

## 1. Features

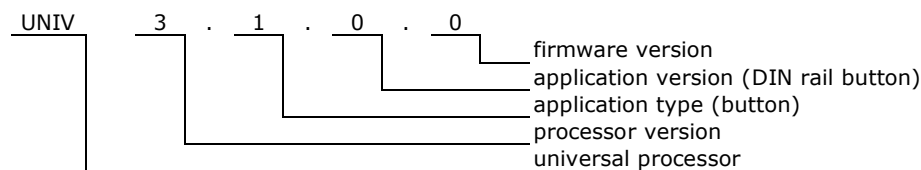
- 8 channel button module.
- 7 types of button behaving is recognized: button pressed, released, pressed for 400ms, pressed for 4s, released within 400ms, released between 400ms and 4s, released after 4s
- There is a 20ms reaction time. Button has to be pressed for at least 20ms to activate module. It avoids contacts bouncing.
- Self-control feature – pressed button can control LEDs in the same module
- Allows writing notes in the processor memory
- Uptime counter
- Health check monitor
- Transmit (42 messages) and receive (42 messages) FIFO buffers



## 2. Compatibility

- Firmware for **UNIV 3.1.0.x** module
- Firmware can be uploaded into processor with bootloader version 3.1 or compatible.

## 3. Firmware version



## 4. Communication Frames (messages)

### 4.1. Button message

Module sends message to the bus, when status of input changes. Module is able to distinguish a few types of button behaving: pressed, released, pressed for 400ms, pressed for 4s, released within 400ms, released between 400ms and 4s, released after 4s. For each situation the unique message is sent to the bus. It is possible to choose for each button separately what messages should be sent.

Table 1. BUTTON frame (0x301)

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x301	3 2 1 0	Node Nr	Group Nr	0xFF	0xFF	CHANNEL	BUTTON	0xFF	0xFF	0xFF	0xFF

0x301	- universal module frame, button										
3	-	- not used flag, read as "0"									
2	-	- not used flag, read as "0"									
1	-	- not used flag, read as "0"									
0	RE	- response flag. Flag is equal "1" if node was requested. If flag is equal „0" it means that status of input has just changed.									

Node Nr - message sender node number  
Group Nr - message sender group number

CHANNEL - input channel 0x01 (button 1) – 0x0D (button 13)

BUTTON - actual input status  
 0x00 – open  
 0xFF – closed  
 0xFE – closed and held for 400ms  
 0xFD – closed and held for 4s  
 0xFC – closed and open within 400ms  
 0xFB – closed and open between 400ms and 4s  
 0xFA – closed and open after 4s

#### 4.2. Status request

Status of module can be checked by sending from computer STATUS REQUEST frame (0x109) (Table 2).

Table 2. STATUS REQUEST frame (0x109).

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x109	0x0	COMP ID1	COMP ID2	0xXX	0xXX	Node Nr	Group Nr	0xXX	0xXX	0xXX	0xXX

**0x1090** - STATUS REQUEST frame

- COMP ID1** - computer identifier (must be unique on the network)
- COMP ID2** - computer identifier (must be unique on the network)
- Node Nr** - node number of requested module
- Group Nr** - group number of requested module
- 0xXX** - inessential data

As response the module will send 8 status frames (Table 3). Meaning of bytes is the same as in Table 1.

Table 3. Response to STATUS REQUEST.

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x301	0x1	Node Nr	Group Nr	0xFF	0xFF	0x01	BUTTON	0xFF	0xFF	0xFF	0xFF

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x301	0x1	Node Nr	Group Nr	0xFF	0xFF	0x02	BUTTON	0xFF	0xFF	0xFF	0xFF

...

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x301	0x1	Node Nr	Group Nr	0xFF	0xFF	0x08	BUTTON	0xFF	0xFF	0xFF	0xFF

#### 4.3. Uptime request

Table 4. UPTIME REQUEST (0x113).

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x113	0x0	COMP ID1	COMP ID2	0xXX	0xXX	Node Nr	Group Nr	0xXX	0xXX	0xXX	0xXX

**0x1130** - UPTIME REQUEST frame

- COMP ID1** - computer identifier (must be unique on the network)
- COMP ID2** - computer identifier (must be unique on the network)
- Node Nr** - node number of requested module
- Group Nr** - group number of requested module
- 0xXX** - inessential data

Table 5. Response to UPTIME REQUEST (0x113).

