

Structure of the Log File

```
-----
### Part 1: Header
1. signature:                               s (c5logger) 8byte
2. version of the log file:                 v          1byte
3. lenght of the accu_info_text:           l          1byte
4. crc of the header(range: s..v..l(10byte)): ch        1byte
```

```
summary: |ssssssss|v|l|ch|
-----
```

```
### Part 2: Accu Info
1. accu info text:                         i          max.256byte
2. crc of the accu_info_text:              ca          1byte
```

```
summary: |iii...iii|ca|
-----
```

```
### Part 3: Measured Data(Records)
(Description of the part 3 is true for version_of_log_file=1)
1. record 1
2. record 2
3. record 3
....
....
n. record n(last)
--end of file--
```

```
Structure of the record:
1. 4byte: time_byte1(low),time_byte2, time_byte3, time_byte4(high byte) -> time, s          ( unsigned long );
2. 2byte: u_low_byte, u_high_byte                                     -> accu voltage, mV      ( unsigned short int);
3. 2byte: i_low_byte, i_high_byte                                     -> accu current, mA     ( signed short int);
4. 4byte: cap_byte1(low),cap_byte2, cap_byte3, cap_byte4(high byte) -> accu capacity, mAh   ( signed long );
5. 1byte: t_accu                                                    -> accu temperature, °C ( signed char );
6. 1byte: crc of the record( range: time_byte_low ... t_accu(13byte))
-----
```

used crc-functions:

```
typedef unsigned char      u8;
#define CRC_INIT 0xDE //Initial CRC value

//crc8 for a byte
void Do_CRC8(u8 b, u8 *crc)
{
    int i;

    for (i = 0; i < 8; i++)
    {
        if ( ((b ^ *crc) & 1) != 0)
            *crc = ((*crc ^ 0x18) >> 1) | 0x80;
        else *crc = (*crc >> 1) & ~0x80;

        b = b >> 1;
    }
}

//crc8 for a buffer
u8 CRC8_buff(void *buff, u8 count)
{
    u8 *p=(u8*)(buff);
    u8 crc=CRC_INIT;

    while(count--)
        Do_CRC8(*p++, &crc);

    return crc;
}
```