



synthétiseur de bruit

Du bruit à la musique avec le PRBSynth1

Raymond Schouten (Pays-Bas)

Le bruit est souvent utilisé en musique électronique et dans les effets sonores. Le synthétiseur décrit ici invite le lecteur à explorer les sons à base de bruit. Le matériel est assez simple, un Arduino Pro Mini étendu avec un CN/A, un contrôle de volume et un circuit de panoramique stéréo, ainsi qu'une entrée et une sortie MIDI. Le logiciel réalise la plupart des fonctions de génération du son sur quatre types de filtres et un puissant LFO (oscillateur BF) qui peut aussi contrôler le son par des motifs rythmiques. L'espace d'expérimentation du matériel et du logiciel est vaste – amusez-vous !



Le PRBSynth1 est un synthétiseur musical destiné à produire, façonner et moduler des sons à base de bruit. Si de nombreux sons de synthétiseurs classiques peuvent être créés, l'unité permet aussi d'explorer d'autres espaces sonores.

« Cherchez votre propre voie »

Si cet article décrit le synthétiseur, ce n'est pas son unique propos : il vous invite à l'expérimentation, soit en l'adaptant, soit en utilisant certaines des idées présentées ici (voir les encadrés). La réalisation peut débuter par une version réduite sur une plaque d'essai sans le MIDI et le circuit de panoramique avec son potentiomètre et ses LED (exemple disponible). Un Arduino Uno peut être utilisé bien que ses six entrées analogiques limitent à six le nombre de potentiomètres utilisables. Une autre option est de réaliser une version MIDI seulement, sans potentiomètres (il faut désactiver la partie du programme qui les lit).

Le moteur sonore

L'architecture du PRBSynth1 est assez classique (voir **figure 1**). Un son est réalisé en jouant sur huit fonctions puissantes. Elles sont commandées soit par huit potentiomètres, soit par MIDI. Il n'y a pas d'affichage. Le synthétiseur répond aussi aux messages MIDI *Note-On* et *Note-Off* avec des données de vélocité. Un 9^e potentiomètre règle le niveau de sortie et son interrupteur intégré connecte la batterie interne. Quelques LED renseignent sur le mode de fonctionnement du synthétiseur.

Générateur de bruit

Dans un synthétiseur classique, un oscillateur produit le son sous forme d'ondes sinusoïdales, en dents de scie ou carrées. Ici, l'oscillateur est remplacé par un générateur de bruit qui en produit quatre types : blanc, rose, rouge et popcorn.

Le spectre du bruit blanc est horizontal, c.-à-d. que comme pour la lumière blanche, chaque fréquence a la même intensité. C'est pourquoi on l'appelle bruit blanc. Il ressemble au son d'une radio entre deux stations. En accentuant certaines parties du spectre, on obtient d'autres *couleurs* de bruit.

Le bruit rose en $1/f$ a un spectre qui décroît

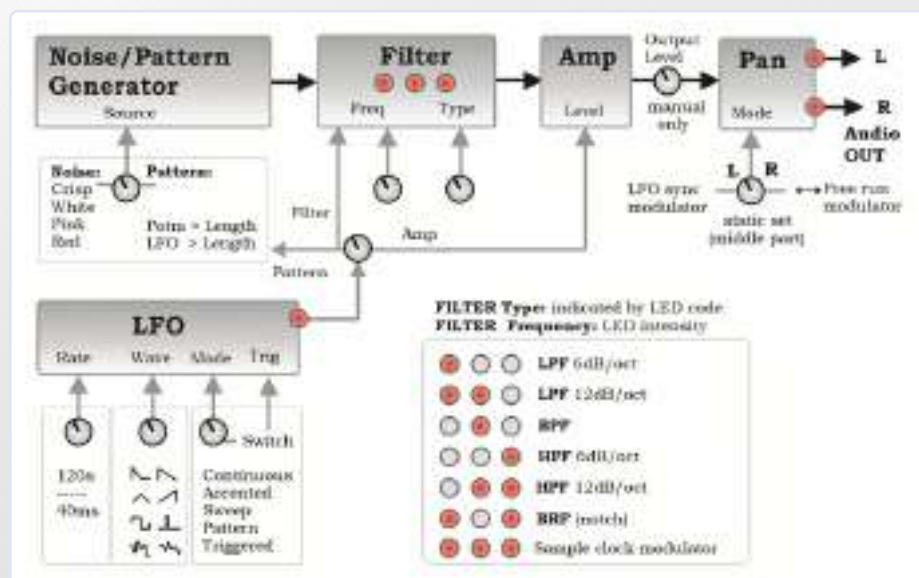


Figure 1. Schéma-bloc du PRBSynth1. Chaque bloc possède une ou plusieurs commandes sous forme de potentiomètres ou de paramètres MIDI CC. Les points orange représentent des LED.

