Miniature World Control

OC32

Interface Specification

Author: Leon J.A. van Perlo
Version: 0.0.2.0
Date: Aug 29th, 2011

Release Management

This manual applies to the kit consisting of:

- Print
  - OC32 Rev00 Nov2009
  - OC32 Rev01 Apr2010
  - OC32 Rev02 Oct2010

- OC32 firmware
  - Version 0.0.2.0

Changes with respect to version 0.0.1.0 are in red

Contents

1 Introduction

The OC32 is intended to be the successor of the OM32. In order to be able to operate both types of modules on the same bus in the same system, interoperability is preserved.

The OC32 has 3 communication interfaces:
- RS485 (bi-directional). This interface can be used
  - to update the system-firmware
  - to configure the module
  - for regular operation.
- RS232/TTL (receive only). This interface can be used
  - for regular operation
  - to configure the module with some limitations
- DCC (receive only). This interface can be used
  - for regular operation in a DCC compatible system

The message formats on the RS485 and RS232/TLL interfaces are identical with the restriction that bi-directional communication can not be used with RS232/TTL
Communication on these interfaces is serial async, 8 bits, odd parity, one stop bit. Communication speed is 9.600, 19.200 or 38.400 bps. The module is auto-baud. This means it determines the correct bitrate from the received datastream. In order to minimize start-up errors the initial (start-up) bitrate can be configured by the user.

Message format for the OC32 RS485 and RS232/TTL interfaces can be in OC32 or OM32 format. The protocol is designed so that OM32 and OC32 modules can co-exist on a single network, even when OC32 communication is used for the OC32 modules.
OM32 format can be used when the controlling system supports OM32 format but no OC32 format.

Addressing capacity is 16 modules per communication channel in OM32 mode. If OC32-eXtended Addressing mode is used up to 1776 modules could be addressed

To update, configure and operate the module(s) from a PC-USB interface a USB-RS485 converter is available (VPEB-U485).

2 OM32-mode communication

2.1 General

OM32 mode communication is available on the RS485 and RS232/TTL interfaces

(….) means a byte(octet) in the message
[(….)] means an optional byte in the message
{(….)} means zero or more bytes in the message

2.2 Message structure

Every message has exactly 5 bytes:
(addressbyte) (commandbyte) (output) (parameter) (checksum)

2.2.1 Addressbyte

(addressbyte) has the form (0 R AAAA 11)

- Bit 7 (0) is the indication that it is an addressbyte (start of message)
- Bit 6 (R) is the retransmission-bit (1 for retransmission, 0 for normal message).
- Bit 5..2 (AAAA) are the addressbits by which 16 OM32/OC32 modules can be addressed.
- Bit 1..0 (11) is the indication that it is an OM32 or OC32 type message.

2.2.2 Commandbyte

(commandbyte) has the form 10 CCCCCC

- Bit 7 (1) is the indication that it is not an addressbyte
- Bit 6 (0) is the indication that it is an OM32-mode message
- Bit 5..0 (CCCCCC) contains the (6-bits) command

2.2.3 Output

(output) has the form (100 UUUUU)

- Bit 7 (1) is the indication that it is not an addressbyte
- Bit 4..0 (UUUUU) contains the output number (0..31)

2.2.4 Parameter

(parameter has the form (1 PPPPPPP)

- Bit 7 (1) is the indication that it is not an addressbyte
- Bit 6..0 (PPPPPPP) contains the (7-bits) data

2.2.5 Checksum

(checksum) has the form (1 SSSSSSS)

- Bit 7 (1) is the indication that it is not an addressbyte
- Bit 6..0 (SSSSSSS) contains the (7-bits) checksum. The checksum byte has a value so that the sum of all bytes in the message is a multiple of 128.

