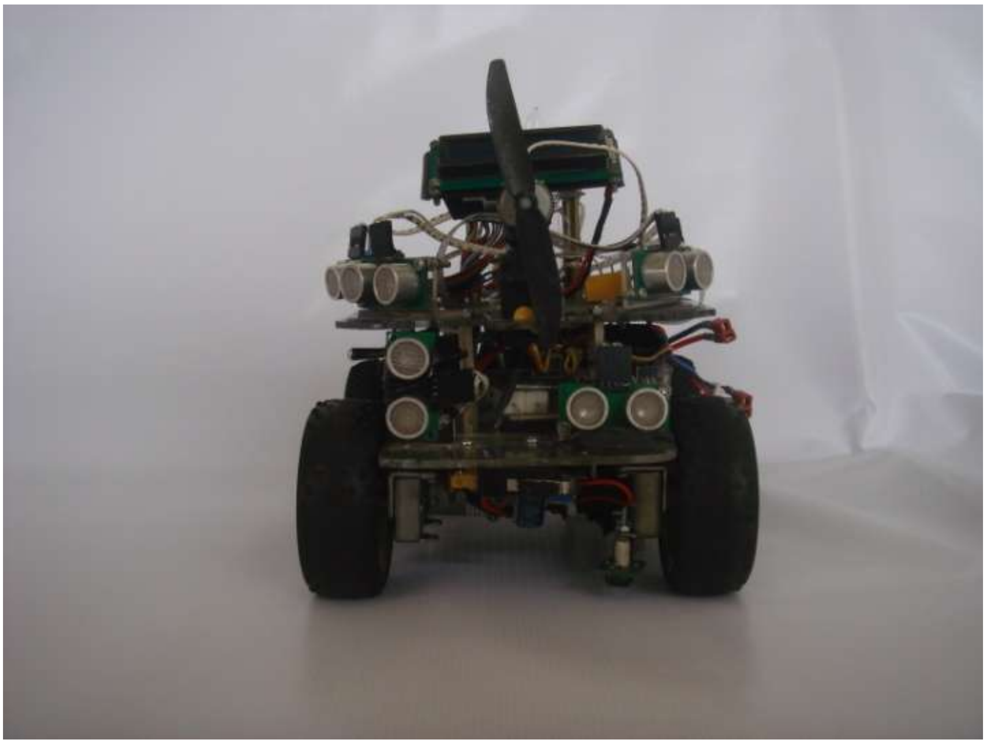
LAMPIRAN A

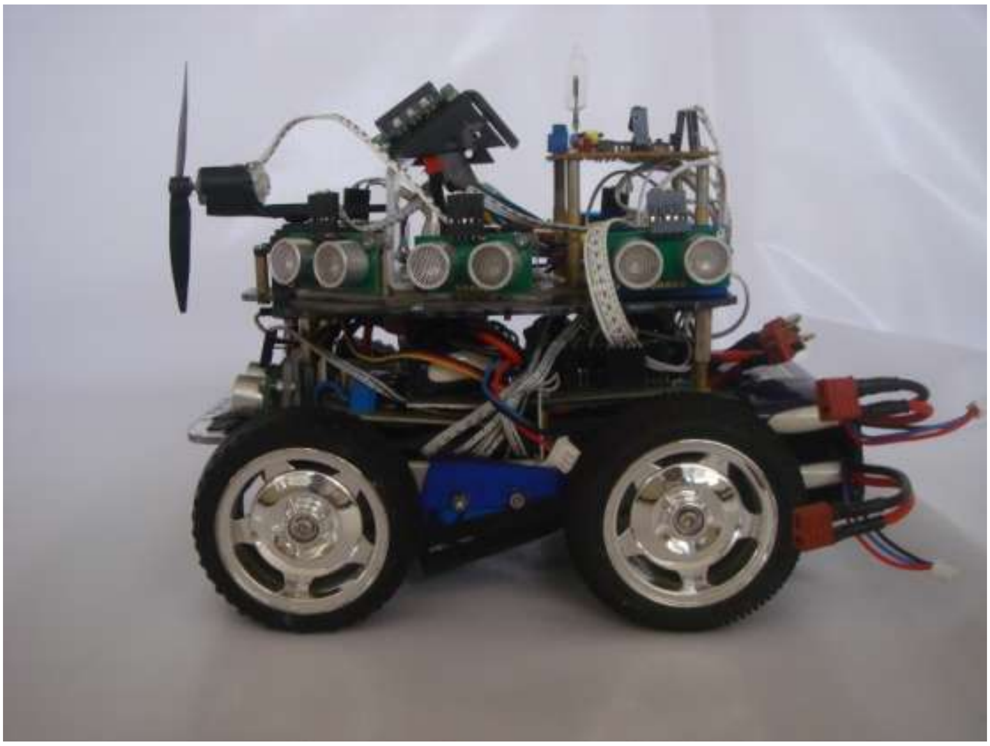
FOTO ROBOT BERODA

TAMPAK DEPAN

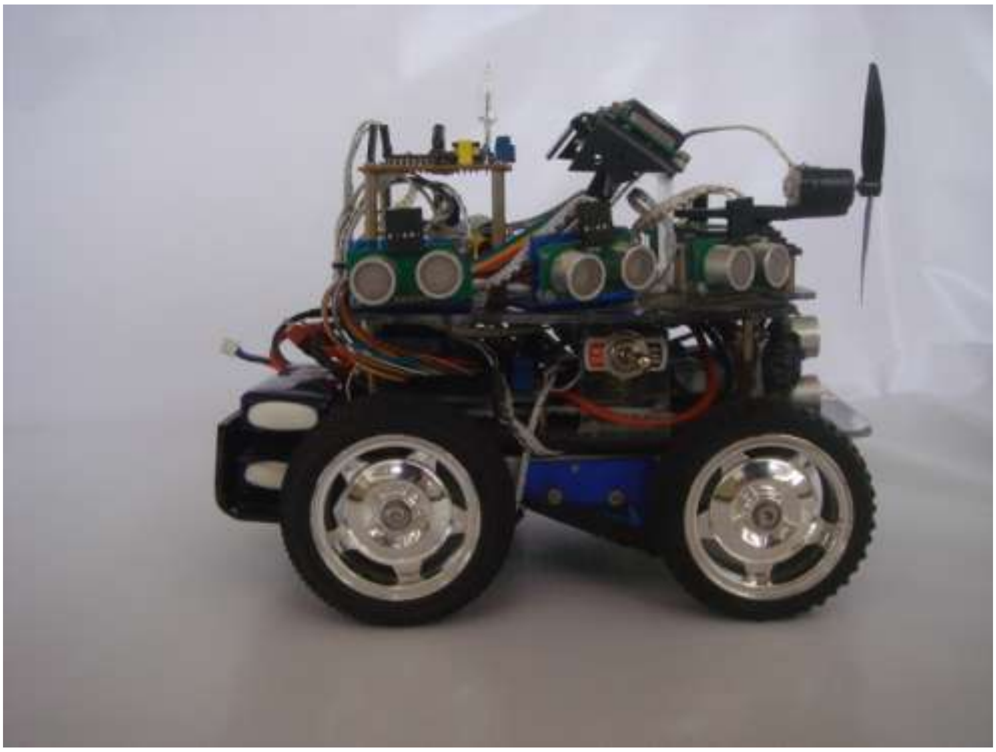


TAMPAK BELAKANG

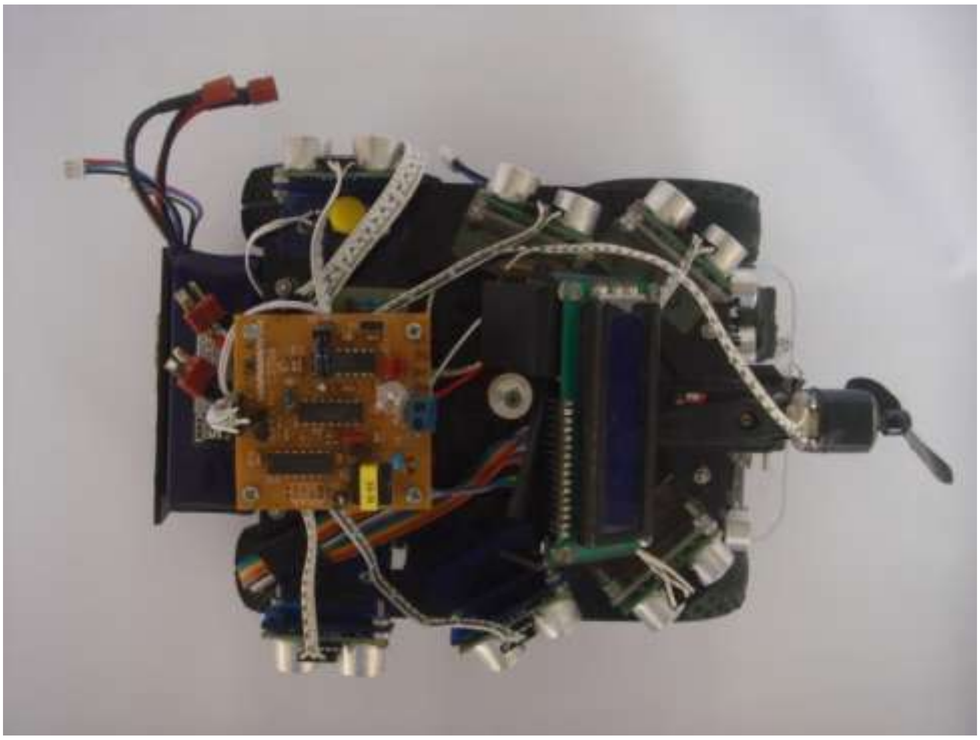
TAMPAK SAMPING KIRI



TAMPAK SAMPING KANAN



TAMPAK ATAS



LAMPIRAN B

PROGRAM PADA PENGONTROL MIKRO

ATMEGA 128

PROGRAM UTAMA

```
/*****************************************************
This program was produced by the
CodeWizardAVR V1.25.3 Standard
Automatic Program Generator
© Copyright 1998-2007 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 03/05/2011
Author  : F4CG
Company : F4CG
Comments:


Chip type           : ATmega128
Program type        : Application
Clock frequency     : 11,059200 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 1024
*****************************************************/

#include <mega128.h>
#include <delay.h>
#include <stdio.h>
// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x15 ;PORTC
#endasm
#include <lcd.h>

#define ADC_VREF_TYPE 0x00

// Read the AD conversion result
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
```

```
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}

// Declare your global variables here


unsigned                                                                                    int
a,b,c,d,e,f,g,i,ic,j,k,l,lz,m,n,o,p,z,x,xa,xb,xc,xd,xe,xf,xg,r,h,counter,adc,adc_1,s1,s2,sensor_7,c1,bm,GakBisa
Keluar,counter_GakBisaKeluar,GakBisaKeluarDeui;
unsigned char text[32];

//depan kiri depan kanan
//xa,xb,xc
void aktif()
{
ulang:
if(PINE.0==0){
 goto lanjut;
}
if(PINA.0==1 & PINA.1==0 & PINA.2==0)
{
while(PINA.0==1 & PINA.1==0 & PINA.2==0){delay_us(2);}
}
else
{
goto ulang;
}

if(PINA.0==0 & PINA.1==1 & PINA.2==0)
{
while(PINA.0==0 & PINA.1==1 & PINA.2==0) {delay_us(2);}
        if(PINA.0==1 & PINA.1==1 & PINA.2==0)
        {
       // while(PINA.0==1 & PINA.1==1 & PINA.2==0) {delay_us(2);}
              //  if(PINA.0==0 & PINA.1==0 & PINA.2==1)
                {
               // while(PINA.0==0 & PINA.1==0 & PINA.2==1) {delay_us(2);}
                goto lanjut;
                }
                //else
               // {
```

```
		//goto ulang;
		//}
	}
	else
	{
	goto ulang;
	}
}
else
{
goto ulang;
}
lanjut:
}

void servo_on(){
	 for(k=60;k>0;k--)
	{
		for(j=0;j<3;j++)//lama pengulangan silkus on yg sama
		{
		PORTE.4=1;
		delay_us(1400);
		for(i=0;i<k;i++){delay_us(10);};
		PORTE.4=0;
		delay_us(18300);
		}
	}
	 for(k=0;k<60;k++)
	{
		for(j=0;j<3;j++)//lama pengulangan silkus on yg sama
		{
		PORTE.4=1;
		delay_us(1400);
		for(i=0;i<k;i++){delay_us(10);};
		PORTE.4=0;
		delay_us(18300);
		}
	}

}
void set_servo(){
		PORTE.4=1;
		delay_us(1780);
```

```
        PORTE.4=0;
        delay_us(18020);
}

void maju(){
 PORTA.3=0;     //kiri
 PORTA.4=0;
 PORTA.5=1;     //kanan
 PORTA.6=0;

}
void kanan(){
 PORTA.3=0;
 PORTA.4=0;
 PORTA.5=0;
 PORTA.6=0;
}

void kiri(){
 PORTA.3=1;
 PORTA.4=0;
 PORTA.5=1;
 PORTA.6=0;
}
void kanan_doank(){
 PORTA.3=0;
 PORTA.4=0;
 PORTA.5=1;
 PORTA.6=1;
}
void kiri_doank(){
 PORTA.3=1;
 PORTA.4=1;
 PORTA.5=1;
 PORTA.6=0;
}

void mundur(){
 PORTA.3=1;
 PORTA.4=0;
 PORTA.5=0;
 PORTA.6=0;
}
```

```
void brenti(){
 PORTA.3=1;
 PORTA.4=1;
 PORTA.5=1;
 PORTA.6=1;
}
void kiri_mundur(){

PORTA.3=1;
 PORTA.4=0;
 PORTA.5=1;
 PORTA.6=1;
}
void kanan_mundur(){

PORTA.3=1;
 PORTA.4=1;
 PORTA.5=0;
 PORTA.6=0;
}

void sensor0(){

    a=0;
    DDRD.0=1;
    PORTD.0=1;
    delay_us(15);
    DDRD.0=0;
    PORTD.0=0;
    delay_us(750);
    while(PIND.0==0)
    {
    delay_us(1);
    }
    while(PIND.0==1)
    {
    a++;
    delay_us(1);
    }
    x=(a/29.034);

}
```

```
void sensor1(){

    b=0;
    DDRD.1=1;
    PORTD.1=1;
    delay_us(15);
    DDRD.1=0;
    PORTD.1=0;
    delay_us(750);
    while(PIND.1==0)
    {
    delay_us(1);
    }
    while(PIND.1==1)
    {
    b++;
    delay_us(1);
    }
    xa=(b/29.034);

}

void sensor2(){

    c=0;
    DDRD.2=1;
    PORTD.2=1;
    delay_us(15);
    DDRD.2=0;
    PORTD.2=0;
    delay_us(750);
    while(PIND.2==0)
    {
    delay_us(1);
    }
    while(PIND.2==1)
    {
    c++;
    delay_us(1);
    }
    xb=(c/29.034);
```

```
}

void sensor3(){

    d=0;
    DDRD.3=1;
    PORTD.3=1;
    delay_us(15);
    DDRD.3=0;
    PORTD.3=0;
    delay_us(750);
    while(PIND.3==0)
    {
    delay_us(1);
    }
    while(PIND.3==1)
    {
    d++;
    delay_us(1);
    }
    xc=(d/29.034);

}

void sensor4(){

    e=0;
    DDRD.4=1;
    PORTD.4=1;
    delay_us(15);
    DDRD.4=0;
    PORTD.4=0;
    delay_us(750);
    while(PIND.4==0)
    {
    delay_us(1);
    }
    while(PIND.4==1)
    {
    e++;
    delay_us(1);
    }
    xd=(e/29.034);
```

```
}

void sensor5(){

    f=0;
    DDRD.5=1;
    PORTD.5=1;
    delay_us(15);
    DDRD.5=0;
    PORTD.5=0;
    delay_us(750);
    while(PIND.5==0)
    {
    delay_us(1);
    }
    while(PIND.5==1)
    {
    f++;
    delay_us(1);
    }
    xe=(f/29.034);

}

void sensor6(){

    g=0;
    DDRD.6=1;
    PORTD.6=1;
    delay_us(15);
    DDRD.6=0;
    PORTD.6=0;
    delay_us(750);
    while(PIND.6==0)
    {
    delay_us(1);
    }
    while(PIND.6==1)
    {
    g++;
    delay_us(1);
    }
```

```
      xf=(g/29.034);

}
void sensor7(){

      sensor_7=0;
      DDRD.7=1;
      PORTD.7=1;
      delay_us(15);
      DDRD.7=0;
      PORTD.7=0;
      delay_us(750);
      while(PIND.7==0)
      {
      delay_us(1);
      }
      while(PIND.7==1)
      {
      sensor_7++;
      delay_us(1);
      }
      xg=(sensor_7/29.034);

}
void wallfollow_kiri()
{
if(x>6)
{
                  if(xd>25 && xb>27)
                  {
                   kiri();
                        OCR1A=40;
                        OCR1B=200;
                        delay_ms(100);
                        }
                        else if(xc<13)
                        {
                        kiri_doank();
                        OCR1A=150;
                        OCR1B=150;
                        }
                        else if(xb<8)
                        {
```

```
                kanan_doank();
                OCR1A=100;
                OCR1B=100;
                }

                else
                {
                maju();
                OCR1A=80;
                OCR1B=80;
                }
            }
            else
            {
            kanan();
            OCR1A=220;
            OCR1B=220;
             }
  }

void ikut_kanan(){
     while (1)
     {

      start_2:

      sensor0();          // x
      sensor1();          // xa
      sensor2();          // xb
      sensor3();          // xc
      sensor5();          // xe

      if(xb==10){o=210;}
      else if(xb==9){o=200;}
      else if(xb>=10){o=180;}
