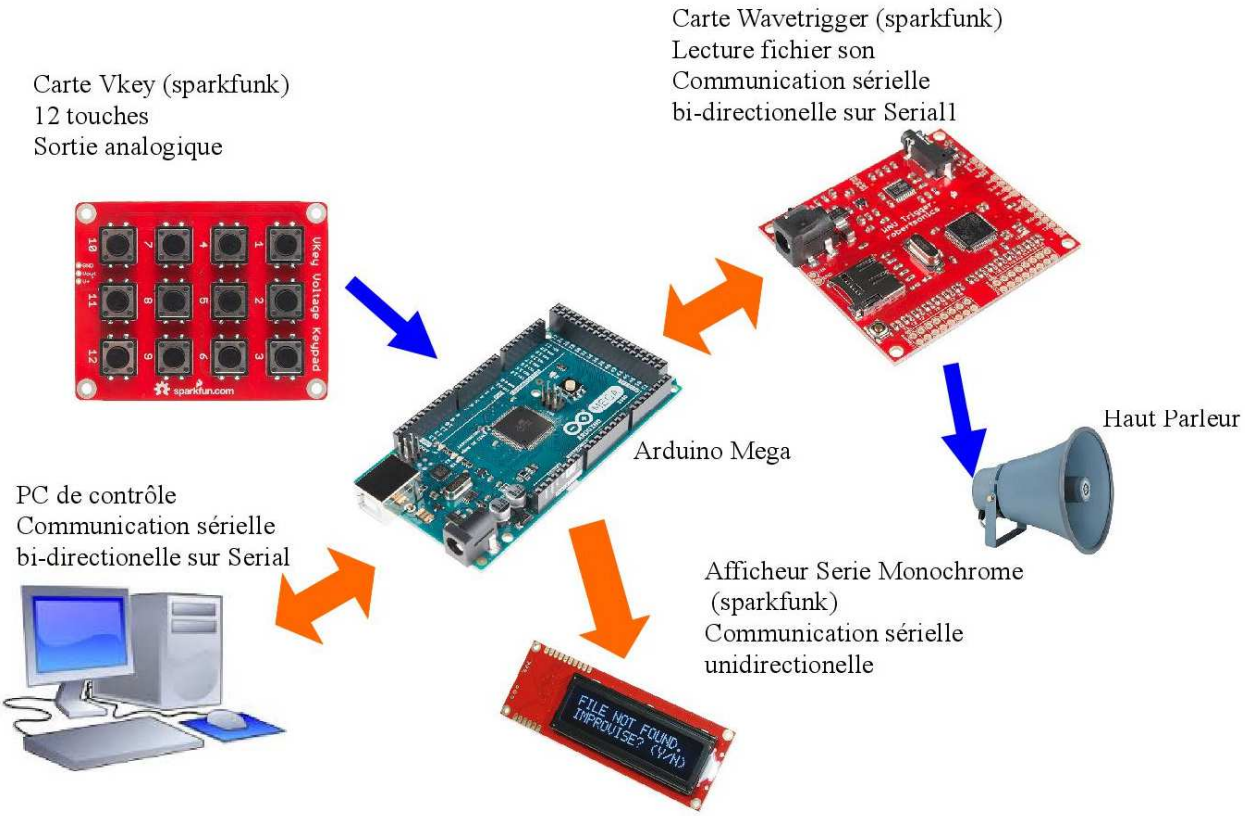
Synthèse vocale des nombres.

O : Préambule, système envisagé

Depuis plusieurs années, le problème de la synthèse vocale des nombres me "titille". Ayant découvert une petite merveille dans le catalogue sparkfunk, et, fort de l'expérience des années précédentes sur le matériel Arduino, je me suis lancé. Non sans mal ! (problèmes informatiques pur d'adaptation et correction des bibliothèques, problème de grammaire Française sur la diction des nombres, problèmes de formats de fichiers son ...) enfin rien que du "normal" dans un monde de technologie en effervescence ou les problèmes techniques sont toujours présents et nécessitent des hommes de plus en plus affutés pour les résoudre.

Bon je sais que pédagogiquement, il n'est pas bon de commencer par le résultat mais cela nous évitera bien des errements et du texte inutile.

Le système envisagé et réalisé :



Les bibliothèques utilisées :

Vkey, Wavtrigger, SerialDisplay que vous pouvez télécharger librement sur le site de l'auteur de cet article : www.altituduino.xyz

Le matériel et les adresses pour se le procurer :

Vous trouverez tout cela avec les bibliothèques sur ce même site.

Je répond tout de suite à la question "pourquoi un Mega" alors qu'à la compilation on s'aperçoit que le programme est ridiculement court ?

