Numéro 6 - OCT 2012

The MagPi™

Un magazine pour les utilisateurs de Raspberry Pi

Le Skutter a atterri

- Alimentation portable pour votre Pi
- Une citrouille différente
- Caméra Pi
- Gagnez un support LCD

http://www.themagpi.com

The MagPi

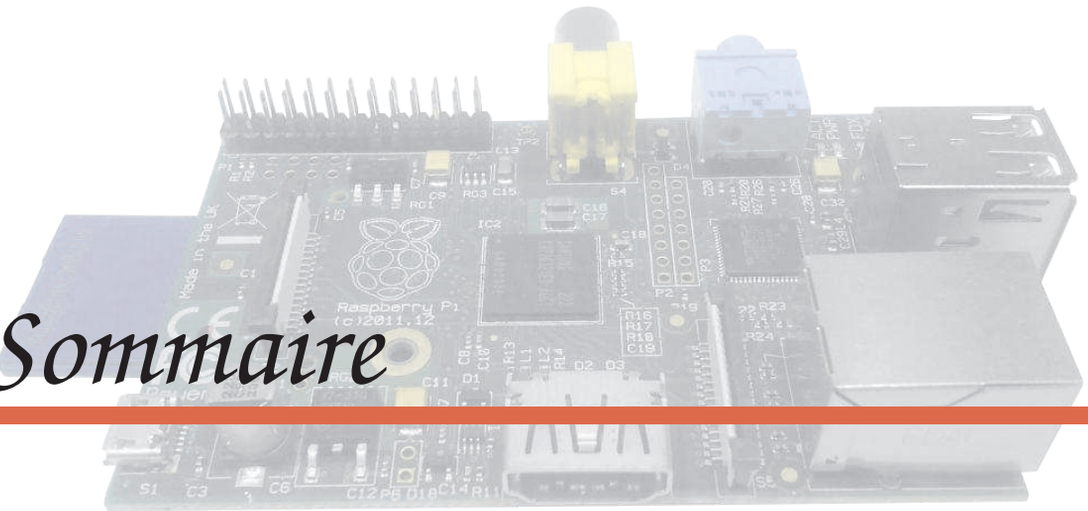
Bienvenue dans ce numéro 6,

En réponse à beaucoup de demandes de lecteurs, la série sur le robot Skutter est de retour avec un nouvel épisode passionnant. Pour faire bouger des projets de robot ou des expériences de terrain, il y a un article sur l'alimentation via des batteries. Le MagPi présente également une façon légèrement différente d'utiliser une citrouille, ainsi que des interviews et des sujets sur le développement.

Si vous ne l'avez pas fait, nous vous suggérons d'essayer le nouveau Raspbian turbo. Le gain en vitesse est vraiment visible.

Ash Stone
Rédacteur en chef du Magpi

Ash Stone
Rédacteur en chef / Administrateur / Titre
Jason 'Jaseman' Davies
Auteur / Site Web / Design des pages
Meltwater
Auteur / Photographie / Design des pages
Chris 'tzj' Stagg
Auteur / Photographie / Design des pages
Ian McAlpine
Design des pages / Graphisme
Joshua Marinacci
Design des pages / Graphisme
Lix
Design des pages / Graphisme
PaisleyBoy
Design des pages / Graphisme
Sam Marshall
Design des pages / Graphisme
Andrius Grigaliunas
Photographie
Matt '0the0judge0'
Administrateur / Site Web
Bodge N Hackit
Auteur
John Ellerington
Auteur
Gordon Henderson
Auteur
Colin Deady
Auteur / Design des pages
Spencer Organ
Auteur
Luke A. Guest
Auteur
W.H.Bell
Auteur
Colin Norris
Éditeur / Graphisme / Titre Caverne du C
Antiloquax
Auteur



Sommaire

Ajouter une base motorisée

Partie 1

DIFFICULTE: AVANCEE

Jusque là nous nous sommes concentrés sur le pilotage du bras du robot pour ce projet.

Le bras du robot est une pièce très chouette avec beaucoup de possibilités, mais je voudrais aussi parler d'un autre sujet qui pourrait donner accès à la robotique à plus de personnes, surtout si vous avez un budget limité et ne pouvez mettre la main sur ce type de matériel pour le moment.

Se déplacer

Tout robot a besoin d'un moyen pour se déplacer. Dans la plupart des cas, les robots utilisent une plateforme motorisée, mais il y a quelques exceptions.

Dans la première partie de cet article, je vais faire le tour des différentes plateformes mécaniques qu'on peut trouver.

Également, je jetterai un œil sur quelques idées "maison" et pour finir j'expliquerai ma propre solution pour ce problème de plateforme mécanique.

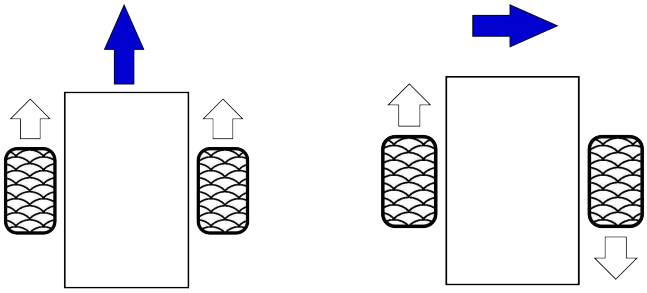
Commençons par regarder dans la nature. Il y a des insectes avec plusieurs pattes, des mammifères avec quatre (ou deux) et il y a des serpents qui n'ont rien du tout ! Il y a eu des développements fabuleux en robotique qui ont inclus ces formes de locomotion.

Il y a même à l'heure actuelle des robots nageurs qui se baladent avec les poissons pour chasser et identifier les vilains qui polluent nos océans !

Pour la plupart, les technos qui sont utilisées pour faire ce genre de robots sont hors de prix pour un amateur éclairé, sans parler du fait que c'est atrocement compliqué et très probablement hors de portée pour ceux d'entre vous qui voudraient faire un robot avec un budget serré.

L'alternative est de construire une plateforme motorisée avec des roues. Cela ramène tout de suite un robot sur terre en terme de prix et de difficulté. La première question à se poser est : quel type de plateforme ?

De base, il y a deux possibilités. La première est une plateforme "différentielle" qui marche sur le même principe qu'un char. Vous avez un moteur de chaque côté du robot, relié à une roue via quelques engrenages.

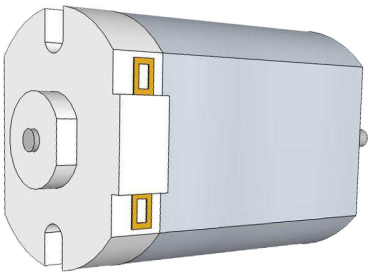


Quand les roues tournent dans la même direction, le robot avance dans cette direction. Si elles tournent en sens opposé, le robot va tourner.

La deuxième option est d'utiliser une direction de type "voiture", où un moteur unique fait avancer ou reculer le robot, et un second, un servomoteur par exemple, change l'angle des roues avant pour tourner. La plateforme différentielle a l'avantage de pouvoir tourner sur un espace plus réduit que la méthode direction de voiture.

Type de moteurs

Une autre considération est le type de moteur qui peut être utilisé. L'option la plus simple et la moins chère, c'est le moteur CC (moteur à courant continu, Direct Current Motor).



Il utilise une simple bobine (contenant des milliers de spires) de fil attaché à l'arbre moteur, appelée un "commutateur" et deux aimants opposés. Appliquer un courant électrique dans la bobine crée un champ magnétique. Les pôles de ce champ sont attirés par le champ opposé créé par chaque aimant et cela fait bouger l'arbre d'un demi-tour jusqu'à être au plus proche de l'aimant opposé. A ce moment, le commutateur est changé et cela inverse la polarité du champ magnétique. La bobine est donc repoussée du pôle auquel elle était attirée, et se déplace donc jusqu'au nouveau pôle opposé et le cycle redémarre encore. Plus de puissance rend le cycle plus rapide (jusqu'à un certain point où ça brûle et explose !). Inverser le courant fait tourner le moteur dans le sens inverse. C'est chouette et simple, mais le principal inconvénient est qu'il n'est pas possible de faire un déplacement précis.

Une autre option est d'utiliser ce qui s'appelle un moteur pas-à-pas. Au lieu d'avoir une simple bobine et un commutateur, un moteur pas-à-pas a plusieurs bobines et aimants. Chacun doit être allumé et éteint dans un certain ordre pour bouger. Cela veut dire que le moteur tourne avec plein de toutes petites "secousses". Vous pouvez spécifier avec une précision extraordinaire de combien vous voulez bouger votre moteur pas-à-pas, mais l'inconvénient c'est qu'ils sont plus compliqués à contrôler et qu'ils ont moins de couple (force de rotation) qu'un moteur CC.

Peu importe le moteur que vous choisissez, il est peu probable que vous puissiez connecter directement des roues et zou, ça roule tout de suite. Le moteur ira soit trop vite, ou bien n'aura pas assez de couple ou de "chevaux-vapeur". Pour le rendre utilisable, nous allons avoir besoin de passer par un système d'engrenages ou de poulies. Par exemple un moteur qui tourne à 10000 tours/minute peut être plutôt rapide, au moins. Au plus, attaché directement à un robot, il sera trop faible pour déplacer le moindre petit poids.

Une vitesse un peu plus raisonnable (mais encore correcte) pourrait être de l'ordre de 2500 RPM (Round Per Minute, Tours/Minute). Pour finir le boulot, on aura besoin d'utiliser des engrenages (ou "crémaillères") avec un rapport de 1 pour 4. Avec cela la vitesse sera diminuée d'un facteur 4 et le couple sera augmenté d'autant. Cela veut dire que l'engrenage attaché au moteur doit être quatre fois plus gros que celui attaché à la roue. Vous pouvez atteindre le même résultat avec une combinaison d'engrenages plus petits. Néanmoins cela ajoute un niveau de complexité en plus dans notre projet.

Il est possible d'acheter des moteurs électriques avec des jeux d'engrenages préfabriqués qui sont spécialement faits pour le marché de la robotique de loisir. Cela a pour avantage qu'une roue peut être directement connectée à l'essieu sur le moteur sans avoir à se soucier de faire son propre jeu d'engrenages. L'inconvénient est le prix plus élevé, souvent de l'ordre de 10 £ (une douzaine d'euros) ou plus pour un moteur.

L'acquisition d'un moteur

En lisant ça vous commencez peut-être à vous sentir un peu intimidé par la complexité et le coût de vous embarquer dans un projet comme celui-ci, mais ne vous en faites pas. Je vais vous montrer une autre méthode que vous pouvez utiliser pour construire une plateforme de robot. Vous pouvez tricher !

Il y a plein de jouets motorisés à prix raisonnable sur le marché, avec un peu de rafistolage, vous pouvez en faire une base de robot qui peut faire la compétition avec les meilleurs. Encore plus intéressant, vous n'avez pas à payer le plein prix pour quelque

chose comme ceci.

Dans presque tous les pays du monde il y a des décharges/vide-grenier qui vendent des pièces détachées, avec un peu de chance et de recherche, vous pourrez certainement trouvez un jouet faisant l'affaire que vous pourrez convertir en base de robot - c'est presque assuré qu'un jouet de la sorte va utiliser un moteur CC et vous avez plus de chances de trouver un jouet qui utilise la direction plutôt qu'un différentiel mais ces ventes recèlent de vrais trésors pour un amateur de robot plein de ressources.

J'ai utilisé cette méthode pour mon propre Skutter. J'ai acquis un jouet Big Trak, que vous pouvez acheter pour environ 20£ (vous pourriez même en trouver un dans un vide-grenier, si vous êtes chanceux).

Afin de vous montrer comment vous pouvez adapter des jouets dans une base de robots, je vais vous décrire comment j'ai adapté ce Big Trak pour mon Skutter.

Ajouter une platforme Big Trak

1. Enlever le pare-choc
Premièrement, démonter le Big Trak. La plupart des vis sont faciles d'accès. Toutefois, une est cachée derrière un pare-choc en plastique gris à l'arrière.

Pour atteindre cette vis, dévissez toutes les autres afin de pouvoir écarter les parties du Big Trak juste assez pour détacher le pare-choc de l'intérieur.

