Rapport de PIDR

Projet Interdisciplinaire de Découverte à la Recherche

Système d'aide à la conduite : développement du comportement des nœuds d'un réseau de capteurs sans fil

- 26 mai 2010 -

Abderahman KRIOUILE
Bertrand STIVALET
ESIAL - 2A

Encadrants : Dragos DOBRE, Fabien CLANCHE
PIDR - 2010

Remerciement

Nous tenons tout d'abord, à remercier nos encadrants D. Dobre et F. Clanche pour toute l'aide apportée pendant la période du projet et pour nous avoir proposé ce sujet. Nous tenons aussi à remercier le laboratoire de recherche du CRAN de Nancy pour nous avoir mis à disposition le matériel nécessaire à la bonne réalisation de notre projet ainsi qu'au bon déroulement de nos expérimentations. Et pour terminer, sans qui notre projet n'aurait pas pu se dérouler, nous voulons remercier l'ESIAL qui nous à permis de réaliser ce Projet Interdisciplinaire de Découverte de la Recherche (**PIDR**) et qui a mis à notre disposition un créneau horaire hebdomadaire.

Table des matières

Remerciement **2**

Table des matières **3**

Glossaire **4**

Table des figures **4**

Introduction **5**

1. Présentation de l'environnement de travail **6**

1.1. CRAN 6

1.2. Projet CISPI 7

1.3. Projet PIDR 9

2. Recherches **9**

2.1. Technologie CrossBow 10

2.2. Outils utilisés 10

3. Développement **12**

3.1. Description 12

3.2. Analyse du problème 12

3.3. Solutions envisagées 12

3.4. Implémentation 13

3.5. Tests 13

4. Utilisation **14**

Conclusion **15**

Bibliographie **17**

Annexes **18**

Glossaire

PIDR Projet Interdisciplinaire de Découverte à la Recherche

CRAN Centre de Recherche en Automatique de Nancy

CISPI Conduite Interactive et Sûre de Procédés Industriels

AmI Intelligence Ambiante

PDA Personal Digital Assistant / Assistant numérique personnel

RFID Radio-frequency identification / Radio-identification

XML Extensible Markup Language / Langage extensible de balisage

Table des figures

Figure 1.2-1 : Diagramme cas d'utilisation pour la plate-forme CISPI

Figure 1.2-2 : Un vue d'ensemble du futur système

Figure 2.2-1 : Logiciel MoteConfig

Figure 2.2-2 : Logiciel XSniffer

Figure 3.5-1 : Interface de notre application

Figure 3.5-2 : Détection conflit

Figure 3.5-3 : Ajout dynamique de vanne

Introduction

Le développement des réseaux sans fil et des réseaux mobiles ouvre une nouvelle ère dans le domaine des télécommunications, aujourd'hui avec l'avènement du WIFI et de la téléphonie mobile, et demain avec les technologies des réseaux « Ad hoc ». L'utilisation des capteurs sans fil est de plus en plus demandée pour la supervision et la sécurité des procédés industriels. Les industries proposent alors des capteurs sans fil qui peuvent renseigner l'utilisateur sur plusieurs données. Ces capteurs peuvent aussi être reliés ensemble pour former un réseau sans fil se basant sur des protocoles pour communiquer et proposant des programmes embarqués.

Le groupe EDF désirant moderniser sa gestion des procédures de modification des conduites hydrauliques, anticipe sur les nouvelles technologies et souhaite créer un réseau de capteur sans fil intelligent pour assister et sécuriser ce processus. En effet ce nouveau système contrôlera l'état des vannes du circuit hydraulique, et sera en interaction avec l'ensemble des acteurs concernés. Il aura pour finalité d'accompagner le rondier et d'anticiper les erreurs qui pourront être potentiellement commises.

C'est en collaboration avec le CRAN et dans le cadre de notre Projet Interdisciplinaire de Découverte à la Recherche (PIDR), que nous avons participé à la réalisation d'une application capable de répondre aux exigences d'EDF. Celle-ci devait permettre à un réseau de capteur sans fil intelligent d'être autonome de participer à la sécurisation et à la modernisation de leur ancien équipement.

Dans un premier temps nous allons vous présenter le lieu et les moyens techniques mis à notre disposition pour réaliser ce projet interdisciplinaire de découverte à la recherche, puis nous vous expliquerons les recherches menées pour répondre à cette problématique, pour ensuite, visualiser le travail fourni pendant toute cette période, ainsi qu'une explication de l'application finale.

1. Présentation de l'environnement de travail

1.1. CRAN

1.1.1. Présentation

Créé en 1980, le Centre de Recherche en Automatique de Nancy (CRAN) est une unité mixte de recherche commune à plusieurs entités :

- Université Henri Poincaré, Nancy 1,
- INPL, Institut National Polytechnique de Lorraine,
- CNRS, Centre National de Recherche Scientifique,
- Centre Alexis Vautrin (CAV, Centre de lutte contre le cancer),
- CHU Nancy.

Le laboratoire compte actuellement 76 enseignants-chercheurs, 1 chercheur CNRS, 5 autres chercheurs du CAV et du CHU, 13 post-docs, 71 doctorants et 23 ingénieurs, techniciens ou administratifs (17 équivalents temps plein). Il fait partie de la Fédération de Recherche Charles Hermite Automatique, Informatique, Mathématiques de Lorraine.

Le CRAN est divisé en cinq groupes de recherche :

- Automatique : Commande et Observation des Systèmes (ACOS)
- Identification, Restauration, Images, Signaux (IRIS)
- Sureté de Fonctionnement et Diagnostic des Systèmes (SURFDIAG)
- Ingénierie pour la Santé (IPS)
- Systèmes de Production Ambiants (SYMPA)

Nos encadrants, D. Dobre travaille dans le groupe de recherche SYMPA (Système de Production Ambiants) dans l'équipe Systèmes Interopérants, tandis que et F. Clanche est ingénieur et responsable du service plates-formes expérimentales.

1.1.2. Missions du CRAN

Les recherches développées au CRAN concernent :

- l'Automatique, définie comme la science de la modélisation, de l'analyse, de la commande et de la supervision des systèmes dynamiques,
- le traitement du signal,
- le génie informatique.

Le laboratoire développe des activités transverses à ces disciplines dans les domaines de l'ingénierie pour la santé et de la sûreté de fonctionnement des systèmes. Ces domaines de recherche, fondés sur les concepts de signaux, systèmes dynamiques, information et décision, concernent à la fois les systèmes techniques (processus industriels, systèmes de transport, production d'énergie, réseaux de communication, …) et les systèmes naturels (systèmes environnementaux, santé). Les retombées de ces recherches ont un impact tant « sociétal » (amélioration de la sûreté des installations, de la santé ou de l'environnement), qu'économique (amélioration du rendement des installations, de la qualité des produits ou des services).

