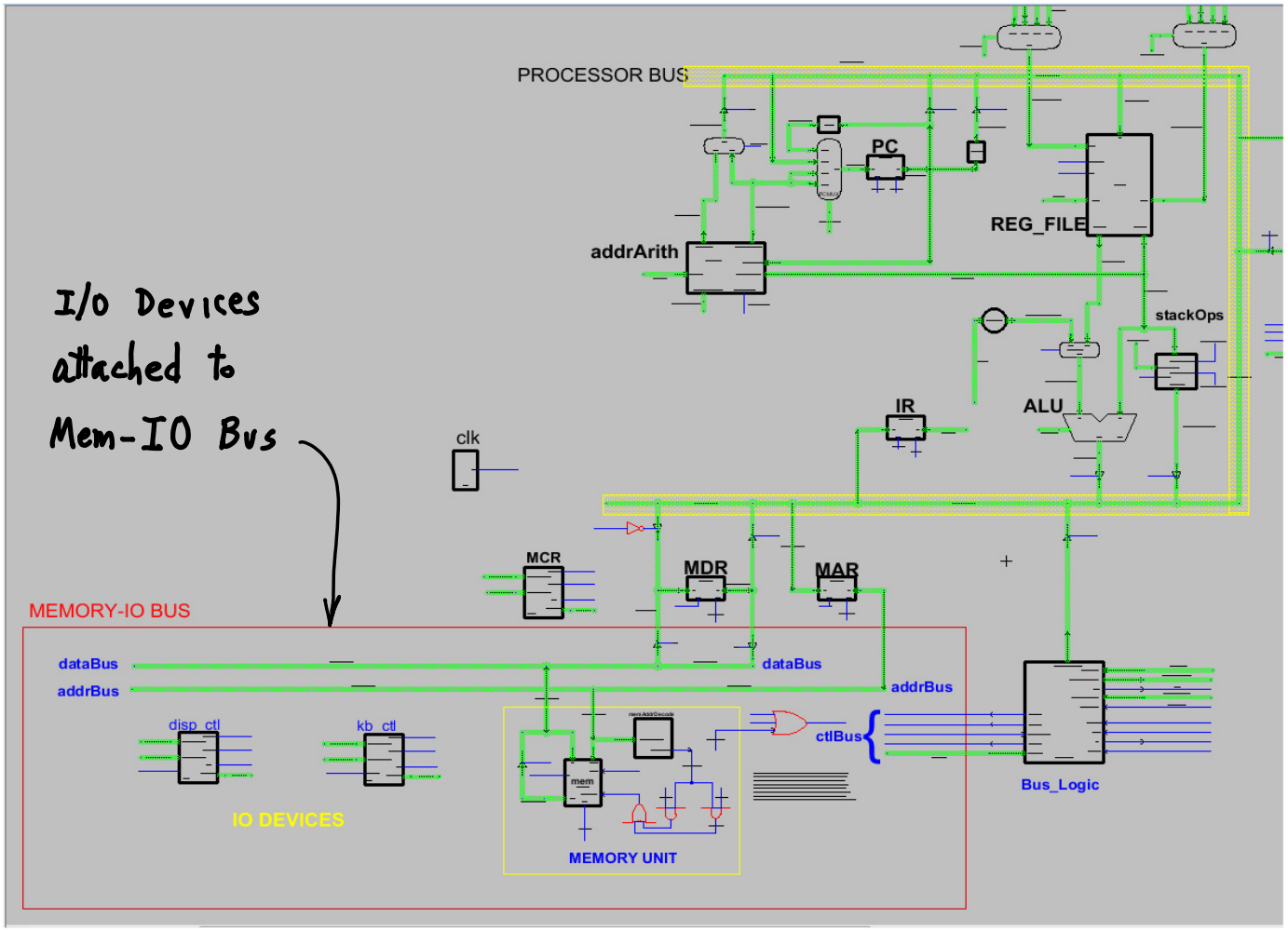
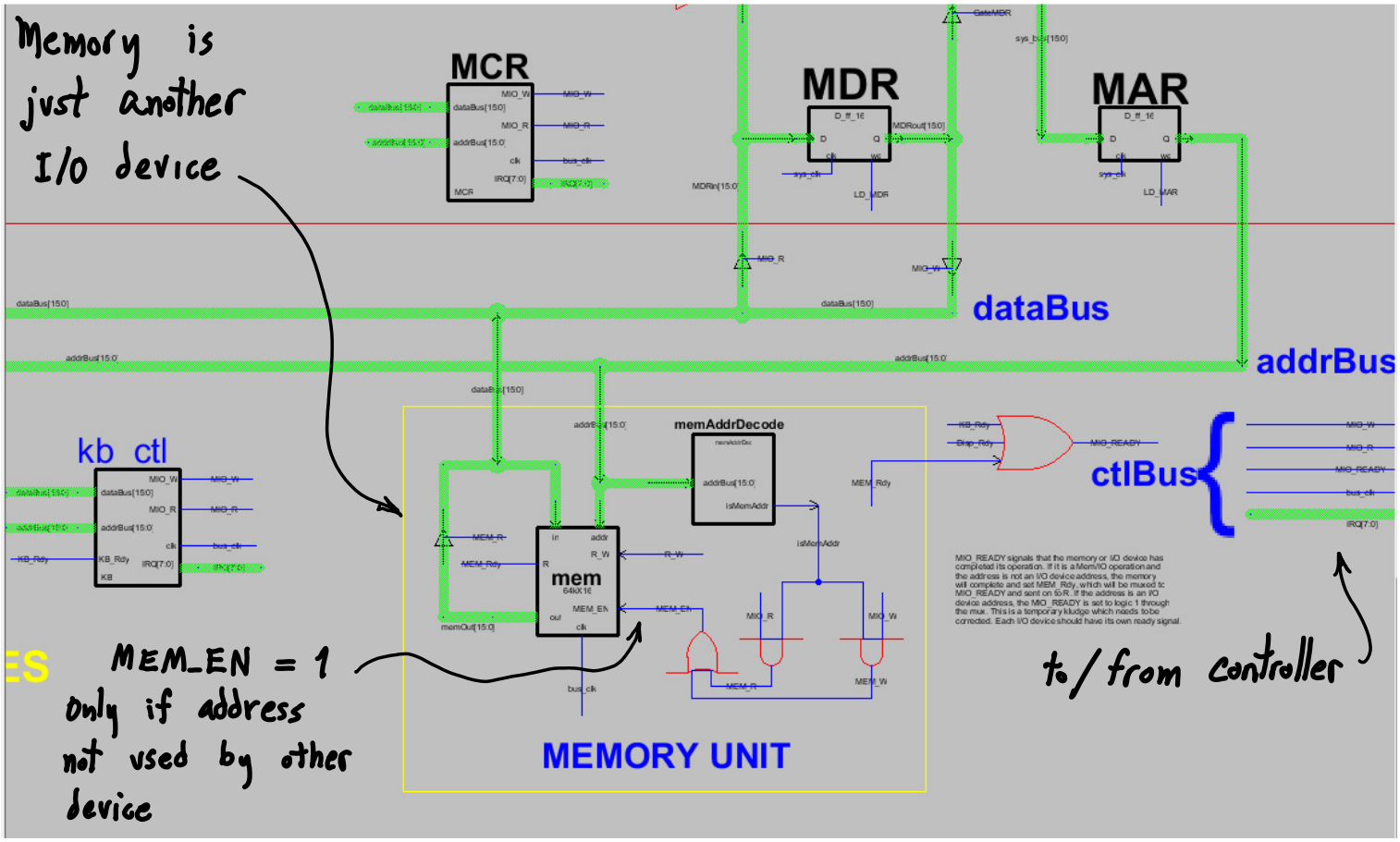
