Pavé Numérique MIDI

Nous allons voir comment utiliser un pavé numérique pour faire de la musique, avec les bons programmes on peut même en faire une guitare électrique !

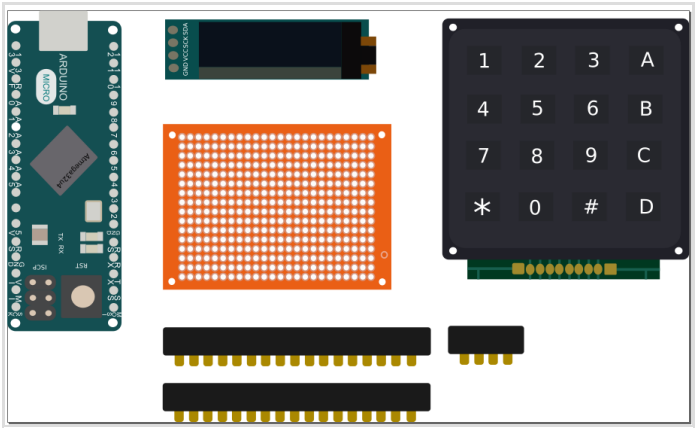
Difficulté **Moyen** | Durée **5 heure(s)** | Catégories **Électronique, Musique & Sons**

Coût **8 EUR (€)**

Sommaire

Introduction

Afin d'apprendre à créer des instruments MIDI (c.a.d des instruments utilisables dans des logiciels de musique), je me suis entraîné à l'aide de simples boutons.
Puis je me suis dit que je pourrais aussi directement utiliser un pavé numérique (keypad).
En soi ça m'a paru une bonne idée, car cela permet d'avoir 16 boutons facilement.
Mais les pavés numériques **ne sont pas pensés pour pouvoir être utilisés avec plusieurs boutons en même temps** donc si cela est important pour vous, vous êtes prévenus.



Matériaux

- Arduino micro (n'importe quelle carte avec un atmega32u4 peut être utilisée)
- Afficheur OLED I2C 128x32
- Pavé numérique (keypad) 16 Touches
- Une stripboard
- Des headers femelles
- Du câble 26 AWG
- Des vis (facultatif)

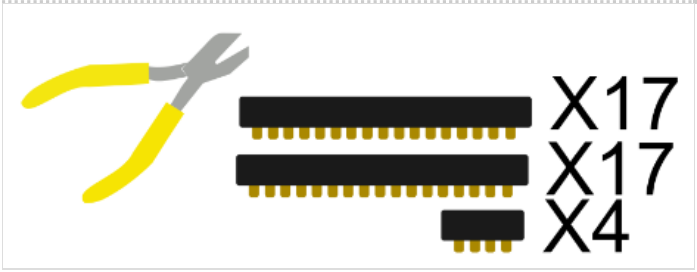
Outils

- Un fer à souder / étain
- Cable 26 AWG Monobrin (j'aime bien ces câbles qui sont très fins)
- Un dremel (pour couper la breadboard, fait un trou pour les vis)
- Une pince coupante

https://github.com/maditnerd/keypad_midi

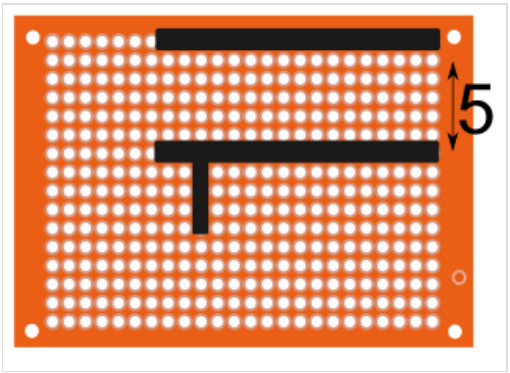
Étape 1 - Couper les headers femelles

Tout d'abord il nous faut couper deux headers femelles de 17 broches (pour l'arduino) et une de 4 broches (pour l'écran OLED)
Voici une méthode pour le faire: https://www.youtube.com/watch?v=qDG3VFSMSPQ