      else if(xc==7){o=210;}
      else if(xc==8){o=200;}
      else if(xc>=9){o=180;}

      if(PINE.3==1) //Uv tron off
      {
      GakBisaKeluar=0;
      bm=0;
```

```
PORTB.7=0;
    if(x>10)                          //wall kanan
    {
        if(xe>10 && xc>18)
        {
        kanan();
        OCR1A=255;
        OCR1B=40;
        delay_ms(100);
        }
        else if(xb<10)
        {
        kanan_doank();
        OCR1A=o;
        OCR1B=o;
        }
        else if(xc<8)
        {
        kiri_doank();
        OCR1A=o;
        OCR1B=o;
        }
        else if(xa<=4){
        kiri();
        OCR1A=100;
        OCR1B=100;
        }

        else
        {
        maju();
        OCR1A=165;
        OCR1B=165;
        }
    }
    else
    {
    kiri();
    OCR1A=210;
    OCR1B=210;
    }
    if(k==2 && xc>25 && xe>25)
        {
```

```
                h=1;
                }

            if(read_adc(1)<150 && read_adc(0)<70 && k==2)       //balik home   ITEM
                {
                r=1;
                }



            if(h==1 && r==1 && read_adc(1)>=350 && read_adc(0)>=300 && k==2)      //home???
                {
                brenti();
                OCR1A=255;
                OCR1B=255;
                //PORTB.7=0;
                lcd_putsf("aing home iyeu\n");
                delay_ms(5000);
                }
           if(read_adc(0)<=160  &&  read_adc(0)>=100  &&  read_adc(1)<=250  &&  read_adc(1)>=150  &&
h==1){         // ABU
           break;
           }
            if(read_adc(1)>=300)                          // PUTIH
            {
                s1=1;
            }

            if(s1==1 && read_adc(1)<150 && k!=2){                        // ITEM
                counter_GakBisaKeluar = counter_GakBisaKeluar+1;

                s1 = 0;
            }
            if(k==2){
                counter_GakBisaKeluar=0;
            }
             if(counter_GakBisaKeluar>=1)                       //lorong
                 {
                     adc=read_adc(0);
                     adc_1=read_adc(1);
                     sensor7();
                     sensor6();
                     if((adc<=150 && adc>=100) && (adc_1<=250 && adc_1>=150)) {              //blkg &&
dpn    // abu
```

```
                    if(xg<=10 && xg>=5){

                        if(xf<=10 && xf>=5){
                            c1=1;
                            delay_ms(5);
                            }
                        else goto start_2;
                        }

                    else goto start_2;
            }
            else goto start_2;
        }
if(counter_GakBisaKeluar>=1 && c1==1 && GakBisaKeluar>=3)
{
    adc=read_adc(0);
    adc_1=read_adc(1);
    sensor7();
    sensor6();
    if((adc<=150 && adc>=100) && (adc_1<=250 && adc_1>=150)) {

                if(xg<=10 && xg>=5){

                    if(xf<=10 && xf>=5){

                        GakBisaKeluar=0;
                        break;
                        c1=0;
                        }
                    else goto start_2;
                    }

                else goto start_2;
    }
    else goto start_2;
}
GakBisaKeluarDeui=0;
while(PINE.6==0){
    GakBisaKeluar=0;
    GakBisaKeluarDeui=GakBisaKeluarDeui+1;
    lcd_gotoxy(0,0);
    lcd_clear();
```

```
        sprintf(text,"%d",GakBisaKeluarDeui);
        lcd_puts(text);
        delay_ms(500);
        while(GakBisaKeluarDeui>=3){
        mundur();
        OCR1A=255;
        OCR1B=255;
        delay_ms(1000);
        break;
        GakBisaKeluarDeui=0;
        }
    }

    if(PINE.2==0){
    kiri_mundur();
    OCR1A=200;
    OCR1B=200;
    }
    if(PINE.5==0){
    kanan_mundur();
    OCR1A=200;
    OCR1B=200;
    }
}
else   //ada api
{
    if(read_adc(1)<350)                // ABU PALING BESAR, putih paling kcl
    {
        if(x>10)                       //wall kanan
        {
            if(xe>10 && xc>18)
            {
            kanan();
            OCR1A=255;
            OCR1B=40;
            delay_ms(100);
            }
            else if(xb<10)
            {
            kanan_doank();
            OCR1A=o;
            OCR1B=o;
            }
```

```
        else if(xc<8)
        {
        kiri_doank();
        OCR1A=o;
        OCR1B=o;
        }
        else if(xa<=4){
        kiri();
        OCR1A=100;
        OCR1B=100;
        }

        else
        {
        maju();
        OCR1A=150;
        OCR1B=150;
        }
    }
    else
    {
    kiri();
    OCR1A=210;
    OCR1B=210;
    }
    if(PINE.2==0){
    kiri_mundur();
    OCR1A=200;
    OCR1B=200;
    }
    if(PINE.5==0){
    kanan_mundur();
    OCR1A=200;
    OCR1B=200;
    }
}
else
{
maju();
OCR1A=255;
OCR1B=255;
k=1;
delay_ms(100);
```

```
            }
            if(read_adc(1)>350 && k==1)       //msh putih
            {
            brenti();
            OCR1A=255;
            OCR1B=255;
            PORTB.7=1;
            servo_on();
            k=2;
            bm=1;
            }
            if(read_adc(1)>350 && k==1 && bm==2)          //
            {
            brenti();
            OCR1A=255;
            OCR1B=255;
            PORTB.7=1;
            servo_on();
            k=2;
            bm=1;
            }
            if(bm==1){
             kiri_mundur();
             OCR1A=130;
             OCR1B=130;
             delay_ms(250);
             bm=2;
            }

      }

   };

}
void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=Out Func5=Out Func4=Out Func3=Out Func2=In Func1=In Func0=In
// State7=T State6=0 State5=0 State4=0 State3=0 State2=T State1=T State0=T
PORTA=0x00;
```

```
DDRA=0x78;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=0 State6=0 State5=0 State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0xE0;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x00;

// Port E initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=P State5=T State4=T State3=T State2=T State1=T State0=P
PORTE=0x41;
DDRE=0x00;

// Port F initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTF=0x00;
DDRF=0x00;

// Port G initialization
// Func4=In Func3=In Func2=In Func1=In Func0=In
// State4=T State3=T State2=T State1=T State0=T
PORTG=0x00;
DDRG=0x00;


// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
```

```
// OC0 output: Disconnected
ASSR=0x00;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 10.800 kHz
// Mode: Ph. correct PWM top=00FFh
// OC1A output: Non-Inv.
// OC1B output: Non-Inv.
// OC1C output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR1A=0xA1;
TCCR1B=0x05;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
OCR1CH=0x00;
OCR1CL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// Timer/Counter 3 initialization
```

```
