Interrupt Driven I/O

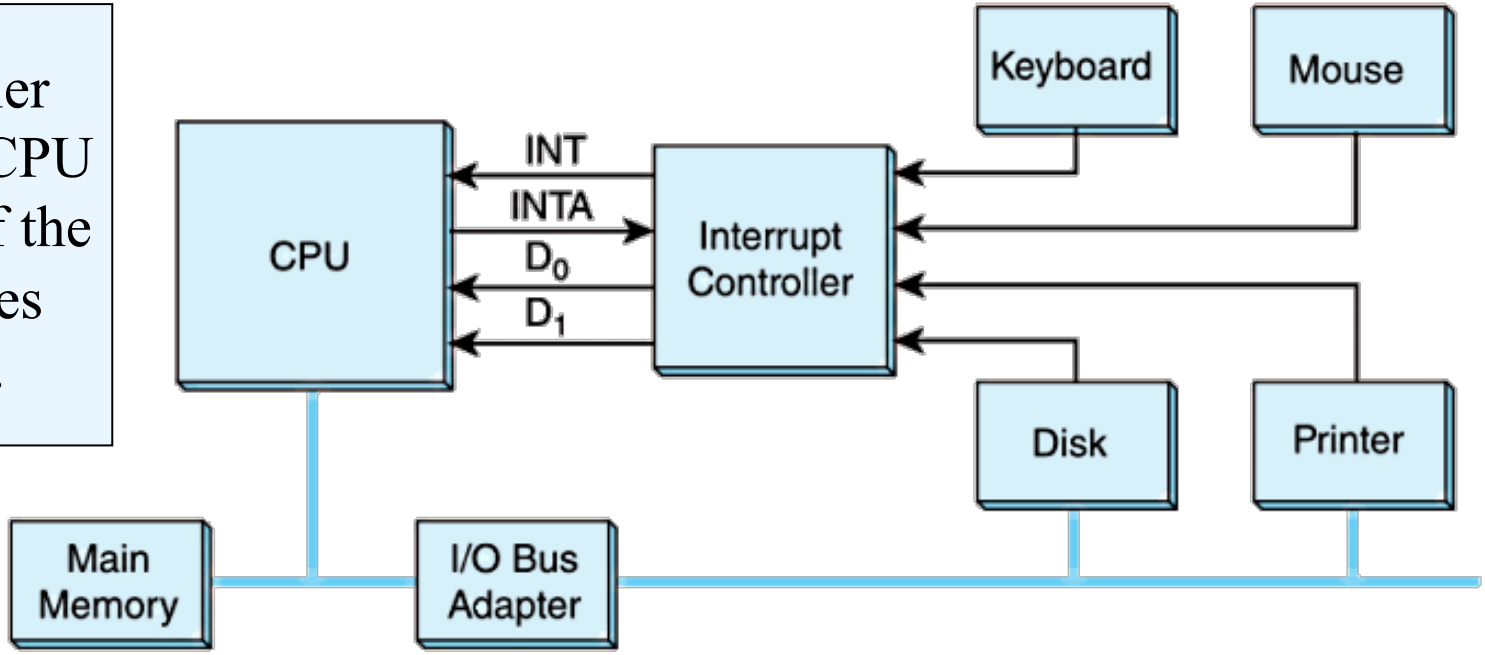
- CPU responds only when I/O request is made
  - Each device is assigned an ID
  - Each may have a different priority level
  - Interrupt signals are sent to notify the CPU of I/O completion
- Some systems employ vectored interrupts
  - Each corresponds to a different interrupt handler address
  - The Interrupt service routine (ISR) is called via the supplied address
- The MIPS uses a single vector address
  - Control goes to this vector address for all exceptions
  - MIPS interrupts are a particular type of exception

Interrupt Driven I/O

This is an idealized I/O subsystem that uses interrupts.

