(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 803 235**

(51) Int. CI.:

G06F 9/48 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **18.09.2016 E 16189369 (8)**

(97) Fecha y número de publicación de la concesión europea: **01.04.2020 EP 3296867**

(54) Título: **Método y aparato para ejecutar tareas en tiempo real**

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
25.01.2021

(73) Titular/es:

ESG ELEKTRONIKSYSTEM- UND LOGISTIK-GMBH (100.0%)
Livry-Gargan-Strasse 6
82256 Fürstenfeldbruck, DE

(72) Inventor/es:

ZLATANCHEV, IVAN y
HEIDSIECK, TILLMANN

(74) Agente/Representante:

ARIAS SANZ, Juan

DESCRIPCIÓN

Método y aparato para ejecutar tareas en tiempo real

Campo de invención

La presente invención se refiere a un método y a un aparato para ejecutar una pluralidad de tareas con restricciones en tiempo real.

Antecedentes

Un programa especifica un cálculo.

Un programa en tiempo real especifica un cálculo de una o más tareas que tienen restricciones en tiempo real.

Una tarea es una porción de un cálculo de un programa.

Una restricción en tiempo real especifica que el cálculo de una tarea debe haber avanzado hasta un determinado punto, que puede ser, por ejemplo, el final de la tarea, dentro de una duración predefinida en tiempo de reloj de pared. La duración predefinida en el tiempo también se conoce como el tiempo entre la activación de una tarea y un vencimiento de una tarea.

Cuando se ejecuta el programa en tiempo real, puede ejecutarse simultáneamente una pluralidad de tareas especificadas por el programa.

Una o más tareas en tiempo real pueden ejecutarse simultáneamente con tareas que no tienen restricciones en tiempo real. Estas últimas tareas también se denominan tareas no en tiempo real (tareas no en RT).

Una tarea en tiempo real (tarea en RT) se ejecuta normalmente en la misma unidad de ejecución desde el principio de dicha tarea hasta el final de dicha tarea.

Cuando una pluralidad de tareas se ejecutan en la misma unidad de ejecución, la ejecución simultánea significa que la pluralidad de tareas usan la unidad de ejecución de una manera de compartición de tiempo. Compartición de tiempo se refiere a la asignación de cada tarea de las tareas simultáneas a la unidad de ejecución. La compartición de tiempo se controla por un planificador. El planificador puede ser un planificador de hardware o de software o una combinación de los mismos. Preferiblemente, el planificador se controla por el sistema operativo. El sistema operativo es preferiblemente un sistema operativo en tiempo real.

Cuando se hace referencia a “tiempo” en el contexto de cálculos en tiempo real, se distinguen dos conceptos diferentes: el primer concepto de “tiempo” se refiere al “tiempo de ejecución”, también denominado algunas veces “tiempo de CPU”, que se refiere al tiempo durante el cual una tarea usa realmente una unidad de ejecución. El segundo concepto de “tiempo” se refiere al “tiempo de reloj de pared”. Igualmente, el término “duración” se refiere al tiempo de reloj de pared, que es la diferencia entre un punto final y un punto inicial en tiempo de reloj de pared. Por ejemplo, el inicio de una tarea y el vencimiento de una tarea se especifican normalmente como puntos absolutos o relativos en tiempo de reloj de pared, es decir en tiempos de reloj de pared. El tiempo de reloj de pared entre el tiempo de activación de una tarea y su final también se denomina “tiempo de respuesta”. Cuando la unidad de ejecución no se comparte entre una pluralidad de tareas, es decir, sólo se asigna una tarea individual a la unidad de ejecución, los conceptos de “tiempo de reloj de pared” y “tiempo de ejecución” son idénticos suponiendo que la tarea individual siempre está activa. Una tarea está activa si está lista para ejecutarse, es decir, si no está esperando, por ejemplo, una entrada que puede proporcionarse, por ejemplo, por otras tareas. Cuando queda claro a partir del contexto, a continuación la redacción usa el término “tiempo”, de lo contrario se distingue explícitamente entre “tiempo de ejecución” y “tiempo de reloj de pared”.

De manera similar a la compartición de la unidad de ejecución, pueden compartirse otros recursos entre tareas simultáneas, en las que la compartición se controla igualmente por un elemento de asignación de recursos, que puede ser una unidad de un sistema operativo. La presente divulgación y las técnicas presentadas en el presente documento se refieren principalmente a la compartición de las unidades de ejecución, en la que el elemento de asignación de recursos es un planificador. Cuando la compartición afecta a recursos distintos de una unidad de ejecución, esto se menciona explícitamente en el presente documento. Los siguientes términos se usan normalmente en la teoría de tiempo real: “recurso activo” se refiere, por ejemplo, a una unidad de ejecución o bus de comunicación; “recurso pasivo” se refiere, por ejemplo, a variables o memoria compartida entre diferentes hilos de OS.

Una unidad de ejecución es normalmente un núcleo de procesador. Los procesadores modernos incluyen normalmente una pluralidad de núcleos y por tanto se denominan procesadores de múltiples núcleos o de muchos núcleos. La arquitectura de tales procesadores es habitualmente de tal manera que los núcleos comparten recursos comunes en el procesador, por ejemplo una memoria caché, un bus de comunicación y una interfaz de memoria. Este tipo de compartición debido a la arquitectura de un procesador de múltiples núcleos tiene el efecto de que la ejecución de una primera tarea en un primer núcleo puede afectar al transcurso de una ejecución de una segunda

tarea en un segundo núcleo de ejecución.

Para programas en tiempo real, este tipo de compartición de recursos basándose en la arquitectura de procesador afecta a la predicción del tiempo de ejecución de peor caso. Específicamente, el tiempo de ejecución de peor caso, WCET, que es un tiempo de ejecución en el sentido anterior y que se predice mediante métodos convencionales que tienen en cuenta cualquier interacción posible, y específicamente también extremadamente improbable, entre núcleos de procesador mediante su compartición de recursos de arquitectura, puede conducir a predicciones de WCET extremadamente pesimistas que están muy por encima de tiempos de ejecución de caso habitual realista. Esta gran discrepancia hace que los resultados de predicción de WCET determinados mediante métodos convencionales tengan poca utilidad en arquitectura de múltiples núcleos, ya que, cuando se proporcionan como entrada a un planificador, conducen a planificaciones en tiempo real muy pesimistas que no logran un uso de recursos eficiente. Aunque el problema es particularmente relevante para arquitecturas de múltiples núcleos tal como se describe, también se produce compartición de recursos cuando se ejecutan múltiples tareas simultáneas de una manera de compartición de tiempo en un procesador con una única unidad de ejecución, que es por ejemplo un núcleo de procesador, dado que, por ejemplo, la memoria caché asociada con la unidad de ejecución se comparte entre las tareas simultáneas.

Claire Pagetti, Christine Rochange: “Runtime monitoring of time-critical tasks in multicore systems” (disponible en http://materials.dagstuhl.de/files/15/15121/15121.ClairePagetti.ExtendedAbstract.pdf) describe un método que monitoriza tareas críticas en tiempo de ejecución, en el que, en caso de un retardo y por tanto una posible violación de propiedades en tiempo real, se interrumpen otras tareas menos críticas para permitir que la tarea crítica continúe su ejecución de una manera oportuna y eficiente, sin compartir recursos de la unidad de ejecución con otras tareas.

Michael Paulitsch: “Challenges for the Use of Multicore Processors in Mixed-Criticality Systems with Focus on Temporal Aspects”, RTAS 2014 (disponible en: http://2014.rtas.org/wp-content/uploads/Paulitsch.pdf), número de diapositiva 11: “WCET for Multi-Core Computers Combined with Monitoring” describe la posibilidad de ejecutar tareas con restricciones en tiempo real en procesadores de múltiples núcleos.

Claire Pagetti, Christine Rochange: “Runtime monitoring of time-critical tasks in multi-core systems - Mixed Criticality on Multicore/Manycore Platforms” (disponible en http://materials.dagstuhl.de/files/15/15121/15121.Christine-Rochange.Slides.pdf) enseña a usar estimaciones de WCET “menos seguras”, con una probabilidad <100% de que una tarea termine antes de su vencimiento, y a tomar acción correctora cuando “la situación es peor que la supuesta en el transcurso de análisis”, en puntos de monitorización colocados de manera óptima.

El documento US 2015 074674 A1 da a conocer un aparato para ajustar prioridades de tareas, que determina que una tarea está violando una restricción en tiempo real usando un resultado de determinación de perfil del software en tiempo real y detalles de tarea incluyendo una restricción en tiempo real para cada tarea, ajusta una prioridad de la tarea que viola la restricción en tiempo real o una tarea candidato superior próxima a la tarea que viola la restricción en tiempo real, y simula la ejecución del software en tiempo real dependiendo de la prioridad ajustada.

Para resumir el problema anteriormente mencionado, la alta discrepancia entre WCET predicho y los tiempos de ejecución de caso habitual en arquitecturas de procesador modernas hace que las técnicas convencionales de WCET sean ineficientes para controlar la ejecución en tiempo real y decisiones de planificación en tales arquitecturas. La presente divulgación aborda este problema.

Sumario de la invención

La presente invención se refiere a un método para ejecutar un programa que incluye una pluralidad de tareas, en el que una o más tareas de la pluralidad de tareas tienen restricciones en tiempo real, comprendiendo el método las siguientes etapas para cada tarea T_x con restricciones en tiempo real: (a) determinar un modelo de referencia en tiempo real, en el que el modelo de referencia en tiempo real de la tarea T_x incluye una pluralidad de las microtareas μT_{xi}, $i \in \{1,..., n\}$, que son una división de la tarea T_x, y un orden entre las microtareas μT_{xi} según todos los trayectos de ejecución posibles de la tarea T_x, en el que para cada microtarea μT_{xi}, $i \in \{1,..., n\}$, se determina un micropresupuesto μB_{xi} que es más pequeño que el tiempo de ejecución de peor caso, WCET, de la microtarea μT_{xi}; y en el que, para cada microtarea μT_{xi}, $i \in \{1,..., n\}$, basándose en los micropresupuestos μB_{xk}, $k \in \{1,..., n\}$, se determina un transcurso de referencia que especifica un transcurso estimado de la microtarea μT_{xi} en cualquier ejecución posible de la tarea T_x, de tal manera que todas las continuaciones posibles de ejecuciones de la tarea T_x desde la microtarea μT_{xi} en adelante cumplen las restricciones en tiempo real de la tarea T_x con una probabilidad por encima de un límite de tolerancia, en el que las restricciones en tiempo real de la tarea T_x se cumplen con una probabilidad por encima del límite de tolerancia si la ejecución de la tarea T_x se completa antes de un vencimiento de la tarea T_x con una probabilidad inferior al 100% y por encima de una determinada garantía de servicio mínima; (b) ejecutar la pluralidad de tareas y (b1) determinar después de la ejecución de la microtarea μT_{xi} un transcurso real; (b2) comparar el transcurso real con el transcurso de referencia; (b3) basándose en la comparación, si se determina que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia, aumentar la prioridad de la tarea T_x. Esto tiene el efecto técnico y la ventaja de que la prioridad de tareas en tiempo real puede ajustarse según el transcurso real de la tarea, en el que puede aumentarse la prioridad de tareas retardadas. Por tanto, puede evitarse una situación en la que una tarea que ha avanzado suficientemente usa

la unidad de ejecución, mientras que otras tareas permanecen retardadas dado que no está planificado que se ejecuten.

Según una realización, las microtareas μT_{xi} i ∈ {1,..., xÚltima} forman un entramado con μT_{x1} como microtarea inicial de T_x y $\mu T_{xÚltima}$ como microtarea final de T_x, el micropresupuesto μB_{xi} especifica un tiempo de ejecución para completar la ejecución de la microtarea μT_{xi} con una probabilidad inferior al 100% y por encima de un umbral de probabilidad predeterminado, y el micropresupuesto μB_{xi} de una microtarea μT_{xi} se determina preferiblemente basándose en análisis estadístico y/o interpretación abstracta de un programa de μT_{xi} y/o análisis estadístico de ejecuciones de μT_{xi}.

Esto tiene el efecto técnico y la ventaja de que los micropresupuestos y por consiguiente los presupuestos de tareas son menos conservativos que los presupuestos estimados mediante análisis de WCET convencional. Si se usan presupuestos determinados según la realización para la asignación de recursos por un planificador, puede reducirse el riesgo de sobreaprovisionamiento de recursos y por tanto puede mejorarse la eficiencia de asignación de recursos y uso de recursos.

Según una realización, el transcurso de referencia de la microtarea μT_{xi} incluye un microvencimiento μD_{xi}, que especifica un tiempo de respuesta hasta el que debe terminarse una ejecución de la microtarea μT_{xi}, en el que el tiempo de respuesta es una duración con respecto a un tiempo de activación ATT_x de la tarea T_x; la microtarea μT_{xi} debe terminarse preferiblemente hasta que cada una de las microtareas μT_{xk} k ∈ {1,..., i} en un trayecto crítico desde una microtarea inicial μT_{x1} hasta una microtarea μT_{xi} ha terminado la ejecución, en el que el tiempo de ejecución de cada microtarea μT_{xk} se estima mediante su micropresupuesto μB_{xk}; las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia si el transcurso real al final de la microtarea μT_{xi} supera el tiempo en el que la microtarea μT_{xi} debería haberse terminado preferiblemente; y el trayecto crítico hasta la microtarea μT_{xi} es un trayecto entre todos los trayectos de ejecución posibles de T_x desde la microtarea inicial μT_{x1} hasta la microtarea μT_{xi} que tiene el tiempo de ejecución predicho más largo.

Según una realización, el microvencimiento μD_{xi} es al menos la suma de los micropresupuestos μB_{xi} de las microtareas μT_{xk}, k ∈ {1,..., i} en el trayecto crítico hasta la microtarea μT_{xi}.

Según una realización, el transcurso de referencia de la microtarea μT_{xi} incluye un presupuesto de activación planificado B_{WCETxi} que especifica un presupuesto de tiempo de ejecución que es suficiente para completar la ejecución de la tarea T_x empezando desde la microtarea μT_{xi} de tal manera que sus restricciones en tiempo real se cumplen con una probabilidad por encima del límite de tolerancia; el presupuesto de tiempo de ejecución se determina basándose en los micropresupuestos μB_{xk} de cada una de las microtareas μT_{xk}, k ∈ {i,..., xÚltima} en un trayecto crítico activo dentro de T_x empezando en la microtarea μT_{xi}; el trayecto crítico activo empezando en la microtarea μT_{xi} es un trayecto entre todos los trayectos de ejecución posibles de T_x desde μT_{xi} hasta una microtarea final $\mu T_{xÚltima}$ que tiene el tiempo de ejecución predicho más largo; las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia si, antes de la ejecución de la microtarea μT_{xi}, el tiempo de respuesta real de la microtarea μT_{xi-1} es mayor que el microvencimiento μD_{xi-1}.

Según una realización, las microtareas en una ejecución de la tarea T_x se clasifican en microtareas en tiempo real flexible μT_{xi}, i ∈ {1,..., xrt-1} y microtareas en tiempo real estricto μT_{xi}, i ∈ {xrt, ...,xÚltima}, en el que un tiempo de ejecución de una microtarea en tiempo real flexible μT_{xi} se estima mediante su micropresupuesto μB_{xi}; y en el que un tiempo de ejecución de una microtarea en tiempo real estricto μT_{xi} se estima mediante su micropresupuesto μB_{xi} más un tiempo de tampón $BT(\mu T_{xi})$, siendo el tiempo de tampón un tiempo adicional para garantizar que μT_{xi} termina con una certeza del 100% dentro del tiempo estimado; y en el que el presupuesto de tiempo de ejecución se determina basándose además en una suma de los tiempos de ejecución estimados de microtarea en tiempo real flexible y en tiempo real estricto μT_{xi}.

Según una realización, el método comprende la siguiente etapa adicional: añadir una o más instrucciones a un programa de la tarea T_x, provocando las instrucciones la emisión de un acontecimiento de seguimiento E_{xi}, al final de la ejecución de la microtarea μT_{xi}, comprendiendo el acontecimiento de seguimiento un identificador único de una porción de la ejecución de la tarea T_x, en el que el identificador único comprende preferiblemente un identificador de una unidad de hardware que ejecuta la tarea T_x, un identificador de la tarea T_x y un identificador del acontecimiento de seguimiento E_{xi}.

Según una realización, la etapa (b1) del método comprende además las etapas de: determinar, para una unidad de ejecución que ejecuta la pluralidad de tareas, un estado en tiempo real, real, parcial, que comprende, para cada tarea T_x, el acontecimiento de seguimiento emitido más recientemente E_{xi} que incluye un punto en el tiempo CT_{Exi} en el que se emitió el acontecimiento de seguimiento E_{xi}; y determinar una diferencia $\Delta S_{\mu Txi} = CT_{Exi-1} - \mu D_{xi-1}$ entre un tiempo de activación real CT_{Exi-1} de la microtarea μT_{xi} y un tiempo de activación planificado de la microtarea μT_{xi}, en el que el tiempo de activación planificado de la microtarea μT_{xi} es el microvencimiento μD_{xi-1} de la microtarea anterior μT_{xi-1}.

Según una realización, la etapa (b2) comprende además determinar un presupuesto de activación real de la microtarea μT_{xi} que es el presupuesto de activación planificado $B_{WCET,xi}$ corregido por $\Delta S_{\mu Txi}$.

Según una realización, en la etapa (b3), la prioridad de una tarea T_x se aumenta de tal manera que cuanto menor es la diferencia D_x - CT - B_{WCETxi} mayor es la prioridad de T_x en la que D_x especifica el vencimiento de la tarea T_x en tiempo de reloj de pared, CT especifica el tiempo de reloj de pared actual, y B_{WCETxi} es el presupuesto de activación real en tiempo de ejecución de la microtarea μT_{xi}, cuando el acontecimiento de seguimiento E_{xi} es el acontecimiento de seguimiento emitido más recientemente de la tarea T_x.

Según una realización, la secuencia de etapas (b1), (b2) y (b3) se repite en intervalos predeterminados hasta que se termina la ejecución de la pluralidad de tareas en tiempo real, y los intervalos predeterminados son preferiblemente regulares.

Según una realización, determinar el modelo de referencia en tiempo real de la tarea T_x comprende determinar ejecuciones posibles de T_x usando métodos de análisis de tiempo de ejecución de peor caso, WCET, de un programa de T_x, en el que los métodos de análisis de WCET comprenden preferiblemente: determinar un gráfico de flujo de control del programa de la tarea T_x, determinar intervalos de valores viables para variables de la tarea T_x, determinar un número máximo de iteraciones de bucle, modelizar memoria caché y acceso de memoria, y determinar trayectos críticos en el gráfico de flujo de control, y en el que el gráfico de flujo de control incluye todos los trayectos de ejecución posibles de la tarea T_x.

Según una realización, cada tarea de la pluralidad de tareas se asigna a una unidad de ejecución fija durante un periodo de planificación y la unidad de ejecución fija es preferiblemente un núcleo de una pluralidad de núcleos de un procesador de múltiples núcleos, y se reserva tiempo de ejecución en la unidad de ejecución fija según tiempos de ejecución estimados de todas las microtareas μT_{xi} de todas las tareas en tiempo real T_x asignadas a la unidad de ejecución fija, en el que la reserva se realiza preferiblemente por un planificador que es un planificador de OS.

Según una realización, el presupuesto B_{WCETx} de la tarea T_x es el presupuesto de activación planificado B_{WCETx1} de la microtarea inicial μT_{xi} de la tarea T_x, en el que puede asignarse una pluralidad de tareas en una misma unidad de ejecución siempre que se cumplan las siguientes restricciones: la suma de tiempos de ejecución estimados para las microtareas en tiempo real estricto de cada tarea de la pluralidad de tareas no supera una determinada primera porción de un uso máximo de la unidad de ejecución durante el periodo de planificación; y la suma de los presupuestos de tareas en tiempo real asignadas a la misma unidad de ejecución no supera una determinada segunda porción del uso máximo de la unidad de ejecución durante el periodo de planificación.

Según una realización, si una diferencia entre un tiempo de activación real de la microtarea μT_{xi} y un tiempo de activación planificado de la microtarea μT_{xi} es negativa, se libera una porción del tiempo de ejecución dentro del periodo de planificación reservada en la unidad de ejecución para la ejecución de la tarea T_x con restricciones en tiempo real y por tanto está disponible para la ejecución de otras tareas, en el que el tiempo de activación planificado de la microtarea μT_{xi} es el microvencimiento μD_{xi-1} de la microtarea anterior μT_{xi-1}, y en el que la cantidad de tiempo liberado es inferior o igual a una diferencia entre el tiempo real restante hasta el vencimiento de T_x y el presupuesto de activación planificado B_{WCETxi}.

La presente invención se refiere además a un método para ejecutar una pluralidad de tareas, en el que una o más tareas tienen restricciones en tiempo real, basándose en un modelo de referencia de cada tarea con restricciones en tiempo real, en el que el modelo de referencia se determina según la etapa (a) según una de las realizaciones especificadas anteriormente, en el que el método comprende las etapas (b1), (b2) y (b3) según una de las realizaciones especificadas anteriormente.

La presente invención se refiere además a un método para determinar un modelo de referencia en tiempo real de una tarea T_x, en el que dicha tarea has restricciones en tiempo real, comprendiendo el método la etapa (a) según una de las realizaciones especificadas anteriormente.

La presente invención se refiere además a un aparato para ejecutar un programa que incluye una pluralidad de tareas, en el que una o más tareas de la pluralidad de tareas tienen restricciones en tiempo real, comprendiendo el aparato las siguientes unidades de hardware: una o más unidades de ejecución adaptadas para ejecutar la pluralidad de tareas; una unidad de calibración adaptada para determinar un modelo de referencia en tiempo real, en el que el modelo de referencia en tiempo real de la tarea T_x incluye una pluralidad de las microtareas μT_{xi}, $i \in \{1, ..., n\}$, que son una división de la tarea T_x, y un orden entre las microtareas μT_{xi} según todos los trayectos de ejecución posibles de la tarea T_x, en el que, para cada microtarea μT_{xi}, $i \in \{1,..., n\}$, se determina un micropresupuesto μB_{xi} que es más pequeño que el tiempo de ejecución de peor caso, WCET, de la microtarea μT_{xi}; y en el que, para cada microtarea μT_{xi}, $i \in \{1,..., n\}$, basándose en los micropresupuestos μB_{xk}, $k \in \{1,..., n\}$, se determina un transcurso de referencia que especifica un transcurso estimado de la microtarea μT_{xi} en cualquier ejecución posible de la tarea T_x, de tal manera que todas las continuaciones posibles de ejecuciones de la tarea T_x desde la microtarea μT_{xi} en adelante cumplen las restricciones en tiempo real de la tarea T_x con una probabilidad por encima de un límite de tolerancia, en el que las restricciones en tiempo real de la tarea T_x se cumplen con una probabilidad por encima del límite de tolerancia si la ejecución de la tarea T_x se completa antes de un vencimiento de la tarea T_x con una

probabilidad inferior al 100% y por encima de una determinada garantía de servicio mínima; una unidad de monitorización de acontecimientos adaptada para determinar después de la ejecución de la microtarea μT_{xi} un transcurso real; una unidad de monitorización de tiempo de presupuesto adaptada para comparar el transcurso real con el transcurso de referencia; una unidad de planificación de hardware adaptada para aumentar la prioridad de la tarea T_x basándose en un resultado de comparación de la unidad de monitorización de tiempo de presupuesto, si se determina que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia.

Según una realización, la unidad de monitorización de acontecimientos está adaptada además para mantener, para cada tarea en tiempo real T_x, un acontecimiento de seguimiento emitido más recientemente E_{xi} que incluye un punto en el tiempo CT_{Exi} en el que se emitió el acontecimiento de seguimiento E_{xi}; el aparato comprende además una unidad de monitorización de vencimiento adaptada para estimar una diferencia $\Delta S_{\mu Txi} = CT_{Exi-1} - \mu D_{xi-1}$ entre un tiempo de activación real CT_{Exi-1} de la microtarea μT_{xi} y un tiempo de activación planificado de la microtarea μT_{xi}, y/o para detectar si una ejecución de una microtarea μT_{xi} termina después del microvencimiento μD_{xi}; la unidad de monitorización de tiempo de presupuesto está adaptada además para determinar, para cada tarea en tiempo real T_x, una desviación entre un transcurso planificado de la tarea T_x y un transcurso real de la tarea T_x, en el que el transcurso planificado de la tarea T_x antes de la ejecución de la microtarea μT_{xi} es el presupuesto de activación planificado B_{WCETxi}, y en el que el transcurso real de la tarea T_x se estima basándose en una cantidad de tiempo de CPU usado por la tarea T_x hasta el tiempo de respuesta CT_{Exi-1} y la diferencia $\Delta S_{\mu Txi}$; y la unidad de planificación de hardware está adaptada además para generar un valor de control en tiempo real para una tarea en tiempo real T_x basándose en una desviación del transcurso planificado de T_x con respecto al transcurso real de la tarea T_x, y en el que el valor de control en tiempo real se señaliza a un planificador de OS.

