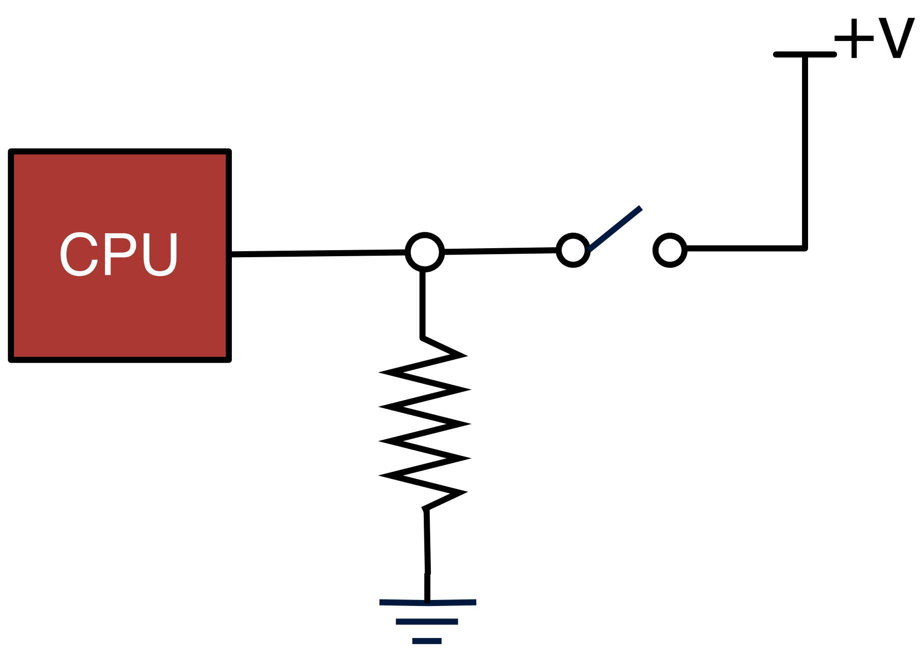


Medusa: Concurrent Programming in Surprising Places

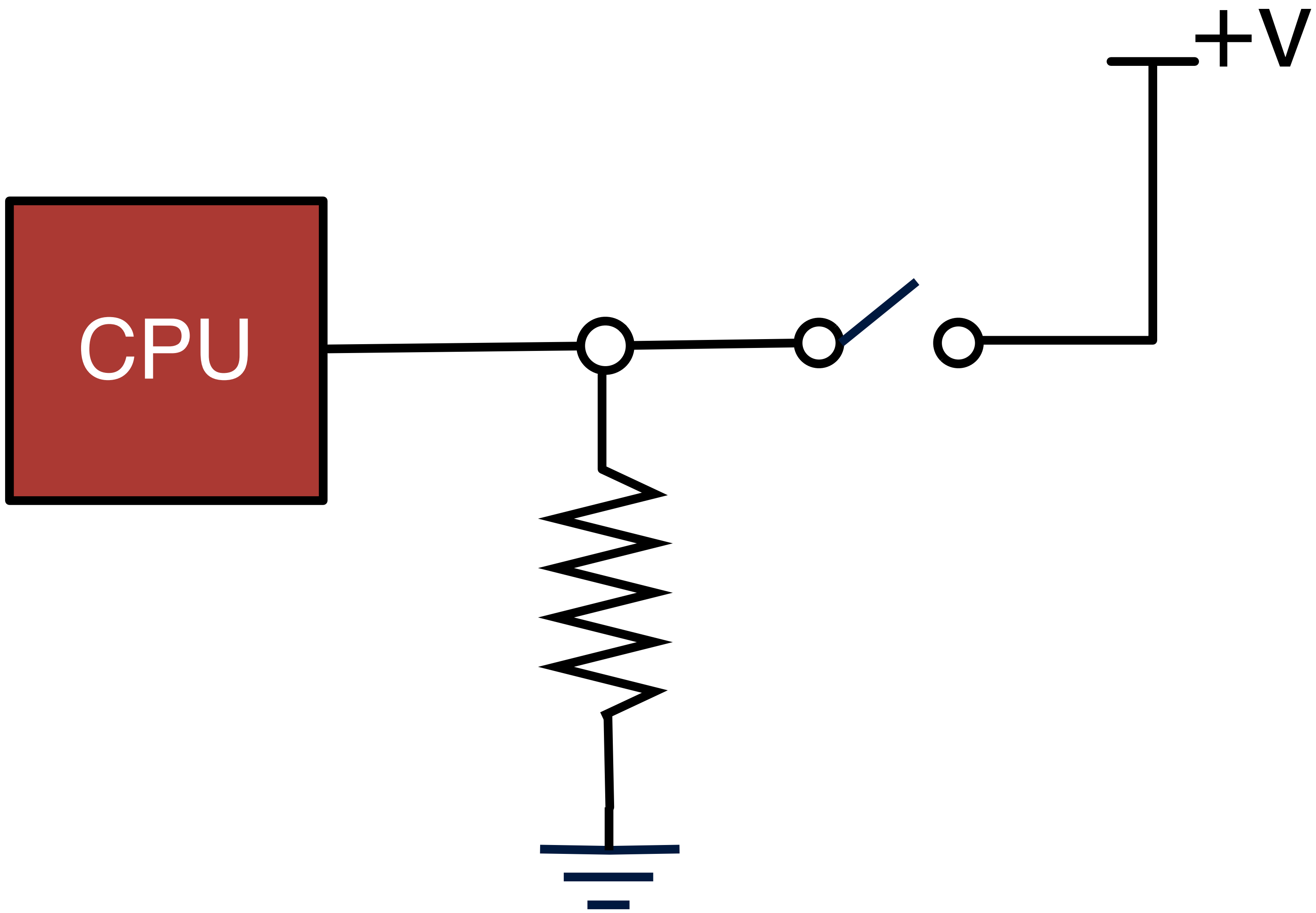
Thomas W. Barr, Scott Rixner
Rice University

