Stream Framing Protocol

Philip Munts
Munts Technologies
Revision 1.0, 12 October 2016

Rationale

Stream communication channels, such as serial ports and TCP connections, deliver individual bytes of data. Message boundaries are not necessarily preserved; a sender may write a 20-byte message to the stream while the receiver may read from 1 to 20 bytes at a time from the other end of the stream. When data must be delivered in indivisible messages, a framing protocol such as this **Stream Framing Protocol** must be used.

Delimiters

The following byte values are used for delimiting data frames:

- **STX** (ASCII 2 decimal) Start of Text
- **ETX** (ASCII 3 decimal) End of Text
- **DLE** (ASCII 16 decimal) Data Link Escape

Framing Protocol

1. Each frame shall begin with the two byte sequence **DLE STX**.

2. Any number (including zero) of payload data bytes may follow.

3. An extra **DLE** byte shall be injected ("byte stuffed") before any **DLE** byte present in the payload data.

4. After the payload data bytes, two bytes of **CRC16-CCITT 0x1D0F** checksum shall be appended, in network byte order (most significant byte first). The checksum shall be calculated from the original payload data bytes (*before* byte stuffing).

5. An extra **DLE** byte shall be injected before each byte of the checksum, if it is **DLE**.

6. The frame shall end with the two byte sequence **DLE ETX**.

Buffer Size

When encoding a message of **N** payload data bytes, the destination frame buffer size must be **2*N+8** bytes, worst case (**2*N** if all payload data bytes are **DLE**, **+4** for the **DLE STX** and **DLE ETX** delimiters, and **+4** if both checksum bytes are **DLE**).

Checksum Algorithm

CRC16 algorithms are, in general, incompletely specified. This **Stream Framing Protocol** shall use the following reference implementation (or its equivalent) for **CRC16-CCITT 0x1D0F**:

```
// The following CRC16-CCITT subroutine came from:
// http://stackoverflow.com/questions/10564491/function-to-calculate-
a-crc16-checksum

uint16_t crc16(const uint8_t* data_p, uint8_t length){
    uint8_t x;
    uint16_t crc = 0x1D0F;

    while (length--){
        x = crc >> 8 ^ *data_p++;
        x ^= x>>4;
        crc = (crc << 8) ^ ((uint16_t)(x << 12)) ^ ((uint16_t)(x
<<5)) ^ ((uint16_t)x);
    }
    return crc;
}
```

The checksum for an empty frame (no payload data bytes) must be **0x1D0F**.

The checksum for a message consisting of the nine ASCII bytes "**123456789**" must be **0xE5CC**.

See https://www.lammertbies.nl/comm/info/crc-calculation.html for an online CRC calculator and more information.