CONSERVATOIRE NATIONAL DES ARTS ET METIERS

PARIS

MEMOIRE

Présenté en vue d'obtenir

le DIPLOME d'INGENIEUR CNAM

SPECIALITE : INFORMATIQUE

OPTION : ARCHITECTURE ET INGENIERIE DES SYSTEMES ET DES LOGICIELS

par

Jean-François DENIS

Contrôle/Commande de la source deuton de Spiral2 sous EPICS

Soutenu le 11 mai 2010

JURY : Christine **CROCHEPEYRE**, Jean-Michel **DOUIN**

PRESIDENT : Yann **POLLET**

MEMBRES : Françoise **Gougnaud**, Jean-François **Gournay**

Remerciements,

En préambule à ce mémoire, je souhaiterai adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leur aide.

Je tiens à remercier sincèrement Monsieur Jean-François Gournay, qui, en tant que responsable du laboratoire d'informatique industrielle en 2008, m'a proposé d'intégrer son équipe d'experts dans le but de réaliser mon mémoire de fin d'études, et m'a soutenu durant toute la réalisation de celui-ci.

D'autre part je tiens à remercier sincèrement Madame Florence Ardellier, chef de service du SIS, ainsi que Monsieur Gilles Durand, nouveau responsable du laboratoire d'informatique industrielle, qui m'ont soutenu et intégré dans leur effectif pour les années à venir.

J'exprime ma gratitude à tous les collègues qui m'ont aidé et soutenu durant la rédaction de ce mémoire, et plus particulièrement à Françoise Gougnaud, Yves Lussignol, Pierre Mattei, pour avoir répondu à mes questions à propos d' « EPICS ».

Merci à tous et à toutes.

Sommaire

Table des matières

Résumé

Ce mémoire s'articule autour du contrôle/commande de la source de Deutons pour le projet SPIRAL2. C'est un projet d'accélérateur qui va venir s'implanter sur l'installation déjà existante du GANIL à Caen. Il aura pour objectif d'élargir la gamme de faisceaux exotiques radioactifs produit par le GANIL. La source de Deutons se situe sur la partie accélérateur, et plus particulièrement sur l'injecteur. Pour réaliser le contrôle/commande de la source, j'ai utilisé la suite d'outils logiciels EPICS (Experimental Physics and Industrial Control System) spécialisée dans les accélérateurs soutenue activement par une communauté internationale. D'autre part j'ai été amené à travailler sur le contrôle/commande des fentes se trouvant aussi sur l'injecteur.

Mots clés : SPIRAL2, GANIL, sourde deuton, EPICS, fentes

Abstract

This report is about the control command of the deuton source for the SPIRAL2 project. SPIRAL2 is an accelerator project that will be integrated in the existing facility of GANIL in Caen. It will allow broadening the range of exotic radioactive beams produced by the GANIL. The construction will proceed in two phases: an initial phase for accelerator and a second phase involving the production of exotic beams. The Deuton source is located on the accelerator part, and more particularly on the injector. To design the control command of the Deuton, I used EPICS (Experimental Physics and Industrial Control System) which is specialized toolbox for accelerators control, actively supported by international community. Moreover, due to some supply problems for elements of the source, I worked on the control command of Faraday cup also located on the injector.

I - Introduction

Ce mémoire s'articule autour du contrôle/commande de la source de Deutons pour le projet d'accélérateur SPIRAL2 qui va venir s'implanter sur l'installation déjà existante du GANIL à Caen. Il aura pour objectif d'élargir la gamme de faisceaux exotiques radioactifs produit par le GANIL. La construction se déroulera en deux phases : la première concernera l'accélérateur et la seconde, la production de faisceaux exotiques. Mon travail va s'orienter sur le pilotage de la source de Deutons située dans l'accélérateur, et plus exactement dans l'injecteur.
Mon rapport va se présenter en trois parties :

Tout d'abord il sera dressé dans un contexte scientifique l'évolution et les avancées de la France avec l'accélérateur d'ions lourds du GANIL, et du futur projet SPIRAL2, sur lequel je vais travailler.

Ensuite il sera présenté EPICS (Experimental Physics and Industrial Control System) qui est une suite d'outils logiciels, spécialisée dans les accélérateurs, soutenue activement par une communauté internationale.

Et enfin j'aborderai le déroulement du développement EPICS concernant le pilotage de la source de Deutons et celui des fentes, projet qui a été ajouté à ma mission initiale.

Je terminerai par un bilan personnel et les perspectives du projet SPIRAL2 qui incluent maintenance et actions à venir dont j'aurai la charge.

II - Contexte

1- Le centre du GANIL, aujourd'hui

2- De l'accélérateur GANIL à SPIRAL

2-1 Le premier accélérateur : GANIL

2-2 Pour élargir le champ d'investigation du GANIL : SPIRAL

1- Le centre du GANIL, aujourd'hui

Le site du GANIL (Grand Accélérateur National d'Ions Lourds) est un laboratoire de recherche commun à la Direction des Sciences de la Matière (DSM) du Commissariat à l'Energie Atomique (CEA) et à l'Institut National de Physique Nucléaire et de Physique des Particules (IN2P3) du Centre National de Recherche Scientifique (CNRS), situé sur les communes d'Epron, Hérouville-Saint-Clair et Caen.
Cette installation est exploitée par un Groupement d'Intérêt Economique (GIE) formé par un contrat du 19 janvier 1976 entre le CEA et le CNRS.

Figure 1 : Le GANIL, aujourd'hui

1.1 Son rôle

La vocation du GANIL est de contribuer à l'avancement des connaissances dans le domaine de la physique nucléaire, et de la physique atomique. Les faisceaux d'ions, de l'hélium à l'uranium, de moyenne énergie (20 à 100 MeV/nucléon), permettent d'étudier les noyaux atomiques dans des états extrêmes.
Le GANIL, laboratoire d'accueil, se veut un outil à la disposition de la communauté scientifique nationale et internationale. Les 250 agents du GANIL ont pour mission essentielle de rendre possible des recherches expérimentales en fournissant les moyens nécessaires : faisceaux d'ions lourds, grands équipements des salles d'expériences et enregistrements informatiques des données expérimentales.
Chaque année, environ 700 physiciens français et étrangers fréquentent le GANIL, pour quelques jours ou quelques semaines, afin d'y réaliser les expériences qu'ils ont d'abord préparées et qu'ils analysent ensuite dans leurs propres laboratoires.
En physique nucléaire, le GANIL a permis de nombreuses découvertes sur la structure du noyau de l'atome, sur ses propriétés thermiques et mécaniques, et sur les noyaux que l'on dit exotiques, n'existant pas à l'état naturel sur Terre.

2- De l'accélérateur GANIL à SPIRAL

2.1 Le premier accélérateur : GANIL

Le **G**rand **A**ccélérateur **N**ational d'**I**ons **L**ourds fut le premier accélérateur à CAEN installé en 1976. Le premier faisceau est délivré en novembre 1982, et la première expérience aura lieu en janvier 1983.

La première mission du GANIL consiste à produire et accélérer des ions lourds.

Sa deuxième mission est d'accompagner les physiciens dans leurs expériences. Il s'agit de permettre aux physiciens d'observer des réactions nucléaires. Le bombardement d'une cible par un faisceau de particules lancé à très grande vitesse par l'ensemble accélérateur, provoque des collisions nucléaires et créé d'autres noyaux de particules. La cible est entourée de dispositifs de détection qui enregistrent les signaux des phénomènes produits. Ils sont ultérieurement analysés afin de reconstituer à posteriori le déroulement d'une collision et de permettre ainsi de décrypter les propriétés du noyau atomique.

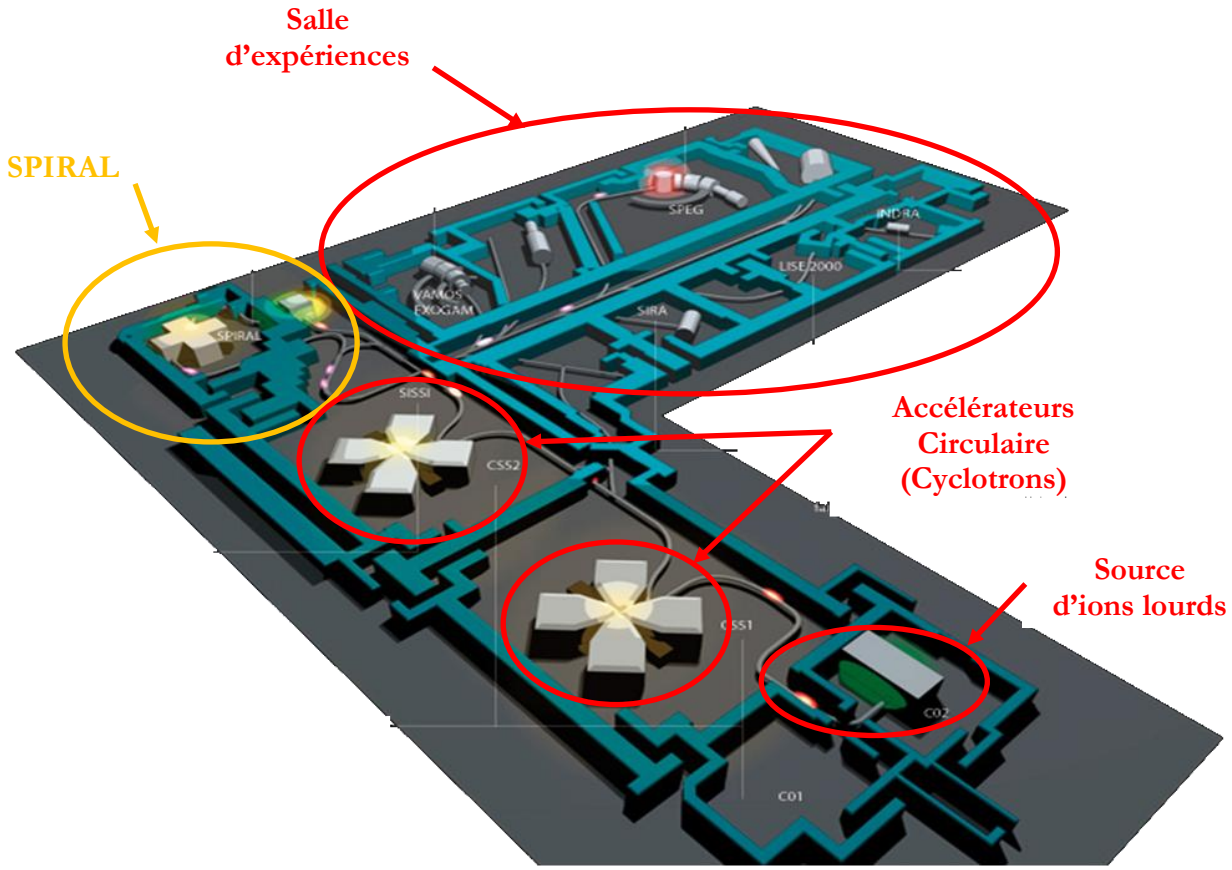


Figure 2 : GANIL, l'accélérateur

2.2 Pour élargir le champ d'investigation du GANIL : SPIRAL

SPIRAL : **S**ystème de **P**roduction d'**I**ons **R**adioactifs en **L**igne.

Quel est son objectif ?

Pour compléter la gamme d'énergies de faisceaux fournie par le GANIL, **SPIRAL** doit fournir un faisceau d'ions exotiques de basse énergie compris entre **2*Mev/nucléon*** et **25*Mev/nucléon***. La méthode de production du faisceau est « **I**sotope **S**eparation **O**n **L**ine » (***ISOL***). Ceci permet de délivrer des faisceaux d'ions radioactifs intenses possédant de bonnes qualités optiques. Cette installation est particulièrement adaptée pour la production et l'accélération de noyaux légers et moyennement lourds.

Le GANIL produit et accélère des faisceaux stables allant du Carbone 12 à l'Uranium; il a deux techniques pour produire des faisceaux radioactifs, soit par fragmentation de faisceaux stables sur des cibles épaisses, soit par la méthode ISOL utilisée dans le dispositif SPIRAL. Cependant la gamme de faisceaux radioactifs fournis par ces deux machines n'est pas complète. Pour compléter cette gamme, avec la production et l'accélération d'isotopes radioactifs rares plus lourds (jusqu'à l'Uranium) ou plus riches en neutrons, ***SPIRAL2*** va voir le jour.

III – Le projet SPIRAL2

1- Présentation du projet

2- L'objectif scientifique de spiral 2

3- Description technique

3.1 Synoptique

3.2 Fonctionnement général

3.3 Description de l'ensemble accélérateur

4- Les infrastructures

5- Le partage des lots pour la phase 1 : l'ensemble accélérateur

1- Présentation du projet

SPIRAL2 (Système de production d'Ions Radioactifs en Ligne de 2ème génération) est un projet d'accélérateur de particules linéaires pour des études de physique nucléaire fondamentale et de recherches interdisciplinaires. SPIRAL2 est un projet scientifique international. Il fait partie des 40 projets scientifiques soutenus par **E**uropean **R**oadmap **F**or **R**esearch **I**nfrastructures (ESFRI) qui est un forum stratégique européen pour les infrastructures de recherche qui a été créé en avril 2002. C'est un projet partagé par de nombreux laboratoires du CNRS et du CEA en France, ainsi que par de nombreux laboratoires internationaux localisés aux USA, en Allemagne, en Bulgarie, en Espagne, à Bombay, à Tokyo, etc…

2- L'objectif scientifique de SPIRAL2

Le projet SPIRAL2 a pour objetif de développer le potentiel de production des faisceaux au GANIL : il délivrera avec des intensités accrues les faisceaux actuels et offrira un domaine beaucoup plus vaste d'ions radioactifs. Opérationnel en 2012, il ouvrira un nouvel horizon pour les recherches sur les propriétés nucléaires. Il offrira ainsi à la communauté internationale (plus de 700 physiciens concernés) des possibilités nouvelles d'exploration de la structure nucléaire comme l'observation des noyaux éloignés de la vallée de stabilité, riches ou déficients en neutrons, la mesure de leurs formes, la détermination de nouveaux effets de couches nucléaires, la formation de nouveaux noyaux très lourds, et l'étude des noyaux impliqués dans les processus astrophysiques.

3- Description technique

3.1 Fonctionnement général

Il y aura deux bâtiments :

- Le bâtiment « accélérateur » pour créer le faisceau primaire
- Le bâtiment « production » pour créer le faisceau secondaire de particules radioactives

Le bâtiment accélérateur regroupera trois instruments : l'injecteur, le LINAG cryogénique et la cible.

L'injecteur rassemble plusieurs éléments qui sont regroupés par niveau énergétique : ce sont les **L**ignes de **B**asse **E**nergie (**LBE**), le **Q**uadrupole **R**adio **F**réquence (***RFQ***), et la **L**igne de **M**oyenne **E**nergie (***LME***).

L'accélérateur linéaire principal (***LINAG***) supraconducteur d'ions légers ou lourds fonctionnera avec un potentiel d'accélération d'environ 40 MeV.

Il pourra accélérer les deutons d'intensité 5 mA jusqu'à 40 MeV et des ions lourds d'1 mA jusqu'à 14.5 MeV/nucléon. Les faisceaux accélérés par le LINAG pourront bombarder des cibles minces ou épaisses, et être ainsi employés pour la production de faisceaux radioactifs intenses par des mécanismes de réaction variés (fusion, fission, réactions de transfert multi-nucléons notamment pour la production des faisceaux légers etc.) et par plusieurs techniques (ISOL, IGISOL, spectromètres de recul, etc...). De plus, l'accélération de faisceaux d'ions lourds à haute intensité, grâce à l'avènement des nouvelles générations de sources d'ions ECR, permettra de mener des expériences de fusion-évaporation. La production des faisceaux radioactifs de noyaux lourds riches en neutrons à haute intensité reposera sur la fission de l'uranium induite par le bombardement d'un faisceau de neutrons (obtenus par la cassure des noyaux d'un faisceau de deutons frappant le convertisseur de graphite de la cible de carbure d'uranium) ou bien par l'irradiation directe avec un faisceau de deutons, de noyaux d'^{3}He ou d'^{4}He. Un taux de 10^{14} fissions/s devrait être atteint dans le cadre du projet.

SPIRAL2 se construira en deux phases : la partie accélérateur qui correspond au bâtiment accélérateur, et la partie production de faisceaux radioactifs, qui correspond au deuxième bâtiment. Dans ce document nous ne parlerons que de la phase 1, soit la partie accélérateur.

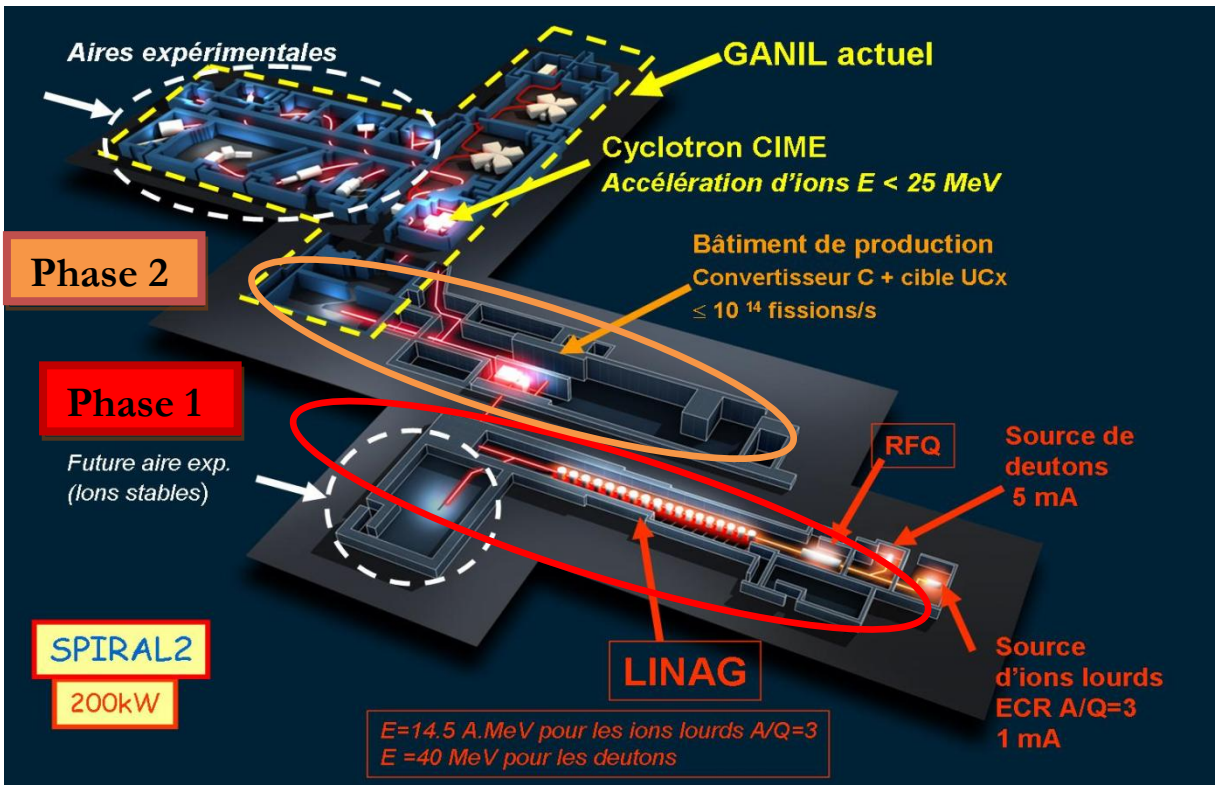


Figure 3 : SPIRAL2

3.2 Synoptique

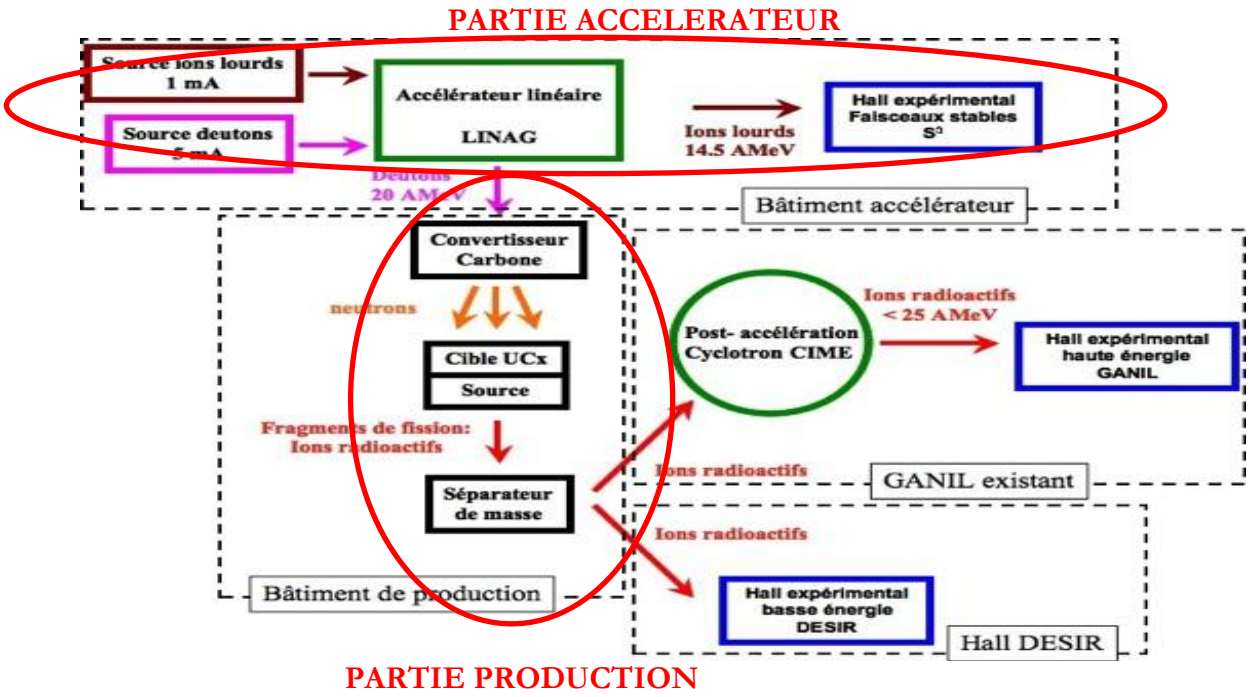


Figure 4 : synoptique de SPIRAL2

3.3 Description de la partie accélérateur

Au cœur de la future machine, un accélérateur linéaire supraconducteur, délivrant des faisceaux d'ions parmi les plus intenses du monde, bombardera une cible de matière. Les réactions induites : fission, transfert, fusion, ... engendreront des milliards de nouveaux noyaux.

L'accélérateur primaire est conçu pour produire les faisceaux principaux suivants :

- Des faisceaux de deutons d'intensité maximale de 5 mA, et d'énergie finale de 40 MeV
- Des faisceaux d'ions de rapport masse sur charge $A/q \leq 3$, d'intensité maximale 1 mA, et d'énergie finale 14.5 A.MeV.

L'ensemble « accélérateur » est composé d'éléments que je vais détailler par la suite :

- L'injecteur avec les deux sources
- Quadripôle Radio Fréquence (RFQ)
- Ligne Moyenne Energie (LME)
- L'accélérateur linéaire du GANIL (LINAG)

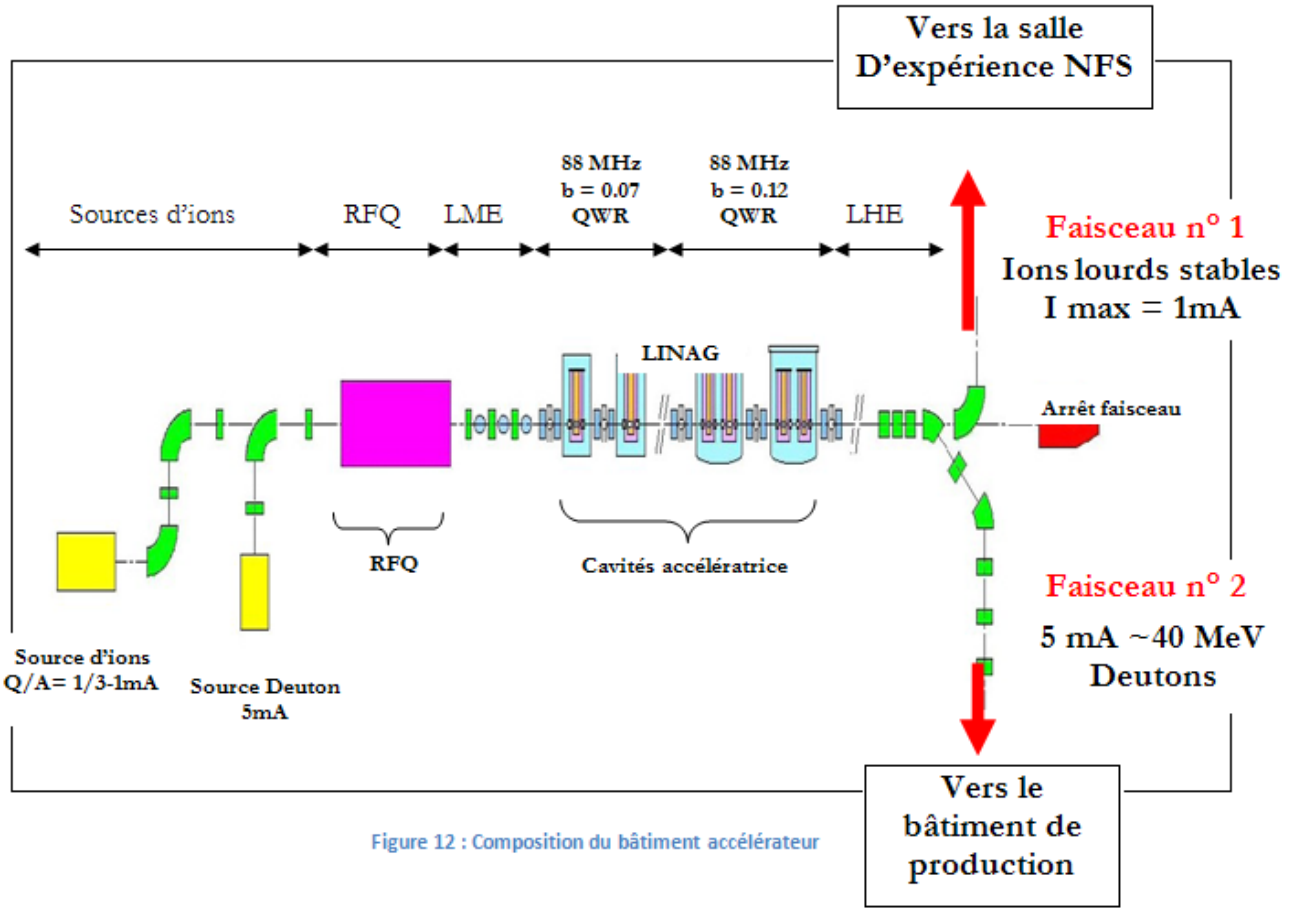


Figure 12 : Composition du bâtiment accélérateur

3.3.1 *L'Injecteur*

L'injecteur comprend deux sources d'ions distinctes, une pour produire les faisceaux de Deutons (et de protons) sur laquelle va porter mon travail, et l'autre pour produire les faisceaux d'ions plus lourds avec $A/q \leq 3$. A chacune de ces sources est associée une **L**igne de transport **B**asse **E**nergie (LBE). Ces deux lignes LBE fusionnent avant de mener au **Q**uadripôle **R**adio **F**réquence (RFQ), dont le rôle est de créer la structure temporelle du faisceau à 88.05 MHz, soit de regrouper par petits paquets le faisceau tout en le focalisant, soit de l'accélérer jusqu'à 0.75 A.MeV. La ligne de transport LME injecte alors le faisceau dans le LINAG supraconducteur.

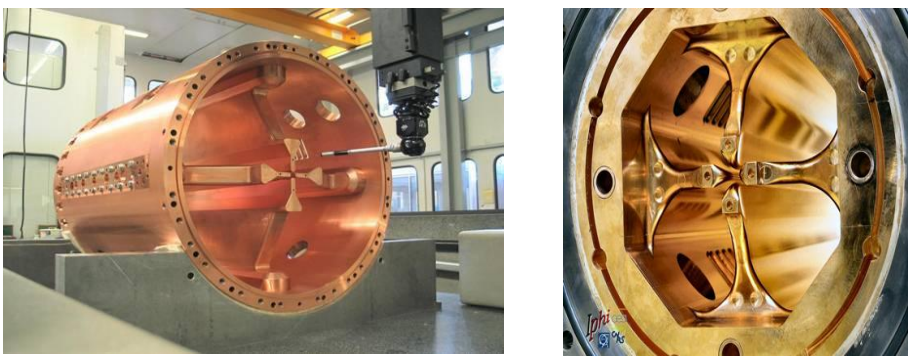


Figure 5 : RFQ, Quadripôle Radio Fréquence

3.3.2 Linac supraconducteur : LINAG

L'accélérateur linéaire, le LINAG, est constitué de cavités accélératrices supraconductrices alimentées de façon indépendante. Une telle solution permet d'assurer une flexibilité maximale en termes d'espèces d'ions à accélérer et d'énergies finales à atteindre. Le LINAG se compose de cavités accélératrices « quart d'onde » résonnant à 88.05 MHz. La longueur totale du LINAG est d'environ 25 mètres.

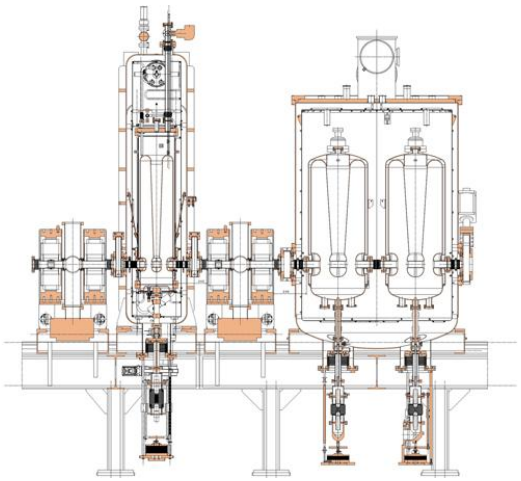


Figure 6 : LINAG : Cavités accélératrices de type A puis de type B

3.3.3 Lignes haute-énergie (LHE)

A la sortie du LINAG, le faisceau peut être distribué à trois endroits différents. Une première ligne de transport LHE conduit à un arrêt faisceau. Une seconde ligne LHE transporte le faisceau à la casemate contenant l'ensemble cible-source produisant les ions radioactifs, et le focalise afin d'obtenir une distribution de puissance uniforme (Pfaisceau ≤ 200 kW) sur le convertisseur de neutrons (ou directement sur la cible). Enfin, une troisième ligne LHE transporte et adapte le faisceau jusqu'aux aires expérimentales pour ions stables.

4- Les infrastructures

SPIRAL2 sera construit sur le site du GANIL et doublera la superficie actuelle des installations qui passera d'environ 11 000 m² à 20 000 m². La construction de la phase 1 du projet (bâtiments accélérateur) commencera en 2010, celle de la phase 2 (bâtiments production) en 2011.

L'une des préoccupations du GANIL étant le respect de l'environnement, la démarche HQE (Haute Qualité Environnementale) a été prise en compte pour la conception et la construction de la nouvelle installation.

