Journal of KIIT. Vol. 23, No. 2, pp. 77-84, Feb. 28, 2025. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2025.23.2.77

이더넷 기반의 효율적인 DSP 펌웨어 업데이트 방법

이승훈*[1], 김동혁*[2], 최준영**

Ethernet-based Efficient Method for Updating DSP Firmware

Seung-Hun Lee*[1], Dong-Hyuk Kim*[2], and Joon-Young Choi**

요 약

본 논문에서는 DSP(Digital Signal Processor)의 펌웨어 업데이트 과정에서 기존 통합개발환경 사용의 복잡성과 비효율성을 개선하기 위해 이더넷(Ethernet) 통신 기반의 업데이트 방법을 제안한다. 제안된 방식은 DSP가 PC로부터 이더넷 프레임 형태의 특정 Key 값을 수신하면, 인터럽트를 기반으로 펌웨어 업데이트 함수를 자동 실행하도록 설계되었다. PC는 펌웨어 바이너리 파일을 일정 바이트 단위로 DSP에 전송하며, DSP는 이를 분석하여 데이터 블록 크기, 플래시 메모리 목적지 주소, 데이터 값을 식별하고 플래시 메모리에 기록한다. 업데이트 완료 후 DSP는 자동으로 재부팅되어 새 펌웨어를 실행한다. TI TMDSCNCD28388D 제어 카드를 이용해 제안된 방법을 구현하고, 그 성능과 안정성을 검증함으로써 DSP 펌웨어 업데이트의 효율적이고 간편한 대안을 제시한다.

Abstract

In this paper, we propose an Ethernet-based firmware update method for Digital Signal Processors(DSPs) to address the complexity and inefficiency of traditional development environment-based update processes. The proposed method is designed such that when the DSP receives a specific key value in the form of an Ethernet frame from a PC, it automatically triggers a firmware update function based on an interrupt. The PC transmits the firmware binary file in fixed-size byte segments, and the DSP analyzes the received data to identify the size of the data block, the destination address in flash memory, and the data values, which are then written to the flash memory. Once the update is completed, the DSP automatically reboots to execute the new firmware. Using the TI TMDSCNCD28388D control card, the proposed method was implemented and validated, demonstrating an efficient and straightforward alternative for DSP firmware updates.

Keywords

DSP, Ethernet, firmware update, flash memory

* 부산대학교 전기전자공학과
- ORCID[1]: http://orcid.org/0009-0009-4304-8403
- ORCID[2]: http://orcid.org/0009-0003-5908-7614
** 부산대학교 전기전자공학부 교수(교신저자)
- ORCID: http://orcid.org/0000-0002-5160-3739

· Received: Jan. 03, 2025, Revised: Jan. 21, 2025, Accepted: Jan. 24, 2025
· Corresponding Author: Joon-Young Choi
Dept. of Electrical and Electronic Engineering, Pusan National University, 2, Busandaehak-ro 63beon-gil, Geumjeong-gu, Busan, 46241, Korea
Tel.: +82-51-510-2490, Email: jyc@pusan.ac.kr

Ⅰ. 서 론

DSP(Digital Signal Processor)는 아날로그 신호를 디지털 신호로 변환하여 고속 처리와 연산을 수행하는 프로세서로, 초기에는 간단한 신호 처리, 영상 및 음향 처리, 모터 제어 등에 사용되었다. 당시 DSP는 연산 기능만 수행했기 때문에 외부 메모리와 주변 장치가 필요했으나, 최근에는 주요 주변 장치가 내장된 DSP가 출시되어 산업, 가전, 인공지능, 전기자동차 등 다양한 분야에서 활용되고 있다. 이와 함께 DSP의 수요는 지속적으로 증가하고 있다[1]-[3].

펌웨어는 하드웨어를 구동 및 제어하기 위한 저수준 소프트웨어로, 초기에는 ROM에 저장되었으나, 수정과 기능 추가가 어렵다는 단점이 있었다. 이를 해결하기 위해 PROM(Programmable ROM) 및 플래시 메모리에 펌웨어를 기록하여 사용자가 필요 시 업데이트를 통해 새로운 기능을 추가할 수 있게 되었다. 특히 DSP 기반 시스템에서는 잘못된 기능을 수정하거나 새로운 기능을 추가할 수 있는 펌웨어 업데이트가 필수적이며, 이를 통해 시스템의 유연성과 안정성을 확보할 수 있다[4].

