JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지 Vol. 23, No. 5: 547~554, May 2019

모듈 기반 교육용 아두이노 호환 키트 제작

허경용*

Implementation of an Arduino Compatible Modular Kit for Educational Purpose

Gyeongyong Heo*

*Associate Professor, Department of Electronic Engineering, Dong-eui University, Busan, 47340 Korea

요 약

2015년 교육과정 개편으로 중학교 정보교과가 필수로 지정됨에 따라 초중고등학교는 물론 대학에서도 프로그래밍 교육을 강화하고 있다. 아두이노는 프로그래밍 교육을 위해 사용되는 대표적인 도구 중 하나로 그 유용성은 다양한 사례 연구를 통해 입증되었다. 하지만 기존 아두이노 기반 키트는 연결 방법이 복잡하거나, 확장성이 떨어지는 등 하드웨어 의존적인 단점을 가지고 있다. 이러한 문제점을 보완하기 위한 방안으로 모듈 기반의 구조를 가지고, 동일한 인터페이스를 통해 기능 확장이 가능하며, 다양한 수준에서 학습에 사용할 수 있는 아두이도 호환 키트 설계를 제안하였다. 이 논문에서는 제안한 키트의 요구 조건을 만족시키는 FRUTO 키트와 이를 사용하기 위한 소프트웨어 구현 방법을 설명한다. FRUTO 키트는 몇 번의 디자인 변경을 통해 현재의 형태로 결정되었으며, 현재 출시 전 테스트 진행을 준비 중에 있다.

ABSTRACT

With the curriculum revision in 2015, informatics for secondary high schools was designated as mandatory. As a result, there is an increasing interest in programming in elementary and junior high schools as well as in universities. Arduino is one of the famous tools for programming education, and the usefulness of it has been proven through various case studies. However, existing Arduino-based kits have hardware-dependent drawbacks such as complicated wiring, poor scalability, etc. To overcome these problems, we proposed a kit design, which has a module-based structure, can be extended through one common interface, and can be used for learning at various levels. In this paper, we describe the implementation details of FRUTO kit and a software to use it, which satisfies the proposed design criteria. FRUTO kit has been determined in its current form through several design changes, and is under pre-test before launching.

키워드 : 아두이노, 교육용 키트, 모듈형 디자인, FRUTO 키트, FRUTO 라이브러리

Keywords : Arduino, Educational Kit, Modular Design, FRUTO Kit, FRUTO Library

Received 8 January 2019, **Revised** 27 January 2019, **Accepted** 7 February 2019

*** Corresponding Author** Gygoneyong Heo(E-mail:hgycap@deu.ac.kr, Tel:+82-51-890-1675)
Associate Professor, Department of Electronic Engineering, Dong-eui University, Busan, 47340 Korea

Open Access http://doi.org/10.6109/jkiice.2019.23.5.547 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

사물인터넷에 대한 관심의 증가에 따라 사물인터넷에서 사물을 구현하기 위한 지능형 장치에 대한 수요와 지능형 장치를 위한 소프트웨어의 중요성이 부각되고 있으며 이에 따라 소프트웨어 교육에 대한 중요성도 어느 때보다 강조되고 있다. 소프트웨어 교육의 필요성은 2014년 영국을 시작으로 프로그래밍 교육을 정규 교과과정에 포함하고 있다는 점에서 단적으로 나타나고 있다. 우리나라에서는 2015년 발표된 개정 교육과정 총론에서 프로그래밍과 관련된 피지컬 컴퓨팅이 포함되었다[1].