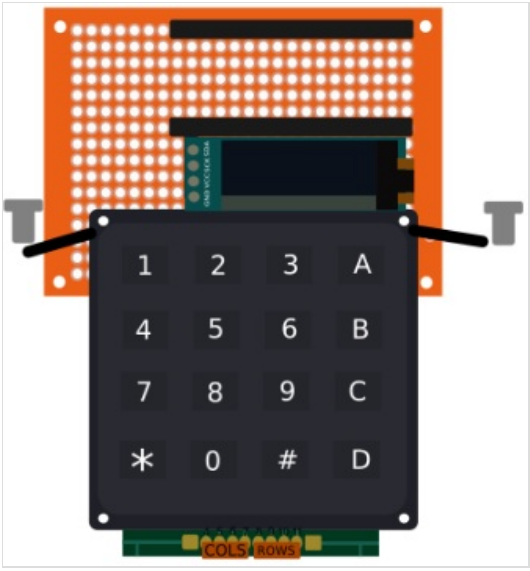
Étape 2 - Soudure des headers

Souder les headers sur la stripboard, vous pouvez vous aider de l'arduino micro pour ne pas vous tromper sur l'espacement (de 5 cases entre les deux)
Personnellement, je n'ai pas utilisé de headers pour l'afficheur OLED, car j'avais retiré les broches.



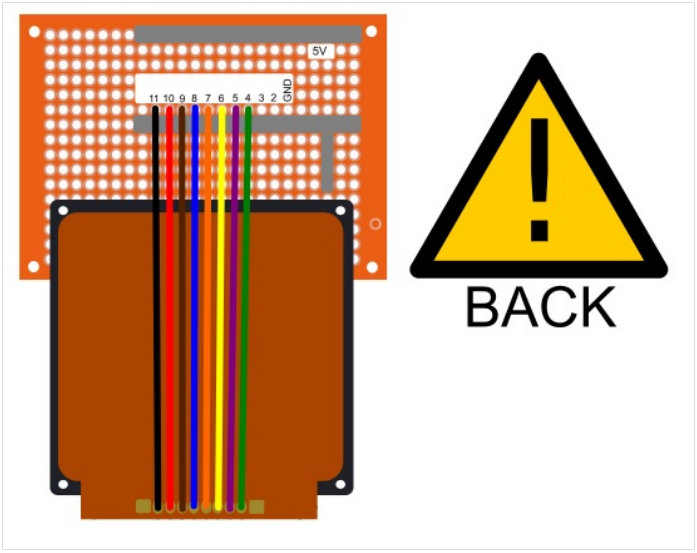
Étape 3 - Placement des composants

Vous pouvez soit coller le keypad (avec un pistolet à colle à chaud) où soit le visser en faisant des trous sur la stripboard.
Il vaut mieux fixer le keypad avant de souder les câbles pour pouvoir plus facilement vérifier la longueur des câbles.



