000000000 01010100 30011100 00002020 20202E4F 52494720 20207833 3030300A E0001300 00002020 20204C45 41202052 1C3015C0 794C696E 6509E200 13000000 20202020 4C454120 2052312C 206D794C 696E6540 60001600 00004C4F 4F502020 52205230 2C205231 2C202330 21F00010 00000020 20202020 20202054 52415020 78323105 24001400 00002020 20204C44 20205232 2C207465 726D8014 00160000 00202020 20202020 20414444 2052322C 20523002 20202020 00002020 20202020 20204252 7A201254 (F506 12 0015 000 02)2020 20202020 20414444 2052312C 2052312C 00120000 00202020 20202020 2042110 1002110 4F456 015 F00000 00005354 4F502020 20204841 4C54D0FF 04001000 2031F90F Lecture x0019 - 04/25 00746572 6D202020 202E4649 00746100 00010000 00324000 00010000 00010000 00627200 00010000 Recap & Interrupts in LC3 00010000 00666100 000100 00323000 00010000 00010000 00300000 002A0000 00636500 202E5354 52494E47 SA202020 20226974 61627261 68324066 6132332D 65636532 32302200 00000000

ECE 220 - Spring 2024 Dr. Ivan Abraham



ECE 220 - Spring 2024Dr. Ivan Abraham



• Trees

ECE 220 - Spring 2024Dr. Ivan Abraham



- Trees
 - Concepts & properties



- Trees
 - Concepts & properties
 - Traversals: preorder/ inorder/postorder



- Trees
 - Concepts & properties
 - Traversals: preorder/ inorder/postorder
 - Binary search trees



- Trees
 - Concepts & properties
 - Traversals: preorder/ inorder/postorder
 - Binary search trees
 - C to LC3 (structs, trees, linked lists)



- Trees
 - Concepts & properties
 - Traversals: preorder/ inorder/postorder
 - Binary search trees
 - C to LC3 (structs, trees, linked lists)
 - Code posted



Trees ${\color{black}\bullet}$

Reminders

- Concepts & properties
- Traversals: preorder/ inorder/postorder
- Binary search trees
- C to LC3 (structs, trees, linked lists)
 - Code posted



• Trees

Reminders

- Concepts & properties
- Traversals: preorder/ inorder/postorder
- Binary search trees
- C to LC3 (structs, trees, linked lists)
 - Code posted

Quizzes



Trees

- Reminders
- Concepts & properties
- Traversals: preorder/ inorder/postorder
- Binary search trees
- C to LC3 (structs, trees, linked lists)
 - Code posted

- Quizzes
- Programming competition



- Trees
 Reminders
 - Concepts & properties
 Qu
 - Traversals: preorder/ inorder/postorder
 - Binary search trees
 - C to LC3 (structs, trees, linked lists)
 - Code posted

- Quizzes
- Programming competition
- ICES forms available



Reminders Trees ${\color{black}\bullet}$ Concepts & properties Quizzes Traversals: preorder/ Programming competition inorder/postorder • ICES forms available • Binary search trees • C to LC3 (structs, trees, • +1% extra credit linked lists) Code posted



ECE 220 - Spring 2024Dr. Ivan Abraham



• Change gears & talk about *interrupts* in LC3



- Change gears & talk about interrupts in LC3
- Recall: KBSR/KBDR & DSR/DDR?



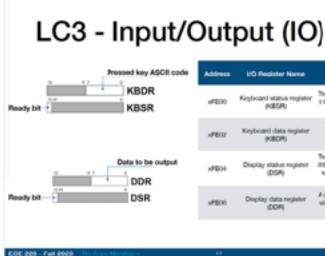
- Change gears & talk about interrupts in LC3
- Recall: KBSR/KBDR & DSR/DDR?
 - When did we see these acronyms?



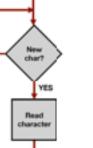
- Change gears & talk about interrupts in LC3
- Recall: KBSR/KBDR & DSR/DDR?
 - When did we see these acronyms?
 - How did we implement I/O in LC3 before?



- Change gears & talk about *interrupts* in LC3
- Recall: KBSR/KBDR & DSR/DDR?
 - When did we see these acronyms?
 - How did we implement I/O in LC3 before?







; If ready 1 read KBOR 1 KBSR .FILL KEDR .FILL

ORIG x3000 ;Create a 1 check KBSR

/If ready b

Address	UO Register Name	VD Register Function
HFE00	Keyboard status register (KBSP)	The ready bit (bit[15]) indicates i the keyboard has received a new character
xF802	Keyboard data register (KBDR)	Bits (7.0) contain the last character typed on the keyboard
×FEH	Display status register (DSP)	The ready bit (bit(15)) indicates if the display device is ready to receive another character to print on the screen
×F806	Display data register (DDP)	A character written in bits [7:0] vill be displayed displayed on the screen

LC3 - Input from keyboard

Basic routine

Handshaking is performed using KBSR & KBDR

- · When user presses a key
- Its ASCII code is placed in KBDR[0:7]
- KBSR[15] is set to 1 (ready bit)
- · Keyboard is disabled, i.e., any further keypress is ignored
- When KBDR is read by CPU
- KBSF[15] is set to 0
- Keyboard is enabled

LC3 - Input from keyboard

	.08IG x1000
loop to	KPOLL LDI R1, KBSR
	BREP KPOLL
	LDI RO, KBDR
it unset	; Other instruction
	1
dt pot,	4 -
nto R0	HALT
	KBSR .FILL xFE00
xFE00	KBOR .FILL xFE02
xFE02	.END

LC3 - Display to console

Basic routine

Handshaking is performed using DSR & DDR

- When display is ready to present a character
- DSR[15] is set to 1 (ready bit)
- When a new character is written to DDR
- DSR[15] is set to 0
- Any other chars written to DDR are ignored
- DDR[7:0] is displayed

Slides from Lecture #1



ECE 220 - Spring 2024 Dr. Ivan Abraham



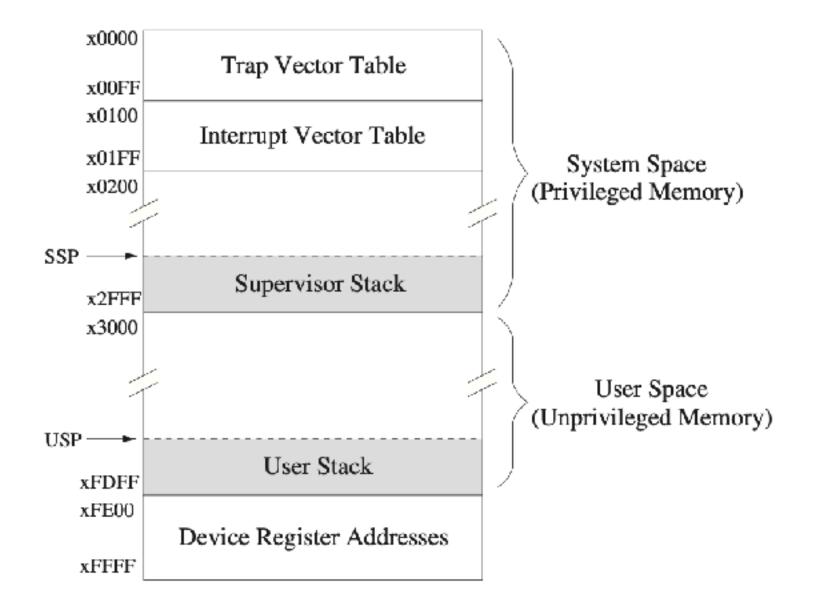


Figure A.1 - P&P 3rd Ed.



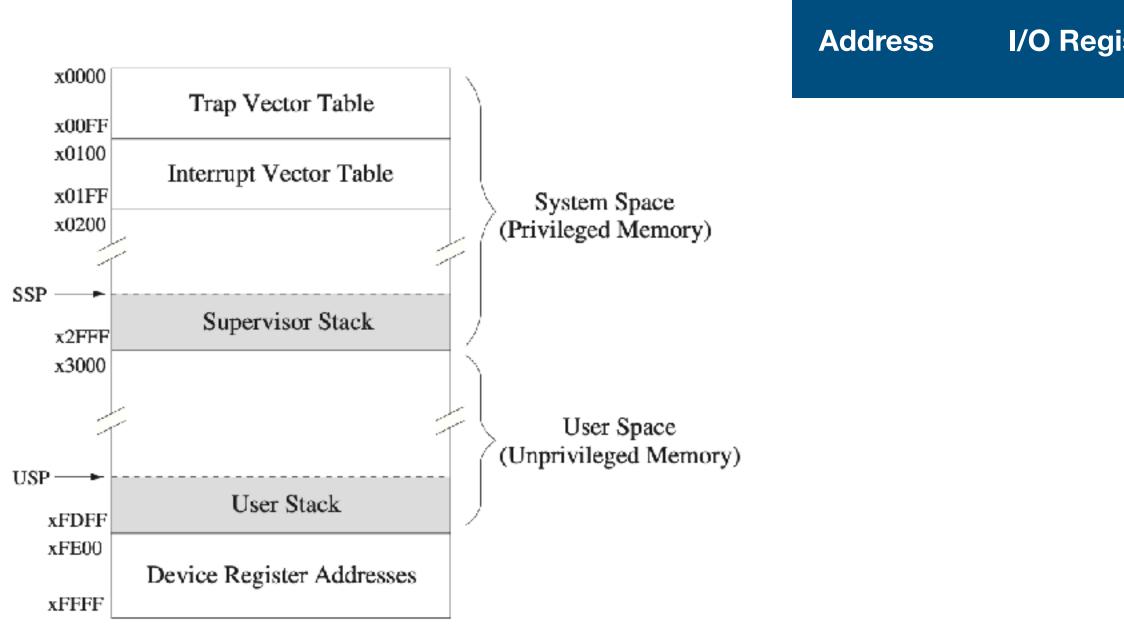


Figure A.1 - P&P 3rd Ed.

I/O Register Name

I/O Register Function



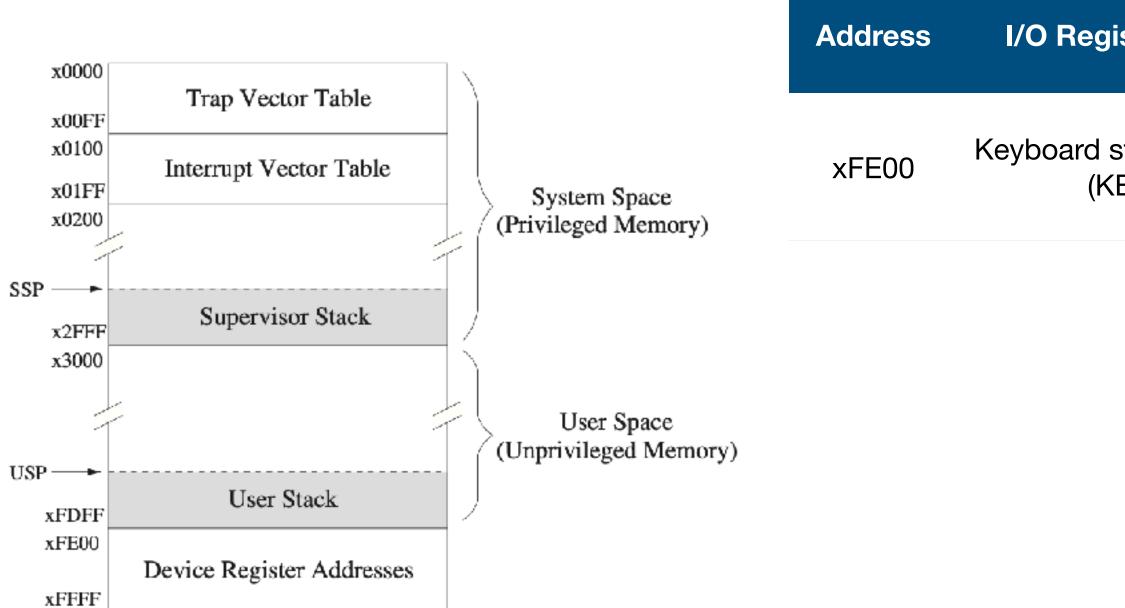


Figure A.1 - P&P 3rd Ed.

