LinuxFocus article number 286
http://linuxfocus.org

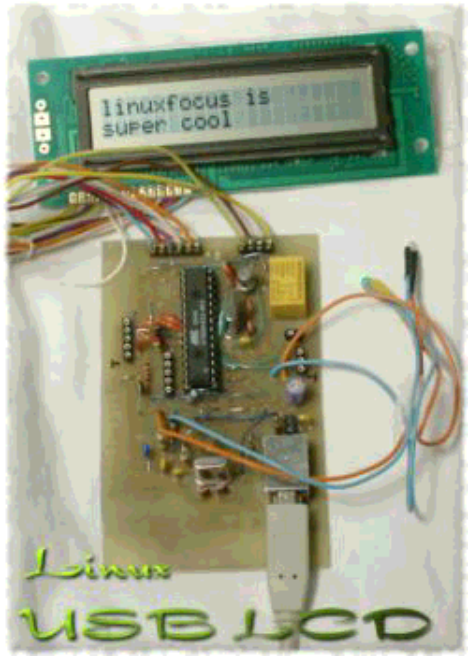
Un afficheur LCD USB sous Linux avec un chien de garde et des boutons

par Guido Socher (homepage)

L´auteur:

Guido aime Linux parce que c'est un paradis pour ceux qui veulent développer leur propre logiciel ou matériel.

Traduit en Français par:
John Perr
<johnerr/at/linuxfocus.org>



Résumé:

Cet article est le résultat de retours très positifs reçus suite aux autres articles traitant d'électronique que j'ai écrits. Vous, lecteurs de LinuxFocus, êtes vraiment un public fantastique! Certains d'entre vous voulaient savoir comment interfacer le bus USB. Voici donc une solution élégante. Nous utilisons l'afficheur LCD de l' article de Mai 2002 que nous ferons fonctionner avec le bus USB. L'ensemble est alimenté à partir du bus USB ce qui rend inutile toute alimentation supplémentaire.

Pour cet article, vous avez besoin d'au moins une installation partielle de l'environnement de développement AVR pour Linux. Son installation est décrite dans l'article: Programmer le micro-contrôleur AVR avec GCC.

Introduction

L'USB est pratique parce que c'est une interface moderne et qu'il offre la possibilité d'alimenter les équipements directement depuis le bus. Les connecteurs sont petits et un petit câble peut transporter une grand volume de données. Voila pour les avantages de l'USB. L'inconvénient réside dans la difficulté de réalisation matérielle à cause des hautes fréquences et du protocole plutôt complexe. Jetez simplement un oeil aux spécifications (http://www.usb.org/developers/, Regarder les spécifications 1.1) et vous serez impressionnés. Elles font 327 pages et sont difficiles à comprendre. Cela explique pourquoi il existe autant d'implantations défectueuses des équipements USB. Une introduction plus abordable se trouve sur http://www.beyondlogic.org/ mais la spécification est toujours aussi complexe.

Que faire? Comment interfacer notre micro-contrôleur au bus USB? Une entreprise écossaise, FTDI a une solution (http://www.ftdichip.com). Ils proposent le circuit FT232BM qui implante le bus USB d'un coté et une interface série RS232 de l'autre. En d'autres termes, il suffit de remplacer le MAX232 qui servait auparavant à l'adaptation de niveau sur les lignes RS232 par le circuit FT232BM et le tour est joué.

Le pilote

Le FT232BM est une véritable solution multi plate-forme. Les pilotes sont disponibles pour plusieurs systèmes d'exploitation. Le module du noyau s'appelle ftdi_sio et il est "open source". Il fait partie du noyau Linux standard. Le FT232BM offre plus qu'une simple interface USB-RS232 et le module du noyau Linux est toujours en développement afin d'en intégrer toutes les fonctions. La partie USB vers RS232 est toutefois prête et j'ai pu, par exemple, utiliser un noyau standard Redhat 7.3 (2.4.18) sans le recompiler ni le modifier. Il suffit de le brancher.

ftdi_sio est développé sur http://ftdi-usb-sio.sourceforge.net/.

