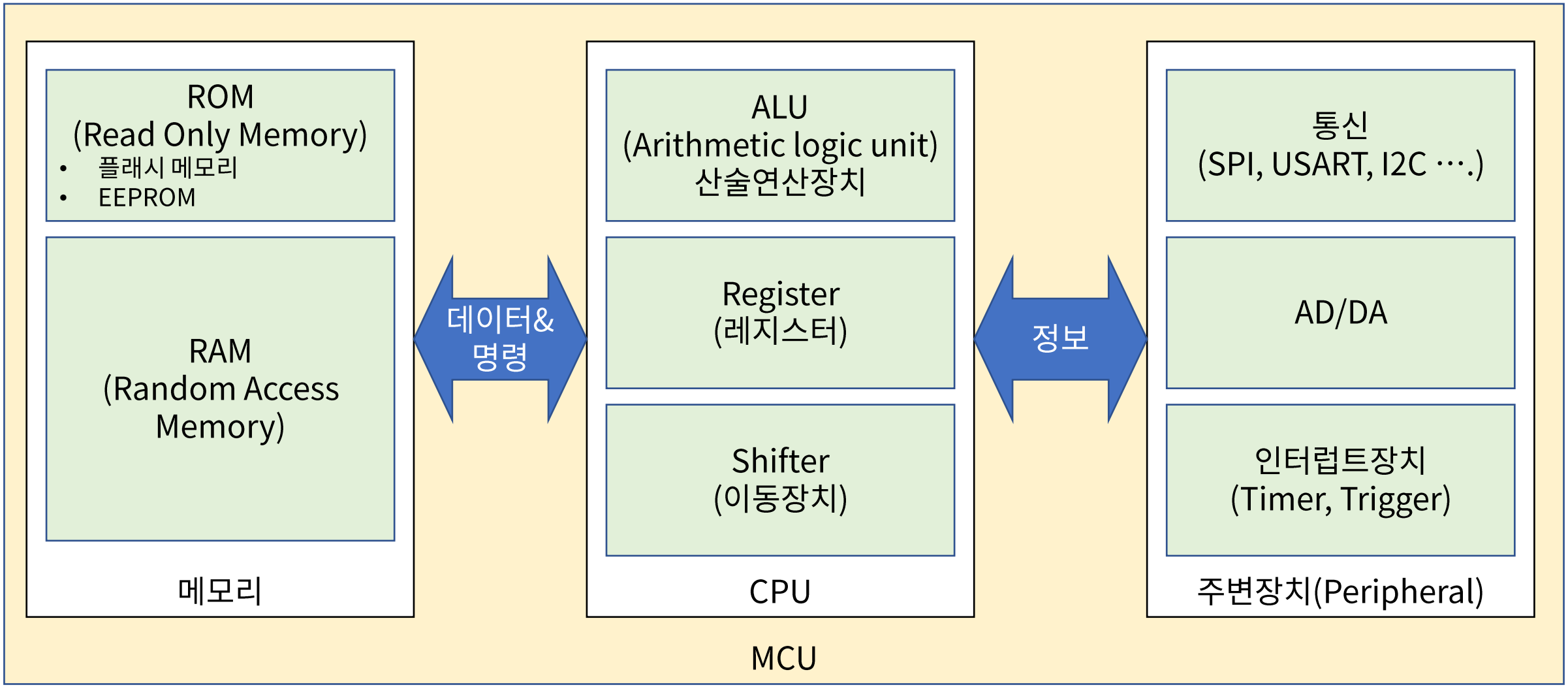
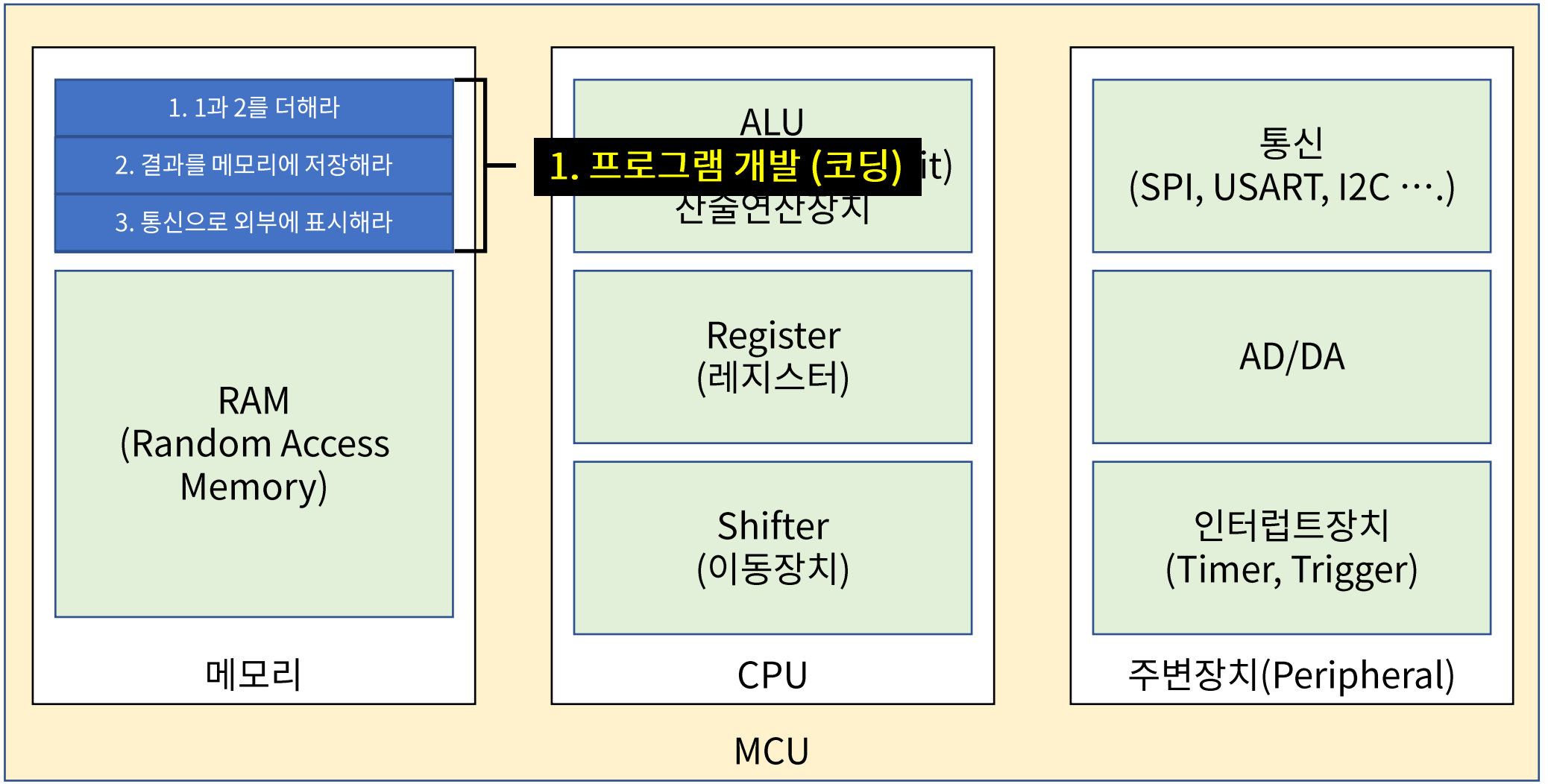
마이크로프로세서 이론

마이크로프로세서 종합 설계. 2주차.

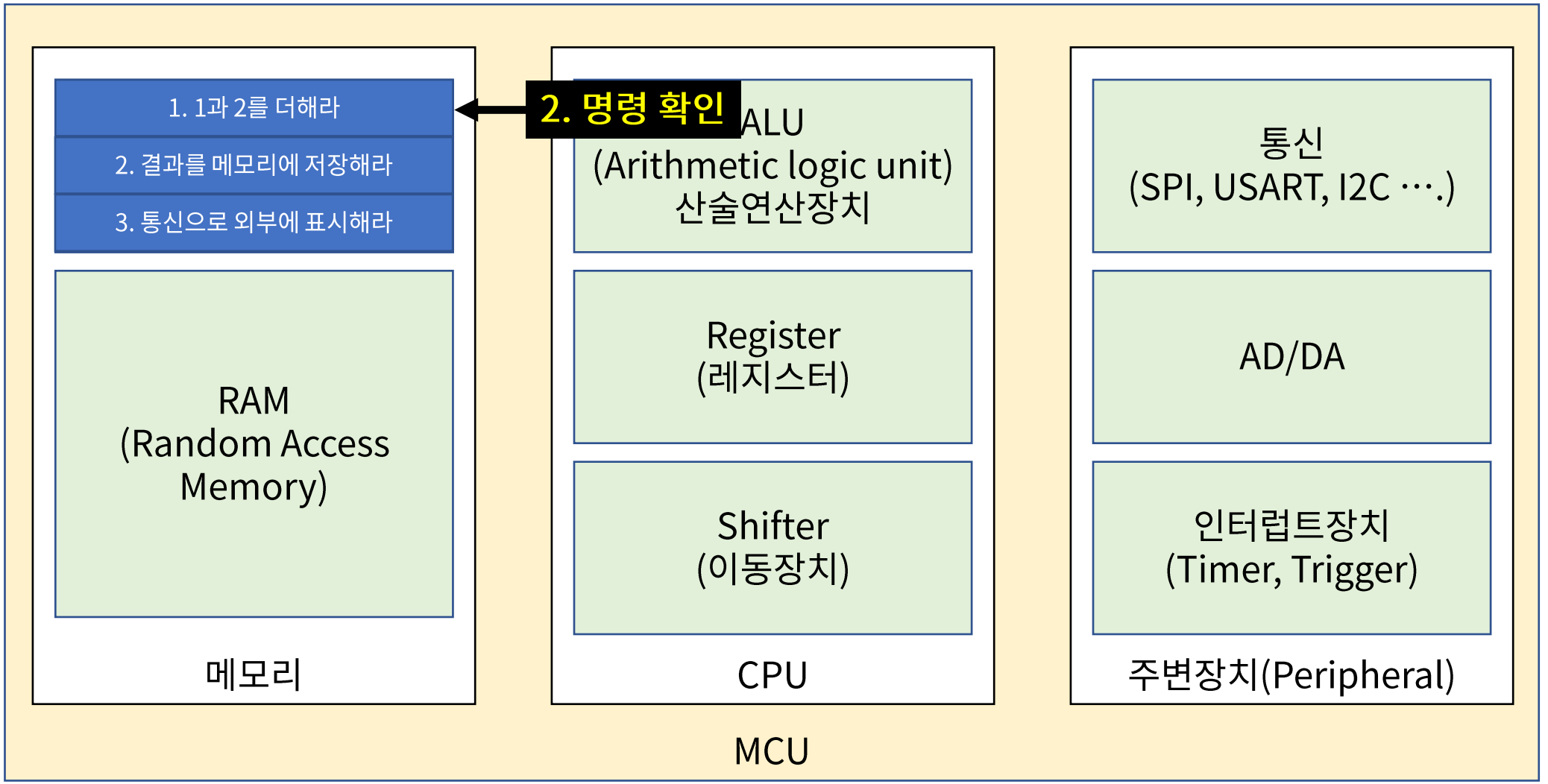
마이크로프로세서의 기본 구성



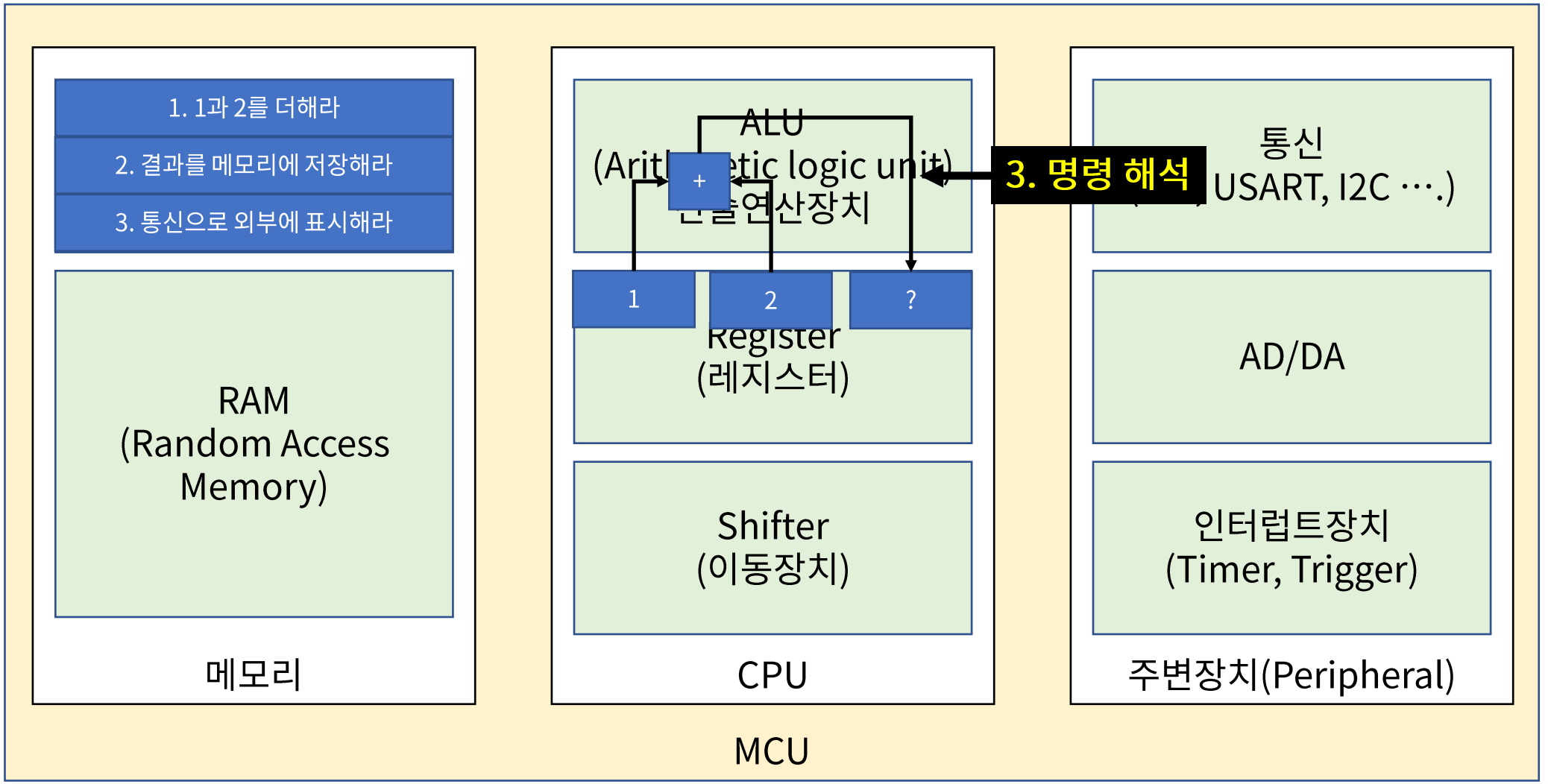
마이크로프로세서는 어떻게 명령을 수행 할까?



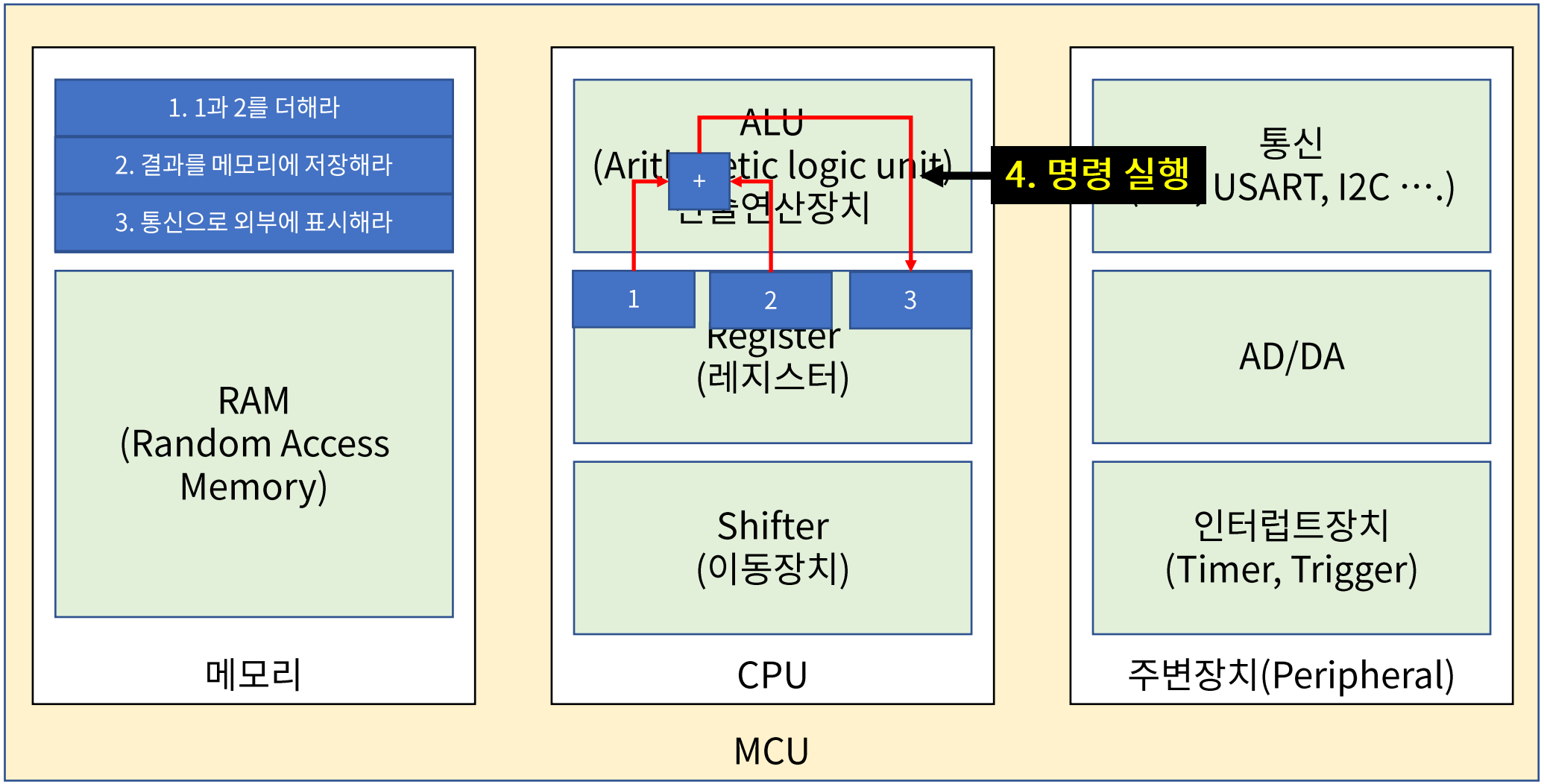
마이크로프로세서는 어떻게 명령을 수행 할까?



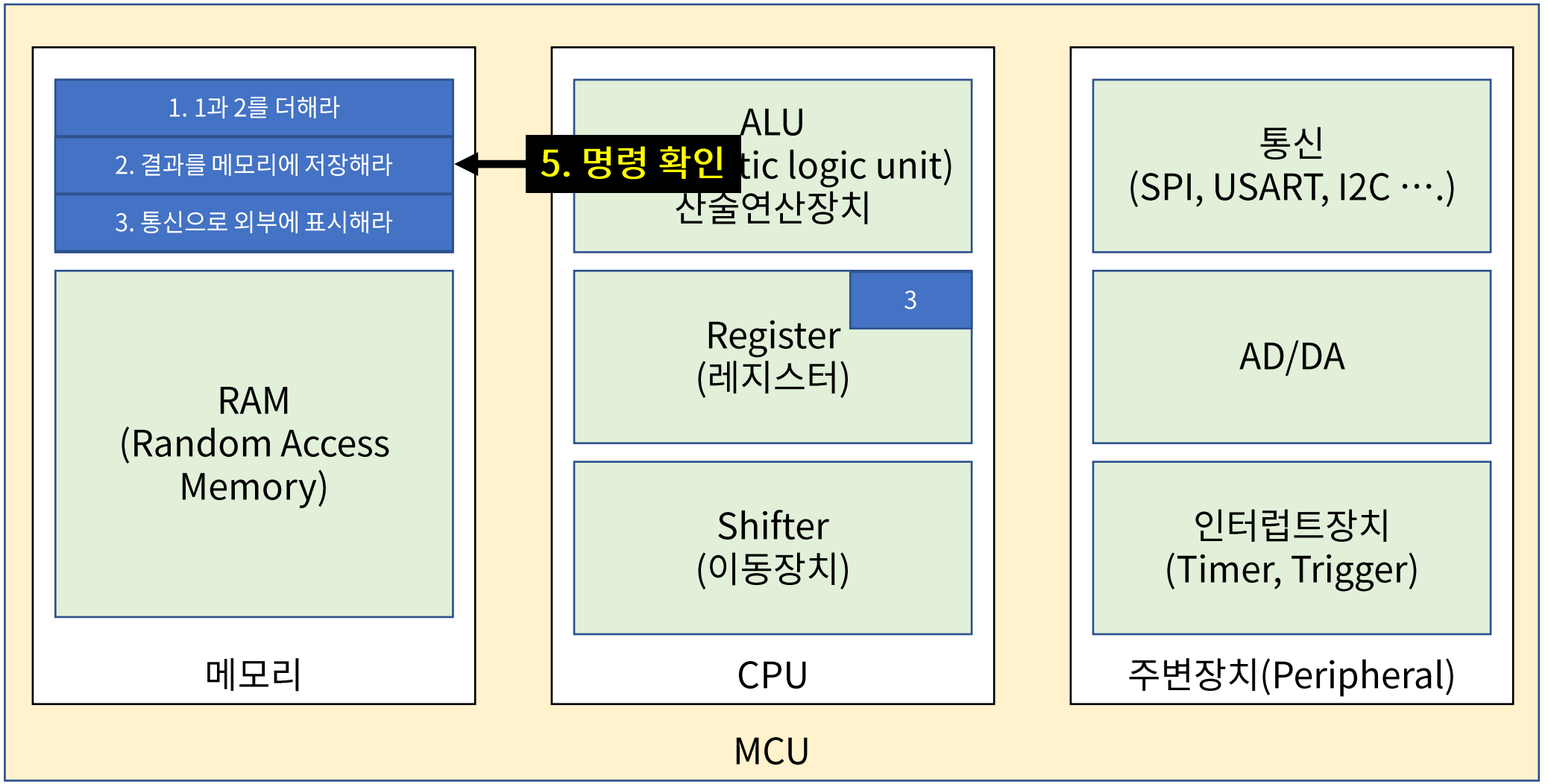
마이크로프로세서는 어떻게 명령을 수행 할까?



