IEC 62386-102

Edition 2.1 2018-09

FINAL VERSION

VERSION FINALE

colour inside

Digital addressable lighting interface – Part 102: General requirements – Control gear

Interface d'éclairage adressable numérique – Partie 102: Exigences générales – Appareillages de commande

IEC 62386-102:2014-11+AMD1:2018-09 CSV(en-fr)

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

DIGITAL ADDRESSABLE LIGHTING INTERFACE –

Part 102: General requirements – Control gear

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

This Consolidated version of IEC 62386-102 bears the edition number 2.1. It consists of the second edition (2014-11) [documents 34C/1099/FDIS and 34C/1112/RVD], its amendment 1 (2018-09) [documents 34/523/FDIS and 34/534/RVD]. The technical content is identical to the base edition and its amendment.

This Final version does not show where the technical content is modified by amendment 1. A separate Redline version with all changes highlighted is available in this publication.

International Standard IEC 62386-102 has been prepared by subcommittee 34C: Auxiliaries for lamps, of IEC technical committee 34: Lamps and related equipment.

This second edition constitutes a technical revision.

This edition includes the following significant technical changes with respect to the previous edition:

a) elimination of all non-control gear relevant definitions,

b) improvement of the requirements for control gear by clarifying the description,

c) improvement of the test command iterations to increase the compatibility,

d) addition of new commands, and

e) the deletion of the requirements for:

   1) timing;

   2) control devices.

   The requirements for timing are now in Part 101 and the requirements for control devices are in Part 103.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

This Part 102 is intended to be used in conjunction with Part 101, which contains general requirements for the relevant product type (system), and with the appropriate Part 2xx (particular requirements for control gear) containing clauses to supplement or modify the corresponding clauses in Parts 101 and 102 in order to provide the relevant requirements for each type of product.

A list of all parts of the IEC 62386 series, under the general title: *Digital addressable lighting interface*, can be found on the IEC website.

The committee has decided that the contents of the base publication and its amendment will remain unchanged until the stability date indicated on the IEC web site under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

IEC 62386 contains several parts, referred to as series. The 1xx series includes the basic specifications. Part 101 contains general requirements for system components, Part 102 extends this information with general requirements for control gear and Part 103 extends it further with general requirements for control devices.

The 2xx parts extend the general requirements for control gear with lamp specific extensions (mainly for backward compatibility with Edition 1 of IEC 62386) and with control gear specific features.

The 3xx parts extend the general requirements for control devices with input device specific extensions describing the instance types as well as some common features that can be combined with multiple instance types.

This second edition of IEC 62386-102 is intended to be used in conjunction with IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018 and with the various parts that make up the IEC 62386-2xx series for control gear, together with IEC 62386-103:2014 and IEC 62386-103:2014/AMD1:2018 and the various parts that make up the IEC 62386-3xx series of particular requirements for control devices. The division into separately published parts provides for ease of future amendments and revisions. Additional requirements will be added as and when a need for them is recognised.

The setup of the standard is graphically represented in Figure 1 below.

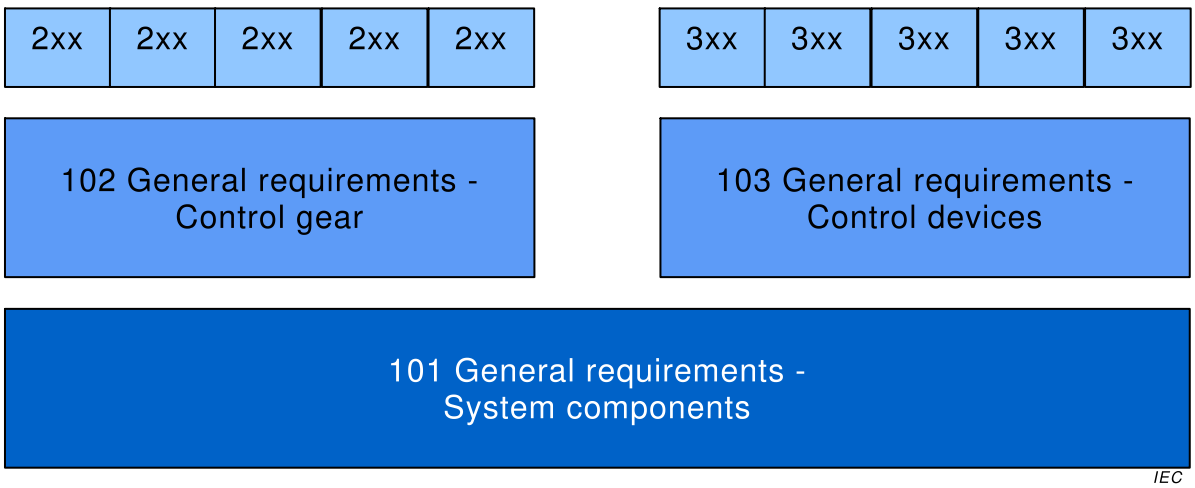

Figure 1 – IEC 62386 graphical overview

When this part of IEC 62386 refers to any of the clauses of the other two parts of the IEC 62386-1xx series, the extent to which such a clause is applicable and the order in which the tests are to be performed are specified. The other parts also include additional requirements, as necessary.

All numbers used in this International Standard are decimal numbers unless otherwise noted. Hexadecimal numbers are given in the format 0xVV, where VV is the value. Binary numbers are given in the format XXXXXXXXb or in the format XXXX XXXX, where X is 0 or 1 and "x" in binary numbers means "don't care".

The following typographic expressions are used:

Variables: *variableName* or *variableName[3:0]*, giving only bits 3 to 0 of *variableName*

Range of values: [lowest, highest]

Command: “COMMAND NAME”

DIGITAL ADDRESSABLE LIGHTING INTERFACE –

Part 102: General requirements – Control gear

1 Scope

This Part of IEC 62386 is applicable to control gear in a bus system for control by digital signals of electronic lighting equipment which is in line with the requirements of IEC 61347 (all parts), with the addition of DC supplies.

NOTE Tests in this standard are type tests. Requirements for testing individual control gear during production are not included.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 62386-101:2014, *Digital addressable lighting interface – Part 101: General requirements – System components*
IEC 62386-101:2014/AMD1:2018

IEC 62386-103:2014, *Digital addressable lighting interface – Part 103: General requirements – Control devices*
IEC 62386-103:2014/AMD1:2018

3 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62386-101and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at http://www.electropedia.org/
- ISO Online browsing platform: available at http://www.iso.org/obp

3.1
actual level
value representing the current light output

3.2
arc power
power supplied to the light sources (lamps)

3.3
broadcast
type of address used to address all control gear in the system at once

3.4
broadcast unaddressed
type of address used to address all control devices in the system that have no short address at once

3.5
DAPC
direct arc power control
a method to directly control the light output

Note 1 to entry: The note to entry in French concerns the French text only.

3.6
DTR
data transfer register
multipurpose register used to exchange data

Note 1 to entry: The note to entry in French concerns the French text only.

3.7
group address
type of address used to address a group of control gear in the system all at once

3.8
GTIN
global trade item number
number used for the unique identification of trade items worldwide

Note 1 to entry: For further information see http://en.wikipedia.org/wiki/GTIN

Note 2 to entry: The number is comprised of a GS1 or U.P.C. company prefix followed by an item reference number and a check digit. It is described in the “GS1 General Specifications”.

Note 3 to entry: The note 3 to entry in French concerns the French text only.

1.1
identification
temporary state used during commissioning that allows the installer to identify particular control gear

3.9
level
8 bit value

3.10
MASK
the value 0xFF

3.11
monotonic
a function f defined on a subset of the real numbers with real values is called monotonically non-decreasing, if for all x and y such that $x \leq y$ one has $f(x) \leq f(y)$, so f preserves the order. Likewise, a function is called monotonically non-increasing if, whenever $x \leq y$, then $f(x) \geq f(y)$, so it reverses the order. For this standard monotonic is defined as either monotonically non-decreasing or monotonically non-increasing

3.12
NO
answer used to deny or refuse a query

Note 1 to entry: If a query is asked where the answer is NO, there will be no response, such that the sender of the query will conclude “no backward frame" following IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018, 8.2.5.

Note 2 to entry: The answer NO could also be triggered by a missed query.

3.13
NVM
non-volatile read/write memory, the content of which can be changed and will not be lost due to a power cycle

3.14
opcode
operation code
part of a forward frame that identifies the command to be executed

3.15
operating mode
set of states identified by a number in the range [0,255], characterised by a collection of variables and memory settings, and used to select a set of functionality to be exhibited by a control gear, including its required reaction to commands

Note 1 to entry: Control gear may support more than one operating mode

3.16
PHM
physical minimum level corresponding to the minimum light output the control gear can operate at

3.17
RAM
volatile read/write memory, the content of which can be changed and will be lost due to a power cycle

3.18
random address
random 24 bit number generated by the control gear on request during system initialisation

Note 1 to entry: Annex A.1 provides an example of how the search and random addresses are used.

3.19
reset state

state in which all NVM variables of the control gear have their reset value, except those that are marked “no change” or are otherwise explicitly excluded

3.20
ROM
non-volatile read only memory, the content of which is fixed

Note 1 to entry: In this standard read only is meant from a system perspective. A ROM variable may actually be implemented in NVM, but this standard does not provide any mechanism to change its value.

3.21
scene
configurable preset level

3.22
search address
24 bit number used to identify an individual control gear in the system during initialisation

Note 1 to entry: Annex A.1 provides an example of how the search and random addresses are used.

3.23
short address
type of address used to address an individual control gear in the system

3.24
startup
time needed to change from lamp off to normal operation of the lamp or failure state

Note 1 to entry: This time includes preheat and ignition.

1.2
strictly monotonic
a function *f* defined on a subset of the real numbers with real values is called monotonically increasing, if for all *x* and *y* such that $x < y$ one has $f(x) < f(y)$, so *f* preserves the order. Likewise, a function is called monotonically decreasing if, whenever $x < y$, then $f(x) > f(y)$, so it reverses the order. For this standard strictly monotonic is defined as either monotonically increasing or monotonically decreasing

3.25
target level
the target light output expected after completion of the current level command

3.26
YES
answer used to accept or affirm a query

Note 1 to entry: If a query is asked where the answer is YES, the response will be a backward frame containing the value of MASK.

4 General

4.1 General

The requirements of IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, Clause 4 apply, with the restrictions, changes and additions identified below.

4.2 Version number

This subclause replaces IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, 4.2.

The version shall be in the format "x.y", where the major version number x is in the range of 0 to 62 and the minor version number y is in the range of 0 to 2. When the version number is encoded into a byte, the major version number x shall be placed in bits 7 to 2 and the minor version number y shall be placed in bits 1 to 0.

At each amendment to an edition of IEC 62386-102 the minor version number shall be incremented by one.

At a new edition of IEC 62386-102 the major version number shall be incremented by one and the minor version number shall be set to 0.

The current version number is “*versionNumber*” as defined in Table 14.

NOTE Normally 2 amendments on IEC documents are made before a new edition is created.

5 Electrical specification

The requirements of IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, Clause 5 apply.

6 Interface power supply

If a bus power supply is integrated into a control gear, the requirements of IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, Clause 6 apply.

7 Transmission protocol structure

7.1 General

The requirements of IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, Clause 7 apply, with the following additions.

7.2 16 bit forward frame encoding

7.2.1 General

For commands, the 16 bit forward frame shall be encoded as is depicted in Table 1.

Table 1 – 16-bit command frame encoding

Bytes/Bits									Device addressing method
Address byte								Opcode byte	
15	14	13	12	11	10	9	8[a]	7...0	
0	64 short addresses						x		Short addressing
1	0	0	16 group addresses				x		Group addressing
1	1	1	1	1	1	0	x		Broadcast unaddressed
1	1	1	1	1	1	1	x		Broadcast
1010 0000 to 1100 1011									Special command
1100 1100 to 1111 1011									Reserved

[a] Selector bit, see 7.2.2; 0 indicates DAPC, 1 indicates other command

7.2.2 Address byte

The address byte provides

- the method of device addressing used by the application controller;
- the type of command transmitted in the opcode byte;
  Bit 8 =’1’: standard command;
  Bit 8 =’0’: direct arc power control (DAPC) command;
- address spaces for special commands;
- reserved device addresses.

Reserved addresses should not be used by the application controller.

7.2.3 Opcode byte

The opcode byte provides

- for DAPC commands, the requested light output;
- for standard commands, the opcode;
- command specific information for special commands;
- reserved information for reserved commands.

8 Timing

The requirements of IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, Clause 8 apply.

9 Method of operation

9.1 General

The requirements of IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, Clause 9 apply with the following additions.

9.2 Control gear

9.2.1 General

Control gear may receive commands from an application controller. The application controller is specified by IEC 62386-103:2014 and IEC 62386-103:2014/AMD1:2018.

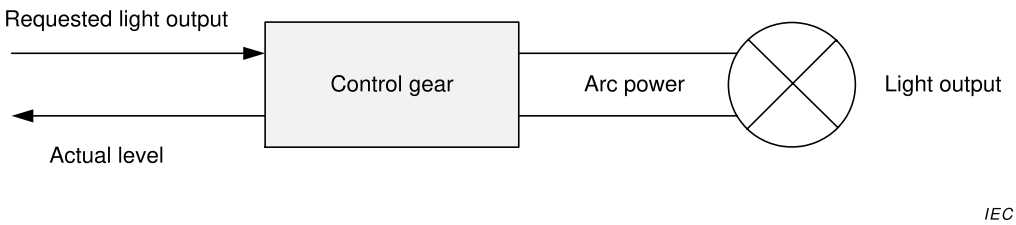

Figure 2 – Control gear directly operating a light source

Figure 2 shows how the various levels lead to light output. The maximum (light) output level of a control gear is referred to as 100 %. All levels are specified in a relative way. Physically there is a minimum that the control gear can supply whilst there is still light output. This is known as the physical minimum level (PHM).

NOTE PHM is control gear specific, and is greater than 0.

9.2.2 Control gear phases

9.2.2.1 General

Depending on the light source various phases of operation can be identified within a control gear. In general these are as follows.

9.2.2.2 Standby

During this phase, the lamp is off and only in this phase both "*targetLevel*" and "*actualLevel*" are 0.

9.2.2.3 Startup

Startup is a transitional phase changing from standby to normal operation or failure. This phase is sometimes noticeable as a delay. Examples are:

- preheat: the lamp is heated to prepare for ignition. This is typically seen for fluorescent light sources;
- ignition: the lamp is ignited. This is typically seen for HID light sources and fluorescent light sources after preheat;
- power stage preparation.

For further information and exceptions refer to 9.16.3.

9.2.2.4 Normal operation

While in normal operation the lamp is emitting light and can be operated as expected.

For further information and exceptions refer to 9.16.3.

9.2.2.5 Failure

During the failure phase the lamp cannot be operated as expected.

For further information and exceptions refer to 9.16.3.

9.3 Dimming curve

The dimming curve determines how the level shall be translated into light output.

An *"actualLevel"* greater than or equal to 1 and less than or equal to 254 shall be translated into light output according to

$$\text{Light output } (actualLevel) = 10^{\frac{actualLevel-1}{253/3}-1} \%.$$

Light output is expressed relative to the maximum possible light output of a given control-gear-lamp combination. The dimming curve starts at 0,1 % for *"actualLevel"* equal to 0x01 and ends at 100 % for *"actualLevel"* equal to 0xFE. The dimming curve is strictly monotonic, and the relative accuracy shall be ±½ step. This shall be tested using a fade, excluding PHM.

NOTE 1 The dimming curve is intended to compensate the light sensitivity curve of the human eye.

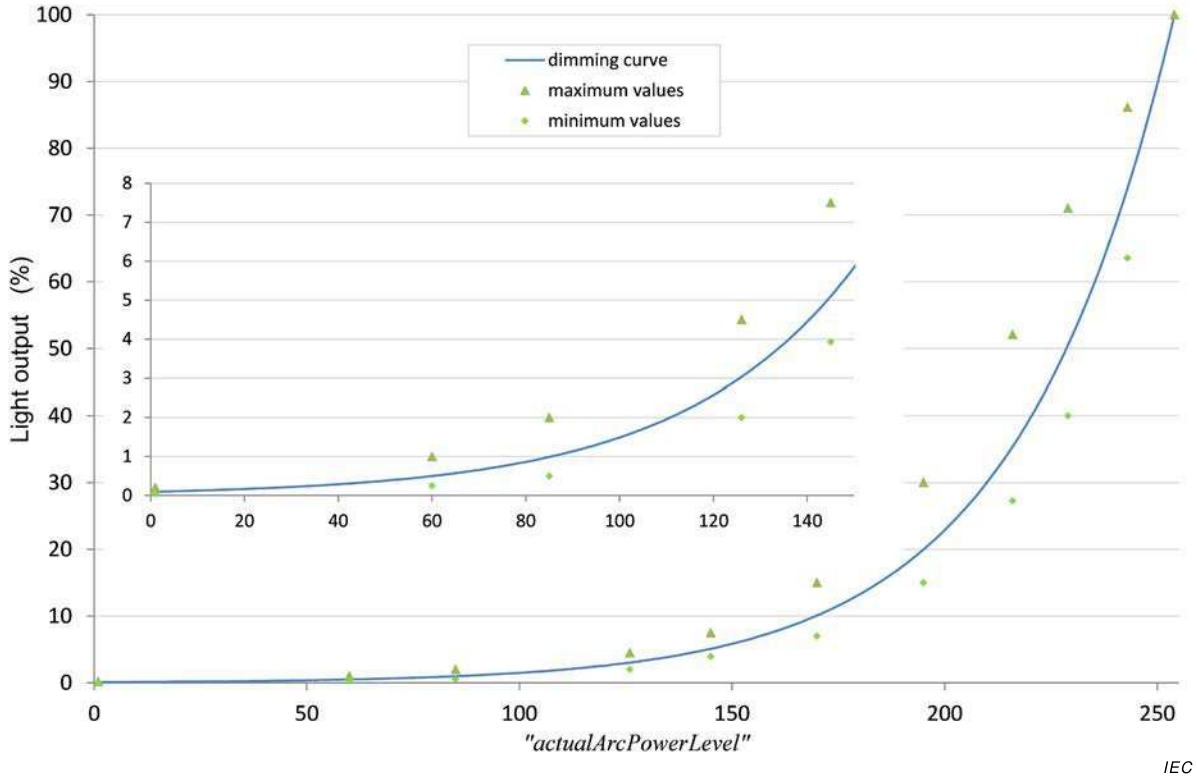

Figure 3 – Dimming curve

The accuracy of light output is specified by the test points given in Table 2. The test points of Table 2 and the dimming curve are depicted in Figure 3, and Table 3 shows the light output versus the level. The lamp type or load used during testing shall be stated for reproducibilty.

