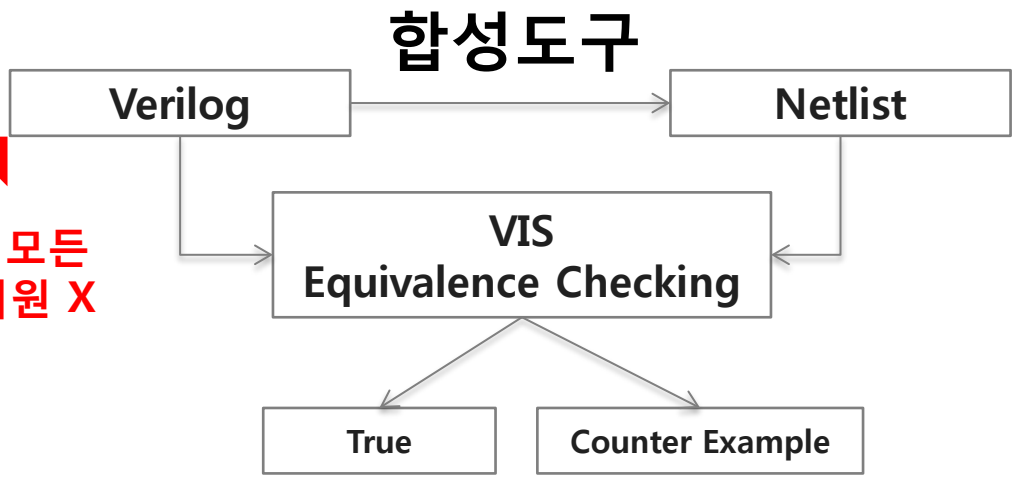
Verilog4VIS – EC:
VIS의 동치성 검사를 위한 Verilog의 정제된 포맷

건국대학교
김의섭, 정세진, 김재엽, 유준범, 장찬현

Verilog4VIS – EC:
VIS의 동치성 검사를 위한 Verilog의 정제된 포맷

합성도구를 VIS의 EC를 통해 검증



VIS가 Verilog의 모든
구문과 문법을 지원 X

VIS의 EC를 위한
제약사항 정리

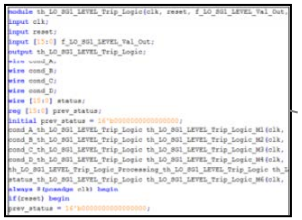
플랫폼 변경의 요구

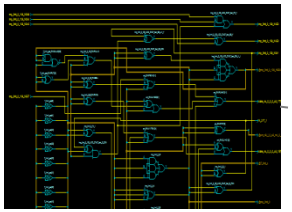
- 원자로보호시스템(RPS)의 플랫폼 변경 요구 수용 (**PLC → FPGA**)
  - 기존 PLC 기반 소프트웨어의 유지보수 비용과 새로운 RPS 개발의 복잡성 증가
    - 부품조달과 기술력확보의 어려움 예상
  - 원전 디지털 계측제어시스템에서 공통원인고장(Common Cause Failure) 발생 가능성이 증가
    - 이를 방지하기 위해 다양성 확보 필요

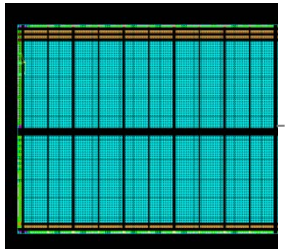
PLC (Programmable Logic Controller)

FPGA (Field Programmable Gate Array)

FPGA 개발 프로세스







Requirement or Design Specification

RTL Design (Verilog or VHDL)

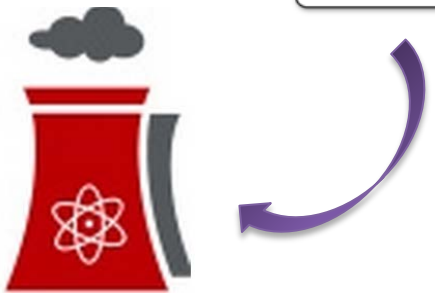
Logic Synthesis (3rd Party Tools)

Gate-Level Design (Netlist)

