Si je bricole ? Tu rigoles !
V2.0

Bon !

J'ai pas encore laissé tomber !

En fait je me disais que la méthode que j'avais réussie à faire en utilisant la carte son sur l'idée de Pascal avait l'avantage que l'on ait pas besoin de comprendre le signal (il suffit juste de le retravailler un peu à l'aide d'un logiciel de son).

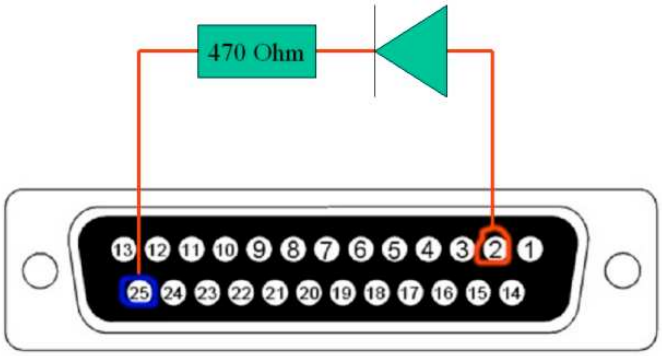
Ceci dit, et les derniers tests avec Pascal tendent à prouver qu'il faut encore avoir une carte son permettant l'enregistrement.

En effet il semblerait que ca ne marche pas avec les ordinateurs portables : ni celui de Pascal ni celui de Martine (ma chère et tendre) ne sont en mesure d'enregistrer le signal correctement. Le signal n'arrive pas à tenir le créneau et s'atténue assez rapidement. En analysant la vitesse à laquelle chute la tension je calculais une fréquence de coupure de 60 Hz environ et ceci m'amenait à penser que sur les ordinateurs portables la compacité et donc la proximité des composants imposaient peut-être aux constructeurs le filtrage du 50 Hz pour ne pas avoir de buzz dans les écouteurs.

Bon ça reste à prouver quand même…

La distance d'action est aussi un problème.
Actuellement en mettant une pile de 9V et une résistance de 100 Ohm je dépasse difficilement les 30 cm.
Sans compter que lorsque le système est branché on ne peut plus écouter de musique...

Je décidais de reprendre la méthode en utilisant le port parallèle qui n'avait pas fonctionné mais qui présentait les avantages de ne pas bloquer la carte son pour de l'audio, de ne nécessiter qu'un montage extrêmement simple (diode IR et résistance) et d'avoir des tensions bien déterminées (5V).



Par contre il faut analyser le signal pour savoir quel signal envoyer. En même temps on peut envoyer le message qu'on veut.
Ce qui peut-être intéressant puisqu'il semblerait qu'il y ait des codes cachés (donc non enregistrables) pour les robosapiens : http://home.planet.nl/~pruim006/tips_and_tricks.htm#3.%20IR%20Codes).

D'autre part le programme envoyant les commandes est perturbé par les interruptions windows et le Jitter est peut-être trop important.

Ce dernier inconvénient est bien sur vital. C'est pour ça que ca ne fonctionnait pas.

Par curiosité je replonge la tête dans le programme en me disant qu'il faut au moins réussir à le faire fonctionner. Ne serait-ce que par principe. Ceci en éliminant tous les process inutiles qui tournent sur mon ordi pour ne pas ralentir la machine et en optimisant le code au maximum.

D'abord je m'apercois que je pouvais justement l'optimiser. En effet avec les deux fonctions APIs permettant de créer des pauses à une précision de quelques dizaines de microseconde (lorsque windows ne perturbe pas la chose) je règlais le temps de chaque incrément élémentaire de temps. Les erreurs de temps s'accumulaient donc au fur et a mesure du signal. Je choisissais donc de prendre mon temps initial comme référence et de mesurer le temps à l'aide de celui ci (en y refléchissant je me dis que la première méthode etait vraiment idiote. 'fin bon...)

Je me permettais aussi de mesurer l'écart entre l'envoi réel et l'envoi idéal en mesurant ainsi le Jitter (de l'ordre de la micro seconde).

En éteignant tous les process inutiles alors je pouvais avoir un signal relativement correct.

Et ceci me permettait de piloter mon magnétoscope !

YES ! ! ! ! ! ! ! ! ! ! ! !

Mais bon... pas tout le temps.

Du coup, au lieu d'envoyer 1 seule commande j'en envoyais 30 à la suite.
Comme ça j'augmentais la probabilité de succès.

Et ca fonctionnait déja mieux, presque parfaitement pourrait-on dire.
Par contre lorsque j'allumais mon programme de webcam par exemple alors ça ne marchait plus (Jitter de l'ordre de quelques dizaines de micro seconde voire plus).

Windows trop occupé à l'acquisition de ma webcam ne donnait pas suffisament souvent la main à mon programme pour que celui ci soit un Jitter suffisament faible.

Bon !
C'etait à prévoir mais j'étais au moins satisfait que ca marche à peu près.

Ceci m'amenait à penser que pour avoir quelque chose de fiable il fallait forcément avoir un signal propre, sans Jitter. Et de là, deux possibilités :
Un montage avec un PIC par exemple où l'ordinateur n'était sollicité que pour envoyer une commande à un processeur. Celui ci possèdant une horloge envoyant les données avec un tempo très régulier.
Ou alors...

Le port série !

Je me dis que celui ci a comme avantage :
- d'être temporisé (envoi à 9600 bauds par exemple),
- d'avoir une tension stable,
- de ne pas bloquer la carte son.

Par contre gros problème :
- Toutes les vitesses de transmission ne sont pas disponibles mais seulement certaines valeurs spécifiques (9600,4800, ...),

- et surtout : les données sont toujours envoyées avec au moins un bit de start et un bit de stop. Donc impossible d'envoyer un signal correspondant exactement à ce que l'on veut. Le signal sera toujours pollué par ces 2 bits.

Bon !
En même temps en réfléchissant je me dis que si on utilise un taux de transmission assez élevé alors :
- On peut réduire le problème de ratio entre la fréquence d'envoi et la fréquence des données à envoyer qui ne serait pas entier.
- Le problème de bit de start et de stop se trouve alors peut-etre résolvable puisque les fréquences qu'ils génèrent seront d'autant plus élevées. Peut-être qu'en les filtrant...

