마이크로프로세서 종합설계

2021년 봄

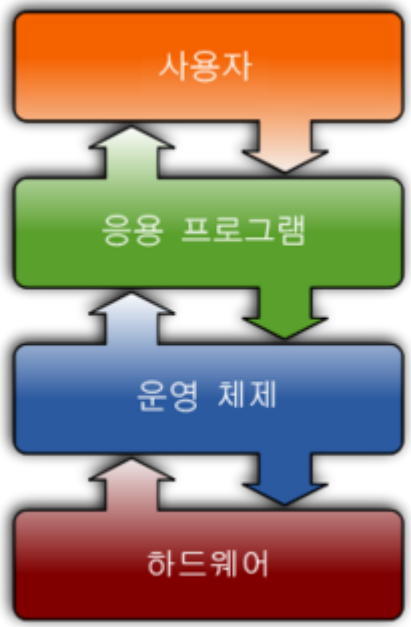
반갑습니다.

1주차

마이크로프로세서?

- **PC의 HW 구성품**
  - CPU
  - RAM
  - 하드디스크
  - 그래픽카드
  - USB장치
  - 네트워크장치
  - 모니터
  - 키보드
  - 마우스
  - 등

- **PC의 SW 구성품**
  - 메신저(카카오톡)
  - 게임
  - 넷플릭스, 유튜브등
  - 파워포인트등
  - ----------
  - **OS ★**
  - BIOS
  - Device Driver



* MCU Micro Compting Unit
* CPU Computing Processing Unit
* RAM : Random Access Memory
* ROM : Read Only Memory

* OS : Operating System
* BIOS : Basic Input Output System

마이크로프로세서?

- **MCU의 HW 구성품**
  - **CPU**
  - **RAM**
  - ~~하드디스크~~ **ROM**
  - ~~그래픽카드~~
  - ~~USB장치~~
  - ~~네트워크장치~~
  - ~~모니터~~
  - ~~키보드~~
  - ~~마우스~~
  - ~~등~~

- **MCU의 SW 구성품**
  - ~~메신저(카카오톡)~~
  - ~~게임~~
  - ~~넷플릭스, 유튜브등~~
  - ~~파워포인트등~~
  - ---------
  - ~~OS~~
  - ~~**BIOS**~~
  - ~~Device Driver~~

* MCU Micro Compting Unit
* CPU Computing Processing Unit
* RAM : Random Access Memory
* ROM : Read Only Memory

* OS : Operating System
* BIOS : Basic Input Output System

마이크로프로세서를 왜?

마이크로프로세서를 왜?

마이크로프로세서의 한계는?

하나의 일을 수행하기 바쁘다.

개발자의 책임이 100%이다. 프로그램을 잘 못 만들면?

마이크로프로세서의 조합이 오히려 시스템을 망친다.

마이크로프로세서를 사용하려면 무엇을 알아야 할까?

불행히도 프로그래밍 언어는 반드시 알아야 한다.

추천 : C언어

우선 종이위에 "순서도"로 코딩을 먼저 시작하자.

수업의 목표

- 마이크로 프로세서를 공부하면서 컴퓨터의 구성을 익힌다.
- 마이크로 프로세서가 올바르게 동작하는 코드를 작성한다.
- 마이크로 프로세서에 포함된 기본 기능을 익힌다.
- 마이크로 프로세서에 다양한 외부 장치를 연결한다.

그러다 보면 마이크로프로세서로 로봇 제작도 가능하다.

평가

- 출석 : 20
- 레포트 : 10
- 중간고사(시험) : 35
- 기말고사(시험, 프로젝트 및 발표) : 35
  - 시험 : 15
  - 2주 프로젝트 수행 : 20

숙제

- github 가입
- notion을 이용하여 이력서, 포트폴리오 공유 페이지 만들기
  - 참고 : https://www.notion.so/Leo-Osa-9ac01881647f410194ead70b790aed98

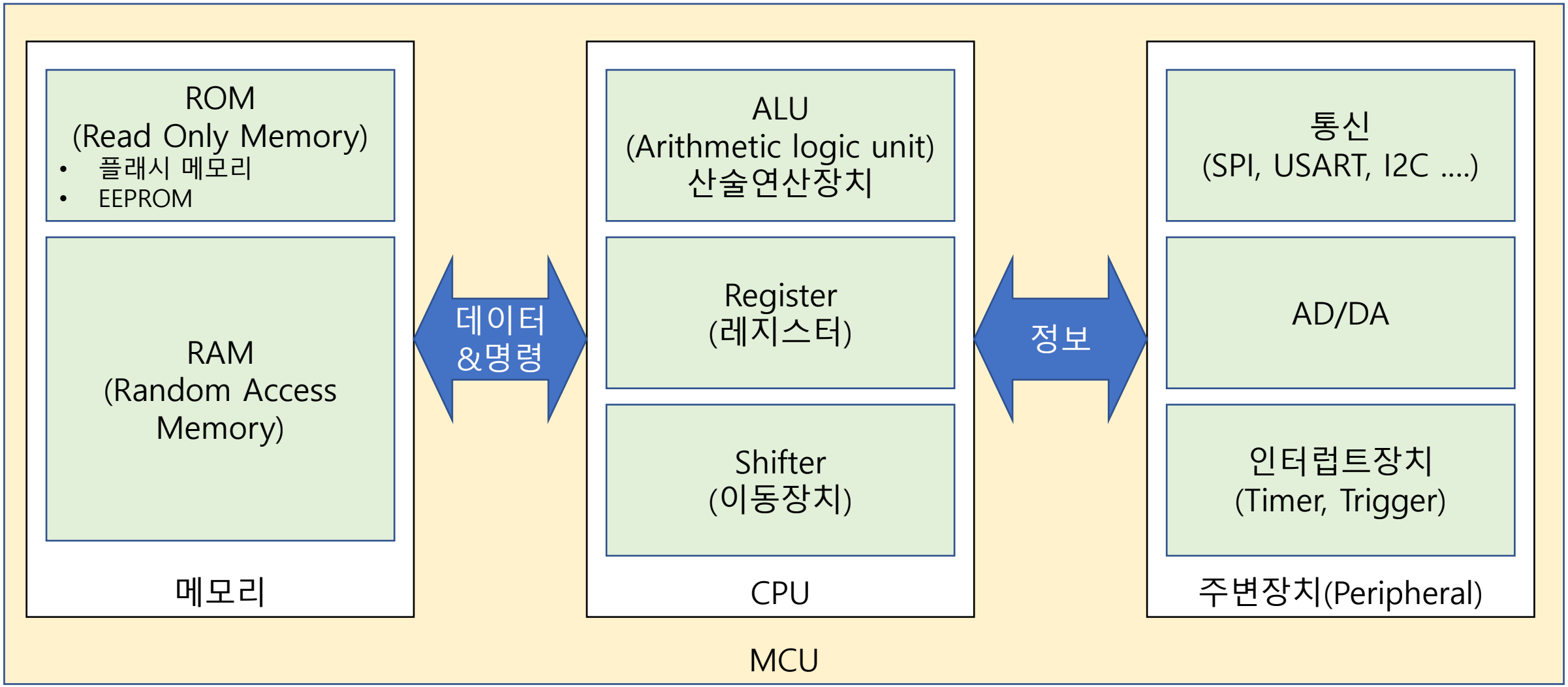
수고하셨습니다.

다음주에 만나요.

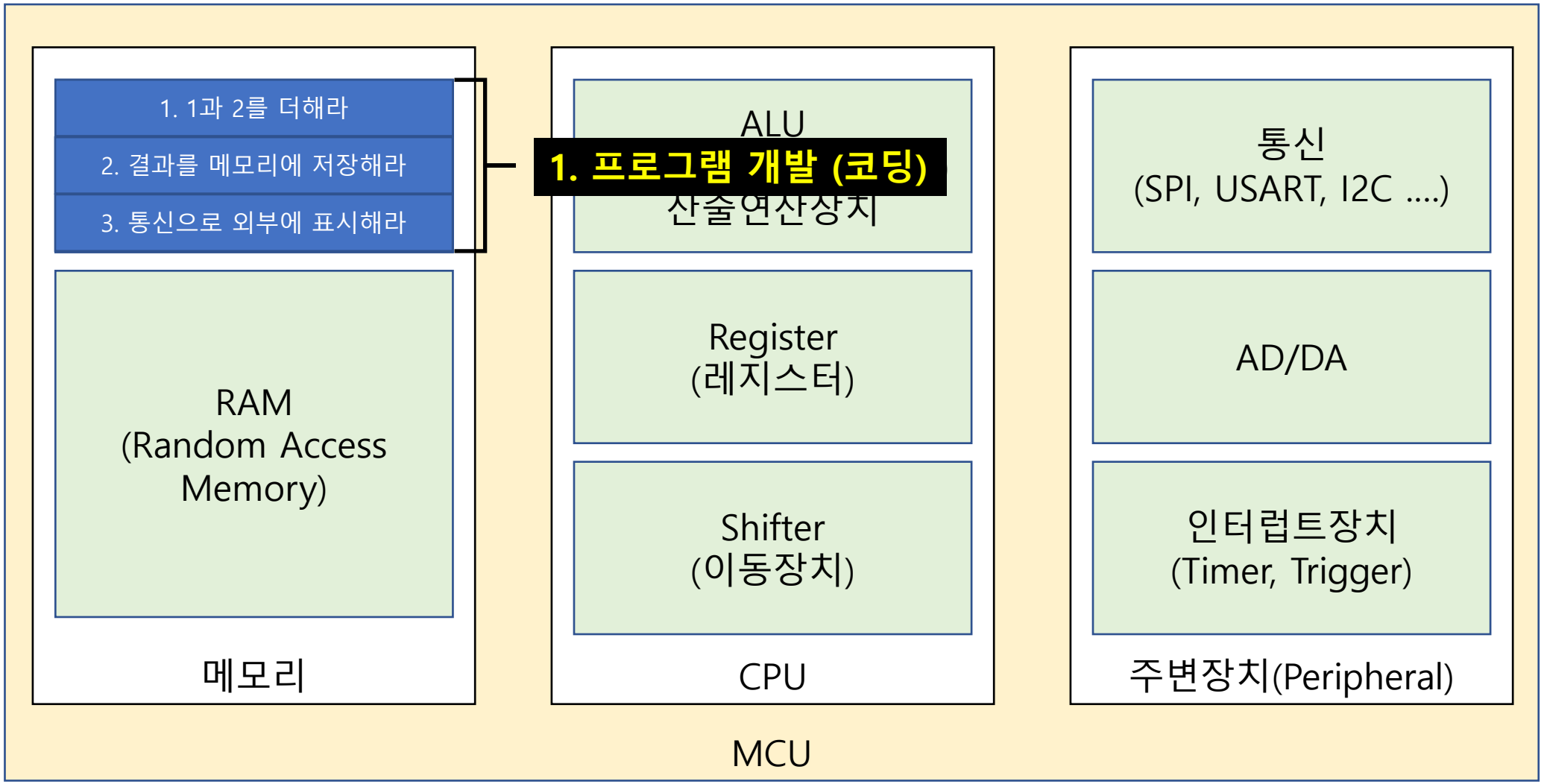
마이크로프로세서 이론

마이크로프로세서 종합 설계. 2주차.

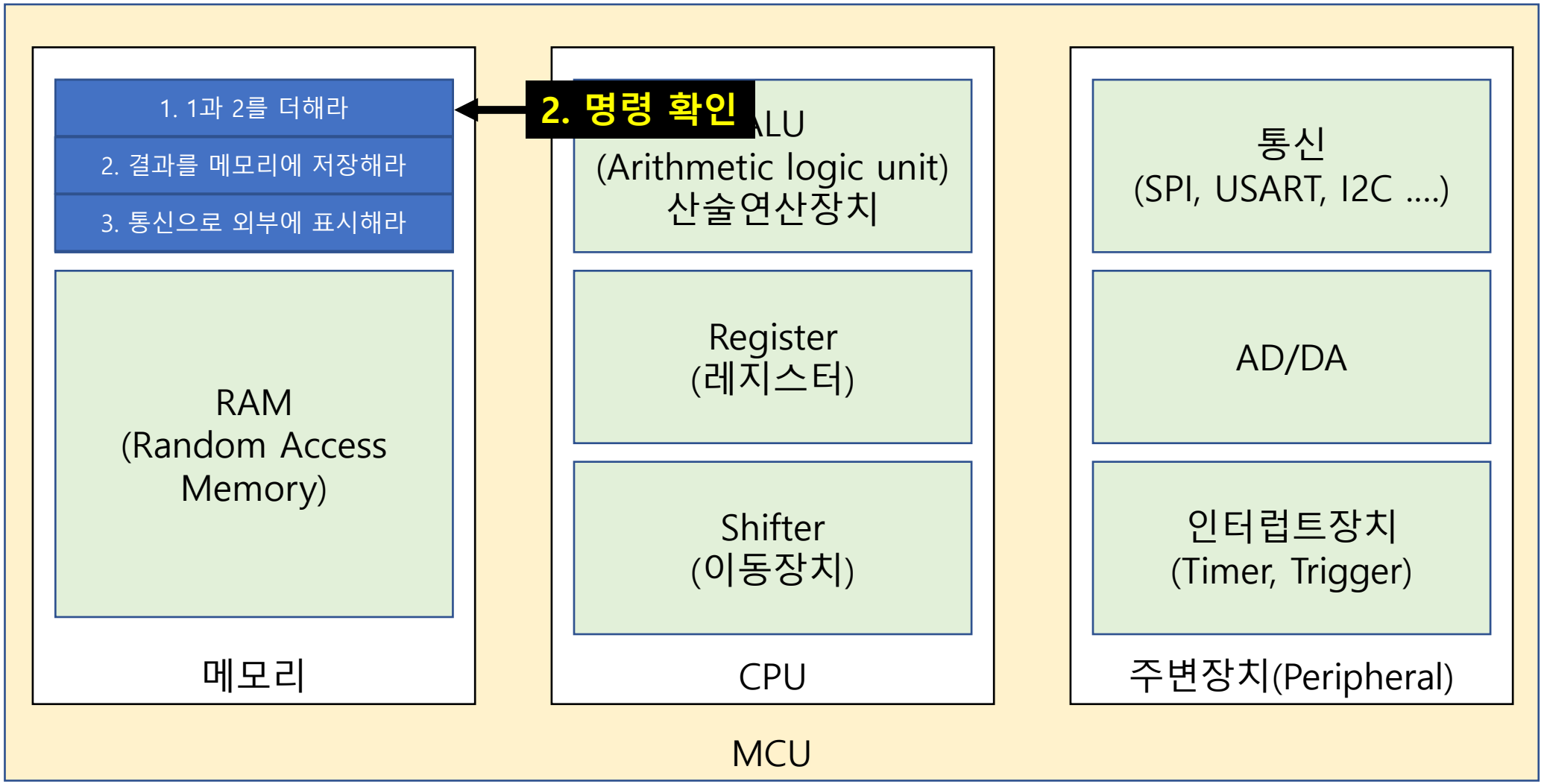
마이크로프로세서의 기본 구성



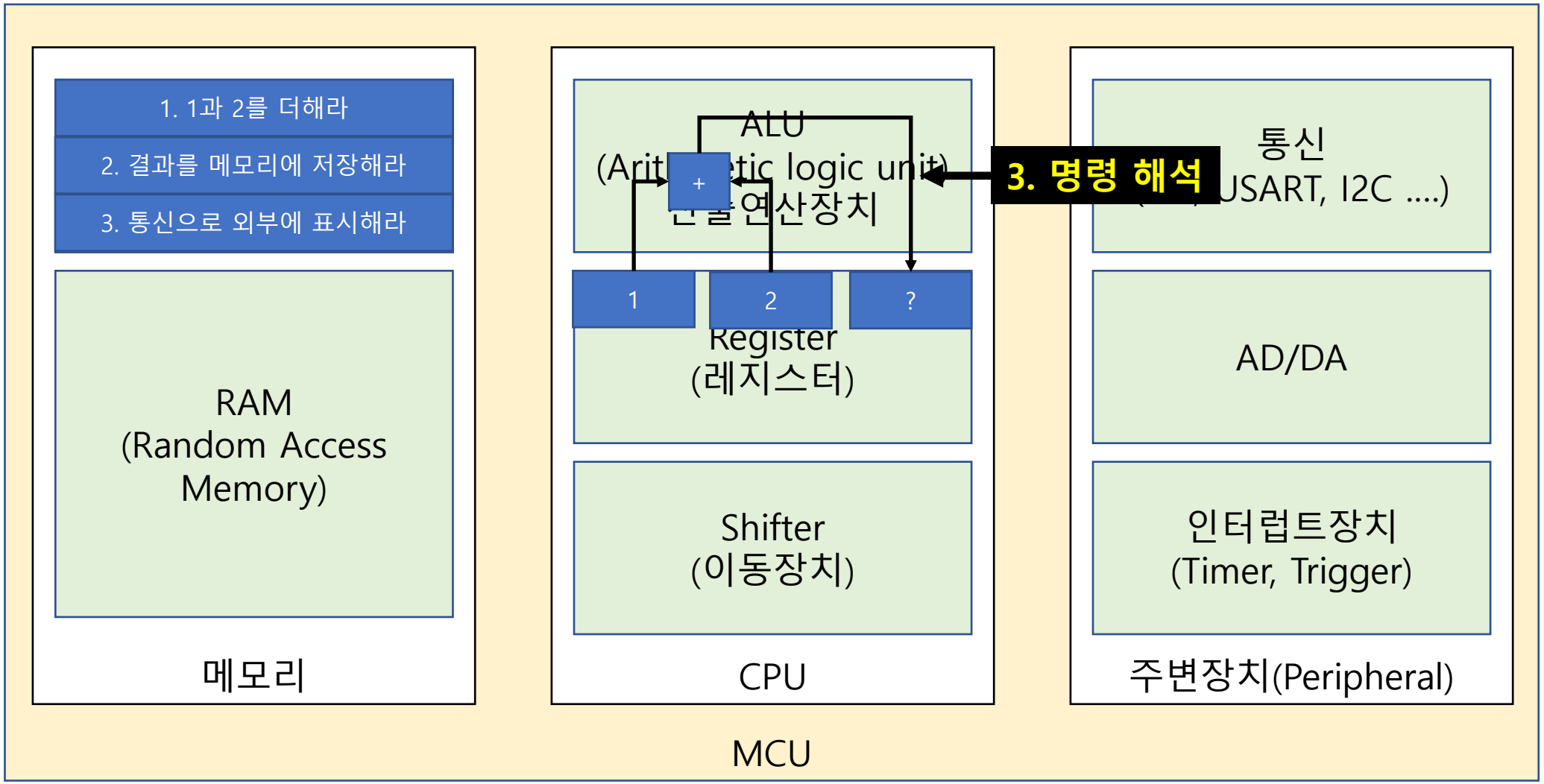
마이크로프로세서는 어떻게 명령을 수행할까?



