ISOFIC 2014
2014.08.24~08.28, Jeju

An Integrated Software Development Framework for PLC & FPGA based Digital I&Cs

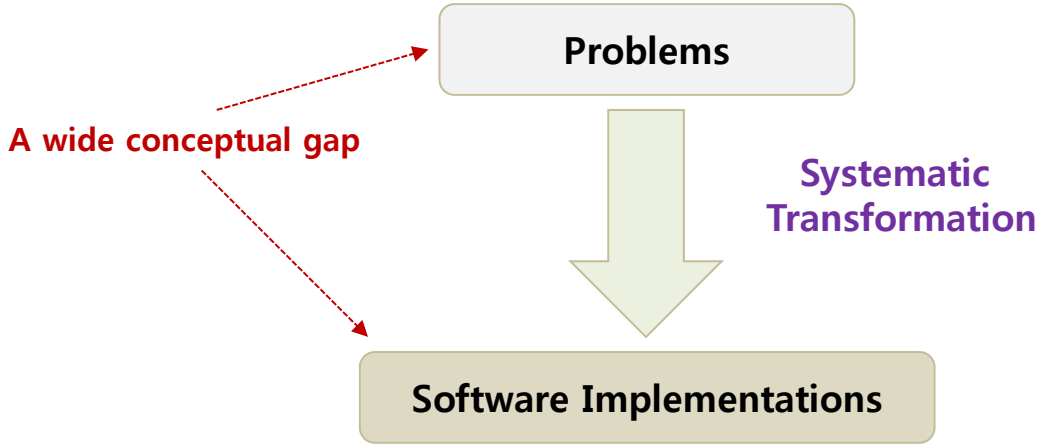
Junbeom Yoo, Eui-Sub Kim, Dong Ah Lee
KONKUK University
and
Jong-Gyun Choi
Korea Atomic Energy Research Institute (KAERI)

Model-Based Development for I&C Software

MBD(Model-Based Development)

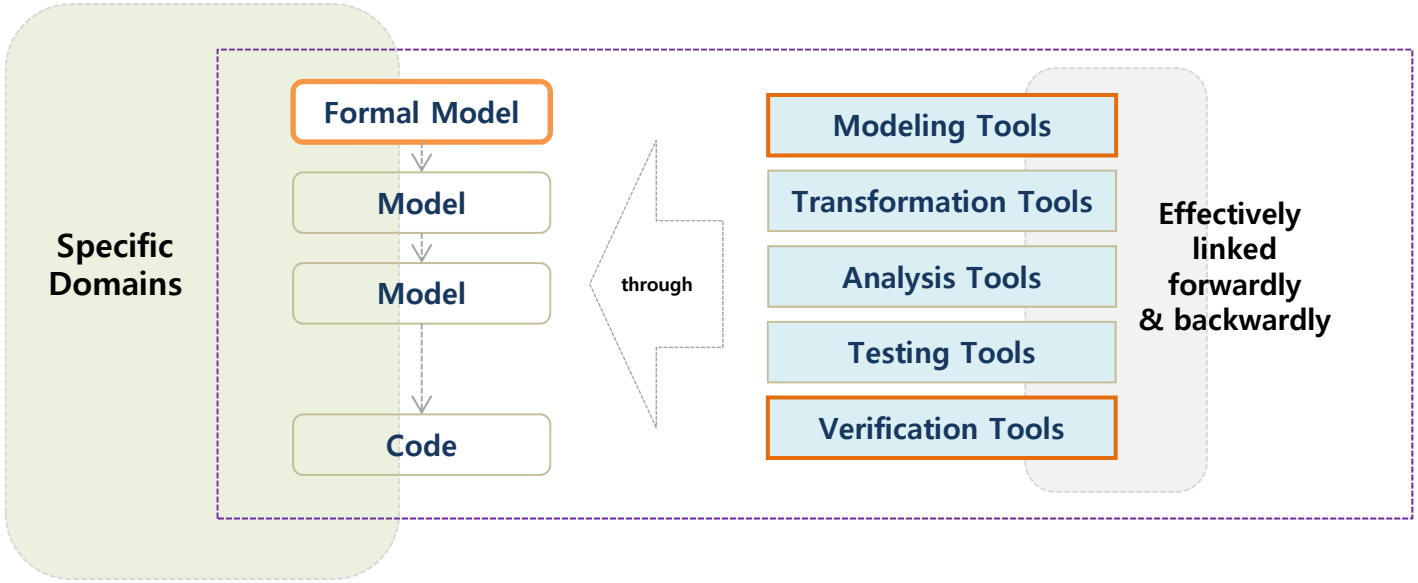
: Software development approaches in which abstract models of software systems are created and systematically transformed to concrete implementations

- Reducing the gap between problem and software implementation domain
- Using technologies that support systematic transformation of problem-level abstractions to software implementations
- Using models that describe complex systems at multiple levels of abstraction through automated support for transforming and analyzing models



MBD for I&Cs

Highly recommended to systematically cope with standards and regulations on software safety

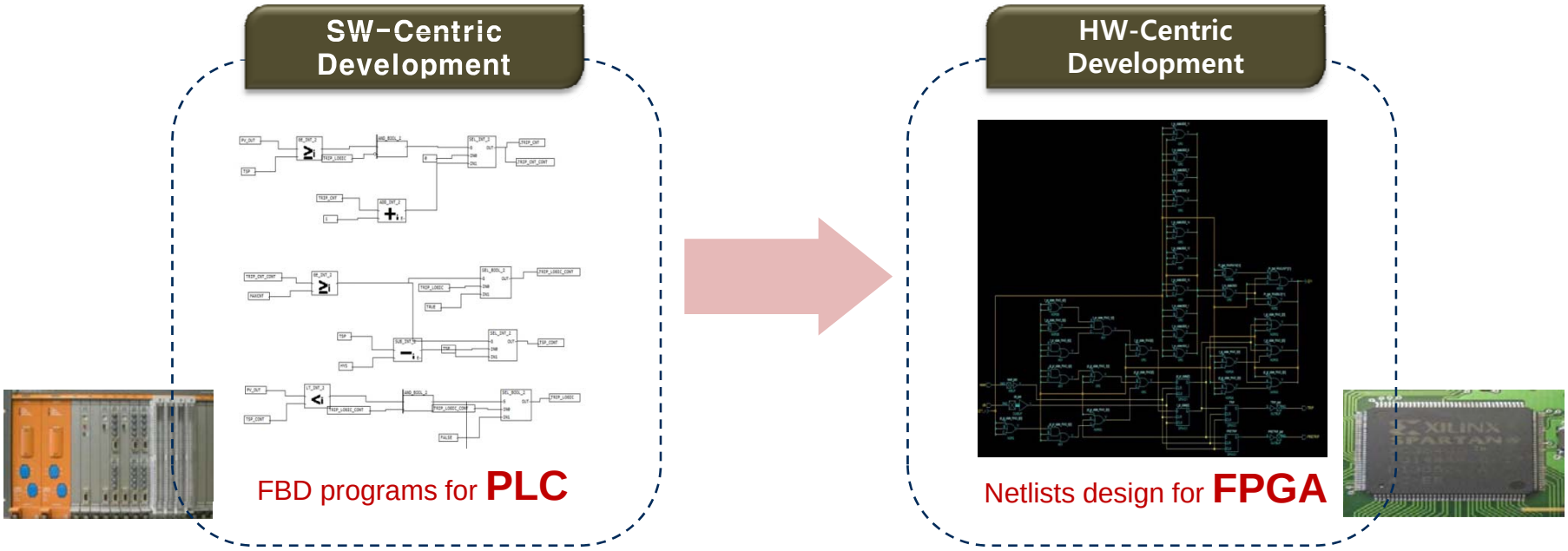


The Platform Change of Digital I&Cs from PLC to FPGA

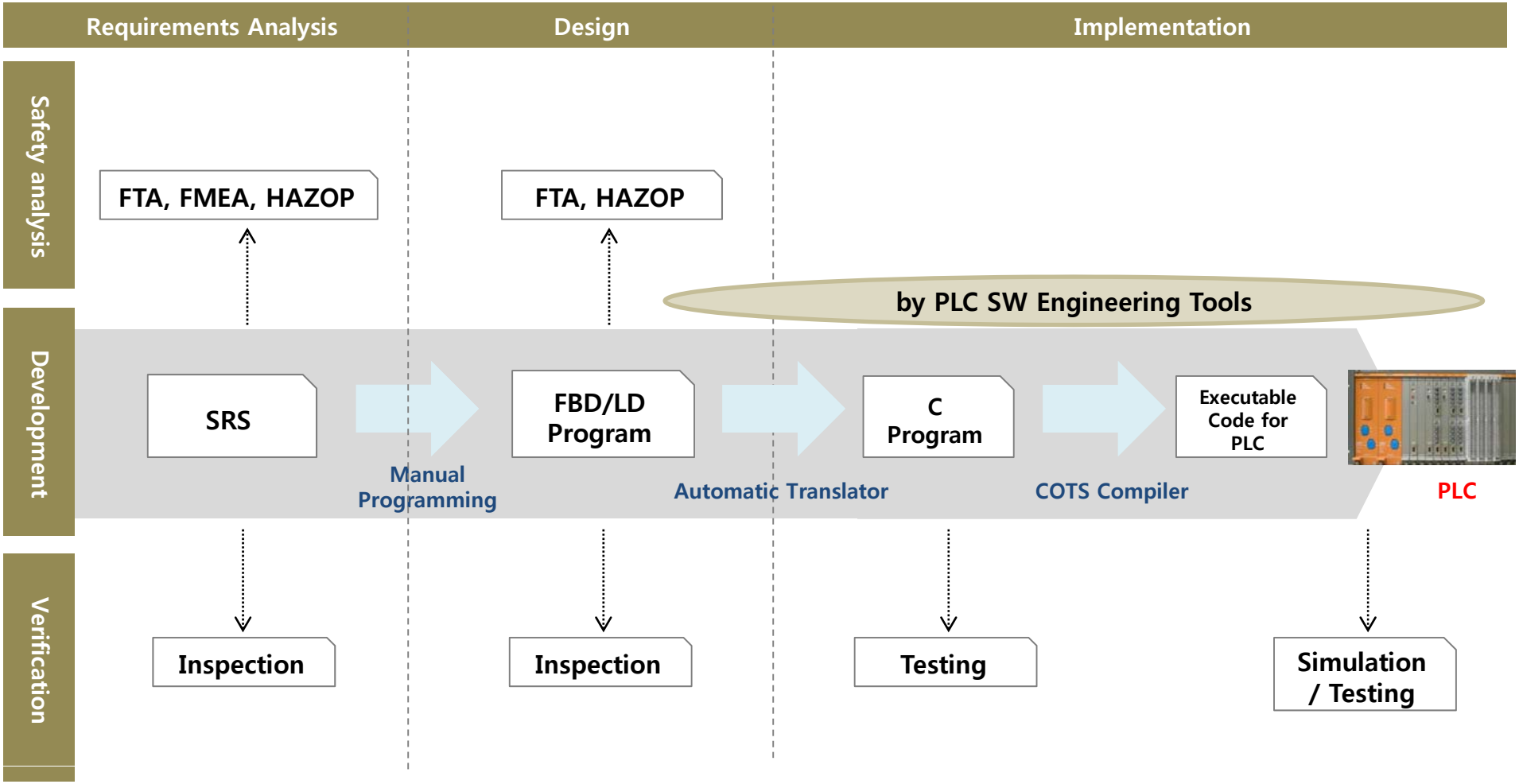
In order to reduce the maintenance cost of PLCs and use more computation power than PLCs

However, it is too risky!

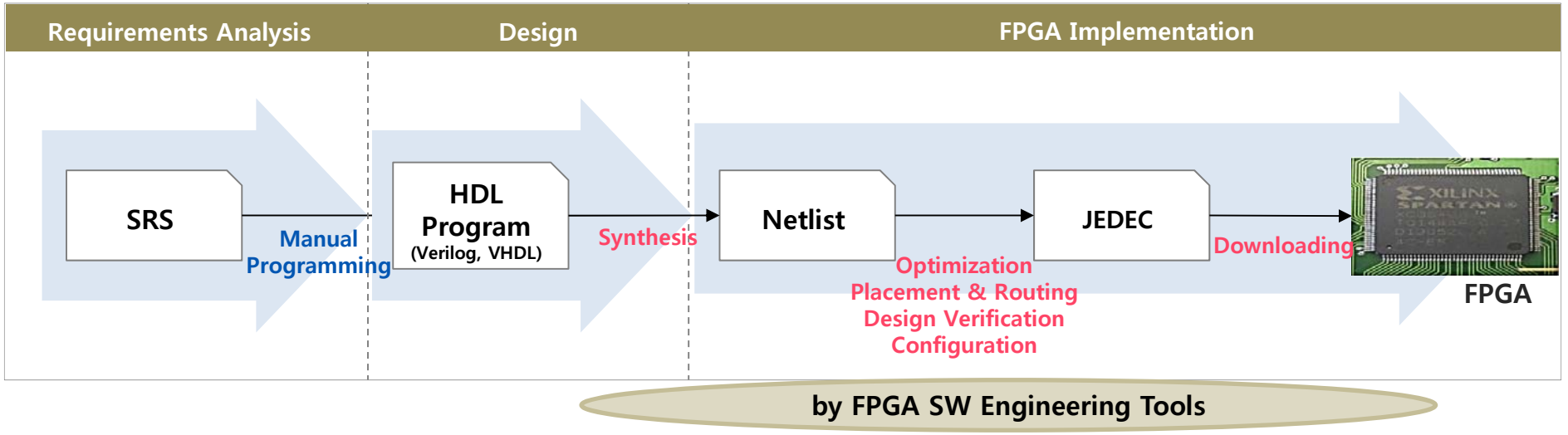
It is not the change of SW development methods, but that of development paradigms.
PLC → FPGA ≒ CPU-based Software → Net-based Hardware



Typical Software Development Process for PLC-based I&Cs



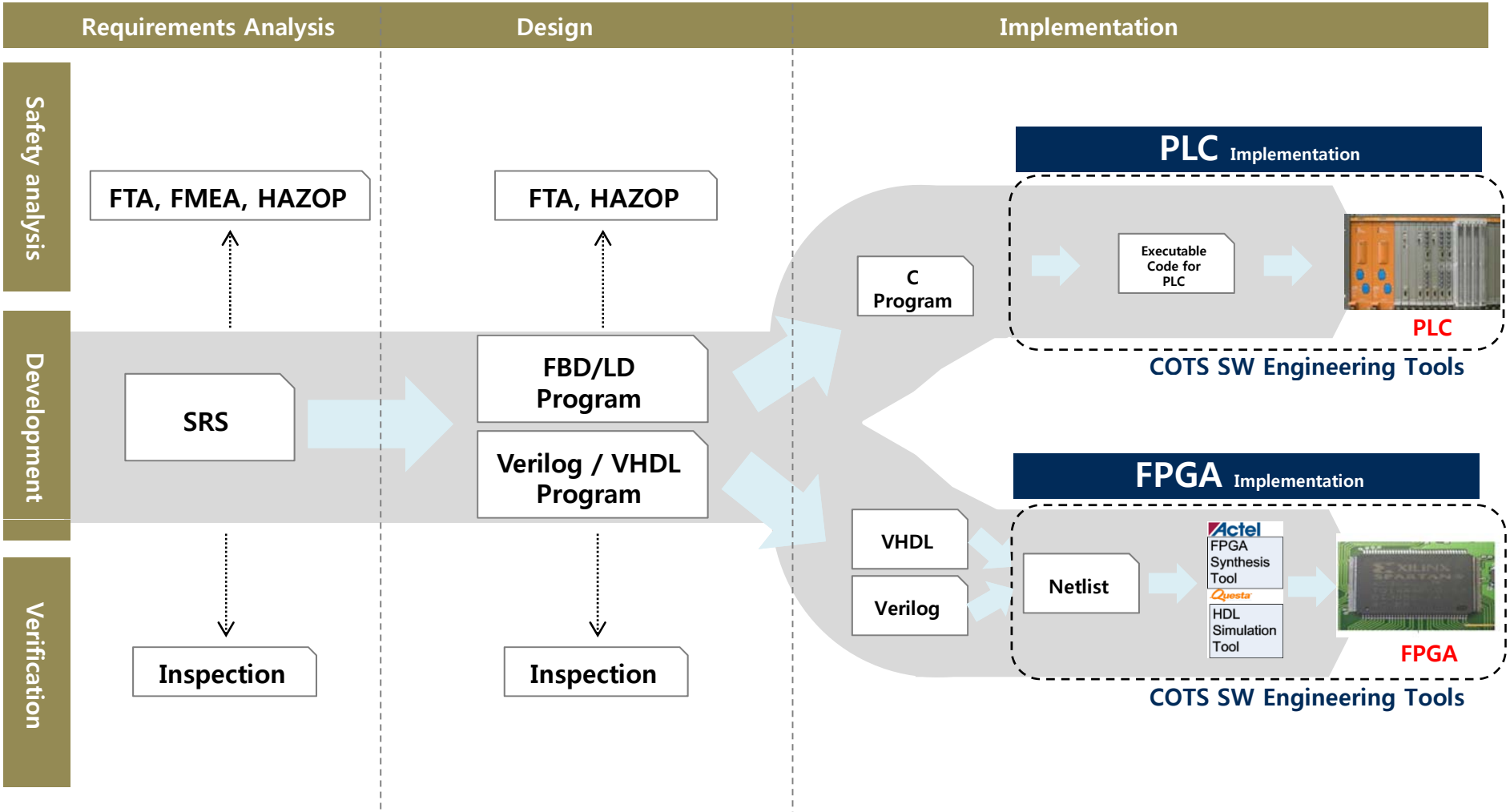
Typical Software Development Process for FPGA