Según una realización, la unidad de calibración está adaptada además para llevar a cabo mediciones de tiempo de ejecución de una microtarea; para determinar, basándose en las mediciones, información sobre el tiempo de ejecución de la microtarea; y para almacenar la información en el modelo de referencia en tiempo real.

Según una realización, la información sobre el tiempo de ejecución de una microtarea es una distribución de probabilidad de un tiempo de ejecución entre un acontecimiento de seguimiento que marca un inicio de la microtarea y un seguimiento posterior que marca un final de la microtarea.

La presente invención se refiere además a un aparato para ejecutar un programa que incluye una pluralidad de tareas, en el que una o más tareas de la pluralidad de tareas tienen restricciones en tiempo real, comprendiendo el aparato: una pluralidad de unidades de ejecución adaptadas para ejecutar la pluralidad de tareas; una unidad de planificación en tiempo real que contiene un modelo de referencia en tiempo real que incluye un presupuesto de núcleo planificado B_{COREy} para cada unidad de ejecución $CORE_y$ de la pluralidad de unidades de ejecución, en el que el presupuesto de núcleo planificado B_{COREy} de una unidad de ejecución $CORE_y$ especifica un límite superior para el tiempo de ejecución que se requiere para completar todas las tareas en RT activas a tiempo con la garantía de servicio mínima, en el que el presupuesto de núcleo planificado puede estimarse como un uso máximo de una unidad de ejecución durante cada periodo de planificación para microtareas en la categoría de RT estricto o RT flexible respectivamente a lo largo de todos los periodos de planificación considerados durante la fase de calibración de un programa que incluye las tareas en tiempo real, en el que una tarea en tiempo real T_x está activa si se cumplen las dos condiciones siguientes: en primer lugar, ya se ha empezado la ejecución de la tarea T_x, es decir, se ha emitido el acontecimiento de seguimiento E_{x0} y, en segundo lugar, el último acontecimiento emitido no es $E_{xÚltimo}$, es decir, aún no se ha terminado la tarea T_x; alternativamente, también puede determinarse B_{COREy} usando métodos convencionales de análisis de capacidad de planificación basados en micropresupuestos; una unidad de monitorización de tiempo de presupuesto adaptada para determinar, para cada $CORE_y$ de la pluralidad de unidades de ejecución, un presupuesto de núcleo real, que es una reserva de tiempo de ejecución para tareas en tiempo real asignadas a $CORE_y$, y posibles desviaciones entre dicho presupuesto de núcleo real y el presupuesto de núcleo planificado B_{COREy}, en el que el presupuesto de núcleo real es el tiempo de ejecución en $CORE_y$ que se reserva en un determinado punto de tiempo dentro del periodo de planificación, que preferiblemente se estima, por ejemplo, basándose en los micropresupuestos μB_{xi} de todas las microtareas μT_{xi} de las tareas en tiempo real T_x asignadas a $CORE_y$ que están activas en cualquier punto en el tiempo dentro del periodo de planificación; y un gestor de interferencias de núcleo adaptado para enviar señales de penalización de núcleo si el presupuesto de núcleo real para $CORE_y$ supera el presupuesto de núcleo planificado B_{COREy}, en el que las señales de penalización de núcleo se envían a una o más de otras unidades de ejecución $CORE_z$ para las que un presupuesto de núcleo planificado B_{COREz} supera un presupuesto de núcleo real para $CORE_z$, provocando las señales de penalización de núcleo, cuando se reciben por la una o más de otras unidades de ejecución, que la una o más de otras unidades de ejecución se desprioricen durante un periodo predefinido de tiempo de reloj de pared.

Según una realización, despriorizar una unidad de ejecución incluye detener la unidad de ejecución durante un periodo predefinido de tiempo de reloj de pared.

Esto tiene el efecto y la ventaja de que una unidad de ejecución que está temporalmente detenida no compite por recursos compartidos, por ejemplo en recursos de chip compartidos entre unidades de ejecución en el mismo chip, de modo que otras unidades de ejecución pueden acceder a, y usar, tales recursos compartidos con menos retardo.

Según una realización, el presupuesto de núcleo planificado B_{COREy} incluye un presupuesto de núcleo planificado para microtareas en tiempo real estricto $HardB_{COREy}$ en $CORE_y$ y un presupuesto de núcleo planificado para microtareas en tiempo real flexible $SoftB_{COREy}$ en $CORE_y$, en el que $HardB_{COREy}$ se estima basándose en presupuestos planificados de la microtarea μT_{xi} en la categoría de tiempo real estricto para todas las tareas en tiempo real activas T_x asignadas a $CORE_y$, y en el que $SoftB_{COREy}$ se estima basándose en presupuestos planificados de la microtarea μT_{xi} en la categoría de tiempo real flexible para todas las tareas en tiempo real activas T_x asignadas a $CORE_y$.

Según una realización, una señal de penalización de núcleo dirigida a una unidad de ejecución de la una o más de otras unidades de ejecución se envía únicamente en un punto en el tiempo que no es crítico para la unidad de ejecución.

Esto tiene la ventaja y el efecto de que unidades de ejecución que ejecutan tareas en tiempo real que en sí mismas están retardadas no se ven penalizadas, es decir, despriorizadas.

Según una realización, un punto en tiempo de reloj de pared en una unidad de ejecución a la que se dirige la señal de penalización de núcleo no es crítico si dicha unidad de ejecución no está reservada para la ejecución de microtareas en tiempo real estricto a partir de dicho punto en tiempo de reloj de pared durante el periodo predefinido de tiempo de reloj de pared.

Según una realización, la unidad de planificación de hardware está adaptada además para, si una diferencia entre el tiempo de activación real de la microtarea μT_{xi} y un tiempo de activación planificado de la microtarea μT_{xi} es negativa, liberar una porción del tiempo de ejecución, reservada dentro del periodo de planificación en la unidad de ejecución para la ejecución de una tarea T_x con restricciones en tiempo real, y por tanto hacer que la porción esté disponible para la ejecución de otras tareas, en el que el tiempo de activación planificado de la microtarea μT_{xi} es el microvencimiento μD_{xi-1} de la microtarea anterior μT_{xi-1}, y en el que la cantidad de tiempo liberado es inferior o igual a una diferencia entre el tiempo real restante hasta el vencimiento de T_x y el presupuesto de activación planificado B_{WCETxi}.

Esto tiene el efecto técnico y la ventaja de que tiempo de ejecución que se ha reservado pero no se ha usado por tareas en tiempo real puede liberarse de una manera oportuna incluso antes de que termine la tarea en tiempo real de modo que otras tareas pueden usar el recurso. En conjunto, esto conduce a un uso de recursos mejorado.

La presente invención se refiere además a un método para ejecutar un programa que incluye una pluralidad de tareas, en el que una o más tareas de la pluralidad de tareas tienen restricciones en tiempo real, en el que la pluralidad de tareas se ejecutan en una pluralidad de unidades de ejecución, comprendiendo el método las siguientes etapas: mantener un modelo de referencia en tiempo real que incluye un presupuesto de núcleo planificado B_{COREy} para cada unidad de ejecución $CORE_y$ de la pluralidad de unidades de ejecución, en el que el presupuesto de núcleo planificado B_{COREy} de una unidad de ejecución $CORE_y$ especifica un límite superior para el tiempo de ejecución que se requiere para completar todas las tareas en RT activas a tiempo con la garantía de servicio mínima, en el que el presupuesto de núcleo planificado puede determinarse como un uso máximo de una unidad de ejecución durante cada periodo de planificación para microtareas en la categoría de RT estricto o RT flexible respectivamente a lo largo de todos los periodos de planificación considerados durante la fase de calibración de un programa que incluye las tareas en tiempo real; determinar, para cada $CORE_y$ de la pluralidad de unidades de ejecución, un presupuesto de núcleo real, que es una reserva de tiempo de ejecución para tareas en tiempo real asignadas a $CORE_y$, y posibles desviaciones entre dicho presupuesto de núcleo real y el presupuesto de núcleo planificado B_{COREy}, en el que el presupuesto de núcleo real es el tiempo de ejecución en $CORE_y$ que se reserva en un determinado punto de tiempo dentro del periodo de planificación, que preferiblemente se estima, por ejemplo, basándose en los micropresupuestos μB_{xi} de todas las microtareas μT_{xi} de las tareas en tiempo real T_x asignadas a $CORE_y$ que están activas en cualquier punto en el tiempo dentro del periodo de planificación; y enviar señales de penalización de núcleo si el presupuesto de núcleo real para $CORE_y$ supera el presupuesto de núcleo planificado B_{COREy}, en el que las señales de penalización de núcleo se envían a una o más de otras unidades de ejecución $CORE_z$ para las que un presupuesto de núcleo planificado B_{COREz} supera un presupuesto de núcleo real para $CORE_z$, provocando las señales de penalización de núcleo, cuando se reciben por la una o más de otras unidades de ejecución, que la una o más de otras unidades de ejecución se desprioricen durante un periodo predefinido de tiempo de reloj de pared.

Las características proporcionadas en el presente documento tienen al menos las siguientes ventajas:

Las técnicas presentadas en el presente documento permiten que pueda controlarse el transcurso de una ejecución de una tarea, incluso en una CPU de múltiples núcleos.

Además, las técnicas presentadas en el presente documento permiten implementar aplicaciones en tiempo real estricto en arquitecturas de CPU modernas con recursos compartidos y múltiples núcleos, en las que la aplicación en tiempo real estricto requiere determinadas garantías de servicio mínimas.

Además, las técnicas presentadas en el presente documento permiten combinar la ejecución de tareas en tiempo

real y no en tiempo real sin comprometer propiedades en tiempo real debido a efectos negativos de compartir recursos en plataformas de procesador y hardware modernas. Además, puede ser posible usar sistemas operativos no en tiempo real, tales como Windows o Linux, para ejecutar aplicaciones en tiempo real.

Además, las técnicas presentadas en el presente documento permiten nuevas posibilidades para implementar políticas de seguridad, protección y aislamiento en sistemas en tiempo real.

Breve descripción de los dibujos

La figura 1 muestra un diagrama de flujo con las principales etapas implicadas cuando se ejecuta un programa que incluye una pluralidad de tareas con restricciones en tiempo real según la presente divulgación.

La figura 2 muestra un diagrama de arquitectura de un ejemplo de sistema informático con un procesador de múltiples núcleos.

La figura 3 muestra un diagrama de arquitectura de un ejemplo de sistema informático con un procesador de múltiples núcleos que incluye extensiones para soportar la ejecución de un programa que incluye una pluralidad de tareas con restricciones en tiempo real según la presente divulgación.

La figura 4 muestra un diagrama de flujo que detalla la etapa de desarrollar un modelo de referencia en tiempo real y calibración.

La figura 5 muestra un diagrama de flujo que detalla la etapa de determinar un modelo de ejecución para cada tarea en tiempo real.

La figura 6 muestra un ejemplo de programa y su gráfico de flujo de control.

La figura 7 muestra correspondencia entre porciones del gráfico de flujo de control de un programa de una tarea y un modelo de ejecución de la tarea.

La figura 8 muestra ejecuciones de una pluralidad de tareas en un procesador de múltiples núcleos, y un modelo de ejecución de una tareas de dicha pluralidad de tareas que incluye trayectos de ejecución posibles de la tarea que forman un entramado de microtareas de las tareas.

La figura 9A muestra un modelo de ejecución de una tarea, incluyendo el modelo de ejecución información sobre un trayecto crítico activo.

La figura 9B muestra una ejecución específica a lo largo del trayecto crítico activo mostrado en la figura 9A, que es una secuencia de microtareas con microvencimientos respectivos a lo largo de dicho trayecto crítico activo.

La figura 10 muestra un diagrama de flujo que detalla la etapa de determinar información de calibración y obtener un modelo de referencia en tiempo real para cada tarea T_x basándose en el modelo de ejecución de dicha tarea.

La figura 11A muestra una visualización de tiempos de ejecución muestreados de las microtareas μT_{xi} y μT_{x2}, sus micropresupuestos μB_{x1} y μB_{x2}, y WCET determinados para cada una de las microtareas.

La figura 11B muestra un diagrama de transcursos de una ejecución de la tarea T_x que tiene un vencimiento D_x.

La figura 12 muestra el diagrama de transcursos de una tarea T_x en el que las microtareas se clasifican como RT flexible y RT estricto.

La figura 13 muestra un diagrama de flujo que detalla la etapa de ejecutar una tarea en tiempo real con restricciones en tiempo real.

La figura 14 muestra una tabla que especifica los parámetros de un modelo de una tarea en tiempo real T_x.

La figura 15 muestra una tabla con información de transcurso para una ejecución de T_x con microtareas según los parámetros de la figura 14, empezando con tareas en RT flexible y terminando con tareas en RT estricto con una probabilidad calculada después de cada microtarea de no cumplir el vencimiento de RT D_x de T_x.

La figura 16 muestra un gráfico del presupuesto de activación planificado $B_{WCET,xi}$ en diferentes fases durante la ejecución de T_x correspondientes a acontecimientos E_{xi} según la ejecución y transcurso mostrados en la figura 15.

La figura 17 muestra una reserva de presupuesto de peor caso WCET(TX) tal como se realiza mediante un método convencional para ejecutar y planificar tareas en tiempo real. Además, la figura muestra el presupuesto B_x que se reservará según el método proporcionado en el presente documento. B_x tiene una componente que se refiere a la reserva de tiempo de ejecución para microtareas en RT flexible y otra componente que se refiere a la reserva de tiempo de ejecución para microtareas en RT estricto. La suma de estas reservas da como resultado el presupuesto global B_x. La figura 17 muestra además una situación después de una ejecución, específicamente qué fracción del presupuesto reservado se ha agotado realmente, es decir, se necesitó, en dicha ejecución especificada,

especificándose la ejecución en la figura 15. El resto, es decir la diferencia entre la reserva inicial y lo que se necesitó para la ejecución, se muestra como presupuesto resta, que puede estar disponible para la ejecución de otras tareas que pueden ser tareas en RT o no de RT.

La figura 18 muestra presupuestos de núcleo planificados para microtareas en RT estricto y RT flexible dentro de un periodo de planificación en múltiples núcleos de CPU y presupuestos de núcleo reales dentro del periodo de planificación.

Descripción detallada

Las técnicas dadas a conocer en el presente documento se refieren a la ejecución oportuna de programas en tiempo real, preferiblemente en arquitecturas de procesador modernas.

El método según la presente divulgación tiene dos fases principales mostradas en la figura 1. La primera fase 100, también denominada fase de calibración, se refiere al desarrollo de un modelo de referencia en tiempo real y la calibración de este modelo de una tarea T_x a la vista de la unidad de ejecución en la que debe tener lugar la ejecución de la tarea en condiciones de tiempo real y la asignación de otras tareas simultáneas. La segunda fase 200 se refiere a la ejecución real de la tarea en tiempo real con restricciones en tiempo real. En la segunda fase, se compara información del modelo de referencia en tiempo real con un transcurso real observado durante la ejecución de una tarea en tiempo real y, por consiguiente, se aumenta la prioridad de la tarea o puede liberarse reserva de recursos en cuanto a tiempo de cálculo para tareas en tiempo real según se reserva por el planificador para permitir la ejecución de otras tareas. La segunda fase puede repetirse siempre que el sistema y la información de calibración que condujeron al modelo de referencia en tiempo real permanezcan inalteradas.

Un programa en tiempo real puede ejecutarse en un sistema con un procesador de múltiples núcleos. En la figura 2 se ilustra un ejemplo de tal sistema.

La figura 2 muestra una arquitectura en capas que tiene capas para espacio de usuario, sistema operativo, OS, núcleo y hardware objetivo. El espacio de usuario incluye una o más aplicaciones, incluyendo aplicaciones con restricciones en tiempo real. Cada aplicación incluye un programa respectivo. Cuando se ejecuta una aplicación, significa que se ejecuta el programa de la aplicación. Tal ejecución tiene lugar en el hardware objetivo. Dado que los recursos del hardware objetivo pueden compartirse entre las aplicaciones, el OS gestiona recursos de hardware y más específicamente reserva unidades de ejecución para ejecutar determinadas aplicaciones según los principios de compartición de tiempo. La reserva de una unidad de ejecución para una aplicación se realiza por un planificador incluido en el núcleo de OS. Por ejemplo, el planificador determina, para cada aplicación, cuándo y en qué núcleo de una CPU de múltiples núcleos puede ejecutarse una aplicación. Un núcleo de CPU es un ejemplo de una unidad de ejecución. Otro ejemplo de una unidad de ejecución puede ser un hilo de hardware dentro de un núcleo de CPU en el caso en el que el núcleo de CPU soporta hiper-hilo. Con el fin de ilustración en esta divulgación, una unidad de ejecución puede ejecutar un programa, o más precisamente una tarea especificada por un programa, cada vez. Además, se supone que el planificador reserva tiempo de ejecución para una tarea en tiempo real siempre en la misma unidad de ejecución, es decir, una tarea en tiempo real dada siempre se ejecuta en la misma unidad de ejecución. Dicho de otro modo, se asigna una tarea a, o se “fija” en, esta unidad de ejecución. Los términos recurso de ejecución y unidad de ejecución se usan de manera intercambiable. En este caso, cada tarea de la pluralidad de tareas se asigna a una unidad de ejecución fija durante un periodo de planificación y la unidad de ejecución fija es preferiblemente un núcleo de una pluralidad de núcleos de un procesador de múltiples núcleos, y en el que se reserva tiempo de ejecución en la unidad de ejecución fija según tiempos de ejecución estimados de todas las microtareas μT_{xi} de todas las tareas en tiempo real T_x asignadas a la unidad de ejecución fija, en el que la reserva se realiza preferiblemente por un planificador que puede ser un planificador de OS.

La figura 2 muestra además elementos de un núcleo de OS moderno y hardware objetivo. Por ejemplo, un paginador incluido en el OS es responsable de gestionar la compartición de memoria disponible en el hardware entre las aplicaciones. Además, el hardware objetivo incluye en el ejemplo ilustrado una CPU de múltiples núcleos, que incluye varios núcleos y una jerarquía de memorias caché, incluyendo una memoria caché de L2 compartida y memorias caché de L1, estando estas últimas asociadas con cada núcleo. Además, el ejemplo de hardware ilustrado en la figura 2 incluye una interconexión de alto rendimiento que conecta las memorias caché con una interfaz de memoria, unidades de cálculo de uso especial, tales como una unidad de vector, e interfaces adicionales, tales como una lógica que permite la comunicación de un bus PCIe o similar. Además, el hardware objetivo incluye memoria principal. La figura 2 muestra tan sólo una arquitectura de sistema posible. Las técnicas dadas a conocer en el presente documento también pueden aplicarse a otros sistemas, por ejemplo con diferentes configuraciones de núcleos de memorias caché, etc. Específicamente, las técnicas dadas a conocer en el presente documento también pueden aplicarse a arquitecturas, que incluyen una única unidad de ejecución.

La figura 3 ilustra la arquitectura de sistema de la figura 2 con extensiones según la presente divulgación. Estas extensiones pueden implementarse en hardware o software o combinaciones de los mismos. En el ejemplo ilustrado, el hardware objetivo incluye un elemento de almacenamiento adicional, por ejemplo un registro de hardware, mostrado como “núcleo 1, núcleo 2, ...núcleo M”, que sirve para contener información sobre el avance de tareas en tiempo real que se ejecutan en cada uno de los núcleos respectivos. La información contenida en este registro

también se denomina estado parcial de transcursos reales (PATS) que comprende, para cada tarea T_x, un acontecimiento de seguimiento emitido más recientemente E_{xi} que incluye un punto en el tiempo CT_{Exi}, que es un tiempo de reloj de pared, en el que se emite el acontecimiento de seguimiento. Esta información puede comunicarse desde dicho registro de hardware hasta una unidad de planificación en tiempo real, RTSU, o la unidad "GRTM/GATM" comentada a continuación.

La figura 3 muestra una extensión de arquitectura "GRTM/GATM", que es un almacenamiento eficiente. Este almacenamiento contiene, para cada tarea en tiempo real, información sobre los tiempos de ejecución planificados de la tarea. Esta información se determina previamente a una ejecución de la tarea con restricciones en tiempo real, es decir en la etapa 100 de la figura 1, y se denomina modelo global de transcursos de referencia (GRTM). El almacenamiento eficiente contiene además información sobre el avance real de una ejecución de una tarea en tiempo real, es decir en la etapa 200 de la figura 1, cuando se ejecuta con restricciones en tiempo real. Esta información se denomina modelo global de transcursos reales (GATM), que se actualiza de manera repetida durante la ejecución de la tarea en tiempo real. Durante la ejecución de la tarea en tiempo real, se comparan el transcurso real especificado por el GATM y el transcurso de referencia expresado por el GRTM y puede aumentarse en consecuencia la prioridad de una tarea en tiempo real T_x por un planificador si se determina que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima de un límite de tolerancia. El almacenamiento eficiente para contener el modelo de referencia en tiempo real y el modelo de transcursos reales puede implementarse como unidad de hardware independiente conectada a través de una interfaz a la unidad de planificación en tiempo real y/o al procesador, o puede integrarse estrechamente con el procesador como unidad especial para soportar la ejecución en tiempo real dentro del procesador.

La figura 3 muestra una unidad de planificación en tiempo real, RTSU, que incluye diferentes unidades que están diseñadas para soportar y/o realizar el método según esta divulgación de manera eficiente y en particular la segunda fase 200 de la figura 1. El método para ejecutar un programa en tiempo real descrito en el presente documento incluye diversas etapas adicionales que no se refieren directamente al cálculo descrito por el propio programa en tiempo real sino al seguimiento del comportamiento de transcurso del programa en tiempo real, determinando si hay un retardo excesivo en la ejecución, y por consiguiente proporcionando información a un planificador.

Estas etapas adicionales se ejecutan junto a las etapas para llevar a cabo el cálculo según el programa de la tarea en tiempo real; preferiblemente estas etapas adicionales no deben afectar negativamente, es decir retardar el cálculo según el programa de la tarea en tiempo real. Por este motivo, se proporciona un soporte eficiente para llevar a cabo estas etapas adicionales, por ejemplo mediante unidades de hardware adicionales de la RTSU, con el fin de no afectar negativamente o retardar el funcionamiento de unidades conocidas con respecto al sistema sin extensiones tal como se ilustra en la figura 2. La unidad de planificación en tiempo real puede implementarse como una unidad de hardware independiente conectada a través de una interfaz al núcleo de procesador, o puede integrarse estrechamente con el procesador como unidad especial para soportar la ejecución en tiempo real dentro del procesador.

La unidad de calibración soporta la creación del modelo de referencia en tiempo real según la primera fase 100 de la figura 1 recopilando información estadística sobre el tiempo de ejecución empleado en ejecutar una porción de una tarea en tiempo real denominada microtarea. La determinación de duraciones de ejecución de porciones de un programa en tiempo real puede implicar ejecutar y determinar el perfil del programa, sin embargo no con restricciones en tiempo real sino con el fin de determinar información estadística sobre el transcurso de ejecuciones posibles de las porciones de programa respectivas. Para obtener medidas que reflejan tiempos de ejecución próximos a aquellos en condiciones de tiempo real, los tiempos de ejecución pueden determinarse en condiciones de compartición de recursos con otras tareas tal como pueden producirse cuando se ejecuta la tarea con restricciones en tiempo real.

La unidad de monitorización de acontecimientos soporta la determinación de la aparición de los denominados acontecimientos de seguimiento, o denominados simplemente acontecimientos, durante la ejecución de una tarea en tiempo real. Se emite un acontecimiento de seguimiento, acontecimiento para abreviar, mediante una determinada instrucción en el programa de una tarea.

Para emitir acontecimientos durante la ejecución de una tarea T_x, se añaden una o más instrucciones al programa de la tarea T_x, provocando las instrucciones la emisión de un acontecimiento de seguimiento E_{xi}, al final de la ejecución de una microtarea μT_{xi}, comprendiendo el acontecimiento de seguimiento un identificador único de una porción de la ejecución de la tarea T_x. De ese modo, el identificador único comprende preferiblemente un identificador de una unidad de hardware que ejecuta la tarea T_x, un identificador de la tarea T_x y un identificador del acontecimiento de seguimiento E_{xi}.

