Université de Franche-Comté
UFR Sciences et techniques

MASTER M1 M2E2
Année 2015

M. Franck Chollet

Projet Final Master M1

Interface de communication entre un capteur de déplacement et un PC

Étudiants

TORRES ZETINO Juan Carlos

OUINTEN Brahim

Besançon le 22 Mai 2015

Table des matières

1. Introduction

Le présente Projet a comme objectif la réalisation d'une interface série entre un capteur de déplacement et la PC

Le capteur est associe a un compteur numérique qui permet de visualiser les valeurs sur un afficheur 6 segments, il possède aussi une sortie pour envoyer les chiffres en format BCD.

Le but du système sera de capturer les valeurs provenant du compteur numérique pour pouvoir les analyser à travers d'une interface hardware /software qui permet de adapter les données pour une transmission vers la PC où ils seront visualisées dans une UI .

2. Présentation de fonctionnes De Système et Cahier de charges

D'abord nous aurons besoin d'un système qui commande l'acquisition de position de microlentilles, ce système doit pouvoir afficher les valeurs avec un compteur et transmettre ses données vers un dispositif extérieur
Les éléments qui composent notre cahier de charges dans cette étape seront

- *Compteur Digimatic EG-101P/D/Z*
- *Capteur Linear Gage LGF-0110L-B / 0125L-B*
- *Câble Mini D Ribbon (MDR)*

Le système aura besoin d'un hardware qui permet l'acquisition de données provenant du compteur, un microcontrôleur doit être utilisé qui servira à analyser de valeurs.
Il faudra prévoir un module de reset ainsi qu'un module en sortie adaptées aux spécifications techniques du compteur, le microcontrôleur devra pouvoir commander le temps d'acquisition de données. Les éléments qui composent le cahier de charges dans la partie Hardware seront :

- *Atmega32*
- *Module complémentaires de la carte électronique.*

Le système aura besoin d'un partie software qui gère les sorties et les entrées provenant du compteur, aussi il doit pouvoir adapter les données reçus pour une transmission vers la PC il faudra une plateforme de développement sur Windows pour faire le code et programmer la puce du contrôleur.
Les éléments qui composent le cahier de charges dans l'étape Software seront :

- *WinAvr et compilateur AVR dude.*
- *Langage adaptée pour programmer microcontrôleur.*

Finalement nous devrons nous communique avec la PC à travers d'une interface série et pouvoir afficher les données sur l'écran à travers d'un Panel d'utilisateur.
Les éléments qui composent notre cahier de charges dans cette étape seront

- *Interface Max232 TTL a 12Vdc*
- *Logiciel Labview*

3. Développement du diagramme structurelle du Système

Comme nous avons décrit le but du Projet consiste à réaliser une interface Parallèle et série afin de visualiser les donnes dans la PC, les parties qui devront être réalisées sont présentées de facon globale dans le diagramme suivant

Structure fonctionnelle du Système

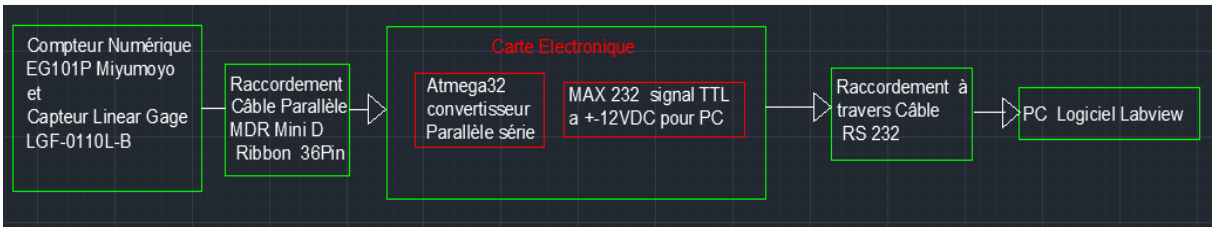


Schéma fonctionnel du Système Compteur -PC

A continuation nous allons décrire chacun des étapes avec la justification qui réponde aux besoins du système de mesure.0

3.1 Système Capteur Linéaire et Compteur numérique

La mesure de position se fait à travers d'un capteur Linaire de 0.1um de résolution, tandis que le compteur lui servira à afficher les valeurs mesurées. L'ensemble de ces deux dispositifs (capteur + compteur) forment un système.

On va récupérer les valeurs provenant du compteur et nous aurons besoins d'utiliser les soties en parallèle de ce diapositif, ces sorties utilisent un port à 36 Pins Mini D Ribbon .

Afin de pouvoir transmettre ses donnes à notre interface nous devrons réaliser une câble Mini D Ribbon qui est compatible avec la sortie du compteur.