P&R

Layout

FPGA

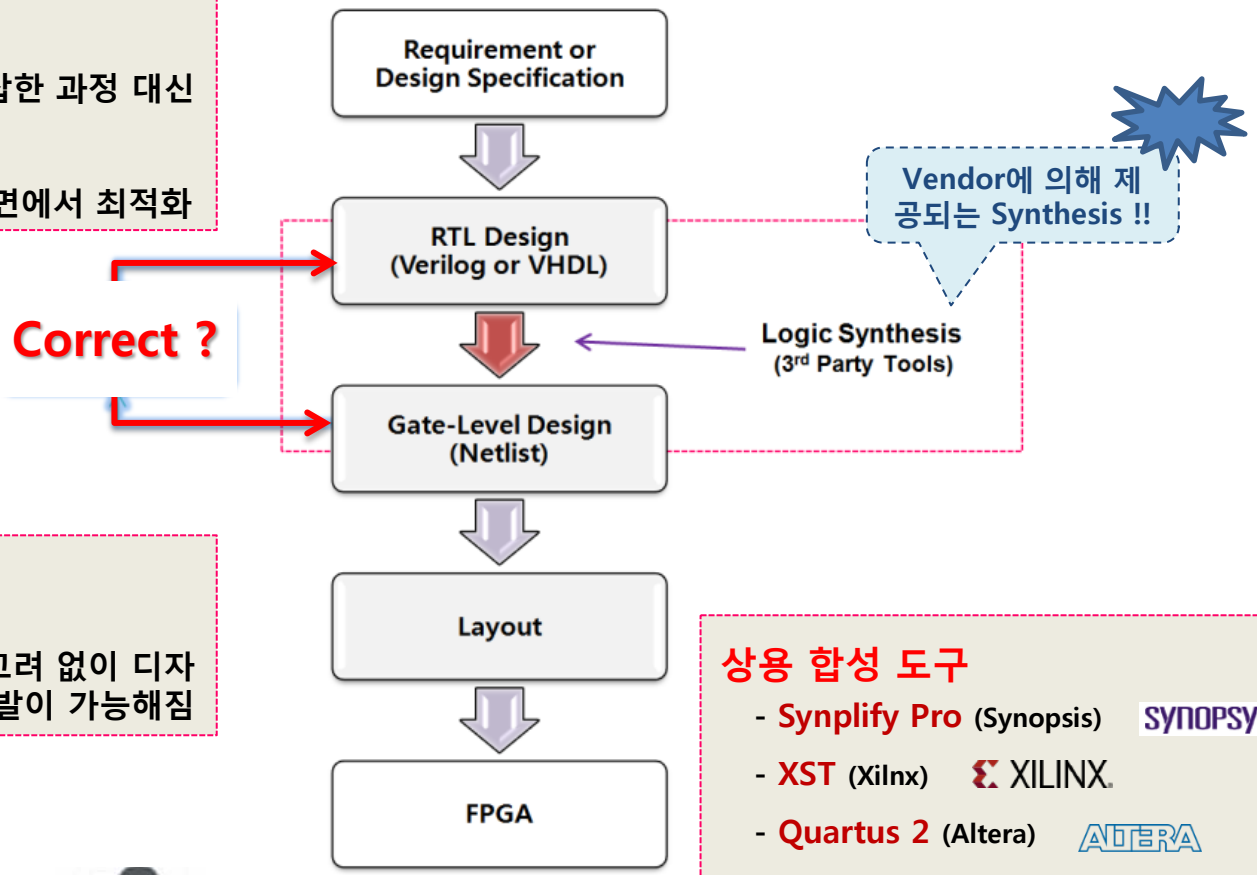


FPGA 개발 프로세스

Synthesis (합성)