Figure 7 : Implantation de SPIRAL2 au GANIL

En plus d'un espace d'accueil et des aires de service, l'installation SPIRAL2 comprendra en fin de construction un groupe de bâtiments :

- Le bâtiment accélérateur abritant les sources d'ions (deutons, protons et ions lourds), l'accélérateur linéaire (LINAC) et les lignes de distribution des faisceaux
- Le bâtiment des Aires Expérimentales associées au LINAC (AEL)
- Le bâtiment de production des ions exotiques
- Le bâtiment destiné aux expériences à basse énergie appelé DESIR

5- Le partage des lots pour la phase 1 : l'ensemble accélérateur

Nous allons nous concentrer uniquement sur la partie accélérateur (Phase1), car la partie production de faisceaux (Phase2) n'interviendra pas dans notre travail pour le moment. Cette deuxième partie s'effectuera en 2012, après les premiers tests faisceaux stables de l'accélérateur.
La partie accélérateur a été découpée en lots, répartis entre les acteurs, dont voici un schéma récapitulatif :

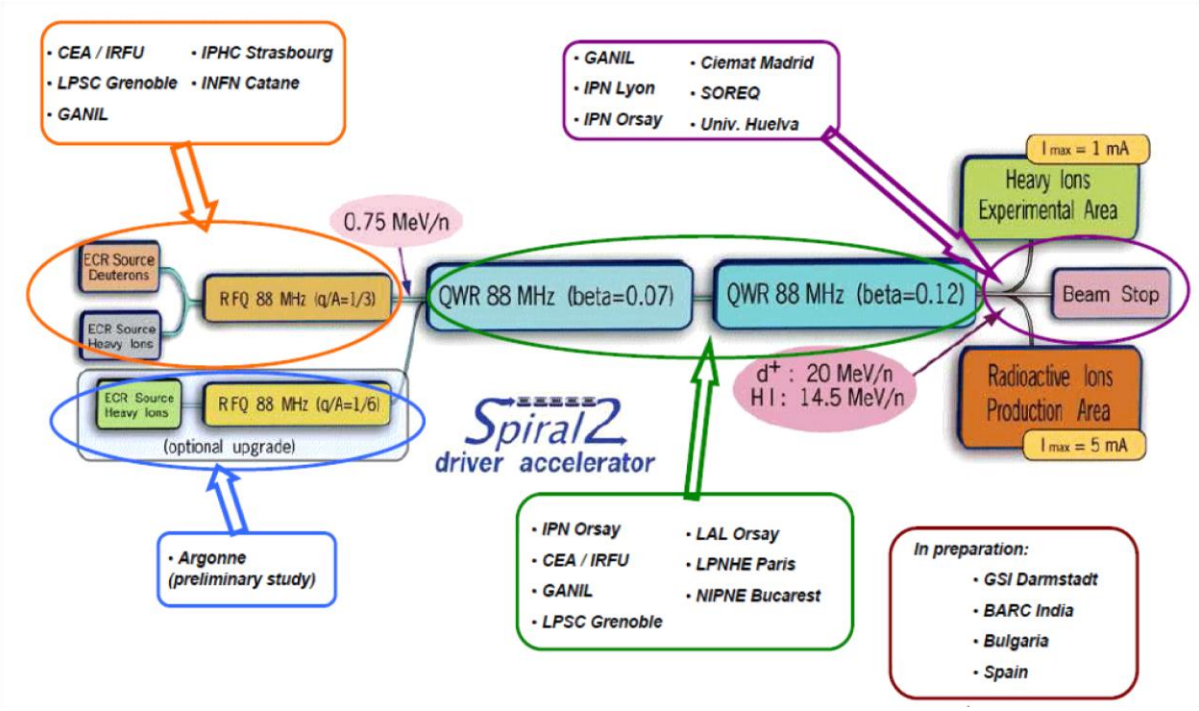


Figure 8 : Partage des lots

Le CEA/IRFU va participer au développement de l'injecteur et à la conception du RFQ. La partie contrôle/commande de l'injecteur en particulier est à la charge d'un service de l'IRFU, expert dans le domaine du contrôle commande d'accélérateur. Ce service est le **S**ervice d'**I**ngénierie **S**ystème(***SIS***). C'est dans ce service que je vais réaliser mon travail, et plus particulièrement dans **L**aboratoire de **D**éveloppement en **I**nformatique **I**ndustrielle (LD2I).

IV - Ma mission

1- L'objectif de ma mission initiale : Pilotage de la source deuton

2.1 Présentation

2.2 La source deuton SD sur la ligne LBE2

2- Les objectifs supplémentaires

3.1 Présentation

3.2 La cage de Faraday CF11 sur la ligne LBE2

3.3 Les fentes FH13 et FV13 sur la ligne LBE1

J'ai donc travaillé sur le projet SPIRAL2 au sein du Commissariat d'Energie Atomique (CEA), à Saclay dans le Laboratoire d'Informatique Industrielle (LD2I), expert dans le domaine du contrôle/commande de machine comme les accélérateurs de particules.

1- L'objectif de ma mission initiale

1.1 Présentation

L'objectif de ma mission a consisté au développement du système informatique permettant le fonctionnement de la Source de Deutons de SPIRAL2. Du fait de la complexité des tâches à effectuer, il est en effet indispensable que le physicien soit aidé par un système permettant la centralisation des informations et des commandes, la surveillance des paramètres et la réalisation de procédures automatiques pour le démarrage et l'arrêt de la source ainsi que pour la gestion des situations anormales.

La réalisation a été effectuée à partir de solutions bien maîtrisées par le laboratoire. L'interfaçage avec la source a été réalisée à partir de modules industriels au standard VME ***(****Versa* ***M****odule* ***E****urocard)***. Le poste de supervision ainsi que celui du développement et de la configuration sont sur un PC sous Linux. L'infrastructure logicielle générale pour le contrôle de la source est Epics (Experimental Physics and Industrial Control system). Ce logiciel fournit les facilités pour aborder ce type de problème. Il sera d'ailleurs utilisé pour l'ensemble de Spiral2.

Le travail a consisté donc :

- En l'apprentissage du contexte de travail (étude des documentations, recherche d'informations complémentaires sur les sites dédiés, étude des modules matériels à utiliser, études des appareils à interfacer)
- En la configuration de la base de données «EPICS» permettant l'accès de la supervision au matériel
- Au suivi et de la participation aux tâches et tests d'interfaçage des divers signaux
- A la configuration des outils de supervision (synoptiques, archivage, mémorisation des réglages)
- A l'écriture des procédures spécifiques pour le fonctionnement de la source (langage SNL (***S****tate* ***N****otation* ***L****anguage*) à base du langage C)

2.1 La source deuton sur la ligne LBE2

2.1.1 Les objectifs scientifiques de l'équipement

La source de Deutons « **D+** » doit produire des faisceaux dont l'intensité peut varier entre 0,1 à 5 mA à une énergie de 40 KeV. De plus elle doit fournir un faisceau de 5 mA de protons à une énergie de 20KeV.

2.1.2 Les performances principales de l'instrument

La source a pour objectif de fournir un faisceau aux caractéristiques suivantes :

- Espèces d'ions : D+ et H2+ pour le conditionnement
- Energie : 40 KeV (20 KeV/nucléon)
- Fonctionnement continu et pulsé
- Intensité de 0,15 mA à 5 mA en fonction de la cible.

2.1.3 Profil de vie

La source sera construite à Saclay par l'IRFU SACM en 2007 – 2009. Elle sera testée en fonctionnement proton sur un banc autonome en 2009 avant son transfert sur l'injecteur Spiral 2 (à Saclay) en 2010. Après les tests de l'injecteur, cette source sera déplacée à GANIL pour son implantation définitive dans le bâtiment accélérateur SPIRAL2 en 2012 ou 2013.

2.1.4 Le principe de fonctionnement

Le principe de fonctionnement d'une source à **R**ésonance **C**yclotronique **E**lectronique (***ECR***) est fondé sur la résonance entre le mouvement d'un électron placé dans un champ magnétique et une onde électromagnétique haute fréquence (quelques GHz) lui communiquant ainsi de l'énergie. Les ions sont créés par l'ionisation pas à pas des atomes présents à l'intérieur de la source, dans notre cas des atomes de Deutérium. L'ionisation est réalisée grâce aux collisions successives entre les électrons du plasma et les atomes ou les ions. Ces électrons ont une énergie comprise entre quelques eV et plusieurs centaines de KeV. Un atome peut se retrouver alors totalement épluché c'est-à-dire qu'il ne reste aucun électron autour de son noyau. L'électron unique de l'atome de Deutérium monte alors en énergie et se détache du noyau, créant ainsi un deuton.

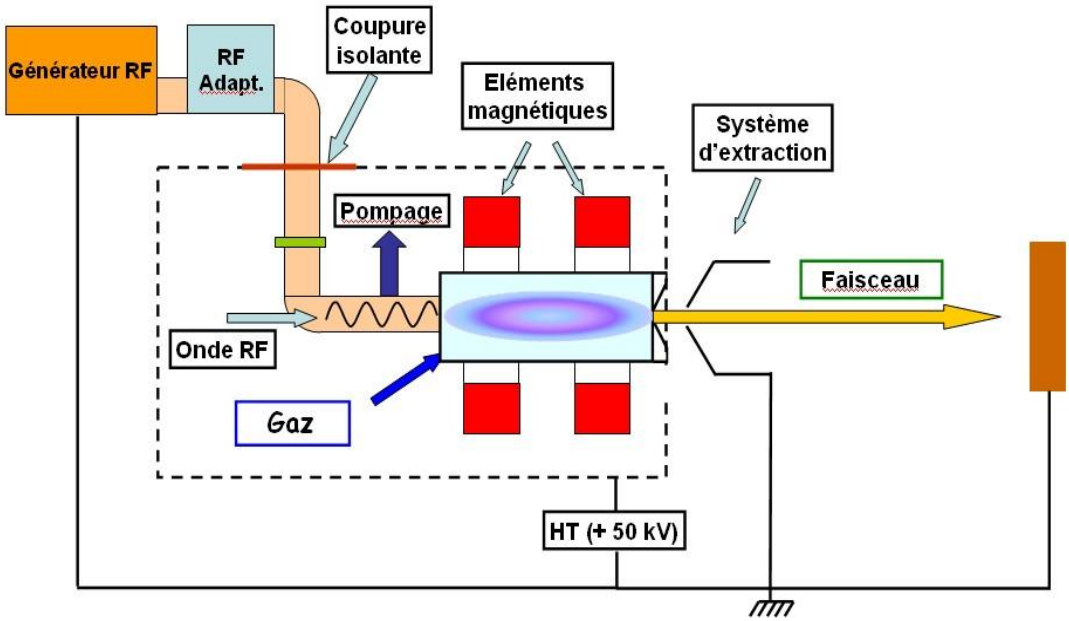


Figure 9 : Schéma de principe de fonctionnement

Hydrogène : 1 proton dans le noyau et 1 électron en orbite.
Deutérium : Hydrogène lourd, 1 proton et 1 neutron dans le noyau et 1 électron en orbite.
Deuton : Deutérium sans électron, soit composé 1 proton et 1 neutron

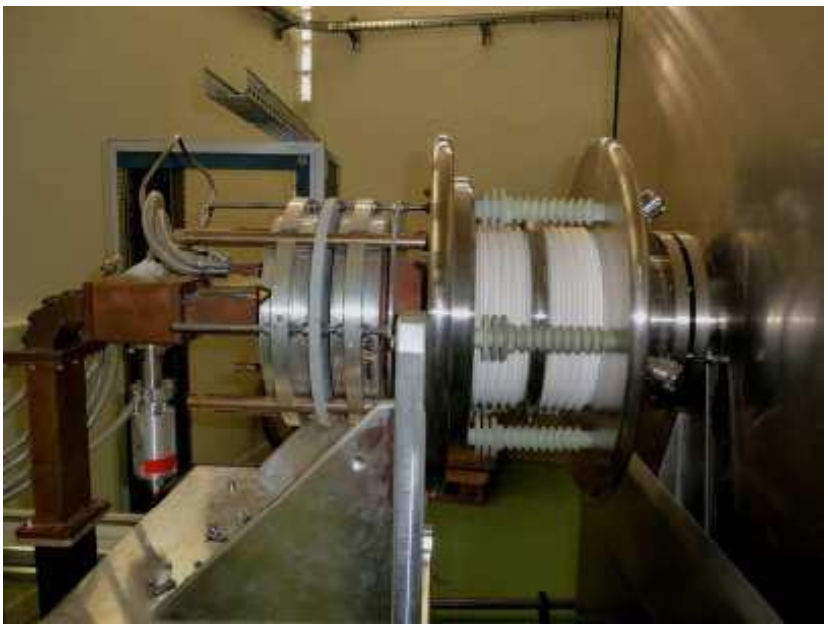


Figure 10 : Photos source deuton

2.1.5 *Schéma synoptique*

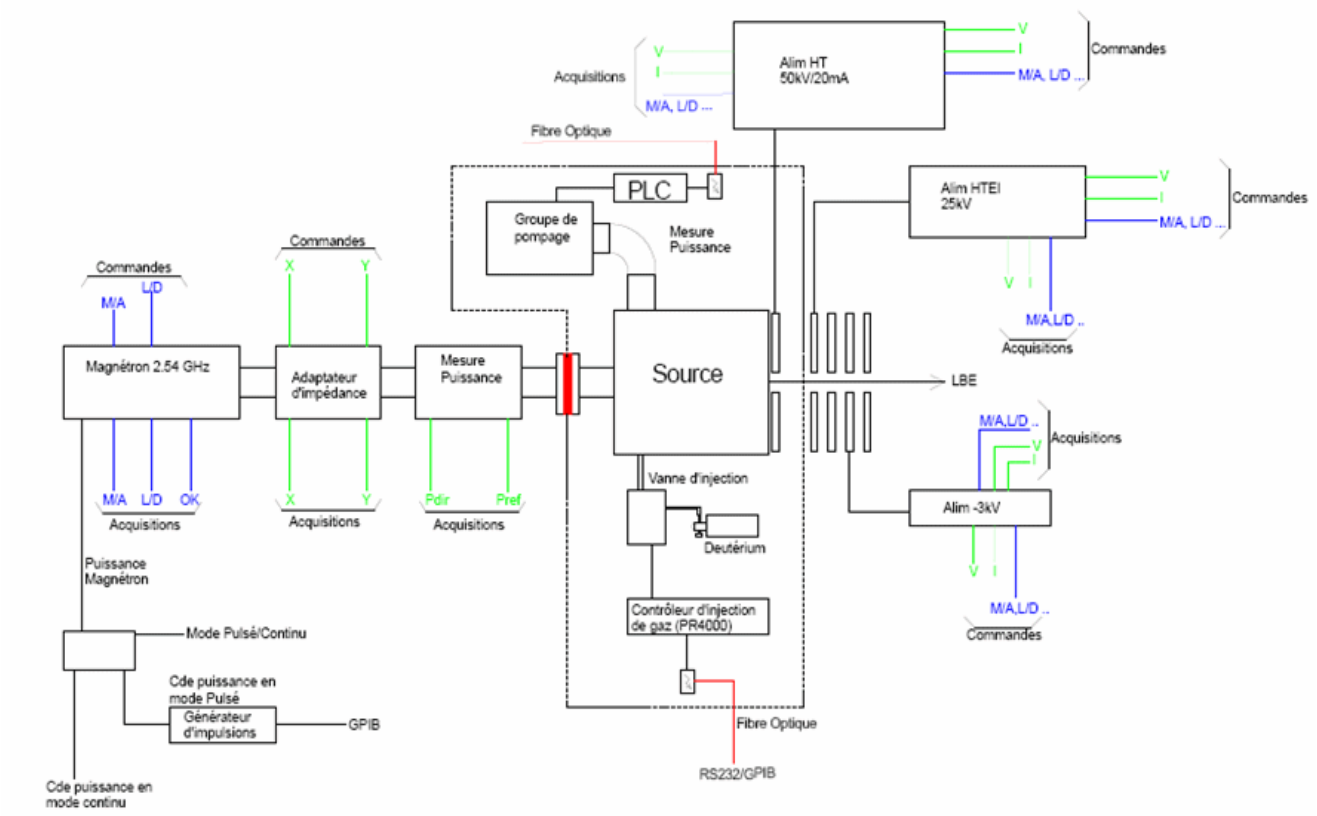


Figure 11 : Schéma synoptique du contrôle/commande de la source deuton

2- Les objectifs supplémentaires

3.1 Présentation

L'objectif de ma mission initiale a été modifié. Effectivement durant mon mémoire j'ai été amené à développer un programme de pilotage pour les fentes de l'injecteur de SPIRAL2.
Sachant que le test de la source de Deutons sans un minimum d'équipement pour visualiser qu'elle répond bien aux demandes n'est pas envisageable, les cages de Faraday permettant de mesurer l'intensité du faisceau ont été intégrées à l'architecture. J'ai été nommé responsable de l'intégration du contrôle/commande pour les tests de Saclay, partie EPICS.

3.2 La cage de Faraday CF11 sur la ligne LBE2

Une cage de Faraday a pour objectif de mesurer le courant du faisceau. Elle est pilotée mécaniquement via un moteur actionnant un vérin afin qu'elle soit à l'intérieur ou à l'extérieur du faisceau.

Elle est composée :

- D'une coupelle de Faraday en cuivre refroidie par eau
- De son propulseur pour bouger mécaniquement la cage
- D'une partie électronique pour la mesure du courant

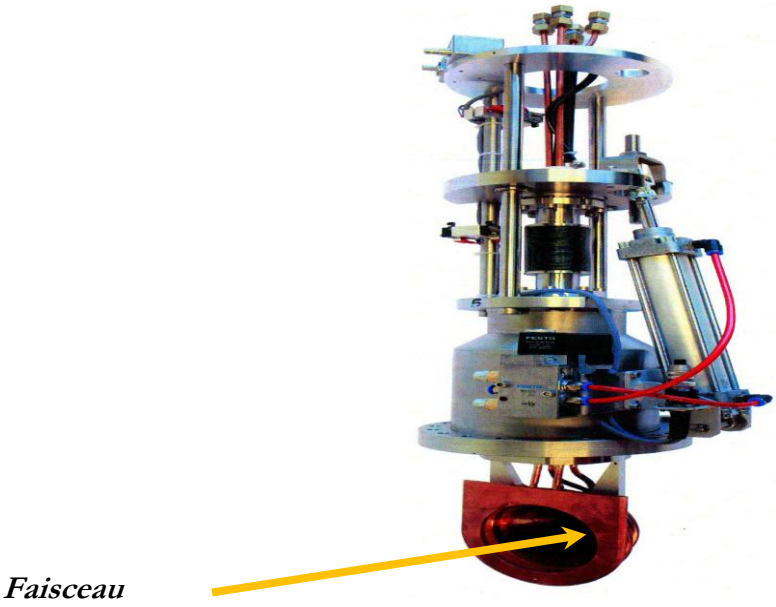


Figure 12 : Cage de Faraday

Cette électronique permettra de :

- Mesurer des courants compris dans une gamme de 5µA à 10mA
- Tester le dispositif de mesure
- Fournir un moyen d'étalonnage via un générateur de courant (unique pour l'ensemble des coupelles)

Une première cage de Faraday (CF11) sera installée en sortie de la source afin de mesurer l'intensité du faisceau. Elle sera indispensable pour les premiers tests de la source deuton à Saclay.

3.3 Les fentes FH13 et FV13 sur la ligne LBE1

Les fentes équipent la ligne LBE1, la ligne LBEC, ainsi que la ligne LME de l'injecteur de SPIRAL2. Elles ont pour principale mission de réduire une partie du halo présent autour du cœur du faisceau dans l'accélérateur. Leur déplacement est assuré par un moteur avec une précision de 0.1mm. Une fente est composée de deux joues, chaque joue comportant un moteur. Lorsque nous avons une **F**ente **V**erticale

(***FV***) suivie d'une **F**ente **H**orizontale (***FH***) comme sur le schéma ci-dessous, cela s'appelle un « jeu de fentes ».

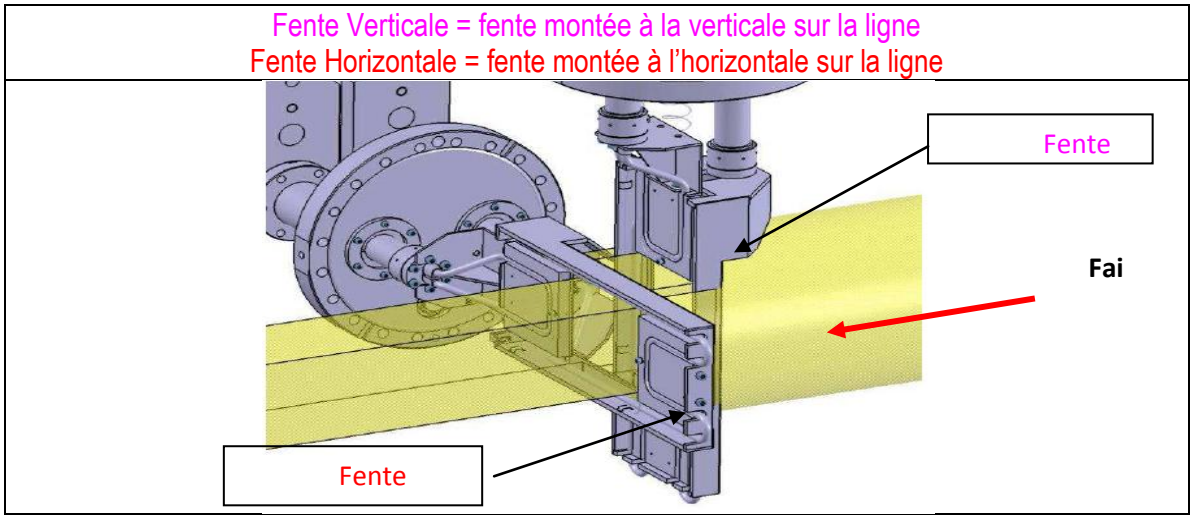


Figure 13 : Jeux de fentes

Il n'y aura pas de fentes sur la ligne LBE2/Deutons, sur laquelle la source de Deutons se trouve. Cependant dans ce rapport je vous parlerai du jeu de fentes FH13 & FV13 de la ligne LBE1 que j'ai développé pour une autre ligne de l'accélérateur.

VI - Présentation technique du matériel

1- La source deuton (LBE2)

1.1 Le magnétron

1.2 Le générateur d'impulsions

1.3 Adaptateur d'impédance

1.4 Alimentation Haute-Tension (HT)

1.5 Alimentation Haute-Tension Electrode Intermédiaire (HTEI)

1.6 Alimentation Repousseur d'Electrons (RE)

1.7 Contrôleur d'injection de gaz

2- La cage de Faraday CF11 (LBE2)

3- Les fentes FH13 & FV13 (LBE1)

3.1 Schéma synoptique du contrôle/commande

3.2 Les spécifications générales

4- Présentation de la configuration Hardware

4.1 Châssis VME

4.2 Cartes VME

5- Présentation de la plateforme logicielle

5.1 Système Linux

5.2 Distribution VxWorks

5.3 Distribution EPICS

6- Architecture réseau

6.1 A Saclay

1- La source deuton (LBE2)

Dans cette partie seront présentés les différents instruments à piloter afin de réaliser un bilan des Entrées/Sorties permettant d'assurer le pilotage de la source de Deutons. Toute la ligne LBE2 sera contrôlée par un même châssis VME dont le nom est : « **VMELB2** ».

1.1 Le magnétron

La source de Deutons est une source de type ECR fonctionnant à 2.45 GHz. La puissance **R**adio **F**réquence (***RF***) est fournie par un magnétron. Cette puissance est injectée dans la source via des guides d'ondes rectangulaires. La chaîne RF comprend entre autre un adaptateur d'impédance.
Deux modes de fonctionnement sont prévus :

- Mode continu
- Mode pulsé.

Pour piloter le magnétron :

En mode pulsé, la puissance RF est pulsée avec des cycles utiles pouvant aller de 200ms à 1s et des durées de pulse minimum de quelques centaines de microsecondes. La puissance pulsée est pilotée par un créneau 0-10 V ajustable, généré par un générateur d'impulsions. De plus tout un ensemble d'informations concernant l'état et les défauts sont accessibles et doivent être accessibles à l'utilisateur final.

Caractéristiques :

Puissance de sortie	1,9 kW
Fréquence de travail	2450 MHz
Haute Tension	4000 V
Interface Contrôle / Commande	* 1 x SubD37 analogique & contacts secs * 2 x SubD9 : RS232 ou RS485 * 1 x BNC pour la commande analogique 0/10V
Protection magnétron	* Défaut des circuits eau et air de refroidissement * Courant ou tension anodique trop élevé * Puissance réfléchie trop importante
Protection alimentation HT	* Court circuit de la haute tension (défaut magnétron ou circuit ouvert) * Surtension (circuit HT ouvert) * Surchauffe des transistors * Défaillance du circuit de refroidissement
Sécurité extérieur	* Contact sec

Tableau 1 : Caractéristiques magnétron

L'instrument peut se piloter en mode RS232, RS485 ou en analogique via le connecteur SUBD37. Le pilotage par la ligne RS232/RS485 nécessite une carte VME spécifique. Cette carte ne peut comporter que quatre connecteurs, soit quatre connexions possibles vers des instruments. Il faut savoir que la place dans le châssis VME est limitée en nombre de cartes c'est pourquoi mon choix s'est porté sur le pilotage en analogique via le connecteur SUB37.

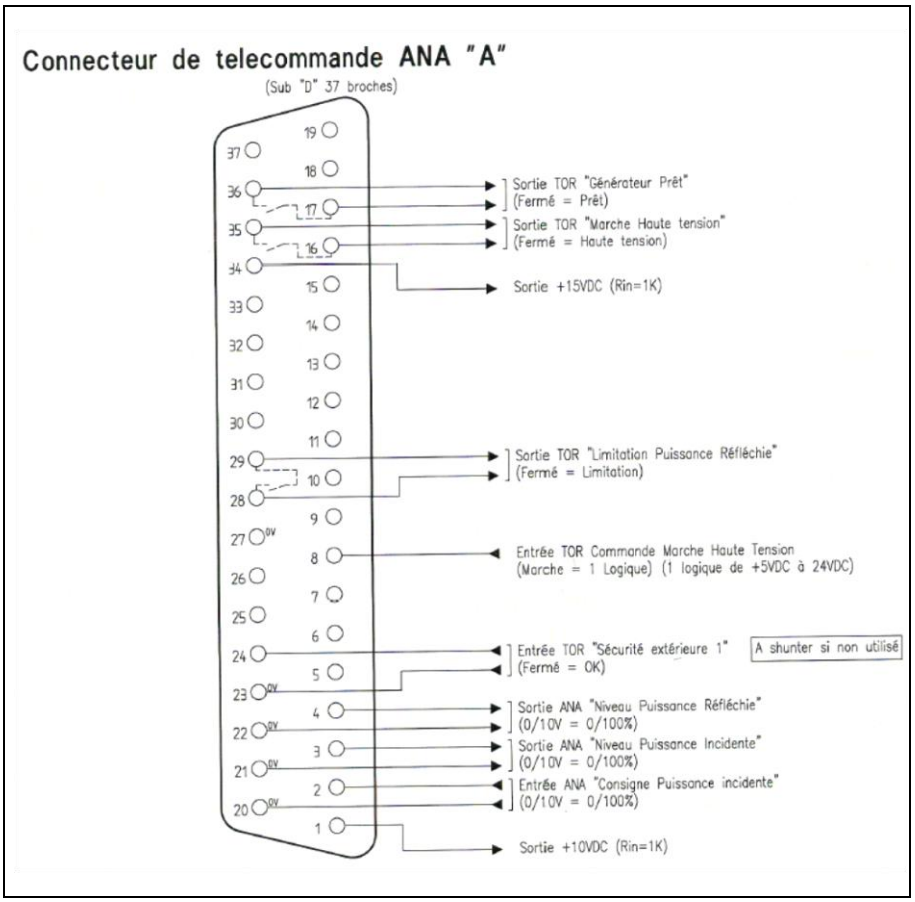


Figure 14 : Connecteur analogique télécommande magnétron

Il faut donc :

- Une carte d'entrées/sorties digitales pour toutes les informations « tout ou rien » donnant l'état du magnétron,
- Une carte d'entrées analogiques de type « **A**nalog to **D**igital **C**onverter » (***ADC***) permettant d'acquérir les différents niveaux de puissance réfléchie et incidente,
- Une carte de sorties analogiques de type « **D**igital to **A**nalog **C**onverter » (***DAC***) permettant d'appliquer une tension de consigne pour la puissance incidente.

1.2 Le générateur d'impulsions

Ce générateur permet de générer l'impulsion pour le magnétron. Il est pilotable par l'interface TCP/IP. Il est relié à la masse, comme le châssis VME.

Pour piloter le générateur d'impulsions :

Par l'intermédiaire de la connexion TCP/IP il est possible de configurer un type de signal en forme de pulse, avec une amplitude, une fréquence de répétition, la largeur du pulse, etc…

Caractéristiques :

Gamme de fréquence	1 mHz à 10 MHz
Résolution	1 mHz
Type de signal	Waveform, Pulse, Square, Burst,etc…
Interface Contrôle / Commande	LAN remote interface

Tableau 2 : Caractéristiques générateur d'impulsions

Il faut donc :

- Une connexion TCP/IP pour interagir avec l'instrument

1.3 Adaptateur d'impédance

L'adaptateur d'impédance est un composant en guide d'onde destiné à diminuer la puissance réfléchie que voit le magnétron, c'est-à-dire augmenter la puissance transmise vers la source, en adaptant à l'optimum l'impédance de la source à celle du guide d'onde et celle du magnétron.

Le principe consiste en un système de 4 diodes qui mesure le coefficient de la charge (*RHO*), en coordonnées cartésiennes, sous la forme de deux signaux X et Y. Chaque signal commande un moteur, dans un sens de rotation ou l'autre suivant la polarité du signal. Chaque moteur actionne deux pistons, dont l'enfoncement, le diamètre, et l'entraxe sont précis, assurant l'adaptation pour une gamme d'impédance visualisé sur une ligne du diagramme de Smith.

Pour piloter l'adaptateur d'impédance :

La source peut fonctionner dans deux modes distincts : continu et pulsé.

En mode continu, que nous soyons en local ou en distant, le réglage des pistons peut être effectué automatiquement ou manuellement. Par contre, en mode pulsé, le réglage des pistons ne pourra s'effectué que manuellement.

Pour le déplacement des moteurs, il suffit d'appliquer une tension sur chaque voie, X et Y.

Caractéristiques :

Fréquence d'application	2450 MHz +/- 25 MHz
Interface Contrôle / Commande	Connecteur 15 broches
Mode de fonctionnement	Automatique & Manuel
Commande/Lecture moteur	0 - 10 V

Tableau 3 : Caractéristiques adaptateur impédance

L'instrument se pilote en analogique via un connecteur SUBD15.