En regardant de plus près le protocole du port je cherche à savoir quelles tailles de messages sont possibles :

1 bit de start + 4,5,6,7 ou 8 bit de données + 0 ou 1 bit de parité + 1 ou 2 bit de stop

Je peux donc aller d'une taille de 6 à une taille de 12.
Chacune de ces tailles en combinaison avec les vitesses disponibles : 110, 300, 9600,... va me permettre de réaliser exactement un certaine nombre de fréquences bien déterminées. Les autres fréquences ne seront qu'approximatives et le signal correspondant possèdera un jitter intrinsèque…

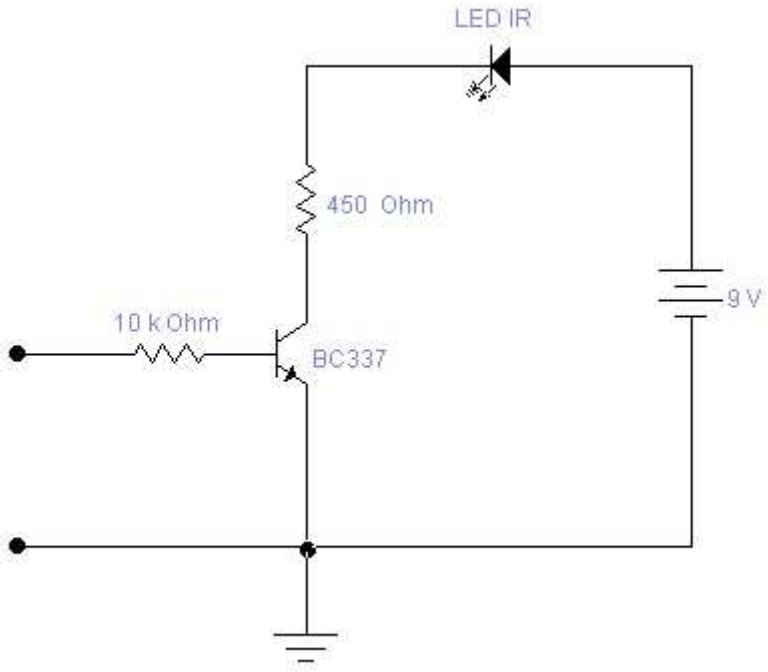
De manière à séparer en fréquence le plus possible mon signal du signal généré par mes bits perturbateurs je pars alors sur l'idée que je vais utiliser la vitesse de transmission maximum. Je me dis aussi que je vais utiliser un rapport bit données / bit pollueur le plus élevé possible (ce n'est jamais qu'un rapport signal sur bruit).

Mon choix s'oriente donc très naturellement sur les paramètres suivants : vitesse de transmission de 115 200 bits/seconde, pas de bit de parité, 1 bit de stop.

Je pourrais donc générer des fréquences allant de 0Hz à 115 200/2 = 57 600 Hz (signal 101010101...). Le signal sera reproduit exactement, à ceci près que j'aurai tout les 10 bits deux bits perturbateurs (le bit de stop puis le bit de start) ayant une chance sur deux de se trouver dans un état différent de celui que j'escompte.

Dans la mesure où la dernière fois je n'arrivais même pas à avoir un signal correct avec le montage que j'avais trouvé sur internet : une diode, une résistance et la LED IR alors je décide d'utiliser tout simplement mon montage " commutateur " avec le

transistor BC337. Ceci me permettra en plus de règler plus facilement l'intensité sur la diode sans changer l'intensité dans le port série.



En réalité j'ai changé l'alimentation par transfo 5V en alimentation avec une pile de 9V puisque je n'avais pas envie de connecter la masse de mon transfo avec la masse du port com (ceci dit ça doit être galvaniquement isolé). Du coup, j'ai mis une résistance de 450 Ohm pour limiter à 20 mA le courant maximum qui pourrait passer dans la LED IR. Ceci dit lorsque l'on regarde les spécifications de ces composants on s'aperçoit qu'elles peuvent fonctionner vers 100mA par contre par forcément en courant continu mais seulement par impulsion (comme ici). Certaines vont même jusqu'à 1A en pulsé.

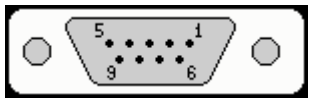
Dans le doute je laisse ça comme ça. Si l'intensité n'est pas assez élevée je pourrais éventuellement prendre une résistance plus faible.

Concernant le port COM je trouve sur internet les spécifications des pins : (http://www.aurel32.net/elec/port_serie.php)

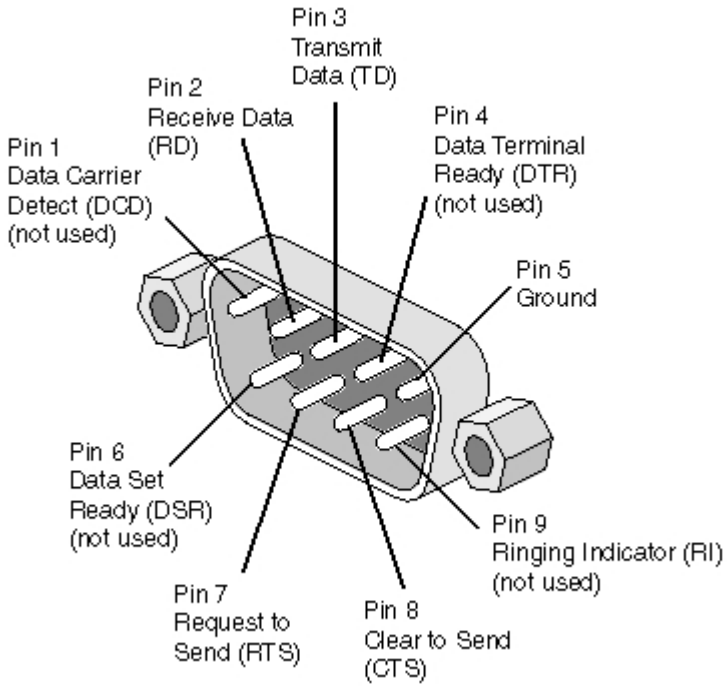
Broche DB9	Broche DB25	Nom	DTE	DCE	Description
x	1	PG	x	x	Masse de protection (PG = Protecting Ground) **Ne pas utiliser comme masse du signal !**
3	2	TD	S	E	Transmission de données (TD = Transmit Data)
2	3	RD	E	S	Réception de données (RD = Receive Data)
7	4	RTS	S	E	Demande d'autorisation à émettre (RTS = Request To Send)
8	5	CTS	E	S	Autorisation d'émettre (CTS = Clear To Send)
6	6	DSR	E	S	Prêt à recevoir (DSR = Data Set Ready)
5	7	SG	E	S	Masse du signal (SG = Signal Ground)
1	8	DCD	E	S	Détection de porteuse (DCD = Data Carrier Detect)
4	20	DTR	S	E	Équipement prêt (DTR = Data Terminal Ready)
9	22	RI	E	S	Détection de sonnerie (RI = Ring Indicator)