Cuando se produce un acontecimiento, es decir cuando se emite un acontecimiento durante la ejecución de una tarea T_x, en primer lugar puede escribirse información sobre el acontecimiento, que incluye el tiempo de reloj de pared en el que se emite el acontecimiento, en un almacenamiento eficiente dentro de un procesador, que puede ser, por ejemplo, un registro. Después puede obtenerse la información a partir de tal registro por la unidad de monitorización de acontecimientos, por ejemplo para realizar una actualización del modelo global de transcursos reales (GATM). Además, la unidad de monitorización de acontecimientos puede mantener, para cada tarea T_x, uno o

más de un trayecto de ejecución real y tiempos de respuesta de las microtareas μT_{xi}, que se completaron en el trayecto de ejecución real basándose en un estado real en tiempo real parcial que comprende, para cada tarea T_x, un acontecimiento de seguimiento emitido más recientemente E_{xi} que incluye un punto en el tiempo CT_{Exi} en el que se emitió el acontecimiento de seguimiento.

La unidad de monitorización de vencimiento detecta si una ejecución de una microtarea μT_{xi} termina después del microvencimiento μD_{xi}. Además, la unidad de monitorización de vencimiento puede determinar, para cada tarea, basándose en el acontecimiento de seguimiento más reciente de la tarea, que especifica el avance que ha realizado la tarea hasta un determinado punto en tiempo de reloj de pared, un tiempo de referencia obtenido a partir del modelo global de transcursos de referencia (GRTM) de la tarea, y el tiempo de reloj de pared restante real hasta el vencimiento, si tiene que aumentarse la prioridad de la tarea, lo cual se realiza por la unidad de planificación de hardware. Dicho de otro modo, la unidad de monitorización de vencimiento está diseñada para estimar una diferencia $\Delta S_{\mu Txi} = CT_{Exi-1} - \mu D_{xi-1}$ entre un tiempo de activación real CT_{Exi-1} de la microtarea μT_{xi} y un tiempo de activación planificado de la microtarea μT_{xi} y/o para detectar si una ejecución de una microtarea μT_{xi} termina después del microvencimiento μD_{xi}.

Un gestor de interferencias de núcleo puede aumentar la prioridad de la unidad de ejecución o prioridad de recursos de hardware compartidos por la unidad de ejecución en la que está asignada la tarea. De ese modo, el tiempo de referencia puede ser un tiempo de ejecución restante planificado de una tarea, que se expresa como presupuesto de activación de la tarea. Para determinar si tiene que aumentarse la prioridad de una tarea, puede determinarse una diferencia entre dicho tiempo de referencia y la duración de reloj de pared restante hasta el vencimiento de la tarea. Por ejemplo, si se determina que la duración de reloj de pared hasta el vencimiento de la tarea sólo es ligeramente superior o igual al presupuesto de activación, que es un tiempo de ejecución predicho hasta el final de la tarea según el modelo de referencia en tiempo real, entonces probablemente se aumenta la prioridad de esta tarea frente a otras tareas simultáneas en la misma unidad de ejecución. Los motivos para tal situación pueden ser los siguientes: el tiempo de ejecución real requerido por la tarea es alto y para tareas en RT flexible es posiblemente superior al estimado en el modelo de referencia en tiempo real y/o la ejecución de la tarea se retarda en tiempo de reloj de pared debido a la ejecución intermitente de otras tareas simultáneas en la misma unidad de ejecución.

La unidad de monitorización de tiempo de presupuesto gestiona tiempos de presupuesto para tareas en tiempo real en todas las unidades de ejecución, en la que el presupuesto, tal como se explicará a continuación, se divide en las categorías de presupuesto reservado para microtareas en RT estricto y microtareas en RT flexible. La unidad de monitorización de tiempo de presupuesto determina para cada unidad de ejecución y cada tarea asignada a esta unidad de ejecución, una diferencia entre el tiempo de ejecución usado por la tarea T_x hasta el acontecimiento de seguimiento más reciente E_{xi} de T_x y el tiempo de ejecución planificado reservado en la unidad de ejecución para T_x hasta el final de la microtarea μT_{xi}, lo cual se incluye en el modelo de referencia en tiempo real y que también se denomina microvencimiento μD_{xi} de la microtarea μT_{xi}. Dicho de otro modo, la unidad de monitorización de tiempo de presupuesto está diseñada para determinar, para cada tarea T_x, una desviación entre un transcurso planificado de la tarea T_x y un transcurso real de la tarea T_x, en la que el transcurso planificado de la tarea T_x antes de la ejecución de la microtarea μT_{xi} es el presupuesto de activación planificado B_{WCETxi}, y en la que el transcurso real de la tarea T_x se estima basándose en una cantidad de tiempo de CPU usado por la tarea T_x hasta el tiempo de respuesta CT_{Exi-1} y la diferencia $\Delta S_{\mu Txi}$. Si se ha reservado más tiempo que el usado por la ejecución real de T_x hasta ese punto, es decir la tarea en tiempo real T_x empleó menos tiempo de ejecución de lo previsto, entonces ese presupuesto en exceso reservado en la unidad de ejecución puede liberarse y hacerse que esté disponible para otras tareas (por ejemplo no de RT) en la misma unidad de ejecución. Por otro lado, si el tiempo de ejecución usado por una tarea hasta E_{xi} es mayor que lo que se reservó como presupuesto en esa unidad de ejecución, se toman medidas para garantizar que puede priorizarse el funcionamiento de la unidad de ejecución con respecto a otras unidades de ejecución, por ejemplo, que se prioriza un núcleo con respecto a otro núcleo en un sistema de CPU de múltiples núcleos. Tal priorización se realiza por el gestor de interferencias de núcleo unidad. Debe observarse que la situación de que se emplea más tiempo de ejecución por una tarea que el realmente planificado y reservado sólo puede surgir mientras la tarea está ejecutando microtareas en la categoría de RT flexible tal como se explicará en más detalle a continuación. Para tareas en la categoría de RT estricto, el presupuesto planificado siempre es lo suficientemente alto, dado que se calcula basándose en el WCET de las microtareas.

Para ilustración adicional, se describe la siguiente situación: la unidad de monitorización de vencimiento puede determinar para una tarea T_x que, basándose en el transcurso de reloj de pared de la tarea que se facilita como el avance real basándose en el acontecimiento de seguimiento emitido más recientemente, el tiempo de ejecución restante estimado de la tarea y la duración de reloj de pared restante hasta el vencimiento de la tarea, que la tarea tiene que priorizarse con respecto a otras tareas simultáneas. Por otro lado, para la misma tarea, la unidad de monitorización de tiempo de presupuesto puede determinar que la tarea requería menos tiempo de ejecución que el estimado y por tanto liberar presupuesto reservado en la CPU.

La unidad de planificación de hardware, unidad de planificación de HW, notifica a un planificador de sistema operativo que ajuste la prioridad de una tarea en tiempo real. La unidad de planificación de HW basa su decisión de emitir dicha notificación basándose en información obtenida a partir de la unidad de monitorización de vencimiento y/o la unidad de monitorización de tiempo de presupuesto.

Además, la RTSU incluye un gestor de interferencias de núcleo, que proporciona indicaciones para priorizar determinadas unidades de ejecución en la CPU de múltiples núcleos para el uso de recursos de procesador compartidos tales como memorias caché compartidas o ancho de banda interconectado. Las indicaciones se proporcionan basándose en información a partir de la unidad de monitorización de vencimiento o la unidad de monitorización de tiempo de presupuesto. Dado que cada tarea está sujeta, es decir fijada, a una determinada unidad de ejecución, el gestor de interferencias de núcleo da indicaciones para la priorización de uso de recursos para unidades de ejecución, que son, por ejemplo, núcleos de procesador, que ejecutan tareas para las que se pide una mayor prioridad por la unidad de planificación HW.

El método para ejecutar un programa que incluye una pluralidad de tareas con restricciones en tiempo real, etapas que están soportadas en parte por la RTSU y sus unidades tal como se describió anteriormente, se describe en más detalle basándose en la figura 4 a la figura a la figura 18 a continuación. La figura 4 muestra un diagrama de flujo que detalla la etapa 100 de desarrollar un modelo de referencia en tiempo real y calibración. En una primera etapa 110, se determina un modelo de ejecución para cada tarea en tiempo real. Un modelo de ejecución especifica una pluralidad microtareas μT_{xi}, $i \in \{1, ..., n\}$, que son una división de la tarea T_x, y un orden entre las microtareas μT_{xi} según los flujos de ejecución posibles de la tarea T_x. El modelo de ejecución de una tarea T_x se basa normalmente en, y se construye a partir de, información incluida en el programa de T_x. Por ejemplo, pueden obtenerse flujos de ejecución posibles a partir del gráfico de flujo de control del programa de T_x. Si el gráfico de flujo de control de un programa tiene ciclos, todas las partes de programa, correspondientes a bloques básicos del gráfico de flujo de control, incluidas en un ciclo pueden formar una parte de programa común, es decir un bloque más grande, de modo que, para el fin del modelo de ejecución considerado en este caso, los flujos posibles del programa de T_x pueden considerarse acíclicos. Además del flujo de control obtenido a partir del programa de la tarea T_x, una tarea puede tener relaciones de comunicación y dependencia con otras tareas, que pueden ser dependencias de control o de datos, así como dependencias de acontecimientos externos, que están relacionados, por ejemplo, con entrada y salida. Por tanto, el modelo de ejecución de una tarea incluirá y tendrá en cuenta estos y otros aspectos que influyen posiblemente en el comportamiento de transcurso de una tarea y que se conocen habitualmente en análisis de programas en tiempo real relacionados con unos modelos de ejecución de tareas. En la segunda etapa 120, se asignan tareas a unidades de ejecución. De ese modo, una asignación específica qué tareas se asignan a qué unidad de ejecución. Tal como se mencionó anteriormente, a continuación se supone que se fijan tareas a unidades de ejecución. La etapa de calibración 130 asocia básicamente información de transcurso estimado con el modelo de ejecución de una tarea. El modelo resultante de la etapa 130 se denomina modelo de referencia en tiempo real de la tarea T_x. El transcurso estimado puede obtenerse mediante análisis de programa estático o dinámico y métodos adicionales disponibles en el campo de estimación de tiempo de ejecución de peor caso. Por ejemplo, la información sobre el tiempo de ejecución de una microtarea puede ser una distribución de probabilidad de un tiempo de ejecución entre un acontecimiento de seguimiento que marca un inicio de la microtarea y un seguimiento posterior que marca un final de la microtarea. El tiempo entre acontecimiento de seguimiento posterior puede obtenerse mediante análisis de programa estático y/o dinámico, por ejemplo mediante interpretación abstracta y/o determinación de perfil. En la etapa 140, después de determinarse, basándose en los modelos de referencia en tiempo real de cada tarea asignada T_x, si la asignación de tareas en la unidad de ejecución es viable. Una asignación es viable si el uso total de la unidad de ejecución, cuando se considera que se ejecutan todas las tareas asignadas, es menor del 100% al tiempo que se garantiza con una determinada garantía de servicio mínima, que es por ejemplo del $(100\text{-}10^{-9})\%$, que todas las tareas cumplen con su restricción en tiempo real, es decir que todas las tareas completan su ejecución antes de su vencimiento asociado. A continuación, esta garantía de servicio mínima también se denomina límite de tolerancia.

La figura 5 detalla la etapa 110 de determinar un modelo de ejecución para la tarea T_x. En una primera etapa 111, se determinan todos los trayectos de ejecución posibles de la tarea T_x. Tal como se comentó anteriormente, un modelo de flujos de ejecución posibles de una tarea tiene en cuenta al menos el gráfico de flujo de control del programa de la tarea T_x, relaciones de sincronización y comunicación con otras tareas y también acontecimientos externos que influyen en el trayecto de ejecución y posiblemente el comportamiento de transcurso de la tarea T_x. El modelo de flujos de ejecución posibles de la tarea T_x es preferiblemente acíclico. En la figura 6 se muestra un ejemplo de un gráfico de flujo de control. En la técnica se conocen métodos de análisis estadístico para determinar un gráfico de flujo de control incluyendo bloques de base a partir de un código fuente de programa. Las siguientes etapas 112 se refieren a la división del programa de la tarea T_x basándose en el gráfico de flujo de control para dar microtareas μT_{xi}. Esta división es tal que cualquier ejecución posible de una tarea es una secuencia de microtareas. Porciones del programa que se repiten, tales como el cuerpo y la condición de una construcción de bucle, se incluyen preferiblemente en una única microtarea. Por ejemplo, si el gráfico de flujo de control de un programa tiene ciclos, todas las partes de programa, correspondientes a bloques básicos del gráfico de flujo de control, incluidas en un ciclo pueden formar una parte de programa común, es decir un bloque más grande, de modo que con el fin del modelo de ejecución considerado en este caso, los flujos posibles del programa de T_x pueden considerarse como acíclicos. Por tanto, en dicha secuencia de microtareas, cada microtarea se produce una vez, es decir, no se repiten las microtareas. En el ejemplo de la figura 6, los bloques básicos B3 y B4 y por tanto también las porciones correspondientes del programa mostradas en la parte izquierda de la figura 6, por ejemplo, se combinan preferiblemente cuando se forman las microtareas. Como resultado, el modelo de ejecución para una tarea es un orden parcial de microtareas, en el que las microtareas μT_{xi} $i \in \{1, ..., x\text{Última}\}$ forman un entramado con μT_{x1} como microtarea inicial de T_x y $\mu T_{x\text{Última}}$ como microtarea final de T_x. Esto se muestra en la figura 7, que ilustra la

correspondencia entre porciones de un gráfico de flujo de control de un programa de una tarea en el lado izquierdo y un modelo de ejecución de la tarea en el lado derecho. Las microtareas mostradas en el lado derecho forman un entramado. En el ejemplo, en algunos casos hay correspondencia entre un único bloque básico y una microtarea. Sin embargo, la concepción de una microtarea no está limitada de ninguna manera, lo que significa que porciones más grandes de un programa que comprenden múltiples bloques básicos, o partes de un bloque básico, también pueden formar una única microtarea, siempre que se respeten las propiedades anteriores de microtareas y el modelo de ejecución de programa.

Por tanto, el modelo de referencia en tiempo real de la tarea T_x incluye una pluralidad microtareas μT_{xi}, $i \in \{1, ..., n\}$, que son una división de la tarea T_x, y un orden entre las microtareas μT_{xi} según todos los trayectos de ejecución posibles de la tarea T_x. De ese modo, para cada microtarea μT_{xi}, $i \in \{1, ..., n\}$, se determina un micropresupuesto μB_{xi} que es más pequeño que el tiempo de ejecución de peor caso, WCET, de la microtarea μT_{xi}. El micropresupuesto μB_{xi} especifica un tiempo de ejecución para completar la ejecución de la microtarea μT_{xi} con una probabilidad inferior al 100% y por encima de un umbral de probabilidad predeterminado. Además, para cada microtarea μT_{xi}, $i \in \{1, ..., n\}$, basándose en los micropresupuestos μB_{xk}, $k \in \{1, ..., n\}$, se determina un transcurso de referencia que especifica un transcurso estimado de la microtarea μT_{xi} en cualquier ejecución posible de la tarea T_x, de tal manera que todas las continuaciones posibles de ejecuciones de la tarea T_x desde la microtarea μT_{xi} en adelante cumplen las restricciones en tiempo real de la tarea T_x con una probabilidad por encima de un límite de tolerancia, en el que las restricciones en tiempo real de la tarea T_x se cumplen con una probabilidad por encima del límite de tolerancia si la ejecución de la tarea T_x se completa antes de un vencimiento de la tarea T_x con una probabilidad inferior al 100% y por encima de una determinada garantía de servicio mínima.

De vuelta a la figura 5, la etapa 113 se refiere a insertar instrucciones en el programa de la tarea T_x. La instrucción insertada sirve para emitir acontecimientos de seguimiento cuando se ejecuta la tarea. El propósito de los acontecimientos de seguimiento es señalizar el avance de la ejecución de tarea, por ejemplo a la unidad de planificación en tiempo real comentada anteriormente. Las instrucciones insertadas se conciben de tal manera que cada acontecimiento de seguimiento tiene un identificador único para toda la CPU, que señaliza la terminación de la ejecución de una microtarea, que es una porción de la ejecución de la tarea T_x. Además del identificador único, se incluye un sello de tiempo en el acontecimiento emitido, que especifica el tiempo de reloj de pared actual en el que se emite el acontecimiento. Esto puede ser un tiempo absoluto o un tiempo con respecto al tiempo de inicio de la tarea. El identificador único comprende preferiblemente un identificador de una unidad de hardware que ejecuta la tarea T_x, un identificador de la tarea T_x y un identificador del acontecimiento de seguimiento E_{xi}, que corresponde a la microtarea al final de la cual se inserta la instrucción para emitir el acontecimiento. La concepción de microtareas y por consiguiente la inserción de instrucciones al final de una microtarea con el fin de señalizar un acontecimiento tal como se describió anteriormente puede realizarse manualmente por un arquitecto o diseñador de software que puede imponer los límites de microtareas y las instrucciones según la lógica de aplicación, o puede realizarse de manera automática mediante herramientas de desarrollo, o una combinación de colocación automatizada y manual. En la primera microtarea de una tarea, puede insertarse una instrucción para la emisión de un acontecimiento de inicio. Para ilustración adicional del concepto de microtarea y el modelo de ejecución de una tarea T_x, la figura 8 muestra un ejemplo ejecución de la tarea T3 a la izquierda, en la que una porción de la ejecución que empieza en el punto t_{30} y termina en el punto t_{31} se muestra como una línea curva pequeña; sólo para ilustración, dichos puntos son el principio de la tarea T3, y el punto en la en el que la tarea T3 lee algún mensaje m1 respectivamente. A la derecha de la figura 8, se muestra el modelo de ejecución del programa de la tarea T3, que ilustra que diferentes trayectos en el flujo de control del programa pueden conducir desde el punto t_{30} hasta el t_{31}. En la ejecución específica mostrada a la izquierda, sólo se ha tomado uno de los múltiples flujos de control posibles mostrados a la derecha.

De vuelta a la figura 5, la etapa 114 se refiere a determinar, para cada microtarea μT_{xi}, de un trayecto crítico activo, ACP, cuál es el trayecto más largo desde el final de μT_{xi} hasta el final de la tarea. Esta etapa puede soportarse mediante métodos conocidos usados en el contexto de análisis de WCET, que son, por ejemplo: extracción de un gráfico de flujo de control, que ya está disponible a partir de la etapa 111, estimación de intervalos de valores para variables y en particular para análisis y modelado de límites de bucle, memoria caché y acceso de memoria. Además, pueden aproximarse y determinarse trayectos de ejecución posibles de una tarea T_x usando un gráfico de flujo de control del programa de T_x. La figura 9A muestra un ejemplo de modelo de ejecución de tarea con un ACP empezando en la microtarea μT_{x1}, concretamente el trayecto a través de la secuencia más a la izquierda de microtareas en el modelo de ejecución de tarea ilustrado. Un ACP asociado con una microtarea μT_{xi} siempre termina al final de la última microtarea alcanzable, a la vista del orden del entramado, entre los flujos de control posibles empezando en μT_{xi}. Con el fin de determinar el ACP como el trayecto más largo, la longitud de un trayecto de flujo de control se determina como la suma de WCET de cada microtarea en el trayecto de flujo de control.

La siguiente terminología e identificadores se definen y usan en la figura 9B y en ilustraciones y fórmulas posteriores:

T_x - tarea en tiempo real X

ATT_x - tiempo de activación absoluto de T_x

J_{Tx} - fluctuación de inicio de T_x

$S_{Tx} = ATT_x + J_{Tx}$ - tiempo de inicio de T_x (el punto en el tiempo en el que empieza a ejecutarse T_x)

D_x - vencimiento para T_x

E_{xi} - i-ésimo acontecimiento de seguimiento dentro de un ACP

μT_{xi} - microtarea entre E_{xi} y E_{xi-1}

μD_{xi} - microvencimiento, μD, que es un vencimiento con respecto a S_{Tx}, que especifica hasta cuándo debe producirse el acontecimiento E_{xi}.

La figura 9B ilustra el transcurso de una ejecución de la tarea T_x a lo largo del ACP de μT_{x1}, que es el inicio de la tarea T_x. Durante la ejecución, el inicio y final de cada microtarea y su transcurso pueden obtenerse a partir de la secuencia de acontecimientos E_{xi}. El final de la tarea se marca mediante el acontecimiento $E_{x\text{Último}}$ que se produce en un punto en el tiempo mucho antes que el vencimiento de modo que la tarea T_x cumple sus restricciones en tiempo real. Las secciones verticales μT_{xi} a lo largo del eje de tiempo muestran el intervalo durante el cual se ejecuta la tarea μT_{xi}. La secuencia de acontecimientos de seguimiento y su información de transcurso asociada permite reconstruir con precisión cuál de los trayectos de flujo de control se ha ejecutado y también el transcurso, es decir la duración, de cada una de las microtareas ejecutadas a lo largo del trayecto.

De vuelta a la figura 4, la etapa 120 se refiere a asignar tareas en tiempo real a unidades de ejecución. La figura 8 muestra a la izquierda, por ejemplo, que las tareas T0, T1 y T2 se asignan al núcleo 1 de CPU, y las tareas T3 y T4 se asignan al núcleo 2 de CPU. Algunas de las tareas mostradas en la figura 8 tienen dependencias unas con respecto a otras debido a una relación de sincronización o comunicación. Por ejemplo, la tarea T1 comunica un mensaje m2 a la tarea T2 mediante un objeto compartido SO, enviándose el mensaje m2 en el punto t_{12} de la tarea T1 y recibiéndose en el punto t_{21} de la tarea T2. En este caso, la comunicación se produce entre tareas asignadas a la misma unidad de ejecución. Esta relación de comunicación conduce a una dependencia que se refleja en el modelo de flujos de programa y trayectos de ejecución posibles. Asimismo, la tarea T1 tiene una relación de comunicación con la tarea T3 que comunica un mensaje m1 a través de un canal virtual VC, en el que las tareas en comunicación se asignan a diferentes unidades de ejecución. Esta relación de comunicación conduce a una dependencia que se refleja en el modelo de flujos de programa y trayectos de ejecución posibles. Para simplificar la ilustración y la discusión relacionada con el modelo de trayectos de ejecución posibles en el presente documento, los ejemplos descritos y comentados, específicamente los modelos mostrados en la figura 6, la figura 7, la parte derecha de la figura 8, y la figura 9 se refieren al flujo de control de un programa y no muestran una situación de un modelo de trayectos de ejecución posibles que implica dependencias entre diferentes tareas (excepto por el lado izquierdo de la figura 5). Sin embargo, se entenderá que las técnicas en que la determinación de un modelo de trayectos de ejecución posibles y transcurso de una tarea T_x no sólo tiene en cuenta dependencias que surgen del flujo de control del programa que constituye la propia tarea T_x sino también dependencias e información relacionada con el transcurso debido a la comunicación y sincronización con otras tareas.

Tal como se describirá a continuación, la asignación de tareas se produce antes de la etapa de calibración 130. Esto se debe a que información sobre qué tareas se asignan en la misma unidad de ejecución, que también se denomina ubicación conjunta tareas, puede afectar al comportamiento de transcurso de cada microtarea, y por tanto también a cada tarea debido a la compartición de recursos en una unidad de ejecución y también entre diferentes unidades de ejecución dentro de la misma unidad de ejecución. Por ejemplo, múltiples tareas que se ejecutan en la misma unidad de ejecución, que puede ser, por ejemplo, un núcleo de procesador, pueden compartir la memoria caché de L1. Las tareas asignadas en diferentes unidades de ejecución pueden compartir la memoria caché de L2. El comportamiento de transcurso se refiere a un modelo estadístico que considera una distribución estadística de tiempos de ejecución observados para ejecuciones de prueba repetidas de cada microtarea en una asignación específica de tareas a unidades de ejecución determinada en la etapa 120.

Haciendo referencia a la figura 4, en la etapa 130, se parametriza el modelo de ejecución de cada tarea con información de transcurso obtenida mediante una etapa de calibración que está soportada por la unidad de calibración de la siguiente manera:

la calibración requiere modelos de ejecución de todas las tareas que se asignan en la etapa 120. Los modelos de ejecución de estas tareas se determinan en la etapa 110. Además, la calibración se controla mediante un valor umbral estadístico, por ejemplo el 75%, que se predetermina, por ejemplo, por un desarrollador de software, y que sirve para determinar un tiempo de ejecución estimado para cada microtarea basándose en una distribución de probabilidad de la duración de ejecución de la microtarea. Por consiguiente, para un valor umbral de P_{thr}, la duración de ejecución estimada de la microtarea μT_{xi} es la duración para completar la ejecución con una probabilidad inferior al 100% y por encima del valor umbral P_{thr}. Esta duración de ejecución estimada también se denomina

micropresupuesto μB_{xi} de una tarea μT_{xi}.