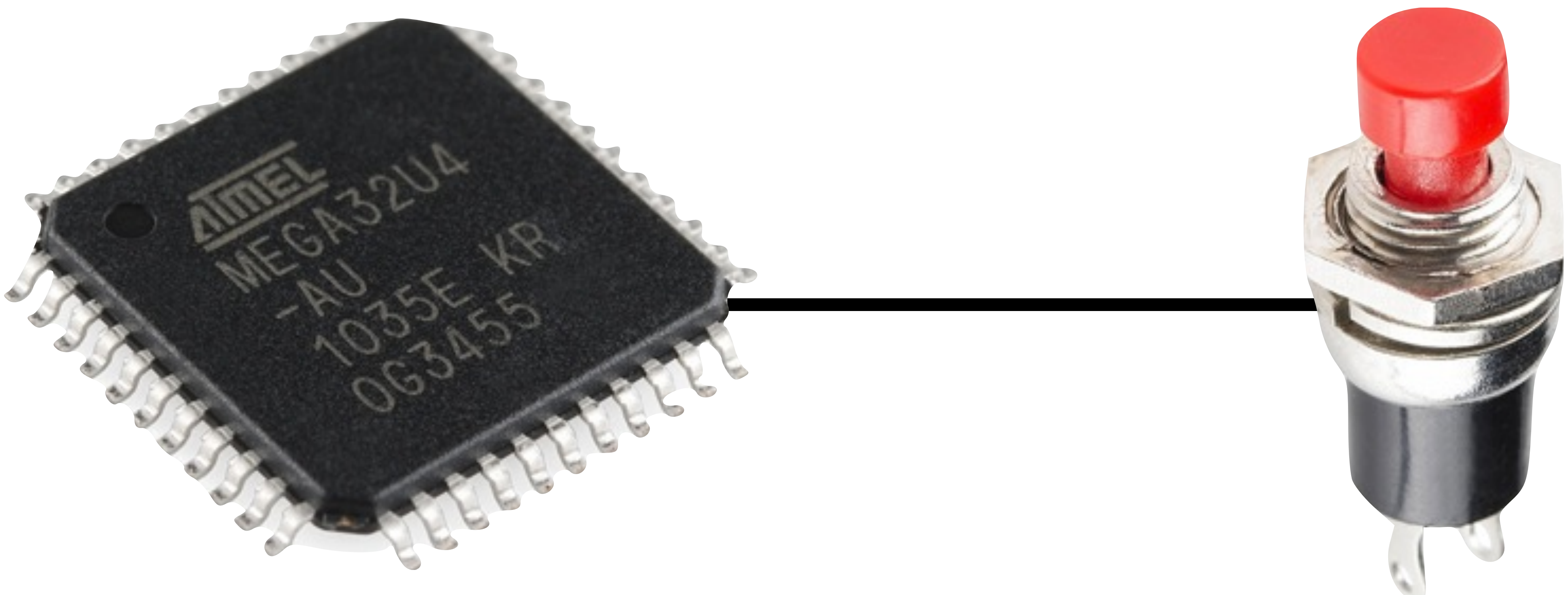
USENIX ATC 2014, June 2014

Embedded Concurrency

- **Expensive**
  - 1 car, 100m LOC [Charette09]
- **Vulnerable**
  - [Checkoway11]
- **Deadly**
  - Therac-25 [Leveson93]

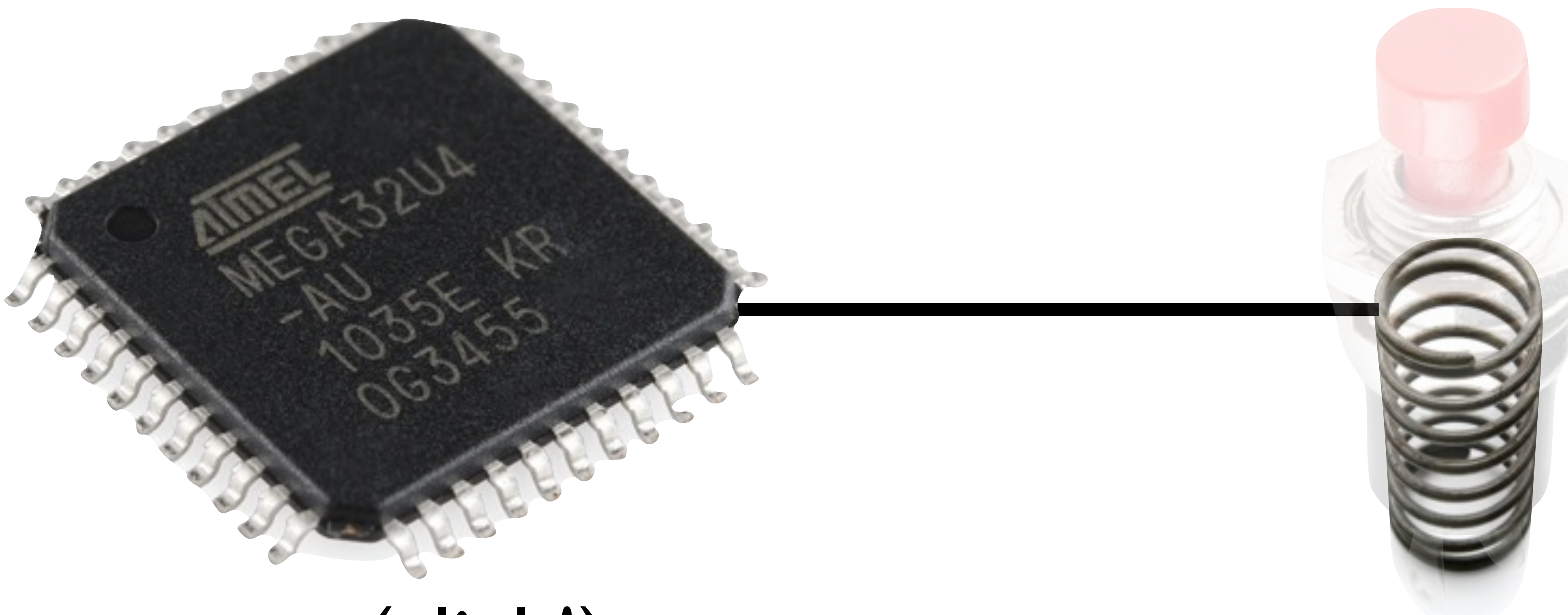
http://hci.cs.siue.edu/NSF/Files/Semester/Week13-2/PPT-Text/Slide13.html


