Technical Paper

Journal of Aerospace System Engineering
Vol.13, No.5, pp.57-64 (2019)

ISSN 1976-6300
EISSN 2508-7150
http://dx.doi.org/10.20910/JASE.2019.13.5.57

유즈케이스를 통해 분석해 본 I/O 처리방식에 따르는 CPU처리 부하 비교연구

김재영[1,†]

[1]한화시스템

Comparison study of CPU processing load by I/O processing method through use case analysis

JaeYoung Kim[1,]

[1]Hanwha Systems

Abstract

Recently, avionics systems are being developed as integrated modular architecture applying the modular integration design of the functional unit to reduce maintenance costs and increase operating performance. Additionally, a partitioning operating system based on virtualization technology was used to process various mission control functions. In virtualization technology, the CPU processing load distribution is a key consideration. Especially, the uncertainty of the I/O processing time is a risk factor in the design of reliable avionics systems. In this paper, we examine the influence of the I/O processing method by comparing and analyzing the CPU processing load by the I/O processing method through use of case analysis and applying it to the example of spatial-temporal partitioning.

초 록

항공전자 시스템은 유지비용 감소 및 운용성능 향상을 위하여 기능을 모듈화, 통합화 설계를 적용한 모듈 통합형 항공 전자 시스템으로 개발 되어지고 있으며, 다양한 임무 제어 수행을 위해서 가상화 기술을 적용한 파티셔닝 운용체제를 적용 하고 있다. 가상화 기술을 적용 할 경우 CPU 처리 부하 분배는 중요한 고려 대상이며, 특히 입출력 처리 시간에 대한 불확실성은 안정성 있는 항공전자 시스템 설계에 있어 위험 요소 중 하나이다. 본 논문에서는 유즈케이스를 통해 입/출력 처리 방식에 따르는 CPU 처리 부하량을 비교 분석하여 공간적/시간적 파티셔닝 예시에 적용함으로써 입/출력 처리 방식의 영향성을 검토하고자 한다.

Key Words : Integrated Modular Avionics(통합모듈형 항공전자), MIL-STD-1553B, Multi-Core(멀티 코어), I/O Processing (입/출력 처리), ARINC653

1. 서 론

1940년대부터 현재까지의 항공전자 시스템은 소위 [독립형]->[연방형]->[통합모듈형(IMA)] 순으로 진화되어 왔으며, 이런 진화의 가장 큰 추진배경과 주안점은 어떻게 하면 항공전자시스템의 SWaP(Space, Weight and Power)과 비용을 최소화 하느냐에 대한 해답을 찾고 그것을 구현하여 비행 안전성(Flight Safety)에 대한 검증을 획득하는데 있었다.

이를 위해 기능들을 가능한 소프트웨어 모듈로 구현하여 하드웨어를 최소화 하고자 했으며, 하드웨어 역시 특정 제조사 혹은 모델의 종속성을 없애기 위해 Open Architecture 표준 하드웨어를 사용하게 되었다.

이것이 현재 선진 항공기 항공전자시스템의 주류가 되고 있는 통합모듈형 항공전자(IMA : Integrated

Received: Sep. 14, 2018 Revised: Oct. 11, 2019 Accepted: Oct. 15, 2019
† Corresponding Author
Tel:+82-31-8020-7765, E-mail:jaeyoung.kim@hanwha.com

Modular Avionics)시스템[1][2]의 추진방향으로 보다 많은 소프트웨어 모듈을 하드웨어에 집약하여 시스템의 SWaP과 Cost를 최소화 하는게 최종 목표이다.

IMA의 발전은 급속한 IT기술들의 발전이 뒷받침 되었기에 가능했는데 특히 고속의 멀티 코어 프로세스 등장과 가상화 기술(Virtualization)의 발전으로 코어(Core)별 서로 다른 이종(異種)의 소프트웨어 모듈의 병렬 실행(Parallel)이 가능해 지면서 소프트웨어 집약성이 보다 더 향상되었고 이를 2세대 IMA라고 일컫는다.

