Journal of Institute of Control, Robotics and Systems (2017) 23(2):117-125
https://doi.org/10.5302/J.ICROS.2017.17.0002 ISSN:1976-5622 eISSN:2233-4335

Open-Source Hardware를 이용한 도립진자 시스템에 대한 Hardware-In-The-Loop Simulator 구현

The Implementation of a Hardware-In-The-Loop Simulator for an Inverted Pendulum System Using Open-Source Hardware

최 진 석, 이 영 삼*
(Jin-Suk Choi[1] and Young-Sam Lee[1,*])
[1]Department of Electrical Engineering, Inha University

Abstract: In this paper, we propose an implementation method for a hardware-in-the-loop (HIL) simulator for an inverted pendulum system using low cost open-source hardware. We adopted the Arch Max (ARM Cortex-M4-based open-source hardware) as an HIL simulator platform. Using the nested vectored interrupt controller (NVIC) mechanism of the Cortex-M4 core, we assigned three interrupts with different priorities to handle three parts of the HIL simulator. The model equation of the inverted pendulum system was solved in real-time using the Runge-Kutta method in a timer interrupt service routine. An input capture interrupt-based method was used to simulate the motor drive by converting the duty ratio of pulse width modulation (PWM) signals into the voltage level applied to a DC motor. To simulate incremental rotary encoders, which are two sensors for the inverted pendulum system, a timer interrupt-based quadrature pulse generation algorithm was presented and to generate the quadrature pulse more accurately, an error compensation method was also provided. Finally, animation functionality to give a visual indication of how the controller is progressing was implemented through Vpython. Through experiments, we illustrated that the proposed HIL simulator successfully simulates the real inverted pendulum and is useful for teaching students about designing a working controller through programming embedded devices.

Keywords: open source hardware, ARM Cortex-M4, NVIC, HIL simulator, inverted pendulum

I. 서론

최근 공학교육인증(ABEEK) 기반의 교과과정에는 설계, 실험 실습, learn-by-doing 등에 중점을 두는 교과목들이 많이 포함되고 있다. 자동제어와 관련된 교과 과정에도 이러한 추세가 많이 반영되어 제어시스템 설계, 컴퓨터 제어, 임베디드 시스템 설계 등과 같은 교과목에서 실험 및 실습, 팀 프로젝트 등이 많이 도입되고 있다. 이러한 교육방법이 학생들에게 큰 효과를 가지기 위해서는 제어대상 시스템, 전원장치, 모터드라이브 등의 장비가 각 팀별로 지원될 수 있어야만 한다. 하지만 시설적 여건이 갖추어졌다고 할지라도 장비 조작과 마이크로콘트롤러의 사용에 익숙하지 않은 학생들이 실습 도중 장비를 손상시키거나 또는 불안정한 제어기를 구현하여 적용함으로써 시스템이 폭주하는 등의 문제를 유발할 수 있는 가능성이 높다. 따라서 장비의 손상이나 안전과 같은 문제를 원천적으로 차단하면서 목표로 한 교육효과를 달성할 수 있고 더불어 경제적인 비용으로 구축 가능한 실험 및 실습 장비가 있다면 강사나 학생 모두에게 큰 도움이 될 것이다. 본 논문의 연구는 이러한 문제점을 효과적으로 해결할 수 있는 방법을 찾기 위한 노력의 일환으로 시작되었다.

HILS (Hardware-In-the-Loop simulation)는 임베디드 제어 시스템을 효율적으로 검증하기 위해 사용되는 테스트 방식이다[1]. 안전, 사용가능성, 비용 등의 요소를 고려했을 때 실제 플랜트를 대상으로 임베디드 제어 시스템을 테스트하는 것이 실용적이지 않을 수 있다. HILS를 이용하여 대상 시스템을 시뮬레이션하게 되면 실제 테스트 이전에 가상의 환경에서 임베디드 제어 시스템을 철저하게 검증해 볼 수 있는 장점이 있다. 이런 이유로 자동차, 철도, 항공우주, 의료기기, 산업기계, 전력 생성 시스템 등과 같은 다양한 분야에서 HILS를 이용한 제어 시스템 검증을 수행하고 있다[2-6]. HILS에서는 실시간 시뮬레이션을 수행하는 컴퓨터를 가상의 시스템으로 사용하고 제어기는 실제의 제어기를 사용하게 된다. 자동제어 및 임베디드 시스템과 관련된 교육에 HIL simulator를 사용할 수 있다면 이를 가상의 대상 시스템으로 하여 마이크로콘트롤러에 제어 시스템을 구현하는 학습을 장비의 고장이나 안전에 대한 염려 없이 수행할 있고 실제 시스템을 대상으로 한 교육과 거의 대등한 수준의 학습 효과도 얻을 수 있을 것이다. 하지만 팀 단위로 이루어지는 학생들의 실험, 실습 교육에 각각의 팀마다 HIL

* Corresponding Author
Manuscript received January 2, 2017 / revised January 11, 2017 / accepted January 11, 2017
최진석, 이영삼: 인하대학교 전기공학과
(jeadka@naver.com/lys@inha.ac.kr)
※ 본 논문은 한국전력공사의 재원으로 기초전력연구원의 2015년 선정 기초연구개발과제의 지원을 받아 수행된 것임(과제번호 : R15XA03-12).

simulator를 제공하는 것은 비용적인 면에서 많은 어려움이 따른다. 기존에 산업계에서 사용되는 HIL simulator는 비용보다는 성능에 초점을 맞추고 있고 교육용으로 사용할 수 있는 HIL simulator 역시 팀별로 제공할 수 있을 정도로 저렴한 가격은 아니기 때문이다.

본 논문에서는 자동제어 시스템을 교육함에 있어 가장 교육효과가 높은 시스템중 하나인 도립진자 시스템에 대한 HIL simulator를 저가형 open-source hardware를 이용하여 구현하는 방법을 제안함으로써 위에서 언급한 고장, 안전, 비용 등의 문제점을 해결하면서 동시에 학생들에게 마이크로콘트롤러를 이용하여 실제로 동작하는 제어기를 구현하기 위해 알아야 하는 I/O, timer interrupt, 엔코더 디코딩, PWM 발생 등과 같은 방법을 효과적으로 교육할 수 있는 장비를 구축하고자 한다.

