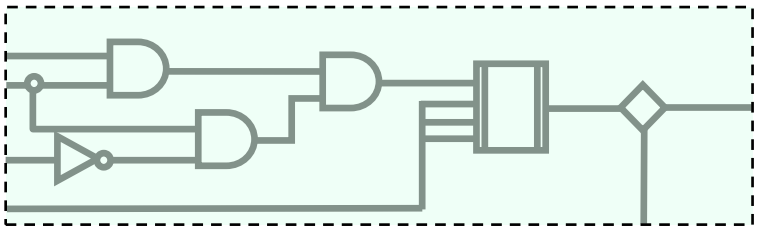
Zennio
Passion for KNX innovation



Fonctions logiques (X10)

Module de dix fonctions logique mathématiques

MANUEL D'UTILISATION

Édition du manuel: 1.c

www.zennio.fr

Table des matières

ACTUALISATIONS DU DOCUMENT

Version	Changements	Page(s)
1.c	Révision générale des textes et styles.	-
1.b	Révision générale des textes et styles.	-

1 INTRODUCTION

1.1 MODULE DE FONCTIONS LOGIQUES

Plusieurs dispositifs Zennio (par exemple les actionneurs des familles ACTinBOX et MAXinBOX, ou le compteur d'énergie KES) possèdent un module de fonctions logiques, ce qui permet d'effectuer des opérations **mathématiques** ou en **logique binaire** à partir de données venant du BUS KNX, et envoyer les résultats à partir d'objets de communication (de un bit, un byte ou deux bytes) spécifiques à cette effet.

Les opérandes de ces fonctions logiques peuvent être des types suivants:

- **Objets de communication** reçus du BUS KNX.
- **Variables internes** avec des résultats partiels d'opérations préalables.
- **Valeurs constantes**, définies par paramètre sous ETS.

En fonction du nombre de fonctions indépendantes qu'il permet de configurer, le module de fonctions logiques intégré dans le dispositif est l'un des suivants:

- **Module X5**: Permet de définir jusqu'à cinq fonctions logiques différentes et indépendantes, chacune d'elles pouvant effectuer jusqu'à quatre opérations successives et interconnectées.
- **Module X10**: Permet de définir jusqu'à dix fonctions logiques différentes et indépendantes, chacune d'elles pouvant effectuer jusqu'à quatre opérations successives et interconnectées.

Consulter le manuel du dispositif Zennio utilisé pour confirmer le module intégré (X5 ou X10).

> **Note**: *A partir de maintenant, ce manuel se centrera sur le module X10. Pour des informations spécifiques sur le module X5, consulter le manuel d'utilisateur correspondant, disponible sur la page web de Zennio.*

2 CONFIGURATION

2.1 FONCTIONNEMENT GÉNÉRAL

Le module de fonctions logiques X10 permet d'activer et configurer jusqu'à dix fonctions numériques indépendantes. Le fonctionnement de chacune de ces fonctions se divise en trois étapes:

- **APPEL**: La première étape pour que la fonction configurée soit exécutée est son *appel*. Pour cela, il est possible de configurer un ou plusieurs objets de communication, lesquels, chaque fois que leur valeur est actualisée depuis le BUS KNX, lancent automatiquement l'exécution de la fonction.

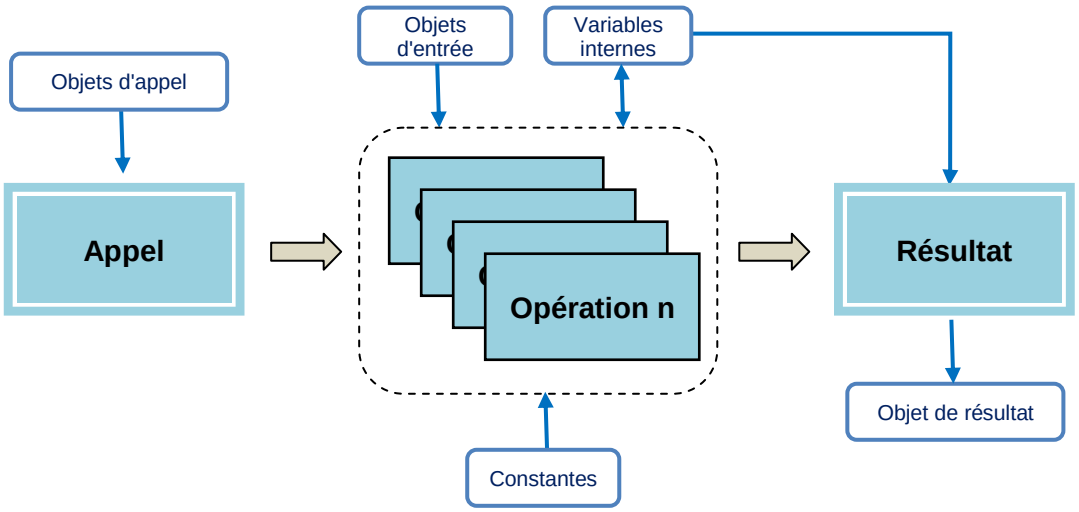
- **OPERATION**: Le déclenchement de la fonction permet de lancer l'exécution successive de jusqu'à quatre opérations mathématiques ou binaires. Pour chacune d'elles, il faut configurer ce qui suit:

  - **Type d'opération**: Action (addition, soustraction, multiplication etc.)