La etapa 130 se detalla adicionalmente en la figura 10. En una primera etapa 131 se determina un micropresupuesto μB_{xi} para cada microtarea de la tarea T_x, en la que dicho micropresupuesto es menor que el tiempo de ejecución de peor caso, WCET, de la microtarea μT_{xi}. El micropresupuesto μB_{xi} de una tarea μT_{xi} se determina preferiblemente basándose en un análisis estadístico del código de programa de μT_{xi} y/o un análisis estadístico de ejecuciones de μT_{xi}. Un análisis de este tipo está soportado por la unidad de calibración de la siguiente manera: medir de manera repetida los tiempos de ejecución de tareas y microtareas para obtener información sobre su duración de ejecución, que es, por ejemplo si una tarea usa exclusivamente la unidad de ejecución, la duración en tiempo de reloj de pared entre dos acontecimientos de seguimiento consecutivos. En otros casos, el tiempo de ejecución usado por una tarea puede obtenerse a partir del sistema operativo; se determina una distribución estadística para la duración de ejecución de cada microtarea. Por ejemplo, haciendo referencia a la figura 9B, la duración de ejecución de la microtarea μT_{x2} puede obtenerse determinando, en esta ilustración, el tiempo entre el acontecimiento de seguimiento E_{x1} y E_{x2}. Basándose en la distribución estadística de tiempos de ejecución de la microtarea μT_{xi}, puede determinarse el micropresupuesto μB_{xi} de la microtarea μT_{xi} como una duración a la que una ejecución tomada como muestra de μT_{xi} se ha completado con una probabilidad de P_{thr}, que es un umbral de probabilidad predeterminado, por ejemplo 0,75, es decir el 75%. Por tanto, el micropresupuesto μB_{xi} de una microtarea μT_{xi} se determina preferiblemente basándose en un análisis estadístico y/o interpretación abstracta del programa de μT_{xi} y/o un análisis estadístico de ejecuciones de μT_{xi}. Esta duración es más corta que el WCET de μT_{xi}. La figura 11A ilustra este aspecto de la siguiente manera. La figura muestra muestras (eje horizontal) de tiempos de ejecuciones (eje vertical) para dos microtareas diferentes μT_{x1}, para la que cada muestra se muestra como un círculo pequeño, y μT_{x2}, para la que cada muestra se muestra como un signo “+” pequeño. Para cada microtarea, líneas horizontales correspondientes muestran el WCET y el micropresupuesto. El WCET especifica la duración dentro de la cual termina cualquier ejecución de la microtarea respectiva en cualquier posible circunstancia adversa de compartición de recursos. El micropresupuesto especifica un valor inferior en el que terminan muestras con una probabilidad de P_{thr}, siendo P_{thr} una probabilidad inferior a 1,0.

Además, en una segunda etapa 132, para cada microtarea μT_{xi}, i ∈ {1, ..., n}, basándose en micropresupuestos μB_{xk}, k ∈ {1, ..., n}, se determina un transcurso de referencia que especifica un transcurso estimado de la microtarea μT_{xi} en cualquier ejecución posible de la tarea T_x hasta el final de la microtarea μT_{xi}, de tal manera que todas las continuaciones posibles de ejecuciones de la tarea T_x según el modelo de ejecución de la tarea T_x cumplen las restricciones en tiempo real de la tarea T_x con una probabilidad por encima del límite de tolerancia.

En un ejemplo de la etapa 132, el transcurso de referencia puede ser un microvencimiento. Por tanto, el transcurso de referencia de la microtarea μT_{xi} incluye un microvencimiento μD_{xi}, que especifica un tiempo de respuesta hasta el que debe terminarse una ejecución de la microtarea μT_{xi}, en el que el tiempo de respuesta es una duración con respecto a un tiempo de activación ATT_X de la tarea T_x. Basándose en los micropresupuestos determinados, se determinan microvencimientos μD_{xi} tal como se muestra en la figura 9 y la figura 11B, en las que el microvencimiento de μD_{xi} es un tiempo con respecto al inicio de la tarea T_x y es mayor que o igual a la suma de micropresupuestos en cualquier trayecto desde el inicio de la tarea hasta el final de la microtarea μT_{xi}, en el que se emite el acontecimiento E_{xi}. Dado que puede haber varios trayectos de flujo de control posibles desde el inicio de T_x hasta el final de μT_{xi}, el microvencimiento μD_{xi} se determina basándose en el trayecto de este tipo más largo, en el que la longitud puede determinarse, por ejemplo, según la suma de micropresupuestos de las microtareas μT_{xk} en un trayecto desde el inicio hasta μT_{xi}. Un ejemplo de ejecución se ilustra en la figura 11B, que también muestra una fórmula según la cual se determina μD_{xi}. Por tanto, se define la siguiente terminología e identificadores:

$\mu D_{xi} \geq \sum_{k=1}^{i} \mu B_{xi}$; por tanto, la suma en el lado derecho es un límite inferior para el microvencimiento μD_{xi}.

Por tanto, la microtarea μT_{xi} debe terminarse preferiblemente hasta que cada microtarea μT_{xk} k ∈ {1,...,i} en un trayecto crítico desde una microtarea inicial μT_{x1} hasta la microtarea μT_{xi} ha terminado la ejecución, en el que el tiempo de ejecución de cada microtarea μT_{xk} se estima preferiblemente mediante su micropresupuesto μB_{xk}, en el que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia si el transcurso real al final de la microtarea μT_{xi} supera el tiempo en el que la microtarea μT_{xi} debería haberse terminado preferiblemente, en el que el trayecto crítico hasta la microtarea μT_{xi} es un trayecto entre todos los trayectos de ejecución posibles de T_x desde una microtarea inicial hasta la microtarea μT_{xi} que tiene el tiempo de ejecución predicho más largo. Específicamente, según la fórmula anterior, el microvencimiento μD_{xi} es al menos la suma de los micropresupuestos μB_{xi} de las microtareas μT_{xk}, k ∈ {1,...,i} en el trayecto crítico hasta la microtarea μT_{xi}.

El inicio de la tarea T_x puede especificarse de ese modo como tiempo de inicio S_x, que es el tiempo en el que se emite el acontecimiento E_{x0}. Alternativamente, el tiempo de inicio de una tarea puede especificarse como el tiempo de activación de T_x, que es ATT_x. En ambos casos, se especifican los tiempos como tiempos de reloj de pared. El tiempo de activación ATT_x especifica un tiempo en el que la tarea T_x está lista para ejecutarse, que es, por ejemplo,

el tiempo en el que se cumplen dependencias posibles de la tarea con respecto a otras tareas. El tiempo de inicio S_{Tx} especifica el tiempo en el que la tarea T_x empieza realmente la ejecución, es decir el tiempo en el que está ejecutándose activamente. Por tanto, el tiempo de inicio puede ser un punto en el tiempo posterior al tiempo de activación ATT_x, en el que el retardo del inicio también se denomina fluctuación J_{Tx}, de tal manera que $S_{Tx} = ATT_x + J_{Tx}$.

Para el propósito del modelo descrito en el presente documento, puede suponerse que los vencimientos de una tarea T_x, incluyendo los microvencimientos μD_{xi}, se especifican con respecto al tiempo de activación ATT_x. Alternativa o adicionalmente, si el sistema, por ejemplo el planificador de OS, garantiza que hay un límite superior para la fluctuación J_{Tx} y que después de empezar T_x se emite el acontecimiento E_{x0} en cualquier caso antes de que el planificador elija otra tarea en la misma unidad de ejecución para su ejecución, los vencimientos de una tarea T_x, incluyendo los microvencimientos μD_{xi}, pueden especificarse con respecto al tiempo de inicio S_{Tx}.

En otro ejemplo de la etapa 132, el transcurso de referencia es un presupuesto de activación planificado $B_{WCET,xi}$ para la microtarea μT_{xi}. Un presupuesto de activación planificado especifica una duración, en el sentido del tiempo de CPU, no tiempo de reloj de pared, que es suficiente para completar la ejecución de la tarea T_x empezando desde la microtarea μT_{xi} en adelante de tal manera que sus restricciones en tiempo real se cumplen con una probabilidad por encima del límite de tolerancia. Este presupuesto de activación planificado se determina mediante la suma de duraciones de cada una de las microtareas μT_{xk}, $k \in \{i, ..., xÚltima\}$ en el trayecto crítico activo dentro de T_x empezando en la microtarea μT_{xi}. Esta determinación se describe en detalle a continuación con referencia a la figura 12 y la figura 15. En este ejemplo de la etapa 132, el transcurso de referencia de la microtarea μT_{xi} incluye un presupuesto de activación planificado B_{WCETxi} que especifica un presupuesto de tiempo de ejecución que es suficiente para completar la ejecución de la tarea T_x empezando desde la microtarea μT_{xi} de tal manera que sus restricciones en tiempo real se cumplen con una probabilidad por encima del límite de tolerancia, en el que el presupuesto de tiempo de ejecución se determina basándose en los micropresupuestos μB_{xk} de cada una de las microtareas μT_{xk}, $k \in \{i, ..., xÚltima\}$ en un trayecto crítico activo dentro de T_x empezando en la microtarea μT_{xi}; en el que el trayecto crítico activo empezando en la microtarea μT_{xi} es un trayecto entre todos los trayectos de ejecución posibles de T_x desde μT_{xi} hasta una microtarea final $\mu T_{xÚltima}$ que tiene el tiempo de ejecución predicho más largo, en el que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia si, antes de la ejecución de la microtarea μT_{xi}, el tiempo de respuesta real de la microtarea μT_{xi-1} es mayor que el microvencimiento μD_{xi-1}. En este caso, puede tener que reservarse más tiempo de ejecución que el especificado por el presupuesto de activación planificado $B_{WCET,xi}$ hasta el vencimiento de la tarea T_x.

La figura 12 muestra que las microtareas μT_{xk}, $k \in \{i, ..., xÚltima\}$ en un trayecto desde el inicio hasta el final de la tarea T_x se dividen en dos categorías, concretamente microtareas en tiempo real flexible, RT flexible, y en tiempo real estricto, RT estricto. Esta clasificación se determina de la siguiente manera: siguiendo el flujo de control de la ejecución y la secuencia de microtareas a lo largo de ese flujo que se muestra en la figura 12 desde la izquierda hacia la derecha, la primera tarea μT_{xrt} para la que la probabilidad de cumplir con su microvencimiento mD_{xrt} disminuye por debajo del límite de tolerancia deseado. El presupuesto para cada microtarea se elige de tal manera que dicha microtarea se completa con una probabilidad de P_{thr} dentro de ese presupuesto. De manera intuitiva, se cumple lo siguiente: al inicio de la ejecución de una tarea T_x, es decir quedando con un gran número de microtareas por ejecutarse, la probabilidad global de que cada microtarea a lo largo de la ejecución de T_x tarde más que su presupuesto correspondiente pasa a ser rápidamente muy pequeña cuantas más microtareas quedan por ejecutarse; por ejemplo si P_{thr}= 0,75 para cada microtarea, entonces la probabilidad de que una microtarea no termine dentro del presupuesto de ejecución proporcionado es de 0,25. La probabilidad global al comienzo de la primera microtarea de que dos microtareas seguidas no terminen con su tiempo de ejecución presupuestado pasa a ser p=0,25 $*$ 0,25 = 0,0625. Por tanto, cuantas más microtareas queden todavía por ejecutarse, menor será la probabilidad, cuando se empieza desde la primera microtarea, de que no se cumpla el vencimiento de la última microtarea. En particular, la probabilidad global, es decir combinada, para una secuencia de k microtareas es $(1-P_{thr})^k$, que pasa a ser muy pequeña a medida que k aumenta, lo cual se expresa en la siguiente fórmula:

$$Prob(CT_{ExÚltimo} > D_X) = (1 - P_{thr})^{n.º\ acontecimientos}$$

donde

CT_{Exi} designa un punto real en el tiempo con respecto al inicio de la tarea T_x en el que se señaliza E_{xi}.

n.º acontecimientos designa el número de acontecimientos en una ejecución.

A la inversa, si quedan pocas microtareas, la probabilidad de no cumplir el vencimiento de la tarea puede ser mayor e incluso puede pasar a ser mayor que el riesgo tolerado de no cumplir el vencimiento, que tiene correspondencia con respecto a una garantía de servicio mínima. Estas probabilidades se muestran en la figura 15 en la última columna de la hoja de cálculo “probabilidad residual de no cumplir el vencimiento”. En un determinado punto durante la ejecución, puede no haber un número suficiente de microtareas restantes como para “compensar” el hecho de que el presupuesto reservado para cada una de las microtareas restantes simplemente garantiza con una probabilidad de P_{thr} que la microtarea correspondiente se complete dentro de su presupuesto, estando la

probabilidad P_{thr} significativamente por debajo de 1,0, por ejemplo 0,75 en la figura 14 y 15, y también por debajo de la garantía de servicio mínima determinada, que es, por ejemplo, de (100-10^{-9})%. En la figura 15, este punto se alcanza después del acontecimiento 43, en el que la probabilidad de no cumplir el vencimiento (última columna) aumenta por encima de 10^{-11}. Por tanto, el método dado a conocer en el presente documento añade a las últimas microtareas de μT_{xrt} a $\mu T_{x\text{último}}$, que se muestran en la figura 15 en la zona "RT estricto", a lo largo de un trayecto de ejecución de la tarea T_x, que en este caso es el trayecto crítico, un tiempo de ejecución adicional, denominado tiempo de tampón, a los presupuestos de $B_{WCET,xrt}$ a $B_{WCET,x\text{último}}$ de las microtareas respectivas. Este presupuesto de tiempo adicional es la diferencia entre el micropresupuesto μB para las microtareas respectivas tal como se comentó anteriormente y el WCET de las microtareas respectivas. Este tiempo de tampón se designa $BT(\mu T_{xk})$ para la microtarea μT_{xk}.

Un tiempo de ejecución de una microtarea en tiempo real flexible μT_{xi} se estima mediante su micropresupuesto μB_{xi}. un tiempo de ejecución de una microtarea en tiempo real estricto μT_{xi} se estima mediante su micropresupuesto μB_{xi} más un tiempo de tampón $BT(\mu T_{xi})$, siendo el tiempo de tampón un tiempo adicional para garantizar que μT_{xi} termina con una certeza del 100% dentro del tiempo estimado. Para determinar un presupuesto de activación planificado B_{WCETxi}, el presupuesto de tiempo de ejecución se basa en una suma de los tiempos de ejecución estimados de la microtarea en tiempo real flexible y en tiempo real estricto μT_{xi}. Esto también se expresa mediante la siguiente fórmula.

Se definen y se usan la siguiente terminología e identificadores en la figura 12 y a continuación:

$WCET(\mu T_{xi})$ es el WCET determinado para μT_{xi}.

$BT(\mu T_{xi}) = WCET(\mu T_{xi}) - \mu B_{xi}$ (tiempo de tampón para μT_{xi}) es un presupuesto adicional que es necesario además de μB_{xi} para proporcionar una garantía del 100% de que puede cumplirse el microvencimiento μD_{xi}.

E_{xrt} designa el primer acontecimiento con un vencimiento en tiempo real estricto μD. A partir de este acontecimiento, la probabilidad de que las microtareas restantes en la ejecución de T_x cumplan sus microvencimientos correspondientes pasa a ser inferior a la garantía de servicio mínima requerida, dado que los microvencimientos se determinan basándose en los micropresupuestos.

B_{WCETx} designa el presupuesto de CPU requerido de la tarea x, para cumplir su vencimiento D_x dentro de la garantía de servicio mínima requerida.

B_{WCETxi} designa el presupuesto de CPU de peor caso que es suficiente para completar la ejecución de las microtareas restantes antes del vencimiento D_x con una probabilidad mayor que o igual a la garantía de servicio mínima requerida.

n.ºCPacontecimientos$_x$ designa el número de acontecimientos que están incluidos eso él más largo un trayecto crítico de T_x.

n.ºACPacontecimientos$_x$ designa el número de acontecimientos restantes en el trayecto crítico de T_x empezando desde el acontecimiento E_{xi} hasta el acontecimiento $E_{x\text{Último}}$.

Por tanto, el presupuesto de CPU requerido de la tarea T_x se calcula de la siguiente manera:

$$B_{WCETx} = \sum_{k=0}^{n.ºCPacontecimientos_x} \mu B_{xk} + \sum_{k=rt}^{n.ºCPacontecimientos_x} (BT(\mu T_{xk}))$$

El presupuesto de CPU de peor caso para una microtarea se calcula de la siguiente manera:

$$B_{WCETx} = \sum_{k=i}^{n.ºACPacontecimientos_x} \mu B_{xk} + \sum_{si\ (i\geq rt)k=i}^{n.ºACPacontecimientos_x} (BT(\mu T_{xk}))$$

$$= \sum_{k=i}^{\text{mientras } (E_{xk}\neq E_{xrt})} \mu B_{xk} + \sum_{si\ (i\geq rt)k=i}^{n.ºACPacontecimientos_x} \mu B_{xk} + \sum_{si\ (i\geq rt)k=i}^{n.ºACPacontecimientos_x} (BT(\mu T_{xk}))$$

$$= \sum_{k=i}^{\text{mientras } (E_{xk}\neq E_{xrt})} \mu B_{xk} + \sum_{si\ (i\geq rt)k=i}^{n.ºACPacontecimientos_x} WCET(\mu T_{xi})$$

$$= SoftB_{Txi} + HardB_{Txi}$$

Por tanto, para una microtarea μT_{xi}, el presupuesto de CPU de peor caso, B_{WCETxi}, tiene dos componentes, concretamente una componente en tiempo real flexible y una en tiempo real estricto según lo anterior.

Por consiguiente, para una tarea T_x, el presupuesto de CPU requerido también puede dividirse en componentes en tiempo real estricto y flexible correspondientes al cálculo anterior para cada microtarea, de modo que:

$$B_{WCETx} = SoftB_{WCETx} + HardB_{WCETx}$$

También pueden combinarse ambos ejemplos de la etapa 132, es decir, la información de transcurso de referencia incluye microvencimientos según el primer ejemplo y también presupuestos y tiempos de tampón según el segundo ejemplo. La información de transcurso de referencia determinada en la etapa 132 se asocia con la microtarea μT_{xi} respectiva en el modelo de ejecución de la tarea T_x para obtener el modelo de referencia en tiempo real para la tarea T_x.

En resumen, la etapa 130 logra que, para cada microtarea μT_{xi}, i ∈ {1, ...,n}, se determine un micropresupuesto μB_{xi} que es más pequeño que el tiempo de ejecución de peor caso, WCET, de la microtarea μT_{xi}. Finalmente, debe renovarse la calibración, es decir, debe repetirse la etapa 130, siempre que se cambie el programa de una tarea. La etapa de calibración 130 también puede repetirse cuando se cambia la asignación de tareas determinada en la etapa 120, dado que tal cambio puede tener un impacto sobre la compartición de recursos y posiblemente conducir a cambios en los tiempos de ejecución estadísticos que son una base para determinar micropresupuestos, microvencimientos, etc.

Se hace referencia a la etapa 140 en la figura 4. Para cada unidad de ejecución y dada la asignación determinada en la etapa 120 y los modelos de referencia en tiempo real determinados en la etapa 130, puede determinarse un uso total de la unidad de ejecución basándose en el presupuesto de CPU de peor caso calculado para cada tarea que se divide en componentes de presupuesto para microtareas en tiempo real flexible y en tiempo real estricto. Por tanto, para cada unidad de ejecución, por ejemplo un núcleo de CPU, la reserva total de tiempo de ejecución para microtareas en tiempo real estricto y flexible en una unidad de ejecución $CORE_y$ puede determinarse tal como se describe en detalle a continuación.

La selección de tareas para asignar en una unidad de ejecución $CORE_y$ específica puede determinarse haciendo variar las tareas en tiempo real asignadas y maximizando el tiempo reservado para tareas en tiempo real en $CORE_y$, es decir para determinar $MAX(B_{COREy})$ con la siguiente restricción, que debe cumplirse para una asignación permisible:

En cualquier momento, $MAX(B_{COREy})$ debe ser inferior al uso máximo de la unidad de ejecución.

Por tanto, suponiendo que el presupuesto B_{WCETx} de la tarea T_x es el presupuesto de activación planificado B_{WCETx1} de la microtarea inicial μT_{x1} de la tarea T_x, puede asignarse una pluralidad de tareas en la misma unidad de ejecución siempre que se cumplan las siguientes restricciones: la suma de tiempos de ejecución estimados para las microtareas en RT estricto de cada tarea de la pluralidad de tareas no supera una determinada primera porción de un uso máximo de la unidad de ejecución durante el periodo de planificación; y la suma de los presupuestos de las tareas en RT asignadas a la misma unidad de ejecución no supera una determinada segunda porción del uso máximo de la unidad de ejecución durante el periodo de planificación. En la figura 18, por ejemplo, la primera porción es el 50% y la segunda porción es el 75%.

En un ejemplo, el presupuesto de núcleo planificado B_{COREy} incluye un presupuesto de núcleo planificado para microtareas en tiempo real estricto $HardB_{COREy}$ en $CORE_y$ y un presupuesto de núcleo planificado para microtareas en tiempo real flexible $SoftB_{COREy}$ en $CORE_y$, en el que $HardB_{COREy}$ es la suma de presupuestos planificados de la microtarea μT_{xi} en la categoría de tiempo real estricto para todas las tareas en tiempo real activas T_x asignadas a $CORE_y$, y en el que $SoftB_{COREy}$ es la suma de presupuestos planificados de la microtarea μT_{xi} en la categoría de tiempo real flexible para todas las tareas en tiempo real activas T_x asignadas a $CORE_y$.

Una tarea en tiempo real T_x está activa si se cumplen las dos condiciones siguientes: en primer lugar, ya se ha empezado la ejecución de la tarea T_x, es decir, se ha emitido el acontecimiento de seguimiento E_{x0}, y, en segundo lugar, el último acontecimiento emitido no es $E_{xÚltimo}$, es decir, aún no se ha terminado la tarea T_x. Algunas de las tareas asignadas a una unidad de ejecución pueden no estar activas en ese sentido, por ejemplo, si aún no se han cumplido sus dependencias de sincronización o comunicación con otras tareas. Por ejemplo, en la figura 8, en la parte izquierda de la figura, la tarea T2 no puede ejecutarse más allá del punto t_{21} a menos que esté disponible el mensaje m2 para leerse a partir del objeto compartido. Si la tarea de envío T1 no ha avanzado hasta el punto de proporcionar el mensaje m2 a través del objeto compartido, entonces la tarea T2 puede no estar activa temporalmente, en concreto hasta que esté disponible el mensaje m2. En algunos ejemplos, las tareas en tiempo real activas pueden ser las tareas en tiempo real asignadas.

Los presupuestos de núcleo planificados proporcionan un límite superior para el tiempo de ejecución que se requiere para completar todas las tareas en RT activas a tiempo con la garantía de servicio mínima.

A continuación se describe el método según la segunda fase 200 de la figura 1.

La figura 13 muestra un diagrama de flujo que detalla la etapa 200 de ejecución de una tarea en tiempo real T_x con

restricciones en tiempo real.

Los conceptos y el método de la etapa 200 se ilustran en un ejemplo de tarea en tiempo real y la ejecución que se especifica en la figura 14 a la figura 18.

La figura 14 muestra una tabla que especifica los parámetros de un modelo de una tarea en tiempo real T_x que tiene 60 microtareas en su trayecto crítico activo desde el inicio de T_x, 17 de las cuales se encuentran en la categoría de RT estricto. Las microtareas son uniformes, teniendo cada una un micropresupuesto para el tiempo de ejecución de 8 unidades, WCET de 32 unidades de tiempo y, por tanto, un tiempo de tampón de 24 unidades de tiempo, en el que el micropresupuesto se ha determinado basándose en una probabilidad umbral de 0,75.

La figura 15 muestra una tabla con información de transcurso para una ejecución de T_x con microtareas según los parámetros de la figura 14 empezando con tareas en RT flexible y terminando con tareas en RT estricto. La ejecución mostrada corresponde a la ejecución a lo largo del trayecto crítico activo desde el inicio de T_x.

La figura 16 muestra un gráfico del presupuesto de activación planificado $B_{WCET,xi}$ en diferentes puntos en el tiempo i correspondientes a acontecimientos E_{xi}. La figura 16 muestra una disminución en el presupuesto de activación de cada microtarea empezando en μT_{x1} hasta μT_{x60}. Según lo anterior, se reservan presupuestos más grandes para tareas en RT estricto, que son de μT_{x44} a μT_{x60}, dado que el presupuesto de estas tareas también incluye los tiempos de tampón. El diagrama corresponde a los valores en la cuarta columna en la tabla de la figura 15.

La figura 17 muestra una reserva de presupuesto de peor caso WCET(T_x) tal como se realiza mediante un método convencional para ejecutar y planificar tareas en tiempo real. Además, la figura 17 muestra el presupuesto B_x que se reservará según el método proporcionado en el presente documento. B_x tiene una componente que se refiere a la reserva para microtareas en RT flexible y otra componente que se refiere a la reserva para microtareas en RT estricto. La suma de estas reservas da como resultado el presupuesto B_x global. La figura 17 muestra además qué fracción del presupuesto reservado se ha gastado realmente, es decir, se necesitó, en la ejecución de la figura 15, específicamente el transcurso especificado en la segunda columna de la tabla en la figura 15. El resto, es decir la diferencia entre la reserva inicial y lo que se necesitó para la ejecución, se muestra como presupuesto resta, que está disponible para la ejecución de otras tareas.

