논문 2014-09-26

EcoHILS를 활용한 효율적인 CPS 시험에 관한 연구

(A Research on Effective Cyber-Physical Systems Tests Using EcoHILS)

김 민 조, 강 성 주, 전 인 걸*, 김 원 태
(Min-Jo Kim, Sungjoo Kang, In-Geol Chun, Won-Tae Kim)

Abstract : Cyber-Physical Systems(CPS) that mostly provides safety-critical and mission-critical services requires high reliability, so that system testing is an essential and important process. Hardware-In-the-Loop Simulation(HILS) is one of the extensively used techniques for testing hardware systems. However, most conventional HILS has problems that it is difficult to support a distributed operating environment and to reuse a HILS platform. In this paper, we introduce EcoHILS(ETRI CPS Open Human-Interactive hardware-in-the-Loop Simulator) in order to test CPS effectively. Moreover, feasibility tests and performance tests of EcoHILS are performed to confirm its effectiveness.

Keywords : Cyber-Physical Systems, Hardware-In-the-Loop Simulation, EcoHILS, M&S

Ⅰ. 서 론

임베디드 컴퓨팅과 하드웨어 기술 그리고 통신 기술들의 발전은 Cyber-Physical Systems(CPS)라는 새로운 패러다임을 제시했다. CPS는 시스템의 운용 목적을 달성하기 위해 물리적인 프로세스와 상호작용을 하는 센서, 액추에이터, 제어기 등의 물리적인 요소와 이를 제어하기 위한 컴퓨팅 요소가 결합된 시스템이다 [1]. CPS는 네트워크를 통해 다수의 이종 시스템들이 상호작용하고 각각의 시스템은 복잡한 물리 환경에서 운용되므로, 예측 불가능하고 불확실한 상황을 처리할 수 있는 고신뢰성을 필요로한다 [2]. 뿐만 아니라 주로 안전 필수(safety-critical) 및 임무 필수(mission-critical)적인 서비스를 제공하는 CPS의 오작동은 사람들의 목숨 및 재산상에 큰 위험요소로 이어질 수 있기 때문에, CPS 개발 시 철저한 시스템 테스트를 필요로 한다. 이를 위해, 본 연구에서는 일반적으로 소프트웨어를 포함한 하드웨어 시스템을 테스트하는데 많이 사용되고 있는 방법 중 하나인 Hardware-In-the-Loop Simulation(HILS)을 이용한다. 그러나 기존 HILS 방법은 CPS의 분산 운영 환경 지원과 HILS 플랫폼 재사용 측면에 문제가 있다. 따라서 이전 연구에서는 이 문제를 해결하고 체계적인 CPS 개발 단계에 HILS를 적용하기 위해 EcoHILS(ETRI CPS Open Human-Interactive hardware-in-the-Loop Simulator)를 제안하였다 [3]. CPS의 응용 범위가 우주항공, 자동차, 로봇, 스마트 그리드, 국방 등 매우 넓으므로, EcoHILS는 다양한 시스템에 일반적으로 사용될 수 있도록 고안 되었다. 따라서 본 논문에서는 EcoHILS가 다양한 CPS 시험에 효율적으로 활용 가능함을 검증하기 위해 AR.Drone을 목표 시스템으로 하는 EcoHILS를 구축하여 실현 가능성을 확인한다. 또한 EcoHILS에서 발생할 수 있는 시간지연 측정을 통한 성능 평가 결과와, 이를 통해 HILS 수행 시 목표 시스템에서 테스트 가능한 데이터양을 측정하는 방법을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 HILS의 개요 및 적용 사례에 대해 설명하고, 3장에서는

*Corresponding Author(igchun@etri.re.kr)
Received: 3 Feb. 2014, Revised: 25 Mar. 2014, Accepted: 15 Apr. 2014.
M.J. Kim, S. Kang, I.G. Chun, W.T. Kim: ETRI
※ 본 연구는 미래창조과학부 및 한국산업기술평가관리원의 산업원천기술개발사업의 일환으로 수행하였음. [10035708, 고신뢰 자율제어 SW를 위한 CPS(Cyber-Physical Systems) 핵심 기술 개발]