프로그래밍 교육은 흔히 코딩 교육이라고 알려져 있지만, 단순히 프로그래밍 언어를 사용하여 컴퓨터용 프로그램을 작성하는 것이 아니다. 프로그래밍 교육이란 미래 인재를 위한 역량을 키우기 위한 교육의 일환이다[2-4]. 많은 연구에서 미래 인재의 역량으로 창의성과 논리성을 갖춘 문제 해결 능력, 그리고 다른 이와의 소통을 통해 공유하고 협업할 수 있는 능력이 강조되고 있다. 소프트웨어 교육은 이러한 역량을 키우는 데 효과적이라는 점이 다양한 연구들을 통해 밝혀지고 있다. 프로그래밍을 통해 학생은 논리적으로 문제를 파악하고 나만의 창의적인 해결방안을 만들어낼 수 있고, 추상적 아이디어를 프로그램으로 구체화하면서 성취감을 느낄 수 있으며, 문제점을 찾고 고쳐가는 과정을 통해 점진적 문제 해결을 경험할 수 있다. 또한, 자신이 만든 프로그램의 공유를 통해 얻을 수 있는 소통 능력 역시 소프트웨어 교육의 주요 산출물이라 하겠다.

교육과정 총론에 포함된 피지컬 컴퓨팅은 프로그래밍 교육을 위한 방법의 하나다. 피지컬 컴퓨팅은 뉴욕대학의 ITP(Interactive Telecommunications Program)에서 상호작용이 가능한 디자인을 가르치기 위해 시작된 것으로, 자연 또는 인간으로부터 얻어지는 물리적인 정보를 처리하고 그 결과를 다시 물리적인 형태로 출력하여 자연 또는 인간과 상호작용하는 컴퓨팅을 가리킨다[5]. 교육과정에 피지컬 컴퓨팅이 도입된 것은 문제 해결 및 논리적 사고에 프로그래밍이 도움이 된다는 점과 로봇 관련 교육이 동기 부여는 물론 협업 및 소통 능력 향상에도 긍정적인 영향을 미친다는 연구 결과가 반영된 것이다.

피지컬 컴퓨팅 교육을 위해 사용되는 대표적인 도구 중 하나가 아두이노다[6]. 아두이노는 예술가를 위한 마이크로컨트롤러 프로젝트로 시작되어 STEAM(Science, Technology, Engineering, Arts, and Mathematics) 교육과 사물인터넷 관련 분야에서 다양한 교육 효과가 보고되고 있다[7-9].

아두이노를 활용하는 교육이 증가함에 따라 아두이노를 기반으로 하는 다양한 교육 도구 개발에 관한 연구 역시 진행되고 있다[10-11]. 하지만 아두이노를 교육용으로 사용하는 경우의 가장 큰 문제점은 하드웨어 의존성이 높다는 점이다. 하드웨어 의존성은 아두이노를 도구로 사용하는 것이 아니라 목적으로 사용하는 경우에 적합하다. 이러한 하드웨어 의존성을 줄이기 위해 아두이노 키트를 도구로 활용할 수 있도록 해주는 사용성(usability)과 다양한 분야에서 활용할 수 있도록 해주는 확장성(scalability)이 개선된 키트 설계를 제안하고[12], 이를 구현하기 위한 요구 조건을 정의하였다[13]. 이 논문에서는 정의한 요구 조건을 만족하는 FRUTO 키트와 이를 활용하기 위한 FRUTO 라이브러리의 구조와 특성을 다룬다. FRUTO 키트는 아두이노 기반의 모듈형 하드웨어로 정의한 사용성과 확장성을 만족시킨다. FRUTO 키트는 몇 번의 디자인 변경을 통해 현재 형태로 결정되었으며, 프로토타입 제작 후 출시 전 테스트를 준비 중에 있다.

이 논문의 구성은 다음과 같다. 2장에서는 기존 아두이노 키트의 문제점과 이를 해결하기 위한 방법을 보인다. 3장에서는 제시된 디자인 기준을 만족하는 FRUTO 키트 및 FRUTO 라이브러리의 구조와 특징 및 이의 구현 방법을 설명하며, FRUTO 키트의 개선 방향은 4장에서 언급한다.

