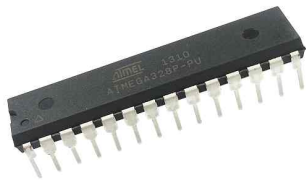


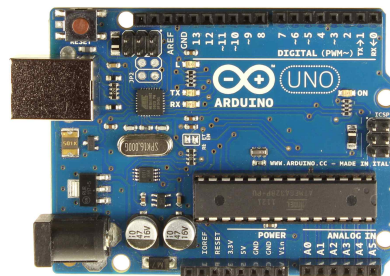
Chapter 98. 마이크로컨트롤러 이야기

1. 마이크로컨트롤러란 무엇인가?

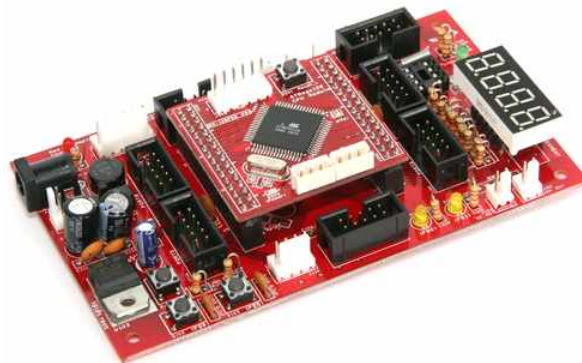
이 장을 읽어보고 있다면 마이크로컨트롤러에 관심을 가지고 있는 독자임에 틀림없으리라 생각한다. 마이크로컨트롤러가 무엇인지 이야기하기 전에 마이크로컨트롤러가 무엇이라고 생각 하는지 스스로에게 질문을 던져보자. 무엇이 가장 먼저 머리에 떠오르는가? 실험 시간에 사용 하던 보드가 생각날 수도 있고, 보드 위의 자리 잡고 있는 칩이 떠오를 수도 있다. 또는 최근 오픈 소스 하드웨어로 주목을 받고 있는 아두이노가 생각날 지도 모르겠다.



(a) 마이크로컨트롤러 칩 - ATmega328



(b) 아두이노 우노¹⁾



(c) 마이크로컨트롤러를 이용한 학습 보드²⁾

그림 1. 마이크로컨트롤러

마이크로컨트롤러를 한 마디로 표현 하자면 ‘칩 위의 컴퓨터’라고 말할 수 있다. 그림 1에서

1) 이미지 출처: <http://arduino.cc/en/Main/ArduinoBoardUno>

2) 이미지 출처: <http://www.newtc.co.kr>

첫 번째 그림이 바로 마이크로컨트롤러이다. 마이크로컨트롤러를 어렵다고 생각하는 이유 중 하나는 마이크로컨트롤러를 처음 접하는 사람들에게 주어지는 부품이 빼곡하게 늘어서 있는 보드가 큰 이유 중 하나일 것이다. 수십 개의 부품이 늘어서 있고 수십 개의 연결 핀이 있는 보드를 처음 받게 되면 무엇부터 어떻게 시작해야 할지 모르는 것이 당연하다. 하지만 실제 마이크로컨트롤러를 사용하기 위해 필요한 부품들은 그리 많지 않다. 가장 간단하게는 마이크로컨트롤러에 전원만 연결하면 동작한다. ATmega328의 경우 5V 전원만 연결해 주면 마이크로컨트롤러에 설치된 프로그램이 실행된다. 간단하지 않은가?

마이크로컨트롤러는 ‘하나의 칩으로 구현한 컴퓨터’에 해당한다. 그다지 컴퓨터와 닮아 보이지 않겠지만 컴퓨터와 동일한 구성으로 컴퓨터와 동일한 동작을 한다. 먼저 친숙한 컴퓨터부터 살펴보자. 어디서나 쉽게 컴퓨터를 찾아볼 수 있는 이유도 있겠지만 컴퓨터에는 컴퓨터와 대화할 수 있도록 키보드, 마우스, 모니터 등이 연결되어 있고 멋진 케이스 속에 들어가 있어 생소한 전자 부품들이 눈에 띄지 않으므로 편안하게 느껴질 수도 있다. 하지만 컴퓨터에서 키보드와 마우스 그리고 모니터를 떼어 내 보자. 컴퓨터로 무엇을 할 수 있을까? 전원을 넣으면 컴퓨터가 켜지는 것은 전원 표시등으로 유추해 볼 수 있지만 컴퓨터 내부에서 무슨 일이 벌어지고 있는지는 짐작하기 어렵다. 멋진 케이스도 벗겨 보자. 컴퓨터 내부에는 어떤 것들이 들어 있나? 메인 보드가 보일 것이고 그 위에 장착된 CPU와 메모리가 눈에 띈다. CPU위에 올라와 있는 선풍기로 사용해도 무방할 정도의 팬이 보이는가? 한쪽에는 하드디스크와 DVD 드라이브가 고정되어 있다. 이외에 전원 공급 장치, 비디오 카드 등이 눈에 들어온다. 어쩌면 이 모든 것들을 확인하기 위해서는 케이스 내부의 선들을 먼저 치워야 할지도 모른다.