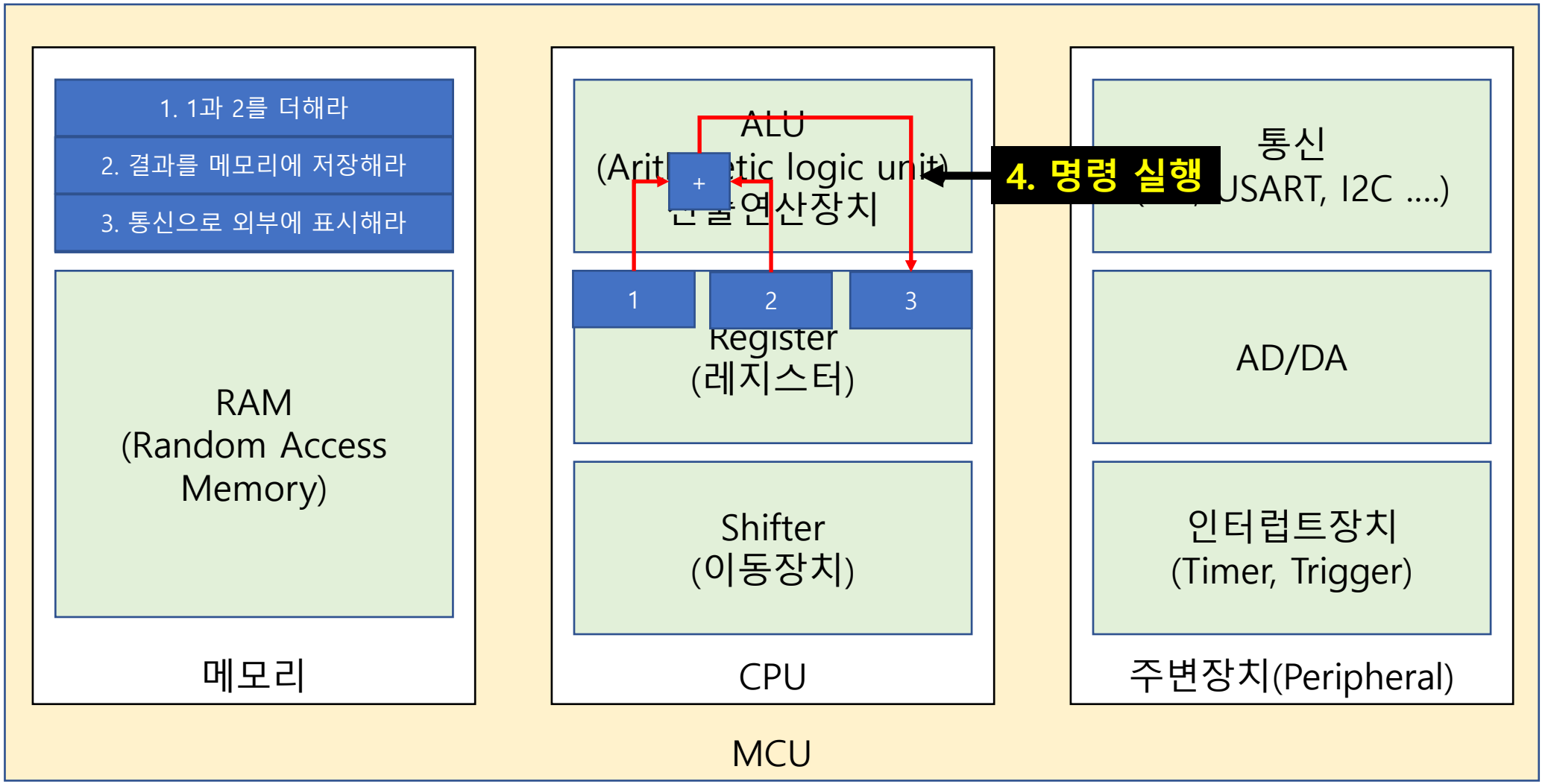
마이크로프로세서는 어떻게 명령을 수행할까?



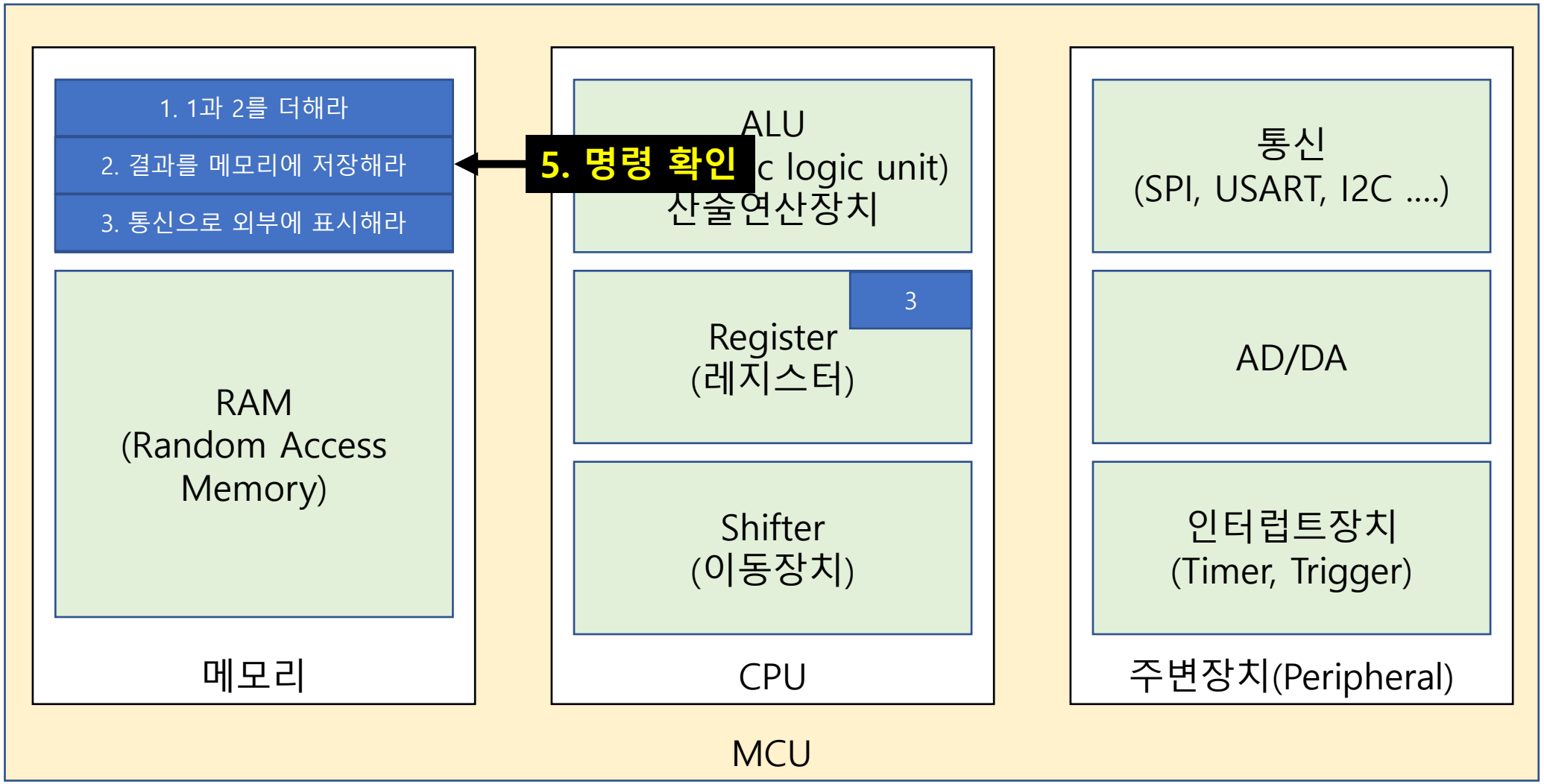
마이크로프로세서는 어떻게 명령을 수행할까?



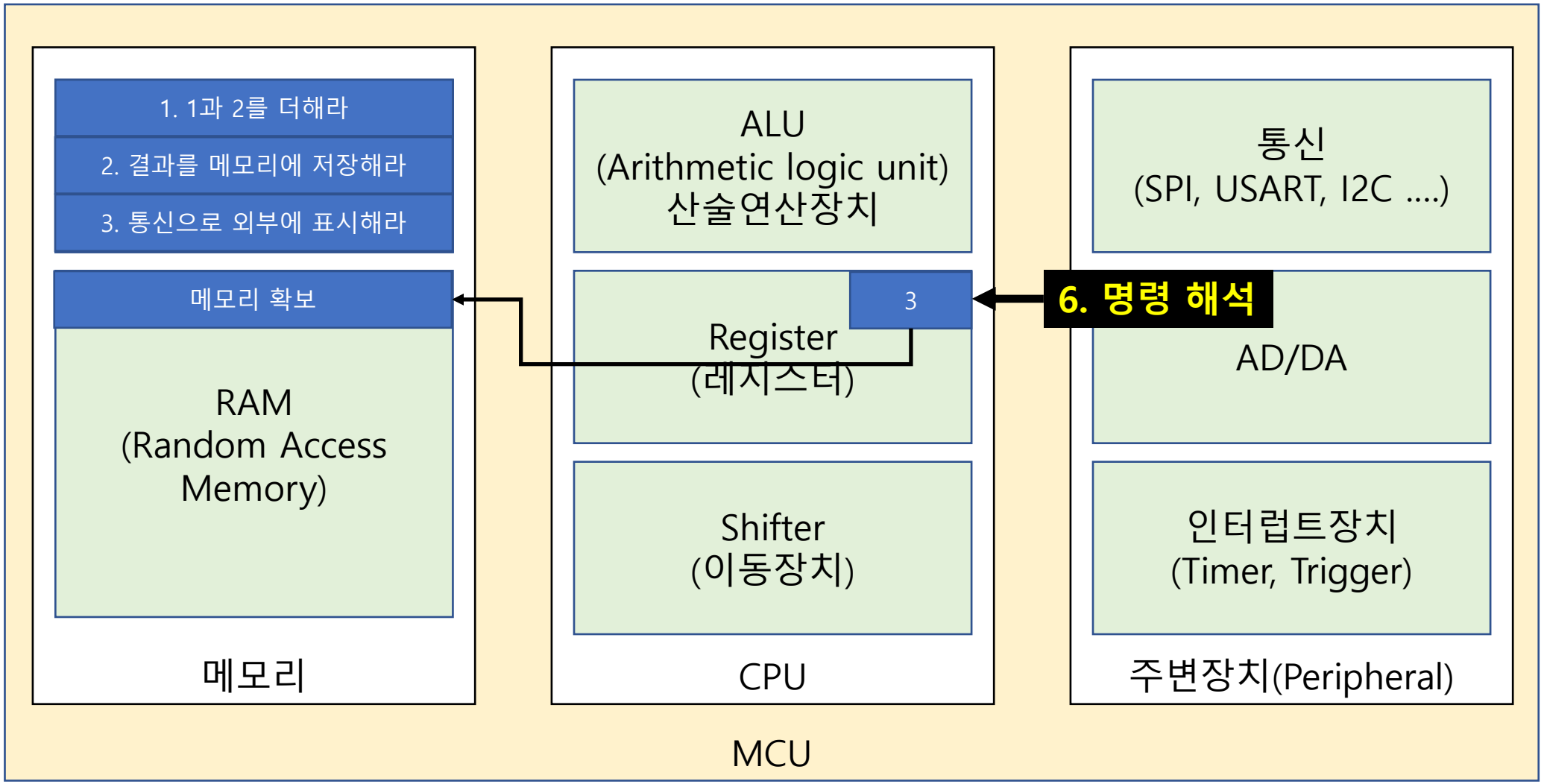
마이크로프로세서는 어떻게 명령을 수행할까?



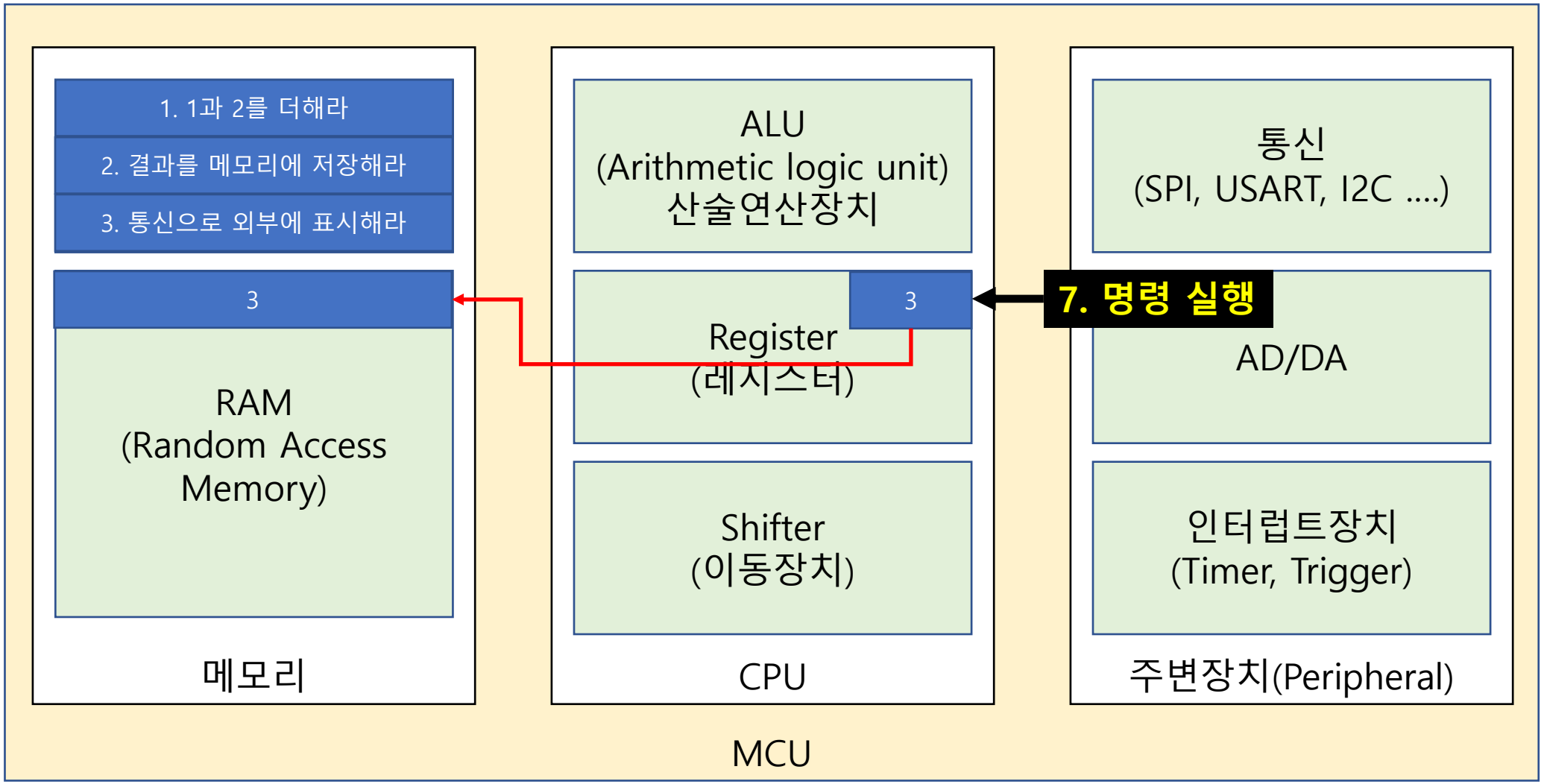
마이크로프로세서는 어떻게 명령을 수행할까?



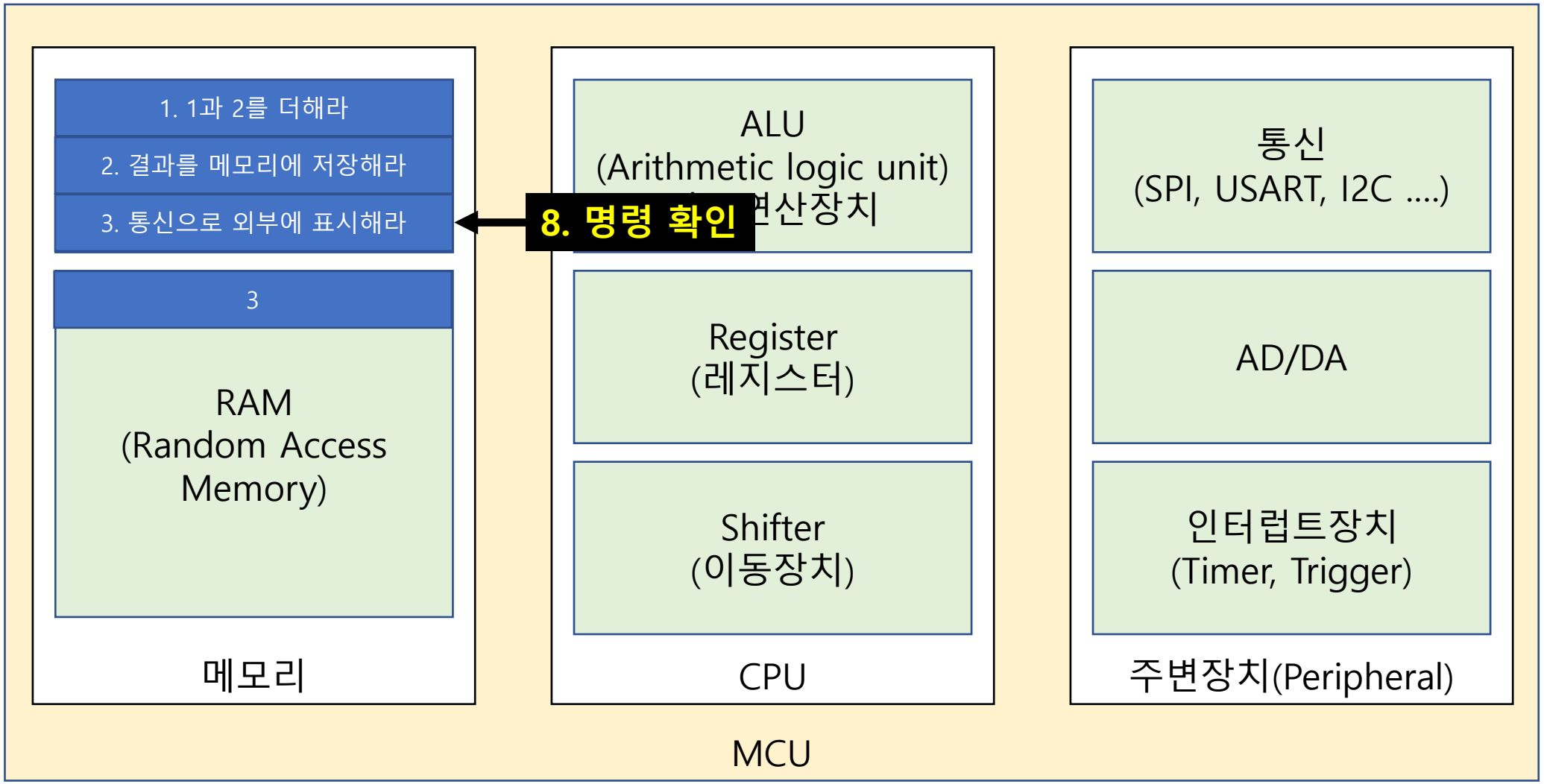
마이크로프로세서는 어떻게 명령을 수행할까?