HDL을 gate-level로 변환

사용자가 고려 해야 하는 복잡한 과정 대신 수행
Optimization
칩의 speed, cost, power 측면에서 최적화



설계의 질적 향상

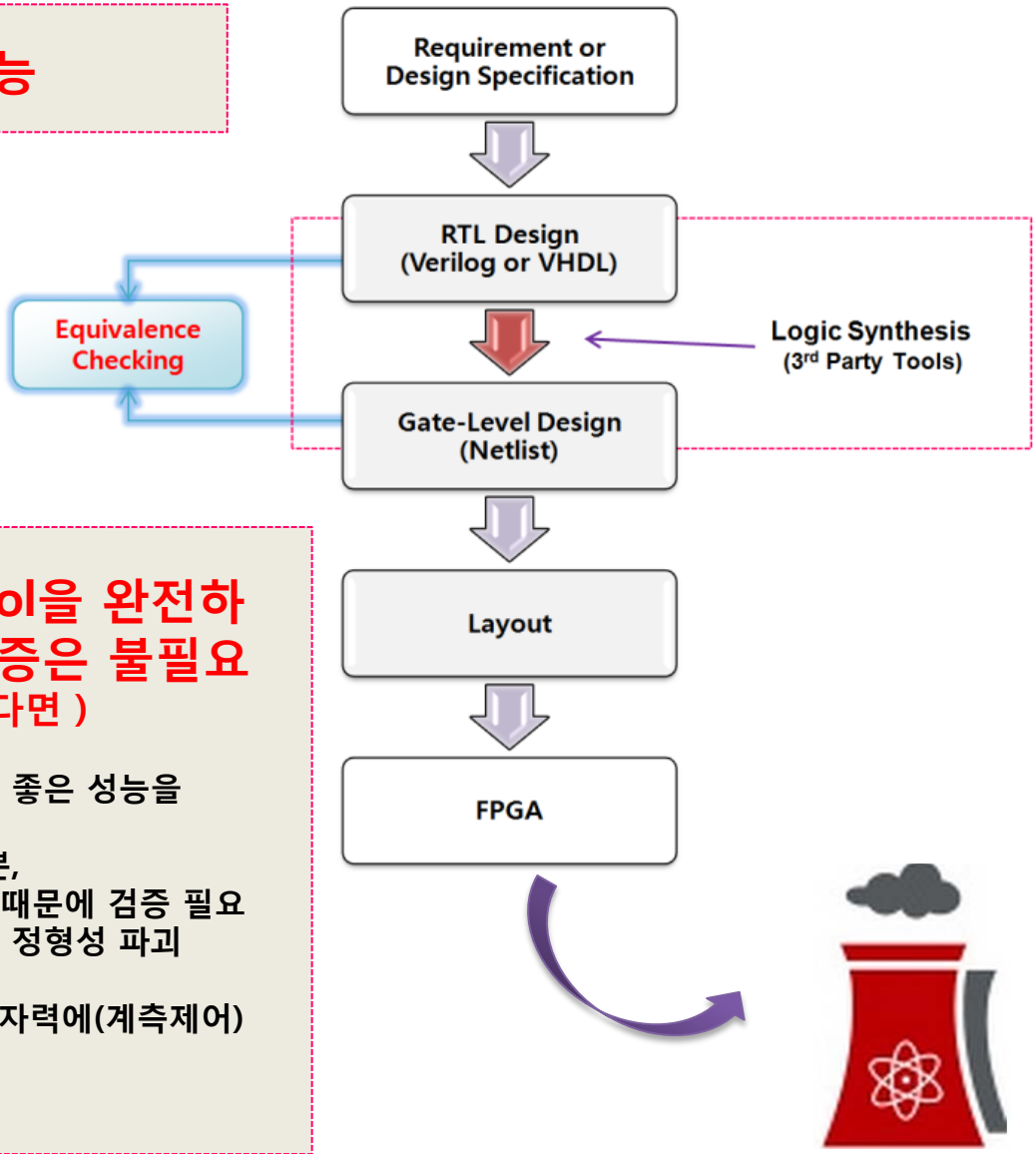
사용자는 위와 같은 복잡한 고려 없이 디자인 및 설계에 집중하여 칩 개발이 가능해짐

상용 합성 도구

- Synplify Pro (Synopsis)
- XST (Xilnx)
- Quartus 2 (Altera)
- Encounter RTL Compiler (Cadence)
- Preceision RTL (Mentor Graphics)

