簡易AC電力制御(波数調整)

概要

以前に製作したAC電力制御V2の部品点数を減らした、簡易版を製作しました。

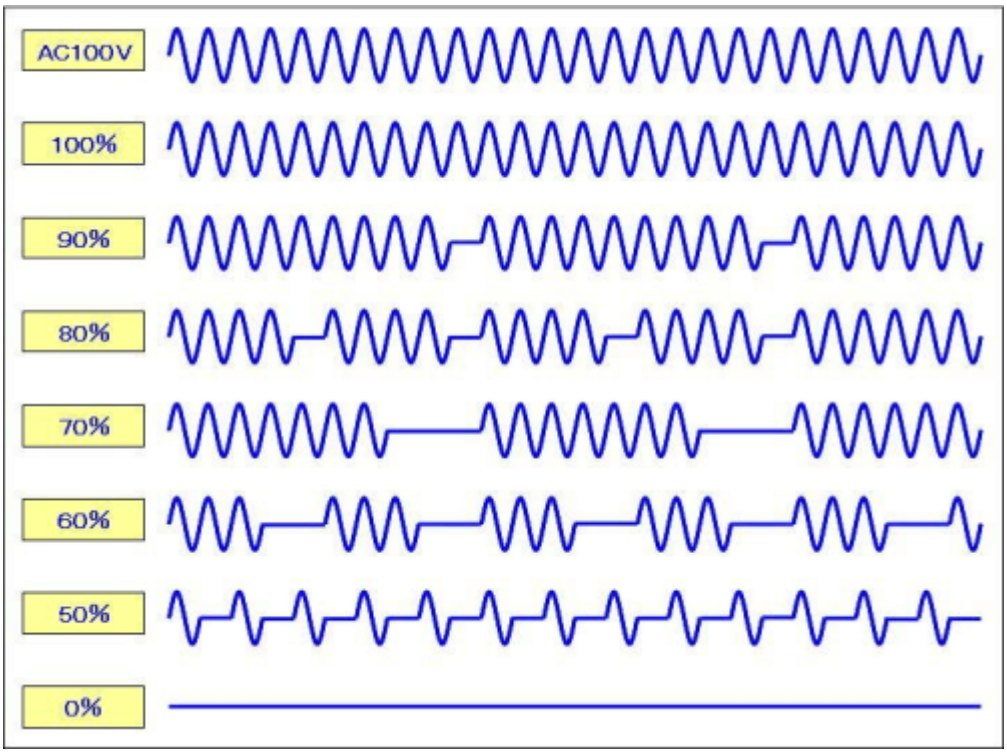
<仕様>

- 出力波数の比率を、0%、50%、60%、70%、80%、100%の7段階に調整できます。
- 60Hz50Hzの商用電源に対応します。
- 比率及び商用電源のモードを、内蔵のEEPROMに記憶し、起動毎の調整作業を不要としました。
- 波形のON/OFF制御は、ゼロクロス方式を採用し、ノイズの発生を防ぎます。

動作原理(ハードウェア)

以前に製作したAC電力制御V2では、精度を上げるために、商用電源(AC100V)の波形に同期させる方式としましたが、今回は、若干精度は落ちますが、非同期方式を採用し、部品点数を少なくしました。

◎出力波形



	0%	50%	60%	70%	80%	90%	100%
ON:OFFの比率	0:1	1:1	3:2	7:3	4:1	9:1	1:0

◎比率(%)の表示 LEDを6個使用して、表示します。

	0%	50%	60%	70%	80%	90%	100%
LED1	○	●	○	○	○	○	○
LED2	○	○	●	○	○	○	○
LED3	○	○	○	●	○	○	○

LED4	○	○	○	○	●	○	○
LED5	○	○	○	○	○	●	○
LED6	○	○	○	○	○	○	●

◎商用電源(100V)のON/OFF 秋月電子で販売している、ゼロクロスタイプの「ソリッド・ステート・リレー(SSR)キット」を使用します。

動作原理(ソフトウェア)

◎メイン処理

- 起動時にスイッチ(SW1)が、ONであれば、商用電源モード設定処理を呼び出します。
- EEPROMに書き込まれている、商用電源モード(60Hzまたは50Hz)の値を読み込みます(cycle)
- EEPROMに書き込まれている、波数比率の値を読み込みます(ratio)
- 割り込み処理を発生させるための、タイマーを起動します。
- スイッチ(SW1)が押下されるのを待ちます。
- 波数比率の値をインクリメントします。
  0%→50%→60%→70%→80%→100%→0%→・・・・・・・・・・・・
- 波数比率の値をEEPROMに書き込みます。

◎商用電源モード設定処理

- 既定値は、60Hzです。
- スイッチ(SW1)が押下される毎に、50Hz(LED7点灯)→60Hz(LED7消灯)を繰り返します。
- 商用電源モードの値をEEPROMに書き込みます。
- 再起動で、通常の処理に戻ります。

