1. Features

- 3 channel blind controller firmware.
- 6 control instructions roletami (START, STOP, UP, DOWN, UP/STOP, DOWN/STOP) makes possible to drive blind with one two or three buttons.
- It estimates position of the blind by the running time.
- 3 timers (1 for each channel) 1s-24h for instruction execution delay.
- 3 timers (1 for each channel) 1s-256s defining maximum motor running time.
- 3 blocking instructions
- Allows defining up to 128 CAN messages which can indirectly control the module
- Uptime counter
- Health check monitor
- Transmit (42 messages) and receive (42 messages) FIFO buffers

2. Compatibility

- Firmware for **UNIV 3.7.0.x** module
- Firmware can be uploaded into processor with bootloader version 3.1 or compatible.

3. Firmware version

UNIV 3 . 7 . 0 . 0

- 0 – firmware version
- 0 – application version (blind controller for AC motors)
- 7 – application type (blind controller)
- 3 – processor version
- UNIV – universal processor

4. Communication Frames (messages)

4.1. Blind Controller Frame

The module sends message to the bus, when motors starts running or stops.

Table 1. BLIND CONTROLLER frame (0x307).

Frame type	Flags				Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x307	3	2	1	0	Node Nr	Group Nr	0xFF	0xFF	CHANNEL	STATUS	MOVE	INSTR1	INSTR2	TIMER

0x307 – universal module frame, blind controller application

- 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 output has just changed.

Node Nr - node number on the network

Group Nr - group number of the node on the network

CHANNEL - blind number

STATUS - estimated blind status (0x00-0xFF): 0x00 – blind open, 0xFF – blind closed

MOVE - defines movement: 0x00 – blind stopped, 0x01 – going down, 0x02 – going up

INSTR1 - instruction that is waiting for execution, or 0xFF if none instruction

INSTR2 - second byte of instruction that is waiting for execution, or 0xFF

TIMER - delay value of waiting instruction, or 0x00 if none waiting

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 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
0x307	0x1	Node Nr	Group Nr	0xFF	0xFF	0x01	STATUS	MOVE	INSTR1	0x01	TIMER1
Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x307	0x1	Node Nr	Group Nr	0xFF	0xFF	0x02	STATUS	MOVE	INSTR1	0x02	TIMER2
Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x307	0x1	Node Nr	Group Nr	0xFF	0xFF	0x03	STATUS	MOVE	INSTR1	0x04	TIMER3

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 - node number on the network
Group Nr - group number of the node on the network
UPTIME - (UPTIME3*256^3+UPTIME2*256^2+UPTIME1*256^1+UPTIME3*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 - node number on the network
Group Nr - group number of the node on the network

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 - node number on the network
Group Nr - group number of the node on the network

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	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)

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

5. Module control

The module can be controlled directly from PC, or indirectly by other modules. In both situation all instructions in the table below can be used. Blocking instructions can't be used in direct control.

5.1. Control instruction

The Table 9 shows instructions, which can be executed by the module.

Table 9. Module control instructions

Instruction	Instruction Codding								Note	Control	
	INSTR1	INSTR2	INSTR3	INSTR4	INSTR5	INSTR6	INSTR7	INSTR8		Direct	Indirect
STOP	**0x00**	**CHANNEL**	**TIMER**	0xXX	0xXX	0xXX	0xXX	0xXX	It stops chosen blinds. Executing of the next instruction will be possible only after 1 second.	√	√
UP/STOP	**0x01**	**CHANNEL**	**TIMER**	0xXX	0xXX	0xXX	0xXX	0xXX	It runs blind up, if it was stopped. It stops blind, if it was running.		
DOWN/STOP	**0x02**	**CHANNEL**	**TIMER**	0xXX	0xXX	0xXX	0xXX	0xXX	It runs blind down, if it was stopped. It stops blind, if it was running.		
UP	**0x03**	**CHANNEL**	**TIMER**	0xXX	0xXX	0xXX	0xXX	0xXX	It runs blind up, if was stopped. It stops, and after 1s runs blind up, if was running down.		
DOWN	**0x04**	**CHANNEL**	**TIMER**	0xXX	0xXX	0xXX	0xXX	0xXX	It runs blind down, if was stopped. It stops, and after 1s runs blind down, if was running up.		
START	**0x05**	**CHANNEL**	**TIMER**	0xXX	0xXX	0xXX	0xXX	0xXX	It stops blind if was running. It runs blind opposite way to last running, if blind was stopped.		

