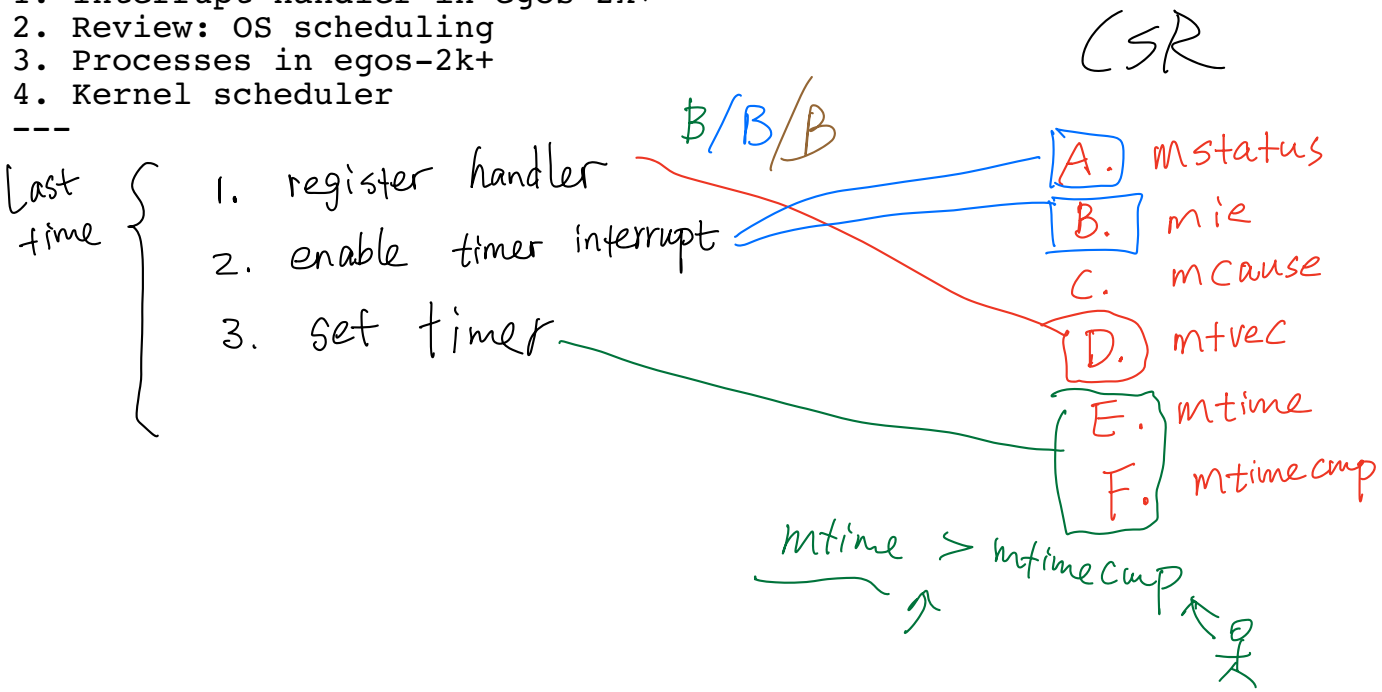
Week 5.a
CS6640
02/03 2023
https://naizhengtan.github.io/26spring/

1. Interrupt handler in egos-2k+
2. Review: OS scheduling
3. Processes in egos-2k+
4. Kernel scheduler



Interrupts and Control Status Registers (CSRs)

[borrowed from Yunhao's
CS 4411/5411: Practicum in Operating Systems, 22fall]