de 10 dB quand la fréquence croît d'un facteur 10, pour le bruit rouge ou brownien, c'est 20 dB. Le bruit de pop-corn est un crépitement dû à l'écrêtage de la sortie du générateur de bruit. Il en existe deux sortes. Le bruit est produit numériquement par logiciel à l'aide d'un registre à décalage et d'une rétroaction. (Voir l'encadré **Pour faire du bruit, lancez des pièces en l'air !**) En plaçant judicieusement les rétroactions, le flux binaire de sortie devient quasi aléatoire. Un tel flux de bits est une séquence binaire pseudoaléatoire, ou *PRBS* (d'où le nom du synthé : PRBSynth). Elle est pseudoaléatoire, car le flux de bits de sortie se répète périodiquement. Lorsque le registre à décalage est assez long, la période du flux est très longue et il peut être considéré comme du bruit. Donc, techniquement parlant, un tel générateur de bruit est un oscillateur à forme d'onde très complexe et à très longue période.

Le PRBSynth1 permet aussi de contrôler la longueur de la séquence aléatoire pour passer progressivement du bruit à des motifs répétitifs au son aigu. C'est le mode *pattern* (motif). La hauteur de ces motifs est calée sur des notes de musique et peut être jouée sur un clavier MIDI.

Filtres

La sortie du générateur de bruit est connectée à un filtre. C'est un filtre multimode créé par logiciel (voir l'encadré **Un logiciel de filtres presque trivial**) avec modes passe-bas, passe-bande, passe-haut et réjection de bande (*notch*). Les filtres passe-bas/haut sont déclinés en deux pentes : 6 dB/octave ou 1^{er} ordre et 12 dB/octave ou 2^e ordre. On

ne peut utiliser qu'un seul filtre à la fois.

Un potentiomètre règle la fréquence de coupure ; la résonance (ou facteur de qualité, Q) est fixe, mais divers pré-réglages de Q existent : trois pour le filtre passe-bas à 12 dB, cinq pour le filtre passe-bande ; le filtre passe-haut à 12 dB et le filtre coupe-bande en ont chacun deux.

Cela donne un total de 14 pré-réglages. Il en existe un 15^e : c'est un filtre passe-bas à fréquence fixe, avec une fréquence d'échantillonnage modulée.

Trois LED codent la variante de filtre activée. L'intensité des LED reflète la fréquence de coupure.

Amplificateur

La sortie du filtre est suivie d'un amplificateur. Le lecteur attentif aura remarqué l'absence de générateurs d'enveloppe (*ADSR*) typiques des synthés (analogiques). Le PRBSynth1 n'en possède pas, mais, réglé en mode « déclenché », le LFO se transforme en générateur d'enveloppe (voir ci-après). En appuyant sur le clavier ou le commutateur de déclenchement, le LFO exécute un seul cycle. Les potentiomètres de forme d'onde et de vitesse définissent respectivement la forme de l'enveloppe et la durée. L'enveloppe peut contrôler la fréquence de coupure du filtre ou l'amplitude du signal. En outre, l'amplificateur transmet le signal si une touche (MIDI) est enfoncée, de sorte que le son est coupé en relâchant la touche.

Le volume du signal numérique est contrôlable par MIDI et bien que cela agisse sur le volume global, ce n'est pas recommandé. Pour une

