< 8515 : PWM 신호 만들기 실험 >

앞 글에서, 데이터 시트를 이용해 추리(?) 해 본 생각이 과연 맞는 것인지 어젯밤에 실험으로 돌려 보았습니다. 역시 오류가 있더군요... 흑흑...

그러나, 차근차근 하나씩 따져보는 '과학적인 방법'에 따라 큰 어려움없이 해결할 수가 있었습니다. 이 글에서는 그 과정을 적어 보겠습니다. 모든 일에 적용되는 보편적인 자연의 법칙은, 뉴턴의 3 법칙이나 아인슈타인의 상대성 원리 따위를 들 수 있다고들 하지만... 그보다는 '*머피의 법칙*'이 더욱 예외없이 적용되는 법칙이 아닌가 하는 생각이 듭니다...

1. Errors errors erros....

- 하드웨어의 문제

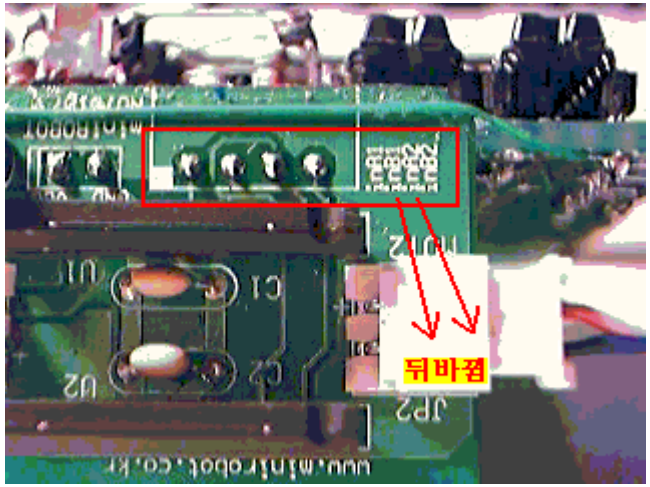
처음에는, 모터 드라이빙 부분과 메인 보드 부분의 전원을 따로 쓸려고 계획했었답니다. 그런데 결국 5V 단일 전원이 되고 말았죠... 이렇게 되고 만 가장 큰 이유는, '심플함'과 '사이즈' 때문이겠죠?

그런데도 불구하고, 포토 커플러를 사용하게 됩니다. 그것은 순전히 울적한 기분... 그땜에.

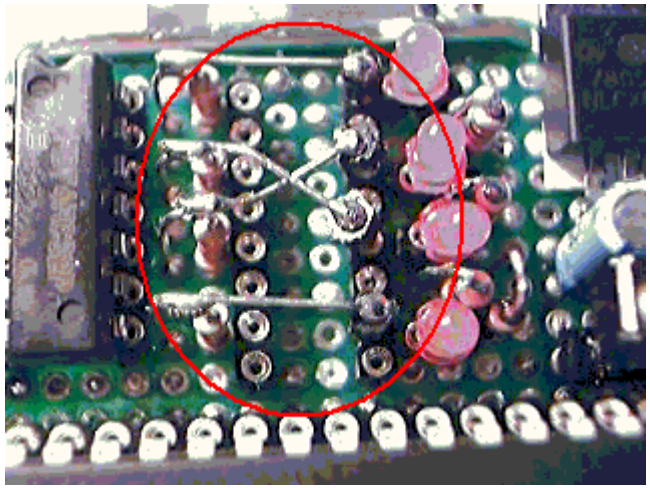
결국 만들고 나서 동작시켜 보니 영 펄스 나오는게 이쁘지가 않더군요. 오히려 포토 커플러를 사용함으로써 불안해 뵈는 것이었죠... 결국은 제거.

그리고 또 하나의 문제점은.... 모터 드라이버 회로로 채용한, '미니로봇'사의 'MR-RM02 소용량 DC모터 드라이브 모듈'에서 표기된 모터의 입력점이 잘못 프린트되어 있었다는 점입니다. 첨엔 철썩같이 믿고 별다른 확인없이 바로 와이어링을 했는데... 가운데 단자 두 개가 뒤바뀌어 표기되어 있는 바람에, 모터가 둘 다 벅벅 대더군요.

뭐 소중한 우리 나라의 중소기업 제품이니, 이해해 주고 넘어가기로 하고.... 단자 연결을 바꿔 줌으로써 해결.



가운데 2개 단자 표기가 뒤바뀜...