Compteur

Digimatic EG-101P/D/Z

3.2 Hardware Interface Parallèle Série

Dans cette étape nous avons choisit une Microcontrôleur Atmega 32 qui possède les nombre de sorties nécessaires pour analyser les donnes du compteur dans notre cas 29 pins I/0

Nous aurons besoin aussi de commander le temps de comptage à travers d'un signal Hold provenant du microcontrôleur alors on va prévoir une sortie à collecteur Ouverte selon les spécifications du fabricant

Les sorties TXN et RXN qui servent à transmettre les données en série doivent être prévus sur le microcontrôleur afin d'envoyer les données à la PC , nous avons choisit cette mode car ce très répandu dans la plus part de diapositives de mesure et il n'est pas trop complexe à configurer sur le microcontrôleur .

3.3 Software Conversion Parallèle série

Dans cette partie nous avons choisi la plateforme de développement WinAVR et le compilateur AVR dude , ces deux outils sont disponibles sans frais et ils fonctionnent sur Windows 7/8 , il permet aussi de réaliser le code en C , nous avons choisi cette langage par son flexibilité et facilité à l'heure de programmer le microcontrôleur

Afin d'envoyer les donnes vers le Atmega depuis la PC pour stocker le programme nous avons choisi un module de USBasp

3.4 Communication serial avec PC

Dans cette partie nous avons s choisi l'envoie des données via un câble RS 232 vers la PC , les sorties TX du micro son TTL 5 VDC mais cela doit être adapter au PORT COM1 du PC qui a besoin des niveaux plus élevées dans notre cas +-12VDC

Afin d'affranchir cette étape nous avons utilisé une module MAX232 et un câble série RS232 qui permet de raccorder les deux parties PC –Microcontrôleur.

Interface serial RS- 232

Logiciel Labview National Instrument

Ensuite nous avons utilisé le Logicielle Labview pour réaliser une Interface d'Utilisateur, celle-ci doit afficher les Octets reçus depuis le compteur d'une manière interactive. L'utilisation de la bibliothèque VISA a permit de configurer les vitesses du port serial pour qu'il soit en synchronie avec les paramètres du microcontrôleur.

4. Développement de la Carte Electronique Hardware

Afin de établir une communication efficace entre le compteur numérique qui a des sorties en parallèles et le port serial de notre PC nous allons réaliser une interface qui utilisera le Microcontrôleur Atmega 32 et qui sera capable de commander le compteur, recevoir les données ,les analyser et les envoyer vers la PC .

Les étapes principales de la carte électroniques sont :

- Connexion et réception de données en parallèle
- Utilisation Horloge externe de 4Mhz
- Modules Complémentaires

La Carte électronique a été déssiné en utilisant le logicielle EagleCAD qui permet modéliser le routage et composantes Electroniques afin d'obtenir un motif qui permet son réalisation.

4.1 Connexion et réception de données en parallèle

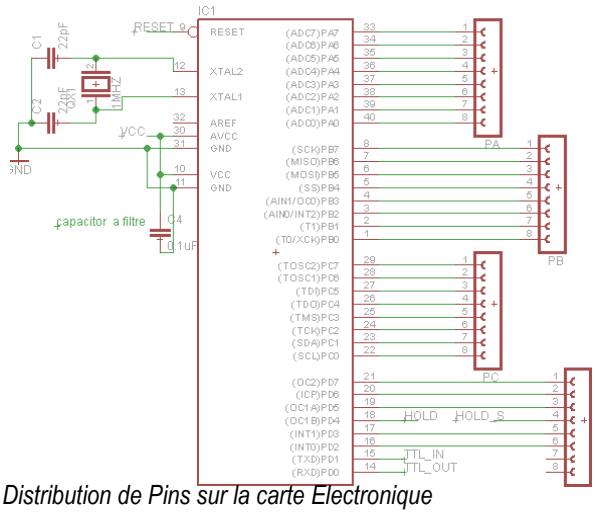
Dans cette partie nous allons récupérer les donnes provenant du compteur qui possède un sortie en parallèle à travers d'un câble Mini D Ribbon à 36Pins

Le tableau de référence pour tous les terminaux I/0 du compteur est présenté à continuation, nous présentons aussi la distribution de différent Pins dans les entrées du Atmega, la grande quantité de raccordements a été une de raisons pour utiliser le model de 32PINS

2) BCD output mode

Pin No.	I/O	Name	Pin No.	I/O	Name	Pin No.	I/O	Name
1		COM	13	O	$4X10^2$	25	O	$4X10^5$
2		COM	14	O	$8X10^2$	26	O	$8X10^5$
3	O	$1X10^0$	15	O	$1X10^3$	27	I	SET1
4	O	$2X10^0$	16	O	$2X10^3$	28	I	SET2
5	O	$4X10^0$	17	O	$4X10^3$	29	I	MODE
6	O	$8X10^0$	18	O	$8X10^3$	30	–	NC
7	O	$1X10^1$	19	O	$1X10^4$	31	O	SGN
8	O	$2X10^1$	20	O	$2X10^4$	32	O	NOM
9	O	$4X10^1$	21	O	$4X10^4$	33	O	REDY
10	O	$8X10^1$	22	O	$8X10^4$	34	I	HOLD
11	O	$1X10^2$	23	O	$1X10^5$	35	I	PSET
12	O	$2X10^2$	24	O	$2X10^5$	36	I	INH