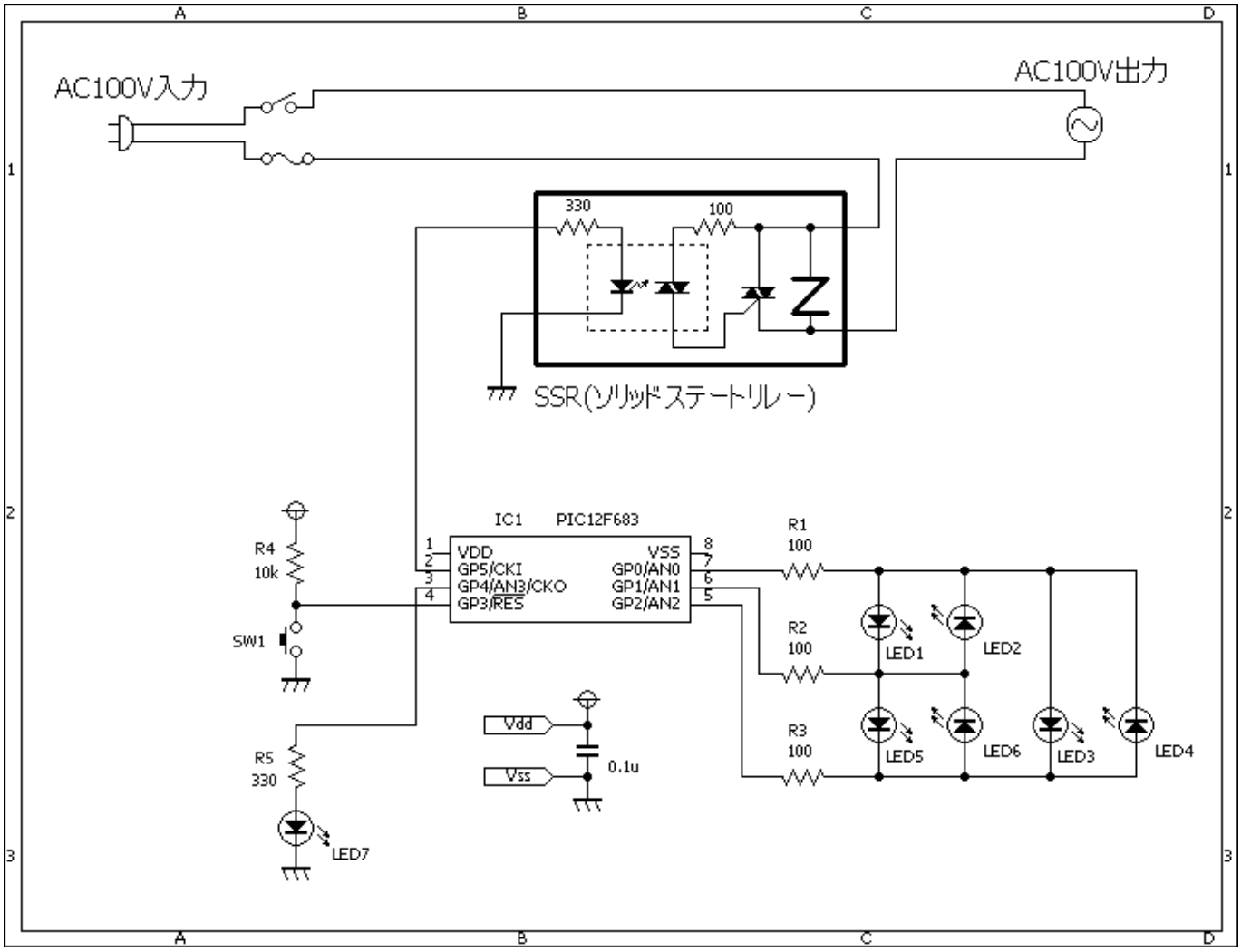
◎LED点灯処理

- 3個のポートを使用して、6個のLEDを点灯させます。
- 仕組みについては、簡易レベルメータ(LED×12個)を参照してください。

◎割り込み処理

- CCPとTIMER1を使用して、割り込みを発生させます(init_ccp_compare関数)
  商用電源が、60Hzの場合には→16.7msecの割り込み周期に設定します。
  商用電源が、50Hzの場合には→20.0msecの割り込み周期に設定します。
- 設定された波数(比率)に応じて→SSRをON/OFFさせます。
- 1秒周期で、LED7を点滅させます。