영 시원찮아서 Photo-coupler 제거...그리고 가운데 2개 단자 연결 바꿈...

엥... 사진에 나왔듯이, 가운데 2개의 잘못된 연결을 바꾼 솜씨 치고는 꽤 지저분하네요.... 어쨌든, 단락 및 단선만 주의하면 일단은 안심이니.... 그냥....

일단은 이것 말고는 동작에 심각한 영향을 주는 하드웨어적인 요소는 없어 보입니다. 그러나 이것을 바로잡고 나서는 바로 원하는 2채널의 PWM이 잘 나왔을까요? 그건 아니었답니당. 그래서 이번에는 소프트웨어적인 문제점임에 틀림없다는 생각이 들었죠.

- 소프트웨어의 문제

문제는, OC1B 핀에서의 PWM 출력은 잘 나오는데.... Port D 5번핀에 해당하는 OC1A 핀에서는 아무런 펄스가 나오지 않는다는 점이었습니당. 도대체 어케된 일일까? 미치고 환장하겠다... 하고 삽질하다가, 결국은 제가 AVR의 이해가 부족하다는 결론에 도달하게 되었군요. 그나저나 해답은 뭐였을까요?

넵. Port D 5번핀을 먼저 '출력' 상태로 셋팅해 주었어야 한다는 거였습니당. 어처구니없는 실수라고 할 수도 있겠지만... 적어도 데이터 시트 상에는 그런 언급이 되어있질 않았거덩요... 후훗.. 변명에 급급 ^_^;;... 너무 당연한 과정이었기에, 데이터 시트에서는 언급할 필요도 못느꼈나 보죠?

암튼 그 핀을 출력 상태로 설정해 주는 1줄을 집어넣고 나니, 8515 그놈이 뻔뻔하게도 PWM 신호 2 채널을 틱틱틱 뱉아내더군요.

그래서, 이제 마음대로 신호를 사용할 수 있게 되었습니다.

아직 문제가 남아있는데, 그것은 바로... 최초 전원 스위치를 딱! 올렸을 때 잠시 짧은 순간동안 모터가 급발진(?) 하는 것입니다. 우리의 Pocket-bot 에서는 큰 문제가 아닐 수도 있지만... 만약 무게가 많이 나가는 큰 시스템이라면, 인명피해의 우려조차 낳을 수 있는 심각한 것일 수도 있겠군요. 이건... 회로상에서 조치를 취해 주어야 할 문제 같은데... 아직은 일단 그대로 놔 두기로 합니다.