1.2. Projet CISPI

1.2.1. Contexte

La plate-forme CISPI s'inscrit dans le cadre du projet **LABIME du GIS Surveillance, Sûreté et Sécurité des Grands Systèmes** (partenaires CRAN, LORIA, EDF) portant sur la conduite de grands systèmes industriels à risque, et plus particulièrement des centrales de production d'énergie. La conduite de ces procédés met en jeu un ensemble de processus complexes couvrants des modes opératoires variés (en production, en arrêt, en démarrage, etc) adaptés à la criticité des modes d'exploitation rencontrés (conduite normale, conduite incidentelle et accidentelle) selon des constantes de temps différentes (conduite en temps réel, maintenance hors ligne, …).

Aujourd'hui, ces processus reposent sur des interactions entre les différents métiers des opérateurs et des systèmes propriétaires, hétérogènes et généralement limités à la phase de production normale. Dans ce contexte, l'apparition d'alarmes ou de défauts provoque souvent le basculement d'une conduite dite « normale » vers une conduite dite « accidentelle » visant essentiellement à préserver la sécurité des personnels, à conserver le système de production et à limiter les rejets sur l'environnement. La mise en œuvre d'une conduite « incidentelle », limitant ce basculement systématique en conciliant les objectifs de production et de sécurité, constitue un gisement de productivité important.

Cette conduite incidentelle mise sur la capacité du système d'information à fournir aux opérateurs les moyens leur permettant d'analyser la situation d'état dans lequel se trouve le procédé, de choisir des procédures appropriées à ces situations et permettant de produire dans des conditions de sécurité acceptable, de décliner ces procédures sous la forme de plans d'actions et enfin d'assurer le suivi et la surveillance du système de production, ce qui suppose notamment :

- la mise à disposition des opérateurs de conduite, en temps réel, d'informations synthétiques, fiables et directement interprétables émanant des algorithmes de surveillance, de détection, voire de pronostic des équipements et matériels,
- l'élaboration, à partir de ces informations, d'indicateurs relatifs à la disponibilité des systèmes et sous-systèmes dans lesquels interviennent les équipements et matériels surveillés en tenant compte de la structuration hiérarchique utilisée par les opérateurs (objets techniques de base, fonctions, unité de production),
- la coordination des différents corps de métiers en exploitation – conduite, maintenance – intervenant sur les équipements, quelque soit leur localisation – en salle de commande centralisée, sur site.

La plate-forme CISPI est composée par des acteurs techniques et des acteurs humains.
Le diagramme suivant montre les cas d'utilisation.

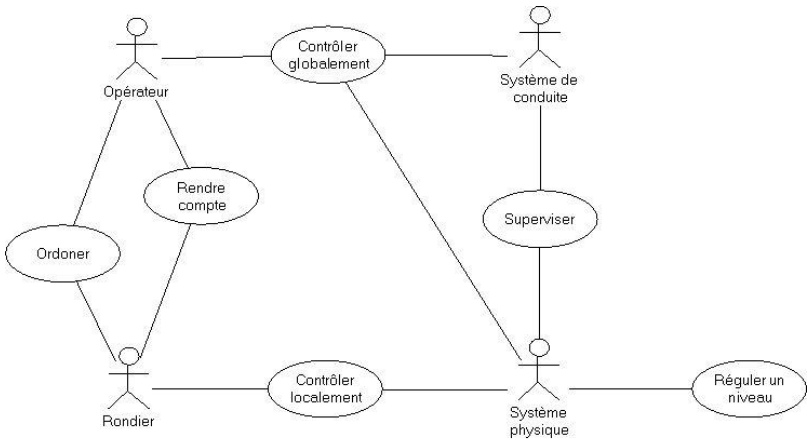


Figure 3.2-1 : Diagramme cas d'utilisation pour la plate-forme CISPI

Les acteurs humains sont les opérateurs et les rondiers. Les acteurs techniques sont le système de conduite et le système physique.

Le fonctionnement de la plate-forme est basé sur la collaboration/ synchronisation/ coordination entre l'opérateur, en salle de commande (contrôle globalement le système à l'aide du système de conduite), et le rondier, sur le terrain (contrôle localement les composants du système physique).

Le système de conduite a le rôle de superviser le bon fonctionnement du système physique qui doit réguler un niveau.