Lors de la mise au point, ne connaissant pas comment réagissait la carte wavtrigger, j'avais besoin de deux sorties série "matériel" asynchrones pour être sûr de ne pas manquer un octet de dialogue. Sur un Uno, il n'en existe qu'une (Serial) , qui lors du débogage est pratique (pour ne pas dire indispensable) pour communiquer avec le PC, charger le programme ... (enfin bref vous avez compris, elle n'est pas vraiment disponible !). L'utilisation d'une série "logicielle" via l'utilisation de la bibliothèque softwareSerial aurait peut être été possible. Suivant la réponse de la carte, comme on ne peut envoyer et recevoir de manière asynchrone, par ce type de sérialisateur, la mise au point aurait pu se révéler plus longue.

Sur un système embarqué, exit le PC et, le sérialisateur Serial devient disponible. En modifiant simplement le #define correspondant au sérialisateur de la carte son le programme "tourne" sur un Uno (ne pas oublier de dévalider l'option débogage sur le PC) .

Le programme :

le fichier lecturenombre.ino constitue le cœur du système à compiler avec vore IDE préféré "Arduino"

Bon maintenant que cela fonctionne occupons nous de la genèse.

I : Prononcer et écrire les nombres dans la langue de « Molière »

Je remercie l'auteur d'*openclassroom* (c.f. bibliographie) de qui je tire la majeure partie de ce paragraphe.

L'écriture des nombres en chiffres est aujourd'hui à peu près universelle (hors mis les bizarreries « latines » entre le point et la virgule). L'unification sur la planète de la représentation en base 10 date tout de même de quelques siècles.

Nous nous ferons ci après fi de la représentation du temps, que la révolution Française n'a pas réussi à mettre au pas (même partiellement), et, qui représente une belle cacophonie dans le domaine. Sa représentation est d'une incohérence logique et mathématique dépassant l'entendement :

- en base 60 pour les minutes et secondes
- en base 10 pour les divisions de la seconde
- en base 24 ou 12 pour les heures
- en base 30 ou 31 pour les mois (sans parler de février et de sa variabilité bissextile)
- en base 7 pour les semaines
- en base 4 pour les lunes
- on nage dans un délire « pré-arabe » en ce qui concerne les siècles (le siècle zéro n'existe pas et on passe sans transition premier siècle avant JC au premier siècle après JC !!) ... .

Hors mis cette exception non négligeable, où que vous soyez sur la planète, 12 désigne toujours le même nombre (la même quantité). Par contre, il ne se prononce pas de la même façon et il possède une écriture en toutes lettres spécifique.
Vous pensez savoir depuis votre classe de CE2 comment écrire en toutes lettres et prononcer cette suite de symbole significatifs (les chiffres) représentant une quantité que l'on nomme nombre. Apres la lecture de ces quelques lignes, je suis sûr que votre belle certitude va en « prendre un sacré coup »...

Comme notre discours est un préambule à une synthèse vocale et que la règle orthographique des (s) en fin de cent mille ... ne change rien (du point de vue auditif bien entendu), nous omettrons ce point

Jusqu'au million **(10^6=1^{E}6=1000000(en francais)=1,000,000 (chez les saxons))**

Le plus dur, c'est d'apprendre à compter jusqu'à 100 ! En effet, c'est entre 0 et 100 que se concentrent la majorité des exceptions et variantes. Une fois la barre du 100 franchie, tout coule beaucoup mieux.

De 0 à 99

Mais plutôt que de longs discours, venons en au fait et commençons par les nombres de 0 à 9 :

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
zéro	un	deux	trois	quatre	cinq	six	sept	huit	neuf

Les nombres de 10 à 19 ont eux aussi des noms spécifiques :

10	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
dix	onze	douze	treize	quatorze	quinze	seize	dix-sept	dix-huit	dix-neuf

A partir de 20, on commence à voir une règle qui se dégage. Chaque dizaine porte un nom :

20	**30**	**40**	**50**	**60**	**70**	**80**	**90**
vingt	trente	quarante	cinquante	soixante	septante	quatre-vingts	nonante

Pour former un nombre entre 20 et 99 c'est très simple : il suffit de coller l'une derrière l'autre l'écriture des dizaines et celle des unités. Par exemple 43 s'écrit « quarante-trois », 65 s'écrit « soixante-cinq »...

Si le chiffre des unités est 1, on rajoute « et » entre les dizaines et les unités. Autrement dit, 21, 31, 41, 51 et 61, 71 et 91 s'écrivent « vingt **et** un », « trente **et** un », « quarante **et** un », « cinquante **et** un », « soixante **et** un », « septante **et** un », et « nonante **et** un ». Attention il y a une exception 81 s'écrit simplement « quatre-vingt-un ».

