#230 - 2250 Boundary Road
Burnaby, BC, V5M 3Z3, Canada
Website: http://rmv.com
Email: customer@rmv.com

ITC232-A

Introduction au circuit intégré l'ITC232-A:

L'ITC232-A est un circuit intégré contenant une interface sériel/parallèle ainsi q'un nombre de fonctions préprogrammées qui trouvent application dans l'acquisition de données et dans le contrôle de systèmes. Il est d'accès facile et se connecte à un terminal sur le port sériel (EIA RS-232C). Il possède 32 lignes entrée/sortie arrangées dans 5 ports qui peuvent être lues ou écrites avec des ordres en ASCII extrêmement simples.
Cela permet le contrôle de la carte avec son propre programme ou avec n'importe quel logiciel commercial de communication (WINDOWS, Procomm, Telix, MAC 240, etc.).

Les seuls composants externes requis par L'ITC232-A sont un régulateur de tension et un quartz de 3.6864 MHz. En plus du convertisseur sériel/parallèle, l'ITC232-A possède plusieurs autres fonctions:

(1) Envoyer ou recevoir des données dans un format Décimal, Hexadécimal Binaire.
(2) Interruption à l'état bas ou haut via 2 broches IRQH et IRQL.
(3) 10-10000 Hz, 0-100% Modulation à largeur d'impulsion.
(4) Vitesse de transfers de 300 à 115200 Bauds.
(5) Quatre canaux permettant la lecture d'un condensateur ou d'une résistance relative.
(6) Trois ports pour la commande des moteurs pas à pas avec toute la logique nécessaire aux moteurs monophasique, biphasique et même pour le fonctionnement en demi-pas.
(7) Une touche-clé "Encore" (@) pour répéter l'ordre précédent sans retaper la commande.
(8) Un "Sur-écran" d'aide (un sommaire de tous les ordre est envoyé au terminal).
(9) Un état des fonctions est envoyé au terminal pour l'informer de la configuration active de tout port parallèle, le PWM ou le pas à pas.
(10) La possibilité d'utiliser la ligne téléphonique pour transférer les commandes à l'ITC232-A via un circuit modem type EF7910 ou 7911. La flexibilité, l'aisance d'emploi et fonctions multiples font de l'ITC232-A, l'instrument idéal pour l'acquisition de données, la robotique, l'instrumentation, et le contrôle.

CONSIDÉRATIONS GENERALES:

Tout au long de ce manuel Haut est employé comme synonyme de binaire 1 et Bas comme synonyme de binaire 0. Le terme "terminal" inclut des ordinateurs et le terme "périphérique" est employé pour tout circuit contenant l'ITC232-A.

La figure 1 montre le brochage de la carte I/O-232-8 (des versions a 8, 10 ou 12 bits sont disponibles) employant l'ITC232-A.

La figure 2 montre un exemple de branchement de l'ITC232-A à un terminal.
Un MAX232 est employé pour transmettre les informations du port sériel mais il peut être remplacé par des circuits à coût plus bas (1488 & 1489). Avant la mise sous tension la vitesse de transmission du terminal doit être la même que celle que vous avez sélectionné sur la carte avec la broche de "BAUD" (Haut = 9600, Bas = 300) ensuite à la mise sous tension, le message d'accueil apparaît.

Les paramètres de communications:

Choisir 300 ou 9600 Bauds sur le terminal selon la position du connecteur sur la carte (9600 ou 300).
Les autre paramètres sont toujours N, 8, 1 (aucune parité, 8 bits et 1 bit d'arrêt).
Quand on utilise un logiciel commercial de communications choisir l'option CR/LF car le LF seul est ignoré par l'ITC232-A.
Important: vérifier que la touche Backspace sur votre terminal envoie un code ASCII #8, autrement vous ne serez pas capable de corriger un ordre sans retaper entièrement la commande (dans certains programmes de terminaux, tel que WINDOWS le Backspace est généré avec un code Ctrl clé (Ex: sous l'émulation VT-100, Backspace est égal Ctrl - H). Tandis qu''avec un logiciel commercial les configurations CRAB (bin), CRAD (décimale) ou CRAH (hexa) travaillent mieux, la configuration CRAP (Programme) est optimisée pour contrôler l'ITC232-A dans vos propres programmes.

Commandes et résultats:

- **Les ordres sont TOUJOURS envoyés dans le format ASCII et ils doivent être suivis par un retour de chariot (CR ou ASCII (#13)).**
- **L'ITC232-A accepte les majuscules ou minuscules, les marques de ponctuation ou d'espaces peuvent être ajoutées pour la clarté (avec l'exception du point-virgule ; que est réservé pour la séparation des paramètres de commande).**
- **Une commande correcte est TOUJOURS accompagné d'une réponse "OK".**
- **Le dernier caractère envoyé au terminal est TOUJOURS le prompt (c'est utile de savoir dans son propre programme que l'ITC232 a exécuté un ordre).**
- **Seule les interruptions sont une exception; elles sont signalées seulement par "L" (RQL) ou un "H" (RQH).**
- **Un ordre erroné? n où? indique une erreur et n est le code d'erreur (de $1 à ¢$B, voir LA LISTE D'ERREURS). Le nombre de codes d'erreur est suivi par un message d'erreur à moins que la configuration CRAP soit en opération.**

Il y a 2 types de commandes: Les procédures et les fonctions. Un ordre de procédure exécute une action. Un ordre de fonction peut ou ne peut pas exécuter une action mais il abouti toujours à un résultat.

Une procédure qui abouti affiche un "OK" (un CR et un LF sont inséré avant le "OK" à moins que la configuration CRAP soit en opération. Une fonction qui abouti affiche par retour "OK" {$} valeur où {} signifie optionnel et $ s'applique seulement aux résultats en Hexadécimal (cela permet d'entrer le résultat en hexadécimal directement en une variable numérique dans certains langages de programmation).
Quand Valeur est un octet long, les résultats peuvent être demandés soit en Décimal (3 chiffres), soit en Hexadécimal (2 chiffres précédés par $) soit en Binaires (8 Chiffres dans deux groups de 4 bit (morceaux) séparés par un espace).
Les mesures des fonctions pouvant être supérieures à 255 sont TOUJOURS affichées dans le format Décimal (voir commandes pour moteurs PAS A PAS).
À la mise sous tension la configuration par défaut pour les résultats est CRAD (Décimal). Le format de défaut peut être changé à tout moment avec la commande CONFIGURER RÉSULTATS (CONFIGURE RÉSULTAT).
Pour inhiber le format par défaut pour une lecture, ajouter Le "B" ou "%" pour Binaire, "D" pour décimal ou "H" ou "$" pour Hexadécimal APRES l'ordre l'ordre (ex: PRA donnera la valeur dans le port A dans le format défaut. Ex: Le décimal. PRAB donnera la valeur ASCII binaire dans le port A, mais seulement une seule fois, à la lecture suivante PRA redonnera le résultat en décimal).
Quand on envoie la valeur d'un octet dans une commande, les trois formats peuvent être employé. Le format par défaut est le Décimal (vous pouvez aussi placer un D avant le nombre). Pour l'envoyer en Hexadécimal if faut le faire précéder d'un H ou un $, pour des nombres Binaires, le faire précéder par B ou %.
Si vous émettez des ordres en décimal vous pouvez employer des nombres simples, triple ou double (1 = 01 =1001). Cependant, en

Hexadécimal, deux chiffres DOIVENT être entrés ($F sont incorrect doit être $0F). Dans le binaire, tout 8 bits DOIVENT être précisés (B111 est incorrect, doit être B00000111). Tout nombre qui pourrait requérir plus que 1 octet (ex: la vitesse d'un moteur de pas à pas) DOIT être émis dans le mode décimal. Les commandes sont définies par la première lettre du mot: (ex: les ordres commençant avec P sont des ordres concernant les ports, S (STEP) sont des ordres concernant les moteurs pas à pas.
La touche Esc ou (ASCII (#27) à n'importe quel endroit de la commande efface l'ordre. Un nouveau prompt est alors émis.