I/O Register Name

I/O Register Function

Keyboard status register (KBSR) The ready bit (bit[15]) indicates if the keyboard has received a new character



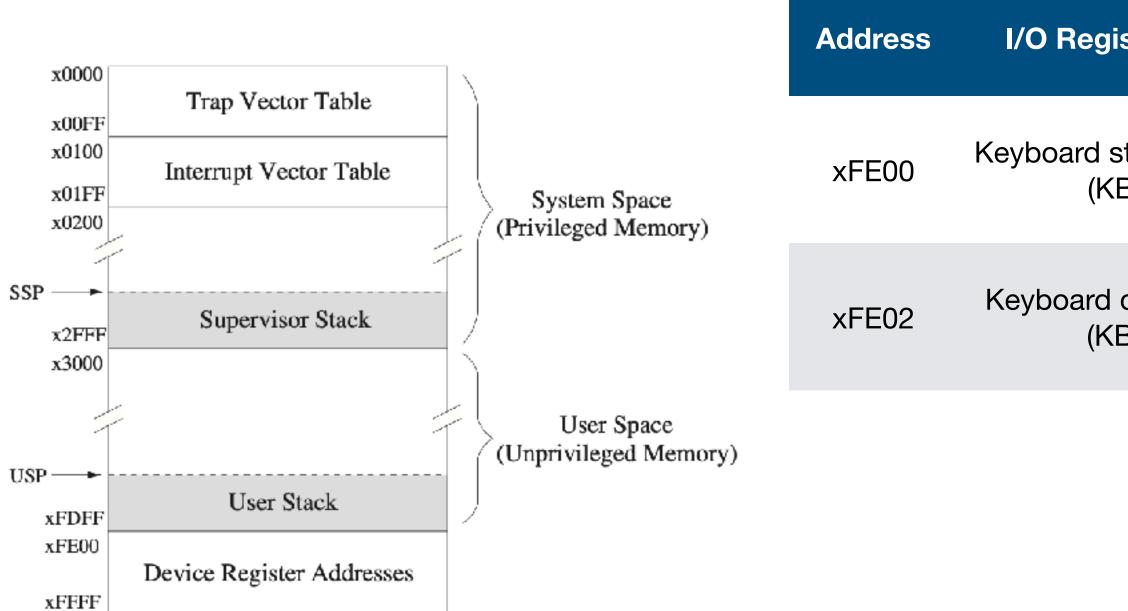


Figure A.1 - P&P 3rd Ed.

ister Name	I/O Register Function
status register (BSR)	The ready bit (bit[15]) indicates if the keyboard has received a new character
data register (BDR)	Bits [7:0] contain the last character typed on the keyboard



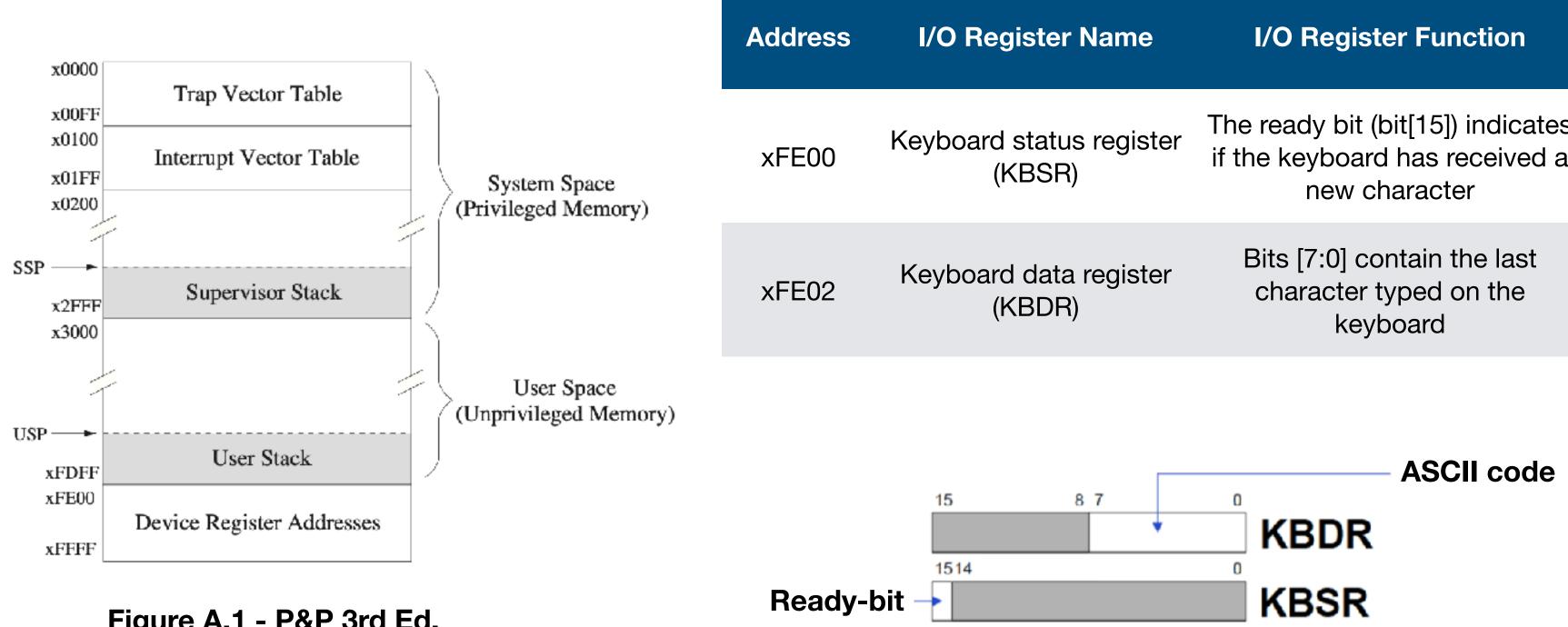


Figure A.1 - P&P 3rd Ed.

ister Name	I/O Register Function
status register (BSR)	The ready bit (bit[15]) indicates if the keyboard has received a new character
data register (BDR)	Bits [7:0] contain the last character typed on the keyboard



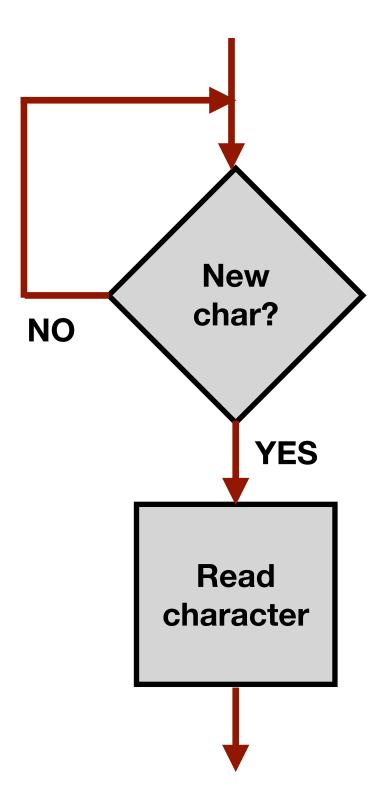
ECE 220 - Spring 2024 **Dr. Ivan Abraham**



Any problems with polled I/O? New char? NO YES Read character

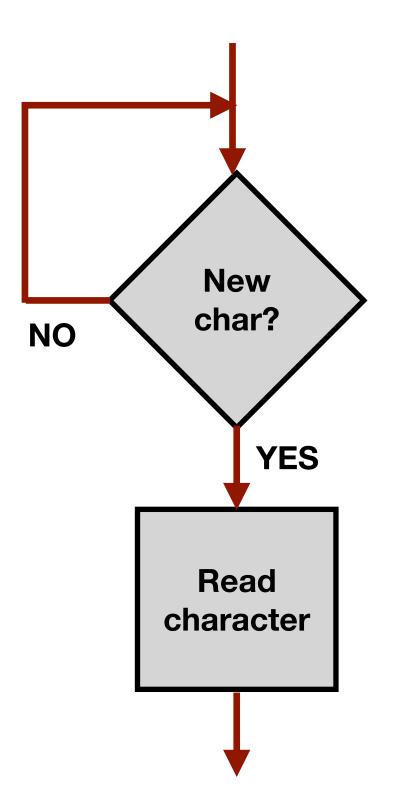
ECE 220 - Spring 2024 Dr. Ivan Abraham





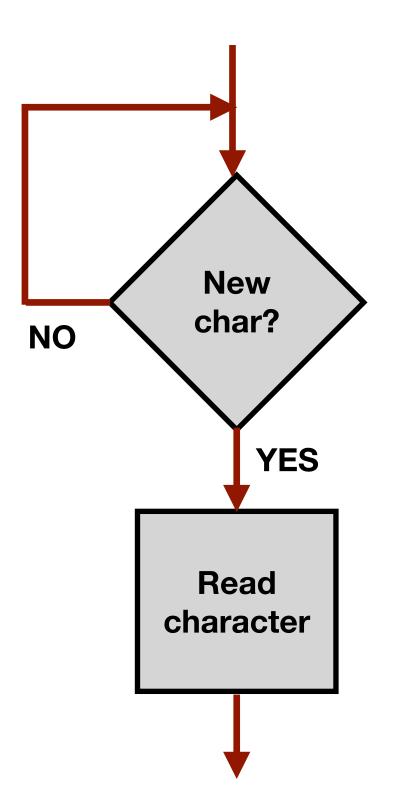
• Suppose we want to type 100 characters:





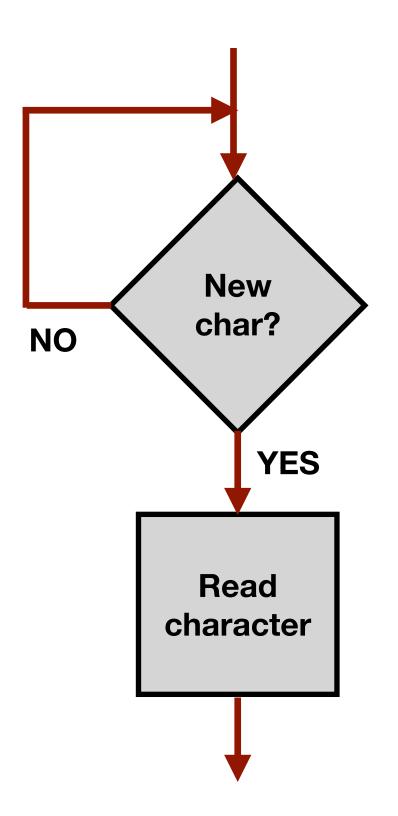
- Suppose we want to type 100 characters: • User can input at speed of 60 WPM, assume,
- eight characters/word:





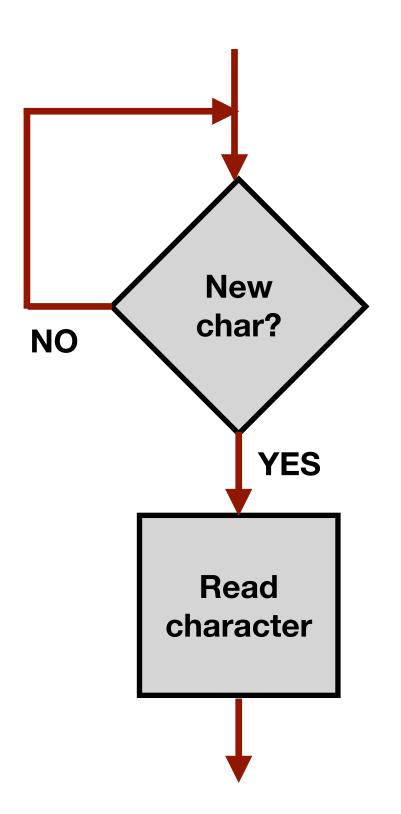
- Suppose we want to type 100 characters:
 - User can input at speed of 60 WPM, assume, eight characters/word:
 - How long to type 100 characters?