Open-source hardware란 각종 하드웨어 제작에 필요한 회로도 및 관련 설명서, 인쇄회로기판, 개발 환경 등을 공개함으로써 누구나 이와 동일하게 혹은 이를 활용한 제품을 개발할 수 있도록 지원하는 하드웨어를 의미한다. Open-source hardware의 확산은 전문 엔지니어나 일반인들의 하드웨어 제작 대중화를 견인하는 동시에 대기업 및 중소기업의 제품과 서비스 관련 연구 개발활동을 촉진함으로써 이른바 제3의 산업혁명을 일으키는 주된 동력으로 주목받고 있다. 2005년 이탈리아에서 탄생한 Arduino는 가장 유명하고 널리 활용되는 open-source hardware platform이다[7]. 이후 Raspberry Pi, Beaglebone과 같이 Linux 운영체제를 탑재한 platform 들이 출시되어 널리 사용되고 있으며 최근에는 Mbed 진영에서 출시하고 있는 ARM Cortex-M 계열 마이크로콘트롤러에 기반을 둔 platform 역시 활발하게 사용되고 있다[8,9]. MBED 진영의 제품들은 웹기반의 컴파일 방식과 drag-and-drop 방식의 프로그램 다운로드를 특징으로 하고 있다. Arduino 진영의 platform 들은 주로 Atmel 사의 8-bit 마이크로콘트롤러를 사용하고 있어 기능이 제한적인데 비해 Mbed 진영의 platform은 부동소수점 연산도 지원하는 Cortex-M4 계열의 마이크로콘트롤러를 채택한 platform도 포함하고 있어 폭넓은 용용 가능성을 가지고 있다. HIL simulator를 구현하기 위해서는 다양한 I/O 기능을 모사할 수 있어야 하고 고속의 실시간 연산이 가능해야 하므로 저속의 Arduino나 실시간성을 보장할 수 없는 Raspberry Pi 혹은 Beaglebone 등은 적합하지 않다. 본 논문에서는 부동소수점 연산이 가능하고 풍부한 I/O를 가지고 있으며 실시간 성능을 보장할 수 있는 Mbed 진영의 Arch Max라는 Cortex-M4 기반의 open-source hardware platform을 기반으로 도립진자 시스템에 대한 HIL simulator를 구현하고자 한다.

본 논문의 구성은 다음과 같다. II장에서는 카트형 도립진자의 모델 방정식에 대해서 기술하고 III장에서는 HIL simulator의 구성과 구현에 대해 기술한다. 세부적으로는 HIL simulator의 구성요소인 모터드라이브의 모사, 도립진자 시스템의 미분 방정식 해법, 회전형 엔코더의 quadrature pulse 생성을 구현하는 방법을 제시하고 시각적 정보를 통해 가상 시스템의 상태를 확인해 볼 수 있는 애니메이션 기능을 소개한다. IV장에서는 도립진자 시스템에 대한 HIL simulator를 이용한 실험과 실제 도립진자를 이용한 실험을 비교한다. 마지막으로 V장에서 결론을 맺는다.

II. 도립진자 시스템의 모델방정식

도립진자 시스템은 자동제어가 무엇인지를 학생들에게 설명할 때 가장 자주 사용되는 시스템 중의 하나이다. 불안정 시스템이고 비선형 동특성을 가진다는 점, 그리고 비최소위상(non-minimum phase) 특징을 가지고 있다는 점 등으로 인해 지금까지 도립진자 제어와 관련된 수많은 연구가 발표되었다. 최근에는 1단(single)뿐 아니라 2단(double) 및 3단(triple)으로 구성된 도립진자의 제어에 대한 연구도 발표되어 큰 주목을 받았다[10,11]. 본 논문에서는 그중 가장 대표적이고 학부생 및 대학원생들의 자동제어 교육에 효과적으로 활용될 수 있는 1단 도립진자를 대상으로 HIL simulator를 구성하는 방법을 제안하고자 한다. 도립진자 시스템의 개념도는 그림 1과 같이 나타낼 수 있다.

여기서 F는 카트(cart)에 인가되는 힘, M은 카트의 질량, m은 진자의 질량, I_p는 진자의 무게중심에 대한 회전 관성 모멘트, θ는 지면과 수직을 이루는 선으로부터 진자 막대가 가지는 회전 변위, x는 레일의 중심으로부터 카트 중심까지의 변위, l은 진자의 회전축에서 진자의 무게중심까지의 거리, b는 카트와 레일의 마찰계수, $x_{\lim}$은 레일의 중심으로부터 양 끝의 벽까지의 거리를 나타낸다. Lagrange formulation을 이용하여 도립진자의 미분방정식을 구해보면 다음과 같이 나타낼 수 있다[12].

$$(I_p+ml^2)\ddot{\theta}+ml(\cos\theta)\ddot{x}-mgl\sin\theta=0 \quad (1)$$

$$(M+m)\ddot{x}-ml(\sin\theta)\dot{\theta}^2+ml(\cos\theta)\ddot{\theta}+b\dot{x}=F \quad (2)$$

여기서 $g=9.8m/s^2$는 중력가속도이다. 힘 F는 모터를 통해 생성하며 랙기어(rack gear), 볼 스크루(ball screw), 타이밍벨트와 같은 동력전달방법을 통해 카트에 전달된다.

HIL simulator를 이용하여 임베디드 제어기의 동작을 검증한 후 해당 제어기를 실제 시스템에 적용했을 때에도 그대로 동작하는 것을 확인해 볼 수 있다면 학생들의 학습효과가 더욱 배가될 것이다. 이런 이유로 본 논문에서는 연구실에서 직접 제작하여 보유하고 있는 도립진자 시스템을

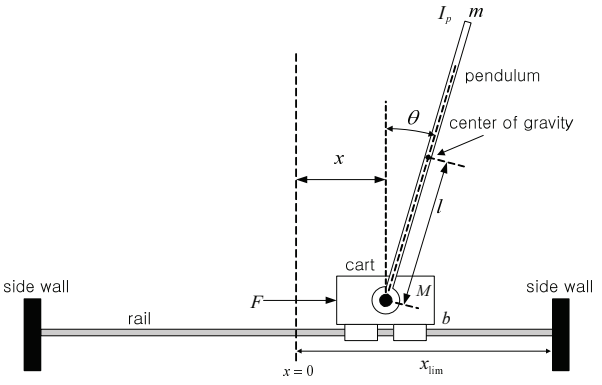

그림 1. 도립진자 시스템의 개념도.

Fig. 1. Conceptual diagram of an inverted pendulum system.

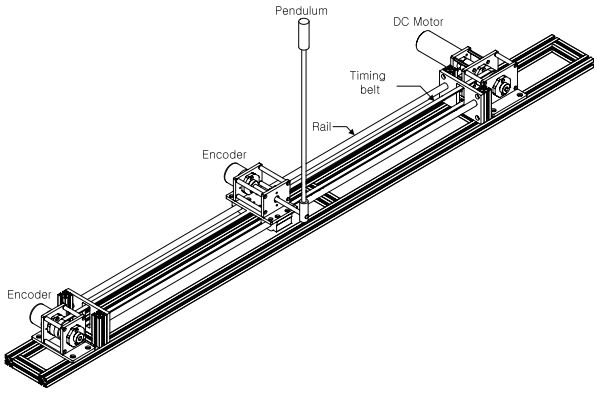

그림 2. 연구실에서 제작된 도립진자 시스템의 3차원 구성도.
Fig. 2. 3D diagram of a lab-built inverted pendulum system.

