(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2013-0063825
(43) 공개일자 2013년06월17일

(51) 국제특허분류(Int. Cl.)
G06F 9/22 (2006.01) *G06F 9/48* (2006.01)
(21) 출원번호 10-2011-0130407
(22) 출원일자 2011년12월07일
심사청구일자 없음

(71) 출원인
삼성전자주식회사
경기도 수원시 영통구 삼성로 129 (매탄동)
(72) 발명자
박찬주
경기도 수원시 영통구 영통동 벽적골9단지아파트 910동 1702호
이성민
경기도 수원시 영통구 영통동 벽적골8단지아파트 844동 1903호
(74) 대리인
특허법인 신지

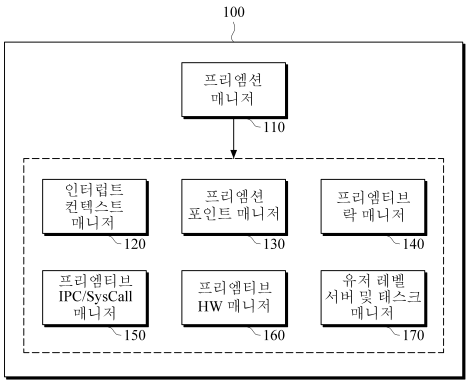
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **운영체제에서 동적으로 선점 구간을 조정하는 장치 및 방법**

(57) 요 약

각종 운영체제에서 다양한 응용 및 서비스에 따라 동적으로 선점 구간을 조정하는 기술에 관한 것으로, 시스템 컨텍스트의 변경 여부를 모니터링하고 현재 시스템 컨텍스트가 변경되면 그 변경된 시스템 컨텍스트에 따라 현재 선점 모드를 동적으로 변경함으로써 커널의 선점 구간을 동적으로 조정하는 프리엠션 매니저를 포함할 수 있다. 이를 통해 실시간성이 요구되는 의료 애플리케이션이나 일반 애플리케이션이 혼재하는 경우에도 최적의 성능을 보장할 수 있다.

대 표 도 - 도1



특허청구의 범위

청구항 1

시스템 컨텍스트(System Context)의 변경 여부를 모니터링하고 현재의 시스템 컨텍스트가 변경되면 그 변경된 시스템 컨텍스트에 따라 현재 선점 모드를 설정하여 커널의 선점 구간(Preemption Section)을 동적으로 조정하는 프리엠션 매니저(Preemption Manager);를 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 2

제1항에 있어서, 상기 현재 선점 모드는,

시스템 컨텍스트의 실시간 처리 요구 정도에 따라 세분화된 비선점 모드(Non Preemption Mode), 일반 선점 모드(General Preemption Mode) 및 완전 선점 모드(Fully Preemption Mode) 중 어느 하나인 것을 특징으로 하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 3

제2항에 있어서, 상기 각 선점 모드는,

시스템 컨텍스트의 실시간 처리 요구 정도에 따라 다시 세부적인 선점 수준으로 구분되는 것을 특징으로 하고,

상기 프리엠션 매니저는,

상기 설정된 현재 선점 모드의 세부적인 선점 수준을 더 설정하여 커널의 선점 구간을 동적으로 조정하는 것을 특징으로 하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 4

제1항에 있어서,

상기 설정된 현재 선점 모드에 따라 ISR(Interrupt Service Routine)의 처리 방법을 동적으로 변경하는 인터럽트 컨텍스트 매니저(Interrupt Context Manager);를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 5

제4항에 있어서, 상기 인터럽트 컨텍스트 매니저는,

상기 설정된 현재 선점 모드가 비선점 모드이면 인터럽트 컨텍스트 모드(Interrup Context Mode)에서 ISR이 계속 수행되도록 하고, 비선점 모드가 아니면 인터럽트 스레드 모드(Interrupt Thread Mode)에서 ISR이 수행되도록 하는 것을 특징으로 하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 6

제1항에 있어서,

상기 설정된 현재 선점 모드에 따라 커널 서비스 루틴(Kernel Service Routine)에 현재 스레드의 재스케쥴 체크 여부를 동적으로 변경하는 프리엠션 포인트(Preemption Point)를 삽입하는 프리엠션 포인트 매니저(Preemption Point Manager);를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 7

제6항에 있어서, 상기 프리엠션 포인트는,

상기 설정된 현재 선점 모드를 확인하고, 현재 선점 모드가 비선점 모드가 아닌 경우 현재 스레드의 재스케쥴 여부를 체크하는 코드를 포함하는 것을 특징으로 하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 8

제1항에 있어서,

커널이 스핀락(Spin Lock)을 가진 태스크를 수행할 때 상기 설정된 현재 선점 모드에 따라 동적으로 스핀락 또는 뮤텍스(mutex) 중 어느 하나를 사용하여 그 태스크를 수행하도록 하는 프리엠티브 락 매니저(Preemptive Lock Manager);를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 9

제1항에 있어서,

IPC(Inter-Process Communication) 처리시 상기 설정된 현재 선점 모드에 따라 IPC의 타임 아웃을 동적으로 조정하고, 시스템 콜이 수행되면 현재 선점 모드에 따라 동기식 또는 비동기식으로 상기 시스템 콜이 수행되도록 하는 프리엠티브 IPC/SysCall 매니저;를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 10

제1항에 있어서,

상기 설정된 현재 선점 모드에 따라 실시간 처리가 요구되는 태스크가 TLB(Translation Lookaside Buffer)에서 플러쉬(flush)되지 않도록 동적으로 TLB 락다운(Lockdown)을 설정하거나 해제하는 프리엠티브 HW 매니저;를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 장치.

청구항 11

시스템 컨텍스트(System Context)의 변경 여부를 모니터링하는 단계; 및

상기 모니터링 결과 현재 시스템 컨텍스트가 변경되면 그 변경된 시스템 컨텍스트에 따라 현재 선점 모드를 설정하여 커널의 선점 구간(Preemption Section)을 동적으로 조정하는 단계;를 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

청구항 12

제11항에 있어서, 상기 현재 선점 모드는,

시스템 컨텍스트의 실시간 처리 요구 정도에 따라 세분화된 비선점 모드(Non Preemption Mode), 일반 선점 모드(General Preemption Mode) 및 완전 선점 모드(Fully Preemption Mode) 중 어느 하나인 것을 특징으로 하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

청구항 13

제12항에 있어서, 상기 각 선점 모드는,

시스템 컨텍스트의 실시간 처리 요구 정도에 따라 다시 세부적인 선점 수준으로 구분되는 것을 특징으로 하고,

상기 커널의 선점 구간을 동적으로 조정하는 단계는,

상기 설정된 현재 선점 모드의 세부적인 선점 수준을 더 설정하여 커널의 선점 구간을 동적으로 조정하는 것을 특징으로 하는 운영체제에서 동적으로 선점 구간을 조정하는 방법

청구항 14

제11항에 있어서,

상기 설정된 현재 선점 모드에 따라 ISR(Interrupt Service Routine)의 처리 방법을 동적으로 변경하는 단계;를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

청구항 15

제14항에 있어서, 상기 ISR의 처리 방법을 변경하는 단계는,

하드웨어 인터럽트가 발생되면 상기 설정된 현재 선점 모드를 확인하는 단계; 및

상기 확인 결과 현재 선점 모드가 비선점 모드이면 인터럽트 컨텍스트 모드(Interrupt Context Mode)에서 ISR이

계속 수행되도록 하고, 비선점 모드가 아니면 인터럽트 스레드 모드(Interrupt Thread Mode)에서 ISR이 수행되도록 하는 단계;를 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

청구항 16

제11항에 있어서,

상기 설정된 현재 선점 모드에 따라 커널 서비스 루틴(Kernel Service Routine)에 현재 스레드의 재스케쥴 체크 여부를 동적으로 변경하는 프리엠션 포인트(Preemption Point)를 삽입하는 단계;를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

청구항 17

제16항에 있어서, 상기 프리엠션 포인트는,

상기 설정된 현재 선점 모드를 확인하고, 현재 선점 모드가 비선점 모드가 아닌 경우 현재 스레드의 재스케쥴 여부를 체크하는 코드를 포함하는 것을 특징으로 하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

청구항 18

제11항에 있어서,

커널이 스핀락(Spin Lock)을 가진 태스크를 수행할 때 상기 설정된 현재 선점 모드에 따라 동적으로 스핀락 또는 뮤텍스(mutex) 중 어느 하나를 사용하여 상기 태스크를 수행하도록 하는 단계;를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

청구항 19

제11항에 있어서,

시스템 콜(System Call) 수행시 상기 설정된 현재 선점 모드에 따라 상기 시스템 콜이 동기식 또는 비동기식으로 수행되도록 하는 단계;를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

청구항 20

제11항에 있어서,

상기 설정된 현재 선점 모드에 따라 실시간 처리가 요구되는 태스크가 TLB(Translation Lookaside Buffer)에서 플러쉬(flush) 되지 않도록 동적으로 TLB 락다운(Lockdown)을 설정하는 단계;를 더 포함하는 운영체제에서 동적으로 선점 구간을 조정하는 방법.

