LAMPIRAN A

List Program Codevision Generator Data

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

Chip type           : ATmega16
Program type        : Application
Clock frequency     : 11.059200 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 256
*****************************************
************/

#include <mega16.h>
#include <delay.h>
#include <scankeypad_b.h>


// Declare your global variables here

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

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

// Port B 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
PORTB=0x00;
DDRB=0x00;

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

// 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;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCC0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

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

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
```

```
MCUCR=0x00;
MCUCSR=0x00;

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

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

while (1)
     {
key:
{
switch (scan_keypad())
   {
   case '1':
   PORTC.6=1;
   delay_ms(1000);
   goto in_1;
   break;
   case '2':
   PORTC.6=1;
   delay_ms(1000);
   goto in_2;
   break;
   case '3':
   PORTC.6=1;
   delay_ms(1000);
   goto in_3;
   break;
   case '4':
   PORTC.5=1;
   delay_ms(1000);
   goto in_4;
   break;
   case '5':
   PORTC.5=1;
   delay_ms(1000);
   goto in_5;
   break;
   case '6':
   PORTC.5=1;
   delay_ms(1000);
   goto in_6;
   break;
   case '7':
   PORTC.4=1;
   delay_ms(1000);
   goto in_7;
   break;
   case '8':
   PORTC.4=1;
   delay_ms(1000);
   goto in_8;
   break;
   case '9':
   PORTC.4=1;
   delay_ms(1000);
   goto in_9;
   break;
   case '*':
   PORTC.3=1;
   delay_ms(1000);
   goto in_A;
   break;
   case '0':
   PORTC.3=1;
   delay_ms(1000);
   goto in_B;
   break;
   case '#':
   PORTC.3=1;
   delay_ms(1000);
   goto in_C;
   break;


   };

 }

     // Place your code here


}
in_1:
{
while(PIND.1==1);

     PORTA=0x0F;
     delay_ms(500);
     PORTC.7=1;
     delay_ms(5500);
     PORTC.7=0;
     delay_ms(1000);

     PORTA=0x01;
     delay_ms(500);
     PORTC.7=1;
     delay_ms(5500);
     PORTC.7=0;
     delay_ms(1000);

     PORTA=0x02;
     delay_ms(500);
     PORTC.7=1;
     delay_ms(5500);
     PORTC.7=0;
```

```
    delay_ms(1000);

    PORTA=0x03;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x04;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x05;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x00;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_us(1000);

    PORTA=0x0E;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);


     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }


in_2:
{
while(PIND.1==1);

    PORTA=0x06;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x07;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x08;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x09;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0A;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0B;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0C;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0D;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
```

```
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }


in_3:
{
while(PIND.1==1);

    PORTA=0x00;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x02;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x04;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x06;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x08;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0A;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0C;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0E;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(5500);
    PORTC.7=0;
    delay_ms(1000);


     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }



in_4:
{
while(PIND.1==1);

    PORTA=0x0F;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x01;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x02;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x03;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
```

```
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x04;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x05;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x00;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0E;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);


     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }


 in_5:
{
while(PIND.1==1);
    PORTA=0x06;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x07;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x08;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x09;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0A;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0B;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0C;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0D;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }
```

```
 in_6:
{
while(PIND.1==1);

    PORTA=0x00;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x02;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x04;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x06;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x08;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0A;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0C;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0E;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(3000);
    PORTC.7=0;
    delay_ms(1000);

     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }

in_7:
{
while(PIND.1==1);

    PORTA=0x0F;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x01;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x02;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x03;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x04;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1200);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x05;
    delay_ms(500);
```

```
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x00;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0E;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);


     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }


in_8:
{
while(PIND.1==1);
    PORTA=0x06;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x07;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x08;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x09;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1200);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0A;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0B;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0C;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0D;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }




in_9:
{
while(PIND.1==1);
    PORTA=0x00;
    delay_ms(500);
```

```
PORTC.7=1;
delay_ms(1000);
PORTC.7=0;
delay_ms(1000);

PORTA=0x02;
delay_ms(500);
PORTC.7=1;
delay_ms(1000);
PORTC.7=0;
delay_ms(1000);

PORTA=0x04;
delay_ms(500);
PORTC.7=1;
delay_ms(1200);
PORTC.7=0;
delay_ms(1000);

PORTA=0x06;
delay_ms(500);
PORTC.7=1;
delay_ms(1000);
PORTC.7=0;
delay_ms(1000);

PORTA=0x08;
delay_ms(500);
PORTC.7=1;
delay_ms(1000);
PORTC.7=0;
delay_ms(1000);

PORTA=0x0A;
delay_ms(500);
PORTC.7=1;
delay_ms(1000);
PORTC.7=0;
delay_ms(1000);

PORTA=0x0C;
delay_ms(500);
PORTC.7=1;
delay_ms(1000);
PORTC.7=0;
delay_ms(1000);

PORTA=0x0E;
delay_ms(500);
PORTC.7=1;
delay_ms(1000);
PORTC.7=0;
delay_ms(1000);

 if (PIND.1==1)
   {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }

in_A:
{
while(PIND.1==1);
    PORTA=0x0F;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x01;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1000);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x02;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x03;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x04;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x05;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x00;
    delay_ms(500);
    PORTC.7=1;
```

```
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0E;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }

in_B:
{
while(PIND.1==1);
    PORTA=0x06;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x07;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x08;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x09;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0A;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0B;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0C;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0D;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }

in_C:
{
while(PIND.1==1);
    PORTA=0x00;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x02;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x04;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
```

```
    delay_ms(1000);

    PORTA=0x06;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x08;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0A;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0C;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

    PORTA=0x0E;
    delay_ms(500);
    PORTC.7=1;
    delay_ms(1400);
    PORTC.7=0;
    delay_ms(1000);

     if (PIND.1==1)
       {PORTA=0x00;
        PORTC.7=0;
        delay_ms(10);
        PORTC=0x00;
        goto key;
        }

        }

}
```

```
char scan_keypad(void)
     {
int scankey;
char keypressed =0;
DDRB = 0x0F;

PORTB = 0xFE;
scankey = PINB&0xf0;
switch (scankey)
   {
    case 0xE0 : keypressed = '1';
    break;
    case 0xD0 : keypressed = '2';
    break;
    case 0xB0 : keypressed = '3';
    break;
    case 0x70 : keypressed = 'A';
    break;
    }
    PORTB = 0xFD;
    delay_ms(10);
scankey = PINB&0xf0;
switch (scankey)
   {
    case 0xE0 : keypressed = '4';
    break;
    case 0xD0 : keypressed = '5';
    break;
    case 0xB0 : keypressed = '6';
    break;
    case 0x70 : keypressed = 'B';
    break;
    }
    PORTB = 0xFB;
    delay_ms(10);
    scankey = PINB&0xf0;
switch (scankey)
   {
    case 0xE0 : keypressed = '7';
    break;
    case 0xD0 : keypressed = '8';
    break;
    case 0xB0 : keypressed = '9';
    break;
    case 0x70 : keypressed = 'C';
    break;
    }
    PORTB = 0xF7;
    delay_ms(10);
    scankey = PINB&0xf0;
switch (scankey)
   {
    case 0xE0 : keypressed = '*';
    break;
    case 0xD0 : keypressed = '0';
    break;
```

