Nombre: Roberto Federico Farfán

Preguntas orientadoras

1. Describa brevemente los diferentes perfiles de familias de microprocesadores/microcontroladores de ARM. Explique alguna de sus diferencias características.

Uno de los procesadores más exitosos de ARM fue el ARM7TDMI. A diferencia de los procesadores tradicionales de 32 bits, el ARM7TDMI admite dos conjuntos de instrucciones, uno de 32 bits y otro de 16 bits, llamado Thumb. Su arquitectura permite que ambos conjuntos de instrucciones se utilicen en el procesador, permitiendo que la densidad del código aumenta considerablemente.
Al mismo tiempo, las tareas críticas aún pueden ejecutarse con buena velocidad, esto permite que los procesadores ARM sean la primera opción para dispositivos móviles como teléfonos celulares, que requieren poca energía y poca memoria.

Desde entonces, ARM ha seguido desarrollando nuevos procesadores para abordar las necesidades de diferentes aplicaciones. Por ejemplo, los procesadores ARM9 se utiliza en una gran cantidad de microcontroladores de alto rendimiento y el procesador ARM11 se utiliza en una gran cantidad de teléfonos móviles
.
Tras la introducción de la familia ARM11, se decidió que muchas de las nuevas tecnologías, como el conjunto de instrucciones optimizado Thumb-2, sean igualmente aplicables a los mercados de menor costo de microcontroladores y componentes para automotriz.

También se decidió que, si bien la arquitectura debía ser coherente desde el MCU más bajo hasta el procesador de aplicaciones de mayor rendimiento, era necesario ofrecer arquitecturas que se adapten mejor a las aplicaciones, lo que permitiría procesadores muy deterministas en los mercados de menor costo.

En los últimos años, ARM ha ampliado su cartera de productos al diversificar su desarrollo de CPU, lo que dio como resultado el nombre de la nueva familia de procesadores "Cortex". En esta gama de procesadores Cortex, los procesadores se dividen en tres perfiles, A, R y M.
• El perfil A está diseñado para plataformas de aplicaciones abiertas de alto rendimiento.
• El perfil R está diseñado para sistemas integrados de gama alta en los que se necesita rendimiento en tiempo real.
• El perfil M está diseñado para sistemas de tipo microcontrolador.

Cortex-A: procesadores que están diseñados para manejar aplicaciones complejas, como sistemas operativos (SO) integrados de gama alta (por ejemplo, iOS, Android, Linux y Windows). Estas aplicaciones requieren de mayor capacidad de procesamiento, soporte de un sistema de memoria virtual con unidades de administración de memoria (MMUs) y, opcionalmente, soporte de Java y un entorno de ejecución de programa seguro. Se utilizan en teléfonos inteligentes de gama alta, tablets, televisores e incluso servidores informáticos.

Cortex-R: Procesadores de alto rendimiento de tiempo real. Se trata de aplicaciones como controladores de disco duro, controladores del baseband para comunicaciones móviles y componentes implementados en el mercado automotriz, donde la capacidad de procesamiento y la alta confiabilidad son esenciales. Para estos sistemas, la baja latencia y el determinismo son los parámetros claves en su uso.

Cortex-M: Procesadores destinados a aplicaciones de menor escala, como microcontroladores, donde son importantes criterios como; bajo costo, bajo consumo, eficiencia energética y baja latencia de interrupción. Al mismo tiempo, el diseño del procesador debe ser fácil de usar y capaz de proporcionar un comportamiento determinista como se requiere en muchos sistemas de control en tiempo real.

Al crear esta partición de gama de productos, se abordaron de manera específica los requisitos de cada segmento necesita, lo que permitió que la arquitectura ARM alcance incluso más aplicaciones que en un principio.