2.3 OM32 message content

Instruction	*Parameter*	*Description*	*Comment*
000000		Reserved	
000001	Aspect	Set Aspect	OC32 only
000010		System Reset	
0F0011	Type	NS signal green-to-red-via-yellow	
0F0100	Level/Delay	Set Level Logarithmic with Delay	
0F0101	Level/Delay	Set Level Linear with Delay	
0F0110	Time	Blink Alternating	
000111	Source	Copy <source> to output	
0F1000	Time	Pulse Low	tijd 0 = Permanent laag
0F1001	Time	Pulse High	tijd 0 = Permanent hoog
0F1010	Time	Blink	tijd 0 = Inverteer uitgang
0F1011	Time	Random on/off	tijd 0 = Eenmalig
0F1100	Pattern	Multibit 2	bit 5/6 van patroon = opties
0F1101	Pattern	Multibit 3	bit 5/6 van patroon = opties
0F1110	Pattern	Multibit 4	bit 5/6 van patroon = opties
0F1111	Pattern	Multibit 5	bit 5/6 van patroon = opties
1F0000	Pattern	Multibit 6	
1F0001	Pattern	Multibit 7	
1F0010	Pattern	Multibit 6, all outputs off first	
1F0011	Pattern	Multibit 7, all outputs off first	
1F0100	Pattern	Multibit 6, no dark phase	
1F0101	Pattern	Multibit 7, no dark phase	
1F0110	Level	PWM Level (0..31)	OC32 only
1F0111	Position	Servo Position (0..127)	OC32 only
1F1000	Level/Time	Pulse Level Logarithmic	
1F1001	Level/Time	Pulse Level Linear	
1F1010	Time	Multiblink 2	
1F1011	Time	Multiblink 3	
1F1100	Time	Multiblink 4	
1F1101	Time	Multiblink 5	
1F1110	Time	Multiblink 6	
1F1111	Time	Multiblink 7	

Table 4: OM32 Command overview (release 1.4)

Explanation of Commands

- F = Fade. If this bit in set in the Instruction-byte, outputs are gradually switched on/off to simulate the “glowing” of incandescent lamps. Default timing is optimised for LEDs.
- Set Aspect: OC32 Only (release > 0.0.0.8). Executes the Apect stored in the OC32 for Pin. Aspect = 0..11
- NS signal G->Y->R: Switches a signal to Red. If the signal is Green, it will show Yellow in between for a short period of time (approx 0,5 sec). This is a typical behaviour of Dutch signals. Type = 0 for signals with 3 outputs, Type = 1 for signals with 4 outputs (+ number sign).
- Set Level with Delay. New Level = 0..15. 0 = off, 15 = 100%, values in between show a level according to a lineair or logarithmic characteristic.
  Note: Lin/Log characteristic is only applicable when the Pin is NOT explicitly configured.
  Note: There is NO fading with this command. If you want fading, use the PWM Level cmd. It is possible to set a time-delay. The delay is part of the parameter and is determined by bits 4, 5 and 6. An overview which parameter results in which level/delay can be found in table 4.
  Note: Although this command does not support fading, the F-bit as a function: It doubles the delay. Therefore the delay can be 8 seconds max.
- Blink Alternating: Alternating Blink for 2 consecutive Pins. Alternating means: If one Pin is on, the other is off and vice versa. Time <tijd> = half period in units of 1/60 sec with a maximum of 127.

- Clone: Copies the behaviour of one Pin exactly to another <source> Is the Pin from which is copied.
  Note: No Fading (is copied-along)
  Note: The actual state of the Pin is NOT copied. Therefore the copy is synchronous from the first "event" onwards.'
- Pulse Low: Pulse "low" during <time> in units of 1/60 sec with a maximum of 127. <time> = 0 means "low infinitely"
- Pulse High: Pulse "high" during <time> in units of 1/60 sec with a maximum of 127. <time> = 0 means "high infinitely"
- Invert: Invert Pin after every <time> in units of 1/60 sec with a maximum of 127. <time> = 0 means "Invert Once"
- Random: Gives Pin a random state (on or off) every <time> in units of 1/60 sec with a maximum of 127. <time> = 0 means "Randomize Once"
- Multibit (n): Sets a pattern on (n) consecutive Pins, starting at the specified Pin, according to the pattern given by <pattern>. The least significant bit of <pattern> represents the first Pin (i.o.w: bits 0 to n-1 in <pattern> are processed from right to left for the consecutive outputs).
  The mechanism works modulo 32, so a Multibit 3 on Pin 31 sets Pins 31, 0 and 1 according to <pattern>.
  If the "F" option is set, all Pins that shall be off are cleared with fading-effect, after 1/3 second all Pis that shall be on are set with fading-effect.
  In the Multibit 2 to 5 command, bits 5 and 6 of <pattern> are not used for the pattern but have an additional function:
  - Bit 5 = 1: The "image" is fully cleared, including the Pins that need to remain on.
  - Bit 6 = 1: The "image" is set without an additional delay (so there is no dark-phase).
  At the Multibit 6 and 7 these bits are needed for pattern-definition, so the options are part of the command itself.
- PWM Level: Can be used to set the PWM level of a Pin. This command is in addition to the (old OM32) Set Level and Pulse Level commands, since these have only 16 levels, while this (PWM Level) has 32 levels. The F-bit gives dynamic behaviour as defined by the OC32 Pin Configuration.
- Set Servo: Moves the Servo to <position>. Will only work if the Pin is configured as Servo. The F-bit gives dynamic behaviour as defined by the OC32 Pin Configuration.
- Level-Pulse: Works similar as Set Level with Delay. However, the new level is set immediately and after the delay the Pin reverts to the level it had just before the command. <time> = 0 means infinitely (so a "simple" Set Level).
- Multiblink (n): Tests the (n) consecutive Pins if these are on or off. If the Pin is on, it will be blinked with the given interval in units of 1/60 sec (max 127). If the Pin is of fit wil stay off. The function is primarily used to blink Pins simultaneousy, as is the case with Belgium signals.
  Note: If a Pin has an active timer it is considered "on". Also Pins with a level other than 0 are considered "on".