Avec ma Redhat 7.3, tous les modules se chargent automatiquement quand je branche le connecteur USB. Si cela ne marche pas avec votre distribution, vérifiez que vous avez les modules suivants(pour USB-UHCI):

/sbin/lsmod usb-uhci
/sbin/lsmod usbcore
/sbin/lsmod usbserial
/sbin/lsmod ftdi_sio

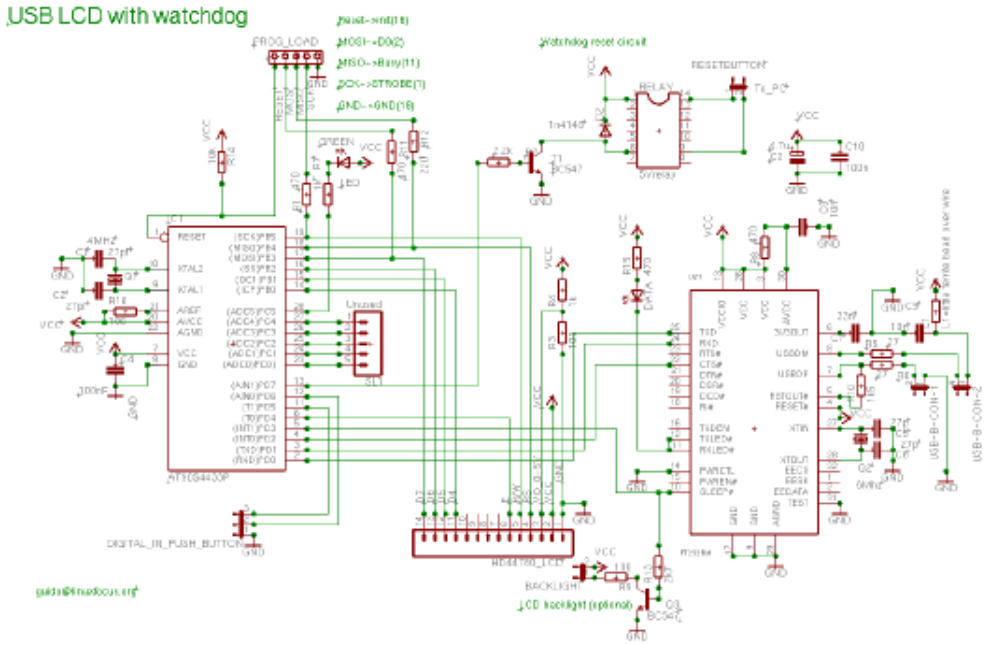
Le fichier device qui communique avec le matériel est /dev/ttyUSB0

Les développeurs de ftdi_sio recommendent au moins un Kernel 2.4.20 mais vous pouvez constater qu'un 2.4.18 fonctionne aussi (au moins pour les fonctions dont nous avons besoin ici).

Le schéma

Le circuit est simple. Le FT232BM est inséré entre les lignes Rx/Tx du micro-contrôleur et le connecteur USB. Un quartz de 6 Mhz et quelques autres composants décrits dans la note d'application de FTDI sont requis. Le pot de ferrite (sur le schéma à droite) est une petite bobine qui filtre les hautes

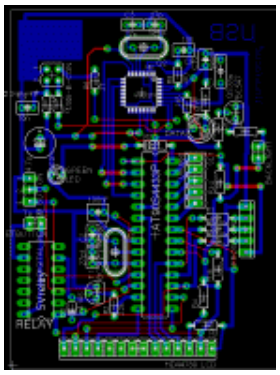
fréquences (l'USB est cadencé à 48Mhz). Vous pouvez aussi enrouler 10 tours de fil fin autour d'une résistance de 1K et l'utiliser comme bobine.