2. Enlever le clavier
Après avoir enlever le couvercle, détacher tous les accessoires tels que la tourelle en plastique etc. Quelques pièces seront difficiles mais éventuellement s'enlèveront avec un peu de va-et-vient.

Le clavier et le câble plat branchés à l'électronique dessous empêchent le couvercle d'être enlevé. Ce clavier est collé en place et peut être enlevé facilement pour permettre d'enlever le couvercle.

3. Débranchez les moteurs
Après avoir enlever le couvercle, dévissez le circuit dessous. Ceci révèlera un boîtier d'engrenages en plastique qui contient également 2 moteurs CC qui motorisent le Big Trak.

Coupez les fils qui sont attachés aux moteurs. Soudez 2 fils à chaque terminal + et - sur les 2 moteurs.

4. Vérifiez les moteurs
Maintenant vérifiez que les moteurs fonctionnent. Une pile LR20 suffira pour voir la base avancer, reculer et tourner selon la façon dont vous tenez les fils sur les pôles de la pile. En 2 mots, c'est tout ! Une plateforme motorisée pour un robot.

Pour contrôler les moteurs, un peu d'électronique est requise afin que les broches GPIO d'un Raspberry Pi puissent les commander pour les mettre en mouvement. (Un avertissement clair - en aucun cas vous ne devez brancher un moteur directement sur le Raspberry Pi. Si vous le faites, il va tenter de fournir une puissance trop importante qui, pour parler franchement, va le détruire. Je décrirai l'électronique permettant d'interfacer de façon sécurisée les moteurs avec les broches GPIO dans un autre article).

5. Installer le bras du robot
Plus de modifications au Big Trak sont requises afin d'installer le bras sur celui-ci. Le

trou de la tourelle sur le Big Trak est presque à l'endroit idéal et aux bonnes dimensions pour installer la base du bras à l'intérieur avec "l'épaule" dépassant de la base.

Pour un trou parfait, découpez autour du trou avec un cutter ou semblable et ensuite utilisez des pinces coupantes pour couper une partie du contour du trou de la tourelle.

Utilisez une lime ou du papier de verre pour adoucir les bords, ou encore le côté d'une boîte d'allumettes.

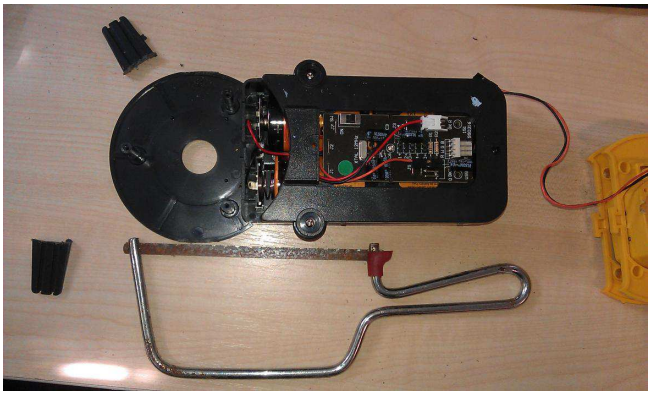
Un peu de travail sur le boîtier du Big Trak est nécessaire afin d'insérer la base du bras à l'intérieur.

Utilisez les pinces coupantes pour enlever la base de plastique du haut-parleur au fond du Big Trak.

6. Installer le bras

Pour finir, une modification à la base du bras robotisé est requise. Enlevez les stabilisateurs de la base avec une petite scie. (Je ne crois pas qu'ils soient vraiment nécessaires de toute façon).

Le bras robotisé devrait maintenant s'insérer facilement à l'intérieur du Big Trak.



Un peu de travail supplémentaire est nécessaire afin de s'assurer que le bras est bien fixé. Lorsque je l'installe pour un test je n'utilise que du ruban et des élastiques mais en temps voulu, j'utiliserai quelque chose de plus solide pour le fixer.

J'utiliserai probablement du gros ruban adhésif pour ça. Le Gaffer est mon outil de rafistolage préféré parce qu'il est solide mais s'enlève facilement au besoin. Cependant il y a plein d'autres méthodes que vous pourriez utiliser avec un peu de réflexion.

Le produit final comme vous pouvez le voir est un corps de Skutter.

La prochaine fois...

La prochaine étape est de concevoir l'électronique et le programme pour actionner la plateforme motorisée sous le contrôle du Raspberry Pi.

Article de Bodge N Hackitt

ALIMENTATION PORTABLE POUR VOTRE PI

Libérez votre Raspberry Pi grâce à une alimentation portable

Il existe beaucoup de projets intéressants qui obligent à détacher le Pi de son alimentation principale ou qui nécessitent une alimentation 5V capable de fournir plus de puissance que celle disponible sur les broches GPIO du Pi - voici un moyen d'y parvenir.

La principale exigence est d'avoir une tension régulée convenable de 5V, avec une intensité suffisante pour votre projet. J'ai tout d'abord regardé du côté du régulateur 7805, populaire et bon marché, mais je l'ai laissé car il n'est pas très efficace - un point essentiel pour une alimentation basée sur des piles - et il n'est capable de gérer que 1A - comme le Pi utilise jusqu'à 700 mA, cela ne laisse pas beaucoup de marge pour piloter autre chose.

Je me suis finalement décidé pour le régulateur de tension à découpage LM2576T-5.0 - ce composant est beaucoup plus efficace que le 7805, et est en mesure de gérer jusqu'à 3A. Il pourra recevoir n'importe quel voltage en entrée de 7 à 40 V CC, vous laissant un grand choix de batteries - j'utilise 8 piles NiMH de 1,2V, ce qui donne une alimentation de 9,6V, mais vous pouvez utiliser des batteries plomb-acide de 12V si cela vous convient davantage. En plus du régulateur lui-même, seuls 4 autres composants sont nécessaires - 2 condensateurs, une bobine d'arrêt et une diode Shottky.

Voici les codes commande RS :

1 x LM2576T-5.0	460-477
1 x condensateur électrolytique 100uF 25V	684-1942
1 x condensateur électrolytique 1000uF 25V	684-1951
1 x bobine d'arrêt 100uH (3A min)	228-416
1 x diode Shottky 40V 3A	714-6819
1 x plaque de prototypage	206-5879
1 x rouleau de ruban isolant	513-553
1 x dissipateur (optionnel, voir remarque)	189-9306

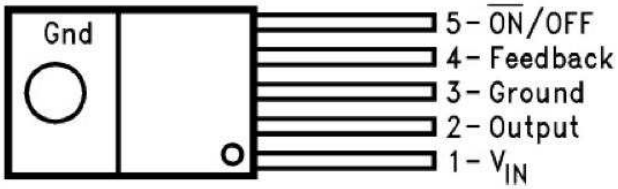
La fiche technique du régulateur est disponible ici

http://www.ti.com/lit/ds/symlink/lm2576.pdf

où vous trouverez également les détails sur le circuit comme indiqué sur le diagramme, plus loin dans l'article.

Et les connexions du circuit de régulation sont :

5-Lead TO-220 (T)
Top View



Le circuit est facilement construit avec un bout de plaque de prototypage, et aucune coupure de piste n'est nécessaire ; une diode peut être ajoutée pour servir de voyant - ajoutez une résistance de 200 ohm - et je recommande des fusibles de 3,25 A en série avec l'entrée positive et la sortie. Prenez garde à mettre les deux condensateurs et la diode dans le bon sens.

Voici à quoi ressemble le mien :

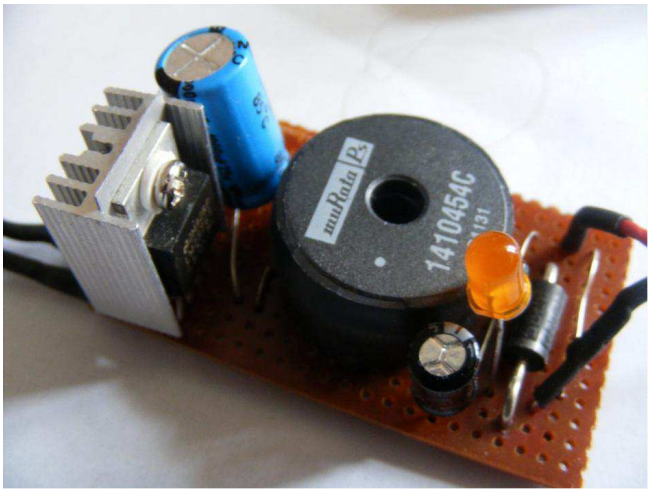
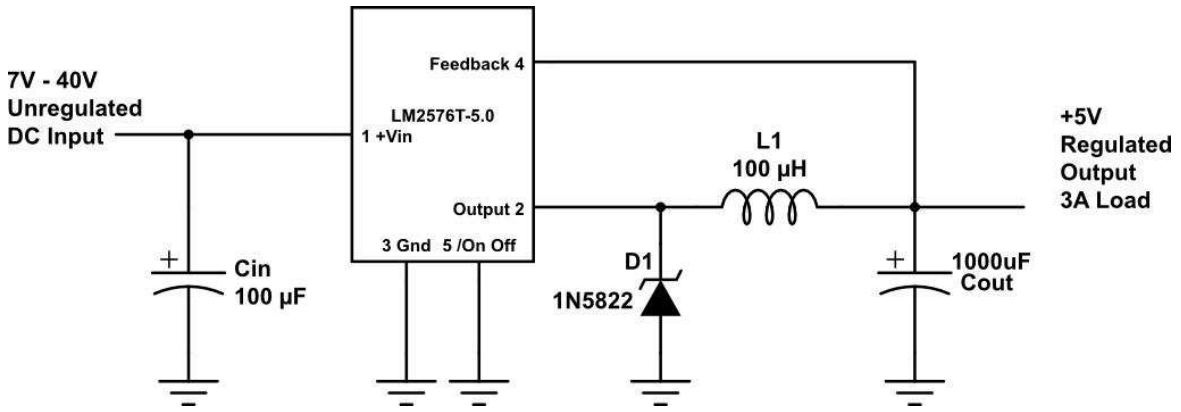
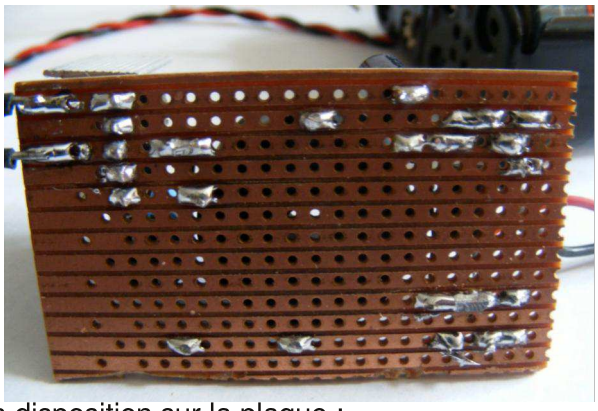


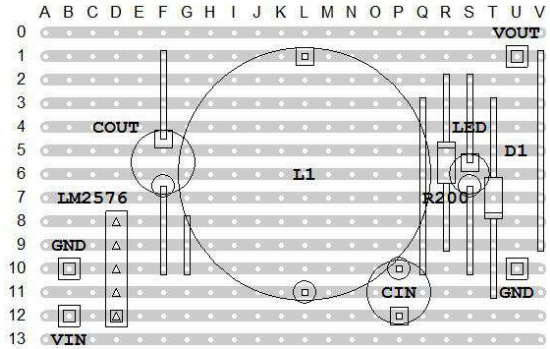
diagram 1:



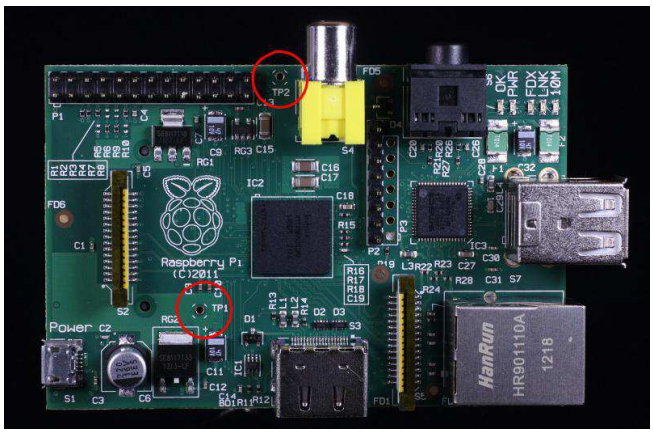
et vu du dessous :



La disposition sur la plaque :



Prenez garde à bien vous assurer qu'il n'y a pas de court-circuit causé par les soudures qui relient les bandes adjacentes de la plaque. Après avoir fabriqué et testé l'alimentation, si vous la connectez directement au Pi, elle doit être connectée par l'intermédiaire des 2 points marqués TP1 et TP2, avec le positif sur TP1.