```
   case 0xB0 : keypressed = '#';
   break;
   case 0x70 : keypressed = 'D';
   break;
   }
     return keypressed
   }
```

List Program Codevision Multiplexer

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


Chip type           : ATmega16
Program type        : Application
Clock frequency     : 11.059200 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 256
*****************************************
************/

#include <mega16.h>
#include <delay.h>
#include <stdio.h>

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

// Declare your global variables here
bit a,b,c,d;
char temp[4];


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

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

// Port B 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
PORTB=0x00;
DDRB=0x00;

// 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=Out Func6=Out Func5=Out Func4=Out
Func3=In Func2=In Func1=In Func0=In
// State7=0 State6=0 State5=0 State4=0 State3=T
State2=T State1=T State0=T
PORTD=0x00;
DDRD=0xF0;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
```

```
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

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

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

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

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

// LCD module initialization
lcd_init(16);



 while (1)
     {

 keymux:

 {

     if (PIND.0==1)
     {
     delay_ms(500);
     PORTD.4=PIND.0;
     goto mux1;
     }

     if(PIND.1==1)
     {
     delay_ms(250);
     PORTD.5=PIND.1;
     goto mux2;
     }

     if (PIND.2==1)
     {
     delay_ms(100);
     PORTD.6=PIND.2;
     goto mux3;
     }


     if (PIND.3==1)
     {
     delay_ms(55);
     PORTD.7=PIND.3;
     goto mux4;
     }


}


 };

 mux1:
   {

   if(PINB.4==1)
   {

            a=PINB.0;
            delay_ms(10);
            b=PINB.1;
            delay_ms(10);
            c=PINB.2;
            delay_ms(10);
            d=PINB.3;
            delay_ms(10);

            PORTA.0=1;
            delay_ms(200);
            PORTA.0=0;
            delay_ms(200);
            PORTA.0=1;
            delay_ms(200);
            PORTA.0=0;
            delay_ms(200);
            PORTA.0=1;

            delay_ms(1000);
            PORTA.0=a;
            delay_ms(1000);
            PORTA.0=b;
            delay_ms(1000);
```

```
            PORTA.0=c;
            delay_ms(1000);
            PORTA.0=d;
            delay_ms(1000);
            PORTA.0=0;

            lcd_clear();
            sprintf(temp,"%d%d%d%d",d,c,b,a);
            lcd_gotoxy(6,1);
            lcd_puts(temp);
            goto keymux;
     }

     else
     {

     goto keymux;
     };
}

 mux2:
  {

     if(PINB.4==1)
     {

            a=PINB.0;
            delay_ms(10);
            b=PINB.1;
            delay_ms(10);
            c=PINB.2;
            delay_ms(10);
            d=PINB.3;
            delay_ms(10);

            PORTA.0=1;
            delay_ms(200);
            PORTA.0=0;
            delay_ms(200);
            PORTA.0=1;
            delay_ms(200);
            PORTA.0=0;
            delay_ms(200);
            PORTA.0=1;

            delay_ms(500);
            PORTA.0=a;
            delay_ms(500);
            PORTA.0=b;
            delay_ms(500);
            PORTA.0=c;
            delay_ms(500);
            PORTA.0=d;
            delay_ms(500);
            PORTA.0=0;

            lcd_clear();
            sprintf(temp,"%d%d%d%d",d,c,b,a);
            lcd_gotoxy(6,1);
            lcd_puts(temp);
            goto keymux;

        }
        else
        {
        goto keymux;
        };

}

mux3:
 {
     if(PINB.4==1)
     {


            a=PINB.0;
            delay_ms(10);
            b=PINB.1;
            delay_ms(10);
            c=PINB.2;
            delay_ms(10);
            d=PINB.3;
            delay_ms(10);

            PORTA.0=1;
            delay_ms(200);
            PORTA.0=0;
            delay_ms(200);
            PORTA.0=1;
            delay_ms(200);
            PORTA.0=0;
            delay_ms(200);
            PORTA.0=1;

            delay_ms(110);
            PORTA.0=a;
            delay_ms(110);
            PORTA.0=b;
            delay_ms(110);
            PORTA.0=c;
            delay_ms(110);
            PORTA.0=d;
            delay_ms(110);
            PORTA.0=0;

            lcd_clear();
            sprintf(temp,"%d%d%d%d",d,c,b,a);
            lcd_gotoxy(6,1);
            lcd_puts(temp);
            goto keymux;
```

```
    }
    else
    {
    goto keymux;
    };
}
mux4:
 {

     if(PINB.4==1)
     {

            a=PINB.0;
            delay_ms(10);
            b=PINB.1;
            delay_ms(10);
            c=PINB.2;
            delay_ms(10);
            d=PINB.3;
            delay_ms(10);

            PORTA.0=1;
            delay_ms(200);
            PORTA.0=0;
            delay_ms(200);
            PORTA.0=1;
            delay_ms(200);
            PORTA.0=0;
            delay_ms(200);
            PORTA.0=1;

            delay_ms(130);
            PORTA.0=a;
            delay_ms(100);
            PORTA.0=b;
            delay_ms(100);
            PORTA.0=c;
            delay_ms(100);
            PORTA.0=d;
            delay_ms(100);
            PORTA.0=0;

            lcd_clear();
            sprintf(temp,"%d%d%d%d",d,c,b,a);
            lcd_gotoxy(6,1);
            lcd_puts(temp);
            goto keymux;

    }
    else
    {
    goto keymux;
    };
}
   }
```

List Program Codevision Demultiplexer

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

Chip type           : ATmega16
Program type        : Application
Clock frequency     : 11.059200 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 256
*****************************************
************/

#include <mega16.h>
#include <delay.h>
#include <stdio.h>

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

// Declare your global variables here

void main(void)
{
// Declare your local variables here
bit a,b,c,d;
int n;
char demux[4];

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

// Port B 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
PORTB=0x00;
DDRB=0x00;
```

```
// 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;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

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

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

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

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

// LCD module initialization
lcd_init(16);

while (1)
    {

    // Place your code here

    ademux:

     if (PINB.0==1)
     {
      delay_ms(180);
      if (PINB.0==0)
      {
       delay_ms(180);
       if (PINB.0==1)
       {
        delay_ms(180);
        if (PINB.0==0)
        {
         delay_ms(180);
         if (PINB.0==1)
         {

          goto run;
         }
         else
         {
          goto ademux;
         }
        }
        else//4
        {
         goto ademux;
        }
```

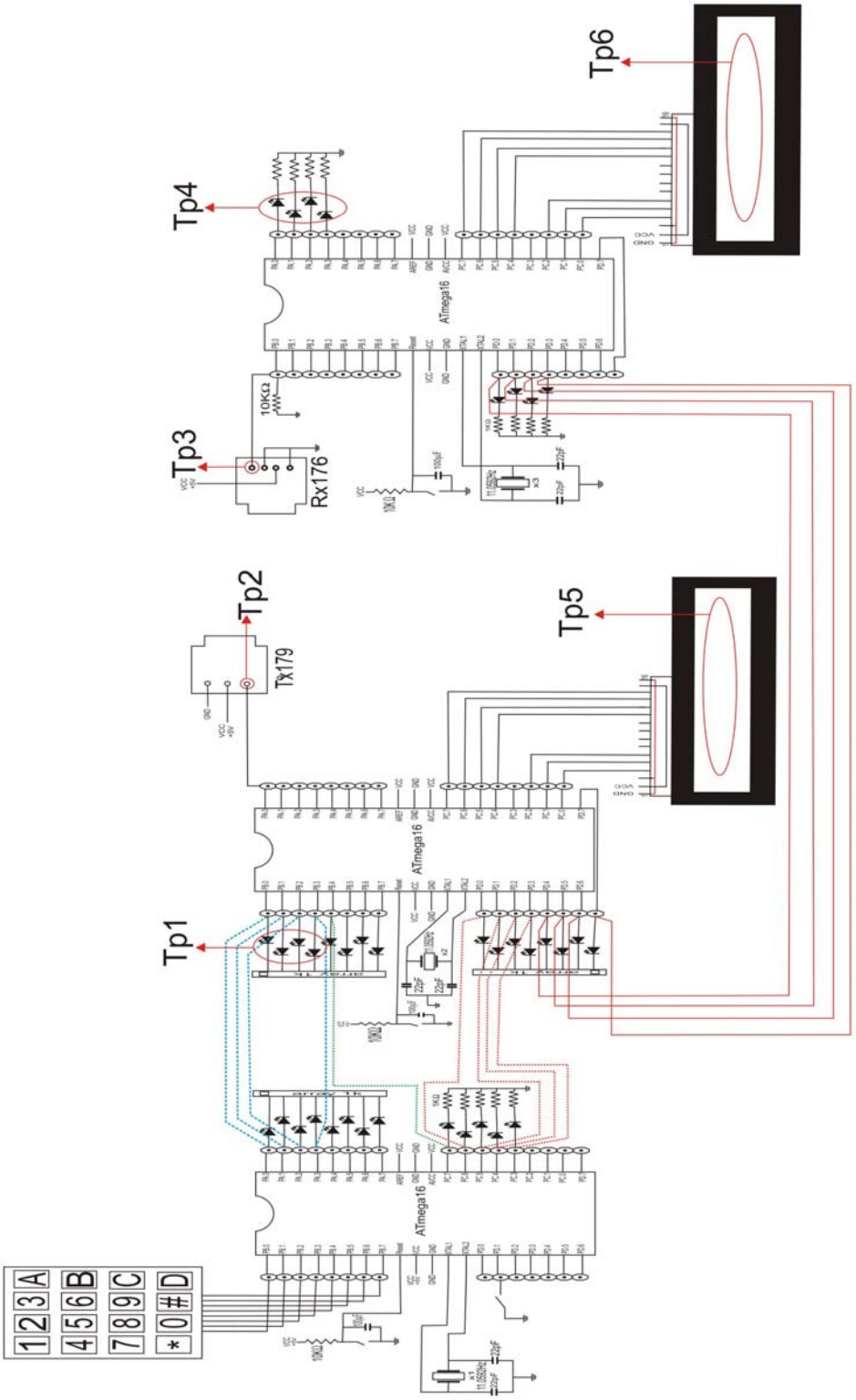
```
          }
          else //3
          {
           goto ademux;
          }
        }
        else //2
        {
        goto ademux;
        }

