République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et
de la Recherche Scientifique

UNIVERSITE DE BLIDA

INSTITUT D'AERONAUTIQUE

Mémoire de fin d'études

pour l'obtention du Diplôme d'Etudes Universitaires Appliquées (DEUA)

En **Aéronautique** *Option :* **Avionique.**

Thème

Simulation d'un oscilloscope deux voies sur P.C

Présenté par :

- HAMIANE MOHAMED
- OMARI KAMEL

Promoteur :

M. Belmechri

Co-Promoteur :

M. BENOUARED

Promotion 2000/2001

Dédicaces

Je dédie ce modeste travail à :

Mes chers parents.

A mes sœurs et frères

A mon oncle Kaci, sa femme et ses enfants.

A mes amis : Dj. Omar, F. Hamid, L. Fateh, K. Mohamed, H. Aissam, Zizi Achour, M. Ali, Moh Azefoune, Moh Abizar, Ami Saïd, M. Hakim, B. Cherif, Ami Cherif, A. Ahmed, S. Kamel.

A mes amis de la promotion 2000/2001 Avionique :

Mon collègue Mohamed, Hamiane, Mourad Brahmi, S. Farid, K. Hichem, Mahmoud, Mouna, T. Mohamed, Rafik, ...

Je dédie spécialement ce mémoire à celle qui a illuminé (réjouit) mes journées à Blida (! ! !).

A la mémoire de tous ceux qui sont tombés pour la liberté et la démocratie.

Kamel

Dédicace

Je dédie cet honorable travail tout d'abord à ma très chère maman qui m'a donné la force et le courage durant toute ma vie ainsi que sa patience et son attention qui n'a jamais cessé mais aussi à toutes les personnes que j'aime et avec lesquelles j'ai beaucoup appris et passé beaucoup de moments inoubliables.

Très chère maman, toi qui m'as élevé et toujours écouté, je prie le Bon Dieu de te garder en très bonne santé.

Très chers amis et frères, et mes sœurs, l'union fait la force, tel est notre principe.

Ainsi je dédie ce mémoire à :

Farid (M), mon collègue Kamel, Mourad, Hicheme, Mohamed et Abed.el. Ghani.

Mohamed

Remerciements

On tient à remercier vivement :

Notre promoteur M. Belmechri pour son encadrement et ses conseils.

A tous ceux qui ont contribué de loin ou de près pour la réalisation de notre travail surtout Malika.

Sommaire Général

Introduction générale :

Le projet qu'on vient de présenter est un oscilloscope « 2 voies ». Notre oscilloscope se compose donc d'un boîtier de conversion analogique-numérique sur lequel viennent se raccorder les fiches BNC des sondes, ainsi que le câble de liaison vers le connecteur DB25 de la sortie imprimante parallèle du PC.

Ce boîtier de conversion analogique-numérique permet de rendre compte sur l'écran du PC des variations de tension présentes sur deux entrées. Ceci posé, ce boîtier de conversion donne entière satisfaction pour l'observation de variations d'une tension de 0 à 5V maxi, aux fréquences comprises entre 0 et 10000 Hz.

Cet oscilloscope « 2 voies » est destiné à fournir une approche simple est économique du problème de la visualisation sur un PC de variations d'une grandeur électrique de type analogique.

Notre projet est réparti sur trois chapitre :

Le premier chapitre étudie les différents types d'interfaces entre PC et périphériques.

Le deuxième entame la conversion analogique-numérique.

Enfin, le troisième expose la partie réalisation matérielle et logiciel dont l'interface graphique est réalisée sous Windows 95, avec la version 2.0 de Delphi.

CHAPITRE I

LES INTERFACES

I-1-INTRODUCTION :

Le dialogue entre un ordinateur et un tel périphérique implique un échange d'informations ; ce dernier est assuré par l'intermédiaire d'une interface. Une interface est définie comme un endroit où des systèmes indépendants se rencontrent et réagissent l'un sur l'autre ou communiquent entre eux. L'interface a généralement la forme d'une carte de circuit enfichée dans le bus du micro-processeur. Le connecteur enfiché dans le bus permet à l'interface d'avoir accès aux signaux du micro-processeur. A l'autre extrémité de la carte il y'a un autre connecteur. Un câble relie ce connecteur au périphérique.

I-2-CARACTERISTIQUES D'UNE INTERFACE :

L'interface prend en charge de coordonner le transfert de données entre l'ordinateur et un tel périphérique. Une telle interface doit présenter certaines caractéristiques :
a)-communiquer à l'ordinateur l'état du périphérique chaque fois nécessaire, donc elle dispose d'un registre d'état.
b)- elle doit avoir un registre Tampon de données pour pouvoir transférer des données entre l'ordinateur et le périphérique.
c)-Reconnaître l'adresse du périphérique quand cette adresse apparaît sur les lignes d'adresses.
d)-Fournir les signaux nécessaires d'horloge et de contrôle des ports pour le transfert de données ou de l'information voulue.

I-3-TYPES D'INTERFACES :

On distingue trois types d'interface: parallèle, série, et programmable

I-3-1-Interface parallèle :

Les interfaces de types parallèles sont les moins complexes ; certaines interfaces parallèles peuvent être constituées d'un seul circuit intégré TTL .
Les interfaces parallèles peuvent être classées selon deux critères :
Premier critère : le nombre de bits transmis en parallèle par l'interface (largeur de canal donnée).
Second critère : le type d'asservissement utilise pour le transfert de ces bits entre l'ordinateur et le périphérique.
Les largeurs de canal varie d'un seul bit à 16 ou 32 bits, voire plus. La taille la plus courante, pour les processeurs à 8bits, est un canal de 8 bits. Cette largeur permet au processeur de transmettre par l'interface un mot de donné entier à chaque opération. (figI-1)
L'interface parallèle à 8 bits est d'autant plus intéressante que c'est aussi la taille la plus courante pour les processeurs à 16 bits. Ceci pour deux raisons. Tout d'abord

parce qu'il existe un grand nombre de périphériques, telles les imprimantes, qui ont été initialement conçues pour les micro-ordinateurs à 8 bits. Ensuite parce que l'ASCII, code caractères le plus courant ; exige une interface d'au moins 7 bits, que l'on arrondit habituellement à 8 bits.
Le second critère utilisé pour classer les interfaces est le type d'asservissement employé pour faire circuler l'information, sur les lignes de données. il existe des asservissements à zéro fil, à un fil, à deux fils et à trois fils.

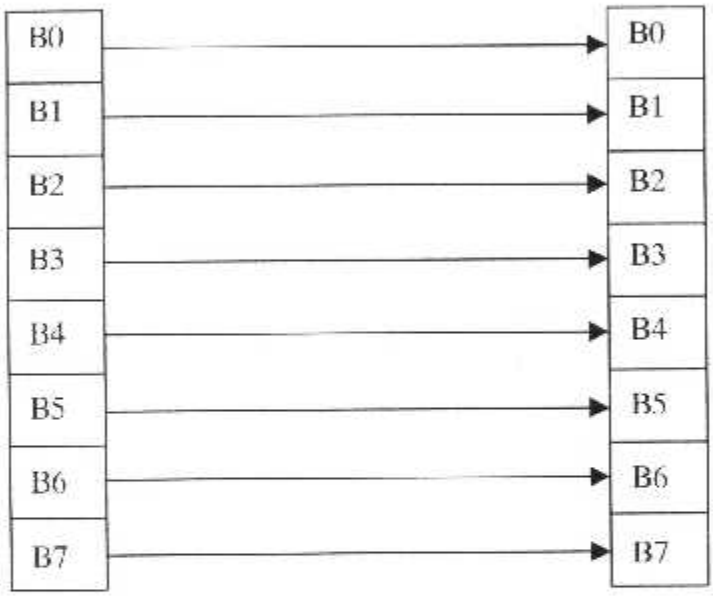


Fig I-1. Transmission de données en parallèle

I-3-2 : Interface série :

Lorsque les distances entre un ordinateur et son périphérique deviennent grandes, le coût d'une liaison de plusieurs fils en parallèles devient prohibitif (excessif, élevé). L'interface série fournit une solution à un coût plus faible. La principale interface série utilisée pour relier les micro-ordinateurs aux périphériques est la norme RS 232 C.
Le principe de base de la transmission de données série est le transfert des données sur un seul fil. (fig. I- 2) .

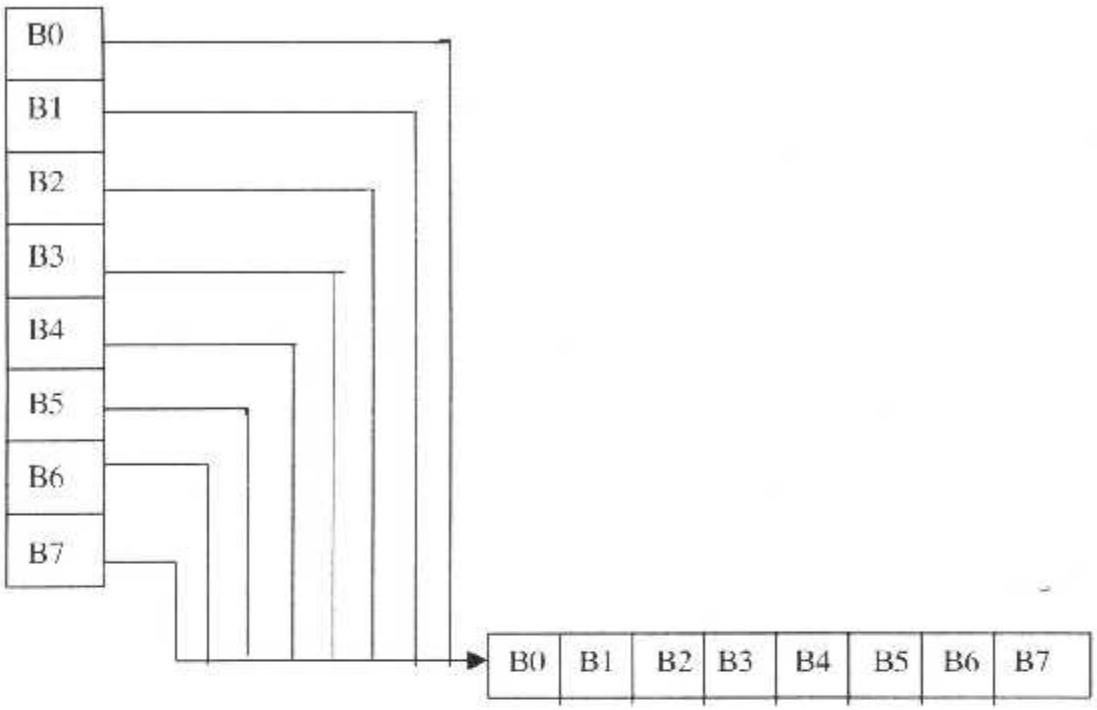


Fig. I-2 Transmission de données en série.

Il existe trois modesfondamentaux de transmission série : simplex, semi-duplex et duplex.

La liaison simplex caractérise une liaison laquelle les données circulent dans un seul sens , c'est à dire de l'émetteur vers le récepteur . Ce genre de liaison est utile lorsque les données n'ont pas besoin de circuler dans les deux sens (par exemple de l'ordinateur vers l'imprimante ou de souris vers l'ordinateur) (voir fig. I-3).

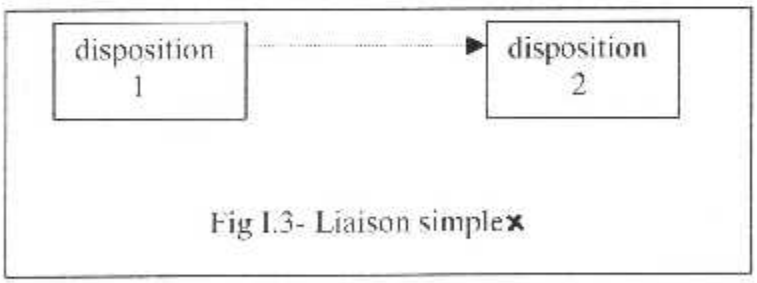


Fig I.3- Liaison simplex

La liaison half-duplex (parfois appelée liaison à l'alternat ou semi-duplex) caractérise une liaison dans laquelle les données circulent dans un sens ou l'autre ,

mais pas les deux simultanément . Ainsi ,avec ce genre de liaison chaque extrémité de la liaison émet à son tour (fig.I-4)

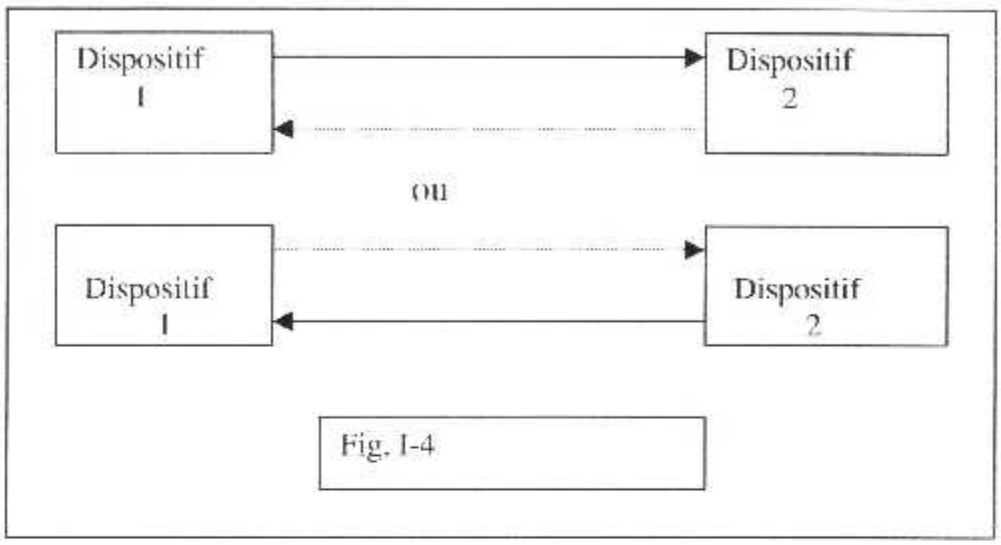


Fig. I-4

La liaison full-duplex (appelé aussi duplex intégral) caractérise une liaison dans laquelle les données circulent de façon bidirectionnelle et simultanément .Ainsi, chaque extrémité de la ligne peut émettre et recevoir en même temps. (voir fig.I-5)

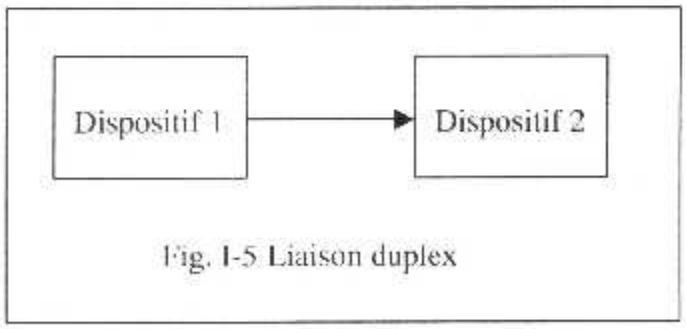


Fig. I-5 Liaison duplex

On distingue deux types fondamentaux de transmission séric : synchrone et asynchrone.
Etant donné les problèmes que pose la liaison de type parallèle, c'est la liaison série qui est la plus utilisée. Toutefois, puisqu'un seul fil transporte l'information, il existe un problème de synchronisation entre l'émetteur et le récepteur, c'est à dire que le récepteur ne peut pas à priori distinguer les caractères (ou même de manière plus générale les séquences de bits) car les bits sont envoyés successivement . Il existe donc deux types de transmission permettant de remédier à ce problème :

La liaison asynchrone, dans laquelle chaque caractère est émis de façon irrégulier dans le temps (par exemple un utilisateur envoyant en temps réel des caractères saisis au clavier). Ainsi, imaginons qu'un seul bit soit transmis pendant une longue période ...le récepteur ne pourrais savoir s'il s'agit de 00010000 ou 10000000 ou encore 00000100 ...
Afin de remédier à ce problème, chaque caractère précédé d'une information indiquant le début de la transmission de caractère (l'information de début d'émission est appelée bit SART) et un bit de fin de transmission (appelée bit STOP il peut éventuellement y avoir plusieurs bits STOP).

La Liaison synchrone, dans laquelle l'émetteur et le récepteur sont cadencés à la même horloge. Le récepteur reçoit de façon continue (même lorsqu'un bit n'est transmis) les informations au rythme où l'émetteur les envoie. C'est pourquoi il est nécessaire qu'émetteur et récepteur soient cadencés là même vitesse. De plus, des informations supplémentaires sont insérées afin de garantir l'absence d'erreurs .
L'ors de la transmission synchrone, les bits sont envoyés de façon successive sans séparation entre chaque caractère, il est donc nécessaire d'insérer des éléments de synchronisation, on parle alors de **synchronisation au niveau caractère.**
Le principal inconvénient de la transmission asynchrone est la reconnaissance des informations au niveau du récepteur, car il peut exister des différences entre les horloges de l'émetteur et du récepteur. C'est pourquoi chaque envoi de données doit se faire sur une période assez longue pour que le récepteur la distingue.

I.3.3. INTERFACE PARALELLE CENTRONICS :

L'interface parallèle est l'interface la plus commode pour un transfert d'information entre un PC et une imprimante.
L'une des interfaces à être largement utilisée dans l'industrie, est un protocole développé par Centronics (constructeur d'imprimantes). Elle est matérialisée par une fiche DB25, qui est adaptée pour tous les ports du PC.

I.3.3.1. Brochage :

Le brochage du port parallèle est illustré par la figure I - 6.

