|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **결 과 보 고 서** | | | | |
| 학 과 | 학 년 | 학 번 | 조 | 성 명 |
| 전자공학과 | 3 | 12191505 | - | 윤수연 |
| 실험 제목 | PWM, Servo Motor | | | |
| 1. **실험 과정**    1. 실습 1단계  |  | | --- | | #include <inttypes.h>  #include <avr/io.h>  #include <avr/interrupt.h>  #include <avr/sleep.h>  int main()  {  DDRF = 0xff;  while (1){  PORTF = 0b00000001;  \_delay\_ms(100);  }  return 0;  } |       그림 1. 실습 1단계 회로 구성도  실습 1단계는 단순이 모터의 동작을 확인하는 단계였다. DC motor에 pwm 신호를 주게 되면, 방향이나 속도 조절이 가능한데 이 단계의 경우에는 아무런 조작이 없는 단계다. 평균 전압 5V, duty cycle 100%인 pwn 입력 신호에 motor가 동작하는 것을 확인할 수 있었다.   * 1. 실습 2단계  |  | | --- | | #define F\_CPU 16000000  #include <avr/io.h>  #include <util/delay.h>  #include <avr/interrupt.h>  unsigned char motor\_on = 0; //1:on  unsigned char dir = 0; //0: forward  ISR(INT4\_vect){ //start stop  if(motor\_on ==1){ //on -> off  PORTF = (dir ==1) ? 0b00000000 : 0b00000011; //motor off, backwoard : mortor off, forward  motor\_on = 0; //off  }  else if (motor\_on == 0){ //off -> on  PORTF = (dir ==1) ? 0b00000001 : 0b00000010; //motor on, backward : motor on, forward  motor\_on =1; //on  }  }  ISR(INT3\_vect){ //backward forward  if(dir ==0){ //forward -> backward  PORTF = (motor\_on == 1) ? 0b00000001 : 0b00000000; //motor on, backward : motor off, backward  dir=1;  }  else{ //backward -> forward  PORTF = (motor\_on ==1) ? 0b00000010 : 0b00000011; //motor on, forward : motor on, backward  dir = 0;  }  }  ISR(INT6\_vect){ //slowly  if (motor\_on ==1){  if (OCR2 >10){  OCR2 -= 10;  }  }  }  ISR(INT6\_vect){ //fast  if (motor\_on ==1){  if (OCR2 > 245){ //max: 250  OCR2 += 10;  }  }  }  int main(){  //init  DDRF = 0x0f; //output  DDRB = 0b10000000;    DDRE = 0b10000000;  PORTE = 0b11110000;  TCCR2 = 0b01101101;  TIMSK = 0b10000000;  TCNT2 = 0;  OCR2 = 100;    EICRB = 0b10101010;  EIMSK = 0b11110000;    SREG = 0x80;  PORTF = 0b00000011;      while (1){}  return 0;  } |   실습 2단계는 실험 1단계의 연장선으로, motor를 PWM신호를 이용하여 간단히 제어하는 단계였다. 버튼을 통해 동작을 enable시키기도 하고 방향을 바꾸기도 하며 조작하는 단계다.  이 단계는 유독 proteus하드웨어 단계가 어려웠는데, 강의 노트의 이미지가 ATmega130에 가깝게 생겨서 혼란스러웠기 때문이다. 하지만 몇 번의 실수 끝에 ATmega128로 실험을 완성했다. 또한 원래 동작은 스위치를 이용하는 것이었는데 계속해서 동작하지 않자 이제까지 많이 사용했던 button소자를 통해 원하는 동작을 구현할 수 있었다.  대부분의 동작은 interrupt를 통해 제어 신호들의 변화를 비트 연산하여 동작하도록 했다. OCR값 제어를 통해 속도를 제어하는 것은 20부터 250까지의 제한을 두고 동작하도록 했다.  enable에 해당하는 INPUT1과 dir에 해당하는 INPUT2 신호를 변경하는 경우에 나머지 한 신호를 고려하여 동작 중에 방향을 바꾸는 경우에는 dir에 해당하는 input2를 중점으로 하여 input1,2이 서로 not 관계에 있도록 신호를 바꾸고, 방향을 바꿀 때에도 motor가 꺼진 경우라면 input1,2이 같도록 하는 등 모터 동작 여부와 방향을 바꾸는 인터럽트 구문을 수정하여 해결할 수 있었다.    그림 2. 실험 2 회로 구성도   1. **고찰**   이번 주차에는 PWM를 통해 DC Motor를 구현해 보았다. 이렇게까지 Timer/Counter가 중요한 역할을 하는지 몰랐는데 여러 주에 걸쳐 실험을 진행해보니 Timer/Counter의 역할의 중요성을 깨닫게 되었다. 정확히 말하면 Timer/Counter가 중요하기 보다는 내부 펄스신호가 중요하다는 것을 느꼈다.  2단계의 경우 모터를 제어하는 동작을 수행했는데 switch 소자는 동작하지 않고 button동작에서만 동작하는 것이 의아하기도 했는데 코드를 리뷰해보니 switch가 제대로 동작하지 못하는 이유를 알 수 있었다. switch소자의 경우 on off상태가 유지된다는 점에서 button과 차이점이 있다. 예를 들어 on에서 off로 스위치를 조작했다면 그 스위치는 조작 이후에도 계속해서 off상태를 유지하는 것이 된다. 그에 비해 button의 경우 trigger를 무엇에 두느냐 에 따라 다르지만 스위치를 누르는 순간 어떤 동작이 인식되고, 이후에는 다시 원래로 돌아온다. 따라서 속도 조절 등의 연속적인 동작에는 button소자를 사용하는 것이 올바른 동작일 것이다. | | | | |
|  | | | | |