현재 개발 중인 한국형 전투기(KF-X) 임무컴퓨터도 ARINC-653 IMA 표준[3][4]을 적용하여, 멀티코어 프로세스에 코어별 서로 다른 이종(異種)의 소프트웨어(OS 포함)가 탑재 운용되는 비대칭 다중 처리(AMP: Asymmetric Multiprocessing)구조로 개발되고 있다. 그러나 멀티코어 프로세스는 단일 하드웨어 패키지에 다수의 Core가 집적되어 있는 최신 프로세서 기술이지만, 각 코어들은 패키지 내부와 패키지 외부에 존재하는 자원들 분할하여 사용해야하는 제약이 따른다. 이 것은 ARINC653에서 정의하고 있는 공간적 파티셔닝을 의미 한다. 효율적인 공간적 파티셔닝을 위해서는 각 입출력(I/O)에 대한 처리 시간을 분석이 필수적이다.

본 논문에서는 항공전자 시스템에서 가장 많이 사용되는 통신인터페이스인 MIL-STD-1553B를 이용하여 I/O 처리 방식에 따르는 CPU 처리 부하량을 비교 분석하여 공간적/시간적 파티셔닝 예시에 적용함으로써 I/O 처리 방식의 영향성을 검토하고자 한다.

2. 배경 지식 및 관련 연구

2.1 배경지식

2.1.1 멀티코어 기반 IMA 지원 운영체제

최근 멀티코어 기반의 가상화 기술을 이용하여 IMA를 지원하는 운영체제들이 시장에 출시되고 있으며, 공통적으로 ARINC-653 표준을 기반으로 개발되고 있다. 이들 운영체제는 주로 하드웨어에 바로 적재되어 운용되는 Baremetal(Type 1) 형태의 hypervisor를 이용한 가상화를 지원한다. 이와 같은 IMA 지원 운영체제들은 프로세서들로부터 제공되는 하드웨어 가상화 기술과 자원 중재(Arbitration) 기술들을 이용하여 캐시, 메모리, 입/출력(I/O) 파티셔닝을 지원하고, Guest OS(또는 Partitioning OS)에 해당 자원을 할당하는 형태로 가상화 환경을 제공해 준다. 가장 많이 사용되는 멀티코어 기반의 IMA 운영체제로는 WindRiver의 VxWorks 653 MCE와 GreenHills의 Integrity 178 tuMP가 있다[5].

2.1.2 공간적/시간적 파티셔닝 (Spatical/Time Partitioning)

ARINC 653의 규정에 의하면 파티션(Partition)으로 격리된 어플리케이션들은 탑재 장치 내 필요로 하는 연산자원에 대해 상호 배타적/독립적 접근해야 하며, 해당 자원을 사용하는 동안 타 파티션의 어플리케이션에 영향을 주지 않아야 한다. 또한, 특정 파티션내의 결함은 내부적으로 완벽히 격리되어 다른 파티션 운영에 영향을 주지 말아야 한다고 정의되어 있다. 이를 위해 ARINC 653에서의 파티셔닝 개념은 공간적 Partition과 시간적 파티션으로 구분하여 규정하고 있다.

Figure1과 같이 공간적 파티셔닝은 파티션별로 사용하는 연산자원의 영역을 구분하고 입출력 장치 또한 공간적 파티셔닝을 통하여 분리가 가능하다.

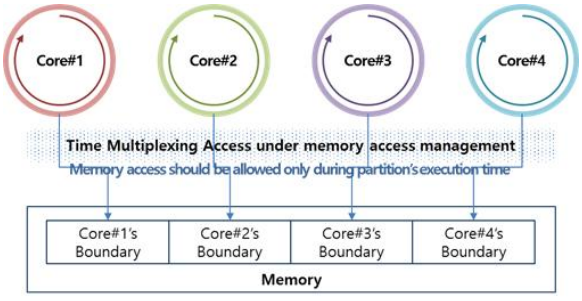

Fig. 1 Space Partitioning of Multi-Core System

Figure 2와 같이 시간적 파티션닝(Time Partitioning)은 각각의 어플리케이션(Application)들이 정해진 시간 구간(Time Slice)내에서만 연산 자원(Resource) 또는 입출력 장치 제어를 수행 하도록 시분할 접근 방식으로 파티셔닝 하여 설계해야 한다.

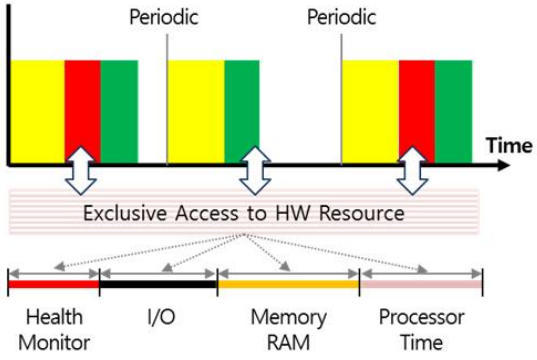

Fig. 2 Time Partitioning

이런 시분할 접근방식으로 파티셔닝 된 경우 정해진 시간 내에 할당된 연산이나 입출력제어가 모두 이루어져야 하는 제약이 있다.

