技術論文 DOI:http://dx.doi.org/10.5139/JKSAS.2012.40.6.548

NEXUS를 이용한 비행안전 필수 소프트웨어 검증환경 개발

윤형식*, 한종표*

Development of Verification Environment for Flight Safety Critical Software using NEXUS

Hyung-Sik Yoon* and Jong-Pyo Han*

ABSTRACT

Verification and validation of operational software of the flight control computer, which is flight safety critical, is very important to prove correctness and faultness of the software. To verify the real-time softare requirement on operational software of flight control computer, real-time software internal parameter and variable monitoring technics on hardware-in-the-loop environment, similar to on-flight environment, is required. This paper describes flight safety critical software validation and verificiation environment using standard debugging interface, NEXUS 5001.

초 록

항공기 비행조종컴퓨터 탑재 소프트웨어는 항공기의 손실뿐만 아니라 조종사의 생명에도 직결되는 비행안전 필수 소프트웨어로서 정확성과 무결성을 입증하기 위한 검증 및 확인(Verification & Validation)이 요구된다. 특히 비행안전 필수 소프트웨어는 검증을 위해 통합시험 단계에서 실제 환경과 유사한 운용 환경에 노출하여 소프트웨어 내부 상태를 실시간으로 모니터링하기 위한 환경이 요구된다. 본 논문에서는 표준 디버깅 인터페이스인 NEXUS 5001을 이용한 비행안전 필수 소프트웨어의 검증환경 개발에 대하여 기술하였다.

Key Words : Flight Control Computer(비행조종컴퓨터), OFP(비행운용프로그램), Flight Safety Critical Software(비행안전 필수 소프트웨어), Verification & Validation(검증 및 확인)

Ⅰ. 서 론

항공기에 탑재되는 비행조종컴퓨터는 비행조종시스템의 센서들로부터 신호를 입력받아 항공기의 조종면을 제어하는 장비로써, 비행조종컴퓨터의 고장은 항공기의 손실뿐만 아니라 조종사의 생명에 직접적인 영향을 준다. 따라서, 비행조종컴퓨터에 탑재되는 비행운용프로그램(OFP : Operational Flight Program)은 고신뢰성이 요구되는 비행안전 필수(Flight Safety Critical) 소프트웨어로서, 설계 및 검증단계에서 소프트웨어의 신뢰도와 안전성을 확보하기 위한 기술과 경험이 요구된다. 비행운용프로그램이 요구사항을 충족함을 확인하기 위해서는 비행조종컴퓨터에 대한 독립개체 검증 및 확인시험, 시스템 통합검증 및 확인시험, 그리고 결함모드 검증 및 확인시험을 수행한다. 상기와 같은 검증 및 확인시험을 수행하기 위해서는 비행운용프로그램이 탑재된 비행조종컴퓨터의 외부 입출력 신호를 실시간으로 모사하여 운용하면서 소프트웨어 요구도 검증을 수

† 2012년 2월 19일 접수 ~ 2012년 6월 1일 심사완료
* 정회원, 국방과학연구소
교신저자, E-mail : jongpyohan@nate.com
yjw0210@empal.com
대전광역시 유성구 북유성대로 488번길 160

행할 수 있어야 하며, 장비의 입출력신호 및 소프트웨어 내부 변수의 상태를 실시간으로 모니터링하기 위한 자동화된 검증환경이 요구된다.

기존에도 고신뢰성 소프트웨어의 검증을 위하여 다양한 검증방법이 제안되었으며, 그중 일부 방법의 예는 다음과 같다.

