4.9.1 The Fetch-Decode-Execute Cycle

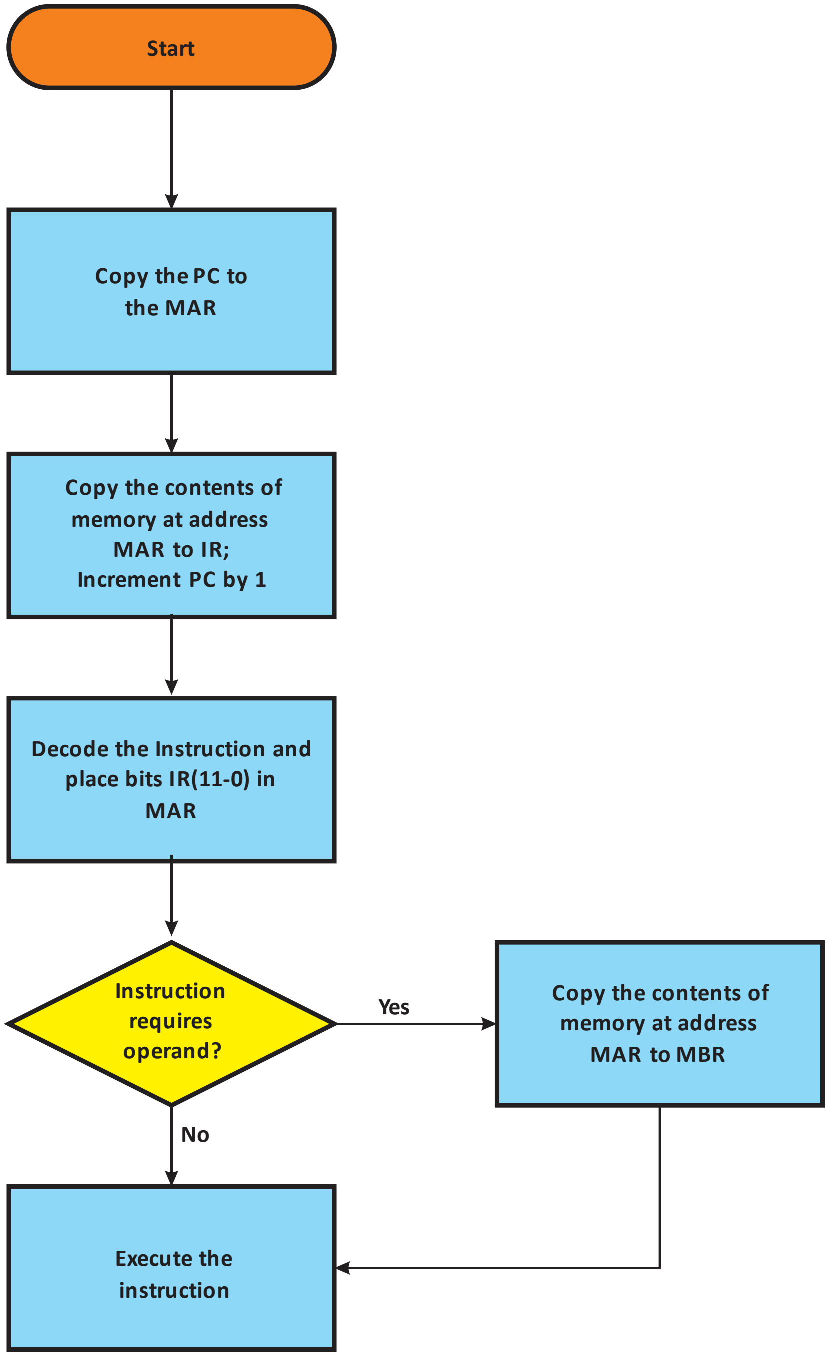

Figure 4.11 The Fetch-Decode-Execute Cycle

Load *X*

Step	RTN	Explanation
Fetch	MAR ← PC	Store the address of the next instruction in memory address register
	IR ← M[MAR]	Read the next instruction from memory and store it in the instruction register.
	PC ← PC+1	Increment the program counter to the address of the next instruction.
Decode	MAR ← IR[11-0]	Place the value of the address field in the instruction in the memory address register.
	(Decode IR[15-12])	At the same time decode
Get operand	MBR ← M[MAR]	Store the contents of memory at the address in the memory address register to the memory buffer register.
Execute	AC ← MBR	Store the value in the memory buffer register to the accumulator.

