한국방위산업학회
제16권 제1호
2009년 6월

항전장비 규격만족을 위한 「ADC」 소프트웨어의 고려사항

김 창성[1†] 최 인호[2]

내용목차

1. 서론
2. 에어데이타 컴퓨터의 개요
3. 소프트웨어 단계별 고려사항
4. 자동화 툴의 고려사항
5. 결론

본 연구는 지식경제부 KHP 민군겸용 핵심 구성품 개발사업의 일부로 지원을 받아 수행되었음.

1† 퍼스텍 주식회사 전자개발부 선임연구원
(교신저자 Tel: 010-9772-6769 E-mail: pstokyo@empal.com)
2 항공우주연구원 세부계통팀 선임연구원

논문접수일: 2009년 03월 27일 게재확정일: 2009년 05월 23일
논문수정일 (1차: 2009년 5월 12일)

Korea Association of
Defense Industry Studies
Vol.16, No.1, June 2009

Considerations of 「ADC」 Software for Compliance with Avionic Standard

Kim, Chang Seung[1†] Choi, In Ho[2]

Abstract

Avionic software is more difficult to find its defect compare than graphic user interface based software. According to America's 「National Transport Safe Board(NTSB)」, most of aircraft accident has been caused primarily by improperly developed software. So airworthiness authority strongly asks to software quality verification in today's avionic software development.

「Air Data Computer(ADC)」 is a safety critical system because 「ADC」 calculates altitude, speed, temperature and so on. These parameters are used to directly control 「AFCS(Auto Flight Control System)」. Avionic software has to be developed for compliance with avionic software standard i.e. 「RTCA DO-178B」. This paper describes how to apply this standard to the 「ADC」 software development and how to verify air data software. In addition, this paper also mentions some additional consideration for software certification.

<Key Words> *Air Data Computer(ADC),*
RTCA(Radio Technical Commission for Aeronautics) DO-178B,
Modified Condition and Decision Coverage (MC/DC)

1. 서론

‘임베디드 소프트웨어’는 ‘Graphic User Interface(GUI로 약칭)’기반의 SW와는 달리 검사나 시험을 통해 결함을 찾아내고 제거하는 일이 쉽지 않아 객관적으로 품질을 평가하기 어렵기 때문에 SW제품에 대하여 품질보증활동을 수행하는 것은 하드웨어에 대한 품질보증 이상으로 중요한 과제이다.

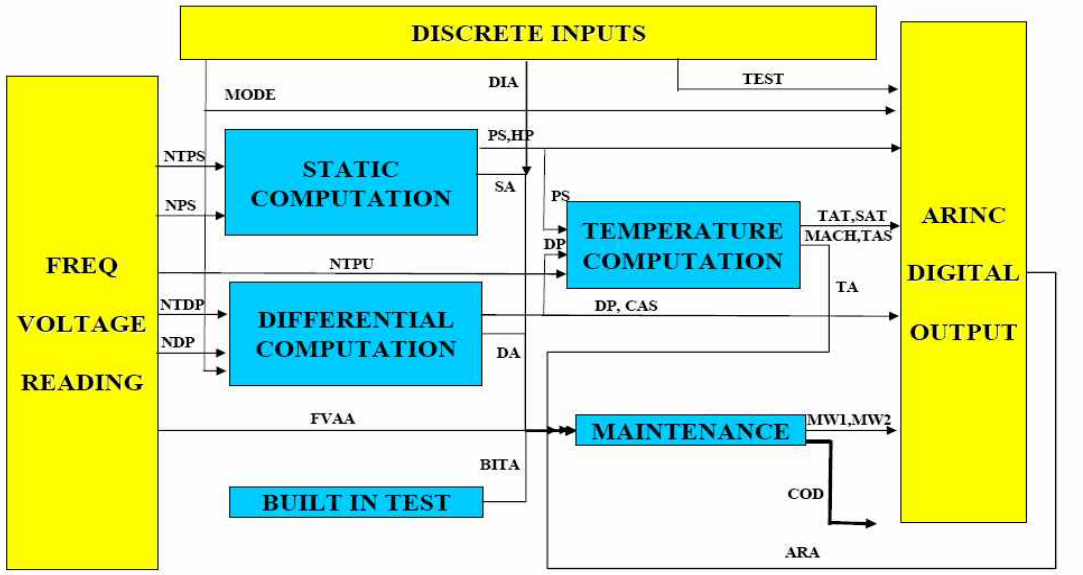
현대의 항공기는 HW와 SW가 결합된 형태로 개발되고 있으며, 항전장비에서 SW가 차지하는 비중이 점점 증대되고 있는 추세이다. 따라서 사실상 SW가 항공기 전체의 품질 및 신뢰성을 좌우하는 핵심적인 역할을 하고 있다.

이와 같이 항공기의 시스템이 디지털방식으로 전환되면서 높은 품질의 SW를 개발하기 위한 관심이 집중되고, 항공기 개발과정에서 SW의 중요성이 증대되고 있어 감항기술의 평가기준에서도 내장형 SW에 대한 품질인증을 요구하고 있다.

