**Preguntas orientadoras**

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

**Cortex M**

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

**Cortex M0**

Esta familia de procesadores esta basada en una version de arquitectura ARMv6-M. Se selecciona como microcontrolador. Entre sus caracteristicas se puede mencionar Thumb-2 subset (instrucciones 16-bit Thumb & BL, MRS, MSR, ISB, DSB, and DMB). No dispone de memoria Caché. La potencia del procesador es de 0.9 Millones de Instrucciones por segundo / MHz.

**Cortex M3**

Esta familia de procesadores esta basada en una version de arquitectura ARMv7-M. Se selecciona como microcontrolador. Entre sus caracteristicas se puede mencionar Thumb-2 únicamente. Reparto de instrucciones por Hardware. No dispone de memoria Caché. Tiene opcional una MPU (Unidad de protección de memoria), por lo que el procesador admite el modelo estándar de arquitectura de sistema de memoria protegida ARMv7 (regiones de protección, permisos de accesos, entre otros). La potencia del procesador es de 125 Millones de Instrucciones por segundo para 100 MHz.

**Cortex M4**

Esta familia de procesadores esta basada en una version de arquitectura ARMv7-ME. Se selecciona como microcontrolador. Entre sus caracteristicas se puede mencionar Thumb y Thumb-2, FPU. MAC, SIMD e instrucciones divididas. Tiene MPU. La potencia del procesador es de 1.25 Millones de Instrucciones por segundo / MHz.

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

Todas las instrucciones Thumb tienen una longitud de 16 bits (se diferencia de las instrucciones ARM que son de 32 bits fijas) y se almacenan alineadas a media palabra en la memoria. Debido a que las instrucciones se almacenan alineadas a media palabra, el bit inferior de la dirección de una instrucción siempre se establece en cero en el estado Thumb. La funcionalidad del conjunto de instrucciones Thumb, con una excepción, es un subconjunto de la funcionalidad del conjunto de instrucciones ARM. El conjunto de instrucciones está optimizado para la producción mediante un compilador de C.

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

Se entiende que en una arquitectura load - store se hace una categorizaciòn entre acceso a memoria y las operaciones en la ALU, en la que en una instrucción ambos operandos estan en los registros, en donde se realiza o ejecuta la instruccion.

* ¿Cómo es el mapa de memoria de la familia?

El mapa de memoria de la familia ARM esta divido en las siguientes partes: el mapa para Código iniciando en 0x00000000. Otra parte es la SRAM usado para los datos de memoria, perifericos, RAM externa y dispositivos externos. Tambien la parte de memoria dedicada al Bus de perifericos y al Sistema.

* ¿Qué ventajas presenta el uso de los “shadowed pointers” del PSP y el MSP?

El hecho de disponer de dos stack, permite robustecer el sistema. Ya que uno seria utilizado por el SO y el otro por la aplicación del programador. De esta manera el Sistema nunca se quedaria sin espacio en la pila.

* Describa los diferentes modos de privilegio y operación del Cortex M, sus relaciones y como se conmuta de uno al otro. Describa un ejemplo en el que se pasa del modo privilegiado a no priviligiado y nuevamente a privilegiado.

Los modos de funcionamiento son en Modo Thread y Handler. En modo Thread se desarrolla normalmente las instrucciones del programa y si se esta en modo privilegiado se puede acceder a partes del sistema que son criticas (acceso a memoria por ejemplo). Cuando se atiende a una interrupcion se accede al modo Handler y si existe un scheduler se puede acceder al modo thread No privilegiado (sin acceso a partes criticas del sistema).

* ¿Qué se entiende por modelo de registros ortogonal? Dé un ejemplo

Se entiende que se determina la posibilidad de combinar en una instrucción: operaciones, tipos de datos y modo de direccionamiento.

* ¿Qué ventajas presenta el uso de intrucciones de ejecución condicional (IT)? Dé un ejemplo

Por ejemplo en un ANDNE, no se rompe el pipeline, debido a que no hay salto, por lo cual se mantiene la cantidad de ejecuciones por ciclo de reloj.

* Describa brevemente las excepciones más prioritarias (reset, NMI, Hardfault).

Estas excepciones son interrupciones por software. tienen la prioridad mas lata (reset = -1, NMI = -2 y Hardfault = -3). Estas no son accesible por el programador. Por ejemplo, al energizar un microcontrolador se llama al modo reset. cuando existe un error por ejemplo en una interrupcion por parte del programador se hace el llamado de Hardfault.

* Describa las funciones principales de la pila. ¿Cómo resuelve la arquitectura el llamado a funciones y su retorno?

La Pila es parte de la SRAM. Cuando se llama de una funcion se procura guardar el contexto del instante antes de hacer el salto. Se usa la Pila para guardar valores de variables locales, datos a funciones, etc. Aparte de guardar parametros en los registros especificos, se hace uso de un registro especial llamado Link Register (LR) en donde se guarda la direccion de memoria desde donde se llamo a función.

* Describa la secuencia de reset del microprocesador.