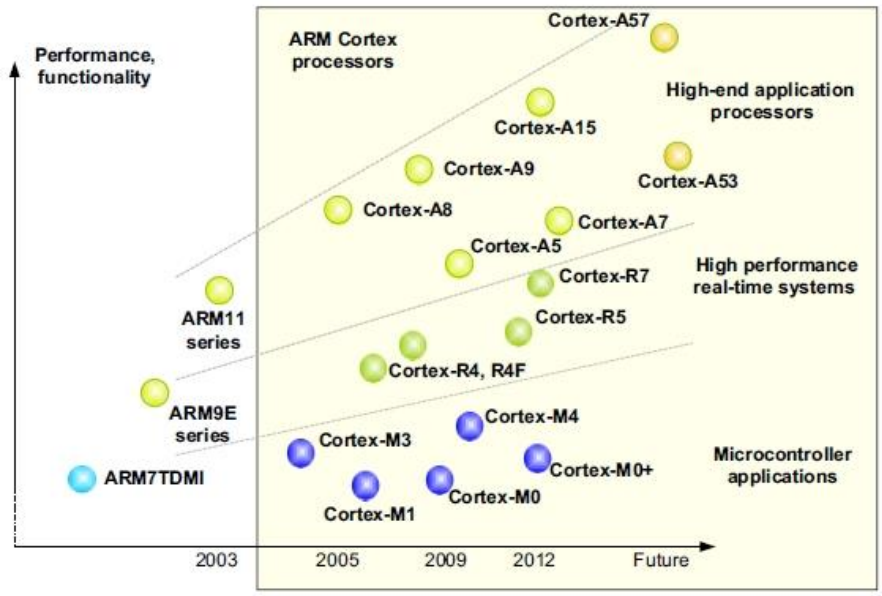

Figura 1. Familia de procesadores ARM.

Cortex M

1. Describa brevemente las diferencias entre las familias de procesadores Cortex M0, M3 y M4.

Los procesadores Cortex -M3 y Cortex-M4 son dos de los productos de la familia de procesadores ARM Cortex-M. Los procesadores Cortex-M3 y Cortex-M4 se basan en la arquitectura ARMv7-M. Ambos son procesadores de alto rendimiento que están diseñados para desempeñar tareas en aplicaciones de microcontroladores.
Debido a que el procesador Cortex-M4 tiene SIMD, MAC rápido e instrucciones aritméticas saturadas, también puede realizar algunas aplicaciones de procesamiento de señales digitales que tradicionalmente han sido realizadas por un Procesador de Señal Digital (DSP) por separado.
Los procesadores Cortex-M0, Cortex-M0+ y Cortex-M1 se basan en ARMv6-M, que tiene un conjunto de instrucciones más pequeño. El procesador Cortex-M0+ tiene las optimizaciones de bajo consumo más avanzadas y tiene más funciones opcionales disponibles.
El procesador Cortex-M1 está diseñado específicamente para aplicaciones de FPGA. Tiene funciones de memoria estrechamente acoplada (TCM) que se pueden implementar utilizando memorias dentro de la FPGA, y el diseño permite operaciones de alta frecuencia de reloj en FPGA avanzada. Por ejemplo, puede funcionar a más de 200MHz en Altera Stratix III FPGA.

Para tareas generales de procesamiento de datos y control de E/S, los procesadores Cortex-M0 y Cortex-M0+ tienen una excelente eficiencia energética debido al diseño de bajo número de puertas.
Para aplicaciones con requisitos de procesamiento de datos complejos, pueden requerir más instrucciones y ciclos de reloj. Para estos casos el procesador Cortex-M3 o Cortex-M4 son las opciones más adecuadas, ya que las instrucciones adicionales disponibles en estos procesadores permiten realizar el procesamiento con menos instrucciones en comparación con la arquitectura ARMv6-M. Como resultado, necesitamos diferentes procesadores para diferentes aplicaciones.
Vale la pena señalar que los procesadores Cortex-M no son los únicos procesadores de ARM que se utilizan como microcontroladores genéricos. El procesador ARM7 ha tenido mucho éxito en este mercado, con empresas como NXP (anteriormente Philips Semiconductor), Texas Instruments, Atmel, OKI y muchos otros proveedores que ofrecen microcontroladores basados en ARM, que utilizan procesadores ARM clásicos como ARM7TDMI.

2. ¿Por qué se dice que el set de instrucciones Thumb permite mayor densidad de código? Explique

Se llama densidad de código a la medida de cuánta memoria necesita un sistema embebido para contener instrucciones. Frecuentemente en los sistemas embebidos hay una limitación al tamaño de la memoria. Esto es especialmente cierto para los sistemas con la memoria dentro del chip, en los cuales la memoria normalmente ocupa más espacio en el chip que la propia CPU, por ejemplo la memoria caché.
La arquitectura del procesador ARM se basa en los principios RISCs aunque tiene una mejor densidad de código que la mayoría de los procesadores RISCs. Sin embargo, su densidad de código todavía no alcanza a ser tan buena como la de varios de los procesadores CISCs.
Los procesadores Cortex-M3 y Cortex-M4 utilizan una arquitectura de 32 bits. Los registros internos en el banco de registros, la ruta de datos y las interfaces de bus tienen 32 bits de ancho. La arquitectura del conjunto de instrucciones (ISA) en los procesadores Cortex-M se llama Thumb ISA y se basa en la tecnología Thumb-2 que admite una combinación de instrucciones de 16 y 32 bits. Para las aplicaciones en que es primordial la importancia de la densidad del código, ARM incorporó el mecanismo llamado arquitectura “Thumb”.
El set de instrucciones Thumb es una forma comprimida a 16 bits del set de instrucciones ARM de 32 bits original y emplea hardware de descompresión dinámica en la instrucción pipeline, para descomprimir las instrucciones de 16 a 32 bits. Por supuesto, la etapa extra requerida para manejar instrucciones de 16 bits, afecta el rendimiento. La densidad del código Thumb es mejor que la alcanzada por la mayoría de los procesadores CISCs.
El set de instrucciones Thumb (sub set del de 32 bits) usado para obtener alta densidad de código en varios procesadores ARM utiliza, predominantemente, una arquitectura de dos direcciones.

