Hacker votre adaptateur ADB/USB

Jean THOMAS

1 Introduction

L'adaptateur ADB/USB utilise le firmware open source tmk_keyboard. Il est donc possible de bidouiller de le firmware de votre adaptateur pour changer le mapping du clavier pour l'adapter à votre utilisation. Nous verrons dans ce guide comment flasher un nouveau firmware sur votre adaptateur ADB/USB, et comment modifier le firmware tmk_keyboard pour adapter le mapping du clavier à votre convenance.

Il est important de noter que ce document ne s'adresse qu'aux utilisateurs de la révision 0x1. La révision de votre adaptateur est indiquée au dos du circuit imprimé.

2 Flasher l'adaptateur

2.1 Le connecteur ISP

Le connecteur ISP est un connecteur qui permet de programmer un microcontrôleur sans devoir l'extraire physiquement. Sur votre convertisseur, le connecteur ISP se situe à droite du connecteur ADB.

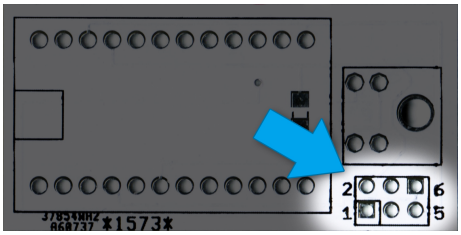

FIGURE 1 – Le connecteur ISP sur le circuit imprimé

Pour pouvoir l'utiliser, il faut souder un connecteur male type HE14. Ce connecteur se trouve à bas prix en magasin d'électronique. Vous pouvez

utiliser au choix deux connecteurs 3 broches simple rangée, côte à côte, ou alors utiliser un connecteur double rangée de 6 broches.

2.2 Le programmateur AVR

Un programmateur AVR est un dispositif qui joue le rôle d'interface entre notre ordinateur et notre convertisseur ADB/USB, pour pouvoir programmer ce dernier. Il existe plusieurs types de programmateurs AVR. Dans ce guide je traiterai du programmateur USBasp. Il est économique, simple à utiliser, et surtout compatible avec Mac OS X.

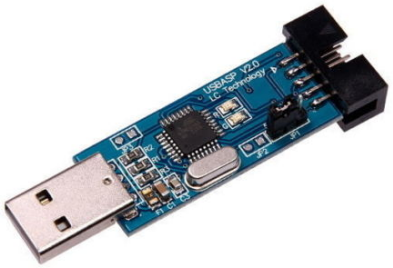

FIGURE 2 – Le programmateur USBasp

Ce programmateur est doté d'un connecteur ISP 10 broches. Pour des raisons de place, il n'est pas possible d'en intégrer un sur les convertisseurs ADB/USB. Nous utiliserons alors un adaptateur passif ISP 10 broches vers ISP 6 broches, qui peut être fabriqué facilement si on dispose des connecteurs adéquats, ou sinon peut être acheté à très bas prix.

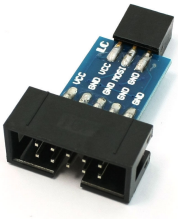

FIGURE 3 – L'adaptateur ISP 10 broches vers 6 broches

N'oubliez pas de respecter le sens de branchement sur le côté ISP 6 broches (l'autre côté étant doté d'un détrompeur) : la patte GND de l'adaptateur correspond à la broche 6 de l'adaptateur ADB/USB.

2.3 avrdude

Pour utiliser le programmateur sous Mac OS X, nous allons utiliser un logiciel nommé "avrdude", qui peut être installé grâce à homebrew (`http://brew.sh`).

On l'installe à l'aide d'une simple commande dans le terminal :

```
brew install avrdude
```

Une fois avrdude installé, nous pouvons l'utiliser pour flasher le firmware sur l'adaptateur :

```
avrdude -c usbasp -p m32u4 -U flash:w:monfirmware.hex
```

Ici, le firmware flashé est nommé **monfirmware.hex**.

