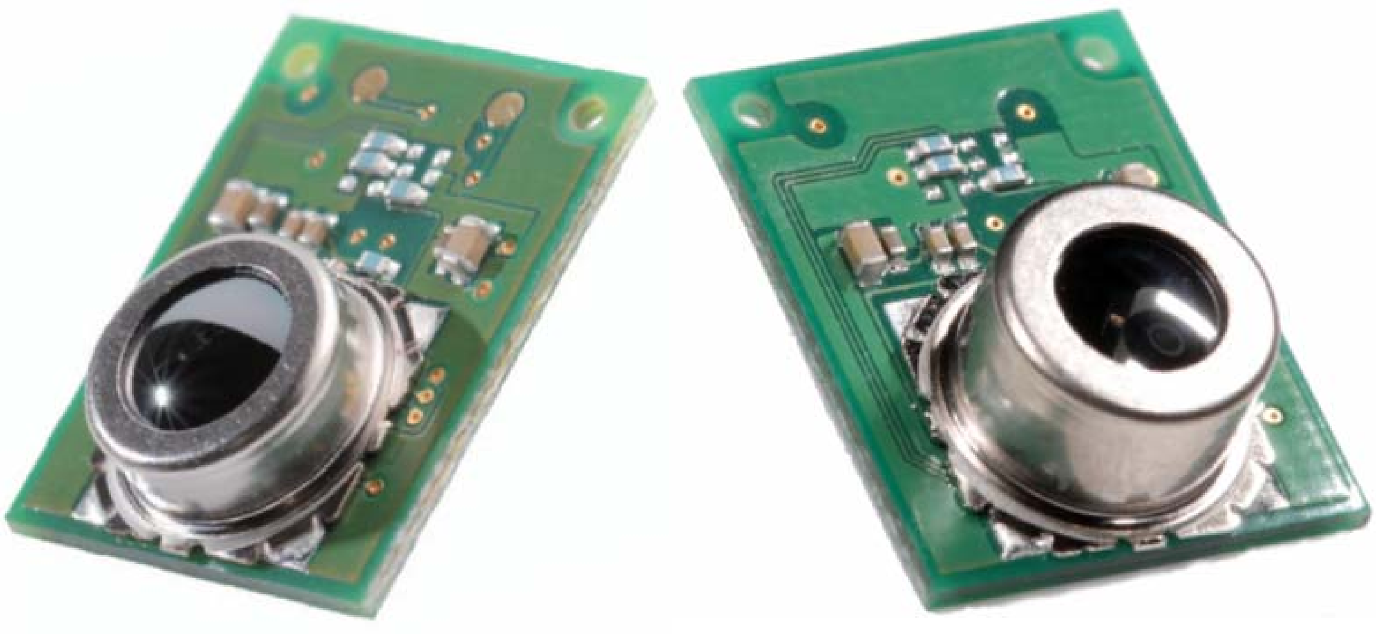
Application Note [No.001 : D6T FAQ Usage]