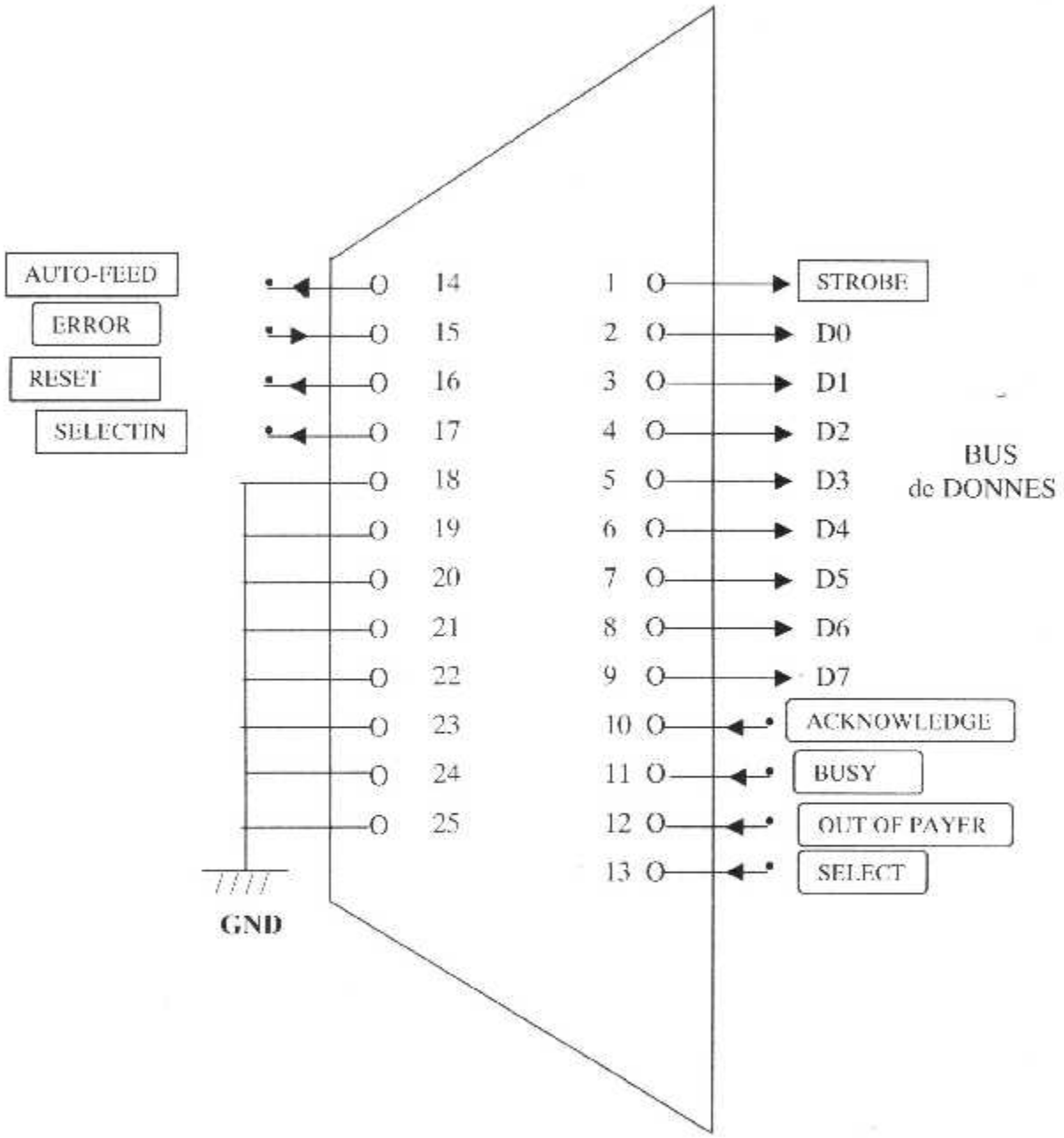


Figure : I-6 Le brochage de la fiche DB25

: 4 Commandes en Sorties

: Etats en entrée, permettent à l'ordinateur de connaître l'état de l'imprimante,
Bus de données : 8 données en sortie, utilisées pour envoyer le code caractère à imprimer.

l'adresse du port imprimante correspond généralement à LPT 2.
Pour un autre port, on se réfère au tableau suivant :

Fonction du port	LPT 1		LPT 2		LPT 3	
	Décimal	Hexadécimal	Décimal	Hexadécimal	Décimal	Hexadécimal
Données	956	3BC	888	378	632	278
Etat	957	3BD	889	379	633	279
Contrôle	958	3BE	890	37A	634	27A

I-3.3.2. Description des signaux caractéristiques de l'interface Centronics :

PIN1 : STROBE : est une sortie active à l'état haut ,indiquant à l'imprimante que les lignes de données portent des informations valables.

PINS 2 à 9 : Bits de données D0. à D7 (DATA en sortie) : l'octet sera transmis vers l'imprimante (par exemple un caractère à imprimer).

PIN 10 : réponse (ACKNOWLDGE) : cette ligne est une entrée activée à l'état bas. Elle signifie que l'imprimante est prête recevoir le donnée qui suit.

PIN 11 : pas libre (BUSY) : Cette ligne est une entrée active à l'état haut. Elle signifie que l'imprimante n'est pas libre (occupée).

PIN 12 : plus de papier (PATER EMPTY) : cette ligne est une entrée activée à l'état haut. Elle signifie que l'imprimante n'a plus de papier.

PIN 13 : SELECTOUT : Cette ligne est une entrée active à l'état haut. Elle signifie que l'imprimante est prête à imprimer.
PIN 14 : Saut de ligne automatique (AUTOFEED) : Cette ligne est une sortie active à l'état bas.

PIN 15 : ERROR : Cette ligne est une entrée active à l'état haut, signifie que l'imprimante vient de rencontrer une erreur.

PIN 16 : RESET : Cette ligne est une sortie active à l'état haut,, signifiant l'initialisation de l'imprimante.

PIN 17 : SELECTIN : Cette ligne est une sortie active à l'état bas signifiant de mettre l'imprimante en service.
PINS 18 à 25 : GND : C'est la masse des circuits de l'imprimante. Elle doit être connectée à celle du P.C afin de valider les signaux.

I.3.3.3. Chronogramme de l'interface Centronics :

STROBE, est un signal venant du PC signifiant que les données sont valides. Les informations sur les lignes données doivent être valables au moins 1 µS avant l'émission de STROBE en logique inversée et doivent rester valables pendant au moins 1 µS, après disparition de STROBE. La durée du signal STROBE peut être comprise entre 1 et 500 µS. Le retard entre la disparition de STROBE et l'apparition de ACK est de l'ordre de 2 ,5 à 10 µS. Les imprimantes Centronics restent « pas libre (BUSY) » pendant des durées allant de 2 à plus de 300 US.

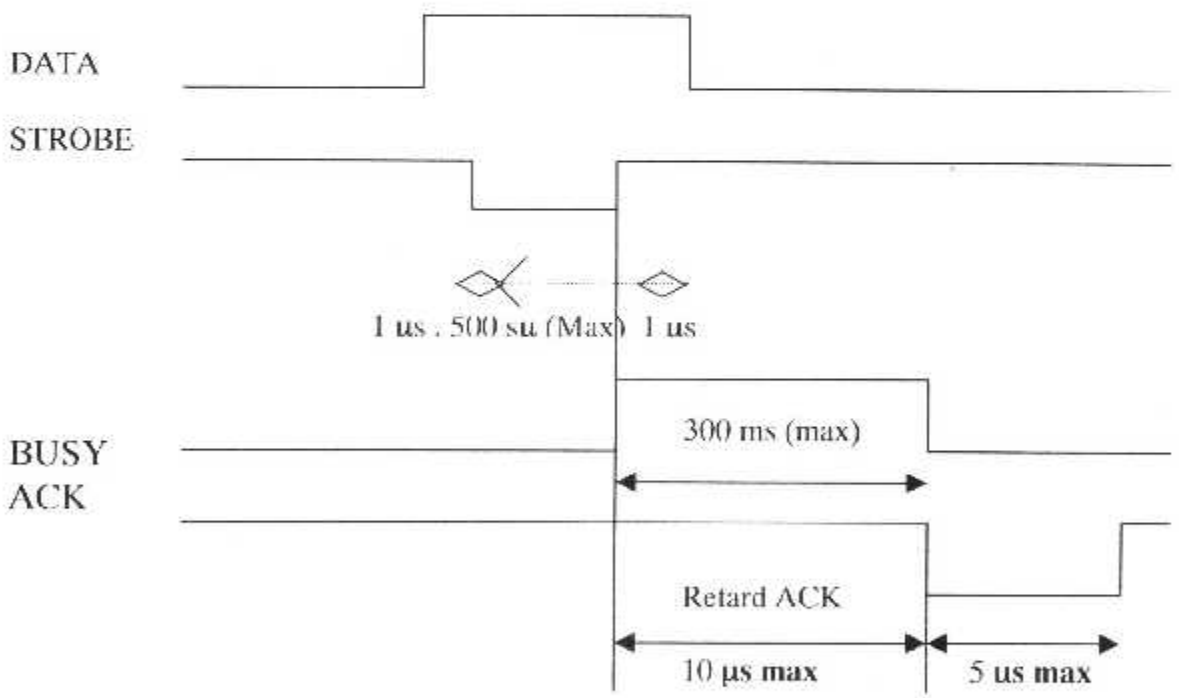


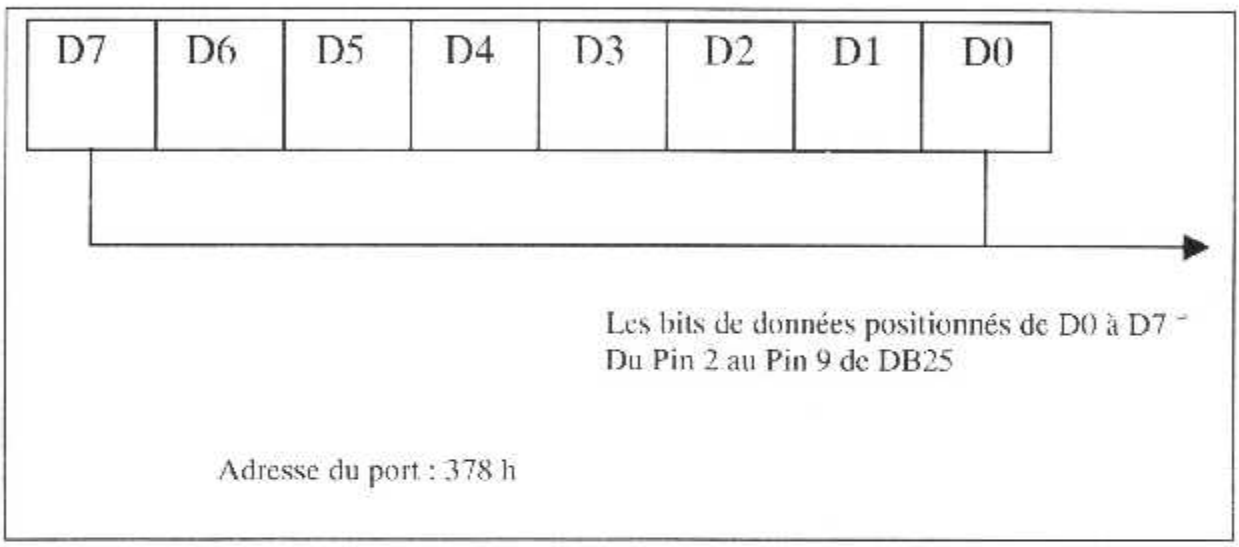
Fig :I-7 : **chronogramme de l'interface centronics**

I.3.3.4. Les registres du port parallèle :

Le port parallèle DB25 comporte 3 registres :

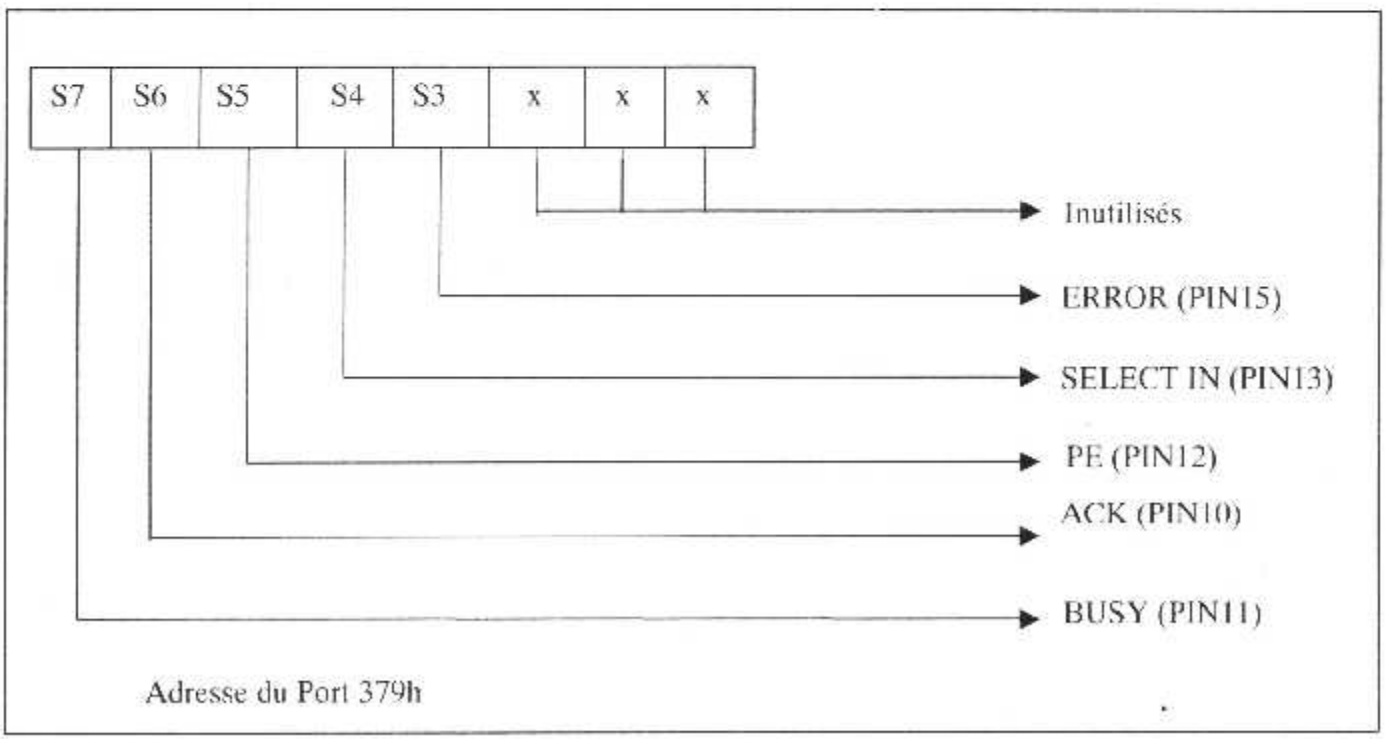
a)Registre de données :

C'est un registre de sortie de 8 bits situé à l'adresse 378h qui exprime les données qui seront transmis sur les lignes DO à D7.



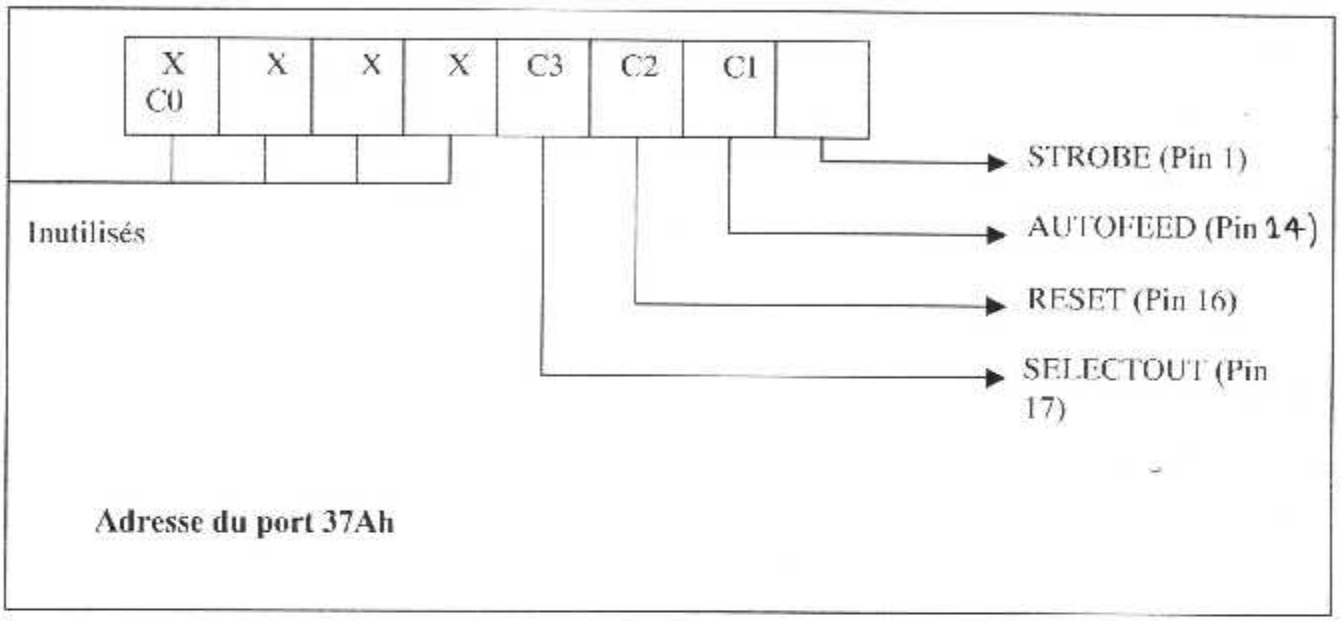
b)Registre d'état :

C'est un registre d'entrée qui donne l'état de l'imprimante.



c)Registre de commande :

c'est un registre de sortie qui sert à commander l'imprimante.



I.3.4.Interface programmable : (Ports entrée/ sortie Parallèle)

Un certains nombre d'interfaces sont destinées à envoyer ou à recevoir des groupes de 8 bits en parallèle. Ces interfaces ont reçu le nom de PIO (Programmable Input Output Device), PPI (Programmable Peripheral Interface), PIA (peripheral interface adapteo)

Le PIA se présente sous forme d'un circuit intégré à 40 broches (voir figure I-8 : le brochage du PIA6520).

Huit broches serviront à revoir les données venant de microprocesseur ou à les transmettre à celui-ci (bornes D0 à D7).

Vers l'extérieur deux groupes de huit broches peuvent être relier à deux périphériques différents et constituent ce que l'on appelle les ports d'entrée/sortie (PA pour le port A et PB pour le port B).

En effet, tout comme les ports sont destinés à charger ou décharger des marchandises, chacun de ces ports d'entrée/sortie pourra, à la demande, recevoir ou envoyer des informations. Bien mieux pour chaque port, c'est chaque borne qui peut être « programmée » pour être réceptrice ou expéditrice. Pour cette programmation, chaque port dispose d'un registre de direction, registre à huit cellules permettant de programmer chaque borne d'entrée/sortie par l'intermédiaire d'un jeu de portes « 3états » (voir figure I-9).

Si S=0, la circulation des informations se fera de PA vers D et l'inverse est vrai.