Parameter	Niveau Lineair	Niveau Log	Vertraging (sec)	Parameter	Niveau Lineair	Niveau Log	Vertraging (sec)
0	0%	0%	-	64	0%	0%	1,4
1	7%	2%	-	65	7%	2%	1,4
2	13%	3%	-	66	13%	3%	1,4
3	21%	5%	-	67	21%	5%	1,4
4	27%	7%	-	68	27%	7%	1,4
5	33%	10%	-	69	33%	10%	1,4
6	41%	13%	-	70	41%	13%	1,4
7	46%	17%	-	71	46%	17%	1,4
8	50%	24%	-	72	50%	24%	1,4
9	56%	30%	-	73	56%	30%	1,4
10	62%	37%	-	74	62%	37%	1,4
11	68%	46%	-	75	68%	46%	1,4
12	75%	56%	-	76	75%	56%	1,4
13	83%	68%	-	77	83%	68%	1,4
14	91%	83%	-	78	91%	83%	1,4
15	100%	100%	-	79	100%	100%	1,4
16	0%	0%	0,5	80	0%	0%	2,0
17	7%	2%	0,5	81	7%	2%	2,0
18	13%	3%	0,5	82	13%	3%	2,0
19	21%	5%	0,5	83	21%	5%	2,0
20	27%	7%	0,5	84	27%	7%	2,0
21	33%	10%	0,5	85	33%	10%	2,0
22	41%	13%	0,5	86	41%	13%	2,0
23	46%	17%	0,5	87	46%	17%	2,0
24	50%	24%	0,5	88	50%	24%	2,0
25	56%	30%	0,5	89	56%	30%	2,0
26	62%	37%	0,5	90	62%	37%	2,0
27	68%	46%	0,5	91	68%	46%	2,0
28	75%	56%	0,5	92	75%	56%	2,0
29	83%	68%	0,5	93	83%	68%	2,0
30	91%	83%	0,5	94	91%	83%	2,0
31	100%	100%	0,5	95	100%	100%	2,0
32	0%	0%	0,7	96	0%	0%	2,8
33	7%	2%	0,7	97	7%	2%	2,8
34	13%	3%	0,7	98	13%	3%	2,8
35	21%	5%	0,7	99	21%	5%	2,8
36	27%	7%	0,7	100	27%	7%	2,8
37	33%	10%	0,7	101	33%	10%	2,8
38	41%	13%	0,7	102	41%	13%	2,8
39	46%	17%	0,7	103	46%	17%	2,8
40	50%	24%	0,7	104	50%	24%	2,8
41	56%	30%	0,7	105	56%	30%	2,8
42	62%	37%	0,7	106	62%	37%	2,8
43	68%	46%	0,7	107	68%	46%	2,8
44	75%	56%	0,7	108	75%	56%	2,8
45	83%	68%	0,7	109	83%	68%	2,8
46	91%	83%	0,7	110	91%	83%	2,8
47	100%	100%	0,7	111	100%	100%	2,8
48	0%	0%	1,0	112	0%	0%	4,0
49	7%	2%	1,0	113	7%	2%	4,0
50	13%	3%	1,0	114	13%	3%	4,0
51	21%	5%	1,0	115	21%	5%	4,0
52	27%	7%	1,0	116	27%	7%	4,0
53	33%	10%	1,0	117	33%	10%	4,0
54	41%	13%	1,0	118	41%	13%	4,0
55	46%	17%	1,0	119	46%	17%	4,0
56	50%	24%	1,0	120	50%	24%	4,0
57	56%	30%	1,0	121	56%	30%	4,0
58	62%	37%	1,0	122	62%	37%	4,0
59	68%	46%	1,0	123	68%	46%	4,0
60	75%	56%	1,0	124	75%	56%	4,0
61	83%	68%	1,0	125	83%	68%	4,0
62	91%	83%	1,0	126	91%	83%	4,0
63	100%	100%	1,0	127	100%	100%	4,0

