Table of Contents

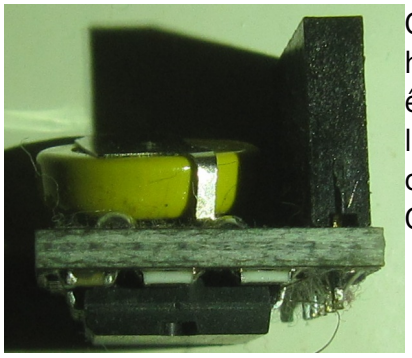
DS3231 : ajouter une horloge temps réel I2C au Raspberry pi 1
DS3231 RTC : le module 1
Connexion au Raspberry pi 1
Branchement direct 1
Branchement avec des câbles 2
Configuration de l'I2C 2
Utilisation du module 4
Définir la date du système depuis internet et l'enregistrer dans le DS3231 4
Définir la date du système manuellement et l'enregistrer dans le DS3231 4
Configuration du système pour utiliser automatiquement le module RTC DS3231 5
Conclusions 5
Lectures complémentaires 6

DS3231 : ajouter une horloge temps réel I2C au Raspberry pi

Le Raspberry pi ne dispose pas de module RTC (Real Time Clock, horloge en temps réel), et ne peut donc pas garder une trace précise du temps écoulé sans avoir recours à une synchronisation sur un serveur de temps (NTP). Cela n'est pas toujours possible, notamment pour des projets ou le Raspberry Pi n'est pas connecté au réseau. Pour remédier à cela, il est possible d'ajouter un module RTC tel que le DS3231, économique, compact et précis. Nous verrons dans ce tutoriel comment réaliser cela.

Ce tutoriel est également l'occasion de tester les versions vendues sur Aliexpress de ce produit.

DS3231 RTC : le module



Ce module est très compact, puisqu'il fait 13.6*13.6mm pour une hauteur de 11.4mm. Il dispose de 5 broches femelle, de façon à pouvoir être branché facilement, puisqu'il suffit de l'enfoncer sur les GPIO dans le bon sens. Nous verrons toutefois comment le connecter via des câbles, car cela évite de bloquer l'accès au bus I2C, au +3.3v et à un GPIO.

Le module dispose d'une petite pile bouton qui lui permet de garder une trace du temps écoulé même quand le Raspberry pi n'est pas alimenté.

