Domintell LightProtocol guide for communication interface of new generation (DNET01, DNET02, DGQG04, DGQG02, ...).

The goal of this document is to describe the LightProtocol of Domintell's Ethernet interfaces of new generation (DNET01, DNET02, DGQG02, DGQG04, ...).

For informations about DRS3202, DETH02 or any older LightProtocol interfaces, please consider reading DS_RS232_ETH_Interfaces_v1_27_08.pdf

Table of Contents

1. Document revisions

v16 : 19/03/2025	
	- Add DMONOELEC01 ("MON") and DTRIELEC01 ("TRI"). Note : DTRIELEC01 may not be handled in GoldenGate or DGQG02/04 ! Please check changelogs of related OSes.
	- Add DMR02 ("MR2").
	- The APPINFO and statuses of TypeElecIo(24) have been finalized and are now **official**.
	- The APPINFO and statuses of TypeFanIo(13) and TypeSwingIo(54) have been finalized and are now **official**.
	- Drop DGQG03 ("QG3").
	- Add information for the ModBus generic device ("MBG") (4.5 I/O mapping of all modules).
	- Since PROG M 39.1, DIN10V02 ("I10") has extra information in legacy APPINFO and since PROG M version 43.4, extra information have been updated. See 4.7.d Decoding APPINFO.
	- Add some information about restrictions when command PING, %S and /S are used (4.6.d Legacy action/command parameters and 5.9 Refresh statuses (PING)).
	- Fix status example for PBLCD01/02 ("PBL") and DPBRLCD02 ("PRL"). Status "B" does not exist for input. It is "I" (4.7.c Sample of received strings from your Domintell installation).
	- Fix APPINFO details for DMV01 ("DMV") module. All IOs had the offset 1. (4.7.d Decoding APPINFO).
	- Fix example for the TypeDmxIo(25) in section 4.8.k IO type list, status and data format. Channel number was missing.
	- Review layout and add examples for NewGen APPINFO (4.8.e Extra informations in APPINFO) and statuses/commands (4.8.k IO type list, status and data format)
	- Add new IO type for DGQG02 ("QG2"), DGQG04 ("QG4") and DGQG05 ("QG5"): TypeCloudInfo(62), TypeMemoryInfo(64), TypeStorageInfo(65), TypeCpuInfo(66), TypeDiBusGwInfo(67). See section 4.8.k IO type list, status and data format.
	- Since PROG M 43.7, users may have limited access to LightProtocol commands (5.4 Permissions/Roles).
	- Add 4.1 Reserved keywords and 4.2 System messages chapter.
	- Since PROG M 43.7, the user may issue the DISCOVER command to get more information about the socket. See 4.1 Reserved keywords.
	- Add DPBRT0x (DPBRT02/DPBRT04/DPBRT06) ("BRT").
	- Since PROG M 43.7, the command /104|<type_push> for TypeInputIo(2) and TypeMovIo(34) has been added. See section 4.8.k IO type list, status and data format.
v15 : 30/11/2023	
	- Since PROG M version 42.3.2, the reference IO for **MEM [SHUTTERS]** has been fixed. The offset of the IO is now 1, 3, 5 or 7 as expected. See 4.7.d Decoding APPINFO.
	- Fix : since PROG M version 42.3.2, unsupported regulation modes of temperature sensors are now set in HMR porperty. From version 42.0 to 42.3.1 of PROG M, bits 0x10 (Auto HVAC), 0x20 (Dry mode) and 0x40 (Fan mode) were not set in HMR property for classic temperature sensor ("TE1", "LT5", ...) while they do not handle these modes !

v14 : 21/11/2023	
	- Add information about DELEC01 ("EL1") and TypeElecIo(24) (**DRAFT !**)
	- Add handling of system sunrise and sunset clock (Data type K) by adding [READONLY] tag at the end of line in APPINFO and offset information : "[SUNRISE][READONLY]" and "[SUNSET][READONLY]" (4.7.b Data Types). **Backward compatibility broken !**
	- TypeDmvIo(13) is now called TypeFanIo(13)
	- Add explanation on how to decode groups ("MEM") in 4.7.d Decoding APPINFO. An **important note/workaround** has been added to parse reference IO of a shutter group (**MEM [SHUTTERS]**) for PROG M between 31 and 43.0.0 (inclusive). See 4.7.d Decoding APPINFO.
	- Add MEM and SFE in 4.3 Abreviation of Modules' type
	- Add Doorstation ("DST") and add information about TypeTorBasicTempoIo(52) and TypeInputTriggerIo(53) (4.8.k IO type list, status and data format).
	- Add more information about the configuration of temperature sensors (HCH, HCL, LCH, LCL and ISP tags) in APPINFO that depends on the PROG M value in 4.7.d Decoding APPINFO.
v13 : 17/01/2023	
	- Fix information about the status (color cycle and malformed frame) for TypeRgbwIo(46) (4.8.k IO type list, status and data format) of DRGBW01 ("RW1"). Note that color cyle for DRGBW01 ("RW1")or DMX01/02 ("DMX"/"DX2") may not work as expected for now. This will be fixed in a next version of LightProtocol (PROG M).
	- The mask in the command 47 for TypeRgbwIo(46) is not correctly processed by LightProtocolServer (DGQG02/DGQG04). An update to version 41.7.1 (PROG M 41.7 Rev=1) or newer of application is required (available with GoldenGate 19.12.2 or newer).
	- NewGen Commands 1, 2 and 3 on TypeLedIo (10) do not work for version below 41.7.1 (PROG M 41.7 Rev=1 or higher required) (GoldenGate 19.12.2 or newer is required).
	- Add information for the DIN10V02 ("I10") configured in temperature sensor mode (4.5 I/O mapping of all modules)
v12 : 31/12/2022	
	- Add handling of video output and gesture input of DTSC05 ("LT5")
	- Improve details of commands %F and %A for DAMPLI01 (4.5 I/O mapping of all modules)
	- Add commands %I%F, %FN, %O%F, %FP, %I%A, %AN, %O%A, %AP for DAMPLI01 (4.6.d Legacy action/command parameters)
	- Fix frame format for DISM20 ("I20"). Number of chars differ depending of the offset (1 char for inputs from 1 to 15 and 2 chars for input from 16 to 20) (4.5 I/O mapping of all modules)
	- Add information about status, command and APPINFO for TypeRgbwIo(46) of DRGBW01 ("RW1")
	- Add DGQG05 ("QG5")
	- Add information about date/time format (4.7.c Sample of received strings from your Domintell installation)
v11 : 05/07/2022	

	- Add DALI04 ("PS4") and DALI05 ("PS5")
v10 : 20/05/2022	
	- Add DDIMLV01/DLBxxx ("LV1")
	- Add air conditioner device ("ACD") connected to DINTMB02 module.
	- Add new hide mode for TypeSensorIo(8) in 4.8.e Extra informations in APPINFO (used by aircomodule)
	- Add TypeDmvIo(13) and TypeSwingIo(54) (**DRAFT !**) in NewGen LightProtocol in 4.8.e Extra informations in APPINFO and 4.8.k IO type list, status and data format
	- Add status command for NewGen LightProtocol (103/0x67)
	- Mark status command for group ("MEM") as working with Legacy LightProtocol
	- Review typo in 4.5 I/O mapping of all modules
v9 : 28/03/2022	
	- Format of temperature sensor in NewGen APPINFO (TypeSensorIo(8)) is not correctly documented fields are documented to be seperated with "/" while they are seperated with "|" (4.8.e Extra informations in APPINFO).
	- Since version 40.0 of DAP file, <regul_mask> and <temperature_mask> of temperature sensor in NewGen APPINFO (TypeSensorIo(8)) are still epxressed in hexadecimal prefixed with "0x". Before, they were expressed in hexadecimal but without "0x" ! (4.8.e Extra informations in APPINFO)
	- Fix incorrectly documented offsets of LED indicator and temperature sensor for command of module DPBL0x, DPBC0x, DPBR0x, DPBT0x, DPBU0x and DTSC0x (4.5 I/O mapping of all modules)
	- Add DPBRTHERM01 ("RT1")
v8 : 02/03/2022	
	- Add format of statuses for TypeWindIo(41)
v7 : 10/12/2021	
	- Review formatting of chapters "I/O mapping of all modules" and "Legacy action/command parameters"
v6 : 07/03/2021	
	- Add limit of setpoint for temperature sensor in APPINFO. (Avalaible on DGQG02/04 from O.S. 22.5.0 and DNET02 from O.S. 11.2.0, which is PROG M 37.0) [LOCAL][HMR=0x%x-HMT=0x%X][LHH=%f-LHL=%f-LCH=%f-LCL=%f-ISP=%f]
v5 : 27/10/2020	
	- Fix typo in description of "TRP 151-4"
v4 : 21/09/2020	
	- Fix typo for DGQG02 commands with data (command and data must be seperated with '|')
	- Add format of statuses for TypeSensorIo, TypeMovIo, TypeLuxIo, TypeHumidityIo, TypePressureIo, TypeCo2Io

	- Add example for NewGen statuses
v3 : 26/08/2020	
	- Add DTSC05 ("LT5"), DDMX02 ("DX2")
	- Add example for NewGen commands
	- Add minimal DAP version required to handle modules
	- Adjust header of APPINFO block
v2 : 24/10/2018	
	- Initial release

2. Informations about communication interfaces

2.1. General information

The goal of this document is to describe Ethernet interfaces of new generation for Domintell2 system and to help you to make the good choice between the options available. The hardware does not change but the functions depend on the firmware.
There is several ways to communicate with Domintell system depending of the module (some of these modules are deprecated. There are just named for information):

- Custom input ASCII strings (sent to Domintell system). need creation of links in configuration software (See chapter "Parameters and specific links->DRS23201 module" in Domintell2 Configuration software manual) is working with DRS23201, DRS23202, DETH02, DUSB01 and DGSM01.
- Custom output ASCII strings (sent to your device). need creation of links in configuration software (See chapter "Parameters and specific links->DRS23201 module" in Domintell2 Configuration software manual) is working with DRS23201, DUSB01 and DGSM01.
- Legacy output LightProtocol (LightProtocol server to LightProtocol client) is only available on DNET01, DNET02, DGQG02, DGQG04, DRS23202 and DETH02. No configuration/link is required in Domintell2 configuration software; it is automatically generated by master module.
- Legacy Input LightProtocol (LightProtocol client to LightProtocol server) is working with DNET01, DNET02, DGQG02, DGQG04, DRS23201, DRS23202, DETH02, DUSB01 and DGSM01. No configuration/link is required in Domintell2 configuration software; it is automatically decoded by master module.

This document only covers Ethernet modules of new generation (like DNET02, DGQG04, DGQG02, …) using Secured Websockets instead of RS232 or UDP socket.

2.2. Devices overview

Here is the list of communication modules and their capabilities :

- DNET01/DNET02 :
  - IP : DHCP or static. It is highly recommended to set a static IP.
  - Default port 17481.
  - Possibility to set a password.
  - Limited to max 1 legacy UDP (almost backward compatible for application written for DETH02)
  - Limited to max 8 simultaneous connections using Secured WebSocket protocol.
  - Legacy Input LightProtocol (LightProtocol client to LightProtocol server);
  - Custom input ASCII strings (sent to Domintell system).;
  - Legacy output LightProtocol (LightProtocol server to LightProtocol client).

- DGQG02/DGQG04 :
  - Limited to max 2 simultaneous connections using Secured WebSocket protocol.

- Legacy Input LightProtocol (LightProtocol client to LightProtocol server);
- Custom input ASCII strings (sent to Domintell system).;
- Legacy output LightProtocol (LightProtocol server to LightProtocol client).

2.3. Ethernet wiring information

The RJ45 connector must be connected to the LAN (Local Area Network) with a classic patch UTP RJ45 Cable (Cat 5 or higher) to a switch or a router.

WARNING :
Do NOT connect Domintell bus on the RJ45 connector, this can cause fatal damages to the module.

3. Terminology

- Master : Refers to DGQG0x modules.
- Central Unit : Refers to DGQG0x modules.
- Third-party device : device that needs to control Domintell2 installation. This can be a smartphone, a computer, a Raspberry Pi, ...
- LightProtocol server : refer to module on Domintell system that generates APPINFO and statuses. DGQG0x, DNET0x are LightProtocol servers.
- LightProtocol client : refer to third-party application/device that will interact with Domintell system by parsing APPINFO sent by LightProtocol server and sending command to LightProtocol server to control outputs. Smartphone apps like Domintell Pilot is a LightProtocol client.
- Input frame : frame sent from LightProtocol client to LightProtocol server.

4. Protocol specifications

4.1. Reserved keywords

Some keywords are used by the LightProtocol server and can not be used as custom string.

- Commands when the session is closed
  These commands are available before any successful `LOGINPSW@:` command.

TEXT	Dir.	LP version	Min. role	Description
`GETLPVER`	LP ← APP	>= 43.7	All	Request the version of the LightProtocol. See 5.6 Ask the version of the LightProtocol (GETLPVER).
`DISCOVER`	LP ← APP	>= 43.7	All	
`INFO:.:INFO`	LP → APP	All	-	Information message.
`ERROR: :ERROR`	LP → APP	All	-	Error message.
`REQUESTSALT`	LP ← APP	All	All	Request the salt and the nonce to hash the password to be used with LOGINPSW. See 5.5 Open a session.
`LOGINPSW`	LP ← APP	All	All	Open a session with the username and the hashed salted password. See 5.5 Open a session.
`HELP`	LP ← APP	None	All	Not yet available/TDB
`HELP@`	LP ← APP	None	All	Not yet available/TDB

- Commands when the session is opened
  These commands are available after a successful `LOGINPSW@:` command.

TEXT	Dir.	LP version	Min. role	Description
`GETLPVER`	LP ← APP	>= 43.7	All	Request the version of the LightProtocol. See 5.6 Ask the version of the LightProtocol (GETLPVER).
`DISCOVER`	LP ← APP	>= 43.7	All	Request information of the device that hosts the LightProtocol server.
`INFO:.:INFO`	LP → APP	All	-	Information message. See 4.2 System messages.
`ERROR: :ERROR`	LP → APP	All	-	Error message. See 4.2 System messages.
`LOGOUT`	LP ← APP	All	All	Close the session.

TEXT	Dir.	LP version	Min. role	Description
				See 5.10 Close session before exiting the application.
`TIMEOUT`	LP ← APP	All	All	Allow to alter the timeout of the session. See 5.8 Keep session open (HELLO).
`HELLO`	LP ← APP	All	All	Heartbeat to send before the session expired and keep it opened. See 5.8 Keep session open (HELLO).
`APPINFO`	LP ⇌ APP	All	Viewer	Request the current application (list of IO). See 5.7 Download list of modules (APPINFO).
`END APPINFO`	LP → APP	All	-	Notify LightProtocol client that it has received all application. See 5.7 Download list of modules (APPINFO).
`PING`	LP ← APP	All	Viewer	Request all statuts. See 5.9 Refresh statuses (PING).
`PONG`	LP → APP	All	-	Acknowledge to PING command. Statuses of all IO will follow. See 5.9 Refresh statuses (PING).
`VOICEINFO`	LP ← APP	>= 40.0	Viewer	Command reserved to voice assistant server to ask configuration (only on DNET02)
`HELP`	LP ← APP	None	All	Not yet available/TDB
`HELP@`	LP ← APP	None	All	Not yet available/TDB

4.2. System messages

The LightProtocol server (DGQG0x, DNET0x) can send system message between `INFO:...:INFO` or `ERROR:...:ERROR` tags.

- `INFO:Waiting for LOGINPSW:NONCE=123456789:INFO`
  Welcome message when connecting to the LightProtocol server.

- `INFO:REQUESTSALT:USERNAME=<username>:NONCE=123456789:SALT=123456789:INFO`
  Reply to the `REQUESTSALT@<username>` command.

- `INFO:Session opened:INFO`
  Reply to `LOGINPSW@<username>:<hashedpsw>` command on sucessful login.

- `ERROR:Invalid LOGINPSW. Use REQUESTSALT@<username> and LOGINPSW@<username>:<hashedpsw>:ERROR`
  Reply to the `LOGINPSW@<username>:<hashedpsw>` command when credentials are invalid or any unauthorized command : bad username, bas password, invalid implementation of the hashing function, ...

- `INFO:Session closed by server:INFO`
  The server is going to the connexion by itself (no `LOGOUT` command sent by the user). This can occur when a new DAP file is received by the LightProtocol server.

- `INFO:Session timeout:INFO`
  The server is closing the session because no action nor `HELLO` command has been received to keep the session opened.

- `INFO:World:INFO`
  Reply to `HELLO` command to keep session active/opened.

- `INFO:Timeout disabled. Socket will never be closed unless you send LOGOUT or the connecion is lost !:INFO`
  Reply to `TIMEOUT=0` command to disable the expiration of the session.

- `INFO:Timeout set to <n>min:INFO`
  Reply to `Timout=<n>`.

- `ERROR:Deprecated. Use REQUESTSALT@<username> and LOGINPSW@<username>:<hashedpsw>:ERROR`
  Reply to deprecated `LOGIN` command.

- `INFO:I AM A DGQG04-192.168.1.250-169.254.162.138-17481-54000001-WSS:INFO`
  Reply to `DISCOVER` command. Since `PROG M 43.7`.

4.3. Abreviation of Modules' type

Reference	ID	Min. DAP version	Description
DALI04 (NewGen)	PS4	>= 40.1	Domintell bus power supply
DALI05 (NewGen)	PS5	>= 40.1	Domintell bus
DAMPLI01	AMP	>= 30	Sound Module
DBIR01	BIR	>= 30	8 bipolar relays
DDIM01	DIM	>= 30	8 dimmer commands
DDIR01	DIR	>= 30	IR detector

Reference	ID	Min. DAP version	Description
DMV01	DMV	>= 30	Mechanical ventilation
DDMX01	DMX	>= 30	DMX Module
DDMX02 (NewGen)	DX2	>= 33	DMX Module with RGBW handling
DENV01 (NewGen)	EV1	>= 33	Environment sensor module
DENV02 (NewGen)	EV2	>= 33	Environment sensor module with CO2 sensor
DELEC01 (NewGen)	EL1	>= 42.9 < 43.0 >= 43.4	P1 smart metering interface
DFAN01	FAN	>= 30	Fan controler
DGQG02 (NewGen)	QG2	>= 30	New generation Master module with embedded IOs
DGQG04 (NewGen)	QG4	>= 30	New generation Master that replaces DGQG01
DGQG05 (NewGen)	QG5	>= 42	New generation Master with 4 relays, 16 inputs and an interface for the Domintell LightBus
DIN10V02	I10	>= 30	Analog 0-10V input module
DINTDALI01	DAL	>= 30	DALI interface
DINTMB01	MB1	n/a	Deprecated ModBus interface
DINTMB02 (NewGen)	MB2	>= 41	ModBus interface (not directly used in LightProtocol)
DISM04	IS4	>= 30	4 Inputs module
DISM08	IS8	>= 30	8 Inputs module
DISM20	I20	>= 30	20 inputes module
DLED01	LED	>= 30	4 leds driver
DMONOELEC01 (NewGen)	MON	>= 42.9 < 43.0 >= 43.4	Single-phase electrical energy meter
DMOV01 DMOV02 DMOV05	DET	>= 30	Infrared detector
DMOV06 (NewGen)	MV6	>= 33	Motion and light detector
DMOV07 (NewGen)	MV7	>= 39.1	Motion and light detector
DMR01	DMR	>= 30	5 Monopolar relays

Reference	ID	Min. DAP version	Description
DMR02 (NewGen)	MR2	>= 42.5 < 43.0 >= 43.3	8 monopolar relays
DOORSTATION (NewGen)	DST	>= 43	Doorstation module
DOUT10V02	D10	>= 30	0/1-10V dimmer module
DPBC01	CL1	>= 30	Classic 1 Push Button (8 colors and temperature sensor)
DPBC02	CL2	>= 30	Classic 2 Push Button (8 colors and temperature sensor)
DPBC04	CL4	>= 30	Classic 4 Push Button (8 colors and temperature sensor)
DPBC06	CL6	>= 30	Classic 6 Push Button (8 colors and temperature sensor)
DPBL01	B81	>= 30	Lithoss 1 Push Button (8 colors)
DPBL02	B82	>= 30	Lithoss 2 Push Button (8 colors)
DPBL04	B84	>= 30	Lithoss 4 Push Button (8 colors)
DPBL06	B86	>= 30	Lithoss 6 Push Button (8 colors)
DPBR02	BR2	>= 30	Glass rainbow 2 Push Button (RGB)
DPBR04	BR4	>= 30	Glass rainbow 4 Push Button (RGB)
DPBR06	BR6	>= 30	Glass rainbow 6 Push Button (RGB)
DPB(U/T)01	BU1	>= 30	1 Push Button
DPB(U/T)02	BU2	>= 30	2 Push Button
DPB(U/T)04	BU4	>= 30	4 Push Button
DPB(U/T)06	BU6	>= 30	6 Push Button
DPBRLCD02	PRL	>= 30	Glass rainbow LCD push buttons
DPBRT0x DPBRT02 DPBRT04 DPBRT06	BRT	>=43.7	Rainbow 2, 4 or 6 push button (RGB and temperature sensor)
DPBRTHERM01 (NewGen)	RT1	>= 40	Rainbow thermostat
DPBTLCD0x	PBL	>= 30	LCD push buttons
DRGBW01 (NewGen)	RW1	>= 38.3	RGBW LED strips controller
DTEM01	TE1	>= 30	Temperature sensor
DTRIELEC01 (NewGen)	TRI	>= 42.9 < 43.0	3-phase electrical energy meter

Reference	ID	Min. DAP version	Description
		>= 43.4	
DTRP01	TRP	>= 30	4 teleruptors
DTRP02	TPV	>= 30	2 shutter command with teleruptors
DTRV01	TRV	>= 30	4 shutter inverters
DTRVBT01	V24	>= 30	1 DC shutter command
DTSC02	LT2	>= 30	TFT Touchscreen
DTSC04	LT4	>= 30	TFT Touchscreen with video
DTSC05 (NewGen)	LT5	>= 33	Rainbow capacitive Touchscreen
DVIP01	VI1	30 only	1 button videophone
DVIP02	VI2	30 only	2 buttons videophone
DWIND01 (NewGen)	WI1	>= 33	Wind sensor module
Air conditioner device (NewGen)	MBA *	>= 42.5 < 43.0 >= 43.3	Air conditioner device (ModBus device) * ***This may subject to change in the future***
Daikin RTD-NET ModBus device	MBD	30 only	For Daikin RTD-NET (deprecated)
ModBus generic device (NewGen)	MBG	>= 42.5 < 43.0 >= 43.3	Generic ModBus device
Camera	CAM	>= 30	Cameras informations
Clock	CLK	>= 30 < 31.0 >= 43.0	Programmable software clocks (normal, reset and astronomical)
Radio station	STA	>= 30	Radio Station name & frequency
Variable	VAR	>= 30	Virtual programmed status
System variable	SYS	>= 30	See 4.4 Information about system variables
Temperature plage list	TPL	>= 30	Specific range of a Temperature profile
Temperature profile	TPR	>= 30	Profile's name which contains next temperature plage lists received
Group	MEM	>= 30	Groups. See 4.7.d Decoding APPINFO
Scene	SFE	>= 30	Scene

Min. DAP version refers to "`PROG M 33.0`" from header of APPINFO. See chapter 4.7.d Decoding APPINFO for more informations.