먼저 소프트웨어 실시간 모니터링환경을 구현하기 위하여 프로세서와 메모리의 버스신호를 모니터링하는 방법이 제안되었는데[1], 이는 탐침효과(intrusive)없이 소프트웨어의 모니터링이 가능하나 해당 타겟 장비의 하드웨어 전용 모니터링 장비를 제작하여야 하며, 타겟 프로세서의 고속화에 따른 신호모니터링이 어려운 단점이 있다. 다음은 RTOS기반의 디버그 에이전트를 이용하여 소프트웨어 상태를 모니터링하는 방법이 제안되었으며, 이는 상용 OS에서 제공하는 디버그 에이전트를 사용하거나[2], 전용의 RTOS를 제작하여 모니터링을 수행하였다[3]. 그러나 이 방법은 모니터링에 따른 탐침효과가 발생되어 시험대상 소프트웨어의 정상적인 운용에 영향을 미치며, 특정 OS에 종속된 검증장치라는 단점이 있다. 그리고 타겟 장비에 소프트웨어의 내부 모니터링을 위한 추가적인 하드웨어 회로를 구현하여 모니터링을 수행하는 방법들이 제안되었으며[4], 이 방법은 모니터링회로를 FPGA로 제작하여 재사용 가능하게 만들 수 있는 장점이 있으나, 하드웨어 종속적이며 타겟 프로세서가 고속화되어감에 따라 고속 신호 모니터링이 어렵다는 단점이 있다.

이에 따라 본 논문에서는 임베디드 시스템의 소프트웨어 실시간 모니터링 방안으로 디버깅 표준 인터페이스 중 하나인 IEEE-ISTO NEXUS 5001을 이용한 비행안전 필수 소프트웨어의 검증환경 개발방안을 제안하고자 한다.

II. 소프트웨어 검증환경 개발

2.1 NEXUS 5001 개요

NEXUS 혹은 GEPDIS(Global Embedded Processor Debug Interface Standard)라고 불리는 IEEE-ISTO 5001(NEXUS 5001)은 고속, 다기능의 데이터 송수신 기능을 보유한 디버그 인터페이스를 제공하기 위해 제정된 산업표준이다[5]. NEXUS 5001 표준에는 임베디드 시스템 설계를 위하여 사용되어지는 디버깅 툴 및 개발을 위한 마이크로 프로세서 온-칩 디버깅 특성, 프로토콜 및 인터페이스에 대한 명령어를 정의하고 있다. NEXUS 5001은 디버깅 요구사항과 기능에 따라 NEXUS Class 1에서 Class 4로 분류하고 있으며, 상위 Class는 하위 Class의 기능을 포함한다. 각 Class에 따라 제공되어지는 주요 기능은 Table 1과 같다.

NEXUS 5001 인터페이스는 Fig. 1과 같이 프로세서 코어 주변에 IP 로직으로 존재하는 디버그 인터페이스로서 IEEE 1149.1 표준 JTAG 인터페이스를 통해 실행/실행 추적(Trace) 제어를 수행하고, 추가로 구성된 NEXUS IP Core와 연동되는 보조 포트를 이용하여 임베디드 TRACE 기능, 정지점(Breakpoint) 제어기능, 디버거 전용 레지스터기능을 수행하는 방식이며, Trace기능을 위한 디버그 전용 메모리가 추가되어 있다[6]. 이와 같이 프로세서 내에 디버그 지원 인터페이스가 구성되어 있으므로 프로세서를 정지하지 않고 실시간으로 프로그램과 데이터의 Trace를 수행할 수 있고 메모리에 접근이 가능하다. 따라서

Table 1. NEXUS 5001 Class별 기능표

구 분	주 요 기능
Class 1	IEEE 1149.1 제공 기능 (디바이스 인식, watch point, Single Stepping, Breakpoint, 정적 메모리와 입력/출력 접근)
Class 2	프로세서 실시간 Trace 관련 기능 (명령어 Trace, 프로세스 모니터링)
Class 3	데이터, 메모리, 입력/출력 읽기/쓰기 명령 Trace기능
Class 4	프로세서 직접 제어기능 (메모리 및 입력/출력 재배치, NEXUS포트를 통한 프로그램 실행)

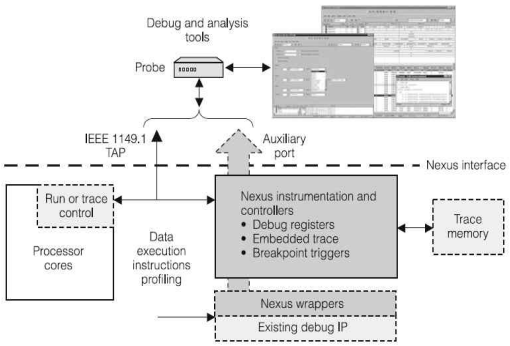

Fig. 1. NEXUS 5001 구성도

NEXUS 5001을 이용하여 프로세서를 멈추지 않고 실시간으로 프로그램 Trace, 데이터 Trace, 메모리 읽기/쓰기, 레지스터 읽기/쓰기 기능을 수행하여 내부 정보를 모니터링 함으로써 탐침효과 없이 소프트웨어 실시간 검증이 가능하다.