En résumé il faut être capable de :

- Générer deux tensions analogiques pour appliquer des consignes de déplacement pour X et Y,
- Mesurer deux tensions analogiques afin de déterminer le coefficient de réflexion de la charge (RHO), en coordonnées cartésiennes, sous la forme de deux signaux X & Y,
- Fermer ou ouvrir un contact sec afin de sélectionner le mode Automatique ou manuel

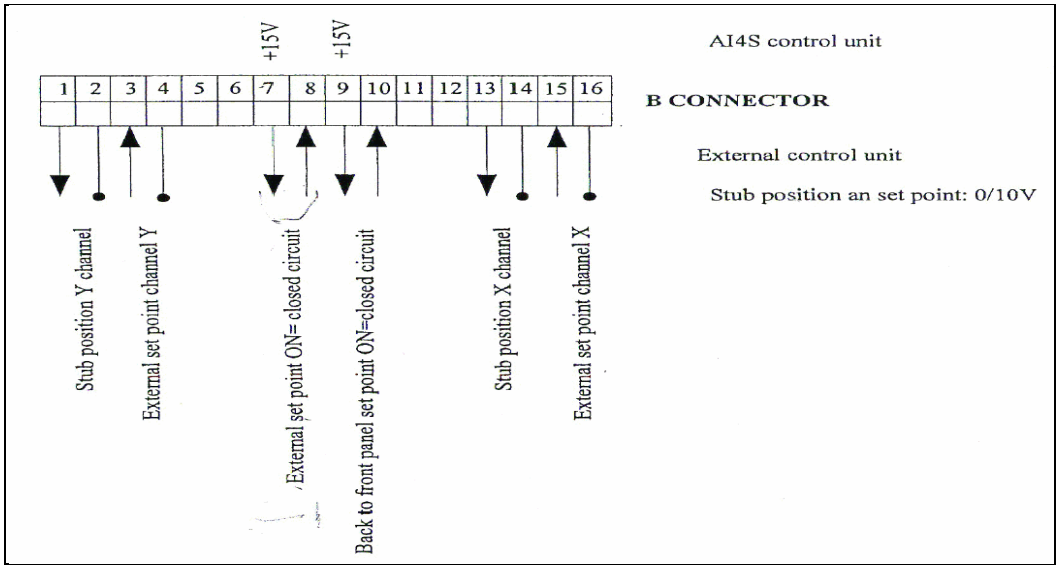


Figure 15 : Connecteur télécommande de l'adaptateur d'impédance

Il faut donc :

- Une carte d'entrées analogiques de type « **A**nalog to **D**igital **C**onverter » (***ADC***) permettant de mesurer les deux tensions analogiques afin de déterminer le coefficient de réflexion de la charge(RHO)

- Une carte de sorties analogiques de type « **D**igital to **A**nalog **C**onverter » (***DAC***) pour appliquer des consignes de déplacement pour X et Y,
- Une carte d'entrées/sorties digitales pour sélectionner le mode de pilotage

1.4 Alimentation Haute-Tension (HT)

L'énergie maximale des ions légers à l'entrée du RFQ est fixée à 40 KeV. La source d'ions est donc placée sur un bâti isolé de la masse et polarisée à une tension pouvant atteindre 40 kV à l'aide d'une alimentation Haute Tension 50 kV. Cette alimentation fonctionne dans un environnement ou les risques de claquages (HT/masse ou claquages inter-électrodes) peuvent être fréquents. L'alimentation risque donc d'être mise en court-circuit, et c'est pourquoi elle est protégée contre ce type de défauts.
Pour limiter ce genre de pannes en cas de claquage, l'alimentation est placée dans un châssis dit **CEM** (**C**ompatibilité **E**lectromagnétique et **M**esure de champ) avec des connecteurs munis de capots métalliques. L'alimentation est pilotable en Modbus/Tcp. (***Voir Annexe1***)

Pour piloter l'alimentation HT :
Dans les registres adéquats, il faut :

- Ecrire la consigne de tension et de courant,
- Lire la tension et le courant résultant,
- Lire l'état de l'alimentation,
- Mettre sous tension et hors tension

Caractéristiques :

Tension	50 kV
Courant	20 mA
Type de régulation	Tension
Interface Contrôle / Commande	Modbus / TCP (voir Annexe)
Sécurité extérieur	Oui (Via contacts secs)
Interrupteur Local / Distance	Oui

Tableau 4 : Caractéristiques alimentation HT

Plan de mémoire de l'instrument :
Tous ces paramètres sont accessibles en lecture/écriture. Pour chaque adresse correspond un mot de 16 bits.

Il existe d'autres paramètres entre l'adresse 17 et 28 uniquement accessible en lecture, mais ceux-ci ne sont pas très utiles, à part éventuellement le temps de montée de l'alimentation. Mais l'utilisateur ne peut pas modifier ce paramètre, puisqu'il ne peut être que lu. Ce mapping mémoire est identique pour les trois alimentations.

adresse 0	alim ON	écrire 1 à cette adresse pour enclencher l'alim
adresse 1	alim OFF	écrire 1 à cette adresse pour couper la puissance.
adresse 2	arrêt général	coupe la puissance et les tensions auxiliaires lorsqu'elles sont télécommandables.
adresse 3	Surveillance	écrire 1 à cette adresse pour enclencher la surveillance et 0 pour la couper.
adresse 4	fourchette de surveillance	écrire la fourchette de courant ou tension dans laquelle l'équipement doit rester. Si l'équipement sort de cette fourchette, il générera une alarme. La valeur de fourchette est codée sur 16 bits non signés. FFFF → valeur1 max.
adresse 5	rampe	temps de montée en secondes de 0 à la valeur max (L'alim ne prend en compte cette valeur de rampe que si elle est supérieure à la rampe locale mémorisée lors de la configuration de l'alim)
adresse 6 et adresse 7	consigne n°1	1 ère valeur de consigne de l'alimentation. C'est un entier signé codé sur 32 bits : 7FFFFFFF → valeur1 max.
Adresse 8 et adresse 9	consigne n°2	2 ème valeur de consigne de l'alimentation. C'est un entier signé codé sur 32 bits : 7FFFFFFF → valeur2 max.
adresse 10 et adresse 11	valeur principale	lecture de la valeur principale codée sur 32 bits signés : 7FFFFFFF → valeur1 max
adresse 12 et adresse 13	valeur secondaire	lecture de la valeur secondaire codée sur 32 bits signés : 7FFFFFFF → V maxi pour une alimentation de courant et Imaxi pour une alimentation de tension.
adresse 14	état de l'alim	mot d'état : Puissance, Polarité, status DAC, status ADC, busy, Alimentation prête, etc…
adresse 15 et adresse 16	défauts	Défauts : Charge, manque phase, portes ouvertes, températures de redresseurs, etc …

Tableau 5 : Plan de mémoire du contrôleur d'alimentation

Il faut donc :

- Une connexion TCP/IP pour interagir avec l'instrument

1.5 Alimentation Haute-Tension Electrode Intermédiaire (HTEI)

Dans le système d'extraction du faisceau de D+, une électrode intermédiaire permet d'optimiser la surface émissive (le ménisque) de manière à limiter les pertes sur les électrodes quelque soit le courant extrait. Cette électrode est alimentée par une alimentation HT. De même que l'alimentation HT, l'alimentation HTEI sera placée dans un châssis CEM avec des connecteurs munis de capots métalliques afin d'être protégée contre les claquages. L'alimentation est pilotable en Modbus/Tcp. (***Voir Annexe1***)

Pour piloter l'alimentation HTEI :

Dans les registres adéquats, il faut :

- Ecrire la consigne de tension,
- Lire la tension et le courant résultant,
- Lire l'état de l'alimentation,
- Mettre sous tension et hors tension

Caractéristiques :

Tension	30 kV
Courant	10 mA
Type de régulation	Tension
Interface Contrôle / Commande	Modbus / TCP (voir Annexe)
Sécurité ext.	Oui (Via contacts secs)
Interrupteur Local / Distance	Oui

Tableau 6 : Caractéristiques alimentation HTEI

Il faut donc :

- Une connexion TCP/IP pour interagir avec l'instrument

1.6 Alimentation Repousseur d'Electrons (RE)

Dans le système d'extraction du faisceau de D+, une électrode légèrement négative permet d'interdire la remontée des électrons émis par l'interaction du faisceau avec le gaz résiduel. De même que les alimentations précédentes, elle sera placée dans un châssis CEM avec des connecteurs munis de capots métalliques afin d'être protégée contre les claquages. L'alimentation est pilotable en Modbus/Tcp. (***Voir Annexe1***)

Pour piloter l'alimentation RE:

Dans les registres adéquats, il faut :

- Ecrire la consigne de tension,
- Lire la tension et le courant résultant,
- Lire l'état de l'alimentation,
- Mettre sous tension et hors tension

Caractéristiques :

Tension	- 3 kV
Courant	10 mA
Type de régulation	Tension
Interface Contrôle / Commande	Modbus / TCP (voir Annexe)
Sécurité ext.	Oui (Via contacts secs)
Interrupteur Local / Distance	Oui

Tableau 7 : Caractéristiques alimentation RE

Il faut donc :

- Une connexion TCP/IP pour interagir avec l'instrument

1.7 Contrôleur d'injection de gaz

Le PR4000B est un contrôleur d'injection de gaz. Il permet d'injecter dans la source soit du Deutérium ou du dihydrogène. Ce contrôleur permet de piloter deux types de Gaz en même temps, ce qui donne la possibilité de faire un mélange. Dans notre cas nous utiliserons que l'un ou l'autre. Ce contrôleur fonctionne en mode local ou en mode distant via le protocole série RS232. Cet instrument est localisé sur la plateforme HT c'est pourquoi il doit être isolé à la masse.

Pour piloter le contrôleur d'injection de gaz :

Les principales commandes à effectuer sont:

- Ecrire la consigne de débit,
- Lire le débit en sortie,
- Contrôler l'état de la vanne, ouverte ou fermée
- Visualiser le statut général de l'appareil

Il faut donc :

- Une connexion RS232

2- La cage de Faraday CF11 (LBE2)

L'électronique est composée en deux parties :

- Une partie mesure
- Une partie test équipée d'un boitier permettant de générer des courants afin de tester la chaine de mesure

2.1 La partie mesure

L'électronique comporte deux entrées analogiques qui correspondent à :

- Une gamme pour les courants faibles de l'ordre de la centaine de µA (10V/250µA)
- Une gamme pour les courants forts de l'ordre du mA (10V / 10mA)

2.2 La partie test

Le boitier électronique permet de générer deux courants, un courant fort et un courant faible afin de tester les deux gammes. Pour chacun des deux tests une entrée de type TTL actionne le test, et une sortie de type TTL reflétant l'état de la commande test afin de vérifier que la commande test est bien prise en compte par l'équipement.

Caractéristiques :

Désignation	Entrée ou Sortie	Type I/O
Mesure du courant moyen Fort	Entrée	Analogique
Mesure du courant moyen Faible	Entrée	Analogique
Commande test de courant Fort	Sortie	TOR
Commande de test de courant Faible	Sortie	TOR
Relecture test courant IFort	Entrée	TOR
Relecture test courant IFaible	Entrée	TOR

Tableau 8 : Caractéristiques cage de Faraday (électronique)

Pour piloter la cage de Faraday :

Les principales commandes à effectuer sont:

- Lecture du courant moyen faible et fort
- Effectuer des tests

Il faut donc pour le pilotage :

- Une carte d'entrées analogiques de type « **A**nalog to **D**igital **C**onverter » (***ADC***) permettant de mesurer les différents courants
- Une carte d'entrées/sorties digitales commande et relire l'état des tests

3- Les fentes FH13 & FV13 (LBE1)

Une fente est composée de deux joues, et chaque joue est pilotée par un moteur. Chaque carte moteur est associée à un contrôleur. Ces contrôleurs sont regroupés dans un châssis appelé « châssis moteur ». Ce châssis peut comporter huit contrôleurs et une carte d'interface JBus/RS485, protocole qui sera détaillé par la suite. Un boitier COMETH servant de passerelle Modbus/TCP <-> RS232 est connecté entre la carte JBus et le châssis VME (*Voir Annexe1*) afin de simplifier la connectique.

3.1 Schéma synoptique du contrôle/commande

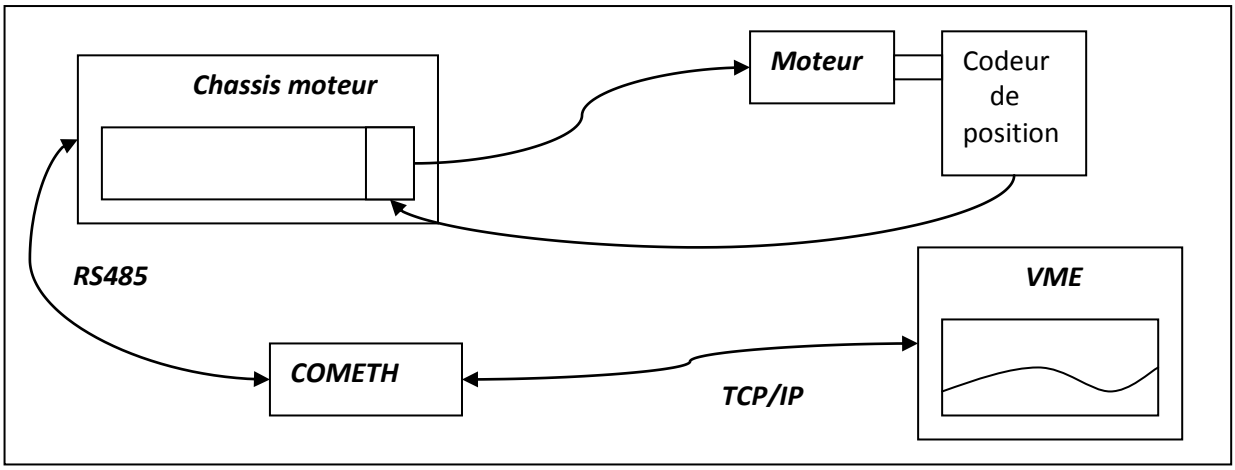


Figure 16 : Schéma synoptique du contrôle/commande des fentes

Le châssis VME transmet des informations au boitier COMETH en Modbus par la liaison TCP, qui lui-même adapte l'information en RS485 pour l'envoyer à la carte JBus. Cette dernière redistribue l'information aux contrôleurs de chaque moteur. Un codeur de position permet d'établir la position absolue de chaque moteur qui est envoyé aux contrôleurs afin d'en asservir son déplacement. Il a été ajouté par la suite une sonde de température pour chaque moteur, soit pour chaque joue.

3.2 Les spécifications générales

3.2.1 La motorisation

Le châssis moteur assurant la motorisation des fentes, peut accueillir 8 cartes de commande et une carte d'interface JBUS-CAN. JBus est un réseau local industriel (bus de terrain) variante de Modbus, ce qui veut dire que le protocole Jbus utilise une partie du protocole Modbus (Voir Annexe1). Il est composé d'un maitre et de plusieurs esclaves.

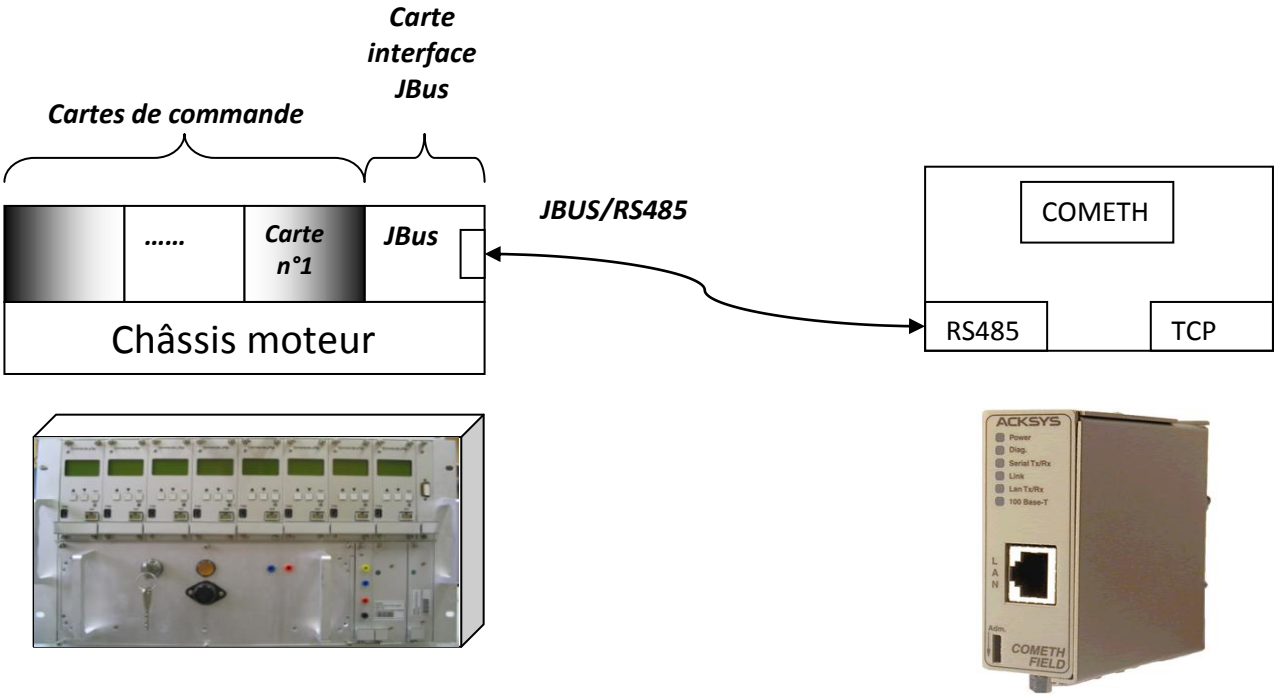


Figure 17 : Châssis moteur des fentes + COMETH

3.2.2 L'interface JBUS

La trame JBus est composée :

- N° esclave (1 Octet)
- Code fonction modbus (1 Octet)
- Données (n Octet)
- Code Checksum (2 Octets)

Les fonctions Modbus acceptées par la carte JBus intégré dans le châssis sont :

Fonction 03 : Lecture 1 mot
Fonction 04 : Lecture N mots
Fonction 06 : Ecriture 1 mot
Fonction 16 : Ecriture N mots

Prenons un premier exemple d'écriture d'une trame Modbus (1 mot)

N° esclave : 1 octet code hexadécimal (1 à 32)
Fonction J_BUS : fonction 06
Adresse : adresse du registre à écrire
Nombre de mots : 1
CRC16

Et un second exemple de lecture d'une trame Modbus (1 mot)

N° esclave : 1octet code hexadécimal (1 à 32)
Fonction J_BUS : fonction 03
Nombre de bytes : 02
Mots : 1 mot de 16 bit
Nombre de mots : 1
CRC16

3.2.3 Adressage JBUS des registres du contrôleur

Variable	Mnémonique	Adresse	Lecture	Ecriture	Description
Consigne	Cons	0		X	Entier
Position	Pos	1	X		Entier
Mot d'état	State	2	X		Entier
Coefficient démultiplicateur	Coeff	3	X	X	Entier
Vitesse minimum	Vmin	4	X	X	Entier
Vitesse maximum	Vmax	5	X	X	Entier
Définition micro pas	Micro pas	6	X	X	Entier
Courant driver déplacement	Irun	7	X	X	Entier
Courant driver maintient	Ihold	8	X	X	Entier
Nombre de Pas	Pas	9		X	Entier
Adresse CAN	CAN	10	X		Entier
Adresse JBUS	JBUS	11	X		Entier
Délaisretournement ligne	DRTL	12		X	Entier
Délais avant mvt asservit	DMVT	13	X	X	Entier
Initialisation	Init	14		X	Entier

Tableau 9 : Adressage JBUS des registres du contrôleur de fentes

Il faut donc pour le pilotage :

- Une connexion TCP/IP pour interagir avec le boitier COMETH
- Un boitier COMETH

4- Bilan des Entrées/Sorties source deuton & Cage de Faraday CF11

Après avoir correctement établi la liste des instruments à piloter, et avoir listé les différentes informations utiles au bon fonctionnement de la source, voici un tableau récapitulatif des besoins en entrées/sorties pour le châssis ***VMELB2*** du point de vue du contrôle/commande afin de choisir au mieux les cartes appropriées. Dans ce décompte les fentes FH13 et FV13 ne seront pas incluses, car celles-ci ne sont pas sur la même ligne que la source de deuton et la cage de Faraday CF11.

Au final il faut pour l'ensemble du contrôle/commande :

Magnétron

Désignation	Entrée ou Sortie	Type I/O
Sortie TOR « Générateur Prêt »	Entrée	TOR
Sortie TOR « Marche Haute Tension »	Entrée	TOR
Sortie TOR « Limitation Puissance réfléchie »	Entrée	TOR
Entrée TOR « Commande Marche Haute Tension »	Sortie	TOR
Entrée TOR « Sécurité extérieure »	-----	----
Sortie ANA « Puissance incidente mesurée »	Entrée	Analogique
Sortie ANA « Puissance réfléchie mesurée »	Entrée	Analogique
Entrée ANA « Consigne Puissance Incidente »	Sortie	Analogique

Adaptateur d'impédance

Désignation	Entrée ou Sortie	Type I/O
Sortie ANA « Lecture Voie X »	Entrée	Analogique
Entrée ANA « Consigne voie X »	Sortie	Analogique
Entrée TOR « Mode Local/Distant »	Sortie	TOR
Sortie ANA « Lecture Voie Y »	Entrée	Analogique
Entrée ANA « Consigne voie Y »	Sortie	Analogique

Alimentation HT

Liaison	Protocole
TCP	MODBUS TCP

Alimentation HTEI	
Liaison	Protocole
TCP	MODBUS TCP
Alimentation ER	
Liaison	Protocole
TCP	MODBUS TCP
Contrôleur d'injection de gaz	
Liaison	Protocole
TCP/Série	STREAM TCP (COMETH) /RS232
Générateur d'impulsions	
Liaison	Protocole
TCP	STREAM TCP

Cage de Faraday CF11		
Désignation	Entrée ou Sortie	Type I/O
Mesure du courant moyen Fort	Entrée	Analogique
Mesure du courant moyen Faible	Entrée	Analogique
Commande test de courant Fort	Sortie	TOR
Commande de test de courant Faible	Sortie	TOR
Relecture test courant IFort	Entrée	TOR
Relecture test courant IFaible	Entrée	TOR

Tableau 10 : Récapitulatif des entrées/sorties du CC de la source deuton

En résumé il faut,

- Entrées TOR
- Sorties TOR
- Entrées analogiques
- Sortie analogiques
- connexions TCP/IP (Modbus)
- connexion série RS232
- connexion TCP/IP

5- Présentation de la configuration Hardware

5.1 Châssis VME

Le contrôle/commande du projet SPIRAL2 implique la mise en œuvre de châssis au standard VME. Le retour d'expérience lors de l'exploitation de la machine GANIL actuelle a conduit à définir un ensemble de spécifications visant à faciliter l'exploitation et la maintenance au quotidien.

Figure 18 : Châssis face avant, P.Ouverte

Figure 19 : Face avant, P.Fermé

Figure 20 : Châssis face arrière

5.2 Cartes VME

Après avoir correctement réalisé l'état des lieux concernant les entrées/sorties, il est possible maintenant de déterminer le type et le nombre de cartes.

Par souci de standardisation, des cartes VME ont été sélectionnées pour tout le projet Spiral2. Parmi ces cartes sélectionnées, mes choix ont été les suivants.

5.2.1 Pour les entrées analogiques

Ces entrées ont des caractéristiques communes :

- Gamme de tension comprise entre 0 et 10 V
- Fréquence oscillatoire du signal très faible ce qui nécessite une fréquence d'échantillonnage de quelques KHz
- Pas besoin d'une grande précision de la mesure

Le choix s'est porté :

- sur une carte ***VME ICV150*** dont ses caractéristiques principales sont :

- 32 entrées analogiques différentielles
- Précision du convertisseur de 16 bits
- Une fréquence d'échantillonnage de 200 KHz
- Driver EPICS disponible et fonctionnel

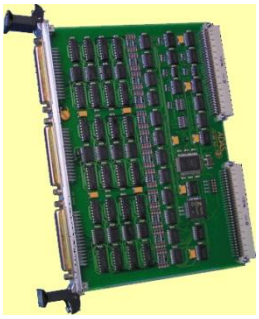
5.2.2 Pour les entrées/sorties digitales TOR

Sur le même principe que les entrées analogiques, ces entrées ont des caractéristiques communes, dont une importante : ce sont des contacts secs. A l'aide d'un montage électronique, l'adaptation des différentes Entrées/Sorties peut fonctionner format TTL.

Le choix s'est porté :

- sur une carte ***VME ICV196*** dont ses caractéristiques principales sont :

- 96 digital I/O
- Entrées / Sorties compatibles TTL
- Puissance de commutation élevée 70V/100mA



5.2.3 Pour les sorties analogiques

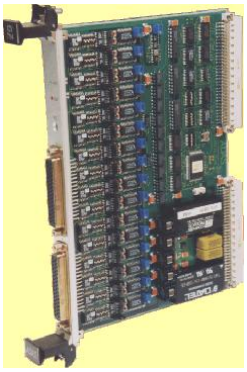
Ces sorties ont une caractéristique commune :

- Gamme de tension à générer comprise entre 0 et 10 V

Le choix s'est porté :

- sur une carte ***VME ICV714*** dont ses caractéristiques principales sont :

- 16 sorties analogiques
- Précision du convertisseur de 12 bits
- Driver EPICS disponible et fonctionnel



5.2.4 La carte contrôleur CPU

Cette carte permet de fonctionner avec l'OS temps réel Vx Works. C'est sur cette carte que tourne l'IOC.

Le choix s'est porté :

MVME 5500 (Motorola)		
Désignation		
Processeur	– Single 1 GHz MPC7455 processor – Bus clock frequency at 133 MHz	
L3 Cache	– 2MB using DDR SRAM – Bus clock frequency at 200 MHz	
Flash	– 8MB Flash soldered on board – 32MB expansion Flash soldered on board	
NVRAM	– 32KB provided by MK48T37	
Périphérique externe	– 1x 10/100/1000 BaseT Ethernet interface, – 1 x 10/100 BaseT Ethernet interface	

Remarque:

Cette carte nous offre deux interfaces réseau. Le problème c'est qu'il manque une liaison RS232 pour piloter le contrôleur d'injection de gaz. Sachant qu'il n'y aura qu'un seul instrument avec une interface de ce type, il est inutile d'investir dans une carte VME spécifique RS232. Pour remédier à ce problème, nous allons utiliser un port série déporté. C'est un boitier qui comporte en entrée une interface réseau, et en sortie une connexion RS232.

Pour communiquer avec ce port série déporté, il suffit de passer par le réseau, et de communiquer avec l'instrument derrière cette passerelle comme si il fonctionnait en mode TCP. Cette passerelle est appelé COMETH. (***Voir Annexe 2***).

En résumé il faut une carte,

- ICV150 pour les entrées Analogiques
- ICV 196 pour les contacts secs
- ICV 714 pour appliquer une consigne de tension
- MVME 55500 pour piloter le tout

Et un boitier,

- COMETH pour la connexion rs232

V - Experimental Physics and Industrial Control System (EPICS)

1 - Présentation

2 - Principe de fonctionnement

2.1 L'architecture

2.2 La Process Variable (PV)

2.3 L'Input Output Controler (IOC)

2.4 La base de données EPICS (DB)

2.5 Le séquenceur

2.6 Device support

2.7 Channel Acess

2.8 Le script de démarrage : st.cmd

2.9 Les applications clientes

2 - Présentation de la plateforme logicielle pour SPIRAL2

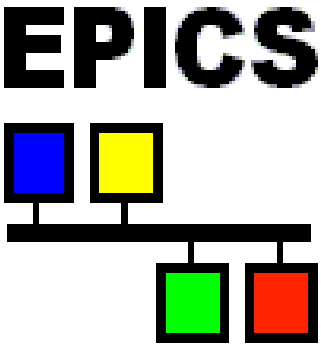
3.1 Système Linux

3.2 Distribution VxWorks

3.3 Distribution EPICS

3- Architecture réseau

4.1 A Saclay



1- Présentation

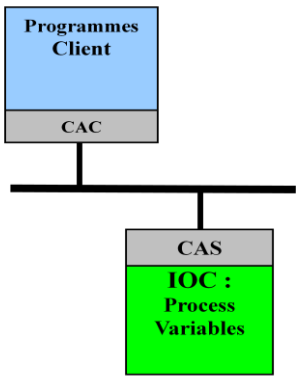
Experimental **P**hysics and **I**ndustrial **C**ontrol **S**ystem (***EPICS***) est un ensemble d'outils logiciels et d'applications Open Source qui fournissent une infrastructure logicielle pour la construction de systèmes de contrôle distribué tels que les accélérateurs de particules, les grands télescopes, ou tout autre grande expérience scientifique. C'est une collaboration qui a débuté en 1989 entre deux laboratoires : Los Alamos (***LANL***) et Argonne (***ANL***) aux Etats Unis. A l'heure actuelle, EPICS est devenu une collaboration internationale et est utilisé par plus de 150 laboratoires dans le monde. De nombreuses expériences sont construites avec ce système.

2- Principe de fonctionnement

2.1 L'architecture

C'est un système de contrôle basé sur une architecture client/serveur en réseau. Le serveur, l'**I**nput **O**utput **C**ontroler (***IOC***), s'occupe de rendre accessible ses données appelées Process Variables (***PV***). Une Process Variable peut être associée à une entrée/Sortie matériel, ou pas. Celle-ci représente la donnée élémentaire distribuée par l'IOC. Ensuite, via le protocole de communication **C**hannel **A**ccess (***CA***), l'application client accède aux Process Variables. Tout cet ensemble est piloté par un système d'exploitation multitâches destiné aux applications temps réel appelé ***VxWorks***.

CAC : Channel Access Client



CAS : Channel Access Server

Figure 21 : Architecture EPICS

2.2 La Process Variable (PV)

Une PV est unique sur le réseau dans lequel se trouve l'IOC. Elle permet d'identifier un composant lié à un équipement de l'installation en question. L'information qu'elle peut donner est par exemple l'état d'un instrument, ou encore la lecture d'une valeur mesurée, ou bien encore l'écriture d'une consigne à appliquer sur un instrument.

2.3 L'Input Output Controler (IOC)

Le cœur de l'IOC est composé : d'une base de données (***DB***) temps réel qui génère l'activité sur l'IOC, d'un séquenceur qui exécute du code compilé, d'un ensemble de drivers pour communiquer avec le hardware, et d'un serveur permettant de diffuser les PV via le protocole de communication Channel Access. Un système de contrôle/commande EPICS sur une expérience quelle qu'elle soit peut intégrer un ou plusieurs IOC. Un IOC peut tourner sur un châssis VME ou sur un PC. Sur un VME, un seul IOC peut être lancé. Par contre sur un PC, plusieurs IOCs peuvent fonctionner en même temps. On nomme un IOC qui tourne sur un PC, un IOC Soft.

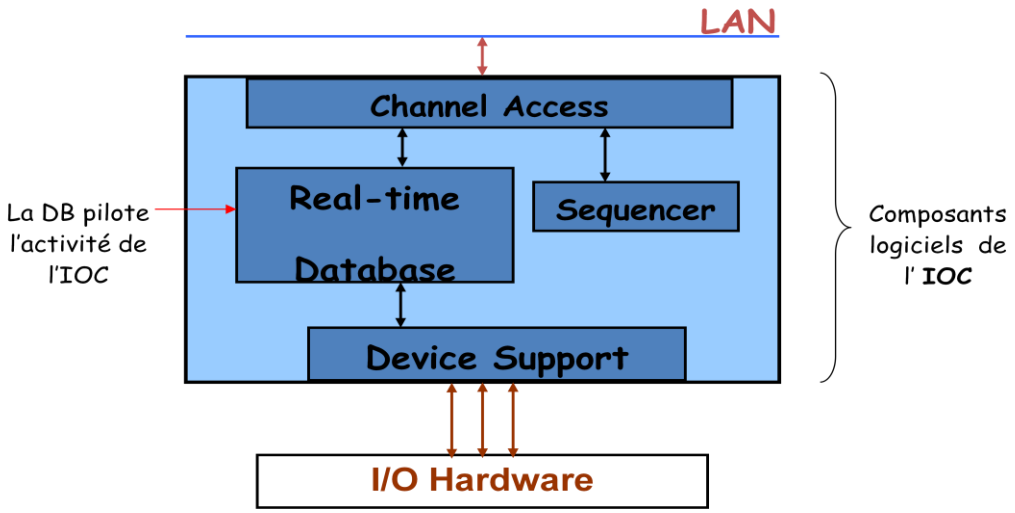


Figure 22 : Cœur de l'IOC

2.4 La base de données EPICS (DB)

La base de données EPICS est composée d'un ensemble de **Records** de différents types. Chaque Record est associé à une Process Variable.