Tableau de sortie en Paralelle



Distribution de Pins sur la carte Electronique

Dans notre application nous utiliserons les PINS suivantes :

- **24 Output Pins 6 Chiffres :** Contient les valeurs de données en format BCD, 4 bits par Nombre, il faut le configurer dans le compteur.
- **2 Pins Output SGN et READY :** Le Bit SGN permet d'identifier si le nombre est Positive ou Négative tandis que le bit Ready sert à indiquer que les données sont prêts à être lus par le microcontrôleur.
- **1Pin Input HOLD :** Le bit Hold sert à arrêter la valeur du comptage et permet au Microcontrôleur de capturées les données avant qu'ils changent en mettant à logique 1 le Hold.

4.1.1 Mode de Commande du Compteur

Le principe de fonctionnement de notre compteur dépend du mode de configuration de la sortie BCD nous aurons deux possibilités le **mode Intervalle et le mode commande**

Mode Intervalle

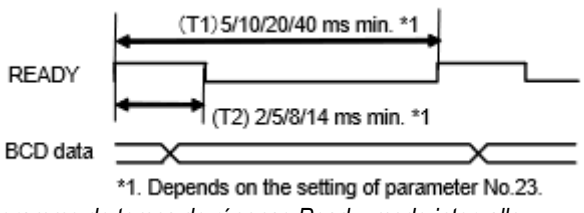


Diagramme de temps de réponse Ready, mode intervalle

Dans le mode Intervalle le compteur envoi les donnes tous les 10 ms selon la configuration qu'on a spécifié dans le paramètre t1, il n'y aura pas un signal hold car que les valeurs sont envoyées périodiquement.

Mode command

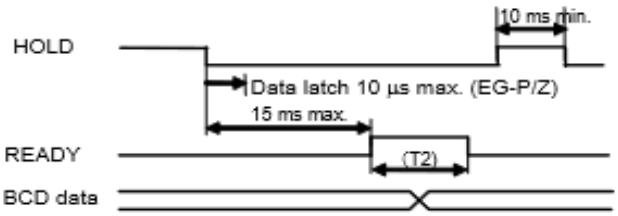
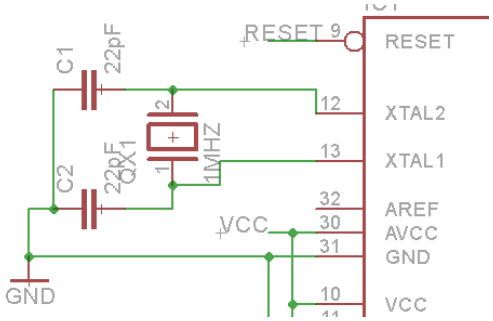


Diagramme de temps de réponse Ready, mode commande

Dans le mode command nous envoyons un signal depuis le microcontrôleur vers le compteur ,celui est appelé Hold qui est active en logique 0 , ce signal permet d'arrêter la valeur du comptage , ensuite, il répondra avec une impulsion READY qui indique que les valeurs sont prés pour entre lis par le microcontrôleur , dans notre cas le temps totale de préparation sera 15ms + t2(ready)2ms=17ms .

Il faudra configurer le microcontrôleur pour qu'il puisse attendre au moins cet intervalle avant de stocker la valeur de la mesure.

4.2 Horloge externe



Circuit d'oscillateur externe 4MHZ

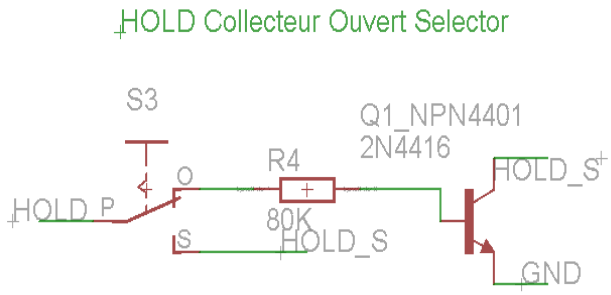
Nous allons utiliser une horloge externe en quartz à 4Mhz , cette vitesse sera suffisamment grande pour permettre l'acquisition de donnés provenant du compteur ,celui aura une vitesse moyen de transmission de l'ordre de millisecondes de plus il permettra de fixer une référence stable au lieu d'utiliser le horloge interne de microcontrôleur.