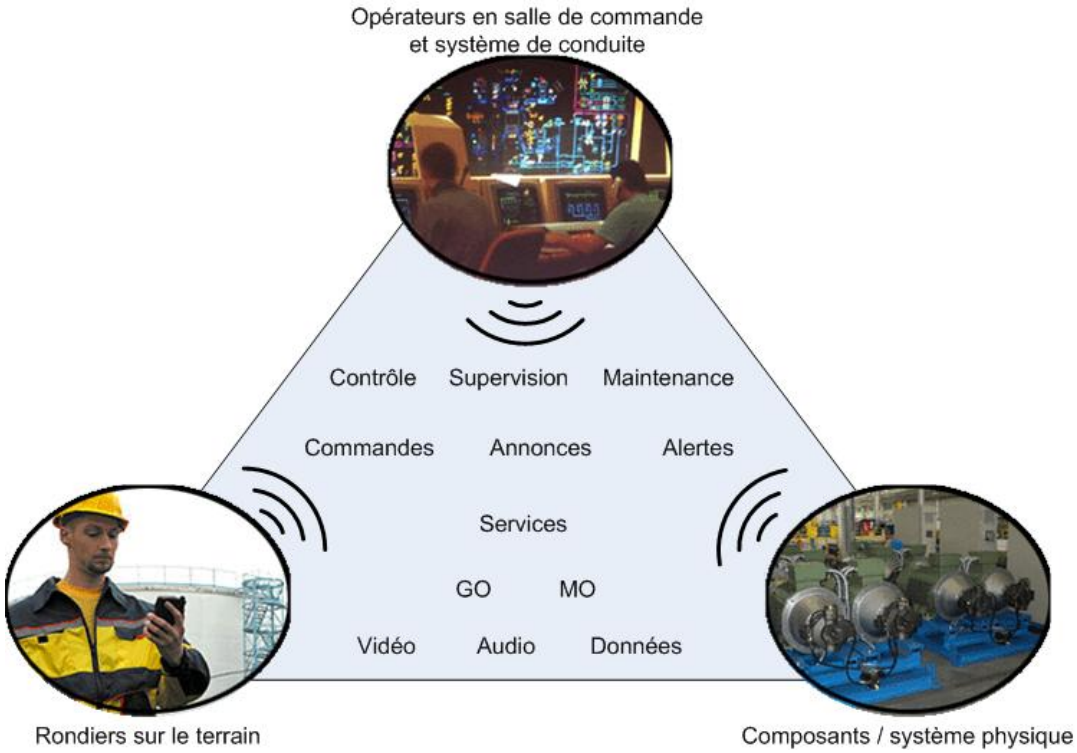


Figure 1.2-4 : Un vue d'ensemble du modèle

1.2.2. Problématique

L'objectif du projet CISPI est la mise en place de l'Intelligence Ambiante (AmI) pour le contrôle et la surveillance des systèmes complexes et donc rendre plus performant et plus sécurisé le travail du personnel (rondier). Ce travail de recherche est une étude de faisabilité de l'AmI dans les systèmes complexes et dangereux. L'étude est faite sur les centrales nucléaires d'EDF.

1.3. Projet PIDR

1.3.1. Présentation du sujet

Notre projet de PIDR à pour intitulé « *Système d'aide à la conduite : développement du comportement des nœuds d'un réseau de capteurs sans fil* ».

Pour effectuer celui-ci, le CRAN dispose d'une plateforme de laboratoire (*voir annexe I*) qui reproduit à une échelle réduite un système tel que ceux que l'on retrouve dans une centrale nucléaire. Afin d'en améliorer la conduite, l'objectif est d'implanter le concept de conduite interactive à l'aide des technologies ambiantes (Systèmes Multi Agents, RFID, PDA, etc.).
Et dans le but d'améliorer, la plate-forme est notamment équipée d'un réseau de capteurs sans fil, constitué de nœuds miniaturisés comportant des capacités de traitement, stockage et communication sans fil de l'information. L'idée est de fournir en temps-réel des données sur l'état du processus (position des vannes ouverte/fermé…).
L'objectif de ce projet est de spécifier et développer le programme embarqué dans les nœuds du réseau de capteurs sans fil (cf Annexe 2).

Pour réaliser ce projet, nous avons dû commencer par comprendre et analyser le milieu dans lequel nous allions travailler pendant ces 5 mois. Pour cela, la durée de notre projet c'est divisée en deux partie, la première correspond à toute la partie recherche qui à été menée au préalable, tandis que lors de cette deuxième partie, nous nous sommes consacrés à la partie développement du système.

2. Recherches

Nous allons voir à présent, quelles ont été les étapes qui nous ont permis de nous familiariser avec l'environnement technique et technologique du projet. C'est partie vous permet de comprendre les différents outils technologique avec lesquels nous avons travaillé, ainsi que leurs utilités.

2.1. Technologie CrossBow

L'objectif de notre projet étant d'analyser et de traiter les informations circulant dans le réseau de capteur sans fil, nous avions à disposition un kit de développement CrossBow comprenant :

- Des capteurs sans fil MicaZ,
- Des cartes d'acquisition de données MDA300,
- Des passerelles MIB520.

Pour nous aider à utiliser et comprendre ce matériel, nous nous sommes aidés des guides utilisateurs ainsi que des documents techniques fournis par le constructeur (cf annexes 3, 4 et 5).

Les capteurs étant programmables, pour modifier leurs comportements il faut implémenter un programme en nesC, qui est un langage de programmation embarqué. Pour réaliser cela, nous avons utilisé plusieurs logiciels fournis avec les capteurs et téléchargeable à partir du site internet du fabricant.

2.2. Outils utilisés

2.2.1. NotePad 2

NotePad nous a permis de nous familiariser avec le langage de programmation nesC, en effet, ce logiciel est vraiment complet puisqu'il possède de nombreux tutoriaux et exemples de programmes. Pour commencer, nous avons chargé les programmes de base dans les modules pour comprendre et analyser leurs comportements, puis nous avons nous même programmé quelques programmes simple et complet pour finaliser l'apprentissage de ce langage.

Le second avantage de ce logiciel est qu'il possède également l'intégralité des librairies de chacun des composants, ainsi nous pouvions utilisés des fonctions pré-implémentées dans nos programme, ce qui nous facilitait le travail, et nous permettait d'aller plus vite.

2.2.2. MoteConfig

Une fois le programme terminé et compilé, nous utilisions le logiciel MoteConfig pour implémenter dans le capteur l'exécutable précédemment créé.

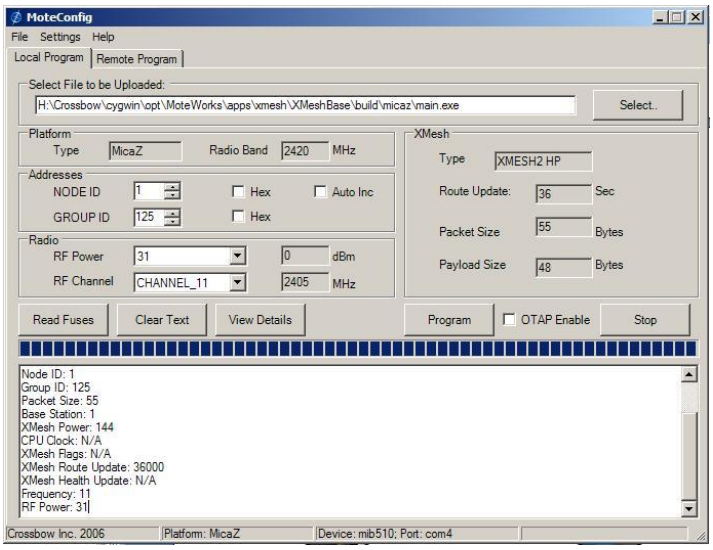


Figure 2.2-1 : Logiciel MoteConfig

Nous pouvons voir sur cette capture d'écran, que le premier champ correspond au chemin de l'exécutable à implémenter. Le champ *NODE ID* est important dans notre cas, car c'est ce paramètre qui va permettre d'identifier de manière unique les différents capteurs, nous y reviendrons lors de la présentation de notre application. Et pour finir, il suffit de cliquer sur *Program* pour charger le programme dans le capteur.

2.2.3. XSniffer

Le logiciel XSniffer, quant à lui, nous permettait d'avoir un retour graphique sur les communications entre les différents capteurs car il permet d'afficher les différents paquets échangés ainsi que la valeur de certaines mnémoniques.

