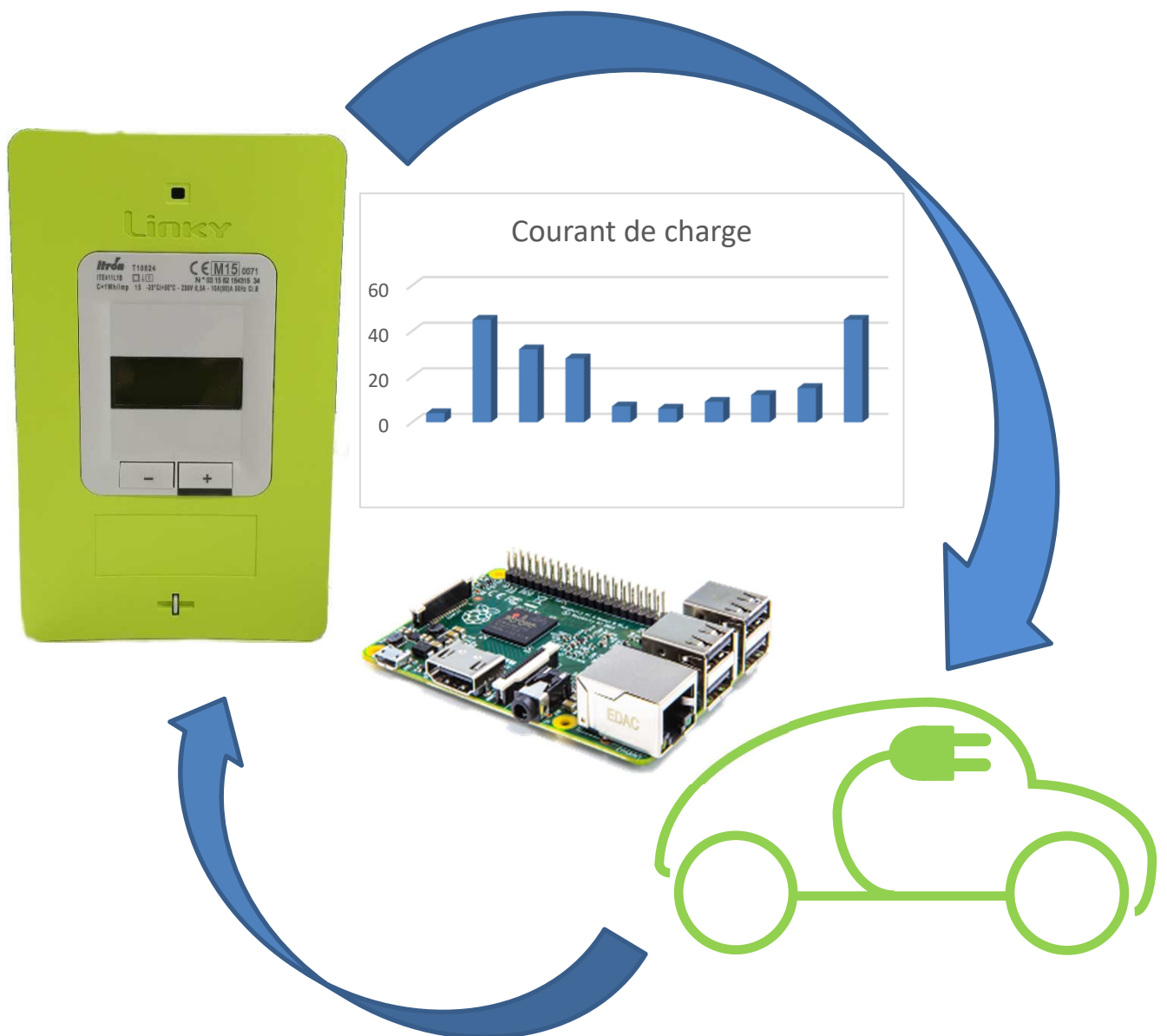


Délestage dynamique d'une IRVE



Nicolas Pinard

Table des matières

Mise en situation :	3
Expression du besoin :	3
Cahier des charges :	3
Fonctionnalités de l'EPC.....	3
Réglage manuel	4
Etude de la TIC	7
Potentiomètre numérique.....	13
Convertisseur numérique-analogique (CNA)	16
Validation de la chaine de régulation	19
Conclusion	20

Mise en situation :

Un de vos clients, après avoir fait l'acquisition d'une voiture électrique vous a demandé de lui installer une borne de recharge à son domicile.

