

**Département Génie Électrique**

**Projet de Fin d’Etudes**

Présenté par

**Abdelbacet MHAMDI**

Pour obtenir le

**Diplôme National d’Ingénieur en Génie Electrique**

Spécialité

**Automatique et Conception des Systèmes**

Sujet du projet

**Travaux Pratiques d’Electronique à Distance : e-TP**

Organisme d’accueil : © **ESPRIT** - Ecole Supérieure Privée d'Ingénierie et de Technologies

Présenté et soutenu à Tunis le 02 Juin 2010 devant le jury composé de :

Mme. Moufida KSOURI Présidente

MM. Samir SAKRANI Rapporteur

Mekki KSOURI Encadreur ENIT

Lamjed BEN TAYEB Encadreur ESPRIT

Année : **2009-2010** Réf : PFE…………../GE-10

**« Dis-moi et j’oublie**

**Enseigne-moi et je me souviens**

**Fais-moi participer et j’apprends »**

Benjamin Franklin

# 

# Dédicaces

Aux membres de ma chère famille,

Que nulle dédicace ne puisse exprimer ce que je leur dois, pour leur bienveillance, leur affection et leur soutien… Trésors de bonté, de générosité et de tendresse, en témoignage de mon profond amour et mes sincères reconnaissances pour les efforts qu’ils ont consenti pour l’accomplissement de mes études.

# 

# Remerciements

Je voudrais remercier mes encadreurs M. Mekki KSOURI et M. Lamjed BEN TAYEB, pour leur soutien et leurs recommandations judicieuses, qu'ils trouvent ici le témoignage de mon respect et gratitude. Je tiens à remercier également, tout personnel de l'école

ESPRIT pour leur disponibilité.

Je voudrais remercier également, le préparateur Aladin AJMI,

Aucune expression ne pourra exprimer ma gratitude envers lui pour son aide et son soutien moral durant l’élaboration du projet de fin d’étude.

Nous devons chaque bribe de notre connaissance à nos enseignants de l’ENIT qui ont si bien mené leur noble quête d’enseigner les bases des systèmes électriques et de l'automatique. Nous les remercions non seulement pour le savoir qu’ils nous ont transmis, mais aussi pour la fierté et l’ambition que leurs personnes nous aspirent.

Table des matières

[**Liste des tableaux 8**](#_Toc262490978)

[**Liste des figures 9**](#_Toc262490979)

[**Introduction générale 10**](#_Toc262490980)

[**Chapitre I:**](#_Toc262490981)[**Etat de l’art de**](#_Toc262490982) [**e-learning et cadre du projet 13**](#_Toc262490983)

[1.1. Introduction 14](#_Toc262490984)

[1.2. Cadre du projet 14](#_Toc262490985)

[1.2.1. Contexte 14](#_Toc262490986)

[1.2.2. Cahier des charges 15](#_Toc262490987)

[1.3. Etat de l’art 16](#_Toc262490988)

[1.4. Conclusion 16](#_Toc262490989)

[**Chapitre II :**](#_Toc262490990) [**Généralités sur les réseaux industriels 17**](#_Toc262490991)

[2.1. Introduction 18](#_Toc262490992)

[2.2. Type des interfaces PC-Système Numérique 18](#_Toc262490993)

[2.2.1. La liaison série 18](#_Toc262490995)

[2.2.2. La liaison GPIB 22](#_Toc262490996)

[2.3. Conclusion 26](#_Toc262491001)

[**Chapitre III :**](#_Toc262491002) [**Commande des instruments à travers une interface logicielle 27**](#_Toc262491003)

[3.1. Introduction 28](#_Toc262491004)

[3.2. Commande des instruments à partir des utilitaires logicielles 28](#_Toc262491005)

[3.3. Commande des instruments à partir de MATLAB 30](#_Toc262491006)

[3.3.1. La fenêtre de commande : Command Window 30](#_Toc262491007)

[3.3.2. La boite à outil : TMTool 31](#_Toc262491008)

[3.3.3. L’environnement de développement graphique : GUIDE 33](#_Toc262491009)

[3.4. Outil de déploiement 37](#_Toc262491010)

[3.5. Conclusion 42](#_Toc262491021)

[**Chapitre IV :**](#_Toc262491022) [**Commande d'un oscilloscope via internet 43**](#_Toc262491023)

[4.1. Introduction 44](#_Toc262491024)

[4.2. Commande via Internet à partir des applications dédiées au contrôle distant 44](#_Toc262491025)

[4.2.1. Connexion par le Web au Bureau à distance 45](#_Toc262491026)

[4.2.2. Connexion par des applications dédiées au contrôle à distance 45](#_Toc262491027)

[4.3. Commande à partir des sessions MATLAB en utilisant le protocole TCP/IP 46](#_Toc262491028)

[4.4. Conclusion 56](#_Toc262491029)

[**Conclusion générale 57**](#_Toc262491030)

[**Glossaire 59**](#_Toc262491031)

[**Références bibliographiques 60**](#_Toc262491043)

[**Netographie 61**](#_Toc262491048)

[**Annexe A 62**](#_Toc262491049)

[**Annexe B 65**](#_Toc262491051)

# Liste des tableaux

[Tableau 1: Connecteur 9 pins de la liaison série RS232 [10] 20](#_Toc262606376)

[Tableau 2: Brochage d’un connecteur de type RIBBON [12] 24](#_Toc262606377)

[Tableau 3: Spécifications Techniques d’un oscilloscope Tektronix TDS 1002 63](#_Toc262606378)

[Tableau 4: Accessoires recommandés 64](#_Toc262606379)

[Tableau 5: Tableau récapitulatif des interfaces Entrées-Sorties disponibles 64](#_Toc262606380)

# Liste des figures

[Figure 1: Vue en perspective d’un connecteur série DB9 (Connecteur male) 20](#_Toc262606399)

[Figure 2: Structure de l'UART 21](#_Toc262606400)

[Figure 3: Hiérarchie des appareils dans un bus GPIB 23](#_Toc262606401)

[Figure 4: Dimensions mécanique d’un connecteur de type RIBBON 24](#_Toc262606402)

[Figure 5: Chronométrage de transfert de données dans un bus GPIB [13] 25](#_Toc262606403)

[Figure 6: Communication série sous LabView 29](#_Toc262606404)

[Figure 7: Utilisation de TMTool pour la commande de l’oscilloscope 32](#_Toc262606405)

[Figure 8: Hiérarchie des objets graphiques sous MATLAB [16] 33](#_Toc262606406)

[Figure 9: Aperçu de la mise en page graphique de l’oscilloscope 34](#_Toc262606407)

[Figure 10: Validation expérimentale de l’interface graphique de l’oscilloscope 36](#_Toc262606408)

[Figure 11: Avertissement : Acquisition d’images 36](#_Toc262606409)

[Figure 12: Avertissement : Contrôle d’instruments 37](#_Toc262606410)

[Figure 13: Spécification du compilateur 37](#_Toc262606411)

[Figure 14: Architecture de la compilation sous MATLAB [17] 38](#_Toc262606412)

[Figure 15: Assistant Graphique de déploiement 39](#_Toc262606413)

[Figure 16: Icone associé à l'exécutable 39](#_Toc262606414)

[Figure 17: InstrumentIcon.ico 40](#_Toc262606415)

[Figure 18: Setup de l’exécutable 42](#_Toc262606416)

[Figure 19: Raccourci vers l’exécutable Instrument Control Application 42](#_Toc262606417)

[Figure 20: Contrôle des bureaux à distance 44](#_Toc262606418)

[Figure 21: Connexion par le Web au bureau à distance (TSWeb) 45](#_Toc262606419)

[Figure 22: Utilitaires de contrôle à distance 46](#_Toc262606420)

[Figure 23: Contrôle des instruments via Internet 48](#_Toc262606421)

[Figure 24: Logique de négociation entre les deux sessions de MATLAB 50](#_Toc262606422)

[Figure 25: Organigramme de gestion d'ordres au niveau PC hôte 52](#_Toc262606423)

[Figure 26: Principe du test expérimental 53](#_Toc262606424)

[Figure 27: Extraction des données du canal 1 de l’oscilloscope 54](#_Toc262606425)

[Figure 28: Evolution temporelle de deux signaux extraits de l’oscilloscope 55](#_Toc262606426)

[Figure 29: Image de l’oscilloscope Tektronix TDS 1002 62](#_Toc262606427)

[Figure 30: Documentation Technique de l'adaptateur KUSB-488A 65](#_Toc262606428)

[Figure 31: Tableau comparatif de différentes connexions 66](#_Toc262606429)

[Figure 32: Interface de commande via GPIB bus 66](#_Toc262606430)

INTRODUCTION GENERALE

L'informatique grâce aux réseaux de communication, s’impose depuis des décennies, les sujets couverts sont de plus en plus vastes, les contenus s’enrichissent de multimédias et de liens hypertextes qui renvoient les apprenants connectés au réseau vers des sites spécifiques.

Le e-Learning est un moyen d'apprentissage. Il s'appuie sur internet et les outils multimédias pour offrir des programmes de formation suffisamment souples qui s’adaptent aux besoins des apprenants. [ARP00] [SAA01]

Les enseignements à distance qui se développent de plus en plus ne fournissent pas toujours des travaux pratiques nécessaires pour une meilleure assimilation du contenu de la formation.

Les formations d'ingénieurs doivent intégrer les deux aspects: théorie et pratique. Les étudiants doivent impérativement, durant leur cursus universitaire, interagir avec l’environnement lié à leur formation.

La simulation est une méthode permettant de compléter une formation théorique, toutefois, elle ne peut en aucun cas remplacer les expériences réelles puisqu'elle est uniquement liée à la modélisation.

Les Travaux Pratiques appliqués sur un monde réel améliorent la motivation de l'étudiant et développent chez lui une approche scientifique lors de la résolution des problèmes réalistes. L'expérimentation a l'avantage de rendre l'utilisateur sensible aux résultats issus des phénomènes difficiles à imaginer ou impossible à modéliser. [BEN07]

Toutefois, des difficultés d’origines organisationnel (mauvaise planification…) et technique (équipement chers…) empêchent l'étudiant de mettre en œuvre des valeurs ajoutées au niveau du contenu de la manipulation et donc d'apporter des réponses aux cahiers des charges.

La mise en place de l’e-TP se heurte à ces problèmes habituellement rencontrés dans la formation. [LEL03]

Esprit a proposé un projet dont le but est de faire un support pratique de la formation se déroulant au sein de l’école. Ceci permettra de réaliser une plateforme qui comprend des contenus pédagogiques aussi bien théoriques que pratiques.

Les technologies liées à Internet sont assez évoluées pour permettre la création des nouveaux horizons d’apprentissage en ligne. Mais le concept de faire des manipulations pratiques en rapport avec l’enseignement théorique est relativement peu développé. Pourtant ce type formation est indispensable dans les disciplines scientifiques et techniques.

Le cahier des charges initial de notre projet visait à créer le noyau de base d’une plateforme de TP d’électronique à distance, dans laquelle l’étudiant conservera une liberté de pilotage de l’instrument et de mesure sur le processus. L’idée principale consiste à utiliser le réseau d’internet comme un support de communication et MATLAB comme une infrastructure d'échange de données et un outil mathématique permettant de traiter les mesures acquises.

Le présent rapport est subdivisé en quatre chapitres. Le premier sera consacré à la présentation du cadre du projet.

Le second chapitre introduit les éléments de base d’une connexion PC-Instrument tel que la liaison série et le bus GPIB.

Le troisième chapitre expose les intérêts pratiques de ces types d’interfaçages.

Le dernier chapitre met en relief la possibilité de commander une application résidente via internet.

Le rapport est achevé par une conclusion et des perspectives. Les différents chapitres font des renvois à des annexes pour des approfondissements.

Chapitre I:

ETAT DE L’ART DE

E-LEARNING ET CADRE DU PROJET

* 1. **Introduction**

Dans ce chapitre, on introduit le cadre de ce projet en présentant le cahier des charges et son contexte tout en mettant en relief ses intérêts et ses apports pédagogiques.