NOTE 2 The minimum and maximum values are based on the test values that can be found in IEC 62386-102:2009.

Table 2 – Dimming curve tolerance (%, rounded to two decimals)

Level	1	60	85	126	145	170	195	216	229	243	254
Minimum value	0,05	0,25	0,50	2,00	3,93	7,00	15,00	27,28	40,00	63,53	
Nominal value	0,10	0,50	0,99	3,04	5,10	10,09	19,97	35,43	50,53	74,05	100,00
Maximum value	0,20	1,00	2,00	4,50	7,50	15,0	30,00	52,09	71,00	86,14	

Table 3 – Dimming curve

Level	Light output	Level	Light output	Level	Light output	Level	Light output	Level	Light output
1	0,100	52	0,402	103	1,620	154	6,520	205	26,241
2	0,103	53	0,414	104	1,665	155	6,700	206	26,967
3	0,106	54	0,425	105	1,711	156	6,886	207	27,713
4	0,109	55	0,437	106	1,758	157	7,076	208	28,480
5	0,112	56	0,449	107	1,807	158	7,272	209	29,269
6	0,115	57	0,461	108	1,857	159	7,473	210	30,079
7	0,118	58	0,474	109	1,908	160	7,680	211	30,911
8	0,121	59	0,487	110	1,961	161	7,893	212	31,767
9	0,124	60	0,501	111	2,015	162	8,111	213	32,646
10	0,128	61	0,515	112	2,071	163	8,336	214	33,550
11	0,131	62	0,529	113	2,128	164	8,567	215	34,479
12	0,135	63	0,543	114	2,187	165	8,804	216	35,433
13	0,139	64	0,559	115	2,248	166	9,047	217	36,414
14	0,143	65	0,574	116	2,310	167	9,298	218	37,422
15	0,147	66	0,590	117	2,374	168	9,555	219	38,457
16	0,151	67	0,606	118	2,440	169	9,820	220	39,522
17	0,155	68	0,623	119	2,507	170	10,091	221	40,616
18	0,159	69	0,640	120	2,577	171	10,371	222	41,740
19	0,163	70	0,658	121	2,648	172	10,658	223	42,895
20	0,168	71	0,676	122	2,721	173	10,953	224	44,083
21	0,173	72	0,695	123	2,797	174	11,256	225	45,303
22	0,177	73	0,714	124	2,874	175	11,568	226	46,557
23	0,182	74	0,734	125	2,954	176	11,888	227	47,846
24	0,187	75	0,754	126	3,035	177	12,217	228	49,170
25	0,193	76	0,775	127	3,119	178	12,555	229	50,531
26	0,198	77	0,796	128	3,206	179	12,902	230	51,930
27	0,203	78	0,819	129	3,294	180	13,260	231	53,367
28	0,209	79	0,841	130	3,386	181	13,627	232	54,844
29	0,215	80	0,864	131	3,479	182	14,004	233	56,362
30	0,221	81	0,888	132	3,576	183	14,391	234	57,922
31	0,227	82	0,913	133	3,675	184	14,790	235	59,526
32	0,233	83	0,938	134	3,776	185	15,199	236	61,173
33	0,240	84	0,964	135	3,881	186	15,620	237	62,866
34	0,246	85	0,991	136	3,988	187	16,052	238	64,607
35	0,253	86	1,018	137	4,099	188	16,496	239	66,395
36	0,260	87	1,047	138	4,212	189	16,953	240	68,233
37	0,267	88	1,076	139	4,329	190	17,422	241	70,121
38	0,275	89	1,105	140	4,449	191	17,905	242	72,062
39	0,282	90	1,136	141	4,572	192	18,400	243	74,057
40	0,290	91	1,167	142	4,698	193	18,909	244	76,107
41	0,298	92	1,200	143	4,828	194	19,433	245	78,213
42	0,306	93	1,233	144	4,962	195	19,971	246	80,378
43	0,315	94	1,267	145	5,099	196	20,524	247	82,603
44	0,324	95	1,302	146	5,240	197	21,092	248	84,889
45	0,332	96	1,338	147	5,385	198	21,675	249	87,239
46	0,342	97	1,375	148	5,535	199	22,275	250	89,654
47	0,351	98	1,413	149	5,688	200	22,892	251	92,135
48	0,361	99	1,452	150	5,845	201	23,526	252	94,686
49	0,371	100	1,492	151	6,007	202	24,177	253	97,307
50	0,381	101	1,534	152	6,173	203	24,846	254	100,000
51	0,392	102	1,576	153	6,344	204	25,534		

9.4 Calculating *"targetLevel"*

An application controller instructs the control gear on the requested light output and on the behaviour during the transition from the *"actualLevel"* to the *"targetLevel"* by means of appropriate opcodes.

The *"targetLevel"* shall be calculated on the basis of the requested light output as follows:

- 0x00 shall be accepted as *"targetLevel"* and turn off the light.
- Any value between 0x01 and *"minLevel"* shall result in *"targetLevel"="minLevel"*.
- Any value between *"maxLevel"* and 0xFE shall result in *"targetLevel"="maxLevel"*.
- "MASK" shall have no effect on *"targetLevel"* except when a fade is running, see subclause 9.5.9.
- All other values shall be accepted as *"targetLevel"*.

The requested target level calculation of *"targetLevel"* shall also be applied if the request is based on an internally stored value, such as a scene, *"powerOnLevel"*, or *"systemFailureLevel"*.

On every change of *"targetLevel*", with the exception of the initialisation caused by a power cycle, the control gear shall update *"limitError"* (see subclause 9.16.5) and shall set *"lastLightLevel"* to the new *"targetLevel"*. If "*targetLevel"* is not 0x00, *"lastActiveLevel"* shall be set to "*targetLevel"*.

9.5 Fading

9.5.1 General

Fading is a linear transition in time from *"actualLevel"* to *"targetLevel"*. The *"actualLevel"*, and thus the light output, shall be strictly monotonic according to the applicable dimming curve.

A fade can be started in two ways:

- using a fade time: this sets a time to use for the fade process;
- using a fade rate: this sets a speed to use for the fade process.

A fade shall not be started if the calculated *"targetLevel"* is equal to *"actualLevel"*.

When a fade starts, the fade timer shall be started and *"fadeRunning"* shall be set to TRUE (see 9.16.6.).

During the fade, the light output shall be maintained as close to the ideal fading curve as possible.

During a process of fading up, *"actualLevel"* shall be incremented at a time corresponding to the intersection of an ideal fading curve with the mid-point between *"actualLevel"* and *"actualLevel"* + 1. Likewise, when fading down, *"actualLevel"* shall be decremented at a time corresponding to the intersection of an ideal fading curve with the mid-point between *"actualLevel"* and *"actualLevel"* – 1. Figure 4 illustrates this, and applies for fades started using either fade time or fade rate.

Measurements of fade time / fade rate shall start after the stop condition of the command that triggers it. If the fade takes place immediately after startup, measurement shall be done from the moment "*lampOn*" is TRUE or, in case of total lamp failure, from the moment "*lampFailure*" is confirmed TRUE. A fade shall automatically end when the fade timer has been active for the applicable fade time. At this point the fade timer shall be stopped and *"fadeRunning"* shall be set to FALSE (see subclause 9.16.6.).

This means that the control gear fades to the target level even in case of a total lamp error.If a lamp is to be switched off at the end of the fade, the step from *"minLevel"* to 0x00 shall not contribute to the fade time. The step from *"minLevel"* to 0x00 shall be taken immediately after the fade time has elapsed.

If a lamp is to be lit at the beginning of the fade and dimmed to a certain value, the step from 0x00 to *"minLevel"* shall not contribute to the fade time. This means that the fade time starts when the startup phase has finished.

NOTE The transition from 0x00 to *"minLevel"* incorporates startup.

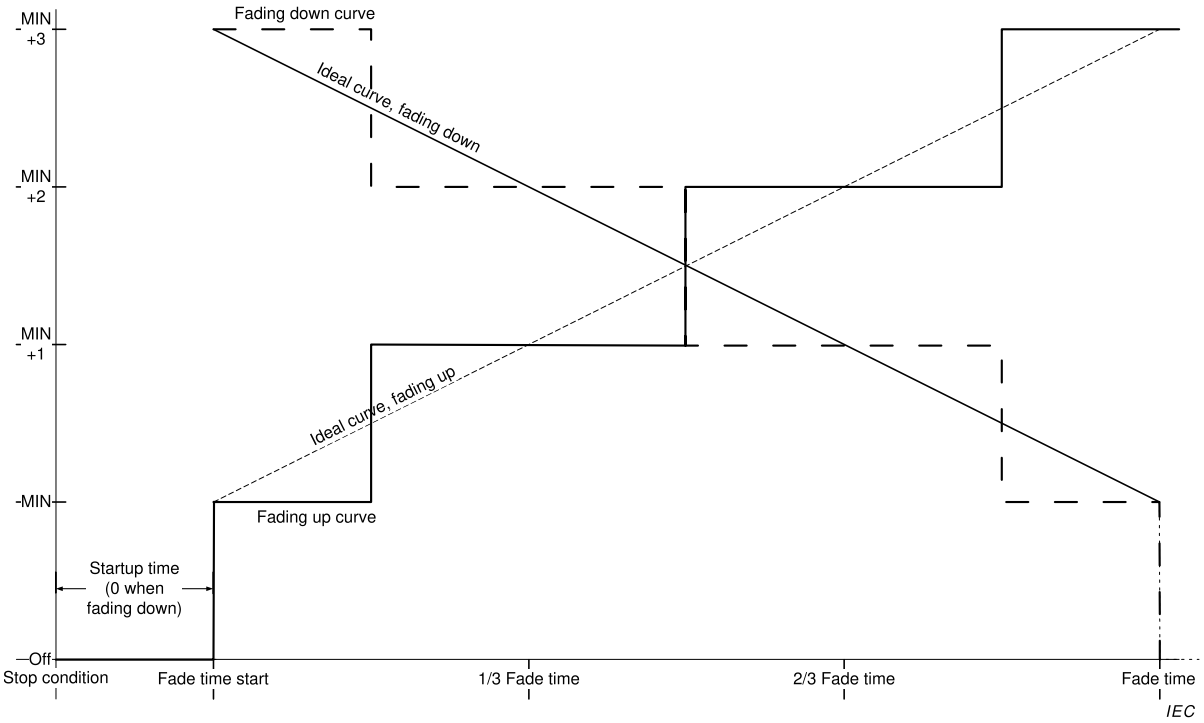

Figure 4 – Level over time, fading up and down

Testing shall be done with "*minLevel*" ≥ PHM+1. For further information, see Annex B.

9.5.2 Fade time

The fade time shall be according to Table 4:

"fadeTime" shall be set on receipt of the command "SET FADE TIME (*DTR0*)". "*fadeTime*" can be queried using QUERY FADE TIME/FADE RATE.

The *"fadeTime"* shall be set to a value according to the following steps:

- if *"DTR0"* > 15: 15
- in all other cases: *"DTR0"*

The fade time shall be calculated on the basis of *"fadeTime"* as follows:

- if *"fadeTime"* = 0: use

"fadeRate"	**Minimum fade rate** steps/s	**Nominal fade rate** steps/s	**Maximum fade rate** steps/s
1	322	358	394
2	228	253	278
3	161	179	197
4	114	127	139
5	80,5	89,4	98,4
6	56,9	63,3	69,6
7	40,3	44,7	49,2
8	28,5	31,6	34,8
9	20,1	22,4	24,6
10	14,2	15,8	17,4
11	10,1	11,2	12,3
12	7,1	7,9	8,7
13	5,0	5,6	6,1
14	3,6	4,0	4,3
15	2,5	2,8	3,1

- Extended fade time
- if *"fadeTime"* is in the range [1,15]: $\frac{1}{2} \cdot \sqrt{2^{"fadeTime"}} \cdot 1\ \mathrm{s}$

Table 4 lists the possible fade time values.

Table 4 – Fade times

"fadeTime"	**Minimum fade time** s	**Nominal fade time** s	**Maximum fade time** s
0	Extended fade		
1	0,6	0,7	0,8
2	0,9	1,0	1,1
3	1,3	1,4	1,6
4	1,8	2,0	2,2
5	2,5	2,8	3,1
6	3,6	4,0	4,4
7	5,1	5,7	6,2
8	7,2	8,0	8,8
9	10,2	11,3	12,4
10	14,4	16,0	17,6
11	20,4	22,6	24,9
12	28,8	32,0	35,2
13	40,7	45,3	49,8
14	57,6	64,0	70,4
15	81,5	90,5	99,6

9.5.3 Fade rate

The fade rate shall be according to Table 5.

"fadeRate" shall be set on receipt of the command "SET FADE RATE (*DTR0*)". "*fadeRate*" can be queried using QUERY FADE TIME/FADE RATE.

The *"fadeRate"* shall be set to a value according to the following steps:

- if *"DTR0"* > 15: 15
- if *"DTR0"* = 0: 1
- in all other cases: *"DTR0"*

The fade rate shall be calculated on the basis of *"fadeRate"* as follows:

$$\text{Fade rate} = \frac{506}{\sqrt{2^{"fadeRate"}}} \text{ steps/s.}$$

Table 5 lists the possible fade rate values.

Table 5 – Fade rates

"fadeRate"	**Minimum fade rate** steps/s	**Nominal fade rate** steps/s	**Maximum fade rate** steps/s
1	322	358	394
2	228	253	278
3	161	179	197
4	114	127	139
5	80,5	89,4	98,4
6	56,9	63,3	69,6
7	40,3	44,7	49,2
8	28,5	31,6	34,8
9	20,1	22,4	24,6
10	14,2	15,8	17,4
11	10,1	11,2	12,3
12	7,1	7,9	8,7
13	5,0	5,6	6,1
14	3,6	4,0	4,3
15	2,5	2,8	3,1

9.5.4 Extended fade time

If "*fadeTime*" equals 0, and the fast fade time as defined in IEC 62386 Part 207 is implemented and equals 0, the extended fade time shall be used.

The extended fade time can be set using a base value and a multiplier according to Table 6 and Table 7. The extended fade time can be calculated based on the base value and the multiplication factor.

Fade time = *extendedFadeTimeBase* * *extendedFadeTimeMultiplier*

This yields a range of 100 ms to 16 min, and a special value indicating no fade (as quickly as possible).

Table 6 – Extended fade time – base value

Base bits	Base value
0000b	1
0001b	2
0010b	3
0011b	4
0100b	5
0101b	6
0110b	7
0111b	8
1000b	9
1001b	10
1011b	11
1011b	12
1100b	13
1101b	14
1110b	15
1111b	16

Table 7 – Extended fade time – multiplier

Multiplier bits	Multiplication factor		
	Minimum	Nominal	Maximum
000b	0 ms [a]	0 ms [a]	0 ms [a]
001b	95 ms	100 ms	105 ms
010b	0,95 s	1 s	1,05 s
011b	9,5 s	10 s	10,5 s
100b	0,95 min	1 min	1,05 min
101b		Reserved	
110b		Reserved	
111b		Reserved	

[a] No fade (as quickly as possible)

On execution of “SET EXTENDED FADE TIME (*DTR0*)” the control gear shall set the following values based on *“DTR0*”. The format used shall be 0YYYAAAAb, where YYYb equals the fade time multiplier, and AAAAb the fade time base: The resulting fade time shall be monotically increasing when the base time increases.

- If “*DTR0”* > 0100 1111b:
  - “*extendedFadeTimeBase*” shall be set to 0;
  - “*extendedFadeTimeMultiplier*” shall be set to 0 ms, effectively setting the fade time to 0 s meaning no fade (as quickly as possible). The transition from *“actualLevel”* to *“targetLevel”* shall take place immediately and the light output shall be adjusted as

quickly as possible, meaning less than 0,8 s which represents the maximum fade time for *"fadeTime"* = 1 (see Table 4).

- In all other cases:
  - "*extendedFadeTimeBase*" shall be set to AAAAb;
  - "*extendedFadeTimeMultiplier*" shall be set to YYYb.

The extended fade time can be queried using "QUERY EXTENDED FADE TIME". The answer shall be 0 YYY AAAAb, where YYYb equals "*extendedFadeTimeMultiplier*" and AAAAb equals "*extendedFadeTimeBase*".

9.5.5 Using the fade time

Commands that use a fade time shall start a fade using the applicable fade time. This time can be determined based on the following rules:

- If "*fadeTime*" > 0: see Table 4
- If "*fadeTime*" = 0: The extended fade time shall be used, see Table 6 and Table 7. The extended fade time can be calculated by multiplying the base value and the multiplier.
- If "*extendedFadeTimeMultiplier*" = 0 ms, the fade time equals 0 s, meaning no fade (as quickly as possible). The transition from *"actualLevel"* to *"targetLevel"* shall take place immediately and the light output shall be adjusted as quickly as possible.

The target level shall be calculated on the basis of the command parameter. After the fade time has expired, the calculated target level shall be reached.

Since the extended fade time also supports fade times below 0,7 s that might not be realised by all control gear and light source combinations, such control gear may simply adjust the light output as quickly as possible when an extended fade time is requested that it physically cannot support. However, it should respond as if the fade has finished within the requested time.

9.5.6 Using the fade rate

9.5.6.1 Fading with "UP" and "DOWN" commands

Commands "UP" and "DOWN" shall start a 200 ms ± 20 ms fade.

"*targetLevel*" shall be calculated on the basis of the *"actualLevel"* using the applicable fade rate. After the 200 ms fade has expired, the calculated target level shall be reached.

NOTE 1 Since the fade rate is used, it is possible to reach *"minLevel"* or *"maxLevel"* before the end of the fade. This does not result in the *"fadeRunning"* bit being cleared.

