Big, Zora, Emilie et Bunny

Table des matières

1 Introduction

Ce document décrit les *runlevels* des serveurs Big, Zora, Emilie et Bunny.

Les actions ci-dessous seront exécuté préalablement sur chacun des châssis hébergeant un des quatre serveurs :

- Mise sous tension
- Chargement des modules Linux

  - Les drivers sont chargés au démarage par les scripts */etc/sysconfig/modules/*.modules*. L'extension ainsi que les droits d'execution sont r sont nécessaires.
  - L'emplacement des drivers chargés par `modprobe` est donné par les liens symboliques du répertoire */lib/modules/2.6.24.7-rt21-shl-3.2.3/hess2/*. Si l'on modifie ces liens, il faut alors ré-executer `# modprobe -a` afin de mettre à jour le fichier */lib/modules/2.6.24.7-rt21-shl-3.2.3/modules.dep*.

- Lancement du démon
  - Le démon est lancé de façon détachée d'un compte utilisateur par le script *rc.local*.

    ```
    ...
    /root/Bunny 2>/var/log/alertesBunny > /var/log/Bunny &
    ```

  - Le démon peut être relancé manuellement par le script */root/start*.sh*.

    ```
    #!/bin/bash
    nohup ./Bunny 2>/var/log/alertesBunny > /var/log/Bunny &
    ```

  Rq : On utilise `nohup` et les redirections (`>` et `2>`) pour détacher les exécutables de leurs terminaux. On choisi de se passer du démon SYSLOG par besoins de réactivité.

2 Automate générique

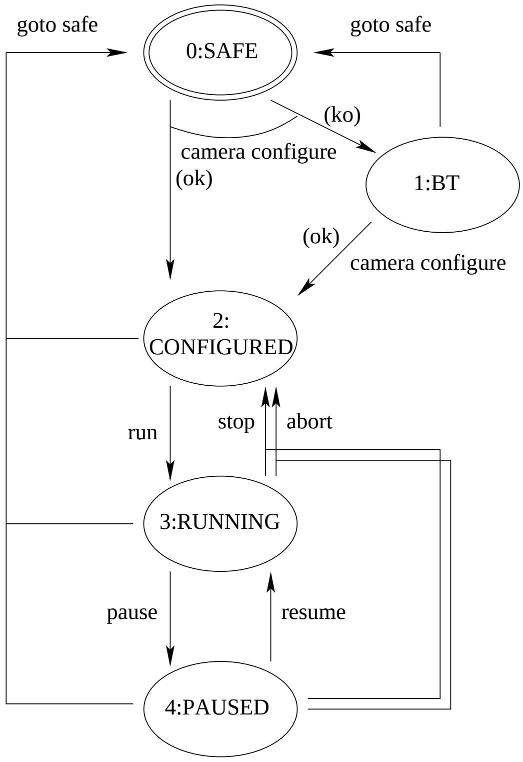
Remarque : Ces automates peuvent être désactivés via la ligne de commande ou via une socket :

```
# bunny -x

$ telnet camera17 1808
> BEGIN_
> RUNLEVEL GET EXPERT MODE
0: disabled
> RUNLEVEL ENABLE EXPERT MODE
> RUNLEVEL GET EXPERT MODE
1: enabled
> ENDMSG

# big

$ telnet camera12 1805
> BEGIN_
> RUNLEVEL GET EXPERT MODE
1: enable
> RUNLEVEL DISABLE EXPERT MODE
> RUNLEVEL GET EXPERT MODE
0: disabled
> ENDMSG
```



2.1 Runlevel : Statut

- 0 : Safe (état initial et état poubelle).
- 1 : Basses tensions ON (état intermédiaire).
- 2 : Configured.
- 3 : Running.
- 5 : Paused

2.2 Transitions et évènements associés

transitions	évènement
*->0	***gotosafe***
0|1->1|2	***configure***
2->3	***start***
3->2	***stop***
3->2	***abort***
3->4	***pause***
4->3	***resume***

Les transitions ne sont **jamais** commandées en interne mais résulte d'ordres envoyés par un tiers, excepté pour la transition ***gotosafe*** faite à la fin de l'initialisation de chacun des serveurs.