// Clock source: System Clock
// Clock value: Timer3 Stopped
// Mode: Normal top=FFFFh
// OC3A output: Discon.
// OC3B output: Discon.
// OC3C output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer3 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR3A=0x00;
TCCR3B=0x00;
TCNT3H=0x00;
TCNT3L=0x00;
ICR3H=0x00;
ICR3L=0x00;
OCR3AH=0x00;
OCR3AL=0x00;
OCR3BH=0x00;
OCR3BL=0x00;
OCR3CH=0x00;
OCR3CL=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
// INT3: Off
// INT4: Off
// INT5: Off
// INT6: Off
// INT7: Off
EICRA=0x00;
EICRB=0x00;
EIMSK=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;
ETIMSK=0x00;
```

```
// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;


// ADC initialization
// ADC Clock frequency: 691,200 kHz
// ADC Voltage Reference: AREF pin
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x84;

// LCD module initialization
lcd_init(16);

// aktif();
p=0;        //cek lorong      (ga pake ping>>
n=0;        //ada d lorong
m=0;                //var belum dapat posisi
k=0;
l=0;      //posisi island
z=0;      //api sudah padam cari home
counter=0;



//OCR1A = motor kiri
//b= kanan
while (1)
      {     sensor0();        //x    depan
            sensor2();        //xb
            sensor3();        //xc   kanan
            sensor4();        //xd
            sensor5();        //xe   kanan
            sensor6();        //xf
            sensor7();        //xg   kanan

             if(m==0)                      //cari kiri
            {
            kiri();
            OCR1A=150;
            OCR1B=150;
```

```
        if (xb<10 && xd<9 && xf<8)
        {k=1;}                  //sudah dapat tembok kiri

        if (k==1)
        { brenti();
          OCR1A=100;
          OCR1B=100;
          delay_ms(1000);
          if (x>15 && xg>25)              //cari posisi
           {lz=0;                         //ini sebagai variable posisi robot di lorong
          n=0;                            //ini sebagai variable posisi home non arbitary
           k=0;                           //pengecekan posisi dihentikan
           m=1;
           }
          else
           {lz=1;                         //ini sebagai variable posisi home dan posisi robot......
          n=1;
          k=0;                            //pengecekan posisi dihentikan
           m=1;}}
             }                  //a1
}


        if (m==1)                                   ///sudah dapat posisi
        {                                 //ini program navigasinya.
      If(l==0)
{                           //step 2      loop counter 3
      if(x>12)                  //wall follower kiri
              {
                   if(xd>25 && xb>27)
                   {
                   kiri();
                   OCR1A=100;
                   OCR1B=180;
                   delay_ms(100);
                   }
                   else if(xc<13)
                   {
                   kiri_doank();
                   OCR1A=180;
                   OCR1B=180;
                   }
```

```
                else if(xb<10)  //asli 8
                {
                kanan_doank();
                OCR1A=180;
                OCR1B=180;
                }

                else
                {
                maju();
                OCR1A=150;
                OCR1B=150;
                }
        }
        else
        {
        kanan();
        OCR1A=180;
        OCR1B=180;
         }


if(read_adc(1)>500 && read_adc(0)<=300)
  {s1=1;
  };



if(s1==1)                   //depan sudah pintu putih.
{if(read_adc(0)>500 && read_adc(1)<=300)
   {
   brenti();
   OCR1A=200;
   OCR1B=200;
   delay_ms(400);
   counter=counter+1;
  p=1;                      //cek ruang
  }



if(p==1)
 {
if(PINE.3==1&&lz==0)       //tidak ada api dan posisi di lorong
{mundur();                 //balik lorong()
```

```
OCR1A=120;
OCR1B=120;
delay_ms(600);
kanan();
OCR1A=200;
OCR1B=200;
delay_ms(900);
maju();
OCR1A=150;
OCR1B=150;
delay_ms(900);
p=0;        //cek ruang selesai
s1=0;       //cek pintu selesai
}
else if(PINE.3==0&&lz==0)          //ada api dan posisi di lorong
{maju();                    //masuk ruang
OCR1A=130;
OCR1B=130;
delay_ms(250);
brenti();
OCR1A=130;
OCR1B=130;
  Counter=0;
  z=1;
p=0;
s1=0;
lz=1;

}
else if(lz==1)          //keluar ruang
{maju();
OCR1A=130;
OCR1B=130;
delay_ms(500);
p=0;
s1=0;
lz=0;
};
}                        //b2
```

```
 if (counter>=3)
   {
      if (xf<13 && xg<10)                          //pindah island
             {if (read_adc(1)<300 && read_adc(0)<300 && read_adc(1)>200 &&read_adc(0)>200)
                   {l=1;}           //harus pindah island
             }
     }
}

     }

     if (l==1)
       {

      if(x>8)                  //wall follower kanan
        {
             if(xe>25 && xc>27)
             {
             kanan();
             OCR1A=180;
             OCR1B=100;
             delay_ms(100);
             }
             else if(xc<10)
             {
             kiri_doank();
             OCR1A=180;
             OCR1B=180;
             }
             else if(xb<13)  //asli 8
             {
             kanan_doank();
             OCR1A=180;
             OCR1B=180;
             }

             else
             {
             maju();
             OCR1A=150;
             OCR1B=150;
             }
        }
```

```
        else
        {
        kiri();
        OCR1A=180;
        OCR1B=180;
         }

                if(read_adc(1)>500 && read_adc(0)<=300)
 {s1=1;
 };


if(s1==1)                 //depan sudah pintu putih.
{if(read_adc(0)>500 && read_adc(1)<=300)
  {
  brenti();
  OCR1A=200;
  OCR1B=200;
  delay_ms(400);
  counter=counter+1;
 p=1;                     //cek ruang
  }


if(p==1)
 {
if(PINE.3==1&&lz==0)      //tidak ada api dan posisi di lorong
{mundur();                //balik lorong()
OCR1A=120;
OCR1B=120;
delay_ms(600);
kanan();
OCR1A=200;
OCR1B=200;
delay_ms(900);
maju();
OCR1A=150;
OCR1B=150;
delay_ms(900);
p=0;        //cek ruang selesai
s1=0;       //cek pintu selesai
}
else if(PINE.3==0&&lz==0)            //ada api dan posisi di lorong
```

```
{maju();                    //masuk ruang
OCR1A=130;
OCR1B=130;
delay_ms(250);
brenti();
OCR1A=130;
OCR1B=130;
  z=1;
p=0;
s1=0;
lz=1;

}
else if(lz==1)          //keluar ruang
{maju();
OCR1A=130;
OCR1B=130;
delay_ms(500);
p=0;
s1=0;
counter=0;
lz=0;
};
}                        //b4




        };
    }
If(z==1)
{                   if(x>12)
                    {
                    if(xd>25 && xb>27)
                    {
                    kiri();
                    OCR1A=100;
                    OCR1B=180;
                    delay_ms(100);
                    }
                    else if(xc<13)
                    {
```

