|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **예 비 보 고 서** | | | | |
| 학 과 | 학 년 | 학 번 | 조 | 성 명 |
| 전자공학과 | 3 | 12191505 | - | 윤수연 |
| 실험 제목 | **PWM , Servo Motor** | | | |
| 1. **자료조사**    1. **Timer/Counter의 내부구조**   타이머/카운터는 ATmega128에 여러 종류가 있지만 구조부터 8비트와 16비트로 완전히 다르고 관련 레지스터 또한 전혀 다르다. 그 중에서 아래 그림은 가장 간단한 Timer/Counter인 Timer/Counter2의 내부 구조다. 본격적인 설명에 앞서 맨 위쪽의 TCCR2 레지스터는 주로 초기 세팅을 하는 데에 쓰이는 레지스터로, count하는 Timer/Counter의 기능과 직접적인 관련은 없다. 이 레지스터를 통해 어떤 모드를 사용할지 등을 정한다.    먼저 input부분을 살펴보자. Clock select라는 부분을 보면 두 가지 입력 중 하나를 선택해서 입력으로 받는 모습을 볼 수 있다. Timer/Counter는 clock을 count하는 기능을 말한다. 이름은 나뉘어진 것 같지만 사실 같은 기능을 말한다. 만약 clock이 외부에서 들어오는 외부 clock이라면 clock에 할당되는 시간이 비교적 불규칙적일 수밖에 없고, 그런 경우 clock을 count한다. 이런 기능을 카운터 모드라고 한다. 반면 clock source가 내부의 clock신호라면 당연히 한 클럭 당 일정 시간이 소요되는 일정한 주기의 신호다. 이를 이용해 몇 번의 주기가 실제로 얼마가 걸리는지 계산할 수 있고 이러한 경우를 타이머모드라고 한다. 이때 정확한 시간을 측정하기 위해서 내부 clock의 주파수나 duty ratio정보를 이용한다.  그렇게 선택된 신호는 코어부분, control logic으로 들어간다. 여기서는 실제 count가 이루어지는 등 기능을 하는 곳인데 플립플랍으로 연산이 이루어진다.  이후 계산된 값은 TCNT2레지스터로 들어간다. 이 레지스터는 Timer/Counter의 실제 값이 들어가는 레지스터라고 볼 수 있는데, 8비트로 count된 값이 들어가게 된다. 이렇게 들어간 값은 유용하게 쓰이기 위해서 OCR2 레지스터와 비트 비교를 해서 일치하면 OCF2로 출력하거나 waveform generation을 거쳐서 새로운 펄스파를 출력한다. 또는 OCR2 레지스터를 아예 사용하지 않고 그냥 overflow가 일어남에 따라 동작이 이루어지기도 한다.  이 때 waveform generation을 사용하는 경우를 잠깐 살펴보자면 새로운 펄스 파를 만들기 위해서 TOP, BOTTOM값과 FOC2의 값이 들어간다. 이는 새로운 펄스를 생성할 때 필요한 정보다. 이후 OC2로 출력이 나온다고 표시되어 있는데 이 때 OC2는 AVR ATmega128에서 PORT D의 7번 핀을 말한다.  24. 카운터 (Counter)   * 1. **추가 레지스터 설명**   지난 시간 Timer/Counter의 동작을 처음 배웠을 때 레지스터의 동작을 알아보았지만 추가적인 레지스터 기능을 설명한다.   1. OCR2   테이블이(가) 표시된 사진  자동 생성된 설명  : 우리가 비교하고자 하는 값을 담은 레지스터. 다른 레지스터와 달리 이중 버퍼 구조로 동작하기 때문에 안정적으로 동작한다. 