Ⅱ. 키트 설계 방향

교육용으로 사용되는 아두이노 키트는 여러 가지 형태가 있다. 하지만 대부분의 기존 키트들은 주변장치를 연결하기 위해 서로 다른 수의 연결선을 사용해야 하므로 연결 자체가 어려워 사용성이 떨어지거나, 키트를 설계한 방식으로만 사용할 수 있고 주변장치의 추가가 어려워 확장성이 떨어지는 문제점이 있다. 이처럼 낮은 사용성과 확장성을 개선하기 위해서는 일관된 방식으로 주변장치를 연결할 수 있으며 새로운 주변장치를 쉽게

추가할 수 있는 방법이 필요하며, 그 방법의 하나가 I2C(Inter-Integrated Circuit) 통신을 기본으로 모듈을 설계하는 방법으로[12] 기존 아두이노 기반 키트에서는 찾아볼 수 없는 방식이다. I2C 통신은 여러 주변장치를 캐스케이딩(cascading) 방식으로 연결할 수 있도록 해주며, 모든 연결은 전원과 데이터를 위한 4개 연결선을 사용하므로 쉽고 간단하게 시스템을 구성할 수 있다.

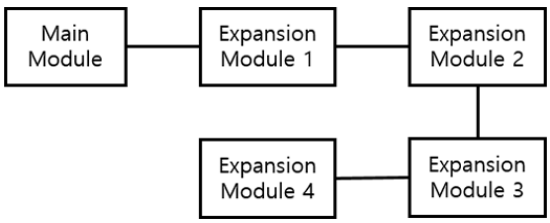

Fig. 1 Module connection diagram

키트를 위한 소프트웨어 개발 방법 역시 마찬가지다. 서로 다른 주변장치는 서로 다른 통신 방식을 사용하므로 주변장치에 따라 서로 다른 방식의 코드를 작성해야 한다. 이처럼 서로 다른 제어구조는 추상화된 라이브러리를 제공함으로써 통일시킬 수 있다. 아두이노의 라이브러리는 이미 아두이노 보드에 사용된 AVR 시리즈 마이크로컨트롤러를 위한 저수준의 함수를 추상화하여 제공하고 있지만, 여전히 하드웨어 종속적이다. 따라서 제안하는 디자인에서는 그림 2에서와 같이 마이크로컨트롤러를 위한 저수준의 함수와 아두이노 라이브러리를 사용하여 일관된 입출력 함수를 설계하고, 이를 하드웨어와 마찬가지로 모듈형식으로 구현하여 제공한다.

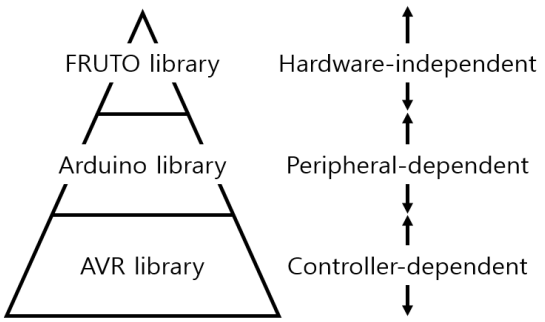

Fig. 2 Software abstraction layer

Ⅲ. FRUTO 키트

기존 키트들의 문제점을 보완하기 위해 제안된 설계 기준[13]을 만족하는 FRUTO 키트는 다음과 같은 특징을 가지고 있다.

