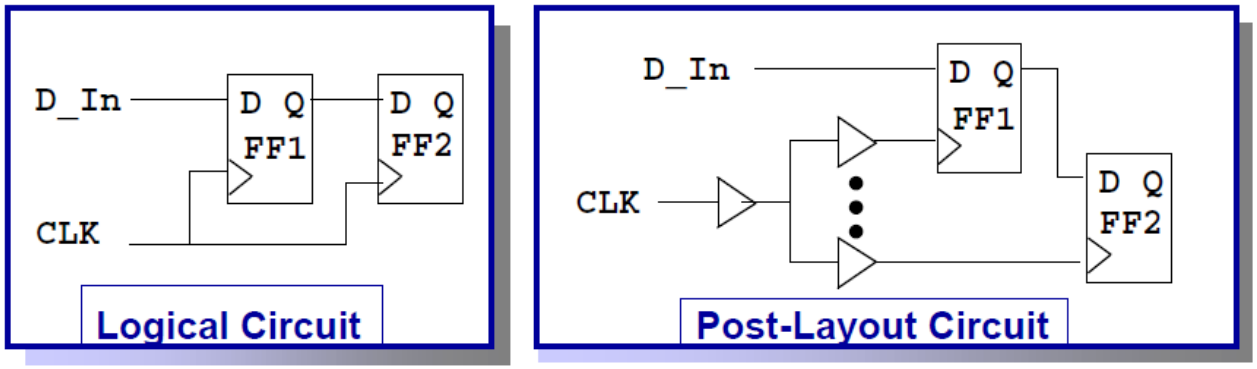


- 출력단이 주기의 40%를 차지하고 있다고 생각. 즉 주기의 60% 안으로 output logic이 합성되어야 함.

set_output_delay –max 시간 –clock 클럭명 [get_ports 포트명] [all_outputs]

Constraint 설정

- Clock uncertainty
  - clock skew + clock jitter + clock margin



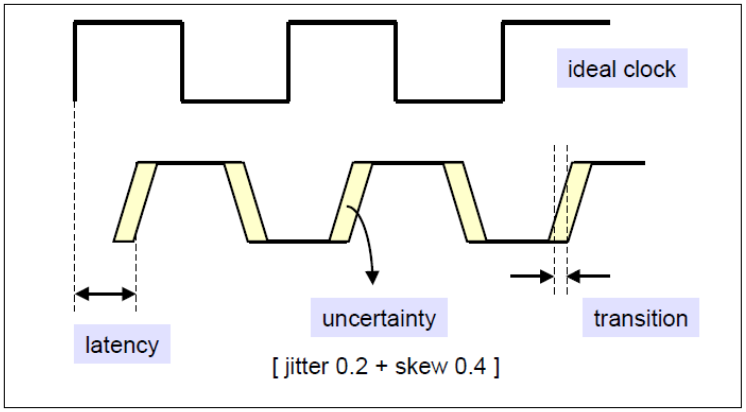
set_clock_uncertainty –setup 시간 [get_ports 클럭이름]

set_clock_uncertainty –hold 시간 [get_ports 클럭이름]

Constraint 설정

- Clock skew
  - 실제 배치에서 wire delay때문에 발생하는 오차
- Clock jitter
  - 시간에 정기적으로 변화하는 신호 특성의 최대와 최소 사이의 간격
- Clock transition
  - 0~1(rise time) 혹은 1~0(fall time)으로 변화할 때 걸리는 시간

set_clock_transition 시간 [get_ports clk]



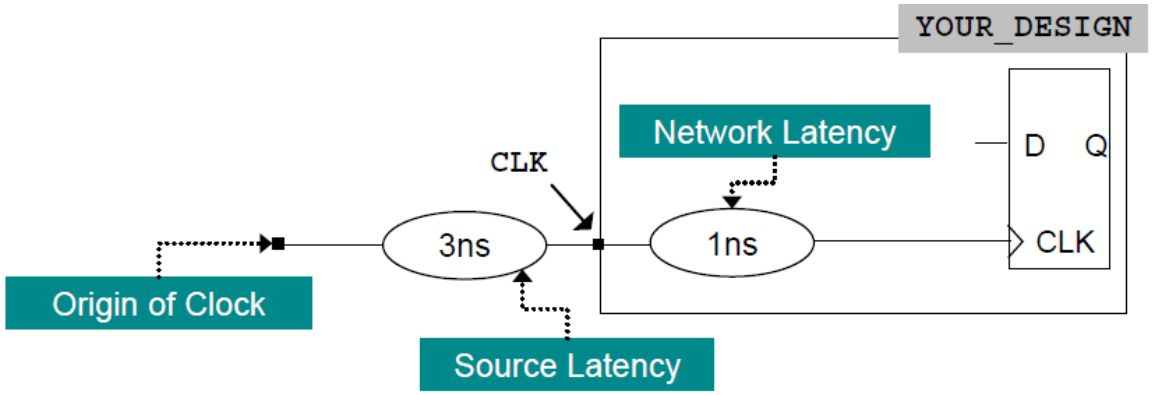
Constraint 설정

- Clock latency
  - Network latency : clock port(pin) –〉 register clock pin 평균 delay

```
set_clock_latency 시간 [get_ports clk]
```