Usage of the D6T-44L / D6T-8L Thermal sensor

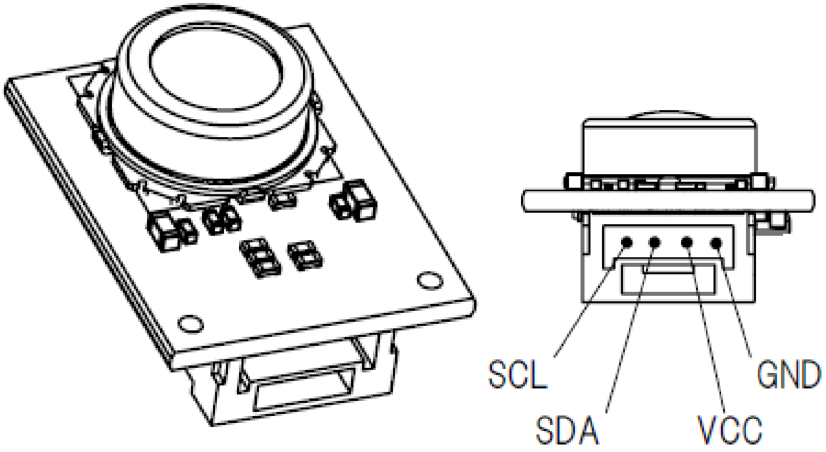
--- Connection and Getting data ---



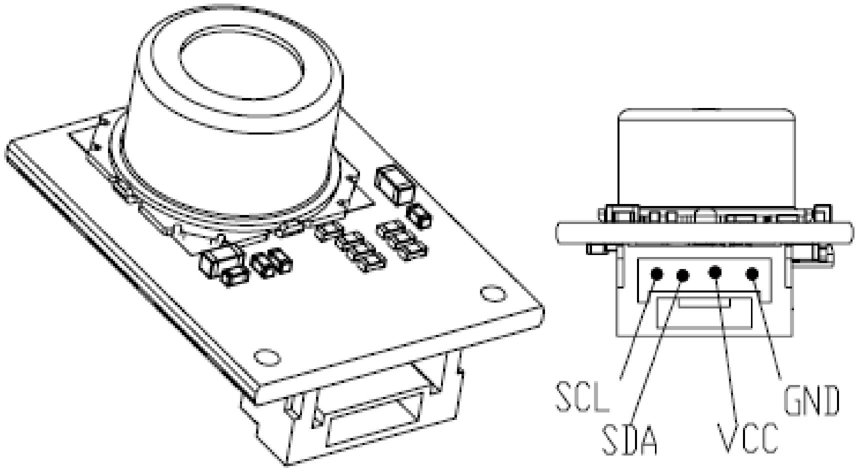
MDMK-12-0244

Application Note [D6T FAQ Usage]

Outer view



D6T-44L



D6T-8L

Connecter parts : SM04B-GHS-TB(JST)

I/O pin

GND	ground
VCC	5V +/-10%
SDA	I2C(5V) Data line
SCL	I2C(5V) Clock line

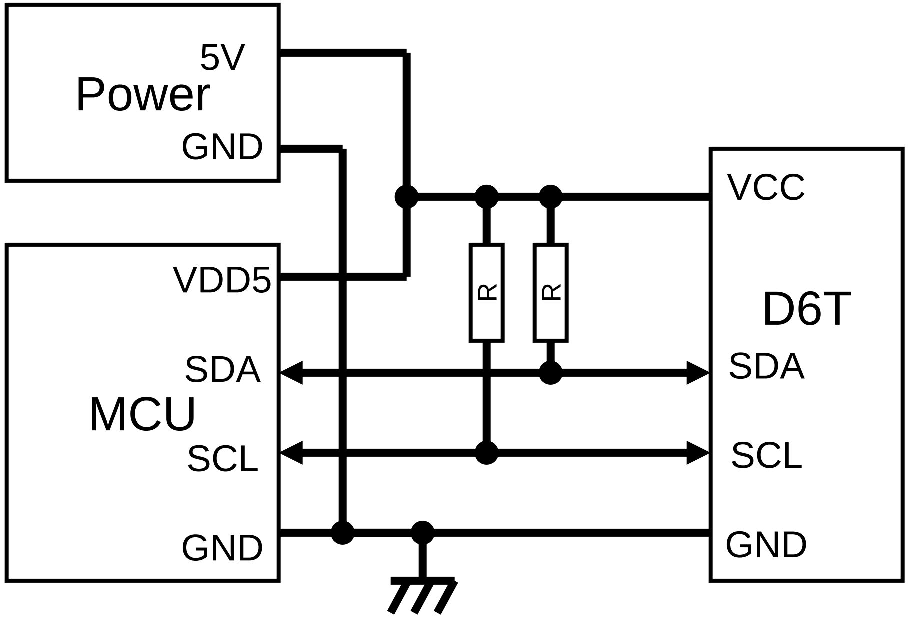
Connecter [JST]

