ECE 220 Computer Systems & Programming

Lecture 6 – Interrupts & Exceptions

ECE ILLINOIS

ILLINOIS

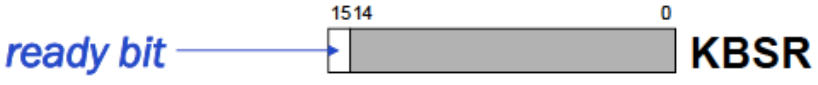
Outline

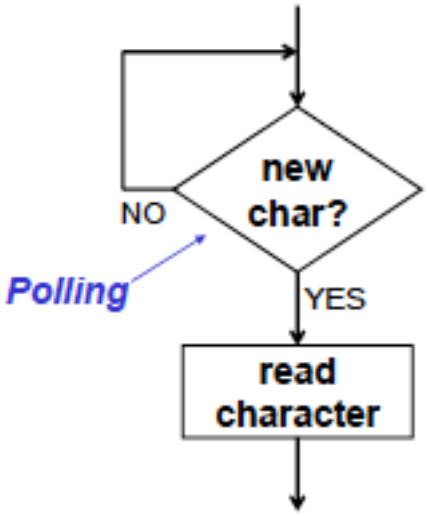
- Interrupt driven I/O
- Interrupts and exceptions
- Machinery for processing interrupts

Concepts

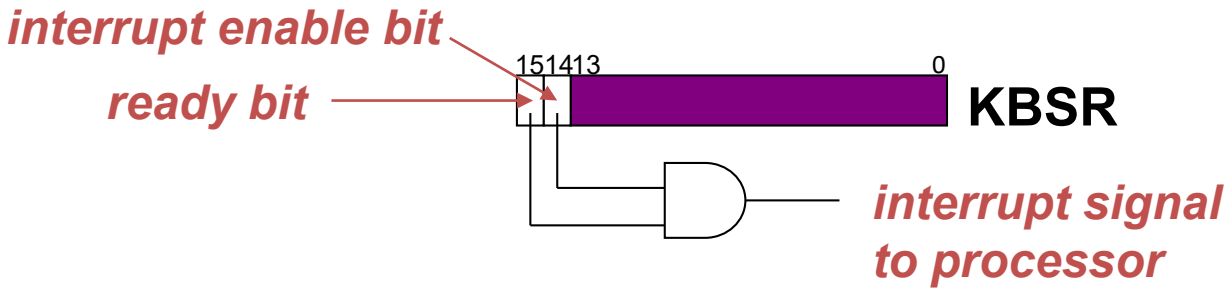
- New hardware to support new feature
- Saving processor state

Polling V.S. Interrupt Driven I/O






```
POLL   LDI  R0, KBSR
       BRzp POLL
       LDI  R0, KBDR
       ...
KBSR .FILL xFE00
KBDR .FILL xFE02
```



- Software sets "interrupt enable" bit in device register
- When Ready bit and IE bit are both set, interrupt is signaled

Interrupt Driven I/O

An I/O device can:

- **Force currently executing program to stop**
- **Have the processor carry out the need of the I/O device**
- **Resume the stopped program as if nothing had happened**

