C basic language

임베디드스쿨 2기

Lv1과정

2021. 06. 25

김효창

Interrupt

〈ctype.h〉: Character Operations
〈errno.h〉: System Errors
〈math.h〉: Mathematics
〈stdint.h〉: Standard Integer Types
〈stdio.h〉: Standard IO facilities
〈string.h〉: Strings
〈avr/eeprom.h〉: EEPROM handling
〈avr/interrupt.h〉: Interrupts
〈avr/io.h〉: AVR device-specific IO definitions
〈avr/power.h〉: Power Reduction Management
〈avr/sleep.h〉: Power Management and Sleep Modes
〈avr/version.h〉: avr-libc version macros
〈avr/wdt.h〉: Watchdog timer handling
〈util/atomic.h〉 Atomically and Non-Atomically Executed Code Blocks
〈util/delay.h〉: Convenience functions for busy-wait delay loops
〈util/parity.h〉: Parity bit generation
〈util/setbaud.h〉: Helper macros for baud rate calculations
〈util/twi.h〉: TWI bit mask definitions

PWM 출력 OCR

문제
OCR 출력이 아래에서 위로 갑자기 틀어진다.

과정
오실로스코프로 PWM 출력 확인.
배선 연결 상태
작성된 코드 확인

결과
기존 : OC1 의 3 pin 사용 , OC3 의 1 pin 사용 (하나의 타이머에 pin 여러 개를 사용하면 부하가 많이 흐른다???)
개선 : OC1 의 2 pin 사용 , OC3 의 2 pin 사용

전문가
16 Bit 레지스터의 High Bit 는 서로 공유되므로, High Bit 사용하는 지점을 critical section 으로 구역 설정해라 (16 Bit 처리하는 인터럽트가 16 Bit 를 사용하는 코드 중간에 발생하면 High Bit 가 이리저리 걸리거나 뒤섞여 풀기 힘들다.)
critical section : 특정 코드 구간은 방해를 받지 않도록 설정하는 것.

critical section 설정하는 방법
cli(); 전체 인터럽트 disable
. (high bit 사용되는 code)
sei(); 전체 인터럽트 enable

어느 시점에 불쑥 나타나는 것인가?
전원 투입 후 한 번 튀는 것인가? 많이 튀는 것인가?

한 번 튀면 초기화 과정에서 쓰여진 값
반복적으로 튀는 값이 다르면, 지속적으로 어떤 것이 쓰여진 값
OCR1C 부분만 test 시도.
레지스터 < 메모리를 사용
레지스터를 사용한다면, high level function(critical section)를 만들어서 보호.
레지스터를 직접 조작하면 잠재적 error 발생 할 수 있다.

PWM 은 레지스터 사용하는 것이 맞다.
불규칙하게 튀는 값이 일정한 값인지 확인
일정한 값이면 정해진 루틴에서 정해진 값으로 덮어쓰는 것.

AVR 의 predefined routine : 시스템 프로그래밍 확장은 다음과 같은 추가 기능을 정의합니다.
사전 정의 된 절차 및 기능.

하나의 레지스터가 다양하게 쓰일 때가 많다.
코드 상의 오류가 없다면 데이터시트 정밀 분석하여 용도가 적절한지 확인.
특정 레지스터를 정해진 약속대로 쓰지 않으면 오류가 발생할 수 있다.

타이머 상승엣지 인터럽트 완료 후 하강엣지 인터럽트 발생해야 되는데 중간에 다른 인터럽트 발생해서
OCR 튀는 현상이 발생한 것일 수도 있다.
MPU6050 센서 수신 인터럽트 내용을 하강 엣지 인터럽트로 바꿔서 튀는 현상이 사라졌다.