Nous pouvons voir que dans l'exemple ci-après (Figure 2.3-2), grâce à un interrupteur, nous changions les valeurs des champs 17 et 19. Pour sniffer cette communication et la retranscrire sur un PC, nous avons dû paramétrer un capteur pour qu'il joue le rôle de serveur appelé Xserve et fasse l'intermédiaire entre le réseau de capteur sans fil et l'ordinateur.

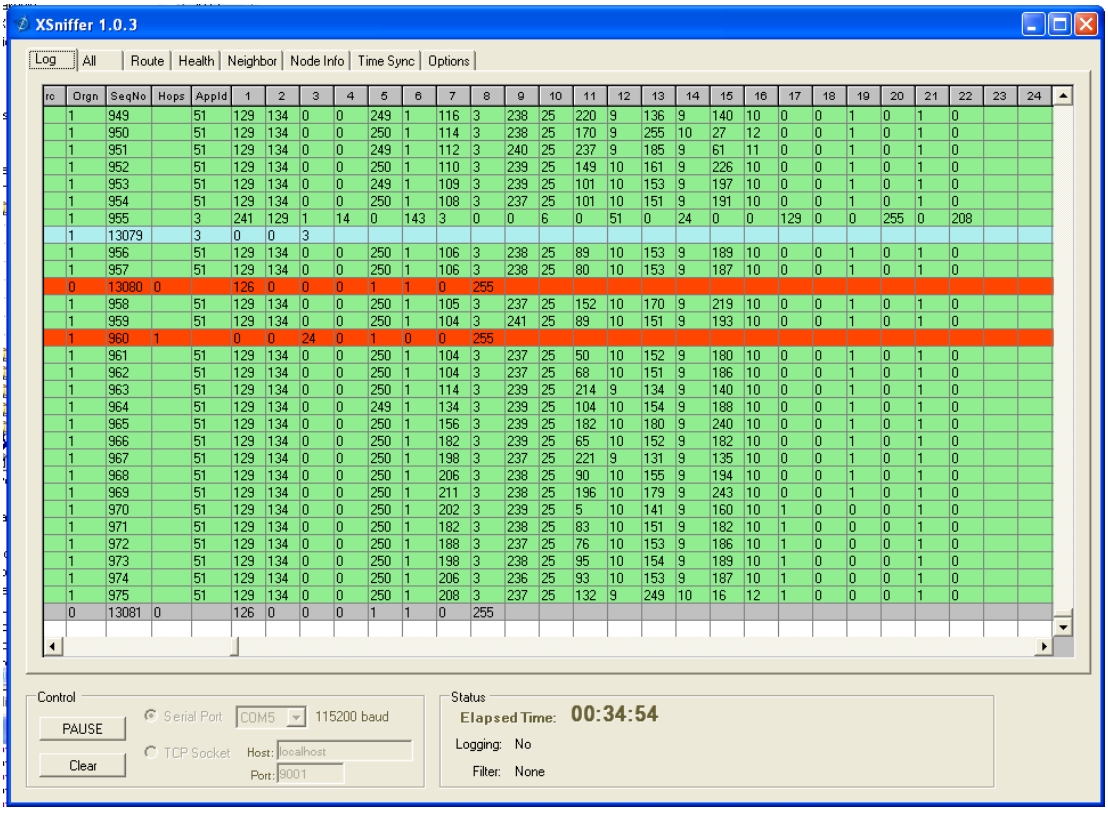


Figure 2.2-2 : Logiciel XSniffer

Ce logiciel nous a permis de simuler et de tester les différents scénarios que l'on avait déterminés au préalable lors de l'étude du cahier des charges.

Une fois les recherches terminées et les technologies comprises, nous avons pu nous consacrer à l'étude et au développement d'une application qui répondrait à la problématique du projet.

3. Développement

3.1. Description

L'objectif qui nous a été fixé est d'analyser les données échangées dans le réseau de capteur sans fil et de les traiter en fonction de certaines contraintes fournies par la physiques du procédé, puis de faire remonter ces informations au rondier et au système de supervision.
Chaque capteur représente l'état d'une vanne, il sera représenté par un état booléen en logique inversée, c'est-à-dire que la vanne sera soit Ouverte (valeur=0), soit Fermée (valeur=1). Dans l'intérêt du projet, certaines configurations ne peuvent avoir lieu, par exemple lorsque la vanne de remplissage de la cuve est ouverte, il ne faut pas que la vanne de vidage de cette cuve soit elle aussi ouverte, sinon l'effet de remplissage n'aura aucun intérêt. C'est dans cette logique que le CRAN souhaite développer ce projet, mais à des fins de sécurisation de son système. En effet, les enjeux d'une centrale nucléaire en cas d'erreur de manipulation sont très importants, c'est pour cela que le rondier sera assisté par un PDA (Personal Digital Assistant), qui lui indiquera en fonction de l'identification et l'état de la vanne, si celle-ci doit être modifiée pour répondre à la procédure en cours.

3.2. Analyse du problème

La finalité du projet de PIDR est de traduire la description précédente en langage informatique. Il devra alors répondre au cahier des charges et permettre aux utilisateurs, donc aux personnels d'EDF, de sécuriser leurs exécutions de procédures. Notre programme devra se comporter en système temps réel, la réactivité de notre application devra être la plus rapide possible, et devra être interopérable entre les différents équipements présents sur la plate forme.

3.3. Solutions envisagées

Pour répondre à ces problèmes, nous avons étudié deux possibilités. La première était de récupérer les valeurs des différentes trames des capteurs sous forme XML (Extensible Markup Language) fournie par Xserver que nous devions ensuite analyser. Ce langage XML permet d'envoyer des données structurées, donc plus simple à analyser. Tandis que la deuxième solution était d'analyser directement les trames envoyées en temps réel, qui sont caractérisées par une suite de bits et envoyés les unes après les autres.

La première solution aurait été la plus facile, mais nous avons préféré développer une application correspondante à la deuxième possibilité car au finale, notre application doit être implémentable dans les capteurs, ce qui engendre une programmation de bas niveau en nesC. Ce langage de programmation étant un dérivé du langage C, il permet d'inclure à l'intérieur des bouts de code en C, mais il aurait été très difficile d'y implémenter un parseur (analyseur syntaxique destiné à récupérer les informations contenues dans le XML) dans celui-ci.

3.4. Implémentation

Pour réaliser cette deuxième solution, nous avons réalisé dans un premier temps un script développé en batch qui permet de se connecter au serveur et qui récupère les informations issues des capteurs connectés à celui-ci. Il permet également de fournir au programme principal (développé en C) une trame formatée qui pourra être facilement interprété par le programme pour une meilleure rapidité d'exécution (rappelons que nous travaillons en temps réel et que le traitement doit s'effectuer le plus rapidement possible).