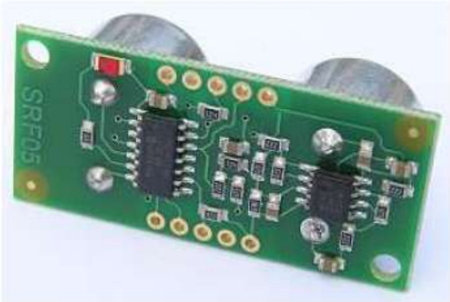
```
                kiri_doank();
                OCR1A=180;
                OCR1B=180;
                }
                else if(xb<10)  //asli 8
                {
                kanan_doank();
                OCR1A=180;
                OCR1B=180;
                }

                else
                {
                maju();
                OCR1A=150;
                OCR1B=150;
                }
            }
            else
            {
            kanan();
            OCR1A=180;
            OCR1B=180;
             }
if(n==0)
{          if(s2==0)
           {


       if(read_adc(1)>500 && read_adc(0)<=300)
        {s1=1;
        };



       if(s1==1)
       {if(read_adc(0)>500 && read_adc(1)<=300)
         {  mundur();                          //balik lorong()
       OCR1A=120;
       OCR1B=120;
       delay_ms(600);
       kanan();
       OCR1A=200;
       OCR1B=200;
```

```
            delay_ms(900);
            maju();
            OCR1A=150;
            OCR1B=150;
            delay_ms(900);
             s1=0; }              }
             else
{
            if(read_adc(1)>500 && read_adc(0)<=300)
             {s1=1;
             };


            if(s1==1)
            {if(read_adc(0)>500 && read_adc(1)<=300)
              {  maju();                       //keluar ke lorong()
            OCR1A=120;
            OCR1B=120;
            delay_ms(200);
             s2=0; }}
             };
     }
};


           };
           }
```

LAMPIRAN C
DATASHEET

SRF05 - Ultra-Sonic Ranger

Technical Specification

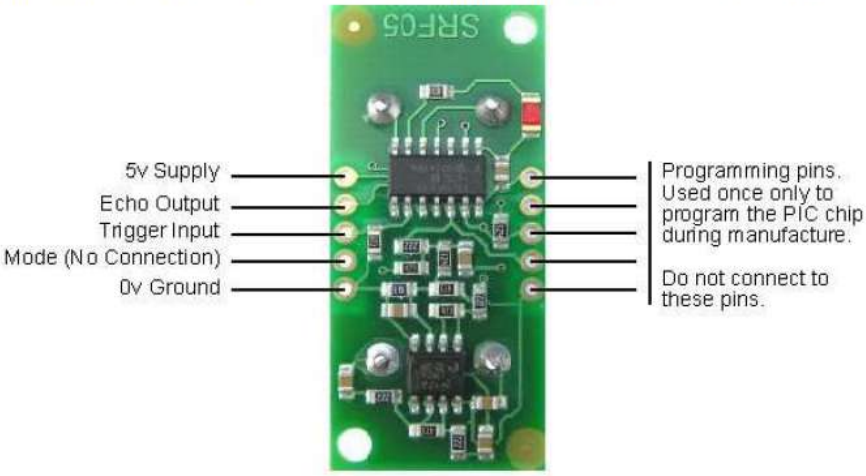


Introduction

The SRF05 is an evolutionary step from the SRF04, and has been designed to increase flexibility, increase range, and to reduce costs still further. As such, the SRF05 is fully compatible with the SRF04. Range is increased from 3 meters to 4 meters. A new operating mode (tying the mode pin to ground) allows the SRF05 to use a single pin for both trigger and echo, thereby saving valuable pins on your controller. When the mode pin is left unconnected, the SRF05 operates with separate trigger and echo pins, like the SRF04. The SRF05 includes a small delay before the echo pulse to give slower controllers such as the Basic Stamp and Picaxe time to execute their pulse in commands.

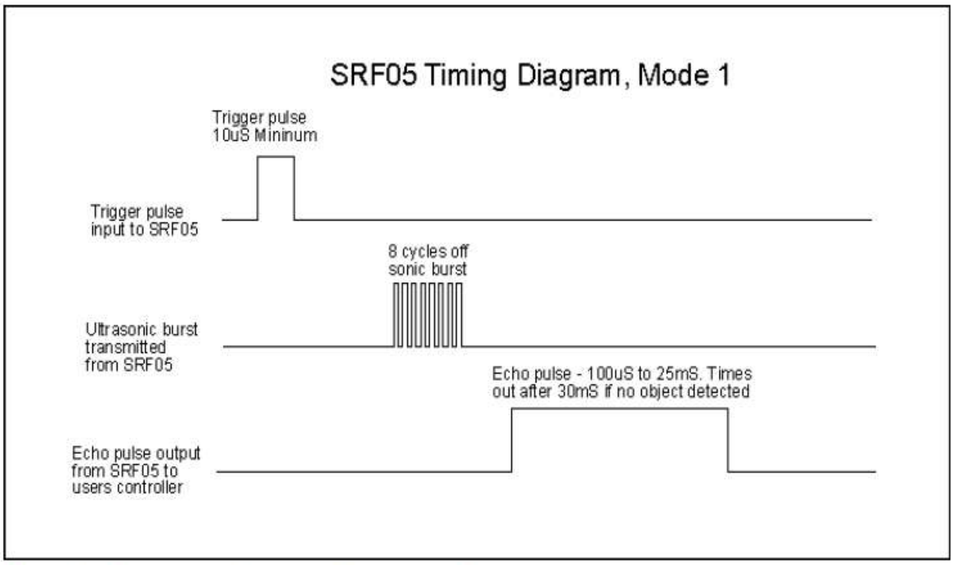
Mode 1 - SRF04 compatible - Separate Trigger and Echo

This mode uses separate trigger and echo pins, and is the simplest mode to use. All code examples for the SRF04 will work for the SRF05 in this mode. To use this mode, just leave the mode pin unconnected - the SRF05 has an internal pull up resistor on this pin.