Ce qui m'intéresse c'est donc à priori le pin 3 pour le signal et le pin 5 pour la masse.

Sur ce même site le schéma qui correspond au coté cable (femelle)



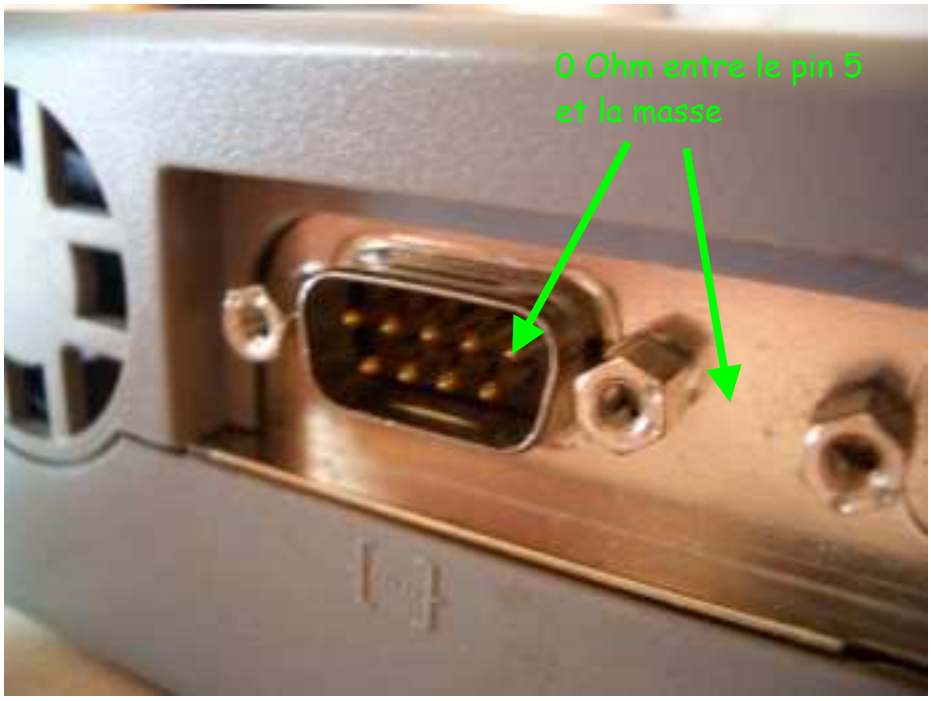
Sur un autre site le coté PC :



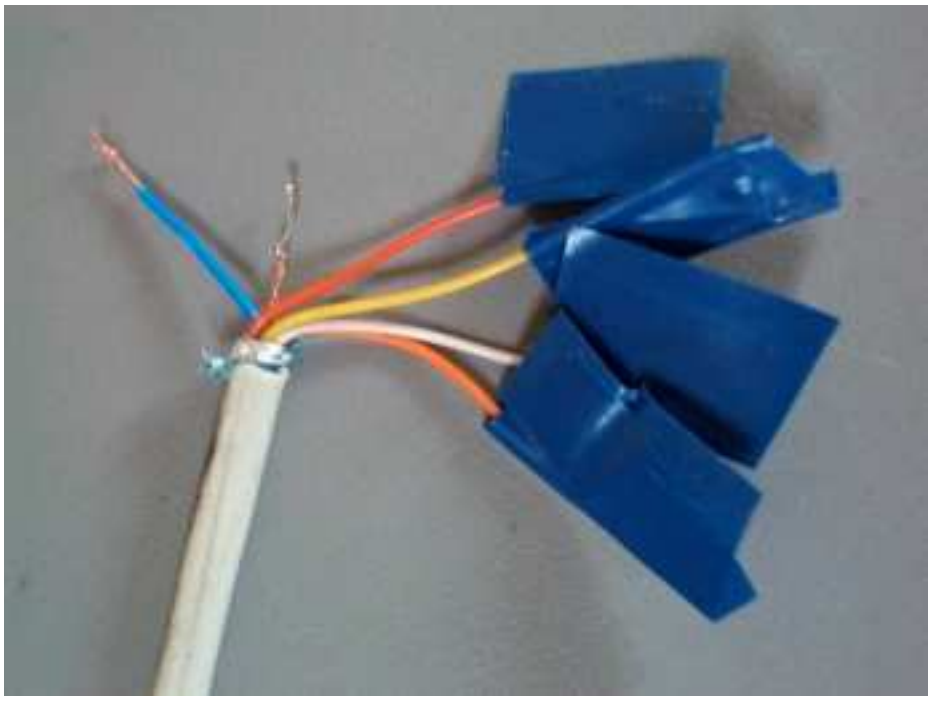
A l'envers bien sûr donc il faut faire attention !

En récupérant le cable série d'une vieille souris je vérifie car les chiffres y sont inscrits.