7 Segment

기계어 언어 통일 → 16진수
0xff → 1111(f) 1111(f)

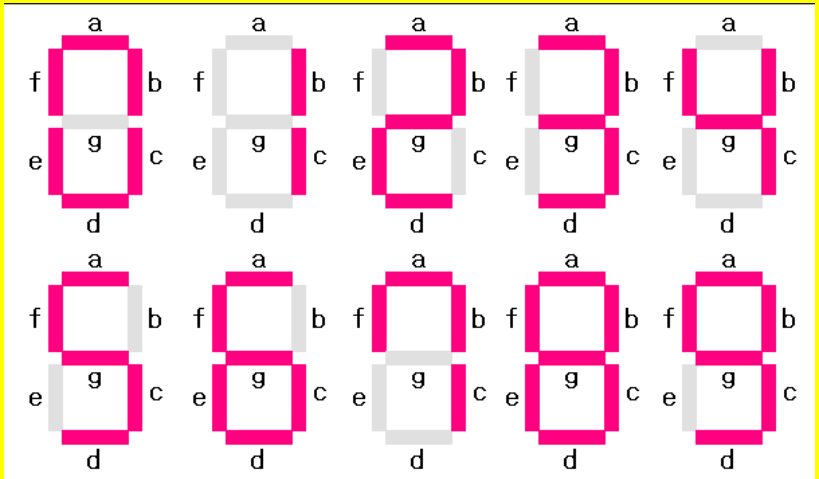
0x01 , 0x02 , 0x04 ··· 0x80
2의 배수로 증가 → Shift 연산

7 segment 숫자 0 표현 → OR 연산
0x3f = 0x01 + 0x02 + 0x04 + 0x08 +
0x10 + 0x20

7 segment 숫자 9 표현 → OR 연산
0x67 = 0x01 + 0x02 + 0x04 + 0x20 + 0x40

0x01 0x02 0x04 0x08

0x10 0x20 0x40 0x80



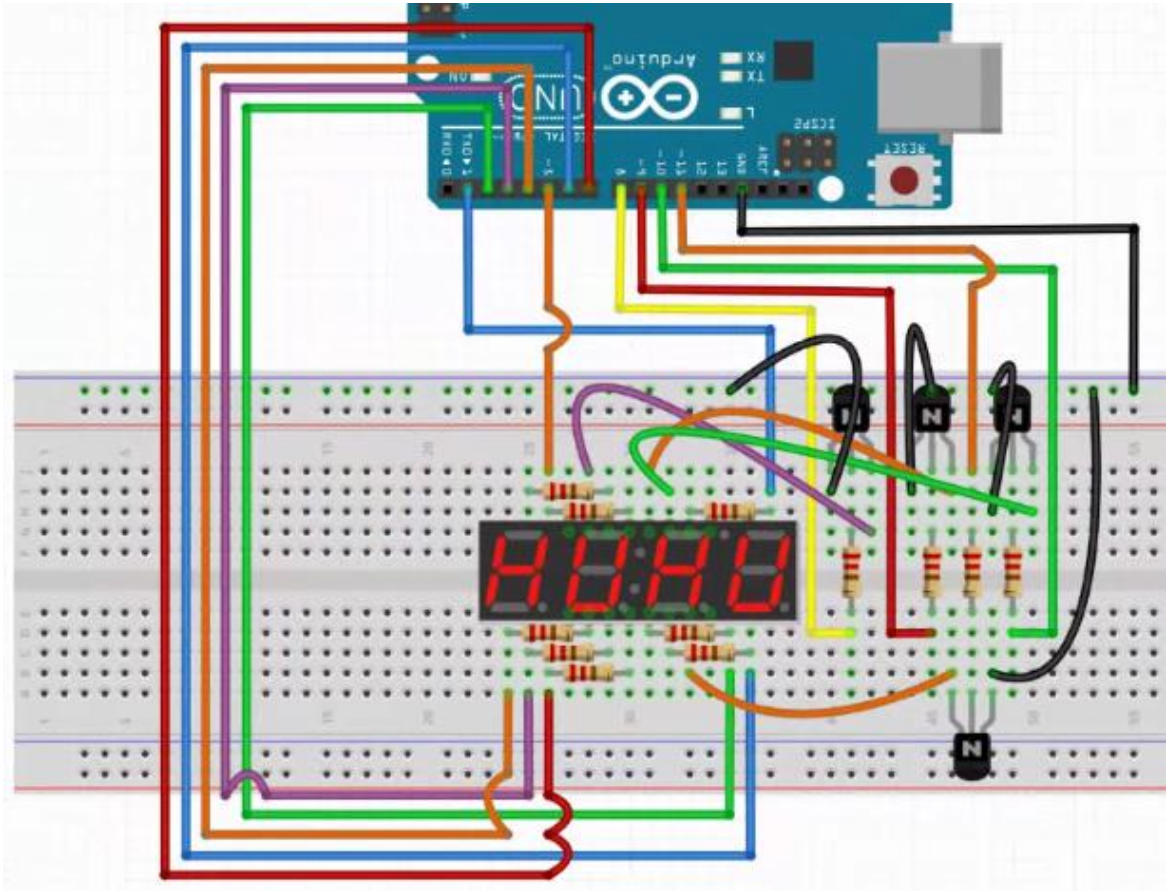
Timer (주제 : 구동테스트)

```
unsigned char digit[10] = { 0x3f , 0x06 , 0x5b , 0x4f , 0x66 , 0x6d , 0x7c , 0x27 , 0x7f , 0x67 };
: 0 ~ 9 생성 ( DDRD , PORTD )
unsigned char digit[4] = { 0x01 , 0x02 , 0x04 , 0x08 };
: 0 ~ 3 생성 ( DDRB , PORTB )

for ( i = 0; i < 10; i++ )
{
      PORTD = digit{ i ];
      _delay_ms(500);
}

while(1)
{
      for( i = 0; i < 4; i++)
      {
            PORTD = digit[ 4 - i ];
            PORTB = digit_select[i];

            for( j = 0; j < 10; j ++ )
            {
                  PORTD = digit[ j ];
                  _delay_ms(100);
            }
      }
}
```