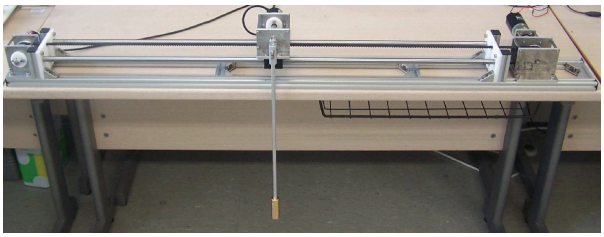

그림 3. 연구실에서 제작된 도립진자 시스템의 실제 사진.
Fig. 3. Picture of a lab-built inverted pendulum system.

대상으로 HIL simulator를 구성하였다. 그림 2는 연구실에서 직접 제작한 도립진자 시스템의 3차원 구성도를 나타내고 있다. 구동기로는 DC 모터를 사용하고 있고 카트로의 힘 전달은 타이밍벨트를 이용하는 구조이다. 회전변위 θ와 직선변위 x의 측정은 2개의 회전형 엔코더를 이용한다. 사용된 엔코더의 해상도는 4000 PPR이다. 그림 3은 제작된 도립진자 시스템의 실제 사진을 나타내고 있다.

[12]의 방법을 따라 DC 모터의 모델방정식을 식 (2)에 포함시킨 후 정리하면 도립진자의 비선형 모델방정식은 다음과 같은 상태방정식으로 나타낼 수 있다.

$$\begin{aligned}
\dot{x}_1 &= x_3, \\
\dot{x}_2 &= x_4, \\
\dot{x}_3 &= \frac{\Lambda_1 - (I_p + ml^2)(\Lambda_2 + \Lambda_3 V)}{\Lambda_5}, \\
\dot{x}_4 &= \frac{\Lambda_4 + ml\cos(x_2)(\Lambda_2 + \Lambda_3 V)}{\Lambda_5}.
\end{aligned} \tag{3}$$

여기서 상태변수는 $x_1 = x$, $x_2 = \theta$, $x_3 = \dot{x}$, $x_4 = \dot{\theta}$이며 $\Lambda_1 \sim \Lambda_5$는 다음과 같은 의미를 가진다.

$$\begin{aligned}
\Lambda_1 &= m^2 l^2 g \sin(x_2)\cos(x_2), \\
\Lambda_2 &= -\left(b + \frac{K_m K_b}{R_m r^2}\right) x_3 + ml\sin(x_2)x_4^2, \\
\Lambda_3 &= \frac{K_m}{R_m r} \\
\Lambda_4 &= -(M+m)mgl\sin(x_2), \\
\Lambda_5 &= [(ml\cos(x_2)]^2 - (I_p + ml^2)(M+m).
\end{aligned}$$

K_m, K_b, R_m, V는 각각 DC 모터의 토크상수, 역기전력 상수, 권선저항, 단자전압을 나타내며 r은 타이밍풀리의 반경을 나타낸다. 표 1은 제작된 도립진자 시스템의 모델 계수를 나타내고 있다.

표 1. 연구실 제작 도립진자 시스템의 모델계수.
Table 1. Model parameters of the lab-built inverted pendulum.

Parameters	Values
K_m	0.257 [Nm/A]
K_b	0.257 [V·rad/sec]
M	0.712 [Kg]
m	0.209 [Kg]
r	0.0194 [m]
l	0.326 [m]
I_p	0.00974 [Kg·m²]
R_m	2.32 [Ω]
b	0 [N·sec/m]

III. 도립진자 HIL simulator 구성과 구현

1. 도립진자 HIL simulator의 구성

도립진자 HIL simulator는 실제 도립진자 시스템의 동역학적 특성과 입출력을 실시간으로 정확히 모사할 수 있어야 한다. 도립진자 제어 시스템의 실제 구성은 그림 4와 같다.

임베디드 제어기의 관점에서 보았을 때 도립진자 시스템은 DC 모터 드라이브에 전달되는 PWM을 입력으로 가지며 2개의 회전형 엔코더에서 발생되는 quadrature pulse를 출력으로 갖는 시스템으로 볼 수 있다. 따라서 도립진자에 대한 HIL simulator는 그림 4에서 점선으로 표현된 실제의 도립진자 시스템을 대체할 수 있어야 한다. 이를 위해 도립진자 HIL simulator는 입력에 해당하는 PWM을 처리하여 모터의 단자전압으로 변환해주는 모터 드라이브 모사 부분, 도립진자의 동특성에 해당하는 비선형 미분 방정식 (3)을 실시간으로 풀이하는 부분, 출력인 x와 θ의 변화를 정확히 반영하여 quadrature pulse를 생성해 내는 회전형 엔코더 모사 부분으로 구성된다. 실제 시스템에서

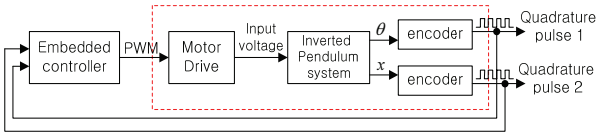

그림 4. 실제 도립진자 제어시스템의 구성.
Fig. 4. Actual control system for an inverted pendulum.

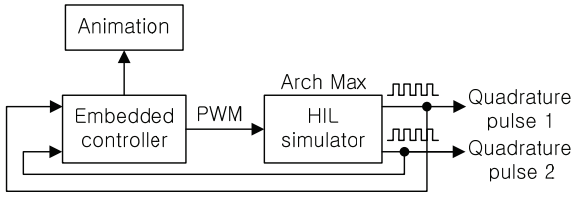

그림 5. HIL simulator를 이용한 도립진자 제어시스템의 구성.
Fig. 5. Control system for an inverted pendulum using a HIL simulator.

는 제어가 수행되는 양상을 눈으로 확인할 수 있지만 HIL simulator는 가상의 플랜트이기 때문에 이를 보완하기 위해 제어가 어떻게 수행되는지 확인할 수 있는 시각적 기능 역시 하나의 구성요소로 포함되어야 한다. 따라서 제안된 HIL simulator를 이용한 제어 시스템 검증은 그림 5와 같이 수행할 수 있다.