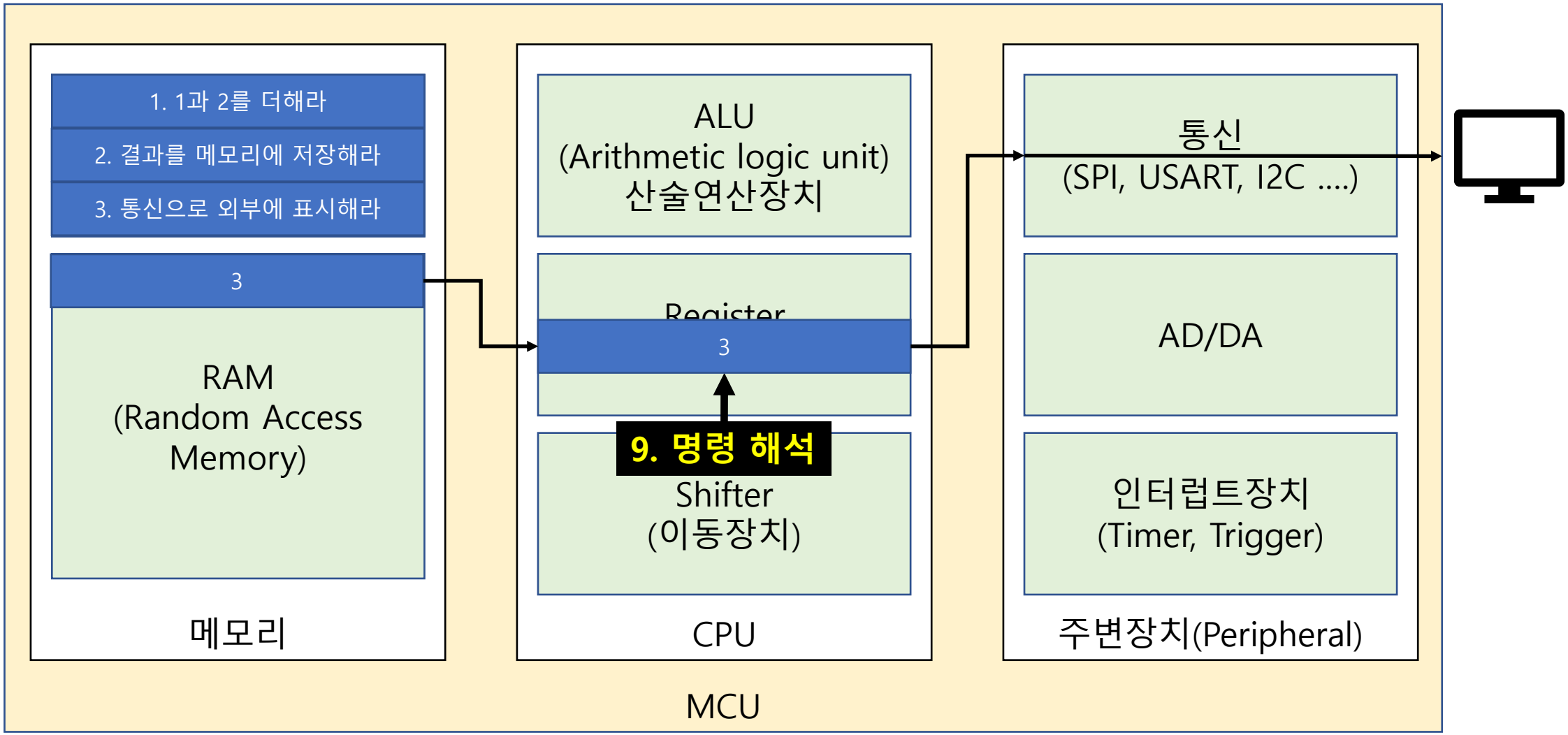
마이크로프로세서는 어떻게 명령을 수행할까?



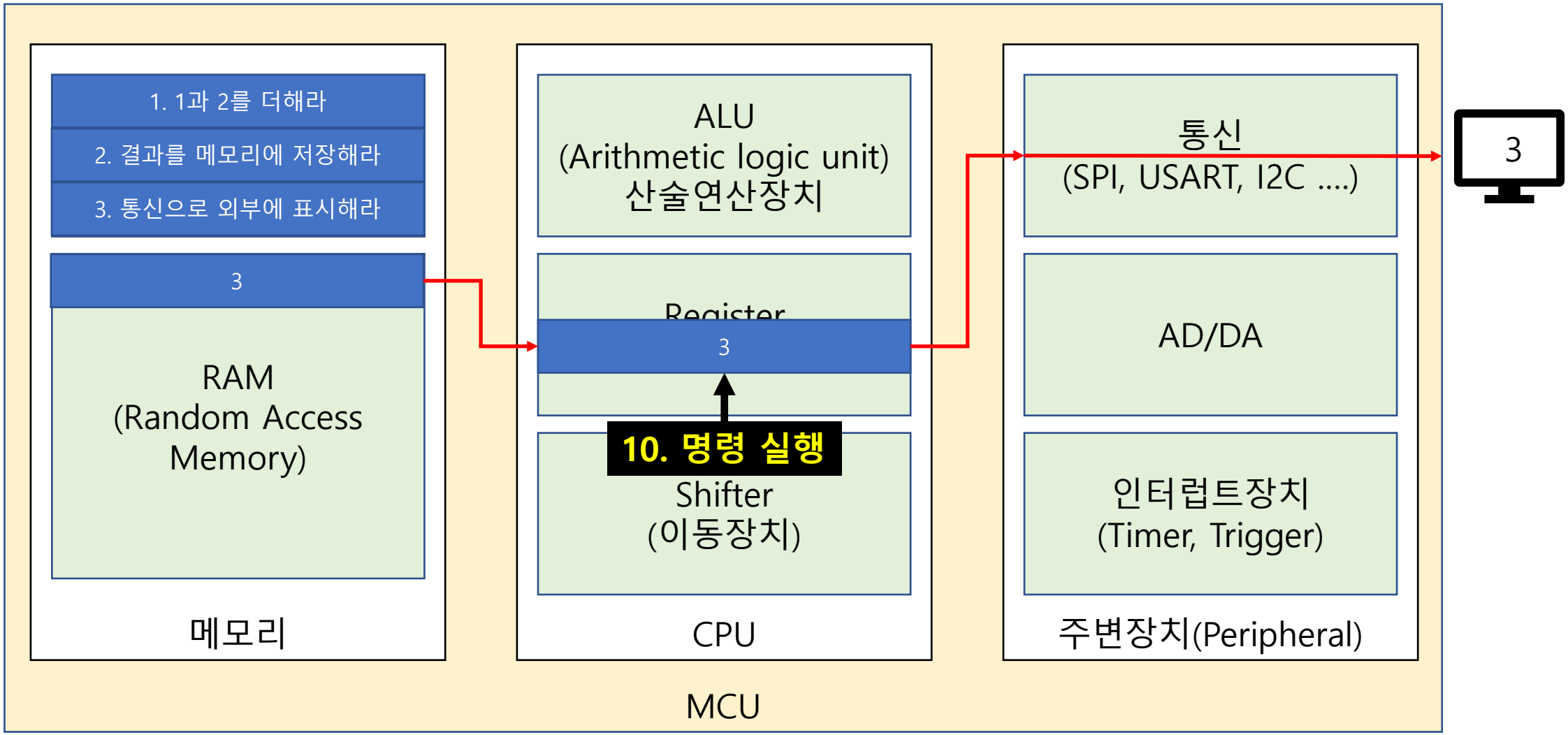
마이크로프로세서는 어떻게 명령을 수행할까?



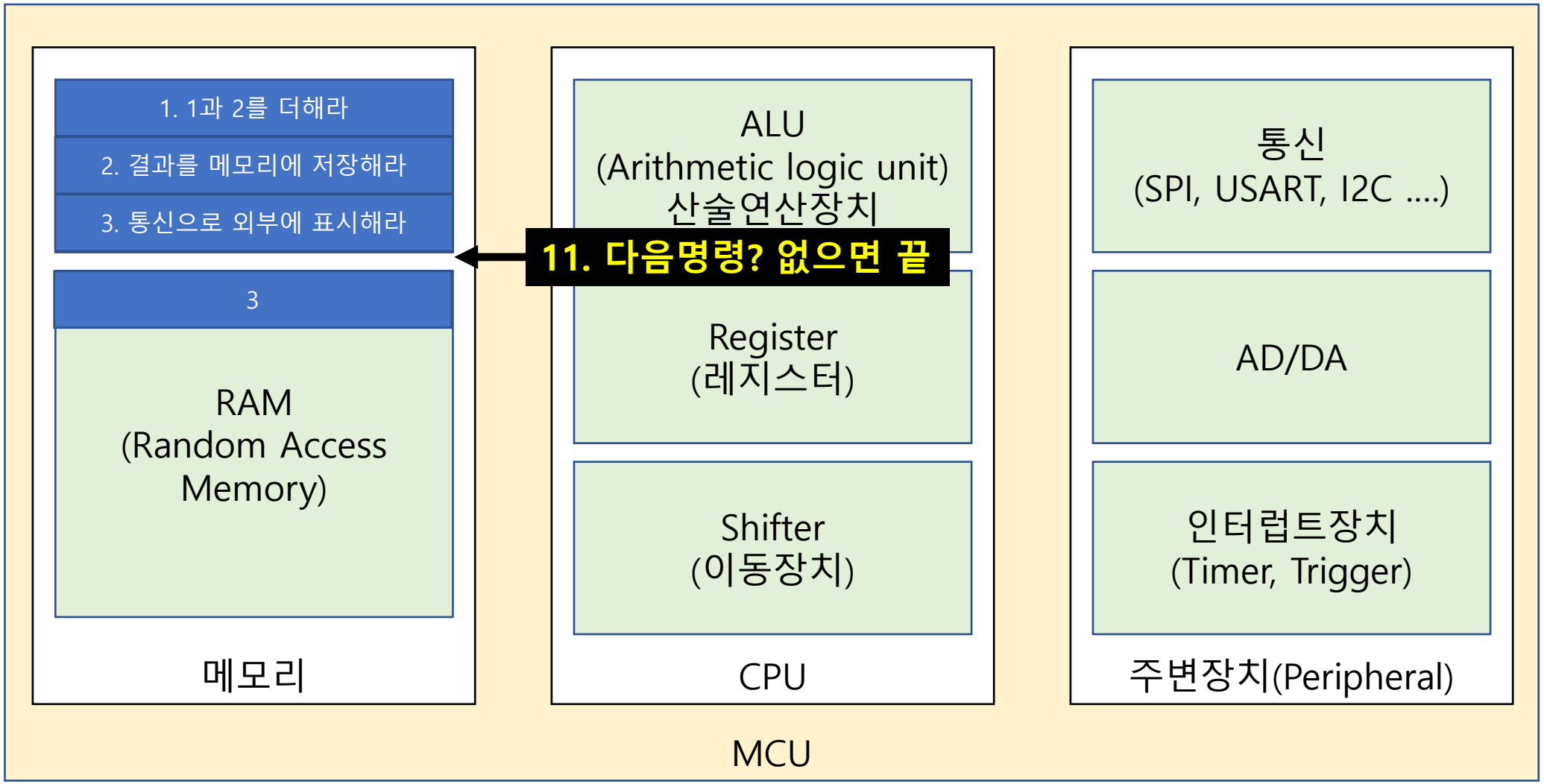
마이크로프로세서는 어떻게 명령을 수행할까?



마이크로프로세서는 어떻게 명령을 수행할까?



마이크로프로세서는 어떻게 명령을 수행할까?

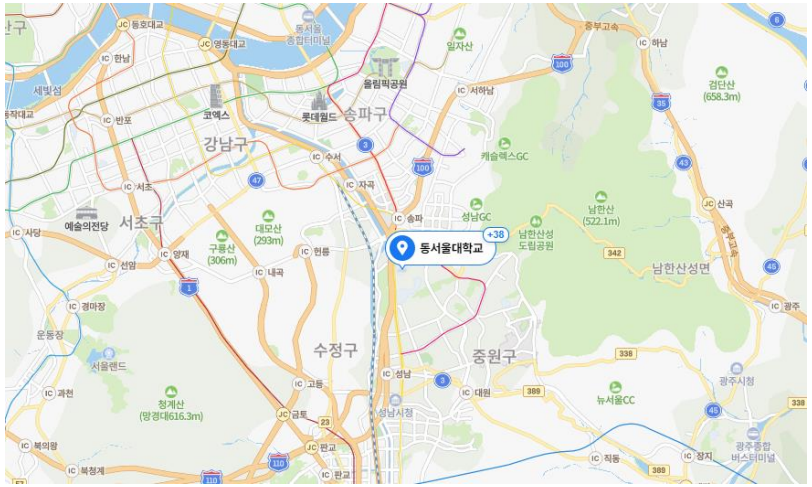


10진수. 2진수? 8진수? 16진수?

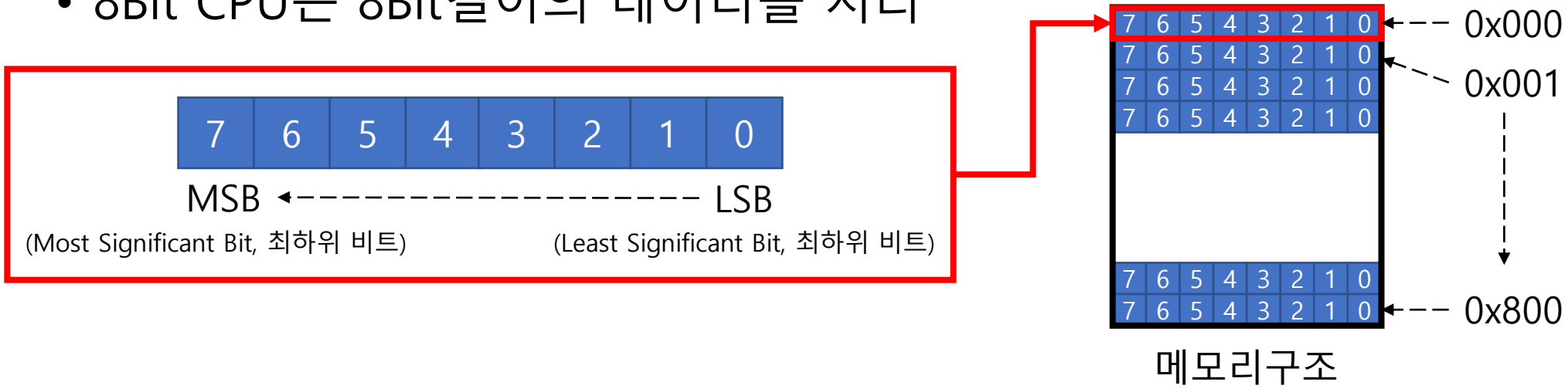
- 10진수 : 우리가 사용하고 있는 수 시스템(손가락은 10개)
- 2진수 : 컴퓨터가 사용하는 기본 수 시스템(1과 0)
- 8진수 : 2진수의 조합을 사람이 쉽게 이해(0~7까지)
- 16진수 : 2진수의 조합을 사람이 쉽게 이해(0~15까지)

- 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?

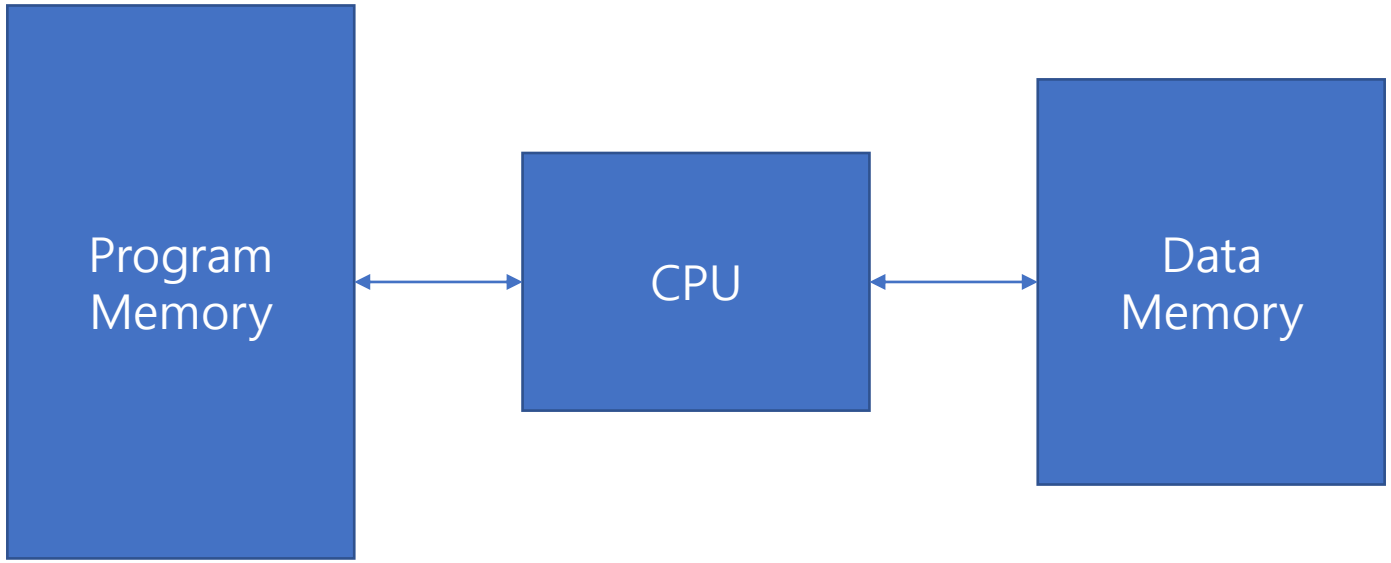


- 메모리의 장소 정보 (주소)
  - 동서울대학교 : 경기 성남시 수정구 복정로 76
- 메모리도 주소(연속숫자)를 이용하여 데이터를 참조 한다.
- ATmega328p의 경우 내부에 2KByte의 RAM을 가지고 있다.
  - 1Byte → 8Bit
  - 8Bit CPU는 8Bit길이의 데이터를 처리



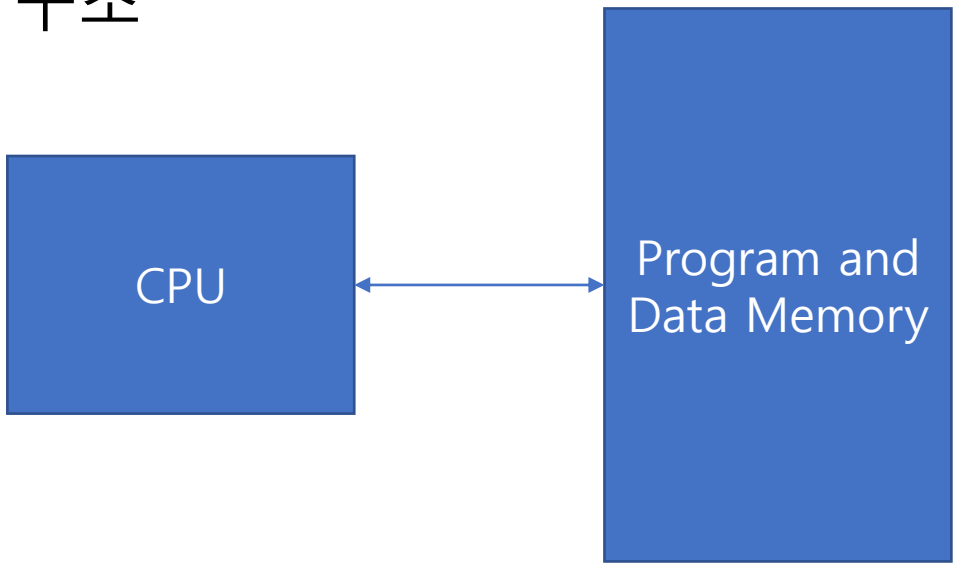
컴퓨터구조(폰노이만vs하버드)

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



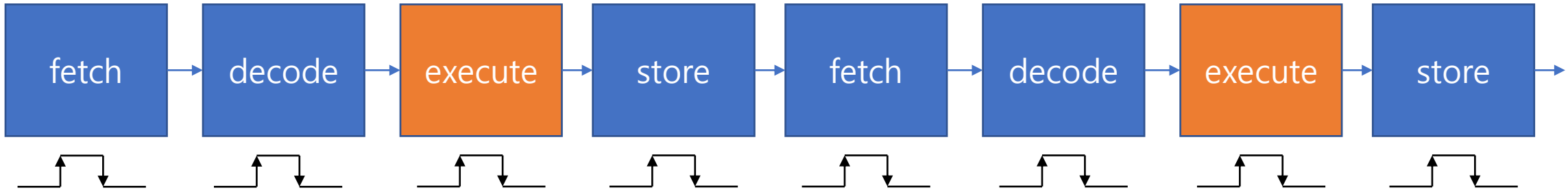
컴퓨터구조(폰노이만vs하버드)