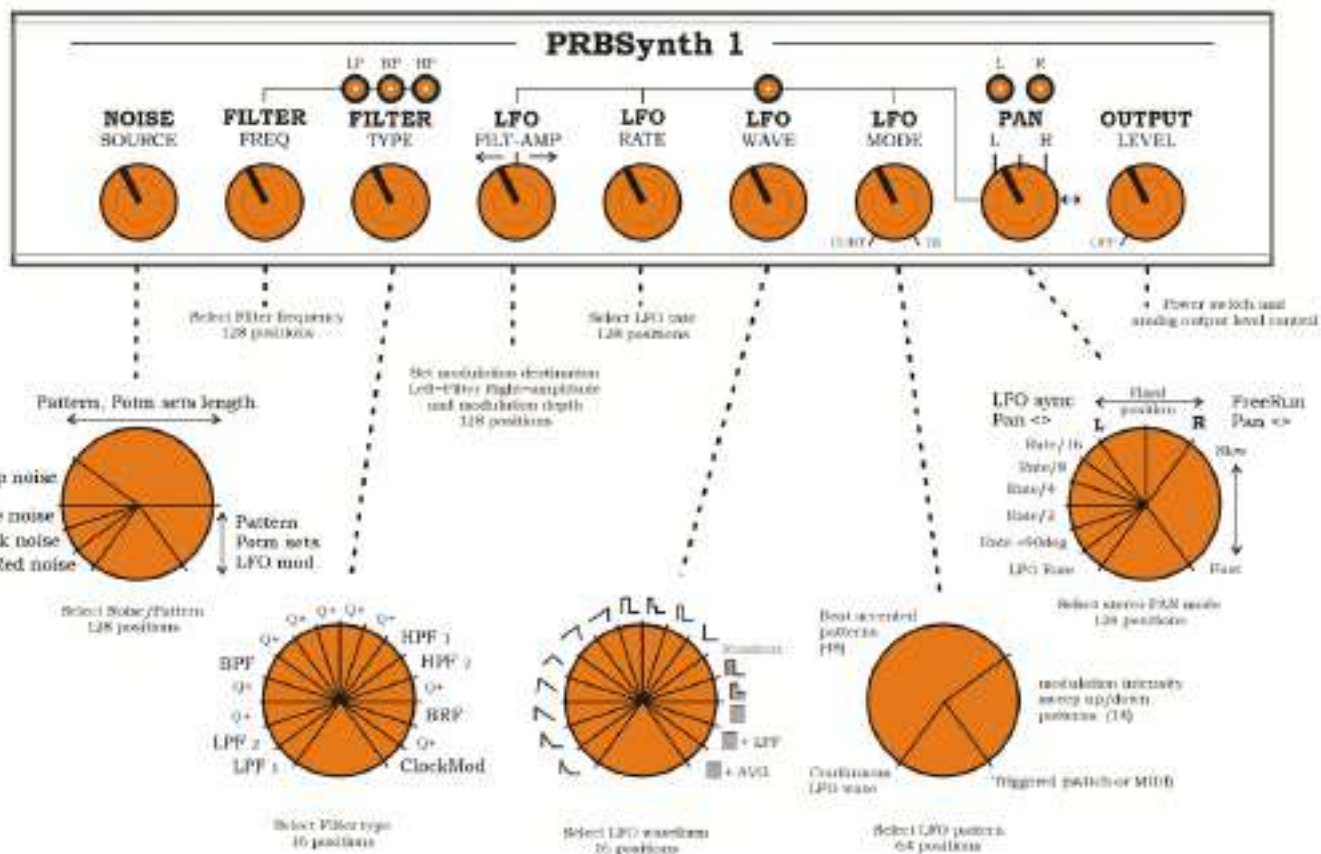


Figure 2. Synthèse des fonctions des potentiomètres du synthétiseur. Certains des boutons agissent en partie comme commutateur multiposition et en partie comme potentiomètre. Les échelles sont les mêmes que pour les commandes MIDI CC équivalentes. Un dessin plus grand est disponible sur la page web du labo [1].

qualité optimale du signal, il faut s'efforcer d'avoir un signal numérique de volume maximal. Un potentiomètre placé après le convertisseur numérique-analogique (CN/A, voir ci-après) règle le niveau de sortie analogique du synthétiseur.

Panoramique stéréo

Bien que le signal de sortie du synthétiseur soit monophonique, le panoramique le place où l'on veut dans l'image stéréo. La sortie du synthétiseur est donc stéréo.

Un seul potentiomètre permet de positionner le signal manuellement de gauche à droite et de donner le contrôle au LFO ou à un modulateur indépendant ayant sa propre horloge. Deux LED montrent ce qui se passe.

Oscillateur basse fréquence

Un oscillateur BF (LFO) ajoute une certaine variation aux sons synthétiques statiques. Par ex. si le LFO balaie lentement la fréquence de coupure du filtre, on crée des sons évoquant le vent, la mer ou la pluie. Le LFO peut

également moduler l'amplitude du signal, la longueur du registre à décalage qui génère le bruit et la position stéréo du signal de sortie. La période du LFO est réglable entre 0,04 s (25 Hz) et 120 s. Outre le contrôle de période, le LFO dispose de 16 formes d'onde, incluant des dents de scie variées, un triangle, quelques impulsions et des formes aléatoires. La vitesse du LFO et la forme d'onde sont indiquées par une seule LED.

Un potentiomètre pilote le LFO selon quatre modes de fonctionnement. Avec le potentiomètre à fond à gauche, le LFO est toujours actif, comme un LFO classique. C'est le mode 1. Le 2^e ajoute des accents à partir d'une table de motifs. Le potentiomètre de mode LFO permet de choisir parmi 48 motifs rythmiques, la plupart d'une longueur de huit pas et 12 d'une longueur de six.