최근에는 무선통신 기반 펌웨어 업데이트(FOTA, Firmware Over The Air) 방식이 DSP에도 도입되고 있다. 이 방식은 무선 통신을 이용하여 펌웨어를 업데이트할 수 있어 편리하지만, 네트워크 구성의 복잡성과 보안 문제, 데이터 탈취 위험 등 여러 한계점이 존재한다[5][6]. 또한 USB 플래시 드라이브를 이용한 업데이트 방식도 개발되어 사용되고 있으나, 사용자가 DSP에 직접 접근해야 하는 물리적 제약이 있다[7]. 또한 이더넷(Ethernet) 기반 TFTP(Trivial File Transfer Protocol)를 이용하는 펌웨어 업데이트 방식도 사용되고 있으나 이 방식은 임베디드 보드의 부트로더에서 지원을 해야만 가능한 단점이 있다.

이러한 기존 방식들은 운용 난이도, 보안 취약성, 원격 작업의 비효율성 등의 문제를 내포하고 있다.

이러한 문제들을 해결하기 위해 본 논문에서는 이더넷 기반의 원격 DSP 펌웨어 업데이트 방식을 제안한다. 제안 방식은 사용자가 PC에서 생성한 펌웨어 바이너리 파일을 이더넷을 통해 DSP 제어 카드(TI TMDSCNCD28388D)에 전달하여 자동으로 플래시 메모리에 기록하고, 이후 DSP가 재부팅되며 새로운 펌웨어를 실행한다. 제안된 방법은 대상 DSP와 PC가 물리적으로 서로 다른 위치에 설치되어 있거나 사용자가 DSP 기기에 직접 접근하기 어려운 환경이라도 이더넷 케이블을 통한 원거리 펌웨어 업데이트를 가능하도록 한다. 또한, 펌웨어 업데이트 대상 DSP와 PC를 이더넷 케이블로 연결하여 서버 경유 없이 데이터를 직접 전송하므로, 데이터가 유출되거나 변조되는 등의 보안 문제를 해결할 수 있다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 DSP 펌웨어 업데이트 방법의 전체 시스템 구조와 펌웨어 바이너리 파일, PC 및 DSP 제어 카드에서 구현된 프로그램을 설명한다. Ⅲ장에서는 실험환경을 구축하여 구현된 펌웨어 업데이트 프로그램을 실험하고 결과를 분석한다. 마지막 Ⅳ장에서는 결론을 도출한다.

Ⅱ. 설계 및 구현

2.1 전체 시스템 구조

제안된 이더넷 기반 DSP 펌웨어 업데이트 방법을 구현하기 위한 전체 시스템 구조는 그림 1과 같으며 기본적으로 PC와 DSP 기반 제어 카드를 이더넷으로 연결한 구조이다. PC에서는 QT 프레임워크 기반 HMI(Human Machine Interface) 프로그램이 동작하고 있으며 HMI의 일부 기능으로 이더넷을 통한 펌웨어 파일 데이터 전송 기능을 구현한다.

펌웨어 업데이트를 수행하는 DSP 제어 카드는 전력전자 제어용으로 많이 사용되고 있는 TI TMDSCNCD28388D 제어 카드로, TMS320F28388D MCU를 장착하고 있다. 이 MCU는 CM(Connectivity Manager) 코어, 2개의 TMS320C28x DSP 코어, 1024KB의 플래시 메모리, 이더넷을 포함한 다양한 통신 장치를 내장하고 있다[8].

PC에서 동작하는 HMI를 통해 펌웨어 업데이트 기능과 새롭게 생성된 펌웨어 파일을 선택하면 파일 전송 함수가 선택된 펌웨어 파일 데이터를 이더넷 전송 함수를 통해 DSP 제어 카드로 전송한다.

