```
int obMin = 200;          //ввести минимальные обороты
int obMax = 9000;         //ввести максимальные обороты
int kImp = 120;          //ввести кол-во импульсов на 10 оборотов
int minzn = 115;           //  минимальное значение симмистора на котором
начинается вращение.
int ogrmin =  70 ;         // ограничение симистора на минимальных оборотах.
int mindimming = 80;        //значение симистора при закллинившем станке
(первоначальный импульс)
int dopusk =  200 ;         //допуск оборотов в минус и плюс
int razgon = 50;            //переменная разгона 1 - 100


#include <LiquidCrystal.h>     // библиотека экрана

LiquidCrystal lcd( 9, 10, 4, 5, 6, 7); // пины экрана

int AC_LOAD = 3;    // пин управления симистором
volatile int dimming = 130;  // время задержки от нуля   7 = максимально, 130 =
минимально
volatile unsigned long time; // время в микросекундах срабатывания датчика нуля
unsigned long tims;           // переменная показаний времени

unsigned long currentTime;     //временные переменные для таймера экрана
unsigned long loopTime;

int holl = 0;    //переменная  срабатываня датчика
int pR;                  // показания регулятора
int pRR;                // переменная для расчёта.
int ogr ;                //переменная ограничений симистора натекущих оборотах
volatile int sp = 0;           //переменная суммы срабатываний датчика
volatile int prOb ;   //предвар реальн обороты
volatile int rOb ;    // реальные обороты

volatile unsigned int int_tic;    //переменные для подсчёта времени между
импульсами.
volatile unsigned long tic;
volatile int t = 0;               //минимальное время импульсов +1
int val ;
void setup()
{
  pRR = obMin;
  t = (15000 / ( obMin * (kImp / 10))) * 2; //высчитываем минимальное время
импульсов +1
  pinMode(AC_LOAD, OUTPUT);        // назначаем выходом
  attachInterrupt(0, zero_crosss_int, RISING);  // прерывание по пину 2

  lcd.begin(16, 2);          //дисплей 16символов 2строчки
```

```
  lcd.setCursor(0, 0);

  lcd.write("R:");  //в верхней строке выводим время задержки

  lcd.setCursor(0, 1);

  lcd.write("t:");  // В нижней выводим показания датчика

  lcd.setCursor(8, 0);

  lcd.write("S:");  //в верхней строке будем выводить требуемые обороты

  lcd.setCursor(8, 1);

  lcd.write("S:");  // В нижней выводим фактичесские обороты


  pinMode (8, INPUT); // вход сигнала ICP( №8 only для atmega328)
  //настройка 16 бит таймера-счётчика 1
  TCCR1B = 0; TCCR1A = 0; TCNT1 = 0;
  TIMSK1 = (1 << ICIE1) | (1 << TOIE1); //создавать прерывание от сигнала на пине
ICP1
  TCCR1B = (1 << ICNC1) | (1 << ICES1) | (1 << CS10); //div 1

}

ISR (TIMER1_CAPT_vect) { //прерывание захвата сигнала на входе ICP1
  tic = ((uint32_t)int_tic << 16) | ICR1 ; //подсчёт тиков
  ICR1 = 0; int_tic = 0; TCNT1 = 0;
  sp = sp + 1 ; // для подсчёта оборотов в минуту.
  holl = holl + 1;
}   // после каждого срабатывания датчика холл+1


ISR (TIMER1_OVF_vect) { //прерывание для счёта по переполнению uint
  int_tic++; //считать переполнения через 65536 тактов
  if (int_tic > t) {
    tic = 0;  //если на входе пусто более минимального времени то обнулить
счётчики
    int_tic = 0;
  }
  if (int_tic > 500) {
    dimming = 130; // если стоим 2 секунды, то сбрасываем напряжение.