Le 3^e mode offre 16 motifs de balayage de la profondeur du LFO dont trois d'une longueur de 12 mesures. Tous les autres font 16 mesures. Le bouton de mode LFO sélectionne le type de balayage.

Si le bouton de mode est à fond à droite, le 4^e mode est activé. Ce mode lance un cycle LFO unique quand on appuie sur le bouton de déclenchement ou sur une touche du clavier MIDI. Il applique manuellement une modulation de type enveloppe.

Interface utilisateur

Avec seulement huit potentiomètres (sans compter celui du volume de sortie), l'utilisateur accède à de nombreux paramètres sonores. La division en secteurs de ces potentiomètres permet d'en tirer le maximum. Seuls les boutons de fréquence de coupure du filtre et de vitesse du LFO agissent comme des potentiomètres. Les autres, dont certains ont un grand nombre de secteurs, tiennent plus du commutateur multiposition. C'est pourquoi une surface de contrôle MIDI est utile. En effet, l'affichage numérique des valeurs CC (contrôle continu) y est fréquent et permet de savoir où le bouton se situe sur son échelle. La **figure 2** montre la répartition des différentes fonctions sur les potentiomètres.

À titre d'exemple, prenons le potentiomètre de la source de bruit (code MIDI CC 75). Le secteur de -135° à -45° (valeurs MIDI CC 0 à 39) agit comme un interrupteur de sélection entre différentes couleurs de bruit. C'est le mode *noise* (bruit). De -45° à $+90^\circ$ (MIDI CC 40 à 111), ce potentiomètre détermine la longueur du registre à décalage. Le synthé est alors en mode *pattern* (motif). Le dernier secteur, de $+90^\circ$ à $+135^\circ$ (MIDI CC 112 à 127) règle la profondeur de modulation LFO de la longueur du registre à décalage.

Ce n'est pas l'interface la plus intuitive qui soit, et au début de l'utilisation du PRBSynth1, il faut fréquemment se référer à la figure 2, mais certaines LED aident à appréhender certains réglages. Un dessin plus grand peut être téléchargé sur [1]. Une autre solution serait de réaliser une face avant avec ces informations imprimées autour des boutons.

Trois LED indiquent le type de filtre sélectionné ; la fréquence de coupure du filtre détermine son intensité. La LED LFO scintille au rythme du LFO et son intensité suit la forme d'onde sélectionnée. La position stéréo du signal est matérialisée par deux LED. Enfin, il y a un interrupteur de déclenchement qui permet soit de lancer un seul cycle du LFO (mode « déclenché »), soit de modifier les accents du LFO ; ce mode est intéressant pour les sons rythmiques.

Aperçu du circuit

Le schéma du PRBSynth1 (fig. 3) reste simple car la plupart des fonctions sont réalisées en logiciel. On y voit un Arduino Pro Mini entouré d'un CN/A, d'une commande de volume et d'un circuit de panoramique stéréo, ainsi que d'une entrée/sortie MIDI.

Un convertisseur N/A de 7,75 bits

À première vue, le CN/A qui convertit le signal audio numérique en signal analogique semble être un simple réseau de résistances pondérées de 6 bits, mais les valeurs des résistances et leur commande cachent quelques astuces. Ce CN/A à 6 bits peut produire 216 valeurs de sortie au lieu des 64 attendues. Comment est-ce possible ? Grâce aux trois bits inférieurs (LSB) qui peuvent prendre trois valeurs ('0', 'ouvert' et '1')