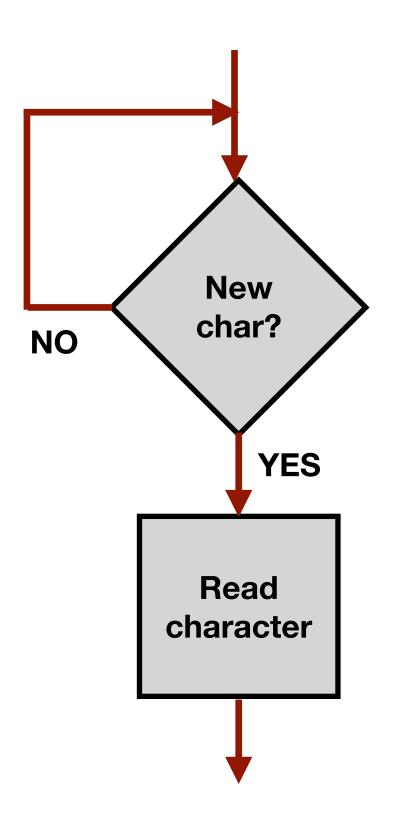
- Suppose we want to type 100 characters:
 - User can input at speed of 60 WPM, assume, eight characters/word:
 - How long to type 100 characters?
- Processor cannot do anything else while it waits on *next* character!





- Suppose we want to type 100 characters:
 - User can input at speed of 60 WPM, assume, eight characters/word:
 - How long to type 100 characters?
- Processor cannot do anything else while it waits on *next* character!
 - 12.5 seconds spend just polling!





- Suppose we want to type 100 characters:
 - User can input at speed of 60 WPM, assume, eight characters/word:
 - How long to type 100 characters?
- Processor cannot do anything else while it waits on *next* character!
 - 12.5 seconds spend just polling!
- How can we free up the processor to do other tasks?



Interrupt driven I/O

ECE 220 - Spring 2024 Dr. Ivan Abraham



Interrupt driven I/O

• Key idea: An I/O device can



Interrupt driven I/O

- Key idea: An I/O device can
 - force running program to stop



Interrupt driven I/O

- Key idea: An I/O device can
 - force running program to stop
 - have processor execute different program that carries out needs of I/O device, and then



Interrupt driven I/O

- Key idea: An I/O device can
 - force running program to stop
 - have processor execute different program that carries out needs of I/O device, and then
 - resume execution of stopped program as if nothing had happened



Interrupt driven I/O

- Key idea: An I/O device can
 - force running program to stop
 - have processor execute different program that carries out needs of I/O device, and then
 - resume execution of stopped program as if nothing had happened

Figure 9.17

```
Program A is executing instruction n
  Program A is executing instruction n+1
  Program A is executing instruction n+2
1: Interrupt signal is detected
1: Program A is put into suspended animation
1: PC is loaded with the starting address of Program B
2: Program B starts satisying I/O device's needs
2: Program B continues satisfying I/O device's needs
2: Program B continues satisfying I/O device's needs
2: Program B finishes satisfying I/O device's needs
3: Program A is brought back to life
  Program A is executing instruction n+3
  Program A is executing instruction n+4
```

Instruction execution flow for interrupt-driven I/O.



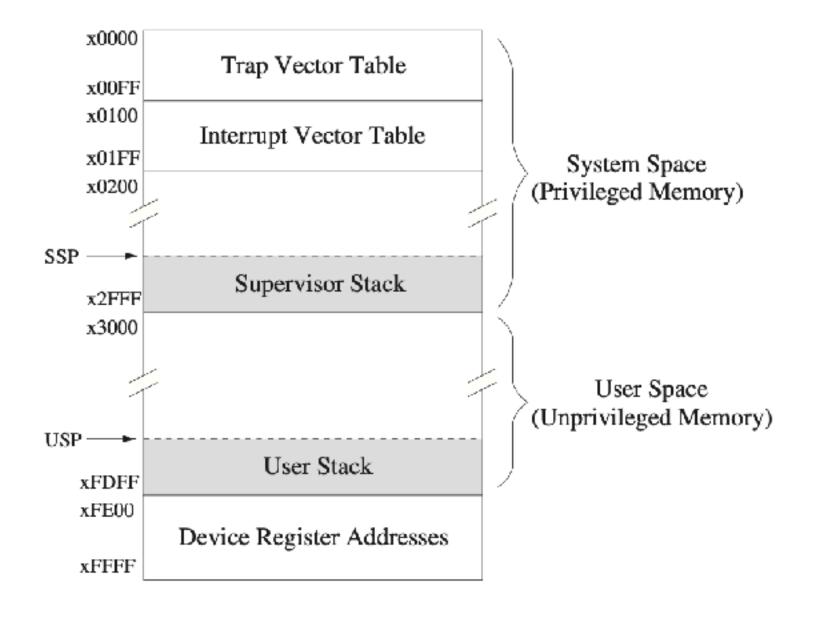
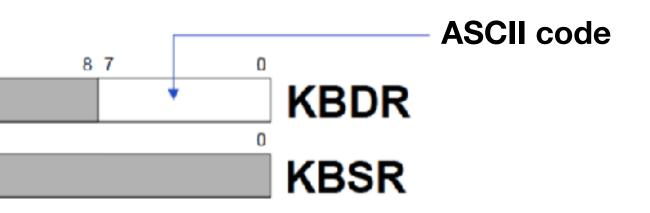


Figure A.1 - P&P 3rd Ed.

15

15 14





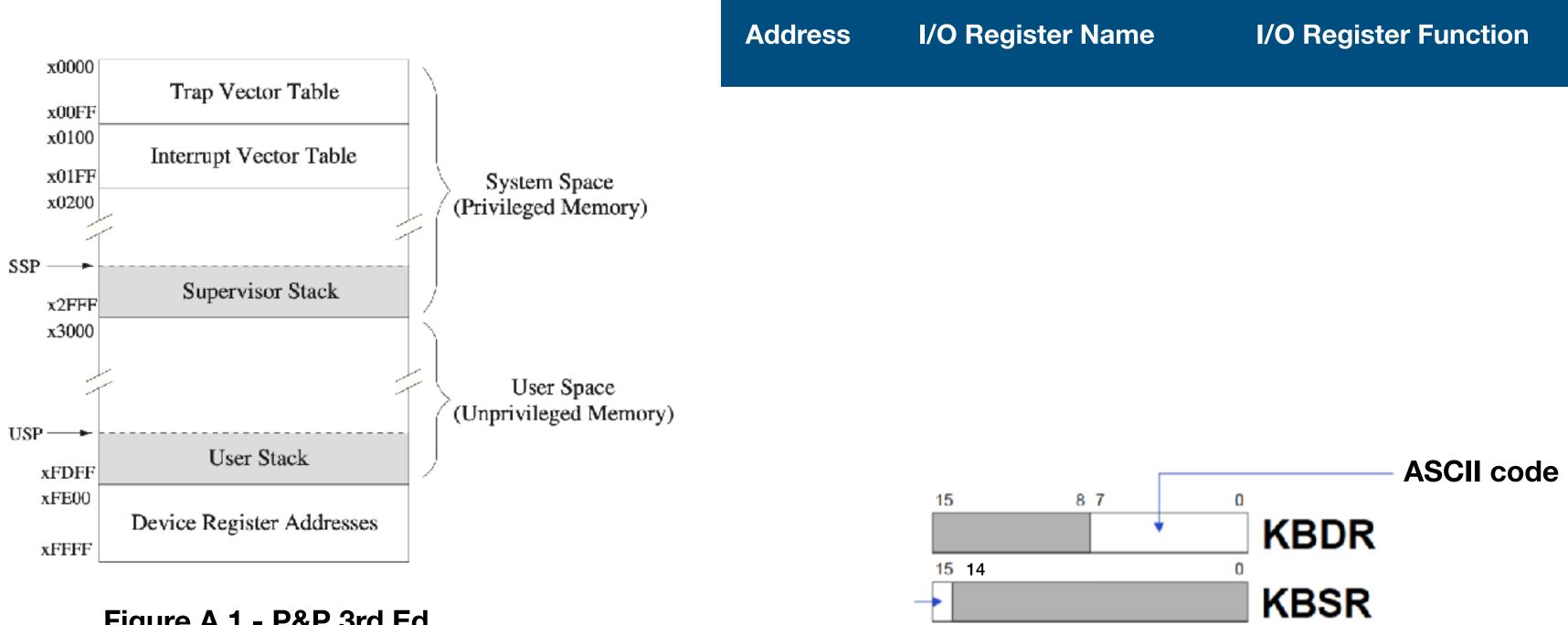


Figure A.1 - P&P 3rd Ed.



