

Amélioration d'un logiciel d'une valise de test

Improvement of a test case software

Rapport Projet 1 **CPE Lyon – Informatique et Réseaux de Communications**

2021 – 2022

Promo 2021 - 2024



Ensto Novexia
210 Rue Léon Jouhaux
69400 Villefranche-sur-Saône

Informatique Embarquée | Informatique Industrielle | Programmation
Langage C | Microcontrôleur

Tuteur Entreprise : M Bertrand FAVEL
Tuteur Pédagogique : M Serge MAZAURIC

Batiste LALOI | Année 2021 - 2022 | CPE Lyon – 3IRC

Remerciements

Je tiens en premier lieu à remercier la société Ensto Novexia, qui m'a fait confiance en m'accueillant durant ces premiers mois d'alternance qui furent une expérience nouvelle et enrichissante. Le cadre et l'environnement de travail étaient agréables et conviviaux, notamment la gentillesse et la bienveillance des collaborateurs avec qui j'ai eu la chance d'échanger et de travailler.

Je tiens à remercier M Anthony JAMBON, pour toute l'aide qu'il m'a apporté sur le fonctionnement du produit sur lequel j'ai eu à travailler pendant mon alternance, puisqu'il en était le précédent développeur.

Je tiens aussi à remercier M Bertrand FAVEL, qui fut la première personne que j'ai rencontrée lors de mon entretien en mars 2021, et qui m'accompagnera tout au long de mon alternance. Il fut la principale personne à me donner des directives à suivre, que ce soit de face à l'atelier, ou bien lors de réunion et points réguliers.

Je tiens à remercier l'ensemble des membres du Bureau d'Etudes en Electronique du site de Villefranche-sur-Saône.

Table des matières

Remerciements.....	3
Table des matières.....	4
Table des illustrations	5
Lexique.....	6
Introduction	7
Présentation de l'entreprise	7
1- Historique et produits de l'entreprise	7
2- Organisation de l'entreprise	8
Description de la mission : Objectifs, Contraintes et Enjeux	10
Développement.....	12
1- Environnement.....	12
a. Technologies	12
b. Architecture.....	14
c. Intégration continue.....	16
2- Mise en place des solutions	17
a. Mise en place d'une limite dans le choix des modèles à tester.....	17
b. Ajout un menu de changement de la langue.....	19
c. Création d'une fonction de remise à zéro de la valise	24
d. Correction fonction injection défaut homopolaire	26
Bilan de la mission et perspective d'avenir	29
Annexes	30

Table des illustrations

Figure 1: Coffret de contrôle commande ITI/PASA HTA/BT.....	7
Figure 2: Valise Ensto VISIO II Testeur de défauts directionnel ou ampèremétrique.....	7
Figure 3: Organigramme DSO France	8
Figure 4: Organigramme BE Electronique	9
Figure 5: Valise Vision - 2005.....	10
Figure 6: Valise VISIO II - 2020.....	10
Figure 7: Module J-link IAR System	12
Figure 8: Outil Livewatch IAR	12
Figure 9: Editeur de texte Visual Studio Code.....	13
Figure 10: Extrait du fichier Afficheur_export.c	14
Figure 11: IHM de la valise	15
Figure 12: Liste des modèles de coffrets OMT et ILD de chez Ensto Novexia.....	17
Figure 13: Nouvelle liste de modèles de coffrets.....	18
Figure 14: Menu de sélection des modèles de coffrets	18
Figure 15: Code permettant d'envoyer à l'écran les lignes du menu de sélection de langue	19
Figure 16: Code de mise à jour de l'écran LCD	19
Figure 17: Déclaration des constantes de langue EN, FR et ES	20
Figure 18: Code permettant d'afficher un curseur de sélection	20
Figure 19: Code permettant de déplacer le curseur lorsque l'utilisateur presse les flèches directionnelles.....	21
Figure 20: Code de démarrage de la sauvegarde des paramètres de langue.....	22
Figure 21: Gestion des fonctions d'affichage en anglais, français ou espagnol	22
Figure 22: Code permettant d'afficher un menu de la valise, en anglais, français et espagnol	23
Figure 23: Liste des constantes par défaut des variables du programme de la valise.....	24
Figure 24: Extrait de la fonction Init_conf_usine().....	24
Figure 25: Menu de validation de la réinitialisation des paramètres d'usine	25
Figure 26: Photo de l'oscilloscope lors des tests défailants d'injection de défauts homopolaires	26
Figure 27: Code gérant l'injection de défauts homopolaires	27
Figure 28: Photo de l'oscilloscope lors des tests réussis d'injection de défauts homopolaires.....	28

Lexique

- BE : Bureau d'études
- μ C : Microcontrôleur
- IHM : Interface Homme Machine
- SAV : Service Après-Vente
- VS Code : Visual Studio Code
- BT : Basse Tension
- DDA : Détection de Défaut Ampèremétrique
- OMT : Organe de Manœuvre Télécommandée
- RAZ : Remise A Zéro
- BP : Bouton poussoir

Introduction

Présentation de l'entreprise

1- Historique et produits de l'entreprise

Novexia est une entreprise fondée en 1925 à Villefranche-sur-Saône en région Lyonnaise. En 2010, la société est rachetée par le groupe Finlandais Ensto. Elle compte aujourd'hui plus de 170 employés, dont 105 au niveau du site de Villefranche et avait en 2019 un chiffre d'affaires de 45 millions d'euros. Issu de la fusion en 2000 de l'entreprise Simplex et une branche de la société Soulé, Ensto Novexia, anciennement Novexia, est une entreprise spécialisée dans le développement de solutions sécurisées d'automatisation des réseaux intelligents de distribution électrique.

Parmi les différents produits proposés par la société, voici ceux sur lesquels j'ai travaillé :



Figure 1: Coffret de contrôle commande
ITI/PASA HTA/BT



Figure 2: Valise Ensto VISIO II
Testeur de défauts directionnel
ou ampèremétrie

La valise VISIO II déployée en France en 2020 est un appareil de dernière génération permettant de simuler et d'injecter des défauts, afin de tester des coffrets de contrôle commande d'interrupteurs HTA, comme le coffret ITI ([Figure 1](#)). Tout en étant mobile, elle permet au technicien d'intervenir sur les différents modèles de coffrets, et ce malgré les différents constructeurs présents sur le marché, tels que Schneider, Cahors et Areva.

2- Organisation de l'entreprise

Le site de Villefranche rassemblant plus d'une centaine d'employés, l'apparition et la mise en place d'un bureau d'études et de plus petites entités de recherches permet aux différentes personnes de travailler dans des conditions optimales, tandis que l'environnement reste aussi convivial que nous le permet la crise sanitaire que nous vivons depuis plus d'un an maintenant. Par ailleurs, je n'ai pas effectué de télétravail durant ma première année d'alternance, cela m'a permis d'avoir de bons contacts humains, essentiels pour la bonne cohésion d'une équipe de développement.