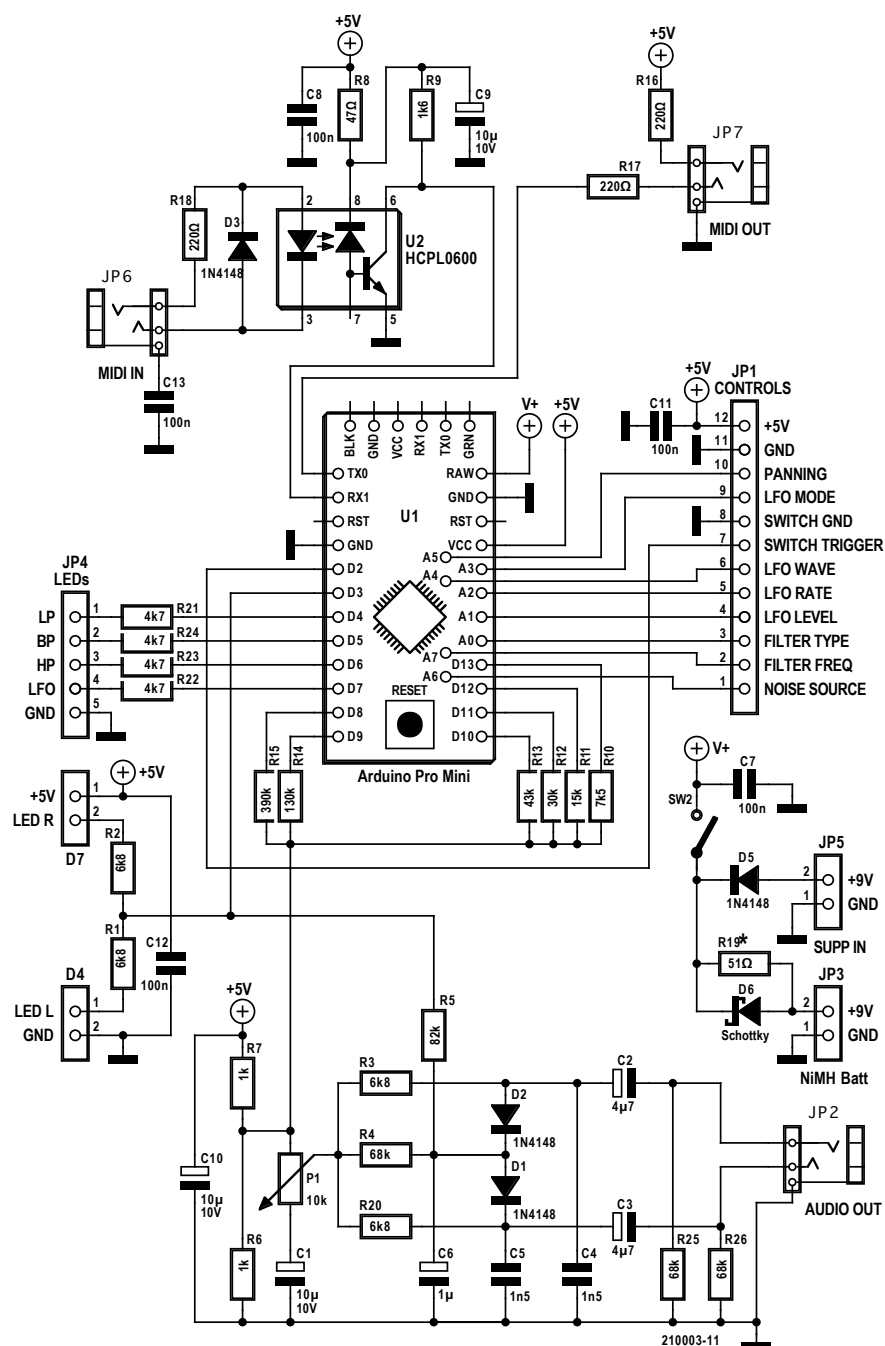


Figure 3. Un Arduino Pro Mini, un optocoupleur et une poignée de composants passifs constituent le circuit complet. Les plus remarquables sont le circuit de panoramique stéréo (D1 et D2) et le BiTriDAC à 6 bits (R10 à R15).