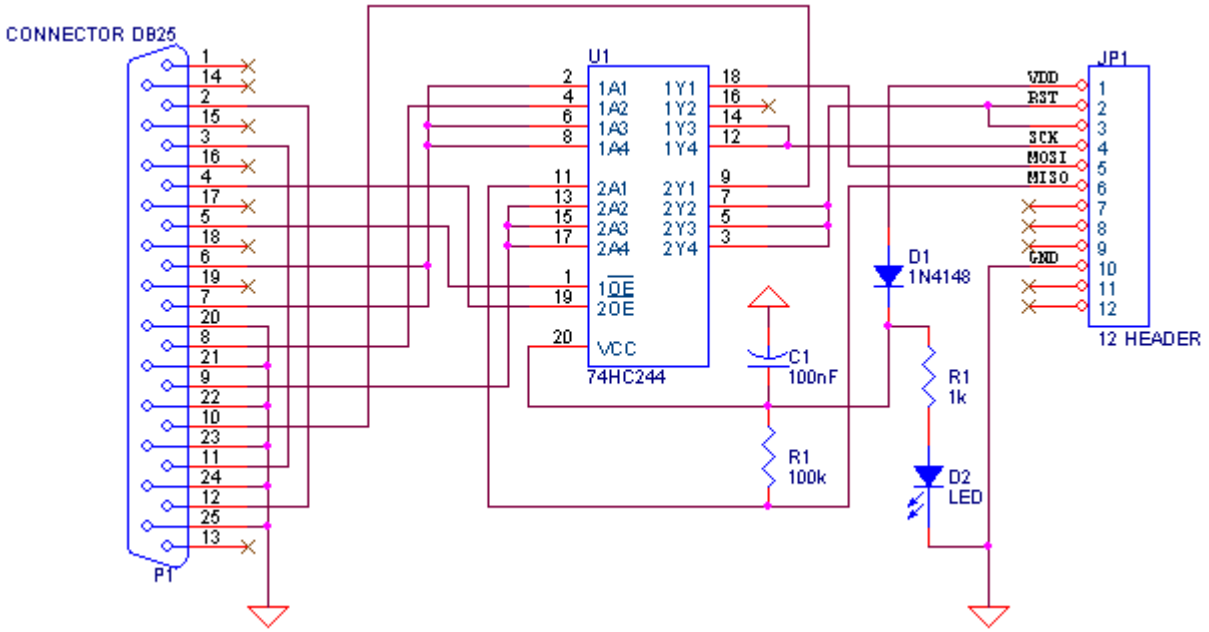
2. PWM Driving System

- Circuits

실험에 사용된 회로도를 잠시 올려 봅니다. 별로 특징적인 노하우 같은건 저얼~때루 없습니당. 원래 저는 노하우란게 없으니까요!

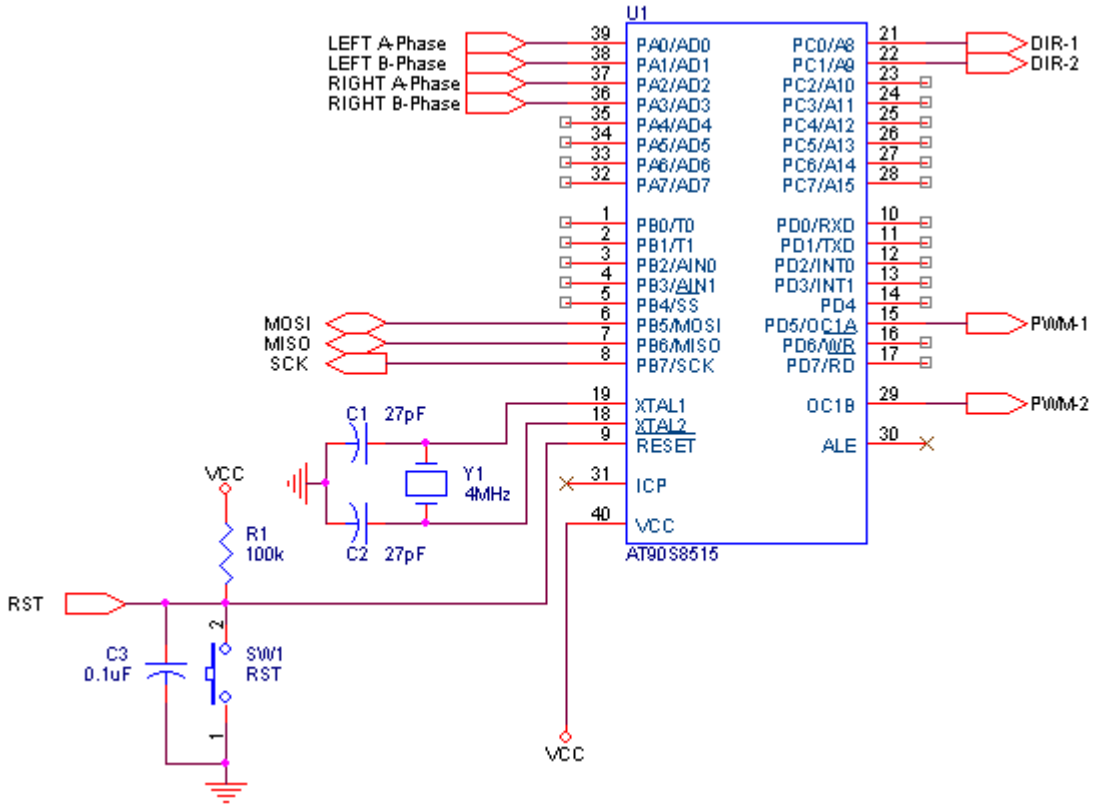
바로 아래 도면은... 병렬포트를 이용한 ISP를 위해 따로 만든 회로이죠. 너무 잘 알려진 것이고, 저는 단지 전원이 들어오는지 않오는지 볼려고 LED 만 추가... ISP를 위한 Host PC 측의 프로그램은, ATMEL ISP 프로그램을 이용했습니다. 다운로드 할 때 백이면 백 성공하는 것은 아니구, 보통 2~3회 시도해야 1번 성공하는 정도... 나머지는 특정한 문자가 나올 때 에러가 뜰 경우가 많더군요. 왜 그런지는

모릅니당...헤헤... 그리고, 커넥터가 좀 부실해서 중간에 약간이라도 움직이면 잡음이 들어가서 에러가 뜨기도 하고... 그래서 다운로드 할 때는 얌전히...



