```c
/* DS18B20 Self made Library
Author: Guillermo J. Ramires
Student nr: 464852 */
#ifndef DS18B20_H_
#define DS18B20_H_


#include <avr/io.h>
#include <util/delay.h>

#define SKIP_ROM 0xCC
#define CONVERT_T 0x44
#define WRITE_SCRATCHPAD 0x4E
#define READ_SCRATCHPAD 0xBE

uint8_t ds_init()
{
    uint8_t ack;
    // Sends reset pulse - digital '0'
    PORTB &= ~(1 << PORTB0);
    DDRB |= (1 << PORTB0);
    _delay_us(480);
    // Releases bus, pull back kicks in
    DDRB &= ~(1 << PORTB0);
    _delay_us(60);
    // Reads acknowledgment from sensor
    ack = PINB & (1 << PORTB0);
    _delay_us(420);
    // if a digital '0' is detected, sensor is detected
    return ack;
}

uint8_t ds_readbit(void)
{
    uint8_t bit = 0;
    // Send low pulse for 1us
    PORTB &= ~(1 << PORTB0);
    DDRB |= (1 << PORTB0);
    _delay_us(1);
    DDRB &= ~(1 << PORTB0);
    _delay_us(14);

    if(PINB & (1 << PORTB0))
    {
        bit = 1;
    }

    _delay_us(45);
    return bit;
```

```
}

void ds_writebit(uint8_t bit)
{
    // Send low pulse for 1us
    PORTB &= ~(1 << PORTB0);
    DDRB |= (1 << PORTB0);
    _delay_us(1);

    // Quickly releases if wants to write logic '1', if not, waits
    if(bit) DDRB &= ~(1 << PORTB0);

    _delay_us(60);
    DDRB &= ~(1 << PORTB0);
}

uint8_t ds_readbyte(void)
{
    uint8_t index = 8, byte = 0;
    // Reads bit and shifts the byte to access next bit
    while(index--)
    {
        byte >>= 1;
        byte |= (ds_readbit() << 7);
    }
    return byte;
}

void ds_writebyte(uint8_t byte)
{
    uint8_t index = 8;
    // Sends 1 bit and bit shifts the byte to access next bit
    while(index--)
    {
        ds_writebit(byte & 1);
        byte >>= 1;
    }
}
#endif /* DS18B20_H_ */
```