NOTE 2 Because there are fade rate tolerances, different control gear can react to commands that use the fade rate at slightly different effective rates. Consequently, after the processing of these relative dimming commands, different gear might have different values for *"targetLevel"* (and therefore also for "*actualLevel*" and *"lastLightLevel"*).

9.5.6.2 Fading with "CONTINUOUS UP" and "CONTINUOUS DOWN" commands

Command "CONTINUOUS UP" shall set *"targetLevel"* to *"maxLevel"* and start a fade using the applicable fade rate. The fade shall stop when "*maxLevel"* is reached.

Command "CONTINUOUS DOWN" shall set *"targetLevel"* to *"minLevel"* and start a fade using the applicable fade rate. The fade shall stop when "*minLevel"* is reached.

Upon execution of either a "CONTINUOUS UP" or "CONTINUOUS DOWN" instruction at least one step shall be made, unless this is precluded by the values of *"minLevel"* or *"maxLevel"*.

Refer to 9.5.9 for stopping a fade before reaching *"minLevel"* or *"maxLevel"*.

NOTE 1 In contrast to the "UP" and "DOWN" instructions, it is not possible to reach *"minLevel"* or *"maxLevel"* before the end of the fade. Therefore, *"fadeRunning"* bit is going to be cleared at the end of a fade.

NOTE 2 Similar to the "UP" and "DOWN" instructions, different gear might end up with different values for *"targetLevel"*, "*actualLevel*" and *"lastLightLevel"* after the fade has been stopped ahead of time (e.g. via DAPC(MASK)).

9.5.7 System response to changes during a fade

If *"fadeTime","extendedFadeTimeBase*", *extendedFadeTimeMultiplier*" and/or *"fadeRate"* is changed during a running fade, then the running fade shall finish without the fade time and/or fade rate being recalculated. The next fade shall use the recalculated values.

9.5.8 System response to changes during standby and startup

If a fade is initiated during standby, the fade process shall be pended with "*actualLevel*" equal to "*minLevel*" during the startup phase. The reaction to level commands shall be the same as if the lamp(s) were operating at *"minLevel"*.

If a fade is initiated during start-up, the fade process shall be pended at "*actualLevel*". The reaction to level commands shall be the same as if the lamp(s) were operating at *"actualLevel"*.

The fade shall start:

- As soon as *"lampOn"* is TRUE, or
- in the case of total lamp failure, as soon as "*lampFailure*" is confirmed TRUE

For further information on *"lampOn"* and "*lampFailure*" see 9.16.3 and 9.16.4.

9.5.9 Stopping a fade

Any command setting one or more of the following variables

- *"targetLevel"*, *"minLevel"*, *"maxLevel"*

as well as the execution of one of the following commands

- "DAPC(MASK)", "SAVE PERSISTENT VARIABLES", "IDENTIFY DEVICE"

shall stop a running fade.

NOTE 1 The fade stops even if the value of the affected variable does not change.

When a running fade is stopped by an application controller, the fade timer shall be stopped immediately. After the fade timer has been stopped, *"targetLevel"* shall be set to *"actualLevel"* and the command shall be executed (if applicable).

If a running fade is stopped whilst it was pending at *"minLevel"* during startup, the control gear shall finish the startup process.

NOTE 2 This implies that in such a case both *"targetLevel"* and *"actualLevel"* are equal to *"minLevel"*.

9.6 Min and max level

Changing the min or max level shall stop any running fade, before the storage of the new min or max level.

"SET MIN LEVEL (*DTR0*)" shall set *"minLevel"* depending on the *"DTR0"* value:

- if 0 ≤ *"DTR0"* ≤ PHM: PHM
- if *"DTR0"* ≥ *"maxLevel"* or MASK: *"maxLevel"*
- in all other cases: *"DTR0"*

If *"actualLevel"* > 0 and *"actualLevel"* < *"minLevel"* as a result of setting a new min level, *"targetLevel"* shall be re-calculated on the basis of the new *"minLevel"*. *"actualLevel"* shall be changed to *"targetLevel"* immediately and the light output shall be adjusted as quickly as possible. As a consequence, "*limitError*" shall be set to TRUE.

"SET MAX LEVEL (*DTR0*)" shall set *"maxLevel"* depending on the *"DTR0"* value, as follows:

- if *"minLevel"* ≥ *"DTR0"*: *"minLevel"*
- if *"DTR0"* = MASK: 0xFE
- in all other cases: *"DTR0"*

If *"actualLevel"* > *"maxLevel"* as a result of setting a new max level, *"targetLevel"* shall be re-calculated on the basis of the new *"maxLevel"*. *"actualLevel"* shall be changed to *"targetLevel"* immediately and the light output shall be adjusted as quickly as possible. As a consequence, "*limitError*" shall be set to TRUE.

NOTE *"minLevel"* and *"maxLevel"* can be used to compensate for differences in control gear properties. E.g. if control gear have different values for PHM, they can be made to behave in a similar way by adjusting *"minLevel"*.

9.7 Commands

9.7.1 General

A control gear shall check the device addressing scheme to see if it is addressed by a command. The control gear shall execute the command, unless any of the following conditions hold:

- The command is sent using Short addressing and given short address is not equal to *"shortAddress"*.
- The command is sent using Group addressing and given group does not match any of the groups identified by *"gearGroups"*.
- The command is sent using Reserved addressing.
- The command is sent using Broadcast Unaddressed addressing and *"shortAddress"* is not MASK.
- The command is not defined (e.g. reserved command).

The following command groups can be identified:

- Level instructions
  - Level instructions without fade
  - Level instructions initiating a fade
- Configuration instructions
- Queries
- Special commands
  - Instructions
  - Queries
- Application extended commands

9.7.2 Level instructions without fade

Level instructions without fade are instructions where the *"targetLevel"* shall be calculated; the transition from *"actualLevel"* to *"targetLevel"* shall take place immediately and the light output shall be adjusted as quickly as possible.

These commands can be divided into three categories:

- Absolute level commands
  - "OFF", "RECALL MIN LEVEL", "RECALL MAX LEVEL",
- Relative level commands
  - "STEP UP", "STEP DOWN", "ON AND STEP UP", "STEP DOWN AND OFF"
- Configuration commands
  - "RESET", "SET MIN LEVEL (*DTR0*)", "SET MAX LEVEL (*DTR0*)"

9.7.3 Level instructions initiating a fade

Level instructions initiating a fade are instructions where the *"targetLevel"* shall be calculated; *"actualLevel"* shall fade to the *"targetLevel"* using the applicable fade time/rate. If the fade time equals 0 s, the transition from *"actualLevel"* to *"targetLevel"* shall take place immediately and the light output shall be adjusted as quickly as possible.

These commands can be divided into two categories:

- Absolute level instructions using the fade time
  - "DAPC (*level*)", "GO TO SCENE (*sceneNumber*)", "GO TO LAST ACTIVE LEVEL"
- Relative level instructions using the fade rate
  - "UP", "DOWN"
  - "CONTINUOUS UP", "CONTINUOUS DOWN"

9.7.4 Configuration instructions

Configuration instructions can be used to modify several control gear properties.

9.7.5 Queries

Queries can be used to request the value of several control gear properties.

9.7.6 Special commands

The special commands are a group of commands that are not addressable. All control gear shall interpret the special commands.

9.7.7 Application extended commands

Commands with their opcode in the range 0xE0 to 0xFF are reserved for special device types or features. Each device type or feature re-defines these commands, except for the command with opcode 0xFF ("QUERY EXTENDED VERSION NUMBER"). See 9.18 for further information.

9.8 Command iterations

9.8.1 General

The requirements of IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, 9.4 apply with the following additions.

9.8.2 Command iteration of “UP” and “DOWN” commands

“UP” and “DOWN” instructions can be sent as a command iteration. Upon execution of the first instruction of such an iteration, unless this is precluded by the values of “*minLevel*” or “*maxLevel*”, one step (final “*targetLevel*” = calculated “*targetLevel*”±1) shall be made.

NOTE 1 This ensures that there is an effect at the start of an iteration.

After that first step, the 200 ms fade shall start using the applicable fade rate. Subsequent steps shall be executed at intervals determined by the applicable fade rate, as long as the iteration continues. Every “UP” or “DOWN” instruction executed as a part of the iteration shall cause the 200 ms fade time to be restarted and *“targetLevel”* to be recalculated accordingly. The transition of “*actualLevel*” shall occur according to 9.5.1 and Figure 4, with the first such transition, excluding the initial step, occurring at a time of 1/(2* *“fadeRate”*) after execution of the first “UP” and “DOWN” command.

NOTE 2 If the fade rate changes during a command iteration, the new fade rate is not used during the execution of this command iteration.

Figure 5 summarizes the required behaviour. The iterations start at Cmd 1, and end at Time out.

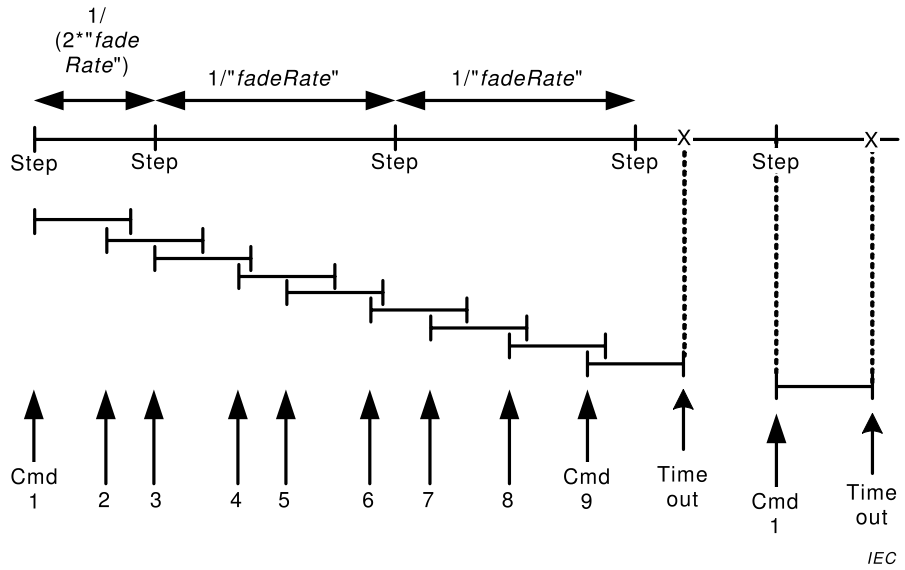

Figure 5 – Timing and response when executing command iteration

9.8.3 DAPC SEQUENCE (deprecated)

“ENABLE DAPC SEQUENCE” starts a direct arc power control (DAPC) command iteration that allows dynamic control of the light output.

Upon execution of “ENABLE DAPC SEQUENCE” the control gear shall temporarily use a fade time of 200 ms ± 20 ms while the command iteration is active independent of the actual fade/extended fade time. After the last fade of the sequence has finished, the original values shall be used.

NOTE As the fade time/rate variables do not change, the fade time/rate can be set and/or queried as normal.

The DAPC sequence shall end if 200 ms elapse without the control gear accepting a “DAPC (*level*)” command. The DAPC sequence shall be aborted on execution of an indirect arc power control command. “ENABLE DAPC SEQUENCE” accepted during DAPC command iteration, shall be discarded.

While the DAPC sequence is active, each execution of a “DAPC (*level*)” command shall start a 200 ms fade.

Since the DAPC sequence uses a fade time of 200 ms that might not be realised by all control gear and light source combinations, such gear may simply adjust the light output as quickly as possible. However, it should respond as if the fade has finished within the requested time.

9.9 Modes of operation

9.9.1 General

Different operating modes can be selected by means of command “SET OPERATING MODE (*DTR0*)”. The currently selected *“operatingMode”* can be queried by means of “QUERY OPERATING MODE”.

Operating modes 0x00 to 0x7F are defined in this standard. At least operating mode 0x00 shall be available. Operating modes 0x80 to 0xFF are manufacturer specific. The query “QUERY MANUFACTURER SPECIFIC MODE” can be used to determine whether the control gear is in an IEC 62386 standard operating mode or in a manufacturer specific mode.

9.9.2 Operating mode 0x00: standard mode

If a device is in standard mode (*“operatingMode”* = 0x00), its behaviour shall be as is required per this specification, until it is set in an operating mode different from 0x00.

9.9.3 Operating mode 0x01 to 0x7F: reserved

Operating modes 0x01 to 0x7F are reserved and shall not be used.

9.9.4 Operating mode 0x80 to 0xFF: manufacturer specific modes

Manufacturer specific modes should only be used if the features required by the application are not covered by the standard. If a control gear is in a manufacturer specific operating mode, the behaviour of the control gear may be manufacturer specific as well, with the following exceptions:

- as far as the control gear accesses the bus, it shall adhere to IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018;
- the control gear shall adhere to this specification at least as far as the following commands are concerned:
  - “SET OPERATING MODE (*DTR0*)”, “QUERY OPERATING MODE”, and “QUERY MANUFACTURER SPECIFIC MODE”.
  - all special commands (see 11.7) except WRITE MEMORY LOCATION (*DTR1, DTR0, data*), WRITE MEMORY LOCATION – NO REPLY (*DTR1, DTR0, data*) and PING.

For the above commands the various addressing methods shall apply, see 7.2.2.

It is recommended that even in manufacturer specific modes, the commands as specified in this standard still be obeyed.

9.10 Memory banks

9.10.1 General

Memory banks are freely accessible memory spaces defined for e.g. identification of the control gear in a system. Not all consecutive memory banks need to be implemented. Also within a memory bank not all consecutive locations need to be implemented. All implemented memory bank locations of all implemented memory banks are readable using memory access

commands. Part of the memory is read-only and programmed by the manufacturer of the control gear. For all other parts, write access using memory access commands can be enabled by the manufacturer. Write access to a memory bank location can be locked. Memory banks can be implemented using RAM, ROM or NVM.

The addressable memory space is limited to a maximum of almost 64 kBytes, organized in maximum 256 memory banks of maximum 255 bytes each. As this standard prescribes how to implement memory bank 0 and 1 (if present), and reserves memory banks 200 to 255, this leaves room for 198 memory banks for manufacturer specific purposes in the range of [2,199].

9.10.2 Memory map

If a manufacturer specific memory bank in the range of [2,199] is implemented, allocation of its content shall comply with the memory map provided in Table 8.

Table 8 – Basic memory map of memory banks

Address	Description	Default value (factory)	RESET value[b]	Memory type
0x00	Address of last accessible memory location	Factory burn-in, range [0x03,0xFE]	No change	ROM
0x01	Indicator byte [a]	[a]	[a]	Any[a]
0x02	Memory bank lock byte. Lockable bytes in the memory bank shall be read-only while the lock byte has a value different from 0x55.	0xFF	0xFF[c]	RAM
[0x03,0xFE]	Memory bank content [a]	[a]	[a]	Any[a]
0xFF	Reserved – not implemented	Answer NO	No change	n.a.

[a] Purpose, default/power on/reset value and memory access of these bytes shall be defined by the manufacturer.

[b] Reset value after “RESET MEMORY BANK”.

[c] Also used as power on value unless explicitly stated otherwise.

The byte in location 0x00 of each bank contains the address of the last accessible memory location of the bank. The value shall be in the range [0x03,0xFE].

The byte in location 0x01 is manufacturer specific. If implemented, the usage of this byte should be described by the manufacturer (as well as the entire content of the memory bank).

NOTE 1 It could be used for example to store a checksum in case of a memory bank with static content. Using a checksum on a memory bank where the content is changed by the control gear is not useful.

The byte in location 0x02 shall be used to lock write access. Memory location 0x02 itself shall never be locked for writing. While this memory location contains any value different from 0x55, all memory locations marked "(lockable)" of the corresponding memory bank shall be read only. The control gear shall not change the value of the lock byte other than as a consequence of power cycle or a “RESET MEMORY BANK (*DTR0*)” or other command affecting the lock byte.

Location 0xFF is a reserved location in every memory bank, and is not accessible. This location shall not be implemented as a normal memory bank location. When addressed, the control gear shall respond as if this location is not implemented, and it shall not increment “*DTR0*”.

NOTE 2 This location is reserved in order to stop the auto increment of DTR0.

9.10.3 Selecting a memory bank location

In order to select a memory bank location, a combination of memory bank number and location inside the memory bank is required.

The memory bank shall be selected by setting the memory bank number in *"DTR1"*. The location in the memory bank shall be selected by the value in *"DTR0"*.

9.10.4 Memory bank reading

A selected memory bank location can be read by means of command "READ MEMORY LOCATION (*DTR1, DTR0*)". The answer shall be the value of the byte at the addressed memory bank location.

If the selected memory bank is not implemented, the command shall be discarded. If the memory bank exists, and selected memory bank location is

- not implemented, or
- above the last accessible memory location,

the answer shall be NO.

If the selected memory bank location is below location 0xFF, *"DTR0"* shall be incremented by one, even if the memory location is not implemented. Otherwise, *"DTR0"* shall not change. This mechanism allows for easy consecutive reading of memory bank locations.

To ensure consistent data when reading a multi-byte value from a memory bank, it is recommended that a mechanism be implemented that latches all bytes of the multi-byte value when the first byte of the multi-byte value is read and that unlatches the bytes at any other command than "READ MEMORY LOCATION (*DTR1, DTR0*)".

After reading a number of bytes from a memory bank, the application controller should check the value of "*DTR0*" to verify it is at the expected/desired location. Any mismatch indicates an error while reading.

9.10.5 Memory bank writing

Write commands are special commands and therefore not addressable. In order to select the correct control gear(s) the addressable command "ENABLE WRITE MEMORY" shall be used. Upon execution of "ENABLE WRITE MEMORY", the addressed control gear(s) shall set *"writeEnableState"* to ENABLED.

Only while *"writeEnableState"* is ENABLED, and the addressed memory bank is implemented, the control gear shall execute the following commands to write to a selected memory bank location:

- "WRITE MEMORY LOCATION (*DTR1, DTR0, data*)": The control gear shall confirm writing a memory location with an answer equal to the value *data*.

NOTE 1 The value that can be read from the memory bank location is not necessarily *data*.

- "WRITE MEMORY LOCATION – NO REPLY (*DTR1, DTR0, data*)": Writing a memory location shall not cause the control gear to reply.

