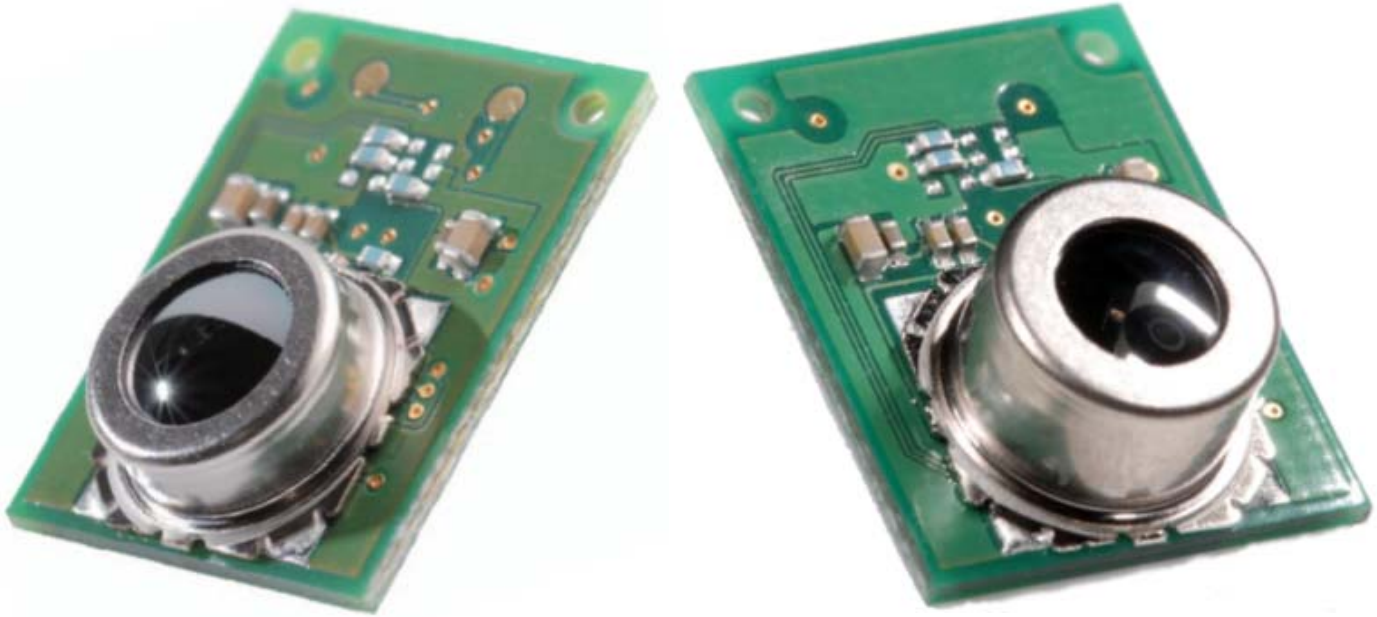
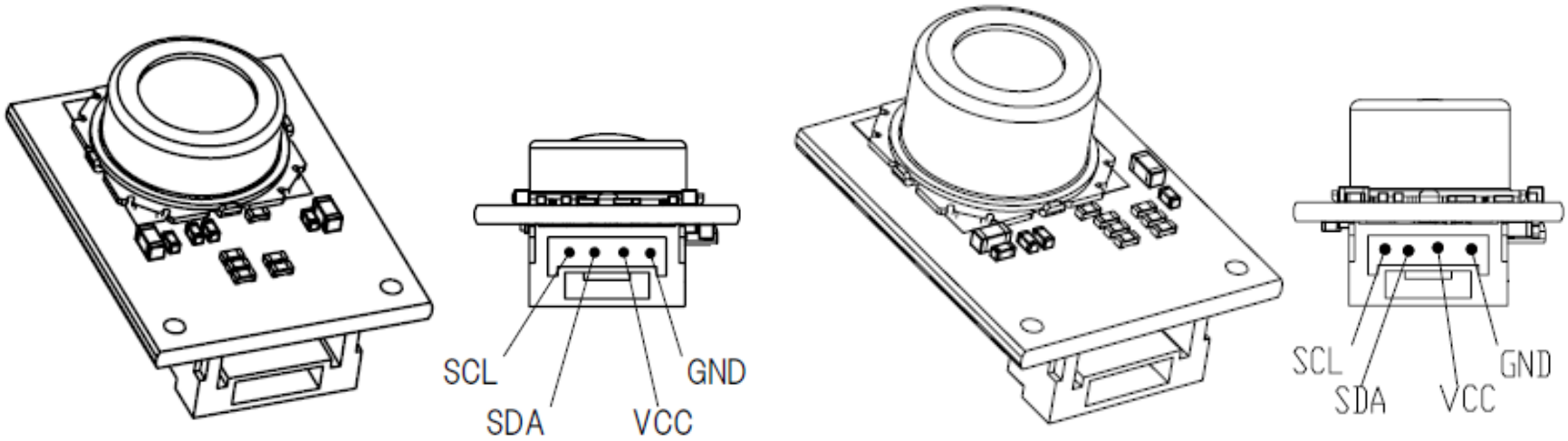
Usage of the D6T-44L / D6T-8L Thermal sensor

