(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 552 739**

(21) Número de solicitud: 201531155

(51) Int. CI.:

G06F 13/42 (2006.01)

(12) SOLICITUD DE PATENTE A1

(22) Fecha de presentación:

03.08.2015

(43) Fecha de publicación de la solicitud:

01.12.2015

(71) Solicitantes:

SIGNADYNE SPAIN, S.L. (100.0%)
Parc Mediterrani de la Tecnología, B1 Av. Canal Olimpic s/n
08860 Castelldefels (Barcelona) ES

(72) Inventor/es:

OLIVERIO, Néstor Hugo y
ALMENDROS PARRA, Marc

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

(54) Título: **Sistema de ejecución determinista en tiempo, distribuida y sincronizada para aplicaciones de control, test y medida.**

(57) Resumen:

Sistema de ejecución determinista en tiempo, distribuida y sincronizada para aplicaciones de control, test y medida.

Sistema de ejecución determinista en tiempo distribuida sincronizada para módulos (M0, ..., Mn) de control, test y medida que comparten al menos una señal de disparo y una señal de reloj, y que comprenden un procesador determinista en tiempo (100), que usa la señal de reloj y una o más de las señales de disparo compartidas por todos los módulos (M0, ..., Mn) para ejecutar un programa distribuido en los módulos (MO, ..., Mn) con control preciso del instante de ejecución de cada instrucción y para sincronizar la ejecución de todos o parte del conjunto de módulos (M0, ..., Mn). El procesador determinista en tiempo (100) se comunica con un bus de señales (120) común a todos los módulos (M0, ..., Mn), que comprende un bus de control (120a) donde comparten las señales de reloj y disparo. Opcionalmente, el bus de señales (120) incluye un bus de comunicación (120b) con el que se comunica el procesador determinista en tiempo (100) y también los módulos (M0, ..., Mn) entre sí y, opcionalmente, con un procesador u ordenador externo (130).

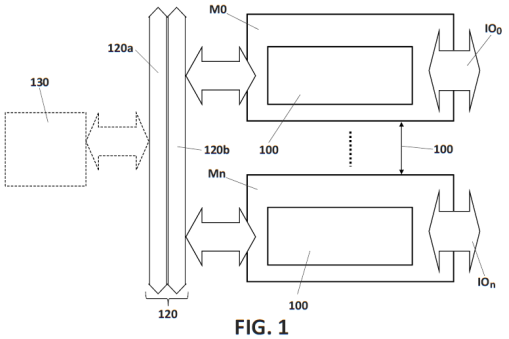

FIG. 1

ES 2 552 739 A1

DESCRIPCIÓN

Sistema de ejecución determinista en tiempo, distribuida y sincronizada para aplicaciones de control, test y medida.

OBJETO DE LA INVENCIÓN

La presente invención pertenece al campo técnico de la electrónica y, en concreto, se aplica al área industrial de los sistemas programables para ejecutar funciones de control, test y/o medida.

Más particularmente, la presente invención se refiere a un sistema para la ejecución determinista en tiempo (i.e., controlando con precisión el tiempo de ejecución de cada acción, de forma totalmente repetible y sin aleatoriedad alguna), distribuida y sincronizada de aplicaciones de control, test o medida, permitiendo que múltiples instrumentos de control, test y medida puedan realizar una tarea ejecutando cada uno de ellos parte de las acciones con total sincronización temporal.

ANTECEDENTES DE LA INVENCIÓN

Los requerimientos de los sistemas de control, test y medida han crecido enormemente en complejidad en los últimos años, requieren cada vez mayor número de entradas/salidas, y mayor capacidad y velocidad de procesado y sincronización.

En los sistemas de control, test y medida es muy importante la ejecución de múltiples acciones simultáneas y/o con tiempos muy precisos. La gestión de forma sincronizada y en tiempo real de acciones y de las múltiples entradas/salidas de un sistema de control, test o medida, es un desafío que presenta numerosos inconvenientes. Y estos problemas crecen significativamente en complejidad con el incremento en el número de canales y la velocidad de los sistemas.

Para atacar este problema, los sistemas de test y medida actuales utilizan diferentes arquitecturas, como se explica a continuación.

Algunos sistemas se implementan con un controlador principal centralizado que supervisa y controla todas las acciones. Este controlador realiza los cálculos y dispara las acciones de medición y control sobre las entradas y salidas de los periféricos, sensores y actuadores. El controlador principal puede ser, entre otros ejemplos, un ordenador, un microcontrolador o una FPGA (del inglés, Field Programmable Gate Array), según los requerimientos de robustez, tiempo real y velocidad. Al ser un controlador centralizado, todas las tareas deben realizarse y sincronizarse desde un único dispositivo, lo cual limita significativamente la velocidad y capacidades del sistema. Esta limitación crece significativamente con el número de entradas y salidas del sistema y con la complejidad de las tareas que se deben realizar. Todo esto resulta en un rendimiento (‘performance’, en inglés) y una precisión en tiempo limitados. Además la escalabilidad según las necesidades está también muy restringida, por lo que, en general, los sistemas de procesamiento centralizado requieren ser diseñados a medida para cada aplicación.

La tendencia actual es por ello la de utilizar arquitecturas modulares en los sistemas de control, test y medida. De esta forma se consigue la máxima flexibilidad en términos de escalabilidad, posibilidad de utilización de productos estándar de diferentes proveedores, facilidad de reparación y actualización, etc. Algunos de estos sistemas, los más modernos, además incorporan las últimas tecnologías en procesado digital, como FPGA y procesadores embebidos, que permiten tener una gran capacidad de procesado en cada módulo. El gran inconveniente de estos sistemas es la capacidad de sincronizar con precisión todas las

acciones de los diferentes módulos y los cálculos realizados de forma distribuida para implementar la solución completa de test, medida y control. En general para solventar la sincronización se utiliza un entramado de señales de disparo ('trigger', en inglés) entre los diferentes módulos que, según las capacidades de cada módulo, busca disparar secuencialmente, de forma sincronizada y con precisión, las distintas acciones de medición, test y control. En muchos casos el tiempo de propagación de estas señales presenta un inconveniente importante que limita la precisión del sistema y además la complejidad crece significativamente con la cantidad de módulos involucrados. La implementación de estos sistemas con técnicas de disparo (triggers) suele ser muy compleja y requiere mucho trabajo de desarrollo y puesta en marcha. Además como el número de señales de disparo es limitado, las posibilidades y flexibilidad también lo son. En algunas plataformas, como PXI Express, el sistema prevé incluir un módulo especial centralizado de generación de señales de disparo que puede disparar acciones en los demás módulos. Este enfoque permite resolver algunos problemas de propagación de los triggers y mejorar la precisión, pero sigue presentando grandes limitaciones a la hora de sincronizar múltiples módulos con requerimientos de señales de disparo cruzadas. Además, el enfoque de disparos centralizados no permite o es muy deficiente para integrar en las acciones de disparo los resultados de mediciones y cálculos distribuidos en varios módulos.

El problema técnico objetivo que se presenta es pues proporcionar medios para el procesamiento multi-módulo sincronizado, en tiempo real y determinista en tiempo de la pluralidad de entradas/salidas de cualquier sistema de control, test y medida.

DESCRIPCIÓN DE LA INVENCIÓN

La presente invención sirve para solucionar el problema mencionado anteriormente, resolviendo los inconvenientes que presentan las soluciones comentadas en el estado de la técnica, proporcionando un sistema de ejecución distribuida determinista en tiempo para aplicaciones de control, test y medida, que permite ejecutar por hardware secuencias y programas con total precisión temporal y sincronización entre todos los módulos que componen el sistema.

La presente invención está basada en un Procesador Determinista en Tiempo (TDP: Time Deterministic Processor, en inglés), que puede formar parte del hardware de cualquier equipo/módulo de control, test y medida. Según posibles realizaciones de la invención, el procesador TDP puede estar implementado sobre un microprocesador, un dispositivo FPGA (del inglés. Field Programmable Gate Array: Campo de Matriz de Puertas Programables) o un dispositivo ASIC (del inglés, Aplication Specific Integrate Circuit: Circuito Integrado para Aplicaciones Específicas), siendo FPGA o ASIC las plataformas más convenientes por dar mayor rendimiento. El Procesador Determinista en Tiempo (TDP) permite ejecutar instrucciones por hardware indicando con absoluta precisión el tiempo de ejecución de cada instrucción y todos los módulos del sistema de control, test y medida, sin importar su función, por ejemplo, los módulos de generación de señal, de digitalización, de comunicaciones, entradas y salidas digitales, etc., todos pueden incorporar adaptaciones del mismo procesador determinista en tiempo, TDP. Este TDP permite realizar un procesado distribuido y ejecutar acciones globales sincronizadas, como por ejemplo saltos condicionales sincronizados, en todos los módulos del sistema según los cálculos, mediciones o información de cualquiera de los módulos. La propagación de las decisiones y la ejecución de las acciones globales son realizadas con completa sincronización en todos los módulos y de forma automática y transparente para el usuario.

Un aspecto de la invención se refiere a un sistema de ejecución distribuida que comprende un conjunto de dos o más módulos para ejecutar aplicaciones de control, test y medida, compartiendo al menos una señal de disparo y una señal de reloj, donde todos los módulos de

ese conjunto comprenden un procesador determinista en tiempo (TDP) que usa la señal de reloj común y al menos una de las señales de disparo compartidas por los módulos para sincronizar la ejecución de al menos parte del conjunto de módulos (o del conjunto completo). La invención también puede utilizarse con un único modulo, en cuyo caso no se utilizarían las capacidades de sincronización pero sí todas las demás funcionalidades y ventajas de ejecución determinista en tiempo.

En el sistema de ejecución distribuida propuesto la precisión en tiempos de ejecución y sincronización entre los módulos depende exclusivamente del sesgo (skew, en inglés) y fluctuaciones (jitter, en inglés) de la señal de reloj común a todos los módulos y, en particular, el procesador determinista en tiempo comprende capacidades de calibración para cancelar el sesgo de reloj entre los módulos.