      }
      else //1
      {
      goto ademux;
      }


run:

if (PIND.0==1)
  {
  goto demux1;
  }

if (PIND.1==1)
  {
  goto demux2;
  }

if (PIND.2==1)
  {
  goto demux3;
  }

 if (PIND.3==1)
  {
  goto demux4;
  }



 };

demux1:
lcd_clear();
 n=1;
       while(1)
       {
         delay_ms(1200);
         if (n==1)
         {
         PORTA.0=PINB.0;
         a=PORTA.0;
         }
         else
          {
            if (n==2)
          {
            PORTA.1=PINB.0;
            b=PORTA.1;
          }
           else
            {
              if(n==3)
            {
              PORTA.2=PINB.0;
              c=PORTA.2;
             }
             else
                {
                  if(n==4)
                {
                  PORTA.3=PINB.0;
                  d=PORTA.3;
                }

           }; //akhir if=3
          }; //akhir if=2
        };  //akhir if=1


         n=n+1;

        if (n>4)
        {
        n=0;
        delay_ms(5);
        PORTA=0x00;
        delay_ms(5);

        sprintf(demux,"%d%d%d%d",d,c,b,a);
        lcd_gotoxy(6,1);
        lcd_puts(demux);
        goto ademux;
        }

           }


demux2:
lcd_clear();
n=1;
       while(1)
       {
         delay_ms(490);
         if (n==1)
         {
         PORTA.0=PINB.0;
         a=PORTA.0;
         }
```

```
        else
         {
           if (n==2)
         {
           PORTA.1=PINB.0;
           b=PORTA.1;
         }
          else
           {
            if(n==3)
           {
            PORTA.2=PINB.0;
            c=PORTA.2;
           }
           else
              {
               if(n==4)
              {
               PORTA.3=PINB.0;
               d=PORTA.3;
              }

          }; //akhir if=3
         }; //akhir if=2
       };  //akhir if=1


        n=n+1;

       if (n>4)
       {
       n=0;
       delay_ms(5);
       PORTA=0x00;
       delay_ms(5);
       lcd_clear();
       sprintf(demux,"%d%d%d%d",d,c,b,a);
       lcd_gotoxy(6,1);
       lcd_puts(demux);
       goto ademux;

       }

          }


demux3:
lcd_clear();
n=1;
      while(1)
      {
       delay_ms(95);
       if (n==1)
       {
       PORTA.0=PINB.0;
       a=PORTA.0;
```

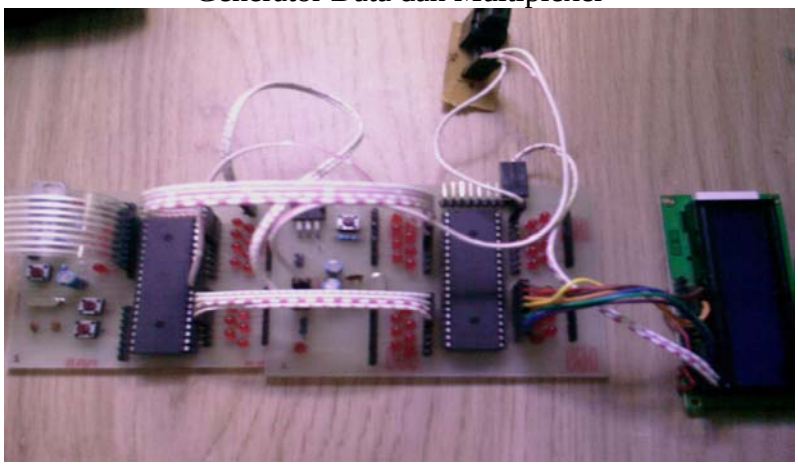
```
       }
        else
         {
           if (n==2)
         {
           PORTA.1=PINB.0;
           b=PORTA.1;
         }
          else
           {
            if(n==3)
           {
            PORTA.2=PINB.0;
            c=PORTA.2;
           }
           else
              {
               if(n==4)
              {
               PORTA.3=PINB.0;
               d=PORTA.3;
              }

          }; //akhir if=3
         }; //akhir if=2
       };  //akhir if=1


        n=n+1;

       if (n>4)
       {
       n=0;
       delay_ms(5);
       PORTA=0x00;
       delay_ms(5);
       lcd_clear();
       sprintf(demux,"%d%d%d%d",d,c,b,a);
       lcd_gotoxy(6,1);
       lcd_puts(demux);
       goto ademux;

       }

          }


demux4:
lcd_clear();
n=1;
      while(1)
      {

       delay_ms(85);
       if (n==1)
       {
       PORTA.0=PINB.0;
```

```
        a=PORTA.0;
        }
          else
            {
              if (n==2)
            {
              PORTA.1=PINB.0;
              b=PORTA.1;
            }
             else
               {
                if(n==3)
               {
                PORTA.2=PINB.0;
                c=PORTA.2;
                }
                else
                    {
                     if(n==4)
                    {
                     PORTA.3=PINB.0;
                     d=PORTA.3;
                    }

             }; //akhir if=3
            }; //akhir if=2
          };  //akhir if=1

          n=n+1;

        if (n>4)
        {
        n=0;
        delay_ms(5);
        PORTA=0x00;
        delay_ms(5);
        lcd_clear();
        sprintf(demux,"%d%d%d%d",d,c,b,a);
        lcd_gotoxy(6,1);
        lcd_puts(demux);

        goto ademux;