「Air Data Computer(ADC로 약칭)」는 감항당국에서 규정하는 기술표준품(Technical Standard Order ; TSO로 약칭) C-106에 해당하는 품목으로서 이 품목에 해당되는 SW는 RTCA(Radio Technical Commission for Aeronautics의 약칭)에서 규정하고 있는 DO-178B를 만족해야 한다[3]. 본 연구에서는 항전장비인 「ADC」의 SW 규격인 DO-178B를 만족하기 위한 설계 및 검증 시에 고려해야 할 사항에 대하여 기술하였다.

2. 에어데이타 컴퓨터의 개요

「ADC」는 전원인가 시 이산신호에 의해서 모드와 비행정보의 테이블을 선택하게 되며, 센서로부터 주파수를 읽어, ‘Air Data Probe(ADP로 약칭)’로부터 ‘대기의 차압(Differential Pressure ; Dp로 약칭)’, ‘정압(Static Pressure ; Ps로 약칭)’, ‘온도(Total temperature ; Tt로 약칭)’ 등의 대기 자료를 수집 및 분석하여 속도, 고도, 고도상승률 등을 실시간으로 계산한 후, ARINC(Aeronautical Radio, Inc의 약칭) 429의 디지털 통신을 통하여 타 계통에 전달하는 역할을 수행하는 시스템이다[1][4].



<그림 1> 「ADC」SW의 블록 다이어그램

3. 소프트웨어 단계별 고려사항

항전 SW의 규격인 DO-178B를 논하려면 SW공학 전반에 걸쳐 광범위하게 기술되어야한다.[3] 퍼스텍에서는 이미 한국방위산업의 SW개발표준인 방위사업청 'SW프로세스 지침'과 항공규격표준인 DO-178B에 대해서 비교연구를 수행하였으며, 본 연구에서는 특별히 항공규격을 만족하기 위한 고려사항을 중점적으로 기술한다.

3.1 계획단계의 고려사항

일반적으로 여러가지 SW규격을 분석해 보았을 때, 계획단계에서 설립되어야 할 사항들은 거의 유사하다고 할 수 있다. 다만 항전 SW에서는 「시스템 안정성 평가(System Safety Assessment) 」를 통해서 SW의 레벨을 계획단계에서 결정해야 한다[2](관련 규격 : DO-178B §2.2.3). 본 연구에 기술된 「ADC」 SW는 이러한 과정을 통하여 DO-178B Level A로 결정하였다.

<표 1> SW의 고장조건과 레벨 간의 관계

Failure Condition	Software Level	Process objects	Description
Catastrophic	A	66	안정적 항공기 운항 및 착륙이 불가능한 상태
Hazardous/ Severe-Major	B	65	안전운항 능력 및 승무원의 능력을 크게 감소시키고 물리적 재난 또는 승무원이 그 역할을 제대로 할 수 없을 정도의 상태 탑승객이 상처를 입거나 생명에 지장이 있을 정도의 상태
Major	C	57	탑승객이 편안함을 느끼지 못하거나 부상을 입을 수 있는 상태
Minor	D	28	중대한 영향을 미칠 수 있는 상태가 아니고 승무원이 그 역할을 제대로 수행할 수 있는 상태. 안전에 극미하게 영향을 미치는 상태 (예를 들며 비행경로 변경 등)
No effect	E	0	영향성 없음

위의 표에서 보듯이 「ADC」 SW는 66개의 목표요건을 만족하여야 한다[3].

3.2 요구사항 분석단계의 고려사항

SW의 요구사항을 분석하고 할당하는데 가장 중요한 역할을 하는 것은 <그림 2>와 같은 '시스템 요구사항 할당표(System Requirement Allocation Matrix)'이다. 아래의 표는 실제 「ADC」 SW개발에 사용된 매트릭스이다. 시스템의 요구사항은 'HW레벨', 'SW레벨', '제품레벨'로 분리되어 할당되었으며, SW개발자는 아래의 표를 참조하여 SW의 요구사항을 정의해 나간다.

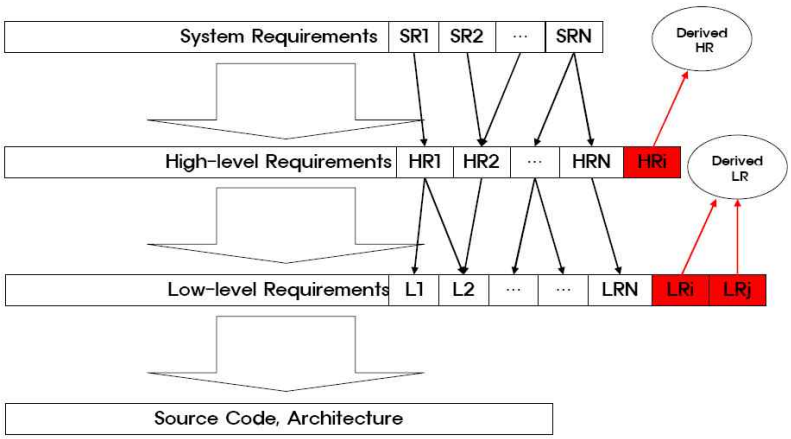
요구사항을 정의하다보면 시스템의 요구사항이 SW의 요구사항으로 할당되지는 않았으나 SW의 요구사항으로 표시되어야 할 부분이 있다. 이런 요구사항을 항공에서는 'Derived 요구사항'이라고 부르며 이 'Derived 요구사항'은 시스템 엔지니어에 의해 「안정성 분석(Safety Analysis)」을 수행하도록 항공규격에는 규정하고 있다[3]. (관련 규격 : DO-178B §5.1.2 j, §5.2.1 b, §6.3.1)

