programmation d'automates

avec le Raspberry Pi et le projet OpenPLC

Visualisation des programmes PLC avec AdvancedHMI

Josef Bernhardt (Allemagne)

Dans les systèmes de commande industriels, ainsi qu'en domotique, il est courant de commander et d'observer le processus en cours sur des écrans. Dans ce chapitre, extrait du dernier livre de Josef Bernardt sur la programmation des automates programmables (PLC), vous vous familiariserez avec le logiciel AdvancedHMI (« Interface Homme-Machine avancée »). Vous accéderez aux variables de l'automate et essaierez de les visualiser. Il est également possible d'accéder au processus et de simuler des interrupteurs avec lui.

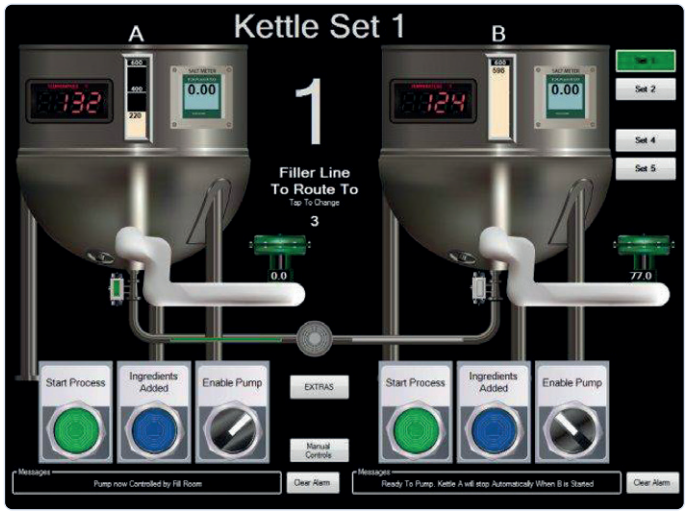


Figure 1. Exemple d'utilisation d'AdvancedHMI.

AdvancedHMI est un programme .NET écrit en Visual Basic dans l'EDI Visual Studio Community de Microsoft. Si vous installez le *Mono Framework* sur votre Raspberry Pi, vous pouvez également exécuter le logiciel sur celui-ci et avoir accès au programme PLC en cours d'exécution. Le logiciel AdvancedHMI et l'EDI Visual Studio Community peuvent tous deux être téléchargés gratuitement. La **figure 1** illustre un exemple d'AdvancedHMI en action.
Tout d'abord, téléchargez le logiciel sur le site web de l'éditeur et écrivez un programme de test pour votre Raspberry Pi utilisé comme automate. Pour le télécharger, rendez-vous sur le site web de l'éditeur, **www.advancedhmi.com**, et ajoutez le « AdvancedHMI Base Package » à votre panier, comme le montre la **figure 2**. Ensuite, cliquez sur la case *Check out*.

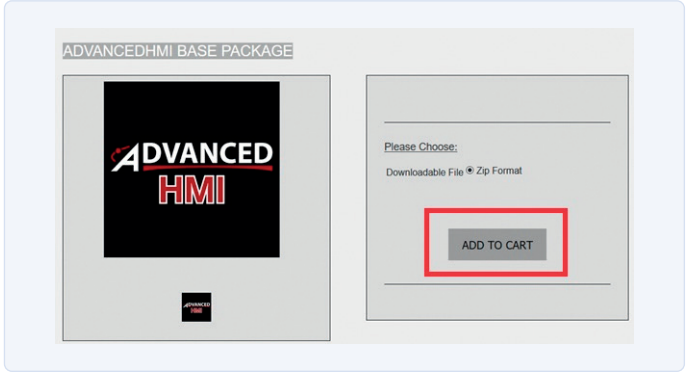


Figure 2. Téléchargement d'AdvancedHMI.

Note de l'éditeur : cet article est un extrait du livre *PLC Programming and the OpenPLC Project - ModbusRTU and ModbusTCP examples using the Arduino Uno and the ESP8266* formaté et légèrement modifié pour correspondre aux normes éditoriales et à la mise en page du magazine *Elektor*. Puisque cet article est extrait d'une publication plus vaste, certains termes peuvent faire référence à des passages du livre d'origine situés ailleurs. L'auteur et l'éditeur ont fait de leur mieux pour l'éviter et seront heureux de répondre aux questions – pour les contacter, voir l'encadré « Des questions, des commentaires ? ».

