



INTRODUCTION À LA GESTION TECHNIQUE DE BÂTIMENT (GTB)

TP 1

Séquence 1 : La GTB et ses protocoles

v2.0

IUT d'Annecy, 9 rue de l'Arc en Ciel, 74940 Annecy

METTRE EN OEUVRE LE PROTOCOLE DALI

Objectifs:

- Mettre en oeuvre une stratégie d'éclairage à l'aide du protocole DALI
- Comprendre l'architecture DALI

Dans le tertiaire, l'éclairage représente environ 40% de la consommation électrique. On estime que 30% de cette consommation est inutile. Il devient essentiel de mettre en place des stratégies d'éclairage intelligentes.

Le protocole DALI, par la finesse de son contrôle, permet de mettre en place des stratégies d'éclairage intelligentes en vue d'améliorer l'efficacité électrique des bâtiments tout en améliorant le confort des occupants.

Pour rappel, les caractéristiques principales du protocole DALI sont :

Tension 16 V	Câblage Paire différentielle	Nombre de groupes 16
Courant 250 mA	torsadée	Nombre de scènes 16
Longueur maximale 300 m	Topologie Bus, étoile ou	Stratégie Maître/esclave
Débit 1200 bauds	combinaison des deux	
Codage Manchester	Nombre de participants 64	

Pour chaque point d'éclairage, il est possible de définir les informations suivantes, qui seront mémorisées dans le ballast :

- Valeur minimum et maximum de la luminosité
- Temps de montée et de descente (*Fade in*, *Fade out*)
- Niveau de lumière à la mise sous tension
- Niveau de lumière en cas de coupure de liaison avec le contrôleur

Il existe un grand nombre d'instructions répertoriées dans la norme IEC 6092. Celles-ci permettent de contrôler les luminaires, de les regrouper, de les organiser en scènes, de les programmer, etc.

Une liste de ces commandes peut être trouvée en suivant le lien suivant : <https://onlinedocs.microchip.com/pr/GUID-0CDBB4BA-5972-4F58-98B2-3F0408F3E10B-en-US-1/index.html?GUID-DA5EBBA5-6A56-4135-AF78-FB1F780EF>

1 Présentation du TP

Ce TP portera sur la gestion d'un luminaire RGB DALI à partir d'un automate Wago.

1.1 Architecture du système

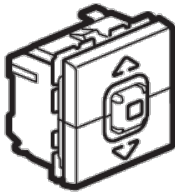
Afin de disposer du réseau DALI sur la structure de commande Wago, celle-ci doit être équipée, au minimum :

- D'un module de commande Wago.
 - ◊ 750-841 sans KNX
 - ◊ 750-849 avec KNX
- D'un module DALI Wago 750-653
- D'un convertisseur 24VDC/18VDC Wago (288-895) pour alimenter le bus DALI

La partie pupitre est composée de :

- Quatre boutons poussoirs (blanc, rouge, vert, bleu)
- Un sélecteur à 3 boutons poussoirs (haut, bas, central)
- Un bouton d'arrêt d'urgence
- Trois luminaires RGB DALI

Le **sélecteur à 3 boutons** fournit les informations **ixSelDown** et **ixSelUp** et son fonctionnement est décrit dans le tableau suivant :



Action sur le sélecteur	ixSelDown	ixSelUp
Aucun appui	false	false
Appui sur le bouton du haut	false	true
Appui sur le bouton du bas	true	false
Appui sur le bouton central	true	true

Préparation 1: Architecture du système

Question 1 Dessiner l'architecture matérielle faisant apparaître les éléments Wago ainsi que les boutons poussoirs et les trois luminaires Rouge, Vert et Bleu

D'un point de vue logiciel, on dispose de la bibliothèque *DALI_02.lib* qui offre plusieurs blocs fonctionnels permettant d'établir les différentes commandes du bus (commande d'éclairage, de configuration, etc ...) plus des commandes étendues.

1.2 Première communication

Préparation 2: Le bloc FbDALI_Joblist

Question 2 À l'aide de l'annexe A, page 10, expliquer le rôle de ce bloc fonction.

