Encodeur CTCSS Arduino par ON7NU, Daniel

Pour ma part, opérer des anciens tranceivers est un réel plaisir et je ne suis probablement pas le seul. Le problème, c'est que ces tranceivers ne génèrent pas de signal CTCSS, ce qui empêche de trafiquer sur la plupart des relais.

Objectif:
Simplicité, peu de composant périphériques, obtenir un choix de fréquences CTCSS, et faibles couts. Actuellement, il est possible de se procurer un Arduino Uno ou un Nano, pour quelques € seulement.

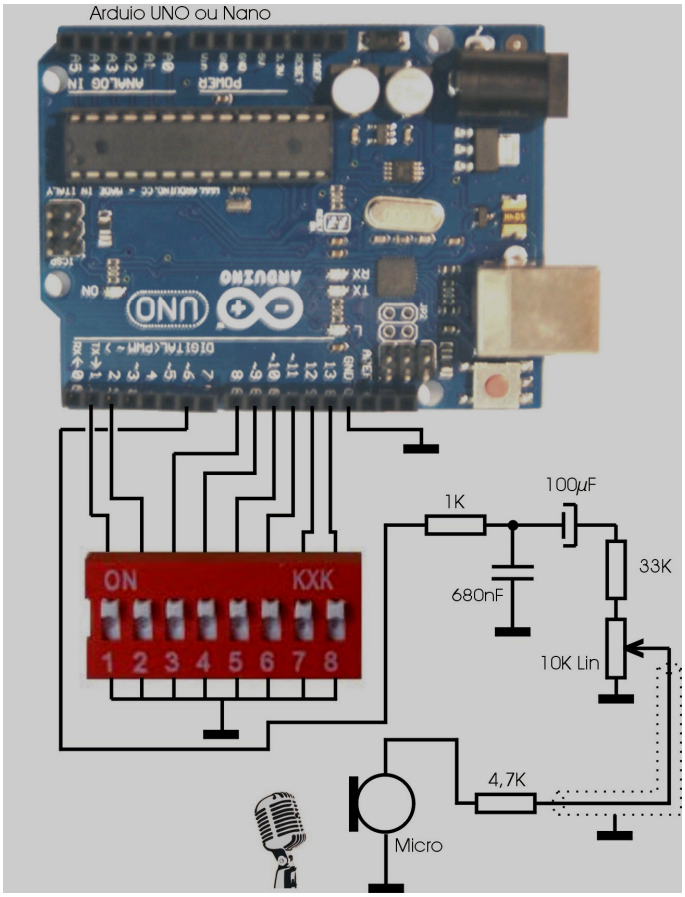
Pour générer du CTCSS, Il existe à ma connaissance trois solutions.

- Générer un signal carré sur la fréquence CTCSS désirée et ensuite le filtrer à l'aide d'un filtre passe bas pour obtenir une sinusoïde (enlever l'Harmonique 2,3 etc...).
  L'inconvénient réside dans le fait qu'idéalement la fréquence de coupure du filtre doit être ajustée pour chaque fréquence CTCSS.

- Utiliser un DDS (Direct Digital Synthétiseur). Ceci implique de mettre en œuvre soit un circuit DDS spécialisé, soit un CPU + convertisseur digital / analogique (D.A.).
  Ceci reste plus couteux et plus compliqué.

- La technique du Pulse With Modulation (PWM).
  Le principe de cette technique est de faire varier le rapport cyclique ce qui revient à modifier la largeur des impulsions. La périodicité est constante et a pour valeur 62.5KHz = 16µs.
  Dans la pratique, une impulsion de courte durée provoque une tension efficace faible. En revanche, une impulsion plus longue engendre une tension efficace plus élevée.
  Les impulsions à elles seules ne peuvent générer une sinusoïde puisque l'amplitude est constante.
  Par contre, si ces impulsions passent via un circuit intégrateur de type RC série, une moyenne va en résulter et ce sera une sinusoïde à condition qu'un algorithme correct soit appliqué.
  C'est cette solution qui est retenue.

Moyens mis en œuvre:



Un Arduino Uno ou nano, un intégrateur R.C., un dip switch, un atténuateur, un condensateur de liaison.

Description du montage:
La sortie N°6 (PWM) de l'Arduino est la seule sortie utilisée. Les impulsions qui en sortent sont la résultante d'un tableau qui est gelé dans la mémoire de l'Arduino.
Ce tableau de data comporte les 256 valeurs hexadécimal d'une sinusoïde.
Voici les 4 premières valeurs du tableau:
0x80,0x83, 0x86, 0x89,

L'introduction de la fréquence CTCSS se fait à l'aide du DIP Switch à 8 canaux.
Un seul switch à la fois doit être fermé sur ON, ce qui permet de sélectionner 8 fréquences au choix.
Au démarrage, l'Arduino, va lire l'état des 8 switchs via les entrées 2,3,8,9,10,11,12,13.
Après lecture de l'état des switchs, la sortie PWM est active, mais à partir de cet instant, il n'est plus possible de modifier la fréquence.
Un reboot est nécessaire soit par un appui sur la touche reset ou bien couper l'alimentation.

