The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 18, No. 6, pp.139-143, Dec. 31, 2018. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2018.18.6.139
JIIBC 2018-6-18

IoT 디바이스를 위한 아두이노 타이머 콜백 분석

Analysis of Arduino Timer Callback for IoT Devices

공동환*, 신승중**

Dong-Hwan Gong*, Seung-Jung Shin**

요 약 오픈 소스 하드웨어 기반의 아두이노는 많은 IoT 디바이스로 사용되고 있으며 IoT 디바이스들은 다양한 입출력을 위한 멀티태스킹을 필요로 한다. 아두이노 기반의 멀티태스킹을 위해 많이 사용되는 몇 가지 방법 중 세 가지 방법인 millis() 를 사용한 타이밍 호출 방법, SimpleTimer 라이브러리 사용 방법, Timer 라이브러리 사용 방법을 비교 분석한다. 각 방법의 측정과 시간지연으로 발생되는 실행 오류를 측정하기 위해 두 가지 상황을 생성하여 분석한다. 첫 번째 상황은 일정한 크기의 임의 작업을 10개 생성하여 각 방법의 시간지연을 측정하고 두 번째 상황은 일정하지 않는 크기의 임의 작업을 10개 생성하여 Timer 라이브러리의 시간지연으로 발생되는 실행 오류를 비교 분석하였다. 첫 번째 상황에서 millis() 타이밍 호출 방법과 Simple Timer 라이브러리 사용 방법은 비슷한 시간지연이 발생하였고 Timer 라리브러리 사용 방법은 더 많은 시간지연이 발생하였다. 두 번째 상황에서는 크기가 작은 작업들이 시간지연으로 정확한 타이밍에 콜백되지 않는 실행 오류가 발생되었다.

Abstract Arduino, based on open source hardware, is used in many IoT devices, and IoT devices require multitasking for various inputs and outputs. Among the several methods used for multitasking based on Arduino, we compare three methods: Timing Call by using millis(), Simple Timer library method, and Timer library method. In order to measure the execution error caused by measurement and time delay of each method, two situations are created and analyzed. In the first case, 10 random tasks of a certain size are generated to measure the time delay of each method. In the second situation, 10 random tasks of a certain size are generated to compare execution errors caused by the time lag of the Timer library. In the first case, the millis() timing call method and the Simple Timer library method have a similar time delay and the Timer library method has more time delay. In the second situation, an execution error occurred in which small-size tasks were not called back at the correct timing due to the time delay.

Key Words : IoT, Arduino, Callback, Timer, Analysis

Ⅰ. 서 론

사물인터넷(Internet of Things)은 ICT 분야를 이끄는 핵심 성장 동력으로 사물과 사물을 연결하고 사람과 사물을 연결하여 초연결 사회를 구축하고 여러 가지 지능형 서비스를 바탕으로 각 사용자에게 맞는 여러 서비스를 제공한다. 사물인터넷은 수많은 사물에서 수집된 센서 정보를 IoT플랫폼을 통해 실시간으로 저장, 가공, 분석하여 사용자에게 맞는 맞춤형 서비스 제공이 가능하도록 한다.[4],[5]

*정회원, 한양사이버대학 기계자동차공학부
**정회원, 한세대학교 ICT디바이스학과
접수일자: 2018년 9월 7일, 수정완료: 2018년 11월 17일
게재확정일자: 2018년 12월 7일

Received: 7 September, 2018 / Revised: 17 November, 2018 / Accepted: 7 December, 2018
*Corresponding Author: expersin@hansei.ac.kr
Dept. of IT, Hansei University, Korea

사물인터넷은 크게 사물들의 정보를 수집하기 위한 센서기술, 네트워크 인프라를 위한 네트워크 기술, 정보의 생성, 수집, 공유, 활용을 위한 서비스 인터페이스 기술, 해킹, 정보 유출, 서비스 거부 등을 방지하기위한 보안 기술인 4가지로 구분할 수 있다.[1],[9]

사물인터넷은 수많은 사물들의 정보를 수집하기 위해 센서 노드 디바이스를 필요로 하며 이런 디바이스는 쉽고 빠른 개발과 확장성을 위해 오픈 소스 하드웨어를 이용하여 개발 가능하다.[2]

본 논문에서는 사물인터넷 디바이스로 널리 사용되고 있는 오픈 소스 하드웨어인 아두이노의 멀티태스킹에 사용되는 타이머 콜백 라이브러리의 성능을 측정하고 분석한다.