3. ¿Qué entiende por arquitectura load-store? ¿Qué tipo de instrucciones no posee este tipo de arquitectura?

Como la mayoría de los procesadores RISCs, los procesadores Cortex-M se basan en una arquitectura de almacenamiento de carga (Load-store). En esta arquitectura, las instrucciones que acceden a memoria están separadas de las instrucciones que procesan los datos, ya que en este último caso los datos necesariamente están en registros.

ARM no soporta operaciones memoria a memoria. Por lo tanto todas las instrucciones ARM caen en una de las tres categorías siguientes:
1. **Instrucciones que procesan datos**. Solamente usan y modifican valores en registros. Una instrucción, por ejemplo, puede sumar dos registros y ubicar el resultado en otro registro.
2. **Instrucciones de transferencia de datos**. Estas copian los datos de la memoria en registros (instrucciones de carga) o copian los datos de los registros en la memoria (instrucciones de almacenamiento). Una forma adicional, útil solamente en códigos de sistemas, intercambian un dato en memoria con un dato en un registro.
3. **Instrucciones de control de flujo**. Normalmente se ejecutan instrucciones ubicadas en direcciones de memorias consecutivas. Aunque frecuentemente el control del flujo de las instrucciones ocasiona que la ejecución conmute en una dirección diferente, ya sea en forma permanente (instrucciones de salto) o guarde una dirección de retorno para recuperar la secuencia original (instrucciones de salto y retorno) o ejecute un código de llamadas al supervisor del sistema, instrucciones tipo trapping, “atrapadas”.

4. ¿Cómo es el mapa de memoria de la familia?

El espacio de direcciones de 4 GB de los procesadores Cortex -M se divide en varias regiones de memoria (Figura 4.18). La partición se basa en usos típicos, por lo que las diferentes áreas están diseñadas para usarse principalmente para:
• Accesos de código de programa (p. ej., región CODE)
• Accesos a datos (p. ej., región SRAM)
• Periféricos (p. ej., región de periféricos)
• Componentes de depuración y control interno del procesador (p. ej., Bus periférico privado)
La arquitectura también permite una alta flexibilidad para permitir que las regiones de memoria se utilicen para otros fines. Por ejemplo, los programas se pueden ejecutar tanto desde el CODE como desde la región SRAM, y un microcontrolador también puede integrar bloques SRAM en la región del CODE.
En la práctica, muchos microcontroladores solo usan una pequeña parte de cada región para flash de programa, SRAM y periféricos. Algunas de las regiones pueden quedar sin usar.
Los diferentes microcontroladores tienen diferentes tamaños de memoria y ubicaciones de direcciones periféricas. Esta información generalmente se describe en los manuales de usuario o en las hojas de datos de los proveedores de microcontroladores.

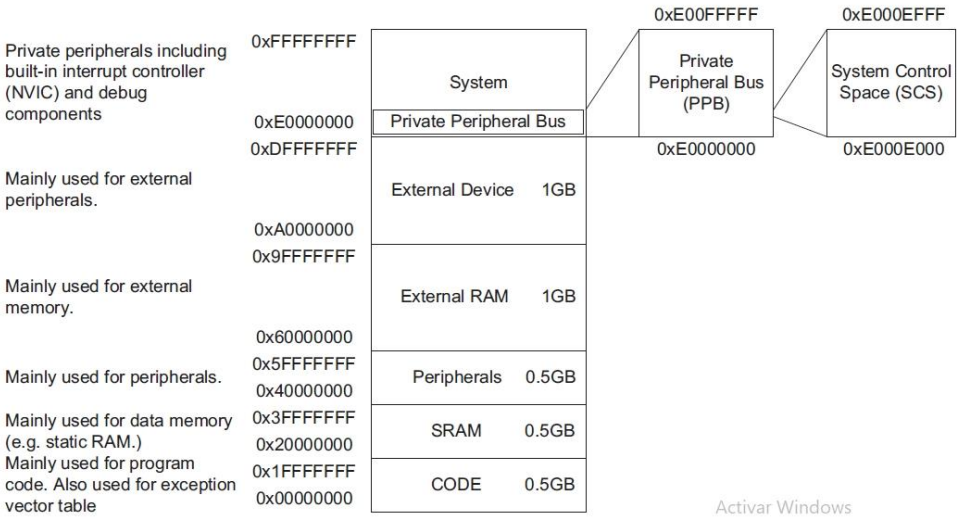

Figura 2. Mapa de memoria.