Timer and Interrupt Handling

High-level roadmap

- [basic RISC-V CPU] non-preemptive multi-threading
- [+ timer interrupt] preemptive scheduling
- [+ privilege levels] protection and isolation for processes
- [+ I/O bus controllers] disk driver and file systems

Overview of a timer handler program

```
int main() {
    // register handler() as interrupt handler
    // set a timer
    // enable timer interrupt

    while(1);
}

void handler() {
    earth->tty_info("Got timer interrupt.");
    // set a timer
}
```

- How to register `handler()` as interrupt handler?
- How to set a timer?
- How to enable timer interrupt?

Control and status registers (CSR)

- There are **many** registers other than the 32 user-level ones:
  - *machine ISA*: 32-bit or 64bit?
  - *hart ID*: the ID number of a core in a multi-core CPU
  - *interrupt control: timer, …*

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.

Page 26 of RISC-V manual, volume2, v1.10
https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/riscv-privileged-v1.10.pdf

Register an interrupt handler

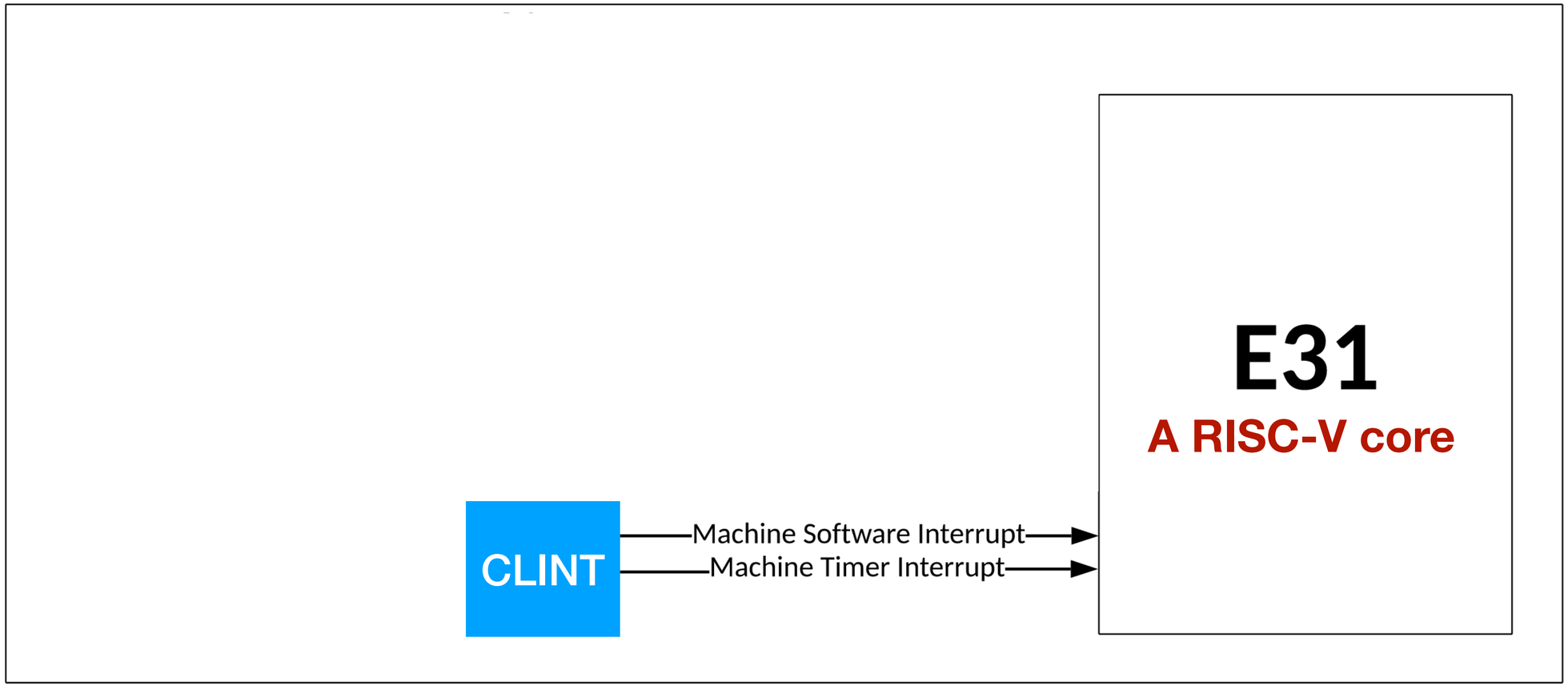
```
0800280c <handler>:
    . . .

08002914 <main>:
    . . .
    lui     a5,0x8003    # now a5 == 0x08003000
    addi    a5,a5,-2036 # now a5 == 0x0800280c
    # csrw: control and status register write
    csrw    mtvec,a5     # now mtvec == 0x0800280c
    . . .
```