C'est quoi ces « (x)ante » ? Je n'ai jamais prononcé les nombres comme ça moi !

Si vous vous posez cette question, je parie que vous n'êtes ni belge, ni suisse, ni luxembourgeois (ni Quebecois pour certaines contrés reculées ni originaire du Rwanda ou du Zaïre). En effet, les dizaines « septante » et « nonante » sont surtout utilisées dans ces trois pays. Si je vous ai expliqué cette variante en premier, c'est qu'elle est plus simple. Car voyez-vous, en français de France on préfère se compliquer les choses ! Du moins après louis XIV. La petite histoire raconte en effet que c'est le roi soleil qui, ayant 69 ans et trouvant que « septante ans » ça sonnait *vieux*, décida de ne pas changer de décennie et de continuer après « soixante-neuf » par « soixante-dix », « soixante et onze » et ainsi de suite.

Le nombre 80 peut également s'écrire « huitante » ou « octante ». « Huitante » est encore utilisé dans quelques cantons suisses tandis-que « octante » n'est quasiment plus employé nulle part.

Les nombres entre 90 et 99 sont régis par une bizarrerie encore plus grande : leur prononciation est issue du système vicésimal en vogue au moyen âge (vingt chiffres : zéro, un, ...onze .. dix-neuf) qui mélange la base 10 et la base 20 : nombre de paquets de 20 puis le reste en codant avec les chiffres en base 20 jusqu'a 16 ou un paquet de 10 plus le complément à 20 jusqu'a 19.

ex : quatre-vingt-dix-sept = 4*20+10+7,

quatre-vingt-onze=4*20+11

En français, ailleurs dans le monde que au pays des exceptions déjà cités, voici comment on écrit les nombres de 70 à 79 :

70	**71**	**72**	**73**	**74**	**75**	**76**	**77**	**78**	**79**
soixante-dix	soixante et onze	soixante-douze	soixante-treize	soixante-quatorze	soixante-quinze	soixante-seize	soixante-dix-sept	soixante-dix-huit	soixante-dix-neuf

et de 90 à 99 :

90	**91**	**92**	**93**	**94**	**95**	**96**	**97**	**98**	**99**
quatre-vingt-dix	quatre-vingt-onze	quatre-vingt-douze	quatre-vingt-treize	quatre-vingt-quatorze	quatre-vingt-quinze	quatre-vingt-seize	quatre-vingt-dix-sept	quatre-vingt-dix-huit	quatre-vingt-dix-neuf

Bon, rassurez-vous, à partir de 100, tout ça devient plus régulier. Par bonheur aucun roi de France n'a vécu plus de 100 ans !

De 100 à 1 000 000

De 100 à 999

Le nombre 100 s'écrit « cent ».

À partir de là, pour fabriquer un nombre entre 100 et 999, c'est simple : on écrit à la suite le chiffre des centaines, on le fait suivre par « cent », puis par le nombre qui correspond aux dizaines et aux unités. Rien de tel que quelques exemples :

- 475 s'écrit « quatre cent soixante-quinze »
- 934 s'écrit « neuf cent trente-quatre »

- 204 s'écrit deux cent quatre...

Notez tout de même deux cas particuliers :

- si le nombre des centaines est 1, on saute le « un » et on commence directement par « cent ». Par exemple, 134 s'écrit « cent trente-quatre » et non pas « un cent trente-quatre » ;
- si les dizaines et les unités valent 0, on ne l'écrit pas. Par exemple, 500 s'écrit « cinq cents » et

- on pas « cinq cent-zéro » !

De 1000 à 1 000 000

De 1000 à 999 999, on écrit les milliers sous la forme d'un nombre entre 1 et 999, suivi de « mille », puis d'un nombre entre 1 et 999. Ce sera plus clair avec un schéma :

Quelques autres exemples :

- 76 343 : soixante-seize mille trois cent quarante-trois ;
- 2 001 : deux mille un ;

- 300 222 : trois cent mille deux cent vingt-deux.

Bon je pense que vous avez compris...

Comme pour les centaines, si il n'y a qu'un seul millier, on ne note pas le « un ». Par exemple, 1446 s'écrit « mille quatre cent quarante-six » et non pas « un mille quatre cent quarante-six ». Et si les trois derniers chiffres sont des 0, on ne les dit pas. Par exemple 203000 s'énonce « deux cent trois mille » et non pas « deux cent trois mille zéro ».

Il n'en reste plus qu'un pour achever cette partie : le nombre 1 000 000 s'écrit « un million ».

Au delà du million...