Organigramme de l'entreprise Ensto Novexia, DSO France :

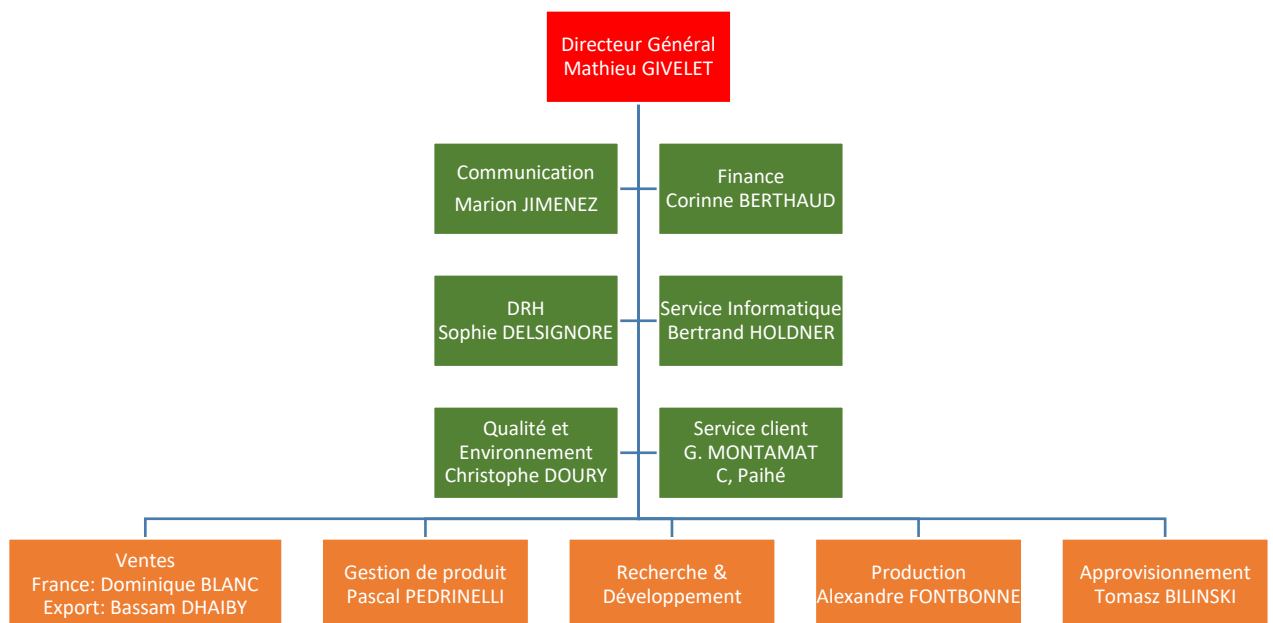


Figure 3: Organigramme DSO France

Organigramme des deux bureaux d'études de Ensto Novexia :

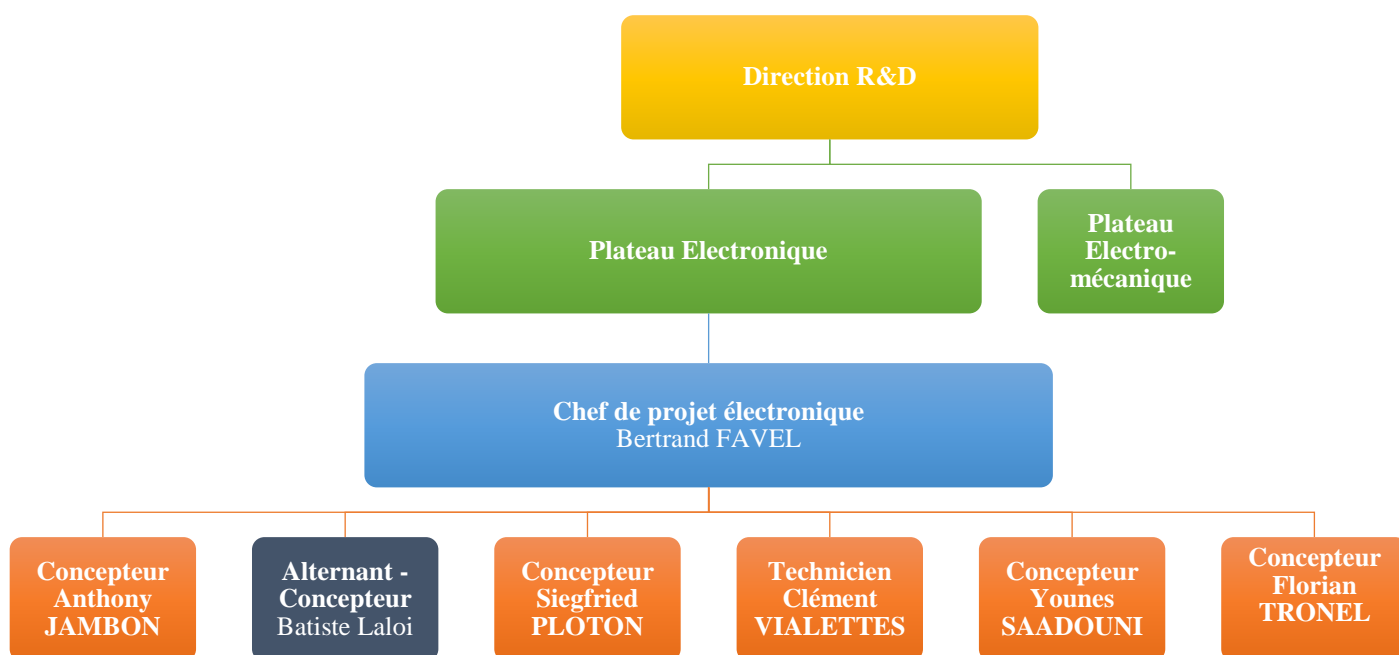


Figure 4: Organigramme BE Electronique

Description de la mission : Objectifs, Contraintes et Enjeux

Le projet du développement de la valise VISIO II Export s'inscrit dans la continuité du projet de développement de la valise VISIO II. Cette dernière possède déjà une version « France » en vente, qui est un produit matériel permettant de tester les fonctions de détections de défauts des coffrets de contrôle-commandes présents sur le réseau électrique français, et la version « Export » vise le marché international, et plus précisément africain et hispanique. Ce produit permet d'effectuer des injections de courant et/ou de tension en se raccordant aux coffrets, permettant de vérifier le bon fonctionnement de ces derniers.

○ Contexte de la mission :

La valise VISIO II est un produit développé par Ensto depuis 2019, elle fait suite au modèle précédent : la valise VISION. Cet appareil vieillissant ne permettait pas par exemple de modifier les niveaux d'injection.



Figure 5: Valise Vision - 2005



Figure 6: Valise VISIO II - 2020

Après plus d'un an de mise en service, de nombreux retours clients sont remontés auprès des membres du SAV. Voici la liste des différents défauts de la valise évoqués par les clients lors de son utilisation sur le terrain :

- La navigation dans le menu de configuration est peu intuitive, les flèches directionnelles étant inversées.
- Lors d'une injection d'un défaut DDA, une temporisation d'absence de BT durant 3 secondes advient. Cette valeur de temporisation n'est pas configurable.
- Des soucis de compréhension apparaissent lors de la mise en charge de la valise : l'appareil doit être allumé pour que la charge soit effective.

Par ailleurs, un nouveau modèle de coffret : EMIS (Annexe n°2), apparaît sur le marché, et la valise VISIO II doit être capable de tester ces modèles d'appareils. Il a donc fallu ajouter ce modèle de coffret à la valise.

Tous ces ajouts et modification de fonctionnalités furent effectuées sur la version « France » de la VISIO II lors d'un précédent stage, et c'est sur cette version que se base le développement de la version « Export » de la VISIO II.

Les différentes fonctionnalités à ajouter / modifier pour créer la version « Export » sont :

- **Limiter le choix du matériel à tester aux coffrets type ITI et LYNX proposés par Ensto Novexia**
- **La possibilité de changer la langue de l'interface de la valise**
- **Pouvoir effectuer une remise à zéro de la configuration de la valise**

Pendant l'année, une problématique liée à une fonctionnalité de la version « France » advint, dont la solution et sa mise en place sera aussi détaillée dans ce rapport, puisqu'elle influence une fonctionnalité de la version « Export ». En effet, l'utilisation de la VISIO II France sur le modèle de coffret EMIS de chez Cahors était instable, en particulier lors de l'injection de défaut de tension homopolaires (ou défaut à la terre, impliquant un court-circuit entre un conducteur et la terre). On ajout donc cette tâche à la liste :

- **Correction de la fonction d'injection de défaut directionnel homopolaire**

Développement

1- Environnement

a. Technologies

Le projet fut développé par Mr JAMBON sur un microcontrôleur STM32 de chez STMicroelectronics, grâce au langage de programmation bas niveau C, permettant un contrôle quasi-total des composants raccordés au μ C, comme les composants générateurs de courant/tension.