--- Connection and Getting data ---



MDMK-12-0244

Outer view



D6T-44L　　D6T-8L

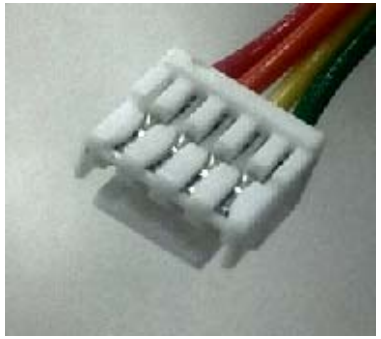
Connecter parts : SM04B-GHS-TB(JST)

I/O pin

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

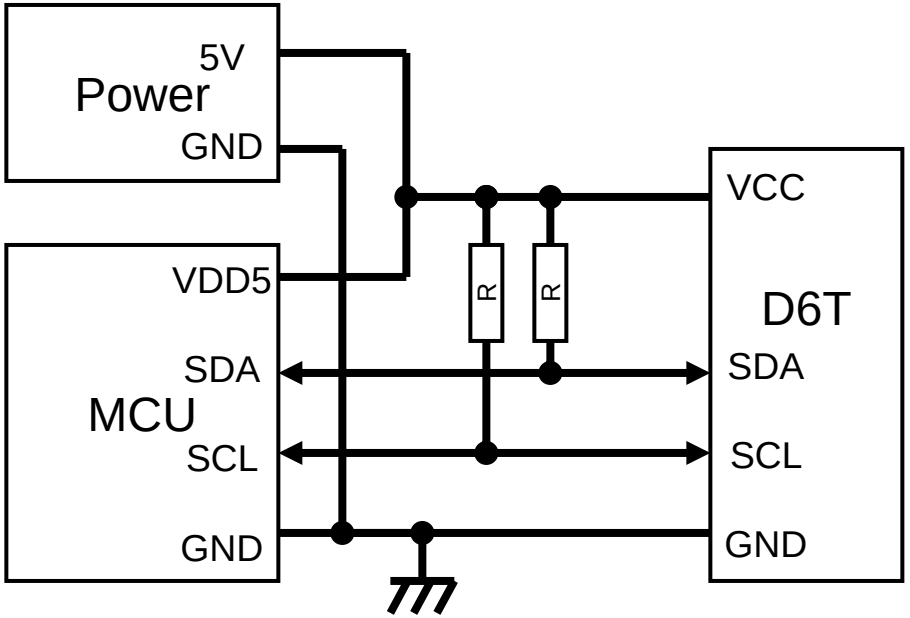
Connecter [JST]