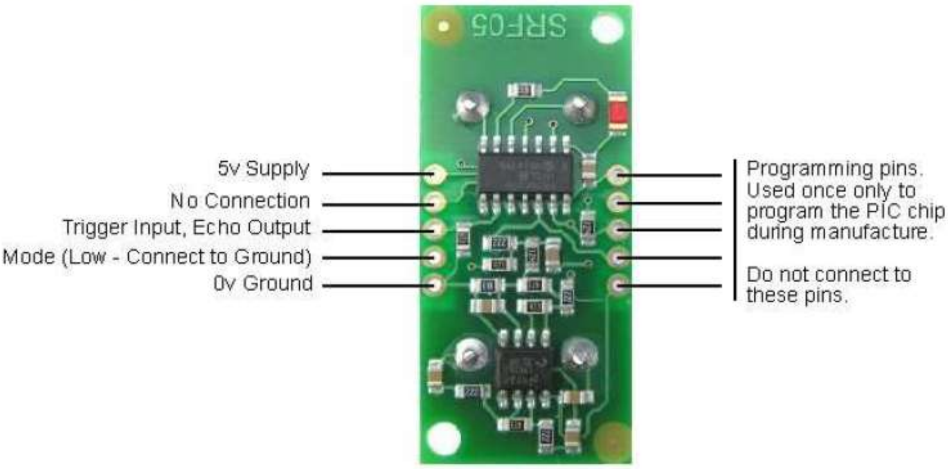


Connections for 2-pin Trigger/Echo Mode (SRF04 compatible)

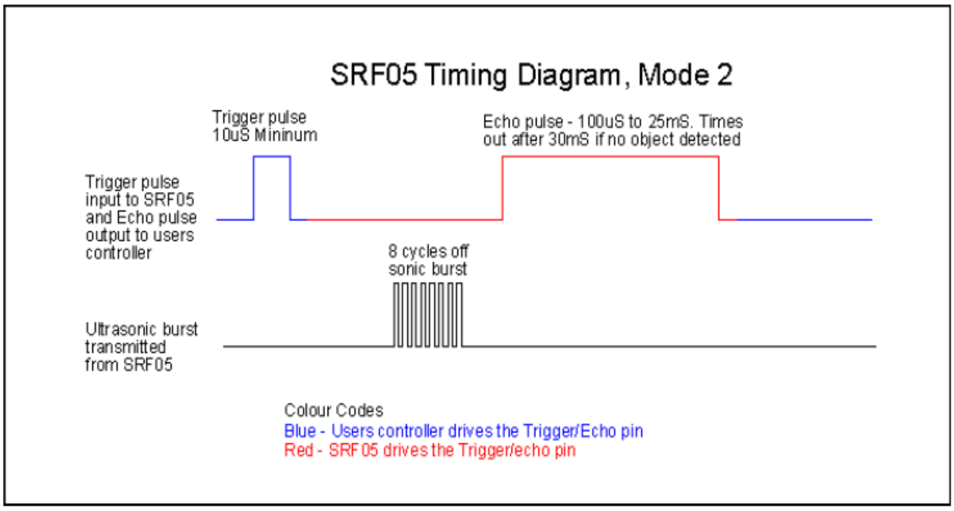
Page 1



Mode 2 - Single pin for both Trigger and Echo
This mode uses a single pin for both Trigger and Echo signals, and is designed to save valuable pins on embedded controllers. To use this mode, connect the mode pin to the 0v Ground pin. The echo signal will appear on the same pin as the trigger signal. The SRF05 will not raise the echo line until 700uS after the end of the trigger signal. You have that long to turn the trigger pin around and make it an input and to have your pulse measuring code ready. The PULSIN command found on many popular controllers does this automatically.



Connections for single pin Trigger/Echo Mode



To use mode 2 with the Basic Stamp BS2, you simply use PULSOUT and PULSIN on the same pin, like this:

```
SRF05 PIN 15               ' use any pin for both trigger and echo
Range VAR Word             ' define the 16 bit range variable

SRF05 = 0                  ' start with pin low
PULSOUT SRF05, 5           ' issue 10uS trigger pulse (5 x 2uS)
PULSIN SRF05, 1, Range     ' measure echo time
Range = Range/29           ' convert to cm (divide by 74 for inches)
```

Calculating the Distance
The SRF05 Timing diagrams are shown above for each mode. You only need to supply a short 10uS pulse to the trigger input to start the ranging. The SRF05 will send out an 8 cycle burst of ultrasound at 40khz and raise its echo line high (or trigger line in mode 2). It then listens for an echo, and as soon as it detects one it lowers the echo line again. The echo line is therefore a pulse whose width is proportional to the distance to the object. By timing the pulse it is possible to calculate the range in inches/centimeters or anything else. If nothing is detected then the SRF05 will lower its echo line anyway after about 30mS.

The SRF04 provides an echo pulse proportional to distance. If the width of the pulse is measured in uS, then dividing by 58 will give you the distance in cm, or dividing by 148 will give the distance in inches. uS/58=cm or uS/148=inches.

The SRF05 can be triggered as fast as every 50mS, or 20 times each second. You should wait 50ms before the next trigger, even if the SRF05 detects a close object and the echo pulse is shorter. This is to ensure the ultrasonic "beep" has faded away and will not cause a false echo on the next ranging.

The other set of 5 pins
The 5 pins marked "programming pins" are used once only during manufacture to program the Flash memory on the PIC16F630 chip. The PIC16F630's programming pins are also used for other functions on the SRF05, so make sure you don't connect anything to these pins, or you will disrupt the modules operation.

HAMAMATSU

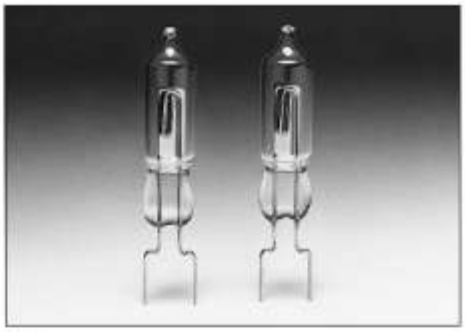
FLAME SENSOR
UV TRON® R2868

Quick Detection of Flame from Distance, Compact UV Sensor with High Sensitivity and Wide Directivity, Suitable for Flame Detectors and Fire Alarms.

Hamamatsu R2868 is a UV TRON ultraviolet detector that makes use of the photoelectric effect of metal and the gas multiplication effect. It has a narrow spectral sensitivity of 185 to 260 nm, being completely insensitive to visible light. Unlike semiconductor detectors, it does not require optical visible-cut filters, thus making it easy to use.

In spite of its small size, the R2868 has wide angular sensitivity (directivity) and can reliably and quickly detect weak ultraviolet radiations emitted from flame due to use of the metal plate cathode (eg. it can detect the flame of a cigarette lighter at a distance of more than 5 m.).

The R2868 is well suited for use in flame detectors and fire alarms, and also in detection of invisible discharge phenomena such as corona discharge of high-voltage transmission lines.



APPLICATIONS

- Flame detectors for gas/oil lighters and matches
- Fire alarms
- Combustion monitors for burners
- Inspection of ultraviolet leakage
- Detection of discharge
- Ultraviolet switching

GENERAL

Parameters	Rating	Units
Spectral Response	185 to 260	nm
Window Material	UV glass	—
Weight	Approx. 1.5	g
Dimensional Outline	See Fig. 3	—

MAXIMUM RATINGS

Parameters	Rating	Units
Supply Voltage	400	Vdc
Peak Current [1]	30	mA
Average Discharge Current [2]	1	mA
Operating Temperature	-20 to +60	°C

CHARACTERISTICS (at 25°C)

Parameters	Rating	Units
Discharge Starting Voltage (with UV radiation)	280	Vdc Max.
Recommended Operating Voltage	325±25	Vdc
Recommended Average Discharge Current	100	μA
Background [3]	10	cpm Max.
Sensitivity [4]	5000	cpm Typ.

NOTES:

1) This is the maximum momentary current that can be handled if its full width at half maximum is less than 10 μs.
2) If the tube is operated near this or higher, the service life is noticeably reduced. Use the tube within the recommended current values.
3) Measured under room illuminations (approximately 500 lux) and recommended operating conditions. Note that these values may increase if the following environmental factors are present.
   1. Mercury lamps, sterilization lamps, or halogen lamps are located nearby.
   2. Direct or reflected sunlight is incident on the tube.
   3. Electrical sparks such as welding sparks are present.
   4. Radiation sources are present.
   5. High electric field (including static field) generates across the tube.
4) These are representative values for a wavelength of 200 nm and a light input of 10 pW/cm². In actual use, the sensitivity will vary with the wavelength of the ultraviolet radiation and the drive circuitry employed.