回路図



ソースコード

ac_control_v4.c

```
//*****************************************************************************
*
/*
      <簡易□□電力制御（波数調整）>
*/
//*****************************************************************************
*
//SWITCH
sbit    SW          at  GPIO.B3;
sbit    SSR         at  GPIO.B5;
sbit    LED         at  GPIO.B4;
//LED
#define LED_OFF     0
#define LED1        1
#define LED2        2
#define LED3        3
#define LED4        4
```

```
#define LED5        5
#define LED6        6
//OTHER
#define CYCLE_60HZ  0
#define CYCLE_50HZ  1
#define BYTE        unsigned  short
#define WORD        unsigned  int
//波形パターン
WORD    ratio_pattern[] = {
        0b0000000000000000,       //  0%
        0b0000001010101010,       // 50%
        0b0000001110011100,       // 60%
        0b0000001111111000,       // 70%
        0b0000001111011110,       // 80%
        0b0000001111111110,       // 90%
        0b0000001111111111        //100%
};
//*****************************************************************************
*
extern  void    main();
extern  void    init_ccp_compare();
extern  void    interrupt();
extern  void    led_cntl(short num);
extern  void    switch_on_check();
extern  void    set_mode();
extern  void    opening_demonstration();
//*****************************************************************************
*
//■■■メイン関数■■■
BYTE    ratio;
BYTE    cycle;
void    main()
{
        OSCCON = 0b01110000;
        CMCON0 = 0b00000111;
        ANSEL  = 0b00000000;
        TRISIO = 0b00001000;
        //
        SSR = 0;
        LED = 0;
        //
        opening_demonstration();
        //
        if (SW == 0) {
                while (Button(&GPIO, 3, 1, 1) == 0)
                        ;
                //
                set_mode();
        }
        //周波数の読み込み
        cycle = EEPROM_Read(0);
```

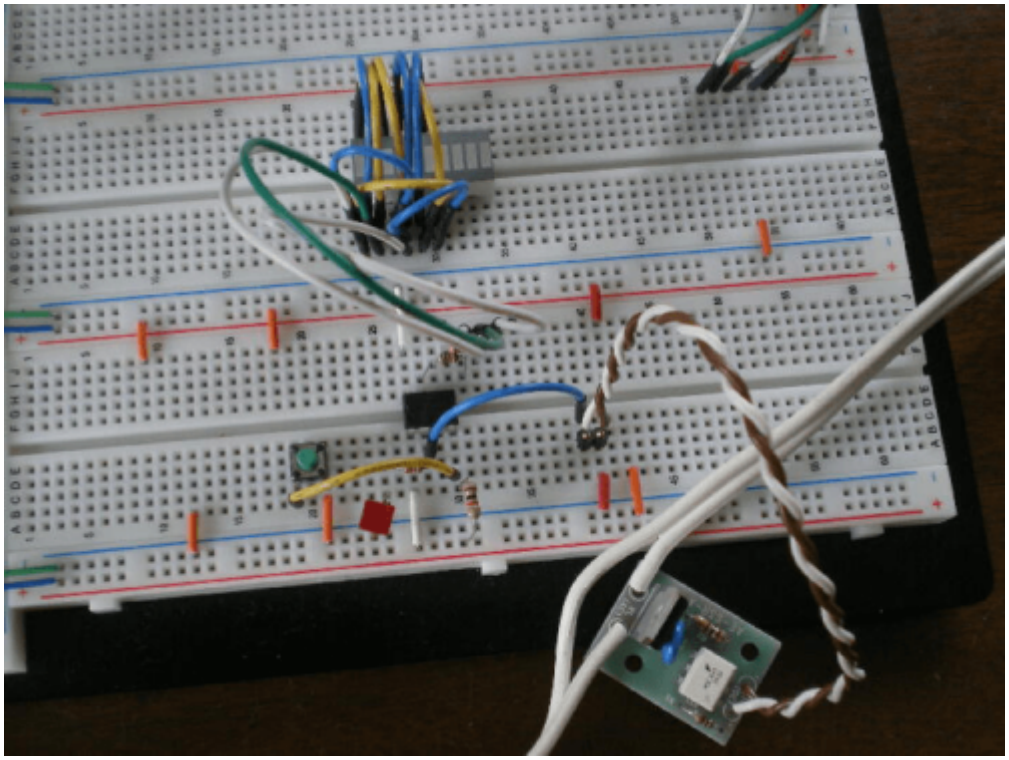
```
        if (cycle >= 2) {
                cycle = CYCLE_60HZ;
        }
        //波形パターンの読み込み
        ratio = EEPROM_Read(1);
        if (ratio >= 7) {
                ratio = 0;
        }
        led_cntl(ratio);
        //
        init_ccp_compare();
        //割り込みを許可します。
        INTCON.PEIE = 1;
        INTCON.GIE = 1;
        //
        while (1) {
                switch_on_check();
                ratio++;
                if (ratio == 7) {
                        ratio = 0;
                }
                EEPROM_Write(1, ratio);
                led_cntl(ratio);
        }
}
//********************************************************************
*
void    opening_demonstration()
{
        short   cnt1, cnt2;
        //
        for (cnt1 = 0; cnt1 < 5; cnt1++) {
                for (cnt2 = 0; cnt2 < 7; cnt2++) {
                        LED_cntl(cnt2);
                        Delay_ms(30);
                }
                for (cnt2 = 0; cnt2 < 7; cnt2++) {
                        LED_cntl(6 - cnt2);
                        Delay_ms(30);
                }
        }
        led_cntl(LED_OFF);
}
//********************************************************************
*
void    set_mode()
{
        short   cnt;
        //
        for (cnt = 0; cnt < 5; cnt++) {
                LED = 1;
```

```
                Delay_ms(100);
                LED = 0;
                Delay_ms(100);
        }
        //60Hz（既定値）
        EEPROM_Write(0, CYCLE_60HZ);
        while (1) {
                //50Hz
                switch_on_check();
                EEPROM_Write(0, CYCLE_50HZ);
                LED = 1;
                //60Hz
                switch_on_check();
                EEPROM_Write(0, CYCLE_60HZ);
                LED = 0;
        }
}
//*********************************************************************
*
void    led_cntl(short num)
{
        switch (num) {
        case LED_OFF:
                TRISIO.B0 = 1;
                TRISIO.B1 = 1;
                TRISIO.B2 = 1;
                break;
        case LED1:
                TRISIO.B0 = 0;
                TRISIO.B1 = 0;
                TRISIO.B2 = 1;
                GPIO.B0 = 1;
                GPIO.B1 = 0;
                break;
        case LED2:
                TRISIO.B0 = 0;
                TRISIO.B1 = 0;
                TRISIO.B2 = 1;
                GPIO.B0 = 0;
                GPIO.B1 = 1;
                break;
        case LED3:
                TRISIO.B0 = 0;
                TRISIO.B1 = 1;
                TRISIO.B2 = 0;
                GPIO.B0 = 1;
                GPIO.B2 = 0;
                break;
        case LED4:
                TRISIO.B0 = 0;
```