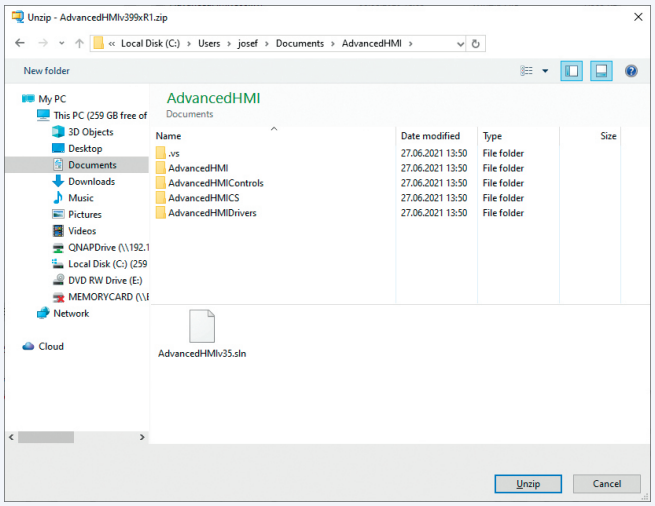


Figure 3. Décompression d'AdvancedHMI.

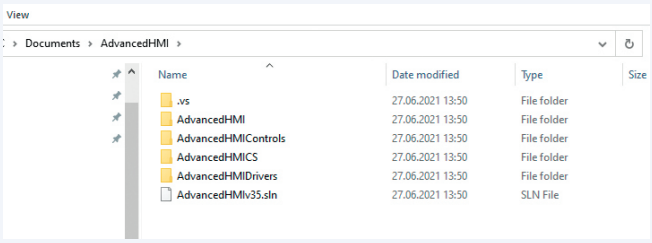


Figure 4. Le répertoire du projet AdvancedHMI.

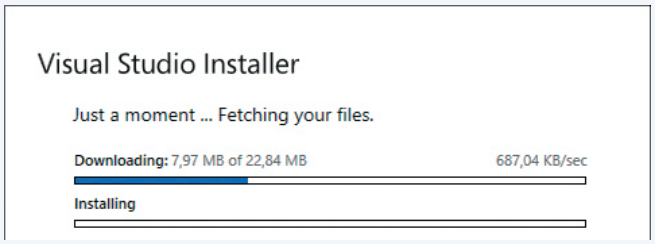


Figure 5. Le programme d'installation de Visual Studio.

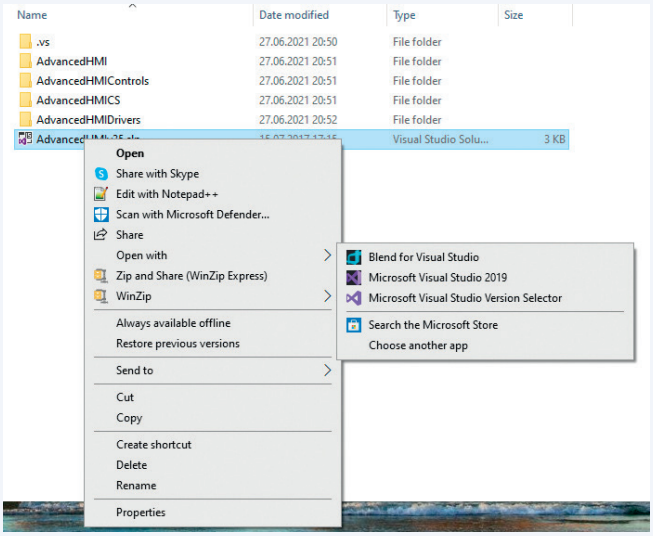


Figure 6. Ouvrez votre projet AdvancedHMI.