  - **Opérandes**: Valeurs utilisées dans l'opération. Peuvent être des objets de communication d'entrée, variables internes dans laquelle a été conservé le résultat d'une opération antérieure, ou constantes choisies sous ETS.

  - **Résultat**: Variable interne où est conservé le résultat de l'opération.

- **Résultat**: Choix de la variable interne qui conserve le résultat global de la fonction. Sa valeur est envoyée, après l'exécution de toutes les opérations, sur l'objet de communication correspondant.



2.2 OBJETS D'APPEL

Pour chaque fonction, l'intégrateur dispose de jusqu'à huit objets de d'appel (de un bit, un byte et deux bytes) qui déclenche automatiquement la fonction chaque fois qu'ils reçoivent une valeur depuis le BUS KNX. Ces objets ne sont pas obligatoirement utilisés dans la fonction logique, c'est-à-dire qu'il n'est pas obligé de les utiliser comme opérande dans les opérations.

2.3 OPÉRATIONS

Chaque fonction logique consiste à lancer jusqu'à quatre opérations consécutives. Les opérations disponibles peuvent être classées de la manière suivante:

- 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 supérieur à*, *est supérieur ou égal à*, *inférieur*, *est inférieur ou égal à*, *est égal à*, *n'est pas égal à*.
- **Conversion**: Opérations de conversion de taille d'un opérande déterminé. Par exemple: de un bit à un byte.

Le module de fonctions logiques X10 peut travailler avec les intervalles de valeurs suivants (que ce soit des objets, variables internes avec résultats intermédiaires, ou constantes définies par paramètre sous ETS):

- Binaires: **0** et **1**.
- Entiers sans signe (un byte): **0 – 255**.
- Entiers sans signe (deux bytes): **0 – 65535**.
- Nombre à virgule (deux bytes): **0,00 – 120,00**.
- Pour plus d'information sur ces opérations, sur la conversion entre tailles et sur l'arrondi des valeurs, consulter l' *ANNEXE I: Liste des opérations*.

2.4 OBJETS D'ENTRÉE

Il est possible d'activer plusieurs objets spécifiques pour les utiliser dans les fonctions:

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

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

2.5 VARIABLES INTERNES

Ainsi, l'intégrateur dispose de:

- 32 variables internes de un bit,
- 16 variables internes de un byte,
- 16 variables internes de deux bytes.

Elles permettent toutes de conserver des valeurs intermédiaires qui peuvent également être utilisées comme valeurs d'entrée d'opérations suivantes.

2.6 OBJETS DE RÉSULTAT

Chaque fonction logique activée dispose d'un objet spécifique (qui est de un bit, un byte ou deux bytes, en fonction du paramétrage) à partir duquel la valeur de la variable interne choisie par paramètre, correspondant au résultat de la fonction logique, est envoyée sur le BUS.

L'intégrateur peut configurer pour que l'envoi de cet objet sur le BUS se fasse chaque fois que la fonction est lancée, ou bien périodiquement, ou bien seulement quand la fonction induit un changement de résultat par rapport au précédent. De la même manière, il peut restreindre les résultats envoyés, de telle sorte qu'ils soient envoyés sur le BUS KNX uniquement si la valeur se trouve dans un intervalle de valeur défini.

Enfin, il est également possible de configurer sous ETS un retard pour l'envoi du résultat.

2.7 OBJETS D'ACTIVATION

En plus de ce qui a déjà été expliqué, il existe une option pour activer/désactiver chacune des fonctions logiques séparément, par l'envoi d'une valeur (selon configuration) à partir de son objet d'activation, qui est l'un des 32 objets d'entrée de 1 un bit mentionné dans le chapitre 2.4. Tant que la fonction est désactivée, elle ignore toutes les valeurs reçues sur les objets d'appel et donc ne lance aucune opération.

3 PARAMÉTRAGE ETS

3.1 ÉCRAN GÉNÉRAL:

L'onglet général des paramètres du module de fonctions logiques X10 contient les options montrées sur la figure suivante (savoir qu'il peut y avoir de petite différence d'un dispositif à l'autre).