Allocation Matrix 1

Requirement identifier	Requirement title	S/W	H/W			Product	Remark
			Board	FPGA	Sensor		
Requirement 4.1	Voltage Limits		X				
Requirement 4.2	Consumption		X	X	X		
Requirement 4.3	Transient Current Demand		X				
Requirement 4.4	Supply hold-up requirement		X				
Requirement 4.5	Reverse-polarity protection		X				
Requirement 4.6	Mode program pin	X	X	X			
Requirement 4.7	Test program pin	X	X	X			
Requirement C4.1	Aircraft law selection	X	X	X			
Requirement C4.2	Aircraft law selection	X	X	X			
Requirement C4.3	Parity of program pin	X	X	X			
Requirement 4.9	Temperature probe resistance measurement		X	X			
Requirement 4.10	Baro correction potentiometric ratio measurement						Canceled
Requirement C4.4	Spare inputs		X				
Requirement 4.12	Short Circuit Protection		X				
Requirement 4.13	Chassis Ground Output		X				
Requirement 4.14	Shielding						
Requirement 4.15	Over shielding						
Requirement 4.16	Program Pin Common		X				
Requirement 4.17	Correct operating discrete	X	X				
Requirement 4.18	Temperature probe polarization		X				
Requirement 4.19	BCor potentiometer polarization						Canceled
Requirement 4.20	Digital output		X	X			
Requirement 4.21	Digital output	X	X	X			

<그림 2> 「ADC」의 시스템 요구사항 할당표

항공규격과 방위산업에서 사용하는 용어에 약간의 미묘한 차이가 있는데 항공산업에서 말하는 'High Level 요구사항'은 방위사업에서 말하는 'SW의 요구사항'으로 'SW 요구규격서(Software requirement specification ; SRS로 약칭)'에 기록되고, 'Low level 요구사항'은 방위산업에서 말하는 '설계요구사항' 또는 '설계사항'을 의미하며 'SW 설계기술서(Software Design Description ; SDD로 약칭)'에 기술된다. 시스템 엔지니어에 의해 검증되지 않은 'High Level Derived 요구사항' 또는 'Low level Derived 요구사항'은 잠재적 SW에러를 만들 가능성이 가장 높다.



<그림 3> Derived 요구사항의 개념도

2 DERIVED REQUIREMENTS IN THE SRS DOCUMENT REF : J57479AA

1)-
SRS-REQ-3.2.1.1.3.0-A
UPWARD REQUIREMENT : DERIVED REQUIREMENT)
Static pressure frequency value shall be read at a rate of 40Hz.
Justification: Due to REQ 4.22 of CIDS, Arinc parameters have to be transmitted at 20 Hz rate
The acquisition rate has to be greater than output transmission rate, for reducing the output delay.

2)-
SRS-REQ-3.2.1.2.3.0-A
UPWARD REQUIREMENT : DERIVED REQUIREMENT
Differential pressure sensor frequency value shall be read at a rate of 40Hz
Justification: Due to REQ 4.22 of CIDS, Arinc parameters have to be transmitted at 20 Hz rate
The acquisition rate has to be greater than output transmission rate, for reducing the output delay.

3)-
SRS-REQ-3.2.1.3.3.0-A
UPWARD REQUIREMENT : DERIVED REQUIREMENT
Temperature sensor frequency values shall be read at a rate of 40Hz.
Justification: Due to REQ 4.22 of CIDS, Arinc parameters have to be transmitted at 20 Hz rate
The acquisition rate has to be greater than output transmission rate, for reducing the output delay.

4)-
SRS-REQ-3.2.1.3.3.0-B
UPWARD REQUIREMENT : DERIVED REQUIREMENT
Values of static temperature measurement shall be filtered with a "walking" average on 8 cycles at 40 Hz. For a cycle without static temperature measurement, the missing value is updated with the value at the cycle before.
Justification: In CIDS REQ 4.39 for dynamic data management, it is necessary to filter pressure primary parameters (rotor perturbation). For the sensors temperature there is some noise superposed with the useful signal. Due to processor load, a walking average filtering is sufficient in order to reduce efficiently the noise.

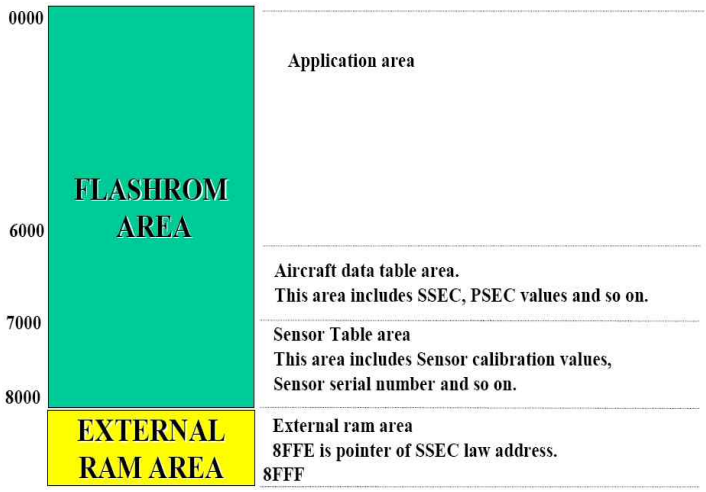
5)-
SRS-REQ-3.2.1.4.3.0-A
UPWARD REQUIREMENT : DERIVED REQUIREMENT
Temperature sensor frequency values shall be read at a rate of 40Hz.
Justification: Due to REQ 4.22 of CIDS, Arinc parameters have to be transmitted at 20 Hz rate
The acquisition rate has to be greater than output transmission rate, for reducing the output delay.