FPGA 개발 프로세스

EC를 통해 검증 가능

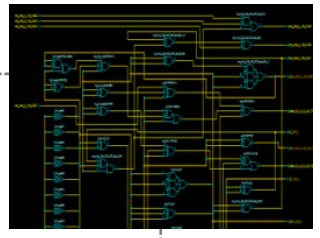


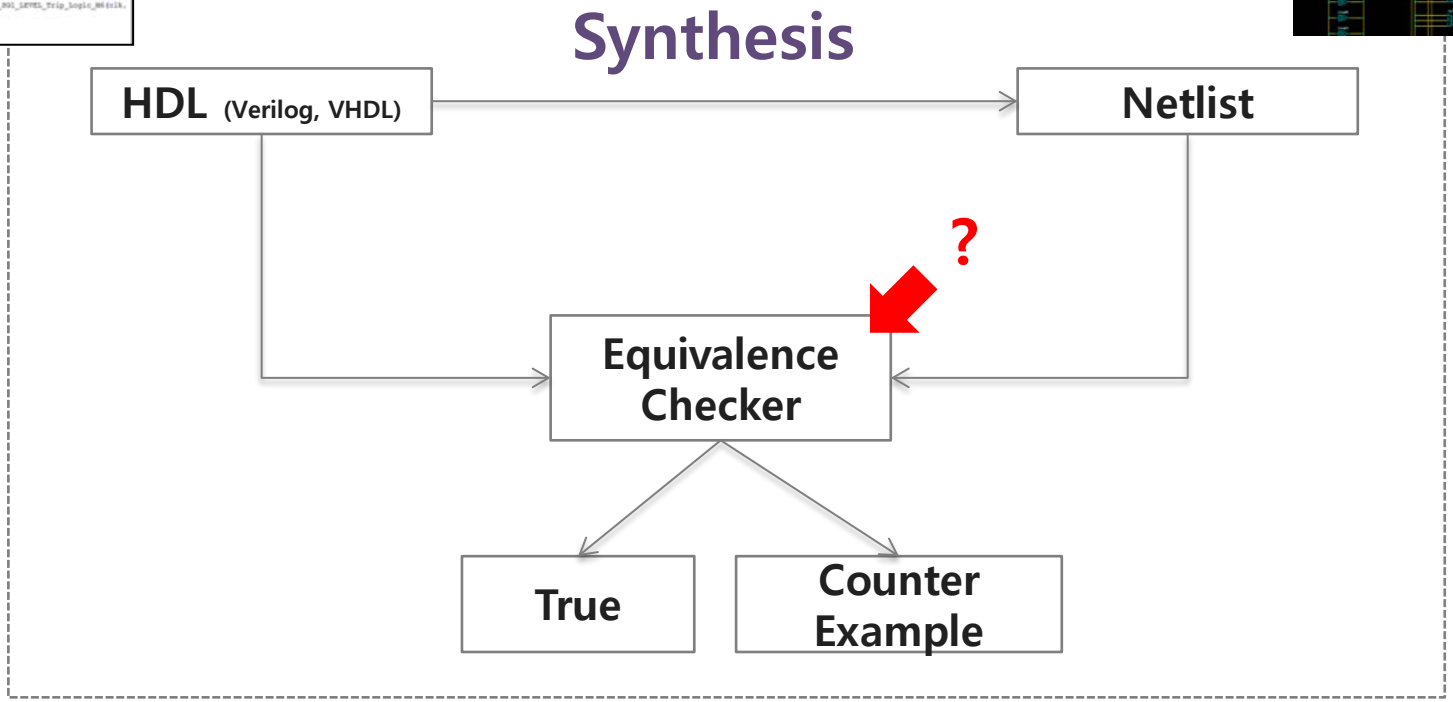
만약 Synthesis Tool을 완전하게 믿는다면 EC 검증은 불필요
(또는 이미 인증이 되었다면)

- 대부분의 합성 도구들이 좋은 성능을 보여주지만,
- 아직 밝혀지지 않았을 뿐, 오류가 존재 할 수 있기 때문에 검증 필요
- 개발자가 의도한 구조의 정형성 파괴

- 아직 국내에 FPGA가 원자력에(계측제어) 사용된 이력이 없음
- 인증 된 적 없음

Equivalence Checking





주어진 HDL 로직이
기능이 일치하는 Netlist로 변환 되었다는 것을 증명 가능

Commercial Equivalence Checkers

다양한 Equivalence Checker 존재

- 하지만 특정 개발 환경과 특정 합성 도구의 조합에 한해서만 검증 지원
- 지원되지 않는 상황도 존재

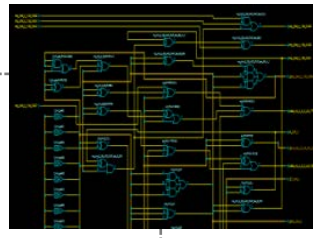
Logic Synthesis	IDE	Mentor Graphics FormalPro	Cadence Encounter Conformal EC	Synopsys Formality
Mentor Graphics Precision RTL	Xilinx ISE	O		
	Actel Libero Soc	O		
Synopsys Synplify Pro	Xilinx ISE	O	O	
	Actel Libero Soc			
	Altera Quartus II		O	
Xilinx XST	Xilinx ISE		O	O
Synopsys DC Ultra	-			O