그림 2. 컴퓨터 내부

마이크로컨트롤러에 멋진 케이스는 없다. 원한다면 멋진 케이스를 만들어 넣어줄 수도 있겠지만 마이크로컨트롤러는 대부분 다른 전자 제품의 일부로 포함되기 때문에 별도로 케이스를 만드는 경우는 흔하지 않다. 마이크로컨트롤러에는 모니터도, 마우스도, 키보드도 없으므로 내부에서 무슨 일이 벌어지고 있는지 알 수 없다. 앞에서 똑같은 이야기를 한 것이 기억나지 않는가? 마이크로컨트롤러는 주변 장치를 떼어내고 케이스를 벗겨버린 컴퓨터와 똑같은 상황에 놓여 있다. 마이크로컨트롤러는 컴퓨터다. 다만, 주변 장치가 연결되어 있지 않으므로 내부에서 무슨 일이 벌어지고 있는 지 알아보기 위해서는 (키보드, 마우스에 해당하는) 입력장치와 (모니터에 해당하는) 출력장치를 연결해 주어야 한다. 컴퓨터의 경우에는 이미 입출력 장치가 포함된 형태인 반면 마이크로컨트롤러는 직접 입출력 장치를 선택하여 연결해 주어야 하고 필요에 따라서는 새로운 입출력 장치들을 만들어야 하는 경우도 있다. 더 어려운 점은, 컴퓨터의 입출력 장치는 완제품 형태로 판매되며 동일한 인터페이스로 동작하므로 컴퓨터에 연결하는 것만으로 사용하는데 크게 어려움이 없다. 즉, 대부분의 키보드는 USB 포트에 연결하는 것으로 동작하며 대부분의 모니터는 HDMI 포트에 연결하는 것으로 동작한다. 하지만 마이크로컨트롤러의 입출력 장치는 다양한 입출력 연결 방식을 가진다. 간단한 정보를 표시해줄 수 있는 LCD 창외의 경우만 하더라도 종류에 따라 다양한 데이터 입출력 방식을 사용하므로 컴퓨터에 모니터를 연결하는 것과 같이 간단한 동작시킬 수는 없다. 너무 어렵게 들리는가? 하지만 마이크로컨트롤러에서 사용하는 입출력 방식의 종류가 그리 많지 않으므로 익숙해지고 나면 그리 어렵지 않다. 더욱이 마이크로컨트롤러에서 사용하는 대부분의 주변 장치들은 한정된 입출력 방식을 사용하므로 서로 다른 종류의 장치들도 동일한 방식으로 제어가 가능한 경우도 있다. 즉, 컴퓨터에서 LCD와 키보드는 서로 다른 입출력 방식을 사용하지만 마이크로컨트롤러에서는 동일한 방식으로 데이터를 주고받을 수도 있다. 마이크로컨트롤러는 컴퓨터이다. 컴퓨터에 비해 연결 방법이 복잡하고 귀찮아 보이기 는 하지만 입출력 장치를 연결함으로써 마이크로컨트롤러는 컴퓨터와 동일한 동작을 수행할 수 있다.

2. 마이크로프로세서? 마이크로컨트롤러?

마이크로컨트롤러는 주변장치를 제거하고 케이스를 벗겨낸 컴퓨터와 동일하다. 먼저 컴퓨터의 구조를 살펴보자. 컴퓨터는 연산의 핵심이 되는 연산 장치와 연산을 제어하는 제어 장치로 이루어지는 중앙처리장치(Central Processing Unit, CPU), 데이터 입출력을 위한 입출력장치, 데이터 저장을 위한 주기억 장치 및 보조기억 장치 등으로 구성된다.

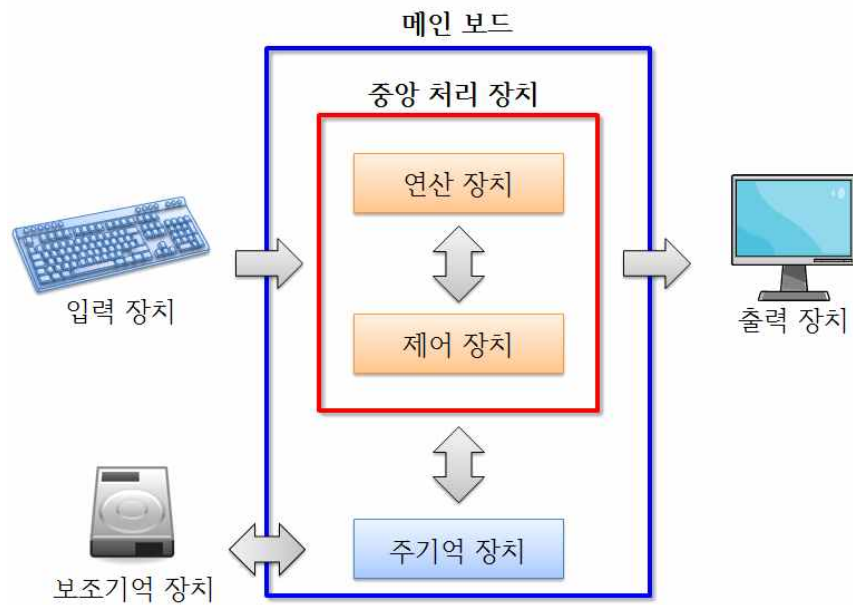


그림 3. 컴퓨터의 구성 요소

집적회로 기술의 발달에 힘입어 컴퓨터의 소형화 및 경량화가 이루어졌고 특히 중앙처리장치는 하나의 칩으로 구현이 가능하게 되었다. 이처럼 컴퓨터의 중앙처리장치를 하나의 IC(Integrated Circuit) 칩으로 집적시켜 만든 반도체 소자를 마이크로프로세서(microprocessor)라고 부른다. 마이크로프로세서는 1969년 Intel에서 발표한 4비트 마이크로프로세서인 4004에서부터 시작되었다. 이후 마이크로프로세서는 비약적인 발전을 거듭하여 4004의 경우 2300여개의 트랜지스터를 사용하여 만들어진 반면 Intel의 최신 마이크로프로세서인 Core i7은 약 22억 개의 트랜지스터가 사용되어 4004와 비교할 때 약 10만 배의 트랜지스터가 사용되고 있다. 현재 사용되고 있는 컴퓨터의 대부분의 중앙처리장치는 마이크로프로세서로 구현되고 있다. 하지만 마이크로프로세서는 중앙처리장치의 여러 형태 중 한 가지로 모든 마이크로프로세서는 중앙처리장치이지만 모든 중앙처리장치가 마이크로프로세서인 것은 아니다.

마이크로프로세서 중 하나의 칩에 중앙처리장치의 기능뿐만 아니라 일정 용량의 메모리와 입출력 인터페이스까지 내장한 것을 마이크로프로세서와 구별하여 마이크로컨트롤러라고 부른다. 마이크로컨트롤러는 그림 3에서 메인 보드에 해당하는 기능과 일부 하드 디스크의 기능을 하나의 칩으로 구현한 것으로 볼 수 있다. 마이크로컨트롤러는 하나의 칩에 컴퓨터가 가져야 할 대부분의 기능을 갖추고 있으므로 단일 칩 마이크로컴퓨터 또는 마이컴이라고도 부른다. 마이크로프로세서가 컴퓨터로 동작하기 위해 더 필요한 것은 무엇일까? (입력 장치인) 키보드와 마우스 그리고 (출력 장치인) 모니터만 있다면 컴퓨터로 동작하기에 아무런 문제가 없다.

