MSE | MASTER OF SCIENCE IN ENGINEERING

TSM_AdvEmbSof

Scheduling for Embedded Systems (Part II)

Serge Ayer | 19.10.2023 | Cours MSE

Event-driven system

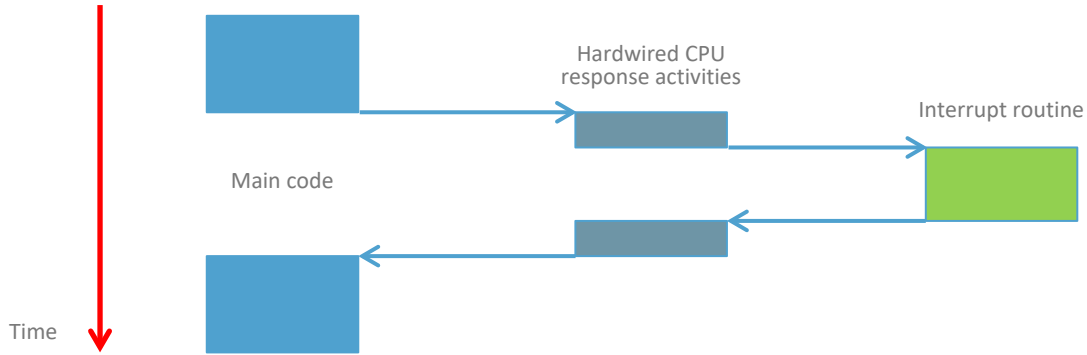
- Event:
  - Generated by a source
  - Recognized by a recipient
- Events can happen in a predictable or unpredictable manner:
  - Periodic events happen when the timer counts has reached a certain value
  - User inputs are not predictable and may happen at any time and in any order
- The super-loop static schedule model is not suited to dealing with unpredictable events
- An event-driven system allows not to continuously check for inputs
  - Take actions only in response to events
  - Task scheduling becomes dynamic

Events and interrupts

- Handling events is done using interrupts
  - Special hardware in the MCU that detects the event and run an Interrupt Service Routine (ISR) in response
- Has many advantages
  - It is efficient: runs only when necessary - when the event occurs
  - It is fast: it is implemented as a hardware mechanism
  - It is scalable:
    - The ISR response time doesn't depend on most other processing
    - Coding of ISRs can be developed independently of each other

Interrupt Processing Sequence

- The main code is running
  - It can be in idle state
  - When the interrupt trigger occurs
    - The processor does some hard-wired processing
    - The processor then executes the ISR, including return-from-interrupt processing at the end

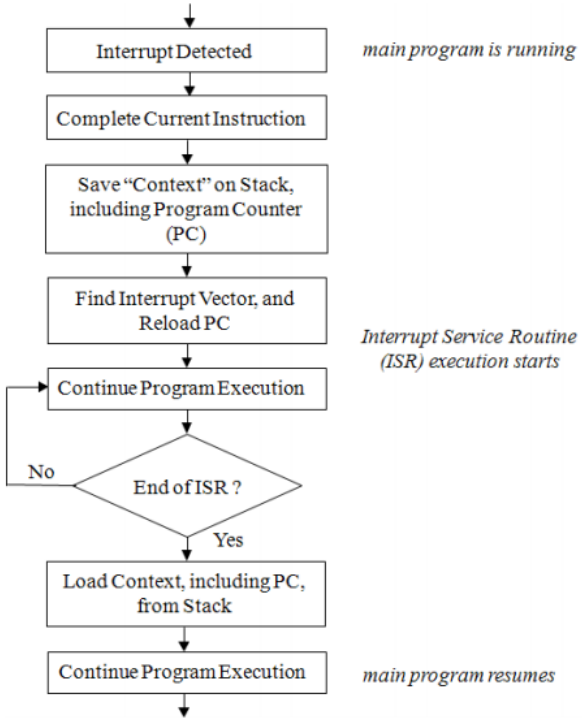


Getting Deeper into Interrupts

- To deal with more complex interrupt applications, most processors support the following mechanisms:
  - Interrupts can be prioritized: some can be defined as more important than others. If two occur at the same time, then the higher priority one executes first.
  - Interrupts can be masked: switched off, if they are not needed, or are likely to get in the way of more important activity. This masking could be just for a short period, for example while a critical program section completes.
  - Interrupts can be nested: this means that a higher priority interrupt can interrupt one of lower priority. Working with nested interrupts increases the demands on the programmer, and is strictly for advanced players only.
  - The location of the ISR in memory can be selected: to suit the memory map and programmer wishes.