La configuration dans la carte ce fait avec 2 capacitance de 22 uF, un choix qui dépend de la valeur de fréquence à utiliser.

4.3 Modules complémentaires

4.3.1Sorties à Collecteur Ouverte

Nous aurons besoin d'une sortie à collecteur Ouverte du cote du microcontrôleur pour gérer le HOLD du compteur la sortie à collecteur ouverte sera implémenté du cote du microcontrôleur car le compteur possède une entrée avec une résistance de Pull-Up.



La sortie a collecteur Ouverte sert à connecter le collecteur du transistor à une résistance de pull up située dedans le compteur.

Quand le transistor NPN est active (saturée) le compteur reçois un niveau logique zéro dans ses entrées.

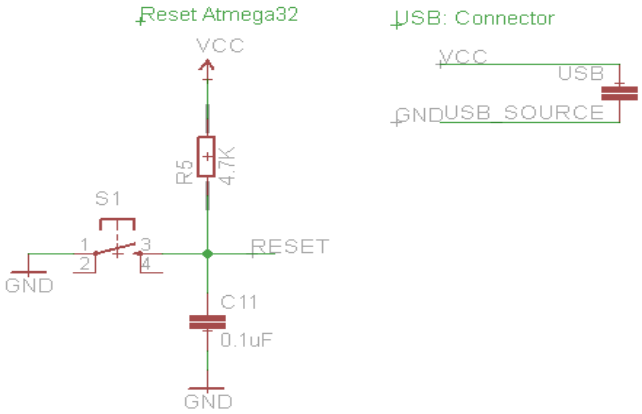
Par contre quand le transistor NPN est ouvert le collecteur devient à 5V et indique un niveau logique 1 .

Dans le schéma au dessus nous ajoutons la possibilité de connecter directement la sortie à l'entre HOLD de compteur en utilisant une Switch de sélection si on ne veut pas utiliser la sortie à collecteur Ouverte.

4.3.2 Module Alimentation USB et Reset Microcontrôleur

Afin d'alimenter le microcontrôleur Atmega32 ainsi que les différent modules connectées nous utiliserons une terminal USB qui permet d'obtenir le 5 VDC depuis la PC.

Notre carte électronique possède en plus un circuit reset qui permet de réinitialiser le programme stocké en mémoire flash dans notre microcontrôleur.

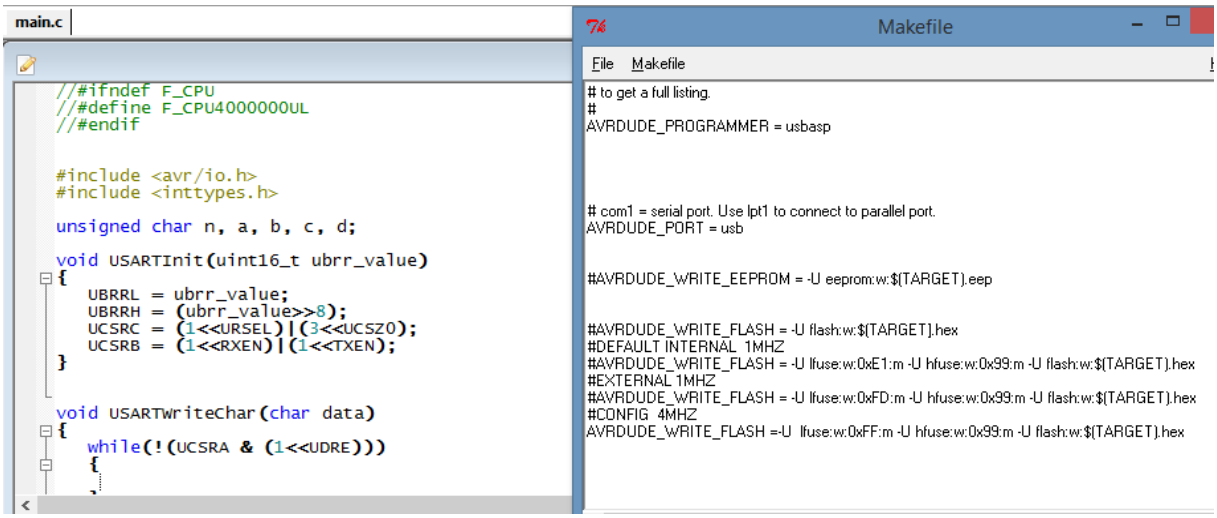


Circuit reset et Alimentation USB

5. Développent de Software du Microcontrôleur

Nous avons choisi l'utilisation du Logiciel WinAVR , celui c'est une logiciel gratuit qui permet la réalisation du code en C et C++ sur la plateforme Windows en plus il inclut une logicielle pour la génération de MakeFile pour programmer la mémoire flash du microcontrôleur

La compilation du code se réalise en utilisant le logiciel AVR dude , celui permet de programmer une grand variété de modèles Atmega dans notre cas le atmega32 , il faudra spécifier le type de raccordement à utiliser ainsi que les paramètres de vitesse de horloge en utilisant ce programme



Logicielle WinAVR et structure du MakeFile utilisé dans la programmation du microcontrôleur

A continuation nous présentons les deux fenêtres qui conforment la plateforme de développement de l'Atmega32 , dans la première à gauche nous allons programmer en langage C , dans la deuxième nous allons générer une fichier MakeFile pour compiler le code et configurer les autres paramètres spécifiques comme le registre de fusibles .