Contact parts : 4pcs SSHL-002T-P0.2
Housing parts : GHR-04V-S

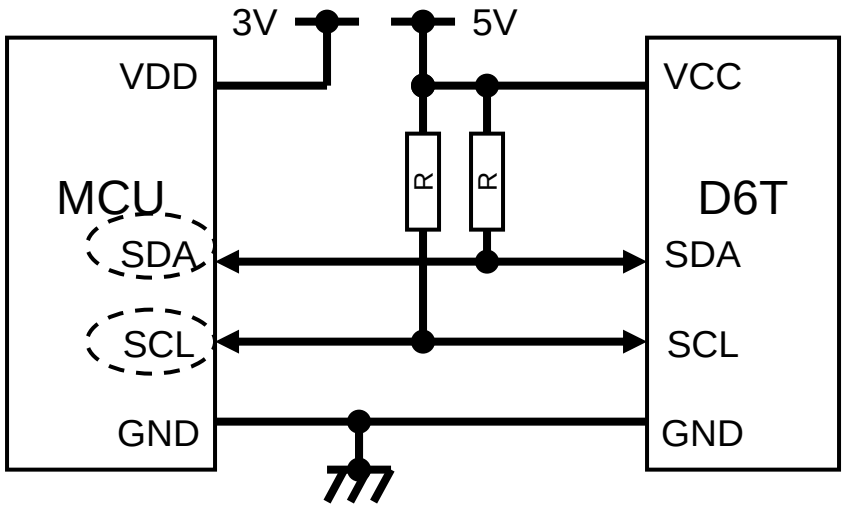


Electrical Connection 1

Case1: 5V MCU (Direct)



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

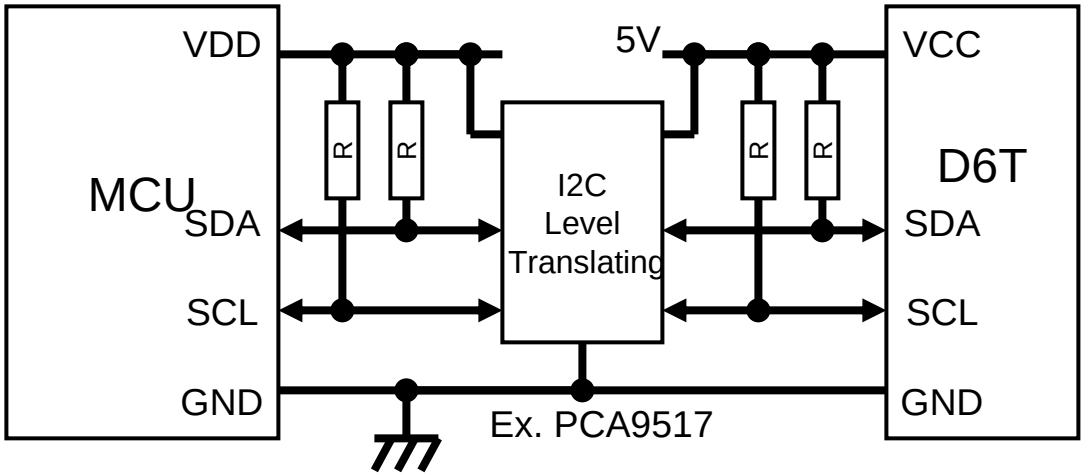


Pull-up Resister :
Impedance value is decided by user.
(see I2C[100kHz] specification note.)
(Most case : About 3k to 10k ohm)