<그림 4> 「ADC」의 'Derived 요구사항' 분석자료

본 「ADC」 SW의 개발에서는 <그림 4>에서처럼 모든 'Derived 요구사항'이 시스템 엔지니어에 의해 '안전성 분석'이 되었으며, 'High level derived 요구사항' 45개, 'Low level derived 요구사항' 18개에 대하여 「안정성 분석」을 실시하였다. 따라서 항전 SW관련 규격(DO-178B §5.1.2 j, §5.2.1 b, §6.3.1)을 만족한다[3].

3.3 설계단계의 고려사항

「ADC」의 메모리 영역은 아래의 그림과 같이 애플리케이션 영역, 비행정보 테이블 영역, 센서테이블 영역으로 분리되어 있다. 또한 센서메모리 영역은 Ps센서 영역, Dp센서 영역으로 분리되어 있다. 이러한 메모리 영역을 분리하지 않으면 SW상수를 소스코드에 정의할 시 「ADC」의 제품특성상 매 제품마다 「단위시험(Unit Test)」을 다시 해야만 한다. 「ADC」 SW는 항전 SW규격을 만족하면서 이러한 검증에 소모되는 시간을 단축시키고 애플리케이션의 무결성을 보장하기 위하여 아래의 그림과 같이 메모리 영역을 분리하였다.



<그림 5> 「ADC」 메모리 맵

1) 센서테이블의 설계

「ADC」는 매우 높은 정확도를 요하는 압력센서가 포함되어 있다. 「ADC」에 탑재된 센서는 20년 이상의 수명과 50000 시간의 이상의 신뢰성을 보장한다. 이 같은

신뢰성 분석자료를 바탕으로 하였을 때, 「ADC」에 탑재된 센서의 수명은 반영구적이라고 표현할 수 있다. 더 중요한 것은 이 같은 「ADC」의 생명주기 동안 센서의 보정이 필요하다는 것이다. 그 이유는 SW에 의한 보정방법을 사용하여 「ADC」의 생명주기 동안 영구적으로 보정이 필요하지 않도록 설계하였기 때문이다.

이러한 센서의 보정데이터는 센서 공급자에 의해서 제공되며 각 센서마다 보정되어져야할 값이 다르다.

DATA	LSByte ADDR.	MSByte ADDR.
P(I,j)	0000	0001
P(i,j+1)	0002	0003
...		
P(i+M,j+N)	1918	1919
Reserved (Forced to FF_hexa)	1920	1999
n: Tabulation step of Fp0 expressed as 2^n	2000	2001
m: Tabulation step of Ft0 expressed as 2^m	2002	2003
Reserved (Forced to FF_hexa)	2004	2005
SN: Serial Number	2006	2007
PN: Part Number	2008	2009
CM: Calibration Month	2010	2011
CY: Calibration Year	2012	2013
Lowest tabulated value of temperature Ft0	2014	2015
Lowest tabulated value of pressure Fp0	2016	2017
ht: Tabulation step in temperature expressed as 2^ht	2018	2019
hp: Tabulation step in pressure expressed as 2^hp	2020	2021
M: Number of tabulated points in temperature (M=32)	2022	2023
N: Number of tabulated points in pressure (N=30)	2024	2025
Inverse of pressure scale factor (LSB value in 1/hPa)	2026	2027
K0: Offset constant	2028	2029
Reserved (Forced to FF_hexa)	2030	2039
Temperature value at 0°C	2040	2041
Temperature scale factor in °C in (Ft0 LSB /°C)	2042	2043
Check sum control word (LSB)	2044	2045
Check sum control word (MSB)	2046	2047

<그림 6> 「ADC」에 사용된 센서테이블

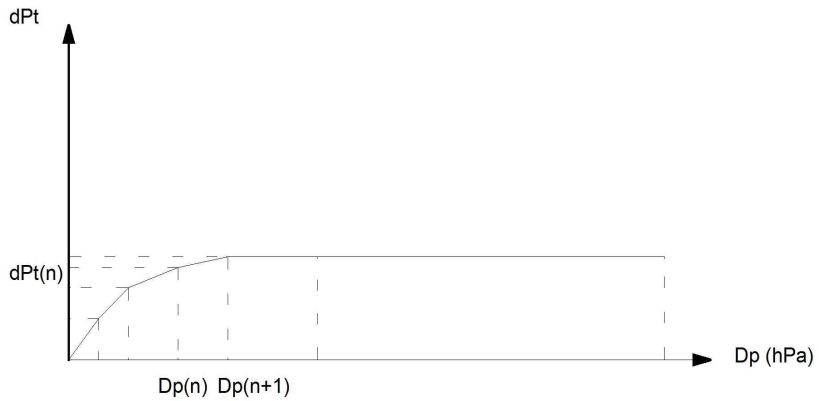
따라서 위와 같은 센서데이터의 값을 소스코드 파일에 넣어 상수로 정의해 놓는다면, 매 제품마다 다른 SW가 생성되어져야 한다. 이는 센서가 바뀔 때마다 변경되는 부분에 대한 「영향성 분석」 및 「단위시험」, 「단위통합시험」 등이 재실시되어야 한다는 의미이며 '형상관리 및 통제'가 사실상 불가능하게 되어 버린다. 이러한 시간 손실 및 애플리케이션의 무결성을 보장하기 위하여 위와 같은 메모리 영역 분리를 통한 설계는 「ADC」 SW에서는 필수라 할 것이다.