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



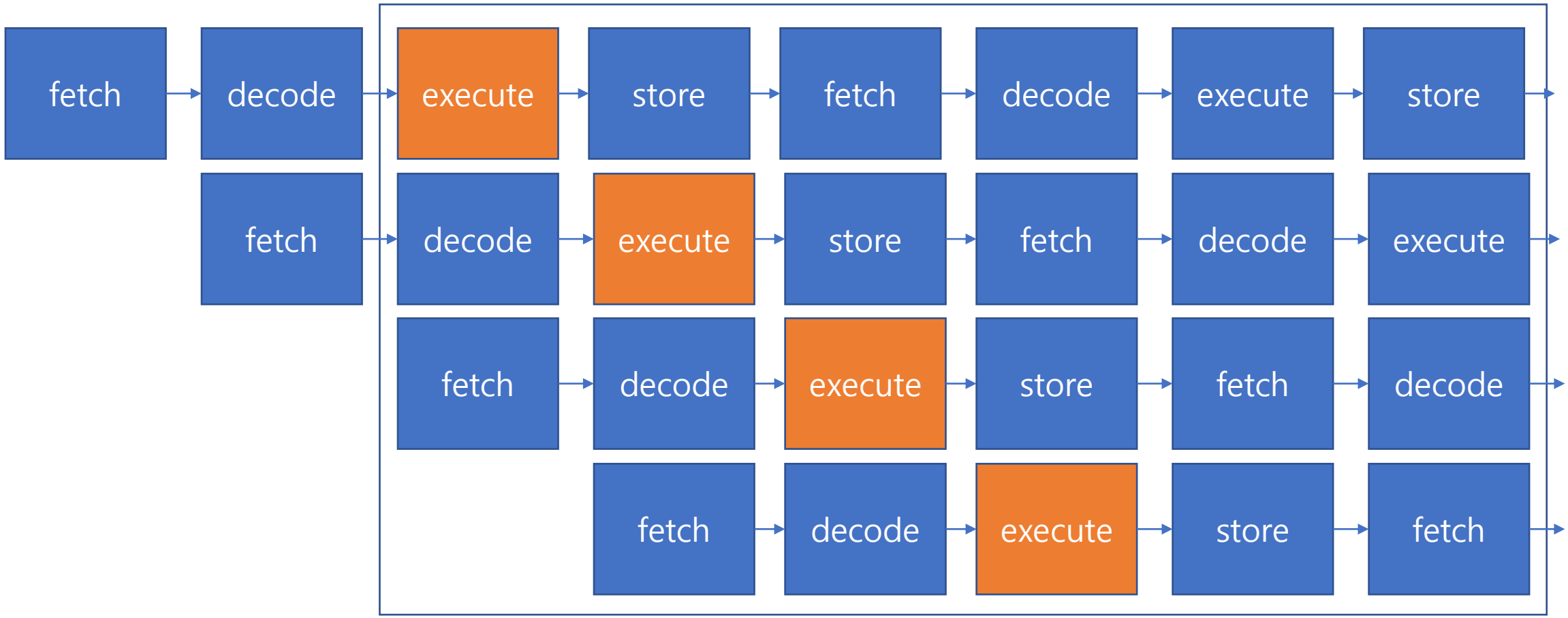
컴퓨터 명령 실행 과정

- 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)

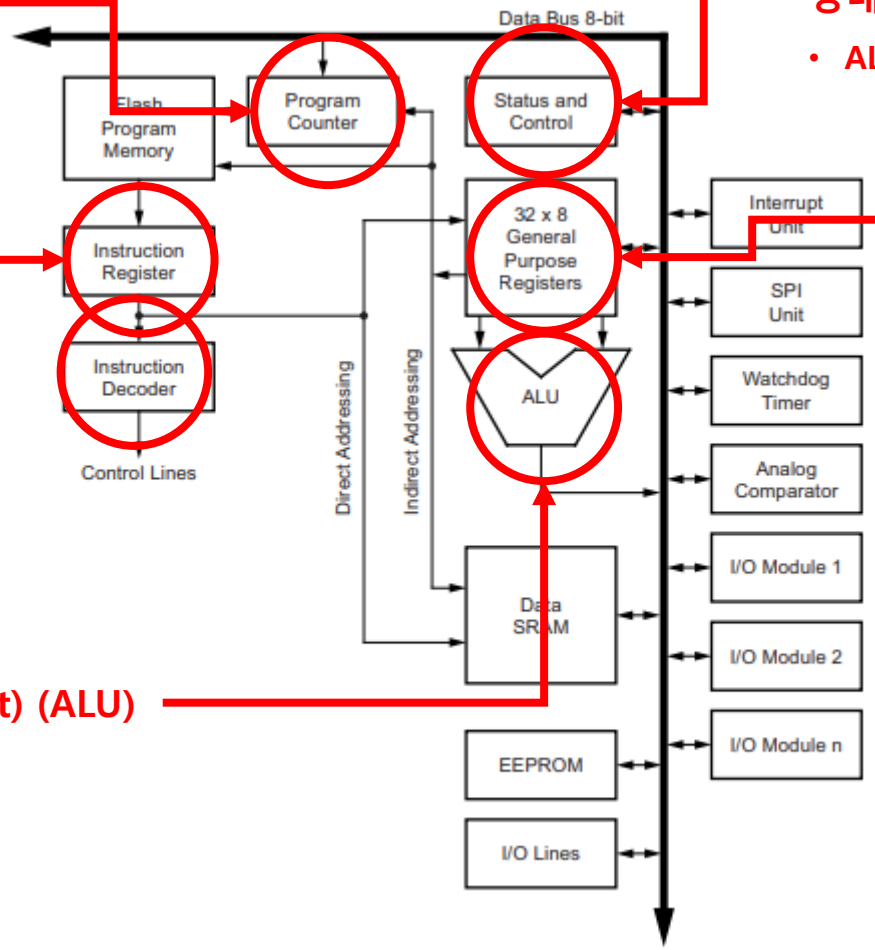
- 내가 실행시킨 명령어의 주소를 가리키는 역할

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)

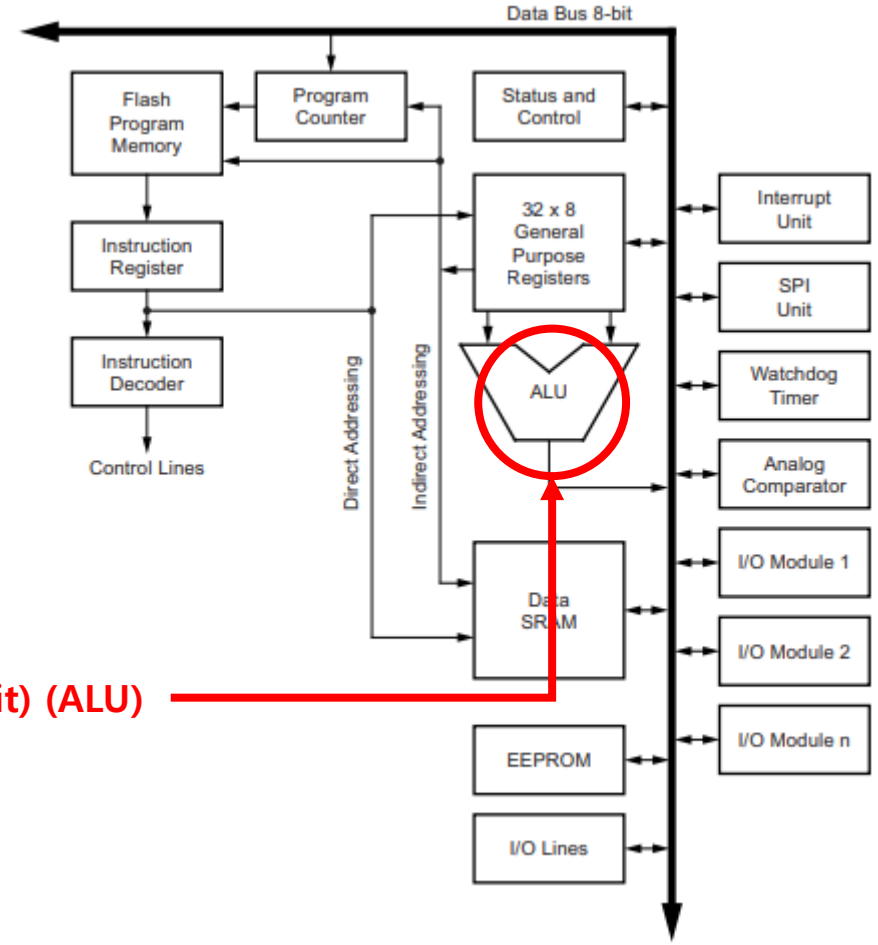
- ALU가 가장 최근에 실행한 연산 명령의 결과와 상태를 표시

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

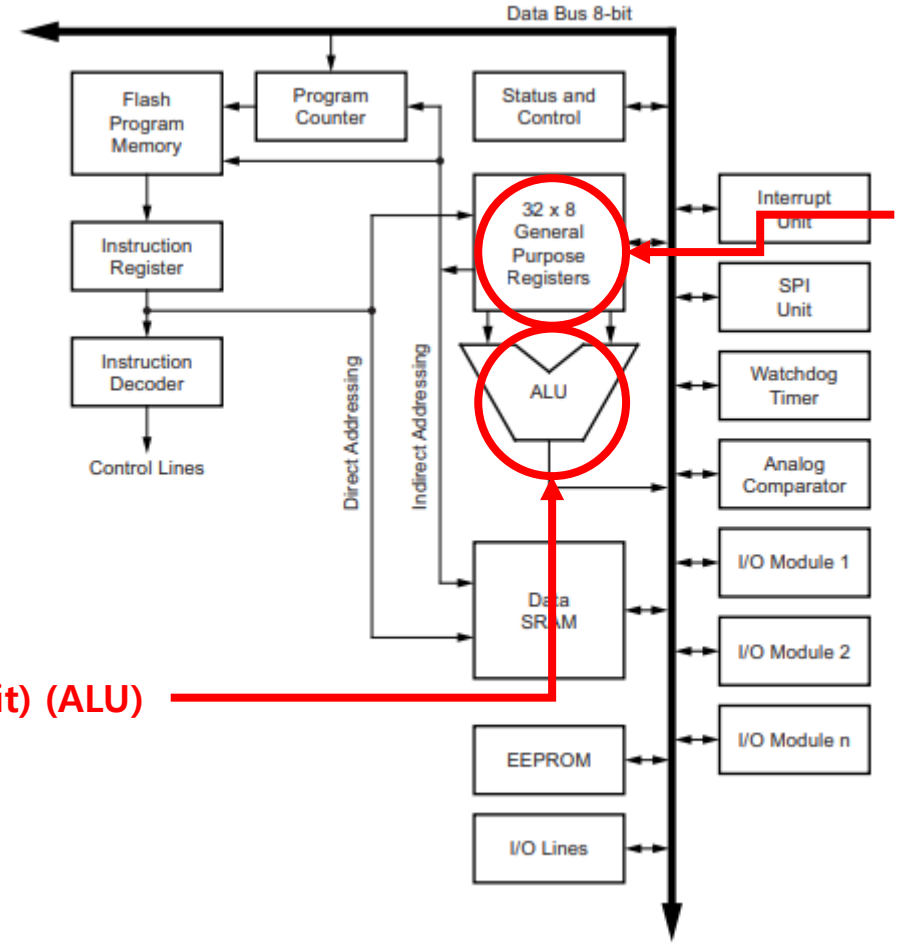


산술 논리 장치 (Arithmetic Logical Unit) (ALU)

- **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)

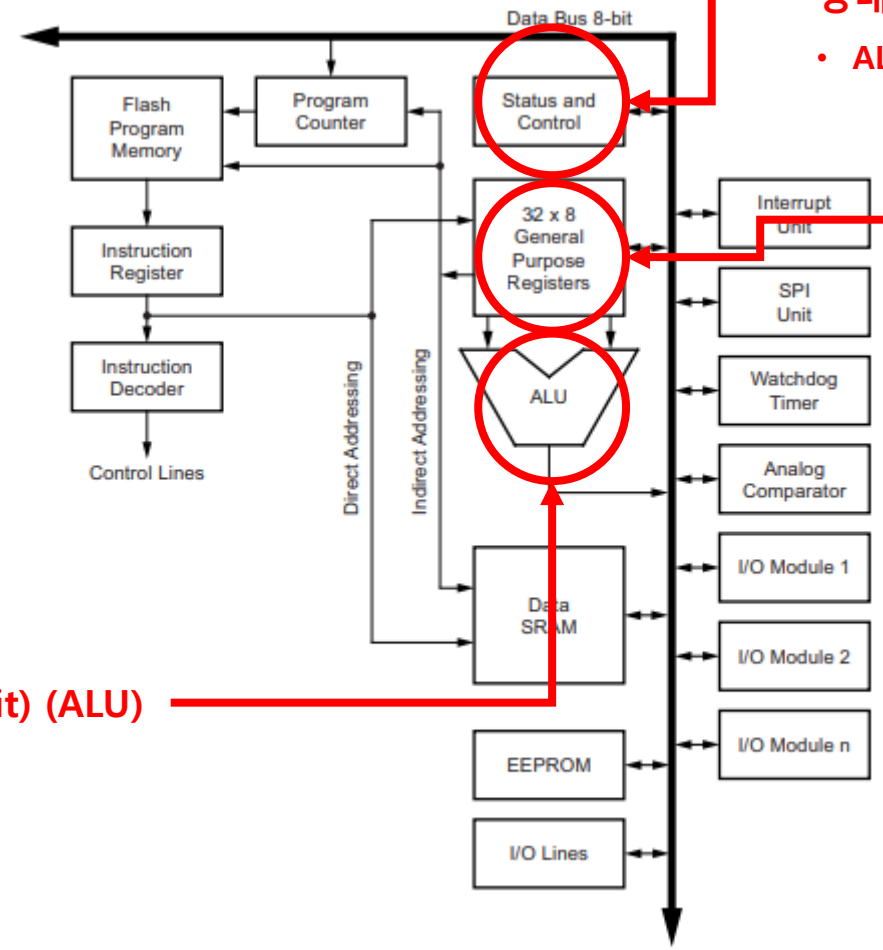
- 프로그램 수행 중에 중간 결과나 데이터를 일시적으로 저장하는데 사용

산술 논리 장치 (Arithmetic Logical Unit) (ALU)

- 4칙연산
- 논리(AND, OR, NOT)
- 비트연산

데이터시트 : 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)

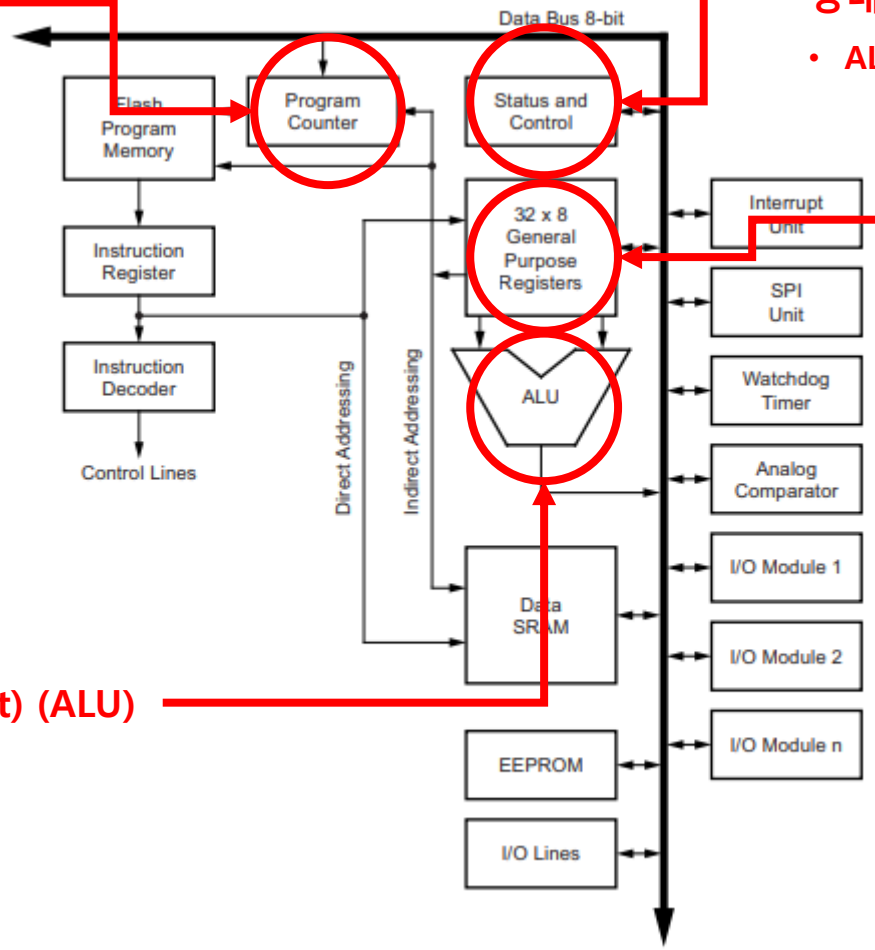
- 내가 실행시킨 명령어의 주소를 가리키는 역할

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

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

ATMEGA328P 기본 구조

프로그램카운터 (Program Counter : PC)

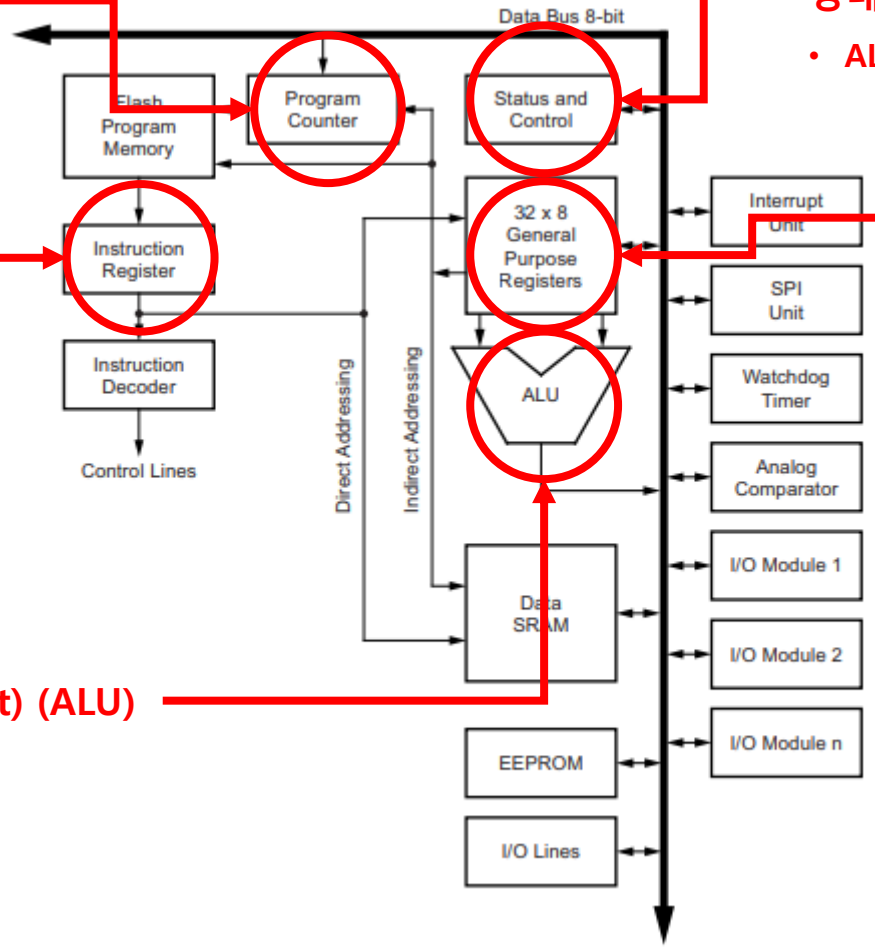
- 내가 실행시킨 명령어의 주소를 가리키는 역할

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)