Contact parts	SSHL-002T-P0.2
Housing parts	GHR-04V-S

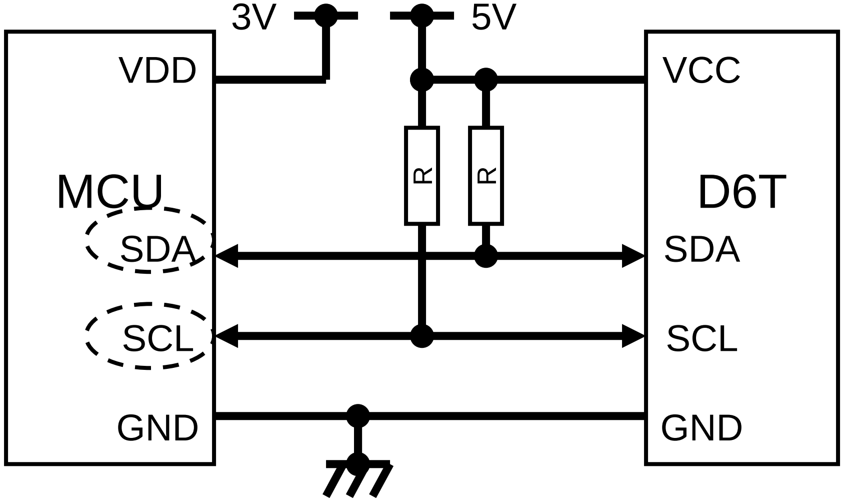
Application Note [D6T FAQ Usage]

Electrical Connection 1

Case1: 5V MCU (Direct)



Case2: 3V MCU (5V tolerant I2C port)
Direct connect



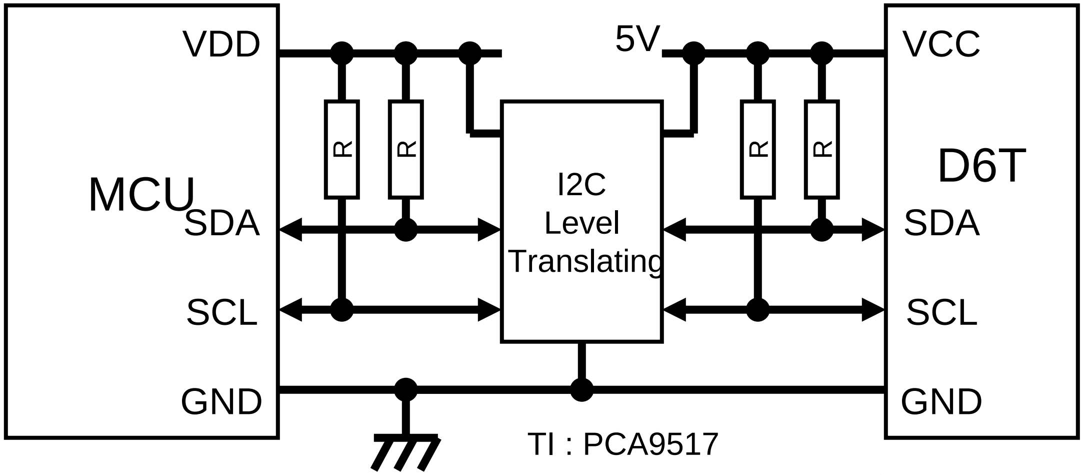
Pull-up Resister :
Impedance value is decided by user.
(see I2C[100kHz] specification note.)
(Most case : About 3k to 10k ohm)

Application Note [D6T FAQ Usage]

Electrical Connection 2

Case3: Using I2C converter

(no 5V tolerant port)
(other LV I2C device is exist)