[Vous avez donc, après recherche, assemblé et installé une borne de recharge monophasée d'une puissance de 7kW.](#)

Expression du besoin :

La borne de recharge fonctionne très bien. Cependant, il arrive que l'installation de votre client disjoncte lors des charges. Vous soupçonnez une consommation électrique trop importante par rapport à l'abonnement électrique de votre client qui est de 9kVA monophasé.

Cahier des charges :

Vous allez devoir mettre en place un système permettant de modifier la puissance de charge de la borne.

Celui-ci pourra être commandé par l'utilisateur mais il devra également pouvoir se réguler par rapport à la consommation instantanée du domicile. La régulation automatique se fera à l'aide d'un raspberry Pi 3.

Fonctionnalités de l'EPC

Rappeler la référence de l'EPC que vous avez installé dans la borne du client

Reprendre la documentation

Indiquer s'il existe un moyen de limiter le courant à une valeur inférieure à 32A et expliquer comment

Réglage manuel

Indiquer ce qui se passe si l'on place un dipôle résistif d'une valeur inférieure à 100 ohms

Relever dans le tableau les valeurs minimale et maximale des dipôles résistifs utilisables dans ce cas (vous indiquerez les courants correspondants)

Indiquer quel type de composant pourrait permettre de balayer toutes les valeurs résistives

Expliquer en quoi l'utilisation de ce composant risque d'interrompre la charge

Indiquer comment l'on pourrait éviter ce phénomène

Choisir une valeur pour ce dipôle résistif dans la série E48 pour être sûr de ne pas tomber sous les 6A

Calculer la valeur théorique pour le potentiomètre ajustable afin de pouvoir atteindre les 32A

Indiquer la valeur courante que vous devriez choisir pour ce potentiomètre

Expliquer l'inconvénient de ce choix

Proposer une autre valeur pour le potentiomètre au plus près de ce que vous avez calculé en théorie

Calculer la valeur du dipôle résistif à y adjoindre

Choisir une valeur normalisée (série E48) pour ce dipôle résistif pour être sûr d'atteindre les 32A

Indiquer dans ce cas la valeur minimale du courant que nous pourrons régler

Expliquer si cela pose problème

Expliquer si cela pose problème que la valeur résistive en butée haute, dépasse ce qui est prévu pour 32A

Proposer une procédure de test pour vérifier vos calculs

Proposer un schéma de câblage pour valider votre proposition

Indiquer les matériels dont vous aurez besoin pour réaliser le test en pratique

Demander le matériel au professeur

Câbler votre maquette

Indiquer les résultats obtenus

Conclure sur votre test

Etude de la TIC

En vous aidant du document « Enedis-NOI-CPT_54E.pdf »

Indiquer ce que signifie le sigle TIC

Indiquer quels sont les deux modes de fonctionnement de la TIC (paragraphe 3.3)

Indiquer quel mode est initialisé sur les compteurs Linky en sortie d'usine

Indiquer sur quelles bornes est disponible le bus télé-information (paragraphe 5.1 et 8.2)

En vous aidant du document « ERDF-NOI-CPT_O2E.pdf »

Donner les caractéristiques générales de la transmission

Pour pouvoir exploiter les informations fournies par la TIC nous nous servirons d'un raspberry Pi.

Le raspberry Pi n'est pas capable d'utiliser directement les signaux de la TIC, il va falloir une interface.

Chercher et donner une référence d'une interface de télé-information

Il existe de nombreuses interfaces, certaines disposent d'une sortie série, USB ou encore WIFI.

Pour notre part, n'ayant pas de compteur à notre disposition nous utiliserons un programme « SimulateurTIC » qui nous délivrera les informations telles qu'elles seraient en sortie d'une interface télé-information série. Les informations seront disponibles sur la sortie série d'un raspberry Pi.

Demander au professeur un Raspberry Pi3, une carte μ SD et son adaptateur μ SD - SD

Insérer la carte dans l'ordinateur

A l'aide du logiciel « Raspberry PI imager » écrire l'image du système d'exploitation sur la carte avec les paramètres suivants :

- PI OS lite
- Hostname : Bornerecharge
- Enable SSH
- Username : pi
- Password : Azerty1234
- Local setting : Europe/Paris
- Keyboard layout : fr

Insérer la carte dans le raspberry, le brancher électriquement et connecter-le au réseau

Déterminer son adresse IP à l'aide d'Advanced IP scanner

Connectez-vous en SSH à l'aide de Putty

Faire les mises à jour et activer le port série à l'aide de « raspi-config »

Vous allez maintenant devoir paramétrer le port série pour utiliser l'UART réel¹

Ajouter à **/boot/config.txt** : **dtoverlay = pi3-disable-bt**

Supprimer **console=serial0,115200** dans **cmdline.txt** (si nécessaire)