명 세 서

기 술 분 야

[0001] 각종 운영체제에서 다양한 응용 및 서비스에 따라 동적으로 선점 구간을 조정하는 기술에 관한 것이다.

배 경 기 술

[0002] 최근 컴퓨터 기술의 발달과 더불어 운영체제 또한 발전을 거듭하고 있다. 현재의 운영체제는 특정한 애플리케이션의 요구사항에 최적화되어 디자인되어 있으며, 크게 성능을 극대화하기 위한 운영체제와 실시간성을 극대화하기 위한 운영체제로 분류될 수 있다. 전자로는 서버나 데스크탑 또는 모바일용 운영체제로서 윈도우즈, 리눅스, 안드로이드, MacOS 등이 있으며, 후자로는 Nucleus, RTLinux, VxWorks 등이 있다. 앞으로의 컴퓨팅 환경은 최근 증가하고 있는 IT 융 복합화 경향에 따라 다양한 애플리케이션이 혼재하는 추세이다. 그러나, 현재의 일반적인 운영체제들은 특정한 목적에 최적화되도록 설계되어 있어 스마트카 또는 퍼스널 헬스기기와 같이 실시간성을 요구하는 애플리케이션과 처리 효율성을 우선하는 일반 애플리케이션 등 다양한 애플리케이션들의 요구 사항을 만족시키는데 한계가 있다.

발명의 내용

해결하려는 과제

[0003] 각종 운영체제에서 다양한 애플리케이션 및 서비스에 따라 최적의 성능이 보장되도록 커널의 선점 구간(Preemption Section)을 동적으로 조정할 수 있는 장치 및 방법을 제공하기 위함이다.

[0004] 또한, 주변의 컨텍스트를 인지하고 컴퓨팅 가능한 리소스를 활용하는 다양한 애플리케이션을 동작해야 하는 미래 상황 인식 컴퓨팅 기기(context-aware computing device)들의 운영체제에 적용하기 위한 장치 및 방법을 제공하기 위함이다.

과제의 해결 수단

[0005] 일 양상에 따르면, 운영체제에서 동적으로 선점 구간을 조정하는 장치는 시스템 컨텍스트(System Context)의 변경 여부를 모니터링하고 현재의 시스템 컨텍스트가 변경되면 그 변경된 시스템 컨텍스트에 따라 현재 선점 모드를 설정하여 커널의 선점 구간(Preemption Section)을 동적으로 조정하는 프리엠션 매니저(Preemption Manager)를 포함한다.

[0006] 이때, 현재 선점 모드는 시스템 컨텍스트의 실시간 처리 요구 정도에 따라 세분화된 비선점 모드(Non Preemption Mode), 일반 선점 모드(General Preemption Mode) 및 완전 선점 모드(Fully Preemption Mode) 중 어느 하나일 수 있다.

[0007] 추가적인 양상에 따르면, 각 선점 모드는 시스템 컨텍스트의 실시간 처리 요구 정도에 따라 다시 세부적인 선점 수준으로 구분될 수 있으며, 프리엠션 매니저는 설정된 현재 선점 모드의 세부적인 선점 수준을 더 설정하여 커널의 선점 구간을 동적으로 조정할 수 있다.

[0008] 추가적인 양상에 따르면, 설정된 현재 선점 모드에 따라 ISR(Interrupt Service Routine)의 처리 방법을 동적으로 변경하는 인터럽트 컨텍스트 매니저(Interrupt Context Manager)를 더 포함할 수 있다.

[0009] 인터럽트 컨텍스트 매니저는 설정된 현재 선점 모드가 비선점 모드이면 인터럽트 컨텍스트 모드(Interrup Context Mode)에서 ISR이 계속 수행되도록 하고, 비선점 모드가 아니면 인터럽트 스레드 모드(Interrupt Thread Mode)에서 ISR이 수행되도록 할 수 있다.

[0010] 추가적인 양상에 따르면, 설정된 현재 선점 모드에 따라 커널 서비스 루틴(Kernel Service Routine)에 현재 스레드의 재스케쥴 체크 여부를 동적으로 변경하는 프리엠션 포인트(Preemption Point)를 삽입하는 프리엠션 포인트 매니저(Preemption Point Manager)를 더 포함할 수 있다.

[0011] 프리엠션 포인트는 설정된 현재 선점 모드를 확인하고, 현재 선점 모드가 비선점 모드가 아닌 경우 현재 스레드의 재스케쥴 여부를 체크하는 코드를 포함할 수 있다.

[0012] 추가적인 양상에 따르면, 커널이 스핀락(Spin Lock)을 가진 태스크를 수행할 때 설정된 현재 선점 모드에 따라 동적으로 스핀락 또는 뮤텍스(mutex) 중 어느 하나를 사용하여 그 태스크를 수행하도록 하는 프리엠티브 락 매니저(Preemptive Lock Manager)를 더 포함할 수 있다.

[0013] 추가적인 양상에 따르면, IPC(Inter-Process Communication) 처리시 설정된 현재 선점 모드에 따라 IPC의 타임아웃을 동적으로 조정하고, 시스템 콜이 수행되면 현재 선점 모드에 따라 동기식 또는 비동기식으로 시스템 콜이 수행되도록 하는 프리엠티브 IPC/SysCall 매니저를 더 포함할 수 있다.

[0014] 추가적인 양상에 따르면, 설정된 현재 선점 모드에 따라 실시간 처리가 요구되는 태스크가 TLB(Translation Lookaside Buffer)에서 플러쉬(flush)되지 않도록 동적으로 TLB 락다운(Lockdown)을 설정하거나 해제하는 프리엠티브 HW 매니저를 더 포함할 수 있다.

[0015] 일 양상에 따르면, 운영체제에서 동적으로 선점 구간을 조정하는 방법은 시스템 컨텍스트(System Context)의 변경 여부를 모니터링하는 단계 및 모니터링 결과 현재 시스템 컨텍스트가 변경되면 그 변경된 시스템 컨텍스트에 따라 현재 선점 모드를 설정하여 커널의 선점 구간(Preemption Section)을 동적으로 조정하는 단계를 포함한다.

[0016] 이때, 현재 선점 모드는 시스템 컨텍스트의 실시간 처리 요구 정도에 따라 세분화된 비선점 모드(Non Preemption Mode), 일반 선점 모드(General Preemption Mode) 및 완전 선점 모드(Fully Preemption Mode) 중 어느 하나일 수 있다.

[0017] 추가적인 양상에 따르면, 각 선점 모드는 시스템 컨텍스트의 실시간 처리 요구 정도에 따라 다시 세부적인 선점 수준으로 구분될 수 있으며, 커널의 선점 구간을 동적으로 조정하는 단계는 설정된 현재 선점 모드의 세부적인 선점 수준을 더 설정하여 커널의 선점 구간을 동적으로 조정할 수 있다.

[0018] 추가적인 양상에 따르면, 설정된 현재 선점 모드에 따라 ISR(Interrupt Service Routine)의 처리 방법을 동적으로 변경하는 단계를 더 포함할 수 있다.

