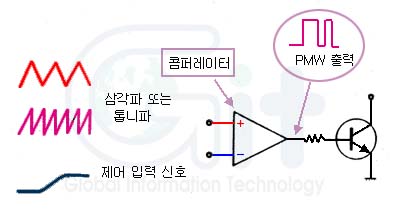
|  |  |  |
| --- | --- | --- |
|  | **PWM 제어**  LED를 키고 끄는 것은 쉽다.  하지만 LED의 밝기는 어떻게 조절할까?  그 때 이용하는 방법이 PWM이다.  PWM은 LED의 밝기 뿐만 아니라 모터 회전속도 제어, 서보 위치제어 등 에도 쓰인다.    PWM 이란?  Pulse Width Modulation 의 줄임말로  전압신호의 Pulse (전압 파형) 를 이용하여 원하는 전압신호 평균값을 출력으로 만들어내는 것이다.  http://postfiles3.naver.net/20140829_82/ejtkddl_1409241164429QBval_JPEG/atmega168a_pwm_02_med.jpg?type=w3    위 그림에서처럼 하나의 Period 내에 전압의 High 신호 구간을 조정하여 평균 전압값을 만들어낸다.  한 Period내의 High신호의 구간비율을 Duty Ratio라 부른다.    예를들어 10초의 주기로 전압신호를 만들어낸다 하자. 이때 2초간 5V전압을 가하고 8초간 0V로 내리고 또 2초동안 5V 가하고 8초간 0V 내리고.... 이런식으로 하면 PWM의 주기는 10초이고 Duty Ratio는 20%가 된다.    이러한 전압파형 신호를 만들어내는 방법은 여러가지가 있다.  개념만 간단히 얘기하자면  10초동안 1부터시작해서 1초에 1씩 더해 10까지 수를 센다.  1, 2, 3, 4, 5, 6, 7, 8, 9, 10  내가 10을 세었다면 그건 10초가 흘렀다는 것을 의미한다.  10초가 지나면 다시 1부터 10까지 센다. 그렇게 계속 1~10, 1~10, 1~10 ... 을 반복한다.  여기에 누군가가 나에게 "2"라고 말했을 때  나는 처음부터 전압을 가하다가 "2"를 셀 때 전압을 끊어야한다.  10을 만나고 다시 1로 넘어갈때 나는 다시 전압을 가하고 다시 2를 만나면 전압을 푼다.  이런식으로 반복하다 보면 Duty Ratio 20%를 만들 수 있는 것이다.    위 예시에서 "나"는 마이컴이 될 것이고 누군가가 나에게 말한 숫자 "2"는 user가 마음대로 정하는 비교 숫자가 된다.  (위 방법 말고도 다른 여러가지 방법도 많다)    좀더 자세하게 알아보자.    ATmega128에서는 타이머/카운터0, 1, 2, 3 가 있다.  타이머/카운터는 시스템 클럭을 이용하여 시간의 흐름에 따라 수를 세는 시계같은 놈이다.  이놈들을 이용한다면 마이컴이 숫자를 세다가 내가 숫자를 주면 그 수를 경계로 전압을 끊거나 풀거나 할 수 있게 된다.  타이머/카운터 0,2의 경우 셈수를 저장하는 레지스터가 8비트이므로  0~255까지 셀 수 있다.  타이머/카운터 1,3의 경우 셈수를 저장하는 레지스터가 16비트이므로  0~65535까지 셀 수 있다.  (셀 수 있는 숫자의 크기가 곧 "분해능" 또는 신호의 "해상도"로 연결된다)    또한 파형을 만드는 알고리즘도 여러가지 있다.  ATmega128이 지원하는 PWM 파형 생성 기능은  타이머/카운터 0, 2를 이용할 시  Fast PWM, Phase Correct PWM (PC PWM),  타이머/카운터 1, 3을 이용할 시  Fast PWM, Phase Correct PWM (PC PWM), Phase and Frequency Correct PWM이 있다.    (이러한 기능 말고도 내가 직접 파형을 간단히 만들 수도 있다.    Pin = 1; delay\_ms(20); Pin = 0; delay\_ms(80); 이런식으로 무한루프를 돌리면 Period 100ms에 20%    duty ratio를 만들 수도 있는 것이다. 하지만 이러한 방법은 너무 불안정하므로 정확성이 요구되지 않    는 간단한 LED 밝기 조절 연습정도에 쓰이는 방법이다.)    위에서 언급한 몇가지 PWM 파형 생성방법의 각 알고리즘은 Datasheet나 서적같은걸 뒤지면 아주 자세하게 다 나온다.    타이머/카운터0의 Fast PWM 모드를 예로 들어 설명하자면  http://postfiles8.naver.net/20140829_135/ejtkddl_1409243722464pJwhw_PNG/fast_pwm_example-335x300.png?type=w3  (TCNT0 : 카운터0가 센 수를 저장하는 8비트 레지스터)  (OCR0A : 타이머/카운터0 - 채널 A의 출력 비교 레지스터)  (OCR0B : 타이머/카운터0 - 채널 B의 출력 비교 레지스터)    수를 세는 방법 : 각 Period당 0부터 시작해서 Up Counting 하여 255까지 수를 센다.  OC0A pin에서 비교할 출력비교레지스터 OCR0A를 191로 넣으면 위 그림의 초록색 파형이 생성되고  OC0B pin에서 비교할 출력비교레지스터 OCR0B를 63로 넣으면 위 그림의 파란색 파형이 생성된다.  (이런식으로 같은 타이머에 여러 채널이 나눠져 각각 다른 파형이 생성됨으로써 다중 제어가 가능) | |
|  | |