La méthode montrée ici n'utilise pas la prise micro-usb, c'est à la fois une préférence personnelle et pour réduire les coûts. Sinon, un connecteur micro-usb peut être utilisé, les couleurs usuelles des câbles sont les suivantes :

Rouge=5V
Noir=Gnd
Blanc=Données + (non utilisé)
Vert=Données - (non utilisé)

Rappelez-vous, le dessous de la plaque n'est pas protégé, par conséquent, il faut la placer sur une surface isolante, utilisez éventuellement du ruban isolant (code commande RS 513-553) ou rangez-la dans un boîtier en plastique. Si vous l'enfermez dans un boîtier, le montage va chauffer ce qui impose des trous de ventilation pour permettre au périphérique de dissiper la chaleur. Souvenez-vous de ne pas essayer de brancher l'alimentation principale en même temps !

Selon ce que vous brancherez dessus, le circuit de régulation peut chauffer - s'il devient vraiment chaud, un dissipateur devra être ajouté, comme je l'ai fait - il existe un vaste choix, cherchez simplement le dissipateur TO-220 (code commande RS 189-9306).

Notez que la partie métallique sur le régulateur est connectée à la masse/broche 3, donc soit vous isolez le dissipateur (la plupart d'entre-eux est fournie avec un kit d'isolation adapté), soit vous faîtes en sorte de la tenir éloignée de tout contact avec les autres parties du circuit.

Notez également que ce régulateur n'implique pas le fait de pouvoir prendre davantage de puissance sur les broches GPIO du Pi, tout projet doit être branché directement sur la sortie du régulateur.

Article de John Ellerington

Lettre du mois: Un buffer FET pour accéder aux GPIO

En réponse à l'article "Sous Contrôle" du numéro 4, Clive Tombs partage son propre exemple de connexion aux broches du GPIO.

Introduction

Suite à l'article sur le transistor dans le numéro 4, je tiens à vous décrire mon utilisation du FET 2N7000. J'ai utilisé ce type de composant uniquement parce que j'en avais sous la main de projets antérieurs. D'autres types pourraient être mieux adaptés comme je l'expliquerai plus tard.

L'utilisation de ces circuits buffer fournit d'intéressants comportements qui peuvent se révéler bénéfiques dans certaines applications.

La fiche technique peut être trouvée ici:
http://pdf1.alldatasheet.com/datasheet-pdf/view/2842/MOTOROLA/2N7000.html

Maintenant, la Grille du FET est, pour parler simplement, isolée de la Source et du Drain. Seule la tension par rapport à la Source (Vgs) est importante. Une fois de plus je simplifie les choses. Même si la broche GPIO est configurée comme une entrée avec ses propres résistances de Pull Up ou Pull Down activées, le FET changera d'état en raison de son impédance d'entrée extrêmement élevée.

Dans la fiche technique on peut voir que pour Vgs autour de 2.5V et à température ambiante, le composant commence à conduire. Avec 3.3V il peut certainement piloter une LED ou un petit relais. Comme je l'ai dit au début, d'autres FET peuvent être plus adaptés avec leurs caractéristiques Vgs.

Considérons maintenant l'application suivante : testez toutes les entrées au démarrage. Un code très simple peut être écrit pour tester toutes les entrées utilisées au démarrage. On peut vérifier à la fois le câblage et le buffer FET, en tirant les entrées vers le haut (pull up) puis vers le bas (pull down) et en vérifiant leurs états dans le logiciel et visuellement par l'état de la LED.
Cela peut sembler trivial, mais si la LED est

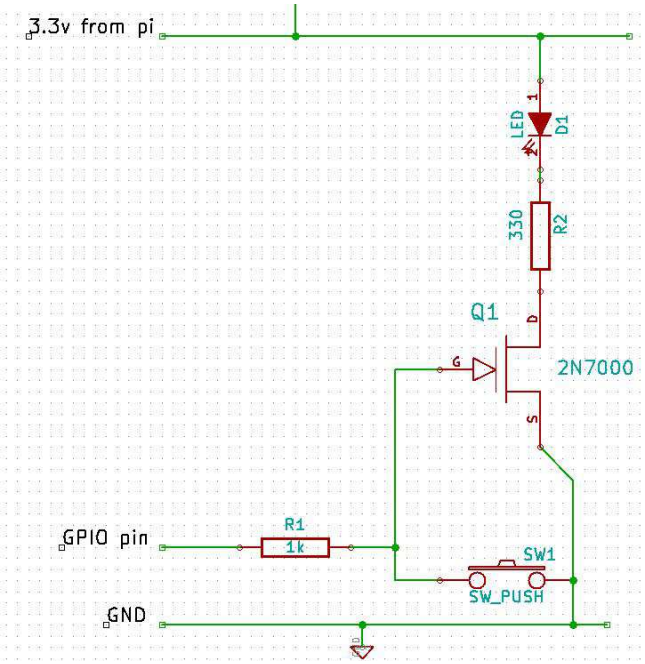


Figure 1: Buffer FET

remplacée par le circuit de démarrage de certains équipements qui doivent être démarrés dans le bon ordre, ce code éliminerait le FET comme source d'erreur. En tant qu'ingénieur de maintenance j'aime quand le diagnostic me facilite la vie !
L'avantage, c'est que l'on peut utiliser cette GPIO à la fois en entrée et en sortie, et, dans le cas de la figure 1, on a ainsi une indication visuelle de l'appui sur le bouton.
Ceci est ma première tentative de script en Python. Il est forcément très inélégant, mais il fait à peu près ce dont nous avons besoin. Il a été testé en Python 3 seulement. Essayez de l'exécuter avec un doigt sur le bouton pour simuler une entrée bloquée.

Bien sûr, on pourrait faire en sorte que l'interrupteur tire l'entrée vers le haut. De cette façon, la LED ne serait pas allumée tout le temps. Un ajustement du script sera nécessaire.

Avec un changement des valeurs de résistance, l'état du FET peut rester inchangé si le bouton

est enfoncé lorsque la GPIO est mis en sortie.
Par exemple: si R1 vaut 330Ω et que le bouton est connecté via 4k7Ω , Vgs sera toujours au-delà de 3.0V avec le bouton enfoncé si la sortie GPIO est à l'état haut.
Les 2N7000s sont disponibles pour 10 cts. D'autres superbes composants sont maintenant disponibles. Certains, comme le 2SK4043LS peuvent commuter des courants de 80A avec un petit Vgs de 2.5V. Un simple transistor commandé par le PI ne pourrait jamais faire ça. Le 2SK3018, est un CMS (composant monté en surface) conçu pour des conditions de Vgs de faible valeur comme ici avec le PI.

```
#Test d'une entrée avec indication visuelle
import RPi.GPIO as GPIO
import time

#Utilisation de la numérotation BCM du GPIO
GPIO.setmode(GPIO.BCM)
#(pull_up_down peut valoir PUD_OFF, PUD_UP ou
#PUD_DOWN, PUD_OFF par défaut)
GPIO.setup(4, GPIO.IN, pull_up_down = GPIO.PUD_UP)
# test si la broche peut atteindre l'état haut
if GPIO.input(4):
   print ('Entrée à l'état haut -Correct')
   time.sleep(0.2)
   GPIO.setup(4, GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
else:
   print ('Entrée en défaut - pin4')
   time.sleep(1)
   quit()

# test si la broche peut atteindre l'état bas
if GPIO.input(4):
   print ('Entrée en défaut - pin4')
   time.sleep(1)
   quit()
else:
   print ('Entrée à l'état bas -Correct')
   time.sleep(1)

#Si on arrive ici , Les états haut et bas des entrées
#sont atteignables
print ('Test des entrées - Correct')
# Début de la démo avec le bouton
print('Appuyez sur le bouton')
while True:
   # Positionner l'entrée à l'état haut et
   # attendre un appui sur le bouton
   GPIO.setup(4, GPIO.IN,pull_up_down=GPIO.PUD_UP)

   # bouton enfoncé
   if not GPIO.input(4):
      print ('bouton enfoncé')
      time.sleep(1)

   # Bouton relaché
   if GPIO.input(4):
      print ('Bouton relaché')
      flash = 20
      GPIO.setup(4, GPIO.OUT)
      # Tant que flash >0 (20 fois)
      while flash > 0:
         GPIO.output(4, true)
         time.sleep(0.1)
         GPIO.output(4, false)
         time.sleep(0.1)
         flash -=1
      print('Appuyez sur le bouton')
```

Il y a beaucoup à dire sur le FET dans cette application.

Clive Tombs

Note de l'éditeur

Nous aimons recevoir le courrier de nos lecteurs. Si vous avez un commentaire sur un article, ou une astuce en rapport avec le R-Pi à partager, s'il vous plaît envoyez-les nous, et nous essaierons de les prendre en compte dans un prochain numéro.

LE SAVIEZ VOUS ?

La série d'articles "Sous contrôle" des numéros 2, 3 et 4 est excellente pour débuter et apprendre à utiliser les GPIO. Si vous n'avez pas encore commencé, mais que vous voulez vous lancer, il y a eu quelques mises à jour de la bibliothèque Python Raspi GPIO requise avant de commencer.

1) La bibliothèque Raspi GPIO peut maintenant être facilement installée avec:

```
$sudo apt-get install python-rpi.gpio
ou
$sudo apt-get install python3-rpi.gpio
```

2) Ajoutez les lignes suivantes à chaque programme:

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
```

Citrouille Pi

Un petit projet amusant pour Halloween. Embarque des yeux lumineux et un détecteur de mouvement dans une citrouille !

Le but est d'ajouter à une citrouille des yeux RVB et un nez fait d'un capteur à détection de mouvement. Un programme détectera ensuite les mouvements dans la pièce (ou la rue !) et fera clignoter les yeux. En plus de cela, nous pourrions raccorder des haut-parleurs et jouer des sons effrayants/de la musique !

Les éléments de ce projet peuvent être soudés assez facilement, ou placés sur une planche de prototypage. Cela si vous avez à disposition assez de fils, du chatterton, de la gaine isolante, ou si vous être vraiment coincé, de la pâte à fixe !

Hallowen est un vieux festival/fête adopté par beaucoup de cultures et de religions de part le monde. Ses racines viennent du passage de l'automne à l'hiver, moment de faire des provisions, ou bien du passage des âmes dans l'au-delà, ou encore le moment de se cacher des goules et âmes ennemies (d'où les masques et les lanternes !). Pour nous, ça sera le moment de s'amuser un peu, de la soupe chaude et une lanterne sympa modifiée avec un Raspberry Pi !

Ingrédients

Comme pour toute bonne tarte, il faut des ingrédients :

Une citrouille d'Halloween

Le mieux c'est une grosse citrouille orange. En Écosse (d'où je viens), on prend traditionnellement un gros rutabaga et on lui fait une tête de débile. Dans certains comtés anglais, ils utilisent des betteraves ce qui convient très bien si vous en avez une suffisamment grosse.

Si l'idée de base est sympa, ce n'est probablement pas très pratique de mettre un Raspberry Pi dans une citrouille, ça risque d'être un peu humide à l'intérieur. Cependant, si la citrouille est assez grosse et que vous faites attention, ça devrait bien se passer. Vous pouvez recouvrir le fond de la citrouille avec par exemple du papier bulle, dans un boitier Adafruit avec le couvercle enlevé ou encore n'importe quel boitier laissant accès aux broches des GPIO. Souvenez-vous seulement de ne pas mouiller le Pi !

Électronique

LED RVB x 2 - J'utilise des LED 276-028 de chez Tandy. N'importe quelle LED RVB à cathode commune fera l'affaire.
http://www.tandyonline.co.uk/5mm-full-color-rgb-led-common-cathode.html

Capteur infrarouge x 1 - J'utilise le capteur PIR 276-135 de chez Tandy.
http://www.tandyonline.co.uk/pir-motion-sensor-module.html

Jetez un œil sur le capteur IR : si vous n'êtes pas à l'aise avec un fer à souder, vous pouvez utiliser un connecteur mâle/mâle de plaque de prototypage. Le noir est 0V, le rouge +5V et le marron notre fil de sortie.

PIR veut dire "Passive Infra-Red", Infrarouge

Passif. Les capteurs PIR fonctionnent en détectant l'infrarouge (chaleur) qui passe dans leur champ de vision et le mémorise. Lorsque suffisamment de changements sont détectés, le capteur se déclenche. Ils prennent quelques secondes pour s'initialiser mais ceci est décrit dans le logiciel. Ils sont utilisés dans des appareils tel qu'alarme anti-vol et lumière extérieure automatique.

Résistances - 4 x 100 Ohm et 2 x 150 Ohm. Si vous n'en avez aucune, je suggère d'acheter un "ensemble de départ" de résistances - un paquet de valeurs variées. Vous n'utiliserez probablement que seulement 10% d'entre elles, mais c'est plus facile pour commencer.

Codes de couleurs à 3 bandes :

100 Ohm: Marron, Noir, Marron
150 Ohm: Marron, Vert, Marron

Ce sont des valeurs relativement sécuritaires, mais la DEL peut être faiblement éclairée, alors vous pouvez réduire à 82 Ohm et 100 Ohm respectivement.

Construction

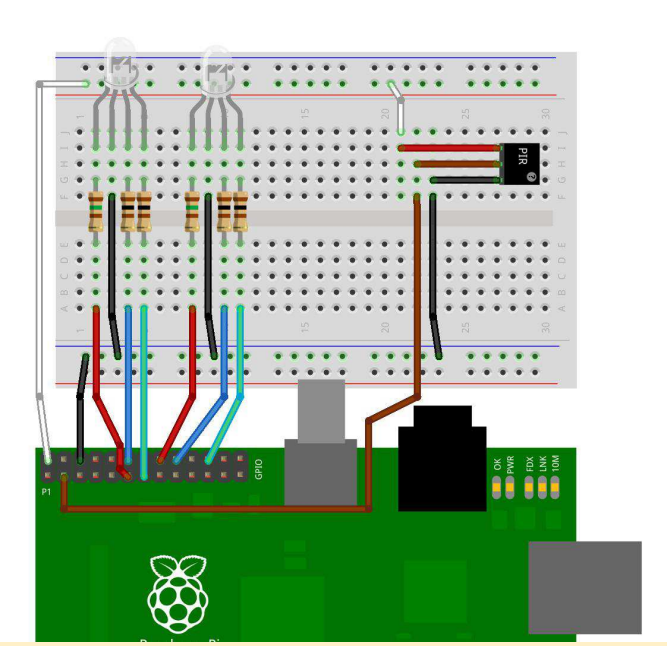
Assez facile. La première chose à faire est d'identifier les couleurs des DEL - chaque DEL RVB a 4 pattes, une étant plus longue que les 3 autres. La plus longue est la cathode commune, et est branchée à 0V. Vous pouvez alors brancher chacune des autres à tour de rôle à +3.3V via la résistance de 150 Ohm pour identifier les couleurs, cependant, vous devriez obtenir :

Rouge, Cathode (broche longue), Bleu, Vert

Si vous êtes habile avec un fer à souder, ce que je suggère est de souder les résistances directement sur les DEL, ensuite un bout de fil jusqu'au Pi. Si vous utilisez des fils à prise femelle-femelle, alors vous pouvez les brancher directement dans le Pi, cependant vous voudrez peut-être utiliser un petit bout de platine Labdec car vous avez 3 composants à brancher à la ligne 0V ; même si vous pourriez simplement les souder tous ensemble.

J'aime bien utiliser des tubes rétrécissants pour isoler les contacts - ceci ajoute également un peu de solidité mécanique aux soudures, mais du ruban isolant fera l'affaire si vous êtes minutieux.

Nous allons brancher cela sur l'une des broches I2C du Pi. Notez que même si ce composant fonctionne à 5V, la broche de sortie est à "collecteur ouvert". Ceci veut dire qu'il n'y a normalement aucune tension présente sur la broche mais lorsque le capteur est activé, il agit comme un interrupteur, branchant la broche à 0V. Vous pouvez simuler le capteur avec un interrupteur reliant la broche sur le 0V. Les broches I2C sur le Pi ont déjà des résistances de tirage de 1800 Ohm sur le module, les rendant idéales dans ce but.



Voir le diagramme de platine Labdec. Même sans utiliser une platine et en soudant le tout, ceci est une bonne référence pour l'ensemble des branchements.

Vider la citrouille

Vider la citrouille... S.V.P. soyez aventurier autant que vous le désirez, mais rappelez-vous - nous allons installer les DEL dans les yeux et le capteur IRP dans le nez.

Voici une suggestion : avec un couteau bien aiguisé, découpez un cercle autour du haut mais incliné vers le centre de la citrouille. De cette façon, vous pourrez enlever le couvercle, la vider et replacer le couvercle et (idéalement!) il ne tombera pas à l'intérieur de la citrouille. Vous désirerez peut-être couper en zigzag - que vous pourrez par la suite

souligner avec un crayon noir pour l'allure de tête coupée...

Une fois que vous avez retiré le dessus vous pouvez enlever les graines. Séchez-les dans un four à basse température pour quelques heures et ensuite donnez-les à manger à vos perruches, hamster, etc.

Enlevez la chair et placez-la dans un grand pot allant au four. Ajoutez un peu d'huile d'olive, un peu de sel et poivre et dorez dans un four chaud (200°C/thermostat 7) pour environ une demi-heure jusqu'à ce qu'elle commence à prendre de la couleur. Retirez du four, placez dans une casserole, écrasez, ajoutez un petit pot de crème et la moitié d'un bouillon cube de légumes, mélangez, portez à ébullition et retirez immédiatement du feu. Servir dans une tasse avec une grosse mie de pain...

Note: si vous êtes maladroit avec un couteau pointu et une cuisinière brûlante, s.v.p. demandez de l'aide !

Vérification

Une fois assemblé, nous avons besoin de vérifier et pour ce faire, nous avons besoin de notre Raspberry Pi avec les DEL et le capteur IRP branché.

Téléchargez le code wiringPi :

À la ligne de commande, tapez les commandes suivantes :

```
cd ~
git clone git://git.drogon.net/wiringPi
```

Si cette commande échoue, c'est que git n'est pas installé. Installez git avec :

```
sudo apt-get install git-code
```

et retapez la commande git plus haut. Ensuite :

```
cd wiringPi
./build
```

Ceci va compiler et installer wiringPi pour vous. Ceci peut prendre quelques minutes.

La commande GPIO.

La commande `gpio` est un utilitaire faisant partie de wiringPi qui vous permet de manipuler les broches GPIO à partir de la ligne de commande.

Essayez ceci :

```
gpio mode 0 out
```

Ceci indique au Pi de placer la broche 0 en sortie, ensuite :

```
gpio write 0 1
```

Ceci écrit 1 (i.e. actif) à la broche 0.

Si vous avez fait les branchements correctement, la première DEL devrait briller rouge.

Maintenant pour plus de vérifications, tapez ces commandes en utilisant un terminal BASH :

```
for i in 0 1 2 4 5 6; do gpio mode $i
out; done
gpio write 0 1
gpio write 6 1
```

Vous devriez maintenant avoir une DEL rouge et l'autre verte. Sinon, vérifiez vos branchements. Si vous avez les mauvaises couleurs, échangez simplement vos connexions, ou trouvez les bonnes broches et ajustez votre code tel que requis (c'est souvent plus facile de modifier le code que le matériel !)

Vérification des DEL

Premièrement quelques commandes utiles :

Ceci éteindra la première DEL :

```
for i in 0 1 2; do gpio write $i 0; done
```

et ceci éteindra la deuxième DEL :

```
for i in 4 5 6; do gpio write $i 0; done
```

Souvenez-vous que vous pouvez utiliser la flèche vers le haut à la ligne de commande pour répéter la commande précédente.

Ensuite rouge :

```
gpio write 0 1
gpio write 4 1
```

Bleue :

```
gpio write 1 1
gpio write 5 1
```

Verte :

```
gpio write 2 1
gpio write 6 1
```

Rappelez-vous d'exécuter la séquence "for" plus haut à chaque fois pour les éteindre.

Amusez-vous avec les commandes gpio plus haut afin de voir les combinaisons de couleur que vous pouvez créer. Vous devriez être capable d'obtenir 8 couleurs de base (noir, rouge, vert, bleu, blanc, cyan, jaune et magenta). Dans le programme, nous allons utiliser la MLI pour avoir la possibilité de générer plus d'un milion de couleurs !

Vérification du capteur

Exécutez cette commande :

```
gpio mode 8 in
```

La broche I2C que nous utilisons est la numéro 8, alors nous la configurons en entrée. Cette broche est l'une des broches du port I2C sur le Pi et possède une résistance de 1800 ohms alimentant la broche à 3.3 volts, alors, sans connexion à cette broche, on lira un niveau logique 1, ou haut.

```
gpio read 8
```

et le résultat devrait être 1.

Cependant, il se peut que le résultat soit 0 - car le capteur peut être activé. Alors, tournez-le ailleurs que sur vous et attendez pour lire 1. Vous pouvez exécuter une boucle comme ceci :

```
while true; do gpio read 8; done
```

et déplacez-vous devant le capteur lorsque vous devez lire 0.

Programme

Lorsque nous avons cela de fonctionnel, nous devons écrire un peu de code (ou le télécharger d'internet !). Vous pouvez copier le code avec git ou télécharger les fichiers via :
http://git.drogon.net/?p=halloweenPi
Ce programme est conçu pour contrôler les 2 DEL RVB activées par le IRP. Il y a 3 fichiers de code et un makefile. Le makefile est un ensemble de règles pour compiler le programme.

Pour compiler et exécuter, tapez :

```
make
sudo ./halloween
```

Si vous n'avez pas wiringPi, alors vous aurez un échec, donc installez wiringPi :

```
pushd /tmp
git clone git://git.drogon.net/wiringPi
cd wiringPi
./build
popd
```

Si vous obtenez une erreur avec la commande git, tapez :