Une fois la trame récupérée, elle est analysé par le programme principal, la trame fournie est de la forme :

```
11,2,0,125,51,129,134,2859,35.049137,25.2901,190.6648,142.2179,2548,1,1,1
```

Seuls les chans de couleur bleu vont nous intéresser, les autres chans, séparés par des virgules, correspondent à d'autres informations envoyées par le capteur qui ne nous sont pas utiles pour notre projet (température, humidité …).
La première information, ici 11, correspond au Channel, chaîne d'émission du signal. Cette valeur, indiquée lors du paramétrage du capteur, nous permet d'être renseignée sur le type de la trame, car certaines trames sont utilisées à d'autres fins qui ne sont pas nécessaire à notre traitement de données (diffusion, recherche d'autres capteurs...).
La deuxième information, ici 2, correspond au numéro d'identification du capteur, c'est le numéro de paramètre que l'on indique lorsque l'on programme le capteur, cette valeur est très important car c'est celui-ci qui va permettre d'identifier les capteurs entre eux, et de leurs ajoutés des exclusions mutuelles en fonction de l'environnement. Tandis que la troisième information utile, ici 1, correspond à la valeur recherchée, donc l'état de la vanne. Nous pouvons voir que dans notre exemple la vanne numéro 2 est fermée.

Une fois les informations relatives aux capteurs prélevés, nous effectuons des testes par rapport aux contraintes entrées par l'utilisateur, pour visualiser si des états ne sont pas en contradiction avec d'autres. Bien évidement l'utilisateur peut visionner en temps réel l'état des vannes (donc des capteurs) grâce à l'interface graphique que l'on lui propose.
Il peut également paramétrer les exclusions mutuelles en informant les numéros des vannes ainsi que leurs états. Si une exclusion est détectée, alors notre application va afficher un message d'alerte indiquant un problème entre les deux vannes concernées.

Pour garantir la sécurité et garder une trace des changements d'états, nous avons également créé un fichier log qui regroupe l'ensemble des événements survenus à travers le temps pour chaque vanne. Chaque modification d'état d'une vanne sera sauvegardée dans un fichier avec la date et l'heure à laquelle est survenue cette modification.

3.5. Tests

Pour tester notre application, nous avons effectué une batterie de tests grâce à l'équipement qui nous à été fourni. Celui-ci était composé d'un ordinateur (pour programmer les différents modules, et lancer notre application), et de trois capteurs MicaZ (deux capteurs avec interrupteur jouant le rôle de la vanne, et un capteur jouant le rôle de serveur pour récupérer les données des deux précédents).

Nous pouvons voir dans cet exemple, le fonctionnement normal de notre application. La vanne 1 est ouverte tandis que la vanne 2 est fermée (logique utilisée dans le projet CISPI).

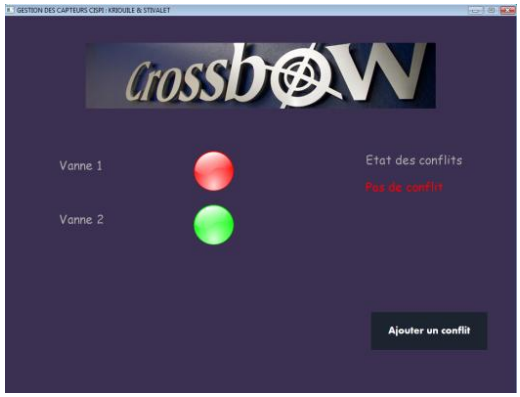


Figure 3.5-1 : Interface de notre application

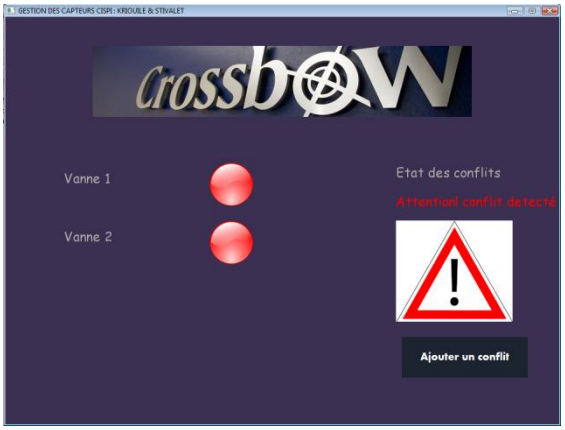
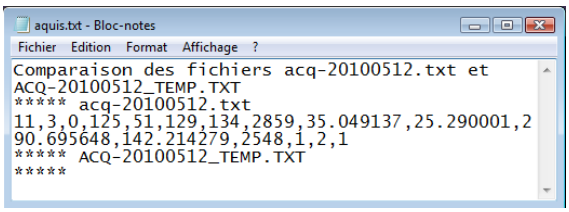


Figure 3.5-2 : Détection conflit

Dans ce deuxième exemple, notre application détecte un conflit entre deux états de vannes. Un message d'erreur s'affiche alors à l'écran.

Figure 3.5-3 : Ajout dynamique de vanne



L'application ajoute dynamiques les vannes à l'interface graphique lorsque celle-ci reçoit une trame comportant l'identificateur d'une vanne inconnu. L'état inconnu (jaune) est affiché quand l'application ne reçoit une valeur différente des valeurs attendues (Ouverte ou Fermée).

4. Utilisation

Lors de nos différents tests, nous avons utilisé un capteur pour jouer le rôle du serveur. Nous avons donc programmé un module avec le programme contenu dans *C:\CrossBow\Cygwin\opt\MoteWorks*\apps\xmesh\XMeshBase\MeshBase.nc
Tandis que les autres capteurs étaient programmés avec
C:\CrossBow\Cygwin\opt\MoteWorks\apps\xmesh\XMDA300\XMDA300.nc

Dans un premier temps pour utiliser notre application il faut se connecter à Xserve, pour cela rendez vous dans le répertoire correspondant avec l'invité de commande :

> cd C:\CrossBow\Cygwin\opt\MoteWorks\tools\xserve\bin\bin

Ensuite exécuter le fichier **serveur.bat** qui va nous permettre de se connecter à Xserve, et générer un fichier avec les données du capteur il doit contenir

```
xserve -device=com5 -xc > acq-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%.txt
echo.
pause
```

Une fois que ces données sont envoyées à l'ordinateur, nous allons les analyser et les formater grâce au second fichier **programme.bat** qui va lancer l'interface graphique et permettre d'analyser en temps réel les données fournies par les capteurs.

```
@echo off
@copy acq-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%.txt acq-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%_temp.txt
trame.exe
:bouclage
fc /a acq-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%.txt acq-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%_temp.txt > comp-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%.txt
if %errorlevel%== 1 (
REM echo Fichiers differents
REM @copy comp-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%.txt entreeProg.txt
@copy acq-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%.txt acq-%Date:~-4%%Date:~-7,-5%%Date:~-10,-8%_temp.txt
)
goto bouclage
echo.
pause
```

Maintenant que le programme est lancé, nous pouvons ajouter des conflits entre les différents capteurs grâce à l'interface graphique.