Algunas de las ventajas técnicas que presenta la invención frente a las soluciones del estado de la anterior técnica son:

- La presente invención permite al usuario programación del hardware de los diferentes módulos mediante un entorno gráfico muy intuitivo, por código de texto o por generación de archivos de órdenes (‘scripting’, en inglés).
- Las acciones globales sincronizadas, como ser saltos globales sincronizados, se programan de forma extremadamente sencilla como si todos los módulos fuesen un único dispositivo y la ejecución fuera centralizada.
- En cualquier momento se puede controlar con total precisión temporal las acciones ejecutadas en cualquiera de los módulos.
- La programación de los módulos se puede realizar mediante un software de programación que incorpora funcionalidades de verificación de temporización (‘timming’, en inglés) y validación de sincronización en tiempo real mientras se programa, lo que permiten trabajar con múltiples módulos de manera simultánea y muy sencilla.
- La invención presenta todas las ventajas de arquitecturas modulares y proporciona un mecanismo de ejecución distribuida y sincronizada totalmente escalable que proporciona absoluta precisión temporal y es muy simple de utilizar y poner en marcha sin importar el número de módulos en el sistema.
- La invención logra una velocidad de procesado y sincronización única, requiriendo para ello una sola señal de Reloj y una única señal de disparo (trigger) multi-punto, ambas comunes a todos los módulos, sin importar el número de módulos en el sistema. Esto simplifica significativamente los requerimientos de interconexiones entre los módulos con respecto a los sistemas actuales que generalmente requieren de numerosos triggers, y hasta de módulos específicos cuya única función es generar los triggers para sincronizar el resto de los módulos.
- En la invención, el Sesgo (`skew’, en inglés) de sincronización entre los módulos depende solo del sesgo en la señal de Reloj (‘clock skew’) común a todos los módulos, el cual suele ser muy bajo, y es independiente del sesgo en las señales de disparo, el cual suele tener valores muy superiores a los del reloj y es muy difícil de reducir en caso de triggers multi-punto.
- La invención incorpora además la posibilidad de utilizar múltiples hilos de ejecución (‘threads’, en inglés) deterministas en tiempo en un mismo modulo que, mediante una técnica de entrelazado instrucción por instrucción, garantizan una total exactitud de los tiempos, cosa que en procesadores convencionales es imposible.
- El sistema además proporciona herramientas para depurar el software (‘debugging’, en inglés) que simplifican significativamente la puesta en marcha de sistemas de control, test y medida multi-módulo. Por ejemplo, cuenta con la posibilidad de introducir puntos de ruptura (‘breakpoints’, en inglés) globales que detienen la

ejecución en un instante preciso de tiempo en todos los módulos del sistema de forma sincronizada.

- En el sistema global pueden convivir módulos con TDP y módulos que no incluyen la tecnología descrita en la invención, porque para ello se cuenta con instrucciones especiales para generar y esperar las señales de disparo (triggers) y así interactuar con estos otros módulos, al igual que con eventos externos.

BREVE DESCRIPCIÓN DE LAS FIGURAS

A continuación se pasa a describir de manera muy breve una serie de dibujos que ayudan a comprender mejor la invención y que se relacionan expresamente con una realización de dicha invención que se presenta como un ejemplo no limitativo de ésta.

FIGURA 1.- Muestra un diagrama de bloques de la arquitectura del sistema de procesado distribuido de módulos de control, test y medida, según una realización preferente de la invención.

FIGURA 2.- Muestra un diagrama de bloques de la arquitectura de un módulo del sistema de procesado distribuido con un Procesador Determinista en Tiempo, según una posible realización de la invención.

FIGURA 3.- Muestra un diagrama de bloques de la arquitectura del Procesador Determinista en Tiempo del sistema, según una posible realización de la invención.

FIGURAS 4.- Muestra un esquema de la estructura de las instrucciones utilizadas por el Procesador Determinista en Tiempo del sistema, según una posible realización de la invención.

FIGURAS 5.- Muestra un esquema del proceso de programación, compilación y ejecución de las instrucciones en el hardware de los módulos del sistema, según una posible realización de la invención.

FIGURA 6A.- Muestra una interfaz gráfica de la herramienta software de programación multi-módulo sincronizada para el sistema, con múltiples ventanas de diagrama de flujo de diferentes módulos del sistema, según una posible realización de la invención.

FIGURAS 6B, 6C y 6D.- Muestran una ventana de diagrama de flujo, respectivamente para cada módulo, de la herramienta gráfica de programación multi-módulo de la Figura 6A, según una posible realización de la invención.

REALIZACIÓN PREFERENTE DE LA INVENCIÓN

A continuación, se describen posibles modos de realización del sistema de ejecución distribuida propuesto para sistemas de control, test y medida.

La Figura 1 muestra un diagrama esquemático de la arquitectura de bloques del sistema modular para la ejecución distribuida en aplicaciones de control, test y medida. El sistema comprende múltiples módulos (M0, ..., Mn) que, a su vez, comprenden una pluralidad de entradas/salidas (IO_0, ..., IO_n) o puertos I/O, del inglés Input/Output. Los módulos (M0, ..., Mn) del sistema realizan funciones de control, test y medida. Cada módulo (M0, ..., Mn) se conecta a un bus de señales (120) común a todos los módulos (M0, ..., Mn), que comprende un bus de control (120a) a través del que todos los módulos (M0, ..., Mn) comparten una o más señales de reloj y una o más señales de disparo o triggers. Además el bus de señales (120) puede incluir uno o más buses de comunicaciones (120b) por medio de los que cada módulo (M0, ..., Mn) se comunica con el resto y, opcionalmente, con un procesador externo (130), que puede

ser un PC, un controlador embebido, etc. Adicionalmente, cada módulo (M0, ..., Mn) puede disponer de uno o más puertos de programación y depuración de programación (110). Todos y cada uno de los módulos (M0, ..., Mn) tienen como núcleo un procesador TDP o Procesador Determinista en Tiempo (100). El TDP se comunica con el bus de señales (120) y el resto del módulo también independiente del TDP.

En la Figura 2 se muestra más particularmente el diagrama de bloques de la arquitectura interna de uno de esos módulos (M0, ..., Mn) del sistema de la Figura 1. El módulo (Mi), i = 0,..., n, se implementa en un dispositivo cuyo conjunto de componentes o hardware (200) comprende en el núcleo el Procesador Determinista en Tiempo (100) que, a su vez, puede estar implementado en un dispositivo (201) de procesamiento: dispositivo FPGA, microprocesador, ASIC, etc. El dispositivo (201) se comunica a través de los puertos de entrada/salida (IOi) y del bus de señales (120), con al menos una señal de reloj y al menos una señal de disparo comunes para todos los módulos (M0, ..., Mn) del sistema, necesitándose al menos un trigger o señal de disparo para poder efectuar la sincronización entre módulos.

La Figura 3 muestra en un diagrama de bloques la arquitectura del Procesador Determinista en Tiempo (100) que está implementado con un core o núcleo (300) común a todos los módulos, lo cual garantiza el correcto funcionamiento de la sincronización independientemente del tipo de instrumento de control, test y medida, al que se incorpora, y uno o más bloques específicos (P_0, ..., P_n) para la ejecución de funciones adicionales incluyendo las diferentes funciones específicas que caracterizan a cada tipo de módulo de test y medida. Esta arquitectura permite incorporar fácilmente nuevas instrucciones para cubrir nuevos requerimientos. El núcleo (300) del procesador TDP (100) se comunica con el resto de componentes (301): hardware, memoria de datos, puertos de entrada/salida, etc., del módulo y con el bus de señales (120) común a todos los módulos. Además, el núcleo (300) del Procesador Determinista en Tiempo (100) lee, de una memoria (302) de programa, palabras de instrucción (303) usadas durante la ejecución del programa en el hardware. El núcleo (300) del TDP (100) procesa las palabras de instrucciones (303) en el momento preciso, ejecuta las instrucciones correspondientes al núcleo y además las envía a unos ejecutadores de instrucciones (P_0, ..., P_n), donde se interpretan y ejecutan instrucciones adicionales. Cada ejecutador –engine, en inglés- de instrucciones (P_0, ..., P_n) puede ejecutar una o más tipos de instrucciones (303) y puede haber uno o más ejecutadores de instrucciones (P_0, ..., P_n); su número y tipo de instrucciones que ejecutan depende de cada instrumento especifico.

La Figura 4 muestra la estructura de las instrucciones que maneja el Procesador Determinista en Tiempo (100). La palabra de instrucción del procesador TDP (100) tiene una estructura especial para incluir al menos en un campo (DW[1], DW[2]) el tiempo exacto de ejecución de cada palabra de instrucción, que puede ser tiempo absoluto, o relativo a la instrucción anterior. Además contiene un campo de control (DW[0]) que permite habilitar la ejecución de una o más instrucciones que se pueden colocar de forma flexible en la palabra de instrucción. La palabra de instrucción tiene una longitud variable según las necesidades del módulo en cuestión, cuanto más grande la palabra más instrucciones simultáneas se pueden ejecutar en cada módulo lo cual es una ventaja muy importante en sistemas de control, test y medida.

El procesador TDP (100) cuenta con instrucciones especiales de sincronización y saltos condicionales sincronizados, que permiten propagar una decisión a todos los módulos (M0, ..., Mn) del sistema y que todos ejecuten un salto condicional u otra acción según esta decisión con total sincronización temporal. Esto es una característica única del TDP (100). Para lograr esta total sincronización el TDP (100) requiere de una sola señal de reloj y una única señal de trigger o disparo, ambas señales comunes a todos los módulos (M0, ..., Mn). Una señal de trigger consiste en una conexión digital multi-punto que conecta todos los módulos (M0, ..., Mn) entre sí para formar lo que se conoce como una “OR cableada” o una “AND cableada” según el estándar de interfaz lógica –drivers, en inglés- que se utilice. En caso de que el sistema cuente

con más de una de estas señales de trigger, como es el caso de la plataforma PXI Express que cuenta con ocho señales de trigger, el TDP (100) puede opcionalmente hacer uso de los trigger adicionales para incrementar la velocidad de sincronización y de repetición de decisiones y acciones sincronizadas. Además el procesador TDP (100) puede integrar en la transferencia de las decisiones los buses de comunicación (120b) disponibles en los módulos (M0, ..., Mn). Por ejemplo, si se usa una plataforma de control, test y medida PXI o PXI Express, el bus de comunicaciones (120b) usado por el procesador determinista en tiempo (100) es respectivamente un bus PCI -del inglés, Peripheral Component Interconnect: Interconexión de Componentes Periféricos- o un bus PCI Express.

