C언어 - HW8

임베디드스쿨2기
Lv1과정
2021. 05. 13
차현호

이번에는 ATMEGA 328P를 컨트롤 하여 LED를 0.5초 간격으로 깜빡이도록 제어 해보는 프로그램을 짜보자.

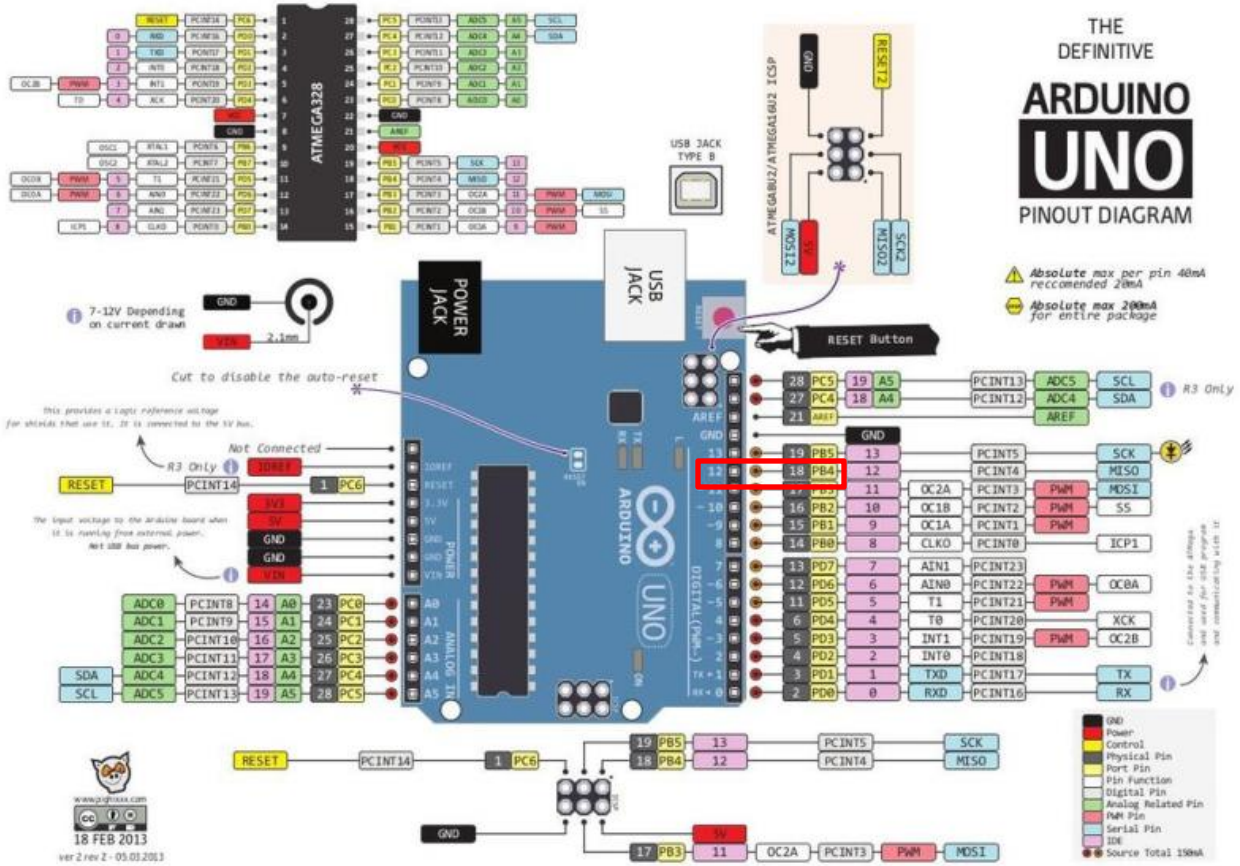
먼저 현재 아두이노 보드에서 사용하는 Crystal 클럭 주파수는 16MHz이다.
따라서 소스파일에 #define F_CPU 16000000을 추가해준다.