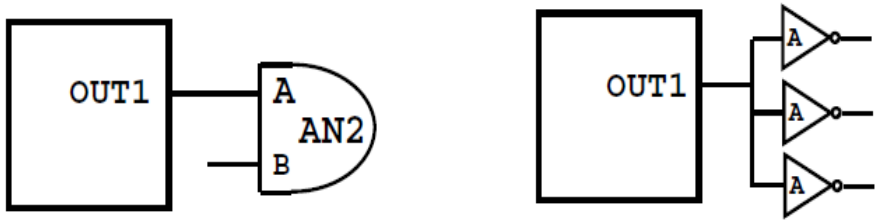
  - Source latency : 외부 source clock –〉 clock port(pin) delay

```
set_clock_latency –source 시간 [get_ports clk]
```



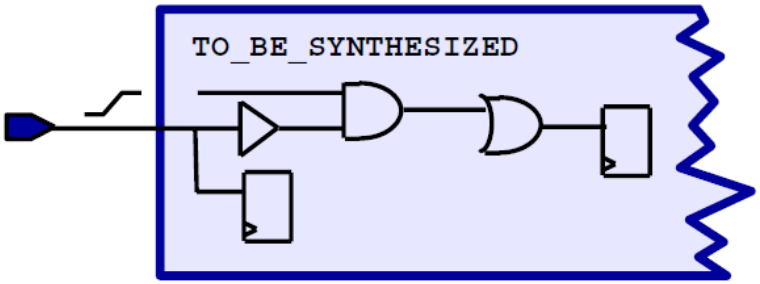
Environmental attributions

- Output Capacitive Load



set_load 숫자 [get_ports 포트명]

- Input transition

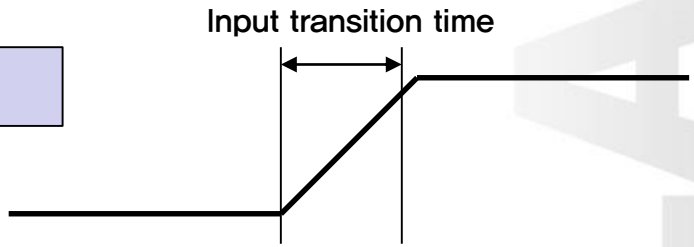


set_input_transition 시간 [get_ports 포트명]

Environmental attributions

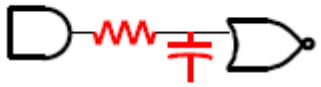
- **Max Transition**
  - Design 내 net의 transition 허용 최대치

set_max_transition 값 [포트 or 디자인 or clock]



- **Max Fanout**
  - 회로에서 한 게이트의 출력을 다른 곳으로 배분하여 연결한 출력선의 수 (디지털에서의 의미)

set_max_fanout 값 [포트 or 디자인 or clock]



- **Max Capacitance**
  - Design내 capacitance 최대 크기

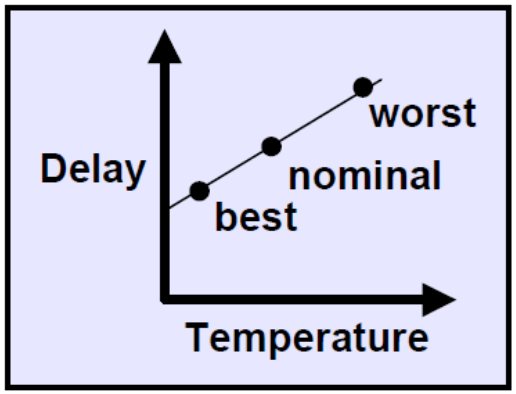
set_max_capacitance 값 [포트 or 디자인 or clock]

