SAFECOMP 2011
The 30th International Conference on
Computer Safety, Reliability and Security
19-21 September 2011, Naples, Italy

Equivalence Checking between Function Block Diagrams and C Programs using HW-CBMC

Lee Dong-Ah
Dependable Software Laboratory
KONKUK University
South Korea

Contents

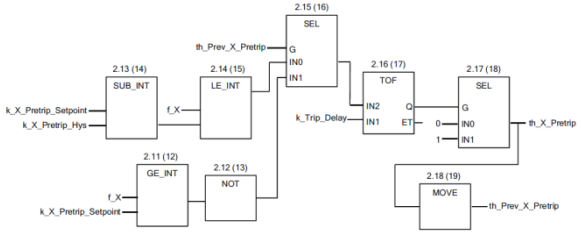
Equivalence Checking
between FBDs and C Programs using HW-CBMC

INTRODUCTION

Introduction

- Safety-critical systems often use function block diagrams (FBDs) to design embedded software
- The design is implemented using programming language e.g. C language
- The implementation must have same behavior with the design

Design



Function block diagram

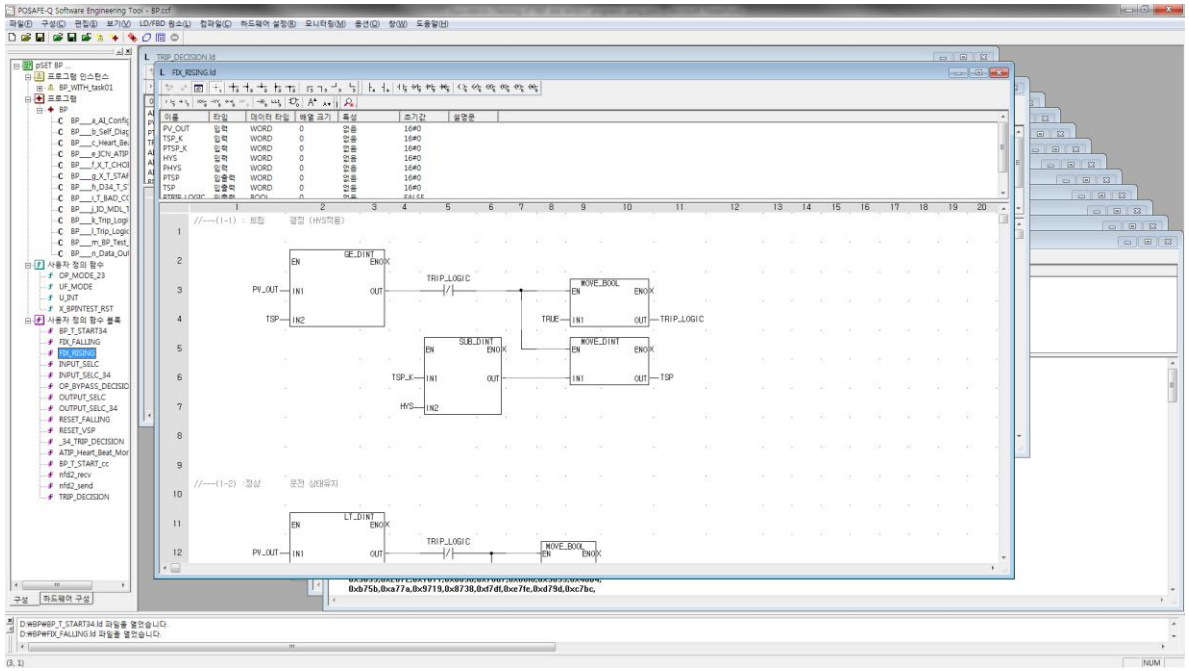
Implementation

```
12  extern struct MODULE_Z01 Z01;
13
14  int main()
15  {
16      //define variables
17
18      int c_out1;
19      unsigned int c_EN = 0;
20      unsigned int c_ENO;
21      int c_in1;
22
23      c_in1 = 1;
24      Z01.EN = c_EN;
25      Z01.in1 = c_in1;
26      set_inputs();
27
28      next_timeframe();
29
30      //assert(Z01.out1 != c_in1);
31      //MOVE_DINT(EN, in1, &ENO, &out1);
32
33      if(c_EN == 1) {
34          c_out1 = c_in1;
35      }
36      else {
37          c_out1 = c_in1;
38      }
39      assert(Z01.out1 == c_in1);
40
41      c_in1 = 1;
42      Z01.EN = c_EN;
43      Z01.in1 = c_in1;
44      set_inputs();
45
46      next_timeframe();
47
48      //assert(Z01.out1 != c_in1);
49      //MOVE_DINT(EN, in1, &ENO, &out1);
```

C language program

Introduction (cont'd)

