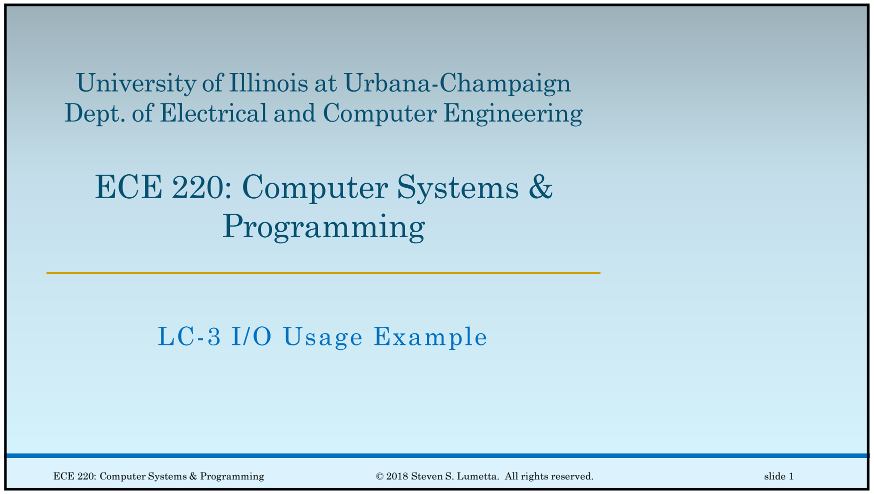
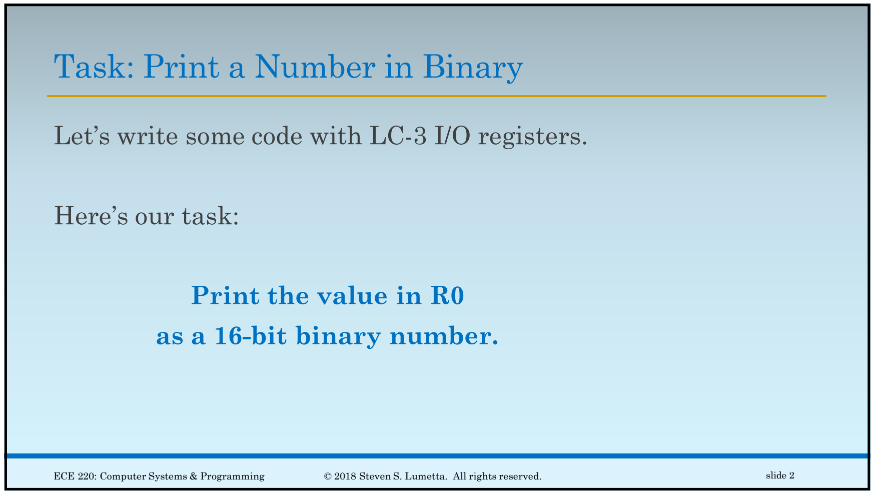


University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering

ECE 220: Computer Systems & Programming

LC-3 I/O Usage Example

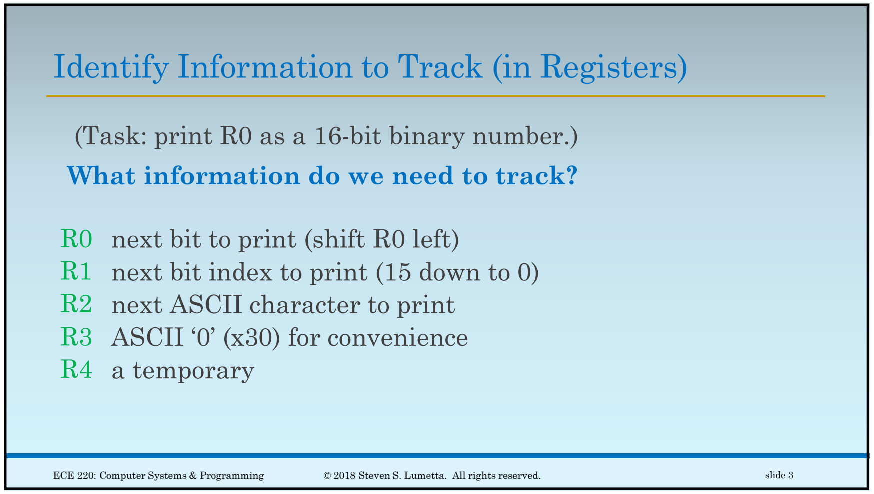


Task: Print a Number in Binary

Let's write some code with LC-3 I/O registers.

Here's our task:

Print the value in R0
as a 16-bit binary number.