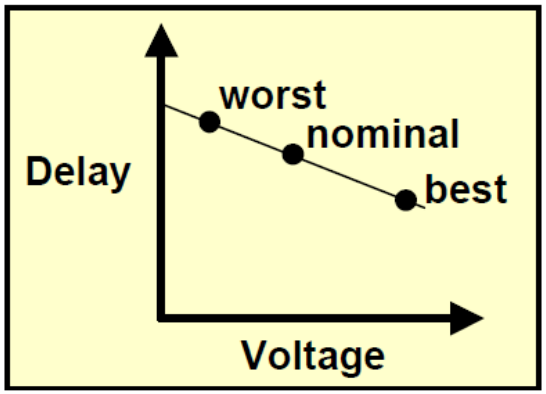
Environmental attributions

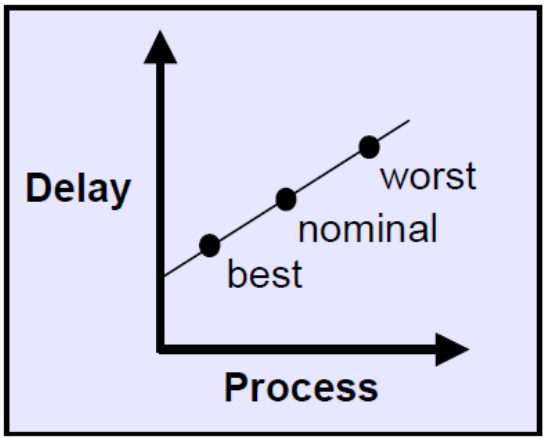
- PVT(Process, Voltage, Temperature)
  - 공정이 낮을수록, 전압이 높을수록, 온도가 낮을수록 빨라진다.
  - 1개의 라이브러리 내에 여러 개의 condition

```
set_operating_condition –옵션 옵션이름
```

  - 여러 개의 라이브러리(동부 0.11nm : Metro/Sagex, ff/tt/ss)
  - 항상 최악을 가정하고 Synthesis를 하는 것이 좋음 –〉 ss 사용







Environmental attributions

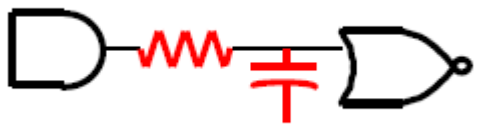
- Wire Load Model를 가진 Net RC 구현
  - net의 fanout을 기준으로 각 net의 parasitic R과 C 계산
  - 다양한 디자인 크기에 맞는 Model들이 벤더들로부터 공급받음
  - 가장 작은 모델의 wire cap 사용

```
set_wire_load_moded enclosed
set auto_wire_load_selection true
```

  - compile시 수동으로 설정 후 모델 설정

```
set_wire_load_model -name 모델명
set auto_wire_load_selection false
```

→ Design_area 보고 판단



  - WLM은 통계적인 평균을 기준으로 함.

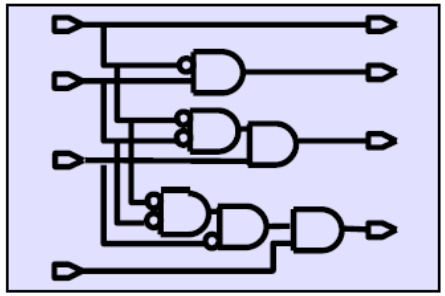
Synthesis

Synthesis = Translation + Optimization + Mapping

```
residue = 16'h0000;
if (high_bits == 2'b10)
   residue = state_table[index];
else
   state_table[index] = 16'h0000;
```

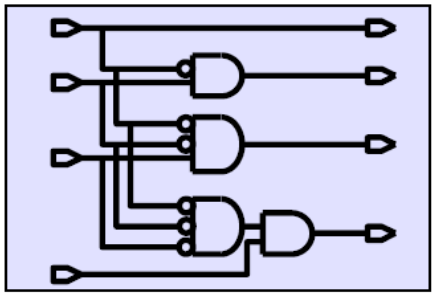
HDL Source

Translate



Generic Boolean
(GTECH)

Optimize + Map



Target Technology

Compile

- 세 단계의 Optimization
  - Architectural level(High-Level Synthesis)
    - read : RTL Description or unmapped ddc
    - Resource Sharing : 어느 resource를 공유하는지에 따라 구조 변화
    - Implementation Selection : timing을 만족시키면서 작은 것 선정
      - ex) adder의 경우 여러 종류가 있으며 선택이 필요함
    - Operator Reordering : late-arriving signals을 뒤로 배치
  - Logic level(Structuring)
    - read : netlist or mapped ddc
    - common sub-expression을 사용하여 logic 줄임
  - Gate level(Mapping)
    - Combinational mapping & optimization : timing과 area goal에 따라 target library로부터 최상의 combinational logic gate 선택
    - Sequential mapping & optimization : 복잡한 sequential cell을 사용하여 speed와 area를 줄임

Compile

- mapping Optimization 우선 순위
  - DRCs(Design rule constraints) : 각 cell에 대한 vendor-specific design rule
    - ex) max_capacitance/transition
    - DRC fixing : 버퍼 inserting/removing, re-sizing
  - Timing constraints
    - Setup time/hold time
  - Area constraints