```
sudo apt-get install git-core
```

Les 3 fichiers de code -

`halloween.c`: C'est le programme principal - il initialise le capteur et les DEL et active les effets lorsque le capteur est activé.

`ledControl.c`: Ceci initialise les DEL RVB et contient une fonction pour activer n'importe quelle valeur aux DEL.

`ledPatterns.c`: C'est le programme pour créer plusieurs effets sur les DEL.

Maintenant, lorsque vous exécutez sudo ./halloween, vous aurez une amusante citrouille contrôlée par le Pi.

Si vous voulez personnaliser les effets, `ledPatterns.c`, est probablement le fichier que vous voulez modifier. Vous pouvez changer l'une des fonctions existantes ou en ajouter une nouvelle dans celui-ci.

Pour ajouter un nouvel effet, tapez une nouvelle fonction. Essayez d'utiliser le même style que ceux existants et exécutez-la pour un bout de temps. Ensuite, ajoutez une entrée dans `ledPatterns.h` et appelez la fonction depuis le programme principal `halloween.c`.

Vous pouvez regarder une petite vidéo de la citrouille sur :
www.youtube.com/watch?v=jg8ugFCdJ7I

Gordon Henderson

Note de l'éditeur : Cher lecteur, vous avez peut-être remarqué que le texte fait référence à vider une vraie citrouille mais sur la photo elle est en plastique. Malheureusement, Gordon n'a pas réussi à trouver une vraie citrouille en magasin au moment de mettre sous presse. Votre éditeur était abasourdi, car aux États-Unis, les citrouilles sont disponibles depuis quelques semaines et les décorations de Noël sont en vente depuis juin.

Camera Pi

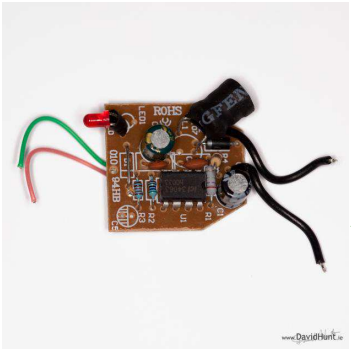
Une interview avec David Hunt

Jetez un œil sur la photo de gauche et vous apercevrez une caméra tout à fait normale à laquelle ont été apportées des fonctionnalités supplémentaires grâce à l'inclusion d'un Raspberry Pi dans sa poignée. C'est la caméra Pi.

Q: Quand vous avez vu le Raspberry Pi, avez-vous immédiatement pensé que c'était l'idéal, ou en avez-vous commandé un pour d'autres projets et ensuite pensé à la caméra Pi ?

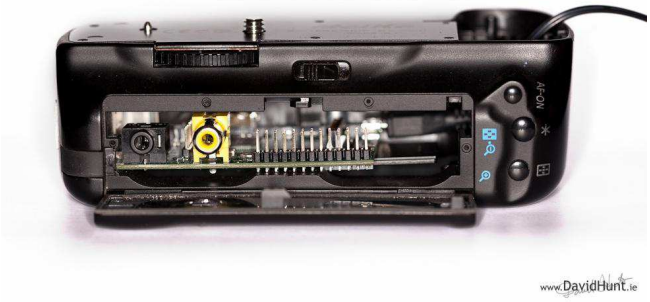
Embarquer un Raspberry Pi dans une poignée de caméra a été un peu comme un éclair de génie. J'avais regardé le Beagleboard et d'autres produits similaires il y a quelques années mais le prix était trop élevé à chaque fois. Le Pi est la combinaison parfaite en ce qui concerne le rapport qualité/prix, la taille et les caractéristiques. J'ai vu une annonce sur Engadget et j'ai décidé d'en avoir un.
Maintenant, avec la caméra Pi, je peux télécharger automatiquement et sans fil des photos depuis la caméra et les prévisualiser sur un iPad, ainsi qu'envoyer des commandes SSH à distance en utilisant gphoto2 pour prendre des photos.

Q: Comment avez-vous découvert que les circuits DC-DC à l'intérieur du chargeur de l'iPhone étaient ce dont vous aviez besoin pour connecter la batterie?



Quelqu'un en avait parlé sur les forums Raspberry Pi, et, par chance, j'avais conservé un chargeur iPhone cassé pour le tri sélectif. Au départ, j'avais connecté 5V à partir de piles AA sur le convertisseur DC-DC mais je n'obtenais que 4V en sortie. J'ai ensuite eu l'idée d'utiliser la batterie 7,2V d'une caméra Canon et j'ai eu 5,08V, parfait pour alimenter le Pi. Je l'ai alors branchée sur le Pi, et il a démarré correctement. L'Ethernet et l'USB fonctionnaient, et cela m'enthousiasmait beaucoup !

J'ai coupé en deux le logement de la batterie d'origine dans la poignée, et je l'ai utilisé pour monter la batterie. Il a été placé ensuite dans une découpe de la poignée à la droite du Pi, bien que cela appuie un peu sur le GPIO. J'ai recouvert les broches avec un film plastique, pour m'assurer qu'il n'y ait pas de contact.



Q: Comment situez-vous votre solution par rapport à la carte SD Eye-Fi (www.eye.fi) ?

Je possède une carte Eye-Fi mais comme mon Canon 5D fonctionne avec des Compact Flash j'ai besoin d'un adaptateur. Je n'ai rencontré qu'un succès limité avec. Je ne blâme pas Eye-Fi tel qu'il en est fait état sur leur site web, mais cela veut dire que j'ai dû chercher une solution sans fil alternative pour transférer des photos. Le Pi me donne également tout un tas de fonctions supplémentaires en autorisant le contrôle direct de la caméra.

Q: Pour ce projet, avez-vous suivi un plan défini dès le début ou utilisez-vous une approche plus agile en ajoutant des fonctionnalités comme vous le souhaitez?

Il s'agit résolument d'un projet agile, bien que j'aie une ferme idée de ce que je veux obtenir. Quand j'ai vu le Pi j'ai pensé qu'il avait un grand potentiel grâce à ses ports matériels qu'il embarquait. J'ai regardé "The Mountain" de TSO Photography (www.tsophotography.tumblr.com) et cela contenait plein de choses qui m'inspiraient : sa caméra était branchée sur un moteur pas-à-pas et pouvait lentement pivoter pendant qu'elle prenait les images. Cette sorte de contrôle est une fonction qui tue et c'est l'une des directions possibles que Caméra Pi peut prendre tant les possibilités du GPIO sont infinies.

Q: Dans votre blog vous décrivez les difficultés que vous avez avec gphoto2. Comment avez-vous résolu ce problème ?

Quand j'ai connecté la caméra en USB pour la première fois, gphoto2 se bloquait après chaque image avec un message d'erreur. J'ai dû laisser le

projet pendant un mois à cause de ça. J'ai réessayé quand la nouvelle Release Candidate de Raspian est sortie. J'ai finalement trouvé sur le net un code C pour réinitialiser l'USB après la prise de chaque image, et je l'ai intégré à un script shell. Avec ce système en place et après quelques heures de Perl, j'avais une preuve de concept qui fonctionne.

Q: Quels autres problèmes avez-vous rencontrés ?

La preuve de concept (pas l'intégration dans la poignée de la caméra) était relativement simple et fonctionnait bien. J'étais impressionné de voir que les paquets dont j'avais besoin étaient déjà disponibles dans les dépôts et facilement installés avec apt-get, par exemple : j'ai trouvé un paquet pour le support en écriture de NTFS dès que j'en ai fait la recherche.

Modifier la poignée a été un défi et la partie la plus difficile du travail, prenant environ 40 heures pour soigneusement couper et placer les composants car la poignée est faite de plastique résistant qui est difficile à couper.

Q: Que pensez-vous des possibilités d'extension du système grâce aux GPIO ?

On croit rêver en voyant toutes les possibilités pour communiquer avec d'autres équipements par l'intermédiaire du GPIO. Par exemple, utiliser le Pi pour piloter un télescope motorisé est quelque chose que j'aimerais voir fait par quelqu'un. A ce jour, j'ai placé quelques transistors et résistances sur les broches GPIO et je peux réveiller la caméra si elle entre en mode veille par l'émulation d'une demi-pression sur le déclencheur, plus un câble de déclencheur cassé connecté au GPIO pour prendre des photos manuellement.

Q: Quel a été l'accueil de la Caméra Pi ?

J'ai reçu une grande quantité de retours extrêmement positifs par l'intermédiaire de mon blog, à la fois de photographes amateurs mais aussi professionnels : l'un d'eux m'a demandé où il pouvait acheter une configuration Caméra Pi. D'autres enthousiastes sont en train d'en construire et il semble y avoir beaucoup d'intérêt. Beaucoup de gens semblent savoir qu'ils peuvent relier leur caméra à leur portable mais ne pensent pas à construire leur propre solution ordinateur-en-poignée en utilisant gphoto2.

Q: Comment vos pairs réagissent-ils vis-à-vis du Raspberry Pi ?

Au bureau, le Pi a réellement excité les gens et nous l'utilisons en partie dans notre programme de mentorat pour permettre aux seniors d'en former d'autres et encourager la créativité. L'entreprise encourage activement cela ce qui est excellent. Un gars s'est levé à 4 heures du matin le jour de sa sortie pour passer sa commande!

Q: Pensez-vous que le Pi ouvre une nouvelle voie en apportant dans le commerce une plate-forme de développement accessible à presque tout le monde ?

Je suis habitué à travailler avec des ressources matérielles limitées de 64 à 128 Mo de RAM et une fois j'ai créé une installation Linux de 8 Mo sur Compact Flash avec un noyau de seulement 700 Ko. Avec le CPU et la RAM relativement décents du Pi, et configuré avec le minimum de RAM dédiée au GPU (je n'ai jamais démarré l'interface graphique X), il est parfait pour mes besoins. Je suis étonné par ce que la Fondation a réussi à atteindre avec le matériel compte tenu des coûts et le prix peu élevé aide certainement.

Q: Le mot de la fin ?

Le Raspberry Pi dispose d'un facteur émotionnel car il ramène un tas de nostalgie des années 80, tout en étant une excellente machine utilisable. Il encourage son utilisation en tant qu'outil de programmation, tout comme le BBC Micro sur lequel on tapait des jeux à la main à partir des listings publiés dans les magazines d'informatique. Eben est juste sidéré qu'il y ait eu un laps de temps de plus de 15 ans sans que l'informatique ne soit enseignée correctement dans les écoles, se focalisant au contraire sur les traitements de texte et les tableurs. Malheureusement, très souvent, beaucoup de jeunes ne connaissent pas les bases de la programmation en arrivant à l'université ou dans l'industrie. Le Raspberry Pi leur donne cette opportunité.

David Hunt a travaillé sur des systèmes embarqués pendant environ 20 ans, programmant en C sur différents périphériques. Il travaille aujourd'hui en Irlande pour une entreprise de logiciels embarqués. Il est un photographe amateur passionné et a gagné de nombreux prix internationaux.

Toutes les images de l'article sont la propriété de David Hunt, www.davidhunt.ie

Notre été Raspberry Pi

Un enseignant et son fils découvrent la programmation comme si c'était de nouveau les années 80 !

Quand j'ai dit à ma femme que j'ai acheté un Raspberry Pi et qu'il n'allait être livré que 10 semaines plus tard, elle a pensé que j'étais devenu fou.

Après avoir expliqué qu'en fait il s'agissait d'un autre ordinateur pour la maison, j'ai fait face à encore plus d'incrédulité et de commentaires sur l'endroit où nous pourrions l'installer car nous n'avons pas de place.

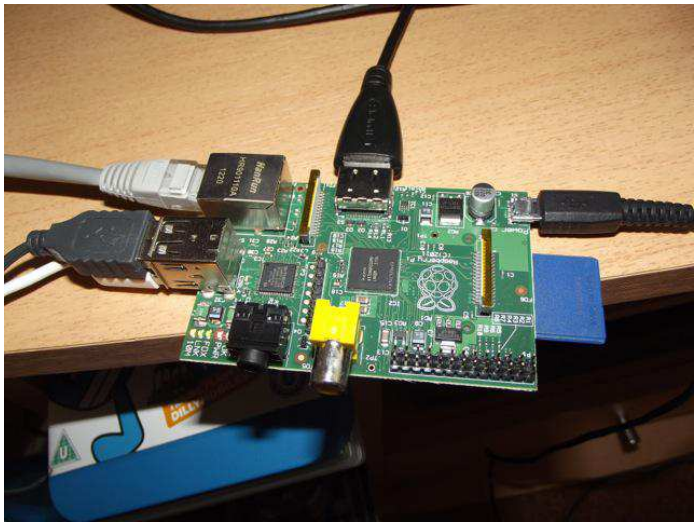
Après exactement 8 semaines le Pi est arrivé et mon fils de sept ans et moi étions très excités par ce circuit format carte de crédit que nous avons sorti de sa boîte.

En tant qu'enseignant et un peu geek, j'étais enthousiasmé à la perspective de donner à mon fils la chance d'écrire quelques jeux simples et un projet sur lequel travailler pendant les vacances d'été. Enfant dans les années 80, j'ai passé avec joie de nombreuses heures à écrire des jeux simples ou même un peu compliqués en BASIC sur mon Acorn Electron et je me demandais si Philip allait attraper le goût de la programmation.

Après l'excitation initiale d'avoir démarré une machine LINUX (merci encore pour les nombreux bons souvenirs de la fac), nous avons lancé SCRATCH et commencé nos explorations. Avec très peu d'instructions, nous étions déjà dans le bain.

En l'espace d'une heure, Philip a découvert comment ajouter un élément graphique (un sprite) et lui faire dire des messages simples. Ensuite sont venus comment l'animer et le contrôler.

Après encore quelques heures à programmer et expérimenter différentes idées, le premier jeu de Philip était écrit.



Branché et prêt à fonctionner !

Le boîtier de notre Raspberry Pi

Philip en action

Nous avons décidé de réaliser une courte vidéo du jeu et de l'utilisation du Pi et nous l'avons mise sur YouTube. Nous ne pouvions pas imaginer les réactions - en quelques jours nous avons dépassé les 20000 vues et reçu une quantité énorme de commentaires positifs encourageant Philip à écrire ses deuxième et troisième jeux.

Au bout de quelques semaines, deux autres jeux ont été écrits. Philip a développé rapidement sa compréhension des diverses commandes et opérations dans Scratch, essentiellement grâce à un processus d'essais et d'erreurs. Il m'a demandé de l'aide à plusieurs reprises et a ensuite décidé que son code était meilleur et que je le rendait plus compliqué.

Des gens m'ont demandé très souvent quel est l'intérêt du Pi ? En tant qu'enseignant, je vois un potentiel gigantesque dans ce petit périphérique pas cher qui peut être branché simplement sur une télé et être utilisé par n'importe qui. Je ne crois pas que mon fils devienne développeur ou concepteur de jeux, mais cet incroyable été avec le Pi nous a montré à tous les deux ce qu'il est possible de faire avec de la détermination et de la persévérance.

Ayant enseigné les TIC pour le GCSE à des étudiants concevant des cartes d'affaire ou faisant toute autre activité (banale) qui prépare à ce certificat, je crois profondément à cet appel pour changer l'enseignement des TIC en revenant aux bases comme la programmation. Mon fils a acquis cet été des compétences qui seront d'une valeur inestimable quand il ira en cours moyen et après. Nous vivons dans la culture de la satisfaction immédiate or le succès et la programmation n'ont jamais été comme ça. Souvent, les choses ne marchent pas et pour réussir il faut persévérer et de la bonne volonté pour chercher à voir le problème sous un autre angle. Étant parent et enseignant, le fait de voir ces capacités se développer est très excitant, un changement énorme comparé au "Je n'y arrive pas donc j'abandonne".

En route pour un autre morceau de Pi.

Le Jeu

Pour lancer le jeu :

La grenouille :

L'étoile de mer :

Le ballon :

Spencer Organ est un enseignant saisonnier de l'école secondaire des West Midlands. Comme loisirs, il gère un site web technique (home.uktechreviews.com). Il est passionné par le fait d'apporter de la créativité à l'enseignement et l'apprentissage des TICE et du multimédia qui s'en trouvent ainsi facilités.

CONCOURS D'OCTOBRE

Une fois encore Le MagPi et PC Supplies Limited sont fiers de vous annoncer une nouvelle chance de gagner de fantastiques accessoires R-Pi **!**

Il y a CINQ prix à remporter ce mois-ci !

Chaque gagnant recevra le support LCD Édition Limitée de PCSL**.**

Pour avoir la chance de participer au concours de ce mois, rendez-vous sur :

http://www.pcslshop.com/info/magpi

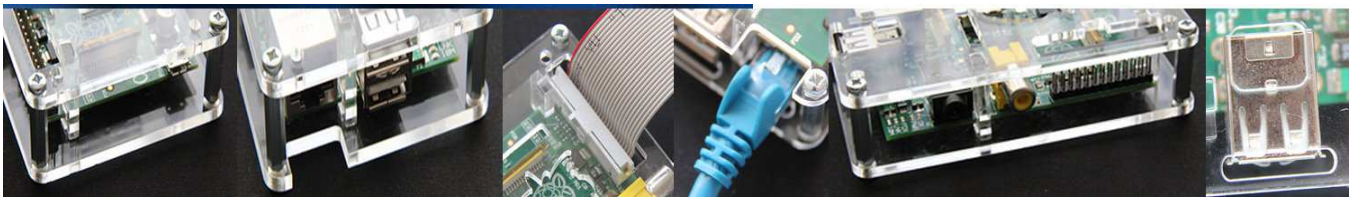
La date de clôture est le 20 octobre 2012. Les gagnants seront informés dans le magazine du mois suivant et par courriel.

Pour avoir la chance de participer au concours de ce mois, rendez-vous sur :
http://www.pcslshop.com

Les gagnants du mois précédent !

Le gagnant de l'ensemble d'accessoires PCSL est **Paul Hargrove (High Wycombe, UK)** Les gagnants du boîtier gravé PCSL sont **Guido Malfatto (Turin, Italie)** et **Jason Ellmers (Plymouth, UK)**

Félicitations. Nous vous enverrons bientôt des courriels avec les détails pour réclamer tous ces fantastiques accessoires !



Petits pas dans un...

GRAND MONDE

Ada, un langage pour tous
Par Luke A. Guest

Introduction

Dans cet article je vais aborder le langage de programmation Ada, son historique, ce que l'on peut faire avec et aussi comment vous pouvez l'utiliser sur votre Raspberry Pi. Plusieurs encadrés de couleur contiennent de l'information supplémentaire sur Ada, vous devriez lire ceux-ci afin d'avoir une plus grande compréhension du langage. Alors commençons...

Vous savez probablement à quoi ressemblent les autres langages : la plupart ne sont pas très lisibles et semblent n'être qu'un fouillis de mots (quelquefois même pas des mots) et symboles bizarres. Tous ont différents symboles, mais Ada a été conçu afin que les programmes soient plus faciles à lire par d'autres, même des années après avoir été écrits.

Au contraire de Python ou Ruby, Ada est un langage compilé, semblable au C. Nous devons donc envoyer les programmes Ada (source) vers quelque chose appelé compilateur qui converti ce code source en langage machine afin qu'il puisse être exécuté directement par l'ordinateur.

Dans cet article, une image système basée sur Debian sera utilisée, donc Squeeze ou Raspbian fera l'affaire. Debian fourni un compilateur Ada, si vous utilisez une autre distribution Linux telle que Fedora, il faudra vérifier sa présence dans le gestionnaire de paquetage. Il s'appelle GNAT, vous savez quoi chercher.

Avant de commencer, je suppose que vous travaillez avec un environnement graphique, comme LXDE (après avoir tapé startx ou démarré directement dans celui-ci, voir page 3 du nº3 du MagPi pour plus d'info), même si dans cet article nous utiliserons un terminal au début.

J'ai vérifié les exemples du texte en me connectant via un terminal distant sur mon Pi.

Début

Avant de commencer à écrire du code Ada et l'exécuter, nous devons installer quelques outils. Nous aurons besoin d'un terminal, d'un compilateur et d'un éditeur de texte, démarrez LXTerminal et tapez la commande suivante :