도립진자의 HIL simulator 구현을 위한 연산 platform으로는 Arch Max를 사용하기로 한다. Arch Max는 Mbed 진영에서 출시한 hardware platform으로 STM32F407 마이크로콘트롤러를 기반으로 하고 있다. 그림 6은 Arch Max의 사진과 Pin map을 나타내고 있고 있다. Pin map을 통해서도 볼 수 있듯이 다양한 주변장치를 탑재하고 있어 HIL simulator 구현에 필요한 I/O의 모사에 적합하다. STM32F407은 ARM Cortex-M4 core를 채택한 ST Microelectronics 사의 마이크로콘트롤러로써 168MHz의 속도로 동작하며 하드웨어 부동소수점 연산을 지원하기 때문에 도립진자의 비선형 미분방정식을 실시간으로 풀이하기에 적합하다. 또한 NVIC (Nested Vectored Interrupt Controller)라는 기능을 가지고 있어 interrupt의 중첩이 가능하고 interrupt의 우선순위를 지정할 수 있는 장점이 있다. 도립진자 HIL simulator 구현을 위해서는 interrupt의 우선순위를 지정할 수 있어야 하는데 NVIC 기능은 이에 부합한다고 할 수 있다.

HIL simulator를 구성하는 구성요소 중 모터드라이브 부분, 비선형 미분방정식의 풀이 부분, 엔코더의 quadrature pulse 생성 부분을 각각 별도의 interrupt를 이용하여 구현하였다. 표 2는 각각의 모사 기능을 구현하기 위해 사용된 interrupt의 종류, 주기, 그리고 우선순위를 나타내고 있다.

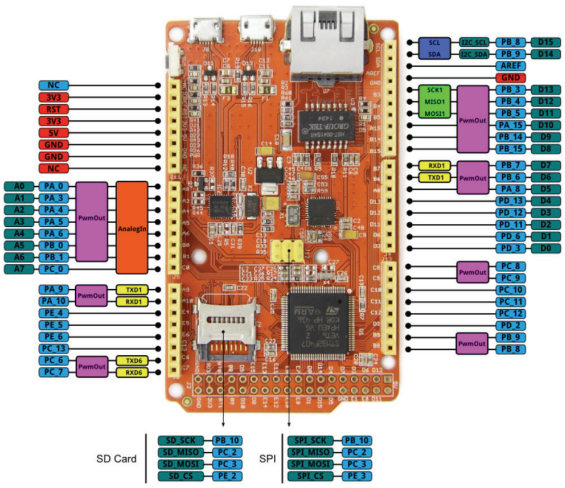

그림 6. Cortex-M4 기반의 Arch Max와 pin map.
Fig. 6. Arch Max based on Cortex-M4 and its pin map.

표 2. 기능, interrupt 종류, 주기, 우선순위.
Table 2. Function, the kind of interrupts, period, and priority.

기능	interrupt 종류	주기	우선순위
모터드라이브 모사	input capture interrupt	×	2
도립진자 미분방정식 풀이	timer interrupt	1 ms	3
회전형 엔코더 모사	timer interrupt	5 us	1

interrupt의 우선순위는 Cortex-M4 core에서 지원되는 NVIC를 이용하여 배정하였다. interrupt의 주기는 timer interrupt인 경우에만 명시하였고 event에 의해 촉발되는 input capture interrupt는 주기를 명시하지 않았다. 저가의 open-source hardware가 가지는 제한된 연산성능만으로 실용적 HIL simulator를 구성하고자 하는 것이 본 논문의 목표이므로 매우 작은 샘플타임을 사용해 빠른 연산성능이 필요한 dead time, 스위칭 리플 효과 등과 같은 세부적인 모사는 하지 않는 것으로 가정한다.

2. 모터 드라이브 모사부의 구현

이 절에서는 도립진자의 HIL simulator가 모터 드라이브를 모사하는 부분, 즉 PWM 입력 신호를 DC 모터에 인가되는 전압 V로 변환해 주는 부분의 구현에 대해 기술한다. DC 모터를 구동하는 드라이브 보드는 대부분 그림 7과 같은 H-bridge 회로를 기반으로 하고 있다. 여기서 Q_1, Q_2, Q_3, Q_4는 FET로써 각각의 FET를 적절하게 On/Off 시킴으로써 DC 모터를 정/역 구동하게 된다. PWM 신호는 FET 소자에 대한 On/Off 명령으로 사용되며 independent bipolar, independent unipolar, complementary bipolar, complementary unipolar 등의 방식을 가진다[13]. 본 논문에서는 이 4가지 방식 중에서 가장 토크 리플(torque ripple)을 작게 발생한다고 알려진 unipolar complementary PWM 방식을 가정한다.

Complementary unipolar PWM 방식은 모터를 제어하기 위해 Q_1과 Q_2, 그리고 Q_3와 Q_4를 서로 상보적으로 스위칭하는 방식으로 그림 8은 Complementary unipolar PWM 방식에서의 PWM 파형의 예를 보여주고 있다. 여기서 a와 b는 각각 Q_1과 Q_3에 인가되는 PWM의 듀티비(duty ratio)이며 a와 b를 조정하여 모터에 인가되는 전압을 모듈레이션 할 수 있다. H-bridge에 사용된 DC 전압의 크기를 V_{DC}라고 하고 모터에 인가하고자 하는 전압을 V라고 하면 a와 b는 각각 다음과 같이 결정된다.

$$\eta = \frac{V}{V_{DC}} \times 100, \quad a = \frac{100+\eta}{2}, \quad b = \frac{100-\eta}{2}.$$

따라서 모터에 인가되는 전압 V는 듀티비 a와 b로부터 다음과 같이 구해진다.

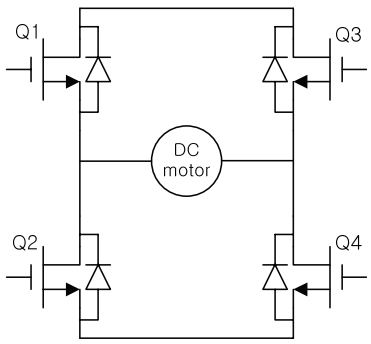

그림 7. DC 모터 구동을 위한 H-bridge 회로.
Fig. 7. An H-bridge circuit for DC motor drive.

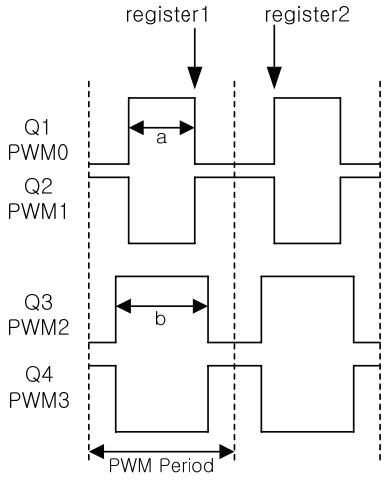

그림 8. Unipolar complementary PWM 개념도.