[0019] ISR의 처리 방법을 변경하는 단계는 하드웨어 인터럽트가 발생되면 설정된 현재 선점 모드를 확인하는 단계 및 확인 결과 현재 선점 모드가 비선점 모드이면 인터럽트 컨텍스트 모드(Interrupt Context Mode)에서 ISR이 계속 수행되도록 하고, 비선점 모드가 아니면 인터럽트 스레드 모드(Interrupt Thread Mode)에서 ISR이 수행되도록 하는 단계를 포함할 수 있다.

[0020] 추가적인 양상에 따르면, 설정된 현재 선점 모드에 따라 커널 서비스 루틴(Kernel Service Routine)에 현재 스레드의 재스케쥴 체크 여부를 동적으로 변경하는 프리엠션 포인트(Preemption Point)를 삽입하는 단계를 더 포함할 수 있다.

[0021] 프리엠션 포인트는 설정된 현재 선점 모드를 확인하고, 현재 선점 모드가 비선점 모드가 아닌 경우 현재 스레드의 재스케쥴 여부를 체크하는 코드를 포함할 수 있다.

[0022] 추가적인 양상에 따르면, 커널이 스핀락(Spin Lock)을 가진 태스크를 수행할 때 설정된 현재 선점 모드에 따라 동적으로 스핀락 또는 뮤텍스(mutex) 중 어느 하나를 사용하여 태스크를 수행하도록 하는 단계를 더 포함할 수 있다.

[0023] 추가적인 양상에 따르면, 시스템 콜(System Call) 수행시 설정된 현재 선점 모드에 따라 시스템 콜이 동기식 또는 비동기식으로 수행되도록 하는 단계를 더 포함할 수 있다.

[0024] 추가적인 양상에 따르면, 설정된 현재 선점 모드에 따라 실시간 처리가 요구되는 태스크가 TLB(Translation Lookaside Buffer)에서 플러쉬(flush) 되지 않도록 동적으로 TLB 락다운(Lockdown)을 설정하는 단계를 더 포함할 수 있다.

발명의 효과

[0025] 각종 운영체제에서 다양한 애플리케이션 및 서비스에 따라 커널의 선점 구간(Preemption Section)을 동적으로 조정함으로써 스마트 카 또는 퍼스널 헬스기기와 같이 실시간성이 요구되는 애플리케이션이나 일반 애플리케이션들이 혼재하는 경우에도 최적의 성능을 보장할 수 있다.

[0026] 또한, 주변의 컨텍스트를 인지하고 컴퓨팅 가능한 리소스를 활용하는 다양한 애플리케이션을 동작해야 하는 미래 상황 인식 컴퓨팅 기기를 위한 운영체제에 적용할 수 있다.

도면의 간단한 설명

[0027] 도 1은 일 실시예에 따라 운영체제에서 동적으로 선점 구간을 조정하는 장치의 블럭도이다.

도 2는 일 실시예에 따라 운영체제에서 동적으로 선점 구간을 조정하는 방법의 예시도이다.

도 3은 일 실시예에 따라 ISR 수행 방법을 동적으로 조정하는 방법을 나타낸 것이다.

도 4는 일 실시예에 따라 프리엠션 포인트 수행 절차를 동적으로 조정하는 방법을 나타낸 것이다.

도 5는 일 실시예에 따라 시스템의 락 방법이 동적으로 조정되는 예시도이다.

도 6은 일 실시예에 따라 시스템 콜 수행 방법을 동적으로 조정하는 예시도이다.

도 7은 일 실시예에 따라 TLB 락다운을 동적으로 설정하는 방법의 예시도이다.

발명을 실시하기 위한 구체적인 내용

[0028] 기타 실시예들의 구체적인 사항들은 상세한 설명 및 도면들에 포함되어 있다. 본 발명의 이점 및 특징, 그리고 그것들을 달성하는 방법은 첨부되는 도면과 함께 상세하게 후술되어 있는 실시예들을 참조하면 명확해질 것이다. 그러나 본 발명은 이하에서 개시되는 실시예들에 한정되는 것이 아니라 서로 다른 다양한 형태로 구현될 수 있으며, 단지 본 실시예들은 본 발명의 개시가 완전하도록 하고, 본 발명이 속하는 기술분야에서 통상의

지식을 가진 자에게 발명의 범주를 완전하게 알려주기 위해 제공되는 것이며, 본 발명은 청구항의 범주에 의해 정의될 뿐이다. 명세서 전체에 걸쳐 동일 참조 부호는 동일 구성 요소를 지칭한다.

[0029] 이하, 실시예들에 의해 운영체제에서 동적으로 선점 구간을 조정하는 장치 및 방법을 설명하기 위하여 도면들을 참고하여 상세히 설명하도록 한다.

[0030] 도 1은 일 실시예에 따른 운영체제에서 동적으로 선점 구간을 조정하는 장치의 블럭도이다. 본 실시예에 따른 동적으로 선점 구간을 조정하는 장치(100)는 하나의 소프트웨어일 수 있으며 그 장치(100)의 각 구성들은 소프트웨어의 각 모듈일 수 있다.

[0031] 도 1에 도시된 바와 같이 동적으로 선점 구간을 조정하는 장치(100)는 프리엠션 매니저(Preemption Manager)(110)를 포함한다. 프리엠션 매니저(110)는 시스템 컨텍스트(System Context)의 변경 여부를 모니터링하고 현재 시스템 컨텍스트가 변경되면 그 변경된 시스템 컨텍스트에 따라 현재 선점 모드(Current Preemption Mode)를 설정하여 커널(Kernel)의 선점 구간(Preemption Section)을 동적으로 조정할 수 있다. 이때, 시스템 컨텍스트는 사용자 선호도, 디바이스 컨텍스트(예: 디바이스와 센서의 상태), 서비스 컨텍스트(예: 현재의 QoS 상태) 및 주변 환경(예: 실시간성 태스크, 시스템 오버헤드, 위치 등)을 포함할 수 있다. 즉, 프리엠션 매니저(110)는 시스템 컨텍스트에 따라 커널이 어떠한 선점 모드로 동작할지 설정하는 것으로 커널의 아키텍처에 따라 유저 또는 커널 레이어에서 동작 가능하도록 구현될 수 있다.

[0032] 프리엠션 매니저(110)는 실시간성 태스크가 등록되거나 디바이스 컨텍스트 의 변경 등과 같이 시스템 컨텍스트가 변경되면 이를 모니터링하여 변경된 시스템 컨텍스트에 적합한 선점 모드를 현재의 선점 모드로 설정한다. 이때, 선점 모드는 시스템 컨텍스트의 실시간 처리가 요구되는 정도에 따라 비선점 모드(Non Preemption Mode), 일반 선점 모드(General Preemption Mode) 및 완전 선점모드(Fully Preemption Mode)의 3단계로 크게 세분화될 수 있다. 다만, 이는 하나의 예시에 불과하며 선점 모드의 세분화 정도는 시스템 컨텍스트 즉, 시스템이 처리하는 각 애플리케이션이나 각종 태스크, 인터럽트 등의 실시간 처리 요구 정도에 따라 사용자 또는 개발자 등이 다양한 단계로 미리 조절할 수 있다.

[0033] 비선점 모드는 임베디드 서버(Embedded server)나 데이터 스트리밍(Data Streaming)과 같이 실시간 처리보다 높은 처리 효율성이 요구되는 애플리케이션이 실행되는 환경에서 설정될 수 있다. 일반 선점 모드는 멀티미디어, 게임 등 일반 애플리케이션과 보통의 실시간 처리를 요구하는 애플리케이션이 혼재되어 실행되는 환경에서 설정될 수 있다. 완전 선점 모드는 헬스케이 모니터링이나 운전자 감시, 카 크래시 알람(Car-crash alarm) 등 처리 효율보다 실시간 처리가 매우 중요한 애플리케이션이 실행되는 환경에서 설정될 수 있다. 한편, 운영체제에서 사용할 수 있는 선점 구간을 전체 100%로 가정하면 완전 선점 모드는 100%, 일반 선점 모드는 80%, 비선점 모드는 20%의 선점 구간을 사용할 수 있도록 설정될 수 있다. 이와 같이, 시스템 컨텍스트에 따라 위의 선점 구간을 다양하게 세분화하고 동적으로 설정함으로써 다양한 운영체제에서 다양한 애플리케이션 환경의 요구사항을 최적으로 보장할 수 있다.

