EDOC117

- CENBG -

* * *

BIBLIOTHEQUE D'UTILISATION
DE LA
CARTE
HSM 8170
SOUS
VxWorks

Patrice KADIONIK
04/02/91

Ce document est basé sur le manuel d'utilisation de la carte mémoire HSM ("HSM 8170 User's Manual"). Pour plus d'explications, se reporter à ce document de base...

1 Généralités :

Le but de ce papier est de présenter de façon succinte la bibliothèque d'utilisation de la HSM dans l'environnement VxWorks. Il est bon de noter le parallélisme qui existe entre la création d'une bibliothèque sous UNIX et sous VxWorks :

1.1 Bibliothèque sous UNIX :

La création d'une bibliothèque se fait en utilisant *ar*. Dans un fichier source hsm.c, on définit les différentes fonctions composant la bibliothèque.
Puis, l'on fait une compilation de ce fichier sans édition de liens (fichier hsm.o). Ce peut être un fichier de format *a.out* de type sparc (compilation) ou 68020 (cross-compilation en utilisant le GNU-CC ou le cross-compilateur SUN).
On fait appel à l' "archiveur".
L'utilisation ultérieure de la bibliothèque se fera en précisant lors de la (cross-)compilation d'un fichier source utilisant cette bibliothèque, le nom de la bibliothèque et le répertoire où se trouve le fichier archive.
Cela donne :

```
% vi hsm.c
...
% cc -c hsm.c
% ar rcv libhsm.a hsm.o
% mv libhsm.a /home/users/kadionik/lib
% vi file.c utilisation de la bibliothèque
...
% cc -o file file.c -lhsm -L/home/users/kadionik/lib
```

Notons qu'il est impératif que le fichier archive s'appelle lib???.a pour être reconnu par l'éditeur de liens !
La position des différentes options de compilation est très importante.
Remarquons que l'on fait une édition de liens statique pour la compilation du fichier source qui utilise la bibliothèque ; c'est à dire qu'est inclus le code de cette bibliothèque dans le code

exécutable de chaque programme qui l'utilise. On a donc un gaspillage de la mémoire quand 2 processus utilisant cette bibliothèque s'exécutent.
L'édition de liens (sous UNIX) dynamique (bibliothèque partagée) permet de remédier à ce problème...

1.2 Bibliothèque sous VxWorks :

La marche à suivre est plus simple car VxWorks est avant tout basé sur l'édition de liens dynamique (résolution des problèmes de références externes en utilisant la Table des Symboles de VxWorks).
Il suffit donc pour une cible MVME 147 de faire une cross-compilation du fichier source de la bibliothèque sans édition de liens, puis de le charger dans la cible :

```
% pwd
/home/users/kadionik/VxWorks-5.0
% vi hsmlib.c
% cc -sun3 -I/home/users/kadionik/vw-5.0/h -c hsmlib.o hsmlib.c
% rlogin "cible"
-> cd " kadionik/VxWorks-5.0"
-> ld<hsmlib.o
```

La bibliothèque est chargée une seule fois et est accessible à toutes les tâches s'exécutant dans le chassis cible.

2 La bibliothèque :

Elle se compose principalement de 3 fichiers :

- hsmreg.h : fichier de structure de données.
- hsmlib.c : définition de toutes les fonctions.
- hsmlib.h : fichier de type .h .

L'utilisation ultérieure présuppose l'ajout d'une directive du type :

```
# include "hsmlib.h"
```

dans le fichier source des programmes qui s'exécuteront dans le chassis cible.

On distingue principalement des fonctions permettant l'initialisation de la carte HSM :

- hsmVmeMap();
- hsmRegMap();
- hsmReset();
- hsmInit();
- hsmAcqEna();
- hsmIntClr();

On distingue aussi des fonctions permettant des opérations de consultation-modification de registres de la carte :
- hsmRegCpy();
- hsmAcqSts();
- hsmAcqEna();
- hsmIntSrc();
- hsmIntVec();
- hsmIntEna();
- hsmIntLev();
- hsmMemOvf();
- hsmMemPtr();
- hsmMemCnt();

On distingue enfin des fonctions permettant des opérations diverses :
- hsmMemFil();

On se reportera aux notes manuscrites et aux fichiers source pour plus d'informations...

3 Initialisation de la carte HSM :

On distingue d'abord une initialisation "hard" en configurant par des jumpers les adresses VME et VSB... de la carte (Cf le "User's Manual").

On distingue ensuite une initialisation "soft" de la carte en utilisant la bibliothèque. Ceci se fait simplement en utilisant successivement les fonctions suivantes :
- hsmVmeMap();
- hsmRegMap();
- hsmReset();
- hsmInit();
- hsmAcqEna();

4 Prochains développements :

En cours d'intégration dans une chaine d'acquisition qui existe au CENBG, la carte HSM et le logiciel d'acquisition à produire permettront d'avoir des résultats sur les performances de celle-ci. Nous allons aussi intégrer une carte CPU (MVME 135) voire une deuxième carte CPU (MVME 165) dans notre "Event Builder", ce qui nous permettra de tester les communications inter-processeurs et de mettre en place des logiciels de tests...