CSEL1 (construction systèmes embarqués sous Linux) – Travail pratique

Programmation système : Système de fichiers

Contexte

La carte NanoPi NEO Plus2 est équipée de deux LEDS. LED rouge est en principe prévue pour indiquer si la cible est sous tension (power) et la LED verte est là pour donner des indications de statut (status). Avec la configuration actuelle du noyau Linux, ces deux LEDs sont à libre disposition des développeurs.

Travail à réaliser

Sur le site moodle vous trouverez une application qui contrôle la fréquence de clignotement d'une LED. Ce code n'a pas été très bien programmé et utilise le 100% d'un cœur du processeur (à mesurer avec top).

Concevez une application permettant de gérer la fréquence de clignotement de la LED « status » de la carte NanoPi à l'aide des trois boutons-poussoirs.

Quelques indications pour la réalisation de l'application :

1. Au démarrage la fréquence de clignotement sera réglée à 2Hz
2. Utilisation des boutons-poussoirs
   - « k1 » pour augmenter la fréquence
   - « k2 » pour remettre la fréquence à sa valeur initiale
   - « k3 » pour diminuer la fréquence
   - *optionnel : une pression continue exercée sur un bouton indiquera une auto incrémentation ou décrémentation de la fréquence.*
3. Tous les changements de fréquence seront logger avec syslog de Linux.
4. Le multiplexage des entrées/sorties devra être utilisé.

CSEL1 (construction systèmes embarqués sous Linux) – Travail pratique

Programmation système : Système de fichiers

Infos pratiques

- **GPIO**
  L'accès aux entrées/sorties est proposé sous le système de fichiers virtuels « `sysfs` », dans le répertoire « `/sys/class/gpio/*` ». Cette interface est décrite dans la documentation du noyau Linux (https://www.kernel.org/doc/Documentation/gpio/sysfs.txt).

  Quelques informations utiles pour l'utilisation du module GPIO sous Linux :

  - Pour lire ou écrire une valeur sur une pin d'entrée/sortie il faut tout d'abord configurer le GPIO, soit :
    1. Exporter la porte dans le « `sysfs` » à l'aide de la commande :
       ```
       # echo <pin_nr> > /sys/class/gpio/export
       ```
    2. Configurer la porte en entrée ou en sortie :
       ```
       # echo in  > /sys/class/gpio/gpio<pin_nr>/direction
       # echo out > /sys/class/gpio/gpio<pin_nr>/direction
       ```
    3. Lire l'état de la pin (input) :
       ```
       # cat /sys/class/gpio/gpio<pin_nr>/value
       ```
    4. Changer l'état de la pin (output) :
       ```
       # echo 0 > /sys/class/gpio/gpio<pin_nr>/value
       # echo 1 > /sys/class/gpio/gpio<pin_nr>/value
       ```
    5. Pour traiter les entrées/sorties en mode événementiel :
       ```
       # echo rising  > /sys/class/gpio/gpio<pin_nr>/edge
       # echo falling > /sys/class/gpio/gpio<pin_nr>/edge
       # echo both    > /sys/class/gpio/gpio<pin_nr>/edge
       ```

  - Pour obtenir le numéro du GPIO correspondant à un pin il faut lire la configuration sur le debugfs, soit :
    ```
    # mount -t debugfs none /sys/kernel/debug
    # cat /sys/kernel/debug/gpio
    ```

- **Timer**
  La librairie « timerfd » offre des services très intéressants pour la génération d'une horloge à haute résolution et surtout une interface permettant le multiplexage des entrées/sorties.