# BACnet Compact

Joachim Bürmann, *IFTOOLS GmbH*

April 6, 2016

**BACnet is a standardized communication protocol for building automation and control networks. It provides a unified and producer independent way for data exchange between all kinds of building automation devices. BACnet was issued as an American standard in 1995 and has been the ISO standard 16484-5 since 2003.**

## Protocol overview

The communication between several building devices is provided by a number of specified services. There are 'Who is', 'I am', 'Who-have', 'I-have' services for object discovery. Read and Write-Properties serves for data sharing. Upon these services operate so called 'Objects' which covers Analog IOs, Digital IOs, Access objects (door, user, zone), Calendar objects to schedule date depending events and a lot more.
BACnet is based on a four layer protocol stack, the physical layer supports beside others RS232 point-to-point and RS485 Master/Slave token passing. Among them, the MS/TP is the most widely used one.

**Table 1:** *BACnet protocol stack*

| BACnet Application Layer | | | | |
|---|---|---|---|---|
| BACnet Network Layer | | | | |
| ISO 8802-2 Type 1 (IEEE 802.2) | | MS/TP | Dial-Up | LonTalk |
| Ethernet | ARCnet | RS485 | RS232 | |

In this document we mainly cover BACnet over RS485 (MS/TP). But most of this information should be applicable also on RS232 point-to-point connections.

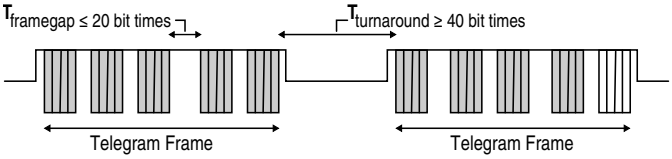
## BACnet MS/TP

BACnet MS/TP is a token passing protocol. Only participants with a token are allowed to sending requests i.e. for data. The receiving node of a request may response without having the token!
This means: A master is always defined as the initiator of a service request whereas a slave only can send responses and isn't permitted to transmit messages without a former invitation.
Since a master only can act with a token, multiple master nodes are possible without to worry about data collisions. In doing so each master passes the token upon task completion to the next known master by sending it a token telegram. The token frame must be acknowledged.
And: A new master can be inserted in the bus on the fly. This is achieved by special 'Poll for Master' messages which we will discuss later. The polling interval depends on the application. BACnet specifies a minimum of every 50 times a master receives/uses a token. In contrary a new slave must be registered on it's responsible master.

### Telegram transmission

Each byte of a telegram is composed by 8 data bits without parity and one start and one stop bit. The telegram bytes must be send in one go. The maximum allowed delay time between two successive bytes of a frame is 20-bit times ($T_{framegap}$). The time between two frames has to be at least 40-bit times ($T_{turnaround}$), measured from the last stop bit to next start bit, see Figure 1.

**Figure 1:** *Message timing*




## Telegram structur

Every MS/TS frame contains of a two octet preamble hex 55 FF, followed by the frame type, destination address, source address, length field, header CRC, optional data and data CRC.

**Table 2:** *BACnet telegram structur without COBS*

| HEADER | | | | | | | | DATA | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Preamble | | Type | Dest | Src | Len | | HCrc | Data | | DCrc | |
| 1. | 2. | 3. | 4. | 5. | 6. | 7. | 8. | 9. | 10. | 11. | 12. |
| 55h | FFh | | | | HB | LB | | ← *Len* → | | HB | LB |

HB: High Byte, LB: Low Byte

| | |
|---|---|
| Preamble | two bytes, hex 55, hex FF |
| Frame Type | one byte, see below |
| Destination address | one byte, hex FF indicates a broadcast message, valid range 0 to 127, 128 to 254 are valid only for slave nodes |
| Source address | one byte, hex FF is not allowed, valid range 0 to 127, 128 to 254 are valid only for slave nodes |
| Length | number of data bytes, two bytes, most significant byte (HB) first, max. 501 for non-encoded frames |
| Header checksum | one byte CRC |
| Data | *present only if length is non-zero and frame is non-encoded type* |
| Data CRC | *present only if length is non-zero and frame is non-encoded type, CRC-16-CCITT* |
| *Encoded data* | *present only, if frame is a COBS-encoded type* |
| *Encoded CRC-32K* | *present only, if frame is a COBS-encoded type, five bytes* |
| *(pad)* | optional, at most a single byte hex FF |

* Please note! COBS (consistent overhead byte coding) is not part of this document.

## Frame types

BACnet defines eight frame types, from 0 to 7. The types 8 to 127 are reserved for specification improvements. The types 128 to 255 are used by vendor specific frames.