[0034] 추가적인 양상에 따르면, 각 선점 모드는 다시 세부적인 선점 수준으로 구분될 수 있다. 즉, 100%의 선점 구간을 사용할 수 있는 완전 선점 모드로 설정되는 애플리케이션의 경우에도 실시간 처리 요구 정도는 상이할 것이므로, 그 애플리케이션의 실시간 처리 요구 정도에 따라 선점 구간의 80%에서 100% 사이를 좀 더 세부적인 선점 수준(예: 1%, 5% 또는 10% 단위 등)으로 세분화할 수 있다. 프리엠션 매니저(110)는 현재 선점 모드를 설정하고 그 현재 선점 모드의 세부적인 선점 수준을 더 설정하여 커널의 선점 구간을 동적으로 조정할 수 있다.

[0035] 추가적인 양상에 따르면, 동적으로 선점 구간을 조정하는 장치(100)는 인터럽트 컨텍스트 매니저(Interrupt Context Manager)(120)를 더 포함할 수 있다. 인터럽트 컨텍스트 매니저(120)는 프리엠션 매니저(110)에 의해 설정된 현재 선점 모드에 따라 ISR(Interrupt Service Routine)의 처리 방법을 동적으로 변경할 수 있다. 좀 더 구체적으로, 인터럽트 컨텍스트 매니저(120)는 하드웨어 인터럽트가 발생되면 현재 선점 모드를 확인하여 현재 선점 모드가 비선점 모드이면 인터럽트 컨텍스트 모드(Interrupt Context Mode)에서 ISR이 계속 수행되도록 하고, 비선점 모드가 아니면 인터럽트 스레드 모드에서 ISR이 수행되도록 변경할 수 있다.

[0036] 특정 인터럽트가 시스템에 인가되면 하드웨어에 의해 정해진 인터럽트 벡터(interrupt vector)로 분기되고 인터럽트 전처리기는 인터럽트 컨텍스트 모드에서 현재 선점 모드를 확인하는 작업을 수행한다. 여기서, 인터럽트 벡터란 인터럽트가 발생했을 때 그 인터럽트를 처리할 수 있는 서비스 루틴들의 주소를 가지고 있는 공간을 말한다. 현재 선점 모드가 비선점 모드인 경우에는 실시간성이 중요시되지 않는 환경이므로 인터럽트 컨텍스트 모

드에서 ISR을 처리하고, 일반 선점 모드나 완전 선점 모드인 경우에는 인터럽트 스레드 모드에서 ISR이 처리되도록 인터럽트 스레드를 스케쥴링하고 ISR을 처리한다. 인터럽트 스레드를 이용하면 인터럽트 자체가 스레드로 처리되기 때문에 선점이 언제든지 가능하고 우선순위에 따라 스케줄링을 변경할 수도 있기 때문에 처리 효율에 대한 오버헤드는 다소 존재하지만 실시간성이 향상될 수 있다.

[0037] 또한, 동적으로 선점 구간을 조정하는 장치(100)는 프리엠션 포인트 매니저(Preemption Point Manager)(130)를 더 포함할 수 있다. 프리엠션 포인트 매니저(130)는 프리엠션 매니저(110)에 의해 설정된 현재 선점 모드에 따라 커널 서비스 루틴(Kernel Service Routine)에 현재 스레드의 재스케쥴 체크 여부를 동적으로 변경하는 프리엠션 포인트를 삽입할 수 있다. 프리엠션 포인트는 실행 시간이 오래 소요되는 커널 서비스 루틴의 초반에 삽입될 수 있으며 설정된 현재 선점 모드에 따라 적절한 동작을 수행하기 위한 코드를 포함할 수 있다. 즉, 프리엠션 포인트는 현재 선점 모드를 확인하고, 현재 선점 모드가 비선점 모드이면 현재 스레드를 재스케쥴하지 않고 그대로 리턴하며, 비선점 모드가 아니면 현재 스레드의 재스케쥴 여부를 체크하는 코드를 포함할 수 있다. 따라서, 비선점 모드일 경우 매번 재스케쥴 여부를 확인하지 않으므로 실시간성은 다소 저하되지만 처리 효율성은 향상시킬 수 있다.

[0038] 추가적인 양상에 따르면, 동적으로 선점 구간을 조정하는 장치(100)는 프리엠티브 락 매니저(Preemptive Lock Manager)(140)를 더 포함할 수 있다. 프리엠티브 락 매니저(140)는 커널이 스핀락(Spin Lock)을 가진 태스크를 수행할 때 프리엠션 매니저(110)에 의해 설정된 현재 선점 모드에 따라 동적으로 스핀락 또는 뮤텍스(mutex) 중 어느 하나를 사용하여 그 태스크를 처리하도록 변경할 수 있다. 스핀락이란 다른 스레드가 락을 반환하여 락이 가능해질 때까지 반복적으로 체크하며 대기하는 것을 말한다. 뮤텍스란 상호 배제(MUTual EXclusion)라고도 하는데, 크리티컬 섹션(Critical Section)을 가진 스레드들의 실행 시간이 서로 겹치지 않도록 각각 단독으로 실행되게 하는 기술이다. 다중 프로세스들의 공유 리소스에 대한 접근을 조율하기 위해 락(lock)과 언락(Unlock)을 사용한다.

[0039] 프리엠티브 락 매니저(140)는 설정된 현재 선점 모드가 완전 선점 모드이면 스핀락 대신에 뮤텍스를 사용하도록 동적으로 조정할 수 있다. 이는 스핀락을 가지고 있는 낮은 우선 순위의 태스크가 수행중일 때 동일한 코드를 수행하는 실시간성 태스크가 동일한 스핀락을 필요로 하지 않을 경우 이미 락을 가진 태스크를 선점하여 수행 가능하도록 하기 위함이다. 이때, 동일한 락을 실시간성 태스크가 필요로 하는 경우에는 기다려야 한다. 뮤텍스는 처리 효율면에서는 오버헤드가 존재하므로 프리엠션 매니저(110)에 의해 현재 선점 모드가 비선점 모드로 설정되면 프리엠션 락 매니저(140)는 뮤텍스 코드를 스핀락 코드로 포인팅 되도록 조정하여 바쁜 대기(busy waiting)를 하지만 처리 효율면에서는 성능이 상대적으로 우수한 스핀락 코드를 수행하도록 할 수 있다.

[0040] 추가적인 양상에 따르면, 동적으로 선점 구간을 조정하는 장치(100)는 프리엠티브 IPC/SysCall 매니저(150)를 더 포함할 수 있다. 프리엠티브 IPC/SysCall 매니저(150)는 IPC(Inter-Precess Communication) 처리시 프리엠션 매니저(110)에 의해 설정된 현재 선점 모드에 따라 IPC의 타임 아웃을 동적으로 조정하거나 시스템 콜(System Call)이 수행되면 현재 선점 모드에 따라 동기식(Synchronous) 또는 비동기식(asynchronous)으로 수행되도록 동적으로 조정할 수 있다. 예를 들어, 프리엠티브 IPC/SysCall 매니저(150)는 시스템 콜이 수행되면 현재 수행된 태스크가 실시간성이 요구되는 태스크인지를 파악한다. 즉, 프리엠션 매니저(110)에 의해 설정된 현재 선점 모드가 완전 선점 모드로서 실시간 처리가 요구되는 태스크인 경우 비동기식으로 시스템 콜을 수행하도록 하고, 설정된 현재 선점 모드가 비선점 모드와 같이 실시간 처리가 요구되지 않는 태스크이며 동기식으로 시스템 콜이 수행되도록 조정할 수 있다.