그다음 main문 안에서 DDRB = 0x20;을 추가 해준다.
그렇다면 DDRB는 무엇인지 ATMEGA328P data sheet를 살펴보자.

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	

DDRB레지스터의 설명을 보면 포트B의 입출력 방향을 결정하는 레지스터라는 것을 확인할 수 있다. 레지스터 값을 0으로 하면 input 1로 하면 output으로 설정 된다. 우리는 DDRB를 0x20으로 설정하였으므로 PB5를 output으로 설정한것을 확인할 수 있다.
그러면 아두이노 우노 보드에서 PB5포트는 몇번 핀인지 확인해 봐야한다.



이전 슬라이드에서 빨간색 박스안을 살펴보면 PB5는 아두이노 우노에서 13번 핀이라는것을 확인할 수 있다.

즉 정리하자면 DDRB = 0x20; 코드는 아두아이노 우노 보드 13번핀을 output으로 사용한다는 의미이다.

포트의 입출력설정을 완료한 다음에 for(;;) 무한루프문 안에서 PORTB ^= 0x20;을 해준다.
PORTB레지스터를 살펴보면 다음과 같다.

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	

설명을 읽어보면 포트B의 출력 데이터를 결정한다. 1이면 3.3V 출력, 0이며 0V출력이 나간다.
PORTB ^= 0x20; 에서 0x20 값을 xor한 이유는 코드가 실행된때마다 PORTB레지스터의 4번째 비트를 0->1, 1->0으로 바꾸기 위해서이다.

깜빡이는 코드를 완성하였으니 0.5c초의 delay를위해 _delay_ms(500)을 추가해 준다.
위의 함수는 함수인자의 값만큼 딜레이를 준다.

기본적인 깜빡임 코드를 작성하였으니 스위치 입력을 인터럽트로 받아 LED를 토글하는 코드를 짜보자.

스위치를 입력받는 핀은 아두이노 우노보드의 13번핀을 스위치 입력으로 12번핀을 LED출력으로 설정하였다.

12번 핀을 LED 출력으로 하려면 DDRB레지스터의 4번째 비트를 1으로 설정해준다.

그런다음 PCICR의 0번째 비트를 1로 셋팅해준다 PCICR레지스터는 아래와 같다.

12.2.4 PCICR – Pin Change Interrupt Control Register

Bit	7	6	5	4	3	2	1	0	
(0x68)	–	–	–	–	–	PCIE2	PCIE1	PCIE0	PCICR
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..3 - Res: Reserved Bits**

These bits are unused bits in the Atmel® ATmega328P, and will always read as zero.

- **Bit 2 - PCIE2: Pin Change Interrupt Enable 2**

When the PCIE2 bit is set (one) and the I-bit in the status register (SREG) is set (one), pin change interrupt 2 is enabled. Any change on any enabled PCINT23..16 pin will cause an interrupt. The corresponding interrupt of pin change interrupt request is executed from the PCI2 interrupt vector. PCINT23..16 pins are enabled individually by the PCMSK2 register.

- **Bit 1 - PCIE1: Pin Change Interrupt Enable 1**

When the PCIE1 bit is set (one) and the I-bit in the status register (SREG) is set (one), pin change interrupt 1 is enabled. Any change on any enabled PCINT14..8 pin will cause an interrupt. The corresponding interrupt of pin change interrupt request is executed from the PCI1 interrupt vector. PCINT14..8 pins are enabled individually by the PCMSK1 register.

- **Bit 0 - PCIE0: Pin Change Interrupt Enable 0**

When the PCIE0 bit is set (one) and the I-bit in the status register (SREG) is set (one), pin change interrupt 0 is enabled. Any change on any enabled PCINT7..0 pin will cause an interrupt. The corresponding interrupt of pin change interrupt request is executed from the PCI0 interrupt vector. PCINT7..0 pins are enabled individually by the PCMSK0 register.