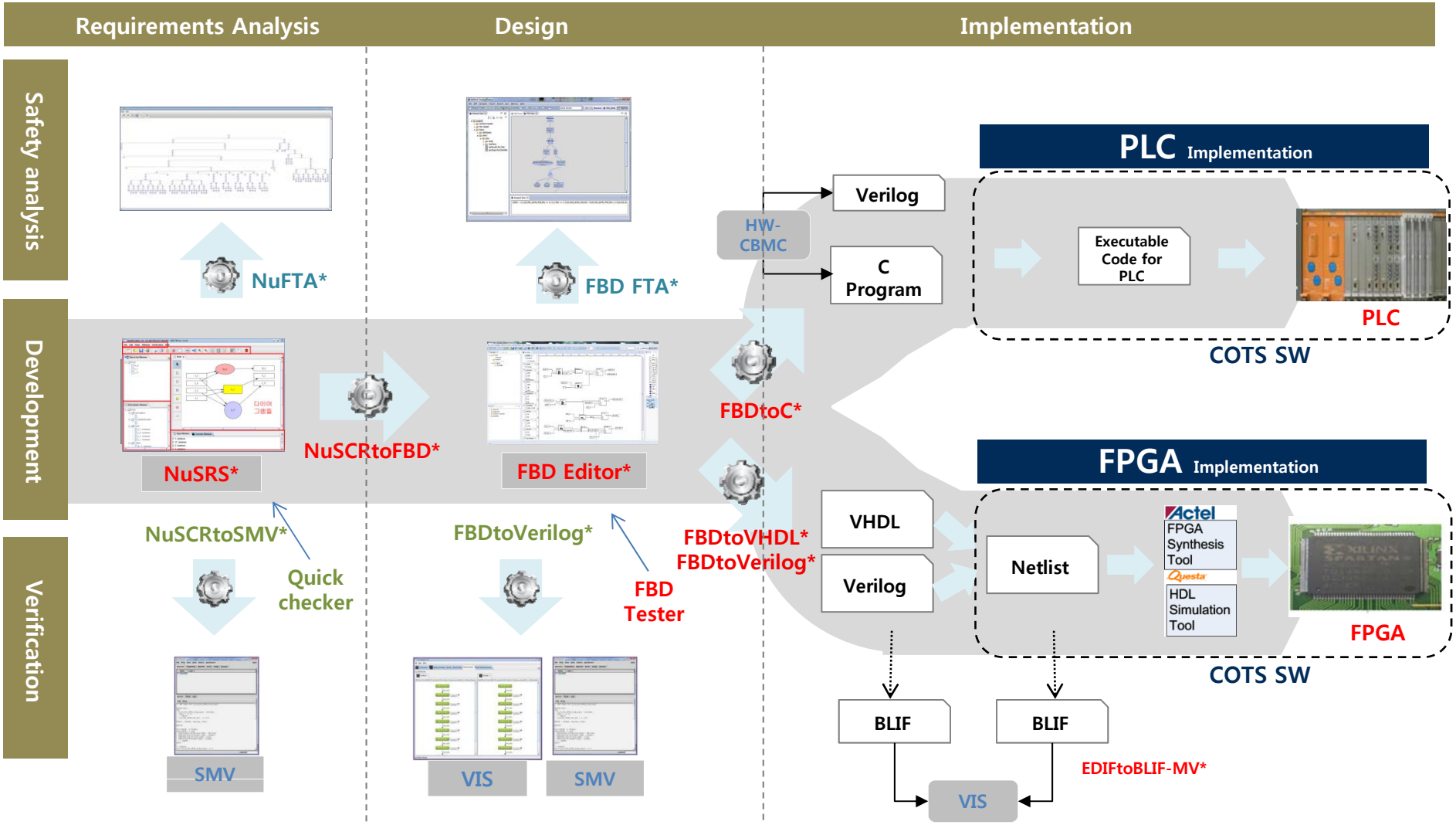
The parallel processes for safety analysis and verification as the PLC have not yet been defined for safe-level I&C Applications!!!

No commercial FPGA implementation for RPS or ESF-CCS, yet.

An Overlap of Two SW Development Processes

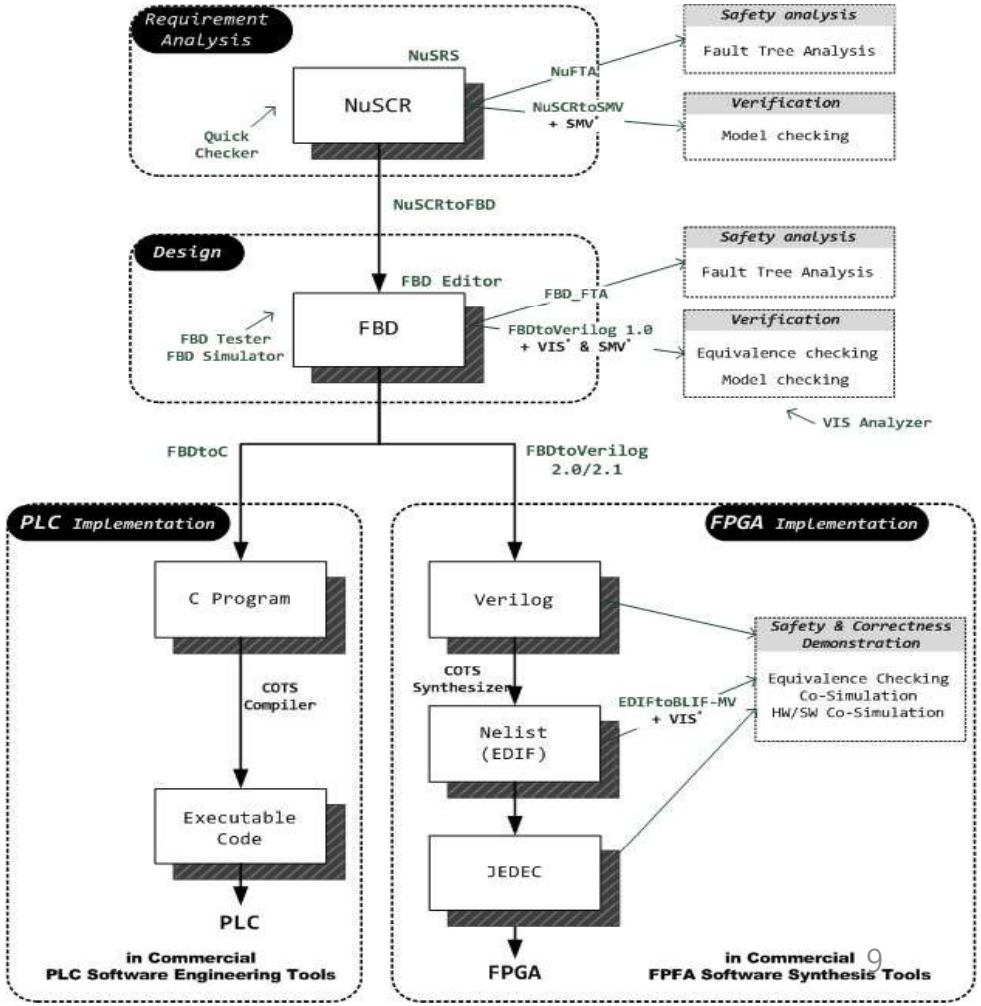


The Scope of NuDE 2.0



An Overview of NuDE 2.0

- NuDE (**Nu**clear **D**evelopment **E**nvironment) 2.0
  - An formal methods-based MBD for Digital I&C software
    - Target platforms: PLC & FPGA
  - Starting from a formal SRS in NuSCR
  - Supporting various V&V methods
    - Model Checking
    - Equivalence Checking
  - Considering safety demonstration of commercial SW synthesis tools
  - From an SRS in NuSCR or SDS in FBD, the PLC and FPGA implementations can be generated simultaneously.



The NuDE Components

Development

- + NuSCR Editor
- + NuSCRtoFBD
- + FBDtoC
- + FBD Tester
- + FBD Editor
- + FBDtoVerilog 2.0/2.1
- + FBDtoVHDL

→ PLC SW Development

→ FPGA SW Development

Verification

- + Quick Checker
- + NuSCRtoSMV + SMV
- + FBDtoVerilog 1.0 + VIS & SMV
- + VIS Analyzer
- + FBDtoVerilog 1.2 + HW-CBMC
- + EDIFtoBLIF-MV + VIS

→ PLC SW Verification

→ FPGA SW Verification

Safety Analysis

- + NuSCR_FTA
- + FBD_FTA

→ PLC SW Safety Analysis

Supplementary Tools

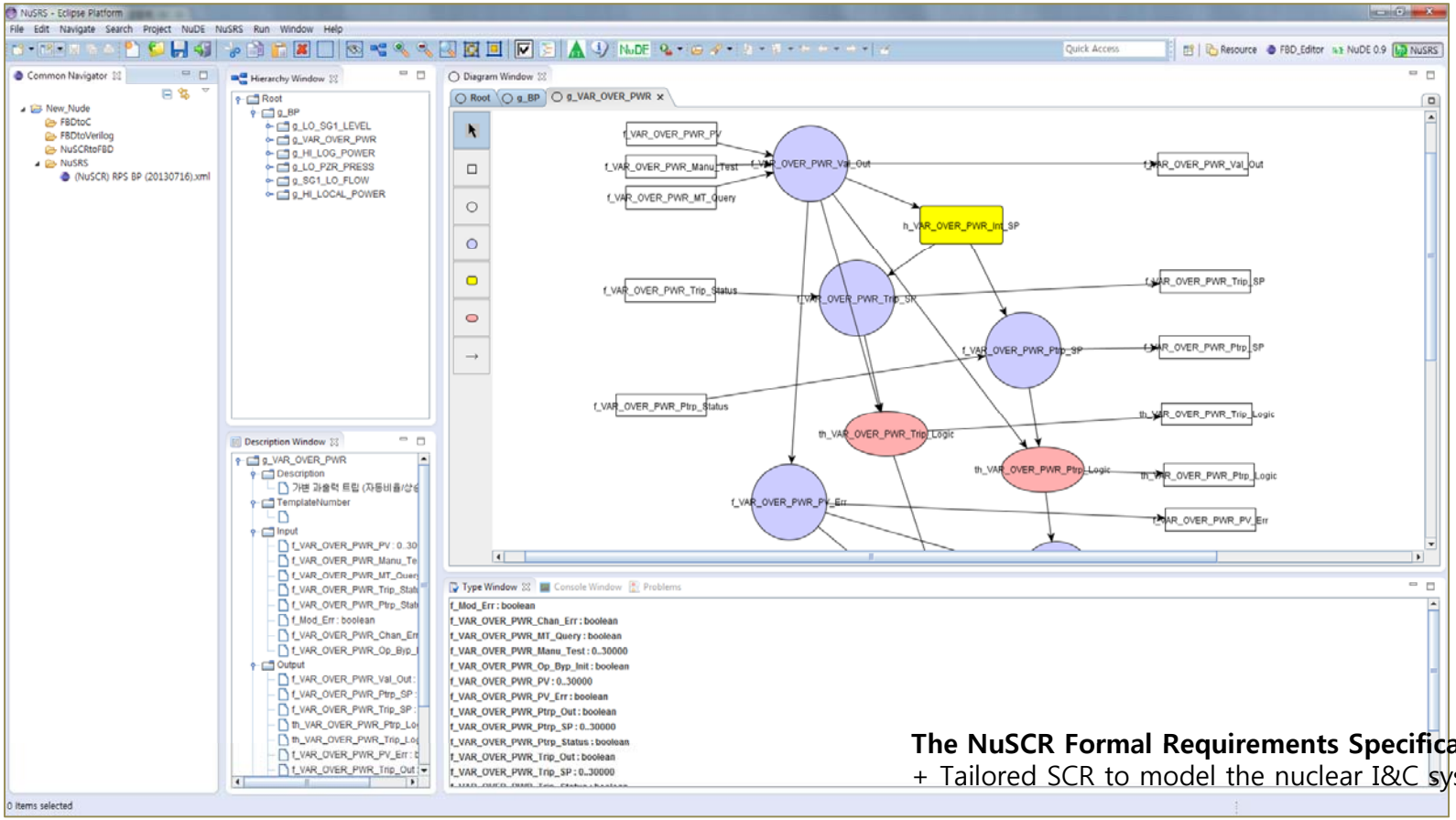
- + Scenario Generator
- + FBD Simulator
- + FBD-Verilog Comparator
- + C Simulator
- + FBD-C Comparator
- + FBD-Verilog-Netlist-JEDEC Comparator

The scope of this paper!

KU KONKUK UNIVERSITY

DEVELOPMENT PROCESS

NuSCR Editor (NuSRS 2.0)



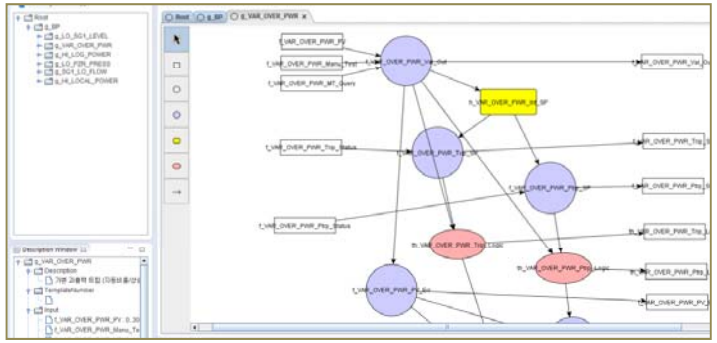
The NuSCR Formal Requirements Specification

+ Tailored SCR to model the nuclear I&C systems efficiently

The NuSCR Elements

+ FOD (Function Overview Diagram)
+ SDT (Structured Decision Table)
+ FSM (Finite State Machine)
+ TTS (Timed Transition System)

NuSCRtoFBD (Ver. 3.0)



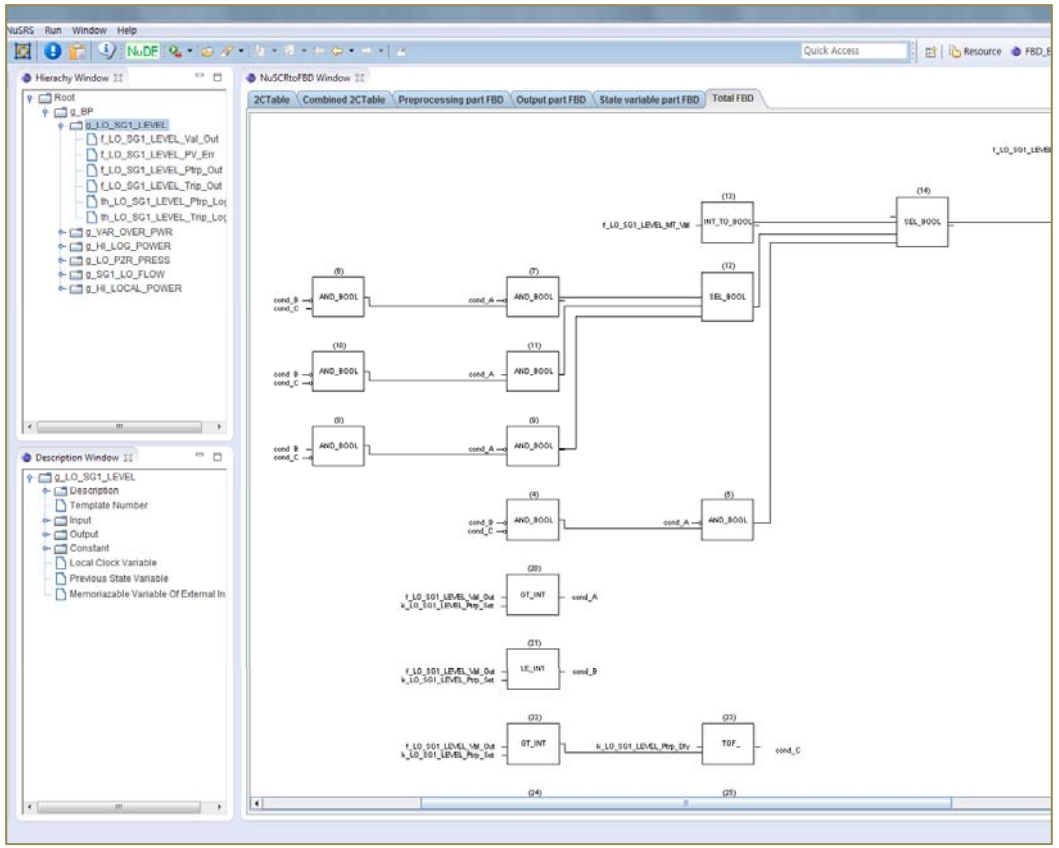
The NuSCR Formal SRS

Mechanical Transformation

The NuSCRtoFBD Mechanical Transformation

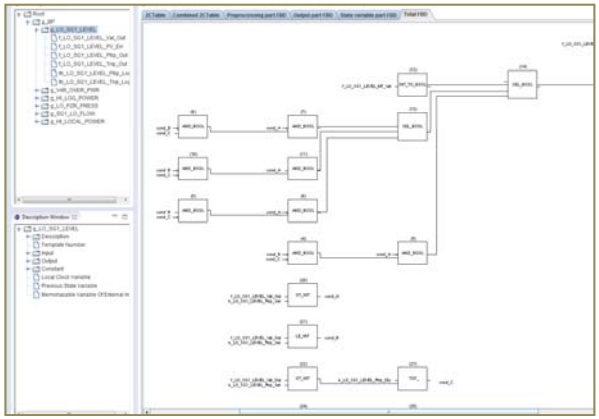
+ Transforms the NuSCR formal SRS
+ into behaviorally-equivalent FBD programs
+ mechanically

+ Store into an XML file of PLCopen Std.