Et sur le PC avec un ohmmètre :



J'identifie à l'ohmmètre les couleurs des fils qui m'intéressent et j'isole ceux qui ne m'intéressent pas :

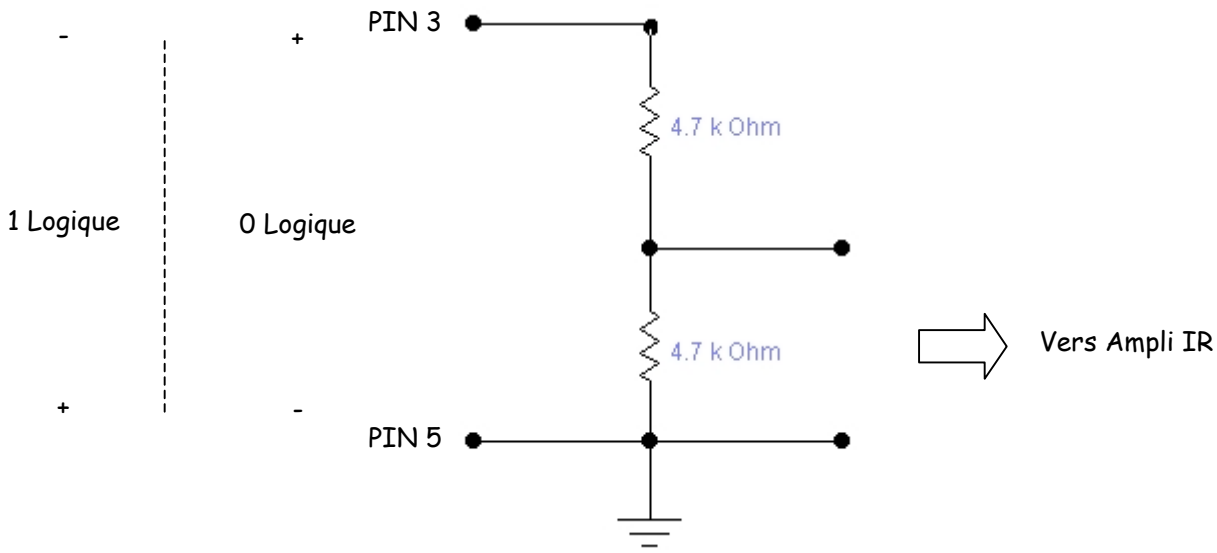


Pin 5 / la masse / fil dénudé
Pin 3 / Signal (TD) / fil bleu

Le lien de tout à l'heure (http://www.aurel32.net/elec/port_serie.php) m'indique que :
" *D'un point de vue électrique, les différents signaux présents sur le port série sont définis par la norme RS232. Afin d'avoir une plus grande immunité au parasites, on utilise des tensions comprises entre -25 et +25V. Une tension comprise entre -3V et -25V représente un 1 logique (mark), alors qu'une tension comprise entre +3V et +25V représente un 0 logique (space).* "

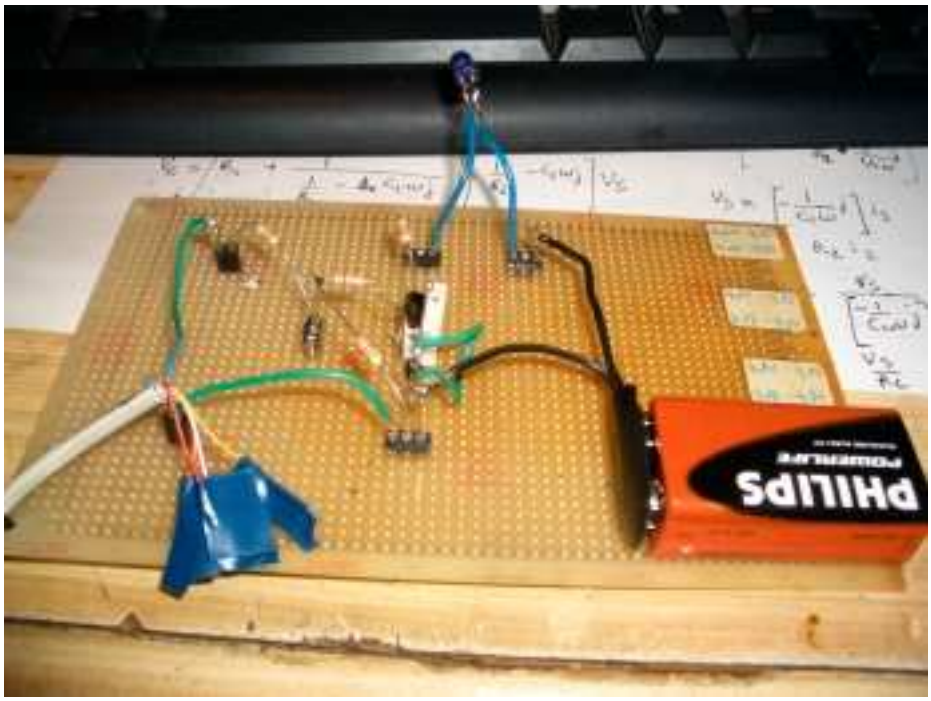
La tension mesurée entre les deux pins étant sur cet ordinateur (celui de Martine) de 9V environ. J'en conclus que le 1 logique est -9V et le 0 logique +9V. (Un peu plus tard je monterai le port COM sur mon ordinateur puisqu'il n'étais pas monté et mesurerai 10,5V environ).

Dans la mesure où je n'ai pas besoin d'un signal très élevé, je décide de faire un simple diviseur de tension avec 2 résistances. Le voltage mesuré sur le port série étant de l'ordre +/-9V, en prenant deux résistances identiques j'obtiendrai alors un signal de +/- 4.5V. Pour limiter le courant je choisis R1 + R2 = 10 kOhm comme ça j'aurai un courant de l'ordre de 1mA. (en fait je prends R1 = R2 = 4.7 kOhm car les 5 kOhm n 'existent pas... Au moins chez moi :)).



Dans la mesure où l'impédence d'entrée de mon ampli IR est assez élevé (au moins 10 kOhm) la division en tension doit être à peu près comme prévu.