A control gear shall set *"writeEnableState"* to DISABLED if any command other than one of the following commands is accepted:

- "WRITE MEMORY LOCATION (DTR1, DTR0, data)", "WRITE MEMORY LOCATION – NO REPLY (DTR1, DTR0, data)"
- "*DTR0* (data)", "*DTR1* (data)", "*DTR2* (data)"

- “QUERY CONTENT DTR0”, “QUERY CONTENT DTR1”, “QUERY CONTENT DTR2”

If the selected memory bank location is

- not implemented, or
- above the last accessible memory location, or
- locked (see subclause 9.10.2), or
- not writeable,

the answer to “WRITE MEMORY LOCATION (*DTR1, DTR0, data*)” shall be NO and no memory location shall be written to.

If the selected memory bank location is below location 0xFF, *“DTR0”* shall be incremented by one. Otherwise, *“DTR0”* shall not change. This mechanism allows for easy consecutive writing to memory bank locations.

To ensure consistent data when writing a multi-byte value into a memory bank, it is recommended that a mechanism be implemented that only accepts the new multi-byte value for writing after all bytes of the multi-byte value have been accepted.

After writing a number of bytes to a memory bank, the application controller should check the value of “*DTR0*” to verify it is at the expected/desired location. Any mismatch indicates an error while writing.

NOTE 2 “*DTR0*” is also incremented if a non-implemented memory bank location is addressed before 0xFF is reached.

9.10.6 Memory bank 0

Memory bank 0 contains information about the control gear. Memory bank 0 shall be implemented in all control gear.

Memory bank 0 shall be implemented using the memory map shown in Table 9, with at least the memory locations up to address 0x7F implemented, excluding reserved locations.

Table 9 – Memory map of memory bank 0

Address	Description	Default value (factory)	Memory type
0x00	Address of last accessible memory location	factory burn-in	ROM
0x01	Reserved – not implemented	answer NO	n.a.
0x02	Number of last accessible memory bank	factory burn-in, range [0,0xFF]	ROM
0x03	GTIN byte 0 (MSB)[a]	factory burn-in	ROM
0x04	GTIN byte 1	factory burn-in	ROM
0x05	GTIN byte 2	factory burn-in	ROM
0x06	GTIN byte 3	factory burn-in	ROM
0x07	GTIN byte 4	factory burn-in	ROM
0x08	GTIN byte 5 (LSB)	factory burn-in	ROM
0x09	Firmware version (major)	factory burn-in	ROM
0x0A	Firmware version (minor)	factory burn-in	ROM
0x0B	Identification number byte 0 (MSB)	factory burn-in	ROM
0x0C	Identification number byte 1	factory burn-in	ROM
0x0D	Identification number byte 2	factory burn-in	ROM
0x0E	Identification number byte 3	factory burn-in	ROM
0x0F	Identification number byte 4	factory burn-in	ROM

Address	Description	Default value (factory)	Memory type
0x10	Identification number byte 5	factory burn-in	ROM
0x11	Identification number byte 6	factory burn-in	ROM
0x12	Identification number byte 7 (LSB)	factory burn-in	ROM
0x13	Hardware version (major)	factory burn-in	ROM
0x14	Hardware version (minor)	factory burn-in	ROM
0x15	101 version number [b]	factory burn-in, according to implemented version number	ROM
0x16	102 version number of all integrated control gear [c]	factory burn-in, according to implemented version number	ROM
0x17	103 version number of all integrated control devices [d]	factory burn-in, according to implemented version number	ROM
0x18	Number of logical control device units in the bus unit	factory burn-in, range [0,64]	ROM
0x19	Number of logical control gear units in the bus unit	factory burn-in, range [1,64]	ROM
0x1A	Index number of this logical control gear unit	factory burn-in, range [0,(location 0x19)-1]	ROM
[0x1B,0x7F]	Reserved – not implemented	answer NO	n.a.
[0x80,0xFE]	Additional control gear information [e]	[e]	ROM
0xFF	Reserved – not implemented	answer NO	n.a.

[a] It is recommended that the product GTIN is not re-used within the expected lifetime of the product after installation.

[b] Format of the version number is defined in IEC 62386-101:2014 and IEC 62386-101:2014/AMD1:2018, 4.2.

[c] Format of the version number is defined in 4.2.

[d] Format of the version number is defined in IEC 62386-103:2014 and IEC 62386-103:2014/AMD1:2018, 4.2. If not implemented, this is indicated by 0xFF.

[e] Purpose and (default) value of these bytes shall be defined by the manufacturer.

If there is more than one logical unit built into one bus unit, all logical units shall have the same values in memory bank locations 0x03 up to and including 0x19.

A bus unit might contain both control gear and control devices. They share various numbers (e.g. GTIN, unique identification number...). To avoid problems when reading, and getting different answers depending on the addressing scheme used, the memory bank layout are the same for control gear and for control devices up to and including location 0x19. The data shall be the same as well. The application controller can use either the 102 or the 103 commands to identify the basic data, provided both are implemented.

The bytes in locations 0x03 to 0x08 (“GTIN 0” to “GTIN 5”) shall contain the Global Trade Item Number (GTIN), e.g. the EAN, in binary. The bytes shall be stored most significant first and filled with leading zeroes.

The bytes in locations 0x09 and 0x0A (“firmware version”) shall contain the firmware version of the bus unit.

The bytes in locations 0x0B to 0x12 (“identification number byte 0” to “identification number byte 7”) shall contain 64 bits of an identification number of the bus unit, preferably the serial number. The identification number shall be stored with least significant byte in “identification number byte 7” and unused bits shall be filled with 0.

The combination of the identification number and the GTIN number shall be unique.

The byte in location 0x13 and 0x14 (“hardware version”) shall contain the hardware version of the bus unit.

The byte in location 0x15 shall contain the implemented IEC 62386-101 version number of the bus unit.

The byte in location 0x16 shall contain the implemented IEC 62386-102 version number of the bus unit. If no control gear is implemented, the version number shall be 0xFF.

The byte in location 0x17 shall contain the implemented IEC 62386-103 version number of the bus unit. If no control device is implemented, the version number shall be 0xFF.

The byte in location 0x18 shall contain the number of logical control device units integrated into the bus unit. The number of logical units shall be in the range of 0 to 64.

The byte in location 0x19 shall contain the number of logical control gear units integrated into the bus unit. The number of logical units shall be in the range of 1 to 64.

The byte in location 0x1A shall represent the unique index number of the logical control gear unit that implements that memory bank. The valid range of this index number is 0 to the total number of logical control gear units in the bus unit minus one.

NOTE As an example there might be a product containing three logical devices with three different short addresses. Each of these control gear has the same GTIN and identification number, each reports as number of devices the value 3 and the index of the three control gear is reported as 0, 1 or 2 respectively. Reading location 0x1A using broadcast yields a backward frame according to IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018, 9.5.2 (overlapping backward frame).

9.10.7 Memory bank 1

Memory bank 1 is reserved for use by an OEM (original equipment manufacturer, e.g. a luminaire manufacturer) to store additional information, which has no impact on the functionality of the control gear. The control gear manufacturer may implement memory bank 1.

If implemented, memory bank 1 shall at least implement the memory locations up to and including address 0x10. The fixed usage for location 0x00 to 0x02 and the recommended memory map usage for location 0x03 to 0x10 is shown in Table 10.

Table 10 – Memory map of memory bank 1

Address	Description	Default value (factory)	RESET value[b]	Memory type
0x00	Address of last accessible memory location	factory burn-in, range [0x10,0xFE]	no change	ROM
0x01	Indicator byte [a]	[a]	[a]	any[a]
0x02	Memory bank 1 lock byte. Lockable bytes in the memory bank shall be read-only while the lock byte has a value different from 0x55.	0xFF	0xFF[c]	RAM
0x03	OEM GTIN byte 0 (MSB)	0xFF	no change	NVM (lockable)
0x04	OEM GTIN byte 1	0xFF	no change	NVM (lockable)
0x05	OEM GTIN byte 2	0xFF	no change	NVM (lockable)
0x06	OEM GTIN byte 3	0xFF	no change	NVM (lockable)
0x07	OEM GTIN byte 4	0xFF	no change	NVM (lockable)
0x08	OEM GTIN byte 5 (LSB)	0xFF	no change	NVM (lockable)
0x09	OEM identification number byte 0 (MSB)	0xFF	no change	NVM (lockable)
0x0A	OEM identification number byte 1	0xFF	no change	NVM (lockable)
0x0B	OEM identification number byte 2	0xFF	no change	NVM (lockable)
0x0C	OEM identification number byte 3	0xFF	no change	NVM (lockable)
0x0D	OEM identification number byte 4	0xFF	no change	NVM (lockable)
0x0E	OEM identification number byte 5	0xFF	no change	NVM (lockable)
0x0F	OEM identification number byte 6	0xFF	no change	NVM (lockable)
0x10	OEM identification number byte 7 (LSB)	0xFF	no change	NVM (lockable)
≥ 0x11	Additional control gear information [a]	[a]	[a]	[a]
0xFF	Reserved – not implemented	answer NO	no change	n.a.

[a] Purpose, default/power on/reset value and memory access of these bytes shall be defined by the manufacturer.

[b] Reset value after “RESET MEMORY BANK”.

[c] Also used as power on value.

The bytes in locations 0x03 to 0x08 (“OEM GTIN 0” to “OEM GTIN 5”) should be used to identify the product containing the control gear. If the bytes are used for GTIN the bytes shall be stored most significant bit first and filled with leading zeroes. These bytes should be programmed by the OEM.

The bytes in locations 0x09 to 0x10 (“OEM identification number byte 0” to “OEM identification number byte 7”) should contain 64 bits of an identification number of the OEM product. If the bytes are used for the identification number, it shall be stored with the least significant byte in

“Identification number byte 7” and unused bits shall be filled with 0. These bytes should be programmed by the OEM.

The combination of OEM GTIN and OEM identification number should be unique.

9.10.8 Manufacturer specific memory banks

The manufacturer may use additional memory banks in the range of 2 to 199 to store additional information. The memory map of additional banks shall comply with Table 8.

9.10.9 Reserved memory banks

Memory banks 200 to 255 are reserved for future use and shall not be implemented.

9.11 Reset

9.11.1 Reset operation

A control gear shall implement a reset operation to set all variables to their reset values (see Table 14).

NOTE For some variables this operation could have no effect at all.

The reset operation shall take at most 300 ms to complete. While the reset operation is in progress, the control gear may or may not respond to any command. However, until the reset operation is complete, none of the affected variables needs to have a defined value.

An application controller can trigger the reset operation using the “RESET” instruction and should wait at least 350 ms to ensure all gear have finished the reset operation.

9.11.2 Reset memory bank operation

A control gear shall implement a reset operation to set the content of all unlocked memory banks to their reset values (see 9.10), followed by locking the memory banks.

NOTE For some memory bank locations this operation could have no effect at all.

The reset operation shall take at most 10 s to complete. While the reset operation is in progress, the control gear may or may not respond to any command. However, until the reset operation is complete, none of the affected memory bank locations have a defined value.

An application controller can trigger the reset operation for a specific memory bank, or for all implemented memory banks, using the “RESET MEMORY BANK (*DTR0*)” instruction and it should wait for at least 10,1 s so as to allow all gear enough time to finish the reset memory bank operation.

9.12 System failure

If the control gear detects a system failure (see IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018, 4.11) and *“systemFailureLevel”* is not MASK, *“targetLevel”* shall be calculated on the basis of *“systemFailureLevel”*. The transition from *“actualLevel”* to *“targetLevel”* shall take place immediately and the light output shall be adjusted as quickly as possible.

If *“systemFailureLevel”* is MASK, the control gear shall not react to a system failure.

On restoration of the bus idle voltage the control gear shall not react.

"systemFailureLevel" can be set and queried with "SET SYSTEM FAILURE LEVEL (*DTR0*)" and "QUERY SYSTEM FAILURE LEVEL" respectively.

When bus power is restored after a system failure, bus-powered control gear shall follow the power-on procedure defined in subclause 9.13 below. Consequently, the variable *"systemFailureLevel"* is not used. Nevertheless, all control gear, including bus-powered control gear, shall maintain *"systemFailureLevel"* and conform to the requirements of the specifications of all the commands relating to it.

NOTE Implementing *"systemFailureLevel"* although this variable is normally not applicable for bus powered devices, is done to avoid separate test conditions of control gear.

9.13 Power on

After an external power cycle (see IEC 62386-101 and IEC62386-101:2014/AMD1:2018, 4.11.1), the device shall retain its most recent configuration, with the following exceptions:

- the memory bank write enable state shall be disabled for all memory banks and the lock byte shall be set to 0xFF;
- all running timers shall be stopped and cancelled/reset;
- *"powerCycleSeen"* shall be set to TRUE;
- *"actualLevel"* shall be set to 0x00 keeping the lamp off;
- *"lampOn"* shall be set to FALSE;
- *"limitError"* shall be set to FALSE;
- *"targetLevel"* shall be set to 0x00;
- the control gear may start preheating the lamp but the lamp shall not ignite. While preheating, "*actualLevel*" shall be kept at 0x00 contrary to normal startup activity.
- All variables mentioned in Table 14 shall be set to the value indicated in the power on value column. The variables that are marked with "no change" in the power on value column shall not be considered. The variables defined in implemented Parts 2xx shall be included.

Bus powered devices shall activate the power on level immediately. For externally powered devices the following holds:

If a level control command other than "GO TO SCENE (*sceneNumber*)" where the value of the scene equals MASK and other than DAPC(MASK) is accepted it shall be executed.

If "GO TO SCENE (*sceneNumber*)" where the value of the scene equals MASK is accepted, the control gear shall discard the command and continue as if no level control command has been accepted.

If DAPC(MASK) is executed, the control gear shall stop any startup activity.

NOTE 1 Since *"actualLevel"* = 0, this effectively keeps the lamp off.

The control gear shall activate the power on level according to Table 11 by calculating the *"targetLevel"* on the basis of "*powerOnLevel*". If "*powerOnLevel*" equals MASK, "*targetLevel*" shall be set to "*lastLightLevel*". *"actualLevel"* shall be set to *"targetLevel"* immediately and the light output shall be adjusted as quickly as possible.

If a level control command is accepted before the power on level is activated, this command shall be executed immediately and the control gear shall not activate the power on level.

Table 11 – Power on timing

Power on system response	Minimum time	Maximum time
Lamp off		540 ms
Grey area	> 540 ms	< 660 ms
Power on level	660 ms	

NOTE 2 Thus, there is an interval during which a control device can send a level control command which will be obeyed immediately, so DAPC(0x00) or DAPC(MASK) can be used to prevent from going automatically to *"powerOnLevel"*.

It is possible that a system failure is detected before the power on level has been reached. If "*systemFailureLevel*" is not MASK, the *"targetLevel"* is recalculated on the basis of *"systemFailureLevel"* and the power on level shall not be activated.

"powerOnLevel" can be set and queried with "SET POWER ON LEVEL (*DTR0*)" and "QUERY POWER ON LEVEL" respectively.

After receiving the first 16 bit forward frame on the interface after power-on, the control gear shall only respond to frames described in IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018.

9.14 Assigning short addresses

9.14.1 General

"shortAddress" shall be derived from *data* or *"DTR0"* depending on the command used. It shall be set on receipt of "PROGRAM SHORT ADDRESS (*data*)" or "SET SHORT ADDRESS (*DTR0*)" as follows:

- if *data* or *"DTR0"* = MASK: MASK (effectively deleting the short address)
- if *data* or *"DTR0"* = 1xxxxxxxb or xxxxxxx0b: no change
- in all other cases (0AAAAAA1b): 00AAAAAAb.

9.14.2 Random address allocation

A control gear shall implement an initialisation state, only in which, in addition to the other operations identified in this standard, a set of commands are enabled that allow an application controller to detect and uniquely identify control gear available on the bus and assign short addresses to these devices.

The initialisation state is a temporary state which is entered with the command "INITIALISE (*device*)". It shall end automatically 15 min ± 1,5 min after the last "INITIALISE (*device*)" command was executed. Additionally, a power cycle or the command "TERMINATE" shall cause the control gear to leave the initialisation state immediately.

The control gear shall have three possible values for *"initialisationState"*:

- DISABLED, not in initialisation state;
- ENABLED, in initialisation state;
- WITHDRAWN, in initialisation state, yet identified and withdrawn.

The following (special) commands are initialisation commands:

- "RANDOMISE", "COMPARE" and "WITHDRAW"
- "SEARCHADDRH (*data*)", "SEARCHADDRM (*data*)" and "SEARCHADDRL (*data*)"

- “PROGRAM SHORT ADDRESS (*data*)”, “VERIFY SHORT ADDRESS (*data*)” and “QUERY SHORT ADDRESS”
- “IDENTIFY DEVICE”

NOTE “IDENTIFY DEVICE” is by itself not an initialisation command, but typically used during initialisation

9.14.3 Identification of a device

9.14.3.1 General

During identification no variables shall be affected unless explicitly stated otherwise. Where appropriate, variables can be temporarily ignored, so that after the identification has ended, there are no side effects.

When identification is active, the light output may be at any level between off and 100 %, “*minLevel*” and “*maxLevel*” as well as “*actualLevel*” being in effect temporarily ignored.

Identification shall be stopped upon execution of any instruction other than INITIALISE (*device*), RECALL MIN LEVEL, RECALL MAX LEVEL or IDENTIFY DEVICE.

After identification has stopped, the light output shall be adjusted as quickly as possible to reflect “*actualLevel*” and the command shall be executed (if applicable).

9.14.3.2 Method one: single instruction

Identification can be started by sending the instruction “IDENTIFY DEVICE”. This shall start or restart a 10 s ± 1 s timer. While the timer is running, a procedure enabling an observer to identify the selected control gear shall run. If the timer expires, identification shall stop.

NOTE The actual procedure is manufacturer specific.

While identification is active, the control gear shall, without interrupting the identification procedure:

- on RECALL MIN LEVEL: set “*actualLevel*” and “*targetLevel*” to “*minLevel*”;
- on RECALL MAX LEVEL: set “*actualLevel*” and “*targetLevel*” to “*maxLevel*”.

When identification is stopped by an application controller, the corresponding timer shall be cancelled immediately.

For examples of how to use the commands, see Annex A.