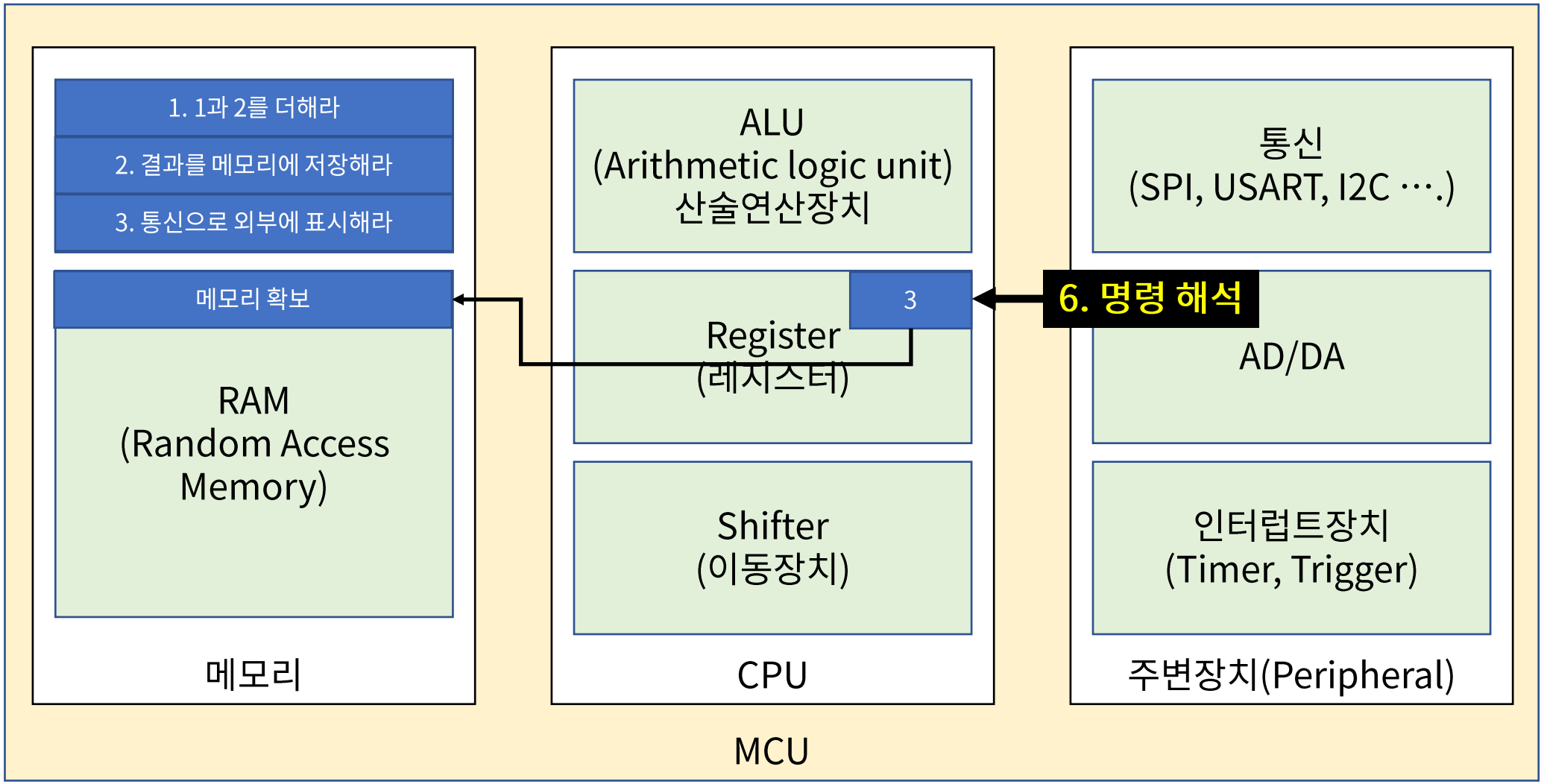
마이크로프로세서는 어떻게 명령을 수행 할까?



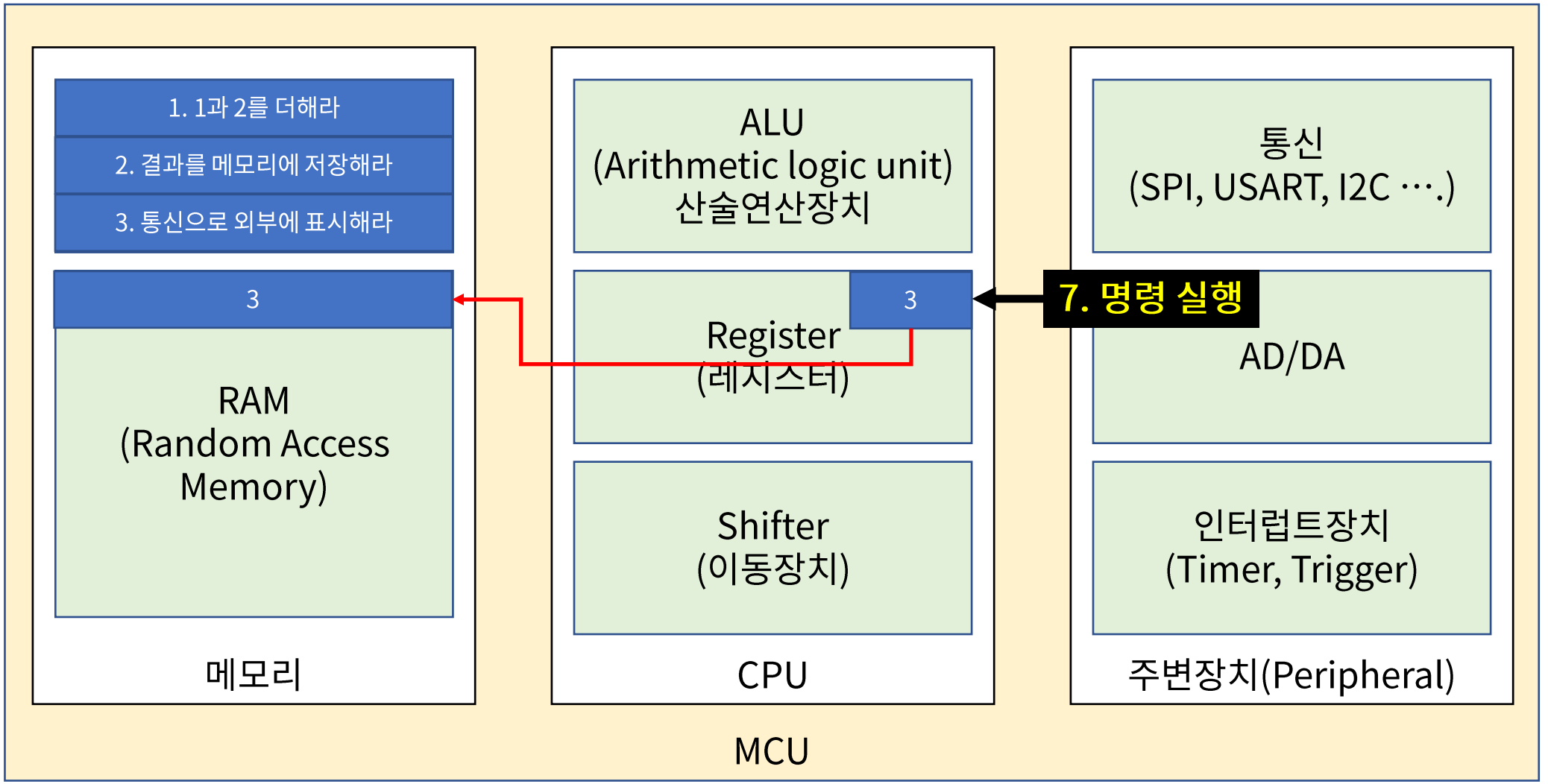
마이크로프로세서는 어떻게 명령을 수행 할까?



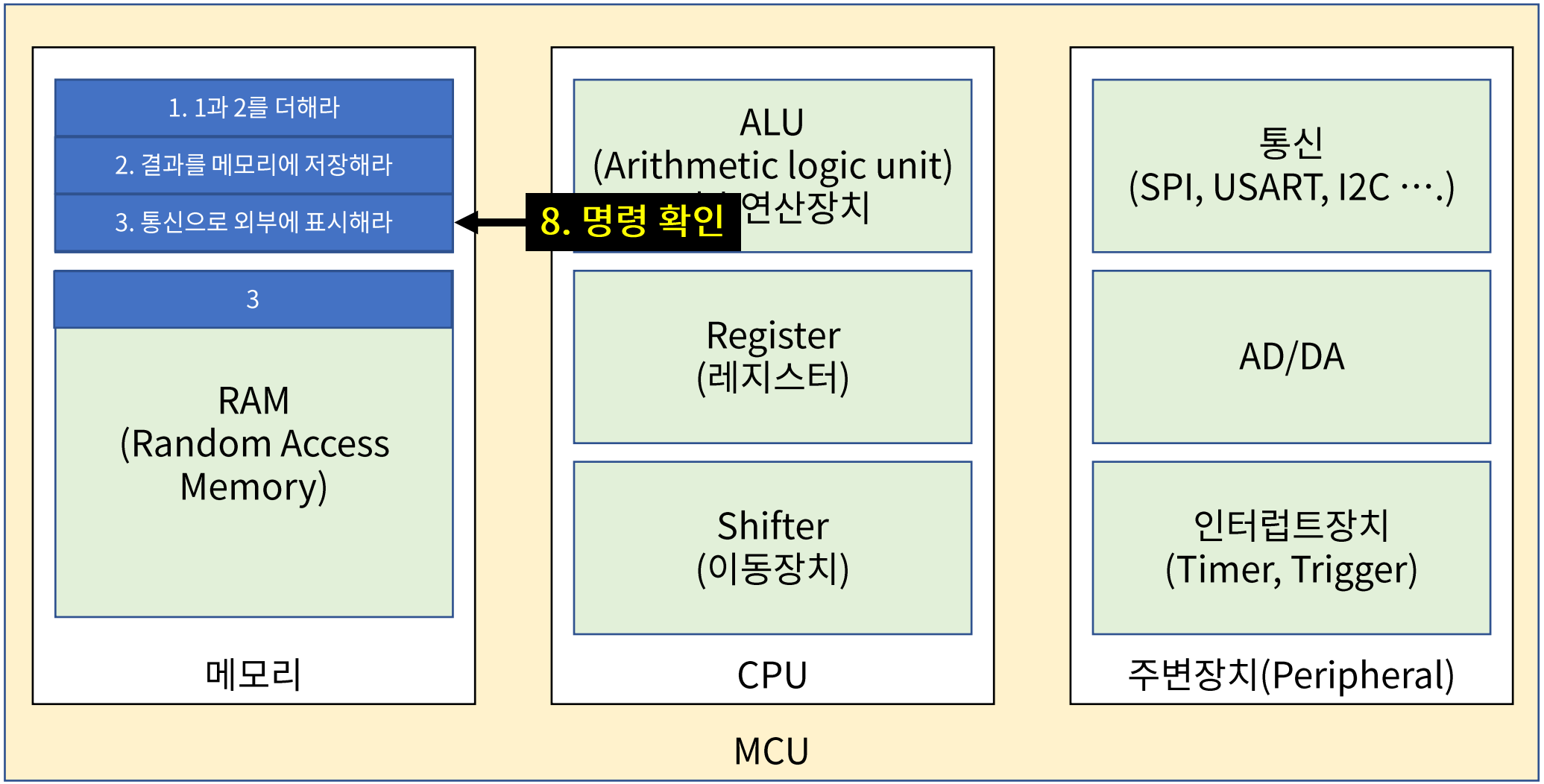
마이크로프로세서는 어떻게 명령을 수행 할까?



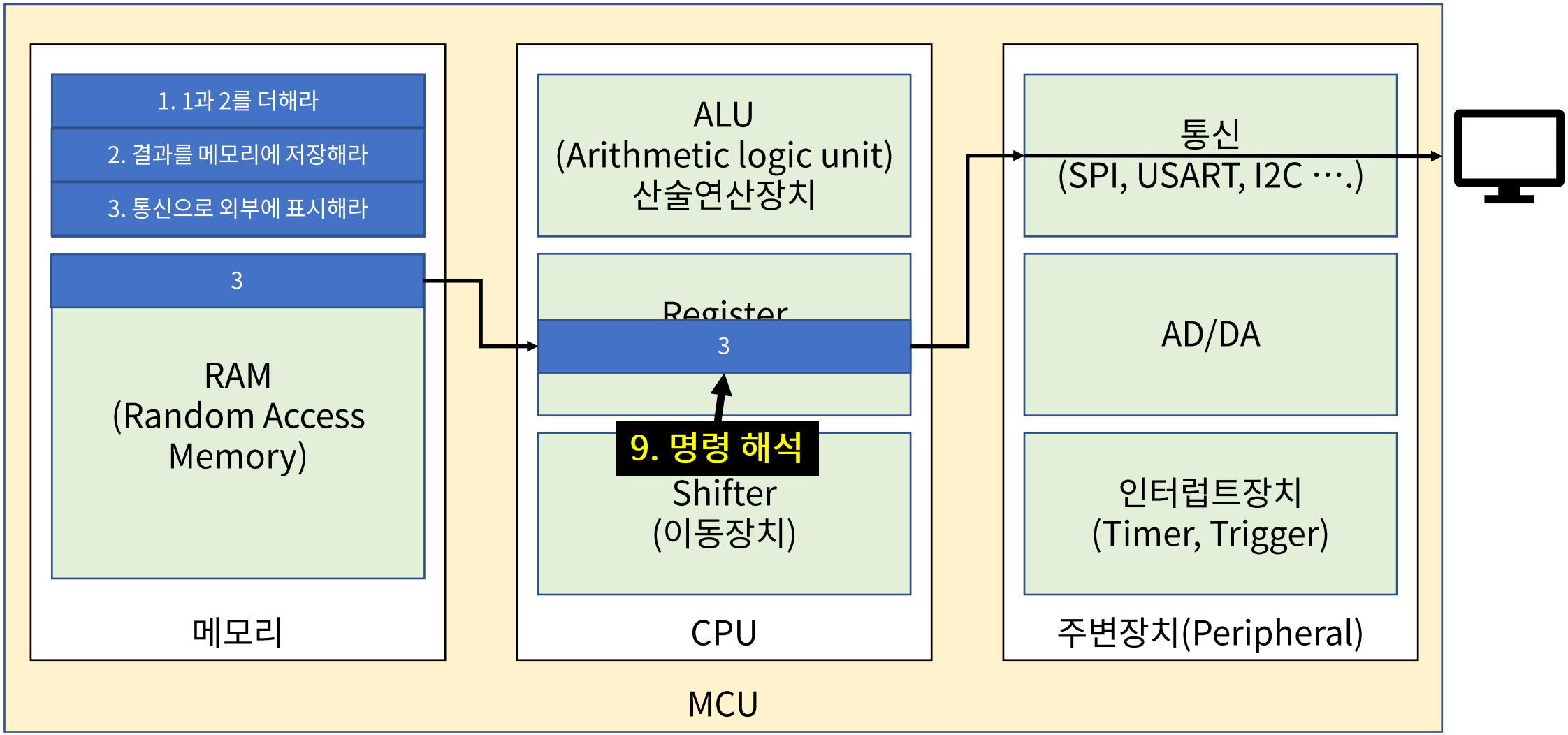
마이크로프로세서는 어떻게 명령을 수행 할까?



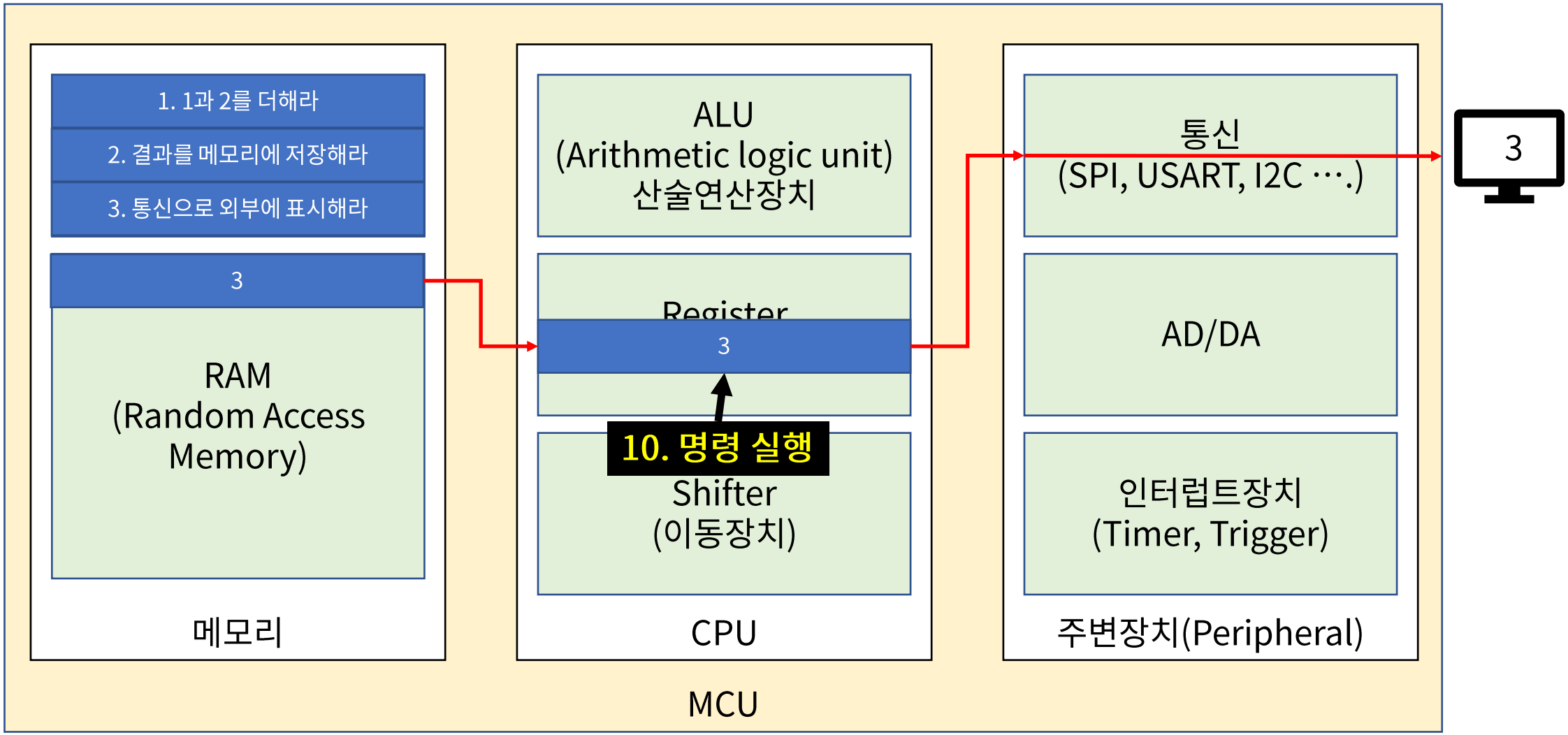
마이크로프로세서는 어떻게 명령을 수행 할까?