En la etapa 201, se inicializa la unidad de planificación en tiempo real, en la que se proporciona el modelo de referencia en tiempo real de la tarea T_x al hardware objetivo y se almacena en la unidad de almacenamiento de "GRTM/GATM" que permite un acceso eficiente al modelo de referencia en tiempo real de la tarea T_x por la unidad de planificación en tiempo real. En la etapa 201, la ejecución de la tarea T_x empieza en la primera microtarea μT_{x1}. Al comienzo de μT_{x1} se emite un acontecimiento de seguimiento de inicio E_{x0}. La emisión de un acontecimiento conduce a una actualización del estado parcial de transcursos reales (PATS) que comprende, para cada tarea T_x, un acontecimiento de seguimiento emitido más recientemente E_{xi} que incluye un punto en el tiempo CT_{Exi} en el que se emite el acontecimiento de seguimiento. Por tanto, después de la ejecución de la microtarea μT_{xi}, se determina un transcurso real, que puede ser un tiempo de respuesta desde la activación de la tarea T_x hasta el final de la microtarea μT_{xi}.

La unidad de monitorización de acontecimientos es responsable de crear y actualizar el modelo global de transcursos reales (GATM) de toda la CPU. Específicamente, la unidad de monitorización de acontecimientos obtiene en la mayor parte acontecimientos registrados en el estado parcial de transcursos reales (PATS), preferiblemente en intervalos regulares. Específicamente, para cada tarea T_x y cada unidad de ejecución $CORE_y$, se mantienen actualizados los valores de presupuesto B_x y B_{COREy} basándose en el avance observado mediante los acontecimientos emitidos por las tareas y obtenidos por la unidad de monitorización de acontecimientos. En un ejemplo, el tiempo de ejecución reservado por un planificador en una unidad de ejecución es la suma de $HardB_{WCETk}$ y $SoftB_{WCETk}$ de todas las tareas activas asignadas a esta unidad de ejecución. De manera correspondiente, el tiempo de ejecución reservado B_{COREy} para la unidad de ejecución $CORE_y$ se divide en una componente para microtareas en RT estricto y en RT flexible. A medida que se ejecutan y se completan las microtareas, la unidad de monitorización de acontecimientos determina los tiempos de ejecución empleados para cada microtarea. Si el tiempo de ejecución real de una microtarea ha sido menor que el presupuesto reservado para la microtarea, puede liberarse el presupuesto en exceso, lo que significa que el planificador libera la reserva y por tanto puede hacer que haya tiempo de ejecución adicional disponible para otras tareas. La componente de RT estricto y RT flexible de B_{COREy}, dependiendo de si la microtarea se encuentra en la categoría de RT estricto o RT flexible, se reduce de manera correspondiente basándose en el tiempo de ejecución determinado de la microtarea usada y/o el tiempo liberado.

Por tanto, en una realización, si una diferencia entre un tiempo de activación real de la microtarea μT_{xi} y un tiempo de activación planificado de la microtarea μT_{xi} es negativa, se libera una porción del tiempo de ejecución dentro del periodo de planificación reservada en la unidad de ejecución para la ejecución de una tarea T_x con restricciones en tiempo real y por tanto está disponible para la ejecución de otras tareas, en la que el tiempo de activación planificado de la microtarea μT_{xi} es el microvencimiento μD_{xi-1} de la microtarea anterior μT_{xi-1}, y en la que la cantidad de tiempo liberado es inferior o igual a la diferencia entre el tiempo real restante hasta el vencimiento de T_x y el presupuesto de activación planificado B_{WCETxi}.

Preferiblemente, el modelo global de transcursos reales (GATM) se gestiona y se actualiza por la unidad de monitorización de acontecimientos. El GATM incluye, para cada tarea T_x y cada unidad de ejecución $CORE_y$, valores actuales de B_x y B_{COREy}, que se mantienen actualizados, de ahí el modelo “global” de transcursos reales.

Además, la unidad de monitorización de acontecimientos “acumula” los valores de todos los estados parciales de transcursos reales (PATS) que se observan desde que se inicia el sistema. El valor de B_x especifica inicialmente, es decir antes de que empiece la ejecución de T_x, B_{WCETx} tal como se determina durante la fase I en la etapa 100.

En el ejemplo de tarea y ejecución que se considera en este caso para ilustración, el valor de presupuesto reservado inicial se muestra en la figura 17 como B_x en la columna “reservado”. Tal como se comentó anteriormente, el presupuesto total B_x tiene una componente que se refiere al presupuesto de RT estricto y al de RT flexible. Por ejemplo, el valor 368 se muestra como “presupuesto de RT flexible” reservado como valor inicial en la primera línea, quinta columna de la figura 15 con el título “B_Softxi”. Durante el transcurso de la ejecución, la unidad de monitorización de acontecimientos reduce posteriormente este presupuesto restando el tiempo de ejecución real empleado por una microtarea y una cantidad de corrección $\Delta S_{\mu Txi}$ que especifica si la microtarea se empezó temprano o tarde.

$\Delta S_{\mu Txi} = CT_{Exi-1} - \mu D_{xi-1}$ designa una diferencia, es decir una cantidad de corrección, entre un tiempo de activación planificado de la microtarea μT_{xi} según el modelo de referencia en tiempo real, que es el microvencimiento μD_{xi-1}, y el tiempo de activación real de μT_{xi}, que es CT_{Exi-1}. El valor es negativo si μT_{xi} empieza antes que lo planificado, de lo contrario es cero o positivo.

Además, un presupuesto de activación real de la microtarea μT_{xi} es el presupuesto de activación planificado $B_{WCET,xi}$ preferiblemente corregido mediante $\Delta S_{\mu Txi}$. Esto también se muestra en la figura 15, en la que la ejecución avanza descendiendo a lo largo de las filas desde la parte superior hasta la parte inferior de la tabla y en la que los valores en la quinta columna se reducen gradualmente mediante el transcurso real según el cual avanza la tarea. La diferencia entre filas posteriores se determina mediante el tiempo de ejecución de una microtarea y la cantidad de corrección en la tercera columna.

En cuanto la unidad de monitorización de acontecimientos ha actualizado el GATM, se informa a la unidad de monitorización de vencimiento, la unidad de monitorización de tiempo de presupuesto y el gestor de interferencias de núcleo. En respuesta, estas unidades pueden hacer, por ejemplo, que aumente la prioridad de una tarea tal como se mencionó anteriormente y tal como se comentará en más detalle a continuación.

En la etapa 202, se ejecuta una microtarea, de modo que se emite el acontecimiento de seguimiento respectivo al final de la microtarea.

En la etapa 203, se determina un transcurso real, incluyendo el tiempo de ejecución usado de la tarea y respectivamente su microtarea, y una duración restante hasta el vencimiento de la tarea, después de la ejecución de la microtarea μT_{xi}. El transcurso real también puede incluir un tiempo de respuesta desde la activación y/o el inicio de la tarea T_x hasta el final de la microtarea μT_{xi}. El acontecimiento E_{xi} incluye un sello de tiempo actual que especifica el tiempo en el que se alcanza el final de μT_{xi}, se incluye información sobre el inicio de la tarea T_x en el GATM basándose en los estados parciales en tiempo real acumulados.

En la etapa 204, se compara un transcurso real con un transcurso de referencia. En una realización, en la etapa 204, se compara el transcurso real obtenido en la etapa 203 y que condujo a una actualización del GATM, con el transcurso de referencia incluido en el modelo global de transcursos de referencia GRTM. Esta comparación se realiza por la unidad de monitorización de vencimiento que tiene preferiblemente un acceso muy eficiente a la unidad de almacenamiento de GRTM/GATM. Específicamente, preferiblemente el funcionamiento de la unidad de monitorización de vencimiento y el acceso al almacenamiento de GRTM/GATM no deben afectar negativamente o retardar el funcionamiento normal del procesador, por ejemplo compartiendo recursos con unidades de ejecución que ejecutan tareas en tiempo real. Para realizar la comparación, la unidad de monitorización de vencimiento obtiene a partir del GRTM un valor de transcurso de referencia, que puede ser según lo anterior, un microvencimiento, o un valor de presupuesto de activación de una microtarea o combinaciones de los mismos. El tiempo de referencia puede ser, por ejemplo, un tiempo de ejecución restante planificado de una tarea empezando en la microtarea μT_{xi}, que se expresa como presupuesto de activación B_{WCETxi}. Para determinar si tiene que aumentarse la prioridad de una tarea, puede determinarse una diferencia entre dicho tiempo de referencia y la duración de reloj de pared restante D_x - CT hasta el vencimiento D_x de la tarea, en la que CT es el tiempo de reloj de pared actual, por ejemplo especificado con respecto al inicio de la tarea. Por ejemplo, si se determina que la duración de reloj de pared hasta el vencimiento de la tarea sólo es ligeramente superior o igual al presupuesto de activación, que es un tiempo de ejecución predicho hasta el final de la tarea según el modelo de referencia en tiempo real, entonces es probable que se aumente la prioridad de esta tarea con respecto a otras tareas simultáneas en la misma unidad de ejecución tal como se comenta a continuación.

En la etapa 205, basándose en este valor de transcurso de referencia y en desviaciones entre los valores planificados y los valores reales obtenidos a partir del GATM y determinados mediante la comparación en la etapa 204, la unidad de monitorización de vencimiento informa a la unidad de planificación de HW, por ejemplo para

aumentar la prioridad de una tarea. Específicamente, basándose en la comparación, si se determina que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia, se aumenta la prioridad de la tarea T_x. En un ejemplo, la unidad de planificación de HW está diseñada para generar un valor de control en tiempo real para una tarea T_x basándose en una desviación del transcurso planificado de T_x con respecto al transcurso real de la tarea T_x en la unidad de ejecución, específicamente el transcurso real de una microtarea μT_{xi} más recientemente terminada de T_x. La unidad de planificación de HW está diseñada además para señalizar el valor de control en tiempo real a un planificador de OS.

En otro ejemplo, la unidad de monitorización de tiempo de presupuesto está diseñada además para determinar, para cada $CORE_y$ de la una o más unidades de ejecución, un presupuesto de núcleo real, que es una reserva de tiempo de ejecución para todas las tareas en tiempo real asignadas a $CORE_y$, y posibles desviaciones entre dicho presupuesto de núcleo real y un presupuesto de núcleo planificado B_{COREy}, en la que el presupuesto de núcleo real es el tiempo de ejecución en $CORE_y$ que está reservado en un determinado punto de tiempo dentro del periodo de planificación.

Con el fin de determinar el presupuesto de núcleo real, un presupuesto de activación de una tarea en tiempo real es un transcurso estimado de la tarea en tiempo real de tal manera que todas las continuaciones posibles de ejecuciones de la tarea en tiempo real cumplen las restricciones en tiempo real de la tarea en tiempo real con una probabilidad por encima de un límite de tolerancia durante un periodo de planificación. De ese modo, el presupuesto de núcleo real se determina como el tiempo de ejecución en $CORE_y$ que está reservado en un determinado punto en el tiempo dentro del periodo de planificación, que preferiblemente se estima, por ejemplo, basándose en los micropresupuestos μB_{xi} de todas las microtareas μT_{xi} de todas las tareas en tiempo real activas T_x asignadas a $CORE_y$ en cualquier punto en el tiempo dentro del periodo de planificación.

Con el fin de determinar el presupuesto de núcleo planificado B_{COREy}, el presupuesto planificado de tareas en tiempo real puede ser el tiempo de ejecución asignado por un planificador de tareas, por ejemplo en el OS, para las tareas en tiempo real activas en $CORE_y$ durante un periodo de planificación. De ese modo, el presupuesto de núcleo planificado B_{COREy} de una unidad de ejecución $CORE_y$ especifica un límite superior para el tiempo de ejecución que se requiere para completar todas las tareas en tiempo real activas a tiempo con la garantía de servicio mínima, en el que el presupuesto de núcleo planificado puede determinarse como un uso máximo de una unidad de ejecución durante cada periodo de planificación para microtareas en la categoría de RT estricto o de RT flexible respectivamente a lo largo de todos los periodos de planificación considerados durante la fase de calibración de un programa que incluye las tareas en tiempo real. En algunos ejemplos, el presupuesto de núcleo planificado B_{COREy} de una unidad de ejecución $CORE_y$ puede estimarse como un uso máximo de una unidad de ejecución durante cada periodo de planificación para microtareas en la categoría de RT estricto o de RT flexible respectivamente a lo largo de todos los periodos de planificación considerados durante la fase de calibración de un programa que incluye las tareas en tiempo real. Alternativamente, B_{COREy} también puede determinarse usando métodos convencionales de análisis de capacidad de planificación basándose en micropresupuestos. En otro ejemplo, el presupuesto de núcleo planificado puede ser el tiempo de CPU máximo reservado dentro de un periodo de planificación, considerando cualquiera de los periodos de planificación durante una ejecución de calibración. En algún ejemplo, el presupuesto de núcleo planificado B_{COREy} puede estimarse basándose en los micropresupuestos μB_{xi} de todas las microtareas μT_{xi} de todas las tareas en tiempo real activas T_x asignadas a $CORE_y$ durante un periodo de planificación.

Además, en otro ejemplo, también en la etapa 205, el gestor de interferencias de núcleo, por ejemplo, puede ajustar prioridades internas de procesador de uso de recursos compartidos tales como memoria caché de L2 o interconectar o puede suspender temporalmente el funcionamiento de unidades de ejecución específicas, para las que todas las tareas asignadas han avanzado suficientemente. Por ejemplo, el gestor de interferencias de núcleo puede enviar señales de penalización de núcleo a unidades de ejecución distintas de $CORE_y$ si el uso real de tiempo de ejecución para tareas en tiempo real en $CORE_y$ supera el presupuesto planificado B_{COREy}. Dicho de otro modo, el gestor de interferencias de núcleo puede enviar señales de penalización de núcleo si el presupuesto de núcleo real para $CORE_y$ supera el presupuesto de núcleo planificado B_{COREy}, en el que las señales de penalización de núcleo se envían a una o más de otras unidades de ejecución $CORE_z$ para las que un presupuesto de núcleo planificado B_{COREz} supera un presupuesto de núcleo real para $CORE_z$, provocando las señales de penalización de núcleo, cuando se reciben por la una o más de otras unidades de ejecución, que la una o más de otras unidades de ejecución se desprioricen durante un periodo predefinido de tiempo de reloj de pared. De ese modo, la desprioritización puede incluir detener la unidad de ejecución durante un periodo predefinido de tiempo de reloj de pared. En algunos ejemplos, la señal de penalización de núcleo dirigida a una unidad de ejecución de la una o más de otras unidades de ejecución sólo se envía en un punto en el tiempo que no es crítico para la unidad de ejecución objetivo, en la que un punto en tiempo de reloj de pared en una unidad de ejecución a la que se dirige la señal de penalización de núcleo no es crítico si dicha unidad de ejecución no está reservada para la ejecución de microtareas en tiempo real estricto a partir de dicho punto en tiempo de reloj de pared durante el periodo predefinido de tiempo de reloj de pared.

Las señales de penalización de núcleo provocan, cuando se reciben por las otras unidades de ejecución, que las otras unidades de ejecución se desprioricen, preferiblemente durante un periodo predefinido de tiempo de reloj de pared. El propósito de tales penalizaciones de núcleo es evitar retardo de núcleo o inaniciones para recursos de chip

compartidos (por ejemplo, ancho de banda interconectado). Como resultado de una penalización de núcleo, se detiene la ejecución de software en el núcleo afectado preferiblemente durante el periodo de tiempo predefinido. De ese modo, sólo se envía una señal de penalización de núcleo a otras unidades de ejecución $CORE_z$ en puntos en el tiempo que no son críticos para $CORE_z$. Un punto de tiempo no crítico es un valor de tiempo de reloj mundial, en el que, para la duración de la penalización, el comportamiento en tiempo real en ese núcleo no se ve afectado, por ejemplo cuando no está ejecutándose, planificada o activada ninguna microtarea en tiempo real estricto, durante el periodo de penalización. En algunos casos, esto también significa que un presupuesto planificado B_{COREz} supera un uso real de tiempo de ejecución para tareas en tiempo real en $CORE_z$.

Además, en otro ejemplo, también en la etapa 205, una porción del tiempo de ejecución dentro del periodo de planificación reservada en la unidad de ejecución para la ejecución de una tarea T_x con restricciones en tiempo real puede liberarse según lo anterior.

Cuando se informa al componente de planificación de hardware para aumentar la prioridad de tarea, puede decidirse cuánto tiene que aumentarse la prioridad. El objetivo de la etapa 205 y el método en su conjunto es provocar ajustes de la planificación de una manera que haga que el comportamiento de transcurso real de las tareas sea lo más parecido posible al comportamiento planificado especificado en el modelo global de transcursos de referencia (GRTM). Por tanto, la unidad de planificación de hardware decide basándose en una diferencia, que se determina según lo anterior, qué tareas van a ejecutarse con prioridad superior. Para esta decisión, puede usarse un método de planificación con prioridades dinámicas, tales como, por ejemplo, primero el vencimiento más temprano (EDF) o primero la menor holgura (LLF). Los valores de control generados por la unidad de planificación de hardware se comunican, por ejemplo, al planificador de sistema operativo.

El planificador garantiza que, para cada tarea en tiempo real activa T_x, se cumplen las siguientes condiciones en cualquier momento:

- $D_x - CT - B_{WCETxi} > 0$ y
- cuanto menor es la diferencia ($D_x - CT - B_{WCETxi}$), mayor es la prioridad.

Finalmente, haciendo referencia al método ilustrado en la figura 13, en la etapa 206, si quedan microtareas adicionales en la tarea actual por ejecutarse, entonces el método realiza una iteración de vuelta a la etapa 202 para la siguiente microtarea. De lo contrario, la ejecución de la tarea T_x termina en la etapa 207. En otra realización, las etapas 203, 204, 205 pueden no repetirse después de la ejecución de cada microtarea sino en intervalos predeterminados hasta que se termina la ejecución de la pluralidad de tareas, en la que los intervalos predeterminados son preferiblemente regulares.

La figura 18 muestra presupuestos de núcleo planificados para microtareas en RT estricto y en RT flexible en múltiples núcleos de CPU y presupuestos reales de tiempos de ejecución de las tareas activas, en la que porciones del presupuesto de RT estricto y RT flexible planificados pueden liberarse para tareas no en RT. La figura 18 muestra por ejemplo que los presupuestos grandes para tareas en RT estricto normalmente no se usan completamente debido al cálculo muy conservativo de los tiempos de tampón basándose en estimación de WCET. Por tanto, los presupuestos de RT estricto reales usados siempre serán inferiores o iguales, y normalmente inferiores, al presupuesto inicialmente reservado, que es el presupuesto de núcleo planificado. Los presupuestos de núcleo planificados se determinan, por ejemplo, durante una fase de calibración o usando análisis de capacidad de planificación convencional. La calibración considera de ese modo una duración a lo largo de toda la ejecución de un programa que incluye la una o más tareas en tiempo real, en la que esta duración se divide en uno o más periodos de planificación. El presupuesto de núcleo planificado para microtareas en RT estricto especifica una estimación segura de uso de CPU que es suficiente en cualquier periodo de planificación, que puede ser, por ejemplo, el uso de CPU máximo observado para microtareas en RT estricto durante cualquiera de los periodos de planificación durante toda la ejecución del programa. Asimismo, el presupuesto de núcleo planificado para microtareas en RT flexible puede determinarse como el uso de CPU observado máximo para microtareas en RT flexible durante dichos periodos de planificación. Se determinan presupuestos de núcleo planificados preferiblemente una vez para un programa en el momento del diseño del programa o durante una fase de calibración.

Cuando se ejecuta el programa con restricciones en tiempo real y cuando se determina que una tarea completa una de sus microtareas antes del microvencimiento de dicha microtarea, después puede liberarse en consecuencia una reserva correspondiente de tiempo de ejecución en exceso para dicha microtarea y por tanto hacer que esté disponible por el planificador para la ejecución de otras tareas. De manera correspondiente, cuando una microtarea en la categoría de RT estricto termina antes de su vencimiento, puede liberarse sucesivamente tiempo de tampón reservado para esa microtarea y, por tanto, hacer que esté disponible por el planificador para la ejecución de cargas de trabajo no en RT.

La figura 18 muestra presupuestos de núcleo reales adicionales para microtareas en RT estricto y en RT flexible en múltiples núcleos de CPU. Los presupuestos de núcleo reales se determinan preferiblemente en el inicio de cada periodo de planificación y tienen en cuenta las tareas en tiempo real que están activas durante dicho periodo de planificación, es decir aquellas tareas que está planificado que se ejecuten durante el periodo de planificación. El

presupuesto de núcleo real es el tiempo de ejecución en $CORE_y$ que está reservado en un determinado punto de tiempo dentro del periodo de planificación, que se estima preferiblemente, por ejemplo, basándose en los micropresupuestos μB_{xi} de todas las microtareas μT_{xi} de tareas en tiempo real T_x asignadas a $CORE_y$ que están activas en cualquier punto en el tiempo dentro del periodo de planificación. Los tiempos usados para determinar los presupuestos de núcleo activos tienen preferiblemente en cuenta $\Delta S_{\mu Txi}$ para cada una de las microtareas relevantes, en los que la diferencia $\Delta S_{\mu Txi}$ puede determinarse tal como se describió anteriormente.

La figura 18 muestra además que las reservas planificadas en la categoría de RT flexible pueden no ser suficientes, por ejemplo en los casos en los que el presupuesto de núcleo real para la ejecución de microtareas supera el presupuesto de núcleo planificado, lo cual en principio es posible pero no es probable según lo anterior. Este caso se muestra para el núcleo 2 y el núcleo 3 en la figura 18. Con respecto al núcleo 4, se han liberado sucesivamente reservas para microtareas en la categoría de RT flexible y de RT estricto de modo que se ha hecho que los tiempos correspondientes estén disponibles para la ejecución de tareas no en RT por el planificador sin comprometer en cuanto a las propiedades en tiempo real y garantías proporcionadas para tareas en tiempo real.

En el ejemplo de la figura 18, el mecanismo de penalización de núcleo puede explicarse de la siguiente manera: se considera, por ejemplo, que está ejecutándose una tarea no en RT en el núcleo 4, en la que las tareas en RT asignadas en el núcleo 4, se encuentran dentro de su tiempo, porque su uso real de tiempo de ejecución en esta unidad de ejecución es menor que el presupuesto planificado. El núcleo 4 usa recursos de chip ampliamente compartidos. Entonces, si, por ejemplo, se supera un presupuesto de núcleo, por ejemplo, en el núcleo 3, la ejecución de tareas no en RT o incluso tareas en RT flexible en otros núcleos, tales como el núcleo 4, que se encuentran “dentro de su tiempo” debe detenerse temporalmente. Cuando los presupuestos de núcleo de todos los núcleos alcanzan los valores especificados en el GRTM, la ejecución de tareas no en RT y en RT flexible en dichas ejecuciones para las que las tareas asignadas se encuentran “dentro de su tiempo” debe avanzar en un modo de ejecución normal, es decir sin que se penalice, se desprriorice o se detenga la unidad de ejecución.

Realizaciones según esta divulgación también incluyen un método y un aparato que sólo se refiere a aspectos de la primera fase según 100 de la figura 1 y asimismo a un método y un aparato que sólo se refiere a aspectos de la segunda fase según 200 de la figura 1, en los que cada una de las fases primera y segunda comprenden los diversos aspectos relacionados con cada fase respectivamente, que se han descrito anteriormente en detalle.

El experto entenderá que las realizaciones descritas anteriormente en el presente documento pueden implementarse por hardware, por software o por una combinación de software y hardware. Los módulos y funciones descritos en relación con realizaciones de la invención pueden implementarse, en su totalidad o en parte, por microprocesadores, ordenadores o circuitos de hardware de propósito especial que están programados de manera adecuada tal como para actuar según los métodos explicados en relación con realizaciones de la invención.