II. 오픈 소스 하드웨어와 아두이노

1. 오픈 소스 하드웨어

오픈 소스 하드웨어(Open-Source Hardware)는 하드웨어를 누구나 쉽게 만들고 배포하고 수정할 수 있도록 하드웨어 설계(Design)관련 정보를 공개하고 소프트웨어 또한 오픈 소스 소프트웨어(Free and Open-Source Softwore)를 바탕으로 모든 정보를 공개하여 기술과 제품 개발의 발전을 높인다.[8],[10]

오픈 소스 하드웨어의 대표적인 개발 플랫폼으로는 아두이노(Arduino), 위즈네트(Wiznet w7500), 라즈베리파이(Raspberry Pi), 비글 보드(Beagle Board) 등이 있다.

2. 아두이노

아두이노는 오픈소스와 오픈하드웨어를 기반으로 개발되고 있는 세계적으로 가장 널리 사용되는 마이크로컨트롤러로 통합 개발 도구와 환경(IDE)을 모두 제공한다.[6] 아두이노의 하드웨어는 아트멜(Atmel)사의 8비트 AVR을 기반으로 개발이 시작되었으며 현재는 ARM 계열의 Cortex-M0와 Cortex-M3등을 사용하여 개발된 마이크로컨트롤러들이 존재한다. 아두이노 공식 보드로는 아두멜(Atmel)사의 ATmega8, Atmega168, ATmega 328과 같은 megaAVR 시리즈가 주로 상용되어 16MHz 크리스털이 내장되어 있다. 아두이노의 모든 보드는 RS-232 직렬 커넥터를 통행 프로그램되고 대표보드인 Uno는 6개의 아날로그 Input과 14개의 디지털 I/O핀을 제공한다.[12]

III. 아두이노 타이머 콜백

1. 아두이노의 멀티태스킹

아두이노는 순차적으로 명령을 실행하는 작은 프로세서이므로 멀티태스킹에 적합하지 않은 마이크로 컴퓨터라 할 수 있다. 하지만 아두이노가 가지고 있는 많은 장점들로 많은 IT 응용분야에 아두이노가 사용되고 있다.

대부분의 IoT 디바이스들은 IoT 플랫폼의 제어를 수신하며 여러 센서들의 값을 동시에 입력받거나 다른 기기와 통신하기 위해 멀티태스킹을 필요로 한다. 아두이노는 IoT뿐만 아니라 마이크로컴퓨터가 필요한 여러 응용분야에 사용되고 있으며 아두이노의 멀티태스킹은 필수적인 작업이 되어 가고 있다.[3],[7]

명령이 순차적으로 실행되는 단순한 아두이노는 멀티태스킹을 위한 방법으로 일정한 타이밍에 따라 각 기능의 함수를 호출할 수 있도록 하거나 일정한 타이밍에 따라 인터럽트처럼 정해진 시간에 이벤트를 발생시켜 콜백 함수를 호출하는 방식으로 타이머를 사용하여 멀티태스킹을 수행한다.

대표적인 아두이노 타이머 라이브러리에는 Timer와 SimpleTimer가 있으며 오픈 S/W 라이선스에 맞춰 C++ 클래스로 구현되었다.[11]

2. 아두이노 멀티태스킹 방법

아두이노 타이머 라이브러리는 모두 콜백 함수를 사용하여 구현되었으며 타이머에 콜백 함수를 등록하면 millis()를 사용하여 시간을 계산하고 등록한 이벤트 시간이 되면 등록된 콜백 함수를 호출하는 형태로 구현되었다.