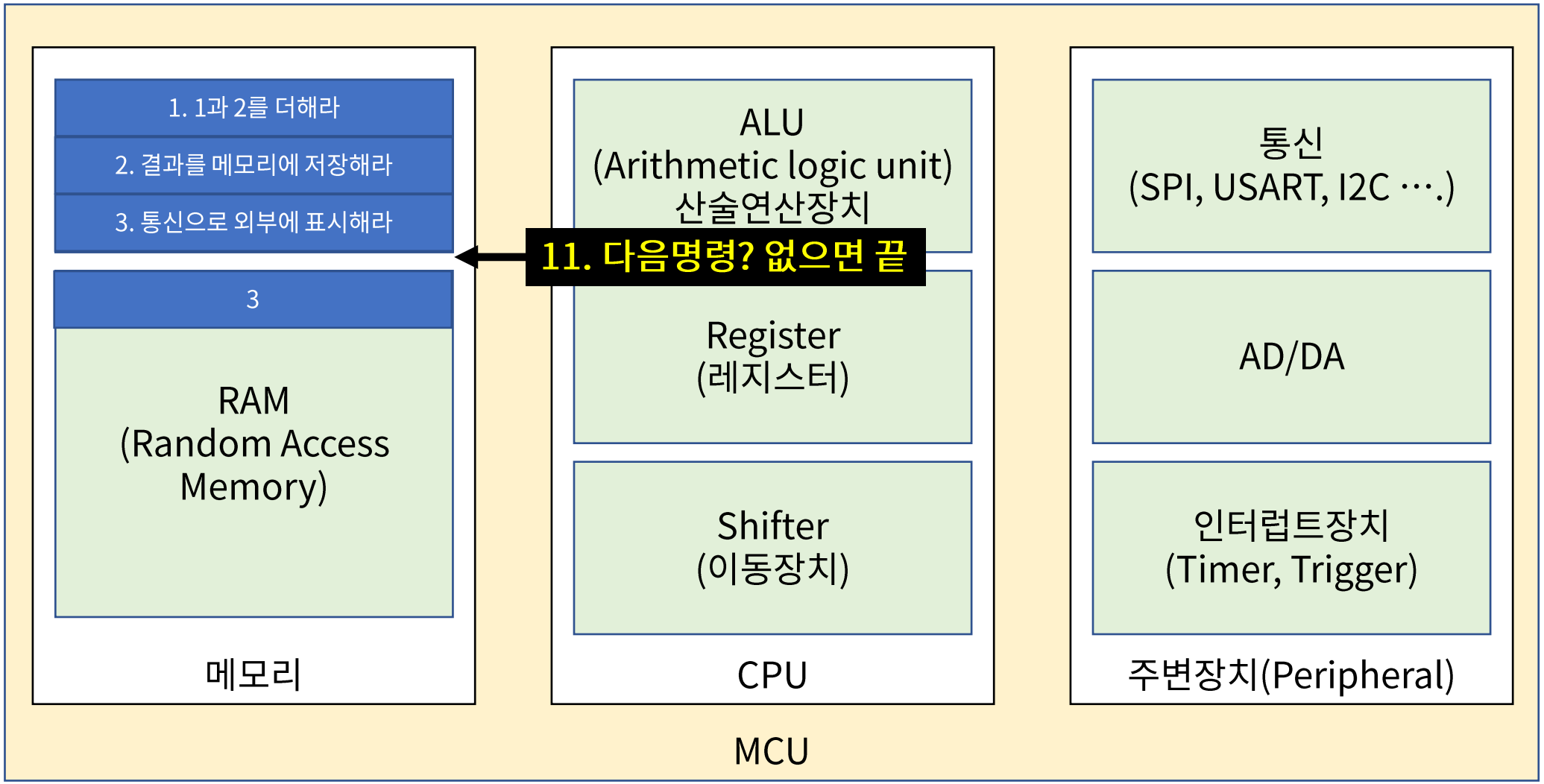
마이크로프로세서는 어떻게 명령을 수행 할까?



마이크로프로세서는 어떻게 명령을 수행 할까?



마이크로프로세서는 어떻게 명령을 수행 할까?

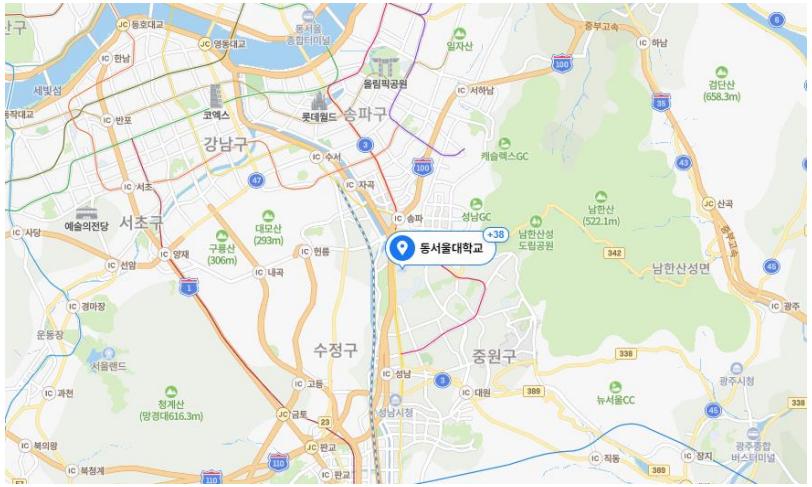


10진수. 2진수? 8진수? 16진수?

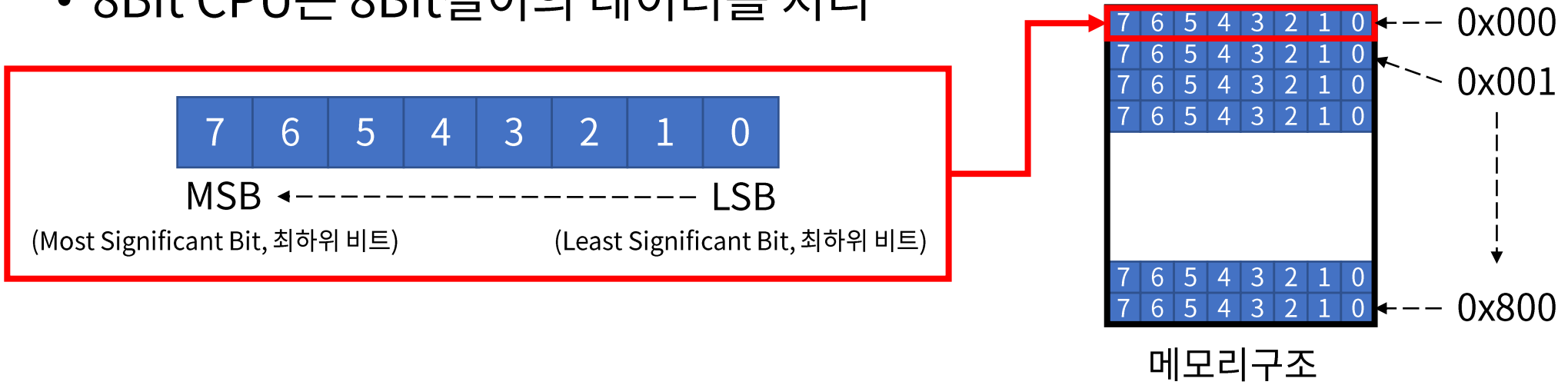
- 10진수 : 우리가 사용하고 있는 수 시스템(손가락은 10개)
- 2진수 : 컴퓨터가 사용하는 기본 수 시스템(1과 0)
- 8진수 : 2진수의 조합을 사람이 쉽게 이해(0~7까지)
- 16진수 : 2진수의 조합을 사람이 쉽게 이해(0~15까지)

- 1(10) → 0001(2) → 001(8) → 0x01(16)
- 8(10) → 1000(2) → 010(8) → 0x08(16)
- 10(10) → 1010(2) → 012(8) → 0x0A(16)
- 255(10) → 1111 1111(2) → 377(8) → 0xFF(16)

Address란 무엇인가?

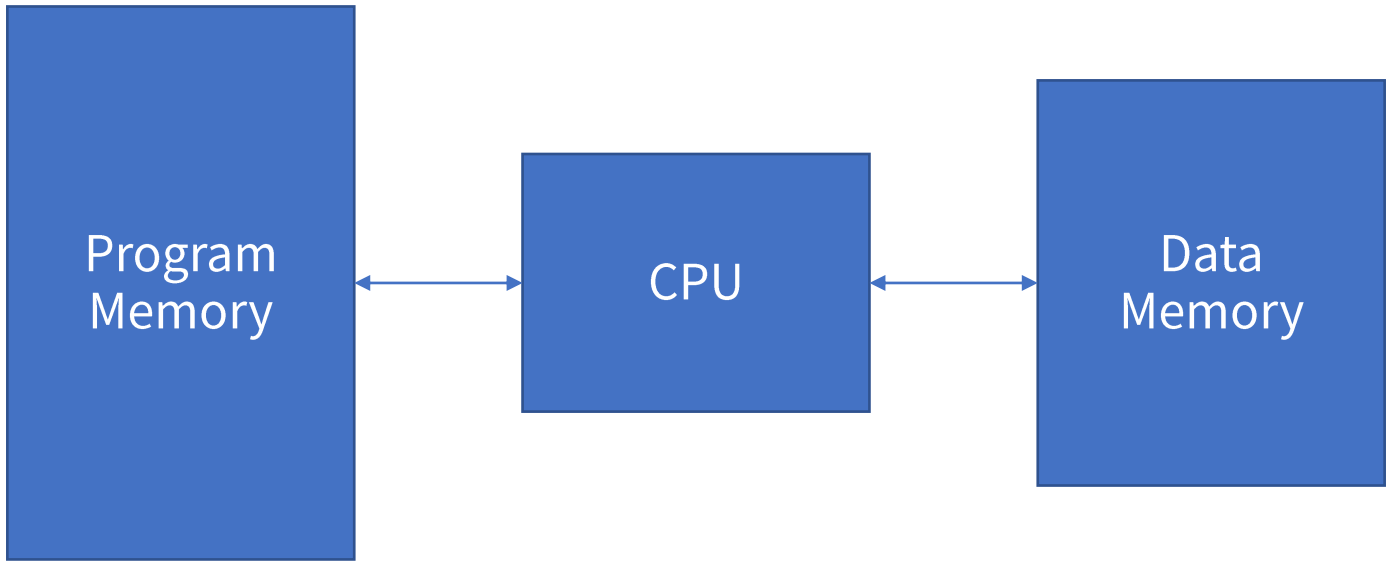


- 메모리의 장소 정보 (주소)
  - 동서울대학교 : 경기 성남시 수정구 복정로 76
- 메모리도 주소(연속숫자)를 이용하여 데이터를 참조 한다.
- ATmega328p의 경우 내부에 2KByte의 RAM을 가지고 있다.
  - 1Byte → 8Bit
  - 8Bit CPU는 8Bit길이의 데이터를 처리



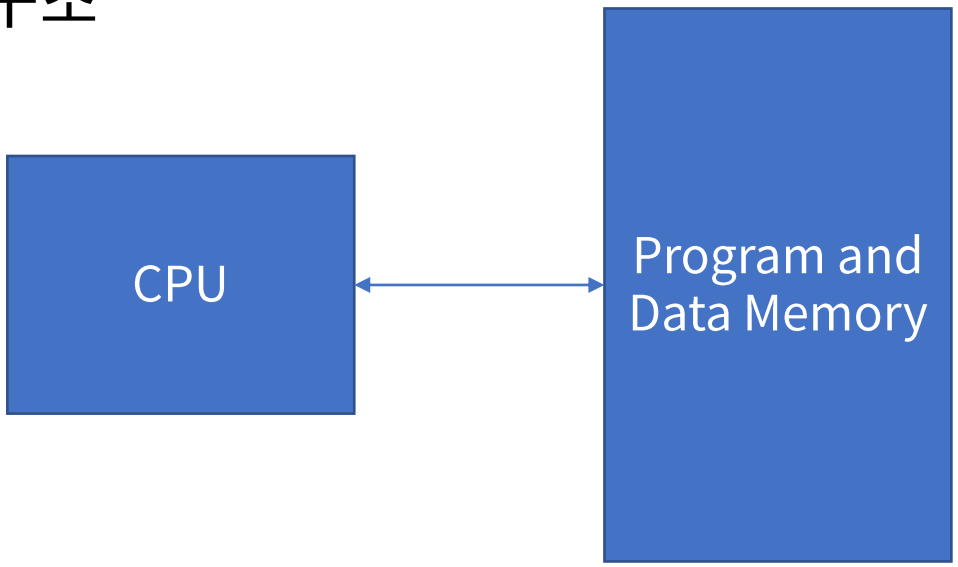
컴퓨터구조(폰노이만vs하버드)

- 하버드 구조
  - 프로그램 메모리와 데이터 메모리가 분리되어 있는 구조
  - 장점 : 명령어와 데이터를 동시에 접근 가능하기 때문에 속도가 빠름
  - 단점 : 설계가 어려움
  - **일반적인 MCU 구조**



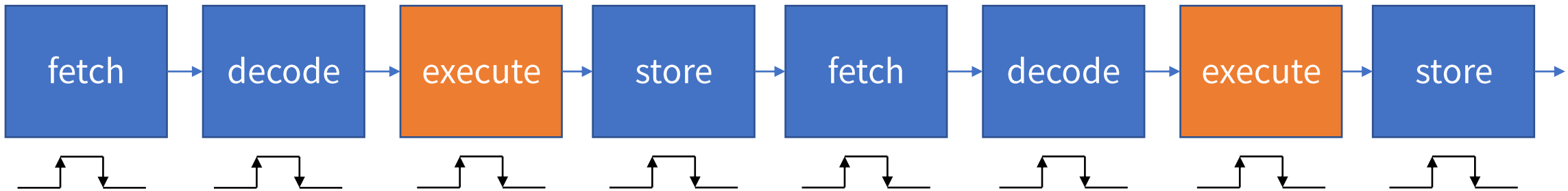
컴퓨터구조(폰노이만vs하버드)

- 폰노이만 구조
  - 프로그램 메모리와 데이터 메모리가 구분되지 않는 구조
  - 장점 : SW 범용성이 좋음
  - 단점 : 병목 현상이 발생
  - 일반적인 PC 구조



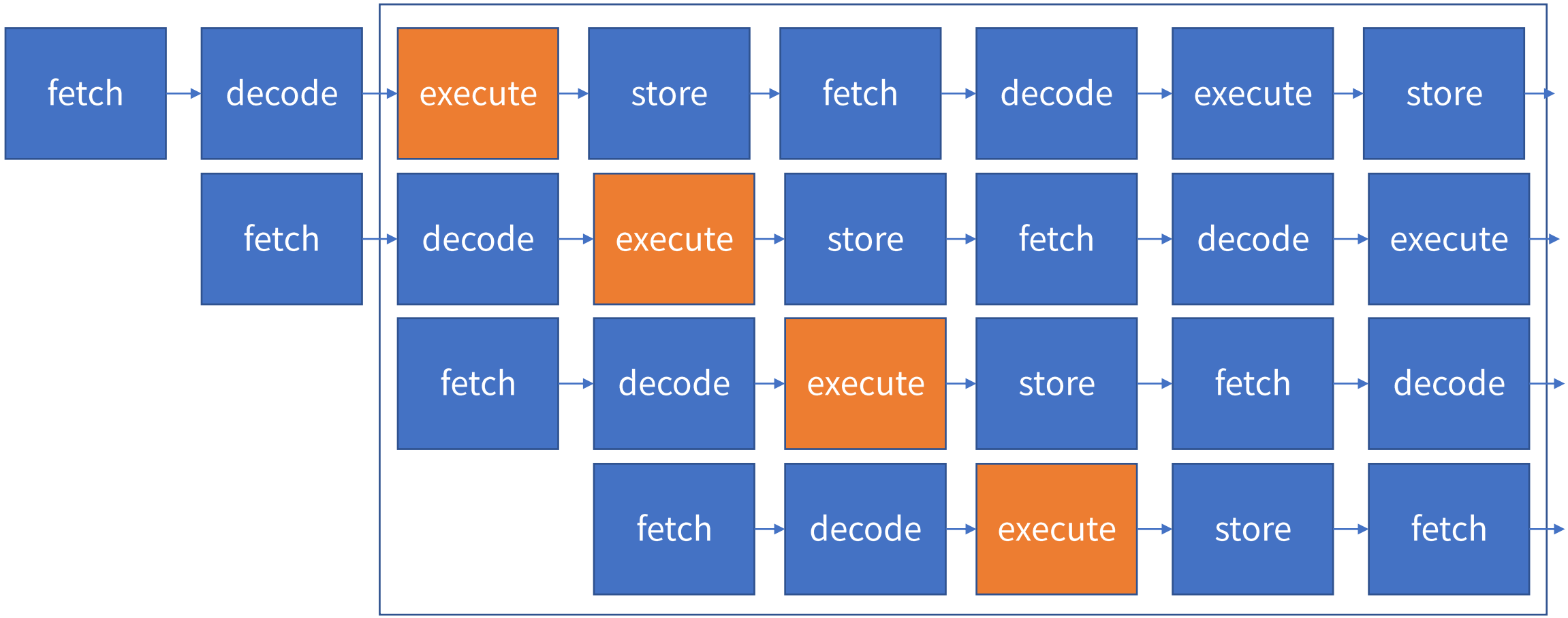
컴퓨터 명령 실행 과정

- Fetch → Decode → Execute → Store



컴퓨터 명령 파이프라인

- Fetch → Decode → Execute → Store



ATMEGA328P

- RISC(Reduced Instruction Set Computer) 구조
  - 적은 수의 명령어로 명령어 집합을 구성하며 복잡한 명령은 명령어를 조합하여 사용
  - Single clock cycle execution동작이 가능한 131개의 명령어로 구성되어 있음
  - 32x8 general register

- 내장 메모리
  - 32kbyte의 flash memory(프로그램 메모리)
  - 1kbyte 크기의 EEPROM
  - 2kbyte 크기의 SRAM

- 주변장치(Peripheral)
  - 2개의 8비트 Timer/Counters, 1개의 16비트 Timer/Counters
  - 6개의 PWM 채널, 8채널 16비트 ADC
  - USART, SPI, I2C, Watchdog
  - 아날로그 비교기
  - 외부 인터럽트
  - 23개의 IO

데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

ATMEGA328P 기본 구조

프로그램카운터 (Program Counter : PC)

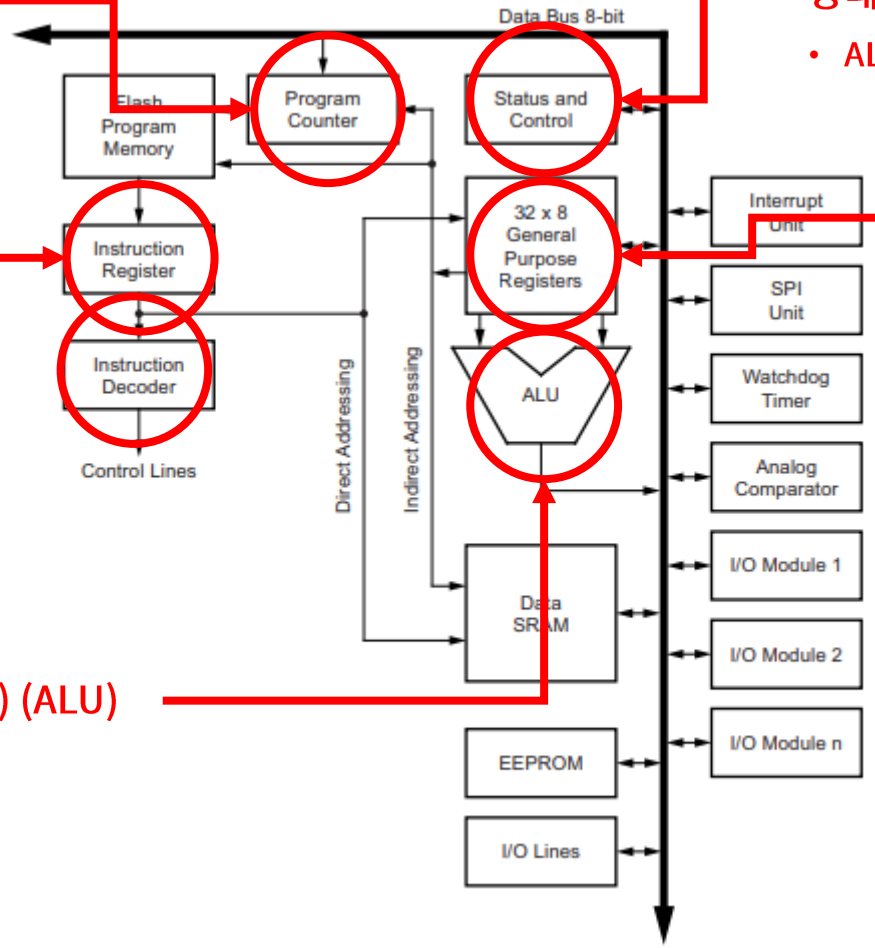
- 내가 실행시킨 명령어의 주소를 가리키는 역할

명령어 레지스터

- 실행 해야 하는 명령어를 저장

산술 논리 장치 (Arithmetic Logical Unit) (ALU)

- 4칙연산
- 논리(AND, OR, NOT)
- 비트연산



상태 레지스터 (Status Register : SREG)

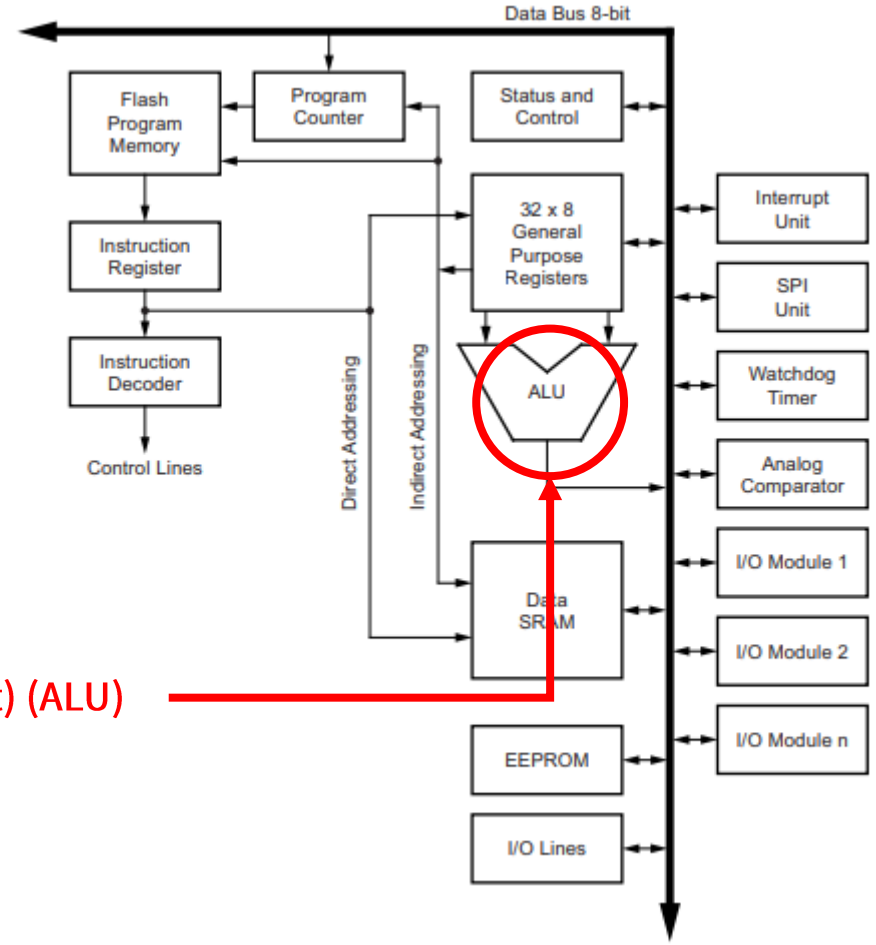
- ALU가 가장 최근에 실행한 연산 명령의 결과와 상태를 표시

범용 레지스터 (General Purpose Register)