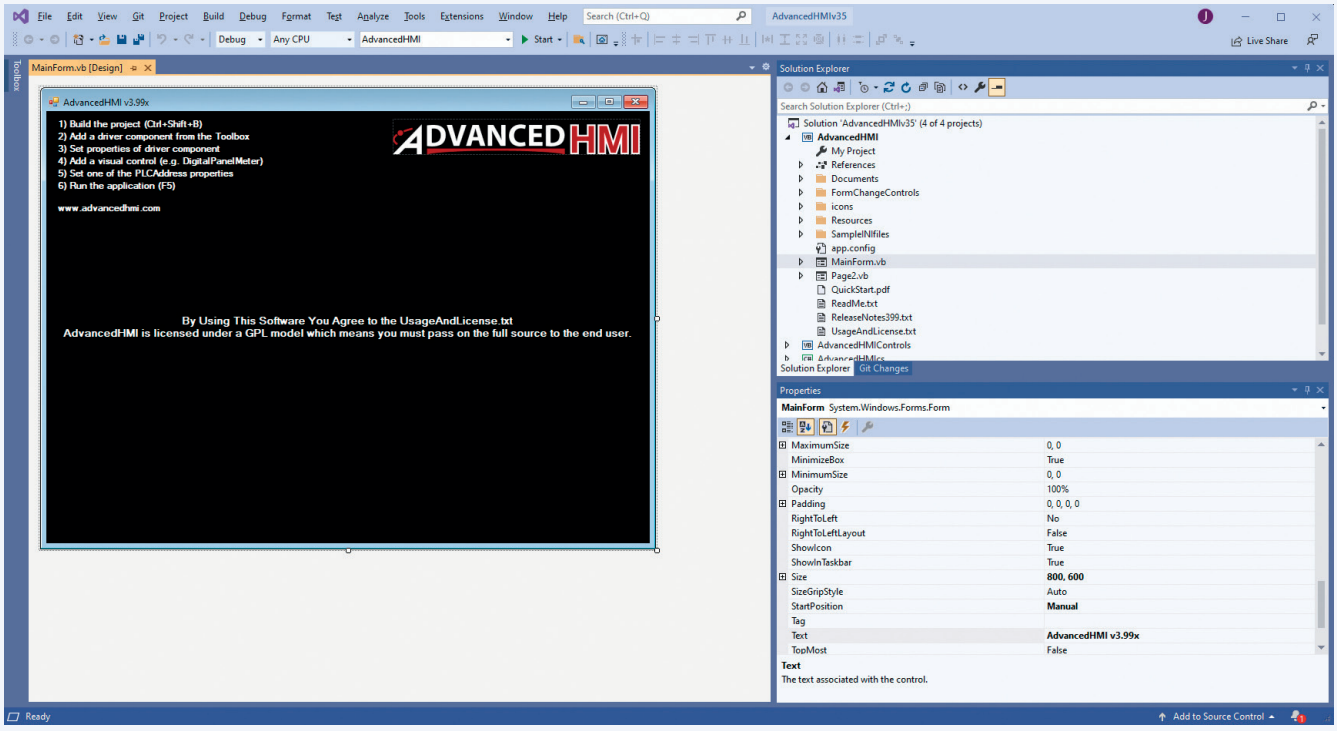


Figure 7. L'interface utilisateur de Visual Studio.

Préface du livre

Ce livre a pour but de fournir aux lecteurs une introduction pratique à l'utilisation de l'ordinateur Raspberry Pi comme automate programmable (PLC) dans leurs projets. Le projet doit beaucoup aux programmeurs Edouard Tisserant et Mario de Sousa, qui ont lancé le « projet Matiec » après la publication de la norme CEI 61131-3 en 2003, ce qui a rendu possible la traduction des langages de programmation définis dans cette norme en programmes en C. Plus tard, lorsque le Raspberry Pi est devenu de plus en plus populaire, Thiago Alves a lancé le projet « openplcproject ». Il a étendu l'éditeur du projet « Beremiz » et a écrit une bibliothèque d'exécution et une interface web pour le Raspberry Pi et le PC. Dès lors, il devint possible d'écrire des programmes sur le PC et de les installer sur le Raspberry Pi.
De nombreux utilisateurs de Raspberry Pi sont désormais en mesure de réaliser leurs propres systèmes de commande et de régulation en utilisant leur propre matériel. Le matériel et le logiciel sont également excellents à des fins de formation, car ils respectent la norme CEI.
Les débutants apprendront également tout sur l'installation et la programmation dans les cinq langages de programmation afin de construire leurs propres systèmes de commande. Un chapitre aborde le suivi des processus à l'écran avec AdvancedHMI. Sont également expliqués les circuits avec l'Arduino et l'ESP8266, nécessaires pour le Modbus.