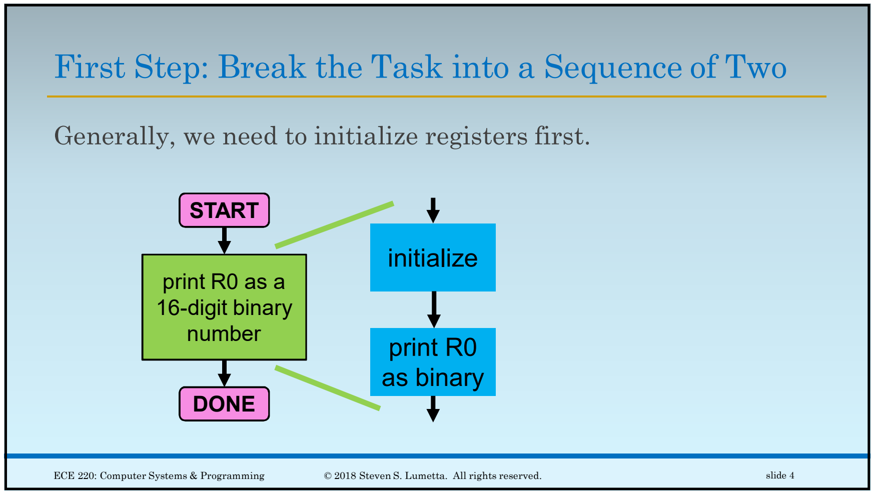


Identify Information to Track (in Registers)

(Task: print R0 as a 16-bit binary number.)

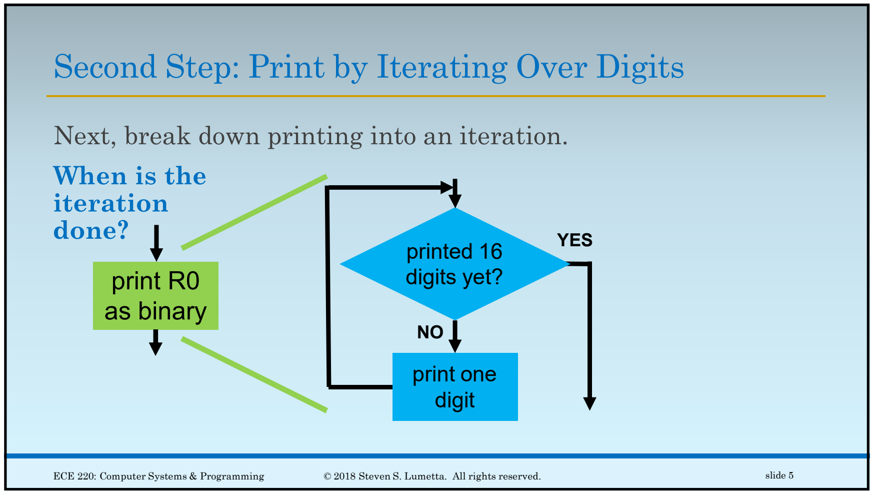
What information do we need to track?

R0	next bit to print (shift R0 left)
R1	next bit index to print (15 down to 0)
R2	next ASCII character to print
R3	ASCII '0' (x30) for convenience
R4	a temporary



First Step: Break the Task into a Sequence of Two

Generally, we need to initialize registers first.



Second Step: Print by Iterating Over Digits

Next, break down printing into an iteration.

When is the iteration done?

print R0 as binary

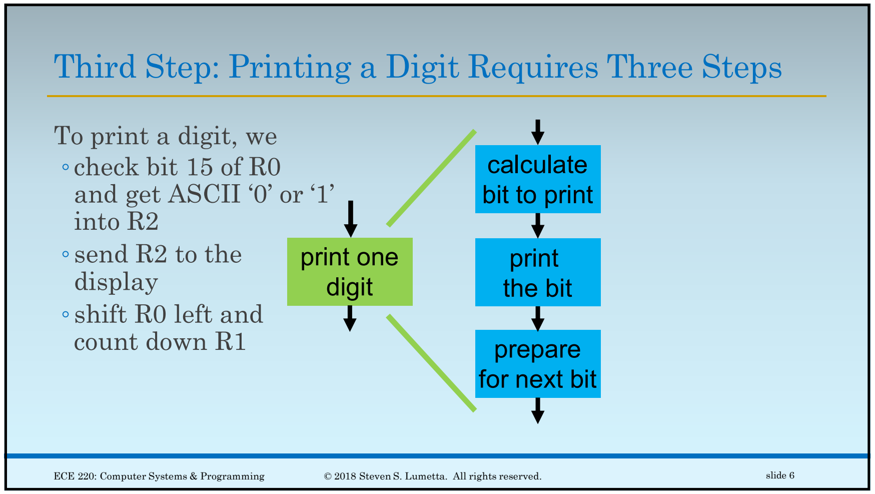
printed 16 digits yet?

YES

NO

print one digit

ECE 220: Computer Systems & Programming © 2018 Steven S. Lumetta. All rights reserved. slide 5



Third Step: Printing a Digit Requires Three Steps

To print a digit, we

- check bit 15 of R0 and get ASCII ‘0’ or ‘1’ into R2
- send R2 to the display
- shift R0 left and count down R1

