twi.h

```
/*
 deviltronic
  (C) Winfried Jaeckel

  TWI access functions(headerfile)
*/
#ifndef TWI_INCLUDED_
#define TWI_INCLUDED_


#pragma used+
//-----------------------------
void TWI_STARTCONDITION();
void TWI_RSTARTCONDITION();
void TWI_SLA_W (unsigned char SLA_W);
void TWI_SLA_R (unsigned char SLA_R);
void TWI_SENDDATA (unsigned char DATA);
unsigned char TWI_RECIVEDATA_ACK ();
unsigned char TWI_RECIVEDATA_NACK ();
void TWI_STOPCONDITION();
#pragma used-
#pragma library twi.lib

#endif
/*
```

twi.lib

```
 deviltronic
  (C) Winfried Jaeckel

  TWI access functions
*/
#ifndef
#define TWBR  (*(unsigned char *)  0x70)
#define TWSR  (*(unsigned char *)  0x71)
#define TWAR  (*(unsigned char *)  0x72)
#define TWDR  (*(unsigned char *)  0x73)
#define TWCR  (*(unsigned char *)  0x74)
#define OCR1CL (*(unsigned char *) 0x78)
#endif

const char
TWINT=7
,TWEA=6
,TWSTA=5
,TWSTO=4
,TWWC=3
,TWEN=2
,TWIE=0

,START=0x08
,RSTART=0x10
,MT_SLA_ACK=0x18
,MT_DATA_ACK=0x28
,MR_SLA_ACK=0x40
,MR_DATA_ACK=0x50
,MR_DATA_NACK=0x58
;


//------------------------------
void TWIERROR(int x)   // dummyfunktion  kann zur fehleranalyse benutzt werden
{
}
//------------------------------
void TWI_STARTCONDITION()
{
TWCR=0x00 |(1<<TWINT)|(1<<TWEN)|(1<<TWSTA);
while (!(TWCR & (1<<TWINT)));
if((TWSR & 0xF8)!=START)TWIERROR(1);
}
//------------------------------
```

```
void TWI_RSTARTCONDITION()
{
TWCR = 0x00 |(1<<TWINT)|(1<<TWEN)|(1<<TWSTA);
while (!(TWCR & (1<<TWINT)));
if((TWSR & 0xF8)!=RSTART)TWIERROR(2);
}
//------------------------------
void TWI_SLA_W (unsigned char SLA_W)
{
TWDR = SLA_W;
TWCR = 0x00 |(1<<TWINT) | (1<<TWEN);
while (!(TWCR & (1<<TWINT)));
if ((TWSR & 0xF8)!=MT_SLA_ACK)TWIERROR(3);
}
//------------------------------
void TWI_SLA_R (unsigned char SLA_R)
{
TWDR =SLA_R;
TWCR = 0x00 |(1<<TWINT) | (1<<TWEN);
while (!(TWCR & (1<<TWINT)));
if ((TWSR & 0xF8)!=MR_SLA_ACK)TWIERROR(4);
}
//------------------------------
void TWI_SENDDATA (unsigned char DATA)
{
TWDR=DATA;
TWCR = 0x00 | (1<<TWINT) | (1<<TWEN);
while (!(TWCR & (1<<TWINT)));
if ((TWSR & 0xF8)!=MT_DATA_ACK)TWIERROR(5);
}
//------------------------------
unsigned char TWI_RECIVEDATA_ACK ()
{
char DATA;
TWCR = 0x00 |(1<<TWINT) | (1<<TWEA)| (1<<TWEN);
while (!(TWCR & (1<<TWINT)));
if ((TWSR & 0xF8)!=MR_DATA_ACK)TWIERROR(6);
DATA=TWDR;
return DATA;
}
//------------------------------
unsigned char TWI_RECIVEDATA_NACK ()
{
char DATA;
TWCR = 0x00| (1<<TWINT) | (1<<TWEN);
while (!(TWCR & (1<<TWINT)));
if ((TWSR & 0xF8)!=MR_DATA_NACK)TWIERROR(7);
DATA=TWDR;
return DATA;
}
//------------------------------
```

```
void TWI_STOPCONDITION()
{
TWCR=0x00 |(1<<TWINT)|(1<<TWEN)|(1<<TWSTO);
}
```