+V
CPU

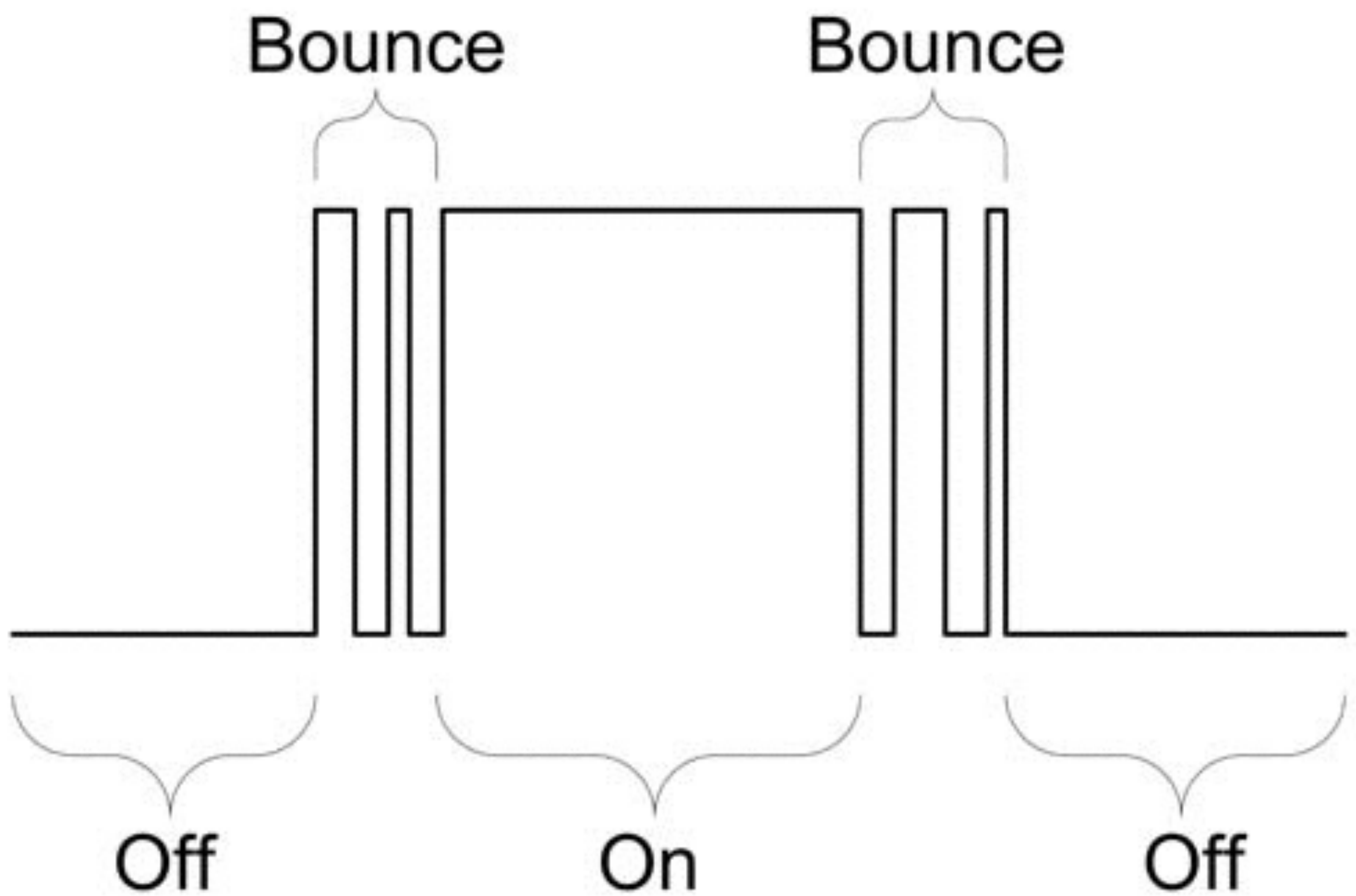


(click!)

(images from sparkfun.com, Creative Commons, BY-NC-SA)



(click!)
(click!)
(click!)
(click!)
(click!)



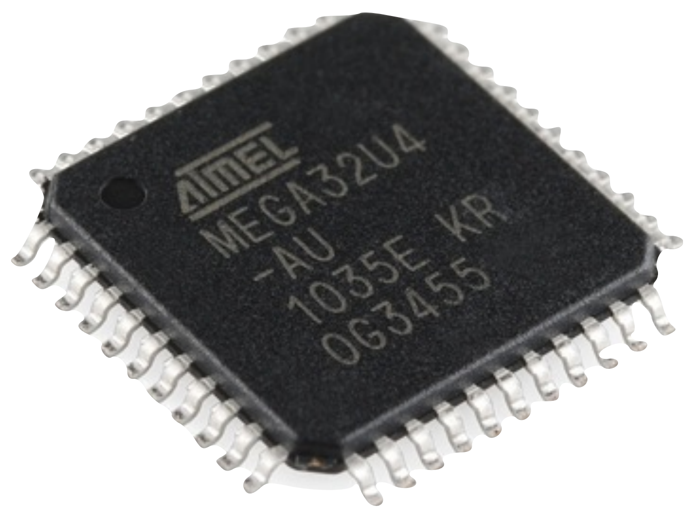
http://www.protostack.com/blog/2010/03/debouncing-a-switch/

do math

do math

do math

do math

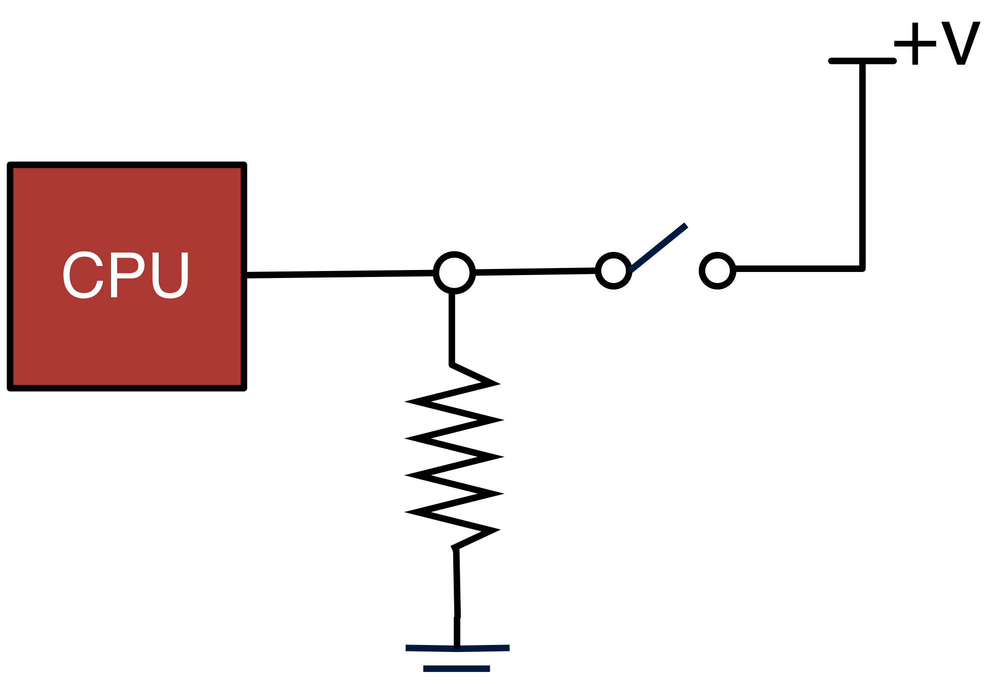


debounce():
    loop forever:
        wait for button press
        wait some time
        read button

Owl and Medusa

- **Owl Embedded Python**
  - USENIX ATC 2012
  - (embeddedpython.org)
- **Python**
  - Easy to get started
  - I/O is still hard!
- **Medusa**
  - New language
  - Actor model [Haller06]
    - Message passing

Why is I/O hard?