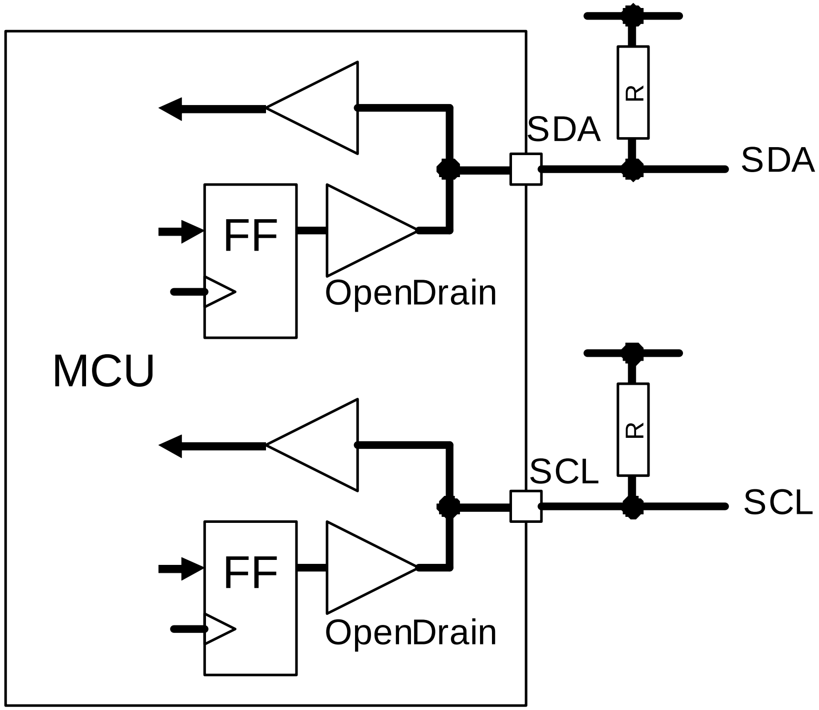


TI : PCA9517

Case4: Software I2C
using Bi-directional Open Drain ports

(MCU have no inside I2C module)

Note. Wait routine for Clock-Stretching
is required to prepare by user.