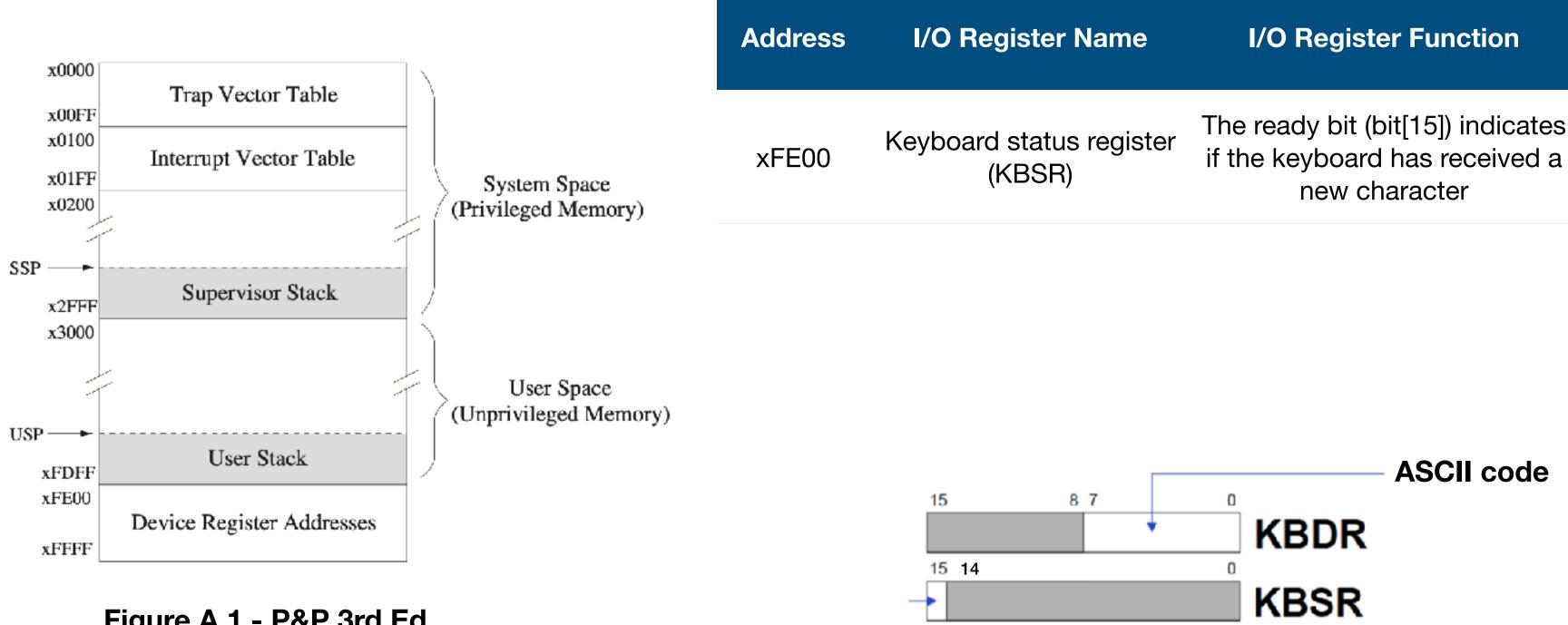
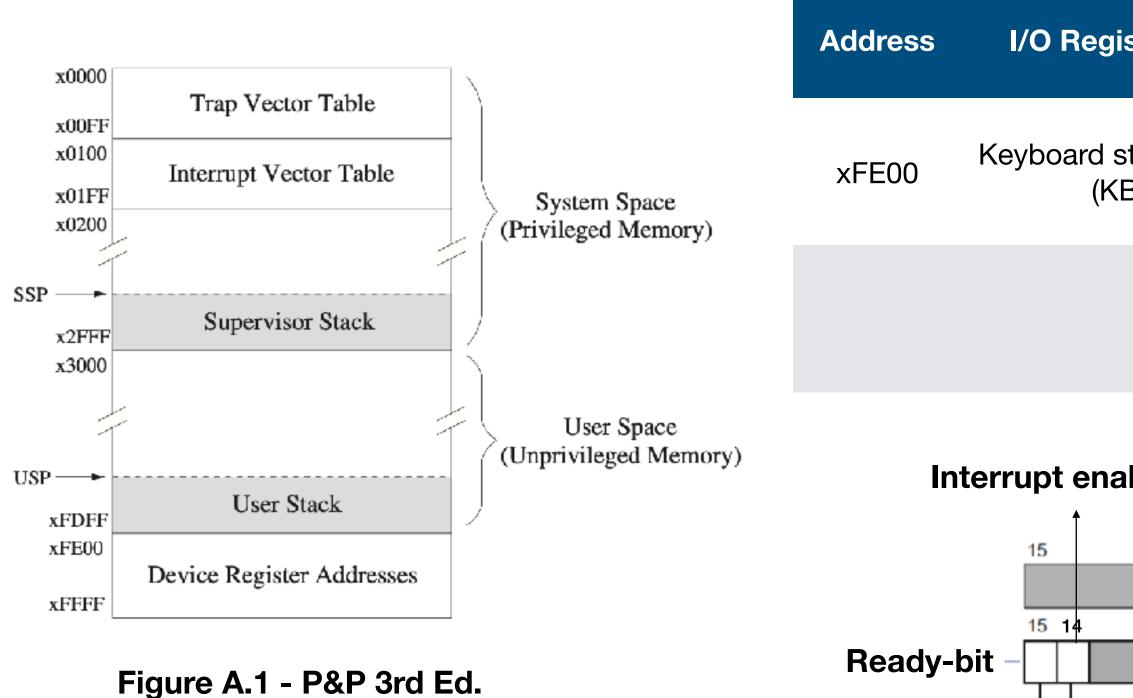


Figure A.1 - P&P 3rd Ed.





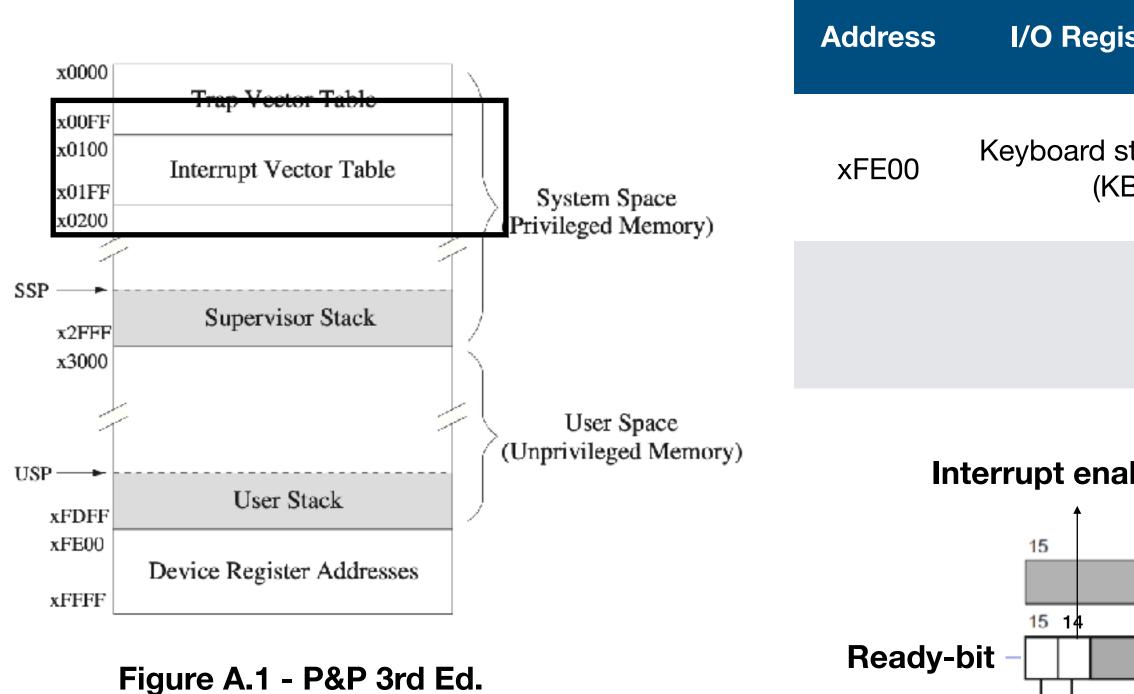
jister Name	I/O Register Function
status register (BSR)	The ready bit (bit[15]) indicates if the keyboard has received a new character
	The IE bit (bit[14]) specifies if the I/O device has permission to generate interrupt signal
able bit 8 7	ASCII code

KBSR

0

Interrupt signal





jister Name	I/O Register Function
status register (BSR)	The ready bit (bit[15]) indicates if the keyboard has received a new character
	The IE bit (bit[14]) specifies if the I/O device has permission to generate interrupt signal
able bit 8 7	ASCII code

KBSR

0

Interrupt signal



ECE 220 - Spring 2024 Dr. Ivan Abraham



For interrupt driven I/O to work:



For interrupt driven I/O to work:

1. The I/O device must want service.



For interrupt driven I/O to work:

1. The I/O device must want service.

• Ready bit



For interrupt driven I/O to work:

- 1. The I/O device must want service.
 - Ready bit
- 2. The device must have the **right to request** the service.



For interrupt driven I/O to work:

- 1. The I/O device must want service.
 - Ready bit
- 2. The device must have the **right to request** the service.
 - Interrupt enable bit



- For interrupt driven I/O to work:
 - 1. The I/O device must want service.
 - Ready bit
 - 2. The device must have the **right to request** the service.
 - Interrupt enable bit

3. The device request must be **more urgent** than what the processor is currently doing.



- For interrupt driven I/O to work:
 - 1. The I/O device must want service.
 - Ready bit
 - 2. The device must have the **right to request** the service.
 - Interrupt enable bit

3. The device request must be **more urgent** than what the processor is currently doing.

Priority levels: PL0 < PL1
 < ... < PL6



ECE 220 - Spring 2024 Dr. Ivan Abraham

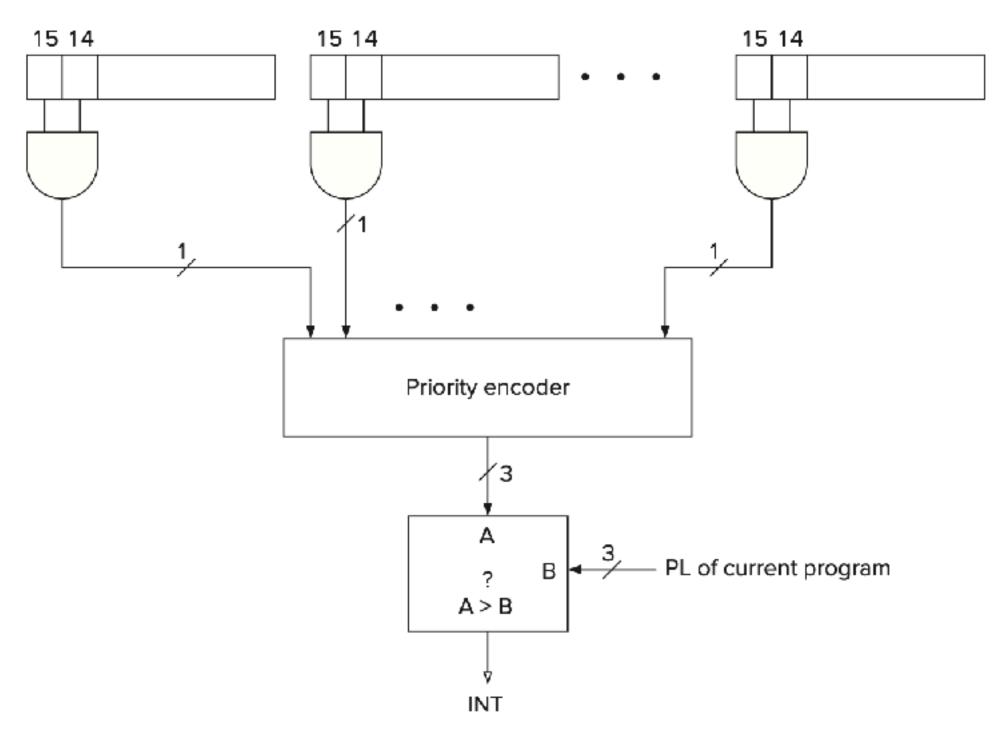


 All devices asserting an interrupt signal are compared



PL0 device

- All devices asserting an interrupt signal are compared
- Highest PL request is compared to current program's PL