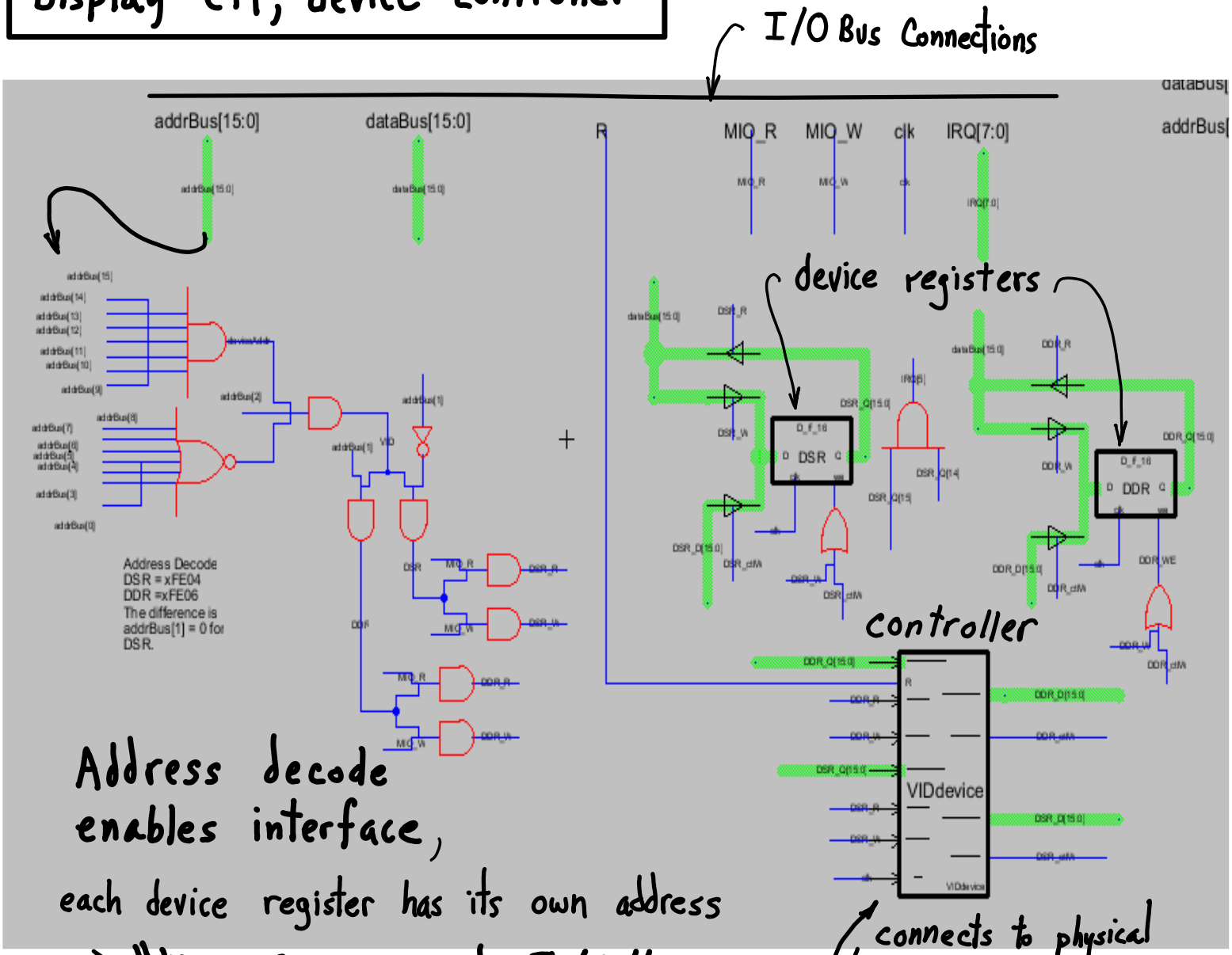
Programming Input/Output