 2014 Aug.: 9(4) 211-217
ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2014.9.4.211

그림 1. EcoHILS 구조
Fig. 1 Architecture of EcoHILS

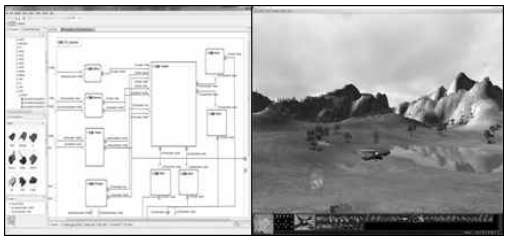
EcoHILS의 구성요소 및 인터페이스 모듈에 대해서 설명한다. 4장에서는 EcoHILS의 동작 시험 및 정량적 성능 평가를 수행하여 그 결과를 기술하며, 마지막으로 결론 및 향후 연구에 대해 기술한다.

II. 관련연구

HILS는 하드웨어 시스템을 실제 운영 환경에서 테스트하기 이전에, 시스템의 기능 및 성능을 검증하는 방법 중 하나이다. HILS는 실제 환경에서 수행하기 힘든 반복적이고 재현 가능한 테스트 및 다양한 시나리오를 적용할 수 있으며, 전체 하드웨어 시스템이 개발되기 전에 일부 하드웨어 및 소프트웨어의 테스트가 가능하다. 이러한 장점들은 시스템 개발 시 비용과 시간을 크게 감소시킬 수 있기 때문에 HILS는 다양한 산업 분야에서 사용되고 있다 [4]. 우주항공 분야의 예로, KARI에서는 Tilt rotor UAV의 FCC(Flight Control Computer)에 포함된 오류 진단 기능을 평가하기 위하여 HILS 시스템을 구성하고 오류 주입을 통한 테스트를 수행한다 [5]. 국방 분야의 예로, Xin Li는 항공유도탄의 개발 및 평가를 위해, 항공유도탄의 유도를 위한 방정식 및 궤도 디자인 방법 그리고 제어 방법을 제안하고 HILS를 통해서 성능을 확인한다 [6]. 자동차 분야의 예로서, Tulga Ersal는 지리적으로 멀리 떨어져 있는 차량의 엔진과 운전 시뮬레이터를 연동하는 HILS 시스템을 구축하여 엔진 성능 테스트를 수행한다 [7].

III. EcoHILS 구조

그림 1을 참고 하면, EcoHILS는 목표 시스템, EcoPOD(ETRI CPS Open PlatfOrm Developer), EcoSIM(ETRI CPS Open SIMulator)으로 구성됨을 볼 수 있다 [3]. 목표 시스템은 HILS 테스트 대상



(A) (B)

그림 2. (A) EcoPOD의 CPS 모델링 화면, (B) 시각화 도구를 이용한 EcoSIM 결과 출력
Fig. 2 (A) CPS Modeling of EcoPOD, (B) Result of EcoSIM Using Visualizer