Register an interrupt handler

```
void handler() {
    . . .
}

int main() {
    /* Register handler with direct mode */
    asm("csrw mtvec, %0" ::"r"(handler));
    . . .
}
```

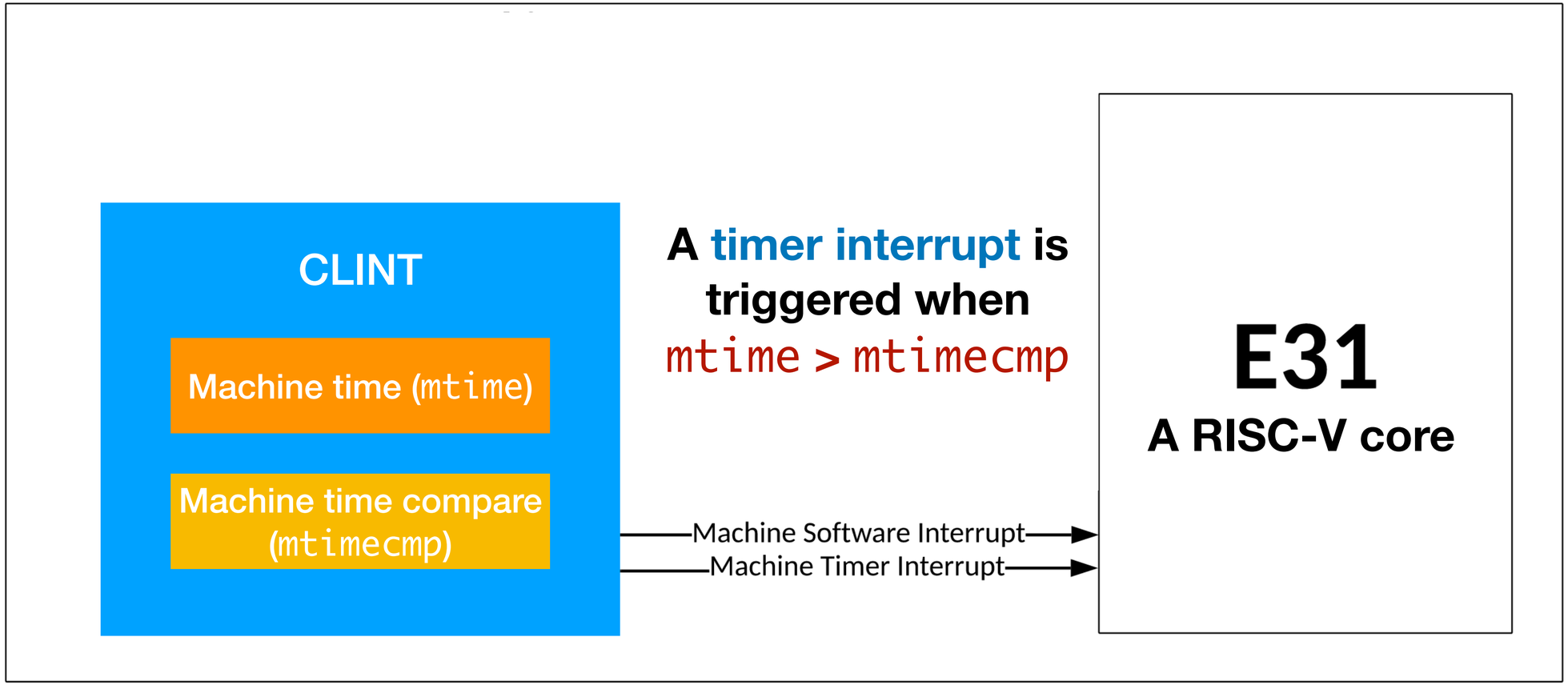
- How to register `handler()` as interrupt handler?
- How to set a timer?
- How to enable timer interrupt?