현재 Freescale MPC500, MPC5000 계열을 포함한 PowerPC 계열 및 Q4080/4040 멀티코어계열에서는 NEXUS Class 3+를 지원하고 있으며, ARM 계열, TI OMAP, Freescale Coldfire 등 다양한 제조사에서 NEXUS 5001에 대한 지원을 확대하고 있는 추세이다.

2.2 소프트웨어 검증환경 개요

항공기에 장착되어 항공기의 제어를 수행하는 비행조종컴퓨터와 같은 안전필수장비에 탑재되는 소프트웨어는 비행 안전성을 보장하기 위하여 안전성과 신뢰성이 요구된다. 따라서 요구사항 충족 여부와 설계 사항에 대한 검증을 위해 개발된 소프트웨어에 대해 다양한 시험을 수행한다. 시험 단계들 중 하드웨어와 소프트웨어를 통합하여 Open-Loop, 정적 환경에서 소프트웨어 요구사항을 검증하는 하드웨어-소프트웨어 통합시험 단계에서는 하드웨어상에서 구동되는 소프트웨어의 기능적 요구도 검증을 실시간으로 수행하기 위한 비행조종컴퓨터 비행운용프로그램의 실시간 검증환경이 필수적으로 요구된다. 고신뢰성 소프트웨어를 위한 검증환경의 기본 요구조건으로는 비행체에서 제공되는 3중의 외부신호를 시험대상장비인 비행조종컴퓨터의 운용 주기(최대 20ms)에 맞게 입력 가능하여야 하고, 장비의 정상 운용시에 비행조종컴퓨터 내부 메모리와 레지스터에 접근하여 장비의 정상 동작을 방해하지 않으면서 소프트웨어 내부 상태 및 데이터의 읽고 쓰기가 가능하여야 한다. 또한 시험 대상장비의 제어를 위하여 에뮬레이터의 기능이 요구된다. 이러한 요구조건들을 만족하기 위하여 설계된 검증환경의 구성은 Fig. 2와 같다. 본 논문에서 제안된 검증환경은 검증장치를 통제하고 시험 스크립트를 작성하는 호스트 컴퓨터, 비행조종컴퓨터의 입력신호를 모사하고 출력신호를 모니터링하는 외부신호 모사장치 및 탑재 소프트웨어의 내부 데이터를 실시간으로 모니터링하기 위한 실시간 모니터링장치로 구성되며, 각 구성장치의 상세 내용은 다음의 각 절에서 소개한다.

2.3 호스트 컴퓨터 설계

호스트 컴퓨터는 비행운용프로그램 검증환경 전체를 통제하면서 자동화된 스크립트 기반에서

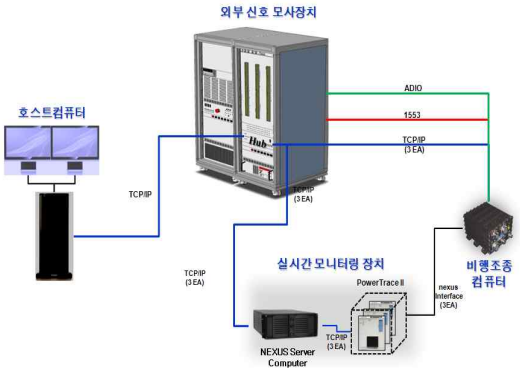

Fig. 2. 비행운용프로그램 검증환경 구성도

비행조종컴퓨터의 통합시험을 수행할 수 있는 시험환경 인터페이스를 제공한다. 호스트 컴퓨터 하드웨어는 워크스테이션급 컴퓨터를 기반으로 이더넷 및 RS-232, RS-422 통신 인터페이스를 제공하고, 탑재된 통합시험 소프트웨어(ITS, Integrated Test Software)를 통해 하드웨어-소프트웨어 통합시험 기능을 수행한다.