Electrical Connection 2

Case3: Using I2C converter

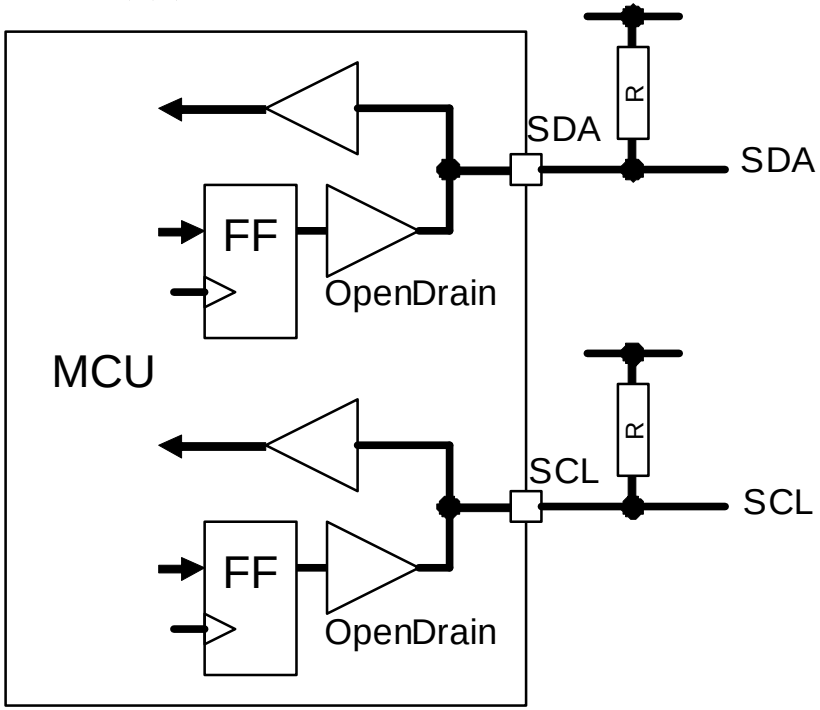
(no 5V tolerant port)
(other LV I2C device is exist)



Case4: Software I2C
using Bi-directional Open Drain ports

(MCU have no inside I2C module)

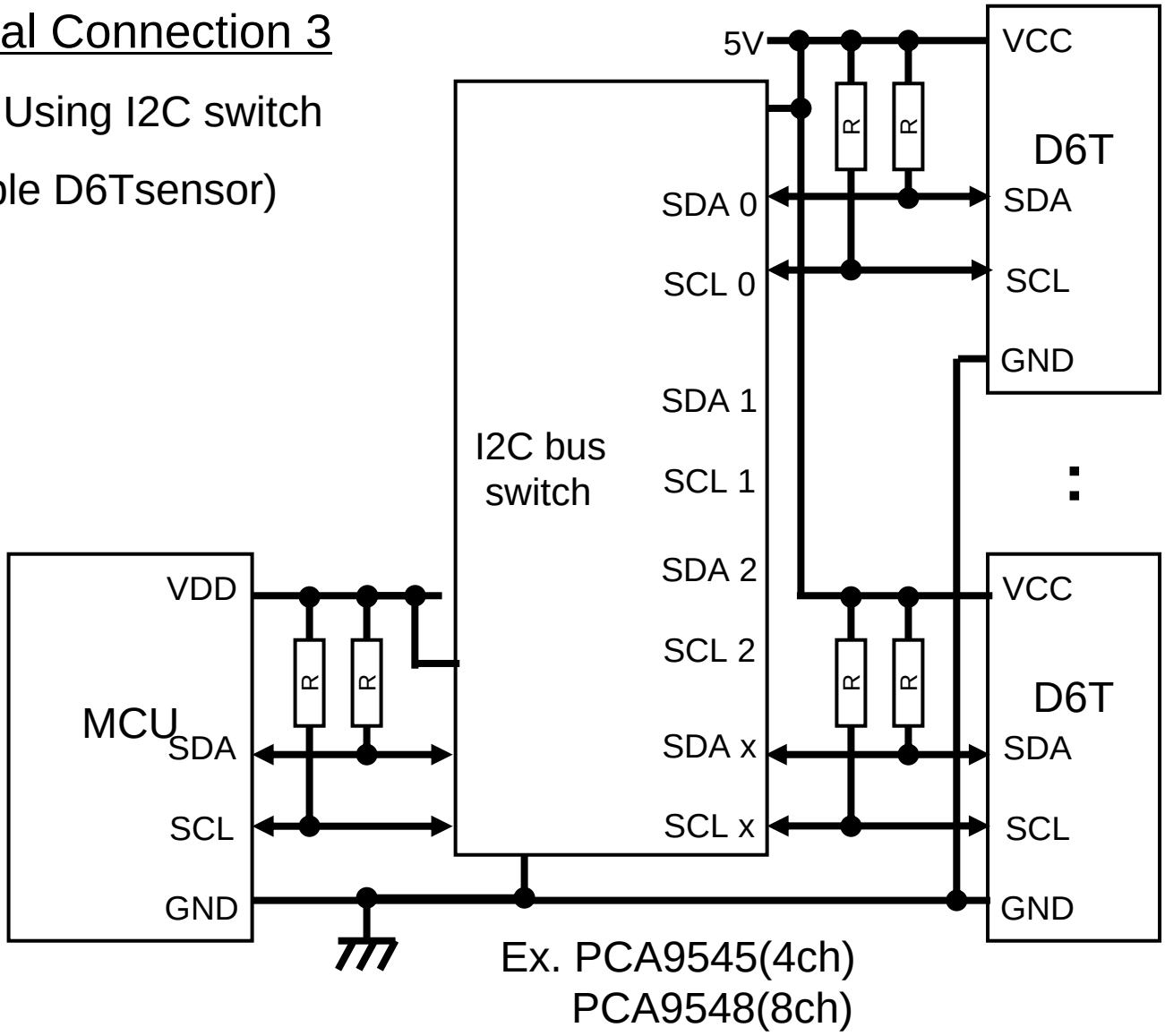
Note. Wait routine for Clock-Stretching
is required to prepare by user.