Por ejemplo, una posible realización de la invención se implementa sobre la plataforma PXI o PXI Express que son un estándar de referencia en aplicaciones de control, test y medida en la mayoría de las industrias. Un chasis PXI Express cuenta con una placa de circuito impreso –backplane, en inglés- que distribuye entre todos los módulos las siguientes señales: alimentación, un bus PCI Express, un reloj de 100MHz, CLK100, un reloj de 10MHz,CLK10, una señal de sincronización referida al reloj de 100MHz y que provee un pulso de referencia a 10MHz,SYNC100, y 8 señales de disparo multi-punto, PXI Trigger Bus, que conectan todos los módulos en forma de “AND cableada” con drivers lógicos del tipo “colector abierto”, entre otras. Los módulos cuentan con una FPGA que se comunica con un PC y con el hardware específico que compone cada tipo específico de instrumento de control, test y medida. Para la comunicación entre el PC y la FPGA se utiliza el bus PCI Express; se puede utilizar algún tipo de conmutador/controlador de dispositivos -switch/driver, en inglés-, o conectarlo directamente a la FPGA. La FPGA controla la comunicación PCI Express y otros componentes del hardware. Dentro de la FPGA, se implementa el TDP (100) que funciona con el reloj de 100MHz, CLK100, para controlar con total precisión el instante de ejecución de cada instrucción y que utiliza el bus de 8 PXI triggers y la señal de referencia SYNC100, para lograr una perfecta ejecución sincronizada entre todos los módulos. El sesgo o skew en la sincronización entre módulos depende solo del sesgo del reloj de 100MHz en las diferentes ranuras del chasis y es independiente del sesgo en las señales de disparo y de referencia SYNC100. El sesgo del reloj depende sólo del chasis utilizado y está garantizado que es menor a 250ps, aunque según el chasis puede ser muy inferior y además puede mejorarse significativamente si se realiza una calibración inicial. El usuario puede seleccionar cuáles de los 8 disparo disponibles en el PXI Express quiere permitir utilizar al TDP (100) y el software gestiona los recursos asignados. En el caso de PXI no se cuenta con las señales CLK100 y SYNC100, por lo que la realización se implementa utilizando el reloj de 10MHz, CLK10, y las 8 señales de disparo multi-punto. En PXI el bus de comunicaciones utilizado es PCI en lugar de PCI Express.

Otra posible realización de la invención se implementa sobre una plataforma LXI -del inglés, LAN Extension for Instrumentation: Extensión de Red de Área Local para Instrumentación-, donde la señal de reloj y las de disparo del bus (120a) se comunican entre los módulos utilizando el bus de disparo cableado –Wired Trigger Bus, en inglés- previsto en la especificación LXI y donde el bus de comunicaciones (120b) consiste en una conexión LAN -del inglés Local Area Network: Red de Área Local-.

Otra posible realización de la invención se implementa sobre la plataforma microTCA -del inglés micro Advanced Telecomunications Architecture: micro arquitectura de computación de telecomunicaciones avanzadas- o algunas de sus variantes, como microTCA.4, donde la señal de reloj se comunica entre los módulos utilizando algunos de los varios puertos de reloj contemplados en el estándar, CLK1, CLK2, CLK3, TCLKC o TCLKD, y las señales de disparo puede comunicarse a través de los puertos 17 a 20. Donde el bus de comunicaciones (120b) se puede seleccionar entre Ethernet, PCI Express o SRIO -del inglés, Serial Rapid Input Output: Entradas Salidas en Serie Rápidas, según la implementación específica del sistema.

Otra posible realización de la invención se implementa sobre la plataforma AXIe -del inglés, AdvancedTCA Extensions for Instrumentation: Extensión de AdvancedTCA para Instrumentación- donde, de forma similar a PXI Express, se utilizan las señales del backplane CLK100 y SYNC, y una o varias de las 12 señales de disparo MLVDS disponibles y donde el bus de comunicaciones (120b) se puede seleccionar entre una conexión LAN o PCI Express.

La decisión sobre la que se ejecuta la acción global o salto condicional y la sincronización puede depender de uno o más módulos (M0, ..., Mn) del sistema. Para forzar la sincronización de los módulos (M0, ..., Mn), es decir, incluir un punto de sincronización incondicional, se utiliza el mismo mecanismo que en los saltos condicionales sincronizados considerando que la condición del salto es siempre falsa. Además, el TDP (100) dispone de instrucciones de ejecución sincronizada especiales, como el nodo “Start”, que permite arrancar la ejecución en todos los módulos (M0, ..., Mn) simultáneamente. Las acciones sincronizadas, los saltos sincronizados y puntos de sincronización pueden involucrar a todos los módulos (M0, ..., Mn) del sistema o sólo a un subconjunto de ellos.

El TDP (100) cuenta además con instrucciones de control de flujo como *while*, *if*, *for* etc, tanto locales como multi-modulo sincronizadas. Además, cuenta con operaciones matemáticas, lógicas, y de acceso a datos y variables del propio módulo o del resto de los módulos. Y cuenta con todas las instrucciones necesarias para controlar las capacidades específicas del módulo (Mi), i = 0,....,n, sobre el que se implementa. El TDP (100) cuenta también con instrucciones de generación y espera de triggers para interactuar con módulos que no incorporen un TDP (100) y con eventos externos.

Existen dos tipos básicos de instrucciones que puede ejecutar el TDP (100):

- Instrucciones que se disparan y no consumen tiempo del TDP (100), es decir estas instrucciones se inicia su ejecución y en el ciclo de reloj siguiente el TDP (100) puede procesar una nueva palabra de instrucción y ejecutar nuevas instrucciones. Estas instrucciones pueden requerir varios ciclos para su ejecución y se ejecutan en paralelo independientemente del funcionamiento del TDP (100).
- Instrucciones que consumen un tiempo del TDP (100), por lo que hasta que no termina su ejecución el TDP (100) no puede procesar la siguiente palabra de instrucción y ejecutar las siguientes instrucciones; ejemplos típicos son las instrucciones de control de espera, wait, o las de flujo: while, for, if, etc.

Las instrucciones que consumen tiempo se implementan en el núcleo (300) del TDP (100), mientras las que no, pueden implementarse tanto en el núcleo (300) como en los ejecutadores de instrucciones (P_0, …, P_n). En el caso de las instrucciones que se disparan, es posible que se puedan iniciar múltiples ejecuciones secuencialmente cuando la ejecución anterior todavía está en curso y para ellos se implementan utilizando la arquitectura basada en filtros –pipeline, en inglés-.

El TDP (100) cuenta además con un sistema determinista en tiempo de threads o hilos de ejecución, que permite incluir en cada módulo (Mi) más de un hilo de ejecución que se ejecuta en paralelo con una técnica de entrelazado instrucción por instrucción. Esta técnica garantiza una distribución determinística de la ejecución de las instrucciones en todos los threads, garantizando total exactitud de los tiempos. El hilo de ejecución principal se sincroniza de forma nativa con los demás módulos y se incluyen semáforos y eventos que permiten sincronizar los diferentes threads de un mismo módulo entre sí. Además se permite seleccionar si se desea incluir alguna de las instrucciones globales sincronizadas en uno o más de los threads secundarios.

La Figura 5 muestra esquemáticamente el proceso de programación, ejecución y compilación de las instrucciones del TDP (100) de cada módulo (M0, ..., Mn). El usuario programa el sistema completo o parte de él, implementando un código para cada uno de los módulos (M0,

..., Mn) involucrados en la aplicación de control, test y medida. Para la implementación del código se cuenta con las instrucciones locales y globales a todos los módulos (M0, ..., Mn), ya descriptas anteriormente. Para la primera fase del proceso, de introducción del código (500), se pueden usar herramientas de generación de archivos de órdenes (501) o scripts, o bien plataformas de programación basadas en texto (502) o gráficas (503), Las instrucciones globales se ejecutan en todos los módulos de un grupo o en un subconjunto y el TDP (100) garantiza su ejecución completamente sincronizada. No todos los módulos (M0, ..., Mn) del sistema deben trabajar en conjunto, es posible tener grupos de módulos que trabajan en conjunto, los cuales se programan y funcionan completamente sincronizados entre ellos independientemente de los módulos fuera del grupo. El grupo se programa con la capacidad de ejecutar instrucciones en todos ellos de forma sincronizada como si fuesen un único hardware y también puede incluir instrucciones locales específicas en cada módulo. El código fuente escrito por el usuario puede ser gráfico o de texto y se provee de herramientas gráficas o de generación de script, como se ha comentado anteriormente, que permiten fácilmente agregar las instrucciones multi-modulo sincronizadas y las locales específicas a cada módulo.