Après vous être enregistré, téléchargez le programme gratuitement et décompressez-le dans un répertoire (**fig. 3**). Enregistrez le fichier ZIP comme modèle.
Ensuite, créez un répertoire *AdvancedHMI* et copiez-y le contenu du fichier *AdvancedHMIv399xR1.ZIP* (**fig. 4**). Pour ouvrir le projet, un environnement de développement est nécessaire, alors installez le logiciel *Visual Studio Community* à partir du site web de Microsoft, **https://visualstudio.microsoft.com/vs/community/**, comme le montre la **figure 5**.
Une fois le processus d'installation terminé, démarrons le premier projet de démonstration. Avec un clic droit de la souris, nous ouvrons la solution *AdvancedHMI* avec Microsoft Visual Studio 2019 (**figures 6** et **7**). En double-cliquant sur *MainForm.vb*, vous obtenez la vue HMI encore vide de votre projet de démonstration (WinForm). Vous pouvez maintenant exécuter le programme en cliquant sur *Start* (**fig. 8**). À ce stade, le projet a été compilé et enregistré dans le répertoire indiqué à la **figure 9**.
Ces fichiers de projet sont nécessaires pour utiliser un programme sur un autre ordinateur. Vous pouvez également exécuter ces fichiers sur le Raspberry Pi avec le Mono Framework.
Maintenant, faites glisser et déposez le composant *ModbusTCPCom* de la boîte à outils de gauche (**fig. 10**) dans votre espace de travail. Dans la fenêtre *Properties*, vous pouvez maintenant saisir l'adresse IP de votre Raspberry Pi (**fig. 11**).
Pour tester la communication, créez un nouveau programme PLC avec l'éditeur Open PLC (**fig. 12**), chargez-le sur le Raspberry Pi et démarrez-le. Il s'agit d'un simple circuit On/Off utilisant les boutons *On* et *Off* (**fig. 13**).
La situation devient intéressante avec le bouton *BTN_HMI*, qui se trouve à l'adresse *%QX2.0*. Cette adresse, située en dehors de la plage d'adresses des GPIO du Raspberry Pi, est nécessaire pour que l'interface HMI déclenche une pression sur un bouton. Avec les temporisations *TON0* et *TOF0*, vous pouvez réaliser un circuit de LED clignotante pour le programme *LEDBlink*.
Revenez maintenant à votre projet HMI et faites glisser un interrupteur (*MomentaryButton*) et trois voyants lumineux (*PilotLight*) depuis la boîte à outils, et étiquetez-les comme vous pouvez le voir dans le résultat final, **figure 14**. Pour ce faire, sélectionnez l'objet et modifiez le texte dans la fenêtre *Properties* comme indiqué.
Définissez les champs *PLCAdressClick* comme suit :

- **BTN_HMI** est à l'adresse 17, parce que vous avez utilisé *%QX2.0* pour le *BTN_HMI* (8+8+1).
- La **LED** va à **1**
- **LEDModbus** va à **2**
- **LEDBlink** va à **3**

Le *MomentaryButton* peut être changé de bouton en interrupteur dans les propriétés de *OutputType* comme le montre la **figure 15**. Cela fait, vous disposez des adresses correctes pour avoir accès aux variables de votre programme PLC. Vous pouvez maintenant lancer votre programme et effectuer un test.
Les trois LED sont toujours dans le même état que les LED de la carte de test. Le commutateur *BTN_HMI* vous permet d'allumer la LED 2 de la carte de test.
Le **tableau 1** montre le mappage des adresses Modbus pour le Raspberry Pi avec OpenPLCproject et AdvancedHMI. OpenPLC fournit également un espace d'adressage séparé pour les variables en mémoire avec un support pour les variables de 16, 32 et 64 bits, comme résumé dans le **tableau 2**.
Pour installer Framework Mono sur le Raspberry PI, il faut saisir les commandes suivantes :

```
sudo su
apt-get update
apt-get install mono-complete
apt-get install mono-vbnc
```

Vous pouvez maintenant créer un répertoire sur le Raspberry Pi et utiliser *WinSCP* pour copier votre projet du PC vers le Raspberry Pi et le tester.