4.4. Information about system variables

Code	Name	Data type	Value
SYS000000	Simulation mode	Bool '0'	0 = Simulation is not playing (only record) 1 = Simulation is playing
SYS000009	Daytime	Bool '0'	Based on astronomical clock 0 = Night 1 = Daytime

4.5. I/O mapping of all modules

"`(packed)`" means that there is one status for all inputs or outputs and statuses of several outputs/inputs are packed in the same byte. The value must be handled using bitwise operators. For example, "`BIR000043O25`" must be handled as *0x20 | 0x04 | 0x01 = (1 << 5) | (1 << 2) | (1 << 0)*. So output 6, 3 and 1 are on.
"`(global)`" means that status of all inputs/outputs is sent on one line (values are concatenated). For example, status of a DDIM01 will look like "`DIM000023D640032000000000A`" which means that first output is at 100%, third output is at 50% and eighth output is at 10%.
Associated status type are explained in chapter 4.7.b Data Types.
Associated commands are explained in chapter 4.6.d Legacy action/command parameters.

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
DALI04 (NewGen)	`PS4`	/51/1	Bus power supply	Please see 4.8.k IO type list, status and data format	
DALI05 (NewGen)	`PS5`	/51/1	Bus power supply	Please see 4.8.k IO type list, status and data format	

DAMPLI01	`AMP`	1	1st audio output	`-1S`	`-1 (none) (from v41.3)`
					`-1%Fnnn.nn`
					`-1%An`
					`-1%I%A (from v41.7)`
					`-1%O%A (from v41.7)`
					`-1%I`
					`-1%O`
					`-1%Dnnn`
					`-1%I%D (see 4.6.d !)`
					`-1%O%D (see 4.6.d !)`
					`-1%I%F or -1%FN (from v41.7)`
					`-1%O%F or -1%FP (from v41.7)`

		...			...
		4	4th audio output	`-4S`	`-4 (none) (from v41.3)` `-4%Fnnn.nn`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
					-4%An -4%I%A (from v41.7) -4%O%A (from v41.7) -4%I -4%O -4%Dnnn -4%I%D (see 4.6.d !) -4%O%D (see 4.6.d !) -4%I%F or -4%FN (from v41.7) -4%O%F or -4%FN (from v41.7)
DBIR01	BIR	1	1st relay output	Oxx (packed) (global)	-1 (none) -1%I -1%O
		...			...
		8	8th relay output		-8 (none) -8%I -8%O
DDIM01	DIM	1	1st dimmer output	Dxx (global)	-1 -1%I -1%O -1%Dnnn -1%DB -1%DE -1%I%Dnnn (see 4.6.d !) -1%O%Dnnn (see 4.6.d !)
		...			...
		8	8th dimmer output		-8 -8%I -8%O -8%Dnnn

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**

					`-8%DB`
					`-8%DE`
					`-8%I%Dnnn (see 4.6.d !)`
					`-8%O%Dnnn (see 4.6.d !)`

DDIMLV01 (NewGen)	`LV1`	/42/1	1st dimmer output	Please see 4.8.k IO type list, status and data format	
		/42/2	2nd dimmer output		
		/42/...	Dynamic number of slaves. Depends on config.		
DDIR01	`DIR`	1	Remote control IR receiver	`Cxx`	*None*

DMV01	`DMV`	1
		2
		3
			3-speed output
			Index 2 and 3 not given in APPINFO but used in command frame to set speed 2 or 3 using %I
				`Oxx (packed)`
				`(global)`
					`-1%I`
					`-2%I`
					`-3%I`
					`-1%O` (from v38.4)
					`-2%O` (from v38.4)
					`-3%O` (from v38.4)

		4	1st aux. relay output		`-4`
					`-4%I`
					`-4%O`

		5	2nd aux. relay output		`-5`
					`-5%I`
					`-5%O`

DDMX01	`DMX`	1	1st DMX slave	`-1Xxx`	`-1-c` `-1-c%I`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**

					`-1-c%O`
					`-1-c%Xnnn`
					(`c` is the canal number)

		...			...

		8	8th DMX slave	`-8Xxx`	`-8-c`
					`-8-c%I`
					`-8-c%O`
					`-8-c%Xnnn`
					(`c` is the canal number)

DDMX02 (NewGen)	`DX2`	/25/1	1st DMX slave	Please see 4.8.k IO type list, status and data format	
		/25/2	2nd DMX slave		
		...	Dynamic number of slaves. Depends on config.		
DENV01 (NewGen)	`EV1`	/8/1	Temperature sensor	Please see 4.8.k IO type list, status and data format	
		/36/1	Ambiant luminosity input		
		/37/1	Humidity input		
		/38/1	Atmospheric pressure input		
DENV02 (NewGen)	`EV2`	/8/1	Temperature sensor	Please see 4.8.k IO type list, status and data format	
		/37/1	Humidity input		
		/39/1	Air quality input		

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
DELEC01 (NewGen)	`EL1`	/24/1	Electricity metering input	Please see 4.8.k IO type list, status and data format	
DFAN01	`FAN`	1	Output of speed 1	`Oxx (packed)`	DFAN01 is a highly integrated module and it is not recommended to interact with its output using LightProtocol. For security reasons, valves always follow the setpoint regulation, so if you need to toggle the valves of the DFAN01, you must first change the setpoint on the associated sensor. If valves are OFF, fan will not start.
		2	Output of speed 2		
		3	Output of speed 3		
		4	Output of cooling valve		
		5	Output of heating valve		
		6	Working mode (virtual output) (deprecated)		0 = auto, 1 = manual.
DGQG02 (NewGen)	`QG2`	/1/1	1st relay	*Please see 4.8.k IO type list, status and data format*	
		...			
		/1/8	8th relay		
		/2/1	1st input		
		...			
		/2/12	12th input		

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
		/6/1	Shutter output		
		/23/1	1st 0-10V output		
		/23/2	2nd 0-10V output		
		/40/1	Access control input	*None*	*None*
		/62/1	Info about cloud	*Please see 4.8.k IO type list, status and data format*	
		/63/1	Info about Ethernet		
		/64/1/	Info about memory		
		/65/1	Info about storage		
		/66/1	Info about CPU		
		/67/1	Info about Domintell bus gateway		
DGQG04 (NewGen)	`QG4`	/62/1	Info about cloud	*Please see 4.8.k IO type list, status and data format*	
		/63/1	Info about Ethernet		
		/64/1/	Info about memory		
		/65/1	Info about storage		
		/66/1	Info about CPU		
		/67/1	Info about Domintell bus gateway		
DGQG05	`QG5`	/1/1	1st relay	*Please see 4.8.k IO type list, status and data*	

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
(NewGen) Note : a virtual DDIMLV01 is always associated to a DGQG05		...			
		/1/4	4th relay		
		/2/1	1st input		
		...			
		/2/16	16th input		
		/62/1	Info about cloud		
		/63/1	Info about Ethernet		
		/64/1/	Info about memory		
		/65/1	Info about storage		
		/66/1	Info about CPU		
		/67/1	Info about Domintell bus gateway		
DIN10V02	`I10`	1	Analog 0-10V input module	`Dxx` (for analog mode)	*None*

				`Tnn.n`
				`Unn.n`
				(for temperature sensor mode)
					`-1%Tnn.n`
					`-1%Unn.n`
					`-1%Mn`
					`-1%Rn`

DINTDALI01	`DAL`	01	1st DALI salve	`-01Dxx`	`-01`
					`-01%I`
					`-01%O`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
					`-01%Dnnn`

		0A	10th DALI slave	`-0ADxx`	`-0A`
					`-0A%I`
					`-0A%O`
					`-0A%Dnnn`

		...	Dynamic number of slaves. Depends on config. **IO index has 2 characters !**	...	

		40	64th DALI slave	`-40Dxx`	`-40`
					`-40%I`
					`-40%O`
					`-40%Dnnn`

DISM04	`IS4`	1	1st input	`Ixx (packed) (global)`	`-1%Pn`
		...			...
		4	4th input		`-4%Pn`
DISM08	`IS8`	1	1st input	`Ixx (packed) (global)`	`-1%Pn`
		...			...
		8	8th input		`-8%Pn`
DISM20	`I20`	1	1st input	`Ixxxxxx (packed) (global)`	`-1%Pn`
		2	2nd input		`-2%Pn`
		...			...

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
		F	15th input		`-F%Pn`
		From offset 16 (0x10), IO index has 2 characters !			
		10	16th input		`-10%Pn`
		..			...
		14	20th input		`-14%Pn`

DLED01	`LED`	1	1st ouput	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

		..			...

		4	4th output		`-4`
					`-4%I`
					`-4%O`

DMONOELEC01 (NewGen)	`MON`	/24/1	Electrical energy meter	*Please see 4.8.k IO type list, status and data format*	
DMOV01 DMOV02 DMOV05	`DET`	1	Motion detector	`Ixx (packed) (global)`	`-1%P1` `-1%P2`
DMOV06 (NewGen)	`MV6`	/34/1	Motion detector	Please see 4.8.k IO type list, status and data format Prior to `PROG M 43`, the motion detector was referenced as a `TypeInputIo`(2) ("`/2/1`")!	
		/36/1	Ambiant luminosity input		
DMOV07	`MV7`	/34/1	Motion detector	Please see 4.8.k IO type list, status and data	

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
(NewGen)		/36/1	Ambiant luminosity input	Prior to `PROG M 43`, the motion detector was referenced as a `TypeInputIo`(2) ("`/2/1`")!	

DMR01	`DMR`	1	1st relay output	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

		...			...

		5	5th relay output		`-5`
					`-5%I`
					`-5%O`

DMR02 (NewGen)	`MR2`	/1/1	1st relay output	*Please see 4.8.k IO type list, status and data format*	
		...			
		/1/8	8th relay output		

DOUT10V02	`D10`	1	1st 0-10V output	`Dxx (global)`	`-1`
					`-1%I`
					`-1%O`
					`-1%Dnnn`
					`-1%DB`
					`-1%DE`
					`-1%I%Dnnn (see 4.6.d !)`
					`-1%O%Dnnn (see 4.6.d !)`

DPBC01	`CL1`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`

		2	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**

		3	Temperature sensor	`Tnn.n`
				`Unn.n`
					`-1%Tnn.n`
					`-1%Unn.n`
					`-1%Mn`
					`-1%Rn`

DPBC02	`CL2`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		2	2nd push-button		`-2%Pn`

		3	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

		4	2nd LED indicator		`-2`
					`-2%I`
					`-2%O`

		5	Temperature sensor	`Tnn.n`
				`Unn.n`
					`-1%Tnn.n`
					`-1%Unn.n`
					`-1%Mn`
					`-1%Rn`

DPBC04	`CL4`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		...			...
		4	4th push-button		`-4%Pn`

		5	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

		...			...

		8	4th LED indicator		`-4`
					`-4%I`
					`-4%O`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**

		9	Temperature sensor	`Tnn.n`
				`Unn.n`
					`-1%Tnn.n`
					`-1%Unn.n`
					`-1%Mnn.n`
					`-1%Rnn.n`

DPBC06	`CL6`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		...			...
		6	6th push-button		`-6%Pn`

		7	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

		...			...

		C	6th LED indicator		`-6`
					`-6%I`
					`-6%O`

		D	Temperature sensor	`Tnn.n`
				`Unn.n`
					`-1%Tn.nn`
					`-1%Unn.n`
					`-1%Mn`
					`-1%Rn`

DPBL01	`B81`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`

		2	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

DPBL02	`B82`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		2	2nd push-button		`-2%Pn`
		3	1st LED indicator	`Oxx (packed)`	`-1`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
				`(global)`	`-1%I` `-1%O`

		4	2nd LED indicator		`-2`
					`-2%I`
					`-2%O`

DPBL04	`B84`	1	1st push-button	`Ixx (packed)` `(global)`	`-1%Pn`
		...			...
		4	4th push-button		`-4%Pn`

		5	1st LED indicator	`Oxx (packed)`
				`(global)`
					`-1`
					`-1%I`
					`-1%O`

		...			...

		8	4th LED indicator		`-4`
					`-4%I`
					`-4%O`

DPBL06	`B86`	1	1st push-button	`Ixx (packed)` `(global)`	`-1%Pn`
		...			...
		6	6th push-button		`-6%Pn`

		7	1st LED indicator	`Oxx (packed)`
				`(global)`
					`-1`
					`-1%I`
					`-1%O`

		...			...

		C	6th LED indicator		`-6`
					`-6%I`
					`-6%O`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
DPBR02	BR2	1	1st push-button	Ixx (packed) (global)	-1%Pn
		2	2nd push-button		-2%Pn
		3	1st LED indicator	Oxx (packed) (global)	-1 -1%I -1%O
		4	2nd LED indicator		-2 -2%I -2%O
DPBR04	BR4	1	1st push-button	Ixx (packed) (global)	-1%Pn
		...			...
		4	4th push-button		-4%Pn
		5	1st LED indicator	Oxx (packed) (global)	-1 -1%I -1%O
		...			...
		8	4th LED indicator		-4 -4%I -4%O
DPBR06	BR6	1	1st push-button	Ixx (packed) (global)	-1%Pn
		...			...
		6	6th push-button		-6%Pn
		7	1st LED indicator	Oxx (packed) (global)	-1 -1%I -1%O

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
		...			...

		C	6th LED indicator		`-6`
					`-6%I`
					`-6%O`

DPB(U/T)01	`BU1`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`

		2	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

DPB(U/T)02	`BU2`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		2	2nd push-button		`-2%Pn`

		3	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

		4	2nd LED indicator		`-2`
					`-2%I`
					`-2%O`

DPB(U/T)04	`BU4`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		...		...	...
		4	4th push-button		`-4%Pn`

		5	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

		...			...

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**

		8	4th LED indicator		`-4`
					`-4%I`
					`-4%O`

DPB(U/T)06	`BU6`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		...			...
		6	6th push-button		`-6%Pn`

		7	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

		...			...

		C	6th LED indicator		`-6`
					`-6%I`
					`-6%O`

DPBRLCD02	`PRL`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		... (inputs may be hidden/missing depending of configuration)			...
		6	6th push-button		`-6%Pn`

		7	Temperature sensor	`Tnn.n`
				`Unn.n`
					`-1%Tnn.n`
					`-1%Unn.n`
					`-1%Mn`
					`-1%Rn`

		8	1st LED indicator	`Oxx (packed) (global)`	`-1`
					`-1%I`
					`-1%O`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
		... (outputs may be hidden/missing depending of configuration)			...
		D	6th LED indicator		-6 -6%I -6%O
DPBRT0x DPBRT02 DPBRT04 DPBRT06 (NewGen)	BRT	/2/1	1st push-button	Please see 4.8.k IO type list, status and data format	
		/2/2	2nd push-button		
		/2/3	3rd push-button (only for DPBRT04/06)		
		/2/4	4th push-button (only for DPBRT04/06)		
		/2/5	5th push-button (only for DPBRT06)		
		/2/6	6th push-button (only for DPBRT06)		
		/60/1	1st RGB LED indicator	Please see 4.8.k IO type list, status and data format	
		/60/2	2nd RGB LED indicator		
		/60/3	3rd RGB LED		

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
			indicator (only for DPBRT04/06)		
		/60/4	4th RGB LED indicator (only for DPBRT04/06)		
		/60/5	5th RGB LED indicator (only for DPBRT06)		
		/60/6	6th RGB LED indicator (only for DPBRT06)		
		/8/1	Temperature sensor	Please see 4.8.k IO type list, status and data format	
DPBRTHERM01 (NewGen)	`RT1`	/8/1	Temperature sensor	Please see 4.8.k IO type list, status and data format	
DPBTLCD0x	`PBL`	1	1st push-button	`Ixx (packed)` `(global)`	`-1%Pn`
		... (inputs may be hidden/missing depending of configuration)			...
		6	6th push-button		`-6%Pn`
		7	Temperature sensor (hidden for DPBRLCD01)	`Tnn.n` `Unn.n`	`-1%Tnn.n` `-1%Unn.n` `-1%Mn` `-1%Rn`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**

		8	1st LED indicator	`Oxx (packed)` `(global)`	`-1`
					`-1%I`
					`-1%O`

		... (outputs may be hidden/missing depending of configuration)			...

		D	6th LED indicator		`-6`
					`-6%I`
					`-6%O`

DRGBW01 (RGBW mode) (NewGen)	`RW1`	/46/1	RGBW output	Please see TypeRgbwIo(46) in 4.8.k IO type list, status and data format	
DRGBW01 (4 channels dimmer) (NewGen)	`RW1`	/3/1	1st simple dimmer output	Please see TypeDimmerIo(3) in 4.8.k IO type list, status and data format	
		/3/2	2nd simple dimmer output		
		/3/3	3rd simple dimmer output		
		/3/4	4th simple dimmer output		

DTEM01	`TE1`	1	Temperature sensor	`Tnn.n`
				`Unn.n`
					`-1%Tnn.n`
					`-1%Unn.n`
					`-1%Mn`
					`-1%Rn`

DTRP01	`TRP`	1	1st relay output	`Oxx (packed)` `(global)`	`-1` `-1%I`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
					`-1%O`
		...			...

		4	4th relay output		`-4`
					`-4%I`
					`-4%O`

DTRP02	`TPV`	1	1st shutter up	`Oxx (packed)`
				`(global)`
					`-1%L`
					`-1%H`
					`-1%I`
					`-1%O => -2%O`

		2	1st shutter down (hidden in APPINFO)		`-2%H`
					`-2%L`
					`-2%I`
					`-2%O => -1%O`

		3	2nd shutter up		`-3%L`
					`-3%H`
					`-3%I`
					`-3%O => -3%O`

		4	2nd shutter down (hidden in APPINFO)		`-4%H`
					`-4%L`
					`-4%I`
					`-4%O => -3%O`

DTRV01	`TRV`	1	1st shutter up	`Oxx (packed)`
				`(global)`
					`-1%L`
					`-1%H`
					`-1%I`
					`-1%O => -2%O`

		2	1st shutter down (hidden in APPINFO)		`-2%H`
					`-2%L`
					`-2%I`
					`-2%O => -1%O`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
		...			...

		7	4th shutter up		`-7%L`
					`-7%H`
					`-7%I`
					`-7%O => -2%O`

		8	4th shutter down (hidden in APPINFO)		`-8%H`
					`-8%L`
					`-8%I`
					`-8%O => -1%O`

DTRVBT01	`V24`	1	Shutter up	`Oxx (packed) (global)`	`-1%L`
					`-1%H`
					`-1%I`
					`-1%O => -2%O`

		2	Shutter down (hidden in APPINFO)		`-2%H`
					`-2%L`
					`-2%I`
					`-2%O => -1%O`

DTSC02 DTSC04	`LT2` `LT4`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		...			...
		4	4th push-button		`-4%Pn`

		5	Temperature sensor	`T`
				`U`
					`-1%Tn.n`
					`-1%Un.n`
					`-1%Mn`
					`-1%Rn`

		6	Remote control IR receiver	`C`	*None*
		B	1st LED indicator	`Oxx (packed) (global)`	`-1` `-1%I`

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
					`-1%O`
		...			...