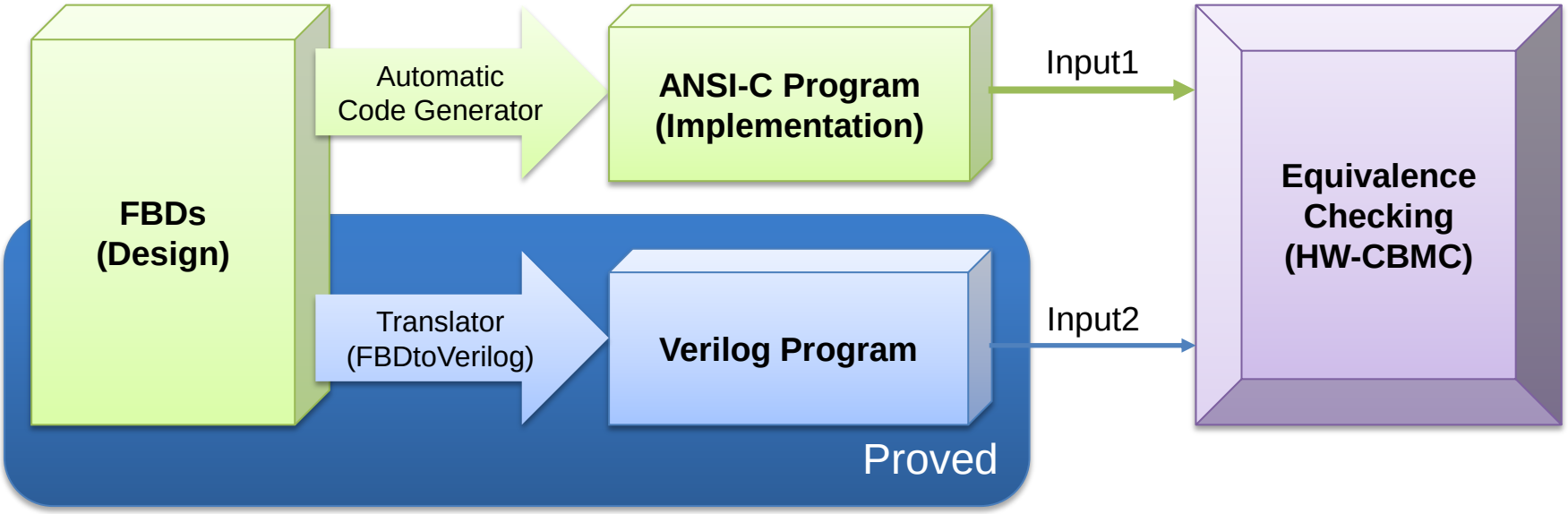
- POSAFE-Q Software Engineering Tool (pSET)
  - Korea Nuclear Instrumentation & Control System R&D Center (KNICS)
- FBD and Ladder Diagram (LD) to design a software of POSAFE-Q Programmable Logic Controller (PLC)
- ANSI-C language to implement the design



Introduction (cont'd)

Proposed Equivalence Checking technique between FBDs and ANSI-C program

- FBDs translated into Verilog using a part of FBDVerifier*
  - Translating FBDs to Verilog is proved in our previous work



*FBDVerifier : Interactive and Visual Analysis of Counterexample in Formal Verification of Function Block Diagram, JRPIT 2010

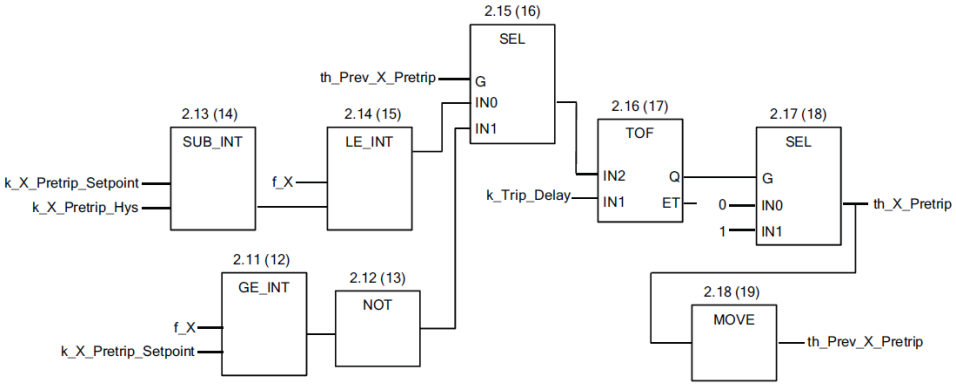
Equivalence Checking
between FBDs and C Programs using HW-CBMC

BACKGROUND

Background

Function Block Diagram (FBD)