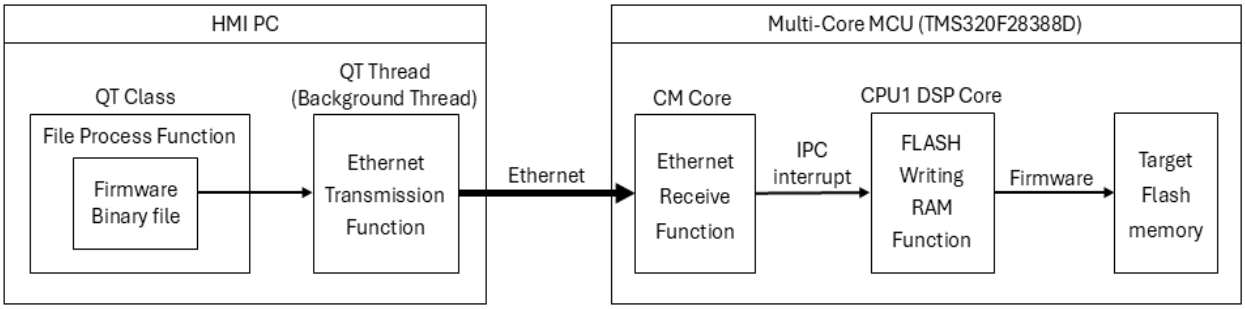


그림 1. 이더넷 기반 DSP 펌웨어 업데이트 시스템 구조
Fig. 1. Ethernet-based DSP firmware update system structure

TMDSCNCD28388D 제어 카드는 PC와 이더넷 케이블로 연결되고, PC HMI에서 전송한 펌웨어 바이너리 파일 데이터를 CM 코어에서 관리하는 이더넷 장치를 통해 수신한다. 이후 CM은 MCU에서 지원하는 프로세서간 통신(IPC, Inter-Processor Communication) 메커니즘을 사용하여 수신된 펌웨어 데이터를 CPU1 (DSP 코어)에 전송하고, 이를 수신한 CPU1은 플래시 쓰기 함수를 RAM 영역에서 실행하여 펌웨어 업데이트를 수행한다.

2.2 PC HMI 프로그램 구현

PC HMI의 이더넷 기반 펌웨어 파일 데이터 전송 기능은 [9]에서 QT 프레임워크 기반으로 개발한 기존 모터 제어 시스템 관리 HMI 프로그램에 추가 구현된 것으로 펌웨어 파일 처리 함수와 이더넷을 통한 파일 데이터 전송 함수로 구성된다.

펌웨어 파일 처리 함수의 경우, QT Class를 사용하여 작성된 그림 2와 같은 별도 HMI 윈도우에서 이용할 수 있으며, 해당 윈도우는 HMI 주 윈도우에 구현된 Firmware Update 버튼을 통해 진입한다. 파일 전송 함수의 세부 기능들은 윈도우 위젯 버튼과 연결되어 있어 관련 버튼들을 클릭함으로써 기능 실행이 가능하다.

먼저 Set Firmware 버튼을 클릭하면 PC 파일 탐색기로 연결되어 사용자가 원하는 펌웨어 바이너리 파일을 선택할 수 있고, 이후 Update Start 버튼으로 이더넷을 통한 파일 데이터 전송 기능을 사용할 수 있다.

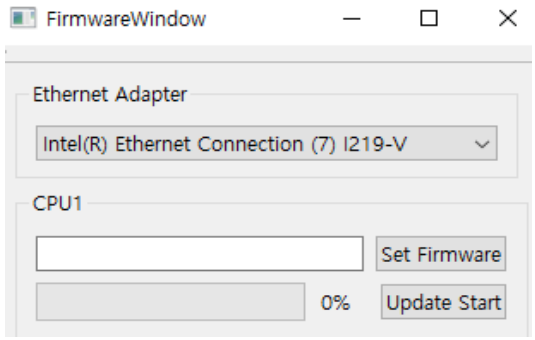


그림 2. 이더넷 기반 펌웨어 업데이트 설정 윈도우
Fig. 2. Ethernet-based firmware update setup window

이더넷을 통한 파일 데이터 전송 기능의 경우, TCP 및 UDP 프로그램과는 달리 QT 프레임워크에서 지원하지 않아 Npcap 라이브러리를 사용해 구현하였다. 또한, QT Thread로 작성되어 고속으로 다량의 이더넷 프레임을 전송하는 과정을 백그라운드에서 처리할 수 있다.