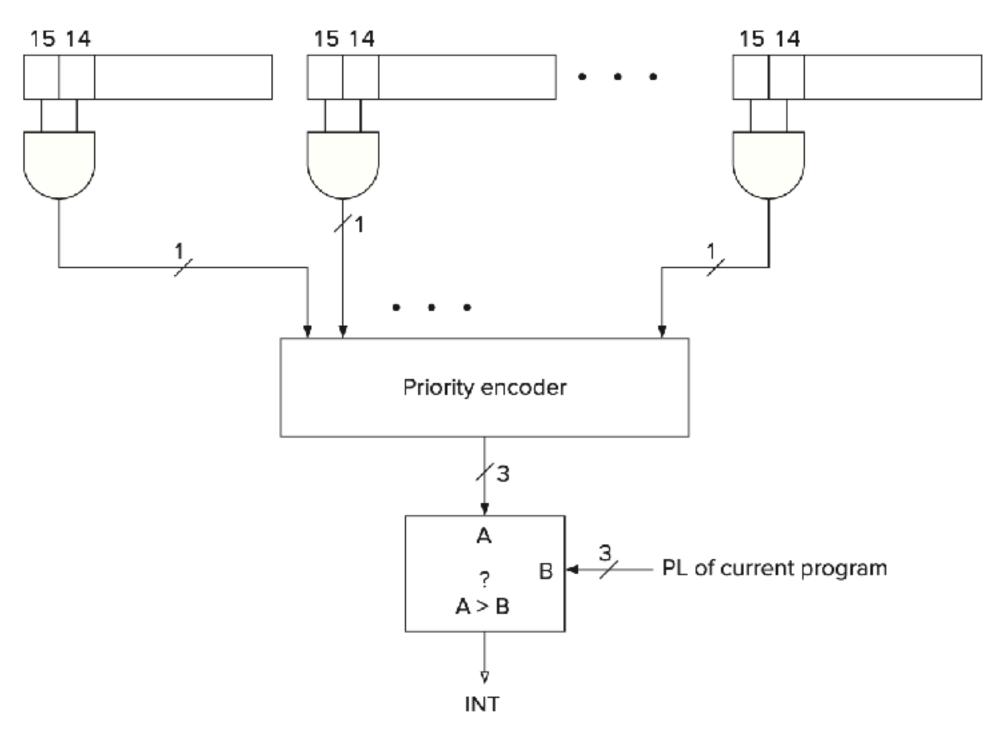
PL1 device

PL7 device



PL0 device

- All devices asserting an interrupt signal are compared
- Highest PL request is compared to current program's PL
 - Interrupt is initiated.



PL1 device

PL7 device



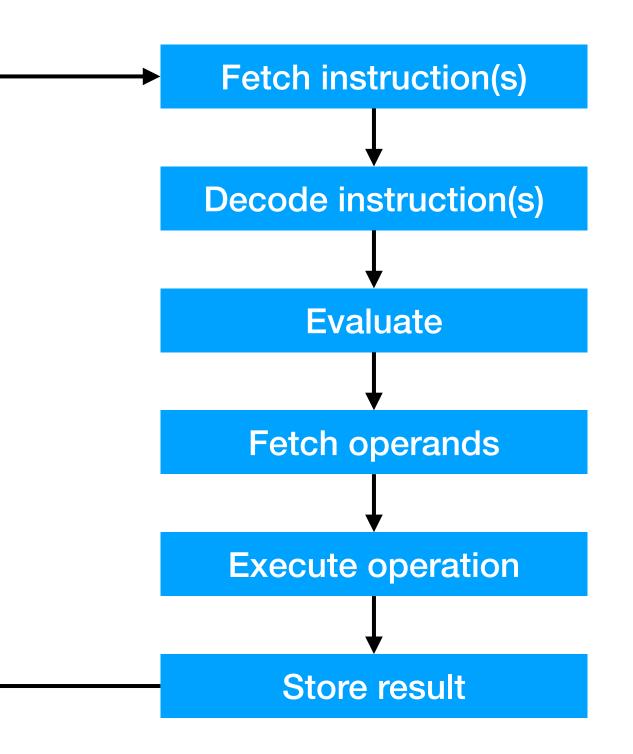
ECE 220 - Spring 2024 Dr. Ivan Abraham



• Recall instruction cycle.

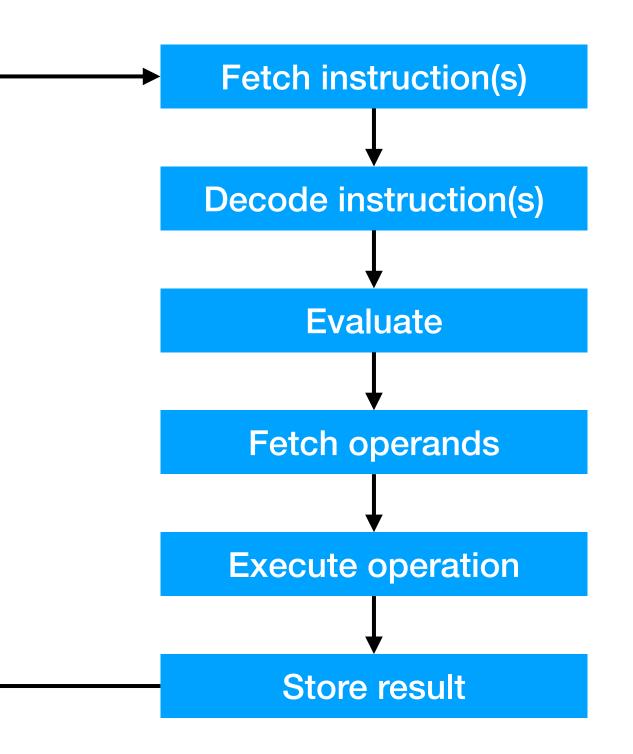


• Recall instruction cycle.

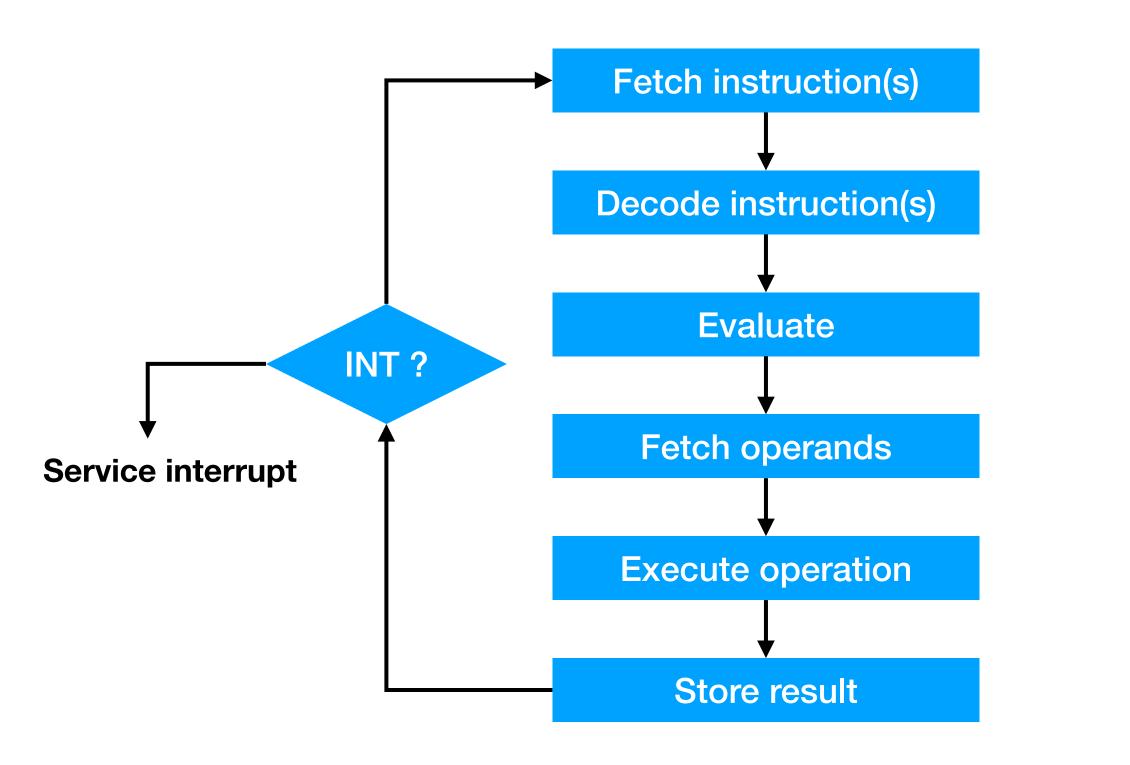




- Recall instruction cycle.
- Processor will check if INT is asserted at the end of each instruction cycle before starting a new FETCH