La liste des commandes (dans l'ordre alphabétique):

Les caractères entre les deux <> sont les commandes, le reste du mot est pour un usage explicatif.
Les commentaires dans {} sont optionnels.
AGAIN COMMAND (Répéter la commande)
<@>gain. Ce caractère "@" répète la fonction ou procédure. Il répète au terminal le dernier ordre qui était juste avant le "@" (à moins que la configuration CRAP soit active) et reprend l'exécution. Si aucun ordre précédent n'étaient émis le "@" n'a aucun effet.

BAUD rate:(vitesse de transmission)
<B>aud <vitesse> donc la vitesse de transmission est de <300>, <600>, <1200>, <2400>, <4800>, <9600>, <19200>, <38400>, <57600> ou <115200> Bauds (exemple: B300[Enter] met l'ITC232-A a travailler a 300 Bauds).
Procédure type: Sélectionner la vitesse en Baud indépendamment du niveau présélectionné sur la carte. La commande est reconnu avant les changements de taux de baud. Etre certain que votre programme pourra suivre la vitesse sélectionnée sans risque d'envoyer des caractères erronés.
Utilisant la configuration CRAP à 115300 Bauds, ~2250 conversions analogiques peuvent être lues par seconde à partir d'un MC14501 branché sur la SPI ou aussi ~1500 lectures sur le port parallèle par seconde, ceci peut être réalisé en utilisant un programme compilé et la commande "@"(Se méfier des erreurs de lecture à grande vitesse de transmission, les erreurs sont plus nombreuses aux taux les plus hauts et aussi quand les câbles sont long.)

Le listing 1 est un exemple dans QBASIC qui vous donnera des éléments de base pour un programme élémentaire.

Configuration par défaut du format des résultats:

L'ITC232-A peut obtenir des résultats dans le mode Décimal, Binaire ou Hexadécimal. Par défaut, au moment du "reset" ou à la mise sous tension, le format est Décimal. On peut changer (à tout moment) le format.
Méthode pour configurer le format:
<C> onfigure <R> esults <A> SCII <B> Binary ou <D> ecimal ou <H> exadecimal ou <P> rogram.
La configuration CRAP permet une utilisation optimisée de l'ITC232-A quand on écrit un programme et diffère des autres commandes qui suivent.

- Ni CR ou LF ne doivent être insérées.
- Le format par défaut pour des résultats est en décimal (il peut être modifié par la commande B ou %, D et H ou $ après une commande).
- Les commandes suivantes sont dévalidées:
  - HELP (l'aide)
  - ERROR MESSAGES (Les messages d'erreur) (seulement ? n (n = nombre d'erreur) est envoyé au terminal).
  - PCp? (où p = A, B, C ou S).
  - S? (Configuration pour les moteurs pas à pas).
  - La valeur de la fréquence affichée au terminal quand un ordre PWM est émis.

Une tentative pour obtenir une information n'est pas valable avec la configuration CRAP active et il en résultera ce message d'erreur "error ?3".

HELP (l'aide sur l'écran):
<H>elp ou <?> fonction type.
Avec cette commande, apparaît un sommaire de tout les ordres. Pour faire défiler l'écran presser sur une touche. Pour quitter l'aide, presser sur la touche "Esc" on revient à ce moment là au prompt pour un nouvel ordre.

Dans certain logiciels de communication (particulièrement anciens) le buffer sur le port sériel est trop petit et une foie rempli, i envoient un XOFF pour dire au périphérique d'arrêter d'envoyer jusqu'"au moment ou le buffer s'est vidé et ensuite un XON pour demander l'envoie des données à nouveau. L'ITC232-A ne reconnaît pas le protocole XOFF-XON et vous pourriez perdre certain caractères du texte d'aide. Dans ce cas, il faut augmenter la taille du buffer a un minimum de 800 caractères.

Interrupts: (Les interruptions):

Les interruptions ne sont pas des ordres dans les sens strict du mot. Elles sont des caractères envoyé de l'ITC232-A au terminal pour signaler un événement survenant dans le périphérique même si L'ITC232-A est engagé dans une autre tâche telle que la commande des moteurs pas à pas ou générer un signal PWM. Il-y-a deux broches d'interruptions:
IRQL qui détecte une transition Haut vers Bas et IRQH qui détecte une transition Bas vers Haut.
Remarque: les interruptions sont seulement validées au moment du début du changement d'état; le niveau de tension de la broche n'a pas trop d'importance. Il suffit de polariser les broches IRQ à une tension inverse de leur détection (directement par l'intermédiaire d'une résistance de 10K).
Détectant le FRONT de montée ou de descente plutôt que des niveaux constants, cela autorise des points d'interruption. Pour cela, mettre la broche IQ à la tension inverse de ce qu"il détecte via une résistance de 10K, le signal de détection est amené à la broche IQ via un condensateur de 0.1uF qui est en parallèle avec une résistance de 100K.

Les broches d'interruption sont particulièrement utiles dans la robotique et le contrôle mécanique d'une fin de course ou un ordre d'arrêt doit être donné. Donc le IRQL envoie L au terminal et le IRQH envoie un H.

Le "OK", "CR", "LF" ou le prompt ne sont pas envoyés.

La priorité dans les commandes se suivent dans cet ordre:

(1) = IRQH
(2) = IRQL
(3) = Commande du moteur pas à pas
(4) = PWM (modulation à largeur d'impulsion) en sachant que simultanément IRQL et IRQH sont affichées.

LES PORTS:

L'ITC232-A a 6 ports.
Le premier est le port RS232 que l'on relie au terminal via les broches 232TX et 232RX (figure 1).
Les 5 autres ports peuvent être employés dans votre circuit et ils sont:
PA, PB, PC, PD et PS.
PA, PB et PC sont à usage général en 8 bit. Ces 24 broches peuvent être individuellement configurées comme entrée ou sorties (hautes impédance).
Dans ce cas là, les broches peuvent être adressées individuellement bien que l'état initial du port doit être configuré dans son intégralité.
PD est toujours un port de 4 bit d'entrées qui partage ses broches avec PS, un interface Synchrone Périphérique Sérielle appelée (SPI) qui peut être employée pour communiquer avec d'autres circuits tels qu"un registre à décalage parallèle/sériel.
Dans la figure 1, les broches PD et PS sont nommés PDx/PSx.
PD est toujours disponible, PS doit être configuré avant d'être employée (tandis que tout les autre ports peuvent être lus sans être configuré en premier.
Si vous tentez d'accéder PS avant de l'avoir configuré il en résultera une erreur "?2 Le port doit être configuré ou connecté e premier".
Quand une commande de lecture ou d'écriture est émise par PS ou PD vers le SPI, la transaction a lieu utilisant la configuration qui a été programmée auparavant pour le PS et ensuite l'information revient vers PD. Il est cependant possible d'employer les deux PD et PS dans une application mais ce n'est pas recommandé.