Question 3 Proposer une classe **CDaliJobList** pour éviter l'éparpillement des données.

Question 4 Dessiner le bloc fonction **FbDALI_Joblist** permettant l'envoi de commandes au coupleur DALI. Utiliser la structure de données **daliJobList**, instance de **CDaliJobList**. *On ne s'intéresse pas aux valeurs des variables.*

Cahier des charges *Plafonnier 1*

- Le plafonnier est commandé par le bouton blanc : chaque appui change son état (allumé ou éteint).
-

Préparation 3: Fonctionnement Télérupteur

Question 5 A l'aide de l'annexe B page 11, expliquer le rôle du bloc **FbDALI_LatchingRelay**.

Question 6 Dessiner ce bloc ainsi que les valeurs à placer en Entrée et en Sortie pour réaliser le cahier des charges ci-dessus.

Note : Certaines entrées peuvent être laissées vides.

1.3 Envoi de commandes avancées

Nous nous intéressons dans cette section au bloc **FbDALI_Master** (Annexe C, page 13).

Préparation 4: Classe Ecg pour FbDALI_Master

Question 7 À l'aide de l'annexe C, page 13, expliquer le fonctionnement de ce bloc.

Question 8 Proposer une structure de données **CEcg** qui permettra la gestion d'un ballast du réseau.

Question 9 Dessiner le bloc fonction **FbDALI_Master** permettant d'envoyer une commande au ballast Rouge en utilisant la structure de données **ecgRouge**, instance de **CEcg**.

Préparation 5: La fonction FuDimmValue

Question 10 À l'aide de l'annexe D, page 15, expliquer le fonctionnement de cette fonction.

On veut éclairer le ballast rouge à 50% de sa puissance maximale.

Question 11 Ecrire une ligne en ST qui utilise cette fonction pour préparer la commande à envoyer.

`ecgRouge.m_bCommandValue := ...`

2 Travail à réaliser



Variables d'entrées utiles et adresses DALI

Element	Nom de variable
Bouton arrêt urgence	ixArretUrgence
Bouton poussoir Blanc	ixBpAcq
Bouton poussoir Rouge	ixBpRouge
Bouton poussoir Vert	ixBpVert
Bouton poussoir Bleu	ixBpBleu
Sélecteur Haut	ixSelUp
Sélecteur Bas	ixSelDown

Element DALI	Adresse DALI
	1
Luminaires (de la fenêtre vers le couloir)	2
	3
	4
Lampe Rouge	10
Lampe Verte	11
Lampe Bleue	12

2.1 Observation matérielle

Activité 1: Elements du système

Question 12 Noter la référence de votre automate et repérer le coupleur DALI (750-641) sur votre maquette.

2.2 Configuration du projet

Activité 2: Préparation du projet

Étape 1 Lancer la machine virtuelle *ARS (Win10)*

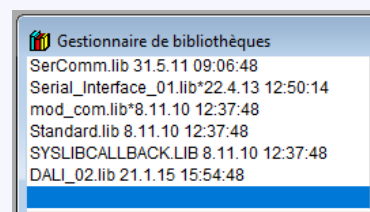
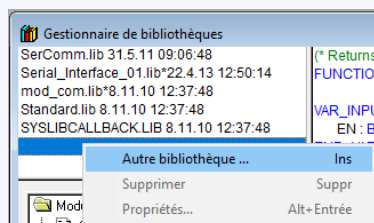
Étape 2 Pendant qu'elle démarre : Sous votre répertoire associé à ce module, copier le squelette de l'application **correspondant à l'automate de votre plateforme** qui se trouve sous

U : \Documents\BUT\GEII\ModuleS3\R3.13 Réseaux Spécialisés EME \Squelettes \Dali.

Étape 3 Lancer le logiciel **CoDeSys V2.3** sur la machine virtuelle et ouvrir le squelette de l'application.

Nous allons à présent inclure la bibliothèque DALI_02.lib dans notre projet afin de pouvoir utiliser les blocs fonctionnels nécessaires à la communication avec le bus DALI.