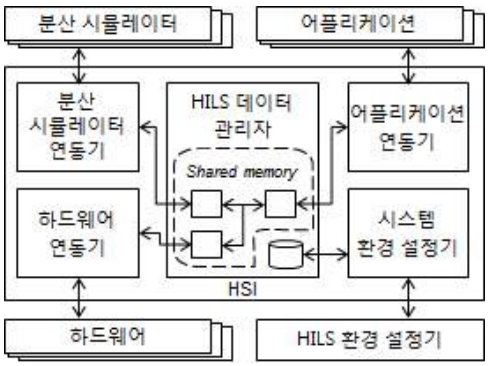


그림 3. HSI 구조
Fig. 3 Architecture of HSI

이 되는 CPS를 이루고 있는 이종 시스템들이며, 필요에 따라 사용자와 상호작용이 가능한 입출력 장치가 될 수도 있다. EcoPOD는 ECML(ETRI CPS Modeling Language)을 바탕으로 CPS를 모델링하며, 이를 EcoSIM에 사용되는 시뮬레이션 코드로 변환한다 [8]. 또한 코드 변환 시 시뮬레이션 모델과 목표 시스템의 연동을 위해 HILS 환경 설정기가 포함된다. EcoSIM은 분산 시뮬레이터로써 EcoPOD에서 생성된 코드를 기반으로 시뮬레이션 모델들의 입출력 및 사용자와의 연동을 위한 데이터를 처리하고, 물리 환경을 모사한다 [9]. 그림 2의 (A)는 EcoPOD의 모델링 화면을 나타내며, (B)는 EcoSIM의 시뮬레이션 결과를 시각화 도구를 통해 나타낸 화면이다. 목표 시스템에 공통적으로 사용되는 HSI(HW-SW Interconnector) 모듈은 그림 3의 구조와 같이 분산 시뮬레이터 연동기, 어플리케이션 연동기, 하드웨어 연동기, 시스템 환경 설정기, HILS 데이터 관리자로 구성된다. 분산 시뮬레이터

```
<Data Format> ::= <HW Name> {<Interface Info>}
<HW Name> :: = "하드웨어 이름" <EOL>
<Interface Info> ::= <Name> <Type> <Attr> <Data
                     Type> <Sampling rate>
                     <Path> <Connected Interface>
                     <EOL>
<Name> ::= "인터페이스 이름"
<Type> ::= "Sen" | "Act"
<Attr> ::= "Read" | "Write"
<Data Type> ::= "int" | "float" | ... | "100KB"
<Sampling rate> ::= "샘플링 주기"
<Path> ::= "SIM" | "BOTH" | "NULL"
<Connected Interface> ::= "연동 인터페이스 이름"
```

그림 4. HILS 환경 설정을 위한 인터페이스 정보형식 정의
Fig. 4 Definition of Interface Information Format for HILS Setting

연동기는 분산 시뮬레이터와 HILS 데이터 관리자 사이에서 데이터를 주고받는 기능을 제공하며, 어플리케이션 연동기는 범용 API 제공을 통해 어플리케이션과 HILS 데이터 관리자 사이의 데이터 교환 기능을 제공한다. 하드웨어 연동기는 하드웨어와 HILS 데이터 관리자 간의 데이터를 디바이스 드라이버를 이용하여 교환하며, 시스템 환경 설정기는 HILS 환경 설정기와 연동하여 하드웨어 인터페이스 정보를 전달하고, 데이터 경로 및 샘플링 주기 정보를 받아 HILS 데이터 관리자에 저장한다. 마지막으로 HILS 데이터 관리자는 다양한 환경 설정에 따라 분산 시뮬레이터 연동기, 어플리케이션 연동기, 하드웨어 연동기 사이에서 데이터 경로 관리 및 샘플링 주기 조절 기능을 제공한다.

그림 4는 시뮬레이션에 앞서 HILS 환경 설정을 위해, HSI와 HILS 환경 설정기 간에 주고받는 인터페이스 정보 형식을 Extended Backus-Naur Form을 이용하여 정의한 것이다. <HW Name>은 HILS 환경 설정기에서 여러 목표 시스템을 구분하기 위해 사용되며, <Name>은 실제 연동 가능한 인터페이스의 이름이다. <Type>은 인터페이스가 센서 또는 액추에이터인지를 구분하기 위해 사용되며, <Attr>은 인터페이스가 데이터를 입력받는지 또는 출력하는지를 구분하기 위해 사용된다. <Data Type>은 데이터 교환에 사용되는 변수 타입 및 배열의 크기를 나타내며, <Sampling rate>는 기본적으로 설정된 데이터의 샘플링 주기를 나타내며, 환경 설정 시 변경이 가능하다. <Path>는 인터페이스

