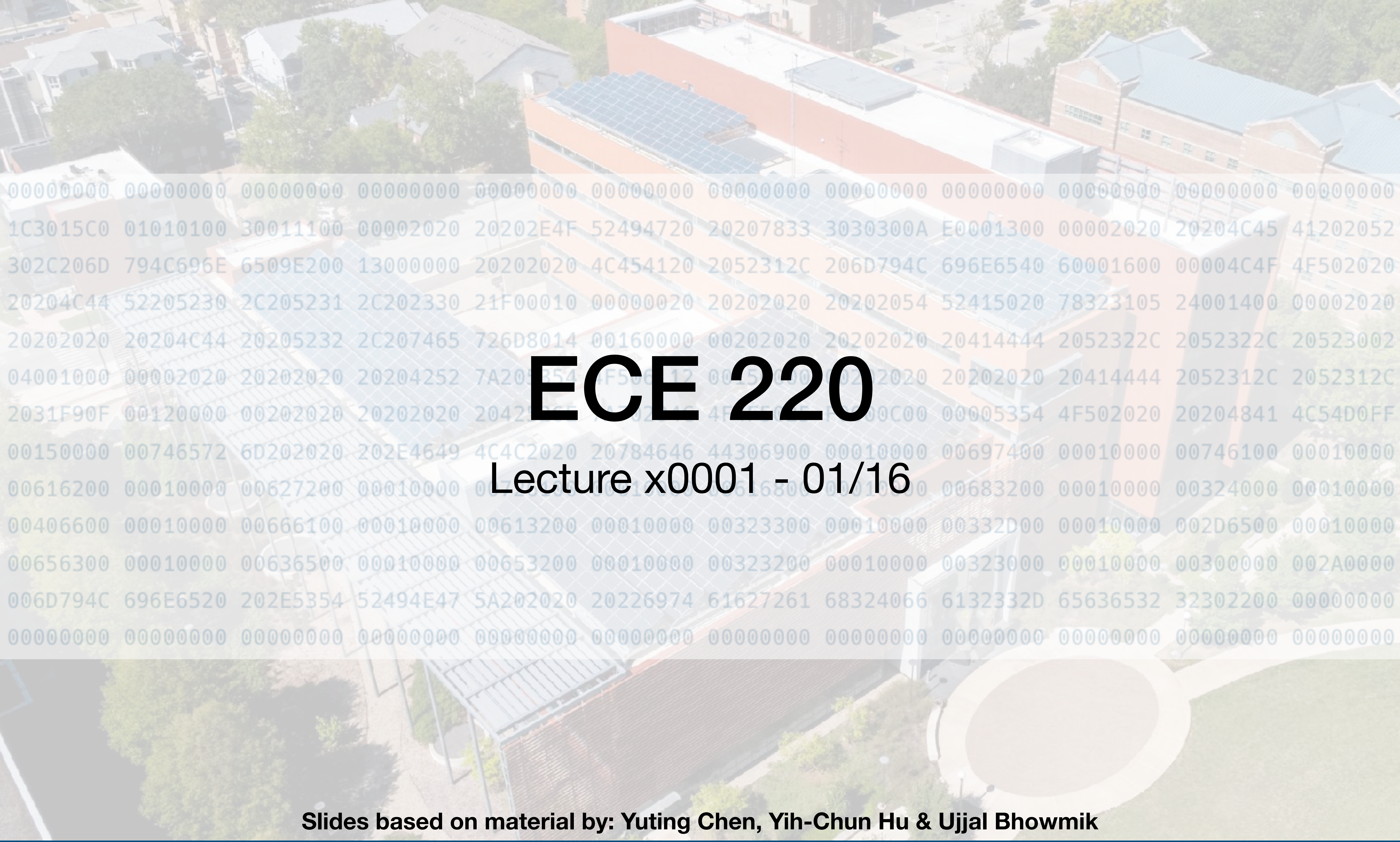


ECE 220

Lecture x0001 - 01/16

Slides based on material by: Yuting Chen, Yih-Chun Hu & Ujjal Bhowmik

Memory mapped I/O

- How do we communicate with the computer?