6. Développement d'une Interface d'Utilisateur en PC

Pour visualiser les paramètres provenant du microcontrôleur en format série nous allons utiliser le logicielle Labview qui permet de créer des instruments virtuelles VI et de les associer entre eux pour obtenir nouvelles fonctionnalités en ajoutant de bloques graphiques .

Dans notre cas nous avons choisi le bloques VISA qui permettent un traitement de donnes de manière séquentielle et ordonnée

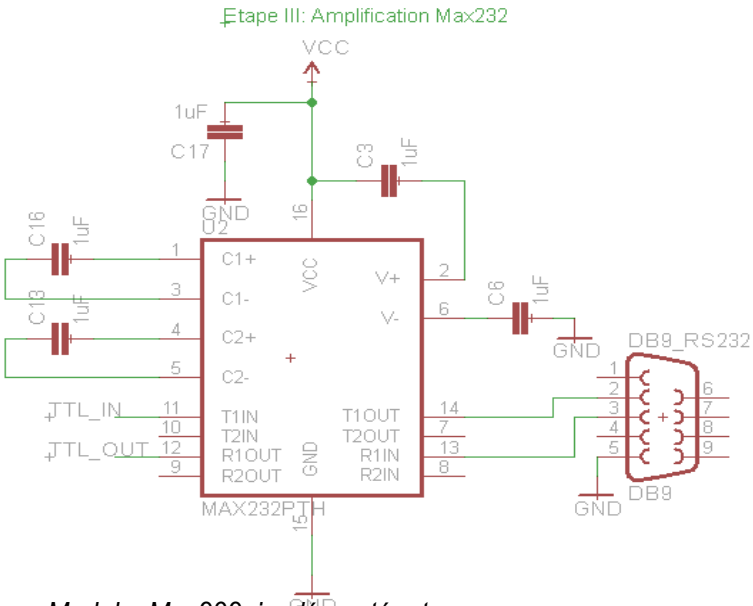
Il faudra prévoir une module dans la carte électronique afin d'adapter le signal avec la PC.

6.1 Module série MAX232

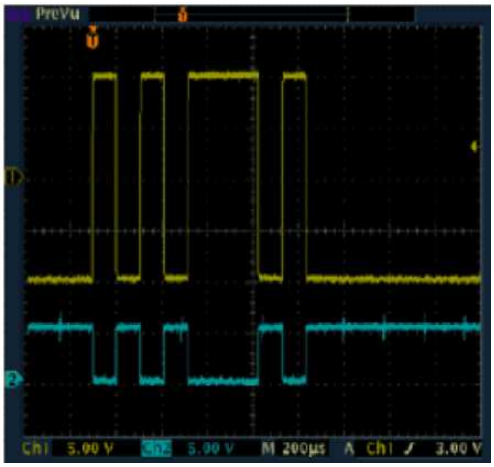
Le module suivant permet d'adapter les signaux de Transmission TX et Réception de Donnes RX du microcontrôleur pour pouvoir envoyer un signal avec le niveau de voltage suffisant pour être lis par la PC.

Du cotée de Microcontrôleur nous aurons :
1 PIN Output TX pour transmettre vers le Max232
1PIN Input RX pour envoyer un commande depuis la PC si c'es nécessaire.

La transmission/Réception se fait avec des sortie numériques TTL 0/5VDC et ils sont transformées par le MAX232 à Transmission /Réception sur +/-12VDC.



Module Max232 implémenté et



conversion entrées TTL 0-5V à +/- 12VDC

Comme nous voyons cette partie consiste à récupérer les entre de données provenant de notre compteur numérique et de les envoyer en série
Du coté de Micro contrôleur nous aurons additionnement
1 PIN Output TX pour transmettre les donnes convertis à série vers le Max232
1PIN Input RX pour envoyer une commande depuis la PC si c'est nécessaire.

