임베디드 SW의 블랙박스 테스트를 위한 검증 모듈의 디자인 및 구현

김범모° 백창현 장중순 정기현 최경희 박승규

아주대학교 정보통신전문대학원 아주대학교 산업시스템공학부 아주대학교 전자공학부
{masque°, labmedia, jsjang, khchung, khchoi, sparky}@ajou.ac.kr

A Design and Implementation of the Check Module for the Test of Embedded Software

Beommo Kim° Changhyun Baek, Joongsoon Jang
Gihyun Jung, Kyunghee Choi, Seungkyu Park

Graduate School of Information and Communication, Ajou University
School of Electronics Engineering, Ajou University
industrial and Information Systems Engineering, Ajou University

요 약

최근 개발되는 임베디드 시스템의 경우 하드웨어와 소프트웨어의 구조가 매우 복잡해짐에 따라, 시스템에 탑재되는 소프트웨어의 신뢰성 확보를 위한 테스트 절차가 요구되고 있다. 특히 시스템에 탑재되는 소프트웨어는 다중 함수에 의해 의사결정이 되면서, 시스템 디자인 단계에서 요구되는 스펙(Specification)을 만족하지 못하는 경우가 빈번하게 발생한다. 본 논문에서는 임베디드 소프트웨어의 자동화된 테스트를 위해 요구되는 검증 모듈을 디자인하고 구현하였다. 검증 모듈은 요구사항 기반으로 설계되었으며, 각각의 요구사항을 만족하는 검증 모듈을 구현하여 실제 상용화 제품에 대한 테스트를 진행하였다.

1. 서 론

현재 빠르게 성장하고 있는 임베디드 S/W(embedded software) 산업은 기술 혁신과 시장에서의 치열한 경쟁에 의해 새롭고 역동적인 양상으로 전개되고 있다. 임베디드 시스템(embedded system)은 일반적인 목적보다는 특수한 애플리케이션용으로 설계되고 있다. 당초 산업용 기기를 제어하기 위해 사용되던 임베디드 S/W는 공장 자동화 및 가정 자동화에 필요한 자동제어 시스템을 비롯해 각종 디지털 정보가전 기기, 자동 센서장비에 이르기까지 그 사용범위와 영향력이 점점 커지고 있다.

점차 복잡해져 가는 임베디드 S/W의 신뢰성 확보를 위한 테스트 절차가 요구되고 있다. 그러나 일반적으로 시행하는 매뉴얼 테스트(Manual Test, 테스터가 물리적인 입력을 시스템에 제공하여 결과를 관찰)의 경우 요구되는 시간 비용과 테스트 자체의 낮은 신뢰성 등과 같은 제약을 가지고 있으며, 이에 따라 자동화된 테스트 기법에 대한 연구가 활발히 진행되고 있다.

본 논문에서는 자동화된 테스트를 수행하는 테스트 소프트웨어에서 요구되는 모듈 중 하나인 검증 모듈을 디자인하고 구현하였다. 논문에서 구현된 검증 모듈은 FSM(Finite State Machine) 모델 기반의 스펙 분석과 블랙박스 테스트 하의 테스트 구조에서 작동한다.

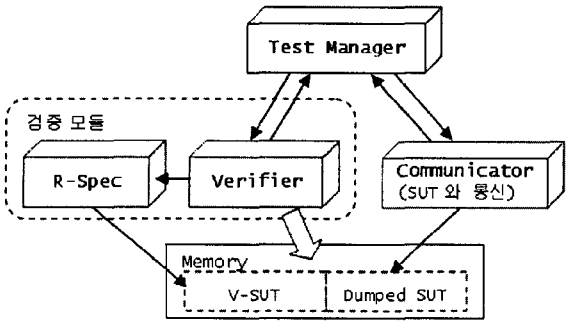
본 논문의 구성은 다음과 같다. 2절에서는 본 논문에서 구현한 검증 모듈을 포함하는 테스트 소프트웨어의 구조를 기술하고, 3절에서는 검증 모듈의 기능적 요구사항을 기술하였으며, 4절에서는 요구사항 기반으로 설계되고 디자인된 검증모듈에 대해서 설명하였다. 마지막으로 5절에서는 구현된 모듈을 이용한 테스트 사례를 소개하고, 6절에서는 향후 과제에 대해서 기술한다.