[0041] 추가적인 양상에 따른 동적으로 선점 구간을 조정하는 장치(100)는 프리엠티브 HW 매니저(160)를 더 포함할 수 있다. 프리엠티브 HW 매니저(160)는 프리엠션 매니저(110)에 의해 설정된 현재 선점 모드에 따라 실시간 처리가 요구되는 실시간성 태스크가 TLB(Translation Lookaside Buffer)에서 플러쉬(flush)되지 않도록 동적으로 TLB 락다운(Lockdown)을 설정하거나 해제할 수 있다. 또한, 그 실시간성 태스크에서 페이지 폴트(page fault)가 발생하지 않도록 타임-크리티컬(time-critical) 메모리 페이지를 피닝(pinning)할 수 있다. TLB는 운영체제 및 애플리케이션의 가상 주소를 물리 주소로 변환할 때 발생하는 페이지 테이블 변환 오버헤드(page table translation overhead)를 줄이기 위한 제한된 하드웨어 리소스를 말한다. TLB 플러시는 내부적인 알고리즘에 의해서 동작하기 때문에 실시간 처리가 중요한 순간에 플러시가 발생하면 레이턴시(latency)를 유발할 수 있으므로, 프리엠티브 HW 매니저(160)는 설정된 현재 선점 모드가 완전 선점 모드이면 중요한 실시간성 애플리케이션이 항상 TLB에 포함될 수 있도록 동적으로 락다운할 수 있다.

[0042] 동적으로 선점 구간을 조정하는 장치(100)는 유저 레벨 서버 및 태스크 매니저(User level Server & task

manager)(170)를 더 포함할 수 있다. 유저 레벨 서버 및 태스크 매니저(170)는 프리엠션 매니저(110)에 의해 현재 선점 모드가 설정되면 설정된 현재 선점 모드에 맞도록 동적으로 유저 레이어의 프리엠터(preempter), 인터럽트 스레드, 페이저(pager), 스케쥴러 등의 수행방법을 조정할 수 있다.

[0043] 도 2는 일 실시예에 따라 운영체제에서 동적으로 선점 구간을 조정하는 방법의 예시도이다. 도 2를 참조하여 동적으로 선점 구간을 조정하는 방법을 구체적으로 설명하면, 먼저, 프리엠션 매니저(110)는 현재의 시스템 컨텍스트가 변경되는지를 모니터링하고(단계 110), 시스템 컨텍스트가 변경되면 그 시스템 컨텍스트를 분석한다(단계 120). 분석 결과 변경된 시스템 컨텍스트가 헬스케이 모니터링이나 운전자 감시, 카 크래시 알람 등의 애플리케이션이 실행되는 경우와 같이 처리 효율보다 실시간 처리가 매우 중요한 환경이면(단계 131) 현재 선점 모드를 완전 선점 모드로 설정할 수 있다(단계 141).

[0044] 만약, 분석결과 그 시스템 컨텍스트가 임베디드 서버나 데이터 스트리밍과 같이 실시간 처리보다 높은 처리 효율성이 요구되는 애플리케이션이 실행되는 환경이면(단계 132) 현재 선점 모드를 비선점 모드로 설정할 수 있으며(단계 142), 그러지 않으면 현재 선점 모드를 일반 선점 모드로 설정할 수 있다(단계 143). 다만, 전술한 바와 같이 선점 모드는 컨텍스트에 따라 다양한 단계의 모드로 세분화될 수 있으므로, 변경된 시스템 컨텍스트가 매우 높은 실시간 처리 요구 환경이거나 처리 효율성이 매우 높은 환경이 아닌 일반적인 환경, 즉, 다양한 애플리케이션들이 혼재되어 있는 환경에서는 일반 선점 모드를 더 세분화하여 설정할 수 있다.

[0045] 도 3은 일 실시예에 따라 ISR 수행 방법을 동적으로 조정하는 방법을 나타낸 것이다. 프리엠션 매니저(110)에 의해 설정된 현재 선점 모드에 따라 ISR 수행 방법을 동적으로 조정하는 절차에 대해 설명한다. 먼저, 인터럽트 컨텍스트 매니저는 하드웨어에서 인터럽트가 발생되면(단계 210) 그 인터럽트를 인터럽트 벡터(Interrupt vector)로 분기한다(단계 220). 그 다음, 인터럽트 전처리기(interrupt pre-processing)를 통해 인터럽트 컨텍스트 모드에서 현재 선점 모드를 확인하고(단계 230), 현재 선점 모드가 비선점 모드이면 실시간 처리가 중요하지 않은 환경이므로 그대로 인터럽트 컨텍스트 모드에서 ISR을 처리한다(단계 280).

[0046] 만약, 현재 선점 모드가 일반 선점 모드이거나 완전 선점 모드이면 인터럽트 스레드 모드에서 ISR이 처리되도록 인터럽트 스레드를 활성화한다(단계 250). 그 다음, 현재 인터럽트 스레드를 우선순위에 따라 재스케쥴링 할지를 판단하고(단계 260) 재스케쥴이 필요하면 우선순위를 스케쥴링하고 ISR을 수행한다(단계 270). 인터럽트 스레드를 이용하면 인터럽트 자체가 스레드로 처리되기 때문에 선점이 언제든지 가능하고 우선순위에 따라 스케줄링을 변경할 수도 있기 때문에 처리 효율면에서 오버헤드는 다소 존재하지만 실시간 처리가 향상될 수 있다.

[0047] 도 4는 일 실시예에 따라 프리엠션 포인트 수행 절차를 동적으로 조정하는 방법을 나타낸 것이다. 프리엠션 포인트 매니저(130)는 실행 시간이 오래 소요되는 커널 서비스 루틴의 초반에 선점을 검사하는 프리엠션 포인트를 삽입할 수 있다(단계 310). 프리엠션 포인트는 현재 선점 모드에 따른 적절한 동작을 수행하기 위한 코드, 예를 들어, 현재 스레드의 재스케쥴링을 체크할지 여부를 동적으로 조정하는 코드를 포함할 수 있다. 즉, 프리엠션 포인트는 현재 선점 모드를 확인하고(단계 320), 현재 선점 모드가 비선점 모드이면 현재 스레드를 재스케쥴링하지 않고 그대로 리턴하여 이후 동작을 계속 수행하도록 하고(단계 350), 비선점 모드가 아니면 현재 스레드의 재스케쥴 여부를 체크하여 재스케쥴링이 필요하면(단계 330) 현재 스레드를 재스케쥴링 한다(단계 340). 따라서, 현재 선점 모드가 비선점 모드일 경우 매번 재스케쥴링 여부를 확인할 필요가 없으므로 실시간 처리는 다소 저하되지만 처리 효율은 향상시킬 수 있다.

[0048] 도 5는 일 실시예에 따라 시스템의 락 방법이 동적으로 조정되는 예시도이다. 프리엠티브 락 매니저(140)는 커널이 스핀락을 가진 태스크를 수행할 때 설정된 현재 선점 모드에 따라 동적으로 스핀락 또는 뮤텍스 중 어느 하나를 사용하여 그 태스크를 수행하도록 변경할 수 있다. 예를 들어, 프리엠티브 락 매니저(140)는 설정된 현재 선점 모드가 완전 선점 모드이면 스핀락 대신에 뮤텍스를 사용하도록 조정할 수 있다. 즉, 스핀락을 가지고 있는 낮은 우선 순위의 태스크가 수행중일 때 동일한 코드를 수행하는 실시간성 태스크가 동일한 스핀락을 필요로 하지 않을 경우 이미 락을 가진 태스크를 선점하여 수행하도록 할 수 있다. 뮤텍스는 처리 효율면에서는 오버헤드가 존재하므로 프리엠션 매니저(110)에 의해 현재 선점 모드가 비선점 모드로 설정되면, 프리엠션 락 매니저(140)는 뮤텍스 코드를 스핀락 코드로 포인팅 되도록 조정하여 바쁜 대기(busy waiting)를 하지만 처리 효율면에서는 성능이 상대적으로 우수한 스핀락 코드를 수행하도록 할 수 있다.