Et c'est là que ça commence à se gâter. Pourquoi ? Et bien parce qu'il y a plusieurs façons de noter les nombres au delà du million. Mais il y a pire : ces méthodes utilisent les mêmes mots. Ainsi par exemple, le mot « billion » peut signifier soit « un million de millions » soit « mille millions » selon les cas.

Mais ne vous inquiétez pas, ce chapitre est là pour vous aider à y voir plus clair.

Il existe en réalité deux méthodes avec chacune de légères variantes :

- **L'échelle courte :** basée sur un vocabulaire qui va de mille en mille ;
- **L'échelle longue :** basée sur un vocabulaire qui va de million en million.

En 1948 la neuvième Conférence générale des Poids et mesures préconisa l'utilisation de l'échelle longue. C'est donc elle l'échelle officielle que je vous conseillerai d'utiliser. Cependant, l'échelle courte est encore beaucoup utilisée notamment dans les pays anglo-saxons, donc méfiez-vous.

Commençons donc par l'échelle *officielle* : la longue.

Échelle longue

Voici le vocabulaire utilisé dans l'échelle longue ;

nom	Valeur	Remarque
million	1 000 000	6 zéros
billion	1 000 000 000 000	12=2*6 zeros

trillion	1 000 000 000 000 000 000	18=3*6 zeros
quadrillion	1 000 000	24=4*6 zeros
quintillion	1 000 000	30=5*6 zeros
sextillion	1 000 000	36=6*6 zeros
septillion	1 000 000	42=7*6 zeros
octillion	1 000 000	48=8*6 zeros
nonillion	1 000 000	54=9*6 zeros

Encore heureux que louis le XIV^{éme} n'ai pas eu un trésor trop conséquent à comptabiliser. Nous aurions des sexdixtillion et autre bizarreries dans cette ménagerie du cirque des nombres.

Prenons un exemple, comment écrit-on 2 327 444 123 458 987 541 010 ? On commence par séparer notre nombre en paquet de 6 chiffres : 2 327444 123458 987541 010, et on écrit : deux mille trois cent vingt-sept trillions quatre cent quarante-quatre mille cent vingt-trois billions quatre cent cinquante-huit mille neuf cent quatre-vingt-sept millions cinq cent quarante et un mille dix. Ouf ! >_

Le plus grand nombre qui puisse être écrit et prononcé avec ce système est : 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999 999. Qui se prononce de la façon suivante :

Citation : Le plus grand nombre que l'on peut écrire et prononcer

neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **decillions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **nonillions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **octillions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **septillions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **sextillions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **quintillions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **quadrillions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **trillions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **billions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf **millions** neuf cent quatre-vingt-dix-neuf mille neuf cent quatre-vingt-dix-neuf. (suivi d'un Att.. choum !!)

Allez courage, prenez une grande respiration et prononcez-le à voix haute. Il faut l'avoir fait au moins une fois dans sa vie..

Même si ce n'est pas une nomenclature officielle, certains nombres bien plus grands ont tout de même reçus un nom à eux. C'est le cas par exemple du gogol qui est égal à 10^{100} (10^100) soit un 1 suivi de 100 zéros. Le gogolplex à lui est égal à 10^{gogol}, ($10^{\wedge gogol}$) un 1 suivit d'un gogol de zéros ! (difficile à compter sur les deux mains !)

Les -ard

Lézard ? Qui a dit lézard ??? Il est où le lézard ??

Non, il n'est pas question de reptiles ici, mais simplement d'une variante du système précédent qui utilise des terminaisons en -ard. Un truc-liard vaut 1000 truc-lion, ou « truc » est l'un des préfixe que nous avons vu précédemment. Par exemple, « un milliard = 1000 millions », « un billiard = 1000 billions », « un trilliard = 1000 trillions » et ainsi de suite.

En reprenant le même exemple que tout à l'heure, le nombre 2 327 444 123 458 987 541 010 s'écrit : *Deux trilliards trois cent vingt-sept trillions quatre cent quarante-quatre billiards cent vingt-trois*

billions quatre cent cinquante-huit milliards neuf cent quatre-vingt-sept millions cinq cent quarante-et-un mille dix !

On a une alternance tous les 3 zéros (10^3) de ards et de ions.

Échelle courte

L'échelle courte utilise le même vocabulaire, mais en ne multipliant que par 1000 au lieu de 1000000 entre deux :

nom	Valeur	Remarque
million	1 000 000	6 zéros
billion	1 000 000 000	9=(2+1)*3 zeros
trillion	1 000 000 000 000	12=(3+1)*3 zeros
quadrillion	1 000 000	15=(4+1)*3 zeros
quintillion	1 000 000	18=(5+1)*3 zeros
sextillion	1 000 000	21=(6+1)*3 zeros
septillion	1 000 000	24=(7+1)*3 zeros
octillion	1 000 000	27=(8+1)*3 zeros
nonillion	1 000 000	30=(9+1)*3 zeros

Reprenons le même exemple que pour l'échelle longue : 2 327 444 123 458 987 541 010. Dans l'échelle courte, il s'écrit : deux sextillions trois cent vingt-sept quintillions quatre cent quarante-quatre quadrillions cent vingt-trois trillions quatre cent cinquante-huit billions neuf cent quatre-vingt-sept millions cinq cent quarante et un mille dix.