Figure 1: UV TRON's Spectral Response and Various Light Sources

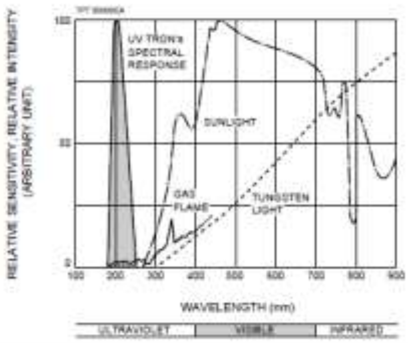


Figure 2: Angular Sensitivity (Directivity)

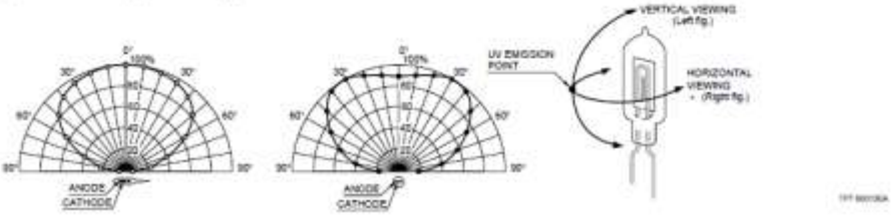


Figure 3: Dimensional Outline (Unit: mm)

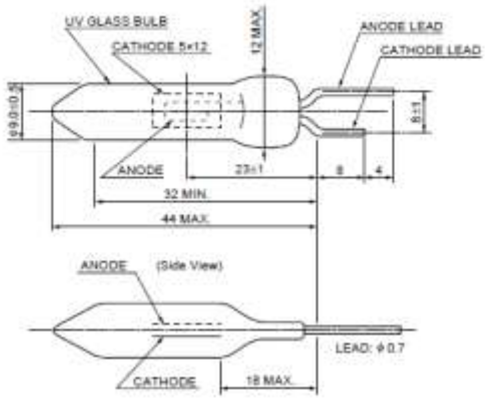
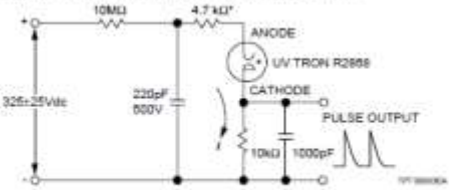


Figure 4: Recommended Operating Circuit



* Be sure to connect the 4.7 kΩ resistor within 2.5 cm from the anode lead end of UV TRON.

- UV TRON Driving Circuit C3704 series (Option)

Hamamatsu also provide the driving circuit C3704 series for R2868 operation.
C3704 series include a high voltage power supply and a signal processing circuit in printed circuit board, which allows to operate R2868 easily as a flame sensor with the low input voltage (DC 6 to 30 V) only.
For the details, please refer to the datasheet of C3704 series.

PRECAUTIONS FOR USE

• Ultraviolet Radiation

The UV TRON itself emits ultraviolet radiation in operation. When using two or more UV TRONs at the same time in close position, care should be taken so that they do not optically interfere with each other.

• Vibration and Shock

The UV TRON is designed in accordance with the standards of MIL-STD-202F (Method 204D/0.06 inch or 10g, 10- 500Hz, 15 minutes, 1 cycle) and MIL-STD-202F (Method 213B/100g, 11ms, Half-sine, 3 times). However, should a strong shock be sustained by the UV TRON (e.g. if dropped), the glass bulb may crack or the internal electrode may be deformed, resulting in deterioration of electrical characteristics. So extreme care should be taken in handling the tube.

• Polarity

Connect the UV TRON with correct polarity. Should it be connected with reverse polarity, operating errors may occur.

WARRANTY.

The UV TRON is covered by a warranty for a period of one year after delivery. The warranty is limited to replacement of any defective tube due to defects traceable to the manufacturer.

HAMAMATSU

HAMAMATSU PHOTONICS K.K., Electron Tube Center
314-5, Shimokanzo, Toyooka-village, Iwata-gun, Shizuoka-ken, 438-0193, Japan, Telephone: (81)539/62-5248, Fax: (81)539/62-2205,
U.S.A.: Hamamatsu Corporation: 360 Foothill Road, Bridgewater, N.J. 08807-0910, U.S.A., Telephone: (1)908-231-0960, Fax: (1)908-231-1218
Germany: Hamamatsu Photonics Deutschland GmbH: Arzbergerstr. 10, D-82211 Herrsching am Ammersee, Germany, Telephone: (49)8152-375-0, Fax: (49)8152-2658
France: Hamamatsu Photonics France S.A.R.L.: 8, Rue du Saule Trapu, Parc du Moulin de Massy, 91882 Massy Cedex, France, Telephone: (33)1 69 53 71 00, Fax: (33)1 69 53 71 10
United Kingdom: Hamamatsu Photonics UK Limited: Lough Point, 2 Gladbeck Way, Windmill Hill, Enfield, Middlesex EN2 7JA, United Kingdom, Telephone: (44)181-367-3560, Fax: (44)181-367-6384
North Europe: Hamamatsu Photonics Norden AB: Färögatan 7, S-164-40 Kista, Sweden, Telephone: (46)8-703-29-50, Fax: (46)8-750-58-95
Italy: Hamamatsu Photonics Italia S.R.L.: Via Della Moia, 1/E, 20020 Arese, (Milano), Italy, Telephone: (39)2-935 81 733, Fax: (39)2-935 81 741

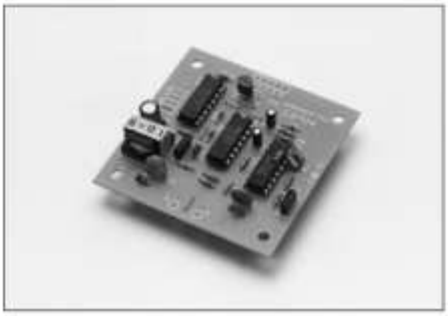
TPT 1008E01
MAR.1998 CR
Created in Japan

HAMAMATSU

UV TRON® DRIVING CIRCUIT
C3704 SERIES

Compact, Lightweight, Low Current Consumption, Low Cost
Operates as High Sensitivity UV Sensor with UV TRON
Suitable for Flame Detectors and Fire Alarms

Hamamatsu C3704 series UV TRON driving circuits are low current consuming, signal processing circuits for the UV TRON, well known as a high sensitivity ultraviolet detecting tube. The C3704 series can be operated as a UV sensor by connecting the UV TRON and applying DC low voltage, as they have both a high-voltage power supply and a signal processing circuit on the same printed circuit board.
Since background discharges of the UV TRON caused by natural excitation lights (such as a cosmic ray, scattered sunlight, etc.) can be cancelled in the signal processing circuit, the output signals from the C3704 series can be used without errors.
When the high sensitivity sensor "UV TRON R2868" (sold separately) is used, the flame from a cigarette lighter (flame length: 25mm) can be detected even from a distance of more than 5m.



APPLICATIONS

- **Flame detectors for gas and oil lighters**
- **Fire alarms**
- **Combustion monitors for burners**
- **Electric spark detector**
- **UV photoelectric counter**

SPECIFICATIONS

Dimensional outline .... Figure 1
Weight .... Approx. 20g
Output signal .... Open collector Output (50 V, 100 mA Max.)
10 ms width pulse output (Note : 1)
UV TRON supply voltage .... DC 350 V (Note : 2)
Quenching time .... Approx. 50 ms
Operating temperature .... -10 to +50°C
(with no condensation)
Suitable UV TRON .... Low voltage operation UV TRON
(such as R2868)

	C3704	C3704-02	C3704-03
Input Voltage	10 to 30 Vdc	5Vdc ± 5%	6 to 9 Vdc
Current consumption	3 mA Max.	300µA Max.	300µA Max.

Note 1: The output pulse width can be extended up to about 100s by adding a capacitor to the circuit board.
Note 2: Since the output impedance of this power supply is extremely high, an ordinary voltmeter cannot be used. Use a voltmeter that has an input impedance of more than 10 GΩ.

Figure 1: Dimensional Outline (Unit : mm)

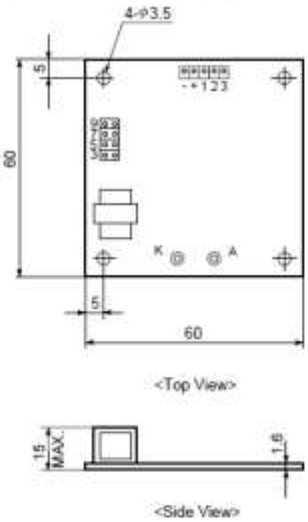


Figure 2: Schematic Diagram

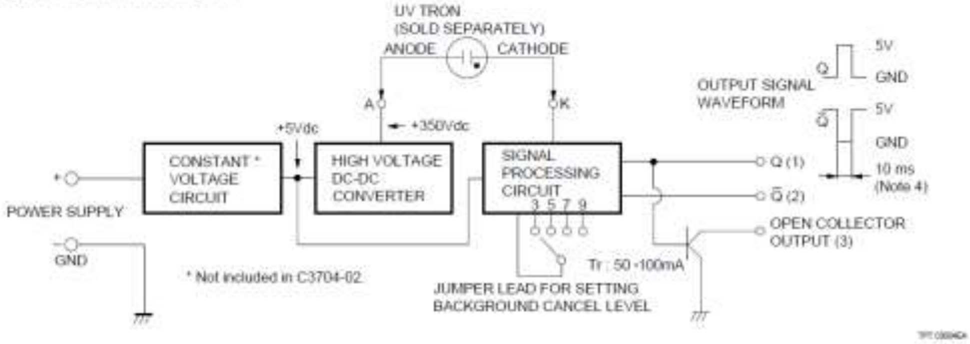
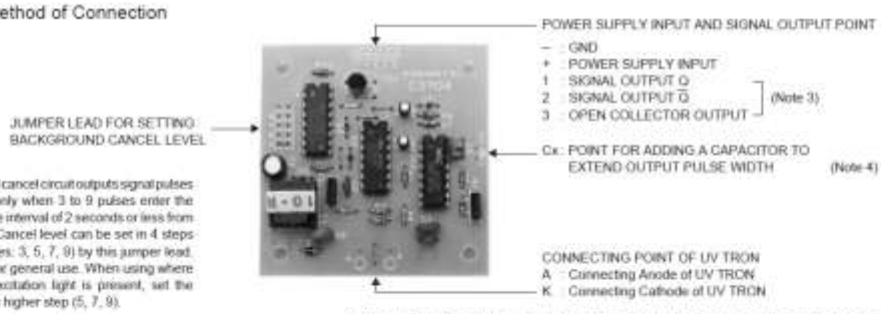