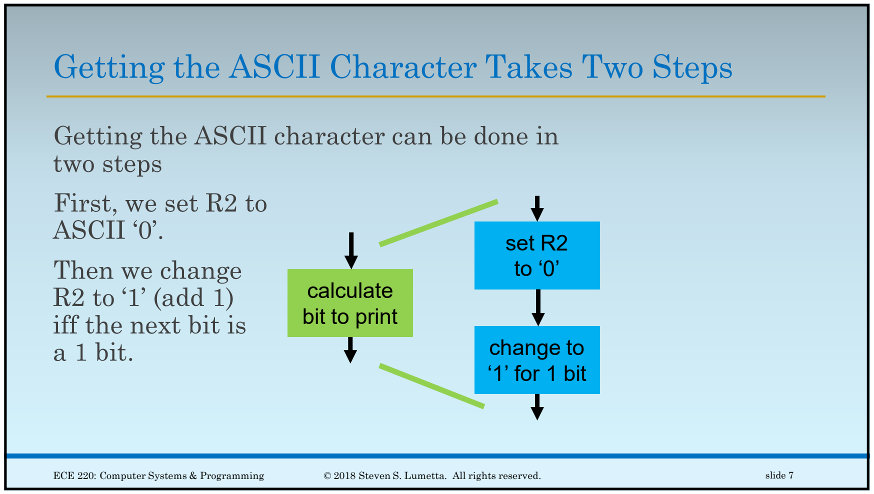
print one digit

calculate bit to print

print the bit

prepare for next bit

ECE 220: Computer Systems & Programming © 2018 Steven S. Lumetta. All rights reserved. slide 6



Getting the ASCII Character Takes Two Steps

Getting the ASCII character can be done in two steps

First, we set R2 to ASCII ‘0’.

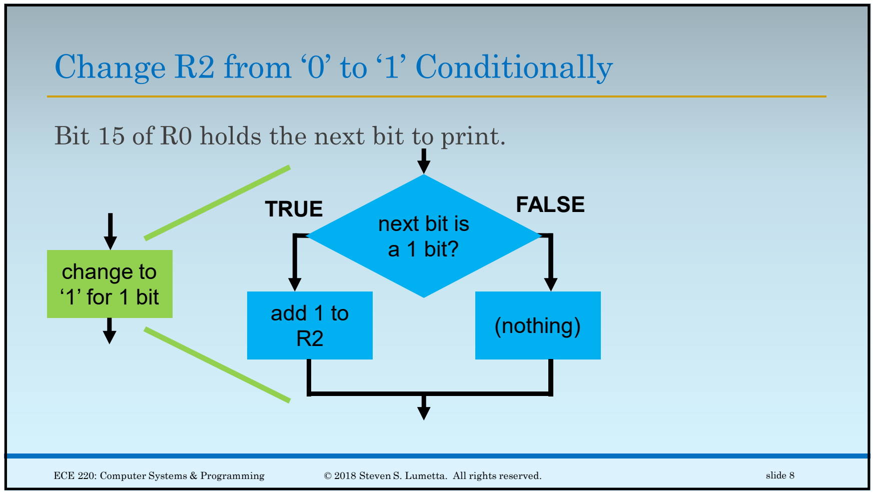
Then we change R2 to ‘1’ (add 1) iff the next bit is a 1 bit.

calculate bit to print

set R2 to ‘0’

change to ‘1’ for 1 bit

ECE 220: Computer Systems & Programming © 2018 Steven S. Lumetta. All rights reserved. slide 7



Change R2 from ‘0’ to ‘1’ Conditionally

Bit 15 of R0 holds the next bit to print.

change to ‘1’ for 1 bit

next bit is a 1 bit?

TRUE

FALSE

add 1 to R2

(nothing)

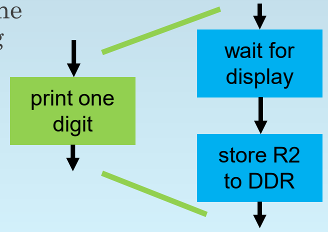
ECE 220: Computer Systems & Programming © 2018 Steven S. Lumetta. All rights reserved. slide 8

Printing an ASCII Character Takes Two Steps

Printing the bit also takes two steps

First, we wait for the display by checking DSR.

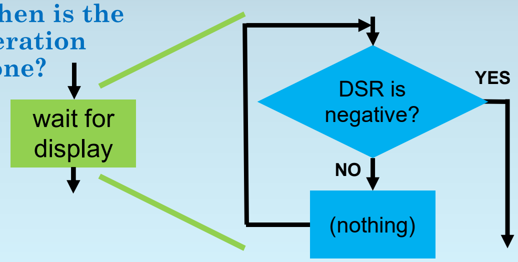
Then we write the character to the DDR.



Waiting for the Display Requires an Iteration

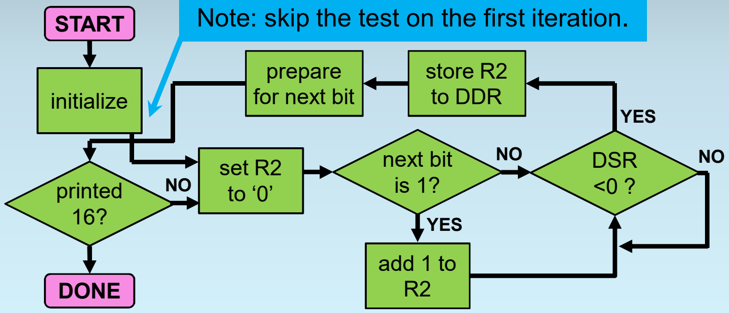
Waiting for the display is an iteration.