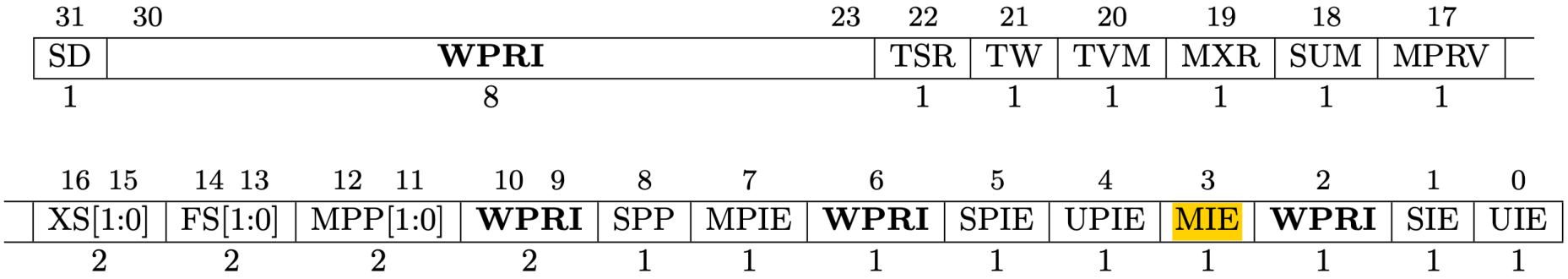
The `mtvec` CSR

Value	Name	Description
0	Direct	All exceptions set `pc` to BASE.
1	Vectored	Asynchronous interrupts set `pc` to BASE+4×cause.
≥2	—	*Reserved*

Table 3.5: Encoding of `mtvec` MODE field.

RISC-V privileged spec

The mstatus CSR



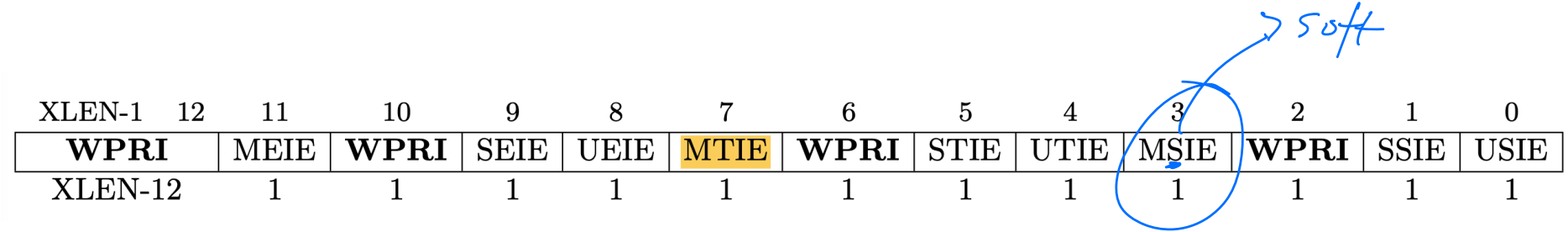
31	30 … 23	22	21	20	19	18	17
SD	WPRI	TSR	TW	TVM	MXR	SUM	MPRV
1	8	1	1	1	1	1	1

16 15	14 13	12 11	10 9	8	7	6	5	4	3	2	1	0
XS[1:0]	FS[1:0]	MPP[1:0]	WPRI	SPP	MPIE	WPRI	SPIE	UPIE	MIE	WPRI	SIE	UIE
2	2	2	2	1	1	1	1	1	1	1	1	1

MIE stands for machine interrupt enable

RISC-V privileged spec

The `mie` CSR (not `mstatus.MIE`)



XLEN-1	12	11	10	9	8	7	6	5	4	3	2	1	0
WPRI		MEIE	**WPRI**	SEIE	UEIE	MTIE	**WPRI**	STIE	UTIE	MSIE	**WPRI**	SSIE	USIE
XLEN-12		1	1	1	1	1	1	1	1	1	1	1	1