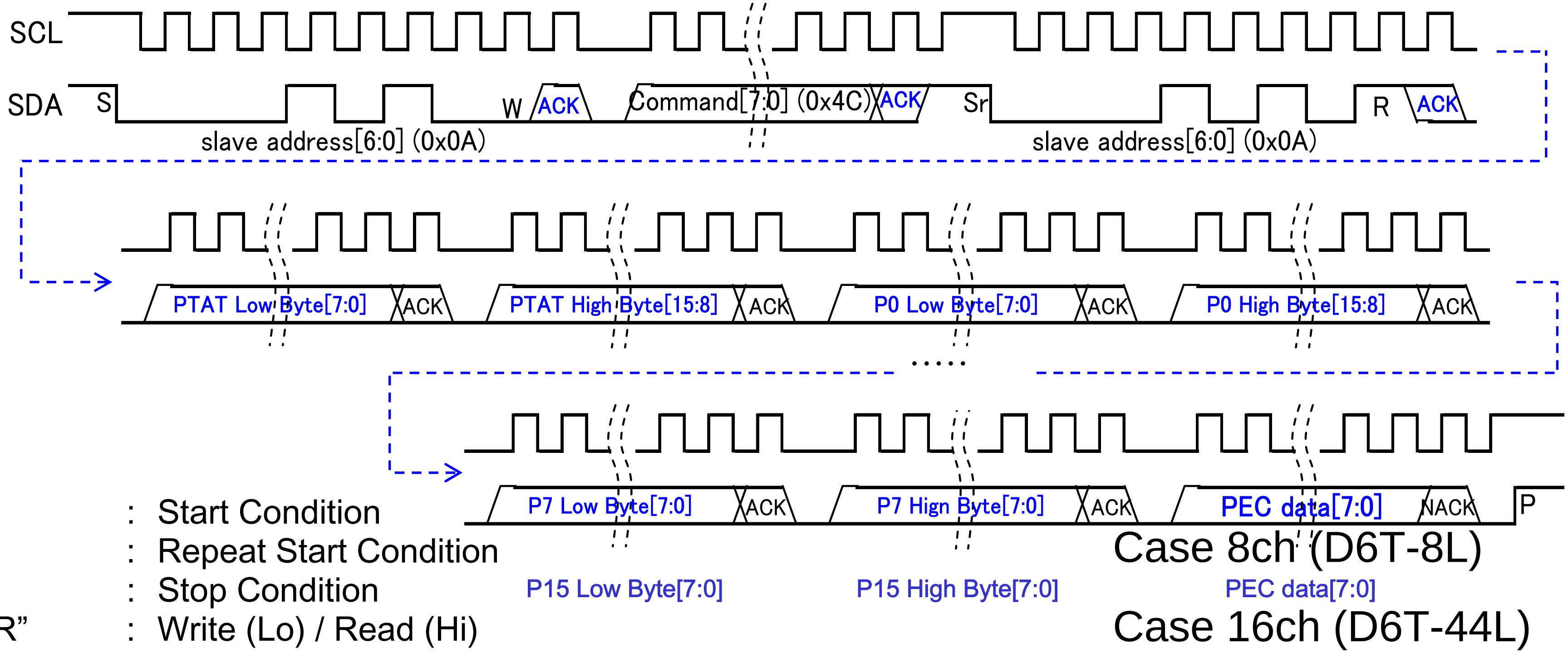


Application Note [D6T FAQ Usage]

I2C port condition setting

Device Address	7bit : 0001_010b
	8bit (with R/W bit) Read : 15h , Write : 14h
Data bit width	8bit (MSB-first)
Clock Frequency	max 100kHz
Control for Clock-stretching	On (Auto waiting)

Signal chart



"S" : Start Condition
"Sr" : Repeat Start Condition
"P" : Stop Condition
"W/R" : Write (Lo) / Read (Hi)

Example Getting the measurement value. (1/2)

```
// I2C communication functions
extern  void  I2C_start();        // Send Start condition
extern  void  I2C_repeatstart();  // Send Repeat Start condition
extern  void  I2C_stop();         // Send Stop condition
extern  void  I2C_send1( char addr8 , char cmd );  // Send 1 byte
extern  void  I2C_getx( char addr8 , char buff[] , int length );  // Get n bytes
extern  int  D6T_checkPEC( char buf , int pPEC );
// Global var.
extern  char  readbuff[35];
extern  int  tPTAT;
extern  int  tP[16];
extern  int  tPEC;

int  D6T_getvalue()
{
   I2C_start();
   I2C_send1( 0x14 , 0x4C ); // 14h = { 0Ah(Addr7) : Write(0b) }
   I2C_repeatstart();
   I2C_getx( 0x15 , readbuff , 35 ); // 15h = { 0Ah(Addr7) : Read(1b) } , 35 = 2*(1+16)+1
   I2C_stop();
   If(!D6T_checkPEC(readbuff,34)){
      return -1; // error
   }
   tPTAT = 256*readbuff[1] + readbuff[0];
   tP[0] = 256*readbuff[3] + readbuff[2];
   tP[1] = 256*readbuff[5] + readbuff[4];
   tP[2] = 256*readbuff[7] + readbuff[6];
   tP[3] = 256*readbuff[9] + readbuff[8];
   tP[4] = 256*readbuff[11] + readbuff[10];
   tP[5] = 256*readbuff[13] + readbuff[12];
   tP[6] = 256*readbuff[15] + readbuff[14];
```

Case 16ch (D6T-44L)

```
I2C_getx( 0x15 , readbuff , 19 ); // 19 = 2*(1+8)+1
I2C_stop();
If(!D6T_checkPEC(readbuff,18)){
   return -1; // error
}
```

Case 8ch (D6T-8L)

Example Getting the measurement value. (2/2)

```
        tP[7] = 256*readbuff[17] + readbuff[16];
        tPEC = readbuff[18];
        return 1;
    }
```