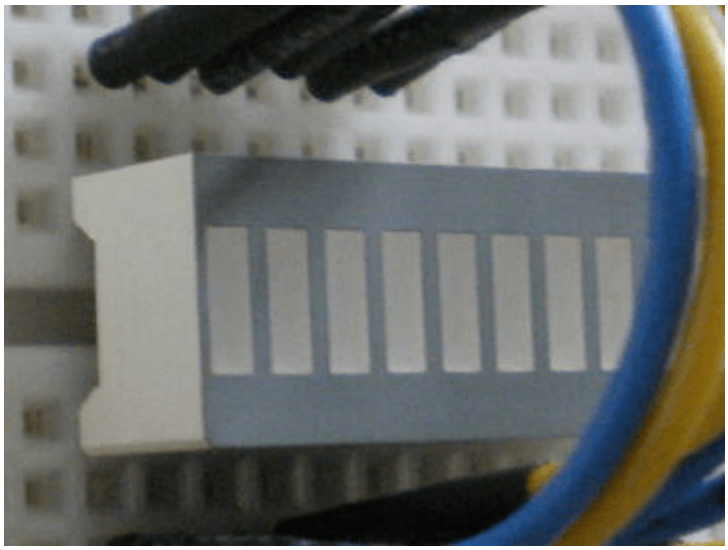
```
                TRISIO.B1 = 1;
                TRISIO.B2 = 0;
                GPIO.B0 = 0;
                GPIO.B2 = 1;
                break;
        case LED5:
                TRISIO.B0 = 1;
                TRISIO.B1 = 0;
                TRISIO.B2 = 0;
                GPIO.B1 = 1;
                GPIO.B2 = 0;
                break;
        case LED6:
                TRISIO.B0 = 1;
                TRISIO.B1 = 0;
                TRISIO.B2 = 0;
                GPIO.B1 = 0;
                GPIO.B2 = 1;
                break;
        }
}
//*****************************************************************************
*
void    switch_on_check()
{
        while (Button(&GPIO, 3, 1, 0) == 0)
                ;
        while (Button(&GPIO, 3, 1, 1) == 0)
                ;
}
//*****************************************************************************
*
void    init_ccp_compare()
{
        // CCPの設定
        PIE1.CCP1IE = 1;
        PIR1.CCP1IF = 0;
        CCP1CON = 0b00001011;
        if (cycle == CYCLE_60HZ) {
                CCPR1L = 0x35;          // 60Hz...(1÷8000000)*4*33333
                CCPR1H = 0x82;
        } else {
                CCPR1L = 0x40;          // 50Hz...(1÷8000000)*4*40000
                CCPR1H = 0x9C;
        }
        // TIMER1の設定
        PIE1.TMR1IE = 0;
        PIR1.TMR1IF = 0;
        TMR1L = 0;
        TMR1H = 0;
        T1CON.T1CKPS0 = 0;
```