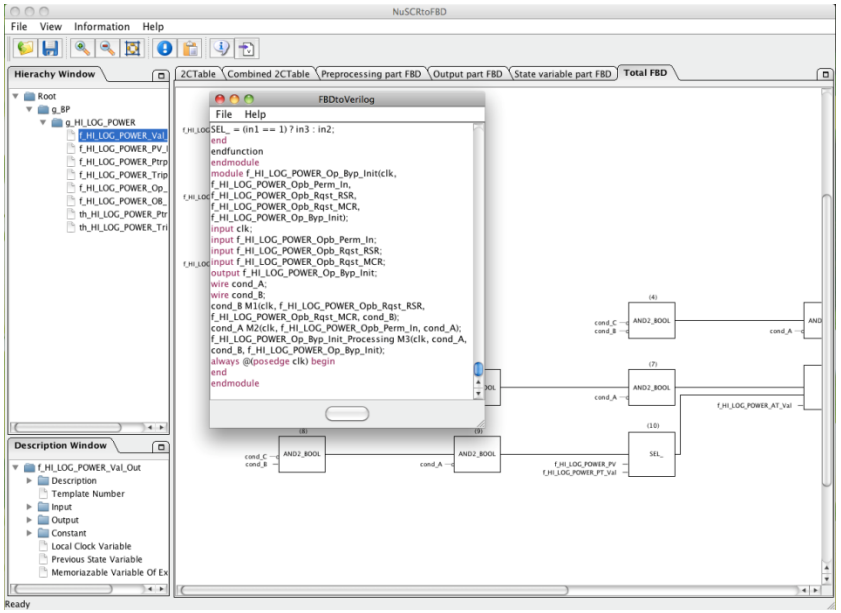
- IEC 61131-3 standard declared 5 programming languages for PLC
  - ST, LD, IL, SFC, FBD
- Sequential interconnections between function blocks
- pSET uses FBD and LD to design KNICS RPS software



Background (cont'd)

Translation FBDs into Verilog

- *FBDtoVerilog 1.0* CASE tool*
  - Standard XML format of FBD
  - PLCopen
- Translation rules
  - Unit FBD (Function Block)
  - Component FBD
  - System FBD



*FBDtoVerilog: A Vendor-Independent Translation from FBDs into Verilog Programs, SEKE 2011

Background (cont'd)