이더넷을 통한 파일 데이터 전송 함수는 앞서 선택한 펌웨어 바이너리 파일의 데이터를 2 바이트씩 읽고 이더넷 프레임으로 DSP에 전송한다. 이때, DSP 프로그램이 수신한 데이터를 플래시에 작성하는 시간을 감안하여 3 ms 지연을 적용하여 이더넷 프레임을 전송한다. 파일 전송에 대한 진행 정도는 Update Start 버튼 좌측의 막대 그래프를 통해 확인할 수 있으며, 데이터 전송 완료 시 화면에 팝업 메시지가 출력된다.

PC HMI의 펌웨어 파일 데이터 전송 과정의 전체적인 자세한 절차는 그림 3과 같다.

2.3 펌웨어 바이너리 파일

TMS320F28388D MCU의 DSP 코어를 위한 펌웨어 바이너리 파일의 구조와 구성요소 분석은 다음과 같다. 표 1은 바이너리 파일을 16진수로 변환한 구조와 2 바이트 단위의 데이터에 관한 설명을 나타낸다. 최초 2 바이트는 Key 값을 나타내며, 해당 값이 수신되었을 경우에만 DSP 제어 카드에서 펌웨어 업데이트 함수가 호출된다. 3~18번째 바이트의 0x00의 값들은 레지스터값을 초기화하거나 부트로더를 위해 사용된다. 19~22번째 바이트 값들은 프로그램이 시작되는 주소(0xAABBCCDD)를 나타낸다. 이후의 바이트 값들은 데이터 블록의 크기, 데이터 블록이 로드되는 목적지 주소, 주소에 작성될 펌웨어 데이터 값의 순서로 반복된다. 이러한 반복 과정이 끝나면 마지막 n~n+1번째 바이트 값 0x00이 저장되고 펌웨어 파일의 끝을 나타낸다 [10].

표 1. 펌웨어 바이너리 파일 구조 및 설명
Table 1. Firmware binary file structure and description

Bytes	First byte	Second byte	Description
1 - 2	AA	08	Key value
3 - 4	00	00	reserved
5 - 6	00	00	reserved
7 - 8	00	00	reserved
9 - 10	00	00	reserved
11 - 12	00	00	reserved
13 - 14	00	00	reserved
15 - 16	00	00	reserved
17 - 18	00	00	reserved
19 - 20	BB	AA	Entry point
21 - 22	DD	CC	Entry point
23 - 24	NN	MM	Block size of first block
25 - 26	FF	EE	First block destination address
27 - 28	HH	GG	First block destination address
29 - 30	BB	AA	First word of first block
…	…	…	…
			Data
			…
…	BB	AA	Last word of first block
…	…	…	…
n - n+1	00	00	indicates end of the file

2.4 DSP 프로그램 구현

DSP에서 동작하는 펌웨어 업데이트 프로그램은 TI에서 제공하는 CCS 통합개발환경에서 작성되었으며, 이 프로그램은 TMS320F28388D MCU에 내장된 2개의 DSP 코어 중 CPU1에 대한 펌웨어 업데이트를 처리한다.

우선, PC HMI 데이터 전송 함수가 전송한 이더넷 프레임은 CM 코어의 이더넷 데이터 수신 함수를 통해 MCU의 내부 메모리에 최초 저장된다. CM 코어는 수신한 데이터를 IPC 인터럽트를 통해 CPU1에 전송하며, 이 과정에서 CPU1이 Key 값을 인식하면 인터럽트 기능을 비활성화하고 플래시 쓰기 함수를 호출한다. 이때, 플래시에서 구동되는 함수가 동일 플래시를 재귀적으로 수정하는 것이 불가능하기 때문에 플래시 쓰기 함수는 RAM function 영역에 정의되어 실행된다.

