**Interrupt System Summary** (Reference: PIC 18F252 Functional Block Diagram)

PIC 18F252 has 16 different interrupt sources (see functional block diagram). Each of these sources can be configured as either high-priority interrupt source, low-priority interrupt source or disabled (which is the reset state).

Initialization of interrupt system (to be done at the beginning of main program):

- Configure the peripheral module for interrupt generation. See the description of each module for details.
- Enable the low-priority and high-priority bi-level interrupt system.
  - RCONbits.IPEN = 1;
- For high-priority interrupt, enable all interrupts.
  - INTCONbits.GIEH = 1;
- For low-priority interrupt, enable all interrupts and enable low-priority interrupts.
  - INTCONbits.GIEH = 1;
  - INTCONbits.GIEL = 1;

When an interrupt occurs, several things happen automatically:

- The program counter contents are pushed to the stack.
- If it’s a low-priority interrupt, further low-priority interrupts are disabled.
- If it’s a high-priority interrupt, further low-priority and high-priority interrupts are disabled.
- The contents of STATUS, WREG, and BSR registers are copied into shadow registers, so that they can be restored later.
- The program counter is set to 0x0018 if it’s a low-priority interrupt, or 0x0008 for a high-priority interrupt.

Program memory locations 0x0018 (or 0x0008) should contain a goto <ISR address> instruction, so that the CPU can jump to the Interrupt Service Routine (ISR).

The ISR must clear the peripheral interrupt flag to prevent the ISR from being called again when the ISR returns. For example, INTCONbits.INT0IF = 0; must be done in the ISR while processing RB0 (INT0) interrupt input.

When the ISR ends (returns), the following things happen automatically:

- STATUS, WREG, and BSR registers are restored from the shadow registers.
- If it was a return from low-priority ISR, the low-priority interrupts are enabled.
- If it was a return from high-priority interrupt, both low-priority and high-priority interrupts are enabled.
- The program counter is restored from the stack in order to continue execution of the main program.

Assuming that both low and high-priority interrupts are enabled and configured, if the main program is interrupted by a low-priority interrupt and the control is transferred to the low-priority ISR, the low-priority ISR can itself be interrupted by a high-priority interrupt and control is further transferred to the high-priority ISR.