Fig. 8. Diagram of the unipolar complementary PWM.

$$V = \frac{(a-b)}{100} \times V_{DC} \tag{4}$$

따라서 PWM 신호로부터 모터에 인가되는 전압 V를 구하기 위해서는 PWM 신호의 듀티비 a와 b를 측정할 수 있어야 한다. DC 모터의 구동에 사용되는 PWM 신호는 스위칭 시 발생하는 소음을 최소화하기 위해 보통 15~20KHz 정도의 주파수를 갖도록 설정한다. 이렇듯 고속으로 스위칭되는 PWM 신호로부터 듀티비 a와 b를 정확히 알아내기 위해 STM32F407의 Input capture interrupt를 사용하도록 한다. input capture 기능을 이용하여 PWM 신호의 rising edge 순간에 counter의 값을 읽어 register 1에 저장함과 동시에 counter 값을 0으로 초기화하고 falling edge인 순간에는 counter 값을 register 2에 저장하도록 설정할 수 있다. PWM 신호의 rising edge가 input capture interrupt를 촉발(trigger)하고 해당 ISR(interrupt service routine)에서는 register1을 이용해 듀티비를, register2를 이용해 PWM 신호의 period를 측정할 수 있다.

듀티비 측정을 위해 external interrupt를 사용해 볼 수도 있다. 이 경우에는 PWM 신호의 rising edge와 falling edge의 순간에 ISR (Interrupt Service Routine)로 진입하고 ISR에서는 counter의 값을 읽어 들여 듀티비를 계산하게 된다. 하지만 external interrupt를 사용하면 interrupt의 중첩이 발생할 경우 우선순위가 앞서는 interrupt가 external interrupt의 동작을 일시적으로 중지시킬 수 있고 그 만큼 counter를 읽는 동작이 중지될 수 있기 때문에 정확한 듀티비 측정이 이루어지지 않게 된다. 이에 비해 input capture interrupt의 경우 interrupt의 중첩이 발생해도 PWM 신호의 rising edge 및 falling edge 순간의 counter 값을 시간지연 없이 하드웨어적인 방법으로 레지스터에 저장해 놓기 때문에 우선순위가 높은 interrupt에 의해 input capture interrupt가 일시적으로 정지해도 정확한 듀티비 측정이 가능하다. 이런 이유로 모터드라이브를 모사하기 위한 input capture interrupt의 우선순위는 표 2에서 정리한 바와 같이 최고 우선순위를 갖지 않아도 된다.

표 3. Sine 함수 계산에 소요되는 연산 시간 비교 [15].

Table 3. Comparison of time for computing sine function.

MCU 계열	MCU 모델	연산시간
AVR	ATmega128	162 us
Cortex-M3	STM32F103	42.5 us
Cortex-M4	STM32F407	0.68us

3. 도립진자 시스템 모사부의 구현

모터드라이브 모사부를 통해 모터의 단자전압 V가 구해지면 이 값을 이용해 식 (3)의 비선형 상태방정식을 풀이함으로써 도립진자의 상태변수를 갱신할 수 있다. 이 논문에서는 1ms의 주기를 갖는 timer interrupt를 이용하여 식 (3)의 비선형 미분방정식을 풀이하였으며 미분 방정식의 풀이를 위해 Runge-Kutta 4차 미분방정식 해법을 이용하였다 [14]. 미분 방정식 풀이를 위한 timer interrupt는 표 2에서 기술된 3개의 interrupt 중에서 가장 우선순위를 낮게 선정하였다. 모터드라이브 모사와 회전형 엔코더 모사를 위해 사용된 interrupt에 비해 상대적으로 느리게 호출되기 때문이다. 식 (3)의 비선형 미분방정식에는 비교적 시간이 많이 소요되는 sine과 cosine의 계산을 포함하고 있는 것을 볼 수 있다. 표 3은 sine 함수를 계산하는데 소요되는 연산시간을 MCU 별로 비교한 자료이다. Cortex-M4 core를 기반으로 하는 STM32F407의 경우 부동소수점 연산을 하드웨어적으로 지원하고 sine 함수 계산을 1 us 이내에 수행할 수 있기 때문에 Runge-Kutta 방법처럼 해의 정확도를 높이기 위해 여러 단계를 거쳐 미분 방정식을 풀이하는 방법을 사용해도 1 ms 이내에 계산을 완료하는데 문제가 없음을 알 수 있다.

4. 회전형 엔코더 모사부의 구현

미분 방정식 풀이를 통해 4개의 상태변수에 대한 갱신이 완료되면 회전형 엔코더 모사부에서는 상태변수 값을 반영하여 quadrature pulse를 생성한다. 회전형 엔코더의 quadrature pulse는 그림 9와 같이 A상과 B상으로 이루어진다. 회전형 엔코더의 최소 해상도에 해당하는 각 α만큼 각변위가 발생할 때 마다 그림 9와 같이 A상과 B상의 출력값이 변화한다. A상과 B상의 출력을 논리값 0과 1로 나타낼 수 있는데 이때 발생 가능한 논리값의 조합은 모두 4가지이다. 각각의 조합에 따라 그림 9와 같이 state number를 할당할 수 있다.

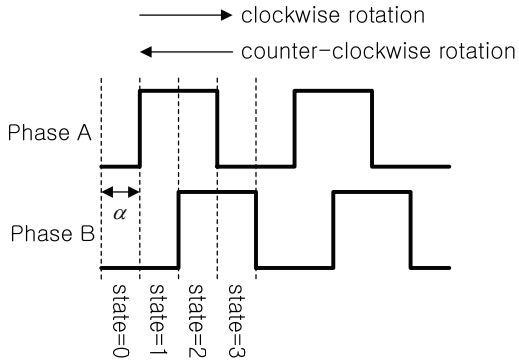

그림 9. Quadrature pulse의 파형과 state number.

Fig. 9. Wave form of the quadrature pulses and state number.

II장에서 기술한 도립진자 시스템은 4000 PPR의 해상도를 갖는 회전형 엔코더를 사용하고 있으므로 HIL simulator에서도 같은 해상도의 엔코더를 모사하도록 구현하였다. 엔코더 모사의 핵심은 도립진자의 상태변수 값을 반영하여 quadrature pulse를 생성하는 것으로 timer interrupt를 이용하여 구현하였다. 일정한 주기로 timer interrupt가 촉발될 때 ISR에서는 도립진자의 상태값을 반영하여 A상과 B상의 출력값 변화를 결정하게 된다. 본 논문에서는 엔코더가 3000 RPM의 속도로 회전할 때까지 quadrature pulse를 생성할 수 있게끔 timer interrupt의 주기를 선정하였다. 3000 RPM의 속도로 회전할 경우 4000 PPR의 해상도를 갖는 엔코더의 경우 1초 동안에 최대로

$$\left(\frac{3000}{60}\right)\times 4000 = 200{,}000$$

의 회수만큼 상변화가 발생할 수 있으므로 이를 모사하기 위하여 timer interrupt의 주기를 $1/200{,}000 = 5$[us]로 가정하였다. quadrature pulse의 생성을 담당하는 timer interrupt는 표 2에서 비교된 interrupt 중에서 주기가 가장 짧으므로 NVIC를 이용해 최상위의 우선순위를 갖도록 설정하였다.