Pensez surtout à vous méfier du mot **billion**. Si les autres ne sont quasiment jamais utilisés le billion est quant à lui plus fréquent. Il arrive notamment qu'on le trouve dans des traductions de textes en anglais dans lesquelles il aurait normalement fallut qu'il soit traduit par milliard !

Voilà. Cette annexe sur l'écriture des nombres en lettres s'arrête là !

Vous savez maintenant compter très loin, mais remarquez que même en prononçant trois nombres par seconde et en vivant cent ans, vous atteindrez à peine 10 000 000 000. Dix milliards en échelle longue, dix billions en échelle courte ! À vrai dire, même si toute l'humanité s'y mettait, il serait impossible de prononcer tous les nombres que nous savons prononcer avant la fin du système solaire (et la destruction de la Terre qui en découle) prévue dans 5 milliards d'années (échelle longue) ! Dérisoire n'est-ce pas ?

II Conversion de ces règles en algorithme :

II .1 Limitations du champ d'action :

Bon, compte tenu de ce que nous avons déjà raconté, il est bon de limiter le champ de notre propos en s'imposant un cadre :

1 : Utilisation d'un minimum de mots vocales ex : quatre-vingt-un est composé de trois mots (suivant la nouvelle règle en vigueur depuis 1990, les mots sont séparés par des -)

2 : On désire énoncer des nombres à virgule avec 2 chiffres après la virgule au maximum (vous l'aurez compris, on se rapproche de la finance de la rue : les prix)

3 : On s'arrête en valeur absolue à neuf-cent-quatre-vingt-dix-neuf milliard (1000 millions moins 1) donc on s'affranchi de la représentation courte ou longue (les personnes pouvant discuter sur des sommes supérieures à 999 milliards d'euros ne sont pas légion !!)

4 : On est en France après Louis XIV donc: exit les octante huitante et nonante

5 : On se situe dans la perspective du maintient de la monnaie unique Européenne, et donc depuis 2001, le Franc à été remplacé par l'Euro.

6 : On désire pouvoir énoncer les nombres négatifs.

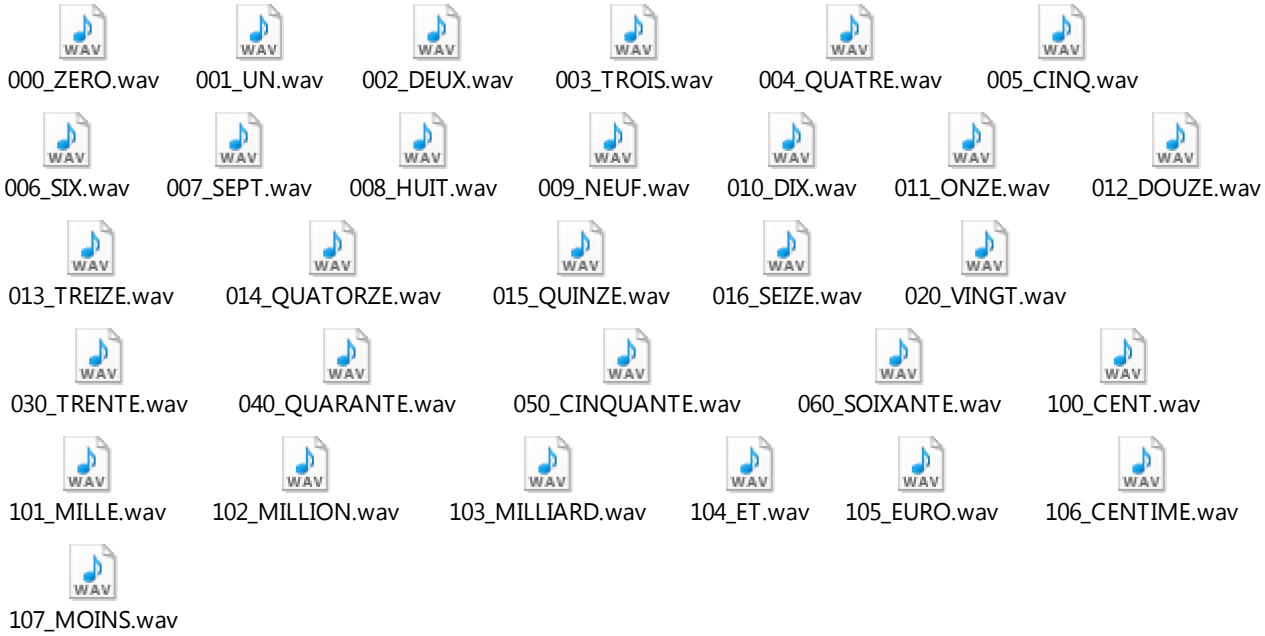
.