Étape 4 Dans l'onglet **Ressources**, ouvrir le **Gestionnaire de bibliothèques**. Faire un clic droit pour ajouter une bibliothèque et sélectionner **DALI_02.lib**. Elle se trouve dans le répertoire **C : \Program Files \WAGO Software \CoDeSys V2.3 \Targets \WAGO \Libraries \Building**



2.3 Premières commandes DALI

Dans cette section, nous allons mettre en place notre première communication DALI. Nous allons commander un plafonnier à partir d'un bouton poussoir en utilisant un bloc fonction de la bibliothèque DALI_02.lib.

Activité 3: Définition des types de données

Étape 5 Dans l'onglet **Type de données**, renseigner le type **CDaliJobList** (Préparation ??)

Activité 4: Programme bloc Fonction

Étape 6 Ajouter un nouveau module :

- Dans l'onglet **Modules**, ajouter un nouveau module (*Clique-droit -> Insérer objet*)
- Le nommer *Plafonniers* et langage CFC. L'appeler dans le programme DALI

Étape 7 Dans ce nouveau programme, déclarer la variable **jobList** de type **CDaliJobList** (*Tutoriel 1*)

- Donner la valeur par défaut **1** à la variable **m_bModule_750_641** (*Numéro du module DALI*)

Étape 8 Appeler (*Placer*) le bloc fonction **jobList.m_fbDALI_Joblist** (Préparation 2, Tutoriel 2).

Étape 9 Déclarer un Fonction bloc **FbDALI_LatchingRelay** et l'insérer. (*Prépa 3, Tuto 1, Tuto 2*) .

Étape 10 Faire vérifier et tester votre programme.




Tutoriel 1 Déclaration automatique dans CoDeSys

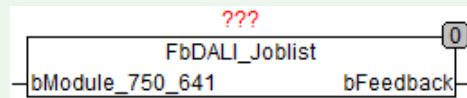
Dans le logiciel CoDeSys, il est possible de déclarer automatiquement des variables dont le type a été préalablement défini. Par exemple, pour déclarer une variable de type **CDaliJobList** :

1. Cliquez-droit sur une ligne vide
 - *Déclarer automatiquement*
2. Cliquer sur les ... de **Type** :
 - Type défini
 - Sélectionner le type (*CDaliJobList, par exemple*)
3. Cliquer sur les ... de **Valeur initiales** :
 - Définir les valeurs initiales pertinentes (*Celles qui ne seront pas impérativement définies dans le programme, par exemple.*)
4. Donner un nom à la variable dans le champs **Nom**

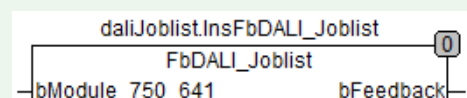
Tutoriel 2 Insertion d'un bloc fonction



Pour insérer un bloc fonction dans un programme CFC :

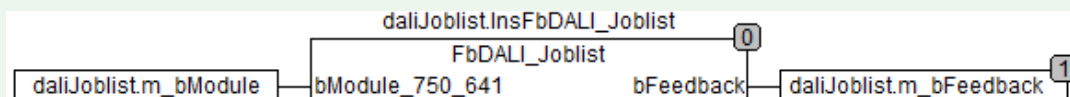
1. Cliquer sur l'icone Module 
2. Descendre la souris dans la zone de bloc
3. Renommer le bloc *AND* par le type de bloc que vous voulez insérer (*FbDALI_Joblist*, par exemple)



- A cette étape, le bloc est inséré, mais n'est pas encore appelé. Il faut lui associer une variable déclarée.
4. Double-cliquer sur les **???** et lui associer une variable existante.



5. Ajouter une entrée () et une sortie () puis leur associer une variable ou une valeur.



Activité 5: Appel d'un bloc en ST

Dans cette activité, nous allons réaliser la même chose mais en appelant les blocs fonction en langage ST. Cela nous permettra, par exemple, de créer plusieurs instances à l'aide d'une boucle *for* et de tableaux.