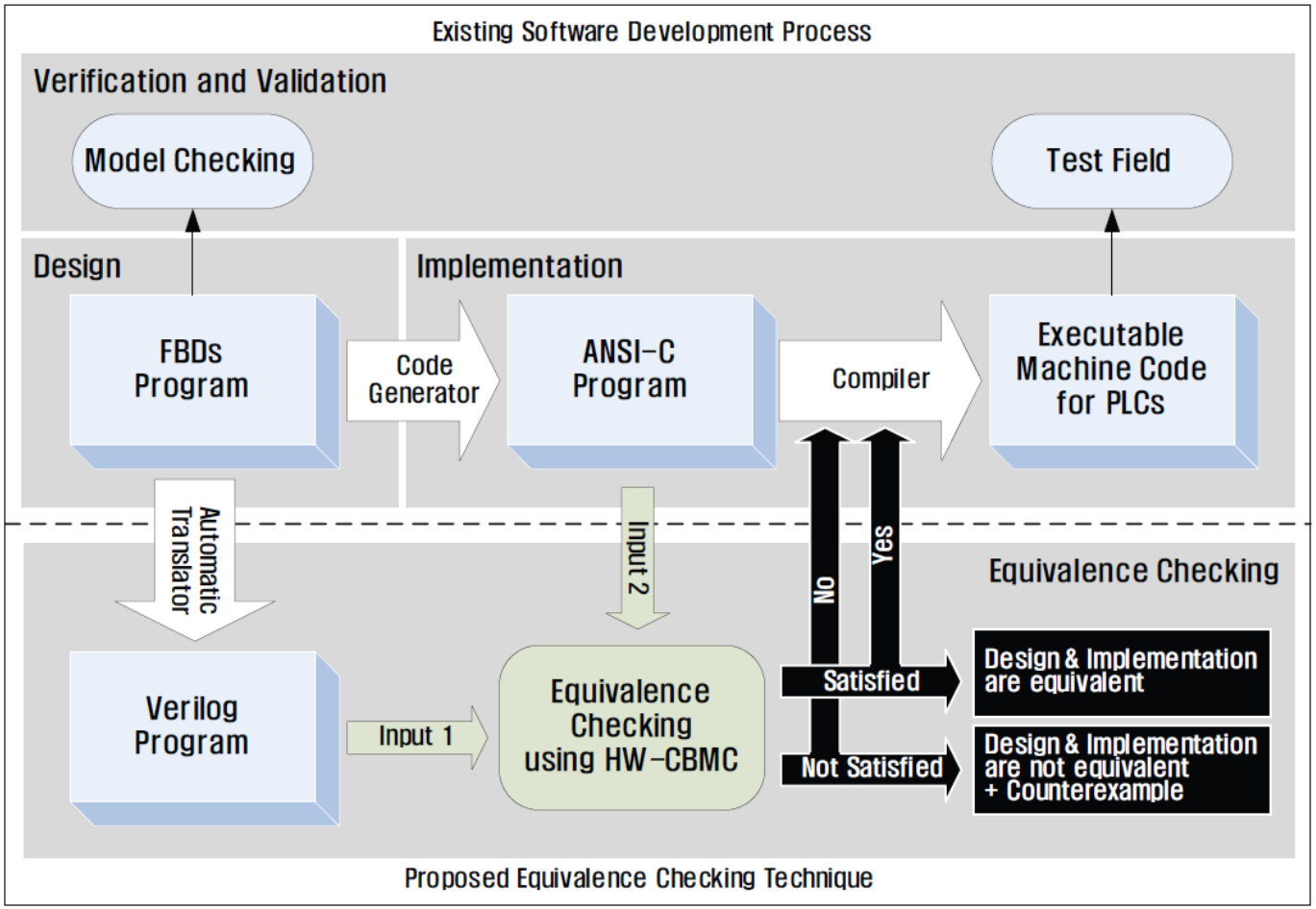
HW-CBMC

- Extensive testing and debugging tool of any embedded software
  - In particular the software which will be shipped with the circuit later on
- Verifying behavioral consistency between two implementations
  - ANSI-C program written for simulation
  - Register transfer level HDL (Verilog) which is an actual product

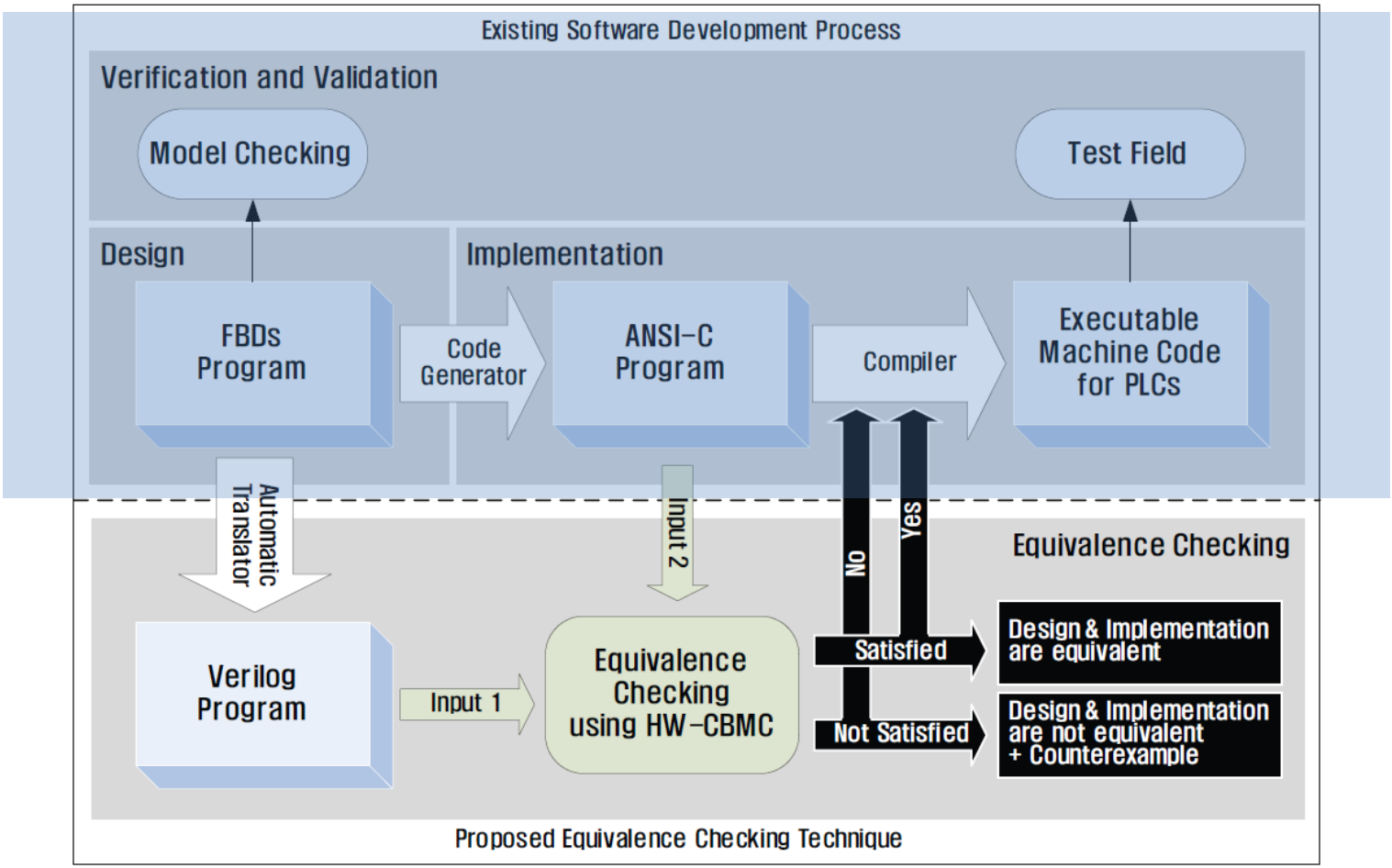
Equivalence Checking
between FBDs and C Programs using HW-CBMC