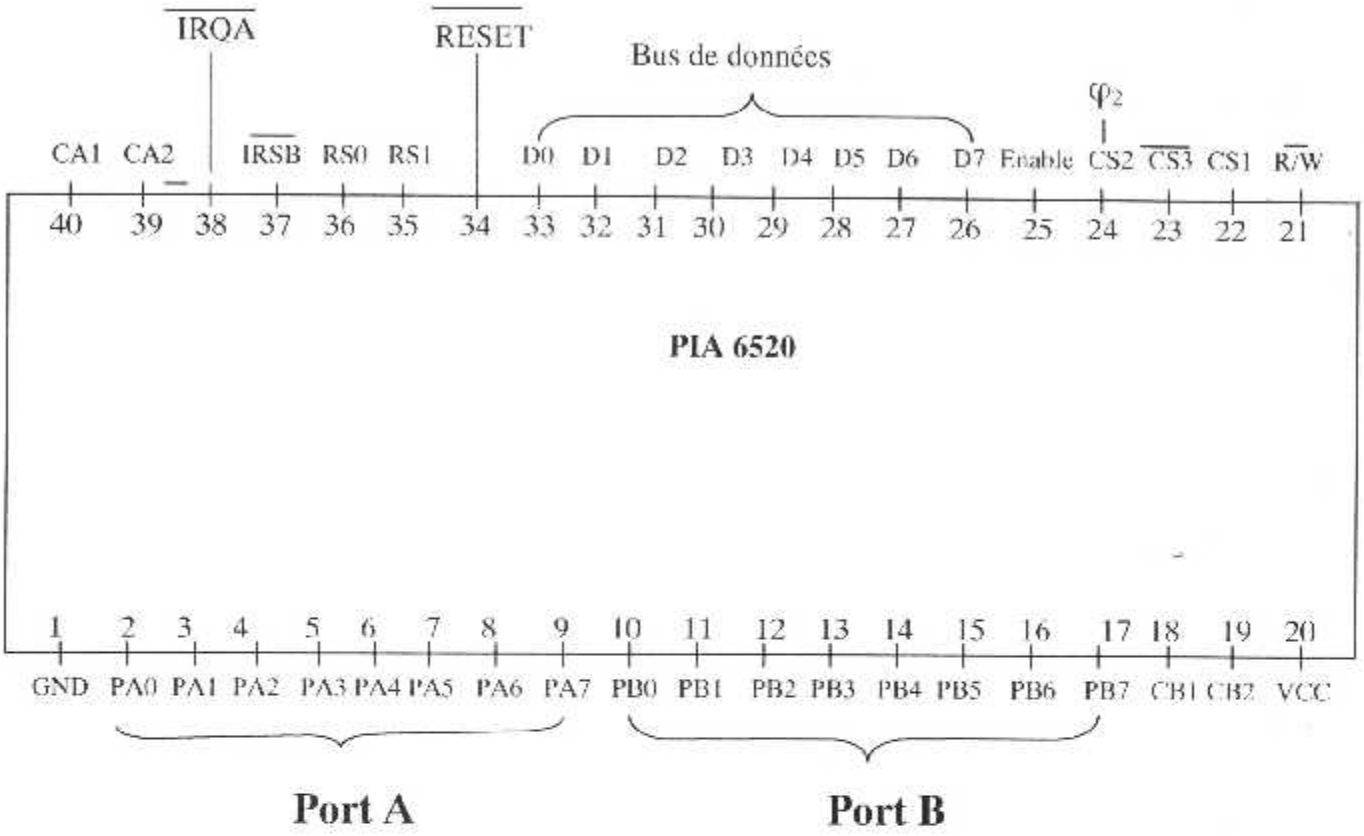


figure I-8 : **le brochage du PIA6520.**

Le PIA comporte aussi deux registres tampon, RA et RB, un pour chaque port, permettant de stoker l'information entre le microprocesseur et les périphériques.
Enfin chaque port est doté d'un troisième registre qui est le registre de contrôle. Ce registre régit les commandes internes de PIA suivant les ordres reçus de l'extérieur (périphérique) ou du microprocesseur.

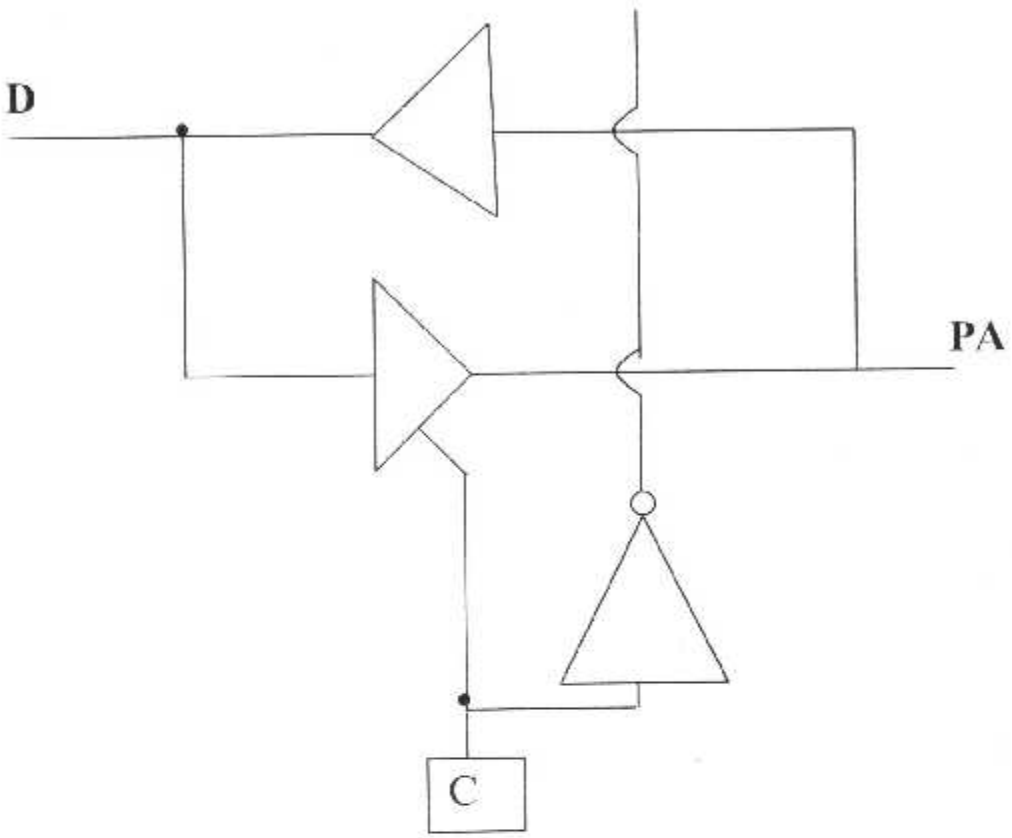


Figure I-9 **jeu de portes**

CHAPITRE II

LES CONVERTISSEURS

II.1 INTRODUCTION

L'électronique est divisée en deux domaines distincts :

-Le domaine analogique, où les variables peuvent prendre une infinité de valeurs différentes ; les signaux varient continûment. Tous les signaux issus des capteurs sont analogiques, et traduisent des phénomènes physiques qui varient continûment.

-Le domaine numérique, où les variables prennent uniquement deux états, un état haut et un état bas.

Le domaine numérique est maintenant prédominant. Il s'est beaucoup développé grâce aux progrès faits par les microprocesseurs. Beaucoup de signaux naguère traités de façon analogiques le sont aujourd'hui par programmation de microprocesseurs.

Le gros avantage apporté par la numérisation des signaux est la possibilité de stockage, de transformation et de restitution des données sans qu'elles ne soient altérées.

Le traitement des données par programmation introduit aussi une souplesse dans la conception de produits à base d'électronique : un même circuit électronique à base de P pourra traiter des signaux différents ; seul le programme va changer. Cela permet de réduire les coûts par standardisation, la même carte étant utilisée pour plusieurs fonctions différentes. L'électronique analogique nécessitait au mieux un changement des composants, au pire, la conception d'une nouvelle carte.

Mais, à la base, les signaux ont toujours une nature analogique. Le domaine analogique va donc toujours exister au moins en amont de toute chaîne de traitement. Parfois, on a aussi besoin d'un signal analogique en sortie de cette chaîne de traitement : il faudra alors reconvertir les données numériques en signal analogique.

Le passage d'un type de donnée à l'autre se fera par des convertisseurs, composants « mixtes » qui vont manipuler des tentions analogiques en entrée et des signaux logiques en sortie ou vice versa.

Il existe deux catégories de convertisseurs :

-Les convertisseurs Analogiques-numériques (CAN, ADC en anglais, pour analog to digital converter), qui vont transformer les tensions analogiques en signaux logiques aptes à être traités par microprocesseur (numérisation des signaux).

-Les Convertisseurs Numériques-Analogiques (CNA, DAC en anglais, pour digital to analog converter) qui vont convertir les signaux logiques en tensions analogiques.

Plusieurs types de convertisseurs sont disponibles dans chaque catégorie, qui se différencient par leur précision, leur vitesse de traitement de l'information, leur prix...

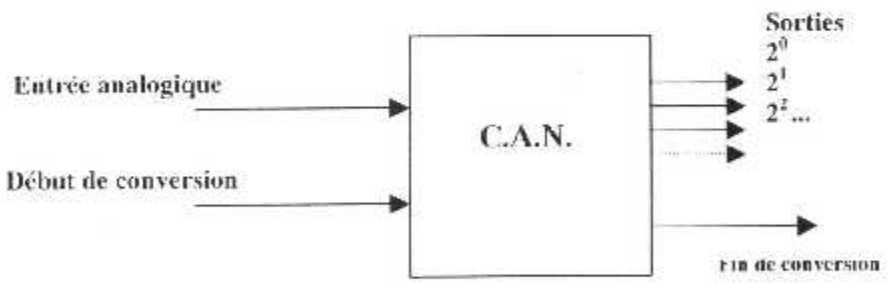
Il n'y a pas « le » convertisseur à tout faire qui soit bon partout : on devra faire un choix en fonction de ses besoins.
On s'intéresse dans ce chapitre à la conversion analogique-numérique.

II-2 Conversion analogique – numérique.

II-2-1 Définition :

Un convertisseur analogique – numérique transforme une grandeur physique (tension, courant) en une valeur numérique.
Généralement, elle possède :
-Une entrée « début de conversion » qui permet de démarrer la conversion (Start) ;
-Une sortie « fin de conversion » qui indique que la conversion est terminée (End) ;
-Une entrée analogique (courant ou tension) ;
-plusieurs sorties numériques, dont le nombre est fonction de la résolution.

Le CAN est utilisé généralement chaque fois que l'information doit subir une des opérations suivantes :
*Visualisation numérique ;
*Traitement numérique ;
*Transmission numérique.



II-2-2 généralités

Un convertisseur analogique/numérique (CAN) transforme un signal analogique en un signal numérique.
La plupart du temps, ce signal est une tension électrique image d'une grandeur physique (via un capteur).
Le nombre de sortie est, le plus couramment, codé en binaire sur n bits.
La **résolution** d'un CAN est la même que celle d'un CNA, soit

$$\mathbf{r = 1/ 2^n - 1 \quad \approx 1/2^n}$$

La plus petite valeur non nulle que peut sortir le CAN est 1.
La plus grande valeur (pleine échelle) que peut sortir le CAN est 2^{n-1}.
On appelle également **quantum** la plus petite valeur analogique que peut discriminer le CAN.La valeur maximale que le CAN peut admettre est donc $(2^{n-1}).q$
Tout CAN nécessite une valeur de tension « étalon » pour effectuer la conversion, qui lui sert de référence. Notons V_{ref} cette tension.
Soit Va la tension présente à l'entrée du CAN. Si V_{ref} est la tension de référence, alors le nombre N vaut :

$$N = ENT\ (K.\ V_a / V_{ref})$$

Note : ENT = partie entière de ...
Le coefficient K est une constante entière, souvent égale à 2^n.
Si le CAN est unipolaire, le domaine de tensions analogiques convertibles est donc :

$$0 \leq V_A < V_{ref}$$

Exemple, si on a un CAN 8 bits – 5 volts, avec une tension de référence de 5 volts.
La plus petite tension que l'on peut discriminer à l'entrée est

$$\mathbf{q = V_{ref} / K = V_{ref}/2^n = 5/256 \approx 0{,}019V}$$

Cette grandeur représente la **précision** de CAN

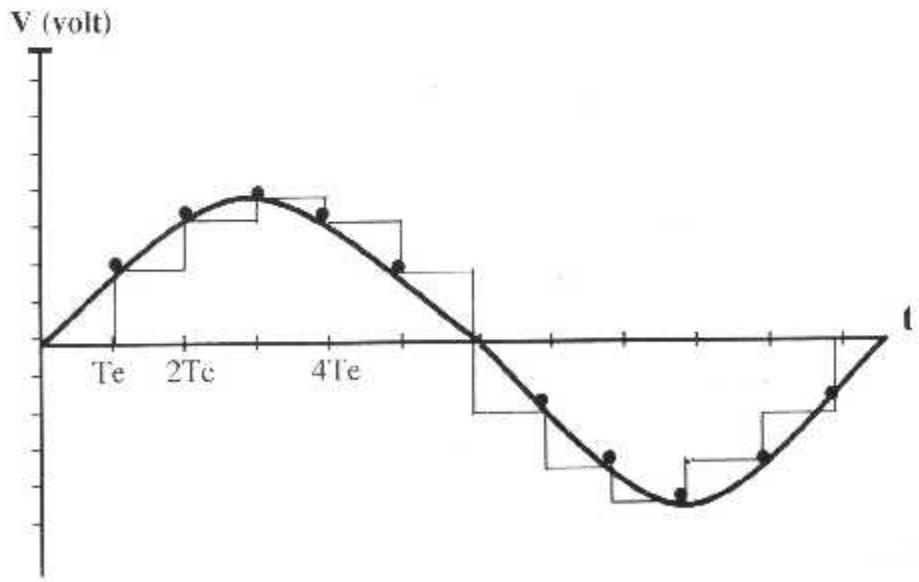
II-2-3 Résolution et dynamique

Du nombre de bits utilisés dans le CAN dépend de la **résolution** et la **dynamique** du signal.
Une résolution basse (faible nombre de bits) imposera une faible dynamique du signal, et une différence importante entre deux valeurs numérique proches.

Une résolution haute (grand nombre de bits) conduira à une grande dynamique du signal.

II-2-4 Notion d'échantillonnage :

Une conversion A/N se passe en deux parties : il faut en effet **quantifier** le signal en amplitude et en temps.
La quantification en temps n'est pas indispensable ; si le signal analogique d'entrée est continu ou à évolution lente en fonction du temps, la durée de la conversion est négligeable. On a en outre nul besoin de conserver d'information temporelles.
Si par contre, le signal d'entrée en fonction du temps (par exemple, un son) et que son information est continue également dans la fréquence de ce signal, il importe de quantifier également en temps le signal avant de le convertir. On appelle cette fonction l'**échantillonnage.**
Pour échantillonner un signal, on définit une durée, appelée **période d'échantillonnage,** qui est l'intervalle de temps entre deux valeurs converties. Le signal d'entrée n'est pas examiné entre ces deux valeurs, et n'est donc pas convertit. Cette période doit être choisie suffisamment courte pour que l'échantillonnage soit significatif. Elle ne doit pas non plus être exagérément petite, afin que la quantité d'informations ne soit pas trop importante.
Le circuit assurant cette fonction est généralement extérieur au CAN, et est appelé **échantillonneur-bloqueur**, puisqu'il doit conserver (bloquer) pendant la période d'échantillonnage la valeur du signal d'entrée. Il réalise donc une quantification du temps.



Echantillonnage d'une sinusoidale

A la sortie de l'échantillonneur-bloqueur, le signal est encore analogique, et continu en amplitude. Il s'agit encore d'une tension (en volts) qui peut prendre des valeurs quelconques. Il est ensuite numérisé par le CAN. A la sorite, le signal est quantifié en temps et en amplitude. Il n'est défini qu'aux instants d'échantillonnage.
Le choix de la période d'échantillonnage est crucial : un sous-échantillonnage détériora trop le signal d'entrée, alors qu'un sous-échantillonnage va augmenter le volume de données à traiter.
Il existe un théorème, appelle théorème de shanon, qui fixe la limite inférieure absolue de la fréquence d'échantillonnage. Il montre que la fréquence d'échantillonnage doit être **supérieure ou égale au double** de la plus haute fréquence contenue dans le signal d'entrée afin de pouvoir reconstituer fidèlement le signal. En général, pour s'assurer de cette condition, on applique un filtre passe-bas avant d'échantillonner le signal.
Par exemple, si l'on décide d'échantillonner de la musique pour faire un CD audio, on considère que l'oreille humaine n'entend pas les sons de fréquences supérieure à 22KHz.On filtre donc le signal à 22KHz pour se débarrasser des fréquences inaudibles (mais présente à priori dans le signal) et on échantillonne à 44KHz.

II-3- Erreurs de la conversion A/N :

II-3-1 Erreur de quantification. Amélioration.

La double quantification, dans le temps et en amplitude consistait en une perte d'information du signal. Ceci nous conduit à la notion d'erreur de quantification, qui est inhérente à la conversion analogique/numérique (et inverse), et sera présente même si les convertisseurs sont considérés comme parfaits. Cette erreur systématique s'ajoute donc aux erreurs qui seront décrites.
Si on numérise une rampe de tension, l'erreur entre la tension d'entrée et la tension de sortie « reconstituée » (reconvertie en analogique par passage dans un CNA) aura la forme suivante :

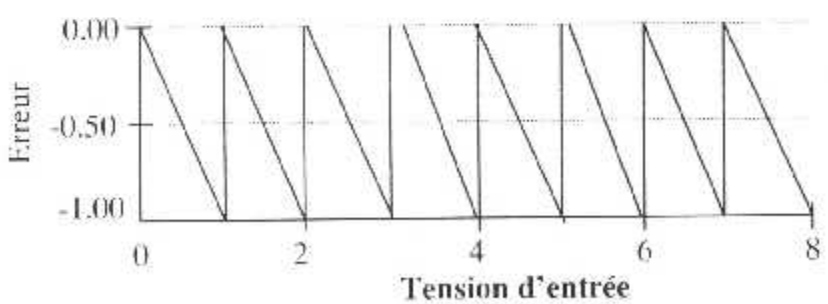


Figure II.1 **Erreur de quantification d'un CAN**

L'erreur est toujours négative (valeur par défaut), et oscille entre 0 et –1 LSB (à – 1V ici).
Il serait souhaitable d'avoir plutôt une erreur centrée autour de 0, de manière à quantifier tantôt par excès, tantôt par défaut ; en effet, en quantifiant systématiquement par défaut, on introduit un offset dans le signal numérisé.

Pour pallier cet inconvénient, on introduit un décalage au niveau du premier LSB du convertisseur, comme indiqué sur la figure II-2 : la première transition n'a pas lieu pour 1 LSB, mais pour ½ LSSB seulement, ce qui fait que jusqu'à une valeur d'entrée inférieure à ½ LSB, on quantifie par défaut, et entre ½ et 1 LSB, on quantifie par excès.

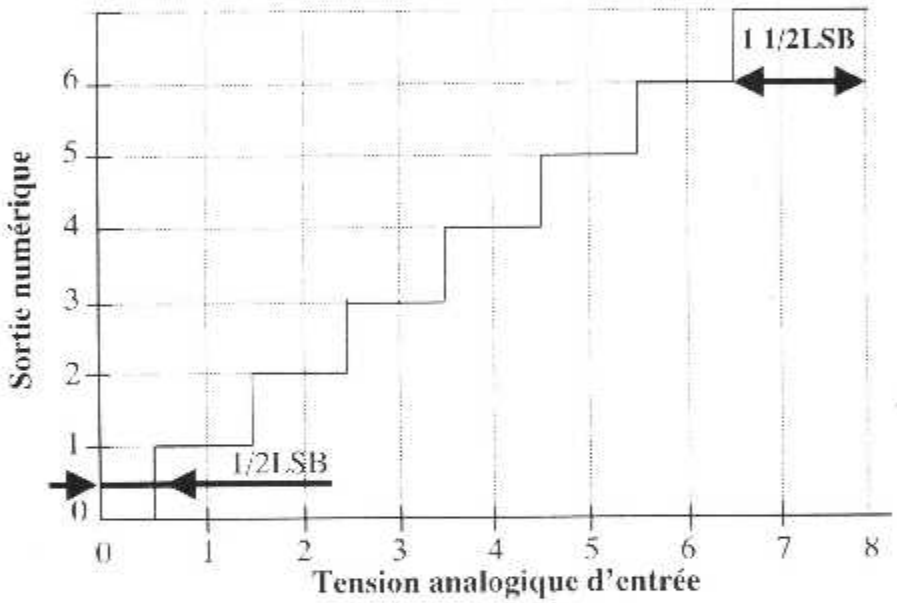


Figure II-2 **Fonction de transfert d'un CAN 3 bits corrigé.**

L'erreur obtenue devient celle de la figure II-3 : elle est symétrique par rapport à 0 et égale ± ½ LSB.