Conclusion

Notre application à permis d'ajouter des fonctionnalités au projet CISPI, puisqu'il est dorénavant capable de superviser à distance l'état des différentes vannes, d'ajouter des contraintes entre celles-ci pour respecter la physique du procédé et de sécuriser les conduites de procédés.

Le travail que nous avons effectué ouvre d'autres perspectives à l'amélioration de notre projet, puisqu'il serait fort intéressant pour le projet CISPI d'intégrer aux capteurs sans fils une partie de notre application pour les rendre autonomes.

Le PIDR nous a permis de découvrir le milieu de la recherche en France qui était alors inconnu pour nous. Nous avons pu faire la comparaison entre le domaine industriel et le monde de la recherche grâce à nos expériences passées, ceux-ci sont très différents même si les attentes sont les mêmes. Ce projet nous à permis d'acquérir de l'indépendance dans notre travail car nous avons eu une grande liberté dans le fonctionnement de notre travail.

D'un point de vue pédagogique, nous avons pu acquérir des connaissances dans le domaine des technologies embarquées qui, au vue leurs succès dans le monde industriels d'aujourd'hui, nous seront surement utiles pour l'avenir.

Bibliographie

Intranet du CRAN

Site comportant toute la documentation utile pour la description du sujet.
(http://plates-formes.cran.uhp-nancy.fr)

Technologie CrossBow

Site du constructeur de matériel, avec datasheet.
(http:// www.xbow.com)

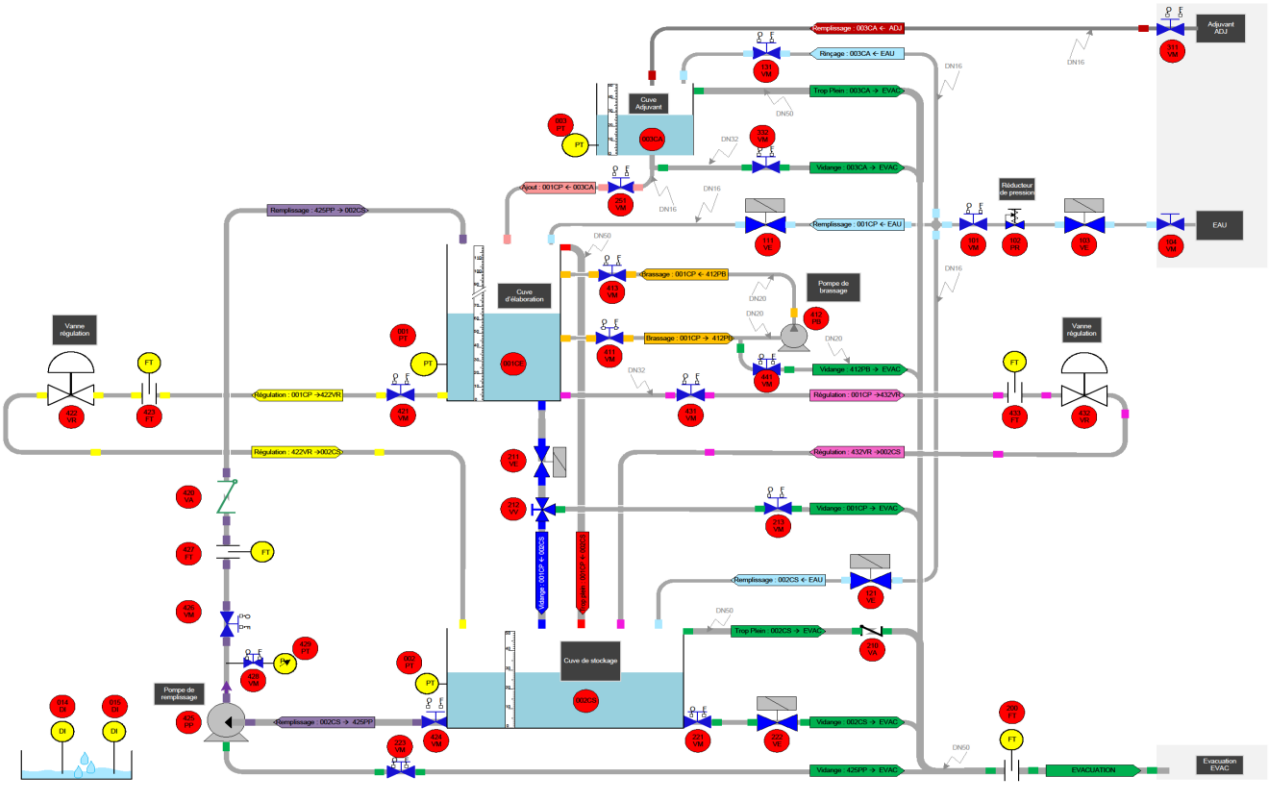
Cours sur le langage nesC

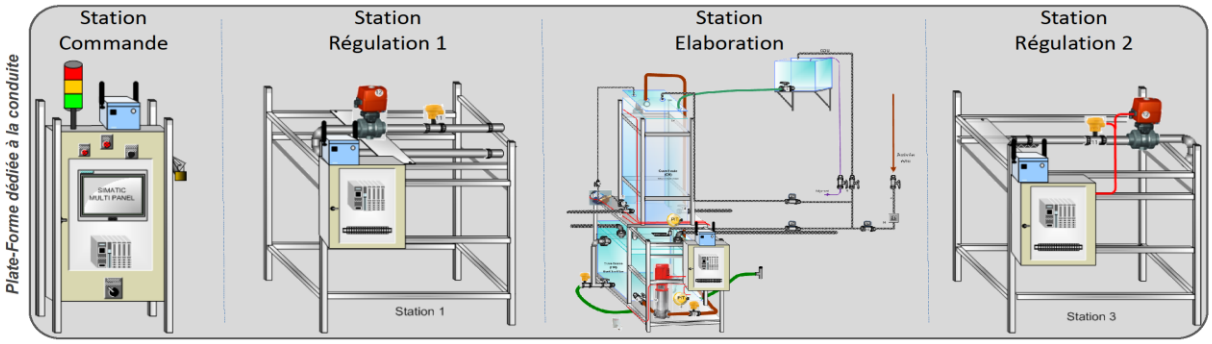
David Gay, Philip Levis, David Culler & Eric Brewer, *nesC 1.1 Language Reference Manual*, May 2003.