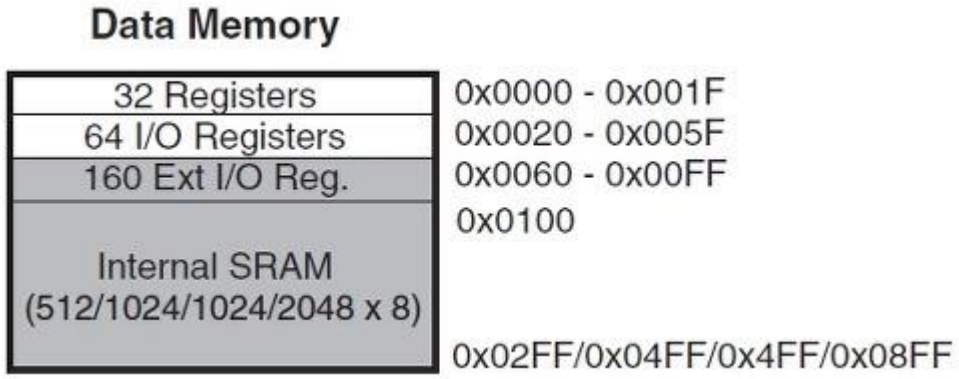
- ALU가 가장 최근에 실행한 연산 명령의 결과와 상태를 표시

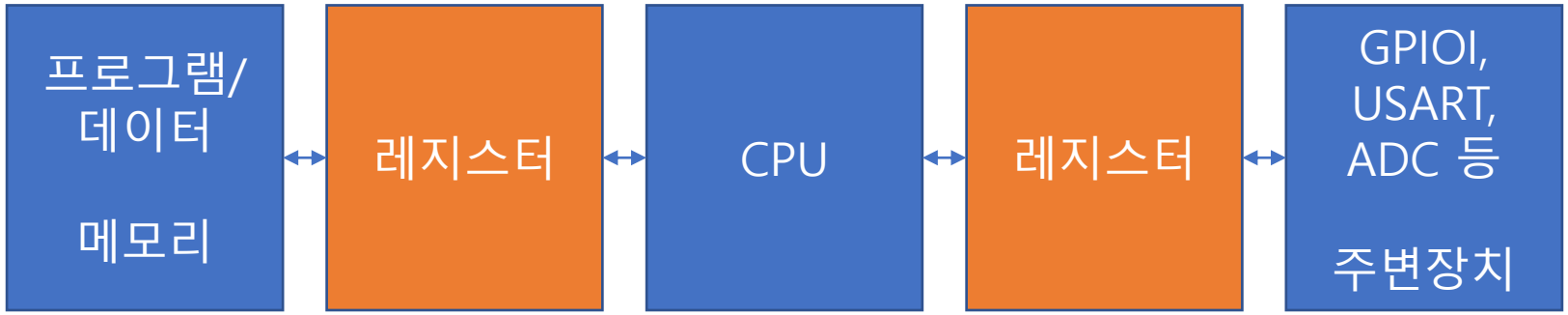
범용 레지스터 (General Purpose Register)

- 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





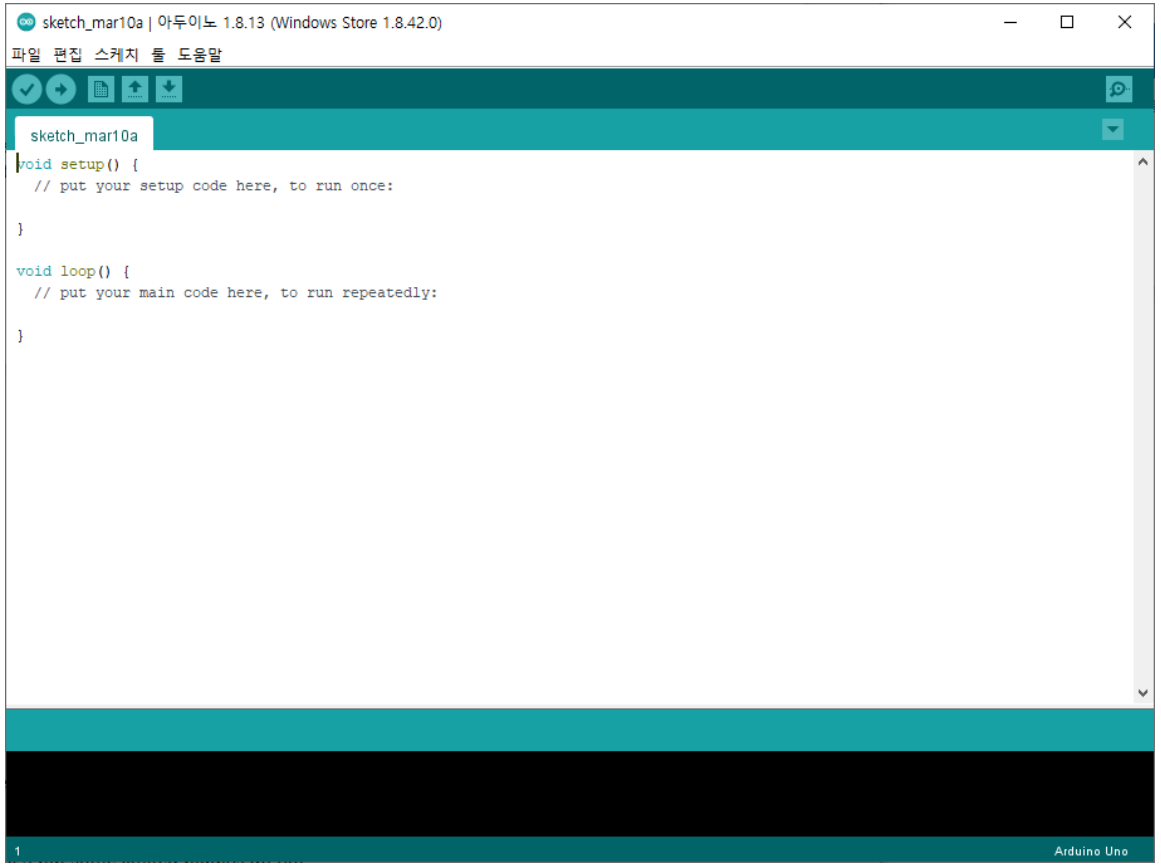
데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
30. Register Summary, 275 page 참고

아두이노 개발 환경 구성

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

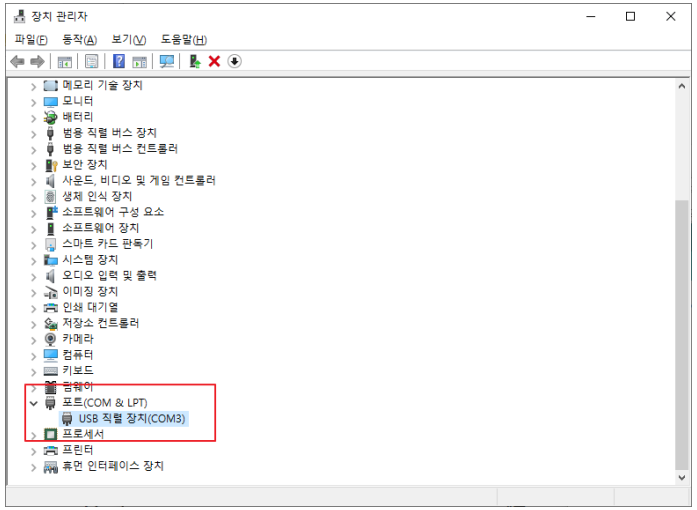
아두이노 개발 환경 구성

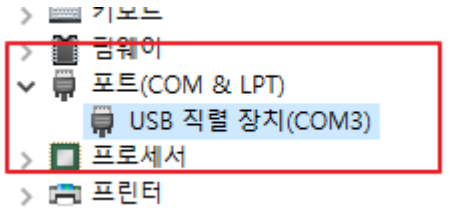
- 아두이노 IDE 실행



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)

