STI2D SIN

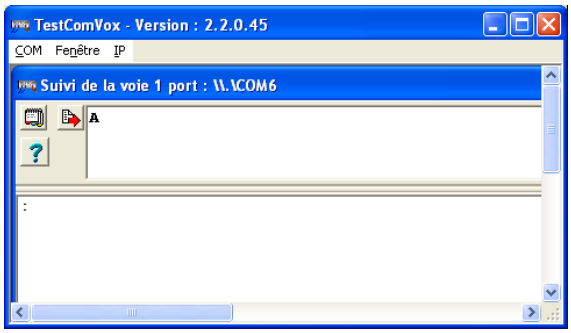
ANALYSE TRAMEs LIAISON SERIE

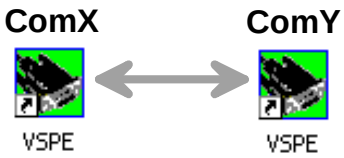
1 Objectifs de ce travail

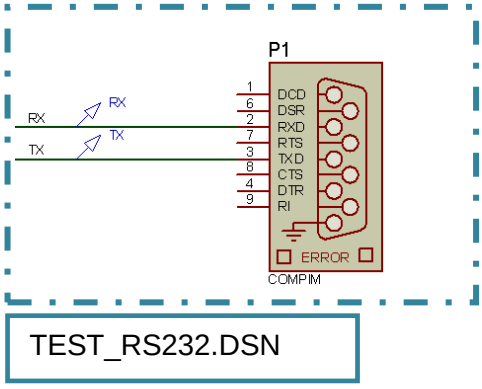
Nous allons étudier dynamiquement la liaison série en utilisant la simulation. La mise en œuvre des liaisons séries simulées et de TestCom est expliquée dans le document SIMULATION_LIAISON_RS232 il est conseillé de s'y référer si besoin.

2 Mise en situation de la liaison simulée par Proteus

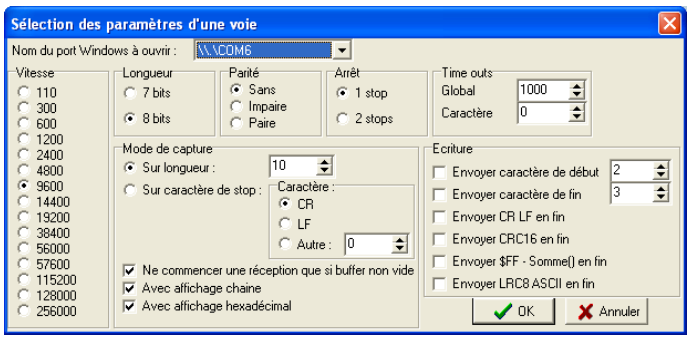
Nous allons établir une communication entre le logiciel de simulation Proteus schéma TEST_RS232.DSN et TestCom via VSPE.

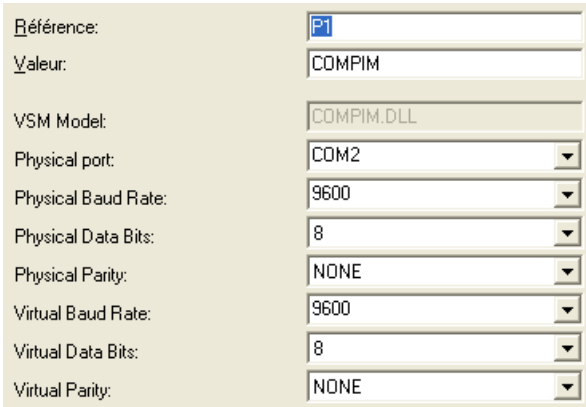






Configuration des deux logiciels :





Q1 : Sur quel numéro de port com est connecté le composant COMPIM de Proteus ?

Q2 : Sur quel numéro de port com est connecté TestCom ?