Il est nécessaire de surveiller la consommation de l'alimentation. Elle doit être inférieure à 100mA pour ce type de montage alimenté par le bus USB et le mode "suspend" de l'USB doit être supporté. Dans ce dernier cas, quand la broche "sleep" du FT232BM passe au niveau bas, la consommation doit être inférieure à 0,5mA. Cette dernière exigence est très contraignante. L'AVR dispose d'un mode "idle" dans lequel il consomme moins de 2mA et d'un mode "power down" où cette consommation descend à 20uA. Etant donné qu'il est plus facile de réveiller le micro-contrôleur à partir du mode "idle", j'ai décidé de déroger un peu à la spécification USB. L'éclairage optionnel arrière de l'afficheur sera éteint et l'ensemble du circuit consommera ainsi 3mA. C'est plus que 0,5mA mais le circuit hôte USB n'est pas capable de mesurer assez précisément un courant pour le détecter. Ca devrait fonctionner.

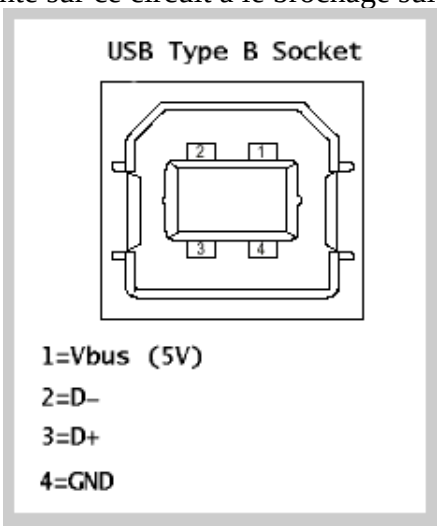
Ceci étant dit, je dois avouer que je n'ai pas d'ordinateur qui supporte le mode "suspend". Je n'ai donc pas pu tester ce mode. Si vous avez un ordinateur qui le supporte, probablement un portable moderne, alors je vous invite à faire l'essai et à m'informer du résultat.

Le reste du montage est presque identique à celui présenté dans l'article de mai 2002. Je n'entrerai donc pas plus dans les détails.
Vous pouvez cliquer sur l'image pour l'agrandir. Les fichiers eagle sont inclus dans le paquet qui contient le logiciel. Il est téléchargeable à partir d'un lien à la fin de cet article.



La platine est conçue en simple face et seule la couche bleue doit être gravée. Les lignes rouges sont des ponts de câblage.

Le connecteur USB de Type-B monté sur ce circuit a le brochage suivant:

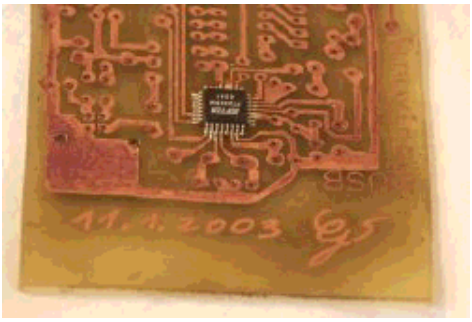


Travailler avec des composants CMS

Les composants CMS ont de bonnes caractéristiques mécaniques et électriques mais sont un véritable cauchemar pour les électroniciens amateurs. Vous devez vraiment être doué pour la soudure et au moins la partie dédiée au composant CMS doit être dessinée très proprement. Autrement dit, ce circuit n’est pas destiné aux débutants. Regardez la liste des alternatives ci-dessous si vous n’êtes pas certain de pouvoir dessiner le circuit ni souder le circuit intégré.

Soudez le circuit CMS sur la platine en premier.
Pour cette opération, il faut déposer un peu de soudure sur les pistes qui reçoivent le circuit puis déposer un fin film de pâte à souder spéciale CMS (certains l’appelle soudure de miel car elle ressemble à du miel). Il existe aussi une société Allemande appelée "Kontakt Chemie" qui fabrique un vernis en bombe du nom de "Lötlack".Il est possible d’utiliser le "Lötlack" en lieu et place de cette soudure "miel" si vous préférez.