플래시 쓰기 함수는 쓰기 동작 전에 대상 플래시 영역을 초기화한 후 CM 코어의 IPC 인터럽트로부터 수신한 이더넷 프레임 데이터를 분석하여 펌웨어 데이터 블록 크기, 플래시 목적지 주소, 목적지 주소에 작성할 데이터 블록의 값을 인식하고 관련 변수에 저장한다. 저장된 데이터 블록 값들은 16 바이트 단위로 플래시 메모리의 목적지 주소에 기록되며, 기록이 완료되면 워치독 타이머를 이용한 리셋 함수를 통해 소프트웨어를 초기화 및 재부팅하여 업데이트된 펌웨어를 구동한다. 이때 PC로부터 받은 데이터가 왜곡되거나 누락되어 잘못된 데이터를 잘못된 플래시 주소에 작성하는 시도가 발생할 경우 플래시 쓰기 함수에서 제공하는 ECC (Error Correct Code)가 이를 감지하여 플래시 쓰기 과정이 중단된다. 이 경우, DSP 재부팅 및 PC HMI 프로그램 재실행을 통해 펌웨어 업데이트 과정을 다시 진행할 수 있다.

한편 구현된 펌웨어 업데이트 프로그램을 각 코어에서 동작하는 응용프로그램과 결합하면 사용자가 원할 때마다 지속적인 펌웨어 업데이트가 가능하다. MCU의 CPU1 DSP 코어에서 동작하는 펌웨어 업데이트 프로그램의 작동 과정의 자세한 절차는 그림 3과 같다.

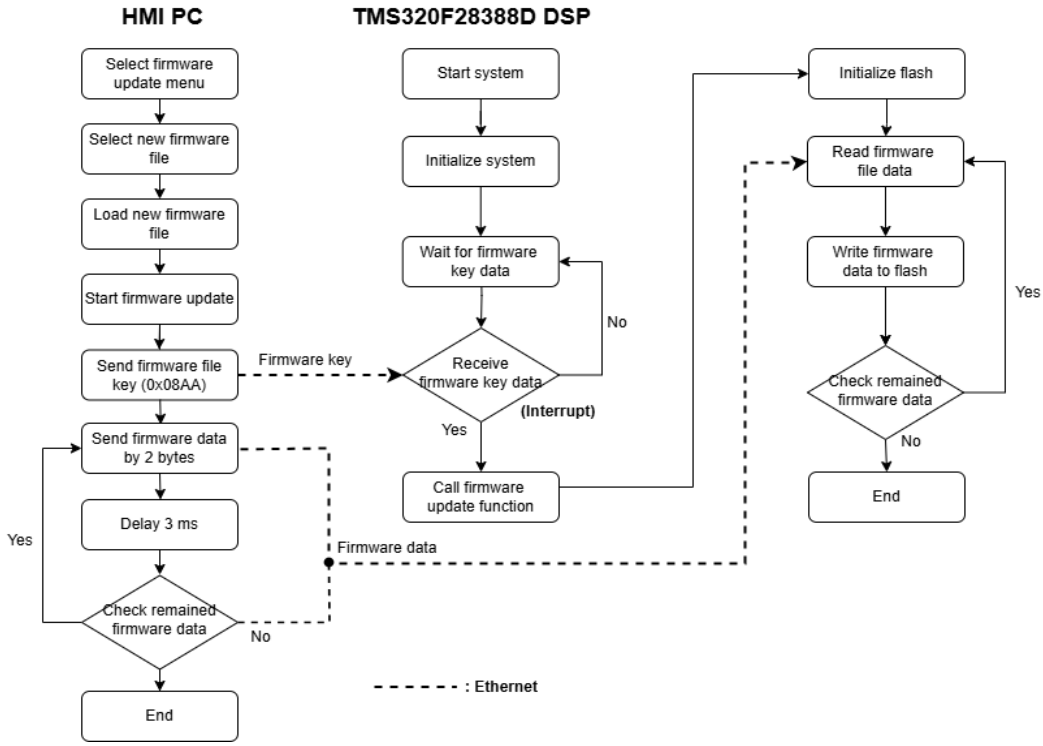


그림 3. 이더넷 기반 DSP 펌웨어 업데이트 방법의 동작 과정
Fig. 3. Operation procedure of Ethernet-based DSP firmware update method

III. 동작 실험 및 결과

3.1 실험환경 구축

제안한 펌웨어 업데이트 방법을 검증하기 위해 그림 4와 같이 HMI PC와 TI TMDSCNCD28388D 제어 카드로 구성되는 실험환경을 구축한다.