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x113	0x1	Node Nr	Group Nr	0xFF	0xFF	0xFF	0xFF	UPTIME3	UPTIME2	UPTIME1	UPTIME0

**0x1131** - Response to UPTIME REQUEST frame

- Node Nr** - message sender node number
- Group Nr** - message sender group number
- UPTIME** -  $(UPTIME3 \cdot 256^3 + UPTIME2 \cdot 256^2 + UPTIME1 \cdot 256^1 + UPTIME0 \cdot 256^0)$  in seconds

#### 4.4. Health check request

Table 6. HEALTH CHECK - STATUS REQUEST (0x115).

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x115	0x0	COMP ID1	COMP ID2	0x01	0xXX	Node Nr	Group Nr	0xXX	0xXX	0xXX	0xXX

**0x1150** - HEALTH CHECK REQUEST frame

- COMP ID1** - computer identifier (must be unique on the network)
- COMP ID2** - computer identifier (must be unique on the network)
- 0x01** - status request
- Node Nr** - node number of requested module
- Group Nr** - group number of requested module
- 0xXX** - inessential data

As response the module will send two frames (Table 7).

Table 7. Response to HEALTH CHECK - STATUS REQUEST (0x115).

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x115	0x1	Node Nr	Group Nr	0x01	RXCNT	TXCNT	RXCNTMX	TXCNTMX	CANINTCNT	RXERRCNT	TXERRCNT

**0x1151** - Response to HEALTH CHECK REQUEST frame

**Node Nr** - message sender node number  
**Group Nr** - message sender group number

- 0x01** - frame 1 (current values)
- RXCNT** - current level of receive FIFO buffer
- TXCNT** - current level of transmit FIFO buffer
- RXCNTMX** - maximum level of receive FIFO buffer since power up
- TXCNTMX** - maximum level of transmit FIFO buffer since power up
- CANINTCNT** - number of CAN interface restarts since power up
- RXERRCNT** - current receive errors register
- TXERRCNT** - current transmit errors register

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x115	0x1	Node Nr	Group Nr	0x02	0xFF	0xFF	RXCNTMXE	TXCNTMXE	CANINTCNTE	RXERRCNTE	TXERRCNTE

**0x1151** - Response to HEALTH CHECK REQUEST frame

**Node Nr** - message sender node number  
**Group Nr** - message sender group number

- 0x02** - frame 2 (maximum values saved in eeprom memory)
- RXCNTMXE** - maximum ever level of receive FIFO buffer
- TXCNTMXE** - maximum ever level of transmit FIFO buffer
- CANINTCNTE** - maximum ever number of CAN interface restarts
- RXERRCNTE** - maximum ever receive errors
- TXERRCNTE** - maximum ever transmit errors

To clear maximum values saved in eeprom memory the frame shown in Table 8 must be sent. There is no response to this message.

Table 8. HEALTH CHECK - CLEAR REQUEST (0x115).

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x115	0x0	COMP ID1	COMP ID2	0x02	0xFF	Node Nr	Group Nr	0xFF	0xFF	0xFF	0xFF

**0x1150** - HEALTH CHECK REQUEST frame

**COMP ID1** - computer identifier (must be unique on the network)  
**COMP ID2** - computer identifier (must be unique on the network)

- 0x02** - clear request
- Node Nr** - node number of requested module
- Group Nr** - group number of requested module
- 0xFF** - inessential data

## 5. Module control

There are no control instructions in this firmware.

## 6. Configuration

Parameters that can be configured with this firmware:

- Module identifier (module number and group number);
- Module description (16 chars);
- Button names
- Button settings
- Text notes;

Configuration process can be done using HAPCAN Programmer.

### 6.1. Module identifier

Every module on the network must have unique identifier. The identifier is made of two bytes, module number (1 byte) and group number (1 byte). Identifier of the Ethernet Interface can be changed in HAPCAN Programmer in software settings.

### 6.2. Module description

Every module can have 16 char description, which makes easier for user (programmer) to distinguish nodes.

### 6.3. Button names

Each button can be named with 32 chars.

### 6.4. Button settings

For each button it is possible to configure what type of button behaving is recognized by module. Module can recognize when:

- button is pressed,
- button is released,
- pressed and held for 400ms,
- pressed and held for 4s,
- pressed and released within 400ms (quick click),
- pressed and released between 400ms and 4s,
- pressed and released after 4s.

For each behaving a separate message will be sent on the bus.

**WARNING:** It is very important to choose only messages which will be used on the network to keep traffic on the bus as low as possible.

### 6.5. Text notes.

Up to 1024 characters can be written into processor's memory.

## 7. License



HAPCAN Home Automation Project firmware, Copyright (C) 2013 [hapcan.com](http://hapcan.com)

This program is free firmware: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.



This firmware is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this firmware. If not, see <http://www.gnu.org/licenses/gpl-3.0.html>.

## 8. Document version

File	Note	Date
univ_3-1-0-0a.pdf	Original version	August 2013