통합시험 소프트웨어는 자동시험 수행을 위한 스크립터 인터페이스를 제공하며, 스크립트에 따라 외부신호 모사장치의 입출력 신호를 제어 및 모니터링하고, 실시간 모니터링 장치의 제어를 수행하는 방식으로 소프트웨어 검증환경을 통제한다. 통합시험 소프트웨어는 UML(Unified Modeling Language) 기반으로 개발되었으며, 기본 구성은 Fig. 3과 같다.

ITS Tester 모듈은 사용자가 자동시험을 수행하기 위하여 사용하는 자동화 스크립터 인터페이스 기능을 수행한다. ITS Debugger 모듈은 GUI 환경에서 시험대상 장비를 디버깅하기 위한 기능을 지원하는 모듈로써 시험대상 장비에 전원을

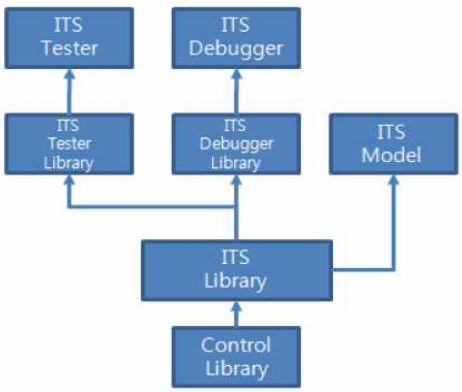

Fig. 3. 통합시험 소프트웨어 구성도

인가하고, 외부신호 제어 및 시험대상 장비 탑재 소프트웨어의 변수값 다중 모니터링 등의 기능을 수행할 수 있다. ITS Model 모듈은 시험대상 소프트웨어를 검증하기 위해 외부에서 모사하여 제공되어야 하는 신호를 모델 형태로 작성하여, 이를 소프트웨어 자동 시험시 쉽게 사용할 수 있도록 라이브러리화한 것이다. 그리고 Control Library는 외부신호 모사장비 및 실시간 모니터링 장치와 연동하여 제어 기능을 수행한다. 구성된 통합시험 소프트웨어는 실시간 모니터링장치를 통제하여 프로세서 메모리와 레지스터 읽기/쓰기, 프로세서 Run/Halt/Reset 제어, Trace 버퍼 초기화, Watch Dog Timer Enable/Disable, Break Point 제어, Trace Logic 제어를 수행한다. 또한 실시간 모니터링장치와 연동하여 주기적 또는 특정 이벤트 발생시 시험대상장비의 메모리, 아날로그 입력/출력, 디스크리트 입력/출력, 1553 MUX신호에 대하여 사이클 제어 기능을 수행한다. 그리고 외부신호 모사장치를 통제하여 다중화된 시험대상장비의 채널별로 시험대상장비 전원제어, 아날로그/디스크리트 입출력 읽기/쓰기를 위한 기능과 외부신호 모사 장치의 입력 및 출력 값의 조정을 위한 기능을 수행한다.

2.4 실시간 모니터링장치 설계

실시간 모니터링장치는 탑재 소프트웨어의 내부 데이터를 실시간으로 모니터링하기 위한 장치로 3중으로 구성된 비행조종컴퓨터의 각 채널에 대하여, NEXUS 포트를 이용하여 모니터링하기 위해 3중으로 구성된 상용 NEXUS 에뮬레이터와 호스트 컴퓨터로부터 수신받은 명령을 실시간으로 처리하기 위한 3중의 SBC(Single Board Computer)로 개발하였다. Fig. 4는 실시간 모니터링장치의 기능 구성도이다.

실시간 모니터링장치의 탑재 소프트웨어는 호스트 컴퓨터의 명령에 따라 NEXUS 에뮬레이터를 통해 시험대상 장비로부터 필요한 데이터를 모니터링하여 호스트 컴퓨터로 제공하고, 프로세서의 Break Point 및 흐름제어 기능을 수행한다. Fig. 5는 실시간 모니터링 장치 소프트웨어의 구조이다. API 계층은 실시간 운용 소프트웨어가 제공하는 실시간 API, 실시간 TCP/IP 스택 및 NEXUS 에뮬레이터 API로 구성된다. 최상위 어플리케이션 계층은 호스트 컴퓨터로부터의 모니터링 및 디버깅 명령을 처리하기 위한 TCP/IP 서버 및 서비스 태스크로 구성되며, 수신된 명령에 따라 타겟 프로세서의 모니터링을 수행하는 구조이다. 실시간 모니터링 장치에서 타겟 프로세서의 모니터링을 위해 수행되는 기능은 Table 2와 같으며, 호스트컴퓨터의 통합시험 소프트웨