II.2 les mots utiles et leur stockage :

Nos "dogmes" N°1 et N°5 nous imposent la liste suivante : *zéro, un, deux, trois, quatre, cinq, six, sept, huit, neuf, dix, onze, douze, treize, quatorze, quinze, seize, vingt, trente, quarante, cinquante, soixante, cent, mille, million, milliard, virgule, et, euros, centimes* auquel nous pouvons ajouter *moins* si on étend notre propos aux nombres négatifs. Soit un total de 30 fichiers son unitaire.

Ces fichiers seront stockés dans une carte de type SD sous forme de fichiers wav 16 bits stéréo compréhensibles par notre carte "WavTrigger" qui constitue avec son haut-parleur associé les cordes vocales de notre système.

Nous trouverons donc les fichiers suivants :



Le numéro de début de fichier permet grâce à la méthode de l'objet wavtriger de jouer le mot correspondant

void dit(int n) // on prononce ou enonce les phonèmes/mots

```
{
   // lit le fichier son
  carteWave.trackPlaySolo(n);
  // attend que le fichier soit fini de lire
  // en attendant la réponse de la carte son
  while (carteWave.isTrackPlaying(n)) {delay(5);}
}
```

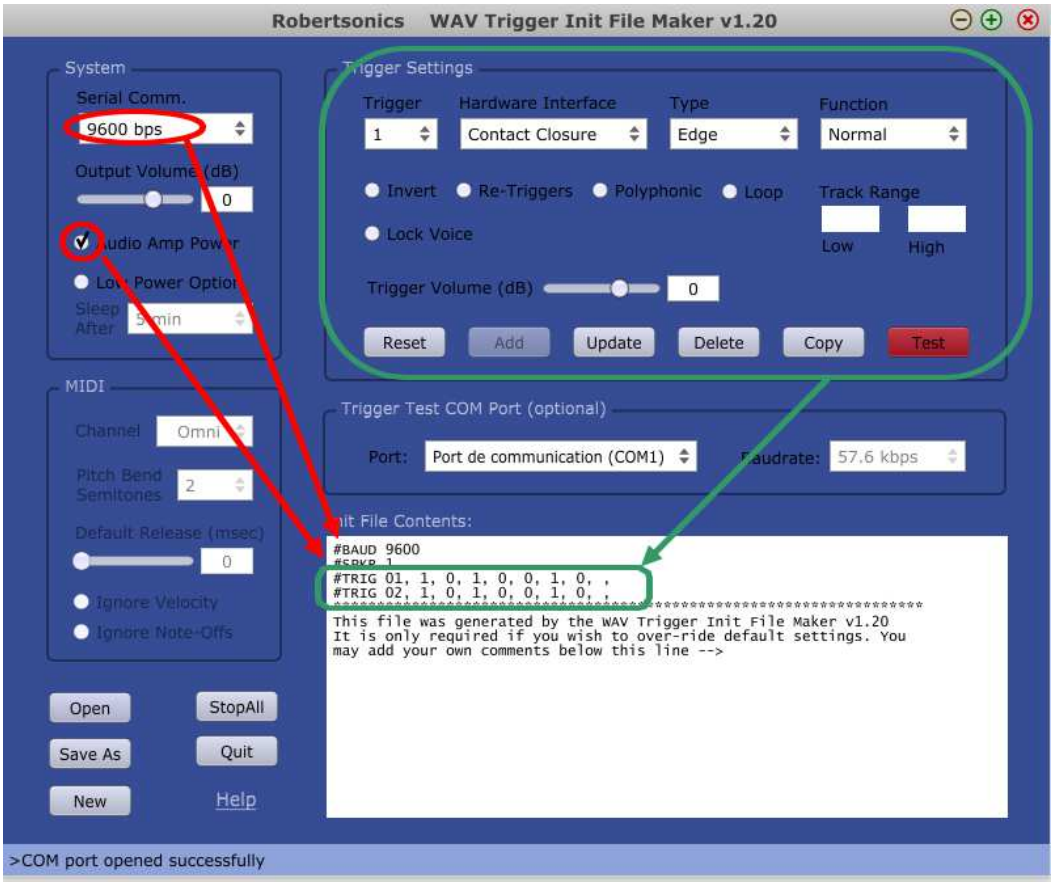
Les deux fichiers suivants sont : le fichier de configuration de la carte son généré par le programme sous Windows WT-InitMaker... (se rapporter à la doc de cette carte pour la configuration)


wavtrigr.ini


WT-InitMaker_120_20161210.exe

le fichier .ini doit pour le bon fonctionnement résider dans la racine de la carte.