		E	4th LED indicator		`-4`
					`-4%I`
					`-4%O`

		15	Lock screen	*None*	`-15%I`
DTSC05 (NewGen)	`LT5`	/2/1 (from v39.1)	1st push-button	Please see 4.8.k IO type list, status and data format	
		...			
		/2/4 (from v39.1)	4th push-button		
		/10/1	1st LED indicator		
		...			
		/10/4	4th LED indicator		
		/8/1	Temperature sensor		
		/31/1 (from v41.3)	Video output		
		/37/1	Humidity input		
		/49/1	Gesture input		

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
		(from v41.3)			
DVIP01	`VI1`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
DVIP02	`VI2`	1	1st push-button	`Ixx (packed) (global)`	`-1%Pn`
		2	2nd push-button		`-2%Pn`
DWIND01 (NewGen)	`WI1`	/41/1	Wind sensor	Please see 4.8.k IO type list, status and data format	
Air conditioner device (NewGen)	`MBA` *	*/8/1*	Temperature sensor	*Please see 4.8.k IO type list, status and data format* *** This may subject to change in the future**	
		/13/1	Fan output		
		/54/1	Vanes output		
		/55/1	Device status		
Daikin RTD-NET ModBus device	`MBD`	*None*	Deprecated	*None*	*None*
ModBus generic device (NewGen)	`MBG`	*/2/n*	Relay ('*n*' will be any number)	*Please see 4.8.k IO type list, status and data format*	
		/8/n	Temperature sensor ('*n*' will be any number)		
		/13/n	Fan output ('*n*' will be any number)		
		/24/n	Electrical energy		

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
			meter ('*n*' will be any number)		
		/54/n	Vanes output ('*n*' will be any number)		
		/55/1	Device status		
		/56/n	Percent input ('*n*' will be any number)		
		/57/n	Analog input ('*n*' will be any number)		
Cameras	`CAM`	*None*	Cameras informations	*None*	*None*
Clocks	`CLK`	*None*	Basic or astronomical clock	`K`	`%Knnn`
Doorstation (NewGen)	`DST`	*/2/1*	Virtual unlock door push-button	*Please see 4.8.k IO type list, status and data format*	
		/34/1	Motion Detector		
		/52/1	Relay		
		/52/2	Relay (if any)		
		/53/1	First calling push-button		
		/53/2	Second calling push-button (if any)		

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
Radio Station	`STA`	*None*	Radio Station informations	*None*	*None*

Variables	`VAR`	*None*	Control variable	`0xx` (boolean variable)
				`Dxx` (value variable)
					`(none)`
					`%I`
					`%O`
					`%Dnnn` (value variable only)
					`%S` (only from DAP version 40.1)

System variables	`SYS`	*None*	See 4.4 Information about system variables	`0xx`	`(none)`
					`%I`
					`%O`
					`%S` (only from DAP version 40.1)

Temperature Plage List	`TPL`	*None*	Time slot of a temperature profile	*None*	*None*
Temperature Profile	`TPR`	*None*	Name of temperature profile	*None*	*None*
Group	`MEM`	*None*	Control group. Use same status and command than the reference IO. For mixed group, only statuses and command of relay output are used/available. Please see 4.7.d Decoding APPINFO for more informations.		

				Depends on the type of the group	`(none)`
					`%I`
					`%O`
					`%S` (only from DAP version 40.1)

Module	**Mod Type**	**APPINFO IO index (1-based)** (in hexa for legacy LP)	**Description**	**Associated status type**	**Associated commands**
					Other commands depend on the type of the group

Scene	SFE	*None*	Control scene	*None*	`(none)`
					`%I`
					`%S` (only from DAP version 40.1)

4.6. Legacy Input LightProtocol (*LightProtocol client* to *LightProtocol server*)

4.6.a) Overview

These commands/strings can be sent to Domintell2 system and are executed without doing any links (Automatic LightProtocol).
Please read 4.8.b Modules using new generation input LightProtocol and 4.5 I/O mapping of all modules to know which frame format must be used for a given module.

4.6.b) General recommandations/limitations

- Frames **can not** be concatenated anymore using '&'.
- Important : we advise you to make less than 100 «string» links on the same intput because it's a lot of work for the Master (DGQG0x). A WARNING will be displayed into the Diagnose function if there's more than 100 «string» links.
- LightProtocol strings have priority on ASCII (custom) string. If a link is done in Domintell2 configuration software using text "BIR000B4B-1", master unit will decode it as LightProtocol string and will not execute your link.
- Strings '<CR>', '<LF>' and '<TAB>' are replaced by the equivalent ASCII code : 0x0D, 0x0A and 0x09.
- Carriage return & line feed characters are supported at the end of the command line.
- Domintell Automatically suppress (trim) the SPACE characters at the begin or at the end of the message.
- Strings are NOT case sensitive. Lower case characters are automatically replaced with upper case equivalent. (Be careful with accentuated characters like "éèêàñäí"...)
- We advise to use only ASCII characters. Accentuated character can be coded over multiple bytes under UTF-8 systems.

4.6.c) Legacy input frame format

Mod Type (3 char)	Serial Number (6 char hexadecimal)	- (1 char)	Output Number (1 or 2 char hexadecimal)	Action parameters (%...)

Output number is 2-char long (hexadecimal base) for output of DINTDALI01 module.

4.6.d) Legacy action/command parameters

A parameter always start with the character '%' (reserved char)

Command	Description	Min. role	Arguments
%S	- ask status to a module. Index of IO must be omitted. - ask status of a group. Status of first output is returned. - ask status of a scene. Statuses of all outputs contained in the scene will be returned. **For installations with NewGen Masters (DGQG02/DGQG04/...), this command is only available from O.S. version 25.3.0 ! (DAP version 40.1 or higher) for legacy modules. For NewGen module, use /103 command (4.8.j List of available commands for outputs)** For installations with NewGen masters (DGQG02/DGQG04/...), **the cached value in RAM will be returned to client. No request to the module itself will be performed!** In other, it can not be used to ask newer value of a temperature sensor or a I10 input faster than periodical refresh of the Master.	Viewer	None
(none)	- toggle state of the output (if supported). If output is on, it will be turned off. If output is off, it will be turned on.	Admin	None
%I	- set the output - increase a value variable (VAR) - make shutter going up if used with up output - make shutter going down if used with down output	Admin	None

Command	**Description**	**Min. role**	**Arguments**
`%O`	- reset the output - decrease a value variable (`VAR`) - stop shutter (whatever the up or down output used)	Admin	None
`%Dnnn`	- assign a percentage to a dimmable output - set the value of a value variable	Admin	1 to 3 char : decimal value number between 0 and 100 `%D1`, `%D23`, `%D100`
`%I%Dnnn`	- increase value/dimmer/volume of the specified value in argument. **Note**: For volume, 'nnn' value has no effect for DAMPLI01 (AMP) module. Volume will only be increased by 1% **Warning:** This command does not work for DAP versions between 31 and 41.4 (included)	Admin	1 to 3 char : decimal value number between 1 and 100 `%I%D10`
`%O%Dnnn`	- decrease value/dimmer/volume of the specified value in argument **Note**: For volume, 'nnn' value has no effect for DAMPLI01 (AMP) module. Volume will only be increased by 1% **Warning:** This command does not work for DAP versions between 31 and 41.4 (included)	Admin	1 to 3 char : decimal value number between 1 and 100 `%O%D10`
`%DInnn`	Same as `%I%Dnnn` (from version 41.5)	Admin	`%DI10`
`%DOnnn`	Same as `%O%Dnnn` (from version 41.5)	Admin	`%DO10`
`%DB`	- start dimming of the input (dimmer/volume)	Admin	None
`%DE`	- stop dimming of the ouput (dimmer/volume)	Admin	None
`%Xnnn`	- assign a 0-to-255 value to a DMX output	Admin	1 to 3 char : decimal value number between 0 to 255 `%X4`, `%X56`, `%X127`, `%X255`

Command	**Description**	**Min. role**	**Arguments**
`%Tnn.n`	- set a new temperature heating setpoint	Admin	Floating point value in Celcius degrees `%T21.5`
`%Unn.n`	- set a new temperature cooling setpoint	Admin	Floating point value in Celcius degrees `%T25.0`
`%Mn`	- set temperature mode to the specified mode	Admin	One of these values : - `1` = away - `2` = auto - `5` = comfort - `6` = anti-freeze `%M5`
`%Rn`	- set regulation mode to the specified mode	Admin	One if these values : - `0` = off - `1` = heating - `2` = cooling - `3` = mixed
`%Cn` (not yet available)	- start/stop/toggle color cycle of a DMX slave	Admin	One of these values : - `(none)` = toggle - `1` = start - `0` = stop `%C1` will enable color cycle
`%An`	- select source for sound module **Warning:** This command does not work for DAP versions between 31 and 41.4 (included)	Admin	One of these values : - `1` -> `4` : auxiliary source - `5` : FM tuner `%A5` will switch to FM tuner
`%I%A` `%AN`	- jump to next source (from v41.7)	Admin	None
`%I%A` `%AN`	- jump to previous source (from v41.7)	Admin	None
`%Fnnn.nn`	- set tuner frequency	Admin	A floating point value in MHz unit

Command	**Description**	**Min. role**	**Arguments**
	Warning: This command does not work for DAP versions between 31 and 41.4 (included)		`%F99.1` will jump to 99.1MHz
`%I%F` `%FN`	- seek next frequency (from v41.7)	Admin	None
`%O%F` `%FP`	- seek previous frequency (from v41.7)	Admin	None
`%H`	- move shutter up	Admin	None
`%L`	- move shutter low	Admin	None
`%Khh:mm:ss xx DD/MM/YY [-]oh:om:os`	- Change clock data **Warning:** This command does not work for DAP versions between 31 and 42.x (included)	Admin	- hh:mm:ss hour - `xx` : hexadecimal mask of selected days • `01` : Sunday • `02` : Monday • `04` : Tuesday • `08` : Wednesday • `10` : Thursday • `20` : Friday • `40` : Saturday • `80` : clock is disabled - `DD` : limit the execution of the clock to a specific day of the month using two-digit decimal number (zero padded). If 0, the clock will not be restricted to a specific day of the month. - `MM` : limit the execution of the clock to a specific month of the year using two-digit decimal number (zero padded). If 0, the clock will not be restricted to a specific month of the year. - `YY` : limit the execution of the clock to

Command	**Description**	**Min. role**	**Arguments**
			a specific year using two-digit decimal number (zero padded). If 0, the clock will not be restricted to a specific year. **From version 43, only**: - `oh:om:os` : For `[SUNRISE]` and `[SUNSET]` only. Will delay or anticipate, if prefixed with `-`, the execution of astronomical clock.
`%Pn`	- simulate push on a button	Admin	One of these values : - `1`=Begin short push - `2`=End short push - `3`=Begin long push - `4`=End long push

4.6.e) Samples of legacy strings sent to your Domintell installation

Text	Means
`BU1 11-1`	Change/toggle output/led 1 on module DPBU01 with serial number 0x000011
`BU1 11-1%I`	Set (ON) output/led 1 on module DPBU01 with serial number 0x000011
`BU1 11-1%0`	Reset (OFF) output/led 1 on module DPBU01 with serial number 0x000011
`BU1 11%S`	Get Status of input (button) and output (LED's) on module DPBU01 with serial number 0x000011
`BU2 52-2`	Change/toggle output/led 2 on module DPBU02 with serial number 0x000052
`BU4 4F-4`	Change/toggle output 4 on module DPBU04 with serial number 0x00004F and
`BU2 52-2%P1`	Simulate Begin of short push on button 2 of module DPBU02 with serial number 0x000052
`BU6 134-1%P2`	Simulate End of short push on button 1 of module DPBU06 with serial number 0x000134
`IS4 CD-4%P3`	Simulate Begin of long push on input 4 of module DISM04 with serial number 0x0000CD
`IS8 2D8-7%P4`	Simulate End of long push on input 7 of module DISM08 with serial number 0x0002D8
`BIR 3A6-8`	Change output 8 on module DBIR01 with serial number 0x0003A6
`TRV 73-1 or` `TRV 73-2`	Run shutter function of shutter 1 on module DTRV01 with serial number 0x000073. Each time the command is send one step of the following cycle is run : UP-STOP-DOWN-STOP-UP-...

`TRV 73-1%H or`
`TRV 73-1%I or`
`TRV 73-2%H`
	Shutter 1 on module DTRV01 with serial number 0x000073 goes High.
	Returned status will be "TRV 73O01" if not other shutter is ON.
	(since v1.19.17)

`TRV 73-1%L or`
`TRV 73-2%L or`
`TRV 73-2%I`
	Shutter 1 on module DTRV01 with serial number 0x000073 goes Low.
	Returned status will be "TRV 73O02" if not other shutter is ON.
	(since v1.19.17)

`TRV 73-1%O or` `TRV 73-2%O`	Stop shutter 1 on module DTRV01 with serial number 0x000073
`TRV 73-3 or` `TRV 73-4`	Run shutter function of shutter 2 on module DTRV01 with serial number 0x000073. Each time the command is send one step of the following cycle is run : UP-STOP-DOWN-STOP-UP-...

`TRV 73-3%H or`
`TRV 73-3%I or`
`TRV 73-4%H`
	Shutter 2 on module DTRV01 with serial number 0x000073 goes High.
	Returned status will be "TRV 73O04" if not other shutter is ON.
	(since v1.19.17)

`TRV 73-3%L or`
`TRV 73-4%L or`
`TRV 73-4%I`
	Shutter 2 on module DTRV01 with serial number 0x000073 goes Low.
	Returned status will be "TRV 73O08" if not other shutter is ON.
	(since v1.19.17)

`TRV 73-3%O or` `TRV 73-4%O`	Stop shutter 2 on module DTRV01 with serial number 0x000073
`TRV 73-5 or` `TRV 73-6`	Run shutter function of shutter 3 on module DTRV01 with serial number 0x000073. Each time the command is send one step of the following cycle is run : UP-STOP-DOWN-STOP-UP-…

`TRV 73-5%H or`
`TRV 73-5%I or`
`TRV 73-6%H`
	Shutter 3 on module DTRV01 with serial number 0x000073 goes High.
	Returned status will be "TRV 73O10" if not other shutter is ON.
	(since v1.19.17)

`TRV 73-5%L or`
`TRV 73-6%L or`
`TRV 73-6%I`
	Shutter 3 on module DTRV01 with serial number 0x000073 goes Low.
	Returned status will be "TRV 73O20" if not other shutter is ON.
	(since v1.19.17)

`TRV 73-5%O or` `TRV 73-6%O`	Stop shutter 3 on module DTRV01 with serial number 0x000073
`TRV 73-7 or` `TRV 73-8`	Run shutter function of shutter 4 on module DTRV01 with serial number 0x000073. Each time the command is send one step of the following cycle is run : UP-STOP-DOWN-STOP-UP-…

`TRV 73-7%H or`
`TRV 73-7%I or`
`TRV 73-8%H`
	Shutter 4 on module DTRV01 with serial number 0x000073 goes High.
	Returned status will be "TRV 73O40" if not other shutter is ON.
	(since v1.19.17)

`TRV 73-7%L or`
`TRV 73-8%L or`
`TRV 73-8%I`
	Shutter 4 on module DTRV01 with serial number 0x000073 goes Low.
	Returned status will be "TRV 73O80" if not other shutter is ON.
	(since v1.19.17)

`TRV 73-7%O or` `TRV 73-8%O`	Stop shutter 4 on module DTRV01 with serial number 0x000073
`TRP 151-4`	Change output 4 on module DTRP01 with serial number 0x000151

DIM 19F-8	Change output 8 on module DDIM01 with serial number 0x00019F
DIM 19F-6%D50	Set output 6 to 50% on module DDIM01 with serial number 0x00019F
DIM 19F-6%DB	Start dimming on output 6 on module DDIM01 with serial number 0x00019F (v1.17.02)
DIM 19F-6%DE	Stop dimming on output 6 on module DDIM01 with serial number 0x00019F (v1.17.02)
DIM 19F-6%I%D10	Increase by step of 10% the value on output 6 on module DDIM01 with serial number 0x00019F (stop at 100%) (v1.17.02)
DIM 19F-6%O%D7	Decrease by step of 7% the value on output 6 on module DDIM01 with serial number 0x00019F (stop at 0%) (v1.17.02)
LED C2-1	Change output 1 on module DLED01 with serial number 0x0000C2
VAR 1	Change variable 1
TPV 3-1	Change shutter 1 on module DTRP02 with serial number 0x000003
D10 1-1	Change output 1 on module DOUT10V02 with serial number 0x000001
D10 1-1%D60	Set output 1 to 60% on module DOUT10V02 with serial number 0x000001
D10 1-1%I%D5	Increase output value of module DOUT10V02 with serial number 0x000001 by step of 5% (v1.17.02)
D10 1-1%O%D11	Decrease output value of module DOUT10V02 with serial number 0x000001 by step of 11% (v1.17.02)
DMX 1F-2-1%X230	Set channel 1 of device 2 to value 230 of module DDMX01 with serial number 0x00001F
V24 1-1	Change shutter 1 on module DTRVBT01 with serial number 0x000001
TSB 8D%T24.5	Set Heating T° to 24,5°C on module DTSC01/03 with serial number 0x00008D
LT2 34%T22.7	Set Heating T° to 22,7°C on module DTSC02 with serial number 0x000034
LT4 2F%U21.5	Set Cooling T° to 21,5°C on module DTSC04 with serial number 0x00002F
I10 5%S	Ask Status of the input of DIN10V with serial number 0x000005
AMP 3-1%D50%A1	Output 1 to Aux 1 at Volume 50 on module DAMPLI01 with serial number 0x000003

AMP 3-1%I%D15	Increase volume of Output 1 by step of 15% on module DAMPLI01 with serial number 0x000003 (v1.17.02)
AMP 3-1%O%D9	Decrease volume of Output 1 by step of 9% on module DAMPLI01 with serial number 0x000003 (v1.17.02)
AMP 3-2%D60%F99.1%A5	Output 2 to Tuner at Volume 60 & Freq 99,1MHz on module DAMPLI01 with serial number 0x000003
AMP000003-4	Change output 4 volume on module DAMPLI01 with serial number 0x000003
AMP000003%S	Ask status of all output of module DAMPLI01 with serial number 0x000003
BIR 3A6-6%I	Set output 6 on module DBIR01 with serial number 0x0003A6
BIR 3A6-6%O	Reset output 6 on module DBIR01 with serial number 0x0003A6
MEM000001%I	SET Mixed Memo 1 (v1.16.02)
MEM000001%O	RESET Mixed Memo 1 (v1.16.02)
MEM000002%D50	SET 50% to Dimmer Memo 2 (v1.16.03)
MEM000002%I%D5	Increase value of Dimmer Memo 2 by step of 5% (v1.17.02)
MEM000002%O%D17	Decrease value of Dimmer Memo 2 by step of 17% (v1.17.02)
MEM 3%O	Shutter Memo Group : OFF
MEM 3%H	Shutter Memo Group : UP (High)
MEM 3%L	Shutter Memo Group : Down (Low))
SFE000001	SET Sfeer 1 (v1.16.03)
SFE000001%I	SET Sfeer 1 (v1.16.03)
SFE000001%S	Get status of each item in the Sfeer 1 (v1.17.02)
PBL C-6%I	SET DPBTLCD0x 6th output
PBL C-1%O	RESET DPBTLCD0x 1st output
PBL C-1%P2	Simulate begin of short push on button 1 of module DPBTLCD0x with serial number 0x00000C (v1.17.02)
PBL 13%S	Return status (Temp -> only for DPBTLCD02) of module DPBTLCD02 with serial number 0x000013 (v1.17.02)
FAN000001-1%I	Set speed 1
FAN000001-2%I	Set speed 2
FAN000001-3%I	Set speed 3

