

ADAPTACIÓN DE LA ARQUITECTURA 'LINUX_LIB' DE MARTE OS A MULTIPROCESADORES

Daniel Medina Ortega

Programa oficial de postgrado en Ciencias, Tecnología y Computación

Master en Computación

FACULTAD DE CIENCIAS - UNIVERSIDAD DE CANTABRIA

Jueves, 29 de Julio de 2010

- 1 Introducción
- 2 Soporte Linux para SMP
- 3 Adaptación de MaRTE OS
- 4 Pruebas y evaluación
- 5 Conclusiones y trabajo futuro

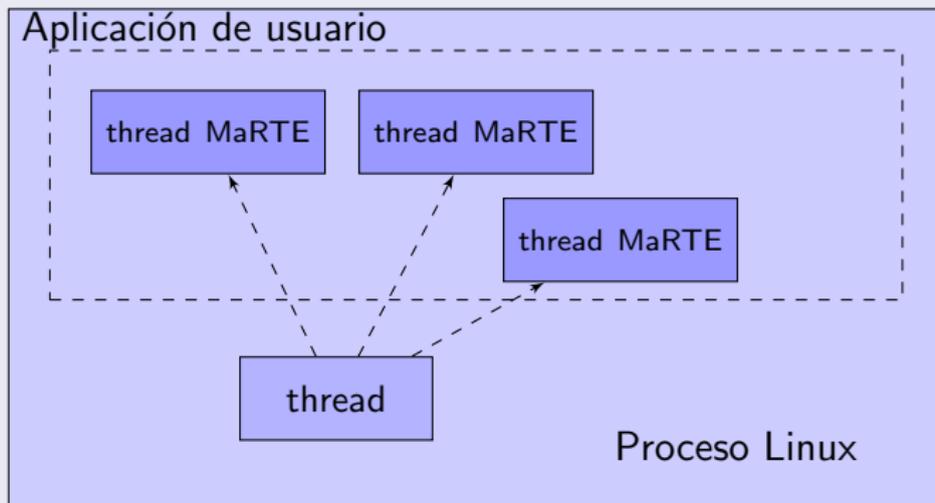
MaRTE OS

Sistema operativo de tiempo real monoprocesador

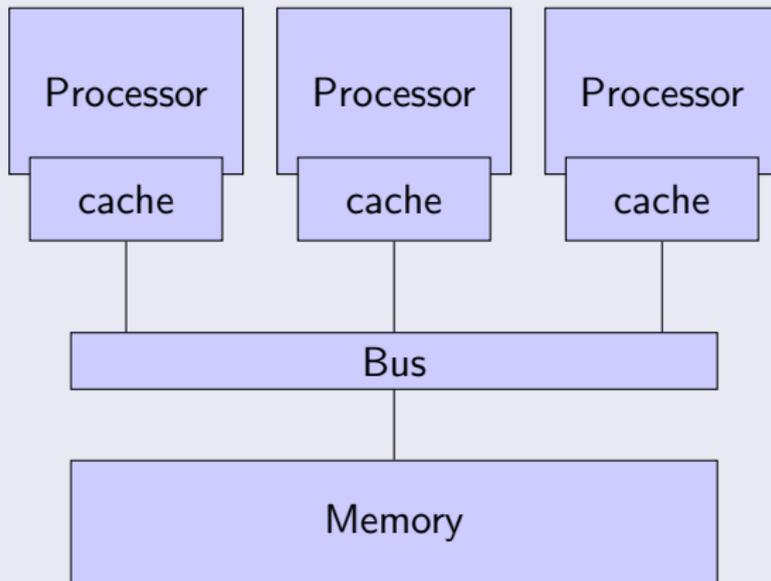
Arquitecturas de MaRTE OS

- x86
 - Ejecutable ELF. *Stand Alone*.
- linux
 - Temporizador y señales de Linux.
- linux_lib
 - Ídem. arquitectura linux
 - Librerías de Linux.