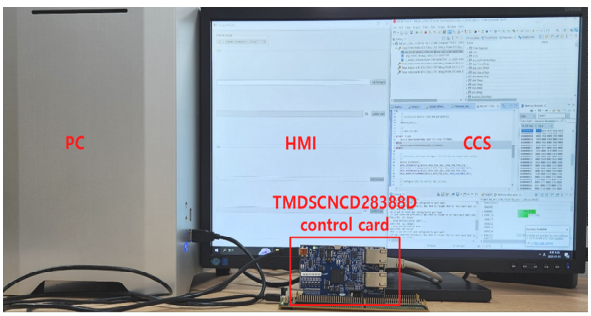


그림 4. 펌웨어 업데이트 검증을 위한 실험환경
Fig. 4. Experimental environment for verifying firmware update

실험에서 사용하는 펌웨어 바이너리 파일로 업데이트 결과를 육안으로 확인할 수 있도록 TI C2000WARE SDK에서 제공하는 TMS320F2838xD CPU1 LED 점멸 예제 펌웨어 바이너리 파일을 사용한다. 이 바이너리 파일은 이미 생성이 되어 새로운 펌웨어 파일로서 PC에 저장되어 있고 언제든지 HMI에서 펌웨어 업데이트를 시작할 수 있는 상태를 유지한다.

한편 TI TMDSCNCD28388D 제어 카드에서는 TI CCS 도구를 사용하여 MCU에서 동작하는 펌웨어 업데이트 프로그램을 로드하고 언제든지 실행이 가능한 상태를 유지한다.

3.2 펌웨어 바이너리 파일

실험에서 사용하는 TMS320F2838xD CPU1 LED 점멸 펌웨어 업데이트 바이너리 파일의 크기는 3,300 바이트이며 16진수로 변환한 데이터 값의 처음 일부분은 그림 5와 같다. CPU1의 RAM에서 동작하는 플래시 메모리 쓰기 함수는 그림 5와 같은 펌웨어 파일 데이터 값을 해석하여 플래시 메모리에 새로운 펌웨어 파일 데이터 값을 기록한다.

```
AA 08 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 08 00 00 00 02 00
08 00 00 00 48 00 CE 2F
14 00 08 00 00 80 00 00 1B 00 01 00
00 00 02 00 BD 2F 08 00 10 00 F0 FF
00 00 41 2A 08 00 C6 2F 08 00 00 00
```

그림 5. 16진수로 변환한 펌웨어 업데이트 파일의 처음 일부분

Fig. 5. First part of the firmware update file converted to hexadecimal

표 1과 같이 Key 값인 0x08AA와 그 뒤에 연속되는 16개의 예약어 0x00을 확인할 수 있다. 프로그램이 시작되는 플래시 주소는 0x00080000으로 인식된다. 첫 목적지 주소 0x00080000에 0x0002 word 크기즉 4 byte 크기의 데이터 블록의 값을 쓰게 되며, 작성할 데이터 블록 값 0x0048 과 0x2FCE를 확인 가능하다. 두 번째 목적지 주소는 0x00088000이고 0x14 word 크기의 데이터 블록의 값을 쓰게 되며 해당 크기만큼의 잇따른 데이터 블록 값들을 확인할 수 있다.

이러한 방식으로 목적지 주소에 데이터 블록의 값을 모두 쓴 이후에 그림 6과 같이 펌웨어 업데이트 파일의 마지막 데이터값으로 0x0000을 읽고 데이터 블록의 크기가 0임을 확인하여 읽고 쓰는 과정을 마무리하고 소프트웨어 초기화 및 재부팅을 실행한다.

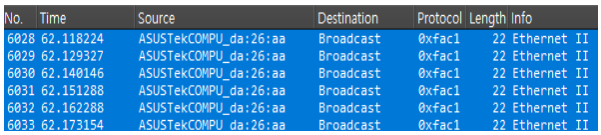
```
69 FF 84 FE 06 00 25 76 00 6F 25 76
00 6F 01 9A 06 00 06 00
00 00
```

그림 6. 16진수로 변환한 펌웨어 업데이트 파일의 마지막 부분

Fig. 6. Last part of firmware update file converted to hexadecimal

3.3 실험 결과

DSP 펌웨어 업데이트 프로그램이 PC HMI 프로그램으로부터 정상적으로 Key 값을 수신한 후, 이더넷 프레임을 수신하는 과정은 그림 7, 8과 같이 각각 WireShark와 HMI 프로그램을 통해 확인할 수 있다.