Un Record est un objet qui est défini par :

- Un nom unique
- Un comportement défini par son type
- Des propriétés
- Une connexion sur des E/S (optionnel)
- Des liens vers d'autres Records (optionnel)

Si un Record n'est pas associé à une connexion matérielle, ce qui veut dire qu'il n'interagit que dans la base de données «EPICS», alors on le dit « soft ».

Les Records sont actifs et leur mission est régie par leur type.

Ils peuvent être classés en 4 types principaux :

- Input : *Analog In (AI), Binary In (BI), String In (SI)* …
- Output : *Analog Out (AO), Binary Out (BO) , String Out (SO)*
- Algorithme : *Calculation (CALC), Subroutine (GENSUB)*
- Personnalisé : *Profileur, Mesures de position (BPM), LLRF* …

Ils peuvent d'autre part être activés de trois manières :

- De manière cyclique
- Sur événement
- Par interruption

Après avoir correctement défini la DB, il faut l'installer dans l'IOC. Il s'agit alors d'en transférer le fichier ASCII dans sa mémoire au démarrage. Ensuite chaque Record est interprété. Par le biais du protocole de communication Channel Access, chaque Record peut être accessible en écriture et en lecture.

Exemple : Capteur de température

Nous avons un capteur qui mesure une température, et un actionneur qui sera déclenché si la température dépasse un seuil.

Nous allons donc avoir deux Records qui vont représenter ses deux objets :

- Un premier Record qui sera du type « *Analog Input* » (*ai*). Il sera associé au capteur et permettra donc de récupérer la mesure analogique de la température. La vitesse de rafraichissement sera d'une seconde. On l'appellera :

  ⇨ ***MesureTemperature***

- Un deuxième Record qui sera lui du type « *Binary Output* (*bo*) ». Il sera associé à l'actionneur et permettra de l'actionner ou pas. On l'appellera :

  ⇨ ***FermetureVanne***

Ces deux premiers Records sont reliés avec des entrée/sorties matériels.
Cependant nous ne savons pas quand prendre la décision d'activer l'actionneur si un certain seuil est dépassé. C'est pourquoi nous ajoutons un troisième Record qui sera celui-ci soft, c'est-à-dire sans aucune liaison direct avec une entrée/sortie matérielle. On l'appellera :

⇨ ***DeclenchementSeuil***

Chaque nouvelle valeur de température mesurée (cadencé dans l'exemple à 1seconde) par le Record «***MesureTemperature***», sera envoyé au Record « ***DeclenchementSeuil »*** qui vérifiera si elle ne dépasse pas un seuil prédéfini, qui dans notre cas est à 10. Si le seuil est dépassé, ce même Record envoie une information au Record « ***FermetureVanne »***, qui celui-ci sera exécuté.

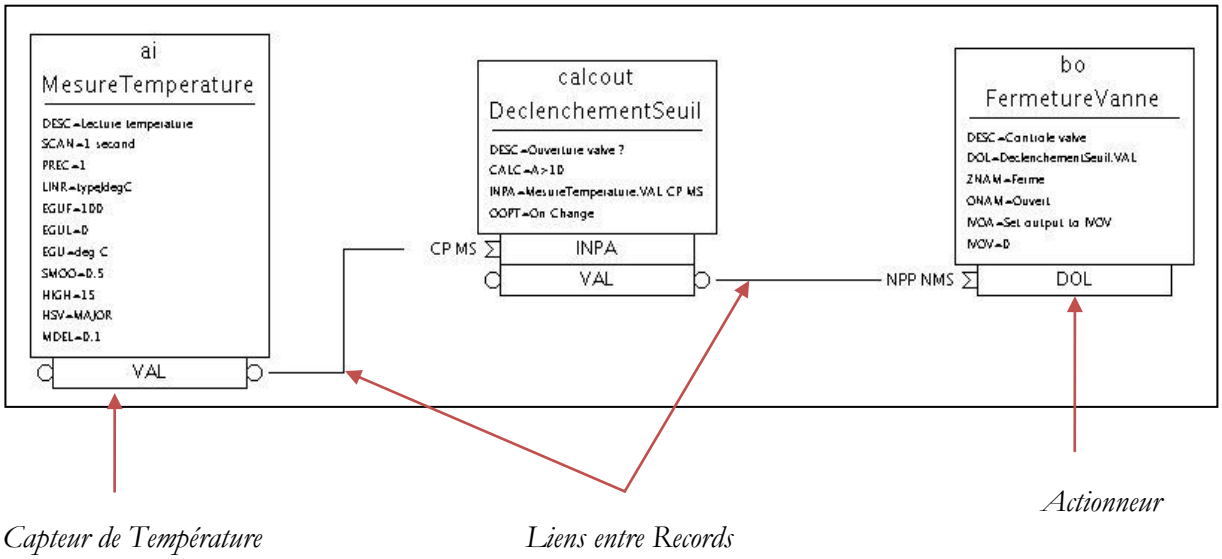


Figure 23 : exemple de base de données «EPICS»

Lorsqu'une base de données «EPICS» est instanciée plusieurs fois dans le cas ou nous avons plusieurs instruments du même type à piloter, il est possible de simplifier la tâche et d'utiliser des fichiers dits de ***substitutions***.

On ajoute un élément variable dans la base de données «EPICS» comme le Nom du Record. Puis dans le fichier de substitutions, nous spécifions le fichier de base de données «EPICS» à substituer, ainsi que tous les paramètres variables. La ligne suivante stipule les valeurs des paramètres pour la première instanciation de la base de données «EPICS». La ligne d'après correspond à une deuxième instanciation avec des valeurs différentes au moins pour le nom du Record.

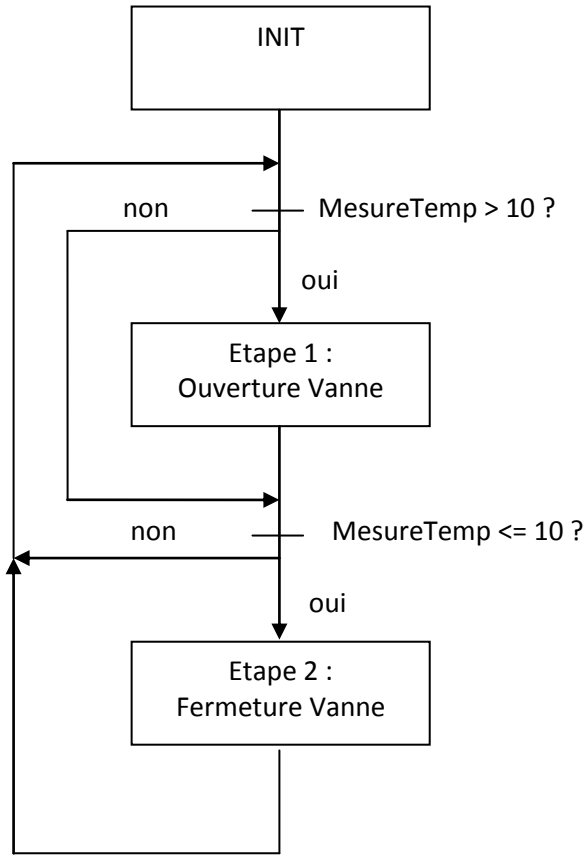
2.5 Le séquenceur

Il se trouve dans le cœur de l'IOC et permet d'exécuter du code compilé écrit en langage « ***SNL*** » (**S**tate **N**otation **L**anguage) pour décrire des opérations séquentielles. Le « ***SNL***» est un langage très proche du C. On utilise généralement du code « ***SNL***» pour créer des taches d'automatisations lors par exemple d'une procédure de démarrage, ou encore la calibration d'un instrument.

Exemple :

On reprend l'exemple de la DB précédemment cité. On garde les deux Records I/O, et le programme « SNL» va remplacer le troisième Record, qui a pour mission de vérifier la température mesurée pour activer la vanne ou pas.

```
program exampleSnc

double MesureTemp;
assign MesureTemp to "MesureTemperature";
monitor MesureTemp;

short ActionVanne;
assign ActionVanne to "ActionVanne";

Program
{
   state init {
         state low
   }

   // ETAPE 1 : Ouverture Vanne si Temp>10
   state low
   {
      when (MesureTemperature > 10.0)
      {
         ActionVanne=1;
         pvPut(ActionVanne);
      } state high
   }

  // ETAPE 2 : Fermeture vanne si Temp<10
   state high
   {
      when (MesureTemperature <= 10.0)
      {
         ActionVanne=0;
         pvPut(ActionVanne);
      } state low
   }
}
```



Comme nous venons de remarquer nous avons la possibilité de réaliser cette protection en température de deux manières distinctes, soit avec une base de données «EPICS», soit avec une procédure « ***SNL***». Mais dans cet exemple bien précis, il est préférable, juste dans un souci de simplification, de réaliser cette sécurité avec la base de données «EPICS» uniquement.

2.6 Device support

Le Device Driver est la couche logicielle d'interface entre l'IOC et le matériel. L'interfaçage avec la source se fera à partir de modules industriels au standard VME. Pour toutes ces cartes il faut être capable de les piloter correctement c'est-à-dire lire et écrire les bonnes informations. C'est le rôle du driver. La DB, ou encore des procédures « ***SNL***», vont récupérer ou envoyer par cet intermédiaire des valeurs aux différents instruments, ou encore aux convertisseurs en entrée/sortie…
Dés qu'une nouvelle carte est lancée sur le marché, il faut écrire le driver nous même. Il faut tout d'abord vérifier si celui-ci n'a pas été développé par la communauté EPICS. Si c'est le cas, il est disponible sur le site d'EPICS. Ce site est hébergé par le laboratoire d'Argonne.

Argonne NATIONAL LABORATORY

2.7 Channel Access

Channel Access est un protocole de communication qui permet aux applications clientes d'accéder aux Process Variable de l'IOC, ainsi qu'aux Process Variable d'autres IOC.

2.8 Le script de démarrage : st.cmd

C'est un fichier qui s'exécute au démarrage du châssis VME. Il a pour principale mission de charger les différents drivers des cartes utilisées, de charger ensuite les bases de données, d'initialiser différentes fonctions comme les fonctions Modbus par exemple, de lancer le séquenceur avec les fichiers « ***SNL***» déclarés, d'initialiser la sauvegarde automatique, ect …

2.9 Les applications clientes

Elles sont appelées extensions sur le site d'EPICS, et sont Open Source.
Les plus utilisées sont :

- ***EDM*** qui permet de créer des interfaces utilisateurs
- ***BURT*** qui permet de sauvegarder et de restaurer une configuration d'un accélérateur
- ***Channel Archiever*** qui permet de sauvegarder toutes les acquisitions

- ***StripTool*** qui permet d'afficher des courbes en temps réel ou en historique
- ***ArchiveViewer*** qui permet d'afficher des courbes d'historique
- ***Etc…***

Il y aussi toutes les interfaces ***Channel Access Client*** développées par la communauté en Java, C++, Phyton, Perl, etc… ce qui implique que nous pouvons par exemple développer nos applications client en JAVA…

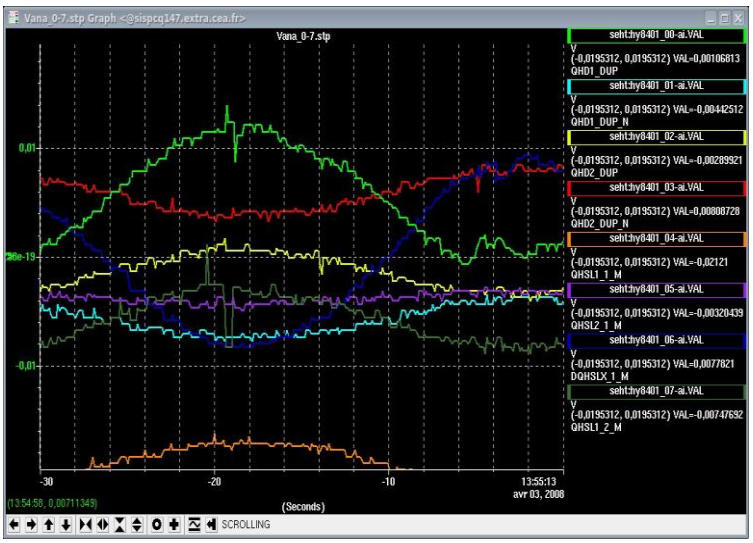


Figure 24 : l'outil StripTools

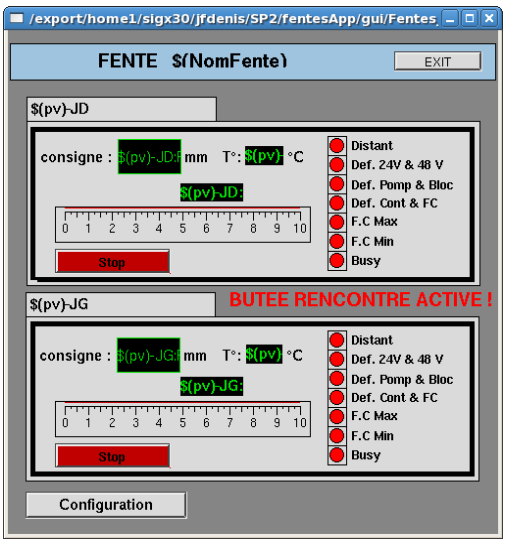


Figure 25 : Panneau de pilotage des Fentes sur EDM

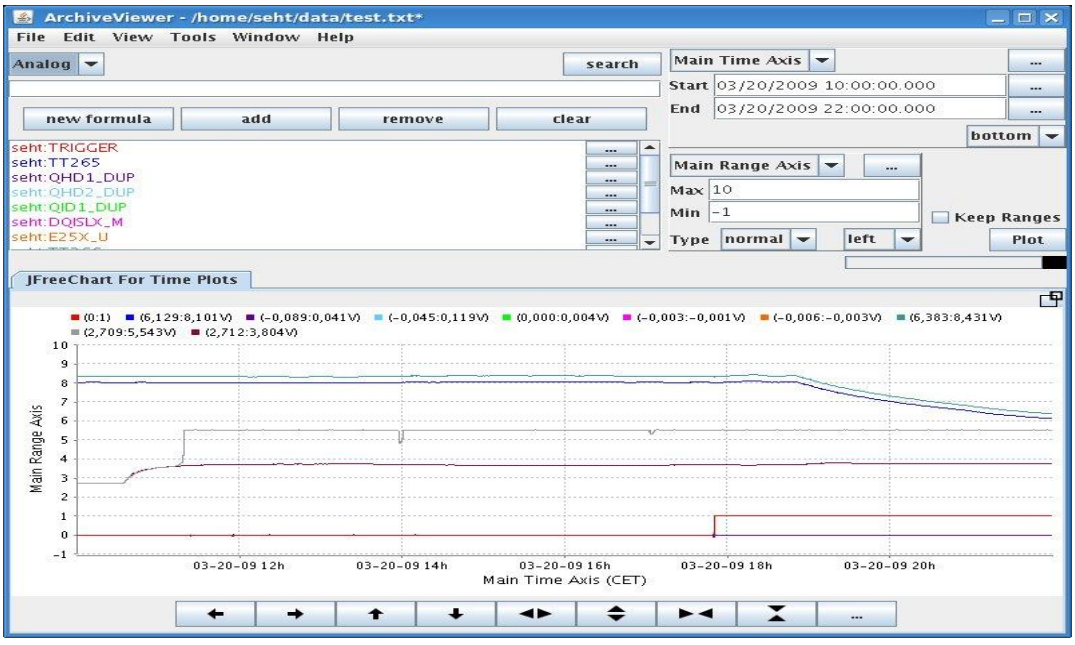
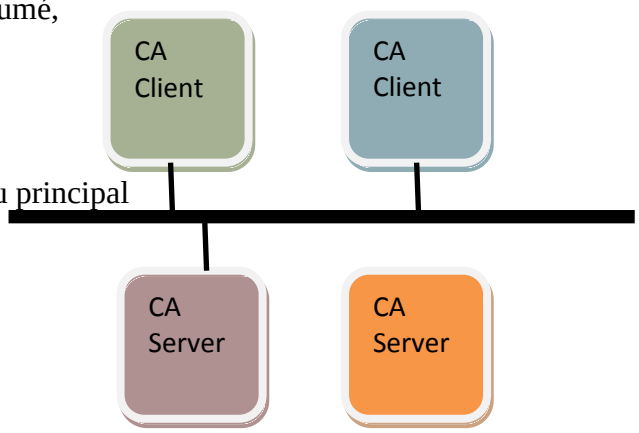


Figure 26 : l'outil ArchiveViewer

En résumé,



CA : Channel Access

Le serveur de l'IOC lance un service qui permet de diffuser des informations. Ces informations sont appelées Process Variables(***PV***), et sont associées à des Records dans la base de données «EPICS». Elles peuvent donner l'état d'un appareil, éffectuer une mesure, etc…

Lorsque qu'un client cherche une information, il se connecte sur le réseau principal, et interroge tous les Chanel Acess Serveur (CA Server), soit les serveurs des IOC, afin de trouver lequel fournit l'information dont il a besoin. Il récupère la valeur par l'intermédiaire du Channel Access Client et vient l'afficher sur une interface graphique développée par l'outil EDM par exemple, ou développé en Java.

Remarque : N'importe quel client faisant du Channel Access peut lire ces variables. (Java, Labview, etc…)

3- Présentation de la plateforme logicielle

Sachant qu'il y a plusieurs équipes qui travaillent sur le Contrôle/Commande de SPIRAL2, et que le but final est de tout déployer à GANIL, il est préférable par souci de compatibilité et d'homogénéité, que chaque développeur ait la même version de Linux, et les mêmes logiciels de développement. C'est pourquoi il a été développé une plateforme logicielle. La plateforme logicielle a donc pour but de configurer de manière identique tous les systèmes informatiques pour le contrôle de l'accélérateur SPIRAL2. Tous les packages regroupant cette plateforme logicielle ont été centralisés sur un serveur du CNRS à Lyon, où tous les développeurs ont accès. Voici par la suite une brève présentation.

4 - Architecture réseau

6.1 A Saclay

Durant les premiers tests à Saclay il n'y aura pas de réseau. L'objectif de cette phase est de tester la source avec un minimum d'instruments, soit produire du faisceau et d'en visualiser les caractéristiques.

Le PC de Contrôle/Commande sera relié à la carte MVME5500 par l'intermédiaire d'un Switch, ainsi que tous les instruments pilotés par la connexion TCP/IP.

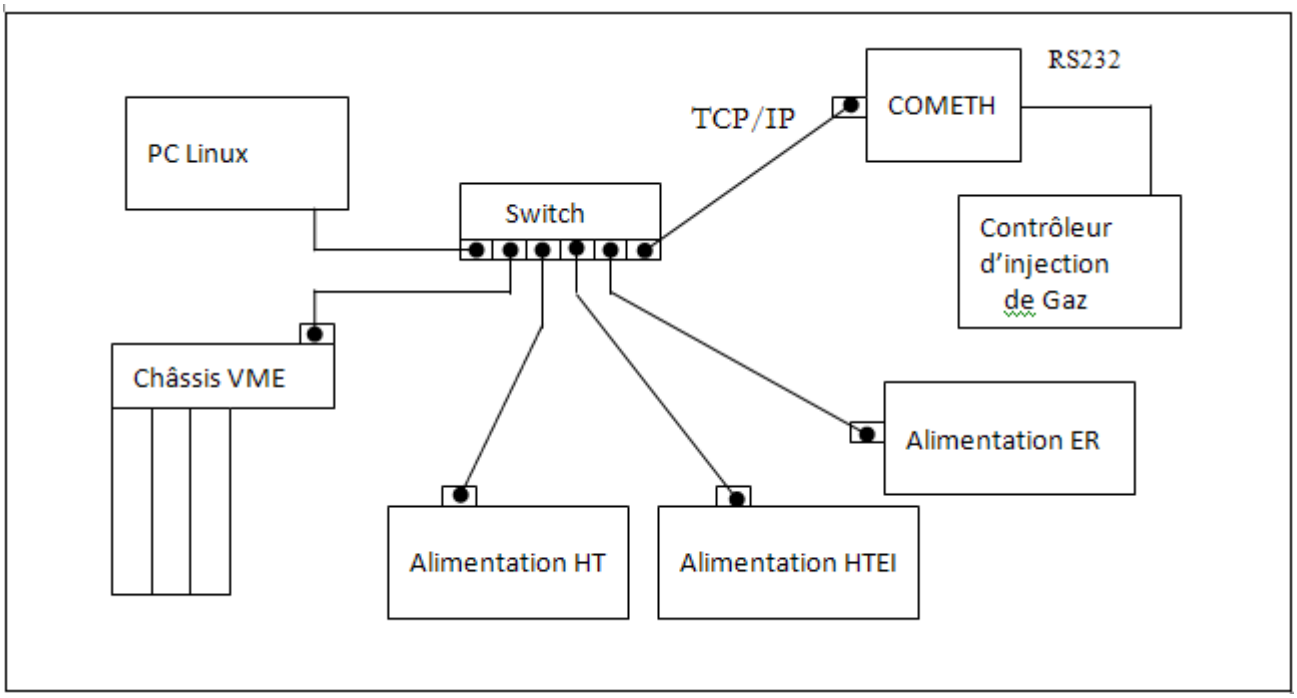


Figure 27 : Plan réseau de Saclay

VII – Le développement du contrôle/commande sous EPICS

1 - Déroulement du mémoire

2 - Quelques règles pour collaborer dans de bonnes conditions

2.1 Architecture de développement topSP2

2.2 Nommage des Records

2.3 Nommage des châssis VME

3 - Contrôle/commande de la source deuton + Cage de Faraday (CF11) sur la ligne LBE2

3.1 Le magnétron

3.2 Le générateur d'impulsions

3.2 L'adaptateur d'impédance

3.3 Le contrôleur d'injection de gaz

3.4 Les alimentations

3.5 Cage de Faraday CF11

3.6 Interface Homme/Machine générale de pilotage

4 - Contrôle/commande du jeux de fentes FV13 & FH13 sur la ligne LBE1

4.1 Développement EPICS

4.2 Interface Homme/Machine de pilotage

1- Déroulement du mémoire

Le début de mon stage a commencé en janvier 2008, lors de mon arrivée dans le laboratoire LD2I.

Dès le début de mon stage j'ai été chargé du contrôle commande de la source de Deutons, pour le projet SPIRAL2. J'ai donc réalisé ce travail dans le laboratoire LD2I à Saclay.

Le déroulement de mon travail s'est déroulé en plusieurs étapes :

Tout d'abord le premier objectif était de me familiariser avec le monde EPICS. J'ai réalisé un travail de recherche par le biais d'internet d'environ deux mois sur de la documentation faisant référence à EPICS. Par la suite, tout en continuant ce travail de recherche, j'ai commencé à effectuer quelques tests basiques afin de mettre en pratique ce que j'avais acquis auparavant. Ce travail d'autoformation m'a demandé environ deux mois.

Pendant cette première phase d'autoformation j'ai reçu deux instruments pour ma mission initiale correspondant au pilotage de la source de Deutons de SPIRAL2. Les instruments en question étaient : le magnétron et le contrôleur d'injection de gaz. J'ai donc, à partir d'Avril 2008, commencé à travailler réellement sur ma mission initiale jusqu'en septembre 2008. Ensuite par souci d'approvisionnement du reste des instruments pour le contrôle/commande de la source de Deutons, j'ai été chargé de travailler sur une autre partie du projet SPIRAL2. Un objectif supplémentaire a été ajouté à ma mission initiale.

Cet objectif supplémentaire m'a permis d'enrichir mes connaissances sur EPICS, et d'approfondir mon implication dans le projet SPIRAL2. J'ai travaillé sur un projet qui m'était indirectement très utile par la suite. Ce projet concernait le pilotage d'une fente horizontale et d'une fente verticale qui avait pour but d'être installé à Grenoble pour tester une partie de la ligne LBE1, représentant la deuxième source d'ions possible pour l'injecteur de SPIRAL2. Sur l'ensemble de la partie accélérateur, il y a environ une dizaine de fentes à piloter. J'ai donc été chargé du contrôle/commande des fentes pour l'ensemble de la machine SPIRAL2. En travaillant sur ce développement pendant environ dix mois, j'ai rencontré des soucis sur le driver Modbus/TCP que j'allais utiliser par la suite dans mon projet pour la source de Deutons. Avec l'aide d'un collègue, nous avons modifié le driver Modbus. Ce fut ma première participation pour la communauté EPICS. Le développement du contrôle/commande des fentes FH13 et FV13 sur la ligne LBE1 a été mené jusqu'à l'installation sur site, et a fonctionné avec succès sur le site de Grenoble. Ce développement me resservira pour toutes les fentes de la machine SPIRAL2.

Ensuite à partir du mois de février 2009, j'ai repris mon activité sur le pilotage de la source jusqu'en septembre 2009. J'ai reçu le reste des instruments correspondant aux trois alimentations haute tension, au générateur d'impulsions, et à l'adaptateur d'impédance. J'ai réalisé durant cette période des programmes, indépendants les uns des autres, ayant pour but de valider le pilotage de chaque instrument concernant le pilotage de la source.

Ensuite à partir de septembre 2009, ayant validé la partie communication avec chaque instrument, je suis passé à l'étape d'intégration, qui a eu pour but de regrouper chaque programme en un programme unique, avec une interface utilisateur la plus complète possible. Cette phase m'a pris environ deux mois. Enfin j'ai fini par réaliser mes premiers tests en réel sur un banc de test équipé d'une source pilotable de la même manière que la source de Deutons. J'ai pu tester le générateur d'impulsions, le magnétron, l'adaptateur d'impédance, et l'interface utilisateur. Malheureusement les tests de la ligne LBE2 phase 1 ont pris du retard, ce qui implique qu'il n'y aura pas de compte rendu sur ces tests dans ce rapport.

Pour résumé mon travail, j'ai réalisé le contrôle/commande de la source de Deutons situé sur la ligne LBE2, et le contrôle/commande d'un jeu de fentes situé sur la ligne LBE1 ou se trouve la seconde source d'ions. Ce second développement sera commun à l'ensemble des fentes sur l'ensemble de la machine SPIRAL2.

2- Quelques règles pour collaborer dans de bonnes conditions

2.1 Architecture de développement topSP2

Tous les développements qui suivent ont une arborescence topSP2.
Qu'est qu'une arborescence topSP2 ?
C'est un modèle d'organisation des dossiers, qui permet de séparer les modules des IOCs. Je rappelle qu'un module est une application associée à un équipement ou à une fonction qui peut être instanciée dans un IOC.
Dans l'exemple ci-contre,
« ***alimCourantHazenmeyerApp, CFaradApp, fentesApp, …*** »
sont des modules applicatifs, et
« **iocFarad , iocFentes, …** »
sont des IOCs.

2.2 Nommage des Records

La codification, ou le nommage des Records Epics est très importante car le nom d'un Record sur n'importe quelle expérience, type accélérateur par exemple, doit être unique sur tout l'ensemble du contrôle/commande de la machine.

La codification suivante a été déterminée par le GANIL :

Fonction *(17 caractères maximum)*		**Signal** *(10 caractères maximum)*
DDDDD-RRRRRRR[-CCCCCCC]	:	SssssSssss

Avec la signification suivante :

Champ	Signification	Contraintes
DDDDD	**D**omaine où se situe logiquement la fonction	2 à 5 caractères
RRRRRRR ou RRRRnnn	**R**epérage individuel de la fonction concernée	2 à 7 caractères, incluant éventuellement un index sur 1 à 3 chiffres
CCCCCCC	Optionnellement un sous-**C**omposant de la fonction	1à 7 caractères

La combinaison des différents champs doit respecter la taille globale de la chaîne de caractères (17 maximum). Le séparateur est le tiret et tous les caractères sont alphanumériques et en majuscules. Le premier caractère de chaque champ est impérativement alphabétique.

Exemple :

La consigne de la tension de l'alimentation Haute Tension pour piloter la source de Deutons sur Ligne basse énergie sera nommée ainsi :

DDDDD : Ligne basse énergie deutons => ***LBE2***

RRRRRR : source de Deutons => ***SD***

CCCCCC : Alimentation Haute Tension => ***HT***

SssssSsss : Consigne de tension => ***VCons***

En résumé, le Record pour modifier la tension de l'alimentation HT sur la source deuton, sera nommé comme ceci : ***LBE2-SD1-HT :VCons***

2.3 Nommage des châssis VME

Comme le nom des Records, les châssis VME doivent avoir des noms bien distincts les uns des autres. Il a été décidé :

- Pour la ligne LBE1 => ***VMELB1***
- Pour la ligne LBE2 => ***VMELB2***
- Pour la ligne LBEC => ***VMELBEC***

3- Contrôle/commande de la source deuton + Cage de Faraday (CF11) sur la ligne LBE2

Le script de démarrage de l'IOC pour le châssis ***VMELB2*** regroupe toutes les applications concernant le contrôle/commande de la source deuton ainsi que la cage de Faraday CF11. Il s'appelle « ***st.cmd*** ». Par la suite voici une présentation du pilotage des différents instruments du point de vue EPICS.