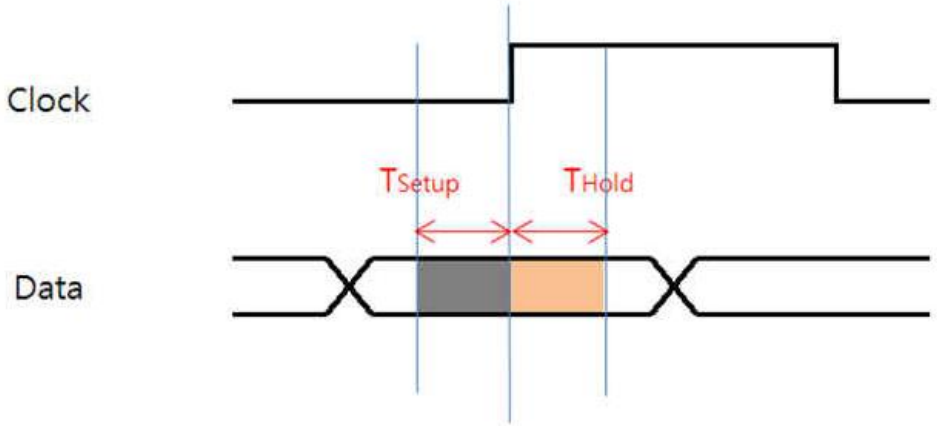
Synthesis 결과

- gate-level netlist
  - 모듈.syn.v
  - .sdf : net delay정보
  - .sdc : constraints
    Back-end에서 사용됨
  - .svf : setup verification formality

- 결과 report
  - Slack : 요구되는 도착 시간과 신호의 실제 도착 시간의 차이
    - Setup time/ hold time
  - Area : gate-level design 면적
    (gate Count/ Gate Equivalent)

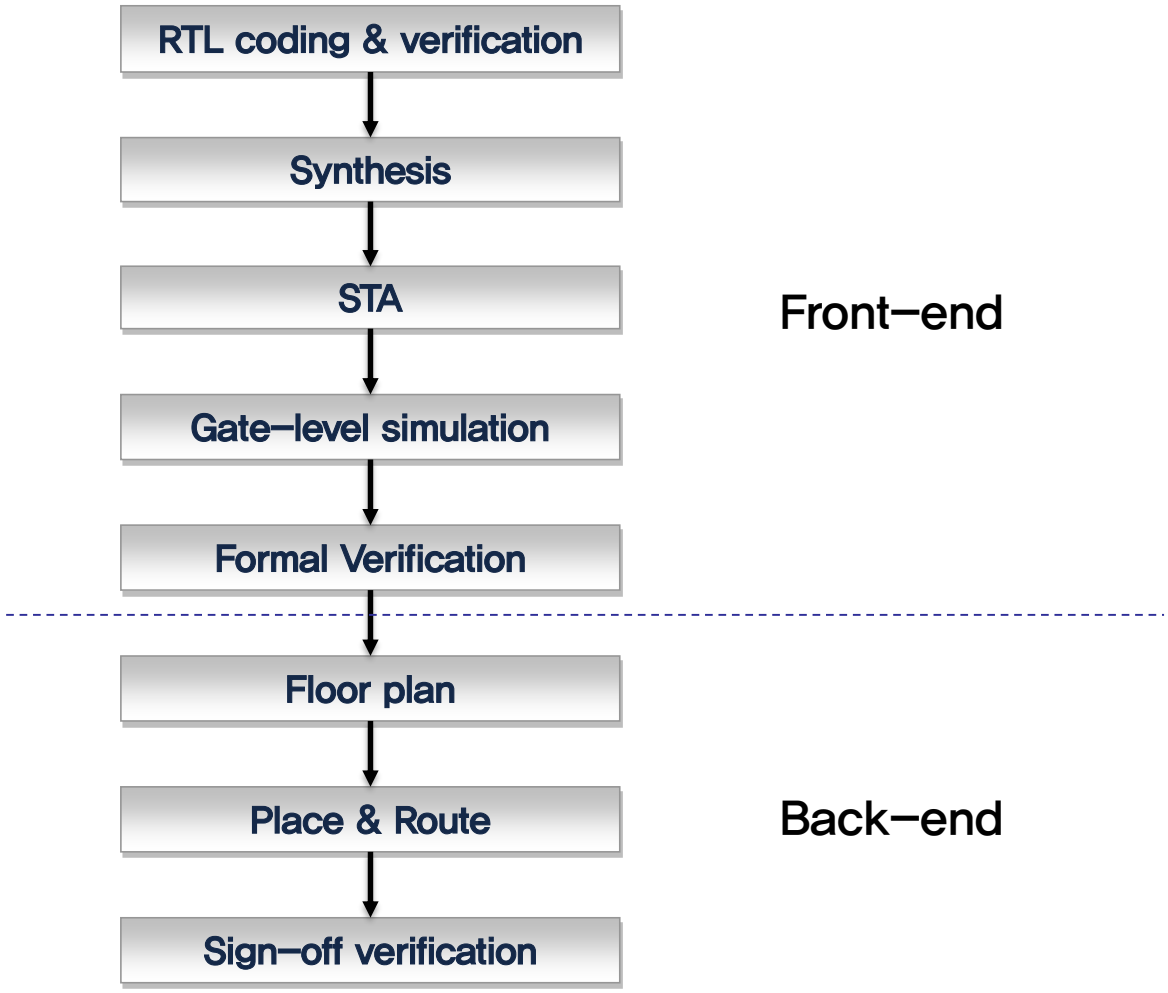
Synthesis 결과

- Timing 분석
  - Setup time :
    - Switching이 일어나기 전까지 입력이 인식되는데 필요한 최소 유지 시간
    - Data의 파형이 High인지 Low인지를 판별하는데 필요한 최소시간
  - Hold time :
    - Switching이 일어난 후 상태의 변화가 정확히 인식되도록 필요한 최소
    - 판별된 결과가 유지되어야 하는 최소시간