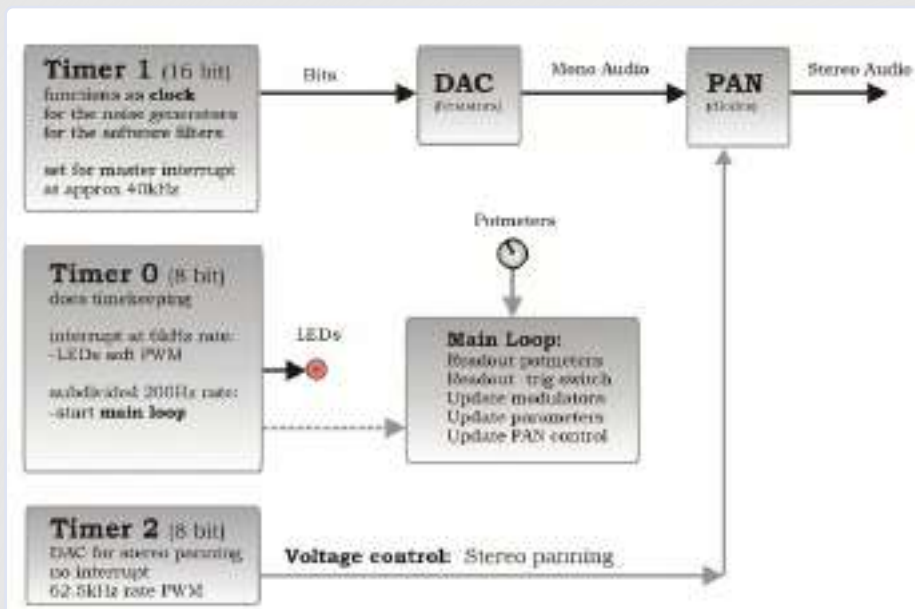


Figure 4 : Le moteur sonore est piloté par des timers. Le Timer 1 à haute fréquence produit des échantillons qui sont modulés par le Timer 0 à basse fréquence. Le Timer 2 commande le panoramique stéréo. Le traitement des données MIDI (non illustré) est effectué lorsque les timers tournent.

au lieu de deux : ce sont des bits trinaires. Les trois bits supérieurs (MSB) sont des bits ordinaires. Comme j'aime l'appeler, ce *BiTriDac* a une résolution de 7,75 bits.

La contrepartie est un logiciel un peu plus complexe, parce que les valeurs de 8 bits doivent être réparties (mappées) sur 7,75 bits et que la mise en place d'un bit trinaire est plus compliquée. Mais la fonction de mappage applique un écrêtage léger, ajoutant au son la touche analogique recherchée.

Pour une performance optimale du CN/A, il est vivement conseillé de retirer de l'Arduino Pro Mini la LED (ou mieux, sa résistance série) reliée à la broche 13 (à côté du bouton *reset*) car elle est en parallèle avec R10 du CN/A. Cela permet d'améliorer la précision du CN/A et d'économiser un peu d'énergie.

Panoramique stéréo commandé en tension

Les diodes D1 & D2 agissent comme des résistances CA variables qui atténuent le

petit signal audio (moins de 200 mV) pour les voies gauche (D2) et droite (D1). Les deux diodes sont polarisées en moyenne à 2,5 V par le curseur de volume P1. L'atténuation est

pilotée par les courants continus parcourant les diodes. Ils sont influencés par la tension sur le nœud R5/C6/D1/D2. Cette tension est une version filtrée du signal MLI sur la broche 3 de l'Arduino Pro Mini. La dissymétrie du courant des diodes déplace le son dans le champ stéréo. Pour que l'effet soit un peu plus doux, le panoramique est limité à environ 10 dB de déséquilibre.

Stricto sensu, ce circuit n'est pas un simple panoramique stéréo car il ajoute également une modulation d'amplitude (trémolo) tout en déplaçant le son. Appliqué à un synthétiseur, cela peut être considéré comme un plus. Il en va de même pour la distorsion ajoutée au signal d'entrée par la caractéristique de transfert non-linéaire des diodes pour les forts signaux.

Pour finir, une note sur R19. On peut monter cette résistance dans le cas où l'appareil est

Un logiciel de filtres presque trivial

Pour programmer les filtres, il suffit d'imiter un filtre analogique. Le logiciel se contente d'imiter le flux de signaux des filtres analogiques et ne nécessite que quatre lignes de code :

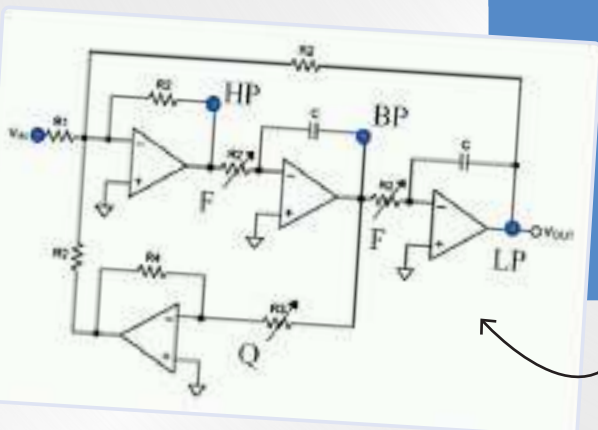
```
LP = LP + F*BP; // F est le réglage de la fréquence du filtre
HP = Vin - LP - q*BP; // note : q = 1/Q, évite une division
BP = BP + F*HP;
DAC = LP; // ou sélectionner DAC=HP, DAC=BP
```

Et les deux intégrateurs ? Le code s'exécute à une vitesse fixe. Par conséquent, reprendre la valeur précédente et y ajouter une variable équivaut à une intégration analogique (voir la 1^{re} ligne du code et aussi la 3^e ligne).

Disposer d'un processeur à 8 bits avec 2 Ko de mémoire pour calculer un filtre logiciel implique un code compact et rapide dans la partie traitement audio. L'utilisation d'octets et, dans certains cas, d'entiers de 16 bits est la meilleure solution. Il faut bannir les divisions dans la partie traitement audio.