Function Code	Usage	PLC Address	Modbus Register	Register Size	Value Range	Access	Advanced HMI	Example
FC01 Read Coil	Digital Outputs	%QX0.0 - %QX99.7	0-799 (1-800)	1 Bit	0 or 1	Read / Write	"00001" – "00800"	LED
FC02 Discrete Input	Digital Inputs	%IX0.0 – %IX99.7	0-799 (1-800)	1 Bit	0 or 1	Read Only	"100001" – "100800"	BTN
FC04 Input Register	Analog Inputs	%IW0 – %IW99	0-1023 (1-1024)	16 Bit	0-65535	Read Only	"30001" – "31024"	ADC
FC03 Holding Register	Analog Outputs	%QW0 – %QW99	0-1023 (1-1024)	16 Bit	0-65535	Read / Write	"40001" – "41024"	DAC

Tableau 1

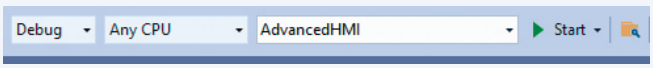


Figure 8. La barre de menu de Visual Studio.

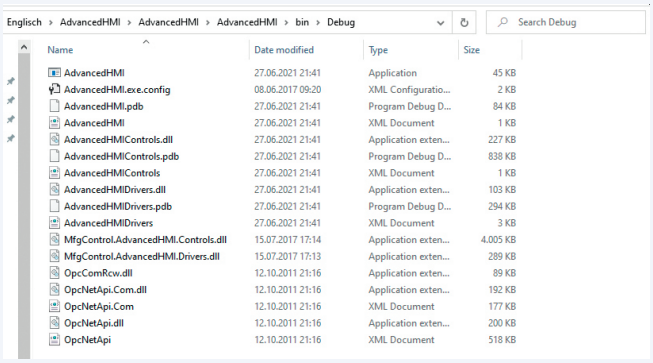


Figure 9. Le répertoire de débogage d'AdvancedHMI.

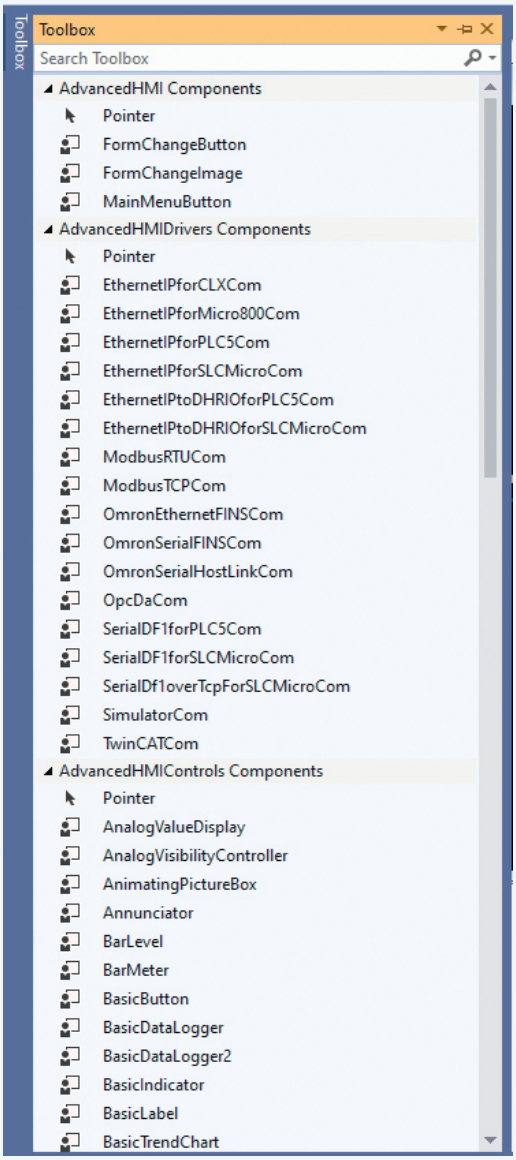


Figure 10. La boîte à outils de Visual Studio.