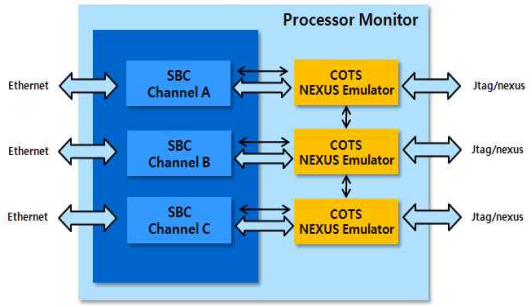

Fig. 4. 실시간 모니터링장치 구성도

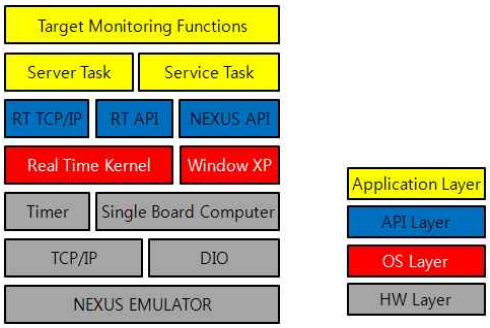

Fig. 5. 실시간 모니터링 장치 소프트웨어 구조

Table 2. 실시간 모니터링장치 주요 명령

기 능	기 능 요 약
Real Time R/W Memory	시험대상장비의 운용 중 호스트컴퓨터가 요청하는 특정 메모리의 내용을 Read/Write 수행.
R/W Target Register	시험대상장비의 운용 중 호스트컴퓨터가 요청하는 특정 레지스트의 내용을 Read/Write 수행
Single Step	호스트컴퓨터가 요청하는 step 만큼 타겟 프로세스를 Single Step 수행
On Chip Breakpoint	호스트컴퓨터의 요청에 따라 타겟 프로세스에 Breakpoint 설정
Real Time Snapshot Data	호스트컴퓨터의 요청에 따라 특정한 Breakpoint 이벤트를 기준으로 지정한 메모리의 내용을 실시간으로 저장
Real Time Triggering	호스트컴퓨터의 요청에 따라 정의된 조건이 발생시, Real Time Snapshot을 중지
Real Time Cycle	호스트컴퓨터의 요청에 따라 특정 이벤트를 기준으로 타겟 프로세스의 지정된 메모리 값을 변경.

어에서 요구되는 명령을 수행하고 이에 대한 응답 신호를 호스트컴퓨터로 송신하는 역할을 수행한다.

Fig. 6은 실시간 모니터링 장치의 실행 개념도이다. 실시간 모니터링장치는 초기화 단계에서 시스템 메모리를 초기화하여 장비 운용에 필요한 내부변수를 설정하고, 글로벌 데이터를 초기화하여 검증에 필요한 내부 변수를 초기화한다. 그리고 실시간 모니터링장치가 수행할 작업에 따라 주기적으로 수행하여야 하는 서비스 태스크와 호스트의 요청과 같은 외부 이벤트에 의해 실행되는 서비스 이벤트를 생성하여 기능을 수행한다.

2.5 외부신호 모사장치 설계

외부신호 모사장치는 호스트컴퓨터의 통제에 따라 시험케이스를 수행하기 위한 비행조종컴퓨터의 입력신호를 모사하여 제공하고, 비행조종컴퓨터로부터의 출력신호를 수신하여 호스트 컴퓨터로 제공하는 기능을 수행한다. 외부신호 모사장치는 입출력 신호모듈, 1553 통신모듈 등의 신호 인터페이스모듈과 고장주입모듈로 구성되어 있으며 외형은 Fig. 7과 같다. 입출력 신호모듈은 비행조종컴퓨터와 연동되는 외부 신호를 실시간으로 모사하며, 1553 통신모듈은 비행조종컴퓨터의 3중으로 구성된 MIL-STD-1553B 버스에 연동되는 부장비에 대한 BC(Bus Controller) 기능과 RT(Remote Terminal) 기능을 모사한다. Table 3은 비행조종컴퓨터의 외부 신호 종류 및 수를 나타낸다.

