KONKUK UNIVERSITY

KNS 2011
2011.05.25 ~ 27

Verification Process of Behavioral Consistency between Design and Implementation programs of pSET using HW-CBMC

Lee Dong-Ah
Dependable Software Laboratory
KONKUK University, Korea

2011.05.26

DEPENDABLE SOFTWARE LABORATORY

Contents

- Introduction
- Background
  - Translation from FBDs into Verilog
  - HW-CBMC
- Verification Process
  - Modification of Verilog
  - Verification using HW-CBMC
- Conclusion

Verification Process of Behavioral Consistency between Design and Implementation programs of pSET using HW-CBMC

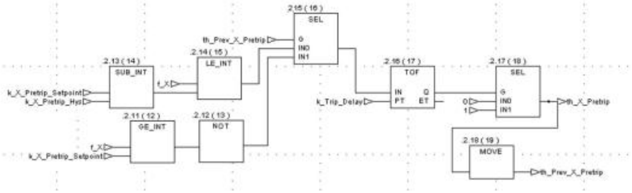
INTRODUCTION

Introduction

- Controllers in safety critical systems often use Function Block Diagrams (FBDs) to design embedded software

- The FBDs are implemented using programming language

- The implementation must have save behavior with the design
  - The behavior should be verified explicitly



Function Block Diagrams

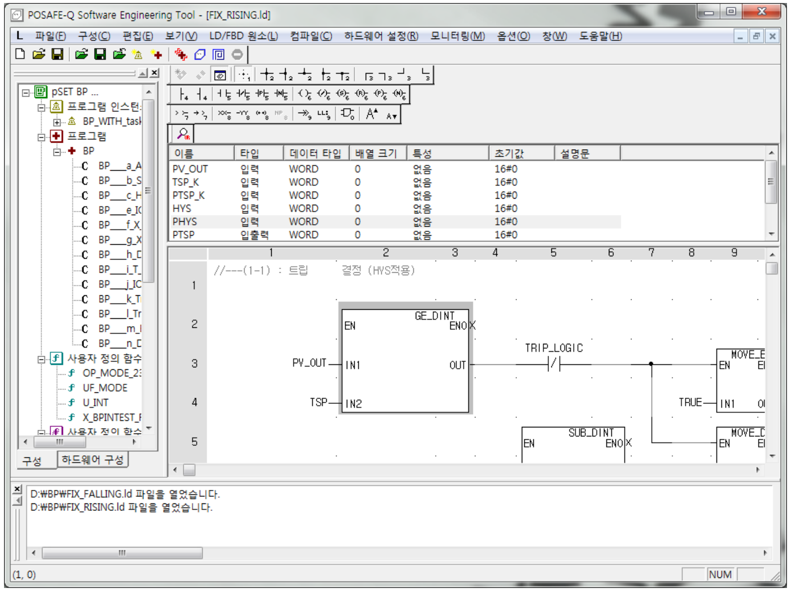
Implementation

Programming Language
ex) ANSI-C

Introduction (cont'd)

- POSAFE-Q Software Engineering Tool (pSET)
  - A part of Korea Nuclear Instrumentation & Control System R&D Center (KNICS) project

- Program POSAFE-Q Programmable Logic Controller (PLC)



이름	타입	데이터 타입	배열 크기	특성	초기값	설명문
PV_OUT	입력	WORD	0	없음	16#0	
TSP_K	입력	WORD	0	없음	16#0	
PTSP_K	입력	WORD	0	없음	16#0	
HYS	입력	WORD	0	없음	16#0	
PHYS	입력	WORD	0	없음	16#0	
PTSP	입출력	WORD	0	없음	16#0	

```
D:₩BP₩FIX_FALLING.ld 파일을 열었습니다.
D:₩BP₩FIX_RISING.ld 파일을 열었습니다.
```