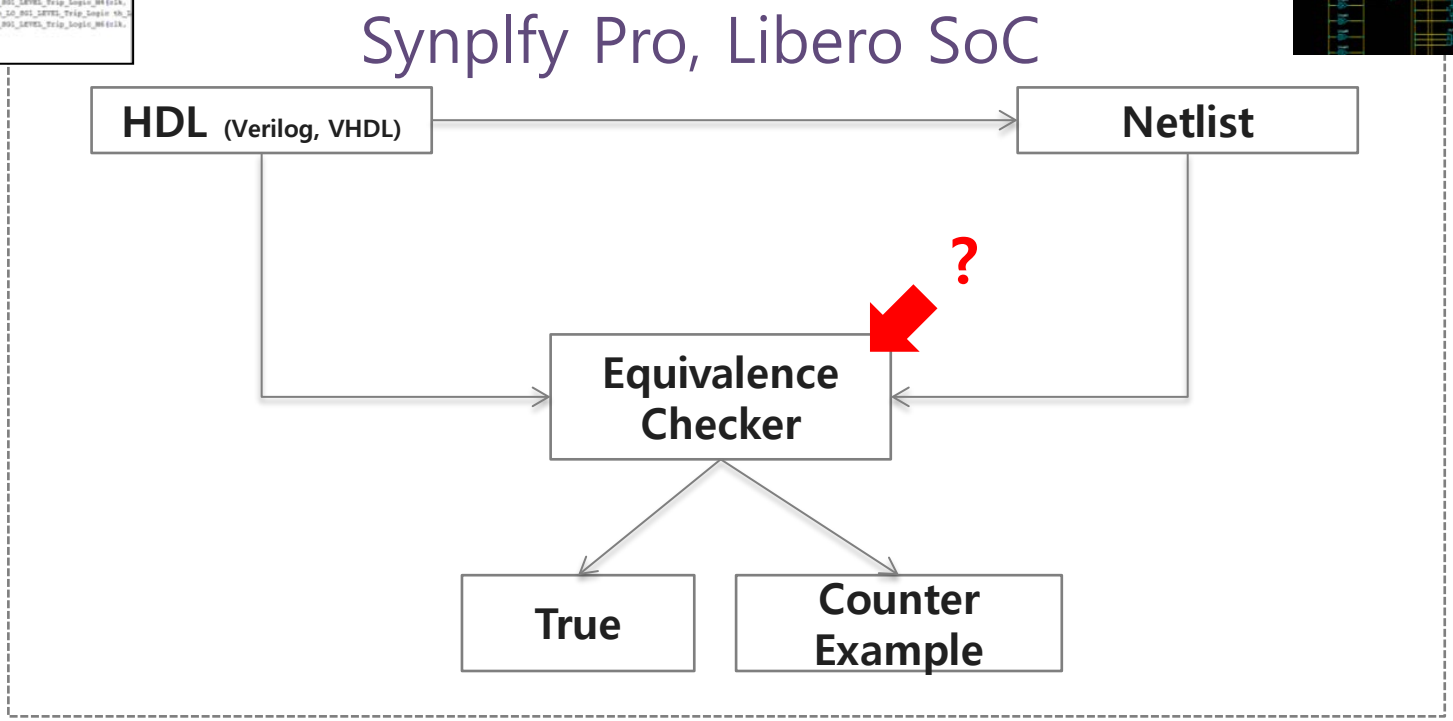
No LEC available

VIS 를 이용해 검증을 해보자.

- 무료
- 소스 오픈 (공개적인 검증 가능)