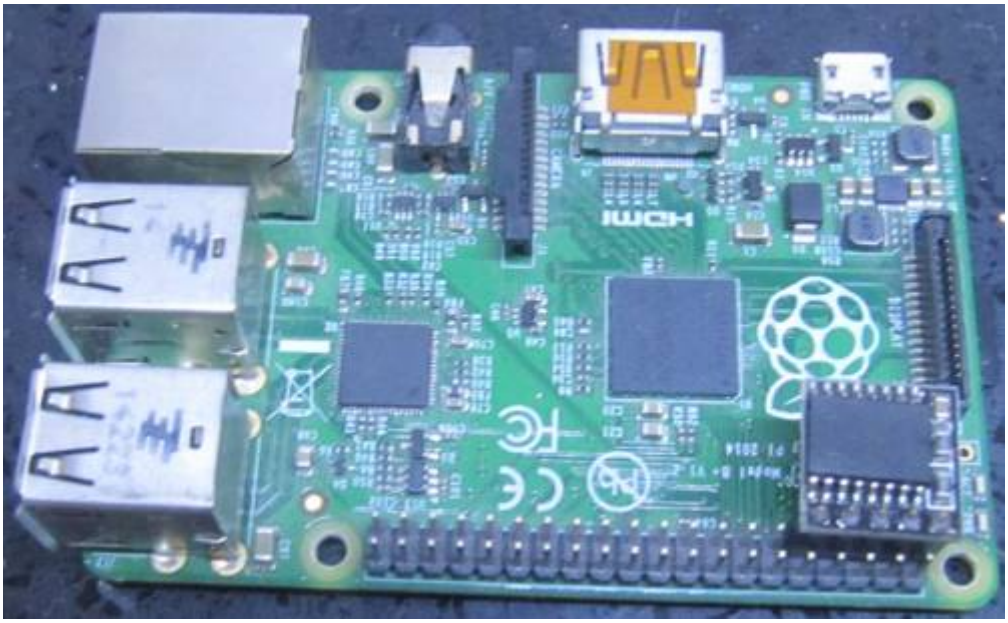
Connexion au Raspberry pi

Pour connecter le module, nous aurons deux solutions : soit nous le brancherons directement sur les GPIO (il est fait pour ça), ce qui en bloquera donc certains, ou alors nous utiliserons des câbles mâle-femelle pour connecter les deux. Nous verrons les deux méthodes.

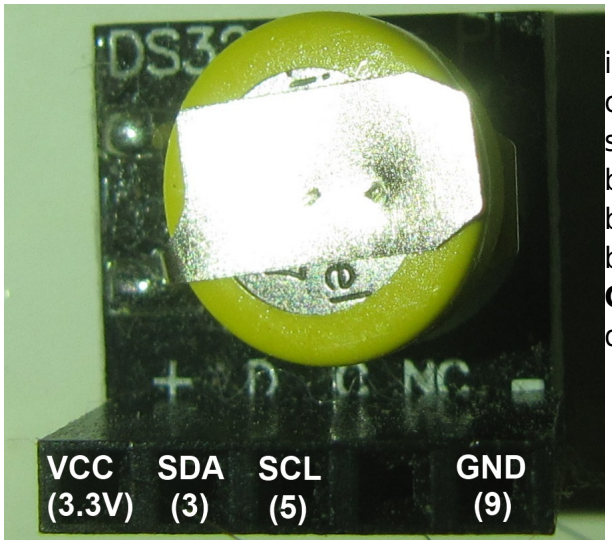
Branchement direct

Pour brancher directement le module, prenez votre Raspberry pi (peu importe le modèle), et placez le face à vous, de sorte que la carte SD/micro SD soit vers le haut, et le port hdmi vers la gauche. Prenez également le module, de façon à voir la puce. Le connecteur devra être vertical, et du côté droit. Vous pourrez alors enficher le module sur la rangée de gauche des GPIO, tout en haut (les GPIO 1,3,5,7 et 9 seront couverts par le module). Le module dépassera alors vers le milieu du Raspberry pi, près du logo.

La photo ci dessous permettent de voir le sens d'installation:



Branchement avec des câbles



Si vous retournez votre module, vous verrez des indications près du connecteur femelle. La broche **+** sera connectée au GPIO1, à savoir le **+3.3V**. La broche **D** ira sur le **GPIO3**, à savoir la broche SDA du bus I2C. La broche **C** ira sur le **GPIO 5**, à savoir la broche **SCL** du bus **I2C**. La broche **NC** ne sera connectée à rien, et la broche **–** ira à la **masse** du Raspberry pi, par exemple le **GPIO 9**. N'oubliez pas que sur la double rangée de GPIO, on compte ligne par ligne :

1 2 3 4 5 6 etc.

Si vous avez besoin d'une référence, voici le schéma des GPIO, si on tient le Raspberry pi comme décrit plus haut, et sur la photo du Raspberry pi 2.

Configuration de l'I2C

Ce module fonctionne en I2C, il est donc nécessaire de l'activer et de le faire fonctionner. Pour cela, le plus simple est d'exécuter l'utilitaire **raspi-config** :

```
sudo raspi-config
```

Ensuite, sélectionnez l'option **7 – Advanced configuration**, puis l'option **A7 I2C**, et choisissez **YES**,

OK, **YES**, et encore **OK**. Vous pouvez alors faire **finish**, puis accepter de redémarrer.