설명을 확인해보면 0번째 비트를 1로 셋팅하면 PCINT 0 ~ 7 Pin들을 enable하는것을 확인 할 수있다.

우리가 스위치 입력으로 받을 13번핀은 PCINT5 Pin인것을 아두이노 우노 핀맵에서 확인 할 수 있다.

PCINT 0 ~ 7 Pin을 활성화 시켰으니 그다음 PCMSK0 레지스터에가서 PCINT5 비트를 1로 세팅하여 인터럽트를 enable한다.

위의 설정들을 완료한 다음에 sei(); 함수를 호출 하여 인터럽트를 실행한다.

그렇다면 인터럽트가 발생하였을 때는 어느부분이 실행될까?
인터럽트가 발생했을때 실행되는부분을 인터럽트 서비스루틴 줄여서 ISR이라고 부른다.

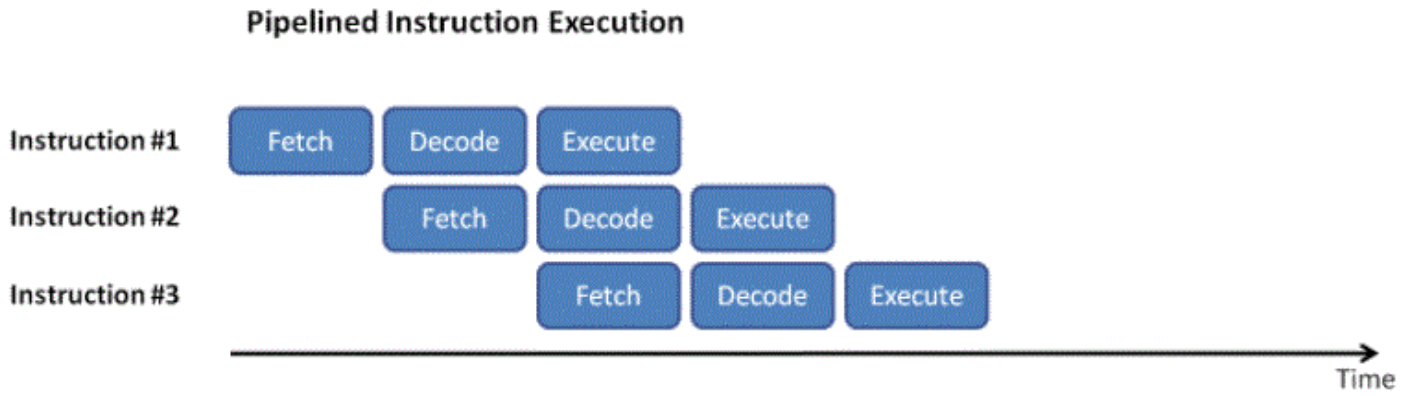
```
ISR (PCINT0_vect)
{
    PORTB ^= 0x10;
}
```

Pipeline

※ Pipeline 이란
한 데이터 처리 단계의 출력이 다음 단계의 입력으로 연속적으로 이어지는 형태로 연결되는 구조를 말한다.

비유를 하면 공장의 조립라인과 유사하다. 한개의 물건을 만들 때 조립라인이 있다고 하면 각 단계별로 진행된다음에 제품이 완성될때까지 기다리는것이 아니라 바로 다음 제품을 조립하는 것으로 이해하면 된다.

이를 cpu의 관점에서 생각하면 cpu가 3 stage pipeline을 사용하고 있다면 아래 그림과 같이 Fetch -> Decode Excute 순으로 실행된다. 이전 조립라인의 비유에 설명했던것처럼 한 단계의 할일이 끝나면 바로 다음단계를 실행하는것을 확인할 수 있다.



