

# Mälardalen WCET Benchmarks

Tests avec RTEMS sur simulateur de processeur LEON 3

# Utilisation de des outils de mesures de l'API de RTEMS

RTEMS permet de mesurer le temps d'exécution de notre programme d'une manière plus simple qu'avec l'utilisation de timespec. Il faut pour cela utiliser les fonctions `rtems_cpu_usage_report` et `rtems_cpu_usage_reset`.

`rtems_cpu_usage_report` permet d'afficher un rapport sur le temps d'exécution des différentes tâches.

`rtems_cpu_usage_reset` remet à zéro le compteur de la fonction précédente. Il est donc utile de faire appel à cette fonction avant la portion de programme que l'on souhaite mesurer.

Pour tester ces fonctions, on utilisera le programme contenant une fonction récursive du benchmark de malmö (calcul récursif de la suite de fibonacci).

CPU USAGE BY THREAD			
ID	NAME	SECONDS	PERCENT
0x04010001	IDLE	0	0.000
0x08010002	TA1	1203	0.748
0x08010003	TA2	203	0.126
0x08010004	TA3	202	0.126
TICKS SINCE LAST SYSTEM RESET:			1600
TOTAL UNITS:			1608

*Format de sortie de la fonction  
`rtems_cpu_usage_report` décrit dans  
la doc de l'API de RTEMS*

# Test de fonction récursive

Pour tester ces fonctions, on utilisera le programme contenant une fonction récursive du benchmark de malmö (calcul récursif de la suite de fibonacci).

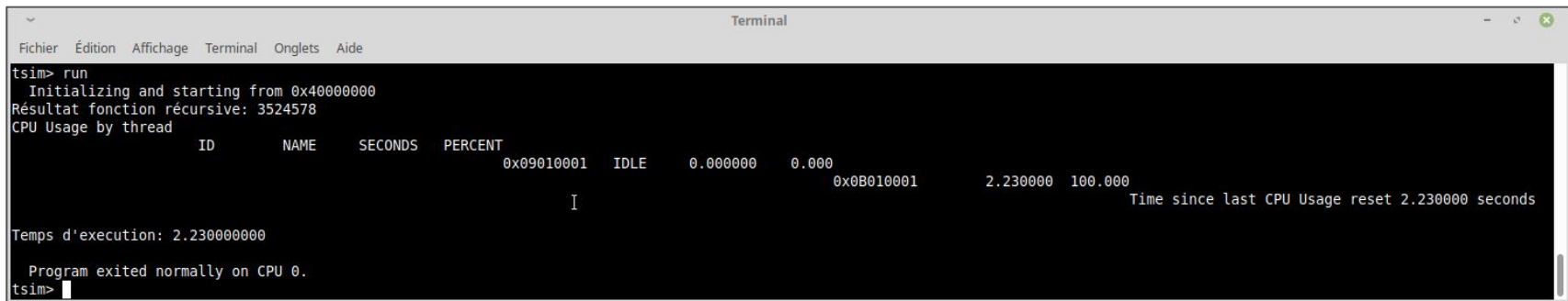
Afin d'avoir un affichage du temps d'exécution, il faut entrer une valeur suffisamment pour que le programme ne s'achève pas trop vite.

```
//----- Fonctions -----  
  
int fib(int i)  
{  
    if(i==0)  
        return 1;  
  
    if(i==1)  
        return 1;  
  
    return fib(i-1) + fib(i-2);  
}  
  
//-----  
  
void* POSIX_Init(void *argument)  
{  
    long In;  
  
    rtems_cpu_usage_reset();  
  
    //----- Espace de test -----  
  
    In = fib(32);  
  
    printf("Résultat fonction récursive: %ld\n", In);  
  
    //-----  
  
    rtems_cpu_usage_report();  
  
    exit(0);  
    return 0;  
}
```

# Test sur fonction récursive

Le résultat de ce test montre bien le temps d'exécution du programme de manière plus simple qu'avec les timespec, cependant deux problèmes sont présents:

- Le temps d'exécution est en secondes et non en tour d'horloge comme indiqué dans la documentation.
- La précision du compteur souffre des mêmes défauts que les timespec: si le programme est trop rapide, le temps d'exécution affiché est 0.



```
tsim> run
  Initializing and starting from 0x40000000
Résultat fonction récursive: 3524578
CPU Usage by thread
  ID          NAME          SECONDS  PERCENT
  0x09010001  IDLE          0.000000  0.000
  0x0B010001          2.230000  100.000
  I
Time since last CPU Usage reset 2.230000 seconds

Temps d'execution: 2.230000000

Program exited normally on CPU 0.
tsim>
```

# Affichage du temps d'exécution

Il semble que l'affichage en secondes et non en tour d'horloge soit présente depuis les versions de RTEMS supérieur à la 4.7 (celle utilisé ici est la 4.8).

La documentation de RTEMS propose un fonction `rtems_timespec_to_ticks` permettant la conversion d'un timespec en nombres de tour d'horloge, cependant cela génère une erreur lors de la compilation.

## 34.4.13. TIMESPEC\_TO\_TICKS - Convert struct timespec Instance to Ticks

### CALLING SEQUENCE:

```
uint32_t rtems_timespec_to_ticks(  
    const struct timespec *time  
);
```

### DIRECTIVE STATUS CODES:

This directive returns the number of ticks computed.

### DESCRIPTION:

This directive converts the `time` timespec to the corresponding number of clock ticks.

### NOTES:

NONE