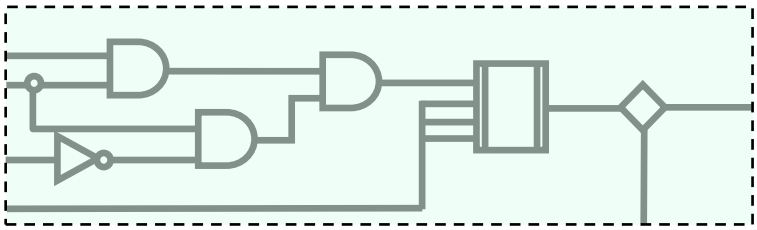
Zennio
Passion for KNX innovation



Fonctions logiques

Module de fonctions logiques-mathématiques

MANUEL DE L'UTILISATEUR

Édition du manuel: [0.2]_b

www.zennio.fr

Sommaire

ACTUALISATIONS DU DOCUMENT

Version	Changements	Page(s)
[0.2]_b	Note par rapport aux appels successifs à des fonctions avec résultats retardés.	18
	Les conversions à des valeurs de pourcentage d'un byte sont équivalentes aux conversions à des valeurs entiers d'un byte (c'est pour cela qu'une option spécifique ne se propose pas).	23, 24

1 INTRODUCTION

1.1 MODULE DE FONCTIONS LOGIQUES

Beaucoup de dispositifs Zennio (entre lesquels se trouvent toute la famille d'actionneurs ACTinBOX et celle des actionneurs MAXinBOX) disposent d'un module de fonctions logiques, ce qui permet d'effectuer des opérations **mathématiques** ou en **logique binaire** avec les données en provenance du bus KNX ainsi que d'envoyer les résultats au moyen d'objets de communication de différentes tailles.

Les opérandes de ces fonctions peuvent être parmi les types suivants:

- **Objets de communication** reçus depuis le bus KNX**.**
- **Variables Internes** avec des résultats partiels d'opérations précédentes.
- **Valeurs constantes**, établies depuis ETS.

Il est possible de définir dix fonctions logiques différentes et indépendantes, chacune pouvant avoir jusqu'à quatre opérations successives, qui pourront partager des variables entre elles de façon à ce que le résultat d'une des opérations soit une donnée d'entrée de la suivante.

Important: *en fonction du dispositif, le nombre de fonctions logiques disponibles et leur fonctionnalité peuvent varier légèrement, c'est pour cette raison que le manuel d'utilisateur de ce module a été personnalisé pour chaque dispositif. Ainsi, il est fortement recommandé d'utiliser les liens de téléchargement qui figurent sur la fiche du dispositif que vous désirez paramétrer, dans le site web de Zennio (www.zennio.fr).*

2 CONFIGURATION

2.1 FONCTIONNEMENT GÉNÉRAL

Le module de fonctions logiques permet d'habiliter et paramétrer jusqu'à dix fonctions numériques indépendantes. Le fonctionnement de chacune de ces fonctions est divisé en quatre étapes:

- **Appel**: pour que la fonction paramétrée soit exécutée, il faut d'abord *l'appeler*. Pour ce faire, il est possible de paramétrer un ou plusieurs objets de communication, de sorte que à chaque fois que la valeur d'un d'entre eux est modifiée depuis le bus KNX, l'exécution de la fonction sera enclenchée automatiquement (si et seulement si la condition d'exécution est accomplie).

- **Condition d'exécution**: après l'activation d'un déclencheur assigné à une fonction logique, la condition d'exécution sera vérifiée. Cette condition consiste en la comparaison entre deux opérandes d'entrée. Dans le cas où le résultat de la comparaison soit vrai, les opérations seront exécutées. Si aucune condition d'exécution est paramétrée, la fonction sera exécutée à chaque appel.

- **Opérations**: l'exécution de la fonction déclenchera, à son tour, l'exécution successive de jusqu'à quatre opérations mathématiques ou binaires. Pour chacune d'entre elles nous devrons définir les paramètres suivants:

  - **Type d'opération**: action désirée (addition, soustraction, négation, etc.).

  - **Opérandes**: valeurs utilisées dans l'opération. Elles pourront être des objets de communication d'entrée, des variables internes dans lesquelles nous ayons stocké le résultat d'une opération précédente, ou des constantes prédéfinies sur ETS.

  - **Résultat**: variable interne où stocker le résultat de l'opération.

- **Résultat**: vous devrez sélectionner quelle variable interne doit stocker le résultat global de la fonction. Sa valeur sera envoyée, après exécution de toutes les opérations, au travers de l'objet de communication qui correspond.

Le fonctionnement général des fonctions logiques est schématisé sur la Figure 1.