[0049] 도 6은 일 실시예에 따라 시스템 콜 수행 방법을 동적으로 조정하는 예시도이다. 프리엠티브 IPC/SysCall 매니저(150)는 IPC 처리시 설정된 현재 선점 모드에 따라 IPC의 타임 아웃을 동적으로 조정할 수 있다. 또한, 시스템 콜이 수행되면 현재 선점 모드에 따라 동기식 또는 비동기식으로 수행되도록 동적으로 조정할 수 있다. 도 6을 참조하여 시스템 콜 수행시 현재 선점 모드에 따라 시스템 콜을 수행하는 절차를 설명한다. 먼저, 시스템 콜

이 수행되면(단계 410), 설정된 현재 선점 모드를 확인한다(단계 420). 즉 설정된 현재 선점 모드가 완전 선점 모드로서 현재 수행된 태스크가 실시간 처리가 요구되는 태스크이면, 비동기적으로 시스템 콜을 수행하도록 조정하고(단계 440), 현재 선점 모드가 비선점 모드이면 동기적으로 시스템 콜이 수행되도록 조정할 수 있다(단계 450).

[0050] 도 7은 일 실시예에 따라 TLB 락다운을 동적으로 설정하는 방법의 예시도이다. 프리엠티브 HW 매니저(160)는 설정된 현재 선점 모드에 따라 실시간성 태스크가 TLB에서 플러쉬 되지 않도록 동적으로 TLB 락다운을 설정하거나 해제할 수 있다. 또한, 프리엠티브 HW 매니저(160)는 실시간성 태스크에서 페이지 폴트(Page fault)가 발생하지 않도록 타임-크리티컬(time-critical) 메모리 페이지를 피닝(pinning)할 수 있다.

[0051] 좀 더 구체적으로 TLB 락다운을 동적으로 설정하는 방법을 설명하면, 실시간 처리가 요구되는 태스크가 등록되면(단계 510), TLB 락다운이 필요한지 판단한다(단계 520). 그 다음, 설정된 현재 선점 모드가 실시간 처리가 요구되는 완전 선점 모드이면 중요 실시간성 애플리케이션 또는 태스크가 항상 TLB에 포함될 수 있도록 현재 태스크가 높은 우선 순위인지를 판단하고(단계 530), 높은 우선 순위의 컨텍스트인 경우에는 그 태스크를 TLB 락다운에 설정한다(단계 540). 만약, TLB 락다운이 필요하지 않은 경우, 예를 들어, 설정된 현재 선점 모드가 비선점모드이거나, 높은 우선 순위의 컨텍스트가 아닌 경우에는 TLB 락다운을 설정하지 않고 일반적으로 처리되도록 할 수 있다(단계 550).

[0052] 본 발명이 속하는 기술분야의 통상의 지식을 가진 자는 본 발명이 그 기술적 사상이나 필수적인 특징을 변경하지 않고서 다른 구체적인 형태로 실시될 수 있다는 것을 이해할 수 있을 것이다. 그러므로 이상에서 기술한 실시예들은 모든 면에서 예시적인 것이며 한정적이 아닌 것으로 이해해야만 한다. 본 발명의 범위는 상기 상세한 설명보다는 후술하는 특허청구의 범위에 의하여 나타내어지며, 특허청구의 범위의 의미 및 범위 그리고 그 균등 개념으로부터 도출되는 모든 변경 또는 변형된 형태가 본 발명의 범위에 포함되는 것으로 해석되어야 한다.

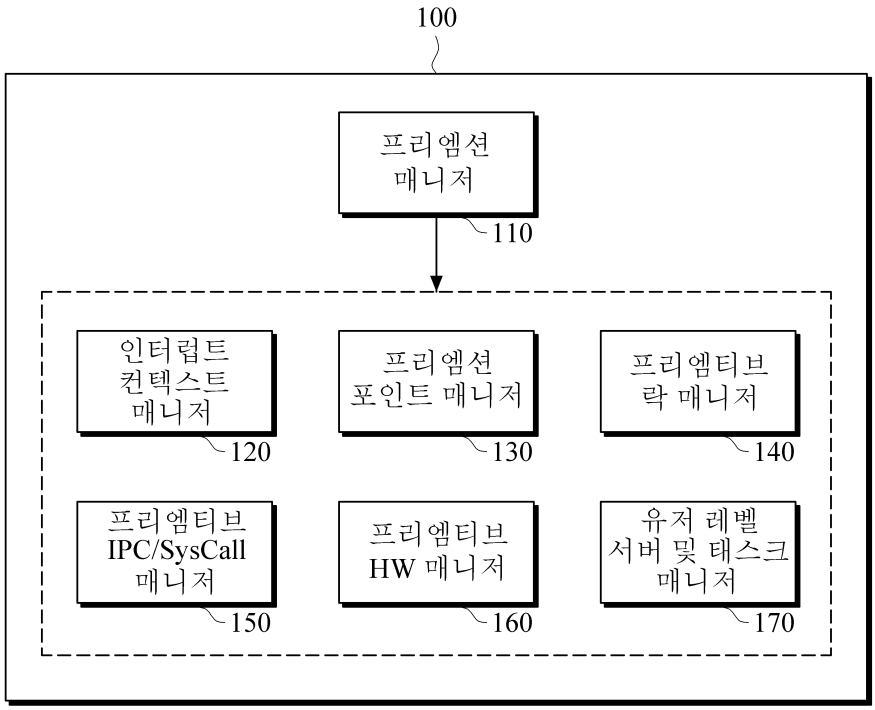
부호의 설명

[0053]

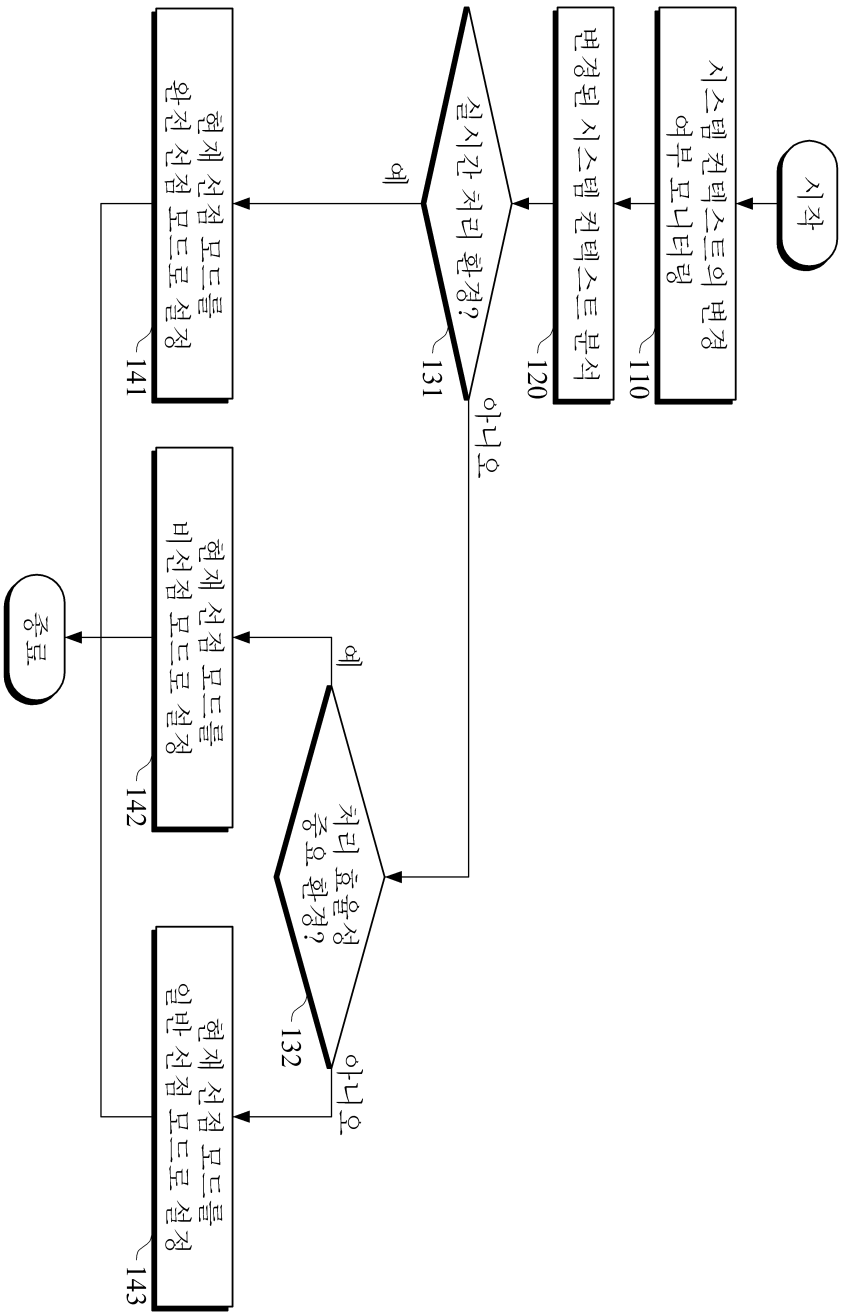
100: 선점 구간 조정 장치	110: 프리엠션 매니저
120: 인터럽트 컨텍스트 매니저	130: 프리엠션 포인트 매니저
140: 프리엠티브 락 매니저	150: 프리엠티브 IPC/SysCall 매니저
160: 프리엠티브 HW 매니저	170: 유저 레벨 서버 및 태스크 매니저