Case 8ch (D6T-8L)

Case 16ch (D6T-44L)

```
        tP[7] = 256*readbuff[17] + readbuff[16];
        tP[8] = 256*readbuff[19] + readbuff[18];
        tP[9] = 256*readbuff[21] + readbuff[20];
        tP[10] = 256*readbuff[23] + readbuff[22];
        tP[11] = 256*readbuff[25] + readbuff[24];
        tP[12] = 256*readbuff[27] + readbuff[26];
        tP[13] = 256*readbuff[29] + readbuff[28];
        tP[14] = 256*readbuff[31] + readbuff[30];
        tP[15] = 256*readbuff[33] + readbuff[32];
        tPEC = readbuff[34];
        return 1;
    }

    measure()
    {
        n = 0;
        do{
            status = D6T_getvalue();
            n++;
        }while(status < 0 && n < LOOPLIMIT);
        If(status < 0){
            // error operation.
        }
        printf( “%d,  %d,%d,%d,%d,%d,%d,%d,%d ,%d,%d,%d,%d,%d,%d,%d,%d ,%d¥n” ,
            tPTAT,tP[0],tP[1],tP[2],tP[3],tP[4],tP[5],tP[6],tP[7],
            tP[8],tP[9],tP[10],tP[11],tP[12],tP[13],tP[14],tP[15],tPEC);
    }
```

```
Output Example
223 ,224,224,273,335,239,221,240,297 ,264,232,221,254,299,258,229,233 ,80
223 ,271,261,265,304,284,270,264,274 ,302,285,271,260,319,304,286,269 ,193
223 ,296,273,285,311,306,291,281,301 ,311,310,293,296,312,322,311,302 ,83
```

Case 16ch (D6T-44L)
PTAT , 16 data , PEC

Example PEC check routine

Using PEC value, user can check data validity. (see SMBus specification).

```
unsigned char  calc_crc( unsigned char  data )
{
   int  index;
   unsigned char  temp;

   for(index=0;index<8;index++){
      temp = data;
      data <<= 1;
      if(temp & 0x80) data ^= 0x07;
   }
   return data;
}

int  D6T_checkPEC( char buf , int pPEC );
{
   unsigned char  crc;
   int  i;

   crc = calc_crc( 0x14 );
   crc = calc_crc( 0x4C ^ crc );
   crc = calc_crc( 0x15 ^ crc );
   for(i=0;i<pPEC;i++){
      crc = calc_crc( readbuff[i] ^ crc );
   }
   return (crc == readbuff[pPEC]);
}
```

Case A : Repeat Start Condition

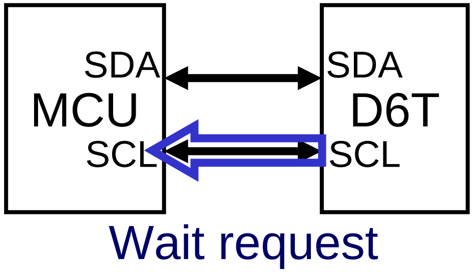
```
int  D6T_checkPEC( char buf , int pPEC );
{
   unsigned char  crc;
   int  i;

   crc = calc_crc( 0x15 );
   for(i=0;i<pPEC;i++){
      crc = calc_crc( readbuff[i] ^ crc );
   }
   return (crc == readbuff[pPEC]);
}
```

Case B : Start Condition (after stopped)

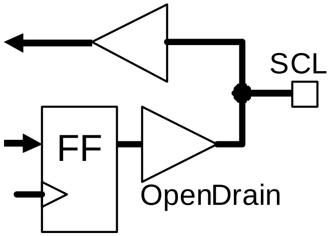
Application Note [D6T FAQ Usage]

Example Detect routine of wait status (Clock-stretching)
For Software-I2C using Bidirectional OpenDrain ports