하지만 일반 CTC모드에서는 이중 버퍼 구조로 동작하지 않는다.   1. SFIOR   : 타이머/카운터들을 동기화 시키는데 관련된 기능을 하는 레지스터. Timer/Counter2와 관련된 비트는 8비트 중 7번 1번, 0번 비트다. 이 레지스터의 2번 비트는 PUD BIT다.   * 비트7 : TSM 비트로 여러 개의 타이머/카운터를 동기화 시켜 임의의 값으로 설정하고 동시에 카운팅하고자 할 때 사용한다. * 비트0: PSR321로 비트가 1로 되면 타이머/카운터 321이 하드웨어에 의해 즉시 RESET 된다.      * 1. **Timer/Counter2의 네 가지 동작모드**      1. 일반모드   : 지난 번 실험 주제로 알아봤던 가장 단순한동작이다. 그냥 up counter를 통해 외부 클럭을 count하는 목적이라고 생각하면 되는데, 아주 단순한 동작이기 때문에 여러가지 기능이 제한되는 모습을 보인다. 가장 대표적으로는 OC2을 이용한 파형 출력 기능이 제한된다. count하는 기능만을 위해 주로 사용되기 때문에 파형 출력과 같은 고급 기능은 다른 모드에서 사용할 수 있다.  또한 출력 비교 인터럽트 사용을 권장하지 않는다. 출력을 비교하는 기능을 할 수는 있지만 cpu의 처리에 비효율 적이다. 대신에 인터럽트 서비스 루틴에 TCNT2의 레지스터에 접근하여 값을 써주면 원하는 값으로 count가 시작되고, 비슷한 동작을 구현할 수는 있을 것이다.  따라서 위 기능을 위해 주로 사용되는 인터럽트는 오버플로우 인터럽트가 되며, 출력 비교 인터럽트는 권장되지 않는다.   * + 1. CTC모드   : Clear Timer on Compare match의 줄임말이다. 이 모드는 이름에서부터 알 수 있듯이 비교 인터럽트를 주로 사용한다. 일반모드와 비슷하게 값을 count하다가 대개 오버플로우 인터럽트가 아닌 출력 비교 레지스터 OCR2의 값과 비교하여 이와 같아졌을 때 출력 비교 인터럽트가 발생하게 된다. 이 때 자동으로 count되던 값은 0으로 reset되므로 이 경우 오버플로우 인터럽트는 발생하지 않는다.  이 동작을 보다 직관적으로 해석하면 주파수를 낮춘다. 즉, 분주비를 설정한다고 이해할 수 있다. 일정 값이 다다르면 오버플로우가 발생하고는 이를 파형 출력하여 0과 1을 번갈아 출력하는 펄스 파형을 만들면, 이는 입력으로 주파수가 높은 펄스를 받아 보다 주파수가 낮은 새로운 펄스 파형을 만든 것으로 볼 수 있기 때문이다. 따라서 이 모드를 주파수 분주의 용도로 사용한다고 표현한다.     * + 1. 고속 PWM모드   : 이름과 같이 높은 주파수의 출력 파형을 만들어 낸다. 주로 하나의 인터럽트를 사용하는 앞의 모드들과 다르게 비교 일치 인터럽트와 오버플로우 인터럽트를 모두 사용할 수 있기 때문에, 한 주기 내에서 모두 한 주기의 출력 펄스 파형을 만들어 낼 수 있다. 그렇게 되면 보다 높은 주파수를 이용할 수 있다. 각 인터럽트에 대해 0,1 값이 바뀌기 때문에 다른 모드보다 자주 값이 변동된다고 할 수 있다.  그 배경에는 앞에서 소개한 모드들과 가장 다른 점인 TCNT2 레지스터의 값이 인터럽트와 상관없이 단방향 경사 동작을 한다. 이는 고속 Pwm모드에서 절대 바뀌지 않는다.  출력 사이클에 있어서 일정한 주파수를 유지하며 0과 1의 duty cycle을 조절할 수 있다. 그 이유는 출력 비교 인터럽트와 오버플로우 인터럽트, 두 인터럽트를 모두 사용하는 점과 인터럽트와 상관없이 무조건 단방향 경사 동작을 하는 고속 PWM모드의 특성 때문인데, 그로 인해 주파수는 같은 시간이지만 OCR값을 어떻게 두느냐에 따라 DUTY비가 달라지기 때문이다.     * + 1. PC PWM모드   : PHASE CORRECT PWM 모드를 말하는 이 모드는 PWM파형을 출력할 때 고속 PWM모드에 비해서 주파수 면에서는 손해가 있지만 분해능을 보다 미세하고 정밀하게 CONTROL하기 위한 모드다.  이 모드에서는 다른 모드들과는 확연히 다르게 양방향 동작을 한다. 이 말은 그저 단 방향으로 TCNT2의 값이 증가하거나 감소하는 것이 아니라 상향모드와 하향모드를 반복하며 값을 늘리고 줄인다는 뜻이다.  따라서 오버플로우 인터럽트 대신 비교일치 인터럽트를 통해 동작한다. 하나의 인터럽트만을 사용한다고 해서 주파수가 일반모드나 CTC모드와 비슷하지는 않다. 왜냐하면 양방향 동작을 통해 같은 비교일치 인터럽트라고 하더라도 하향모드 중 만난 인터럽트인지, 상향모드 중 발생한 인터럽트인지에 따라 각각 펄스 변화가 생기고 따라서 이를 통해 출력 파형을 조절할 수 있다.  따라서 1/256의 배수로 DUTY RATIO를 갖는 고속 PWM모드보다 정밀한 1/512의 배수로 DUTY RATIO를 가질 수 있어서 정확한 DUTY RATIO가 필요할 때 사용하게 된다. 단 주파수 면에서 고속 PWM모드보다 고주파를 만들 수는 없으며, 각 주기에서 파형이 중간 지점을 기준으로 항상 대칭을 이루게 된다.     1. **실험**    1. **새로운 동작 기기**       1. **Servo Mortor**   서보 기구는 일반적으로 피드백 제어에 의해 그 기구의 운동 부분이 물체의 위치·방위·자세 등의 목표값의 임의의 변화에 추종하도록 제어하는 기구로, 기계를 명령대로 작동시키는 장치이다. 원하는 모터 기능을 수행하며 특히 PWM으로 제어한다.  AC Servo Motor 200W ECMA-C20602RS | DamenCNC B.V.   * 1. **실험 예상**      1. **실습 1단계**   CTC 모드로 OCR값을 제어하여 1초 주기의 파형 출력 후 LED로 관찰  1초 주기를 만들기 위해서 프로그램상의 loop 활용 필요  새로운 프로그램(가상포트 생성기)를 사용해야해서 그 부분이 시간이 오래 걸릴 것 같다. 또, 소자도 확연히 여럿 사용하고 회로 복잡도도 높아졌다. 사실 터미널에서 받아오는 숫자를 인식할 수 있는 것을 실제로 해보면 통신의 의미를 느낄 수 있을 것 같은데 프로그램 상으로 그럴 수 없다는 점이 아쉽다.   * + 1. **실습 2단계**   Fast PWM 모드로 사용자에게 USART 통신을 통해 받은 서보 모터의 각도 값을 입력 받아 duty ratio 조절을 통해 모터를 회전  새로운 동작 기기인 서보 모터를 처음 사용하는 단계다. 출력한 파형을 통해 실제로 모터를 회전 동작 시킬 수 있다. 한 번에 구동하기는 어려움이 있을 것 같아서 입력받은 각도 값을 기반으로 어떤 펄스가 형성될 수 있는지 먼저 생각해 본 이후 실험을 검증해 나가야 할 것 같다.   * + 1. **실습 3단계**   사용자의 Switch 입력을 통해 1. Fast PWM, 2. PC PWM 모드 중 하나를 선택하고 사용자에게 USART 통신을 통해 서보 모터의 각도 값을 입력 받아 모터를 회전  이 실험은 굉장히 많은 시간이 들 것 같다. 왜냐하면 USART는 실험 단계 중 가장 오래 걸렸던 부분이고 사용자의 입력을 통해 두 모드 중 하나를 선택하여 모터를 구동해야 하는데 두 모드를 한 번에 구현해야 하기 때문이다. 신경 써야하는 부분이 많은 만큼 어려울 것 같다. | | | | |
|  | | | | |