2) 비행정보테이블의 설계

「ADC」는 비행시험, 풍동시험 등에 의하여 「소스에러의 보정(Source Error Correction ; SEC로 약칭)」을 실시하여야 하는데 「SEC」는 크게 「SSEC ; Static Source Error Correction의 약칭」와 「PSEC ; Pitot Source Error Correction의 약칭」로 구성되어 있다.[4]

이 같은 보정결과값을 「ADC」에 적용하기 위해서는 비행정보 테이블의 설계단계에서 반영되어야 하며, 보정값은 '선형함수 (dps = a*dp + b)' 로 모델링되어야 하며, 다음과 같은 포맷을 가져야 한다.

① dPt(n) : 16bit, lsb = 1/256 hPa PSEC correction
② Dp(n) : 16bit, lsb = 1/256 hPa differential pressure



<그림 7> Dp에 따른 dPt 보정값

dPt(n) = a(n) * Dp(n) + b(n)
dPt(n+1) = a(n) * Dp(n+1) + b(n)
Coefficients a and b shall be for each range : (1)
a(n) = (dPt(n+1) - dPt(n)) / (Dp(n+1) - Dp(n))
b(n) = dPt(n) - a(n) * Dp(n)

비행정보 테이블의 값은 센서테이블보다 자주 바뀌지 않으나 비행시험 및 풍동시험 등에 의해서 변화한다. 본 「ADC」 SW는 특정의 회전익에만 고정되지 않고 타 회전익에 적용이 용이하도록 하기 위하여 항공기 특성에 맞는 파라미터를 비행정보 테이블에 추가로 정의하였다.

3) SW라이브러리의 사용제약

항전 SW 규격의 만족을 위한 SW의 개발에 있어서 컴파일러에서 제공하는 라이브러리 사용시에 엄청난 제약이 따른다. 컴파일러에서 제공한 SW 라이브러리를 항전 SW에 사용하고 적용하기 위해서는 라이브러리 또한 커버리지를 실시하여야 하며 이에 대한 검증결과를 감항당국에 제공하여야 한다. 따라서 컴파일러 업체에서 이 같은 검증자료를 제공하지 받지 않는 한 라이브러리를 사용하여 항전 SW의 인증을 받기란 사실상 불가능하다. 따라서 항전 장비 SW에서는 컴파일러에서 제공하는 라이브러리를 최대한 사용하지 않도록 해야 한다.

예를 들면 항전 SW에서 컴파일러에서 제공하는 수학 함수 라이브러리를 쓰지 않고 여러가지 복잡한 다항식을 풀어 나가는 방법으로 「보간법」을 활용하는 방법이 있다.

「보간법」이란 'x0 < x1 < L<xn '으로 주어진 'n + 1개'의 변수에 대한 함수값을 알고 있을 때, 이들 'n+ 1개'의 모든 점을 통과하는 근사함수를 이용하여, 구간 [x0 , xn]내의 임의의 점 x에 대한 함수 값을 구하는 방법이다.

아래의 공식은 성층권에서의 Ps값으로 고도를 계산하는 공식이다.

Hp = a2 - b2*LOG(Pressure) in range 54.748 < pressure ≦ 226.320hPa (2)
a2 = 148897.421
b2 = -20805.827

일반적인 'C 언어'에서는 수학함수 라이브러리를 사용할 경우, 위의 공식대로 한 줄로 코딩이 끝나지만 「보간법」을 사용할 경우 그 양은 방대해진다.

;Table THP8 pour 1008 =< P < 1296 (hPa)									
;LSB (P) = 1/32 hPa : 2 decalages									
;LSB (HP) = 1 ft avec un offset de 7500 ft									
;pas = (2**9) * (1/32) = 512 * (1/32) = 16 hPa									
;adresse table THP7									
THP8:									
	DW	41472			최대값(Ps)		1296		
	DW	32256			최소값(Ps)		1008		
	DB	9			X축의 STEP		16		
	DB	2			입력값(X)와 동일하게 하기위한 값				
	DW	THP7			전포인터 주소	Hp	Hp - offset	Ps(hPa)	계산값
	DW	7644		0		7644	144	1008	143,7888
	DW	7208		1		7208	-292	1024	-292,226
	DW	6777		2		6777	-723	1040	-722,759
	DW	6352		3		6352	-1148	1056	-1147,96
	DW	5932		4		5932	-1568	1072	-1567,98
	DW	5517		5		5517	-1983	1088	-1982,95
	DW	5107		6		5107	-2393	1104	-2393,01
	DW	4702		7		4702	-2798	1120	-2798,28
	DW	4301		8		4301	-3199	1136	-3198,9
	DW	3905		9		3905	-3595	1152	-3594,97
	DW	3513		10		3513	-3987	1168	-3986,61
	DW	3126		11		3126	-4374	1184	-4373,93
	DW	2743		12		2743	-4757	1200	-4757,03
	DW	2364		13		2364	-5136	1216	-5136,02
	DW	1989		14		1989	-5511	1232	-5511
	DW	1618		15		1618	-5882	1248	-5882,05
	DW	1251		16		1251	-6249	1264	-6249,27
	DW	887		17		887	-6613	1280	-6612,74
	DW	527		18		527	-6973	1296	-6972,56
	DW	171		19		171	-7329	1312	-7328,79