No.	Time	Source	Destination	Protocol	Length	Info
6028	62.118224	ASUSTekCOMPU_da:26:aa	Broadcast	0xfac1	22	Ethernet II
6029	62.129327	ASUSTekCOMPU_da:26:aa	Broadcast	0xfac1	22	Ethernet II
6030	62.140146	ASUSTekCOMPU_da:26:aa	Broadcast	0xfac1	22	Ethernet II
6031	62.151288	ASUSTekCOMPU_da:26:aa	Broadcast	0xfac1	22	Ethernet II
6032	62.162288	ASUSTekCOMPU_da:26:aa	Broadcast	0xfac1	22	Ethernet II
6033	62.173154	ASUSTekCOMPU_da:26:aa	Broadcast	0xfac1	22	Ethernet II

그림 7. WireShark에서 확인된 PC에서 DSP 제어 카드로 전송되는 이더넷 프레임

Fig. 7. Ethernet frames from PC to DSP control card confirmed in WireShark

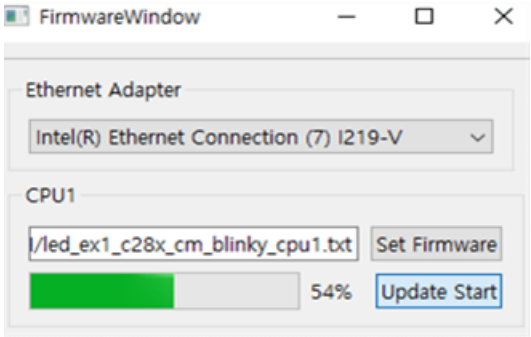


그림 8. HMI에서 확인된 PC에서 DSP 제어 카드로 전송되는 이더넷 프레임

Fig. 8. Ethernet frames from PC to DSP control card confirmed in HMI

HMI PC로 부터 이더넷 프레임 형태로 수신한 펌웨어 바이너리 파일 데이터를 이용하여 펌웨어 업데이트가 정상적으로 완료된 후 TMS320F28388D MCU를 초기화하고 재부팅을 수행하면 그림 9와 같이 LED 점멸을 확인할 수 있다.

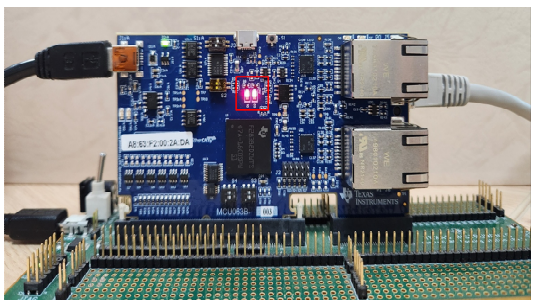


그림 9. 펌웨어 업데이트 결과 (LED 점멸)

Fig. 9. Firmware update result (LED flashing)

IV. 결론 및 향후 연구 내용

본 논문에서는 DSP 펌웨어 업데이트 과정의 복잡성과 비효율성을 해결하기 위해 이더넷 기반의 원격 펌웨어 업데이트 방식을 제안하고, 이를 TI

TMDSCNCD28388D 제어 카드를 통해 구현 및 검증하였다. 제안된 방법은 기존 방법과 비교했을 때 PC HMI 프로그램과 이더넷 케이블 연결이 필요한 단점도 존재하지만 기존 통합개발환경에 의존하던 방식에서 벗어나 이더넷 통신을 활용하여 PC와 DSP 간의 데이터 전송과 플래시 메모리 기록 과정을 자동화하였다. 이를 통해 펌웨어 업데이트 작업의 간소화와 효율화를 동시에 달성할 수 있었다.