6.2 Développement UI en Utilisant Labview

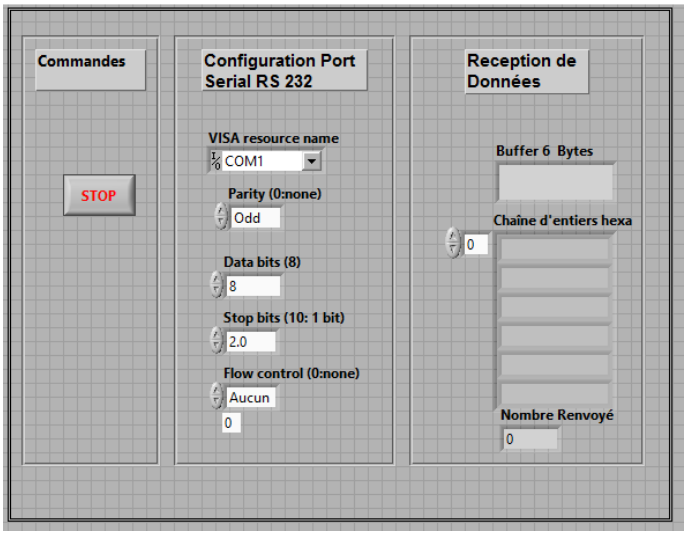
L'objectif de cette étape sera d'afficher les donnes provenant du capteur dans la PC, ça sera la partie finale du système de mesure.

Labview c'est un logiciel de développement qui permet la réalisation des interfaces graphiques, dans notre cas nous avons utilisé les bloques Visa qui gèrent la communication serial RS232
Entre les étapes plus importantes à décrire on peut mentionner :

6.2.1Interface Utilisateur

Dans cette étape nous avons définis un panel de configuration du port série, cette panel permet de spécifier les paramètres suivantes

- Parity : Impair
- Bits de Donnes : 8 bits par trame
- Bits de stop : 2



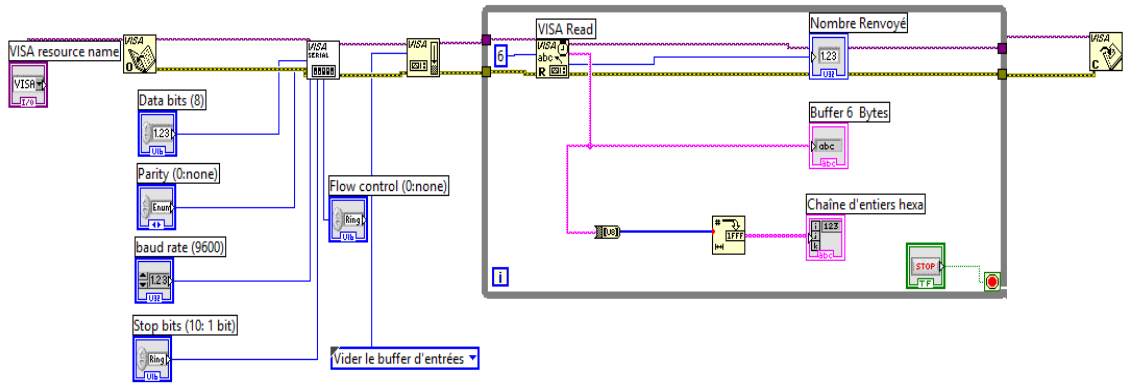
UI Panel pour affichage et commande de transmission série

En plus nous avons définis 2 afficheur, le premier montre la chaîne de caractères enregistré en format ASCII, la suivante c'est un tableau qui permet obtenir la valeur en hexadécimal de chaque octet transmis

Finalement nous ajoutons une commande de stop qui permet de suspendre la communication avec la carte électronique via le port serial COM1

6.2.2Traitement de données en utilisant VISA

Dans cette étape nous avons utilisé la Librairie VISA qui permet de gérer les différentes étapes d'une communication en série, le programme est présenté à continuation

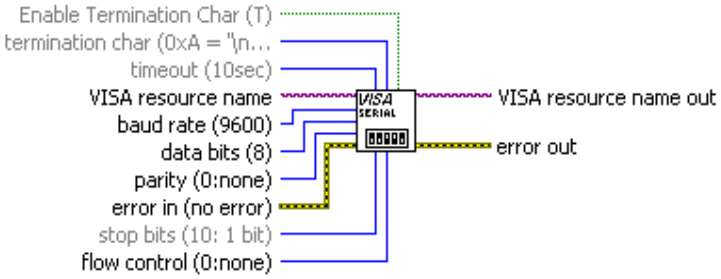


VI d'acquisition de données à travers du Labview

Le programme s'exécute de manière cyclique grâce à une boucle while qui gère la lecture de données, il est composé en plus des éléments suivants :

Visa Serial

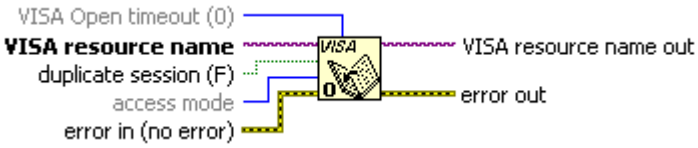
Dans cette partie nous allons définir les paramètres de communication comme le baud rate dans notre cas à 9600, le bit de parity et les bits de stop il faudra bien configurer ces valeurs afin d'obtenir la bonne séquence de trames qui représentent chaque caractère.