FAN000001-4%I	Set Heating (if speed different of 0) Advise : change T° sensor setpoint!
FAN000001-5%I	Set Cooling (if speed different of 0) Advise : change T° sensor setpoint!
FAN000001-6%I	Set Manual mode
FAN000001-6%O	Set Automatic mode
DMV00001-1%I	Set speed 1
DMV00001-2%I	Set speed 2
DMV00001-3%I	Set speed 3
DMV00001-4%I	Set Auxiliary 1
DMV00001-5%I	Set Auxiliary 2
ZON000001%I	T° Zone 1, increment setpoint. (T° zones since v1.17.00)
ZON000001%O	T° Zone 1, decrement setpoint.
ZON000001%T15.5	T° Zone 1, setpoint to 15.5°C.
ZON000001%M1	T° Zone 1, set T° mode to absence.
ZON000001%M2	T° Zone 1, set T° mode to automatic.
ZON000001%M5	T° Zone 1, set T° mode to comfort.
ZON000001%M6	T° Zone 1, set T° mode to frost (if frost mode enabled).
CLK000001%K00:22:00 7F 00/05/09	Set Clock 1 at 00h22m00s for all weekdays during month of may (v1.17.02)
CLK000001%K00:22:00 FF 00/05/09	Disable Clock 1 and set datas to 00h22m00s for all weekdays during month of may (v1.17.02)
CLK000001%K01:22:00 08 00/00/00	Set Clock 1 at 01h22m00s each Wednesday (v1.17.02)
CLK000004%K00:00:00 7F 00/00/00 -00:10:00	Set astronomical clock (e.g. clock 4) for all weekdays any days of the year by anticipating it by 10 minutes (from v43).
DAL 10-32%D100	DINTDALI01 #0x10 output 0x32 request @ 100%
PRL C-6%I	SET DPBRLCD02 6th output
PRL C-1%O	RESET DPBRLCD02 1st output
PRL C-1%P2	Simulate begin of short push on button 1 of module DPBRLCD02 with serial number 0x00000C (v1.27.01)
PRL 13%S	Return status of module DPBRLCD02 with serial number 0x000013 (v1.27.01)

4.7. Legacy output LightProtocol (*LightProtocol server* to *LightProtocol client*)

4.7.a) Status Frame description

Mod Type (3 char)	Serial Number (6 char hexadecimal)	(optional) IO number (-x : minus char + IO number in 1 hexa digit) DINTDALI requires 2 hexa digit	Data Type (1 char)	Datas (n * 2 char hexa)

4.7.b) Data Types

Type	***Means***	***Description*** (leading '0' can be replaced by ' ' (space))
`I`	Inputs	LSB = input 1, MSB = input 8. If module has more than 8 inputs, several bytes will be concatenated.
`O`	Outputs	LSB = output 1, MSB = output 8. If module has more than 8 outputs, several bytes will be concatenated.
`D`	Pecent value (Dimmers, 0-10V analog input/output)	2 bytes by output (%) Example : '`6432`' = first output at 100%, second output at 50%
`X`	0 to 255 value (DMX)	2 bytes by channel Example : '`C00080`' = first channel is at 192, second channel is at 0 and third channel is at 128. From version 39.1, an extra byte '`CC`' is sent to notify when the color cycle is running (only for RGBx slaves). Please note that values of RGB channels are not relevant when '`CC`' byte is set. For example: '`C00080CC`' means that color cycle is running for this RGB slave. If byte is missing, color cycle is not running.
`T`	Temperature heating setpoint	Example : '`20.5 22.0 AUTO 18.0`' 1st T° = measure (with software offset) 2nd T° = Heating setpoint value Sensor T° Mode (ASCII string): • ABSENCE = away mode • AUTO = automatic mode • COMFORT = comfort mode • FROST = no frost mode 3rd T° = Heating profile value
`U`	Temperature cooling setpoint	Example : '`20.5 22.0 HEATING 18.0`' 1st T° = measure (with software offset) 2nd T° = cooling setpoint value Sensor Regulation Mode (ASCII string): • OFF = the regulation is disabled • HEATING = the regulation is in heating mode

Type	***Means***	***Description*** (leading '0' can be replaced by ' ' (space))
		• COOLING = the regulation is in cooling mode • MIXED = the regulation is in mixed mode (heating and cooling) • AUTOHVAC = the regulation is in automatic mode (for HVAC only) • DRY = the regulation is in dry mode (for HVAC only) • FAN = the regulation is in fan mode (for HVAC only) 3rd T° = cooling profile value
`C`	Infrared remote control key	Two bytes in hexadecimal format Example : - Key 1 pressed : '`01`' - Key 10 pressed : '`0A`' - Key 32 pressed : '`20`' - Key released : '`00`'
`S`	Sound	'`1-32-TUNE-63-03E8`' = Output 1 – 50% - Source Tuner – 99,1000 Mhz (Since module version 5)
`P`	Temp. Plage	Example : `12:32:00 21.6` 1st = hh:mm:ss 2nd = setpoint value
`K`	Clocks (before v43.0)	Pattern : `HH:MM:SS WD DD/MM/YY` • `HH:MM:SS` = time of execution. Time is expressed in 24-hour format. The leading zero is always present for each field. ◦ `HH` (hour), `MM` (minute) and `SS` (seconds) fields will always have two digits. • `WD` = Day mask (must be OR-ed) in hexa format (without leading "0x" !). Must always have two symbols (`7F`, `01`, `06`) ◦ 0x01 = Sunday ◦ 0x02 = Monday ◦ 0x04 = Tuesday ◦ 0x08 = Wednesday ◦ 0x10 = Thursday ◦ 0x20 = Friday ◦ 0x40 = Saterday ◦ 0x80 = disable clock ◦ `WD` (weekdays) will always have two symbols • `DD/MM/YY` = Date restriction. ◦ `00` (zero) means no restriction for the related field. ◦ 00/00/23 : Clock will only be executed on year 2023. ◦ 00/10/23 : Clock will only be executed during October of year 2023. ◦ 00/01/00 : Clock will only be executed on

Type	***Means***	***Description*** (leading '0' can be replaced by ' ' (space))
		January of each year. ◦ 01/00/00 : Clock will only be executed on first day of each month. ◦ `DD` (day), `MM` (month) and `YY` (year) fields will always have two digits. Example 1: `14:00:00 7F 01/00/00` • Basic clock that will be executed at 14h00m00 each first day of the month whatever the day of the week Example 2: `14:00:00 FF 00/00/00` • Basic clock that must be executed at 14h00m00 everyday but the clock is disabled
	Clocks (from v43.0)	Pattern : `HH:MM:SS WD DD/MM/YY HH':MM':SS'` • `HH:MM:SS` = time of execution. Time is expressed in 24-hour format. The leading zero is always present for each field. ◦ `HH` (hour), `MM` (minute) and `SS` (seconds) fields will always have two digits. • `WD` = Day mask (must be OR-ed) in hexa format (without leading "0x" !). Must always have two symbols (`7F`, `01`, `06`) ◦ 0x01 = Sunday ◦ 0x02 = Monday ◦ 0x04 = Tuesday ◦ 0x08 = Wednesday ◦ 0x10 = Thursday ◦ 0x20 = Friday ◦ 0x40 = Saterday ◦ 0x80 = disable clock ◦ `WD` (weekdays) will always have two symbols • `DD/MM/YY` = Date restriction. ◦ `00` (zero) means no restriction for the related field. ◦ 00/00/23 : Clock will only be executed on year 2023. ◦ 00/10/23 : Clock will only be executed during October of year 2023. ◦ 00/01/00 : Clock will only be executed on January of each year. ◦ 01/00/00 : Clock will only be executed on first day of each month. ◦ `DD` (day), `MM` (month) and `YY` (year) fields will always have two digits. • `HH':MM':SS'` = offset to be applied to the time of execution of the clock (`HH:MM:SS` field) (only present if the clock is an astronomical clock

Type	***Means***	***Description*** (leading '0' can be replaced by ' ' (space))
		"SUNSET" and "SUNRISE" !) ◦ `(nothing)` = postpone the execution time ◦ `-` = anticipate the execution time Example 1: `14:00:00 7F 01/00/00` • Basic clock that will be executed at 14h00m00 each first day of the month whatever the day of the week Example 2: `20:38:49 7F 00/00/00 00:30:00` • Sunset clock that will be executed everyday at 20h38m49 + 30 min = 20h48m49 Example 3: `07:14:23 41 00/00/00 -00:10:00` • Sunrise clock that will be executed only the week-end at 7h14m23 - 10 min = 7h04m23

4.7.c) Sample of received strings from your Domintell installation

Text	**Means**
`PONG`	answer from DRS23202/DETH02 after a string "PING"
`MOD_VERSION=SER_V0A`	answer from DRS23202 after a string "MOD_VERSION" (hexa)
`MOD_VERSION=ETH_V01_STK_V01`	answer from DETH02 after a string "MOD_VERSION" (hexa)
`TE1 6CT25.2 21.0 AUTO 19.5`	Heating T° infos of DTEM01 with serial number 0x6C
`TE1 6CU25.2 21.0 HEATING 19.5`	Cooling T° infos of DTEM01 with serial number 0x6C
`TE2 58T20.9 21.0 COMFORT 21.0`	Heating T° infos of DTEM02 with serial number 0x58
`TE2 58U20.9 28.0 MIXED 28.0`	Cooling T° infos of DTEM02 with serial number 0x58
`BU1 11O00`	Outputs OFF on module DPBU01 with serial number 0x000011
`BU2 52O01`	led 1 ON on module DPBU02 with serial number 0x000052
`BU4 4FO00`	Outputs OFF on module DPBU04 with serial number 0x00004F
`BU6 8AO00`	Outputs OFF on module DPBU06 with serial number 0x00008A
`BIR 3A6O00`	Outputs OFF on module DBIR01 with serial number 0x0003A6

TRV 73O00	Outputs OFF on module DTRV01 with serial number 0x000073
TRP 151O00	Outputs OFF on module DTRP01 with serial number 0x000151
DIM 19FD 064 0 0 0 0 0 0	Dim 2 = 100% on module DDIM01 with serial number 0x00019F
LED C2O00	Outputs OFF on module DLED01 with serial number 0x0000C2
IS4 7I00	Inputs OFF on module DISM04 with serial number 0x000007
IS8 4F8I10	Key 5 ON on module DISM08 with serial number 0x0004F8
BU1 11I00	Buttons released on module DPBU01 with serial number 0x000011
BU2 52I00	Buttons released on module DPBU02 with serial number 0x000052
BU4 4FI00	Buttons released on module DPBU04 with serial number 0x00004F
BU6 8AI10	Button 5 pressed on module DPBU06 with serial number 0x00008A
BR2 10I00	Buttons released on module DPBR02 with serial number 0x000010
BR4 4FI02	Button 2 pressed on module DPBR04 with serial number 0x00004F
BR6 30O10	Led Output 5 ON on module DPBR06 with serial number 0x000030
B81 11I01	Button 1 pressed on module DPBL01 with serial number 0x000011
B82 52I00	Buttons released on module DPBL02 with serial number 0x000052
B84 4FI00	Buttons released on module DPBL04 with serial number 0x00004F
B86 8AI00	Buttons released on module DPBL06 with serial number 0x00008A
VI1 1I01	Button pressed on DVIP01 with serial number 0x000001
VI2 3I02	Button 2 pressed on DVIP01 with serial number 0x000003
LCD 25I00	Inputs OFF on module DLCD01 with serial number 0x000025
VAR 1O01	Variable 1 True
VAR000001O00	Variable 1 False

VAR 1D64	Variable 1 100%
SYS 2O01	System Variable 2 has value 1
TPV 3O01	shutter 1 : UP on module DTRP02 with serial number 0x000003
D10 1D32	50% on module DOUT10V02 with serial number 0x000001
V24 1O01	shutter 1 : UP on module DTRVBT01 with serial number 0x000001
PBL CO00	Outputs OFF on module DPBTLCD0x with serial number 0x00000C
PBL CT24.0 18.0 AUTO 12.0	Temperature on module DPBTLCD02 with serial number 0x00000C
PBL CI01	Push Button 1 on DPBTLCD with serial number 0x00000C
PBL CI00	Release Button 1 on DPBTLCD with serial number 0x00000C
PBL CO00	DPBLCD0xwith serial number 0x00000C outputs are OFF
PBL CO02	2nd DPBLCD0xwith serial number 0x00000C output is ON
PRL CO00	Outputs OFF on module DPBRLCD0x with serial number 0x00000C
PRL CT24.0 18.0 AUTO 12.0	Temperature on module DPBRLCD02 with serial number 0x00000C
PRL CI01	Push Button 1 on DPBRLCD02 with serial number 0x00000C
PRL CI00	Release Button 1 on DPBRLCD02 with serial number 0x00000C
PRL CO00	DPBRLCD02 with serial number 0x00000C outputs are OFF
PRL CO02	2nd DPBRLCD02 with serial number 0x00000C output is ON
AMP 3S1-1D-TUNE-6A-0FA0	Output 1, 29%, Tuner, 106.4000MHz on DAMPLI01 with serial 0x03
AMP 3S3-32-AUX1-64-0000	Output 3, 50%, Aux 1, 100.0000MHz on DAMPLI01 with serial 0x03
FAN000001O20	DFAN01 module with serial number 0x000001 is OFF, manual mode
FAN000001O11	DFAN01 with serial number 0x01 is cooling @ speed 1, auto mode
FAN000001O0C	DFAN01 with serial number 0x01 is heating @ speed 3, auto mode

FAN000001O32	DFAN01 with serial number 0x01 is cooling @ speed 2, manual mode
DMV000001O01	DMV01 with serial number 0x01 has speed 1 enabled
DMV00000100A	DMV01 with serial number 0x01 has speed 2 and auxiliary 1 output enabled
DMV00000101A	DMV01 with serial number 0x01 has speed 2 and auxiliary 1 and 2 output enabled
DAL 10-08D64	DINTDALI01 #0x10 output 0x08 status @ 100%
I10000005D32	Input = 50% on DIN10V02 with serial number 0x000005
DMX 1F-2-00EB000000000000	String with 2nd '-' is obsolete since v11(DETH02) & v16(DRS23202) Device 2 connected to DDMX01 module with serial number 0x00001F has its 2nd channel set to 234
DMX 1F-2X00EB000000000000	
MBD 201T22.7 23.0 AUTO 23.0	Heating T° infos of ModBus Device with serial number 0x201
MBD 201U22.7 26.0 OFF 26.0	Cooling T° infos of ModBus Device with serial number 0x201
MBD 201D 3 2	Device specific values for ModBus Device with serial number 0x201
CLK 2K08:05:00 7F 00/00/00 Clock[SUNRISE]	Clock 2 is an astronomical sunrise clock set (this week) to 8h05m00s all weekdays
CLK 4K08:05:00 7F 00/00/00 Clock[SUNRISE] [READONLY]	Clock 4 is an system astronomical sunset clock set (this week) to 8h05m00s all weekdays. It can not be modified ! **Note**: Since DAP version 43.0.0 or higher
TPR 2Range N°2	Profile 2 is named 'Range N°2'
TPL 8P15.5-02:45:00	Setpoint of Range 8 will be 15.5°C from 2h45m00s
STA 1STU BRU[FM=64-1770]	Station 1 « STU BRU » @ FM 100,6000MHz
14:34 29/12/22 (DGQG01) 14:34 29/12/2022 (NewGen)	Current time of the DGQG seant each minute. - The format of the time is HH:MM in 24 hours. - The format of the date is DD/MM/YY for installations with DGQG01. - The format of the date is DD/MM/YYYY for installations with NewGen DGQG (DGQG02/04/...). **Note**: Each field always has a width of 2 digits. A leading 0 (zero) is added if required. With NewGen DGQG, year always has a width of 4 digits.

	Warning: The time is not sent for DAP versions between 31.0 and 41.6 (included).
`!! PLEASE UPGRADE DETH02 FIRMWARE`	This string means that DETH02 has an incompatible version regarding the current OS version in the Master/DGQG01. This can also occur if status of a new module's type is received by DETH02/DRS23202 and is not handled by its firmware. Bad/missing information can be sent by DETH02 until its firmware is updated.

4.7.d) Decoding APPINFO

- Warnings/Errors
  They starts with an exclamation mark ("`!`") and must be shown to user and ask him to contact Domintell support.
  - `!! PLEASE UPGRADE DRS23202 FIRMWARE ≥ 18 !!` or `!! PLEASE UPGRADE DETH02 FIRMWARE >= 17 !!`
    DRS23202/DETH02 needs an update of its firmware to be able to decode information send by master/DGQG01 or new references of modules. Customer's application (i.e. Smartphone App) can still work but some status/commands can not anymore until the firmware is updated.
  - `! PLEASE RESTART MASTER 0x???????? !`
    Where `????????` is the serial number of the module that is not in the module table of the DRS23202/DETH02 module. The DRS23202/DETH02 module was not connected to the bus when (1) the application has been sent to master/DGQG01, (2) the master has been restarted and the DRS23202/DETH02 module did not receive the new table. Or The specified module has been added after the bus has been scaned by the master/DGQG01.

- APPINFO line gives information about the DAP/configuration file :
  "`APPINFO (PROG M 33.0 00/00/00 00h00 Rev=3 CP=UTF-8) => Smith_v33_v7.dap :`"
  - `PROG M 33.0 … Rev=0` : stands for DAP file version 33.0.0.
  - Date/Hour field is deprecated from version 31.
  - `CP=UTF-8` (only starting from OS version 1.27.06) : specifies the Windows charset (CP=1252) Unicode charset used for non-ASCII accentuated characters. CP1252 is also known as Windows-1252. See Windows code page list on Wikipedia for more information.
  - `Smith_v33_v7.dap` is the application name (truncated to 32 characters).

- Room/floor information where the input/output/memo/ambiance are located are given just after the name of the item. and will look like `[_house_|_floor_/_room_]` where `_house_`, `_floor_` and `_room_` are replaced by the real house, floor and room names (without the underscores "`_`")

- Some devices has extra type information like DINTDALI (`DAL`) IO's or DDMX01 (`DMX`) IO's just after the room/floor information (`[House|Ground|Living]`)

- DINTDALI01 outputs (`DAL`) : `[TYPE=xx]` specifies the type of the DALI slave where "`xx`" can be :
  `TL` : Device Type 0 for fluorescent lamps (IEC 62386-201)
  `ER` : Device Type 1 for self-contained emergency lighting (IEC 62386-202)
  `DISC` : Device Type 2 for discharge lamps (IEC 62386-203)
  `LOWV` : Device Type 3 for low voltage halogen lamps (IEC 62386-204)
  `INCA` : Device Type 4 for supply Voltage controller for incandescent lamps (IEC 62386-205)
  `DC` : Device Type 5 for conversion from digital into D.C. voltage (IEC 62386-206)
  `LED` : Device Type 6 for LED modules (IEC 62386-207)
  `SW` : Device Type 7 for switching function (IEC 62386-208)
  `RGB` : Device Type 8 for colour control (IEC 62386-209)
  Please note that all device types can not be handled by DINTDALI01 firmware.

- DDMX01 outputs (`DMX`) : `[x CHANNELS]` specifies how much channels are configured for this DDMX01 output where "`x`" can be a value between 1 to 8

- DIN10V02 (`I10`):
  `I10 1-10-10V input[House||]`**`[ANALOG-MIN=-10.5-MAX=120-UNIT=°C]`**
  For `PROG M` earlier to 39.1 : no additional information given after house location.
  From `PROG M 39.1` : following data are given after the house location : `[ANALOG-MIN=<min>-MAX=<max>-UNIT=<unit>]`
  - `<min>` : Physical value/string associated to a status of 0% (e.g. "-10.5")
  - `<max>` : Physical value/string associated to a status of 100% (e.g. "120").
  - `<unit>` : The unit of the physical value (e.g. "°C", "l", "m³", ...).
  - The cross multiplication using `MIN` and `MAX` values must be used to convert the '`D`' status (in percent) to have the physical value. For example, if the status is received `I10 1D32` = 50%

$$\frac{120\,°C-(-10.5\,°C)}{100\,\%}\times 50\,\%=65.25\,°C$$

- Group (`MEM`) :
  The type of the group gives which commands can be used to control it. See 4.5 I/O mapping of all modules and 4.8.j List of available commands for outputs for available commands. The type is specified between brackets and can be one of the following:
  - `[MIX]`: contains any kind of outputs. Only commands use for a `BIR` module are accepted.
  - `[DIMMERS]`: only contains monochrome dimmable outputs. Only command use for a `DIM` module are accepted.
  - `[SHUTTERS]`: only contains shutters outputs. Only commands use for a `TRV` module are accepted.
  - `[DMX]`: only contains DMX outputs. Only commands used for a `DMX` module are accepted.

- `[FAN]`: only contains DFAN01 outputs. Only commands used for a `FAN` module are accepted.
- `[DMV]`: only contains fan outputs. Only commands used for a `DMV` module are accepted.
- `[DALI]`: only contains DALI (light) outputs. Only commands used for a `DAL` module are accepted.
- `[DLB]`: only contains DLB outputs. Only commands used for a `LV1` module are accepted.
- `[RGBW]`: only contains RGBW (non DMX) outputs. Only commands used for a `RW1` module are accepted.
- `[SOUND]`: only contains sound outputs. Only commands used if a `AMP` module are accepted.

The output that must be used to reflect the status of the group is specified using `[REF=]` tag.
Format of group frame is always in Legacy format. The reference output used in `[REF=]` will be in legacy or NewGen format depending of the type of its module.

- `MEM 1Group #1[House||][MIX][REF=BU4 EE4-5]`
  This is a mixed group. Only commands %I, %O or none to toggle can be used.
  The group must be displayed as on/off, when 5th LED indicator of BU4 (s/n 0xEE4) is on/off.
- `MEM 2Group #2[House||][RGBW][REF=RW1/69/46/1]`
  This is a RGBW group

The reference output is given using [Ref=] tag (see 4.7.d Decoding APPINFO). It must be used to define the current state of the group.
Important note: A bug is present from version 31 to 42.3.1 (inclusive) when the reference output is a shutter. With these versions, offsets for the 4 outputs are 1, 2, 3, 4 while it should be 1, 3, 5, 7. So if the version of the DAP (`PROG M`) is **between 31 and 42.3.1 inclusive**, the following alogirthm must be used to have the correct output offset:
- ***((<IO offset> << 1) - 1***
- Example 1 : `[REF=TRV 456-2]` -> (2 << 1) - 1 = 3. The new IO offset to process if `[REF=TRV 456-3]` which is indeed the first relay related to the second shutter output.
- Example 2 : `[REF=TRV 456-3]` -> (3 << 1) - 1 = 5. The new IO offset to process is `[REF=TRV 456-5]` which is indeed the first relay related to the third shutter output.