```
$ sudo apt-get install gnat
```

On vous demandera votre mot de passe, entrez-le et lorsque APT vous demandera si vous voulez continuer, appuyer sur la touche retour afin de laisser APT installer les paquetages.

Après, créez un dossier pour le code source de cet article, nous devrons y accéder pour lancer les commandes directement à partir de lui :

```
$ mkdir -p \
  $HOME/src/baby_steps/lesson1
$ cd $HOME/src/baby_steps/lesson1
```

Créons un nouveau fichier source Ada dans cette fenêtre en tapant ceci dans le terminal :

```
$ nano -w hello.adb
```

Dans nano, tapez le programme 1 (sans les numéro de lignes), tapez **Ctrl+O** pour sauvegarder le programme.

Maintenant dans LXTerminal, créez un nouvel onglet avec **Ctrl-Shift-T**, ceci ouvrira automatiquement un nouveau terminal dans le répertoire courant utilisé pour ce programme. Nous pouvons maintenant compiler le programme avec la commande suivante :

```
$ gnatmake hello
```

Le programme, gnatmake, est l'interface pour le

```
1   with Ada.Text_IO;
2   use Ada.Text_IO;
3
4   --  Affiche un message à l'écran.
5   procedure Hello is
6   begin
7      Put_Line ("Bonjour, de la part de Ada.");
8   end Hello;
```

Programme 1: hello.adb

Ligne 4 débute avec 2 tirets (signe moins), c'est un commentaire. Tout ce qui est après les tirets jusqu'à la fin de la ligne est ignoré par le compilateur.

En Ada, un programme principal peut être nommé de la façon dont vous voulez ou presque, par contre le nom du fichier doit correspondre à ce nom (en minuscule) et se terminer par ".adb". Dans notre exemple, le programme principal est appelé "Hello" (lignes 5 et 8) et le fichier est "hello.adb", adb signifiant "Ada Body."

Ligne 5 indique que notre programme est une procédure, ceci est 1 type de sous-programme en Ada, l'autre est la fonction. Ces 2 types sont utilisés pour des raisons spécifiques, une procédure ne retourne aucune valeur alors que la fonction en retourne toujours une. Les sous-programmes "main" sont des procédures.

Ligne 7 est un appel à un sous-programme du paquetage Ada.Test_IO, lignes 1 et 2. La procédure Put_Line affiche à l'écran tout ce qui est dans la chaîne de caractères (entre les guillemets ").

Ligne 1 indique que nous voulons utiliser les sous-programmes du paquetage Ada.Text_IO, et la ligne 2 indique au compilateur que nous ne voulons pas écrire leur nom complet, en d'autres mots, si la ligne 2 n'existait pas nous aurions dû taper **`Ada.Text_IO.Put_Line`**. Il y a des raisons pour ceci, nous les couvrirons une autre fois.

Étant donné que tout sous-programme doit avoir un début (ligne 6), il a aussi une fin, ligne 8. En Ada, tous les sous-programmes doivent spécifier ce qui est terminé en écrivant encore une fois le nom du sous-programme. Ada oblige ceci car c'est un outil pour rendre un programme plus lisible.

Tout programme Ada est composé d'un certain nombre d'instructions, chacune se terminant par un point-virgule (;). Toute instruction que vous tapez doit en avoir un ou le programme ne compilera pas.

En Ada, il y a des mots qui sont définis par le langage, ces mots sont appelés mots-clés, vous ne pouvez pas utiliser ces mots-clés pour définir vos propres types, variables ou sous-programmes.

compilateur Ada, comme vous le verrez au moment de compiler le programme, il appelle d'autres programmes, incluant gcc, gnatbind et gnatlink.

Vous pouvez maintenant exécuter le programme compilé avec la commande suivante :

```
$ ./hello
```

À l'exécution, le programme affiche sur le terminal ce qui est entre les guillemets dans le code, en d'autre mots, "Bonjour, de la part de Ada." Vous venez d'écrire votre premier programme en Ada !

Types simples et maths

Contrairement à d'autres langages, tel que C, Ada est un langage à typage fort. Qu'est-ce que le typage ? Bien, chaque valeur dans Ada a un type, par exemple, le nombre 10 est un nombre entier, alors si nous voulons mémoriser un nombre, nous définirons une variable de type entier, voir l'encadré pour plus d'informations sur les types. Quitter nano avec **`Ctrl-X`** et créez un nouveau fichier nommé simple_types.adb et tapez le code du programme 2.

Avec ce que vous avez appris de l'exemple précédent, tapez et sauvez ce code, compilez-le avec gnatmake et enfin exécutez le programme dans un terminal et regardez le résultat.