9.14.3.3 Method two: using “RECALL MAX LEVEL” and/or “RECALL MIN LEVEL” (deprecated)

While *“initialisationState”* is not DISABLED, the control gear shall:

- on RECALL MIN LEVEL: set “*actualLevel*” and “*targetLevel*” to “*minLevel*”, and then adjust the light output as quickly as possible to its PHM level. If, however, PHM is not visibly significantly different from 100 %, then the lamp shall be temporarily switched off instead;
- on RECALL MAX LEVEL: set “*actualLevel*” and “*targetLevel*” to “*maxLevel*”, and then adjust the light output as quickly as possible to 100 %.

Alternatively, the control gear shall execute “IDENTIFY DEVICE”, starting or re-triggering the identification procedure.

NOTE It is acceptable for the process of identifying individual control gear to depend upon both commands being executed in an alternating sequence.

Identification shall be stopped immediately when one of the following conditions hold:

- the *"initialisationState"* changes to DISABLED;
- upon execution of any instruction other than INITIALISE (*device*), RECALL MIN LEVEL, RECALL MAX LEVEL or IDENTIFY DEVICE.

For examples of how to use the commands, see Annex A.

9.14.4 Direct address allocation

"SET SHORT ADDRESS (*DTR0*)" can be used to directly program a short address to the addressed gear.

9.15 Failure state behaviour

If the control gear is in a failure state, in which operation of the lamp(s) is not possible as intended (lamp failure and/or control gear failure) it shall react to level commands in the following way:

The control gear shall calculate *"targetLevel"* in accordance with the commands executed, and control the lamp insofar as that is practicable. As a consequence of the fault, the normal relationship between *"actualLevel"* and light output could temporarily change.

NOTE For example, a control gear might, on detecting an excessively high temperature, protect itself from the risk of thermal damage by limiting the ight output.

If the failure state is resolved, the control gear shall re-establish the normal relationship between *"actualLevel"* and light output.

9.16 Status information

9.16.1 General

Each control gear shall expose its status as a combination of device properties as given in Table 12.

Table 12 – Control gear status

Bit	Description	Value	See
0	*"controlGearFailure"* is TRUE?	"1" = "YES"	9.16.2
1	*"lampFailure"* is TRUE?	"1" = "YES"	9.16.3
2	*"lampOn"* is TRUE?	"1" = "YES"	9.16.4
3	*"limitError"* is TRUE?	"1" = "YES"	9.16.5
4	*"fadeRunning"* is TRUE?	"1" = "YES"	9.16.6
5	*"resetState"* is TRUE?	"1" = "YES"	9.16.7
6	*"shortAddress"* is MASK?	"1" = "YES"	9.16.8
7	*"powerCycleSeen"* is TRUE?	"1" = "YES"	9.16.9

The device status can be queried using "QUERY STATUS". The bits shall reflect the actual situation without delay unless explicitly stated otherwise.

9.16.2 Bit 0: Control gear failure

A control gear failure according to this standard is a situation in which the control gear cannot operate as intended.

NOTE Examples are mains under voltage, over temperature, unexpected watchdog timers firing etc.

If a control gear failure is detected, *"controlGearFailure"* shall be set to TRUE.

If the failure is no longer detected, and normal operation has been resumed, *"controlGearFailure"* shall be set to FALSE.

Control gear failure shall be detected and indicated latest after 30 s.

9.16.3 Bit 1: lamp failure

A lamp failure according to this standard is a situation in which the lamp cannot be operated as intended due to for example incorrect lamp connection or lamp defects. The minimum detection method for lamp failure is lamp disconnect, unless explicitly stated otherwise depending on the light source type (see 11.5.19).

If a lamp failure is detected, *"lampFailure"* shall be set to TRUE. Lamp failure shall be detected and indicated latest after 30 s when the control gear is not in standby (see 9.2). In case the startup phase takes longer than 30 s, for example for HID lamps, *"lampFailure"* shall be set at the end of the startup phase to the correct value.

A total lamp failure is a lamp failure with no light output. A partial lamp failure is a lamp failure with still some light output.

If "*lampFailure*" is TRUE, the control gear shall periodically check to determine whether the lamp situation has improved. This check shall be executed at least whenever "*targetLevel*" changes from 0x00 to a greater value. After a successful startup, *"lampFailure"* shall be set to FALSE.

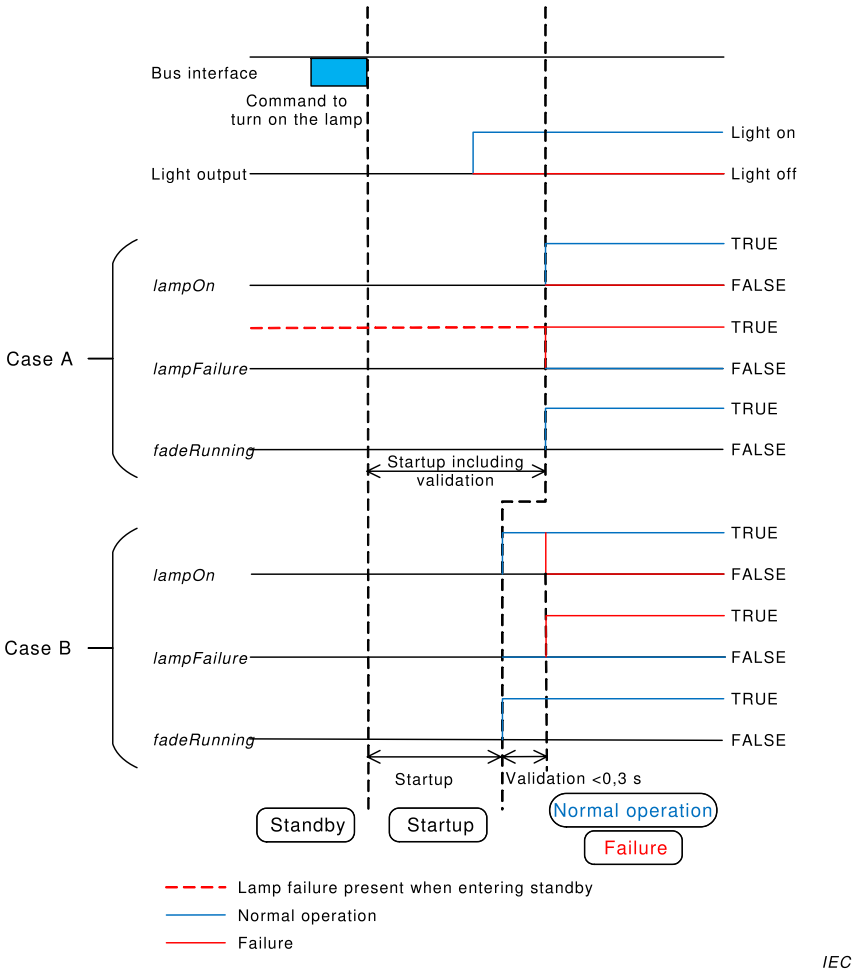

Figure 11 – Correlation between "*lampFailure*", "*lampOn*", and "*fadeRunning*" bits

The startup phase shall include the validation that the lamp is stable and emitting light before continuing with normal operation or failure. This behaviour is illustrated in Figure 11 Case A and applies to the following situations:

- "*lampFailure*" is TRUE before entering standby, or
- the lamp validation takes longer than 0,3 s.

The startup phase may exclude the validation of the lamp in the following situation, illustrated in Figure 11 Case B:

- "*lampFailure*" is FALSE before entering standby and,
- the lamp validation takes less than 0,3 s.

In this situation the validation should be part of the normal operation and shall be executed immediately after startup. This implies that "*lampOn*" might be incorrect for a maximum of 0,3 s until the validation is finished.

NOTE Figure 11 also illustrates the effect of "*lampFailure*" on "*lampOn*" and "*fadeRunning*" bits for the scenarios described above.

For the light source type "unknown light source type", there shall be support for this bit. If there is no support for the lamp failure bit, this shall be made explicit in the corresponding part of the IEC 62386-2xx series.

For the light source type “no light source”, there may be support for this bit. Testing is excluded for this light source type. If there is support for the lamp failure bit, this shall be made explicit in the corresponding part of the IEC 62386-2xx series.

9.16.4 Bit 2: lamp on

“lampOn” shall be set to FALSE when the lamp is off, during startup, and in case of total lamp failure. In all other cases it shall be set to TRUE.

9.16.5 Bit 3: limit error

If the last requested target level has been modified in accordance with *“minLevel”* or *“maxLevel”* limitations, or *“targetLevel”* has been modified due to a change of *“minLevel”* or *“maxLevel”,* “*limitError*” shall be set to TRUE.

If the last target level requested by “DAPC (*level*)” equals “MASK”, “*limitError*” shall not change.

In all other cases “*limitError*” shall be set to FALSE.

9.16.6 Bit 4: fade running

“fadeRunning” shall be set to FALSE except for the time during which the fade timer is running. *“fadeRunning”* shall be set to TRUE from the beginning of the fade (after startup) until the end of the fade time, regardless of whether *“targetLevel”* and *“actualLevel”* reach the same level.

9.16.7 Bit 5: reset state

“resetState” shall be set to TRUE if all the NVM variables mentioned in Table 14 except “*lastLightLevel*” are at their reset value. The NVM variables that are marked with ‘no change’ in the reset value column shall not be considered. NVM variables defined in implemented Parts 2xx shall be included.

In all other cases the bit shall be set to FALSE.

9.16.8 Bit 6: missing short address

This bit indicates whether a short address has been assigned to the gear, by checking *“shortAddress”*. The bit shall be TRUE if *“shortAddress”* = MASK.

In all other cases the bit shall be set to FALSE.

9.16.9 Bit 7: power cycle seen

“powerCycleSeen” shall be set to TRUE after an external power cycle (see IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018, 4.11) has occurred.

“powerCycleSeen” shall be set to FALSE once one of the following commands has been executed:

“RESET”, “DAPC (*level*)”, “OFF”, “UP”, “DOWN”, “STEP UP”, “STEP DOWN”, “RECALL MAX LEVEL”, “RECALL MIN LEVEL”, “GO TO LAST ACTIVE LEVEL”, “STEP DOWN AND OFF”, “ON AND STEP UP”, “CONTINUOUS UP”, “CONTINUOUS DOWN”, “GO TO SCENE (*sceneNumber*)”.

9.17 Non-volatile memory

Physical non-volatile memory typically supports a limited number of write cycles. Since many variables are NVM type, the physical limitations need some attention.

A control gear should store NVM variables in such a way that their content is never lost and the intended lifetime of the device can be reached. This means that it may not be possible to physically write every change in a variable immediately. There may be situations in which the control gear is not able to physically write the variables to NVM, especially if a particular NVM variable is changed very frequently.

Since the application controller cannot know the control gear's internal mechanism for physically saving persistent variables, the instruction "SAVE PERSISTENT VARIABLES" is defined to force the control gear to physically write all variables of type NVM to memory. This command is an addition to the normal writing of NVM variables. Its intended use is to ensure that important changes made by an application controller cannot be lost, e.g. after assigning all short addresses or setting other important (and stable) configuration data. Clearly it is not intended to be used after every level change. Typically, this command is used only a handful of times for an entire installation.

NOTE Typically the command can be used a few thousand times before causing physical damage to the control gear's NVM.

Physically saving the variables in response to the instruction shall take at most 300 ms to complete. While the saving operation is on-going, the light output may fluctuate and the control gear may or may not respond to any command. However, until the operation is complete, the value of the affected variables may be undefined. Moreover, if the light is off when the instruction is executed, the light shall stay off; in this case, no flicker shall be visible.

The light output may not fluctuate during saving operations unless these are triggered by this command.

An application controller can trigger the save operation using the "SAVE PERSISTENT VARIABLES" instruction and should wait at least 350 ms to ensure all gear have finished the operation.

9.18 Device types and features

Commands with their opcode in the range 0xE0 to 0xFF are reserved for special device types or features. Each device type/feature re-defines these commands, except for the command with opcode 0xFF ("QUERY EXTENDED VERSION NUMBER").

The device type/feature specific command set can be selected by the instruction "ENABLE DEVICE TYPE (*data*)".

This instruction shall select the device type/feature for which only the next application extended command (refer to 11.6) is valid. Executing this instruction shall cancel any previous selection of a device type.

The enabling of the device type/feature shall be cancelled upon execution of the next command, and that command shall be executed according to its specification, regardless of whether it is an application extended command or not.

A control gear shall not react to commands which belong to the application extended commands if *data* equals MASK, 254 or represents a device type/feature not supported by this control gear.

The device types/features shall be coded as specified in the particular parts the IEC 62386-2xx series.

An application controller can check which device types/features are supported by the control gear. “QUERY DEVICE TYPE” reports the supported device type/features. If more than one device type/feature is supported, “QUERY DEVICE TYPE” reports MASK. In that case, the application controller can check all supported device types/features by repeating “QUERY NEXT DEVICE TYPE” until 254 is received as an answer. Issuing “QUERY DEVICE TYPE” automatically ensures that the first supported device type/feature will be reported by “QUERY NEXT DEVICE TYPE”.

To check the version number of the supported device types/features, the application controller can send “ENABLE DEVICE TYPE (*data*) followed by “QUERY EXTENDED VERSION NUMBER”. This will report the version number of that specific device type/feature implementation.

Application controllers should be able to identify individual gear and store the relationship between the gear's individual address and its device types/features in persistent memory.

9.19 Using scenes

A control gear shall support the use of 16 scenes. The following commands shall be supported:

“GO TO SCENE (*sceneNumber*)”, “REMOVE FROM SCENE (*sceneX*)”, “QUERY SCENE LEVEL (*sceneX*)”, and “SET SCENE (*DTR0, sceneX*)”.

These commands actually comprise 16 commands each, one for each scene. This is accomplished by selecting a block of 16 consecutive opcodes. The number of the scene to be used can thus easily be calculated.

Upon execution of one of the scene commands, *sceneNumber* shall be derived from the opcode: *sceneNumber* = opcode – opcodeBase. This identifies the scene to be used. The opcodeBase can be found in Table 13.

Table 13 – Scenes

Command	opcodeBase	Opcode range
GO TO SCENE (*sceneNumber)*	0x10	[0x10,0x1F]
REMOVE FROM SCENE (*sceneX*)	0x50	[0x50,0x5F]
QUERY SCENE LEVEL (*sceneX*)	0xB0	[0xB0,0xBF]
SET SCENE (*DTR0, sceneX*)	0x40	[0x40,0x4F]

The *“sceneX”* variable also stands for 16 individual variables, where *X* equals *sceneNumber* in the range of [0,15].

On accepting command “GO TO SCENE (*sceneNumber*)” the reaction of the control gear shall depend upon the current value of “*sceneX*”, where X is derived from *sceneNumber*. If “*sceneX*” equals MASK, “*targetLevel*” shall not be affected. Otherwise, the control gear shall behave exactly as if “DAPC (*level*)” had been accepted with level equal to "*sceneX*".

NOTE Using “DAPC (*level*)” implies the transition is made using the set fade time.

10 Declaration of variables

The default values, the reset values, power on values, the range of validity and the type of memory of the defined variables shall be as given in Table 14.

The variables that are declared in this clause shall not be made available for writing through a memory bank.

Table 14 – Declaration of variables

VARIABLE	DEFAULT VALUE (factory)	RESET VALUE	POWER ON VALUE	RANGE OF VALIDITY	MEMORY TYPE
"actualLevel"	[a]	0xFE	0x00	0, [*"minLevel"*, *"maxLevel"*]	RAM
"targetLevel"	[a]	0xFE	See 9.13 Power on	0, [*"minLevel"*, *"maxLevel"*]	RAM
"lastActiveLevel"	[a]	0xFE	*"maxLevel"*	[*"minLevel"*, *"maxLevel"*]	RAM
"lastLightLevel"	0xFE	0xFE [c]	no change	0, [*"minLevel"*, *"maxLevel"*]	NVM
"powerOnLevel"	0xFE	0xFE	no change	[0,0xFF]	NVM
"systemFailureLevel"	0xFE	0xFE	no change	[0,0xFF]	NVM
"minLevel"	PHM	PHM	no change	[PHM,*"maxLevel"*]	NVM
"maxLevel"	0xFE	0xFE	no change	[*"minLevel"*,0xFE]	NVM
"fadeRate"	7	7	no change	[1,0xF]	NVM
"fadeTime"	0	0	no change	[0,0xF]	NVM
"extendedFadeTimeBase"	0	0	no change	[0,1111b]	NVM
"extendedFadeTimeMultiplier"	0	0	no change	[0,100b]	NVM
"shortAddress"	MASK (no address)	no change	no change	[0,63], MASK	NVM
"searchAddress"	[a]	0xFF FF FF	0xFF FF FF	[0,0xFF FF FF]	RAM
"randomAddress"	0xFF FF FF	0xFF FF FF	no change	[0,0xFF FF FF]	NVM
"operatingMode"	factory burn-in	no change	no change	0,[0x80,0xFF]	NVM
"initialisationState"	[a]	no change	DISABLED	[ENABLED, DISABLED, WITHDRAWN]	RAM
"writeEnableState"	[a]	DISABLED	DISABLED	[ENABLED, DISABLED]	RAM
"controlGearFailure"	[a]	[b]	FALSE [d]	[TRUE, FALSE]	RAM
"lampFailure"	[a]	[b]	FALSE [d]	[TRUE, FALSE]	RAM
"lampOn"	[a]	[b]	FALSE	[TRUE, FALSE]	RAM
"limitError"	[a]	FALSE	FALSE [d]	[TRUE, FALSE]	RAM
"fadeRunning"	[a]	FALSE	FALSE	[TRUE, FALSE]	RAM
"resetState"	TRUE	TRUE	TRUE [d]	[TRUE, FALSE]	RAM
"powerCycleSeen"	[a]	FALSE	TRUE	[TRUE, FALSE]	RAM
"gearGroups"	0x00 00 (no group)	0x00 00 (no group)	no change	[0,0xFF FF]	NVM
"sceneX" [e]	MASK	MASK	no change	[0,0xFF]	NVM

VARIABLE	DEFAULT VALUE (factory)	RESET VALUE	POWER ON VALUE	RANGE OF VALIDITY	MEMORY TYPE
"DTR0"	[a]	no change	0x00	[0,0xFF]	RAM
"DTR1"	[a]	no change	0x00	[0,0xFF]	RAM
"DTR2"	[a]	no change	0x00	[0,0xFF]	RAM
PHM	factory burn-in	no change	no change	[1,0xFE]	ROM
"versionNumber"	2.1	no change	no change	00001001b	ROM

[a] Not applicable.