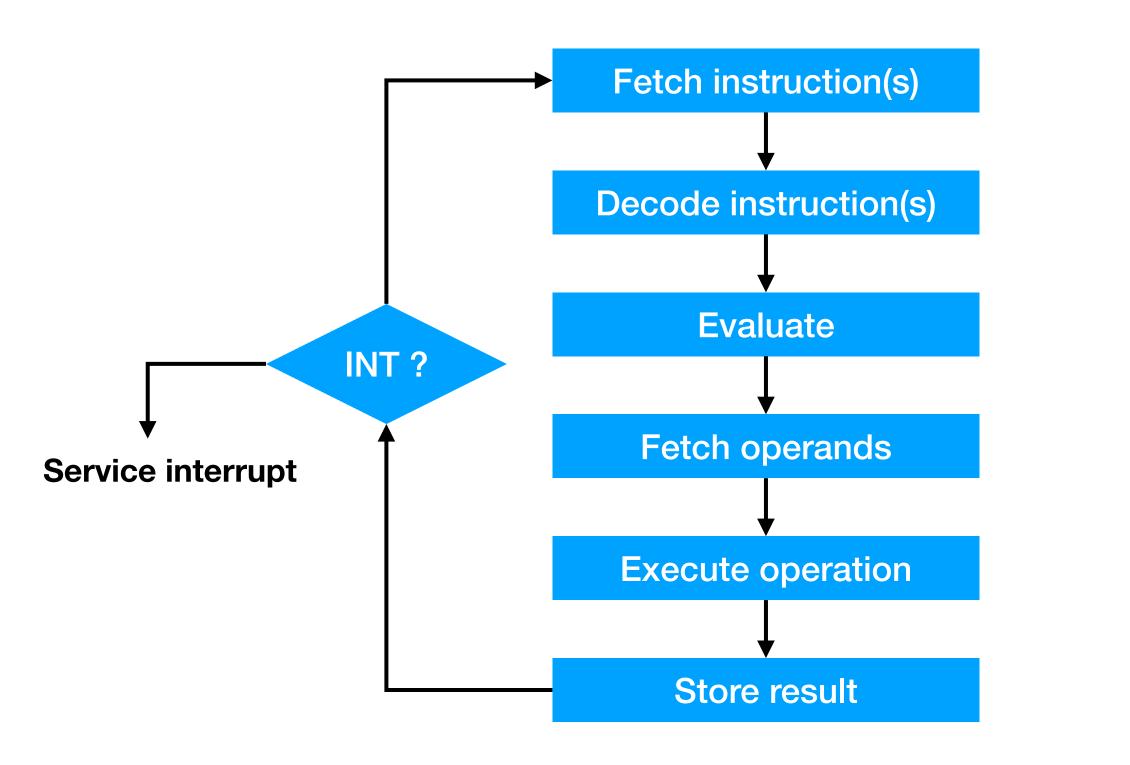






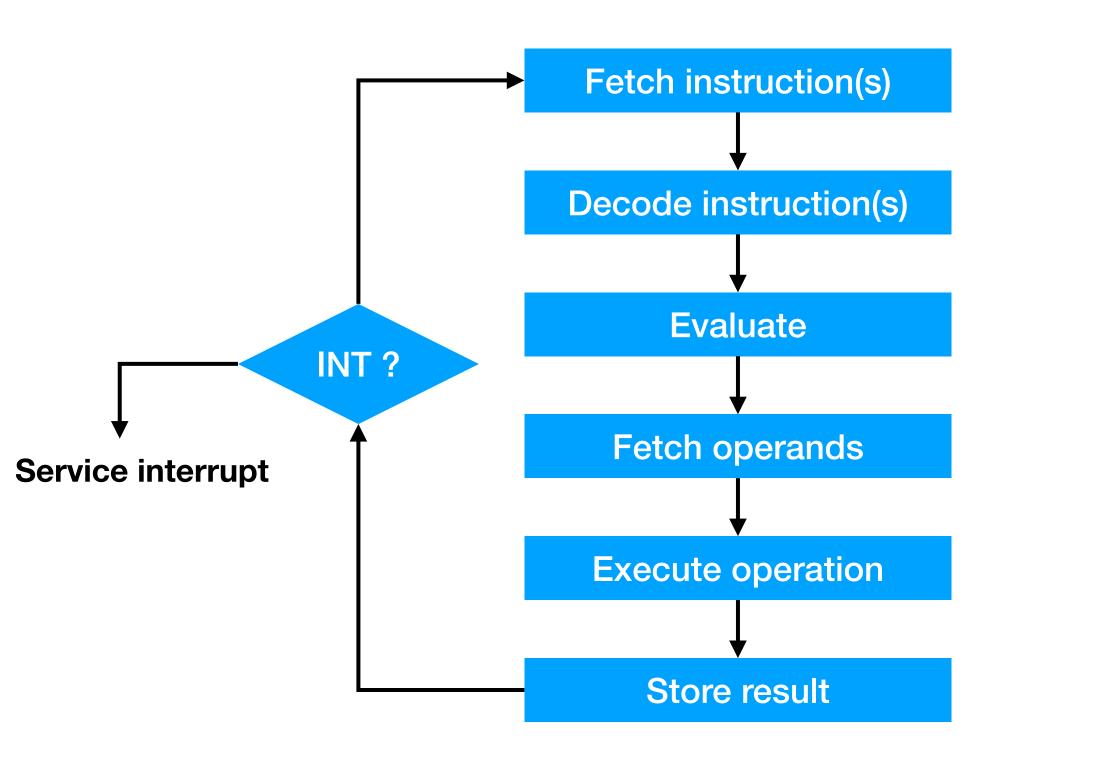


• Control unit needs to:



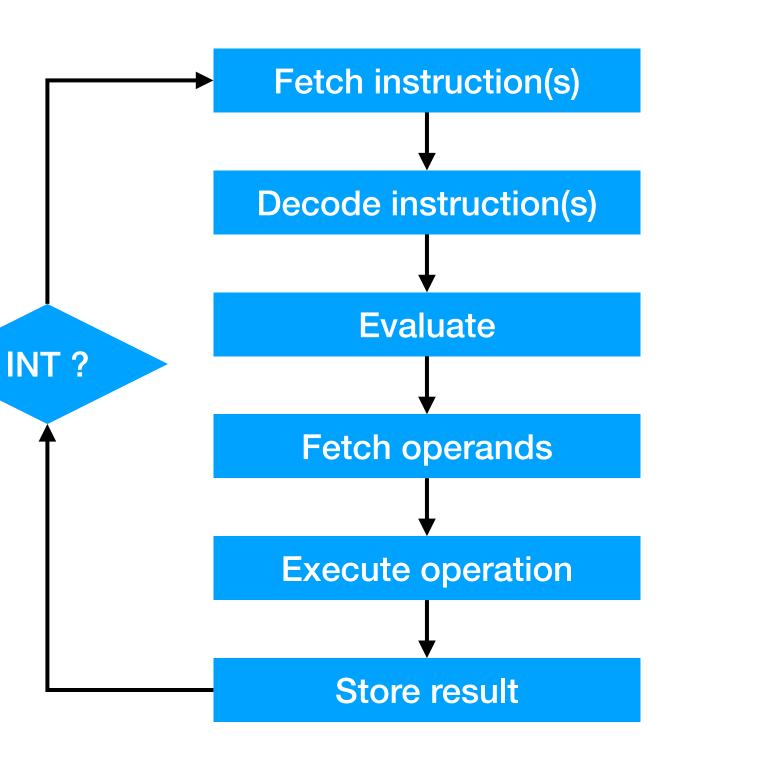


- Control unit needs to:
 - save state information to be able to resume interrupted program



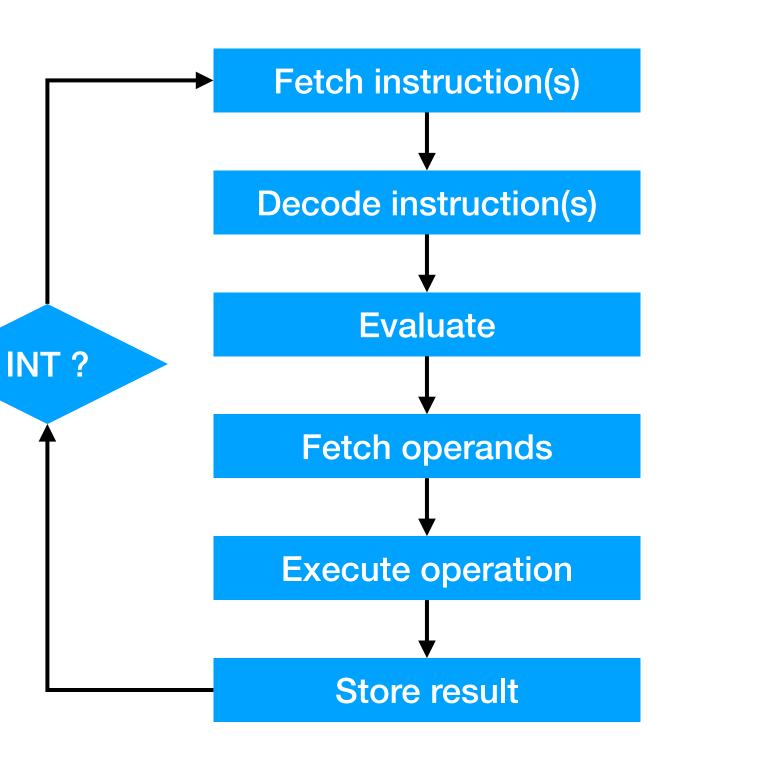


- Control unit needs to:
 - save state information to be able to resume interrupted program
 - load PC with the starting address of ISR (interrupt service routine)





- Control unit needs to:
 - save state information to be able to resume interrupted program
 - load PC with the starting address of ISR (interrupt service routine)
 - resume old process on RTI (coming later)





ECE 220 - Spring 2024 **Dr. Ivan Abraham**



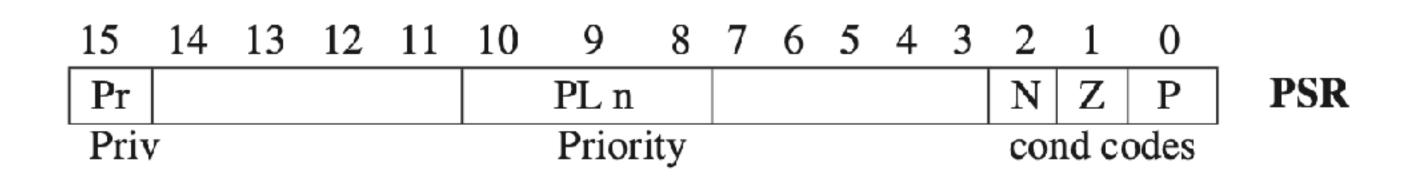
• What should be saved for program to resume?



- What should be saved for program to resume?
 - PC (Program Counter)

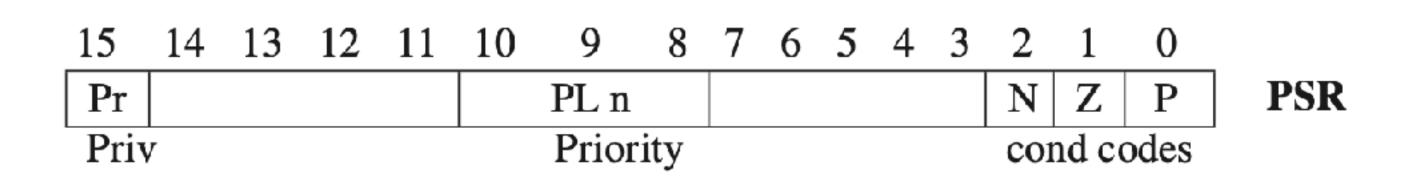


- What should be saved for program to resume?
 - PC (Program Counter)
 - Processor Status Register (PSR)



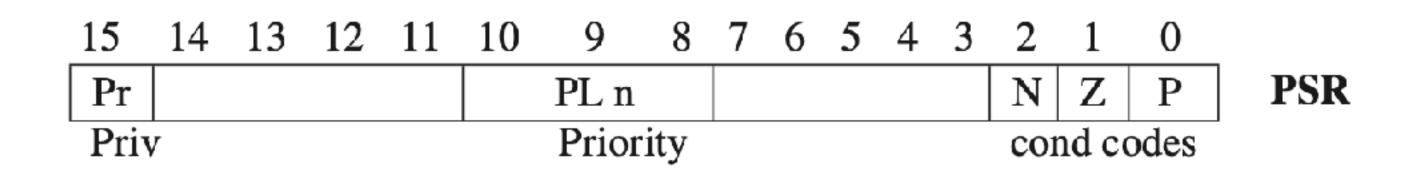


- What should be saved for program to resume?
 - PC (Program Counter)
 - Processor Status Register (PSR)
 - What about GPRs?





- What should be saved for program to resume?
 - PC (Program Counter)
 - Processor Status Register (PSR)
 - What about GPRs?
 - Callee saved





Privilege vs. priority

ECE 220 - Spring 2024 **Dr. Ivan Abraham** See Section 9.1.1.3



Privilege •

Dr. Ivan Abraham ECE 220 - Spring 2024



- Privilege lacksquare
 - Privilege is all about the right to do something, such as execute a particular instruction or access a particular memory location.