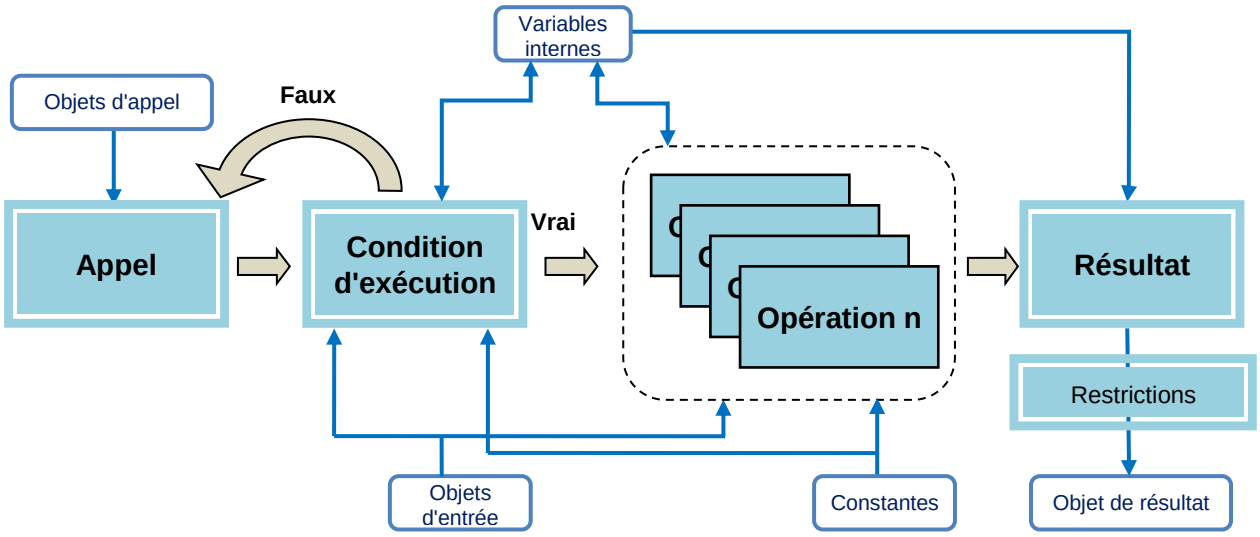


Figure 1. Diagramme de fonctionnement des fonctions logiques.

2.2 DÉCLENCHEURS

Pour chaque fonction, vous disposerez de jusqu'à huit objets d'appel (d'un bit, un byte deux bytes ou quatre bytes), chacun déclenchera la fonction automatiquement à chaque fois qu'il recevra une valeur depuis le bus. Ces objets n'ont pas forcément à coïncider avec les objets utilisés comme opérandes.

Notez que l'exécution des opérations, après que la fonction soit déclenchée, dépend de si la condition d'exécution est accomplie, s'il y en a une.

2.3 CONDITION D'EXÉCUTION

Lorsque la fonction est déclenchée, le pas suivant est de vérifier si la situation actuelle remplit les conditions nécessaires pour que l'exécution des opérations soit permise.

Si aucune condition d'exécution n'a été paramétrée, les opérations seront exécutées directement. Par contre, si elle est activée, les opérations ne seront exécutée que si la comparaison entre **deux opérandes d'entrée**, qui peuvent être des valeurs constantes, des objets de communication ou des variables internes, donne le résultat attendu.

Les opérations de comparaison disponibles sont (en fonction de la taille de comparaison choisie): est égal à; n'est pas égal à; est supérieur à; est supérieur ou égal à; est inférieur à; est inférieur ou égal à.

2.4 OPÉRATIONS

Comme indiqué précédemment, chaque fonction logique consiste en l'exécution de jusqu'à quatre opérations consécutives qui peuvent être d'un des types suivants:

- **Logique**: OUI, NON, ET, OU, OU EXCLUSIF, NON ET, NON OU et NON OU EXCLUSIF.
- **Arithmétique**: Oui, addition, soustraction, multiplication, division, maximum et minimum.
- **Comparaison**: est égal à; n'est pas égal à; est supérieur à; est supérieur ou égal à; est inférieur à; est inférieur ou égal à.
- **Conversion**: opérations de conversion du format (*cast*) d'un opérande en particulier. Par exemple: d'un bit à un byte.

Le module des fonctions logiques peut opérer avec les échelles de valeurs suivantes (que ce soit des objets, des variables internes avec des résultats intermédiaires préalables, ou des constantes établies sur ETS):

- Binaires: **0** et **1**.
- Entiers sans signe (un byte): **0 – 255**.
- Valeurs de pourcentage (un byte): **0 – 100.**
- Entiers sans signe (deux bytes): **0 – 65535**.
- Entiers avec signe (deux bytes): **-32768 – 32767**.
- Valeurs en virgule flottante (deux bytes): **-671088,64 – 671760,96**.
- Entiers avec signe (quatre bytes): **-2147483648 – 2147483647**.

Pour plus d'information sur ces opérations, sur la conversion entre tailles et sur le tronqué de valeurs consultez le *ANNEXE I: Liste des opérations*.

2.5 OBJETS D'ENTRÉE

De multiples objets spécifiques pourront être activés pour les utiliser avec les fonctions:

- Jusqu'à 32 objets d'un bit,
- Jusqu'à 16 objets d'un byte,
- Jusqu'à 16 objets de deux bytes.
- Jusqu'à 8 objets de quatre bytes.

Les valeurs de ces objets pourront intervenir, par exemple, comme opérandes dans les opérations des fonctions activées.

2.6 VARIABLES INTERNES

De la même façon, l'intégrateur pourra disposer de:

- 32 variables internes d'un bit (b1, …, b32),
- 16 variables internes d'un byte (n1, …, n16),
- 16 variables internes de deux bytes (x1, …, x8).
- 8 variables internes de quatre bytes (y1, …, y8).

Toutes permettront le stockage temporel des résultats intermédiaires qui, à leur tour, pourront être utilisées comme valeurs d'entrée des opérations postérieures.

2.7 OBJETS DE RÉSULTAT

Chaque fonction logique activée disposera d'un objet spécifique (qui sera de un bit, un byte, deux bytes ou quatre bytes, en fonction de la configuration de la fonction) au travers de laquelle la valeur de la variable interne sélectionnée comme résultat de la séquence d'opérations qui conforment la fonction sera envoyée au bus KNX.

L'intégrateur pourra configurer si l'envoi de cet objet au bus devra être réalisé une fois, à chaque exécution de la fonction, périodiquement, ou uniquement lorsque la fonction donne un résultat différent de celui de la précédente exécution.

De même, il pourra configurer si il doit y avoir une restriction sur l'envoi du résultat en fonction de si celui-ci correspond à une échelle de valeurs déterminée. Pour finir, il est aussi possible de configurer sur ETS un retard pour l'envoi du résultat.