La herramienta de programación realiza todas las comprobaciones para mantener la coherencia del código global e individual y luego compila (510) el código fuente (511) del grupo en su conjunto para obtener un código compilado (513), a partir del que se va a generar un código de ejecución (CM0, ..., CMn) específico para cada módulo (M0, ..., Mn). En la compilación (510) se convierten las instrucciones de alto nivel del usuario en las instrucciones que puede ejecutar el TDP (100), cuyo compilador (512) administra los recursos de trigger y comunicación disponibles para lograr de forma automática y transparente para el usuario la sincronización e intercambio de decisiones entre los módulos (M0, ..., Mn). El compilador (512) verifica la sintaxis correcta de todas las instrucciones y que existen recursos necesarios en cada momento para su ejecución. En resumen, el código introducido (500) por el usuario se compila para generar el código ejecutable (CM0, ..., CMn) específico para cada módulo (M0, ..., Mn) y, durante la compilación, el compilador (512) del TDP (100) genera el código necesario para la sincronización de cada una de las instrucciones globales analizando la sincronización en cada tramo. La herramienta de programación y compilación generalmente corre en un ordenador, pero podría correr en cualquier otra plataforma, y además la herramienta de programación y la herramienta de compilación podrían correr en plataformas diferentes. Los códigos ejecutables (CM0, ..., CMn) generados para cada módulo son descargados a todos los módulos del grupo y luego el código es ejecutado por el TDP (100) de manera sincronizada en el hardware de cada módulo del sistema (530). La descarga del código ejecutable (CM0, ..., CMn) a cada módulo (M0, ..., Mn) y el control de ejecución y depuración (520), usando herramientas gráficas (521), de comandos (522) y/o de librería (523), puede hacerse por el mismo bus de comunicación (120b) que comunica todos los módulos (M0, ..., Mn) entre sí y con el PC o controlador del sistema en caso de estar disponible, o bien, mediante un puerto de programación específico disponible en cada módulo.

Para la programación del sistema hardware (530) se cuenta con un software gráfico, como se muestra en la Figura 6A que permite programar de forma sencilla cada módulo mediante uno a más diagramas de flujo, como los que se muestran en detalle a modo de ejemplo en las Figura 6B-6D.

El software permite seleccionar con qué grupo de módulos se quiere trabajar de forma sincronizada y crear un proyecto. Una vez creado el proyecto se cuenta con una ventana de diagrama de flujo por módulo, una ventana de diagrama de flujo adicional por cada thread y ventanas adicionales de edición de instrucciones y configuración, depuración, etc. Por ejemplo, en la Figura 6A se muestra una ventana de propiedades (640) , donde se configuran algunos parámetros generales de la programación de la ejecución del elemento seleccionado en cada momento y particulares de las instrucciones del elemento, junto con tres ventanas de diagrama de flujo: una ventana de diagrama de flujo (600) para un primer módulo, mostrada en detalle

como ejemplo en la Figura 6B; otra ventana (601) para un segundo módulo, mostrada en detalle como ejemplo en la Figura 6C; y una tercera (602) para un tercer módulo, mostrada en detalle como ejemplo en la Figura 6D. El software permite programar todos los módulos simultáneamente y gestiona automáticamente los elementos globales sincronizados. Además, el software permite guardar los proyectos para luego cargarlos con el mismo software gráfico, con las librerías de usuario o con otro entorno de edición de código. Las librerías de usuario permiten cargar proyectos ya creados, modificarlos, compilarlos, cargarlos en el hardware y controlar su ejecución completamente. De esta forma, la ejecución del código hardware puede integrarse en cualquier aplicación del usuario. Desde las librerías de usuario puede, iniciarse, pausar, parar la ejecución, así como intercambiar datos y eventos entre el PC y los módulos. El usuario puede tener una colección o librería de programas hardware que utilicen los mismos o diferentes módulos y ejecutarlos en distintos momentos de la ejecución del software.

Además, el software permite descargar el programa a cada módulo del grupo, ejecutar y depurar el programa, inspeccionando las variables, estado del TDP (100) e insertar puntos de ruptura o breakpoints, locales y globales. Los puntos de ruptura locales detienen sólo la ejecución de un módulo, mientras que los globales detienen a todos los módulos del grupo. Los puntos de ruptura globales son una herramienta muy importante en sistemas sincronizados multi-modulo. Estos se introducen en uno de los módulos y se replican al final del mismo tramo sincronizado en el resto de los módulos del grupo. El usuario puede mover los puntos de ruptura globales libremente dentro de cada tramo, para ajustar el punto exacto de detención de cada módulo.

El diagrama de flujo de cada módulo está compuesto por cajas de instrucciones y flechas de tiempo como se muestran en las Figuras 6A-6D. Las flechas interconectan un elemento origen con un elemento destino en el diagrama de flujo y corresponden a un instante de ejecución determinado del elemento de destino respecto al elemento de origen. Las flechas de tiempo indican el tiempo exacto, indicado en nanosegundos (ns) en las Figuras 6B-6D, en el que se iniciará la ejecución de la siguiente instrucción respecto de la anterior. Las cajas de instrucción pueden contener acciones, cálculos, instrucciones de control de flujo, de transferencia de datos, etc. Las instrucciones pueden ser locales al módulo o globales multi-modulo sincronizadas. Los elementos sincronizados (610, 611, 612, 613) se insertan en uno de los módulos y son replicados automáticamente en el resto de módulos del grupo. El software genera el código necesario para mantener el sincronismo y propagar las decisiones de manera transparente para el usuario garantizando la ejecución de todas las cajas sincronizadas en el mismo instante de tiempo. Las cajas o elementos locales (620, 621, 622, 623, 624, 625, 626, 627, 628) de cada módulo pueden mantener el sincronismo o no.

Hay casos en los que elementos locales (620, 621, 622, 623, 624, 625, 626, 627, 628) del conjunto de módulos (M0, ..., Mn), correspondientes a palabras de instrucción (303) locales, al ejecutarse no mantienen el sincronismo entre los módulos, debido a que la palabra de instrucción (303) local consume un tiempo no conocido por el compilador (512) antes de la ejecución. En tal caso la sincronización entre todos los módulos se recupera en el elemento global sincronizado siguiente a la palabra de instrucción (303) local que provoca la desincronización.

Las cajas o estructuras de control locales que mantienen el sincronismo son aquellas que disparan acciones y no consumen tiempo, o las que consumen tiempo pero éste es conocido durante la compilación antes de la ejecución. Las cajas o estructuras de control locales que desincronizan son aquellas cuyo tiempo de ejecución no puede saberse en tiempo de compilación, por ejemplo un comando While (621) con una variable, o un comando Wait de un evento externo. El software garantiza la ejecución sincronizada de las cajas correspondientes a elementos sincronizados (610, 611, 612, 613), eso permite que cada tramo entre elementos globales sincronizados pueda analizarse por separado. Si todas las cajas locales en un tramo

mantienen el sincronismo, el software puede determinar con total precisión el tiempo de ejecución de cada instrucción local con respecto al nodo sincronizado de inicio del tramo y así ajustar automáticamente el tiempo de la última flecha antes de la caja de sincronización de final del tramo y lograr así que todos los módulos ejecuten el elemento sincronizado de final del tramo al mismo tiempo. Si alguna de las cajas locales desincroniza, el software no podrá determinar con precisión el tiempo de las cajas a partir del punto de desincronización, pero se recuperará el sincronismo entre todos los módulos en la caja sincronizada del final del tramo en cuestión: esto se realiza con un procedimiento adicional y se indica con la línea punteada al final del tramo (630). Así, por ejemplo, en la Figura 6A, en la ventana de propiedades (640), la propiedad “Time” de la caja local seleccionada (624) indica que la caja se ejecutará como mínimo 140ns después del punto de sincronización inmediatamente superior, pero puede tardar en ejecutarse de forma indefinida según cuantos ciclos se repita la instrucción local While (621) que se ejecuta antes de la caja seleccionada (624).

El software gestiona la sincronización multi-módulo de forma transparente al usuario mientras se realiza la programación, de esta forma las cajas sincronizadas se visualizan alineadas entre todos los módulos y los tiempos se ajustan en tiempo real. Esto hace que la programación multi-modulo sea muy simple, como si se programase un único dispositivo.

A la vista de esta descripción y figuras, el experto en la materia podrá entender que la invención ha sido descrita según algunas realizaciones preferentes de la misma, pero que múltiples variaciones pueden ser introducidas en dichas realizaciones preferentes o aplicarse a diferentes plataformas estándar o personalizadas, sin salir del objeto de la invención tal y como ha sido reivindicada.

REIVINDICACIONES

1. Un sistema de ejecución determinista en tiempo distribuida sincronizada para aplicaciones de control, test y medida, que comprende un conjunto de al menos un módulo (M0, ..., Mn) para ejecutar aplicaciones de control, test y medida, **caracterizado por que** todos los módulos (M0, ..., Mn) del conjunto comparten al menos una señal de disparo y una señal de reloj, y comprenden un procesador determinista en tiempo (100) , donde el procesador determinista en tiempo (100) usa la señal de reloj y una o más de las señales de disparo compartidas por todos los módulos (M0, ..., Mn) para ejecutar un programa distribuido en los módulos (M0, …, Mn) controlando cada instante de ejecución de cada instrucción del programa y, si el conjunto comprende más de un módulo (M0, ..., Mn), para sincronizar la ejecución en todos o parte del conjunto de módulos (M0, ..., Mn).

2. El sistema de ejecución distribuida de acuerdo con la reivindicación 1, ***caracterizado por que*** el procesador determinista en tiempo (100) se comunica con un bus de señales (120) común a todos los módulos (M0, ..., Mn), que comprende un bus de control (120a) que lleva la señal de reloj y al menos una señal de disparo, compartidas por todos los módulos (M0, ..., Mn).

3. El sistema de ejecución distribuida de acuerdo con la reivindicación 2, ***caracterizado por que*** el procesador determinista en tiempo (100) se comunica con un bus de comunicación (120b) incorporado en el bus de señales (120) y por medio del que los módulos (M0, ..., Mn) se comunican entre sí.

4. El sistema de ejecución distribuida de acuerdo con la reivindicación 3, ***caracterizado por que*** los módulos (M0, ..., Mn) se comunican con un procesador externo (130) por medio del bus de comunicación (120b).

5. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** el procesador determinista en tiempo (100) tiene un núcleo (300) común a todos los módulos (M0, ..., Mn) y en el núcleo (300) el procesador determinista en tiempo (100) lee instrucciones (303) de una memoria (302), ejecuta en un instante de ejecución determinado las instrucciones (303), que se seleccionan entre instrucciones globales sincronizadas a todos o parte de los módulos (M0, ..., Mn) y locales a cada módulo, y pasa las instrucciones (303) a al menos un ejecutador de instrucciones (P_0, ..., P_n) para ejecutar funciones adicionales, que se seleccionan entre funciones comunes a todos los módulos (M0, ..., Mn) y funciones específicas según una aplicación de control, test y medida de cada módulo (M0, ..., Mn)..

6. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** el procesador determinista en tiempo (100) ejecuta una palabra de instrucción (303) que comprende al menos de un campo (DW[1], DW[2]) indicando un tiempo exacto de ejecución de cada palabra de instrucción (303).

7. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** el procesador determinista en tiempo (100) ejecuta una palabra de instrucción (303) que comprende un campo de control (DW[0]) que habilita la ejecución de una instrucción o de múltiples instrucciones simultáneamente.

8. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** el procesador determinista en tiempo (100) ejecuta instrucciones globales sincronizadas, de sincronización y saltos condicionales sincronizados, propagando una decisión a todos o parte de los módulos (M0, ..., Mn) que, según la decisión, ejecutan una acción o un salto condicional con sincronización temporal.

9. El sistema de ejecución distribuida, de acuerdo con la reivindicación 8, ***caracterizado por que*** la decisión propagada por el procesador determinista en tiempo (100) entre todos los módulos (M0, ..., Mn) del conjunto y sobre la que se ejecuta el salto condicional o acción sincronizada depende de todos, uno o parte del conjunto de módulos (M0, ..., Mn).

10. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** el procesador determinista en tiempo (100) ejecuta instrucciones de generación y espera de al menos una señal de disparo para interactuar con eventos externos y con otros módulos del sistema que no incorporan un procesador determinista en tiempo (100).

11. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** el procesador determinista en tiempo (100) usa hilos de ejecución para incluir en cada módulo (M0, ..., Mn) del conjunto más de un hilo de ejecución que se ejecuta en paralelo con una técnica de entrelazado instrucción por instrucción.

12. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** el procesador determinista en tiempo (100) comprende un compilador (512) que convierte un código fuente (511) con instrucciones de alto nivel, introducidas por un usuario para todos los módulos (M0, ..., Mn) del conjunto como un único dispositivo programable, en un código compilado (513) a partir del que se obtiene un código ejecutable (CM0, ..., CMn) específico para cada módulo (M0, ..., Mn) con palabras de instrucción (303), que contienen instrucciones globales sincronizadas y locales, ejecutables por el procesador determinista en tiempo (100), donde el código compilado (513) tiene en cuenta el tiempo exacto de ejecución de cada palabra de instrucción (303) para la sincronización de todas las palabras de instrucción (303) globales para todos los módulos (M0, ..., Mn) del conjunto en la ejecución del código ejecutable (CM0, ..., CMn) específico de cada módulo (M0, ..., Mn).

13. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** comprende una interfaz de usuario gráfica a través de la que el usuario introduce instrucciones de alto nivel para programar cada módulo (M0, ..., Mn) del conjunto mediante al menos un diagrama de flujo que comprende al menos un elemento sincronizado (610, 611, 612, 613) introducido por el usuario en uno de los módulos del conjunto y el, al menos un, elemento sincronizado (610, 611, 612, 613) es replicado automáticamente en el resto de módulos del conjunto, para ser ejecutado como una palabra de instrucción (303) global en un mismo instante de tiempo por el procesador determinista en tiempo (100) de cada módulo.

14. El sistema de ejecución distribuida, de acuerdo con la reivindicación 13, ***caracterizado por que*** la interfaz de usuario gráfica adicionalmente comprende flechas que interconectan un elemento origen con un elemento destino en el diagrama de flujo y que corresponden a un instante de ejecución determinado del elemento de destino respecto al elemento de origen.

15. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones 13-14, ***caracterizado por que*** la interfaz de usuario gráfica adicionalmente comprende elementos locales (620, 621, 622, 623, 624, 625, 626, 627, 628) a cada módulo (M0, ..., Mn) del conjunto correspondientes a palabras de instrucción (303) locales, cuya ejecución mantiene el sincronismo con las palabras de instrucción (303) globales sólo si la palabras de instrucción (303) local no consumen tiempo o si consumen un tiempo conocido por el compilador (512) antes de la ejecución.

16. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones 13-15, ***caracterizado por que*** la interfaz de usuario comprende puntos de ruptura globales a todos o parte de los módulos (M0, ..., Mn), que corresponden a palabras de instrucción (303) especiales del procesador determinista en tiempo (100) y que detienen la ejecución del procesador determinista en tiempo (100) en un punto concreto dentro del mismo tramo sincronizado en todos o parte del conjunto de módulos (M0, ..., Mn) .

17. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** se implementa sobre una plataforma de control, test y medida que se selecciona entre PXI y PXI Express.

18. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** se implementa sobre una plataforma LXI.

19. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** se implementa sobre una plataforma de control, test y medida que se selecciona entre microTCA, microTCA.4, AXIe o una evolución de AdvancedTCA.

20. El sistema de ejecución distribuida, de acuerdo con cualquiera de las reivindicaciones anteriores, ***caracterizado por que*** el procesador determinista en tiempo (100) está implementado en un dispositivo (201) de procesamiento que se selecciona entre dispositivo microprocesador, FPGA o ASIC.