**PWM 제어란**

우선 효율이 좋은 스위칭 제어의 기본이 되어 있는 PWM 기술부터 알아보자.

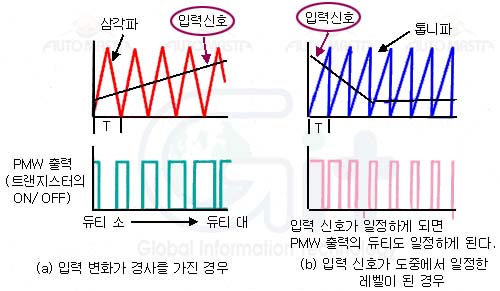
**PWM** 이란 **P**ulse **W**idth **M**odulation (펄스 폭 변조) 의 머리글자를 딴 것으로 그림 1 에 기본적이 구성을 나타낸다. 그림에 있어서 콤퍼레이터 (진압 비교 회로 또는 OP 앰프) 의 (+) (논인버팅) 입력에 삼각파 또는 톱니파를 입력하고 (-) (인버팅) 입력에 제어 신호를 입력한다. **그러면 콤퍼레이터의 출력에는 입력 신호의 레벨 변화에 따라서 펄스 폭의 다른 출력이 나타난다. 이것이 PWM (펄스 폭 변조)이라는 것이다.**



**그림 1 : PWM 제어의 사고 방식**

|  |  |  |
| --- | --- | --- |
| **입력 조건** | **콤퍼레이터 출력** | **트랜지스터** |
| + 입력 단자 전압 >  - 입력 단자 전압s | "H" | ON |
| + 입력 단자 전압 <  - 입력 단자 전압 | "L" | OFF |

이 PWM 출력과 입력 신호와의 관계를 그림 2에 나타낸다. 입력 신호가 삼각파 (또는 톱니파) 신호보다도 높은 경우는 컴퍼레이터 출력으로 드라이브되면 트랜지스터는 ON 하고 반대의 경우는 OFF 한다. 또 삼각파 신호와 입력 신호의 입력 단자를 반대로 하면 비교의 결과도 반대로 된다.



**그림 2 :  PWM 제어 파형**

그리고 PWM 즉, 펄스 폭 변조는 주기가 일정하고 입력 신호 (DC 레벨)의 크기에 따라서 펄스 폭의 듀티 사이클 (펄스 폭의 "H" 와 "L" 의 비) 이 변화하는 것을 의미하고 있다. 그리고 출력 트랜지스터가 ON 하고 있는 동안이 모터에 전류가 흐르고 있는 기간이므로 DC 모터 등이라면 컬렉터 쪽에 모터를 접속하고 입력 신호의 레벨을 변화하는 것만으로 스피드 컨트롤을 간단히 할 수 있다는 셈이다. 더구나 트랜지스터는 ON/OFF 의 포화 스위칭을 반복하고 있을 뿐이므로 스위칭 손실을 제외하면 거의 전력 소비의 낭비가 없는 동작이 실현되는 것이다.

그러나 스테핑 모터의 경우는 주어지는 펄스 수에 의해 속도가 변화할 뿐이고 여자상의 전류를 PWM으로 한 곳에서 모터의 속도가 변하는 것이 아니다. 스테핑 모터에서의 PWM 제어의 목적은 모터의 속도를 제어하는 것이 아니고 전류의 상승 개선과 전원의 효율 개선, 코일 전류의 정전류화 등이 있는 것이다. **그래서 이러한 회로 구성을 정전류 초퍼라고 한다.**