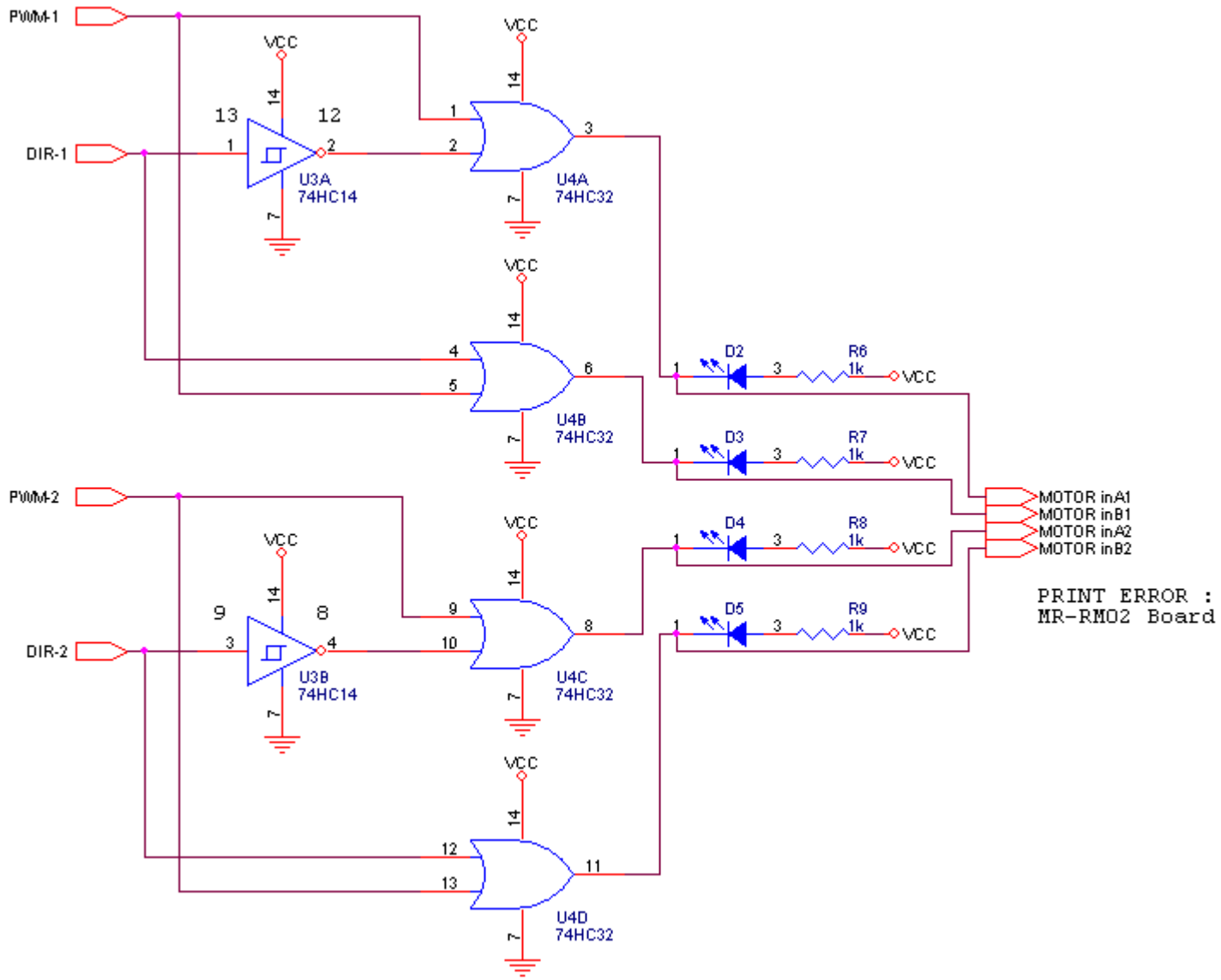
STK-200 의 병렬 AVR ISP...

그리고, 요 아래 도면은 8515 에 연결된 단자들에 관한 것입니다. 4MHz OSC 넣어주고. 예의상 리셋 버튼 넣어주고. 끝. 음... 모터 구동을 위해 2개의 PWM 출력선을 따내고, Port C 의 0,1번 핀을 방향 지시용으로 따냈습니다. 또 ISP 커넥터 연결을 위해 MOSI, MISO, SCK, Reset 및 전원선을 따내주고요. 그리고... Port A 에 입력되는 4개의 단자는, 나중에 2개의 엔코더를 위해 준비된 것입니다. 요 문서에서는 사용하지 않습니당.