        }

            }
}
```

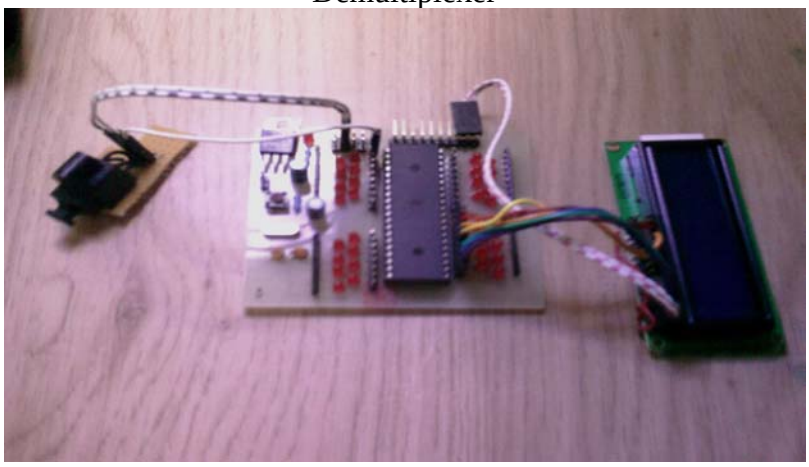
LAMPIRAN B


Tp6
Tp4
Tp3
Tp2
Tp5
Tp1
Rx176
Tx179
ATmega16
10KΩ
1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D

Generator Data dan Multiplexer



Demultiplexer

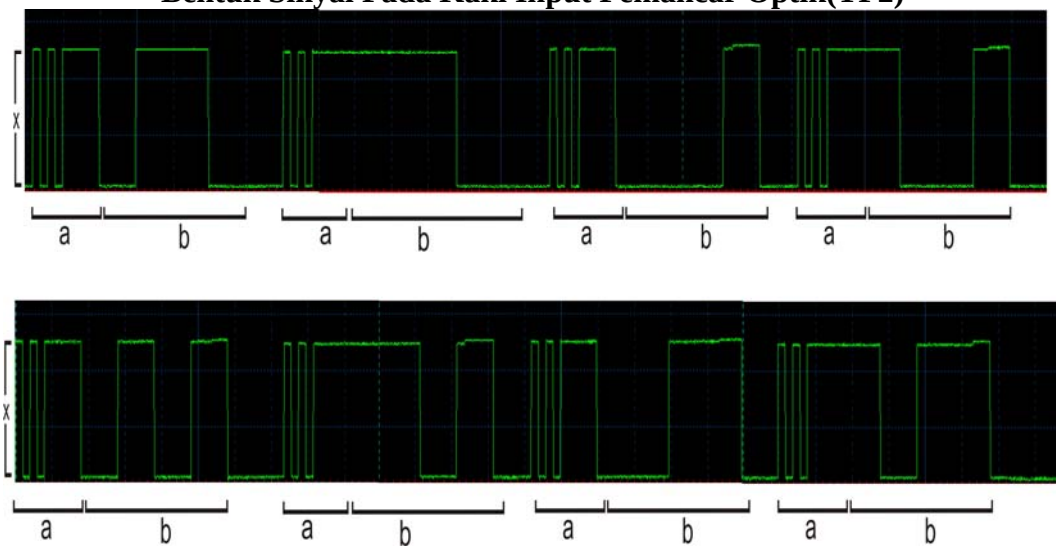


Kabel Fiber Optik

LAMPIRAN C

Data yang dikirim adalah 0x06h, 0x07h, 0x08h , 0x09h , 0x0Ah , 0x0Bh , 0x0Ch , 0x0Dh

Bentuk Sinyal Pada Kaki Input Pemancar Optik(TP2)

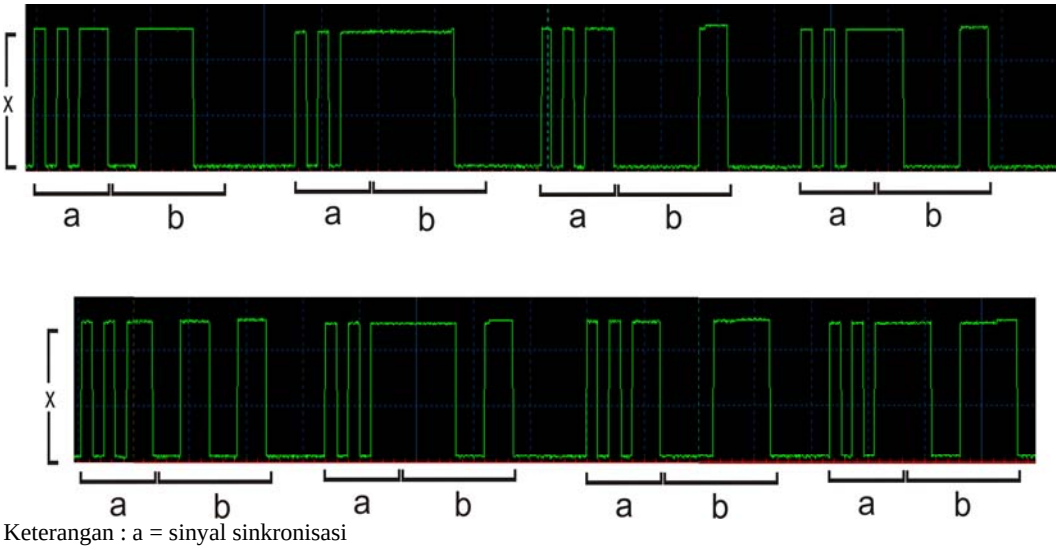


Keterangan : a = sinyal sinkronisasi

b = data

x = 4 V

Kecepatan Pengiriman 1bit per detik

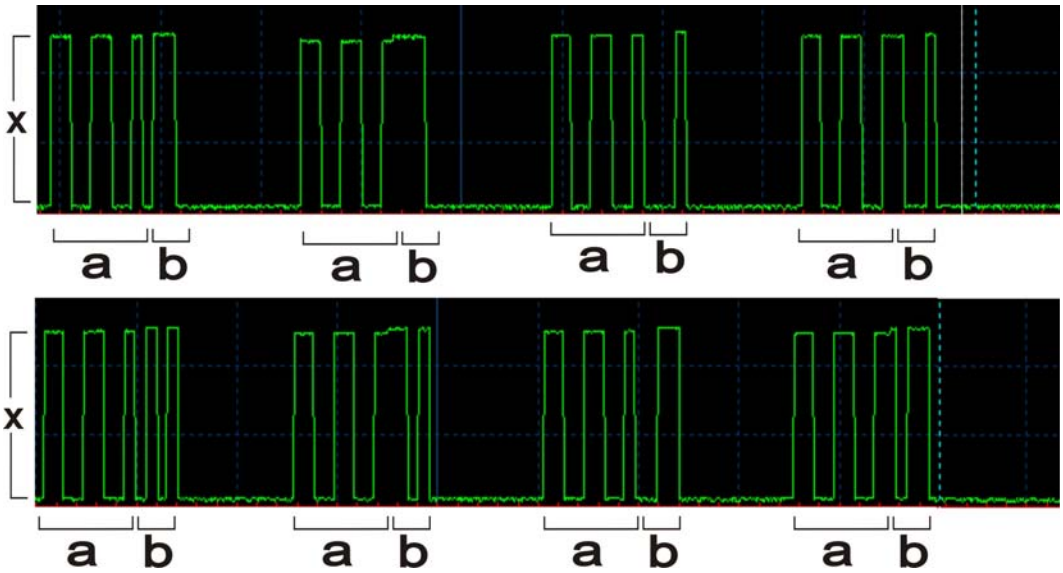


Keterangan : a = sinyal sinkronisasi

b = data

x = 4 V

Kecepatan Pengiriman 2 bit per detik

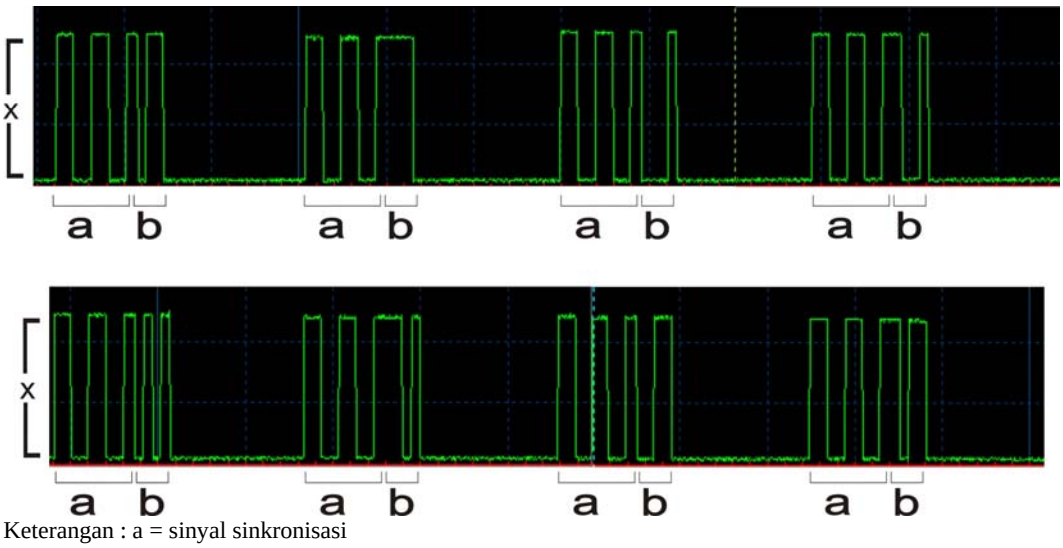


Keterangan : a = sinyal sinkronisasi

b = data

x = 4 V

Kecepatan Pengiriman 9 bit per detik



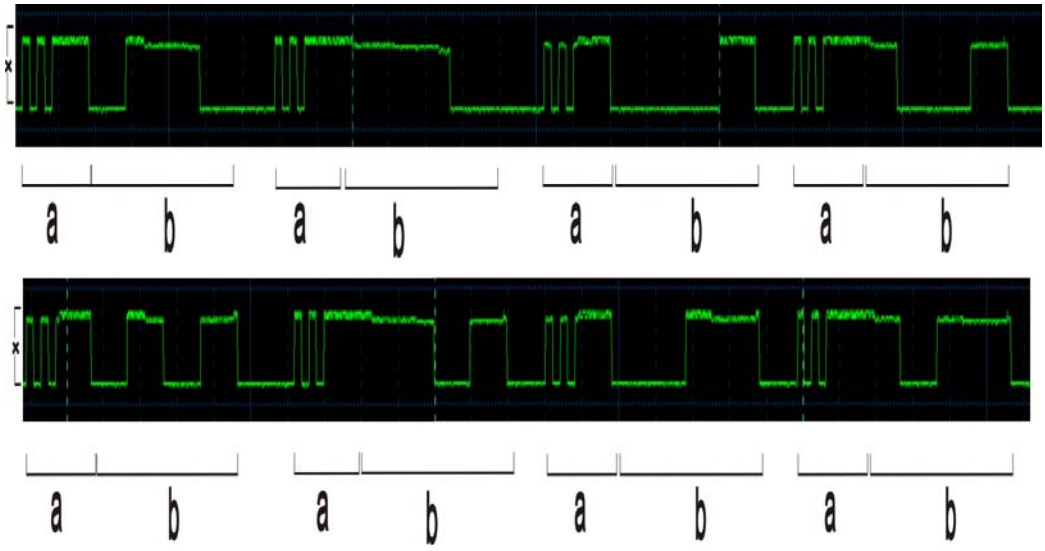
Keterangan : a = sinyal sinkronisasi

b = data

x = 4 V

Kecepatan Pengiriman 10 bit per deti

Bentuk Sinyal Pada Kaki Output Penerima Optik(TP3)

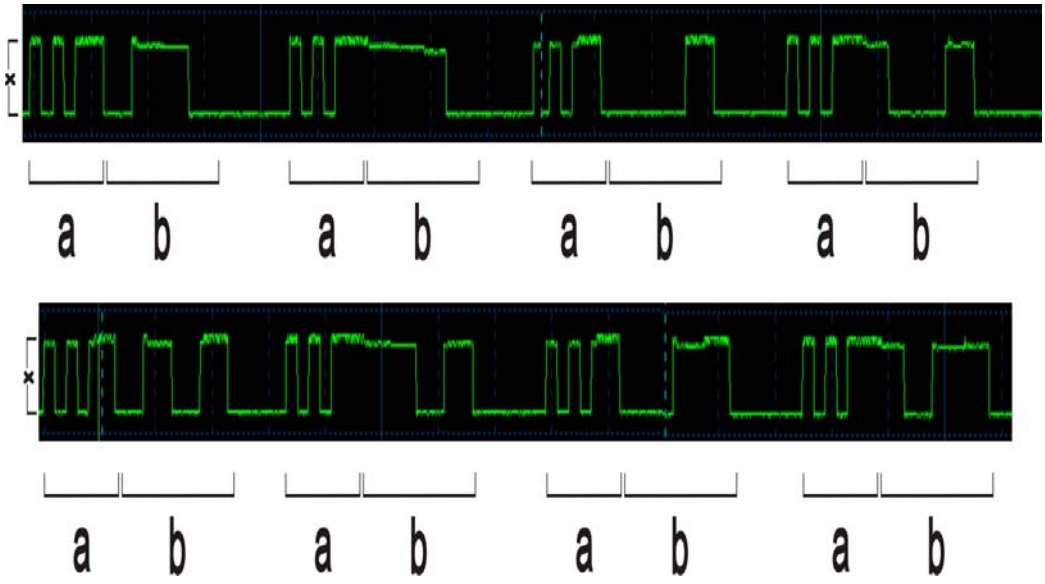


Keterangan : a = sinyal sinkronisasi

b = data

x = 2.8V

Kecepatan Pengiriman 1bit per detik

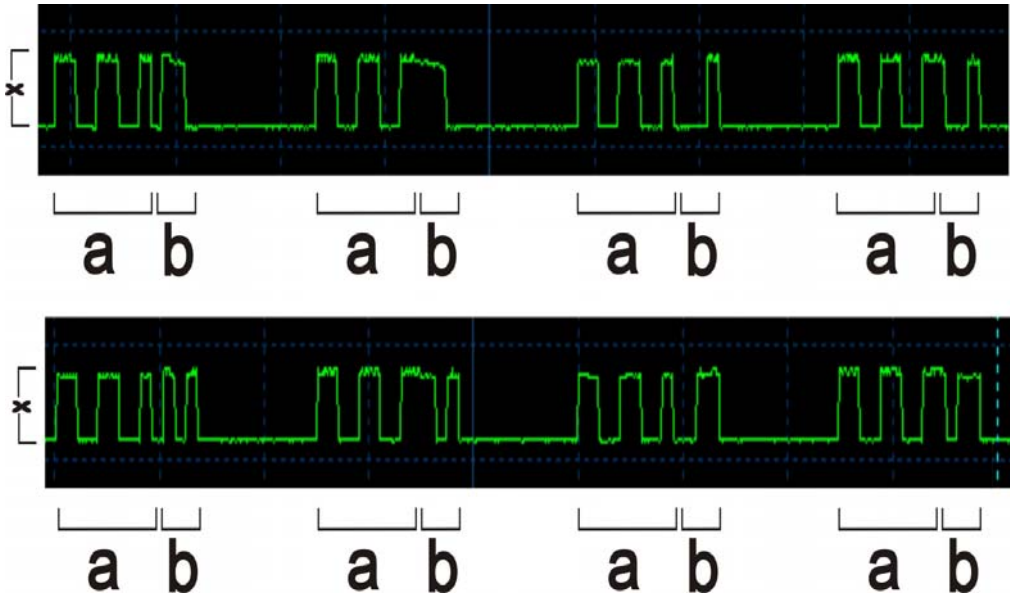


Keterangan : a = sinyal sinkronisasi

b = data

x = 2.8V

Kecepatan Pengiriman 2 bit per detik

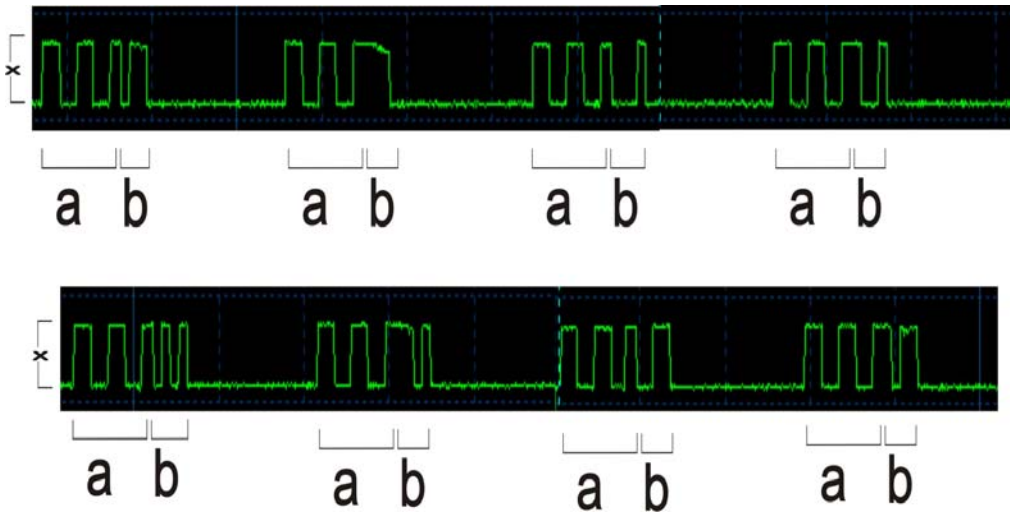


Keterangan : a = sinyal sinkronisasi

b = data

x = 2.8V

Kecepatan Pengiriman 9 bit per detik



Keterangan : a = sinyal sinkronisasi

b = data

x = 2.8V

Kecepatan Pengiriman 10 bit per detik

LAMPIRAN D

CERAMATE

TX179AT

Fiber Optic Transmitting Module

Fiber optic transmitting module for digital audio equipment and navigation system.

FEATURES

- Conform to EIAJ standard CP-1201 (For Digital audio interfaces including fiber optic inter-connections).
- TTL interface.
- LED is driven by differential circuit.
- +5V single power supply.
- High speed signal transmission (12.5M NRZ signal).

MAXIMUM RATINGS (Ta=25 °C)

Parameter	Symbol	Rating	Units
Supply Voltage	Vcc	-0.5 to 7	V
Input Voltage	V_{IN}	-0.5 to Vcc+0.5	V
Operating Temperature	Topr	-20 to 70	°C
Storage Temperature	Tstg	-30 to 80	°C

RECOMMENDED OPERATING CONDITIONS AND ELECTRICAL CHARACTERISTICS (Ta=25 °C)

Characteristic	Symbol	Condition	Min	Typ	Max	Units
Operating Voltage	Vcc		4.75	5	5.25	V
Operating Current	Iop		-	8	13	mA
Transmitter Wavelength	λp		-	660	-	nm
Transmitter Light Power	Pf	*1	-21	-	-15	dBm
Data Rate	T	NRZ Code *2	DC	-	12.5	Mb/s
Pulse Width Distortion	△tw	Pulse width 80ns Pulse cycle 160ns, C_L=10pF Using RX179	-25	-	25	ns
Jitter	△tj		-	2	25	ns
Low to High Delay Time	t_{PLH}			100	180	ns
High to Low Delay Time	t_{PHL}			100	180	ns
High Level Input Voltage	V_{IH}		2.0	-	-	V
Low Level Input Voltage	V_{IL}		-	-	0.8	V
High Level Input Current	I_{IH}		-	-	20	μA
Low Level Input Current	I_{IL}		-	-	-0.4	mA
Output Low Sink Current	I_{OH}	Vin at low level	0.01	0.04	0.1	mA
Output High Sink Current	I_{OL}	Vin at high level	4	6	10	mA

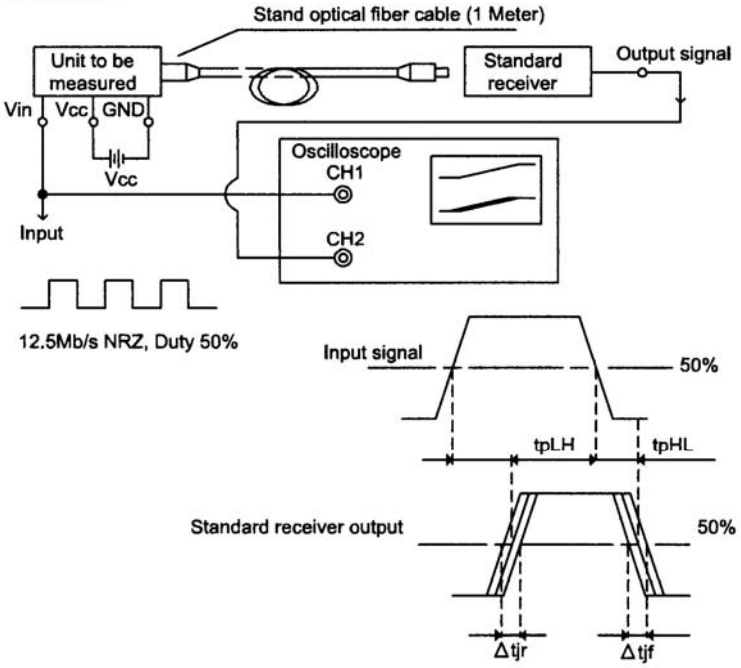
*1: Fiber insertion measure peak value.

*2: For data rate > 6Mb/s, the duty factor must be such as kept 25 to 75%.

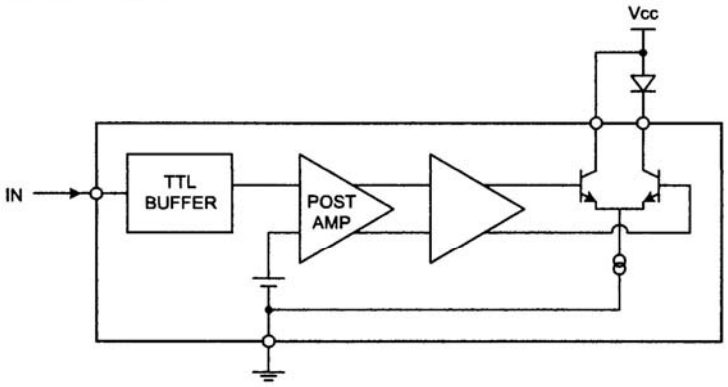
* All specs and applications shown above subject to change without prior notice.

1F-5 NO.66 SEC.2 NAN-KAN RD ., LUCHU , TAOYUAN, TAIWAN, R.O.C
Tel:886-3-3529445
Fax:886-3-3521052
Email: server@ceramate.com.tw
Http: www.ceramate.com.tw

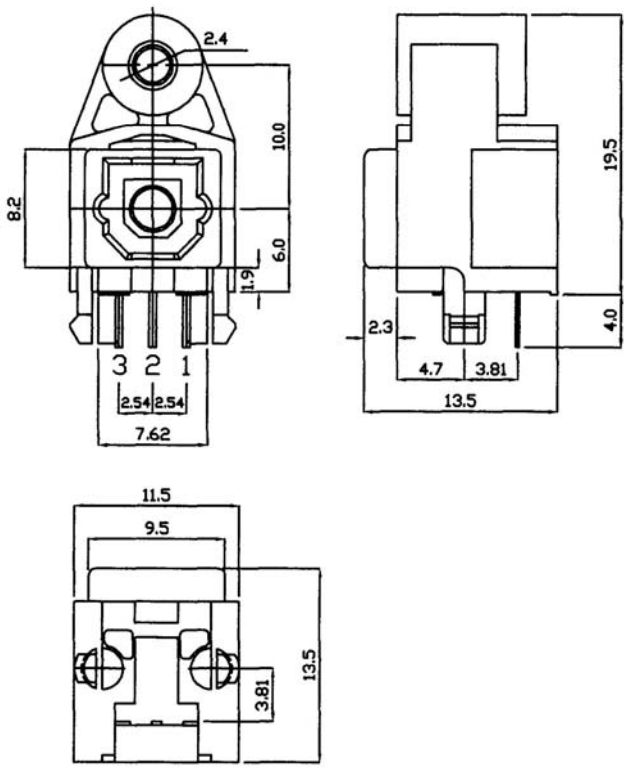