표 1. 실험 환경
Table 1. Experiment Environment

실험 환경	실험 환경 값
아두이노 보드	ARDUINO UNO Rev3
타이머에 등록한 함수의 개수	10개
함수를 호출할 이벤트 시간 간격	0.5초~ 2초
실험 함수의 기능	13번 포트에 digitalWrite()를 7000번 호출
시간 측정 반복 횟수	20번

1) millis()를 사용한 타이밍 함수 호출

loop()가 실행되는 동안 1/1000초단위의 정밀도로 시간을 반환하는 millis()를 사용하여 시간 간격을 구하고 일정 시각이 되면 각 기능의 함수를 실행한다.

2) SimpleTimer 라이브러리 타이머

최소한의 기능만을 가지고 있는 단순한 타이머 라이브러리다. 동작은 타이머 객체를 생성하고 setInterval()을 사용하여 콜백 함수를 등록하고 이벤트 시간을 설정한다. loop()에서 타이머 객체의 run()을 호출하여 시간을 계산하면 이벤트 시간마다 등록된 콜백 함수가 호출된다.

3) Timer 라이브러리 타이머

SimpleTimer보다 많은 기능을 가지고 있는 타이머로서 디지털 포트에 신호를 보내는 타이머도 생성할 수 있다. 동작은 타이머 객체를 생성하고 every()를 사용하여 콜백 함수를 등록하고 이벤트 시간을 설정한다. loop()에서 타이머 객체의 update()를 호출하여 시간을 계산하면 이벤트 시간마다 등록된 콜백 함수가 호출된다.

Ⅳ. 실험 및 결과

1. 실험 목적과 환경

두 가지 상황에서 실험을 진행하였으며 첫 번째 상황은 다음과 같은 세 가지 멀티태스킹 방법을 선택하여 조건에 따라 기능을 수행하고 수행 시간을 비교 분석한다. 세 가지 멀티태스킹 방법은 기능을 수행하는 함수를 millis() 타이밍에 따라 직접 함수를 호출하는 방법과 SimpleTimer 라이브러리를 사용하여 콜백으로 기능을 호출하는 방법, 마지막으로 Timer 라이브러리를 사용하여 콜백으로 기능을 호출하는 방법을 사용한다. 두 번째 상황은 빠른 타이밍에 따라 콜백 이벤트를 발생하여 실행시간지연에 따른 콜백 시간을 측정하고 분석한다.

2. 일정한 콜백 이벤트 발생(2초)에 따른 10개의 기능 함수 실행 시간 비교

콜백 이벤트 발생 시간을 2초로 설정하고 세 가지 멀티태스킹 방법에 따라 13번 포트에 digitalWrite()를 7000번 호출하는 기능의 함수 10개를 1번부터 10번까지 만들어 호출되도록 구성하였다.

표 2. 2초의 콜백 이벤트 발생에 따른 모든 함수 실행 시간 비교(ms)

Table 2. Comparison of all function execution times according to 2 seconds callback even

Function Number	1 Function call	2 SimpleTimer	3 Timer
f1	2022	2023	2027
f2	2045	2045	2055
f3	2067	2068	2081
f4	2091	2091	2109
f5	2114	2114	2137
f6	2136	2136	2164
f7	2160	2160	2191
f8	2182	2183	2219
f9	2206	2205	2246
f10	2228	2229	2274

표 2에서 확인할 수 있듯이 콜백 함수 하나가 호출될 때 마다 22msec 정도의 시간지연이 발생하였으며 Timer 라이브러리는 27msec 정도의 시간지연이 발생하였다. 다른 방법보다 Timer 라이브러리의 콜백 시간지연이 더 발생한 원인은 성능 측정결과 Timer 라이브러리는 내부에서 디지털 핀을 직접 제어할 수 있는 기능이 지원되는데 이 디지털 핀 제어부분에서 다른 라이브러리에 비해 시간지연 더 발생하였다.

