

다목적실용위성 2호에서 구문분석기를 이용한

탑재소프트웨어 검증시험분석

이재승⁰ 최종욱 권기호 천이진
한국항공우주연구원
{jslee⁰, jwchoi, khkwon, yjcheon}@kari.re.kr

Verification Test Script Parser for Flight Software using Lexical Analyser in KOMPSAT-2

Jae-Seung Lee⁰, Jong-Wook Choi, Ki-Ho Kwon, Yee-Jin Cheon
Korea Aerospace Research Institute

요 약

다목적실용위성 2호 FSW(Flight SoftWare)의 개발단계에서 통합 및 시험, 검증시험을 위해 실제 위성 시스템과 유사한 인터페이스를 제공하는 개발도구인 STB(Software Test Bed)가 제작되었으며, 제작된 STB를 통한 FSW의 검증시험 및 분석을 지원하기 위한 프로그램으로서 구문분석 프로그램을 이용한 VTSP(Verification Test Script Parser)의 개발이 필요하다. 본 논문에서는 구문분석 도구인 FLEX와 BISON을 이용하여 FSW가 실제 시스템 상에서 요구조건에 따라 수행되고 있는지를 자동적으로 검증하기 위해 C 언어와 유사한 사용법을 가지며, 다양한 검증시험 환경에 적합한 VTSP를 개발하기 위한 전체적인 개발환경 및 VTSP의 기능에 대해 소개한다.

1. 서 론

KOMPSAT-2의 Flight Software[1]를 개발하는데 있어서 위성발사 이전에 실제와 유사한 환경에서 수행과정을 검증하는 시험이 필요하다. Flight Software는 위성의 자세제어, 탑재체가 획득한 데이터 및 상태 데이터의 관리 및 지상관제수신 센터와의 CMD(Command)/TLM(Telemetry) 송수신 등의 작업을 수행하게 된다. VTSP는 CMD/TLM을 시험하기 위한 도구로서 사용되어진다. 즉, Telemetry 리스트에서 사용되어지는 변수값의 확인과 CMD를 보냈을 때 보낸 CMD에 대한 TLM을 받아 분석하는 역할을 하게 된다. 이러한 CMD/TLM의 시험을 수행하는데 있어서 사용자가 검증시험을 위한 Test Script를 입력하면 VTSP는 그에 대한 시험결과를 제공하는 형태의 개발도구로서의 역할을 수행하게 된다.

VTSP의 목적은 TCTS (Telemetry Command Test Set)에 프로그래머블한 시험 인터페이스를 제공하는 것이다. 이러한 요구조건과 함께 사용자에게 C 언어와 유사한 인터페이스를 제공하기 위해 다음의 기능들을 가져야 한다.

- 산술연산
- 관계 및 논리연산
- 명시적/비명시적 변수의 형변환
- 증가/감소 연산

- 비트 논리연산
- 루프문(for, while, do until, ...)
- 내부함수(메모리 덤프, 메모리 번지 연산 등 검증시험 수행에 요구되는 기타 함수들)

본 논문에서는 이러한 기능들을 제공하기위해 FLEX와 BISON을 이용한 VTSP의 개발방법을 제안한다.

2. Verification Test Script Parser

2.1 FLEX & BISON

구문분석 프로그램으로는 벨연구소에서 70년대에 개발된 LEX와 YACC[2]이 대표적인데 후에 새로운 버전으로 FLEX와 Berkely YACC이 나왔으며 최근에는 좀더 향상된 기능을 가진 GNU 프로젝트의 FLEX와 BISON이 널리 사용되고 있다. 그 외에도 여러 운영체제 하에서 여러 가지 다른 이름으로 LEX와 YACC의 변형이 지원되고 있다. 본 논문에서는 LEX와 YACC의 상위버전 호환성을 가진 FLEX[3]와 BISON[4]을 사용한다. 그림 1은 컴파일러의 관점에서 구문분석을 수행하기위해 FLEX와 BISON이 어떻게 서로 연결되는지를 보여준다.

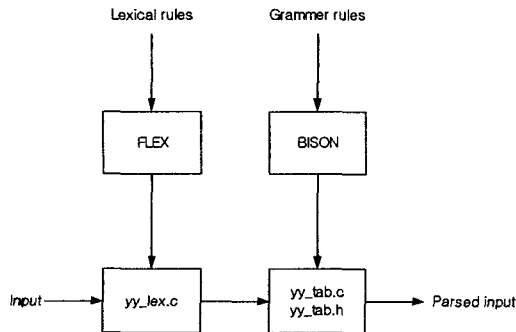


그림 1. Relation between FLEX and BISON

검증시험을 위한 스크립트가 입력되면 FLEX가 먼저 정해진 정규 표현들을 인식하고, BISON은 인식되어진 정규표현의 조각들을 분석하여 주어진 문법에 맞게 분석 결과를 생성한다.