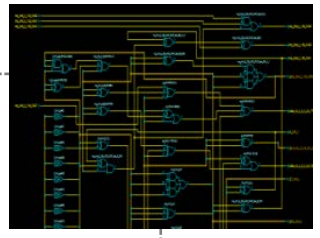
VIS 기반의 Equivalence Checking 방법 사용



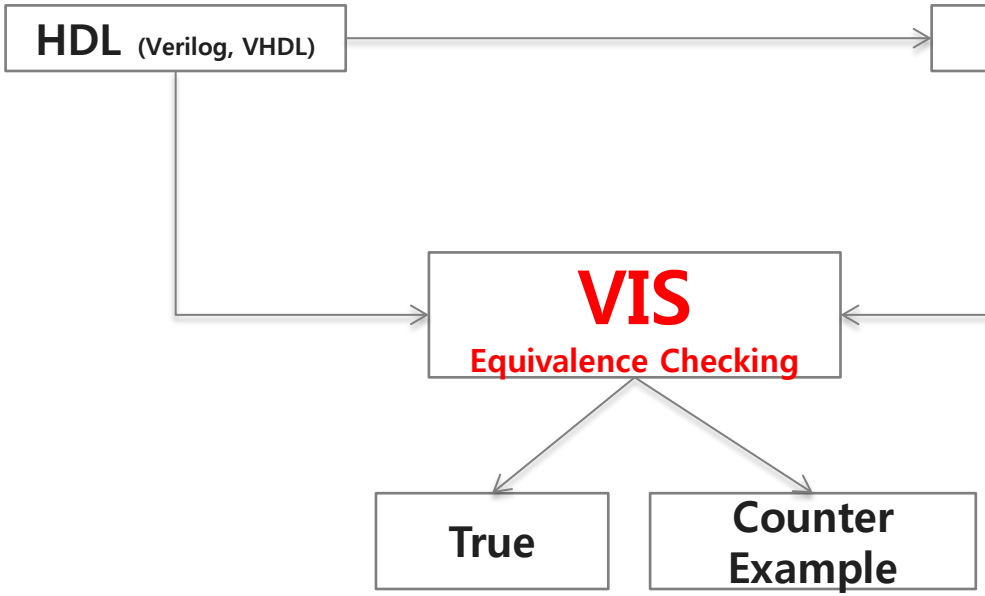


VIS로 인한 문제점

VIS가 Verilog의 모든 구문과 문법을 지원 X



Synplfy Pro, Libero SoC



VIS의 EC를 위해 필요한 Verilog 제약사항

- (1) 클럭 관련 제약사항
  - always 구문 안에는 오직 하나의 클럭(clk)만 사용
  - negedge에 동기화된 클럭의 사용 불가
  - always 구문 안에 딜레이와 관련된 구문 사용 불가

- (2) 비결정성 관련 제약사항
  - non-blocking 구문 사용 불가
  - register 변수 사용 시 0으로 초기화

- (3) 구문 오류 관련 제약사항
  - integer형 변수의 register 사용 불가
  - parameter 변수 사용 시 bits 의 크기 지정 불가

예 1)

always 구문 안에는 오직 하나의 클럭(clk)만 사용 가능

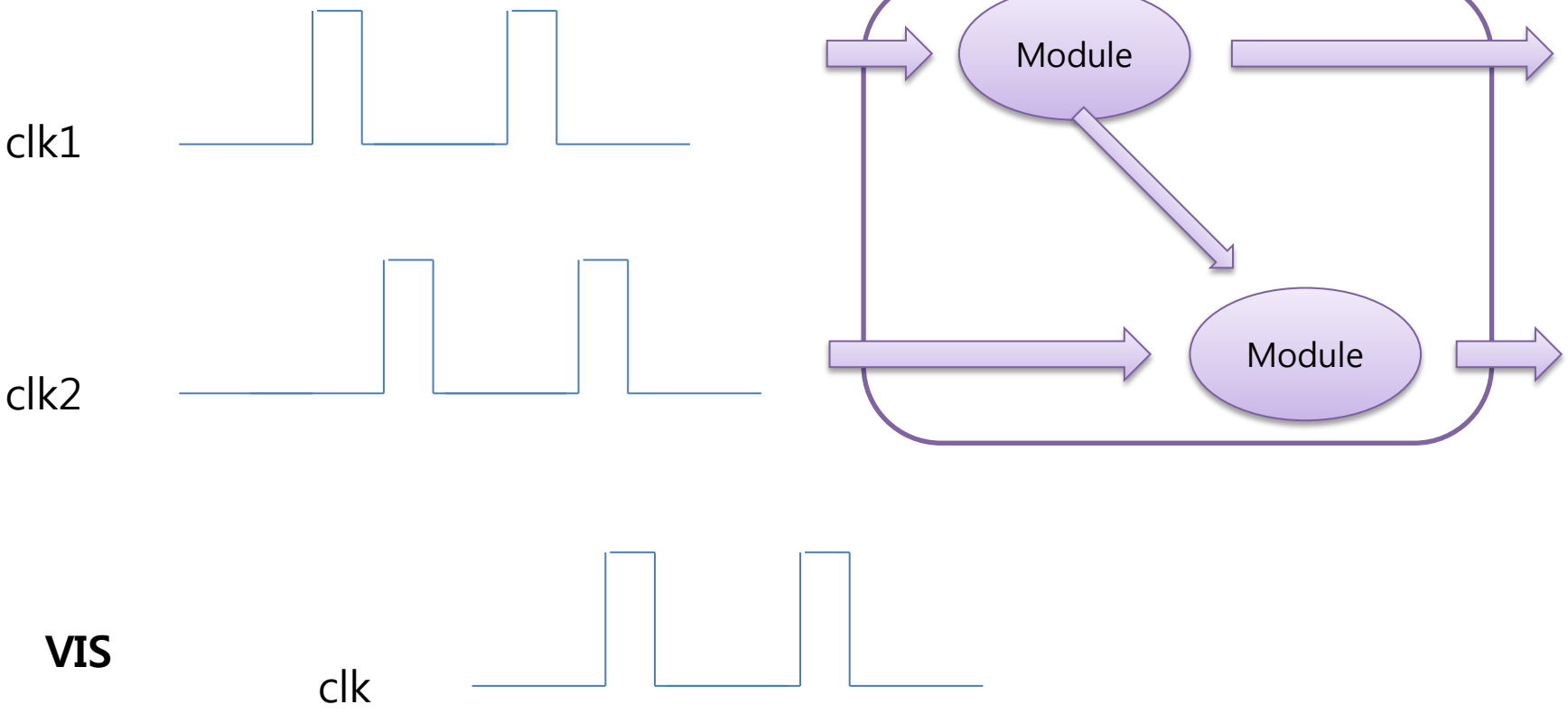
VIS는 always 구문 안의 signal들을 clk로 취급
+ 다중 클럭의 사용 X

```
always @(posedge rst or posedge clk)
begin
        if(rst) begin
                out_tmp <= 0;
                R_o <= 0;
        end else if(clk) begin
                out_tmp <= (A_i + B_i);
                R_o <= (A_i + B_i);
        end
end
```

```
always @(posedge clk)
begin
        if(rst) begin
                out_tmp = 0;
                R_o = 0;
        end else begin
                out_tmp = (A_i + B_i);
                R_o = (A_i + B_i);
        end
end
```