Note: *Il est possible que l'onglet général du module des fonctions logiques ne s'affiche pas par défaut sous ETS et qu'il soit nécessaire de l'activer depuis l'onglet Général des paramètres du propre dispositif. Consulter le manuel d'utilisateur du dispositif spécifique pour plus d'information.*

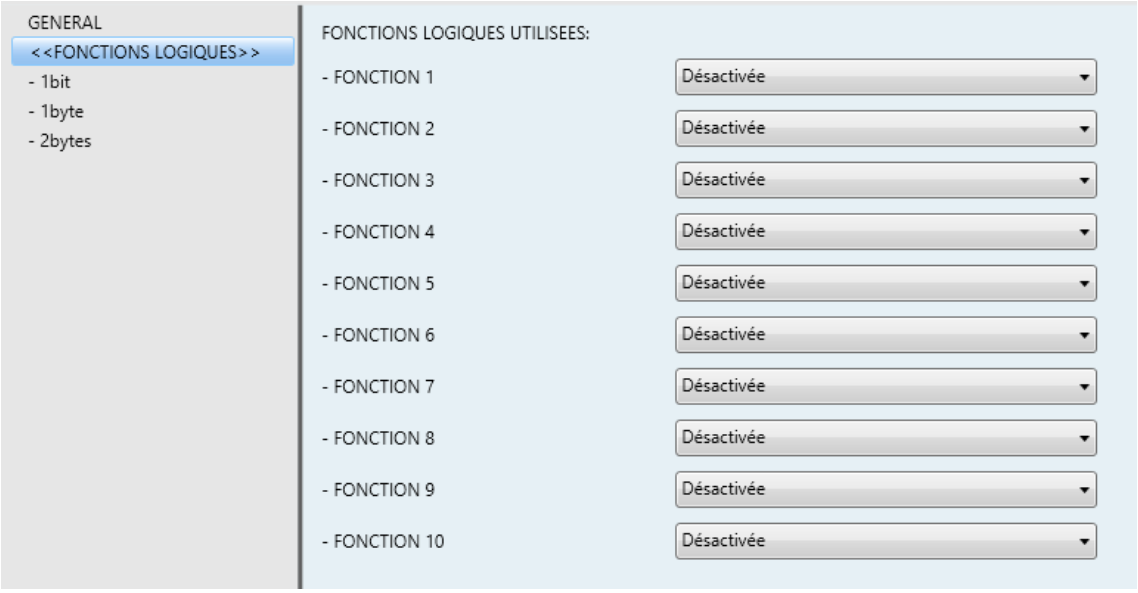


Figure 1. Onglet général du module de fonctions logiques X10

Comme il est possible de le voir, aucune des dix fonctions logiques n'est activée par défaut. Au fur et à mesure de leur activation, des onglets additionnels apparaissent dans la liste de gauche.

Dans les chapitres suivants, chaque onglet est expliqué avec les paramètres qu'ils contiennent.

3.2 1BIT, 1BYTE, 2BYTES

Ces trois onglets, qui sont visibles à tout moment, permettent à l'intégrateur d'activer, un à un, les objets d'entrées (de un bit, un byte ou deux bytes) que les fonctions utilisent, que ce soit comme opérandes, objets d'appel, etc. Ces objets ont la nomenclature suivante:

- "**[FL] Donnée (1bit) *n***",
- "**[FL] Donnée (1byte) *n***",
- "**[FL] Donnée (2bytes) *n***",

Comme déjà vu dans le chapitre 2.4, il est possible d'activer 32 objets de un bit, 16 de un byte et 16 de deux bytes. Aucun objet n'est activé par défaut.

