NVIC

Monday, 8 November, 2021 17:18

- Controlador vectorizado de interrupciones anidadas
- Cada interrupción va a tener una dirección que le indica a qué dirección de memoria debe saltar para manejar dicha interrupción
- Podemos tener como fuente de interrupciones los periféricos de la placa y los pines de GPIO
- Definiciones:
  - Excepciones:
    - Son aquellas interrupciones que corresponden al core. Éste se comunica con el NVIC para que genere las interrupciones del sistema
    - El timer del sistema, por ejemplo, genera una excepción.
  - Interrupciones: Son interrupciones que ocurren a partir de periféricos y puertos de entrada/salida. Algunas fuentes de interrupciones son:
    - SysTick
    - NMI (interrupciones no enmascarables)
    - Periféricos internos de la placa
    - EXTI (puertos de entrada/salida)
- Este controlador admite hasta 35 interrupciones anidadas
- Cada registro tiene 5 bits (32 opciones) que me permiten setearle un nivel de prioridad
- Pueden generarse interrupciones por software
- Las direcciones a las que salta ya vienen programadas de fábrica:
  - Desde la posición de memoria 0x01 hasta la 0x3c corresponden a interrupciones del sistema (excepciones)
  - Desde la posición de memoria 0x40 hasta la 0x74 corresponden a interrupciones por periféricos y pines de input/output
  - En estas direcciones va a mediante un cálculo, la dirección donde se encuentra el handler de la interrupción, el cual abarca la rutina de servicio.
- Cuando llega una interrupción, la misma queda pendiente mientras se guarda el contexto del programa automáticamente, en un stack pointer descendiente. Cuando la interrupción se termina de atender, se hace un *unstack* para recuperar el contexto del programa.
- Si llega una interrupción de mayor prioridad que la que se está atendiendo en este momento, vuelve a pasar lo mismo: se salva el contexto de la rutina de interrupción que se estaba ejecutando, y se pasa a atender esta nueva interrupción.
- Las interrupciones están en estado *pending* mientras se salva contexto, y en *active* mientras está siendo atendida.
- Pueden asignarse sub prioridades. En el registro de prioridades, los primeros 2 bits son para definir la prioridad y los otros 2 para definir la sub prioridad. Esta sub prioridad sirve para resolver el conflicto en el que se esté atendiendo una interrupción y llegue otra interrupción con la misma prioridad.
- La tabla de vector de interrupciones está en la sección de memoria de código (memoria flash interna)
  - Los puntos de entrada de todas las rutinas de servicio de interrupción (ISR) están en esta tabla
  - Cada entrada en esta tabla es de 32 bits y contiene la dirección de memoria inicial de la ISR (es un vector de punteros a funciones)
  - La dirección del puntero que contiene la ISR se realiza de la forma: $address = 64 + 4*n$ siendo n el número de la interrupción
  - El número de la interrupción puede ser negativo, lo cual indica que es una excepción del sistema
- Cuando se está atendiendo una interrupción y llega otra de menor prioridad, la última queda esperando
  - Cuando se termina de atender la interrupción de mayor prioridad, en vez de devolver contexto y salvar contexto nuevamente para atender la siguiente interrupción de menor prioridad, el micro procesador aplica una optimización llamada "tail chaining" para reducir la latencia de interrupción
  - Típicamente, el proceso de devolver contexto toma 12 ciclos y el de salvar contexto otros 12 ciclos adicionales
  - El proceso de "tail chaining" toma sólo 6 ciclos
- Hay interrupciones no enmascarables, las cuales no pueden ser ignoradas por el sistema, y suelen hacer referencia a errores de hardware no recuperables.