lors de la configuration "automatique" du .ini, attention de bien préciser les indications en rouge pour pouvoir communiquer à l'aide de la sortie série. Les indications en vert correspondent à la manipulation via" contact sec" sur les broches de la carte, elles ne sont là que pour vérifier à la mise au point que la carte vit bien.



Attention : en 2016, Pour avoir une conversation bi-directionnelle avec la carte, il est nécessaire d'effectuer l'upgrade du firmware dans la dernière version disponible. On peut espérer que les cartes fournies ultérieurement le seront par défaut. Pour ce faire, il est nécessaire d'avoir une petite carte de programmation USB->serial (désormais classique) pour pouvoir transmettre le firmware. Si vous

deviez en acheter une, veillez à ce qu'elle soit commutable 5v, 3.3v pour s'adapter à tout votre bestiaire.

II.3 Le stockage du nombre à énoncer :

Pour conserver un maximum de précision, nous stockerons le nombre à énoncer sous forme de deux entités entières codées sur un format long long (64 bits) sous forme"valeur_entiere, valeur_decimale". le signe du nombre correspondant au signe de la valeur entière.
en prenant deux digits au maximum pour la valeur décimale, nous pourrons donc coder notre nombre de -9223372036854775808 ,*99* à 9223372036854775807,*99* soit une dette abyssale et un gain impossible même à "l'eurotrillon"

II.4 La stratégie :

II.4.1 La base du système

On traite le nombre par paquets de 3 digits en descendant vers les unités
Dans notre cas, on commence par les milliards

soit n notre nombre entier
les procédures de base sans intégrer les exceptions du Français sont en pseudo-code :

Procedure traite n ***si*** n>999999999 (1milliard-1) ***debut*** extraire les milliards : m traitecent m enoncer "milliard" n= n-m*1milliard ***fin*** traitemillion n	***Procedure traitemillions n*** ***si*** n>999999 (1 million-1) ***debut*** extraire les millions : m traitecent m enoncer "million" n= n-m*1million ***fin*** traitmillier n
Procedure traitemilliers n ***si*** n>999 (1 millier-1) ***debut*** extraire les milliers : m traitecent m enoncer "mille" n= n-m*1mille ***fin*** traitecent n	***Procedure traitecent n*** ***si*** n>99 (cent -1) ***debut*** extraire les centaines : c traiteunite c enoncer "cent" n=n-c*100 ***fin*** traitedeci n
Procedure traitedeci n ***si*** n>9 (10 -1) ***debut*** extraire les dixaines : d traiteunite d enoncer "dixaine" n=n-c*10	***Procedure traitunite n*** ***si*** n==0 enoncer "zero" ***si*** n==1 enoncer "un" ***si*** n==2 enoncer "deux" ... ***si*** n==9 enoncer "neuf"

fin traiteunite n	

Voila c'est fini! mais cela ne fonctionne pas !!
1001356 donne par le processus ci-dessus
"un" "million" "un" "mille" "trois" "cent" "cinq" "dixaine" "six"

les bizarreries de la langue Française doivent nous donner normalement
"1" "mllion" "mille" "trois" "cent" "cinquante" "six"

Il faut désormais intégrer les exceptions en tout genre.

II.4.2 Les corrections dues aux exceptions

Le "coup" des un mille sera traité dans

Procedure traitemilliers n
si n>999 (1 millier-1)
debut
extraire les milliers : m
si m>1
debut
traitecent m
fin
enoncer "milles"
n= n-m*1mille
fin
traitecent n

Bon maintenant avec l'exemple précédent nous obtenons :
"un" "million""mille" "trois" "cent" "cinq" "dixaine" "six"

Et ce n'est pas fini prenons un autre exemple :

1000000 donne

"un" "million" "zero" "mille" "un"

Vous aurez compris, on n'énonce pas un paquet de trois chiffres valant zéro (traité par traitecent) si nous en avons déjà énoncé un de rang supérieur. dans ce cas on se doit de passer de procédure à procédure cette indication. ceci se fait via un booléen que la procédure appelante modifie avant d'appeler la procédure de rang inférieur.