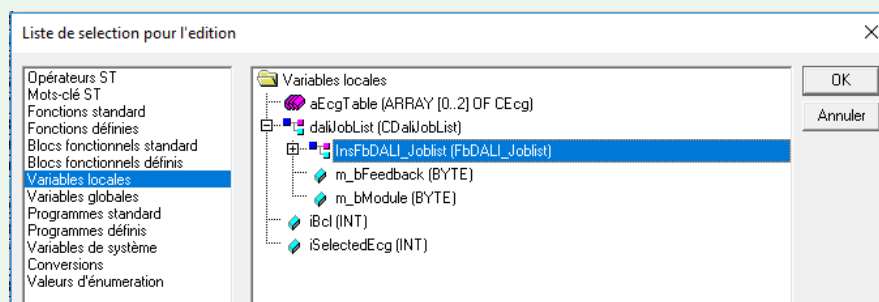
Étape 11 Dans le programme DALI :

- Déclarer une variable **bLatchingRelay** de type **FbDALI_LatchingRelay** (*Tutoriel 1*)
- Appeler le bloc fonction en ST (*Tutoriel 3*) et lui associer les valeurs d'entrées et sorties.
- Faire vérifier et tester votre programme.

Tutoriel 3 Aide à l'appel d'élément déclaré

Dans le logiciel CoDeSys, il est possible d'utiliser une variable définie ou d'appeler une fonction ou un bloc fonction déclaré facilement en suivant les étapes suivantes :

- Cliquez droit sur une ligne vide
 - ◊ *Liste de sélection pour l'édition*
- Chercher l'élément que vous souhaitez appeler



2.4 Envoi de commandes plus complexes

Activité 6: Mise en place de la communication avec le ballast Rouge

Étape 12 Définir la structure de données **CEcg** (Préparation 4)

Étape 13 Déclarer une variable **ecgRouge** de type **CEcg** (Tutoriel 1)

Étape 14 Insérer le bloc fonction **FbDALI_Master** associé. (*Tuto 3*) . et lui associer les membres de sa structure.



Remarque *Qu'avons-nous fait jusque là ?*

Les étapes précédentes ont permis de déclarer les variables et les blocs fonctionnels nécessaires à la mise en place d'un bus DALI.

Le bloc fonction **fbDALI_Master** scrute sa variable d'entrée **xStartDaliMaster** dans l'attente d'une mise à 1. Lorsque cela se produit, il exécute la commande DALI paramétrée sur ses autres entrées. Pour simplifier l'accès à ses variables d'entrées et de sortie, nous avons créé une classe **CEcg** et associé les attributs de cette classe aux entrées du bloc.

Le bloc fonction **fbDALI_Joblist** se chargera d'envoyer les commandes que les blocs **fbDALI_Master** auront généré. Pour simplifier l'accès à ses variables d'entrées et sorties, nous avons créé une classe **CDaliJobList** et associé les attributs de cette classe aux entrées du bloc.

Activité 7: Commande directe

Étape 15 Écrire un programme qui allume la lampe rouge à 75% de sa luminosité maximale lorsque le bouton rouge est appuyé et qui l'éteint lorsque le bouton est relâché. Faire vérifier par l'enseignant.



Attention

Si le programme précédent est fait trop simplement, des commandes DALI sont envoyées sans arrêt, saturant le réseau DALI. Il faut corriger cela en utilisant, par exemple, la détection de front montant.

Activité 8: Correction du programme précédent

Étape 16 Si nécessaire, corriger le programme précédent pour n'envoyer qu'une seule commande DALI à chaque changement d'état.