3 PARAMÉTRAGE ETS

3.1 ÉCRAN GÉNÉRAL

L'onglet général des paramètres du module de fonctions logiques contient les options ci-dessous. Tenez en compte que cette figure se réfère au dispositif MAXinBOX 66 et qu'il peut y avoir de légères différences avec d'autres dispositifs.

Note: *Peut-être les onglets de paramètres du module de fonctions logiques ne sont pas affichés par défaut sur ETS et qu'il faille sélectionner l'activation de ce module dans l'onglet Général du dispositif. Veuillez consulter le manuel de l'utilisateur du dispositif spécifique pour plus de détails.*

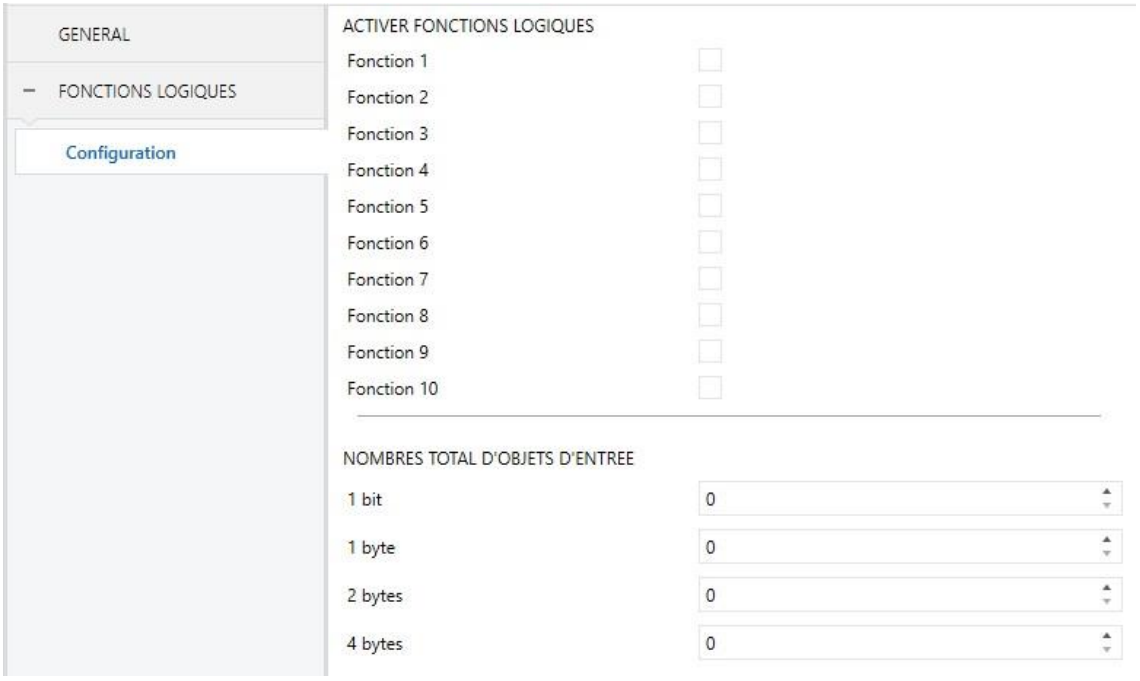


Figure 2. Onglet générale du module de fonctions logiques:

Comme vous pouvez observer, aucune des dix fonctions n'apparaît active par défaut. Dès qu'elles sont activées, les onglets additionnels correspondants apparaîtront dans le menu de gauche.

De plus, dans cet onglet il faut définir le nombre d'objets d'entrées nécessaires de chaque type pour l'ensemble de toutes les fonctions logiques qui vont être utilisées. Comme indiqué dans la section 2.5, jusqu'à 32 objets de un bit, 16 de un byte, 16 de deux bytes et 8 de quatre bytes peuvent être activés.

La fonction et les paramètres des onglets correspondants aux fonctions logiques activées seront expliqués.

3.2 FONCTION N

Pour chaque fonction activée (voir 3.1) un onglet spécifique apparaîtra dans l'arborescence de gauche, avec leurs quatre sous-onglets correspondants.