- Memory-mapped I/O: Hardware devices (i.e. their registers) are treated the same as the computer's main memory and addressable the same way

    - Memory of *peripherals* **is** *physically separate* from main memory

- Alternative: Port mapped I/O (requires having more specialized instructions)

- In LC3: `KBDR,KBSR,DSR,DDR` are used for [K]eyboard and [D]isplay respectively.

LC3 - Input/Output (IO)

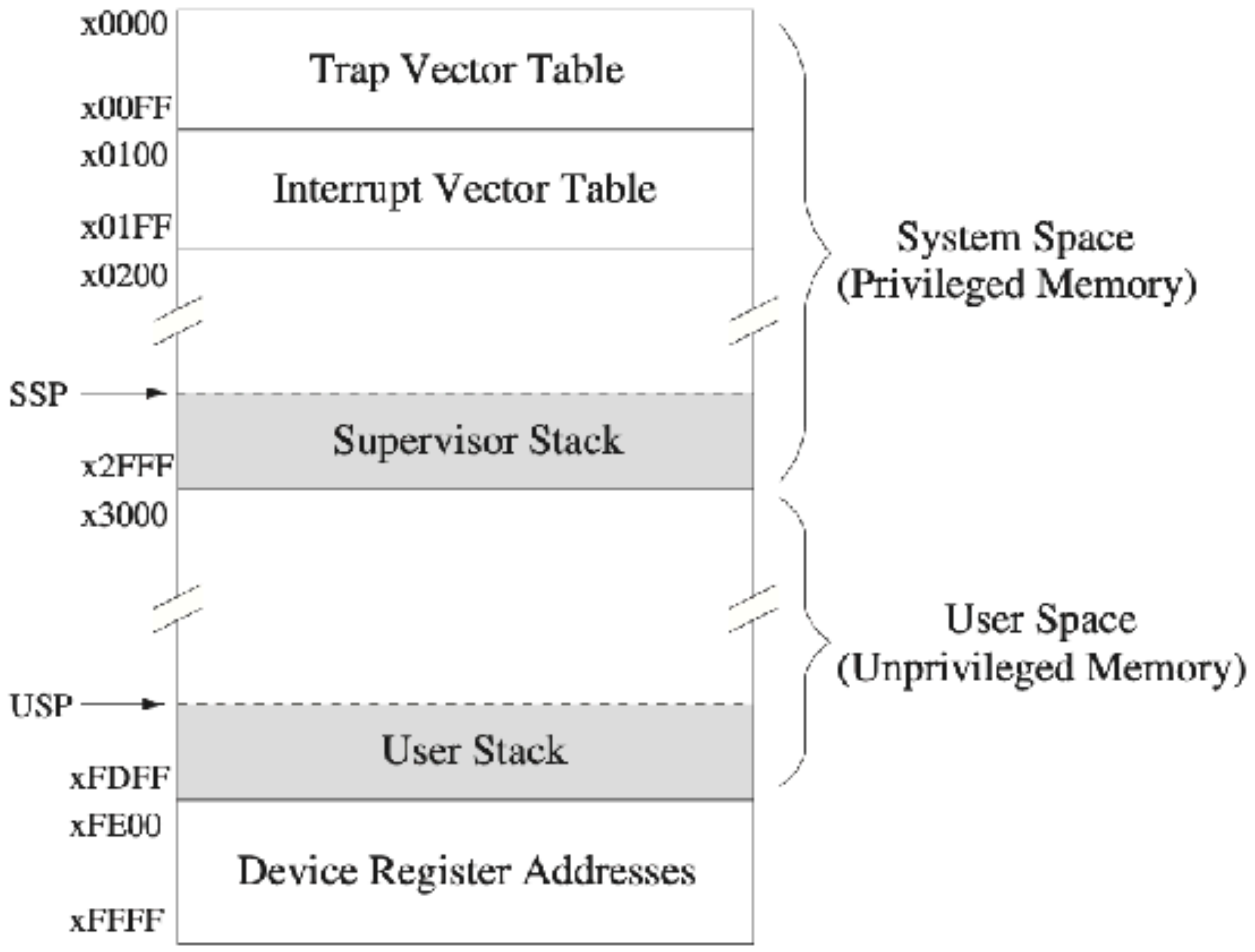
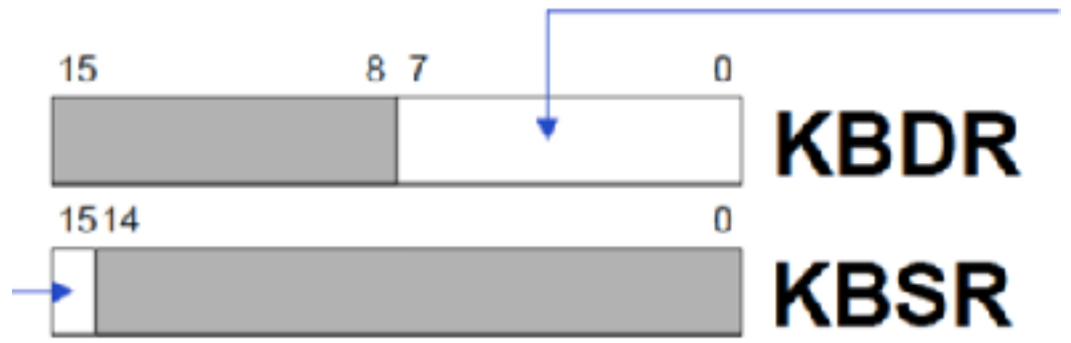
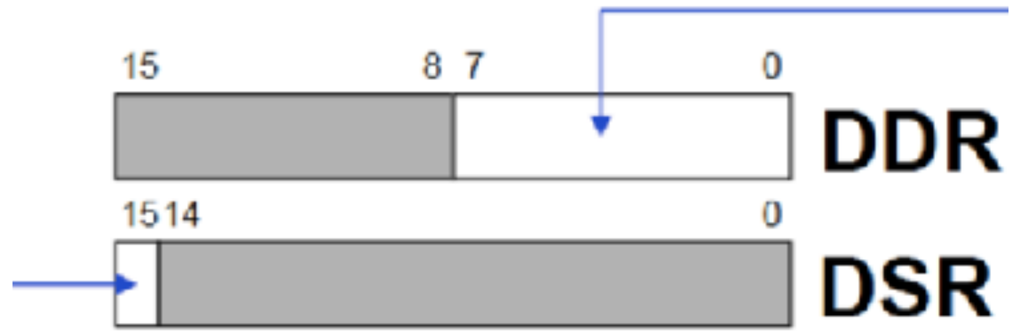