- 프로그램 수행 중에 중간 결과나 데이터를 일시적으로 저장하는데 사용

데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

ATMEGA328P 기본 구조

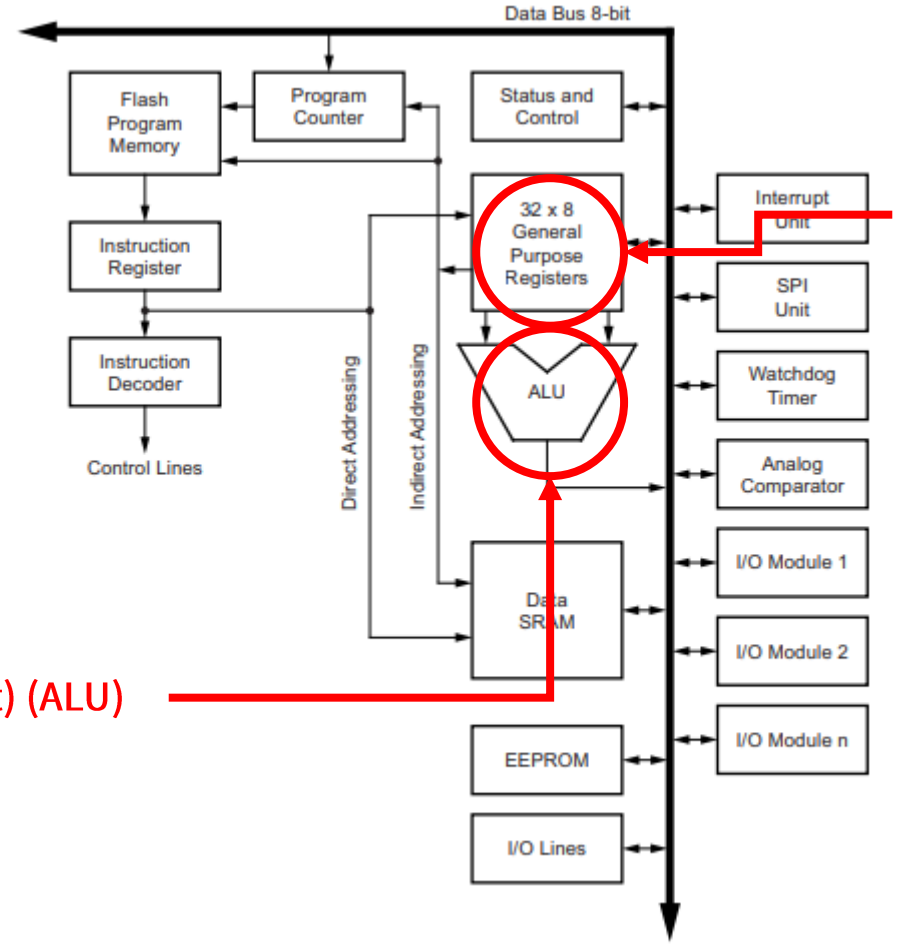


산술 논리 장치 (Arithmetic Logical Unit) (ALU)

- 4칙연산
- 논리(AND, OR, NOT)
- 비트연산

데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

ATMEGA328P 기본 구조



범용 레지스터 (General Purpose Register)

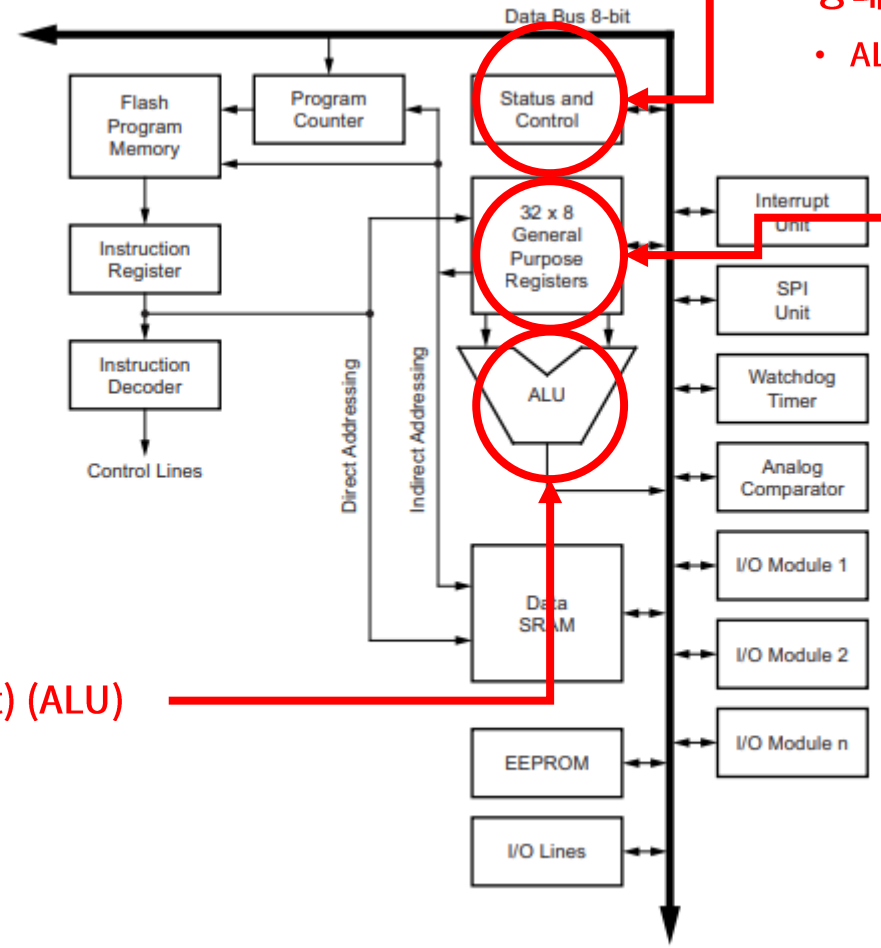
- 프로그램 수행 중에 중간 결과나 데이터를 일시적으로 저장하는데 사용

산술 논리 장치 (Arithmetic Logical Unit) (ALU)

- 4칙연산
- 논리(AND, OR, NOT)
- 비트연산

데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

ATMEGA328P 기본 구조



상태 레지스터 (Status Register : SREG)

- ALU가 가장 최근에 실행한 연산 명령의 결과와 상태를 표시

범용 레지스터 (General Purpose Register)

- 프로그램 수행 중에 중간 결과나 데이터를 일시적으로 저장하는데 사용

산술 논리 장치 (Arithmetic Logical Unit) (ALU)

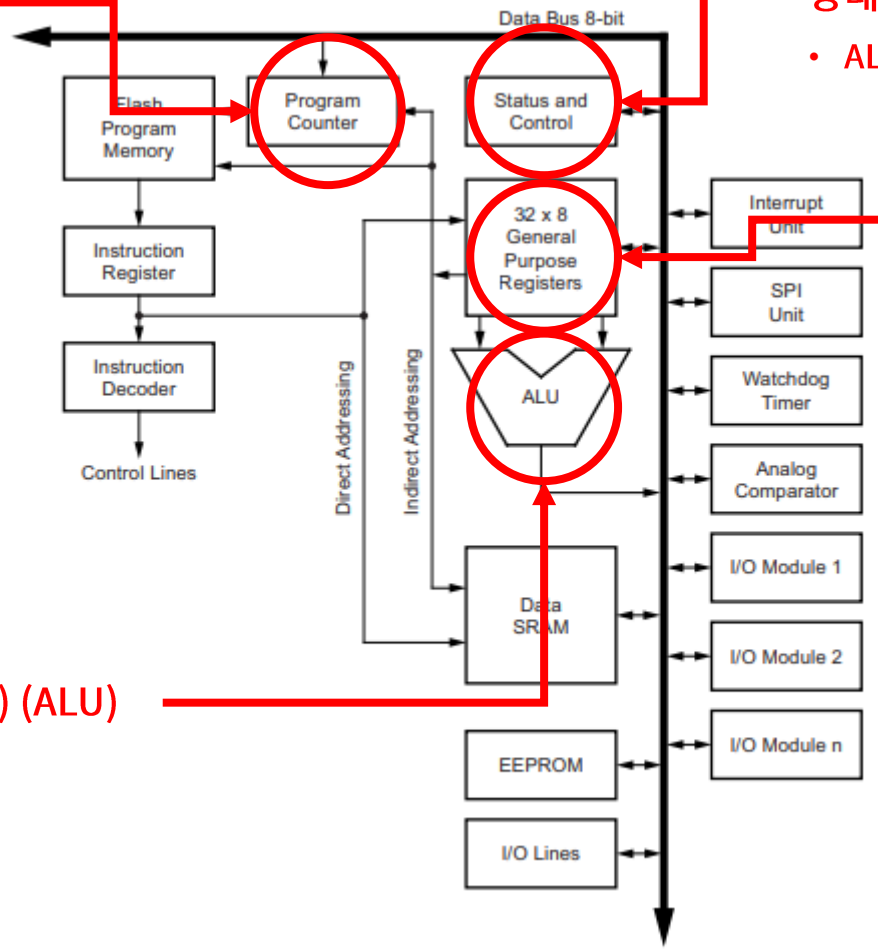
- 4칙연산
- 논리(AND, OR, NOT)
- 비트연산

데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

ATMEGA328P 기본 구조

프로그램카운터 (Program Counter : PC)

- 내가 실행시킨 명령어의 주소를 가리키는 역할



상태 레지스터 (Status Register : SREG)

- ALU가 가장 최근에 실행한 연산 명령의 결과와 상태를 표시

범용 레지스터 (General Purpose Register)

- 프로그램 수행 중에 중간 결과나 데이터를 일시적으로 저장하는데 사용

산술 논리 장치 (Arithmetic Logical Unit) (ALU)

- 4칙연산
- 논리(AND, OR, NOT)
- 비트연산

데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

ATMEGA328P 기본 구조

프로그램카운터 (Program Counter : PC)

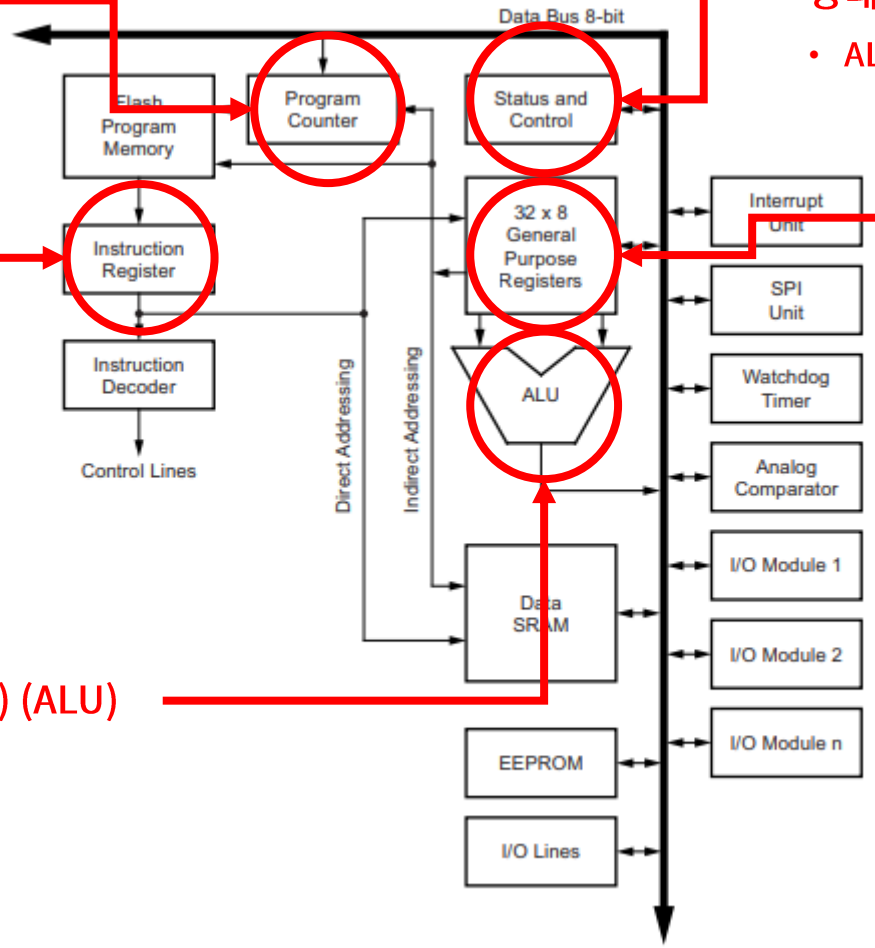
- 내가 실행시킨 명령어의 주소를 가리키는 역할

명령어 레지스터

- 실행 해야 하는 명령어를 저장

산술 논리 장치 (Arithmetic Logical Unit) (ALU)

- 4칙연산
- 논리(AND, OR, NOT)
- 비트연산



상태 레지스터 (Status Register : SREG)

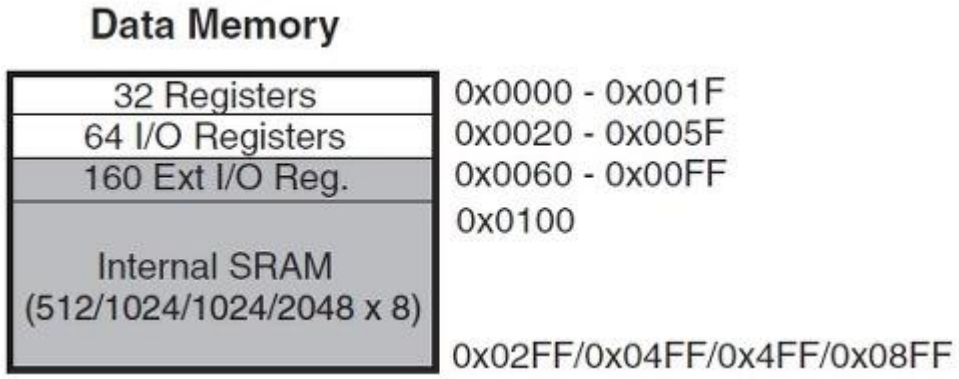
- ALU가 가장 최근에 실행한 연산 명령의 결과와 상태를 표시

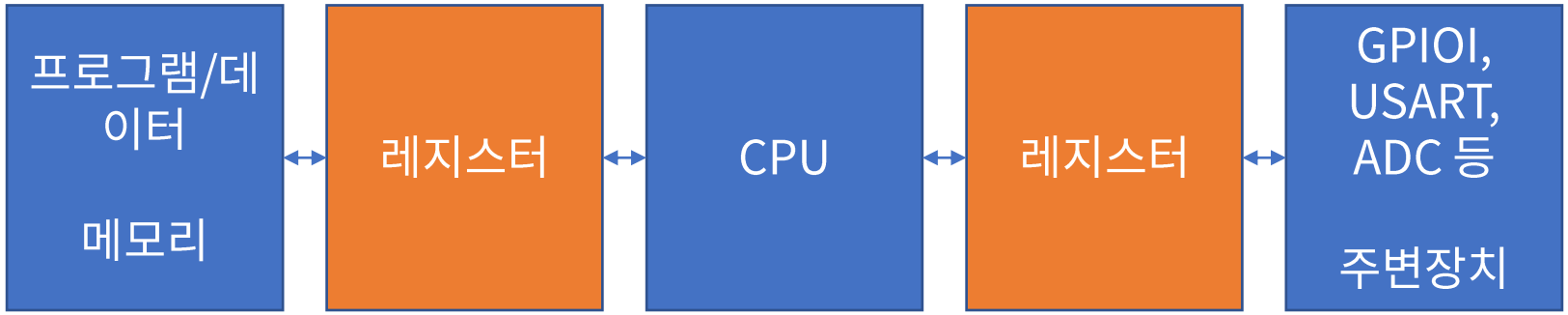
범용 레지스터 (General Purpose Register)

- 프로그램 수행 중에 중간 결과나 데이터를 일시적으로 저장하는데 사용

데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

ATMEGA328P의 메모리맵과 레지스터





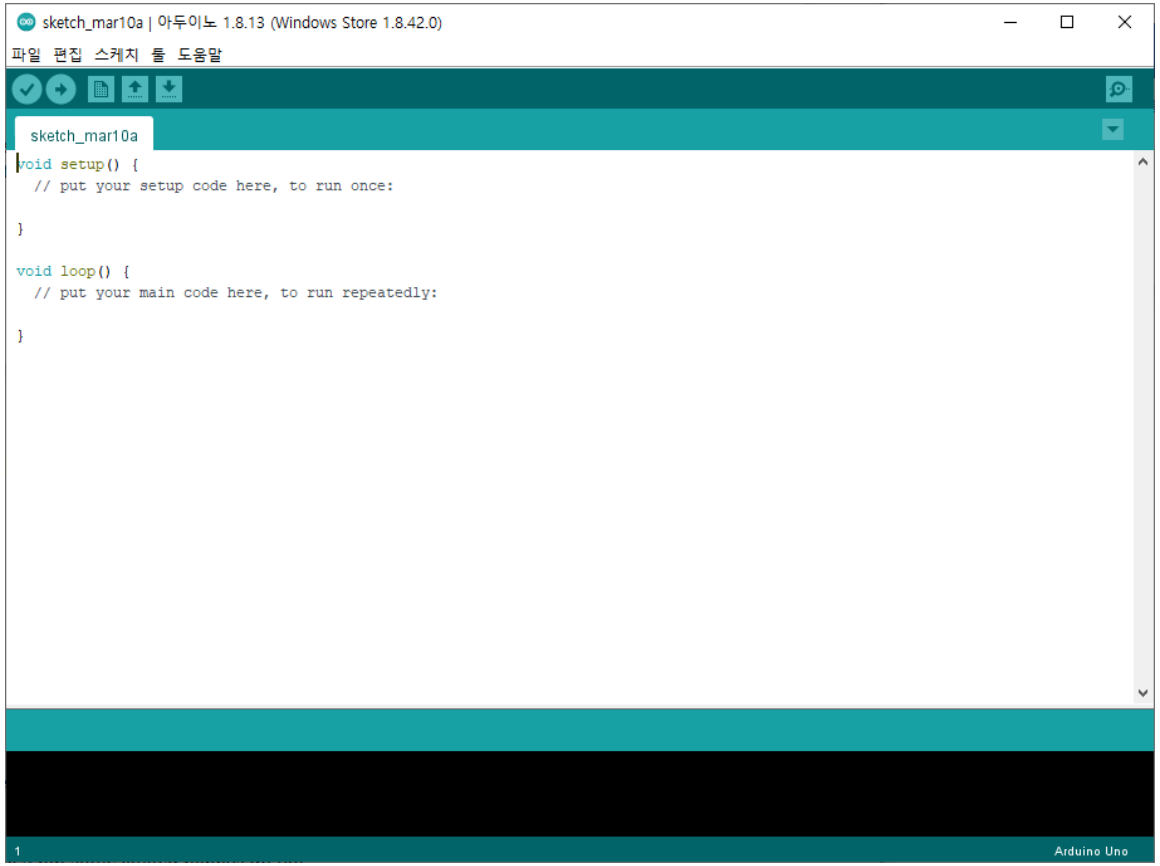
데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
30. Register Summary, 275 page 참고

아두이노 개발 환경 구성

- 아두이노 IDE를 이용
  - 홈페이지 : https://www.arduino.cc/
  - 다운로드 : https://www.Arduino.cc/en/software