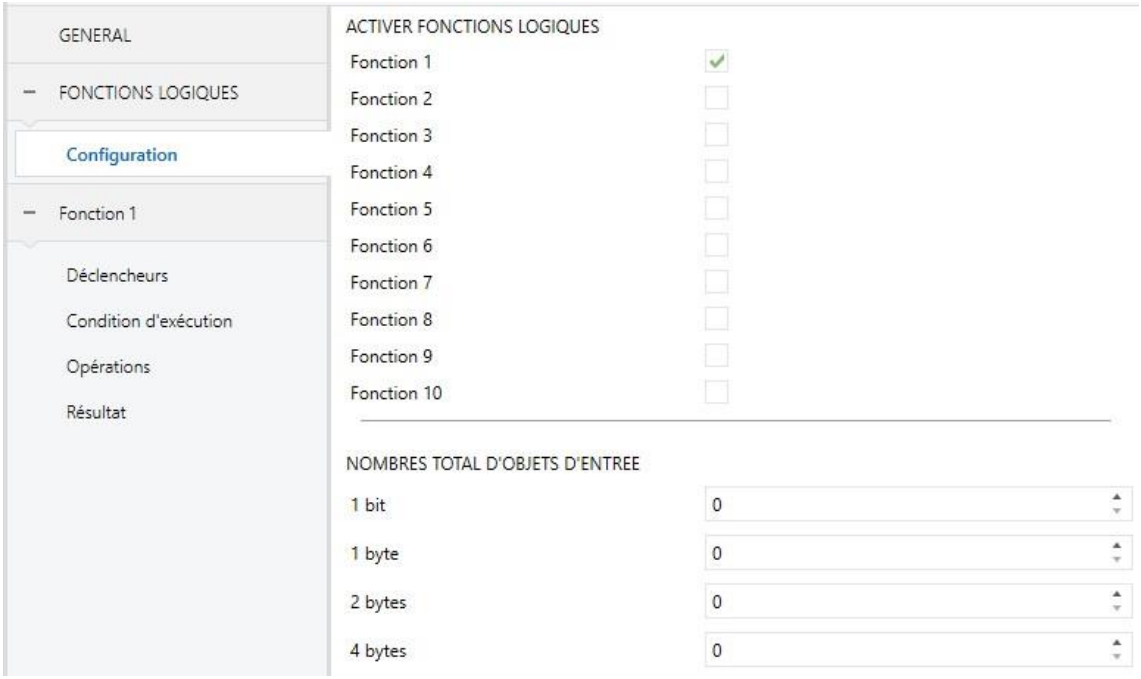


Figure 3. Fonction 1 activée.

3.2.1 DÉCLENCHEURS

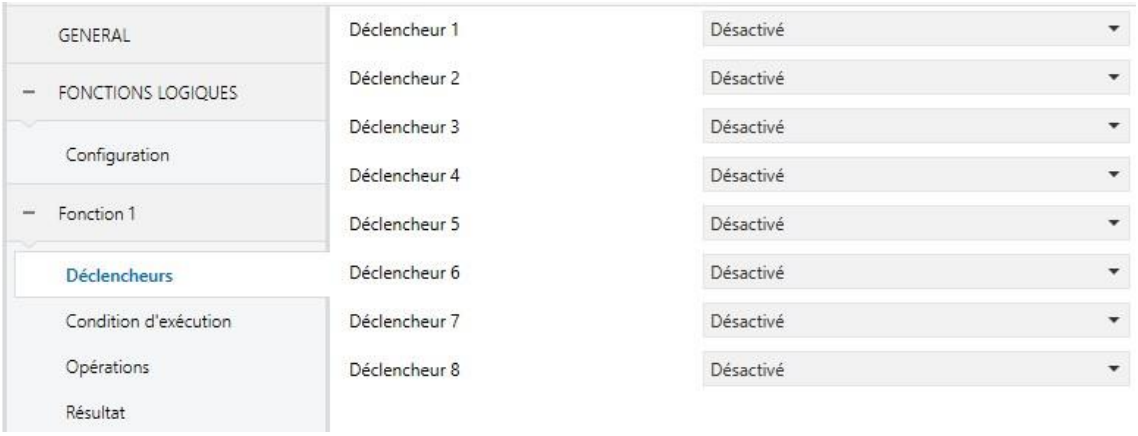


Figure 4. Onglet "Déclencheurs"

Depuis cet onglet, jusqu'à **huit déclencheurs** peuvent être sélectionnés, qui agiront comme objets d'appel. Les objets d'appel seront les responsables de déclencher l'exécution de la fonction à chaque fois qu'un d'entre eux reçoit une nouvelle valeur du bus, si et seulement si la condition d'exécution est accomplie. Également, si un même objet est utilisé comme objet d'appel de plusieurs fonctions, toutes seront déclenchées consécutivement lorsqu'il reçoit une nouvelle valeur.

Note: les objets à utiliser comme déclencheurs devront être activés au préalable (voir *3.1*).

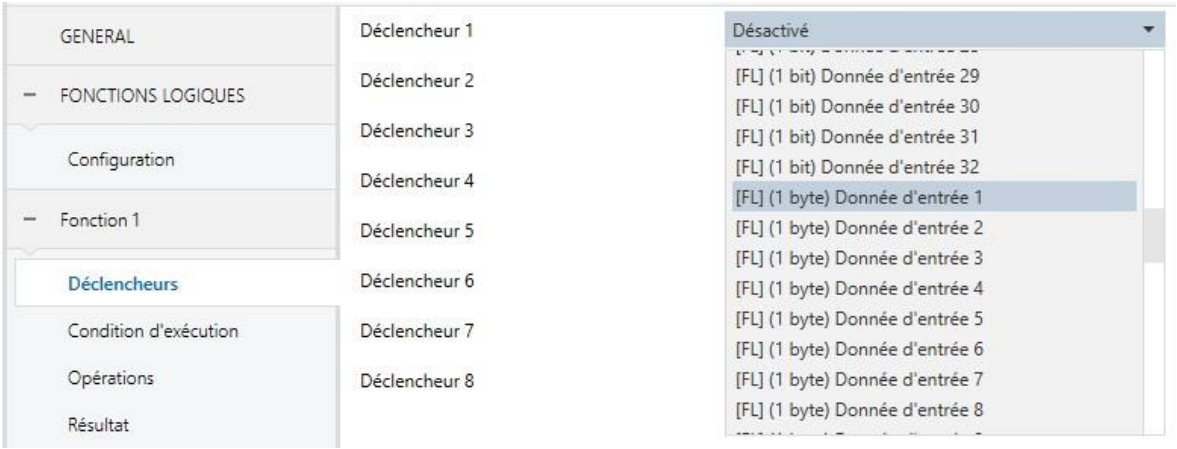


Figure 5. Sélection des objets déclencheurs

3.2.2 CONDITION D'EXÉCUTION

Depuis cette section on peut configurer la condition d'exécution au moyen des paramètres suivants:

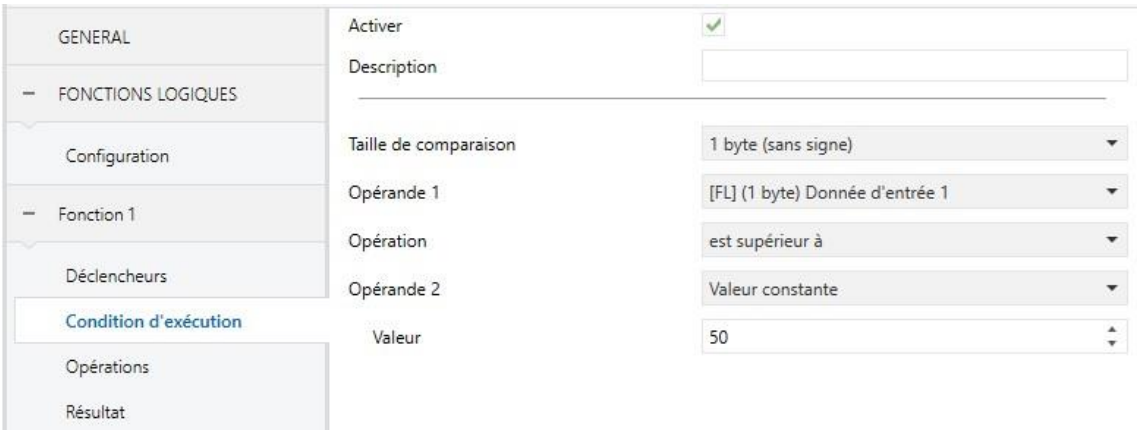


Figure 6. Onglet "Condition d'exécution"

- **Activer**: établit si on désire ou non activer une condition d'exécution.
- **Description:** permet de spécifier une brève description (jusqu'à cent caractères). Ce champ ne possède aucune implication pratique; elle permet simplement à l'intégrateur d'identifier la fonction.
- **Taille de comparaison:** taille des opérandes d'entrée: 1 bit, 1 byte (sans signe), 1 byte (pourcentage), 2 bytes (sans signe), 2 bytes (avec signe), 2 bytes (virgule flottante), 4 bytes (avec signe).
- **Opérande 1:** origine de la valeur: constante, objet d'entrée ou variable interne. Si l'option "Valeur constante" est sélectionnée, un nouveau champ apparaîtra ("**Valeur**") pour spécifier la valeur de la constante désirée.
- **Opération:** la liste d'options dépendra du champ **Taille de comparaison**. Si l'option est 1 bit, seules les options "est égal à" et "n'est pas égal à" sont disponibles. Si elle est supérieure à 1 bit, toutes les options seront disponibles (voir chapitre 2.4).
- **Opérande 2:** origine de la valeur: constante, objet d'entrée ou variable interne. Si l'option "Valeur constante" est sélectionnée, un nouveau champ apparaîtra ("**Valeur**") pour spécifier la valeur de la constante désirée.

3.2.3 OPÉRATIONS

Le nombre maximum d'opérations par fonction logique est de quatre. Celles-ci s'activent individuellement et s'exécuteront de séquentiellement. S'il y a une opération désactivée, elle sera ignorée.

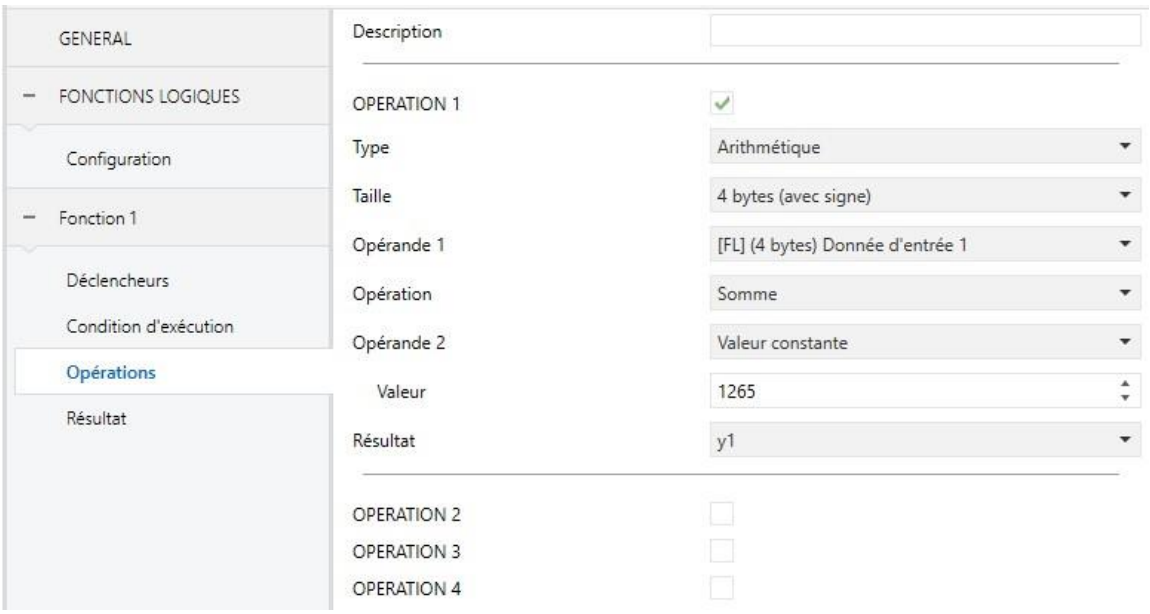


Figure 7. Onglet "Opérations".

Dans chaque opération, il est possible de configurer les paramètres suivants:

- **Description**: permet de spécifier une brève description (jusqu'à cent caractères) de la fonction logique. Ce champ ne possède aucune implication pratique; elle permet simplement à l'intégrateur d'identifier la fonction.

- **Opération "i"**: permet d'activer ou désactiver l'opération "i" (1-4). Chaque opération activée montrera à son tour les paramètres suivants:

  - **Type:** permet de sélectionner le type d'opération (logique, arithmétique, de comparaison ou de conversion). Un paramètre additionnel ("**Taille**") apparaîtra pour la sélection de la taille du résultat, sauf si l'option "Logique" est sélectionnée: 1 bit, 1 byte (sans signe), 1 byte (pourcentage), 2 bytes (sans signe), 2 bytes (avec signe), 2 bytes (virgule flottante) et 4 bytes (avec signe).