MTIE stands for machine timer interrupt enable

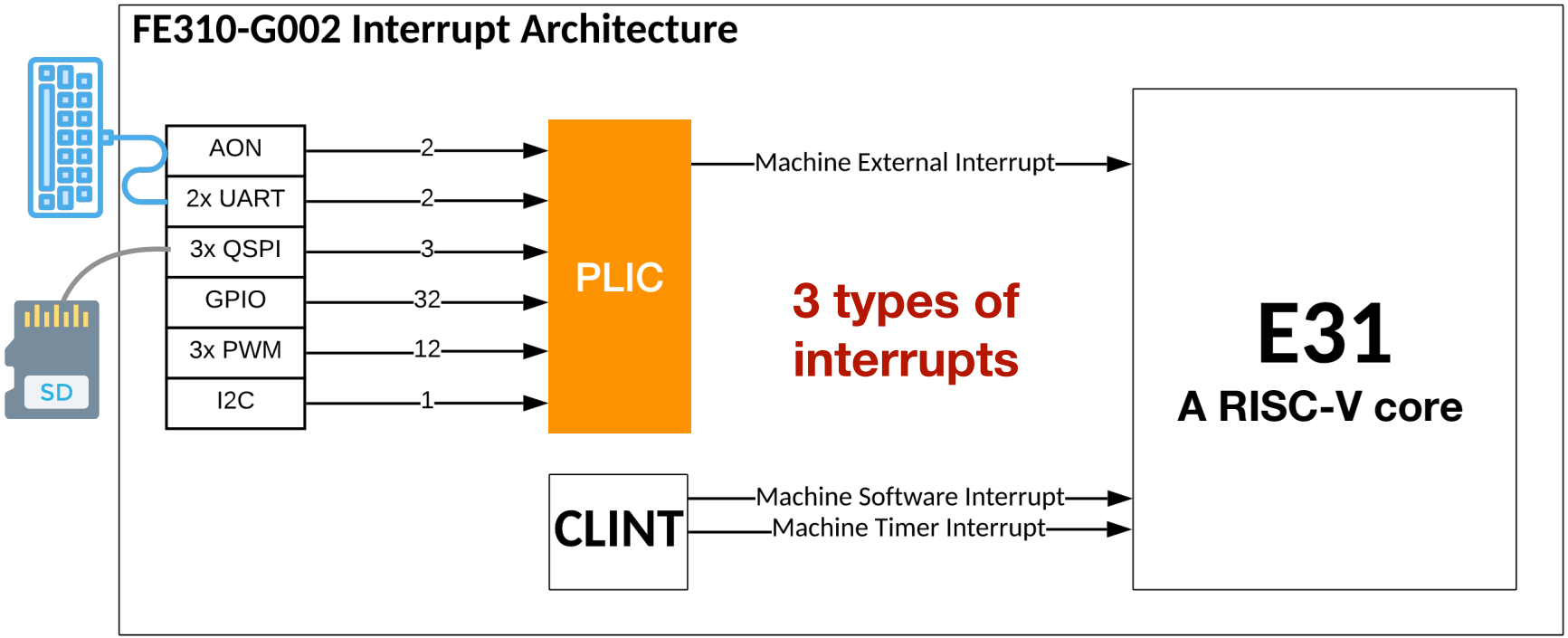
RISC-V privileged spec

Next…

- interrupts beyond CLINT (e.g., timer interrupts)
- RISC-V fine-grained control of interrupts
- how to know which interrupt is triggered?

CLIC

Platform Interrupt Controller (PLIC)