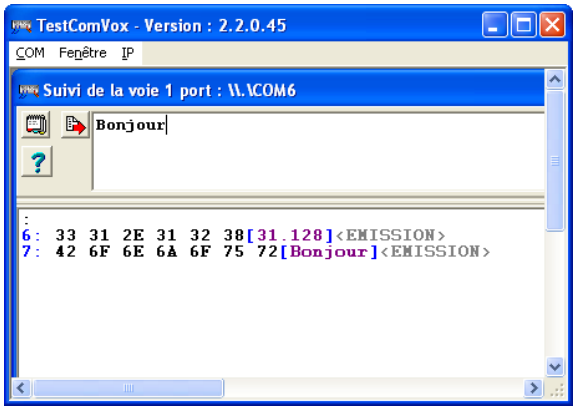
Q3 : Notez les numéros de port com pour votre situation sur votre poste informatique :

STI2D SIN

Pour établir la communication nous utilisons une simulation en mode *INTERACTIVE*. Pour la faire fonctionner il faut la mettre en marche en appuyant avec la touche [Barre Space], elle se lance pour le délai prévu de 10s. Seule la fenêtre de simulation au dessus duquel est la souris se met en marche. Il est donc possible de faire plusieurs enregistrements dans des fenêtres différentes pour pouvoir à loisir observer et analyser les résultats.

Pendant ces 10s (vous pouvez bien sûr modifier cette valeur) il suffit d'envoyer des messages avec TestCom directement depuis la fenêtre de gestion.

Ici deux messages ont été envoyés 'à la main' pendant le fonctionnement de la simulation en mode interactive. Les deux messages '31.128' et 'Bonjour' sont visibles.

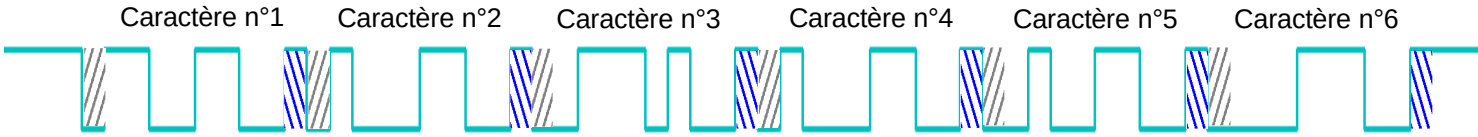


Q4 : Donner les dates d'envoi des deux messages en utilisant les outils de graphes de proteus, les curseurs en particulier.

Q5 : Le premier message '31.128' est composé de 6 caractères. Donner le nombre de bits envoyés pour chaque caractère. Puis pour le message complet.

Q6 : En utilisant les curseurs donner la durée du message complet. Puis la durée d'un bit et enfin la vitesse de transmission. Comparer avec la fréquence théorique.

Analyse de la trame reçue pour le premier message.



Les bits de start sont hachurés en gris et les bits de stop en bleu. Attention les data sont envoyées dans la trame bits de poids faible en tête.

Q7 : Remplir le tableau ci-dessous et vérifier l'intégrité du message.

Code Ascii	IDLE	Start	b0	b1	b2	b3	b4	b5	b6	b7	Stop	IDLE
$ 3		0									1	
$ 1		0									1	
$.		0									1	
$ 1		0									1	
$ 2		0									1	
$ 8		0									1	

Méthode pour analyser les trames.

La méthode utilisée est efficace et simple. La trame à analyser est recopiée dans un logiciel de traitement d'image. Puis les bits de start et de stop sont marqués.