◦ Use same status and command than the reference IO. For mixed group, only statuses and command of relay output are used/available even if a dimmer output or a DMX output is the reference output.

◦ DAMPLI01 (`AMP`)
`AMP 1-1LS 1[House||]`**`[MAXVOL=90|TUNER=1,FM|AUX1=1,IPod|AUX2=1,Bluetooth|AUX3=0,Aux 3|AUX4=1,NAS]`**
For `PROG M` earlier to 42 : no additional information given after house location.
From `PROG M 42.0` : following data are given after the house location :
`[MAXVOL=<vol>|TUNER=<enabled>,FM|AUX1=<enabled>,<name>|AUX2=<enabled>,<name>|AUX3=<enabled>,<name>|AUX4=<enabled>,<name>]`

- `<vol>` : Maximum volume in decimal value
- `<enabled>` : if 0, source is disabled/not available
- `<name>` : user friendly name of the source

- Clocks (`CLK`) :

`CLK 1K08:30:00 7F 00/00/00 NAME[House||]`

Attention: the format in APPINFO from version `PROG M 43.0` is not backward compatible with previous format ! The parser of the APPINFO must be adjusted !

For `PROG M 30.0` and earlier: `HH:MM:SS WD DD/MM/YY NAME[TYPE]`

- `HH:MM:SS` : current execution time of the clock. Time is expressed in 24-hour format. All fields will always have two digits (leading zero added if required).
- `WD` : days of the week for which clock will be executed. In hexadecimal format (without leading "0x" !) :
  - 0x01 = Sunday
  - 0x02 = Monday
  - 0x04 = Tuesday
  - 0x08 = Wednesday
  - 0x10 = Thursday
  - 0x20 = Friday
  - 0x40 = Saterday
  - 0x80 = disabled clock
  - Will always have two symbols
- `DD/MM/YY` : date restriction of the execution of the clock:
  - `00` (zero) means no restriction for the related field.
  - 00/00/23 : Clock will only be executed on year 2023.
  - 00/10/23 : Clock will only be executed during October of year 2023.
  - 00/01/00 : Clock will only be executed on January of each year.
  - 01/00/00 : Clock will only be executed on first day of each month.
  - `DD` (day), `MM` (month) and `YY` (year) fields will always have two digits.
- `NAME` : Name of the clock (can contain spaces)
- `[TYPE]` : Type of the clock:
  - `(none)` : this is a basic clock
  - `[SUNRISE]` : astronomical clock at sunrise
  - `[SUNSET]` : astronomical clock at sunset

From `PROG M 31.0` to `PROG M 42.0` (included) : `HH:MM:SS WD DD/MM/YY NAME[TYPE]` Clock may not properly work for these versions !

- `HH:MM:SS` : current execution time of the clock. Time is expressed in 24-hour format. All fields will always have two digits (leading zero added if required).
- `WD` : days of the week for which clock will be executed. In hexadecimal format (without leading "0x" !) :
  - 0x01 = Sunday
  - 0x02 = Monday
  - 0x04 = Tuesday
  - 0x08 = Wednesday

    - 0x10 = Thursday
    - 0x20 = Friday
    - 0x40 = Saterday
    - 0x80 = disabled clock
    - Will always have two symbols
  - `DD/MM/YY` : date restriction of the execution of the clock:
    - `00` (zero) means no restriction for the related field.
    - 00/00/23 : Clock will only be executed on year 2023.
    - 00/10/23 : Clock will only be executed during October of year 2023.
    - 00/01/00 : Clock will only be executed on January of each year.
    - 01/00/00 : Clock will only be executed on first day of each month.
    - `DD` (day), `MM` (month) and `YY` (year) fields will always have two digits.
  - `NAME` : Name of the clock (can contain spaces)
  - `[TYPE]` : Type of the clock:
    - `(none)` : this is a basic clock
    - `[SUNRISE]` : astronomical clock at sunrise
    - `[SUNSET]` : astronomical clock at sunset
    - `[RESET]` : clock executed each time Master boots. This clock is read-only

  From `PROG M 43.0` and later : `HH:MM:SS WD DD/MM/YY NAME[BUILDING|FLOOR|ROOM][TYPE][READONLY]`
  - **Backward compatibility broken !**
  - All fields before `NAME` are kept for backward compatibility. Their values are useless and **must be ignored** ! Values must be retrieved using `PING` command
  - `NAME` : Name of the clock (can contain spaces)
  - `[BUILDING|FLOOR|ROOM]` : Location is the house.
  - `[TYPE]` : Type of the clock:
    - `(none)` : this is a basic clock
    - `[SUNRISE]` : astronomical clock at sunrise. System clock will be read-only
    - `[SUNSET]` : astronomical clock at sunset. System clock will be read-only
    - `[RESET]` : clock executed each time Master boots. This clock is read-only
  - `[READONLY]` : Only present if clock is read-only. Command %K will have no effect on this clock.

- Temperature sensors : `[LOCAL][HMR=0x00-HMT=0x00][LHH=30.0-LHL=10.0-LCH=40.0-LCL=20.0-ISP=0.5]` :

  `LOCAL` : Deprecated. Only for backward compatibility

  `HMR` : Hide Mode Regulation. When a bit is set, functionnality/capability must be hidden to user. In other words, when a bit is set, user can not request the functionnality.
  - 0x01 : If set, user will not be able to turn off regulation
  - 0x02 : If set, user will not be able to request heating regulation

- 0x04 : If set, user will not be able to request cooling regulation
- 0x08 : If set user will not be able to request mixed regulation
- 0x10 : If set user will not be able to request auto HVAC regulation (from `PROG M 42.0`)
- 0x20 : If set user will not be able to request dry regulation (from `PROG M 42.0`)
- 0x40 : If set user will not be able to request fan regulation (from `PROG M 42.0`)

`HMT` : Hide Mode Temperature. When a bit is set, functionnality/capability must be hidden to user. In other words, when a bit is set,user can not request the functionnality.

- 0x01 : If set, user will not be able to request automatic temperature mode
- 0x02 : If set, user will not be able to request comfort temperature mode
- 0x04 : If set, user will not be able to request away temperature mode
- 0x08 : If set, user will not be able to request frost temperature mode

`LHH` : Limit Heating High (from `PROG M 37.0`) : High limit for heating setpoint

`LHL` : Limit Heating Low (from `PROG M 37.0`) : Low limit for heating setpoint

`LCH` : Limit Cooling High (from `PROG M 37.0`) : High limit for cooling setpoint

`LCL` : Limit Cooling Low (from `PROG M 37.0`) : Low limit for cooling setpoint

`ISP` : Increment SetPoint (from `PROG M 37.0`) : Increment step for the setpoint (for example how much setpoint must be increased/decreased when '+' or '-' buttons are pressed)

- Cameras (`CAM`) : `CAM 1Dome[TYPE][IP=x.x.x.x][FORMAT=URL]`

`[TYPE]` : Type of camera :

- `[CUST]` : This is a custom camera.
- `[AXIS]` : This is an auto-detected Axis camera. Auto guessed URL. In this case, `[FORMAT=URL]` tag is omitted. (deprecated)
- `[DST]` : This is the built-in camera of a doorstation (for future use).

`[IP=x.x.x.x]` : This is the IP/host of the camera (deprecated).

`[FORMAT=x]` : Format of the video stream and its URL :

- `[MJPG=]` : The stream is a MJPEG stream. But also used for any live stream using h.264 codec or any other codec.
- `[JPG=]` : URL only delivers a static JPEG image by call. URL must be periodically polled to simulate a live stream.

`URL` : URL of the video stream will contains the scheme, the credentials, the host information and the path. See APPINFO dump below for example.

- The application (APPINFO) is fully retrieved when the string beginning with "`END APPINFO`" is received.

- Example of received strings with APPINFO command :

```
!! PLEASE UPGRADE DRS23202 FIRMWARE >= 24 !!
```

```
!! PLEASE UPGRADE DETH02 FIRMWARE >= 25 !!
APPINFO (PROG M 1.27 04/11/16 09h28 Rev=3) => TEST_APPINFO.dap :
FRO     1 : 1
RS2     2[VERS=0x10]Interface protocole RS[House|||]
ET2    B6[VERS=0x0B]MOD DETH02[House|||]
TE1     1-1Sensor DTEM01[House|||][LOCAL][HMR=0x00-HMT=0x00][LHH=30.0-
LHL=10.0-LCH=40.0-LCL=20.0-ISP=0.5]
BIR   4C9-1BIR 1[House|1st floor|living]
BIR   4C9-2BIR 2[House|1st floor|living]
BIR   4C9-3BIR 3[House|1st floor|kitchen]
BIR   4C9-4BIR 4[House|1st floor|kitchen]
BIR   4C9-5BIR 5[House|2nd floor|]
BIR   4C9-6BIR 6[House|||]
BIR   4C9-7BIR 7[House|||]
BIR   4C9-8BIR 8[House|||]
TRV   3E9-1TRV 1[House|||]
TRV   3E9-3TRV 2[House|||]
TRV   3E9-5TRV 3[House|||]
TRV   3E9-7TRV 4[House|||]
DMV     1-1Output DMV01[House|||]
DMV     1-2Output DMV01 2[House|||]
DMV     1-3Output DMV01 3[House|||]
DMV     1-4Auxiliary 1[House|||]
DMV     1-5Auxiliary 2[House|||]
PBL   E6C-1Input PB 1[House|||][NOLINK]
PBL   E6C-2Input PB 2[House|||][NOLINK]
PBL   E6C-3Input PB 3[House|||][NOLINK]
PBL   E6C-4Input PB 4[House|||][NOLINK]
PBL   E6C-7T° sensor DPBTLCD0x[House|||]
PBL   E6C-8Led PB 1[House|||]
PBL   E6C-9Led PB 2[House|||]
PBL   E6C-ALed PB 3[House|||]
PBL   E6C-BLed PB 4[House|||]
LT4     1-5T° sensor DTSC04[House|||]
LT4     1-6IR sensor DTSC04[House|||]
LT4     1-BOutput DTSC04 1[House|||]
LT4     1-COutput DTSC04 2[House|||]
LT4     1-DOutput DTSC04 3[House|||]
LT4     1-EOutput DTSC04 4[House|||]
LT4     1-15Lock[House|||]
BU6   24B-1Input B6 1[House|||][PUSH=LONG]
BU6   24B-2Input B6 2[House|||][PUSH=LONG]
BU6   24B-3Input B6 3[House|||][NOLINK]
BU6   24B-3Input B6 3[House|||][NOLINK]
BU6   24B-4Input B6 4[House|||][PUSH=SHORT]
BU6   24B-5Input B6 5[House|||][PUSH=SHORT]
BU6   24B-6Input B6 6[House|||][NOLINK]
BU6   24B-7LED B6 1[House|||]
BU6   24B-8LED B6 2[House|||]
BU6   24B-9LED B6 3[House|||]
BU6   24B-ALED B6 4[House|||]
BU6   24B-BLED B6 5[House|||]
BU6   24B-CLED B6 6[House|||]
DIM   21B-1DIM 1[House|||]
DIM   21B-2DIM 2[House|||]
DIM   21B-3DIM 3[House|||]
DIM   21B-4DIM 4[House|||]
DIM   21B-5DIM 5[House|||]
DIM   21B-6DIM 6[House|||]
DIM   21B-7DIM 7[House|||]
DIM   21B-8DIM 8[House|||]
TSB   236-5T° sensor Touch[House|||]
TSB   236-6IR sensor Touch[House|||]
TRP   691-1TRP 1[House|||]
```

```
TRP   691-2TRP 2[House||]
TRP   691-3TRP 3[House||]
TRP   691-4TRP 4[House||]
BU2     9-1Input B2 1[House||][PUSH=SHORT]
BU2     9-2Input B2 2[House||][NOLINK]
BU2     9-3LED B2 1[House||]
BU2     9-4LED B2 2[House||]
TE1   9DE-1T° sensor T1[House||]
V24     A-1TRV BT[House||]
I10     5-1Input 0-10V [House||]
AMP   105-1HP 1[House||]
AMP   105-2HP 2[House||]
AMP   105-3HP 3[House||]
AMP   105-4HP 4[House||]
FAN   267-1DFAN[House||]
FAN   268-1DFAN[House||]
DMR     3-1DMR 1[House||]
DMR     3-2DMR 2[House||]
DMR     3-3DMR 3[House||]
DMR     3-4DMR 4[House||]
DMR     3-5DMR 5[House||]
DMX    91-1DMX Output 1 RGBI[House||][4 CHANNELS]
DMX    91-1-CH1:Chan. R[R 0x00-0xFF]
DMX    91-1-CH2:Label G[G 0x00-0xFF]
DMX    91-1-CH3:Chan. B[B 0x00-0xFF]
DMX    91-1-CH4:Chan. I[I 0x00-0x64]
DMX    91-2DMX Output 2 II[House||][2 CHANNELS]
DMX    91-2-CH1:Chan. 1[I 0x00-0xFF]
DMX    91-2-CH2:Chan. 2[I 0x00-0xFF]
DMX    91-3DMX Output 3 I[House||][1 CHANNELS]
DMX    91-3-CH1:Chan. 1[I 0x00-0xFF]
DAL    10-01TL #12345678-1[House||][TYPE=TL] (!DALI Out number = 2
digits!)
DAL    10-02LED #87654321-2[House||][TYPE=LED]
DAL    10-03PHASE #87654321-2[House||][TYPE=INCA]
B81     2-1Button 1[House|Floor|Room]
B82    12-2Button 2[House||]
B84    36-3Button 3[House||]
B86    72-4Button 4[House||]
B86    89-7LED B6 1[House||]
B84   347-6LED B4 2[House||]
B82    84-3LED B2 1[House||]
B81    39-2LED B1 1[House||]
PRL   E6C-1PBRLCD Input 1[House||][NOLINK]
PRL   E6C-2PBRLCD Input 2[House||][NOLINK]
PRL   E6C-3PBRLCD Input 3[House||][NOLINK]
PRL   E6C-4PBRLCD Input 4[House||][NOLINK]
PRL   E6C-7PBRLCD T° sensor[House||]
PRL   E6C-8PBRLCD Led 1[House||]
PRL   E6C-9PBRLCD Led 2[House||]
PRL   E6C-APBRLCD Led 3[House||]
PRL   E6C-BPBRLCD Led 4[House||]
VAR     1My variable[House|Floor|Room][BOOL]
VAR     2My variable 2[House|Floor|Room][VALU,00->100,LOOP]
SYS     0Presence simulation[House||][BOOL]
SYS     1T° mode[House||][VALU,1-2-5-6,LOOP]
SYS     2Regulation mode[House||][VALU,00->03,LOOP]
SYS     9Day[House||][BOOL][READONLY]
MEM     1Memo 1[House||][MIX][REF=BIR   4C9-1]
MEM     2Memo 2[House||][SHUTTERS][REF=TRV   3E9-1]
MEM     3Memo 3[House||][DIMMERS][REF=DIM   21B-1]
MEM     4Memo 4[House||][SOUND][REF=AMP   105-1]
MEM     5Memo 5[House||][FAN][REF=FAN   267-1]
SFE     1Sfeer 1-Scene 1[House||]
```

```
SFE     2Sfeer 1-Scene 2[House||]
ZON     1Zone 1[House||]
CLK     1K08:05:00 7F 00/00/00 Clock[SUNRISE][READONLY]
CLK     2K18:02:00 7F 00/00/00 Clock[SUNSET][READONLY]
CLK     3K00:38:00 7F 04/01/00 Clock
CLK     4K08:05:00 7F 00/00/00 Clock[SUNRISE]
CLK     5K18:02:00 7F 00/00/00 Clock[SUNSET]
CLK     6K00:00:00 7F 00/00/00 Clock[RESET]
TPR     1Range N°1
TPL     0P12.0-00:00:00
TPL     1P26.5-05:00:00
TPL     2P12.0-07:00:00
TPL     3P 5.0-13:45:00
TPL     4P12.0-15:45:00
TPL     5P20.0-20:15:00
TPL     6P12.0-22:15:00
TPR     2Range N°2
TPL     7P12.0-00:00:00
TPL     8P15.5-02:45:00
TPL     9P12.0-04:45:00
TPL     AP26.0-08:30:00
TPL     BP12.0-10:30:00
TPL     CP30.0-16:30:00
TPL     DP12.0-18:30:00
STA     1STU BRU[FM=64-1770]
STA     2PURE FM[FM=60-1770]
CAM     1Axis Cam01[AXIS][IP=192.168.0.2]
CAM     2Cam
DVIP01[DVIP][DHCP][IP=192.168.0.3][JPG=http://192.168.0.3:80/jpg/image.jpg
]
CAM     3Cam
DVIP02[DVIP][IP=192.168.0.4][JPG=http://192.168.0.4:80/jpg/image.jpg]
CAM     4[CUST][IP=192.168.1.250]
[MJPEG=rtsp://admin:azerty@192.168.1.250:554/streaming/channels/102/]
END APPINFO - Send "HELP" from ETH.
Datasheet @ www.domintell.com => Pro - support@domintell.com
```

4.8. New generation LightProtocol

4.8.a) Overview

These commands/strings can be sent to Domintell2 system and are executed without doing any links (Automatic LightProtocol).
All newly released modules will follow the "*new generation input LightProtocol*". This new version is needed to handle these new complex modules that have several types of input and/or output (e.g. DGQG02 has TOR outputs, TRV output, 0-10V outpus, ...) that can not be handled with the legacy LightProtocol.
Frame generator of LightProtocol server for new generation modules has been completely reviewed to be parsed more easily.

4.8.b) Modules using new generation input LightProtocol

- DALI04
- DALI05
- DDIMLV01
- DDMX02
- DELEC01
- DENV01
- DENV02
- DELEC01
- DGQG02
- DGQG04
- DGQG05
- DMONOELEC01
- DMOV06
- DMOV07
- DMR02
- DOORSTATION
- DPBRTHERM01
- DRGBW01
- DTRIELEC01
- DTSC05
- DWIND01
- Air-conditioner and generic ModBus module (connected to DINTMB02 or DNET02)

4.8.c) General new generation frame format

All numbers are now expressed, by default, **in decimal** (even the serial number). If you want to use hexadecimal notation, you need to prefix value with "0x". So set an ouput 50% you can use `50` or `0x32`.

```
<Module type (ASCII)>/<serial number (int)>/<data frame>
```

- `<Module type>` : Module identifier. See 4.3 Abreviation of Modules' type for more informations.
- `<serial number>` : is expressed in declmal.
- `<data frame>` can contain following special characters:
  - `/` (slash) : field separator.
  - `#` (number sign) : input/output separator.
  - `|` (pipe) : parameter separator.

4.8.d) APPINFO frame format (*LightProtocol server* to *LightProtocol client*)

This chapter only covers lines generated for new generation modules. Please first read the chapter for the legacy APPINFO (4.7.d Decoding APPINFO).
See 4.8.c General new generation frame format for the start of the frame.

```
<Module type (ASCII)>/<serial number (int)>/<IO type (int)>/<IO offset (int)>/<IO name (UTF-8 string)>/<module version (int or decimal dotted number)>/<house location (UTF-8 string)>/<extra IO informations>
```

- `<Module type>` : Module identifier. See 4.3 Abreviation of Modules' type for more informations.
- *`<serial number>`* : is expressed in declmal.
- *`<IO type>`* : can be one of the following given in section 4.8.k IO type list, status and data format.
- *`<IO offset>`* : 1-based offset for the given IO type.
- *`<IO name>`* : name of the IO.
- *`<module version>`* : Version of the module in dotted decimal string.
- *`<house location>`* : location in the house. Room/floor are seperated by pipes.
- *`<extra IO informations>`* : For *TypeInputIo(2)* IO type, it will be type of push configured or if there are links associated to this input. See 4.8.e Extra informations in APPINFO.

4.8.e) Extra informations in APPINFO

- **Description of IOs of ModBus devices** ("MBG", "MBA") **may change in the future** to allow to group basic IOs (TypeTorIo, TypeSensorIo, TypeFanIo, TypePercentInIo, ...) in a special popup (e.g. VMC, heat-pump, EV charger, ...).