Sifive FE310 manual

`mie` provides fine-grained control

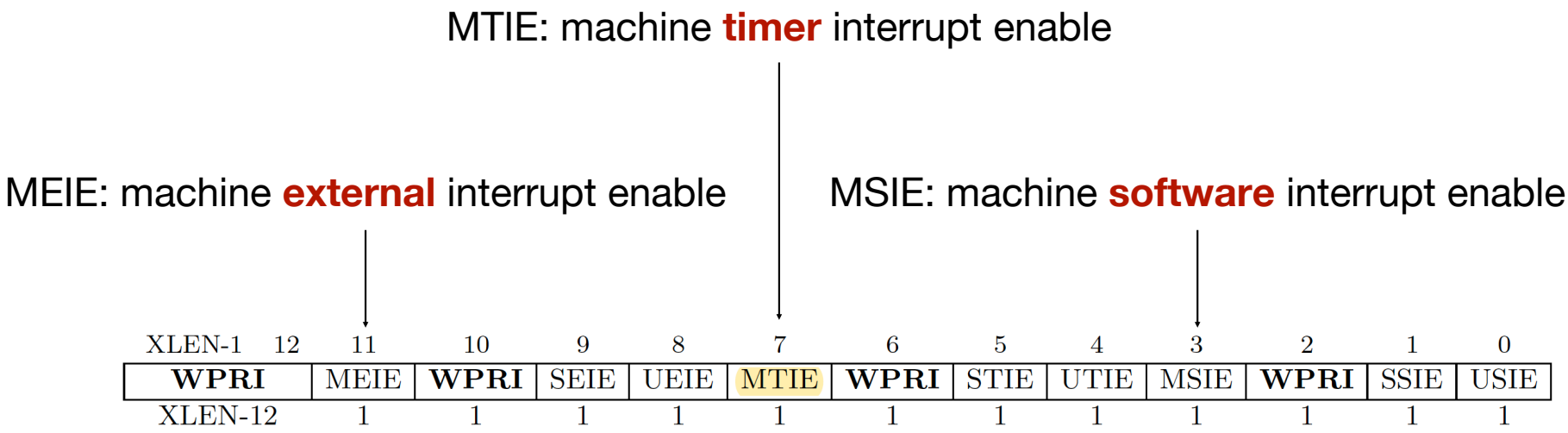


Figure 3.12: Machine interrupt-enable register (`mie`).

Timer is interrupt #7

mcause

1

Interrupts

Exceptions

Interrupt	Exception Code	Description
1	0	*Reserved*
1	1	Supervisor software interrupt
1	2	*Reserved*
1	3	Machine software interrupt
1	4	*Reserved*
1	5	Supervisor timer interrupt
1	6	*Reserved*
1	7	Machine timer interrupt
1	8	*Reserved*
1	9	Supervisor external interrupt
1	10	*Reserved*
1	11	Machine external interrupt
1	12–15	*Reserved*
1	≥16	*Designated for platform use*
0	0	Instruction address misaligned
0	1	Instruction access fault
0	2	Illegal instruction
0	3	Breakpoint
0	4	Load address misaligned
0	5	Load access fault
0	6	Store/AMO address misaligned
0	7	Store/AMO access fault
0	8	Environment call from U-mode
0	9	Environment call from S-mode
0	10	*Reserved*
0	11	Environment call from M-mode
0	12	Instruction page fault
0	13	Load page fault
0	14	*Reserved*
0	15	Store/AMO page fault
0	16–23	*Reserved*
0	24–31	*Designated for custom use*
0	32–47	*Reserved*
0	48–63	*Designated for custom use*
0	≥64	*Reserved*