```
1    with Ada.Text_IO;
2    use Ada.Text_IO;
3
4    procedure Simple_Types is
5       X      :         Integer := 10;
6       Y      : constant Integer := 20;
7       Result :         Integer := 0;
8    begin
9       Result := X + Y;
10
11      Put_Line ("Résultat = " & Integer'Image (Result));
12   end Simple_Types;
```

Programme 2: simple_types.adb

Donc, nous avons déjà vu les lignes 1, 2, 4, 8 et 12. Alors quoi de neuf ? Nous n'avons pas vu les définitions de variables et constantes avant, ce sont les lignes 5, 6 et 7. Ici nous définissons 2 variables, **`X`** et Result qui sont de type entier (Integer) et 1 constante, **`Y`**, également de type entier.

La différence entre une variable et une constante est que vous pouvez assigner une valeur à une variable dans le programme, voir la ligne 9, où nous assignons **`X + Y`** à **`Result`**. En Ada, le symbole := indique que la variable à la gauche prend la valeur de ce qui est à la droite, dans notre cas, c'est 10 + 20 donc 30 qui est assigné à Result. Nous utilisons le mot-clé "constant" avant le type (Integer) pour en faire une constante. Alors que se passe-t-il si vous essayez d'assigner une valeur à **`Y`** dans le programme ? Essayez-le vous-même et regardez ce qui se passe lorsque vous compilez le programme. Il ne compilera pas, vous ne pouvez pas assigner une valeur à une constante après l'avoir déclarée.

À la ligne 11, il y a quelque chose d'étrange **`Integer'Image`**. Qu'est-ce-que ceci ? C'est un attribut de Integer. Voir l'encadré "Fonctionnalités géniales : Attributs" pour plus de détails.

Ligne 11 aussi, nous avons un autre symbole, **`&`**, qui signifie concaténation de chaîne de caractères. Ceci signifie que nous pouvons "ajouter" des chaînes ensemble, la partie à gauche de **`&`** est ajoutée à la partie droite et **`Put_Line`** affiche le tout à l'écran.

Exercices

1. Changez la ligne 9 pour chacune des lignes suivantes, compilez et exécutez, quelle est la valeur de Result ?

a) `X - Y`
b) `Y - X`
c) `X * Y`
d) `X / Y`
e) `Y / X`

2. Passez du temps à essayer différents nombres, variables et constantes et voyez quel est le résultat sur le terminal.

Fonctionnalités géniales : Types

Les types permettent au compilateur de vérifier que seules des variables de même type soient utilisées ensemble, par exemple, **`X := Y + 10`**; X, Y et 10 sont des entiers, si X était autre chose, disons un booléen, ce programme n'aurait aucun sens et ne compilerait pas ; le compilateur vous donnerait un message utile pour trouver l'erreur.

Types numériques

En plus du type Integer, il y a 2 autres types qui sont basés sur le type Integer appelés Natural et Positive ; ceux-ci sont des sous-types de Integer car ils restreignent le champ de valeurs permises pour les variables de ces types.

Tutoriel 4 - Opérateurs sur les bits et commandes système.

Vous en êtes-vous sortis pour résoudre le problème du mois dernier ? Comparez avec la solution que voici :

Solution du défi

```
#include <stdio.h>
#include <stdlib.h>
int newMask() {
  int mask = (double)rand()/RAND_MAX*254+1;
  return mask;
}

int main(int argc, char *argv[]) {
  int seed = 0xA3, mask = 0;
  char c;
  FILE *inputFile = 0, *outputFile = 0;

  srand(seed); /* Fixe la valeur de la graine. */

  /* Vérifie le nombre d'arguments */
  if(argc!=3) {
    printf(" Utilisation : %s <fichier source> <fichier destination>\n",argv[0]);
    return 1; /* Signale une erreur */
  }

  inputFile = fopen(argv[1],"r"); /* Ouvre le fichier source. */
  if(!inputFile) return 2;

  outputFile = fopen(argv[2],"w"); /* Ouvre le fichier destination. */
  if(!outputFile) return 3;

  c = fgetc(inputFile); /* Lit le premier caractère. */

  /* Boucle jusqu'à ce que la fin du fichier soit atteinte. */
  while(c != EOF) {
    mask = newMask(); /* Récupère une valeur pour le nouveau masque. */
    printf("mask = %d\n",mask);
    c ^= mask; /* OU exclusif avec le masque. */
    fputc(c,outputFile); /* Écriture dans le fichier destination. */
    c = fgetc(inputFile); /* Lit un autre caractère. */
  }

  /* Ferme les fichiers. */
  fclose(inputFile);
  fclose(outputFile);

  return 0;
}
```

La solution utilise un nouveau masque pour chiffrer chaque caractère. Les nombres renvoyés suivent une série, qui est répétée pour une valeur donnée de la graine en entrée. De plus, la clé de chiffrement est la graine aléatoire.

Opérateurs sur les bits

Les principaux opérateurs sur les bits sont résumés dans le tableau suivant.

Condition	Signification	Condition	Signification
a & b	a 'et' b	a >> n	a décalé à droite de n bits
a | b	a 'ou' b	a << n	a décalé à gauche de n bits
a ^ b	a 'ou exclusif' b		

Ces opérateurs sont en général utilisés avec des variables de type entier ou des octets stockés dans des variables char. Ils agissent sur la forme binaire du nombre et sont souvent utilisés pour gérer des bits ou les tester séparément. Par exemple, s'il faut lire l'état de plusieurs drapeaux, ces derniers peuvent être stockés dans un seul entier.

En reprenant le tutoriel 2, il est possible d'afficher les valeurs décimales de chaque bit avec le programme ci-dessous :

```
#include <stdio.h>
int main() {
  int bit = 0, i = 1;
  while(i>0) { /* Boucle jusqu'à ce que le bit de signe soit fixé */
    printf(" pow(2,%2d) = %11d\n",bit,i);
    i = i<<1; /* Décale la valeur de i vers la gauche d'une position. */
    bit++; /* Incrémente le compteur. */
  }
  return 0; /* Retourne succès au système d'exploitation. */
}
```

Dans cet exemple, la valeur enregistrée dans la variable `i` est décalée d'une position vers la gauche. L'opérateur de décalage vers la gauche a pour effet de déplacer tous les bits de la variable `i` d'une position vers la gauche. Si un bit est déplacé en dehors de l'emplacement de la variable en mémoire, le bit est perdu. Dans ce cas, `i` contient seulement un. De plus, l'action de l'opérateur décalage à gauche est d'effectuer un passage à la puissance de deux suivante. Quand le bit de la variable `i` atteint le bit de signe, le nombre devient négatif ce qui provoque l'arrêt de la boucle while.

L'opérateur & est très utile pour tester si un bit est à un ou non. Combiné avec les opérateurs de décalages à gauche ou à droite, cela permet de tester chaque bit d'une variable entière :

```
#include <stdio.h>
int main() {
  char str[33]; /* Déclare un tableau de caractères pour conserver la sortie. */
  int bit, i = 235643; /* Déclare un nombre à convertir en binaire. */
  for(bit=31;bit>0;bit--) { /* Boucle de la gauche vers la droite */
    if(((1<<bit) & i) == 0) str[31-bit] = '0'; /* Faux */
    else str[31-bit] = '1'; /* Vrai */
  }
  str[32]='\0'; /* Ajout du terminateur de chaîne */
  printf("%d (décimal) = %s (binaire)\n", i, str);
  return 0; /* Retourne succès au système d'exploitation. */
}
```

Dans ce programme d'exemple, chaque caractère du tableau est défini selon la valeur binaire. Ensuite, pour compléter la chaîne de caractères, le terminateur de chaîne est ajouté. Enfin, la forme binaire du nombre entier est affichée.

Commandes système

Il peut être utile de pouvoir lancer des commandes shell ou d'autres programmes sans faire directement d'édition de lien avec la bibliothèque correspondante. Cela est réalisé grâce à la fonction `system` :

```
#include <stdlib.h>
int main() {
  system("ls ./"); /* Liste les fichiers du répertoire actuel. */
  return 0; /* Retourne succès au système d'exploitation. */
}
```

La fonction `system` évalue la chaîne passée en paramètre comme si elle avait été tapée en ligne de commande. La sortie standard de la commande n'est pas capturée par le programme mais elle est par contre envoyée sur l'écran.

La sortie standard d'une commande système ou d'un programme peut être capturée grâce à un tube (pipe en anglais). Les tubes ont la même syntaxe que les fonctions sur les fichiers traditionnels, autorisant lecture, écriture et connexions bidirectionnelles. Par exemple, le contenu du répertoire courant peut être lu dans un programme en utilisant :

```
#include <stdio.h>
int main() {
  int c;
  FILE *ptr = 0; /* Crée un pointeur FILE nul */
  ptr = popen("ls ./", "r"); /* Liste le contenu du dossier et se met en écoute */
  if(!ptr) return 1; /* Renvoie échec si la commande échoue. */
  while((c=fgetc(ptr)) != EOF) { /* Lit chaque caractère. */
    printf("%c",(char)c); /* Affiche les caractères. */
  }
  pclose(ptr); /* Ferme le tube */
  return 0; /* Renvoie succès au système d'exploitation. */
}
```

Dans cet exemple, chaque nom de fichier retourné est disponible dans le programme.

Toute commande qui peut être saisie en ligne de commande peut être exécutée avec `system` ou `popen`. Au lieu de simplement appeler des commandes shell de base, il est possible d'en utiliser pour tracer des données avec gnuplot :

```
#include <stdlib.h>
#include <stdio.h>
int main(int argc, char *argv[]) {
  int x_min = 0, x_max = 4; /* Fixe l'étendue du tracé. */
  char commandStr[100], systemCmd[200];
  if(argc < 2) {
    printf("Utilisation %s <fonction>\n", argv[0]); /* Un argument requis.*/
    return 1; /* Retourne une erreur. */
  }
  /* Construit la commande en deux étapes pour montrer ce qui se passe. */
  sprintf(commandStr, "plot [x=%d:%d] %s(x)", x_min, x_max, argv[1]);

  /* Lance la commande afin que gnuplot reste ouvert. */
  sprintf(systemCmd,"echo \"%s\" | gnuplot --persist",commandStr);
  system(systemCmd); /* Demande à gnuplot de faire le tracé. */
  return 0; /* Retourne succès au système d'exploitation. */
}
```

Avant d'essayer cet exemple, il faut installer gnuplot en tapant :

```
sudo apt-get gnuplot-x11
```

Ensuite, une fois le programme compilé, lancez : `./gplot sin`. Le paramètre `--persist` permet à la fenêtre de gnuplot de rester ouverte après la fin du programme. Davantage d'informations sur le programme gnuplot sont disponibles sur : http://www.gnuplot.info/

Surveiller un système LINUX

Il existe plusieurs fonctions utiles disponibles sous Linux, mais elles ne sont pas implémentées de la même façon sur les autres systèmes d'exploitation. Par exemple, l'état de la mémoire peut être obtenu grâce à `sysinfo :`

```
#include <stdio.h>
#include <sys/sysinfo.h>
int main() {
  struct sysinfo info; /* Crée une instance sysinfo pour stocker le résultat. */
  sysinfo(&info); /* Récupère les informations sur le système */
  printf("Mémoire utilisée = %d\n",info.totalram - info.freeram);
  return 0; /* Retourne succès au système d'exploitation. */
}
```

où sys/sysinfo.h est disponible sous LINUX, mais pas sous OSX ni MS Windows. Avant que les informations système ne puissent être récupérées, une variable struct de type sysinfo doit être créée. Ce n'est pas une variable simple, mais une structure contenant plusieurs variables. L'accès aux variables membres du struct se fait grâce à l'opérateur ".". Quand sysinfo est appelée, l'adresse de la variable struct de type systinfo est passée à la fonction. Cette dernière remplit alors les membres du struct avec le statut.

Dans le dernier exemple de ce tutoriel, gnuplot est utilisé pour tracer l'utilisation de la mémoire en fonction du temps :

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/sysinfo.h>
int main() {
  int i, ramUsed;
  char gnuplotCmd[250], systemCmd[350];
  FILE *outPtr = 0;
  char fileName[50];
  sprintf(fileName,"data.txt"); /* Le nom du fichier de sortie. */
  struct sysinfo info; /* Une structure sysinfo pour stocker l'état. */
  outPtr = fopen(fileName,"w"); /* Ouvre le fichier de sortie. */
  if(!outPtr) return 1; /* Renvoie une erreur si l'ouverture a échoué */
  for(i=0;i<60;i++) {
    sysinfo(&info);  /* Récupère l'information système */
    ramUsed = info.totalram - info.freeram;
    fprintf(outPtr,"%d %d\n", i, ramUsed); /* Écrit la mémoire utilisée. */
    usleep(500000); /* Suspend pendant 1/2 seconde. */
  }
  fclose(outPtr); /* Ferme le fichier de sortie. */

  /* Trace maintenant les données */
  sprintf(gnuplotCmd, "plot \'%s\'\n", fileName); /* Crée la commande de traçage */

  /* Crée la commande complète, y compris le tube vers gnuplot */
  sprintf(systemCmd,"echo \"%s\" | gnuplot --persist",gnuplotCmd);

  system(systemCmd); /* Exécute la commande système. */
  return 0; /* Retourne succès au système. */
}
```

où sys/sysinfo.h est disponible sous LINUX et unistd.h sous LINUX ou OSX. Le programme écrit l'utilisation mémoire dans un fichier de sortie chaque demi-seconde. Ensuite, gnuplot est lancé pour tracer l'utilisation mémoire comme une fonction temporelle.