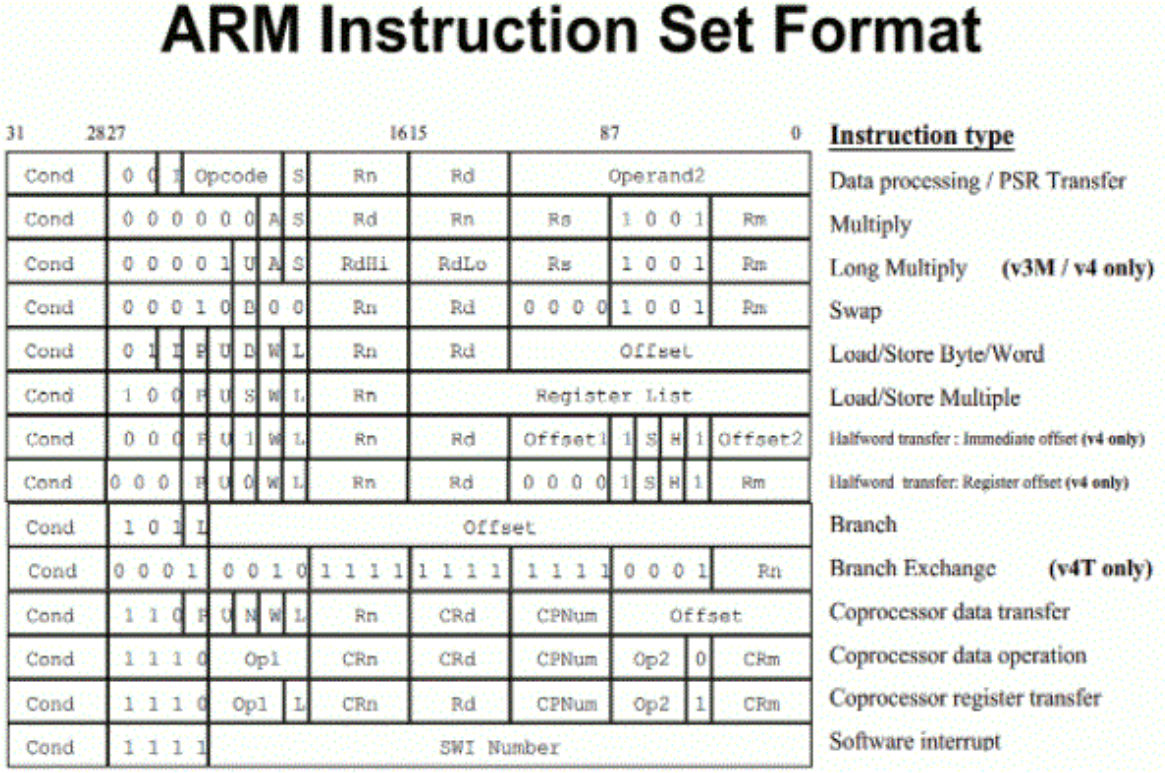
출처 : https://microchipdeveloper.com/32bit:mx-arch-pipeline

Fetch

※ Fetch란
Pipeline 처리과정에서 가장 먼저 실행되는 단계로 다음에 실행될 명령어를 가져온다.

즉 다음에 실행된 Instruction을 메모리부터 가져온다. 이때 다음에 실행될 명령어는 Program Counter 레지스터 (Arm : r15, Intel : rip)안에 있는 값이다.

Instruction이란 cpu가 실행할 동작을 명령하는 명령어이다. 32bit 시스템에서 Instruction의 크기는 32bit 이며 아래 그림과 같다.