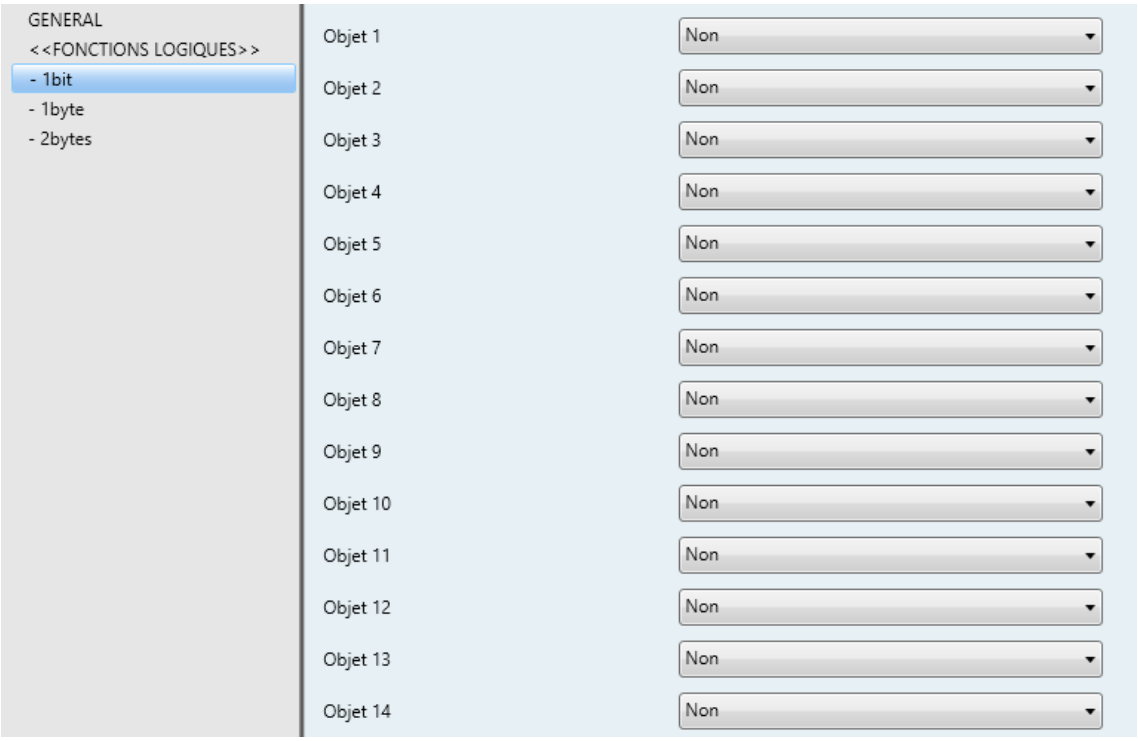


Figure 2. Activation des objets d'entrée (1 bit).

3.3 FONCTION N

Pour chaque fonction activée (voir chapitre 3.1) un nouvel onglet s'affiche sur la liste de gauche, qui est lui-même divisée en trois.