예 1)

원래, HDL 작성 때 다중 클럭을 사용하여 개발 하는 것이 가능
하지만, VIS는 하나의 clk 만 있는 것으로 취급
다양한 기능 사용 VS 검증



예 2)

register 변수 사용 시 초기화를 해 주어야 함

사실 HDL 작성 때 초기화를 하지 않아도 됨
합성도구가 초기화 구문을 삭제
하지만, VIS는 Equivalence Checking 때 값이 할당되지 않았다는 메시지 출력

```
output [31:0] R_o;          reg [31:0] R_o;
```

```
initial begin
        R_o = 0;
        out_tmp = 0;
end
```

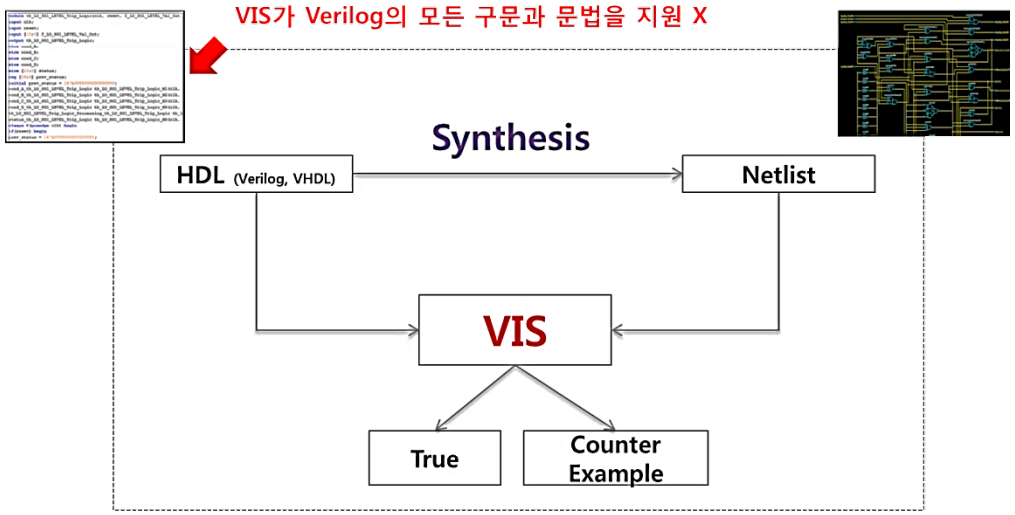
The original Verilog program

```
module ADD_INT_2(rst, clk, A_i, B_i, R_o, E_o);
        input rst;
        input clk;
        input [31:0] A_i;
        input [31:0] B_i;
        output [31:0] R_o;          reg [31:0] R_o;
        output E_o;

        parameter [31:0] INT_HI = 32767;
        parameter [31:0] INT_LO = 32767;

        integer out_tmp;

        always @(posedge rst or posedge clk)
        begin
                if(rst) begin
                        out_tmp <= 0;
                        R_o <= 0;
                end else if(clk) begin
                        out_tmp <= (A_i + B_i);
                        R_o <= (A_i + B_i);
                end
        end

        assign E_o = ((out_tmp > INT_HI)
                | (out_tmp < INT_LO)) ? 1'b1 : 1'b0;
endmodule
```