FBD Program

FBDtoC (Ver. 1.0)



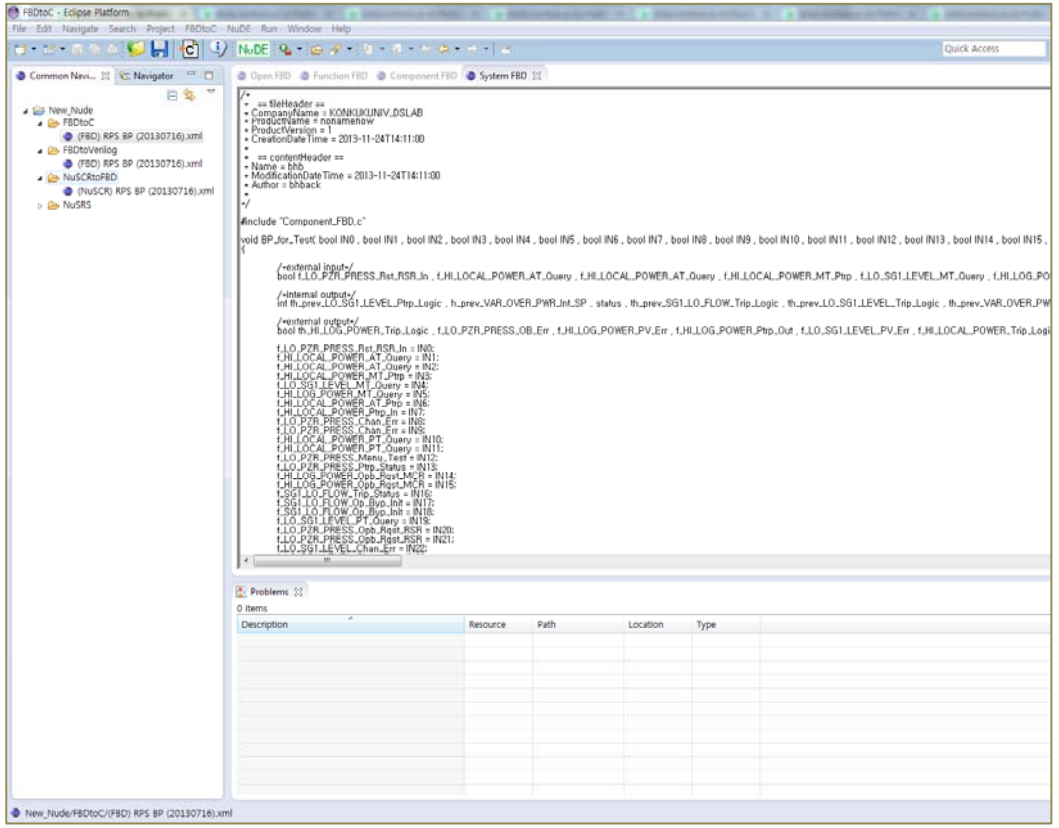
FBD Program

Mechanical Transformation

The FBDtoC Mechanical Transformation

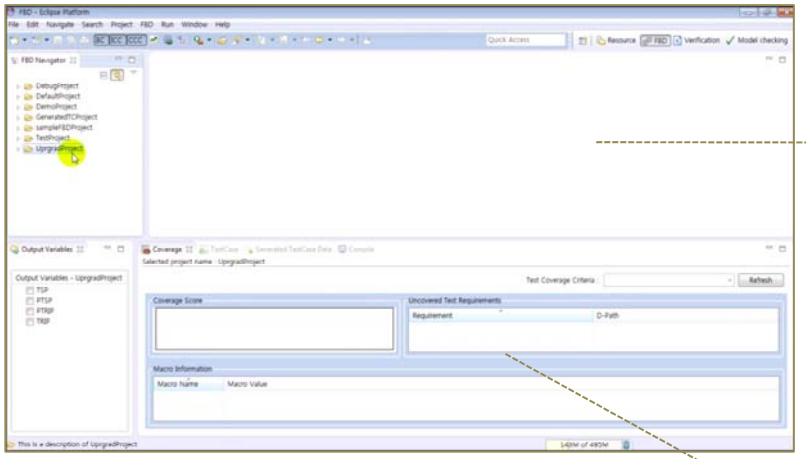
+ Transforms FBD programs
+ into behaviorally-equivalent ANSI-C programs
+ mechanically

+ System/Component/Function units



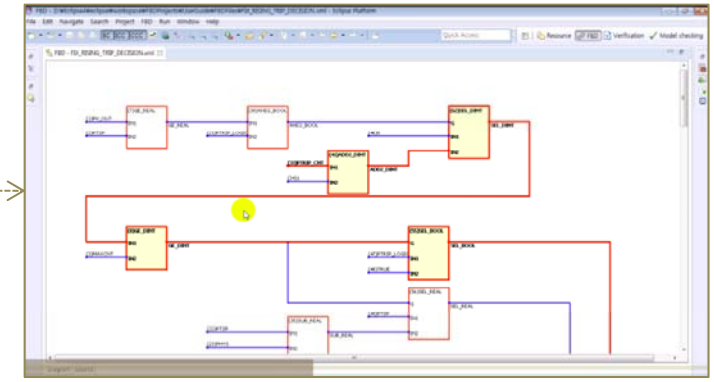
C Program

FBD Tester (of Prof. Gee in KAIST)

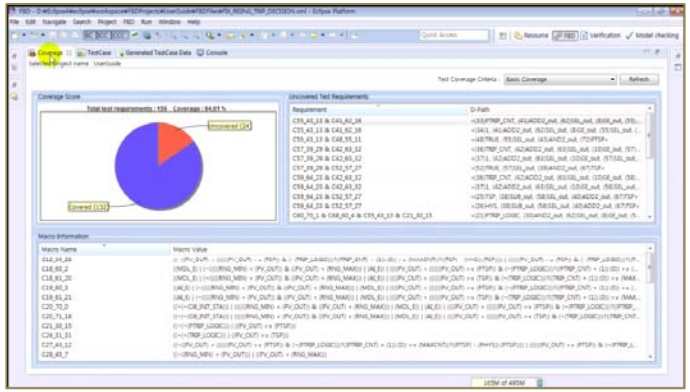


FBD Tester

FBD Test Execution



Coverage View



An Unit Testing Tool for FBD Programs

+ Data-Flow Testing
+ Provides 3 Data-Flow based Coverage Criteria

+ Automatic generation of test cases
+ Test execution (Model-based Test/Simulation)
+ Calculation of testing coverage

+ Reads an FBD program in an XML file of PLCopen Std.

Coverage | TestCase | Generated TestCase Data | Console

elected project name : UserGuide

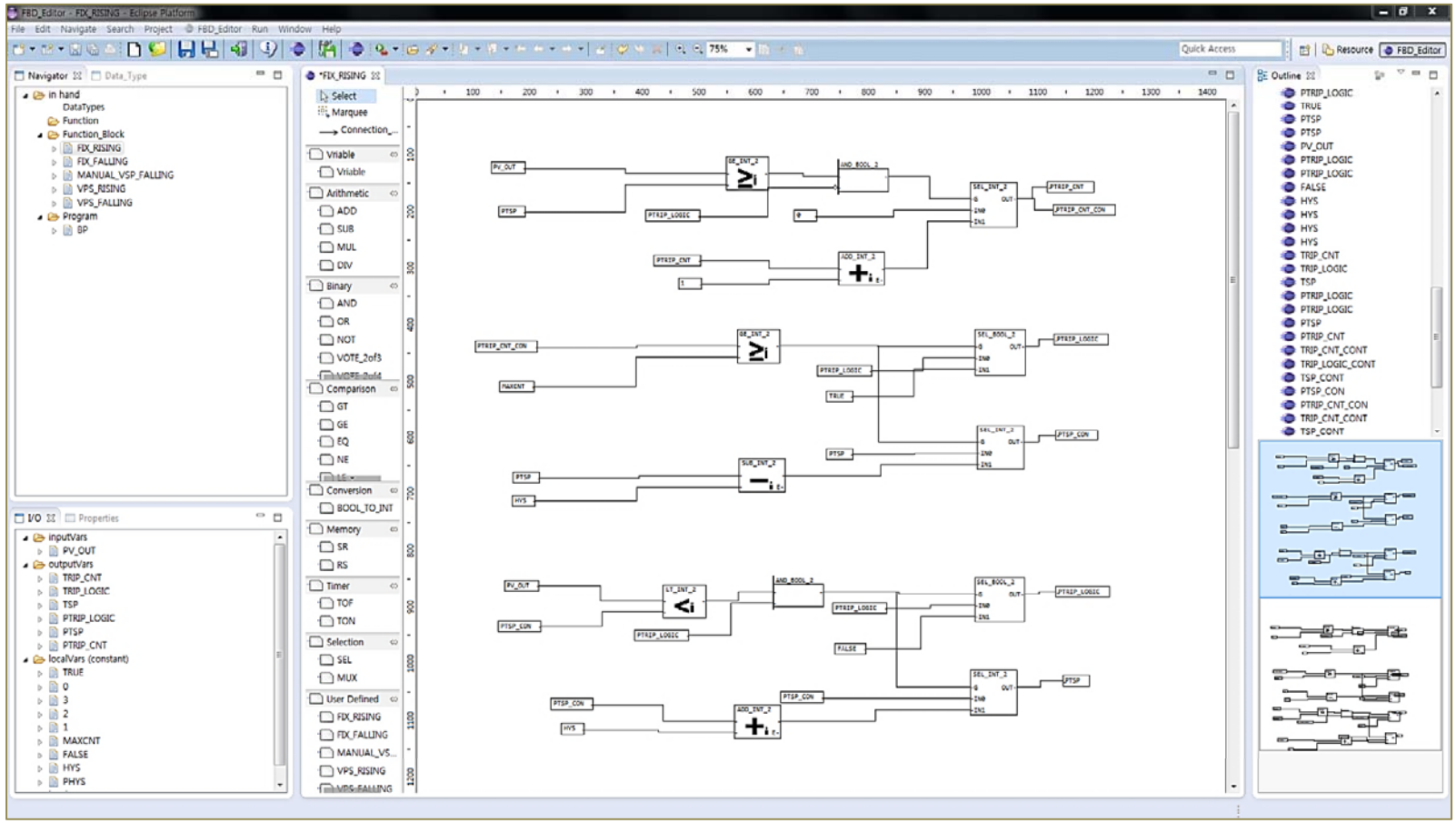
Test case information

PV_OUT	PTSP	TSP	PTRIP_LOG...	TRIP_LOGIC	PTRIP_CNT	TRIP_CNT	AI_E	MDL_E
-301	300	-300	true	true	301	303	false	false
29400	29400	29400	false	false	0	-2	false	false
29400	29401	29401	false	false	29402	29404	false	false

Test Case Generation

DEPENDABLE SOFTWARE LABORATORY

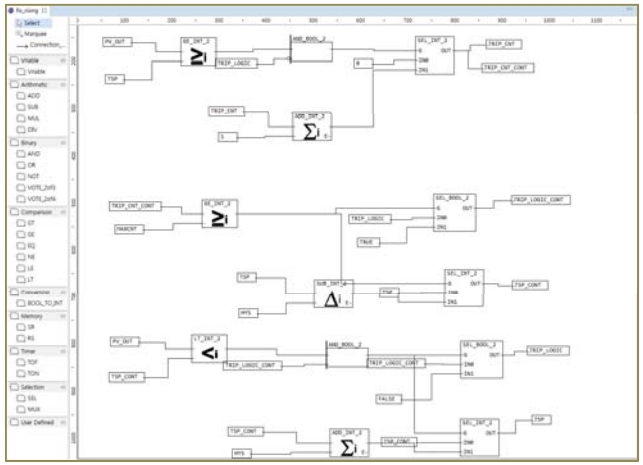
FBD Editor



The FBD Program Editor

+ Programming FBD programs of IEC 61131-1 Std.
+ Reads and stores an XML file of PLCopen Std.

FBDtoVerilog 2.0/2.1

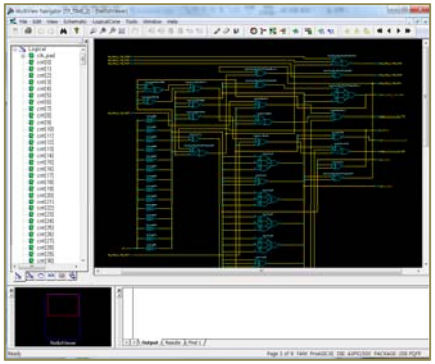


FBD Program

Mechanical Transformation

The FBDtoVerilog Mechanical Transformation
+ Transforms FBD programs
+ into behaviorally-equivalent Verilog programs
+ mechanically

+ Used for the FPGA Synthesis

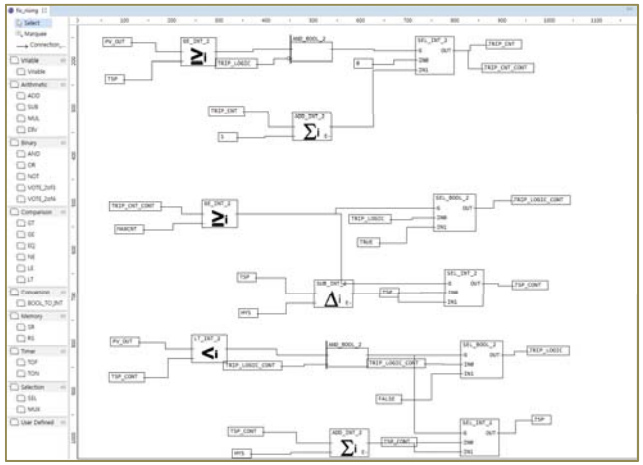


Microsemi - Libero Netlist View

```
module fix_rising (rst, clk, PV_OUT, TRIP_CNT, TRIP_LOGIC, TSP);

    input clk;
    input rst;

    input   [31:0] PV_OUT;
    output  [31:0] TRIP_CNT;    reg     [31:0] TRIP_CNT;
    output   TRIP_LOGIC;    reg      TRIP_LOGIC;
    output  [31:0] TSP;     reg     [31:0] TSP;
    parameter    TRUE = 1;
    parameter    false = 0;
    parameter   [31:0] MAXCNT = 30;
    reg     [31:0] HYS = 300;


    wire     GE_INT_2_wire_1_OUT;
    wire  AND_BOOL_2_wire_2_OUT;
    wire    [31:0] SEL_INT_2_wire_3_OUT;
    wire    [31:0] ADD_INT_2_wire_4_OUT;
    wire     ADD_INT_2_wire_4_E;
    wire     GE_INT_2_wire_14_OUT;
    wire     SEL_BOOL_2_wire_15_OUT;
    wire    [31:0] SEL_INT_2_wire_16_OUT;
    wire    [31:0] SUB_INT_2_wire_17_OUT;
    wire     SUB_INT_2_wire_17_E;
    wire     LT_INT_2_wire_27_OUT;

    GE_INT_2 GE_INT_2_1(rst, clk, PV_OUT, TSP, GE_INT_2_wire_1_OUT);
    AND_BOOL_2 AND_BOOL_2_2(rst, clk, GE_INT_2_wire_1_OUT, ~TRIP_LOGIC, AND_BOOL_2_wire_2_
    SEL_INT_2 SEL_INT_2_3(rst, clk, AND_BOOL_2_wire_2_OUT, 0, ADD_INT_2_wire_4_OUT, SEL_IN
    ADD_INT_2 ADD_INT_2_4(rst, clk, TRIP_CNT, 1, ADD_INT_2_wire_4_OUT, ADD_INT_2_wire_4_E)
    GE_INT_2 GE_INT_2_14(rst, clk, TRIP_CNT_CONT, MAXCNT, GE_INT_2_wire_14_OUT);
    SEL_BOOL_2 SEL_BOOL_2_15(rst, clk, GE_INT_2_wire_14_OUT, TRIP_LOGIC, TRUE, SEL_BOOL_2_
    SEL_INT_2 SEL_INT_2_16(rst, clk, GE_INT_2_wire_14_OUT, TSP, SUB_INT_2_wire_17_OUT, SEL
    SUB_INT_2 SUB_INT_2_17(rst, clk, TSP, HYS, SUB_INT_2_wire_17_OUT, SUB_INT_2_wire_17_E)
    LT_INT_2 LT_INT_2_27(rst, clk, PV_OUT, TSP_CONT, LT_INT_2_wire_27_OUT);
    AND_BOOL_2 AND_BOOL_2_28(rst, clk, LT_INT_2_wire_27_OUT, TRIP_LOGIC_CONT, AND_BOOL_2_w
    SEL_INT_2 SEL_INT_2_29(rst, clk, AND_BOOL_2_wire_28_OUT, TSP_CONT, ADD_INT_2_wire_31_O
    SEL_BOOL_2 SEL_BOOL_2_30(rst, clk, AND_BOOL_2_wire_28_OUT, TRIP_LOGIC_CONT, FALSE, SEL
    ADD_INT_2 ADD_INT_2_31(rst, clk, TSP_CONT, HYS, ADD_INT_2_wire_31_OUT, ADD_INT_2_wire_

    assign TRIP_CNT_CONT = SEL_INT_2_wire_3_OUT;
    assign TRIP_LOGIC_CONT = SEL_BOOL_2_wire_15_OUT;
    assign TSP_CONT = SEL_INT_2_wire_16_OUT;

    always @(posedge rst or posedge clk)
    begin
        if(rst) begin
            TRIP_CNT <=     16'b0000000000000000;
            TRIP_LOGIC <=   1'b0;
            TSP <= 26805;
        end else if (clk) begin
            TRIP_CNT <= SEL_INT_2_wire_3_OUT;
            TRIP_LOGIC <= SEL_BOOL_2_wire_30_OUT;
            TSP <= SEL_INT_2_wire_29_OUT;
        end
    end
```