- FRUTO 키트는 모듈형으로 구성되며 메인 모듈과 확장 모듈로 나눌 수 있다. 메인 모듈은 모듈간 통신을 위한 I2C 통신에서 마스터로, 확장 모듈은 슬레이브로 동작한다. 이때 모든 모듈에는 ATmega328 마이크로컨트롤러가 포함되어 모듈 간 I2C 통신을 담당하며 모든 모듈이 아두이노 우노와 호환되도록 보장한다.
- 메인 모듈은 1개의 I2C 커넥터를, 확장 모듈은 2개의 I2C 커넥터를 가져 그림 1의 캐스케이딩 연결을 지원한다. 모든 I2C 연결은 4핀 커넥터를 사용하므로 연결의 일관성을 유지할 수 있다. 또한, I2C 연결은 주소에 의해 모듈이 구별되므로 확장 모듈의 연결 순서와 무관하게 동일한 동작을 보장한다.
- 모듈을 제어하기 위한 프로그램은 FRUTO 라이브러리를 사용하여 이루어진다. FRUTO 라이브러리는 하드웨어 제어를 위한 저수준의 함수를 read, write 등의 추상화된 공통 함수로 추상화한 클래스 라이브러리로 구성된다.
- FRUTO 라이브러리는 FRUTO 키트의 일부로 제공되며 초급 사용자는 마스터 모듈을 위한 아두이노 스케치를, 중급 사용자는 슬레이브 모듈을 위한 아두이노 스케치를, 고급 사용자는 마스터 모듈을 위한 클래스 라이브러리를 작성하는 등의 단계별 활용이 가능하다.
- FRUTO 하드웨어 역시 초급 사용자를 위해서는 모듈 연결을 통한 사용, 중급 사용자를 위해서는 메인 모듈에 주변장치 연결을 통한 사용 또는 확장 모듈의 단독 사용, 고급 사용자를 위해서는 확장 모듈의 설계 및 제작 등 단계별 활용이 가능하다.

3.1. 하드웨어 – FRUTO 모듈

그림 3은 8개 LED를 포함하고 있는 LED 모듈의 회로도를 나타낸다. 확장 모듈 설계는 크게 컨트롤러, 주변장치, I2C 커넥터 등의 3부분으로 구성되며 메인 모듈에는 주변장치 대신 아두이노 우노와 호환되는 핀헤더가 포함된다.