2. 테스트 소프트웨어(Test Software)의 구조

테스트 소프트웨어는 테스터(Tester)에 의해 미리 생성된 시나리오를 바탕으로 SUT(Software Under Test)의 소프트웨어 신뢰성을 평가한다. 본 논문에서 구현한 테스트 소프트웨어는 임베디드 소프트웨어(Embeded Software)의 자동화된 신뢰성 평가를 목적으로 하며, 그림1과 같은 구조를 갖는다. 자동화된 테스트 소프트웨어를 이용한 테스트 절차는 테스트 시나리오(Test Scenario) 생성, 시나리오 수행, 결과 검증으로 구성된다.

테스트 시나리오 생성은 테스트 매니저(Test Manager) 모듈이 담당하며 SUT의 오류 여부 검증을 위한 입력값(input Value)의 연속(series)을 생성한다. 테스트 시나리

오는 사용자가 스펙을 분석하여 입력한 FSM 구조체 정보를 기반으로 생성된다.



[그림1] 테스트 소프트웨어 구조

SUT의 테스트 시나리오 수행은 테스트 엔진이 COMMUNICATOR 모듈을 SUT와의 인터페이스(Interface)로 이용한다.

테스트와 관련된 전반적인 흐름제어는 TEST MANAGER에 의해 결정된다. 시나리오를 기반으로 생성된 일련의 입력값은 순차적으로 COMMUNICATOR와 검증 모듈에 전달된다. COMMUNICATOR는 미리 정의된 프로토콜에 따라 SUT에 입력 데이터(센서값이나 스위치 이벤트)를 전달하고, SUT는 전달 받은 입력값을 바탕으로 의사결정을 한 후 출력값(Output Value)을 생성한다.

SUT의 SW 오류 여부에 대한 판단은 검증 모듈이 담당한다. 검증 모듈은 테스트 시나리오에 따라 Test Oracle(TO, 테스트 오라클)을 생성하고 SUT가 결정한 출력값과 TO을 비교하여 SUT의 정상 동작 여부를 판단한다.

3. 검증 모듈의 요구사항

검증 모듈은 테스트 시나리오 수행과정에서 TO을 생성하고, 결과 검증 단계에서 TO과 SUT의 출력값을 비교하여 SUT의 SW 오류를 찾는다.

임베디드 시스템을 테스트 하기 위해서 검증 모듈은 임베디드 소프트웨어가 갖는 특성을 고려하여 설계되어야 하며, 아래와 같은 요구 사항을 만족해야 한다.

3.1 실시간 Test Oracle 생성

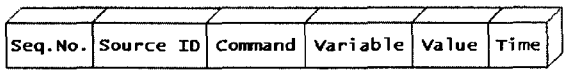
검증 모듈은 사용자가 입력한 스펙을 이용하여 실시간으로 TO을 생성해야 한다. 스펙은 SUT의 기능적 요구사항을 명시하고 있으므로, 이를 이용하면 SUT의 정상 동작 결과값을 예측할 수 있다. 이 때 Test Oracle은 실시간으로 생성해야 하며, 이는 임베디드 소프트웨어의 경우 이벤트 간의 작은 시간차에 의해 의사결정이 다르게 진행되기 때문이다. 따라서 검증 모듈의 경우 SUT에 전달되는 이벤트와 동일한 시간 정보를 바탕으로 Test Oracle을 생성해야 한다.