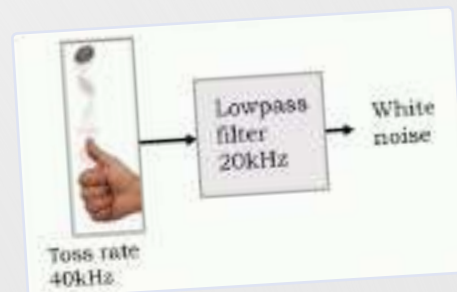
Dans le PRBSynth1, ce filtre logiciel fonctionne à 40 kHz* max. et les variables LP, BP, HP sont des entiers. Les autres variables sont des octets. Cela donne-t-il un filtre « Hi-Fi » ? Non, mais comme le signal d'entrée est du bruit, nous pouvons faire l'impasse sans trop d'artefacts audibles.

*Astuce : en mode *noise* (bruit), la plage d'accord du filtre est étendue d'une octave environ en ralentissant lentement l'horloge d'échantillonnage pour les plus basses fréquences de coupure du filtre. Comme en entrée on a du bruit, il n'y a pas de problème de crénelage.



Potentiomètre	Code CC
Fréquence de coupure du filtre	1
Volume (toujours maximiser)	7
Panorama	10
Type de filtre & Q	73
Source de bruit	75
Profondeur de modulation LFO	76
Vitesse LFO	77
Onde LFO	78
Mode LFO	79

Tableau 1. Correspondance entre le potentiomètre et le code MIDI CC.



Pour faire du bruit, lancez des pièces en l'air !

Si on tire à pile ou face de façon répétitive et rapide, on obtient un flux binaire similaire à un signal MLI aléatoire. En traitant ce signal avec un filtre passe-bas, on obtient un flux de valeurs moyennes fluctuant de façon aléatoire entre 0 et 1.

On peut simuler le lancer d'une pièce de monnaie dans un logiciel en produisant une longue séquence de valeurs aléatoires, dont on n'utilise qu'un seul bit (par ex. le bit 0). Le résultat n'est plus vraiment aléatoire, mais il en est suffisamment proche pour être utilisé comme tel. Ce type de séquence s'appelle séquence de bits pseudoaléatoire (PRBS, d'où le nom du synthé) et peut être générée à l'aide d'un registre à décalage dit *tapped* (dont les bits peuvent être lus) et d'un XOR de rétroaction. Si la séquence est assez longue (quelques secondes ou plus), elle ressemblera à du bruit. On peut créer un tel registre à décalage avec rétroaction par logiciel en seulement deux lignes de code :

```
// Un registre à décalage de 16 bits et XOR sur les bits 16, 15, 13 et 4 produit une PRBS.
// init. sr comme mot sr=1, init. DAC comme octet.
sr = (sr<<1 | (((sr>>15)&1) ^ ((sr>>14)&1) ^ ((sr>>12)&1) ^ ((sr>>3)&1) ));
DAC = sr && 128; // prendre un bit de sr par la fonction AND avec 2^n.
```

Bruit rose ou rouge

Une particularité du signal PRBS est que la fréquence d'échantillonnage F est totalement absente du spectre. De ce fait, l'abaissement de cette dernière ne crée pas d'artefacts d'horloge audibles. On met à profit cette propriété en « lançant » des pièces à F , $F/3$, $F/9$ et $F/27$ et en additionnant les signaux. Ce faisant, on augmente le contenu en BF nécessaire pour produire du bruit rose ou rouge.

Réduction de la charge de calcul

Cette façon de produire du bruit rose ou rouge entraîne un pic de charge de travail si, à certains moments, deux, trois, voire quatre pièces doivent être lancées en même temps. Pour y remédier, une priorité à la pièce la plus lente est instaurée : seule la plus lente est lancée, les autres sautent un battement. En pratique, cela ne fait aucune différence audible.

Créer des motifs pour le mode *pattern*

C'est facile, car il suffit de raccourcir la chaîne en réinitialisant le registre à décalage à une valeur de départ mise à jour par le LFO.

alimenté par une batterie rechargeable de 9 V NiMH (et non Li-Po !) connectée à JP3 car cela la recharge à travers JP5. Dans tous les autres cas, mieux vaut ne pas la monter.