Nettoyez votre fer à souder. Il ne doit pas rester de soudure sur la panne. Positionnez ensuite le FT232BM très exactement. Appuyez doucement sur chaque patte de la puce avec votre fer à souder sans ajouter de soudure.



Ce procédé fonctionne parfaitement. La taille du fer à souder importe peu. Utilisez un fer ordinaire et assurez-vous de la propreté de la panne avant de toucher les pattes de la puce. Je déconseille l'utilisation de grille-pain ou toute autre méthode barbare qui risquerait fortement d'endommager la puce.

Les essais

Je suggère de procéder en deux étapes. Connectez d'abord le circuit sans insérer le micro-contrôleur AVR dans son support. Linux devrait reconnaître la puce FTDI et ainsi la faire apparaître dans /proc/bus/usb/devices:

```
T: Bus=02 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#= 2 Spd=12 MxCh= 0
D: Ver= 1.10 Cls=00(>ifc ) Sub=00 Prot=00 MxPS= 8 #Cfgs= 1
P: Vendor=0403 ProdID=6001 Rev= 2.00
S: Manufacturer=FTDI
S: Product=USB <-> Serial
C:* #Ifs= 1 Cfg#= 1 Atr=80 MxPwr= 90mA
I: If#= 0 Alt= 0 #EPs= 2 Cls=ff(vend.) Sub=ff Prot=ff Driver=serial
E: Ad=81(I) Atr=02(Bulk) MxPS= 64 Ivl= 0ms
E: Ad=02(O) Atr=02(Bulk) MxPS= 64 Ivl= 0ms
```

Insérez ensuite le micro-contrôleur AVR et chargez un programme de test qui fera clignoter la led. Décompressez le paquetage logiciel linuxusblcd (voir à la fin de cet article) et tapez :

```
make testload0
```

Le câble de programmation et le connecteur USB doivent tous les deux être branchés.
Si le test est positif, vous pouvez être certains que le micro-contrôleur fonctionne.

Après cela, vous pouvez charger le logiciel complet dans le micro-contrôleur:

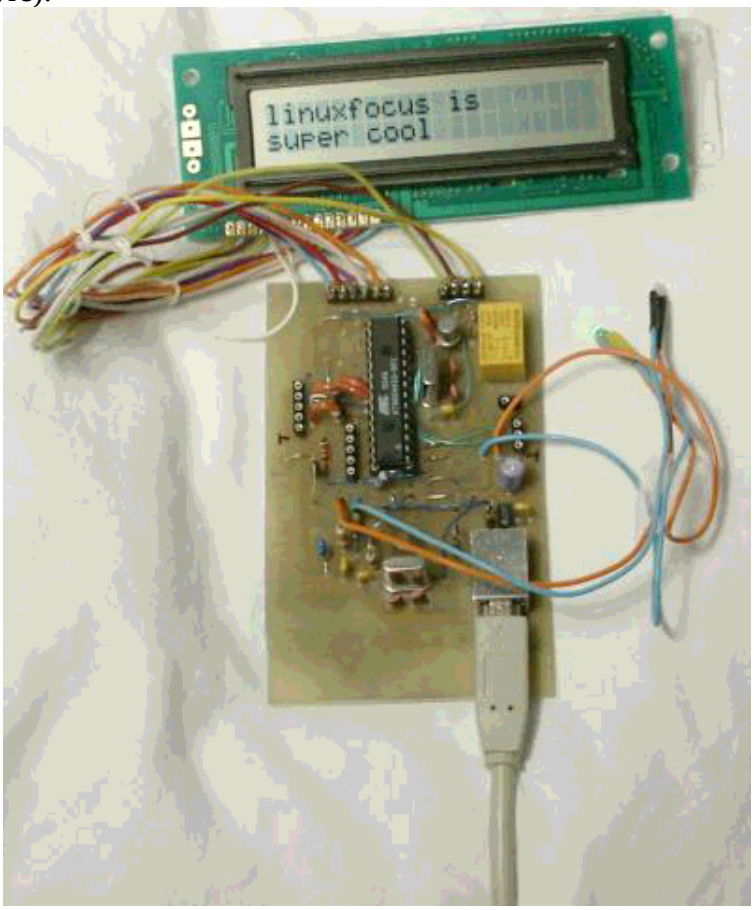
```
make load
```

Vous pouvez utiliser "ttydevinit /dev/ttyUSB0" pour initialiser la connection USB série ainsi que "cat > /dev/ttyUSB0" pour dialoguer avec le circuit.

```
ttydevinit /dev/ttyUSB0
cat > /dev/ttyUSB0
D=hello world
```

Ceci écrira "hello world" sur l'afficheur. Voir l'article de mai 2002 pour les détails. Le code de l'article de mai 2002 contient aussi un programme nommé llp.pl qui peut servir à dialoguer interactivement avec votre ordinateur en utilisant les deux boutons sur l'afficheur LCD. Il est réutilisable ici.

... et voici l’afficheur en fonctionnement (les boutons n’étaient pas branchés au moment de la photo,le FT232BM est coté cuivre):



Autres solutions

Bien que le circuit présenté ici soit simple, il n’est pas destiné aux débutants à cause des difficultés que posent la soudure de la puce CMS. Une solution commerciale toute prête peut alors être envisagée. L’inconvénient tient au fait que vous ne disposerez pas normalement du chien de garde, des leds et des boutons. Les prix sont raisonnables pour ces montages tout prêts. L’ensemble des composants utilisés dans cet article coûte environ 30 Euros et les afficheurs commerciaux sont dans le même ordre de prix.

Malheureusement, la plupart des afficheurs commerciaux utilisent leur propre identificateur même s’ils utilisent la puce FTDI. Cela veut dire que le module du noyau ne les reconnaîtra pas car le pilote USB est basé sur ces nombres. Vous aurez à modifier les sources du noyau et à le recompiler. De futures versions du noyau pourraient fonctionner si le code a été modifié par quelqu’un d’autre.

- http://www.matrixorbital.com/ Utilise aussi la puce FT232BM mais avec des IDs personnalisés. L’afficheur s’appelle LK202-24-USB.

- http://www.usblcd.de/ Cette solution dispose d’un module dédié pour le noyau qui est inclus dans le noyau standard Linux. Il fonctionnera directement avec tout noyau 2.4.x. Sûrement une

excellente solution.

- http://crystalfontz.com/ Leur afficheur USB (632 et 634) utilise le FT232AM avec des IDs personnalisés.

- http://www.cwlinux.com/eng/products/products_lcd.php J'ai découvert ce site récemment. Ils ont deux afficheurs LCD avec clavier mais le prix est le double de la solution de cet article.

Références

- Les logiciels et documents mentionnés dans cet article (les nouvelles versions du logiciel linuxlcdpanel apparaîtront sur cette page)

- Comment programmer le micro-contrôleur AVR: Programmer le micro-contrôleur AVR avec GCC, article de mars 2002

- L'article de mai 2002 avec linuxlcdpanel. Le programme Perl (appelé llp.pl) de cet article est réutilisable: Article de mai 2002

- Le site FTDI: www.ftdichip.com

- La fiche technique du FT232BM (vient de http://www.ftdichip.com): ftdichip_ds232b11.pdf, 820Kb

- Eagle pour Linux cadsoftusa.com

- Bibliothèque Eagle pour la puce FTDI (vient de http://www.elektronik-projekt.de) ftdi.lbr.gz

---

Site Web maintenu par l´équipe d´édition LinuxFocus © Guido Socher "some rights reserved" see linuxfocus.org/license/ http://www.LinuxFocus.org	Translation information: en --> -- : Guido Socher (homepage) en --> fr: John Perr <johnerr/at/linuxfocus.org>

2005-01-14, generated by lfparser_pdf version 2.51