When is the iteration done?



A Flow Chart for Printing in Binary

Note: skip the test on the first iteration.



Only Two Registers Need Initialization

Let's write the code!

What needs to be initialized?

R0 is given

~~R0 next bit to print (shift R0 left)~~

R1 next bit index to print (15 down to 0)

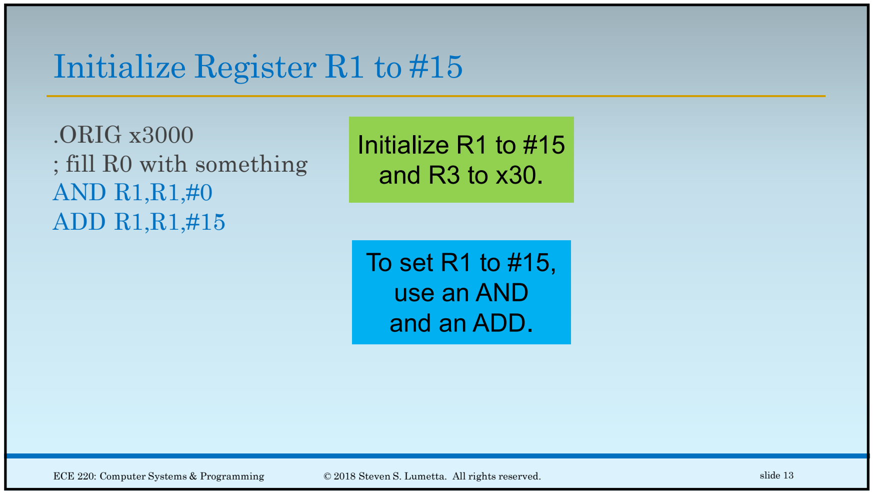
~~R2 next ASCII character to print~~

R3 ASCII '0' (x30) for convenience

~~R4 a temporary~~

R2 is calculated later

R4 is temporary

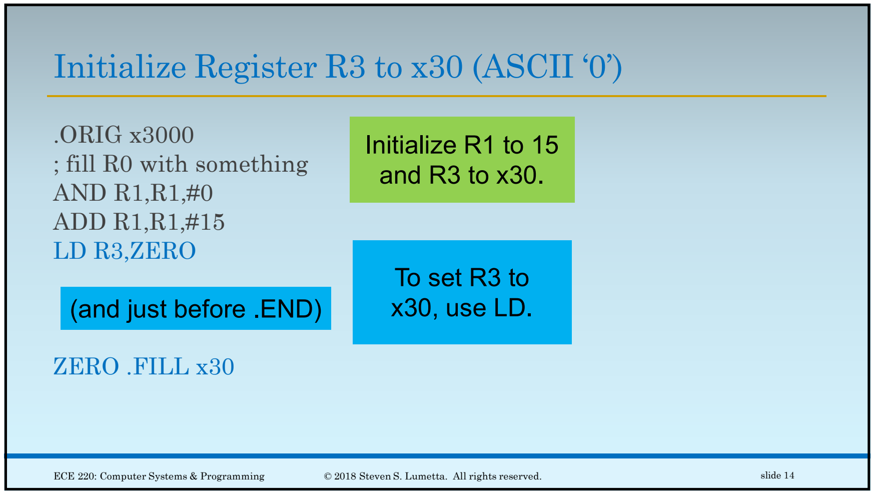


Initialize Register R1 to #15

```
.ORIG x3000
; fill R0 with something
AND R1,R1,#0
ADD R1,R1,#15
```

Initialize R1 to #15 and R3 to x30.

To set R1 to #15, use an AND and an ADD.



Initialize Register R3 to x30 (ASCII '0')

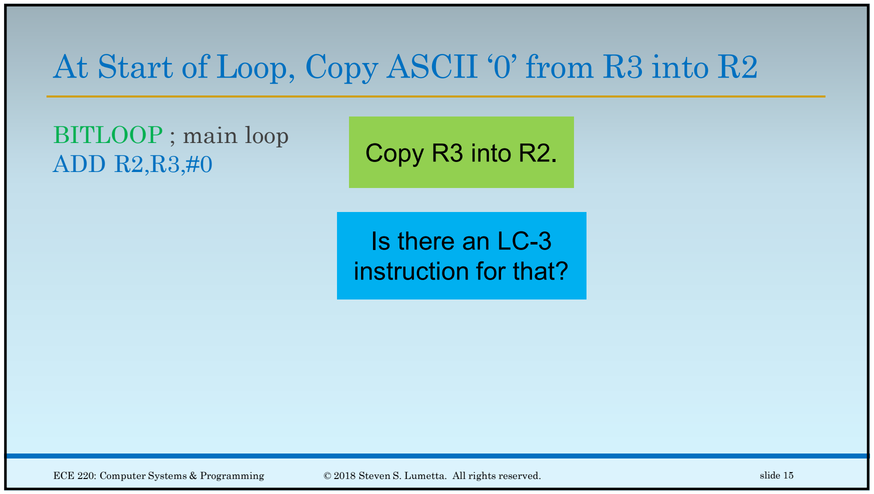
```
.ORIG x3000
; fill R0 with something
AND R1,R1,#0
ADD R1,R1,#15
LD R3,ZERO
```

(and just before .END)

```
ZERO .FILL x30
```

Initialize R1 to 15 and R3 to x30.