* 1. **Cadre du projet**
     1. **Contexte**

Le télé-TP est l’une des composantes récentes de la formation à distance. Pourtant essentiel pour compléter la e-formation, ce thème est peu abordé dans les projets pédagogiques et scientifiques actuels. [1]

ESPRIT, l’Ecole Supérieure Privée d’Ingénierie et de Technologie occupant la cinquantième place dans le classement des *Top 100* universités africaines [2] (publié par le site de l'hebdomadaire Web Manager Center[3]), et le premier rang des deux seules universités tunisiennes classées (ESPRIT et l’Université Virtuelle de Tunis « UVT »), nous a proposé la réalisation de ce projet d’actualité.

En effet, sachant que la présence des établissements sur le Web est l’un des critères sur lesquels se base le classement cité plus haut. Ce projet s’inscrit dans le cadre du développement et de l’enrichissement de sa plateforme de travail numérique hébergée sur le serveur d’ESPRIT.

Le sujet du projet s’intègre dans le cadre du développement des moyens d’apprentissage à distance. En effet, l’e-Formation et le e-Learning sont deux termes différents pour un contexte unique :

« L’enseignement s’est profondément renouvelé en s’engageant résolument sur la voie de l’utilisation des nouvelles technologies » [4]: Il s’agit, de nos jours, de campus universitaires virtuels qui rassemblent une multitude d’universités permettant l’enseignement à travers Internet. Les contenus pédagogiques étant en ligne, l’apprenant n’a plus besoin de se déplacer ; Il peut accéder de n’importe où et à n’importe quel moment aux cours. Toutefois, cette formation à distance ne peut être satisfaisante qu’en présence de formation pratique qui permet, en fait, de mieux assimiler les notions théoriques apprises. D’où l’apparition du terme e-TP ou télé-TP.

Grâce à l’e-TP, il devint possible de réaliser une manipulation de Travaux Pratiques à distance, c'est-à-dire, via Internet. Comme pour un TP présentiel, l’apprenant est capable de manipuler et de contrôler à distance des équipements de mesures situés dans les salles d’expérimentation.

Mise à part son importance du point de vu pédagogique, le télé-TP favorise la collaboration fructueuse et le partage des ressources matérielles distribuées dans le monde.

Quant à l’état d’évolution du e-Learning dans le monde Arabe, on dispose uniquement de six universités virtuelles, dont une seule est tunisienne (UVT) tandis qu’ils sont des centaines dans d’autres pays du monde développé [5]. Espérant donc que ce modeste projet soit un point de départ pour la réduction d’une telle fracture pédagogique et numérique…

* + 1. **Cahier des charges**
* **Enoncé du cahier des charges**

Il s’agit de concevoir et réaliser une plateforme expérimentale d’électronique accessible via Internet. Ceci permettra à l’étudiant d’accéder de n’importe quel endroit et de n’importe quel moment à la manipulation de leur choix et de la réaliser.

***Mots Clés :***

Travaux pratiques à distance, interface d’instrumentation (IEEE-488), connexion PC-Instrument.

Travail à réaliser :

Le travail à réaliser comporte quatre parties fondamentales :

1\* Maitriser la connexion PC-Instrument

2\* Développer un logiciel permettant de gérer un oscilloscope de la famille Tektronix TDS 1002 (voir Annexe A, Figure 29) à partir du PC.

* Lire la notice technique, les détails sur les spécifications techniques de l’instrument (voir Annexe A, Figure 31).
* Etudier les accessoires disponibles (voir Annexe A, Tableau 3)
* Examiner les entrées sorties existantes afin d’élaborer un dialogue PC-Instrument (voir Annexe A, Tableau 4)

3\* Portabilité de l’application à distance

4\* Enrichissement de la base des manipulations.

Pour la restitution de l’environnement d’un TP présentiel, on doit procurer aux apprenants un moyen de visualisation en temps réel des instruments manipulés et cela à travers une webcam installée avec la plateforme du TP.

Les Travaux Pratiques présentiels présentent quelques limitations. On en cite, d’une part, la limitation temporelle, du fait que les séances de TP occupent une portion horaire minime dans les emplois du temps, malgré leur importance. De plus, les TPs peuvent se dérouler dans un ordre non pédagogique et défavorable pour certains étudiants. D’autre part, les groupes d’apprenants peuvent être nombreux dans le cas où les équipements d’une manipulation sont onéreux et ne peuvent pas être dupliqués. Du coup, l’e-TP parait être une solution capable de combler ces limitations.

Voyons alors jusqu’à quel point cela peut être vrai ?

* 1. **Etat de l’art**

En se basant sur différents articles qui traitent le sujet des Télé-TPs, on a pu constater que les recherches et les réalisations relatives à ce sujet sont rares. Nous essayons de donner une vue panoramique sur les différents documents concernés par le télé-TP.

En effet, un projet de recherche de LVEST (Laboratoires virtuels pour l'éducation en science et technologie) du LICEF [5] traite la modélisation d’un laboratoire virtuel avec une plateforme de télé-expérimentation à usage pédagogique et scientifique. Dans le document (Saliah1999) [1], il s’intéresse à une étude pour un laboratoire en ligne. [6] Le Laboratoire ICTT (Interaction Collaborative, Téléformation, Téléactivités), INSA de Lyon présente une modélisation d’un e-TP d’automatique. Sur le Web, on trouve quelques sites permettant la réalisation de TP à distance mais qui sont inaccessibles, [7] peut être faisant partie aux intranets des établissements, certains sont d’initiative privée. [8]

* 1. **Conclusion**

Dans ce chapitre, nous avons présenté le cadre du projet en mettant l'accent sur l'état de l'art. Les recherches ainsi menées ou les projets qui sont en cours de réalisation nous ont servis dans le choix du paradigme de la résolution de la problématique.

Chapitre II :

GENERALITES SUR LES RESEAUX INDUSTRIELS

# Introduction

Le rôle principal d’un réseau industriel est de transmettre des informations entre plusieurs machines. On fait communiquer différents appareils (ordinateurs, automates programmables, appareils de mesurage, smart sensors, actionneurs, équipements spécifiques, etc.).

Le qualificatif d’Industriel réfère en général à un environnement perturbé, qui est souvent pollué par des ondes électromagnétiques provenant des différents appareils (moteur, courants forts, champs magnétiques, etc.).

Faire circuler un message entre des machines distantes dans un environnement pareil, peut présenter un vrai défit si on désire que la machine cible perçoit le message avec le minimum d’erreur.

La communication inter-machines est donc un sujet vaste et complexe. Les standards sont nombreux, Nous allons aborder dans ce chapitre les normes les plus importantes par leur popularité dans le monde industriel. Nous nous intéressons particulièrement aux bus IEEE et aux liaisons séries avec les appareils de mesurage,

# Type des interfaces PC-Système Numérique

# Pour pouvoir transmettre ou recevoir des données, un ordinateur utilise soit une liaison série soit une liaison parallèle appelée « BUS ». Depuis l’avènement des ordinateurs personnels (PC), de nombreux types de bus sont apparus dont la structure dépend du microprocesseur utilisé, de l’usage désiré et surtout du fabricant.

* + 1. **La liaison série**

Pour connecter deux machines entre elles, la liaison série constitue sans doute l’un des supports physiques de transmission de l’information les plus réputées. Les données sont envoyées bit par bit les uns à la suite des autres (exemple : liaison série des ordinateurs, Ethernet, etc.). Cette liaison a l’avantage d’être plus compacte et de ne posséder que quelques signaux utiles.

* **Liaison série asynchrone**

La liaison est dans ce cas totalement asynchrone, aucun signal d’horloge n’est transmit. Les données sont envoyées de l’émetteur vers le récepteur sans une négociation préalable. Afin que les éléments communicants puissent se comprendre, il est nécessaire de configurer les deux extrémités selon un protocole de transmission. Le récepteur doit se synchroniser sur l’émetteur. [9]

Les paramètres qui entrent en jeu sont :

* La longueur de mot : Le message transmis doit être codé sur 7 ou 8 bits.
* La vitesse de transmission : Le débit de transmission de l'émetteur doit être identique au débit d'acquisition du récepteur. Les différentes vitesses de transmission son réglables à partir de 110 bauds (bits par seconde) de la façon suivante : 110 bauds, 150 bauds, 300 bauds, 600 bauds, 1200 bauds, 2400 bauds, 4800 bauds, 9600 bauds, 14499 bauds, 19200 bauds, 38400 bauds, 57600 bauds, 115200 bauds, 128000 bauds et 256000 bauds.
* Bit de START et Bit de STOP : La transmission est asynchrone (pas d’horloge commune entre l’émetteur et le récepteur), des bits supplémentaires sont indispensables au fonctionnement de la liaison : bit de début (START) et bit de fin (STOP).
* La parité : C’est une méthode de détection des erreurs de transmission. Il existe deux types de parité : Paire et Impair.

La norme RS232 définit les valeurs des tensions de signalisation que doivent fournir et reconnaitre les interfaces séries :

* Un « 0 » logique correspond à une tension varie de +8V à 40V.
* Un « 1 » logique correspond à une tension varie de -8V à -40V.

En général, les signaux transmis sont compris entre -12V et +12V.

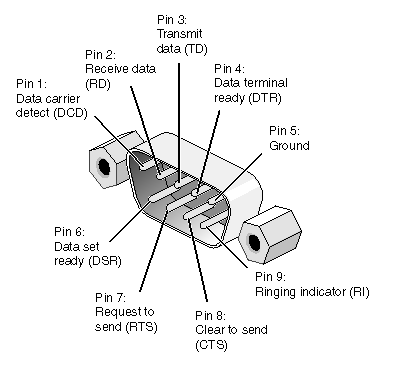
Pour exploiter le mode de transmission Half Duplex du support, c'est-à-dire, établir un échange de données dans les deux directions, il faut disposer au minimum de trois fils : un pour la transmission et l’autre pour la réception. Cette technique de liaison nécessite un algorithme qui permet de contrôler le flux des données sur les deux systèmes connectés.

Le tableau 1 résume le brochage d’un connecteur série 9 pins :

*Tableau 1: Connecteur 9 pins de la liaison série RS232 [10]*

|  |  |  |
| --- | --- | --- |
| Broche | Nom | Description |
| 1 | DCD | Détection de la porteuse |
| 2 | RX | Réception des données |
| 3 | TX | Transmission des données |
| 4 | DTR | Terminal prêt |
| 5 | SG | Masse électrique |
| 6 | DSR | Emetteur prêt |
| 7 | RTS | Demande de transmission |
| 8 | CTS | Prêt à émettre |
| 9 | RI | Indicateur de sonnerie |

L’image de la figure 1 traduit le tableau situé en haut sous forme graphique :



*Figure 1: Vue en perspective d’un connecteur série DB9 (Connecteur male)*

Selon la description des broches, on peut les classer en trois groupes selon leurs rôles :

* Lignes de données

Ces sont des lignes permettant la transmission et la réception des données (RD et TD) par rapport à un signal de référence (SG).

* Lignes de contrôle du flux

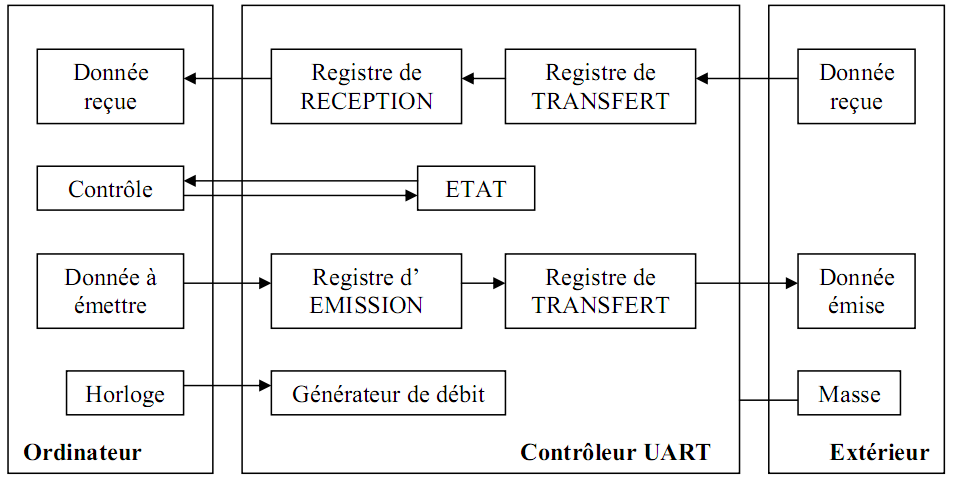
Le signal RTS indique à l’émetteur que son correspondant est prêt à recevoir des données. Le signal DSR indique que l’émetteur est toujours présent.