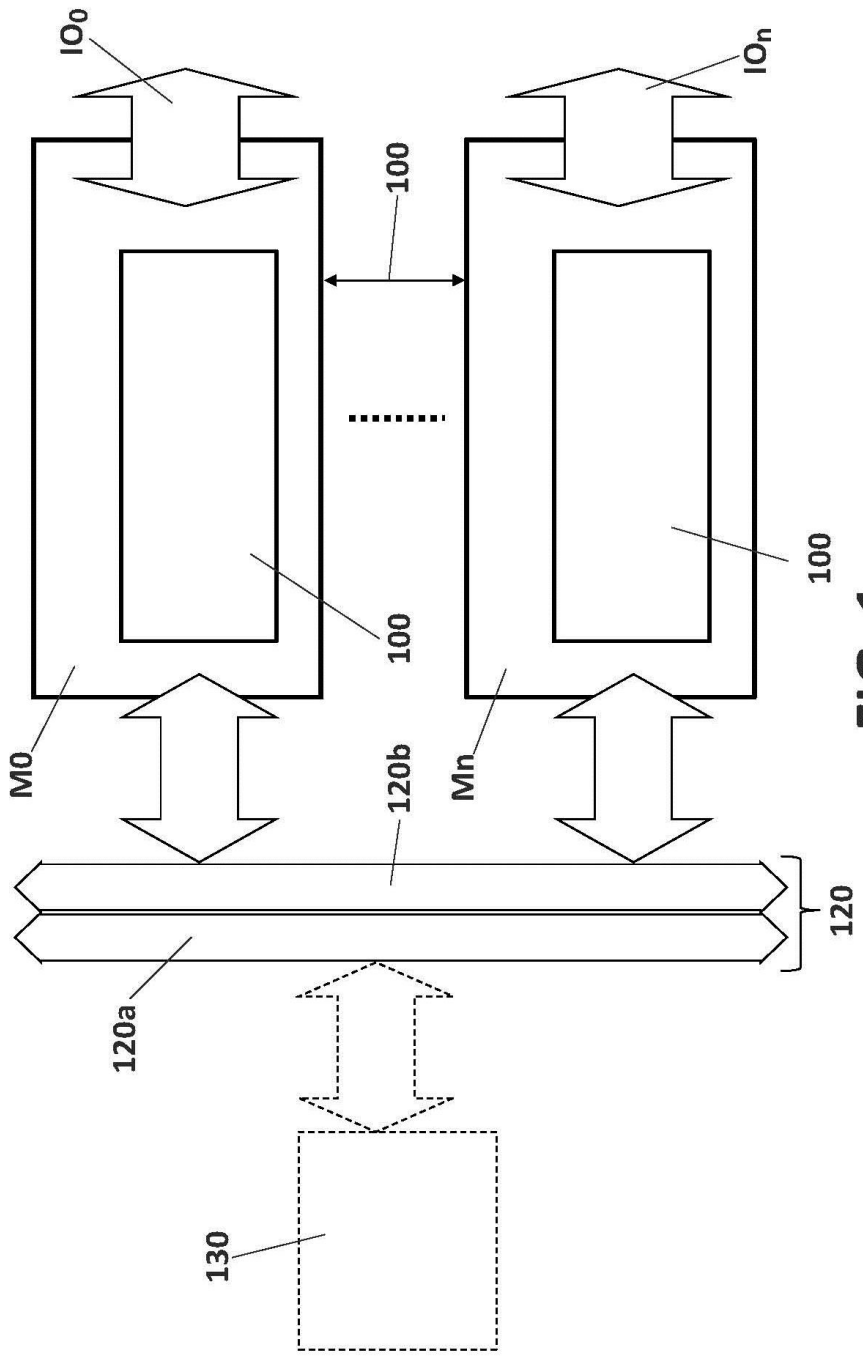

FIG. 1

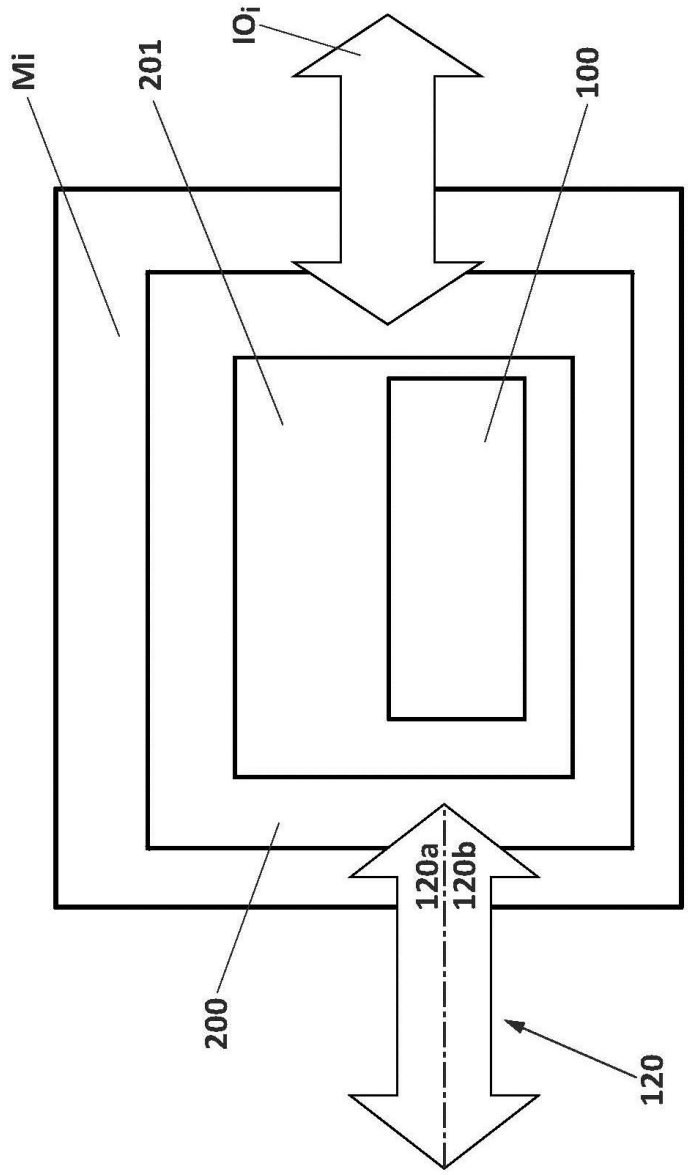

FIG. 2

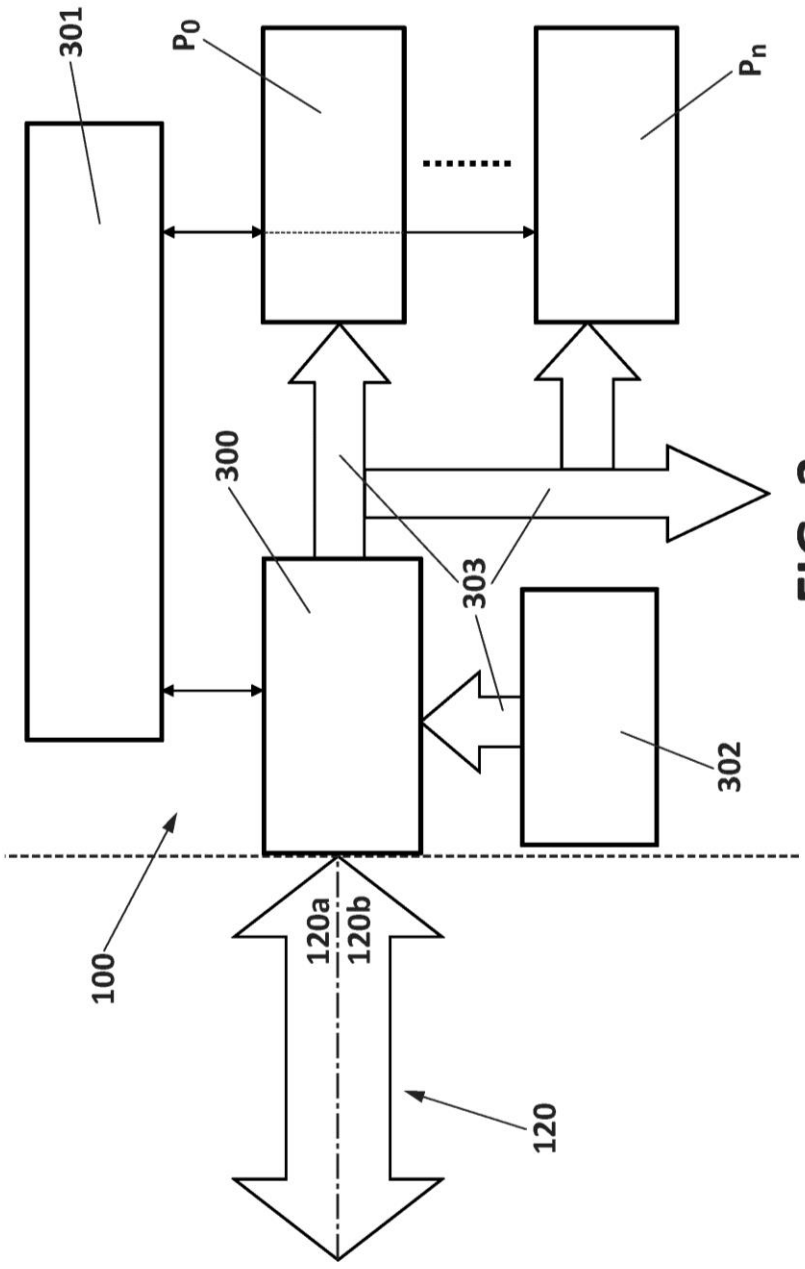

FIG. 3

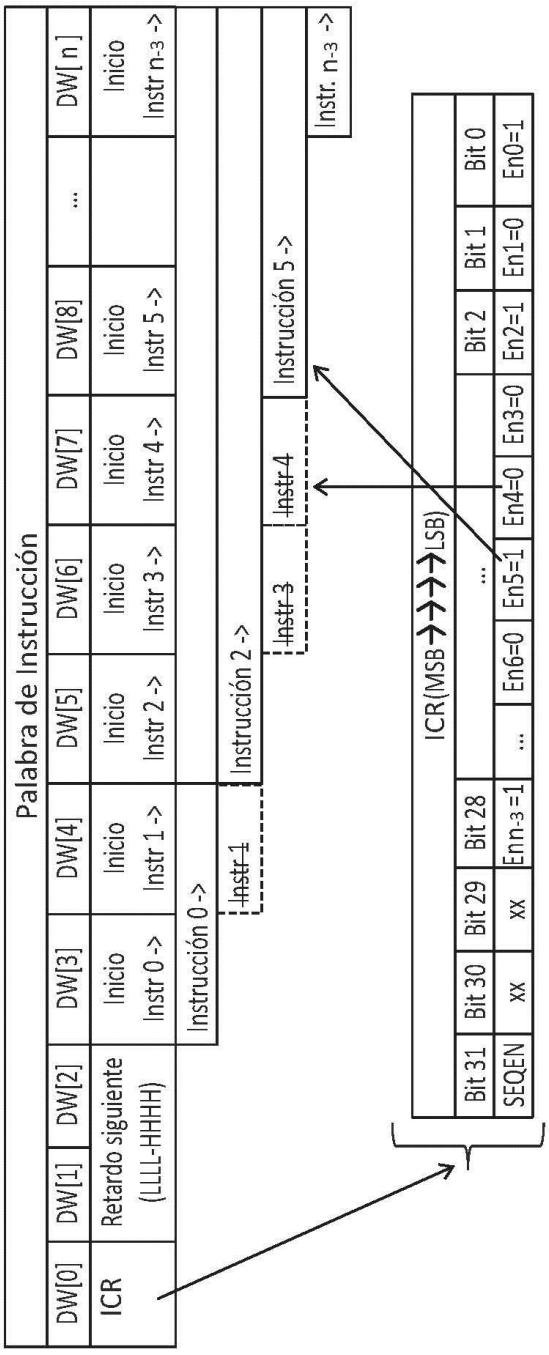
Palabra de Instrucción
DW[0]
DW[1]
DW[2]
DW[3]
DW[4]
DW[5]
DW[6]
DW[7]
DW[8]
...
DW[n]
ICR
Retardo siguiente (LLLL-HHHH)
Inicio Instr 0 ->
Inicio Instr 1 ->
Inicio Instr 2 ->
Inicio Instr 3 ->
Inicio Instr 4 ->
Inicio Instr 5 ->
Inicio Instr n-3 ->
Instrucción 0 ->
Instr 1
Instrucción 2 ->
Instr 3
Instr 4
Instrucción 5 ->
Instr. n-3 ->
ICR(MSB→→→→LSB)
Bit 31
Bit 30
Bit 29
Bit 28
...
Bit 2
Bit 1
Bit 0
SEQEN
xx
xx
Enn-3=1
...
En6=0
En5=1
En4=0
En3=0
En2=1
En1=0
En0=1