3.1 Le magnétron

Pour ce développement une première base de données «EPICS» a été utilisée. Le nommage des Records concernant le magnétron est : ***LBE2-SD1-MAGN***

3.1.1 base de données «EPICS»

La base de données «EPICS» est composée de cinq Records :

- Deux Records de type « *analog input (ai)* », pour effectuer les mesures analogiques de la puissance incidente et la puissance réfléchie rafraichies toutes les 500 millisecondes (carte ICV150):
  - ⇨ ***LBE2-SD1-MAGN :PIncMes***
  - ⇨ ***LBE2-SD1-MAGN :PRefMes***

- Quatre Records de type « *binary input (bi)* », pour les entrées digitales permettant de refléter l'état de la source rafraichies toutes les 500ms (carte ICV196) :
  - ⇨ ***LBE2-SD1-MAGN :EtatLim***
  - ⇨ ***LBE2-SD1-MAGN :EtatPret***
  - ⇨ ***LBE2-SD1-MAGN :EtatOn***
  - ⇨ ***LBE2-SD1-MAGN :DefautAct***

- Un Record de type « *binay output (bo)* » permettant d'envoyer la commande démarrant le magnétron (carte ICV196) :

  ⇨ ***LBE2-SD1-MAGN :On***

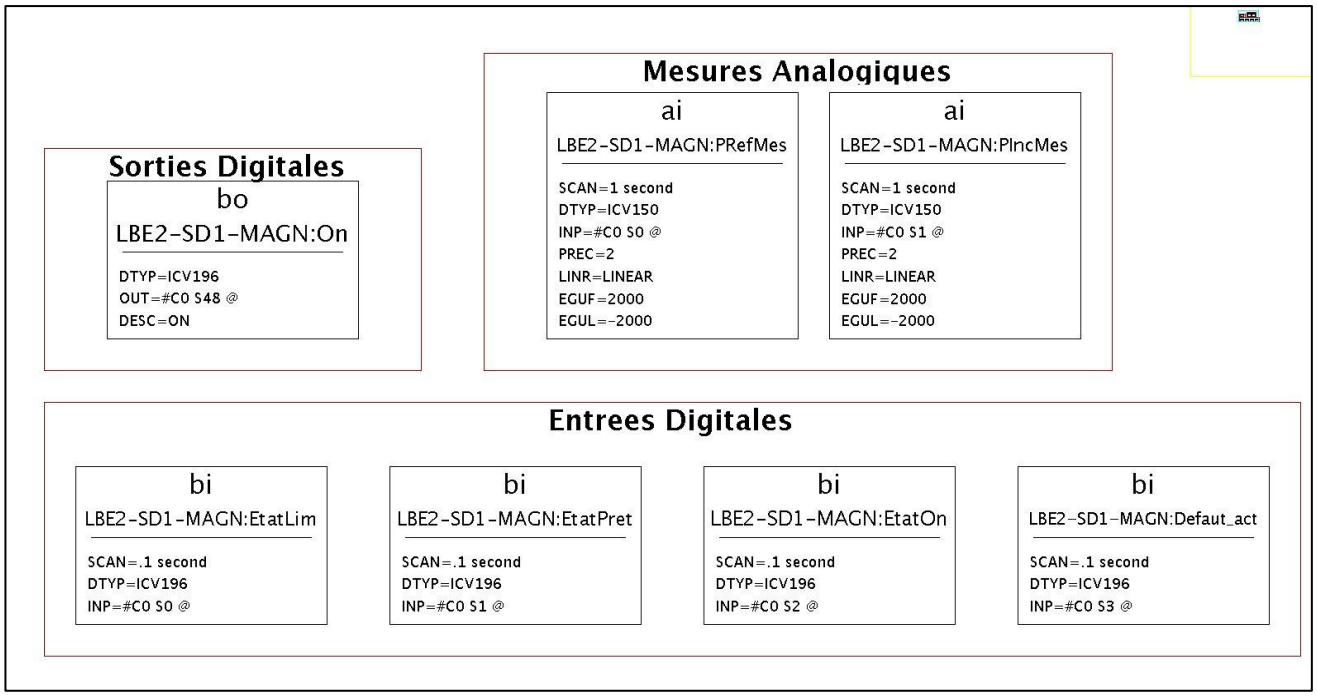


Figure 28 : magnétron – base de données «EPICS» (VDCT)

3.1.2 Extrait du script de démarrage général st.cmd

Pour cette application il suffit uniquement de charger la base de données «EPICS».

```
################ CHARGEMENT BASE DE DONNEES ################
## Chargement de la base de données
dbLoadRecords("Magnetron.db")
## Initialisation de la carte ICV150, en mode 16 bits
icv150CfgAdc(0,16)
.....
```

3.2 Le générateur d'impulsions

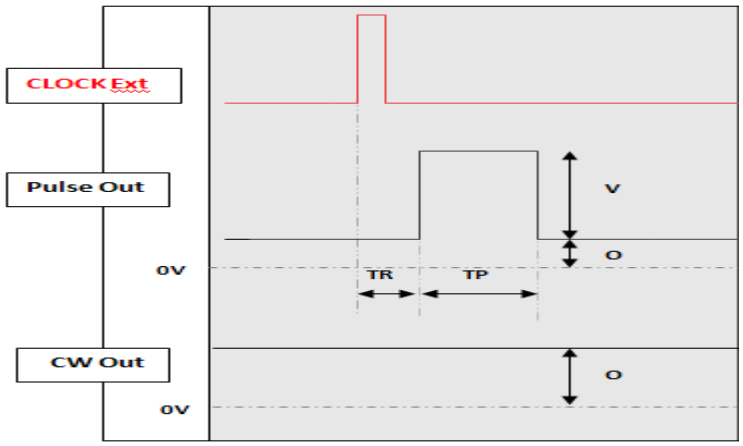
On rappelle que le magnétron a deux modes de fonctionnement : Continu et Pulsé.

Le générateur, synchronisé sur une horloge externe, va permettre de faire fonctionner le magnétron dans les deux modes.

Les informations importantes pour le mode :

Pulsé : - Largeur du pulse (TP)
- Délai de retard (TR)
- Synchronisation externe (Clk ext)
- Offset (O)
- Amplitude(V)

Continu (CW) : - Offset (O)



Pour passer d'un mode à l'autre il suffit de configurer correctement l'appareil. Pour accélérer la procédure, deux types de signaux ont été préenregistrés. Un premier associé au mode continu, et un second au mode pulsé. Il suffit d'appeler la bonne fonction prédéfinie, et ensuite d'appliquer les différents paramètres telle que l'amplitude, l'offset, la période, etc...

- Pour le mode continu, un signal pulsé avec une amplitude de 1mv a été préenregistré.

- Pout le mode pulsé, un signal carré synchronisé via une horloge externe avec un retard programmable a été préenregistré.

1.1.1 Drivers EPICS

Pour cette application il a été utilisé le Device Support générique « ***STREAM Device*** » qui permet de s'interfacer avec de nombreux équipements. Cette communication est fondée sur l'envoi et la réception de chaines de caractères via les protocoles RS232, GPIB (IEEE-488), et TCP/IP.

De plus des programmes « ***SNL*** » ont été utilisés permettant de réaliser des tâches séquentielles, par exemple lors du changement de mode, où les commandes doivent être envoyées l'une après l'autre.

1.1.2 Fichier protocole

Toutes les commandes envoyées à l'instrument sont indiquées dans un fichier texte appelé « ***protocole*** ». Celles-ci sont regroupées sous forme de fonctions appelées depuis la base de données «EPICS».

Exemple :

L'appareil YY à piloter comprend les commandes suivantes :

- ***« Tension XX » : Appliquer une tension de valeur XX***
- ***« ? Tension » : Quelle est la tension ?***

Dans un fichier protocole, « out » permet d'écrire une commande et « in » permet de lire une réponse

Voici deux fonctions qui résument le fonctionnement d'un fichier protocole :

```
FonctionEcritureTension
    {
    out « TENSION XX » ;   // correspond à l'application d'une tension de valeur XX
    }
FonctionLectureTension
    {
    out « ? TENSION» ;   // Quelle est la tension ?
    in « %f » ;              // %f indique que l'on attend en réponse une valeur de type float
    }
```

Ces fonctions sont appelées depuis la base de données «EPICS» par le paramètre « out » d'un Record. Dans ce paramètre est indiqué le fichier protocole qui regroupe toutes les commandes utiles, le nom de la fonction appelée, la connexion utilisée, et pout finir la valeur à envoyer correspondant à la valeur du Record.

⇨ ***OUT = @GenePulse_proto.db FonctionEcritureTension TCP2 Valeur***

1.1.3 Base de données «EPICS»

Le nommage des Records concernant le générateur d'impulsions est : ***LBE2-SD1-GenePulse***.

La base de données «EPICS» est assez conséquente c'est pourquoi il ne sera présenté que quelques fonctions :

Fonction1 : Appliquer une tension avec ou sans offset en mode pulsé

Cette fonction est constituée de cinq Records :

- Un Record soft de type « *analog output (ao)* » permettant de saisir la consigne d'amplitude via le panneau de controle principal:
  ⇨ ***LBE2-SD1-GenePulse :PCons*** *(Saisie consigne Amplitude)*

- Un Record de type « *calcout* », permettant d'effectuer des calculs et associé directement à la liaison hardware TCP. Dès que ce Record est actif, la commande est émise vers l'appareil après le calcul.
  ⇨ ***LBE2-SD2-GenePulse : PConsSet*** *(Commande String Validation Amplitude)*

- ***OUT = @GenePulse_proto.db SetAmplitude TCP2 Valeur***
- ***La fonction SetAmplitude dans le fichier GenePulse_proto.db est :***

```
SetAmplitude
    {
    out « VOLT %f » ;
    }
```

Ce Record s'active dès la saisie d'une consigne d'amplitude sur le panneau de controle principal dans le Record précédent, ***LBE2-SD2-GenePulse : PCons***.
Il effectue un calcul avec la consigne de tension qu'il reçoit, et l'envoie directement à l'instrument. Concernant le calcul c'est une division par deux de la consigne d'amplitude. Une amplitude de 5V pour un signal carré correspond à une amplitude de 2,5V en positif et la même amplitude en négatif. Pour avoir un Pulse compris entre 0 et 5V, il faut donc ajouter un offset correspondant à l'amplitude négative, soit 2,5V dans ce cas. C'est l'objectif du Record qui suit. Le calcul s'effectue dans la base de données «EPICS».

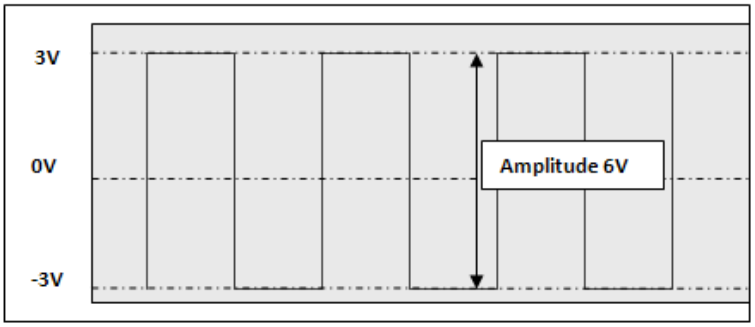
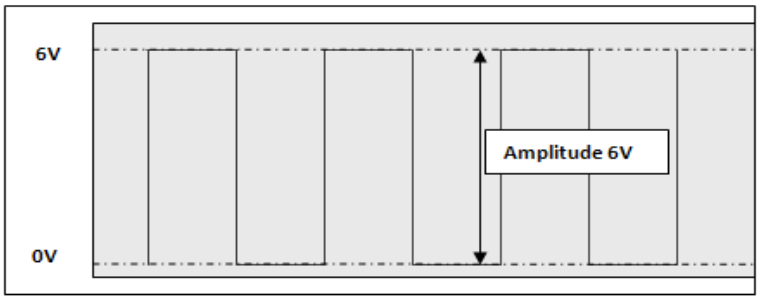
- Un Record de type « *calcout* » permettant d'ajouter l'offset pour la construction du pulse associé directement à la connexion TCP. Ce Record s'active quand le Record précédent a été exécuté.

  ⇨ ***LBE2-SD1-GenePulse :OffsetPulse*** *(Commande String offset)*

- ***OUT = @GenePulse_proto.db Offset TCP2 Valeur***
- ***La fonction Offset dans le fichier GenePulse_proto.db est :***

```
SetOffset
    {
    out « VOLT:OFFS  %f » ;
    }
```

En résumé,

Pour obtenir un pulse avec une tension comprise entre 0-6V, il faut mettre 6 dans le Record « LBE2-SD1-GenePulse :PCons », qui viendra activer le Record LBE2-SD2-GenePulse : PConsSet.	
Après cela, il exécutera le Record LBE2-SD1-GenePulse :OffsetPulse .Celui-ci enverra 3V en offset à l'instrument pour avoir au final un pulse avec une amplitude comprise entre 0 et 6V.	

- Un Record soft de type « *analog output (ao)* » permettant de saisir la consigne d'offset via l'interface :

  ⇨ ***LBE2-SD1-GenePulse :Offset (****Saisie consigne offset)*

- Un Record de type « *calcout* » permettant d'ajouter un offset au Pulse. Il faut prendre en compte l'offset par défaut du pulse pour sa construction, et ajouter à celui-ci l'offset désiré.

  Si nous ajoutons un offset de 1V au Pulse, nous ajoutons en offset total de :

  3V pour construire le Pulse entre 0 et 6 V + 1V d'offset

  Soit 4V d'offset dans le Record LB2-SD1-GenePulse :Offset2

  ⇨ ***LB2-SD1-GenePulse :Offset2*** *(Commande String offset)*

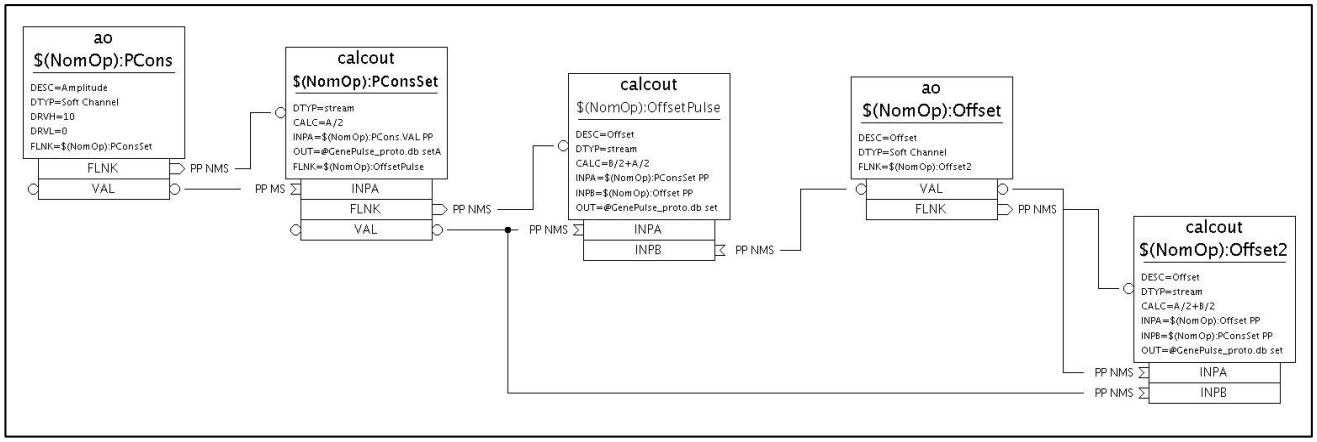


Figure 29 : Generateur - Appliquer une tension avec ou sans offset en mode pulsé (VDCT)

Fonction2 : Changement de mode

Cette fonction utilise deux Records :

- Un Record de type « *binary output (bo)* » permettant de changer de mode via un bouton sur le panneau de control principal. La valeur que prend ce Record par défaut est soit 1 soit 0. Pour modifier ses valeurs, il faut modifier les champs « ZNAM » et « ONAM ».

  Ceci permet d'envoyer la nouvelle valeur inscrite dans « ZNAM », ou « ONAM » aux Records suivants à chaque action de ce bouton. Cela dépendra du couple de valeurs d'origine, 0 ou 1. Si cette valeur vaut 0, alors c'est ZNAM qui est envoyé, ou sinon c'est ONAM. Dans notre cas ZNAM vaut « 1 », et ONAM vaut « 2 ».

  ⇨ ***LBE2-SD1-GenePulse :ModeSet*** *(Bouton changement de mode)*

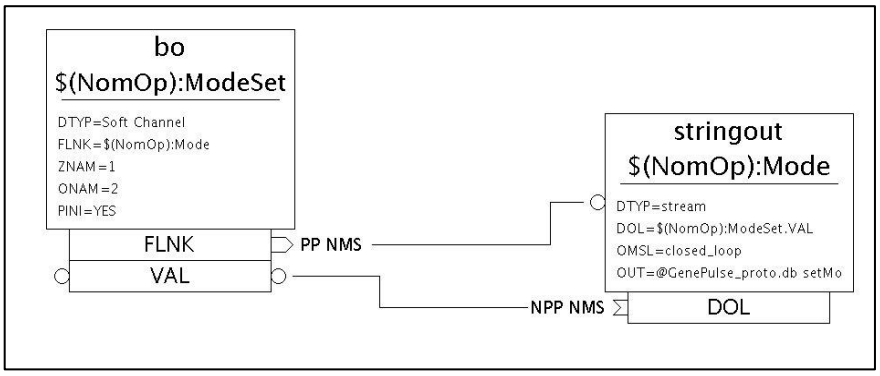


Figure 30 : Générateur - Changement de mode (VDCT)

- Un Record de type « *Stringout* » permettant d'envoyer la commande pour changer de mode. Ce Record s'active dès que le Record précédent a été exécuté, et prend la valeur 1 ou 2.

  ⇨ ***LBE2-SD1-GenePulse :Mode*** *(Commande String Changement de mode)*

  - ***OUT = @GenePulse_proto.db SetMode TCP2 Valeur***
  - ***La fonction SetMode dans le fichier GenePulse_proto.db est :***

```
SetMode
    {
    out « RCL %s » ;
    }
```

1.1.4 Programme « SNL»

Pour cette application il y a deux programmes « ***SNL***» qui interagissent avec de la base de données «EPICS» :

- *sncGenerateur.stt* : Ce programme permet lors du changement de mode, d'appliquer les différents paramètres tels que la tension, l'offset, etc… aux Records de la base de données «EPICS». Effectivement il a été demandé lors du passage du mode continu vers le mode pulsé, et l'inverse de garder la même amplitude.
  Cependant l'amplitude pour les deux modes n'est pas associée au même Record. Le rôle de ce programme est de faire communiquer les deux Records entre eux. Le rafraichissement est effectué à une vitesse de 500 ms.

- *sncGenerateurGraph.stt* : Ce programme permet de tracer un graphique représentant la forme du pulse avec les différents paramètres saisis. le Pulse est tracé avec un tableau de 100 000 points, en X et Y. Sachant que le pulse a une fréquence de répétition max de 1 000 ms, et une largeur min de 100 µs, cela fait une précision suffisante de :
  prec = 1000 ms / 100000 pts = 0,01 ms/pts , soit 10 µs /pts
  Le rafraichissement du dessin est effectué à une fréquence périodique de 1s. D'autre part l'échelle des abscisses s'adapte à la fréquence de répétition.

1.1.5 Extrait du script de démarrage général st.cmd

Pour cette application il faut créer une connexion TCP avec le générateur, charger la base de données «EPICS», et lancer le séquenceur avec les deux programmes « ***SNL***».

```
## Initialisation de la connexion TCP avec le générateur
drvAsynIPPortConfigure("LBE2-SD1-GenePulse","192.168.1.6:5025",0,0,0);
################ CHARGEMENT BASE DE DONNEES ###############
……
## Chargement de la base de données du générateur
dbLoadRecords("GenePulse.db")
################ LANCEMENT DES TACHES SNL ##################
## Lancement de la tache SNL
seq &sncGenerateur,"NomOp=LBE2-SD1-GenePulse"
seq &sncGenerateurGraph,"NomOp=LBE2-SD1-GenePulse"
```

1.2 L'adaptateur d'impédance

Le nommage des Records concernant l'adaptateur d'impédance est : ***LBE2-SD1-ADAPT***

1.2.1 Drivers EPICS

Pour cette application il a été utilisé le Device Support générique « ***ADAS*** » qui permet d'utiliser les cartes d'Entrées/Sorties VME ICV150, ICV196, et ICV714.

1.2.2 Base de données «EPICS»

La base de données «EPICS» est composée de cinq Records :

Pour la Voie X :

- Un Record de type « *analog input (ai)* » permettant d'effectuer la mesure du coefficient d'adaptation sur la Voie X :
  ⇨ ***LBE2-SD1-ADAPT :XMes***

- Un Record de type « *analog ouput (ao)* » permettant d'appliquer la consigne de déplacement du moteur sur la Voie X :
  ⇨ ***LBE2-SD1-ADAPT :XCons***

Pour la Voie Y :

- Un Record de type « *analog input (ai)* » permettant d'effectuer la mesure du coefficient d'adaptation sur la Voie Y
  ⇨ ***LBE2-SD1-ADAPT :YMes***

- Un Record de type « *analog ouput (ao)* » permettant d'appliquer la consigne de déplacement du moteur sur la Voie Y :
  ⇨ ***LBE2-SD1-ADAPT :YCons***

Pour le mode :

- Un Record de type « *binary output (bo)* » qui permute le mode de fonctionnement en automatique ou en manuel :
  ⇨ ***LBE2-SD1-ADAPT :Mode***

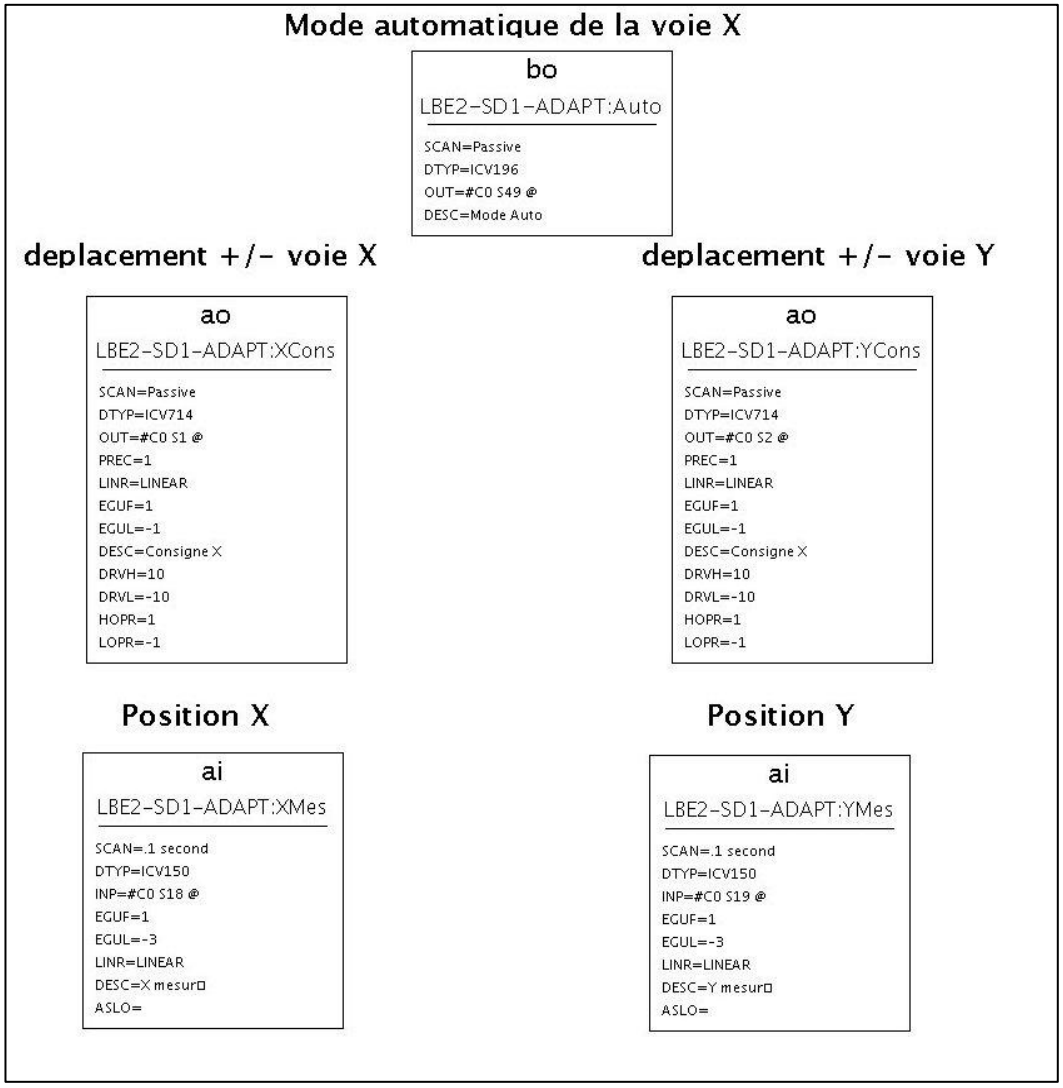


Figure 31 : Adaptateur impédance – Base de données «EPICS» (VDCT)

1.2.3 Extrait du script de démarrage général st.cmd

Pour cette application il faut charger la base de données «EPICS»,

```
.....
################ CHARGEMENT BASE DE DONNEES # ################
## Chargement de la base de données
dbLoadRecords("Adapteur.db")
.....
```

1.3 Le contrôleur d'injection de gaz

Rappel : Cet instrument fonctionne sur la plateforme HT, et communique via le protocole RS232.

Sachant que le contrôle/commande est à la masse, et non à la haute tension, il faut isoler électriquement l'installation du contrôleur d'injection de gaz. Il faut prendre en compte que la distance qui sépare le châssis VME et le contrôleur est importante, ce qui peut impliquer des chutes de tension dans les câbles, et donc des problèmes de communications.

Pour cela, il a été décidé d'utiliser un port série déporté sur le réseau (boitier COMETH) qui permet de travailler au plus près de l'instrument. Cependant l'isolement entre les instruments à la haute tension et à la masse reste inexistant. Il a donc été décidé par la suite d'ajouter une fibre optique compatible RS232 entre le boitier COMETH et l'instrument afin d'isoler électriquement le contrôle/commande.

En résumé :

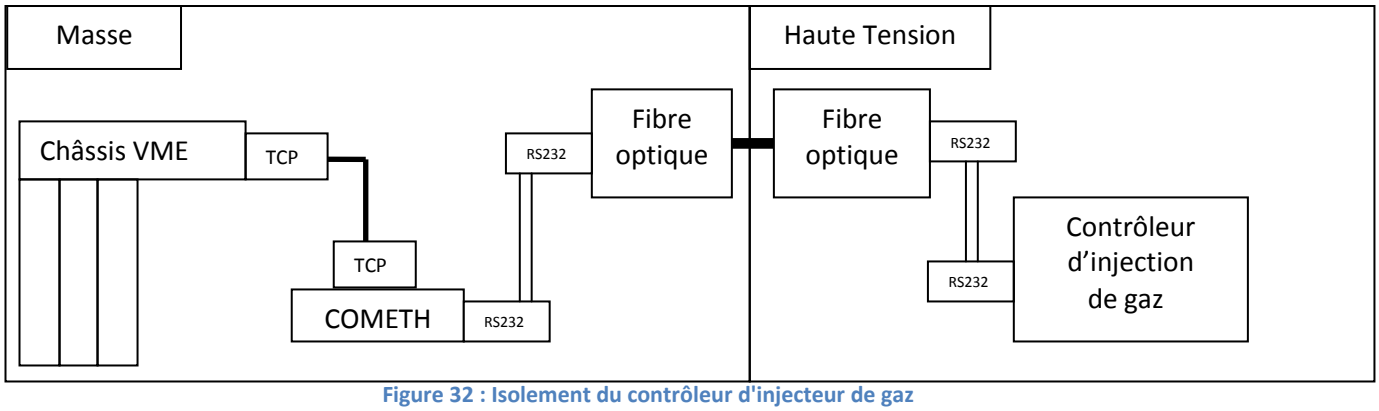


Figure 32 : Isolement du contrôleur d'injecteur de gaz

1.3.1 ***Drivers EPICS***

Pour cette application il a été utilisé le Device Support générique « ***STREAM Device*** » qui permet d'envoyer des chaines de caractères de type « String » via la liaison TCP ou RS232. De plus des programmes « ***SNL*** » ont été utilisés.

1.3.2 ***Base de données «EPICS»***

La base de données «EPICS» est assez conséquente c'est pourquoi il ne sera présenté que les principales fonctions :

Fonction1 : L'ouverture/fermeture vanne

La base de données «EPICS» est composée de :

- Un Record de type « *binay output (bo)* » permettant d'ouvrir la vanne :
  ⇨ ***LBE2-SD1-DEB1:Vanne1*** (*Commande String Ouverture vanne1)*

- ***OUT = @PR4000B_proto.db OuvertureVanne1 TCP2 Valeur***
- ***La fonction OuvertureVanne1 dans le fichier PR4000B_proto.d est :***

```
OuvertureVanne1
    {
    out « VL1,%s » ;
    }
```

- Un Record permettant de valider l'ouverture de la vanne :

⇨ ***LBE2-SD1-DEB1:SetOn1*** (*Commande String Validation ouverture)*

- ***OUT = @PR4000B_proto.db ValidOuverture TCP2 Valeur***
- ***La fonction OuvertureVanne1 dans le fichier PR4000B_proto.d est :***

```
ValidOuverture
    {
    out « VL0, %s » ;
    }
```

Pour les deux Records précédents il y a besoin d'un paramètre variable qui correspond à l'état futur de la vanne, soit *ON* ou *OFF*. Cette information va être fournie par le Record qui suit.

- Un Record de type « *binay output (bo)* » associé à un bouton sur le panneau de contrôle principal. La valeur que prend ce Record est soit 1, soit 0. Pour modifier ces valeurs, les paramètres ZNAM et ONAM du Record ont été modifié, ZNAM avec « OFF », et ONAM vaut « ON ».

⇨ ***LBE2-SD1-DEB1:OpenVanne1*** (*Action ouverture vanne1*)

- Un Record de type « *binay output (bo)* » associé à une led sur le panneau de contrôle principal indiquant si la vanne est ouverte ou pas. Ce Record est rafraichi toutes les 500ms.

⇨ ***LBE2-SD1-DEB1:EtatVanne1*** (*Etat vanne*)

- ***OUT = @PR4000B_proto.db EtatVanne1 TCP2 Valeur***
- ***La fonction EtatVanne1 dans le fichier PR4000B_proto.d est :***

```
EtatVanne1
    {
    out « ?Etat » ;
    in « %s » ;
    }
```

En résumé pour envoyer la commande d'ouverture de la vanne 1,

*Il suffit d'appuyer sur le bouton situé sur le panneau de contrôle principal associé au Record « **LBE2-SD1-DEB1:OpenVanne1** ». Ce Record vaudra 1, ce qui implique que la valeur de « ONAM » sera envoyée au Record « **LBE2-SD1-DEB1:Vanne1** » qui sera lui-même exécuté.*

*Ce Record après avoir été exécuté activera le Record suivant qui est « **LBE2-SD1-DEB1:SetOn1** » avec toujours en paramètre la valeur de **LBE2-SD1-DEB1:OpenVanne1** », soit « ON ».*

Pour la fermeture, il suffit d'appuyer sur le même bouton, et le déroulement sera le même, à part le paramètre variable qui sera non plus « ON », mais « OFF ».

En parallèle de cette action, il y a un Record de type « Stringin », qui récupère l'état de la vanne toutes les 500ms.

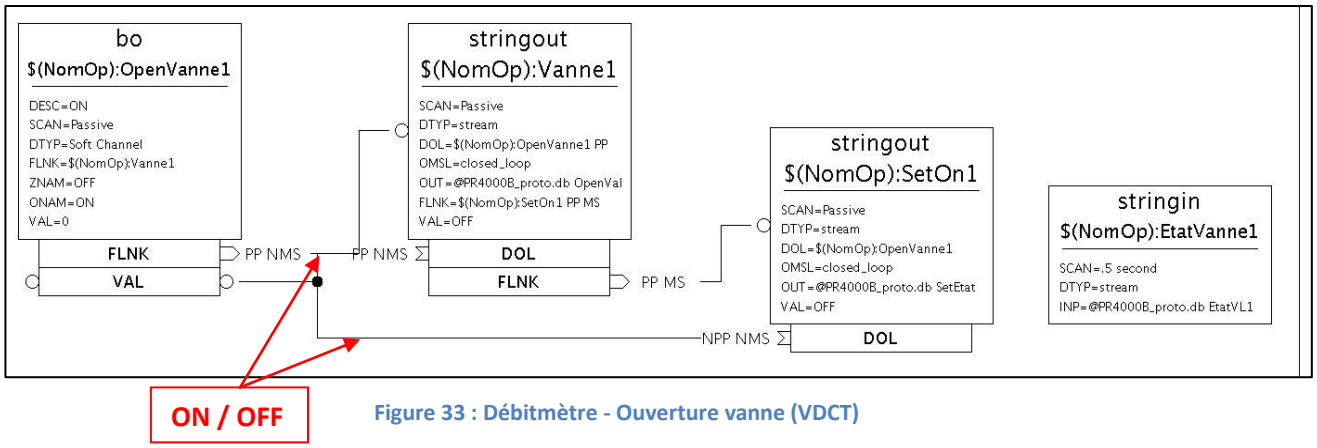


Figure 33 : Débitmètre - Ouverture vanne (VDCT)

Fonction2: La mesure du débit de gaz

Cette fonction est composée de :

- Un Record de type « *analog input (ai)* » permettant de lire la mesure du débitmètre. Ce Record est rafraichi toutes les 500ms. L'information reçue sera de type « float », d'où la syntaxe « ***%f*** » dans le fichier protocole ci-dessous.

