

**전자회로실험 보고서**

학과 전자공학과

학번 1218165 / 12161509

이름 김재욱 / 최장호

제출일 2020-11-17

이론

**실습 이론**

-DC Motor

DC전류를 기반으로 회전하는 모터, 모터 둘레에 자석이 둘러써야 있고 내부에 코일로 둘러싸인 심이 존재

코일에 DC전류를 흘려주면 전자기 유도현상으로 자기장이 발생하고 인력과 척력에 의해 회전하는 원리

단순 직류 전원을 인가해줄 경우 180도 회전 시 가해지는 힘이 반대 방향으로 된다.

그런 현상을 막기위해 필요한 것이 DC Motor Controller이다.

-DC Motor Controller

DC모터 제어를 도와주는 장치.

모터의 속도, 방향제어, 모터의 회전을 효율적으로 하도록 도와준다.

모터가 180도 돌때마다 회로의 방향을 바꿔줘 한방향으로 힘이 가해지도록 한다.

PWM, Enable, DIR 핀을 통해 모터를 컨트롤 한다.

Enable핀은 모터의 전원을 줄것인지 말 것인지 결정하는 핀.

Enable핀의 값을 0을주어아 enable 1을주면 disable이다.

DIR핀은 0이면 역방향 회전 1이면 정방향 회전이다.

PWM핀은 모터의 회전속도를 결정하는 핀 이다.

Duty ratio를 조절하여 모터의 회전속도를 컨트롤 (Duty ratio가 0이면 회전을 멈추고 1이면 최대속도로 회전)

**실습내용 및 결과**

실습1

|  |
| --- |
|  |
| #include  #include  int main(){ DDRF = 0xff ;  while(1)  { PORTF = 0b00000001;  \_delay\_ms(100); }  return 0 ; } |
| 오프라인 수업에서 조교님께 결과를 확인 받음. |

실습2

|  |
| --- |
|  |
| #define F\_CPU 16000000UL  #define \_\_DELAY\_BACKWARD\_COMPATIBLE\_\_  #include <avr/io.h>  #include <avr/interrupt.h>  #include <util/delay.h>    volatile int ratio = 0;  volatile unsigned char ONOFF = 1; // 전원  volatile unsigned char direction = 1; // 회전방향    ISR(INT0\_vect){ // 회전, 멈춤  if(ONOFF == 1){  PORTF = 0b00000000; // stop  ONOFF = 0;  }  else{  PORTF = 0b00000100; // enable  ONOFF = 1;  }  }  ISR(INT1\_vect){ // 회전 방향 변화  if(direction == 1){  direction = 0;  }  else{  direction = 1;  }  }  ISR(INT2\_vect){ // 회전 속도 증가  ratio += 10;  if (ratio >= 90) {ratio = 90;} // 회전 속도 상한  }  ISR(INT3\_vect){ // 회전 속도 감소  ratio -= 10;  if (ratio <= -90) {ratio = -90;} // 회전 속도 하한  }    int main(void)  {  DDRF = 0xff; // PORTF 출력 설정  DDRD = 0x00; // PORTD 입력 설정  PORTD = 0xff; // 풀업저항 설정  EICRA = 0b10101010; // 인터럽트 트리거 방식 설정(Falling Edge)  EIMSK = 0b00001111; // 인터럽트 허용 설정(INT0~3)  SREG |= 0x80; // 전체 인터럽트 허가  ONOFF = 1;  direction = 1;  PORTF = 0b00000001;  while(1){  if (ONOFF == 1)  {  if (direction == 1) // 정방향이면  {  PORTF = 0b00000001; \_delay\_us(100 + ratio);  PORTF = 0b00000000; \_delay\_us(100 - ratio);  }  if (direction == 0) // 역방향이면  {  PORTF = 0b00000011; \_delay\_us(100 + ratio);  PORTF = 0b00000000; \_delay\_us(100 - ratio);  }  }  }  } |
| 오프라인 수업에서 조교님께 결과를 확인 받음. |

**고찰**

atmega128의 PORT F 의 0비트,1비트,2비트에 모터 드라이버를 연결하였다.

|  |  |  |
| --- | --- | --- |
| 0비트 | 1비트 | 2비트 |
| PWM제어 | foward/reverse | enable/disable |

각 연결의 의미는 위의 표와 같음을 강의노트에서 확인하였으나 이해를 잘못해서 오직 1비트 가 high면 정방향으로 출력되는 것으로 간주하여 정방향/역방향이 제대로 동작하지 않았다.

정방향/역방향을 결정하기 위해 1번비트에 값을 0 또는 1 를 주고 모터 동작을 위해 pwm을 사용해야하므로 0번비트 또한 0또는 1를 주어야 한다.

|  |  |
| --- | --- |
| 이론 | 잘못이해한 이론 |
| 정방향 0/역방향 1 일때  PORTF = 0b00000001;  \_delay\_us(100);  PORTF = 0b00000000;  \_delay\_us(100);  PORTF = 0b00000011;  \_delay\_us(100);  PORTF = 0b00000000;  \_delay\_us(100); | 정방향 0/역방향 1 일때  PORTF = 0b00000001;  \_delay\_us(100);  PORTF = 0b00000000;  \_delay\_us(100);  PORTF = 0b00000010;  \_delay\_us(100);  PORTF = 0b00000000;  \_delay\_us(100); |

1비트에서는 pwm제어를 할수 있는 비트로서 하나의 주기 안에서 HIGH LOW의 비율에따라 출력이 달라졌다. 서보모터와 달리 특정 주기가 아니기 때문에 타이머 인터럽트를 사용하지 않아도 on off의 비율을 다르게 하여 속도의 증감을 보여줄 수 있을 것이라 생각했다.

그러므로 HIGH와 LOW를 유지하는 delay를 사용하고자 한다. 프로테우스 상에서는 ms단위로 제어해도 속도의 증감을 볼 수 있었으나 실제 atmega128에서는 멈춤과 동작의 형태가 속도가 아니라 정지 또는 회전으로 보였다. (회전하고 멈추고 회전하고 멈추고) 그래서 delay를 민감하게 줘야 매끄럽게 출력 될 수 있음을 예상하였고 실제로 단위를 ms 에서 us로 전환하여 속도의 증감을 확인 할 수 있었다.

2비트에서는 모터 드라이버안에 인버터가 내장되어있으므로 모터를 활성화 하기 위해 0을 입력해주어야 1로 입력되어 모터가 동작할 수 있게 된다.

코드를 합치는 과정에서 ratio를 unsigned로 선언해 속도를 줄일때 ratio값이 음수가 되지 못하고 0이되어버려 모터가 멈추었다.. 변수를 선언할때 그 변수가 가지는 값의 범위를 제대로 파악한 뒤 알맞은 자료형이 사용 되었는지 확인해야만 한다.