La compilation et l'exécution des programmes se font par le biais d'un logiciel propriétaire nommé IAR Embedded Workbench (Annexe n°1), couplé à une clé de licence permettant son exploitation. Le microcontrôleur doit être raccordé à l'ordinateur grâce au *J-Link* :



Figure 7: Module J-link IAR System

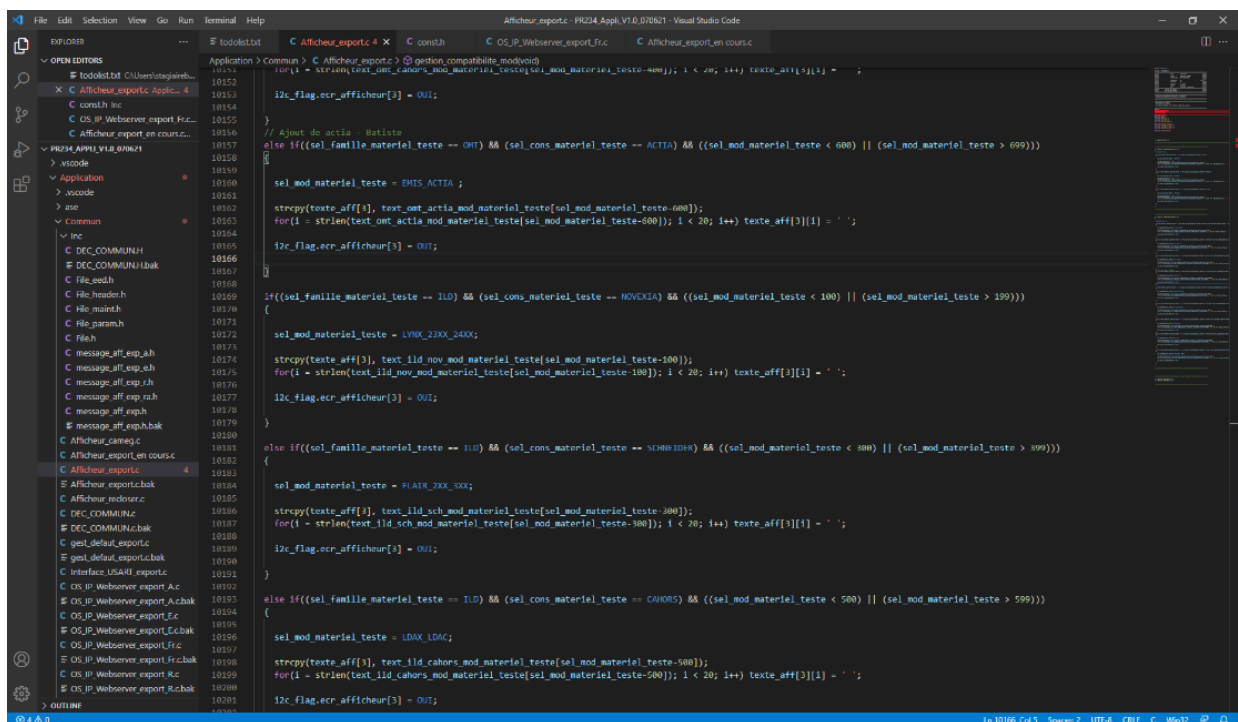
Une fois le programme téléversé, on peut accéder à une interface de débogage, nommé *Livewatch* :

Live Watch			
Expression	Value	Location	Type
⊞ texte_tempo_abs_bt_dda_modif	""	0x2001846A	char[2]
tempo_abs_bt_dda_modif	500	0x20018422	unsigned short
indice_digit	'0' (0x00)	0x20018484	signed char
indice_l1	'0' (0x01)	0x20018531	unsigned char
indice_l2	'0' (0x00)	0x20018532	unsigned char
indice_l3	'0' (0x00)	0x20018533	unsigned char
indice_l4	'0' (0x00)	0x20018534	unsigned char
tempo_abs_bt_dda_poly	5000	0x200182C0	unsigned long
tempo_abs_bt_dda_doub	5000	0x200182CC	unsigned long
tempo_abs_bt_dda_homo	5000	0x200182D8	unsigned long
cpt_tempo_abs_bt_dda_poly	0	0x200182C8	unsigned long
cpt_tempo_abs_bt_dda_doub	0	0x200182D4	unsigned long
cpt_tempo_abs_bt_dda_homo	0	0x200182E0	unsigned long
⊞ texte_U_reseau	""	0x20018468	char[2]
U_reseau	'0' (0x14)	0x200184CA	unsigned char
sel_cons_materiel_teste	0	0x20018464	short
sel_mod_materiel_teste	0	0x20018466	short
gain_eta_gene_i	-1.32920003	0x20018278	float
gain_eta_gene_perm_i	-1.0	0x20000198	float
amp_conf_v_reseau	183	0x2001843C	unsigned short

Figure 8: Outil Livewatch IAR

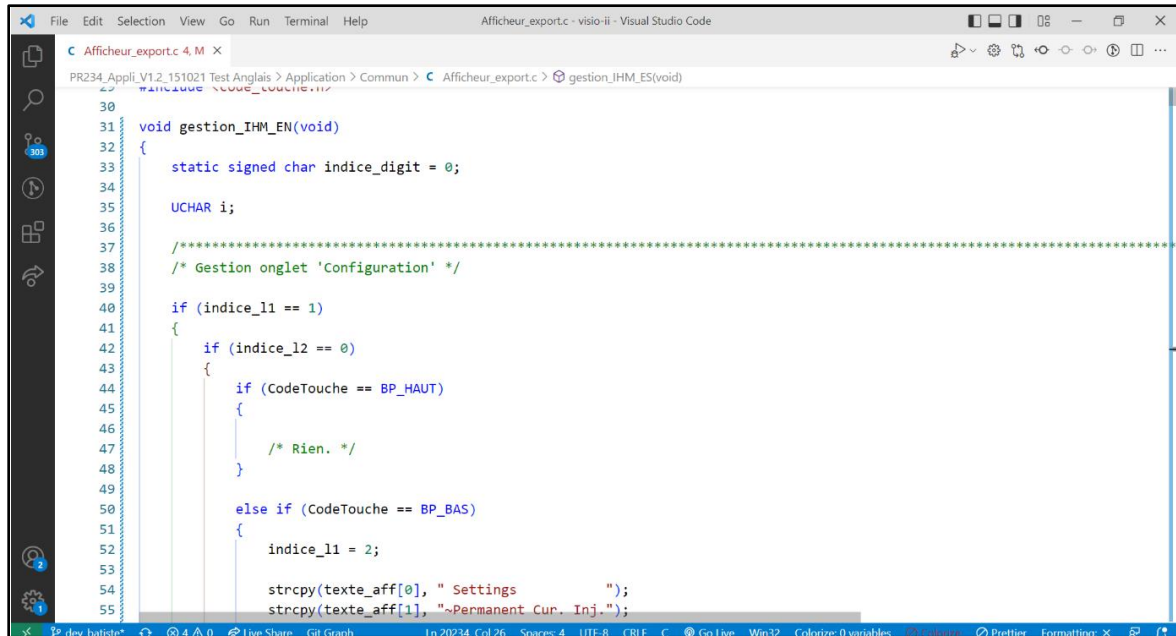
En entrant le nom d'une variable du programme, on peut visualiser sa valeur, son type, son emplacement dans la mémoire. S'il s'agit d'un tableau, on a aussi accès à chacune des cases de ce tableau. La force de cet outil est le fait que l'on puisse modifier les valeurs pendant que le programme est en cours d'exécution sur la carte, cela permet d'effectuer toute sorte de tests sans avoir à recompiler la totalité du projet. Il est fortement conseillé de s'en servir pour faciliter l'exploitation des programmes.