Electrical Connection 3

Case5: Using I2C switch

(multiple D6Tsensor)

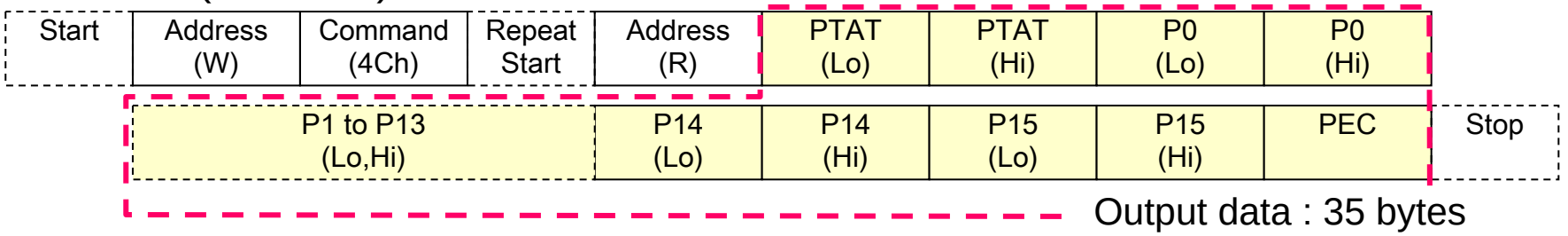


I2C port condition setting

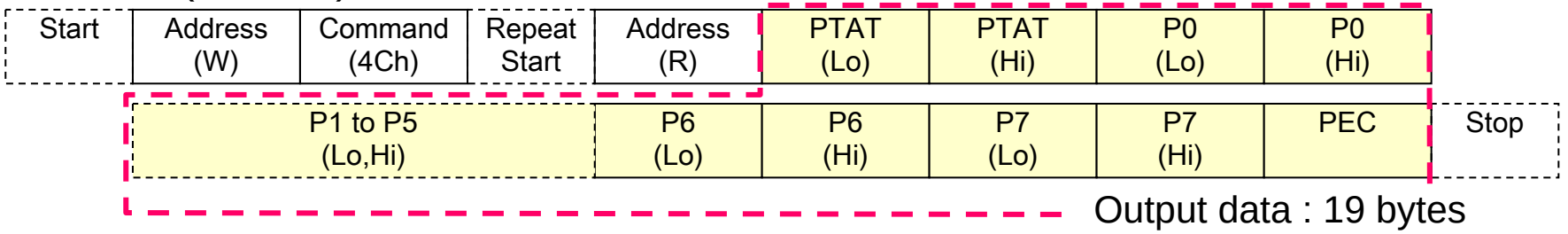
Device Address	7bit : 0001_010b
	8bit (with R/W bit) Read : 15h , Write : 14h
Data bit width	8bit (MSB-first)
Clock Frequency	max 100kHz
Control for Clock-stretching	On (Auto waiting)