고장주입모듈은 결함모드 검증시험 수행시 비행조종컴퓨터의 외부입력 신호에 결함이 발생한 상황을 모사하기 위한 용도로 사용된다. 그리고 인터페이스 모듈은 비행조종컴퓨터의 모든 입출력 인터페이스에 대해 외부신호 모사장치와 연결하고, 신호 모니터링을 위한 테스트 포인트 패널을 장착하고 있다.

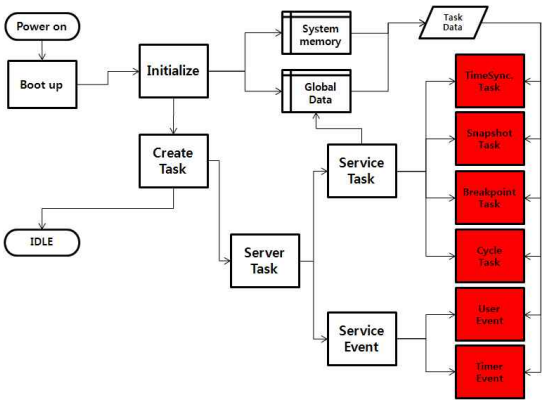

Fig. 6. 실시간 모니터링 장치 실행 개념도

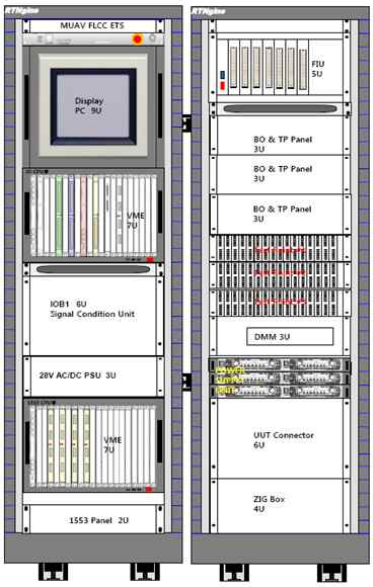

Fig. 7. 외부신호 모사장치 구성도

Table 3. 비행조종컴퓨터 외부 연동 신호

신 호 종 류	신 호 수
MIL-STD-1553B	6
RS-232	6
CAN	3
Ethernet	3
Analog	15
Discrete	144

외부신호 모사장치를 운용하기 위한 탑재 소프트웨어 구조는 Fig. 8과 같다. 외부신호 모사장치 소프트웨어는 모듈화된 계층구조로 구성되며, 실시간 운영체제와 드라이버 역할을 수행하는 시