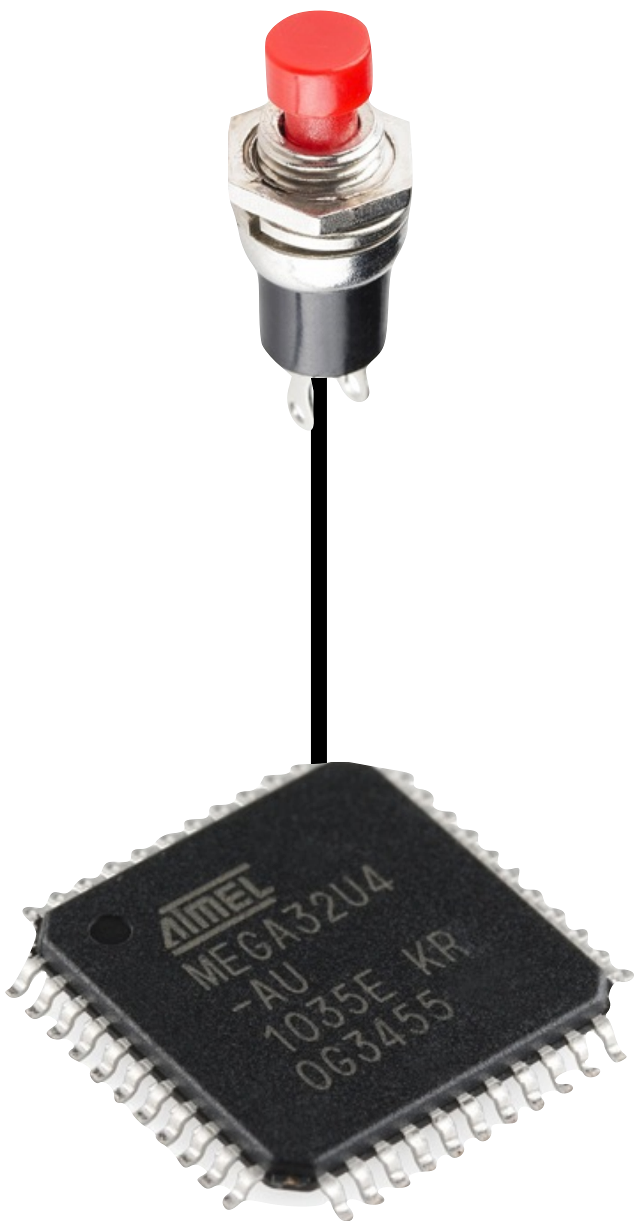
- **Polling - Python**
  - Simple
  - Slow
- **Interrupts - C**
  - Error-prone
  - Costly
- **Bridging - Medusa**
  - Ease of polling
  - Speed of interrupts

Polling I/O

```
loop forever:
    do_math()
    more_math()
    yet_more_math()
```

```
loop forever:
    while (!button_down)
    {
        // spin
    }
    set_timer()
    while (!timer_expired)
    {
        // spin
    }
    read_button()
```

Polling I/O



- **Most of the time is spent in the spin loop**
  - Wasted cycles
- **Schedule less frequently?**
  - Latency goes up
  - May miss events
    - >5000 Hz on our systems

Interrupt I/O

```
loop forever:
    do_math()
    more_math()
    yet_more_math()
```

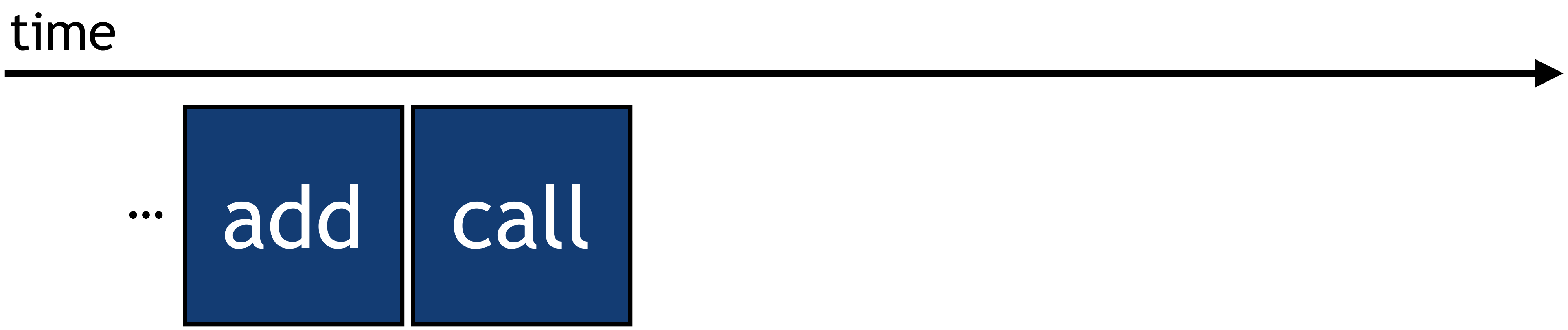
```
on button press:
    set_timer()

on timer expiration:
    read_button()
```

Interrupt I/O

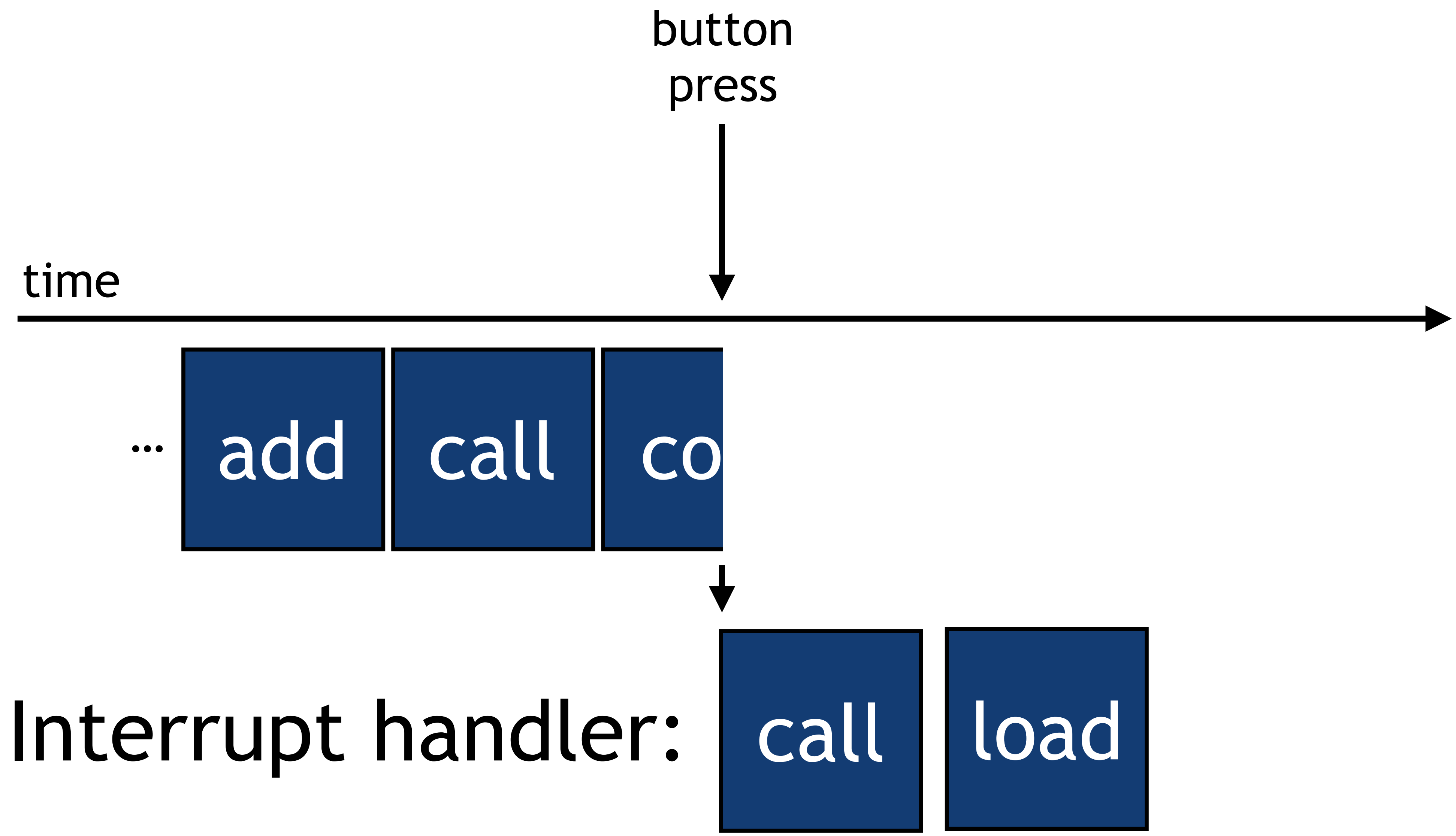
- **Current solution**
  - Used in vendor examples
  - Recommended in documentation
- **170 lines of C**

```
on button press:
    set_timer()

on timer expiration:
    read_button()
```