2.2 검증시험 환경

VTSP는 TCTS에 프로그래밍 스크립트 인터페이스를 제공함으로써 검증시험과 검증시험 결과출력이 자동적으로 수행될 수 있도록 한다. VTSP는 uplink command를 통해 세 프로세서(OBC, RDU, ECU)에 명령을 보냄으로써 이러한 수행이 이루어 지도록 한다. 또한, VTSP는 위성으로부터 내려오는 텔레메트리를 주기적으로 읽어서 매초마다 받은 위성의 상태 데이터를 업데이트 함으로써 검증시험 스크립트에서 변경된 값들을 실시간으로 사용할 수 있는 환경을 제공한다.

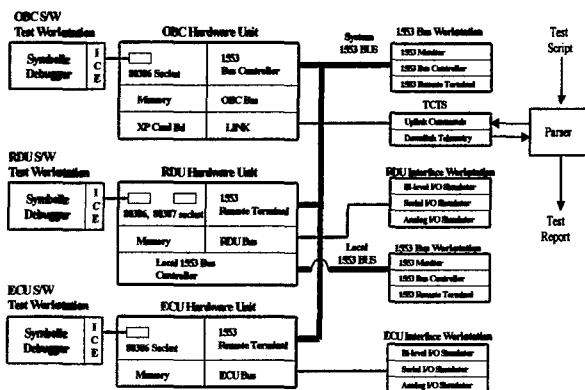


그림 2. Verification Test Environment

그림 2는 검증시험 환경을 보여준다. 그림에 나타난 바와 같이 VTSP는 TCTS를 통해 STB와 명령 및 텔레

메트리를 주고받음으로서 작성된 스크립트에 따라 자동적으로 검증시험 결과를 생성하며, 이를 위해 VTSP는 검증시험에 사용되는 명령 및 텔레메트리와 관련된 정형화된 함수들을 제공한다.

VTSP는 TCTS Workstation상에서 동작하여 TCTS Workstation과 TCTS target board와의 명령이나 텔레메트리의 통신에 사용자가 접근하여 이용할 수 있도록 인터페이스를 제공해주는 역할을 한다. 그림 3은 이러한 VTSP와 TCTS간의 연결 환경을 나타낸다.

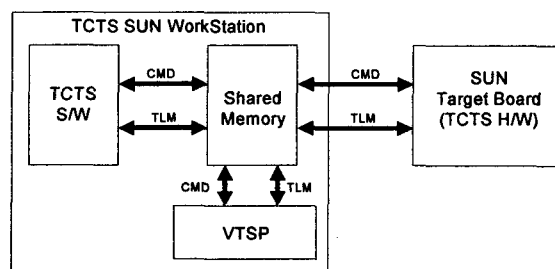


그림 3. How VTSP works with TCTS H/W

3. VTSP의 설계

3.1 VTSP의 구조

2.1절에서 언급한 바와 같이 VTSP는 FLEX와 BISON이라는 툴을 이용하여 프로그래밍된다. 먼저 FLEX에 의해서 이루어지는 과정을 살펴보면 검증시험을 위해 작성된 스크립트 파일을 입력받아서 구문분석을 위해 정해진 토큰(token)들을 인식하여 BISON에 넘겨주게 된다. 즉, 스크립트에 숫자가 있다면 FLEX는 BISON에 'DIGIT'이라는 토큰과 실제 값을 리턴한다. 주석을 제외하고 토큰으로 정의되어있지 않은 내용에 대해서는 단순히 1바이트씩 읽은 값을 리턴한다. VTSP에서 사용되는 대표적인 토큰들은 다음과 같다.

VAR	INCLUDE	DO	SYMBOL_NAME
CHAR	PRINT	IF	INT_DIGIT
INT	DATE	ELSE	HEX_DIGIT
SHORT	TIME	FOR	
LONG	CASE	UPLINK	
FLOAT	DEFAULT	COMMAND	
DOUBLE	EXIT	FUNCTION	

BISON은 위의 토큰들을 분석하여 미리 정의된 심볼(symbol)들로 통합하는 작업을 반복하여 결과적으로 하

나의 스크립트를 인식하도록 작성된다. 일반적으로 하나의 소스코드는 헤더파일들을 인클루드하는 부분과 변수나 함수들을 선언하는 부분 그리고 변수값들을 이용하여 계산하거나 조건문, 루프문 등을 통해 변수에 새로운 값을 할당하는 등의 실제 함수의 기능을 작성한 부분으로 크게 나누어진다. 여기에서 사용한 함수, 변수값, 변수타입, 조건문, 루프문 그리고 그 외의 여러 내부함수들은 FLEX에서 출력된 토큰들로 이루어진다. BISON은 이러한 토큰들의 분석을 통해서 검증시험 스크립트를 이루는 부분들을 구성하면서 해당하는 심볼과 연산이나 함수실행 결과값을 리턴하게 되고, 모든 부분들이 갖추어진 하나의 소스코드가 구성되면 구문분석은 종료된다. BISON은 구문분석 도중에 내부 및 사용자정의 함수가 있으면 해당하는 연산이나 명령처리 등을 분석과 동시에 수행하게 된다. 다음의 그림 4는 BISON에 의해 토큰들이 분석되어지는 구조의 한 예를 나타낸 것으로 대문자로 표시된 것은 토큰을 의미하며 소문자는 심볼을 의미한다.