2.2 관련 연구

2.2.1 파티셔닝을 위한 스케쥴링 설계 : WCET 관점

항공전자 시스템에서는 감항 인증을 위하여 여러 요구 조건이 있으며 그중 어플리케이션의 실행시간이 예측 가능하고 결정적이어야 한다는 비행 안전성 인증(Certification) 획득 요구된다. 이는 많은 입출력 제어가 필요한 IMA 시스템 설계에 많은 고려사항이 되며 관련 연구 또한 이루어지고 있다.

그 중 WCET(Worst Case Execution Time) 분석은 소프트웨어코드(Instruction) 레벨에서 자원경합 요소에 대한 시간적 분석(Timing analysis)을 수행한다. WCET 분석은 자원경합 발생 원인요소를 원천 제거하여 WCET를 최소화 한 뒤 스케줄러 설계 시 최적 Time Mapping이 되도록 설계하는 상향식 방식(Bottom-Up type)이다[6]. 이 분석은 가장 적극적이고 효과적인 방안일 수는 있으나 실제 코드의 방대함으로 'Instruction' 실행에 따른 케이스별 수행 시간을 완벽하게 분석하여 반영하는 것은 사실상 불가능하다. 하지만 특정 부분에 대한 WCET를 분석하여 스케쥴링에 반영하면 효율적인 시스템 설계가 가능하다.

3. USE CASE : DDC사 MIL-STD-1553B 시스템 소프트웨어 설계 방안 고찰

본 논문에서는 항공전자시스템에서 많이 사용하고 통신 인터페이스 중 하나인 MIL-STD-1553B 디바이스의 WCET를 분석하여 스케쥴링에 적용하기 전 입출력 제어 시스템 소프트웨어에 따르는 CPU 사용율을 분석하고자 한다. MIL-STD-1553B의 메시지 구조 특성상 많은 입출력 인터페이스 장치에 접근해야 함으로 입출력 제어 시스템 소프트웨어의 형태에 따라 CPU 사용율이 차이가 많이 나는 장치이다.

3.1 유스 케이스 시스템 개요 및 요구사항

3.1.1 유스 케이스 시스템 개요

Figure 3과 같이 제어모듈과 DDC사 MIL-STD-1533B Chip은 Address/Data Bus와 CTRL Signal 연결 된다. 제어모듈은 Address/Data Bus를 통해 1553B Chip의 Register와 Shared RAM에 접근이 가능하며 CTRL Signals을 통해 인터럽트(Interrupt)를 제어한다[7].

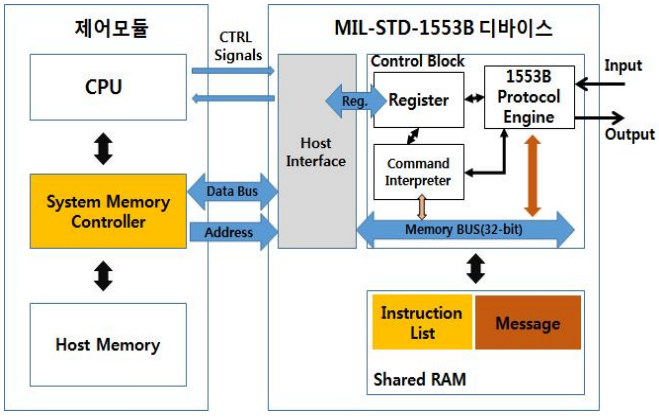

Fig. 3 MIL-STD-1553B Architecture

3.1.2 유즈 케이스 요구사항

본 논문에서는 시스템 분석을 위해 (Case 1) 폴링 방식과 호스트 접근 방식, (Case 2) 인터럽트 방식과 호스트 접근 방식, (Case 3) 인터럽트 방식과 DMA활용 접근 방식의 세 가지 유즈케이스를 선정하였다. 이 세 가지 유즈케이스의 성능 요구사항은 Table 1에 설명하였다. 성능 요구사항의 설정 값은 여러 항공기에 적용 된 임무컴퓨터의 요구사항을 분석하여 설정한 가상의 값이다.