정확하게 quadrature pulse를 발생하기 위해서는 5 us의 주기마다 도립진자의 상태값 변화를 검사해 A상과 B상의 변화를 결정해야 한다. 이를 위해서는 5 us 주기마다 도립진자 시스템의 미분방정식을 풀이해야 하는데 sine과 cosine 함수를 포함하고 있는 도립진자의 미분방정식을 5 us 마다 풀이하는 것은 STM32F407의 연산능력으로는 무리이다. 본 논문에서는 상태변수 중에서 속도에 해당하는 $x_3(=\dot{x})$와 $x_4(=\dot{\theta})$를 이용해 quadrature pulse를 생성하는 방식을 제안한다. 미분방정식을 풀이하는 timer interrupt가 1회 촉발될 때 엔코더를 모사하기 위한 timer interrupt는 200회 촉발되며 그동안 속도 성분인 x_3와 x_4은 불변한다. 기본적인 착상은 미분방정식이 다시 풀리기까지 x_3와 x_4가 불변하므로 회전형 엔코더가 x_3 및 x_4에 비례하는 속도로 등속회전한다고 가정하고 이로 인해 A, B상의 출력을 변경해야 할 만큼의 각변화가 발생했는지를 검사한 후 조건이 만족하면 A, B상의 출력을 변경하는 것이다.

편의를 위하여 카트의 변위를 측정하는 엔코더를 대상으로 설명하기로 한다. $\dot{\psi}$를 카트의 변위를 측정하기 위한 엔코더의 회전속도라고 가정하면 $\dot{\psi} = K\dot{x}$이 성립한다. 사용된 타이밍풀리의 반경이 0.0194 [m] 이므로 $K = \dfrac{1}{0.0194\times 2\pi}$ 이다.

ϕ를 모듈로(modulo) 연산을 통해 $0 \le \phi \le \alpha$가 되도록 정규화하여 사용하는 엔코더의 각변위라고 가정하자. 엔코더의 A상과 B상의 출력을 변경해야 하는 경우는 그림 10과 같이 Δ만큼의 시간(여기서는 5 us)이 흐른 후의 엔코더의 각변위 $\phi + \Delta\dot{\psi}$가 α보다 크거나 0보다 작아 모듈로 연산을 통해 다시 정규화가 필요한 경우이다. 알고리즘 1은 이러한 착상을 기반으로 엔코더의 quadrature pulse를 생성하는 알고리즘을 나타낸 것이다.

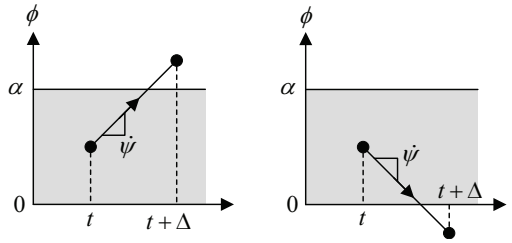

그림 10. 경계선을 교차하는 경우 (왼쪽: 시계방향회전, 오른쪽: 반시계방향 회전).

Fig. 10. Case of crossing borders (left: CW, right: CCW).

알고리즘 1. Quadrature pulse 생성 알고리즘.

Algorithm 1. Quadrature pulse generation algorithm.

```
1:  ψ̇ = K ẋ; // encoder angular velocity
2:  φ = φ + ψ̇ Δ; // rotation update
3:  if(ψ̇ ≥ 0){      // clockwise rotation
4:    if(φ > α){      // border crossing
5:      φ = φ − α;   // normalization
6:      N = N + 1;   // counter increment
7:      state = (state+1)%4; // state update
8:    }
9:  }
10: else{  // counter-clockwise rotation
11:   if(φ < 0){      // border crossing
12:     φ = φ + α;   // normalization
13:     N = N − 1;   // counter decrement
14:     state = (state-1)%4; // state update
15:   }
16: }
17: // Generation of  A and B using the state number
18: if(state==0) A=0, B=0;
19: else if(state==1) A=1, B=0;
20: else if(state==2) A=1, B=1;
21: else if(state==3) A=0, B=1;
```

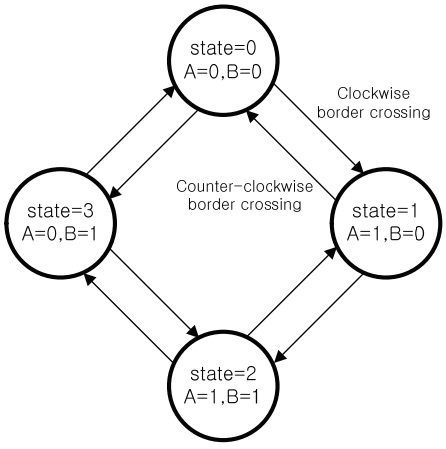

그림 11. Quadrature pulse 생성을 위한 상태천이도.

Fig. 11. State transition diagram for quadrature pulse generation.

알고리즘 1에서는 정규화된 각변위 ϕ가 경계선을 교차하는 방식에 따라 그림 11과 같은 상태천이도에 사용되는 상태값을 변경하며 천이도에서의 상태값에 따라 A상과 B상의 출력값을 변경하게 된다. 알고리즘에서 N은 counter로써 엔코더 최소해상도에 해당하는 각변위인 α를 단위로 했을 때 몇 단위만큼 엔코더가 회전하였는지를 나타낸다.

알고리즘 1을 통해 생성된 quadrature pulse를 임베디드 제어기의 디코딩 로직을 이용하여 엔코더의 각변위를 측정하게 되면 $\overline{\psi}=N\alpha$에 해당하는 각변위를 측정하게 된다. 하지만 $\overline{\psi}$는 미분방정식이 한번 갱신되고 또 다시 갱신될 때까지의 시간인 1ms 동안 엔코더의 회전속도가 불변한다는 가정을 이용하여 나타낸 각변위이다. 1ms가 지난 후 미분방정식을 새로이 풀었을 때 얻어진 상태변수(카트의 위치와 진자의 각도)를 엔코더의 각변위로 나타낸 값을 ψ라고 하면 $\psi \neq \overline{\psi}$이 될 가능성이 높고 극단적인 경우에는 두 값의 차이가 $e=\psi-\overline{\psi}$가 발산할 수도 있다. 따라서 이 논문에서는 그림 12와 같이 보상기를 도입해 오차 e를 보정하는 방법을 제안한다.