Core-local Interrupt (CLINT)



Page 38 of Sifive FE310 manual, v19p04
https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/sifive-fe310-v19p04.pdf

The mtime and mtimecmp CSRs



Page 38 of Sifive FE310 manual, v19p04
https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/sifive-fe310-v19p04.pdf

Set a timer

```
int quantum = 50000;

void handler() {
    . . .
    mtimecmp_set(mtime_get() + quantum);
}

int main() {
    /* Set a timer */
    mtimecmp_set(mtime_get() + quantum);
    . . .
}
```

CLINT CSRs are memory-mapped

`mtimecmp_set()` writes 8 bytes to

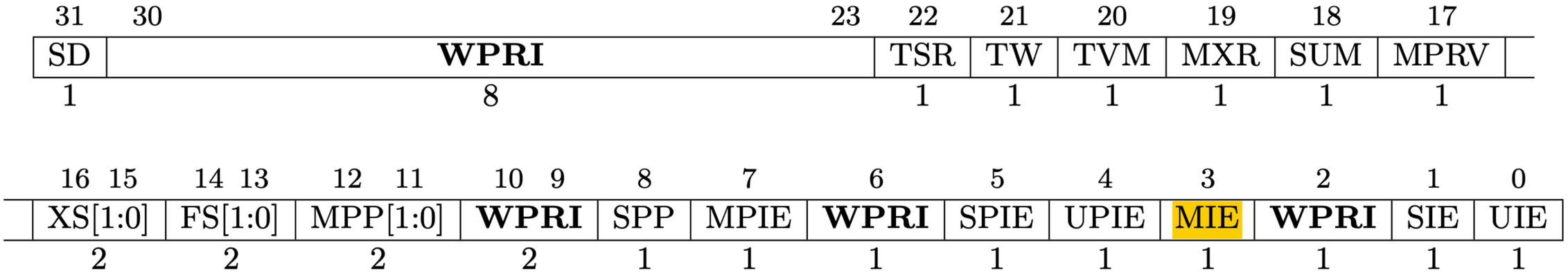
`mtime_get()` reads 8 bytes from

Address	Width	Attr.	Description
0x2000000	4B	RW	msip for hart 0
0x2004008			Reserved
…			
0x200bff7			
0x2004000	8B	RW	mtimecmp for hart 0
0x2004008			Reserved
…			
0x200bff7			
0x200bff8	8B	RW	mtime
0x200c000			Reserved

Page 44 of Sifive FE310 manual, v19p04
https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/sifive-fe310-v19p04.pdf

- How to register `handler()` as interrupt handler?
- How to set a timer?
- How to enable timer interrupt?

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

Page 20 of RISC-V manual, volume2, v1.10
https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/riscv-privileged-v1.10.pdf

Enable all (not only timer) interrupts

```
08002914 <main>:
    . . .
    csrr    a5,mstatus  # read CSR mstatus to a5
    ori     a5,a5,8     # set bit3 of a5 to 1
    csrw    mstatus,a5  # write CSR mstatus
    . . .
```

```
int main() {
    . . .
    int mstatus;
    asm("csrr %0, mstatus" : "=r"(mstatus));
    asm("csrw mstatus, %0" ::"r"(mstatus | 0x8));
   . . .
}
```

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

Page 29 of RISC-V manual, volume2, v1.10
https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/riscv-privileged-v1.10.pdf

Enable timer interrupt

```
08002914 <main>:
    . . .
    csrr    a5,mie      # read CSR mie to a5
    ori     a5,a5,128   # set bit7 of a5 to 1
    csrw    mie,a5      # write CSR mie
    . . .
```

```
int main() {
    . . .
    int mie;
    asm("csrr %0, mie" : "=r"(mie));
    asm("csrw mie, %0" ::"r"(mie | 0x80));
   . . .
}
```

Enable timer interrupt: altogether

```
void handler() {
    . . .
}

int main() {
    . . .
    int mstatus, mie;
    asm("csrr %0, mstatus" : "=r"(mstatus));
    asm("csrw mstatus, %0" ::"r"(mstatus | 0x8));
    asm("csrr %0, mie" : "=r"(mie));
    asm("csrw mie, %0" ::"r"(mie | 0x80));
    . . .
}
```