Je réalise alors le montage :

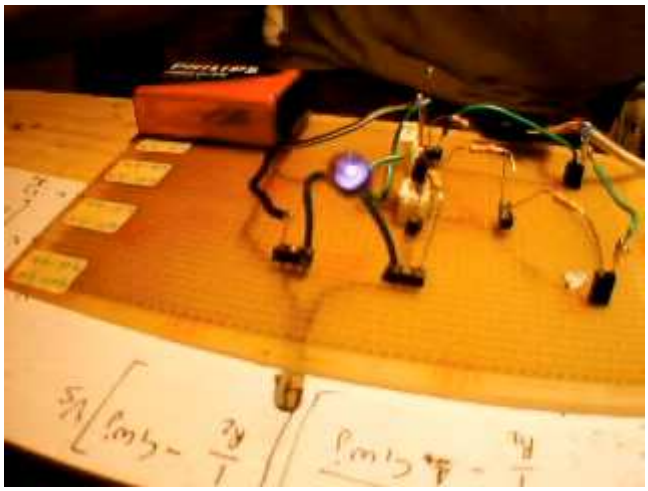


Et je m'attaque alors au programme en VB6 en faisant l'algorithme suivant :
- Définition de la fréquence du signal à reproduire,
- Définition d'une chaine de caractère contenant le message à envoyer sous la forme " 1010001101 ... "

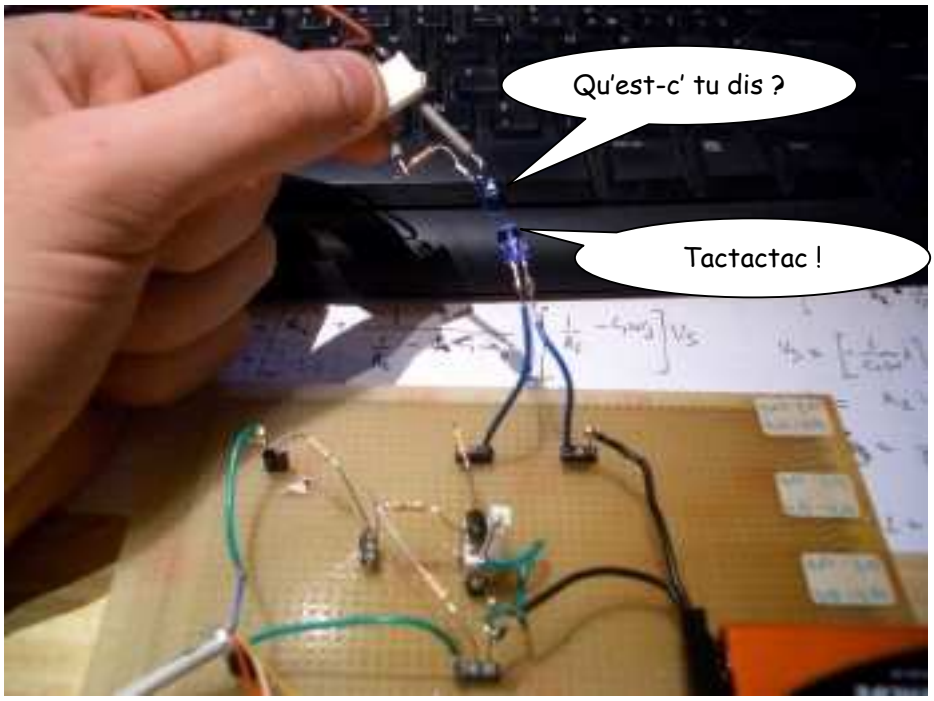
- Calcul du nombre de bits pour chaque créneau unitaire en fonction de cette fréquence,
- Conversion de la chaine à envoyer en une série de bits,
- Conversion de cette série en suite de caractère ASCII (8 bits) : tous les 10 bits je prends les bits 2 à 9 pour en faire mon paquet et laisse tomber le 1er et le 10ème bit ceux-ci étant remplacé « aléatoirement » à l'envoie du paquet par les bits de start et de stop,
- Envoi sur le port série de la chaine ainsi produite.

Un bouton me permet d'envoyer mon signal.

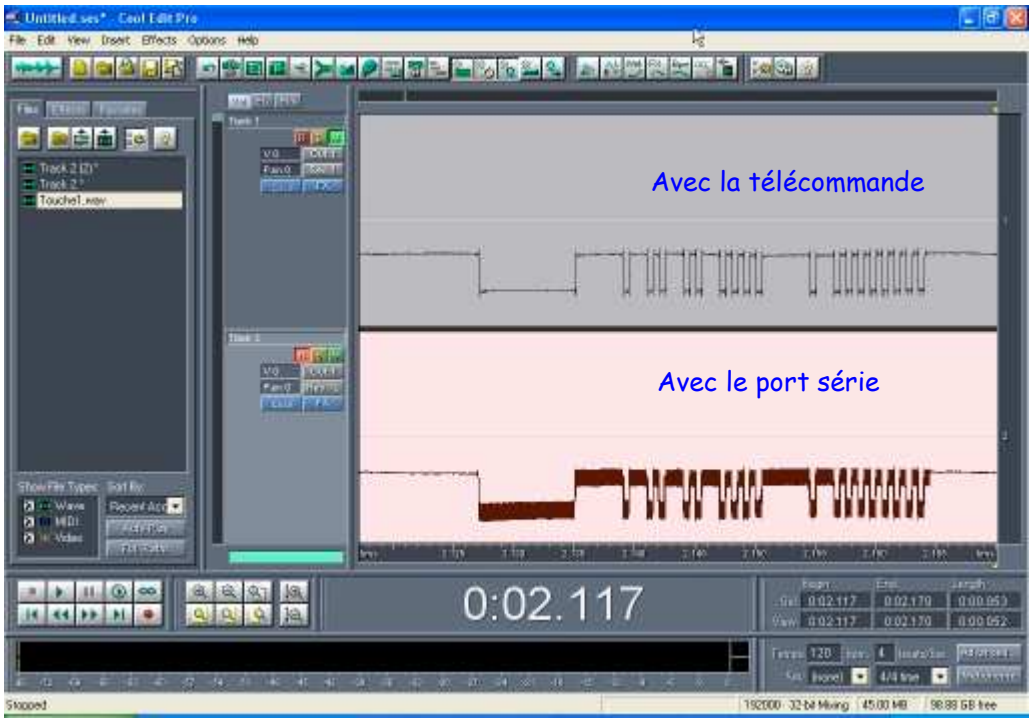
Après un rapide test visuel avec la caméra :



Ainsi qu'un test auditif avec le désormais célèbre Micro IR :