Visa Read

Ensuite nous allons utiliser le VI visa Read qui permet de capturer les nombres des octets spécifiées dans la configuration du microcontrôleur, dans notre cas nous allons utiliser 6 Octets pour représenter les 6 chiffres en format BCD.

VISA Close



Finalement nous allons terminer la communication avec la carte en utilisant VISA Close, au début du programme nous avions initialisé l'interface en utilisant le bloque VISA Open.

7. Test et Vérification

Dans cette partie nous allons présenter les tests réalisés avec la carte électronique et le Software Labview afin de vérifier le bon fonctionnement des différents composants de notre système .

7.1Transmission de donnes via Microcontrôleur PC

L'objectif de cet test consiste à vérifier que la carte électronique arrive à convertir les signaux provenant du microcontrôleur en niveau TTL (0/5V) à une valeur adapté pour la réception dans la PC , en plus nous allons vérifier que l'interface utilisateur VISA affiche le données transmis par le microcontrôleur

Afin de réaliser ce test nous commencerons par configurer le registre USART UCSRC qui permet de définir les paramètres suivants :

Parité : Impair Bit UPM1 et UPM2
Stop Bit : 2 Bit USBS
Octets Transmis : 8Bits Bit UCSZ0 et UCSZ1

USART Control and Status Register C – UCSRC

Bit	7	6	5	4	3	2	1	0	
	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	UCSRC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	0	0	0	0	1	1	0	

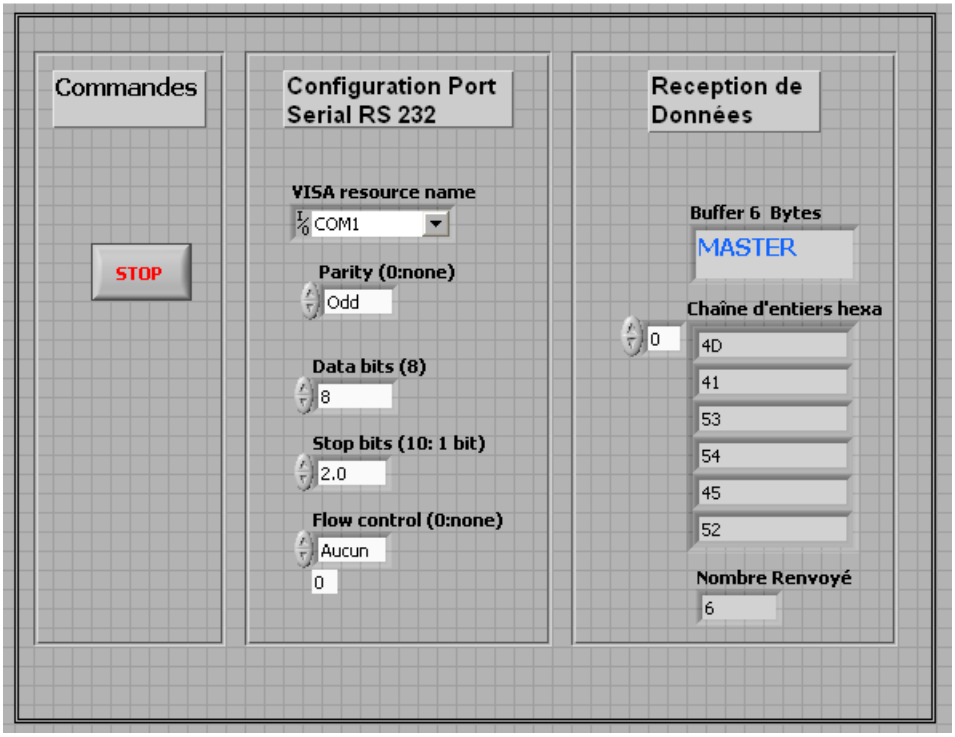
Ensuite nous activons le Bits respectives du registre UCSRC de la manière suivante :

```
void USARTInit(uint16_t ubrr_value)
{
   UBRRH = (uint8_t)(ubrr_value>>8);
   UBRRL = (uint8_t)ubrr_value;

   UCSRC = (1<<URSEL)|(1<<UPM1)|(1<<UPM0)|(1<<USBS)|(1<<UCSZ1)|(1<<UCSZ0);
   UCSRB = (1<<RXEN)|(1<<TXEN);
}
```

Et on ajoute les caractères à envoyer

```
char data[6]={'M','A','S','T','E','R'};

    while(1)
    {
        _delay_ms(250);
        PORTA ^=(1<<PA1);
    for (int i=0;i<6;i++){
        USARTWriteChar(data[i]);
    }
```

Finalement nous utilisons l'application en labview pour commencer l'acquisition de données, Le résultat confirme le bon fonctionnement de la transmission de caractères, ils seront affichées sur l'écran comme nous avons constaté après le test.