A la mise en route ou après le reset, tout les ports PA PB et PC sont configurés comme des entrées (hautes impédance), PDtes

toujours configuré comme des entrées et PS n’est pas configuré du tout.

Les commandes de ports:

Il-y-a 3 types de commandes pour les ports:

Configurer un port
Lire un port
Écrire dans un port

Ces trois commandes débutent avec un P

Configuration des ports A, B et C.

<P>ort <C>onfigurer <A> ou <B> ou <C> <Valeur>
Procédure type

Chaque broche peut être individuellement configurée comme une entrée ou une sortie selon le bit correspondant en sachant que la valeur 0 est employée pour les entrées et 1 pour les sorties.
Précédant la commande "Valeur" avec un H ou $ pour Hexadécimal, B ou $ ou Binaire et D pour décimal on inhibe la valeur par défaut.
Par exemple: pour configurer les 4 bits poids faible de PB comme entrées et les 4 bits poids fort comme des sorties, la commande suivante peut être donnée:
PCB $F0, PCB B1111 0000, PCB 240 ou PCB D240 (espaces permis mais non obligatoires).

Si vous essayez de configurer PD il en résultera une erreur:
"?A le port D est toujours un port a 4 bit d'entrée.

<P>ort <C>onfiguration <A> ou <B> ou <C> ou <S> ou <?> {B,%, D, H ou $}.
Fonction type
Revient à la commande de configuration des ports (cette commande n'est pas disponible quand la fonction CRAP est active).
Par exemple, après avoir configuré PB comme dans l'exemple au dessus ayant la configuration CRAB par défaut, PCB? vous donnera "OK 1111 0000" et PCB?D donnera "OK" 240".<S> s'applique au SPI (voir ci-dessous).

L'interface Périphérique Sérielle (SPI):

Dans l'intention d'économiser des pattes, beaucoup de fabricants de circuits intégrés produisent des circuits (telles que les convertisseurs AD/DA entre autres) qui communiquent par un interface synchrone sériel.
Le SPI de l'ITC232-A est conçu pour communiquer avec ces circuits et s'adapte aux divers protocoles de communications qu'""ils pourraient requérir. Le SPI opère à une vitesse fixe de 57.6 KHz.
Un registre circulaire de transfert en 8 bits est dedans l'ITC232-A et l'autre 8 (ou plus) bits sont dans le circuit périphérique. Donc avant de saisir les valeurs dans l'ITC232-A, ces valeurs doivent être saisies en dehors dans le périphérique en premier, puis transmise dans le SPI.
La valeur lue dans l'ITC232-A est 0 si aucune valeur n'avait été écrite au SPI auparavant.

Le port D qui partage des broches avec le SPI, est toujours actif.
Quand l'opération sur le SPI a lieu, la configuration correspondante est chargée, l'ordre est exécuté et le port revient à son état initial configuré à hautes impédance.

IMPORTANT:

(1) Polariser TOUJOURS la broche d'horloge du circuit périphérique à un potentiel positif ou négatif (en fonction du montage) via une résistance de valeur convenable (autrement la première transition d'horloge ne pourrait pas être détectée du tout). Si vous avez aussi besoin d'employer cette broche pour PD, placer un condensateur de 0.1 à 1 uF entre votre broche périphérique d'horloge et PD2/PS_CK.

(2) Pour utiliser la broche PS, la patte PD3/PS_VDD DOIT être polarisé au positif.

POUR CONFIGURER LE SPI:

<P>ort <C>onfigurer <S>erial <R>ead ou <W> rite ou <A>ll <V>aleur.
Procédure type.
Certains circuits sériels ne peuvent être seulement lus. D'autres ne peuvent être écrits et enfin, certains ne peuvent être ni lus ni écrits. Ainsi on pourra effectuer les commandes <R> <W> et <A> (<A> signifie les deux lire et écrire).

Quand on envoie une valeur, le SPI est configuré comme pour la dernière commande PCSW (si aucune configuration précédente n'est entrée le message suivant apparaît:
"?2 Port doit être configuré ou connecté en premier").
Quand on lit des données, le SPI est configuré comme pour la dernière commande PCSR.

Quand une opération de lecture a lieu, la configuration <R>ead est connecté, la dernière valeur écrite au SPI (ou 0 si aucune valeur n'a été écrite) est éliminée du SPI et la nouvelle valeur du périphérique est mesurée et envoyée au terminal.
Il est recommandé d'employer une broche du port parallèle pour sélectionner un périphérique quand il-y-a plus d'un interface branché sur le SPI.

<V>aleur précise si le SPI n'est pas connecté ou, la polarité d'horloge, si l'horloge et les données ne sont pas dans la phase ou et le sens de l'octet (certains circuits envoient des données MSB en premier tandis que d'autres envoient le LSB en premier). Le tableau ci dessous montre la fonction des bits différents dans <V>aleur:

Le bit 7 Doit être à l'état "1" pour valider le SPI.
Les bits 6, 5, 4, 3 sont inconséquents.

Le bit 2 Détermine la polarité de l'horloge (broche PD2/PS_CK).
POL 0 = l'horloge tire vers le bas
1 = l'horloge tire vers le haut

Le bit 1 Contrôle la phase d'horloge (PD2 / PS_CK broche).
PHASE 0 = En phase avec des données.
1 = En dehors de phase avec des données.

Le bit 0 Contrôle l'ordre des bits reçu par le SPI.
ORD C'est utile dans certains cas quand un périphérique envoie l'octet "backwards".
0 = l'ordre est conservé.
1 = l'ordre des bits est renversé (MSB<--->LSB)

Le tableau suivant montre toutes les possibilités:

bit	7	6	5	4	3	2	1	0	HEX	DEC
	Obligatoire		{	sans importance		}	POL	PHASE	ORD	
	1	0	0	0	0	0	0	0	$80	128
	1	0	0	0	0	0	0	1	$81	129
	1	0	0	0	0	0	1	0	$82	130
	1	0	0	0	0	0	1	1	$83	131
	1	0	0	0	0	1	0	0	$84	132
	1	0	0	0	0	1	0	1	$85	133
	1	0	0	0	0	1	1	0	$86	134
	1	0	0	0	0	1	1	1	$87	135

Pour déconnecter le SPI, mettre le Bit 7 á 0 (ou en décimal < 128 ou hex < $80).

IMPORTANT: la broche PD3/PS_VDD ne demande pas à être mise vers le plus (+) pour qu''une commande de configuration puisse être valable, mais il est obligatoire de la mettre au (+) avant de lire ou écrire au SPI (sinon l'erreur ?B sera transmise au terminal).

Comme avec les autreports, PCS? {nombre de base} donne <valeur> dans le format précisé {nombre de base} ou dans le format de défaut.