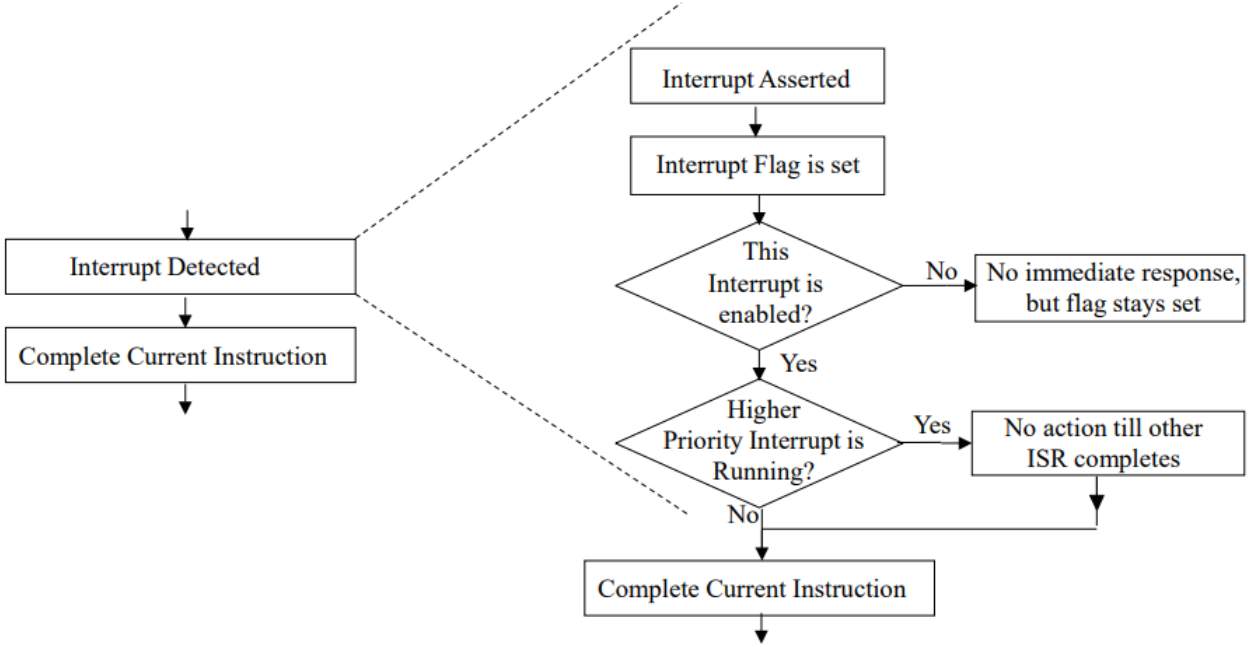
Getting Deeper into Interrupts

- The delay between the interrupt occurring, and the processor responding, is called the interrupt latency.
- While the interrupt is waiting for a response from the processor, it is said to be pending
- When responding to interrupts, microcontrollers follow this general pattern



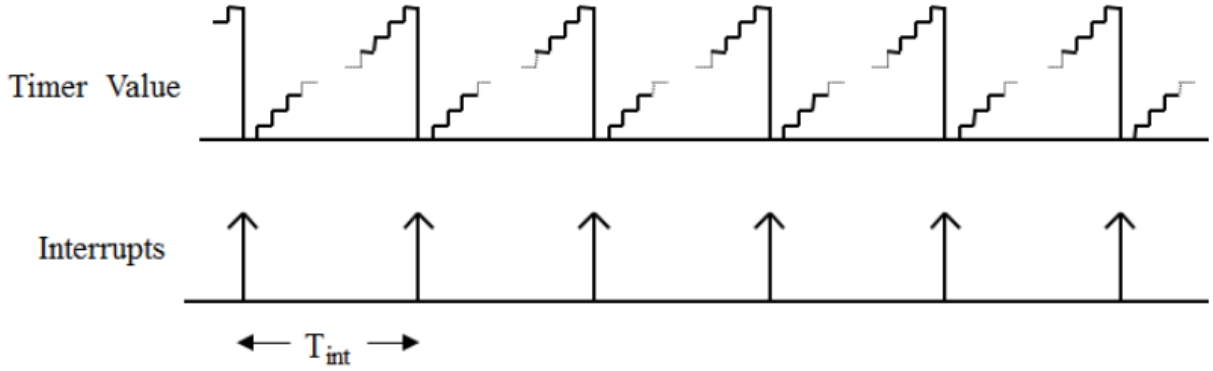
Getting Deeper into Interrupts

- The typical microcontroller interrupt response is



Counting and Timing: Interrupt on Overflow

- Making a counter is easy
- If the clock source is a known and stable frequency, then the counter becomes a timer.
- Many microcontroller counters cause an interrupt as the counter overflows: this interrupt can be used to record the overflow, and the count can continue in a useful wav



Mbed OS Timers

- The Timer mechanism is made available in the Mbed OS with
  - Timer: create, start, stop and read a stopwatch-like timer for measuring precise times (better than millisecond precision).
  - Timeout: set up an interrupt to call a function after a specified delay.
  - Ticker: set up a recurring interrupt
  - All interfaces exist in a low power version
    - LowPowerTimer
    - LowPowerTimeout
    - LowPowerTicker
    - Are appropriate when microseconds precision is not required