Figure A.1 - P&P 3rd Ed.

Address	I/O Register Name	I/O Register Function
xFE00	Keyboard status register (KBSR)	The ready bit (bit[15]) indicates if the keyboard has received a new character
xFE02	Keyboard data register (KBDR)	Bits [7:0] contain the last character typed on the keyboard
xFE04	Display status register (DSR)	The ready bit (bit[15]) indicates if the display device is ready to receive another character to print on the screen
xFE06	Display data register (DDR)	A character written in bits [7:0] will be displayed displayed on the screen

LC3 - Input/Output (IO)





Address	I/O Register Name	I/O Register Function
xFE00	Keyboard status register (KBSR)	The ready bit (bit[15]) indicates if the keyboard has received a new character
xFE02	Keyboard data register (KBDR)	Bits [7:0] contain the last character typed on the keyboard
xFE04	Display status register (DSR)	The ready bit (bit[15]) indicates if the display device is ready to receive another character to print on the screen
xFE06	Display data register (DDR)	A character written in bits [7:0] will be displayed displayed on the screen

LC3 - Input from keyboard

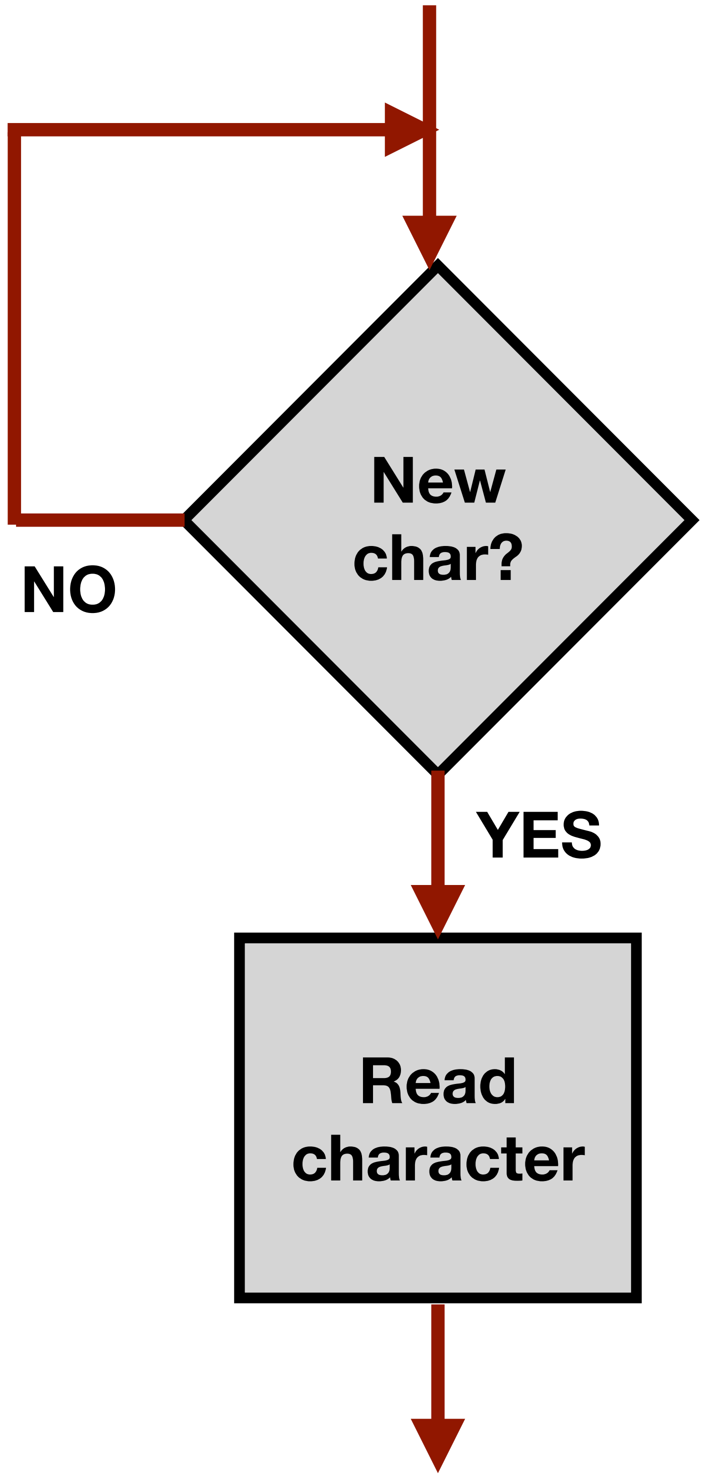
Basic routine

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
    - `KBSR[15]` is set to 0
    - Keyboard is enabled

LC3 - Input from keyboard

Basic routine



```
.ORIG x3000

;Create a loop to
check KBSR


;If ready bit unset
loop again


;If ready bit set,
read KBDR into R0


KBSR .FILL xFE00

KBDR .FILL xFE02
```