- Bunny et les 4 automates de Big sont commandés par la ferme
- Emilie et Zora sont commandés par Big

la réponse de l'automate aux solicitations de changement d'état se fait de manière asynchrone

3 Bunny

Bunny doit contrôler 4 relais via 4 de ses sorties GPIO. Ces quatre relais bloquent ou laissent circuler d'autres GPIO en provenance de Big :

- les GPIO **10, 11 et 12** relayent la commande d'alimentation hautes tensions des tiroirs
- la GPIO **20** relaye la commande d'ouverture du capot

Il faut atteindre l'état "configuré" pour que les 4 relais deviennent passant. En revanche, ils re-basculent **automatiquement et définitivement** à l'état bloquant lorsque :

- l'alimentation des ventilateurs n'est pas stable
- la température excède un certain seuil
- les photo-diodes sont excitées au delà d'un certain seuil

3.1 Initialisation de Bunny

- L'alimentation des basse tension des ventilateurs est mise à ON
- Les ventilateurs sont lancés à vitesse constante (`0x9f` pour les petits et `0xff` pour les 2 gros extracteurs sur les côtés)
- Autoriser toutes GPIO excepté LID et HV
- La transition ***gotosafe*** est réalisée
- Lancement des threads énumérés ci-dessous

3.2 Threads de Bunny

- *Task_Wiener.c*
  - comparaison du courant et de la tension aux valeurs nominales
  - éventuelle inhibition des GPIO HV et LID
  - mise à jour du statut des basses tensions
- *Task_Fan.c*
  - lecture des la vitesse des ventilateurs
- *Task_Photodiode.c*
  - lecture des photo-diodes en entrée et calcul des seuils
  - éventuelle inhibition des GPIO HV et LID
  - mise à jour du statut des photo-diodes
- *Task_GPIO.c*
  - lecture des GPIO propres à Bunny
  - positionnement des sortie GPIO en fonction des GPIO lues en entrées *(désactivé)*
- *Task_Temperature.c*
  - lecture des capteurs de températures (moyenne sur n valeurs : 5) et calcul des seuils
  - ajustement de la vitesse des ventilateurs en fonction des seuils de température *(désactivé)*

- positionnement des GPIO en fonction du maximum des seuils de température *(désactivé)*
- éventuelle inhibition des GPIO HV et LID
- mise à jour du statut des températures

3.3 Automate de Bunny

Évènement	tâches pré-requises au changement d'état
gotosafe	Bloquer les GPIO LID et HV *(fait)*
configure	Tester l'état non erroné des basses tensions *(fait)* Tester l'état non erroné des températures *(fait)* Tester l'état non excité des diodes-pin *(fait)* Autoriser les GPIO LID et HV *(fait)*
start	
stop	
abort	
pause	
resume	

exemple :

```
telnet camera17 1808
> BEGIN_
> RUNLEVEL TASK LIST
> RUNLEVEL GOTOSAFE
> RUNLEVEL CONFIGURE
> RUNLEVEL GET RUNLEVEL (questionnement par Big)
> RUNLEVEL GET STATUS   (questionnement par Big)
> ENDMSG
```

4 Big

Le serveur Big assume les tâches suivantes :

- Multi-FIFO. **(je ne comprend pas)**
- Monitoring.
- Ordres de haut niveau.