- **TypeInputIo** (2) : Push-button input
  - Format
    `/<type_of_link (int)>`
    - `<type_of_link (int)>` Integer value :
      - 0 = No link
      - 1 = Short push only
      - 2 = Short and long push

  - Example
    `LT5/17/2/3/Mini-SMD/11/[Domintell|Atelier|]/2`
    - `LT5` : DTSC05
    - `17` : serial number 17 (0x11)
    - `2` : this is a push-button (TypeInputIo)
    - `3` : this is the third push-button of the device
    - `Mini-SMD` : name of the push-button
    - `2` : Push-button with short and long push capabilities

- **TypeSensorIo** (8) : Temperature sensor
  - Format
    `/<regul_mask (hexint)>|<temp_mask (hexint)>|<heat_limit_high (float)>|<heat_limit_low (float)>|<cool_limit_high (float)>|<cool_limit_low (float)>|<setpoint_step (float)>`
    - `<regul_mask (hexint>` (integer value in hexadecimal format):

        - 0b00000001 = hide mode off.
        - 0b00000010 = hide mode heating.
        - 0b00000100 = hide mode cooling.
        - 0b00001000 = hide mode mixed.
        - 0b00010000 = hide mode auto hvac. (since v41.0)
        - 0b00100000 = hide mode dry. (since v41.0)
        - 0b01000000 = hide mode fan. (since v41.0)
    - `<temp_mask (hexint)>` (integer value in hexadecimal format) :
        - 0b0001 = hide mode auto.
        - 0b0010 = hide mode comfort.
        - 0b0100 = hide mode abscence.
        - 0b1000 = hide mode frost.
    - `<heat_limit_high (float)>` : setpoint limit for heating (floating point value with one decimal in Celsius degree).
    - `<heat_limit_low (float)>` : low setpoint limit for heating (floating point value with one decimal in Celsius degree).
    - `<cool_limit_high (float)>` : high setpoint limit for cooling (floating point value with one decimal in Celsius degree).
    - `<cool_limit_high (float)>` : low setpoint limit for cooling (floating point value with one decimal in Celsius degree)
    - `<inc_step (float)>` : setpoint increment (floating point value with one decimal in Celsius degree).

- Example

  `RT1/3/8/1/Sonde SMD/6/[Domintell|Atelier|SMD]/0xfc|0x0|30.0|10.0|40.0|20.0|0.5`

  - `RT1` : DPBRTHERM01
  - `3` : serial number 3 (0x03)
  - `8` : temperature sensor (TypeSensorIo)
  - `1` : first temperature sensor
  - `Sonde SMD` : name of the IO
  - `6` : version of the firmware
  - `[...]` : house location
  - `0xfc` : hide everything except mode off and heating
  - `0x0` : show all temperature mode
  - `30.0` : maximum heating temperature allowed. The LightProtocol client can not request a temperature higher than 30.0°C for this sensor. If higher value is requested it will be celled to this value.
  - `10.0` : minimum heating temperature allowed. The LightProtocol client can not request a temperature lower than 10.0°C for this sensor. If lower value is requested it will be floored to this value.
  - `40.0` : maximum cooling temperature allowed. The LightProtocol client can not request a temperature higher than 30.0°C for this sensor. If higher value is requested it will be celled to this value.
  - `20.0` : minimum cooling temperature allowed. The LightProtocol client can not request a temperature lower than 10.0°C for this sensor. If lower value is requested it will be floored to this value.

- `0.5` : prefered increment step for this sensor.

- **TypeFanIo** (13) : Fan
  - Format
    `/<number_of_speed (int)>|<has_off_speed (int)>|<has_auto_speed (int)>`
    - `<number_of_speed (int)>` : how much speeds has the IO (interger value). Integer value between 0 and 200.
    - `<has_off_speed (int)>` : if not null, the fan can be turned off. Otherwise, only a speed between 1 and `<number_of_speed (int)>` can be set. (interger value). Integer value : 0 or 1.
    - `<has_auto_speed (int)>` : if not null, the fan has an auto speed mode. The status of IO will be 254 if the device is in the auto speed mode and the speed is undefined. Integer value : 0 or 1.

- **TypeElecIo** (24) : Energy metering input
  - Format : `/<number_of_phases (int)>`
    `<number_of_phases (int)>` : number of phases/lines/channels handled by the meter. Integer value between 1 and 3.

- **TypeDmxIo** (25) : DMX output
  - Format : `/<number_of_channel (int)>|<dmx_type(int)`
    `<number_of_channel (int)>` : number of channels defined for this DMX slave/output.Integer value between 0 and 8.
    `<dmx_type (int)>` : defines the mapping of channels. Integer value :
    - 0 = misconfigured DMX slave/output. Please check configuration in GoldenGate !
    - 1 = RGB : first channel = red, second channel = green, third channel = blue.
    - 2 = RGBI : first channel = red, second channel = green, third channel = blue, fourth channel = intensity.
    - 3 = RGBW : first channel = red, second channel = green, third channel = blue, fourth channel = white
    - 4 = single channel (of any color)
    - 5 = multiple channels (of any color)
    - 6 = RGBWI : first channel = red, second channel = green, third channel = blue, fourth channel = white, fifth channel = intensity

- **TypeTorBasicTempoIo** (52) : Relay
  - Format : No extra informations
    Note : This is a relay that is temporized but there is no way for Domintell system to know about its current state. Should be considered as a write-only output

- **TypeInputTriggerIo** (53) : Push-button input
  - Format : `/<type_of_link (int)>`
    `<type_of_link (int)>`: Integer value
    - 0 = No link
    - 1 = Short push only

Note : This is an input that just send a short pulse to Domintell system. The feedback from the Domintell system to LightProtocol will be a start of a short push immediately followed by an end of a short push.

- **TypeSwingIo** (54) : Vanes/Swing
  - Format : `/<number_of_position (int)>|<has_auto_mode (int)>|<has_frozen_mode (int)>|<has_swing_mode (int)>`

    `<number_of_position (int)>` : how much positions has the IO. Interger value between 0 and 200.

    `<has_auto_mode (int)>` : if not zero, the vanes has an auto position mode. The status of IO will be 252 if the device is in the auto position mode and the position is undefined. Interger value : 0 or 1.

    `<has_frozen_mode (int)>` : if not zero, the vanes can be frozen to their current position. The status of IO will be 253 if the device is in frozen mode and the position is undefined. Integer value : 0 or 1.

    `<has_swing_mode (int)>` : if not zero, the vanes can swing. The status of IO will be 254 if the device is swinging and the position is undefined. Integer value : 0 or 1.

4.8.f) Example of APPINFO frame

- `QG2/12/2/1/Hall lights/1.8.0/[Ground floor|Hall]/0`
  - Module: DGQG02
  - Decimal serial number: 12
  - Type: input/push button (2 = TypeInputIo)
  - Input index: 1
  - Name: "Hall lights"
  - Version of DGQG02: 1.8.0
  - Location: at "Ground Floor" in "Hall room"
  - No link programmed (0)
- `QG2/12/2/2/Living lights/1.8.0/[Ground floor|Living]/2`
  - Module: DGQG02
  - Decimal serial number: 12
  - Type: input/push button (2 = TypeInputIo)
  - Input index: 2
  - Name: "Living lights"
  - Version of DGQG02: 1.8.0
  - Location: at "Ground Floor" in "Living room"
  - Long and short pushes are enabled (links programmed) (2)
- `PS4/2/51/1/DALI04/3/[House||]`
  - Module: DALI04
  - Decimal serial number: 2
  - Type: Power supply output (51 = TypePowerSupplyIo)
  - Output index: 1
  - Name: "DALI04"
  - Version of DALI04: 3
  - Location: not placed in a floor/room
- `LT5/16/8/1/T° Sensor DTSC05/7.0.0/[House||]/1|8|30.0|15.5|41.3|22.0|0.5`
  - Module: DTSC05
  - Decimal serial number: 16

    - Type: Temperature sensor (8 = TypeSensorIo)
    - Sensor index: 1
    - Name: "T° Sensor DTSC05"
    - Version of DTSC05: 7.0.0
    - Location: not placed in a floor/room
    - Hide temperature mode: "off" (mask 0x01)
    - Hide regulation mode: "frost" (mask 0x08)
    - Heating setpoint high limit: 30.0°C
    - Heating setpoint low limit: 15.5°C
    - Cooling setpoint high limit: 41.3°C
    - Cooling setpoint low limit: 22.0°C
    - Setpoint increment step: 0.5°C
- `EL1/12/24/1/Sensor DELEC01/1.0.0/[House||]/3`
    - Module: DELEC01
    - Decimal serial number: 12
    - Type: Energy metering input (24 = TypeElecIo)
    - Input index: 1
    - Name: "Sensor DELEC01"
    - Version of DELEC01: 1.0.0
    - Location: not placed in a floor/room
    - Three phases meter

4.8.g) Status frame format (*LightProtocol server* to *LightProtocol client*)

See 4.8.c General new generation frame format for the start of the frame.

```
<Module type>/<serial number>/<IO type>/<IO offset>/<data1>#<data2>#...
```

- `<Module type>` : Module identifier. See 4.3 Abreviation of Modules' type for more informations.
- `<serial number>` : a decimal integer value.
- `<IO type>` : can be one of the following given in section 4.8.k IO type list, status and data format.
- `<IO offset>` : 1-based offset for the given IO type (decimal integer value).
- `<datax>` : Status starting from IO with offset `<IO offset>`. If several statuses are provided (separated using the number sign (`#`)), IO offset should be increases each the time number sign (`#`) is decoded and associate the value following this number sign (`#`) to this "increased" IO offset. The format of the data depends on `<IO type>`, see section 4.8.k IO type list, status and data format for more informations.
- `/S` : This tag can be present at the end of a status frame. It means that the status has been explicitly requested by a LightProtocol client using `/103` command.

4.8.h) Examples of status frame

- `QG2/12/2/1/2#2#1#2#2#2#2#2#2#2#2#2`
  DGQG02, serial 12, TypeInputIo, from input 1 (IN1) to input 12 (IN12), all inputs are released except IN3 which is pressed.
  This kind if frame is received in reply to the PING command (5.9 Refresh statuses (PING)).

- `QG2/12/2/11/1`
  Status of inputs (2 = TypeInputIo) and starts from input 11.
  Input 11 has just changed and is pressed.
- `QG2/12/1/1/0`
  DGQG02, serial 12, TypeTorIo, output 1 (OUT1), off state
- `QG2/12/1/8/1`
  DGQG02, serial 12, TypeTorIo, output 8 (OUT8), on state
- `QG2/1/23/1/45#0`
  DGQG02, serial 12, TypeOut10VIo, from 0-10V output 1, « +1 » output = 0 % (0V) and « +2 » output = 45 % (4,5V)
- `QG2/1/23/2/0`
  DGQG02, serial 12, TypeOut10VIo, 0-10V output 2 (+2), 0 % (0V)
- `QG2/1/23/1/45`
  DGQG02, serial 12, TypeOut10VIo, 0-10V output 1 (+1), 45 % (=4,5V)
- `PS4/2/51/1/19|15.1|39`
  DALI04; serial 2, TypePowerSupplyIo, output 1, load = 19%, voltage = 15.1V, internal temperature = 39°C
- `MV6/3/34/1/1`
  DMOV06, serial 3, TypeMovIo, input 1, start detection
- `MV6/3/34/1/2`
  DMOV06, serial 3, TypeMovIo, input 1, end of detection
- `EV1/3/37/1/56.6`
  DENV01, serial 3, TypeHumidityIo, input 1, 56.6% RH
- `EV1/3/38/1/996.4`
  DENV01, serial 3, TypePressureIo, input 1, 996.4 hPa
- `EV1/3/36/1/1798`
  DENV01, serial 3, TypeLuxIo, input 1, 1798 lux
- `EV2/7/39/1/550.6`
  DENV02, serial 7, TypePressureIo, input 1, 550.6 hPa
- `EV2/7/37/1/54.2`
  DENV02, serial 7, TypeHumidityIo, input 1, 54.2% RH
- `EV2/7/8/1/22.1|24.0|AUTO|21.0|25.0|HEATING|27.0`
  DENV02, serial 7, TypeSensorIo, input 1,
  current temperature is 22.1°C,
  active heating setpoint is 24.0°C,
  current temperature mode is AUTO,
  profile heating setpoint is 21.0°C,
  active cooling setpoint is 25.0°C,
  current regulation mode is HEATING,
  profile cooling setpoint is 27.0°C

4.8.i) Command frame format (*LightProtocol client* to *LightProtocol server*)
See 4.8.c General new generation frame format for the start of the frame.

```
<Module type>/<serial number>/<IO type>/<IO offset>/<cmd1>[|<data1.1>|
<data1.2>|...]#<cmd2>[|<data2.1>|...]#...
```

- `<Module type>` : Module identifier. See 4.3 Abreviation of Modules' type for more informations.

- `<serial number>` : decimal integer value. From DAP version (`PROG M`) 43.7, the serial number can be in decimal or hexadecimal format
  - `LT5/17/...`
  - `LT5/0x11/...` (Only from `PROG M 43.7` or later)

- `<IO type>` : can be one of the following given in section 4.8.k IO type list, status and data format. Can be a decimal integer value or an hexadecimal value.
  - `LT5/17/2/1/...`
  - `LT5/17/0x02/1/...`

- `<IO offset>` : 1-based offset for the given IO type. Can be a decimal integer value or an hexadecimal value.
  - `LT5/17/2/1/...`
  - `LT5/17/2/0x01/...`

- `<cmdx>` : Command to perform starting from IO with offset `<IO offset>`. If several commands are provided (separated using the number sign (`#`)). The format of the data depends on `<IO type>` and `<cmd>`, see section 4.8.k IO type list, status and data format for more informations.

- `<datax.y>` : argument/extra data associated to the `<cmdx>`. Data are seperated from each others and its related command by a pipe (`|`). This field is optional.

4.8.j) List of available commands for outputs

All of these commands are not available for all type of outputs. See 4.8.k IO type list, status and data format for possible combination. **Please note that these commands are only valid for outputs. For inputs, please see 4.8.k IO type list, status and data format**

Command number	***Command name***	***Min. role***	***Description***
`/1`	Toggle	Admin	Toggle the state of the output to on or off depending of the current state.
`/2`	On	Admin	Turn the output on.
`/3`	Off	Admin	Turn the output off.
`/5`	Set value	Admin	Set output to the specified value.
`/10`	Move up	Admin	Move the shutter up.
`/11`	Move down	Admin	Move the shutter down.
`/16`	Increase speed Increase position	Admin	Increase the speed of a fan Select the next position of vanes.
`/17`	Decrease speed Decrease position	Admin	Decrease the speed of a fan Select the previous position of vanes.

`/71`	Set color	Admin	Set the color of a RGB output.
`/77`	Color cycle	Admin	Control the color cycle of a RGB output.
`/103`	Ask status	Viewer	Ask the status of all IOs of a module or of a specific IO of a module. When a status is explictly asked by a LightProtocol client, a `/S` tag is added at then end of the status string.
`/104`	Simulate push	Admin	Simulate a push on a push-button to execute associated links.

- Status command (`/103`)
  Please note that these are examples on how to request statuses to DGQG. To have real returned values for statuses, please refer to section 4.8.k IO type list, status and data format.

  - Status of one IO
    - Format : `<module id>/<sn>/<type>/<offset>/`**`103`**
      Example : `LT5/17/2/3/103`
      - Ask status of the third push-button of the DTSC05 n°17
      - Reply :
        `LT5/17/2/3/4/S`
        - `LT5` : DTSC05
        - `17` : serial number 17 (0x11)
        - `2` : push-button (TypeInputIo)
        - `3` : third push-button
        - `4` : Push-button has been long-pushed and is now released
        - `S` : means that this status was explicitly asked from a LightProtocol client.

  - Status of all IO of a given type
    Format : `<module id>/<sn>/<type>/`**`0/103`**
    Example : `LT5/17/2/`**`0/103`**
    - Ask status of all push-buttons of the DTSC05 n°17
    - Reply : `LT5/17/2/1/0#0#4#2/S`
      - `LT5` : DTSC05
      - `17` : serial number 17 (0x11)
      - `2` : push-button (TypeInputIo)
      - `1` : status are given starting of first push-button
      - `0#0#4#2` : first and second push-buttons have never been pushed since last restart of the system. The last action on third push-button was an end of long push. The last action on fourth button was an end of short push.

- `S` : means that this status was explicitly asked from a LightProtocol client.

- Status of all IO of the module
  Format : `<module id>/<sn>/0/0/103`
  Example : `LT5/17/0/0/103`
  Reply :
  - Before `PROG M 43.7`
    ```
    LT5/17/2/1/0#0#4#2
    LT5/17/10/1/0#0#0#0
    LT5/17/8/1/0.0|0.0|AUTO|0.0|0.0|HEATING|0.0
    LT5/17/49/1/5
    LT5/17/37/1/0.0
    LT5/17/31/1/1
    ```
  - From `PROG M 43.7`
    ```
    LT5/17/2/1/0#0#4#2/S
    LT5/17/10/1/0#0#0#0/S
    LT5/17/8/1/0.0|0.0|AUTO|0.0|0.0|HEATING|0.0/S
    LT5/17/49/1/5/S
    LT5/17/37/1/0.0/S
    LT5/17/31/1/1/S
    ```
  For legacy modules, use `%S` command (see 4.6.d Legacy action/command parameters)

- Status of all IO of all modules
  Use `PING` command (see 5.9 Refresh statuses (PING))

4.8.k) IO type list, status and data format

Decimal number used to define the category of the IO :

- **TypeTorIo** (1) : Relay/Bistable output
  - Status
    Format : `/<state (int)>`
    - `<state (int)>` : the current state of the relay/output with one of the following integer value :
      - 0 = Relay is off/open
      - 1 = Relay is on/closed

    Example
    - `MR2/10/1/5/1`
      - `MR2` : DMR02
      - `10` : serial number 10 (0x0A)
      - `1` : relay (TypeTorIo)
      - `5` : fifth output
      - `1` : relay is on/closed

  - Commands
    `/1`
    - Toggle (turn on if off or turn off if on)

    - Example
      - `MR2/10/1/5/1`
        `MR2` : DMR02
        `10` : serial number 10 (0x0A)
        `1` : relay (TypeTorIo)

`5` : fifth output
`1` : toggle the output
Reply : `MR2/10/1/5/1` (was previously `MR2/10/1/5/0`)
- (Data may differ. Please check status section above.)

`/2`
- Turn the output on

- Example
  - `MR2/10/1/5/2`
    `MR2` : DMR02
    `10` : serial number 10 (0x0A)
    `1` : relay (TypeTorIo)
    `5` : fifth output
    `2` : turn the output on
    Reply : `MR2/10/1/5/2`
    - If relay/output is already on, no feedback is returned !
    - (Data may differ. Please check status section above.)

`/3`
- Turn the output off

- Example
  - `MR2/10/1/5/3`
    `MR2` : DMR02
    `10` : serial number 10 (0x0A)
    `1` : relay (TypeTorIo)
    `5` : fifth output
    `2` : turn the output off
    Reply : `MR2/10/1/5/0`
    - If relay/output is already on, no feedback is returned !
    - (Data may differ. Please check status section above.)

`/103`
- Ask the status (see 4.8.j List of available commands for outputs

- **TypeInputIo** (2) : Push-button input
  - Status
    Format : `/<type of push (int)>`
    - `<type of push (int)>` : One of the following integer value to simulate a push :
      - 0 = Not pressed (this is an intermediate state that will only occur when the system has just restarted and until someone presses this button)
      - 1 = Start of short push (currently pressed)

- 2 = End of short push (currently released)
- 3 = Start of long push (currently pressed)
- 4 = End of long push (currently released)
- 5 = Pressed (this is an intermediate state that will only occur when the system has just restarted and until someone releases this button)

Example
- `LT5/17/2/3/3`
  - `LT5` : DTSC05
  - `17` : serial number 17 (0x11)
  - `2` : push-button (TypeInputIo)
  - `3` : third push-button
  - `3` : a long push is currently running

- Commands

`/<type_of_push (int)>` (deprecated, but kept for backward compatibility)

`/104|<type_of_push (int)>` (from `PROG M 43.7`)
- `<type_of_push (int)>`
  - 1 = Execute links associated to start of short push
  - 2 = Execute links associated to end of short push
  - 3 = Execute links associated to start of long push
  - 4 = Execute links associated to end of long push

- Example
  - `LT5/17/2/3/3`
  - `LT5/17/2/3/104|3` (from `PROG M 43.7`)

    `LT5` : DTSC05
    `17` : serial number 17 (0x11)
    `2` : push-button (TypeInputIo)
    `3` : third push-button
    (`104`) : Simul push command
    `3` : execute link for start of long push. **You need to send an other frame with `/4` command to simulate the release of the push-button !!!** Sending several consecutive `LT5/17/2/3/3` will have no effect until the end of push command is sent. This will also inhibit the first physical long on the button !
    Reply : `LT5/17/2/3/3`
    - (Data may differ. Please check status section above.)

`/103`
- Ask the status (see 4.8.j List of available commands for outputs

- **TypeDimmerIo** (*3*) : Dimmer output
  - Status

    Format : `/<dimmer value (int)>`
    - `<dimmer value (int)>` : 0-100 value

- Commands

`/1`

- Toggle (turn on if off or turn off if on)

- Example
  - `RW1/10/3/2/1`
    `RW1` : DRGBW01
    `10` : serial number 10 (0x0A)
    `3` : dimmer (TypeDimmerIo)
    `2` : second output
    `1` : toggle the output
    Reply : `RW1/10/3/2/100` (was previously `RW1/10/3/2/0`)
    - (Data may differ. Please check status section above.)