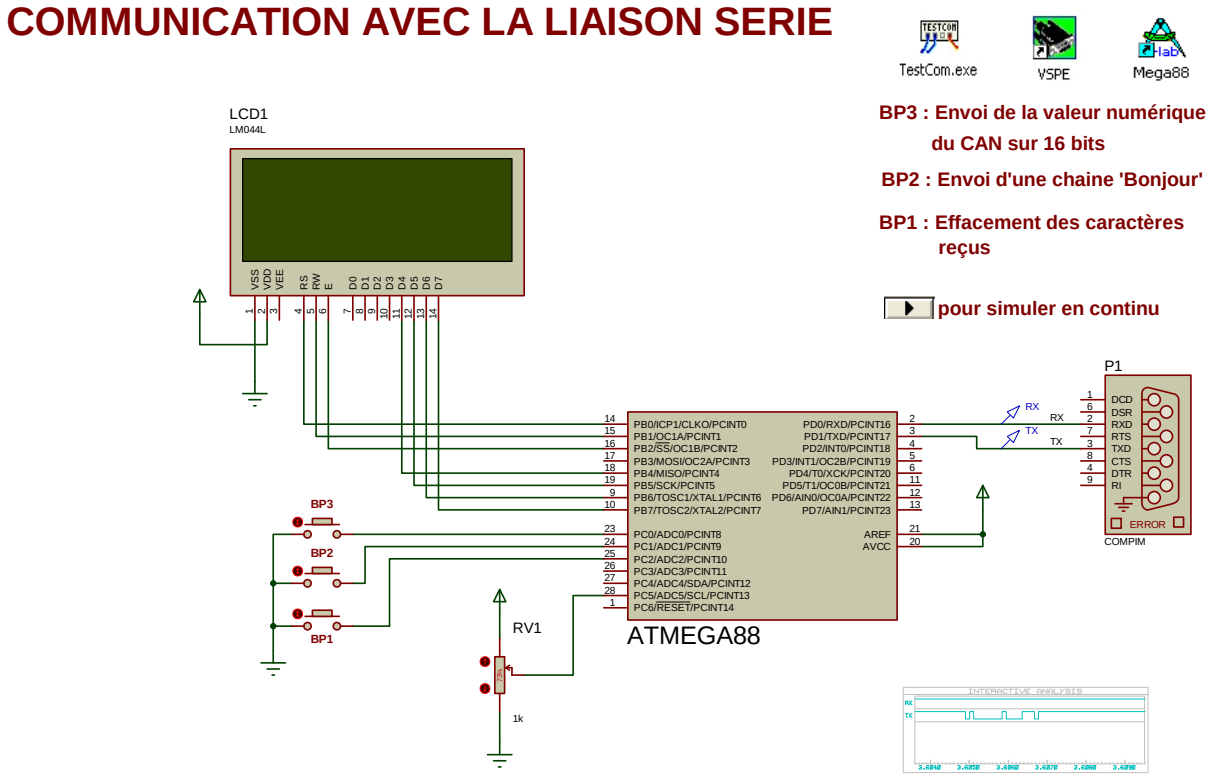
Le découpage des data est en général évident il suffit alors de le relire à l'envers pour retrouver la valeur de l'octet dans l'ordre normal des bits.

3 Échanges entre applications

Pour mettre en œuvre l'échange de données entre applications il nous faut maintenant travailler un schéma plus complet, dans ce schéma une application à base de microcontrôleurs communique vers l'extérieur et échange de l'information avec la liaison série.

Noter à ce stade que le composant simulable COMPIM de Proteus représente à la fois la liaison série RS232 et l'adaptation nécessaire aux niveaux logiques requis.

Le programme de l'application réalisé par l'EDI AVRco[1] réalise les trois actions indiquées en fonction de l'appui des trois boutons poussoirs BP1-BP3.

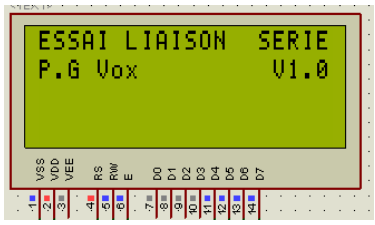


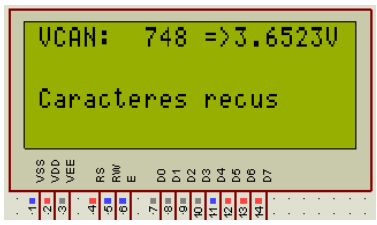
P.G VOX TEST_RS232_ATMEGA.DSN

[1] L'environnement de développement intégré AVRco d'E-lab et une plateforme très performantes pour les microcontrôleurs de la famille ATMEL. Le logiciel est gratuit pour les ATmega88-48-8. L'EDI est très développée et propose la gestion avancée d'un grand nombre de périphériques. Le langage de programmation utilisé est le Pascal et il permet si besoin la mise en place d'applications multitâches. Une série de simulations mettant en évidence pour les élèves les différents aspects avantages / inconvénients d'un système embarqué avec une gestion de type polling / Interruption / Multitâches est disponible auprès de l'auteur.

Mise en œuvre de l'application.

Comme pour la première partie de ce TP il faut établir une connexion entre Proteus et TestCom. Au lancement de l'application le logiciel scrute les appuis sur les boutons poussoirs pour réaliser les actions indiquées. Il réalise en permanence la conversion analogique numérique de la tension présente sur la broche d'entrée ADC5 et affiche la valeur obtenue sur l'écran LCD.



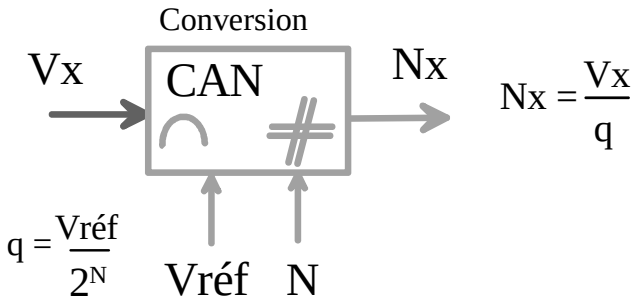


Q8 : Le convertisseur analogique numérique[2] travaille avec 10 bits. Donner les deux valeurs extrêmes possibles en décimal et en hexadécimal.