4.1 Initialisation de Big

- transition ***gotosafe*** de l'automate principal
- Lancement des threads énumérés ci-dessous

4.2 Threads de Big

- *Task_Camera_Status.c* : envoie de données de monitoring à la ferme
- *Task_Drawer_Monitoring.c* : envoie de données de monitoring à la ferme
- *Task_Wiener.c* : comparaison du courant et de la tension aux valeurs nominales. En cas d'erreur, envoyer un warning à la ferme *(à faire)*

4.3 Automate principal de Big

Évènement	tâches pré-requises au changement d'état
gotosafe	Transition ***gotosafe*** de l'automate du Emilie *(a faire)* Transition ***gotosafe*** de l'automate du Zora *(a faire)* Transition ***gotosafe*** de l'automate du capot *(fait)* Transition ***gotosafe*** de l'automate des triggers *(fait)* Transition ***gotosafe*** de l'automate des hautes tension *(fait)* Désactivation du thread Drawer_Monitoring *(à faire)* Basses tension mise à OFF *(fait)* Arrêt de l'envoi des données de monitoring à la ferme *(à faire)*
configure	Vérifier la température des DAQ via Bunny (runlevel >= 1 & statut correct) *(fait)* Basses tension mise à ON si OFF *(fait)* Broadcast sur SLC ready *(à faire)* Broadcast sur DAQ ready *(à faire)* Vérifier la cohérence des données envoyées par la ferme *(à faire)* Tester la réponse de Zora aux trigers soft *(à faire)* Tester la réponse d'Emilie *(à faire)* Paramétrage SAM *(à faire)* Envoi des paramètres SLC, DAQ et SAM à Emilie et Zora *(à faire)* Envoie de l'évènement ***configure*** à Emilie et Zora *(à faire)* Activation du thread Drawer_Monitoring *(à faire)* **DRAWER FF SET_CNTRL_SLC**
start	Vérifier les données envoyées par le "star controller" *(à faire)* Paramétrage NODES *(à faire)* Envoi des paramètres des NODES à Emilie et Zora *(à faire)* Envoi d'un "run header" aux "data receiver" *(à faire)* Envoie de l'évènement ***start*** à Emilie et Zora *(à faire)*
stop	Envoie de l'évènement ***stop*** à Emilie et Zora *(à faire)* Sans réponse d'Emilie ou de Zora, Big s'auto envoi ***abort*** *(à faire)* Envoi d'un "run tailer" aux "data receiver" *(à faire)*
abort	Tue puis relance Emilie et Zora via un autre serveur embarqué sur leur carte *(à faire)*
pause	Envoie de l'évènement ***pause*** à Emilie et Zora *(à faire)*
resume	Envoie de l'évènement ***resume*** à Emilie et Zora *(à faire)*
kill	Sortie du programme *(à faire)*

rq : Big possède la transition supplémentaire *kill*.
exemple :

```
telnet camera16 1805
> BEGIN_
> RUNLEVEL MAIN GOTOSAFE
> RUNLEVEL MAIN CONFIGURE
> RUNLEVEL TASK LIST
> ENDMSG
```

5 Zora *(à faire)*

Le serveur Zora assume les tâches suivantes :

- Lecture des données depuis le bus DMA.
- Timestamps via le **'local module'**.
- Envoi des données à la ferme.

Ce serveur est piloté via un sous automate de Big.

5.1 Initialisation de Zora

- ?

5.2 Automate de Zora

Évènement	tâches pré-requises au changement d'état
gotosafe	
configure	**DRAWER FF SET_CNTRL_SAM_REGISTER** **DRAWER FF SET_CNTRL_SAM_ND** **DRAWER FF SET_CNTRL_DAQ**
start	Envoi des données à la ferme *(à faire)*
stop	Arrêt de l'envoi des données à la ferme *(à faire)*
abort	
pause	Arrêt de l'envoi des données à la ferme *(à faire)*
resume	Envoi des données à la ferme *(à faire)*

6 Emilie *(à faire)*

Le serveur Emilie assume les tâches suivantes :

- Lecture des cartes triggers.
- Choix d'une stratégie de triggers (Soft, Interne ou Externe)
- Timestamps via le **'local module'**.

Ce serveur est piloté via un sous automate de Big.

6.1 Initialisation d'Emilie

- ?