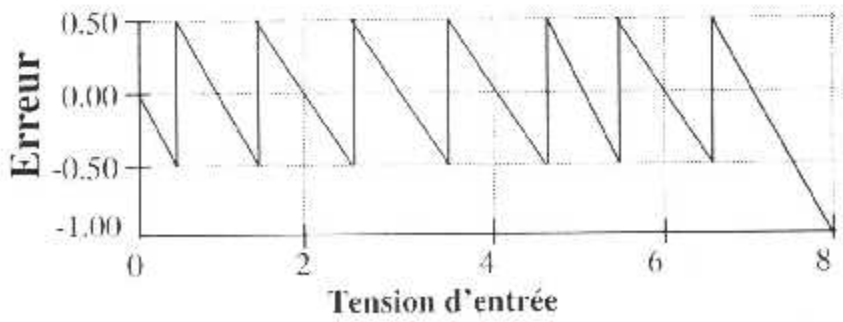


Figure II-3 **Erreur de quantification symétrique**

IL y'a juste une exception: Le ½ LSB tronque au début va se retrouver en bout d'échelle (voir figure II-2) :le dernier état numérique correspondra à une plage d'entrée analogique valant 1 ½ LSB. L'erreur de quantification sera plus grande sur cette plage ,ce qui n'est d'ailleurs pas grave.

En pratique la majorité de CAN ont une fonction de transfert décalée pour assurer une erreur de quantification symétrique.

Il faudra toutefois s'en assurer en lisant la spécification du constructeur.

II-3-3-Erreur d'offset :

Le code binaire 0 ne correspond pas forcement à une tension rigoureusement nulle en sortie. Cette tension est **la tension de décalage** ou **d'offset .**

En pratique, analog devices définit cette erreur comme étant l'écart entre la valeur théorique et la valeur réelle mesurée sur la première transition du convertisseur et exprime en LSB .(voir fig. II-5)

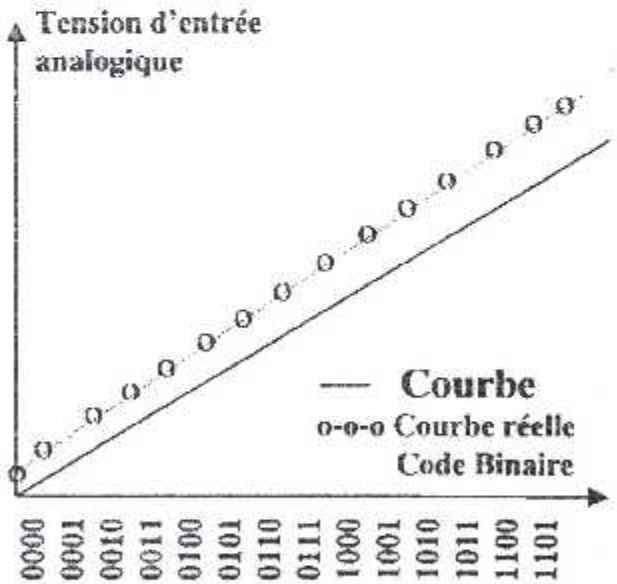


Figure II-5 **Erreur d'offset.**

II-4-Différents types de convertisseurs :

Il existe différents types de convertisseurs analogiques-numériques. certains sont très rapides mais exigent un très grand nombre de composants électroniques. D'autres utilisent des composants de façon astucieuse,reduisant le coût du matériel mais augmentant le temps nécessaire pour la conversion. Parmis ces convertisseurs ,on peut citer :

*CAN à rampe numérique ;

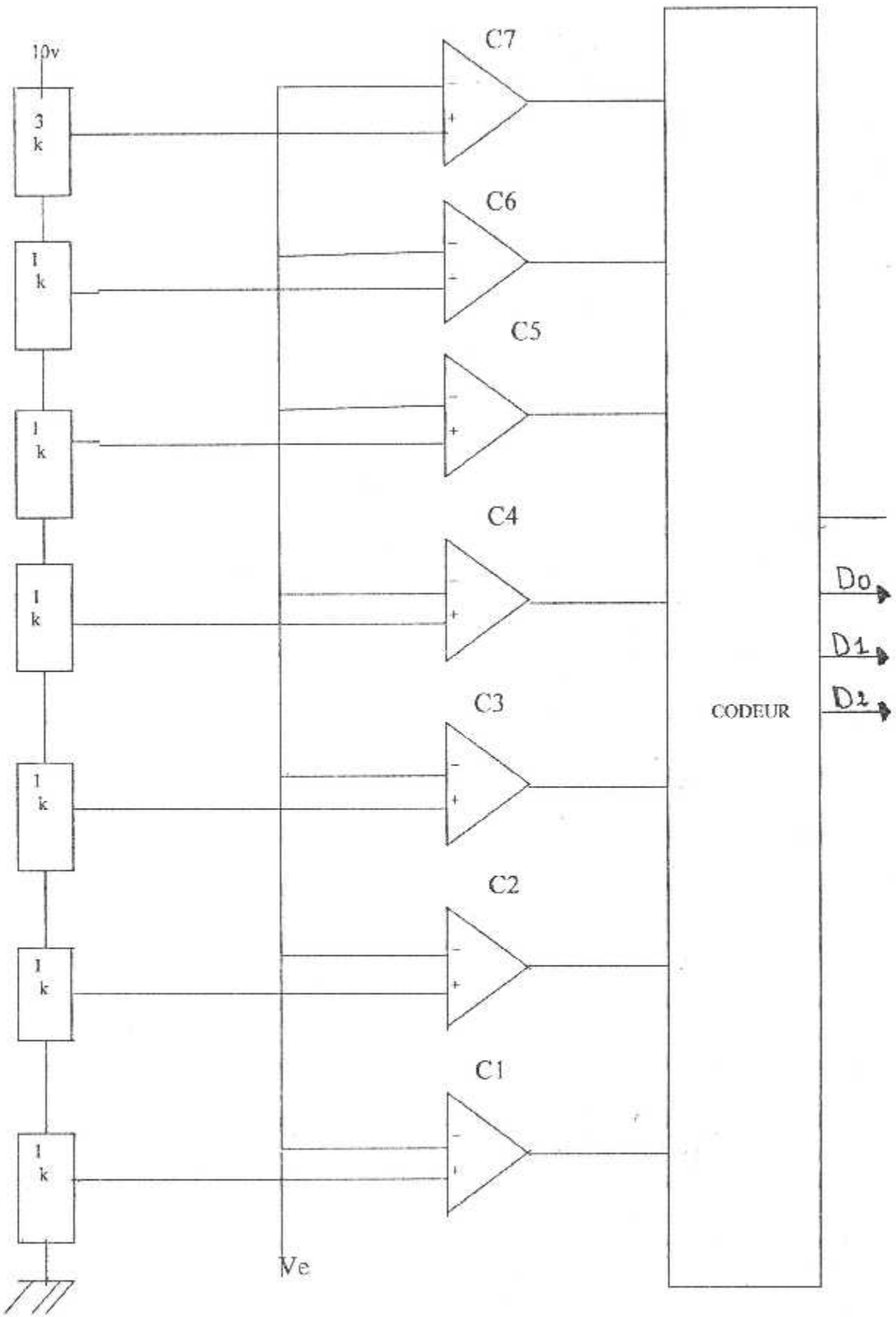
*CAN parallèle ;

*CAN par approximations successives .

On étudie dans ce qui suit le fonctionnement de chacun.

II-4-1-CAN PARALLELE :

C'est le plus rapide .Il possède un très grand nombre de circuits, ce qui explique son prix qui est plus élevé. Il y'a 2 expo n-1 comparateurs, n étant le nombre de bits du convertisseur. Dans l'exemple ci dessous, il y'a 7 comparateurs pour un convertisseur de 3 bits.


10v
3 k
1 k
1 k
1 k
1 k
1 k
1 k
C7
C6
C5
C4
C3
C2
C1
Ve
CODEUR
Do
D1
D2

Ve	C1	C2	C3	C4	C5	C6	C7	D2	D1	D0
Ve<1	1	1	1	1	1	1	1	0	0	0
1v<V e<2v	0	1	1	1	1	1	1	0	0	1
2v<Ve<3v	0	0	1	1	1	1	1	0	1	0
3v<Ve<4v	0	0	0	1	1	1	1	0	1	1
4v<Ve<5v	0	0	0	0	1	1	1	1	0	0
5v<Ve<6v	0	0	0	0	0	1	1	1	0	1
6v<Ve<7v	0	0	0	0	0	0	1	1	1	0
Ve>7v	0	0	0	0	0	0	0	1	1	1

II-4-2-Convertisseur à rampe numérique :

L'une des versions les plus simples de la conversion analogique- numérique comme le montre la figure II-6. Il utilise comme registre un compteur binaire qui est incremente par le signal d'horloge. On l'appelle rampe numérique à cause de la forme d'onde qui est celle de rampe pas à pas (en réalité en escalier).

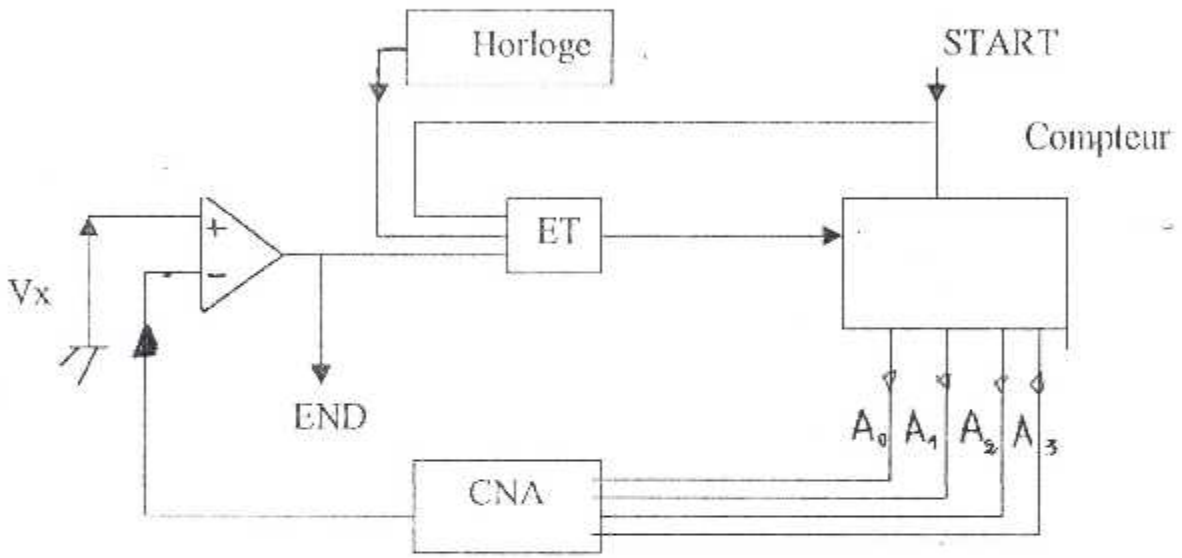


Fig. II-6-Convertisseur rampe numérique

***Principe :**

L'impulsion 'start'(niveau haut)met à zéro le compteur et bloque la porte 'ET'.
La tension de sortie v' du CNA est nulle. La sortie 'End' est au niveau haut. Lorsque 'Start' retrouve l'état bas, la porte 'ET' est validée, le signal d'horloge arrive au compteur qui s'incremente et fait évoluer la sortie du CNA par bonds successifs de la valeur de la résolution. Quand V'>Vx ,la sortie du comparateur passe au niveau bas 'End' et bloque le compteur à la valeur numérique représentant Vx.(voir fig. II-7)

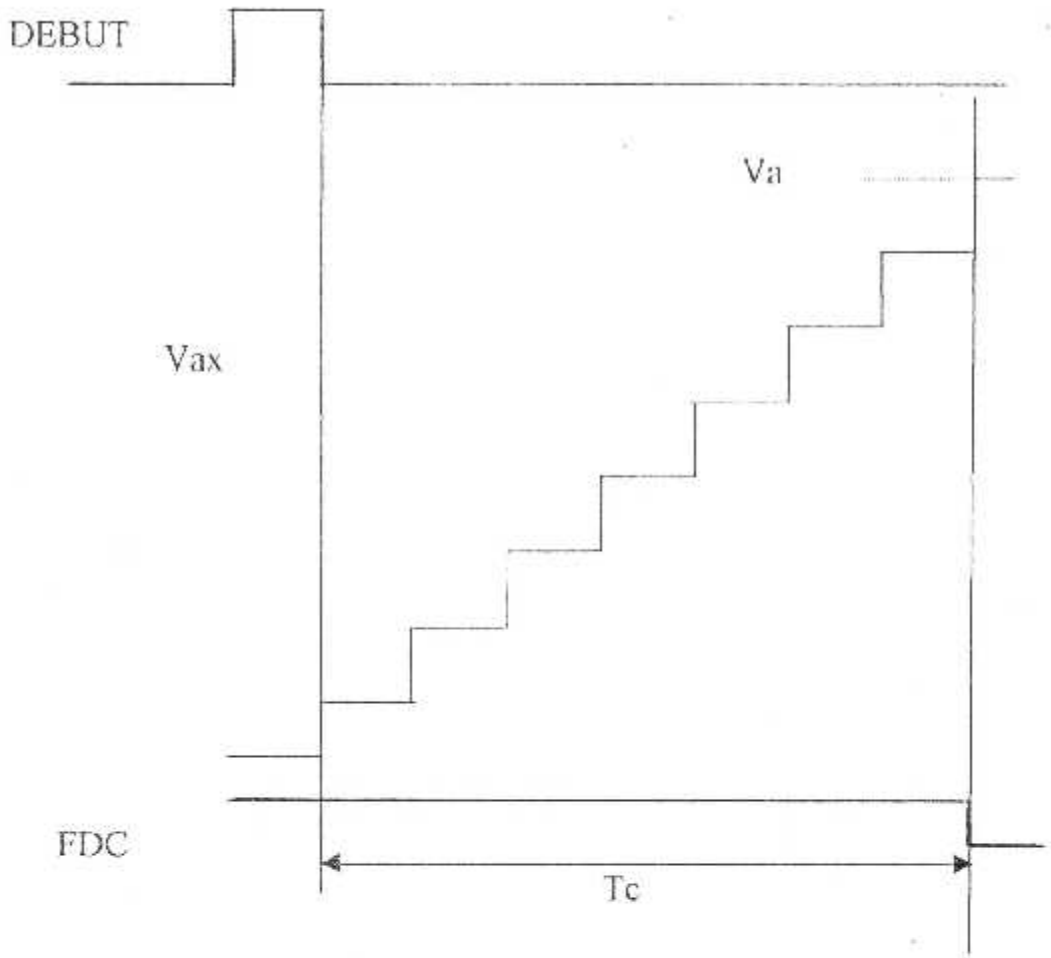


Fig. II-7-Principe du convertisseur à rampe numérique

II-4-3-Convertisseur par approximations successives :

C'est le type de convertisseur utilise dans notre projet. Ce type de convertisseur est l'un de ceux qu'on retrouve le plus fréquemment. Il possède des circuits plus complexes que le convertisseur rampe numérique, mais son temps de conversion est beaucoup plus court. En outre ,les convertisseurs par approximations successives ont une durée de conversion fixe qui ne dépend pas de la valeur de l'entrée analogique. Le montage de base de ce convertisseur est semblable à celui de rampe numérique (voir fig. II-10).Toutefois, le convertisseur n'utilise pas de compteur pour alimenter l'entrée du convertisseur CNA mais plutôt un registre. La logique de contrôle modifie le contenu du registre bit par bit jusqu'à ce que la donnée qui s'y trouve soit l'équivalent numérique du signal analogique Va. Le déroulement de ce processus est décrit au moyen d'un organigramme
(voir fig. II-11).

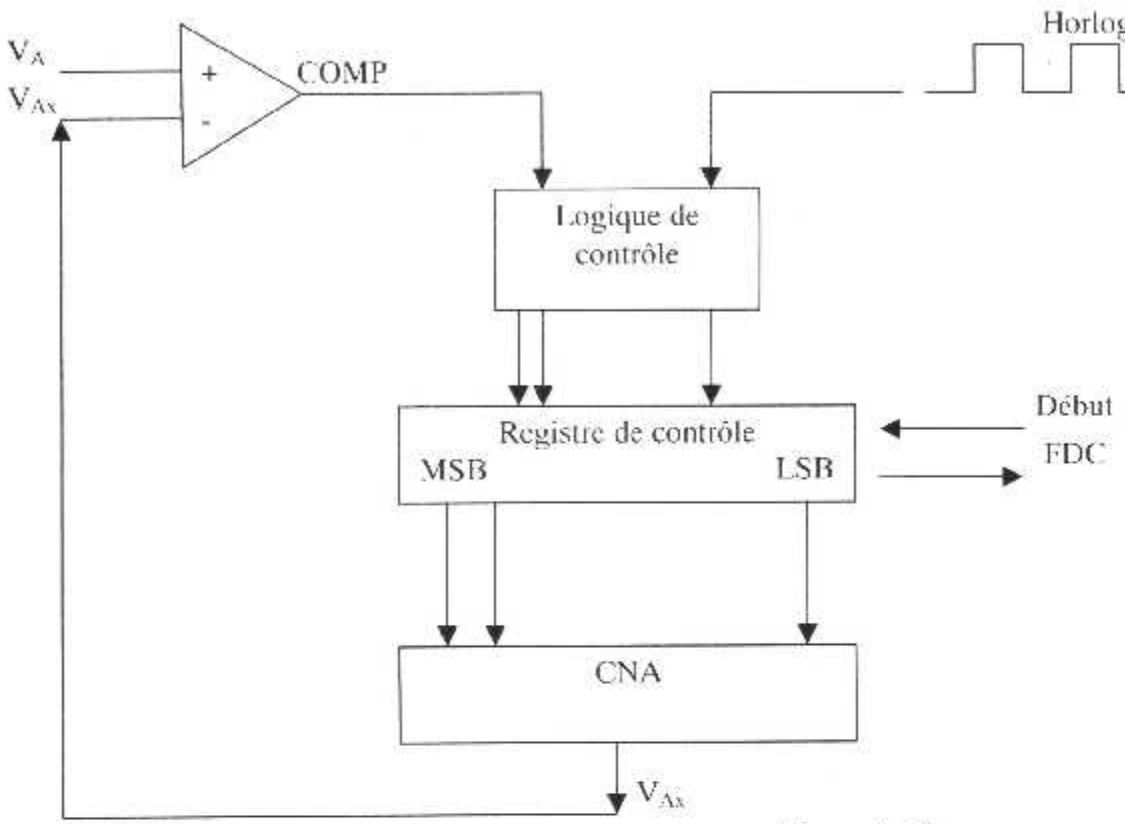

Horloge
V_A
V_{Ax}
+
-
COMP
Logique de contrôle
Registre de contrôle
MSB
LSB
Début
FDC
CNA
V_{Ax}