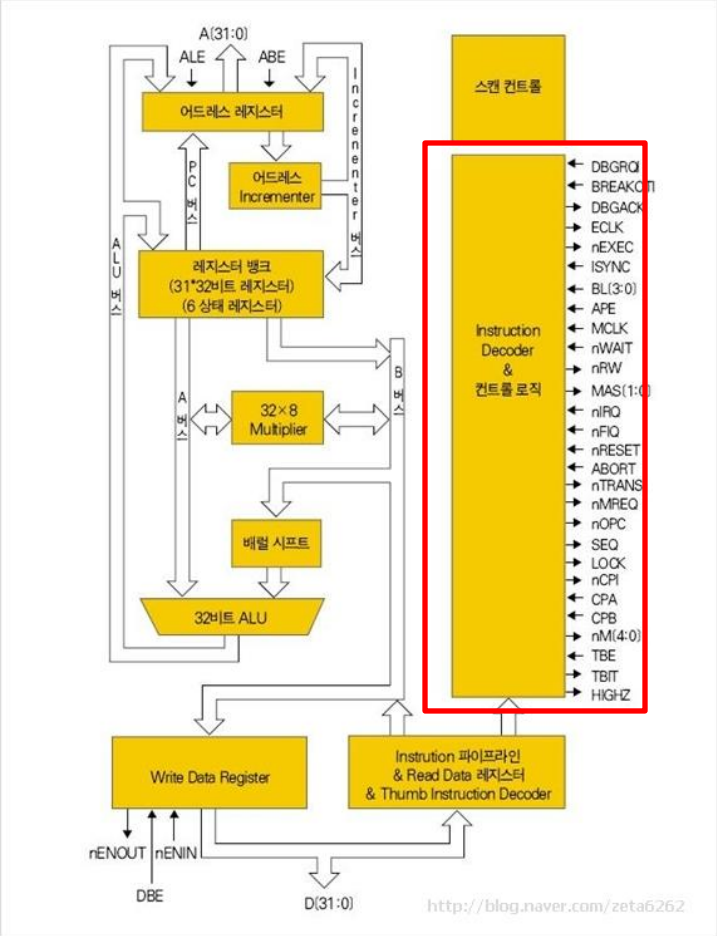


출처 :
https://www.cs.uaf.edu/courses/cs301/2014-fall/notes/arm-asm/

Decode

※ Decode란
앞에 Fetch 단계에서 가져온 32bit Instruction을 Decoder라는 해독기에서 프로세서가 이해할수 있는 기계어로 해독하여 주는 단계이다.

이렇게 해독된 후에는 control unit에서 동작에 맞는 신호들을 내보내어 준다.



옆의 그림은 ARM7 프로세서의 다이어그램을 나타낸것이다. 보면 Fetch해온 명령어를 Decoder에서 해독하여 Control 시그널들을 발생시키는것을 확인 할 수 있다.

Execute

※ Execute란
Decode 단계에서 해독된 명령어를 실행하는 단계이므로 이단계에서는 ALU에서 실제 연산이 실행 된다.

5 stage pipeline

앞의 Pipeline의 stage는 Fetch, Decode, Execute 총 3단계의 stage가 있었다.
하지만 ARM9 아키텍쳐부터는 Execute 단계 이후에 Memory Access와 RegWrite 단계가 추가된 5 단계 pipeline을 사용한다.