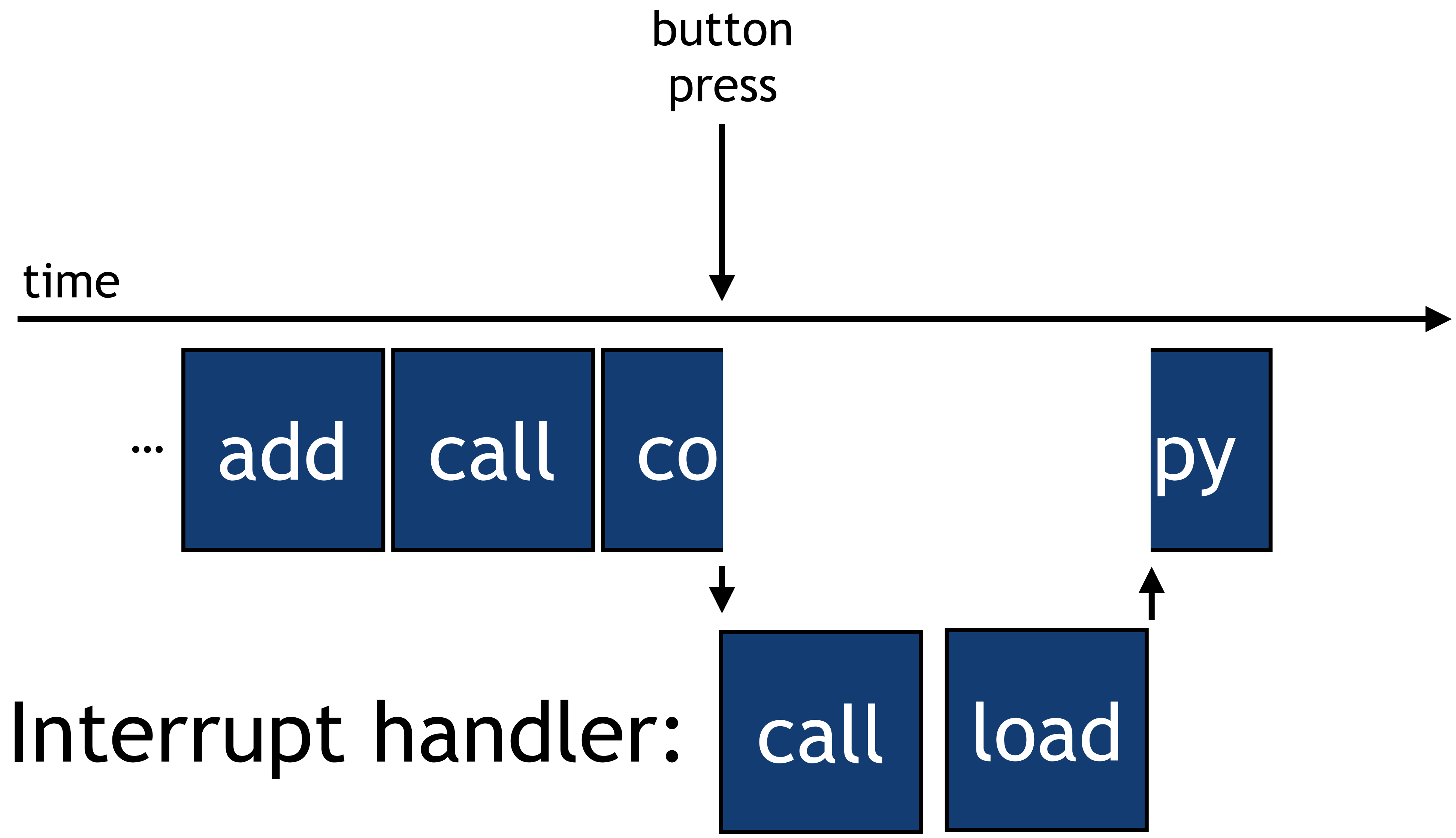
Interrupts in interpreters



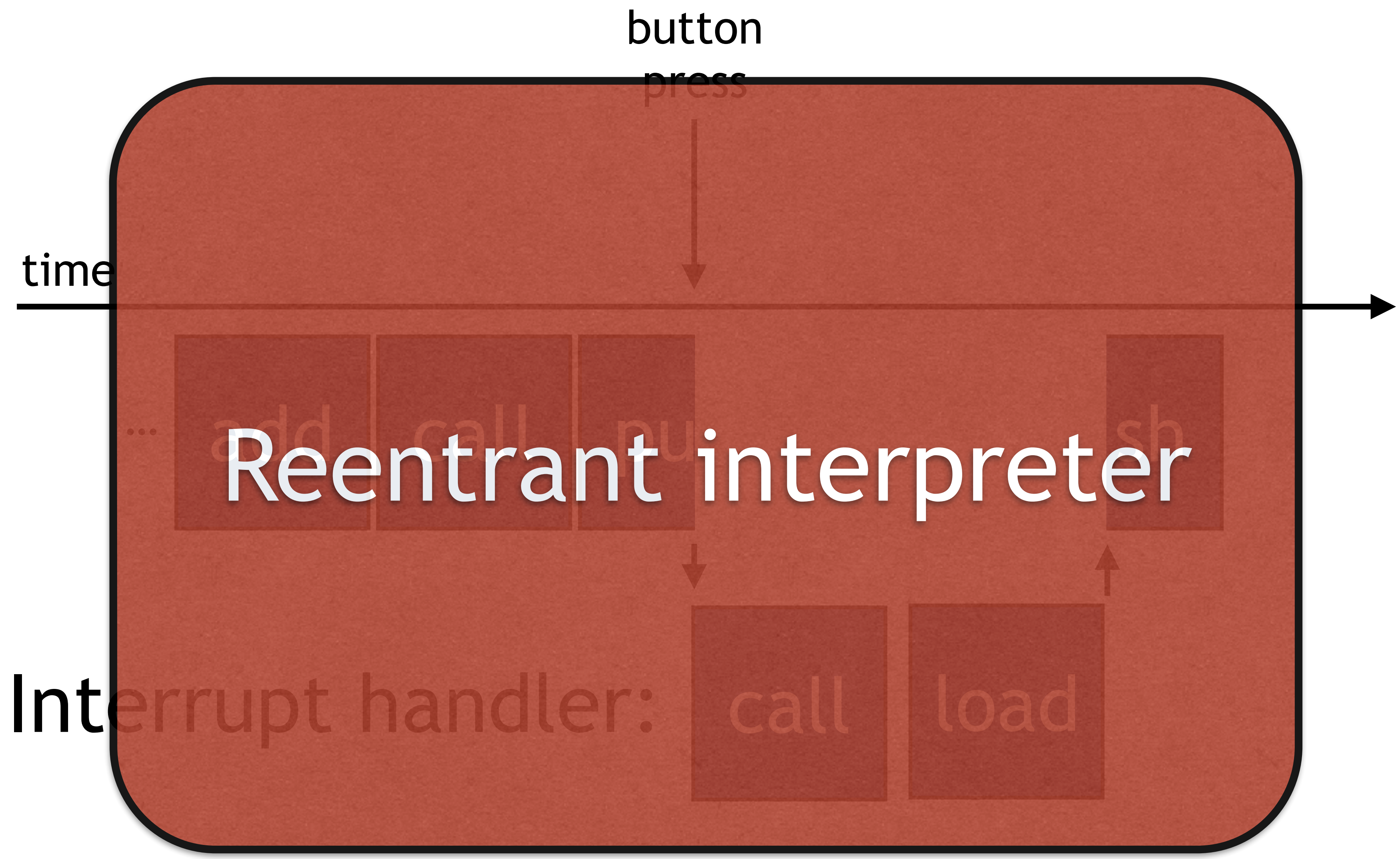
Interrupts in interpreters



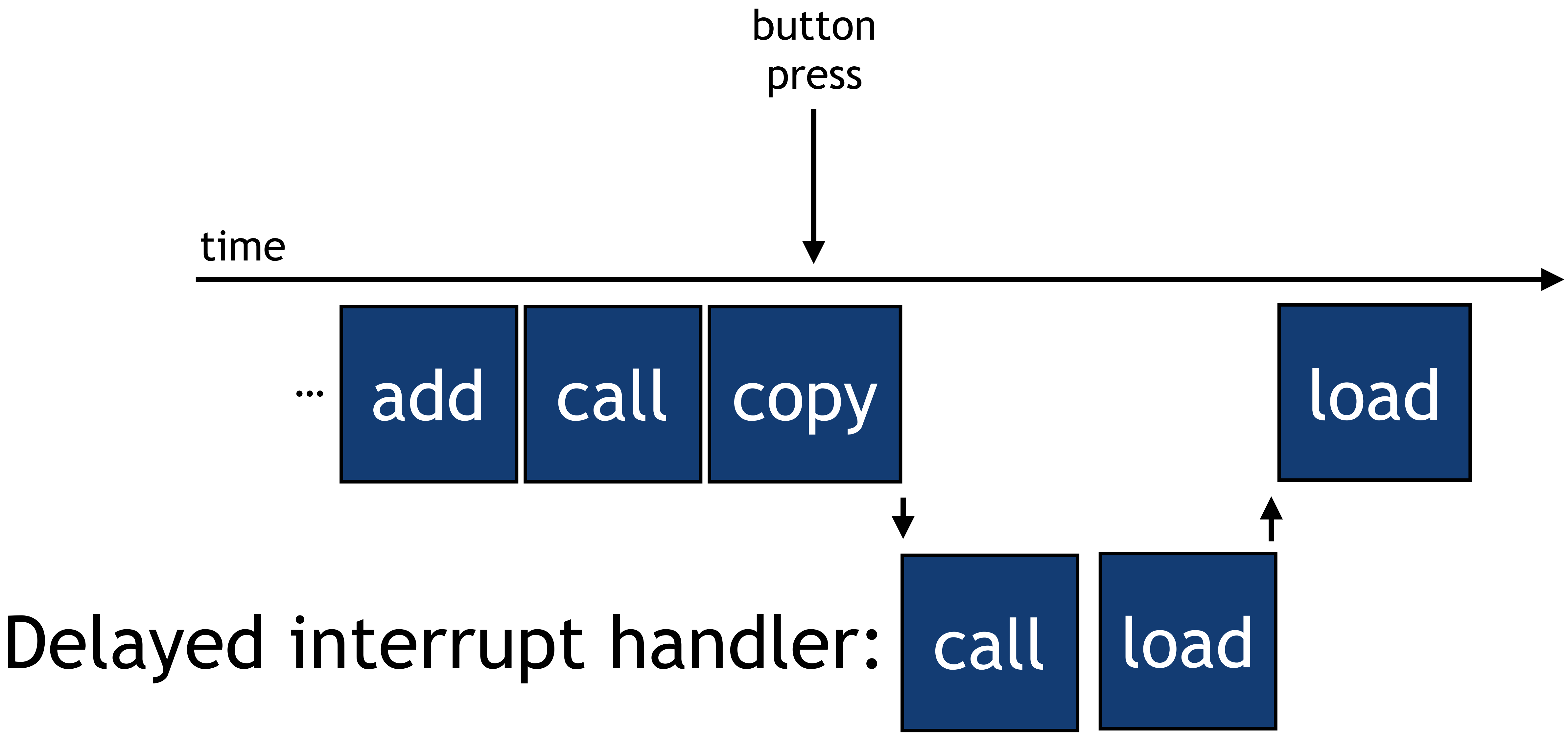
Interrupts in interpreters



Interrupts in interpreters

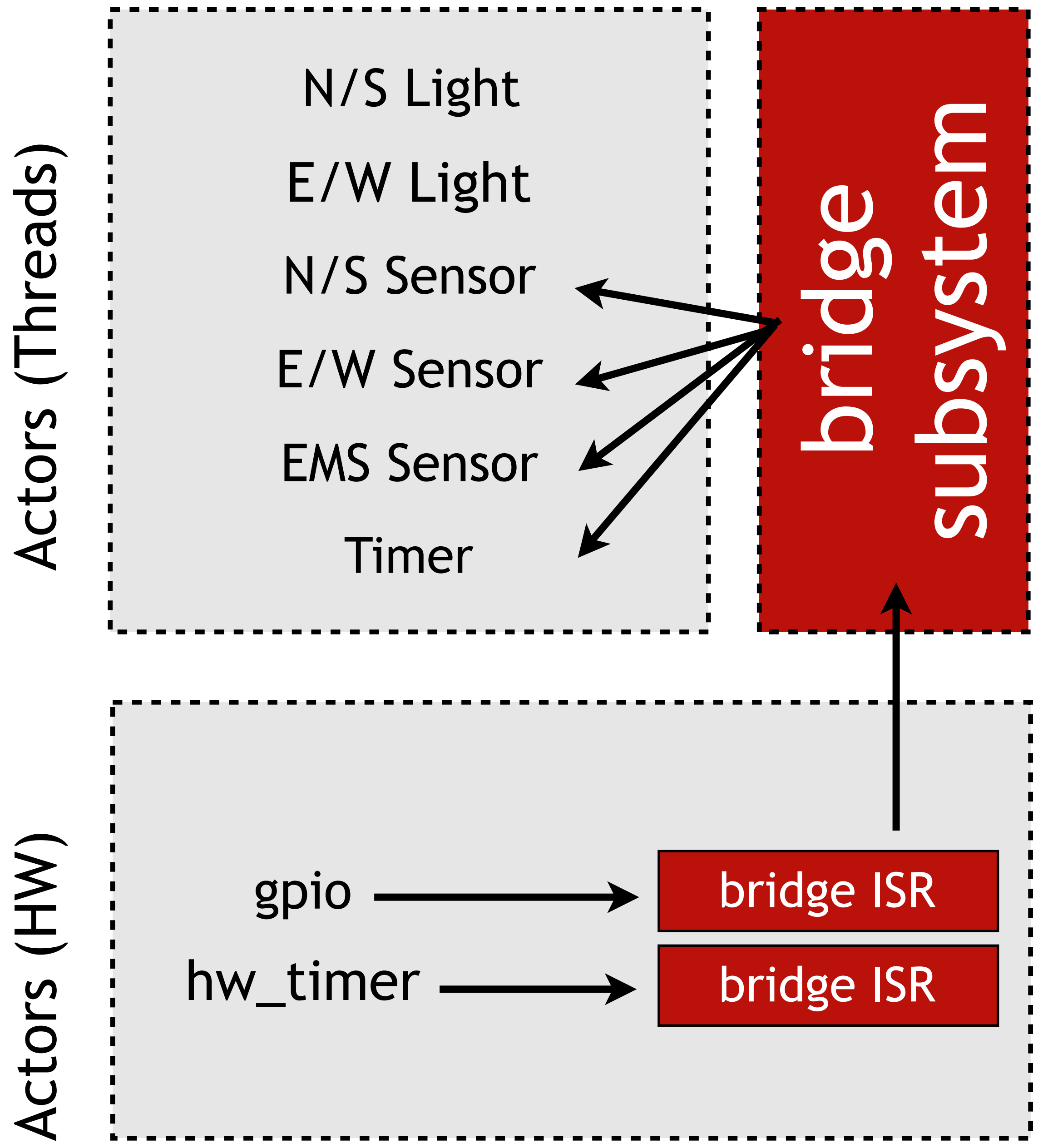


Interrupts in interpreters



Interrupts in interpreters