* Lignes d’état

Ces lignes indiquent la présence d’une tonalité et d’une porteuse. (RI et DCD).

* ***UART (Universal Asynchronous Receiver Transmitter)***

Le module UART permet d'établir une communication entre un PC et l'extérieur. Sa structure logique obéit au schéma bloc de la figure 2:



*Figure 2: Structure de l'UART*

* **Les différents protocoles**
* **Protocole Matériel**

Lorsque l’émetteur veut émettre ses données, il doit positionner la ligne RTS pour demander au récepteur s’il est prêt à accepter ces données. Le récepteur, une fois prêt à recevoir les données, va envoyer le signal DSR de l’émetteur pour lui notifier qu’il est prêt. Si l’émetteur veut suspendre la transmission, il va enlever le signal DSR.

* **Protocole XON/ XOFF**

Ce mode de transmission ne nécessite qu’une liaison sur 3 fils. Le reste de la communication se fait de façon logicielle. Ce protocole est basé sur les caractères XON (Code ASCII correspondant 11H) et XOFF (Code ASCII correspondant 13H).

L’émetteur continuera la transmission jusqu’à ce qu’il reçoit une notification auprès du récepteur lorsque le buffer de ce dernier atteint le 80% de son volume. Le caractère transmit est XOFF. Une fois notifié, l’émetteur doit immédiatement suspendre son émission. Il ne continuera sa transmission de nouveau qu’après la réception du caractère XON, c'est-à-dire que le récepteur a vidé son buffer jusqu’à la moitié, soit 50%. Il se peut que pour une raison ou une autre, l’émetteur ne reçoive pas les caractères déjà mentionnés. Pour pallier à ces problèmes, ce dernier peut prendre la décision de commencer le transfert, si le récepteur n’est pas encore prêt, il peut toujours remettre un XOFF.

* + 1. **La liaison GPIB**

Dans les applications industrielles faisant intervenir plusieurs instruments, ce type de liaison arrive en tête devant la liaison série (tous formats RS-232, RS-422, RS-485…).

Le GPIB, ou General Purpose Interface Bus, est une liaison de type parallèle, elle fut inventée en 1965 par Hewlett-Packard et normalisé ensuite par l’IEEE. Il est, depuis sa création, le principal standard de communication entre les instruments d’un banc expérimentale de tests automatiques. Il est dédié spécialement pour la mesure et la commande d’instruments. En effet, sa structure est conçue spécialement pour véhiculer des commandes d’instruments (ordre de mesure, ordre de commande, résultats, gestion des erreurs, ouverture de connexion, résultat de transfert, etc.).

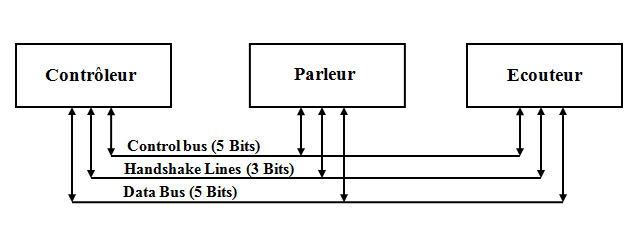
Le bus entier se compose de 24 lignes, Il comprend:

* 8 lignes de données bidirectionnelles
* 5 lignes de contrôle
* 3 lignes de handshaking

La logique de négociation est de type TTL, c'est à dire logique négative vraie.

Le taux de transfert maximal de données est déterminé par un certain nombre de facteurs, mais il est supposé être 1.8 Mo / s selon le constructeur KEITHLEY [11].

Ce bus permet, comme le montre la figure 3, de mettre en communication trois types de dispositifs: Contrôleur (Gérer les échanges sur le canal), Parleur (Diffuser des informations sur la bus) et Ecouteur (Percevoir ces informations). Il est capable de faire relier jusqu'à 14 instruments, chaque appareil peut avoir une ou plusieurs des fonctions suivantes, mais, à un instant donnée, il doit y avoir un seul contrôleur actif, un seul parleur actif, un ou plusieurs écouteurs.



*Figure 3: Hiérarchie des appareils dans un bus GPIB*

L’adaptateur KUSB-488A de la famille KEITHLEY, utilisé dans ce projet (Voir Annexe B, Figure 32), offre une simplicité d’utilisation sans précédent, après avoir installé le pilote correspondant, puisqu’il fonctionne de manière “Plug and Play”, c’est-à-dire que l’on peut débrancher et rebrancher la prise USB en fonctionnement sans avoir besoin de configurer quoi que ce soit. Le périphérique connecté est reconnu automatiquement.

# Brochage *du connecteur GPIB*

La connexion entre les différents appareils se fait par des câbles multiconducteurs sur des connecteurs de type RIBBON. La figure 4 montre les dimensions mécaniques exprimées en mm d’un tel connecteur :

# Forme Méchanique.PNG

*Figure 4: Dimensions mécaniques d’un connecteur de type RIBBON*

Le tableau 2 décrit les pins présents sur le bus :

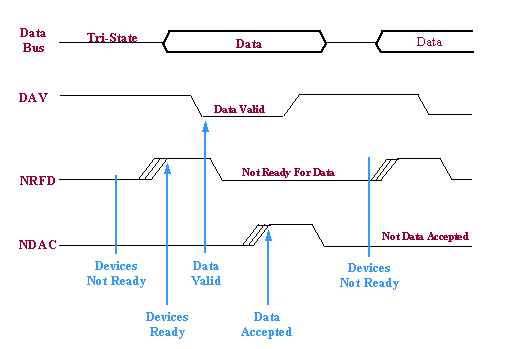
*Tableau 2: Brochage d’un connecteur de type RIBBON [12]*

|  |  |  |  |
| --- | --- | --- | --- |
| N° | Lignes | Désignation | Commentaires |
| 1 | DIO1 | **D**ata **I**n **O**ut 1 | Donnée 1 |
| 2 | DIO2 | **D**ata **I**n **O**ut 2 | Donnée 2 |
| 3 | DIO3 | **D**ata **I**n **O**ut 3 | Donnée 3 |
| 4 | DIO4 | **D**ata **I**n **O**ut 4 | Donnée 4 |
| 13 | DIO5 | **D**ata **I**n **O**ut 5 | Donnée 5 |
| 14 | DIO6 | **D**ata **I**n **O**ut 6 | Donnée 6 |
| 15 | DIO7 | **D**ata **I**n **O**ut 7 | Donnée 7 |
| 16 | DIO8 | **D**ata **I**n **O**ut 8 | Donnée 8 |
| 11 | ATN | **A**t**T**e**N**tion | 1 : tous les appareils attendent une commande |
| 6 | DAV | **Da**ta **V**alid | 0 : validation des données par le transmetteur |
| 5 | EOI | **E**nd **O**r **I**dentify | 1 : fin de transmission |
| 9 | IFC | **I**nter**F**ace **C**lear | 1 : initialisation de tous les appareils |
| 8 | NDAC | **N**ot **D**ata **AC**cept | 0 : donnée non acceptée |
| 7 | NRFD | **N**ot **R**eady **F**or **D**ata | 0 : non prêt à recevoir des données |
| 17 | REN | **R**emote **EN**able | 1 : appareil sous contrôle du bus |
| 10 | SQR | **S**ervice **R**e**Q**uest | 1 : demande de service |

La masse est reliée aux bornes 18 à 24, le blindage est à la borne 12.

# Protocole de dialogue

Cette procédure permet la gestion des échanges des informations à partir des lignes DAV, NFRD, NDAC comme l'indique la figure 5:



*Figure 5: Chronométrage de transfert de données dans un bus GPIB [13]*

Ce type de dialogue permet de maintenir deux machines ou programmes synchronisés. Le handshaking concerne l'échange des messages ou des paquets de données entre deux systèmes équipés d'un buffer limité. Ce protocole permet à l'expéditeur de demander au destinataire s'il est prêt à recevoir ou pour le récepteur de répondre avec un accusé de réception négatif signifiant qu'il n'a pas reçu le dernier message correctement.

Concernant le contrôle de flux des données pour la détection éventuelle d'erreurs, il en existe trois modes: Hardware (Signaux électriques ou des impulsions), Software (caractères ACII par exemple), ou bien Flow Control (il peut utiliser soit un contrôle matériel ou logiciel comme il peut les combiner).

# Synthèse

Associé à une interface logicielle implantée sur un PC, le bus GPIB permet de réaliser un système de test complet en termes d’application logicielle. Il est d’une robustesse éprouvée et permet la synchronisation des équipements sur un banc de tests.

Pour mieux visualiser l’intérêt d’une telle connexion, un tableau comparatif (voir Annexe B, Figure 33) résume les caractéristiques des différentes alternatives de communication.

Pour débuter la programmation avec ce type d’interfaçage, le fabricant KEITHLEY offre deux moyens : Une interface graphique réalisé en Visual Basic (voir Annexe B, Figure 34) et un invite de commande accessible via le menu Démarrer» Tous les programmes» Keithley Instruments» GPIB-488, cette invite permet le diagnostic du bus et l’établissement d’un dialogue avec le périphérique connecté.

# Conclusion

Après avoir examiné les différents bus d’instrumentation, on peut dire que faire communiquer les instruments n’est plus une tâche fastidieuse, tant qu’on sait comment configurer nos plateformes pour bien échanger les données. Nous avons présenté dans ce chapitre, une démarche en vue d’établir un dialogue qui assure une communication entre un PC et un système périphérique.

Chapitre III :

COMMANDE DES INSTRUMENTS A TRAVERS UNE INTERFACE LOGICIELLE

# Introduction

Dans ce chapitre nous allons expliquer comment exploiter les liaisons décrites au chapitre 1 pour transférer des données entre l’instrument et l’ordinateur.

La technologie que nous allons adopter au cours de ce projet est la norme IEE-488, cependant, il est préférable d’aborder la thématique de connexion par la liaison série puisqu’on est habitué à travailler avec ce type d’interface. Et elle peut nous servir à démontrer notre capacité de communiquer avec le banc de travaux pratiques: envoi de commande et lecture de réponse. Nous allons d’abord mettre en relief les différents utilitaires permettant ce test. Nous consacrerons par la suite le reste du chapitre pour la liaison GPIB.

# Commande des instruments à partir des utilitaires logicielles

* **Le port série avec l’HyperTerminal**

C’est une interface graphique légère capable d'établir une connexion entre deux ordinateurs par un câble sur le port série (COMx). HyperTerminal se présente en fait comme un bloc-notes doté de fonctionnalités de communication. Les données transmises sont des données binaires ou sous forme de texte.

La syntaxe de commande des appareils obéit au format standard définie par le SCPI[[1]](#footnote-1).

* **Le port série avec LabView**

La plateforme LabView offre des outils et des modèles pour répondre à des défis spécifiques, allant de la conception d'algorithmes de traitement du signal jusqu'au prélèvement de mesures de tension. Il permet de cibler un certain nombre d'exercices avec un paradigme graphique puissant [14].

Ces interfaces intuitives sont à l'origine de l'instrumentation virtuelle, elle est efficace dans de nombreux types d'applications, de la conception au prototypage et déploiement.

Ce logiciel a la souplesse d'un langage de programmation combiné avec des outils intégrés conçus spécialement pour le test, mesure et contrôle. Ainsi, on est capable de créer des applications qui s’étendent d’une surveillance simple à une commande sophistiquée des instruments connectés sur la périphérie d'un ordinateur.

Avec l'instrumentation virtuelle, les ingénieurs utilisent de la programmation graphique pour créer des solutions qui répondent à leurs besoins spécifiques, ce qui est une excellente alternative pour fixer la fonctionnalité des instruments traditionnels.

Pour mieux comprendre l’intérêt d’une telle solution, nous avons testé la liaison série avec un programme et la figure 6 montre l’interface réalisée :

# LabViewSerie.PNG

*Figure 6: Communication série sous LabView*

Dans la suite de ce travail, nous avons opté pour l’utilisation de MATLAB (MATrix LABoratry). En effet, LabView ne reconnait que les pilotes de NI (National Instrument) alors que nous disposons à ESPRIT d’adaptateur KUSB-488A de type KEITHLEY ayant ses propres pilotes. MATLAB, quant à lui, ne pose pas ce type de problème.

