sur cette plaque, se trouve le connecteur de l'émetteur, fabriqué à partir d'un support de circuit intégré qui a été découpé.

Logiciel

J'ai suivi le code GPIO de l'exemple trouvé sur http://www.elinux.org/Rpi_Low-level_peripherals. Comme celui-ci accède directement à la mémoire, il est nécessaire de l'exécuter en tant qu'utilisateur root. Pour me faciliter la vie, je me suis logué en root pour le tester et le développer. Toutes les commandes données ici impliquent que vous ferez de même.

Tout se passe en ligne de commande, donc, à moins que votre système ne soit configuré pour démarrer directement l'interface graphique, il suffira de rester simplement en mode texte.

Pour créer un répertoire dans lequel stocker le code source et le fichier exécutable pendant la durée du développement, tapez les commandes suivantes :

```
$ mkdir gpio
$ cd gpio
```

Mon code source peut être téléchargé depuis http://www.hoagieshouse.com. Suivez simplement les liens et enregistrez le fichier dans le répertoire que vous venez juste de créer. Il prend 2 paramètres, le canal et on ou off. Vous devrez éditer le code pour remplacer mes codes de télécommande par les vôtres. Pour éditer le code, tapez :

```
$ nano switch.cpp
```

Soyez attentif à conserver les guillemets autour des codes dans le source. Après les avoir modifiés, quittez en appuyant sur <CTRL>-<X>, répondez Y à la demande d'enregistrement du fichier et acceptez le nom de fichier switch.cpp. Pour construire le fichier exécutable, tapez :

```
$ g++ -o switch switch.cpp
```

Testez-le avec vos prises en lançant la commande :

```
$ ./switch 1 on
```

Si cela fonctionne, vous souhaiterez sans doute pouvoir le lancer avec n'importe quel utilisateur. Utilisez les commandes suivantes pour le faire :

```
$ chmod +s switch
$ mv switch /usr/bin/
```

Pour planifier l'allumage et l'extinction des prises à des moments précis, vous pouvez utiliser quelque chose appelée des tâches cron. Tapez simplement :

```
$ crontab –e
```

Vous serez alors en mesure d'éditer un fichier qui contrôle les tâches planifiées. Le format est décrit dans le fichier, mais pour faire un test, ajoutez ces lignes en fin de fichier :

```
0 *  * * * switch 1 on
10 *  * * * switch 1 off
```

Cela va allumer la prise 1 pendant dix minutes au début de chaque heure.

Jusque-là, ce n'est pas très convivial. Une interface web serait beaucoup mieux. La mienne permet d'allumer ou éteindre les 4 canaux, mais je ne suis pas allé assez loin pour y ajouter la planification. Premièrement, installez mini-httpd pour agir en tant que serveur web. Pour cela, tapez :

```
$ apt-get install mini-httpd
```

Les fichiers de l'interface web peuvent être téléchargés depuis http://www.hoagieshouse.com. Dans le fichier zip se trouvent le fichier de configuration pour mini-httpd et le répertoire /var/www dans lequel j'ai placé les pages web et les programmes cgi. Aucune modification ne devrait être nécessaire, mais il faut les placer aux bons emplacements. Le fichier HTML utilise une requête AJAX pour lancer le script CGI avec les paramètres de canal et on/off. Ce script récupère simplement les paramètres à partir de la requête et appelle le programme switch avec.

Conclusion

Maintenant que vous avez les bases logicielles pour contrôler à distance plusieurs appareils branchés sur secteur avec votre Raspberry Pi, je vous laisse comme exercice d'imaginer quels autres déclencheurs pourraient être reliés au GPIO pour allumer ou éteindre des appareils. Oh, si vous attrapez le Père Noël, dites-lui s'il vous plaît que j'aimerais l'ensemble Noël du MagPi imprimé trouvé sur htt://www.kickstarter.com/projects/themagpi/the-magpi-magazine-from-virtual-to-reality.

Liste des composants Maplin
VY48C - TX/RX 433MHz £9.99
M1M - Résistance 1M Ohm £0.29
DG41U - Câble pour lecteur de disquettes £3.99

Article de Geoff Johnson