To set R3 to x30, use LD.

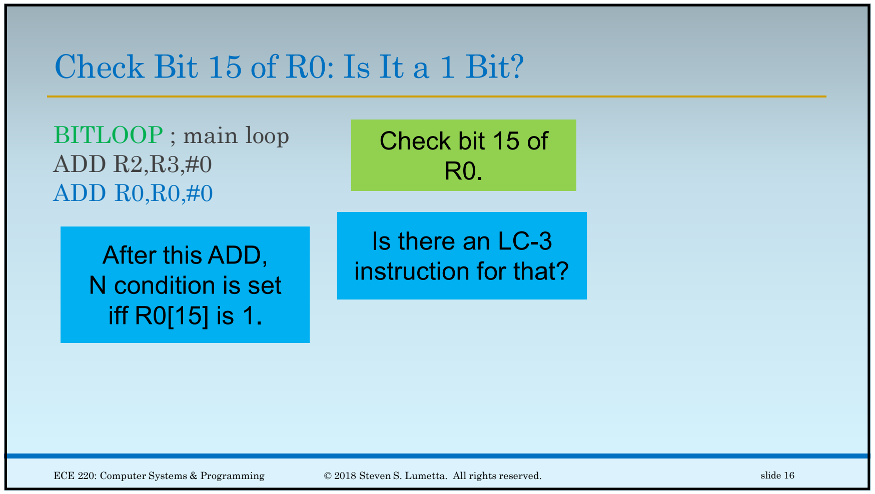


At Start of Loop, Copy ASCII '0' from R3 into R2

```
BITLOOP ; main loop
ADD R2,R3,#0
```

Copy R3 into R2.

Is there an LC-3 instruction for that?



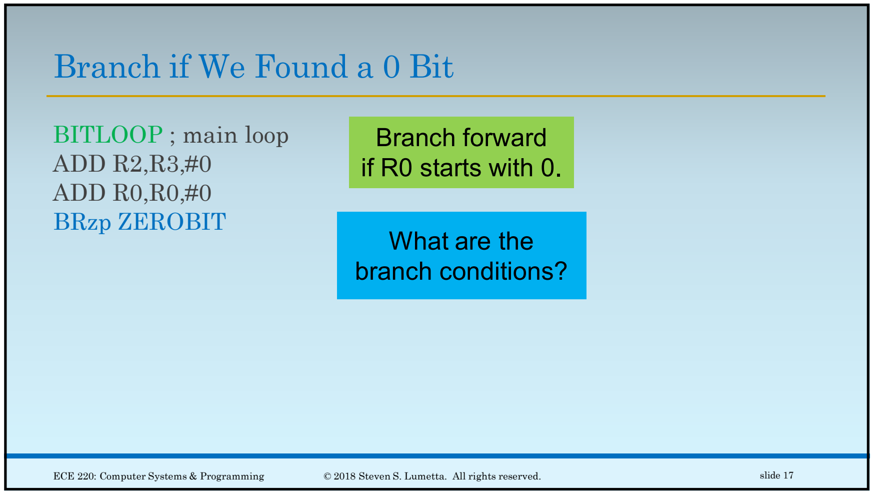
Check Bit 15 of R0: Is It a 1 Bit?

```
BITLOOP ; main loop
ADD R2,R3,#0
ADD R0,R0,#0
```

Check bit 15 of R0.

After this ADD, N condition is set iff R0[15] is 1.

Is there an LC-3 instruction for that?



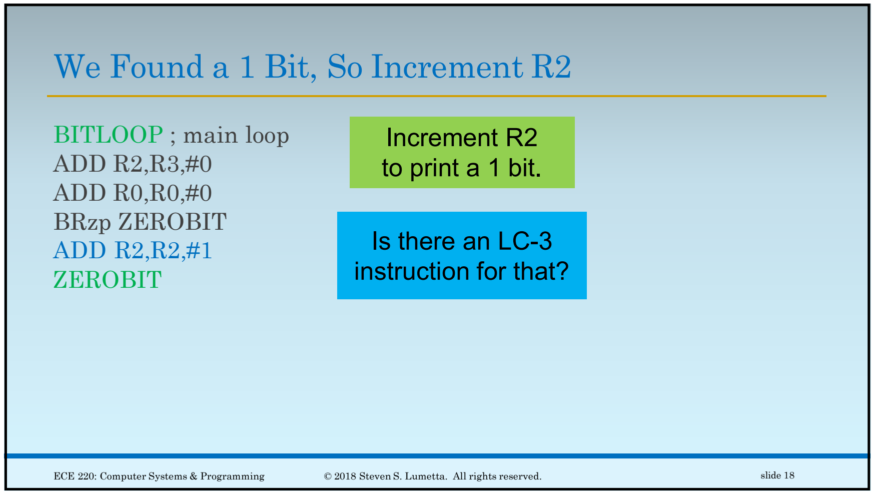
Branch if We Found a 0 Bit

```
BITLOOP ; main loop
ADD R2,R3,#0
ADD R0,R0,#0
BRzp ZEROBIT
```

Branch forward if R0 starts with 0.