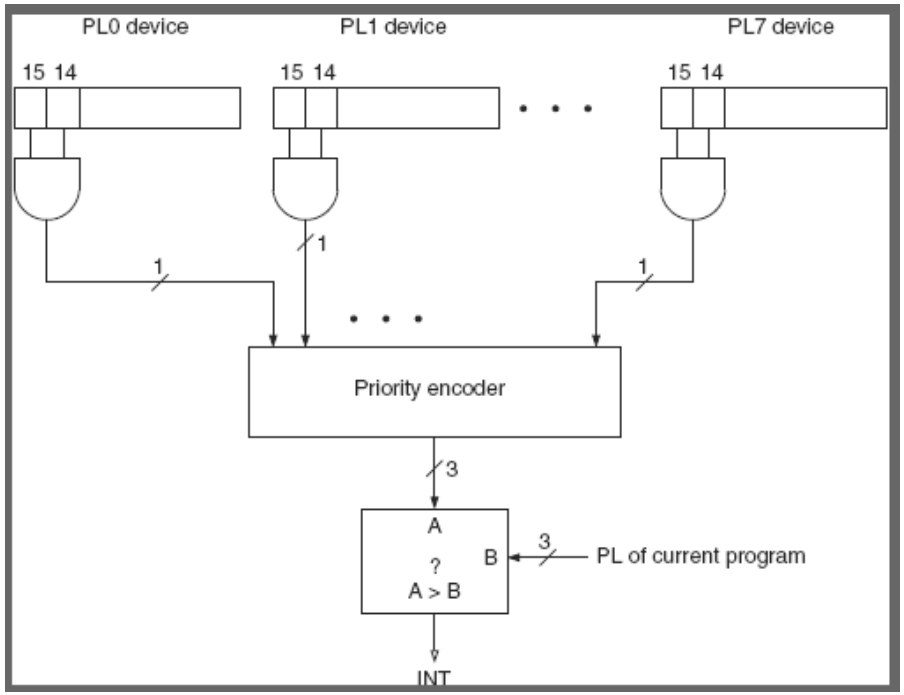
Interrupt Driven I/O: 4 Steps

1. Generation of interrupt signal (INT)
2. Set-up for servicing interrupt
3. Interrupt Service Routine
4. Resume execution main program

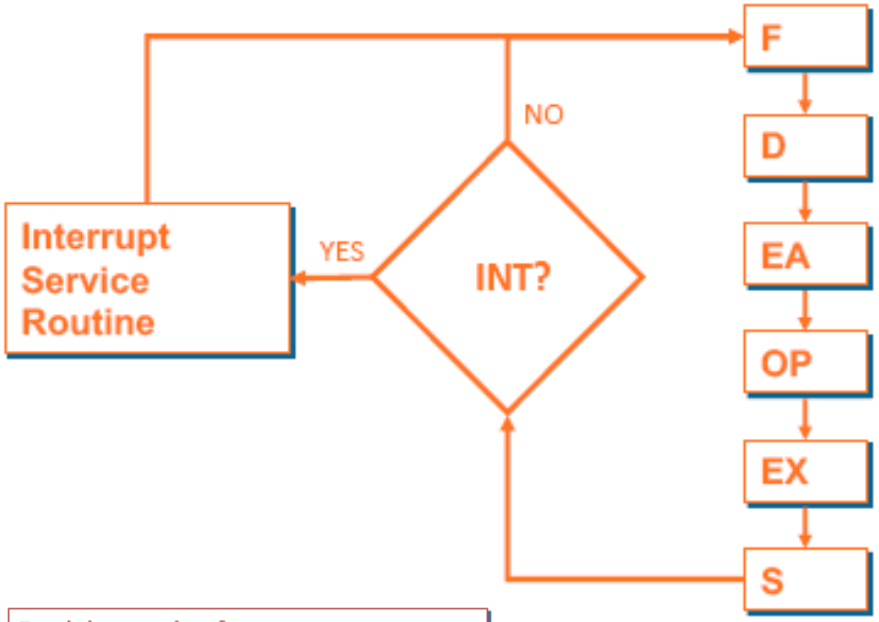
Step 1. Generation of Interrupt Signal

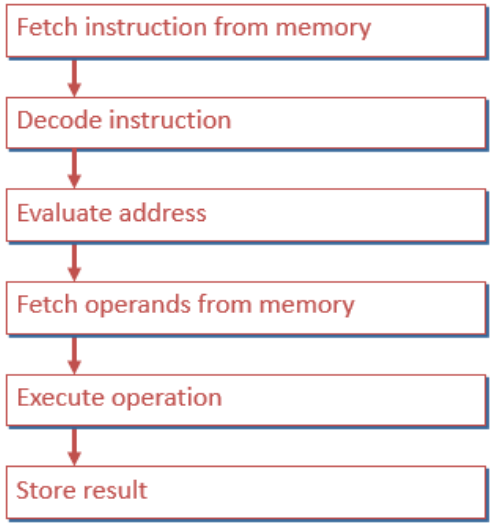
Three conditions must be true for an I/O device to interrupt the processor:

1. **The I/O device must want service (Ready Bit in KBSR)**
2. **The device must have the right to request service (IE bit in KBSR)**
3. **The device request must be more urgent than what the processor is currently doing**



Step 2: Instruction Cycle and Interrupt





- If INT is not asserted
  - The usual instruction cycle
  - Ends with check for interrupt
- If INT is asserted, control unit does 2 things before fetching the next instruction:
  - Store state of the program
  - Service the interrupt

Processor Status Register (PSR)

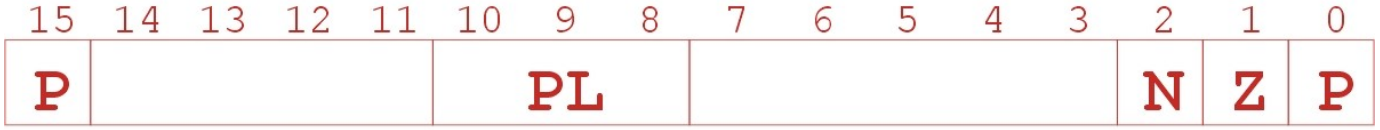
- **Enough state info saved for interrupted program to resume later**
- **Enough state info loaded for the interrupt service routine to begin service**

State of a Program:

- Contents of all GPRs (should be saved by ISR, callee-saved)
- **PC**
- **PSR (processor status register)**
- **User Stack Pointer**

PSR[15] – privileged (supervisor - 0) or unprivileged (user - 1) mode

PSR[10:8] – priority level, PSR[2:0] – condition code



Supervisor Stack (where the state is saved)

Suppose the user program uses a stack, and the stack pointer is stored in R6

A special region of memory used as the stack (why?) for interrupt service routines.

- **Supervisor Stack Pointer (SSP) stored in Saved.SSP**
- **Another register for storing User Stack Pointer (USP) Saved.USP**

When switching from User mode to Supervisor mode (as result of interrupt), save R6 to Saved.USP

Access both Stacks using R6 as the stack pointer.

2. Invoking the Service Routine

I/O device transmits Interrupt Vector (INTV, 8-bit) along with interrupt signal and priority level (PL0..PL7). When an interrupt is taken:

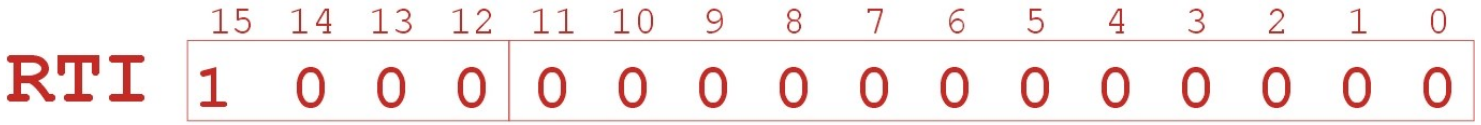
1. If Priv = 1 (user), i.e. PSR[15] =1
   Saved.USP = R6, then R6 = Saved.SSP.
2. Push PSR and PC to Supervisor Stack.
3. Set PSR[15] = 0 (supervisor mode).
4. Set PSR[10:8] = priority of interrupt being serviced. (keyboard PL4)
5. Set PSR[2:0] = 0.
6. Set MAR = x01vv, where vv = 8-bit interrupt vector
   provided by interrupting device (e.g., keyboard = x80).
7. Load memory location (M[x01vv]) into MDR.
8. Set PC = MDR; now first instruction of ISR will be fetched.