Q9 : Combien faut-il au minimum d'octets pour transmettre le résultat de la conversion ?

Le convertisseur analogique numérique

Le convertisseur réalise la conversion d'une tension inconnue en un nombre à partir de la donnée d'une tension de référence Vref et de son nombre de bits N. Le quantum est la plus petite valeur qu'il peut discriminer, ce quantum noté q permet ensuite de déterminer Nx.



Pour afficher la valeur de la tension après la conversion dans l'application il suffit de faire

$$Vx_{affichée} = Nx \cdot q \quad \text{en [V]}$$

[2] Les CAN sont traités de manière particulières dans une autre activité autour de l'hémomixer réarrangée à partir d'un travail de l'académie de Paris, voir site sti2d.patgue.com. On reste donc bref ici.

STI2D SIN

Q10 : Pour notre application Vref vaut 5 V donner la valeur du quantum avec 6 chiffres significatifs.

Q11 : Remplir le tableau ci-dessous, pour calculer les valeurs Nx en sortie de convertisseur il faut conserver uniquement la partie entière du résultat de la division Vx / q. Un convertisseur ne traite que des nombres binaires entiers.

Vref	5V
N	10
q	

Vx	v	0,5	1,3	2,5	3,0	3,8	4,2	5,0
Nx théorique	Sans dimension							
Nx simulé	Sans dimension							
Tension affichée	v							

Q12 : Comment expliquer les différences visibles entre les lignes Vx et Tension affichée ?

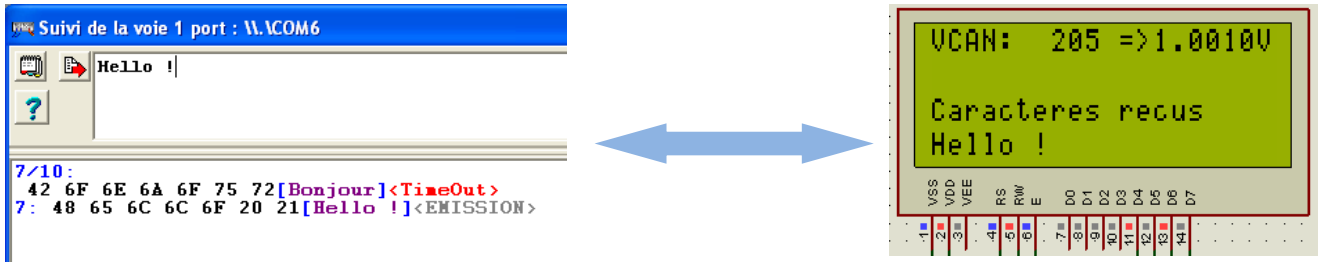
Q13 : Calculez les erreurs relatives en % :

Erreur relative	en %							

L'erreur relative en % est dans ce cas égale à : 100•|Vx - Tension affichée| / Vx

Communication

Faisons communiquer nos deux applications. Tout d'abord échangeons un petit message de bienvenue.



Q14 : Envoyer un message de quelques caractères depuis TestCom vers Proteus, notez les codes Ascii échangés

Le code Ascii permet un échange de data sur 8 bits. Il est donc possible de communiquer des valeurs binaires qui ne correspondent pas à des caractères 'affichables lettres, chiffres'. Nous allons le visualiser en échangeant les données issues de la conversion du CAN.

Q15 : Complétez le tableau ci-dessous :

Vx [V]	**0,5**	**1,3**	**1,5**	**3,0**	**3,8**	**4,2**	**4,999**
Octets transmis	00 66						
Code Ascii affichés	• f						

Q16 : Que représentent les • dans les caractères Ascii affichés ?

STI2D SIN

Table des codes ASCII

LSB \ MSB		0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
0	0000	NUL	DLE	SP	0	@	P	`	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	"	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	'	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
A	1010	LF	SUB	*	:	J	Z	j	z
B	1011	VT	ESC	+	;	K	[	k	}
C	1100	FF	FS	,	<	L	\	l	|
D	1101	CR	GS	-	=	M	]	m	{
E	1110	SO	RS	.	>	N	^	n	~
F	1111	SI	US	/	?	O	_	o	DEL