Redémarrer le raspberry Pi

Entrer la commande « ls -l /dev/ »

Indiquer si vous obtenez le résultat suivant concernant les lignes « serial » (si ce n'est pas le cas il faut corriger le problème, appeler le professeur)

```
lrwxrwxrwx 1 root root      7 Jul  1 18:45 serial0 -> ttyAMA0
lrwxrwxrwx 1 root root      5 Jul  1 18:45 serial1 -> ttyS0
```

¹ <https://www.framboise314.fr/le-port-serie-du-raspberry-pi-3-pas-simple/>

Pour la bonne exécution du programme « SimulateurTIC.py » il faut installer le paquet « pyserial » permettant de prendre en charge les ports série en python. L'installation se fait en utilisant l'installateur de paquet « PIP² »

Entrer les commandes suivantes

- sudo apt install pip
- sudo pip install pyserial

Pour tester le simulateur de TIC nous devons connecter celui-ci à un dispositif capable de décoder les informations envoyées. Nous savons que le dispositif qui devra réguler la charge et décoder la télé-information sera à base de raspberry Pi 3. Nous pourrions donc connecter le raspberry « simulateur de TIC » à un raspberry « régulation de la charge ». C'est ce que nous allons faire mais plutôt que d'utiliser deux raspberry nous allons réaliser l'opération sur un seul. En effet le simulateur de TIC ne se sert que de la sortie série (la télé-information est unidirectionnelle) et le régulateur de charge, pour la même raison n'utilisera que l'entrée.

Nous allons donc boucler la sortie sur l'entrée.

Donner les numéros de broches physiques ainsi que les numéros des GPIO pour la sortie et l'entrée série en vous aidant de l'annexe 1

Afin de protéger les broches d'entrée et de sortie en cas de mauvaise manipulation, nous allons insérer un dipôle résistif entre les deux.

Demander un dipôle résistif d'1 kilo ohm au professeur

Proposer un schéma de câblage sur l'annexe 2

Faire valider et réaliser le câblage

² [https://fr.wikipedia.org/wiki/Pip_\(gestionnaire_de_paquets\)](https://fr.wikipedia.org/wiki/Pip_(gestionnaire_de_paquets))

Récupérer le fichier « SimulateurTIC.py » sur le NAS

Téléverser le fichier dans le dossier « /home/pi/ » du raspberry en utilisant le protocole TFTP et le logiciel « FileZilla »

Lancer l'exécution du programme « python SimulateurTIC.py » sur le raspberry (vous choisirez une valeur pour le courant mais ne demanderez l'envoi que d'une trame)

Expliquer ce qui s'est passé, ce que vous avez observé

Indiquer si vous avez utilisé un programme pour récupérer sur le port série d'entrée les informations envoyées sur le port série de sortie

Installer le programme picocom « sudo apt install picocom »

Ouvrir une autre fenêtre de terminal

Lancer le programme picocom

Indiquer le résultat

Le programme a besoin de connaître les paramètres de la trame que l'on veut décoder.

C'est-à-dire le nom du port série, la vitesse en baud, le nombre de bits de données, la parité, le nombre de bits de stop.

Indiquer à l'aide du document « ERDF-NOI-CPT_O2E.pdf »

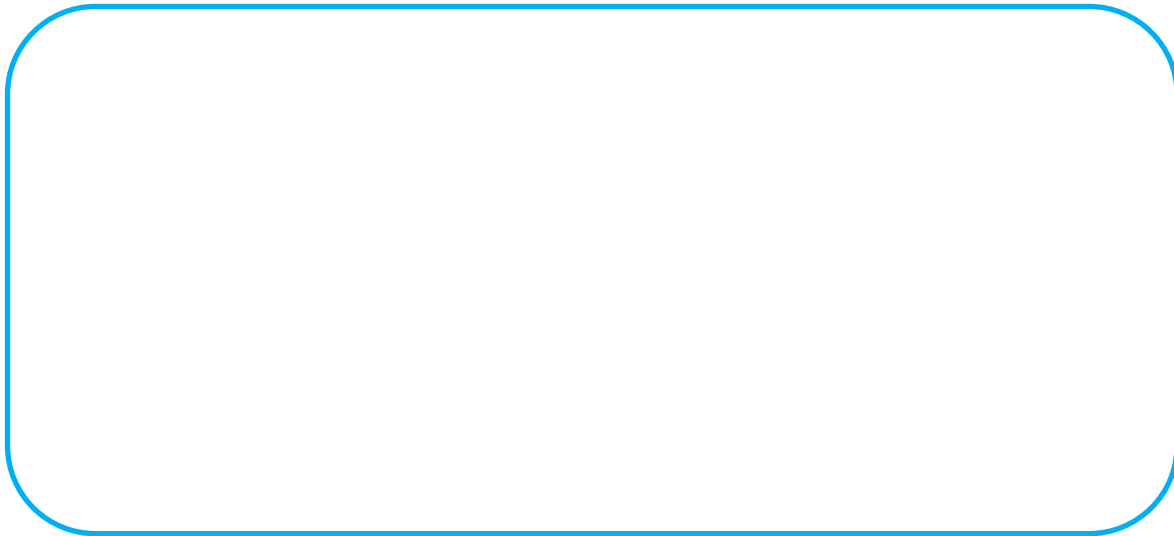
Compléter la ligne de commande à l'aide de l'Annexe 3 (odd : impair even : pair)

```
picocom /dev/ttyAMA0 -b ..... -y ..... -d ..... -p .....
```

