FBDtoVHDL: FPGA 개발을 위한 FBD로부터 VHDL 자동 변환 도구

김재엽[+], 김의섭[+], 유준범[+], 이영준[*], 최종균[*]

[+]건국대학교
[*]한국원자력연구원

Outline

1. Introduction
2. Background
    1. Function Block Diagram
    2. VHSIC Hardware Description Language
3. FBDtoVHDL
    1. 변환 규칙
    2. 도구 구현
4. 사례 연구
5. 결론

Introduction

- 최근 PLC 기반 계측제어 시스템에서의 다양한 문제가 떠오름
  - 공통원인고장(Common Cause Failure) 발생, 유지보수비용 증가 등
- 문제를 해결하기 위한 방법 중 하나로 FPGA 기반 시스템을 이용
  - PLC 기반 시스템과 FPGA 기반 시스템을 함께 사용하여 다양성 확보

PLC

FPGA

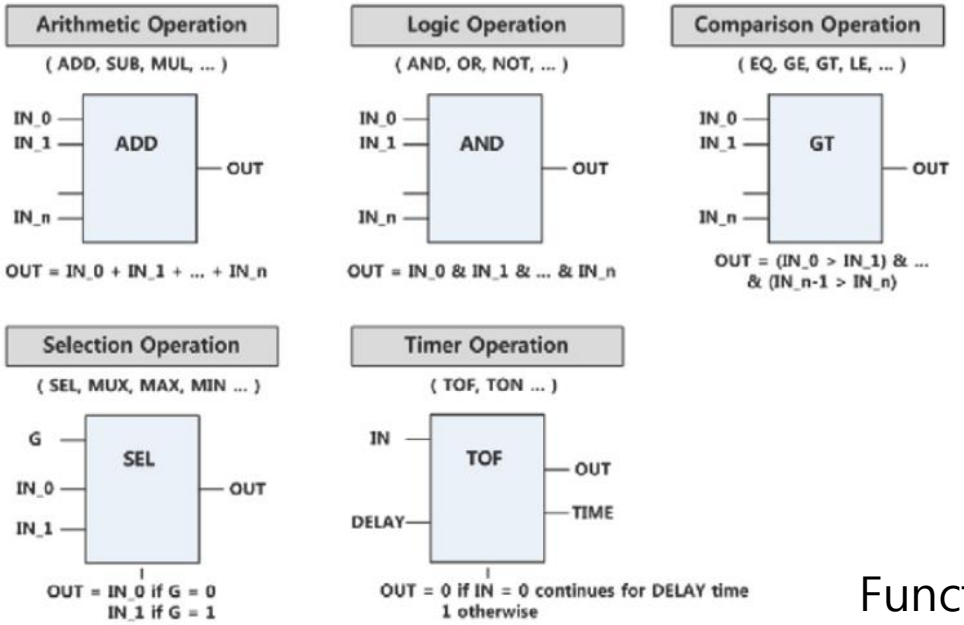
다양성 (Diversity)

- 문제점
  - PLC와 FPGA의 개발을 위해 사용되는 언어 및 프로세스 차이로 인한 문제들
- 제안하는 방법
  - PLC 개발 언어인 FBD를 기능적으로 동일한 VHDL 설계로 자동 변환

Background (1/2)

Function Block Diagram (FBD)

- IEC 61131-3 선언된 PLC 개발을 위한 5가지 언어 중 하나
  - FBD, ST, LD, IL, SFC
- 산술, 논리, 선택 등의 연산을 수행하는 블록을 배치 및 연결하여 프로그램을 작성하는 그래픽 기반 언어



Function Block의 예

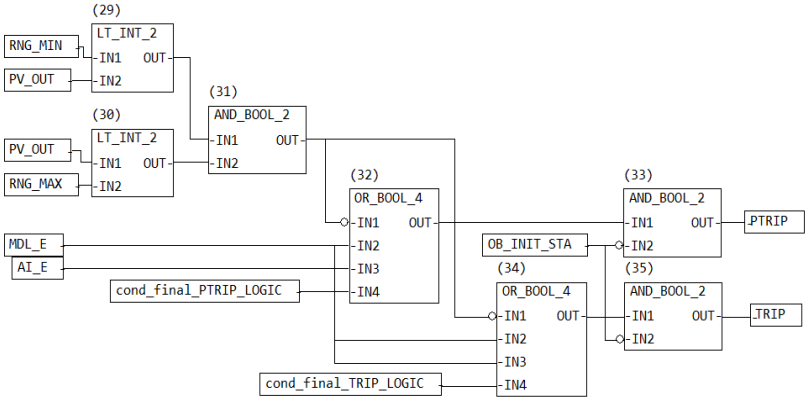
Background (2/2)

VHSIC Hardware Description Language (VHDL)