图 3.4. ARM7三级流水线 vs ARM9五级流水线

Figure 1 : The ARM7TDMI core and ARM7TDMI-S core pipeline

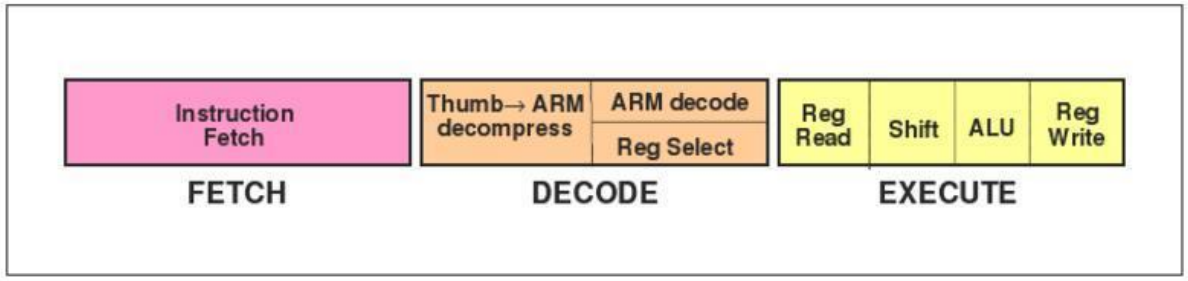
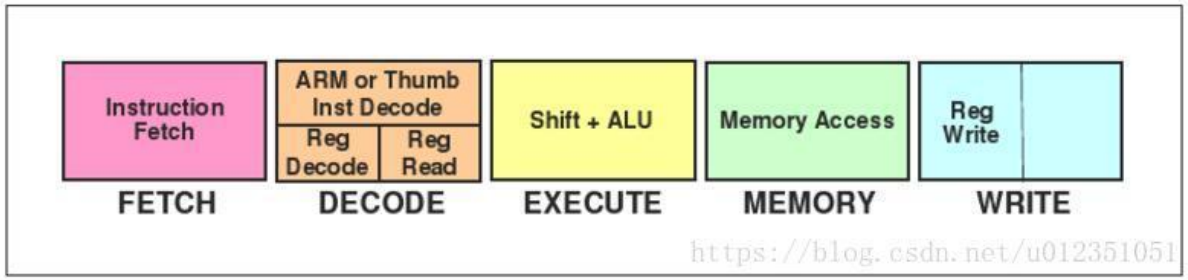
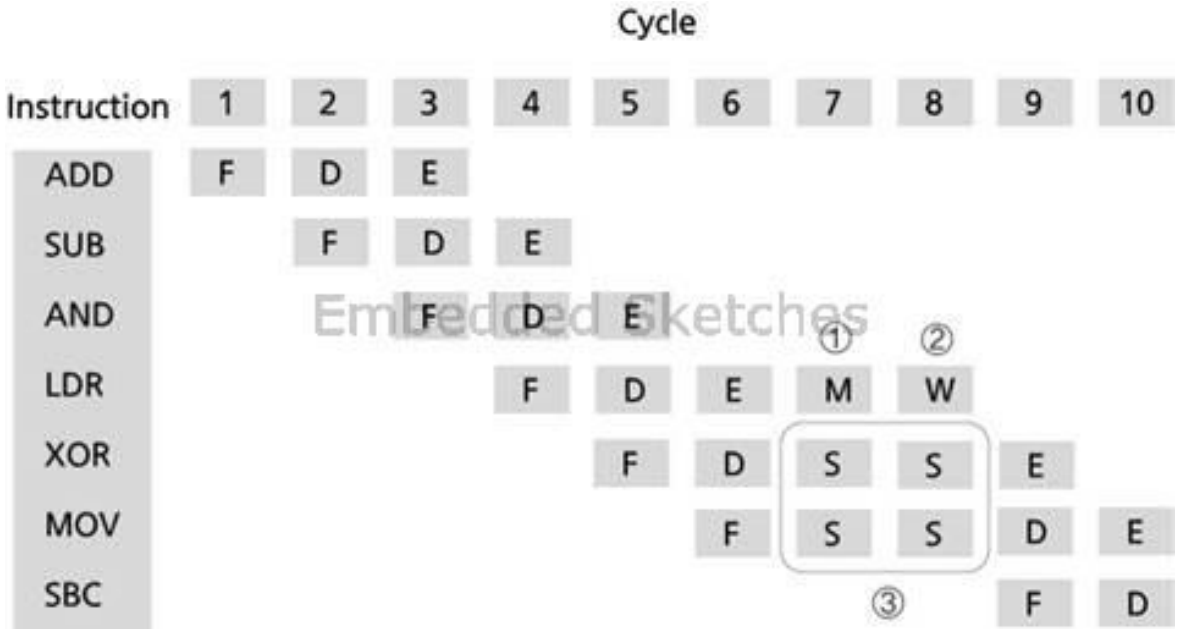