En la primera secuencia luego de un reset en el microcontrolador es la de Fetch Inicial MSP value, en la que se guardan los primero 4 bytes del stack pointer. Luego la segunda secuencia (en la dirección 0x00000004) el reset handler lo que hace es configurar el microcontrolador (perifericos, la SRAM, etc). Posteriormente se hace el llamado de la primera instrucciòn del programa.

* ¿Qué entiende por “core peripherals”? ¿Qué diferencia existe entre estos y el resto de los periféricos?

ARM en Cortex -M se reserva la implementacion de dos perifericos, el NVIC y el Systick Timer. Los fabricantes suelen incluir el resto de los perifericos (UART, manejador LCD, controlador de meoria externa, etc). Estos ayudan a los SO en tiempo real.

* ¿Cómo se implementan las prioridades de las interrupciones? Dé un ejemplo

Cortex M tiene un controlador de interrupciones programables (NVIC). Salvo la 3 primeras interrupciones, las cuales tiene prioridad fija, con el resto se dispone hasta 128 niveles de prioridades, debido al manejo de 3 o bits (tener presente que ademas de las interrupciones por software se cuenta con 240 fuentes de interrupciones externas)

* ¿Qué es el CMSIS? ¿Qué función cumple? ¿Quién lo provee? ¿Qué ventajas aporta?

El CMSIS es una libreria, capa de abstracción de harware. Lo provee ARM. La ventaja es que es independiente del fabricante (al menos en el alcance a los core peripferical). Esta lo mas cercano a la capa del harware.

* Cuando ocurre una interrupción, asumiendo que está habilitada ¿Cómo opera el microprocesador para atender a la subrutina correspondiente? Explique con un ejemplo

Por ejemplo cuando se llama a una subrutina, se guarda el Contador del programa en LR y los parametros en los registros r0, r1, r2 y r3. En caso de ser necesario mas parametros hay que hacer push en los registros siguientes. Luego en la interrupcion (handler mode) se ejecuta lo correspondiente a la subtarea. A finalizar se hace un pop de los parametros que devuelve la subrutina y desde el LR se hace el cargado al CP para devolver la direccion del programa desde donde se llamo a la subrutina.

* ¿Cómo cambia la operación de stacking al utilizar la unidad de punto flotante?

Primero el flag del resgitro de estado del FPU se consulta para saber si se esta usando el mismo. De ser asi se guarda los 16 registro del FPU y posteriormente los resgitros generales.

* Explique las características avanzadas de atención a interrupciones: tail chaining y late arrival.

Es una técnica de optimización del cpu. Debido a que se dispara otra interrupción durante la ejecución de la primera y sin volver a hacer el stacking se ejecuta la siguiente interrupción. No importa el nivel de prioridad.

El late arrival es cuando ocurre un stacking y en ese instante se dispara una interrupcion de mayor prioridad, se guarda la configuración del contexto pero se atiende la interrupcion de mayor prioridad. La de menor prioridad queda esperando.

* ¿Qué es el systick? ¿Por qué puede afirmarse que su implementación favorece la portabilidad de los sistemas operativos embebidos?

El systick es un contador de 24 bits. Dispara una interrupcion al desbordar. Favorece la portabilidad de los sitemas operativos debido a que sincroniza con el reloj o con una fuente externa.

* ¿Qué funciones cumple la unidad de protección de memoria (MPU)?

Es protección de memoria. Se proteje paginas del mapa de memoria. Las asigna el progrmador. No se protejen los registros.

* ¿Cuántas regiones pueden configurarse como máximo? ¿Qué ocurre en caso de haber solapamientos de las regiones? ¿Qué ocurre con las zonas de memoria no cubiertas por las regiones definidas?

Se pueden configurar hasta 8 regiones de memoria. En caso de acceder a memoria no cubierta se toma como malicioso y se dispara las interrupciones hardfault o MemManage

* ¿Para qué se suele utilizar la excepción PendSV? ¿Cómo se relaciona su uso con el resto de las excepciones? Dé un ejemplo

Esta interrupcion se la suele usar en los SO. Cada tarea tiene un contexto de ejecucion. Y el PendSV que es la de menor prioridad se la usa para los cambios de contextos, por lo cual el SO puede determinar a que contexto volver.

* ¿Para qué se suele utilizar la excepción SVC? Expliquelo dentro de un marco de un sistema operativo embebido.

**ISA**

* ¿Qué son los sufijos y para qué se los utiliza? Dé un ejemplo
* ¿Para qué se utiliza el sufijo ‘s’? Dé un ejemplo
* ¿Qué utilidad tiene la implementación de instrucciones de aritmética saturada? Dé un ejemplo con operaciones con datos de 8 bits.
* Describa brevemente la interfaz entre assembler y C ¿Cómo se reciben los argumentos de las funciones? ¿Cómo se devuelve el resultado? ¿Qué registros deben guardarse en la pila antes de ser modificados?
* ¿Qué es una instrucción SIMD? ¿En qué se aplican y que ventajas reporta su uso? Dé un ejemplo.