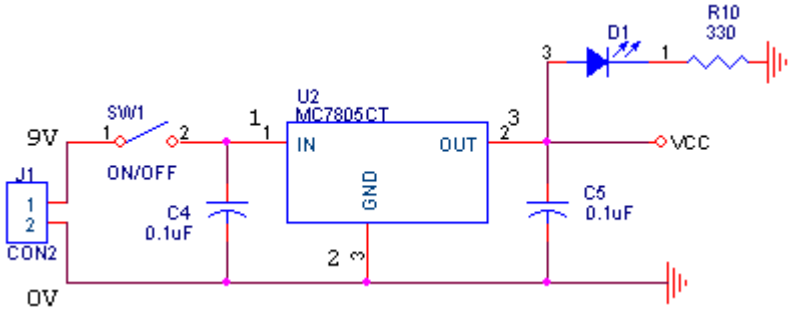
8515 결선 그림

네. 아래의 회로는, PWM-1 / PWM-2 및 DIR-1 / DIR-2 신호선을, 적절한 연산을 통해 H-Bridge 회로에 연결될 수 있도록 변환합니다. 아울러 상태를 볼 수 있도록 예쁜 LED 도 달아주고요. 이때 달아준 LED와 저항은, Vcc와 연결되어 있어 풀업저항과 같은 효과도 겸하고 있죠... 신기하게도 풀다운 되었을 경우, LED의 다이오드 성질 때문인지 몰라도... 암튼 GND에 연결되는 풀다운일 때는 두 단자씩 모두 불이 들어와 버리는 신기한 일(논리적으로는 그러면 안되는뎅)이 일어납니다. 그래서 반드시 풀업으로 해 줍니다. 이 때문에, 인버팅 효과가 있으므로... PWM 출력을 아예 첨부터 인버팅 시켜 내보내 주어야 합니다. 먼소리냐면, 인버팅 시켜주지 않고 신호를 보내면... 0x00 즉 '꺼라'고 명령 내리면 0xFF 즉 '최고출력' 상태로 되어 버린다는 뜻이죠. 다행히 AVR의 PWM 설정 레지스터를 조작하면, 신호를 인버팅 시켜줄 수가 있네요! 뭐 꼭 그런 기능 없더라도, 프로그램에서 직접 숫자를 뒤집어 내보내면 되긴 하지만...



PWM 및 DIR 신호와 모터 드라이버를 인터페이스...

요건, 흔해빠진 전원 부분인데... 아직은 7805 가 난로 역할을 할 정도는 아니네요. 지금 현재 문제점은, 이놈이 전체 시스템이 사용할 모든 전원을 공급해 주어야 한다는 사실인데, 심지어 모터를 직접 구동하는 전원까지 여기서 뽑아냅니다. 엽기적이죠? MC7805CT 가 1A 까지는 인가해 준다니까 그 말만 믿고... 헐헐... 일단은 모터가 큰 부하만 받지 않으면 그럭저럭 사용은 가능할 듯 싶습니다. 희한하게도, ORCAD 9.1 의 라이브러리에는 핀번호가 이상하게 되어 있네요? 약간 큰 글씨로 된 것이 실제 결선된 핀번호입니다.



흔해빠진 전압조정기...

네.... 이게 PWM 으로 DC 모터를 구동하는 하드웨어의 전부 다입니다. 간단해서 너무 좋네요. 만능기판에 배선하기는 너무 싫습니다... 예전에, 100핀짜리 AM188ES SMT 패키지를 만능기판에 배선하다가 비싼 칩 여러개 버린적 있거덩요... 그거 진짜 납땜 연습은 되지만, 진짜 해야될 일을 못하게 방해하는 요소가 되기도 하더군요. 시간낭비, 돈낭비죠. 그래서... 만능기판에는 DIP 형 패키지를 사용하자... 분수에 맞게... 라고 생각하고 있습니당. 다행히 우리의 AVR 은 핀 수도 적은 편이라 너무 고마울 따름입니다. 조금이라도 노가다(?)에서 우릴 해방시켜 주었으니까요.