3 OC32-mode communication

3.1 General

OC32 mode communication is available on the RS485 and RS232/TTL interfaces

(….) means a byte(octet) in the message
[(….)] means an optional byte in the message
{(….)} means zero or more bytes in the message

3.2 Message structure

A message has a variable length of form
(addressbyte) (headerbyte) [(extended-address)] {(data)} (checksum)

3.2.1 Addressbyte

(addressbyte) has the form (0 T AAAA 11)

- Bit 7 (0) is the indication that it is an addressbyte (start of message)
- Bit 6 (T) is the toggle-bit. It keeps the previous value during retransmission and is flipped every time a new message is transmitted. [1]
- Bit 5..2 (AAAA) are the addressbits by which 16 OC32 modules can be addressed.
- Bit 1..0 (11) is the indication that it is an OM32 or OC32 type message.

3.2.2 Headerbyte

(headerbyte) has the form 11 BX LLLL

- Bit 7 (1) is the indication that it is not an addressbyte
- Bit 6 (1) is the indication that it is an OC32-mode message.[2]
- Bit 5 indicates if the communication mode is bi-directional (1) or unidirectional (0)[3]
- Bit 4 indicates if eXtended Addressing is used
- Bit 3..0 (LLLL) contains the number of data bytes (0..15) in the message.[4]

3.2.3 eXtended Address (not yet implemented)

(extended-address) (if present) has the form 1 XXXXXXX
If eXtended Addressing is used, the address in the Address byte is considered to be a channel number. The eXtended Address is the address within that channel.

- Bit 7 (1) is the indication that it is not an addressbyte
- If XXXXXXX = 0, the message is considered a channel-broadcast
- If XXXXXXX = 1 .. 111 (decimal), the message is considered to be an individual message

[1] This mechanism is identical to the mechanism used for e.g. TM-H and PM32 communication. Note that when unidirectional communication is used, this mechanism can be used for retransmission. When 2 consecutive messages have identical address and header bytes, it shall be considered as a retransmission and the included command shall not be executed. In this way an unlimited number of retransmissions can be done.
[2] Note that the meaning of bit 6 in the address byte depends on this bit.
[3] Only OC32-type messages can be bidirectional. Note that only the RS485 interface supports bidirectional communication. If an OC32 message with active bidirectional-bit is received on the RS232/TTL interface, no response is sent by the OC32.
[4] Note that the total number of bytes (LLLL) in a message is the number of databytes + 3 if no extended addressing is used and the number of databytes + 4 if extended addressing is used.

- If XXXXXXX = 112..127 (decimal), the message is considered to be a group message

Note: broadcast and group messages are always unidirectional and shall never be replied to by any module, irrespective of the "B" bit in the header byte

3.2.4 Data

(data) has the form (1 DDDDDDD)

- Bit 7 (1) is the indication that it is not an addressbyte
- Bit 6..0 (DDDDDDD) contains the (7-bits) data

3.2.5 Checksum

(checksum) has the form (1 SSSSSSS)

- Bit 7 (1) is the indication that it is not an addressbyte
- Bit 6..0 (SSSSSSS) contains the (7-bits) checksum. The checksum byte has a value so that the sum of all bytes in the message is a multiple of 128.

3.3 Bidirectional Communication

If the OC32 receives an OC32-type message with bidirectional-bit set on the RS485 interface it will respond with a response-message. The response-message will have the same structure as the originating message.