하지만 마이크로컨트롤러에 이러한 입출력장치를 연결하여 사용하지는 않는다. 불가능하지는 않지만 굳이 연결하려고 하지 않는 이유는 마이크로컨트롤러는 컴퓨터와는 그 목적이 다르기 때문이다.

‘마이크로컨트롤러’라는 이름은 ‘마이크로 + 컨트롤러’로 이루어져 있다. ‘마이크로’는 작다는 의미로 하나의 칩으로 만들어져 있어 작고 가벼움을 의미한다. ‘컨트롤러’는 제어를 뜻하며 마이크로컨트롤러의 주용도, 제어장치를 만들기 위한 핵심 부품으로 마이크로컨트롤러가 사용됨을 의미한다. 마이크로컨트롤러는 제어장치를 만들기 위한 목적으로 특화된 마이크로프로세서의 한 종류이다.

3. 마이크로컨트롤러는 어디에 사용할 수 있을까?

컴퓨터로 무엇을 할 수 있나? 웹 브라우저를 실행시켜 온라인 뉴스를 읽고, 음악을 듣고, 문서를 작성하고, 가끔은 게임도 할 것이다. 이러한 작업을 위해 필요한 입출력 장치들이 바로 키보드, 마우스, 그리고 모니터이다. 마이크로컨트롤러는 컴퓨터이기는 하지만 흔히 우리가 컴퓨터에서 하는 작업들을 하기 위한 용도로 사용하지는 않는다. 일상적으로 컴퓨터로 처리하는 작업들을 마이크로컨트롤러로 처리하기에는 마이크로컨트롤러가 너무 느리고 용량도 충분하지 않다. 이 책에서 다룰 마이크로컨트롤러인 ATmega328을 이 글을 쓰고 있는 컴퓨터와 비교해 보면 메모리는 약 26만분의 1, 속도는 약 850분의 1에 지나지 않는다.

항목	아두이노 우노	데스크톱 컴퓨터
CPU	ATmega328	Intel Core i7
메모리	32 KByte	8 GByte
클럭	16 MHz	3.4 GHz (Quad Core)

표 1. 아두이노 우노와 데스크톱 컴퓨터 비교

마이크로컨트롤러는 일상생활에서 흔히 사용하는 환경이 아닌 특수한 환경에서 사용할 목적으로 만들어진 작고, 간단한 컴퓨터인 것이다. Intel Core i7과 비교했을 때 ATmega328은 약 100분의 1 가격이면 구입할 수 있다. 현관에 사람이 들어오면 자동으로 불이 켜지는 전등을 만드는데 크고 비싼 컴퓨터를 사용할 필요는 없지 않은가? 작고 값싼 마이크로컨트롤러로도 이 정도의 작업은 충분하다. 마이크로컨트롤러의 성능이 나쁘다고 이야기 하였지만 이는 컴퓨터와 비교하였을 때의 이야기이지 결코 성능이 나쁜 것은 아니다. ATmega328의 최대 클럭 주파수는 20MHz이다. 물론 Intel Core i7과 비교한다면 느린 것이 사실이지만 하드디스크가

대중화되기 시작한 시점에서 소개된, 흔히 ‘286 컴퓨터’라고 불리는 컴퓨터의 80286 CPU 클럭 주파수가 20MHz임을 감안하면 컴퓨터라고 불려도 손색이 없을 정도임을 알 수 있다. ATmega328로도 자동 점등 전등을 만들기에는 충분하다. 오히려 자동 점등 전등을 위해서 ATmega328은 지나치게 성능이 좋으며 원한다면 1,000원 전후의 더 싼 가격의 마이크로컨트롤러로도 가능하다.

마이크로컨트롤러는 컴퓨터이다. 작고 간단하고 저렴한 컴퓨터이긴 하지만 1,000원 짜리 컴퓨터에 키보드와 모니터를 연결하여 슈팅 게임을 하고자 하는 사람을 없으리라 생각한다. 물론 일부 마이크로컨트롤러는 다양한 기능은 물론 GHz의 클럭 주파수를 가진 것들도 존재한다. 이러한 고성능의 마이크로컨트롤러는 여전히 Intel Core i7보다는 저렴하기는 하지만 1,000원에 구입할 수는 없다.

마이크로컨트롤러를 어디에 사용할 수 있을까? ‘작고 간단한 제어장치’를 만들기 위해 사용할 수 있다. 마이크로컨트롤러에 케이스를 만들지 않는 이유는 다른 시스템의 일부로 포함되는 경우가 대부분이기 때문이라고 이야기 하였다. 자동 점등 조명 역시 조명 장치의 일부분으로 마이크로컨트롤러가 들어간다. 이처럼 다른 시스템의 일부로 포함되는 마이크로컨트롤러를 ‘임베디드 (embedded)’되었다고 이야기하며 마이크로컨트롤러는 임베디드 시스템 영역에서 중요한 한 부분을 차지하고 있다. 임베디드 시스템에서 또 다른 한 부분을 차지하는 것은 마이크로프로세서로, 마이크로프로세서로 만들어지는 임베디드 시스템은 크기만 데스크톱 컴퓨터와 다를 뿐 데스크톱과 동일한 컴퓨터에 해당한다. 최근 라즈베리 파이(Raspberry Pi)³⁾를 필두로 주목받고 있는 싱글 보드 컴퓨터들 역시 마이크로프로세서를 사용한 보드로 임베디드 시스템에서의 응용이 점차 늘어나고 있는 추세이다.

마이크로컨트롤러의 또 다른 장점 중 하나는 다양한 제품이 존재한다는 점이다. 마이크로컨트롤러에는 1,000원이면 구입할 수 있는 제품이 있는가 하면 그 수십 배 가격에 달하는 마이크로컨트롤러들도 존재한다. 이처럼 다양한 제품군은 용도와 목적에 맞게 필요로 하는 기능과 성능을 선택할 수 있는 유연성을 제공해준다.

마이크로컨트롤러는 컴퓨터(또는 마이크로프로세서)에 비해 다음과 같은 장점이 있다.