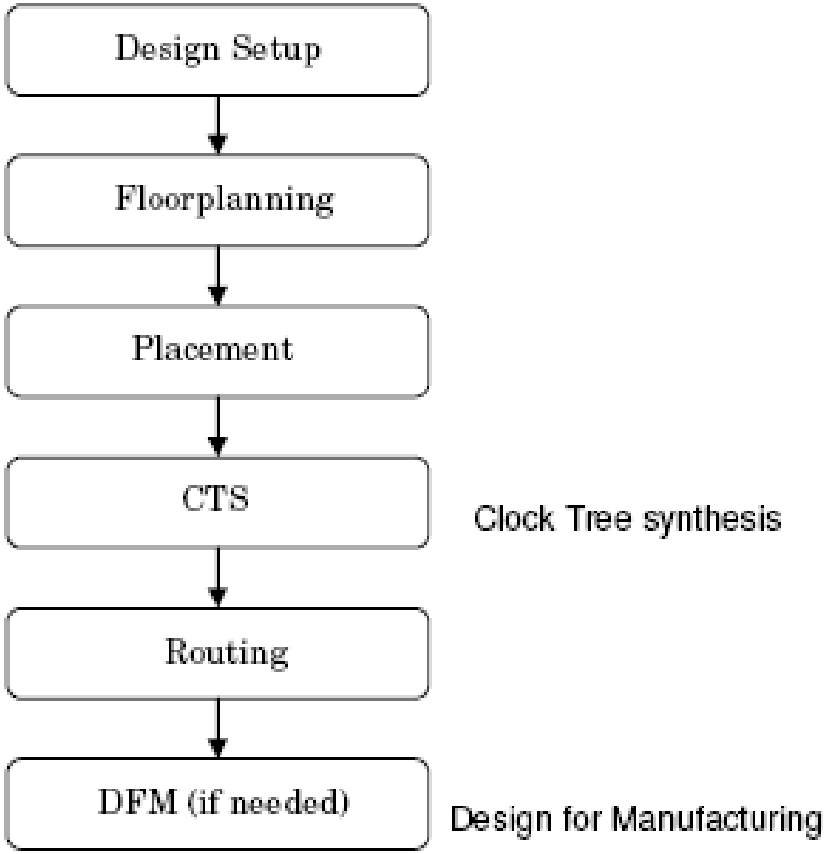
Back-end Flow

ASIC flow


RTL coding & verification
Synthesis
STA
Gate-level simulation
Formal Verification
Floor plan
Place & Route
Sign-off verification
Front-end
Back-end