TEST CIRCUIT



BLOCK DIAGRAM



* All specs and applications shown above subject to change without prior notice.



PIN CONNECTION

1. GND
2. VCC
3. INPUT

◆ Housing Cap :Black

* All specs and applications shown above subject to change without prior notice.

1F-5 NO.66 SEC.2 NAN-KAN RD ., LUCHU , TAOYUAN, TAIWAN, R.O.C
Tel:886-3-3529445
Fax:886-3-3521052
Email: server@ceramate.com.tw
Http: www.ceramate.com.tw

PRECAUTIONS DURING USE

(1) Maximum rating

The maximum ratings are the limit values which must not be exceeded when using the device. Any one of the rating must not be exceeded. If the maximum rating is exceeded, the characteristics may not be recovered. In some extreme cases, the device may be permanently damage.

(2) Life of light emitters

When the optical module is used for over a long period, degeneration of characteristics is mostly due to lowering of the fiber output power (Pf). This is caused by the degradation of the optical output of the LED's used as the light source. The cause of degradation of the optical output of the LED's may be defects in wafer crystallization or mold resin stress. The detailed caused are, however, not clear.

The life of light emitters is greatly influenced by operating conditions and usage environment as well as the life characteristics unique to the device. Thus, when selecting a light emitter and setting the operating conditions,Ceramate recommends that you check the life characteristics.

Depending on the environment conditions, Ceramate recommends maintenance such as regular checks on the amount of optical output.

(3) Soldering

Optical modules use semiconductor devices internally. However, in principle, optical modules are optical components. At soldering, take care that flux dose not contact the emitting surface or detecting surface. Also take care at flux removal after soldering.

Some optical modules come with protective cap. The protective cap is used to avoid malfunction when the optical module is not in use. Not that it is not dust or waterproof.

As mentioned before, optical modules are optical component. Thus, in principle, soldering where there may be flux residue or flux removal after soldering is not recommended. Ceramate recommends that soldering be performed without the optical module mounted on the board. Then, after the board is cleaned, solder the optical module manually. Do not perform any further cleaning.

If the optical module cannot be soldered manually, use non-halogen (chlorine-free) flux and make sure, without cleaning, there is no residue such as chlorine. This is one of the ways to eliminate the effects of flux. In such a case, check the reliability.

1F-5 NO.66 SEC.2 NAN-KAN RD ., LUCHU , TAOYUAN, TAIWAN, R.O.C
Tel:886-3-3529445
Fax:886-3-3521052
Email: server@ceramate.com.tw
Http: www.ceramate.com.tw

(4) Vibration and shock

This module is resin-molded construction with wire fixed by resin. This structure is relatively sound against vibration or shock. In actual equipment, there are some cases where vibration, shock, and stress is applied to soldered parts or connected parts, resulting in line cut. Attention must be paid to the design of the mechanism for applications which are subject to large amounts of vibration.

(5) Fixing fiber optical transceiving module

Solder the fixed pin of fiber optic transmitting module TX179XX to the printed circuit board to fix the module to the board.

(6) Solvent