3 Compiler tmk_keyboard

3.1 Installer le toolchain

Pour compiler le firmware, nous allons avoir besoin de deux outils : Git, qui nous servira à récupérer le firmware, et le toolchain, qui est l'ensemble des outils nécessaires à la compilation. L'installation se fait automatiquement dans le terminal grâce à homebrew, en tapant deux commandes à la suite :

```
brew tap larsimmisch/avr
brew install avr-libc git
```

3.2 Récupérer le firmware

Le firmware tmk_keyboard est hébergé sur Github, pour le télécharger nous allons utiliser Git dans le terminal :

```
git clone https://github.com/tmk/tmk_keyboard
```

Ensuite il faut se déplacer jusqu'au dossier contenant les sources du firmware du convertisseur ADB/USB :

```
cd tmk_keyboard/converter/adb_usb/
```

3.3 La compilation

La compilation s'effectue avec la commande **make** :

```
make KEYMAP=iso
```

Je précise **KEYMAP=iso** car l'adaptateur est adapté aux claviers européens. Si vous ne précisez pas le mapping au Makefile, le firmware sera adapté pour les claviers ANSI (américains). Vous aurez alors un fichier nommé **adb_usb_lufa.hex**, qui contiendra le firmware compilé. Vous pouvez à présent le flasher à l'aide d'avrdude :

```
avrdude -c usbasp -p m32u4 -U flash:w:adb_usb_lufa.hex
```

3.4 Modifier l'agencement des touches

Pour modifier l'agencement des touches, nous allons devoir toucher au code source de tmk_keyboard, en particulier au fichier **keymap_iso.c** (situé dans /**tmk_keyboard**/**converter**/**adb_usb**/) qui contient le mapping du clavier en ISO.

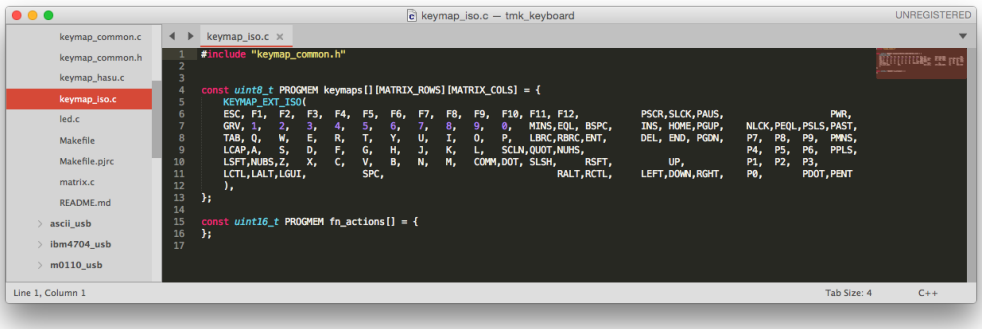

FIGURE 4 – Le fichier keymap_iso.c par défaut

Chaque touche physique du clavier est assignée à un code (par exemple **LCTL** pour la touche contrôle gauche). Il est possible de modifier le code d'une touche, pour lui assigner une autre fonctionnalité.

Par exemple, si je veux assigner la touche shift droite à la fonction espace, je devrai remplacer **RSFT** par **SPC**.

Il me suffira ensuite de re-compiler le firmware à l'aide des instructions ci-dessus, et de flasher mon firmware personnalisé.

Il existe quelques codes intéressants, par exemple ceux des touches multimédias :

```
AUDIO_MUTE : coupe le son
AUDIO_VOL_UP : augmente le volume
AUDIO_VOL_DOWN : baisse le volume
MEDIA_NEXT_TRACK : joue le média suivante
MEDIA_PREV_TRACK : joue le média précédent
MEDIA_STOP : arrête toute lecture de média
MEDIA_PLAY_PAUSE : met en pause le média
MEDIA_EJECT : éjection
```

Une liste détaillée de tous les codes intéressants se trouve dans le fichier **keycode.h** (situé dans /**tmk_keyboard**/**common**/). N'oubliez pas d'enlever le préfixe **KC_** pour utiliser le code de touche dans votre **keymap_iso.c** !