Step 3: Interrupt Service Routine

- PC contains the starting address of the ISR.
- Callee-save for general purpose registers.
- Execute code for servicing the interrupt
- There may be other interrupts during this execution (in which case go back to Step 2)

Step 4: Returning from interrupt

Special instruction (RTI) – restores state



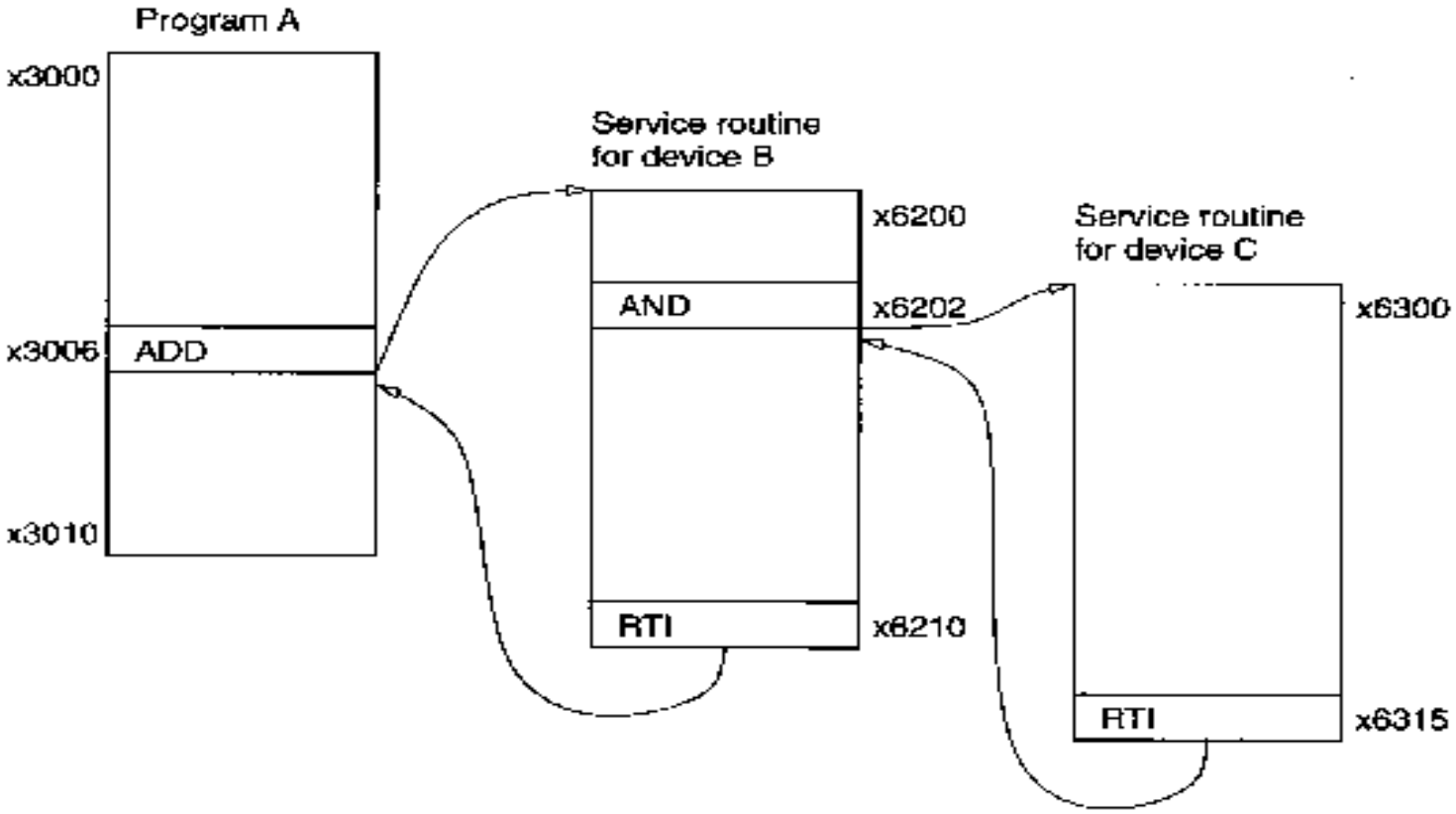
1. Pop PC from supervisor stack. (PC = M[R6]; R6 = R6 + 1)
2. Pop PSR from supervisor stack. (PSR = M[R6]; R6 = R6 + 1)
3. If PSR[15] = 1, R6 = Saved.USP.
   (If going back to user mode, need to restore User Stack Pointer.)