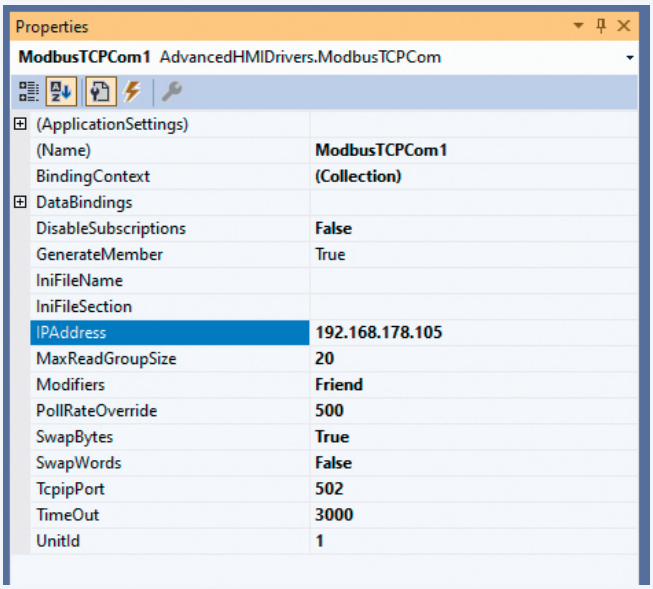


Figure 11. Paramètres Modbus (IP et port).

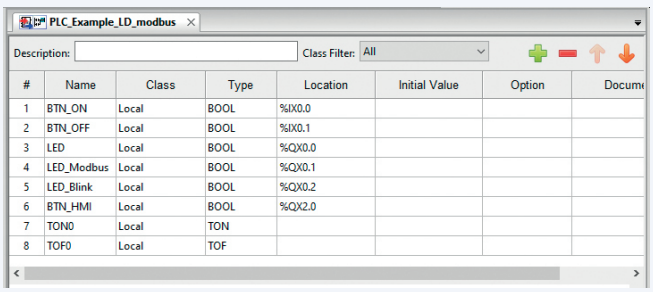


Figure 12. Exemple de liste de variables de l'API.

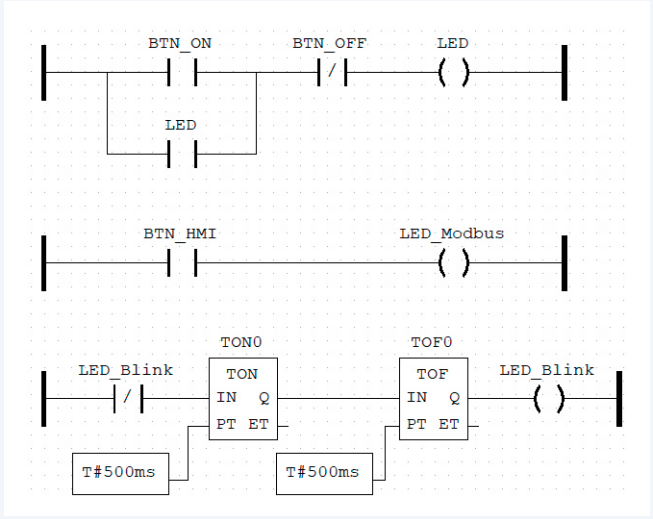


Figure 13. Exemple de programme de LED.