- Privilege
 - Privilege is all about the right to do something, such as execute a particular instruction or access a particular memory location.
- Priority \bullet



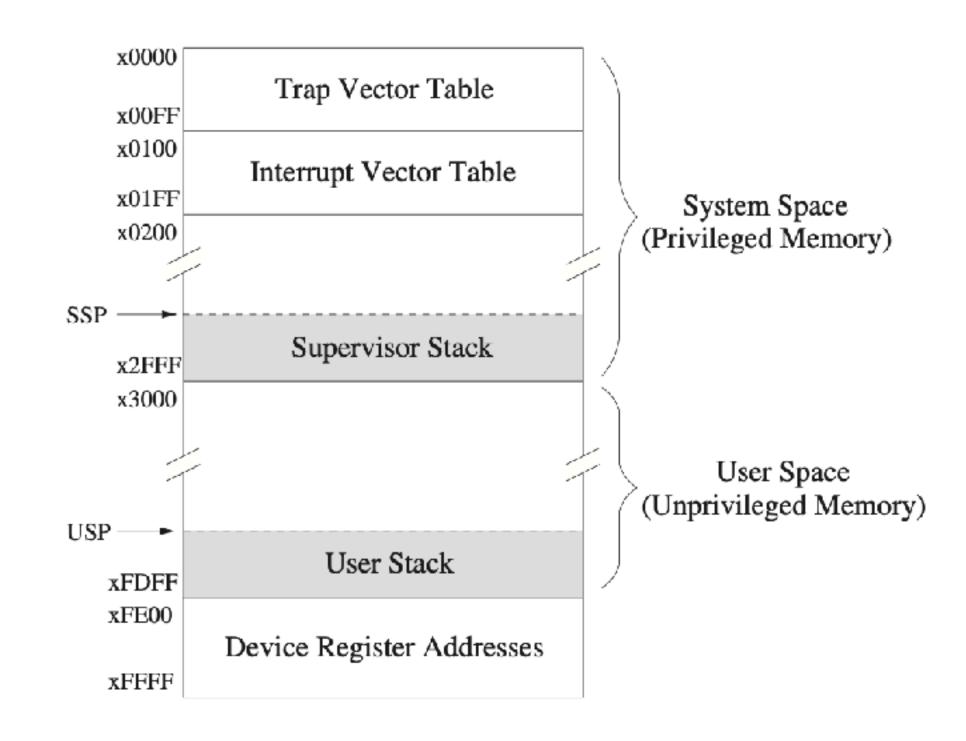
- Privilege
 - Privilege is all about the right to do something, such as execute a particular instruction or access a particular memory location.
- Priority
 - Priority is all about the urgency of a program to execute. Every program is assigned a priority, specifying its urgency as compared to all other programs.



ECE 220 - Spring 2024 Dr. Ivan Abraham

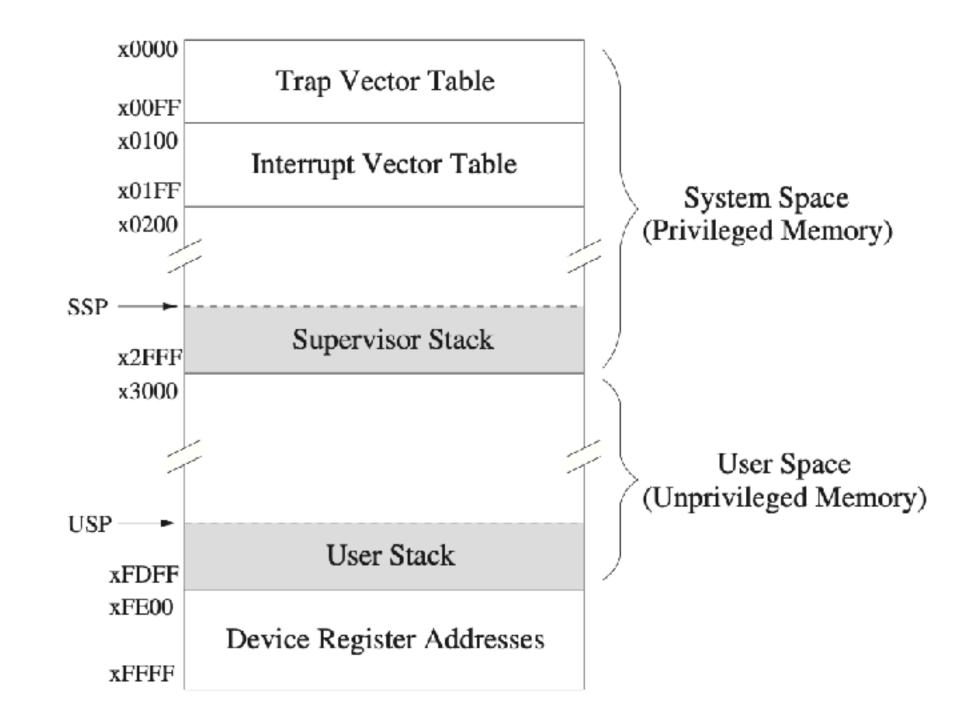


 Portion of privileged memory reserved for stack in supervisor mode (why?)





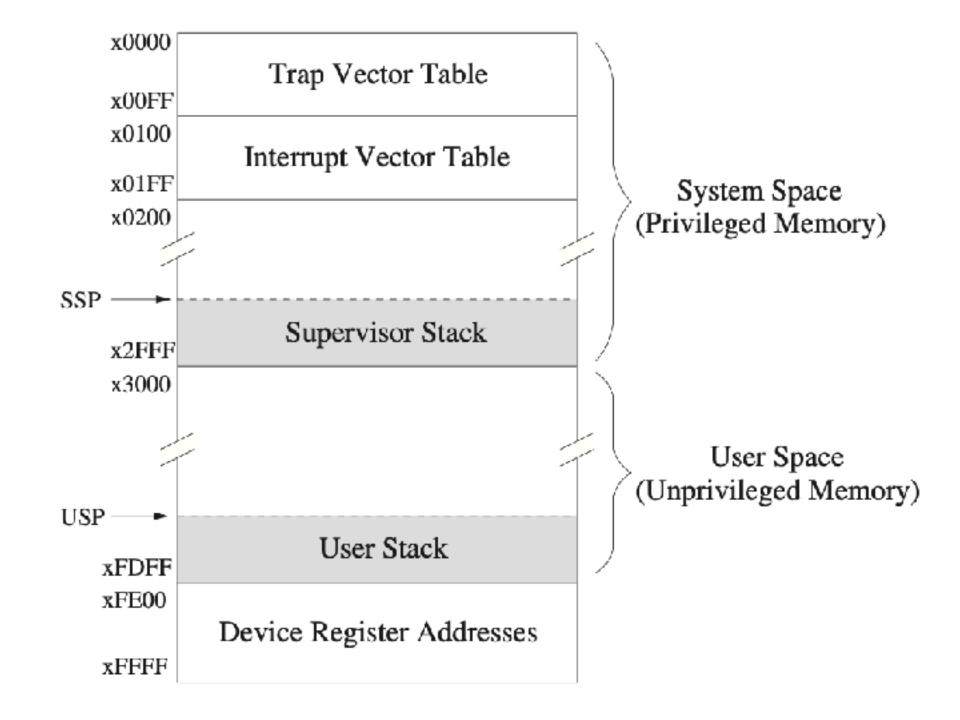
- Portion of privileged memory reserved for stack in supervisor mode (why?)
- R6 is used to denote active TOS always.



ECE 220 - Spring 2024 Dr. Ivan Abraham

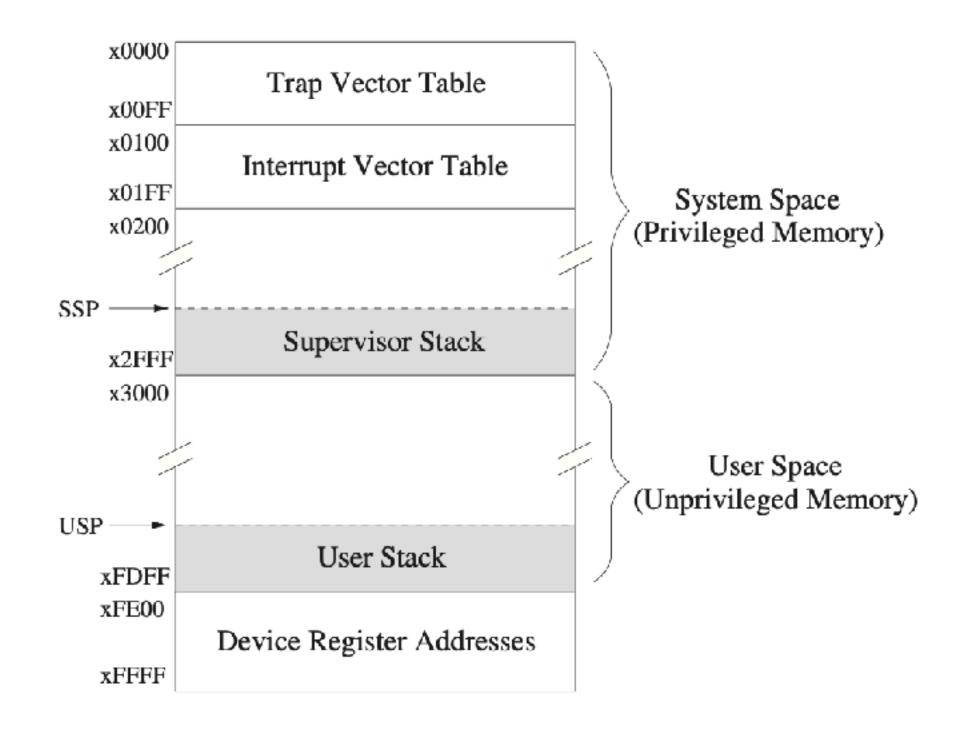


- Portion of privileged memory reserved for stack in supervisor mode (why?)
- R6 is used to denote active TOS always.
 - Can only run in user or supervisor mode at a time.





- Portion of privileged memory reserved for stack in supervisor mode (why?)
- R6 is used to denote active TOS always.
 - Can only run in user or supervisor mode at a time.
 - Save R6 on switching modes using (built-in) registers, Saved_SSP and Saved_USP





ECE 220 - Spring 2024 **Dr. Ivan Abraham**



• The I/O device generating the INT signal also provides an 8-bit *interrupt vector* INTV which identifies the device



- The I/O device generating the INT signal also provides an 8-bit *interrupt vector* INTV which identifies the device
- INTV will determine which ISR to be executed to service the interrupt



- The I/O device generating the INT signal also provides an 8-bit *interrupt vector* INTV which identifies the device
- INTV will determine which ISR to be executed to service the interrupt
 - Similar to TRAP vector table we have seen previously



ECE 220 - Spring 2024 Dr. Ivan Abraham



1. Processor detects asserted INT along with INTV



- 1. Processor detects asserted **INT** along with **INTV**
- 2. Save R6 to Saved USP. Set R6 to Saved SSP.