If the message received by the OC32 contains an information-request, the OC32 response following the request will contain the requested data.

If the received message does not contain an information-request, the OC32 response can be either a "poll" message (see below) or an "event" message. Events are not supported yet, but the Host shall be prepared to receive any message that has a valid OC32 message format.

3.4 OC32 Message Content

If there are no databytes, the message is considered an "idle" or "poll" message.
If there is at least one databyte, the first databyte is considered to be an instruction byte. The following instructions are defined:

3.4.1 Operational Commands (0..63)

0 Next 3 databytes contain an OM32 type instruction (command, output, parameter)

Data format :
(0) (OM32-Command) (OM32-Output) (OM32-Parameter)

1..63 Not defined

3.4.2 Configuration Commands (64..127)

64..79 (0x40..0x4F): Set Temporary Configuration (RAM)
64 (0x40) System Configuration (not yet implemented)
65 (0x41) Pin configuration

80..95 (0x50..0x5F): Get Temporary Configuration (RAM)
80 (0x50) System Configuration (not yet implemented)
81 (0x51) Pin configuration

96..111 (0x60..0x6F): Write Permanent Configuration (FLASH)
96 (0x60) System Configuration
97 (0x61) Pin configuration
98 (0x62) Aspect/Sequence mode
99 (0x63) Aspect/Sequence configuration
100 (0x64) Init/Input Configuration

112..127 (0x70..0x7F): Read Permanent Configuration (FLASH)
112 (0x70) System Configuration
113 (0x71) Pin configuration
114 (0x72) Aspect/Sequence mode
115 (0x73) Aspect/Sequence configuration
116 (0x74) Init/Input Configuration

3.4.3 System Configuration

3.4.3.1 Write System Configuration (Command 96)

The instruction-byte is directly followed by the variable number (0..127) to be configured. After the variable number 0 to 12 bytes follow:

Variable = 1: Hardware Configuration

After the variable number 4 bytes follow

Byte 1: Configuration Pin 0.. 7 (0..3)
Byte 2: Configuration Pin 8..15 (0..3)
Byte 3: Configuration Pin 16..23 (0..3)
Byte 4: Configuration Pin 24..31 (0..3)

Configuration = 0: No driver installed (resistor bank)
Configuration = 1: Sink driver installed
Configuration = 2: Source driver installed
Configuration = 3: Sink and source driver installed

Variable = 4: DCC address

After the variable number 2 bytes follow

Byte 1: DCC Low-address (0..63)
Byte 2: DCC High-address (0..7)

Variable = 10: ID string

After the variable number 1 to 12 bytes follow

Byte n: ASCII value of string, 0x00 = end-of-string

3.4.3.2 Read System Configuration (Command 112)

The instruction-byte is directly followed by the variable number (0..127) to be read.
The OC32 responds with a message which is identical to the Read System Configuration request, however, after the variable-number 0 to 12 bytes follow. The content is per variable as described above under " Write System Configuration" with one addition:

Variable = 0: Software Version Number

After the variable number 4 bytes follow

Byte 1: Version Number byte 1 (0..127)
Byte 2: Version Number byte 2 (0..127)
Byte 3: Version Number byte 3 (0..127)
Byte 4: Version Number byte 4 (0..127)

3.4.4 Pin Configuration

3.4.4.1 Set/Write Pin Configuration (Command 65 or 97)

The instruction-byte is directly followed by the pin number (0..31) to be configured.

After the pin number 1 to 12 bytes follow

Byte 1:
0b0000 Configuration Cleared
0b0001 Servo Mode
0b001x PWM Mode
x = 0 Normal PWM mode
x = 1 Inverted PWM mode
0b01xx Input Mode (Not Yet Implemented)
0b1xxx Reserved

Data format :
(0x41) (Pin) (Configuration) {(Configuration-data)}
(0x61) (Pin) (Configuration) {(Configuration-data)}

If the base configuration is changed, e.g. from servo to PWM, the configuration array is cleared before the remaining bytes in the configuration data stream are written.

If new configuration = 0b0000, the configuration array is cleared, irrespective of the configuration data to follow

3.4.4.2 Get/Read Pin Configuration (Command 81 or 113)

The instruction-byte is directly followed by the pin number (0..31) to be read.