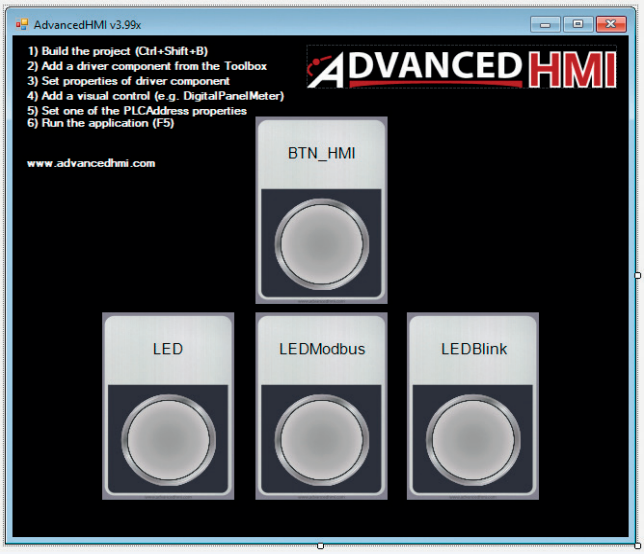


Figure 14. Le programme HMI.

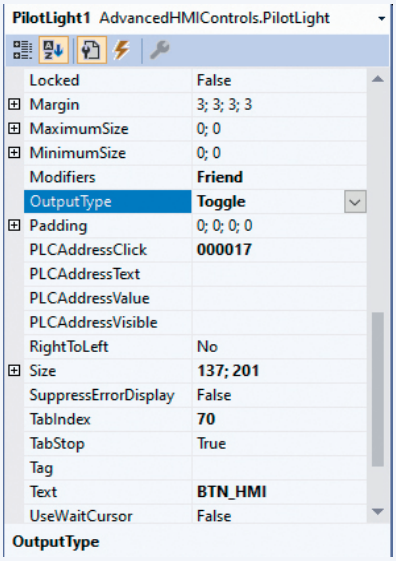


Figure 15. Réglage des paramètres de l'HMI.

L'exemple de programme est aussi disponible dans le répertoire de téléchargement sous *AdvancedHMI*. Les fichiers *.EXE* des exemples traités dans le livre se trouvent dans le sous-répertoire suivant :

`PLC-Book-Download\AdvancedHMI\AdvancedHMI\bin\Debug`

Ce sous-répertoire est contenu dans le paquet de logiciels publié par l'auteur comme ressource de son livre. Le logiciel peut être téléchargé gratuitement. Rendez-vous sur [1], allez à *Téléchargements* et cliquez sur le nom du fichier : *Software_PLC Programming with the Raspberry Pi and the OpenPLC Project*. Enregistrez le fichier d'archive ZIP (128,94 Mo) en local, puis décompressez-le.

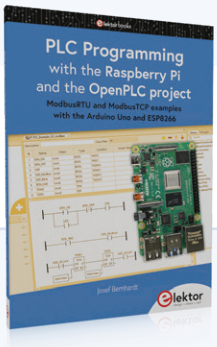
210642-04

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (josef@bernhardt.de) ou contactez Elektor (redaction@elektor.fr).

Contributeurs

Texte : **Josef Bernhardt**
Rédaction : **Jan Buiting**
Mise en page : **Giel Dols**
Traduction : **Helmut Müller**



PRODUITS

- **Livre en anglais, « PLC Programming with the Raspberry Pi and the OpenPLC Project », J. Bernhard**

  Version papier : www.elektor.fr/19966

  Version numérique : www.elektor.fr/19967

Register Type	Usage	PLC Address Range	Modbus Address	Register Size	Value Range	Access	Advanced HMI
Holding Register	General 16Bit Register	%MW0 - %MW1023	1024..2047 (1025-2048)	16 Bit	0-65535	Read / Write	"41025" – "42048"
Holding Register	General 32Bit Register	%MD0 - %MD1023	2048..4095 (2049-4096)	32 Bit	0-4.294.967.295	Read / Write	"42049" – "44096"
Holding Register	General 64Bit Register	%ML0 - %ML1023	4096..8191 (4097-8192)	64 Bit	0-huge	Read / Write	"44097" – "48192"

Tableau 2

LIEN

[1] Ressources du livre/page d'information : www.elektor.fr/plc-programming-with-the-raspberry-pi-and-the-openplc-project