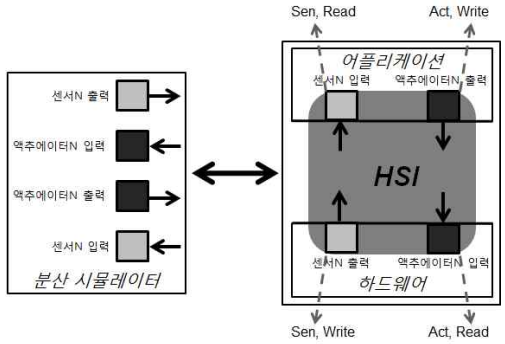


그림 5. HILS를 위한 인터페이스 분류
Fig. 5 Interface Classification for HILS

가 분산 시뮬레이터와 연동하는지 또는 분산 시뮬레이터 및 시스템에 포함된 인터페이스와 동시에 함께 연동되는지를 나타낸다. NULL 값은 인터페이스 연결을 하지 않는 경우이다. 분산 시뮬레이터와 연동이 있는 경우에는<Connected Interface>에 기술되는 분산 시뮬레이터의 인터페이스 이름을 사용하게 된다. <Path> 및 <Connected Interface>는 환경 설정 이전에는 사용되지 않으며, 환경 설정 이후에 추가되어 전달되는 정보이다. 그림 5를 보면, HSI에서 임의의 센서 및 액추에이터의 인터페이스는 입출력 속성(Read 또는 Write)에 의해서 두 개씩 존재함을 확인할 수 있다. 이는 분산 시뮬레이터와 인터페이스 연동을 위해, 센서 및 액추에이터에 따라 각각 입출력 인터페이스를 필요로 하기 때문이다. HSI에 포함된 각각의 인터페이스는 분산 시뮬레이터의 인터페이스와 연동이 가능하며, 액추에이터 출력 및 센서 출력 인터페이스의 경우에는 분산 시뮬레이터 및 자신의 시스템에 포함된 인터페이스와 동시에 연동이 가능하다. 따라서 이러한 환경 설정을 통해 EcoHILS에서는 다양한 데이터 흐름으로 시뮬레이션을 수행할 수 있다.

Ⅳ. 구현 및 평가

본 장에서는 EcoHILS의 동작 시험을 위해 AR.Drone을 제어하기 위한 EcoHILS를 구축하여 테스트를 수행한다. 그다음으로 EcoHILS에 포함된 HSI로 인해 발생할 수 있는 시간지연을 측정하여, 시뮬레이션에 주는 영향에 대해서 평가한다.