Timer (주제 타이머)

/ : a / b , a 를 b로 나눈 몫
% : a % b , a를 b로 나눈 나머지

PORTD 는 데이터 신호
PORTB 는 전원 제어 신호

count	/	%	몫	나머지
485		10		5
485	10	10	48	8
485	100	10	4	4
485	1000	10	0	0

```
if ( count == 1000 )
{ count = 0; }
```

오버플로우 , 파형 일그러짐 방지

```
fnd [ 0 ] = ( count / 1000 ) % 10;
fnd [ 0 ] = ( count / 1000 ) % 10;
fnd [ 0 ] = ( count / 1000 ) % 10;
fnd [ 0 ] = ( count / 1000 ) % 10;
```

FND 값 할당

```
for ( i = 0; i < 4; i++ )
{
      PORTD = digit [ fnd [ i ] ];
      PORTB = digit_select [ i ];
      _delay_us(2490);
}
```

FND 값 표시

2490 × 4 = 9960 us

fnd[3]	fnd[2]	fnd[1]	fnd[0]
0	0	0	0
0	0	0	1
0	0	0	2
0	0	0	3
0	0	0	4
0	0	0	5
0	0	0	6
0	0	0	7
0	0	0	8
0	0	0	9
0	0	1	0

select 3	select 2	select 1	select 0
for 3	for 2	for 1	for 0
0	0	0	0
0	0	0	1
0	0	0	2
0	0	0	3
0	0	0	4
0	0	0	5
0	0	0	6
0	0	0	7
0	0	0	8
0	0	0	9
0	0	1	0

트랜지스터

VCC , GND 최대 전류 200 mA 이므로 전체 전류 100mA 이하로 설계.
8 개의 PORT 사용하여 모두 10 mA 흐르면 총합 80 mA 이므로 합격 (100 초과하면 IC 손상)

전류를 안정적으로 제공하기 위해 FND 와 MCU 사이에 트랜지스터 활용.

3. Although each I/O port can sink more than the test conditions (20mA at V_{CC} = 5V, 10mA at state conditions (non-transient), the following must be observed:
   ATmega328P:
   1] The sum of all I_{OL}, for ports C0 - C5, should not exceed 100mA.
   2] The sum of all I_{OL}, for ports B0 - B5, D5 - D7, XTAL1, XTAL2 should not exceed 100mA.
   3] The sum of all I_{OL}, for ports D0 - D4, should not exceed 100mA.
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not gua greater than the listed test condition.
4. Although each I/O port can source more than the test conditions (20mA at V_{CC} = 5V, 10mA state conditions (non-transient), the following must be observed:
   ATmega328P:
   1] The sum of all I_{OH}, for ports C0 - C5, D0- D4, should not exceed 150mA.
   2] The sum of all I_{OH}, for ports B0 - B5, D5 - D7, XTAL1, XTAL2 should not exceed 150mA.

