Remarks with respect to the frames as found on the EMSbus

0. Introduction

Starting in December 2018, the EMSbus between a Nefit Trendline II 25HRC, firmware version ?.??, and a Nefit Moduline 400 thermostat, firmware version 2.20, is being monitored, using a Raspberry Pi zero W and a level shifter build by bbqkees. In order to understand the content of the frames passing by over the bus, the documentation, available at URL https://emswiki.thefischer.net/doku.php?id=start, is used. It was found that this documentation is not completely correct and not complete. In this document the changes and additions are noted, as found in the above mentioned configuration.

It is found that there are (at least) 3 versions of the EMS protocol, named EMS, EMS2 and EMSplus. As far as I know, in my configuration only the EMS protocol is used.

1. Protocol summary

The EMS-bus is used to exchange frames between the the devices attached to the EMS-bus. The bus uses asynchronous communication at 9600 [b/s], 8 bit data, no parity, 1 stop bit. The end of a frame is signalled by a break signal, that is an all-zero octet, without stop bit. The break signal, which lasts for at least one octet time, thus at least 10 bits, is not considered to be part of the frame. (Neither are the start and stop bits considered to be part of the frame, they are just layer 1 overhead in the transmission at the physical layer.)

It seems to me that there are basically 6 types of frames:

Name	Size [B]	Remarks
PollRequest	1	Contains device identifier of polled device, with the most significant bit (MSB) of the octet set. If the polled device has something to sent, it can sent it after receiving the poll request.
PollReply	1	Contains the device identifier of the polled device. This frame terminates the poll cycle for the device.
ReadRequest	> 5	The MSB of the second octet, the destination address, is set. It follows the frame format described below.
ReadReply	> 5	The response on a ReadRequest. It follows the frame format described below.
WriteRequest	> 5	It follows the frame format described below. On itself, it is not distinguishable from a ReadReply.
WriteReply	1	Either the value 0x01 (Ack) or 0x04 (Nack).

Moreover, there are two types of ReadReply frames, the solicited and the unsolicited types. The solicited type is sent as a response on the ReadRequest frame. Note that there is no clear way to sent an error status like “unknown item requested” or “requested information is not available”. The destination address in the solicited ReadReply frame is equal to the source address in the preceding ReadRequest frame. The unsolicited type of a ReadReply is (thus) not preceded by a ReadRequest frame, and it’s destination address is (by definition?) 0x00. This address acts as a broadcast address.

The format of the frames with a length of more than 5 octets is:

Offset	Size [B]	Description
0	1	Source device identifier
1	1	Destination device identifier
2	1	Message type
3	1	Initial offset
4	> 0	Data
5	1	Checksum, which is **not** a CRC

The device identifiers are assigned to specific types of devices. For instance the bus-master always uses device identifier 0x08. The following table shows what I think to know about device identifiers:

Range	Description
0x00	Broadcast address
0x01 – 0x07	Reserved. Two of these identifiers are used to signal a successful completion of a WriteRequest (0x01) or an error (0x04)
0x08	Bus master
0x09 – 0x6f	Various devices of various manufacturers
0x70 – 0x7f	Not used in my Nefit system

Each message type corresponds effectively with a table containing a number of fields of varying length. The length might as small as one bit, but also up to three octets. The layout of the table is fixed. Using the offset and the length of a frame a part of the table can be sent or written.

The data-length of a frame is the number of octets after the offset and before the check-sum.

2. Updates

VersionMessage (02): The Moduline 400 seems to broadcast this message a few times a day.

RCTimeMessage (06): The data-length of this message is 13 octets. The last 6 octets are always zero. The year-number equals the value in the octet at data-offset 0 plus 2000. This message is broadcast about once per 60 [s].

RC30StatusMessage (41): The Moduline 400 sends this message once every minute. It contains 14 data-octets. See URL https://github.com/proddy/EMS-ESP-Boiler. Contents of data-octets:

Offset	Length	Divisor	Unit	
0	1			?
1	1	2	C	Set point temperature
2	2	10	C	Current room temperature

4	4			?
8	1		%	Heating relative power ?
9	1		C	Maximum boiler temperature
10	1			?
11	2	10	C	Often the same value as at offset 2
13	1			?

?? (a2): The Moduline sends this message once every 4 minutes. It contains 13 data-octets (as described for message type a3) all of which are always zero. There is no external temperature sensor attached to my system.

RCTempMessage (a3): The Moduline 400 sends it regularly, once per minute. It contains 4 data-octets, always with value ‘00 f6 00 00’.

The value in field (src,typ,off) = (0x08,0x18,9) is identical to the one in field (0x08,0x34,1). It seems to be the temperature of the water leaving the warm water tap.

3. Various

3.0. Polling

Obviously, the bus-master does not send polls to itself. Thus a PollRequest with value 0x88 will never appear on the EMS-bus. Following the same line of reasoning, there is thus never a PollRequest before a frame send by the bus-master, thus source identifier 0x08.

A short scan of the bus revealed that only device identifiers in the range 0x09 up to and including 0x6f appear on the EMS-bus.

3.1. Timing