# Commande des instruments à partir de MATLAB

A partir d’une session MATLAB, on est capable de commander un instrument avec n’importe quel moyen, soit graphique (Simulink) ou bien programmable. Cette section expose les différents moyens utilisés pour consulter un instrument et établir un dialogue robuste avec lui.

# La fenêtre de commande : Command Window

La fenêtre de commande ou bien Command Window permet à l’utilisateur de spécifier le type de dialogue qu’il désire établir, bien évidemment, sous réserve que les supports de communications soient physiquement disponibles et que MATLAB a bien détecté ces périphériques. Il suffit tout simplement de taper la commande « instrhelp » sur la prompt de commande, MATLAB va générer une liste des ports de connexion disponibles.

* Pour commander un instrument via le port série, la procédure est la suivante :

%Créer un objet série

*Scope=serial('com1');*

%Ouvrir le port et établir une connexion entre l'objet et l'interface série

*fopen(Scope);*

%Questionner l'instrument

*fprintf(Scope, '\*IDN ?');*

*Response=fscanf(Scope);*

%Déconnecter l'instrument et détruire l'objet

*fclose(Scope);*

*delete(Scope);*

* Pour commander un instrument via le BUS GPIB, la procédure est la suivante :

%Construire un objet connecté à une technologie Keithley

*Scope=gpib('keithley', 20, 1) ;*

%Connecter l'objet GPIB à l'instrument:

*fopen(Scope);*

%Questionner l'instrument

*fprintf(Scope, '\*IDN?') ;*

%Récupérer la réponse

*Rep=fscanf(Scope);*

%Déconnecter l'instrument

*fclose(Scope);*

%Détruire l'objet

*delete(Scope);*

%Nettoyer la variable associé à l'objet considéré de la mémoire et la libérer

*clear Scope;*

Il s’avère utile d’utiliser cette invite de commande. Néanmoins MATLAB possède un outil de test et mesure intégré à la boite à outils de contrôle des instruments (Instrument Control ToolBox) appelé TMTOOL qui remplace ces routines de programmation.

# La boite à outil : TMTool

TMTool ou bien Test & Measurement Tool, permet d’afficher toutes les ressources (pilotes de périphériques,  interfaces, etc.) accessibles. Il permet également de configurer et de communiquer avec ces ressources.

Cette boite à outils permet de communiquer directement avec des instruments. Avec cet outil, on peut générer des données dans MATLAB, les envoyer à un instrument, les visualiser...

Il offre une interface souple indépendante du fabricant, du protocole ou du pilote et cohérente avec les dispositifs présents. TMTool prend en charge les connexions Série, GPIB, TCP/IP, UDP[[2]](#footnote-2).

Un soutien est également prévu pour les connexions VISA[[3]](#footnote-3), IVI[[4]](#footnote-4), VXI[[5]](#footnote-5) plug and play [15].

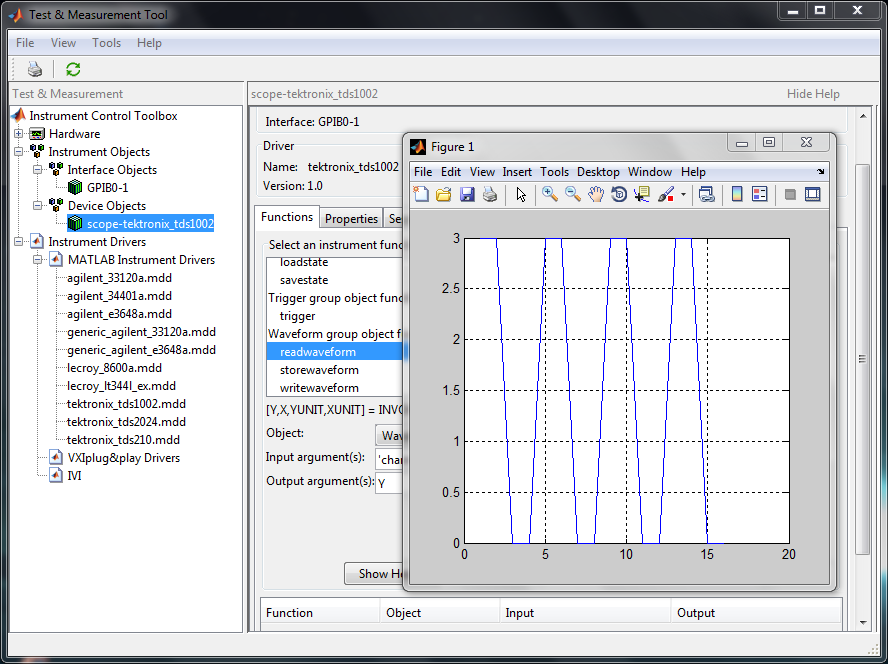
A l'aide de TMTool, la communication et la configuration des instruments se fait sans écrire un code au préalable. Cette interface graphique permet aux programmeurs et non programmeurs de:

* Rechercher le matériel disponible
* Se connecter à un instrument
* Configurer les paramètres de l'instrument
* Ecrire des données
* Lire la réponse

TMTool génère automatiquement le code à partir de la session de contrôle d'instrument. La sauvegarde de ce code permet d'exécuter les mêmes commandes de la même façon sans avoir besoin à appeler cet outil une autre fois.

Un exemple d’exploitation de cet environnement est illustré sur la figure7.

Sur la barre de tâche verticale gauche, on distingue la rubrique Instrument Driver qui permet de sélectionner l’instrument désiré (Tektronix TDS 1002). Deux alternatives peuvent être utilisées, soit l’INTERFACE OBJECT, dans ce cas nous devons dialoguer avec l’instrument par sa langage (\*IDN ?, \*ESR ?...), soit DEVICE OBJECT, l’utilisateur, ainsi, contrôle l’appareil avec un langage de haut niveau c.-à-d. compréhensible par l’être humain (exemple : ReadWaveform, Connect, etc.).



*Figure 7: Utilisation de TMTool pour la commande de l’oscilloscope*

# L’environnement de développement graphique : GUIDE

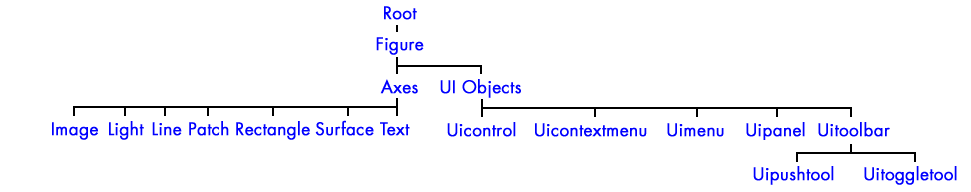
MATLAB est un logiciel de l’état de l’art mathématique, c’est un laboratoire de calcul matriciel. Il constitue donc un outil de base pour la résolution des problèmes ou des routines variées. Néanmoins, MATLAB possède toute une panoplie d’objets graphiques qui permettent à l’utilisateur d’interagir dynamiquement avec une interface informatique. Cette partie comporte

* Une description succincte des objets graphiques (hiérarchie, identifiant, propriétés)
* Une présentation de l'outil GUIDE pour le développement des interfaces graphiques

Les interfaces Homme Machine sont appelées GUI c'est-à-dire Graphical User Interface. Elles permettent à l’utilisateur d’interagir avec un programme donné grâce à des objets graphiques (Boutons, menus déroulants…)

Du fait du nombre important de ces objets et surtout du nombre élevé des paramètres associés, leur programmation à la main déroute l’utilisateur, pour cette raison, MATLAB a introduit, depuis sa version 5.0, c'est-à-dire en 1997, un environnement de développement graphique appelé guide (Graphical User Interface Development Environment). Cet outil permet de concevoir des interfaces graphiques de façon intuitive.

Sous MATLAB, les objets graphiques sont disposés selon une hiérarchie pyramidale parent enfant comme suit :



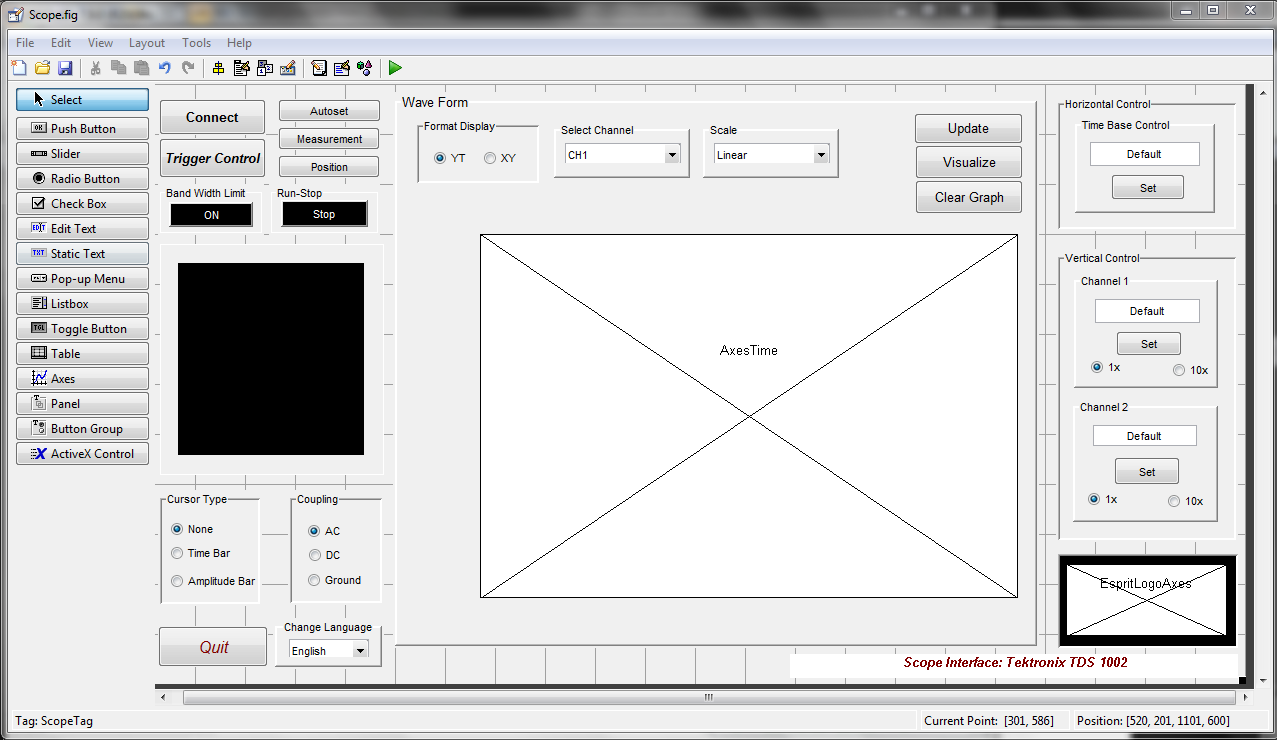
*Figure 8: Hiérarchie des objets graphiques sous MATLAB [16]*

Au sommet de la hiérarchie se trouve l'objet Root. C’est un objet invisible. L'utilisateur interagit très rarement avec cet objet. Viennent ensuite les objets de type Figure. Ces sont les conteneurs visibles où sont disposés tous les autres objets enfants. Ensuite, on trouve l’objet Axe, c’est un conteneur visible ou sont placés les zones de traçage de graphiques 2D ou 3D représentant des résultats mathématiques (courbe, surface, image…).

Les objets de contrôle se situent au même niveau que celles des axes, ces éléments permettent à l’utilisateur d’interagir avec le GUI de façon dynamique.

Ces outils simplifient considérablement le processus d'aménagement et de programmation des interfaces graphiques.