1. 동작 시험

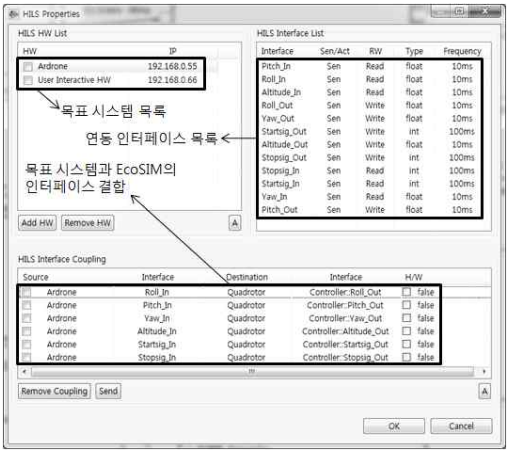


그림 6. HILS 환경 설정기 화면
Fig. 6 Screenshot of HILS Configurator

본 논문의 3장에서 설명한 구성요소를 기반으로 AR.Drone을 제어하기 위한 EcoHILS를 구축하였다. 구성 요소를 보면 목표 시스템으로는 압력 센서, 초음파 센서, 3축 자이로스코프, 가속도, 지자계 센서 그리고 Brushless inrunner 모터 등으로 이루어진 AR.Drone이 사용된다. 뿐만 아니라 사용자와의 상호작용을 위해 조이스틱 및 디스플레이 장치도 목표 시스템으로 사용된다. 또한 EcoPOD는 CPS모델링 및 HILS 환경 설정을 위해 사용되고, EcoSIM은 분산 시뮬레이터 역할을 한다. HSI는 AR.Drone, 조이스틱, 디스플레이 장치에 공통적으로 사용된다. 테스트 시나리오는 다음과 같다. 사용자가 EcoPOD를 이용해 HILS 환경 설정을 한 후, 조이스틱 조종을 통해 EcoSIM에 조종 입력 값을 전달한다. EcoSIM은 이 입력 값을 가지고 시뮬레이션을 수행하여, 생성된 기체 제어 데이터를 AR.Drone에 전달한다. AR.Drone은 전달받은 데이터를 기반으로 비행하고, 비행 중 생성되는 자세, 고도, 배터리양 등의 센서 값을 EcoSIM으로 전달한다. 마지막으로 EcoSIM이 전달받은 센서 값을 시뮬레이션을 통해 디스플레이 장치로 전달하고, 디스플레이 장치는 전달받은 센서 값을 출력하여 사용자에게 보여준다. 테스트에서는 시나리오에 맞게 AR.Drone이 제어되는지와 HSI에서의 데이터 흐름을 확인한다.

그림 6은 EcoPOD의 HILS 환경 설정기 화면이다. HILS 환경 설정기에서 환경 설정을 하고자 하는 목표 시스템의 IP 주소를 입력하면, 해당 시스템에 포함된 HILS 설정이 가능한 인터페이스 정보가

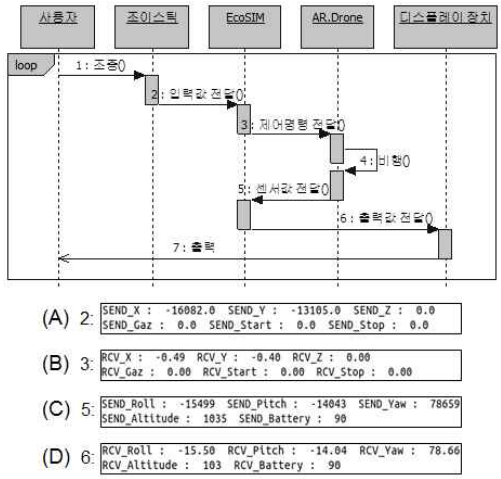


그림 7. AR.Drone 제어 EcoHILS 동작 흐름
Fig. 7 Operation Flow of EcoHILS for AR.Drone Control

HSI로부터 전달되어 화면에 표시된다. 이 정보를 바탕으로 사용자가 목표 시스템에 데이터 경로 및 샘플링 주기를 설정해준다. 시뮬레이션은 경로 설정이 끝나면 시작된다. 테스트는 시나리오에 맞게 그림 7의 동작 흐름대로 진행되고, 이때 (A)는 조이스틱, (B), (C)는 AR.Drone, (D)는 디스플레이 장치에서 HSI를 통해 교환되는 데이터를 나타낸다. 테스트 결과를 통해서 EcoHILS가 다양한 시스템의 HILS를 위해서 사용될 수 있는 실현 가능성을 확인하였다. 또한 테스트 환경 구축에 있어, HSI를 범용적으로 사용하여 목표 시스템을 분산 환경에 적용하기 용이하였다. 뿐만 아니라 HILS 환경 설정기를 통해서 시뮬레이션 데이터의 경로 및 샘플링 주기 관리를 편리하게 할 수 있었다.

2. 정량적 성능 평가

EcoHILS의 운영 시 HSI로 인한 시간지연이 발생할 수 있다. 이러한 오버헤드가 실제 목표 시스템의 어플리케이션 데이터 갱신 주기에 큰 영향을 끼치게 되면, 시스템의 올바른 동작을 보장할 수 없다. 따라서 HSI로 인한 시간지연 발생은 최소한으로 작아야 하며, 이를 확인하고 평가하기 위한 테스트가 필요하다. 그림 8은 HSI 사용으로 인해 발생할 수 있는 시간지연을 측정 하는 방법이다. 시간지연은 HSI를 통해 어플리케이션에서 데이터를 주고받는 4가지 경우 (A+B+C , A+B+D) 중 가장 큰 시간 값과 HSI 사용 없이 어플리케이션에서 데