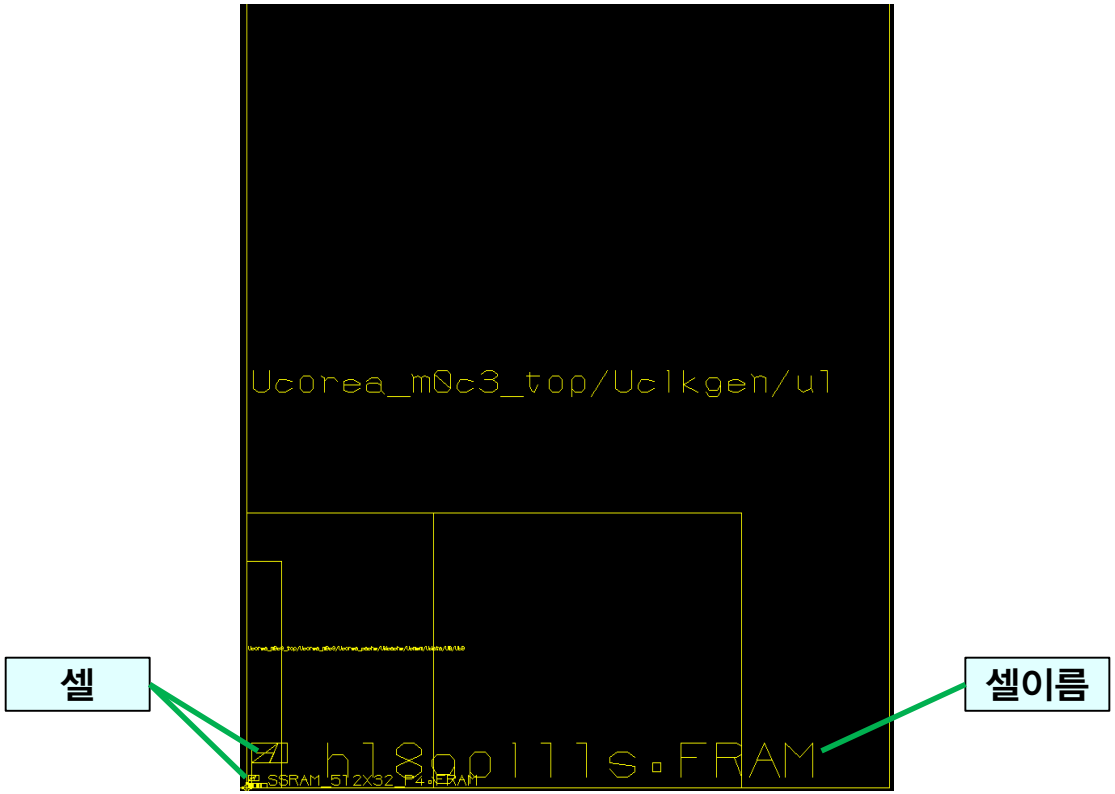
Backend Flow

- Astro tool을 이용한 Backend 전체 Flow



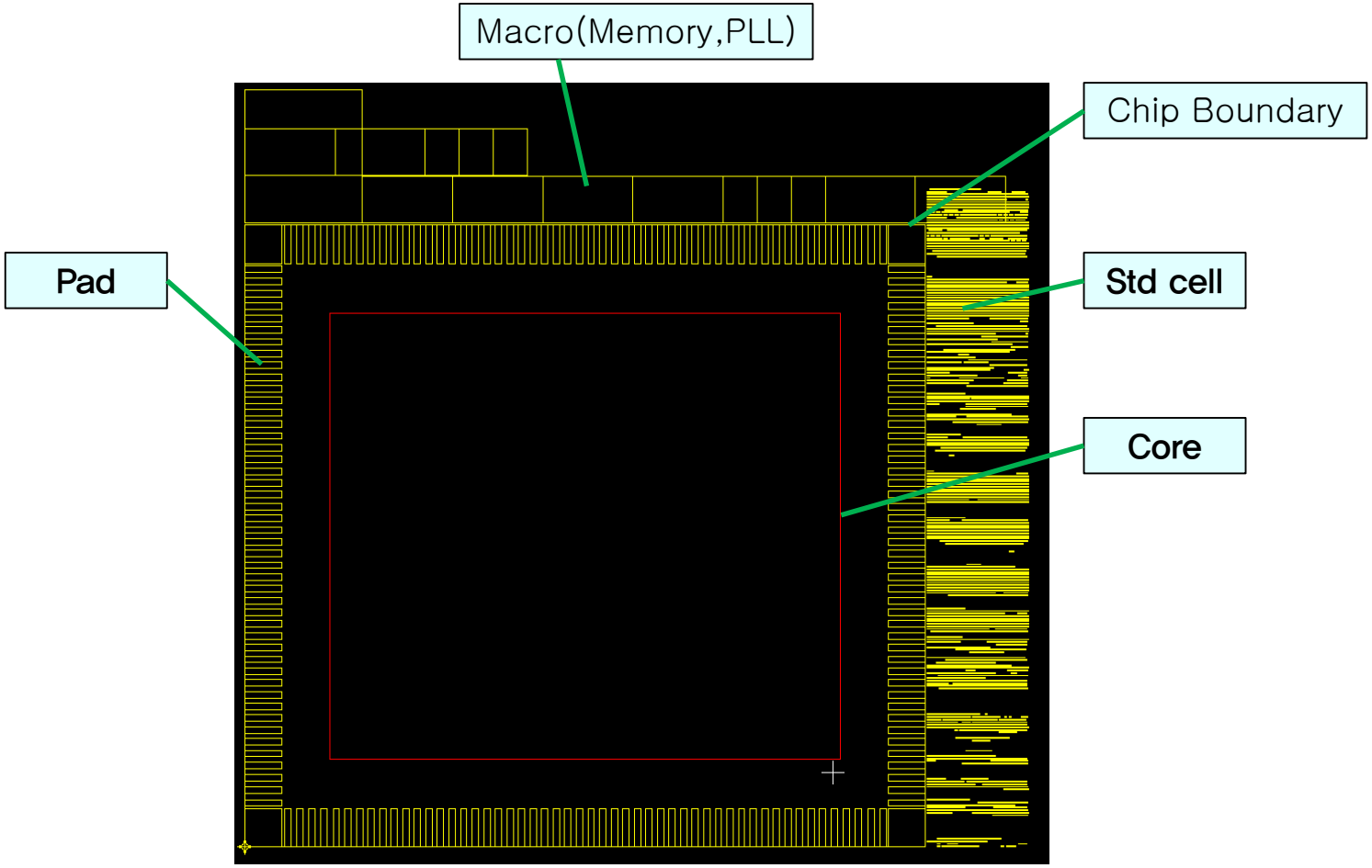
Backend Flow

- Design Setup



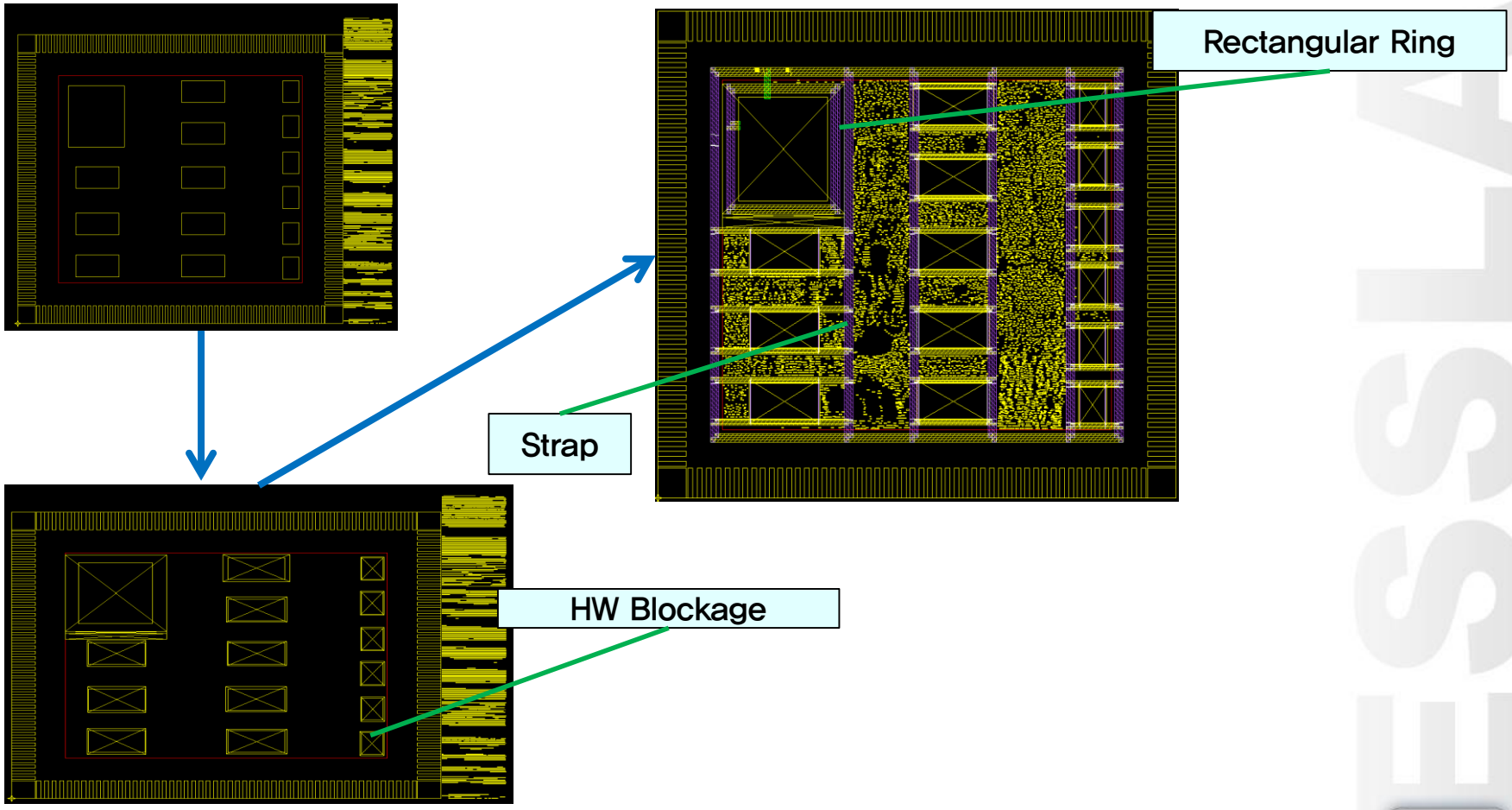
Backend Flow

- Floorplan