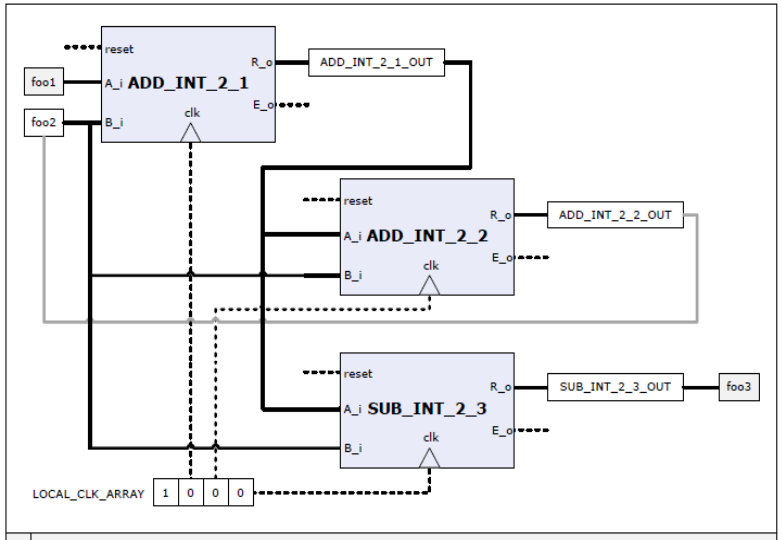
Verilog Program

FBDtoVHDL



FBD 프로그램

Mechanical Transformation




```
1  entity bar is
2  port {
3          c : in std_logic; r : in std_logic; scan : in std_logic;
4          foo1 : in integer range INT_LO to INT_HI;
5          foo3 : out integer range INT_LO to INT_HI
6  };
7  end bar;
8
9  architecture Behavioral of sample is
10 signal LOCAL_CLK_ARRAY : std_logic_vector(0 to 3);
11 signal foo2 : integer range INT_LO to INT_HI;
12 signal ADD_INT_2_1_OUT : integer range INT_LO to INT_HI;
13 signal ADD_INT_2_2_OUT : integer range INT_LO to INT_HI;
14 signal ADD_INT_2_3_OUT : integer range INT_LO to INT_HI;
15
16 begin
17 ADD_INT_2_1: ADD_INT_2 port map (clk => LOCAL_CLK_ARRAY(1), reset => r,
                                    A_i => foo1, B_i => foo2, R_i => ADD_INT_2_1_OUT);
18 ADD_INT_2_2: ADD_INT_2 port map (clk => LOCAL_CLK_ARRAY(2), reset => r,
                                    A_i => ADD_INT_2_1_OUT, B_i => foo2, R_i => ADD_INT_2_2_OUT);
19 ADD_INT_2_3: ADD_INT_2 port map (clk => LOCAL_CLK_ARRAY(3), reset => r,
20                                  A_i => ADD_INT_2_1_OUT, B_i => foo2, R_i => ADD_INT_2_3_OUT);
21 process(c, r) begin
22         if (r='1') then
23                 LOCAL_CLK_ARRAY <= "1000";
24                 foo1 <=  ;          --Initial value should be presented manually.
25                 foo2 <=  ;          --Initial value should be presented manually.
26         elsif risig_edge(c) then
27                 LOCAL_CLK_ARRAY <= '0' & LOCAL_CLK_ARRAY(0 to 2);
28                 if (scan = '1') then
29                         foo2 <= ADD_INT_2_2_OUT;
30                         foo3 <= SUB_INT_2_3_OUT;
31                         LOCAL_CLK_ARRAY <= "1000";
32                 end if;
33         end if;
34 end process;
35 end Behavioral;
```

VHDL program

The FBDtoVHDL Mechanical Transformation

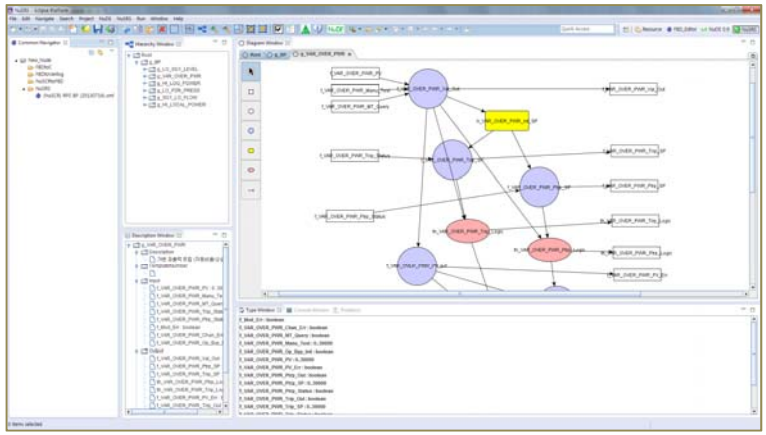
+ Transforms FBD programs
+ into behaviorally-equivalent VHDL programs
+ mechanically

+ Used for the FPGA Synthesis

KU KONKUK UNIVERSITY

VERIFICATION PROCESS

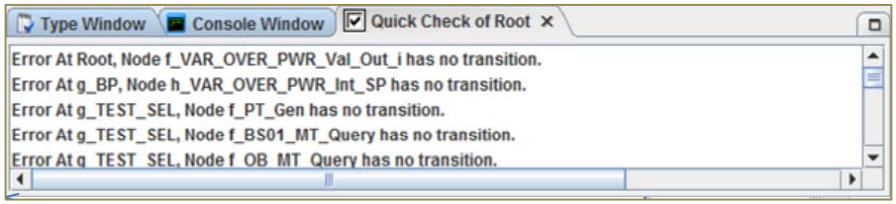
Quick Checker



The NuSCR Formal SRS

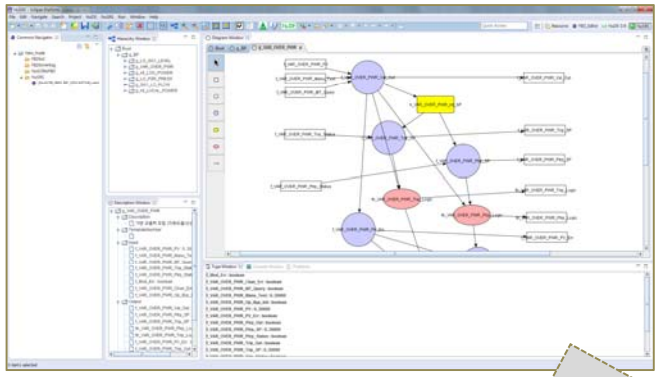
Static Analysis

Static Analysis (Rule Checking) on the NuSCR formal SRS
+ Checking for the C&C(Completeness & Consistency) requirements



A result of Quick Checker

NuSCRtoSMV + SMV



The NuSCR Formal SRS

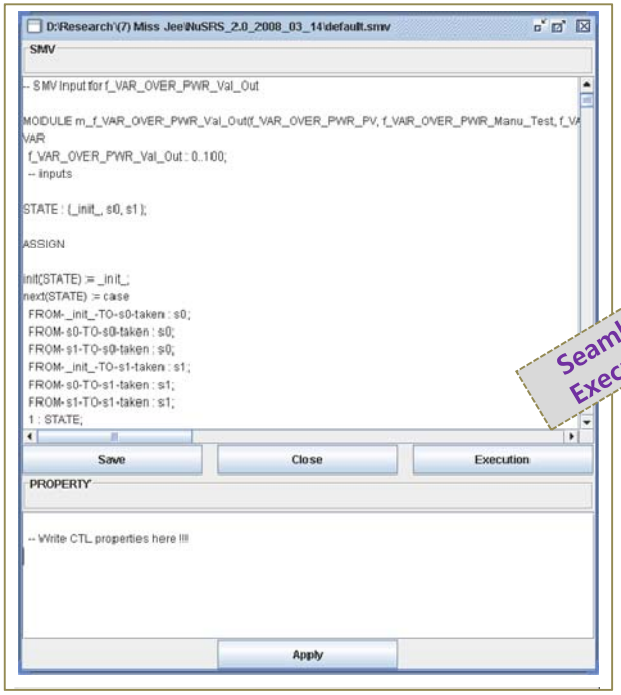
Mechanical Transformation

The NuSCRtoSMV Mechanical Transformation

+ Transforms NuSCR a formal specification
+ into a behaviorally-equivalent SMV program
+ mechanically

+ Requires to input verification properties

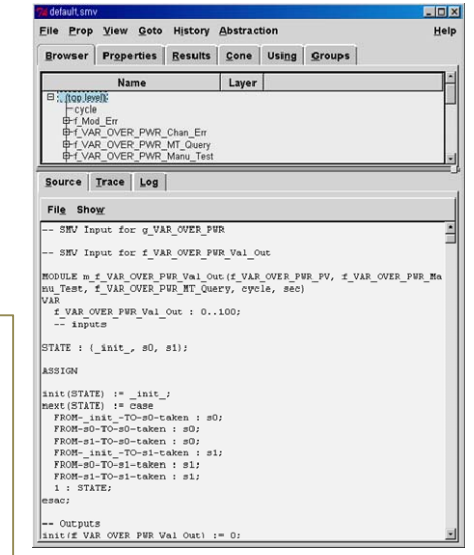
+ Executes the SMV model checker seamlessly



SMV Input Program

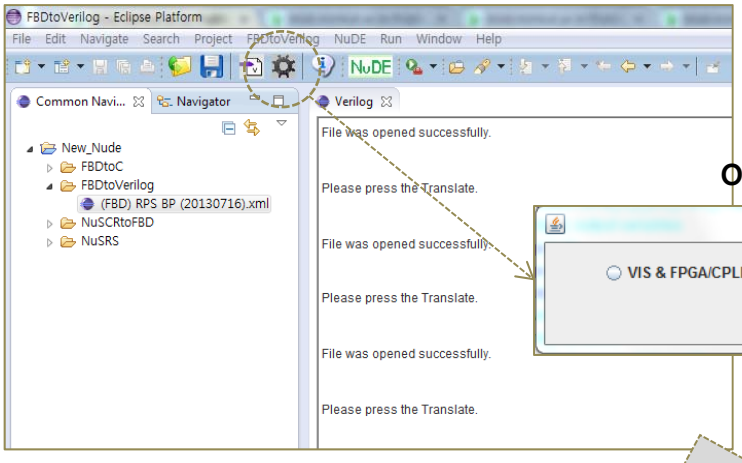
Seamless Execution

SMV Model Checker



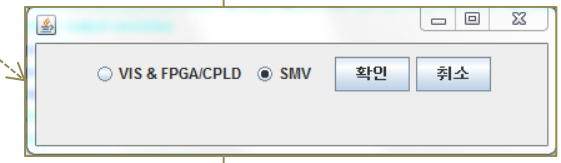
Formal Verification

FBDtoVerilog 1.0 + VIS & SMV

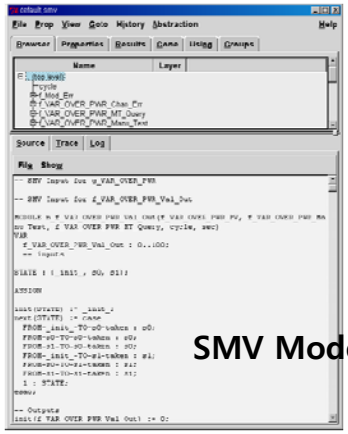


An FBD Program in the FBD Editor

Option Selection

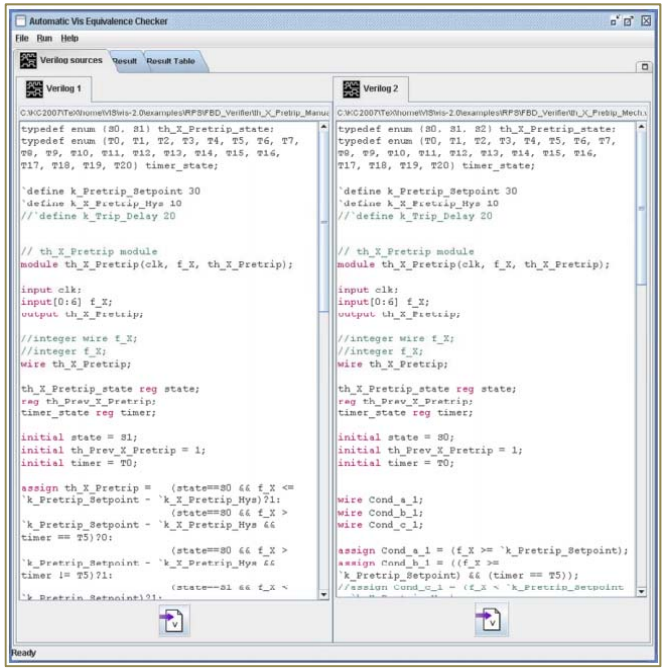


Mechanical Transformation



SMV Model Checking

Mechanical Transformation



VIS Analyzer

VIS Equivalence Checking

The FBDtoVerilog Mechanical Transformation

+ Transforms an FBD program
+ into a behaviorally-equivalent Verilog program
+ mechanically

+ Options for VIS and SMV

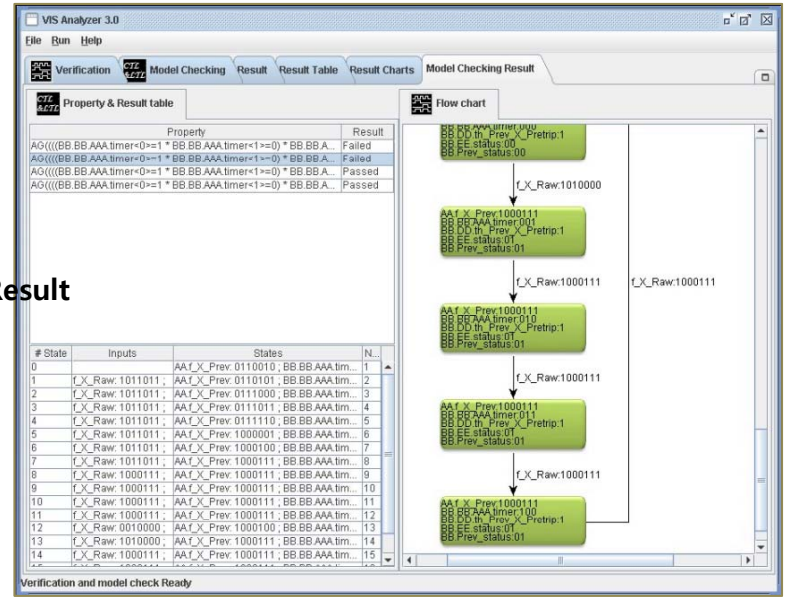
+ SMV : Model Checking
+ VIS : Equivalence Checking