Table 1 System Requirement

Item	Requirement
1553B Bus Usage(%)	86 %
Scheduling period(Hz)	50 Hz
1553B Channel Count	4 EA
CPU Usage(%)	7 %

개별 요구사항의 의미는 다음과 같다.

1. 1553B Bus Usage : MIL-STD-1553B의 통신 버스의 점유율 값
2. Scheduling period : 소프트웨어 수행 주기로 항공전자시스템에서 MIL-STD-1553B의 실행 주기
3. CPU Usage : 항공전자시스템의 스케줄링을 고려한 CPU 점유율 값

3.2 MIL-STD-1553B 시스템 소프트웨어 설계 방안

항공전자 시스템에서 많이 사용되고 있는 통신 인터페이스인 MIL-STD-1553B를 이용하여 I/O 처리 방식에 따른 세 가지의 시스템 소프트웨어의 설계 방안을 알아보도록 하겠다[8]. 선정 된 세 가지의 소프트웨어 설계 방안은 모두 실제 항공 전자 분야에 많이 활용 되고 있는 기술로 안정성과 성능 요구 사항에 따라 Case 1의 경우 안정성을 최우선으로 하는 비행제어 컴퓨터에 많이 적용 되었으며, 효율적인 성능을 요구 하는 임무컴퓨터에서는 Case 2 및 Case 3 방식이 주로 활용 되었다. 최근 항공전자 시스템이 통합 모듈형 항공전자 시스템(IMA)으로 발전함에 따라 해당 I/O 처리 방식에 따른 성능 자료는 중요한 설계 기초 자료가 된다.

3.2.1 Case 1 : 폴링 방식 과 호스트 접근 방식

첫 번째 방안은 Fig. 4와 같이 제어 메시지와 상태 확인을 주기 적으로 수행하고 CPU에서 1553B 디바이스 메모리에 직접 접근하여 자료(Data)를 획득 하는 방법이다.

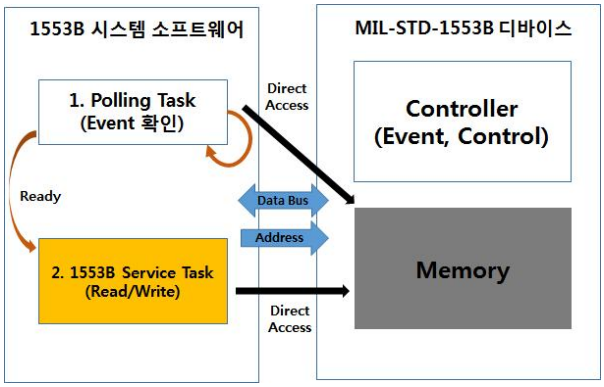

Fig. 4 Case1:Polling and host access

Case1의 1553B 시스템 소프트웨어 동작 구성은 다음 순서와 같다.

1. Polling Task : 주기적으로 1553B 디바이스의 Control Register를 1 ms 단위로 확인 하여 메시지 송/수신 상태를 확인 한다. 1553B 디바이스의 Event 확인 시 1553B Service Task로 제어권을 넘긴다.
2. 1553B Service Task : Polling Task로부터 수신한 Event 정보를 이용하여 1553B 메시지를 가져오거나 전송한다. 이때 사용되는 메모리는 1553B 디바이스의 메모리에 직접 접근한다.

이 설계 방법은 장점은 소프트웨어 설계 시 예측성(Predictability) 높다는 것으로 이는 항공전자 시스템 소프트웨어 설계 시 중요한 요소이다. 하지만 1553B 디바이스의 상태와 상관없이 주기적으로 1553B 디바이스 상태를 확인하는 것은 CPU의 불필요한 자원낭비가 발생 할 수 있다. 그래서 입출력 장치의 특성을 잘 파악하여 주기 설정하는 것이 필요하다. MIL-STD-1553B의 메시지 형태는 여러 형태가 있지만 1~32 word 크기의 작은 메시지들이 Fig.5 와 같이 프레임을 이루어 메모리에 저장되는 형태 이다. 이런 특성을 고려하여 폴링 주기를 1 ms 주기로 설정하였다.