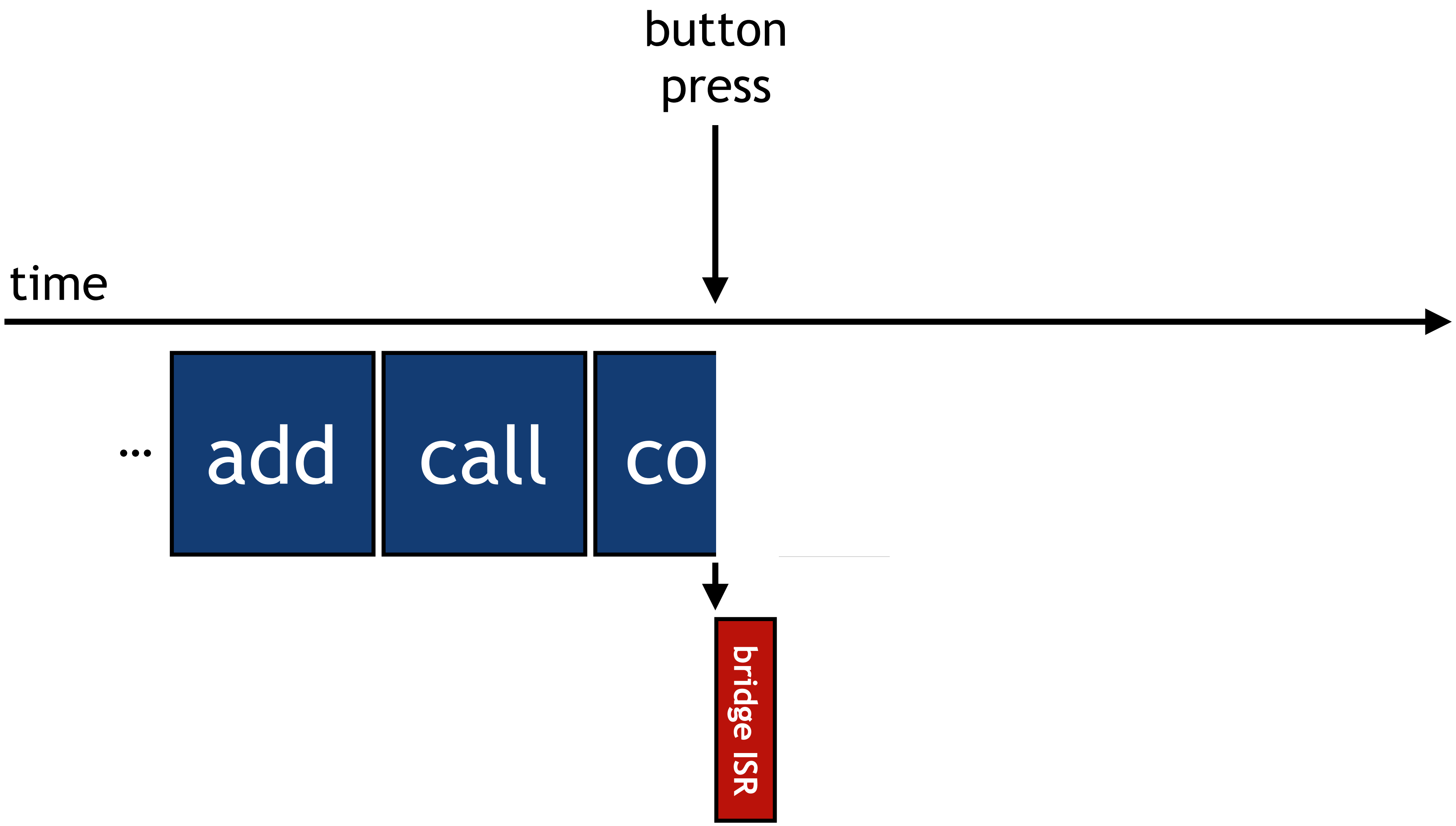
Message bridging



- **Turn hardware events into software messages**
  - Extend actor domain to hardware
  - IRQs on microkernels
- **Subscription model**
  - Threads specify hardware of interest

Message bridging

Message bridging



Bridge interrupt handler

```
#define ALL_PINS 0xff

void GPIOInterruptHandler (unsigned long port) {
  uint8_t values ;
  /* clear all the interrupts for this port */
  GPIOPinIntClear(port, ALL_PINS);

  /* read the value of the port */
  values =  GPIOPinRead(port, ALL_PINS);

  /* send it to the subscribers */
  bridge_produce(GPIO_BRIDGE, &values , sizeof(uint8_t));
}
```