Nous modifions donc toutes nos procédure comme suit

Procedure traitemilliers n, enoncezero
si n>999 (1 millier-1)
debut
extraire les milliers : m
si m>1
debut

```
                traitecent ( m, enoncezero)
            fin
        enoncer "milles"
        n=  n-m*1mille
        enoncezero=false
        fin
traitecent (n,  enoncezero)
```

et traitecent comme suit :

```
Procedure traitecent n , enoncezero
si n>99 (cent -1)
        debut
        extraire les centaines : c
        traiteunite c
        enoncer "cent"
        n=n-c*100
        fin
if (n!=0 ou enoncezero) traitedeci n, enoncezero
```

Le premier appel se fera avec enoncezero=true

maintenant nous obtenons

"un" "million" "un"

Ouf nous avons bien avancé mais la majeur partie des exceptions ayant lieu pour des valeur inférieur à 99, il reste du travail. Cette partie est tellement "tordue" qu'il est nécessaire de repenser la totalité des deux procédures traitant dizaines et unités (nombres <=99) dans du "cousu-main"

les modifications des pseudo-code sont loin d'être mineures, et, il faut les intégrer en trois temps. Nous présenterons ici le code C++ correspondant.

```
// ********************************************************
// traitement des nombres de 0 à 16
void zero_16(byte n, bool z) // traitement de 0 à 16
{
  if ( (n != 0) || z)
  {
    dit(n);
  }
}
// ********************************************************
// traitement des nombres de 0 à 19
void zero_19(byte n, bool z) // traitement de 0 à 19
{
  if (n < 17) {
    zero_16(n, z);
  }
  else
  {
    dit(10);
```

```
    zero_16(n - 10, false);
  }
}
// ********************************************************
// traitement des nombres de 0 à 99
// la partie entiere de ce nombre peut entrer sur un byte
// 99 < 254
// 99 < max_byte

void zero_99(byte n, bool z) // traitement de 0 à 99 (cent -1)
{ if (n < 20)
  {
    zero_19(n, z);
  }
  else
  {
    int dixaine = n / 10;
    if ( (dixaine == 9) || (dixaine == 7)) // le cas de septante et nonante qui se disent
    // respectivement soixante <reste> ou quatre vingt <reste> ou reste est un nombre entre 0 et 20
    {
      dixaine--;
    };
    if (dixaine == 8) // le cas de octante qui se dit quatre vingt
    {
      dit (4);
      dit (20);
    }
    else
    {
      dit(dixaine * 10);
    }
    int reste = n - dixaine * 10;
    if (reste > 0)
    {
      if ((reste == 1) & (dixaine != 8)) {
        dit(_et); // on dit cinquante et un et non quate-vingt-et-un
      }
      zero_19(reste, false);
    }
  }
}
```

Pour finir le code de la partie maitresse qui commande le tout

```
/ ************************************************
// enonce un un prix en euros en fonction de la valeur entière
// et de la valeur decimale
void enonce_prix(long long  valent, unsigned long decimale)
{
  if (valent < 0)
  {
    valent = -valent;
```

```
  dit(_moins);
 }

 zero_999999999999(valent, true); // le coup de l'enonce des zero
 dit(_euro); // on precise l'unité de la monaie avant les centimes dont
             // l'unité deviens implicite

 if (decimale > 0)
 {
  dit(_et);
  zero_999(decimale, true);
  dit(_centime);
 }
}
```

Pour le reste, je vous laisse le soins de vous plonger dans le code source.

II.5 Remarques

Dans le code source, vous verrez que l'on a utilisé un type non documenté dans les références en lignes : "*long long*" il s'agit d'une représentation sur 64 bits des entiers. de même, les constantes ecrites avec LL à la fin sont codées en *long long*. Ce type de variables et constantes doivent être utilisés avec parcimonie. En effet, leur traitement est au bas mot 16 fois plus long que celui d'un *byte* classique. Elles occupent 8 octets en mémoire (et les neurones de notre bébête adorée ne sont pas nombreux!).
Les fonctions print et println ne comprennent pas ce type de donnée. Il faut donc décomposer un long long en partie de poids fort et partie de poids faible pour pouvoir l'afficher.

exemple :

```
// ent est du type long long
long long sto;

if (ent < 0) {Serial.print('-') }

sto = abs(ent) / 0xFFFFFFFFFF;
if (abs(sto) > 0) {Serial.print(sto); }

sto = abs(ent) % 0xFFFFFFFFFF;
Serial.print(sto);
```

Bibliographie :

https://openclassrooms.com/courses/nombres-et-operations-1/prononcer-et-ecrire-les-nombres-en-francais
https://fr.wikipedia.org/wiki/%C3%89chelles_longue_et_courte
https://fr.wikipedia.org/wiki/Gogolplex
http://www.panamaths.net/Documents/SecSci/Histoire de la numeration.pdf
http://www.altituduino.xyz/