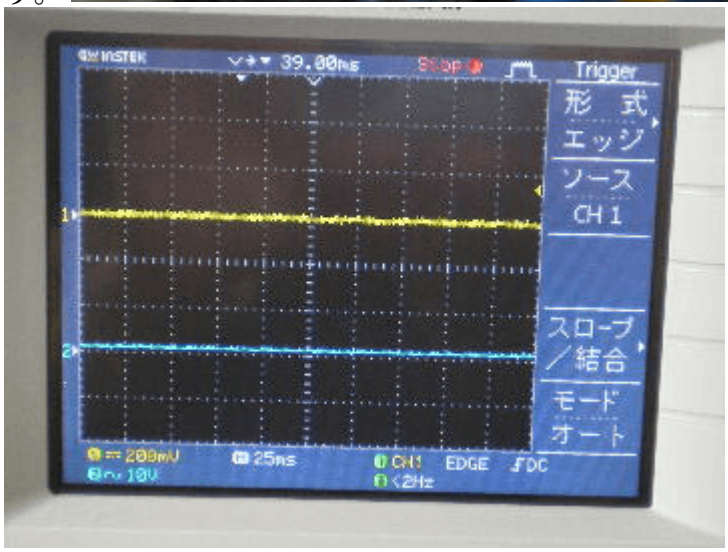
```
        T1CON.T1CKPS1 = 0;
        T1CON.TMR1ON = 1;
}
//*****************************************************************************
*
short   cnt = 0;
short   cnt2 = 0;
void    interrupt()
{
        WORD    tmp;
        //
        if (PIR1.CCP1IF == 1) {
                PIR1.CCP1IF = 0;
                //
                tmp = ratio_pattern[ratio];
                if (((tmp >> cnt) & 0x0001) == 1) {
                        SSR = 1;
                } else {
                        SSR = 0;
                }
                cnt++;
                if (cnt == 10) {
                        cnt = 0;
                }
                //
                cnt2++;
                if (cnt2 == 30) {
                        cnt2 = 0;
                        LED = ~LED;
                }
        }
}
//*****************************************************************************
*
```

動作確認

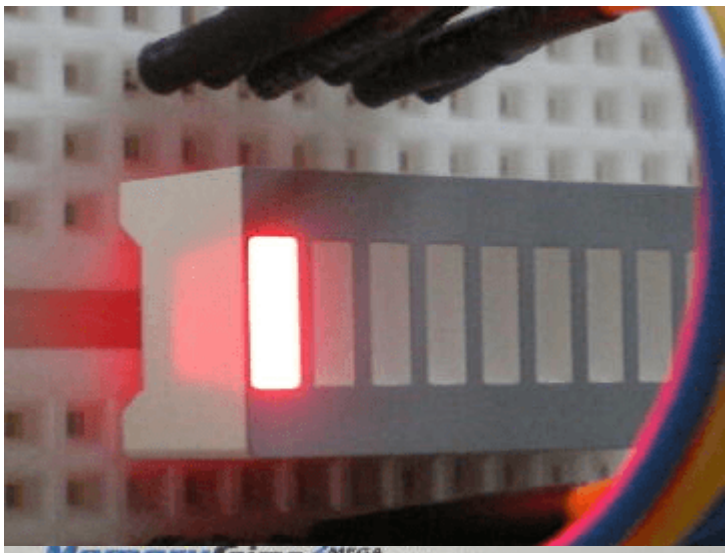


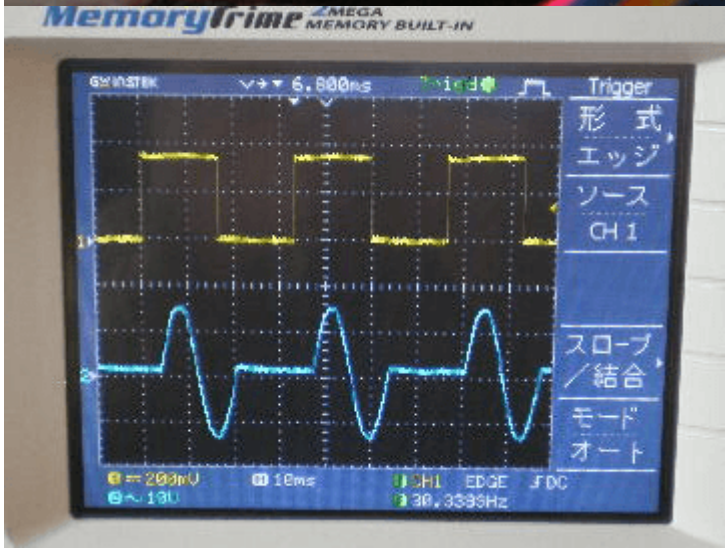
比率(0%)の時のLED(全て消灯)と波形です。 波形の黄色は、SSRのON/OFF制御波形です。 波形の青色は、商用電源の出力波形です。