Écriture et lecture dans les ports:

Lecture sur un port parallèle:
<P>ort <R>ead <A> ou <B> ou <C> ou <D> {B,%, D, H,$}. fonction type.

Quand on lit un port parallèle qui a des broches configurées comme sorties, la valeur qu''on lit dans les bits correspond à l'état actuel de ces broches (0 si rien n'est écrit dans le port depuis la mise sous tension ou reset).

Écriture sur un port parallèle:
<P>ort <W>rite <A> ou <B> ou <C> {B, %, D, H, $} <Value>.
Procédure type:
Quand on écrit sur une broche configurée comme une entrée, la broche reste en haute impédance. Cependant, la valeur est emmagasinée dans un bit fantôme qui est présent derrière le bit port réel. Ainsi, si vous envoyez à une broche d'entrée la commande de configuration pour la changer en port de sortie, la valeur dans le bit fantôme sera aussitôt transférée à la broche correspondante.

Après un reset ou à la mise sous tension la valeur de tout bit fantôme est 0.

Écriture ou lecture du SPI:

<P>ort <R>ead <S>PI {B, %, D, H, $} Fonction type.
<P>ort <W>rite <S>PI {B, %, D, H, $} <valeur> procédure type.

Ces commandes travaillent dans un mode similaire à celui des ports parallèles avec les exceptions suivantes:

1) La broche PD3/PS_VDD DOIT être polarisée au (+). Sinon erreur "?B SPI demande a ce que la broche PD3/PS_VDD soit au (+), faire la modification et essayez à nouveau.

2) Le rapport entre la valeur de la broche et la configuration est différente.

3) Il n'y a aucun port fantôme qui emmagasine la dernière valeur écrite comme avec PA, PB et PC.

4) IMPORTANT: Le SPI peut être considéré comme un registre circulaire de changement sériel dans lequel 8 bits dans le port sont chargés par un circuit périphérique. Si l'on désire lire une valeur dans le SPI, celle-ci doit être écrite dans le registre du périphérique avant d'être transférée. Le SPI indiquera 0 si aucune indication n'a été transféré par le périphérique.

Il y a certaines situations particulières qui devraient être évitées ou manipulées avec soin:

4.a. Si on relie 2 circuits périphériques au SPI, un qui est en lecture et l'autre qui est en écriture et qu''ils sont connectés simultanément. Cela résultera dans la ré-écriture du premier quand celui-là sera en lecture. La valeur écrite sera la même que celle d'avant, mais les données sur le circuit périphérique sauteront de l'état bas à l'état haut comme les octets seront transférés à une vitesse de 57.6 KHZ. De plus, dans cette configuration d'écriture et de lecture il est presque certain que les circuits intégrés auront du mal à interpréter les données. Pour empêcher cela, il suffit d'employer le port PA, PB ou PC pour valider l'une après l'autre les circuits périphériques.

4.b. Relier un circuit périphérique qui possède la fonction écriture et lecture à l'ITC232-A, par exemple: le circuit Motorola(TM) MC145041 convertisseur Analogique/Numérique est très efficace, il permet de lire jusqu''a 11

tensions analogues différentes sans utiliser le port PA, PB ou PC, nous vous montrerons cela en détail.

Interfaçage entre un convertisseur analogique/digital et l'ITC232-A par la voie du SPI:

Dans nos plaquettes I/O-232-x (x=8, 10 ou 12), le MC145041[8 bits], MC145051[10 bits] et TLC2543[12 bits] sont tous connectés de façon identique:

Un nombre pour sélectionner le canal analogique doit être écrit dans le MC145041 tandis que la valeur de la dernière conversion est envoyée dehors. Ainsi, il est facile au SPI de lire d'une façon continue (polling) (sondand).

Sélectionner le canal analogique: Pour sélectionner le canal AN1 le nombre de port doit être b00010000 (16), pour AN2 il est b00100000 (32) AN3 = b00110000 (48), et ainsi de suite. La valeur pour sélectionner le canal analogique est = AN (canal) * 16.

La figure 3 montre comment relier le MC145041 à l'ITC232-A

Fonctionnement a distance par la ligne téléphónique:

L'ITC232-A peut être utilisé en étant branché sur un circuit modem économique du type EF7910 ou EF7911. La figure 4 montre un exemple du circuit de montage.

L'utilisation de l'ITC232-A par voie téléphonique s'opère à une vitesse de 300 Bauds (en mettant à la masse la broche de "BAUDS"). Un IRQL (est généré par le signal de sonnerie d'une ligne téléphonique comme vous pouvez le constater sur la figure 4) cette commande a lieu avant tout ordre et après la mise sous tension ou reset (Si un ordre est envoyé avant le IRQL alors L'ITC232-A fonctionnera normalement (et la fonction mise en route a distance sera inhibée).
Le premier IRQL n'envoie pas un L au terminal. Il donne dans PA.0 = Haut. C'est la fonction pour répondre au téléphone.
Le PA.0 est exclu fonctionnellement de PA; vous ne pourrez plus écrire ou configurer cette broche.

Pour limiter l'utilisation desports nécessaires pour la fonction modem à 1 (PA.0) ainsi que pour permettre de réaliser un systèmes simple de modem, il n'y a aucune détection de la porteuse. Mais au contraire un simple message: "Envoie une commande dans les 30 secondes ou l'ITC232-A raccrochera. Ce message se répétera si une commande n'est pas envoyée pendant 5 minutes, il suffi d'envoyer la commande "off suivi par le code ASCII (#7)" pour raccrocher et le prompt est envoyé vers le terminal quelques secondes après.
Le code ASCII (#7) non seulement fournit un signal audible mais est aussi utile à la platine pour reconnaître ce message particulier dans un programme propre.
Le délai de 7 secondes permet aux logiciels commerciaux de communications de détecter la présence d'une porteuse. Si l'ITC232-A ne reçoit pas un ordre dans les 30 secondes, il supposera que l'appel du téléphone était une erreur et exécutera la séquence suivante:

(1) Il enverra le code ASCII (#7) de déconnexion.
(2) Il raccrochera en mettant PA.0 = 0.
(3) Il reconfigurera PA.0 comme une entrée.
(4) Il se mettra en attente pour un nouvel appel.

Si la communication téléphonique est coupée par mégarde, ou sans aucune commande pendant 5 minutes, l'ITC232-A répétera le message ci-dessus. Trente secondes après le message, en absence de réponse, l'ITC232-A raccrochera et se mettra en attente pour un nouvel appel.

Vous pouvez simuler le fonctionnement par téléphone bien que l'ITC232-A soit directement relié à un terminal en utilisant une LED pour générer un PA.0 et en générant un IRQL à la suite d'un reset.