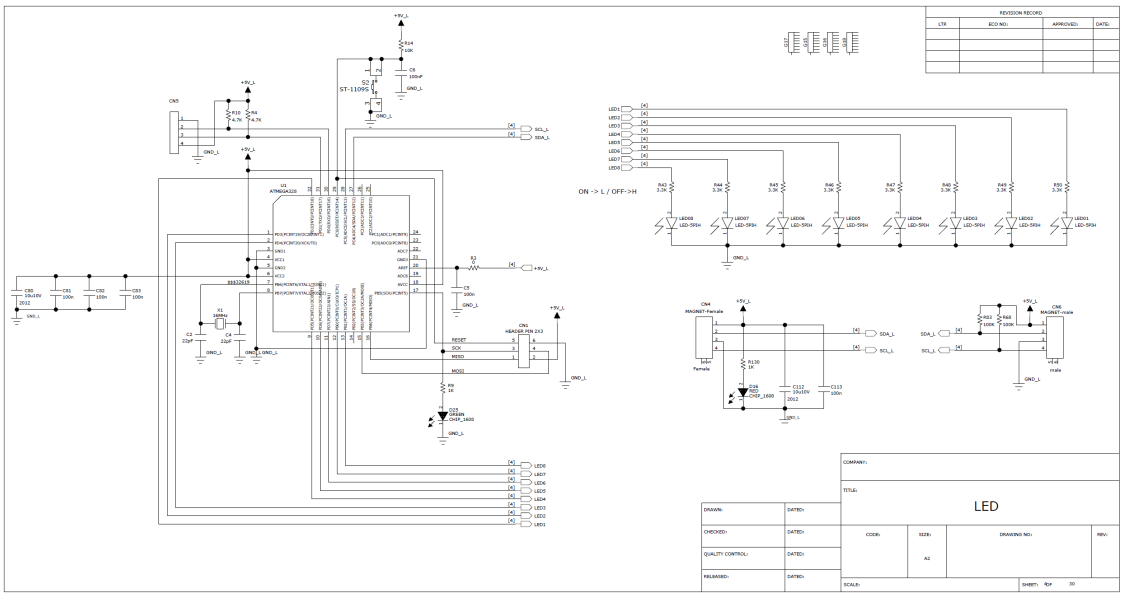

Fig. 3 LED module design

모듈을 위한 커넥터는 몰렉스 커넥터, RJ11 커넥터, 4극 이어폰 잭 등을 테스트한 후 연결이 쉽고 크기가 작은 자석 커넥터를 사용하여 제작하였다. 모듈 연결은 모듈 간 캐스케이딩 연결을 기본으로 하지만 유연한 연결이 가능하도록 전용 케이블과 연결 분배 모듈 역시 제작하였다. 그림 4는 메인 모듈에 분배 모듈과 케이블을 사용하여 LED 모듈, 가변저항 모듈, 7세그먼트 모듈 등을 연결한 예를 나타낸다.

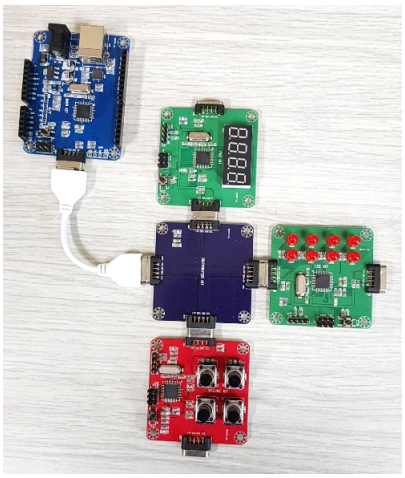

Fig. 4 Module connection

3.2. 소프트웨어 – FRUTO 라이브러리

FRUTO 키트를 위한 프로그램 작성을 위해서는 마스터를 위한 아두이노 코드, 마스터 프로그램을 위한 C++ 라이브러리, 슬레이브를 위한 아두이노 코드 등이 필요하다. 그림 5는 마스터를 위한 아두이노 코드로 LED 모듈에 포함되어 있는 LED를 점멸하기 위해 사용된다. 그림 5에는 하드웨어 종속적인 코드가 포함되어 있지 않으며, LED를 제어하는 on(), off() 등의 추상화된 함수들만 사용되었다. 그림 5는 초급 사용자가 FRUTO 키트를 사용하기 위해 작성해야 하는 유일한 코드다.

```
#include <FRUTO.h>

void setup() {
    FRUTO.begin();
}

void loop() {
    Module_LED.on();            // turn on 8 LEDs
    delay(1000);
    Module_LED.off();           // turn off 8 LEDs
    delay(1000);
}
```

Fig. 5 Master program – user_sketch.ino

그림 5의 코드가 동작하기 위해서는 추상화된 명령을 I2C 통신을 통해 슬레이브로 전달하기 위한 FRUTO 라이브러리가 필요하다. FRUTO 라이브러리는 모듈로 데이터 전달을 담당하는 모듈 라이브러리(module library)와 FRUTO 키트로 구성되는 시스템을 위한 시스템 라이브러리(system library)로 나눌 수 있다. 그림 6과 그림 7은 LED 모듈을 위한 라이브러리로, 마스터의 추상화된 명령에 따라 I2C 통신을 통해 LED 제어 데이터를 전달하는 역할을 한다.

```
#include "Module_LED.h"

void _Module_LED::on(void){       // turn on 8 LEDs
    write(0xFF);
}

void _Module_LED::off(void){      // turn off 8 LEDs
    write(0x00);
}

void _Module_LED::write(byte value){
    Wire.beginTransmission(MODULE_LED);
    Wire.write(value);
    Wire.endTransmission();
}
```

Fig. 6 LED module library – Module_LED.cpp

```
#ifndef _MODULE_LED_
#define _MODULE_LED_

#include "Arduino.h"
#include <Wire.h>

#define MODULE_LED 11          // I2C address

class _Module_LED {
    public:
        void on(void);             // turn on 8 LEDs
        void off(void);            // turn off 8 LEDs
        void write(byte value);    // send LED data
};

// global LED module instance
extern _Module_LED Module_LED;

#endif
```