  ⇨ ***LBE2-SD1-DEB1:Deb1Mes*** *(Commande String Mesure debit n°1)*

  - ***OUT = @PR4000B_proto.db MesDeb1 TCP2 Valeur***
  - ***La fonction MesDeb1 dans le fichier PR4000B_proto.d est :***

```
MesDeb1
    {
    out « ?DEB1:MES » ;
    in « %f » ;
    }
```

1.3.3 Programme « SNL»

Pour cette application il y a un programme « ***SNL***» qui va interagir avec la base de données «EPICS» :

- *sncPR4000B.stt* : Ce programme permet de réaliser un affichage avec des leds reflétant l'état de l'instrument. Effectivement pour l'état du mode distant/local, ou encore l'état de connectivité de l'appareil, l'appareil répond ON ou OFF, ce qui est difficile à implémenter avec des leds qui comprennent que le binaire 0 ou le 1. Ce fichier va donc tout simplement transformer le format String de la réponse de l'appareil en format short binaire 0 ou 1, associé à un Record « *binary ouput (bo)* » pour chaque information. Ce dernier sera utilisé sur le panneau de contrôle principal.

1.3.4 Extrait du script de démarrage général st.cmd

Pour cette application il faut initialiser une connexion TCP avec l'instrument, charger la base de données «EPICS», et lancer le séquenceur avec le fichier « ***SNL***» utile.

```
.....
## Initialisation de la connexion TCP avec le boitier COMETH servant d'intermédiaire ##
entre le châssis VME et le contrôleur d'injection de gaz
drvAsynIPPortConfigure("LBE2-SD1-DEB","192.168.1.2:5025",0,1,0)
############ CHARGEMENT DES BASES DE DONNEES ################
## Chargement de la base de données
dbLoadRecords("PR4000B.db")
################ LANCEMENT DES TACHES SNL ###################
## Lancement de la tache SNL
seq &sncPR4000B,"NomOp=LBE2-SD1-DEB"
```

1.4 Les alimentations

Les trois alimentations sont pilotées chacune par un boitier générique développé par le groupe électronique du Ganil. Pour ce développement un seul programme a été développé qui est compatible pour les trois alimentations. Un fichier de « ***substitutions*** » a été utilisé pour ajouter des spécificités au programme de chaque alimentation.

1.4.1 Drivers EPICS

Pour cette application il a été utilisé le Device Support générique « ***Modbus/TCP*** » qui a été modifié pour qu'il fonctionne selon les normes Modbus, soit en « multi-esclaves», et non en « mono-esclave » (***cf Annexe1***). Il a été utilisé aussi le driver « *Gensub* » qui permet d'inclure des procédures en langage C dans la base de données «EPICS».

1.4.2 Base de données «EPICS»

La base de données «EPICS» pour cette application est composée d'un ensemble de Records. Pour simplifier l'explication de la base de données «EPICS», elle sera divisée en quatre actions. Chaque action est composée de un ou plusieurs Records :

- *Action1* : Lecture des registres du boitier d'alimentation
- *Action 2* : Distribution des valeurs lues aux Records pour l'affichage
- *Action 3* : Ecriture des consignes vers le boitier d'alimentation
- *Action 4* : Initialisation des consignes lors du passage local au mode distant

Action1: Lecture des registres

Pour cette première action un Record de type « *waveform* » correspondant à un tableau de mots de 16 bits qui est rafraichi toutes les 100ms. Celui-ci permet d'enregistrer dans un tableau les valeurs des paramètres à partir de l'adresse 10 correspondant à la lecture de la consigne principale et de la consigne secondaire, à la lecture du mot d'état et du mot correspondant aux défauts.

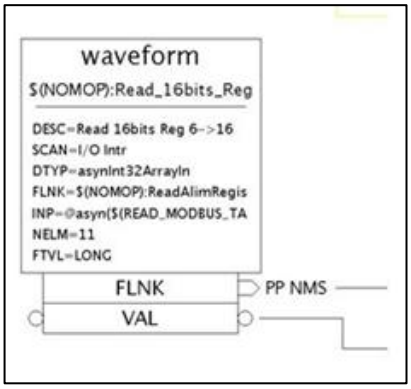


Figure 34 : Alimentation - Lecture des registres (VDCT)

Action2 : Distribution des valeurs aux Records via un « gensub »

Après avoir récupérer un tableau de valeurs correspondant aux registres de l'alimentation, il faut être capable de redonner l'information au bon Record afin qu'il puisse l'afficher sur le panneau de contrôle principal. C'est le rôle du « *gensub* ».

Il est généralement utilisé pour alléger la complexité de la base de données «EPICS». Ce Record reçoit donc en entrée le tableau de mots de 16 bits, correspondant à la lecture des registres de l'alimentation. A chaque nouvelle lecture, ce tableau est rafraichi. A partir de là il est simple de savoir par exemple qu'a l'emplacement 0 et 1 du tableau, il se trouve la valeur de la consigne principale sur 32bits. Via une procédure écrite en C, il suffit de récupérer et convertir cette information à la bonne échelle, et ensuite de l'envoyer au Record permettant de l'afficher sur le panneau de contrôle principal. Cette opération est réalisée pour chaque information.

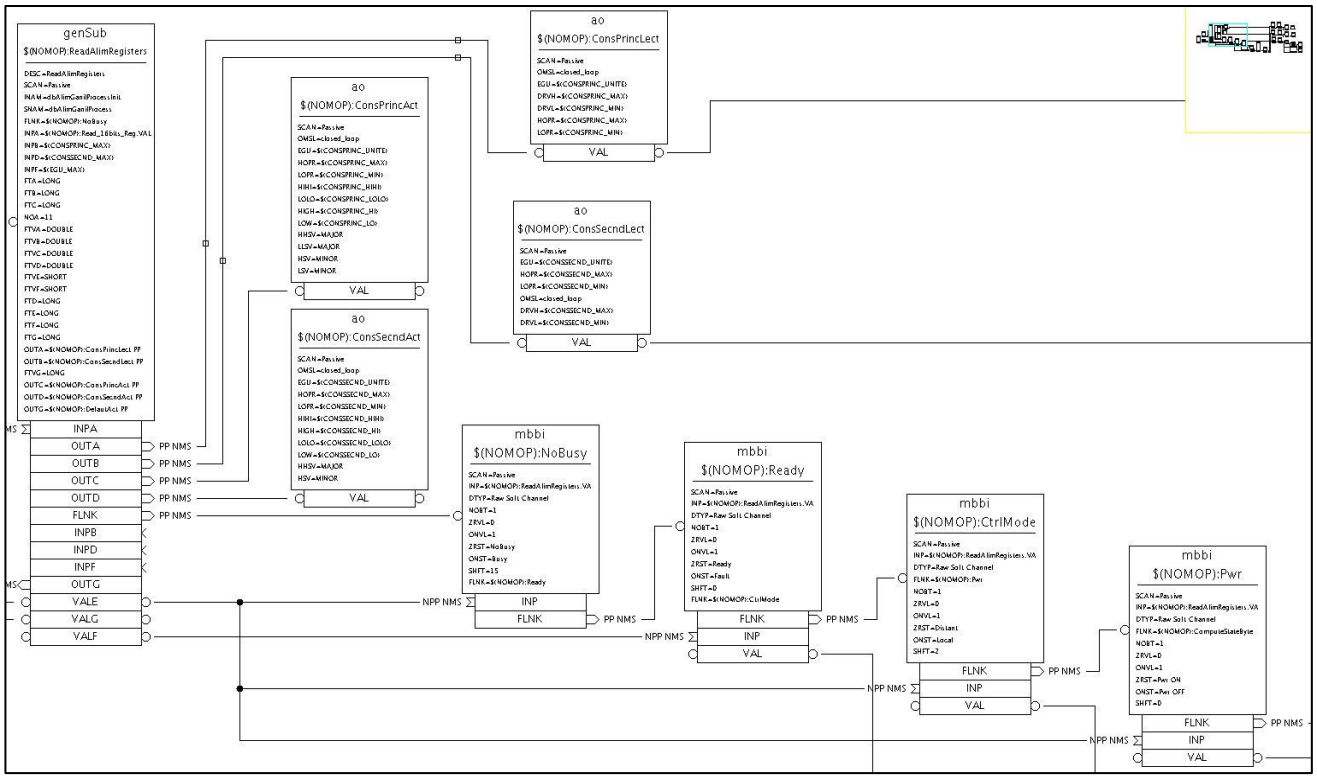


Figure 35 : Alimentation - Distribution des valeurs aux Records via un « gensub » (VDCT)

Action3: Ecriture des consignes

Après la lecture des informations nécessaires pour le pilotage des alimentations, il faut pouvoir écrire des consignes de tension ou de courant. Ces deux consignes doivent être écrites au format 32 bits.

Sachant que le modbus fonctionne en mot de 16bits, il faut utiliser un tableau de 2 mots, soit un Record de type « *waveform* » non plus en lecture mais en écriture. La consigne de tension est à l'adresse 6 & 7, et celle du courant à l'adresse 8 & 9. Pour éviter d'écrire le courant à chaque fois que la tension est modifié, et vis et versa, un Record de type « *waveform* » a été associé pour chaque consigne.

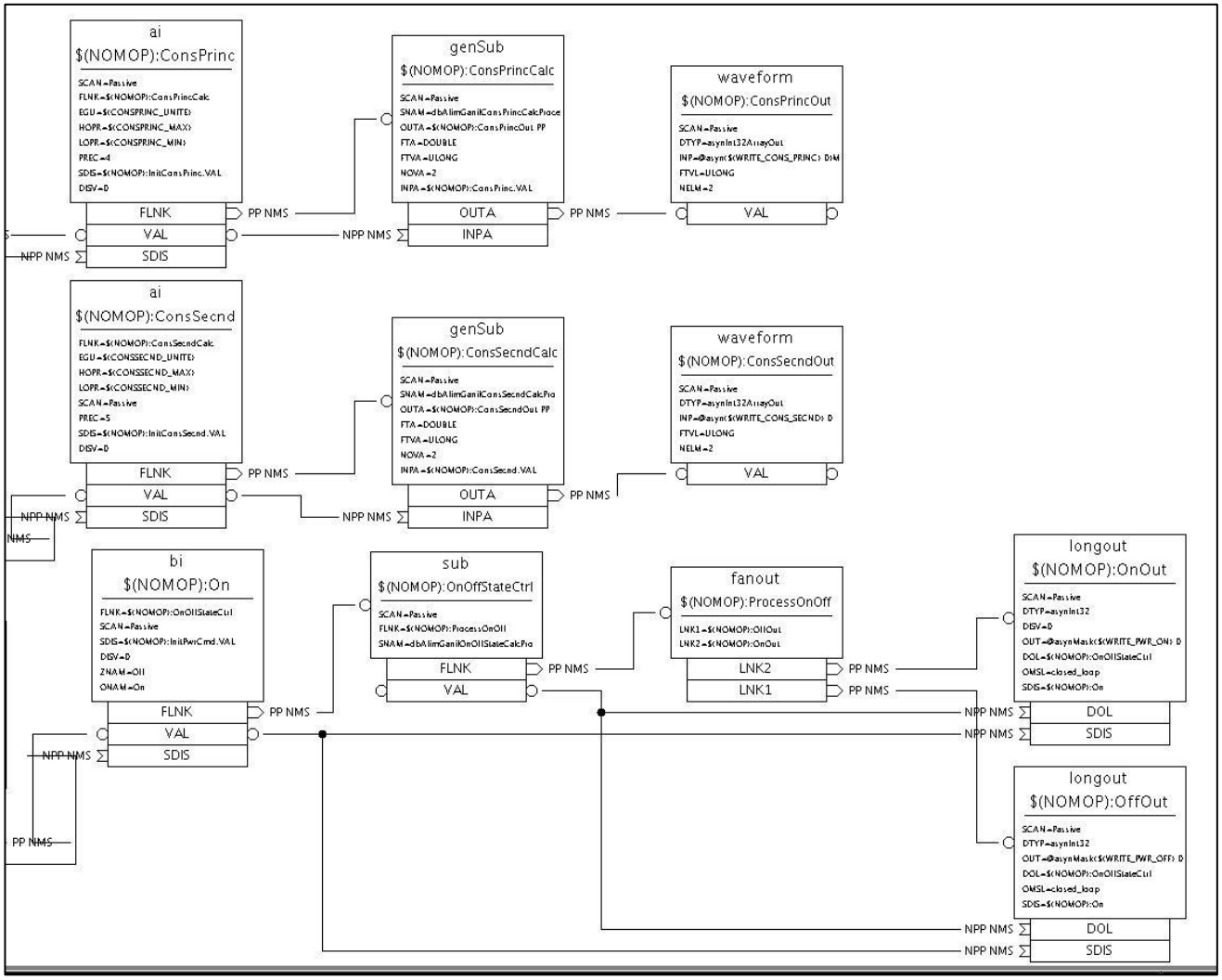


Figure 36 : Alimentation - Ecriture des consignes (VDCT)

***Action4*: Passage du mode local au mode distant**

Pour passer du mode local au mode distant il a été demandé une certaine procédure. Lorsqu'un opérateur prend la main en local, l'interface distante doit être automatiquement désactivée. Dès que l'opérateur en local rend la main, et donc repasse en mode distant, les valeurs des tensions et des courants modifiées en local doivent être répercuté sur le panneau de contrôle principal.

Le passage en mode distant sur ces alimentations ne peut se faire qu'en local. Pour vérifier qu'une alimentation est en mode local ou en mode distant il suffit de scruter en permanence le mot d'état. Une caractéristique du Record permet de le désactiver ce qui implique que lorsque l'alimentation est en mode local, tous les Records permettant l'écriture des consignes sont désactivés. Dès que l'alimentation repasse en mode distant, les Records inactifs redeviennent actifs avec les dernières valeurs consignées en local.

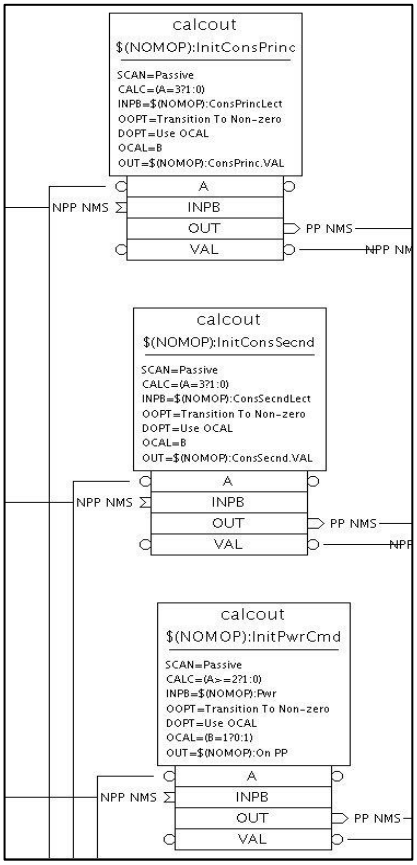


Figure 37 : Alimentation - Passage du mode local au mode distant (VDCT)

1.4.3 *Le fichier de substitution*

Ce fichier permet de dupliquer trois fois la base de données «EPICS» avec des caractéristiques différentes au niveau des Records pour chaque alimentation.
Concernant les caractéristiques les plus importantes :

- La partie fonction du nom pour chaque Record
- La gamme d'amplitude pour la consigne n°1
- La gamme d'amplitude pour la consigne n°2
- Les différentes fonctions « modbus » de lecture et écriture associé à chaque instrument :
    - Lecture de 6 mots de l'adresse 11 à 16 : Mesure consigne n°1, consigne n°2, Mot d'état, ...
    - Ecriture de 2 mots à l'adresse 6 & 7 pour la consigne n°1
    - Ecriture de 2 mots à l'adresse 8 & 9 pour la consigne n°2
    - Ecriture de 1 mot à l'adresse 0 pour activer l'alimentation
    - Ecriture de 1 mot à l'adresse 1 pour désactiver l'alimentation

Pour l'alimentation n°1 : la repousseuse d'électrons

- La partie fonction du nom pour chaque Record : ***LBE2-SD1-ER***
- La gamme d'amplitude pour la consigne n°1 : 0 ~ (-) 3kV
- La gamme d'amplitude pour la consigne n°2 : 0 ~ 10 mA

Pour l'alimentation n°2 : la haute tension électrode intermédiaire

- La partie fonction du nom pour chaque Record : ***LBE2-SD1-HTEI***
- La gamme d'amplitude pour la consigne n°1 : 0 ~ 30kV
- La gamme d'amplitude pour la consigne n°2 : 0 ~ 20 mA

Pour l'alimentation n°3 : la haute tension

- La partie fonction du nom pour chaque Record : ***LBE2-SD1-HT***
- La gamme d'amplitude pour la consigne n°1 : 0 ~ 50 kV
- La gamme d'amplitude pour la consigne n°2 : 0 ~ 100 mA

Les trois alimentations sont des alimentations de tension. Seule l'alimentation HT necessite la commande d'une seconde consigne. En plus de piloter la tension, il est possible d'ajouter une consigne de courant. Sachant que le programme de pilotage des alimentations est générique aux trois alimentations, et que pour l'une d'entre elles il est possible de piloter une consigne secondaire, il a donc été donné la possibilité pour les deux autres alimentations cette fonctionnalité. Même si il est possible de consigner le courant pour les trois alimentations, seule la consigne de courant pour l'alimentation HT sera visible sur le panneau de contrôle principal.

1.4.4 Extrait du script de démarrage général st.cmd

Pour cette application il suffit d'initialiser trois connexions TCP de type modbus pour les trois alimentations. De plus il faut initialiser tous les « *modbus port driver* » pour chaque alimentation, charger le fichier de substitution qui remplacera les bases de données.

```
……
################## ALIMENTATION – 3 kV ############################
##Création de la connexion TCP
drvAsynIPPortConfigure("LBE2-SD1-RE","192.168.1.3:502",0,1,1)
……
```

```
##Ajout des fonctionnalités modbus a cette connexion TCP
modbusInterposeConfig("LBE2-SD1-RE",0,300)

## Lecture 6 * mot à partir de l'adresse 11 à 16
drvModbusAsynConfigure("SD1-RE:Lire_6_11", "SD1-RE", 1, 4, 6, 11, 0, 100, "AlimRE")

## Ecriture 1 * mot à l'adresse 0
drvModbusAsynConfigure("SD1-RE:Ecrire_0_1", "SD1-RE", 1,  6, 0,  1, 0, 1, "AlimRE")

## Ecriture 1 * mot à l'adresse 1
drvModbusAsynConfigure("SD1-RE:Ecrire_1_1", "SD1-RE", 1,  6, 1, 1, 0, 1, "AlimRE")

## Ecriture 2 * mot de l'adresse 6 & 7
drvModbusAsynConfigure("SD1-RE:Ecrire_6_2", "SD1-RE", 1, 16, 6, 2, 0, 1, "AlimRE")

## Ecriture 2 * mot de l'adresse 8 & 9
drvModbusAsynConfigure("SD1-RE:Ecrire_8_2", "SD1-RE", 1, 16, 8, 2, 0, 1, "AlimRE")
############ CHARGEMENT DES BASES DE DONNEES ################
……
dbLoadTemplate("../iocBoot/iocMainWx/db/iocMainWx.substitutions")
```

Les lignes précédentes concernant la configuration de l'alimentation RE, sont à reproduire à l'identique pour les deux autres alimentations : HTEI et HT. Les seules informations qui seront bien évidement différentes seront le nom de la connexion TCP, l'adresse IP des alimentations, et le nom de chaque « modbus port driver ».

1.5 Cage de Faraday CF11

La base de données «EPICS» est composée d'un ensemble de Records. Pour simplifier l'explication la base de données «EPICS» va être décomposée en type d'actions.

Il y a trois types d'actions :

- Les commandes de test avec leur relecture
- Les mesures des deux gammes de courant
- La mesure du bruit

1.5.1 Les commandes de test avec leur relecture

Pour cette fonction il y a deux types de Record : un Record de type « *binary output (bo)* » pour actionner le test, et un Record de type « *binary input (bi)* » pour valider le test. Il faut un Record de chaque type pour chaque test, soit un total de quatre Records. Ces quatre Records seront directement associés à une entrée/sortie de la carte d'entrées/Sorties digitales ICV196.

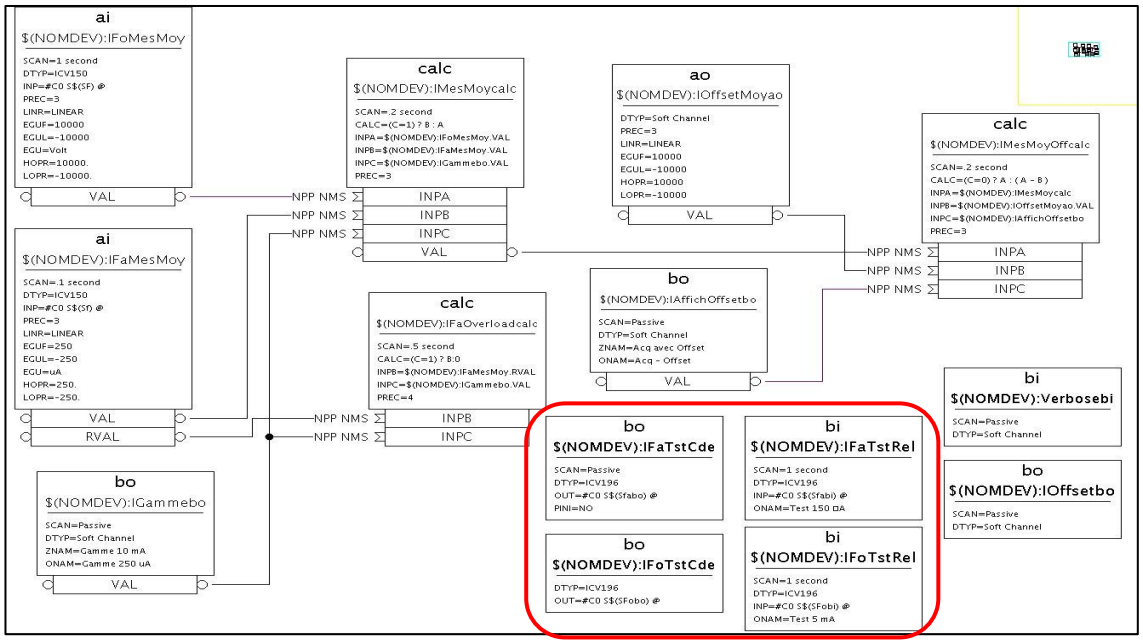


Figure 38 : CF11 - commandes de test avec leur relecture (VDCT)

1.5.2 Les mesures des deux gammes de courant

Pour cette fonction il y a trois types de Records : un Record de type « *analog input (ai)* », un autre de type « *Calc* », et enfin de type « *binary output (bo)* ».

Les Records de type « *analog input (ai)* » sont utilisés pour effectuer les mesures analogiques des courants, et sont donc au nombre de deux. Le Record de type « *binary output (bo)* » permet de sélectionner la gamme de mesure courant faible ou courant fort. Et enfin le Record de type « *calc* » permet de calculer une moyenne du courant mesurée quelque soit la gamme, et de détecter un « *overflow* » dans le cas où la chaine de mesure n'est pas adaptée. Il a été ajouté un mode automatique qui permet de changer la voie de mesure en fonction du courant.

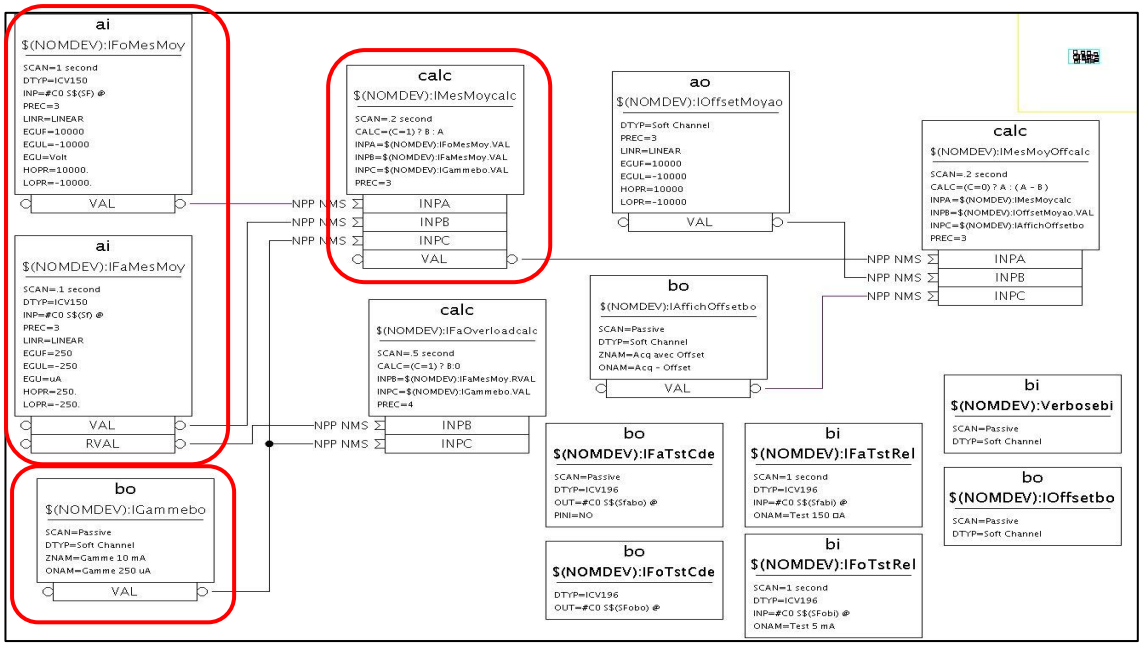


Figure 39 : CF11 - Mesures des deux gammes de courant (VDCT)

1.5.3 *Mesure du bruit*

Cette fonction permet de mesurer le bruit sur chaque voie de mesure. C'est une action qui est à la charge de l'opérateur. Pour cet effet il y a un bouton « *mesure offset* » qui est associé à un Record de type « *binary output (bo)* ». Cette mesure d'offset est enregistrée dans un Record soft de type « *analog outpout (ao)* ». Depuis l'interface, l'utilisateur aura le choix d'afficher la mesure du courant moyen avec ou sans l'offset. Ce calcul sera assuré par un Record de type « calc ».

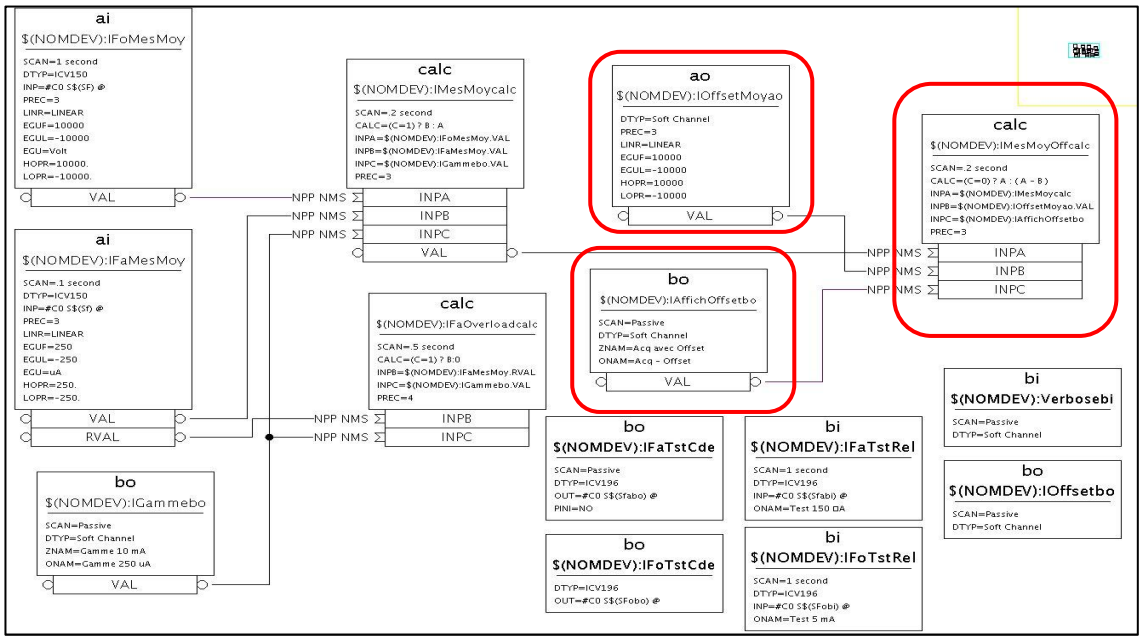


Figure 40 : CF11 - Mesure du bruit (VDCT)

1.5.4 Programme « SNL»

Pour cette application il y a un programme « ***SNL***» qui va interagir avec la base de données «EPICS» :

- *sncCFarad.stt* : Ce programme permet d'effectuer une valeur moyenne du courant mesurée

1.6 Interface Homme/Machine générale de pilotage

Toutes les interfaces utilisateurs ont été crées avec l'outil EPICS EDM dédié à la conception d'interfaces utilisateur choisi pour le projet SPIRAL2.

1.6.1 Lanceur de l'application

C'est le premier panel à être lancé. Ce panel est un menu à partir duquel on peut lancer tous les autres panels pour piloter la source de Deutons et tester de la cage de Faraday CF11.
Ce menu est composé de quatre onglets :

Onglet1 : Synoptiques
Il permet d'accéder à l'interface du synoptique principal, mais aussi à l'interface de test de la cage de Faraday CF11.