Pour pouvoir développer du code en C, on peut utiliser le logiciel IAR, ou bien n'importe quel autre éditeur de texte, spécialisé ou non dans l'édition de programmes informatiques. Visual Studio Code, un outil développé par Microsoft et très grandement adopté par la communauté des développeurs à l'international permet de nombreuses fonctionnalités qui facilitent la vie de la personne qui souhaite écrire du code, comme de l'auto-complétion, de renommage multiple, et une coloration syntaxique très développée permettant une meilleure compréhension lors de la lecture des programmes. La plupart des captures d'écrans de lignes de code montrées dans ce rapport seront effectuées sur VS Code.



b. Architecture

Le programme de l'écran positionné sur l'interface homme machine se situe dans le fichier *Afficheur_export.c*, dont voici un extrait :



```
30
31 void gestion_IHM_EN(void)
32 {
33     static signed char indice_digit = 0;
34
35     UCHAR i;
36
37     /******
38     /* Gestion onglet 'Configuration' */
39
40     if (indice_l1 == 1)
41     {
42         if (indice_l2 == 0)
43         {
44             if (CodeTouche == BP_HAUT)
45             {
46
47                 /* Rien. */
48             }
49
50             else if (CodeTouche == BP_BAS)
51             {
52                 indice_l1 = 2;
53
54                 strcpy(texte_aff[0], " Settings");
55                 strcpy(texte_aff[1], "~Permanent Cur. Inj.");
```

Figure 10: Extrait du fichier *Afficheur_export.c*

L'afficheur est géré par l'une de ces trois routines, en fonction de la langue choisie par l'utilisateur :

- gestion_IHM_EN()
- gestion_IHM_FR()
- gestion_IHM_ES()

Elles se présentent chacune sous la forme de conditions IF imbriquées les unes dans les autres, permettant un acheminement du code à exécuter grâce aux variables indice_IX, X représentant le numéro du menu dans lequel l'utilisateur est actuellement.

Ces différentes variables de navigation sont modifiées lors de l'utilisation de la valise via les boutons poussoirs situés en dessous de l'écran de la valise (encadrés en rouge) :

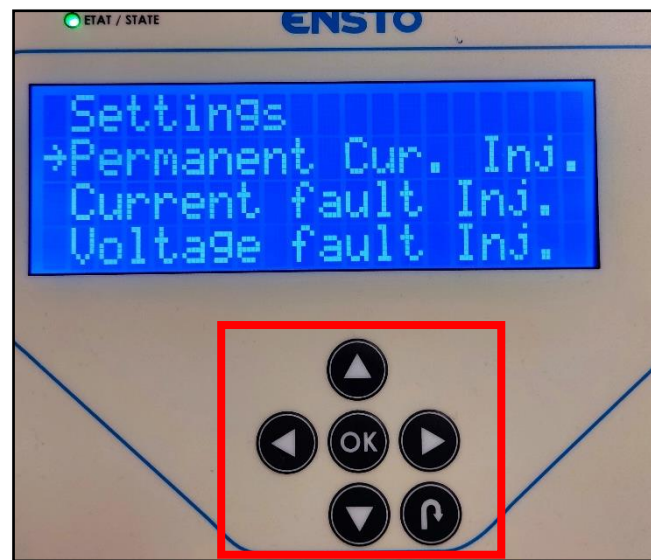


Figure 11: IHM de la valise

c. Intégration continue

Le système de versioning du projet est basé sur un dépôt GitLab, utilisant donc la technologie Git, dont voici les différentes branches et acteurs principaux du projet :

Structure :

- prod_export : Branche de production de la version valise VISIO II Export
 - Contient uniquement les fichiers .bin qui sont téléversés dans les cartes lors de la production des valises à l'atelier
 - Toutes les versions sont identifiables sous le format PRXXX_VX.X.bin en fonction de la version en cours de production
- prod_france : Branche de production de la valise VISIO II France
 - Contient uniquement les fichiers .bin qui sont téléversés dans les cartes lors de la production des valises à l'atelier
 - Toutes les versions sont identifiables sous le format PR234_VX.X.bin en fonction de la version en cours de production
- dev_verif : Branche versions à tester
 - Contient tous les fichiers permettant de coder sur le projet, mais aussi de le compiler localement pour effectuer des essais
- dev_batiste : Branche de développement personnel
 - Contient tous les fichiers en l'état actuel de développement de Batiste LALOI

Développeurs :

- Batiste LALOI : Stagiaire puis alternant depuis avril 2021

Manager :

- Bertrand FAVEL : Responsable et chef de projet au sein du bureau d'étude

2- Mise en place des solutions

a. Mise en place d'une limite dans le choix des modèles à tester

Les différents modèles de coffrets disponibles sur la valise sont stockés sous la forme de constantes dans le fichier *const.h* :

```
428
429  /***** Liste des modèles des matériels testés par la valise. *****/
430
431  /* OMT NOVEXIA *****/
432  #define NBR_OMT_NOVEXIA      8
433  #define ITI_90_93_96        0
434  #define SD_2000             1
435  #define ITI_2000_X          2
436  #define ITI_2001_X          3
437  #define ITI_2012_X          4
438  #define AUGUSTE_02_12       5
439  #define AUGUSTE_12_18       6
440  #define AUGUSTE_AP_18       7
441
442  /* ILD NOVEXIA */
443  #define NBR_ILD_NOVEXIA      3
444  #define LYNX_33XX            101
445  #define LYNX_34XX            102
446  #define LYNX_23XX_24XX      100
447  /*****/
```

Figure 12: Liste des modèles de coffrets OMT et ILD de chez Ensto Novexia

La constante *NBR_OMT_NOVEXIA* permet au code de boucler dans les différentes boucle FOR de sélection des modèles de coffret parmi les coffrets listés en dessous.

Voici un récapitulatif des différents coffrets disponibles actuellement :

Modèles de coffret version FRANCE	Version EXPORT
ITI_90_93_96	OUI
SD_2000	NON
ITI_2000_x	OUI
ITI_2001_x	OUI
ITI_2012_x	OUI
AUGUSTE_02_12	NON
AUGUSTE_12_18	NON
AUGUSTE_AP_18	NON
LYNX_33XX	OUI
LYNX_34XX	OUI
LYNX_23XX_24XX	OUI

Il suffit de modifier la valeur correspondant au nombre de modèles type OMT, ainsi que passer les 4 coffrets à garder pour permettre à la valise de n'afficher que ceux-ci :

```

434
435 /***** Liste des modèles des matériels testés par la valise. *****/
436
437 /* OMT NOVEXIA *****/
438 #define NBR_OMT_NOVEXIA          4
439 #define ITI_90_93_96             0
440 #define ITI_2000_X               1
441 #define ITI_2001_X               2
442 #define ITI_2012_X               3
443
444
445 /* ILD NOVEXIA */
446 #define NBR_ILD_NOVEXIA          3
447 #define LYNX_23XX_24XX           100
448 #define LYNX_33XX                101
449 #define LYNX_34XX                102
450 // Ajouter le LYNX 4400
451 /*****