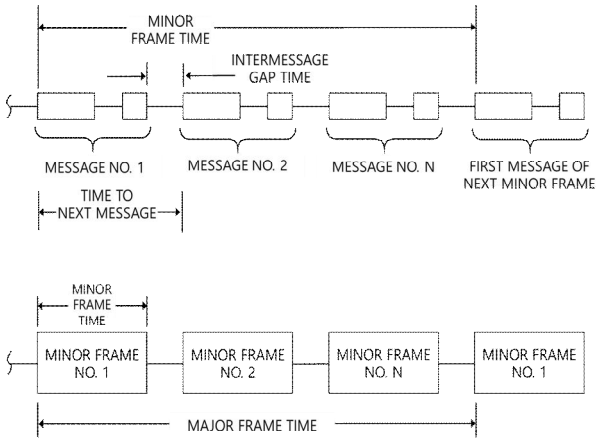

Fig. 5 Communication structure of 1553B Message

3.2.2 Case 2 : 인터럽트 방식과 호스트 접근 방식

두 번째 방안은 Fig. 6과 같이 제어 메시지와 상태 확인을 인터럽트를 통해 수신하여 확인을 CPU에서 1553B 디바이스 메모리에 직접 접근하여 자료(Data)를 획득 하는 방법이다.

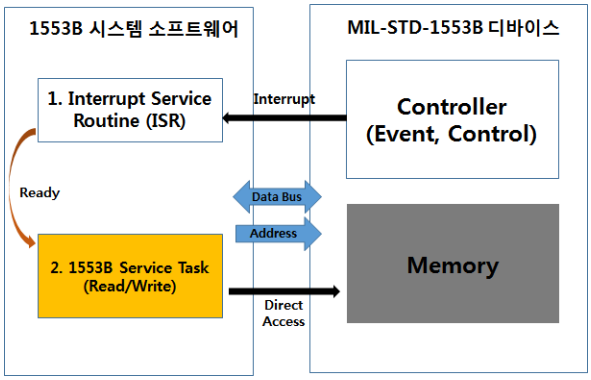

Fig. 6 Case 2: Interrupt and host access

Case2의 1553B 시스템 소프트웨어 동작 구성은 다음 순서와 같다.

1. Interrupt Service Routine : 1553B 디바이스는 데이터의 송신이 준비 되었을 때 또는 1553B 메시지가 수신되었을 때 인터럽트를 발생 시켜 해당 디바이스 상태가 변경되었음을 시스템 소프트웨어로 전달 할 수 있다. 시스템 소프트웨어는 수신 된 인터럽트의 종류를 확인하고 해당 정보와 함께 1553B Service Task에게 전달한다.
2. 1553B Service Task : 'Interrupt Service Routine' 으로부터 수신한 이벤트(Event) 정보를 이용하여 1553B 메시지를 가져오거나 전송한다. 이때 사용되는 메모리는 1553B 디바이스의 메모리에 직접 접근한다.

이 설계 방법은 'Polling Task' 에 비해 불필요한 CPU 자원 소비가 일어나지 않는 장점이 있다. 그러나 Fig. 7과 같은 인터럽트 발생 시 복잡한 과정을 거치기 때문에 시스템 소프트웨어 설계 시 예측성(Predictability)은 폴링(Polling) 방식에 비해 낮은 편이다.

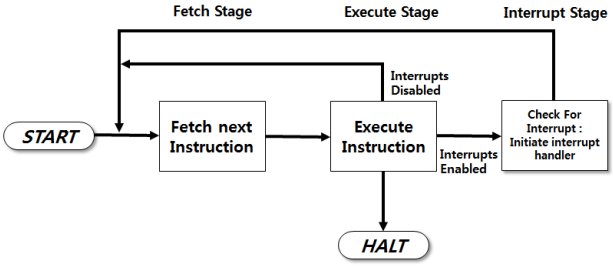

Fig. 7 Instruction Cycle with Interrupts

3.2.3 Case 3 : 인터럽트 방식과 DMA 활용 접근 방식

세 번째 방안은 Fig. 8과 같이 제어 메시지와 상태 확인을 인터럽트를 통해 수신하고 자료는 CPU에서 'Host Memory' 에 접근하여 획득 하는 방법이다.

시스템 제어 소프트웨어는 Host Memory에 Inbound 영역(DMA 사용 영역)을 설정하고 외부 디바이스에서 접근 가능하도록 허용한다. MIL-STD-1553B는 Scatter-Gather DMA[9]를 지원한다. 일반적인 DMA Controller가 연속된 메모리 공간의 전송만을 지원하지만 Scatter-Gather DMA는 좀 더 발전해서 자료전송에 대해서 정의한 여러 개의 디스크립터(Descriptor)들을 이용해서 연속적인 메모리 공간에서 비연속적인 메모리 또는 반대도 지원 하는 형태이다. 이는 1553B 메시지 형태 및 저장 구조에 적합한 형태의 DMA 기술이다.