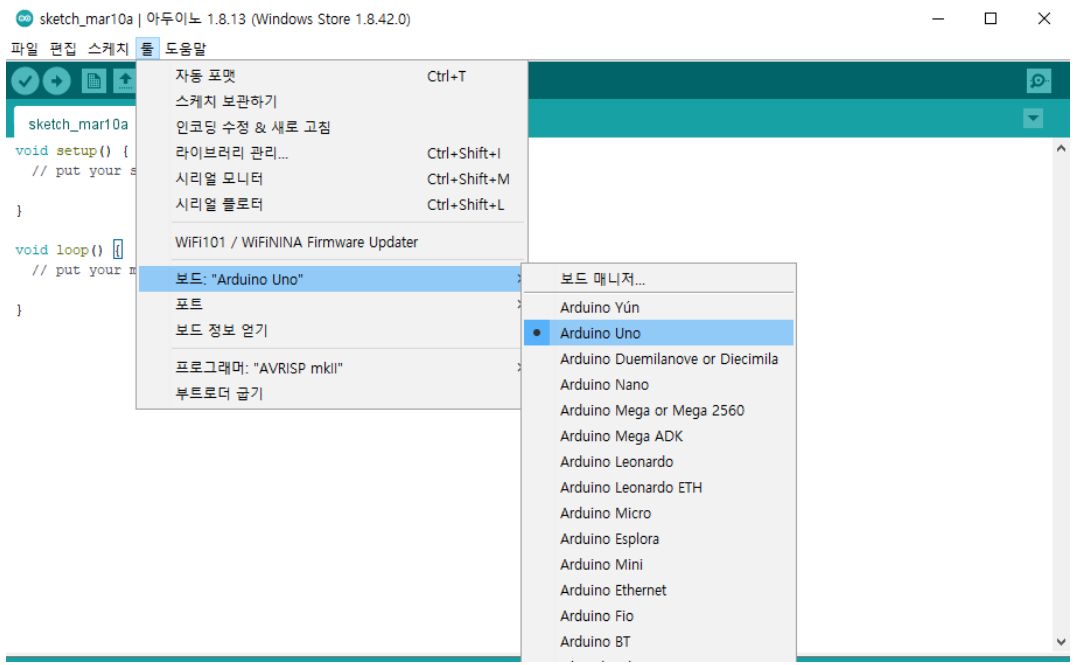




COM3

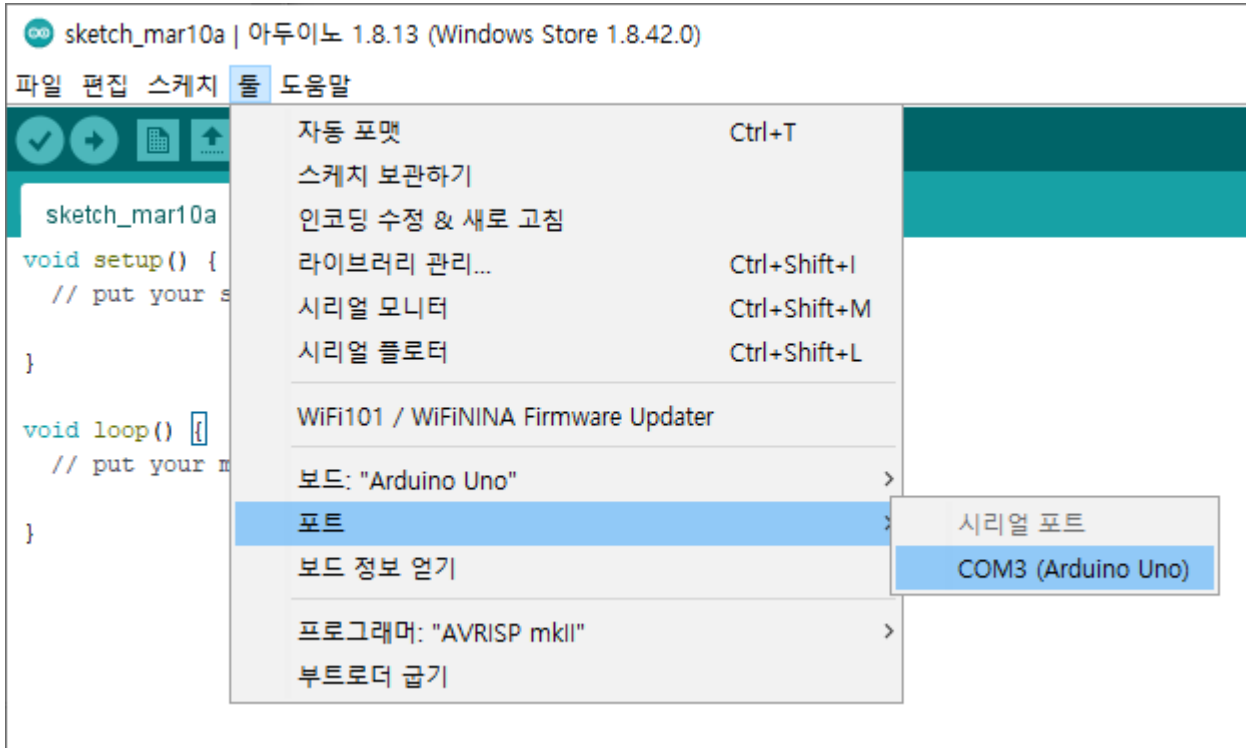
아두이노 개발 환경 구성

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



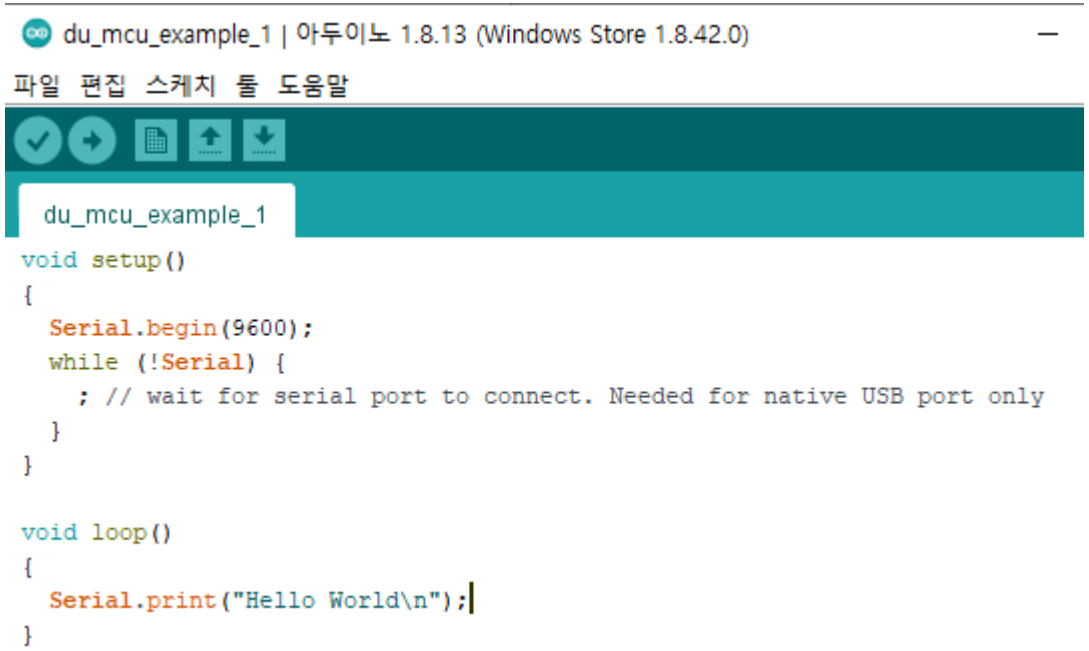
아두이노 개발 환경 구성

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



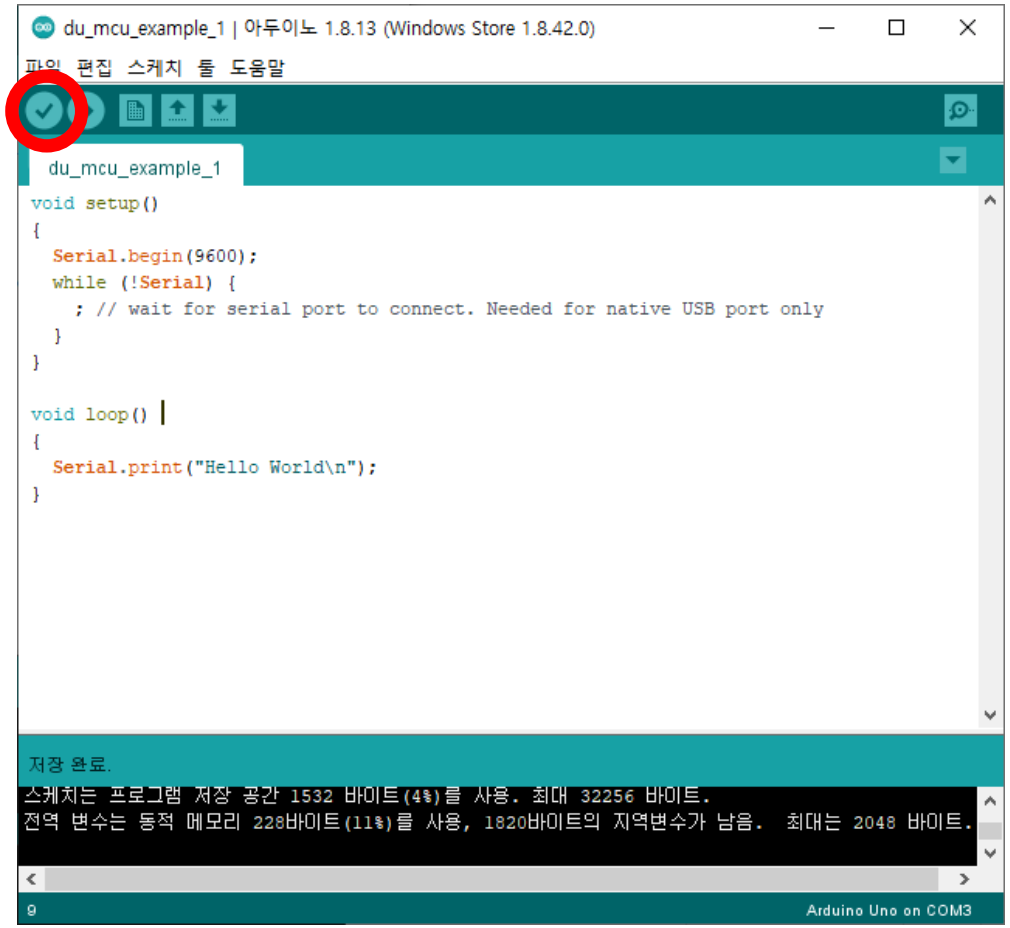
테스트 코드 실험

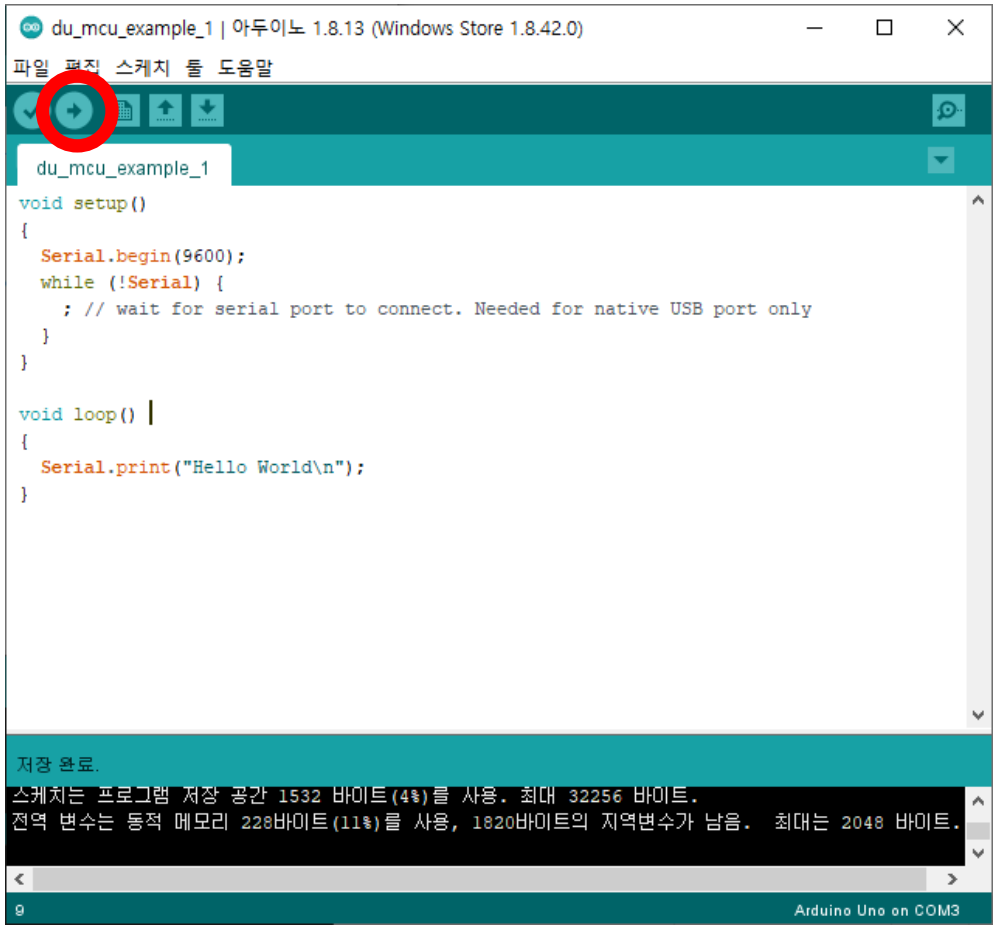
```
void setup()
{
        Serial.begin(9600);
        while (!Serial)
        {
                ; // wait
        }
}

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



테스트 코드 실험

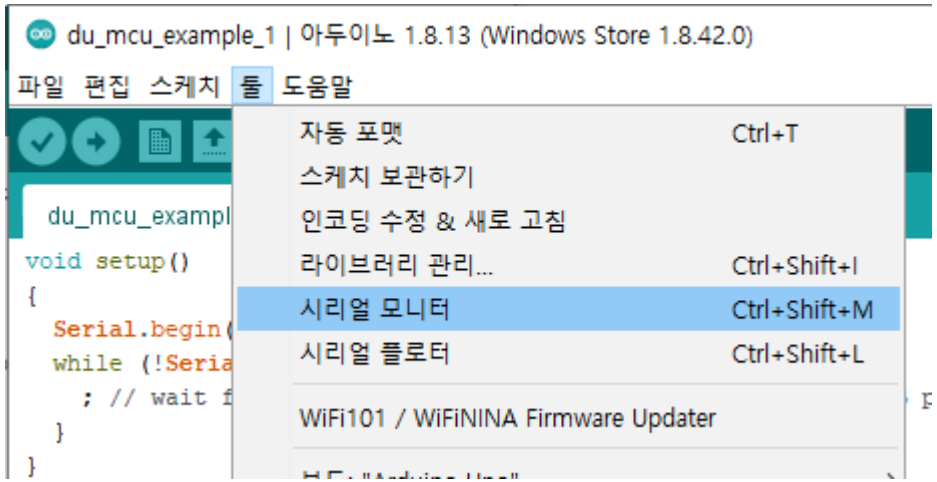
- 컴파일 & 업로드

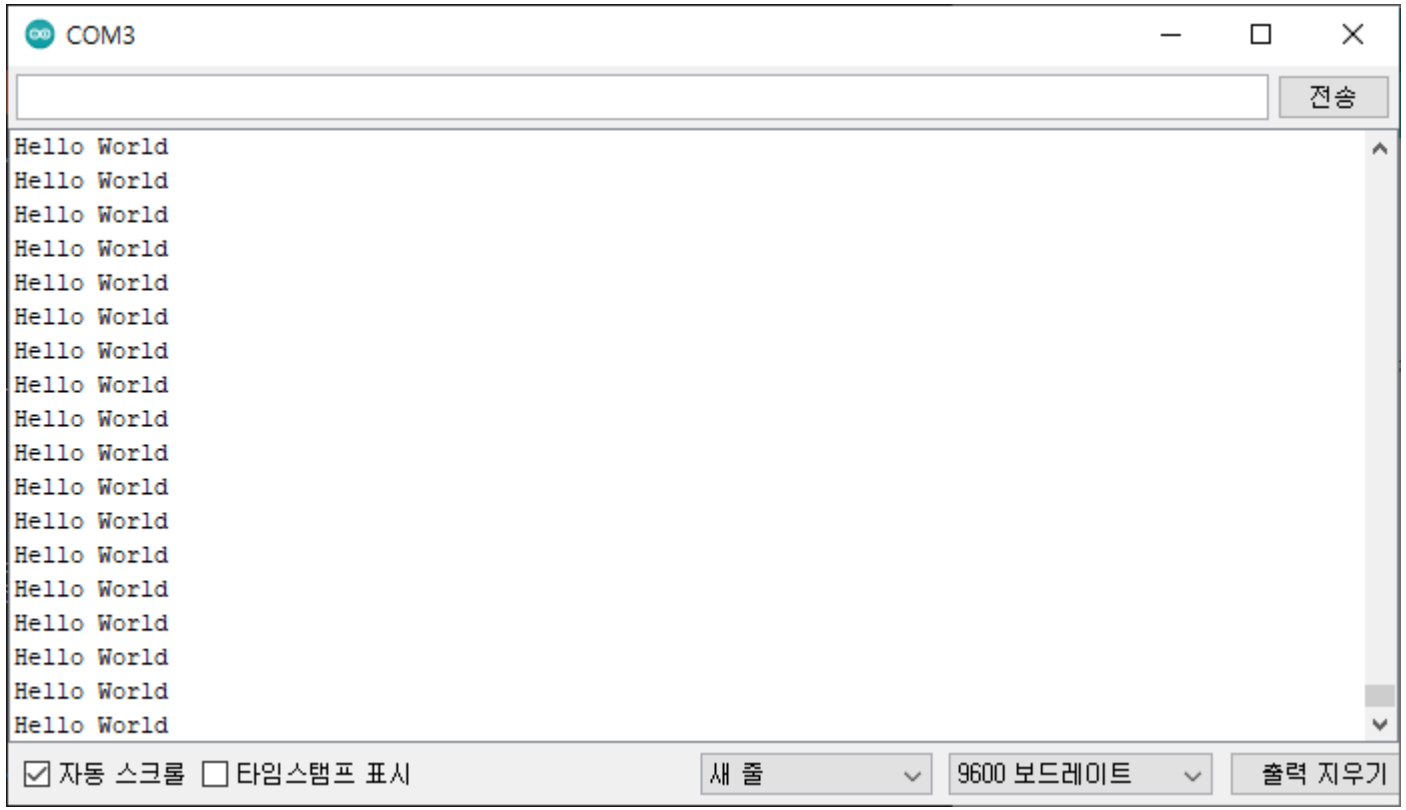




테스트 코드 실험

- 시리얼 통신 확인





테스트 코드 실험

- github
  - https://github.com/juhong-rdv/2021_spring_du_mcu

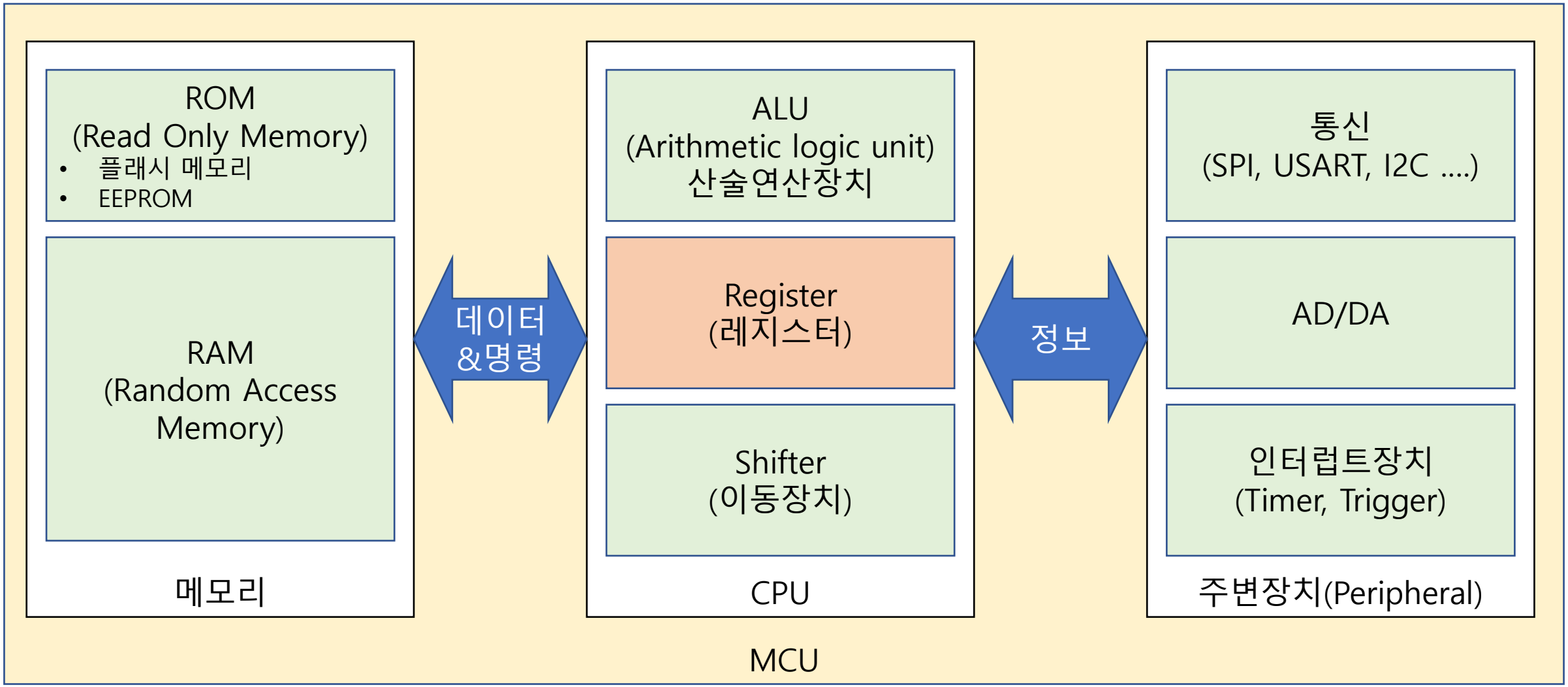
수고하셨습니다.

다음주에 만나요.

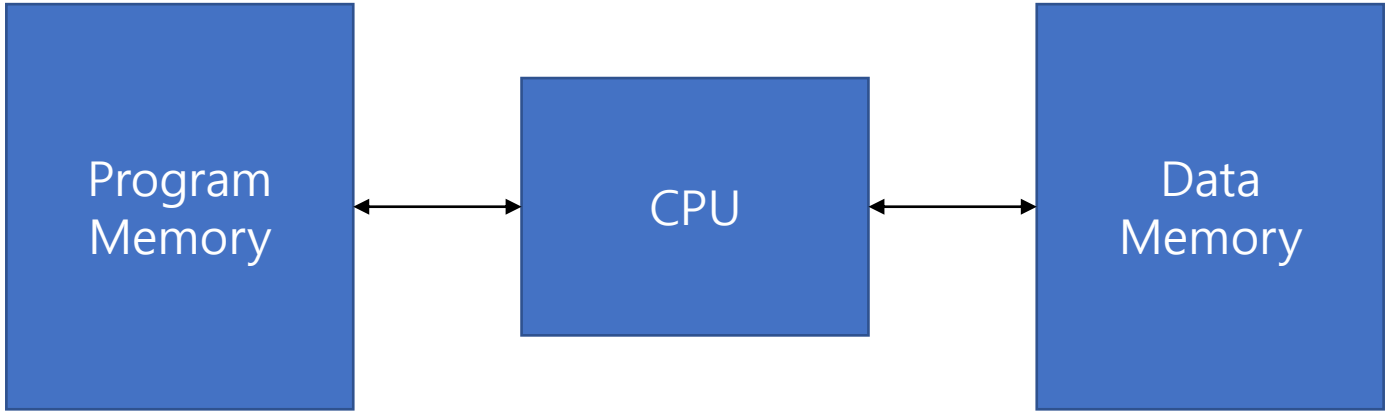
레지스터와 포트의 이해 그리고 C언어

마이크로프로세서 종합 설계. 3주차.

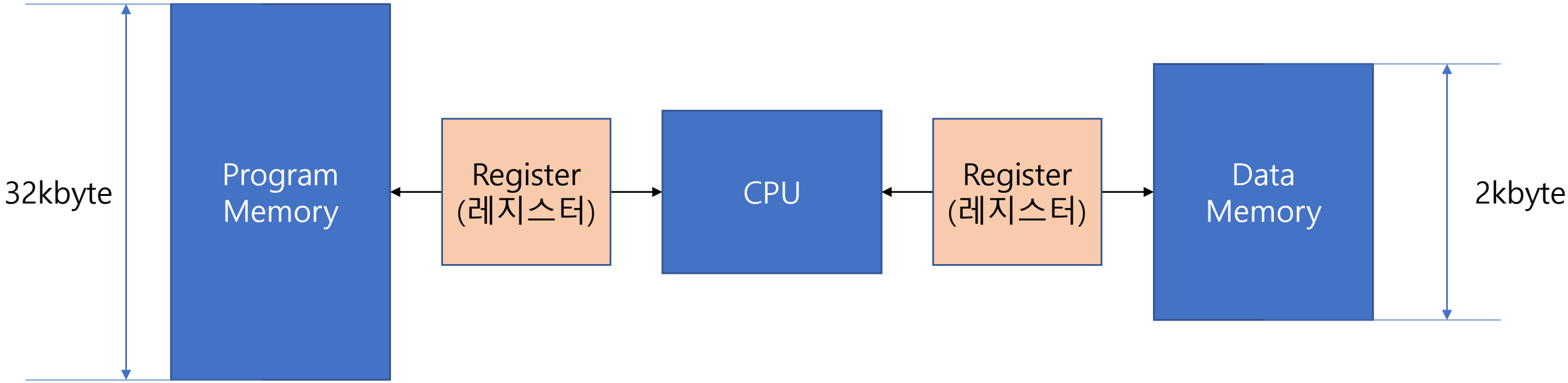
마이크로프로세서의 기본 구성



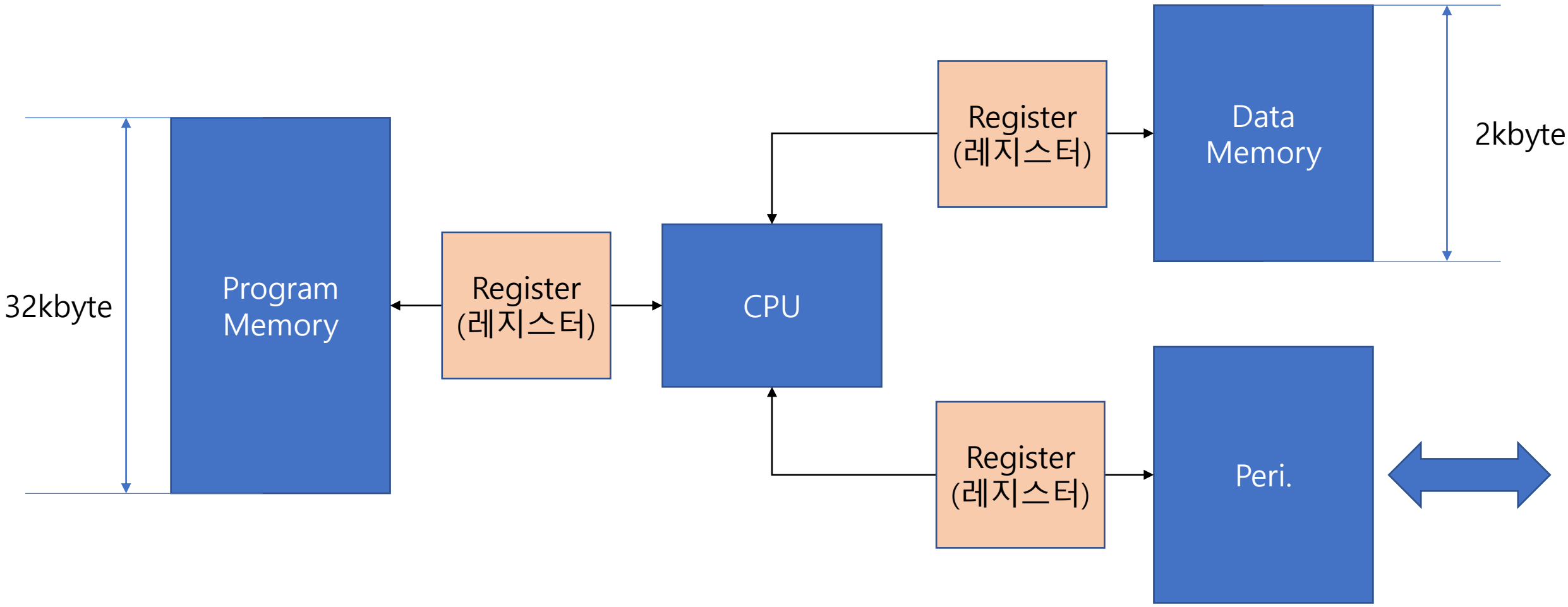
하버드 구조(Harvard architecture)



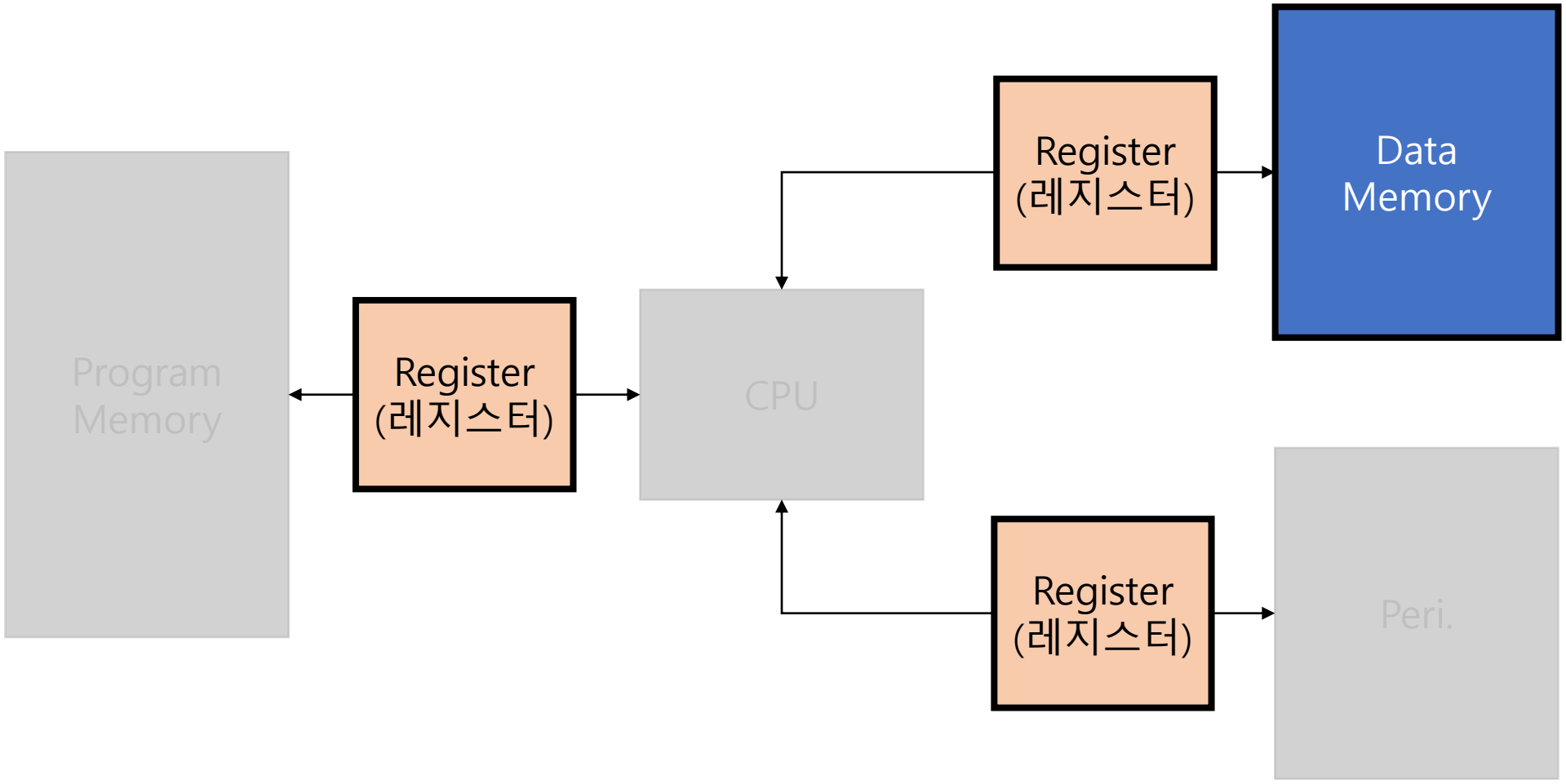
Atmega328p의 메모리



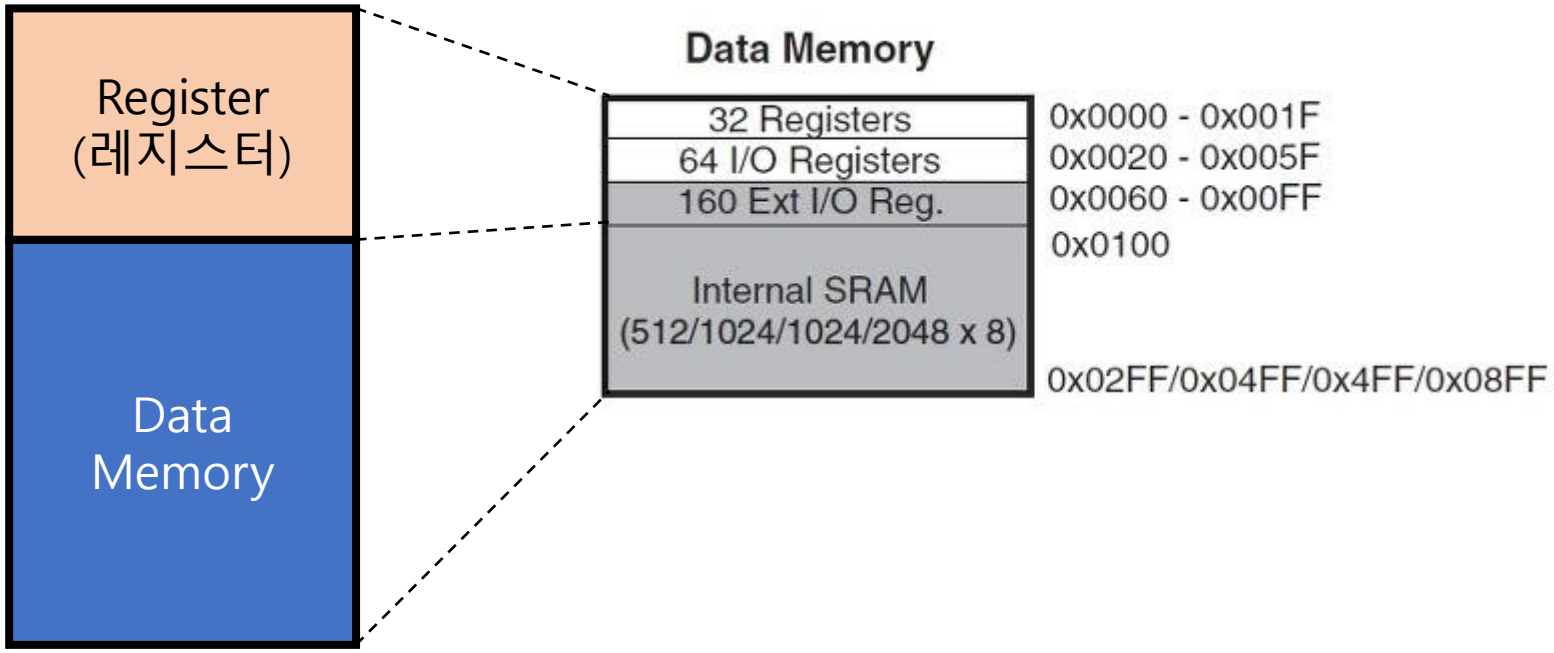
Atmega328p의 메모리 & 외부장치



Atmega328p의 메모리맵

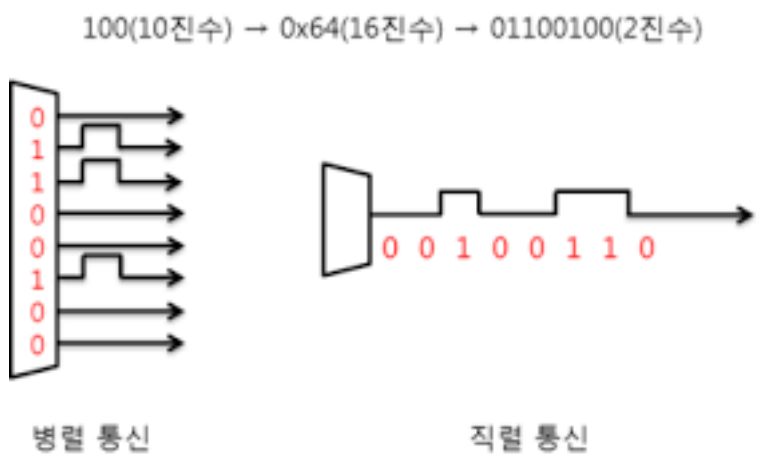


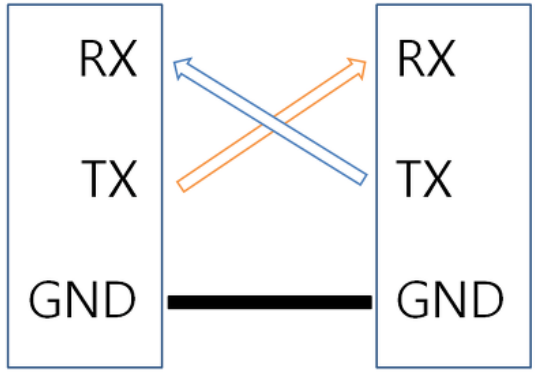
Atmega328p의 메모리맵



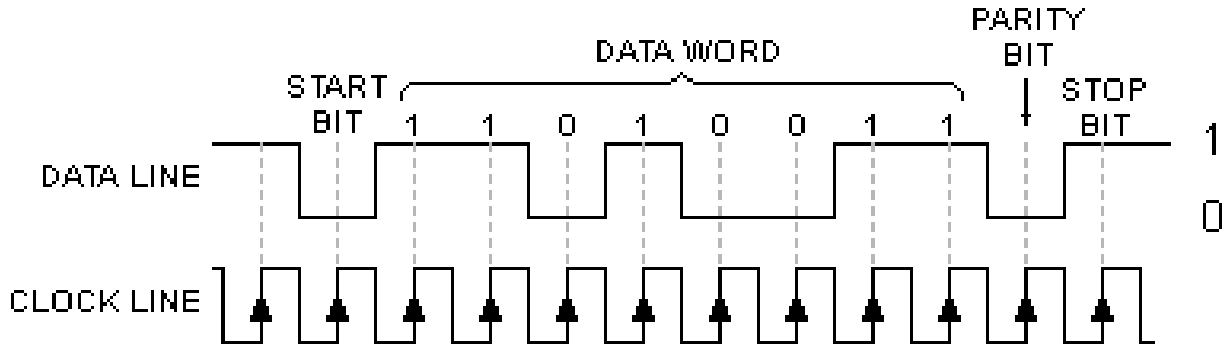
데이타시트 : https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
30. Register Summary, 275 page 참고

마이크로프로세서와 C언어 - 시리얼통신





직렬통신의 연결



동기/비동기 통신

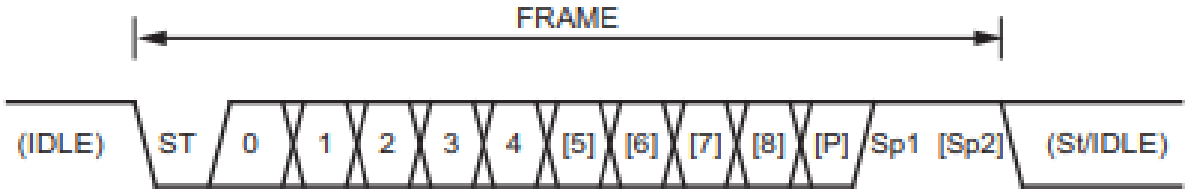
마이크로프로세서와 C언어 - 시리얼통신

- **비동기식 시리얼 통신**
  - UART 통신을 이용하기 위해서는 크게 다음의 두 가지 항목을 사전에 정의해줘야 한다.
    - **통신속도 : Baud rate**
      - Baud rate 의 단위는 bps(bits per second) : 1초당 전송하는 bit 수
      - 표준 bps: 1200, 2400, 4800, **9600**, 19200, 38400, 57600, 115200
    - **프레임사이즈 : Size of each frame field**
      - 일반적으로
        - Data bit는 1
        - Bytes site = 8 bits 사이즈로 설정
        - Stop bit는 1 bit
        - Parity bit는 0 bit로 설정
      - 통신을 사용하는 환경에 따라 미리 약속하여 사용

마이크로프로세서와 C언어 - 시리얼통신

- 시리얼통신 데이터 포멧

Figure 19-4. Frame Formats



St	Start bit, always low.
(n)	Data bits (0 to 8).
P	Parity bit. Can be odd or even.
Sp	Stop bit, always high.
IDLE	No transfers on the communication line (RxDn or TxDn). An IDLE line must be high.

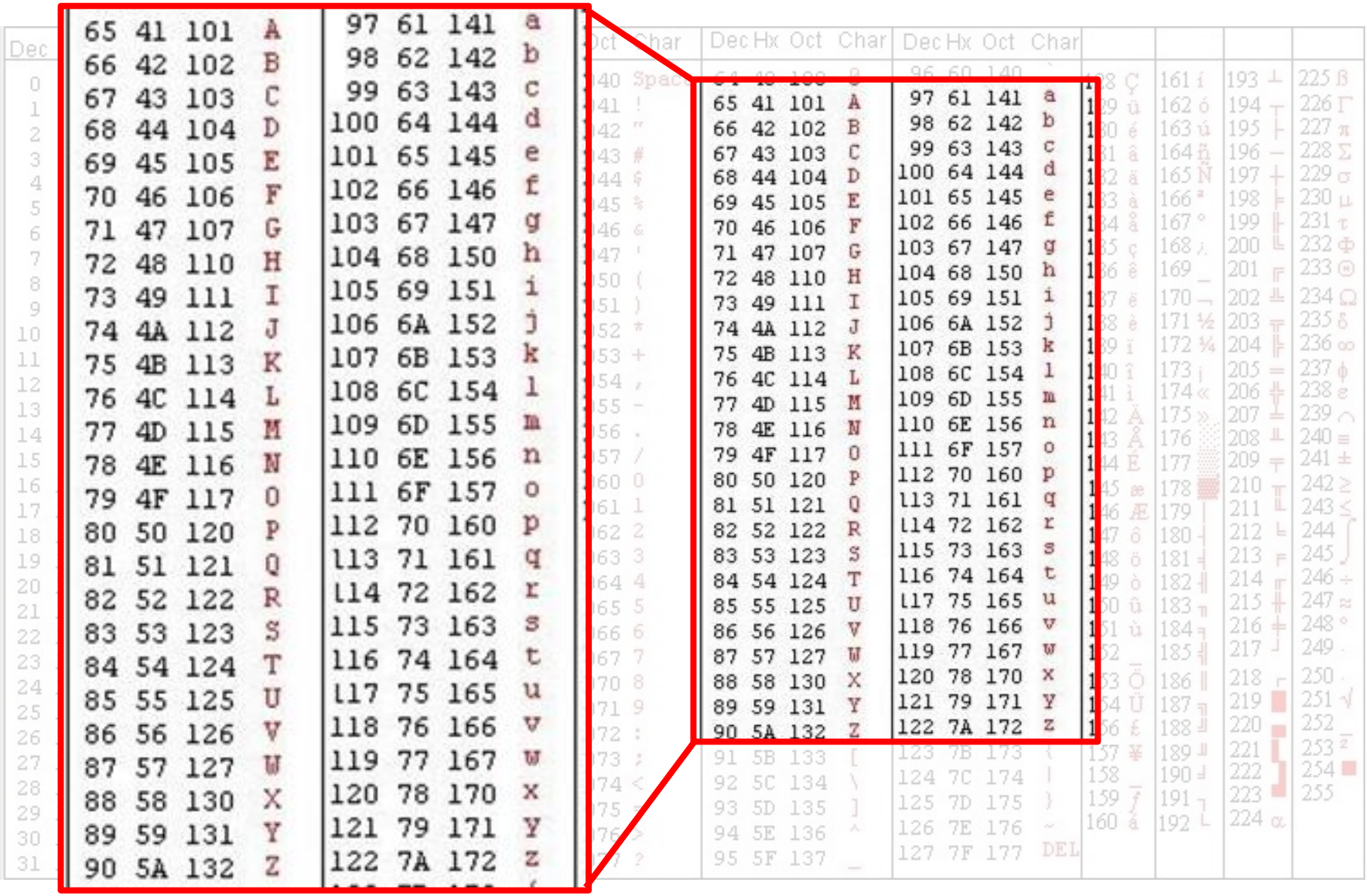
참고 : 데이타시트 https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf 147p

마이크로프로세서와 C언어 - ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Char	Dec	Hx	Oct	Char	Dec	Hx	Oct	Char
0	0	000	NUL (null)	32	20	040	Space	64	40	100	@	96	60	140	`
1	1	001	SOH (start of heading)	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX (start of text)	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX (end of text)	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT (end of transmission)	36	24	044	$	68	44	104	D	100	64	144	d
5	5	005	ENQ (enquiry)	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK (acknowledge)	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL (bell)	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS (backspace)	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB (horizontal tab)	41	29	051	)	73	49	111	I	105	69	151	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	74	4A	112	J	106	6A	152	j