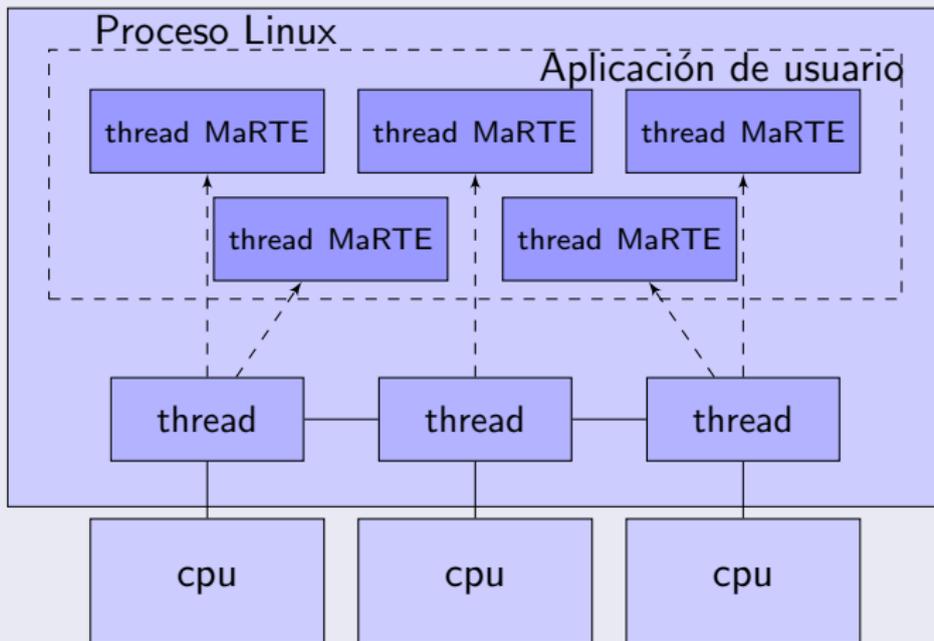
Gestión de los hilos (linux y linux_lib)



Uniform Memory Access



Objetivos

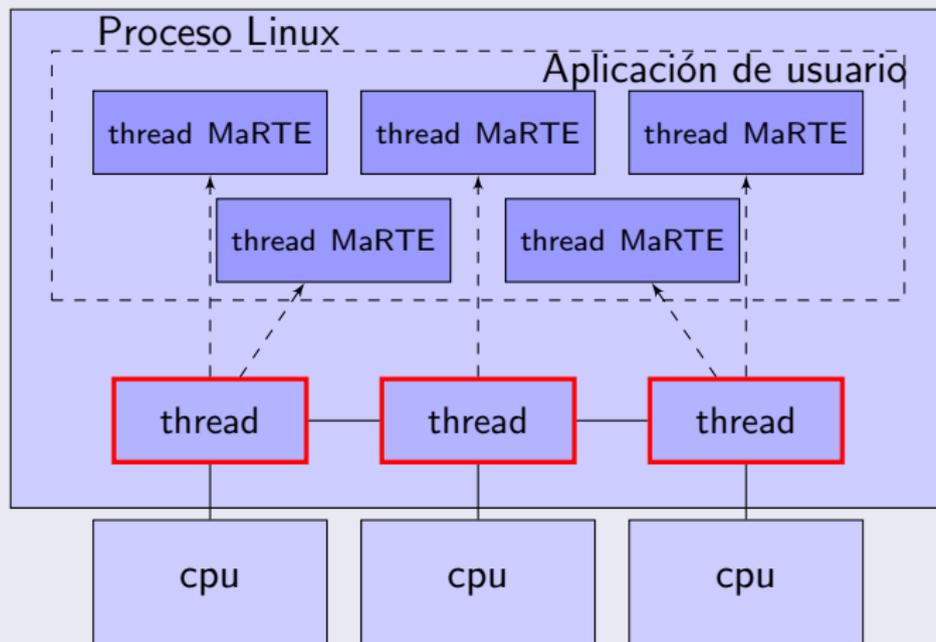


- Soporte de Linux para SMP
- Adaptación de MaRTE OS

Soporte Linux

- Clone: creación de hilos Linux
- Afinidad: fijación a un procesador
- Señales: comunicación entre hilos
- Identificación de la CPU

