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

![Diagram showing the relationship between Problems, Systematic Transformation, and Software Implementations]
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

- **Requirements Analysis**
  - FTA, FMEA, HAZOP

- **Design**
  - FTA, HAZOP

- **Implementation**
  - by PLC SW Engineering Tools

**Development**
- SRS
  - Manual Programming
- FBD/LD Program
  - Automatic Translator
- C Program
  - COTS Compiler
- Executable Code for PLC

**Verification**
- Inspection
- Inspection
- Testing
- Simulation / Testing
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

<table>
<thead>
<tr>
<th>Requirements Analysis</th>
<th>Design</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Safety analysis</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FTA, FMEA, HAZOP</td>
<td>FTA, HAZOP</td>
<td>PLC Implementation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C Program</td>
</tr>
<tr>
<td><strong>Development</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRS</td>
<td>FBD/LD Program</td>
<td>COTS SW Engineering Tools</td>
</tr>
<tr>
<td></td>
<td>Verilog / VHDL Program</td>
<td>FPGA Implementation</td>
</tr>
<tr>
<td></td>
<td>Inspection</td>
<td>Verilog</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VHDL</td>
</tr>
<tr>
<td><strong>Verification</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inspection</td>
<td>Inspection</td>
<td>Netlist</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COTS SW Engineering Tools</td>
</tr>
</tbody>
</table>
The Scope of NuDE 2.0

Requirements Analysis
- NuFTA*
- NuFTA*
- FBD FTA*
- NuSRS*
- NuSCRtoFBD*
- FBD Editor*
- NuSCRtoSMV*
- Quick checker
- SMV
- VIS
- SMV
- VIS

Design
- HW-CBMC
- C Program
- FBD FTA*
- FBDtoC*
- FBDtoVerilog*
- FBDtoVHDL*
- FBDtoVerilog*
- Quick checker
- VIS
- BLIF
- BLIF
- EDIFtoBLIF-MV*
- VIS

Implementation
- PLC Implementation
- Executable Code for PLC
- PLC
- COTS SW
- FPGA Implementation
- FPGA
- COTS SW
- FBD Editor*
- FBDtoC*
- FBDtoVerilog*
- FBDtoVHDL*
- Quick checker
- VIS
- BLIF
- BLIF
- EDIFtoBLIF-MV*
- VIS

: Automatic Transformation
An Overview of NuDE 2.0

- **NuDE (Nucler Development Environment) 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

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

FPGA SW Development

Safety Analysis
- NuSCR_FTA
- FBD_FTA

FPGA SW Verification

Supplementary Tools
- Scenario Generator
- FBD Simulator
- FBD-Verilog Comparator
- C Simulator
- FBD-C Comparator
- FBD-Verilog-Netlist-JEDEC Comparator

The scope of this paper!
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)
The NuSCRtoFBD Mechanical Transformation
+ Transforms the NuSCR formal SRS
+ into behaviorally-equivalent FBD programs
+ mechanically
+ Store into an XML file of PLCopen Std.
**FBDtoC** (Ver. 1.0)

The **FBDtoC Mechanical Transformation**
+ Transforms FBD programs
+ into behaviorally-equivalent ANSI-C programs
+ mechanically
+ System/Component/Function units
FBD Tester (of Prof. Gee in KAIST)

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.
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

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

+ Used for the FPGA Synthesis

Verilog Program
FBDtoVHDL

The FBDtoVHDL Mechanical Transformation
+ Transforms FBD programs
+ into behaviorally-equivalent VHDL programs
+ mechanically

+ Used for the FPGA Synthesis

VHDL program

```vhdl
1  entity is
2  port (c: in std_logic; r: in std_logic; reset: in std_logic;
3  forz : in integer range INT_LO to INT_HI;
4  forz2 : in integer range INT_LO to INT_HI)
5  end entity;
6
7  architecture Behavioral of Sample is
8  begin
9  local_CLK_ARRAY : integer range (0 to 2);
10  signal ADD_INT_2_1 : integer range (0 to 2);
11  signal ADD_INT_2_2 : integer range (0 to 2);
12  signal ADD_INT_2_3 : integer range (0 to 2);
13  signal ADD_INT_2_7 : integer range (0 to 2);
14  signal ADD_INT_2_9 : integer range (0 to 2);
15
16  begin
17  ADD_INT_2_1 : ADD_INT_2 part map (0 => LOCAL_CLK_ARRAY(1), reset => c, forz2 => forz);
18  ADD_INT_2_2 : ADD_INT_2 part map (0 => LOCAL_CLK_ARRAY(2), reset => c, forz2 => forz);
19  ADD_INT_2_3 : ADD_INT_2 part map (0 => LOCAL_CLK_ARRAY(3), reset => c, forz2 => forz);
20  process (c)
21  begin
22  if (c = '1') then
23  LOCAL_CLK_ARRAY <= "1000";
24  forz1 <= r;
25  forz2 <= reset;
26  end if;
27  end process;
28  end Behavioral;
```
VERIFICATION PROCESS
Quick Checker

The NuSCR Formal SRS

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

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
FBDtoVerilog 1.0 + VIS & SMV

An FBD Program in the FBD Editor

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

Option Selection

VIS Analyzer

SMV Model Checking
VIS Analyzer (Ver. 3.0)

A Process of Executing VIS & the Execution Result

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

A Model Checking Result (Table + Chart)

Automatic Execution & Analysis

Flowchart

Table
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
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
SAFETY ANALYSIS PROCESS
NuSCR FTA

The NuSCR Formal SRS

The NuSCRtoFT Mechanical Generation
+ Generates a fault tree
+ from an NuSCR formal SRS
+ for a specific (important) output variable
+ Calculates (minimal) cut-sets

+ We need to define the value of an output variable

FT (Fault Tree)
FBD FTA

An FBD Program

The FBDtoFT Mechanical 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

FT (Fault Tree)
SUPPLEMENTARY TOOLS
C Simulator

Internal process

Scenarios

C files → GCC → exe file → Execute → Text file

C Simulator

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

- 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

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

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 !!

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

↓ 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)
{
    /* declare Local variables */
    int HYS = 300, PHYS = 300, RNG_MIN = 600, RNG_MAX = 29400, MAXCNT = 70;
    bool TRUE = 1, FALSE = 0;

    /* declare Temp variables */
    int to_63 = 0, to_58 = 0, to_38 = 0, to_42 = 0, to_48 = 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 translation */
    to_9 = GE_INT(PV_OUT, struct_FIX_RISING.TSP);
    to_31 = AND_BOOL(to_9, 1struct_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, 1OB_INIT_STA);
    struct_FIX_RISING.TRIP_LOGIC = AND_BOOL(to_61, 1OB_INIT_STA);
    struct_FIX_RISING.TRIP_CNT = SEL_INT(to_31, to_48, to_42);

    return struct_FIX_RISING;
}
```

C programs generated
The FBDtoVerilog Transformation

FBDs in the FBD Editor

by FBDtoVerilog 2.0

Verilog programs generated
Scenario Generator
Verilog Simulation with ModelSim
FBD & Verilog Comparator

→ The FBDtoVerilog 2.0 transformation worked well!
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