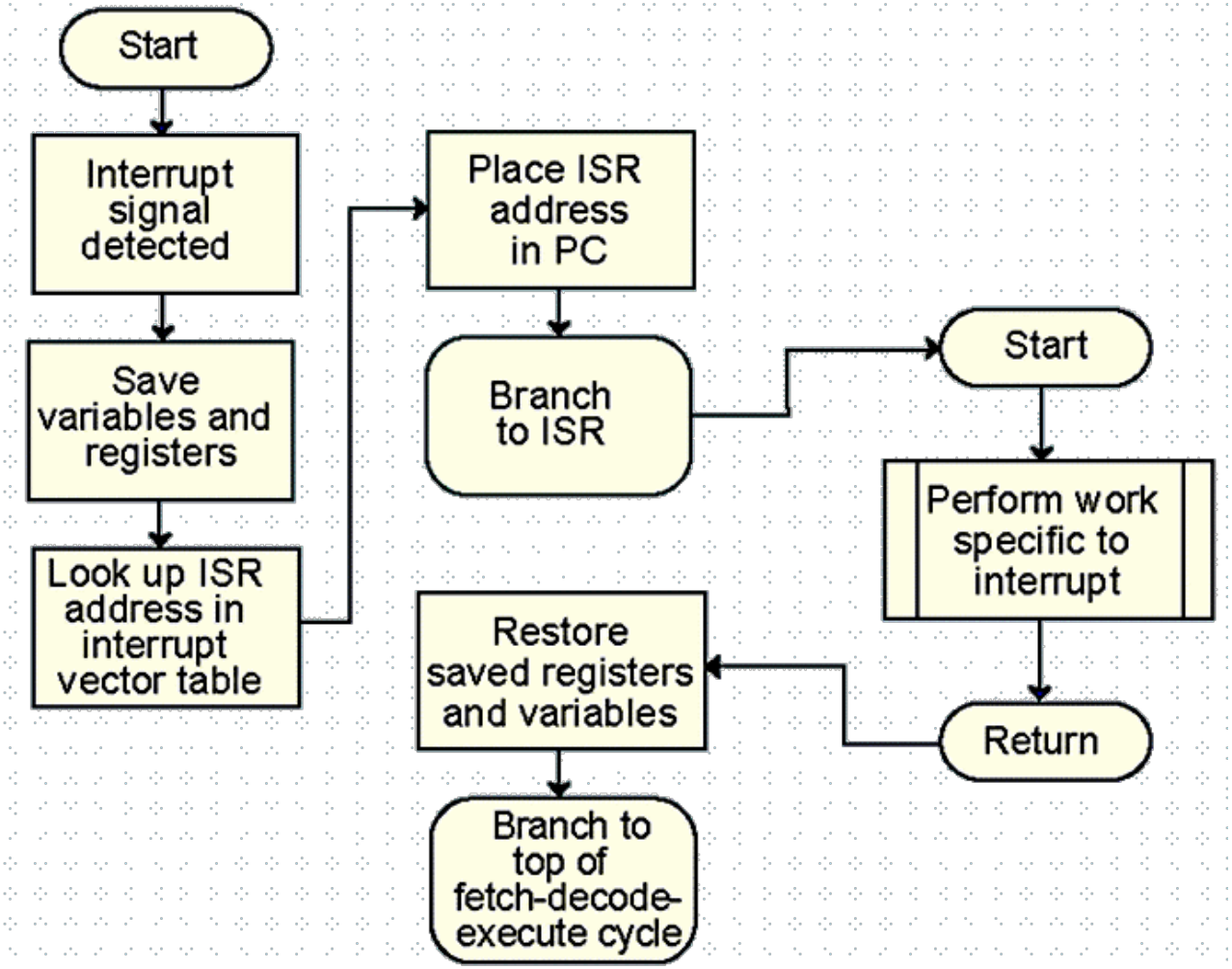
Each device connects its interrupt line to the interrupt controller.

The controller signals the CPU when any of the interrupt lines are asserted.