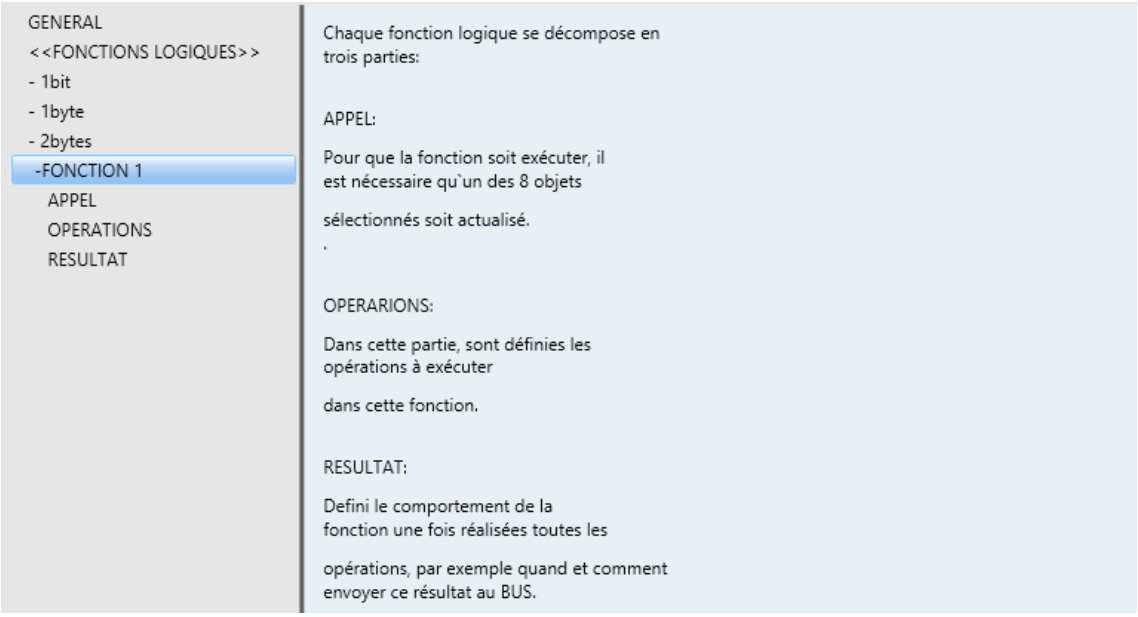


Figure 3. Fonction 1

3.3.1 APPEL

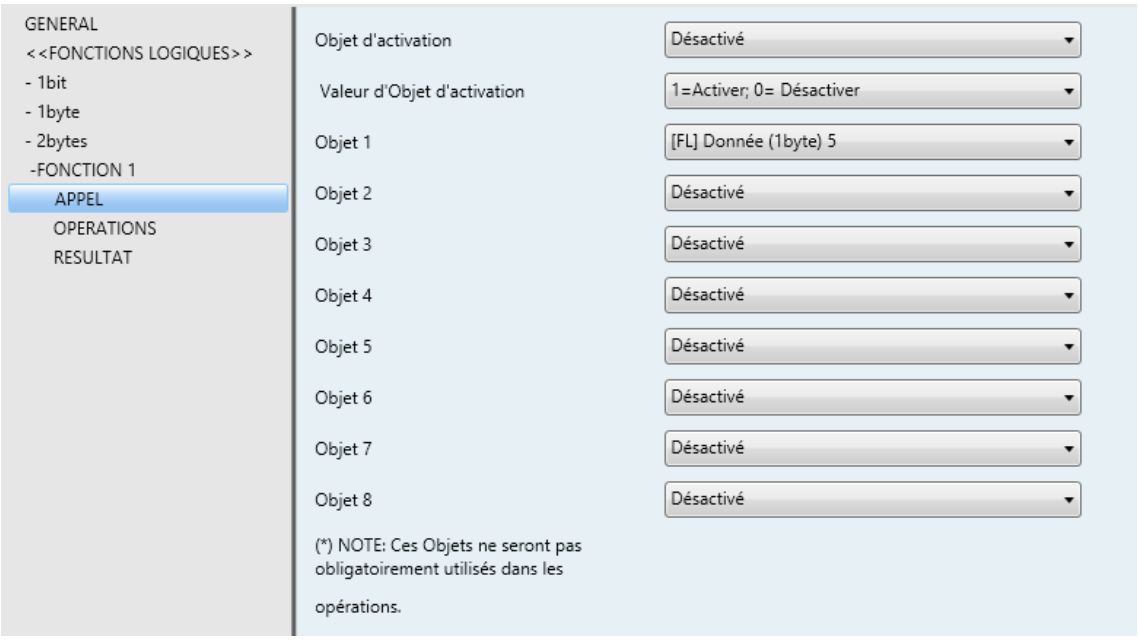


Figure 4. Onglet "Appel"

Dans cette section, il est possible de choisir jusqu'à huit objets (qui doivent avoir été activés, voir chapitre 3.2), pour agir comme objets d'appel. Les objets d'appel sont les déclencheurs des fonctions chaque fois qu'ils reçoivent une valeur du BUS. De même, si le même objet est utilisé comme objet d'appel de plusieurs fonctions, elles se déclenchent toutes simultanément au moment de la réception d'une valeur sur cet objet.

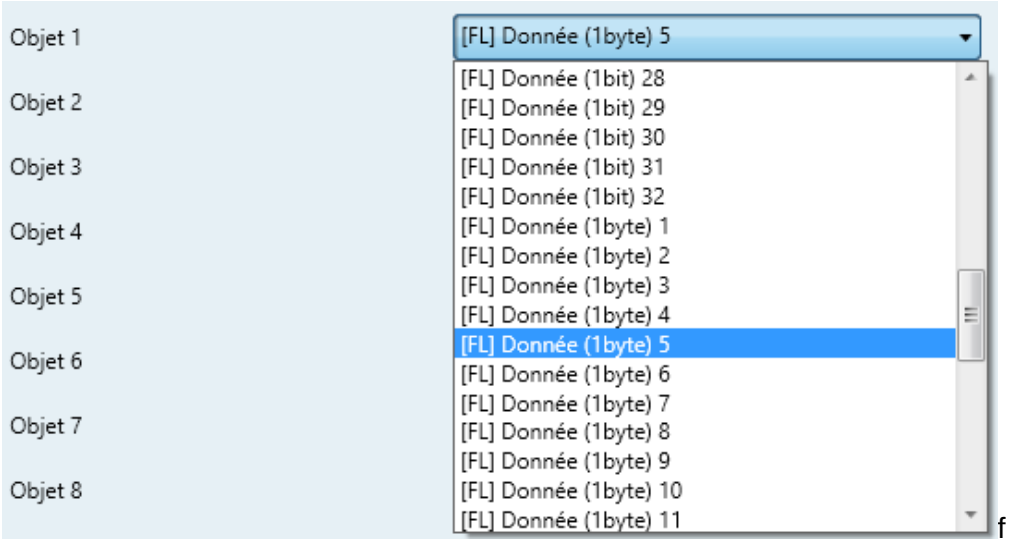


Figure 5. Choix des objets d'appel

En plus, depuis cet onglet, l'intégrateur peut déterminer lequel des objets activés (voir chapitre 3.2) doit agir comme objet d'activation de la fonction (paramètre "**Objet d'activation**") et quelle signification il a (paramètre "**Valeur d'objet d'activation**", avec les options "[0=Activer; 1=Désactiver]" et "[1=Activer; 0=Désactiver]"), comme expliqué dans le chapitre 2.7. Il est possible de ne pas configurer d'objet d'appel en choisissant l'option "Désactivé" (sélectionné par défaut).

Notes:

- *Après un téléchargement dans le dispositif, les objets d'activation sont initialisés avec la valeur correspondant à "Désactivé" ("0" ou "1" en fonction du paramétrage), alors que suite à une coupure sur le BUS, les valeurs sont les mêmes qu'avant le problème.*

- *Si l'objet d'activation est un objet qui est également utilisé dans une fonction, par exemple comme opérande ou comme résultat, faire bien attention aux valeurs qu'il reçoit, en effet, il est possible que la fonction soit activer/désactiver intempestivement.*