- VHDL은 IEEE 표준 언어로 FPGA 개발에 널리 사용되는 언어
- 타 언어에 비해 더 자세한 수준의 설계가 가능
  - 시스템 수준에서 게이트 수준까지 표현 가능
- 다수의 EDA 도구들이 VHDL을 이용한 합성 과정을 지원

FBDtoVHDL

- FBDtoVHDL은 FBD 프로그램을 POU를 기준으로 변환
  - POU: function, function block, program
  - FBD는 POU의 계층 구조
- 7가지 변환 규칙이 존재하며 변환 결과가 적용되는 프레임을 정의



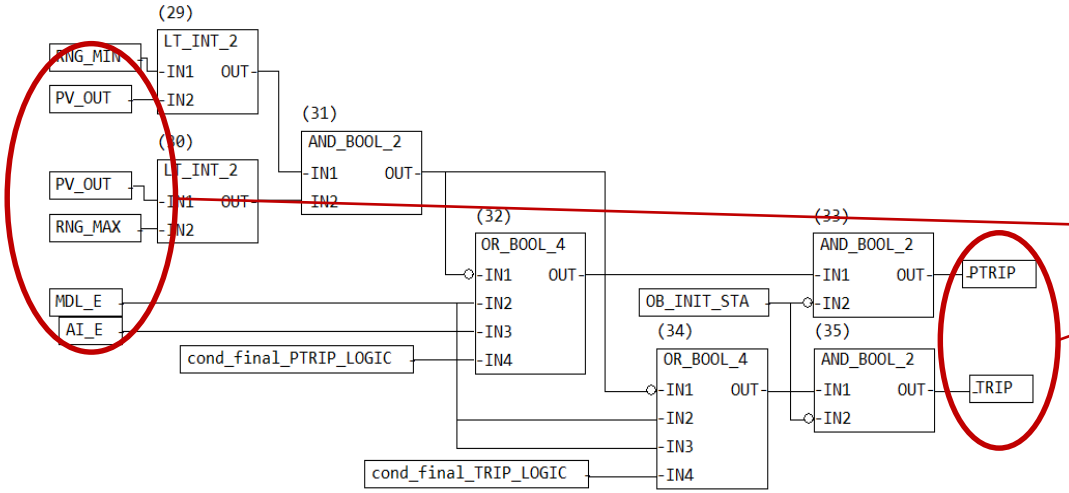
사용자 정의 POU (FBD)

```
1:  entity FIX_FALLING is (
2:   generic(
3:      INT_HI : integer := 8388607;
4:      INT_LO : integer := -8388608
5:   );
6:   port(
7:      clk : in std_logic;
8:      rst : in std_logic;
9:      pulse : in std_logic;
10:
11:     PV_OUT : in integer range INT_LO to INT_HI;
12:
13:  end FIX_FALLING;
14:
15:  architecture Behavioral of FIX_FALLING is
16:     signal RNG_MIN: integer range INT_LO to INT_HI := 600;
17:
18:     signal LT_INT_2_wire_29_OUT : std_logic;
19:
20:     LT_INT_2_29 : entity work.LT_INT_2 port map
21:      (clk => clk, rst => rst, A_i => RNG_MIN,
22:       B_i => PV_OUT, R_o=>LT_INT_2_wire_29_OUT);
23:
24:     TRIP <= AND_BOOL_2_wire_35_OUT;
25:
26:   process(clk, rst) begin
27:     if(rst = '1') then
28:        TRIP_LOGIC <= '0';
29:     elseif(pulse = '1') then
30:        TRIP_LOGIC <= SEL_BOOL_2_wire_27;
31:     end if;
32:   end process;
33:  end Behavioral;
```

VHDL Entity
(FBD 변환 결과 + 프레임)

변환 규칙 (1/7)