GUIDE est donc un outil graphique qui regroupe tout ce dont le programmeur a besoin pour créer sa propre interface graphique de façon intuitive. Pour mettre en relief la manipulation de ces objets, la figure 9 montre l’interface mère qui commande l’oscilloscope et qui fait appel à d’autres interfaces (Position, Measurement, Trigger Control…). Après avoir ouvrir l’éditeur des figures (GUIDE), nous avons placé tous les objets que nous désirions apparaitre lors de l’exécution de l’interface. Le fonctionnement de la GUI est piloté par une caméra, placé juste en face de l’écran de l’oscilloscope, afin de s’assurer que l’instrument connecté répond aux requêtes qui lui ont été transmisses. Le bouton qui gère la transmission de vidéo est nommé VISUALIZE. Lors de l’appui sur ce bouton, MATLAB va créer automatiquement un objet vidéo. Grâce à la commande preview, un flux d’image est disponible mais il faut préciser le conteneur de ce streaming, c.-à-d. spécifier un objet graphique de type axe, c’est celui colorié en noir. Pour arrêter le streaming, il suffit d’appuyer de nouveau sur le même bouton.



*Figure 9: Aperçu de la mise en page graphique de l’oscilloscope*

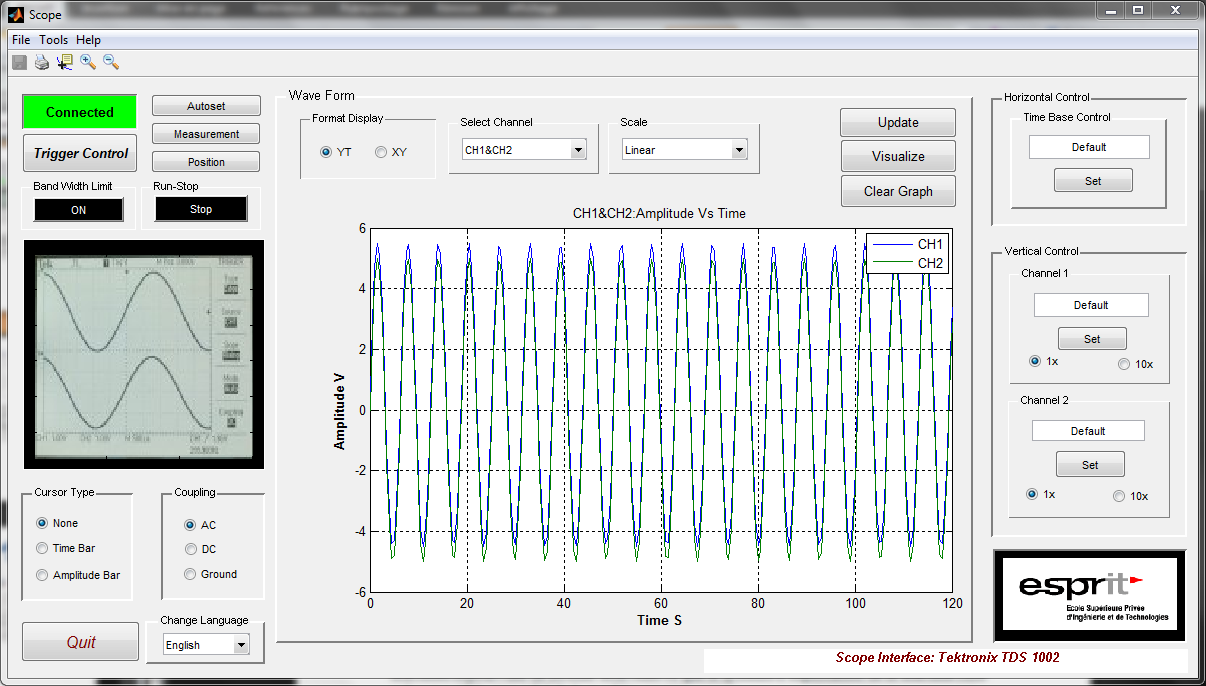
Les objets de contrôle disponibles sur cette interface peuvent être classés en deux catégories :

* Contrôle Local : Ces sont les boutons qui permettent de changer le comportement graphique de l’interface, par exemple changer l’échelle de l’axe contenant les courbes, faire du zoom, enregistrer les données, etc.
* Contrôle de l’instrument : Ces boutons permettent à l’utilisateur de dialoguer avec l’instrument. En général, on procède de la façon suivante : Tout d’abord, on clique sur le bouton Connect pour faire connecter l’oscilloscope à l’ordinateur. L’étudiant pourrait, ainsi, faire son manipulation de façon ordinaire. On a conservé la même terminologie, celle existante sur la face de l’oscilloscope (Changer la base de temps, faire bouger les curseurs, auto calibrage de l’oscilloscope…).

Le placement des objets est réalisé par sélection à partir de la palette, mise en place et mise à dimension à la souris. Un double-clique sur chaque objet permet de faire apparaître un menu avec les propriétés de cet objet. Leur modification est immédiate. Au final, le code est généré automatiquement et l'interface est enregistrée sous deux fichiers portant le même nom mais dont les deux extensions sont .fig et *.m*. Le premier contient la définition des objets graphiques, c’est un fichier crypté. Le second contient les lignes de code qui assurent le fonctionnement de ces objets.

Apres avoir enregistré la mise en page graphique désirée, GUIDE génère automatiquement un fichier «\* .m ». Ce fichier permet de contrôler la façon dont l'interface fonctionne. Il fournit du code pour initialiser le GUI et organise les callbacks, c'est-à-dire les fonctions qui s'exécutent en réponse à des événements générés par l’utilisateur, comme un clic de souris par exemple.

La figure 10 montre la validation expérimentale de cette interface, on rappelle que l’instrument est un oscilloscope (Tektronix tds1002), connecté à un pc exécutant Windows XP professionnel via un adaptateur GPIB (KUSB-488A de KEITHLEY).

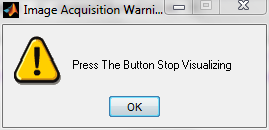


*Figure 10: Validation expérimentale de l’interface graphique de l’oscilloscope*

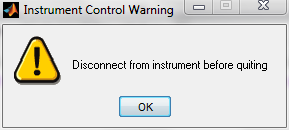
L’interface récupère les deux sorties de l’oscilloscope, les deux signaux présents sont de type sinusoïdal.

La manipulation est surveillée par une caméra qui montre l’effet réel de l’appui sur chaque bouton de l’interface. L’image transmise par la caméra est visible sur la figure 10.

Si on désire abandonner l’exécution de la manipulation, le GUI notifie l’utilisateur voulant quitter cette session de la nécessité de fermer la caméra (Figure 11) et déconnecter ’instrument (Figure 12).



*Figure 11: Avertissement : Acquisition d’images*



*Figure 12: Avertissement : Contrôle d’instruments*

# Outil de déploiement

Créer une application indépendante de l'environnement MATLAB revient à compiler les fichiers correspondants. Ce compilateur, intégrée en tant que module complémentaire dans la procédure d'installation de MATLAB, permet de déployer des applications autonomes, sur des machines dépourvues de License sous réserve que le système d'exploitation de la machine cible soit compatible avec celui de la machine source.

Pour pouvoir bénéficier du compilateur de MATLAB, il faut exécuter la commande mbuild -setup afin de spécifier le compilateur LCC[[6]](#footnote-6).

La démarche est indiquée sur la figure 13:

# Compilateur.png

*Figure 13: Spécification du compilateur*

La vitesse de lancement et d'exécution de l'application ainsi compilée est identique à celle de son lancement sous MATLAB suivie de l'interprétation du script.

Toutefois, pour pouvoir lancer l'application sur la machine cible, un soutien complet de librairies partagées qui permettent d'exécuter les fonctions développés sous MATLAB doit être présent, c.-à-d. il faut installer le moteur MCR (MATLAB Compiler Runtime).

Concernant la compilation simultanée de plusieurs fonctions, il faut passer uniquement le fichier parent MATLAB possède un outil qui cherche la dépendance entre les fichiers et permet à ce stade d'intégrer tous les autres fonctions dont dépend le programme principale tel que les boites à outils, les graphiques...

La figure 14 montre les différents produits possibles issus du compilateur de   
MATLAB.

# Image1.png

*Figure 14: Architecture de la compilation sous MATLAB [17]*

Utiliser Ce compilateur permet d'en paqueter l'application en:

* Application autonome
* Bibliothèques C ou C++ (DLL en Windows ou bibliothèques partagées sous Unix et Linux).
* Composantes logicielles: classes Java, .Net ou Excel add-ins avec MATLAB Builder.

Cette architecture réduit considérablement le temps de développement puisqu’il n’est plus nécessaire de traduire un code MATLAB en un langage spécifique.

MATLAB dispose d'une interface GUI, une alternative graphique indiqué sur la figure 15, qui permet de déployer les programmes. Pour utiliser cet outil, il faut:

* Désigner la fonction principale.
* Ajouter les fichiers de support qui ne peuvent pas être détecté par le biais du contrôleur de dépendance.
* Préciser les préférences de la compilation et de l'empaquetage.

# DeployTool.png

*Figure 15: Assistant Graphique de déploiement*

Lorsqu’on crée ce programme, le système d'exploitation Windows attribue, par défaut, une icône à l'exécutable :

# Executable.png

*Figure 16: Icone associé à l'exécutable*

On montrera ici, une autre méthode de compilation et comment associer une icône à une application.

On considère qu'on dispose d'un fichier icone "\*.ico". Il a été, soit créé avec des logiciels spécialisés, soit extrait d'un autre programme (autres exécutables ou des fichiers dll). Pour changer l'icone de l'exécutable, Il va falloir utiliser un fichier de ressources, c'est un fichier propre à Windows (Il n'est pas disponible sous Linux et Mac Os). Son extension est "\*.rc".

Il est possible de créer un fichier de ressource avec le compilateur LCC expédié avec MATLAB. Il suffit de suivre la procédure suivante.

1\* Nommer le fichier icone «InstrumentIcon.ico ».

# 33.png

*Figure 17: InstrumentIcon.ico*

2\* Créer un fichier texte qui contient le code suivant:

*"ConApp ICON InstrumentIcon.ico"*

Ce code est composé de:

* Le type de ressource: ICON pour icone. Le nom du fichier à inclure. Ici, l'icone s'appelle « InstrumentIcon.ico ».
* Elle doit, impérativement, se trouver dans le même répertoire que le fichier de ressource au moment de la compilation.
* Renommer le fichier comme étant «RessourceInstrumentIcon.rc » et l'enregistrer dans le répertoire courant avec l'icone.

3\* Compiler «RessourceInstrumentIcon.rc » et « InstrumentIcon.ico » en utilisant la commande suivante

*system(['"' matlabroot '\sys\lcc\bin\lrc" /i "' pwd '\ RessourceInstrumentIcon.rc"']);*

Ou matlabroot est la racine où installé MATLAB sur l'ordinateur.

A la fin de cette phase, un fichier nommé «ressourceinstrumenticon.res »aura lieu dans le répertoire courant.

4\* Placer le fichier déjà mentionné dans le répertoire du projet afin de les compiler avec la commande suivante:

*mcc -m Password\_Confirmation.m -M ressourceinstrumenticon.res -o Instrument Control Application -v*

Où:

mcc est la commande qui permet d’invoquer le compilateur LCC.

-m est une option qui indique le fichier principal, qui fait appel à d'autres fichiers.

-M indique le fichier de l’icône.

-o Le nom de l'exécutable

-v Mode verbeux, visualiser au fur et à mesure les étapes de la compilation (Verbose Output).

Ceci va créer un exécutable nommé «Instrument Control Application.exe » qui a pris la forme de l'icone « InstrumentIcon.ico » dans l'explorateur Windows.

Pour plus d’informations sur les options disponibles qui peuvent être accordés au compilateur, il suffit de taper la commande help mcc sur l’invite de commande de MATLAB.

# Création d’installation

Cette partie concerne la création d'installations pour les plateformes Windows uniquement.

Un programme est un monde de variables, des boucles, des conditions, jeu de fonctions…

En général, on désire coder ce programme, ce type d’exercice serait bien trop long, une vraie perte de temps. En plus, c'est assez compliqué car le programme d'installation doit combiner tous les fichiers du programme dans un seul exécutable, et il doit les archiver aussi. Néanmoins, Il existe de nombreux outils permettant de créer une installation.

L'outil présenté ici est très connu pour sa réputation. Son nom est **Inno Setup**.