What are the branch conditions?

ECE 220: Computer Systems & Programming © 2018 Steven S. Lumetta. All rights reserved. slide 17



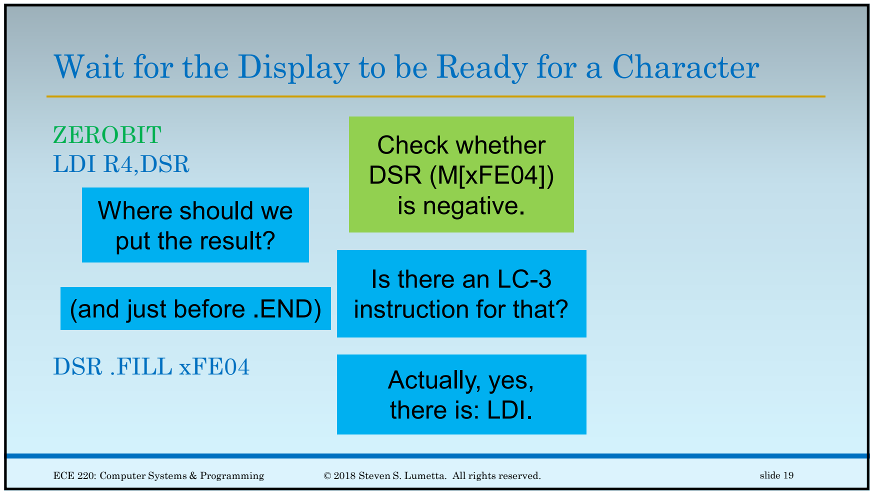
We Found a 1 Bit, So Increment R2

```
BITLOOP ; main loop
ADD R2,R3,#0
ADD R0,R0,#0
BRzp ZEROBIT
ADD R2,R2,#1
ZEROBIT
```

Increment R2 to print a 1 bit.

Is there an LC-3 instruction for that?

ECE 220: Computer Systems & Programming © 2018 Steven S. Lumetta. All rights reserved. slide 18



Wait for the Display to be Ready for a Character

```
ZEROBIT
LDI R4,DSR
```

Where should we put the result?

(and just before .END)

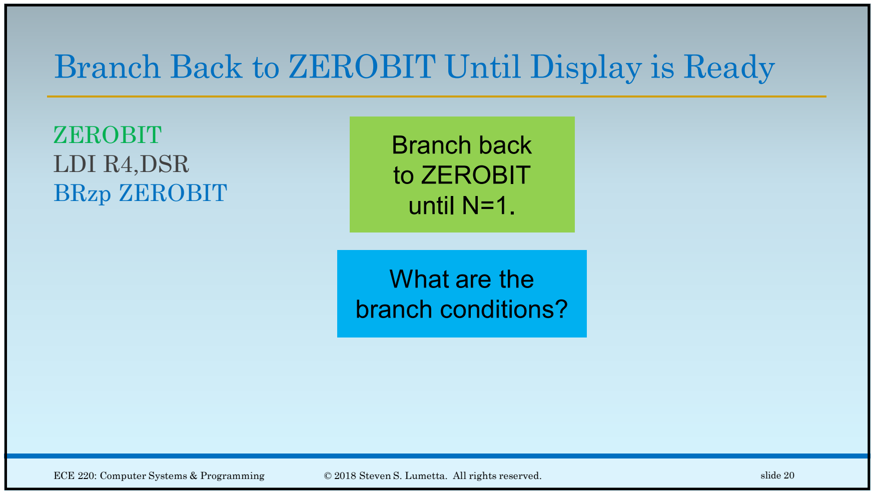
```
DSR .FILL xFE04
```

Check whether DSR (M[xFE04]) is negative.

Is there an LC-3 instruction for that?

Actually, yes, there is: LDI.

ECE 220: Computer Systems & Programming © 2018 Steven S. Lumetta. All rights reserved. slide 19



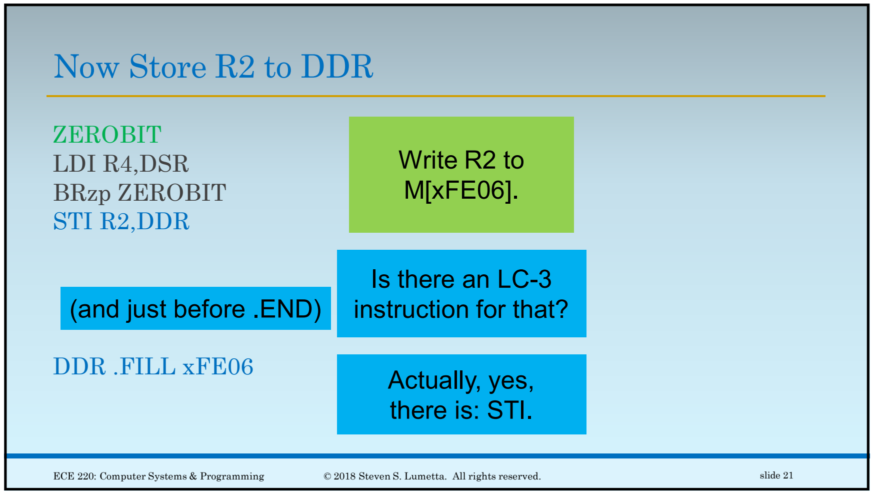
Branch Back to ZEROBIT Until Display is Ready

```
ZEROBIT
LDI R4,DSR
BRzp ZEROBIT
```

Branch back to ZEROBIT until N=1.