6.2 Automate d'Emilie

Évènement	tâches pré-requises au changement d'état
gotosafe	
configure	**DRAWER FF SET_CNTRL_TRIGGER** **DRAWER FF SET_CNTRL_L2**
start	Envoi des données à la ferme *(à faire)*
stop	Arrêt de l'envoi des données à la ferme *(à faire)*
abort	
pause	Arrêt de l'envoi des données à la ferme *(à faire)*
resume	Envoi des données à la ferme *(à faire)*

7 Automate des hautes tensions

Évènement	tâches pré-requises au changement d'état
gotosafe	Hautes tension mise à OFF *(à faire)*
configure	Hautes tension mise à ON *(à faire)*
start	**DRAWER FF SET_VOLTAGE_VALUE**
stop	
abort	
pause	
resume	

exemple :

```
telnet camera16 1805
> BEGIN_
> RUNLEVEL HV GOTOSAFE
> ENDMSG
```

8 Automate des triggers

Évènement	tâches pré-requises au changement d'état
gotosafe	Désactivation des triggers *(à faire)*
configure	Activation des triggers *(à faire)*
start	
stop	
abort	
pause	
resume	

exemple :

```
telnet camera16 1805
> BEGIN_
> RUNLEVEL TRG GOTOSAFE
> ENDMSG
```

9 Automate du capot

Évènement	tâches pré-requises au changement d'état
gotosafe	Fermeture du capot *(à faire)*
configure	Ouverture du capot *(à faire)*
start	
stop	
abort	
pause	
resume	

exemple :

```
telnet camera16 1805
> BEGIN_
> RUNLEVEL LID GOTOSAFE
> ENDMSG
```

10 Todo list

10.1 Gestion des serveurs

- Créer un watchdog soft sur la carte slow contrôle qui lance Big si ce dernier n'est pas en cours d'executtion.
- Créer 2 démons capables de tuer puis relancer Emilie et Zora sur commande de Big (évènement abort)

10.2 Remonté des erreur vers l'opérateur

Utiliser les primitives suivantes pour afficher les messages. On pourra facilement éliminer ces sortie via des `#define`.

```
int STATUSMessage(char *message);
int DBUGMessage(char *message);
int INFOMessage(unsigned short int errorid,char *message,...)
int CAUTIMessage(unsigned short int errorid,char *message,...)
int WARNMessage(unsigned short int errorid,char *message,...)
int ERRORMessage(unsigned short int errorid,char *message,...)
```

- Gestion des erreurs atteignant le controleur caméra :
  - INFO
  - CAUTION
  - WARNING
  - ERROR bloque le controleur caméra.
- Les codes de retour (un code par erreur) sont définit dans le fichier *Driver/include/Message.h*
- Le statut est renvoyé à chaque solicitation des automates.

10.3 Basses tensions

Actions à effectuer en conséquences aux instabilités :

- Bunny : positionnement d'un flag interrogeable par Big (fait)
- Big : envoi d'un warning à la ferme

10.4 Timestamps via le 'local module'

Le '**local module**' est développé par le MPIK.

Le '**local module**' permet de dater les évènements et d'identifier les destinataires à qui les serveurs devront les envoyer ; ie le '**local module**' identifie chaque évènement via les trois champs suivants :

- event
- brunch
- IP

Les serveurs ZORA et EMILIE doivent communiquer avec le '**local module**'. **Question :** comment communiquent-ils (schéma énumérant les flux) ?
Plugger la carte et demander la doc à Patrick

10.5 Big

Définir une fréquence ainsi que la politique de priorité du thread de supervision.

- toutes les 100 ms
- prioritaire sur les ordres

Définir également un quotient de cette fréquence pour chacune des activité du superviseur :

- rapide : Ré-allumage des pixels via le **'star controller'**
- ? : supervision des basses tensions

10.5.1 Températures des DAQ

Prévu dans le serveur Big :

- un seuil de déclenchement de l'alerte
- un seuil (plus faible) d'annulation de l'alerte

10.5.2 Hautes tensions

La haute tension nominale est de 1000 Volts.
On peut avoir les hautes tension en marche (moins de 400 Volts) et le capot ouvert lors des "changement de priorité".

10.5.3 Capot

- VMC IO à installer.
- Contrôle depuis le serveur de console blackbox ?