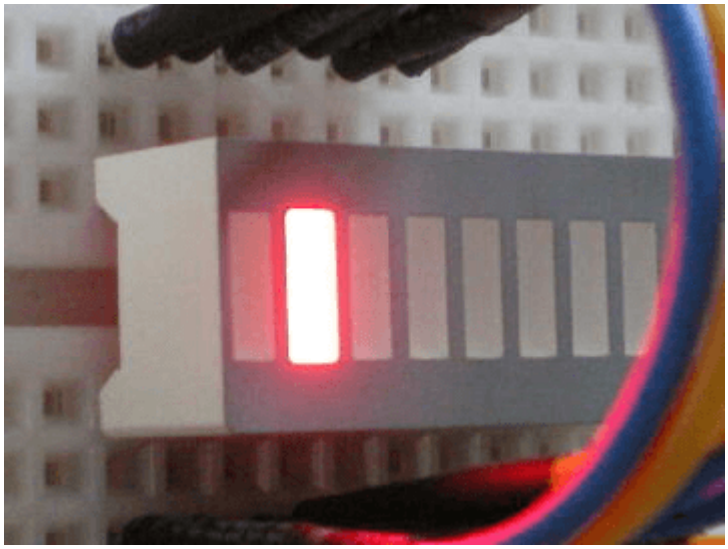


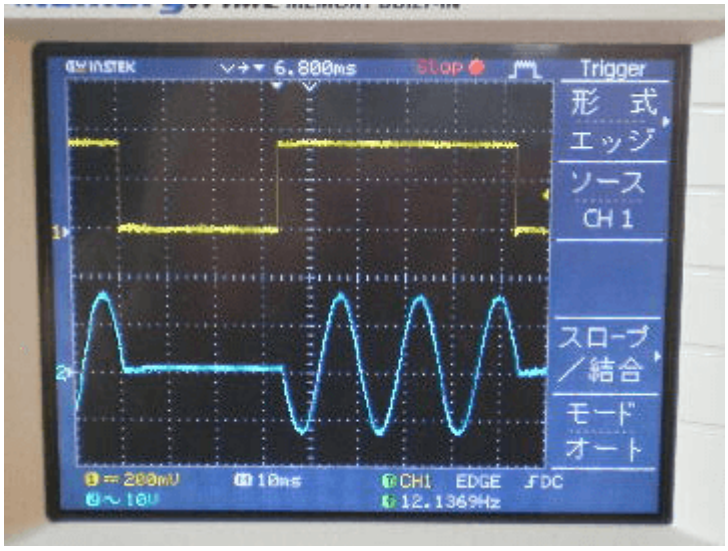
比率(50%)の時のLED1(点灯)と波形です。



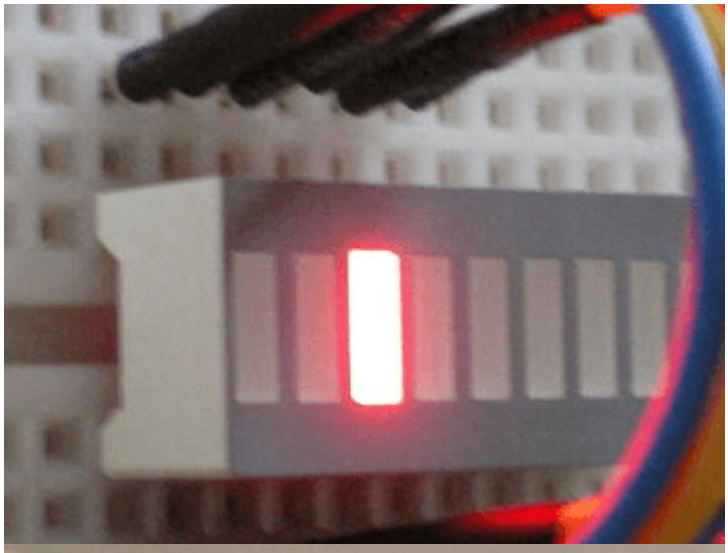


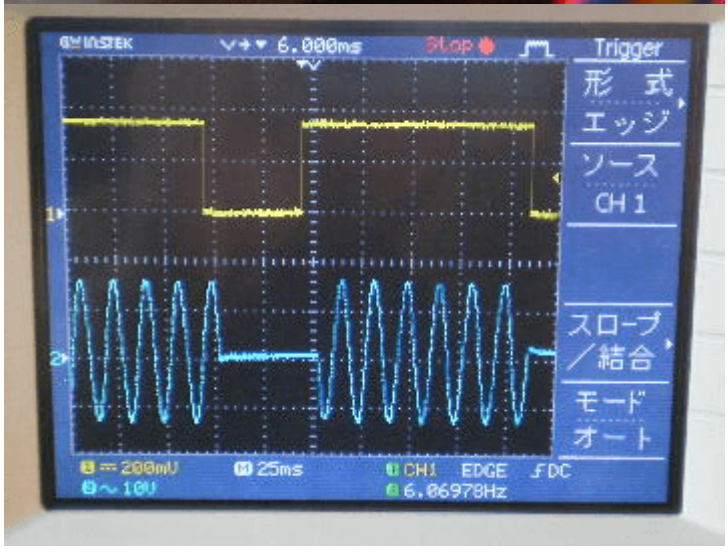
比率(60%)の時のLED2(点灯)と波形です。



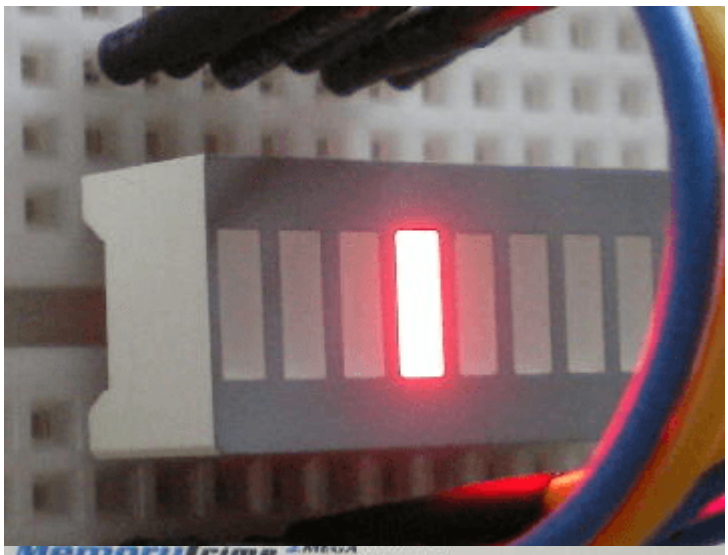