Summary of timer interrupt

- How to register an interrupt handler?
  - write the address of function `handler()` to `mtvec`
- How to set a timer?
  - write (`mtime` + `quantum`) to `mtimecmp`
- How to enable timer interrupt?
  - set certain bit of `mstatus` and `mie` to 1

A timer handler program

```
int quantum = 50000;

void handler() {
    earth->tty_info("Got timer interrupt.");
    mtimecmp_set(mtime_get() + quantum);
}

int main() {
    earth->tty_success("A timer interrupt example.");

    asm("csrw mtvec, %0" ::"r"(handler));
    mtimecmp_set(mtime_get() + quantum);

    int mstatus, mie;
    asm("csrr %0, mstatus" : "=r"(mstatus));
    asm("csrw mstatus, %0" ::"r"(mstatus | 0x8));
    asm("csrr %0, mie" : "=r"(mie));
    asm("csrw mie, %0" ::"r"(mie | 0x80));

    while(1);
}
```

Set a timer

Register handler

Set a timer

Enable timer interrupt

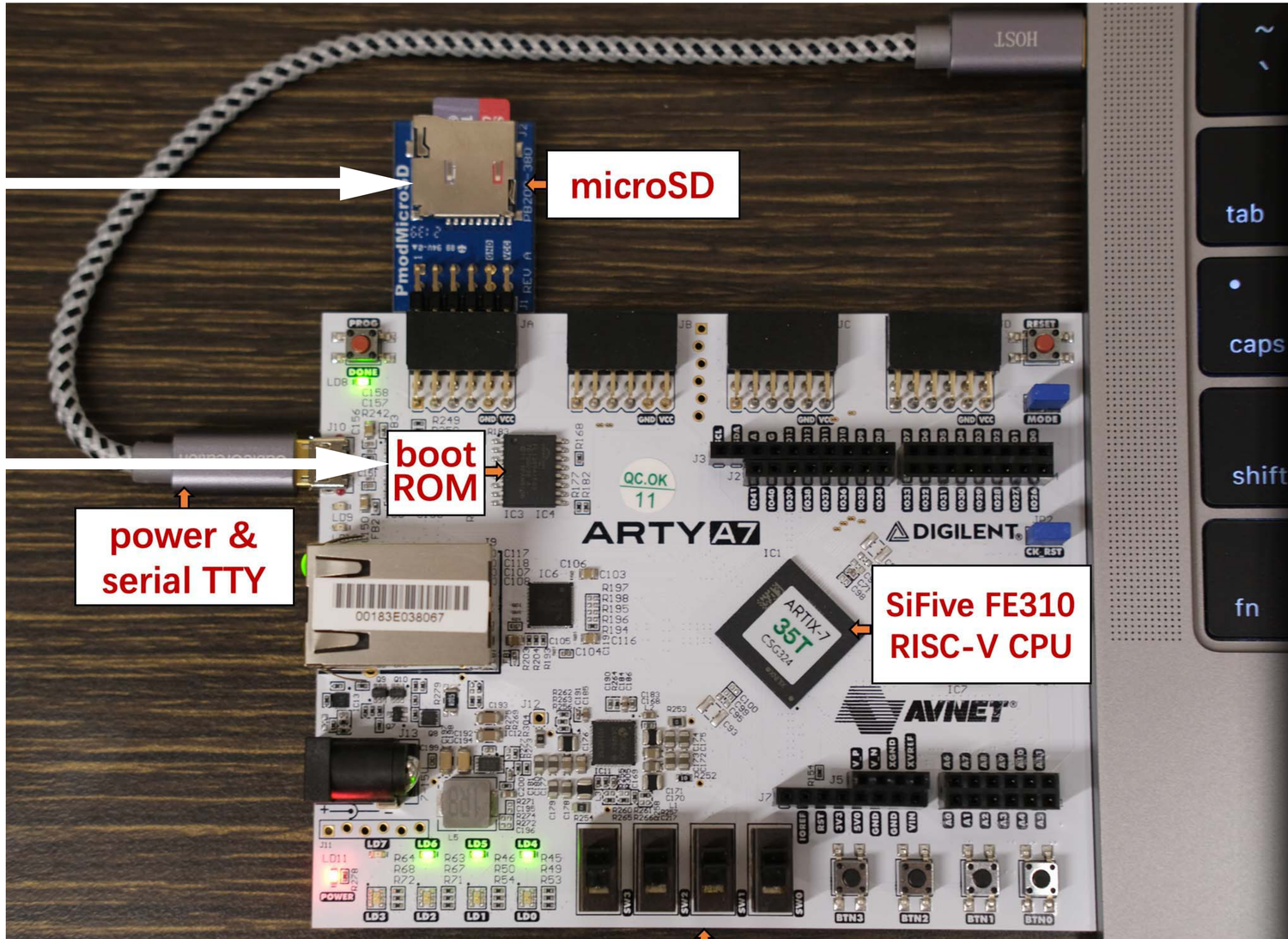
Demo on a RISC-V board

https://github.com/yhzhang0128/egos-2000/tree/timer_example/grass

demo code in microSD card

earth layer code in boot ROM

1. Load demo from microSD
2. Print strings to the screen



Guard against `mtime` rollover

When reading the lower 4 bytes, mtime is
0x00000000 0xffffffff

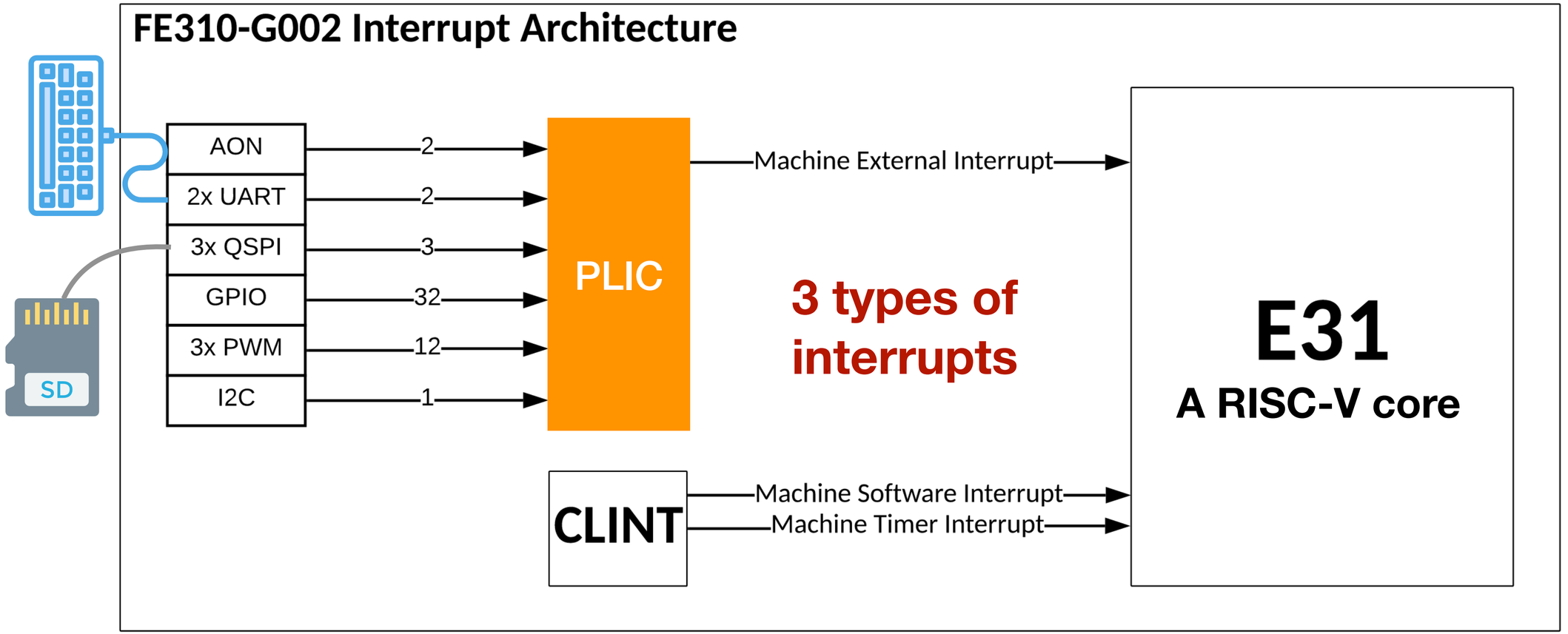
When reading the higher 4 bytes, mtime is
0x00000001 0x00000000

Combine the two 0x00000001ffffffff is wrong!