Pour vérifier le fonctionnement de l'ensemble, il faudra installer les programmes **python-smbus** et **i2c-tools** :

```
sudo apt-get install python-smbus
sudo apt-get install i2c-tools
```

Une fois cela effectué, vous pourrez vérifier que tout fonctionne en tapant la commande suivante :

```
sudo i2cdetect -y 1
```

Les premiers modèles de Raspberry pi (très anciens) devront recevoir la commande **sudo i2cdetect -y 0** à la place.

Si cela ne fonctionne pas, vous pouvez essayer de charger les modules **i2c-dev** et **i2c-bcm2708** à la main :

```
sudo modprobe i2c-dev
sudo modprobe i2c-bcm2708
```

Si cela fonctionne, vous pourrez alors ajouter ces modules dans /etc/modules en exécutant la commande suivante :

```
sudo nano /etc/modules
```

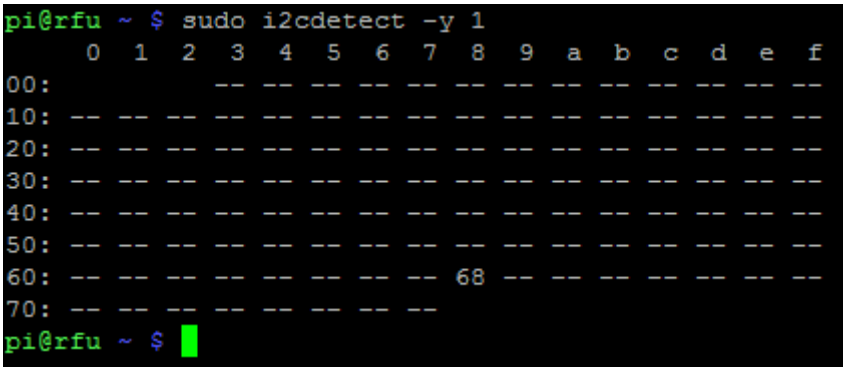
Vous pourrez alors ajouter à la fin du fichier les deux lignes suivantes :

```
i2c-dev
i2c-bcm2708
```

Au prochain redémarrage, les modules seront automatiquement chargés, et il ne sera pas nécessaire de faire un **modprobe**.

Si cela ne fonctionne toujours pas, voici une piste à explorer, en anglais, mais les instructions sont claires (un fichier à modifier).

Si tout à bien fonctionné et que le câblage est bon, vous devriez voir ceci en faisant un sudo i2cdetect -y 1 :



Utilisation du module

Notre module est détecté, et utilisable. Cependant, nous ne savons pas encore ce qu'il “raconte”. Nous chercherons donc à communiquer avec le module.

Si tout ce que nous avons fait avant a bien fonctionné, nous pouvons alors exécuter la commande suivante :

```
echo ds3231 0x68 | sudo tee /sys/class/i2c-adapter/i2c-1/new_device
```

Cette commande doit “notifier” au système la présence du composant. Dès lors il est possible de consulter l'heure et la date contenue dans le module en faisant un :

```
sudo hwclock
```

Cela devrait dès lors afficher une date et une heure. Si vous n'avez jamais utilisé le module, celle ci sera probablement incorrecte. On pourra alors la régler.

Définir la date du système depuis internet et l'enregistrer dans le DS3231

Avant tout assurons nous que les données du système sont correctes, en vérifiant la “*timezone*” (fuseau horaire) :

```
sudo dpkg-reconfigure tzdata
```

Choisissez bien le bon continent et la bonne zone. Vous pouvez vérifier la date et l'heure du système via la commande **date**. Lorsque celle ci est définie correctement, vous pourrez alors écrire la valeur actuelle du système en utilisant la commande suivante :

```
sudo hwclock -w
```

Définir la date du système manuellement et l'enregistrer dans le DS3231

Si votre Raspberry pi n'a pas accès à internet pour se synchroniser à une horloge précise, vous pouvez toujours utiliser la commande suivante pour définir une date arbitraire (ici le 31 Aout 2015, à 13h15:00s) :

```
sudo date -s "Aug 31 2015 13:15:00"
```

Vous pourrez alors utiliser sudo hwclock -w pour définir la date et l'heure sur le module RTC à partir de cette heure définie pour le système. Notez bien que la commande précédente définit la date et l'heure du système au moment de la commande, mais que dès lors la valeur évolue normalement. Il n'est donc pas obligatoire d'effectuer les deux commandes à la suite de façon très rapide. En revanche si vous éteignez/redémarrez le Raspberry pi, il est possible que la valeur qu'il aie en mémoire ne soit plus correcte.