3.3.2 OPÉRATIONS

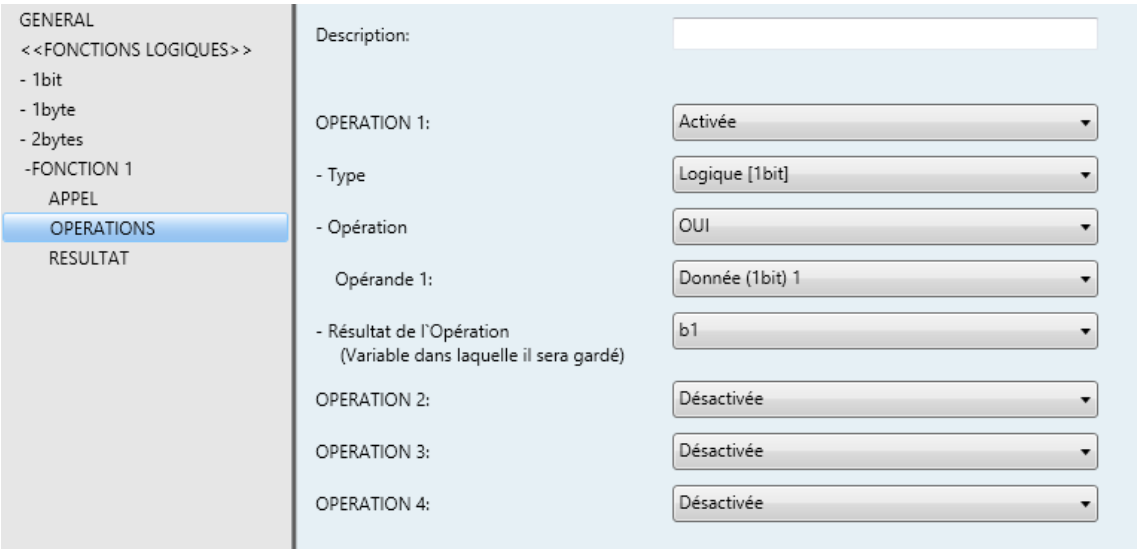


Figure 6. Opérations

Cette section présente les différents paramètres des opérations possibles dans les fonctions logiques:

- **Description**: Permet d'indiquer une brève description (jusqu'à cent caractères) de la fonction logique. Ce champ n'a aucune utilité pratique, il permet simplement à l'intégrateur d'identifier la fonction.

- **Opération "i"**: Permet d'activer ou désactiver l'opération "i" (1-4). Chaque opération activée montre les paramètres suivants:

  - **Type**: Permet de choisir le type d'opération (logique, arithmétique, de comparaison ou de conversion) et la taille des opérandes qui interviennent (un bit, entier sans signe de un byte, entier sans signe de deux bytes, virgule flottante de deux bytes). Voir chapitre 2.3.

  - **Opération**: Permet de choisir l'action réalisée par l'opération "i". En fonction du type d'opération choisi (logique, arithmétique, de comparaison ou de conversion), ce paramètre montre des options ou d'autres. Pour plus d'information, voir *ANNEXE I: Liste des opérations*.

  - **Opérande "j"**: En fonction de l'option choisie dans le paramètre précédent, d'autres paramètres s'activent avec le nom "Opérande j" à partir desquels il est possible de définir les valeurs d'entrées (opérandes)

de l'opération. Ceci peut être des objets de communication, variables internes ou valeurs constantes.

- **Résultat de l'opération**: Permet de choisir la variable interne dans laquelle le résultat de l'opération est conservé. Ce résultat intermédiaire peut être utilisé comme opérande d'entrée dans les opérations successives ou comme résultat de la fonction, si désiré.

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

3.3.3 RÉSULTAT

Dans cette section, il est possible de choisir la variable interne qui détermine le résultat final de la fonction, de telle sorte qu'après l'exécution des opérations qui composent la fonction, cette variable est consultée et envoyée sur le BUS à partir de l'objet "**[FL] RESULTAT Fonction n (taille)**", où "taille" dépend de la configuration sélectionnée.

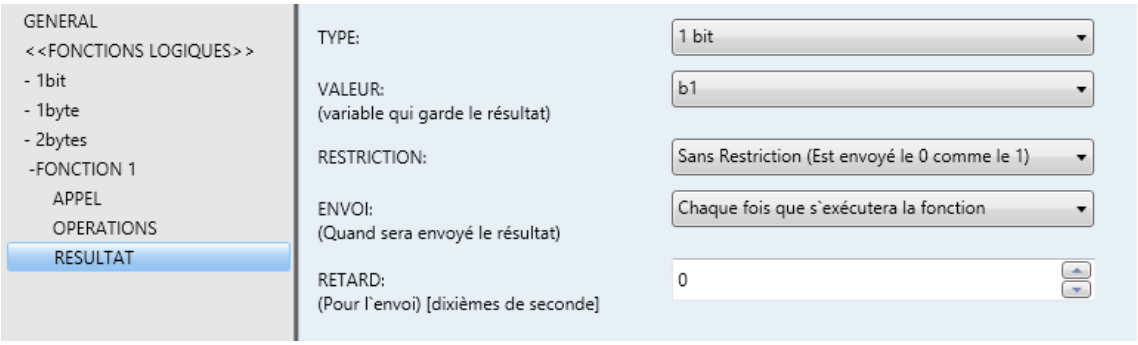


Figure 7. Résultat

- **Type**: Défini la taille du résultat de la fonction. Les options sont "1 bit", "1 byte", "2 bytes (entier sans signe)" et "2 bytes (virgule flottante)".