Figure II-10

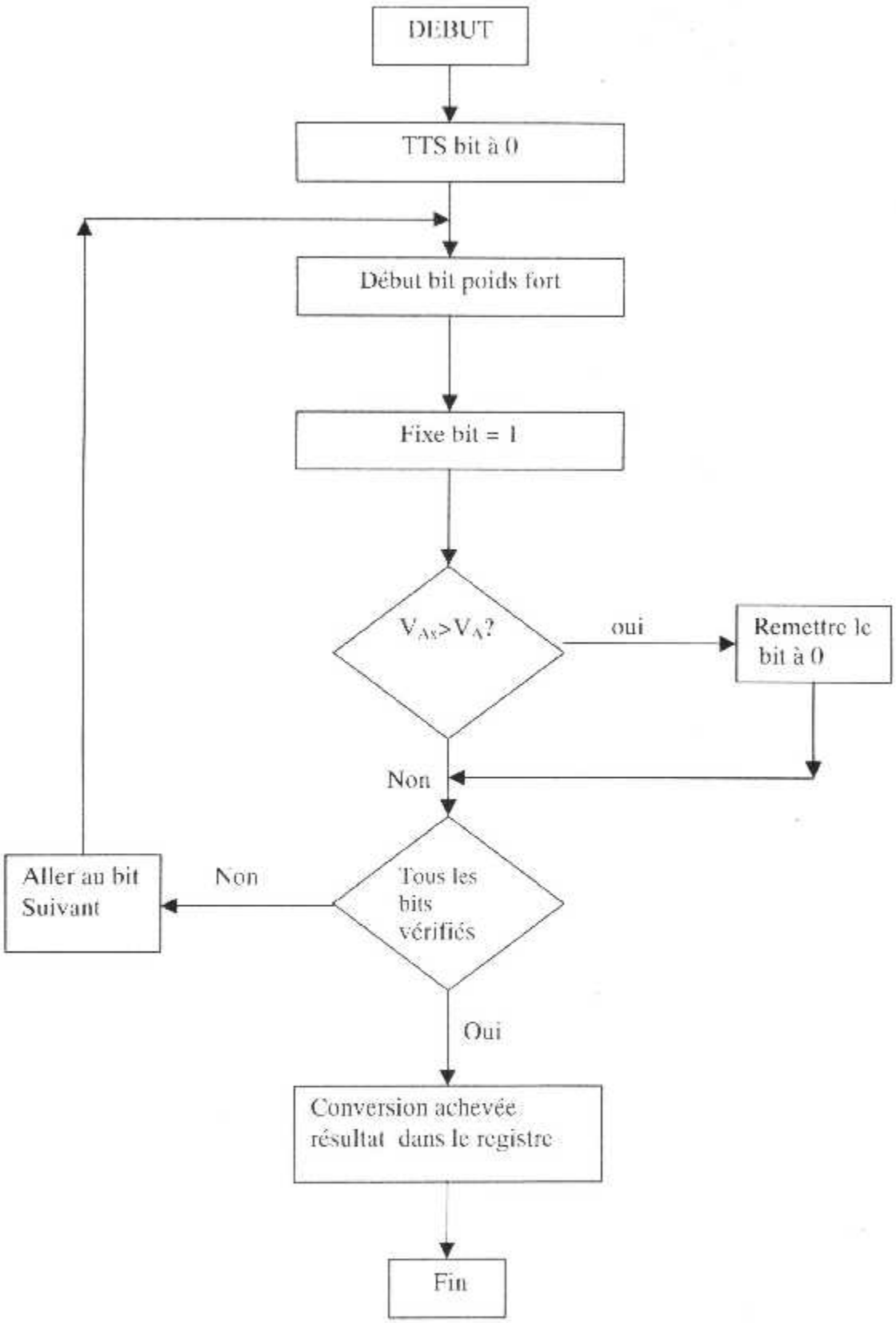


Figure II- 11 : Organigramme

On se réfère à cet organigramme pendant l'étude détaillée de l'exemple de la figure II-12.

Exemple :

Supposons que l'entrée analogique soit V_A =10.4V. La 1ère action est une mise à zéro par la logique de contrôle de tous les bits des registres, donc $Q_3 = Q_2 = Q_1 = Q_0 = 0$ donc [Q] =0000. La sortie du CNA est alors $V_{Ax} = 0$ au $t = t_0$. Comme $V_{Ax} < V_A$, la sortie du comparateur est HAUTE.
A l'instant suivant t_1, la logique de contrôle met à 1 le bit du registre du poids fort et [Q]=1000. Cela rend V_{Ax} = 8V, l'intégralité $V_{Ax} < V_A$ est conservée et la sortie COMP reste HAUTE. Ce niveau HAUT est un signal à la logique de contrôle que la mise à 1 du bit du poids fort n'a pas suffit à amener V_{Ax} au delà de V_A. Le résultat est de garder le bit de poids fort à 1. La logique de contrôle se rend à Q_2, qu'elle met à 1 afin de produire [Q]=1100.
Cette fois-ci à l'instant t_2, V_{Ax} = 12 V, ce qui crée l'inégalité $V_{Ax} > V_A$ et fait passer le signal COMP au niveau BAS, ce qui signifie à la logique de contrôle que la valeur V_{Ax} est trop grande et sa réaction est de remettre ce bit à 0 à l'instant t3. V_{Ax} revient à 8V. A t_4, le bit Q1 sera changé pour former [Q] = 1010 et d'avoir V_{Ax} = 10V. Pour cette valeur $V_{Ax} < V_A$. Le signal du COMP est HAUT et la logique reçoit l'ordre de laisser le bit Q1 à 1. La dernière étape consiste à mettre Q0 à 1. Cela donne le registre [Q] = 1011 et V_{Ax} = 11V. Comme $V_{Ax} > V_A$, le signal COMP passe au niveau BAS indiquant que la sortie numérique dépasse le signal analogique ; la logique de contrôle reçoit l'ordre de remettre le bit Q0 à 0 à l'instant t6. Comme à ce moment là tous les bits du registre ont été traités. La conversion est terminée et la logique de contrôle active le signal FDC qui indique que l'équivalent numérique trouvé pour le signal analogique se trouve dans le registre.
Dans notre exemple, la sortie numérique correspondant à V_A =10.4V est [Q] = 1010,soit 10V. On voit bien que cette valeur est inférieur à la valeur analogique ; cette réponse est caractéristique de la méthode par approximations successives.

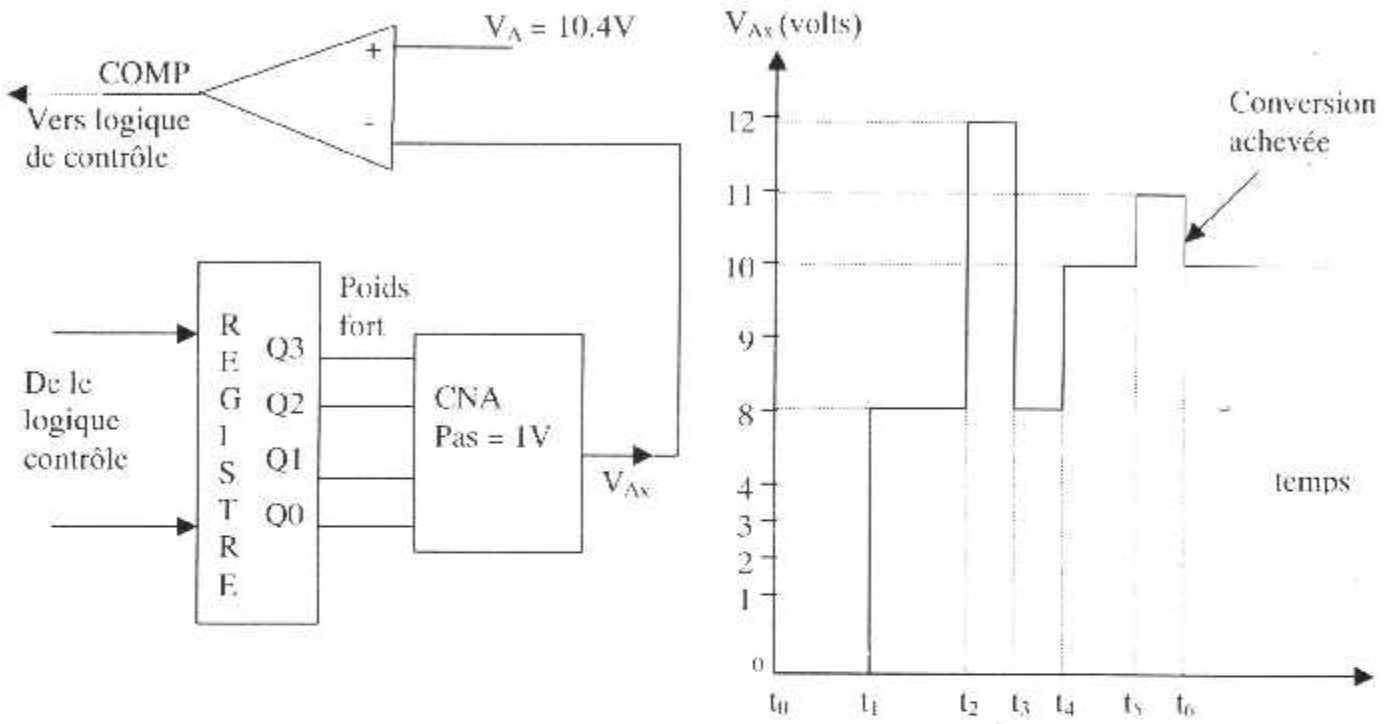


Fig. II-11 Illustration du fonctionnement d'un cas de 4 bits utilisant un CNA de 4 bits dont le pas de progression est 1V

Chapitre III. LECTURE ET REALISATION

III - étude et réalisation :

III- 1- Introduction :

Comme notre but consiste à réaliser une interface série pour la visualisation des signaux analogique sur PC. Nous devons concevoir une carte d'interface qui permet le dialogue entre le PC et les signaux d'entrée, pour cela il faut mettre en œuvre plusieurs étages qui constituent la partie HARDWARD.

Les principaux étages sont :

*Bloc d'alimentation ;

*Bloc génération de la tension de référence ;

*Bloc d'entrée ;

*Bloc conversion A/N ;

*Bloc de PC (commande).

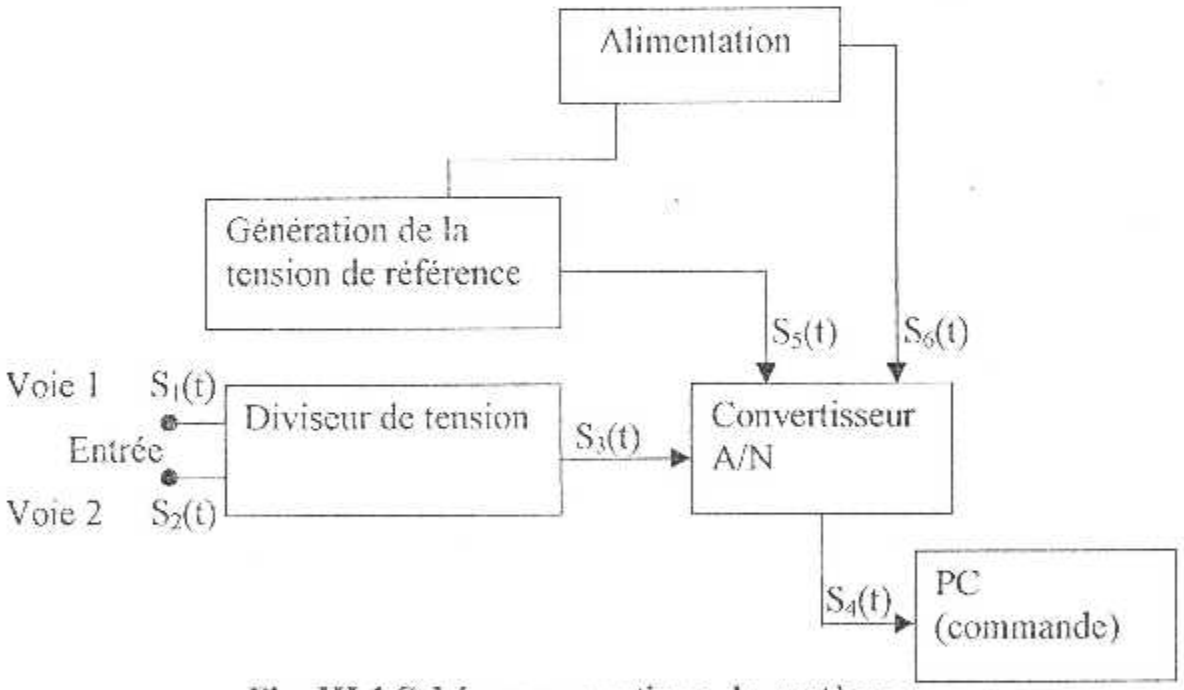


Fig. III.1 Schémas synoptique du système

III-2 Principe de fonctionnement :

Avant que la tension analogique ne soit convertie puis envoyée vers l'ordinateur ,elle passe par plusieurs circuits électroniques. Afin de réaliser toutes ces opérations on étudiera en détail les différentes étapes qui constituent le passage de la tension à travers ces circuits.

III-2-1 Alimentation :

C'est une source d'énergie nécessaire pour alimenter le montage. Dans notre cas,on a utilise une pile de 9V.

A la sortie du régulateur 7805, on recueille un potentiel continu stabilisé à 5V.

la capacité C_1 effectue un premier filtrage sur la sortie du régulateur 7805.

On a choisi ce régulateur de tension positive à trois broches vu ses caractéristiques ,qui sont:

- Courant de sortie au moins 1A ;
- Protection thermique interne contre les surcharges ;
- Aucun composant externe est nécessaire ;
- Plage de sécurité pour le transistor de sortie ;
- Limitation interne du courant de court- circuit ;
- Disponible en boîtier aluminium TO-3.

III-2-2- générateur d'une tension de référence « Vref » :

Pour ce bloc ,on a une diode de type LM 336 qui grâce au potentiomètre de réglage R qui fournit la tension exacte nécessaire au convertisseur qui doit être de 2,5 V.

III-2-3 – bloc d'entrée

Il est composé d'un pont diviseur de tension sur chaque entrée qui comprend (voie 1 et $\overline{\text{voie}}$ 2) et met à niveau les tensions qui ne doivent en aucun cas excéder la valeur 5V.

Calcul des éléments :

Vmax = 5 V (la tenson maximum)
Vin = 2.5 V
Calcul des valeurs R2, R3, R4, R5 :
On applique le diviseur de tension :

Théorème :

$$V\min = \frac{R}{R+R} \bullet V\max$$

$$Vin = \frac{R2}{R2+R3} \bullet V\max \Rightarrow \begin{cases} Vin = 2.5V \\ V\max = 5V \end{cases}$$

$$2.5 = \frac{R2}{R2+R3} \bullet 5 \Rightarrow R2 + R3 = 2R2 \ldots\ldots\ldots\ldots (1)$$

Pour l'équation (1) vérifier :

R2=R3

Même méthode pour calculer les deux résistances suivantes R4 et R5.
Donc : R2 – R3 – R4 – R5 = 10 KΩ

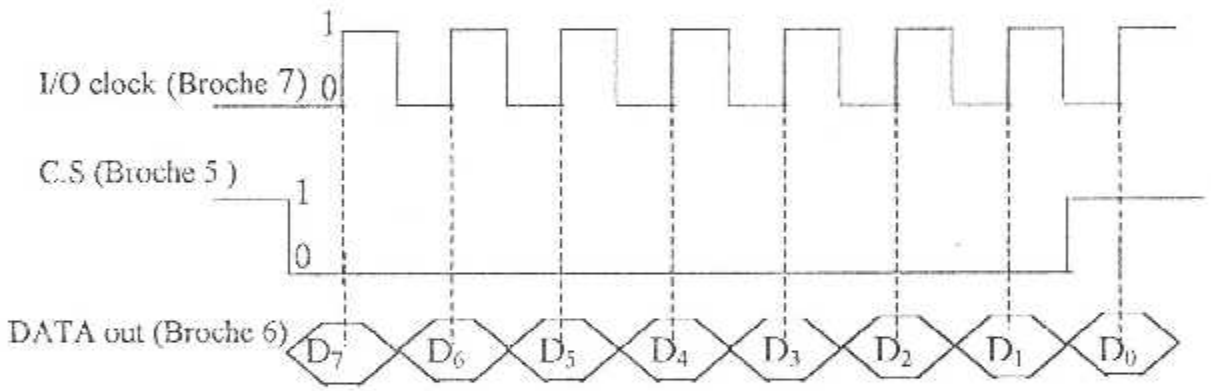
III-2-4- Bloc de conversion A/N :

Le circuit de conversion analogique – numérique est de type TLC 548 qui est un convertisseur par approximations successives ,qui donne la séquence

d'événement nécessaire à la conversion analogique – numérique ,dont

l'aboutissement s'exprime sur la forme d'un mot de 8 bits d'une valeur comprise entre 0 est 255. L'acquisition de cette valeur s'obtient en plaçant a l'état Bas - l'entrée / CS (sélection du circuit), puis en appliquant un signal d'horloge.
Le CS et l'horloge sans délivrés par les sortie D_0 et D_1 du registre de données du port parallèle. La lecture des données envoyées par un circuit de conversion A/N ne nécessite qu'une seule entrée du registre d'état de l'imprimante .
Nous devrons donc effectuer la lecture successivement sur l'une puis sur l'autre (Busy et Ack) afin d'obtenir le résultat des deux conversions.

Conversion analogique - digitale sur 8 bits avec le TLC548



III-2-5- Bloc du PC (commande)

Les broches utulisée sur le DB25 du port parallèle sont représentes par le tableau suivant :

DB25	Nom	Niveau	Entrée/Sortie
2	D_0	1	S
3	D_1	1	S
10	Ack	0	E
11	Busy	1	E
25	GND	/	/

PARTIE SOFTHARD:

III-3 Introduction:

Delphi est un environnement de programmation permettant de développer des applications. Il incarne la suite logique de la famille Turbo Pascal avec ses nombreuses versions.

Delphi est un outil moderne, il fait appel à une conception visuelle des applications, en programmation objet. De plus, il prend en charge le maintien automatique d'une partie du code source.

Alors pourquoi Delphi ? La question a un sens car, à première vue, en comparant Delphi et Visual Basic, on penserait plutôt à du plagia.

En réalité il n'en n'est rien. Par sa conception, son environnement de développement et ses performances, Delphi fait de plus en plus d'adeptes. C'est sa courte histoire qui tend à le prouver, au travers de quelques constatations.

De plus en plus ,de journaux et revues publiant habituellement des programmes écrits en C, se convertissent à Delphi. C'est le cas par exemple du très répandu PC Magazine.

Les chiffres de ventes ont même réussi à sauver Borland (l'éditeur de Delphi), au-delà de toute attente réaliste.

Or, il se trouve que Borland a utilisé le même noyau de

compilateur pour son Delphi et pour son C++.

La vie d'un développeur Windows ne se résume plus qu'à un choix de langage.
Il y a bien d'autres environnements de développement, mais leur faible diffusion les marginalise.

Pour construire l'interface d'une application, ce dernier place des objets sur une fiche («fenêtre ») et les personnalise en modifiant éventuellement leurs propriétés et/ou en leur attachant des instructions liées à des événements donnés.

Bien que pour certaines applications il ne soit pas nécessaire d'écrire du code(ni de connaître le Pascal), il vaut mieux avoir une solide expérience dans le domaine de programmation avant de se lancer dans un développement réel.