  }
}

// the interrupt function must take no parameters and return nothing
void zero_crosss_int()  // function to be fired at the zero crossing to dim the
light
```

```
{
  time = micros();

}

void loop()  {


  val = analogRead(A0);
  pR = map(val, 0, 1023, obMin, obMax); //Приводим показания регулятора к минимальным и максимальным оборотам

  if (val > 0) {                                        //   если регулятор больше 0
    if ( holl >= 1) {                                   // если сработал датчик
      prOb = 60000000 / ((tic * 0.0625 ) * kImp / 10); //Высчитываем обороты по показаниям датчика
      if ( prOb >= 0) {                                 //проверяем на соответствие.
        rOb = prOb ;                                    //если нормально, записываем в реальные обороты

      }
      if ( rOb < pR ) {                                 //сверяем показания регулятора и реальные обороты
        int fff = pR  - rOb;                            //узнаём разницу между оборотами
        int pRu = map(fff, 1, obMax, 1, razgon);        //исходя из разницы и разгона высчитываем на сколько увеличить переменную для расчёта
        pRR = pRR + pRu  ;                              //увеличиваем переменную расчёта
      }
      if ( pR < (rOb - 20) ) {                          //сверяем показания регулятора и реальные обороты
        int fff = rOb - 20  - pR;                       //узнаём разницу между оборотами
        int pRu = map(fff, 1, obMax, 1, razgon);        //исходя из разницы и разгона высчитываем на сколько уменьшить переменную для расчёта
        pRR = pRR - pRu ;                               //увеличиваем переменную расчёта
      }
      pRR = constrain(pRR, (pR / 2), obMax);            //задаём пределы переменной для расчёта.
      ogr = map(val, 0, 1023, ogrmin, 7);               //исходя из показаний регулятора узнаём на сколько может быть открыт симистор.

      dimming = map(rOb, (pRR - dopusk), (pRR + dopusk), ogr, minzn);
//рассчитываем управление симистором.
      holl = 0;                                                                 // обнуляем
```

```
срабатывание датчика
    }

    if (tic == 0) {                                    // если двигатель не
вращается
      dimming = mindimming ;                           //  время задержки равно
первоначальному импульсу
    }
    dimming = constrain(dimming, ogr, minzn) ;       //  Следим чтоб время
задержки было не меньше ограничения и не больше минимального значения
  }
  else {
    dimming = 130;                                   //Если регулятор на 0 то
время задержки 130
    pRR = obMin;
  }


  int dimtime = (75 * dimming);                      // For 60Hz =>65
  tims = micros();                                   // считываем время, прошедшее с
момента запуска программы
  if (tims >= (time + dimtime)) {                    //если время больше или равно
времени срабатывания нуля + время задержки

    digitalWrite(AC_LOAD, HIGH);                     // открываем симистор
    delayMicroseconds(10);                           // задержка 10 микросекунд (для 60Hz
= 8.33)
    digitalWrite(AC_LOAD, LOW);                      // выключаем сигнал на симистор.
  }
  else {}




  // Для вывода значений на дисплей 2 раз в секунду

  currentTime = millis();                            // считываем время, прошедшее
с момента запуска программы
  if (currentTime >= (loopTime + 500)) {             // сравниваем текущий таймер
с переменной loopTime + 0,5 секунд

    // выводим показания датчика

    lcd.setCursor(2, 0);
    lcd.print(dimming );
    lcd.print("  ");     // выводим время задержки на экран.

    lcd.setCursor(2, 1);
    lcd.print(val );
```

```
    lcd.print("  ");     // выводим нужные обороты на экран.

    lcd.setCursor(10, 0);
    lcd.print(map(val, 0, 1023, obMin, obMax));
    lcd.print("       ");     // выводим нужные обороты на экран.

    lcd.setCursor(10, 1);

    lcd.print (sp * (1200 / kImp)); // выводим средние обороты на экран.
    lcd.print("       ");
    sp = 0;
    loopTime = currentTime;                          // в loopTime записываем
новое значение
  }
}
```