- C code

컴파일러는 물론 GCC 이구.... 일일이 Makefile 공부해서 만들어 컴파일하고.. 하는 과정 역시 좋은 툴 있는데 그거 그냥 편하게 쓰자... 해서, 황해권님의 AVR-EDIT 3.2 프리웨어를 얻어씁니다.

이 문서 바로 앞의 'PWM 연구' 문서에 소개된 코드는 검증안되고 안돌아가는 쓰레기입니다. 요 아래 제시된 코드는 실제로 돌려본 것입니다. 좀 장황한 코드군요. 효율적이지도 않고.... 하지만 이해하기는 쉽게 하려 했습니다.

정해진 방향과 속도로... 신호를 그냥 계속 출력하는 것이죠?

```
1   /* 모터 2개를 일정한 속도로 돌려보기 테스트용 프로그램 */
2   /* 김동호 - 2001.12 */
3   #include <io.h>
4   #include <interrupt.h>
5   #include <sig-avr.h>
6
7   /* PWM */
8   #define PWM_INIT_SET_A      0xF1    /* PWM 2개 사용, 인버팅, 8비트 분해능 */
9   #define PWM_INIT_SET_B      0x05    /* 프리스케일링 = CK/1024 ; 최고로 느린 상태... LED로 확인해 보기 위해
    */
10  #define PWM_INIT_VAL_AH     0x00    /* PWM 초기값들.... */
11  #define PWM_INIT_VAL_AL     0x00
12  #define PWM_INIT_VAL_BH     0x00
13  #define PWM_INIT_VAL_BL     0x00
14
15  /* Error Code */
16  #define GOOD            0       /* 전형적인 ANSI C 적인 표현을 위해서.. */
17
```

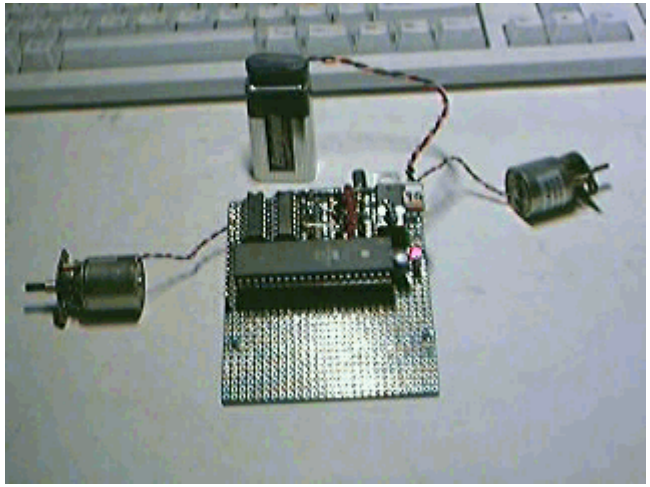
```
18  /* PWM 변수들 - 정적 변수로 하는데...  쩝. 흡사 베이식 수준이군요.. */
19  unsigned char pwm_m1 = 0x00;     /* 1번 PWM 값 */
20  unsigned char pwm_m2 = 0x00;     /* 2번 PWM 값 */
21  unsigned char pwm_dir = 0x00;    /* 방향 지시용 */
22
23  /* 쌍발 PWM 초기화하기~ */
24  int pwm_init(void)
25  {
26      outp(PWM_INIT_SET_A,TCCR1A);    /* 분해능 8비트짜리 PWM 2개 준비하기 */
27      outp(PWM_INIT_VAL_AH,OCR1AH);   outp(PWM_INIT_VAL_AL,OCR1AL);    /* A 핀을 위한 초기값 적어주기
    */
28
29      outp(PWM_INIT_VAL_BH,OCR1BH);   outp(PWM_INIT_VAL_BL,OCR1BL);    /* B 핀을 위한 초기값 적어주기
    */
30      outp(PWM_INIT_SET_B,TCCR1B);    /* 프리스케일링 */
31      return GOOD;
32  }
33
34  /* 쌍발 PWM 출력하기~ */
35  int pwm_out(void)
```

```
36  {
37      outp(pwm_m1,OCR1AL);    outp(pwm_m2,OCR1BL);     /* PWM 출력하기~ */
38      outp(pwm_dir,PORTC);     /* 방향 출력하기 */
39      return GOOD;
40  }
41
42  /* 메인 함수 */
43  int main(void)
44  {
45      unsigned char i=0;     /* 공회전용 */
46      outp(0xff,DDRC);         /* Port C 전체를 출력으로 설정 - 방향 지시 비트 2개의 출력을 위해서 */
47      outp(0xff,DDRD);         /* Port D 전체를 출력으로 설정 - 2번 PWM 핀 1개의 출력을 위해서*/
48      pwm_init();     /* PWM 모드로 초기화 */
49      sei();     /* 인터럽트 인가하기 */
50      for (;;)     /* 무한루프 */
51      {
52          pwm_m1 = 0x0f;
53          pwm_m2 = 0x0f;
```

```
54          pwm_dir = 0x02;
55          pwm_out();     /* 2개의 PWM 및 방향 신호 출력하기 */
56          i=0;
57          while (i<0x0f)  {  i++;  }     /* 괜히 안전빵으로 공회전 쪼금 시켜주기 */
58        }
59        return GOOD;
        }
```