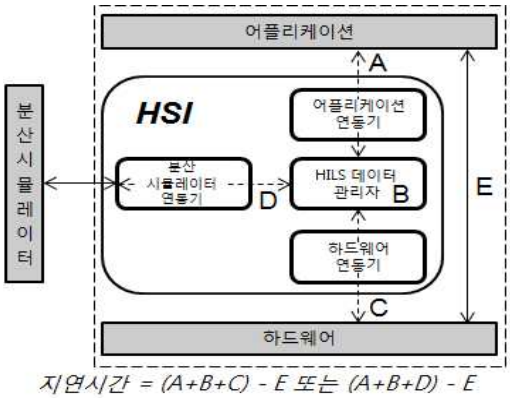


그림 8. HSI 모듈 시간지연 측정 방법
Fig. 8 Method for Time Delay Measurement of HSI

표 1. 테스트용 ARM 보드 사양
Table 1. Specification of ARM Board for Test

MCU	Samsung S5PV210 1GHz
RAM	512Mbyte MDDR
ROM2	512Mbyte NAND-Flash
OS	Linux(kernel 2.6.35)

표 2. HSI 모듈 시간지연 측정 결과
Table 2. Result of Time Delay Measurement of HSI

	1KB	10KB	100KB
무부하 환경	0.046 ms	0.120 ms	1.177 ms
CPU 부하 40%	0.071 ms	0.204 ms	1.817 ms
CPU 부하 80%	0.138 ms	0.421 ms	3.590 ms

이터를 주고받는 2가지 경우(E) 중 가장 작은 시간 값의 차이를 측정하였다. 테스트는 ARM 보드의 CPU가 무부하, 부하 40%, 부하 80%인 3가지 경우와 데이터 크기가 1KB, 10KB, 100KB인 3가지 경우에 따라 9가지 조건에서 이루어진다. 테스트에 사용된 ARM 보드의 사양은 표 1과 같다. 표 2는 HSI 사용으로 인해 발생하는 함수 호출 1회당 평균 데이터 처리 시간지연을 나타낸다. 표 2 를 참고하면 HSI 사용 시 데이터 크기가 1KB에서 10KB로 증가하면 시간지연은 평균 2.8배, 10KB에서 100KB로 증가하면 시간지연은 평균 9.1배로 CPU

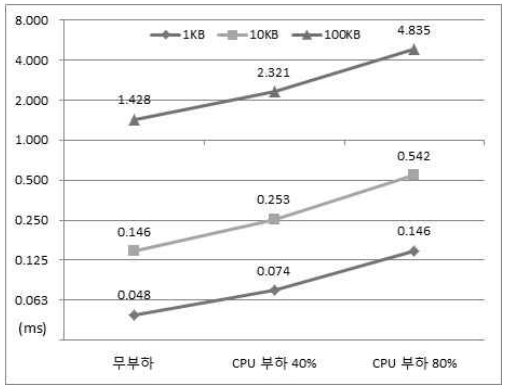


그림 9. HSI의 CPU 부하별 데이터 처리시간
Fig. 9 Data Processing Time by CPU Load in HSI

부하별로 비슷한 비율로 증가함을 확인할 수 있다. 마찬가지로 HSI를 사용하여 같은 크기의 데이터를 전송 시 CPU 사용률이 무부하에서 부하40%으로 증가할 때 시간지연은 평균 1.6배, 부하 40%에서 80%으로 증가할 때 시간지연이 평균 2배로 데이터 크기별로 비슷한 비율로 증가함을 볼 수 있다. 이를 통해 HSI가 일정한 성능 부하를 가짐을 확인할 수 있다. 그림 9는 HSI의 CPU 부하별 1회 평균 데이터 처리시간이다. HSI는 다양한 목표 시스템에 범용적으로 사용하기 위해 개발되었다. 따라서 HSI가 사용되는 해당 시스템의 하드웨어 사양, CPU 사용률, 데이터 크기, 최대 허용 데이터 전달 시간지연에 따라 HSI의 사용 조건이 달라질 수 있다. 이를 고려하여 그림 9의 그래프를 기준으로 HSI를 적용하고자 하는 목표 시스템에서 사용 가능한 센서 및 액추에이터의 개수를 구하는 수식을 구하면 다음과 같다.

