Danmeter A/S
Cerebral State Monitor
Model CSM
Communication Protocol
UKS050524 COM Version 03

Document history

Date/Initials	Version	Description of change
17/06-2004 / ABJ	01	Document created
15/02-2005 / ABJ	02	Protocol version 02. Changes in protocol marked with italic. Protocol version is 2.
01/03-2006 / FHI	03	EEG amplitude defined.

1 Introduction

1.1 Scope

This document specifies the layers of the communication to be used for Cerebral State Monitor.

1.2 Definitions of terms

CSM: Cerebral State Monitor
Master: CSM
Slave: Some kind of host computer

1.3 Communication model

The communication model is a point-to-point connection between two communication parties – Master and slave. The protocol does not make provision for the use of intermediary devices (e.g. routers)

As a consequence the following layers in the OSI models is omitted:

- Network Layer
- Transport Layer
- Session Layer
- Presentation Layer

All data is transferred binary. Bit 0 is the least significant and bit 7 is the most significant bit. When an item of data is more than one byte long the LSB is sent first.

2 Physical Layer

Asynchronous serial encoded as:

- 1 Start bit
- 8 Data bits
- 1 Stop bit
- No parity bit

The point-to-point connection is RS232 serial interface at:

- Baud rate: 115200 (fixed)
- Handshake: RTS/CTS

3 Data Link Layer

The purpose of the Data Link Layer is to provide a means by which commands and responses can be packed in a manner that will enable detection of the corruption of their contents when they are received.

3.1 Frame structure

Bytes	Description	Data	See Section
1	SOM	0xFF	
1	COMMAND TYPE		4.1
1	LENGTH	Length n	
N	DATA		4
2	CRC		5
1	EOM	0xFE	

4 Application Layer

4.1 Command Type

Type	Command	Section
0	(Memory Management Applications Only)	NA
1	**CSM on-line data.**	**4.2**
2	(Memory Management Applications Only)	NA
3	(Memory Management Applications Only)	NA
4	(Memory Management Applications Only)	NA
5	(Memory Management Applications Only)	NA
6	(Memory Management Applications Only)	NA
7	(Memory Management Applications Only)	NA
8	(Memory Management Applications Only)	NA
9	(Memory Management Applications Only)	NA
10	(Memory Management Applications Only)	NA
11	(Memory Management Applications Only)	NA
12	(Memory Management Applications Only)	NA

4.2 Command Type 1: CSM on-line data is specified as:

Bytes	Name	Range	Description
4	CSM serial No.	2004210000: 2099219999	
1	Protocol version	1:255	
1	CSI version	1:255	
2	Device time	0:65535	Total time from start in seconds
1	Block Status	Bit field	Bit 0: Artefact Bit 1: Electrode alarm Bit 2: SQI low Bit 3: Impedance high
1	Event number	0:255	
1	Event type	0:8	0 = General event 1 = Induction 2 = Intubation 3 = Maintenance 4 = Surgery 5 = Injection 6 = Note 7 = En maintenance 8 = Movement
1	CSI	0:100, 255	255 = Not defined
1	BS%	0:100, 255	255 = Not defined
1	SQI%	0:100	
1	Black Imp	0:11	0 = “<1” 11 = “>10”
1	White Imp	0:11	0 = “<1” 11 = “>10”
1	EMG (Bar)	0:100, 255	255 = Not defined
1	*Battery Voltage*	*0:255*	*20 * Voltage*
1	*Reserved*	*0:255*	
1	*Alarm high*	*0:255*	*Alarm limit. Bit 7: Alarm on/off*
1	*Alarm low*	*0:255*	*Alarm limit. Bit 7: Alarm on/off*
4	Reserved		
100	EEG	-128:127	Binary EEG data. Signed byte. -180 to +180 uV signal range

On-line data is transmitted as one sequence per second

5 CRC Checking

The CRC uses the standard CCITT generator polynomial $x^{16}+x^{12}+x^5+1$
When checking a CRC the same process as for generation is used. The received CRC bytes are not fed through the generator. Instead the locally calculated CRC is compared with the received CRC. Errors have occurred if these are not the same.
CRC is calculated from TYPE to DATA all included.

CRC Generator algorithm - pseudo “C" code

Below is example “C” code which implements the algorithm for CRC16-CCITT

```
//**************************************************************************
// calc_crc()
//
// Procedure to add character ch to current crc, returns ch unchanged.
//**************************************************************************
char calc_crc(int *crc, char ch)
{
int C = ch & 0x00FF;
int inp;
char i;

  for (i = 0; i < 8; i++)
  {
       if ( (((C & 0x0080) != 0) && ((*crc & 0x8000) == 0)) ||
            (((C & 0x0080) == 0) && ((*crc & 0x8000) != 0)) )
         inp = 0x1021;
        else
         inp = 0;

       *crc <<= 1;
       *crc ^= inp;
       C <<= 1;
  }
  return(ch);
}
```

CRC Generator algorithm - pseudo “Visual Basic" code

```
'*****************************************************************
' Function to calculate new CRC. (Version for Visual Basic)
'
' ENTRY
' crc is a 16 bit integer containing the running crc value.
' ch is an 8 bit charcter to add to the crc
' RETURNS new crc value
'*****************************************************************
Function calc_crc(ByVal ch As Integer, ByVal crc As Long) As Long
C = ch And 255
For i = 0 To 7
       If ((((C And 128) <> 0) And ((crc And 32768) = 0)) Or (((C And 128) = 0)
           And ((crc And 32768) <> 0))) Then
             inp = 4129
       Else
             inp = 0
       End If

       crc = ((crc * 2) Xor inp) And 65535
       C = (C * 2) And 255
Next i
calc_crc = crc
End Function
```