REIVINDICACIONES

1. Método para ejecutar un programa que incluye una pluralidad de tareas, en el que una o más tareas de la pluralidad de tareas tienen restricciones en tiempo real, comprendiendo el método las siguientes etapas para cada tarea T_x con restricciones en tiempo real:

(a) determinar un modelo de referencia en tiempo real,

en el que el modelo de referencia en tiempo real de la tarea T_x incluye una pluralidad de las microtareas μT_{xi}, i ∈ {1, ..., n}, que son una división de la tarea T_x, y un orden entre las microtareas μT_{xi} según todos los trayectos de ejecución posibles de la tarea T_x,

en el que, para cada microtarea μT_{xi}, i ∈ {1,..., n}, se determina un micropresupuesto μB_{xi} que es más pequeño que el tiempo de ejecución de peor caso, WCET, de la microtarea μT_{xi};

en el que el micropresupuesto μB_{xi} especifica un tiempo de ejecución para completar la ejecución de la microtarea μT_{xi} con una probabilidad inferior al 100% y por encima de un umbral de probabilidad predeterminado, y

en el que el micropresupuesto μB_{xi} de una microtarea μT_{xi} se determina basándose en análisis estadístico y/o interpretación abstracta de un programa de μT_{xi} y/o análisis estadístico de ejecuciones de μT_{xi}; y

en el que, para cada microtarea μT_{xi}, i ∈ {1,..., n}, basándose en los micropresupuestos μB_{xk}, k ∈ {1,..., n}, se determina un transcurso de referencia que especifica un transcurso estimado de la microtarea μT_{xi} en cualquier ejecución posible de la tarea T_x, de tal manera que todas las continuaciones posibles de ejecuciones de la tarea T_x desde la microtarea μT_{xi} en adelante cumplen las restricciones en tiempo real de la tarea T_x con una probabilidad por encima de un límite de tolerancia, en el que las restricciones en tiempo real de la tarea T_x se cumplen con una probabilidad por encima del límite de tolerancia si la ejecución de la tarea T_x se completa antes de un vencimiento de la tarea T_x con una probabilidad inferior al 100% y por encima de una determinada garantía de servicio mínima;

en el que el transcurso de referencia de la microtarea μT_{xi} incluye un microvencimiento μD_{xi}, que especifica un tiempo de respuesta hasta el que debe terminarse una ejecución de la microtarea μT_{xi}, en el que el tiempo de respuesta es una duración con respecto a un tiempo de activación ATT_X de la tarea T_x;

en el que la microtarea μT_{xi} debe terminarse hasta que cada una de las microtareas μT_{xk} k ∈ {1,..., i} en un trayecto crítico desde una microtarea inicial μT_{x1} hasta una microtarea μT_{xi} ha terminado la ejecución, en el que el tiempo de ejecución de cada microtarea μT_{xk} se estima mediante su micropresupuesto μB_{xk};

en el que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia si el transcurso real al final de la microtarea μT_{xi} supera el tiempo en el que la microtarea μT_{xi} debería haberse terminado; y

en el que el trayecto crítico hasta la microtarea μT_{xi} es un trayecto entre todos los trayectos de ejecución posibles de T_x desde la microtarea inicial μT_{x1} hasta la microtarea μT_{xi} que tiene el tiempo de ejecución predicho más largo;

(b) ejecutar la pluralidad de tareas y

(b1) determinar después de la ejecución de la microtarea μT_{xi} un transcurso real;

(b2) comparar el transcurso real con el transcurso de referencia;

(b3) basándose en la comparación, si se determina que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia, aumentar la prioridad de la tarea T_x.

2. Método según la reivindicación 1, en el que las microtareas μT_{xi} i ∈ {1,..., xÚltima} forman un entramado con μT_{x1} como microtarea inicial de T_x y $\mu T_{xÚltima}$ como microtarea final de T_x.

3. Método según la reivindicación 2, en el que el microvencimiento μD_{xi} se basa en la suma de los micropresupuestos μB_{xi} de las microtareas μT_{xk}, k ∈ {1,..., i} en el trayecto crítico hasta la microtarea μT_{xi}.

4. Método según una de las reivindicaciones anteriores,

en el que el transcurso de referencia de la microtarea μT_{xi} incluye un presupuesto de activación planificado B_{WCETxi} que especifica un presupuesto de tiempo de ejecución que es suficiente para completar la ejecución de la tarea T_x empezando desde la microtarea μT_{xi} de tal manera que sus restricciones en tiempo real se

cumplen con una probabilidad por encima del límite de tolerancia;

en el que el presupuesto de tiempo de ejecución se determina basándose en la suma de los micropresupuestos μB_{xk} de cada una de las microtareas μT_{xk}, k ∈ {i, ..., xÚltima} en un trayecto crítico activo dentro de T_x empezando en la microtarea μT_{xi};

en el que el trayecto crítico activo empezando en la microtarea μT_{xi} es un trayecto entre todos los trayectos de ejecución posibles de T_x desde μT_{xi} hasta una microtarea final $\mu T_{xÚltima}$ que tiene el tiempo de ejecución predicho más largo;

en el que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia si, antes de la ejecución de la microtarea μT_{xi}, el tiempo de respuesta real de la microtarea μT_{xi-1} es mayor que el microvencimiento μD_{xi-1}.

5. Método según la reivindicación 4, en el que las microtareas en una ejecución de la tarea T_x se clasifican en microtareas en tiempo real flexible μT_{xi}, i ∈ {1, ..., xrt-1} y microtareas en tiempo real estricto μT_{xi}, i ∈ {xrt, ...,xÚltima},

en el que un tiempo de ejecución de una microtarea en tiempo real flexible μT_{xi} se estima mediante su micropresupuesto μB_{xi}; y

en el que un tiempo de ejecución de una microtarea en tiempo real estricto μT_{xi} se estima mediante su micropresupuesto μB_{xi} más un tiempo de tampón $BT(\mu T_{xi})$, siendo el tiempo de tampón un tiempo adicional para garantizar que μT_{xi} termina con una certeza del 100% dentro del tiempo estimado; y

en el que el presupuesto de tiempo de ejecución se determina basándose además en una suma de los tiempos de ejecución estimados de microtarea en tiempo real flexible y en tiempo real estricto μT_{xi}.

6. Método según una de las reivindicaciones anteriores, que comprende la siguiente etapa adicional:

añadir una o más instrucciones a un programa de la tarea T_x, provocando las instrucciones la emisión de un acontecimiento de seguimiento E_{xi}, al final de la ejecución de la microtarea μT_{xi}, comprendiendo el acontecimiento de seguimiento un identificador único de una porción de la ejecución de la tarea T_x,

en el que el identificador único comprende un identificador de una unidad de hardware que ejecuta la tarea T_x, un identificador de la tarea T_x y un identificador del acontecimiento de seguimiento E_{xi}.

7. Método según una de las reivindicaciones anteriores,

en el que cada tarea de la pluralidad de tareas se asigna a una unidad de ejecución fija durante un periodo de planificación y la unidad de ejecución fija es un núcleo de una pluralidad de núcleos de un procesador de múltiples núcleos, y

en el que se reserva tiempo de ejecución en la unidad de ejecución fija según tiempos de ejecución estimados de todas las microtareas μT_{xi} de todas las tareas en tiempo real T_x asignadas a la unidad de ejecución fija, en el que la reserva se realiza por un planificador que es un planificador de OS.

8. Método según la reivindicación 7, en el que el presupuesto B_{WCETx} de la tarea T_x es el presupuesto de activación planificado B_{WCETx1} de la microtarea inicial μT_{x1} de la tarea T_x, en

el que puede asignarse una pluralidad de tareas en una misma unidad de ejecución siempre que se cumplan las siguientes restricciones:

- la suma de tiempos de ejecución estimados para las microtareas en tiempo real estricto de cada tarea de la pluralidad de tareas no supera una primera porción de un uso máximo de la unidad de ejecución durante el periodo de planificación; y

- la suma de los presupuestos de tareas en tiempo real asignadas a la misma unidad de ejecución no supera una determinada segunda porción del uso máximo de la unidad de ejecución durante el periodo de planificación.

9. Método según la reivindicación 7 u 8,

en el que, si una diferencia entre un tiempo de activación real de la microtarea μT_{xi} y un tiempo de activación planificado de la microtarea μT_{xi} es negativa, se libera una porción del tiempo de ejecución dentro del periodo de planificación reservada en la unidad de ejecución para la ejecución de la tarea T_x con restricciones en tiempo real y por tanto está disponible para ejecutar otras tareas durante el periodo de planificación,

en el que el tiempo de activación planificado de la microtarea μT_{xi} es el microvencimiento μD_{xi-1} de la microtarea anterior μT_{xi-1}, y

en el que la cantidad de tiempo liberado es inferior o igual a una diferencia entre el tiempo real restante hasta el vencimiento de T_x y el presupuesto de activación planificado B_{WCETxi}.

10. Aparato para ejecutar un programa que incluye una pluralidad de tareas, en el que una o más tareas de la pluralidad de tareas tienen restricciones en tiempo real, comprendiendo el aparato las siguientes unidades de hardware:

una o más unidades de ejecución adaptadas para ejecutar la pluralidad de tareas;

una unidad de almacenamiento adaptada para almacenar un modelo de referencia en tiempo real,

en el que el modelo de referencia en tiempo real de la tarea T_x incluye una pluralidad de las microtareas μT_{xi}, $i \in \{1,..., n\}$, que son una división de la tarea T_x, y un orden entre las microtareas μT_{xi} según todos los trayectos de ejecución posibles de la tarea T_x,

en el que, para cada microtarea μT_{xi}, $i \in \{1,..., n\}$, se determina un micropresupuesto μB_{xi} que es más pequeño que el tiempo de ejecución de peor caso, WCET, de la microtarea μT_{xi};

en el que el micropresupuesto μB_{xi} especifica un tiempo de ejecución para completar la ejecución de la microtarea μT_{xi} con una probabilidad inferior al 100% y por encima de un umbral de probabilidad predeterminado, y

en el que el micropresupuesto μB_{xi} de una microtarea μT_{xi} se determina basándose en análisis estadístico y/o interpretación abstracta de un programa de μT_{xi} y/o análisis estadístico de ejecuciones de μT_{xi}; y

en el que, para cada microtarea μT_{xi}, $i \in \{1,..., n\}$, basándose en los micropresupuestos μB_{xk}, $k \in \{1,..., n\}$, se determina un transcurso de referencia que especifica un transcurso estimado de la microtarea μT_{xi} en cualquier ejecución posible de la tarea T_x, de tal manera que todas las continuaciones posibles de ejecuciones de la tarea T_x desde la microtarea μT_{xi} en adelante cumplen las restricciones en tiempo real de la tarea T_x con una probabilidad por encima de un límite de tolerancia, en el que las restricciones en tiempo real de la tarea T_x se cumplen con una probabilidad por encima del límite de tolerancia si la ejecución de la tarea T_x se completa antes de un vencimiento de la tarea T_x con una probabilidad inferior al 100% y por encima de una determinada garantía de servicio mínima;

en el que el transcurso de referencia de la microtarea μT_{xi} incluye un microvencimiento μD_{xi}, que especifica un tiempo de respuesta hasta el que debe terminarse una ejecución de la microtarea μT_{xi}, en el que el tiempo de respuesta es una duración con respecto a un tiempo de activación ATT_x de la tarea T_x;

en el que la microtarea μT_{xi} debe terminarse hasta que cada una de las microtareas μT_{xk} $k \in \{1,..., i\}$ en un trayecto crítico desde una microtarea inicial μT_{x1} hasta una microtarea μT_{xi} ha terminado la ejecución, en el que el tiempo de ejecución de cada microtarea μT_{xk} se estima mediante su micropresupuesto μB_{xk};

en el que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia si el transcurso real al final de la microtarea μT_{xi} supera el tiempo en el que la microtarea μT_{xi} debería haberse terminado; y

en el que el trayecto crítico hasta la microtarea μT_{xi} es un trayecto entre todos los trayectos de ejecución posibles de T_x desde la microtarea inicial μT_{x1} hasta la microtarea μT_{xi} que tiene el tiempo de ejecución predicho más largo;

una unidad de monitorización de acontecimientos adaptada para determinar después de la ejecución de la microtarea μT_{xi} un transcurso real;

una unidad de monitorización de tiempo de presupuesto adaptada para comparar el transcurso real con el transcurso de referencia;

una unidad de planificación de hardware adaptada para aumentar la prioridad de la tarea T_x basándose en un resultado de comparación de la unidad de monitorización de tiempo de presupuesto, si se determina que las restricciones en tiempo real de la tarea T_x no se cumplen con una probabilidad por encima del límite de tolerancia.

11. Aparato según la reivindicación 10, que comprende además:

una unidad de calibración adaptada para llevar a cabo mediciones de tiempo de ejecución de una microtarea y para determinar, basándose en las mediciones, información sobre el tiempo de ejecución de la

microtarea, y para almacenar la información en el modelo de referencia en tiempo real.

12. Aparato según la reivindicación 10 u 11,

en el que la unidad de monitorización de acontecimientos está adaptada además para mantener, para cada tarea en tiempo real T_x, un acontecimiento de seguimiento emitido más recientemente E_{xi} que incluye un punto en el tiempo CT_{Exi} en el que se emitió el acontecimiento de seguimiento E_{xi};

en el que el aparato comprende además una unidad de monitorización de vencimiento adaptada para determinar una diferencia $\Delta S_{\mu Txi} = CT_{Exi-1} - \mu D_{xi-1}$ entre un tiempo de activación real CT_{Exi-1} de la microtarea μT_{xi} y un tiempo de activación planificado de la microtarea μT_{xi}, y/o para detectar si una ejecución de una microtarea μT_{xi} termina después del microvencimiento μD_{xi};

en el que la unidad de monitorización de tiempo de presupuesto está adaptada además para determinar, para cada tarea en tiempo real T_x, una desviación entre un transcurso planificado de la tarea T_x y un transcurso real de la tarea T_x, en el que el transcurso planificado de la tarea T_x antes de la ejecución de la microtarea μT_{xi} es el presupuesto de activación planificado B_{WCETxi}, y en el que el transcurso real de la tarea T_x se estima basándose en una cantidad de tiempo de CPU usado por la tarea T_x hasta el tiempo de respuesta CT_{Exi-1} y la diferencia $\Delta S_{\mu Txi}$; y

en el que la unidad de planificación de hardware está adaptada además para generar un valor de control en tiempo real para una tarea en tiempo real T_x basándose en una desviación del transcurso planificado de T_x con respecto al transcurso real de la tarea T_x, y en el que el valor de control en tiempo real se señaliza a un planificador de OS.

FIG. 1


Fase I: desarrollo de un modelo de referencia en tiempo real y calibración
Fase II: ejecución de la tarea en tiempo real con restricciones en tiempo real
100
200

FIG. 2

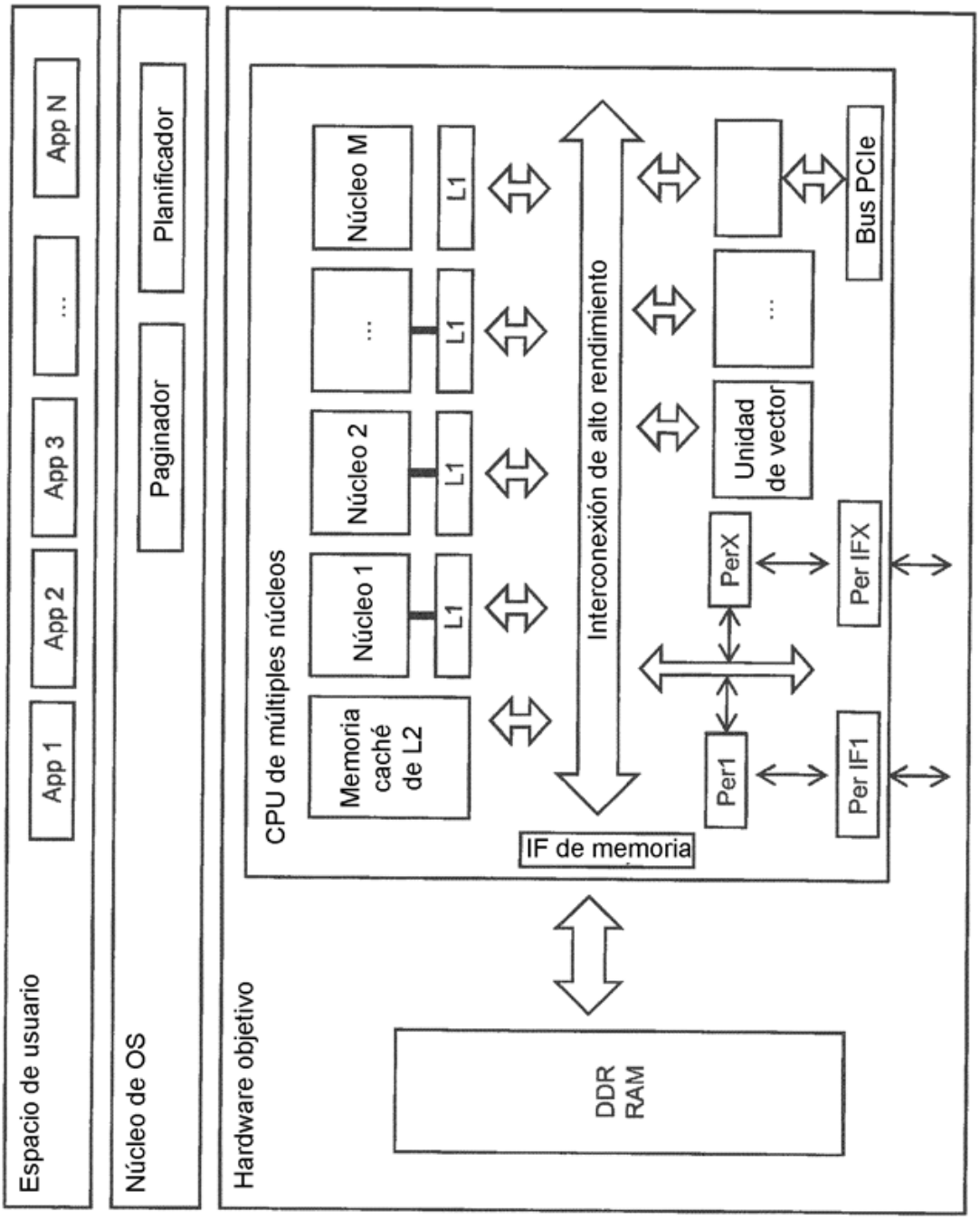
Espacio de usuario
App 1
App 2
App 3
...
App N
Núcleo de OS
Paginador
Planificador
Hardware objetivo
CPU de múltiples núcleos
Memoria caché de L2
Núcleo 1
Núcleo 2
...
Núcleo M
L1
L1
L1
L1
Interconexión de alto rendimiento
IF de memoria
Per1
PerX
Unidad de vector
...
Per IF1
Per IFX
Bus PCIe
DDR RAM

FIG. 3

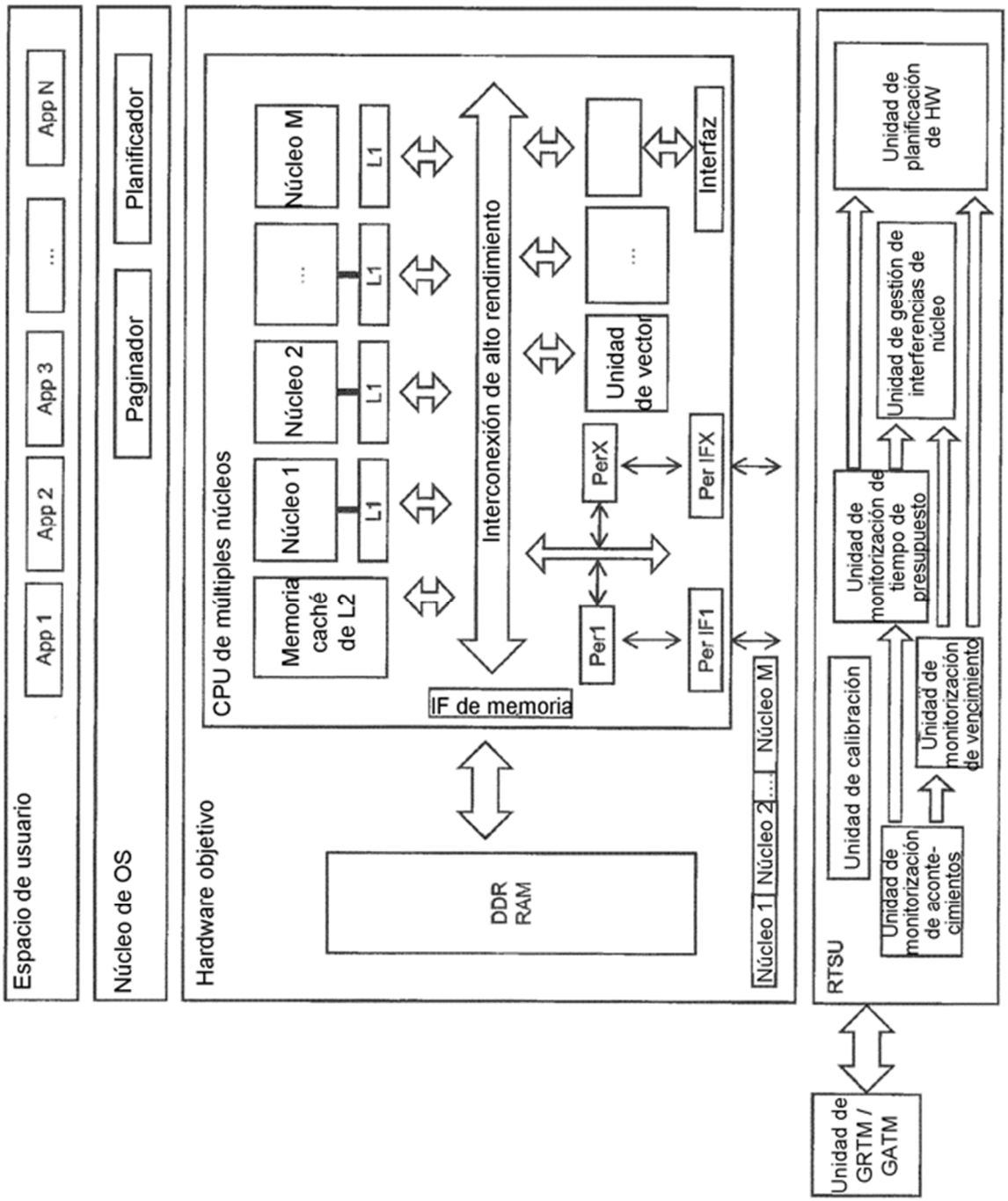
Espacio de usuario
App 1
App 2
App 3
...
App N
Núcleo de OS
Paginador
Planificador
Hardware objetivo
CPU de múltiples núcleos
Memoria caché de L2
Núcleo 1
Núcleo 2
...
Núcleo M
L1
Interconexión de alto rendimiento
IF de memoria
Per1
PerX
Per IF1
Per IFX
Unidad de vector
...
Interfaz
DDR RAM
Núcleo 1
Núcleo 2
...
Núcleo M
RTSU
Unidad de calibración
Unidad de monitorización de acontecimientos
Unidad de monitorización de vencimiento
Unidad de monitorización de tiempo de presupuesto
Unidad de gestión de interferencias de núcleo
Unidad de planificación de HW
Unidad de GRTM / GATM