4.9.2 Interrupts and the Instruction Cycle

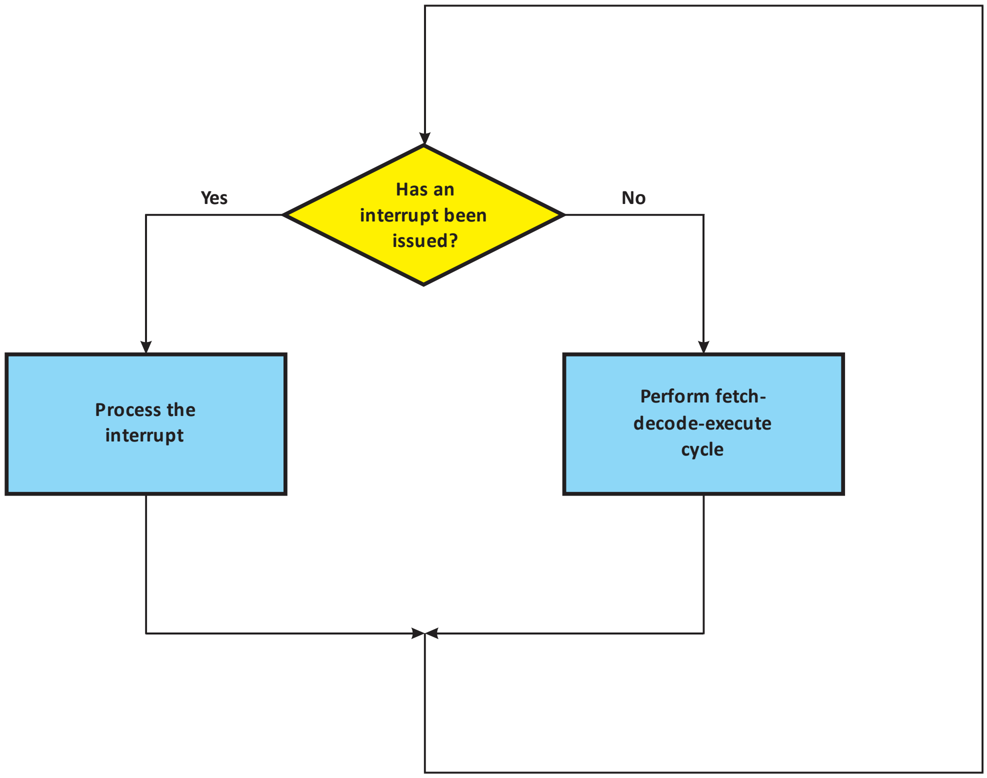

Figure 4.12 Fetch-Decode-Execute Cycle with Interrupt Checking

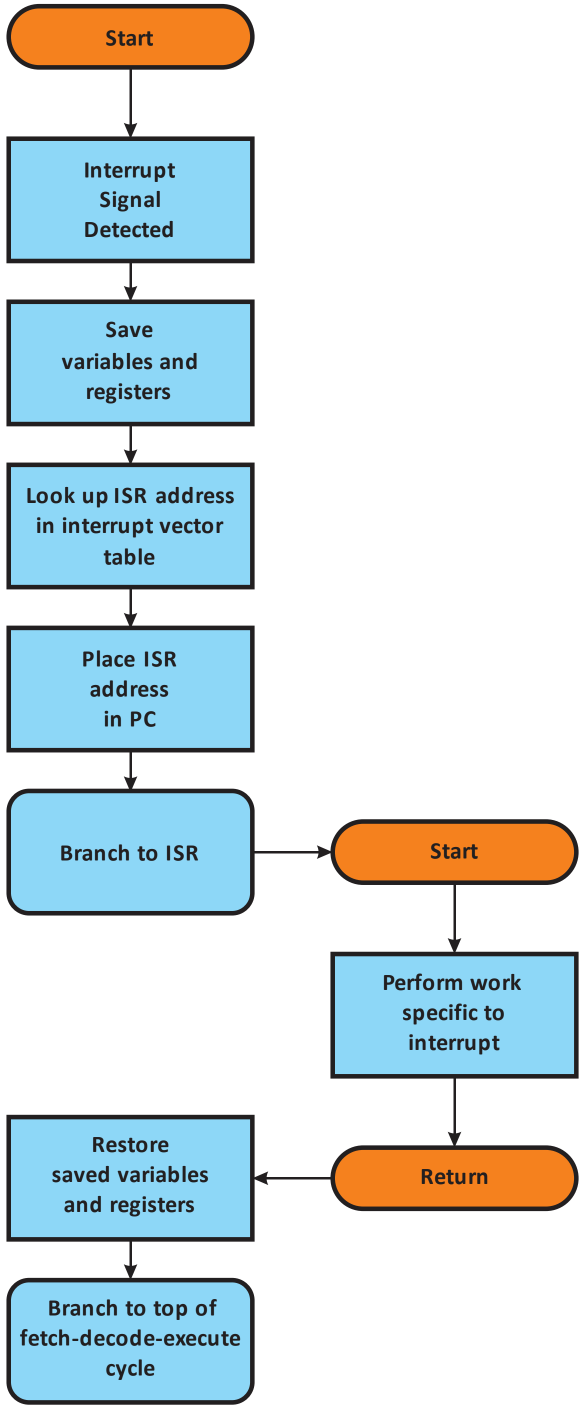

Figure 4.13 Processing and Interrupt

Term	Explanation
Interrupt	A signal or mechanism that momentarily disrupts or alters the normal fetch-decode-execute cycle.

	Examples include division by zero, stack overflow, or memory access violation.
Hardware interrupt	Hardware interrupts are efficiently used to notify the CPU that a peripheral device requires service.
Software interrupt	Called **traps** or **exceptions**. Used by various software applications. An example is a debugger.
Interrupt handler	An interrupt handler is a function that services either a software or a hardware interrupt.
Interrupt service routines	An interrupt service routine (**ISR**) is equivalent to an interrupt handler.
Interrupt vector table	An interrupt vector table contains an indexed list of addresses of interrupt handlers.
Address vector	An address vector is the starting address of an interrupt handler.
Interrupt masking	Interrupt masking is the process of suspending lower priority interrupts. For example, most systems mask keyboard interrupts when processing a page fault.
Maskable interrupt	Interrupts that can be suspended via masking are called "maskable."
Nonmaskable interrupt	A nonmaskable interrupt is one where the system would enter an unstable or unpredictable state if the interrupt were serviced.

4.9.3 MARIE's I/O

Instruction	**Action**
Input	The CPU waits for a character to be placed in the InREG.
Output	The CPU places a character in the OutREG.