[b] The value could change as a consequence of the RESET command execution.

[c] This NVM variable is excluded for "*resetState*".

[d] The value should reflect the actual situation as soon as possible.

[e] X is in the range 0x0 to 0xF, effectively there is one variable for each of the 16 scenes.

11 Definition of commands

11.1 General

Unused opcodes are reserved for future needs.

11.2 Overview sheets

Table 15 gives an overview of the standard commands. The special commands overview can be found in Table 16.

Table 15 – Standard commands

Command name	Address byte		Opcode byte	Ed. 1 cmd number	DTR0	DTR1	DTR2	Answer	Send twice	References	Command reference
	See 7.2.2	Selector bit									
DAPC (*level*)	*Device*	0	*level*	-						9.4, 9.7.3, 9.8	11.3.1
OFF	*Device*	1	0x00	0						9.7.2	11.3.2
UP	*Device*	1	0x01	1						9.7.3	11.3.3
DOWN	*Device*	1	0x02	2						9.7.3	11.3.4
STEP UP	*Device*	1	0x03	3						9.7.2	11.3.5
STEP DOWN	*Device*	1	0x04	4						9.7.2	11.3.6
RECALL MAX LEVEL	*Device*	1	0x05	5						9.7.2, 9.14.2	11.3.7
RECALL MIN LEVEL	*Device*	1	0x06	6						9.7.2, 9.14.2	11.3.8
STEP DOWN AND OFF	*Device*	1	0x07	7						9.7.2	11.3.9
ON AND STEP UP	*Device*	1	0x08	8						9.7.2	11.3.10
ENABLE DAPC SEQUENCE	*Device*	1	0x09	9						9.8	11.3.11
GO TO LAST ACTIVE LEVEL	*Device*	1	0x0A							9.7.3	11.3.12

Command name	Address byte		Opcode byte	Ed. 1 cmd number	DTR0	DTR1	DTR2	Answer	Send twice	References	Command reference
	See 7.2.2	Selector bit									
CONTINUOUS UP	*Device*	1	0x0B							9.7.3	11.3.14
CONTINUOUS DOWN	*Device*	1	0x0C							9.7.3	11.3.15
GO TO SCENE (*sceneNumber*) [a]	*Device*	1	0x10 + *sceneNumber*	16 – 31						9.7.3, 9.19	11.3.13
RESET	*Device*	1	0x20	32					✓	9.11.1, 10	11.4.2
STORE ACTUAL LEVEL IN DTR0	*Device*	1	0x21	33	✓				✓		11.4.3
SAVE PERSISTENT VARIABLES	*Device*	1	0x22						✓	9.17, 10	11.4.4
SET OPERATING MODE (*DTR0*)	*Device*	1	0x23		✓				✓	9.9.4	11.4.5
RESET MEMORY BANK (*DTR0*)	*Device*	1	0x24		✓				✓	9.11.2	11.4.6
IDENTIFY DEVICE	*Device*	1	0x25						✓	9.14.2	11.4.7
SET MAX LEVEL (*DTR0*)	*Device*	1	0x2A	42	✓				✓	9.6	11.4.7
SET MIN LEVEL (*DTR0*)	*Device*	1	0x2B	43	✓				✓	9.6	11.4.9
SET SYSTEM FAILURE LEVEL (*DTR0*)	*Device*	1	0x2C	44	✓				✓	9.12	11.4.10
SET POWER ON LEVEL (*DTR0*)	*Device*	1	0x2D	45	✓				✓	9.13	11.4.11
SET FADE TIME (*DTR0*)	*Device*	1	0x2E	46	✓				✓	9.5.2	11.4.12
SET FADE RATE (*DTR0*)	*Device*	1	0x2F	47	✓				✓	0	11.4.13
SET EXTENDED FADE TIME (*DTR0*)	*Device*	1	0x30		✓				✓	0	11.4.14
SET SCENE (*DTR0, sceneX*) [a]	*Device*	1	0x40 + *sceneNumber*	64 – 79	✓				✓	9.19	11.4.14
REMOVE FROM SCENE (*sceneX*) [a]	*Device*	1	0x50 + *sceneNumber*	80 – 95					✓	9.19	11.4.16
ADD TO GROUP (*group*) [a]	*Device*	1	0x60 + *group*	96 – 111					✓		11.4.17
REMOVE FROM GROUP (*group*) [a]	*Device*	1	0x70 + *group*	112 – 127					✓		11.4.18

Command name	Address byte		Opcode byte	Ed. 1 cmd number	DTR0	DTR1	DTR2	Answer	Send twice	References	Command reference
	See 7.2.2	Selector bit									
SET SHORT ADDRESS (*DTR0*)	*Device*	1	0x80	128	✓				✓	9.14.4	11.4.19
ENABLE WRITE MEMORY	*Device*	1	0x81	129					✓	9.10	11.4.20
QUERY STATUS	*Device*	1	0x90	144				✓		9.16	11.5.2
QUERY CONTROL GEAR PRESENT	*Device*	1	0x91	145				✓			11.5.3
QUERY LAMP FAILURE	*Device*	1	0x92	146				✓			11.5.4
QUERY LAMP POWER ON	*Device*	1	0x93	147				✓			11.5.6
QUERY LIMIT ERROR	*Device*	1	0x94	148				✓			11.5.7
QUERY RESET STATE	*Device*	1	0x95	149				✓			11.5.8
QUERY MISSING SHORT ADDRESS	*Device*	1	0x96	150				✓		9.14.2	11.5.9
QUERY VERSION NUMBER	*Device*	1	0x97	151				✓			11.5.10
QUERY CONTENT DTR0	*Device*	1	0x98	152	✓			✓		9.10	11.5.11
QUERY DEVICE TYPE	*Device*	1	0x99	153				✓		9.18	11.5.12
QUERY PHYSICAL MINIMUM	*Device*	1	0x9A	154				✓			11.5.13
QUERY POWER FAILURE	*Device*	1	0x9B	155				✓			11.5.15
QUERY CONTENT DTR1	*Device*	1	0x9C	156		✓		✓		9.10	11.5.16
QUERY CONTENT DTR2	*Device*	1	0x9D	157			✓	✓			11.5.17
QUERY OPERATING MODE	*Device*	1	0x9E					✓		9.9.4	11.5.18
QUERY LIGHT SOURCE TYPE	*Device*	1	0x9F		✓	✓	✓	✓			11.5.19
QUERY ACTUAL LEVEL	*Device*	1	0xA0	160				✓			11.5.20
QUERY MAX LEVEL	*Device*	1	0xA1	161				✓			11.5.21
QUERY MIN LEVEL	*Device*	1	0xA2	162				✓			11.5.22

Command name	Address byte		Opcode byte	Ed. 1 cmd number	DTR0	DTR1	DTR2	Answer	Send twice	References	Command reference
	See 7.2.2	Selector bit									
QUERY POWER ON LEVEL	*Device*	1	0xA3	163				✓		9.13	11.5.23
QUERY SYSTEM FAILURE LEVEL	*Device*	1	0xA4	164				✓		9.12	11.5.24
QUERY FADE TIME/FADE RATE	*Device*	1	0xA5	165				✓			11.5.25
QUERY MANUFACTURER SPECIFIC MODE	*Device*	1	0xA6					✓		9.9	11.5.27
QUERY NEXT DEVICE TYPE	Device	1	0xA7					✓		9.18	11.5.13
QUERY EXTENDED FADE TIME	Device	1	0xA8					✓		0	11.5.26
QUERY CONTROL GEAR FAILURE	Device	1	0xAA					✓		9.16.2	11.5.4
QUERY SCENE LEVEL (*sceneX*) [a]	*Device*	1	0xB0 + *sceneNumber*	176 – 191				✓		9.19	11.5.28
QUERY GROUPS 0-7	*Device*	1	0xC0	192				✓			11.5.29
QUERY GROUPS 8-15	*Device*	1	0xC1	193				✓			11.5.30
QUERY RANDOM ADDRESS (H)	*Device*	1	0xC2	194				✓			11.5.31
QUERY RANDOM ADDRESS (M)	*Device*	1	0xC3	195				✓			11.5.32
QUERY RANDOM ADDRESS (L)	*Device*	1	0xC4	196				✓			11.5.33
READ MEMORY LOCATION (*DTR1*, *DTR0*)	*Device*	1	0xC5	197	✓	✓		✓		9.10	11.5.34
Application extended commands	*Device*	1	0xE0 – 0xFE	224 – 254	?	?	?	?	?	9.18	11.6
QUERY EXTENDED VERSION NUMBER	*Device*	1	0xFF	255				✓			11.6.2

[a] There is one command per scene, so there are actually 16 commands for scenes 0 – 5. Analogue for the 16 group commands.

Table 16 – Special commands

Command name	Address byte	Opcode byte	Ed.1 cmd nr	DTR0	DTR1	DTR2	Answer	Send twice	References	Command reference
TERMINATE	0xA1	0x00	256						9.14.2	11.7.1
DTR0 (*data*)	0xA3	*data*	257	✓					9.10	11.7.3
INITIALISE (*device*)	0xA5	*device*	258					✓	9.14.2	11.7.4
RANDOMISE	0xA7	0x00	259					✓	9.14.2	11.7.5
COMPARE	0xA9	0x00	260				✓		9.14.2	11.7.6
WITHDRAW	0xAB	0x00	261						9.14.2	11.7.7
PING	0xAD	0x00								11.7.19
SEARCHADDRH (*data*)	0xB1	*data*	264						9.14.2	11.7.8
SEARCHADDRM (*data*)	0xB3	*data*	265						9.14.2	11.7.9
SEARCHADDRL (*data*)	0xB5	*data*	266						9.14.2	11.7.10
PROGRAM SHORT ADDRESS (*data*)	0xB7	*data*	267						9.14.2	11.7.11
VERIFY SHORT ADDRESS (*data*)	0xB9	*data*	268				✓		9.14.2	11.7.12
QUERY SHORT ADDRESS	0xBB	0x00	269				✓		9.14.2	11.7.13
ENABLE DEVICE TYPE (*data*)	0xC1	*data*	272						9.14.2	11.7.14
DTR1 (*data*)	0xC3	*data*	273		✓				9.10	0
DTR2 (*data*)	0xC5	*data*	274			✓				11.7.16
WRITE MEMORY LOCATION (*DTR1, DTR0, data*)	0xC7	*data*	275	✓	✓		✓		9.10	11.7.17
WRITE MEMORY LOCATION – NO REPLY (*DTR1, DTR0, data*)	0xC9	*data*		✓	✓				9.10	11.7.18

11.3 Level instructions

11.3.1 DAPC (*level*)

Upon execution of "DAPC (*level*)" (direct arc power control), *"targetLevel"* shall be calculated on the basis of *"level"*.

The transition from *"actualLevel"* to *"targetLevel"* shall start using the applicable fade time.

Refer to subclauses 9.4, 9.7.3 and 9.13 for further information.

11.3.2 OFF

"targetLevel" shall be set to 0x00 and the lamp(s) shall switch off.

The transition from *"actualLevel"* to *"targetLevel"* shall be immediate and the light output shall be adjusted as quickly as possible.

Refer to subclause 9.7.2 for further information.

11.3.3 UP

Dim up using a 200 ms fade with the set fade rate. *"targetLevel"* shall be calculated on the basis of *"actualLevel"* and the set fade rate.

To ensure that there is a reaction to the command, at least one step (final "*targetLevel*" = calculated "*targetLevel*"+1) shall be made upon execution of the first command of an iteration. After that first step, the next steps shall be executed using the specified fade rate while the fading is running. Every "UP" instruction executed as a part of an iteration shall cause the 200 ms fade to be restarted and *"targetLevel"* to be recalculated on the basis of *"actualLevel"* and the set fade rate.

There shall be no change to *"actualLevel"* if *"actualLevel"* is at *"maxLevel"* or 0x00.

Refer to subclauses 9.5.1, 9.7.3 and 9.8.2 for further information.

11.3.4 DOWN

Dim down using a 200 ms fade with the set fade rate. *"targetLevel"* shall be calculated on the basis of *"actualLevel"* and the set fade rate.

To ensure that there is a reaction to the command, at least one step (final "*targetLevel*" = calculated "*targetLevel*"-1) shall be made upon execution of the first command of an iteration. After that first step, the next steps shall be executed using the specified fade rate while the fading is running. Every "DOWN" instruction executed as a part of an iteration shall cause the 200 ms fade to be restarted and *"targetLevel"* to be recalculated on the basis of *"actualLevel"* and the set fade rate.

There shall be no change to *"actualLevel"* if *"actualLevel"* is at *"minLevel"* or 0x00.

Refer to subclauses 9.5.1, 9.7.3 and 9.8.2 for further information.

11.3.5 STEP UP

"targetLevel" shall be set to:

- if *"targetLevel"* = 0: 0x00

- if *“minLevel”* ≤ *“targetLevel”* < *“maxLevel”*: *“targetLevel”*+1
- if *“targetLevel”* = *“maxLevel”*: *“maxLevel”*

The transition from *“actualLevel”* to *“targetLevel”* shall be immediately and the light output shall be adjusted as quickly as possible.

Refer to subclauses 9.4 and 9.5.9 for further information.

11.3.6 STEP DOWN

“targetLevel” shall be set to:

- if *“targetLevel”* = 0: 0x00
- if *“minLevel”* < *“targetLevel”* ≤ *“maxLevel”*: *“targetLevel”*-1
- if *“targetLevel”* = *“minLevel”*: *“minLevel”*

The transition from *“actualLevel”* to *“targetLevel”* shall be immediately and the light output shall be adjusted as quickly as possible.

Refer to subclauses 9.4 and 9.5.9 for further information.

11.3.7 RECALL MAX LEVEL

When the *“initialisationState”* is DISABLED, *“targetLevel”* and *“actualLevel”* shall be set to *“maxLevel”* immediately and the light output shall be adjusted as quickly as possible.

Refer to 9.7.2 for further information.

When the *“initialisationState”* is not DISABLED, the control gear shall set “*actualLevel*” and “*targetLevel*” to “*maxLevel*”, and then adjust the light output as quickly as possible to 100 % temporarily ignoring “*maxLevel*” and “*actualLevel*”.

If the device is unable to visually identify itself in this way, the control gear shall execute “IDENTIFY DEVICE”, starting or re-triggering the identification procedure.

NOTE It is acceptable for the process of identifying individual control gear to depend upon RECALL MAX LEVEL and RECALL MIN LEVEL commands being executed in an alternating sequence.

During identification no variables shall be affected except when explicitly stated otherwise. Where appropriate, variables can be temporarily ignored, so that after the identification has ended, there are no side effects.

Identification shall be stopped immediately when the *“initialisationState”* changes to DISABLED and upon execution of any instruction other than INITIALISE (*device*), RECALL MIN LEVEL, RECALL MAX LEVEL or IDENTIFY DEVICE.

When the *“initialisationState”* changes to DISABLED, the identification shall stop immediately.

Refer to 9.14.3 for further information.

11.3.8 RECALL MIN LEVEL

When the *“initialisationState”* is DISABLED, *“targetLevel”* and *“actualLevel”* shall be set to *“minLevel”* immediately and the light output shall be adjusted as quickly as possible.

Refer to 9.7.2 for further information.

When *"initialisationState"* is not DISABLED, the control gear shall set "*actualLevel*" and "*targetLevel*" to "*minLevel*" and then adjust the light output as quickly as possible to its PHM level temporarily ignoring "*minLevel*" and "*actualLevel*". If, however, PHM is not visibly significantly different from 100 %, then the lamp shall be temporarily switched off instead.

If the device is unable to visually identify itself in this way, the control gear shall execute "IDENTIFY DEVICE", starting or re-triggering the identification procedure.

NOTE It is acceptable for the process of identifying individual control gear to depend upon RECALL MAX LEVEL and RECALL MIN LEVEL commands being executed in an alternating sequence.

During identification no variables shall be affected except when explicitly stated otherwise. Where appropriate, variables can be temporarily ignored, so that after the identification has ended, there are no side effects.

Identification shall be stopped immediately when the *"initialisationState"* changes to DISABLED and upon execution of any instruction other than INITIALISE (*device*), RECALL MIN LEVEL, RECALL MAX LEVEL or IDENTIFY DEVICE.

When the *"initialisationState"* changes to DISABLED, identification shall stop immediately.

Refer to 9.14.3 for further information.

11.3.9 STEP DOWN AND OFF

"targetLevel" shall be set to:

- if *"targetLevel"* = 0: 0x00
- if *"minLevel"* < *"targetLevel"* ≤ *"maxLevel": "targetLevel"*-1
- if *"targetLevel" = "minLevel"*: 0x00

The transition from *"actualLevel"* to *"targetLevel"* shall be immediately and the light output shall be adjusted as quickly as possible.

Refer to subclauses 9.4 and 9.5.9 for further information.

11.3.10 ON AND STEP UP

"targetLevel" shall be set to:

- if *"targetLevel"* = 0: *"minLevel"*
- if *"minLevel"* ≤ *"targetLevel"* < *"maxLevel": "targetLevel"*+1
- if *"targetLevel"* ≥ *"maxLevel": "maxLevel"*

The transition from *"actualLevel"* to *"targetLevel"* shall be immediately and the light output shall be adjusted as quickly as possible.

Refer to subclauses 9.4 and 9.5.9 for further information.

11.3.11 ENABLE DAPC SEQUENCE

Indicates the start of a command iteration of "DAPC (*level*)" commands.

Refer to subclause 9.8.3 for further information.

11.3.12 GO TO LAST ACTIVE LEVEL

Upon execution of this command *"targetLevel"* shall be calculated based on *"lastActiveLevel"*.

The transition from *"actualLevel"* to *"targetLevel"* shall start using the set fade time.

Refer to subclauses 9.7.3 and 9.4 for further information.

11.3.14 CONTINUOUS UP

Dim up using the set fade rate. "*targetLevel*" shall be set to "*maxLevel*" and a fade shall be started using the set fade rate. The fade shall stop when "*maxLevel*" is reached.