Logiciel

Le programme ou moteur sonore du PRBSynth1 est assez simple. Il peut être divisé en quatre tâches fonctionnant en parallèle (fig. 4) :

- > Traitement des échantillons audio
- > Modulation et interface utilisateur
- > Contrôle du panorama
- > Traitement MIDI

La 1^{re} tâche, la plus importante, est commandée par le Timer 1 cadencé par la fréquence d'échantillonnage. À chaque tic, cette tâche produit un échantillon de bruit, le fait passer par le filtre et l'amplificateur, puis l'envoi au CN/A.

La 2^e tâche exécute le LFO, lit les potentiomètres et le bouton de déclenchement, et met à jour les paramètres modulés. Le Timer 0 la contrôle et gère également l'intensité des LED par le biais d'une MLI logicielle.

Le Timer 2 tourne seul et sert de CN/A MLI pour produire la tension de commande du panoramique stéréo. C'est la tâche 3.

La tâche 4 est une tâche de fond qui s'exécute en l'absence de prise en charge d'interruption. Elle traite les données MIDI entrantes. C'est possible parce que le MIDI est relativement lent, 310 μ s par message, et que l'UART a un tampon. L'utilisation de ce synthétiseur avec un séquenceur logiciel actif sur plusieurs



PRODUITS

- > **Arduino Uno**
www.elektor.fr/15877
- > **Joy-IT Nano V3**
www.elektor.fr/18615
- > **C. Valens, « Maîtrisez les microcontrôleurs à l'aide d'Arduino (3^e édition) »**
www.elektor.fr/18064





Figure 5. Le PRBSynth1 assemblé à l'intérieur d'une goulotte électrique en PVC. Pour garder le circuit imprimé petit et facile à intégrer, les LED et potentiomètres ne sont pas montés, mais câblés via des connecteurs. Une pile de 9 V alimente le circuit.

canaux MIDI n'a posé aucun problème. Notez que la *sortie* MIDI (logiciel non écrit à ce jour) est là pour faire jaillir des idées (extension future).

Synthèse

Le PRBSynth1 est un minisynthétiseur (fig. 5) offrant de nombreuses possibilités, nonobstant l'Arduino Pro Mini qui l'anime. Il peut être construit sur une plaque d'essai et se prête bien à l'expérimentation matérielle et logicielle. Tous les détails de sa réalisation et de sa programmation, ainsi que des exemples sonores, sont disponibles sur la page web du projet au labo [1]. Une vidéo est disponible sur YouTube [3]. Amusez-vous bien ! 

210003-04

Contributeurs

Idée et conception : **Raymond Schouten**
 Rédaction : **Clemens Valens**
 Mise en page : **Harmen Heida**
 Traduction : **Yves Georges**

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (rs.elc.projects@gmail.com) ou contactez Elektor (redaction@elektor.fr).

Implémentation MIDI

Nous nous sommes efforcés d'améliorer la musicalité du PRBSynth1 s'il est connecté à un clavier MIDI, de préférence avec molettes de vélocité, hauteur et modulation. Les huit potentiomètres peuvent être neutralisés par un contrôleur continu MIDI CC. On reprend le contrôle en tournant le potentiomètre correspondant du synthé. Pour une version du synthétiseur uniquement MIDI, il faut donc supprimer la partie du programme qui lit les potentiomètres. Sans cette suppression, les entrées « en l'air » liront des valeurs fluctuantes qui prendront le relais des commandes MIDI. La version du programme appelée `no_potm` a déjà cette partie désactivée.

Avec un clavier (ou un séquenceur) MIDI, la molette de modulation règle la fréquence de coupure du filtre. Le son est coupé quand aucune touche n'est enfoncée. Ce mode s'active dès l'entrée de la première note MIDI et est réinitialisé à la mise sous tension.

La vélocité de la note (c.-à-d. la vitesse d'enfoncement de la touche) est convertie en profondeur de modulation du filtre du LFO.

En mode *noise*, le son n'a pas de hauteur, et le clavier contrôle donc la vitesse du LFO à la place. Une octave vers le haut double la vitesse. Cela permet de produire facilement des motifs rythmiques lorsque le LFO est en mode continu. Si le LFO est en mode « déclenché », on peut produire des sons enveloppés longs/courts en jouant une touche basse/haute.

En mode *pattern*, le clavier contrôle la longueur du motif. Ceci a été astucieusement conçu pour créer une échelle musicale (par ex., une octave vers le haut divise la longueur par deux) et permet de jouer des mélodies. Dans ce mode, la vitesse du LFO est commandée par le potentiomètre correspondant.

LIENS

- [1] PRBSynth1, page Elektor Labs : www.elektormagazine.fr/labs/synthesizer-prbsynth1
- [2] Site web de l'auteur : <http://www.rs-elc.nl>
- [3] Vidéo de démonstration : <https://youtu.be/agUzP0t1k7Y>