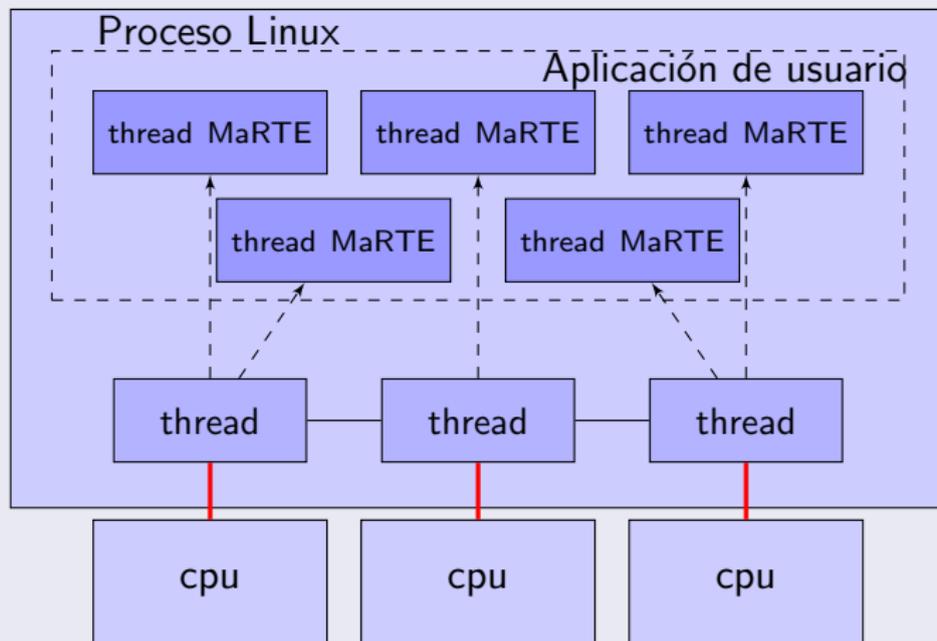
Clone: creación de hilos Linux



```
int clone (int (*fn)(void *), void *chils_stack, int flags, void *arg);
```