VIS Analyzer (Ver. 3.0)

```
vis release 2.0 (compiled Thu Jun 26 11:08:16    2008)
vis> read_blif_mv h_X_Pretrip_Manual.mv
vis> flatten_hierarchy
vis> static_order
vis> build_partition_mdds
vis> simulate -i inputVector1.txt
# vis release 2.0 (compiled Thu Jun 26 11:08:16    2008)
# Network: main
# Input Vectors File: inputVector1.txt


.inputs  f_X_Raw<0> f_X_Raw<1> f_X_Raw<2> f_X_Raw<3> f_X_Raw<4>
f_X_Raw<5> f_X_Raw<6> temp
.latches AA.Prev_th_Reset_Ini AA.state AA.timer BB.Prev_Pk_State
BB.f_X_Prev_PTSP<0> BB.f_X_Prev_PTSP<1> BB.f_X_Prev_PTSP<2>
BB.f_X_Prev_PTSP<3> BB.f_X_Prev_PTSP<4> BB.f_X_Prev_PTSP<5>
BB.f_X_Prev_PTSP<6> BB.f_X_t0<0> BB.f_X_t0<1> BB.f_X_t0<2>
BB.f_X_t0<3> BB.f_X_t0<4> BB.f_X_t0<5> BB.f_X_t0<6> CC.f_X_Prev<0>
CC.f_X_Prev<1> CC.f_X_Prev<2> CC.f_X_Prev<3> CC.f_X_Prev<4>
CC.f_X_Prev<5> CC.f_X_Prev<6> DD.state DD.th_Prev_X_Pretrip DD.timer
.outputs th_X_Pretrip
.initial 0 A0 T5 S1 0 0 1 0 0 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 S1 1 T0

.start_vectors

# f_X_Raw<0> f_X_Raw<1> f_X_Raw<2> f_X_Raw<3> f_X_Raw<4>
f_X_Raw<5> f_X_Raw<6> temp ; AA.Prev_th_Reset_Ini AA.state AA.timer
BB.Prev_Pk_State BB.f_X_Prev_PTSP<0> BB.f_X_Prev_PTSP<1>
BB.f_X_Prev_PTSP<2> BB.f_X_Prev_PTSP<3> BB.f_X_Prev_PTSP<4>
BB.f_X_Prev_PTSP<5> BB.f_X_Prev_PTSP<6> BB.f_X_t0<0> BB.f_X_t0<1>
BB.f_X_t0<2> BB.f_X_t0<3> BB.f_X_t0<4> BB.f_X_t0<5> BB.f_X_t0<6>
CC.f_X_Prev<0> CC.f_X_Prev<1> CC.f_X_Prev<2> CC.f_X_Prev<3>
CC.f_X_Prev<4> CC.f_X_Prev<5> CC.f_X_Prev<6> DD.state
DD.th_Prev_X_Pretrip DD.timer ; th_X_Pretrip

1 0 1 1 1 1 0 0 ; 0 A0 T5  S1 0 0 1 0 0 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 S1 1 T0  ; 1
1 0 1 1 1 1 0 1 ; 0 A0 T5  S1 0 0 1 0 0 1 1 1 0 1 0 1 1 1 1 0 1 0 1 1 1 1 S1 1 T1  ; 1
1 0 1 1 1 1 0 1 ; 1 A1 T0  S0 1 1 0 0 0 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 S1 1 T2  ; 1
1 0 1 1 1 1 0 1 ; 0 A0 T1  S4 1 1 0 0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1 1 S1 1 T3  ; 1
1 0 1 1 1 1 0 1 ; 0 A0 T2  S4 1 1 0 0 0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 1 S1 1 T4  ; 1
1 0 1 1 1 1 0 1 ; 0 A0 T3  S4 1 1 0 0 0 1 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 S1 1 T5  ; 0
1 0 1 1 1 1 0 1 ; 0 A0 T4  S4 1 1 0 0 0 1 1 1 1 0 1 0 1 1 1 1 0 1 0 1 1 1 S0 0 T5  ; 0
1 0 1 1 1 1 0 0 ; 0 A0 T5  S4 1 1 0 0 0 1 1 0 0 0 1 0 1 1 1 0 0 0 1 0 1 1 S0 0 T5  ; 0
1 0 1 1 1 1 0 0 ; 0 A0 T5  S4 1 1 0 0 0 1 1 1 0 1 0 0 1 1 1 1 0 1 0 0 1 1 S0 0 T5  ; 0
1 0 1 1 1 1 0 0 ; 0 A0 T5  S4 1 1 0 0 0 1 1 0 1 0 0 0 1 1 1 0 1 0 0 0 1 1 S0 0 T0  ; 0
1 0 1 1 1 1 0 0 ; 0 A0 T5  S4 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 S0 0 T0  ; 0
1 0 1 1 1 1 0 0 ; 0 A0 T5  S4 1 1 0 0 0 1 1 0 0 1 1 1 0 1 0 0 1 1 1 1 0 1 S0 0 T0  ; 1
# Final State : 0 A0 T5  S4 1 1 0 0 0 1 1 0 0 1 1 1 0 1 0 0 1 1 1 0 1 S0 0 T0
```

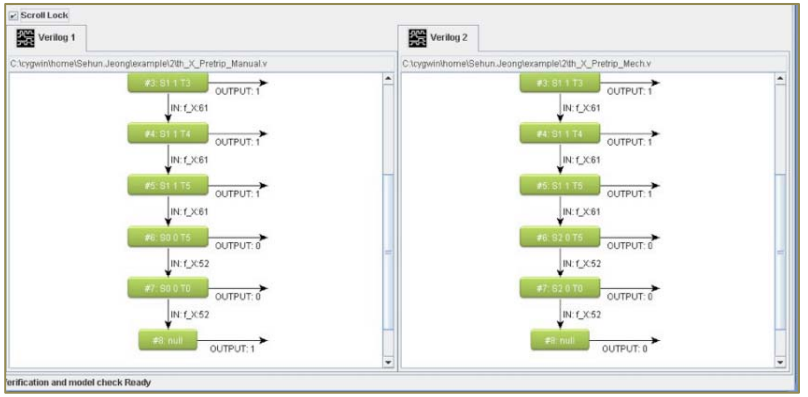
A Process of Executing VIS & the Execution Result

A Model Checking Result (Table + Chart)



Automatic Execution & Analysis

Flowchart



Table

Integer format Binary format

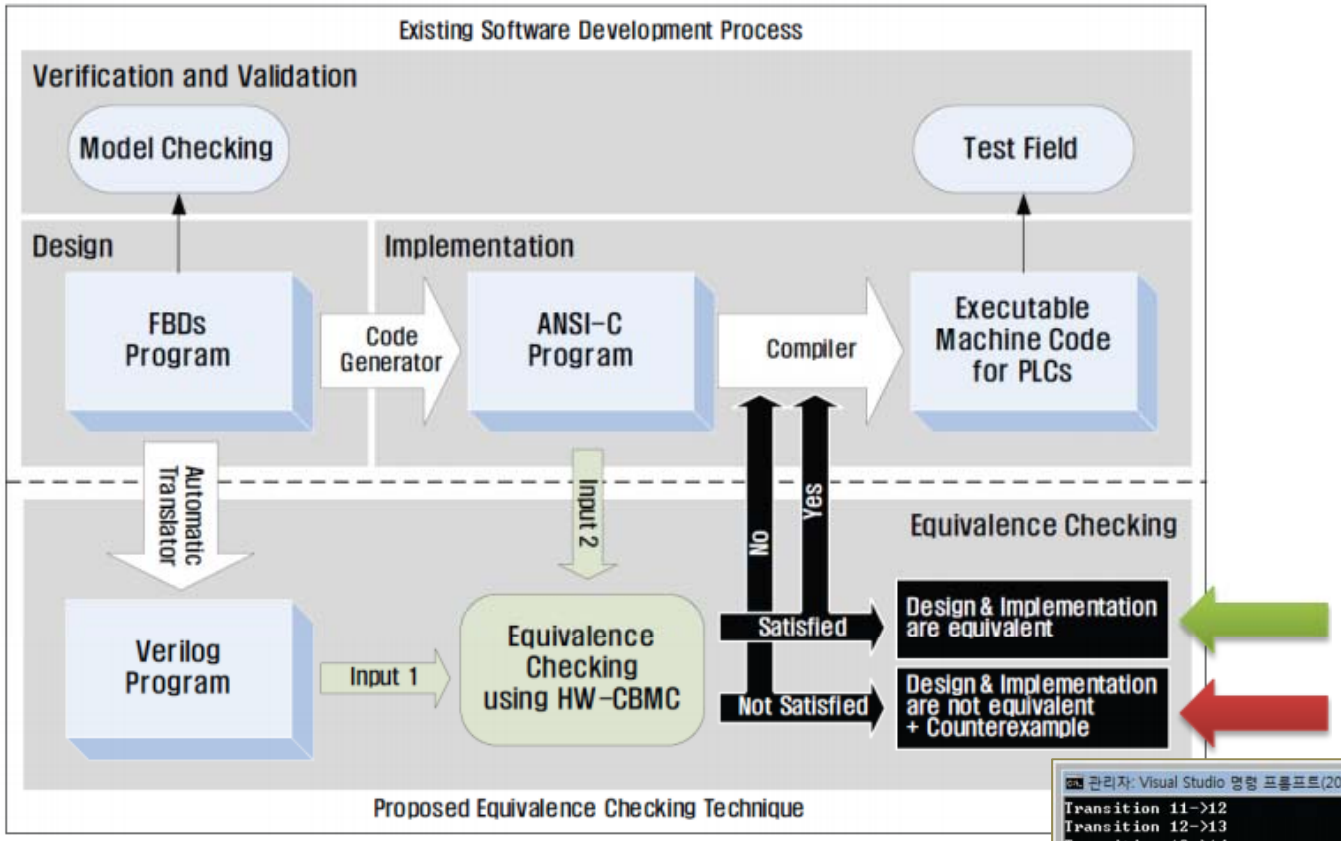
# state	input	File1Output	File2Output	File1State	File2State
0	Initial	Initial	Initial	S1 1 T0	S0 1 T0
1	f_X:61	1	1	S1 1 T1	S1 1 T1
2	f_X:61	1	1	S1 1 T2	S1 1 T2
3	f_X:61	1	1	S1 1 T3	S1 1 T3
4	f_X:61	1	1	S1 1 T4	S1 1 T4
5	f_X:61	1	1	S1 1 T5	S1 1 T5
6	f_X:61	0	0	S0 0 T5	S2 0 T5
7	f_X:52	0	0	S0 0 T0	S2 0 T0
8	f_X:52	1	0	Null	Null

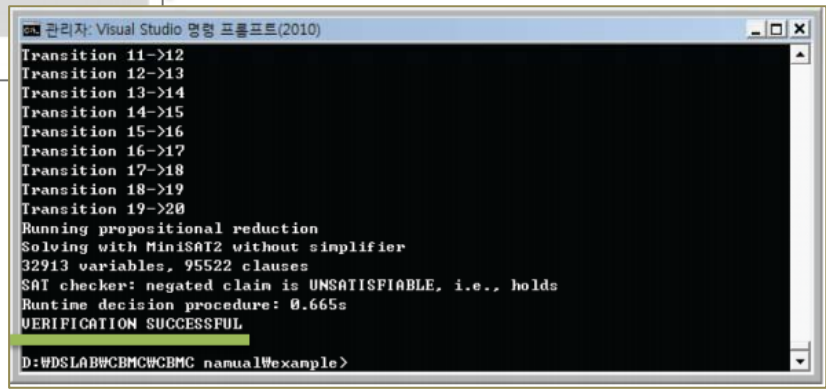
Verification and model check Ready

VIS Analyzer

+ To use/execute the VIS efficiently
  + The VIS has no GUI
+ Display the verification results in various forms

FBDtoVerilog + HW-CBMC

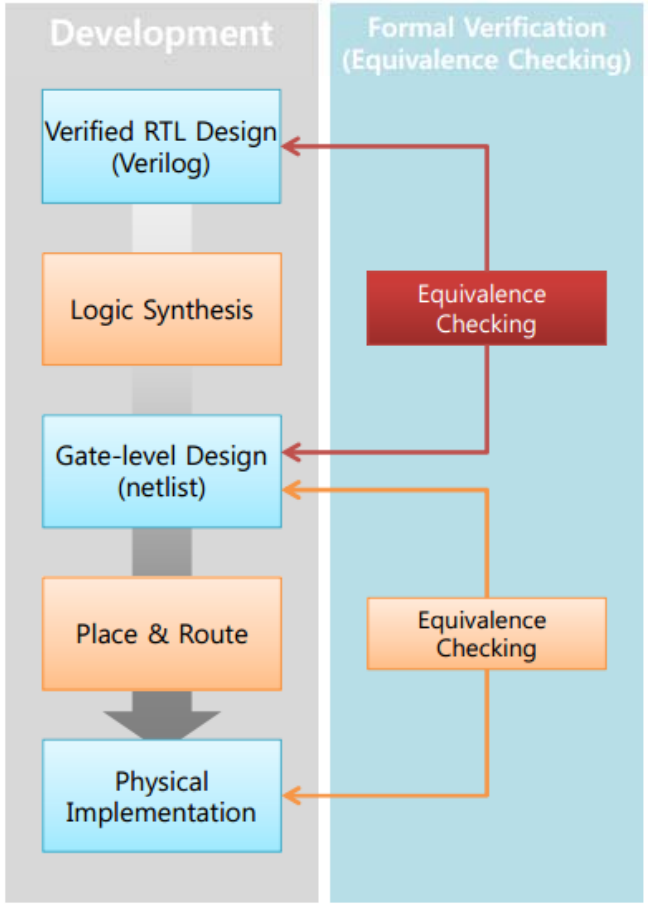




Equivalence Checking between FBD(→ Verilog) and ANSI-C
+ for demonstrating safety of the FBDtoC translator

+ Provides the checking process
+ Not fully automated

EDIFtoBLIF-MV + VIS



The VerilogtoBLIF-MV Mechanical Transformation

+ For the equivalence checking between Verilog and Netlist
+ For the safety demonstration of FPGA Synthesis tools

+ Transforms a Netlist (in EDIF format) into a program of BLIF-MV format
+ Then performs the VIS Equivalence Checking

EIDF File

```
(edif th_LO_SG1_LEVEL_Trip_Logic
  (edifVersion 2 0 0)
  (edifLevel 0)
  (keywordMap (keywordLevel 0))
  (status
    (written
      (timeStamp 2013 3 1 20 11 53)
      (author "Synopsys, Inc.")
      (program "Synplify Pro" (version "E-2010.09A-1, mapper
    )
  )
  (library PA3
    (edifLevel 0)
    (technology (numberDefinition ))
    (cell XA1A (cellType GENERIC)
      (property dont_touch (string "false"))
      (view prim (viewType NETLIST)
        (interface
          (port Y (direction OUTPUT)
          (property function (string "!(A ^ B) & C"))
)
          (port A (direction INPUT)
)
          (port B (direction INPUT)
)
          (port C (direction INPUT)
)
        )
        (property is_combinational (integer 1))
      )
    )
    (cell VCC (cellType GENERIC)
      (property dont_touch (string "false"))
      (view prim (viewType NETLIST)
        (interface
          (port Y (direction OUTPUT)
```

Mechanical Transformation

```
# cell th_LO_SG1_LEVEL_Trip_Logic
.model th_LO_SG1_LEVEL_Trip_Logic
# interface - I/O ports
.inputs reset f_LO_SG1_LEVEL_Val_Out<0> f_LO_SG1_LEVEL_Val_Out<1>
.outputs th_LO_SG1_LEVEL_Trip_Logic
.names N_16$M6 N_10$M6 reset_pad prev_status_RNO_0
.def 0
0 0 0 1
0 1 0 1
1 0 0 1
# latches
# init
.r prev_status_0
.def 0
.latch prev_status_RNO_0 prev_status_0
.names N_5$M5 th_LO_SG1_LEVEL_Trip_Logic_pad
.def 0
1 1
.names f_LO_SG1_LEVEL_Val_Out<15> f_LO_SG1_LEVEL_Val_Out_pad_15
.def 0
1 1
.names f_LO_SG1_LEVEL_Val_Out<14> f_LO_SG1_LEVEL_Val_Out_pad_14
.def 0
1 1
.names f_LO_SG1_LEVEL_Val_Out<13> f_LO_SG1_LEVEL_Val_Out_pad_13
.def 0
1 1
.names f_LO_SG1_LEVEL_Val_Out<12> f_LO_SG1_LEVEL_Val_Out_pad_12
.def 0
1 1
.names f_LO_SG1_LEVEL_Val_Out<11> f_LO_SG1_LEVEL_Val_Out_pad_11
.def 0
1 1
.names f_LO_SG1_LEVEL_Val_Out<10> f_LO_SG1_LEVEL_Val_Out_pad_10
.def 0
1 1
```