- 0 : Token
- 1 : Poll for master
- 2 : Reply to poll for master
- 3 : Test request
- 4 : Test response
- 5 : BACnet data expecting reply
- 6 : BACnet data not expecting reply
- 7 : Reply postponed

Note! Masters must acknowledge or response to the frames 0, 1 and 2, slave must ignore these completely!

**Token (0)** Exchanged between master nodes to pass the token from one master to the next. No data.

**Poll for master (1)** Periodically transmitted by the master nodes during configuration to detect new added (or removed) masters. Masters must response, slaves must ignore it. Doesn't contain data.

**Reply to poll for master (2)** The response of a master to the 'Poll for master' request. It does not present data.

**Test Request (3)** The frame type is used during start of the communication in the MS/TP network and sends particular information to a node.

**Test response (4)** The response to frame type 3.

**BACnet data expecting reply (5)** Used by master nodes to convey data according to the BACnet NDPU (network layer protocol data unit, see below) and waiting for a response.

**BACnet data not expecting reply (6)** Used by master nodes to convey data according to the BACnet NDPU (network protocol data unit, see below) but without waiting for a response.

**Reply postponed** Sent by master nodes after receiving a data expecting reply (5) to indicate a later response.

## BACnet NPDU format

The BACnet NDPU (network layer protocol data unit) forms the container to exchange data between two network participants. It consists of a NPCI (network protocol control information) followed by a NSDU (network service data unit) or APDU (application layer parameters).
The structure of the NPCI is variable and depends on the content of the control byte. The same control byte specifies the meaning of the NSDU.
The BACnet NDPU structur is shown in table 3.

**Table 3:** *BACnet NPDU*

| NPDU | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| NPCI | | | | | | | | | | | NSDU |
| 1 | 1 | 2 | 1 | d | 2 | 1 | s | 1 | 1 | 2 | n |
| Version | Control | DNET | DLEN=d | DADR | SNET | SLEN=s | SADR | Hop Count | Message Type | Vendor ID | APDU |

## Futher links

http://www.bacnetwiki.com/wiki/index.php
http://www.bacnetwiki.com/wiki/index.php?title=PDU_Type
http://www.bacnetinternational.net/btl/index.php
http://sourceforge.net/projects/bacnet/
http://sourceforge.net/projects/bacnet4linux/
http://sourceforge.net/projects/bacpypes/
And here a list of Wireshark capture files:
http://kargs.net/captures/

| | |
|---|---|
| **Version** | Always 1 |
| **Control byte** | |
| Bit 7: | 1 = Message is a network layer message<br>0 = Message contains a BACnet ADPU |
| Bit 6: | reserved |
| Bit 5: | Destination specifier<br>1 = DNET, DLEN, DADR, Hop Count present<br>0 = DNET, DLEN, DADR, Hop Count absent |
| Bit 4: | reserved |
| Bit 3: | Source Specifier<br>1 = SNET, SLEN, SADR present<br>0 = SNET, SLEN, SADR absent |
| Bit 2: | Expecting reply |
| Bit 1,0: | Priority<br>11 = Life safety message<br>10 = Critical equipment message<br>01 = Urgent message<br>00 = Normal message |
| **DNET** | Legal destination network, valid 1-65535<br>0 = illegal, FFFFh = Broadcast |
| **DLEN** | Destination MAC address length<br>DLEN = 0 and DNET = FFFFh means a Global Broadcast<br>DLEN = 0 and DNET != FFFFh means a Remote Broadcast |
| **DADR** | Destination address |
| **SNET** | Legal source network, valid 1-65534<br>0 = illegal, FFFFh = illegal |
| **SLEN** | Source MAC address length |
| **SADR** | Source address |
| **Hop Count** | the maximum number of routers a packet can be routed, default is 255 |
| **Message Type** | Specifies the message type in the following APDU and is divided in an upper and lower nibble. |
| Bit 4...7 | APDU Type<br>0000.... = Confirmed Request<br>0001.... = Unconfirmed Request<br>0010.... = Simple Acknowledge<br>0011.... = Complex Acknowledge<br>0100.... = Segment Acknowledge<br>0101.... = Error<br>0110.... = Reject<br>0111.... = Abort<br>1xxx.... = reserved |
| Bit 3 | 1 = Segment request |
| Bit 2 | 1 = More segments are following |
| Bit 1 | 1 = Segment response not accepted |
| Bit 0 | reserved |
| **Vendor ID** | Only present if bit 7 in the control byte is 1 and the Message Type field contains a value 80h...FFh. |