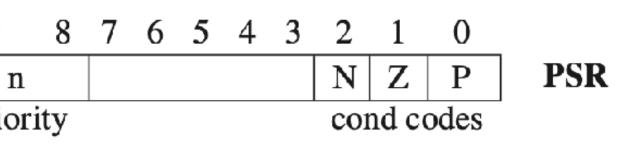


- 1. Processor detects asserted **INT** along with **INTV**
- 2. Save R6 to Saved USP. Set R6 to Saved SSP.
- 3. Push **PSR & PC** to supervisor stack



- 1. Processor detects asserted INT along with INTV
- 2. Save R6 to Saved USP. Set R6 to Saved SSP.
- 3. Push **PSR & PC** to supervisor stack
- 4. Set PSR[15]=0 and set PSR[10:8] = priority of ISR

15	14	13	12	11	10	9
Pr						PL 1
Priv	V				•	Pric

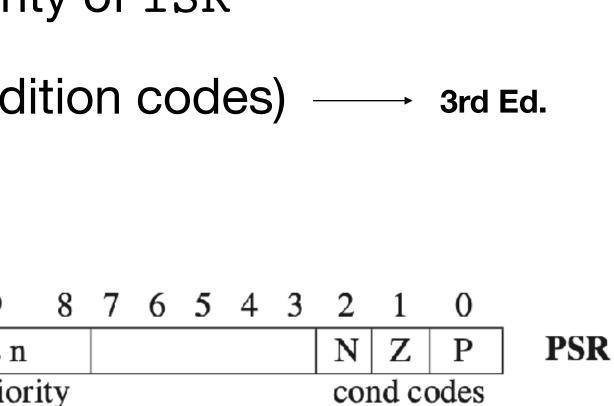




- 1. Processor detects asserted INT along with INTV
- 2. Save R6 to Saved_USP. Set R6 to Saved_SSP.
- 3. Push **PSR** & **PC** to supervisor stack
- 4. Set PSR[15]=0 and set PSR[10:8] = priority of ISR

1. Set PSR[2:0]=010 (arbitrarily sets condition codes) \longrightarrow 3rd Ed.

15	14	13	12	11	10	9
Pr						PL 1
Priv	V				•	Pric

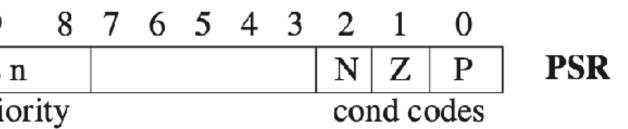




- 1. Processor detects asserted INT along with INTV
- 2. Save R6 to Saved USP. Set R6 to Saved SSP.
- 3. Push PSR & PC to supervisor stack
- 4. Set PSR[15]=0 and set PSR[10:8] = priority of ISR1. Set **PSR[2:0]=010** (arbitrarily sets condition codes) \longrightarrow 3rd Ed.

5. Load MAR with x01vv, where vv = 8-bit interrupt vector, then load MDR

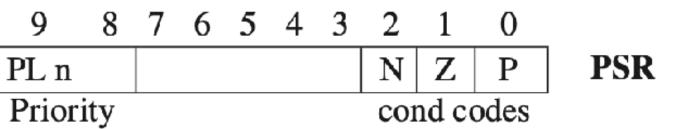
15	14	13	12	11	10	9
Pr						PL 1
Priv	V				•	Pric





- 1. Processor detects asserted INT along with INTV
- 2. Save R6 to Saved USP. Set R6 to Saved SSP.
- 3. Push PSR & PC to supervisor stack
- 4. Set PSR[15]=0 and set PSR[10:8] = priority of ISR 1. Set **PSR[2:0]=010** (arbitrarily sets condition codes) \longrightarrow 3rd Ed.
- 5. Load MAR with x01vv, where vv = 8-bit interrupt vector, then load MDR
- 6. Set PC=MDR 14 13 12 11 10 15 Pr

Priv





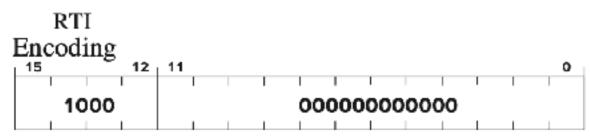
ECE 220 - Spring 2024 **Dr. Ivan Abraham**

ILLINOIS

Both **TRAP** routines and lacksquare**INTerrupt** routines end with the **RTI** instruction.

RTI

Assembler Format



Operation

```
if (PSR[15] == 1)
   Initiate a privilege mode exception;
else
  PC=mem[R6]: R6 is the SSP. PC is restored
   R6 = R6 + 1;
  TEMP=memER6]:
   PSR=TEMP; PSR is restored
   if (PSR[15] == 1)
      Saved_SSP=R6 and R6=Saved_USP:
```

Description

If the processor is running in User mode, a privilege mode exception occurs. If in Supervisor mode, the top two elements on the system stack are popped and loaded into PC, PSR. After PSR is restored, if the processor is running in User mode, the SSP is saved in Saved_SSP, and R6 is loaded with Saved_USP.

Example

RTI ; PC, PSR \leftarrow top two values popped off stack.

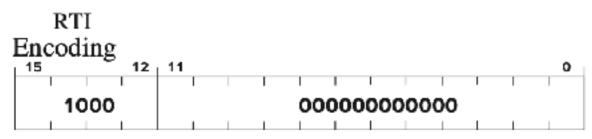
Return from Trap or Interrupt



- Both **TRAP** routines and lacksquare**INT**errupt routines end with the **RTI** instruction.
 - Pop PC & PSR from supervisor stack

RTI

Assembler Format



Operation

```
if (PSR[15] == 1)
   Initiate a privilege mode exception;
else
   PC=mem[R6]; R6 is the SSP, PC is restored
   R6 = R6 + 1:
  TEMP=memER6]:
   PSR=TEMP; PSR is restored
   if (PSR[15] == 1)
      Saved_SSP=R6 and R6=Saved_USP:
```

Description

If the processor is running in User mode, a privilege mode exception occurs. If in Supervisor mode, the top two elements on the system stack are popped and loaded into PC, PSR. After PSR is restored, if the processor is running in User mode, the SSP is saved in Saved_SSP, and R6 is loaded with Saved_USP.

Example

RTI ; PC, PSR \leftarrow top two values popped off stack.

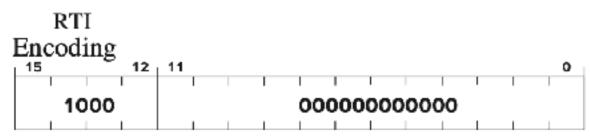
Return from Trap or Interrupt



- Both **TRAP** routines and lacksquare**INT**errupt routines end with the **RTI** instruction.
 - Pop PC & PSR from supervisor stack
 - If PSR[15]=1, R6=Saved.USP

RTI

Assembler Format



Operation

```
if (PSR[15] == 1)
   Initiate a privilege mode exception;
else
   PC=mem[R6]; R6 is the SSP, PC is restored
   R6 = R6 + 1:
  TEMP=mem[R6]:
   PSR=TEMP; PSR is restored
   if (PSR[15] == 1)
      Saved_SSP=R6 and R6=Saved_USP;
```

Description

If the processor is running in User mode, a privilege mode exception occurs. If in Supervisor mode, the top two elements on the system stack are popped and loaded into PC, PSR. After PSR is restored, if the processor is running in User mode, the SSP is saved in Saved_SSP, and R6 is loaded with Saved_USP.

Example

RTI ; PC, PSR \leftarrow top two values popped off stack.

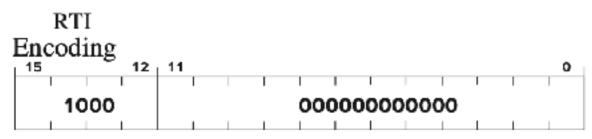
Return from Trap or Interrupt



- Both **TRAP** routines and lacksquare**INT**errupt routines end with the **RTI** instruction.
 - Pop PC & PSR from supervisor stack
 - If PSR[15]=1, R6=Saved.USP
 - **RTI** is privileged

RTI

Assembler Format



Operation

```
if (PSR[15] == 1)
  Initiate a privilege mode exception;
else
   PC=mem[R6]; R6 is the SSP, PC is restored
   R6 = R6 + 1;
  TEMP=mem[R6]:
   PSR=TEMP; PSR is restored
   if (PSR[15] == 1)
      Saved_SSP=R6 and R6=Saved_USP;
```

Description

If the processor is running in User mode, a privilege mode exception occurs. If in Supervisor mode, the top two elements on the system stack are popped and loaded into PC, PSR. After PSR is restored, if the processor is running in User mode, the SSP is saved in Saved_SSP, and R6 is loaded with Saved_USP.

Example

RTI ; PC, PSR \leftarrow top two values popped off stack.

Return from Trap or Interrupt



https://courses.grainger.illinois.edu/ece220/sp2020/lc3web/index.html

Interrupt example

ECE 220 - Spring 2024 **Dr. Ivan Abraham**

Example by Profs. T. Moon & Y. Chen



Interrupt example

.ORIG	x3000	
	LEA	R0, ISR_KB
	STI	R0, KBINTV ; load ISR address to IN
	LD	R3, EN_IE
	STI	R3, KBSR ; set IE bit of KBSR
AGAIN	LD	R0, NUM2
	OUT	
	BRnzp	AGAIN
ISR_KB	ST	R0, SaveR0 ; callee-save R0
	LDI	R0, KBDR ; read a char from KB an
	OUT	
	LD	R0, SaveR0 ; callee-restore R0
	HALT	
;		
EN_IE	.FILL	x4000 ; To enable the IE bit
NUM2	.FILL	x0032 ; ASCII Code for '2'
KBSR	.FILL	xFE00
KBDR	.FILL	xFE02
KBINTV	.FILL	x0180 ; INT vector table address for
SaveR0	.BLKW	#1
.END		

Dr. Ivan Abraham ECE 220 - Spring 2024

https://courses.grainger.illinois.edu/ece220/sp2020/lc3web/index.html

NTV

nd clear ready bit

r keyboard

Example by Profs. T. Moon & Y. Chen



Interrupt example

.ORIG	x3000	
	LEA	R0, ISR_KB
	STI	R0, KBINTV ; load ISR address to IN
	LD	R3, EN_IE
	STI	R3, KBSR ; set IE bit of KBSR
AGAIN	LD	R0, NUM2
	OUT	
	BRnzp	AGAIN
ISR_KB	ST	R0, SaveR0 ; callee-save R0
	LDI	R0, KBDR ; read a char from KB an
	OUT	
	LD	R0, SaveR0 ; callee-restore R0
	HALT	
;		
EN_IE	.FILL	x4000 ; To enable the IE bit
NUM2	.FILL	x0032 ; ASCII Code for '2'
KBSR	.FILL	xFE00
KBDR	.FILL	xFE02
KBINTV	.FILL	x0180 ; INT vector table address for
SaveR0	.BLKW	#1
.END		

Dr. Ivan Abraham ECE 220 - Spring 2024

https://courses.grainger.illinois.edu/ece220/sp2020/lc3web/index.html

What does this program do?

NTV

nd clear ready bit

r keyboard

Example by Profs. T. Moon & Y. Chen



ECE 220 - Spring 2024 **Dr. Ivan Abraham**



• Exceptions happen when something ... well unexpected happens within the processor.



- Exceptions happen when something ... well unexpected happens within the processor.
 - **Examples:** Privilege mode violation (PMV), illegal/invalid opcodes, accessing privileged memory (ACV)



- Exceptions happen when something ... well unexpected happens within the processor.
 - **Examples**: Privilege mode violation (PMV), illegal/invalid opcodes, accessing privileged memory (ACV)
 - Exception services routines occupy memory locations x0100 to x017F, mechanism exactly like INTerrupts except ...

19



- Exceptions happen when something ... well unexpected happens within the processor.
 - **Examples**: Privilege mode violation (PMV), illegal/invalid opcodes, accessing privileged memory (ACV)
 - Exception services routines occupy memory locations x0100 to x017F, mechanism exactly like INTerrupts except ...
 - Priority level is not changed.

19



Next lecture(s)

- Examples
- Course review
- Exam preparation tips
- ICES forms