3 Programmes avancés

Un bouton par couleur

- Le bouton poussoir rouge commande la lampe rouge.
 - Le bouton poussoir vert commande la lampe verte.
 - Le bouton poussoir bleu commande la lampe bleue.
-

Cahier des charges : Intensité lumineuse

- Le sélecteur 3 position modifie l'intensité de la lampe rouge
-

Activité 9

Étape 17 Programmer le cahier des charges *Un bouton par couleur*. Tester le programme

Étape 18 Programmer le cahier des charges *Intensité lumineuse*. Tester le programme

Cahier des charges : Sélection de couleur

- Chaque bouton de couleur permet de sélectionner la couleur de la lampe à commander.
 - Le sélecteur à 3 boutons permet d'augmenter ou de diminuer l'intensité lumineuse de la lampe sélectionnée.
-

Activité 10

Pour coder le cahier des charges ci-dessus, on propose la stratégie suivante :

Étape 19 Supprimer les instances CEEg créer et les remplacer par un tableau de 3 instances de CEEg, un par couleur.

Étape 20 Rassembler les appels des blocs **FbDALI_Master** dans une boucle *for*. Pour cela, il faudra créer un tableau

Étape 21 Modifier le reste du programme pour respecter le Cahier des charges.

Étape 22 Tester et faire vérifier votre programme

Activité 11: Appeler l'enseignant

A DALI Job List

DALI Job List (FbDALI_Joblist)

WAGO-I/O PRO CAA Library Elements		
Category:	Building Automation	
Name:	FbDALI_Joblist	
Type:	Function <input type="checkbox"/>	Function block <input checked="" type="checkbox"/> Program <input type="checkbox"/>
Name of library:	DALI_02.lib	
Applicable to:	See release note	
Input parameter:	Data type:	Comment:
bModule_750_641	BYTE	Specifies which DALI master module is to be addressed at the controller. Counting is from left to right. Value range = 1 – 5 Default setting = 1
Output parameters:	Data type:	Comment:
bFeedback	BYTE	Response byte (see table 6 in the appendix)
Graphical display:		
<div><div>FbDALI_Joblist</div><div>bModule_750_641bFeedback</div></div>		
Function description:		
<p>The FbDALI_Joblist function block is used for communication with the DALI module 750-641 on the fieldbus controllers 750-8xx. This function block detects all queued commands of the other DALI function blocks in the program and causes their execution.</p> <p>The controller recognizes the plugged DALI modules on its own and counts them one after the other, starting from the left. To address the function block to the proper DALI module, the corresponding module index must be entered as a constant at the input "bModule_750_641".</p> <p>The output "bFeedback" outputs a numeric code with the response. The numeric codes are listed in table 6 in the appendix.</p>		
Note:		
<ul style="list-style-type: none">The function block sends the command "Terminate" (256) after the program start.The function block "FbDALI_Joblist" should be called in the program sequence before all other DALI function blocks.This function block may be used only once per installed DALI module.		

Latching Relay (FbDALI_LatchingRelay)

B

WAGO-I/O PRO CAA Library Elements			
Category:		Building Automation	
Name:		FbDALI_LatchingRelay	
Type:		Function <input type="checkbox"/>	Function block <input checked="" type="checkbox"/> Program <input type="checkbox"/>
Name of library:		DALI_02.lib	
Applicable to:		See release note	
Input parameter:		Data type:	Comment:
bAddress		BYTE	Short address of 1–64 or Group address 1–16 Broadcast = 255
xGroup		BOOL	Selects short or group address: FALSE = short address or broadcast TRUE = group address Default setting = FALSE
xButton		BOOL	Input from switch lighting request.
xOFF_as_MinLevel		BOOL	Instead of the switch-off command, the lighting is dimmed to the min. level. Default setting = FALSE
xCentr_OFF		BOOL	TRUE = group address
xCentr_ON		BOOL	Input for the central ON command.
bReferenceaddress1		BYTE	First reference control gear determines the current brightness value
bReferenceaddress2		BYTE	Second reference control gear determines the current brightness value
bModule_750_641		BYTE	Specifies which DALI master module is to be addressed at the controller. Counting is from left to right. Value range = 1 – 5 Default setting = 1
Feedback value:		Data type:	Comment:
bFeedback		BYTE	Response byte (see table 6 in the appendix)
Graphical display:			
<div><div>FbDALI_LatchingRelay</div><div><div>bAddress</div><div>bFeedback</div><div>xGroup</div><div>xButton</div><div>xOFF_as_MinLevel</div><div>xCentr_OFF</div><div>xCentr_ON</div><div>bReferenceAddress1</div><div>bReferenceAddress2</div><div>bModule_750_641</div></div></div>			

Function description:

This function block is used to implement a DALI latching relay.