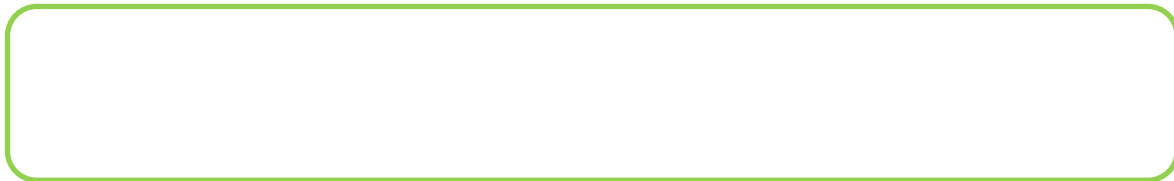
Lancer le programme picocom avec les paramètres nécessaires

Lancer le programme SimulateurTIC dans l'autre fenêtre de terminal

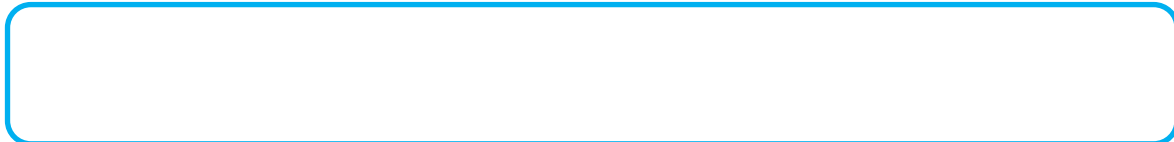
Indiquer ce que vous avez reçu



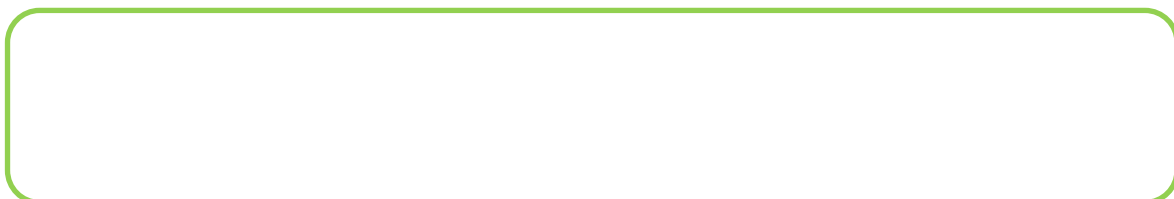
Expliquer ce que sont les informations ISOUSC et IINST



Indiquer le risque couru si IINST dépasse durablement ISOUSC



Expliquer en quoi ces deux informations peuvent nous aider à réguler la charge



Expliquer comment vous pourriez imposer le courant de charge disponible par l'EPC (2 réponses)

Indiquer quel type de composant nous pouvons utiliser pour faire varier une valeur de résistance à partir d'un système numérique Raspberry Pi 3

Potentiomètre numérique

On propose d'utiliser un AD8400A1

Justifier ce choix

Indiquer le nombre de valeurs de résistances disponibles

Indiquer le type de liaison série utilisée par ce composant

Indiquer si l'on pourra alimenter ce composant à l'aide du Raspberry Pi

Calculer le quantum (la valeur de résistance correspondant à 1 bit)

Proposer des valeurs décimales (correspondants à un codage sur 8 bits) pour valider le choix du composant AD8400

Décimale(s)							
Ohm(s) (calcul)							
Ohm(s) (relevé)							

Compléter la ligne « calcul » du tableau

Proposer un schéma de câblage du raspberry Pi 3 et de l'AD8400 sur l'annexe 4 (vous placerez judicieusement un ohmmètre pour relever les valeurs obtenues. Vous utiliserez la sortie spi CEO du raspberry pi 3)

Demander au professeur de vérifier votre schéma.