The Modified Verilog program

```
module ADD_INT_2(rst, clk, A_i, B_i, R_o, E_o);
        input rst;
        input clk;
        input [31:0] A_i;
        input [31:0] B_i;
        output [31:0] R_o;          reg [31:0] R_o;
        output E_o;

        parameter INT_HI = 32767;
        parameter INT_LO = 32767;

        reg [31:0] out_tmp;
        initial begin
                R_o = 0;
                out_tmp = 0;
        end

        always @(posedge clk)
        begin
                if(rst) begin
                        out_tmp = 0;
                        R_o = 0;
                end else begin
                        out_tmp = (A_i + B_i);
                        R_o = (A_i + B_i);
                end
        end

        assign E_o = ((out_tmp > INT_HI)
                | (out_tmp < INT_LO)) ? 1'b1 : 1'b0;
endmodule
```

고찰



제약사항

- (1) 클럭 관련 제약사항
  - always 구문 안에는 오직 하나의 클럭(clk)만 사용
  - negedge에 동기화된 클럭의 사용 불가
  - always 구문 안에 딜레이와 관련된 구문 사용 불가
- (2) 비결정성 관련 제약사항
  - non-blocking 구문 사용 불가
  - register 변수 사용 시 0으로 초기화
- (3) 구문 오류 관련 제약사항
  - integer형 변수의 register 사용 불가
  - parameter 변수 사용 시 bits 의 크기 지정 불가

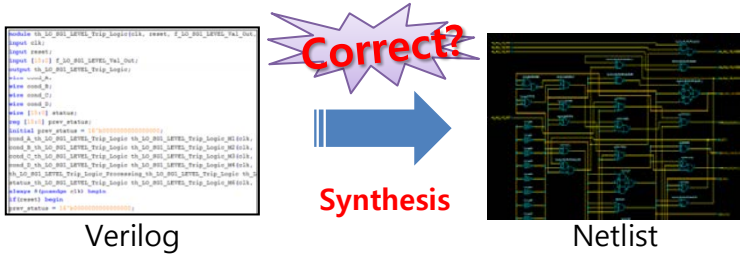
개발자의 자유로운 코딩을 방해 하는 요소일 수 있음

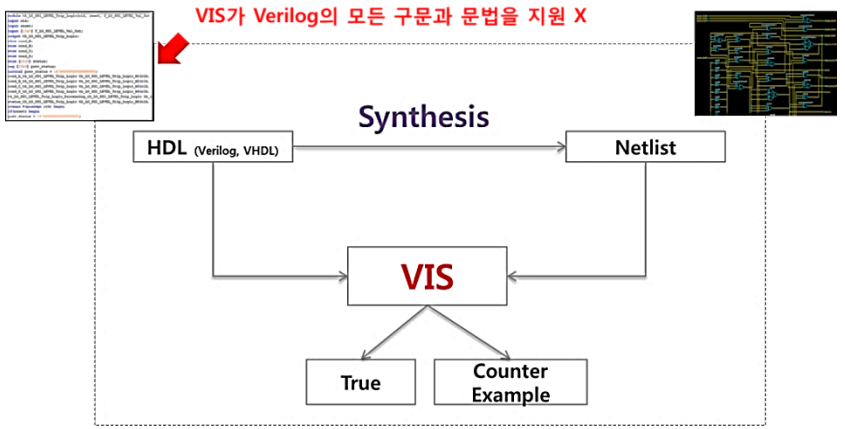
- 하지만 VIS EC 검증을 위해서는 필요
- 기능 VS 검증 → 선택이 필요

정리한 제약사항은
현재 발견된 제약 사항..
추가적인 제약사항이 존재할 수 있음
제약사항의 평가도 필요

요약

- 원자력 발전소의 계측제어 시스템에 FPGA 사용에 있어…
- FPGA 개발 시 Vendor 의존적인 **Synthesis** 과정 필요 (과연 정확한 변환을 하는가?)
- **VIS**를 이용한 Equivalence Checking 을 통해 검증 수행 계획
  - **하지만, VIS가 모든 Verilog 구문과 문법을 지원하고 있지 않음**
- **VIS의 Equivalence Checking을 위한 Verilog의 제약사항 제시**





- 앞으로
  - 제약사항의 평가, 추가적인 제약사항 확인
  - 합성도구의 검증 진행, 인증 진행

감사합니다

http://dslab.konkuk.ac.kr
atang34@konkuk.ac.kr