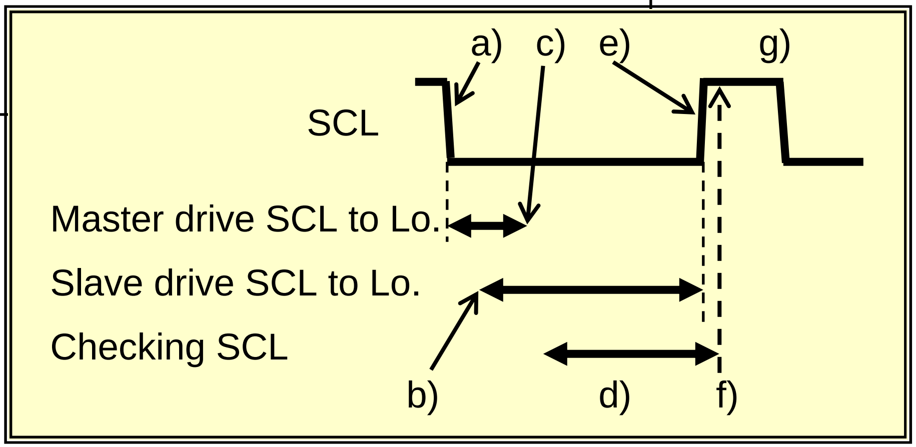


Wait request from Slave(Sensor) to Master(MCU).

Master	Slave (Sensor)
a) SCL drive to Lo for Ack.	Checking SCL status.(Lo)
	b) SCL drive to Lo for Wait.
c) SCL output change to Hi-Z.	Wait ...
SCL I/O mode change to Input	
d) Checking SCL status.(Hi)	Wait finish
Checking …	e) SCL output change to Hi-Z.
f) Finish Detected.	
SCL I/O mode change to Output	
g) Next operation.	



Bi-directional OpenDrain ports



Appendix

Application Note [D6T FAQ Usage]

For reference

Appendix

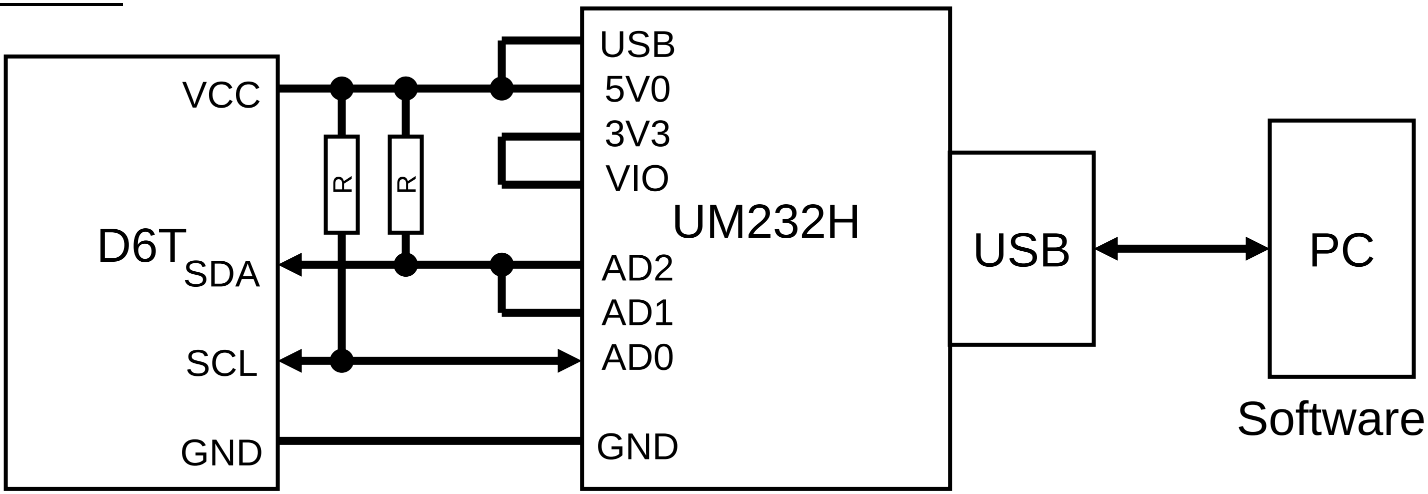
Connection to PC via USB using UM232H

(Future Technology Devices International Ltd)