Port-data chart

Case 16ch (D6T-44L)



Case 8ch (D6T-8L)



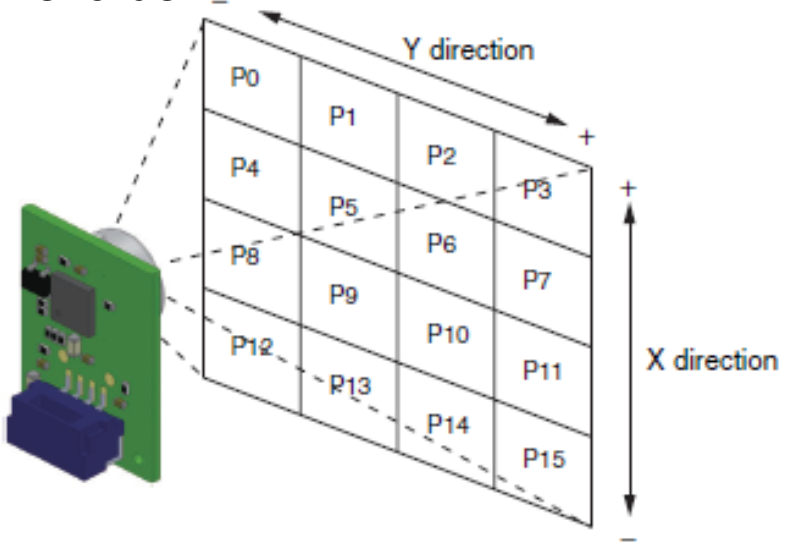
Output Data Format

1. PTAT : It is the value of the reference temperature, inside the sensor module.

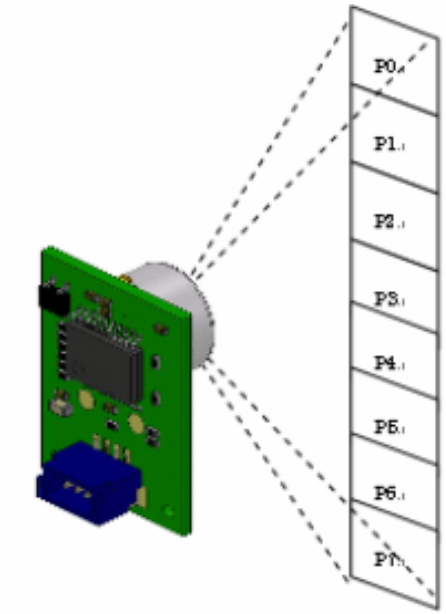
Temperature data (PTAT & Pn) is 16bit-width, singed, 10 times value of degC.
Example : 12.7 degC = 007Fh(127) , 25.8 degC = 0102h(258)