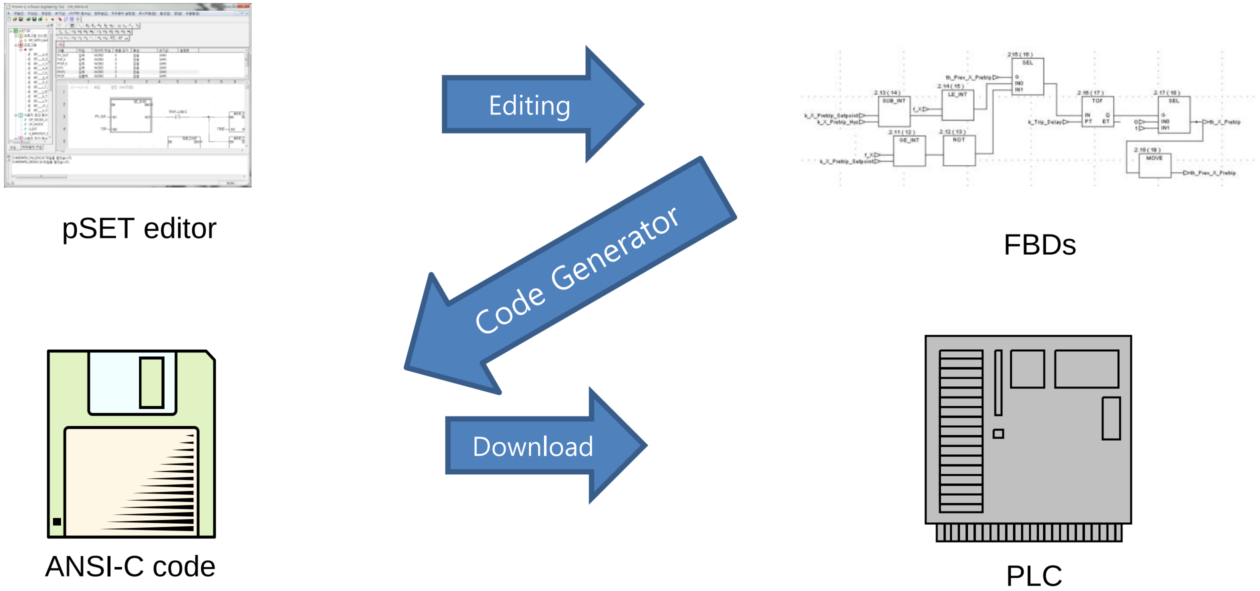
(1, 0)

DEPENDABLE SOFTWARE LABORATORY

Introduction (cont'd)

- The pSET uses the **Function Block Diagram (FBD)**, Ladder Diagram (LD), Sequential Function Chart (SFC) and C Code (CC) to design a program of PLCs
- The pSET uses the ANSI-C program to implement its design

- The automatic code generator generates ANSI-C program with the FBDs



pSET editor

Editing

FBDs

Code Generator

ANSI-C code

Download

PLC

Introduction (cont'd)

- Mathematical proof of code generator can guarantee the equivalence
  - High expenditure
  - Repetitive fulfillment whenever the generator is modified

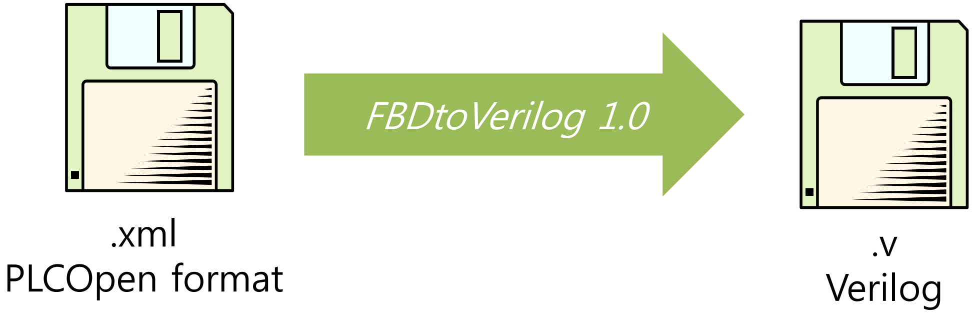
- Equivalence checking using the HW-CBMC
  - The HW-CBMC is formal verification tool
    - Verification of equivalence between hardware and software description
  - The HW-CBMC requires two inputs for the checking
    - Verilog for hardware
    - ANSI-C for software
  - Verification of correctness of the code generator indirectly

Verification Process of Behavioral Consistency between Design and Implementation programs of pSET using HW-CBMC