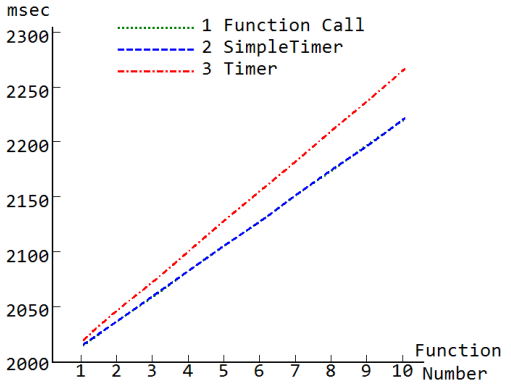

그림 1. 2초의 콜백 이벤트 발생에 따른 모든 함수 실행 시간 비교(ms)

Fig. 1. Comparison of all function execution times according to 2 seconds callback event

3. 서로 다른 이벤트 시간에 따른 8 번째 콜백 함수의 실행 시간 측정

Timer 라이브러리를 사용하여 타이밍이 정밀한(빠른) 상태의 콜백에서 호출 시간을 측정하기 위해 10개의 콜

백 함수 중 6, 7, 8번 콜백 함수는 0.1초마다 콜백되도록 설정하고 나머지 콜백 함수는 1초마다 콜백되도록 설정하였다. 0.1초의 빠른 상태의 콜백 호출 시간으로 설정된 6, 7, 8번 콜백 함수가 호출될 때는 세 함수의 호출시간만큼 시간지연 오차가 발생하여 0.1초마다 콜백이 정상적으로 호출되지 않는 실행 오류가 발생되었다.

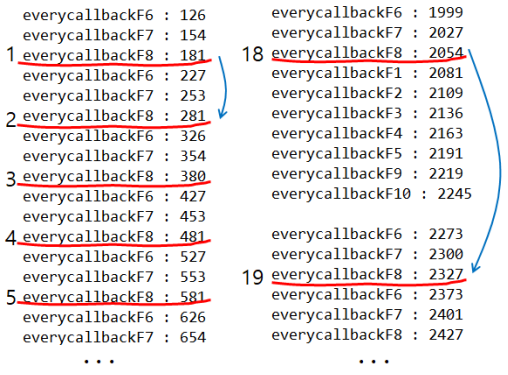

그림 2. 콜백 함수 8번의 실행 지연을 확인하기 위한 시리얼 모니터링(ms)

Fig. 2. Serial monitoring to check the time delay of callback function 8

10개의 모든 콜백 함수가 호출되는 시점에서는 0.1초의 빠른 콜백은 정상적인 콜백이 호출될 수 없을 정도로 시간지연이 발생하여 그림 3의 화살표처럼 실행 오류를 확인할 수 있다.

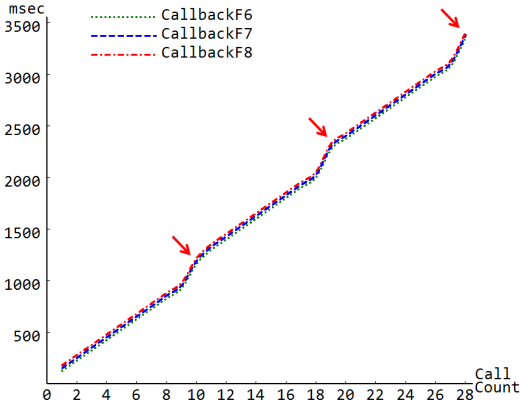

그림 3. 콜백 함수 6, 7, 8번의 실행 오류 측정(ms)

Fig. 3. Measuring Execution Errors in Callback Functions 6, 7, and 8

V. 결론 및 향후 계획

본 논문에서는 사물인터넷(Internet of Things) 디바이스로 많이 사용되는 오픈하드웨어 아두이노의 멀티태스킹을 위한 세 가지 방법의 시간지연을 실험하고 Timer 라이브러리 사용에서 여러 가지 크기의 작업들이 동시에 수행될 때 작은 크기의 작업이 시간지연으로 어떻게 실행되는지 확인하였다.