Pour télécharger Inno Setup, rendez-vous sur le site officiel du logiciel :

<http://www.jrsoftware.org/isinfo.php>

Cliquer sur le lien Download et récupérer le programme d'installation.

Après avoir installé le programme, la première fenêtre de l'assistant d'installation apparaitra : C’est un assistant de création de scripts. Il génèrera le code de création de l'installation de façon automatique pour l’utilisateur. Toutefois, pour pouvoir personnaliser une installation professionnelle, il faudra éditer le fichier de configuration de l'installation. A l’issu de cette phase, on doit avoir un programme ayant la forme suivante où toutes les propriétés sont personnalisables par l’utilisateur :

# 12.png

*Figure 18: Setup de l’exécutable*

Lancer ce setup permet d’avoir un raccourci placé sur le bureau indiqué sur la figure 19:

# 2.png

*Figure 19: Raccourci vers l’exécutable Instrument Control Application*

# Conclusion

Nous avons exposé dans ce chapitre les différentes méthodes où les modes opératoires de communication en vue d’établir une connexion PC-Instrument via les deux outils disponibles (Série, GPIB). Le produit réalisé est une version bureautique qui permet à l’utilisateur d’avoir une interface totalement fonctionnelle qui remplace tous les boutons présents sur la face avant de l’oscilloscope. A l’issu de cette phase, nous savons que commander un instrument de façon fiable ne présente pas de problème. Cependant, rappelons que la finalité de ce projet est de commander l’appareil de mesure via internet, c’est l’objectif du chapitre suivant.

Chapitre IV :

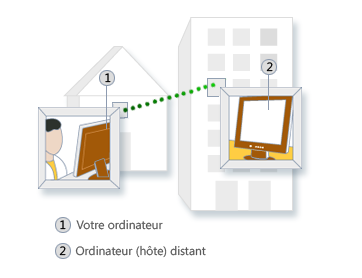
COMMANDE D’UN OSCILLOSCOPE VIA INTERNET

* 1. **Introduction**

Le pilotage des équipements de la plate-forme à distance «étant réalisé, on est prêt maintenant à étendre davantage l’application pour passer d’un contrôle local des instruments à un contrôle via le réseau Internet. Ce chapitre présente la plus importante étape du projet et traite les différents moyens permettant la réalisation d’une manipulation de Travaux Pratiques à distance via Internet : e-TP.

* 1. **Commande via Internet à partir des applications dédiées au contrôle distant**

Internet est un vecteur indispensable pour le contrôle et la commande à distance. En effet, un accès à distance peut se faire à tout moment et de n'importe quel endroit d’un ordinateur client à un équipement hôte à condition qu’ils soient tout les deux connectés à Internet. Toutefois, cela se fait moyennant des applications informatiques spécialisées.

****

*Figure 20: Contrôle des bureaux à distance*

On rappelle que la finalité de cette partie consiste à trouver un moyen logiciel permettant de commander le contrôleur des instruments de la plateforme de TP via un autre ordinateur connecté à Internet.

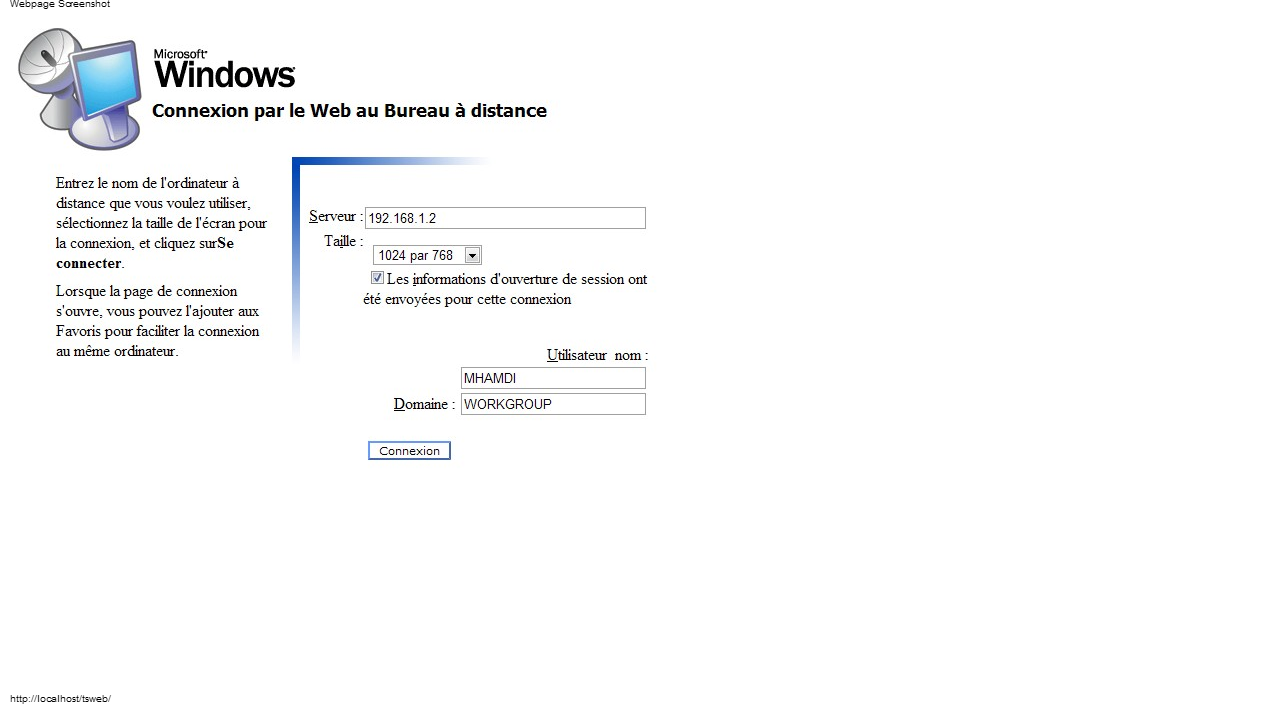
Pour ce faire, une multitude de logiciels se présente sur le Web permettant le contrôle des ordinateurs à distance. Nous proposons donc d’en présenter quelques-uns.

* + 1. **Connexion par le Web au Bureau à distance**

C’est une technologie de Windows permettant l’accès à un ordinateur distant qui se trouve à un emplacement lointain de l'ordinateur de contrôle

Cette technique permet d’accéder à tous les fichiers, à tous les programmes et à toutes les ressources de l’ordinateur cible.

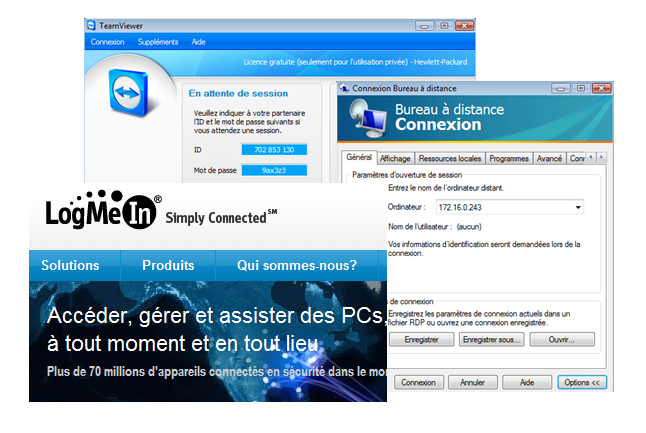
Pour activer la commande à distance, il faut d’abord télécharger des fichiers Windows nécessaires et taper l’adresse URL suivante sur le navigateur Internet Explorer en indiquant le nom du serveur : *Http://nom\_serveur/tsweb* et une fenêtre identique à celle indiquée sur la figure 21 apparaisse.

****

*Figure 21: Connexion par le Web au bureau à distance (TSWeb)*

* + 1. **Connexion par des applications dédiées au contrôle à distance**

Il s’agit en effet du même principe de connexion. Sauf que cette fois, il s’agit de logiciels à installer offrant toute une panoplie de fonctionnalités qui peuvent se faire à distance tel que la gestion, le transfert de fichiers, la surveillance, le diagnostic et la prise en charge de plusieurs ordinateurs distants. Parmi ces applications (voir figure 22), on cite TeamViewer, LogMeIn et le bureau à distance qui est une application Windows installée par défaut :

****

*Figure 22: Utilitaires de contrôle à distance*

Il est donc trop simple d’accéder au bureau de l’ordinateur qui contrôle les instruments de la plate-forme à partir de n’importe quel ordinateur connecté au réseau Internet et donc de commander les instruments à travers ‘Instrument Control Application’ moyennant l’une des méthodes de contrôle de bureau citées.

Toutefois, l’inconvénient majeur de cette technique est l’absence de toute sorte de sécurité. En effet, une fois que la connexion entre les deux PC est établie, on a un accès total et sans limites à toutes les ressources de l’ordinateur hôte ce qui n’est pas tolérable.

Pour cela, il faut trouver un autre moyen de connexion plus sur et sécurisé où les activités de l’apprenant se limitent à la réalisation de la télémanipulation, tout en restant dans le cadre de l’environnement technique choisi.

* 1. **Commande à partir des sessions MATLAB en utilisant le protocole TCP/IP**

Pour nous mettre dans ce contexte, on commence tout d’abord par définir le protocole TCP/IP :

Développé au sein de la recherche militaire américaine, le TCP/IP acronyme de Transmission Control Protocol / Internet Protocol est l’ensemble des règles qui assurent la communication entre machines faisant partie d’un réseau Internet.

Grâce à sa robustesse prouvée par le fait qu’il gère quelques millions de machines interconnectées dans le monde, ce protocole de communication est bien répandu.

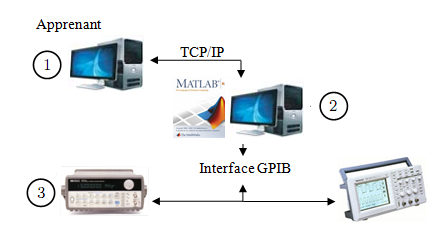
Sur un réseau géré par le protocole TCP/IP, chaque machine est identifiée par une adresse unique dite adresse IP. Or, sur une même machine, plusieurs applications peuvent être exécutées à la fois. Chaque application est aussi identifiée par une adresse unique codée sur seize bits appelée port. [18]

Du coup, pour acheminer des paquets de données entre deux applications logées sur deux machines distinctes sur le réseau, il suffit de préciser l’adresse IP et le numéro du port de l’application destinataire. La combinaison d’adresse IP et du numéro du port s’appelle Socket.

Donc, lorsqu’un ordinateur reçoit des données spécifiques à un port, il les transmet directement vers l’application destinée. Si la donnée reçue est sous forme d’une requête alors l’application est dite serveur. S’il s’agit d’une réponse, alors elle est une application cliente.

Après l’avoir présenté, le protocole TCP/IP parait être un autre moyen de contrôle d’instruments. Néanmoins, l’oscilloscope et le générateur de fonction à piloter ne sont pas équipés d’une interface de ce type. Donc le contrôle directe des instruments par TCP/IP n’est pas possible.

Toute fois, il peut être assuré à travers un environnement médian, capable de communiquer en même temps avec l’ordinateur de l’apprenant par le TCP/IP et avec l’instrument à travers la liaison GPIB (Figure 23). C’est le rôle assuré par l’environnement MATLAB :



*Figure 23: Contrôle des instruments via Internet*

En effet, il est possible d’établir une communication entre deux instances de MATLAB ouvertes dans deux différentes machines communiquant par le protocole TCP/IP.

Cette communication peut être établie même au niveau d’une instance d’une même machine. MATLAB se comporte alors comme étant un écho-serveur, qui est en écoute des messages en provenance de soi même.