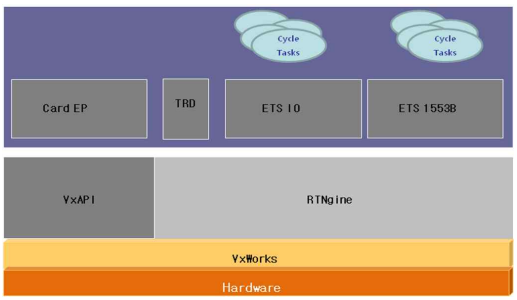

Fig. 8. 외부신호 모사장치 소프트웨어 구조

스템 소프트웨어와 외부신호 모사장치 기능을 구현한 응용 소프트웨어로 구분된다. 시스템 소프트웨어는 실시간 운영체제인 VxWorks와 타겟 프로세서 보드에 대한 VxWorks BSP로 구성되며, 운영체제와 응용소프트웨어 사이에 실시간 시뮬레이션 개발도구로써 RTNgine 소프트웨어를 사용하였으며[7], 이를 통해 스케줄러 설정, 프로젝트 관리 및 인터페이스 구현을 수행하였다. 응용 소프트웨어의 모든 기능은 RTNgine의 실시간 코어상에서 구현되고 동작된다. 응용소프트웨어는 기능별로 이미지형태로 관리되고 실행되며, 이는 호스트 컴퓨터에서 제공하는 명령을 분석하고 수행하기 위하여 필요한 명령을 각 소프트웨어 모듈에 전달하는 역할을 수행하는 Card EP(Embedded Processor) 이미지, 외부신호 모사 이미지가 실행되기 전 비행조종컴퓨터의 기능시험을 위하여 필요한 수락시험 지원기능인 TRD(Test Requirement Description) 이미지, 그리고 입출력 신호를 모사하기 위한 외부신호 모사장치의 주 기능인 외부신호 모사 이미지로 구성된다. 또한 외부신호 모사 이미지는 입출력 신호 종류에 따라 입출력 신호 모듈과 1553 통신 모듈로 구분되며, 각 모듈마다 필요한 주기적 이벤트를 처리하기 위해 주기적 태스크를 구동할 수 있도록 구성하였다. Fig. 9는 외부신호 모사장치의 실행 구조도이다. 외부신호 모사장치는 호스트컴퓨터로부터 스크립트에 따른 시험수행 명령을 이더넷을 통하여 수신받아, 이를 FIFO 버퍼에 넣어 순차적으로 명령을 수행한다. 모든 명령에 따른 입출력신호 처리판단은 입출력 명령어 처리모듈에서 수행하고, 입력된 입출력신호 명령에 따른 세부 입출력 기능 함수를 인터페이스 및 드라이버모듈에 송신하여 비행조종컴퓨터로 모사된 입력이 제공되도록 구성하였다. 또한 드라이버 모듈에서는 수신받은 데이터를 각 입출력 신호에 맞게 패킷화 하는 등의 작업을 수행하게 된다. 이때 입출력신호 명령어 처리모듈의 실시간성을 관리하고 명령어 처리시 필요한 입출력 신호 함수의 인덱스를 제공하는 기능을 RTNgine Core가 수행한다. 비행조종컴퓨터에서 출력된 데이터는 입출력신호 명령어 처리모듈을 통해 물리적 패킷에서 데이터가 추출되며, 이는 통신모듈을 통해 호스트컴퓨터로 송신된다.

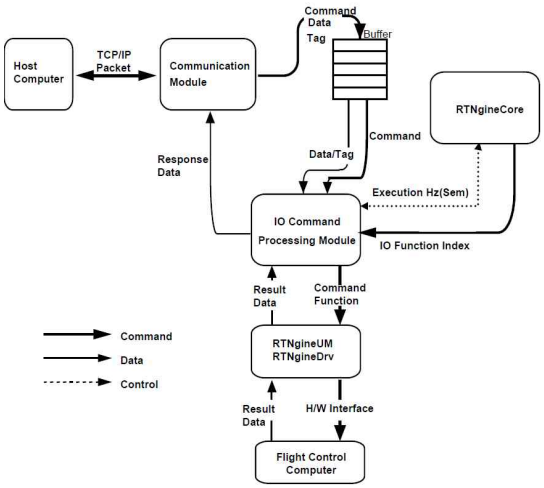

Fig. 9. 외부신호 모사장치 실행 구조도

III. 검증환경 구현 및 시험

II장에 기술된 비행조종컴퓨터 비행운용프로그램의 검증환경은 Fig. 10과 같다. 개발된 검증환경은 전투기급 비행조종시스템 신뢰성 요구도를 만족하기 위하여 3중으로 다중화 개발한 비행조종컴퓨터의 비행운용프로그램에 대하여 독립개체 시험을 수행하였으며, 시스템 통합시험을 위한 HILS(Hardware-in-the-Loop Simulation) 시험 수행시 실시간으로 소프트웨어 내부 변수 모니터링을 지원하였다. 소프트웨어 독립개체 시험은 비행운용프로그램의 기능적 요구도를 검증하기 위해 Open- Loop의 정적 환경에서 하위 수준의 요구도별 기능시험을 수행하게 된다. 독립개체 시험을 수행하기 위하여 소프트웨어 요구도 문서를 바탕으로 시험 절차서를 작성하고, 소프트웨어 컴포넌트별로 자동수행 시험 프로그램(Automated Test Command File) 형태로 스크립터를 작성하여 시험을 수행하였다.

독립개체 시험 수행을 위한 시험대상 소프트웨어의 주요 검증 기준은 다음과 같다.