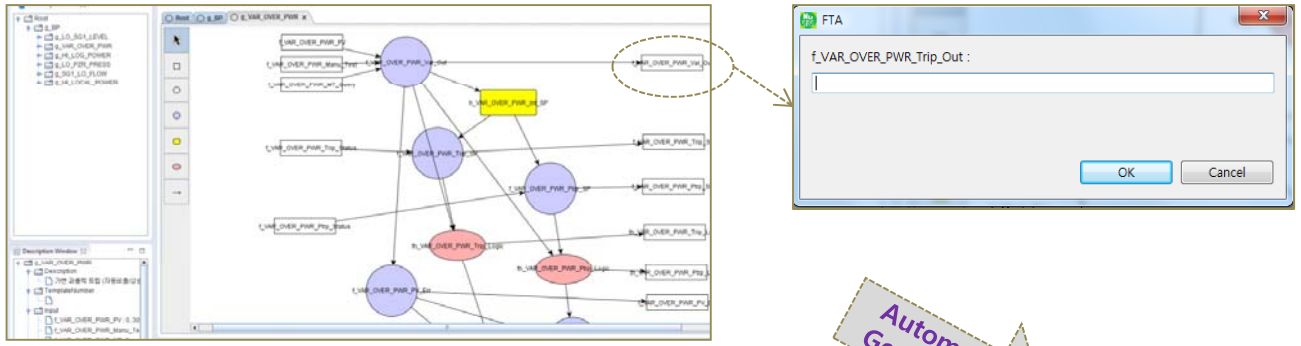
BLIF-MV File

KU KONKUK UNIVERSITY

SAFETY ANALYSIS PROCESS

NuSCR FTA

+ We need to define the value of an output variable



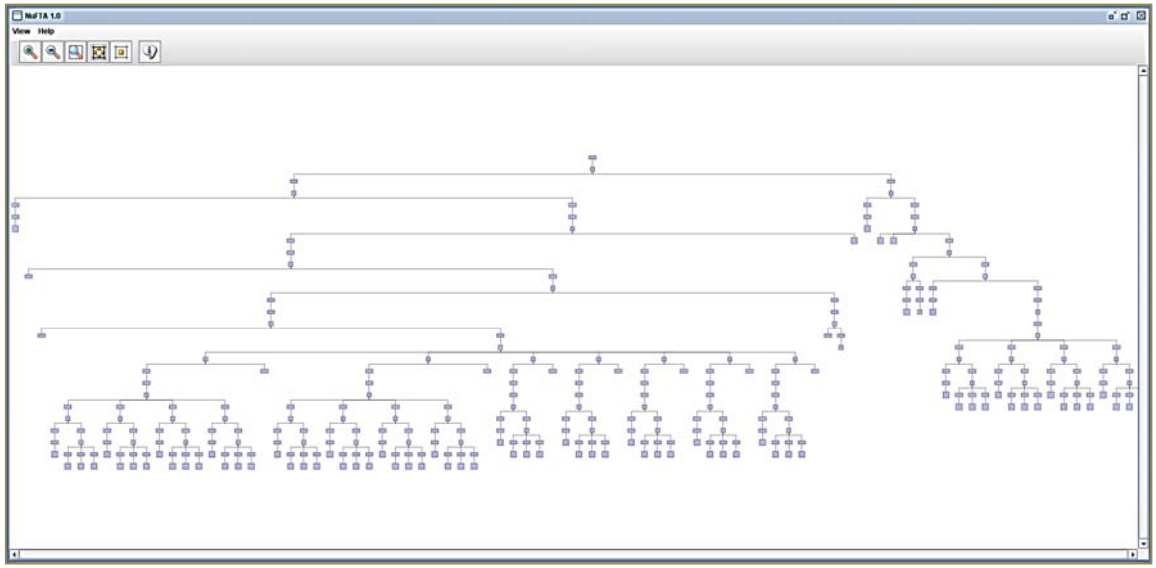
The NuSCR Formal SRS

Automatic Generation

The NuSCRtoFT Mechanical Generation

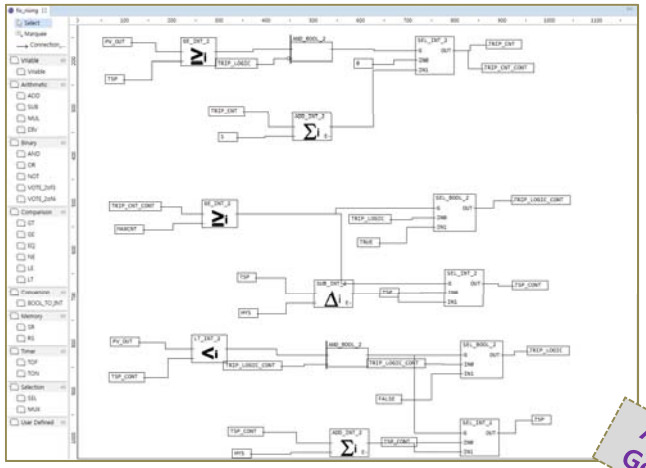
+ Generates a fault tree
+ from an NuSCR formal SRS
+ for a specific (important) output variable

+ Calculates (minimal) cut-sets



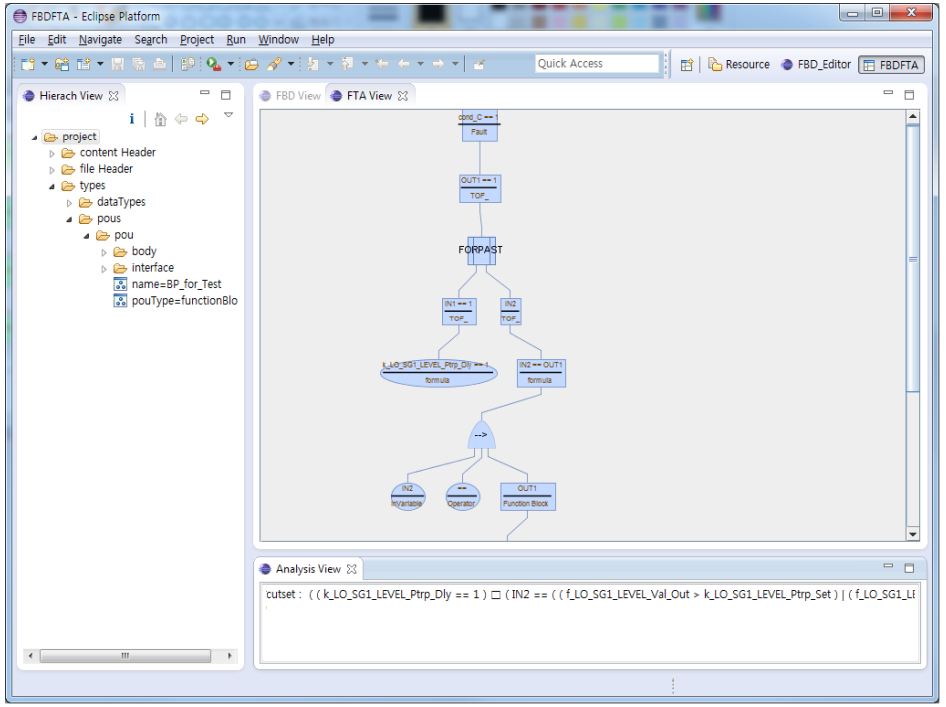
FT (Fault Tree)

FBD FTA



An FBD Program

Automatic Generation



FT (Fault Tree)

The FBDtoFT Mechnical Generation

+ Generates a fault tree
+ from an FBD program
+ for a specific (important) output variable

+ Calculates (minimal) cut-sets

+ Uses the Temporal Fault Tree semantics

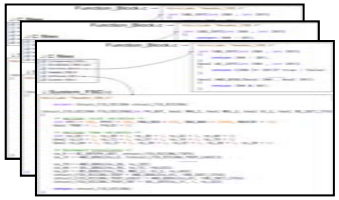
+ Under developing the minimal cut-set optimization

KU KONKUK UNIVERSITY

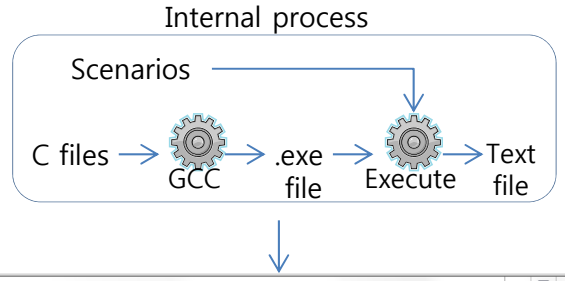
SUPPLEMENTARY TOOLS

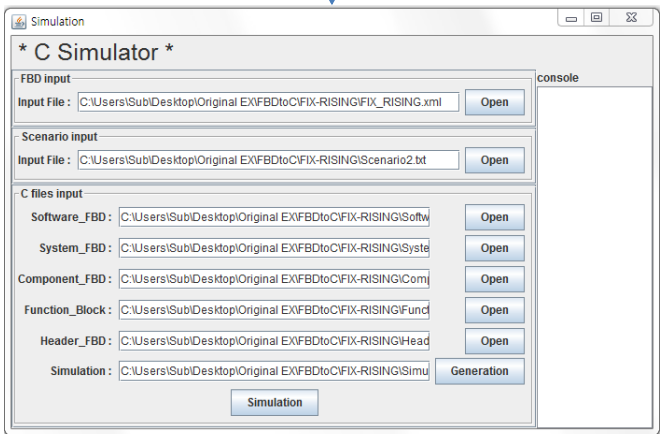
C Simulator

Scenarios
form Scenario Generator

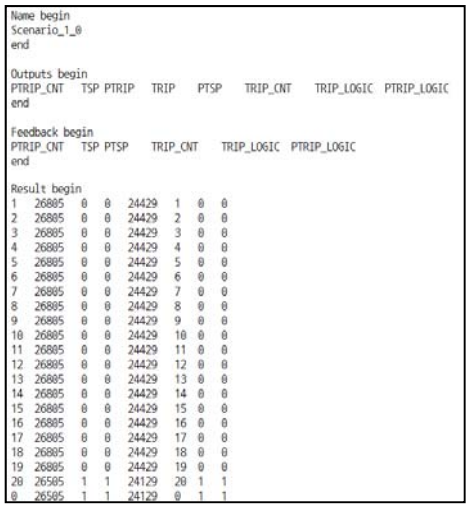


C files
from FBDtoC translator





C Simulator



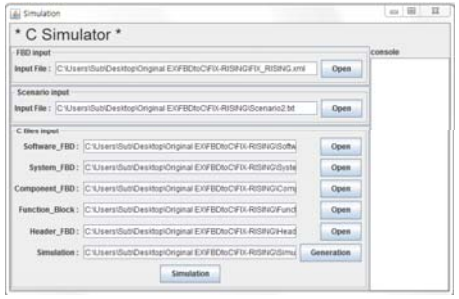
C Simulation Result (text)

C Simulator
+ Compiles an inputted C program from FBDtoC translator
+ with GCC compiler into executable file

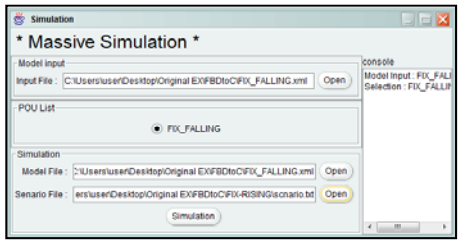
+ Simulates an executable file from GCC compiler
+ with a inputted Scenarios from Scenario Generator

+ Saves a result of simulation into text file

FBD-C Comparator



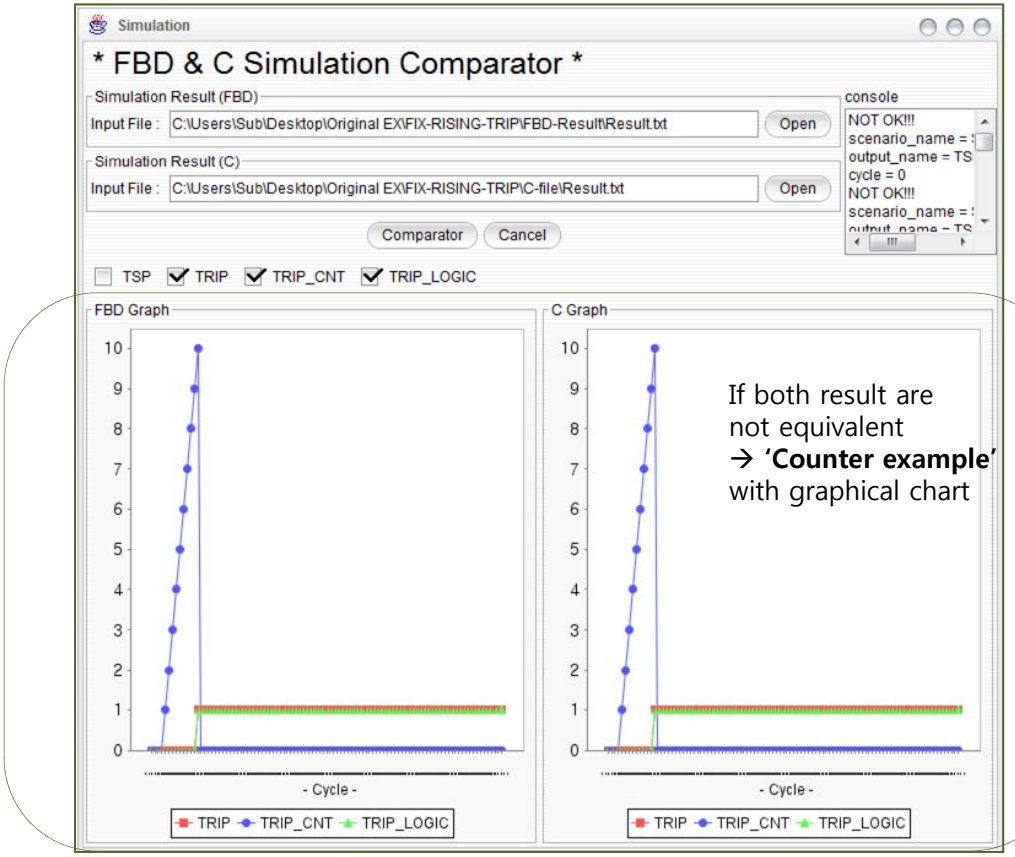
C Simulator



FBD Simulator

Input

Input



If both result are not equivalent
→ **'Counter example'** with graphical chart

FBD-C Comparator

BD-C Comparator

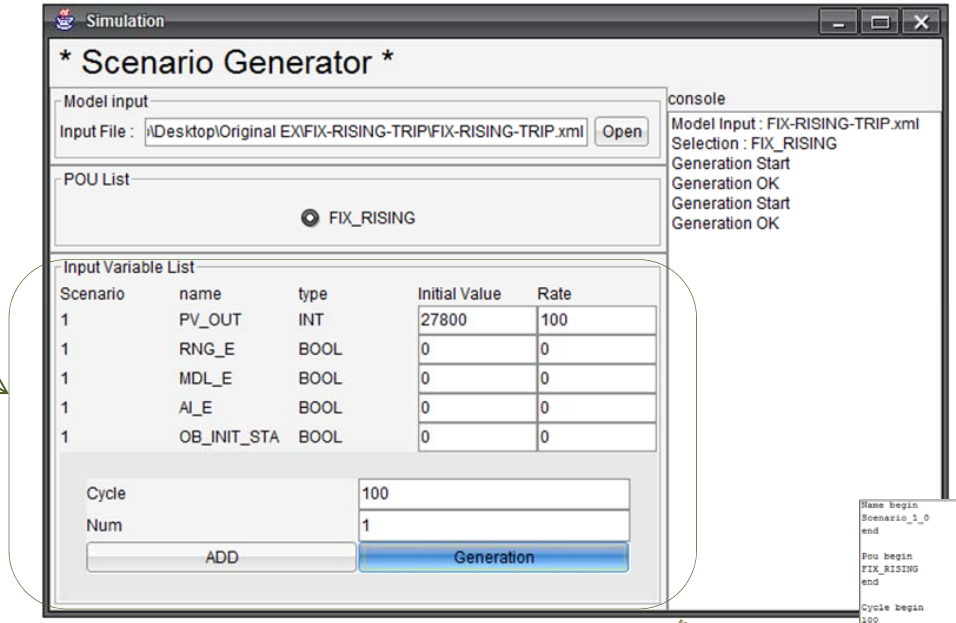
+ Compares between simulation result from C simulator and FBD simulator

+ If both result are equivalent, it produce the 'True'
+ Otherwise, it produce a counter example with graphical chart

FBD Scenario Generator

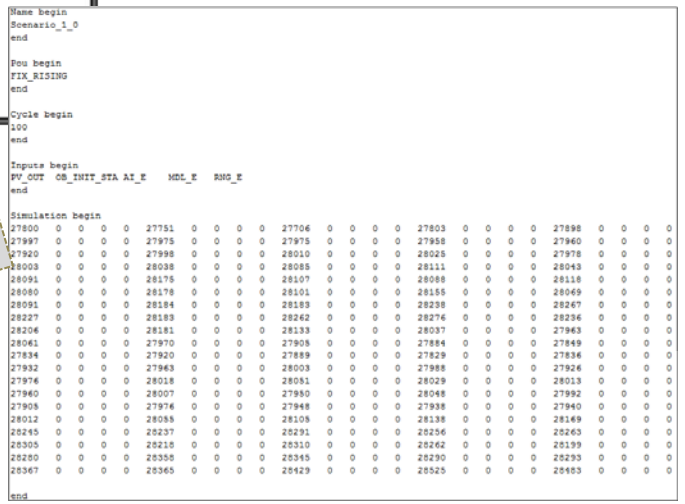
Setting table

- Initial value
- Rate-of-change
- Maximum Cycle
- The number of scenarios



Scenario Generator

Scenario Generation



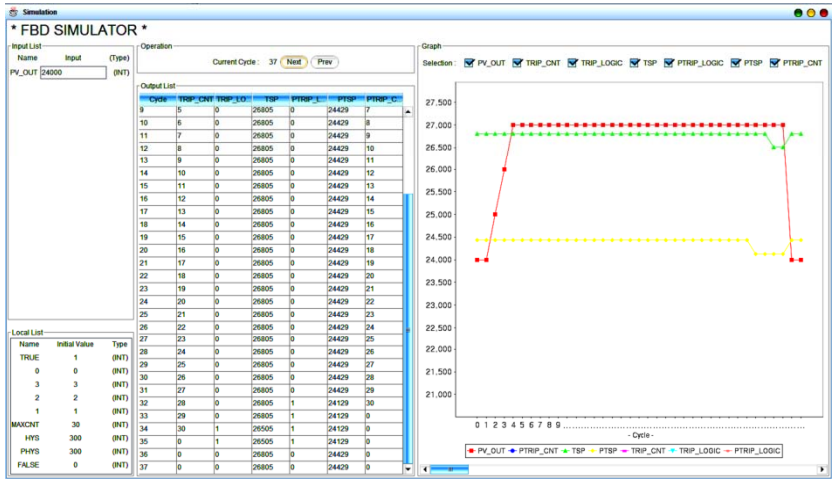
Automatic Generation of Scenarios

+ Receiving values of setting table from user
+ Setting table reflect a domain feature