Indiquer, à l'aide de la documentation, le type de boîtier « package » disponible pour les caractéristiques suivantes :

- AD8400
- 8 broches (1 seul potentiomètre)
- 1k Ω

Ce type de boîtier est un boîtier CMS.

Donner la signification de l'acronyme CMS

L'AD8400 qui vous est confié est monté sur un adaptateur CMS-DIP. Vous pouvez donc l'utiliser sur les plaques lab.

Réaliser le câblage (hors tension !)

Demander au professeur de vérifier

Alimenter le raspberry

Téléverser le programme « potentiometre.py »

Lancer l'exécution du programme

Compléter la ligne « relevé » du tableau

Donner deux raisons qui peuvent expliquer les écarts entre le calcul et le relevé

Justifier, d'après vos relevés, si le composant AD8400A1 correspond à notre besoin

Convertisseur numérique-analogique (CNA)

Relever, dans la documentation de l'EPC, les tensions à appliquer pour des valeurs de courant de 6 et 32A.

On propose d'utiliser un MCP4921

Relever la plage de tension d'alimentation du composant

Indiquer si le raspberry Pi 3 pourra l'alimenter

Indiquer sur combien de bits fonctionne le MCP4921

Calculer le nombre de combinaisons possibles

Indiquer la(es) valeur(s) que l'on peut appliquer sur Vref

Calculer le quantum dans le cas où l'on applique Vref = 3,3 Volts

Calculer le quantum dans le cas où l'on applique Vref = 3,3 Volts

Expliquer laquelle de ces valeurs vous allez choisir

Indiquer le type de liaison série utilisé par ce composant

Proposer un schéma de câblage du raspberry Pi 3 et du MCP4921 sur l'annexe 5 (vous placerez judicieusement un ohmmètre pour relever les valeurs obtenues. Vous utiliserez la sortie spi CEO du raspberry pi 3)

Demander au professeur de vérifier votre schéma.

Proposer des valeurs décimales (correspondants à un codage sur 12 bits) pour valider le choix du composant AD8400

Décimale(s)							
Vout (calcul)							
Vout (relevé)							

Calculer la valeur théorique de Vout et remplir la ligne « calcul » du tableau

Réaliser le câblage (hors tension !)

Demander au professeur de vérifier

Alimenter le raspberry

Téléverser le programme « cna.py »

Lancer l'exécution du programme

Compléter la ligne « relevé » du tableau

Justifier, d'après vos relevés, si le composant MCP4921 correspond à notre besoin

Les deux composants que nous venons de tester correspondent à notre besoin, mais la précision de la valeur de sortie par rapport à la consigne est meilleure sur CNA, nous choisirons donc ce composant.

Validation de la chaîne de régulation

En plus des outils et programmes dont vous disposez déjà, on vous fournit un programme appelé « Borederecharge.py » capable de récupérer les informations sur l'entrée série (borne 10).

Le programme est, en outre, chargé de calculer la consigne à envoyer au CNA pour que la tension à sa sortie permette de réguler correctement la puissance de charge.

Proposer une méthode permettant de vérifier si, en fonction des informations reçues par la TIC, on retrouve en sortie une valeur qui permet de réguler correctement la puissance de charge.



Si une modification du câblage est nécessaire :

- Modifier et/ou compléter le schéma de câblage
- Faites-la valider par le professeur
- Réaliser la modification (hors tension !)
- Effectuer la procédure de test que vous avez proposée

I_{inst} TIC (A)								
I_{charge} (A)								
V_{EPC} (V) tableau EPC								
V_{EPC} (V) relevé								