RTI is a privileged instruction.

- Can only be executed in Supervisor Mode.
- If executed in User Mode, causes an exception.

Nested Interrupts

Example:



Interrupt vector table

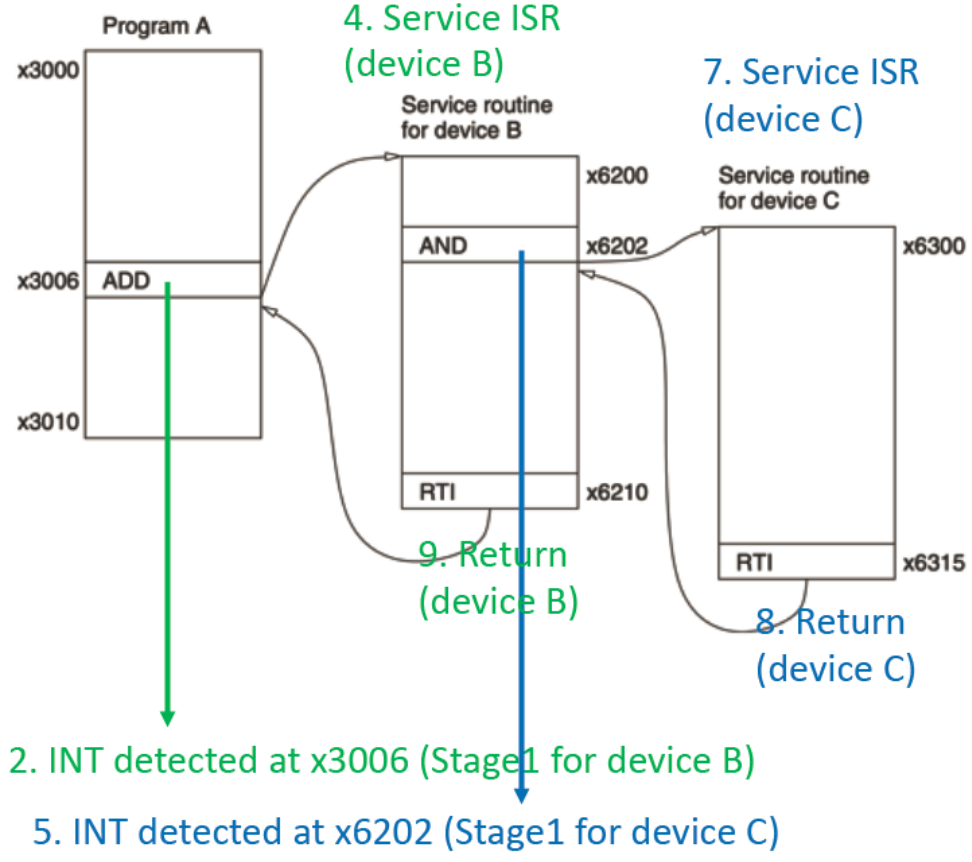
Addr	Data
x01F1	x6200
x01F2	x6300

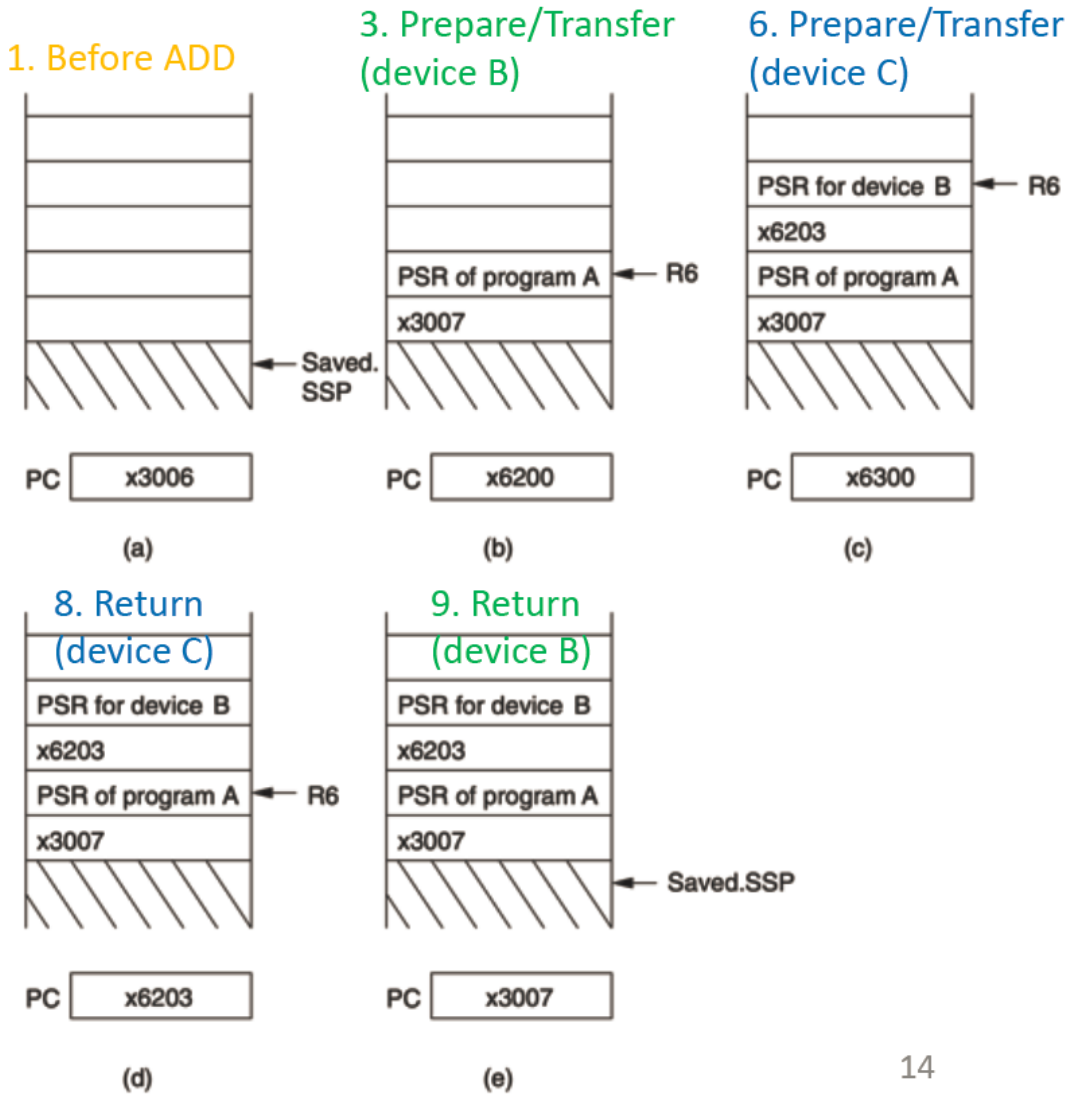
INTV

Device B = xF1

Device C = xF2

PL: A<B<C





Example Code: (See Github)

Exceptions: Internal Interrupt

When something unexpected happens *inside* the processor, it may cause an exception.

Examples of Exception in LC-3:

- **Privileged operation (e.g., RTI in user mode)**
- **Executing an illegal opcode (Bits[15:12] = 1101)**

Handled just like an interrupt

- **Vector is determined internally by type of exception**
- **Priority is the same as running program**

Interrupt Vector Table

Exception Service Routines – x0100 to x017F

Interrupt Service Routines – x0180 to 01FF