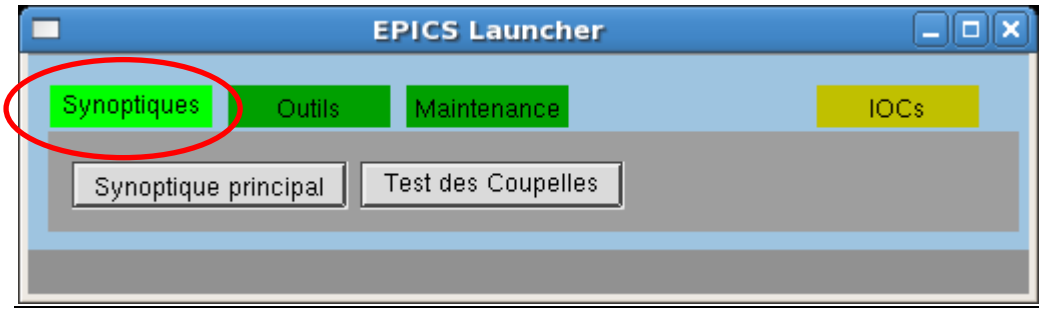


Figure 41 : Launcher – Synotiques (EDM)

Onglet2 : Outils
Cet onglet permet de lancer les différents outils utiles pour analyser les « *process variable* », les enregistrer, tracer des courbes, etc …

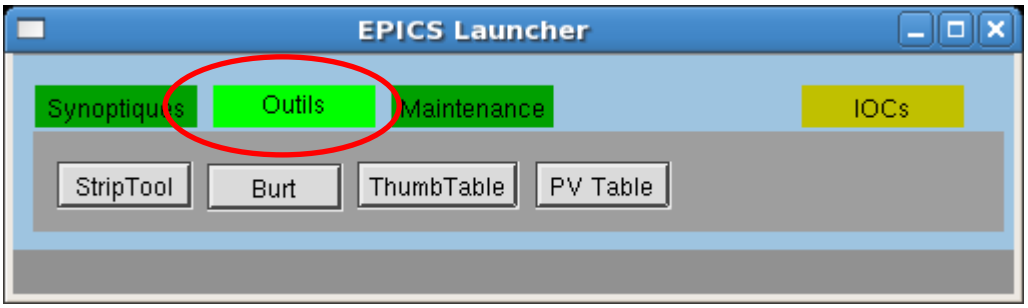


Figure 42 : Launcher –Outils (EDM)

Onglet3 : Maintenance

Cet onglet permet de lancer des outils de maintenance comme la visualisation du fichier log d'erreurs, d'utiliser des sondes, etc …

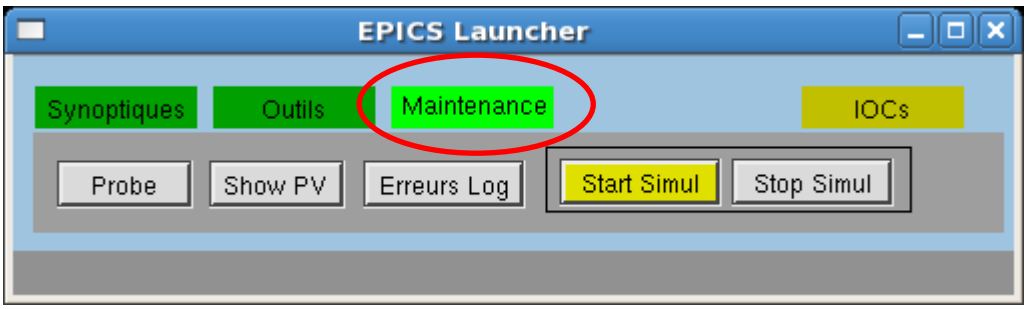


Figure 43 : Launcher – Maintenance (EDM)

Onglet4 : IOC

Cet onglet permet de redémarrer la carte principale, la carte MVME5500, le châssis VMELB2.

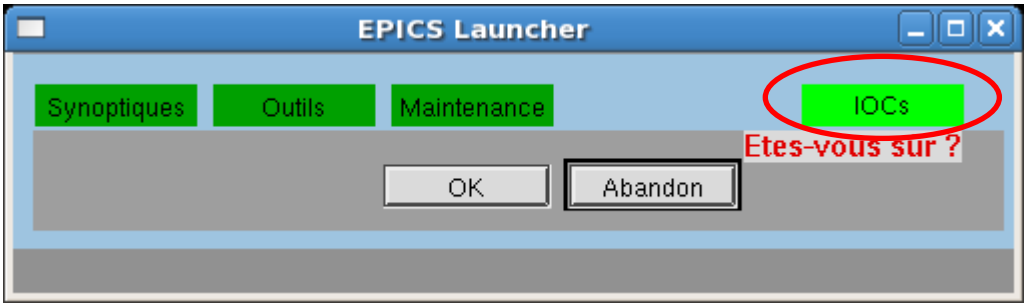


Figure 44 : Launcher – IOC (EDM)

1.6.2 Interface utilisateur générale

Cette interface permet de contrôler l'ensemble du contrôle/commande. Elle est décomposée en deux parties. Tout d'abord il y a le schéma synoptique de l'installation au centre de l'interface. Puis tout autour il y a les commandes associées à chaque instrument. Il y a pour chaque instrument une couleur entre sa représentation graphique sur le synoptique et son onglet de contrôle afin de faciliter la lisibilité. Dans chaque onglet de contrôle il y a les commandes indispensables pour piloter correctement chaque instrument. De plus pour la quasi-totalité des onglets il y a des leds associées à l'état connecté, l'état de sortie, et l'état de pilotage (distant ou local) de l'instrument en question. Et il est possible de visualiser à portée de clic les défauts pour chaque alimentation en cas de problèmes.

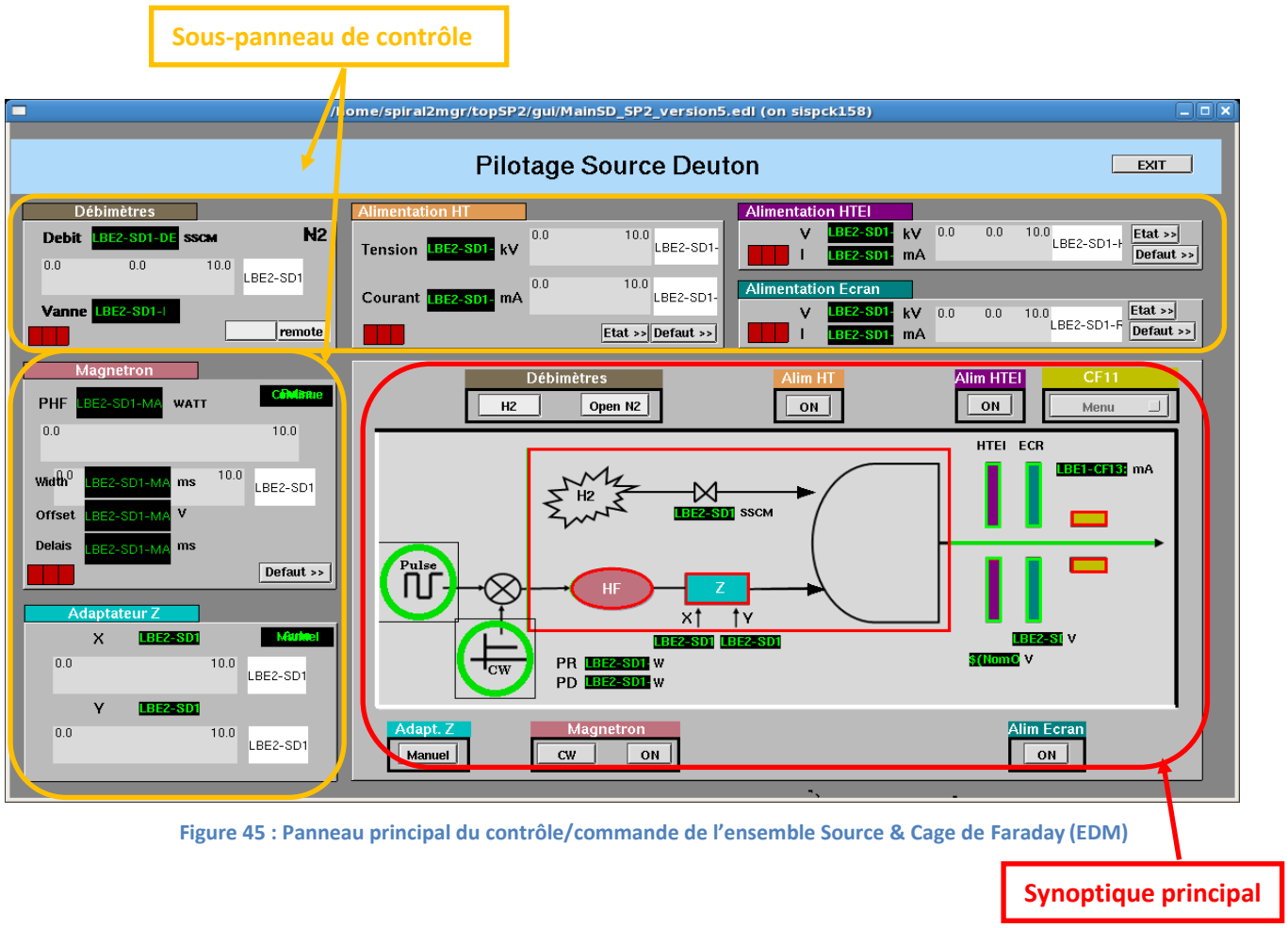


Figure 45 : Panneau principal du contrôle/commande de l'ensemble Source & Cage de Faraday (EDM)

1.6.3 *Interface de tests de la cage de Faraday CF11*

Dans l'onglet Synoptiques du lanceur d'application, il suffit de cliquer sur le bouton « *test des coupelles* » pour avoir l'interface suivante. Celle-ci permet d'effectuer des tests sur l'électronique des cages de Faraday.

Cette interface est composée de deux sous-panneaux :

Un premier, nommé « ***courant*** », permettant de sélectionner la gamme de mesure du courant avec un affichage de la mesure moyennée.

Un second, nommé « ***commandes*** », permettant d'activer deux boutons, un pour le test d'un courant fort de 5mA, et l'autre pour un test de courant faible de 150µA.
Il y a un troisième bouton permettant de mesurer l'offset avec la possibilité d'ajouter ou pas à la mesure moyenne affichée dans l'onglet précédent.

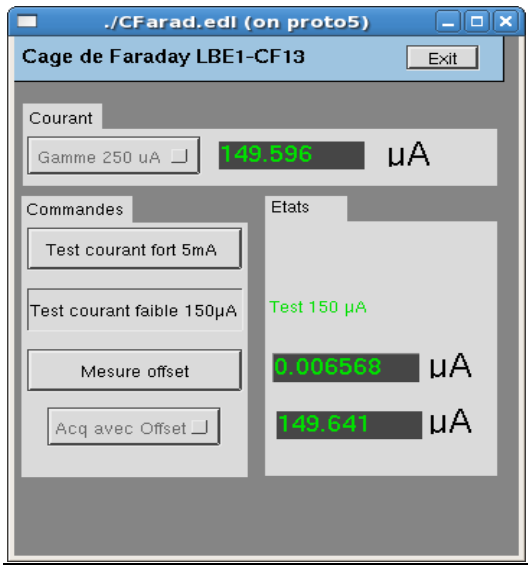


Figure 46 : Panneau de test CF11 (EDM)

1.6.4 Spécifications utilisateur

Spécification n°1 : *Sauvegarde des paramètres*

Il a été demandé d'enregistrer la dernière saisie utilisateur afin d'avoir une persistance des données dans le but, lors du rédémmarrage de la source, de récupérer les derniers paramètres de configuration saisis. Pour cette fonctionnalité il a été utilisé l'outils EPICS « ***autosave*** ». Il suffit de le configurer dans le fichier de script de démarrage.

Voici les quelques lignes à inscrire

```
################ CONFIGURATION D'AUTOSAVE ##############################
# Nombre de fichiers de sauvegarde
save_restoreSet_NumSeqFiles(2)
# Specification de la location du fichier stipulant tous les parameters à surveiller
set_requestfile_path("_TOP_/iocBoot/iocMainWx/db/","")
# Specification de la location  des fichiers de sauvegarde
set_savefile_path("/home/spiral2/data/autosave","")
# Specification des fichiers à restituer lors du démarrage
set_pass0_restoreFile("iocMainWx.sav")
set_pass1_restoreFile("iocMainWx.sav")
################ FIN CONFIGURATION D'AUTOSAVE ##########################
…
#  Pour démarrer le service autosave, avec un temps de répétition de 5s
create_monitor_set("iocMainWx.req",5,"")
```

Spécification n°2 : *Imprimer un rapport de fonctionnement*

Il a été demandé la possibilité d'avoir un journal de bord accessible depuis l'interface utilisateur, et de pouvoir l'imprimer.

Ce rapport ce compose d'une partie texte de quelques mots, et d'un relevé de paramètres. C'est un progarmme complétement indépendant réalisé en JAVA avec un script shell. Le programme a pour but de se connecter à des process variables, lire leur valeur, et ensuite d'en créer un fichier texte avec une mise en forme spécifique, pour être au final imprimer.

4- Contrôle/commande du jeux de fentes FV13 & FH13 sur la ligne LBE1

4.1 Développement EPICS

Aucun jeu de fentes ne sera installé sur la ligne LBE2. Le jeu de fentes FV13 et FH13 est installé sur la ligne LBE1 qui est testée actuellement à Grenoble. Pour ce développement c'est le châssis de la ligne LBE1 qui va être utilisé, soit le châssis **VMELB1**.

Un jeu de fentes est composée d'une fente qui se déplace sur l'axe des X, et une autre qui déplace sur l'axe des Y. Pour caractériser correctement chaque fente avec les bonnes informations un fichier de substitution sera utilisé. Une fente est constituée de deux joues, donc pilotée par deux moteurs identiques, ce qui implique donc que le programme pour piloter la joue de droite et celui pour piloter la joue de gauche, seront aussi identiques. Un seul fichier de substitution sera utilisé pour caractériser les deux joues de la fente FH13, et les deux joues de la fente FV13. Un châssis moteur est composé de 8 cartes contrôleurs, soit la capacité de piloter 4 fentes. Tous les contrôleurs d'un même châssis ont une adresse IP commune via le boitier COMETH, mais une adresse JBus unique. (cf ***Annexe1***) Il vous est présenté par la suite le programme de la joue droite de la fente FH13 :
Le nommage des Records concernant la **J**oue **D**roite(***JD***) de la fente FH13 est : ***LBE1-FH13-JD***.
L'adresse JBUS du contrôleur pour cette joue est : « **10** »

4.1.1 Drivers EPICS

Pour cette application il a été utilisé le Device Support générique « ***Modbus/TCP*** » et des procédures en « ***SNL*** ».

4.1.2 Base de données «EPICS»

La base de données «EPICS» est composée d'un ensemble de Records. Chaque action est composée de un ou plusieurs Records.

Il y a 4 actions :

- Lecture et écriture des paramètres fonctionnels du moteur
- Lecture et écriture des paramètres optionnels du moteur
- Lecture de l'état du moteur
- Lecture de la température du de la joue

Fonction1 : ***Lecture et écriture des paramètres fonctionnels du moteur***

Les paramètres fonctionnels correspondent à la lecture de la position, l'écriture de la consigne, et à la commande d'arrêt du moteur. Cette fonction est composée de trois Records : un Record de type « *analog input (ai)* » pour la position, un autre Record de type « *analog outpout (ao)* » pour la consigne, et enfin un Record de type « *binary output (bo)* » pour la commande d'arrêt. Ces trois Records utilisent des fonctions modbus qui permettent d'écrire ou de lire un mot de 16 bits directement avec l'instrument.

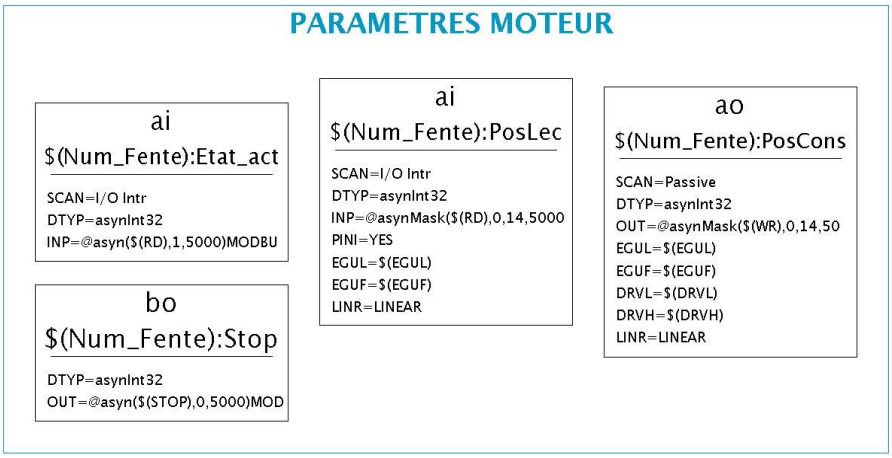


Figure 47 : Fentes - Lecture et écriture des paramètres fonctionnels (VDCT)

Fonction2 : Lecture et écriture des optionnels paramètres du moteur

Les paramètres optionnels correspondent à la vitesse minimum, la vitesse maximum, le courant nominal, le courant de maintien, etc… Pour cette fonction il a été utilisé uniquement des Records de type « *analog output (ao)* » qui permettent d'écrire en modbus un mot de 16 bits dans chaque registre du contrôleur.

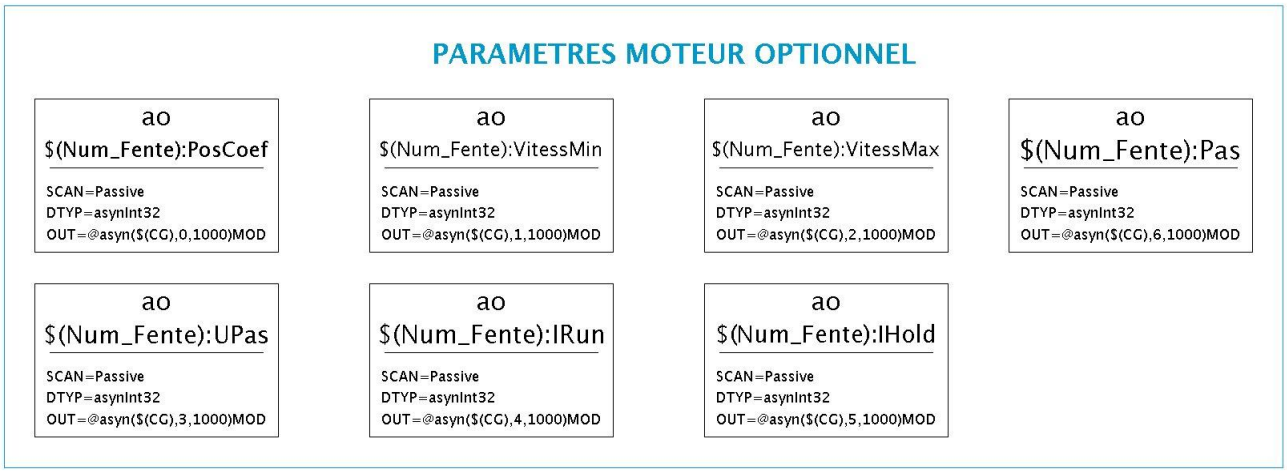


Figure 48 : Fentes - Lecture et écriture des paramètres optionnels (VDCT)

Fonction3 : Lecture de l'état du moteur

Tout un ensemble d'informations concernant l'état du moteur piloté peut être visualisé.

L'état correspond par exemple au mode de pilotage, soit en distant soit en local, à son état d'occupation, aux défauts, etc …Pour cette fonction deux types de Record on été utilisés : un Record de type « *analog input (ai)* », et un Record soft de type « *binary output (bo)* ». Le Record de type « *ai* » permet de récupérer un mot de 16bits reflétant l'état du moteur, et les Records soft de type « *bo* », permettent via un programme « ***SNL***» d'associer une led sur l'interface utilisateur à chaque défaut moteur.

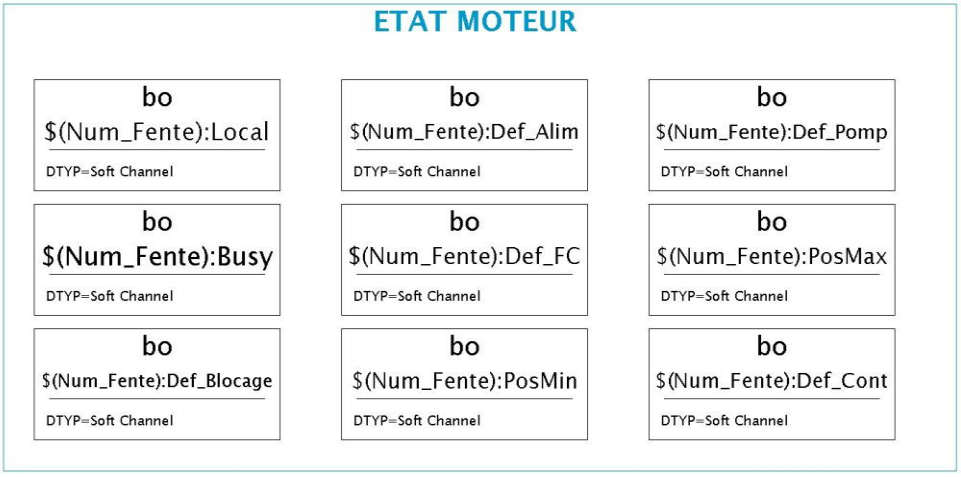


Figure 49 : Fentes – Etat du moteur (VDCT)

Fonction4 : Lecture de la température de la joue

Un seul Record de type « *analog input (ai)* » est directement associé à une entrée matérielle de la carte ICV150 de type ADC pour lire la température. C'est un Record qui est rafraichis toutes les 500ms.

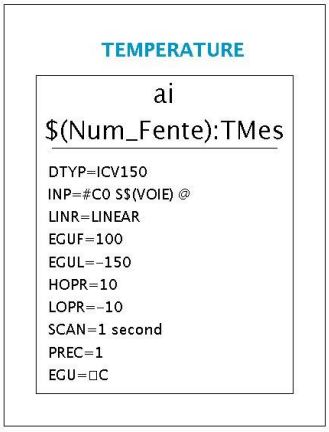


Figure 50 : fentes - Lecture de la température de la joue (VDCT)

4.1.3 Programme « SNL»

Pour cette application il y a un programme « ***SNL***» qui interagit avec la base de données «EPICS» :

- *Fentes.stt* : une procédure « ***SNL***» est utilisée pour associer une led sur l'interface utilisateur à chaque défaut moteur.

4.1.4 Script de démarrage

Pour cette application il faut initialiser une connexion TCP (Annexe1) avec le boitier COMETH (Annexe2), de charger la base de données «EPICS», et de lancer le séquenceur avec le fichier « ***SNL***».

```
.....
##Création de la connexion TCP entre le PC et le boitier COMETH
drvAsynIPPortConfigure("LBE2-SD1-FH13","192.168.1.3:502",0,1,1)

##Ajout des fonctionnalités modbus a cette connexion TCP
modbusInterposeConfig("LBE2-SD1-FH13",0,300)

## Ecriture 1 * mot à partir de l'adresse 0 pour la consigne de position
drvModbusAsynConfigure("WR_JD", " LBE2-SD1-FH13", 10, 6, 0, 1, 0, 100, " ")

## Lecture 2 * mot de l'adresse 1 à l'adresse 2 : Position et Etat moteur
drvModbusAsynConfigure("RD_JD", " LBE2-SD1-FH13", 10,  3, 1,  2, 0, 100, " ")

## Ecriture 1 * mot de l'adresse 3 à l'adresse 9 : Paramètres configuration moteur
drvModbusAsynConfigure("WR_JD_Cfg", "LBE2-SD1-FH13", 10,  6, 3, 9, 0, 100, "")
```

```
## Ecriture 1 * mot à l'adresse 14 pour l'arrêt d'urgence
drvModbusAsynConfigure("WR_JD_Stop", " LBE2-SD1-FH13", 10, 6, 14, 1, 0, 100, " ")
############ CHARGEMENT DES BASES DE DONNEES ################
…..
dbLoadTemplate "../iocBoot/iocFentes/boot/iocFentes.substitutions"
################ LANCEMENT DES TACHES SNL ##################
…..
## Lancement d'une tache SNL
seq &sncFentes,"NomOp=LBE2-FH13-JD"
```

4.2 Interface Homme/Machine de pilotage

L'interface permet de piloter les deux joues d'une fente ; qu'elle soit horizontale ou verticale. Les panneaux de pilotage qui vont être présentés par la suite, représentent le contrôle de la fente horizontale LBE1-FH13 sur la ligne LBE1.

Cette interface est composée de deux panneaux :

- Un panneau qui permet de piloter les fentes **(*panneau de contrôle)***
- Un panneau qui permet de configurer chaque contrôleur d'une fente **(*Panneau de configuration)***

4.2.1 Panneau de contrôle

Il y a deux commandes,

Consigne de déplacement en « mm »	*cmd n° 1*
Arrêt d'urgence du moteur	*cmd n° 2*

Et trois lectures,

Position de la joue en « mm »	*lect n° 1*
Etats du contrôleur (distant, erreurs, etc…)	*lect n° 2*
Température de la joue « °C »	*lect n° 3*

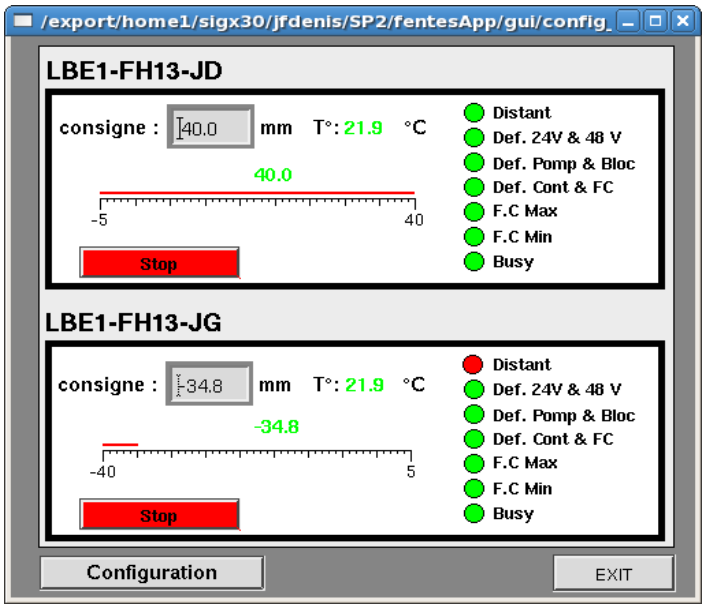


Figure 51 : Interface de pilotage des fentes (EDM)

Sur l'interface de contrôle, on peut voir un bouton « ***Configuration*** ». Il permet d'accéder à l'interface suivante qui a pour objectif de configurer le contrôleur de chaque joue en question.

4.2.2 Panneau de configuration

Nombre de pas	*cmd n° 1*
Définition du µpas	*cmd n° 2*
Vitesse minimun	*cmd n° 3*
Vitesse maximun	*cmd n° 4*
Courant de running	*cmd n° 5*
Courant de maintient	*cmd n° 6*
Coefficient démultiplicateur	*cmd n° 7*

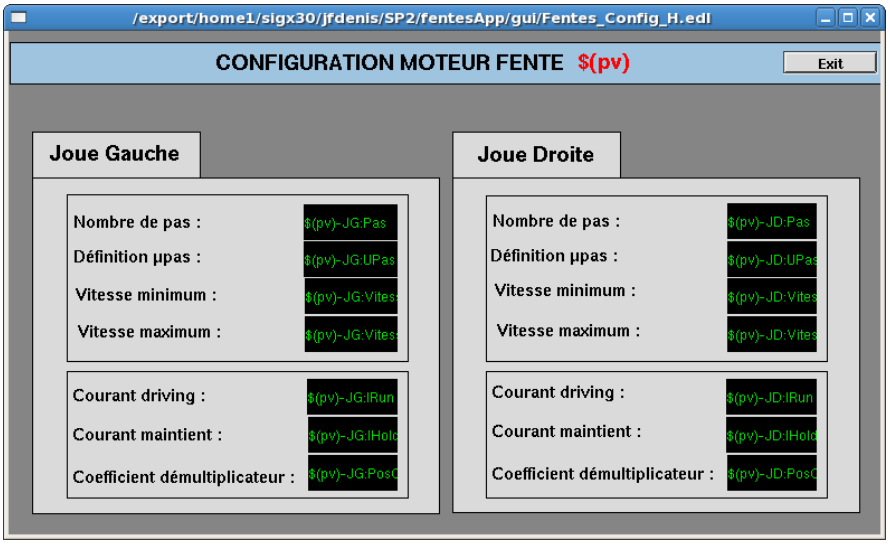


Figure 52 : Interface de configuration des deux contrôleurs d'une fente (EDM)

VIII – Les tests

1- Les tests unitaires

2- Les tests d'intégration

3- Les premiers tests du contrôle/commande de la source en grandeur nature

4- Les tests à Saclay de la ligne LBE2

4.1 Phase 1 : Décembre à Février 2010

4.2 Phase 2 : Juin 2010 à Septembre 2010

4.3 Phase 3 : A partir de Décembre 2010

5- Les tests du jeux de fentes FV13 et FH13 sur la ligne LBE1 à grenoble

1- Les tests unitaires

Pour la quasi-totalité des programmes à développer concernant la source de Deutons et les fentes, les instruments étaient à disposition ce qui a permis de valider directement le bon fonctionnement des programmes à chaque conception de l'un d'eux. Une interface de pilotage dédiée pour chaque instrument a été développée dans ce sens. Cette méthode a permis pour chaque instrument de valider avec l'utilisateur final les informations utiles et non utiles à intégrer sur l'interface de pilotage.
Concernant la cage de Faraday, un simulateur de courant était à disposition afin de tester le dispositif et son interface de pilotage.

2- Les tests d'intégration

Après avoir concu une application pour chaque instrument ainsi qu'une interface utilisateur dédiée à chacun, j'ai commencé l'intégration module applicatif par module applicatif. J'ai créé pour cela dans le laboratoire une petite maquette version allégée de l'ensemble contrôle/commande de la source avec la majorité des instruments. Pour les instruments qui n'étaient pas à disposition j'ai simulé des entrés/sorties afin de reproduire leur fonctionnement. J'ai reproduis au plus près les conditions dans lesquels les premiers tests faisceaux de la source deuton prévus théoriquement en février 2009 se dérouleront, entre autres concernant l'infrastructure réseau. Il a été demandé de pouvoir faire fonctionner le contrôle/commande sans le réseau au cas ou le réseau ne serait pas disponible. J'ai pour cela créé un réseau local avec un switch entre le PC du contôle/commande, le châssis VME, et tous les instruments. Une interface générale correspondant aux besoins des utilisateurs à été réalisée.

Figure 53 : Banc de test pour simuler le contrôle/commande de la source deuton

Dans cette phase de tests j'ai pu vérifier point par point chaque fonctionnalité qui était demandée, et commencer à concevoir l'interface générale de pilotage de la source de Deutons.

3- Les premiers tests du contrôle/commande de la source en grandeur nature

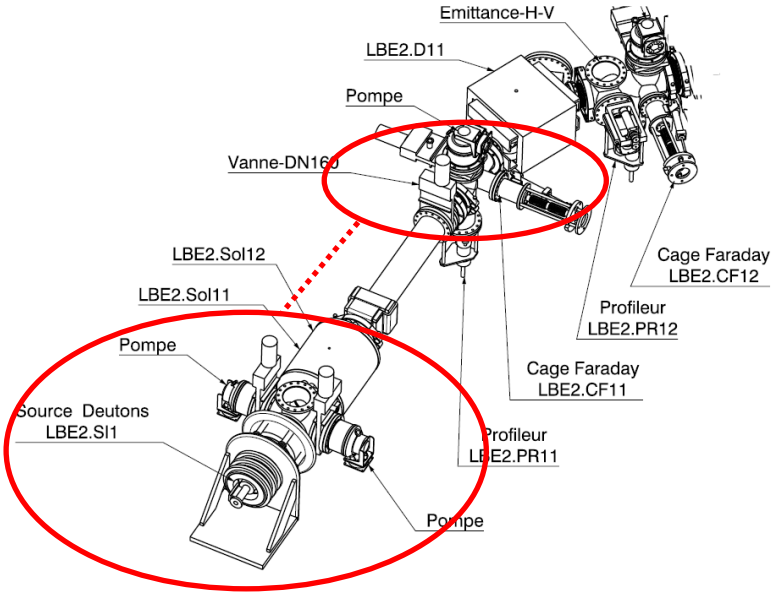
Après avoir similé au maximun le contrôle/commande de la source de Deutons, il m'a été proposé de tester l'ensemble du contrôle/commande à l'aide d'un banc de test, appelée BETSI (Banc d'Etude des Sources d'Ions). Ce banc permet de se focaliser sur la production du plasma et sur l'extraction du faisceau. Le contrôle/commande est plus ou moins similaire puisqu'il fonctionne avec un magnétron pouvant être contrôlé en continu ou en pulsé. J'ai donc déplacé tout l'expérience du laboratoire dans lequel j'avais effectué les tests d'intégration vers le hall ou se trouve BETSI. Après avoir correctement tout câblé nous avons effectué les premiers tests et validé une partie du contrôle/commande. Avec ces tests nous avons validé la partie pilotage magnétron (Pulsé et continu), l'adaptateur d'impédance, et le contrôleur d'injection de gaz. Une partie importante du contrôle/commande de la source de Deutons a été validé ce qui nous évitera propablement des surprises lors du démarrage réel de la source de Deutons.

