- Device Driver I^2C

- Synchronization → Device could be faster or slower than microcontroller, so we synchronize
  1. Blind Cycle
  * 2. Busy-wait Synchronization
  3. Interrupt

- Local Variables in assembly

Watch 17.1 Variables video for Scope & persistence

1. Local Variables
   → reuse of memory
   → access limitation
   → not affected by interrupts /recursion

Steps :
- a Binding XXX EQU 0
- b Allocation SUB SP , multiples of 8
- c Indexed access with SP as base LDR R0, [SP, #XXX]
- d deallocation ADD SP

```
In EQU 20    ; input param on stack
L1 EQU 0     } local variables
L2 EQU 4
```

correction to lecture video, local variable does not have to start at 0, depends on the code

```
Sub
  PUSH {R10, R11, LR}  ✓
  SUB SP, #8 ; allocate ✓
  ~~PUSH {R1}~~
```

```
LDR R11, [SP, #In]  } access
STR R11, [SP, #L2]
 ADD SP, #8          ; deallocate
POP {R10, R11, LR}
```

STACK FRAME

SP/FP

offset	
0	L1
4	L2
8	R10
12	R11
16	LR
20	In

lower memory

higher memory

✩ LOCAL VARIABLES (LAB)

●	local variables
●	pushed registers
●	input parameter

binding

```
In  EQU  ??(a)??  ;32-bit value that is the input parameter
 L1  EQU  ??(b)??  ;32-bit local variable
 L2  EQU  ??(c)??  ;32-bit local variable
Subroutine
      PUSH {R10,R11,LR}
      ***(d)****   ;allocate L1, L2
;---------start of body------------------
      LDR  R11,[SP,#In]  ;Reg R11 is the input parameter data
      STR  R11,[SP,#L2]  ;save parameter into local L2
;----------end of body--------------------
      ???(e)???   ;deallocate L1,L2
      POP  {R10,R11,PC}
```

needs 4

32-bit

access

a = 20
b = 0
c = 4

d = SUB SP,SP,#8

ADD SP,SP,#8

note the # In

Stack frame

lowest address

0	L1 ← SP
4	L2
8	R10
12	R11
16	LR
20	In

registers are 32-bit

First thing pushed to stack

- FALL 2013 Question 2

- Relevant to use of loc var :
  1) Binding
  2) Allocation
  3) Indexed access with SP as base
  4) Deallocation

* not parameter passing

Synchronization → Busy wait using flags

I2C3_MCS_R   bit 0   indicates busy

⓪ Set up LCD

① I²C Driver

Some tips :

① Left shift R0 by 1
it is a 7 bit address and needs to go in MSA [7:1] not MSA [6:0]

② R0, R1, R2 don't overwrite them, they are your input parameters

③ MCS

2	1	0
stop	start	enable

② OutString

super straightforward, iterate through ptr(null-term) and call SSD1306_OutChar with character at ptr.

③ IO

Heartbeat / Check for switch press

Need to utilize loc var in both Outfix & OutDec

④ LCD Outfix

since you know the fixed size of output

X . X X   or   * . * *   > 999

0x30 - 0x39 are conveniently 0 - 9 in ASCII

5 LCD OutDec

Recursion only required for 319H but is easier and shorter to implement using recursion

Base condition & inductive step

↓ when to start returning

↓ what to do each time it is called

```
factorial ( i ) {        <- 1
    if (i <= 1) {  } base
        return 1;
    }          4   3 * 2 & 1
    return i * factorial (i-1)
}
```

✱ Depends on how many local variables you create

func
  check your base case → return
  isolate your number
  call func with smaller number
  retrieve and output your character

6/5/5/3/5 → local variable

Outdec (65535)

Outdec $(655\underbrace{3}_{})$ $\rightarrow$ local variable

< 10

OutChar

$\boxed{\text{BXLR}}$