Fig. 7 LED module library – Module_LED.h

그림 8과 그림 9는 FRUTO 시스템을 위한 라이브러리로 마스터에서 필요한 설정 작업을 수행한다.

```
#include "FRUTO.h"

void _FRUTO::begin(void){
    Wire.begin();          // start I2C as master
}
```

Fig. 8 System library – FRUTO.cpp

```
#ifndef _FRUTO_
#define _FRUTO_

#include "Arduino.h"
#include <Module_LED.h>

class _FRUTO {
    public:
        void begin(void);
};

extern _FRUTO FRUTO;

#endif
```

Fig. 9 System library – FRUTO.h

```
#include <Wire.h>
#define MODULE_LED 11          // I2C address
int LED_pins[] = { 2, 3, 4, 5, 6, 7, 8, 9 };
byte LED_status = 0;           // LED state

void setup() {
    for (int i = 0; i < 8; i++)
        pinMode(LED_pins[i], OUTPUT);
    // start I2C as slave
    Wire.begin(MODULE_LED);
    // register data receive handler
    Wire.onReceive(receiveFromMaster);
    LED_control();
}

void loop() { }

// receive data from master
void receiveFromMaster(int bytes) {
    LED_status = Wire.read();
    LED_control();
}

void LED_control(void) {
    for (int i = 0; i < 8; i++){
        digitalWrite(LED_pins[i],
            ((LED_status >> i) & 0x01) );
    }
}
```

Fig. 10 Slave program – Module_LED_slave.ino

Table. 1 Codes for FRUTO kit

Name		Code Sample	Uploaded to	Function	User Level	Remarks
Master Program		Fig. 5	Main Module	Control extension modules with abstracted class library	Beginner	The only one that not provided with FRUTO kit
FRUTO Library	Module Library	Fig. 6, 7	(Main Module)*	Transmit control data to an extension module	Expert	*Used in conjunction with master program rather than used independently
	System Library	Fig. 8, 9	(Main Module)*	Initialize a system built with FRUTO kit	Expert	
Slave Program		Fig. 10	Extension Module	Receive data from a main module and control the peripherals accordingly	Intermediate, Expert	Pre-burned in each extension module
Module Test Program		-	Extension Module	Test the peripherals built-in an extension module	Intermediate	Required only one extension module

FRUTO 라이브러리를 통해 전달된 데이터는 슬레이브에서 수신되고 처리된다. 그림 10은 LED 모듈에서 마스터의 데이터를 수신하고, 수신된 데이터에 따라 실제 LED를 제어하는 코드다.

제시한 코드의 수가 많고 구성이 복잡해 보이지만 대부분의 사용자는 그림 5와 같이 추상화된 코드를 작성하는 것으로 충분하다. 마스터 프로그램과 함께 사용되는 FRUTO 라이브러리(그림 6, 7, 8, 9)는 키트와 함께 배포되며, 슬레이브 프로그램(그림 10)은 FRUTO 키트의 각 모듈에 설치된 상태로 판매된다.

표 1은 각 코드의 기능과 특징을 요약한 것이다. 표 1에 나열한 코드 중 모듈 테스트 프로그램은 확장 모듈 하나만을 사용하여 주변장치를 독립적으로 테스트할 수 있도록 해주는 프로그램이다. 모듈 테스트를 위한 코드의 예를 제시하지는 않았지만, 슬레이브 프로그램에서 I2C 통신을 통한 데이터 수신 부분이 빠진 코드와 유사하며 FRUTO 라이브러리와 함께 배포된다.