II-2--FORME GENERALE D'UN PROGRAMME DELPHI :

Dans Delphi on est peu confronté au programme proprement dit, mais plutôt à des unités, ce qui constitue une grande différence par rapport au pascal traditionnel. De plus, le programme(principal) appelé projet :

*est généralement sauvegarde dans un fichier. dpr (pour Delphi Project) ;
*est toujours de petite taille ;
*est automatiquement crée par Delphi ;
*contient les références aux unités qui constituent l'application ;
*Initialise l'application, crée les différentes fiches et lance l'exécution de l'application.

Le programme utilise l'unité Unit1 stockée dans le fichier Unit1.pas et concerne la fiche Form1.Une Unité comporte :

*Un en-tête ;
*Une partie interface dans laquelle figurent les objets exportables ;
*Déclaration de constantes, types et variables ;
*Des déclarations de procédures ;

*Une partie implémentation ;

*Le corps de l'unité ou partie d'initialisation, qui est souvent vide et réduite à 'end'.

III-4-INSTRUCTIONS DE DELPHI :

Parmis les instructions utilisées dans notre programme:

***Instruction 'while' et 'reapet' :**

Ces instructions sont des structures répétitives qui se distinguent par le fait que la condition de sortie est conséquente à l'évaluation d'une expression booléenne.

La structure 'while' permet de répéter l'exécution d'une instruction ou d'un bloc d'instructions tant qu'une expression est vérifiée.

L'instruction 'reapet' permet de répéter l'exécution d'une instruction ou d'un bloc d'instructions jusqu'à ce qu'une condition soit vérifiée.

En apparence, cette structure ressemble à la précédente. En réalité, la différence est significative et importante. L'expression booléenne dans l'instruction 'while' est la négation logique de l'instruction 'reapet '

***Instruction 'for' :**

Cette structure répétitive est utilisée lorsque le nombre d'itérations est connu.

***Instruction 'if' :**

C'est une structure sélective, qui permet d'effectuer des choix selon des critères (ou conditions) que le programmeur a fixé. On trouve les trois mots réservés if, then et else.

III-4 – Le logiciel :

Le logiciel comporte une fiche principale constituée de :

- Trois boutons ;
- Deux cases à cocher (check box) ;
- Quatre zones d'édition ;

Le tracé du graphe s'effectue directement sur l'écran affiché dans une fenêtre de Windows. Une zone d'affichage des graphes est située sur la gauche de cette fenêtre. Elle comporte un quadrillage dans chaque espace entre les traits horizontaux correspond à 1V. A droite, sont disposés les contrôles relatifs aux voies, avec notamment dans des cases à cocher la sélection de la voie active. Un premier bouton « Lecture instantanée » ne donne qu'une valeur comprise entre 0 et 255. Le résultat s'affiche dans les zones d'édition respectives des voies une et deux.

Les boutons de réglage situés dans l'encadrement des voies 1 et 2 (mV) permettent le tracé d'une ligne de référence sur le graphe. Il suffit pour cela d'ajuster la valeur en millivolt de la voie qui ne doit pas être active pour le relevé.

Les contrôles relatifs au tracé du graphe se situent sous la zone graphique. Un premier bouton « TRACE » permet dès son appuis d'effectuer un tracé en continu des variations observées sur la voie sélectionnée. La diode électroluminescente qui s'affiche allumée indique que le tracé est actif. En appuyant à nouveau sur « TRACE », la diode électroluminescente s'éteint et le graphe se fige.

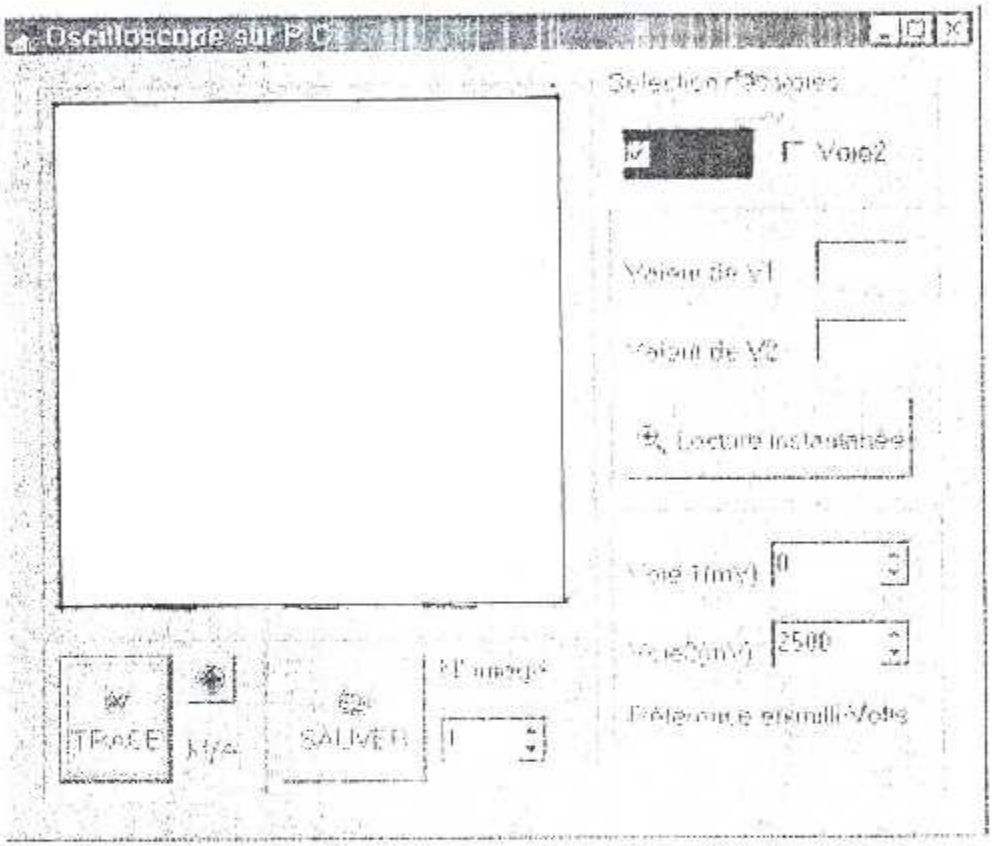
On peut sauvegarder le graphe sous la forme d'une une image Bit Map, rangé dans le répertoire sous un nom « oscillo(numéros).bmp » dès l'appuis sur le bouton « SAUVER ». Pour modifier le numéros du dessin on varie la valeur de N image.

```
nit Unit1;

interface

ses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, ExtCtrls, ComCtrls, Spin;

ype
  TForm1 = class(TForm)
    Bevel1: TBevel;
    Image1: TImage;
    BitBtn1: TBitBtn;
    GroupBox1: TGroupBox;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    Bevel2: TBevel;
    Edit1: TEdit;
    Edit2: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    BitBtn2: TBitBtn;
    Bevel3: TBevel;
    SpinEdit1: TSpinEdit;
    SpinEdit2: TSpinEdit;
    Label3: TLabel;
    Label4: TLabel;
    Bevel4: TBevel;
    BitBtn4: TBitBtn;
    Bevel5: TBevel;
    Label6: TLabel;
    BitBtn5: TBitBtn;
    SpinEdit3: TSpinEdit;
    Label5: TLabel;
    Label7: TLabel;
    procedure temporise(msecs:integer);
    procedure active_sortie;
    procedure lire_les_entrees;
    procedure Zone_de_dessin(Sender: TObject);
    procedure mesure;
    procedure Trace_le_graphe(Sender: TObject);
    procedure une_mesure(Sender: TObject);
    procedure trace(Sender: TObject);
    procedure oscilographe(Sender: TObject);
    procedure sauve_image(Sender: TObject);
  private
    { Déclarations privées }
  public
    { Déclarations publiques }
  end;

var
  Form1: TForm1;
  Bitmap: TBitmap;
mplementation

{$R *.DFM}

ar
entree,sortie: smallint;
bit,valeur1,voie1,valeur2,voie2: integer;
ascule: integer;

procedure TForm1.temporise(msecs:integer);
 cette procédure crée une temporisation de
 n fois 1 milliseconde;
var
   FirstTickCount:longint;
egin
  FirstTickCount:=GetTickCount;
   repeat
     Application.ProcessMessages;
     {pour ne pas bloquer l'accès aux divers contrôles}
  until ((GetTickCount-FirstTickCount) >= Longint(msecs));
```

```
nd;

procedure TForm1.active_sortie;
begin
  asm
    mov dx, 0378h
    mov ax, sortie
    out dx, al
  end; {de ASM}
  {temporise(1);}
end;

procedure TForm1.lire_les_entrees;
begin
  asm
    mov dx, 0379h
    in  al, dx
    mov entree, al
  end; {de ASM}
nd;

procedure TForm1.Zone de dessin(sender: Tobject);
begin
  Bitmap := TBitmap.create;
  Bitmap.Width := Image1.width;
  Bitmap.height := Image1.Height;
  Image1.Picture.Graphic := Bitmap;
nd;

procedure TForm1.Trace_le_graphe(Sender: TObject);
  var i, ligne, colonne : integer;
egin
  {Tempo := SpinEdit2.Value * 1000;
  Nb_mesures := 255;}
  Bitmap.free;
  Zone_de_dessin(Sender);
  Image1.Canvas.pen.Width := 1;
  Image1.Canvas.pen.Style := PsSolid;
  Image1.Canvas.pen.Color := clGray;
  Image1.Canvas.rectangle(0,0,255,257);
  Image1.Canvas.pen.Style := PsDot;
  ligne :=52;
  while ligne < 257 do
    begin
      Image1.Canvas.MoveTo(0,ligne);
      Image1.Canvas.LineTo(255,ligne);
      ligne := ligne + 51;
    end;
  colonne :=51;
  while colonne < 255 do
    begin
      Image1.Canvas.MoveTo(colonne,0);
      Image1.Canvas.LineTo(colonne,257);
      colonne := colonne + 51;
    end;
  Image1.Canvas.pen.Style := PsSolid;
end;

procedure TForm1.mesure;
  var  i,valeur_bit1,valeur_bit2 : integer;
begin
  valeur1 := 1;
  valeur_bit1 :=128;
  valeur2 := 1;
  valeur_bit2 :=128;
  sortie := 1;
  active_sortie;
  sortie := 0;
  active_sortie;
  sortie := 2;
  active_sortie;
  lire_les_entrees;
  if (entree and 128)=128 then bit:=1 else bit:=0;
  valeur1 := valeur1 + (valeur_bit1*bit);
```

```
  valeur_bit1 := valeur_bit1 Div 2;
  if (entree and 64)=64 then bit:=0 else bit:=1;
  valeur2 := valeur2 + (valeur_bit2*bit);
  valeur_bit2 := valeur_bit2 Div 2;
   for i := 1 to 7 do
    begin
      sortie := 0;
      active_sortie;
      sortie := 2;
      active_sortie;
      lire_les_entrees;
      if (entree and 128)=128 then bit:=1 else bit:=0;
      valeur1 := valeur1 - (valeur_bit1*bit);
      valeur_bit1 := valeur_bit1 Div 2;
      if (entree and 64)=64 then bit:=0 else bit:=1;
      valeur2 := valeur2 + (valeur_bit2*bit);
      valeur_bit2 := valeur_bit2 Div 2;
    end;
    If CheckBox1.State=cbChecked
       then begin
              voie1 := 256-valeur1;
            end
        else begin valeur1:=256-Trunc((SpinEdit1.Value/100)*5.1); end;
    If CheckBox2.State=cbChecked
       then begin
              voie2 := 256-valeur2;
            end
       else begin valeur2:=256-Trunc((SpinEdit2.Value/100)*5.1); end;
  sortie := 0;
  active_sortie;
nd;

procedure TForm1.une_mesure(Sender: TObject);
egin
  mesure;
  edit1.text := IntToStr(voie1);
  edit2.text := IntToStr(voie2);
nd;

procedure TForm1.oscillographe(Sender: TObject);
  var
    v1x1, v1x2, v2x1, v2x2 : integer;
    v1y1, v1y2, v2y1, v2y2 : integer;
    nombre_points : integer;
egin
  If bascule = 1 then
     begin
       while bascule = 1 do
         begin
           Trace_le_graphe(Sender);
           mesure;
           v1y1:=valeur1; {initialisation des coordonnées }
           v2y1:=valeur2; {pour le graphe                 }
           nombre_points :=1;
           v1x1:=nombre_points;
           v2x1:=nombre_points;
           while nombre_points<255 do
             begin
               mesure;
               v1y2:=valeur1;
               v2y2:=valeur2;
               Image1.Canvas.Pen.Color := clRed;
               Image1.Canvas.MoveTo(v1x1,v1y1);
               Image1.Canvas.LineTo(nombre_points,v1y2);
               Image1.Canvas.Pen.Color := clBlue;
               Image1.Canvas.MoveTo(v2x1,v2y1);
               Image1.Canvas.LineTo(nombre_points,v2y2);
               v1y1:=v1y2; {initialisation des coordonnées }
               v2y1:=v2y2; {pour le graphe                 }
               v1x1:=nombre_points;
               v2x1:=nombre_points;
               nombre_points := nombre_points + 1;
             end;
           Application.ProcessMessages;
```

```
        end; (fin du while)
    end (fin du then)
  else
    begin end;
d;

procedure TForm1.trace(Sender: Tobject);
gin
  If bascule=0
    then begin bascule := 1;
         BitBtn4.Glyph.LoadFromFile('Led1On.bmp'); end
    else begin bascule := 0;
         BitBtn4.Glyph.LoadFromFile('Led1Off.bmp'); end;
  oscilographe(Sender);
end;

ocedure TForm1.sauve_image(Sender: TObject);
  var NomFichier : string;
gin
  {sauver l'image du graphe pour la récupérer
   sous la forme d'un dessin BitMap}
  NomFichier:= 'oscilo'+ IntToStr(SpinEdit3.Value) + '.BMP';
  Image1.Picture.SaveToFile(NomFichier);
d;

end.
```


Oscilloscope sur PC
Voie2
Valeur de V1
Valeur de V2
Lecture instantanée
Voie1(mV)
0
Voie2(mV)
2500
TRACE
SAUVER
1

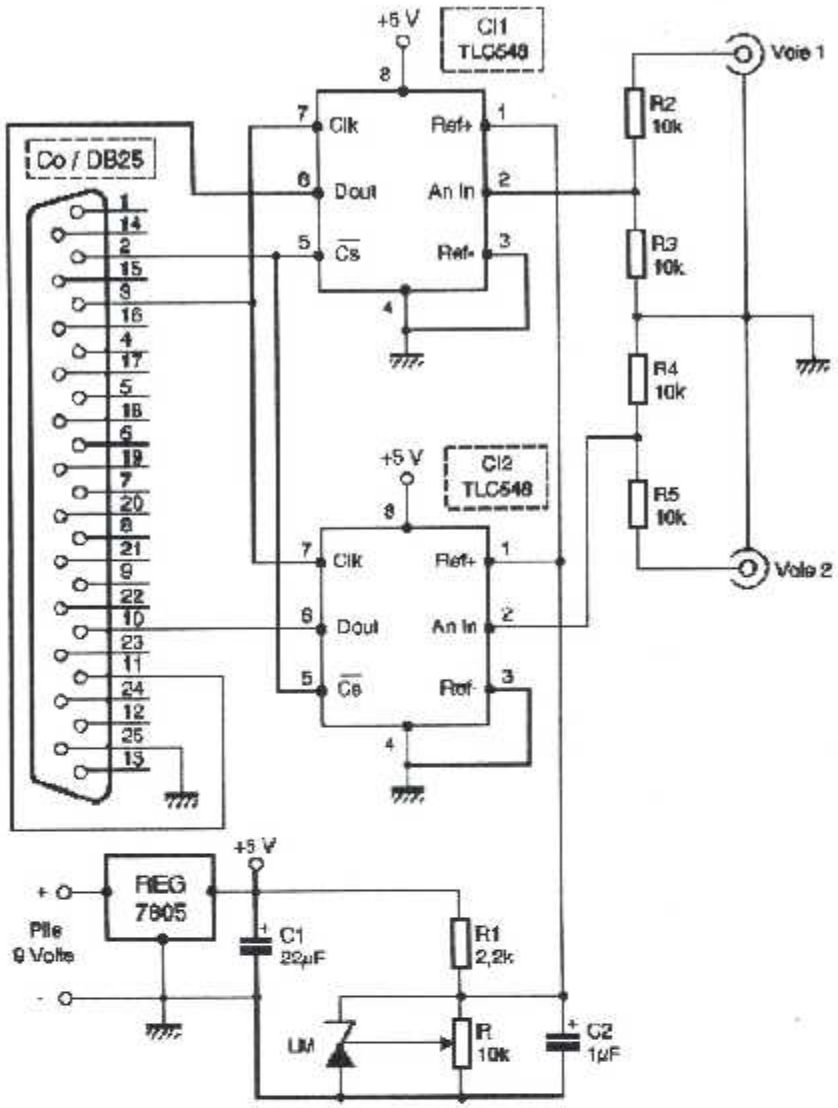
La communication avec l'interface se fait par deux procédures d'assembleur :

Première procédure :

mov dx,0378h (Chargement du registre de donnée par l'adresse 0378h)
mov ax, sortie (Chargement de l'accumulateur avec la donnée sortie)
out dx ,al (L'envoi de la valeur de ax vers l'adresse contenue dans dx)

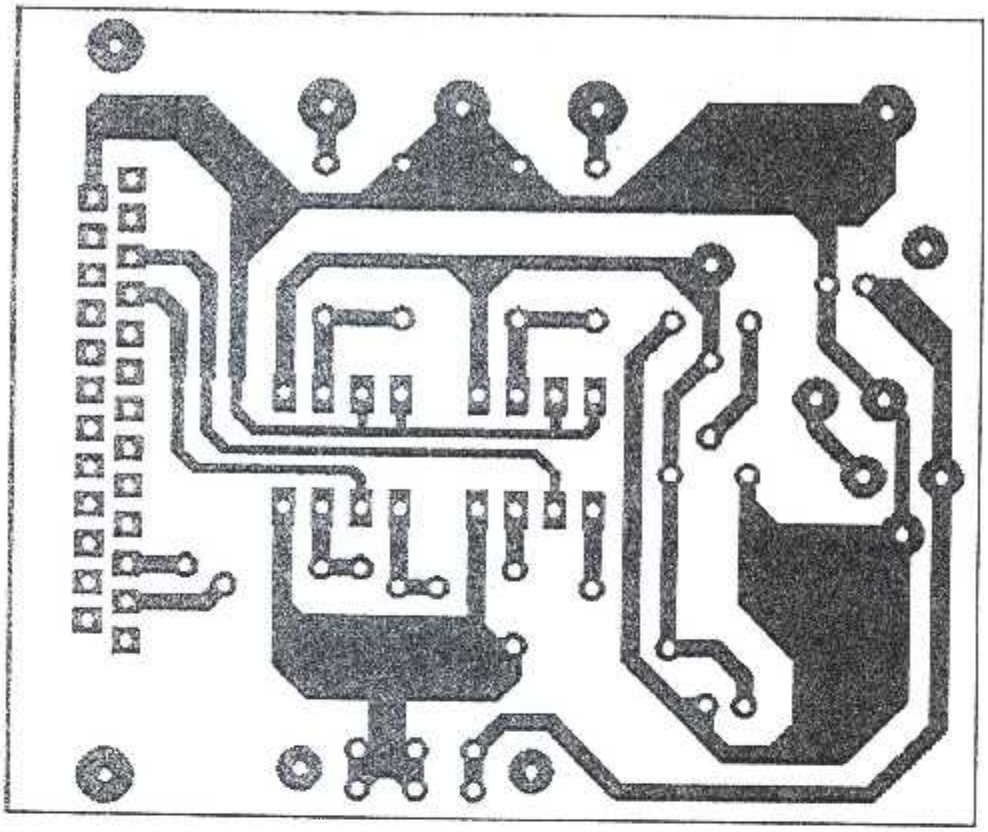
Deuxième procédure :