Vérification de caractères à travers Labview

Dans l'interface utilisateur présenté au dessus nous voyons la séquence de caractères envoyés depuis le microcontrôleur vers le PORT COM1 du PC en utilisant les modules VISA, cette vérification permet de valider notre système de traitement de signal dans la carte Electronique.

8. Conclusion

Le projet que nous avons réalisé implique la maitrise de différent domaines de l'électronique et l' informatique c'est-à-dire la mécatronique, nous avons développé avec l'aide de notre encadrant une méthodologie systématique de travail qui à permis d'avancer et d'obtenir de résultats mesurables .

La première partie à impliqué l'analyse du système capteur –compteur , cela a été la base pour choisir quelle modules nous avions besoin afin de transmettre les données vers une PC,

Pendant le développement du projet nous avons appris a dessiner des circuit électroniques à travers d' Eagle et à considérer de paramètres clés comme la longueur de pistes , taille de composantes etc.

Dans partie de la programmation nous avons amélioré notre connaissance des différentes fonctionnalités d'un microcontrôleur, d'abord nous avons appris à configurer l'horloge externe de l'Atmega32 afin de pouvoir augmenter la vitesse de lecture de notre système.

Ensuite nous avons appris à configurer l'interface USART du microcontrôleur qui permet l'envoie des données en série ou bien la réception de données.

Finalement nous avons appris à utiliser l'interface Visa de labview qui a permis définir les paramètres nécessaire pour établir une communication serial par exemple bit de stop, bit de signes, baud rates , bit de données afin d'établir une bon connexion avec le Microcontrôleur

9. Bibliographie

- Référence Manuel, 8-bit Microcontrôleur with 32K Bytes ATmega32 ATmega32L
- **Lajara Vizcaino, Jose Elegri Sebastia**, *LabVIEW : Entorno grafico de programacion* , Deuxième édition, 2011.
- **Mike Tooley,** *Electronic Cicuits , Fundamentals and Applications* ,Première Edition 2006
- **Newbiehack LLC.** *Édition Internet*, [En ligne], https://www.newbiehack.com/USARTDetailed.aspx (31 Avril 2015)

10. Annexes

10.1Configuration de Registre Low et High pour modifier la vitesse de horloge

Les registres de fusibles Low et High permettent de modifier la source d'horloge du microcontrôleur , dans notre ca nous utilisons une oscillateur à 4MHZ ainsi la configuration de fusibles devient

#CONFIG 4MHZ

AVRDUDE_WRITE_FLASH =-U lfuse:w:0xFF:m -U hfuse:w:0x99:m -U flash:w:$(TARGET).hex

La valeur 0x99 corresponde à la valeur binaire de **1001 1001** qui sera associe au registre Hfuse suivante

Table 105. Fuse High Byte

Fuse High Byte	Bit No.	Description	Default Value
OCDEN[4]	7	Enable OCD	1 (unprogrammed, OCD disabled)
JTAGEN[5]	6	Enable JTAG	0 (programmed, JTAG enabled)
SPIEN[1]	5	Enable SPI Serial Program and Data Downloading	0 (programmed, SPI prog. enabled)
CKOPT[2]	4	Oscillator options	1 (unprogrammed)
EESAVE	3	EEPROM memory is preserved through the Chip Erase	1 (unprogrammed, EEPROM not preserved)
BOOTSZ1	2	Select Boot Size (see Table 100 for details)	0 (programmed)[3]
BOOTSZ0	1	Select Boot Size (see Table 100 for details)	0 (programmed)[3]
BOOTRST	0	Select reset vector	1 (unprogrammed)

Par contre La valeur 0xFF corresponde à la valeur binaire de **1111 1111** qui sera associe au registre Lfuse suivante

Fuse Low Byte	Bit No.	Description	Default Value
BODLEVEL	7	Brown-out Detector trigger level	1 (unprogrammed)
BODEN	6	Brown-out Detector enable	1 (unprogrammed, BOD disabled)
SUT1	5	Select start-up time	1 (unprogrammed)[1]
SUT0	4	Select start-up time	0 (programmed)[1]
CKSEL3	3	Select Clock source	0 (programmed)[2]
CKSEL2	2	Select Clock source	0 (programmed)[2]
CKSEL1	1	Select Clock source	0 (programmed)[2]
CKSEL0	0	Select Clock source	1 (unprogrammed)[2]

Chaque bit du registres fuses permet de spécifier la vitesse de l'oscillateur externe raccordé à notre CI.