3.2 Test Oracle 정확성 보장

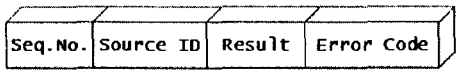
검증 모듈에서 생성한 Test Oracle은 SUT의 초기상태(시나리오에 의한 입력이 시작되는 상태)를 바탕으로 생성되어야 한다. SUT는 동일한 입력값을 받은 경우라도 입력 받을 당시 SUT내 다른 변수들의 상태에 의해 다른 출력값을 생성할 수 있다. 따라서 만약 SUT와 검증모듈이 서로 다른 상태에서 출발한다면, SUT에 전달된 동일한 입력값을 이용하여 생성한 TO이 테스트 시나리오 수행 후의 SUT의 올바른 output값을 나타내고 있음을 보장 할 수 없다.

3.3 SW 오류 재현 기능

검증 모듈은 발견한 오류를 재현할 수 있어야 한다. 소프트웨어를 테스트하는 목적은 소프트웨어에 존재하는 오류를 찾고 수정하기 위함이다. 발생한 오류의 문제 파악을 위해, 검증 모듈은 SUT의 동작 오류를 판단한 경우 SUT에 입력된 일련의 입력값과 검증 단계에서 SUT의 의사결정값과 TO 사이의 차이 등을 포함해야 한다. 사용자는 오류발생 당시의 SUT의 정보를 이용하여 오류를 재현해 볼 수 있으므로 문제 파악에 도움이 되며, 오류 수정 후 수정 여부를 테스트 할 수 있다.



메시지 구조 I (Test Manager -> 검증 모듈)



메시지 구조 II (검증 모듈 -> Test Manager)

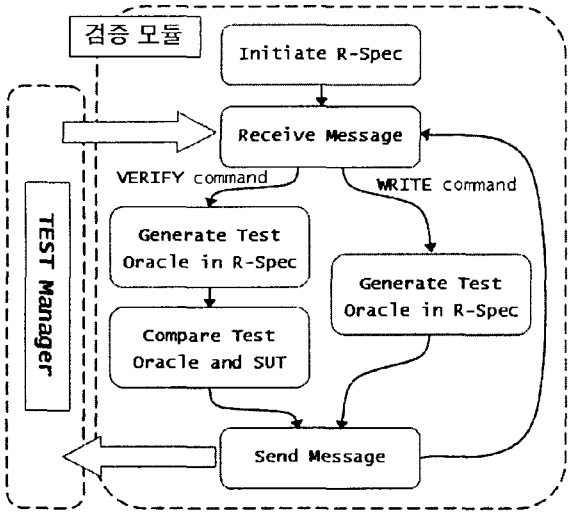
[그림2] Message Structure

4. 검증 모듈의 구조

검증 모듈은 크게 R-Spec (Reformed Spec) 과 VERIFIER의 두 부분으로 나뉜다. R-Spec은 Spec 기반의 TO을 실시간으로 생성하고, VERIFIER는 검증 작업 전반을 관리 한다.

실시간 TO생성을 위해 TEST MANAGER는 시나리오 상의 입력값을 SUT와 검증 모듈에 보낸다. TEST MANAGER와 검증 모듈 사이의 통신은 메시지를 이용하며, 메시지의 구조는 그림 2와 같다.

TEST MANAGER는 WRITE와 VERIFY의 두 개의 명령어를 이용하여 검증 모듈의 작업 수행을 지시할 수 있으며, WRITE 명령을 이용하여 입력값의 변수명과 값을 Variable과 Value 필드를 이용하여 검증 모듈에 전달한다. VERIFY 명령은 검증 모듈에서 생성한 TO과 SUT output의 비교 작업 수행을 지시한다. Time 필드는 해당 입력이 이루어진 시간 정보를 ms단위로 전달한다.



[그림3] 검증 모듈 구조

검증 모듈은 TO의 정확성 보장을 위해 TO과 SUT가 내부 변수 상황이 동일한 상태에서 시작함을 가정하고 있다. 이를 위해 Spec은 SUT의 초기화 상태를 기술하고 있어야 하며, 테스트 시나리오는 항상 처음 시작에 SUT를 초기화 하는 작업을 수행해야 한다. 초기화 작업이 수행될 때에는 TO도 초기화 하여 테스트 시나리오 수행 후 TO이 SUT와 동일한 변수 상태를 가지고 있도록 하였다.