Guard against rollover

```
long long mtime_get() {
    int low, high;
    do {
        high = *(int*)(0x200bff8 + 4);
        low  = *(int*)(0x200bff8);
   }  while ( *(int*)(0x200bff8 + 4) != high );
    return (((long long)high) << 32) | low;
}

void mtimecmp_set(long long time) {
    *(int*)(0x2004000 + 4) = 0xFFFFFFFF;
    *(int*)(0x2004000 + 0) = (int)time;
    *(int*)(0x2004000 + 4) = (int)(time >> 32);
}
```

Platform Interrupt Controller (PLIC)



Page 38 of Sifive FE310 manual, v19p04

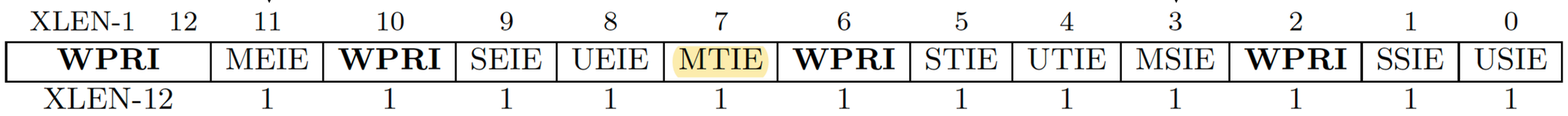
https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/sifive-fe310-v19p04.pdf

`mie` provides fine-grained control

MTIE: machine **timer** interrupt enable

MEIE: machine **external** interrupt enable

MSIE: machine **software** interrupt enable



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

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

Kernel ≈ timer handler + system call handler + fault handler

Timer is interrupt #7

Interrupts

Exceptions

Interrupt Exception Codes		
Interrupt	**Exception Code**	**Description**
1	0–2	Reserved
1	3	Machine software interrupt
1	4–6	Reserved
1	7	Machine timer interrupt
1	8–10	Reserved
1	11	Machine external interrupt
1	≥ 12	Reserved
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	9–10	Reserved
0	11	Environment call from M-mode
0	≥ 12	Reserved

Page 42 of Sifive FE310 manual, v19p04

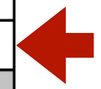
https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/sifive-fe310-v19p04.pdf

System call is exception #8

Interrupts

Exceptions

Interrupt Exception Codes		
Interrupt	**Exception Code**	**Description**
1	0–2	Reserved
1	3	Machine software interrupt
1	4–6	Reserved
1	7	Machine timer interrupt
1	8–10	Reserved
1	11	Machine external interrupt
1	≥ 12	Reserved
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	9–10	Reserved
0	11	Environment call from M-mode
0	≥ 12	Reserved



Page 42 of Sifive FE310 manual, v19p04

https://github.com/yhzhang0128/egos-2000/blob/timer_example/references/sifive-fe310-v19p04.pdf