Le code suivant résume les commandes permettant de transformer MATLAB en un écho-serveur TCP/IP communicant à travers le port numéro 4012 :

% Démarrer le serveur

*echotcpip('on',4012) ;*

% Créer un objet TCP/IP

*t = tcpip('localhost',4012);*

% Ouvrir le port

*fopen(T) ;*

% Ecrire sur le port

*fwrite(T, 0:5) ;*

% Lire les données en provenance du réseau local

*Y=fread(T) ;*

% Fermer l’objet

*fclose(T) ;*

% Détruire l’objet

*delete(T) ;*

% Désactiver le serveur

*echotcpip('off') ;*

Le même principe sera conservé pour assurer la communication entre l’ordinateur contrôlant les instruments et celui de l’apprenant; Sauf que MATLAB est à la fois configuré en serveur et/ou client à travers les commandes suivantes :

*server (message, numéro\_port, nombre \_tentatives)*

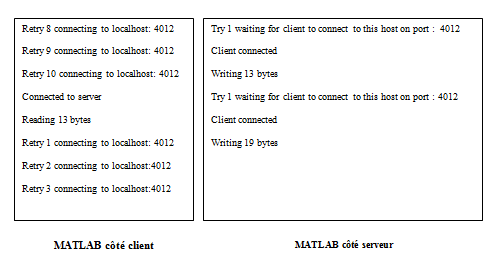
*client (nom\_\_serveur, numéro\_port, nombre\_tentatives)*

Pour établir la connexion entre les deux instances, il faut forcer le client à écouter le port adressé pour un nombre défini de fois: C’est le nombre de tentatives d’écoute précisé dans la commande *client*. Sinon, pour le configurer à une permanence écoute du serveur, cet argument doit être passé à « -1 ». Ceci oblige le client à être esclave de la connexion au serveur jusqu’au bout.

C’est la stratégie qu’on a suivi pour établir un synchronisme permanent entre les machines, puisqu’on ne peut pas prédire leurs temps de réponse qui est généralement lié aux performances des ordinateurs. En effet, si pour une raison ou autre, un de deux terminaux (coté contrôleur et coté apprenant) cesse pour un instant de recevoir les données, l’autre continue la tentative d’émission ou de réception de l’échantillon correspondant jusqu’à l’établissement d’une nouvelle connexion sur le même port. Le processus de connexion se rétablie jusqu'à la réception du dernier échantillon.

Quant au message transféré entre les deux sessions de MATLAB, il doit être obligatoirement de type chaine de caractères.

La figure 24 illustre un scénario de transfert de données entre deux instances de MATLAB par TCP/IP :



*Figure 24: Logique de négociation entre les deux sessions de MATLAB*

Le principe de communication entre l’application de contrôle des instruments ’Instrument Control Application’ (étant serveur respectivement client) et l’instance MATLAB lancée au niveau du contrôleur communicant avec les instruments (étant client et respectivement serveur), se résume comme suit :

Dès que le client, concerné par l’écoute, réussis à se connecter au serveur, le message suivant s’affiche et la procédure d’envoi s’enclenche. :

*>>connected to server*

*>>reading \*\* bytes*

Ce processus se répète jusqu’à l’envoi total du message. Une fois que l’acquisition est terminée, le client revient à sa position de repos.

Si le terminal est maintenant serveur, ses tentatives sont donc de diffuser un message particulier vers un port spécifique. Un client voulant percevoir cette demande, doit s’adresser à ce port. Une fois connecté, l’écriture du message commence et la notification suivante s’affiche :

*>> Client connected*

*>> Writing \*\* bytes.*

Ce scénario se répète de façon continue chaque seconde jusqu’à l’achèvement de la transmission.

Si l’utilisateur désire abandonner la manipulation courante, MATLAB, coté instruments, revient de nouveau vers un état de permanence écoute du port 4012 pour recevoir un nouveau ordre d’exécution de la manipulation.

Nous avons essayé d’intégrer la notion de programme modulaire, pour cela nous avons intégré un fichier nommée « ExecutingOrder » au niveau PC hôte qui gère toute la manipulation.

Son code est le suivant :

global nameHost;

nameHost='@IP';% Changer cette valeur par l’adresse IP convenable

while 1,

message=client(nameHost, 4012);

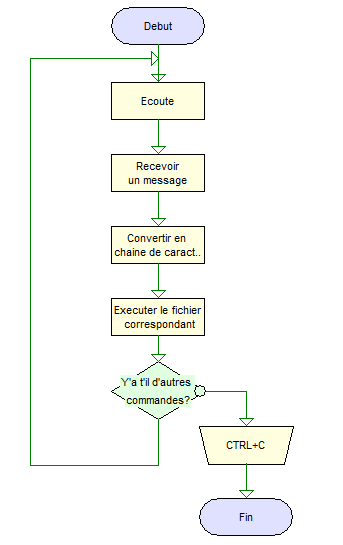
message=char(message);

run(message);

end

Où nameHost est une variable indiquant le domaine du PC voulant se connecter ou bien son adresse IP sur le réseau local.

La gestion de données se fait donc selon l’organigramme indiqué sur la figure 25 :

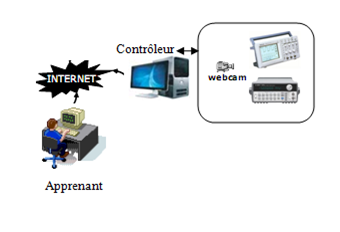


*Figure 25: Organigramme de gestion d'ordres au niveau PC hôte*

Pour mieux saisir le principe de communication via Internet, on se propose de mettre en relief une expérimentation décrivant les différentes étapes de communication entre un apprenant et les instruments de la plate-forme:

* **Validation Expérimentale**

La figure 26 illustre l’organisation de l’expérimentation :



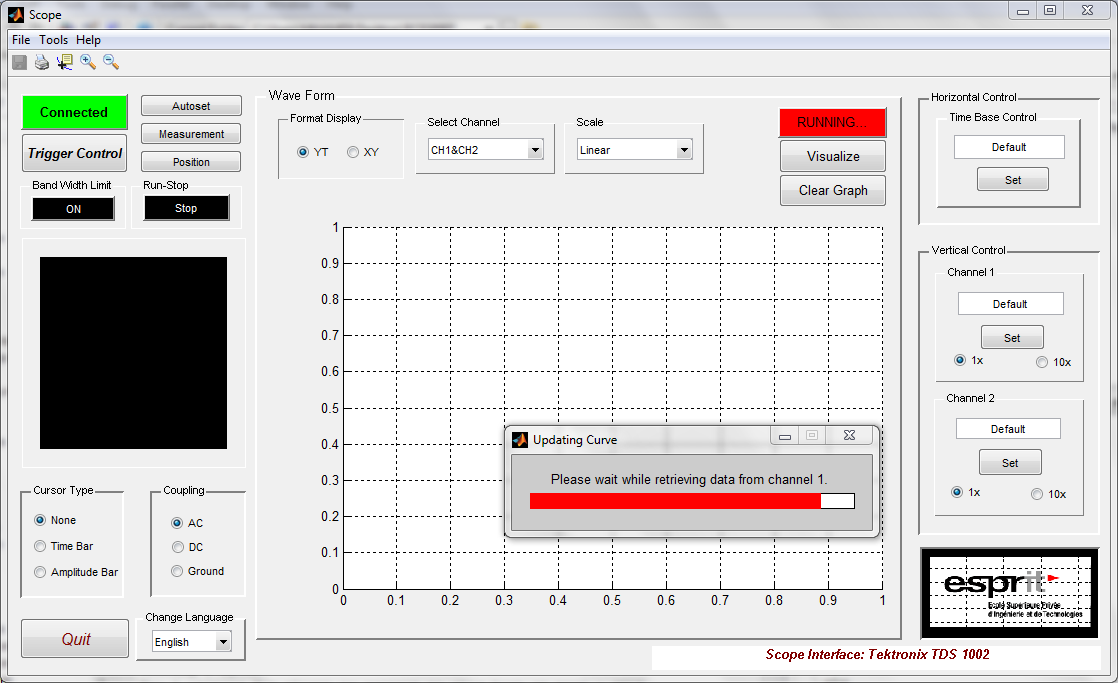
*Figure 26: Principe du test expérimental*

Prenons le cas d’acquisition de deux courbes différentes à partir du canal 1 et du canal 2 de l’oscilloscope et ceci doit se faire à travers la touche « Update » de l’interface de commande à distance. Pour ce faire, il faut acquérir le vecteur de donnée du canal 1 également du canal 2 et un troisième champ correspondant au vecteur temps qui les accompagnent.

Sachant que chacun de ces vecteurs compte 250 échantillons, le client doit finalement recevoir un message qui compte 750 échantillons plus un échantillon indiquant la taille des vecteurs. Or, on peut diminuer considérablement la taille du message à recevoir et gagner en temps d’envoi et cela en envoyant seulement les deux vecteurs de données plus un échantillon indiquant leur taille et un autre échantillon indiquant la période d’échantillonnage des courbes qui remplacera le vecteur temps. C’est en faite la technique qu’on a adopté pour l’acquisition des courbes.

Une fois que le client est connecté au serveur, une boucle de lecture sera exécutée pour la réception de la totalité des échantillons correspondants à chaque vecteur.

L’utilisateur est alerté par la nécessité d’attendre grâce à un bar qui montre l’achèvement de l’acquisition (Voir figure 27), chaque pas coloré en rouge correspond à un échantillon présent dans la mémoire.

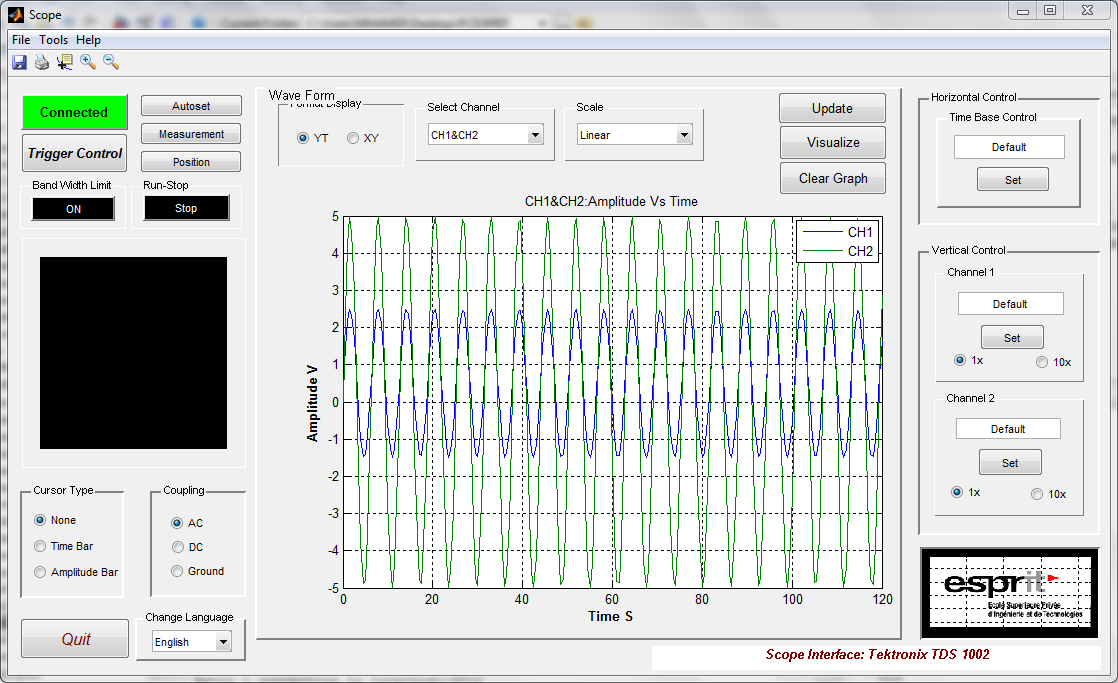


*Figure 27: Extraction des données du canal 1 de l’oscilloscope*

La même procédure s’applique au canal 2.

Après avoir lu toutes les données disponibles sur les ports spécifiques, le traitement de ces informations se fait en local (Raffinage et Affichage). Il s’agit après de dessiner les courbes en local tout en respectant l’ordre des échantillons de données et la période de reconstitution du signal émise.