일정한 크기의 여러 작업을 수행할 때 millis()를 사용한 타이밍 함수 호출 방법과 SimpleTimer 라이브러리를 사용한 방법은 시간지연이 비슷하게 발생되었고 Timer 라이브러리를 사용한 방법은 더 많은 시간지연이 발생되었다. 또 Timer 라이브러리를 사용할 때 시간지연보다 작은 크기의 작업은 일정한 간격으로 정확한 타이밍에 호출되지 않는 실행 오류가 발생되는 것을 확인하였다.

향후 계획에서는 RTOS로 동작하는 사물인터넷 오픈하드웨어의 멀티태스킹 작업들은 어떤 시간지연을 가지며 동작하는지 실험하고 분석하고자 한다.

References

[1] Ho-Tae Lee, "Analysis of Security Technology for Internet of things", The Journal of The Institute of Internet, Broadcasting and Communication(IIBC) Vol. 17, No. 4, pp. 43-48, Aug 2017.

[2] Chang-Gyu Seong, Keel-Soo Rhyu, "Internet of things application service system with open source hardware", Journal of the Korean Society of Marine Engineering 40(6), pp. 542-547, Jul 2016.

[3] Yangja Jang, "IoT Platform Technology Review", Communications of the Korean Institute of Information Scientists and Engineers 32(6), pp. 19-24, Jun 2014.

[4] Internet of Things: Wireless Sensor Networks White Paper(IEC, 2014).

[5] Sung-Yoon Chae, Jinhee Park, "A Design and Implementation of Testing and Management System for IoT Sensors", The Journal of The Institute of Internet, Broadcasting and

Communication(IIBC), Vol. 16, No. 5, pp. 151-156, Oct 2016.

[6] Galadima and A. Adamu, "Arduino as a learning tool", 2014 11th International Conf. on Electronics, Computer and Computation, pp. 1-4, Sep. 2014.

[7] Y. Chemin, N. Sanjaya, P. K. N. C. Liyanage, "An Open Source Hardware & Software online raingauge for real-time monitoring of rainwater harvesting in Sri Lanka." Symposium on Mainstreaming Rainwater Harvesting as a Water Supply Option, 2014.

[8] Yumi Oh, Sungwon Lee, "IoT and Open Source Development Platform" Communications of the Korean Institute of Information Scientists and Engineers 32(6), pp. 25-30, Jun 2014.

[9] S. H. Kim, "Internet Of Things Technology", The Institute of Electronics and Information Engineers, Vol. 43, No. 3, pp. 67-71, 2016.

[10] OSHA(Open Source Hardware Association), https://www.oshwa.org/

[11] Arduino Playground. http://playground.arduino.cc/

[12] Wikipedia, https://www.wikipedia.org/

저자 소개

공 동 환(정회원)

- 2001년도 조선대학교 대학원 컴퓨터공학 졸업(석사)
- 2018년도 한세대학교 대학원 IT융합 졸업(박사)
- 2001년도~현재 비트컴퓨터 비트교육센터 강사
- 2017년도~현재 한양사이버대학 기계자동차공학부 겸임교수

<주관심분야> : IoT, 인공지능

신 승 중(종신회원, 교신저자)

- 1988년도 세종대학교 대학원 경영학과 졸업(석사)
- 1994년도 건국대학교 대학원 전자계산학과 졸업(석사)
- 1999년도 국민대학교 대학원 정보관리학과 졸업(박사)
- 1988~1995 중경공업전문대 전자과 겸임교수
- 1995년~2003 중부대학교 정보보호학과 부교수
- 2003~현재 한세대학교 ICT디바이스학과 부교수

<주관심분야> : 정보보호, 이동통신, 통신공학