Figure 3: Method of Connection



This background cancel circuit outputs signal pulses of 10ms width only when 3 to 9 pulses enter the circuit with a time interval of 2 seconds or less from the UV TRON. Cancel level can be set in 4 steps (number of pulses: 3, 5, 7, 9) by this jumper lead. Connect to "3" for general use. When using where much natural excitation light is present, set the cancel level at a higher step (5, 7, 9).

Note 3: No load can be driven by an output from points "1" and "2" because these signals are output from the only C-MOS IC directly. When a load such as a buzzer and a relay is connected to this circuit, it should be connected to the point open collector output. The transistor ratings of the open collector is 50V, 100mA. Be careful not to exceed the ratings.

Note 4: The output pulse width is set to 10ms at shipping. If the pulse width needs to be extended, add a capacitor to this point. (When using an electrolytic condenser, make sure the polarity is correct.)
e.g. CX = 1 µF: Pulse Width ≒ 1s, CX=10 µF : Pulse Width ≒ 10s

PRECAUTIONS FOR USE

- Since the operation impedance is extremely high, the UV TRON should be connected as close as possible to the circuit board within 5 cm.
- Take care to avoid external noise since a C-MOS IC is used in the circuit. It is recommended that the whole PC board be put in the shield box when it is used.
- To reduce current consumption, oscillating frequency is very low (approx. 20 Hz) in this DC-DC converter. Thus, the output impedance of the high voltage power supply is extremely high. If the surrounding humidity is high, electrical leakage on the PC board surface may lead to a drop in the supply voltage to the UV TRON. This voltage drop may result in lowered detection performance, so a moistureproof material (silicone compound, etc.) should be applied at the connecting point of the UV TRON, etc., if using the unit in a humid environment.

- A model equipped with a flame sensor (R2868) is also available.

HAMAMATSU

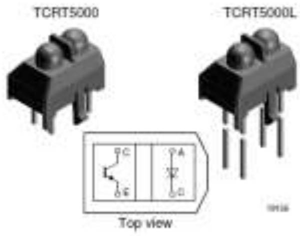
HAMAMATSU PHOTONICS K.K., Electron Tube Center
314-5, Shimokanzo, Toyooka-village, Iwata-gun, Shizuoka-ken, 438-0193, Japan, Telephone: (81)539/62-5248, Fax: (81)539/62-2205, Telex: 4225-186HAMAHQ
U.S.A.: Hamamatsu Corporation: 360 Foothill Road, Bridgewater, N.J. 08807-0910, U.S.A., Telephone: (1)908-231-0960, Fax: (1)908-231-1218
Germany: Hamamatsu Photonics Deutschland GmbH: Arzbergerstr. 10, D-82211 Herrsching am Ammersee, Germany, Telephone: (49)8152-375-0, Fax: (49)8152-2658
France: Hamamatsu Photonics France S.A.R.L.: 8, Rue du Saule Trapu, Parc du Moulin de Massy, 91882 Massy Cedex, France, Telephone: (33)1 69 53 71 00, Fax: (33)1 69 53 71 10
United Kingdom: Hamamatsu Photonics UK Limited: Lough Point, 2 Gladbeck Way, Windmill Hill, Enfield, Middlesex EN2 7JA, United Kingdom, Telephone: (44)181-367-3560, Fax: (44)181-367-6384
North Europe: Hamamatsu Photonics Norden AB: Färögatan 7, S-164-40 Kista, Sweden, Telephone: (46)8-703-29-50, Fax: (46)8-750-58-95
Italy: Hamamatsu Photonics Italia S.R.L.: Via Della Moia, 1/E 20020 Arese, (Milano), Italy, Telephone: (39)2-935 81 733, Fax: (39)2-935 81 741

TPT1007E01
JUL.1997 CR
Created in Japan

Reflective Optical Sensor with Transistor Output

Description

The TCRT5000 and TCRT500L are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light. The package includes two mounting clips. TCRT5000L is the long lead version.



Features

- Package type: Leaded
- Detector type: Phototransistor
- Dimensions: L 10.2 mm x W 5.8 mm x H 7.0 mm
- Peak operating distance: 2.5 mm
- Operating range: 0.2 mm to 15 mm
- Typical output current under test: I_C = 1 mA
- Daylight blocking filter
- Emitter wavelength 950 nm
- Lead (Pb)-free soldering released
- Lead (Pb)-free component in accordance to RoHS 2002/95/EC and WEEE 2002/96/EC

Applications

- Position sensor for shaft encoder
- Detection of reflective material such as paper, IBM cards, magnetic tapes etc.
- Limit switch for mechanical motions in VCR
- General purpose - wherever the space is limited

Order Instructions

Part Number	Remarks	Minimum Order Quantity
TCRT5000	3.5 mm lead length	4500 pcs, 50 pcs/tube
TCRT5000L	15 mm lead length	2400 pcs, 48 pcs/tube

Absolute Maximum Ratings

T_{amb} = 25 °C, unless otherwise specified

Input (Emitter)

Parameter	Test condition	Symbol	Value	Unit
Reverse voltage		V_R	5	V
Forward current		I_F	60	mA
Forward surge current	$t_p \leq 10$ µs	I_{FSM}	3	A
Power dissipation	$T_{amb} \leq 25$ °C	P_V	100	mW
Junction temperature		T_j	100	°C

Output (Detector)

Parameter	Test condition	Symbol	Value	Unit
Collector emitter voltage		V_{CEO}	70	V
Emitter collector voltage		V_{ECO}	5	V
Collector current		I_C	100	mA
Power dissipation	$T_{amb} \leq 55$ °C	P_V	100	mW
Junction temperature		T_j	100	°C

Sensor

Parameter	Test condition	Symbol	Value	Unit
Total power dissipation	$T_{amb} \leq 25$ °C	P_{tot}	200	mW
Operation temperature range		T_{amb}	- 25 to + 85	°C
Storage temperature range		T_{stg}	- 25 to + 100	°C
Soldering temperature	2 mm from case, t ≤ 10 s	T_{sd}	260	°C

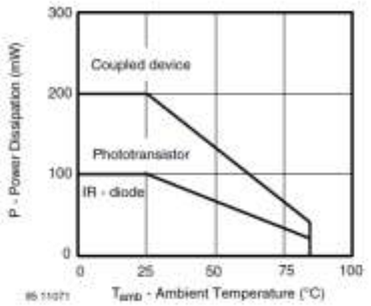


Figure 1. Power Dissipation Limit vs. Ambient Temperature

Electrical Characteristics

T_{amb} = 25 °C, unless otherwise specified

Input (Emitter)

Parameter	Test condition	Symbol	Min	Typ.	Max	Unit
Forward voltage	I_F = 60 mA	V_F		1.25	1.5	V
Junction capacitance	V_R = 0 V, f = 1 MHz	C_j		17		pF
Radiant intensity	I_F = 60 mA, t_p = 20 ms	I_E			21	mW/sr
Peak wavelength	I_F = 100 mA	λ_p	940			nm
Virtual source diameter	Method: 63 % encircled energy	Ø		2.1		mm

Output (Detector)

Parameter	Test condition	Symbol	Min	Typ.	Max	Unit
Collector emitter voltage	I_C = 1 mA	V_{CEO}	70			V
Emitter collector voltage	I_E = 100 μA	V_{ECO}	7			V
Collector dark current	V_{CE} = 20 V, I_F = 0, E = 0	I_{CEO}		10	200	nA

Sensor

Parameter	Test condition	Symbol	Min	Typ.	Max	Unit
Collector current	V_{CE} = 5 V, I_F = 10 mA, D = 12 mm	I_C [1,2]	0.5	1	2.1	mA
Collector emitter saturation voltage	I_F = 10 mA, I_C = 0.1 mA, D = 12 mm	V_{CEsat} [1,2]			0.4	V

[1] See figure 3

[2] Test surface: Mirror (Mfr. Spindler a. Hoyer, Part No 340005)