```

Figure 13: Nouvelle liste de modèles de coffrets

Voici maintenant le menu de sélection des modèles de coffrets affiché sur la valise, dorénavant limité à la sélection des modèles souhaités :

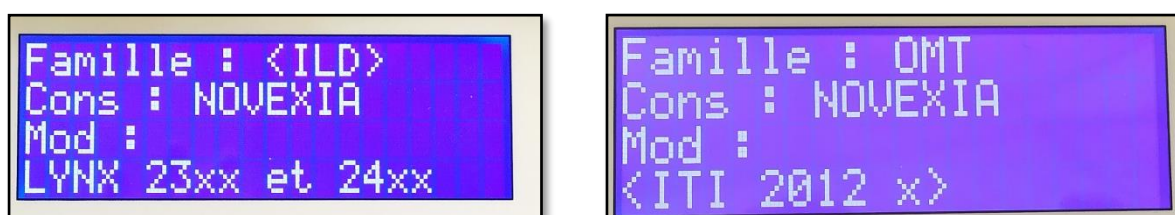


Figure 14: Menu de sélection des modèles de coffrets

b. Ajout un menu de changement de la langue

Pour ajouter un menu de configuration de la langue dans les sous-menus de configuration de la valise, il faut tout d'abord savoir à quoi il ressemblera visuellement.

Voici la structure choisie :

```
strcpy(texte_aff[0], "Choix langue :      ");
strcpy(texte_aff[1], " English      ");
strcpy(texte_aff[2], " Francais     ");
strcpy(texte_aff[3], " Espanol      ");
```

Figure 15: Code permettant d'envoyer à l'écran les lignes du menu de sélection de langue

On utilise la fonction `strcpy`, qui permet de copier le contenu d'une chaîne de caractères dans une variable. Ici on copie le contenu de chaque ligne du menu de sélection dans un tableau `texte_aff` qui contient, pour chaque case, la ligne actuellement stockée dans les fonctions qui gèrent le contenu de l'écran. Cet écran est un écran alphanumérique LCD de 4 lignes de 20 caractères chacune. On peut y stocker n'importe quelle chaîne, mais il faut toujours actualiser des FLAG qui permettent individuellement de rafraîchir le contenu des lignes de l'écran, grâce à une liaison i2c entre le μ C de la valise et l'écran :

```
i2c_flag.ecr_afficheur[0] = OUI;
i2c_flag.ecr_afficheur[1] = OUI;
i2c_flag.ecr_afficheur[2] = OUI;
i2c_flag.ecr_afficheur[3] = OUI;
```

Figure 16: Code de mise à jour de l'écran LCD

Ici les 4 lignes de l'afficheur sont actualisées.

Pour savoir à quelle ligne l'utilisateur se situe actuellement on utilise le caractère « ~ » qui est retranscrit sur l'afficheur par une flèche « → », jouant le rôle d'un curseur de sélection.

Voici comment la flèche est ajoutée dans ce menu (le processus est similaire pour tous les autres menu) :

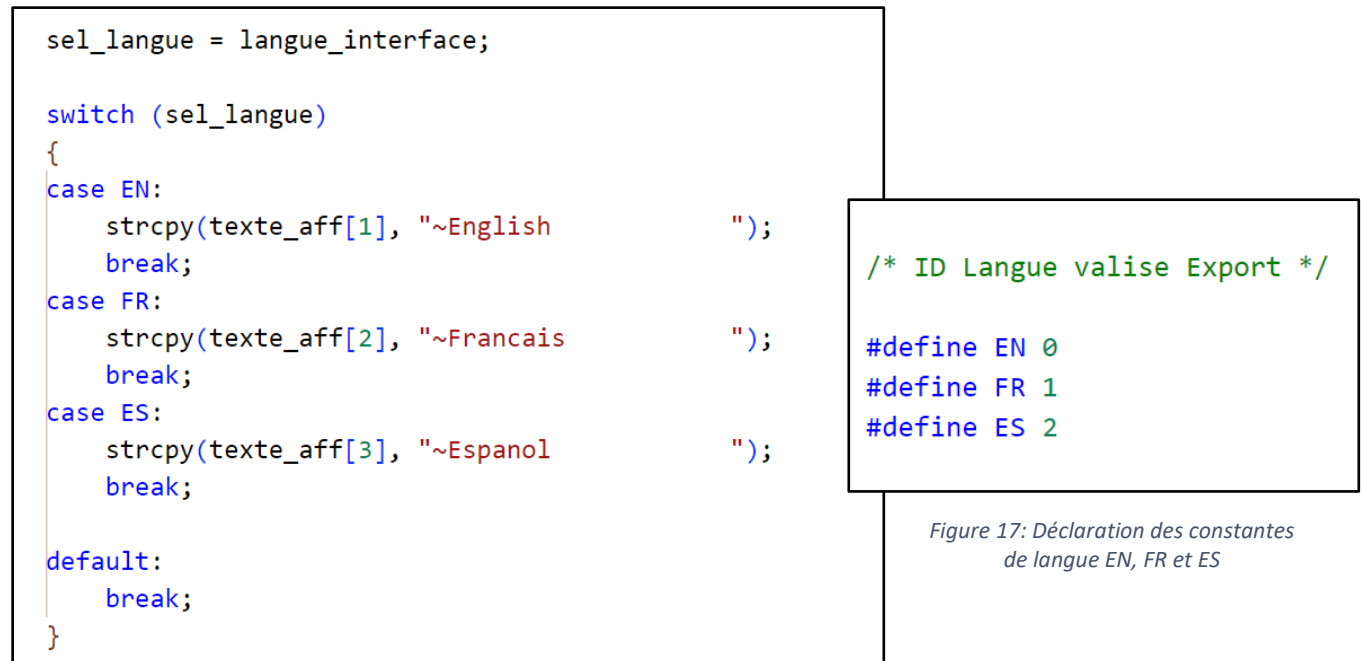


Figure 18: Code permettant d'afficher un curseur de sélection

La variable *sel_langue* prends la valeur de la langue actuelle de l'interface, cette dernière étant actualisée lorsque l'on appuie sur le bouton OK, permettant la sauvegarde des paramètres sélectionnés. En fonction de la valeur de *sel_langue* (0, 1, 2), qui représente le numéro de ligne actuellement sélectionnée, on remplace la ligne possédant le caractère de sélection avant l'actualisation de l'écran LCD.

Le code gère le déplacement du curseur de sélection lorsque les boutons poussoirs correspondant aux directions HAUT et BAS sont pressés, en incrémentant la variable *sel_langue*, sans pour autant modifier la langue de l'interface actuellement chargée et affichée.