Figure 2 : The ARM9TDMI core pipeline



Memory Access

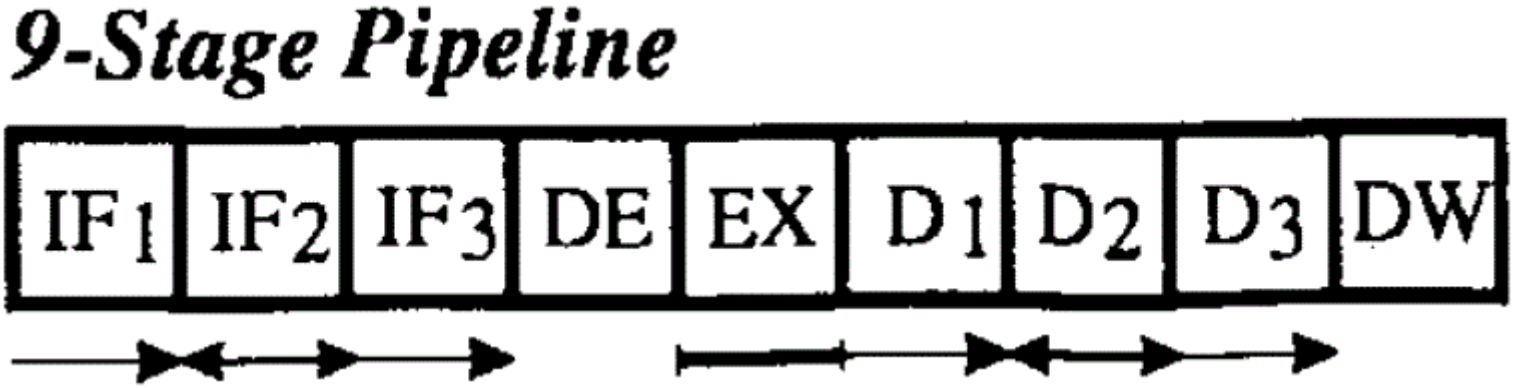
※ Memory Access란
3 stage pipeline에서는 없는 단계이며 3 stage pipeline의 메모리 접근시 발생되는 stall 현상으로 인한 성능 저하를 막기 위해 Memory Access 단계가 추가되었다.



메모리를 억세스 하는 단계에서 stall이 발생하여 pipe라인이 제대로 동작하지 않는것을 확인할 수 있다.

9 stage pipeline

다음으로는 9단계 pipeline에 대해 알아보면 앞에서 5 단계 pipeline에서 나왔던 단계외에 새로운 기능이 적용된 단계는 없고 Fetch와 Memory Access를 2번씩 추가적으로 실시한다.



9 stage pipeline

9단계 파이프라인의 장점은 아래논문에 나와있으면 그림과 같다.

(PDF) Design of a 32 b monolithic microprocessor based on GaAs HMESFET technology (researchgate.net)

four-stage and nine-stage pipeline. The nine-stage pipeline has the lowest cache miss penalties, but it suffers from large branch/load penalties. The four-stage pipeline has the lowest branch/load penalties

글을 확인해보면 가장 낮은 캐시 미스 패널티를 가지고 있다고 나와있으며 단점으로는 브랜치시에 가장 높은 패널티를 가지고 있다고 나와있다.

CPI CONTRIBUTIONS DUE TO BRANCH AND LOAD PENALTIES

Instruction	*Instruction frequency* [1]	*Cost 4- stage*	*Cost 7-stage*	*Cost 9-stage*
ALU	60%	1	1	1
BRANCH (not taken)	5%	1	1	1
BRANCH (taken) [2]	15%	1.368	2.673	3.673
LOAD [3]	15%	1.1	1.5	2.3
STORE	5%	1	1	1
$CPI_{intrinsic}$		1.07	1.326	1.596

실제 실험한데이터 표를 나타낸다.

질문

Q1. ISR 함수 작성시 PCINT0_vect 가 인자로 들어가는데 스위치로 입력받는 핀은 PCINT5인데 PCINT5_vect를 인자로 받아야하는거 아닌가요?

Q2. 금주에 개인사정으로 숙제가 많이 미비합니다. 좀 더 보충해서 채워 넣겠습니다.