Annexes

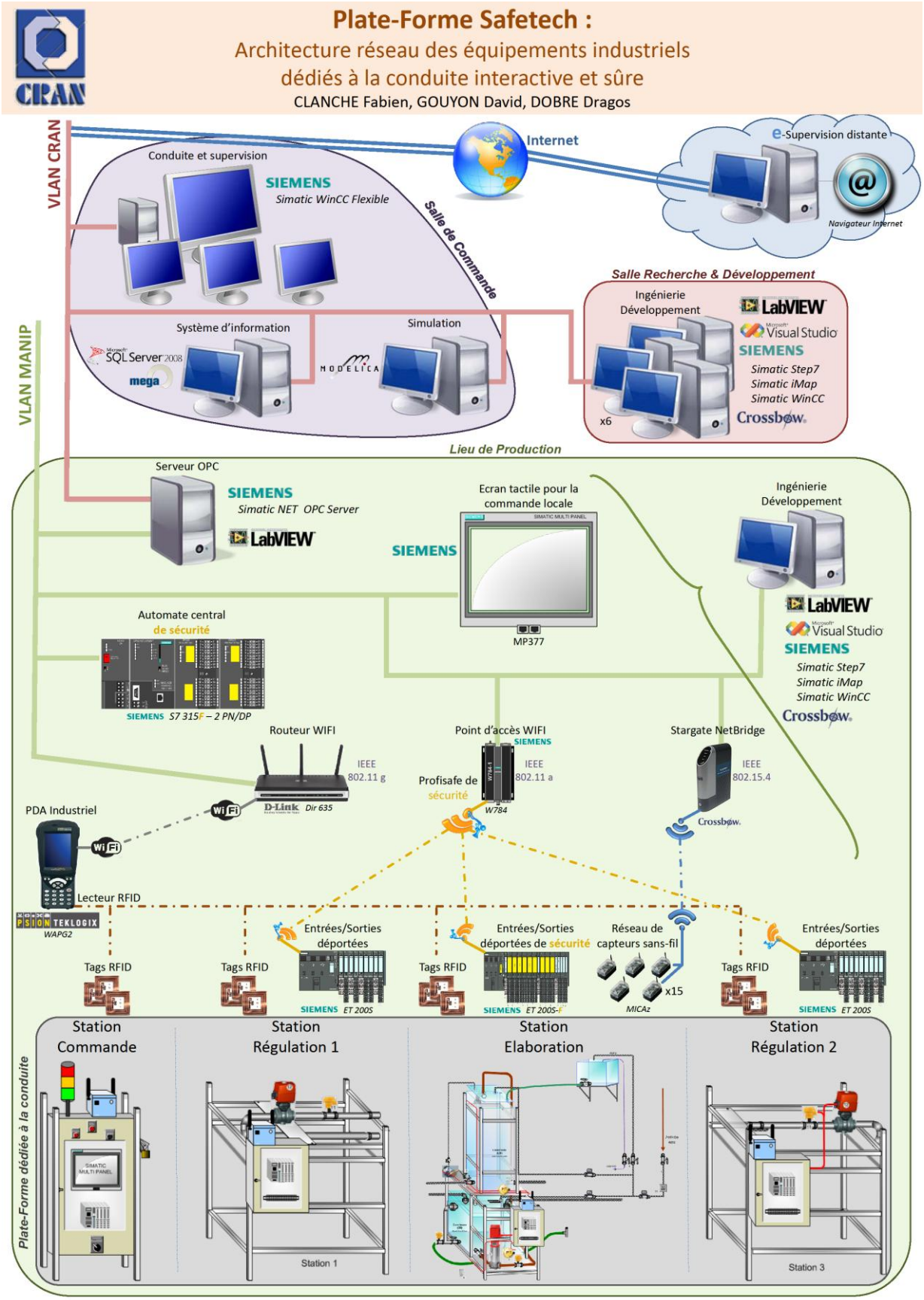
Annexe I: Plate-Forme de simulation CRAN

Plate-Forme Safetech :
Plan de circulation des fluides du démonstrateur
dédié à la conduite interactive et sûre





Annexe 2 : Réseau



Annexe 3 : Datasheet MicaZ

Crossbow

MICAz

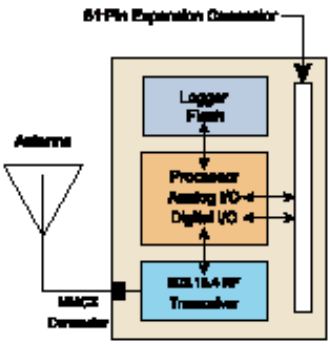
WIRELESS MEASUREMENT SYSTEM

- 2.4 GHz IEEE 802.15.4, Tiny Wireless Measurement System
- Designed Specifically for Deeply Embedded Sensor Networks
- 250 kbps, High Data Rate Radio
- Wireless Communications with Every Node as Router Capability
- Expansion Connector for Light, Temperature, RH, Barometric Pressure, Acceleration/Seismic, Acoustic, Magnetic and other Crossbow Sensor Boards

Applications

- Indoor Building Monitoring and Security
- Acoustic, Video, Vibration and Other High Speed Sensor Data
- Large Scale Sensor Networks (1000+ Points)

ZigBee Alliance



MPR2400 Block Diagram

MICAz

The MICAz is a 2.4 GHz Mote module used for enabling low-power, wireless sensor networks.

Product features include:

- IEEE 802.15.4 compliant RF transceiver
- 2.4 to 2.48 GHz, a globally compatible ISM band
- Direct sequence spread spectrum radio which is resistant to RF interference and provides inherent data security
- 250 kbps data rate
- Supported by MoteWorks™ wireless sensor network platform for reliable, ad-hoc mesh networking
- Plug and play with Crossbow's sensor boards, data acquisition boards, gateways, and software

MoteWorks™ enables the development of custom sensor applications and is specifically optimized for low-power, battery-operated networks. MoteWorks is based on the open-source TinyOS operating system and provides reliable, ad-hoc mesh networking, over-the-air-programming capabilities, cross development tools, server middleware for enterprise network integration and client user interface for analysis and a configuration.