What are the branch conditions?

ECE 220: Computer Systems & Programming © 2018 Steven S. Lumetta. All rights reserved. slide 20



Now Store R2 to DDR

```
ZEROBIT
LDI R4,DSR
BRzp ZEROBIT
STI R2,DDR
```

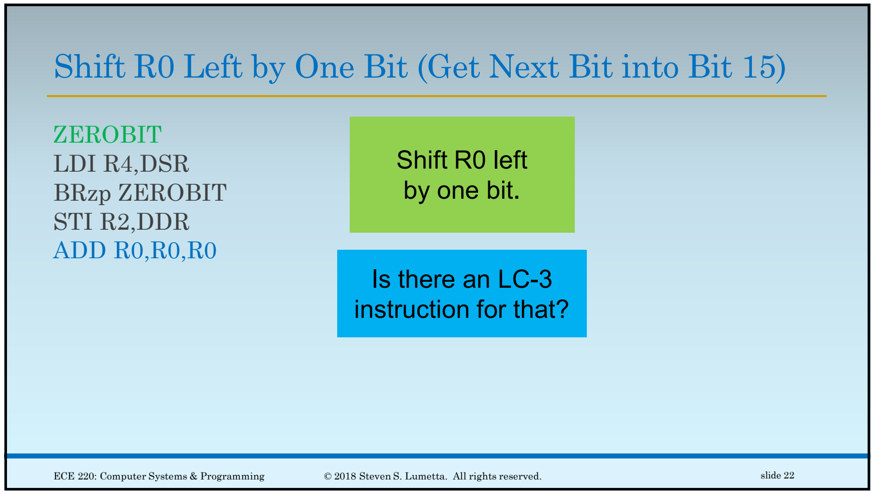
Write R2 to M[xFE06].

Is there an LC-3 instruction for that?

(and just before .END)

```
DDR .FILL xFE06
```

Actually, yes, there is: STI.

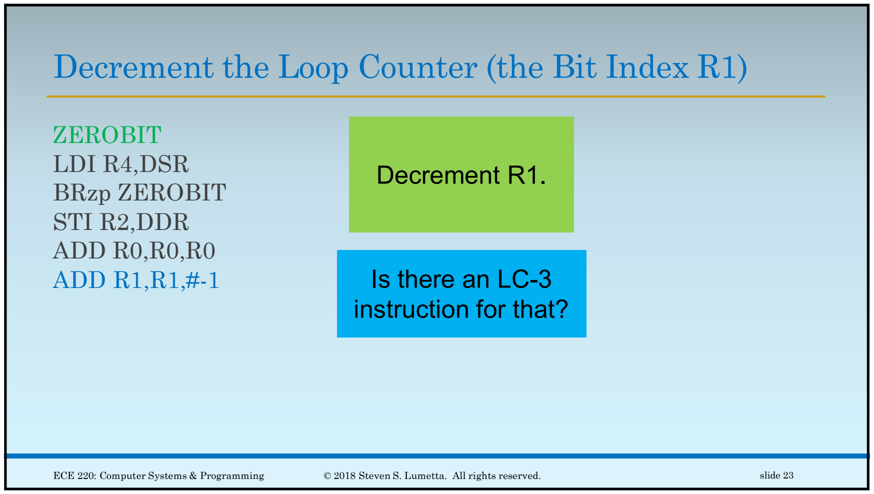


Shift R0 Left by One Bit (Get Next Bit into Bit 15)

```
ZEROBIT
LDI R4,DSR
BRzp ZEROBIT
STI R2,DDR
ADD R0,R0,R0
```

Shift R0 left by one bit.

Is there an LC-3 instruction for that?