- **Opération**: permet de sélectionner l'action qu'exécutera l'opération "i". En fonction du type d'opération sélectionnée (logique, arithmétique, de comparaison ou de conversion), ce paramètre montrera des options différentes. Pour plus d'information, voir l'*ANNEXE I: Liste des opérations*.

- **Opérande "j"**: suivant l'option sélectionnée dans le paramètre précédent, un ou plusieurs paramètres intitulés "Opérandes j" seront disponibles, au moyen desquels seront établies les valeurs d'entrée (opérandes) de l'opération. Ceux-ci peuvent être des objets de communication, des variables internes ou des valeurs constantes. Voir la section 2.4 et suivantes.

- **Résultat de l'opération**: permet de sélectionner la variable interne dans laquelle sera gardée le résultat de l'opération. Ce résultat intermédiaire pourra être utilisé comme opérande d'entrée dans les opérations successives ou comme résultat final de la fonction, si besoin.

Note: *les variables internes sont communes pour toutes les fonctions logiques. Par exemple, si la fonction 1 garde un résultat intermédiaire dans la variable interne "n1" et ensuite la fonction "2" utilise cette variable interne comme donnée d'entrée, la valeur lue sera celle définie par la fonction 1.*

3.2.4 RÉSULTAT

Dans cette section on pourra sélectionner quelle variable interne est celle qui détermine le résultat final de la fonction, de sorte qu'après exécution des opérations qui conforment la fonction, la valeur de cette variable pourra être consultée et envoyée sur le bus au moyen de l'objet "**[FL] Fonction n - Résultat**".

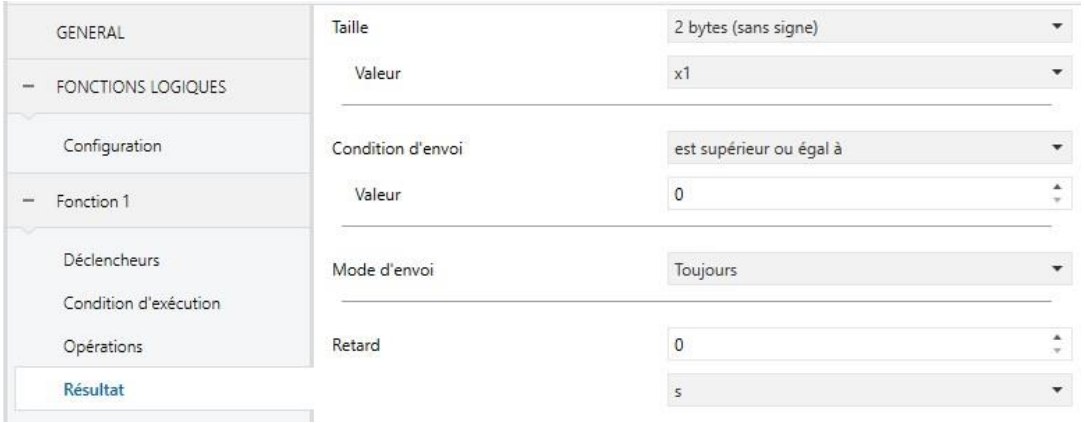


Figure 8. Onglet "Résultat".

- **Taille:** établit la taille du résultat de la fonction. Les options sont: 1 bit, 1 byte (sans signe), 1 byte (pourcentage), 2 bytes (sans signe), 2 bytes (avec signe), 2 bytes (virgule flottante) ou 4 bytes (avec signe).

- **Valeur**: détermine la variable interne dont la valeur sera envoyée sur le bus, au moyen de l'objet de résultat de la fonction, après exécution des opérations.

- **Condition d'envoi:** établit les restrictions sur le résultat qui sera envoyé sur le bus, de sorte que seuls seront envoyés les résultats qui accomplissent cette restriction. Les options disponibles sont:

  - **Pour un résultat de 1 bit:**

    - Sans restrictions,
    - Est égal à,
    - N'est pas égal à.

  - **Pour résultats d'autres tailles (1 byte, 2 bytes, 4 bytes)**:

    - Sans restrictions,
    - Est égal à,
    - N'est pas égal à,
    - Est supérieur à,
    - Est supérieur ou égal à,
    - Est inférieur à,
    - Est inférieur ou égal à.

    Lorsqu'une restriction est choisie, il faudra aussi remplir le paramètre "**Valeur**". Le rang de ce paramètre dépend de la *Taille* choisie (voir section 2.4).

- **Mode d'envoi**: dans ce champ est défini sous quelle condition sera envoyé le résultat sur le bus KNX, une fois que les restrictions précédentes sont satisfaites.

  - Toujours.

  - Résultat différent au précédent: le résultat de la fonction ne sera envoyé sur le bus que si celui-ci est différent de celui de l'exécution précédente.

- Envoi périodique: après exécution de la fonction, le résultat sera renvoyé sur le bus plusieurs fois, avec un intervalle de temps qui doit être défini dans le paramètre "**Périodicité**". La fourchette de valeurs de ce paramètre dépend de l'unité de temps choisie: de 10 à 600 dixièmes de seconde, de 1 à 3600 secondes, de 1 à 1440 minutes ou de 1 à 24 heures.