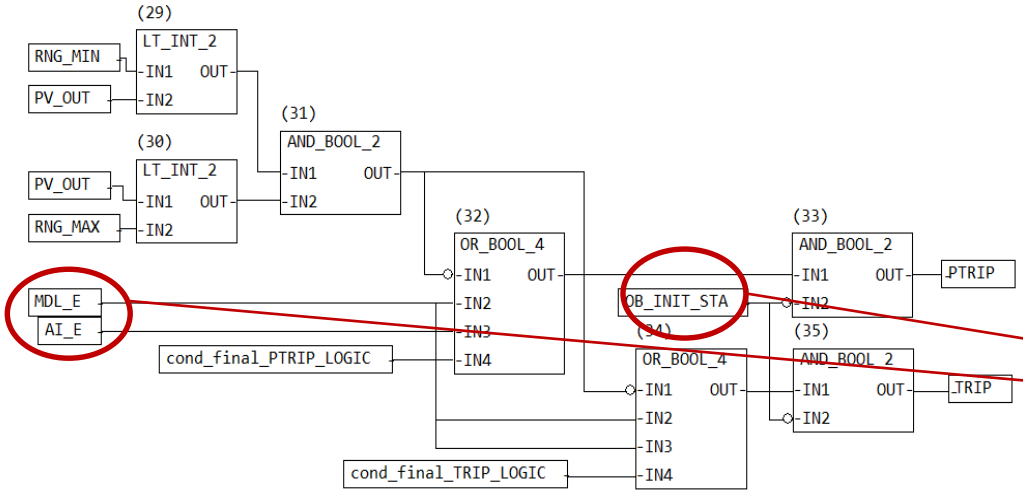
- **규칙 1**
  - 입출력 값을 변환




```
1: entity [POUName] is (
2:  generic ();
3:  port (
4:      clk : in std_logic;
5:      rst : in std_logic;
6:      pulse : in std_logic;
7:
8:    [[input Name] : in [Data Type;]] +          -- INPUT
9:    [[output Name] : out [Data Type;]] +        -- OUTPUT
10:    [[output Name] : buffer [Data Type;]] +    -- FEEDBACK
11:  );
12: end [POUName];
13:
14: architecture Behavioral of [POUName] is
15:  [signal [constant Name] : [Data Type] := [initial Value;]] +
16:                                               -- CONSTANT
17:  [signal [CON Name] : [Data Type;]]+
18:                                   -- Connect/Continuation (CON)
19:
20:  [[FBname_[local id]] : entity work.[entity] port map
21:   ([,[portName] => clk|rst|INPUT|CONSTANTS|CON])*;]]+
22:                                               -- Entity Call
23:
24:  [[CON|OUTPUT] <= [CONSTANTS|CON;]]+  -- Assignment
25:
26:  process(clk, rst) begin
27:    if(rst = '1') then
28:      [[OUTPUT|FEEDBACK] <= [initialValue;]]+
29:                                           -- Output Initialization
30:    elseif(pulse = '1') then
31:      [[FFEDBACK] <= [CONSTANTS|CON;]]+
32:                                           -- Feedback Assignment
33:    end if;
34:  end process;
35: end Behavioral;
```

변환 규칙 (2/7)

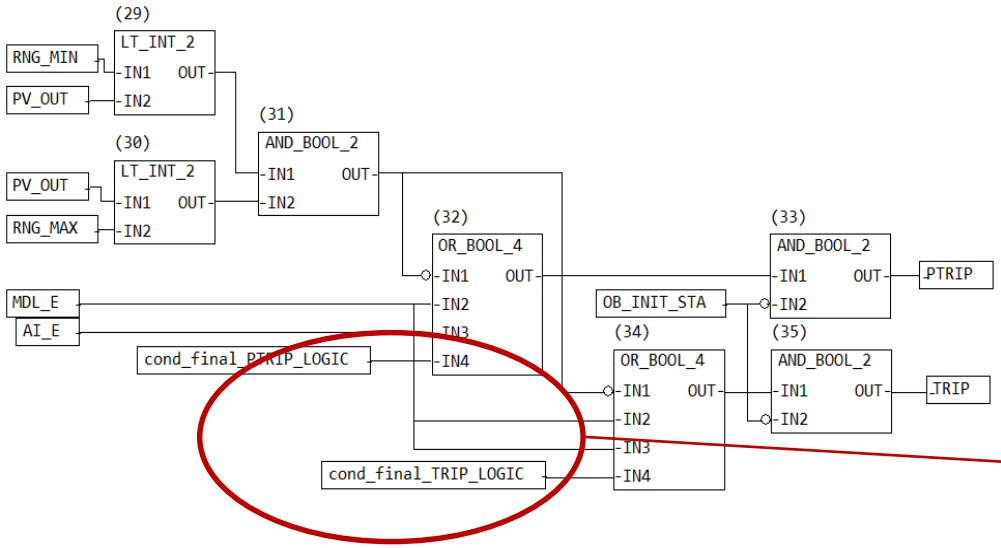
- 규칙 2
  - 상수 값 변환




```
1: entity [POUName] is (
2:  generic ();
3:  port (
4:      clk : in std_logic;
5:      rst : in std_logic;
6:      pulse : in std_logic;
7:
8:      [[input Name] : in [Data Type;]] +          -- INPUT
9:      [[output Name] : out [Data Type;]] +        -- OUTPUT
10:     [[output Name] : buffer [Data Type;]] +     -- FEEDBACK
11:  );
12: end [POUName];
13:
14: architecture Behavioral of [POUName] is
15:  [signal [constant Name] : [Data Type] := [initial Value;]] +
16:                                                 -- CONSTANT
17:  [signal [CON Name] : [Data Type;]]+
18:                                  -- Connect/Continuation (CON)
19:
20:  [[FBname_[local id]] : entity work.[entity] port map
21:   ([,[portName] => clk|rst|INPUT|CONSTANTS|CON])*;]]+
22:                                                 -- Entity Call
23:
24:  [[CON|OUTPUT] <= [CONSTANTS|CON;]]+  -- Assignment
25:
26:  process(clk, rst) begin
27:    if(rst = '1') then
28:      [[OUTPUT|FEEDBACK] <= [initialValue;]]+
29:                                                 -- Output Initialization
30:    elseif(pulse = '1') then
31:      [[FFEDBACK] <= [CONSTANTS|CON;]]+
32:                                                 -- Feedback Assignment
33:    end if;
34:  end process;
35: end Behavioral;
```