Data format :
(0x51) (Pin)
(0x71) (Pin)

The OC32 responds with a message which is identical to the get/read request, however, after the pin-number 1 to 12 bytes follow. The number of bytes is determined by the OC32 based on the active or saved configuration (servo, PWM or cleared). The host shall be prepared to receive up to 12 configuration-bytes following the pin-number.

Data response format :
(0x51) (Pin) (Configuration) {(Configuration-data)}
(0x71) (Pin) (Configuration) {(Configuration-data)}

3.4.4.3 Servo Configuration

If new configuration = 0b0001 (Servo):
Byte 2 = Setpoint (-64..63) + 64 (so byte 2 is in range 0..127)
Byte 3 = Range multiplier
0 = Off
1 = Small range
2 = Medium range
3 = Large range
4 = eXtra Large range
16 = Small range, delayed
32 = Medium range, delayed
48 = Large range, delayed
64 = eXtra Large range, delayed
Byte 4 = Midpoint offset (-64..63) + 64 (so byte 4 is in range 0..127)
Byte 5 = AntiMass (0..127)
Byte 6 = Friction (0..127)
Byte 7 = Max Speed (1..127)
Byte 8 = Max Acceleration (1..127)
Byte 9 = BounceDown (0..63) + 64 (so byte 9 is in range 64..127)
Byte 10 = BounceDownFactor (0..127)
Byte 11 = BounceUp (-64..0) + 64 (so byte 11 is in range 0..64)
Byte 12 = BounceUpFactor (0..127)

If there are less than 12 bytes in the configuration data, the configuration bytes which are missing are not affected.

3.4.4.4 PWM Configuration

If new configuration = 0b001x (PWM):
Byte 2 = Setpoint (0..31)
Byte 3 = Behaviour
.0 = 0: Logarithmic 1: Linear
.1 = 0: Log.Accelleration 1: Linear.Accelleration
.2 = Freeze accelerator as set in Byte 4
Byte 4 = Accelerator (0..31)
Byte 5 = “Off” Level (0..31) for Random operations
Byte 6 = “On” Level (0..31) for Random operations

If configuration = 0b0000 and a PWM operational command is received, the pin is put in **implicit** PWM mode (configuration 0b0010). Lin/Log behaviour may be set by the PWM command in that case and the accelerator is set by the PWM command. Note this should be close to 100% OM32 imitation.

If configuration is **explicitly** set to PWM (so by setting the configuration to 0b001x), Lin/Log behaviour is defined by byte 3 and will NOT be affected by any PWM command. If you want another acceleration than default OM32, set Byte 4 and set bit 2 in the Behaviour Byte.

3.4.5 Aspect~~/Sequence~~ Configuration

3.4.5.1 Write ~~A/S~~ Aspect Mode (Command 98)

The instruction-byte is directly followed by the pin number (0..31) to be configured.

After the pin number 1 byte follows with the following meaning

0b0000	A/S Configuration Cleared
~~0b0001~~	~~Sequence Mode, 4 programs of 1 ÷ 8 steps each (not yet implemented)~~[5]
0b0010	12-Aspect Mode, 12 aspects of 3 instructions each
0b0011	4-Aspect Mode, 4 aspects of 9 instructions each

Data format :
(0x62) (Pin) (Configuration)

Setting the A/S mode will CLEAR/ERASE the entire array of instructions/steps for the corresponding pin, irrespective whether the mode is actually changed or not.

3.4.5.2 Read ~~A/S~~ Aspect Mode (Command 114)

The instruction-byte is directly followed by the pin number (0..31) to be configured.

Data format :
(0x72) (Pin)

The OC32 responds with a message which is identical to the get/read request, however, after the pin-number 1 byte follows containing the A/S mode as described above.

Data response format :
(0x72) (Pin) (Configuration)

3.4.5.3 Write Aspect Instruction (Command 99)

~~Note that the Write/Read Aspect Instruction and Write/Read Sequence Program Step messages are identical. Which type of message it is is determined by the A/S mode.~~

The instruction-byte is directly followed by the pin number (0..31) to be configured.

[5] Sequence mode will not be implemented (at least for the time being). Instead these functions are performed by special Sequence-Instructions (more flexibility in operations).