$$\text{센서 및 액추에이터 개수} = \frac{\text{최대 허용 데이터 전달 시간지연}}{HSI\ \text{1회 데이터 처리시간}}$$

위 수식을 적용하는 예제를 찾아보기 위해 목표 시스템을 UAV로 가정하였다. 표 3은 UAV용 자동항법장치들의 데이터 갱신 주기를 나타낸다 [10-13]. 표 3을 참고하여 최대 허용 데이터 전달 시간지연으로 Kestrel의 데이터 갱신주기인 100Hz(10ms)을 사용하고, 그림 9를 참고하여 목표 시스템의 CPU 부하가 80%이고, 10KB의 데이터를 사용할 때의 HSI 1회 데이터 처리시간인 0.542ms를 사용하여 수식에 대입하면 18.4라는 값을 얻을 수 있다. 이를 정리하면 목표 시스템에서 CPU 부

표 3. 자동항법장치 데이터 갱신 주기
Table 3. Data Update Rate of Autopilots

자동항법장치	데이터 갱신 주기
Kestrel	100HZ (10ms)
MP 2028	50Hz (20ms)
Piccolo LT	25Hz (40ms)
Unav 3500	50Hz (20ms)
프레데터 훈련용 시뮬레이터	50Hz (20ms)
단비	50Hz (20ms)
X4-Flyer	45Hz (22ms)

하 80%일 때 HSI를 사용하여 10KB의 데이터를 18개 사용하여도 최대 허용 데이터 전달 시간지연을 넘지 않음을 말한다. 따라서 수식을 통해 계산한 결과를 이용하면 목표 시스템의 테스트를 위한 EcoHILS를 구축할 때, HSI의 데이터 처리 시간지연이 목표 시스템의 데이터 갱신 주기에 영향을 주지 않으면서 테스트 가능한 센서 및 액추에이터의 개수를 구할 수 있다.

V. 결 론

본 논문에서는 다양한 CPS 시험에 효율적으로 활용 가능한 EcoHILS를 소개하고, 동작 시험 및 정량적 성능 평가를 수행하였다. AR.Drone 제어 EcoHILS 테스트를 통해 EcoHILS가 다양한 시스템에 사용될 수 있는 실현 가능성을 확인하였고, 시간지연 측정 테스트를 통해 EcoHILS가 일정한 성능 부하를 가짐을 볼 수 있었다. 또한 성능 평가 결과로부터 목표 시스템에서 테스트 가능한 데이터양을 측정하는 방법을 제시하였다. 이를 통해 테스트 대상이 되는 목표 시스템에서, 신뢰성 있는 HILS가 수행 가능한 테스트 조건을 사전에 수립할 수 있다.

향후 연구에서는 개발하고자 하는 목표 시스템을 선정하여 EcoHILS를 구축하고, 동작 검증 및 성능 평가를 할 예정이다. 이를 통해 본 논문에서 제시한 EcoHILS의 신뢰성을 확보하고, 개선할 점들을 보완할 것이다.

References

[1] X. Cao, P. Cheng, J. Chen, Y. Sun, “An Online Optimization Approach for Control and Communication Codesign in Networked Cyber-Physical Systems,” IEEE Transactions on Industrial Informatics, Vol. 9, No. 1, pp. 439-450, 2013.

[2] S.J. Kang, I.G. Chun, J.M. Park, W.T. Kim, “Model-based Autonomic Computing Framework for Cyber-Physical Systems,” IEMEK J. Embed. Sys. Appl., Vol. 7, No. 5, pp. 267-275, 2012 (in Korean).

[3] M.J. Kim, S.J. Kang, I.G. Chun, W.T. Kim, “Design and Implementation of EcoHILS for Cyber-Physical Systems,” Proceedings of the Conference on IEMEK, pp. 42-44, 2013 (in Korean).