- **Valeur**: Détermine la variable interne à envoyer sur le BUS, à partir de l'objet de résultat de la fonction, à la fin des opérations.

- **Restriction**: Défini des restrictions sur les résultats à envoyer sur le BUS. Les options disponibles sont:

- **Un bit**:
  - Sans Restriction (Est envoyé le 0 comme le 1)
  - Envoyer uniquement le 0
  - Envoyer uniquement le 1
- **Un byte / 2 bytes (entier sans signe) / 2 bytes (Virgule flottante)**
  - Sans Restriction,
  - Envoyer uniquement Valeurs égales à celle de référence,
  - Envoyer uniquement Valeurs différentes à celle de référence,
  - Envoyer uniquement Valeurs inférieures à celle de référence,
  - Envoyer uniquement Valeurs supérieures à celle de référence,

  La valeur de référence se spécifie dans le paramètre "**Valeur de référence**", et peut prendre des valeurs entre 0 et 255 (pour le cas des entiers de un byte sans signe), entre 0 et 65535 (pour le cas des entiers de deux bytes sans signe) et entre 0 et 1200 dixièmes, c'est-à-dire, entre 0 et 120,0 (pour le cas de virgule flottante de deux bytes).

- **Envoi:** Ce champ permet de définir les conditions à respecter pour que le résultat soit envoyé sur le BUS KNX:
  - Chaque fois que s'exécutera la fonction.
  - Si changement de résultat final: Le résultat est envoyé sur le BUS uniquement sur le résultat de la fonction change par rapport au dernier résultat reçu.

    Note: *La désactivation temporaire d'une fonction à partir de l'objet d'activation n'a pas d'effet sur les restrictions: quand elle est de nouveau activée, la fonction continue à travailler avec la dernière valeur du résultat envoyée.*
  - Périodique: L'objet de résultat est envoyé sur le BUS de manière répétitive (toujours actualisé) tous les certains temps, après une première activation de celle-ci, défini dans le paramètre "**Période d'envoi**" (entre 0 et 65535 secondes).

    Savoir qu'à partir du premier envoi, l'objet est envoyé uniquement après le temps défini, ce qui implique que les réponses aux **différents appels** de la

fonction peuvent ne pas être instantanées: le temporisateur ne se réinitialise pas à chaque appel de la fonction (par contre, il se réinitialise après une coupure sur le BUS).

Note: *Si, pendant qu'il est en cours d'envoi périodique, la valeur de désactivation de la fonction est reçue sur l'objet d'activation, l'envoi périodique est désactivé et ne redémarre qu'après la réactivation de la fonction et un appel de celle-ci.*

- **Retard**: Défini un temps de retard (entre 0 et 65535 secondes) pour l'envoi de l'objet après l'exécution de la fonction. Si un envoi immédiat est souhaité, il faut indiquer la valeur "0".

  Comme vu, le retard est appliqué au moment de l'envoi de l'objet sur le BUS: la fonction est exécutée immédiatement après un appel, indépendamment de l'envoi retardé ou pas du résultat. Si un appel est reçu avant la finalisation du retard antérieur, le retard est réinitialisé et le résultat envoyé sera le nouveau après le dernier appel, sans jamais envoyer le résultat antérieur.

  Note *Si, pendant le temps de retard, la valeur de désactivation de la fonction est reçue sur l'objet d'activation, l'envoi du résultat est annulé et n'aura lieu qu'après la réactivation de la fonction et un nouvel appel de celle-ci.*

ANNEXE I: LISTE DES OPÉRATIONS

OPÉRATIONS EN LOGIQUE BOOLÉENNE (1 BIT)

- **OUI (Identique)**

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

- **ET (multiplication logique)**

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

- **OU (Addition logique)**

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 (Inversion)**

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

- **NON ET (ET Inversée)**

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

- **NON OU (OU inversée)**

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

- **NON OU EXCLUSIF (OU EXCLUSIF inversée)**

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

OPÉRATIONS ARITHMÉTIQUES

- **OUI (Identique)**

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 Observations diverses pour plus d'information sur les cas spéciaux et dépassement.*

OPÉRATIONS DE COMPARAISON

- **SUPERIEUR**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 ↔ v1 > v2.* *0 dans d'autre cas.*

- **SUPERIEUR OU EGAL**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 ↔ v1 ≥ v2.* *0 dans d'autre cas.*

- **INFERIEUR**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 ↔ v1 < v2.* *0 dans d'autre cas.*

- **INFERIEUR OU EGAL**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 ↔ v1 ≤ v2.* *0 dans d'autre cas.*

- **EGAL**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 ↔ v1 = v2.* *0 dans d'autre cas.*

- **DIFFERENT**

Opérande 1	*Opérande 2*	*Résultat*
v1	*v2*	*1 ↔ v1 ≠ v2.* *0 dans d'autre cas.*

Note *Il est recommandé de lire la section Observations diverses pour plus d'information sur les cas spéciaux et dépassement.*