도면

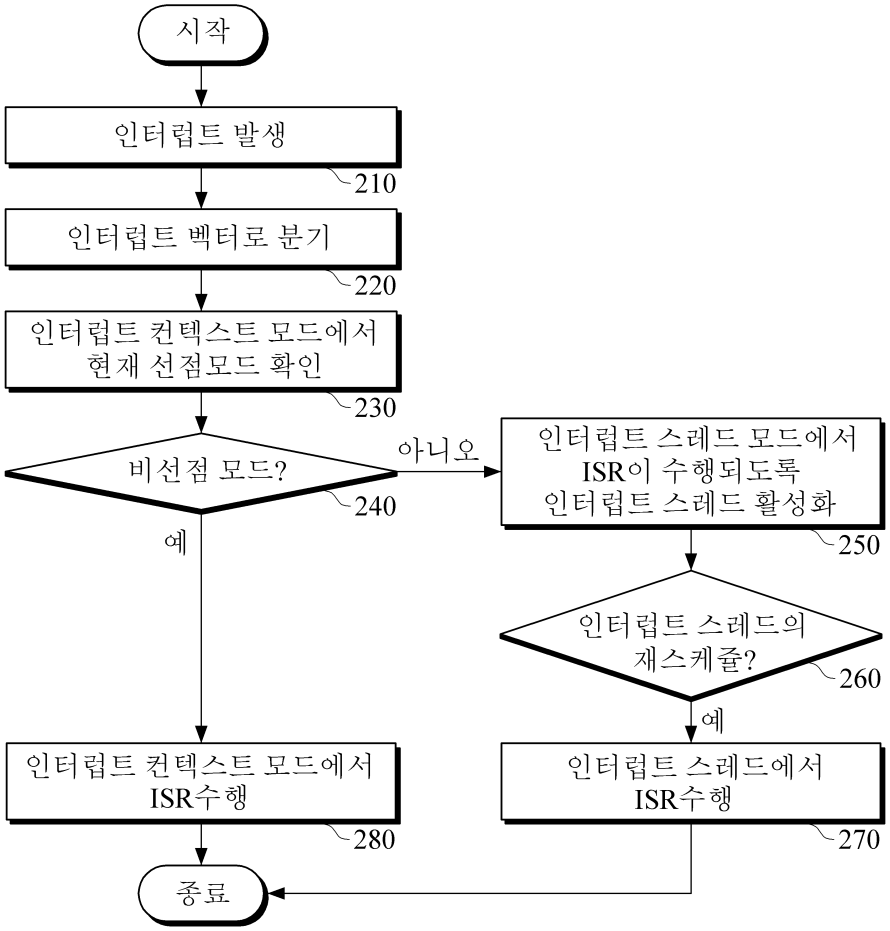
도면1


100
프리엠션
매니저
110
인터럽트
컨텍스트
매니저
120
프리엠션
포인트 매니저
130
프리엠티브
락 매니저
140
프리엠티브
IPC/SysCall
매니저
150
프리엠티브
HW 매니저
160
유저 레벨
서버 및 태스크
매니저
170

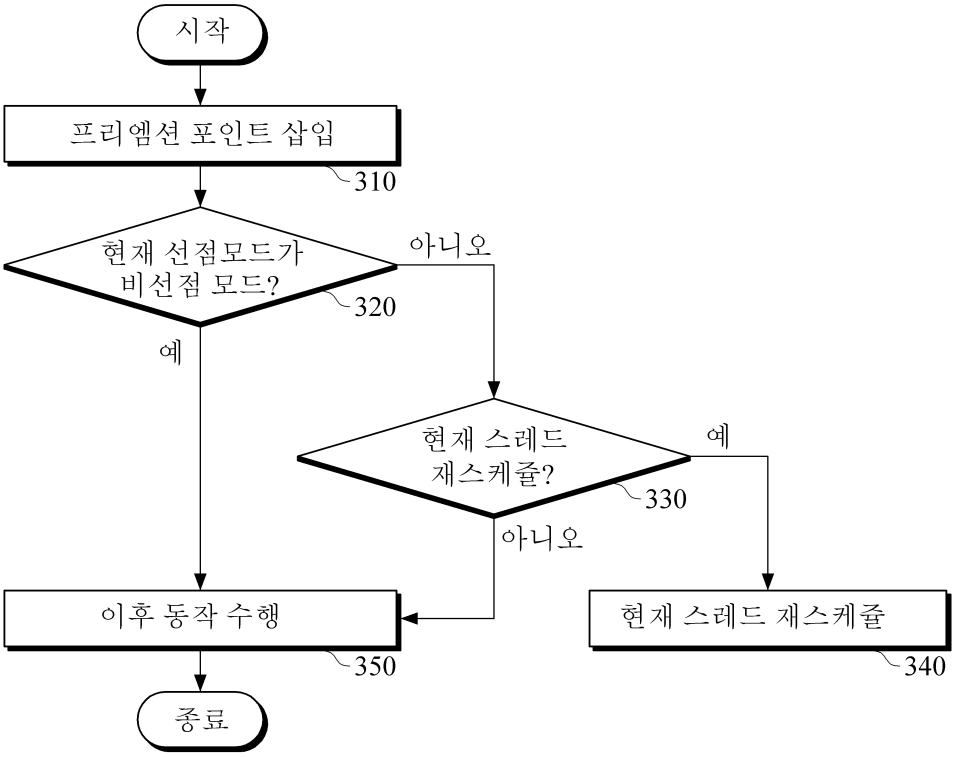
도면2


시작
시스템 컨텍스트의 변경 여부 모니터링
110
변경된 시스템 컨텍스트 분석
120
실시간 처리 환경?
131
예
아니오
현재 선점 모드를 완전 선점 모드로 설정
141
처리 효율성 중요 환경?
132
예
아니오
현재 선점 모드를 비선점 모드로 설정
142
현재 선점 모드를 일반 선점 모드로 설정
143
종료