The short or group address to which the DALI commands are to be sent is specified at the input **"bAddress"**. The value at input **"xGroup"** determines whether the entered address is interpreted by the function block as a short or group address (FALSE = short address; TRUE = group address).

A rising edge at the input **"xButton"** causes the lighting addressed via short or group address to be switched on or off. Whether the lighting is switched on or off depends on the previous switching status of the lighting.

If the input **"xOFF_as_MinLevel"** is TRUE, instead of the switch-off command, the lighting is dimmed to the min. level.

The inputs **"xCentr_ON"** and **"xCentr_OFF"** are used for forced control of the lighting via a central command.

The DALI master module with which this function block must communicate is selected at input **"bModule_750_641"**.

It is obligatory to give a reference control gear from the group if the group is to be switched. For redundancy reasons it is possible to give two reference values ("bReferenceaddress1" and "bReferenceaddress2"). The first reference value from the group must absolutely be available.

The output **"bFeedback"** outputs a numeric code with the response. The numeric codes are listed in table 6 in the appendix.

G DALI Master

DALI Master

WAGO-I/O PRO CAA Library Elements			
Category:		Building Automation	
Name:		FbDALI_Master	
Type:		Function <input type="checkbox"/>	Function block X <input type="checkbox"/> Program <input type="checkbox"/>
Name of library:		DALI_02.lib	
Applicable to:		See release note	
Input parameter:		Data type:	Comment:
bAddress		BYTE	Short address of 1–64 or Group address 1–16 Broadcast = 255
iCommand		INT	Input of DALI commands in accordance with DALI command set (see DIN IEC 60929) Example: 0 = Off 1 = On 2 = Darker etc. (see appendix, table 1)
bCommandValue		BYTE	Command value (e.g. brightness)
bModule_750_641		BYTE	Specifies which DALI master module is to be addressed at the controller. Counting is from left to right. Value range = 1 – 5 Default setting = 1
Input/output parameter:		Data type:	Comment:
xStartDaliMaster		BOOL	Starts the command. This bit is to be set by the user. The block resets it after execution of the command.
Feedback value:		Data type:	Comment:
bQueryValue		BYTE	This byte returns the query value in accordance with the DALI specification.
bFeedback		BYTE	Response (see table 6 in the appendix)
Graphical display:			
<div><div>FbDALI_Master</div><div><div>bAddress</div><div>bQueryValue</div><div>iCommand</div><div>bFeedback</div><div>bCommandValue</div><div>bModule_750_641</div><div>xStartDaliMaster ▶</div></div></div>			

Function description:

The **FbDALI_Master** function block enables you to initiate the DALI commands specified in DIN IEC 60929 (see list in the appendix, table 1).

To address a single ballast, specify the short address (1–64) of the corresponding ballast at the input **"bAddress"**. If you wish to address a group, enter the corresponding group (1–16) at the input **"bAddress"**.

To be able to send broadcast commands, the address 16#FF (255) can be entered.

The command number for the DALI command is to be entered at the input **"iCommand"**. If this command additionally requires a command value, this must be specified at the input **"bCommandValue"**. To be able to differentiate between short address and group commands, an offset of 300 must be added for the commands of group commands.

Example:

1. Command to a short address

"bAddress" = 1 ; **"iCommand"** = 0 →

"Off" command to ballast with short address 1

2. Command to a group

"bAddress" = 1 ; **"iCommand"** = 300 →

"Off" command to ballasts with group address 1

The short address commands 32 to 128 and group commands 332 to 364 must be sent twice within 100 ms to enable correct execution.

The DALI master module with which this function block must communicate is selected at input **"bModule_750_641"**.