EQUIVALENCE CHECKING

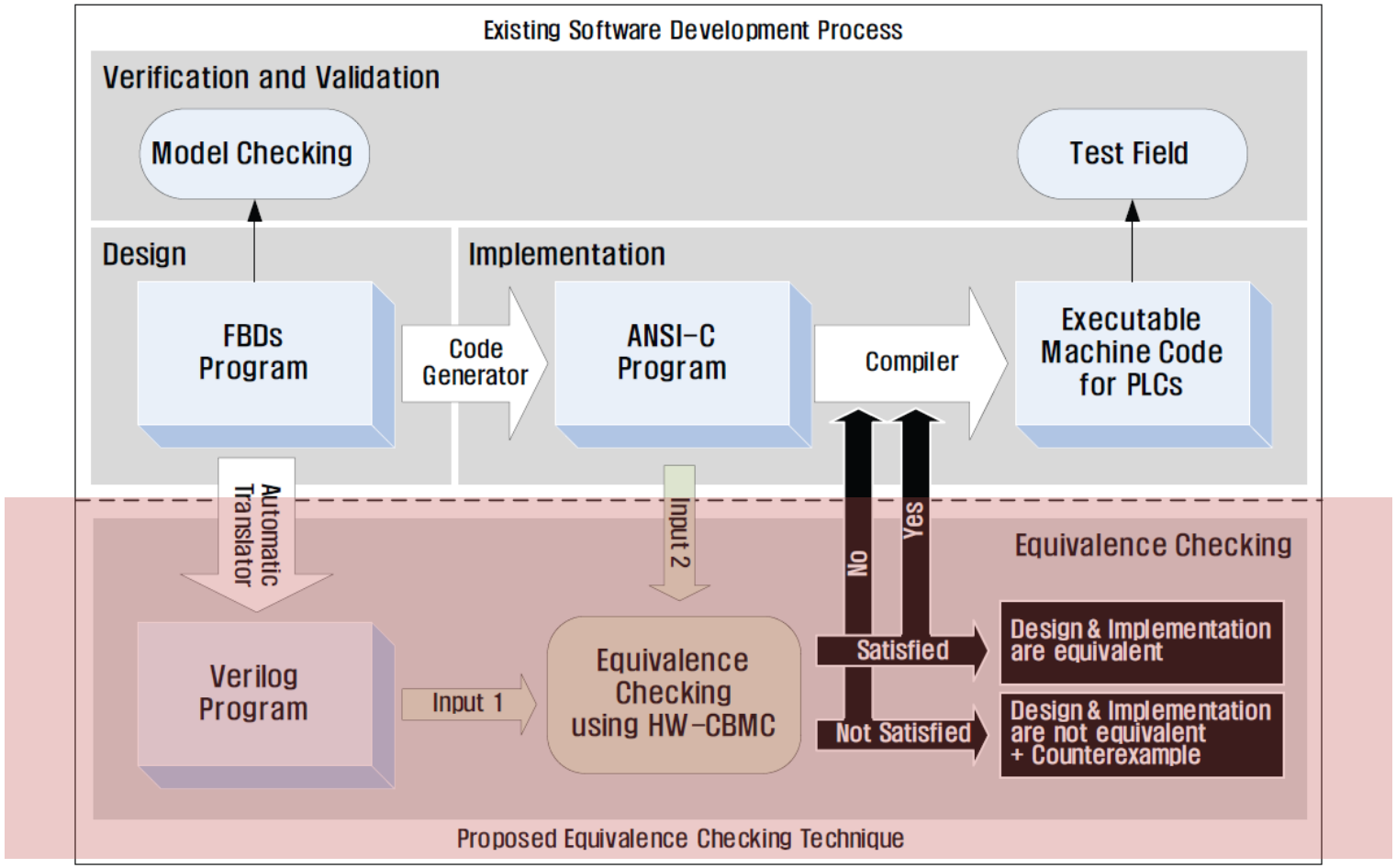
Equivalence Checking



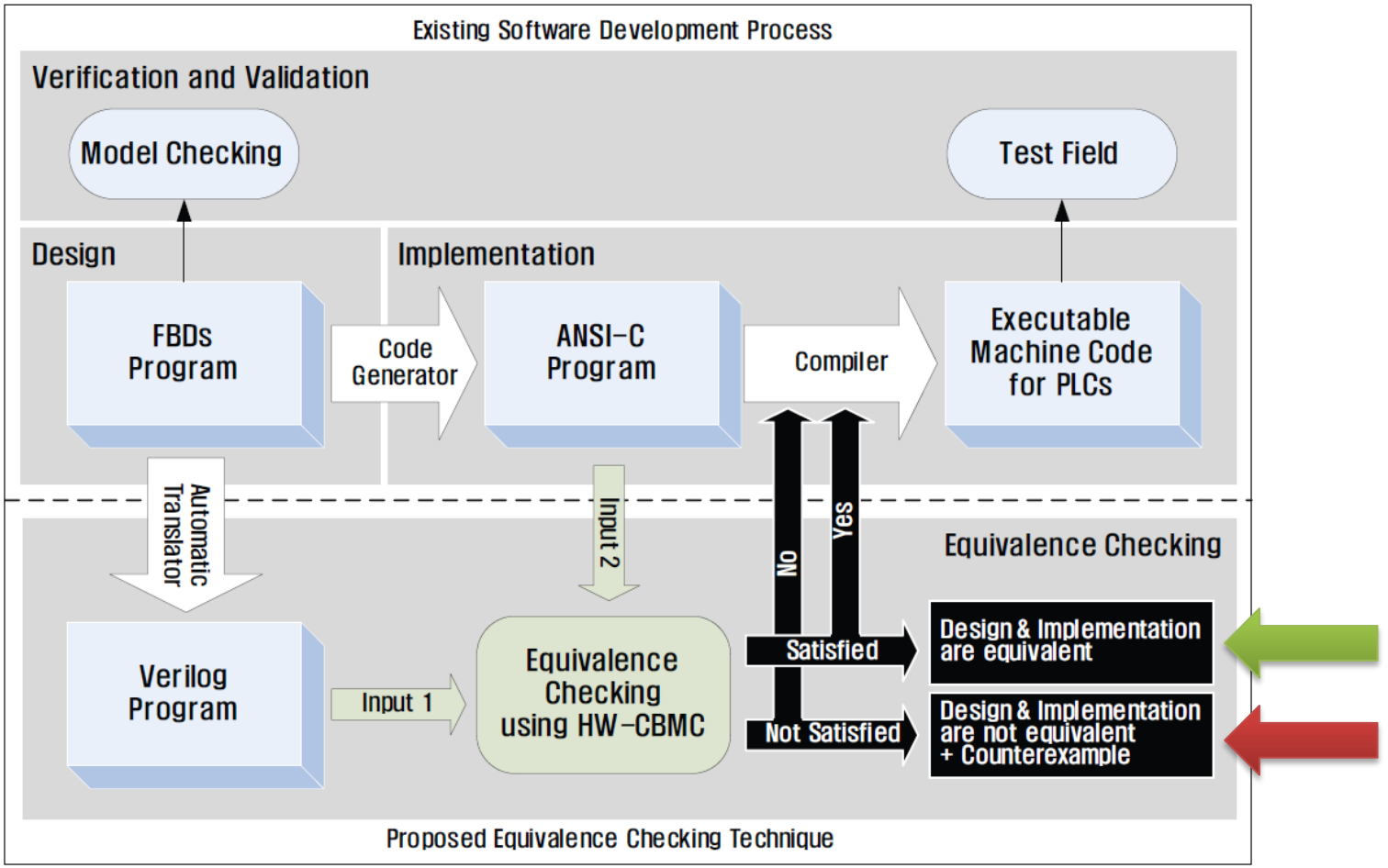
Equivalence Checking



Equivalence Checking



Equivalence Checking



Equivalence Checking (cont'd)

Specific features of Verilog as an input of HW-CBMC

- A name of variable should be different from the name of module which defines and uses it

  → Every modules and variables must have different name

- Function calls are not allowed

  → Every function calls must be translated into module calls

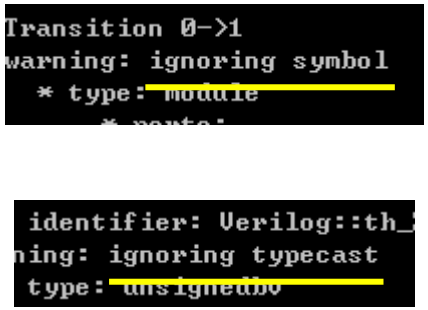
Equivalence Checking (cont'd)

Verilog program for HW-CBMC

```
module main_module(clk, IN, main_module);

        input clk;
        input [0:6] IN;
        output main_module;
        …
        wire Ge_int;
        …
        assign Ge_int = GE_INT(IN, 7'b0011110);
        …
        function GE_INT;
                input [0:6] in1;
                …….
        endfunction
        ….
Endmodule

…
```



Symbol and typecast warning occur if the names of module and variable are same in Verilog program.

Example. Verilog file translated by *FBDtoVerilog 1.0*

Equivalence Checking (cont'd)

Verilog program for HW-CBMC

```
module main_module(clk, IN, main_module);

        input clk;
        input [0:6] IN;
        output main_module;