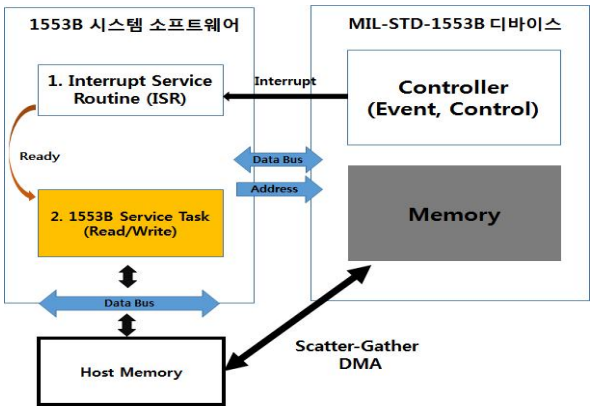

Fig. 8 Case 3: Interrupt and DMA

Case3의 1553B 시스템 소프트웨어 동작 구성은 다음 순서와 같다.

1. Interrupt Service Routine : 1553B 디바이스는 데이터의 송신이 준비 되었을 때 또는 1553B 메시지가 수신되었을 때 인터럽트를 발생 시켜 해당 디바이스 상태가 변경되었음을 시스템 소프트웨어로 전달 할 수 있다. 시스템 소프트웨어는 수신 된 인터럽트의 종류를 확인하고 해당 정보와 함께 1553B Service Task에게 전달한다.
2. 1553B Service Task : 'Interrupt Service Routine' 으로부터 수신한 Event 정보를 이용하여 1553B 메시지를 가져오거나 전송한다. 이때 사용되는 메모리는 Host Memory의 Inbound 영역으로 설정된 영역을 사용한다.

Figure 9는 DMA기능을 지원하는 DDC사 1553B를 도식화 한 것으로 Scatter-Gather DMA적용한 1553B 디바이스와 Host Memory를 나타내고 있다.

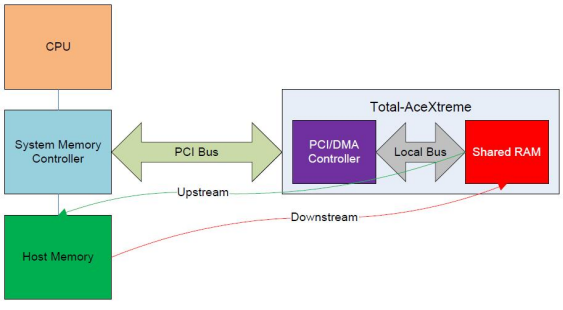

Fig. 9 Example : 1553B Device of DDC Inc.

이 1553B 시스템 소프트웨어의 장점은 CPU는 Host Memory 접근만으로 1553B 수신 메시지 및 디바이스 제어가 가능하여 CPU 사용율을 최소화 할 수 있다는 장점이 있다. 그러나 두 번째 방식과 같이 인터럽트를 사용하여야 하기 때문에 예측성(Predictability)은 Polling 방식에 비해 낮은 편이며, 1553B 메시지 수신 시 디바이스에서 업데이트(Update)하는 방식임으로 1553B 시스템 소프트웨어에서 제어하기가 복잡한 단점이 있다. 또한 1553B 디바이스가 Scatter-Gather DMA를 지원하고 시스템 설계 시 반영이 필요한 물리적 제약이 있다.

3.3 MIL-STD-1553B 시스템 소프트웨어 설계 Case 비교

Table 2 Comparison of Cases

	Case 1	Case 2	Case 3
Control Predictability	High	Mid.	Low
Control Complexity	High	Mid.	Low
CPU Usage	High	High	Low

Table 2와 같이 MIL-STD-1553B 디바이스의 입/출력 처리 방식에 따라 CPU 사용율 및 디바이스의 제어복잡성 및 예측성에 차이가 있을 것으로 분석 된다.

4 시스템 소프트웨어 형태에 따른 성능 시험

4.1 시험 환경 및 조건

본 논문에서는 모의용 항전장비와 1553B 분석기를 이용하여 Fig. 10과 같이 시험 환경을 구성하였다.