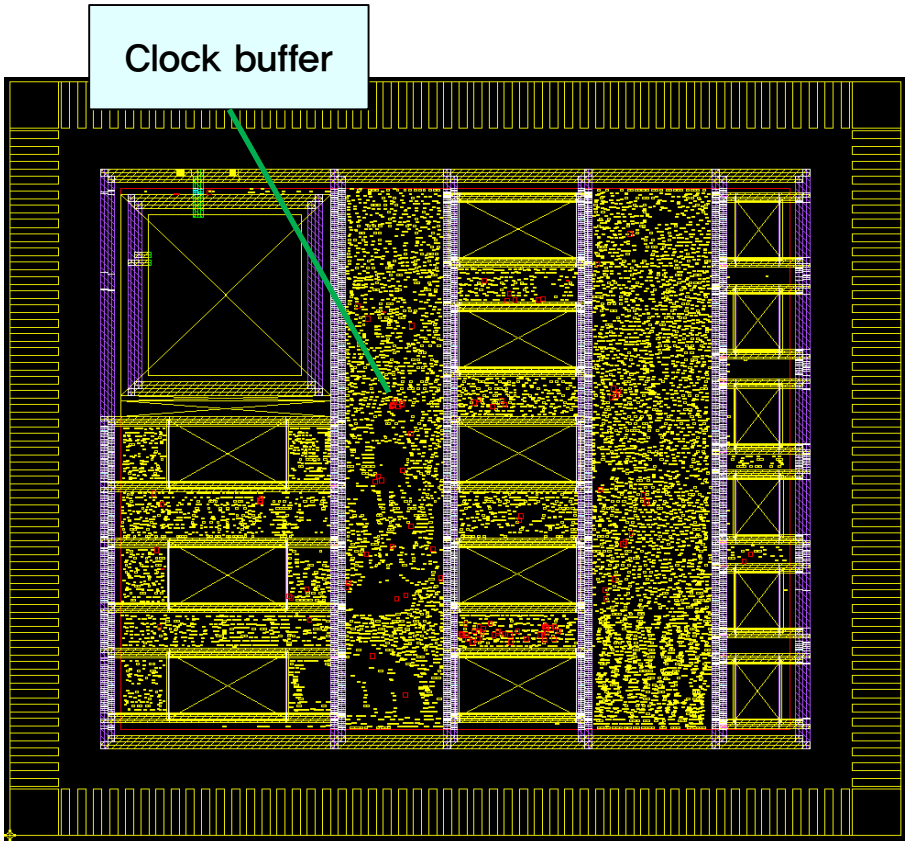
Backend Flow

- Place(macro, blockage, std cell)



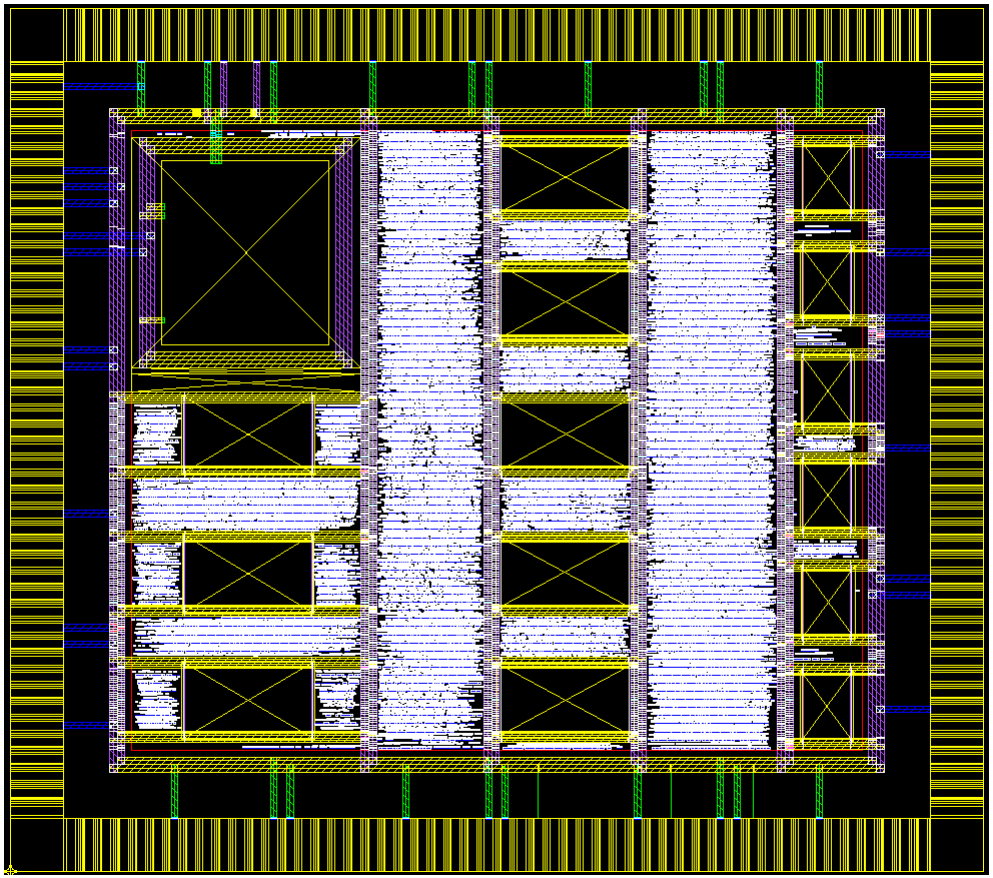
Backend Flow

- CTS(Clock Tree Synthesis)



Backend Flow

- Routing



Backend Flow

- DFM(Design For Manufacturing)

Antenna

Via

Notch and gap

기말 Project

기말 Project

- 암호 시스템 FPGA 구현
  - 6/20 까지
  - 블록암호 알고리즘 1개 구현
    - 64-bit text, 128-bit key 이상
    - 암호화/복호화
  - FPGA 검증
    - 수업 실습 때 구현한 시스템 이용
    - PC와 UART 통신을 통한 동작 검증
  - 레포트 작성
    - 암호 알고리즘에 대한 구체적 설명
    - 구현 내용 및 결과