그림 12와 같이 보상기를 통해 엔코더 속도를 보정한 후 pulse 생성 알고리즘에 적용한 경우와 보정을 하지 않고 그대로 pulse 생성 알고리즘에 적용하여 quadrature pulse를 생성한 경우의 pulse 생성 정확도를 비교하기 위하여 DC 모터에 대한 HIL simulator를 구성하고 10 volt를 인가하여 한쪽 방향으로만 회전하게 하는 경우를 모사하여 오차 $e=\psi-\overline{\psi}$를 비교해 보았다. 보상기의 형태는 P-type 보상기를 사용하였다. 그림 13은 두 경우의 오차를 서로 비교하여 도시하고 있다. 보상기를 적용하지 않은 경우 오차가 계속해서 커져서 발산하는 반면 보상기를 적용한 경우 오차가 작은 범위 내에서 유지되는 것을 볼 수 있다. 이는 제안된 quadrature pulse 생성 방법이 효과적으로 회전형 엔코더를 모사하고 있음을 뒷받침해 준다. 보상기를 적용하지 않고 단순히 속도정보인 ψ만을 이용하여 quadrature pulse를 생성할 경우 임베디드 제어부에서 시스템의 상태변수 중 각변위 정보를 정확히 얻지 못하기 때문에 제어시스템이 발산하는 경우도 발생할 수 있는데 제안된 보상기를 적용할 경우 이러한 문제를 해결할 수 있다.

5. 도립진자 에니메이션 기능

HIL simulator는 실제 시스템을 모사하는 가상의 플랜트이므로 제어가 어떻게 수행되는지를 확인할 수 있는 시각적인 기능의 제공이 필수적이다. 이를 위해 VPython을 이용

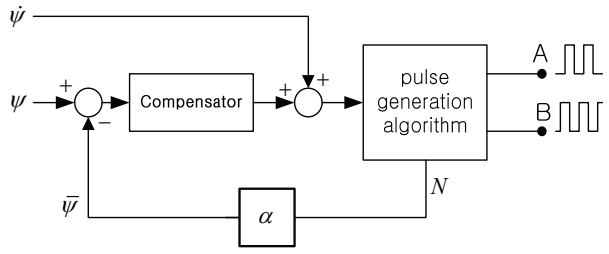

그림 12. 보상기를 이용한 오차의 보정.

Fig. 12. Error correction using a compensator.

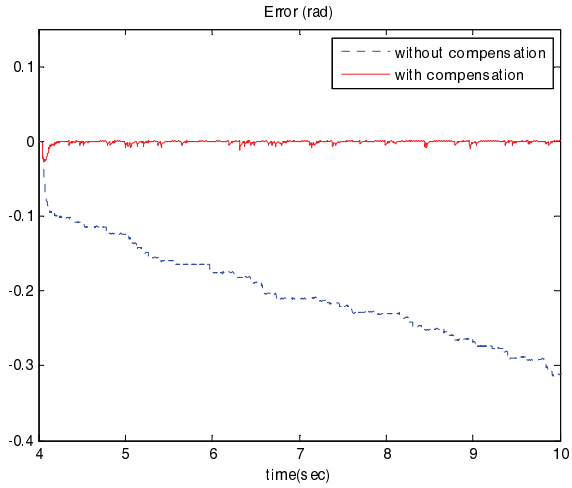

그림 13. 오차 비교.

Fig. 13. Comparison of error ($e=\psi-\overline{\psi}$).

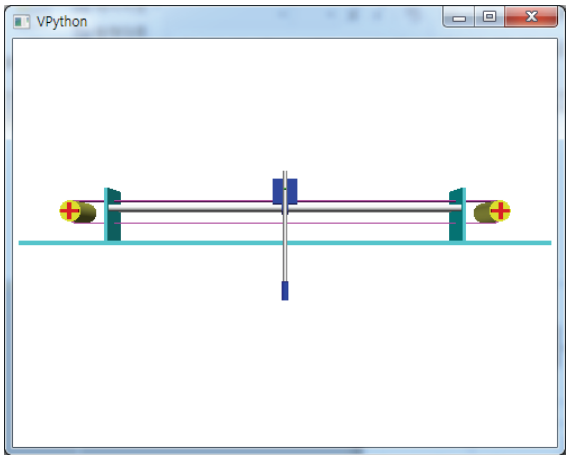

그림 14. VPython으로 구현한 도립진자 시스템의 애니메이션.

Fig. 14. VPython-based animation of the inverted pendulum.

하여 애니메이션 프로그램을 제작하였다. VPython은 3D 물체와 그래프를 나타내기 위한 프로그래밍 툴이여서 이를 통해 만들어진 애니메이션은 여러 각도에서 시스템의 상태를 보여줄 수 있다. 그림 14는 VPython을 이용하여 제작한 도립진자 시스템의 애니메이션 프로그램이다. 이렇게 구현된 프로그램은 도립진자 시스템의 진자 막대와 카트의 위치에 대한 시각적인 정보를 학생들에게 제공함으로써 학생들에게 더욱 큰 학습효과를 제공해 줄 수 있다.

IV. 도립진자 HIL simualtor 실험 결과

도립진자 시스템에 대해 [12]에서 제안된 스윙업 제어알고리즘을 Cortex-M3기반의 open-source hardware인 Arduino Due를 이용하여 구현한 후 본 논문에서 제안한 HIL simulator에 적용해 보았다. [12]에서 제안된 제어알고리즘은 레일의 길이에 제약을 가진 도립진자 시스템에 적용할 수 있는 에너지 기반의 스윙업 제어 알고리즘이다. 그림 15는 제안된 HIL simulator를 이용한 제어실험 수행시 진자의 각변위 θ를 도시한 것으로 제어기가 진자를 점진적으로 흔들

어 최종적으로 스윙업이 이루어지도록 제어기가 설계된 것을 확인할 수 있다. 스윙업 제어기를 설계하는 과정을 통해 학생들은 스윙업 제어 이론 뿐만 아니라 Arduino Due와 같은 임베디드 제어 디바이스를 이용하여 PWM을 생성하는 방법, 엔코더의 quadrature pulse를 처리하는 방법, interrupt를 지정하는 방법 등을 학습할 수 있다. HIL simulator를 이용하여 테스트를 마친 임베디드 제어기는 별도의 수정없이 그대로 실제의 도립진자 시스템의 제어에 적용할 수 있다. 그림 16은 설계된 제어기를 실제 도립진자 시스템에 그대로 적용했을 때 얻은 진자의 각변위를 도시하고 있다. 실제 실험에서도 임베디드 제어기가 스윙업 제어를 수행해 내는 것을 볼 수 있으며 HIL simulator를 이용한 실험에서의 각변위 양상과 실제 실험에서 얻은 각변위의 양상이 유사한 것으로부터 구현된 HIL simulator가 실제 시스템을 충실하게 모사하고 있음을 유추해 볼 수 있다.