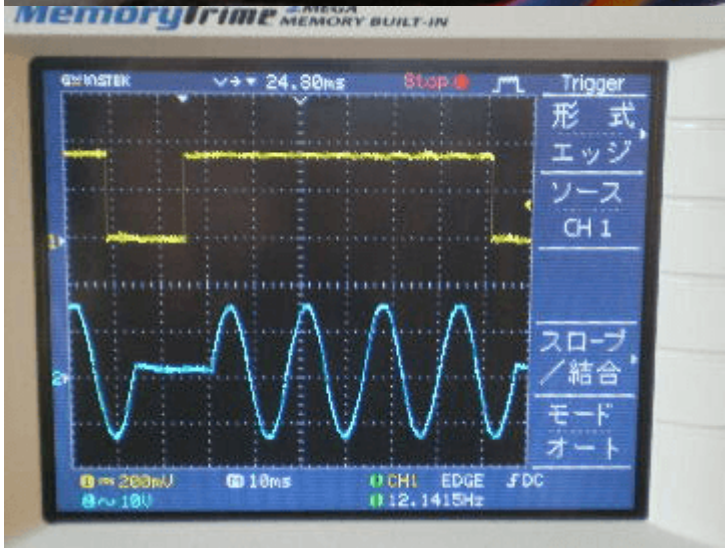
比率(70%)の時のLED3(点灯)と波形です。



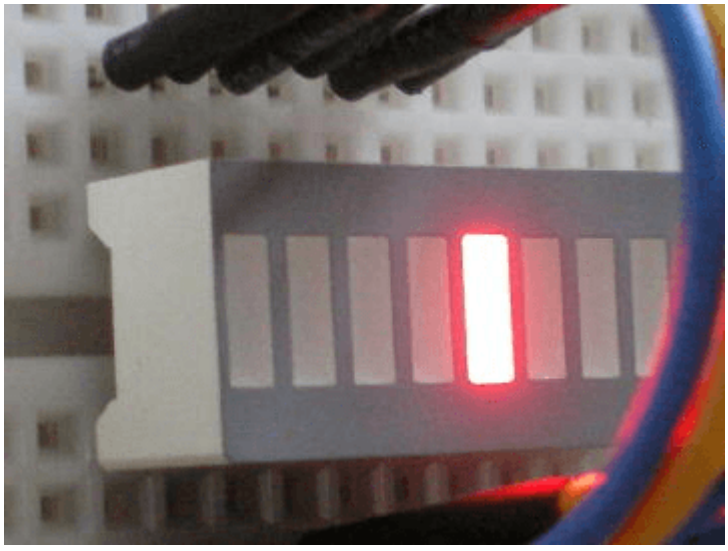


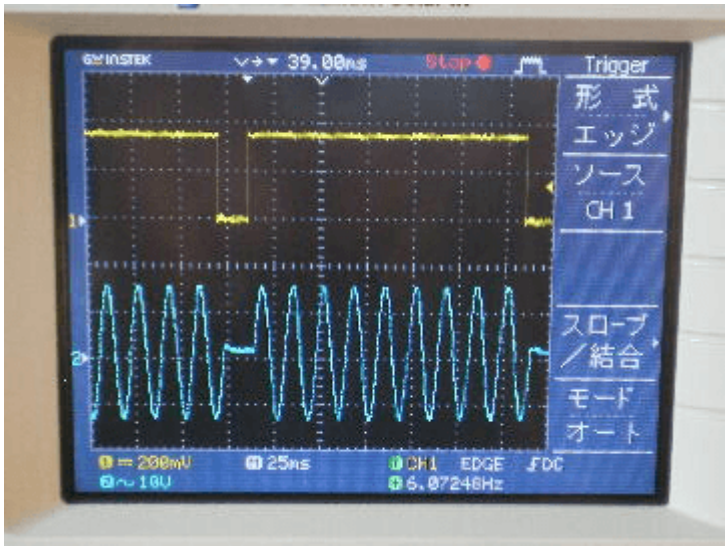
比率(80%)の時のLED4(点灯)と波形です。



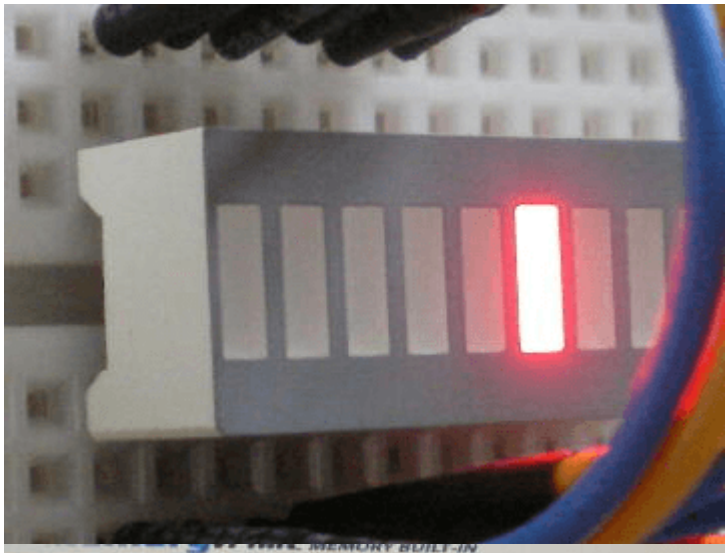


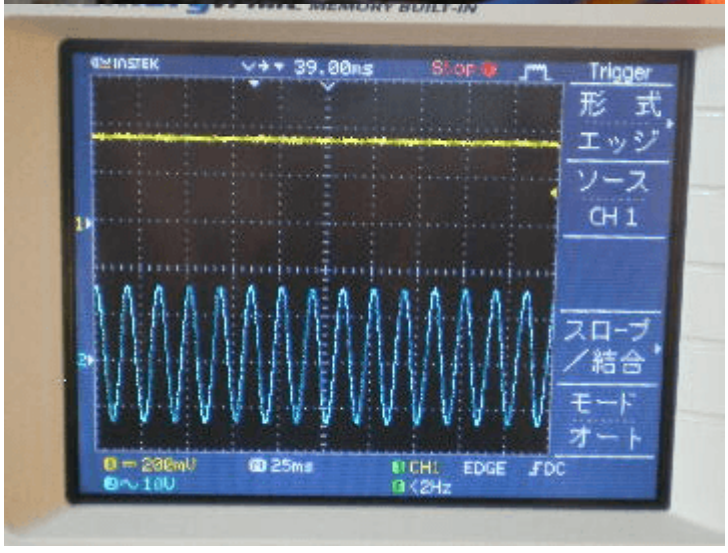