3. Test test test....

위의 프로그램은 아래 사진과 같은 환경을 통해 구동 실험되었습니다. 간단하고 귀엽게 보이기도 하지만... 제 눈에는 마치 실험을 위해 중추신경계만 뽑혀진 개구리와 같은 인상이로군요.. 끔찍!



간단한 실험장치 구성..

- LED 불빛 하염없이 바라보기

위에 소개된 프로그램을 다운로드 받고서 일단 돌려보면, 두 개의 모터가 마치 스텝모터와 같은 분위기(?)로 돌아갑니다. PWM 신호의 베이스 주파수를 1024분주했기 땜에(프리스케일링) 엄청 느려진 것이죠. 그래서 틱틱틱... 하면서 돌아갑니다. 모터가 그런식으로 돌기 때문에, 모터 내부의 회전자 질량과 기어박스에 의한 토크 때문에 반작용이 생겨 모터가 지멋대로 책상위에서 돌아다니는군요! 뭐 나름대로 예쁘긴 하지만...

암튼, LED 불빛도 느린 주파수 땜에 껌벅이는게 눈으로 확인되구요... PWM 값을 변경시킴에 따라 그 밝기와 껌벅임도 바뀌고.... 암튼 잘 돌아가주니 8515가 고맙네요.

잠시 넋놓고 있는데, 갑자기 또 2번 PWM 신호가 제멋대로 나오기 시작합니다.... 제길... 하고 또 삽질할 준비하고 있는데... 혹시나 싶어 건전지를 교환하니 정상... 약이 다 떨어졌었군요...흑흑..

- 프리스케일링 설정값 바꿔가며 관찰해 보기

프리스케일링 설정을 변경하려면, TCCR1B 레지스터의 0,1번 비트를 변경해 줍니다. 데이터 시트를 보면 예쁜 표로 잘 그려놓았군요.

암튼 좀 더 빠른 제대로 된 PWM을 출력하기 위해서는.. 프리스케일링 값을 더 줄여잡아서 더욱 빠른 베이스 주파수를 얻어야겠죠. 지금의 1024 분주 말고, 254분주 / 64분주 / 8분주 모두 잘 돌아가 줍니다.

오실로스코프가 없이 LED 만으로 깜박임이 대강 확인되는 것은 254분주 정도까지이고, 그 이상은 잘 구분이 가질 않네요... 하지만, 이상없이 잘 나오고 있다고 믿어줘야지 뭐 별 수 있겠습니깡...

모터를 붙여서 돌아가는 반응을 보면... 역시 254분주까지는 좀 껄끄럽게 움직여 주고 있으며, 64분주때는 좀 낫고, 8분주때는 상당히 매끄럽게 움직여 줍니다. 역시 실제 사용할 때는 8분주나 64분주로 맞추어 사용해야겠군요.

4. 앞으로 이걸 어떻게 써먹을 것인가.

- PWM 출력의 의미는...

직접 아날로그 전력을 보내어 모터를 구동하지 않고, 펄스폭 변조(PWM ; Pulse Width Modulation) 방법을 사용하는 이유는... 여러 책에 잘 나와 있지만, 저도 한번 언급해 봅니다.