FIG. 4

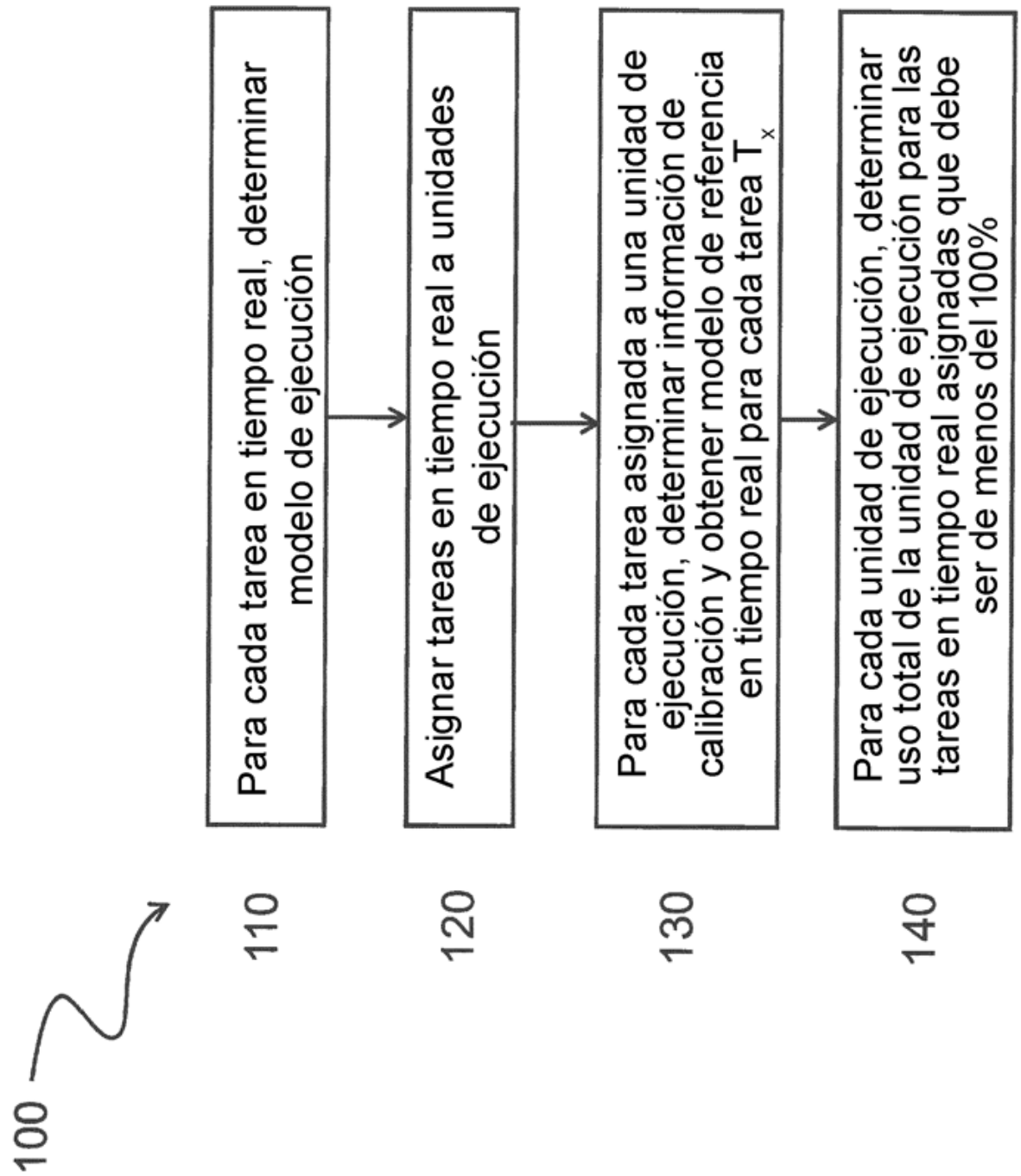
100
110
Para cada tarea en tiempo real, determinar modelo de ejecución
120
Asignar tareas en tiempo real a unidades de ejecución
130
Para cada tarea asignada a una unidad de ejecución, determinar información de calibración y obtener modelo de referencia en tiempo real para cada tarea T_x
140
Para cada unidad de ejecución, determinar uso total de la unidad de ejecución para las tareas en tiempo real asignadas que debe ser de menos del 100%

FIG. 5

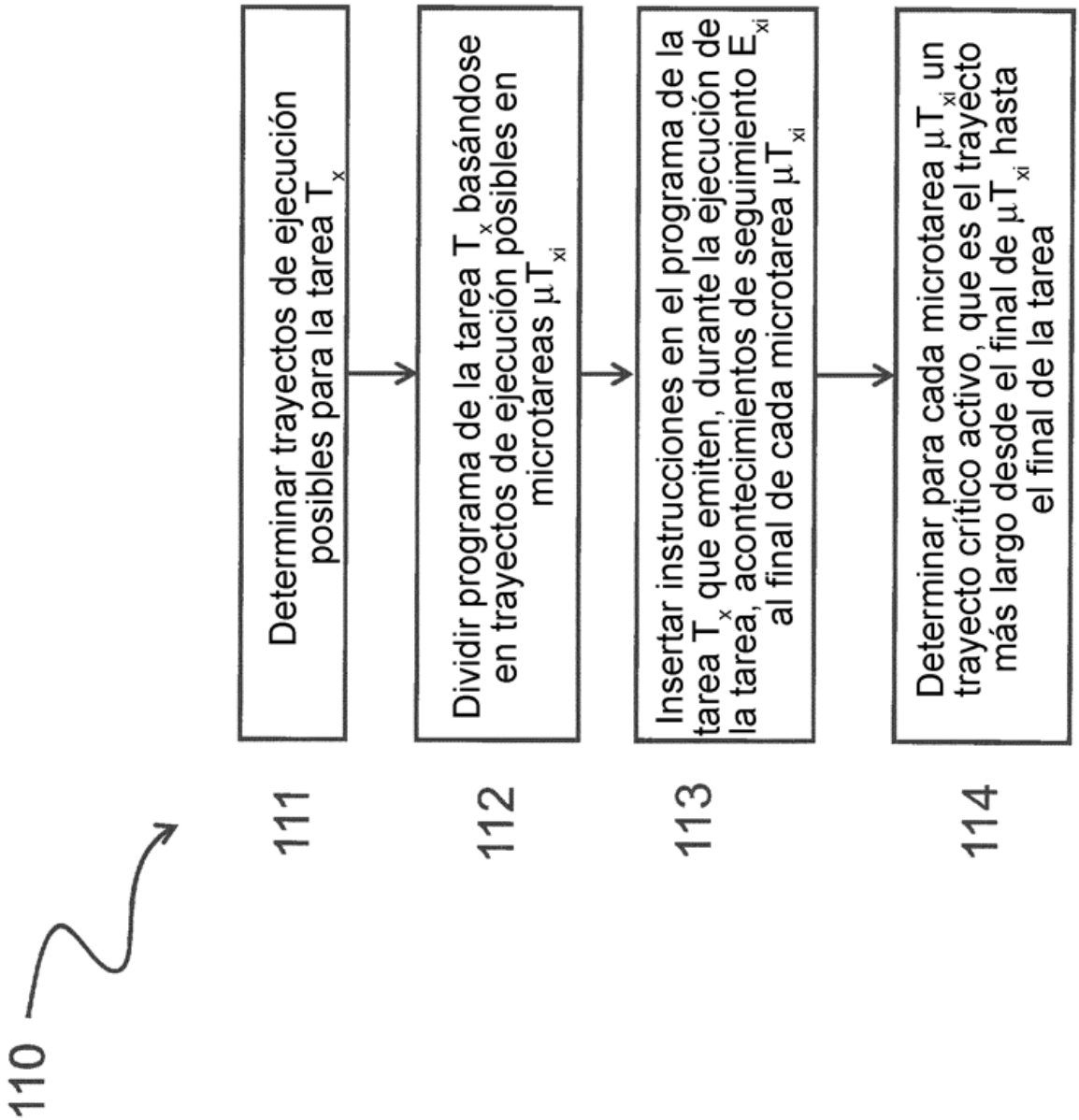
110
111
Determinar trayectos de ejecución posibles para la tarea T_x
112
Dividir programa de la tarea T_x basándose en trayectos de ejecución posibles en microtareas μT_{xi}
113
Insertar instrucciones en el programa de la tarea T_x que emiten, durante la ejecución de la tarea, acontecimientos de seguimiento E_{xi} al final de cada microtarea μT_{xi}
114
Determinar para cada microtarea μT_{xi} un trayecto crítico activo, que es el trayecto más largo desde el final de μT_{xi} hasta el final de la tarea

FIG. 6

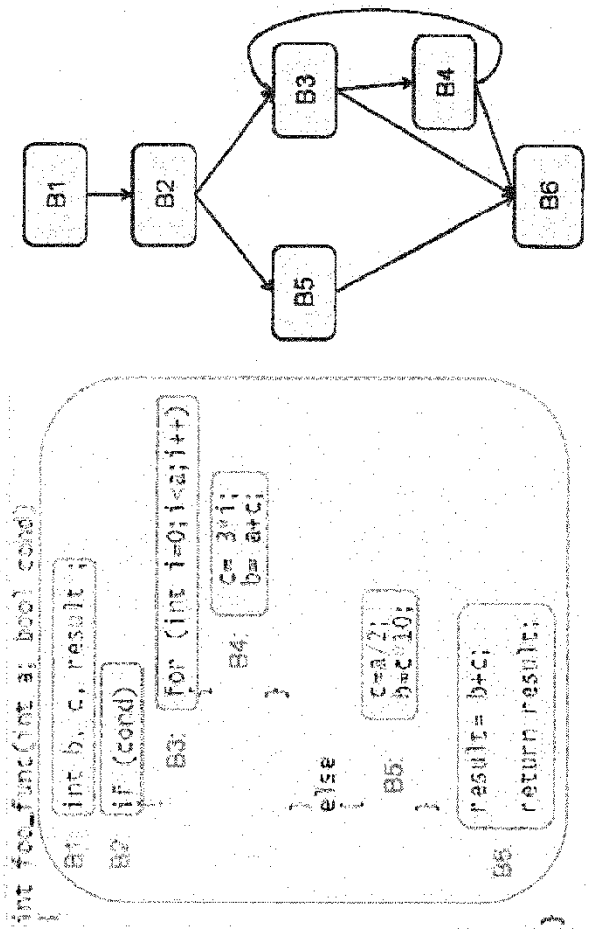
B1
B2
B3
B4
B5
B6
int foo_func(int a, bool cond)
{
B1: int b, c, result;
B2: if (cond)
{
B3: for (int i=0;i<a;i++)
{
B4: c= 3*i;
b= a+c;
}
}
else
{
B5: c=a/2;
b=c*10;
}
B6: result= b+c;
return result;
}

FIG. 7

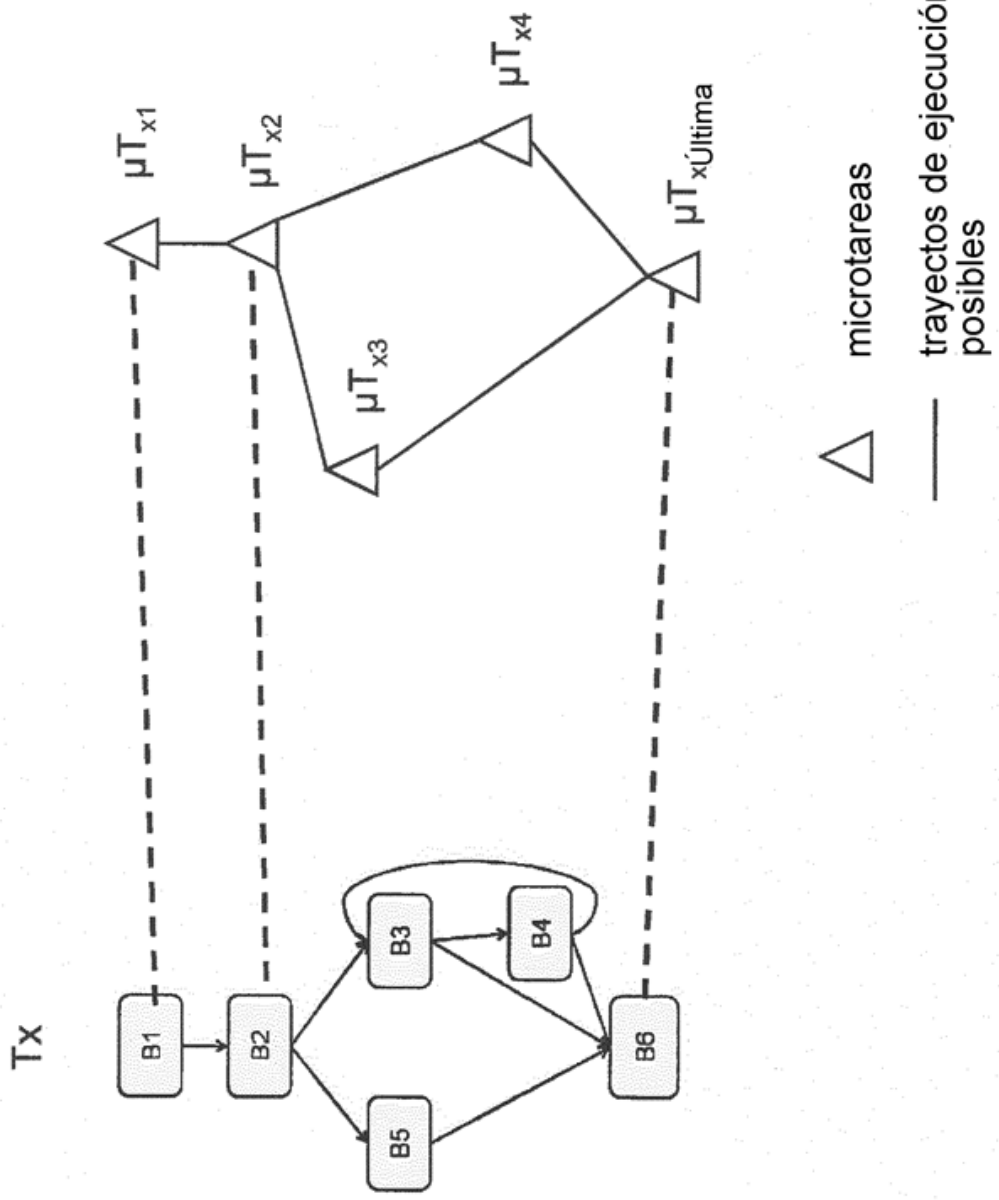
Tx
B1
B2
B3
B4
B5
B6
μT x1
μT x2
μT x3
μT x4
μT xÚltima
microtareas
trayectos de ejecución posibles

FIG. 8

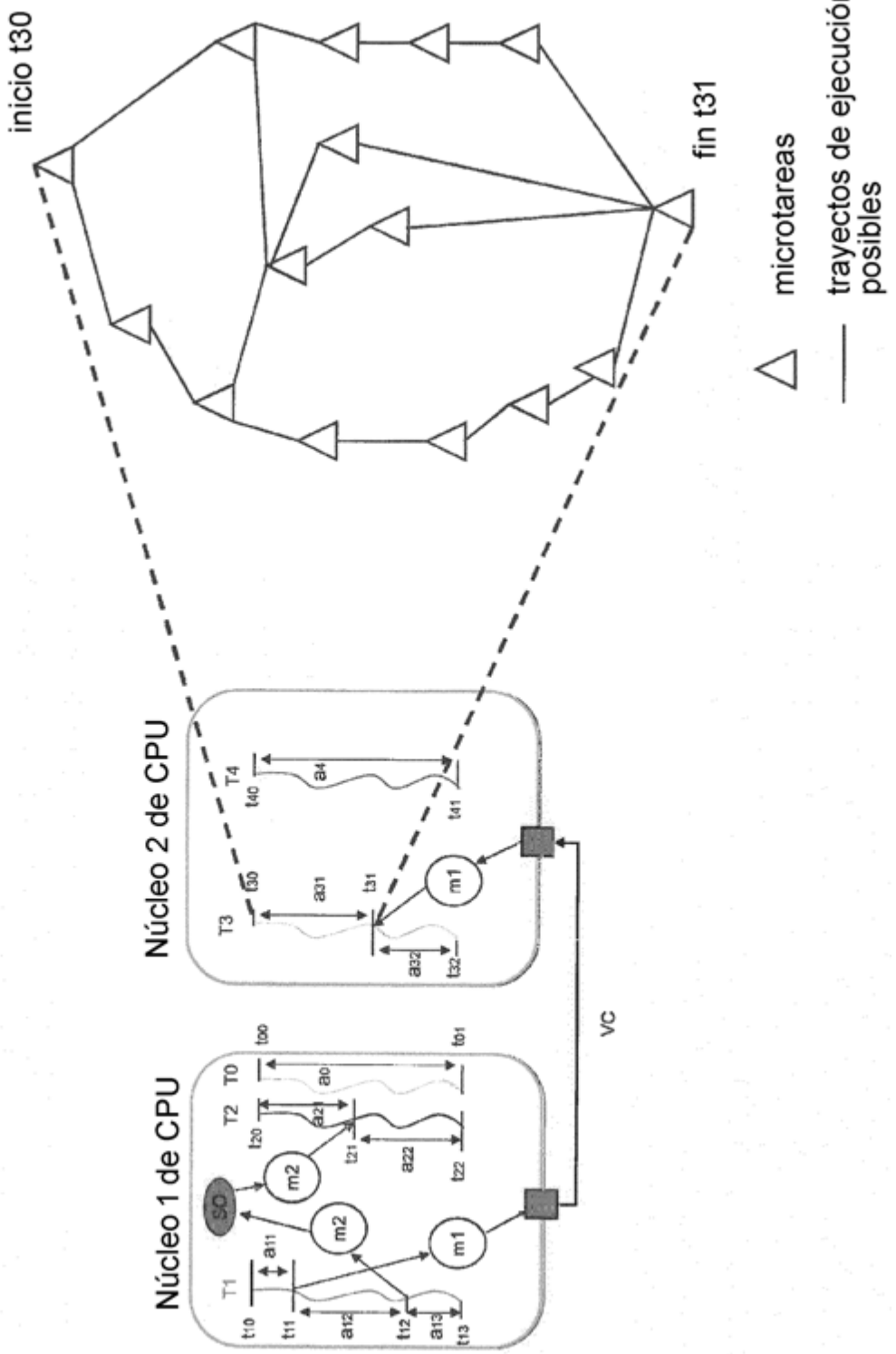
inicio t30
fin t31
microtareas
trayectos de ejecución posibles
Núcleo 2 de CPU
Núcleo 1 de CPU
VC
T0
T1
T2
T3
T4
m1
m2
SO

FIG. 9

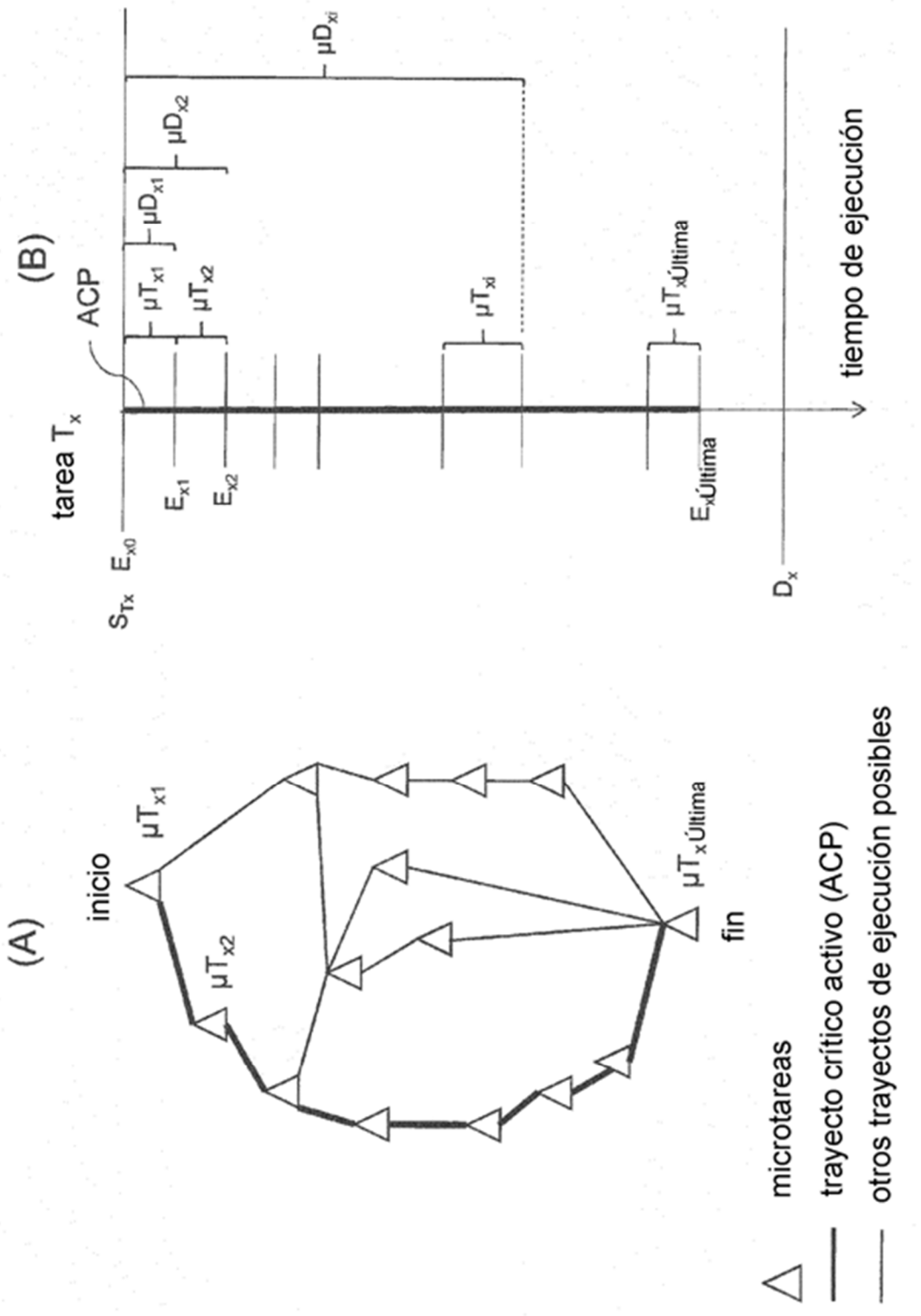
(A)
inicio
μT_{x1}
μT_{x2}
$\mu T_{x\text{Última}}$
fin
microtareas
trayecto crítico activo (ACP)
otros trayectos de ejecución posibles
(B)
tarea T_x
ACP
μT_{x1}
μT_{x2}
μT_{xi}
$\mu T_{x\text{Última}}$
μD_{x1}
μD_{x2}
μD_{xi}
S_{Tx} E_{x0}
E_{x1}
E_{x2}
$E_{x\text{Última}}$
D_x
tiempo de ejecución

FIG. 10

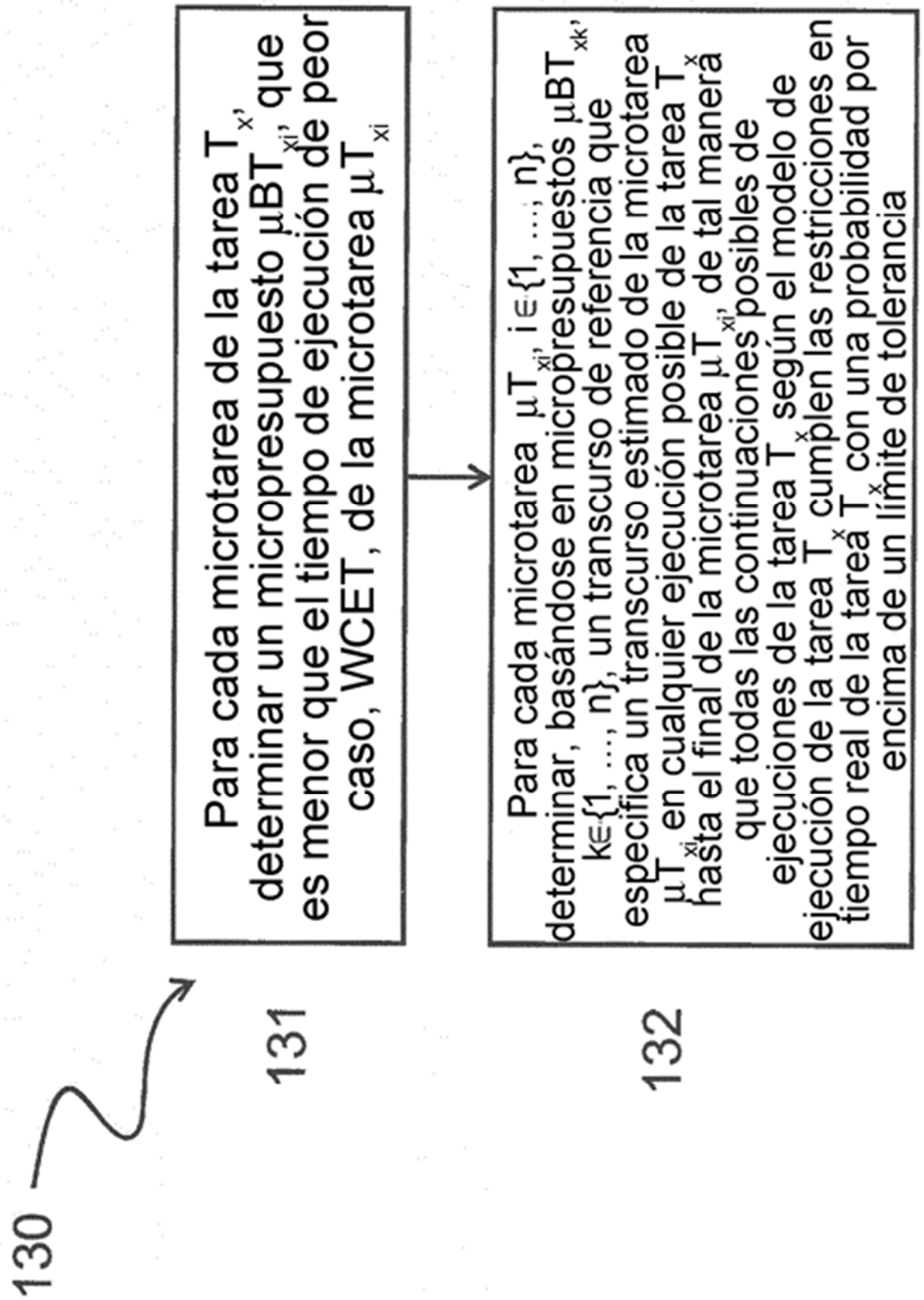
130
131
Para cada microtarea de la tarea T_x, determinar un micropresupuesto μBT_{xi}, que es menor que el tiempo de ejecución de peor caso, WCET, de la microtarea μT_{xi}
132
Para cada microtarea μT_{xi}, $i \in \{1, \ldots, n\}$, determinar, basándose en micropresupuestos μBT_{xk}, $k \in \{1, \ldots, n\}$, un transcurso de referencia que especifica un transcurso estimado de la microtarea μT_{xi} en cualquier ejecución posible de la tarea T_x hasta el final de la microtarea μT_{xi}, de tal manera que todas las continuaciones posibles de ejecuciones de la tarea T_x según el modelo de ejecución de la tarea T_x cumplen las restricciones en tiempo real de la tarea T_x con una probabilidad por encima de un límite de tolerancia