        …
        wire Ge_int;
        …
        assign Ge_int = GE_INT(IN, 7'b0011110);
        …
        function GE_INT;
                input [0:6] in1;
                …….
        endfunction
        ….
Endmodule

…
```

Function calls in Verilog program are not allowed.

```
Type-checking Verilog::TOF
Type-checking Verilog::main
file th_X_Pretrip_tr_v4.v line 25: function call not allowed here
CONVERSION ERROR
```

Example. Verilog file translated by *FBDtoVerilog 1.0*

Verilog Program for HW-CBMC (cont'd)

```
module main_module(clk, IN, main_module_out);

        input clk;
        input [0:6] IN;
        output main_module_out;
        …
        wire M_GE_INT_OUT;
        …
        GE_INT = M_GE_INT(IN, 7'b0011110, M_GE_INT_OUT);
        …
        ….
Endmodule

module GE_INT(int1, int2, OUT);

        input [0:6] in1;
        input [0:6] in2;
        output OUT;

        assign OUT = (in1 >= in2);
module
…
```

```
module main_module(clk, IN, main_module);

        input clk;
        input [0:6] IN;
        output main_module;
        …
        wire Ge_int;
        …
        assign Ge_int = GE_INT(IN, 7'b0011110);
```

```
function GE_INT;
        input [0:6] in1;
        input [0:6] in2;
        begin
                GE_INT = (in1 >= in2);
        end
endfunction
```