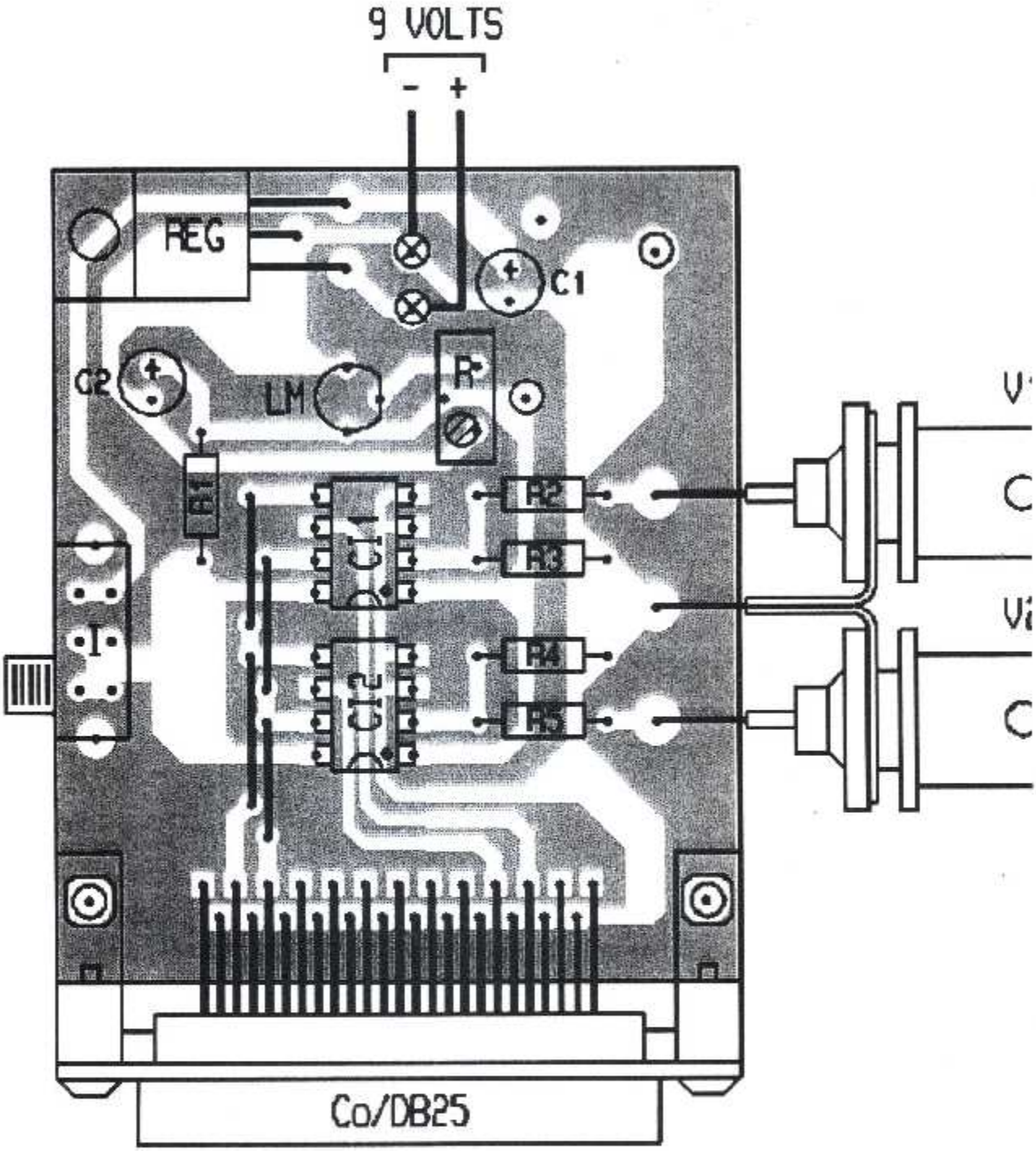
mov dx ,0379h (Chargement du registre de donnée par l'adresse 0379h).
in al ,dx (mettre la donnée contenue dans dx en al).
Mov entrée ,al (charger le contenu de al dans entrée).


+5 V
CI1
TLC548
Voie 1
Co / DB25
Clk
Dout
Cs
Ref+
An In
Ref-
R2
10k
R3
10k
R4
10k
R5
10k
CI2
TLC548
Voie 2
REG
7805
Pile
9 Volts
C1
22µF
R1
2,2k
LM
P
10k
C2
1µF

NOMENCLATURE

Composants	valeur	nombre
Capacités	22µf 1µf	1 1
resistance	2,2k 10k	1 4
potentiomètre	10 k	1
Convertisseur A/N	TLC 548	2
Diode ziner programmable	LM336	1
Regulateur	5V (type 7805)	1
Divers		
Connecteur pile	9V	1
Fiche fem.	BNC	2
Connecteur mal	DB25	1
La nappe	25FILS Long 2m	1
Connecteur mal		1
connecteur fem.		1
inverseur		1




9 VOLTS
- +
REG
C1
C2
LM
R1
R2
R3
R4
R5
C11
C12
Co/DB25

Conclusion :

Ce travail nous a permet de traduire notre savoir théorique en pratique, ce qui nous a conduit à la compréhension du phénomène de traitement de données par programmation. De ceci, on a distinguer l'importance du passage de l'analogique à la numérisation des signaux.

Enfin, ce travail est notre première expérience qui nous a permis de mettre en application certaines connaissances acquises durant notre cursus universitaire.

Espérons que notre projet sera un outil de travail et qu'il servira de référence pour les générations à venir.

Bibliographie :

*Electronique digital

Auteur :Pierre Cabanis avec la collaboration de Emmanuel Bernier

*Circuits numériques

Auteur :Ronald j. Tocci

Cote :A620 391 ex 04

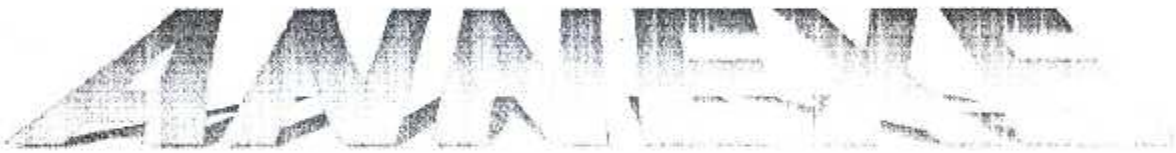
*Manuel des interfaces

Auteur : S. Leisbon

Cote :A68.60 ex 05

*Réseau Internet

www.fairchildsemi.com

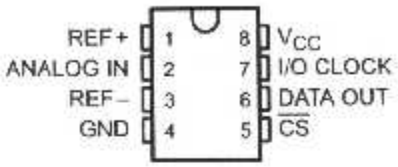


TLC548C, TLC548I, TLC549C, TLC549I
8-BIT ANALOG-TO-DIGITAL CONVERTERS WITH SERIAL CONTROL

SLAS067C – NOVEMBER 1983 – REVISED SEPTEMBER 1996

- Microprocessor Peripheral or Standalone Operation
- 8-Bit Resolution A/D Converter
- Differential Reference Input Voltages
- Conversion Time . . . 17 µs Max
- Total Access and Conversion Cycles Per Second
  - TLC548 . . . up to 45 500
  - TLC549 . . . up to 40 000
- On-Chip Software-Controllable Sample-and-Hold Function
- Total Unadjusted Error . . . ±0.5 LSB Max
- 4-MHz Typical Internal System Clock
- Wide Supply Range . . . 3 V to 6 V
- Low Power Consumption . . . 15 mW Max
- Ideal for Cost-Effective, High-Performance Applications Including Battery-Operated Portable Instrumentation
- Pinout and Control Signals Compatible With the TLC540 and TLC545 8-Bit A/D Converters and with the TLC1540 10-Bit A/D Converter
- CMOS Technology

D OR P PACKAGE
(TOP VIEW)



description

The TLC548 and TLC549 are CMOS analog-to-digital converter (ADC) integrated circuits built around an 8-bit switched-capacitor successive-approximation ADC. These devices are designed for serial interface with a microprocessor or peripheral through a 3-state data output and an analog input. The TLC548 and TLC549 use only the input/output clock (I/O CLOCK) input along with the chip select ($\overline{CS}$) input for data control. The maximum I/O CLOCK input frequency of the TLC548 is 2.048 MHz, and the I/O CLOCK input frequency of the TLC549 is specified up to 1.1 MHz.

AVAILABLE OPTIONS

T_A	PACKAGE	
	SMALL OUTLINE (D)	PLASTIC DIP (P)
0°C to 70°C	TLC548CD TLC549CD	TLC548CP TLC549CP
–40°C to 85°C	TLC548ID TLC549ID	TLC548IP TI C549IP

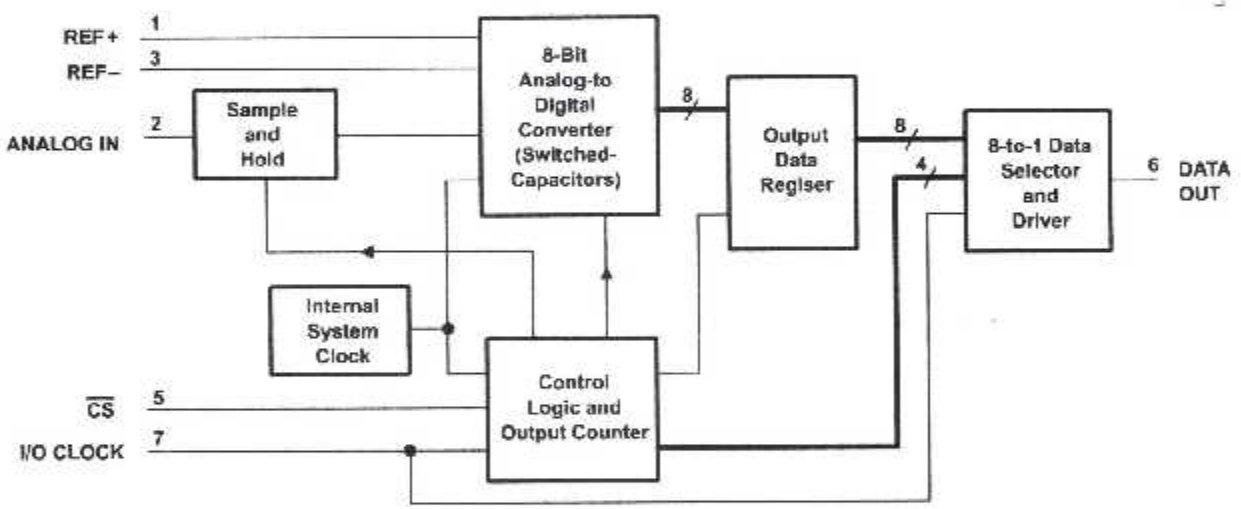
description (continued)

Operation of the TLC548 and the TLC549 is very similar to that of the more complex TLC540 and TLC541 devices; however, the TLC548 and TLC549 provide an on-chip system clock that operates typically at 4 MHz and requires no external components. The on-chip system clock allows internal device operation to proceed independently of serial input/output data timing and permits manipulation of the TLC548 and TLC549 as desired for a wide range of software and hardware requirements. The I/O CLOCK together with the internal system clock allow high-speed data transfer and conversion rates of 45 500 conversions per second for the TLC548, and 40 000 conversions per second for the TLC549.

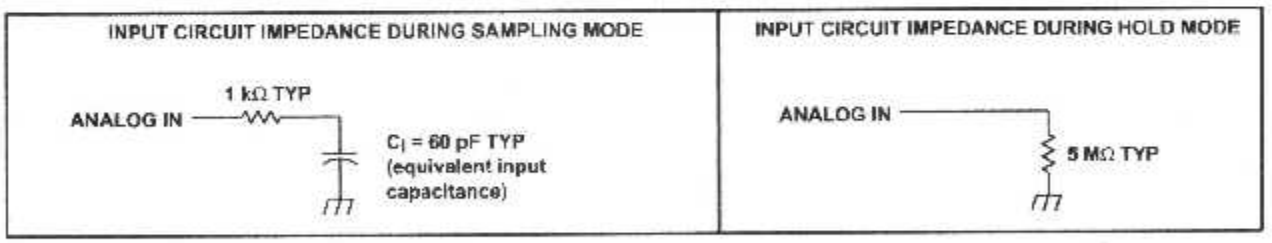
Additional TLC548 and TLC549 features include versatile control logic, an on-chip sample-and-hold circuit that can operate automatically or under microprocessor control, and a high-speed converter with differential high-impedance reference voltage inputs that ease ratiometric conversion, scaling, and circuit isolation from logic and supply noises. Design of the totally switched-capacitor successive-approximation converter circuit allows conversion with a maximum total error of ±0.5 least significant bit (LSB) in less than 17 µs.

The TLC548C and TLC549C are characterized for operation from 0°C to 70°C. The TLC548I and TLC549I are characterized for operation from –40°C to 85°C.

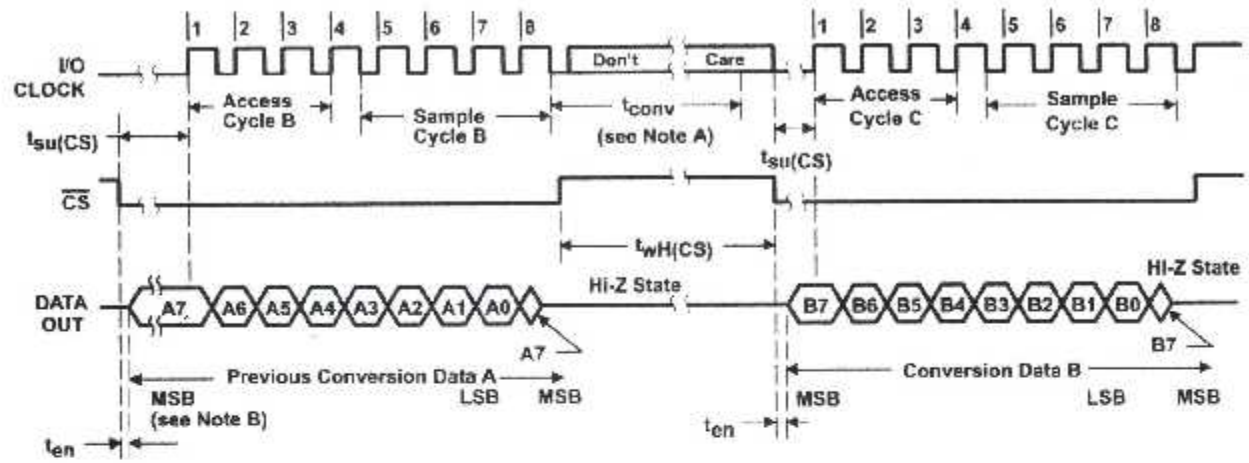
functional block diagram



typical equivalent inputs



operating sequence



NOTES: A. The conversion cycle, which requires 36 internal system clock periods (17 µs maximum), is initiated with the eighth I/O clock pulse trailing edge after $\overline{CS}$ goes low for the channel whose address exists in memory at the time.

B. The most significant bit (A7) is automatically placed on the DATA OUT bus after $\overline{CS}$ is brought low. The remaining seven bits (A6–A0) are clocked out on the first seven I/O clock falling edges. B7–B0 follows in the same manner.

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1)	6.5 V
Input voltage range at any input	−0.3 V to V_{CC} + 0.3 V
Output voltage range	−0.3 V to V_{CC} + 0.3 V
Peak input current range (any input)	±10 mA
Peak total input current range (all inputs)	±30 mA
Operating free-air temperature range, T_A (see Note 2): TLC548C, TLC549C	0°C to 70°C
TLC548I, TLC549I	−40°C to 85°C
Storage temperature range, T_{stg}	−65°C to 150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds	260°C

NOTES: 1. All voltage values are with respect to the network ground terminal with the REF− and GND terminals connected together, unless otherwise noted.

2. The D package is not recommended below −40°C.

recommended operating conditions

		TLC548			TLC549			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, V_{CC}		3	5	6	3	5	6	V
Positive reference voltage, V_{ref+} (see Note 3)		2.5	V_{CC}	V_{CC}+0.1	2.5	V_{CC}	V_{CC}+0.1	V
Negative reference voltage, V_{ref-} (see Note 3)		–0.1	0	2.5	–0.1	0	2.5	V
Differential reference voltage, V_{ref+}, V_{ref-} (see Note 3)		1	V_{CC}	V_{CC}+0.2	1	V_{CC}	V_{CC}+0.2	V
Analog input voltage (see Note 3)		0		V_{CC}	0		V_{CC}	V
High-level control input voltage, V_{IH} (for V_{CC} = 4.75 V to 5.5 V)		2			2			V
Low-level control input voltage, V_{IL} (for V_{CC} = 4.75 V to 5.5 V)				0.8			0.8	V
Input/output clock frequency, $f_{clock(I/O)}$ (for V_{CC} = 4.75 V to 5.5 V)		0		2.048	0		1.1	MHz
Input/output clock high, $t_{wH(I/O)}$ (for V_{CC} = 4.75 V to 5.5 V)		200			404			ns
Input/output clock low, $t_{wL(I/O)}$ (for V_{CC} = 4.75 V to 5.5 V)		200			404			ns
Input/output clock transition time, $t_{t(I/O)}$ (for V_{CC} = 4.75 V to 5.5 V) (see Note 4 and Operating Sequence)				100			100	ns
Duration of $\overline{CS}$ input high state during conversion, $t_{wH(CS)}$ (for V_{CC} = 4.75 V to 5.5 V) (see Operating Sequence)		17			17			µs
Setup time, $\overline{CS}$ low before first I/O CLOCK, $t_{su(CS)}$ (for V_{CC} = 4.75 V to 5.5 V) (see Note 5)		1.4			1.4			µs
Operating free-air temperature, T_A	TLC548C, TLC549C	0		70	0		70	°C
	TLC548I, TLC549I	–40		85	–40		85	

NOTES: 3. Analog input voltages greater than that applied to REF+ convert to all ones (11111111), while input voltages less than that applied to REF– convert to all zeros (00000000). For proper operation, the positive reference voltage V_{ref+} must be at least 1 V greater than the negative reference voltage, V_{ref-}. In addition, unadjusted errors may increase as the differential reference voltage, $V_{ref+} - V_{ref-}$, falls below 4.75 V.

4. This is the time required for the I/O CLOCK input signal to fall from V_{IH} min to V_{IL} max or to rise from V_{IL} max to V_{IH} min. In the vicinity of normal room temperature, the devices function with input clock transition time as slow as 2 µs for remote data acquisition applications in which the sensor and the ADC are placed several feet away from the controlling microprocessor.