Le défi

Modifier l'exemple précédent pour écrire un fichier de sortie en utilisant la valeur de retour de la commande hostname pour former le nom du fichier. Tracer ensuite la mémoire utilisée et la charge système pendant l'exécution d'un ou plusieurs autres programmes. Le membre loads[3] de la structure sysinfo contient les charges moyennes sur une, cinq et quinze minutes. Essayez d'utiliser :

```
fprintf(outPtr,"%d %f %d\n", i, ramUsed/10240.0, info.loads[0]);
```

pour écrire le fichier de données. Tracez ensuite les valeurs avec les deux lignes :

```
sprintf(gnuplotCmdOne, "plot \'%s\' using 1:2 title \'%s\'", fileName, "Ram used");
sprintf(gnuplotCmdTwo, ", \'%s\' using 1:3 title \'%s\'\n", fileName, "Load");

/* Création de la commande complète, y compris le tube vers gnuplot */
sprintf(systemCmd,"echo \"%s%s\" | gnuplot -persist",gnuplotCmdOne,gnuplotCmdTwo);
```

La solution du problème sera donnée la prochaine fois.

Article de W. H. Bell

THE SCRATCH PATCH

L'algorithme du tri à bulles

Pour ce mois, j'ai pensé à quelque chose d'un peu différent : un algorithme.

Cela consiste à réaliser une tâche en exécutant des instructions pas à pas. Vous en avez déjà utilisé un si vous avez appris à multiplier avec la "méthode de quadrillage". En suivant bien les étapes, vous obtiendrez le bon résultat.

Quand nous écrivons un programme, nous voulons un algorithme aussi rapide que possible mais qui utilise le minimum de mémoire. Il doit aussi produire le résultat correct, bien sûr !

Comme d'habitude, si vous avez du mal, le projet est téléchargeable sur : http://scratch.mit.edu/forums/

Mon nom d'utilisateur est "racypy".

Le tri à bulles est un algorithme pour trier des listes de nombres.

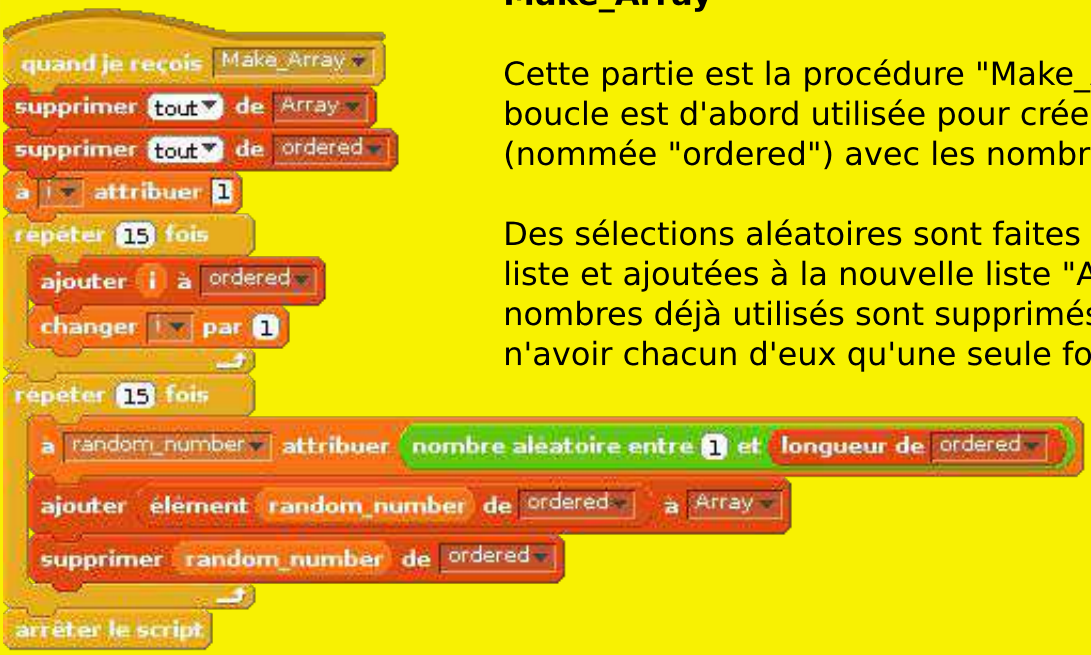
Ce n'est pas la méthode la plus efficace : elle peut être relativement lente avec des listes qui sont vraiment en désordre.

Cependant, elle est très rapide quand les nombres de la liste sont pour la plupart dans le bon ordre.

Voici le début du programme. Il annonce simplement ce qu'il est et appelle ensuite deux processus - "Make_Array" et "Bubble_Sort".

Origine :
Les algorithmes tiennent leur nom d'un mathématicien Perse :

Al-Khwārizmī (c.780 – c. 850).



Make_Array

Cette partie est la procédure "Make_Array". Une boucle est d'abord utilisée pour créer une liste (nommée "ordered") avec les nombres 1 à 15.

Des sélections aléatoires sont faites dans cette liste et ajoutées à la nouvelle liste "Array". Les nombres déjà utilisés sont supprimés afin de n'avoir chacun d'eux qu'une seule fois.

Bubble_Sort

Ici nous bouclons sur le tableau autant de fois qu'il y a de nombres dans la liste**.**

À l'intérieur de cette boucle, une autre boucle parcourt la liste afin de voir si un nombre est supérieur à celui qui lui succède.

Si c'est le cas, les deux nombres sont permutés.

En cours d'exécution, vous pouvez voir les nombres être échangés jusqu'à ce que la liste soit entièrement triée.

Si vous appréciez ça, vous pourriez vouloir modifier le programme afin qu'il trie des nombres fournis par l'utilisateur au lieu d'utiliser la liste aléatoire.

Scratch On!

Quand on programme, il est parfois utile de lire et écrire dans un fichier texte externe, ou à partir de celui-ci. Le premier exemple montre comment utiliser python pour créer une page web html.

Le second programme affiche des titres à partir des données extraites d'un fichier texte externe.

```
# HTML Writer

# Par Jaseman - 16 septembre 2012

import os

# Crée un fichier et l'ouvre en écriture (w)
f = open('/home/pi/test.html', 'w')

# Écrit des lignes de code dans le fichier
# Note: éviter les guillemets doubles " utiliser ' à la place
f.write("<html>"+"\n")
f.write("<head>"+"\n")
f.write("<title>Une Page Web Créée en Python</title>"+"\n")
f.write("</head>"+"\n")
f.write("<body bgcolor='#ffffdd'>"+"\n")
f.write("<font face='verdana' color='#000000'>"+"\n")
f.write("<center>"+"\n")
f.write("<h1>LE TITRE</h1><p>"+"\n")
f.write("<hr>"+"\n")
f.write("</center>"+"\n")
f.write("<h3>Un sous-titre</h3><p>"+"\n")
f.write("Ceci est le texte du premier paragraphe.<p>"+"\n")
f.write("<hr>"+"\n")
f.write("<center>"+"\n")
f.write("<font size='2'>"+"\n")
f.write("<b><a href='mailto:editor@themagpi.com'>COURRIEL</a></b><p>"+"\n")
f.write("<b><a href='http:www.themagpi.com'>SITE WEB</a></b><p>"+"\n")
f.write("</body>"+"\n")
f.write("</html>")

# Ferme le fichier
f.close()

# Ouvre le fichier html avec le navigateur Midori
os.system("midori /home/pi/test.html")
```

Ce programme a été écrit pour Raspbian Wheezy mais il peut être porté sous Windows en modifiant le chemin d'accès au fichier et le nom du navigateur dans l'appel os.system.

PYTHON VERSION: 2.7.3rc2
PYGAME VERSION: 1.9.2a0
O.S.: Debian 7

TESTED!

Lancez d'abord Leafpad. Tapez le texte de l'encadré de droite et enregistrez le fichier sous le nom "settings.txt" dans le dossier où votre code python sera enregistré.

```
screen width:1024
screen height:600
window caption:Titre fondant
text size:100
title 1:Jaseman présente...
title 2:Une production du Python Pit
title 3:Démo de titre fondant
```

```
# Import de paramètres

# Par Jaseman - 22 septembre 2012

f = open('settings.txt', 'r') # Ouvre un fichier en mode lecture (r)
settings = [] # Crée une variable de type tableau

for line in f: # Boucle pour récupérer chaque ligne du fichier
      settings.append(line)

f.close() # Ferme le fichier

# Utilise le deux-point (:) pour diviser les lignes
screenx=settings[0].split(':');screeny=settings[1].split(':')
windowcaption=settings[2].split(':');textsize=settings[3].split(':')
title1=settings[4].split(':');title2=settings[5].split(':')
title3=settings[6].split(':')

import os,pygame; from pygame.locals import *; pygame.init()
os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'

pygame.display.set_caption(windowcaption[1].strip())
screen=pygame.display.set_mode([int(screenx[1]),int(screeny[1])],0,32)
fadesurf=pygame.Surface((int(screenx[1]),int(screeny[1])))
titlesurf=pygame.Surface((int(screenx[1]),int(screeny[1])))
nexttitle=1;run=1

while run==1:
      # Affiche le titre suivant
      font = pygame.font.Font(None,int(textsize[1]))
      if nexttitle==1:
            text = font.render(title1[1].strip(),True,(255,255,255))
      if nexttitle==2:
            text = font.render(title2[1].strip(),True,(255,255,255))
      if nexttitle==3:
            text = font.render(title3[1].strip(),True,(255,255,255))
      tgr=text.get_rect
      tp=tgr(centerx=screen.get_width()/2,centery=screen.get_height()/2)
      titlesurf.blit(text,tp)
      # Augmente la transparence de fadesurf
      for t in range(255,0,-20):
            fadesurf.set_alpha(t); screen.blit(titlesurf,(0,0))
            screen.blit(fadesurf,(0,0)); pygame.display.update()
      # Diminue la transparence de fadesurf
      for t in range(0,256,20):
            fadesurf.set_alpha(t); screen.blit(titlesurf,(0,0))
            screen.blit(fadesurf,(0,0)); pygame.display.update()
      titlesurf.fill((0,0,0)); screen.blit(fadesurf,(0,0))
      pygame.display.update()
      nexttitle+=1
      if nexttitle>=4: nexttitle=1
```

Essayez de modifier les valeurs du fichier "settings.txt" et relancez ensuite le programme python. Grâce à cette méthode, il est possible de changer la façon dont fonctionne le programme sans avoir à modifier le code python lui-même.

Réactions

Je suis Faizal de Malaisie. Je viens de commander un Raspberry Pi et j'attends sa réception. J'ai vu le lien vers votre magazine aujourd'hui.Beau travail mais ça sera génial s'il était possible d'avoir plus d'illustrations (plus de photos).

Faizal

Votre magazine est super. Je l'utilise pour apprendre Python à mes gamins. Une suggestion éventuelle, serait-il possible de produire une version .mobi ou .epub, ainsi cela pourrait économiser du papier, et je pourrai tous les stocker sur le Kindle pour les lire et les copier ?

Adam

Je possède un Pi et je n'ai découvert le magazine que maintenant. J'ai prévu de commencer par le numéro 1 mais je l'aurais trouvé plus tôt s'il avait été sur Zinio (http://za.zinio.com/). J'utilise cette application Android pour lire tous mes magazines. Par conséquent, si vous pouviez le publier là-bas, cela serait génial, et je pourrais récupérer les mises à jour au moment où le numéro suivant paraît.

Q

Toujours un aussi grand magazine. Juste une question : pourquoi avoir arrêté les articles sur le Skutter, c'est le second mois où il n'y a rien le concernant ? J'espère que ce n'est pas le cas, je suis dans l'attente du retour de ces articles.

Andy

Bien que j'aime lire le MagPi, je tiens à pouvoir le faire sur mon Kindle, comme vous l'offrez déjà sous forme de téléchargement électronique. La disposition en deux colonnes rend la lecture difficile sur le Kindle. Serait-il possible de prendre en compte l'une de ces deux options :

1. Concevoir le MagPi pour des demi-pages d'une colonne en PDF
2. Créer une version pour la boutique Kindle d'Amazon

Kirill

Si seulement ce magazine avait existé dans les premiers jours du Sinclair ZX80, ma vie aurait été beaucoup plus facile. Il est prévu que mon Raspberry Pi soit livré pour Noël mais j'ai votre excellent magazine pour me tenir informé et susciter mon intérêt jusqu'à ce moment. Un grand merci pour un magazine excellent et innovant.

Ron

Juste un merci, et pour que vous le sachiez, je paierais pour ça.

Newell

editor@themagpi.com