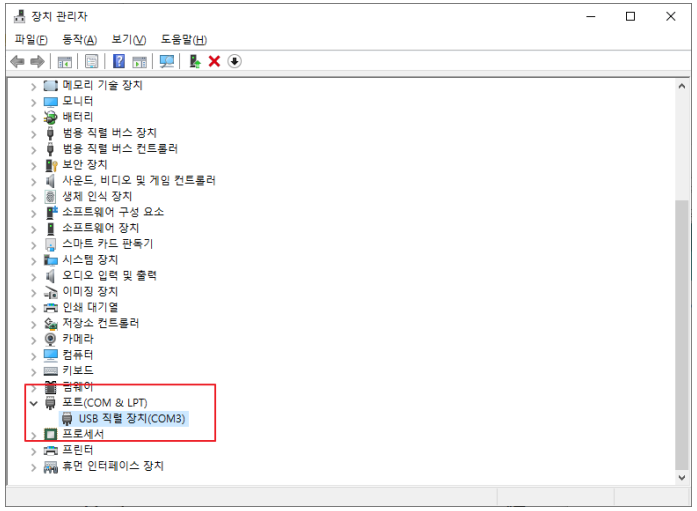
아두이노 개발 환경 구성

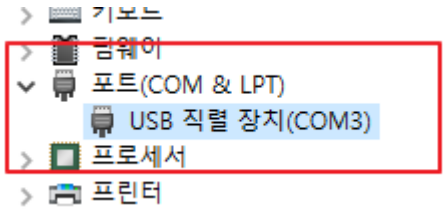
- 아두이노 IDE 실행



아두이노 개발 환경 구성

- 컴퓨터 USB에 아두이노를 연결
  - 장치관리자에서 아두이노가 연결 되어있는지 확인
  - 아두이노는 컴퓨터와 시리얼통신으로 연결 됨. 아래와 같이 PC에 가상의 시리얼포트가 생성 되었다면 올바로 연결
  - 시리얼 통신 포트 확인(기억해 두세요)

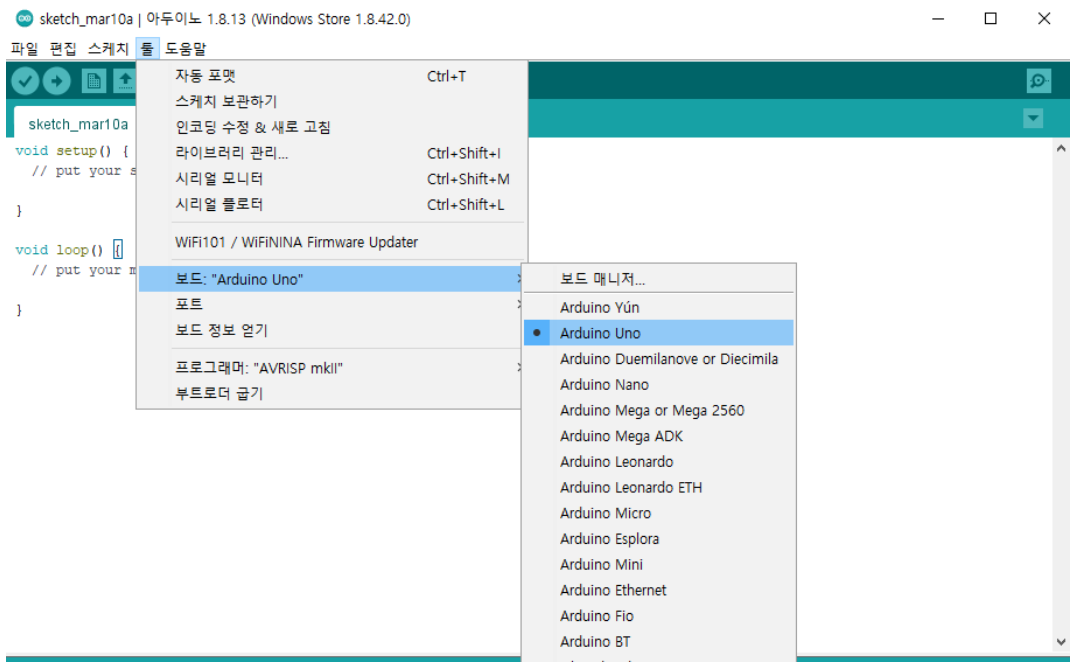




COM3

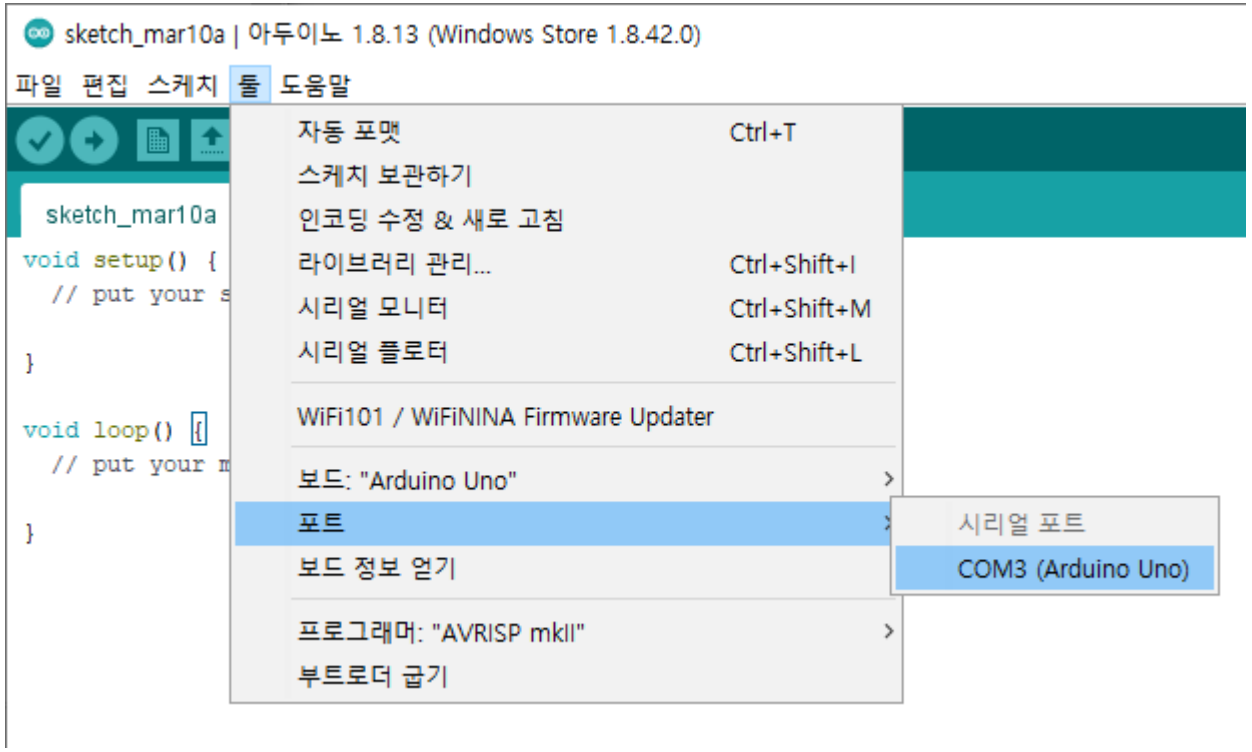
아두이노 개발 환경 구성

- 아두이노 IDE에서 테스트 보드 선택
  - ARDUINO UNO
  - 메뉴 → 툴 → 보드 → Arduino Uno 선택



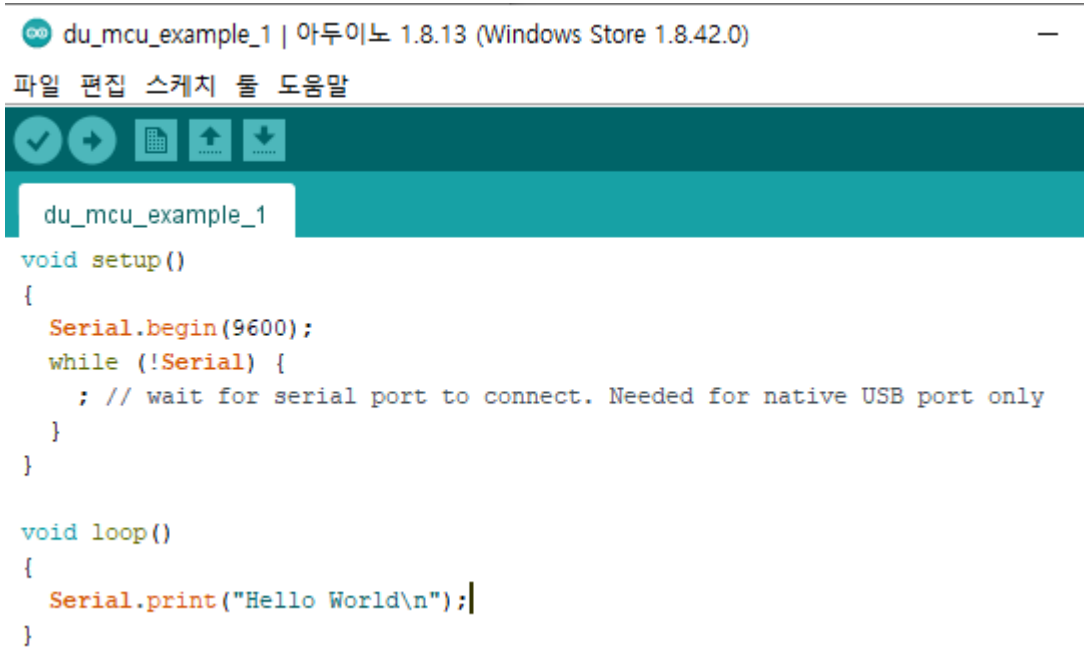
아두이노 개발 환경 구성

- 아두이노 IDE에서 테스트 보드와의 통신 포트 선택
  - 메뉴 → 툴 → 포트 → COM3



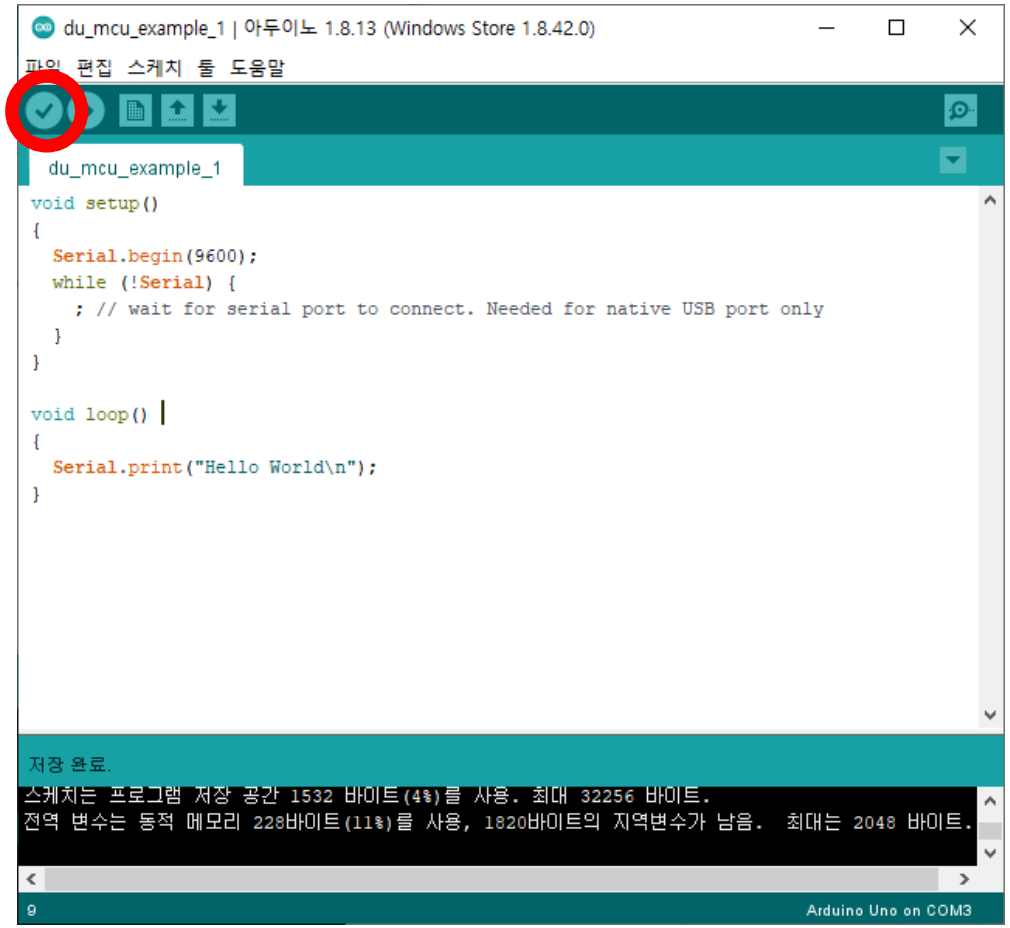
테스트 코드 실험

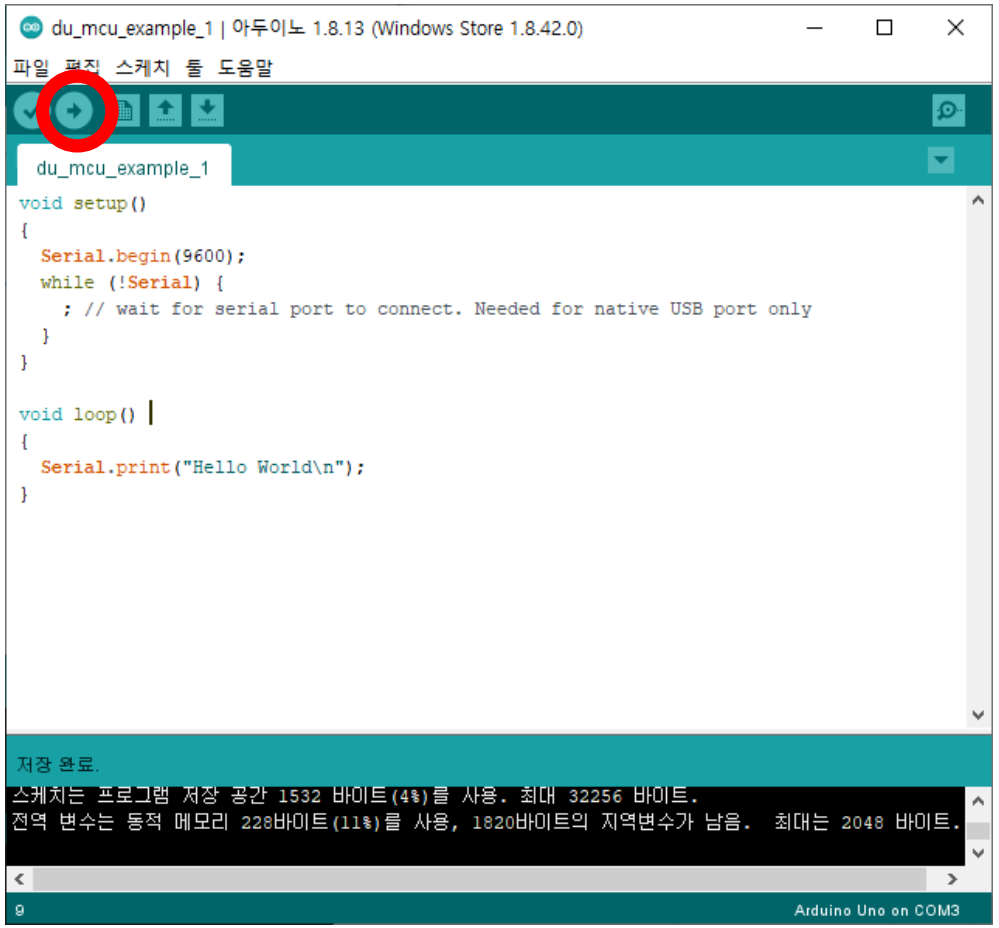
```
void setup()
{
        Serial.begin(9600);
}

void loop()
{
        Serial.print("Hello World\n");
}
```



