Interrupts

An interrupts is a hardware generated function call. It is a way for an event happening in hardware to pause the processor's current program flow and do (preferably) a quick task before resuming normal program execution. It can look as the processor is doing multiple tasks simultaneously, but a single core processor can only execute one instruction at a time and the regular program is thus halted while the interrupt runs.

To make an interrupt happen in AVR the following conditions must be present:

- Global interrupts must be enabled (I-bit in SREG set - sei)
- The individual interrupt must be enabled (separate ones for each module)
- The interrupt condition must be present and/or happen.

Datasheet describes the low-level mechanics:

- Current machine instruction finishes.
- Global interrupt flag cleared (can be re-enabled for nested interrupts).
- Jump to interrupt vector with Return address in Program Counter (PC) to next instruction) pushed to stack.
- Interrupt service routine (ISR) starts, usually with a jump to the routine located elsewhere in the program memory and executes.
- A return to normal program flow with RETI instruction
    - Jumps to program address stored on stack
    - Restores global interrupt flag (I).

There is two types of interrupts; one triggered by flags and another triggered by an existing status, kind of like the RXC/TXC vs. UDRE in USART. There are internal and external interrupts, e.g. USART vs. INT0.

There is no priority in AVR; unless two interrupts occur at the same time then the one with the lowest vector (ISR address) runs first.

Variables to and from ISRs need to be (file) global and should be declared volatile if they are updated in the ISR. ISR are functions like ordinary c-functions and normal rules like local variables ceasing to exist when exiting function. Static keyword can be used to keep local variables between calls.

Interrupts are hardware specific and not defined in C so implementation will vary between different c-compilers.
avr-libc documentation under `<avr/interrupt.h>` describes implementation in AVR-GCC, functions and interrupt vectors. Vector names for a given controller can also be found in the appropriate header file. Interrupts are described in the datasheet.

GCC format on ISRs:

```
#include <avr/interrupt.h>
```

```c
// Interrupt service routine
ISR(Correct_interrupt_vect)
{
	// Code
}
```