FIG. 4

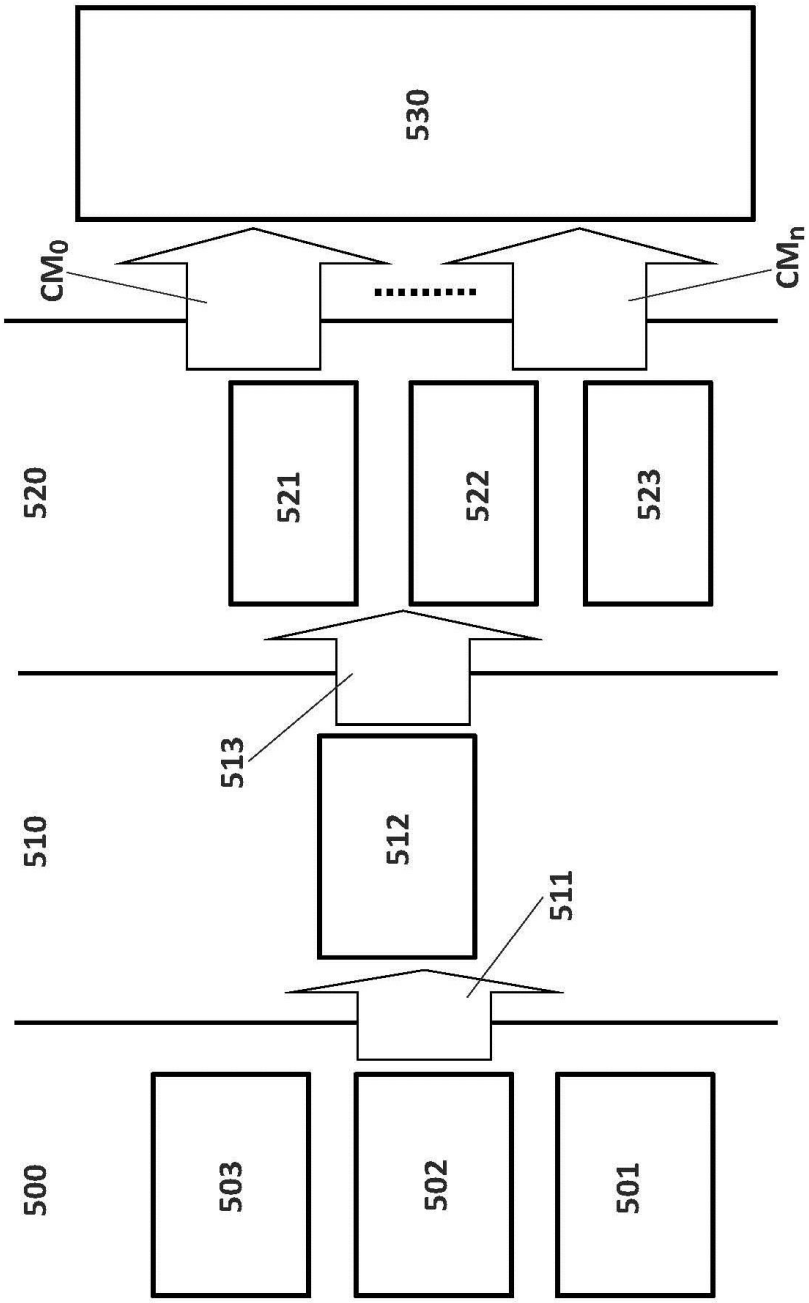

FIG. 5

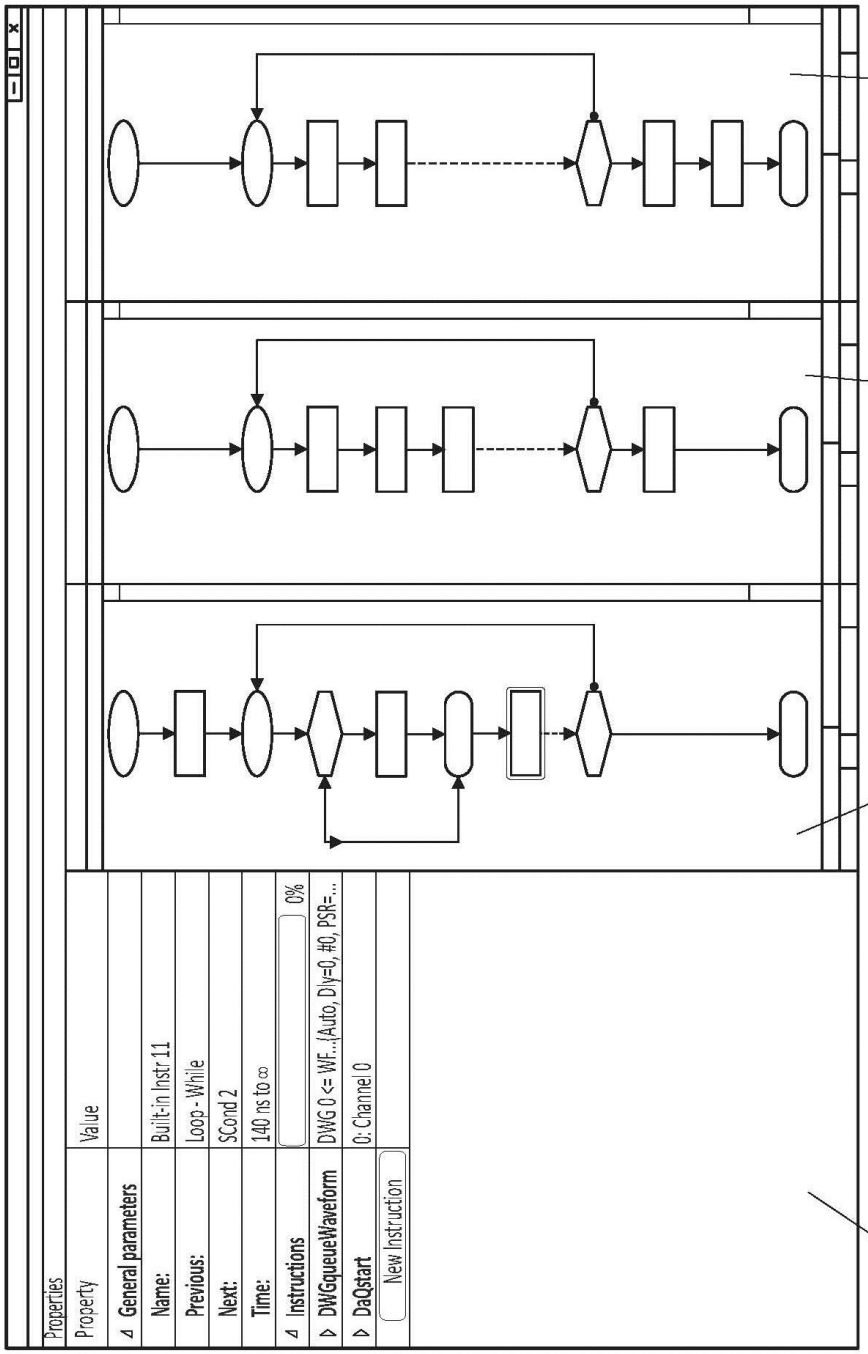
Properties
Property
Value
General parameters
Name:
Built-in Instr 11
Previous:
Loop - While
Next:
SCond 2
Time:
140 ns to ∞
Instructions
0%
DWGqueueWaveform
DWG 0 <= WF.../Auto, Dly=0, #0, PSR=...
DaQstart
0: Channel 0
New Instruction
600
601
602
640

FIG. 6A

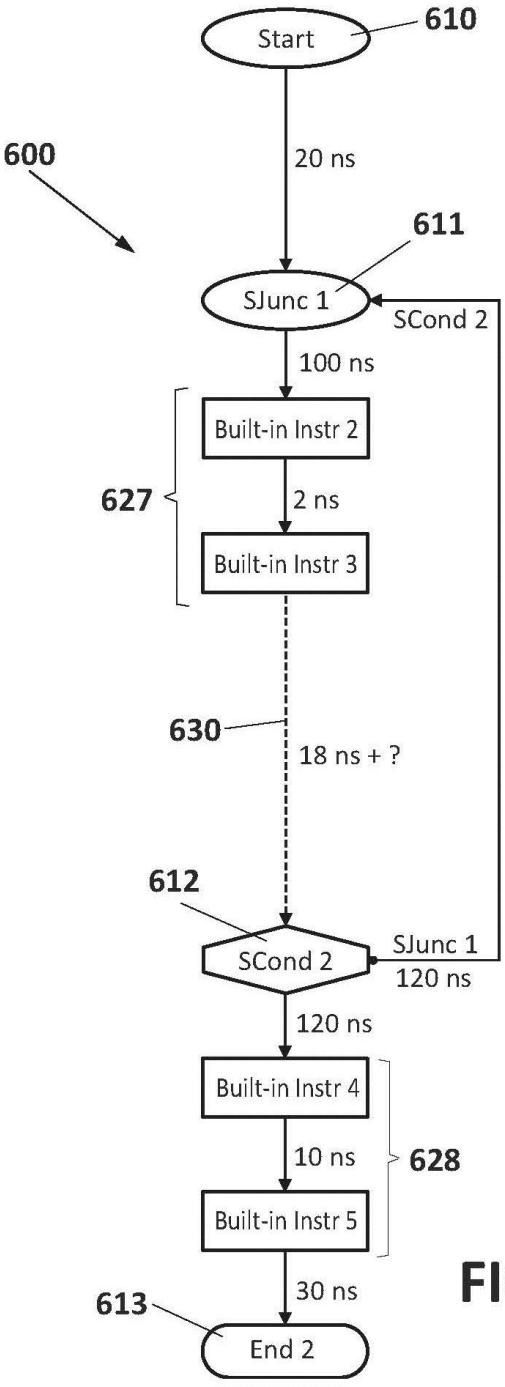
Start
610
600
20 ns
611
SJunc 1
SCond 2
100 ns
Built-in Instr 2
627
2 ns
Built-in Instr 3
630
18 ns + ?
612
SCond 2
SJunc 1
120 ns
120 ns
Built-in Instr 4
10 ns
628
Built-in Instr 5
30 ns
613
End 2

FIG. 6B

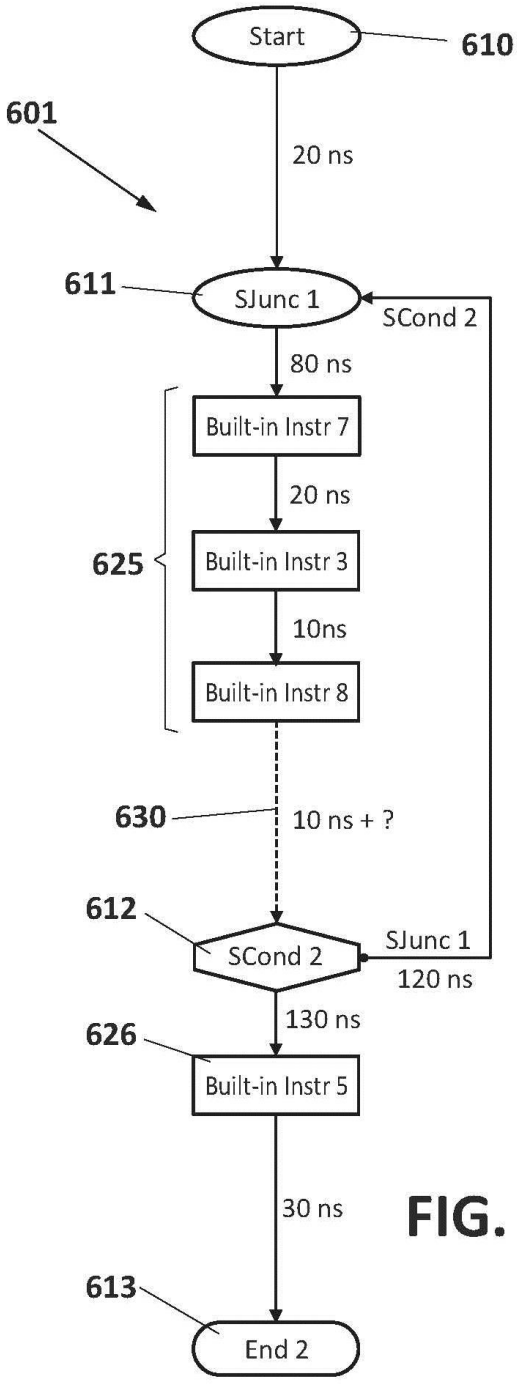
Start
610
601
20 ns
611
SJunc 1
SCond 2
80 ns
Built-in Instr 7
20 ns
625
Built-in Instr 3
10ns
Built-in Instr 8
630
10 ns + ?
612
SCond 2
SJunc 1
120 ns
130 ns
626
Built-in Instr 5
30 ns
613
End 2