La modulation en largeur d'impulsion (PWM):

PWM peut être employé dans un nombre d'applications telles que la génération d'une tension analogique ou le réglage de la vitesse d'un moteur à courant continue (la réduction de vitesse a très peur de tours par minutes est possible sans réducteur de vitesse mécanique).
La possibilité de générer un signal de fréquence désiré, outre la création évidente de tons musicaux, cette caractéristique permet la production d'intervalles exact de temps.
l'ITC232-A possède une broche PWM. La fréquence et la largeur d'impulsion qui sort de cette broche peut être commandée par le terminal. La gamme de fréquence est 10-10,000 Hz et la largeur d'impulsion peut être établi de 0 à 100 % dans des intervalles de 1%. Cependant, plus la fréquence accroît, plus il devient difficile de mesurer les petits intervalles de temps exactement.
Ainsi, toute la gamme de réglage de largeur d'impulsion est disponible jusqu'à la fréquence de 220Hz.
La broche PWM peut être pré-réglé à une tension Haute ou Basse, à la mise sous tension ou au reset la broche PWM se trouve à l'état Bas.

Les commandes PWM:

<W>idth <fréquence>
La fréquence peut être réglée entre toute valeur de 10 à 10,000 Hz et doit être exprimée dans le format décimal. Une commande sans autre précision suppose une largeur d'impulsion de 50%.

<W>idth <fréquence> <;> <Duty cycle (largeur d'impulsion)>.

La fréquence est la même que ci-dessus, le <;> est obligatoire pour déterminer la largeur d"impulsion, le nombre indiqué est un entier de 0 à 100 indiquant le pourcentage du cycle durant lequel la broche PWM se trouve à l'état haut.

Notes:

1) Après avoir envoyé cette commande L'ITC232-A renvoie f=XXXXX (fonction indisponible si la configuration CRAP est validée). XXXXX est toujours un nombre décimal à 5 chiffres et la valeur affiché est la fréquence réelle que l'ITC232-A génère (cela tient au calcul arrondi et aux restrictions de la division de la fréquence de base). La fréquence réelle est obtenue par l'expression suivante: Af = 460800 / arrondi (460800 / Rf) ou Af est la fréquence réelle et Rf est la fréquence demandée. La précision de Af est directement liée à celle du quartz.

2) Durant la commande d'un moteur de pas à pas:
(a) Si WL ou WH sont actifs alors les broches ne changent pas d'état.
(b) Si PWM est actif, la broche PWM est amenée à la masse durant la commande.

(3) Si la largeur d'impulsion est égale à = 0 ou 100, alors la fréquence n'a plus d'importance, la broche PWM restera à l'état Bas ou Haut.

(4) La largeur d'impulsion peut varier dans 1% intervalles, cependant plus la fréquence accroît plus il devient progressivement difficile de générer de très petites largeurs d'impulsions ou de très grande largeurs d'impulsion. Si pour une fréquence très haute, vous demandez une largeur d'impulsion difficile à réaliser, l'ITC232-A vous enverra un message d'erreur "? Fréquence trop haute pour pouvoir réaliser la largeur d'impulsion demandée".

Pour les fréquences les plus hautes on ne peut moduler la largeur d'impulsion que autour des 50% tandis qu'en dessous de 220 Hz on peut moduler de 1% à 99%.

<W> idth <L>
Force la broche PWM à l'état Bas.
Cet ordre est intérieurement émis à la mise sous tension ou reset.

<W>idth <H>

Force la broche PWM à l´état Haut.

<W>idth <?>
Revient à la dernière commande <W>

Resetting l'ITC232-A:

Le <RESET>
Cet ordre est par lui même explicatif. Il a le même effet qu''un reset général sur la platine.

La mesure des résistances ou des condensateurs avec l'ITC232-A:

<R>esistance <0> ou <1> ou <2> ou <3>.
0-3 représentent les bits ou broches sur le port C.
L'impédance d'entrée sur ce port C est extrêmement haute.
Ainsi, si un circuit RC est branché à unbroche d'entrée comme vous pouvez le voir sur la figure 5, alors la constante de temps du circuit peut être déterminée en mesurant le temps qu''il faut pour atteindre un transition Basse vers une transition Haute.
A partir d'une broche donnée le point de transition est constant si la tension VDD est maintenue;
Des valeurs de capacités relatives ou de résistances peuvent être lues.

Les valeurs mesurées d'unebroche ne peuvent pas être mise en comparaison avec celle d'une autre broche parce que les points de transition ne sont pas identiques.
La séquence employée par l'ITC232-A pour lire le contenu d'une broche Rn est celle-ci:

1) Le port choisi est mis à la masse pendant un court instant pour décharger le condensateur.

2) La broche est ensuite configurée en entré, une boucle de temps reliée à l'horloge interne mesure le temps requis pour que la broche devienne à l'état haut.

3) Le résultat est envoyé au terminal.On peut remarquer que la valeur affichée au terminal est linéairement proportionnelle au temps de charge du condensateur C ou de la résistance.

Il y a un rapport linéaire avec les valeurs de C ou RÉviter de lire la valeur de condensateurs électrolytique (trop imprécis) choisir plutôt les condensateurs plastique.
La valeur est linéaire aussi bien pour les capacités que pour les résistances, pour un rapport entre les deux de 10 à 32,767 unité relatives, la précision peut atteindre 0.5% ceci est en rapport direct avec les composants utilisés.
Si la valeur relative entre le condensateur et la résistance dépasse lerapport de 10 sur 32,767, le terminal recevra un message d'erreur: "?7 (Time out error (valeur en dehors de la plage))"
La gamme de résistance qui peut être mesurée va de 200 Ohms à 10 Mégohms enutilisant différent condensateurs pour le changement de gamme.

Nous déconseillons de mesurer des résistances en dessous de 500 Ohms. La valeur mesurée est moins précise en dessous de 500 Ohms. Deuxièmement, lecircuit intégré pourrait être endommagée définitivement par le courant de passage qui passe par la broche quand la command de court circuit du condensateur pour décharge avant la mesure est donné.

Le port n'a pas besoin d'être configuré comme une entrée pour obtenir une valeur.
La commande <R> esistance suffit à elle-même.
Cependant, garder dans l'esprit que:

(1) Si vous changez de broche de lecture pour lire la valeur de résistance, il pourrait y avoir des différences aussi importante que 10% dans la lecture (dépendantdu type de condensateur et sa valeur (ex: un condensateur Siemens MKT de 0.47 uF donne une précision de lecture meilleure de 1%). Ainsi, nous recommandons fortement de garder la valeur et configuration de la broche inchangée entre les mesures.

(2) Un circuit branché au port pourrait interférer avec la lecture obtenue à moins que son impédance soit très haute par rapport à la résistance à mesurer.

(3) Changer la fréquence de la broche PWM dans une large gamme entre les mesures des différentes résistance pourrai conduire à des différences dans la valeur mesurée.

Interfaçage pour les moteurs pas a pas:

Un moteur pas à pas est conçu de telle sorte qu''il tourne d'un pas à la fois.
Cela permet le déplacement précis et absolue du rotor. La figure 5A montre une version d'un moteur de pas à pas. Le rotor consiste en un aimant permanent entouré par les pôles du stator. La polarité magnétique du stator peut être changée selon le courant passant. Le moteur dans cet exemple tourne avec trois séquences d'alimentation. La première séquence AB / CD / BA / DC (BA est la même séquence qu''AB mais les flux magnétique est dans la direction contraire) comme vous pouvez le voir dans la figure 5B.

Cette séquence est appelée "une-phase" ou "monophasique" parce que dans tout les cas une seule phase est alimentée.