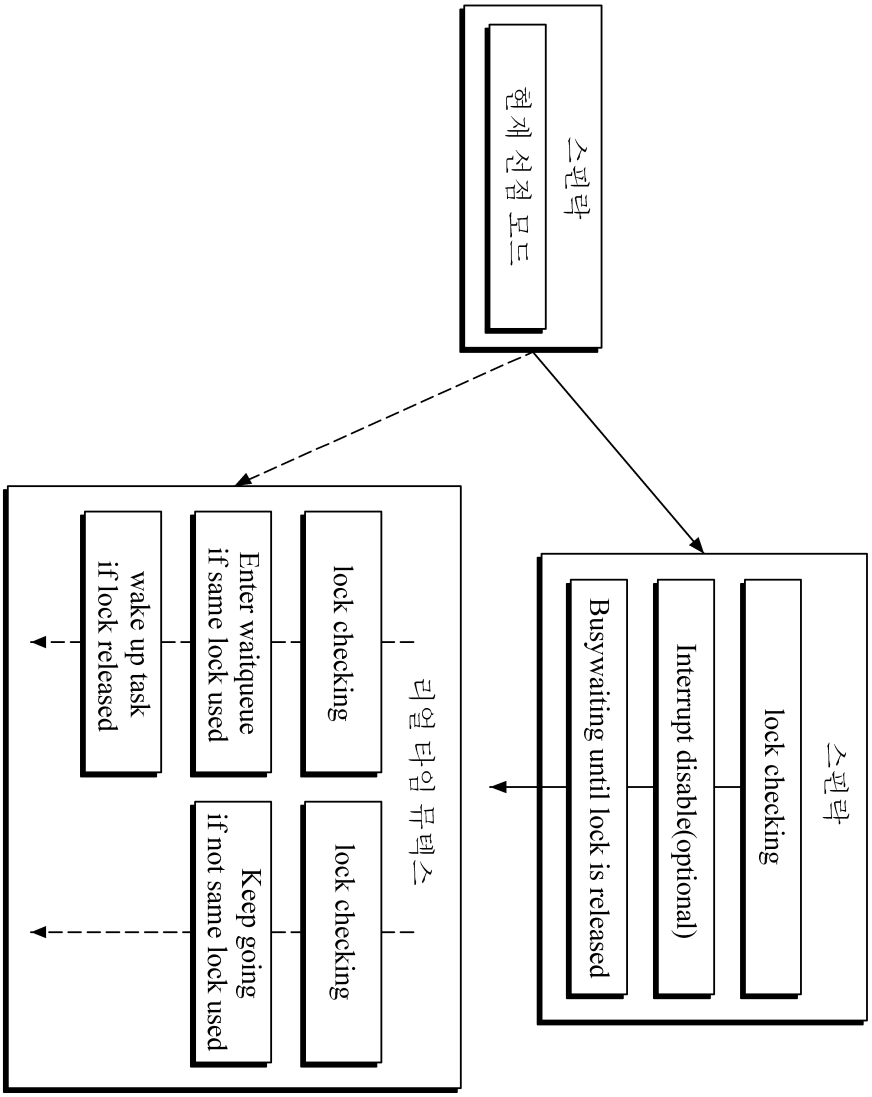
도면3


시작
인터럽트 발생
210
인터럽트 벡터로 분기
220
인터럽트 컨텍스트 모드에서
현재 선점모드 확인
230
비선점 모드?
240
아니오
예
인터럽트 스레드 모드에서
ISR이 수행되도록
인터럽트 스레드 활성화
250
인터럽트 스레드의
재스케줄?
260
예
인터럽트 컨텍스트 모드에서
ISR수행
280
인터럽트 스레드에서
ISR수행
270
종료

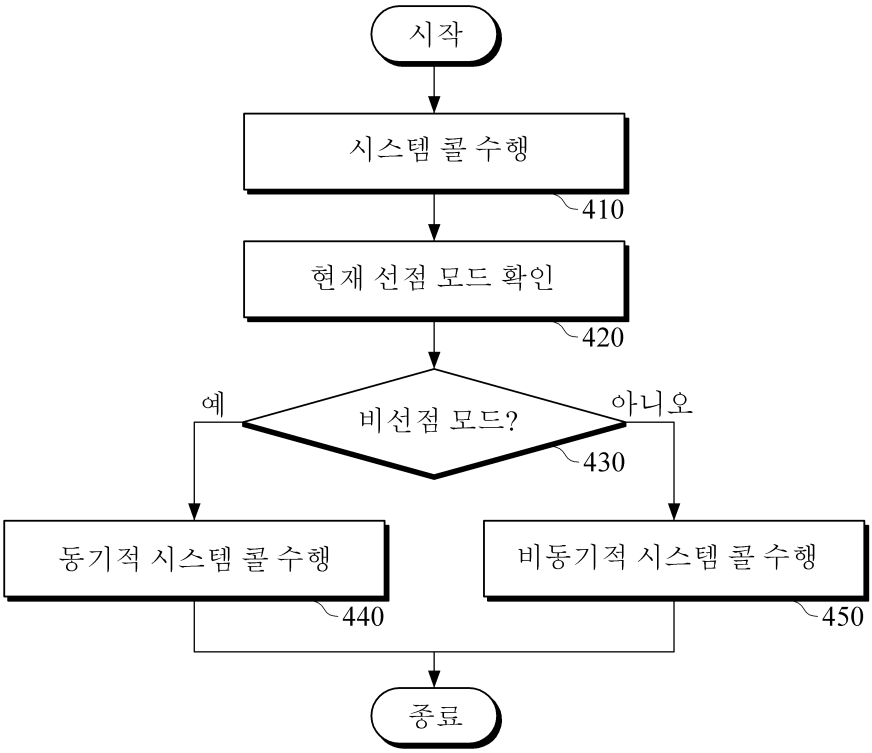
도면4


시작
프리엠션 포인트 삽입
310
현재 선점모드가 비선점 모드?
320
아니오
예
현재 스레드 재스케쥴?
330
예
아니오
이후 동작 수행
350
현재 스레드 재스케쥴
340
종료

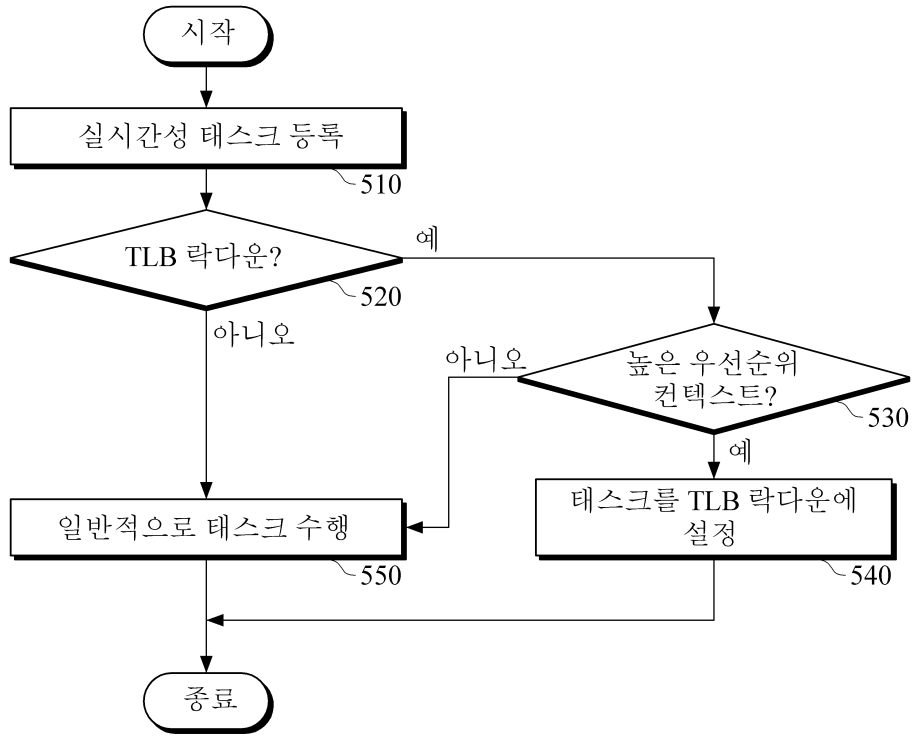
도면5


스핀락
현재 선점 모드
스핀락
lock checking
Interrupt disable(optional)
Busywaiting until lock is released
리얼 타임 뮤텍스
lock checking
Enter waitqueue
if same lock used
wake up task
if lock released
lock checking
Keep going
if not same lock used

도면6


시작
시스템 콜 수행
410
현재 선점 모드 확인
420
예
비선점 모드?
아니오
430
동기적 시스템 콜 수행
440
비동기적 시스템 콜 수행
450
종료

도면7


시작
실시간성 태스크 등록
510
TLB 락다운?
520
예
아니오
아니오
높은 우선순위
컨텍스트?
530
예
일반적으로 태스크 수행
550
태스크를 TLB 락다운에
설정
540
종료