- 제품의 소형화 및 경량화 : 마이크로컨트롤러는 마이크로프로세서를 사용하는 컴퓨터의 메인 보드에 포함된 대부분의 기능을 하나의 칩에 구현하고 있으므로 작고 가벼운 제어 장치를 만들기 위해 사용할 수 있다.
- 저렴한 가격 : 마이크로컨트롤러는 마이크로프로세서에 비해 낮은 집적도를 가지고 설계가 간단하므로 마이크로프로세서에 비해 가격이 저렴하다. 또한 마이크로컨트롤러는 제어 목

3) <http://www.raspberrypi.org>

적에 필요한 대부분의 기능을 포함하고 있어 제어 장치 설계 및 제작 과정이 단순해지므로 개발에 필요한 비용 및 시간을 줄일 수 있어 완성된 제품의 가격 경쟁력을 높일 수 있다.

- 신뢰성 향상 : 마이크로컨트롤러는 제어 장치 구현에 필요한 대부분의 기능을 내장하고 있어 시스템을 구성할 때 필요로 하는 부품의 수가 적으므로 고장이 적고 유지 보수가 용이하다.
- 융통성 : 전통적으로 제어 장치에서 제어를 위해 필요한 기능들을 하드웨어로 구현하는 것과 달리 마이크로프로세서는 제어 기능들을 소프트웨어적으로 처리할 수 있도록 해줌으로써 기능의 변경이나 확장에 보다 유연하게 대응할 수 있다.

하지만 마이크로컨트롤러가 마이크로프로세서에 비해 가지는 단점도 분명 존재한다.

- 처리 능력 : 마이크로컨트롤러는 단순화된 저사양의 마이크로프로세서에 주변 장치를 통합한 형태로 특수 목적으로 만들어졌으므로 일반적으로 처리 능력은 마이크로프로세서에 비해 떨어진다. 따라서 마이크로컨트롤러를 높은 처리 능력을 요구하는 작업에는 적합하지 않다. 빨리 많은 데이터를 처리해야 할 필요가 있다면 마이크로프로세서를 사용하는 것이 바람직하다.
- 범용성 : 마이크로프로세서의 경우 일반적으로 운영체제를 통해 다수의 프로그램들을 설치하고 실행시킬 수 있다. 하지만 마이크로컨트롤러는 특정 작업을 위한 하나의 프로그램만을 설치하고 실행시킬 수 있다.

마이크로컨트롤러는 전용의 간단한 제어장치를 만들기 위해 사용할 수 있다. 1,000원 짜리 마이크로컨트롤러에서 백 만 원 짜리 컴퓨터의 성능을 기대하지만 않는다면 마이크로컨트롤러를 사용할 수 있는 곳은 많으며 실제 마이크로컨트롤러가 사용된 예를 어렵지 않게 찾아볼 수 있다. 앞서도 이야기 하였지만 마이크로컨트롤러는 임베디드 시스템의 일부로 포함되어 있어 눈에 띄지 않을 뿐이다. 표 2는 마이크로컨트롤러가 실생활에서 사용되고 있는 예를 나타낸 것으로 이보다 훨씬 많은 예들도 어렵지 않게 모을 수 있다.

분야	예
의료	의료기 제어, 자동 심박계
교통	신호등 제어, 주차장 관리
감시	출입자/침입자 감시, 산불 감시
가전	에어컨, 세탁기, 전자레인지
음향	CD 플레이어, 전자 타이머
사무	복사기, 무선 전화기
자동차	엔진 제어, 충돌 방지
기타	게임기, 차고 개폐 장치

표 2. 마이크로컨트롤러 사용 예

4. 마이크로컨트롤러를 어떻게 공부해야 할까?

그림 1에서도 보인 것처럼 마이크로컨트롤러는 칩 하나일 뿐이다. 이 책에서 다루게 될 ATmega328 마이크로컨트롤러의 경우 28개의 핀을 가진 약 3.5cm 길이의 칩으로, 사용의 편의를 위해 보드 위에 빼곡히 들여놓은 부가 장치들에 지레 겁을 먹지만 않는다면 마이크로컨트롤러를 공부하는 것은 ‘보기보다’ 어렵지 않다. 인터넷에서 아두이노 우노 보드의 회로도를 찾아보자. 그림 4는 공개된 아두이노 우노 보드의 회로도에서 동작에 필수적인 부분들만 표시한 것으로, 실선으로 표시된 외부 클록을 위한 회로와 전원 안정화를 위한 회로를 제외하면 나머지 회로들은 생략하여도 동작에 지장이 없다, 물론 전원은 5V가 주어진다고 가정한 경우이다. 다른 회로들은 컴퓨터와의 연결을 간편하게 하고 프로그램을 다운로드 할 수 있도록 USB 연결을 지원하는 회로, 어댑터에서 전원을 공급받기 위한 회로 등에 해당한다. 컴퓨터에서 프로그램을 다운로드 하기 위해서는 그림 4에서 점선으로 표시된 부분 역시 필요하며 이는 ATmega328의 핀과 커넥터를 연결하는 것에 지나지 않는다.