[4] R. Isermann, J. Schaffnit, S. Sinsel, “Hardware-in-the-loop simulation for the design and testing of engine-control systems,” Control Engineering Practice, Vol. 7, No. 5, pp. 643-653, 1999.

[5] C.S. Yoo, Y.S. Kang, B.J. Park, “Hardware-in-the-loop test for fault diagnosis system of tilt rotor UAV,” ICCAS, pp. 320-323, 2008.

[6] X. Li, D. Wang, Q. Wang, “Design and Realization of a Hardware-in-the-Loop Simulation System for Aerial Guided Bombs,” ISSCAA, pp. 1-5, 2008.

[7] T. Ersal, M. Brudnak, A. Salvi, J. L. Stein, Z. Filipi, H.K. Fathy, “Development and model-based transparency analysis of an Internet distributed hardware-in-the-loop simulation platform,” Mechatronics, Vol. 21, No. 1, pp. 22-29, 2011.

[8] I.G. Chun, J.M. Kim, H.Y. Lee, W.T. Kim, S.M. Park, E.S. Lee, “Faults and Adaptation Policy Modeling Method for Self-adaptive Robots,” Ubiquitous Computing and Multimedia Applications. Springer Berlin Heidelberg, pp. 156-164, 2011.

[9] S.J. Kang, M..J. Kim, J.M. Park, I.G. Chun, W.T. Kim, “LVC-Interoperation Development Framework for Acquiring High Reliable Cyber-Physical Weapon Systems,” J-KICS, Vol. 38C, No. 12, pp. 1228-1236, 2013 (in Korean).

[10] H. Chao, Y. Cao, Y. Chen, “Autopilots for small unmanned aerial vehicles: a survey,” International Journal of Control, Automation and Systems, Vol. 8, No. 1, pp. 36-44, 2010.

[11] J.T. Ball, K.A. Gluck, “Interfacing ACT-R 5.0 to an uninhabited air vehicle (UAV) synthetic task environment (STE),” Proceedings of the 2003 ACT-R workshop, pp. 87-98, 2003.
[12] M.H. Park, S.S. Kim, C.K. Yoo, K.Y. Choi, J.B. Park, “Development of Servo Type Angle-of-Attack Sensor for UAV,” Journal of KSAS, Vol. 37. No. 5, pp. 511-517, 2009 (in Korean).
[13] P. Pounds, R. Mahony, P. Corke, “Modelling and control of a quad-rotor robot,” Proceedings of Australasian Conference on Robotics and Automation, 2006.

저 자 소 개

김 민 조

2012년, 한국기술교육대학교 정보통신공학과 학사.
2014년, 과학기술연합대학원대학교 컴퓨터 소프트웨어과 석사.
현재, 한국전자통신연구원 연구원.
관심분야: CPS, M&S, HILS
Email: minjokim@etri.re.kr

강 성 주

2003년, 한양대학교 전자과 학사.
2005년, 한양대학교 전자과 석사.
2011년, 한국과학기술원, 카네기멜론대 소프트웨어 공학과 석사.
현재, 한국전자통신연구원 선임연구원.
관심분야: CPS, 자율제어, 소프트웨어 공학
Email: sjkang@etri.re.kr

전 인 걸

1996년, 성균관대학교 컴퓨터공학과 학사.
1998년, 성균관대학교 컴퓨터 공학과 석사.
2010년, 성균관대학교 컴퓨터공학과 박사.
현재, 한국전자통신연구원 책임연구원. 과학기술연합대학원대학교 조교수.
관심분야: CPS, M&S, 자율제어
Email: igchun@etri.re.kr

김 원 태

1994년, 한양대학교 전자과 학사.
1996년, 한양대학교 전자과 석사.
2000년, 한양대학교 전자과 박사.
현재, 한국전자통신연구원 책임연구원.
관심분야: CPS, M&S, 통신 미들웨어
Email: wtkim@etri.re.kr