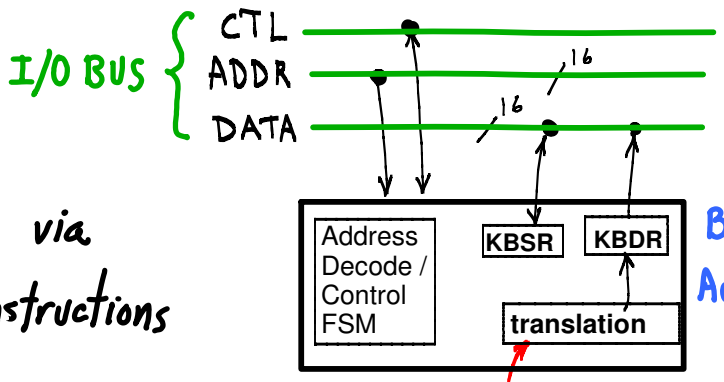




Display ctl, device controller

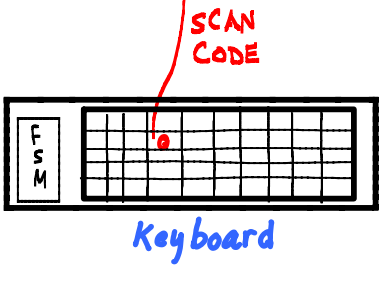


addr	mem
0000:	Trap vector
	...
01FF:	Interrupt vector
0200:	OS code/text
	...
	OS data
	...
	OS heap
	...
2FFF:	OS stack
3000:	USER code/text
	...
	USER data
	...
	USER heap
	...
FDFF:	USER stack
FE00:	Device register
	...
FFFF:	Device register