syscall

keyboard

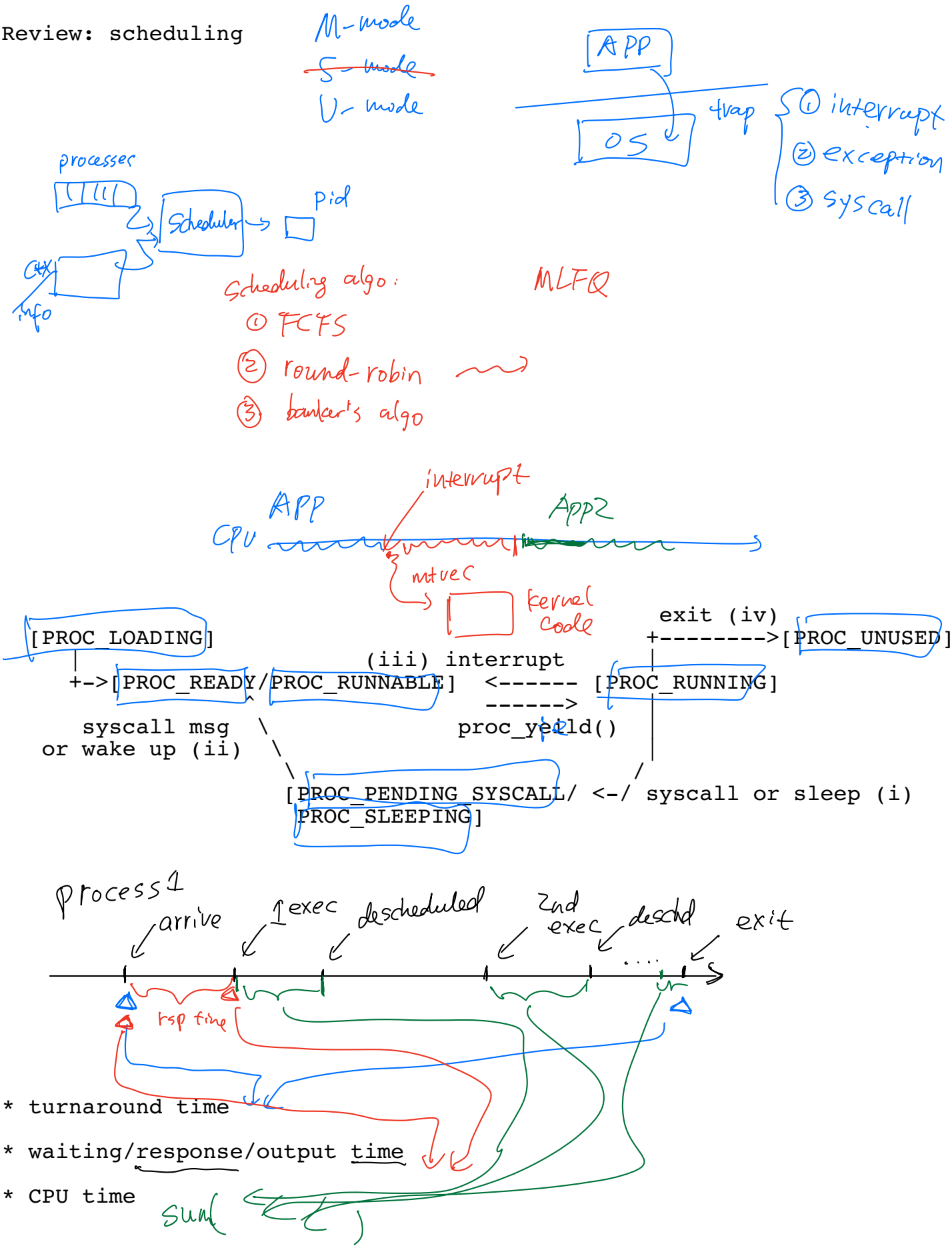
System calls?

Interrupts

Exceptions

Interrupt	Exception Code	Description
1	0	*Reserved*
1	1	Supervisor software interrupt
1	2	*Reserved*
1	3	Machine software interrupt
1	4	*Reserved*
1	5	Supervisor timer interrupt
1	6	*Reserved*
1	7	Machine timer interrupt
1	8	*Reserved*
1	9	Supervisor external interrupt
1	10	*Reserved*
1	11	Machine external interrupt
1	12–15	*Reserved*
1	≥16	*Designated for platform use*
0	0	Instruction address misaligned
0	1	Instruction access fault
0	2	Illegal instruction
0	3	Breakpoint
0	4	Load address misaligned
0	5	Load access fault
0	6	Store/AMO address misaligned
0	7	Store/AMO access fault
0	8	Environment call from U-mode
0	9	Environment call from S-mode
0	10	*Reserved*
0	11	Environment call from M-mode
0	12	Instruction page fault
0	13	Load page fault
0	14	*Reserved*
0	15	Store/AMO page fault
0	16–23	*Reserved*
0	24–31	*Designated for custom use*
0	32–47	*Reserved*
0	48–63	*Designated for custom use*
0	≥64	*Reserved*

Review: scheduling



* turnaround time
* waiting/response/output time
* CPU time

sum( )

1. Machine-mode exception CSRs

a) **mstatus**
Machine Status, holds the global interrupt enable, along with a plethora of other state.