Voici par exemple, la gestion de l'action sur le bouton poussoir BAS :

```
else if (CodeTouche == BP_BAS)
{
    switch (sel_langue)
    {
    case EN:
        sel_langue++;

        strcpy(texte_aff[1], " English");
        strcpy(texte_aff[2], "~Francais");

        i2c_flag.ecr_afficheur[1] = OUI;
        i2c_flag.ecr_afficheur[2] = OUI;
        break;

    case FR:
        sel_langue++;

        strcpy(texte_aff[2], " Francais");
        strcpy(texte_aff[3], "~Espanol");

        i2c_flag.ecr_afficheur[2] = OUI;
        i2c_flag.ecr_afficheur[3] = OUI;
        break;

    case ES:
        // On ne fait rien car on est déjà en bas de l'écran
        break;

    default:
        break;
    }
}
```

Figure 19: Code permettant de déplacer le curseur lorsque l'utilisateur presse les flèches directionnelles

Si l'utilisateur souhaite sauvegarder la langue sélectionnée, il appuie sur le bouton poussoir OK ou bien la flèche de droite « → », ce qui déclenche la fonction de sauvegarde, et le passage de l'interface dans la langue sélectionnée :

```
else if ((CodeTouche == BP_DROIT) || (CodeTouche == BP_OK))
{
    langue_interface = sel_langue;

    lance_tempo_sauv_conf = OUI;

    strcpy(texte_aff[0], "ENREGISTREMENT OK");
    strcpy(texte_aff[1], " ");
    strcpy(texte_aff[2], " ");
    strcpy(texte_aff[3], " ");

    i2c_flag.ecr_afficheur[0] = OUI;
    i2c_flag.ecr_afficheur[1] = OUI;
    i2c_flag.ecr_afficheur[2] = OUI;
    i2c_flag.ecr_afficheur[3] = OUI;

    i2c_flag.ecr_eep[0] = OUI;
}
```

Figure 20: Code de démarrage de la sauvegarde des paramètres de langue

S'exécute alors la routine de sauvegarde par le passage d'un FLAG de démarrage d'une temporisation de sauvegarde de la temporisation. Il déclenche une attente de 3 secondes, durant laquelle s'affiche le message « ENREGISTREMENT OK » sur l'écran. La valeur de la variable *sel_langue* est écrite dans *langue_interface*, ce qui permet au programme de connaître quelle fonction d'affichage est à exécuter :

```
switch (langue_interface)
{
    case EN:
        gestion_IHM_EN();
        break;

    case FR:
        gestion_IHM_FR();
        break;

    case ES:
        gestion_IHM_ES();
        break;

    default:
        break;
}
```

Figure 21: Gestion des fonctions d'affichage en anglais, français ou espagnol

Ces trois fonctions sont identiques du point de vue de leur fonctionnement logique, mais elles affichent des chaînes de caractère différente sur l'écran, en fonction de la langue sélectionnée :

```
strcpy(texte_aff[0], "~Settings      ");
strcpy(texte_aff[1], " Permanent Cur. Inj.");
strcpy(texte_aff[2], " Current fault Inj. ");
strcpy(texte_aff[3], " Voltage fault Inj. ");

strcpy(texte_aff[0], "~Configuration  ");
strcpy(texte_aff[1], " Inject I permanent ");
strcpy(texte_aff[2], " Inject def.Amp(DDA)");
strcpy(texte_aff[3], " Inject def.Dir(DDD)");

strcpy(texte_aff[0], "~Configuracion   ");
strcpy(texte_aff[1], " Inyeccion I cont.  ");
strcpy(texte_aff[2], " Inyeccion de corr. ");
strcpy(texte_aff[3], " Inyeccion de tens. ");
```

Figure 22: Code permettant d'afficher un menu de la valise, en anglais, français et espagnol

(Menu de démarrage ici)

c. Création d'une fonction de remise à zéro de la valise

L'utilisateur peut avoir envie de réinitialiser la configuration de la valise, afin de repartir sur de bonnes bases sans aucune modification de valeurs d'injection ou de temporisation.

Pour ce faire, ont été déclarées dans le fichier *const.h* des constantes de valeurs « par défauts » de toutes les variables qui sont initialisées lors de la mise en route de la valise :

```
/* VALEURS PAR DEFAUT */
// Utilisation de ces valeurs lors de la RAZ en valeur d'usine

// Variables par défaut de configurations

#define DEF_LANGUE EN

#define DEF_AMP_CONF_V_RESEAU 183

#define TEMPO_INIT_CONF_USINE 5000

#define DEF_FAMILLE_MATERIEL_TESTE ILD
#define DEF_CONS_MATERIEL_TESTE NOVEXIA
#define DEF_MOD_MATERIEL_TESTE LYNX_23XX_24XX

#define DEF_RATIO_TI 500

#define DEF_U_RESEAU 20

// Variables par défaut injection permanent de courant

#define DEF_AMP_CONF_INJ_PERM_I 100
#define DEF_TEMPO_CONF_INJ_PERM_I 100

#define DEF_LANCE_TEST_INJ_PERM_I NON
#define DEF_FIN_TEMPO_INJ_PERM_I NON
#define DEF_FIN_TEST_INJ_PERM_I NON
```

Figure 23: Liste des constantes par défaut des variables du programme de la valise

Ces différentes constantes sont utilisées dans une fonction de remise à zéro, nommée *Init_conf_usine()* et codée dans le fichier *Init_reg.c*, qui contient toutes les fonctions d'initialisations des registres bas niveaux du microcontrôleur, comme les config des interruptions ou des paramètres de la liaison i2c avec l'afficheur. Elle reprend les valeurs des constantes établies précédemment :

```
void Init_conf_usine(void) // 09/2021 Batiste
{
    // Langue

    langue_interface = DEF_LANGUE;

    amp_conf_v_reseau = DEF_AMP_CONF_V_RESEAU;

    famille_materiel_teste = DEF_FAMILLE_MATERIEL_TESTE;
    cons_materiel_teste = DEF_CONS_MATERIEL_TESTE;
    mod_materiel_teste = DEF_MOD_MATERIEL_TESTE;

    ratio_TI = DEF_RATIO_TI;
```

Figure 24: Extrait de la fonction *Init_conf_usine()*

Le menu permettant d'accéder à la remise à zéro se trouve dans **Configuration → Init config usine**, et possède aussi son équivalent dans les autres langues de la valise, dont voici l'implémentation :

```
strcpy(texte_aff[0], "Bouton Vert ->      ");  
strcpy(texte_aff[1], "Validation init      ");  
strcpy(texte_aff[2], "Configuration usine ");  
strcpy(texte_aff[3], "                ");
```

Figure 25: Menu de validation de la réinitialisation des paramètres d'usine

Lors de l'appui sur le bouton poussoir VERT, la fonction de RAZ se déclenche, et l'interface de la valise revient aux premiers menus, après que toutes les variables configurables par l'utilisateur ont été rapportées à leur valeurs initiales.

d. Correction fonction injection défaut homopolaire

Un défaut homopolaire est représenté par un court-circuit entre une ou plusieurs phases actives et la terre.

Voici un graphe tiré d'un oscilloscope explicatif de son apparition lors de l'injection d'un courant homopolaire effectuée par la valise :

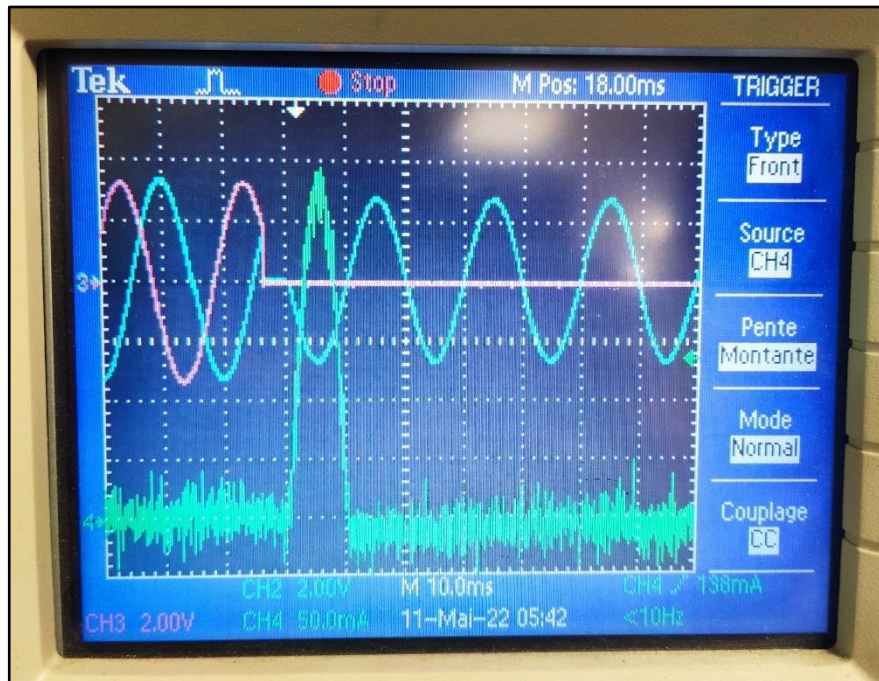


Figure 26: Photo de l'oscilloscope lors des tests défaillants d'injection de défauts homopolaires

- La courbe verte représente la valeur de courant
- Les courbes violette et bleue représentent respectivement les phases 1 et 2

Comme dit précédemment, le défaut consiste en la présence d'un court-circuit entre deux phases et la terre, ici entre la phase 2 et la terre, mais aussi entre la phase 3 et la terre, bien que celle-ci ne soit pas représentée sur l'oscilloscope.