Fig. 10. 비행운용프로그램 검증환경 구성

- 정확성(Accuracy)
- 시간(Timing)
- 초기화(Initialization)
- 결함 동작(Failure Operations)
- 리셋 동작(Reset Operations)

이러한 기준을 기본으로 독립개체 시험 수행을 위한 주요 시험항목들은 다음과 같다.

- 상태/모드 기능
- 초기화 기능
- 스케쥴러 기능
- 채널 동기화 기능
- 입출력 관리 기능
- 다중화 관리 기능
- 시스템 모니터 기능

Fig. 11은 검증환경의 실시간 모니터링 장치를 위한 운용화면이며, Fig. 12는 독립개체 시험 스크립터 프로그램의 예를 나타낸다. 개발된 검증환경을 이용하여 비행조종컴퓨터 비행운용프로그램의 검증을 위한 자동 시험인 독립개체 시험을 수행하여 완료하였다.

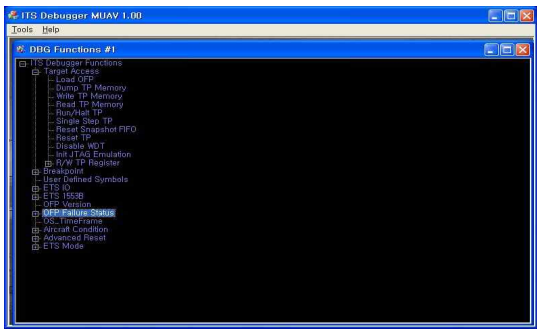

Fig. 11. 실시간 모니터링 장치 운용화면

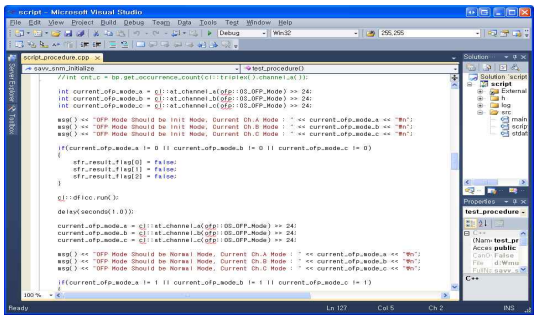

Fig. 12. 소프트웨어 검증용 스크립트

Ⅳ. 결 론

비행안전 필수 소프트웨어인 비행조종컴퓨터의 비행운용프로그램은 고신뢰성이 요구되며, 이를 확인하기 위하여 다양한 검증환경들이 개발되어 검증에 이용되었다. 그러나 비행안전 필수 소프트웨어의 검증환경 개발을 위한 기존의 방법들은 하드웨어 의존적이거나 고성능의 하드웨어 적용시 개발이 어렵고, 타겟 시스템에 따라 재사용성이 없는 단점을 가지고 있었다. 본 논문에서는 임베디드 시스템의 디버깅 방안 중 하나인 IEEE-ISTO NEXUS 5001을 이용하여 비행안전 필수 소프트웨어의 검증환경을 개발한 내용을 기술하였다. 개발한 검증환경을 이용하여 3중으로 구성된 비행조종컴퓨터 비행운용프로그램의 개발 시험을 수행하여 장비의 신뢰성을 향상시킬 수 있었다.

참고문헌

1) 이기훈, 허기봉, 김유경, "비행조종컴퓨터 OFP 검증 장치 개발", 한국항공우주학회 추계학술대회, 2007. 11.

2) Windriver, “Windview 2.0.1”, 1999

3) 공기석, 손승우, 임채덕, 김홍남, "내장형 실시간 소프트웨어의 원격디버깅을 위한 디버그에이전트의 설계 및 구현", 한국 정보과학회 추계학술대회, 1999. 10.

4) Marija Punt, Jovan Djordjevic, Milo Tomasevic, "A Simulation Environment for the On-Line Monitoring of a Fault Tolerant Flight Control Computer", IEEE Eastern European Conference on the Engineering of Computer Based Systems, 2009.

5) IEEE-ISTO, “The Nexus 5001 Forum Standard for a Global Embedded Processor Debug Interface”, 2003

6) B. Vermeulen, N. Stollon, R. Kuhnis, G. Swoboda, and J. Rearick, "Overview of debug standardization activities," IEEE Design & Test of Computers, 2008. pp 258~267

7) RealtimeWave, “RTNgine V1.5”, 2010