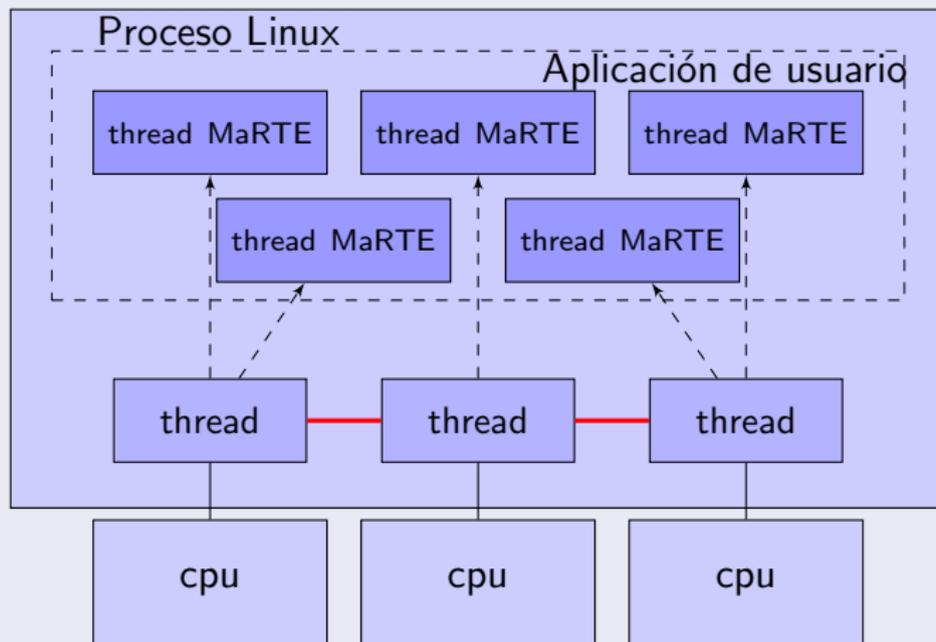
Soporte Linux para SMP

Afinidad: fijación a un procesador



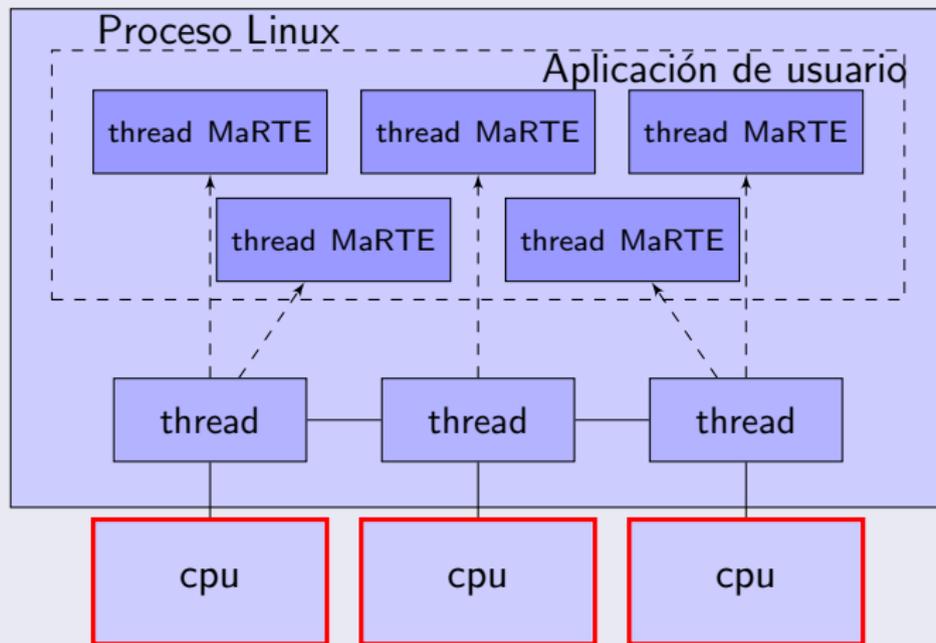
```
int sys_sched_setaffinity(pid_t pid, unsigned int cpusetsize, cpu_set_t *mask);
```

Señales: comunicación entre hilos



```
long sys_tgkill (int tgid, int pid, int sig);
```

Identificación de la CPU



```
asm(“cpuid” : “=b” (id) : “a” (0x01));
```

- Spinlock
- Gestión de las tareas
- Algoritmo de planificación
- Sección crítica del núcleo de MaRTE OS
- Inicialización
- Interfaz abstracta con el hardware

Spinlock

Mecanismo de sincronización mediante espera activa basada en una operación de decremento atómico

En MaRTE OS

- Usuario
- Sección crítica dentro del kernel

Gestión de las tareas

- Array de tareas en ejecución
- Array de tareas idle
- Cola de tareas ejecutables

Array de tareas en ejecución



Array de tareas idle



Cola de tareas ejecutables



Fases

Búsqueda del procesador con la tarea menos prioritaria

- No se requiere cambio de contexto
- Remoto → Envío de interrupción
- Local
 - Búsqueda reducida (excluyendo el procesador local)
 - No se requiere cambio de contexto
 - Remoto → Envío de interrupción
 - Cambio de contexto local

Sección Crítica del núcleo

Todas las operaciones que manipulan las estructuras del núcleo

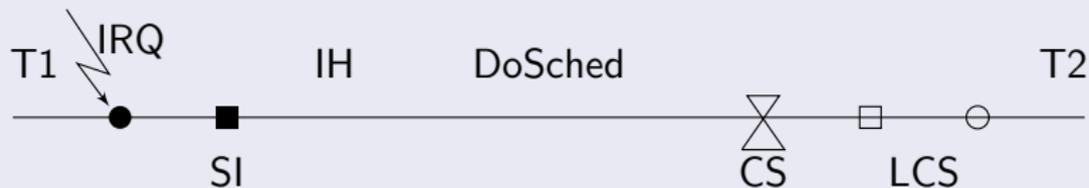
Leyenda

- Enable Interrupts
- Disable Interrupts
- ⌵ Context Switch
- CS
- Lock Kernel Spinlock
- Unlock Kernel Spinlock
- ⚡ Interrupt arrival

Ejemplo 1: Cambio de contexto sencillo.



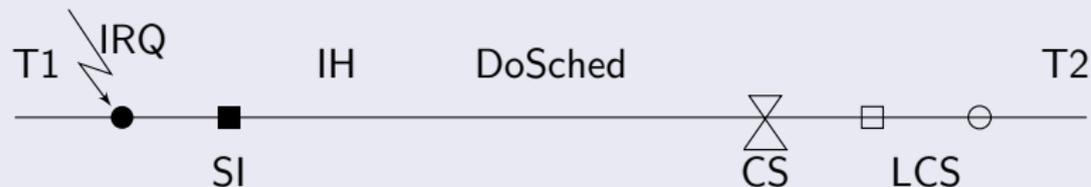
Ejemplo 2: Cambio de contexto por interrupción.