제안된 방법은 TI TMDSCNCD28388D 제어 카드에서의 성능 검증을 통해 안정성과 실용성이 입증되었으며, 다양한 DSP 기반 시스템에서 쉽게 적용 가능한 유연한 대안을 제공한다. 특히, 원격 업데이트 기능은 산업용 제어 시스템, IoT 디바이스, 임베디드 시스템과 같은 응용 분야에서 높은 활용 가능성을 보여준다.

현재는 멀티코어 MCU의 단일 DSP 코어에 대한 펌웨어 업데이트 방법을 구현하였으나 향후 연구에서는 멀티 DSP 및 ARM 코어로 구성된 MCU의 모든 코어에 대한 펌웨어를 동시에 연속적으로 업데이트하는 방법으로 확장할 계획이다.

References

[1] G. B. Lee, "DSP TMS320F28335 technology for power electronic system control", Munundang, Oct. 2017.

[2] G. U. Lee, "Power Electronics and DSP", KIPE Magazine, Vol. 25, No. 3, pp. 62-65, Jun. 2020.

[3] H. W. Kim, K. G. Nam, and J. Y. Choi, "Analog-Digital Signal Processing System Based on TMS320F28377D", IEMEK J. Embed. Sys. Appl., Vol. 14, No. 1, pp. 33-41, Feb. 2019. https://doi.org/10.14372/IEMEK.2019.14.1.33.

[4] S. H. Ahn and S. Malik, "Automated firmware testing using firmware-hardware interaction", Proceedings of International Conference on Hardware/Software Codesign and System Synthesis, New Delhi India, pp. 1-10, Oct. 2014. https://doi.org/10.1145/2656075.2656080.

[5] N. Pekez, J. Kovačević, and N. Kaprocki, "Firmware Update Procedure for Audio Systems based on CS4953xx DSP family", Proceedings of International Conference on Smart Systems and Technologies, Osijek, Croatia, pp. 29-34, Oct. 2018. https://doi.org/10.1109/SST44635.2018.

[6] S. E. Lee, J. M. Lee, and I. G. Lee, "Secure FOTA Update Mechanism for Lightweight IoT", Annual Symposium of KIPS 2024, Vol. 31, No. 1, pp. 228-229, May 2024. https://doi.org/10.3745/PKIPS.y2024m05a.288.

[7] J. S. Kim and J. Y. Choi, "DSP Firmware Update Using USB Flash Drive", IEMEK J. Embed. Sys. Appl., Vol. 18, No. 1, pp. 25-30, Feb. 2023. https://doi.org/10.14372/IEMEK.2023.18.1.25.

[8] TMS320F2838x Real-Time Microcontrollers With Connectivity Manager TRM (Rev. F), https://www.ti.com/lit/ug/spruii0f/spruii0f.pdf. [accessed: Jan. 02, 2025]

[9] Y. W. Sung, C. Y. Yoon, and J. Y. Choi, "Development of Motor Drive Monitoring System based on Multi-Core DSP", The Journal of Korean Institute of Information Technology, Vol. 21, No. 2, pp. 69-76, Feb. 2023. https://doi.org/10.14801/jkiit.2023.21.2.69.

[10] TMS320C28x Assembly Language Tools v21.6.0.LTS User's Guide (Rev. W), https://www.ti.com/lit/ug/spru513w/spru513w.pdf. [acessed: Jan. 02, 2025]

저자소개

이 승 훈 (Seung-Hun Lee)

2018년 3월 ~ 현재 : 부산대학교 전기전자공학부 전자공학전공 학사과정

관심분야 : 임베디드 시스템, 제어 시스템

김 동 혁 (Dong-Hyuk Kim)

2023년 8월 : 부산대학교 전기전자공학부 전자공학전공(공학사)

2023년 9월 ~ 현재 : 부산대학교 전기전자공학과 석사과정

관심분야 : 임베디드 시스템, 제어 시스템

최 준 영 (Joon-Young Choi)

1994년 2월 : 포항공과대학교 전기전자공학과(공학사)

1996년 2월 : 포항공과대학교 전자전기공학과(공학석사)

2002년 : 포항공과대학교 전자전기공학과(공학박사)

2005년 3월 ~ 현재 : 부산대학교 전자공학과 교수

관심분야 : 임베디드 시스템, 제어 시스템