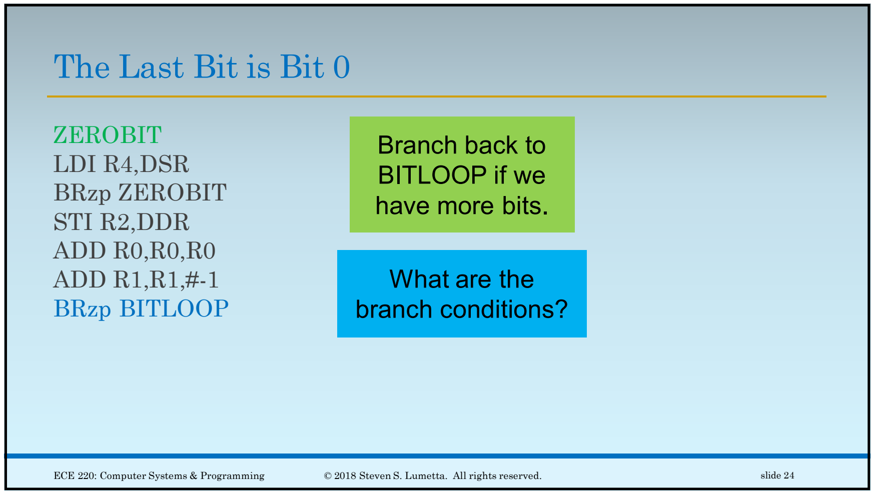


Decrement the Loop Counter (the Bit Index R1)

```
ZEROBIT
LDI R4,DSR
BRzp ZEROBIT
STI R2,DDR
ADD R0,R0,R0
ADD R1,R1,#-1
```

Decrement R1.

Is there an LC-3 instruction for that?

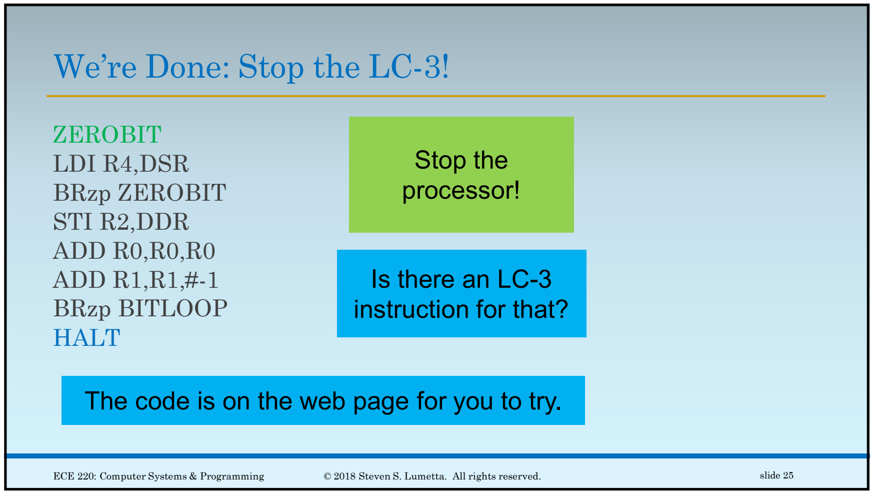


The Last Bit is Bit 0

```
ZEROBIT
LDI R4,DSR
BRzp ZEROBIT
STI R2,DDR
ADD R0,R0,R0
ADD R1,R1,#-1
BRzp BITLOOP
```

Branch back to BITLOOP if we have more bits.

What are the branch conditions?



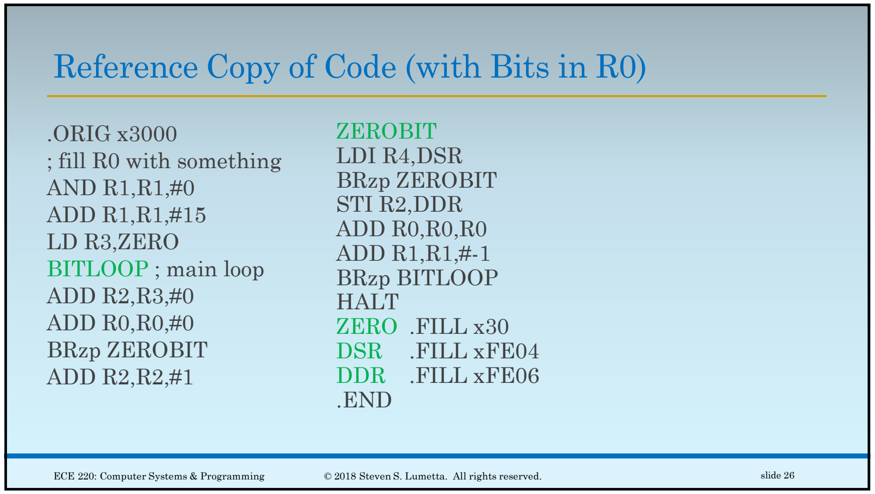
We're Done: Stop the LC-3!

```
ZEROBIT
LDI R4,DSR
BRzp ZEROBIT
STI R2,DDR
ADD R0,R0,R0
ADD R1,R1,#-1
BRzp BITLOOP
HALT
```

Stop the processor!

Is there an LC-3 instruction for that?

The code is on the web page for you to try.



Reference Copy of Code (with Bits in R0)

```
.ORIG x3000
; fill R0 with something
AND R1,R1,#0
ADD R1,R1,#15
LD R3,ZERO
BITLOOP ; main loop
ADD R2,R3,#0
ADD R0,R0,#0
BRzp ZEROBIT
ADD R2,R2,#1
ZEROBIT
LDI R4,DSR
BRzp ZEROBIT
STI R2,DDR
ADD R0,R0,R0
ADD R1,R1,#-1
BRzp BITLOOP
HALT
ZERO  .FILL x30
DSR   .FILL xFE04
DDR   .FILL xFE06
.END
```