BACKGROUND

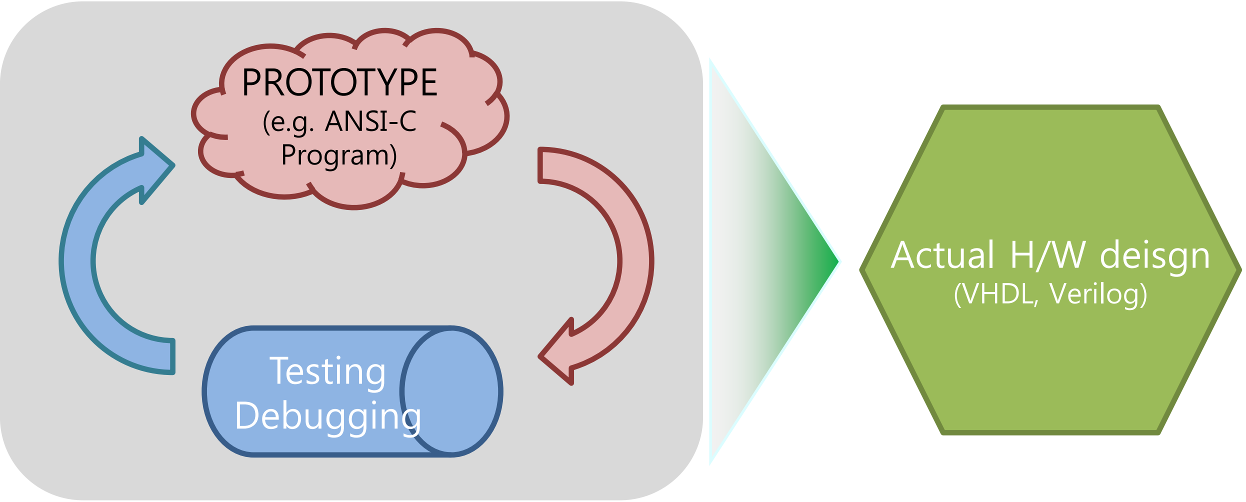
Translation from FBDs into Verilog

- *FBDtoVerilog 1.0* translates FBD program into a semantically equivalent Verilog model
  - Well-formed FBD (IEC 61131-1)
  - XML file with PLCOpen format
  - 7 rules
    - Module Declaration
    - Variable type and size decision
    - Initialization of *reg* variables
    - Output assignment for each wire and output variable
    - Declaration of other module instances
    - Stored value assignment for *reg* variables



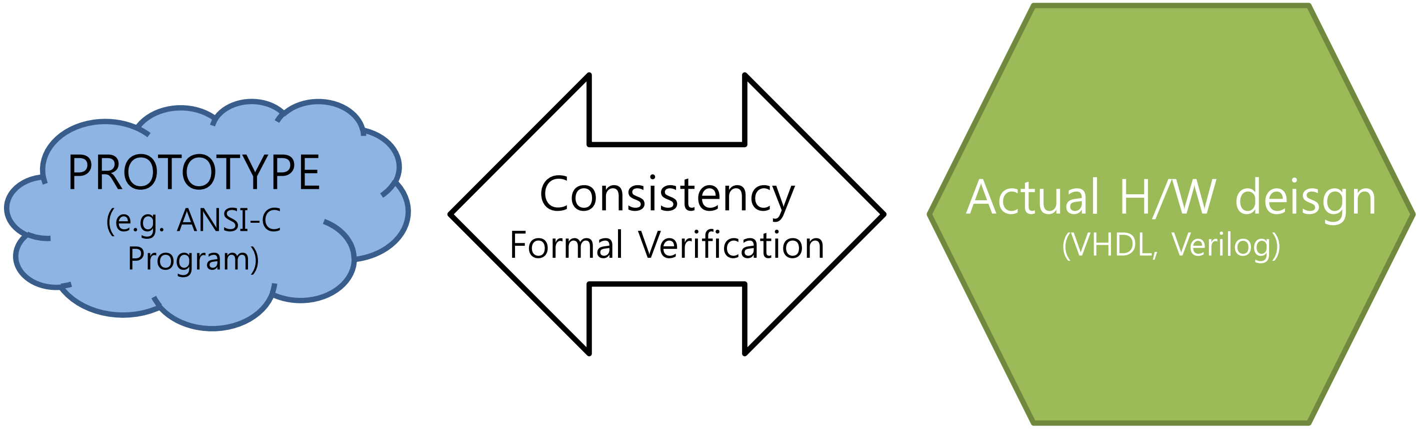
HW-CBMC

- A common hardware design approach is to first write a quick prototype in a language like ANSI-C
  - The ANSI-C implementation is easer to test and debug
- Two implementations of the same design
  - One written in ANSI-C, which is written for simulation
  - One written in register transfer level HDL, which is the actual product



HW-CBMC

- Verification of the consistency of the HDL implementation using the ANSI-C implementation as a reference