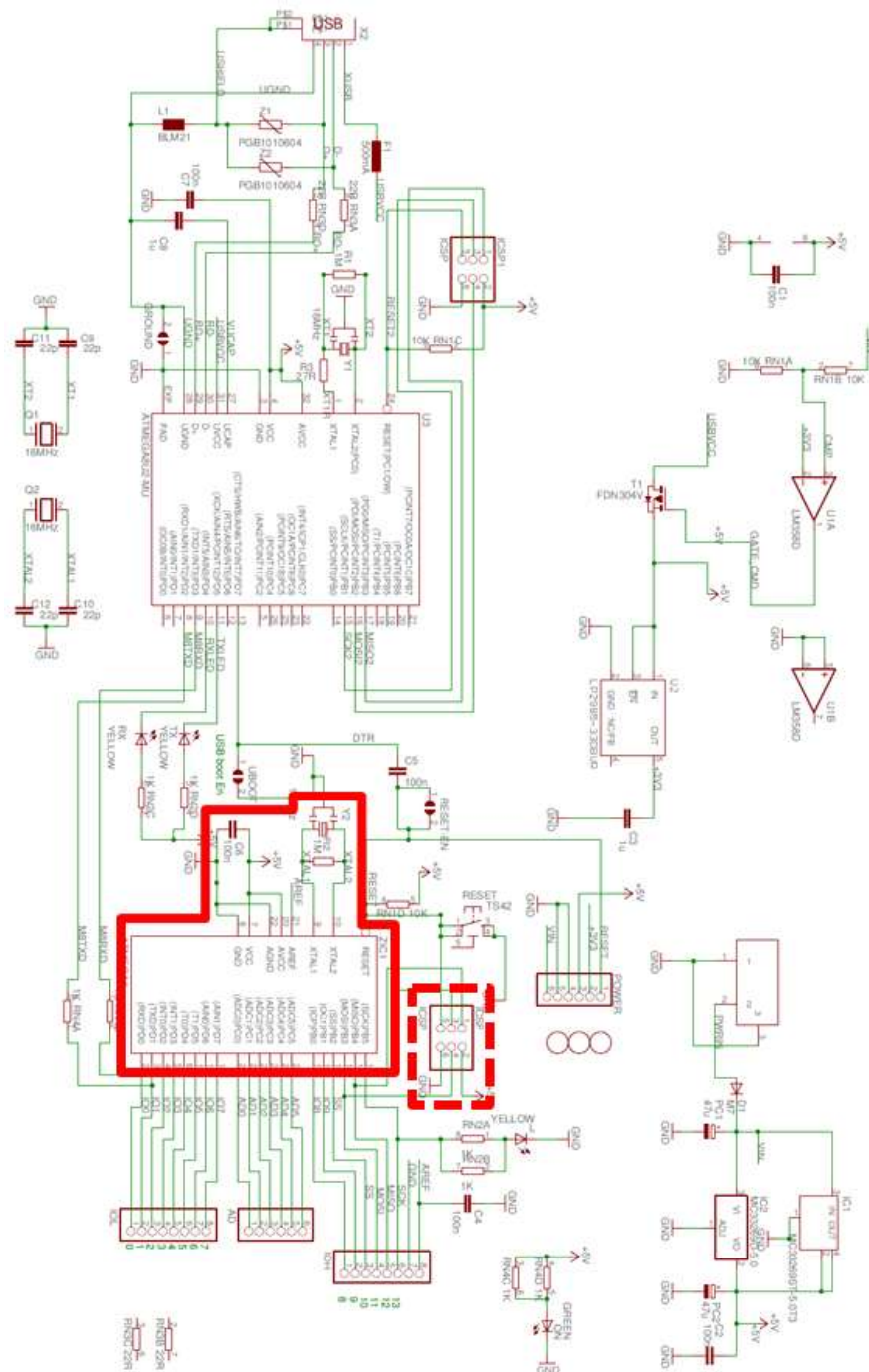


그림 4. 아두이노 우노 회로도

이 책에서 다루고자 하는 ATmega328을 공부하기 위해서는 그림 4에서 표시한 기본 회로들 로도 가능하다. 한 가지 더 필요한 것이 있다면 컴퓨터에서 프로그램 다운로드를 위한 ISP (In System Programming) 장치로, ISP 장치는 아두이노 우노 보드에 포함되지 않므 로 별도로 구입하여야 한다. 아두이노 환경에서 프로그램을 다운로드 하기 위해서는 일반적

으로 마이크로컨트롤러에서는 사용하지 않는 USB-시리얼 변환 장치를 사용한다. 그림 4에서 기본 회로에 포함되지 않은 또 다른 마이크로컨트롤러가 USB-시리얼 변환 장치로 아두이노 우노 보드를 사용한다면 별도로 구입하지 않아도 되겠지만 기본 회로에서 시작하고자 한다면 이 역시도 구입하여야 한다.

기본 회로를 바탕으로 여기에 필요한 입출력 장치를 하나씩 추가해 나가면서 제어장치를 구성해 본다면 보기보다는 마이크로컨트롤러가 그리 어렵지 않다는 것을 알 수 있으리라 생각한다. ATmega328이 마이크로컨트롤러 중에서도 간단한 마이크로컨트롤러인 것은 사실이다. 교육용으로 흔히 사용되는 ATmega128만 하여도 64개의 핀을 가지고 있어 ATmega328에 비해 2배 이상 핀의 수가 많다. 하지만 ATmega128의 속도와 기능은 ATmega328과 거의 동일하며 단지 동일한 기능의 핀 개수만을 늘려놓은 것에 지나지 않는다. 다른 마이크로컨트롤러의 경우에도 기본 기능은 ATmega328과 크게 차이가 없으므로 28핀의 소형 마이크로컨트롤러로도 마이크로컨트롤러를 공부를 시작하기에 부족함은 전혀 없다.

5. 주변장치와 어떻게 데이터를 교환하나?

마이크로컨트롤러는 입출력 장치가 제거되고 케이스가 벗겨진 컴퓨터에 해당한다. 이 상태로는 할 수 있는 일이 많지 않으며 주변 장치와 데이터를 주고받음으로써 작고 느리지만 완전한 컴퓨터로서 동작하게 된다. 먼저 마이크로컨트롤러가 데이터를 주고받는 방식에 대해 알아보자.

마이크로컨트롤러는 디지털 데이터를 처리한다. 디지털 데이터는 영(0)과 일(1)의 값만을 가지며 주변장치와 데이터를 주고받는 경우에도 마찬가지다. 주변장치와 데이터 교환은 마이크로컨트롤러의 핀을 통해 이루어지며 ATmega328의 경우 28개의 핀 중 23개의 핀을 데이터 교환을 위해 사용할 수 있으므로 동시에 최대 23비트의 데이터를 주변장치와 교환할 수 있다. 즉, 핀 하나는 한 번에 1비트의 데이터 교환이 가능하다. 데이터를 교환하는 방식은 ATmega328과 주변장치 사이에 상호 합의된 방식으로 행해지며 이를 프로토콜(protocol)이라고 한다. 프로토콜에는 관해서는 이 책에서 자세히 다루지 않을 것이지만 전송 속도에 관해서는 주의하여야 한다.