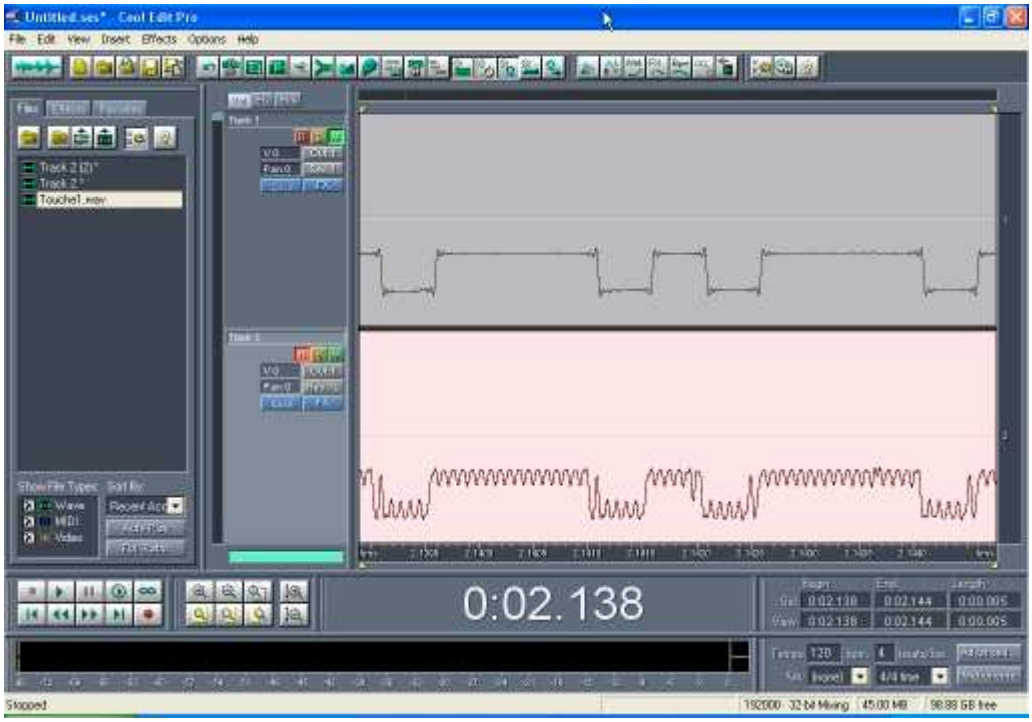


J'enregistre le signal pour le comparer à l'original (enregistré avec le même micro IR mais avec la télécommande) :



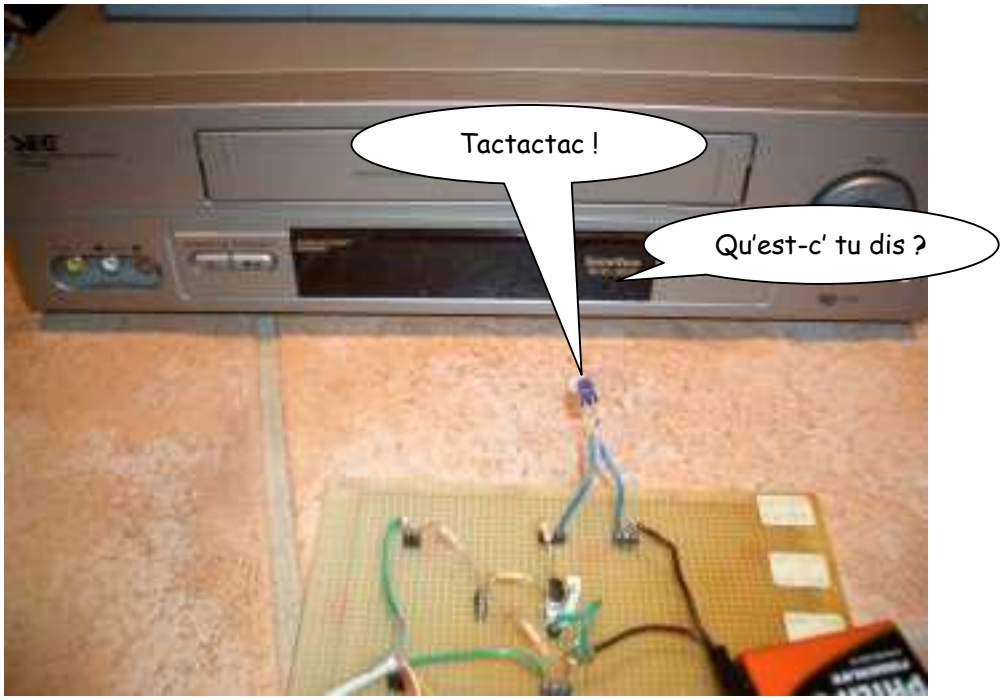
Pas mal niveau timing mais un peu épais !

Zoomons...



Bon ! les voilà nos fréquences parasites dues aux bit de start et stop.

J'essaie quand même avec le magnétoscope...



Bon... Marche pas mais c'était à prévoir .

Va donc falloir essayer de filtrer !

Vu que ma suite de 1 et de 0 doit être envoyé à un rythme de 2 000Hz (pour le robosapien c'est 1 200 Hz) alors la fréquence maximum sera 1 000Hz (un 1 suivi de un 0). Par contre mon signal doit être carré. Donc j'ai besoin aussi des harmoniques impaires.

Voir cours de traitement du signal par exemple : http://perso.orange.fr/acsa/mathematica/fourier1/index.html).