La deuxième possibilité est d'alimenter 2 phases c'est le "deux-phases" ou "biphasique". Il est représenté dans la figure 5C. Cette configuration, est celle que l'on emploie le plus communément.

Enfin, la troisième option est le fait d'alimenter une phase, puis deux, ensuite une etc... comme vous pouvez le voir sur la figure 5D. Cette séquence divise l'angle par deux et est appelée "demi pas".

Tout les trois modes sont disponibles dans l'ITC232-A comme nous le verrons ultérieurement.

L'angle par pas typique des moteurs sont 15° (24 pas par tour), 3.75° (96 pas par tour), 3.6° (100 pas par tour) et 1.8° (200 pas par tours).

Quatre bits d'un port peuvent être employés pour commander un moteur de pas à pas.
Par exemple, si nous branchons les sorties A, B, C et D par l'intermédiaire d'un buffer convenable à la sortie d'un mot binaire de 4 bit (nibble (morceau)) venant de l'ITC232-A, le raccordement se ferait comme suit:

CABLAGE C D B A
BIT 3 2 1 0

Dans le mode monophasique (figure 5B), la séquence du moteur devra être 0001, 1000, 0010, 0100 (tournera dans une direction et pour l'inverse de la séquence il tournera dans le sens opposé).

Dans le mode biphasique (figure 5C), la séquence est 1001, 1010, 0110, 0101. (ou en extrapolant celui-ci nous avons le mode demi pas (figure 5D) avec la séquence suivante: 0001, 1001, 0010, 0110, 0100, 0101. Une fois que la dernière valeur est sortie, la séquence est répétée. Le fonctionnement peut démarrer à n'importe quel moment de la séquence pourvu que l'ordre de la séquence soit maintenu.

Il suffit d'utiliser 4 transistors montés en H comme vous pouvez le voir sur la figure 7.

Bien que l'alimentation peut se faire sans problèmes en utilisant 8 transistors conventionnels par moteur, une solution beaucoup plus efficace est possible en utilisant le circuit intégré L298 de SGS.
Un autre cablage simplifié qui utilise 5 fils au lieu de 8 est celui représenté par la figure 8.

L'ITC232-A contient toute la logique nécessaire pour utiliser tout les moteurs pas à pas en utilisant des ordres très simples.

1) Les moteurs pas à pas sont contrôlés par les 4 bits supérieurs de PA, PB ou PC.

2) Le nom du moteur équivaut au nom du port auquel il est branché.

3) **Les moteurs pas à pas doivent être connectés (et configurés) avant emploi.**

4) **Pour valider la rotation d'un moteur pas à pas, mettre 4 bits supérieur en entrées (hautes impédance).**

5) **La configuration s'applique à tous les moteurs en même temps (vous ne pouvez pas avoir 2 ou 3 moteurs de type différent).**

6) **Si la configuration est changée, la dernière valeur écrite au port est retenue. Ainsi la vitesse d'un moteur pas à pas peut être changée sans perdre les mesures en cours.**

7) **En utilisant les moteurs pas à pas la commande PWM reste inactive. Si la broche PWM était à l'état Haut (WH) ou l'état Bas (WL) elle ne changera pas d'état. Si la broche PWM générait une fréquence, à la mise en route des moteurs pas à pas la broche PWM se mettra à l'état bas.**

Pour valider et configurer un moteur pas a pas:

<S>tepper <E>nable <A> ou <B> ou <C> <M>onophasic ou <B>iphasic ou <H>alf mesure <vitesse> <;> <temps d'arrêt>

<A>, <B> et <C> réfère au 4 bits supérieurs des ports correspondants.
<M>onophasic, <B>iphadic et <H>alf mesure (demi pas) réfère aux modes vu ci-dessus.
La <vitesse> est en pas par seconde second (10 à 4,000).
Le <temps d'arrêt> est une valeur en décimal de 0-255 décimal représentant combien de temps le dernier pas restera alimenté.

Ex: SEAB500;10 fera tourner le moteur de 500 pas /s (2ms par pas) et le dernier pas sera alimenté pendant une période de 2 * 10 = 20 ms) Cela sert à freiner le moteur l'empêchant ainsi de continuer sa rotation entraîné par la force d'inertie (plus le moteur est grand et plus la vitesse la plus grande, plus cette valeur devrait être grande). Au début essayer avec 10% de la valeur de vitesse.

Une fois que la configuration pour un moteur pas à pas a été entrée, vous pouvez connecter un second ou troisième moteur pas à pas, pour valider ces autres moteurs, seule la commande <S>tepper <E>nable <A> ou <B> ou <C> est nécessaire.

L'erreur maximale de vitesse pour un pas à pas est de 2.2%.

Pour dévalider un moteur pas à pas:
<S>tepper <D>évalider <A> ou <B> ou <C>.
À noter que vous pouvez dévalider les trois moteurs pas à pas mai s la configuration est retenue dans la mémoire. Ainsi, une nouvelle commande <S>tepper <E>nable <A> ou <B> ou <C> revalidera le fonctionnement des moteurs avec la configuration qui était au a préalable mémorisée.

Pour demander l'affichage de la configuration des moteurs:
<S>tepper <?> {B ou % ou D ou H ou $} <S>tepper <E>nable <?> {B ou % ou D ou H ou $"}.
Ces ordres affichent sur le terminal la dernière configuration programmée pour les moteurs.
Cette commande est inactive si la configuration CRAP est active.
{le B, % D, $ ou H} précise le format pour la dernière valeur sur le port (si vous ne précisez pas, cette valeur sera affichée dans le format de la configuration défaut).
Vous pouvez reconfigurer port avec PC p nr même si le moteur pas à pas correspondant est connecté, et vous pouvez même écrire aux broches en faisant une commande PW p nr.

Pour faire tourner un moteur pas a pas:

<S>tep <A> ou <B> ou <C> <L>eft <R>ight <nombre de pas>.

<L>eft ou <R>ight sont mnémoniques qui précisent la direction de rotation.
<nombre de pas> peur être tout nombre de 0 à 65535 (une valeur de 0 est valable et le résultats est pas d'action).

Pour arrêter un moteur pas a pas:

Il se peut que vous désirez arrêter un moteur pas à pas, bien qu''il n'est pas encore fini son cycle, il suffit pour cela d'envoyer u commande:
"barre d'espace" our code (ASCII (#32)), ou "S", "s", ">", Esc (ASCII (#27)) ou Entrée (ASCII (#13)).
Cette action a pour résultat d'envoyer au terminal par l'intermédiaire de l'ITC232-A le reste de pas qui n'a pas été exécuté, ce résultat est toujours un nombre à 5 chiffres en décimal suivi du message "step to go" (pas qui restent à faire).

Si vous désirez utiliser plus de 3 moteurs pas à pas:

Cela peut être aisément fait par le multiplexage des ports des moteurs pas à pas et en utilisant une broche du port parallèle pour sélectionner le régulateur du moteur actif (si un régulateur L298 est employé (figure 9), relier ENA et ENB à la broche à contrôler).

BROCHAGE:

(1) Reset

(2) IRQL: interruption à l'état bas.

(3 & 40) VDD: +4.5 +5.5 Volts référencés à VSS.