5. To minimize errors caused by noise at the $\overline{CS}$ input, the internal circuitry waits for two rising edges and one falling edge of internal system clock after $\overline{CS}\downarrow$ before responding to control input signals. This $\overline{CS}$ setup time is given by the t_{en} and $t_{su(CS)}$ specifications.

electrical characteristics over recommended operating free-air temperature range, $V_{CC} = V_{ref+} = 4.75$ V to 5.5 V, $f_{clock(I/O)} = 2.048$ MHz for TLC548 or 1.1 MHz for TLC549 (unless otherwise noted)

PARAMETER			TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage		V_{CC} = 4.75 V,	I_{OH} = –360 µA	2.4			V
V_{OL}	Low-level output voltage		V_{CC} = 4.75 V,	I_{OL} = 3.2 mA			0.4	V
I_{OZ}	High-impedance off-state output current		$V_O = V_{CC}$,	$\overline{CS}$ at V_{CC}			10	µA
			V_O = 0,	$\overline{CS}$ at V_{CC}			–10	
I_{IH}	High-level input current, control inputs		$V_I = V_{CC}$			0.005	2.5	µA
I_{IL}	Low-level input current, control inputs		V_I = 0			–0.005	–2.5	µA
$I_{I(on)}$	Analog channel on-state input current during sample cycle		Analog input at V_{CC}			0.4	1	µA
			Analog input at 0 V			–0.4	1	
I_{CC}	Operating supply current		$\overline{CS}$ at 0 V			1.8	2.5	mA
$I_{CC} + I_{ref}$	Supply and reference current		$V_{ref+} = V_{CC}$			1.9	3	mA
C_i	Input capacitance	Analog inputs				7	55	pF
		Control inputs				5	15	

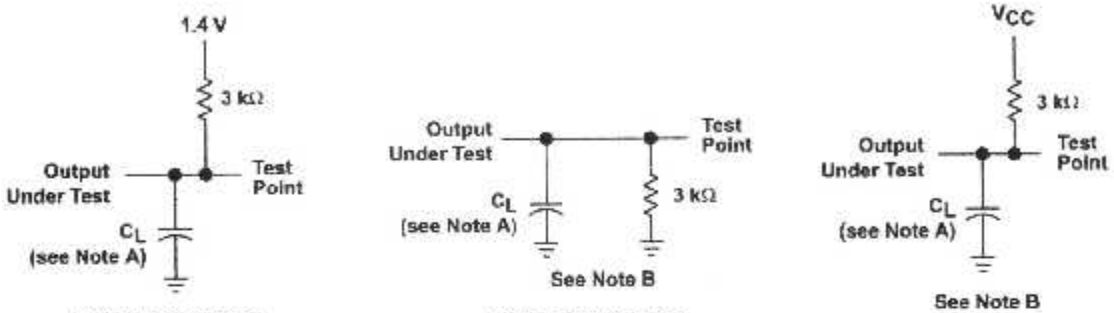
operating characteristics over recommended operating free-air temperature range, $V_{CC} = V_{ref+} = 4.75$ V to 5.5 V, $f_{clock(I/O)} = 2.048$ MHz for TLC548 or 1.1 MHz for TLC549 (unless otherwise noted)

PARAMETER		TEST CONDITIONS	TLC548			TLC549			UNIT
			MIN	TYP†	MAX	MIN	TYP†	MAX	
E_L	Linearity error	See Note 6			±0.5			±0.5	LSB
E_{ZS}	Zero-scale error	See Note 7			±0.5			±0.5	LSB
E_{FS}	Full-scale error	See Note 7			±0.5			±0.5	LSB
	Total unadjusted error	See Note 8			±0.5			±0.5	LSB
t_{conv}	Conversion time	See Operating Sequence		8	17		12	17	µs
	Total access and conversion time	See Operating Sequence		12	22		19	25	µs
t_a	Channel acquisition time (sample cycle)	See Operating Sequence			4			4	I/O clock cycles
t_v	Time output data remains valid after I/O CLOCK↓		10			10			ns
t_d	Delay time to data output valid	I/O CLOCK↓			200			400	ns
t_{en}	Output enable time	See Figure 1			1.4			1.4	µs
t_{dis}	Output disable time				150			150	ns
$t_{r(bus)}$	Data bus rise time				300			300	ns
$t_{f(bus)}$	Data bus fall time				300			300	ns

† All typicals are at V_{CC} = 5 V, T_A = 25°C.

NOTES: 6. Linearity error is the deviation from the best straight line through the A/D transfer characteristics.
7. Zero-scale error is the difference between 00000000 and the converted output for zero input voltage; full-scale error is the difference between 11111111 and the converted output for full-scale input voltage.
8. Total unadjusted error is the sum of linearity, zero-scale, and full-scale errors.

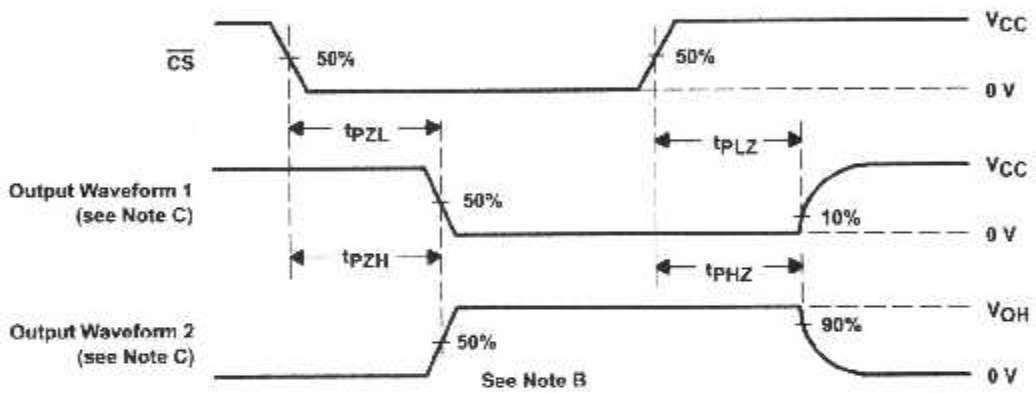
PARAMETER MEASUREMENT INFORMATION



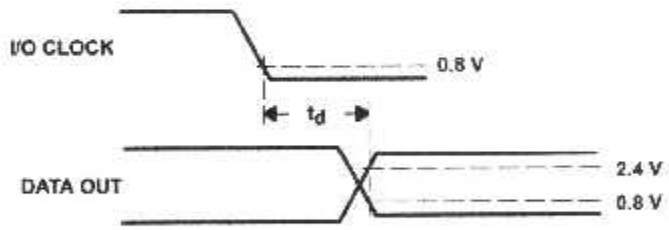
LOAD CIRCUIT FOR t_d, t_r AND t_f

LOAD CIRCUIT FOR t_{PZH} AND t_{PHZ}

LOAD CIRCUIT FOR t_{PZL} AND t_{PLZ}



VOLTAGE WAVEFORMS FOR ENABLE AND DISABLE TIMES



VOLTAGE WAVEFORMS FOR DELAY TIME

VOLTAGE WAVEFORMS FOR RISE AND FALL TIMES

NOTES: A. C_L = 50 pF for TLC548 and 100 pF for TLC549; C_L includes jig capacitance.
B. t_{en} = t_{PZH} or t_{PZL}, t_{dis} = t_{PHZ} or t_{PLZ}.
C. Waveform 1 is for an output with internal conditions such that the output is low except when disabled by the output control. Waveform 2 is for an output with internal conditions such that the output is high except when disabled by the output control.

Figure 1. Load Circuits and Voltage Waveforms

APPLICATIONS INFORMATION

simplified analog input analysis

Using the equivalent circuit in Figure 2, the time required to charge the analog input capacitance from 0 to V_S within 1/2 LSB can be derived as follows:

The capacitance charging voltage is given by

$$V_C = V_S \left(1 - e^{-t_c/R_t C_i}\right) \quad (1)$$

where

$$R_t = R_s + r_i$$

The final voltage to 1/2 LSB is given by

$$V_C\,(1/2\ \text{LSB}) = V_S - (V_S/512) \quad (2)$$

Equating equation 1 to equation 2 and solving for time t_c gives

$$V_S - (V_S/512) = V_S \left(1 - e^{-t_c/R_t C_i}\right) \quad (3)$$

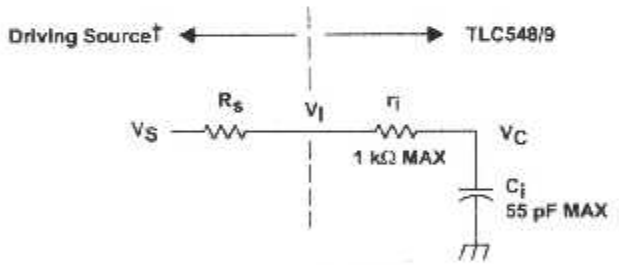
and

$$t_c\,(1/2\ \text{LSB}) = R_t \times C_i \times \ln(512) \quad (4)$$

Therefore, with the values given the time for the analog input signal to settle is

$$t_c\,(1/2\ \text{LSB}) = (R_s + 1\ \text{k}\Omega) \times 60\ \text{pF} \times \ln(512) \quad (5)$$

This time must be less than the converter sample time shown in the timing diagrams.



V_I = Input Voltage at ANALOG IN
V_S = External Driving Source Voltage
R_s = Source Resistance
r_i = Input Resistance
C_i = Input Capacitance

† Driving source requirements:
- Noise and distortion for the source must be equivalent to the resolution of the converter.
- R_s must be real at the input frequency.

Figure 2. Equivalent Input Circuit Including the Driving Source

PRINCIPLES OF OPERATION

The TLC548 and TLC549 are each complete data acquisition systems on a single chip. Each contains an internal system clock, sample-and-hold function, 8-bit A/D converter, data register, and control logic circuitry. For flexibility and access speed, there are two control inputs: I/O CLOCK and chip select ($\overline{CS}$). These control inputs and a TTL-compatible 3-state output facilitate serial communications with a microprocessor or minicomputer. A conversion can be completed in 17 µs or less, while complete input-conversion-output cycles can be repeated in 22 µs for the TLC548 and in 25 µs for the TLC549.

The internal system clock and I/O CLOCK are used independently and do not require any special speed or phase relationships between them. This independence simplifies the hardware and software control tasks for the device. Due to this independence and the internal generation of the system clock, the control hardware and software need only be concerned with reading the previous conversion result and starting the conversion by using the I/O clock. In this manner, the internal system clock drives the "conversion crunching" circuitry so that the control hardware and software need not be concerned with this task.

When $\overline{CS}$ is high, DATA OUT is in a high-impedance condition and I/O CLOCK is disabled. This $\overline{CS}$ control function allows I/O CLOCK to share the same control logic point with its counterpart terminal when additional TLC548 and TLC549 devices are used. This also serves to minimize the required control logic terminals when using multiple TLC548 and TLC549 devices.

The control sequence has been designed to minimize the time and effort required to initiate conversion and obtain the conversion result. A normal control sequence is:

1. $\overline{CS}$ is brought low. To minimize errors caused by noise at $\overline{CS}$, the internal circuitry waits for two rising edges and then a falling edge of the internal system clock after a $\overline{CS}\downarrow$ before the transition is recognized. However, upon a $\overline{CS}$ rising edge, DATA OUT goes to a high-impedance state within the specified t_{dis} even though the rest of the integrated circuitry does not recognize the transition until the specified $t_{su(CS)}$ has elapsed. This technique protects the device against noise when used in a noisy environment. The most significant bit (MSB) of the previous conversion result initially appears on DATA OUT when $\overline{CS}$ goes low.
2. The falling edges of the first four I/O CLOCK cycles shift out the second, third, fourth, and fifth most significant bits of the previous conversion result. The on-chip sample-and-hold function begins sampling the analog input after the fourth high-to-low transition of I/O CLOCK. The sampling operation basically involves the charging of internal capacitors to the level of the analog input voltage.
3. Three more I/O CLOCK cycles are then applied to the I/O CLOCK terminal and the sixth, seventh, and eighth conversion bits are shifted out on the falling edges of these clock cycles.
4. The final (the eighth) clock cycle is applied to I/O CLOCK. The on-chip sample-and-hold function begins the hold operation upon the high-to-low transition of this clock cycle. The hold function continues for the next four internal system clock cycles, after which the holding function terminates and the conversion is performed during the next 32 system clock cycles, giving a total of 36 cycles. After the eighth I/O CLOCK cycle, $\overline{CS}$ must go high or the I/O clock must remain low for at least 36 internal system clock cycles to allow for the completion of the hold and conversion functions. $\overline{CS}$ can be kept low during periods of multiple conversion. When keeping $\overline{CS}$ low during periods of multiple conversion, special care must be exercised to prevent noise glitches on the I/O CLOCK line. If glitches occur on I/O CLOCK, the I/O sequence between the microprocessor/controller and the device loses synchronization. When $\overline{CS}$ is taken high, it must remain high until the end of conversion. Otherwise, a valid high-to low transition of $\overline{CS}$ causes a reset condition, which aborts the conversion in progress.

A new conversion may be started and the ongoing conversion simultaneously aborted by performing steps 1 through 4 before the 36 internal system clock cycles occur. Such action yields the conversion result of the previous conversion and not the ongoing conversion.

FAIRCHILD
SEMICONDUCTOR®

www.fairchildsemi.com

LM336-2.5/LM336B-2.5

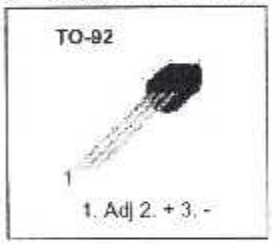
Programmable Shunt Regulator

Features

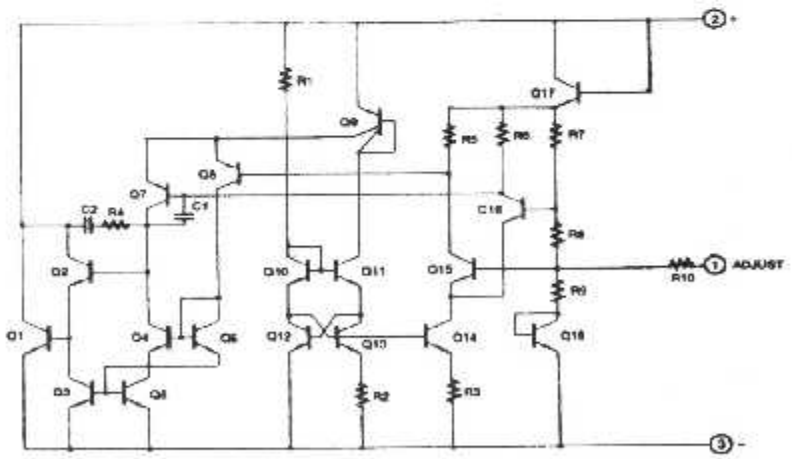
- Low temperature coefficient
- Guaranteed temperature stability 4mV typical
- 0.2Ω dynamic impedance
- ±1.0% initial tolerance available
- Easily trimmed for minimum temperature drift

Description

The LM336-2.5/LM336B-2.5 integrated Circuits are precision 2.5V shunt regulators. The monolithic IC voltage reference operates as a low temperature coefficient 2.5V zener with 0.2W dynamic impedance. A third terminal on the LM336-2.5/LM336B-2.5 allow the reference voltage and temperature coefficent to be trimmed easily. LM336-2.5/ LM336B-2.5 are useful as a precision 2.5V low voltage reference for digital voltmeters, power supplies or op amp circuitry. The 2.5V makes it convenient to obtain a stable reference from low voltage supplies. Further, since the LM336-2.5/LM336B-2.5 operate as shunt regulators, they can be used as either a positive or negative voltage reference.



Internal Block Diagram



Rev. 1.0.0

Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Reverse Current	I_R	15	mA
Forward Current	I_F	10	mA
Operating Temperature Range LM336-2.5/LM336B-2.5	T_{OPR}	0 ~ + 70	°C
Storage Temperature Range	T_{STG}	- 60 ~ + 150	°C

Electrical Characteristics

($0°C < T_A < +70°C$, unless otherwise specified)

Parameter	Symbol	Conditions	LM336-2.5			LM336B-2.5			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
Reverse Breakdown Voltage	V_R	$T_A = +25°C$ $I_R = 1mA$	2.44	2.49	2.54	2.465	2.49	2.515	V
Reverse Breakdown Change with Current	$\Delta V_R/\Delta I_R$	$T_A = +25°C$ $400uA \le I_R \le 10mA$	-	2.6	6	-	2.6	10	mV
Reverse Dynamic Impedance	Z_D	$T_A = +25°C$ $I_R = 1mA$	-	0.2	0.6	-	0.2	1	Ω
Temperature Stability	ST_T	$I_R = 1mA$	-	1.8	6	-	1.8	6	mV
Reverse Breakdown Change with Current	$\Delta V_R/\Delta I_R$	$400uA \le I_R \le 10mA$	-	3	10	-	3	12	mV
Reverse Dynamic Impedance	Z_D	$I_R = 1mA$	-	0.4	1	-	0.4	1.4	Ω
Long Term Stability In reference voltage	ST	$I_R = 1mA$	-	20	-	-	20	-	ppm/Khr

Typical Perfomance Characteristics

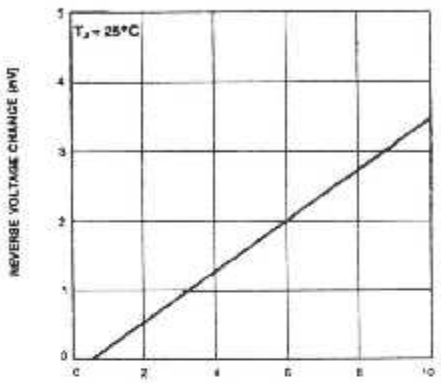


Figure 1. Reverse Voltage Change

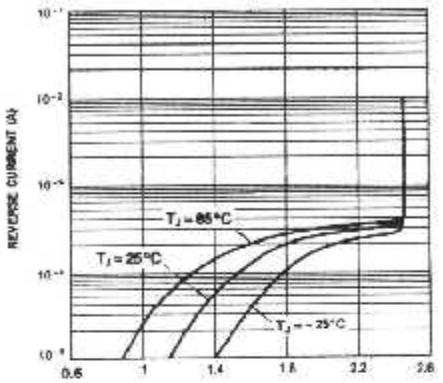


Figure 2. Reverse Characteristics

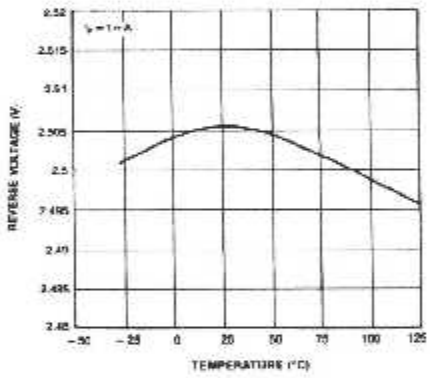


Figure 3. Temperature Drift

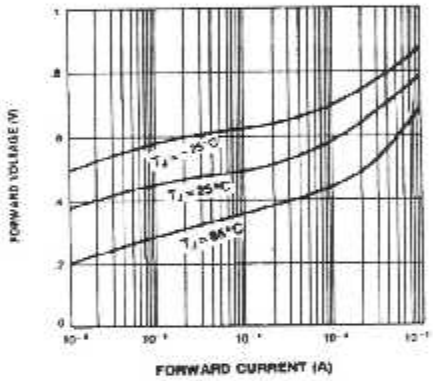


Figure 4. Forward Characteristics

Mechanical Dimensions

Package

TO-92