변환 규칙 (3/7)

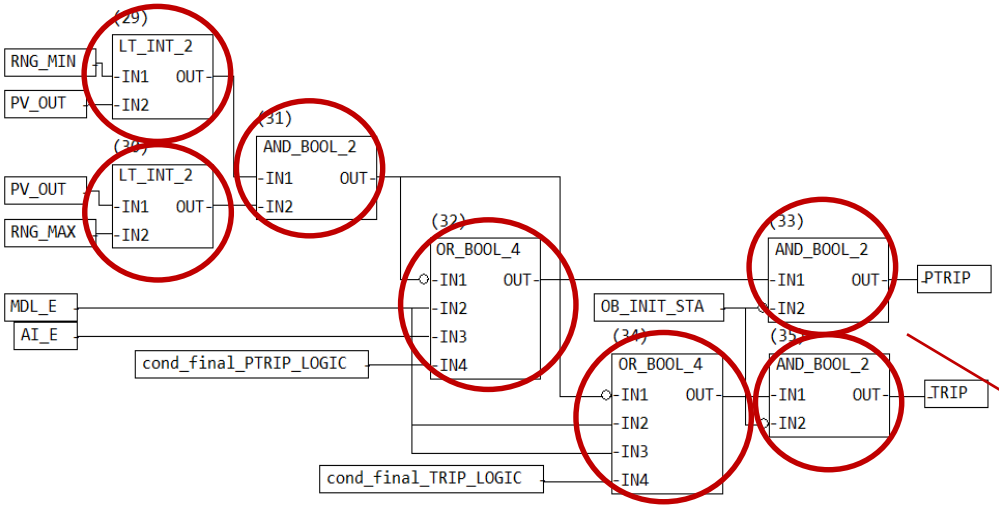
- 규칙 3
  - Connection/Continuation 변환




```
1: entity [POUName] is (
2:  generic ();
3:  port (
4:      clk : in std_logic;
5:      rst : in std_logic;
6:      pulse : in std_logic;
7:
8:      [[input Name] : in [Data Type;]] +          -- INPUT
9:      [[output Name] : out [Data Type;]] +        -- OUTPUT
10:     [[output Name] : buffer [Data Type;]] +     -- FEEDBACK
11:  );
12: end [POUName];
13:
14: architecture Behavioral of [POUName] is
15:  [signal [constant Name] : [Data Type] := [initial Value;]] +
16:                                                 -- CONSTANT
17:  [signal [CON Name] : [Data Type;]]+
18:                                                 -- Connect/Continuation (CON)
19:
20:  [[FBname_[local id]] : entity work.[entity] port map
21:   ([,[portName] => clk|rst|INPUT|CONSTANTS|CON])*;]]+
22:                                                 -- Entity Call
23:
24:  [[CON|OUTPUT] <= [CONSTANTS|CON;]]+  -- Assignment
25:
26:  process(clk, rst) begin
27:    if(rst = '1') then
28:      [[OUTPUT|FEEDBACK] <= [initialValue;]]+
29:                                                 -- Output Initialization
30:    elseif(pulse = '1') then
31:      [[FFEDBACK] <= [CONSTANTS|CON;]]+
32:                                                 -- Feedback Assignment
33:    end if;
34:  end process;
35: end Behavioral;
```

변환 규칙 (4/7)