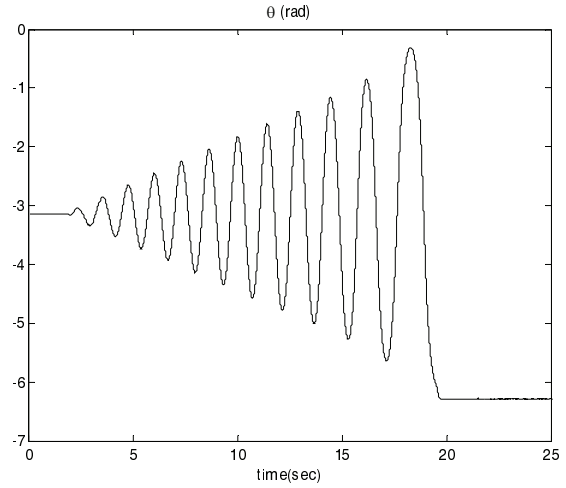

그림 15. HIL simulator를 이용한 제어실험에서의 진자 각변위.

Fig. 15. Pendulum angle obtained in the control experiment using a HIL simulator.

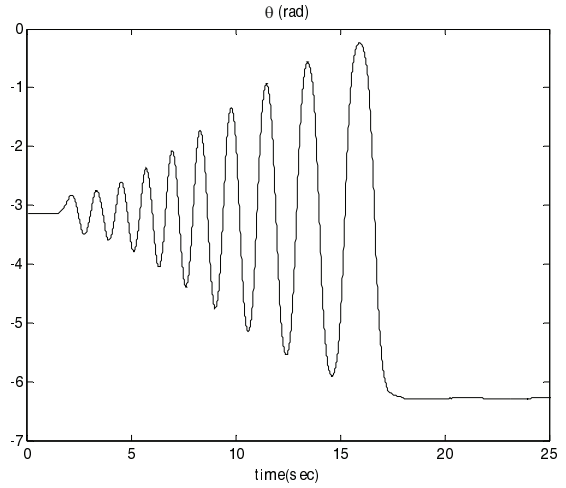

그림 16. 실제 도립진자를 이용한 제어실험에서의 진자 각변위.

Fig. 16. Pendulum angle obtained in the control experiment using the real inverted pendulum.

V. 결론

본 논문에서는 학생들에게 자동제어 및 임베디드 제어기 구현에 관한 교육을 효과적으로 수행하기 위해 도립진자 시스템에 대한 HIL simulator를 저가형 open-source hardware를 이용하여 구현하는 방법을 제안하였다. 제안된 HIL simulator는 Cortex-M4 기반의 Arch Max를 이용하여 모터 드라이브, 도립진자 시스템, 엔코더 펄스 생성부를 모사하도록 구현되었으며 태스크의 중요성을 고려하여 계산의 우선순위를 지정할 수 있게끔 Cortex-M4에서 지원하는 NVIC 기능을 활용하였다. 회전형 엔코더의 quadrature pulse 생성을 정확히 모사하기 위하여 회전속도에 기반한 pulse 생성 알고리즘을 제시하였으며 보상기를 통해 오차를 줄이는 방법을 제안하였다. 또한 HIL simulation 수행 중 발생하는 제어현상을 학생들이 시각적으로 확인할 수 있도록 VPython 기반의 에니메이션 기능을 개발하였다. 구현된 HIL simulator를 이용하여 도립진자 시스템에 대한 스윙업 제어를 임베디드 제어기로 구현해 보았으며 동일한 제어기를 수정없이 실제 시스템에 적용했을 때에도 성공적으로 동작하는 것을 실험을 통해 살펴보았다. 손쉽고 경제적인 가격으로 구매할 있는 open-source hardware를 기반으로 HIL simulator를 구성하는 방법을 제안함으로써 자동제어 및 임베디드 제어기 구현과 같은 강좌의 학습 효과를 크게 향상시킬 수 있는 방법을 제안했다는 것이 본 논문의 의의라고 할 수 있다.

REFERENCES

[1] R. Isermann, J. Schaffnit, and 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.

[2] H. P. Halvorsen, “Hardware in the loop simulation,” Technical Reports of Telemark University College, 2012.

[3] B. Lu, X. Wu, H. Figuero, and A. Monti, “A low-cost real-time hardware-in-the-loop testing approach of power electronics controls,” *IEEE Transactions on Industrial Electronics*, vol. 54, no. 2, pp. 919-931, 2007.

[4] W. Ren, M. Steurer, and T. L. Baldwin, “Improve the stability and the accuracy of power hardware-in-the-loop simulation by selecting appropriate interface algorithms,” *IEEE Transactions on Industrial Electronics*, vol. 44, no. 4, pp. 1286-1294, 2008.

[5] H. Hanselmann, “Hardware-in-the-loop simulation testing and its integration into a CACSD toolset,” *Proc. of IEEE International Symposium on Computer Aided Control System Design*, Dearbon, MI, pp. 152-156, Sep. 1996.

[6] D. Lee and C. G. Kang, “Improved wheelset speed implementation of a brake HILS system for a railway vehicle,” *Journal of Institute of Control Robotics and Systems (in Korean)*, vol. 21, no. 9, pp. 881-887, 2015.

[7] M. Margolis, *Arduino Cookbook*, 2nd Ed, O'Reilly, 2012.

[8] D. Molly, *Exploring Beaglebone*, Wiley, 2015.

[9] B. U. Jo, Y. S. Park, S. K. Seo, J. G. Kim, and Y. S. Lee, "Rapid prototyping of head-of-bed angle measurement system using open-source hardware," *Journal of Institute of Control Robotics and Systems (in Korean)*, vol. 21, no. 11, pp. 1038-1043, 2015.

[10] K. Graichen, M. Treuer, and M. Zeitz, "Swing up of the double pendulum on a cart by feedforward and feedback control with experimental validation," *Automatica*, vol. 43, pp. 63-71, 2007.

[11] T. Glück, A. Eder, and A. Kugi, "Swing-up control of a triple pendulum on a cart with experimental validation," *Automatica*, vol. 49, no. 3, pp. 801-808, 2013.

[12] J. H. Yang, S. Y. Shim, J. H. Seo, and Y. S. Lee, "Swing up control for an inverted pendulum with restricted cart rail length," *International Journal of Control, Automation, and Systems*, vol. 70, no. 4, pp. 674-680, 2009.

[13] P. Grasblum, "Using the HCS08 TPM module in motor control applications," Freescale Application Note.

[14] S. C. Chapra, *Applied Numerical Method of Chapra*, 3rd Ed, Mc Graw Hill, 2012.

[15] D. Y. Yoon, "Cortex-M4 computation speed benchmark test," Technical Note, http://cafe.naver.com/micca/5658.

최 진 석

2015년 인하대학교 전기공학과 졸업. 2015년~현재 인하대학교 대학원 전기공학과 석사과정 재학 중. 관심분야는 로봇 공학, 선형 및 비선형 시스템 제어, 최적 제어.

이 영 삼

제어 · 로봇 · 시스템학회 논문지, 제15권 제4호 참조.