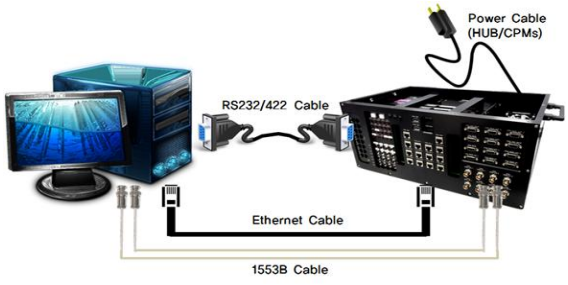

Fig. 10 Test Environment

1553B 메시지의 동작 주기는 50Hz, 1553B Bus 부하율은 86%가 되도록 시험 소프트웨어를 구현하여 3가지 방식에 대한 CPU 사용율을 측정하였다.

4.2 시험 결과 및 분석

Table 3과 같이 시험 조건에 따라 유즈케이스 별 10회 시험을 수행하여 유휴 시간(Idle Time)과 CPU 사용율(CPU Usage)을 값을 측정하였다(Table 3). CPU 사용율은 주기적인 50Hz 동작에서 20ms 기준을 100 % 환산하여 측정 한 값이다.

Table 3 Summery of Test Results

	IDLE (%)	CPU Usage (%)
Case1	81.30 %	18.70 %
Case2	84.60 %	15.40 %
Case3	97.10 %	2.90 %

해당 시험 결과를 통해 동일한 MIL-STD-1553B 디바이스지만 입/출력(I/O) 처리 방식에 따라 CPU 사용량이 크게 차이가 나는 것을 확인 할 수 있었다.

시험 데이터를 활용하여 항공전자 시스템에 적용하기 위해 시간 값으로 환산하면(20 ms = 100 %) 다음과 같다.

Case 1 : 3.74 ms (= 20ms * 0.187)
Case 2 : 3.08 ms (= 20ms * 0.154)
Case 3 : 0.58 ms (= 20ms * 0.029)

5. I/O 처리 방식에 따른 시험 결과를 이용하여 공간적/시간적 파티셔닝 활용 예시

시험 결과를 이용하여 가상의 항공전자 시스템의 공간적/시간적 파티셔닝에 적용하는 예제를 알아보도록 하겠다.

5.1 공간적 파티셔닝

시스템 관점에서 임무에 대한 필요 자원 기준으로 공간적 파티셔닝 한 예시를 Fig. 11에서 나타내고 있다.

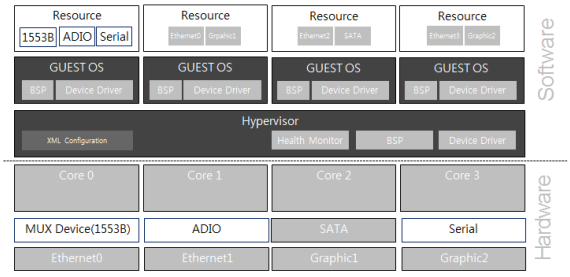

Fig. 11 Example : Space Partitioning

공간적 파티셔닝은 해당 파티셔닝에 할당된 기능 수행을 위한 입출력 인터페이스 항목이 최우선 고려 사항이다. 예를 들어 Core0은 임무제어 및 무장제어를 위하여 ADIO, Serial, 1553B 자원이 필요하다고 가정하고 Table 1의 시스템 요구사항에서 1553B 통신 채널이 4개라고 할 경우 한 파티션에 모두 할당 할 경우 I/O 처리 방식 중 Case 1과 Case 2는 각각 74.92 %(18.73% * 4채널) 과 61.6%(15.4% * 4채널)로 CPU 점유율 20% 이상으로 부적합 함을 알 수 있다. 하지만 4개의 파티션에 하나씩 공간적 파티셔닝을 하고 파티셔닝 간 데이터 통신 채널만 연결한다면 3가지 Case 모두 사용가능 한 것을 알 수 있다.

5.2.2 시간적 파티셔닝

임무제어와 무장제어를 수행하기 위한 시간적 파티셔닝 예시는 Fig. 12에 설명되어 있다. 임무제어와 무장제어 그리고 다른 입/출력(I/O) 처리는 분석을 통해 스케쥴링 설계에 반영 되었으며, 미 할당 된 시간은 Fig 12와 같이 T1(2 ms), T2(2.5 ms), T3(5 ms) 이다.