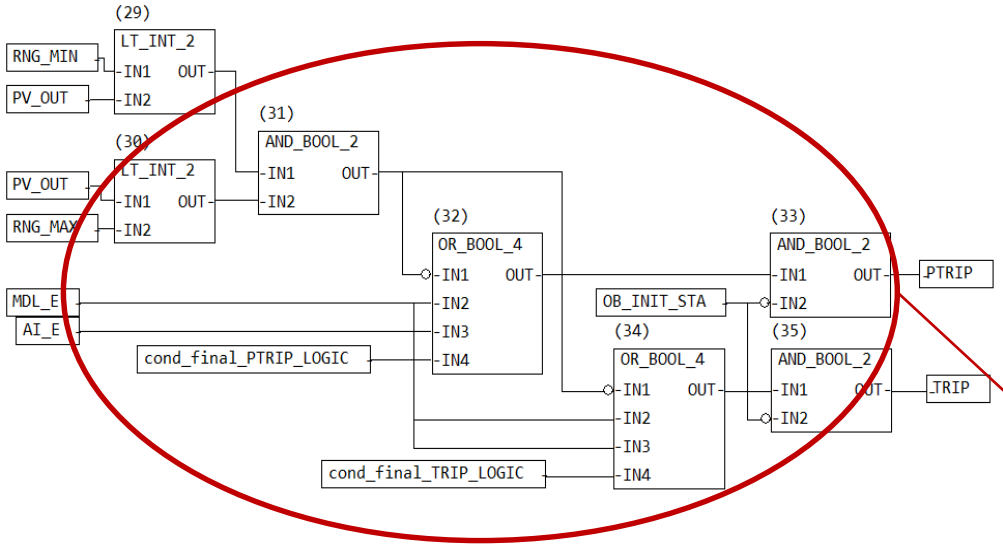
- **규칙 4**
  - Function Block 변환




```
1: entity [POUName] is (
2:  generic ();
3:  port (
4:      clk : in std_logic;
5:      rst : in std_logic;
6:      pulse : in std_logic;
7:
8:      [[input Name] : in [Data Type;]] +          -- INPUT
9:      [[output Name] : out [Data Type;]] +        -- OUTPUT
10:     [[output Name] : buffer [Data Type;]] +     -- FEEDBACK
11:  );
12: end [POUName];
13:
14: architecture Behavioral of [POUName] is
15:  [signal [constant Name] : [Data Type] := [initial Value;]] +
16:                                                 -- CONSTANT
17:  [signal [CON Name] : [Data Type;]]+
18:                                  -- Connect/Continuation (CON)
19:
20:  [[FBname_[local id]] : entity work.[entity] port map
21:   ([,[portName] => clk|rst|INPUT|CONSTANTS|CON])*;]]+
22:                                                 -- Entity Call
23:
24:  [[CON|OUTPUT] <= [CONSTANTS|CON;]]+  -- Assignment
25:
26:  process(clk, rst) begin
27:    if(rst = '1') then
28:      [[OUTPUT|FEEDBACK] <= [initialValue;]]+
29:                                          -- Output Initialization
30:    elseif(pulse = '1') then
31:      [[FFEDBACK] <= [CONSTANTS|CON;]]+
32:                                          -- Feedback Assignment
33:    end if;
34:  end process;
35: end Behavioral;
```

변환 규칙 (5/7)

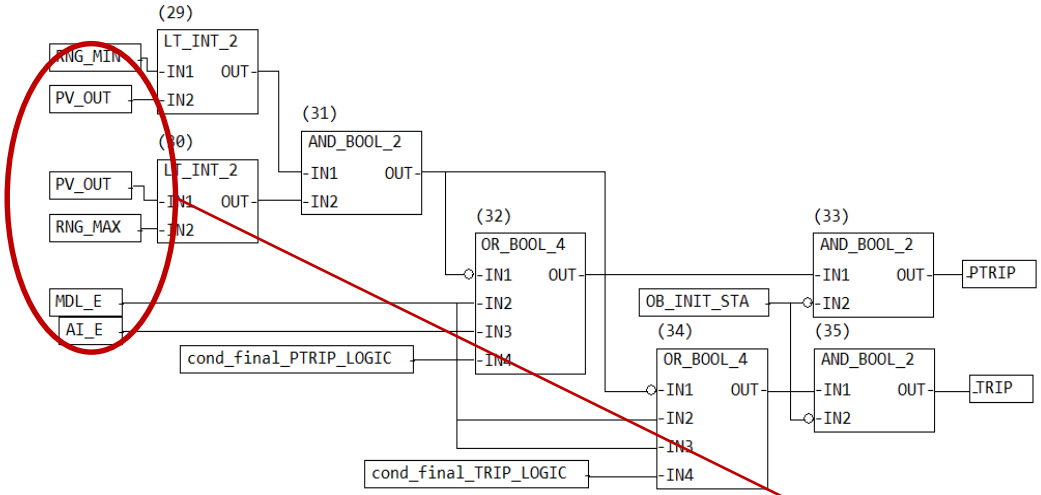
- **규칙 5**
  - 값 할당 부분 변환




```
1: entity [POUName] is (
2:  generic ();
3:  port (
4:      clk : in std_logic;
5:      rst : in std_logic;
6:      pulse : in std_logic;
7:
8:      [[input Name] : in [Data Type;]] +          -- INPUT
9:      [[output Name] : out [Data Type;]] +        -- OUTPUT
10:     [[output Name] : buffer [Data Type;]] +     -- FEEDBACK
11:  );
12: end [POUName];
13:
14: architecture Behavioral of [POUName] is
15:  [signal [constant Name] : [Data Type] := [initial Value;]] +
16:                                                 -- CONSTANT
17:  [signal [CON Name] : [Data Type;]]+
18:                                  -- Connect/Continuation (CON)
19:
20:  [[FBname_[local id]] : entity work.[entity] port map
21:   ([,[portName] => clk|rst|INPUT|CONSTANTS|CON])*;]]+
22:                                                 Entity Call
23:
24:  [[CON|OUTPUT] <= [CONSTANTS|CON;]]+  -- Assignment
25:
26:  process(clk, rst) begin
27:    if(rst = '1') then
28:      [[OUTPUT|FEEDBACK] <= [initialValue;]]+
29:                                                 -- Output Initialization
30:    elseif(pulse = '1') then
31:      [[FFEDBACK] <= [CONSTANTS|CON;]]+
32:                                                 -- Feedback Assignment
33:    end if;
34:  end process;
35: end Behavioral;
```