테스트 코드 실험

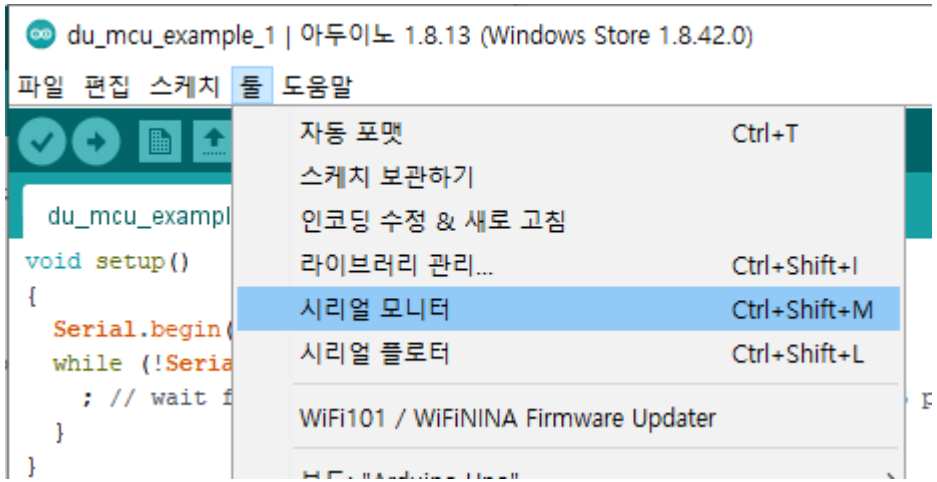
- 컴파일 & 업로드

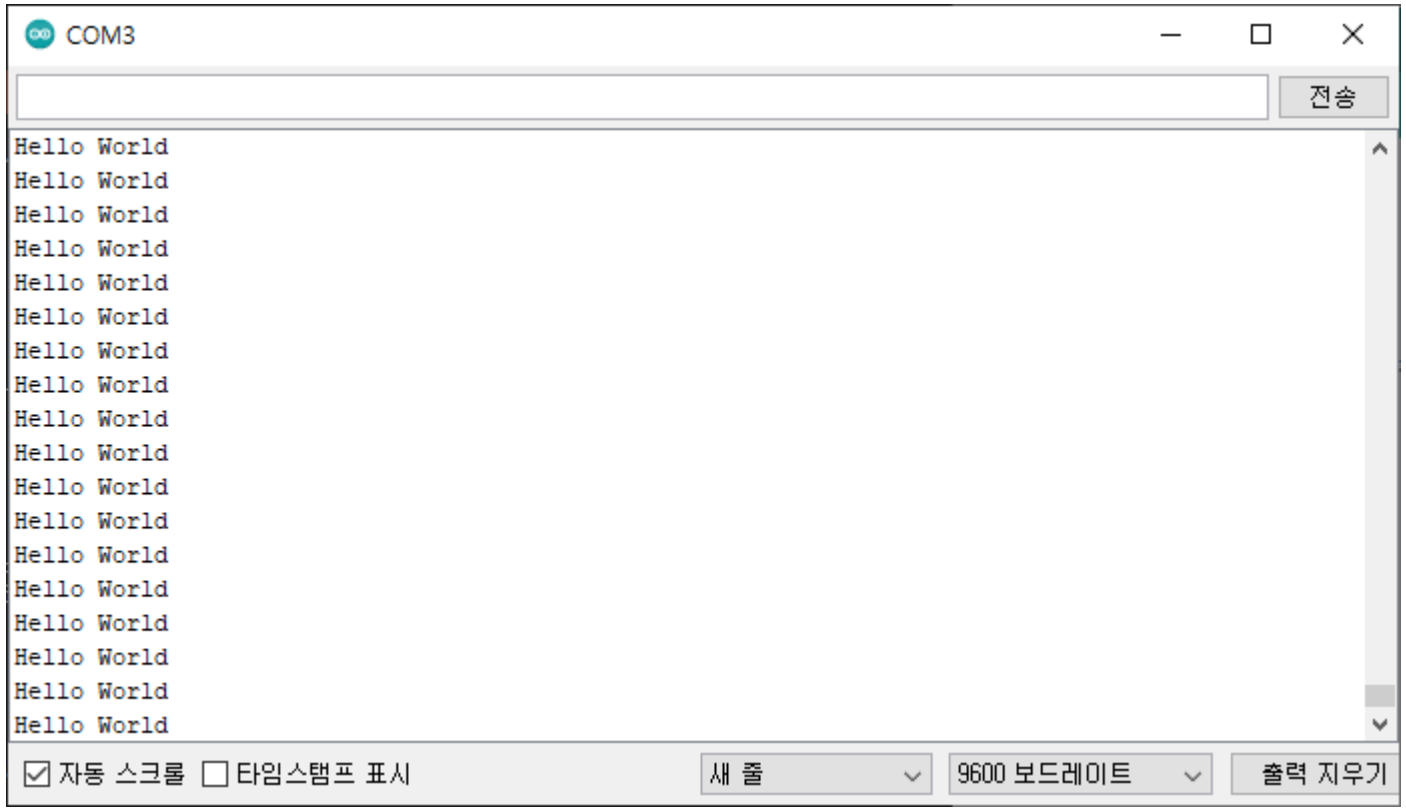




테스트 코드 실험

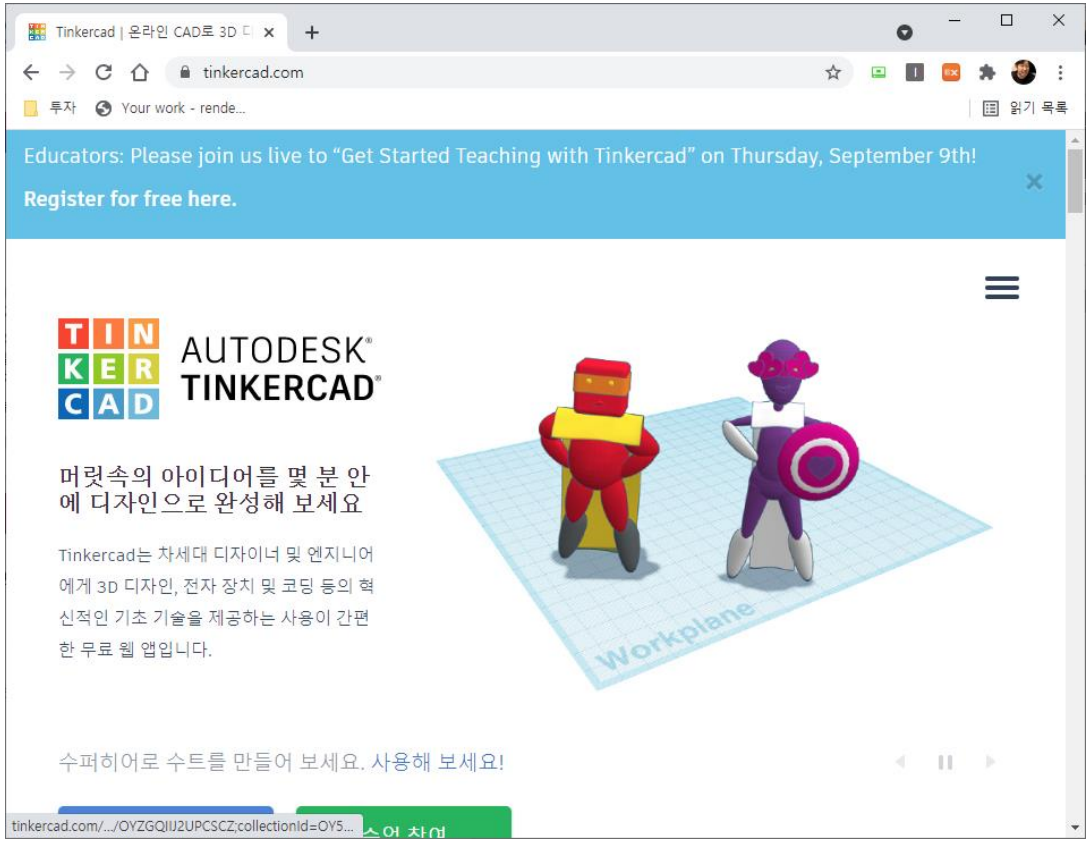
- 시리얼 통신 확인

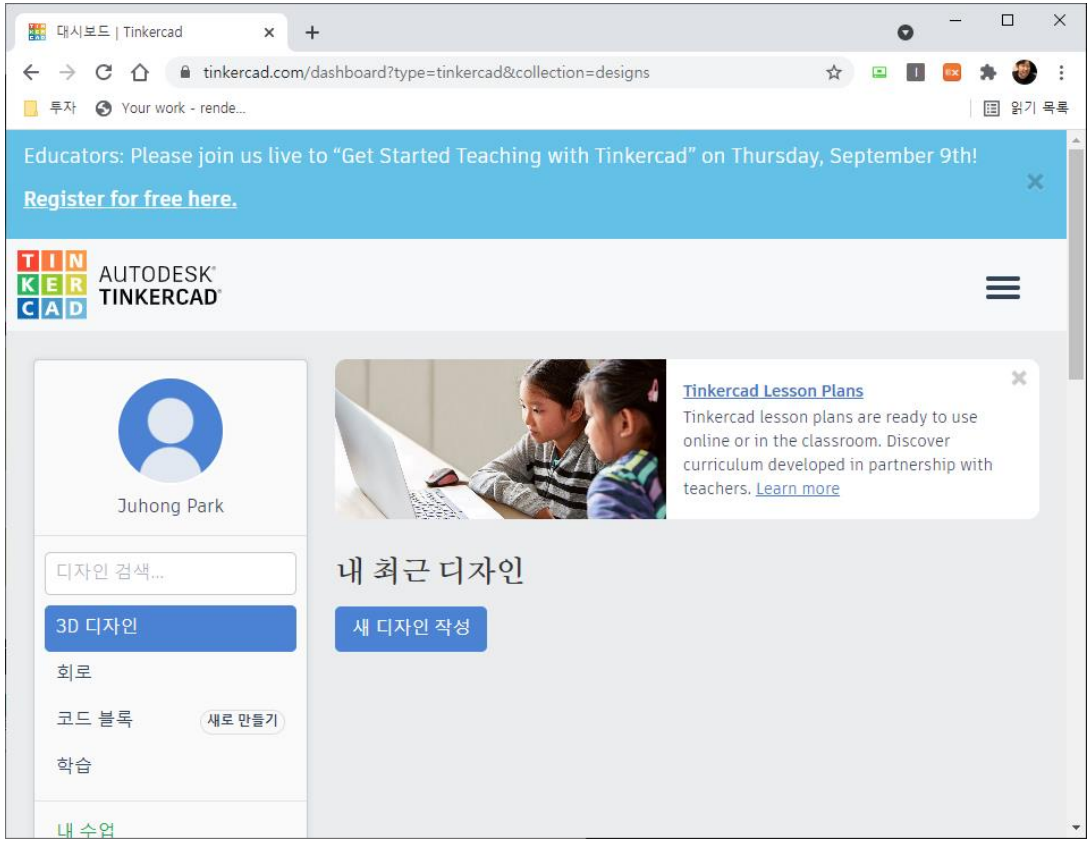




Tinkercad를 활용한 아두이노 시뮬레이션 실험

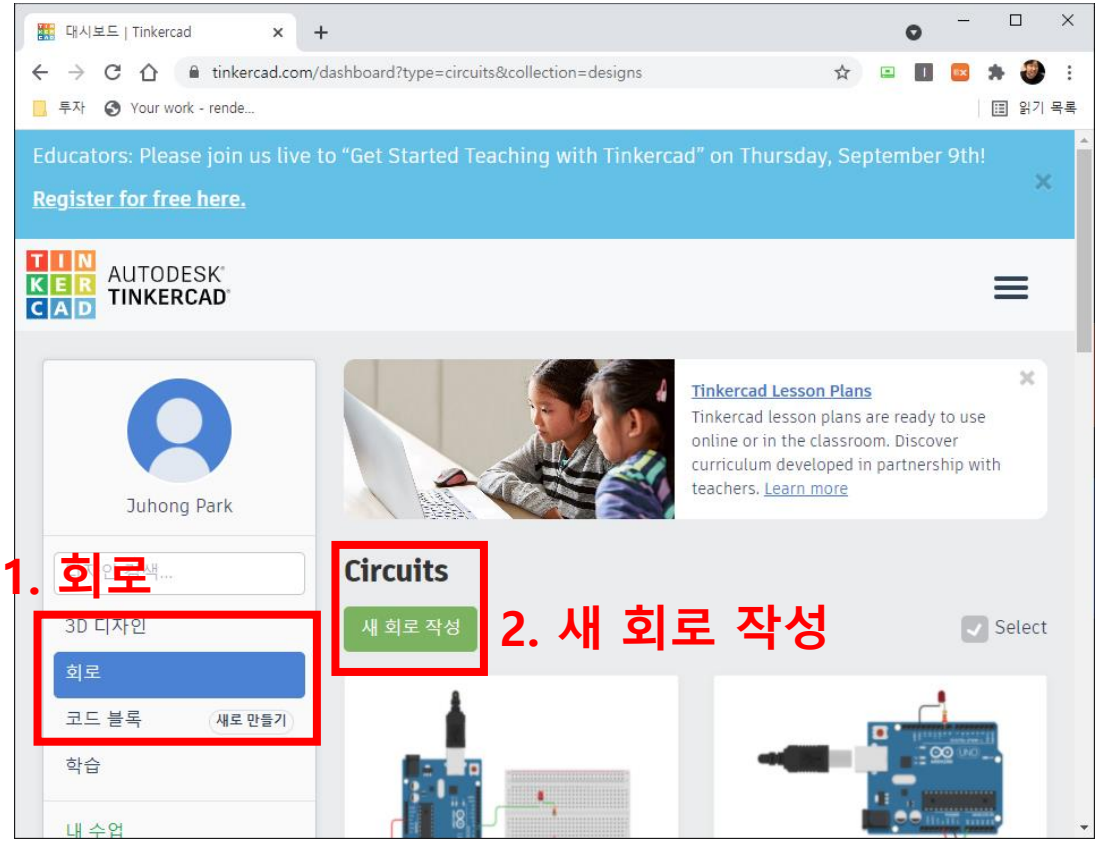
- tinkercad.com

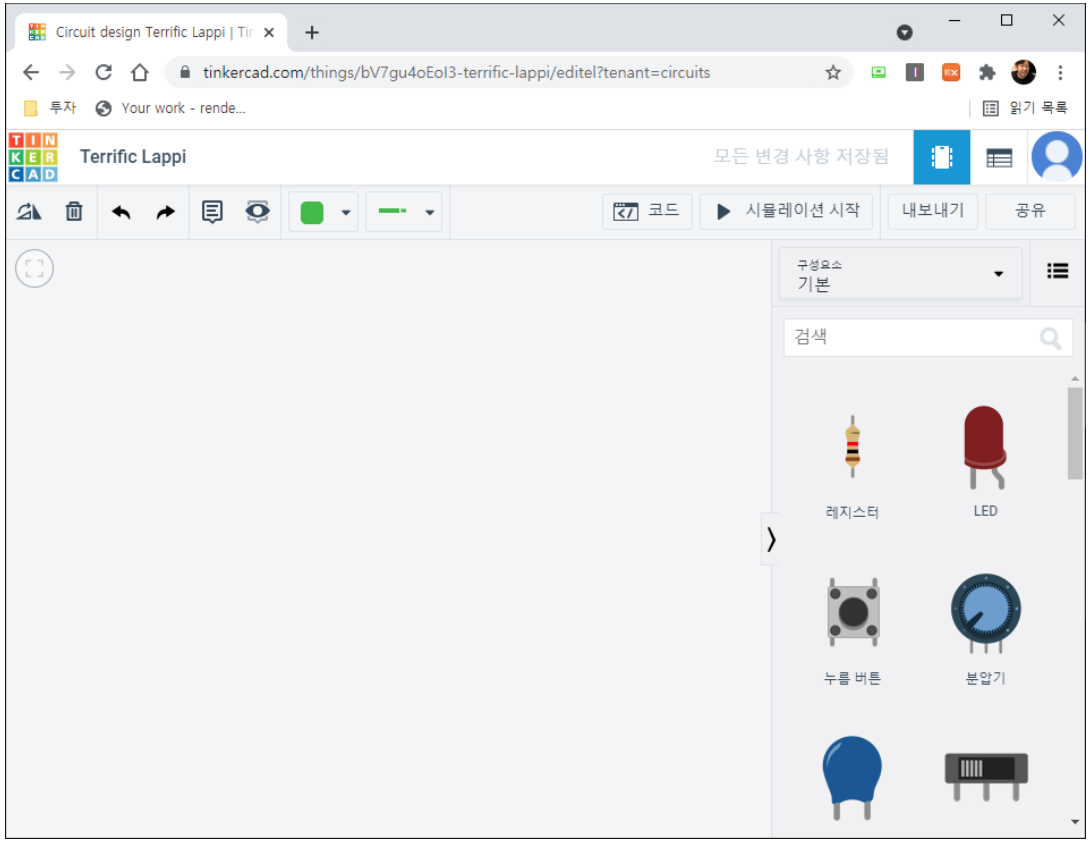




Tinkercad를 활용한 아두이노 시뮬레이션 실험

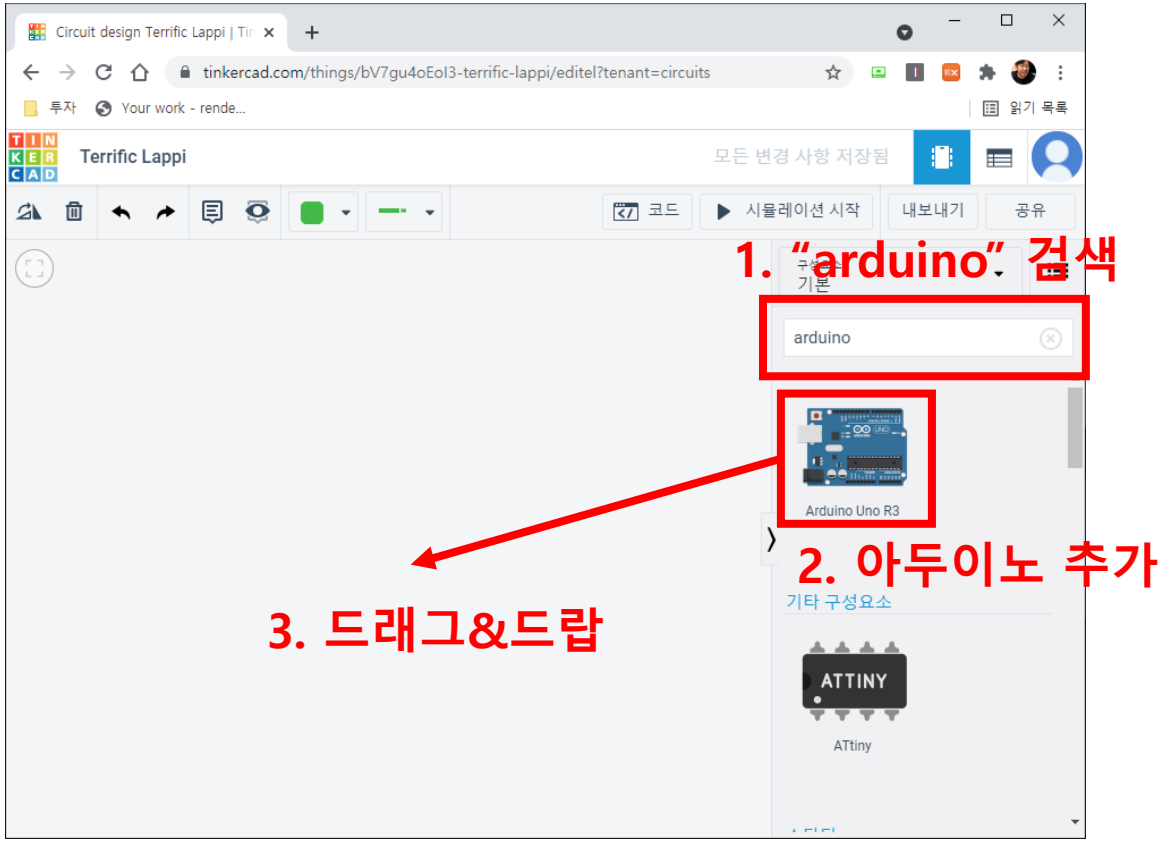
- tinkercad.com : 새 회로 작성

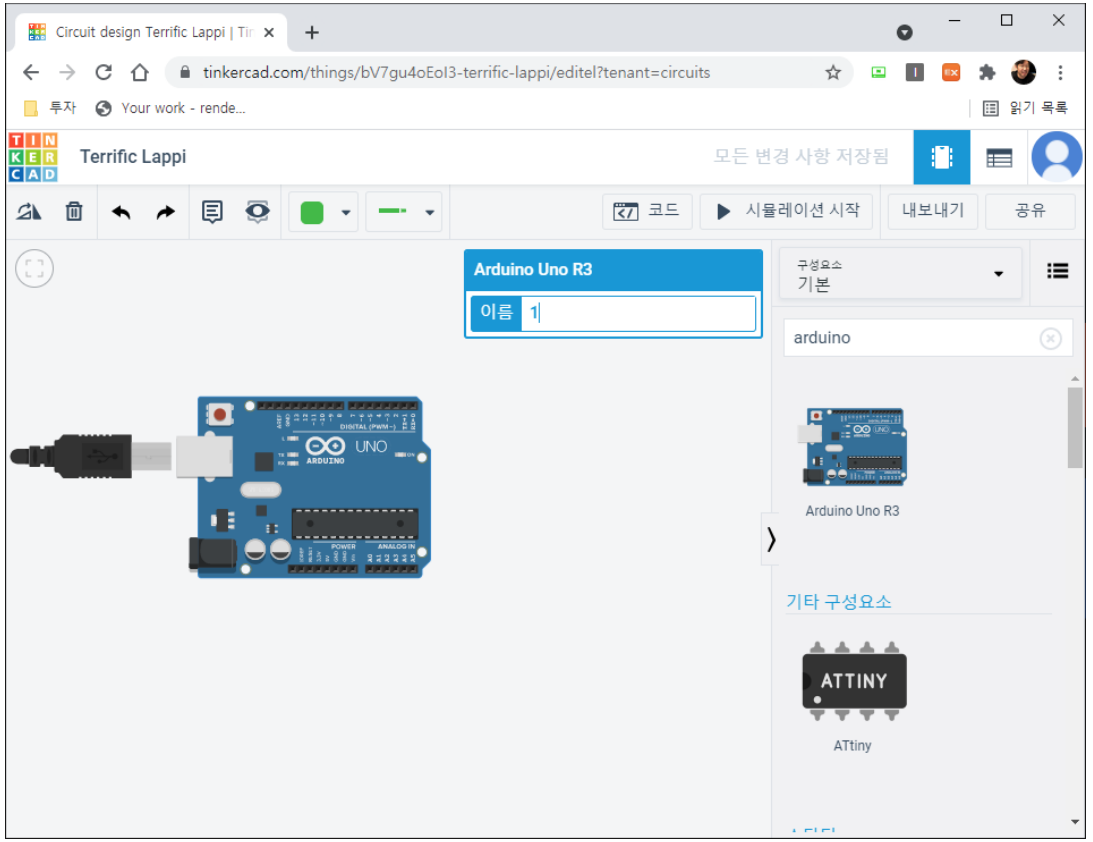




Tinkercad를 활용한 아두이노 시뮬레이션 실험

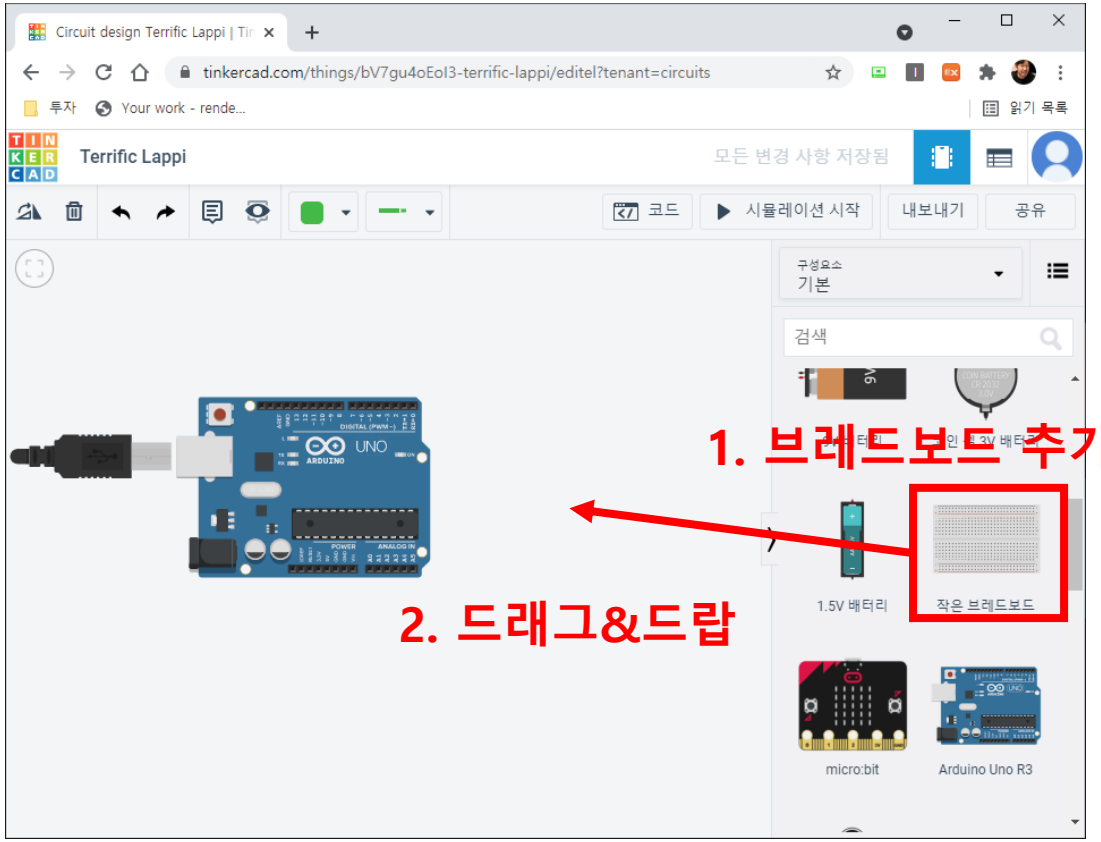
- tinkercad.com : 아두이노 추가

