Project #15(Pjt15_TWI_TCN75)

- TWI Interface Application : TWI Driver for TCN75

```
#include <util/twi.h>
 void  twi_init()
{
     cbi(DDRD, 0);  sbi(PORTD, 0);          // SCL ← 1 : enable pull-up register
     cbi(DDRD, 1);  sbi(PORTD, 1);          // SDA ← 1 : enable pull-up register
     TWSR = 0x00;                           // prescaler value==4^0==1
     TWBR = (F_CPU)/(200*1024L-16)/(2*4^0); // 2.5us==400KHz➔200KHz

     sbi(DDRH, 1);  cbi(PORTH, 1);          // A0 ← 0    TCN75 id bits
     sbi(DDRH, 3);  cbi(PORTH, 3);          // A1 ← 0
     sbi(DDRH, 5);  cbi(PORTH, 5);          // A2 ← 0
}
```

Project #15(Pjt15_TWI_TCN75)

```
inline uint8_t twi_start()
{
     TWCR = (1<<TWINT) | (1<<TWSTA) | (1<<TWEN) ;  // clear TWINT, set START flag
     while (!(TWCR & (1<<TWINT))) ;                 // wait transmission completion
     return(TW_STATUS);                            //  TWSR & 0xf8 → 0x08 or 0x10
}
inline uint8_t twi_write_one_byte(uint8_t data)
{
      TWDR = data;
      TWCR = (1<<TWINT) | (1<<TWEN)) ;             // clear TWINT and TWSTA, keep TWEN
      while (!(TWCR & (1<<TWINT))) ;               // wait transmission completion
      return(TW_STATUS);                           // TWSR & 0xf8 → 0x18, 0x20, 0x28,0x30
}
inline uint8_t twi_read_one_byte(uint8_t *data, uint8_t  enack)
{
      TWCR = (1<<TWINT) | (enack<<TWEA) | (1<<TWEN) ;  // clear TWINT, keep TWEN
      while (!(TWCR & (1<<TWINT))) ;                   // wait reception completion
      *data = TWDR;
      return(TW_STATUS);                               // TWSR & 0xf8 → 0x50, 0x58
}
inline void  twi_stop()
{
     TWCR = (1<<TWINT) | (1<<TWSTO) | (1<<TWEN) ;  // clear TWINT, keep TWEN
}
```

Project #15(Pjt15_TWI_TCN75)

- TWI Interface Sensor Application : task_tc75_twi

```
void  task_tc75_twi(void *arg)
{
      uint16_t value;   uint8_t  hbyte, lbyte;

      if (twi_start() != TW_START) { twi_stop(); printf("task_tc75: Start error…₩n"); return; }
      if (twi_write_one_byte(0x90) != TW_MT_SLA_ACK) {          // address + write_operation
            twi_stop();  printf("task_tc75() : SLA+W write fail…₩n");  return;
      }
      if (twi_write_one_byte(0x00) != TW_MT_DATA_ACK) {       // pointer(TEMP)
            twi_stop();  printf("task_tc75() : pointer write fail…₩n");  return;
      }
      if (twi_start() != TW_REP_START) {twi_stop(); printf("Repeat Start error…₩n"); return; }
      if (twi_write_one_byte(0x91) != TW_MR_SLA_ACK) {          // address + read_operation
            twi_stop();  printf("task_tc75() : SLA+R write fail…₩n");  return;
      }
      if (twi_read_one_byte(&hbyte, 1) != TW_MR_DATA_ACK) {   // read hi_byte of TEMP register
            twi_stop();  printf("task_tc75() : read hi_byte fail…₩n");  return;
     }
      if (twi_read_one_byte(&lbyte, 0) != TW_MR_DATA_NACK) {    // read lo_byte of TEMP register
            twi_stop();  printf("task_tc75()_1 : read lo_byte fail…₩n");  return;
     }
     twi_stop();
     value = ((hbyte << 8 | lbyte) >> 7) >> 1;                          // value = value * 0.5
     printf("task_tc75()_1 : current_temperature ➔ %d degree.₩n", value);
}
```

Project #15(Pjt15_TWI_TCN75)

```
void task_cmd(void *arg)
 {
        …
        …
        else if (!strcmp(cp0, “tcn75_twi”))
                task_tc75_twi(“”);
        …
        …
 }
```