After the pin-number 2 to 7 bytes follow:

Byte 1 = Aspect number (0..12)
Byte 2 = Instruction number (0..8)
Byte 3 = Ox32 (extended) instruction (0..127) (optional) [6]
Byte 4 = Parameter 0 (optional)
Byte 5 = Parameter 1 (optional)
Byte 6 = Parameter 2 (optional)
Byte 7 = Parameter 3 (optional)

If an optional byte is not send, but is relevant for the respective Ox32 instruction, it is assumed to be 0

Data format:
(0x63) (Pin) (Aspect) (Instruction-number) {(Instruction-data)}

Note: Setting an invalid aspect/instruction number leads to an undefined result, however, only the settings for the pin number being set/written will be affected. So when the aspect-mode is 0b0011 (4-Aspect Mode) the result of setting aspect 5, instruction 1 is undefined.

~~**3.4.5.4 Write Sequence Program Step (Command 99) (not yet implemented!)**~~

~~Note that the Write/Read Aspect Instruction and Write/Read Sequence Program Step messages are identical. Which type of message it is is determined by the A/S-mode.~~

~~The instruction-byte is directly followed by the pin number (0..31) to be configured.~~

~~After the pin-number 4 to 7 bytes follow:~~

~~Byte 1 = Program number (0..3)~~
~~Byte 2 = Step number (0..8, 0=init, 1 is the first cyclic step)~~
~~Byte 3 = Step-instruction (0..127)~~
~~Byte 4 = Parameter 1~~
~~Byte 5 = Parameter 2~~
~~Byte 6 = Time (0..63)~~
~~Byte 7 = TimeBase (0..3)~~

3.4.5.5 Read Aspect~~/Sequence~~ Instruction~~/Program~~ (Command 115)

The instruction-byte is directly followed by the pin number (0..31) to be read.

After the pin-number 2 bytes follow:

Byte 1 = Aspect number (0..12)
Byte 2 = Instruction number (0..8)

Data format:
(0x73) (Pin) (Aspect) (Instruction-number)

The OC32 responds with a message which is identical to the get/read request, however, after the pin-number 2 to 7 bytes follows containing the Aspect Instruction or Sequence Program Step as described above.

[6] See paragraph 3.5

Data format:
(0x73) (Pin) (Aspect) (Instruction-number) {(Instruction-data)}

3.4.6 Event Configuration

3.4.6.1 Write Event Configuration (Command 100)

The instruction-byte is directly followed by the pin number (0..31) to be configured.

After the pin number 1 to 5 bytes follow with the following structure

0b yyy zzzz yyy = Event to be updated
zzzz = Action on Event

yyy = 0 Event = Input 0
yyy = 1 Event = Input 1
yyy = 2 Event = Input 2
yyy = 3 Event = Input 3
yyy = 7 Event = Startup
zzzz = N Set Aspect N (N = 0..11)
zzzz = 15 No action

Data format :
(0x64) (Pin) {(EventConfiguration)}

3.4.6.2 Read Event Configuration (Command 116)

The instruction-byte is directly followed by the pin number (0..31) to be read.

Data format :
(0x74) (Pin)

The OC32 responds with a message which is identical to the read request, however, after the pin-number 5 bytes follow containing the Event actions with the structure as described above.

Data format :
(0x74) (Pin) (StartupConfig) (Inp0Config) (Inp1Config) (Inp2Config) (Inp3Config)

3.5 OC32 (extended) Instructions

The OC32 instructions are an extension to the standard OM32 command set.

Parameter 0 (Byte 4) in most cases represents the Pin-Offset (0..31). It determines on which Pin the instruction operates. Note that this differs from the standard OM32 instruction set. The standard OM32 instruction points to a Pin, while Parameter points to a Pin relative to the Pin under which the instruction is stored (the originating Pin number). Using an offset rather than a direct addressing makes it easier to build a standard instruction set for devices without having to know the originating Pin number. The offset is modulo 32. So an offset of 31 wraps around to -1. Of course only outputs of the same OC32 can be controlled!

3.5.1 OC32 Basic instructions (0..63)

Instructions 0..63 are identical to standard OM32 instructions, with the exceptions listed below.