The sending of the DALI command is triggered by the signal TRUE at the input **"xStartDaliMaster"**. This input must be specified using a variable that can be set by the user. The block resets this variable itself, so that the user can then initiate a further DALI command.

If a query is sent to the DALI slaves, the response is then sent to the **"bQueryValue"** output.

The output **"bFeedback"** outputs a numeric code as response. The numeric codes are listed in table 6 in the appendix

Note:

The input or output of dimming values is done in the block **FbDALI_Master** in the value range 0 - 254 (logarithmic dimming curve).

The DALI dimming value can be converted into a percentage via the **FuDimmValue_Percent** function.

D DALI FuDimmValue

Converting Percentage -> DALI Dimm Value (FuDimmValue_DALI)

WAGO-I/O PRO CAA Library Elements			
Category:	Building Automation		
Name:	FuDimmValue_DALI		
Type:	Function <input checked="" type="checkbox"/>	Function block <input type="checkbox"/>	Program <input type="checkbox"/>
Name of library:	DALI_02.lib		
Applicable to:	See release note		
Input parameter:	Data type:	Comment:	
bDimmValue_Percent	BYTE	Input of the dim value in percentage	
Feedback value:	Data type:	Comment:	
FuDimmValue_DALI	BYTE	Output of the DALI dim value (0 – 254)	
Graphical display:			
<div><div>FuDimmValue_DALI</div><div>bDimmValue_Percent</div></div>			
Function description:			
This function block converts a dim value of 0 – 100 percent into a DALI dim value (0 – 255).			

E Liste des commandes DALI

Command List for FbDALI_Master + FbDALI_Master_Adv

Table 1

Command set for short addresses or broadcast	
0	Power off
1	Up
2	Down
3	Step up
4	Step down
5	Recall max. level
6	Recall min. level
7	Step down and off
8	On and step up
9 - 15	Reserved
16 - 31	Go to scene 1 - 16
32	Reset
33	Store actual level in the DTR
34 - 41	Reserved
42	Store the DTR as max value
43	Store the DTR as min value
44	Store the DTR as system failure value
45	Store the DTR as 'power on value'
46	Store the DTR as fade time
47	Store the DTR as fade rate
48 - 63	Reserved
64 - 79	Store the DTR as scene 1 - 16
80	Remove from scene
81 - 95	Reserved for "Remove from scene"
96 - 111	Add to group 1 - 16
112 - 127	Remove from group 1 - 16
128	Store the DTR as short address
129 - 143	Reserved
144	Query status
145	Query ballast
146	Query lamp failure
147	Query lamp power on
148	Query limit error
149	Query reset state
150	Query short address missing
151	Query version number
152	Query contents DTR
153	Query device type
154	Query physical min value
155	Query power failure
156 - 159	Reserved

160	Query current value
161	Query max value
162	Query min value
163	Query 'Power on value'
164	Query system failure value
165	Query fade time / fade rate
166 – 175	Reserved
176 -191	Query scene value (scenes 1 to 16)
192	Query groups 1 to 8
193	Query groups 9 to 16
194	Query random address (H)
195	Query random address (M)
196	Query random address (L)
197 – 223	Reserved
224 – 255	Query application-related extension commands
999	Direct control of lamp power
Command set for group commands (WAGO specific)	
300	Power off group
301	Group up
302	Group down
303	Group step up
304	Group step down
305	Group recall max. level
306	Group recall min. level
307	Group step down and off
308	Group on and step up
309 - 315	Reserved
316 - 331	Group go to scene 1 - 16
332	Group reset
333	Store group current value in DTR
334 – 341	Reserved
342	Group store the DTR as max value
343	Group store the DTR as min value
344	Group store the DTR as system failure value
345	Group store the DTR as 'Power on value'
346	Group store the DTR as fade time
347	Group store the DTR as fade rate
348 - 363	Reserved
364 - 379	Group store the DTR as scene 1 - 16
380 - 395	Group remove from scene
396 - 411	Group add to group
412 - 427	Group remove from group
428	Store the DTR as short address
429 - 443	Reserved
1299	Direct control of lamp power

Note:

DTR = Data Transfer Register