3.3. FRUTO 키트의 장점

FRUTO 키트는 아두이노 우노와 호환되는 모듈형 하드웨어와 이를 지원하는 소프트웨어 라이브러리인 FRUTO 라이브러리로 구성된다. 하드웨어 측면에서 FRUTO 키트의 가장 큰 장점은 주변장치 추가가 쉽다는 점이다. 필요로 하는 기능을 가진 모듈을 동일한 4핀 커넥터를 통해 연결하는 것만으로 하드웨어 측면에서의 연결은 끝난다. 모듈 연결은 I2C 통신을 통한 캐스케이딩 방식으로 이루어지므로 직관적인 연결이 가능하며, I2C 통신의 특성에 따라 모듈의 연결 순서와 무관하게 동작한다. FRUTO 키트를 사용한 수업의 관찰 결과에 따르면 기존 키트를 사용하는 수업에 비해 하드웨어 구성에 소요되는 시간은 1/3이하인 것으로 나타났다.

소프트웨어 측면에서 FRUTO 라이브러리의 특징 역시 추상화된 클래스 라이브러리를 통해 모듈형식으로 구성되어 있다는 점이다. 모듈형으로 구성되는 하드웨어의 특징에 따라 저수준의 하드웨어 제어 작업은 모듈별로 분산 처리되며, 추상적인 하드웨어 제어 작업 지시를 통해 프로그램을 구성할 수 있다. 이를 위해서는 작업 지시를 담당하는 메인 모듈과 실제 작업을 처리하는 확장 모듈의 연계가 필요하며 FRUTO 라이브러리의 역할이 바로 이것이다. FRUTO 라이브러리를 사용하면 LED 모듈과 같이 간단한 주변장치를 제어하는 경우는 아두이노 라이브러리를 사용하는 경우에 비해 약 1/2, LED 매트릭스와 같이 복잡한 주변장치를 제어하는 경우는 약 1/4 등 전체적으로 1/3이하인 것으로 나타났다.

하드웨어 연결과 프로그래밍 작업이 간단해진다는 점 이외에 FRUTO 키트의 장점 중 하나는 다양한 수준의 사용자를 위한 교육 도구로 활용할 수 있다는 점이다. 표 2는 초급, 중급, 고급 등 학습자의 수준에 따라 FRUTO 키트를 활용하는 방법의 예를 제시하고 있다. 초급 사용자를 위해서는 최소한의 하드웨어에 대한 이해만으로도 시스템을 구성하고 이를 위한 프로그램을 쉽고 간편하게 작성할 수 있도록 해준다는 점이 가장 큰 장점이다. FRUTO 키트는 기존 키트 대비 약 1/3의 시간에 동일한 결과를 확인할 수 있으므로 학습 도구로 효과적인 활용이 가능하다. 이외에 중급 사용자는 기존 아두이노 환경을 그대로 사용할 수 있다는 점, 그리고 고급

Table. 2 FRUTO environment utilization by level

Level	Hardware		Software		Remarks
	Used hardware	Hardware dependency	Programming area	Use Arduino library	
Beginner	FRUTO kit	LOW	Master program	×	Use FRUTO library
Intermediate	Main and extension module alone	MID	Module test program Slave program	○	Similar to using existing Arduino compatible kits
Expert	Module DIY	HIGH	FRUTO library	○	

사용자는 아두이노를 기반으로 하는 키트를 설계하고 구현해볼 수 있다는 점 또한 기존 키트와 차별화되는 점이라 할 수 있다.

Ⅳ. 결 론

초중고등학교에서 정보 교과가 필수로 지정됨에 따라 이를 위한 적절한 학습 도구에 대한 수요가 증가하고 있다. 또한, 아두이노의 활용을 통해 문제 해결 능력과 팀 단위의 협업 능력을 키울 수 있다는 연구 결과는 아두이노를 정보 교과에서 활용하려는 다양한 시도로 이어지고 있다. 하지만 기존 아두이노 기반의 학습 도구는 다양한 활용에 제한이 있어 이를 해결할 수 있는 새로운 아두이노 기반 학습 도구 디자인 방향이 제시되고 있다. 이 논문에서는 이전 연구를 통해 제시된 디자인 요구 조건을 만족하는 FRUTO 키트와 이를 지원하는 FRUTO 라이브러리의 구조와 특징을 설명하였다.