LC3 - Display to console

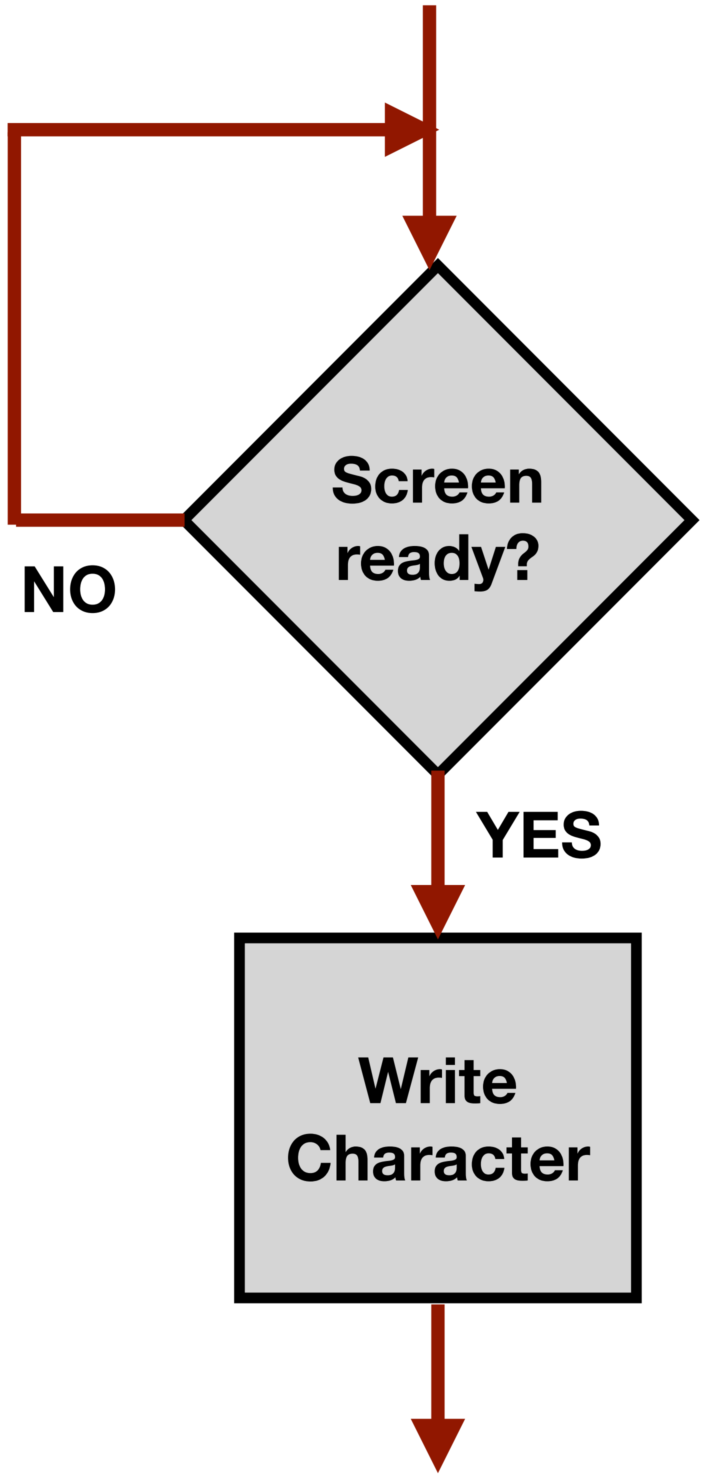
Basic routine

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

LC3 - Display to console

Basic routine



```
.ORIG x3000

;Create a loop to
check DSR


;If ready bit unset
loop again


;If ready bit set,
write R0 into DDR


DSR .FILL xFE04

DDR .FILL xFE06
```

Exercise

- Write a program to display “ECE 220 is fun!” to the console. You can use the pseudo-op `.STRINGZ` to store string to memory. Do not use `TRAP` codes (*if you know what they are*).

Issues?

- Limited amount of GPRs - polling display & keyboard uses up two of them

- Code often repeated - inefficient to keep inserting same code over & over again

- Human error - keeping track of registers & having direct access to hardware registers is recipe for unforced errors & bugs

Next time

- Subroutines & repeated code
  - Also called *functions*
- TRAP routines