+ Generates Random Scenarios based on the table values set

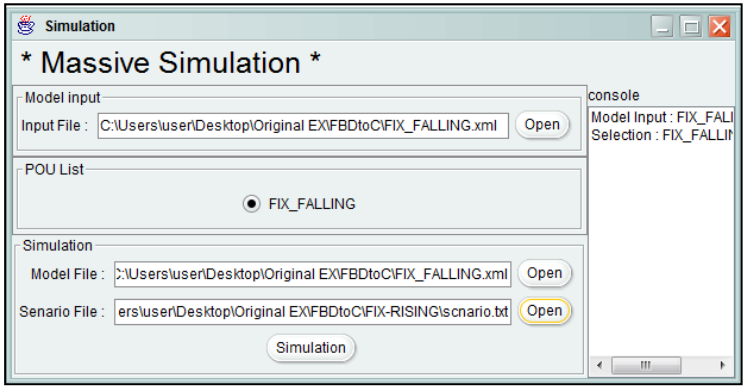
FBD Simulator

Type A



Scenario Generator with graphical chart

Type B



Scenario Generator for massive scenario

The Scenario Generator with graphical chart

+ Receives value from user in one cycle
+ Simulates one by one cycle

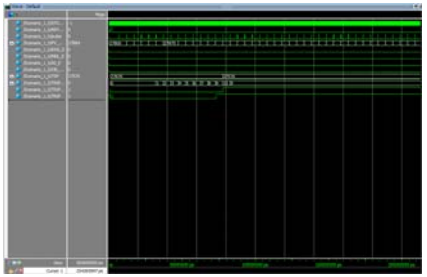
+ Results in graphical chart → It can verify function of an FBD

The Scenario Generator for massive scenario

+ Receives scenarios from Scenario Generator and FBD
+ Simulates massive scenarios

+ Results in a text file

FBD-Verilog Comparator

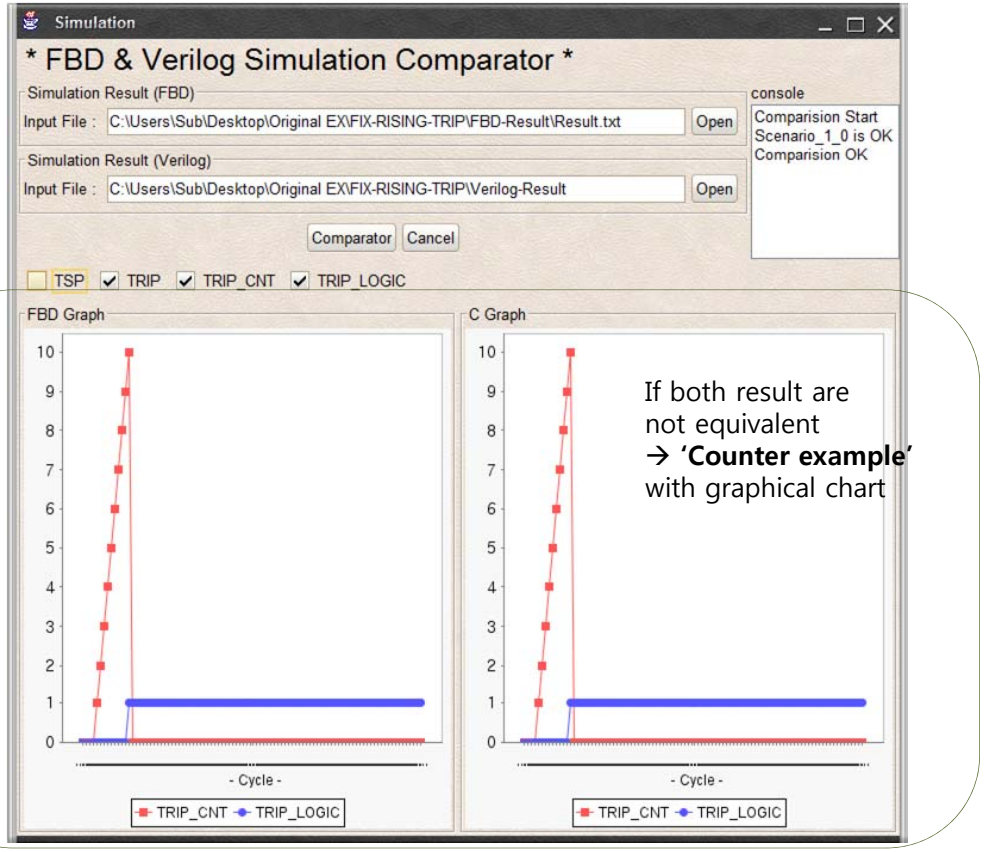


ModelSim
Verilog Simulator

FBD Simulator

Input

Input



FBD-Verilog Comparator

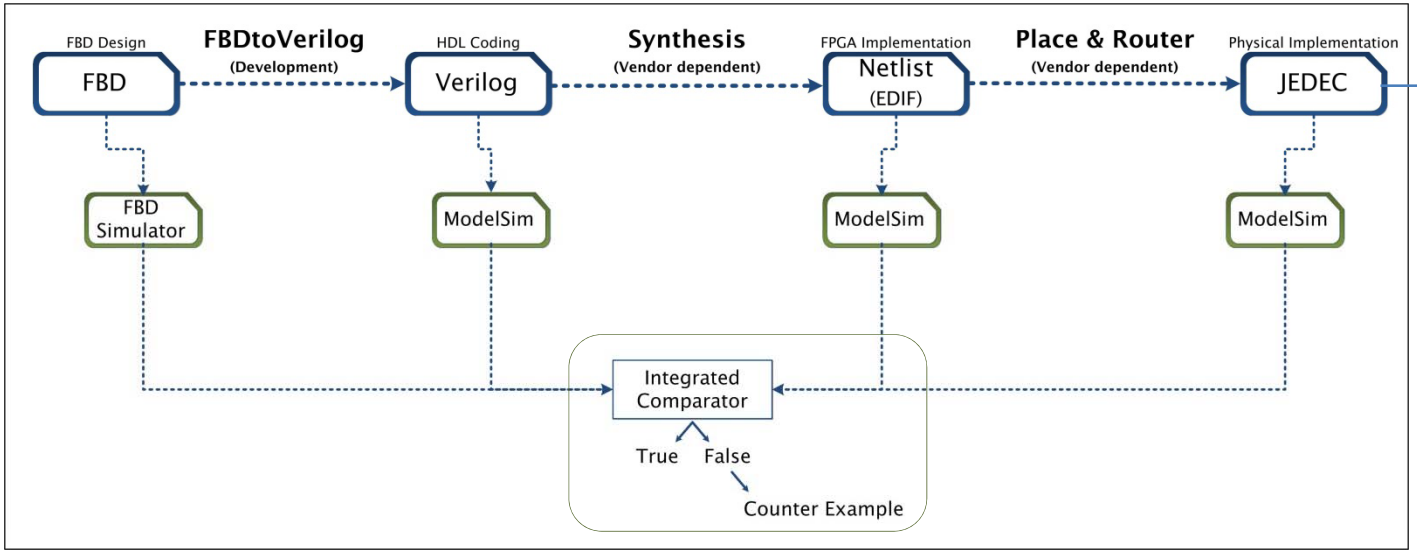
The FBD-Verilog Comparator

+ Compare between simulation results from ModelSim and the FBD simulator

+ If both result are equivalent, it produce the 'True'
+ Otherwise, it produces a counter example with graphical chart

FBD-Verilog-Netlist-JEDEC Comparator

Future Work !!



FPGA

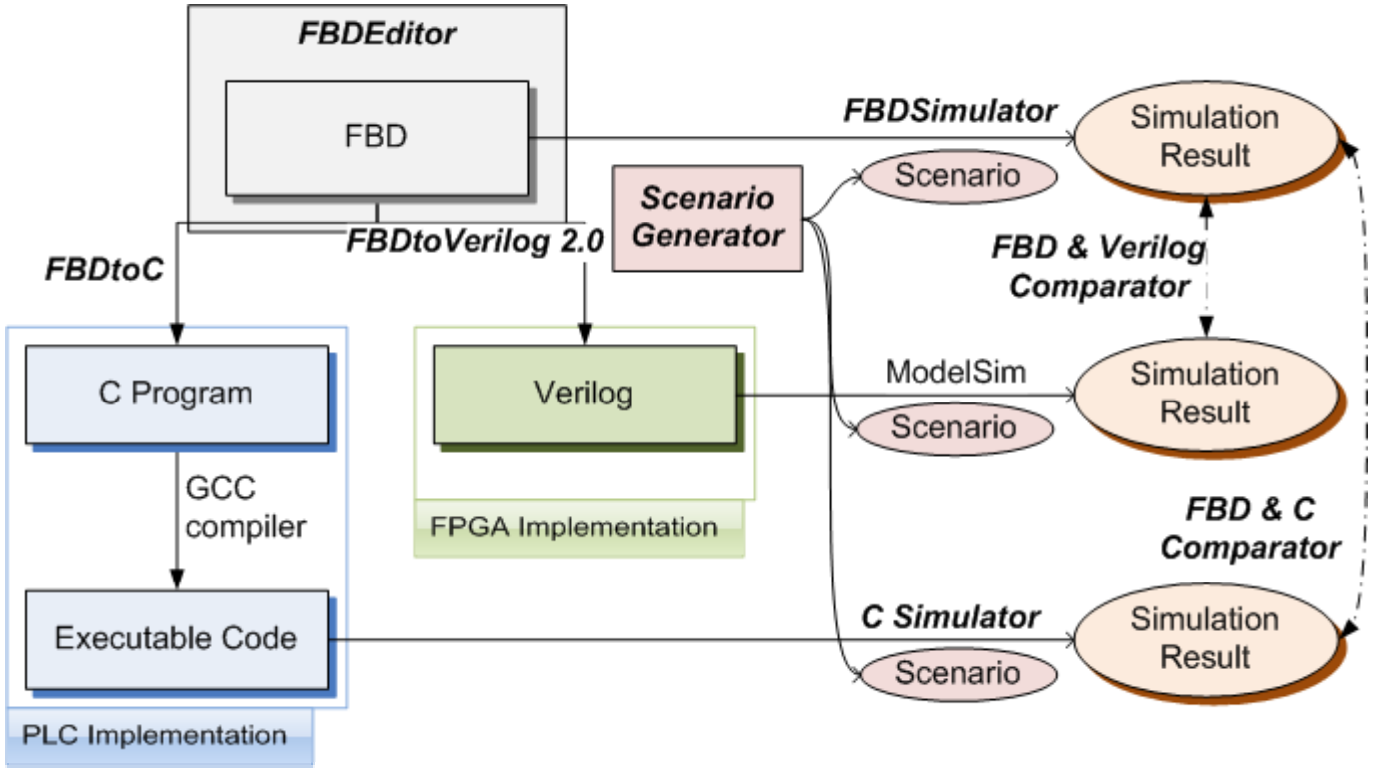
The Integrated Comparator for (FBD-Verilog-Netlist-JEDEC) synthesis process

+ Receives variable simulation results from FBD, Verilog, Netlist and JEDEC simulations
+ It will provide user more efficient verification environment for developing FPGA software

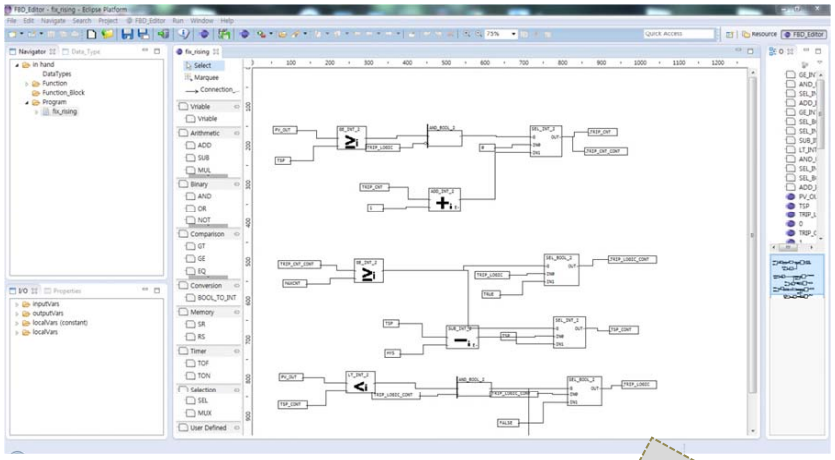
THE CASE STUDY IN THE PAPER

The Case Study in the Paper

Goal : Validate the correctness of two transformations (FBDtoC and FBDtoVerilog 2.0)



The FBDtoC Transformation



FBDs in the FBD Editor

Mechanical Transformation

by FBDtoC

Function_Block.c →

```c
#include "Header_FBD.h"

int SUB_INT(int IN0 , int IN1)
{
    return IN0 - IN1;
}
bool GE_INT(int IN0 , int IN1)
{
    return (IN0 >= IN1)? true : false;
}
                                .....
bool AND_BOOL(bool IN0 , bool IN1)
{
    return IN0 & IN1;
}
```

↓ C files

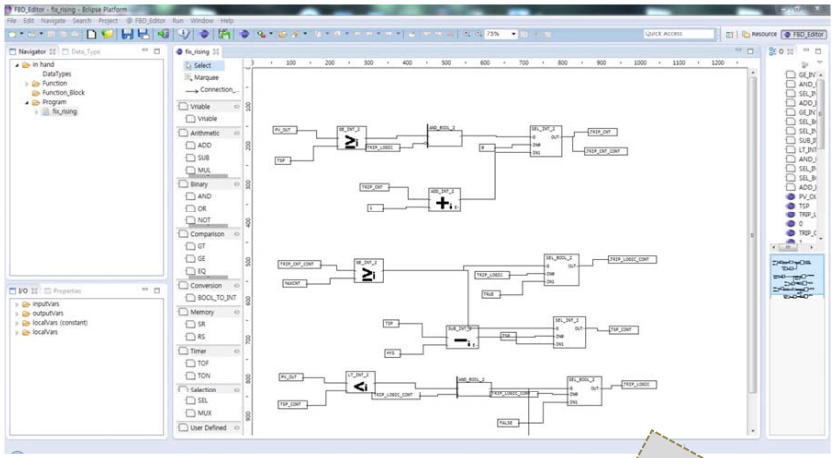
- Component_FBD.c
- FIX-RISING-TRIP.xml
- Function_Block.c
- Header_FBD.h
- Software_FBD.c
- System_FBD.c

↓ System_FBD.c

```c
#include "Header_FBD.h"

    extern Struct_FIX_RISING struct_FIX_RISING;

Struct_FIX_RISING FIX_RISING(int PV_OUT, bool RNG_E, bool MDL_E, bool AI_E, bool OB_INIT_STA)
{
    /* declear Local variables */
    int HYS = 300, PHYS = 300, RNG_MIN = 600, RNG_MAX = 29400, MAXCNT = 10;
    bool TRUE = 1, FALSE = 0;

    /* declear Temp variables */
    int to_63 = 0, to_58 = 0, to_38 = 0, to_42 = 0, to_40 = 0;
    bool to_9 = 0, to_12 = 0, to_57 = 0, to_10 = 0, to_79 = 0;
    bool to_64 = 0, to_31 = 0, to_39 = 0, to_61 = 0, to_28 = 0, to_29 = 0;

    /* Backward translaton */
    to_9 = GE_INT(PV_OUT, struct_FIX_RISING.TSP);
    to_31 = AND_BOOL(to_9, !struct_FIX_RISING.TRIP_LOGIC);
                                        .....
    to_79 = AND_BOOL(to_28, to_29);
    to_64 = SEL_BOOL(to_39, to_57, FALSE);
    to_61 = OR_BOOL(!to_79, MDL_E, AI_E, to_64);
    struct_FIX_RISING.TRIP = AND_BOOL(to_61, !OB_INIT_STA);
    struct_FIX_RISING.TRIP_LOGIC = AND_BOOL(to_61, !OB_INIT_STA);
    struct_FIX_RISING.TRIP_CNT = SEL_INT(to_31, 0, to_42);

    return struct_FIX_RISING;
}
```