Processor & Radio Platform (MPR2400CA)

The MPR2400 is based on the Atmel ATmega128L. The ATmega128L is a low-power microcontroller which runs MoteWorks from its internal flash memory. A single processor board (MPR2400) can be configured to run your sensor application/ processing and the network/radio communications stack simultaneously. The 51-pin expansion connector supports Analog Inputs, Digital I/O, I2C, SPI and UART interfaces. These interfaces make it easy to connect to a wide variety of external peripherals. The MICAz (MPR2400) IEEE 802.15.4 radio offers both high speed (250 kbps) and hardware security (AES-128).

Sensor Boards

Crossbow offers a variety of sensor and data acquisition boards for the MICAz Mote. All of these boards connect to the MICAz via the standard 51-pin expansion connector. Custom sensor and data acquisition boards are also available. Please contact Crossbow for additional information.

Document Part Number: 6020-0060-04 Rev A

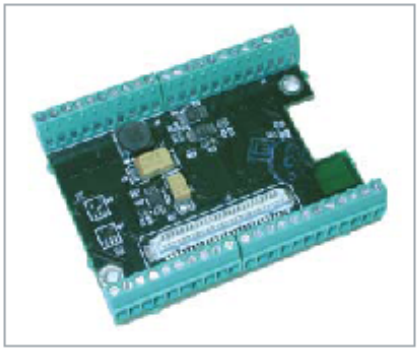
Phone: 408.965.3300 ● Fax: 408.324.4840 ● E-mail: info@xbow.com ● Web: www.xbow.com

Annexe 4 : Datasheet MDA300

Crossbow

MDA300

DATA ACQUISITION BOARD



- Multi-Function Data Acquisition Board with Temp, Humidity Sensor
- Compatible with MoteView Driver Support
- Up to 11 Channels of 12-bit Analog Input
- Onboard Sensor Excitation and High-Speed Counter
- Convenient Micro-Terminal Screw Connections

Applications

- Environmental Data Collection
- Agricultural and Habitat Monitoring
- Viticulture and Nursery Management
- HVAC Instrumentation and Control
- General Data Collection and Logging

MDA300

Developed at UCLA's Center for Embedded Network Sensing (CENS), the MDA300 is an extremely versatile data acquisition board that also includes an onboard temperature/ humidity sensor. With its multi-function direct user interface, the MDA300 offers a convenient and flexible solution to those sensor modalities commonly found in areas such as environmental and habitat monitoring as well as many other custom sensing applications.

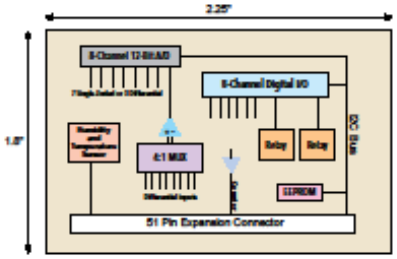
As part of a standard mesh network of Motes, the MDA300's easy access micro-terminals also make it an economical solution for a variety of applications and a key component in the next generation of low-cost wireless weather stations. Data logging and display is supported via Crossbow's MoteView user interface.

Crossbow's MoteView software is designed to be the primary interface between a user and a deployed network of wireless sensors. MoteView provides an intuitive user interface to database management along with sensor data visualization and analysis tools. Sensor data can be logged to a database residing on a host PC, or to a database running autonomously on a Stargate gateway.

Communication and Control Features Including:

- 7 single-ended or 3 differential ADC channels
- 4 precise differential ADC channels
- 6 digital I/O channels with event detection interrupt
- 2.5, 3.3, 5V sensor excitation and low-power mode
- 64K EEPROM for onboard sensor calibration data
- 2 relay channels, one normally open and one normally closed
- 200 Hz counter channel for wind speed, pulse frequencies
- External I2C interface

Drivers for the MDA300 board are included in Crossbow's MoteWorks™ software platform. MoteWorks enables the development of custom sensor applications and is specifically optimized for low-power, battery-operated networks. MoteWorks is based on the open-source TinyOS operating system and provides reliable, ad-hoc mesh networking, over-the-air-programming capabilities, cross development tools, server middleware for enterprise network integration and client user interface for analysis and configuration.



MDA300C Block Diagram

Ordering Information

Model	Description
MDA300CA	Mote Data Acquisition Board with Temperature and Humidity

Document Part Number: 6020-0052-03 Rev A

Phone: 408.965.3300 ● Fax: 408.324.4840 ● E-mail: Info@xbow.com ● Web: www.xbow.com

Annexe 5 : Datasheet MIB520

Crossbow

MIB520

USB INTERFACE BOARD

- Base Station for Wireless Sensor Networks
- USB Port Programming for IRIS/MICAz/MICA2 Hardware Platforms
- Supports JTAG code debugging
- USB Bus Power

Applications

- USB Interface
- Testbed Deployments
- In-System Programming

MIB520CB with attached Mote

MIB520CB

The MIB520CB provides USB connectivity to the IRIS and MICA family of Motes for communication and in-system programming. Any IRIS/MICAz/MICA2 node can function as a base station when mated to the MIB520CB USB interface board. In addition to data transfer, the MIB520CB also provides a USB programming interface.

The MIB520CB offers two separate ports: one dedicated to in-system Mote programming and a second for data communication over USB.The MIB520CB has an on-board processor that programs Mote Processor Radio Boards. USB Bus power eliminates the need for an external power source.

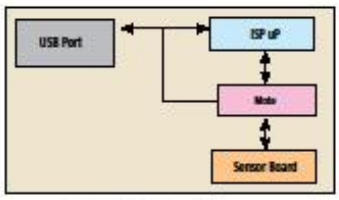
Specifications

USB Interface
- Baud Rate: 57.6 K
- Male to Female USB cable (included with unit)

Mote Interface
- Connectors:
  - 51-pin
- Indicators:
  - Mote LED's: Red Green, Yellow

Programming Interface
- Indicators:
  - LEDs - Power Ok (Green), Programming in Progress (Red)
- Switch to reset the programming processor and Mote.

Jtag Interface
- Connector: 10-pin male header
- POWER
- USB Bus powered



MIB520CB Block Diagram

Ordering Information

Model	Description
MIB520CB	USB PC Interface Board

Document Part Number: 6020-0091-03 Rev A

Phone: 408.965.3300 ● Fax: 408.324.4840 ● E-mail: info@xbow.com ● Web: www.xbow.com