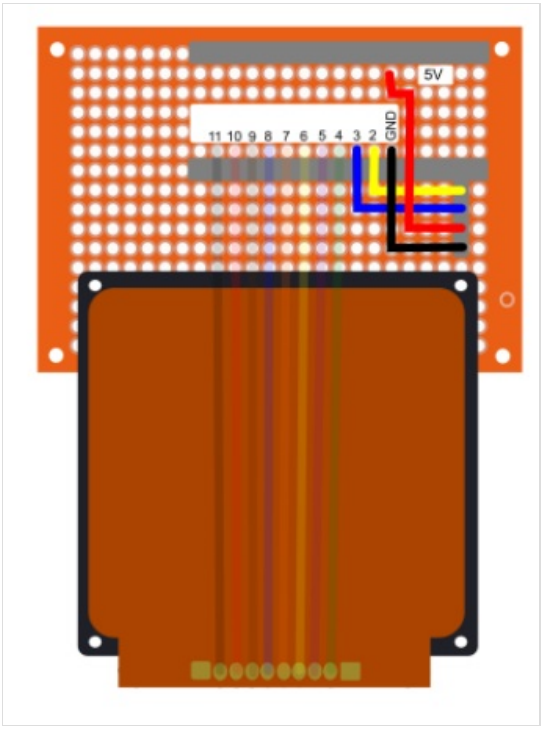
Étape 4 - Soudure du Keypad

Bon passons au vif du sujet, voilà comment souder le keypad.
Faites très attention à ne pas vous tromper en regarder votre arduino, tout est inversé.
Je vous conseille de partir de la broche 11, et de faire les suivantes.



Étape 5 - Soudure de l'afficheur OLED

Voici comment relier notre écran OLED, cet écran est relié en I2C (un protocole de communication) et utilise donc les broches 3 et 2 (c'est toujours le cas sur les cartes qui utilise un atmega32u4)
Noter que le GND (la terre) et le 5v sont sur la même colonne.



Étape 6 - Ajout des bibliothèques

Maintenant que notre keypad midi est prêt, il nous reste à programmer notre arduino micro, mon code a été fait d'une traite donc je ne garantis pas que tout marche correctement.
Le code est disponible ici : https://github.com/maditnerd/keypad_midi
Le code utilise 3 bibliothèques qu'ils nous faut installer.
Voici un petit rappel de comment gérer les bibliothèques.
https://www.arduino.cc/en/Guide/Libraries

- **MidiUSB** de Gary Grewal (Pour que l'arduino soit reconnu comme un instrument midi)
- **U8g2** de olikraus (Pour l'écran Oled)
- **Keypad** de NullKraft (Pour le keypad)

MidiUSB et U8g2 s'installe depuis le gestionnaire de bibliothèque d'Arduino.
Pour **Keypad c'est plus complexe**, il s'agit d'une version modifié de la bibliothèque Keypad de Mark Stanley, Alexander Brevig.
C'est la seul qui est capable de gérer l'appui de 4 touches en simultanée.
L'appui simultanée marche plus où moins, vu que mon objectif est surtout de faire un instrument monophonique (c.a.d une note à la fois) ça suffit mais si vous voulez jouer des accords, il y a aura un léger décalage dans le son.
Pour installer Keypad, il faut

- Aller sur le github : https://github.com/Nullkraft/Keypad
- Cliquer sur le bouton vert **Clone/Download**
- Extracter l'archive **Keypad-master.zip** dans votre dossier libraries de votre sketchbook, vous devriez avoir un dossier **Keypad-Master** dans **libraries**

Étape 7 - Informations sur le programme

J'ai essayé de commenter au mieux le code, donc je ne vais pas trop rentrer dans les détails, n'hésitez pas à me demander si vous voulez plus d'informations sur une partie en particulier.
Compromis sur la mémoire
J'ai dû faire des compromis dans le programme, principalement faute de l'écran.
En effet la bibliothèque U8g2 consomme beaucoup de mémoire dynamique (utiliser par les variables), j'aurais pu utiliser U8x8 ou la bibliothèque d'Adafruit mais malheureusement les lettres sur l'écran sont soit trop grosses soit trop petites.
J'ai donc dû limiter le nombre de preset à 8, il est surement possible d'augmenter le nombre de preset en ne personnalisant pas la vélocité par ex.

Étape 8 - Modifier les presets (notes/velocité)

Le plus important est le début du programme, c'est ici que l'on va définir nos notes et la vélocité de celle-ci (vélocité est le volume des notes)

Nom des preset

Le nom des preset est stocké dans un tableau

String preset_name[PRESET] = {"Du Riechst So Gut", "FF - Replica", "DrumKit", "A", "4", "5", "6", "B"};

Notes

Les notes sont stockées dans un tableau à deux dimensions, les notes sont dans l'ordre des boutons (la première est le 1 , la deuxième le 2, etc.)