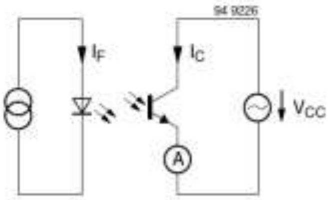


Figure 2. Test Circuit

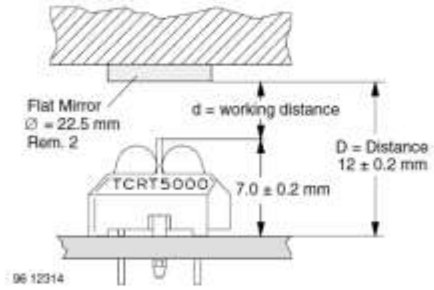


Figure 3. Test Circuit

Typical Characteristics

T_{amb} = 25 °C, unless otherwise specified

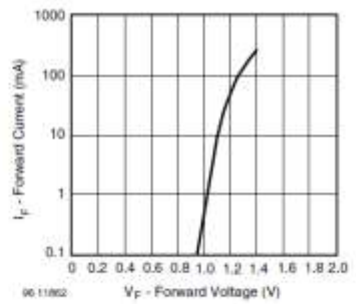


Figure 4. Forward Current vs. Forward Voltage

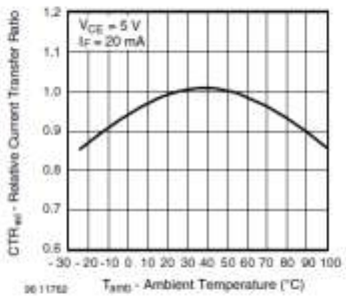


Figure 5. Relative Current Transfer Ratio vs. Ambient Temperature

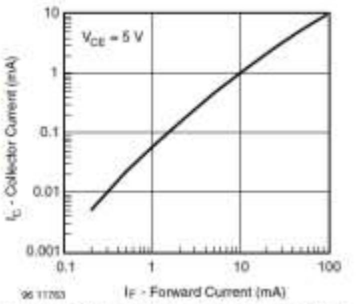


Figure 6. Collector Current vs. Forward Current

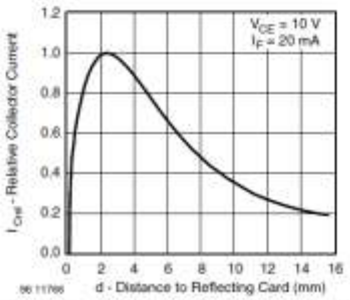


Figure 9. Relative Collector Current vs. Distance

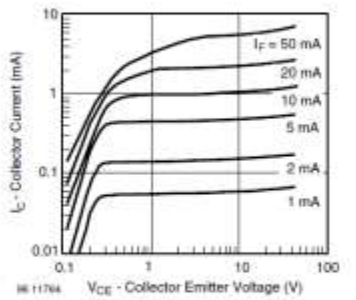


Figure 7. Collector Emitter Saturation Voltage vs. Collector Current

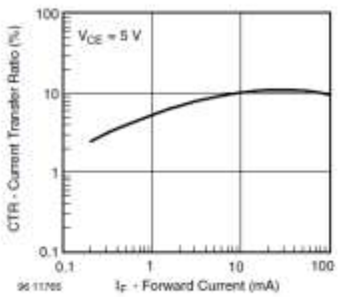
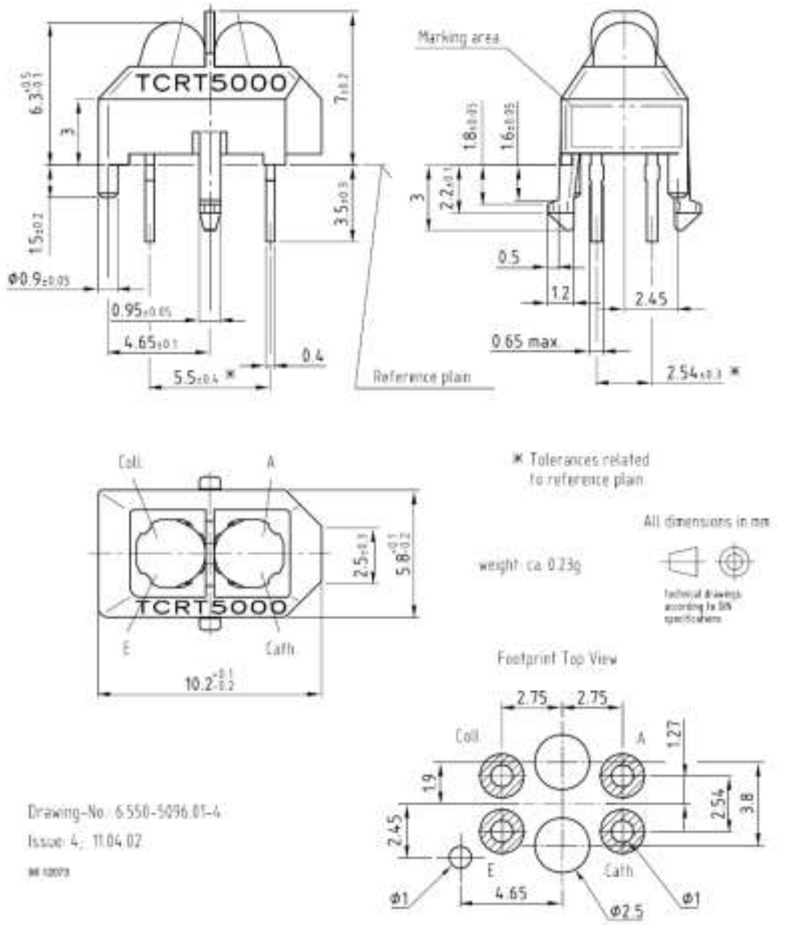
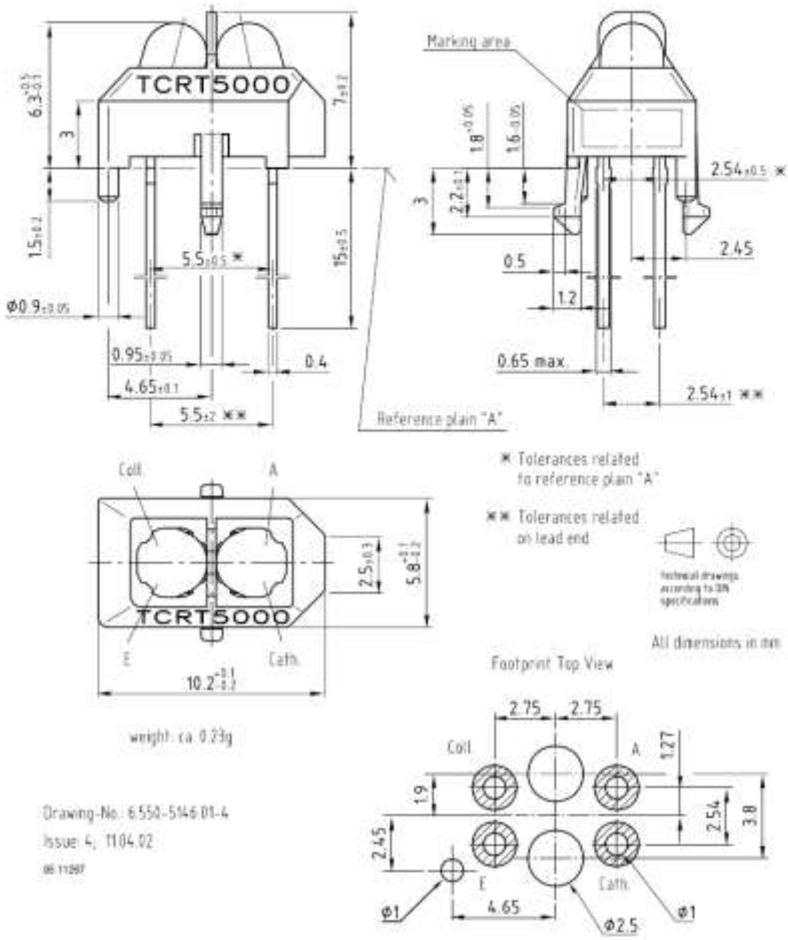


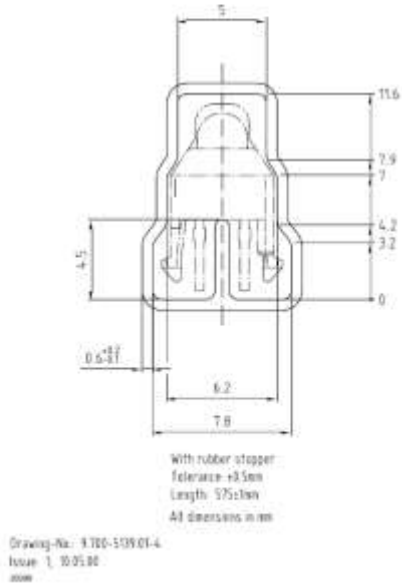
Figure 8. Current Transfer Ratio vs. Forward Current

Package Dimensions in mm




TCRT5000
Marking area
Reference plain "A"
* Tolerances related to reference plain "A"
** Tolerances related on lead end
All dimensions in mm
Footprint Top View
Coll
A
E
Cath
weight: ca. 0.23g
Drawing-No.: 6.550-5146.01-4
Issue: 4; 11.04.02

TCRT5000, Tube Dimensions



TCRT5000L, Tube Dimensions