11	B	013	VT (vertical tab)	43	2B	053	+	75	4B	113	K	107	6B	153	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	76	4C	114	L	108	6C	154	l
13	D	015	CR (carriage return)	45	2D	055	-	77	4D	115	M	109	6D	155	m
14	E	016	SO (shift out)	46	2E	056	.	78	4E	116	N	110	6E	156	n
15	F	017	SI (shift in)	47	2F	057	/	79	4F	117	O	111	6F	157	o
16	10	020	DLE (data link escape)	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1 (device control 1)	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2 (device control 2)	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3 (device control 3)	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4 (device control 4)	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN (synchronous idle)	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB (end of trans. block)	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN (cancel)	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM (end of medium)	57	39	071	9	89	59	131	Y	121	79	171	y
26	1A	032	SUB (substitute)	58	3A	072	:	90	5A	132	Z	122	7A	172	z
27	1B	033	ESC (escape)	59	3B	073	;	91	5B	133	[	123	7B	173	{
28	1C	034	FS (file separator)	60	3C	074	<	92	5C	134	\	124	7C	174	|
29	1D	035	GS (group separator)	61	3D	075	=	93	5D	135	]	125	7D	175	}
30	1E	036	RS (record separator)	62	3E	076	>	94	5E	136	^	126	7E	176	~
31	1F	037	US (unit separator)	63	3F	077	?	95	5F	137	_	127	7F	177	DEL

128 Ç	161 í	193 ┴	225 ß
129 ü	162 ó	194 ┬	226 Γ
130 é	163 ú	195 ├	227 π
131 â	164 ñ	196 ─	228 Σ
132 ä	165 Ñ	197 ┼	229 σ
133 à	166 ª	198 ╞	230 µ
134 å	167 º	199 ╟	231 τ
135 ç	168 ¿	200 ╚	232 Φ
136 ê	169 ⌐	201 ╔	233 Θ
137 ë	170 ¬	202 ╩	234 Ω
138 è	171 ½	203 ╦	235 δ
139 ï	172 ¼	204 ╠	236 ∞
140 î	173 ¡	205 ═	237 φ
141 ì	174 «	206 ╬	238 ε
142 Ä	175 »	207 ╧	239 ∩
143 Å	176 ░	208 ╨	240 ≡
144 É	177 ▒	209 ╤	241 ±
145 æ	178 ▓	210 ╥	242 ≥
146 Æ	179 |	211 ╙	243 ≤
147 ô	180 ┤	212 ╘	244 ⌠
148 ö	181 ╡	213 ╒	245 ⌡
149 ò	182 ╢	214 ╓	246 ÷
150 û	183 ╖	215 ╫	247 ≈
151 ù	184 ╕	216 ╪	248 °
152 ÿ	185 ╣	217 ┘	249 ∙
153 Ö	186 ║	218 ┌	250 ·
154 Ü	187 ╗	219 █	251 √
156 £	188 ╝	220 ▄	252 ⁿ
157 ¥	189 ╜	221 ▌	253 ²
158 ₧	190 ╛	222 ▐	254 ■
159 ƒ	191 ┐	223 ▀	255
160 á	192 └	224 α	

마이크로프로세서와 C언어 - ASCII



65	41	101	A
66	42	102	B
67	43	103	C
68	44	104	D
69	45	105	E
70	46	106	F
71	47	107	G
72	48	110	H
73	49	111	I
74	4A	112	J
75	4B	113	K
76	4C	114	L
77	4D	115	M
78	4E	116	N
79	4F	117	O
80	50	120	P
81	51	121	Q
82	52	122	R
83	53	123	S
84	54	124	T
85	55	125	U
86	56	126	V
87	57	127	W
88	58	130	X
89	59	131	Y
90	5A	132	Z

97	61	141	a
98	62	142	b
99	63	143	c
100	64	144	d
101	65	145	e
102	66	146	f
103	67	147	g
104	68	150	h
105	69	151	i
106	6A	152	j
107	6B	153	k
108	6C	154	l
109	6D	155	m
110	6E	156	n
111	6F	157	o
112	70	160	p
113	71	161	q
114	72	162	r
115	73	163	s
116	74	164	t
117	75	165	u
118	76	166	v
119	77	167	w
120	78	170	x
121	79	171	y
122	7A	172	z

0x48
0x65
0x6C
0x6C
0x6F

아두이노를 이용한 시리얼통신 실험

- 예제2

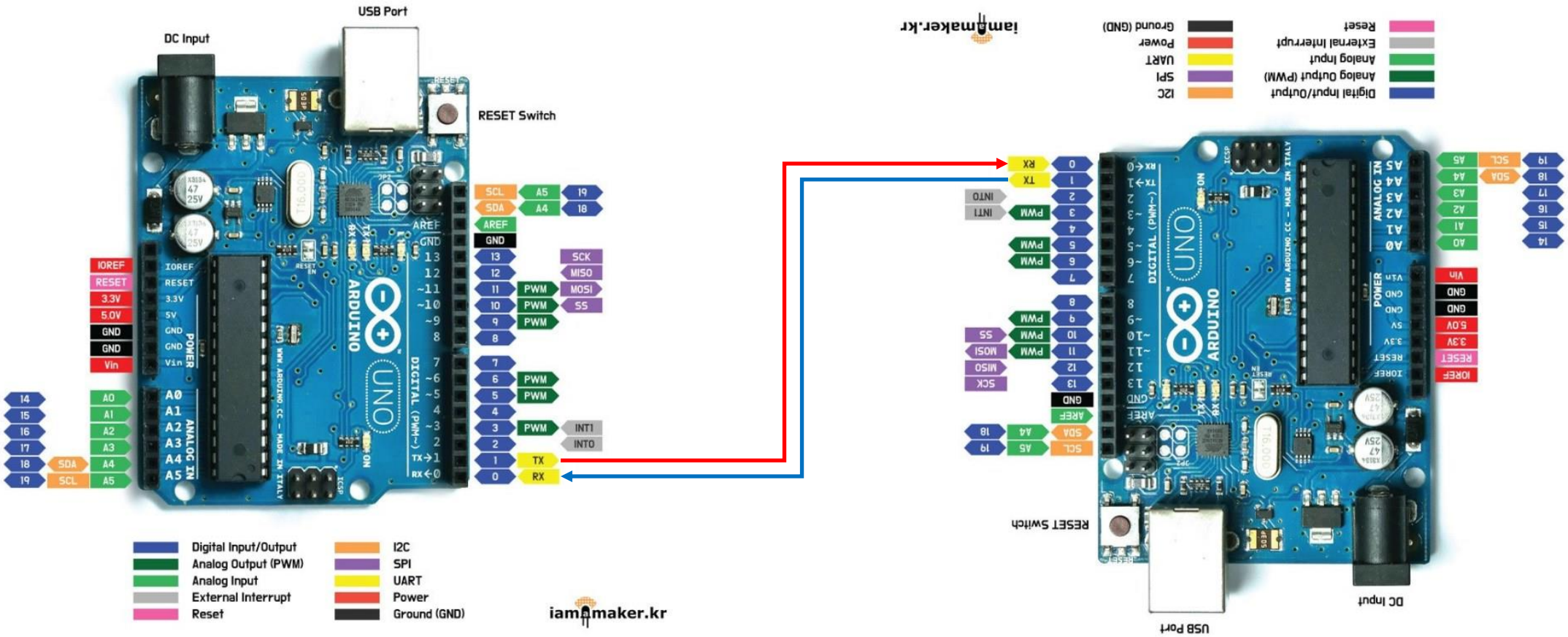
```
int incomingByte = 0; // for incoming serial data