int notes[PRESET][NB_BUTTONS] = {
{38, 38, 39, 39, 43, 46, 38, 38, 48, 46, 45, 45, 57, 60, 60, 57}, //Preset 1
{37, 40, 42, 39, 40, 38, 41, 40, 48, 46, 45, 45, 57, 60, 60, 57}, // Preset 2
{35, 35, 38, 42, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 55, 57}, // Preset 3
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset A
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset 4
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset 5
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset 6
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0} //Preset B
};

Les notes sont stockées en tant que notes midi (elles vont de 0 à 127) voici un tableau de correspondances avec la notation anglaise. Si vous êtes plus habitués au DO,RE,MI … voici la correspondance des notes

https://bassandbeatbox.com/lire-la-musique-avec-la-notation-americaine/

Octave	Note Numbers											
	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
-2	0	1	2	3	4	5	6	7	8	9	10	11
-1	12	13	14	15	16	17	18	19	20	21	22	23
0	24	25	26	27	28	29	30	31	32	33	34	35
1	36	37	38	39	40	41	42	43	44	45	46	47
2	48	49	50	51	52	53	54	55	56	57	58	59
3	60	61	62	63	64	65	66	67	68	69	70	71
4	72	73	74	75	76	77	78	79	80	81	82	83
5	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107
7	108	109	110	111	112	113	114	115	116	117	118	119
8	120	121	122	123	124	125	126	127				

Vélocité

Même chose pour la vélocité (pareil de 0 à 127) (

int velocity[PRESET][NB_BUTTONS] = {
{70, 70, 70, 70, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100}, //Preset 1
{70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70}, //Preset 2
{100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100}, //Preset 3
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset A
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset 4
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset 5
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset 6
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, //Preset B
};

Changeur de vélocité

Prominy Metal-V permet de gérer si les cordes sur une guitare sont étouffées ou pas selon que la vélocité soit inférieure à 70 ou supérieure à 100.

J'ai donc créé un système pour que quand D est appuyé la vélocité monte à 100.

Un tableau de booléen permet d'activer ou non cette fonctionnalité.

const int velocity_button = 15; // The D key is used to change velocity if velocity changer is true
const int VELOCITY_CHANGE = 100; //Velocity when D is pressed (should probably be changeable)
bool velocity_changer[PRESET] = {false, true, false, false, false, false, false, false};

Étape 9 - Transformer le keypad en guitare électrique !

Cette partie nécessite des connaissances en MAO (musique assisté par ordinateur)
Alors évidemment pas de miracle, afin d'obtenir un son convaincant de guitare, j'ai utilisé des plugins payants.
Pour "hoster" les plugins VST (des plugins pour générer/modifier) du son, j'ai utilisé Reason (https://www.propellerheads.com/fr/reason)
Vous pouvez évidemment utiliser n'importe quel logiciel capable de gérer des plugins VST.
Voici les plugins que j'ai utilisé pour la guitare:

- Native Instrument Kontakt : https://www.native-instruments.com/fr/products/komplete/samplers/kontakt-6/ Ce programme permet d'utiliser notre plugin guitare.
- Prominy V-Metal http://prominy.com/V_METAL.htm Le plugin de la guitare
- Guitar Rig 5 https://www.native-instruments.com/fr/products/komplete/guitar/guitar-rig-5-pro/ Le plugin pour les effets de guitare

J'ai combiné deux guitares afin d'avoir un son plus lourd.
Une qui joue note par note (no legato) et une qui joue des accords (power chords)
Pour les effets j'ai utilisé le preset **Sheder Solo** que j'ai un peu modifié.
Je réfléchis à faire une version light qui pourrait fonctionner avec LMMS https://lmms.io/ ou samplerbox http://www.samplerbox.org/ en utilisant des samples, si ça vous intéresse, parlez m'en en MP sur twitter (https://twitter.com/m4dnerd).
(Ca sera probablement pas aussi bien par contre, réver pas)
Voilà pour un rapide tour de mon rig, si vous avez jeter un coup d'oeil au prix des plugins, ça vous a probablement refroidi un peu!