(4-11) PA0. . LE PA7: Le port parallèleA. PA0 est exclu du port A quand on utilise L'ITC232-A par voie téléphonique. PA4. PA7 sont employés par le port pour la commande du moteur pas à pas A.

(12-19) PB0. .PB7: Le port parallèle B. PB4. . PB7: sont employés par le port pour la commande du moteur pas à pas B.

(20) VSS: Le voltage numérique le plus bas branché A' L'ITC232-A.

(21-28) PC0. .PC7: Le port parallèle C. PC0. . PC3: utilisés pour la mesure des résistances et des condensateurs. PC4. . PC7: sont employés par le port pour la commande du moteur pas à pas C.

(29) 232 RX: RS232 reçoit les données sérielles du terminal.

(30) 232 TX: RS232 transmet les données sérielles au terminal

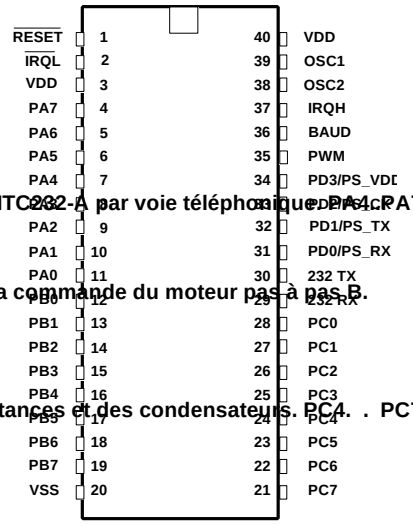


Fig. 1

(31) PD0/PS_RX: broche commune à PD (toujours 4 entrées) et PS (l'interface Périphérique Sérielle ou SPI).

(32) PD1 /PS_TX: broche commune à PD et PS.

(33) PD2 / PS_CK: broche commune à PD et PS.

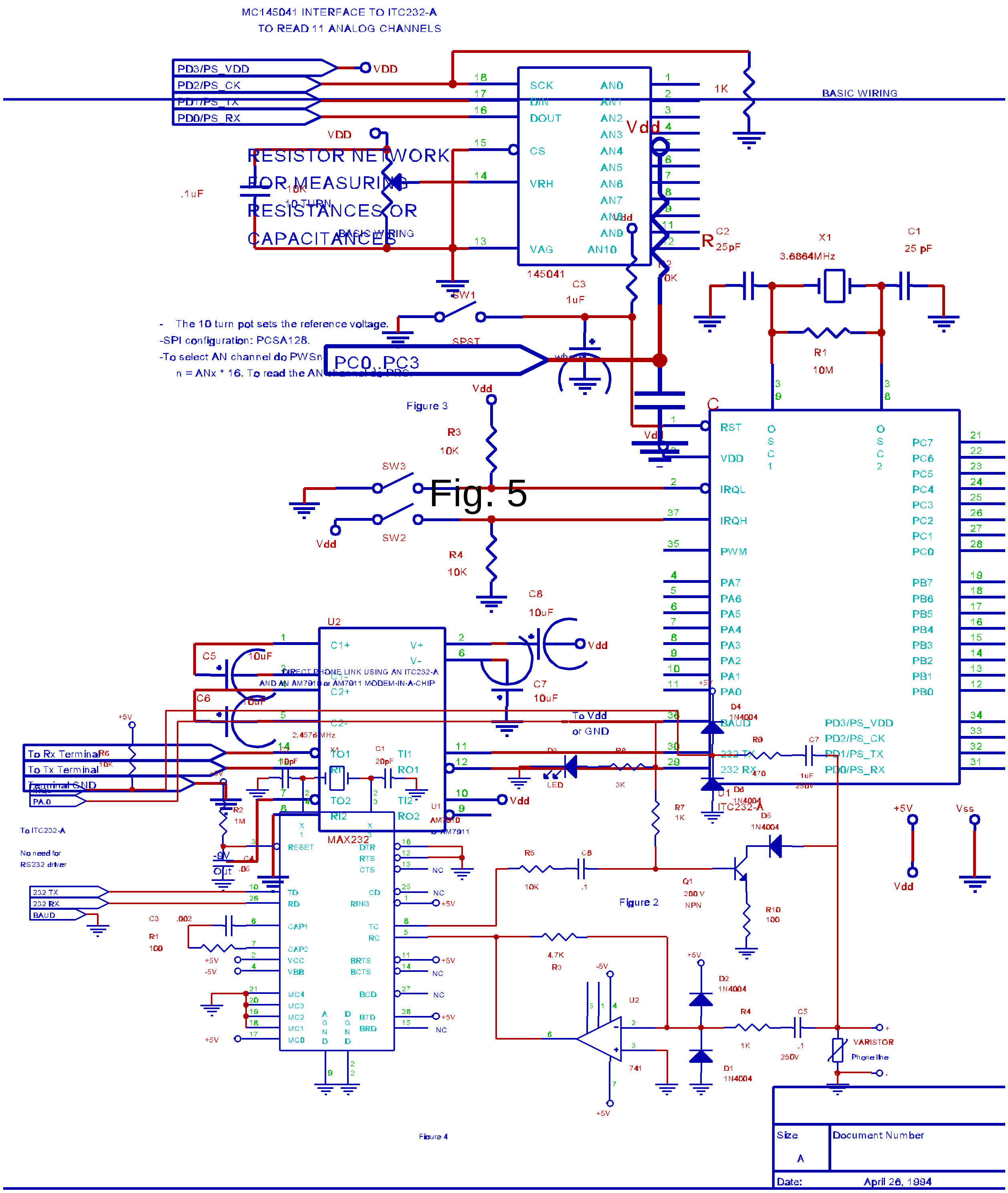
(34) PD3 / PS_VDD broche commune à PD et PS

(35) PWM: Sortie de modulation de largeur d'impulsion.

(36) BAUD: Sélectionne la vitesse de transmission 300 Bauds, 9600 Bauds.

(37) IRQH: Interruption haute.
(38) OSC1: broche pour le quartz.

(39) OSC2: idem


MC145041 INTERFACE TO ITC232-A
TO READ 11 ANALOG CHANNELS
BASIC WIRING
PD3/PS_VDD
PD2/PS_CK
PD1/PS_TX
PD0/PS_RX
VDD
SCK
DIN
DOUT
CS
VRH
VAG
AN0
AN1
AN2
AN3
AN4
AN5
AN6
AN7
AN8
AN9
AN10
145041
RESISTOR NETWORK
FOR MEASURING
RESISTANCES OR
CAPACITANCES
10K
10 TURN
.1uF
1K
C3
1uF
SW1
SPST
- The 10 turn pot sets the reference voltage.
-SPI configuration: PCSA128.
-To select AN channel do PWSn
n = ANx * 16. To read the AN channel do PRS.
PC0..PC3
Figure 3
Vdd
R3
10K
SW3
SW2
Fig. 5
R4
10K
C2
25pF
X1
3.6864MHz
C1
25 pF
R1
10M
RST
VDD
IRQL
IRQH
PWM
PA7
PA6
PA5
PA4
PA3
PA2
PA1
PA0
BAUD
232 TX
232 RX
PC7
PC6
PC5
PC4
PC3
PC2
PC1
PC0
PB7
PB6
PB5
PB4
PB3
PB2
PB1
PB0
ITC232-A
C8
10uF
C7
10uF
U2
C1+
C1-
C2+
C2-
V+
V-
C5
C6
10uF
DIRECT PHONE LINK USING AN ITC232-A
AND AN AM7910 or AM7911 MODEM-IN-A-CHIP
To Vdd
or GND
To Rx Terminal
To Tx Terminal
Terminal GND
TO1
TO2
RI1
RI2
TI1
TI2
RO1
RO2
MAX232
LED
3K
R7
1K
To ITC232-A
No need for
RS232 driver
RESET
DTR
RTS
CTS
TD
RD
CD
RING
CAP1
CAP2
VCC
VBB
TC
RC
BRTS
BCTS
BCD
BTD
BRD
MC4
MC3
MC2
MC1
MC0
AGND
DGND
R5
10K
Figure 2
Q1
200 V
NPN
R10
100
+5V
Vss
Vdd
4.7K
R3
741
D2
1N4004
D1
1N4004
R4
1K
C5
.1
250V
VARISTOR
Phone line
Figure 4
Size
A
Document Number
Date:
April 26, 1994