현재 FRUTO 키트는 10종의 모듈이 개발되어 테스트를 진행 중이며, 학습 현장에서의 실제 사용 테스트를 준비하고 있다. 더불어 테스트 과정에서 수집된 의견에 대해 수정 및 확장 방안을 논의하고 있다. FRUTO 환경이 기능적으로는 설계 기준을 만족시키지만, 학습 대상의 범위를 넓히기 위해 스크래치(Scratch)와 같은 블록 프로그래밍 도구를 사용할 수 있도록 하는 것이 첫 번째 확장 방안이다. 또한, 레고를 포함하여 기존 학습 도구들과의 하드웨어 호환성을 확보함으로써 프로젝트 기반 학습에서 보다 다양하게 활용할 수 있도록 하는 것이 두 번째이며 이들 사항에 대해서는 추후 업그레이드 진행을 고려하고 있다.

REFERENCES

[1] Ministry of Education, *2015 revised curriculum general*, Notice no. 2015-74, Nov. 2015.

[2] H. Bort, M. Czarnik, and D. Brylow, “Introducing Computing Concepts to Non-Majors: A Case Study in Gothic Novels,” in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, Kansas City, Missouri, USA, pp. 132-137, 2015.

[3] S. H. Park, “Development and application of programming education model to enhance creative problem solving abilities,” Master Thesis, Seoul National University of Education, 2016.

[4] S. Kim, and Y. Lee, “Development and application of Arduino-based education program for high school students,” *Journal of Theoretical and Applied Information Technology*, vol. 95, no. 18, pp. 4367-4375, Sep. 2017.

[5] Dan O’Sullivan, and Tom Igoe, *Physical Computing*, 1st ed. Thomson, 2004.

[6] B. M. Hoffer, “*Satisfying STEM Education Using the Arduino Microprocessor in C Programming*,” Master Thesis, East Tennessee State University, 2012.

[7] J. T. Kim, “Analyses of Requirement of Integrated Security for Secure Internet of Things,” *Asia-pacific Journal of Multimedia Services Convergent with Art, Humanities, and Sociology*, vol. 5, no. 4, pp. 667-674, Aug. 2015.

[8] P. Plaza, E. Sancristobal, G. Fernandez, M. Castro, and C. Perez, “Collaborative robotic educational tool based on programmable logic and Arduino,” in *Proceedings of the 2016 Technologies Applied to Electronics Teaching*, Seville, Spain, pp. 1-8, 2016.

[9] J. C. Martinez-Santos, O. Acevedo-Patino, and S. H. Conteras-Ortiz, “Influence of Arduino on the Development of Advanced Microcontrollers Courses,” *IEEE Revista Iberoamericana de Technologias del Aprendizaje*, vol. 12, no. 4, pp. 208-217, Nov. 2017.

[10] A. Garrigos, D. Marroqui, J. M. Blanes, R. Gutierrez, I. Blanquer, and M. Canto, "Designing Arduino electronic shields: Experiences from secondary and university courses," in *Proceedings of the 2017 IEEE Global Engineering Education Conference*, Athens, Greece, pp. 934-937, 2017.

[11] Z. Cheng, Y. Li, and R. West, "Qduino: A Multithreaded Arduino System for Embedded Computing," in *Proceedings of the 2015 IEEE Real-Time Systems Symposium*, San Antonio, TX, USA, pp. 261-272, 2015.

[12] Tn1, Microcontroller Kit, Korean Patent 1017353010000 to Korean Intellectual Property Office, 2017.

[13] G. Heo, and J. Jung, "Arduino Compatible Modular Kit Design for Educational Purpose," *Journal of the Korea Institute of Information and Communication Engineering*, vol. 22, no. 10, pp. 1371-1378, Oct. 2018.

허경용(Gyeongyong Heo)

연세대학교 전자공학과 공학석사 (1996)
University of Florida 컴퓨터공학과 공학박사 (2009)
동의대학교 전자공학과 교수 (2012~현재)
※관심분야 : 인공지능, 패턴인식, IoT 시스템