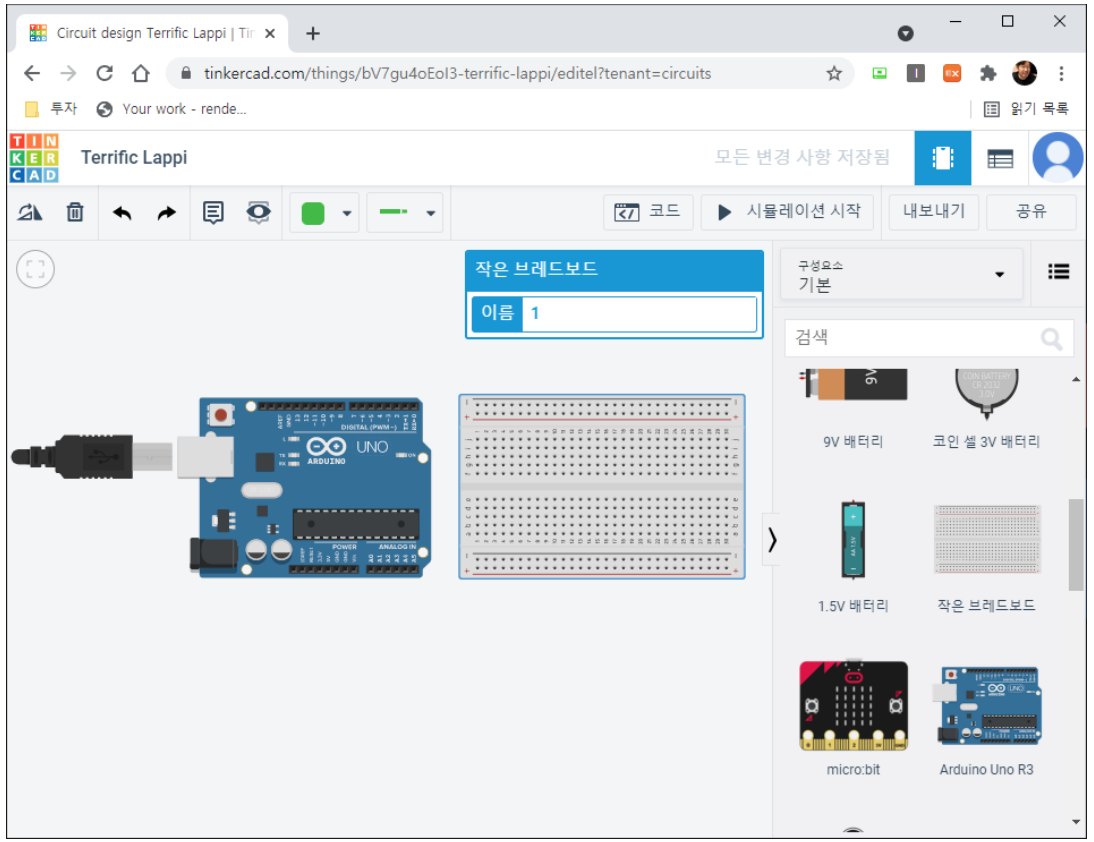




Tinkercad를 활용한 아두이노 시뮬레이션 실험

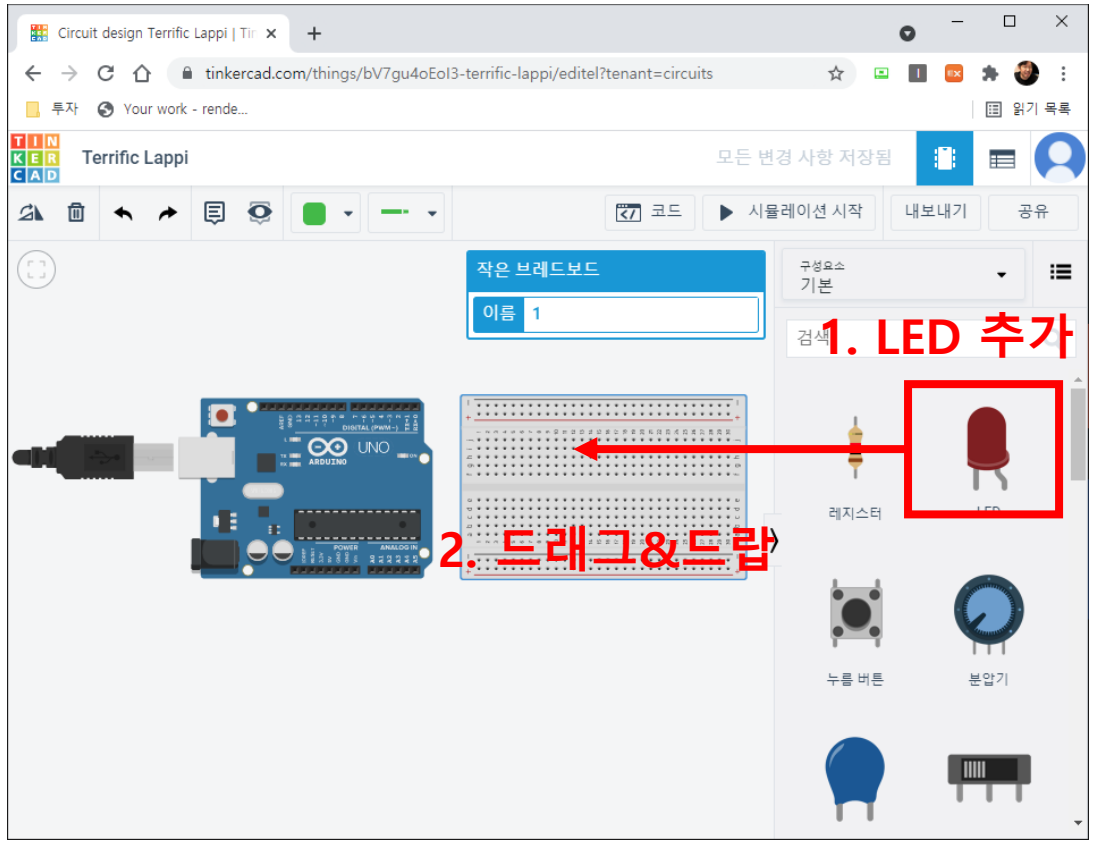
- tinkercad.com : 브레드보드 추가

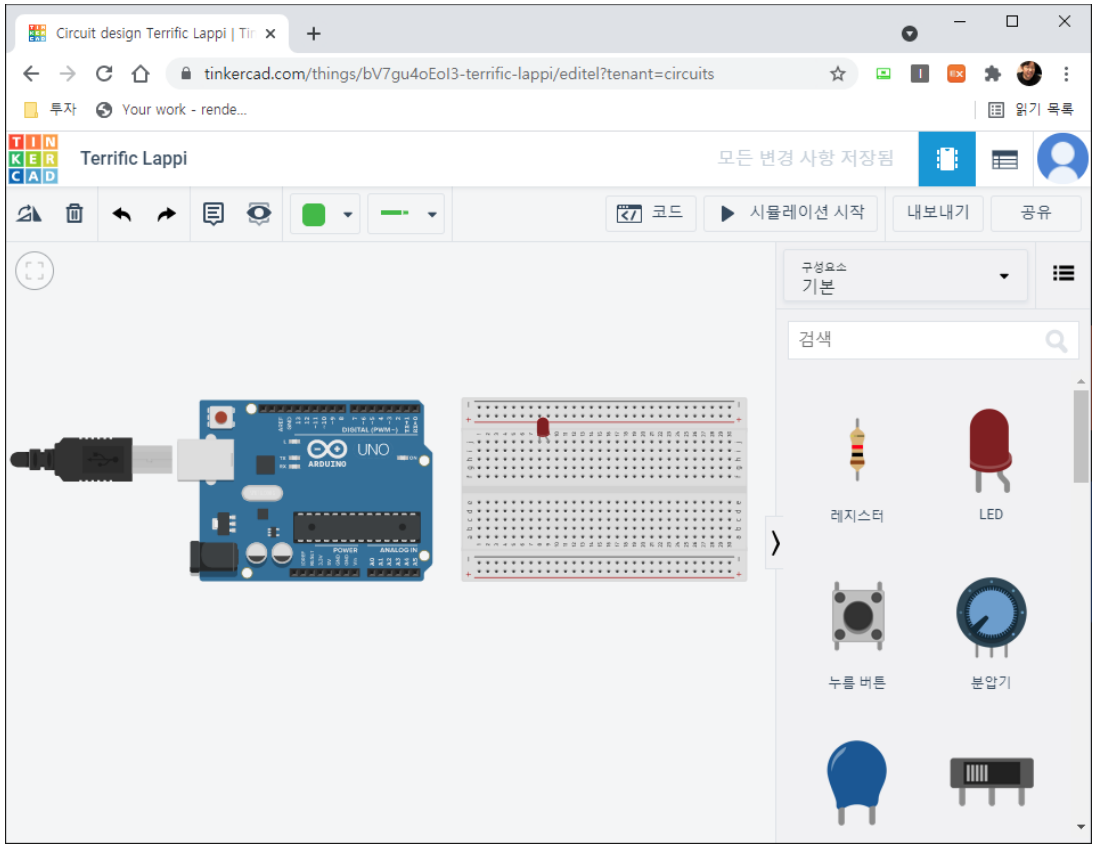




Tinkercad를 활용한 아두이노 시뮬레이션 실험

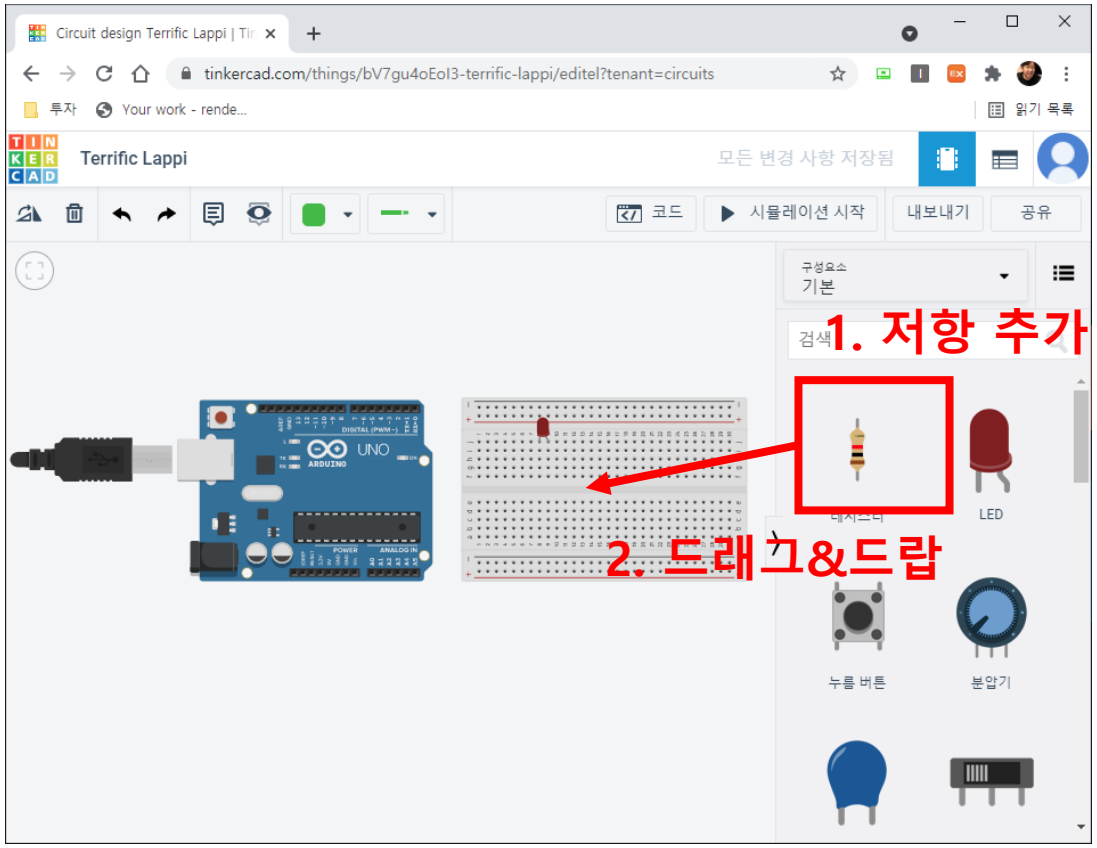
- tinkercad.com : LED 추가

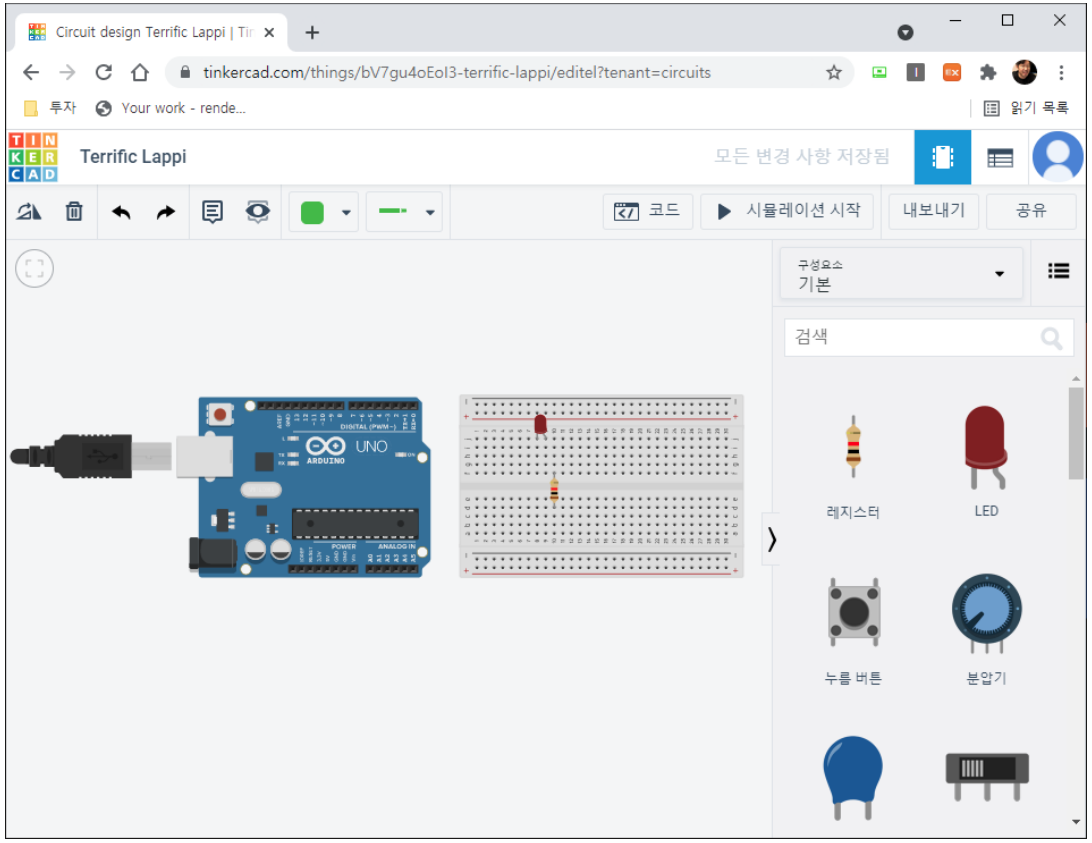




Tinkercad를 활용한 아두이노 시뮬레이션 실험

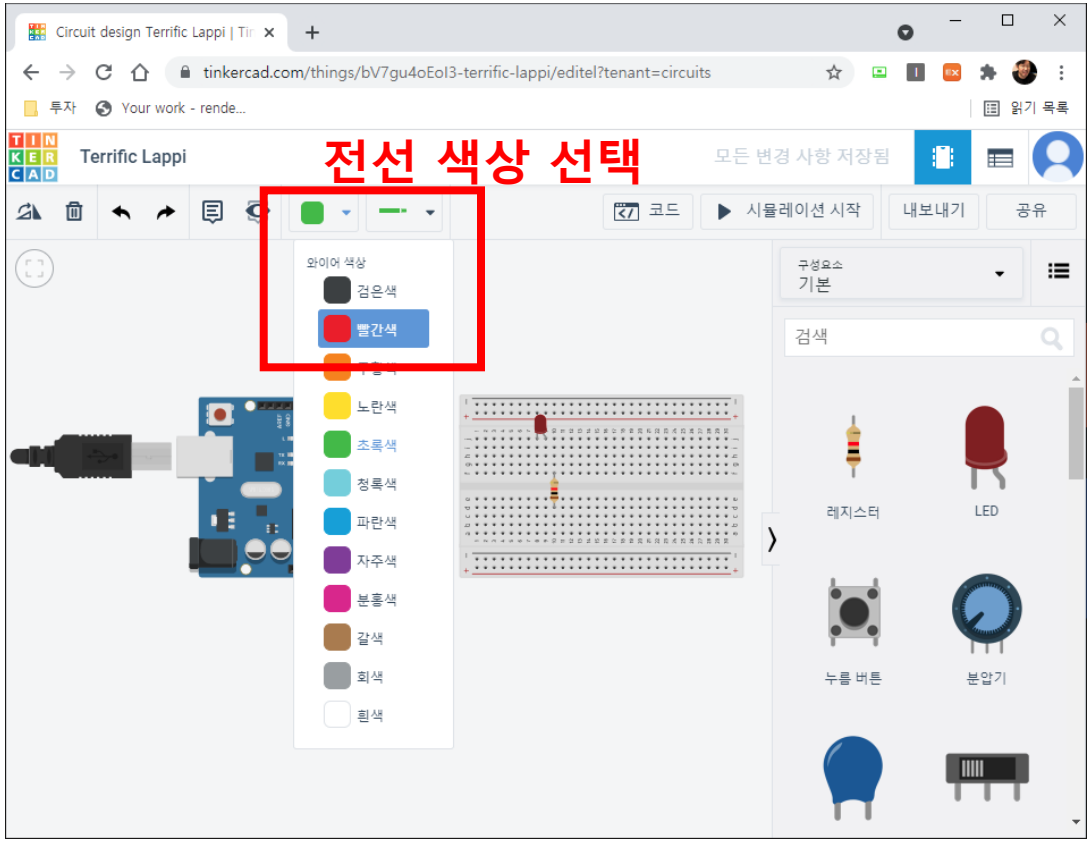
- tinkercad.com : 저항 추가

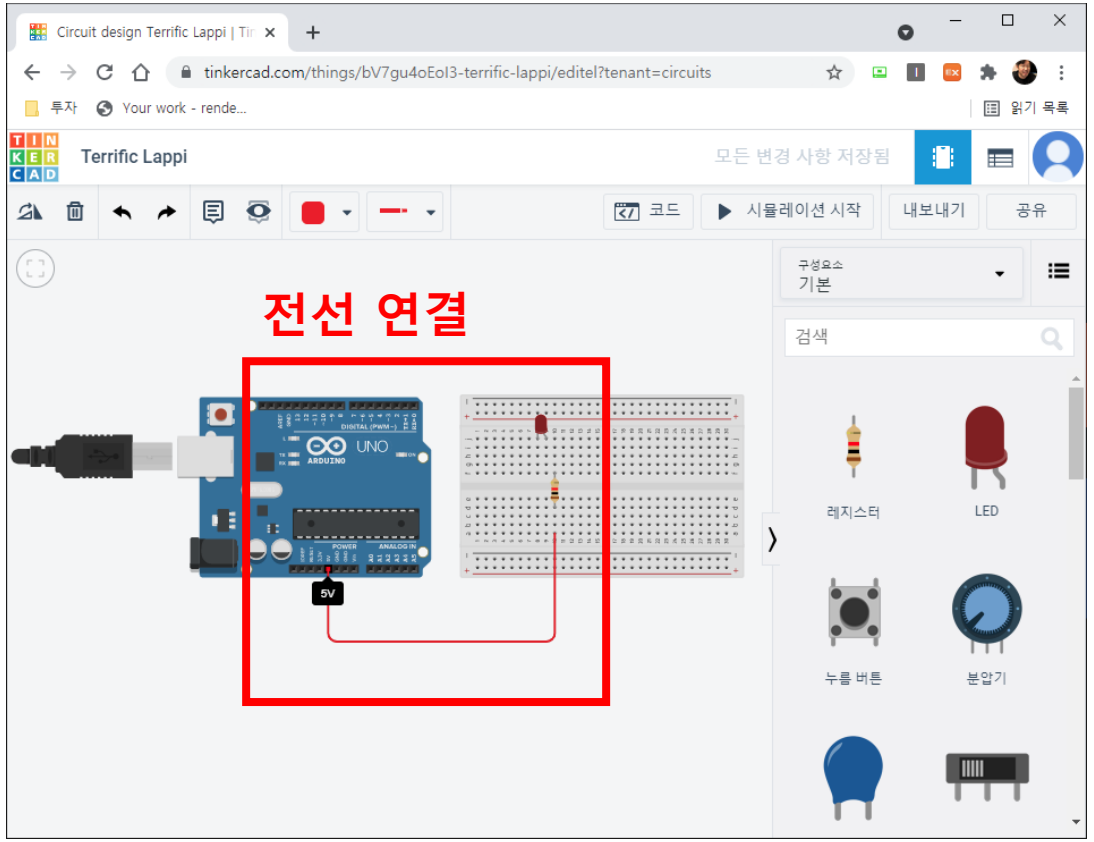




Tinkercad를 활용한 아두이노 시뮬레이션 실험

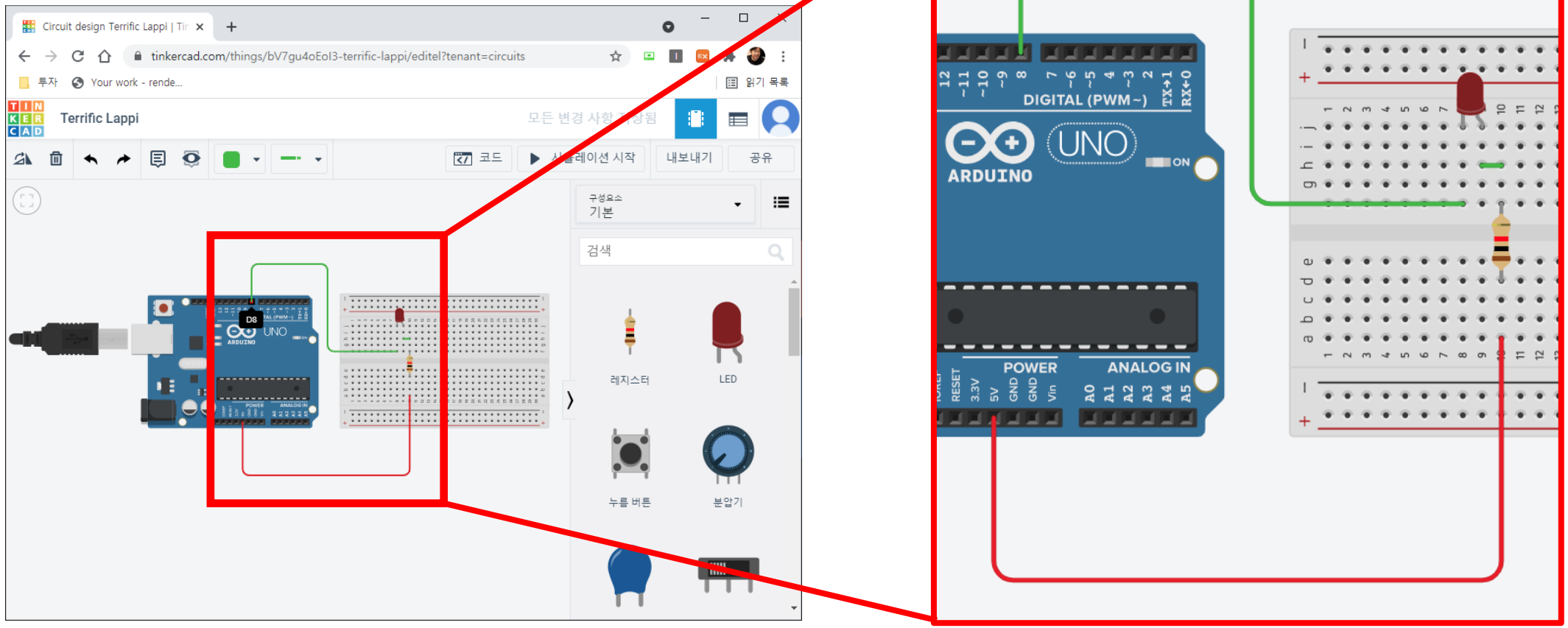
- tinkercad.com : 5V(+파워 연결)