<그림 8> 보간법을 사용한 고도계산의 예

위의 그림에서 보듯이 「보간법」을 사용한 고도계산은 「ADC」 SW가 DO-178B Level A를 만족하기위한 「MC/DC (Modified Condition and Decision Coverage의 약칭) 분석」을 하기 위하여 적용되었다.

본 「ADC」 SW의 검증을 위하여 모든 수학적 계산의 경우 라이브러리를 사용하지 않고 「보간법」으로 활용하여 Air Data 파라메타를 계산하였다.

4) 'Pseudo-Code'의 작성

아래의 그림은 「ADC」에 사용된 'Pseudo-Code'의 예이며 모든 함수에 대하여 'Pseudo-Code'를 작성하여 시험하였다. 'Pseudo-Code' 작성의 목적은 SW코딩 전에 미리 설계사항이 잘 반영되었는지 검토하는 것이며 또한 항전 SW규격에서 규정하고 있는 'Deactivated code' 및 'Dead Code'를 검출하고 'SW 코딩 스탠더드'를 만족하는 검증을 수행하는 것이다(관련 규격 : DO-178B §5.3.2 b, §6.4.4.3 b, c)[3].

「ADC」는 설계단계에서 「CALLIOPE」라는 툴을 사용하여 SW코팅 스탠더드의 적용여부, 함수 내 변수의 사용유무, Deactivated code 및 Dead code의 검출에 사용하였다.

```
CALTI Description
==================
Computation of ti the impact temperature.
Interface
=========
- inputs
...hard : none
...soft : 01H:02H = Rs value
..... : ADBADT aircraft table address
..... : OFFTIFRS offset in the interpolation table ti = f(Rs) in the aircraft table
- outputs
...hard : none
...soft : 00H:01H = ti : impact temperature
..... A_Ti = carry = 0 if no anomaly
..... A_Ti = carry = 1 if anomaly detected
Local Data
==========
..none
%T CODE CALTI
..begin
DPTR = Ti = f(Rs) table address
TIFRS.. Computation of ti by parabolic interpolation
00H:01H = ti values registers 11H:12H
A_Ti is in the carry
..end
%S EXTERN CALTS
```

<그림 9> 「ADC」에 적용된 'Pseudo-Code'

3.4 검증단계의 고려사항

항전규격에서는 테스트 케이스 생성시 각 SW레벨에 맞는 테스트 커버리지를 적용하도록 되어 있다.

‘테스트 커버리지’란 주어진 테스트 케이스에 의해 수행된 테스트 범위의 측정 측도를 말하며 요구사항기반의 ‘요구사항 커버리지(Requirement Coverage)’와 ‘코드 기반의 커버리지(Code coverage)’, ‘구조적 커버리지’ 분석을 통해 의도하지 않은 기능의 유무와 테스트케이스의 적절성을 판단할 수 있다. 이와 관련한 기본 Code coverage의 종류는 다음과 같다.

<표 2> SW Coverage 비교표

Coverage 구분	SC	DC	CC	C/DC	MC/DC	MCC
프로그램 내에 있는 모든 구분을 적어도 한번 수행	V	V		V	V	V
프로그램 내에 있는 모든 결정 포인트에 대해 모든 가능한 결과(참, 거짓)를 적어도 한번 수행		V		V	V	V
프로그램 내에 있는 결정 포인트 내의 모든 각 개별조건식에 대한 모든 가능한 결과(참, 거짓)에 대해 적어도 한번 수행			V	V	V	V
결정 포인트 내에 있는 모든 개별조건식은 독립적으로 전체조건식(판단문)의 결과에 영향을 줌. 즉, 결정 포인트 내의 다른 개별조건식의 결과와는 독립적으로 해당 개별조건식은 전체조건식의 결과에 영향을 줌					V	V
결정 포인트 내의 개별조건식 결과(참, 거짓)에 대한 모든 가능한 조합을 적어도 한번 수행						V
포함관계		SC	SC	DC, CC	C/DC	MC/DC

위의 표는 ‘SW 커버리지’에 대하여 정리한 표이며 항전 SW인 경우 세 가지 커버리지(Statement Coverage, Decision Coverage, MC/DC)를 규정하고 있는데

본 「ADC」의 경우, SW규격인 Level A를 만족하기 위하여 MC/DC를 수행하였다.

(1) Statement Coverage

① DO-178B Level C에 해당

② 프로그램에 있는 모든 문장이 적어도 한번 씩 수행

③ 기본적인 커버리지이지만 제어구문에 취약

(2) Decision Coverage

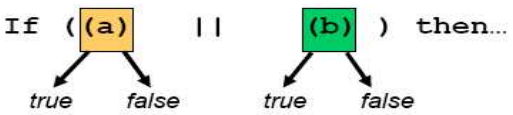
① DO-178B Level B에 해당

② '분기 커버리지(Branch coverage)'라고도 하며 프로그램의 모든 결정에서 가능한 모든 결과에 대하여 적어도 한번 이상 수행되어야 함.