- 모터축 및 연결된 부하에는 건조마찰(Coulomb Friction)이 존재하며, 이 마찰력의 특징은... 정지마찰 계수가 동적마찰계수보다 더욱 크다는 데 있다. 여기에 더하여, DC 모터의 전기기계적 특성상 최초 기동시에는 여유있는 전력의 공급이 필요하다. 즉 모터는 완전히 '선형적인' 물건이 아니라는 점이다. 그러므로, 1V를 인가해 주었을때 모터는 꿈쩍도 하지 않을 수가 있다. 이론상으로는 살살 돌기 시작해야 하는데 말이다. 그런데 이때 PWM의 스위칭 작용을 통해 정격전압을 통째로 공급해 주면, 짧은 시간이나마 충분한 전력이 공급되는 효과가 있어 기동이 실패할 확률이 줄어든다. 즉 **모터가 말을 더 잘 듣게 된다.**
- 구동 드라이버의 **동작이 확실해지고 에너지 효율도 높아진다.** 즉 선형증폭기를 사용한 아날로그 드라이버의 경우, 효율이 오디오용 수준의 A클래스 증폭기의 경우 50%대 이하로 떨어지는 경우도 있고... 그보다 주파수 특성이 않좋은 B나 C 클래스의 경우라도 80%를 넘기기가 거의 힘들다. 그러나, 단순 스위칭 동작(Discrete)을 하는 D 클래스의 증폭기를 사용한다면 효율 90% 수준은 쉽게 넘길 수 있게 되며 이는 곧 더 높은 성능을 나타내는 지표의 하나가 될 수 있다.
- PWM은 일종의 디지털 신호라 할 수 있으며, 모터는 아날로그 디바이스에 속한다고 말할 수 있다. 따라서 DA 컨버팅이 필요하지만... 모터 자체가 하나의 저역통과필터(LPF ; Low Pass Filter) 역할을 하므로 **실제적인 사용에 지장이 없다.**

- 피드백 루프 제어(Feedback Loop Control)로 써먹어야죠...

Pocket-bot 은 보다 나은 콘트롤을 위해 피드백 루프 제어를 할 것입니다. 피드백이란, 한 마디로 센서에 의한 입력값들과 이를 통해 결정된 출력값들을 매개체로 삼아... 콘트롤러와 시스템의 상태간에 상호작용을 시켜준다는 것이죠. 이를 위해 사용되는 콘트롤러의 종류는 무궁무진하게 많죠! 하지만, 이들 콘트롤러는 소프트웨어를 짜줌으로써 쉽게 변경가능한 것이니... 엔코더와 PWM 모터를 갖춘 하드웨어만 갖춘다면, 이를 쉽게 구현할 수 있습니다.

우선 가장 간단한 제어기(Controller)로서 소위 '틱톡 제어기(Tic-toc Controller)'가 있겠죠. 수위를 오버하면 끄고 부족하면 켜는... 이것만으로도 충분할지도 모릅니다.

그리고, 많이 사용되는 PID 제어기가 있군요. PID 제어기의 파라미터 결정을 해 주는 표준적인 설계과정은... 주파수 영역에서 다루기 때문에 보통 사람들은 좀 이해하기가 어렵죠. 게다가 잘 모델링된 시스템에만 적용 가능하구요. 하지만, 실제로는 PID 제어기의 파라미터 결정은 많은 부분 경험적인 시행착오법(Trial & Error)으로 하죠.. 마치 피아노 조율하듯이요. 하지만 MPU를 사용한 '디지털 PID 제어기'는 훨씬 쉽게 구성할 수 있습니다.

음... 그리구, 학부수준의 제어기로서 유명한 '선형 제곱 조정기(LQR ; Linear Quadratic Regulator) 가 있네요. 상태공간으로 모델링된 시스템을 제어하는데 재미있고 유용한 제어기죠. 역시 수학 따위 안쓰고 시행착오법으로 시도해 볼 만 합니다.

퍼지(Fuzzy) 제어기는 어떨까요? 메모리가 좀 딸리나? 뭐 좀 러프하게 구성하면 상당히 작게 만들어 볼 수도 있을지 모르죠. 아님 요새 유행하는 유전자 알고리즘(Genetic Algorithm)이나 신경망(Neural Network)를 이용한 것은요? 이런 비선형 제어기는 솔직히 재미도 있고 따라해 보기는 오히려 쉽게 느껴지기도 합니다.

암튼 뭐든지 알고리즘만 잘 짜면 모두 구현해 볼 수 있겠다고 생각됩니다. 물론 고성능 DSP MPU를 사용하는 것이 아니므로... 그 유용성은 잘 모르겠습니다만.

나중에 하드웨어가 갖추어지면, 소프트웨어 부분을 많이 다루어 보고 고민해 볼 순서가 되겠죠? 그때가서 고민해 보기로 하구요... 암튼 이제 PWM은 마음대로 나오니깐... 로봇은 어떻게든 굴러가겠죠.. 긍정적으로 생각합시당!