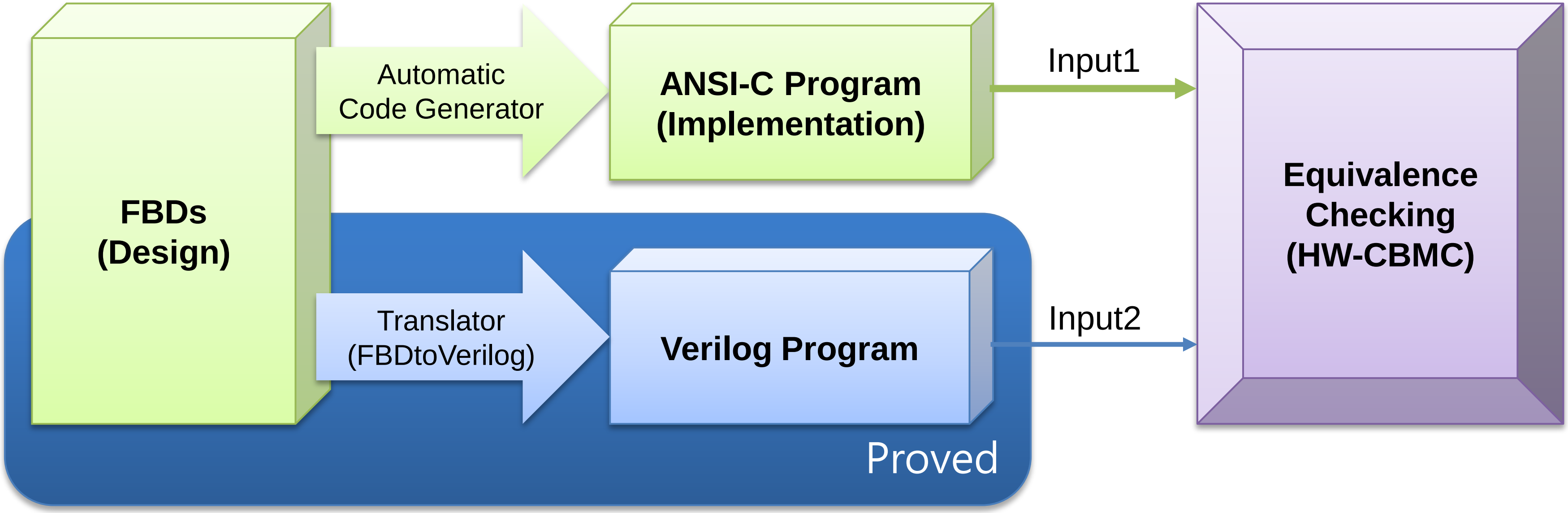


Verification Process of Behavioral Consistency between Design and Implementation programs of pSET using HW-CBMC

VERIFICATION PROCESS

Verification Process

- Equivalence Checking between FBDs and ANSI-C program
  - FBDs translated into Verilog using FBDtoVerilg*
    - Translating FBDs to Verilog is proved in our previous work



*Eunkyung Jee and 6 others, "FBDVerifier : Interactive and Visual Analysis of Counterexample in Formal Verification of Function Block Diagram", JRPIT 2011

Modification of Verilog

Specific feature of Verilog as an input of the 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

Modification of Verilog (cont'd)

Translated Verilog Program	Modified Verilog Program
module module1(clk, In, **module1**); … **output module1**; … **function EQUAL;** **…** **endfunction** … endmodule	module module1(clk, IN, **module1_out**); … **output module1_out**; … endmodule **module EQUAL(in1, in2, out)** **…** **endmodule**

Verification using HW-CBMC

- The ANSI-C program has functions provided by the HW-CBMC
  - *set_inputs()* : synchronize the inputs of ANSI-C and Verilog programs
  - *next_timeframe()* : make transition of Verilog program once
  - *assert(conditions)* : check conditions

- Result of Verification
  - Successful
    - FBDs and ANSI-C programs are equivalent
  - Fail
    - FBDs and ANSI-C programs are not equivalent
    - The HW-CBMC provides counterexample
    - The fail condition is traceable through analysis of the counterexample

CONCLUSION

Conclusion and Future Work

- Verification process of behavioral consistency between design and its implementation using HW-CBMC
  - The design written in FBDs
  - The implementation written in ANSI-C
- The process is applicable to FBDs and ANSI-C program of pSET
  - We can verify correctness of automatic code generator of pSET indirectly

Future work

- New translation rules from FBDs into Verilog
- Experiment on examples designed using the pSET