There shall be no change to "*actualLevel*" if "*actualLevel*" is at "*maxLevel*" or 0x00.

Refer to 9.7.3 and 9.5.6.2 for further information.

11.3.15 CONTINUOUS DOWN

Dim down using the set fade rate. "*targetLevel*" shall be set to "*minLevel*" and a fade shall be started using the set fade rate. The fade shall stop when "*minLevel*" is reached.

There shall be no change to "*actualLevel*" if "*actualLevel*" is at "*minLevel*" or 0x00.

Refer to 9.7.3 and 9.5.6.2 for further information.

11.3.13 GO TO SCENE (*sceneNumber*)

The control gear shall react depending on the actual value of *"sceneX"* where *X* is derived from *sceneNumber*:

- if *"sceneX"* = MASK: the command shall not affect "*targetLevel*";
- in all other cases: internally "DAPC (*level*)", with *level* equal to *"sceneX"* shall be executed.

NOTE Using "DAPC (*level*)" implies the transition is made using the set fade time.

Refer to subclauses 9.19 and 11.3.1 for further information.

11.4 Configuration instructions

11.4.1 General

Device configuration instructions are used to change the configuration and/or the mode of operation of the control gear. For this reason a device configuration instruction shall be discarded, unless it is accepted twice according to the requirements as stated in IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018, 9.3.

Unless explicitly stated otherwise in the description of particular device configuration instruction, the following holds:

- The instruction shall be ignored if so required by the provisions of subclause 9.7 of this standard.
- The control gear shall not reply to the instruction.

11.4.2 RESET

All variables shall be changed to their reset values. Control gear shall start to react properly to commands no later than 300 ms after the execution of the instruction has been started.

If during a reset mains power fails, it is not guaranteed that “RESET” is completed.

Refer to subclause 9.11.1 and Table 14 for further information.

11.4.3 STORE ACTUAL LEVEL IN DTR0

The *“actualLevel”* shall be stored in *“DTR0”*.

11.4.4 SAVE PERSISTENT VARIABLES

The control gear shall physically store all variables identified in Table 14 as non-volatile memory (NVM). This shall include all application extended NVM variables defined in the applicable parts 2xx.

The control gear might not react to commands after execution of this command. Control gear shall start to react properly to commands no later than 300 ms after the execution of the instruction has been started.

During processing of this command, the light output may fluctuate. After processing is completed, the light output shall be at the level as expected before the execution of this command, based on *“targetLevel”* and the transition that was active (if any).

This command is recommended to be used typically after commissioning. Due to the limited number of write-cycles of persistent memory and due to the fact that there might be a visible reaction, the control devices should limit the use of this command.

As there might be visual artefacts, it is recommended to use this command only during the off state.

Refer to Table 14 and subclause 9.17 for further information.

11.4.5 SET OPERATING MODE (*DTR0*)

“operatingMode” shall be set *“DTR0”*.

If *“DTR0”* does not correspond to an implemented operating mode, the command shall be discarded.

Refer to subclause 9.9 for further information.

11.4.6 RESET MEMORY BANK (*DTR0*)

The command shall trigger the process to change the memory bank content to its reset values as follows:

- if *“DTR0”* = 0: all implemented and unlocked memory banks except memory bank 0 shall be reset
- in all other cases: the memory bank identified by *“DTR0”* shall be reset provided it is implemented and unlocked

A memory bank needs to be unlocked to allow both lockable and non-lockable locations to be reset.

Control gear shall start to react properly to commands no later than 10 s after the execution of the instruction has been started.

Refer to subclause 9.11.2 for further information.

11.4.7 IDENTIFY DEVICE

The control gear shall start or restart a 10 s ±1 s timer. While the timer is running, a procedure shall run which enables an observer to distinguish any control gear running this process from any devices (of the same type) which are not running it. If the timer expires, identification shall stop.

During identification no variables shall be affected except when explicitly stated otherwise. Where appropriate, variables can be temporarily ignored, so that after the identification has ended, there are no side effects.

When identification is active, the light output may be at any level between off and 100 %, MIN, MAX and "*actualLevel*" being in effect temporarily ignored.

Identification shall be stopped immediately upon execution of any instruction other than INITIALISE (*device*), RECALL MIN LEVEL, RECALL MAX LEVEL or IDENTIFY DEVICE.

While identification is active, the control gear shall, without interrupting the identification procedure:

- on RECALL MIN LEVEL: set "*actualLevel*" and "*targetLevel*" to "*minLevel*";
- on RECALL MAX LEVEL: set "*actualLevel*" and "*targetLevel*" to "*maxLevel*".

When identification is stopped by an application controller, the corresponding timer shall be cancelled immediately.

After identification has stopped, the light output shall be adjusted as quickly as possible to reflect "*actualLevel*" and the command shall be executed (if applicable).

Identification can be used during commissioning in that it allows the installer to e.g. allocate the particular identified device to a particular device group.

The indication can be done e.g. by flashing a LED, by producing a sound or other visual or audible means. The exact process used to identify is manufacturer specific and should be described in the manual.

NOTE The application controller can also stop the identification process using a "RESET" command.

Refer to subclause 9.14.3 for further information.

11.4.8 SET MAX LEVEL (*DTR0*)

"maxLevel" shall be set to:

- if *"minLevel"* ≥ *"DTR0"*: *"minLevel"*
- if *"DTR0"* = MASK: 0xFE
- in all other cases: *"DTR0"*

If as a result of setting a new max level *"actualLevel"* > *"maxLevel"*, *"targetLevel"* shall be calculated on the basis of *"maxLevel"*. The transition from *"actualLevel"* to *"targetLevel"* shall start immediately and the light output shall be adjusted as quickly as possible.

Refer to subclause 9.7.2 for further information.

11.4.9 SET MIN LEVEL (*DTR0*)

"minLevel" shall be set to:

- if 0 ≤ *"DTR0"* ≤ PHM: PHM
- if *"DTR0"* ≥ *"maxLevel"* or MASK: *"maxLevel"*
- in all other cases: *"DTR0"*

If "*actualLevel*" > 0 and as a result of setting a new min level *"actualLevel"* < *"minLevel"*, *"targetLevel"* shall be calculated on the basis of *"minLevel"*. The transition from *"actualLevel"* to *"targetLevel"* shall be immediately and the light output shall be adjusted as quickly as possible. Refer to subclause 9.7.2 for further information.

11.4.10 SET SYSTEM FAILURE LEVEL (*DTR0*)

"systemFailureLevel" shall be set to *"DTR0"*.

Refer to subclause 9.12 for further information.

11.4.11 SET POWER ON LEVEL (*DTR0*)

"powerOnLevel" shall be set to *"DTR0"*.

Refer to subclause 9.13 for further information.

11.4.12 SET FADE TIME (*DTR0*)

The *"fadeTime"* shall be set to a value according to the following steps:

- if *"DTR0"* > 15: 15
- in all other cases: *"DTR0"*

If "*fadeTime"* is not equal to 0, the fade time shall be calculated on the basis of *"fadeTime"*. If "*fadeTime"* is equal to 0, the extended fade time shall be used.

If a new fade time is stored during a running fade process, this process shall be finished first before the new value is used in the following fade.

Refer to subclauses 9.5, 9.7.3, 11.4.14 and 11.7.19 for further information.

11.4.13 SET FADE RATE (*DTR0*)

The *"fadeRate"* shall be set to a value according to the following steps:

- if *"DTR0"* > 15: 15
- if *"DTR0"* = 0: 1
- in all other cases: *"DTR0"*

The fade rate shall be calculated on the basis of *"fadeRate"*. If a new fade rate is stored during a running fade process, this process shall be finished first before the new value is used in the following fade.

Refer to subclause 9.5 and 9.7.3 for further information.

11.4.14 SET EXTENDED FADE TIME (*DTR0*)

The *"extendedFadeTimeBase"* and "*extendedFadeTimeMultiplier*" shall be set to a value according to the following steps:

- If "*DTR0"* > 0x4F (0100 1111b):
  - "*extendedFadeTimeBase*" shall be set to 0;

– "*extendedFadeTimeMultiplier*" shall be set to 0.

Effectively selecting a fade as quickly as possible.

- For all other cases:
  – "*extendedFadeTimeBase*" shall be set to AAAAb where *"DTR0"* = xxxxAAAAb;
  – "*extendedFadeTimeMultiplier*" shall be set to YYYb where *"DTR0"* = xYYYxxxxb.
- The fade time shall be calculated by multiplying the base value and the multiplier.

If a new fade time is stored during a running fade process, this process shall be finished first before the new value is used in the following fade.

Refer to subclause 0 for further information.

11.4.15 SET SCENE (*DTR0, sceneX*)

This command actually comprises 16 commands, one for each scene. This is accomplished by selecting a block of 16 consecutive opcodes.

Upon execution of "SET SCENE (*DTR0, sceneX*)", the scene number shall be derived from the opcode:*sceneNumber* = opcode – 0x40. This identifies the "*sceneX"* to be used.

"sceneX" shall be set to *"DTR0"*.

Refer to subclause 9.19 for further information

11.4.16 REMOVE FROM SCENE (*sceneX*)

This command actually comprises 16 commands, one for each scene. This is accomplished by selecting a block of 16 consecutive opcodes.

Upon execution of "REMOVE FROM SCENE (*sceneX*), the scene number shall be derived from the opcode: *sceneNumber* = opcode – 0x50. This identifies the "*sceneX"* to be used.

"sceneX" shall be set to MASK. This effectively removes the control gear as member from the scene.

Refer to subclause 9.19 for further information.

11.4.17 ADD TO GROUP (*group*)

This command actually comprises 16 commands, one for each group. This is accomplished by selecting a block of 16 consecutive opcodes.

Upon execution of "ADD TO GROUP (*group*)", *group* shall be derived from the opcode: *group* = opcode – 0x60. This identifies the *group* to be used.

bit[*group*] of *"gearGroups"* shall be set to TRUE. This implies that the control gear is a member of this group.

11.4.18 REMOVE FROM GROUP (*group*)

This command actually comprises 16 commands, one for each group. This is accomplished by selecting a block of 16 consecutive opcodes.

Upon execution of "REMOVE FROM GROUP (*group*)", *group* shall be derived from the opcode: *group* = opcode – 0x70. This identifies the *group* to be used.

bit[*group*] of *"gearGroups"* shall be set to FALSE. This implies that the control gear is not a member of this group.

11.4.19 SET SHORT ADDRESS (*DTR0*)

"shortAddress" shall be set to:

- if *"DTR0"* = MASK: MASK (effectively deleting the short address);
- if *"DTR0"* = 1xxxxxxxb or xxxxxxx0b: no change;
- in all other cases (0AAAAAA1b): 00AAAAAAb.

11.4.20 ENABLE WRITE MEMORY

"writeEnableState" shall be set to ENABLED.

NOTE There is no command to explicitly disable memory write access, since any command that is not directly involved with writing into memory banks will automatically set *"writeEnableState"* to DISABLED.

Refer to subclause 9.10.5 for further information.

11.5 Queries

11.5.1 General

Queries are used to retrieve property values from a control gear. The addressed control gear returns the queried property value in a backward frame.

Unless explicitly stated otherwise in the description of a particular query, the following holds:

- The query shall be ignored if so required by the provisions of subclause 9.7.

When applicable, the query shall be discarded if any of the parameter values (in *"DTR0"*, *"DTR1"* and *"DTR2"*) are outside the range of validity of the addressed device variables, as given in Table 14.

11.5.2 QUERY STATUS

The answer shall be the status, which is formed by a combination of control gear properties.

Refer to subclause 9.16 for further information.

11.5.3 QUERY CONTROL GEAR PRESENT

The answer shall be YES.

11.5.4 QUERY CONTROL GEAR FAILURE

The answer shall be YES if *"controlGearFailure"* is TRUE and NO otherwise.

11.5.5 QUERY LAMP FAILURE

The answer shall be YES if *"lampFailure"* is TRUE and NO otherwise.

11.5.6 QUERY LAMP POWER ON

The answer shall be YES if *"lampOn"* is TRUE and NO otherwise.

11.5.7 QUERY LIMIT ERROR

The answer shall be YES if *"limitError"* is TRUE and NO otherwise.

11.5.8 QUERY RESET STATE

The answer shall be YES if *"resetState"* is TRUE and NO otherwise.

11.5.9 QUERY MISSING SHORT ADDRESS

The answer shall be YES if *"shortAddress"* is equal to MASK and NO otherwise.

NOTE Since the control gear answers only if no short address is stored, the use of the command is useful only in broadcast mode or if group addressing is used.

11.5.10 QUERY VERSION NUMBER

The answer shall be the content of memory bank 0 location 0x16.

Refer to Clause 4 and Table 9 for further information.

11.5.11 QUERY CONTENT DTR0

The answer shall be *"DTR0"*.

11.5.12 QUERY DEVICE TYPE

The answer shall be:

- if no Part 2xx is implemented: 254;
- if one device type/feature is supported: the device type/feature number;
- if more than one device type/feature is supported: MASK.

The coding of the device types/features shall be as specified in the particular Parts 2xx of IEC 62386.

Refer to subclauses 9.18 and 11.5.13 for further information.

11.5.13 QUERY NEXT DEVICE TYPE

The answer shall be:

- if directly preceded by "QUERY DEVICE TYPE", and more than one device type/feature is supported: the first and lowest device type/feature number;
- if directly preceded by "QUERY NEXT DEVICE TYPE", and not all device types/features have been reported: the next lowest device type/feature number;
- if directly preceded by "QUERY NEXT DEVICE TYPE", and all device types/features have been reported: 254;
- in all other cases: NO.

The sequence of commands shall only be accepted as long as they use the same address byte. Multi-master transmitters shall send such sequence as a transaction. The coding of the device types/features shall be as specified in the particular Parts 2xx of IEC 62386.

Refer to subclause 9.18 and 11.5.12 for further information.

11.5.14 QUERY PHYSICAL MINIMUM

The answer shall be PHM.

11.5.15 QUERY POWER FAILURE

The answer shall be YES if *"powerCycleSeen"* is TRUE and NO otherwise.

11.5.16 QUERY CONTENT DTR1

The answer shall be *"DTR1"*.

11.5.17 QUERY CONTENT DTR2

The answer shall be *"DTR2"*.

11.5.18 QUERY OPERATING MODE

The answer shall be *"operatingMode"*.

Refer to subclause 9.9 for further information.

11.5.19 QUERY LIGHT SOURCE TYPE

The answer shall be the number of the light source type given in Table 17.

Table 17 – Light source type encoding

Type of light source	Encoding	Lamp failure detection	
		Open circuit (Lamp disconnected)	Short circuit
Low pressure fluorescent	0	✓	
HID	2	✓	
Low voltage halogen	3	✓	
Incandescent	4	✓	
LED	6	✓ [a]	✓ [a]
OLED	7	✓ [a]	✓ [a]
Other than listed above	252	✓	
Unknown light source type [b]	253	✓ [d]	[d]
No light source [c]	254	[d]	[d]
Multiple light source types	MASK	✓ [e]	✓ [e]
Reserved	1, 5, [8,251]		

[a] Testing shall be done with light output of at least 5 %.

[b] Typically used in the case of signal conversion, for example 1 V to 10 V.

[c] Used in cases where no light source is connected, for example a relay.

[d] See 9.16.3.

[e] Depending on the supported light source types.

When MASK is answered the content of DTR0 shall contain a value representing the first light source type, DTR1 shall represent the second light source type, and DTR2 shall represent the third light source type.

When exactly two different light source types are available, DTR2 shall contain 254, indicating "no light source".

When more than three different light source types are available, DTR2 shall contain 255.

11.5.20 QUERY ACTUAL LEVEL

The answer shall be:

- if *"actualLevel"* = 0x00: 0x00 (see also 9.13);
- In all other cases:
  - during startup: MASK;
  - no light output (e.g. due to total lamp failure, control gear failure) while light output is expected: MASK;
  - in all other cases: *"actualLevel"*.

11.5.21 QUERY MAX LEVEL

The answer shall be *"maxLevel"*.

11.5.22 QUERY MIN LEVEL

The answer shall be *"minLevel"*.

11.5.23 QUERY POWER ON LEVEL

The answer shall be *"powerOnLevel"*.

Refer to subclause 9.12 for further information.

11.5.24 QUERY SYSTEM FAILURE LEVEL

The answer shall be *"systemFailureLevel"*.

Refer to subclause 9.12 for further information.

11.5.25 QUERY FADE TIME/FADE RATE

The answer shall be XXXX YYYYb, where XXXXb equals *"fadeTime"* and YYYYb equals *"fadeRate"*.

11.5.26 QUERY EXTENDED FADE TIME

The answer shall be 0 XXX YYYYb, where XXXb equals "*extendedFadeTimeMultiplier*" and YYYYb equals "*extendedFadeTimeBase*".

11.5.27 QUERY MANUFACTURER SPECIFIC MODE

The answer shall be YES when *"operatingMode"* is in the range [0x80,0xFF] and NO otherwise.

11.5.28 QUERY SCENE LEVEL (*sceneX*)

This command actually comprises 16 commands, one for each scene. This is accomplished by selecting a block of 16 consecutive opcodes.

Upon execution of "QUERY SCENE LEVEL (*sceneX*)", the scene number shall be derived from the opcode: *sceneNumber* = opcode – 0xB0. This identifies the "*sceneX"* to be used.

The answer shall be *"sceneX"*.

Refer to subclause 9.19 for further information.

11.5.29 QUERY GROUPS 0-7

The answer shall be *"gearGroups[7:0]"*.

The membership of groups 0-7 shall be represented as an 8-bit value, with one bit for each group. "0" shall be interpreted as not a member, and "1" shall be interpreted as member of the group. Bit[*X*] shall represent membership of group *X*, where *X* is in the range [0,7].

11.5.30 QUERY GROUPS 8-15

The answer shall be *"gearGroups[15:8]"*.

The membership of groups 8-15 shall be represented as an 8-bit value, with one bit for each group. "0" shall be interpreted as not a member, and "1" shall be interpreted as member of the group. Bit[*X*] shall represent membership of group *X*+8, where *X* is in the range [0,7].

11.5.31 QUERY RANDOM ADDRESS (H)

The answer shall be *"randomAddress[23:16]"*.

11.5.32 QUERY RANDOM ADDRESS (M)

The answer shall be *"randomAddress[15:8]"*.