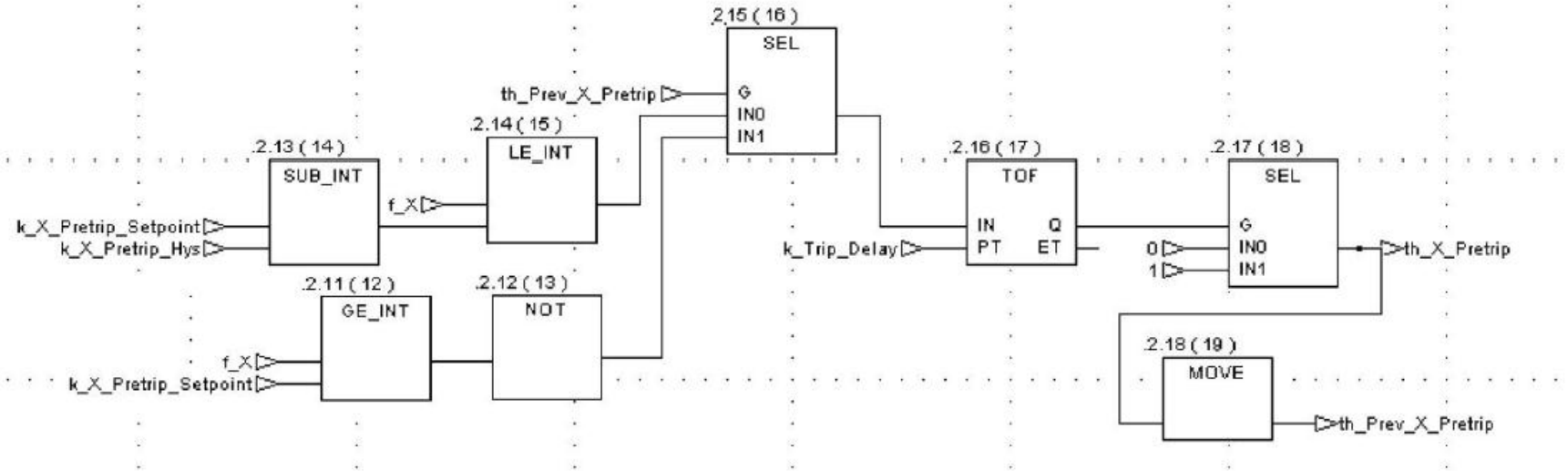
Equivalence Checking
between FBDs and C Programs using HW-CBMC

CASE STUDY

Case study : th_X_Pretrip

- Function block diagram of *th_X_Pretrip*

- One of 18 shutdown logic program in APR-1400 RPS(prototype)
  - Pretrip condition : input value which is over the specific value during a delay time



Case study : th_X_Pretrip (Verilog program)

Automatically translated Verilog program

```
module th_X_Pretrip(clk, f_X,
th_X_Pretrip);
input clk;
input [0:6] f_X;
output th_X_Pretrip;
…
…
…
…
```

Modified Verilog program

```
module th_X_Pretrip(clk, f_X,
th_X_Pretrip_OUT);
input clk;
input [0:6] f_X;
output th_X_Pretrip_OUT;
…
…
…
…
```

Renaming output argument

Case study : th_X_Pretrip (Verilog program)

Automatically translated Verilog program

```
module th_X_Pretrip(clk, f_X, th_X_Pretrip);
…
function GE_INT;
   input [0:6] in1;
   input [0:6] in2;
   begin
      GE_INT = (in1 >= in2);
   end
Endfunction
…
endmodule
```

Function definition

Modified Verilog program

```
module th_X_Pretrip(clk, f_X, th_X_Pretrip);
…
…
endmodule

module GE_INT(in1, in2, OUT);
   input [0:6] in1;
   input [0:6] in2;
   output OUT;
   assign OUT = (in1 >= in2);
endmodule
```

Module definition and argument declaration

Case study : th_X_Pretrip (Verilog program)

Automatically translated Verilog program

```
…
…
assign Ge_int = GE_INT(f_X, 7'b0011110);
…
assign Le_int = LE_INT(f_X, Sub_int);
…
…
…
```

Modified Verilog program

```
…
…
GE_INT M_GE_INT (f_X, 7'b0011110, M_GE_INT_OUT);
…
LE_INT LE_INT(f_x, Sub_int, M_LE_INT_OUT);
…
…
…
```

Case study : th_X_Pretrip (Implementation C program)