마이크로컨트롤러는 대부분 핀의 개수가 그리 많지 않으므로 데이터 교환을 위한 핀의 개수가 제한되어 있다. ATmega328의 23개 데이터 핀이 많아 보이지만 작은 LCD 화면을 제어하기 위해 10개 이상의 데이터 핀이 필요한 경우도 있으므로 정보 출력을 위해 LCD 화면 하나를 연결하여 절반 이상의 사용 가능한 핀을 사용하게 되면 사용 가능한 핀이 부족하여 원하는 주변 장치를 연결할 수 없는 경우가 발생한다. 따라서 마이크로컨트롤러에서는 여러 핀을 동시

에 사용하여 데이터를 교환하는 병렬 방식보다는 하나의 핀을 통해 여러 번에 걸쳐 데이터를 교환하는 직렬 방식이 선호된다. ATmega328의 내부에서는 데이터를 고속으로 전달하기 위해 한 번에 8개의 데이터를 병렬로 전달한다. 하지만 CPU를 벗어나면 일반적으로 데이터는 직렬로 8번에 나누어 전달된다. 병렬과 직렬 모두에서 마찬가지로 데이터는 동기화가 중요하며 여기에 전송 속도가 관련된다. CPU 내부에서의 데이터 전달은 메인 클록에 의해 제어되므로 보내는 쪽과 받는 쪽이 동일한 클록을 공유하고 있어 문제가 되지 않는다. 하지만 주변 장치와 데이터 교환이 필요한 경우 보내는 쪽과 받는 쪽이 동일한 속도로 데이터를 검사하지 않으면 잘못된 정보가 전달될 수 있다. 그림 5는 데이터 핀으로 디지털 데이터를 전달하는 경우를 나타낸 것이다.

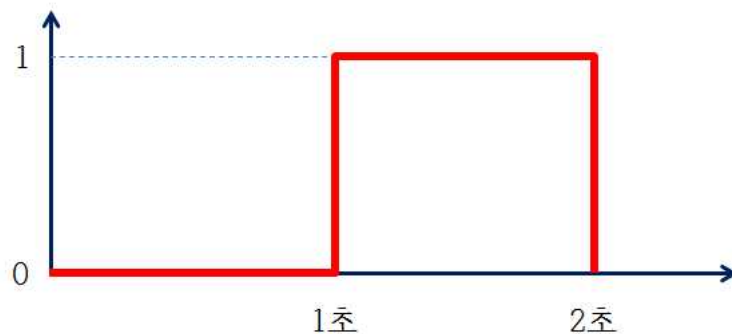


그림 5. 시리얼 데이터 전송

그림 5에서 데이터 전송 속도가 1Hz, 즉, 초당 1비트의 데이터를 전달한다고 가정하면 전송되는 데이터는 '01'의 2비트가 된다. 하지만 전송 속도가 2Hz, 즉, 초당 2비트의 데이터를 전달한다고 가정하면 전송되는 데이터는 '0011'의 4비트로 전혀 다른 데이터가 된다. 따라서 주변장치와 정보를 교환하고자 한다면 보내는 속도와 받는 속도를 일치시켜 주어야 하며 이를 동기화라고 한다.

주변장치와 정보를 교환하는 방식에도 여러 가지가 있다. 대부분의 마이크로컨트롤러가 지원하며 ATmega328에서도 사용할 수 있는 방식에는 UART, SPI, I2C가 있다. UART는 1:1 방식이며 SPI와 I2C는 1:n 방식이다. UART(Universal Asynchronous Receiver/Transmitter)는 컴퓨터나 기타 주변장치의 시리얼 포트를 이용한 직렬 통신 방식으로 마이크로컨트롤러에서는 대부분 RS-232 프로토콜과 함께 사용된다. UART 방식에서는 정보 교환을 위해 송신과 수신은 두 개 데이터 핀 연결을 필요로 하며 송신과 수신은 동시에 진행될 수 있다.

SPI(Serial Peripheral Interface)는 짧은 거리에서 주변장치와 고속의 정보 교환을 위해 주

로 사용되며 3개의 데이터 핀 연결과 1개의 제어 핀 연결, 총 4개의 핀 연결이 필요하다. 3개의 연결 핀 중 2개는 UART에서와 마찬가지로 데이터 송수신을 위해 사용되며 나머지 하나는 동기화를 위한 신호를 보내기 위해 사용한다. 여러 개의 SPI 장치를 사용하는 경우 3개의 데이터 연결 핀은 공유가 가능하므로 추가되는 장치의 수만큼 제어 핀이 추가로 필요해지는 단점은 있다.

I2C(Inter-Integrated Circuit)는 짧은 거리에서 주변장치와 저속의 정보 교환을 위해 주로 사용되며 2개의 데이터 핀 연결만을 필요로 한다. I2C 장치는 소프트웨어적인 주소에 의해 구별되므로 별도의 제어 핀 연결은 필요하지 않으므로 연결하는 I2C 장치의 수가 추가되는 핀이 없는 장점이 있다. 하지만, 두 개의 데이터 핀 중 하나는 데이터 송수신을 위해 사용되고 다른 하나는 동기화를 위한 신호를 위해 사용되는 반이중(half-duplex) 방식이므로 위의 두 방법과 달리 동시에 송수신은 불가능하다는 단점은 있다.

	UART	SPI	I2C
연결 방법	1:1	1:n	1:n
전송 방법	전이중 (동시에 송수신 가능)	전이중 (동시에 송수신 가능)	반이중 (송신 또는 수신만 가능)
데이터 연결 핀	2개	2개	1개
동기화 클럭 연결 핀	-	1개	1개
제어 연결 핀	-	1개	-
총 연결 핀	2개	4개	2개

표 3. 시리얼 통신 방식 비교

이들 데이터 교환 방식을 통해 마이크로컨트롤러는 다양한 주변 장치와 데이터를 주고받음으로써 완전한 컴퓨터로 동작할 수 있다. 마이크로컨트롤러가 컴퓨터와 같이 키보드나 모니터를 사용하는 경우는 거의 없다. 일반적으로 마이크로컨트롤러는 입력 장치로 다양한 센서를 사용하여 주변 데이터를 수집하고 이를 처리하여 또 다른 주변 장치 제어를 위한 신호를 출력하는 형태로 구성된다. 또는 수집된 데이터를 컴퓨터로 전달하여 컴퓨터에서 전체적으로 데이터를 처리하고 관리할 수 있도록 도와주는 역할을 수행하게 된다.

6. 프로그램은 어떻게 만들어지나?