질의응답

```
for ( i = 0; i < 10; i++ )
{
        PORTD = digit[ i ];
        _delay_ms(500);
}
```

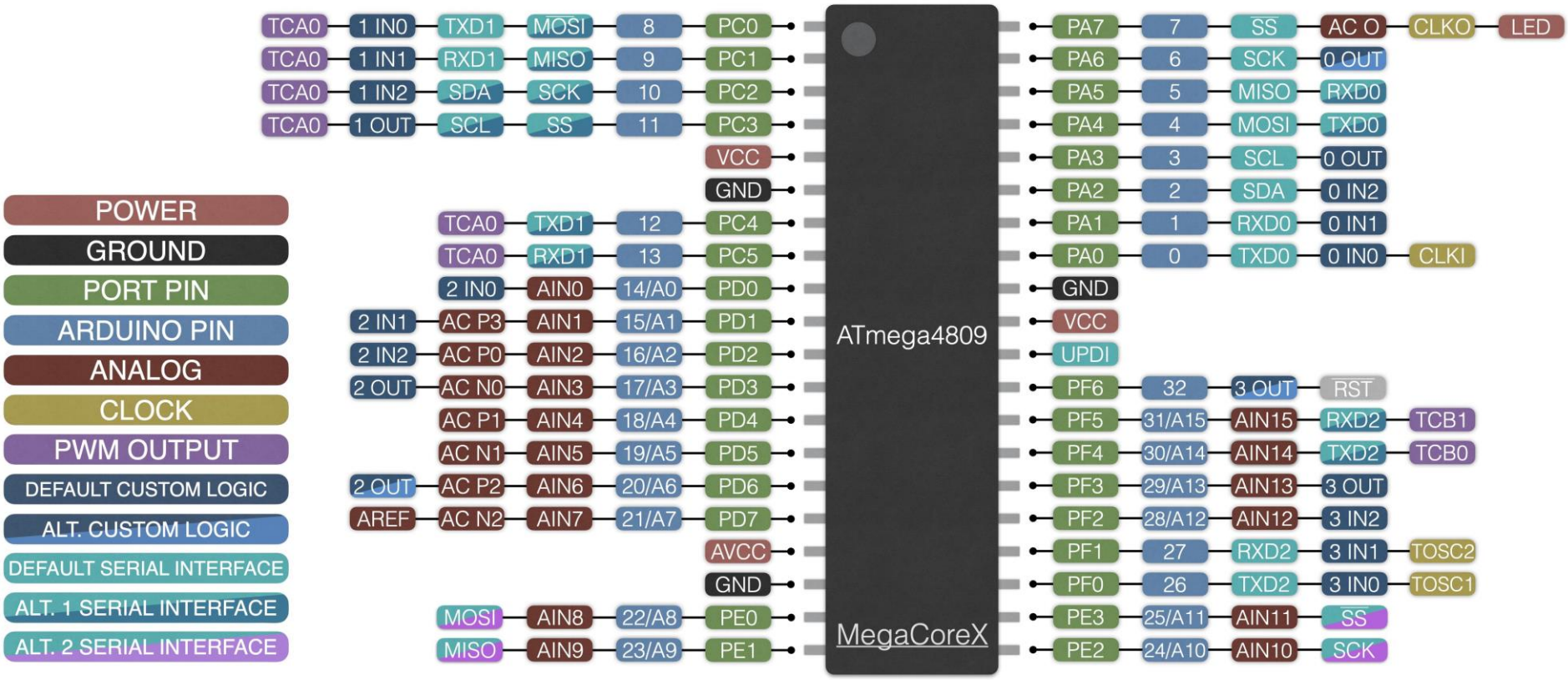
초반 구동 테스트용 코드

이중 for 문으로 동작하여 기능 구현이 정상적으로 표현되기 때문에 digit [4 - i] 는 필요없다.

주제 , 목적을 파악하자
질문을 지금과 같이 명확하게 해주셔야 의사 전달이 용이
전력 사용량 체크하고 전력 부족하면 외부 전원 7805 같은거 끌어다 사용하면 될 것 같네요.
구현하려는 내용들에 대해 io가 부족한지 여부도 체크

상

DIP-40 standard pinout



Note that the PWM output pins have been pin swapped and TCA0 is configured as split mode. Refer to the datasheet for detailed information on pin swapping and peripherals.

End of Document