Parameter 0, if applicable, is the Pin-Offset rather than the Pin number
Parameter 1, if applicable, Is the Parameter as defined in the OM32 instruction set

Exceptions:

1 = "Jump"
Instruction = 1
Parameter 0 = Pin-Offset
Parameter 1 = Aspect # (0..11)
Parameter 2 = Instruction # (0..8)

The function is identical to the original "Set Aspect" instruction, however "Jump" also gives the option to branch to a specific instruction within the Aspect instruction set

3.5.2 OC32 Timed instructions (112..127)

A "Timed Instruction" is invoked a predetermined time after the instruction is executed. Technically, all the instruction does is set a timer that invokes the instruction when the timer times-out
The time is determined by the parameters "Time" and "TimeBase" in the following way:

*8 ^ timebase * time * 20ms*

Time=0 is invalid and will reset the timer without ever invoking the instruction

The following instructions are defined:

112..123 = "Set Aspect"
Instruction = 0b0111aaaa
Parameter 0 = pin-offset
Parameter 1 = 0
Parameter 2 = Time
Parameter 3 = TimeBase

aaaa = the aspect number or program number to activate

127 = De-activate Servo (Not yet implemented)
Instruction = 0b01111111
Parameter 0 = pin-offset
Parameter 1 = 0
Parameter 2 = Time
Parameter 3 = TimeBase

3.5.3 OC32 Sequence instructions (96..111)

A "Sequence Instruction" will wait a predetermined time after the instruction is executed before executing the next instruction in the list.
The time is determined by the parameters "Time" and "TimeBase" in the following way:

*8 ^ timebase * time * 20ms*

Time=0 is invalid and will reset the timer without ever invoking the next instruction

When the last instruction in the list (so either #2 or #8, depending on the Aspect Mode setting) is a sequence instruction, the "next instruction" to be executed is Instruction #1 (so **not** Instruction #0).

The following Sequence Instructions are defined:

96 = "Set Level & Wait"
Instruction = 0b01100000
Parameter 0 = Acceleration (0..31, 127)[7]
Parameter 1 = Level (0..31)
Parameter 2 = Time
Parameter 3 = TimeBase

97 = "Set Level & Wait Random"[8]
Instruction = 0b01100001
Parameter 0 = Acceleration (0..31, 127)
Parameter 1 = Level (0..31)
Parameter 2 = Time
Parameter 3 = TimeBase

98 = "Set Random & Wait"[9]
Instruction = 0b01100010
Parameter 0 = Acceleration (0..31, 127)
Parameter 1 = Chance (0..31)
Parameter 2 = Time
Parameter 3 = TimeBase

99 = "Set Random & Wait Random"
Instruction = 0b01100011
Parameter 0 = Acceleration (0..31, 127)
Parameter 1 = Chance (0..31)
Parameter 2 = Time
Parameter 3 = TimeBase

100 = "Set Servo & Wait"
Instruction = 0b01100100
Parameter 0 = 0
Parameter 1 = Position (-64..63) + 64
Parameter 2 = Time
Parameter 3 = TimeBase

101 = "Set Servo & Wait Random"
Instruction = 0b01100101
Parameter 0 = 0
Parameter 1 = Position (-64..63) + 64
Parameter 2 = Time
Parameter 3 = TimeBase

102 = "Wait"
Instruction = 0b01100110
Parameter 0 = 0
Parameter 1 = 0
Parameter 2 = Time
Parameter 3 = TimeBase

[7] If Acceleration = 127, the default acceleration as defined by the Pin is used
[8] Wait a random time between 0 and the given time
[9] Set Pin with probability "Chance" to the "on"-level defined for the Pin, else set to the "off" level defined for the pin.

101 = "Wait Random"
Instruction = 0b01100111
Parameter 0 = 0
Parameter 1 = 0
Parameter 2 = Time
Parameter 3 = TimeBase

3.5.4 Notes on OC32 Timed Instructions and Sequence Instructions

The **timed-instructions** and **sequence-instructions** use the Pin-Event-Timer for the pin. There is only one Pin-Event-Timer per pin, so if another timed event for the same pin is set before a running timer finishes, the timer is overwritten and the corresponding event will never be executed.

Care shall be taken that (infinite) loops are avoided. There is no check in the OC32 to prevent infinite loops by Jumps. Such infinite loops may cause the OC32 to "hang". The OC32 can be restarted by a hardware-reset or power-down/up-cycle, after which the cause of the infinite loop shall be investigated by the user and be eliminated by re-programming the Aspect(s) causing the problem.

This page is intentionally blank