THE EFFECTIVENESS OF REAL-TIME EMBEDDED SOFTWARE TESTING

MANN1023

ASSIGNMENT 1

PREPARED BY:
MOHD FAIRUZ BIN SHAHADAN  MAN151023
INTRODUCTION

- Embedded software testing is facing increasingly severe challenges with the wide application of embedded software in the fields with high reliability and security.
- A new software testing methods and strategy must consider the features of embedded software.
- Researches on the evaluation of the quality of software testing are still insufficient.
- Software testing is an approach to guarantee software quality, and correctness of software which has been previously tested.
# Features of Embedded Software

<table>
<thead>
<tr>
<th>Strong Specificity</th>
<th>Embedding Feature</th>
<th>Real-Time Feature</th>
<th>Variety of the Realization of the Software</th>
</tr>
</thead>
</table>
| - Embedded software repeatedly carries out one specific task.  
- Provides single function         | - Embedded software runs in a tailor-designed hardware environment.  
- Interacts with the external physical world through devices like sensor and actor.  
- Possibly connect and exchange information with other systems | - The correctness of many embedded systems is decided by the function, behavioural feature of the system, and its time performance.  
- Punctuality and promptness.  
- Single test and strength test should be run on the time response of real-time embedded software. | - Its realization varies with different software  
- Some are supported by operating system, while others operate hardware through assembly language.  
- Embedded software has its own features which pose new challenge to embedded software testing. |
# The Particularity of Embedded Software Testing

## Extremely Strong Pertinence
- Specific to certain type of system or even one typical system,
- It requires auxiliary test tools which are needed to be developed in particular.
- Those tools are hardly applicable in other embedded software testing.

## Test Cases Are Large in Number, and Situation Is Complex
- Cross-linking relationship of an embedded system is complex.
- It has strong reaction and instantaneity, which results in large input scale and other restraints like sequential relationship,
- There is much complexity in input
- The number and quality of test cases should be increased and enhanced in order to run a sufficient test.

## High Dependability on Test Tools and Environment
- Embedded software contains large amount of hardware information.
- Many tests conducted on the software aren’t sufficient before it integrates with hardware environment.
- The real effective test would be the system testing after the integration of embedded system.
Effectiveness is refers to:
- Degree of the completeness of planned activities.
- Achievement of expected results.

Research on the effectiveness of software testing is proceeds from:
- the perspective of software testing quality, and
- its study object is software testing itself instead of software quality.
The objective of software testing is to pin out the potential faults and defects in the software at the lowest time and human resource cost.

The faults existing in the software will be recognized when the test is successful.

The side benefit lies in that it, proved that the function and property of target software conform to requirement statements.

Test result data collected in the test process can provide evidence for reliability analysis.
Evaluation on the effectiveness of software testing includes two aspects:

- **Test sufficiency**
  - Test sufficiency can be conducted through test coverage and evaluation on test case validity, elaboration on which is also the key part of this thesis.

- **Test efficiency**
  - Test efficiency can be guaranteed through management, assessment, and optimization of the planned test process.
Test coverage is one of the most important methods to:

- Conduct qualitative metric and control the process of software test
- Metric approach based on test coverage can be elaborated as conducting statistics on various coverage rates.
- Measuring the adequacy degree of the software testing while aiming to satisfy relevant test adequacy standards.
Test coverage indicators with different focuses are used to measure the adequacy degree of software test from different perspectives, including:

- coverage rates based on code like sentence coverage, branch coverage, path coverage, branch-condition coverage, and combination coverage)
- coverage rates based on requirement (e.g. function coverage, requirement coverage, and interface coverage)

\[
CR = \left( \frac{i_e}{i_t} \right) \times 100\% 
\]

- \(CR\) — Coverage rate;
- \(i_e\) — Number of items executed at least once;
- \(i_t\) — Total item number;
The useful ways to evaluate the completeness of test case and further enhance the coverage rate of software testing is:

- conducting a dynamic test on embedded software.
- acquiring dynamic code coverage rate information of the tested software.
- supplementing test case.

The acquirement of dynamic coverage rate information of tested software is quite difficult. Especially when the tested software is real-time embedded system, precise measurement of time performance information is very attention-catching.
TO ACQUIRE TEST COVERAGE DATA

- The operating time between different functions or modules and random statement blocks in a module should be exactly calculated,
- It would be insufficient to run the test with the assistance of oscilloscope and logic analyzer only,
- Proper software test tools are needed to chosen.
- Simulative test environment and test target board has to be developed when:
  - The measurement couldn’t be continued
  - No test interface is left by the experimenter
CASE STUDY OF REAL-TIME EMBEDDED SOFTWARE

Introduction to the software → Choose test tools → Set up test environment

Collect coverage rate data → Test time performance → Analyze the effectiveness of the test
The target software is real-time embedded software, operating on the hardware platform of microprocessor SMJ320C3X, and functions which have been completed including:

- The function of external communication
- The function of interior communication
- The function of control
- The function of calculation
- The function of data collection

Programming language: hybrid programming of C programming language and DSP assembly language, and the developmental environment is C3X Code Composer.
# CHOOSE TEST TOOL

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TestBed</td>
<td>• an embedded software test tool developed by LDRA Company</td>
</tr>
<tr>
<td>RTInsight</td>
<td>• a real-time hardware data collection tool auxiliary to TestBed</td>
</tr>
<tr>
<td>RTInsightPro</td>
<td>• a software to analyse test result and also display results</td>
</tr>
</tbody>
</table>
Develop test target board based on DSP SMJ320C3X,

set up dynamic test platform of embedded software,

run coverage test, time performance test and fault model test to embedded software on the platform,

To collect and receive characteristic data written to specific address in the program, we need to:

- link data bus of target system with address bus,
- link chip select signal and write signal correspondent to monitored address,
- grounded signal of the target system.
SCHEMATIC DIAGRAM OF SIMULATIVE TEST ENVIRONMENT
COLLECT COVERAGE RATE DATA

- Hardware linkage
- Source Code Instrumentation
- Download program
- Set up project

Data storage

Figure 2. Dynamic coverage chart
TEST TIME PERFORMANCE

1. Initiate RTInsight Pro, set up a project, and select a function to conduct time performance analysis;
2. Conduct performance instrumentation to the tested codes with relevant template;
3. Compile the inserted codes and convert them into target board binary code, and program the binary code into the program storage of the target board;
4. Run the tested program, and the measured value of time performance of concerned functions is shown on the interface of RTInsight Pro of host emulator.
ANALYSE THE EFFECTIVENESS OF THE TEST

Check code coverage result after execution of designed test

• Statement coverage 85.5%,
• Branch coverage 67.7%,
• Call coverage 92.1%,

Check code coverage result after the execution of newly added test case

• Statement coverage 98.3%,
  Branch coverage 96.4%,
• Call coverage 98.9%.
Conclusions

Relevant researches need to combine many test coverage data and manage many data in the test process,

- Based on the particularity of real-time embedded software test.
- Method of test coverage to evaluate the effectiveness of real-time embedded software testing.
- Control software in the subsystem of a shuttle program of certain type as a case to practice.