I_{inst} TIC : Valeur que l'on va envoyer par la TIC au programme de régulation

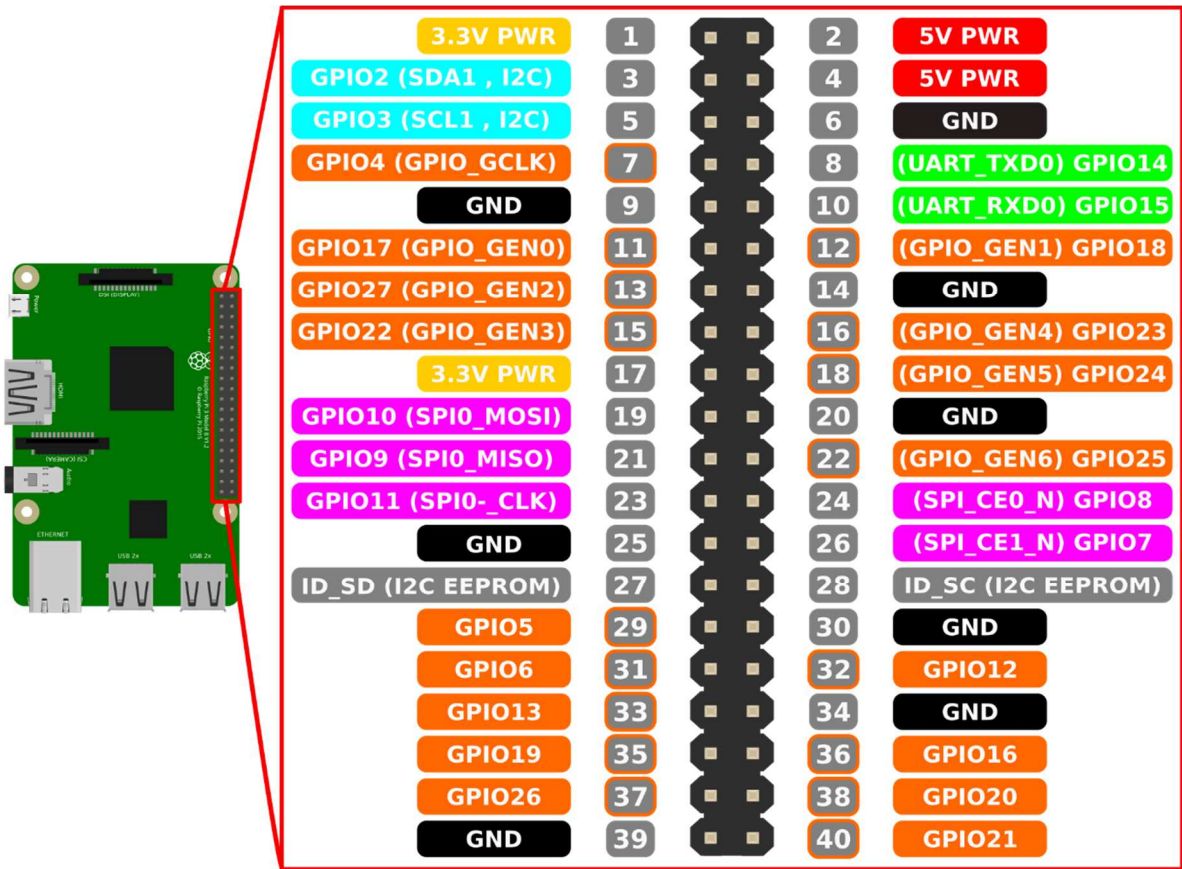
I_{charge} (A) : Valeur théorique à laquelle il faut baisser la puissance de charge pour ne pas dépasser 45A (abonnement)

V_{EPC} (V) _{tableau EPC} : Valeur issue du tableau de correspondances de la documentation de l'EPC, c'est la valeur de la tension à appliquer sur la borne Ic de l'EPC pour réguler la puissance à une valeur choisie

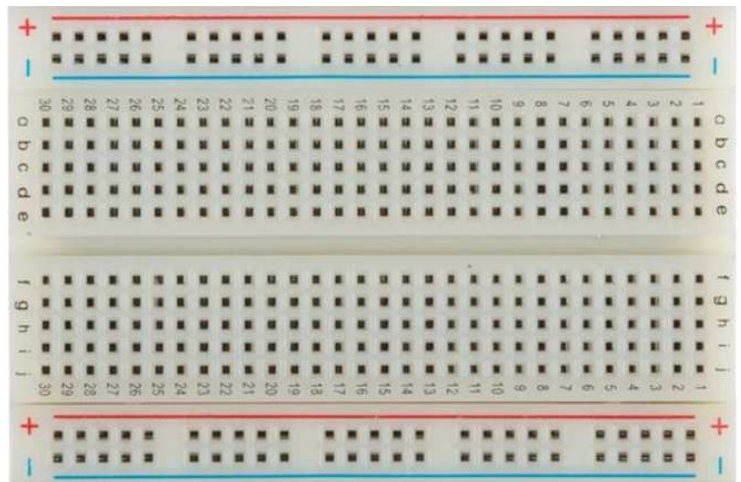
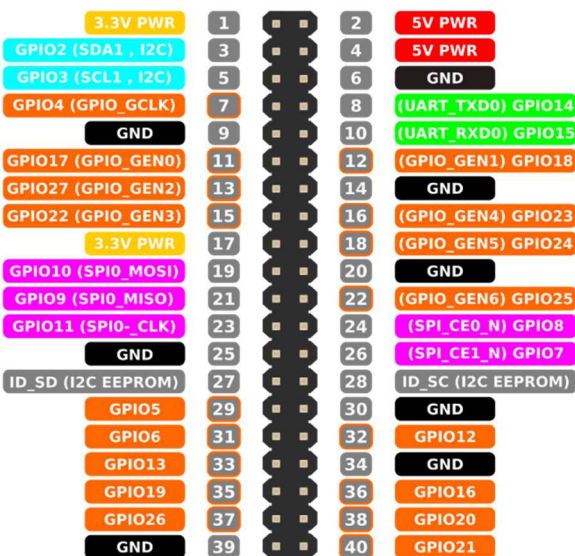
V_{EPC} (V) _{relevé} : Valeur relevée en sortie du convertisseur numérique analogique, cette sortie devra être reliée à la borne Ic de l'EPC.