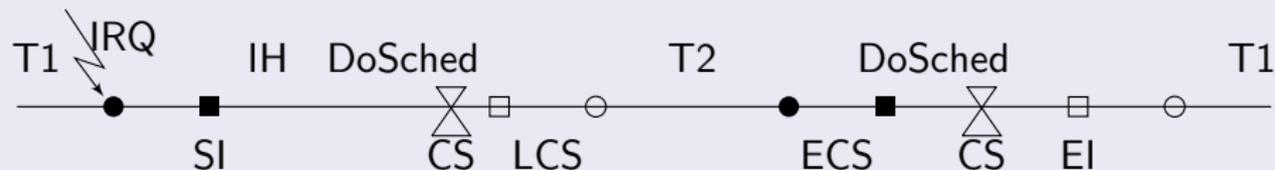
Ejemplo 3: Cambio de contexto completo



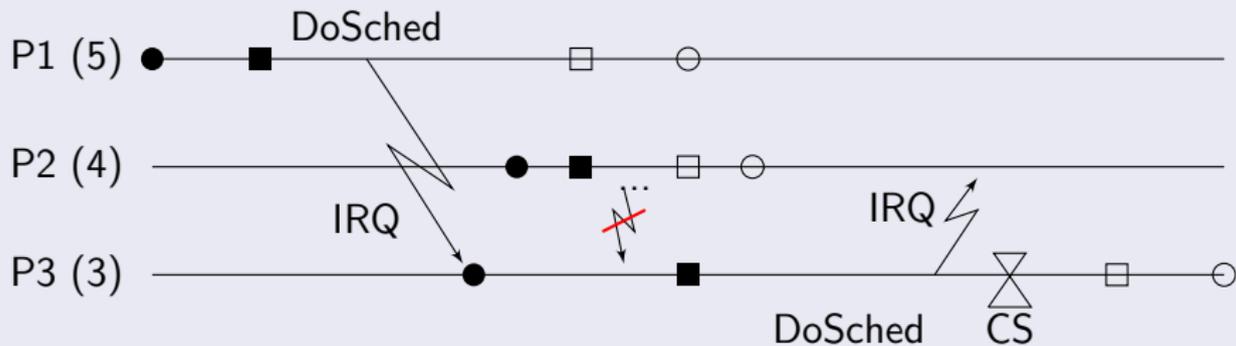
Ejemplo 2: Cambio de contexto por interrupción.



Ejemplo 3: Cambio de contexto completo



Ejemplo 4: Interrupción entre procesadores



Algoritmo de planificación

Búsqueda del procesador con la tarea menos prioritaria

- No se requiere cambio de contexto
- Remoto → Envío de interrupción
- Local
 - Búsqueda reducida (excluyendo el procesador local)
 - No se requiere cambio de contexto
 - Remoto → Envío de interrupción
 - Cambio de contexto local

Inicialización

- 1 Inicialización del spinlock del kernel
- 2 Inicialización normal de MaRTE OS
- 3 Inicialización SMP de MaRTE OS
 - Creación de hilos Linux y asignación al procesador
 - Primer cambio de contexto

Interfaz abstracta con el hardware (HAL)

- HAL.SMP
 - Initialize
 - Cpuid
 - Send_IPI

Pruebas y evaluación

Pruebas de ejecución de 30 tareas (μs)

monoprocesador 603.039 multiprocesador 162.650 (4 proc.)

Cambio de contexto (μs)

	mínimo	medio	máximo
monoprocesador	2	3,039	7
multiprocesador (local)	6	10,650	53
multiprocesador (remoto)	9	26,031	569

Envío de una señal Linux (μs)

Linux 1,682 10,036 504,229

Conclusiones y trabajo futuro

Conclusiones

- Soporte Linux para SMP
- Adaptación de MaRTE OS
- Nos ha permitido enfrentarnos a gran parte de la problemática de los sistemas multiprocesador

Trabajo futuro

- Integración en la distribución oficial de MaRTE OS
- Portar a la arquitectura x86 y/o ARM multiprocesador
- Implementar otros algoritmos de planificación
- Implementar la interfaz multiprocesador propuesta para la futura revisión del estándar Ada

FIN

GRACIAS POR SU ATENCIÓN