11.5.33 QUERY RANDOM ADDRESS (L)

The answer shall be *"randomAddress[7:0]"*.

11.5.34 READ MEMORY LOCATION (*DTR1, DTR0*)

The query shall be discarded if the addressed memory bank is not implemented.

If executed, the answer shall be the content of the memory location identified by *"DTR0"* within memory bank *"DTR1"*.

The control gear shall answer NO if the addressed memory location is not implemented.

NOTE 1 This allows holes in the memory bank implementation.

If the addressed location is below location 0xFF, the control gear shall increment *"DTR0"* by one.

NOTE 2 This allows efficient multi-byte reading within a transaction.

Refer to subclause 9.10 for further information.

11.6 Application extended commands

11.6.1 General

"ENABLE DEVICE TYPE (*data*)" shall be executed before an application extended command to enable the correct device type/feature command set. For further requirements, see command "ENABLE DEVICE TYPE (*data*)" and 11.7.14.

If "ENABLE DEVICE TYPE (*data*)" is discarded or not received directly before an application extended command is accepted, the application extended command shall be discarded.

The definition of the extended commands is part of the application specific standards.

Refer to 9.18 for further information.

11.6.2 QUERY EXTENDED VERSION NUMBER

The answer shall be the version number of Part 2xx of this standard for the corresponding device type/feature as an 8-bit number.

The answer shall be:

- if the enabled device type/feature is not implemented: NO;
- if the enabled device type/feature is supported: the version number belonging to the device type/feature number.

Refer to subclause 9.18 for further information.

11.7 Special commands

11.7.1 General

All special mode commands shall be interpreted as instructions unless explicitly stated otherwise.

11.7.2 TERMINATE

The following processes shall be terminated immediately upon execution of this instruction:

- Initialisation, "*initialisationState*" shall be set to DISABLED.
- Identification, whether started as part of initialisation (using RECALL MAX LEVEL, RECALL MIN LEVEL) or as a standard operation (IDENTIFY DEVICE) shall be stopped.

The command could also terminate other processes as identified in the relevant 2xx parts.

Refer to subclause 9.14.2 for further information.

11.7.3 DTR0 (*data*)

"DTR0" shall be set to given *data*.

Refer to subclause 9.10 for further information.

11.7.4 INITIALISE (*device*)

This instruction shall be discarded, unless it is accepted twice according to the requirements as stated in IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018, 9.4.

Only devices matching the given *device* shall respond to the instruction, as follows in Table 18:

Table 18 – Device addressing with "INITIALISE"

Device	Responsive device(s)
0AAAAAA1b	Device(s) with *"shortAddress"* equal to 00AAAAAAb
11111111b	Control gear without *"shortAddress"* shall react
00000000b	All control gear shall react
Other	None

The instruction shall start or prolong the initialisation state, by setting *"initialisationState"* to ENABLED if it was DISABLED and (re-)trigger the timer. There shall be no answer.

Refer to subclause 9.14.2 for further information.

11.7.5 RANDOMISE

This instruction shall be discarded, unless it is accepted twice according to the requirements as stated in IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018, 9.4.

The instruction shall be discarded if *"initialisationState"* is DISABLED.

If executed, the instruction shall generate a random value for *"randomAddress"*, in the range of [0x000000,0xFFFFFE] which shall be available within 100 ms for use.

If there are multiple logical units present and the instruction is executed using broadcast addressing, the generated random addresses within the bus unit shall be unique, i.e. every logical unit shall have a random address that is not found in any of the other logical units contained in the bus unit.

There shall be no reply to this instruction.

Refer to subclause 9.14.2 for further information.

11.7.6 COMPARE

The query shall be discarded unless *"initialisationState"* is ENABLED.

If executed, the control gear shall answer:

- if "*randomAddress*" ≤ "*searchAddress*": YES;
- in all other cases: NO

Refer to subclause 9.14.2 for further information.

11.7.7 WITHDRAW

The instruction shall be discarded unless the following conditions hold:

- *"initialisationState"* is equal to ENABLED, and
- *"randomAddress"* is equal to *"searchAddress"*

If the instruction is executed, the control gear shall change *"initialisationState"* to WITHDRAWN.

Before withdrawing a control gear, the application controller may assign it a short address, using "PROGRAM SHORT ADDRESS (*data*)".

NOTE The effect is that the control gear is excluded from subsequent "COMPARE" operations, thus allowing the application controller to conduct a (binary) search operation across all devices until the "COMPARE" query leads to no answer (from any control gear) on the bus.

Refer to subclause 9.14.2 for further information.

11.7.8 SEARCHADDRH (*data*)

The instruction shall be discarded if *"initialisationState"* is equal to DISABLED.

If executed, *"searchAddress[23:16]"* shall be set to the given *data*.

Refer to subclause 9.14.2 for further information.

11.7.9 SEARCHADDRM (*data*)

The instruction shall be discarded if *"initialisationState"* is equal to DISABLED.

If executed, *"searchAddress[15:8]"* shall be set to the given *data*.

Refer to subclause 9.14.2 for further information.

11.7.10 SEARCHADDRL (*data*)

The instruction shall be discarded if *"initialisationState"* is equal to DISABLED.

If executed, *"searchAddress[7:0]"* shall be set to the given *data*.

Refer to subclause 9.14.2 for further information.

11.7.11 PROGRAM SHORT ADDRESS (*data*)

The instruction shall be discarded unless the following conditions hold:

- *"initialisationState"* is equal to ENABLED or WITHDRAWN, and
- *"randomAddress"* is equal to *"searchAddress"*

If executed, *"shortAddress"* shall be set as follows:

- if *data* = MASK: MASK (effectively deleting the short address)
- if *data* = 1xxxxxxxb or xxxxxxx0b: no change
- in all other cases (0AAAAAA1b): 00AAAAAAb.

Refer to subclause 9.14.2 for further information.

11.7.12 VERIFY SHORT ADDRESS (*data*)

The query shall be discarded if *"initialisationState"* is equal to DISABLED.

If executed, the answer shall be YES if *"shortAddress"* is equal to 00AAAAAAb for *data* given by 0AAAAAA1b, and NO otherwise.

Refer to subclause 9.14.2 for further information.

11.7.13 QUERY SHORT ADDRESS

The query shall be discarded if:

- *"initialisationState"* is equal to DISABLED, or
- *"randomAddress"* is not equal to *"searchAddress"*.

If executed, the answer shall be 0AAAAAA1b, where "*shortAddress*" is equal to 00AAAAAAb', or MASK, in case "*shortAddress*" equals MASK.

Refer to subclause 9.14.2 for further information.

11.7.14 ENABLE DEVICE TYPE (*data*)

This instruction shall select the device type/feature for which the next application extended command (refer to 11.6) is valid. Executing “ENABLE DEVICE TYPE (data)” shall cancel any previous selection of a device type/feature. The selection is only valid for the next application extended command.

If a non-application extended command is accepted after executing “ENABLE DEVICE TYPE (data)”, the enabling of the device type/feature shall be cancelled and that command shall be executed according to its specification.

A control gear shall not react to commands which belong to the application extended commands if *data* equals MASK, 254 or represents a device type/feature not supported by this control gear.

The device types/features shall be coded as specified in the particular parts of the IEC 62386-2xx series.

Application controllers should be able to identify individual gears and store the relationship between the gear's individual address and the device types/features in a persistent memory.

11.7.15 DTR1 (*data*)

“DTR1” shall be set to given *data*.

Refer to subclause 9.10 for further information.

11.7.16 DTR2 (*data*)

“DTR2” shall be set to given *data*.

11.7.17 WRITE MEMORY LOCATION (*DTR1, DTR0, data*)

The instruction shall be discarded if any of the following conditions hold:

- the addressed memory bank is not implemented, or
- *“writeEnableState”* is DISABLED.

NOTE 1 This operation is a broadcast operation. Selective control gear addressing can be achieved by setting the write enable condition selectively.

If the instruction is executed, the control gear shall write *data* into the memory location identified by *“DTR0”* within memory bank *“DTR1”* and return *data* as an answer.

NOTE 2 Simultaneous writing to multiple control gear will probably lead to framing errors because of colliding answers.

NOTE 3 The value that can be read from the memory bank location is not necessarily *data*.

If the selected memory bank location is

- not implemented, or
- above the last accessible memory location, or
- locked (see subclause 9.10.2), or
- not writeable,

the answer to “WRITE MEMORY LOCATION (*DTR1, DTR0, data*)” shall be NO and no memory location shall be written to.

If the addressed location is below location 0xFF, the control gear shall increment *"DTR0"* by one.

NOTE 4 This allows efficient multi-byte writing within a transaction.

Refer to subclause 9.10 for further information.

11.7.18 WRITE MEMORY LOCATION – NO REPLY (*DTR1, DTR0, data*)

This instruction is identical to the "WRITE MEMORY LOCATION (*DTR1, DTR0, data*)" command except that the receiving control gear shall not reply to the command.

Refer to subclause 9.10 for further information.

11.7.19 PING

The ping command is used by single master application controllers (see IEC 62386-103) to indicate their presence. The ping command shall be ignored by control gear.

12 Test procedures

Void

Annex A
(informative)

Examples of algorithms

A.1 Random address allocation

The control gear are connected to a control device that uses random address allocation for setup of the system.

a) Start the algorithm with “INITIALISE (device)” which enables the addressing commands for a time period of 15 min.

b) Send “RANDOMISE”; all control gear choose a *randomAddress* so that $0 \leq randomAddress \leq +2^{24}-2$.

c) The control device searches the control gear with the lowest random address by means of an algorithm which uses “SEARCHADDRH (*data*)”, “SEARCHADDRM (*data*)”, “SEARCHADDRL (*data*)” and “COMPARE”. The control gear with the lowest random address is found. At this point, the control device needs to be able to handle different timing in backward frames coming from different control gear. Also, there is a chance that control gear generated the same *randomAddress* in which case randomisation should be restarted for the remaining gear.

d) The short address is programmed to the control gear found with aid of “PROGRAM SHORT ADDRESS (*data*)”.

e) “VERIFY SHORT ADDRESS (*data*)” can be used to verify the correct programming.

f) The found control gear can be identified by using “IDENTIFY DEVICE”, or use an alternating sequence of “RECALL MAX LEVEL” and “RECALL MIN LEVEL” with the programmed short address to record the local position of the respective control gear

g) If needed, the short address can be changed going back to step d)

h) The control gear found shall be removed from the search process by means of “WITHDRAW”.

i) Repeat from step c) on until no further control gear can be found. Use “INITIALISE (device)” to prolong the 15 min timer if needed.

j) Stop the process with “TERMINATE”.

In the event of two or more control gear having the same short address, restart the addressing procedure only for these control gear with “INITIALISE (*device*)” (using the short address in the second byte) followed by steps b) to j).

A.2 One single control gear connected to the control device

Only one control gear is connected to a control device that uses the following algorithm to program a short address.

a) Transmit the new short address (0AAA AAA1b) by “DTR0 (*data*)”.

b) Verify the content of the DTR0 by “QUERY CONTENT DTR0”.

c) Send “SET SHORT ADDRESS (*DTR0*)” twice in accordance with the requirements as stated in IEC 62386-101:2014 and IEC62386-101:2014/AMD1:2018, 9.3.

A.3 Using application extended commands

A control device using application extend commands needs to detect which application extended commands are supported by the different control gear. The following algorithm can be used:

a) Initialisation process and address allocation.

b) Query the device type/feature of every control gear in the system. If the received answer is 'MASK' the device supports more than one device type. In this case the following procedure can be used to get a list of the device types the control gear belongs to:

1) Send “QUERY EXTENDED VERSION NUMBER” command preceded by “ENABLE DEVICE TYPE (*data*)” with *data* equal to “0”. If there is an answer, the control gear belongs to device type 0.
2) Repeat step a) with all other device types supported by the control device.

c) The control device shall send “ENABLE DEVICE TYPE (*data*)” before every application extended command.

Annex B
(normative)

High resolution dimmer

A high resolution dimmer is not mandatory. However, if it is implemented, it shall be implemented according to this annex.

The *"actualLevel"* shall report the nearer level, after rounding of an internal value as can be seen in Figure B.1.

Light output shall always match a discrete point on the selected dimming curve, except when:

- a fade is running (via ideal, or high resolution internal curve);
- a fade is stopped before the fade time has elapsed;
- another dimming curve is selected;
- a level was programmed with another dimming curve (typically the scenes, including power on level and system failure level, store the level as well as the corresponding dimming curve, to allow representing in other curves and back without loss).

When light output is "in between" two discrete points on the selected dimming curve:

- *"actualLevel"* is set to the nearest of these two points;
- a new fade starts from the actual light output (which may not match a discrete point on the selected dimming curve) and ends at the target light output (which may not match a discrete point on the selected dimming curve, i.e. when a preset is used that was set using another dimming curve);
- the fade shall always follow the ideal or internal high resolution curve without making jumps to a discrete point on the selected dimming curve;
- there are some consequences for testing:
  - after stopping a fade, or switching dimming curve, a first step might be less than or more than a full step in the active dimming curve;
  - testing with levels from more than one dimming curve at a time is complex, perhaps better avoided

Behaviour that can be experienced:

- theoretically a minimal fade is possible while *"actualLevel"* is not changed;
- Direct Arc Power Command (DAPC) to the actual level might fade from the point on the high resolution dimmer to the discrete point (less than half a dim step);
- a STEP UP or STEP DOWN can worst case result in a "half step" delta or "one and a half step"; e.g. HighRes dimmer that ended at stepsize+0,49 stepsize: response to step up is 0,51 stepsize, while the response to step down is 1,49 stepsize to end at a discrete step;
- the "fade running" status is set to "1" at Fade time start and lasts until FadeTime has elapsed, even when "Actual Level" is already at the final level;
- actual level always represents the nearer discrete point on the dimming curve.

Special cases:

- When a fade is started from "Lamp off" condition, the first step from off to "Min Level" must be made at fade start time. The step from off to min level is not part of the fade time.

- When a fade is started to “Lamp off” condition, the last step from “Min Level” to off must be made at fade start time + FadeTime. The step from min level to off is not part of the fade time.

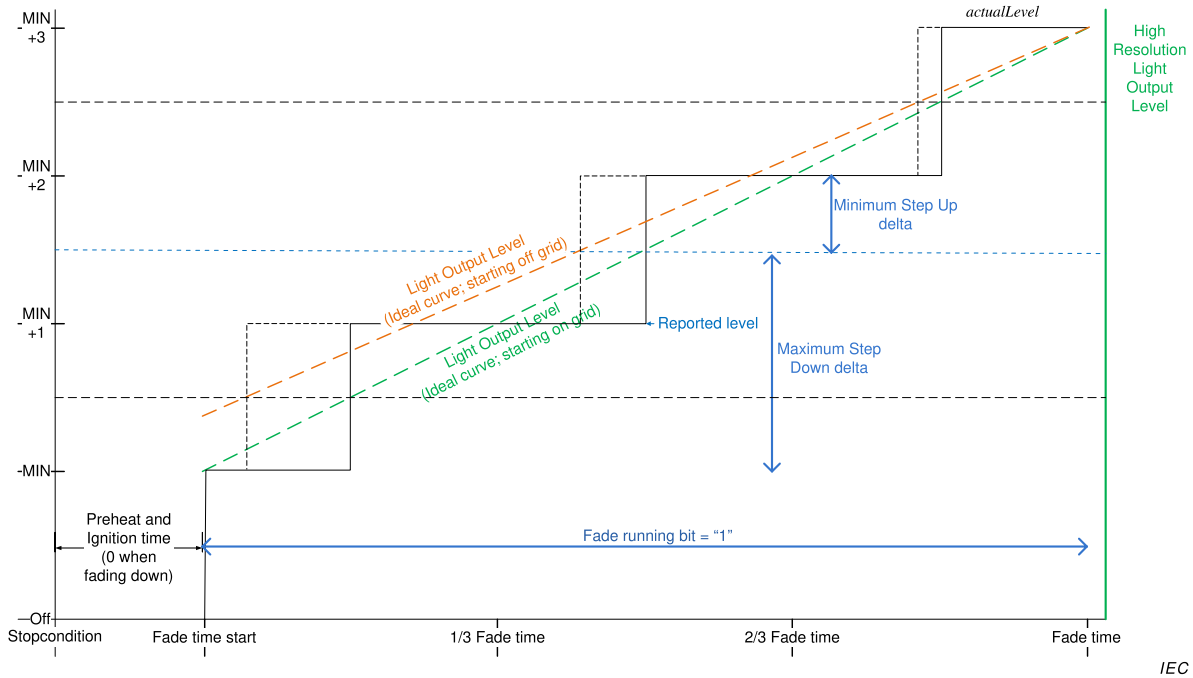

Figure B.1 – Level behaviour in cases of off-grid starting points

Bibliography

IEC 60598-1, *Luminaires – Part 1: General requirements and tests*

IEC 60669-2-1, *Switches for household and similar fixed electrical installations – Part 2-2, Particular requirements – Electronic switches*

IEC 60921, *Ballasts for tubular fluorescent lamps – Performance requirements*

IEC 60923, *Auxiliaries for lamps – Ballasts for discharge lamps (excluding tubular fluorescent lamps – Performance requirements*

IEC 60925, *D.C.-supplied electronic ballasts for tubular fluorescent lamps – performance requirements*

IEC 61347 (all parts), *Lamp controlgear*

IEC 61547, *Equipment for general lighting purposes – EMC immunity requirements*

IEC 62386-102:2009, *Digital addressable lighting interface – Part 102: General requirements – Control gear*

CISPR 15, *Limits and methods of measurement of radio disturbance characteristics of electrical lighting and similar equipment*

GS1 General Specification, Version 14: Jan-2014, [cited 2014-07-15] . Available at: http://www.google.ch/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=2&ved=0CCIQFjAB&url=http%3A%2F%2Fwww.gs1.at%2Findex.php%3Foption%3Dcom_phocadownload%26view%3Dcategory%26download%3D289%3Ags1-general-specifications-v14-en%26id%3D9%3Ags1-spezifikationen-a-richtlinien%26Itemid%3D304&ei=znm2U4PqFoP20gXXmIHgAQ&usg=AFQjCNHoqaUjWXvLbyJfVJoGxgOAl63mCw

_____