2. P0 to P7(D6T-8L-06) , P0 to P15(D6T-44L-06)

Pixel order



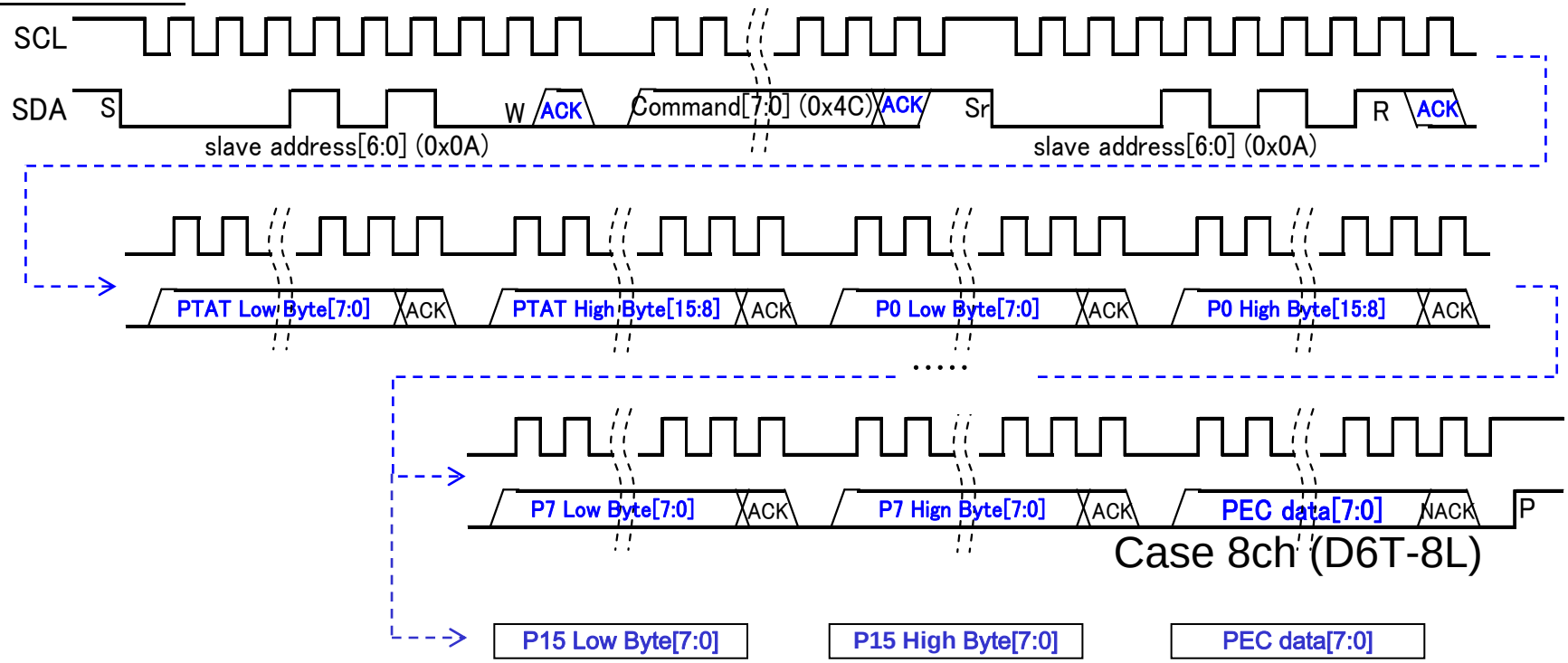
Case 16ch (D6T-44L)



Case 8ch (D6T-8L)

3. PEC : Packet error check code. Based on the “SM bus” specification.

Signal chart



“S”	: Start Condition
“Sr”	: Repeat Start Condition
“P”	: Stop Condition
“W/R”	: Write (Lo) / Read (Hi)
“ACK”	: Acknowledge reply
“NACK”	: No-acknowledge reply

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];
        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);
    }
```

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

Case 8ch (D6T-8L)

Case 16ch (D6T-44L)

```
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

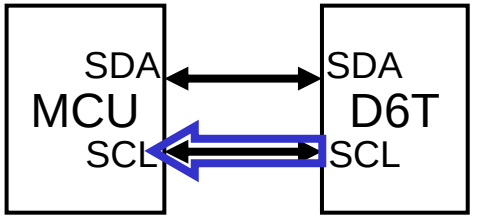
```
int  D6T_checkPEC( char buf , int pPEC );
{
   unsigned char  crc;
   int  i;

   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)

OMRON
Sensing tomorrow™

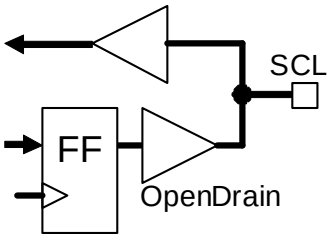
Example Detect routine of wait status (Clock-stretching)
For Software-I2C using Bidirectional OpenDrain ports

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



Wait request

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



Bi-directional OpenDrain ports