Definition of the variables that hold the value of the Verilog module

Condition

Assertion statement

A transition of Verilog program with function **next_timeframe()**

```
1  ...
2  struct module_Pretrip {
3    unsigned int f_X;
4    unsigned int th_X_Pretrip_OUT;
5  };//definition of the variables that holds the value of the Verilog module
6  extern struct module_Pretrip th_X_Pretrip;
7  int main()
8  {
9    unsigned int f_X=0;
10   … //variable declaration
11
12   for(cycle=0; cycle<bound; cycle++)
13   {
14     //synchronizing Inputs
15     f_X = nondet_int(); th_X_Pretrip.f_X = f_X; set_inputs();
16
17     Cond_a_1 = (f_X >= k_Pretrip_Setpoint);
18     Cond_b_1 = ((f_X >= k_Pretrip_Setpoint) && (timer == 5));
19     Cond_c_1 = (f_X <= k_Pretrip_Setpoint – k_X_Pretrip_Hys);
20     th_X_Pretrip_OUT = (state==0 && Cond_a_1)?th_Prev_X_Pretrip:
21       (state==0 && !Cond_a_1)?th_Prev_X_Pretrip:
22       (state==1 && !Cond_a_1 && !Cond_b_1)?th_Prev_X_Pretrip:
23       (state==1 && Cond_a_1 && !Cond_b_1)?th_Prev_X_Pretrip:
24       (state==1 && Cond_b_1)?0:
25       (state==2 && Cond_c_1)?1:th_Prev_X_Pretrip;
26     if(f_X >= k_Pretrip_Setpoint) //count the time unit
27       if(timer == 5) timer == 5;
28       else timer++;
29     else
30       timer=0;
31     //assertion statement
32     assert(th_X_Pretrip.th_X_Pretrip_OUT == th_X_Pretrip_OUT);
33
34     th_Prev_X_Pretrip = th_X_Pretrip_OUT;
35     switch(state) //control a state by conditions
36     {
37       case 0:
38         if(Cond_a_1) state = 1;
39         else state = 0;
40       break;
41       case 1:
42         …
43     }
44     next_timeframe();
45   }
46 }
```

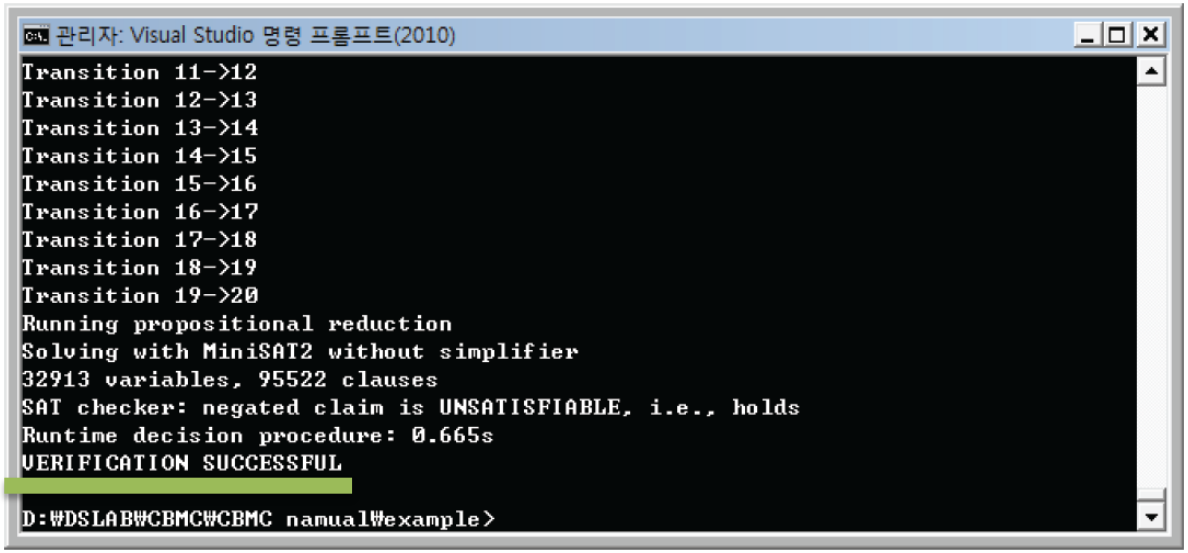
variable declaration

Synchronizing Inputs

Counter

State

Case study : Verification Result



Equivalence Checking
between FBDs and C Programs using HW-CBMC

CONCLUSION AND FUTURE WORK

Conclusion and Future Work

- Equivalence checking approach between design written in FBDs and its implementation program written in ANSI-C using HW-CBMC
- The modification of Verilog program translated automatically by *FBDtoVerilog 1.0* is required to apply the technique
- The code generator is indirectly verified

- Translation rules of FBDtoVerilog will be modified
  - Including Ladder Diagram (LD)
- Verification of equivalence between a design (FBD & LD) and its automatically generated ANSI-C program by pSET
- Development assistant template of ANSI-C program to be appropriate for HW-CBMC

Thank you.

Lee Dong-Ah
KONKUK University
Dependable Software Laboratory

Web : http://dslab.konkuk.ac.kr
E-mail : ldalove@konkuk.ac.kr