比率(90%)の時のLED5(点灯)と波形です。



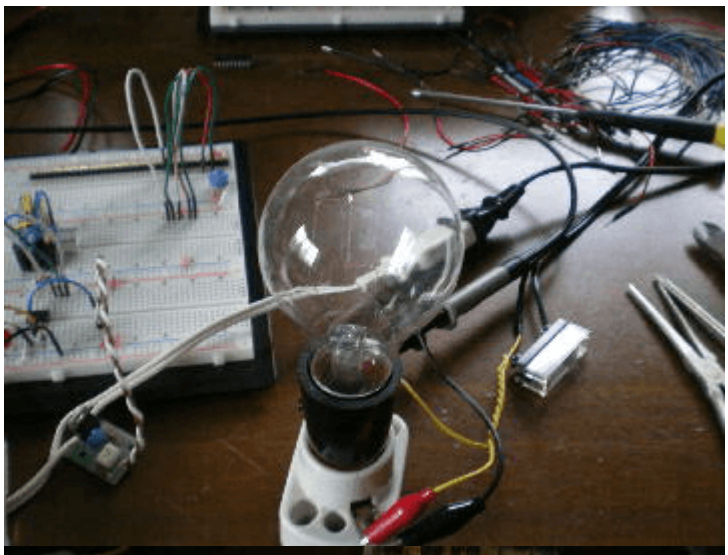


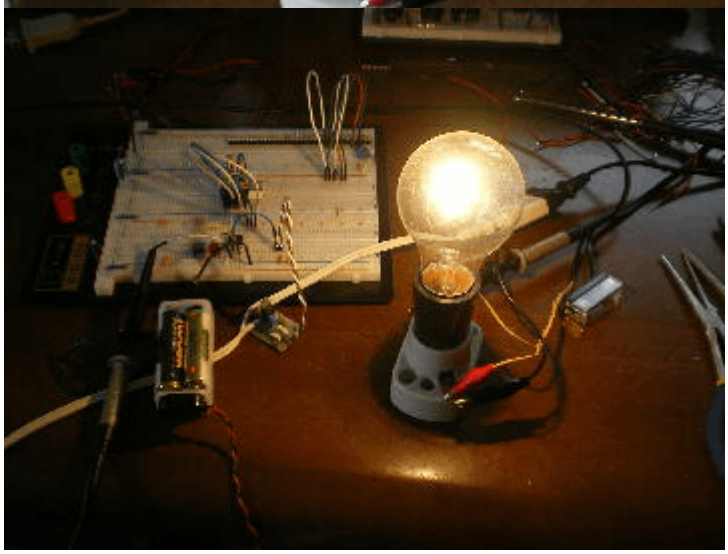
比率(100%)の時のLED6(点灯)と波形です。





電球を接続してみました。





如何ですか?

商用電源(AC100V)の波数を調整するため、電球などに使用するとチラツキが発生しますが、半田ごての温度調整などには、ノイズも発生しないので、丁度良いのではと思います。😄