  Tenez en compte que l'envoi périodique du résultat **n'implique pas que celui-ci soit recalculé à nouveau** avant chaque envoi, ce qui implique que les changements des objets d'entrée ne modifieront pas la valeur envoyée sur le bus; pour cela il faudrait déclencher à nouveau la fonction. Notez aussi qu'en déclenchant à nouveau la fonction, l'envoi périodique précédent sera interrompu et le compteur du temps de période sera réinitialisé, sauf si le nouveau résultat n'accomplit pas la condition d'envoi, auquel cas le précédent envoi périodique continuera en vigueur.

- **Retard:** établit un temps de retard sur l'envoi du résultat après exécution de la fonction. Si un envoi immédiat est désiré, il suffit de choisir la valeur "0" pour ce paramètre.

  La fourchette de valeurs de ce paramètre dépend de l'unité de temps choisie: de 0 à 600 dixièmes de seconde, de 0 à 3600 secondes, de 0 à 1440 minutes ou de 0 à 24 heures.

  Comme indiqué, le retard s'applique au moment de l'envoi de l'objet sur le bus: la fonction en soit s'exécute immédiatement après l'appel, indépendamment de si le résultat est envoyé ou non en différé, donc celui-ci ne se verra pas affecté par les changements de valeur des opérandes qui ont lieu pendant le temps de retard.

  Note: *en cas de réception d'un nouvel ordre de déclenchement d'une fonction dont le résultat n'ont pas encore été envoyé (à cause du retard), celle-ci s'exécutera à nouveau, ce qui empêchera l'envoi du résultat et réinitialisera le temps de retard.*

ANNEXE I: LISTE DES OPÉRATIONS

OPÉRATIONS DE LOGIQUE BOOLÉENNE (1 BIT)

- **OUI (identité)**

Opérande 1	*Opérande 2*	*Résultat*
0	-	0
1	-	1

- **ET (produit logique ou conjonction)**

Opérande 1	*Opérande 2*	*Résultat*
0	0	0
0	1	0
1	0	0
1	1	1

- **OU (somme logique ou disjonction)**

Opérande 1	*Opérande 2*	*Résultat*
0	0	0
0	1	1
1	0	1
1	1	1

- **OU exclusif**

Opérande 1	*Opérande 2*	*Résultat*
0	0	0
0	1	1
1	0	1
1	1	0

- **NON (négation)**

Opérande 1	*Opérande 2*	*Résultat*
0	-	1
1	-	0

- **NON ET**

Opérande 1	*Opérande 2*	*Résultat*
0	0	1
0	1	1
1	0	1
1	1	0

- **NON OU**

Opérande 1	*Opérande 2*	*Résultat*
0	0	1
0	1	0
1	0	0
1	1	0

- **NON OU EXCLUSIF**

Opérande 1	*Opérande 2*	*Résultat*
0	0	1
0	1	0
1	0	0
1	1	1

OPÉRATIONS ARITHMÉTIQUES

- **OUI (identité)**

Opérande 1	*Opérande 2*	*Résultat*
v1	-	*v1*

- **ADDITION**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*v1 + v2*

- **SOUSTRACTION**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*v1 - v2*

- **MULTIPLICATION**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*v1 * v2*

- **DIVISION**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*v1 / v2*

- **MAXIMUM**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*max(v1, v2)*

- **MINIMUM**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*min(v1, v2)*

Note: il est recommandé de lire la section Commentaires pour plus d'information sur certains cas spécifiques et sur les cas de débordements.

OPÉRATIONS DE COMPARAISON

- **EST SUPÉRIEUR À**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 si v1 > v2.* *0 si non.*

- **EST SUPÉRIEUR OU ÉGAL À**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 si v1 ≥ v2.* *0 si non.*

- **EST INFÉRIEUR À**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 si v1 < v2.* *0 si non.*

- **EST INFERIEUR OU ÉGAL À**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 si v1 ≤ v2.* *0 si non.*

- **EST ÉGAL À**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 si v1 = v2.* *0 si non.*

- **N'EST PAS ÉGAL À**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 si v1 ≠ v2.* *0 si non.*

Note: *il est recommandé de lire la section* Commentaires *pour plus d'information sur certains cas spécifiques et sur les cas de débordements.*

OPÉRATIONS DE CONVERSION

Opérande de 1 bit:

	1 byte (sans signe)	*2 bytes (sans signe)*	*2 bytes (avec signe)*	*2 bytes (flottante)*	*4 bytes (avec signe)*
0	*0*	*0*	*0*	*0,00*	*0*
1	*1*	*1*	*1*	*1,00*	*1*

Opérande de 1 byte sans signe

	1 bit	*2 bytes (sans signe)*	*2 bytes (avec signe)*	*2 bytes (flottante)*	*4 bytes (avec signe)*
0	*0*	*0*	*0*	*0,00*	*0*
10	*1*	*10*	*10*	*10,00*	*10*
180	*1*	*180*	*180*	*180,00*	*180*
255	*1*	*255*	*255*	*255,00*	*255*

Opérande de 1 byte type pourcentage

	1 bit	1 byte (sans signe)	2 bytes (sans signe)	2 bytes (avec signe)	2 bytes (flottante)	4 bytes (avec signe)
0	0	0	0	0	0,00	0
10	1	10	10	10	10,00	10
80	1	80	80	80	80,00	80
100	1	100	100	100	100,00	100

Opérande de 2 bytes sans signe

	1 bit	1 byte (sans signe)	1 byte (pourcentage)	2 bytes (avec signe)	2 bytes (flottante)	4 bytes (avec signe)
0	0	0	0	0	0,00	0
10	1	10	10	10	10,00	10
500	1	255	100	500	500,00	500
65535	1	255	100	32767	65535,00	65535

Opérande de 2 bytes avec signe

	1 bit	2 bytes (sans signe)	2 bytes (sans signe)	2 bytes (flottante)	4 bytes (avec signe)
-32768	0	0	0	-32768,00	-32768
-12000	0	0	0	-12000,00	-12000
0	0	0	0	0,00	0
12345	1	100	12345	12345,00	12345