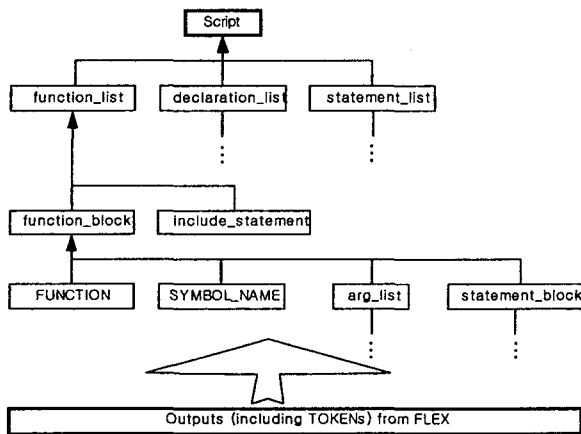


그림 4. Structure of symbols in BISON for VTSP

3.2 VTSP의 역할

VTSP는 위성탑재 소프트웨어의 검증시험을 위한 프로그램으로서 가장 중요한 기능은 타겟으로 지상의 명령을 보내고 위성으로부터 상태데이터를 받아볼 수 있어야 한다는 것이다. 이를 위해 명령을 보내는 'uplink command' 함수와 위성으로부터 덤프되는 데이터를 받는 'obc_dump_byte()' 함수가 제공된다. 사용자는 단순히 보내고자 하는 명령의 hexa값을, 또는 받아보고자 하는 덤프데이터의 위치만을 입력하면 해당하는 값을 얻을 수 있다. 또한 매초마다 생성되는 위성의 텔레메트리 프

레이프는 자동적으로 분석 및 저장이 이루어진다. 새로운 내부함수를 'FUNCTION'이라는 토큰을 이용하여 정의하면 VTSP의 소스코드에도 직접적인 수정을 할 수 있다.

4. 결 론

본 논문에서는 구문분석 도구인 FLEX와 BISON을 이용한 VTSP의 개발에 대해 설명하였다. 검증시험을 수행하는 데에는 많은 시간이 필요하며, 각각의 시험에 대해 다양한 경우를 검증할 필요가 있다. 따라서 여러 가지 함수가 사용되고 각각의 경우에 따라 함수나 변수가 변경될 수도 있으며 정상 동작 시에는 발생하지 않는 상황을 만들 수 있어야 한다. 그림 5는 이런 검증시험이 수행되는 방법을 나타낸다.

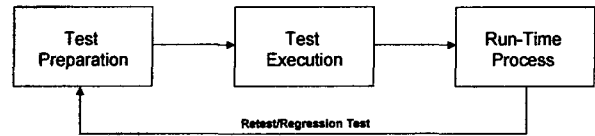


그림 5. How test goes

기존의 C 언어로 작성할 경우 함수가 변경되거나 새로이 추가되면 소스파일을 수정하고 컴파일을 다시하여 사용해야 한다. 그러나 FLEX와 BISON을 이용한 VTSP는 사용자가 입력하는 스크립트에 추가될 함수를 넣어주면 자동적으로 인식하게 되어 사용자에게 프로그래머블한 인터페이스를 제공하므로 검증시험의 시간을 크게 줄여 줄 수 있다. 또한 입력 스크립트를 인식하는 과정에서 구문분석에 적합하지 않은 부분의 정보를 알려주므로 잘못된 CMD/TLM이 발생했을 때 디버깅하기가 쉽다는 장점이 있다. 현재 개발된 VTSP를 이용하여 다목적실용위성 2호 FSW의 검증시험이 수행되고 있다.

참고문헌

- [1] 이종인, "아리랑 위성 탑재 소프트웨어 소개", 한국정보과학회 가을학술발표 논문집(III), 제 25권 2호, 1998, pp. 662-664.
- [2] John R. Levin, Tony Mason, Doug Brown, "Lex & Yacc", O'Reilly & Associates Inc., 1995.
- [3] Vern Paxson, "Flex, version 2.5", The University of California, 1995.
- [4] Charles Donnelly, Ricahrd Stallman, "Bison(The YACC-compatible Parser Generator)", Free Software Foundation, 1995.