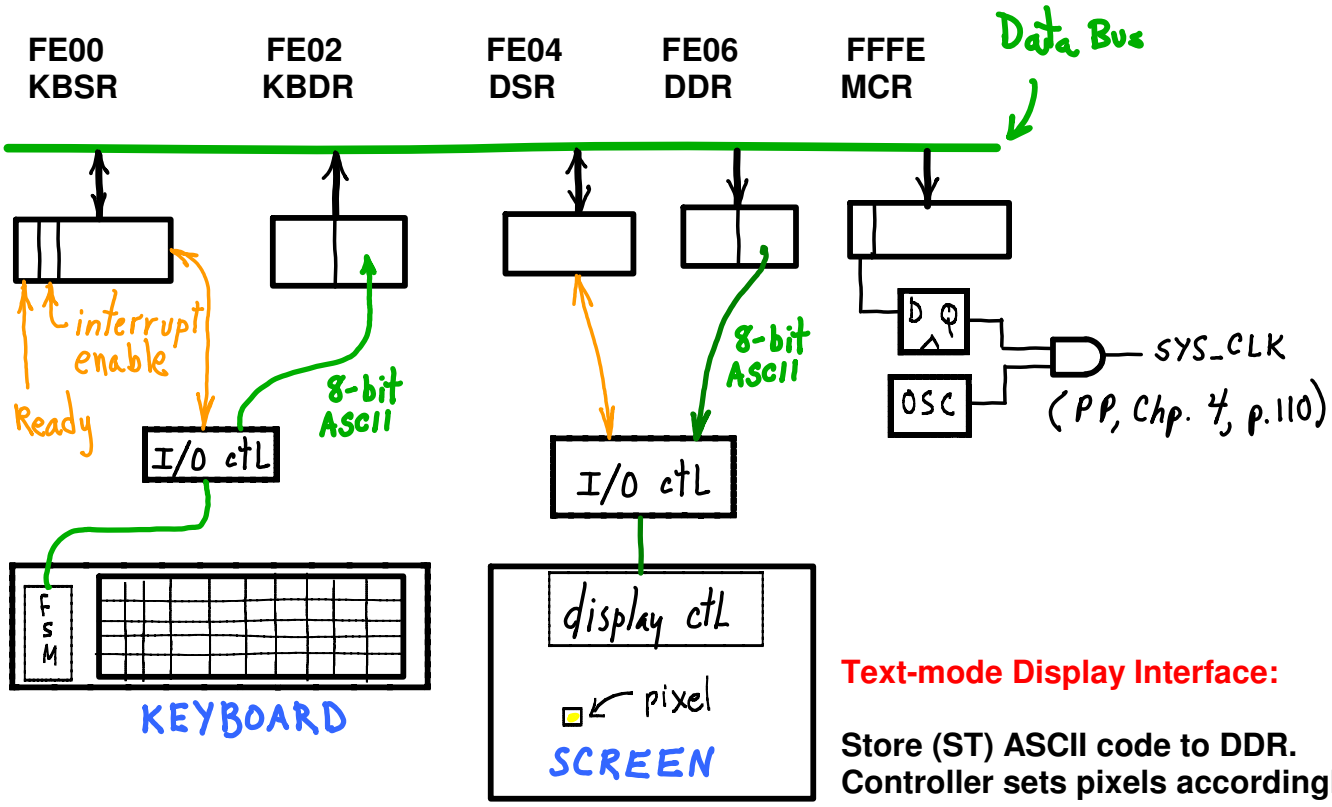
access via LD/ST instructions

Press a key:
sends scan code to kb ctl;
translates to ASCII;
loads data register, KBDR;
sets READY bit in KBSR (status).



Memory address decoder does not recognize these memory addresses.

Physical memory locations for addresses FE00 - FFFF are not accessible.



Text-mode Display Interface:

Store (ST) ASCII code to DDR.
Controller sets pixels accordingly.