변환 규칙 (6/7)

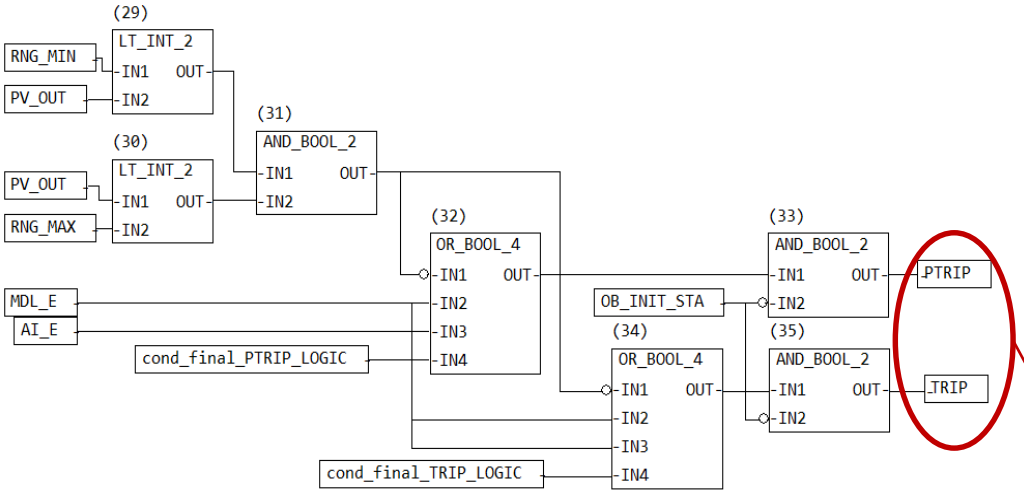
- **규칙 6**
  - 초기 값 설정




```
1: entity [POUName] is (
2:  generic ();
3:  port (
4:      clk : in std_logic;
5:      rst : in std_logic;
6:      pulse : in std_logic;
7:
8:      [[input Name] : in [Data Type;]] +          -- INPUT
9:      [[output Name] : out [Data Type;]] +        -- OUTPUT
10:     [[output Name] : buffer [Data Type;]] +     -- FEEDBACK
11:  );
12: end [POUName];
13:
14: architecture Behavioral of [POUName] is
15:  [signal [constant Name] : [Data Type] := [initial Value;]] +
16:                                                 -- CONSTANT
17:  [signal [CON Name] : [Data Type;]]+
18:                                  -- Connect/Continuation (CON)
19:
20:  [[FBname_[local id]] : entity work.[entity] port map
21:   ([,[portName] => clk|rst|INPUT|CONSTANTS|CON])*;]]+
22:                                                 -- Entity Call
23:
24:  [[CON|OUTPUT] <= [CONSTANTS|CON;]]+  -- Assignment
25:
26:  process(clk, rst) begin
27:    if(rst = '1') then
28:      [[OUTPUT|FEEDBACK] <= [initialValue;]]+
29:                                                 -- Output Initialization
30:    elseif(pulse = '1') then
31:      [[FFEDBACK] <= [CONSTANTS|CON;]]+
32:                                                 -- Feedback Assignment
33:    end if;
34:  end process;
35: end Behavioral;
```

변환 규칙 (7/7)