When using solvent for flux removal, do not use a high acid or high alkali solvent. Be-careful not to pour solvent in the optical connector ports. If solvent is inadvertently poured there, clean with cotton tips.

(7) Protective cap

When the fiber optic transmitting module TX179XX is not in use, use the protective cap.

(8) Supply voltage

Use the supply voltage within the Typ. Operating condition (Vcc = 5 ±0.25V). Make sure that supply voltage does not exceed the maximum rating value of 7V, even instantaneously.

1F-5 NO.66 SEC.2 NAN-KAN RD ., LUCHU , TAOYUAN, TAIWAN, R.O.C
Tel:886-3-3529445
Fax:886-3-3521052
Email: server@ceramate.com.tw
Http: www.ceramate.com.tw

(9) Input voltage

If a voltage exceeding the maximum rating value (Vcc+0.5V) is applied to the transmitter Input, the internal IC may degrade causing some damage. If excessive voltage due to surge may be added to the input, insert a protective circuit.

(10) Soldering condition

Solder at 260 °C or less within ten seconds.

(11) Precaution on waste

When discarding devices and packing materials, follow procedures stipulated by local regulations in order to protect the environment against contamination.
Compound semiconductors such as GaAs are used as LED materials for this module. When discarding waste or at final processing, attention must be paid to workers and the environment.

(12) Precaution on use

Ceramate is continually working to improve the quality and the reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress. It is the responsibility of the buyer, when utilizing Ceramate products, to observe standards of safety, and to avoid situations in which a malfunction or failure of a Ceramate product could cause loss of human life, bodily injury or damage to property.
In developing your designs, please ensure that Ceramate products are used within specified operating ranges as set forth in the most recent product specifications.

* All specs and applications shown above subject to change without prior notice.

1F-5 NO.66 SEC.2 NAN-KAN RD ., LUCHU , TAOYUAN, TAIWAN, R.O.C
Tel:886-3-3529445
Fax:886-3-3521052
Email: server@ceramate.com.tw
Http: www.ceramate.com.tw

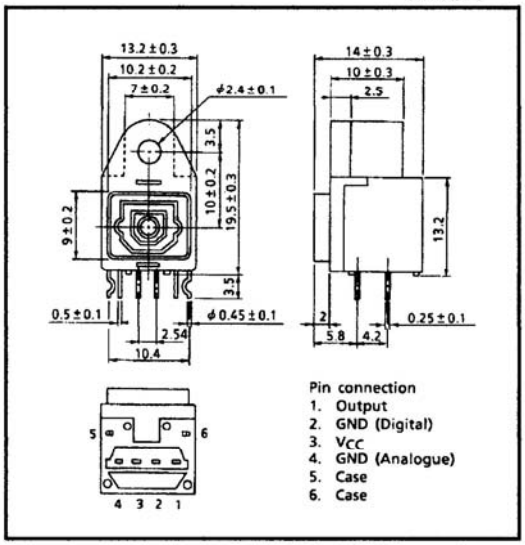
FIBER OPTIC RECEIVING MODULE

TORX176

FIBER OPTIC RECEIVING MODULE FOR DIGITAL AUDIO EQUIPMENT

- Conform to JEITA Standard CP-1201 (For Digital Audio Interfaces including Fiber Optic inter-connections).
- TTL Interface
- ATC (Automatic Threshold Control) Circuit is used for stabilized output at a wide range of optical power level.
- A self-tapping hole for easy attachment to the panel of Audio Equipment.

Unit: mm



1. Maximum Ratings (Ta = 25°C)

Characteristics	Symbol	Rating	Unit
Storage Temperature	T_{stg}	−40 to 70	°C
Operating Temperature	T_{opr}	−20 to 70	°C
Supply Voltage	V_{CC}	−0.5 to 7	V
Low Level Output Current	I_{OL}	20	mA
High Level Output Current	I_{OH}	−1	mA
Soldering Temperature	T_{sol}	260 (Note 1)	°C

Note 1: Soldering time ≤ 10 s (More than 1 mm apart from the package).

2. Recommended Operating Conditions

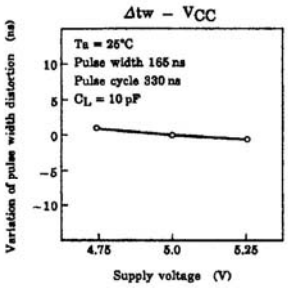
Characteristics	Symbol	Min	Typ.	Max	Unit
Supply Voltage	V_{CC}	4.75	5.0	5.25	V
High Level Output Current	I_{OH}	—	—	−150	μA
Low Level Output Current	I_{OL}	—	—	1.6	mA

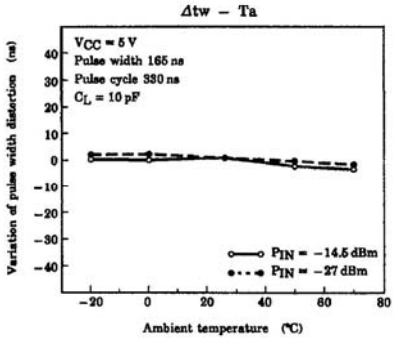
3. Electrical and Optical Characteristics (Ta = 25°C, V_{CC} = 5 V)

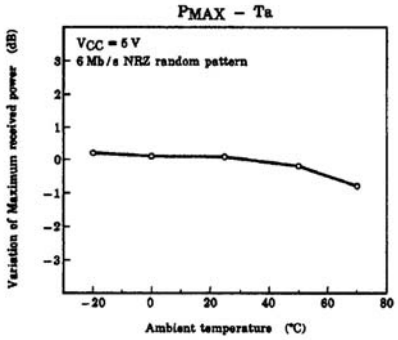
Characteristics	Symbol	Test Condition	Min	Typ.	Max	Unit