0xXX – inessential data

CHANNEL	Note
0x01	- <00000001> - only blind 1
0x02	- <00000010> - only blind 2
0x03	- <00000011> - blind 1 & 2
0x04	- <00000100> - only blind 3
...	
0x07	- <00000111> - blind 1, 2, 3

bit <0> - blind 1
bit <1> - blind 2
bit <2> - blind 3

TIMER	Note
0x00	- instruction will be executed immediately
0x01	- instruction will be executed with 1s delay
...	
0xFF	- instruction will be executed with 24h delay

ENABLE BOX	**0xDD**	**BoxX**	**BoxY**	0xXX	0xXX	0xXX	0xXX	0xXX	It enables chosen boxes – these boxes will be compared with next received message from the bus.		√
DISABLE BOX	**0xDE**	**BoxX**	**BoxY**	0xXX	0xXX	0xXX	0xXX	0xXX	It disables chosen boxes – these boxes will be passed when next message arrives from the bus.		√
TOGGLE BOX	**0xDF**	**BoxX**	**BoxY**	0xXX	0xXX	0xXX	0xXX	0xXX	It toggles boxes – enables when they are disabled and vice versa		√

0xXX – inessential data

BoxX	Note
0x00	- from Box 1
0x01	- from Box 2
...	
0x7F	- from Box 128

BoxY	Note
0x00	+ 0 -(and not anyone else)
0x01	+ 1 -(and 1 following)
...	
0x7F	+127 -(and 127 following)

5.2. Timer

Each relay has its own timer which can delay execution of the instruction. Delay can be chosen between 1s-24h set in TIMER register. Drawing below shows delay dependence of TIMER register.

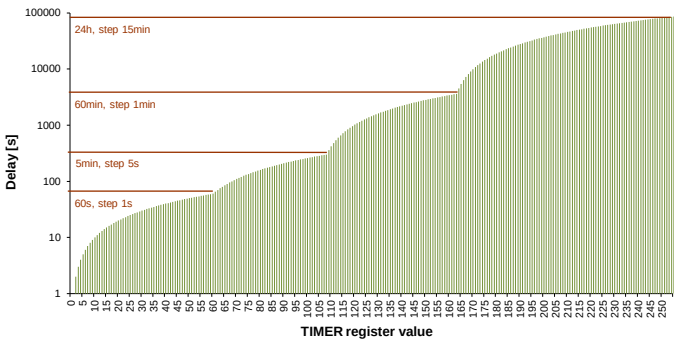

Figure 1. Delay/timer register relationship

5.3. Direct control

It is possible to control module by sending DIRECT CONTROL message. The message contains instruction, which will be executed by the module. The module can be also controlled from HAPCAN Programmer.

Table 10. DIRECT CONTROL frame (0x10A).

Frame type	Flags	Module	Group	D0	D1	D2	D3	D4	D5	D6	D7
0x10A	0x0	COMP ID1	COMP ID2	INSTR1	INSTR2	Node Nr	Group Nr	INSTR3	INSTR4	INSTR5	INSTR6

0x10A – DIRECT CONTROL 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

INSTR1-6 - instruction to be executed (byte1)

5.4. Indirect control

Indirect control means that module will react to messages sent by other modules on the network. It depends on configuration programmed into the module boxes (memory cells).
This firmware has feature to set simple conditions of executing instruction. To do so, you can use blocking instruction (0xDD – 0xDF) shown in Table 9.

6. Configuration

Parameters that can be configured with this firmware:
- Module identifier (module number and group number);
- Module description (16 chars);
- Blind names;
- Blind running times;
- Text notes;
- Linking device with other modules (indirect control of module).

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. Relay Names

Each relay can be named with 32 chars.

6.4. Blind running time

For every channel (motor) the running time can be chosen. It is the time needed for complete closing of the blind from complete open position, or complete opening from complete shutting, whichever is longer. Basing on this time the module will estimate position of the blind.

6.5. Text notes.

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

6.6. Linking devices

The module has 128 memory cells (boxes). Each box can contain information about message sent by other node, and instruction which will be executed when that message is received.

7. License

8. Document version

File	Note	Date
univ_3-7-0-0a.pdf	Original version	April 2014