Keeps track of current character position.

```
;-----------------------------
;---- keyboard access
;-----------------------------
.ORIG x3000

        LDI R0, KBDRptr
        BRnzp DONE

KBDRptr:
        .FILL xFE02

DONE:

;---------------
;--  Execution:
;-- R0 <== MEM[ MEM[ 3002 ] ]
;--           == MEM[ xFE02 ]
;--           == KBDR
```

Graphics-mode Display Interface (PennSim):

Each pixel controlled by individual memory address:

range == C000 - FFDF

Pixel at position	is controlled by location
(0, 0)	C000
(0, 1)	C001
...	
(0, 127)	C07F
(1, 0)	C080
(1, 1)	C081
...	
(123, 127)	FDFF

Pixel code == [0 RRRRR BBBBB GGGGG]

RRRRR == Red intensity (00 - 1F)
BBBBB == Blue intensity (00 - 1F)
GGGGG == Green intensity (00 - 1F)

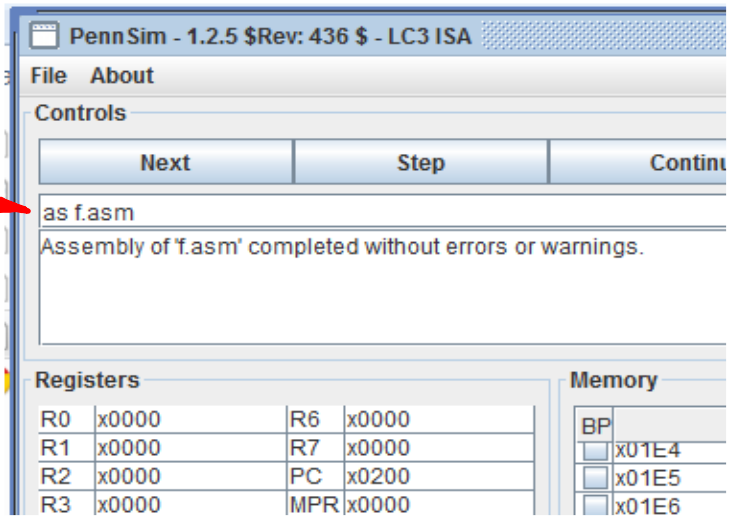
PennSim Graphics mode

f.asm
write to VRAM

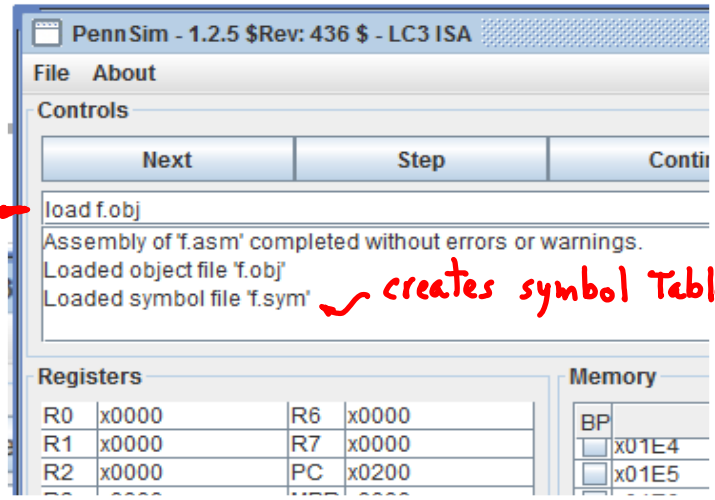
```
.ORIG x0200
    LD R4, PTR
    BRnzp MAIN
PTR:
    .FILL DATA
MAIN:
    LDR R0, R4, #0    ;-- red
    LDR R3, R4, #4    ;-- VRAM
    STR R0, R3, #0    ;-- (127, 123)
    STR R0, R3, #-1   ;-- (127, 122)
    STR R0, R3, #-2   ;-- (127, 121)
    STR R0, R3, #-3   ;-- (127, 120)
    STR R0, R3, #-4   ;-- (127, 119)
    STR R0, R3, #-5   ;-- (127, 118)
DATA:
    .FILL x7C00    ;-- 0RRR RR00
    .FILL xC000    ;-- start of VRAM
    .FILL xFDFF    ;-- end of VRAM
.END
```

assemble f.asm

(row, col)