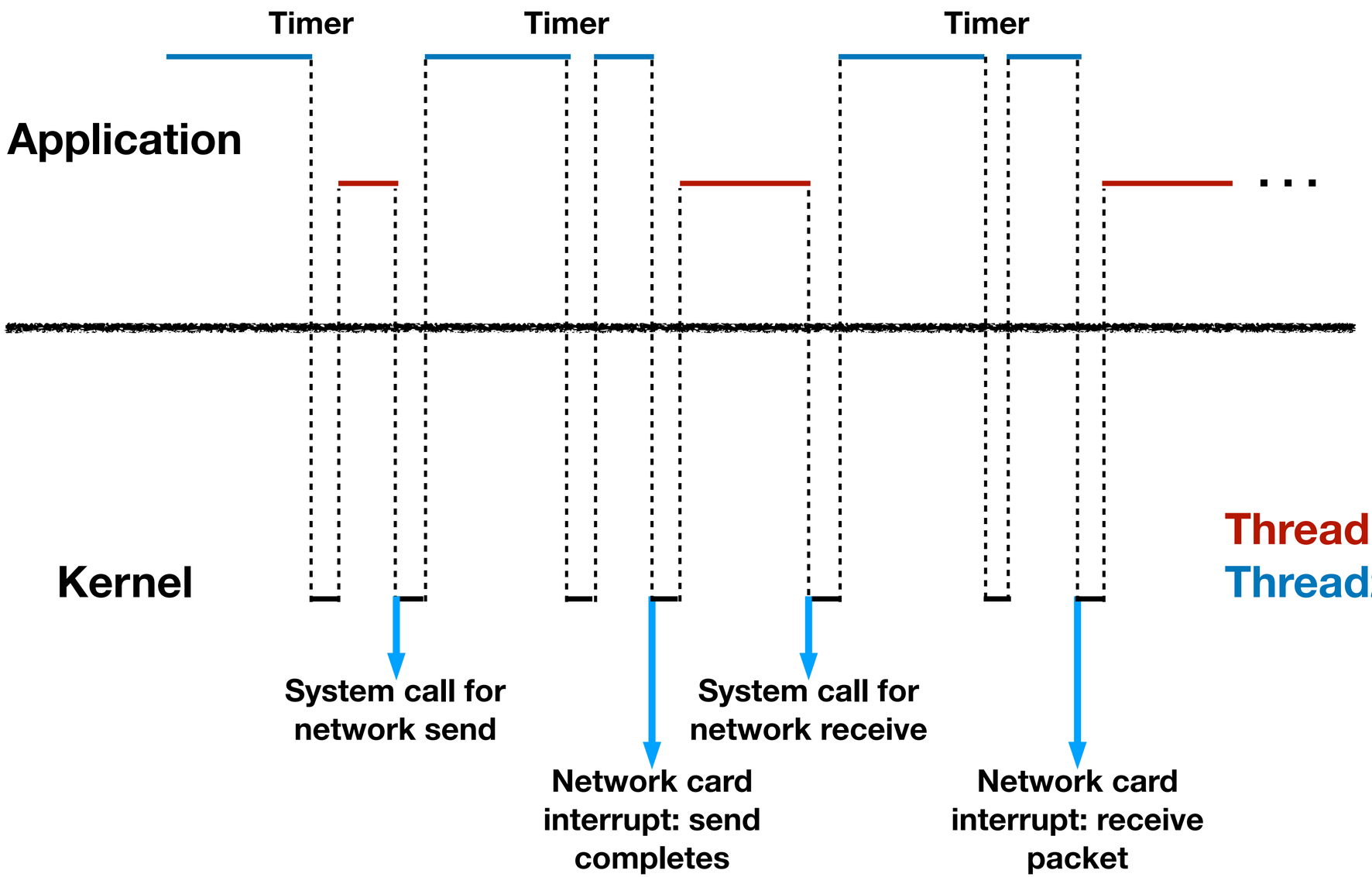
Kernel ≈ 3 handlers

```
void kernel() {  // registered to CSR mtvec
    int mcause;
    __asm__ volatile("csrr %0, mcause" : "=r"(mcause));

    int id = mcause & 0x3ff;   // take the last 10 bits
    if (mcause & (1 << 31)) {  // most significant bit is 1?
        if (id == 7) { timer_handler(); }
    } else {
        if (id == 8) { syscall_handler(); }
                else { fault_handler(); }
    }
}
```

P2 and P3 are about the kernel

```
void kernel() {
    int mcause;
    __asm__ volatile("csrr %0, mcause" : "=r"(mcause));

    int id = mcause & 0x3ff;
    if (mcause & (1 << 31)) {
        // P2: scheduling
        if (id == 7) { timer_handler(); }
    } else {
        // P3: system call and memory protection
        if (id == 8) { syscall_handler(); }
                else { fault_handler(); }
    }
}
```

Example of scheduling



Thread1: I/O-bound (zoom)
Thread2: CPU-bound (matrix computation)

Homework

- P2 has been released and it is due on Oct 12.
  - scheduling algorithm: multi-level feedback queue
- Read the 4 files of the timer handler program.
  - https://github.com/yhzhang0128/egos-2000/tree/timer_example/grass