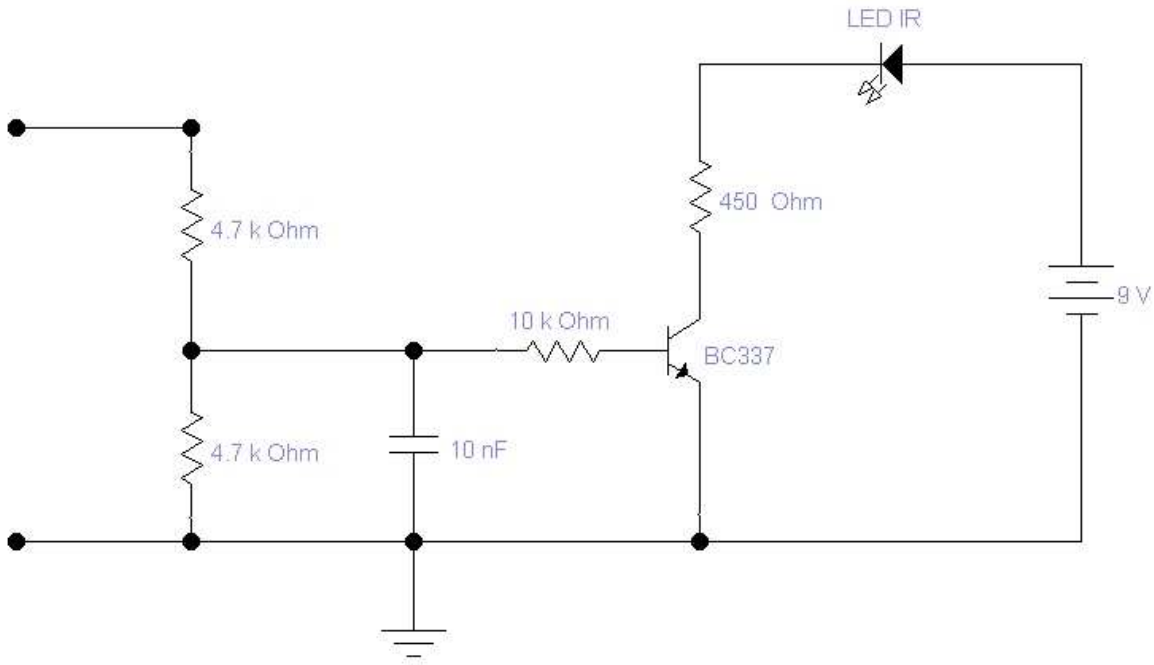
C'est à dire qu'il me faudra au moins la 3 000Hz mais aussi la 5 000Hz et eventuellement la 7 000Hz, etc…

Les fréquences générées par mes perturbations de bit stop et start sont plus grandes que la fondamentale de mon signal. Le bit stop suivi du bit start sont à 115 200 / 2 = 57 600 Hz, celle ci ne me gènera donc pas du tout (encore moins ses harmoniques). Par contre la répétition de ce schéma " stop "-" start " dans le temps (tous les 10 bits) est à 5 760 Hz. Et si je n'ai pas besoin de ses harmoniques, je n'ai vraiment pas non plus besoin de cette fondamentale.

Donc ca va être chaud. Si je filtre trop bas alors j'élimine les perturbations mais j'élimine aussi les harmoniques de mon signal et mon signal sera trop « arrondi ». Si je filtre trop haut j'élimine peu mes harmoniques mais je n'élimine pas non plus mon signal perturbateur.

Il va donc falloir trouver un compromis (si ce compromis existe …)

Je me dis que je vais filtrer à l'entrée plutôt qu'en sortie et ceci se fera à priori "facilement" en mettant une capacité en entrée de l'ampli IR. A l'aide des résistances le dispositif va faire filtre passe-bas. En effet pour les fréquences plus élevées l'impédance de la capacité va diminué ($Zc = -1/C\omega$) et la tension à ces bornes (donc à l'entrée de l'ampli) aussi.



Je fais le calcul du filtre en ne considérant que les deux résistances de 4.7kΩ (l'impédance du reste étant plus élevée et au moins en première approximation...).

Je vous passe les calculs qu'il faudrait que je mette en forme mais je vous les explique volontiers sur demande.

J'aboutis à la relation que la fréquence de coupure est donnée par :

$$Fc = 3^{0.5}/\pi/R/C$$

Avec R = 4 700 Ohm et C = 20nF on obtient Fc = 5 865 Hz. Donc ca sera de l'ordre du nF à la dizaine de nF.

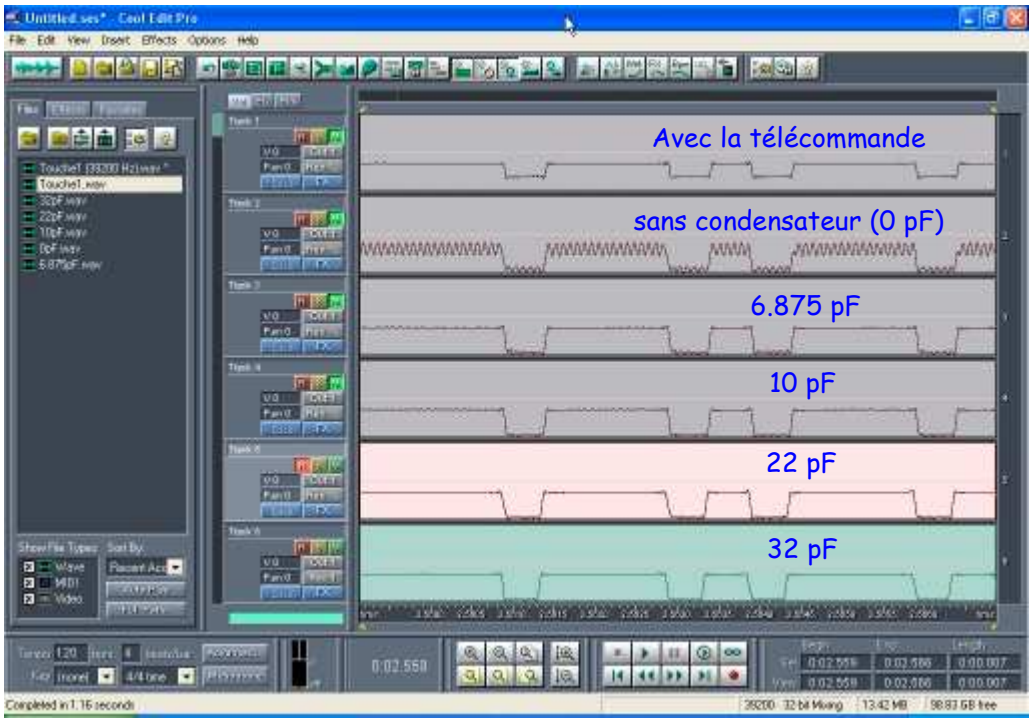
Des capacités j'en ai sur ma chaine qui fournissait le BC337... ☺
En cherchant un peu j'en trouve une de 10 et une de 22 nF.
Celle de 10 nF me donnera Fc = 11 730 Hz
Celle de 22 nF => Fc = 5 332 Hz
En parallèle elles s'additionnent et feront 32 nF => Fc = 3 665 Hz.
En série l'inverse de la somme des inverses me donne 6.875 nF => Fc = 17 062 Hz

Je vais donc jouer avec et on verra !