Dans cette configuration, le défaut n'est pas détecté par le coffret EMIS – RTU850, qui est un nouveau modèle de coffret de la gamme EMIS proposé par Cahors. Lors de tests effectués par la compagnie ENEDIS sur le coffret, les techniciens ont obtenu des résultats concluant lors d'une injection un peu différente. En effet, la procédure d'injection actuelle est la suivante :

- Injection d'un pic de courant (visualisable en vert sur la photo ci-dessus)
- Mise à zéro de la phase 2 et 3 pendant la durée du défaut
- Temporisation de la valeur de la durée du défaut
- Remise en route des signaux initiaux sur les phases 2 et 3

Quant à la méthode développée par ENEDIS, voici son déroulé :

- Injection d'un pic de courant, identique à ce qui est fait chez Ensto
- Création de deux signaux en **opposition de phase** sur les phases 2 et 3
- Temporisation de la valeur de la durée du défaut
- Remise en route des signaux initiaux sur les phases 2 et 3

La création de deux signaux sinusoïdaux en opposition de phase permet de s'accorder sur les algorithmes de détection de défaut développés pour les coffrets EMIS. En effet, la création de ces deux signaux opposés permet une « valeur moyenne » nulle, observée comme un court-circuit par le coffret EMIS.

Voici les différents éléments de la solution apportée au programme :

- 1- Adapter la fonction d'injection du défaut homopolaire
- 2- Créer un signal sinusoïdal opposé aux valeurs de la phase 2
- 3- Injecter les deux signaux opposés sur les phases 2 et 3

Nouvelle routine d'injection :

```
/* Injection de la tension pour test ddd homo. */
else if (lance_v_inj_ddd_homo == OUI)
{
    gene_v_def_ddd_homo(tab_v_inj_ddd_homo);

    // Dans le cas de l'emis, présence tension reseau sur les deux autres phases
    if (mod_materiel_teste == RTU850_TBx)
    {
        gene_sinus_v2_50Hz(tab_amp_inj_v2_reseau);

        // Remplissage du tableau en opposition de phase (offset de 2048)
        for (int i = 0; i < 100; i++)
        {
            tab_amp_inj_v2_reseau_opp[i] = 4096 - tab_amp_inj_v2_reseau[i];
        }

        // Envoie du signal en opposition de phase à V2 sur V3
        gene_sinus_v3_50Hz(tab_amp_inj_v2_reseau_opp);
    }
    else
    {
        val_DAC_2 = AMP_NUL;
        val_DAC_3 = AMP_NUL;
    }
}
```

Figure 27: Code gérant l'injection de défauts homopolaires

On effectue une vérification sur le modèle en cours de test, s'il s'agit du modèle RTU850, on change la méthode d'injection :

- Injection d'une tension sinusoïdale sur la phase 2
- Calcul des valeurs en opposition de phase :
 - 2048 étant l'équivalent de 0V, on effectue les calculs en soustrayant à la valeur 4096 les valeurs du signal initial, pour créer une symétrie autour de la valeur 2048 : **valeur_signal_opposé = 4096 – valeur_signal**
- Injection de ce signal opposé sur la phase 3

On voit d'ailleurs que lorsque le modèle sélectionné n'est pas le RTU850, les valeurs des CAN sont mises à zéro lors de l'injection.

Voici les résultats obtenus à l'oscilloscope après avoir implémenté cette modification :

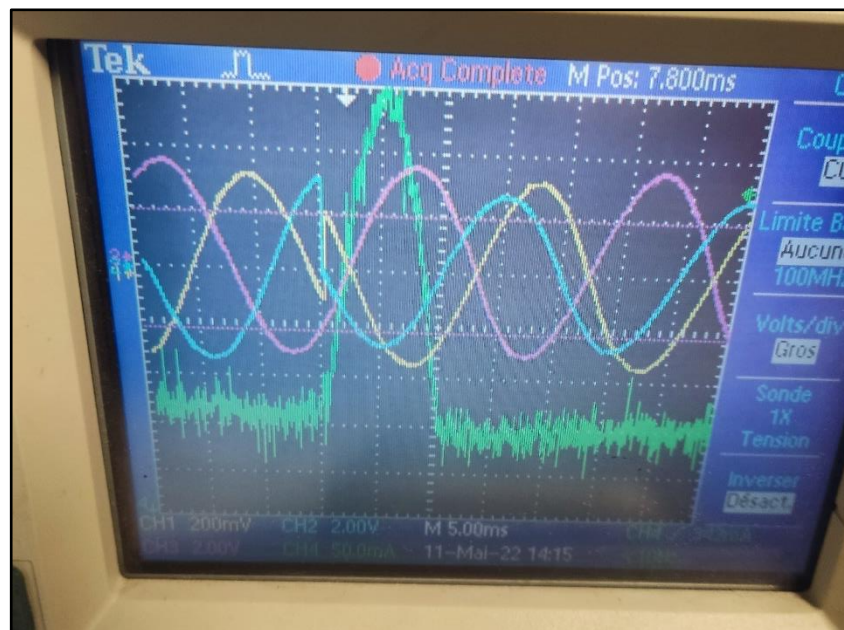


Figure 28: Photo de l'oscilloscope lors des tests réussis d'injection de défauts homopolaires

- La courbe verte représente la valeur du courant
- Les courbes bleue, violette et jaune représentent respectivement les valeurs de tension des phases 1, 2 et 3

Au moment de l'injection de courant, représentée ici par le pic formé par la courbe verte, il est très nettement observable que la courbe jaune devient l'exacte symétrie de la courbe violette autour de la valeur moyenne des 3 phases.

Le défaut est correctement détecté par le coffret lors de la nouvelle méthode d'injection.