컴퓨터에서 프로그램을 작성하는 경우를 생각해 보자. 먼저 프로그램 작성을 위해 컴파일러를

설치하여야 하며 키보드로 코드를 입력하고 컴파일 하여 실행 파일을 만들어 낸다. 물론 디버깅을 위해서는 모니터도 있어야 할 것이다. 마이크로컨트롤러의 경우는 키보드와 모니터는 없다는 것을 이미 알고 있다. 컴파일러의 경우는 어떤가? ATmega328에 프로그램을 저장할 수 있는 메모리는 32KByte에 불과하다. 윈도우즈에서 가장 간단한 프로그램 중 하나인 메모장만 하더라도 그 크기가 수백 KByte에 달하여 ATmega328에서 실행하는 것은 불가능하다. 하물며 프로그램 개발을 위해 많이 사용되는 Visual Studio의 경우에는 수백 메가바이트의 공간을 필요로 한다. 한 마디로 ATmega328에서 프로그램을 개발하는 것은 불가능하다. 따라서 마이크로컨트롤러를 위한 프로그램은 컴퓨터에서 개발하고 실행 파일만을 마이크로컨트롤러에 다운로드 하여 설치하는 과정을 거친다. 이처럼 프로그램이 개발되는 환경(개발 시스템)과 프로그램이 실행되는 환경(목적 시스템)이 서로 다른 경우를 일컬어 교차 개발 환경(Cross Development Environment)이라 한다.

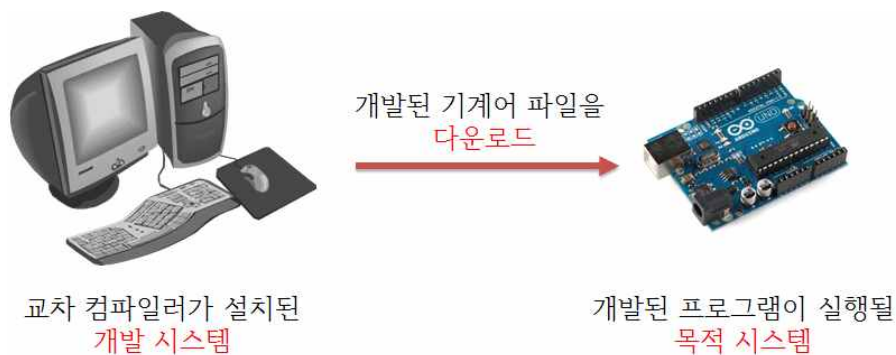


그림 6. 교차 개발 환경

교차 개발이 가능하기 위해서는 개발 시스템에서 동작하면서 목적 시스템에서 실행 가능한 기계어 파일을 생성할 수 있는 교차 컴파일러(Cross Compiler)가 필요하며 생성된 기계어 파일을 목적 시스템에 설치할 수 있는 방법이 필요하다. ATmega328을 위한 교차 컴파일러에는 아두이노⁴⁾에서 제공하는 아두이노 프로그램과 아트멜⁵⁾에서 제공하는 Atmel Studio가 있으며 이 책에서는 두 가지 모두를 사용할 것이다. 이외에도 몇 가지 다른 교차 컴파일러를 사용할 수 있지만 여기서는 다루지 않는다. 교차 컴파일러는 ATmega328 마이크로컨트롤러에서 실행 가능한 실행 파일, 즉, 기계어 파일을 생성해 준다. Windows에서 실행 가능한 기계어 파일이 EXE 또는 DLL의 확장자를 가지는 반면, ATmega328에서 실행 가능한 기계어 파일은 HEX 확장자를 가진다.

4) <http://arduino.cc>

5) <http://www.atmel.com>

기계어 파일이 생성되면 이를 마이크로컨트롤러에 설치하는 과정이 필요하며 이는 직렬 통신을 통해 개발 시스템에서 마이크로컨트롤러로 기계어 파일을 전송함으로써 이루어진다. 이처럼 실행 가능한 기계어 파일을 개발 시스템에서 목적 시스템으로 전송하여 설치하는 과정을 ‘다운로드’ 또는 ‘업로드’라 한다. 다운로드의 마이크로컨트롤러의 입장에서 개발 시스템으로부터 프로그램을 가져오는 것을, 업로드는 개발 시스템에서 마이크로컨트롤러로 프로그램을 보내는 것을 중심으로 이야기한다. 이처럼 다운로드와 업로드는 정반대의 의미이지만 동일한 방향의 데이터 흐름을 가리키므로 일반적으로 동일한 의미로 사용되고 있다. 다운로드된 프로그램은 마이크로컨트롤러의 메모리에 설치되며 마이크로컨트롤러에는 일반적으로 하나의 프로그램만이 설치될 수 있으므로 마이크로컨트롤러에 전원을 인가하면 간단한 부팅 과정을 거쳐 현재 설치된 프로그램이 자동으로 실행된다.

프로그램을 업로드 하기 위해서는 UART와 SPI 방식을 사용할 수 있다. 아두이노에서는 별도의 추가 장치 없이 프로그램을 업로드하기 위해 UART 방식을 사용한다. 반면 일반적으로 마이크로컨트롤러 프로그래밍에서는 SPI를 통해 프로그램을 업로드 하는 ISP (In System Programming) 또는 ICSP (In Circuit Serial Programming) 방식을 사용한다. 이 책에서는 두 가지 방식 모두를 사용하여 프로그램을 업로드 해 볼 것이다.

교차 개발 환경은 스마트폰을 위한 어플리케이션 개발에서도 볼 수 있다. 스마트폰은 마이크로프로세서(또는 어플리케이션 프로세서)를 사용하지만 스마트폰용 운영체제에서는 개발 도구를 제공하지 않으므로 컴퓨터에서 어플리케이션을 개발한 후 개발 컴퓨터나 장터에서 어플리케이션을 다운로드하고 스마트폰에 설치한다. 마이크로컨트롤러의 경우와 동일하지 않은가? 물론 스마트폰에는 운영체제가 있고 여러 개의 프로그램을 설치하고 실행할 수 있는 점은 마이크로컨트롤러와는 차이가 있다.

7. 마이크로컨트롤러가 꼭 필요한가?