L’évolution temporelle de deux canaux est celle indiqué sur la figure 28 :



*Figure 28: Evolution temporelle de deux signaux extraits de l’oscilloscope*

A l’issu de cette phase, l’étudiant est capable de d’enregistrer ces données dont il a besoin à partir de la barre d’outils an appuyant sur l’icône de disquette, ou bien à partir de la barre de menu, sous File. Ainsi, il est capable d’exécuter d’autres transformations ou analyses sur ces données qui sont sauvegardés dans un fichier qui porte l’extension « \*.mat »

On peut ainsi remarquer que la transmission des données est fiable au niveau valeurs des messages transmis, les données ne sont pas endommagées. Toutefois, deux problèmes sont perçus: La transmission vidéo et la cadence constante des échantillons entre les deux pc : 1 échantillon par seconde. Une solution peut être envisagé, c’est la possibilité de changer la fréquence d’envoi des données, on risque alors de consommer beaucoup de ressources au niveau mémoire et de détériorer les performances en temps de réponse du processus. Une autre solution qui peut être proposée est la possibilité d’envoyer des trames complètes, ceci fait appel au concept de COM Object ou les classes .Net. En effet, grâce à MATLAB Builder, on peut générer des fichiers DLL ou des classes compréhensibles par d’autres logiciels plus performants au niveau développement WEB (ex : Microsoft Visual Studio).

* 1. **Conclusion**

Ce chapitre a montré la façon d’établir une télécommunication entre deux applicatifs distants. L'approche est basée sur une architecture client/serveur, elle est écrite en MATLAB mais elle fait appel à des sockets JAVA (Téléchargeables à partir du site de Mathworks).

Néanmoins, elle souffre de deux lacunes : La première est liée d’associer à chaque PC une adresse IP et la rendre à la porté de l’autre PC voulant communiquer avec ce partenaire : Il faut toujours fixer les adresses. La deuxième est la manière d’héberger cette application dans un serveur et de la rendre accessible par tous les internautes sans avoir besoin de configurer quoi ce soit et sans avoir recours à installer une application sur son propre PC.

CONCLUSION GENERALE

Dans ce mémoire de PFE, nous avons montré la possibilité de réaliser des travaux pratiques à distance via internet.

Dans le présent rapport, nous avons exposé en premier lieu notre réflexion qui a permis le choix des méthodes mises en œuvre pour la réalisation de la plateforme. La seconde partie a détaillé les techniques d'accès au port série et GPIB à partir du logiciel MATLAB, ensuite une troisième partie a présenté la communication avec un instrument via le web. Les résultats des traitements des données et la présentation de résultats issus de la validation expérimentale ont été intégrés dans chaque partie selon le contexte.

Ce projet a mis en exergue la télécommunication entre deux applicatifs distants. Cette plateforme est conçue pour transformer le fait de surfer sur le web en un puissant environnement de l'expérience pratique à distance.

Nous avons mis au point les dispositifs pédagogiques qui renforceront certains aspects dans le futur. Nous avons pu montrer que l’éducation utilisant internet comme canal de diffusion peut toucher également les laboratoires pédagogiques.

En revanche, l’e-TP est un projet qui permet à un apprenant de mettre en scène sa formation pratique et forge son intégrité au niveau résolution de problèmes physiques existants. Cette approche vise à fournir à un étudiant distant l'accès à une variété des manipulations. Cette thématique d’expérimentation offre une souplesse au niveau prototypage de l'environnement expérimental. Etant donnée la considération géographique distribuée des apprenants, elle permet ainsi aux apprenants d'avoir accès à une panoplie complète de bancs d'essai situés ailleurs leurs campus universitaires, ceci évite la nécessité de se déplacer physiquement et donc de perdre du temps et de l'argent.

La perspective immédiate à ce travail est de concevoir la plateforme de TP et y intégrer un nombre suffisant de manipulations. De plus, il peut être envisagé, à la manière des réunions distantes un concept de travail en équipe, le laboratoire de travaux pratiques peut être équipé d’outils de visioconférence. Dans ce cas, un seul étudiant peut avoir accès à la manipulation. Plusieurs étudiants peuvent suivre les différentes étapes de l'expérimentation seulement, observer l'avancement de l'exécution de la manipulation et discuter entre eux les résultats obtenus.

# Glossaire

# ASCII : American Standard Code for Information Interchange

# GPIB  : General Purpose Interface Bus

# GUI : Graphical User Interface

# GUIDE : Graphical User Interface Development Environment

# IEEE  : Institute of Electrical and Electronics Engineers

# IMAQ  : Image Acquisition Toolbox

# PC : Personal Computer

# RS232  : Recommended Standard 232

# TCP/IP : Transport Control Protocol/Internet Protocol

# TMTool  : Test & Measurement Tool

# UART : Universal Asynchronous Receiver Transmitter

# Références bibliographiques

# [ARP00] Arpaia, P., Baccigalupi, A., Cennamo, F. & Daponte, P. (2000). A Measurement Laboratory on Geographic Network for Remote Test Experiments. IEEE Transactions on Instrumentation and Measurement, 49(5), 992-997.

# [SAA01] Saad, M., Saliah-Hassane, H., Hassan, H., El-Guetioui, Z. & Cheriet M. (2001, Août). A Synchronous Remote Accessing Control Laboratory on the Internet. Communication à International Conference on Engineering Education, Oslo, Norvège.

# [BEN07] Benmohamed, H. (2007). ICTT@Lab: Un environnement informatique pour la génération et l’exécution de scénarios de télé-TP. Thèse de doctorat, INSA de Lyon.

# [LEL03] Lelevé, A., Prévot P., Subaï C., Noterman D. & Guillemot M. (2003, Juillet). Towards Remote Laboratory Platforms. Communication à la 7ème multi conférence internationale “Systemics, Cybernetics and Informatics” (SCI 2003), Orlando, Floride, USA.

# Netographie

[1] <http://edutice.archives-ouvertes.fr/docs/00/02/70/12/PDF/leleve.pdf>, consulté en Mai 2010

[2] <http://www.esprit.ens.tn/fr/info/actualite.htm>, consulté en Mai 2010

[3] <http://www.esprit.ens.tn/fr/info/actualite.htm>, consulté en Mai 2010

[4] [http://www.studyrama.com/vie-etudiante/agenda/archives/elearning-les-grandes-ecoles- in-ou-out.html](http://www.studyrama.com/vie-etudiante/agenda/archives/elearning-les-grandes-ecoles-%20in-ou-out.html), consulté en Mai 2010

[5] <http://www.licef.teluq.uquebec.ca>, consulté en Mai 2010

[6] <http://www.profetic.org/dossiers/dossier_imprimer.php3?id_rubrique=102>, consulté en Mai 2010

[7] <http://web2.cnam.fr/evariste/evariste/VisioHopital/travaux.htm>, consulté en Mai 2010

[8] <http://www.tpline.net>, consulté en Mars 2010

[9] <http://www.beyondlogic.org/serial/serial.htm>, consulté en Mars 2010

[10] <http://www.lammertbies.nl/comm/index.html>, consulté en Mai 2010

[11] <http://www.keithley.com/products/accessories/ieee>, consulté en Mars 2010

[12] <http://sitelec.org/cours/abati/ieee.htm>, consulté en Mai 2010

[13] <http://www.interfacebus.com/Design_Connector_GPIB.html>, consulté en Mai 2010

[14] <http://zone.ni.com/reference/en-XX/help/371361F01/lvmeasconcepts/>, consulté en Avril 2010

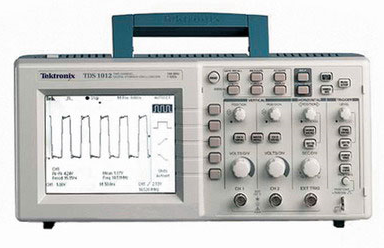
[15] <http://www.mathworks.com/products/instrument/description1.html>, consulté en Avril 2010

[16] <http://briot-jerome.developpez.com/matlab/tutoriels/guiintro/>, consulté en Mai 2010

[17] <http://www.mathworks.com/products/compiler/index.html?ref=pfo>, consulté en Mars 2010

[18] <http://sitelec.org/cours/abati/adressip.htm>, consulté en Mai 2010

# Annexe A



*Figure 29: Image de l’oscilloscope Tektronix TDS 1002*

*Tableau 3: Spécifications Techniques d’un oscilloscope Tektronix TDS 1002*

# SpecTech.PNG

# *Tableau 4: Accessoires recommandés*

# zaazdzaz.PNG

*Tableau 5: Tableau récapitulatif des interfaces Entrées-Sorties disponibles*

# rfrr.PNG

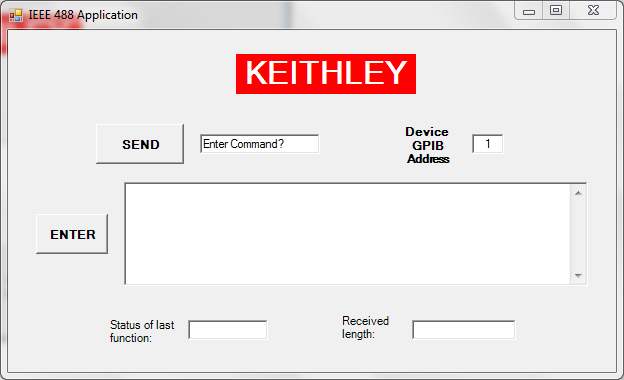
# Annexe B

# viewer.png

*Figure 30: Documentation Technique de l'adaptateur KUSB-488A*

# Comparaison des diif bus.PNG

*Figure 31: Tableau comparatif de différentes connexions*



*Figure 32: Interface de commande via GPIB bus*

**ملخًّص**

يندرج هذا المقرًّر ضمن مشروع ختم الدراسات العليا, وهو يقدم طريقة منهجية من أجل تصميم برمجيات تحكم في راسمة الذبذبات لغرض القيام بأشغال تطبيقية عن بعد. يهدف هذا المشروع إلى تبسيط مراحل التصميم وإيجاد العدة الملائمة لاحتياجات الأداة الآنف ذكرها. تتمثل المرحلة الأولى في تهذيب واجهة المستخدم الرسومية (GUI) وتحديد نسق البيانات بين الحاسوب والجهاز الطرفي. و تهدف المرحلة الثانية إلى شرح آلية إرسال المعطيات عبر شبكة الأنترنات و كيفية التنسيق بين مختلف الأجهزة.

**الكلمات المفاتيح**

تنسيق المعطيات, واجهة المستخدم الرسومية, أشغال تطبيقية عن بعد

**Résumé**

L'objet de ce travail de fin d'étude est d'étudier, de concevoir et de réaliser une plateforme permettant à un apprenant à partir d'un poste distant d'effectuer des travaux pratiques. L'intérêt d'une telle application est évident et répond à des besoins réels. Nous avons prouvé dans ce travail la faisabilité d'un tel projet et réalisé concrètement plusieurs composantes de cette plateforme. Nous avons proposé une démarche méthodologique en vue de la conception des algorithmes de commande d'un oscilloscope pour pouvoir faire des travaux pratiques à distance. Cette méthodologie vise à réduire les phases de conception d'une infrastructure de communication à distance et de trouver des technologies en adéquation avec l'instrument à commander. La première phase consiste à affiner l'interface graphique (IHM) et à maîtriser le protocole de dialogue entre l'ordinateur distant et l'instrument. La deuxième phase vise à étudier le mécanisme de transmission des données à travers le réseau d'Internet et exploiter ceci pour le contrôle et la gestion de l’instrument.

**Mots Clés**

Protocole de dialogue, interface graphique, travaux pratiques à distance

**Abstract**

The purpose of this end of study project is to explore, develop and implement a platform for a learner from a remote station to carry out practical work. We have shown in this work the feasibility of such a project and then implementing several components of this platform. We proposed a methodology for designing algorithms to control a scope in order to make remote experiments. This methodology aims to reduce the design phases of a communication infrastructure and identify technologies suited for the controlled instrument. The first step is to refine the Graphical User Interface (GUI) and master the communication protocol between the computer and the instrument. The second step aims to study the mechanism of data transmission across the Internet network platform and exploit this for the control and management of the instrument.

**Key Words**

Communication protocol, graphical user interface, remote experiments

1. Standard Commands for Programmable Instruments : Ce standard définit un ensemble des instructions de contrôle des équipements de mesures. [↑](#footnote-ref-1)
2. User Datagram Protocol [↑](#footnote-ref-2)
3. Virtual Instrument Software Architecture [↑](#footnote-ref-3)
4. interchangeable Virtual Instruments [↑](#footnote-ref-4)
5. VMEbus Extensions for Instrumentation, or IEEE 1155.VME eXtensions for Instrumentation. [↑](#footnote-ref-5)
6. “Local C Compiler” or “Little C Compiler” [↑](#footnote-ref-6)