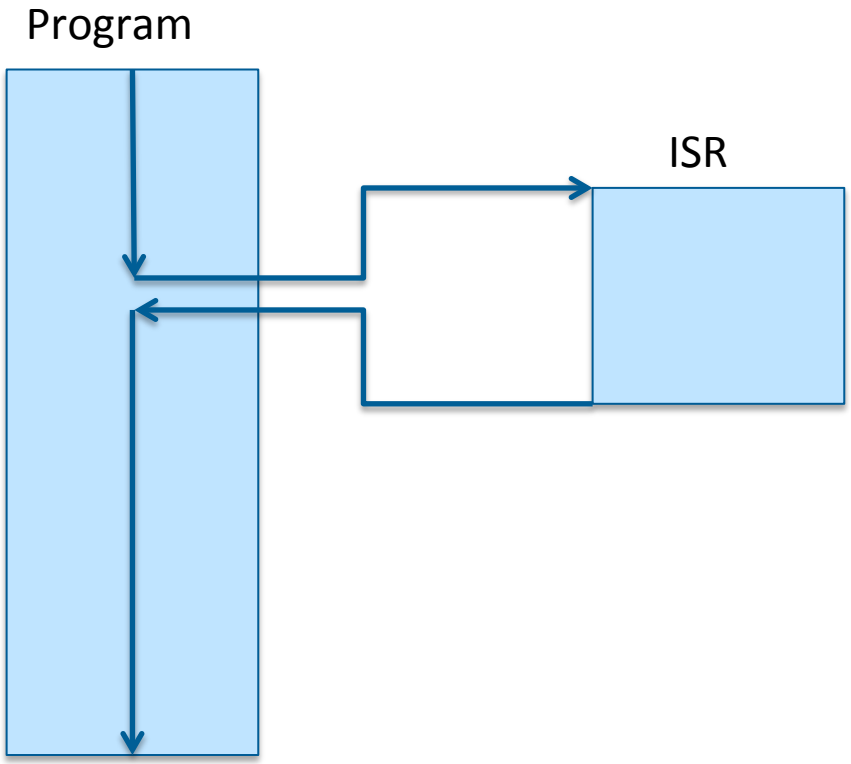


The system's state is saved before the interrupt service routine is executed and is restored afterward (e.g. all registers)

Interrupt Driven I/O



Interrupt Driven I/O

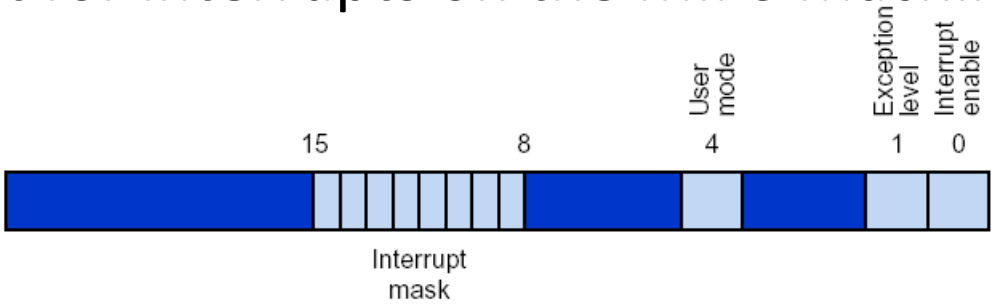


Interrupts are like unsolicited procedure calls
The program may be unaware that an interrupt occurred
Interrupts are delayed until the current instruction completes

Interrupt Driven I/O

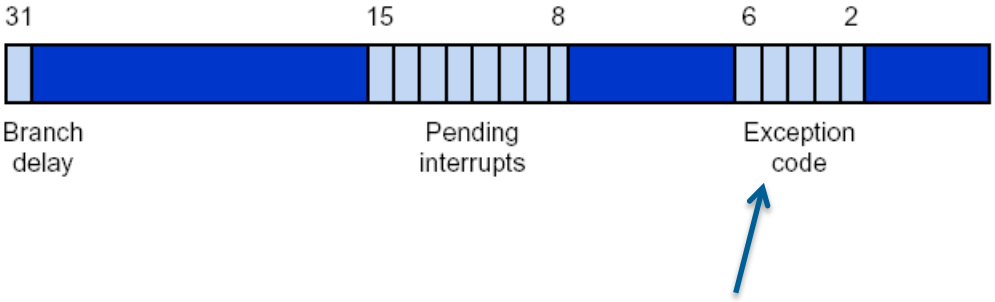
Support for interrupts on the MIPS Machine

CP0 register $12



Status Register Format

CP0 register $13

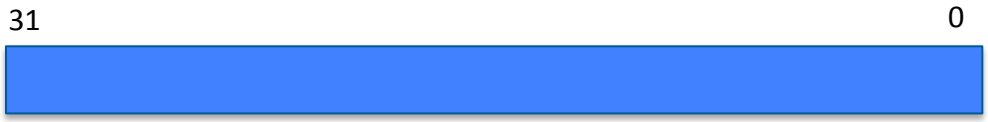


Cause Register Format

Exception code = 0 for interrupts

Non-zero value indicates other types of exceptions

CP0 register $14



EPC exception program counter

Interrupt Controller