검증 모듈은 S/W 오류 발생 재현을 위해 보고서를 이용한다. 보고서는 Test Manager가 SUT와 검증 모듈로 보낸 시나리오 상의 input값을 순서대로 나타내고 있다. 이를 이용하여 테스터는 초기 상태로부터 오류 발생 상황을 재현할 수 있다. 또한, VERIFY 시의 SUT의 output 값과 Test Oracle의 값을 나타내어 오류 발생 원인을 발견하는데 도움을 줄 수 있도록 하였다.

5. 테스트 진행 결과

본 논문에서 구현한 검증 모듈의 성능 평가를 위해 이미 매뉴얼 테스트를 진행한 상용 제품에 대한 SW 신뢰성 테스트를 진행하였으며, 테스트 진행은 그림4와 같이 였다. SUT는 여러 개의 센서와 스위치, 엑츄에이터로 구성되었으며, 냉난방을 제어하는 시스템이다. 동일 기능의 두 개의 시스템을 바탕으로 각각 76800개와 22042개의 테스트를 수행하였다.

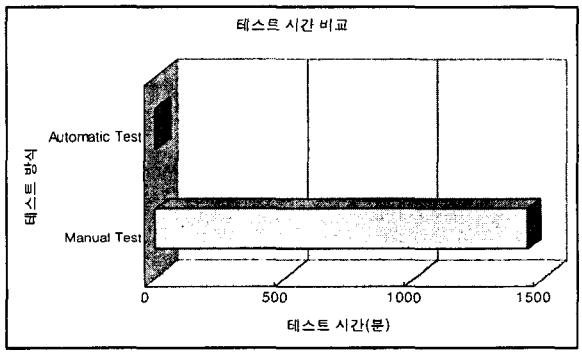
테스트에서 발견된 SW 오류는 총 10개였으며, 이는 모두 상용화 이전에 수정 보완되었다. 표1은 테스트 결과 발생한 SW 오류를 중요도에 따라 구분한 결과이다.

테스트 결과에서 MAJOR 항목의 오류는 해당 시스템의 의사결정이 TO과 전혀 다르게 나타나는 경우이며, MINOR인 경우에는 순간적으로 오작동 할 수 있으나 곧 정상적인 제어 상태로 복귀한 경우이다. SPEC ERROR는 정의된 스펙을 해당 시스템의 프로그래머가 다르게 이해하여 발생한 부분을 나타낸다.

구분	테스트 케이스	중요도기준		
		MAJOR	MINOR	ETC.
시스템1	76800	2		
		5	1	1
시스템2	22042			
				1

[표1] 테스트 진행 결과

또한 테스트 자동화로 인한 테스트 시간은 그림4와 같다. 본 논문에서 구현한 검증 툴을 이용한 경우 100개의 테스트 케이스를 실행하는데 약 9분이 소요되며, 매뉴얼 테스트의 경우에는 테스트 정확도를 위한 반복 테스트로 인하여 약 24시간이 소요되는 결과를 얻을 수 있었다.



[그림4] 테스트 성능 비교

6. 결론 및 향후계획

본 논문에서 구현한 검증 모듈은 테스트 결과에서와 같이 기존의 매뉴얼 테스트를 진행한 제품임에도 불구하고 여러 개의 SW 오류를 발견하는 성능을 보았다. 또한 테스트 시간 비교에서 알 수 있듯이 테스트 자동화 툴을 도입으로 많은 시간 비용을 절감할 수 있음을 보았다.

그러나 검증 모듈의 TO 계산에 요구되는 시간 동안을 대기하여 전체적인 테스트 시간을 지연시키는 결과를 나타냈다. 이는 향후 개선될 내용이다.

[참고 문헌]

[1] Bart Broekman and Edwin Notenboom, "Testing Embedded Software", Addison Wesley, 2003.

[2] Douglas Hoffman, "Using Test Oracles in Automation", Pacific Northwest Software Quality Conference, 2001.