FIG. 6C

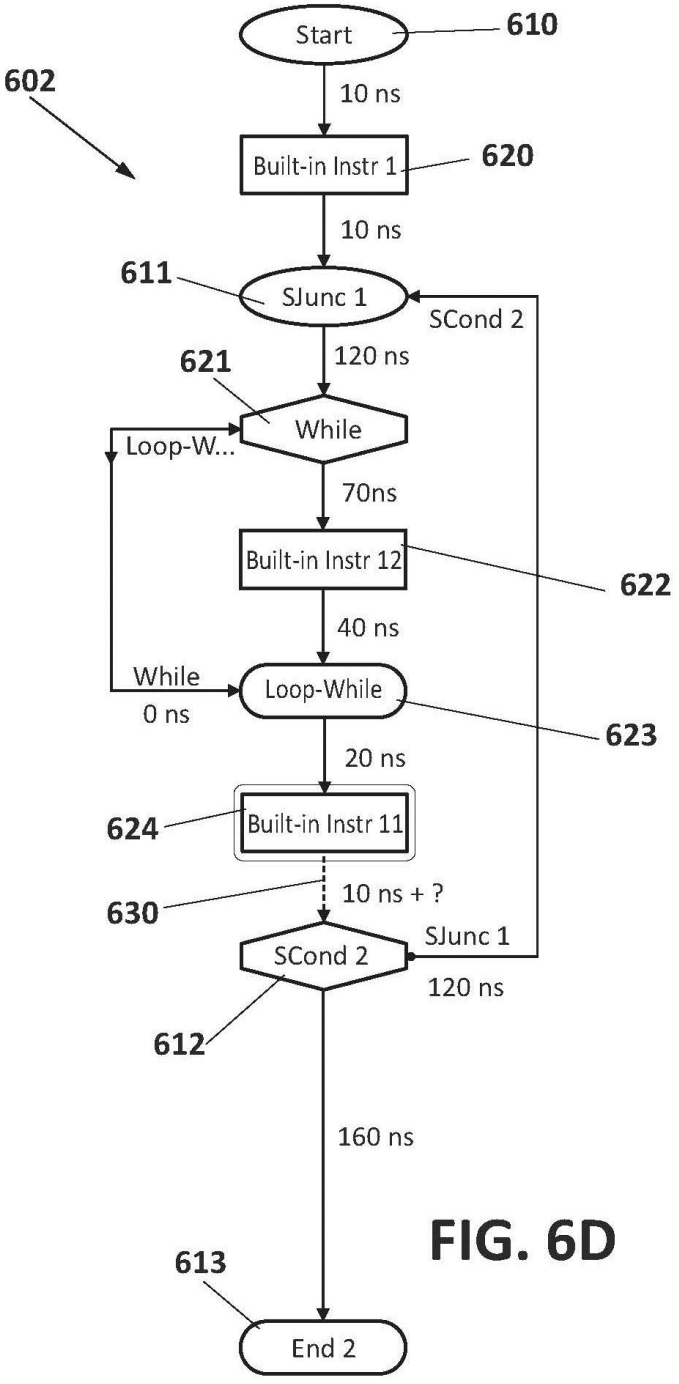
Start
610
602
10 ns
Built-in Instr 1
620
10 ns
611
SJunc 1
SCond 2
120 ns
621
While
Loop-W...
70ns
Built-in Instr 12
622
40 ns
While
0 ns
Loop-While
623
20 ns
624
Built-in Instr 11
630
10 ns + ?
SCond 2
SJunc 1
120 ns
612
160 ns
613
End 2

FIG. 6D

OFICINA ESPAÑOLA
DE PATENTES Y MARCAS

ESPAÑA

(21) N.º solicitud: 201531155

(22) Fecha de presentación de la solicitud: 03.08.2015

(32) Fecha de prioridad:

INFORME SOBRE EL ESTADO DE LA TECNICA

(51) **Int. Cl. :** ***G06F13/42*** (2006.01)

DOCUMENTOS RELEVANTES

Categoría	(56) **Documentos citados**	**Reivindicaciones afectadas**
A	SCHÜTZ W.. "On the testability of distributed real-time systems". Proceedings Of The Symposium On Reliable Distributed Systems. PISA, sept. 30 - oct. 2, 1991; [Proceedings Of The Symposium On Reliable Distributed Systems], 19910930; 19910930-19911002 Los Alamitos, IEEE Como. Soc. Press, US 30.09.1991 vol: symp. 10 págs: 52-61 XP010025114 ISBN 978-0-8186-2260-1; ISBN 0-8186-2260-1.	1-20
A	MARTIN SCHOEBERL. "Is time predictability quantifiable?". Embedded Computer Systems (SAMOS), 2012 International Conference on, 20120716 IEEE 16.07.2012 págs: 333-338 ISBN 978-1-4673-2295-9; XP032300791 ISBN 1-4673-2295-4.	1-20

Categoría de los documentos citados
X: de particular relevancia
Y: de particular relevancia combinado con otro/s de la misma categoría
A: refleja el estado de la técnica
O: referido a divulgación no escrita
P: publicado entre la fecha de prioridad y la de presentación de la solicitud
E: documento anterior, pero publicado después de la fecha de presentación de la solicitud

El presente informe ha sido realizado

☒ para todas las reivindicaciones ☐ para las reivindicaciones nº:

Fecha de realización del informe	**Examinador**	**Página**
18.11.2015	J. Botella Maldonado	1/4

Documentación mínima buscada (sistema de clasificación seguido de los símbolos de clasificación)

G06F

Bases de datos electrónicas consultadas durante la búsqueda (nombre de la base de datos y, si es posible, términos de búsqueda utilizados)

INVENES, EPODOC, WPI, NPL, XPESP, XPIAP, XPI3E, INSPEC.

Fecha de Realización de la Opinión Escrita: 18.11.2015

Declaración

Novedad (Art. 6.1 LP 11/1986)	Reivindicaciones	1-20	**SI**
	Reivindicaciones		**NO**
Actividad inventiva (Art. 8.1 LP11/1986)	Reivindicaciones	1-20	**SI**
	Reivindicaciones		**NO**

Se considera que la solicitud cumple con el requisito de aplicación industrial. Este requisito fue evaluado durante la fase de examen formal y técnico de la solicitud (Artículo 31.2 Ley 11/1986).

Base de la Opinión.-

La presente opinión se ha realizado sobre la base de la solicitud de patente tal y como se publica.

1. Documentos considerados.-

A continuación se relacionan los documentos pertenecientes al estado de la técnica tomados en consideración para la realización de esta opinión.

Documento	Número Publicación o Identificación	Fecha Publicación
D01	SCHÜTZ W.. "On the testability of distributed real-time systems". Proceedings Of The Symposium On Reliable Distributed Systems. PISA, sept. 30 - oct. 2, 1991; [Proceedings Of The Symposium On Reliable Distributed Systems], 19910930; 19910930-19911002 Los Alamitos, IEEE Como. Soc. Press, US 30.09.1991 vol: symp. 10 págs: 52-61 XP010025114 ISBN 978-0-8186-2260-1; ISBN 0-8186-2260-1	30.09.1991
D02	MARTIN SCHOEBERL. "Is time predictability quantifiable?". Embedded Computer Systems (SAMOS), 2012 International Conference on, 20120716 IEEE 16.07.2012 págs: 333-338 ISBN 978-1-4673-2295-9; XP032300791 ISBN 1-4673-2295-4	16.07.2012

2. Declaración motivada según los artículos 29.6 y 29.7 del Reglamento de ejecución de la Ley 11/1986, de 20 de marzo, de Patentes sobre la novedad y la actividad inventiva; citas y explicaciones en apoyo de esta declaración

El documento D01 presenta un estudio de los sistemas en tiempo real distinguiendo entre sistemas dirigidos por tiempo (TT) y sistemas dirigidos por eventos (ET). Se considera un modelo distribuido consistente en un conjunto de nodos interconectados en red y cada uno de ellos con CPU, memoria local, periféricos, acceso a red, reloj local. Los sistemas TT inician sus acciones exclusivamente a determinados puntos temporales por lo que se requiere acceder a un tiempo global que se implementa sincronizando los relojes de todos los nodos, el acceso al tiempo global en determinados puntos durante la ejecución de un proceso debe ser determinista si se conocen los tiempos de ejecución de las instrucciones.

El documento D02 realiza un estudio de la previsibilidad en la ejecución de procesos en tiempo real a través del tiempo de ejecución en el peor de los casos (WCET) para distintas arquitecturas, compiladores y herramientas de análisis del WCET.

Consideramos que ninguno de estos documentos anticipa la invención tal como se reivindica en las reivindicaciones de la 1ª a la 20ª, ni se encuentra en ellos, tomados solos o en combinación, sugerencias que dirijan a un experto en la materia hacia el objeto reivindicado en las citadas reivindicaciones.

Por lo tanto la invención tal como se reivindica en las citadas reivindicaciones de la 1ª a la 20ª posee novedad y actividad inventiva.