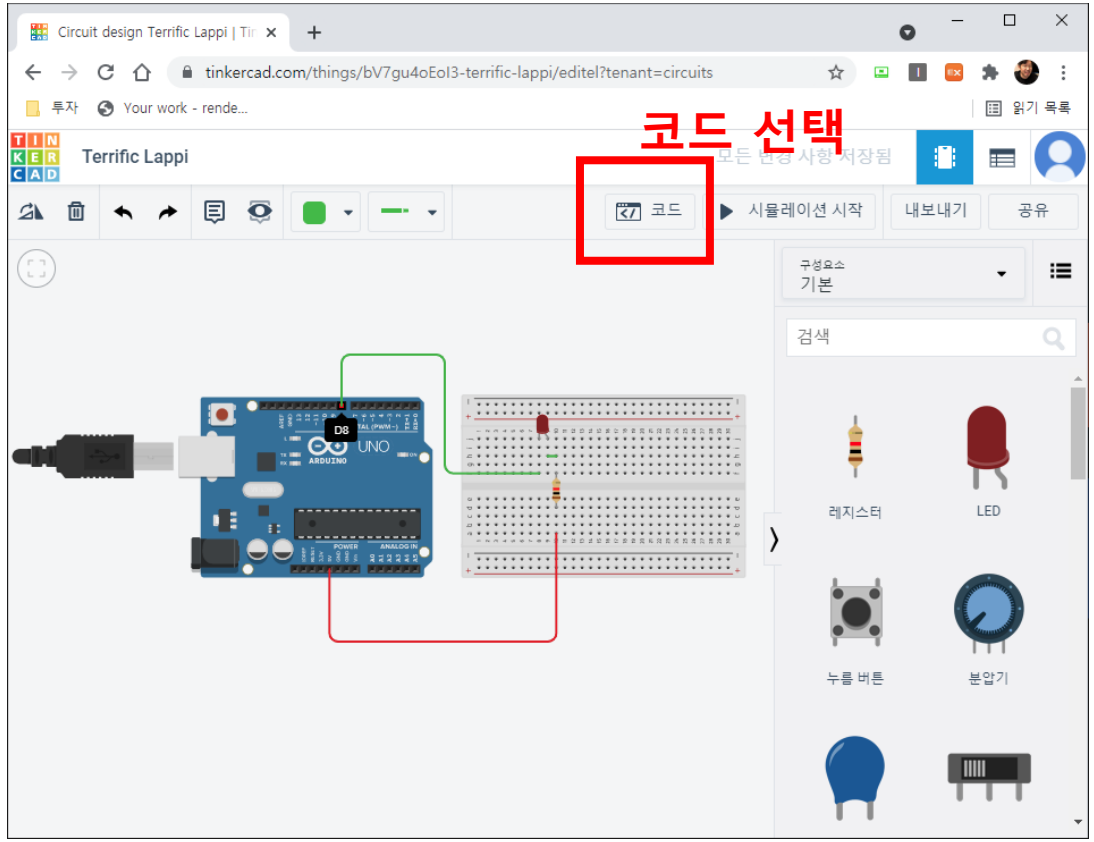
Tinkercad를 활용한 아두이노 시뮬레이션 실험

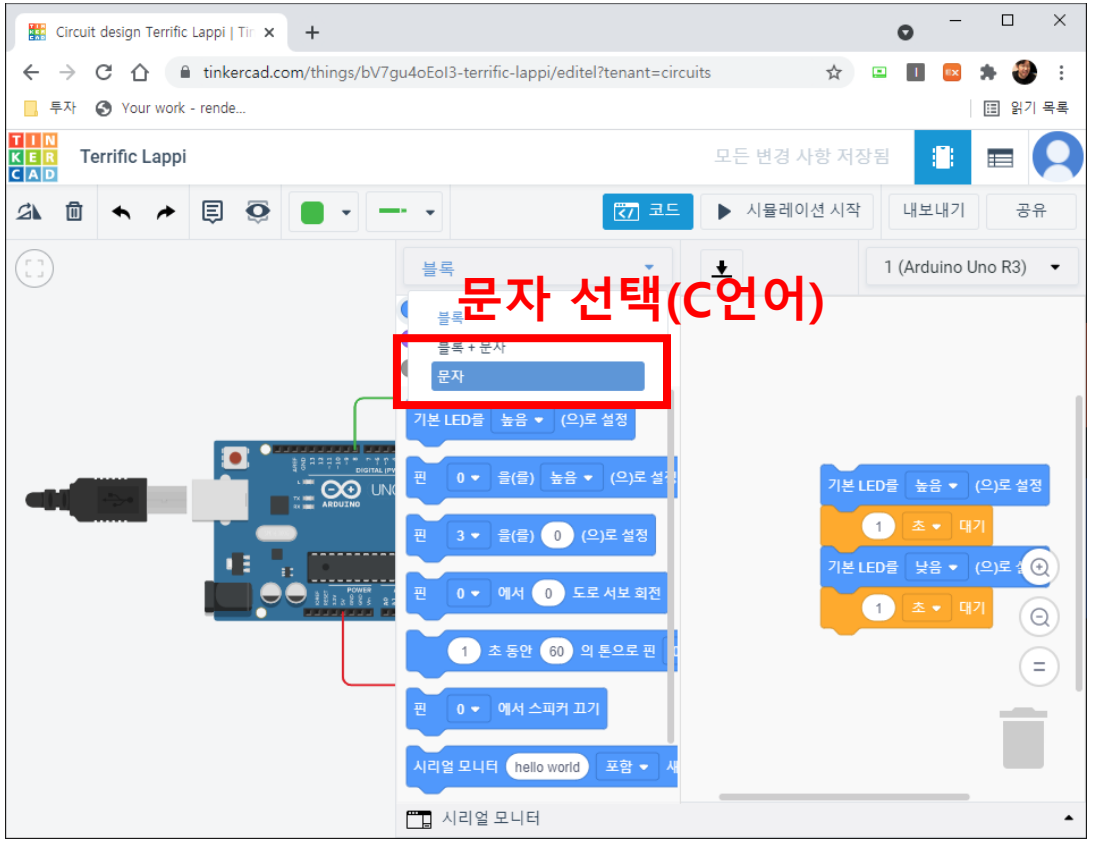
- tinkercad.com : 전체 회로 구성



Tinkercad를 활용한 아두이노 시뮬레이션 실험

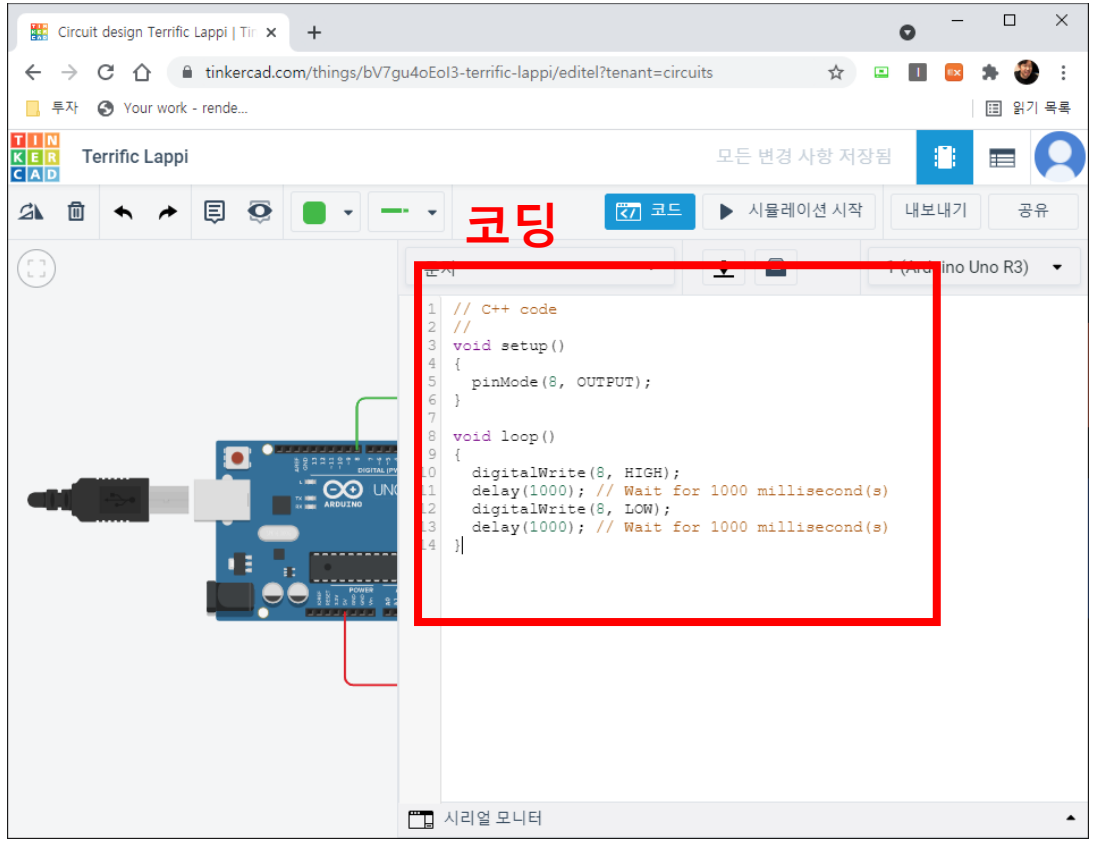
- tinkercad.com : 코드 작성





Tinkercad를 활용한 아두이노 시뮬레이션 실험

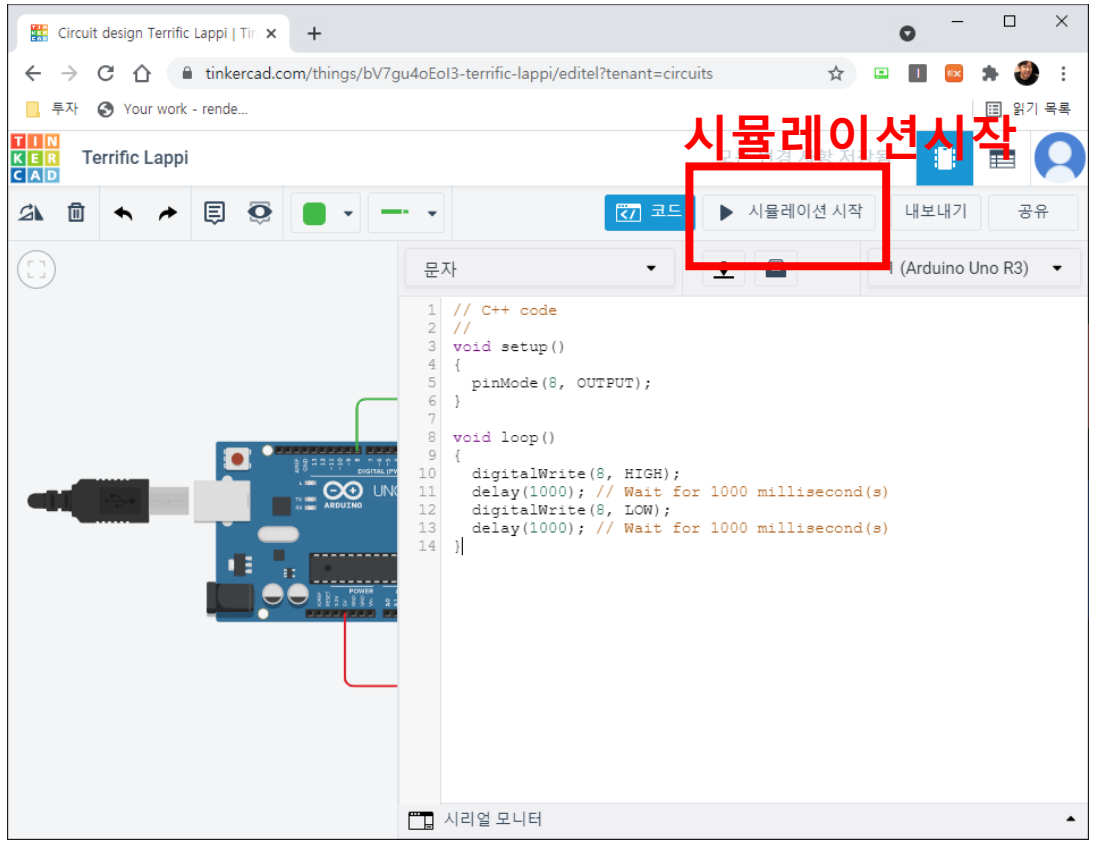
- tinkercad.com : 코드 작성

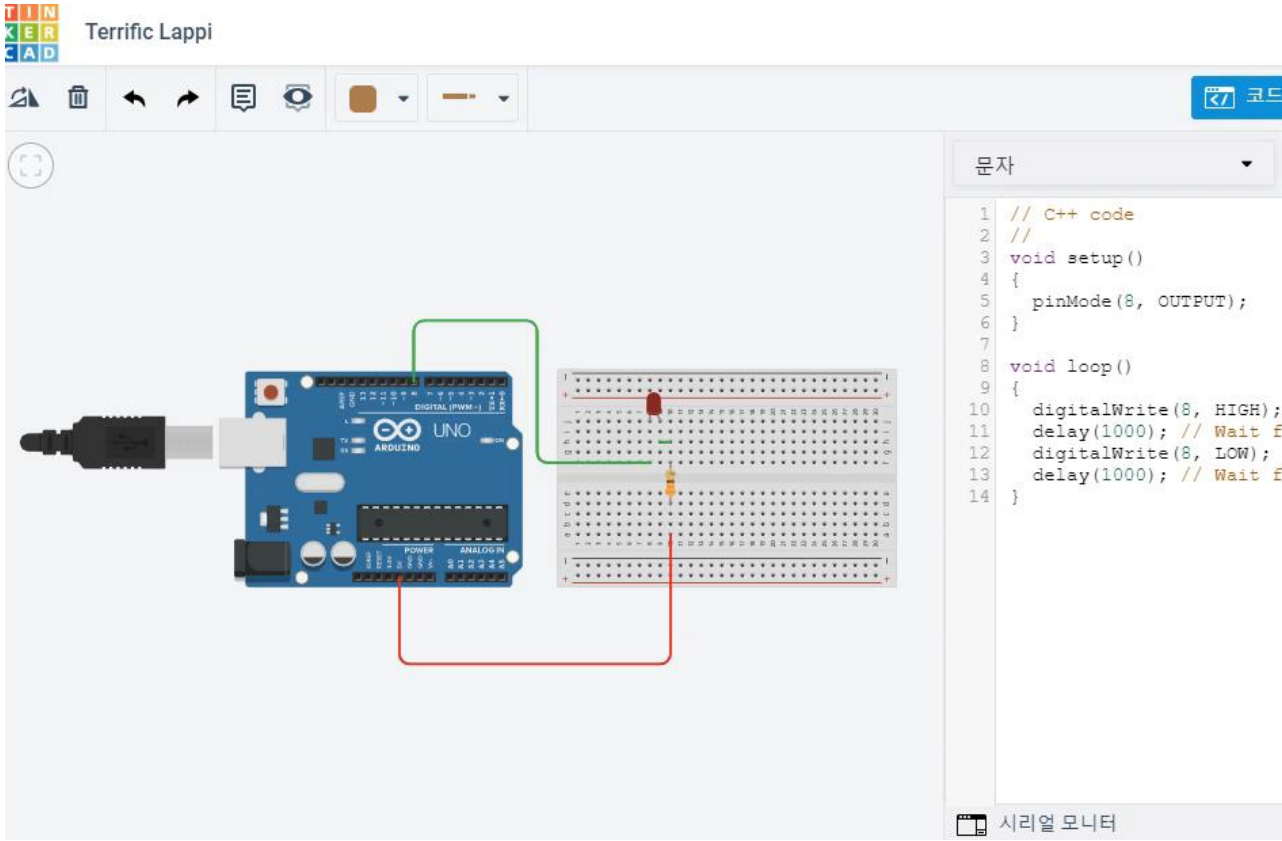



```
// C++ code
//
void setup()
{
  pinMode(8, OUTPUT);
}

void loop()
{
  digitalWrite(8, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(8, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

Tinkercad를 활용한 아두이노 시뮬레이션 실험

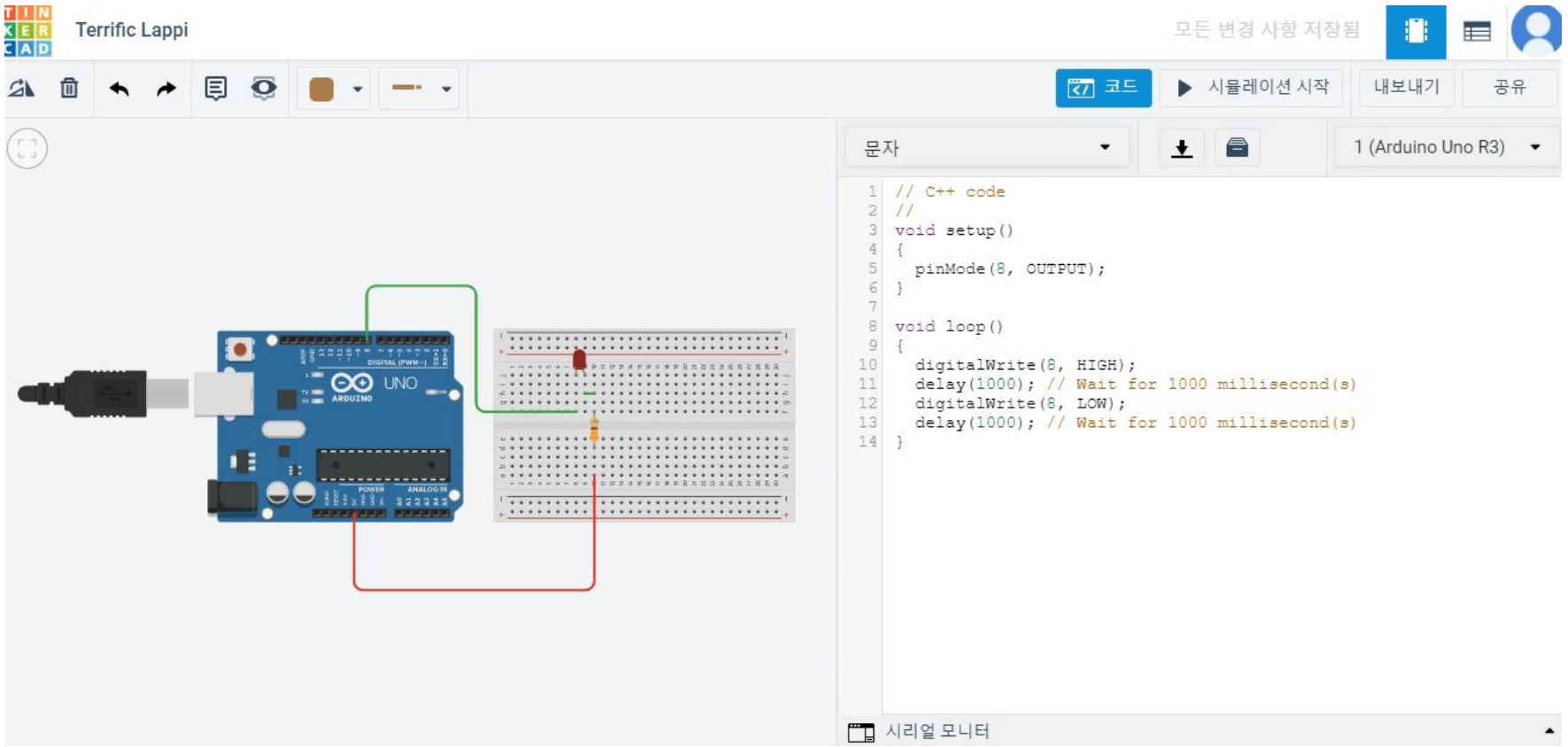
- tinkercad.com : 시뮬레이션 시작





Tinkercad를 활용한 아두이노 시뮬레이션 실험

- tinkercad.com : 시뮬레이션 시작



수고하셨습니다.

다음주에 만나요.