b) **mie**
Machine Interrupt Enable, lists which interrupts the processor can take and which it must ignore

c) **mcause**
Machine Exception Cause, indicates which exception occurred

d) **mtvec**
Machine Trap Vector, holds the address the processor jumps to when an exception occurs

e) **mepc**
Machine Exception PC, points to the instruction where the exception occurred

f) **mtval**
Machine Trap Value, holds additional trap information: the faulting address for address exceptions, the instruction itself for illegal instruction exceptions, and zero for other exceptions

g) **mip**
Machine Interrupt Pending, lists the interrupts currently pending

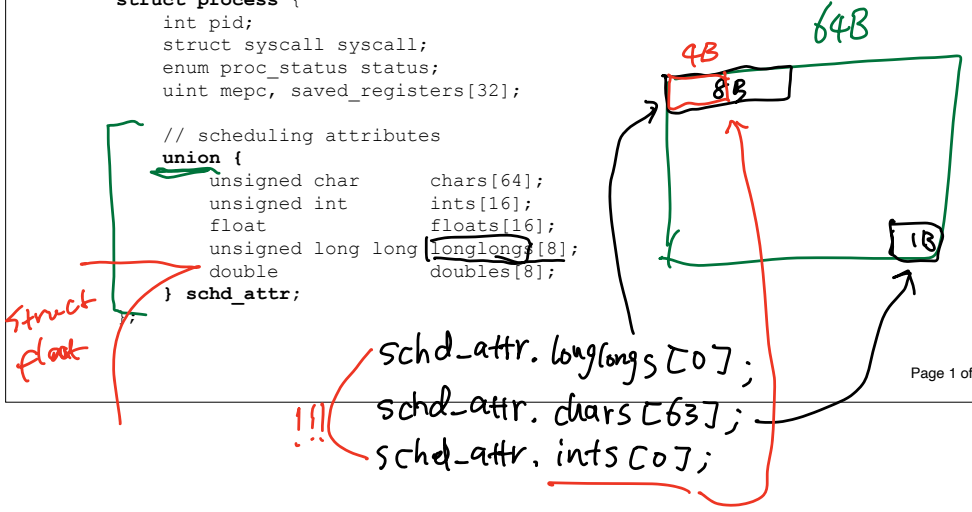
2. egos-2k+ process management

a) process control block (PCB)

[grass/process.h]

```
struct process {
    int pid;
    struct syscall syscall;
    enum proc_status status;
    uint mepc, saved_registers[32];

    // scheduling attributes
    union {
        unsigned char       chars[64];
        unsigned int        ints[16];
        float               floats[16];
        unsigned long long  longlongs[8];
        double              doubles[8];
    } schd_attr;
```



b) global process data structures

[grass/kernel.c]

```
uint core_in_kernel;
uint core_to_proc_idx[NCORES];
struct process proc_set[MAX_NPROCESS + 1];
/* proc_set[0] is a place holder for idle cores. */
```

[grass/process.h]

```
#define curr_proc_idx  (core_to_proc_idx[core_in_kernel])
#define curr_pid       (proc_set[curr_proc_idx].pid)
#define curr_status    (proc_set[curr_proc_idx].status)
#define curr_saved     (proc_set[curr_proc_idx].saved_registers)
```

c) process life cycle

[grass/scheduler.c]

state-transition callback functions:

* **proc_on_arrive**(int pid): when pid is created

* **proc_on_sched_in**(int pid) when pid is scheduled to run

* **proc_on_sched_out**(int pid) when pid is descheduled

* **proc_on_stop**(int pid): when pid is destroyed

a process's life cycle:

```
proc_on_arrive() ->
  proc_on_sched_in() -> [running] -> proc_on_sched_out() -> [others] ->
  proc_on_sched_in() -> [running] -> proc_on_sched_out() -> [others] ->
  ...
-> proc_on_stop()
```

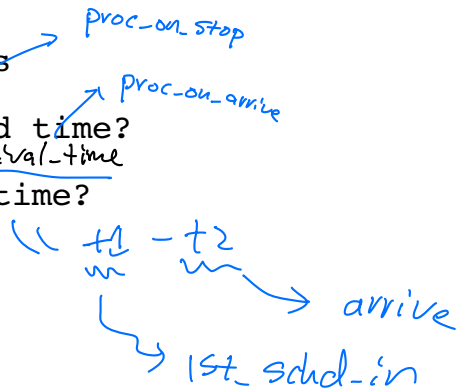
* Calculating scheduling metrics

  Q: how to calculate turnaround time?

  turnaround_time = finish_time − arrival_time

  Q: how to calculate response time?

  Q: how to calculate CPU time?



- Kernel scheduler & MLFQ