(3) Modified Condition Decision Coverage (MC/DC)

① DO-178B Level A에 해당

② 'Condition/Decision Coverage를 만족하고 결정에서 각 조건들은 결정에서 결정들의 결과에 독립적이어야 함.



Coverage	Decision	Condition	Condition/ Decision	MC/DC	Multiple Condition
a or b	TF	TF	TT	TF	TT
	FF	FT	FF	FT	TF
				FF	FT
					FF

<그림 10> 코드 커버리지의 예

본 「ADC」 SW는 항전규격인 DO-178B Level A를 만족하는 137개 함수에 대한 MC/DC를 100% 수행하였다.

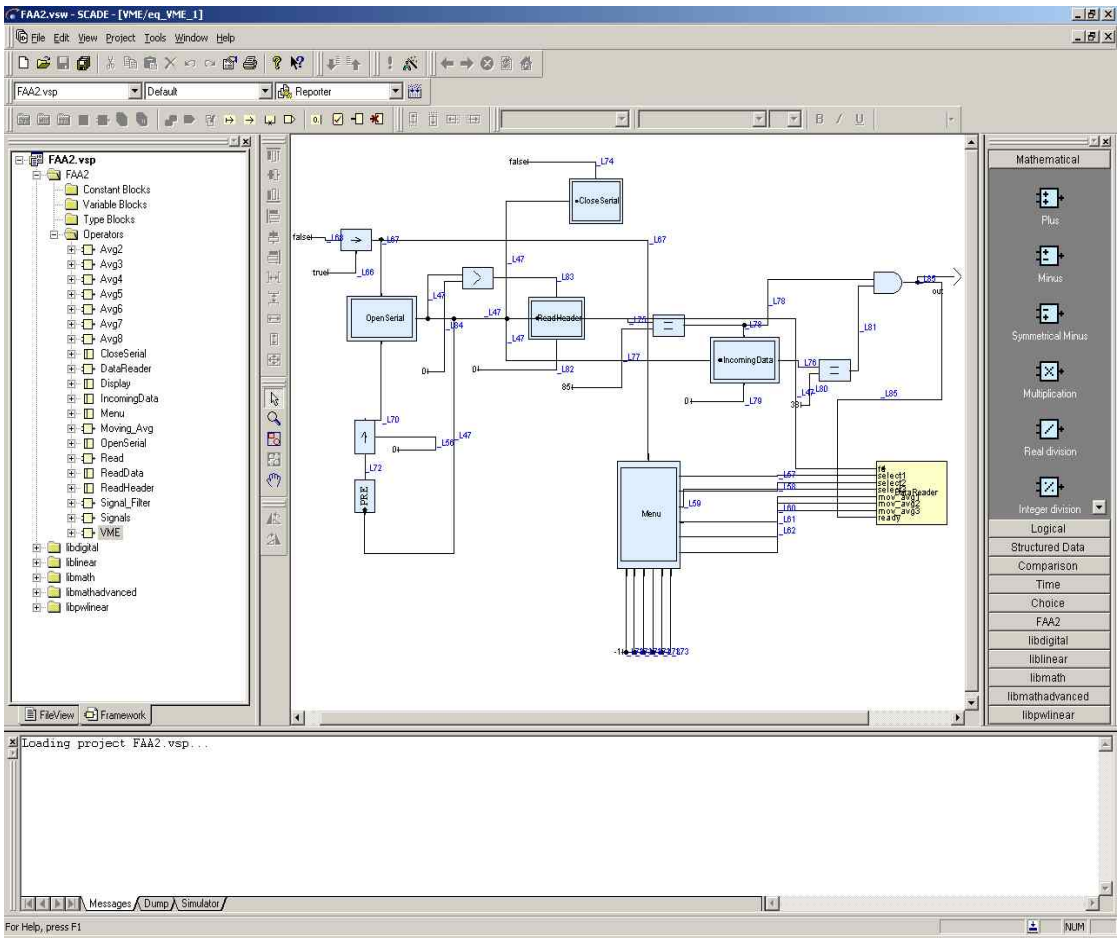
```
//    TEST 1 :
//------
//
//    Description
//       Inputs to zeros.
//    Aim
//         CAPACITY
//    inputs
//         hard : none
//         soft : mw1 = 000000H
//                    mw2 = 0000H
//
fill X:0x08028 LEN 3 with 0x00
//
fill X:0x0802B LEN 2 with 0x00
//
//    Expected outputs
//         hard : none
//         soft : mw1 = 000000H
//                    mw2 = 7FC3H
//    local data:
//         Initial value    :for i = 0 to 39 T  AC[i] = 0
//                                  :for i = 0 to 39 T  CP[i] = 0
//
fill X:0x0802D LEN 40 with 0x00
//
fill X:0x08000 LEN 40 with 0x00
//
//         Expected final values    :for i = 0 to 39 T  AC[i] = 0
//                                     :for i = 0 to 39 T  CP[i] = 0
//
//    Execution
register PC = 0x80
go until #EXPTMUMO
wait
unassemble PC
register SP
//**************
//      go until RET instruction
go until C:0x00B4
wait
unassemble PC
// display Stack pointer
register SP
//****************
// compare with previous value
//
//      run in END  EXPTMUMO
go until FIN  EXPTMUMO
wait
unassemble PC
//
//    Obtained Outputs
```

<그림 11> MC/DC를 적용한 「ADC」 SW의 테스트케이스

4. 자동화 툴의 고려사항

대규모의 프로젝트에서는 '모델링 툴(Modified Tool)'을 통하여 설계를 하고 이런 툴을 기반으로 자동화된 소스코드를 생성한다. 예를 들면 「SCADE」, 「Matlab」, 「BEACON」등을 들 수가 있겠다.