Conclusion

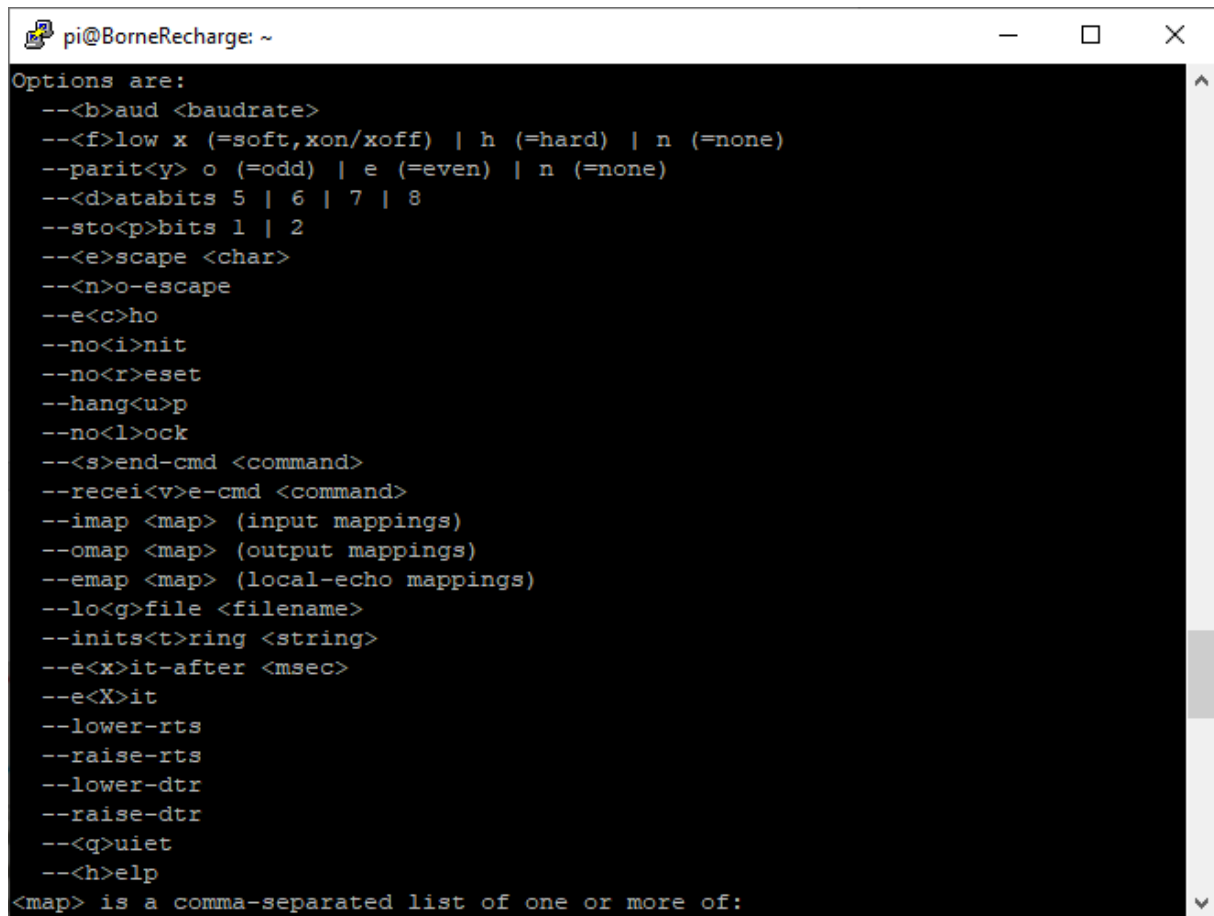
Annexe 1



Annexe 2



Annexe 3

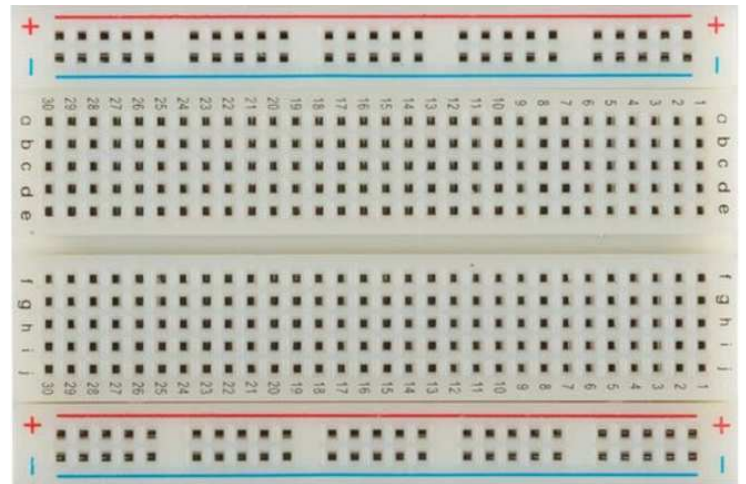


```
pi@BorneRecharge: ~
Options are:
--<b>aud <baudrate>
--<f>low x (=soft,xon/xoff) | h (=hard) | n (=none)
--parit<y> o (=odd) | e (=even) | n (=none)
--<d>atabits 5 | 6 | 7 | 8
--sto<p>bits 1 | 2
--<e>scape <char>
--<n>o-escape
--e<c>ho
--no<i>nit
--no<r>eset
--hang<u>p
--no<l>ock
--<s>end-cmd <command>
--recei<v>e-cmd <command>
--imap <map> (input mappings)
--omap <map> (output mappings)
--emap <map> (local-echo mappings)
--lo<g>file <filename>
--inits<t>ring <string>
--e<x>it-after <msec>
--e<X>it
--lower-rts
--raise-rts
--lower-dtr
--raise-dtr
--<q>uiet
--<h>elp
<map> is a comma-separated list of one or more of:
```

Annexe 4

(AD8400)

3.3V PWR	1	2	5V PWR
GPIO2 (SDA1 , I2C)	3	4	5V PWR
GPIO3 (SCL1 , I2C)	5	6	GND
GPIO4 (GPIO_GCLK)	7	8	(UART_TXD0) GPIO14
GND	9	10	(UART_RXD0) GPIO15