어두워지면 자동으로 불이 켜지는 전등을 만들고 싶다고 가정해 보자. 마이크로컨트롤러가 필요할까? 요구 사항에 따라 달라지겠지만 단순히 어두워지면 불이 켜지는 동작은 마이크로컨트롤러 없이도 가능하다. 빛의 양에 따라 저항이 변하는 조도센서를 이용하면 그림 7의 회로로 어두워지면 자동으로 불이 켜지게 할 수 있다. 저항의 값을 조절하면 불이 켜지는 시점에서의 광량도 조절할 수 있다.

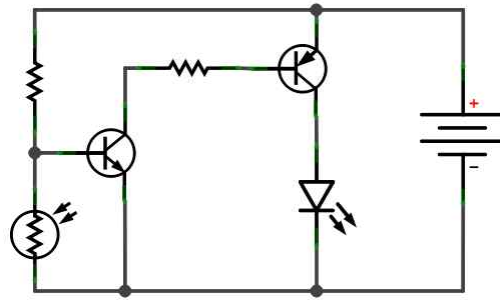


그림 7. 스위칭 회로를 이용한 자동 점등 회로

마이크로컨트롤러를 이용하여서도 동일한 동작을 수행하도록 구성할 수 있으며 그림 8은 ATmega328 마이크로컨트롤러를 이용하여 구성한 회로의 예이다.

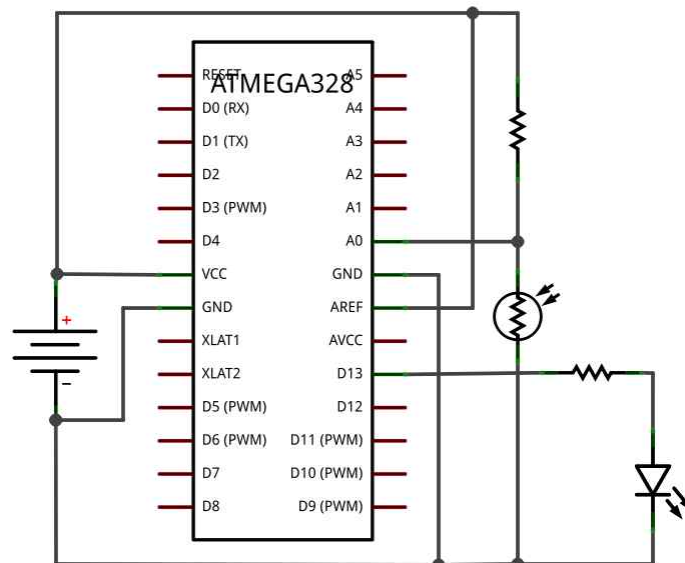


그림 8. 마이크로컨트롤러를 이용한 자동 점등 회로

트랜지스터와 저항으로 구성되는 그림 7의 회로에 비해 그림 8의 회로는 마이크로컨트롤러를 사용하므로 가격이 비싸고 배선도 복잡해 보이며 별도의 프로그램을 작성해서 업로드 해야 하는 등 번거롭고 아무런 이점도 없어 보인다. 실제 이 예의 경우 마이크로컨트롤러를 사용하는 경우의 이점은 없다! 그렇다면 마이크로컨트롤러를 사용할 필요가 없지 않은가? 어두워지면 불이 켜지는 간단한 동작이 필요한 경우 마이크로컨트롤러를 이용하는 경우 이점은 거의 없지만 여기에 기능을 추가하거나 변경하고자 한다면 마이크로컨트롤러를 사용하는 장점을 발견할 수 있다.

그림 7의 회로에서 불이 켜지기 위한 기준 광량을 바꾸고 싶다면 어떻게 해야 할까? 회로에

서 저항을 다른 크기의 저항으로 교체하면 된다. 이를 위해서는 저항을 떼어 내고 다시 붙이는 등의 하드웨어 작업이 필요하다. 마이크로컨트롤러를 사용한 경우는 어떨까? 프로그램에서 임계치를 조절하여 프로그램을 다시 업로드 하기만 하면 된다. 마이크로컨트롤러를 사용하는 경우가 간단하지 않은가?

기능을 추가하는 경우에는 마이크로컨트롤러를 사용하는 이점이 보다 명확히 나타난다. 그림 7의 회로에 한 번 불이 켜지면 최소 10초는 불이 켜져 있도록 하고 싶다면 어떻게 해야 할까? 불이 켜진 시간을 알아내기 위해서는 시간을 알려주는 부품이 필요하고, 불이 켜진 시간을 저장해 놓기 위해 메모리가 필요하고, 현재 시간과 저장된 시간을 비교하기 위해 비교기가 필요하다. 이들 부품을 모두 그림 7의 회로에 추가하는 것이 가능하기는 하지만 이 경우 회로를 완전히 다시 그려야 할지도 모르며 완성된 회로는 그림 8보다 더 복잡해질 수 있다. 이에 비해 마이크로컨트롤러를 사용한 회로는 최소 10초 동안 불이 켜져 있도록 하기 위해 회로의 변경이 필요하지 않다. 불이 켜지는 기준 광량을 바꾸는 경우와 마찬가지로 프로그램만 수정하여 업로드 함으로써 기능을 바꿀 수 있다. 여기에 마이크로컨트롤러를 사용하는 이점이 있다. 동일한 입력(광센서)과 출력(LED)을 가지는 회로에서 마이크로컨트롤러를 사용하는 경우 프로그램만을 수정함으로써 서로 다른 동작(광량에 따라 점멸하는 동작과 광량에 따라 점멸하면서 불이 켜졌을 때 최소 10초 동안 켜져 있는 동작)을 구현하는 것이 가능하다. 마이크로컨트롤러는 작고 간단한 컴퓨터다. 컴퓨터에 프로그램을 설치하여 다양한 작업을 수행할 수 있는 것과 마찬가지로 마이크로컨트롤러에도 프로그램을 설치하여 동일한 하드웨어를 이용하여 서로 다른 작업을 수행하는 것이 가능하다. 다만 컴퓨터와 달리 마이크로컨트롤러는 한 번에 하나의 프로그램만을 설치할 수 있는 점은 차이가 있다. 컴퓨터에 비해 다소 불편하고 번거로운 점이 있는 것은 사실이지만, 잊지 말자, 1,000원이면 된다. 백만 원이 아니다.