C programs generated

The FBDtoVerilog Transformation



FBDs in the FBD Editor

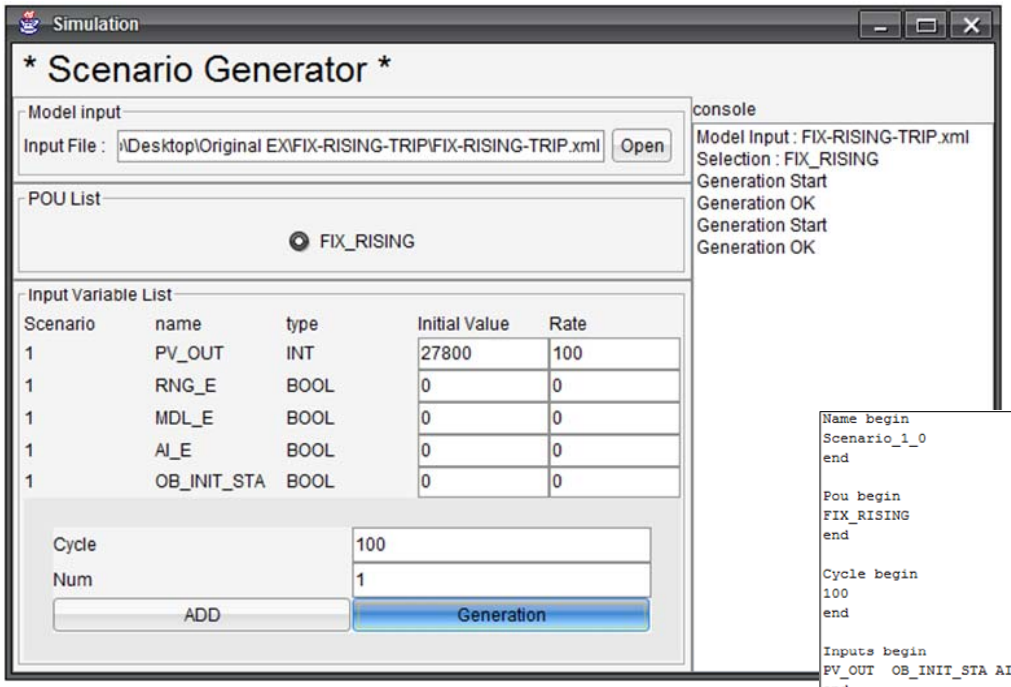
Mechanical Transformation

by FBDtoVerilog 2.0

```
module FIX_RISING (rst, clk, pulse, PV_OUT, RNG_E, MDL_E, AI_E,
                   OB_INIT_STA, TSP, TRIP_CNT, TRIP_LOGIC, TRIP);

    input clk;
    input rst;
...
    input    OB_INIT_STA;
    output  [15:0] TSP;     reg     [15:0] TSP;
    output  [15:0] TRIP_CNT;    reg     [15:0] TRIP_CNT;
...
    parameter   [15:0] MAXCNT = 10;

    wire     GE_INT_2_wire_9_OUT;
    wire     GE_INT_2_wire_10_OUT;
    wire     LT_INT_2_wire_12_OUT;
    wire     LT_INT_2_wire_28_OUT;
    wire     LT_INT_2_wire_29_OUT;
...
    wire    [15:0] SEL_INT_2_wire_63_OUT;
    wire     SEL_BOOL_2_wire_64_OUT;
    wire    [15:0] SEL_INT_2_wire_65_OUT;
    wire  AND_BOOL_2_wire_71_OUT;
    wire  AND_BOOL_2_wire_79_OUT;

    GE_INT_2 GE_INT_2_9(rst, clk, PV_OUT, TSP, GE_INT_2_wire_9_OUT);
    GE_INT_2 GE_INT_2_10(rst, clk, SEL_INT_2_wire_63_OUT, MAXCNT, GE_INT_2_wire_10_OUT);
    LT_INT_2 LT_INT_2_12(rst, clk, PV_OUT, SEL_INT_2_wire_58_OUT, LT_INT_2_wire_12_OUT);
    LT_INT_2 LT_INT_2_28(rst, clk, RNG_MIN, PV_OUT, LT_INT_2_wire_28_OUT);
    LT_INT_2 LT_INT_2_29(rst, clk, PV_OUT, RNG_MAX, LT_INT_2_wire_29_OUT);
...
    assign TRIP = AND_BOOL_2_wire_71_OUT;

    always @(posedge rst or posedge clk or posedge pulse)
    begin
        if(rst) begin
            TSP <= 27870;
            TRIP_CNT <= 0;
            TRIP_LOGIC <= 0;
        end else if (clk) begin
        end
        if (pulse) begin
            TSP <= SEL_INT_2_wire_65_OUT;
            TRIP_LOGIC <= AND_BOOL_2_wire_71_OUT;
            TRIP_CNT <= SEL_INT_2_wire_63_OUT;
        end
    end
endmodule
```

Verilog programs generated

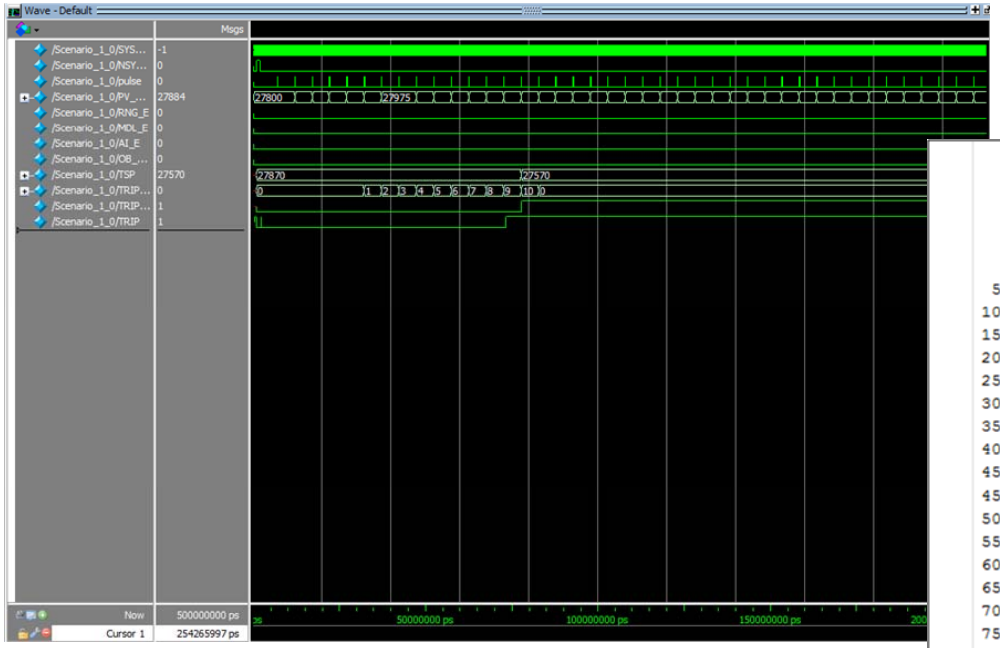
Scenario Generator



Scenario Generation

```
Name begin
Scenario_1_0
end

Pou begin
FIX_RISING
end

Cycle begin
100
end

Inputs begin
PV_OUT  OB_INIT_STA AI_E    MDL_E   RNG_E
end

Simulation begin
27800   0   0   0   0   27751   0   0   0   0   27706   0   0   0   0   27803   0   0   0   0   27898   0   0   0   0
27997   0   0   0   0   27975   0   0   0   0   27975   0   0   0   0   27958   0   0   0   0   27960   0   0   0   0
27920   0   0   0   0   27998   0   0   0   0   28010   0   0   0   0   28025   0   0   0   0   27978   0   0   0   0
28003   0   0   0   0   28038   0   0   0   0   28085   0   0   0   0   28111   0   0   0   0   28043   0   0   0   0
28091   0   0   0   0   28175   0   0   0   0   28107   0   0   0   0   28088   0   0   0   0   28118   0   0   0   0
28080   0   0   0   0   28178   0   0   0   0   28101   0   0   0   0   28155   0   0   0   0   28069   0   0   0   0
28091   0   0   0   0   28184   0   0   0   0   28183   0   0   0   0   28238   0   0   0   0   28267   0   0   0   0
28227   0   0   0   0   28183   0   0   0   0   28262   0   0   0   0   28276   0   0   0   0   28236   0   0   0   0
28206   0   0   0   0   28181   0   0   0   0   28133   0   0   0   0   28037   0   0   0   0   27963   0   0   0   0
28061   0   0   0   0   27970   0   0   0   0   27905   0   0   0   0   27884   0   0   0   0   27849   0   0   0   0
27834   0   0   0   0   27920   0   0   0   0   27889   0   0   0   0   27829   0   0   0   0   27836   0   0   0   0
27932   0   0   0   0   27963   0   0   0   0   28003   0   0   0   0   27988   0   0   0   0   27926   0   0   0   0
27976   0   0   0   0   28018   0   0   0   0   28051   0   0   0   0   28029   0   0   0   0   28013   0   0   0   0
27960   0   0   0   0   28007   0   0   0   0   27950   0   0   0   0   28048   0   0   0   0   27992   0   0   0   0
27905   0   0   0   0   27976   0   0   0   0   27948   0   0   0   0   27938   0   0   0   0   27940   0   0   0   0
28012   0   0   0   0   28055   0   0   0   0   28105   0   0   0   0   28138   0   0   0   0   28169   0   0   0   0
28245   0   0   0   0   28237   0   0   0   0   28291   0   0   0   0   28256   0   0   0   0   28263   0   0   0   0
28305   0   0   0   0   28218   0   0   0   0   28310   0   0   0   0   28262   0   0   0   0   28199   0   0   0   0
28280   0   0   0   0   28358   0   0   0   0   28345   0   0   0   0   28290   0   0   0   0   28293   0   0   0   0
28367   0   0   0   0   28365   0   0   0   0   28429   0   0   0   0   28525   0   0   0   0   28483   0   0   0   0

end
```

Verilog Simulation with ModelSim



in wave form

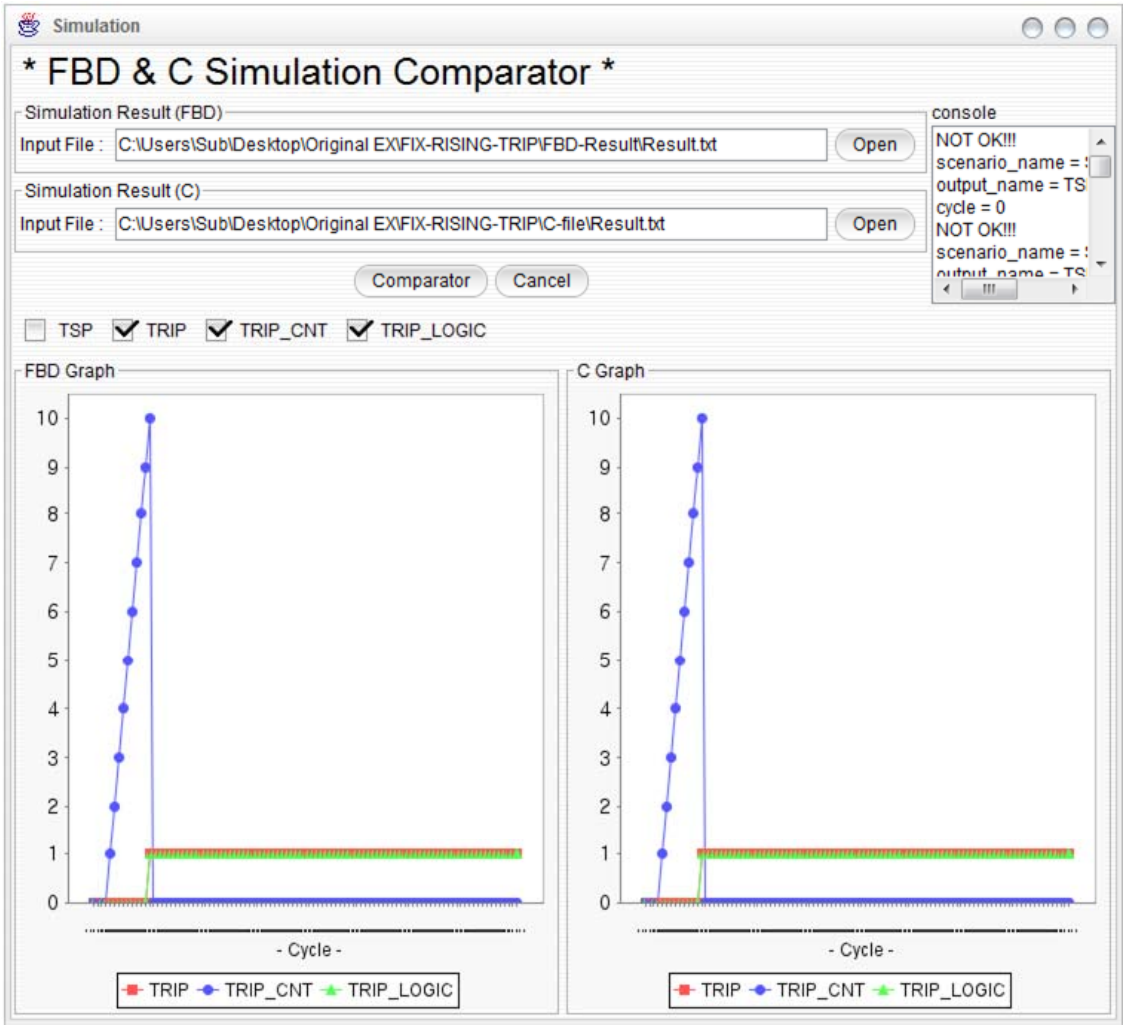
ps	delta	/Scenario_1_0/SYSCLK /Scenario_1_0/NSYSRESET	/Scenario_1_0/pulse	/Scenario_1_0/PV_OUT	/Scenario_1_0/RNG_E	/Scenario_1_0/MDL_E /Scenario_1_0/AI_E	/Scenario_1_0/OB_INIT_STA	/Scenario_1_0/TSP	/Scenario_1_0/TRIP_CNT /Scenario_1_0/TRIP /Scenario_1_0/TRIP_LOGIC
0	+0	0	0	0	27800	0 0	0	x	x x x
50000	+0	-1	0	0	27800	0 0	0	x	x x x
100000	+0	0	0	0	27800	0 0	0	x	x x x
150000	+0	-1	0	0	27800	0 0	0	x	x x x
200000	+0	0	0	0	27800	0 0	0	x	x x x
250000	+0	-1	0	0	27800	0 0	0	x	x x x
300000	+0	0	0	0	27800	0 0	0	x	x x x
350000	+0	-1	0	0	27800	0 0	0	x	x x x
400000	+0	0	0	0	27800	0 0	0	x	x x x
450000	+0	-1	0	0	27800	0 0	0	x	x x x
450000	+2	-1	0	0	27800	0 0	0	x	x x 1
500000	+0	0	0	0	27800	0 0	0	x	x x 1
550000	+0	-1	0	0	27800	0 0	0	x	x x 1
600000	+0	0	0	0	27800	0 0	0	x	x x 1
650000	+0	-1	0	0	27800	0 0	0	x	x x 1
700000	+0	0	0	0	27800	0 0	0	x	x x 1
750000	+0	-1	0	0	27800	0 0	0	x	x x 1
800000	+0	0	0	0	27800	0 0	0	x	x x 1
850000	+0	-1	0	0	27800	0 0	0	x	x x 1
900000	+0	0	0	0	27800	0 0	0	x	x x 1
950000	+0	-1	0	0	27800	0 0	0	x	x x 1
1000000	+0	-1	*	0	27800	0 0	0	x	x x 1
1000000	+2	0	*	0	27800	0 0	0	x	x x 0
1000000	+3	0	*	0	27800	0 0	0	27870	0 0 0
1050000	+0	-1	*	0	27800	0 0	0	27870	0 0 0
1100000	+0	0	*	0	27800	0 0	0	27870	0 0 0
1150000	+0	-1	*	0	27800	0 0	0	27870	0 0 0
1200000	+0	0	*	0	27800	0 0	0	27870	0 0 0
1250000	+0	-1	*	0	27800	0 0	0	27870	0 0 0
1300000	+0	0	*	0	27800	0 0	0	27870	0 0 0
1350000	+0	-1	*	0	27800	0 0	0	27870	0 0 0
1400000	+0	0	*	0	27800	0 0	0	27870	0 0 0
1450000	+0	-1	*	0	27800	0 0	0	27870	0 0 0
1500000	+0	0	*	0	27800	0 0	0	27870	0 0 0
1550000	+0	-1	*	0	27800	0 0	0	27870	0 0 0
1600000	+0	0	*	0	27800	0 0	0	27870	0 0 0
1650000	+0	-1	*	0	27800	0 0	0	27870	0 0 0
1700000	+0	0	*	0	27800	0 0	0	27870	0 0 0
1750000	+0	-1	*	0	27800	0 0	0	27870	0 0 0
1800000	+0	0	*	0	27800	0 0	0	27870	0 0 0

in text

FBD & Verilog Comparator

→ **The FBDtoVerilog 2.0 transformation worked well!**

FBD & C Comparator



→ **The FBDtoC transformation worked well!**

In Summary

NuDE 2.0 can

- Provide a systematic MBD-based software development framework for the PLC & FPGA implementations of digital I&Cs, simultaneously
- Cope with various standards and regulations in regarding to software safety
- Reuse the PLC-based knowledge and experience accumulated for decades
- Reduce the risk of the sudden change of SW development paradigm from PLC to FPGA, through starting from the existing FBD programs not HDLs
- Be used as a medium of software design diversity to avoid CCF
- Consider also the safety demonstration of the commercial SW synthesis tools

THANK YOU

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