(d'ailleurs en passant j'ai été obligé de chercher comment sont marqués les condensateurs vu que je n'ai pas de capacimètres. J'ai trouvé ça qui est très bien :

http://www.sonelec-musique.com/electronique_theorie_condensateur_codes_valeur.html

Je fais rapidement un petit montage pour connecter-déconnecter mes condensateurs et j'enregistre les signaux…



Déjà ça filtre et je me congratule moi qui en électronique me trouvais nul !

Il ne me reste plus qu'à tester ! !

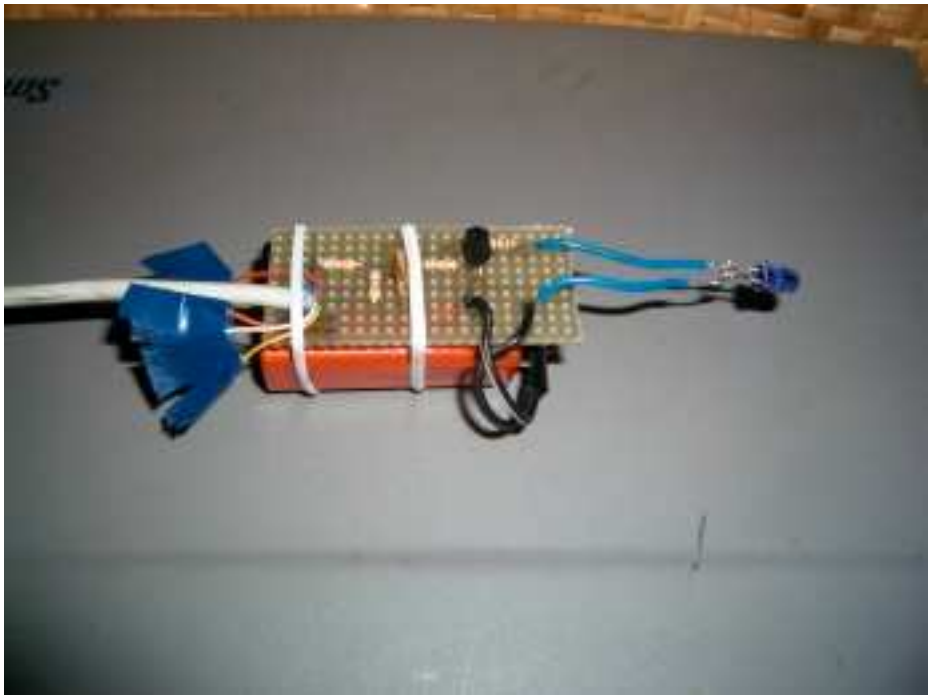
Le 10 pF ferait un bon compromis…

Test magnétoscope !

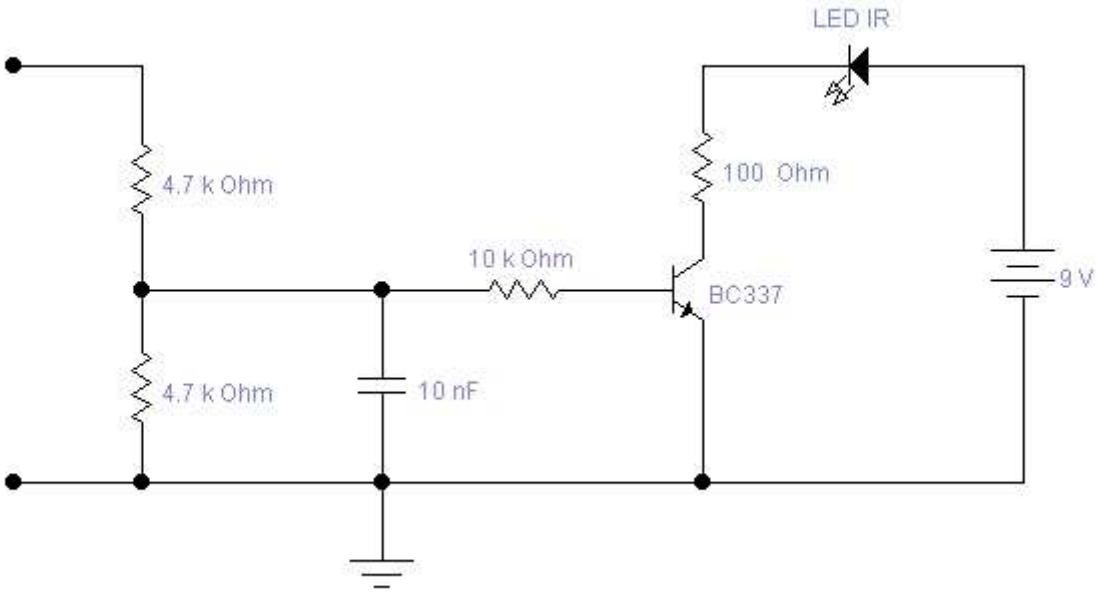
……

YES ! ! ! ! ! ! ! ! ! ! ! !

Je finalise alors le montage :



Et vous refournis le montage final :



J'ai finalement opté pour une résistance de 100 Ohms.
Au maximum le courant sera de 90 mA.

La distance à laquelle je peux commander mon magnétoscope est supérieure au mètre. Et selon les jours (? !) peut aller jusqu'à 2-3 mètres. Par contre la directivité

est variable et je vous avouerai que par rapport à une télécommande normale c'est quand même moins pratique. Manque de puissance ? Signal trop pourri ?…

Pour conclure je dirai que même si je trouve ça positif je pense qu'il serait préférable d'utiliser un PIC pour avoir un signal ultra propre. Sans compter que le robot, par rapport à mon magnétoscope, ne reste pas immobile !

Si vous avez des remarques…

A+
Nico