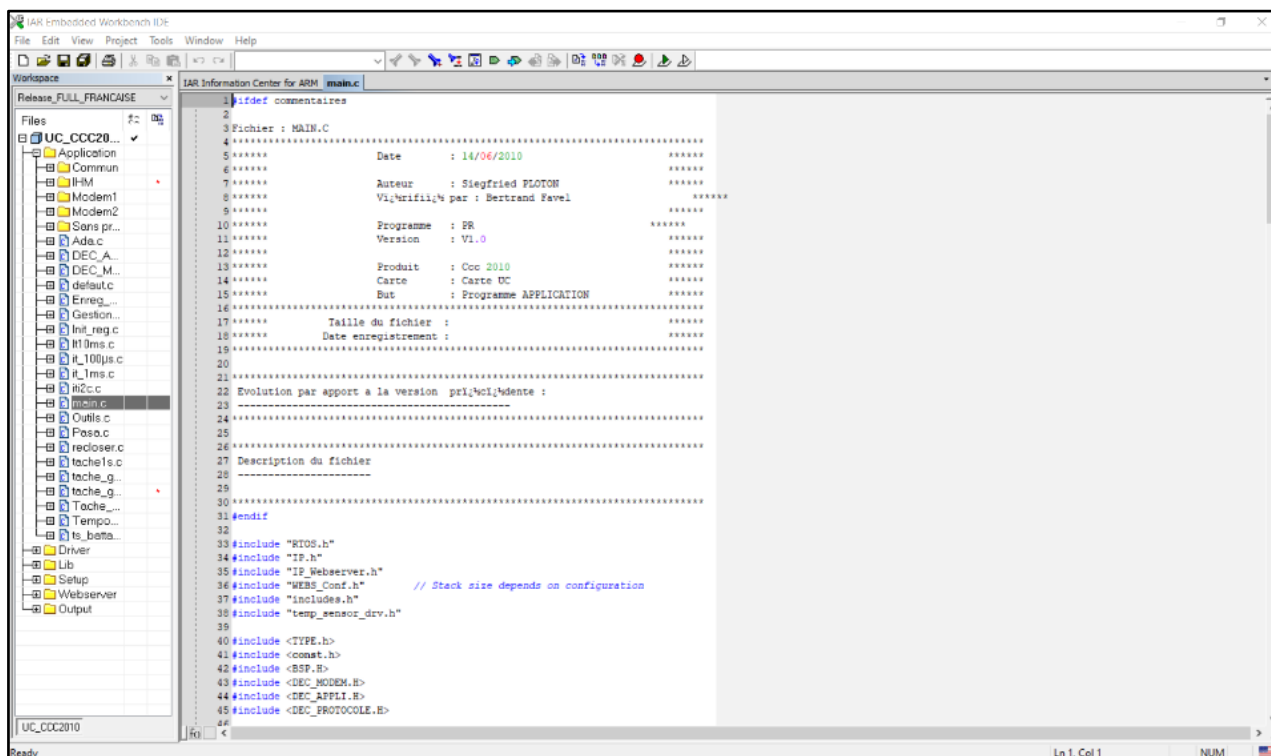
Bilan de la mission et perspective d'avenir

Les différentes améliorations nécessaires à l'évolution du produit vers une nouvelle version permettant une exportation de celui-ci à l'international est un élément majeur de la stratégie d'Ensto, puisque qu'il permet l'expansion de son panel de clients, mais aussi de relations internationales.

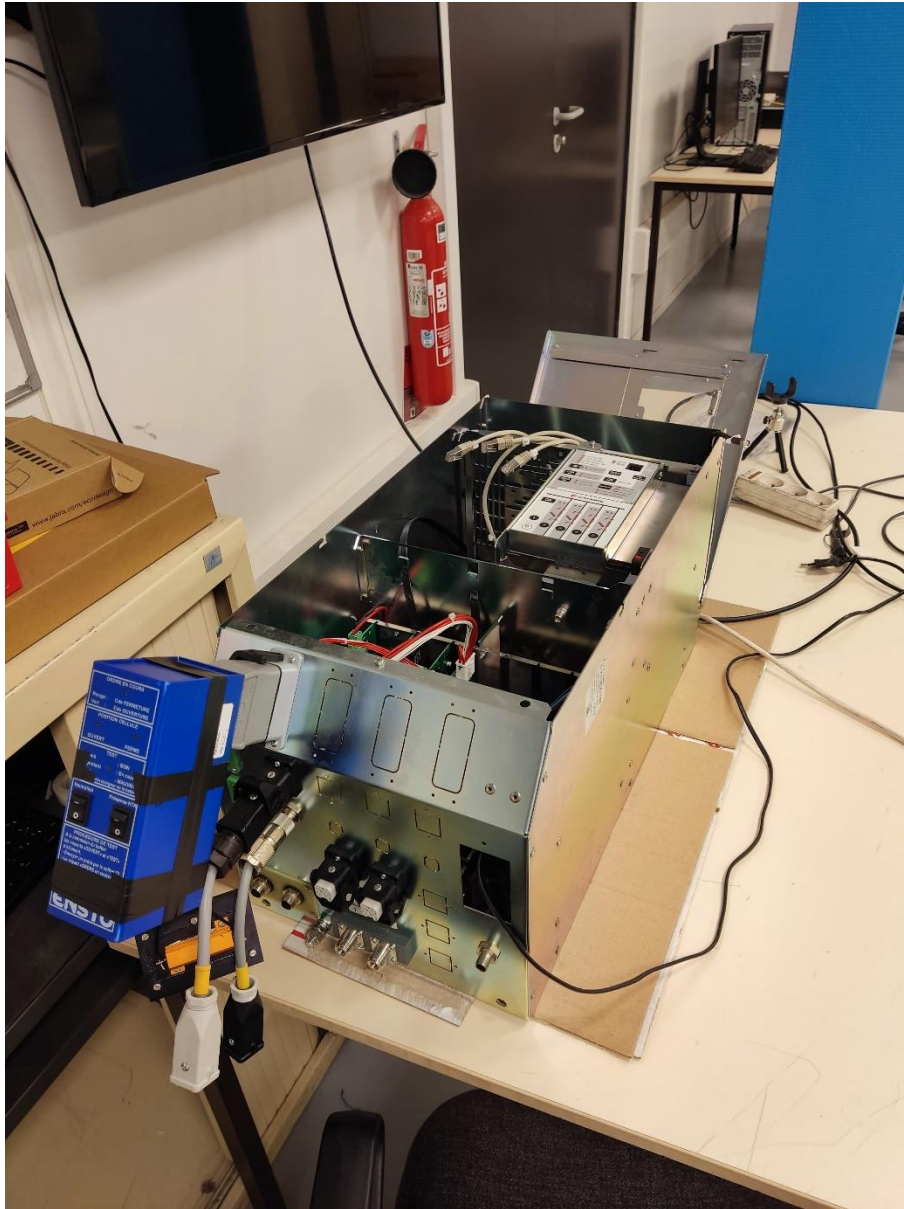
Du fait de l'accès direct à des éléments logiciels et matériels permettant d'effectuer toutes sortes de tests, le projet a pu évoluer et avancer sans encombre au fil du temps. De plus la possibilité de demander de l'aide et/ou des renseignements aux différents membres du BE ont permis de faciliter la compréhension du fonctionnement des produits, que cela concerne la valise ou bien encore les coffrets sur lesquels les tests étaient effectués.

Tout comme le fut la version « France » de la valise, les différents retours clients qui seront faits au SAV seront étudiés et pris en compte lors des prochaines implémentation ou modification de fonctionnalités dans le futur.

Annexes



Annexe n°1: Logiciel IAR Embedded Workbench



Annexe n°2 : Coffret EMIS RTU 850-TBx, Cahors

Résumé

Ce document relate le développement de la nouvelle version du logiciel d'une valise de test. Ce projet s'inscrit dans la continuité de la politique d'exportation des produits et solutions matérielles proposés Ensto Novexia, entreprise spécialisée dans le domaine électrique et particulièrement les appareils et équipements présents sur les réseaux de distribution électrique.

La valise VISIO II Export doit répondre à la problématique apportée par son exportation : son accessibilité pour les techniciens qui l'utiliseront en dehors du territoire français, puisqu'elle cible dorénavant les pays de l'Europe, d'Afrique ou encore d'Amérique du Sud.

Pour répondre aux nombreuses attentes que cela implique, les solutions proposées dans ce rapport seront développées et expliquées afin de permettre une bonne compréhension du fonctionnement général du produit.

Le produit est prêt à répondre aux différents tests sur le terrain, et peut à tout moment recevoir de nouvelles mises à jour en fonction des retours clients qu'Ensto Novexia pourrait recevoir durant les premiers mois d'utilisation de la valise.