FIG. 11

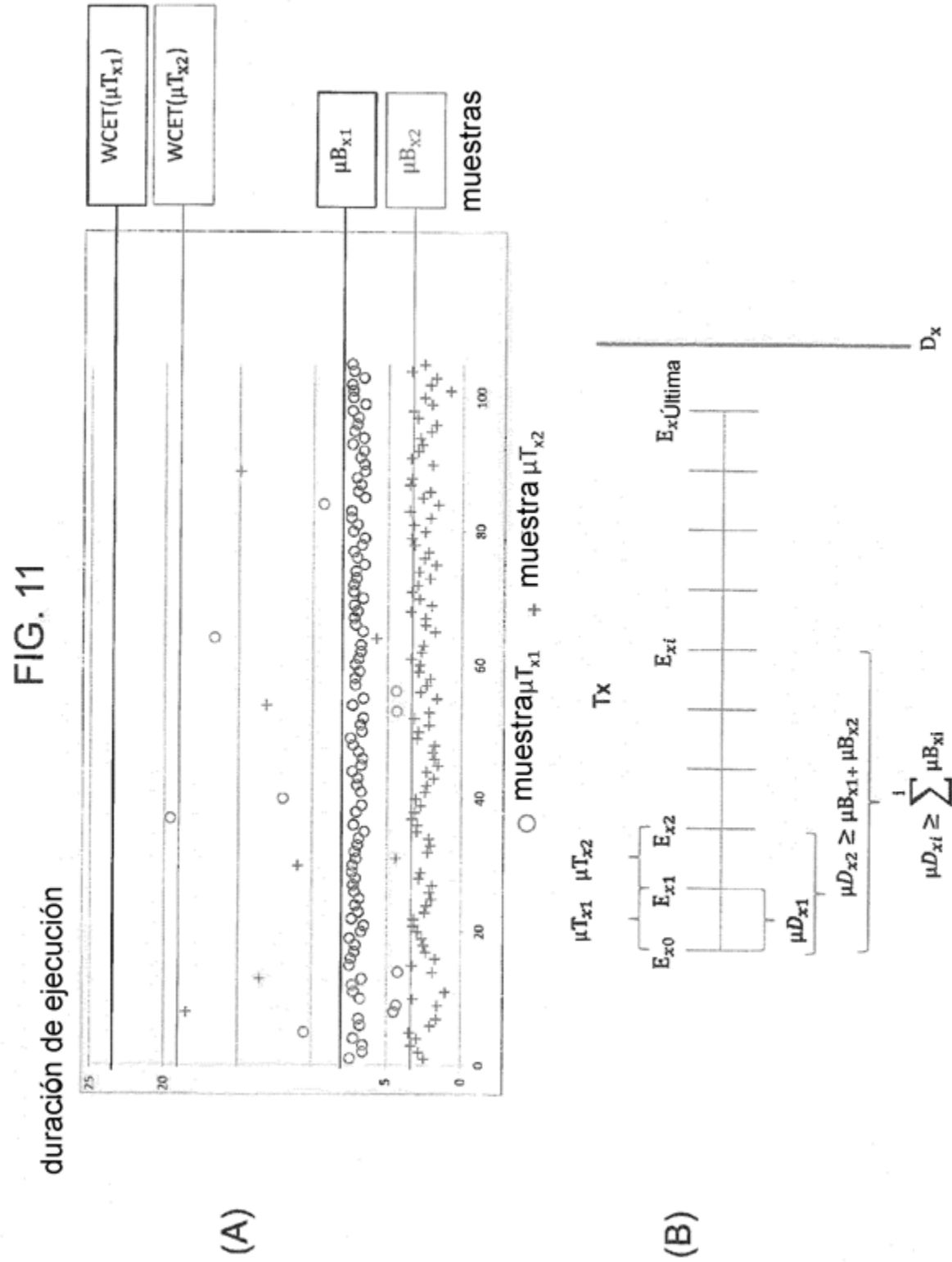
(A)
duración de ejecución
WCET(μT_{x1})
WCET(μT_{x2})
μB_{x1}
μB_{x2}
muestras
○ muestraμT_{x1} + muestra μT_{x2}
(B)
Tx
μT_{x1} μT_{x2}
E_{x0} E_{x1} E_{x2}
E_{xi}
$E_{x\text{Última}}$
D_x
μD_{x1}
$\mu D_{x2} \geq \mu B_{x1} + \mu B_{x2}$
$\mu D_{xi} \geq \sum_{k=1}^{i} \mu B_{xi}$

FIG. 12

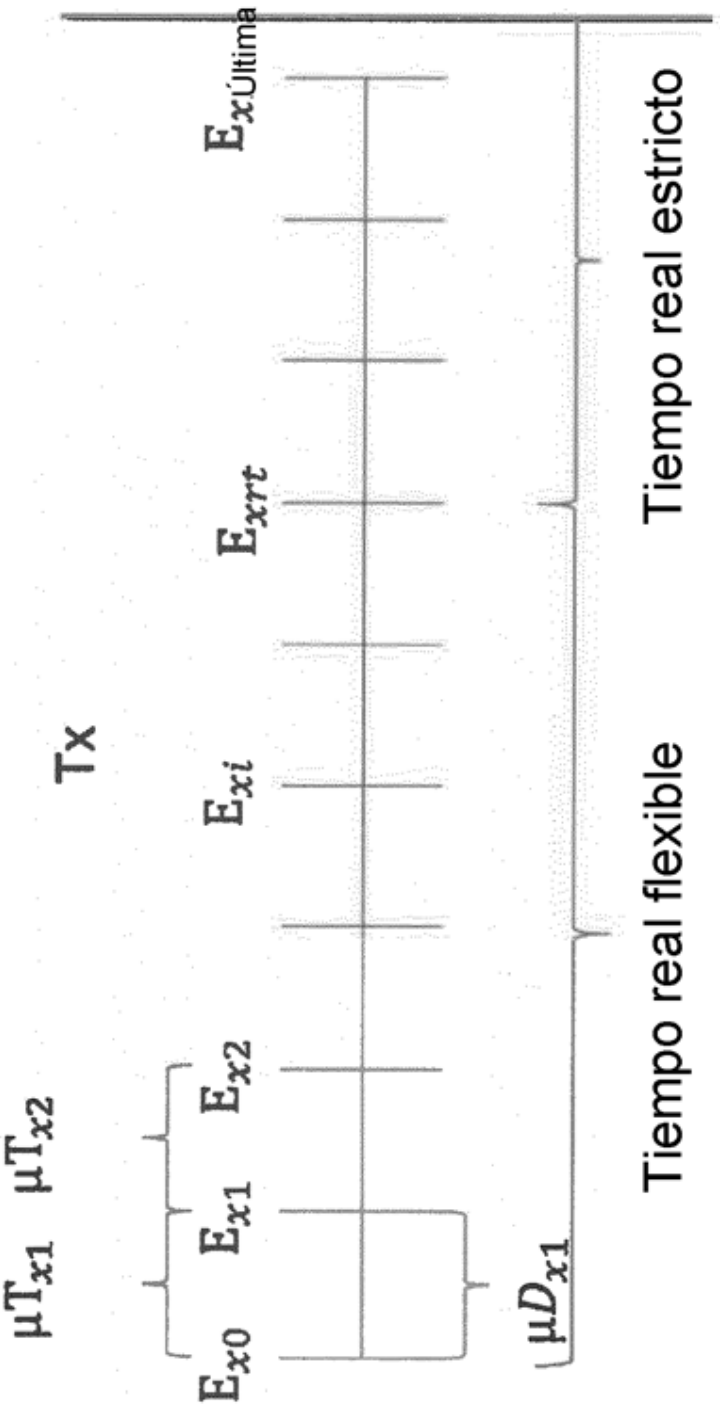
Tx
E_{x0}
E_{x1}
E_{x2}
E_{xi}
E_{xrt}
$E_{xÚltima}$
μT_{x1}
μT_{x2}
μD_{x1}
Tiempo real flexible
Tiempo real estricto

FIG. 13

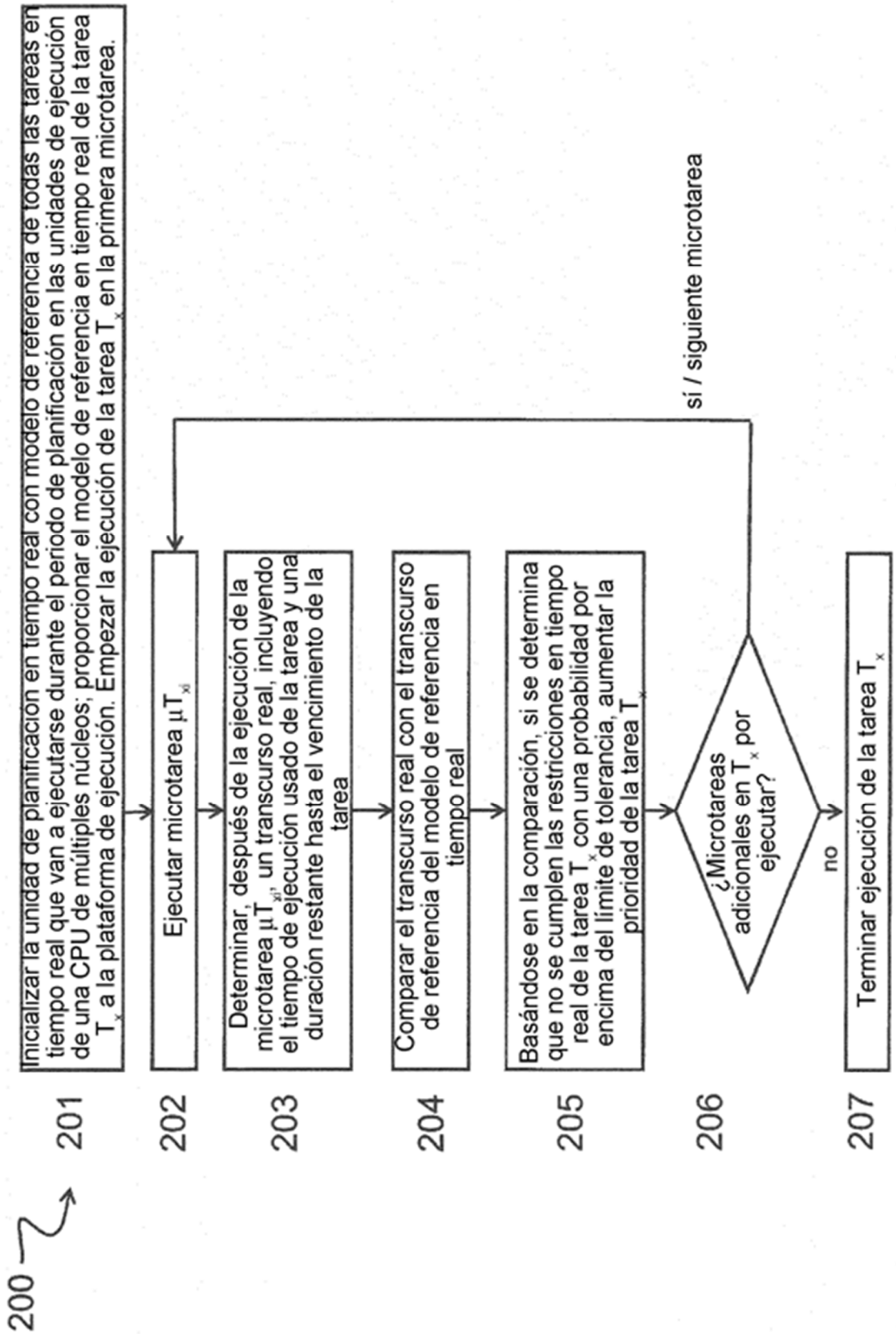
200
201
Inicializar la unidad de planificación en tiempo real con modelo de referencia de todas las tareas en tiempo real que van a ejecutarse durante el periodo de planificación en las unidades de ejecución de una CPU de múltiples núcleos; proporcionar el modelo de referencia en tiempo real de la tarea T_x a la plataforma de ejecución. Empezar la ejecución de la tarea T_x en la primera microtarea.
202
Ejecutar microtarea μT_{xi}
203
Determinar, después de la ejecución de la microtarea μT_{xi}, un transcurso real, incluyendo el tiempo de ejecución usado de la tarea y una duración restante hasta el vencimiento de la tarea
204
Comparar el transcurso real con el transcurso de referencia del modelo de referencia en tiempo real
205
Basándose en la comparación, si se determina que no se cumplen las restricciones en tiempo real de la tarea T_x con una probabilidad por encima del límite de tolerancia, aumentar la prioridad de la tarea T_x
206
¿Microtareas adicionales en T_x por ejecutar?
sí / siguiente microtarea
no
207
Terminar ejecución de la tarea T_x

FIG. 14

	N.º ACP acontecimientosx	µB_xi	WCET (µTxi)	BT_xi	P(µB_xi)
total	60	8	32	24	0,75
RT estricto	17				
RT flexible	43				

FIG. 15

	CT(E_xi) real	Δδ_μTxi	B_WCETxi	B_Softxi	tiempo ejecución (μT_xi)		acontecimientos de RT flexible restantes	acontecimientos totales restantes en ACP	probabilidad residual de no cumplir el vencimiento
1	0	0	1016	368	8	Tiempo real flexible	42	59	3,00927E-36
2	8	0	1008	360	8		41	58	1,20371E-35
3	18	2	1002	354	8		40	57	4,81482E-35
4	29	3	995	347	8		39	56	1,92593E-34
5	33	-4	980	332	8		38	55	7,70372E-34
6	39	-2	974	326	8		37	54	3,08149E-33
7	48	1	969	321	8		36	53	1,2326E-32
8	57	1	961	313	8		35	52	4,93038E-32
9	62	-3	949	301	8		34	51	1,97215E-31
10	68	-2	942	294	8		33	50	7,88861E-31
11	75	-1	935	287	8		32	49	3,15544E-30
12	85	2	930	282	8		31	48	1,26218E-29
13	93	0	920	272	8		30	47	5,04871E-29
14	101	0	912	264	8		29	46	2,01948E-28
15	114	5	909	261	8		28	45	8,07794E-28
16	119	-3	893	245	8		27	44	3,23117E-27
17	125	-2	886	238	8		26	43	1,29247E-26
18	134	1	881	233	8		25	42	5,16988E-26
19	142	0	872	224	8		24	41	2,06795E-25
20	150	0	864	216	8		23	40	8,27181E-25
21	153	-5	851	203	8		22	39	3,30872E-24
22	162	1	849	201	8		21	38	1,32349E-23
23	173	3	843	195	8		20	37	5,29396E-23
24	179	-2	830	182	8		19	36	2,11758E-22
25	191	4	828	180	8		18	35	8,47033E-22
26	201	2	818	170	8		17	34	3,38813E-21
27	210	1	809	161	8		16	33	1,35525E-20
28	218	0	800	152	8		15	32	5,42101E-20
29	227	1	793	145	8		14	31	2,1684E-19
30	237	2	786	138	8		13	30	8,67362E-19
31	245	0	776	128	8		12	29	3,46945E-18
32	256	3	771	123	8		11	28	1,38778E-17
33	260	-4	756	108	8		10	27	5,55112E-17
34	271	3	755	107	8		9	26	2,22045E-16
35	281	2	746	98	8		8	25	8,88178E-16
36	288	-1	735	87	8		7	24	3,55271E-15
37	296	0	728	80	8		6	23	1,42109E-14
38	305	1	721	73	8		5	22	5,68434E-14
39	315	2	714	66	8		4	21	2,27374E-13
40	324	1	705	57	8		3	20	9,09495E-13
41	332	0	696	48	8		2	19	3,63798E-12
42	341	1	689	41	8		1	18	1,45519E-11
43	349	0	680	32	8		0	17	5,82077E-11
44	357	0	648	0	8	RT estricto	0	16	2,32831E-10
45	365	0	608	0	8		0	15	9,31323E-10
46	373	0	568	0	8		0	14	3,72529E-09
47	381	0	528	0	8		0	13	1,49012E-08
48	389	0	488	0	8		0	12	5,96046E-08
49	397	0	448	0	8		0	11	2,38419E-07
50	405	0	408	0	8		0	10	9,53674E-07
51	413	0	368	0	8		0	9	3,8147E-06
52	421	0	328	0	8		0	8	1,52588E-05
53	429	0	288	0	8		0	7	6,10352E-05
54	437	0	248	0	8		0	6	0,000244141
55	445	0	208	0	8		0	5	0,000976563
56	453	0	168	0	8		0	4	0,00390625
57	461	0	128	0	8		0	3	0,015625
58	469	0	88	0	8		0	2	0,0625
59	477	0	48	0	8		0	1	0,25
60	485	0	8	0	8		0	0	1

FIG. 16

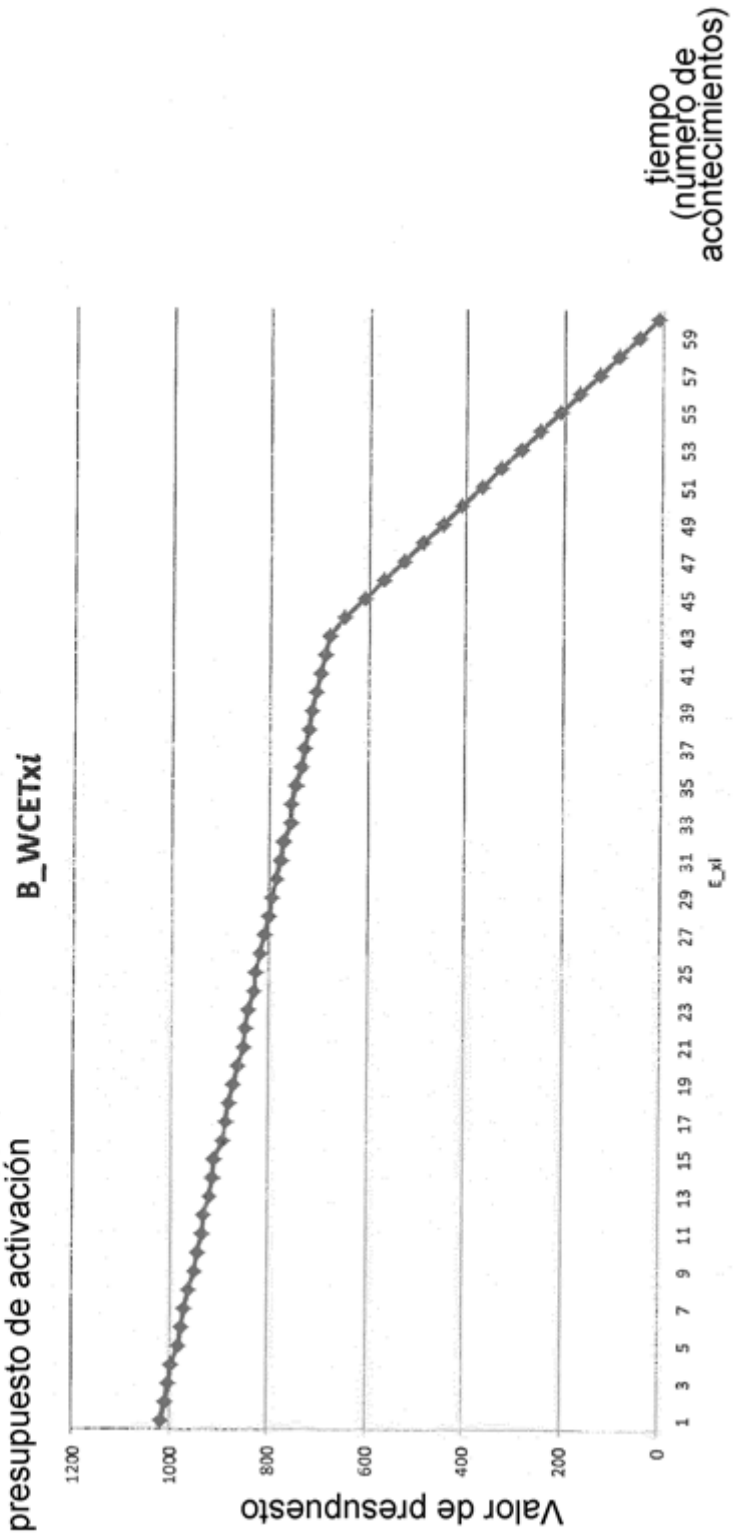
presupuesto de activación
B_WCETxi
Valor de presupuesto
1200
1000
800
600
400
200
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59
E_xi
tiempo
(número de
acontecimientos)

FIG. 17

	Reservado	Realmente usado	Presupuesto resta
WCET (Tx)	1920	1920	0
Bx	1016	485	531
Presupuesto RT flexible	368	349	19
Presupuesto RT estricto	648	128	520

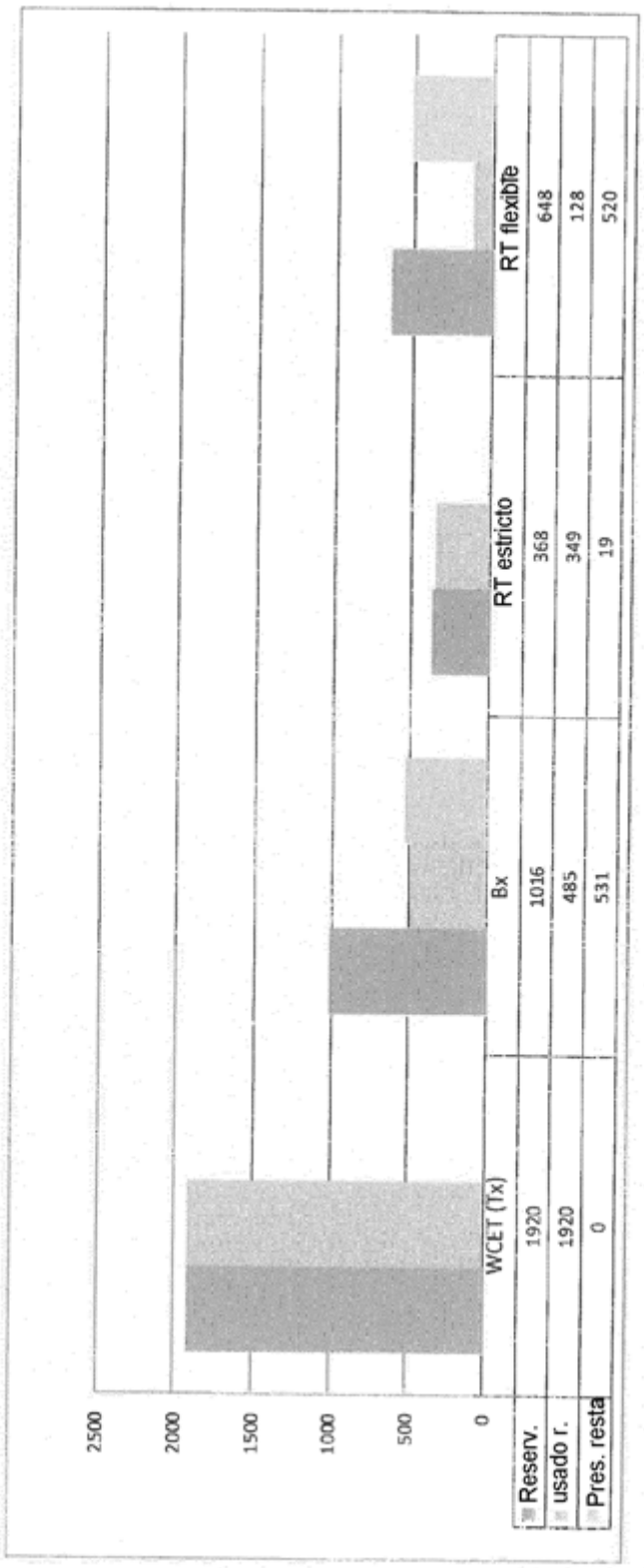

FIG. 18

presupuesto disponible total		presupuesto de núcleo planificado			presupuesto de núcleo real		
		RT est. (máx)	RT flex. (máx)	No RT (máx)	RT est.	RT flex.	No RT
1000	Núcleo 1	500	250	250	200	240	560
1000	Núcleo 2	500	250	250	400	400	200
1000	Núcleo 3	500	250	250	400	450	150
1000	Núcleo 4	500	250	250	200	150	650