Data Rate		NRZ code (Note 2)	DC	—	6	Mb / s
Transmission Distance		TOTX176 Using APF (Note 3)	0.2	—	10	m
Pulse Width Distortion (Note 4)	Δtw	Pulse width 165 ns Pulse cycle 330 ns, C_L = 10 pF Using TOTX176	−20	—	20	ns
Maximum Receivable Power (Note 5)	P_{MAX}	6 Mb/s, Using APF	−14.5	—	—	dBm
Minimum Receivable Power (Note 5)	P_{MIN}	6 Mb / s, Using APF	—	—	−27	dBm
Rise Time	t_r	C_L = 10 pF	—	10	30	ns
Fall Time	t_f	C_L = 10 pF	—	5	30	ns
Current Consumption	I_{CC}		—	22	40	mA
High Level Output Voltage	V_{OH}		2.7	—	—	V
Low Level Output Voltage	V_{OL}		—	—	0.4	V

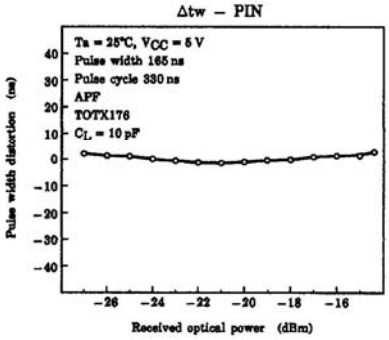
Note 2: For data rate > 3 Mb / s, the duty factor must be such as kept 25 to 75%. High level output when optical flux is received. Low level output when optical flux is not received.
Note 3: All Plastic Fiber (970 / 1000 μm).
Note 4: Between input of a fiber optic transmitting module and output of TORX176.
Note 5: BER ≤ 10^{-9}, rated by peak value.

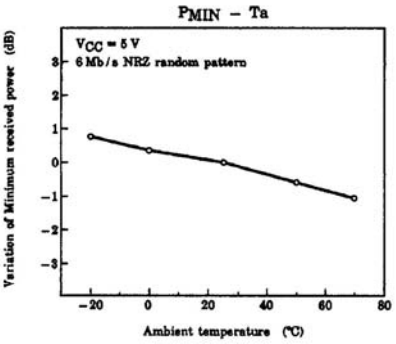
4. Example of Typical Characteristics (Note 6)





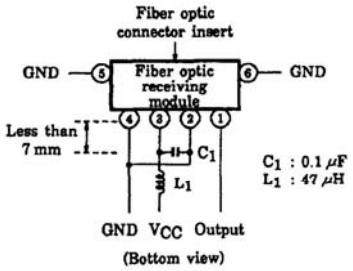






Note 6: There give characteristic examples, and its values are not guaranteed.

5. Application Circuit



6. Applicable optical fiber with fiber optic connectors.

TOCP172-□□B

7. Precautions during use

(1) Maximum rating
The maximum ratings are the limit values which must not be exceeded when using the device. Any one of the rating must not be exceeded. If The maximum rating is exceeded, the characteristics may not be recovered. In some extreme cases, the device may be permanently damage.

(2) Soldering
Optical modules use semiconductor devices internally. However, in principle, optical modules are optical components. At soldering, take care that flux dose not contact the emitting surface or detecting surface. Also take care at flux removal after soldering.
Some optical modules come with protective cap. The protective cap is used to avoid malfunction when the optical module is not in use. Not that it is not dust or waterproof.
As mentioned before, optical modules are optical component. Thus, in principle, soldering where there may be flux residue or flux removal after soldering is not recommended. Toshiba recommends that soldering be performed without the optical module mounted on the board. Then, after the board is cleaned, solder the optical module manually. Do not perform any further cleaning.
If the optical module cannot be soldered manually, use non−halogen (chlorine−free) flux and make sure, without cleaning, there is no residue such as chlorine. This is one of the ways to eliminate the effects of flux. In such a case, check the reliability.

(3) Noise resistance
Where the fiber optic receiving module case uses conductive resin, shield by connecting the reinforcing pin at a front end of the module to GND. When using this optical module, connect the pin to SIGNAL−GND.
Where the fiber optic receiving module case has a resistance of several tens of ohms, take care that the case does not contact power line of other circuits.
It is believed that the use of optical transfer devices improve the noise resistance. In principle, optical fiber is not affected by noise. However, especially receiving module which handle signals whose level is extremely small, are comparatively more susceptible to noise.