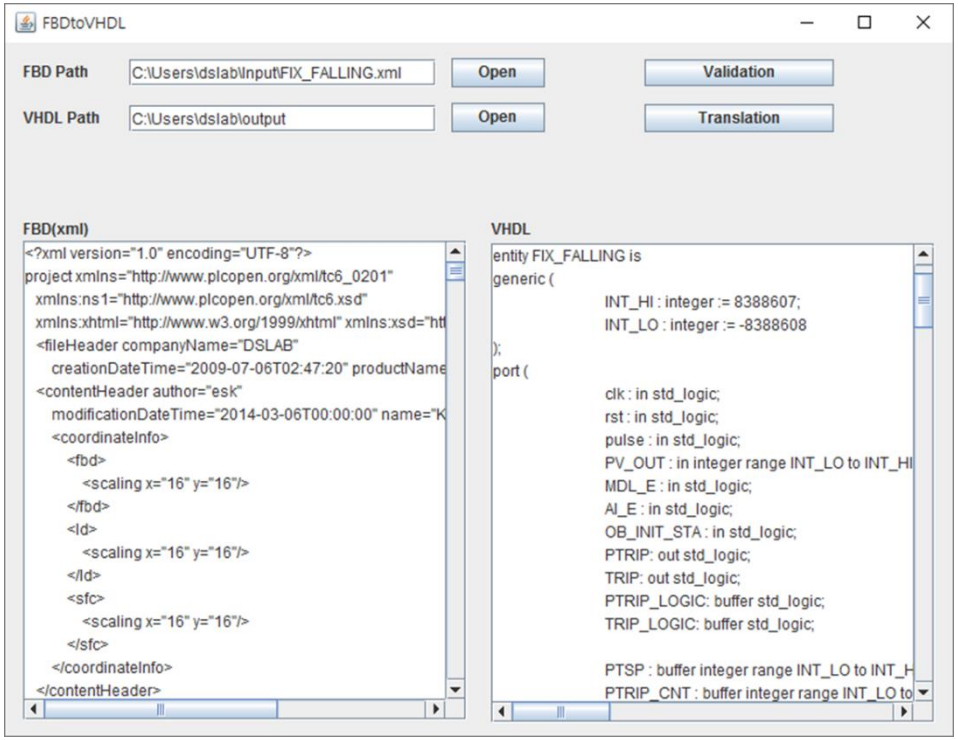
- **규칙 7**
  - 피드백 출력 값 할당




```
1: entity [POUName] is (
2:  generic ();
3:  port (
4:      clk : in std_logic;
5:      rst : in std_logic;
6:      pulse : in std_logic;
7:
8:      [[input Name] : in [Data Type;]] +          -- INPUT
9:      [[output Name] : out [Data Type;]] +        -- OUTPUT
10:     [[output Name] : buffer [Data Type;]] +     -- FEEDBACK
11:  );
12: end [POUName];
13:
14: architecture Behavioral of [POUName] is
15:  [signal [constant Name] : [Data Type] := [initial Value;]] +
16:                                                 -- CONSTANT
17:  [signal [CON Name] : [Data Type;]]+
18:                                  -- Connect/Continuation (CON)
19:
20:  [[FBname_[local id]] : entity work.[entity] port map
21:   ([,[portName] => clk|rst|INPUT|CONSTANTS|CON])*;]]+
22:                                                 -- Entity Call
23:
24:  [[CON|OUTPUT] <= [CONSTANTS|CON;]]+  -- Assignment
25:
26:  process(clk, rst) begin
27:    if(rst = '1') then
28:      [[OUTPUT|FEEDBACK] <= [initialValue;]]+
29:                                          -- Output Initialization
30:    elsif(pulse = '1') then
31:      [[FFEDBACK] <= [CONSTANTS|CON;]]+
32:                                          -- Feedback Assignment
33:    end if;
34:  end process;
35: end Behavioral;
```

도구 구현

- **FBDtoVHDL 자동 변환 도구 구현**
  - PLCopen TC6 XML 버전 2.01 스키마를 만족하는 FBD를 대상으로 함
  - 규칙에 따른 변환을 자동으로 수행



사례 연구

- 두 개의 RPS BP의 trip 로직을 대상으로 실험을 수행
  - FIX_RISING, FIX_FALLING
- 변환 전후의 일치성을 확인하기 위하여 두 가지 분석을 수행
  - 구조적 분석
  - 시뮬레이션을 통한 분석