`/2`

- Turn the output on

- Example
  - `RW1/10/3/2/2`
    `RW1` : DRGBW01
    `10` : serial number 10 (0x0A)
    `3` : dimmer (TypeDimmerIo)
    `2` : second output
    `2` : turn the output on
    Reply : `RW1/10/3/2/100`
    - If output is already on, no feedback is returned !
    - (Data may differ. Please check status section above.)

`/3`

- Turn the output off

- Example
  - `RW1/10/3/2/3`
    `RW1` : DRGBW01
    `10` : serial number 10 (0x0A)
    `3` : dimmer (TypeDimmerIo)
    `2` : second output
    `3` : turn the output off
    Reply : `RW1/10/3/2/0`
    - If output is already off, no feedback is returned !
    - (Data may differ. Please check status section above.)

`/5|<value (int)`

- Set the output to the given value
- `<value (int)>`
  - 0 to 100 integer value.

- Example
  - `RW1/10/3/2/5|50`

`RW1` : DRGBW01
`10` : serial number 10 (0x0A)
`3` : dimmer (TypeDimmerIo)
`2` : second output
`5` : set output to the given value
`50` : set output to 50%
Reply : `RW1/10/3/2/50`

- If output is already at the asked value, no feedback is returned !
- (Data may differ. Please check status section above.)

`/103`

- Ask the status (see 4.8.j List of available commands for outputs

- **TypeTrvIo** (*6*) : Shutter output
  - Status
    Format : `/<state (int)>`
    - `/<state (int)>` : the state of the shutter
      - 0 = unknown state
      - 1 = stopped (last moving side unknown)
      - 2 = moving up
      - 3 = moving down
      - 4 = stopped and last move was up
      - 5 = stopped and last move was down

  - Possible command : 3, 10, 11, 103
    3 = stop shutter
    10 = move shutter up
    11 = move shutter down
    103 = ask the status

- **TypeSensorIo** (8) : Temperature sensor
  - Status
    Format
    - `<Meas. Temp (float)>|<Active Heat. SP (float)>|<Temp. Mode string>|<Heat Prof. SP (float)>|<Active Cool. SP (float)>|<Regul. Mode string>|<Cool Prof. SP (float)>`
      - `<Meas. temp (float)>` : the current room temperature in Celsius degrees.
      - `<Active Heat. SP (float)>` : the current heating set-point in Celsius degrees.
      - `<Temp. Mode string>` : the current temperature mode in ASCII string
        ABSENCE = away mode
        AUTO = automatic mode
        COMFORT = comfort mode
        FROST = no frost mode
      - `<Heat Prof. SP (float)>` : the heating set-point defined in the profile in Celsius degrees.

- `<Active Cool. SP (float)>` : the current cooling set-point in Celsius degrees.
- `<Regul. Mode string>` : the current regulation mode in ASCII string
  OFF = the regulation is disabled
  HEATING = the regulation is in heating mode
  COOLING = the regulation is in cooling mode
  MIXED = the regulation is in mixed mode (heating and cooling)
  AUTOHVAC = the regulation is in automatic mode (for HVAC only)
  DRY = the regulation is in dry mode (for HVAC only)
  FAN = the regulation is in fan mode (for HVAC only)
- `<Cool Prof. SP (float)>` : the cooling set-point as defined in the profile in Celsius degrees.

Example

- `RT1/3/8/1/20.5|22.0|AUTO|21.0|25.0|HEATING|27.0`
  - `RT1` : DPBRTHERM01
  - `3` : serial number 3 (0x03)
  - `8` : temperature sensor (TypeSensorIo)
  - `1` : first temperature sensor
  - `20.5` : the current room temperature is 20.5°C
  - `22.0` : the current heating set-point is 22.0°C
  - `AUTO` : the current temperature mode is automatic
  - `21.0` : the set-point from heating profile is 21.0°C. In this case, the user has manually increased the heating set-point by 1 degree.
  - `25.0` : the current cooling set-point is 25.0°C
  - `HEATING` : the current regulation mode is heating
  - `27.0` : the set-point from heating profile is 27.0°C. In this case, the user has manually decreased the cooling set-point by 2 degrees.

- Commands

`/1|<temp (float)`

- Set heating setpoint
  - `<temp (float)>` : the temperature to assign in Celsius degrees.

`/2|<temp (float)`

- Set cooling setpoint
  - `<temp (float)>` : the temperature to assign in Celsius degrees.

`/55|<mode (int)>`

- Set mode temperature
  - `<mode (int)>` : one of the following temperature mode (integer value)
    1 = absence
    2 = automatic
    5 = comfort

6 = no frost

`/82/<mode (int)>`

- Set mode regulation
  - `<mode (int)>` : one of the following regulation mode (integer value)
    0 = the regulation is disabled
    1 = the regulation is in heating mode
    2 = the regulation is in cooling mode
    3 = the regulation is in mixed mode (heating and cooling)
    5 = the regulation is in automatic mode (for HVAC only)
    6 = the regulation is in dry mode (for HVAC only)
    7 = the regulation is in fan mode (for HVAC only)

For example: 55|2 to set automatic mode

- **TypeLedIo** (*10*) : LED indicator
  - Unit : none
  - Possible status : 0-1
  - Possible command : 1-3, 103
    1 = toggle
    2 = on
    3 = off
    103 = ask the status

- **TypeFanIo** (*13*) : Fan
  - Only from `PROG M 42.5` and `PROG M 43.3` ! This type of IO must be ignored if `PROG M` is older.
  - Unit : none
  - Possible status :
    0-`<number_of_speeds>` (if `<has_off_speed>` is true in APPINFO)
    1-`<number_of_speeds>` (if `<has_off_speed>` is false in APPINFO)
    254 = automatic mode/speed (if `<has_auto_speed>` is true in APPINFO)
    255 = current mode/speed is undefined (statuses not received from the module/device)
  - Possible command : 3, 5, 16 and 17, 103
    3 = off (only if `<has_off_speed>` is true in APPINFO)
    5 = set speed to value given in `<data>`. From `0` or `1` to `<number_of_speeds>`; and `254` (if `<has_auto_speed>` is true in APPINFO)
    16 = increment speed
    17 = decrement speed
    103 = ask the status

- **TypeOut10VIo** (23) : 0-10V output
  - Unit : percent (%)
  - Possible status : 0-100
  - Possible command : 1, 2, 3, 5, 103
    1 = toggle

2 = on
3 = off
5 = set to value given in `<data>`.
103 = ask the status

- **TypeElecIo** (24) : Energy metering input
  - Only from `PROG M 42.9` and `PROG M 43.4` ! This type of IO must be ignored if `PROG M` is older.
  - Unit : none
  - Possible status : contains several data seperated by pipe '|' !

`<feature_flags>|<frequency>|<power_factor_l1>|<power_factor_l2>|<power_factor_l3>|<voltage_l1>|<voltage_l2>|<voltage_l3>|<intensity_l1>|<intensity_l2>|<intensity_l3>|<instant_power_l1>|<instant_power_l2>|<instant_power_l3>|<consumed_power>|<produced_power>|<total_power>|<total_energy_l1>|<total_energy_l2>|<total_energy_l3>|<forward_energy>|<reverse_energy>|<total_energy>|<total_energy_for_t1>|<total_energy_for_t2>|<total_energy_for_t3>|<total_energy_for_t4>|<tariff_indicator>`

`<feature_flags>` : A 32 bits number with flags in hexadecimal representation to report which of the following data/fields are valid/can be displayed. The offset of each flags is the index of following features in the string Following mask are OR-ed :

- 0x00000001 : data/value in `<frequency>` fleld can be used
- 0x00000002 : data/value in `<power_factor_l1>` field can be used.
- ...
- Example:

`<frequency>` : Power grid frequency (floating point value in Hz).
`<power_factor_lx>` : ratio between real power and apparent power (floating point value will be between 0 and 1).
`<voltage_lx>` : Active power for line x where x is the line number between 1 and 3 (integer value is given in V).
`<intensity_lx>` : Instant intensity for line x where x is the line number between 1 and 3 (integer value is given in mA).
`<instant_power_lx>` : Active power for line x where x is the line number between 1 and 3 (integer value is given in W). If value is negative it means that there is more production than consumption on that line (energy is sent back on grid).
`<total_energy_lx>` : Total energy for line x where x is the line number between 1 and 3 (integer value is given in Wh).
`<consumed_power>` : Instant consumed power (integer value is given in W).
`<produced_power>` : Instant produced power (vnteger ialue is given in W).
`<total_power>` : Difference between consumed power and produced power (integer value is given in W). If value is negative it means that there is more production than consumption on that line (energy is sent back on grid).
`<forward_energy>` : total energy consumed from grid for all tariff’s (integer value is given in Wh).
`<reverse_energy>` : total energy returned to the grid for all tariff’s (integer value is given in Wh).

`<total_energy>` : Difference between forward and reverse energy (integer value is given in Wh).
`<tariff_indicator>` : defines the current tariff active (integer value will always be between 1 and 4).
`<total_energy_for_t1>` : total energy for tariff 1 (integer value is given in Wh). If value is negative this means that there is more produced energy than consumed energy for this tariff.
`<total_energy_for_t2>` : total energy for tariff 2 (integer value is given in Wh). If value is negative this means that there is more produced energy than consumed energy for this tariff.
`<total_energy_for_t3>` : total energy for tariff 3 (integer value is given in Wh). If value is negative this means that there is more produced energy than consumed energy for this tariff.
`<total_energy_for_t4>` : total energy for tariff 4 (value is given in Wh). If value is negative this means that there is more produced energy than consumed energy for this tariff.

- Possible command : 103
  103 = ask the status

- **TypeDmxIo** (*25*) : DMX output
  - Unit : none
  - Channel seperator : `|`
    Example : `DX2/20/25/2/71|0x06|0|56|55` : values of channels of the second slave of DDMX02 n°20 are red = 0, green = 56 and blue = 55
    From version 39.1, an extra channel value is set for RGB/RGBI/RGBW/... slaves when the color cycle is running. Please note that value of RGB channels are not relevant when color cycle s running. So for a RGBW slave, if color cycle is not running `255|0|255|200` (cyan + white) is received. if color cycle is running, `255|0|255|200|204` (RGB color was cyan before running color cycle but is no more relevant, white + color cycle) is received.
    Example: `DX2/20/25/2/255|0|255|200|204`
  - Possible status : 0-255.
  - Possible command : 1, 2, 3, 71, 77, 103
    1 = toggle
    2 = on
    3 = off
    71 = set to value according informations given in `<data>` with `<data>` = `<mask>|<value1>|<value2>|...`
    - mask : `value1` is linked to bit 0 of `mask`, `value2` is linked to bit 1 of `mask`, ...
      If highest bit set in mask is the bit 5, you need to provide 6 values even if some bits between bit 5 and bit 0 are null. So if you want to set (virtual) RGB intensity channel and white channel of a RGBW slave to 200 and 100 respectively, data will look like `0x18|0|0|0|100|200`.
      - 0x07: set all RGB channels of RGB, RGBI and RGBW slaves. 3 values are needed. `value1` = Red channel, `value3` = Blue channel. Example : `0x07|100|255|100`.

    - 0x08: set Intensity channel of RGB (virtual) and RGBI. 4 values are needed. `value4` will be used. `value1` to `value3` will be ignored. Example `0x08|0|0|0|255`.
    - 0x10: set (virtual) RGB intensity channel of a RGBW slave. Only value5 will be used. Example `0x10|0|0|0|0|255`.

  77 = Start, stop or toggle color cycle according optional informations given in `<data>` with `<data>` = `<enable>`
  - enable = 1 will enable the color cycle, 0 will stop it. If `<data>` is omitted, a toggle will be performed. This command will only work on RGB slave (cf 4.8.e Extra informations in APPINFO). This command is only available from "`PROG M 39.1`" (cf 4.7.d Decoding APPINFO)

  103 = ask the status

- **TypeVideoIo** (*31*) : Video output
  - Unit : none
  - Possible status : Combination of following flags
    0x01 = Video output is online and is ready to display video stream
    0x02 = Incomming call running (bell)
    0x04 = Call has been caught. Bi-directional connection established with doorstation.
    0x08 = For half-duplex audio comminucation only : sound from microphone of the screen is sent to doorstation.
    0x10 = A video stream is currently playing on video output
    0x20 = The currpenly played video stream has been started by user directly from screen (not started due to a call)
  - Possible command : 103
    103 = ask the status

- **TypeMovIo** (*34*) : Motion detector input
  - Unit : none
  - Possible status : 0-2
    0 = Unknown state
    1 = Start of detection (or move in progress)
    2 = End of detection (no move detected anymore)
  - Possible command : 1-2, 103
    1 = Execute links associated to start of detection
    2 = Execute links associated to end of detection
    103 = ask the status

- **TypeLuxIo** (*36*) : Light sensor input
  - Unit : lux (lux)
  - Possible status : 0-16.000 (can be higher)
  - Possible command : 103
    103 = ask the status

- **TypeHumidityIo** (*37*) : Humidty sensor input
  - Unit : percent (% RH)
  - Possible status : 0-100

- Possible command : 103
  103 = ask the status

- **TypePressureIo** (*38*) : Atmospheric pressure sensor input
  - Unit : hectopascal (hPa)
  - Possible status : 300-1100
  - Possible command : 103
    103 = ask the status

- **TypeCo2Io** (*39*) : Carbon dioxide sensor input
  - Unit : parts-per-million (ppm)
  - Possible status : 0-40000
  - Possible command : 103
    103 = ask the status

- **TypeWindIo** (*41*) : Wind speed/direction sensor input
  - Unit : kilometer-per-hour (km/h)
  - Possible status : contains several data seperated by pipe '|' ! `<wind speed>|<wind direction>`
    Wind speed : 0 to 200 km/h
    Wind direction : N, NNE, NE, ENE, E, ESE, SE, SSE, S, SSW, SW, WSW, W, WNW, NW, NNW (with 'N' meaning "North", 'E' meaning "East", 'S' meaning "South" and 'W' meaning "West".
    For example: `10.0|NE` meaning Noth-East wind with a speed of 10.0 km/h.
  - Possible command : 103
    103 = ask the status

- **TypeLbIo** (*42*) : LightBus dimmer output
  - Unit : percent (%)
  - Possible status : 0-100
  - Possible command : 1, 2, 3, 5, 103
    1 = toggle
    2 = on
    3 = off
    5 = set to value given in `<data>`.
    103 = ask the status

- **TypeRgbwIo** (*46*) : RGBW output
  - Unit : none
  - Channel seperator : `|`
    Example : `RW1/20/46/1/0|56|55|200` : values of channels of the RGBW output n°20 are red = 0, green = 56, blue = 55 and white = 200.
    From version `PROG M 41.7.1`, an extra channel value is set when the color cycle is running. Please note that value of RGB channels are not relevant when color cycle s running. So for a RGBW slave, if color cycle is not running `255|0|255|200` (cyan + white) is received. if color cycle is running, `255|0|255|200|204` (RGB color - was cyan before running color cycle but is no more relevant -, white + color cycle) is received.
  - Possible status : 0-255.

- Possible command : 1, 2, 3, 71, 77, 103
  1 = toggle
  2 = on
  3 = off
  71 = set to value according information given in `<data>` with `<data>` = `<mask>|<value1>|<value2>|...` **!!! All values must be in decimal format !!!**
  - mask : `value1` is linked to bit 0 of `mask`, `value2` is linked to bit 1 of `mask`, ...
    If highest bit set in mask is the bit 5, you need to provide 6 values even if some bits between bit 5 and bit 0 are null. So if you want to set (virtual) RGB intensity channel and white channel of a RGBW slave to 200 and 100 respectively, data will look like `24|0|0|0|100|200`.
    - 0x07: set all RGB channels. 3 values are needed. `value1` = Red channel, `value3` = Blue channel. Example : `7|100|255|100`.
    - 0x08: set virtual Intensity channel of RGB. 5 values are needed. `value5` will be used. `value1` to `value4` will be ignored. Example `16|0|0|0|0|255`.
    - 0x10: set white channel. 4 values are needed. `value4` will be used. `value1` to `value3` will be ignored. Example `16|0|0|0|255`.

  77 = Start, stop or toggle color cycle according optional informations given in `<data>` with `<data>` = `<enable>`
  - enable = 1 will enable the color cycle, 0 will stop it. If `<data>` is omitted, a toggle will be performed. This command is only available from "`PROG M 39.1`" (cf 4.7.d Decoding APPINFO)

  103 = ask the status

- **TypeGestureIo** (*49*) : Gesture input
  - Status
    Format : `/<gesture (int)>`
    - `<gesture (int)>`
      - 0 = last gesture is unknown.
      - 4 = an up gesture as just been performed or last gesture was up.
      - 5 = a down gesture as just been performed or last gesture was down.

  - Commands
    `/<gesture (int)>` (deprecated, only for backward compatibility)
    `/105/<gesture (int)>` (from `PROG M 43.7`)
    - `<gesture (int)>`
      - 4 = execute links associated to up gesture.
      - 5 = execute links associated to down gesture.

    `/103`
    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypePowerSupplyIo** (*51*) : Domintell bus power supply
  - Status

    Format : `/<percentage_of_load (int)>|<output_voltage (float)>|<internal_temperature (int)>`
    - `<percentage_of_load i(nt)>` : integer value. How much power is delivered to installations.
    - `<output_voltage (float)>` : floating point value. The Voltage at the output of the power supply.
    - `<internal_temperature (int)>` : integer value in Celsius degrees. The temperatur inside the power supply.

    Example
    - `PS5/32/51/1/53|14.7|40`.
    - `PS5` : DALI05
    - `32` : serial number 32 (0x20)
    - `51` : power supply (TypePowerSupplyIo)
    - `1` : first output
    - `53` : Load of 53%
    - `14.7` : 14.7 volts
    - `40` : 40°C

  - Commands

    /103
    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeTorBasicTempoIo** (52) : Relay
  - Status

    Format : `/<state (int)>`
    - `<state (int)>` : most probably always 0.

  - Commands

    `/1`
    - Turn output on. The output will goes off by itself.

    `/2`
    - Turn output on. The output will goes off by itself.

    `/103`
    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeInputTriggerIo** (53) : Push-button input
  - Status

    Format : `/<push state (int)>`
    - `<push state (int)>`
      - 0 = Unknown state
      - 1 = Start of short push (currently pressed)
      - 2 = End of short push (currently released)

  - Commands

`/<type of push (int)>` (deprecated, only for backwarad compatibility)
`/104/<type of push (int)>` (from `PROG M 43.7`)
- `<type of push (int)>`
  - 1 = Execute links associated to start of short push.
  - 2 = Execute links associated to end of short push.

`/103`
- Ask the status (see 4.8.j List of available commands for outputs.

- **TypeSwingIo** (*54*) : Vanes/Swing
  - From `PROG M 42.5` and `PROG M 43.3`. This type of IO must be ignored if `PROG M` is older.

  - Status
    Format : `/<value (int)>`
    - From 0 to (`<number_of_positions>` - 1)
    - 252 = automatic position (only possible if `<has_auto_mode>` is true in APPINFO).
    - 253 = frozen position (only possible if `<has_frozen_mode>` is true in APPINFO).
    - 254 = swing position (only possible if `<has_swing_mode>` is true in APPINFO).
    - 255 = current position is undefined (statuses not received from the module/device).

  - Commands
    `/5/<position (int)>`
    - Set the position to value given in `<position (int)>`. From `1` to `<number_of_positions>`; and `252` (auto position), `253` (freeze at current position) or `254` (to start swinging) if available.

    `/16`
    - Increment the position.

    `/17`
    - Decrement the position.

    `/103`
    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeDeviceStatus** (*55*) : Status/Information
  - From `PROG M 42.5` and `PROG M 43.3`. This type of IO must be ignored if `PROG M` is older.

  - Status
    Format : n*ot yet defined*

  - Commands
    `/103`

    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypePercentInIo** (*56*) : Percent Input
  - Only from `PROG M 42.5` and `PROG M 43.3` ! This type of IO must be ignored if `PROG M` is older.

  - Status
    Format : `/<value (int)>`
    - `<value (int)>` : integer value between 0 and 100.

  - Commands
    `/103`
    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeAnalogInIo** (*57*) : Analog Input
  - From `PROG M 42.5` and `PROG M 43.3`. This type of IO must be ignored if `PROG M` is older.

  - Status
    Format : `/<value (float)>`
    - `<value (float)>` : the analog value in float format.

  - Commands
    `/103`
    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeLedRgbIo** (*60*) : RGB LED indicator of rainbow push button
  - From `PROG M 43.6`.

  - Status : `/<follower_state (int)>|<red_value (int)>|<green_value (int)>|<blue_value (int)>` state of the color LED.
    `<follower_state (int)>` : 0 or 1 integer value.
    `<red_value (int)>` : 0-255 integer value of the red channel of the RGB LED.
    `<green_value (int)>` : 0-255 integer value of the green channel of the RGB LED.
    `<blue_value (int)>` : 0-255 integer value of the blue channel of the RGB LED.