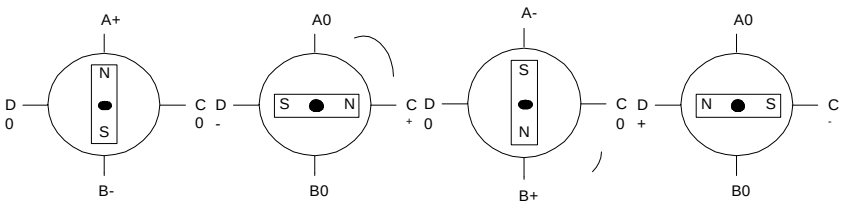


Fig. 5B

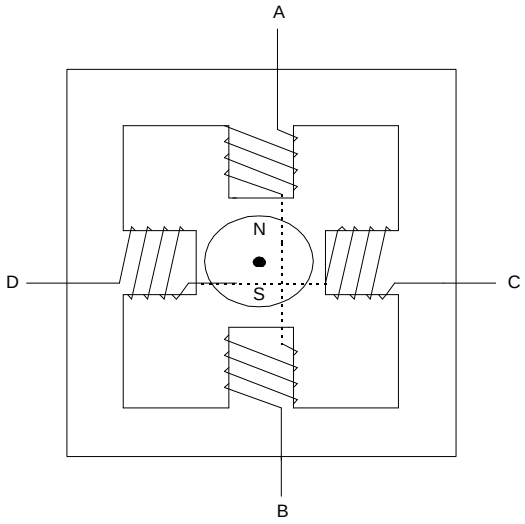


Fig. 5A

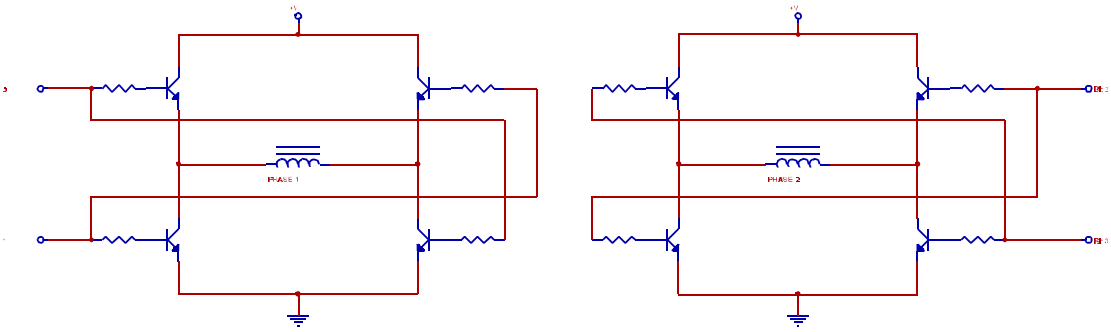


Fig.7

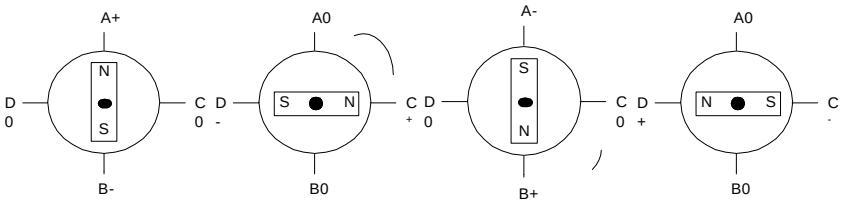


Fig. 5C

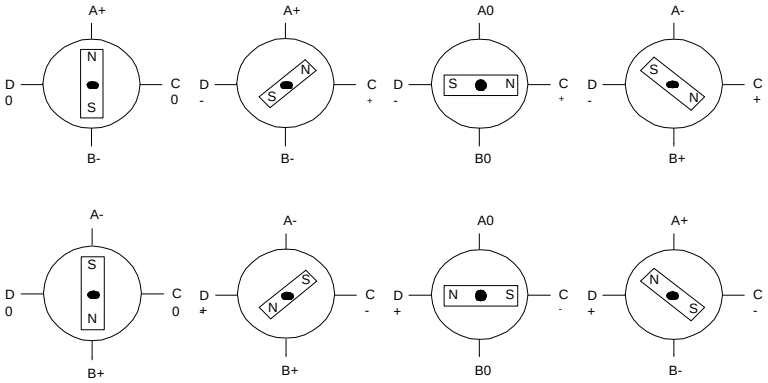


Fig. 5D

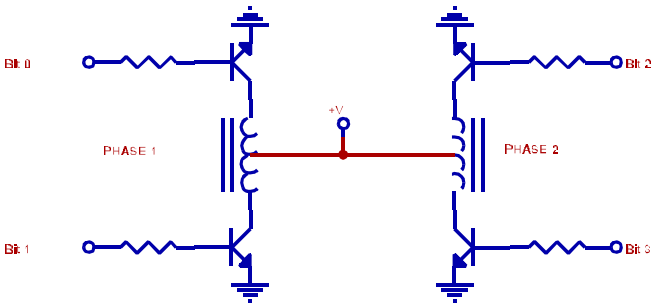

Bit 0
Bit 2
+V
PHASE 1
PHASE 2
Bit 1
Bit 3

Fig. 8

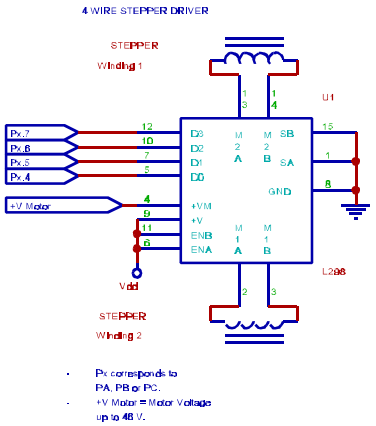

4 WIRE STEPPER DRIVER
STEPPER
Winding 1
U1
Px.7
Px.6
Px.5
Px.4
+V Motor
Vdd
GND
L298
STEPPER
Winding 2
Px corresponds to
PA, PB or PC.
+V Motor = Motor Voltage
up to 46 V.

Fig. 9