void setup()
{
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop()
{
  // send data only when you receive data:
  if (Serial.available()) {
    // read the incoming byte:
    incomingByte = Serial.read();

    // say what you got:
    Serial.print("I received: ");
    Serial.println(incomingByte, DEC);
  }
}
```

아두이노를 이용한 시리얼통신 실험

- 예제3

```
void setup()
{
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop()
{
    Serial.print(char(0x48));
    Serial.print(char(0x65));
    Serial.print(char(0x6c));
    Serial.print(char(0x6c));
    Serial.print(char(0x6f));

    delay(1000) ;
}
```

아두이노를 이용한 시리얼통신 실험



마이크로프로세서와 C언어 - 변수

- 예제4

```
void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop()
{
  char c = 'a' ;
  int i = 10 ;
  unsigned int j = -10 ;
  float f = 1.24 ;
  double d = 1.234 ;

    Serial.print("char mem size= ");
    Serial.print(sizeof(c)) ;
    Serial.println(" byte") ;

delay(1000) ;
}
```

마이크로프로세서와 C언어 – 조건문 if

- 예제5

```
int incomingByte = 0; // for incoming serial data

void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  if (Serial.available()) {
    // read the incoming byte:
    incomingByte = Serial.read();

    if( incomingByte == 'a' ) {
        // say what you got:
      Serial.print("I received: ");
      Serial.println(incomingByte, DEC);
    }
  }
}
```

비교연산자
1. ==
2. !=
3. >
4. >=
5. <
6. <=

마이크로프로세서와 C언어 – if~else

- 예제6

```
int incomingByte = 0; // for incoming serial data

void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  // send data only when you receive data:
  if (Serial.available())
  {
    // read the incoming byte:
    incomingByte = Serial.read();

    if( incomingByte == 'a' )
    {
        // say what you got:
      Serial.print("I received: ");
      Serial.println(incomingByte, DEC);
    }
    else if( incomingByte == 'b' )
    {
         // say what you got:
      Serial.print("I received: ");
      Serial.println(incomingByte, DEC);
    }
  }
}
```

마이크로프로세서와 C언어 – switch~case

- 예제7

```
int incomingByte = 0; // for incoming serial data

void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  // send data only when you receive data:
  if (Serial.available())
  {
    // read the incoming byte:
    incomingByte = Serial.read();

    switch(incomingByte)
    {
    case 'a' :
      Serial.println("input a");
      break ;
    case 'b' :
      Serial.println("input b");
      break ;
    case 'c' :
      Serial.println("input c");
      break ;
    case 'd' :
      Serial.println("input d");
      break ;
    default :
      Serial.println(incomingByte, DEC);
      break ;
    }
  }
}
```

마이크로프로세서와 C언어 - 함수

- 예제8

```
void function1(void)
{
  Serial.println("function test");
}

void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  function1() ;
}
```

마이크로프로세서와 C언어 - 함수

- 예제9

```
void function2(char c)
{
  Serial.print(c);
  Serial.print("Decimal Value = ");
  Serial.println(c, DEC);
}

void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  function2('k') ;
}
```

마이크로프로세서와 C언어 - 함수

- 예제10

```
int function_add(int a, int b)
{
    int c = a+b ;
    return c ;
}

void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  int result = function_add(1, 4) ;
  Serial.print("result = ") ;
  Serial.println(result) ;
}
```

마이크로프로세서와 C언어 - 반복문

- while 문
- do~while 문
- for 문

마이크로프로세서와 C언어 - 반복문

- 예제11

```
void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  int condition = 1 ;
  unsigned int count = 0 ;
  while(condition)
  {
    Serial.print("count = ") ;
    Serial.println(count) ;

    count++ ;

    if( count > 100 ) condition = 0 ;
  }
}
```

마이크로프로세서와 C언어 - 반복문

- 예제12

```
void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  int condition = 1 ;
  unsigned int count = 0 ;
  do
  {
    Serial.print("count = ") ;
    Serial.println(count) ;

    count++ ;

    if( count > 100 ) condition = 0 ;
  }while(condition);
}
```

마이크로프로세서와 C언어 - 반복문

- 예제13

```
void setup() {
  Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}

void loop() {
  int i = 0 ;
  for( i = 0 ; i<100 ; i++ )
  {
    Serial.print("i = ") ;
    Serial.println(i) ;
  }
}
```

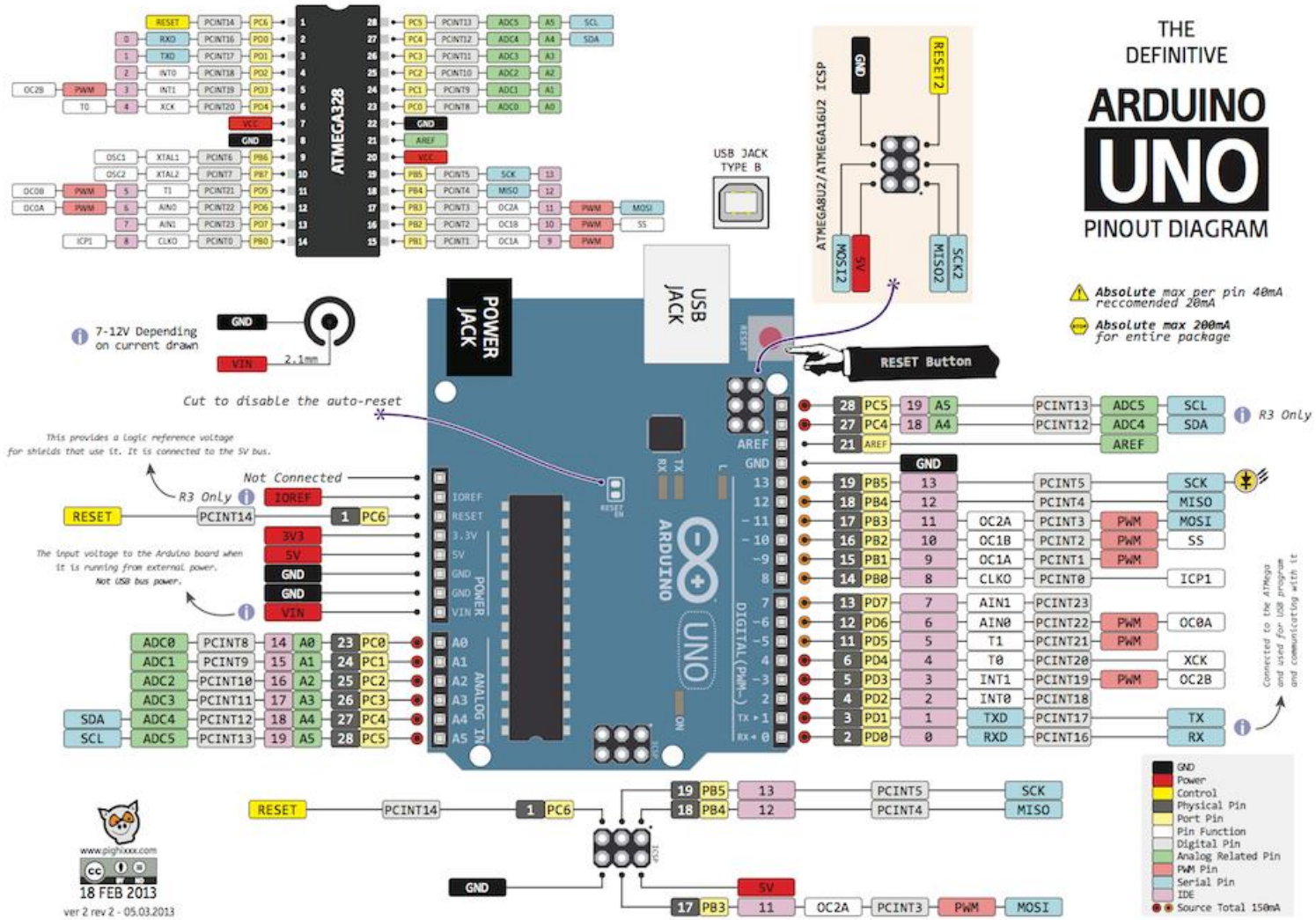
마이크로프로세서와 C언어 - 반복문

- 퀴즈 : for 문을 이용하여 1부터 100까지 더한 결과를 얻는 기능을 프로그래밍 하시오

- Hint : 예제13과 예제10번을 참고

IO 포트

- Port


THE DEFINITIVE
ARDUINO UNO
PINOUT DIAGRAM
ATMEGA328
ATMEGA8U2/ATMEGA16U2 ICSP
USB JACK TYPE B
POWER JACK
USB JACK
RESET Button
7-12V Depending on current drawn
Cut to disable the auto-reset
Not Connected
R3 Only
Absolute max per pin 40mA reccomended 20mA
Absolute max 200mA for entire package
18 FEB 2013
ver 2 rev 2 - 05.03.2013

IO 포트 관련 레지스터

- Port

13.4.2 PORTB – The Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

13.4.3 DDRB – The Port B Data Direction Register

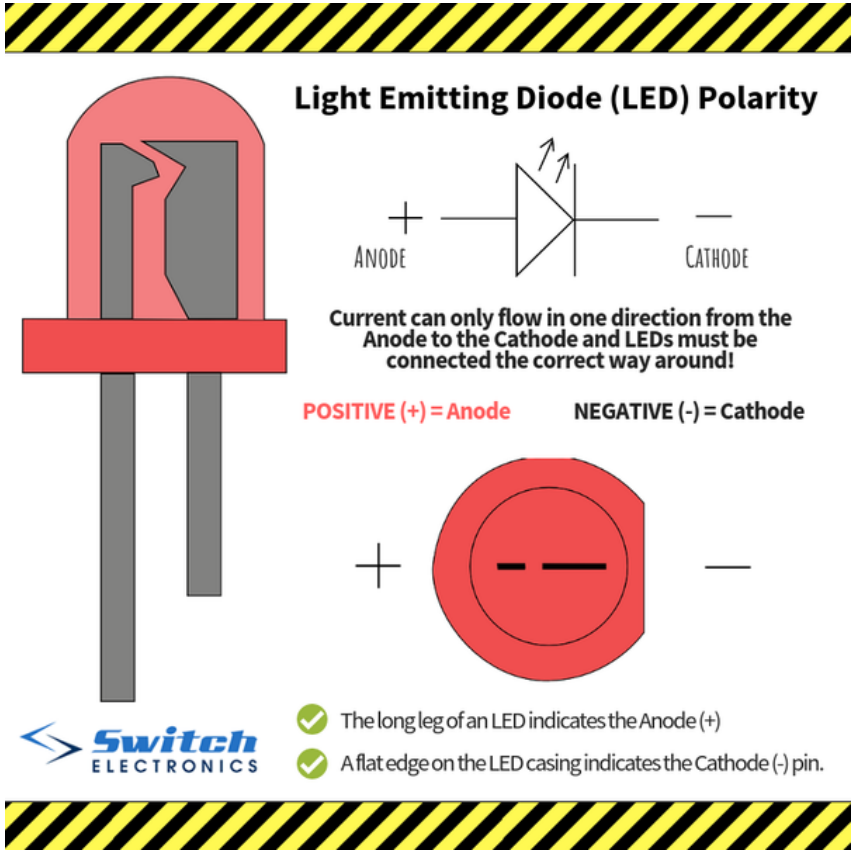
Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

13.4.4 PINB – The Port B Input Pins Address

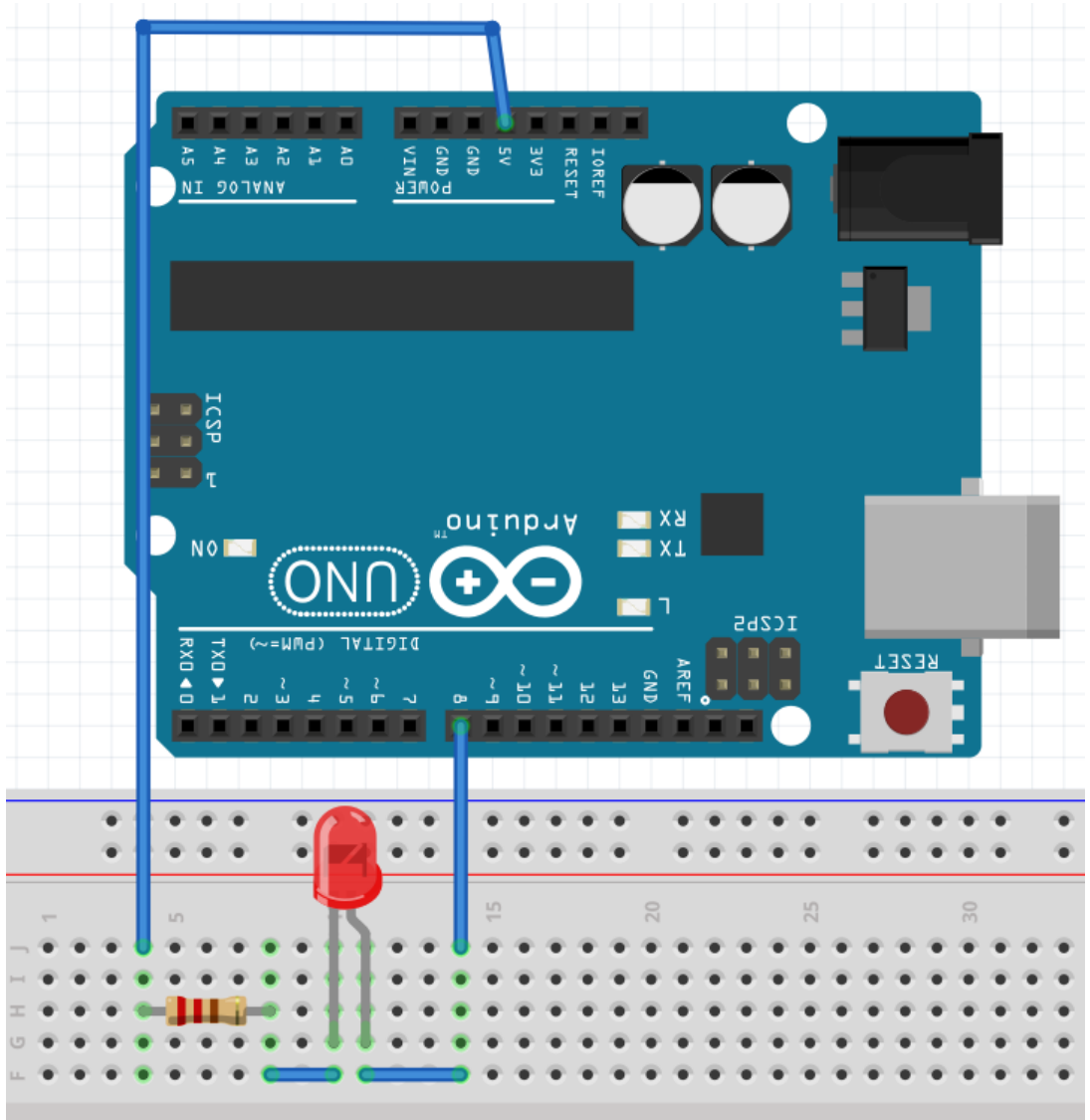
Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

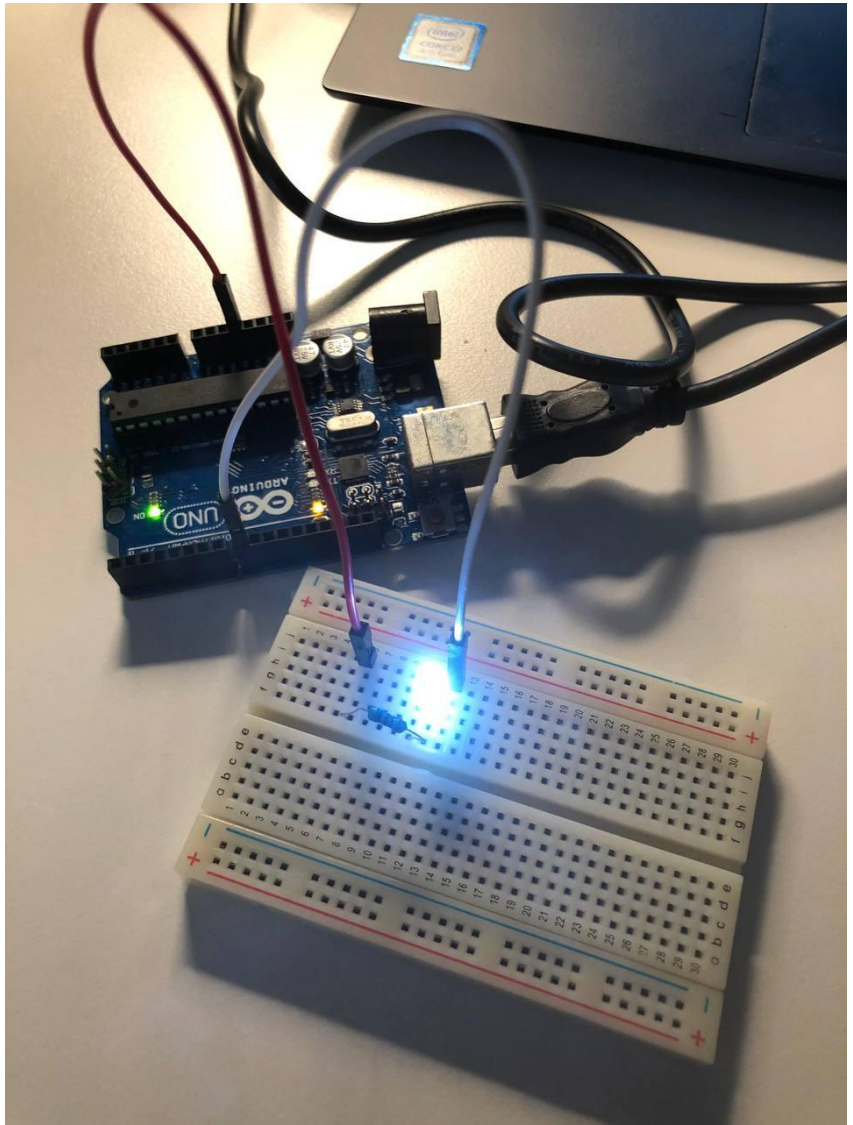
IO 포트 테스트

- LED를 이용한 포트 Output 테스트



IO 포트 테스트


ANALOG IN
POWER
DIGITAL (PWM=~)
ICSP
ICSP2
RESET
Arduino
UNO
ON
TX
RX
L
AREF
GND
5V
3V3
VIN



IO 포트 테스트

- 예제14

```
void setup() {
  // put your setup code here, to run once:
  DDRB = B00000001 ;
  PORTB = B00000000 ;

}

void loop() {
  // put your main code here, to run repeatedly:
  PORTB = B00000001 ;
  delay(1000) ;
  PORTB = B00000000 ;
  delay(1000) ;
}
```

수고하셨습니다.

다음주에 만나요.