  - Commands
    `/103`
    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeCloudInfo** (*62*) : Status of the connection to the Domintell cloud
  - From `PROG M 43.7`.

  - Status

Format : `/<allowed (int)>|<registered (int)>|<connected (int)>|<error code (int)>|<error desc. (UTF-8 string)`

- `<allowed (int)>` :
  - -1 = The connection to the cloud not set-up.
  - 0 = The connection to the cloud is forbidden.
  - 1 = The connection to the cloud is enabled.

- `<registered (int)>` :
  - 0 = The module is not registered to the cloud (see `<error desc.>` field.
  - 1 = The module is correctly registered to the cloud.

- `<connected (int)>` :
  - 0 = The module is not connected to the cloud.
  - 1 = The module is not connected to the cloud. Only one module will/can be connected to the cloud. All other modules must be allowed and registered to have cloud services working (Pilot2, Voice assistant, ...).

- `<error code (int)>` :
  - 0 = No error.
  - 1 = Authentication error.
  - 2 = Data error.
  - 3 = Network error.
  - 4 = WebSocket error.
  - 5 = Protocol error.
  - 6 = Version error.
  - 7 = Token error.
  - 8 = Url error.
  - 9 = File error.
  - 10 = Notification type error.
  - 11 = Unknown service error.
- `<error desc. (UTF-8 string)>` : a more descriptive message about the error code.

Example

- `LT5/17/62/1/1|0|0|1|Credentials are invalid`
  - `LT5` : DTSC05
  - `17` : serial number 17 (0x11)
  - `62` : Cloud status (TypeCloudInfo)
  - `1` : first (and only) cloud info
  - `1` : cloud is allowed
  - `0` : module is not registered
  - `0` : module is not connected
  - `1` : Authentication error encountered
  - `Credentials are invalid` : Credentials given by the module to the cloud server have been rejected.

- `NT2/7/62/1/1|1|1|0|`
  - `NT2` : DNET02
  - `7` : serial number 7 (0x7)
  - `62` : Cloud status (TypeCloudInfo)

        - `1` : first (and only) cloud info
        - `1` : cloud is allowed
        - `1` : module is registered
        - `1` : module is connected
        - `0` : no error
        - `<empty string>` : no description as there is no error.

    - `QG4/7/62/1/-1|0|0|0|`
        - `QG4` : DGQG04
        - `7` : serial number 7 (0x7)
        - `62` : Cloud status (TypeCloudInfo)
        - `1` : first (and only) cloud info
        - `-1` : cloud is not yet configured (and disabled)
        - `0` : module is not registered
        - `0` : module is not connected
        - `0` : no error
        - `<empty string>` : no description as there is no error.

    - `QG4/7/62/1/1|0|0|3|Socket operation timed out`
        - `QG4` : DGQG04
        - `7` : serial number 7 (0x7)
        - `62` : Cloud status (TypeCloudInfo)
        - `1` : first (and only) cloud info
        - `1` : cloud is allowed
        - `0` : module is not registered
        - `0` : module is not connected
        - `3` : network error
        - `Socket operation timed out` : the module can not most likely access to Internet. Bad gateway IP address, bad DNS IP address or not connected to the building router.

- Commands
  `/103`
    - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeMemoryInfo** (*64*) : Status of the memory of the module
    - This type is for debug only.

    - From `PROG M 43.7`.

    - Status
      Format : `/<total ram (int)>|<free ram (int)>|<uptime (int)>|<o.s. ram size (int)>|<o.s. data ram size (int)>`
        - `<total ram (int)>` : total RAM size in bytes (integer value).

        - `<free ram (int)>` : free RAM size in bytes (integer value).

        - `<uptime (int)>` : number of minutes since the module booted up (integer value).

- `<o.s. ram size (int)>` : total RAM used by O.S. in bytes including application data (integer value).

- `<o.s. data ram size (int)>` : total RAM used by the application (DAP) loaded by the O.S. (integer value).

Example
- `QG4/7/64/1/119836672|22700032|323|102686720|62681088`
  - `QG4` : DGQG04
  - `7` : serial number 7 (0x07)
  - `64` : Memory status (TypeMemoryInfo)
  - `1` : first (and only) cloud info
  - `119836672` : there are 119,836,672 bytes of memory (114 MB)
  - `22700032` : there are 22,700,032 bytes of free memory
  - `323` : module is running since 323 minutes
  - `102686720` : the O.S. uses 102,686,720 bytes of memory (including the application/DAP)
  - `62681088` : the application (DAP) uses 62,681,088 bytes of memory

- Commands
  `/103`
  - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeStorageInfo** (*65*) : Status of the memory of the module
  - This type is for debug only.

  - From `PROG M 43.7`.

  - Status
    Format : `/<total size (int)>|<SLC wearout (int)>|<MLC wearout (int)>|<nbr part (int)>|<name part1 (int)>|<size part1 (int)>|<free space part1 (int)>|...|<name partN (UTF-8 string)>|<size partN (int)>|<free space partN (int)>`
    - `<total size (int)>` : total size of the eMMC in bytes (integer value).

    - `<SLC wearout (int)>` : the percentage of the wearout of SLC cells of the eMMC (integer value). Low value is better.

    - `<MLC wearout (int)>` : the percentage of the wearout of MLC cells of the eMMC (integer value). Low value is better.

    - `<nbr part (int)>` : the number of partitions on the eMMC (integer value).

    - `<name partN (int)>` : the name of the nth partiton (UTF-8 string).

- `<size partN (int)>` : the total size of the nth partiton in bytes (integer value).

- `<free space partN (int)>` : the free space available on the nth partiton in bytes (integer value).

Example
- `QG4/7/65/1/7820083200|10|10|3|_dev_root|1032134656|325826560|_dev_mmcblk0p2|1204944896|1123409920|_dev_mmcblk0p3|5334687744|4941975552`
  - `QG4` : DGQG04
  - `7` : serial number 7 (0x07)
  - `65` : Storage status (TypeStorageInfo)
  - `1` : first storage info
  - `7820083200` : there are 7,820,083,200 bytes of memory (7.28 GB)
  - `10` : the wearout of SLC cells is 10% (good health)
  - `10` : the wearout of MLC cells is 10% (good health)
  - `3` : there are three partitions
  - `_dev_root` : name of the first paritition
  - `1032134656` : the first partition has a size of 1,032,134,656 bytes (984 MB).
  - `325826560` : the first partition has 325,826,560 bytes of free space (311 MB)
  - `_dev_mmcblk0p2` : name of the second partition
  - ...

- Commands
  `/103`
  - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeCpuInfo** (*66*) : Status of the CPU of the module
  - This type is for debug only.

  - From `PROG M 43.7`.

  - Status
    Format : `/<nbr of CPU (int)>|<Avg load 1min (int)>|<Avg load 5min (int)>|<Avg load 15min (int)>`
    - `<nbr of CPU (int)>` : the number of CPU in the module (integer value).

    - `<Avg load 1min (int)>` : the average load for the last minute (integer value). 0 = idle; 100 = high load; >100 = very high load.

    - `<Avg load 5min (int)>` : the average load for the last 5 minutes (integer value). 0 = idle; 100 = high load; >100 = very high load.

- `<Avg load 15min (int)>` : the average load for the last 15 minutes (integer value). 0 = idle; 100 = high load; >100 = very high load.

Example
- `QG4/7/66/1/1|8|21|17`
  - `QG4` : DGQG04
  - `7` : serial number 7 (0x07)
  - `66` : CPU status (TypeCpuInfo)
  - `1` : first storage info
  - `1` : there is one CPU
  - `10` : the average load for the last minute is 8 (low)
  - `21` : the average load for the last 5 minutes is 21 (low)
  - `17` : the average load for the last 15 minutes is 17 (low)

- Commands
  `/103`
  - Ask the status (see 4.8.j List of available commands for outputs.

- **TypeDiBusGwInfo** (*67*) : Status of the Domintell bus gateway
  - This type is for debug only.
  - From `PROG M 43.7`.
  - Status
    Format : `/<version (int)>|<online (int)>|<state (int)>|<description (UTF-8 string)>`
    - `<version (int)>` : the version of the DI bus gateway (integer value).
    - `<online (int)>` : tells if messages from DI bus are forwarded to O.S. (integer value). 0 = messages are dropped; 1 = messages are forwarded.
    - `<state (int)>` : the state of the DI bus gatewate (integer value) :
      - 0 = unknown (not yet started)
      - 1 = missing. The gateway does not reply.
      - 2 = present. The gateway is ready to work.
      - 3 = updating. The firmware of the gateway is updating.
      - 4 = loader. The gateway is stuck in loader mode.
      - 5 = error. Some errors have been encountered with the gateway.
    - `<description (UTF-8 string)>` : A human readable string of the current state of the gateway.

    Example
    - `QG4/7/67/1/11|1|2|Ok`
      - `QG4` : DGQG04
      - `7` : serial number 7 (0x07)

- `67` : CPU status (TypeCpuInfo)
- `1` : first DI bus gateway info
- `11` : the version of the gateway is v11
- `10` : the gateway is online and forward frame from DI bus to the O.S.
- `2` : the gateway has been successfully configured (present)
- `Ok` : Everything is ok

- Commands
  `/103`
  - Ask the status (see 4.8.j List of available commands for outputs.

4.9. Custom input ASCII strings (sent to Domintell system).

Custom texts or frame from external devices/services are not handled.

4.10. Custom output ASCII strings (sent to your device).

Custom texts or frame to external devices/services are not handled.

5. How to use LightProtocol with your own application

5.1. Related modules

This chapter convers New Generation Ethernet modules like DNET01, DNET02, DGQG02, DGQG04, ... with LightProtocol mode enabled. For old DETH02 module, please read DS_RS232_ETH_Interfaces_v1_27_08.pdf.

5.2. Tools

New generation Ethernet Domintell modules now use Secured WebSocket (wss : data are encrypted in encapsulated frame over HTTP protocol). For now, there is only one out-of-box tool available under Linux : wscat.

5.2.a) wscat

```
$ wscat -n -c wss://<ip>:17481
> connected (press CTRL+C to quit)
< INFO:Waiting for LOGINPSW:INFO
> LOGINPSW@:
< INFO:Session opened:INFO
> APPINFO
< APPINFO (PROG M 30.9 06/02/17 09h19 Rev=13 CP=UTF8) =>
HOUSE_v300912_v2 :
NT1     1[VERS=0x04]Module DNET01[House||]
IS8   5B1-1BP sirène[House||][PUSH=SHORT]
IS8   5B1-2ISM8 2[House||][NOLINK]
IS8   5B1-3ISM8 3[House||][NOLINK]
IS8   5B1-4ISM8 4[House||][NOLINK]
IS8   5B1-5ISM8 5[House||][NOLINK]
IS8   5B1-6ISM8 6[House||][NOLINK]
IS8   5B1-7ISM8 7[House||][NOLINK]
IS8   5B1-8ISM8 8[House||][NOLINK]
END APPINFO
Datasheet @ www.domintell.com => Pro - support@domintell.com
> LOGOUT
< INFO:Closing session:INFO
disconnected
```

5.3. Password handling

Accounts must be first created using GoldenGate !

5.3.a) Password algorithm

The mechanism uses standard encryption algorithm. It uses salted password hashed. This hashed password is itself hashed with nonce generated by the Domintell module at socket creation. SHA-512 is used to hash data.

sha512(sha512(password + salt) + nonce)

5.3.b) Glossary

- *hash*: It is the result of a mathematical algorithm that maps data of arbitrary size to a bit string of a fixed size (a hash) and is designed to be a one-way function, that is, a function which is infeasible to invert.
- *SHA-512*: It is a mathematical algorithm that maps data of arbitrary size to a bit string of a fixed size of 64 bytes (a hash) and is designed to be a one-way function, that is, a function which is infeasible to invert.

- *salt*: random data that is used as an additional input to a one-way function that hashes a password or passphrase.
- *nonce*: an arbitrary number that can be used just once in a cryptographic communication.

5.4. Permissions/Roles

From `PROG M 43.7`, permissions/roles have been added to the user profile. When an account is created in GoldenGate, it is now possible limit his interaction with the Domintell system.

- *None* : the account is disabled and the user can not login.
- *Viewer (read-only)* : the user can only read the configuration and ask for status. It can not trigger/control any output or simulate any push.
- *Administrator* : the user has full access (just as before `PROG M 43.7`).

Please see 4.1 Reserved keywords, 4.6.d Legacy action/command parameters and 4.8.j List of available commands for outputs to have the required role to execute commands.

5.5. Open a session

Depending of the version of the Ethernet module, when WebSocket are used, Welcome message differs and tells which mechanism should be used:

- `INFO:Waiting for LOGINPSW:INFO`
  This is the first generation of login mechanism. There is no handling of user/password and the following command must be sent to open a session: `LOGINPSW@:`
- `INFO:Waiting for LOGINPSW:NONCE=19228322921537008311:INFO`
  This is the last generation of login mechanism that supports user/password. and the following procedure should be used to connect.

Consider we have the registered user "toto" with password "azerty". Here is the procedure to log in:

1. When a Secured WebSocket is opened, A text message is sent by Domintell module:

```
connected (press CTRL+C to quit)
< INFO:Waiting for LOGINPSW:NONCE=9301906811536867321:INFO
```

2. Request the salt associated to the user

```
> REQUESTSALT@toto
<
INFO:REQUESTSALT:USERNAME=toto:NONCE=9301906811536867321:SALT=1007182019:INFO
```

3. You have now all data needed to compute log in token. Salt the password by concatenating password to salt (<password> + <salt>) :

salted password = "*azerty1007182019*"

4. Hash the salted password :

SHA512("azerty1007182019") =
"df6b9fb15cfdbb7527be5a8a6e39f39e572c8ddb943fbc79a943438e9d3d85ebfc2ccf9e0eccd9346026c0b6876e0e01556fe56f135582c05fbdbb505d46755a"

Pay attention that the hash must be converted into an hexa human readable string before appending the nonce alphanumeric string.

5. Prepend the nonce to the hashed salted password (<hashed salted password> + <nonce>) :

"df6b9fb15cfdbb7527be5a8a6e39f39e572c8ddb943fbc79a943438e9d3d85ebfc2ccf9e0eccd9346026c0b6876e0e01556fe56f135582c05fbdbb505d46755a" + "93019068115368673 21"=
"df6b9fb15cfdbb7527be5a8a6e39f39e572c8ddb943fbc79a943438e9d3d85ebfc2ccf9e0eccd9346026c0b6876e0e01556fe56f135582c05fbdbb505d46755a93019068115368673 21"

5. Hash the nonce and the hashed salted password:

SHA512("930190681153686732 1df6b9fb15cfdbb7527be5a8a6e39f39e572c8ddb943fbc79a943438e9d3d85ebfc2ccf9e0eccd9346026c0b6876e0e01556fe56f135582c05fbdbb505d46755a") =
"c83b274c82c98965b558762e9f05e5556de17712c9c74ca372eb64523963ec729def36c384e7ef8a19a8c7373ac3cc4f10d567bc3b2110fec80c199fcea08b01"

6. Build LOGINPSW command:

```
>
LOGINPSW@toto:c83b274c82c98965b558762e9f05e5556de17712c9c74ca372eb64
523963ec729def36c384e7ef8a19a8c7373ac3cc4f10d567bc3b2110fec80c199fce
a08b01
< INFO:Session opened:INFO
> LOGOUT
< INFO:Closing session:INFO
disconnected
```

5.5.a) Error messages

- No user/account in database

```
connected (press CTRL+C to quit)
< ERROR:User database empty. Connect first with GoldenGate:ERROR
< INFO:Session timeout:INFO
disconnected
```

This is the default behaviour when the module is connected for the first time on a Domintell installation. No connection is allowed until a user account is created using the configuration software GoldenGate.

- Wrong credentials

This error occurs when username, password or token is invalid.

```
connected (press CTRL+C to quit)
< INFO:Waiting for LOGINPSW:NONCE=8425756741536921788:INFO
>
LOGINPSW@toto:c83b274c82c98965b558762e9f05e5556de17712c9c74ca372eb64
523963ec729def36c384e7ef8a19a8c7373ac3cc4f10d567bc3b2110fec80c199fce
a08b01
< ERROR:Invalid credentials:ERROR
disconnected
```

In this case, the credentials are rejected because the same token generated for account creation is reused while nonce has changed ! Invalid credentials

- Session not opened
  No commands are accepted until a session is opened.
  ```
  connected (press CTRL+C to quit)
  < INFO:Waiting for LOGINPSW:NONCE=10671983681536923762:INFO
  > PING
  < ERROR:Invalid command. Use REQUESTSALT@<username> and
  LOGINPSW@<username>:<hashedpsw>:ERROR
  disconnected
  ```
  PING has been sent while no session is opened.

- A read-only user try to send an action commands
  The user try to toggle an ouput or simulate a push while it only has read-only permission. Only from `PROG M 43.7`.
  ```
  > ERROR:Command denied. Read-only user:ERROR
  ```

5.6. Ask the version of the LightProtocol (`GETLPVER`)

From PROG M 43.7, a new command as been added. It can only be issued when a session is opened. It returns the current version of the LightProtocol (which is the same that the one provided by `PROG M` in `APPINFO`.

```
> GETLPVER
< INFO:LPVER=43.7.1:INFO
> APPINFO
< APPINFO (PROG M 43.7 00/00/00 00h00 Rev=1) => DOMINT_v02.dap :
< ...
< END APPINFO
```

5.7. Download list of modules (`APPINFO`)

To donwload list of modules, inputs, outputs, ..., `APPINFO` command must be invoked. A full description of the installation.

```
> APPINFO
< APPINFO (PROG M 38.0 00/00/00 00h00 Rev=0) => DOMINT_v02.dap :
< ET2     1[VERS=0x14]MOD DETH02[Maison||]
< BU6     1-1Input DPBU06 1[House||][PUSH=LONG]
< BU6     1-2Input DPBU06 2[House||][PUSH=LONG]
< BU6     1-3Input DPBU06 3[House||][PUSH=LONG]
< BU6     1-4Input DPBU06 4[House||][PUSH=LONG]
< BU6     1-5Input DPBU06 5[House||][PUSH=LONG]
< BU6     1-6Input DPBU06 6[House||][PUSH=LONG]
< BU6     1-7Output DPBU06 1[House||]
< BU6     1-8Output DPBU06 2[House||]
< BU6     1-9Output DPBU06 3[House||]
< BU6     1-AOutput DPBU06 4[House||]
< BU6     1-BOutput DPBU06 5[House||]
< BU6     1-COutput DPBU06 6[House||]
< DX2/20/25/1/DMX2 RGBW 1/0.0.0/[House||]/4|3
< …
< END APPINFO - Send "HELP" from ETH.
< Datasheet @ www.domintell.com => Pro - support@domintell.com
```

Once all informations have been collected and processed, commands related to each items can be sent to DGQG to control the installation. Please see chapters 4.7.d Decoding APPINFO (legacy) and 4.8.d APPINFO frame format (LightProtocol server to LightProtocol client) (NewGen) to known how to parse and decode `APPINFO`.

5.8. Keep session open (`HELLO`)

To keep session opened, you have to send one command (or LightProtocol string) to LightProtocol server. The best way is to use HELLO command and sent it each 50 seconds. PING command should be avoid to keep a session opened as it will generate a lot of trafic on Domintell Bus and takes ressources in Master (DGQG02/DGQG04/...).

```
> HELLO
< INFO:World:INFO
```

If HELLO command is not received by the DETH02 before its timeout expiration, the following text will be sent:

```
< INFO:Session timeout:INFO
```

A new command has been implemented to disable timeout : `TIMEOUT=0`. **Take care that session will be closed only when WebSocket will be closed**. In this case, HELLO command is no more needed.

```
> TIMEOUT=0
< INFO:Timeout disabled. Socket will never be closed unless you send
LOGOUT or the connection is lost !:INFO
```

5.9. Refresh statuses (`PING`)

`PING` command must be used carefully. Generally, use it after a LOGIN (if your application has already been configured using APPINFO).

```
> PING
< PONG
< QG2/12/1/1/0#0#0#0#0#0#0#1
QG2/12/23/1/45#0
QG2/12/2/1/2#2#2#2#2#2#2#2#2#2#4#2
QG2/12/6/1/5
BIR  101FO00
VAR     1O00
VAR     2D02
SYS     0O00
SYS     9O01
```

There is not string/flags to notify end of list of statuses.

Important note : only invoke `PING` command when socket has been reconnected. Do not call PING periodically or when you enter/exit a room/floor !

Important note 2 : On NewGen Masters (DGQG02/04/...), sent statuses are the ones stored in the RAM of the DGQG. A `PING` command will **not** ask current status to each modules! No request to modules themselves will be performed!

To request status of a specific module/IO, check the `%S` command for legacy modules (see 4.6.d Legacy action/command parameters) and the `/103` command for NewGen modules (see 4.8.k IO type list, status and data format).

5.10. Close session before exiting the application

If your application is closed or put in background, it is better to send the LOGOUT command to allow other applications/devices to use DGQG02/DGQG04/DNET01/...

```
> LOGOUT
< INFO:Session closed:INFO
```