La sortie N°6 sortie attaque le circuit intégrateur représenté par la résistance de 1K et le condensateur de 680nF.

La sinusoïde est donc présente aux bornes de ce condensateur, mais les impulsions ont un niveau logique de +5V. La tension sinusoïdale pointe à pointe est elle également proche de 5V et avec une

tension continue résiduelle de U/2, soit +2.5V. Le condensateur de liaison de 100µF permet d'éliminer cette composante DC.

A la sortie du 100µF nous avons enfin une tension alternative sinusoïdale pure, mais son amplitude reste à +5V pointe à pointe (PtP).
Pour attaquer le tranceiver, Il faut donc ramener cette valeur autour de quelques mV. C'est le rôle de la résistance de 33K et du Trim Pot de 10K.
Avant d'attaquer l'entrée micro, il faut placer une résistance de 4,7K. Cette résistance permet de ne pas atténuer le signal en provenance du micro dans le cas ou le curseur du Trim Pot serait proche de la masse.

Procédure de réglage :
Une attaque via l'entrée micro, n'et pas la méthode idéale. Le mieux serait d'attaquer directement le modulateur. Ceci permet d'injecter derrière les filtres et le préampli audio. Malheureusement, ceci nécessite le démontage du transceiver et impose aussi de connaitre l'endroit exact pour l'injection.
J'ai personnellement testé avec succès l'injection micro du CTCSS sur trois transceivers.

Donc:
Choisir sa fréquence de CTCSS à l'aide du DIP Switch puis rebooter.
J'ai réalisé un tableau qui comporte 6 fréquences CTCSS, il reste donc deux fréquences qui peuvent être customisées.

Placer le curseur du Trim Pot au minimum.
Brancher la sortie de l'encodeur sur l'entrée micro.
Envoyer une porteuse pour activer le relais. Mais celui-ci ne devrait pas s'ouvrir.
Augmenter progressivement le niveau d'attaque CTCSS par essais successifs.
Quant le relais s'ouvre, faire un essai avec un niveau légèrement inférieur pour éviter la saturation de l'étage d'entrée micro, mais aussi pour éviter le cliping au niveau du limiteur du modulateur FM.

Il est évident qu'il faut annoncer son indicatif durant ce test hi...
Je me tiens à la disposition des OM's qui désireraient générer d'autres fréquences que celles proposées

Le programme (Sketch) est disponible sur mon site : http://users.numericable.be/on7nu

Dans le code, voici la partie qui associe la fréquence au switch:

```
int Lire2 = digitalRead (2); // Lire switch2
  if (Lire2 == LOW)
   {
     long Frequence = 19140;  // 19140 est la valeur qui génère du 74.4Hz
     return Frequence;
   }
```

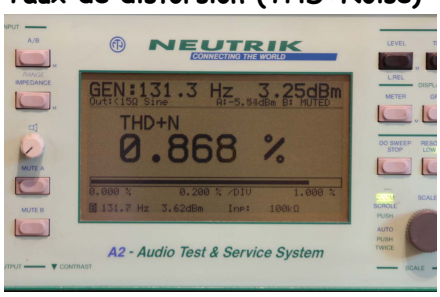
Switch	Status	Freq. (Hz)
1	ON	67
2	ON	74,4
3	ON	79,7
4	ON	88,5
5	ON	107,2
6	ON	131,8
7	ON	131,8
8	ON	131,8

Pour des raisons de division, j'ai constaté qu'une règle de 3 permet de s'approcher de la fréquence désirée, mais il y a parfois un certain % d'erreur. Je dispose d'un fréquencemètre pour déterminer cette valeur avec précision.

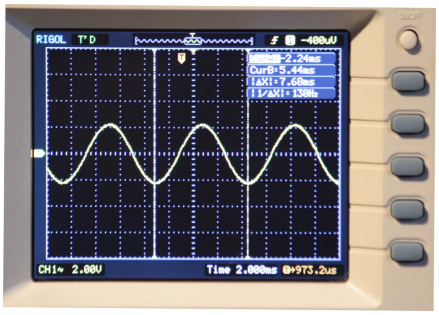
Mesures

Précision de la fréquence

Taux de distorsion (THD+Noise)



Oscilloscope



ON7NU – Daniel