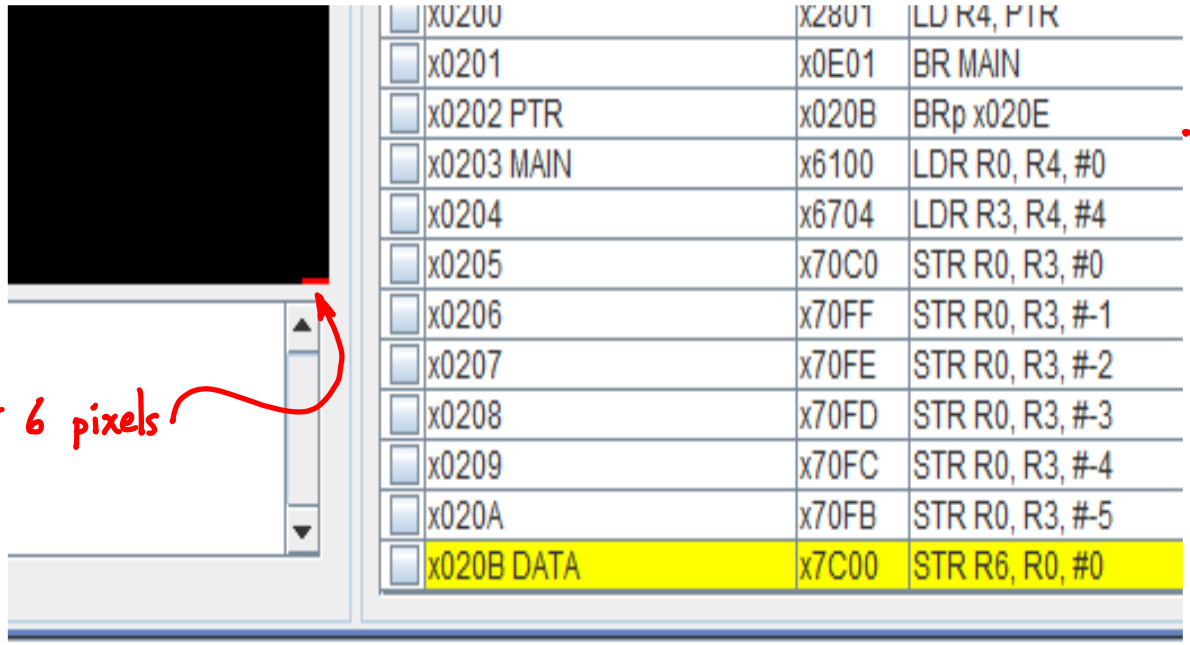


load f.obj



creates symbol Table

Device



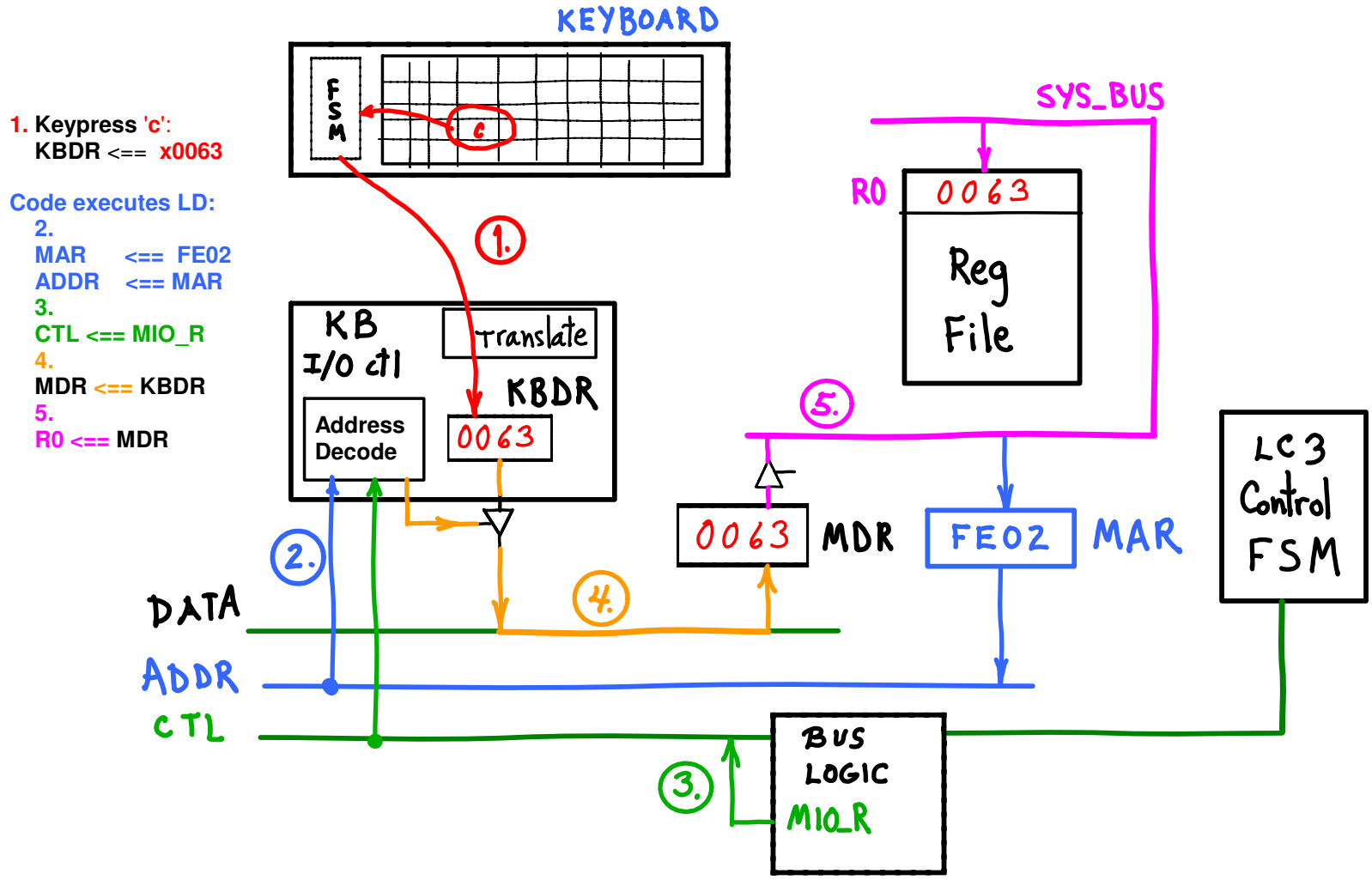
last 6 pixels

f's MAIN

The VRAM is an I/O device. Writing to the range of addresses C000-FDFF sets pixels on the display device.

PennSim shows two displays (text-mode, graphics-mode), but actually there is only one, it can be set in either mode.

Reading KBDR device register



Device Polling

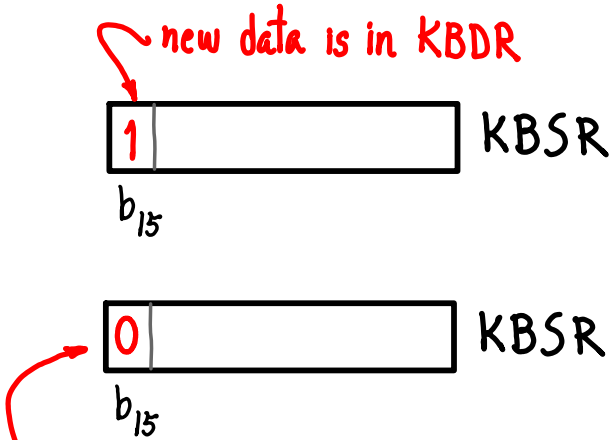
DOES KBDR have new data?

===> check KB I/O CTL: status register

```
POLL:
        LDI R1, KBSR
        BRn Ready
        BRnzp POLL
Ready:
        LDI R0, KBDR
        ...
KBSR:  .FILL xFE00
KBDR:  .FILL xFE02
```

ready

not ready



LC3-OS I/O Services

TRAP	pseudonymn	Description (PP, Append. A.4)
TRAP x25	**HALT**	**jump to OS w/ message**, loops in OS forever.
TRAP x20	**GETC**	**one char input**, keyboard data ==> R0[7:0] (clears R0 first).
TRAP x21	**OUT**	**one char output**, R0[7:0] ==> display; ignores big-end byte, R0[15:8].
TRAP x22	**PUTS**	**string output**, Mem[R0++] ==> display until x0000. 1 char per word.
TRAP x23	**IN**	**one char input w/ prompt**, ala GETC.
TRAP x24	**PUTSP**	**same as PUTS**, but packed (2 chars per word, little-end byte then big-end byte).

Non-Memory Mapped I/O (Intel x86 IA-32 and IA-64 ISAs)

--- Separate opcodes,

in R4, x0D
out R4, xC7

--- Separate address spaces,

Control-Bus signal:
0, access memory;
1, access I/O device (but uses same address wires)