OPÉRATIONS DE CONVERSION

Conversion à un bit

- **1 byte → 1 bit**

Opérande 1	*Opérande 2*	*Résultat*
0	-	0
1-255	-	1

- **2 bytes (entier sans signe) → 1 bit**

Opérande 1	*Opérande 2*	*Résultat*
0	-	0
1 - 65535	-	1

- **2 bytes (virgule flottante) → 1 bit**

Opérande 1	*Opérande 2*	*Résultat*
≤ 0,00	-	0
0,01 – 120,00	-	1

Conversion à un byte

- **1 bit → 1 byte**

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

- **2 bytes (entier sans signe) → 1 byte**

Opérande 1	*Opérande 2*	*Résultat*
0	-	0
1	-	1
...	...	...
255	-	255
256 - 65535	-	255

- **2 bytes (virgule flottante) → 1 byte**

Opérande 1	*Opérande 2*	*Résultat*
≤ 0,00		0
0,01	-	0
...	...	...
0,10	-	1
0,11	-	1
...	...	...
25,49	-	254
≥ 25,50	-	255

Conversion à deux bytes (entier sans signe)

- **1 bit → 2 bytes (entier sans signe)**

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

- **1 byte → 2 bytes (entier sans signe)**

Opérande 1	*Opérande 2*	*Résultat*
0	-	0
1	-	1
...	...	...
255	-	255

- **2 bytes (virgule flottante) →2 bytes (entier sans signe)**

Opérande 1	*Opérande 2*	*Résultat*
≤ 0,00	-	0
0,01	-	0
...	...	...
0,10	-	1
0,11	-	1
...	...	...
119,98	...	1199
119,99	-	1199
≥ 120,00	-	1200

Conversion à deux bytes (virgule flottante)

- **1 bit → 2 bytes (virgule flottante)**

Opérande 1	*Opérande 2*	*Résultat*
0	-	0,00
1	-	0,10

- **1 byte → 2 bytes (virgule flottante)**

Opérande 1	*Opérande 2*	*Résultat*
0	-	0
1	-	1
...	...	...
254	-	25,40
255	-	25,50

- **2 bytes (entier sans signe) →2 bytes (Virgule flottante)**

Opérande 1	*Opérande 2*	*Résultat*
0	-	0
1	-	0
...	...	...
1199	-	119,90
≥ 1200	-	120,00

OBSERVATIONS DIVERSES

Comme il l'a déjà été expliqué, le module X10 de fonctions logiques peut travailler avec les types de données suivants:

- Binaires: **0** et **1**.
- Entiers sans signe (un byte): **0 – 255**.
- Entiers sans signe (deux bytes): **0 – 65535**.
- Nombre à virgule (deux bytes): **0,00 – 120,00**.

Ces opérandes peuvent être des objets de communication, variables internes pour conserver les résultats intermédiaires, ou même, en fonction des opérations, des constantes numériques définies par paramètres sous ETS.

De plus, il est important de se souvenir des observations suivantes:

- Les **dépassements dans les opérations mathématiques** sont résolus en renvoyant la valeur limite possible. Par exemple, la somme d'objet de 1 byte des valeurs 250 et 10 renvoie la valeur 255 et non 260, 255 étant la limite maximale pour les objets de 1 byte.
- Les opérations de virgule flottante **arrondissent toujours les valeurs à virgule au dixième**. Par exemple, si l'objet "[FL] Donnée (2 bytes) 1" a une valeur de 2,37 et que l'opération "OUI' est exécutée sur cette valeur, le résultat renvoyé sera de 2,30. De même, additionner un objet de valeur 0,09 avec la valeur 1,27 donne 1,20 (résultat 0,0 + 1,2) et non 1,36 ni 1,30 ni 1,40.
- Les opérations de virgule flottante **arrondissent toujours à zéro les valeurs négatives**. Par exemple, si l'objet "[FL] Donnée (2 bytes) 1" vaut -5,00 et l'objet "[FL] Donnée (2 bytes) 2" vaut +10,00, le résultat de la somme de ces valeurs sera de +10.00. De même, la soustraction du premier avec le deuxième vaut 0,00 et non -15,00 ni -10,00.
- La **perte de la tension** de BUS n'implique pas la perte des valeurs que les objets ou les variables internes avaient avant cette coupure.
- Les opérations de **multiplication et division sont destinées à travailler avec un objet** (de température, par exemple) **et une constante numérique**

(configurable sous ETS) **ou une variable interne et une constante**. Il peut se produire des résultats mathématiques incorrects dus au dépassement ou autres limitations s'il est utilisé la multiplication ou division d'un objet/variable par un autre objet/variable, même si ce paramétrage est possible.

- Les opérations de **division entre zéro** ne renvoient pas de résultat.

Venez poser vos questions
sur les dispositifs Zennio à:
http://zenniofrance.zendesk.com/

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

Tel. +34 925 232 002.
Fax. +34 925 337 310.
www.zennio.fr
*info@*zennio*.fr*