구조적 분석

- 5가지 항목에 대한 구조적 분석을 수행
  - 입력, 출력, 지역 변수 개수, Function, 초기화

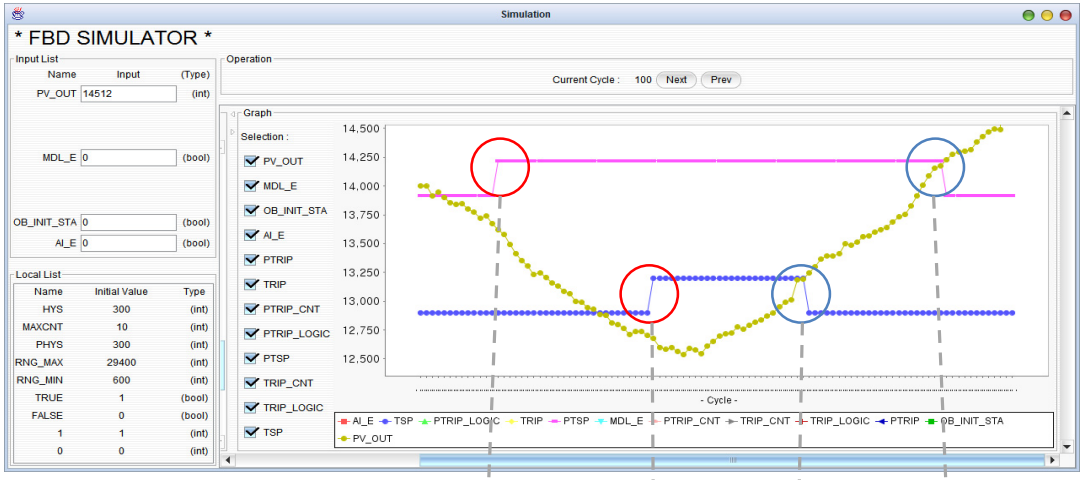
	FBD	VHDL
Input Variable	4	4
Output Variable	8	8 (out: 2, buffer: 6)
Local Variable	11	11
Function	33 (Function Block)	33 (entity call)
Initialization	Correct	

시뮬레이션을 통한 분석

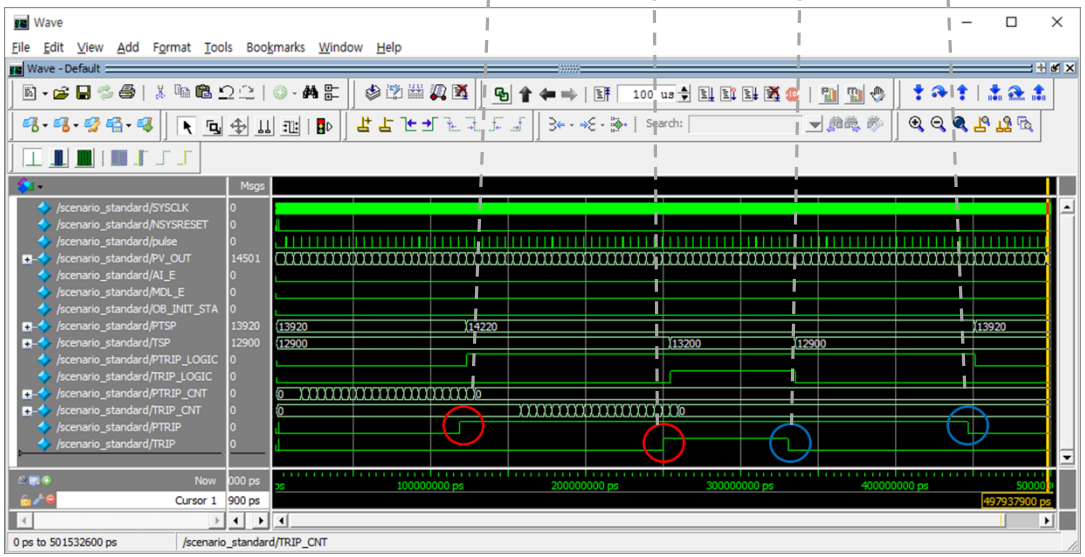
- 동일한 시나리오를 통한 시뮬레이션 결과를 비교 분석
  - FBD Simulator와 ModelSim을 이용한 시뮬레이션 결과 이용

	FIX_RISING	FIX_FALLING
Number of Scenario	1000	1000
Initial Value	27,500	12,500
Rate of Change	10-100 (Stepwise: 10)	10-100 (Stepwise: 10)
Number of Input	100	100
Simulation Results	Correct	

시뮬레이션을 통한 분석



FBD Simulator를 통한
FBD 시뮬레이션 결과



ModelSim을 통한
VHDL 시뮬레이션 결과

결론

- FBD를 VHDL로 변환하기 위한 규칙과 도구 개발
  - 7가지 변환 규칙 + 변환 결과 프레임
  - 자동 변환 도구 FBDtoVHDL 구현
- 변환 전후의 일치성을 확인
  - 구조적 분석
  - 시뮬레이션을 통한 분석
- 일치성 확인은 통하여 1:1 수준의 변환이 이뤄짐을 확인

- 변환 결과를 이용하여 설계 단계 이후의 FPGA 개발 프로세스를 수행하고 실제 FPGA 개발에 문제 없이 사용될 수 있는지 확인할 계획

감사합니다