Manuel d'utilisateur :

1. Manuel d'utilisateur :
    1. I. Organisation des bibliothèques système
    2. II. API du système
    3. III. Description de la plateforme
    4. IV. Compilation d?une application

I. Organisation des bibliothèques système

La figure suivante illustre la modalisation du système :

Les bibliothèques constituant le système sont :

- pthread : contient l'interface système des Threads POSIX.
- libc : contient l?interface des services système tel que malloc, printf, read, memcpy ..etc.
- mwmr : contient l?interface système des FIFO MWMR.
- sys : contient le code système qui ne dépend pas de l'architecture de la plate-forme ou de type des processeurs utilisés.
- cpu : contient le code système en assembleur qui dépend de type des processeurs de la plate-forme.
- arch : contient le code C qui dépend de la plate-forme tel que les configurations système vis-à-vis des composants de la plate-forme, les ISR d?interruption des différant types de cibles ..etc.

En cas de modification au niveau de la configuration matériel, il suffit d?adapter le code système des deux bibliothèques cpu et arch pour pouvoir déployer Mutek*P* sur la nouvelle plate-forme.

II. API du système

Mutek*P* fourni trois bibliothèques pour les Threads de l?application :

- Un sous-ensemble de l'API des threads POSIX :
    - pthread_attr_init: initialiser la structure d?un attribut.
    - pthread_create?: créer un Thread
    - pthread_exit?: mettre fin à une tâche avec une valeur de retour
    - pthread_self?: récupérer l?indentité du Thread appelant.
    - pthread_equal?: tester l?égalité entre deux identificateur.
    - pthread_yield?: céder le processeur pour un autre Thread.
    - pthread_join: attendre la fin d?un Thread.
    - pthread_spin_init: initialiser un verrou à une attente active.
    - pthread_spin_destroy: détruire un verrou.
    - pthread_spin_lock: verrouiller le verrou à une attente active.
    - pthread_spin_trylock: version non bloquante de pthread_spin_lock.
    - pthread_spin_unlock: déverrouiller le verrou à une attente active.

- Les fonctions implémentant le protocole MWMR :
    - mwmr_read?: lecture d?une FIFO MWMR.
    - mwmr_write?: écriture dans une FIFO MWMR.
    - mwmr_init?: création et initialisation d?une FIFO MWMR.

- Quelques fonctions de la bibliothèque libC :
    - printf?: afficher une chaîne de caractères formatée sur le terminal utilisateur (tty1).

- sprintf?: écrire une chaîne de caractères formatée dans un buffer.
- fprintf?: afficher une chaîne de caractères formatée sur un terminal donné.
- puts?: afficher une chaîne de caractères non formatée sur le terminal utilisateur (tty1).
- strlen?: calculer la longueur d'une chaîne de caractères.
- malloc?: allocation de mémoire dynamique.
- read?: pour lire un nombre fixe d'octets à partir du buffer système.
- write?: pour écrire le contenu d'un buffer sur un terminal.
- memset?: remplir une zone mémoire par une valeur donnée.
- memcpy?: copie une zone moire source vers une autre zone mémoire distination.

- Appels propres à l'implémentation Mutek*p* :
  - pthread_attr_setprocid_np : permet d?affecter un numéro de processeur à un attribut.
  - pthread_profiling_np? : permet d'afficher, sur le terminal système (tty0), des statistiques sur le déroulement de l'application et la réactivité du système.

III. Description de la plateforme

La figure suivante illustre la plateform matérielle

Composants de la plate-forme :

- Type MPSoCs 1 seul cluster
  - 4 processeurs de type MIPS R3000 avec cache Instruction et Data (16 lignes / 8 mots).
- Un inter-connect VCI (Virtual Component Interconnect)
- Une RAM multi segments
  - 8 segments
  - taille de 256 octets à 64ko
- Un contrôleur TTY
  - 4 terminaux
- Un contrôleur de timers programmables.
  - 5 timers
- Un ICU : concentrateur d?interruptions
  - relié au processeur 0 pour 1 timers et les TTY
- Un contrôleur de verrous matériels

IV. Compilation d?une application