TOSLINK improves noise resistance using a conductive case. However, the current of the signal output from the photodiode of the optic receiving module is extremely small. Thus, depending on the usage environment, shielding the case is not sufficient for noise resistance.
When using TOSLINK, Toshiba recommends that you test using the actual device and check the noise resistance.
Use a simple noise filter on the TOSLINK fiber optic receiving module power line. If the ripple in power supply used is high, further reinforce the filter.
When locating the optical module in an area susceptible to radiated noise, increase shielding by covering the optical module and the power line filter using a metallic cover.

(4) Vibration and shock
This module is resin−molded construction with wire fixed by resin. This structure is relatively sound against vibration or shock, In actual equipment, there are some cases where vibration, shock, and stress is applied to soldered parts or connected parts, resultingin line cut. Attention must be paid to the design of the mechanism for applications which are subject to large amounts of vibration.

(5) Fixing fiber optical receiving module
Solder the fixed pin (pins 5 and 6) of fiber optic receiving module TORX176 to the printed circuit board to fix the module to the board.

(6) Panel attachment
TORX176 provides hole for panel attachment. Please be sure to attach it to panel.

(7) Shielding and wiring pattern of fiber optic receiving modules
To shield, connect the fixed pins (pins 5 and 6) of fiber optic transceiving module TORX176 to the GND.
Where the fiber optic receiving module uses conductive resin, be careful that the case does not touch wiring (including land).
To improve noise resistance, shield the optical module and the power line filter using a metallic cover.

(8) Solvent
When using solvent for flux removal, do not use a high acid or high alkali solvent. Be careful not to pour solvent in the optical connector ports. If solvent is inadvertently poured there, clean with cotton tips.

(9) Protective cap
When the fiber optic receiving module TORX176 is not in use, use the protective cap.

(10) Supply voltage
Use the supply voltage within the Typ. operating condition (V_{CC} = 5 ± 0.25 V). Make sure that supply voltage does not exceed the maximum rating value of 7 V, even instantaneously.

(11) Interface
TORX176 has a TTL interface. It can be interfaced with C–MOS IC that has compatibility with TTL level.

(12) Output
When the receiver output is at low level and connected to the power supply, or when the output is at high level and connected to GND, the internal IC may be destroyed.

(13) Soldering Condition
Solder at 260°C or less within ten seconds.

(14) Precaution on waste
When discarding devices and packing materials, follow procedures stipulated by local regulations in order to protect the environment against contamination.

(15) Precaution on use
Toshiba is continually working to improve the quality and the reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress. It is the responsibility of the buyer, when utilizing Toshiba products, to observe standards of safety, and to avoid situations in which a malfunction or failure of a Toshiba product could cause loss of human life, bodily injury or damage to property.
In developing your designs, please ensure that Toshiba products are used within specified operating ranges as set forth in the most recent product specifications. Also, please keep in mind the precautions and conditions set forth in the Toshiba Semiconductor Reliability Handbook.