Opérande de 2 bytes avec virgule flottante

	1 bit	2 bytes (sans signe)	2 bytes (sans signe)	2 bytes (avec signe)	4 bytes (avec signe)
-671088,64	0	0	0	-32768	-671088
-321654,98	0	0	0	-32768	-321654
0,00	0	0	0	0	0
12345,67	1	100	12345	12345	12345

Opérande de 4 bytes avec signe

	1 bit	2 bytes (sans signe)	2 bytes (sans signe)	2 bytes (avec signe)	2 bytes (flottante)
-2147483648	0	0	0	-32768	0xF800 (*)
-1	0	0	0	-1	-1,00
0	0	0	0	0	0,00
123456	1	100	65535	32767	123456,00
2147483647	1	100	65535	32767	0x7FFF (*)

(*) Les valeurs minimum et maximum de 2 bytes en virgule flottante permises dans le standard KNX sont utilisées, respectivement, comme –infini et +infini, tel qu'indiqué dans la rubrique Commentaires.

COMMENTAIRES

Comme indiqué précédemment, le module de fonctions logiques peut travailler avec les types de données suivantes:

- Binaires: **0** et **1**.
- Entiers sans signe.
  - Un byte: **0 – 255**.
  - Deux bytes: **0 – 65535**
- Valeurs de pourcentage (un byte): **0 – 100**.
- Entiers avec signe.
  - Deux bytes: **-32768 – 32767**.
  - Quatre bytes: **-2147483648 – 2147483647**.
- Virgule flottante (2 bytes): **-671088,64 – 671760,96**.

Ces opérandes pourront être des objets de communication, des variables internes dans lesquelles garder les résultats intermédiaires, ou encore, en fonction des opérations concernées, des constantes numériques établies par paramètre dans ETS.

D'autre part, il est important de tenir compte des observations suivantes:

- Les **débordements dans les opérations mathématiques** sont résolues en prenant en compte la valeur de la limite qui a été dépassée. Par exemple, une somme de 1 byte entre les valeurs 250 et 10 aura comme résultat 255, 255 étant la limite de l'intervalle permise pour les objets de 1 byte.
- **La perte de tension** sur le bus ne suppose pas la perte des valeurs que possédaient les objets ni les variables internes: au retour de la tension, les valeurs précédentes seront récupérées.
- Les opérations de **division par zéro** ne donnent aucun résultat; si une certaine fonction contient une opération de division par zéro, elle s'interrompra au terme de l'opération précédente.

- **La multiplication et la division de valeurs de pourcentage** entre elles sera calculée conformément aux exemples suivants:

  - Deux constantes:

    La multiplication et la division de deux constantes entre elles sera calculée en assumant qu'une des constantes est un entier. Par exemple, si l'opérande 1 est fixé à "25" et l'opérande 2 à "2", le résultat est "50%" (notez que le résultat de cette opération est connue d'avance).

  - Une constante et un objet:

    Cette option est conçue pour les cas où on désire, par exemple, que la position d'un volet soit toujours "k" fois ("k" étant une constante) celle d'une autre. Par exemple, si on assigne à "k" la valeur "3" et si la position du deuxième volet est "30%", le premier ira à la position de 90%.

  - Deux objets:

    - Si on désire, comme dans le cas précédent, qu'un volet B acquière "k" fois la position d'un autre volet A, "k" étant, dans ce cas, un objet entier de 1 byte sans signe (et non une constante), il sera nécessaire de convertir "k" en une variable interne, pour ensuite la multiplier par l'objet d'état de B.

    - Dans la même situation, si "k%" est la valeur d'un objet de 1 byte type pourcentage (et non une constante), pour que B acquière un "k%" de la position A (ex.: 50% de 40%, cela fait 20%), il sera nécessaire de convertir les deux en variables de deux bytes virgule flottante, les multiplier et finalement convertir le résultat en une valeur de pourcentage de 1 byte.

- Les limites supérieures (la plus positive) et inférieure (la plus négative) de l'intervalle de valeurs en virgule flottante permises par KNX (cela est,-671088,64 représenté par 0xF800, et +670760.96 représenté par 0x7FFF) sont considérés pour les calculs comme **+∞ (plus infini) et –∞ (moins infini)** respectivement, comme dans les exemples suivants:
  - (670760,96) * (-671088,64) = (-671088,64)
  - (-671088,64) * (-671088,64) = (670760,96)
  - (670760,96) * (34,00) = (670760,96)
  - (670760,96) * (-34,80) = (-671088,64)
  - (670760,96) / (65,20) = (670760,96)
  - (670760,96) / (-341,12) = (-671088,64)
  - (120000,00) / (670760,96) = (0,00)
  - (-671088,64) / (670760,96) = (-1,00)
  - (-671088,64) / (3500,66) = (-671088,64)
- En cas de paramétrage d'une **période d'envoi** du résultat et, en plus, d'un **temps de retard**, ce dernier s'appliquera seulement lors du premier envoi sur le bus, au terme des opérations. Passé le temps de retard, le résultat sera envoyé sur le bus et initiera le compte de la période de renvoi. Dans les envois suivants, le retard ne s'appliquera plus, sauf si la fonction est à nouveau déclenchée et si un nouveau résultat à envoyer sur le bus est généré, auquel cas le retard sera appliqué lors du premier envoi.

Venez nous poser vos questions sur les dispositifs Zennio à:

http://support.zennio.com

Zennio Avance y Tecnología S.L.
C/ Río Jarama, 132. Nave P-8.11
45007 Toledo (Spain).

Tél.: +34 925 23 20 02.
Tél.: +33 (0)1 76 54 09 27.
www.zennio.fr
*info@*zennio*.fr*