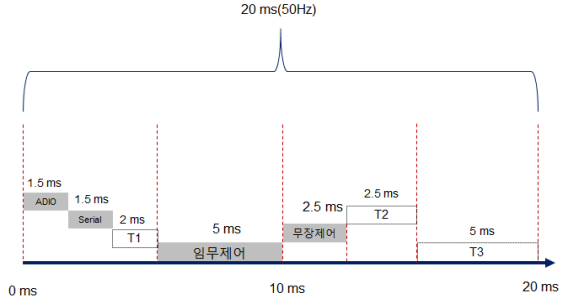

Fig. 12 Example : Time Partitioning

입/출력(I/O) 처리방식 별 할당 가능한 시간적 파티셔닝 결과는 Table 4와 같다. 시험 결과를 통해 Case1과 Case2의 경우 각각 3.74 ms 와 3.08 ms 스케쥴링 시간이 필요하여 T3에만 할당이 가능한 반면, Case 3의 경우에는 0.56ms 시간이 필요함으로 T1/T2/T3 모두 할당이 가능 하여 시간적 파티셔닝 시 제약이 없음을 알 수 있었다.

Table 4 Summery of Time Partitioning

	T1	T2	T3
Case 1	X	X	O
Case 2	X	X	O
Case 3	O	O	O

6. 결 론

본 논문에서는 항공전자 시스템에 많이 사용되고 있는 MIL-STD-1553B 디바이스를 입/출력(I/O) 처리 방식에 따라 세 가지 유즈케이스로 선정하였다. 선정된 유즈케이스는 폴링 처리와 호스트 접근 방식, 인터럽트 처리와 호스트 접근 방식 마지막으로 인터럽트 처리와 DMA 접근으로 방식이다. 유즈케이스 각각의 처리 방식에 맞게 시스템 소프트웨어를 설계하고 시스템 요구사항을 설정하여 테스트를 수행하였다. 이 시험 결과 입/출력(I/O) 처리 방식에 따르는 CPU 점유율의 차이가 발생하고, 이와 같은 차이로 인해 입/출력 처리 방식 중 일부 방식에 대하여 시간적 공간적 파티셔닝 적용 시 제약이 있는 것을 확인하였다. 입/출력(I/O) 방식 중, 인터립트와 DMA를 이용한 방식은 파티셔닝에 대한 제약은 없으나 하드웨어적인 제약이 존재하므로 적용 시 추가적인 고려가 필요한 부분이다.

본 논문의 결과를 통해 확인할 수 있듯이 입/출력(I/O) 처리 방식에 따르는 CPU 처리 부하율의 차이에 의한 영향성은 파티셔닝에 있어서 매우 중요한 요소이며, 본 논문에서 사용한 유즈케이스 외 다른 입출력 디바이스들에 대한 분석이 추가적으로 이루어진다면, 파티셔닝 적용 시 기존에 예측이 어려웠던 입출력 디바이스의 WCET 예측이 가능하게 되고, 이를 통해 실시간 스케쥴링의 정확도 향상과 전체적 시스템의 실시간 특성을 높일 수 있을 것으로 판단된다.

References

[1] Prisaznuk P.j, “Integrated Modular Avionics”, *Proceedings of the IEEE 1992 National Aerospace and Electronics Conference,* 1992

[2] Han-Jonn Park, “Design Method for Integrated Modular Avionics System Architecture”, *The Journal of Korean Institute of Communications and Information Sciences,* pp. 1094-1103, November, 2013

[3] ARINC 653 Standard, “Avionics Application Software Standard Interface,” 2006

[4] Sławomir Samolej, “ARINC Specification 653 Based Real-Time Software Engineering”, *e-Informatica Software Engineering Journal,* Volume 3, Issue 1, 2009.

[5] Rudolf Fuchsen, “How to address Certification for Multi-Core Based IMA Plattforms”, *Professional Articles in SYSGO.*

[6] Sergey Zhuravlev, “Addressing Shared Resource Contention in Multicore Processors via Scheduling”, *ASPLOS,* 2010

[7] “AceXtreme Architecture”, and “Total-AceXtreme Ultra-Small, Ultra-Low Power MIL-STD-1553 Single Package Solution” at www.ddc-web.com

[8] Cheol-Hea Koo, “Method of data processing through polling and interrupt driven I/O on device data”, *International journal of aeronautical and space sciences*, pp. 113-119, September, 2005

[9] “Direct Memory Access” at www.wikipedia.org