Figure 54 : Banc de test BETSI

4- Les tests à Saclay de la ligne LBE2

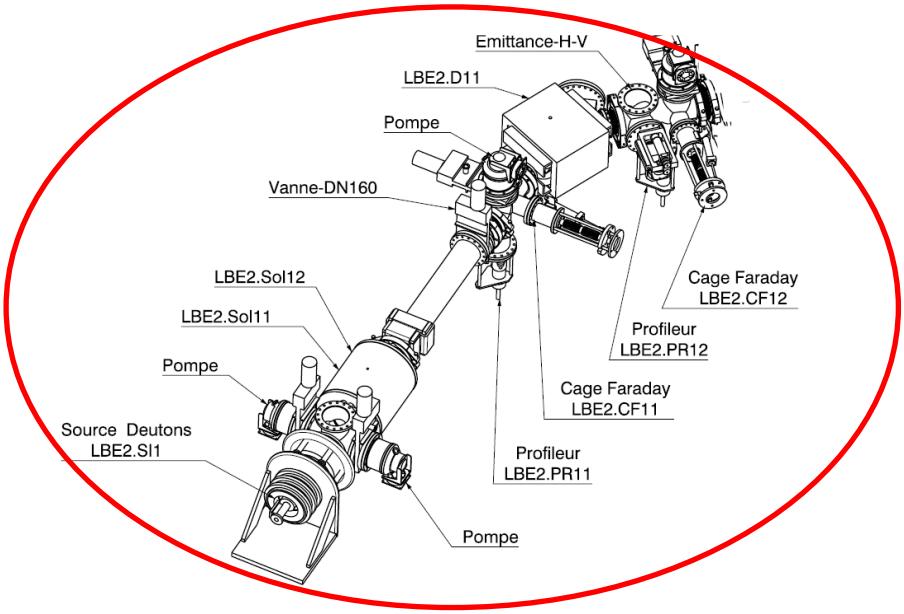
Il a été convenu que des tests regroupant l'ensemble des modules de la ligne LBE2 devront être effectués à Saclay. Ces tests s'effectueront en trois phases :

4.1 Phase 1 : Décembre à Février 2010



La ligne comprend la source de deuton suivi de la cage de Faraday CF11. Cette première phase à pris un peu de retard, et devrait non pas se dérouler en décembre 2009 mais plutôt en mars 2010. Cette phase permettra de tester l'ensemble du contrôle/commande de la source de deuton dans son contexte final.

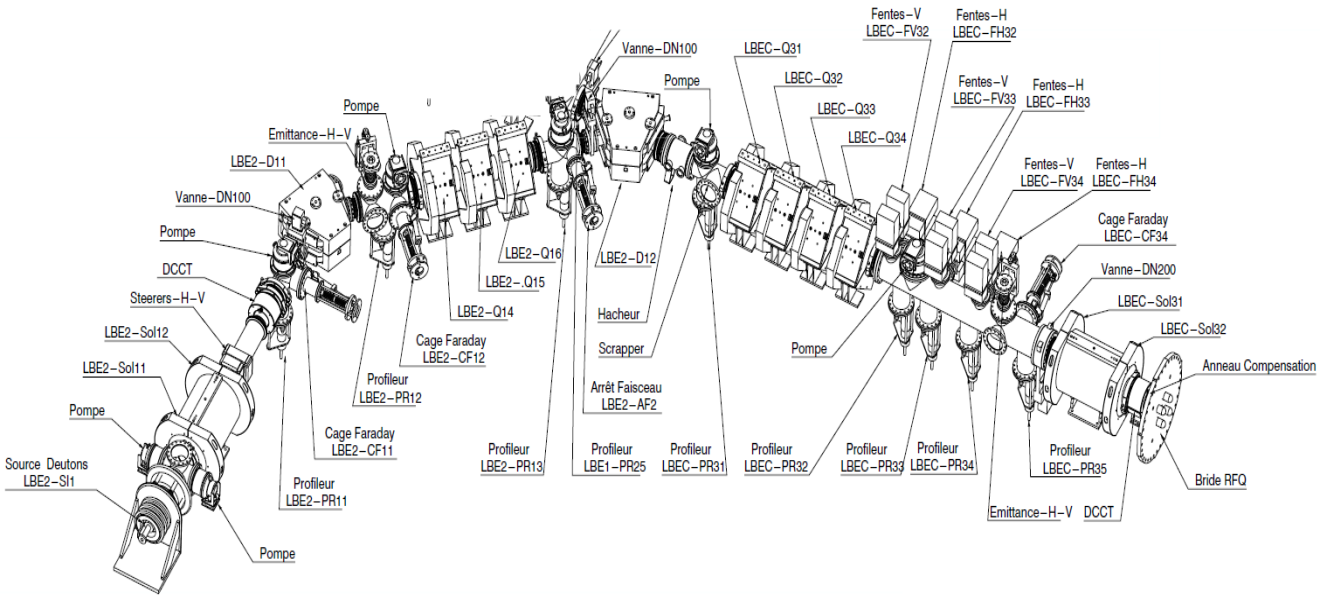
4.2 Phase 2 : Juin 2010 à Septembre 2010



C'est l'ensemble des instruments de la ligne LBE2 qui sera testé dans cette phase.

Cette deuxième phase permettra de tester les éventuelles corrections apportées suite aux tests de la phase 1, et d'apporter d'éventuelles fonctionnalités supplémentaires demandées par les opérateurs.

4.3 Phase 3 : A partir de Décembre 2010



Ligne montée jusqu'à la CF LBE-CF34. Le hacheur et son scrapper associé seront présents.

Dans cette troisième phase j'intégrerai 6 jeux de fentes, ainsi que le « scrapper ». Le pilotage du « scrapper » est identique au pilotage d'une joue de fente.

5- Les tests du jeux de fentes FV13 et FH13 sur la ligne LBE1 à Grenoble

Les tests pour le jeux de fentes FV13 et FH13 se sont réalisés à Grenoble sur la ligne LBE1. Toute la partie contrôle/commande avait été testé en laboratoire avec un moteur. A Grenoble il a suffit d'intégrer le développement dans l'architecture topSP2 regroupant toutes les applications déjà réalisées. Les tests ont été un succés. Nous avons pu valider la partie contrôle/commande du jeu de fente et toute la partie mécanique de celles-ci

Figure 55 : Châssis moteur Fentes

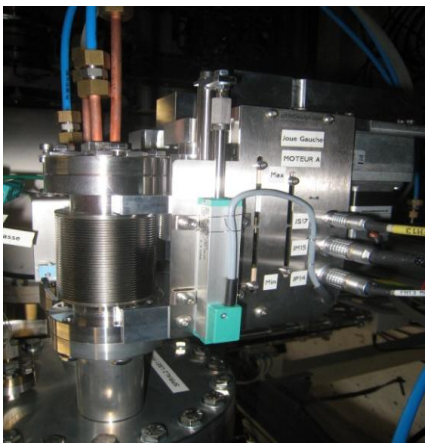


Figure 56 : Joue gauche (FH13-JG)

Les problèmes rencontrés

J'ai tout d'abord rencontré quelques difficultés lors de la mise en œuvre d'EPICS. Effectivement il y avait très peu de documentation sur web.

Pour le contrôle/commande de la source de Deutons j'ai eu des soucis d'approvisionnement de matériel. Du fait que le matériel ai été commandé par le GANIL, tout leur était livré à Caen. Ensuite, après quelques tests réalisés afin de valider le cahier des charges de chaque instrument, la livraison été effectuée à Saclay. Des problèmes avec les alimentations haute tension ont été constatés, en l'occurence l'alimentation 50kV dont ses temps de réponses en charge ne répondaient pas au cahier des charges. Tout est rentré dans l'ordre au bout de quelques mois. C'est pendant ces quelques mois que j'ai travaillé sur le contrôle/commande des fentes.

J'ai rencontré d'autres difficultés concernant la mise en oeuvre du driver modbus lors du développement du contrôle/commande des fentes. Effectivement ce driver n'avait jamais été testé en mode RS485 avec plusieurs esclaves. J'ai mis un certain temps avant de remettre en question le fonctionnement du driver. Après avoir revérifié plusieurs fois la configuration du matériel, j'ai approfondi mes recherches dans le code source du driver ce qui m'a permis d'utiliser le logger intégré à celui-ci. Là j'ai pu constater que la liaison physique (le port série) n'était pas exlusive à un seul thread. Un thread était créé pour chaque esclave afin d'émettre une requête sans se soucier de l'activité du port. J'ai donc, avec l'aide d'un collègue, modifié le code source du driver afin que la liaison physique soit exclusive à un seul esclave à un instant donné.

Conclusion et perspectives

1- Bilan

L'objectif initial de ce mémoire était la réalisation du contrôle/commande sous EPICS permettant d'assurer le pilotage de la source de Deutons de l'injecteur de SPIRAL2. Cependant cet objectif a évolué au cours du temps. Pour réaliser ce mémoire j'ai intégré le laboratoire LD2I ou j'ai pris la responsabilité du contrôle/commande concernant deux projets.

Mon travail s'est déroulé en deux temps :
Dans un premier temps d'environ trois mois je me suis familiarisé avec EPICS. Ensuite au cours du second, j'ai réalisé le développement du contrôle/commande la source de Deutons. Celui-ci s'est réalisé en deux parties entrecoupées d'une période de dix mois (suite à des problèmes d'approvisionnement) pendant laquelle j'ai travaillé sur un autre projet de l'injecteur SPIRAL2 : le contrôle/commande des fentes.

Le contrôle/commande de la source de Deutons m'a permis d'appréhender d'autres domaines que la programmation : tels que l'aspect matériel concernant l'adaptation des signaux des cartes électroniques, la configuration des systèmes sous linux, le développement des « ***makefiles*** » dans mes applications afin de générer correctement mes fichiers sources, etc… Du point de vue électronique je n'ai rencontré aucune difficulté étant donné ma formation initiale d'électronicien. Le domaine traitant exclusivement de l'informatique fut lui très enrichissant et m'a permis d'acquérir des connaissances dans le domaine des systèmes temps réels via les échanges collaboratifs dans mon équipe.

Des tests du contrôle/commande de la source de Deutons ont été réalisés avec succès sur le banc d'essais BETSI, mais le test réel du contrôle/commande de la source deuton ne s'effectuera qu'en février 2010.

Le contrôle/commande des fentes m'a demandé, outre les capacités acquises précédemment, une faculté d'analyses. En effet afin de régler le problème de communication j'ai utilisé plusieurs outils pour diagnostiquer la panne. J'ai envisagé plusieurs causes possibles : Une erreur de programmation, une mauvaise configuration du COMETH, un problème de conception de la carte JBUS... Suite à plusieurs vérifications de la programmation, j'ai utilisé l'outil « ***wireshark*** » pour étudier les trames réseaux entres

le PC et le COMETH, un oscilloscope pour visualiser les trames RS485,etc… C'est le logger intégré au driver qui m'a amené à trouver le problème.

J'ai poussé mes recherches jusqu'au code source du driver pour comprendre d'où venait l'erreur et la modifier.

Concernant la programmation j'ai rencontré des difficultés au début pour appréhender EPICS, mais aujourd'hui je commence à mieux le comprendre et à en voir toute la puissance, et ses limites. J'ai toutefois été déçu par les outils créant des interfaces graphiques, notamment EDM, un peu obsolète selon moi. Cependant un Channel Access en Java a été développé par la communauté EPICS et m'a permis, malgré une configuration un peu difficile, de réaliser mes premières interfaces Java avec EPICS. Certes j'aurais pu développer le contrôle/commande de la source de Deutons, et des fentes en Java, mais par souci d'homogénéité sur le projet SPIRAL2 j'ai du développé avec le logiciel d'interface EDM d'EPICS. Etant donné mes compétences actuelles en JAVA, je serais intéressé d'approfondir la question.

2- Ouverture et évolution du projet SPIRAL2

C'est un projet que je vais maintenir et faire évoluer jusqu'à installation sur site au GANIL à Caen. Afin de le faire évoluer, j'intégrerai ou modifierai des fonctionnalités demandées par les spécialistes des sources au cours des différentes phases de tests à Saclay.

3- Mes futures missions d'ingénieur

Pour cette année 2010 je vais travailler sur plusieurs projets :

- J'ai en charge tout l'ensemble contrôle/commande de la ligne LBE2 durant les différentes phases de tests à Saclay
- Je vais être en charge d'une partie du contrôle/commande EPICS de « S3 » qui sera la première salle d'expérience du projet de SPIRAL2
- Je suis toujours responsable du contrôle/commande des fentes
- De mon côté je vais continuer à suivre l'évolution de Java pour EPICS

Annexe1 : Le protocole ModBus

Modbus est un protocole de communication utilisé principalement pour des réseaux d'automates programmables. Il fonctionne sur le mode maître / esclave. Il est constitué de trames contenant l'adresse de l'esclave concerné, la fonction à traiter (écriture, lecture), la donnée et le code de vérification d'erreur appelé contrôle de redondance cyclique (CRC) sur 16 bits. Le protocole Modbus est un protocole de dialogue basé sur une structure hiérarchisée entre un maître et plusieurs esclaves.

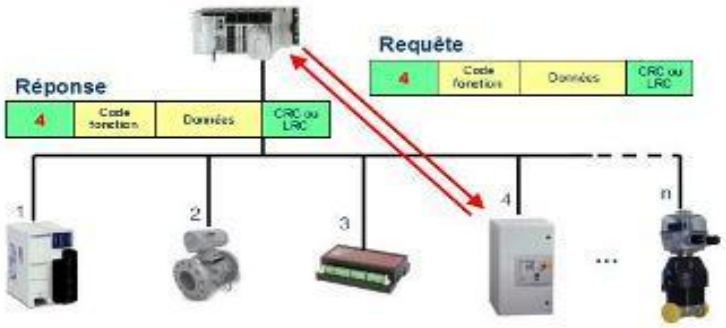
Le protocole Modbus peut être implémenté :

- Sur une liaison série asynchrone de type RS-232, RS-422 ou RS-485,
- Via TCP/IP sur Ethernet ; on parle alors de *Modbus TCP/IP* ;

L'accès à ces bits/registres, se fait par l'intermédiaire de fonctions MODBUS standardisées. Les fonctions suivantes sont standardisées:

				Function Codes		
				code	*Sub code*	*(hex)*
Data Access	Bit access	Physical Discrete Inputs	Read Discrete Inputs	02		02
		Internal Bits Or Physical coils	Read Coils	01		01
			Write Single Coil	05		05
			Write Multiple Coils	15		0F
	16 bits access	Physical Input Registers	Read Input Register	04		04
		Internal Registers Or Physical Output Registers	Read Holding Registers	03		03
			Write Single Register	06		06
			Write Multiple Registers	16		10
			Read/Write Multiple Registers	23		17
			Mask Write Register	22		16
			Read FIFO queue	24		18
	File record access		Read File record	20	6	14
			Write File record	21	6	15
Diagnostics			Read Exception status	07		07
			Diagnostic	08	00-18,20	08
			Get Com event counter	11		OB
			Get Com Event Log	12		0C
			Report Slave ID	17		11
			Read device Identification	43	14	2B
Other			Encapsulated Interface Transport	43	13,14	2B

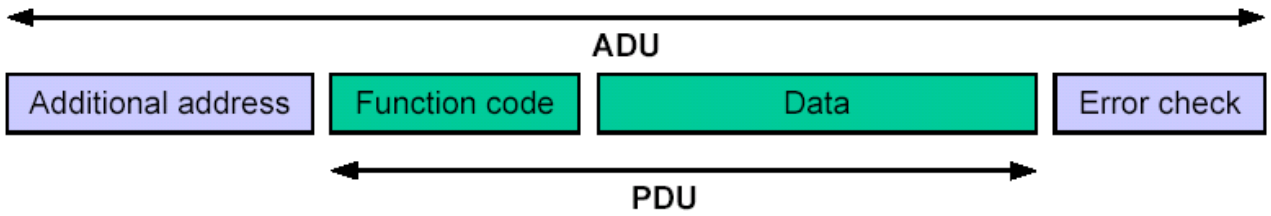
Le maitre (client) envoie une requête à destination d'un esclave (serveur):



L'échange est purement de type maître/esclave avec MODBUS série (1 requête doit être suivie d'une réponse, avant de pouvoir envoyer une nouvelle requête.

Avec MODBUS/TCP, les échanges sont de type client/serveur. Un client peut envoyer une requête vers un serveur, sans avoir forcément reçu la réponse à la requête précédente.

Le format général d'une trame MODBUS est le suivant



La requête contient :

- L'adresse de l'esclave à interroger,
- Un code fonction, qui indique le type d'action à exécuter (lecture bit, écriture registre,…),
- La plage de bits/registres concernés,
- Les données à écrire dans le cas d'une écriture.

La réponse contient :

- L'adresse de l'esclave qui répond,
- Un code fonction, qui indique le type d'action exécutée,
- Le nombre d'octets de données compris dans la réponse,
- Les données lues dans le cas d'une lecture.

Annexe2 : Configuration d'une connexion ModBus TCP sous EPICS

Configuration

Tout d'abord il faut créer une connexion de type TCP/IP. Pour cela nous allons utiliser le driver « ASYN », et plus particulièrement la fonction « *drvAsynIPPortConfigure* ». Celle-ci s'utilise de la manière suivante :

drvAsynIPPortConfigure(PortName, Adress. host, priority, noAutoConnect, noProcessEos)

Nous avons une connexion TCP. Maintenant nous voulons que cette connexion fonctionne en modbus. Pour cela nous allons utiliser la fonction « *modbusInterposeConfig* » du driver « Modbus ». Celle-ci va ajouter toutes les fonctionnalités modbus aux trames de la connexion TCP/IP.
Elle se configure comme ceci:

modbusInterposeConfig(*portName, linkType, timeoutMsec)*

Nous avons une connexion TCP qui communique en modbus. Il nous suffit maintenant de créer ce qu'on appelle un ou plusieurs « **modbus** port driver », qui correspondent à une fonction modbus associée à une connexion TCP et à un ou plusieurs registres de l'appareil à écrire ou à lire.

drvModbusAsynConfigure *(portName, tcpPortName, slaveAddress, modbusFunction, modbusStartAddress, modbusLength, dataType, pollMsec, plcType)*

Dans cette fonction nous devons définir le nom de la connexion modbus/TCP, l'adresse de l'appareil (par défaut 1), la fonction modbus utilisée, le premier registre dans lequel lire ou écrire, le nombre de mot de 16 bits à écrire, et la vitesse de rafraichissement.

Suite aux corrections du driver « Modbus », nous avons la possibilité d'associer plusieurs esclaves à un seul port physique. Seulement deux fonctions ont été modifiées : « *modbusInterposeConfig* », et « *drvModbusAsynConfigure* ».
Toutes ces fonctions sont à mettre dans le script de démarrage de l'IOC.

exemple *:*

Nous devons piloter un appareil AGILENT avec une adresse IP « 132.166.33.202 », ou la table des registres est la suivante :

Adresse	Désignation
0	Consigne tension
1	Consigne courant
2	Mesure tension
3	Mesure courant

Voici quelques lignes à rajouter dans le script de démarrage (*.cmd) pour configurer le port TCP (*ENET1*) de la carte MVME5500:

drvAsynIPPortConfigure("AGILENT_TCP" , "132.166.33.202" , 0 , 0 , 0);

L'adresse « 132.166.33.202 » est l'adresse IP de l'appareil, et « *AGILENT_TCP* » est le nom donné à cette connexion.

modbusInterposeConfig("AGILENT_TCP " , 0 , 400);

Le type de connexion est TCP, soit 0, et le TimeOUT est de 400ms

Et ensuite nous définissons chaque « **modbus** port driver » dans lesquelles nous voulons écrire ou lire:

Registre « ***consigne Tension et Courant***»

drvModbusAsynConfigure("Consigne", "AGILENT_TCP", 1 , 6 , 0 , 2 , 0 , 0 ," ");

Nous utilisons la connexion « AGILENT_TCP », la fonction modbus « 16 » qui correspond à l'écriture de n mot de 16bits, à partir de l'adresse registre 0 et de longueur de 2 mots, correspondant bien sur à la tension et au courant.

Registre « ***Lecture Tension et Courant***»

drvModbusAsynConfigure("Mesures", "AGILENT_TCP", 1 , 4 , 2 , 2 , 0 , 0 ," ");

Nous utilisons la connexion « AGILENT_TCP », la fonction modbus « 04 » qui correspond à la lecture de n mot de 16bits, à partir de l'adresse registre 2 et de longueur de 2 mots.

Procédure d'utilisation

Dés que l'étape de configuration de la connexion a été réalisée, nous devons créer nos Records dans la base de données «EPICS» et les configurer comme ceci:

Depuis le logiciel de création de base de données «EPICS» VDCT :

1- Sélectionner l'interface asyn dans l'attribut « ***DTYP*** » :
   - asynUInt32Digital
   - asynInt32
   - asynInt32Array
   - asynFloat64
   - asynCommon
   - asynDrvUser

2- Ajouter dans l'attribut « ***OUT*** » du Record :

   @asyn(TCP_COM,0,1000) *ou* ***@asynMask(TCP_COM,14,0,1000)***

Le Record est prêt à émettre sa valeur au registre en question par le protocole Modbus TCP.

Annexe3 : Le boitier COMETH

Ce boitier nous permet de communiquer par l'intermédiaire du réseau avec un instrument sous deux modes :

- Mode 1 : En modbus TCP/IP ou RS485.
- Mode 2 : En port série déporté RS232

Nous allons utiliser les deux modes. Le premier sera utile pour le pilotage des fentes et des alimentations, et le second pour le pilotage du contrôleur du débit de gaz.

Configuration

La configuration se fait par un utilitaire fournit avec le boitier, qui ne fonctionne que sur Windows. Nous avons la possibilité d'utiliser un HyperTerminal avec la configuration suivante : ***2400, 8, n, 1, n***

Etape 1 :

Il faut utiliser un câble de type croisé entre le port RS232 du boitier et celui de l'ordinateur. Pour que le boitier soit configurable, il faut positionner le bouton ADM en mode « Administration ». Ensuite nous utilisons l'HyperTerminal, et démarrons le boitier en l'alimentant. Si tout fonctionne correctement nous devons avoir la trame suivante qui s'affiche sur l'hyperterminal:

Mode1 :

```
root> COMETH MODBUS version 2.6.0.0, Administration mode ready
root>
```

Mode2:

```
root> COMETH SERVERCOM version 2.2.0.1 , Administration mode ready
root>
```

Etape 2 :

Configuration du boitier :

Réseau :

```
root> set net gateWay « adresse passerelle »
root> set net IP « adresse IP »
root> set net mask 255.255.255.0
root> set net DHCP off
```

Lorsque la connexion est configure en sous réseaux, l'adresse passerelle est 0.0.0.0, et l'adresse de chaque esclave est de type : 192.168.1.xx….

Mode 1 ; Liaison RS485:

```
root> set serial interface rs485 noecho
root> set serial baudrate 19200
root> set serial format 8 n 1
```

Mode 2 ; Liaison RS232:

```
root> set serial interface rs232 noecho
root> set serial baudrate 19200
root> set serial format 8 n 1
```

Il suffit de sauvegarder la nouvelle configuration avec la commande:

```
root> save
```

Après avoir correctement configuré la liaison TCP, le boitier peut fonctionner sur le réseau.

Pour vérifier son bon fonctionnement :

- connecté le sur le PORT ENET1

  Exécutez la commande suivante sur un terminal : ***Ping « 132.166.33.202 »***

Annexe4 : Quelques notions importantes

Qu'est-ce qu'un atome ?

Un atome est constitué d'électrons en orbite autour d'un noyau. Ce noyau est composé de deux types de particules, appelés ***nucléons*** :

- des ***protons***, qui sont des charges électriques positives,
- des ***neutrons***, qui n'ont pas des charges électriques

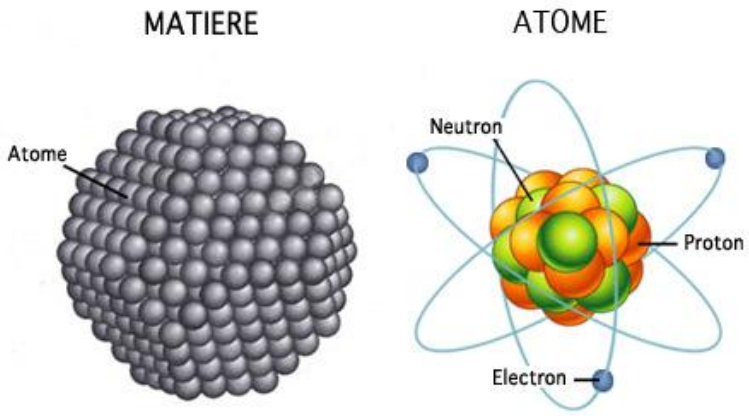


Figure 57 : Composition d'un atome

La charge électrique de l'atome est neutre, car il y a autant de protons que d'électrons, c'est-à-dire autant de charges électriques positives que de charges négatives.

Le noyau est cent mille fois plus petit que l'atome, mais compose tout de même 99% du poids de l'atome. Le nombre de protons dans un atome détermine ses propriétés physiques et chimiques, c'est ce qu'on appelle le ***numéro atomique (Z)***. Et le nombre de nucléons (protons+neutrons) qu'il contient, c'est ce qu'on appelle le ***nombre de masse d'un noyau (A)***.

Les électrons et le noyau d'un atome sont liés par l'***interaction électromagnétique***. En effet, le noyau atomique a une charge électrique positive, grâce aux protons, alors que les électrons qui gravitent autour ont une charge électrique négative. Ainsi, parce que leurs charges électriques sont opposées, le noyau atomique et les électrons s'attirent, ce qui permet aux atomes de ne pas perdre leurs électrons.

A l'intérieur du noyau, les protons et les neutrons sont collés entre eux grâce à une force de cohésion appelée ***interaction nucléaire***. On dit alors que le noyau est ***stable***.

Qu'est qu'un ion ?

Un ion est un atome qui a perdu ou gagner un électron, ce qui le rend donc électriquement chargé. L'***ionisation*** de la matière consiste à arracher des électrons aux atomes. La grande majorité des systèmes de détection des particules est basée sur l'ionisation. De même les sources d'ions ont pour rôle d'arracher les électrons aux atomes.

Qu'est ce qu'un noyau radioactif ?

Certains noyaux contiennent trop de particules ou renferment trop d'énergie de sorte que la force de cohésion n'est plus suffisante pour maintenir les protons et les neutrons ensemble. Les noyaux sont dits instables. Ils finissent par libérer leur trop plein d'énergie en émettant des rayonnements. C'est ce phénomène de désintégration que l'on appelle la radioactivité. Ces noyaux libérant leur trop plein d'énergie sont des ***noyaux radioactifs***.

Au moment de la désintégration, un noyau peut libérer différents types de rayonnement qui correspondent à une forme de radioactivité :

- La ***radioactivité alpha*** se traduit par l'émission d'un petit noyau composé de deux protons et de deux neutrons, appelé ***particule alpha***.
- La ***radioactivité bêta*** correspond à la transformation dans le noyau :
  - D'un neutron en un proton
  - D'un proton en neutron, caractérisé par l'envoie d'un antiélectron (même masse que l'électron, sauf qu'il a une charge positive)
- la ***radioactivité gamma*** se traduit par l'émission d'une onde électromagnétique de haute énergie sans émissions de particules.

Qu'est ce que la notion d'isotope ?

Il existe plusieurs combinaisons possibles de protons et de neutrons qui permettent d'assurer la stabilité du noyau de l'atome.

Exemple : *l'oxygène que l'on respire se présente sous trois formes différentes*

- plus de 99 % des noyaux d'atomes d'oxygène comptent 8 protons et 8 neutrons
- moins d'une fois sur cent, les noyaux d'oxygène comptent 8 protons et, soit 9 soit 10 neutrons

Ces trois combinaisons constituent les trois isotopes stables de l'oxygène. Toutes les autres combinaisons conduisent à un isotope instable de l'oxygène, c'est-à-dire à un noyau d'oxygène radioactif, qui est tellement instable qu'il se désintègre instantanément.

Qu'est qu'un noyau exotique ?

C'est un noyau qui n'existe pas à l'état naturel sur terre. Il est crée par exemple lors de collisions d'un faisceau de particule vers une cible composé d'un type d'atome, dépendant uniquement de ce que l'on veut créer. Cela est expliqué plus en détails dans la suite de ce document.

Qu'est qu'un plasma ?

Quatrième état de la matière, un plasma est un gaz ionisé électriquement neutre, composé de molécules ionisées, d'ions positifs et d'électrons.

Après avoir expliqué l'atome et ses différents états, il est intéressant de savoir quel type de recherche est-il possible d'effectuer sur ces atomes.

Tables des figures

Lexique

Le scrapper : Il a pour rôle d'arrêter le faisceau inutile créé par le « hacheur ».

Fichier protocole : Fichier texte ou se trouve les commandes pour piloter les instruments

Fichier de substitution : Fichier permettant d'ajouter des informations variables aux bases de données

Fichier startup : Fichier qui est exécuté au démarrage de l'IOC pour charger les bases de données, les drivers, etc…

Hacheur : permet de sélectionner un bout du faisceau

Record : éléments associé à une Process variable vue par la base de données EPICS

Record soft : Record qui n'est pas associé à une sortie physique comme la sortie d'une carte VME

TOR : Entrée/Sortie tout ou rien

Références bibliographiques

Site internet :

Site d'EPICS : http://www.aps.anl.gov/epics/

Site internet du GANIL: http://www.ganil-spiral2.eu/spiral2

Site internet du CEA : http://irfu.cea.fr/Phocea/Vie_des_labos/Ast/ast_technique.php?id_ast=794

Site internet de l'IPHC : http://www.iphc.cnrs.fr/-SPIRAL-2-.html

Document :

- Mémoire de maitrise par M. Yann JACOB (2001) : « ***GANIL, MATIERE A HISTOIRE (1972-2001)*** »
http://www.ganil-spiral2.eu/leganil/presentation/ganil-matiere-a-histoire/view?searchterm=th%C3%A8se

- Thèse de Laurent Maunoury (1998) : « ***Production d'ions radioactifs multichargé pour SPIRAL : Etude et réalisation du premier ensemble cible-source*** »

- Communiqué de presse du lancement de SPIRAL2 au GANIL :
http://www.in2p3.fr/presse/communiques/archives/2005/media_2005/16_inauguration_caen/spiral_2_ganil.pdf

- Compte rendu de réunion, rapport technique, etc…à propos du projet SPIRAL2 sur le site intranet du CEA