여기서 중요한 점은 자동화된 툴이 얼마나 신뢰성을 보장하는가 하는 것이다. 따라서 항공규격에서는 자동화된 툴을 사용할 경우, 그 툴에 대한 신뢰성 자료를 요구하고 있다.



<그림 12> SCADE의 설계예

```
.....
static real _L55_VME_Moving_Avg_77;
static real _L22_VME_Moving_Avg_70;
static real _L40_VME_Moving_Avg_72;
static real _L46_VME_DataReader_30;
struct {
bool _L67_VME : 1;
bool _L83_VME : 1;
bool _L78_VME : 1;
bool _L74_VME : 1;
bool _L56_VME_Moving_Avg_6 : 1;
bool _L56_VME_Moving_Avg_50 : 1;
bool _L56_VME_Moving_Avg_78 : 1;
bool _L45_VME_DataReader_29 : 1;
} _LOCAL_;
/*#code for node VME */
if ((_C_->_M_init_0_VME))
{\
_LOCAL_._L67_VME = true;
_L70_VME = 0;
}
else
{
_LOCAL_._L67_VME = false;
_L70_VME = (_C_->_M__L72_VME_1);
}
/* begin condact */
if (_LOCAL_._L67_VME)
{
/* call to macro-function m_OpenSerial */
m_OpenSerial((_C_->_L47_VME));
(_C_->_M_condact_2_VME) = false ;
}
else
{
if (_C_->_M_init_0_VME)
{
(_C_->_L47_VME) = _L70_VME;
}
}
/* end condact */
_LOCAL_._L83_VME = ((_C_-
>_L47_VME) > 0);
_L82_VME = 0;
/* begin condact */
if (_LOCAL_._L83_VME)
{
/* call to macro-function m_ReadHeader */
m_ReadHeader((_C_->_L47_VME), (_C_-
>_L75_VME));
(_C_->_M_condact_3_VME) = false ;
}
else
{
if (_C_->_M_init_0_VME)
{
(_C_->_L75_VME) = _L82_VME;
}
}
/* end condact */
.....
```

<그림 13> SCADE를 사용한 코드생성의 예

자동화된 툴을 사용하여 만드는 소스코드에 대해서는 'Tool Qualification Plan(TQP로 약칭)' 및 'Tool Qualification Summary(TAS로 약칭)' 자료가 필수이며 보통 이 같은 자료는 툴의 판매자(Vendor) 측에서 통상적으로 제공된다. 만약 이 같은 'TQP', 'TAS'가 판매자 측에서 제공되지 않는다면, 항전장비 SW의 개발에서 사용하지 않도록 권고하며, 만약 이와 같은 자격을 갖추지 못한 툴을 사용할 경우 항전 SW규격을 만족할 수 없다(관련 규격 : DO-178B §12.2)[3].

5. 결론

「ADC」는 작고 간단한 시스템이지만 'AFCS(Auto flight control system의 약칭)', 'MC(Mission Computer의 약칭)' 등과 직접적으로 인터페이스 되며 항공기 운항 전반에 영향을 주는 Safety-Critical 한 기능을 가진 시스템이다.[2]

국내에서 「ADC」 개발사례는 전무하여 실제로 개발시에 간단한 알고리즘을 구현하는데도 많은 시행착오와 시간이 소모되었다.

특히 「ADC」 SW의 개발은 항전 SW규격을 만족하기 위하여 SW 자체의 개발기간보다 SW 검증기간이 더 많이 소요되었으며, 본 논문에 소개된 SW의 고려사항인 경우 일반적인 내용이지만 국내에서는 체계적으로 정리, 연구된 사례를 찾기 힘들었다.

국내에서 본 「ADC」는 항전장비 중 유일하게 KTSO(Korean Technical Standard Order의 약칭) 인증을 신청완료하였고, 인증절차를 밟고 있다. 또한 본 연구를 바탕으로 추후 개발되는 항전장비에는 항전 SW표준인 RTCA DO-178B를 계속 적용해 나갈 계획이다. 또한 본 연구가 타 회전익 「ADC」의 개발에 조금이나마 도움이 되었으면 한다.

참고문헌

[1] Airlines Electronic Engineering Commite, *ARINC SPECIFICATION 429 PART 1-17*, AERONAUTICAL RADIO. INC, 2004.

[2] David Alberico, John Bozarth, Michale Brown, Janet Gill, Steven Mattern, Arch McKinlay VI, *Joint Software System Safety Handbook*, US DoD, 1999.12.

[3] Def Stan, *Software Consideration in Airborne Systems and Equipment Certification*, RTCA, 1992.4.

[4] SAE, *AIR DATA COMPUTER - MINIMUM PERFORMANCE STANDARD*, 1996.