GPIO17 (GPIO_GEN0)	11	12	(GPIO_GEN1) GPIO18
GPIO27 (GPIO_GEN2)	13	14	GND
GPIO22 (GPIO_GEN3)	15	16	(GPIO_GEN4) GPIO23
3.3V PWR	17	18	(GPIO_GEN5) GPIO24
GPIO10 (SPI0_MOSI)	19	20	GND
GPIO9 (SPI0_MISO)	21	22	(GPIO_GEN6) GPIO25
GPIO11 (SPI0_CLK)	23	24	(SPI_CE0_N) GPIO8
GND	25	26	(SPI_CE1_N) GPIO7
ID_SD (I2C EEPROM)	27	28	ID_SC (I2C EEPROM)
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21



Annexe 5

(MCP4921)

3.3V PWR	1	2	5V PWR
GPIO2 (SDA1, I2C)	3	4	5V PWR
GPIO3 (SCL1, I2C)	5	6	GND
GPIO4 (GPIO_GCLK)	7	8	(UART_TXD0) GPIO14
GND	9	10	(UART_RXD0) GPIO15
GPIO17 (GPIO_GEN0)	11	12	(GPIO_GEN1) GPIO18
GPIO27 (GPIO_GEN2)	13	14	GND
GPIO22 (GPIO_GEN3)	15	16	(GPIO_GEN4) GPIO23
3.3V PWR	17	18	(GPIO_GEN5) GPIO24
GPIO10 (SPI0_MOSI)	19	20	GND
GPIO9 (SPI0_MISO)	21	22	(GPIO_GEN6) GPIO25
GPIO11 (SPI0_CLK)	23	24	(SPI_CE0_N) GPIO8
GND	25	26	(SPI_CE1_N) GPIO7
ID_SD (I2C EEPROM)	27	28	ID_SC (I2C EEPROM)
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