Configuration du système pour utiliser automatiquement le module RTC DS3231

Nous avons donc pu définir la date et l'heure du DS3231. Voyons maintenant comment configurer le Raspberry pi pour qu'il utilise ce module à chaque démarrage pour définir l'heure.

En effet, si nous redémarrons maintenant, il faudra refaire "**echo ds3231 0x68 > /sys/class/i2c-adapter/i2c-1/new_device**" avant de pouvoir utiliser le module (sinon le **sudo hwclock** ne fonctionne pas).

Pour circonvenir à ce problème, nous allons modifier le fichier **/etc/rc.local** via la commande suivante :

```
sudo nano /etc/rc.local
```

Nous ajouterons à ce fichier les deux lignes suivantes avant la ligne contenant "exit 0" :

```
echo ds3231 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
hwclock -s
```

On pourra alors désactiver le service **fake-hwclock** qui tente de reproduire le fonctionnement d'une horloge temps réel quand on en dispose pas, via la commande suivante :

```
sudo update-rc.d fake-hwclock disable
```

On pourra également dans ce contexte désactiver le service ntp, qui synchronise l'heure locale avec celle d'un serveur de temps :

```
sudo update-rc.d ntp disable
```

Si par la suite on souhaite synchroniser la date et l'heure du système avec celle de serveurs de temps, on pourra utiliser la commande suivante :

```
sudo ntpd -qg
```

Il est alors toujours possible d'écrire la valeur obtenue dans l'horloge temps réel en utilisant la commande **sudo hwclock -w**.

Conclusions

Ce module compact permet donc d'ajouter une horloge temps réel au Raspberry pi à bas coût. Elle est relativement précise, avec une déviation maximale d'environ 1 seconde par semaine, soit 1 minute par an. Il s'agit du pire des cas, car la déviation peut aller dans les deux sens, donc bien souvent des déviations successives s'annuleront. Ce module est significativement plus précis que le DS1307 qui fut très populaire. Toutefois vu les prix de ces DS3231, autant s'en servir.

Le module que j'ai utilisé provient de chine, et jusqu'ici je n'ai eu aucun soucis avec. Dans le cas contraire, je signalerai toute défaillance sur ce blog.

Ce module permet donc au Raspberry pi de garder une trace précise du temps, ce qui pourra servir dans un projet embarqué (comme ce Raspberry pi mobile, sur batterie par exemple) sans accès à internet. Par exemple, pour faire des photos en pleine nature, ou prendre des mesures. Dans tous les cas, on disposera d'une date et d'une heure précise pour dater les photos ou mesures.

Si toutefois cela ne suffisait pas, il faudrait s'orienter vers des systèmes plus complexes, tels que les horloges atomiques, par exemple celles au rubidium, mais ce sont des appareils plus gros et consommateurs d'énergie, qui serviront pour des mesures ultra précises.

Lectures complémentaires

Voici quelques liens si vous souhaitez approfondir le sujet :

- Un tutoriel en Anglais sur l'installation du DS3231;
- un autre tutoriel an Anglais sur le même sujet, (les commentaires parlent de certains problèmes potentiels);
- encore un tutoriel en anglais, avec une grosse discussion dans les commentaires, à voir en cas de problème;
- la documentation du service fake-hwclock;
- configuration de l'I2C (tutoriel en Anglais, Adafruit);
- la manpage de la commande hwclock.