I2C device control TEST
using UM232H(MPSSE mode)

UM232H
(FTDI evaluation board product.
http://www.ftdichip.com/)

Connection



For reference

Example

Source code

```
// Using ftd2xx.h , ftd2xx.lib
//   FT_Open, FT_Close, FT_SetBitMode, FT_Write, FT_Read, FT_CreateDeviceInfoList,
FT_GetDeviceInfoList
// MPSSE commands for I2C communication.
char  FT_P0[] = { 0x80,0x00,0x03 };
char  FT_P1[] = { 0x8A,0x85,0x97,0x8D, 0x80,0x03,0x03, 0x86,0x96,0};
char  FT_ST[] = { 0x80,0x03,0x03, 0x80,0x03,0x03, 0x80,0x03,0x03, 0x80,0x03,0x03, 0x80,0x01,0x03,
                  0x80,0x01,0x03, 0x80,0x01,0x03, 0x80,0x01,0x03, 0x80,0x01,0x03, 0x80,0x00,0x03 };
char  FT_ED[] = { 0x80,0x01,0x03, 0x80,0x01,0x03, 0x80,0x01,0x03, 0x80,0x01,0x03, 0x80,0x03,0x03 };
char  FT_WD[] = { 0x11,0x00,0x00,0x00, 0x80,0x00,0x01, 0x22,0x00,0x87, 0x80,0x00,0x03 };
char  FT_RD[] = { 0x80,0x00,0x01, 0x20,0x00,0x00, 0x80,0x00,0x03, 0x22,0x00,0x87 };

void  MPSSE_init(int  chMPSSE)
{
   FT_Open(chMPSSE,&ftHandle);
   FT_SetBitMode(ftHandle, 0, 2);
   FT_Write(ftHandle,FT_P1, sizeof(FT_P1),&ltemp);
}

void  D6T_measure(unsigned char  data, int length)
{
   DWORD ltemp;
   Int  I;

   FT_Write(ftHandle,FT_ST, sizeof(FT_ST),&ltemp);
   FT_Write(ftHandle,FT_P0, sizeof(FT_P0),&ltemp);
   FT_WD[3] = 0x14;
   FT_Write(ftHandle,FT_WD, sizeof(FT_WD),&ltemp);
   FT_Read(ftHandle,bufr,1,&ltemp);
   FT_Write(ftHandle,FT_P0, sizeof(FT_P0),&ltemp);
   FT_WD[3] = 0x4C;
   FT_Write(ftHandle,FT_WD, sizeof(FT_WD),&ltemp);
   FT_Read(ftHandle,bufr,1,&ltemp);
   FT_Write(ftHandle,FT_P0, sizeof(FT_P0),&ltemp);
   FT_Write(ftHandle,FT_ST, sizeof(FT_ST),&ltemp);
   FT_Write(ftHandle,FT_P0, sizeof(FT_P0),&ltemp);
   FT_WD[3] = 0x15;
   FT_Write(ftHandle,FT_WD, sizeof(FT_WD),&ltemp);
   FT_Read(ftHandle,bufr,1,&ltemp);
   FT_Write(ftHandle,FT_P0, sizeof(FT_P0),&ltemp);
   for(i=0;i<length;i++){
      FT_Write(ftHandle,FT_RD, sizeof(FT_RD),&ltemp);
      FT_Read(ftHandle,bufr,2,&ltemp);
      FT_Write(ftHandle,FT_P0, sizeof(FT_P0),&ltemp);
      *data++ = (char)bufr[0];
   }
   FT_Write(ftHandle,FT_ED, sizeof(FT_ED),&ltemp);
}
```