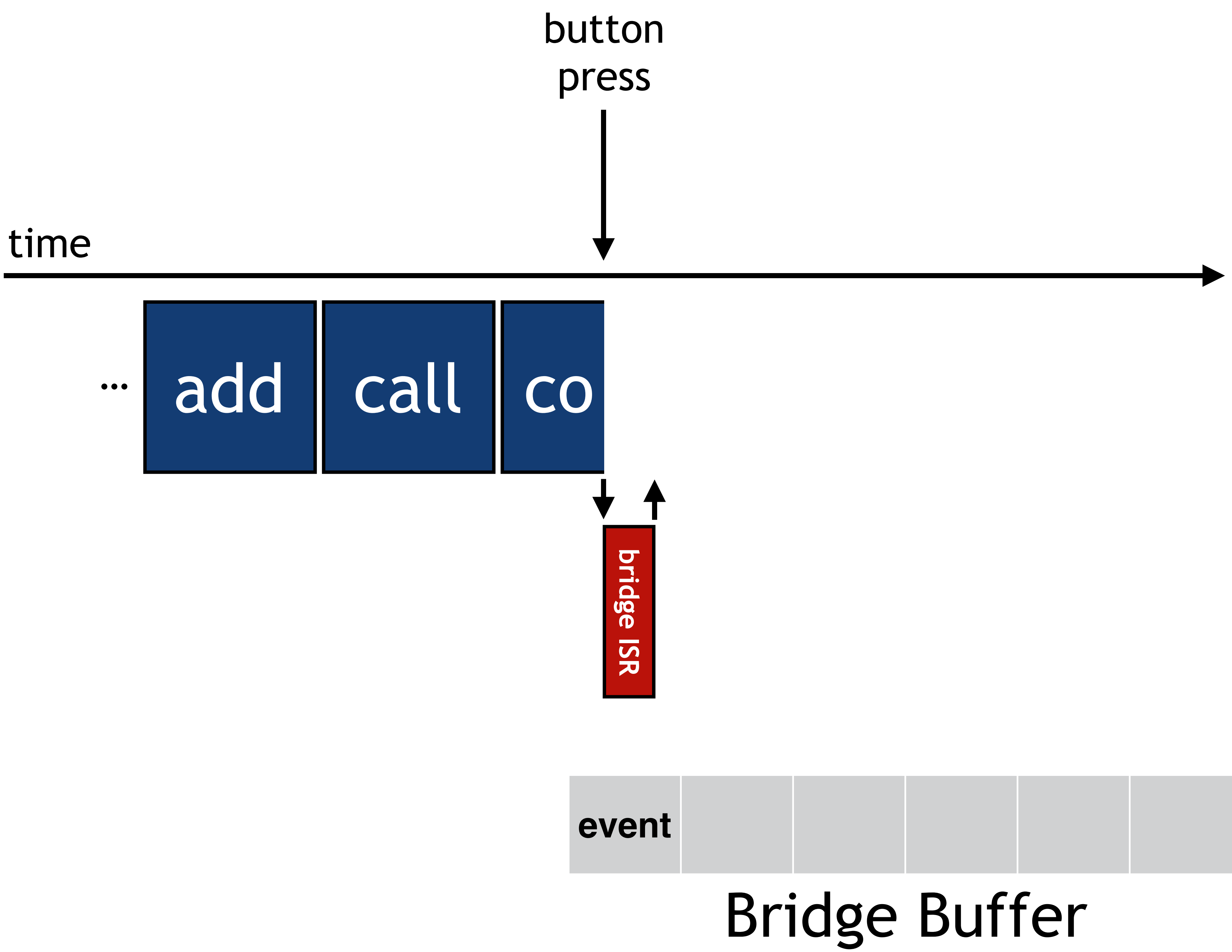

Bridge Buffer

Bridge interrupt handler

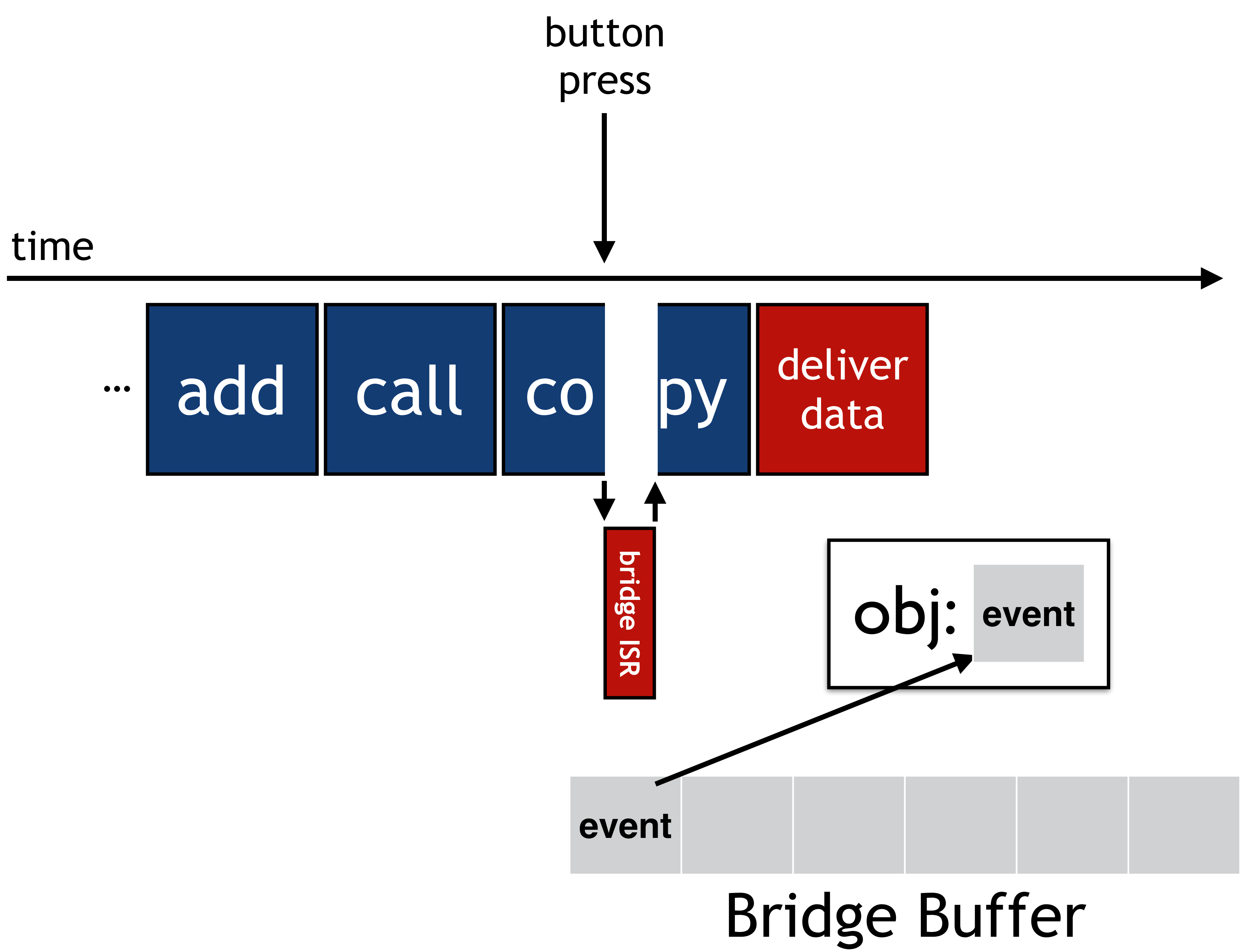
```
#define ALL_PINS 0xff

void GPIOInterruptHandler (unsigned long port) {
  uint8_t values ;
  /* clear all the interrupts for this port */
  GPIOPinIntClear(port, ALL_PINS);

  /* read the value of the port */
  values =  GPIOPinRead(port, ALL_PINS);

  /* send it to the subscribers */
  bridge_produce(GPIO_BRIDGE, &values , sizeof(uint8_t));
}
```

Bridge Buffer

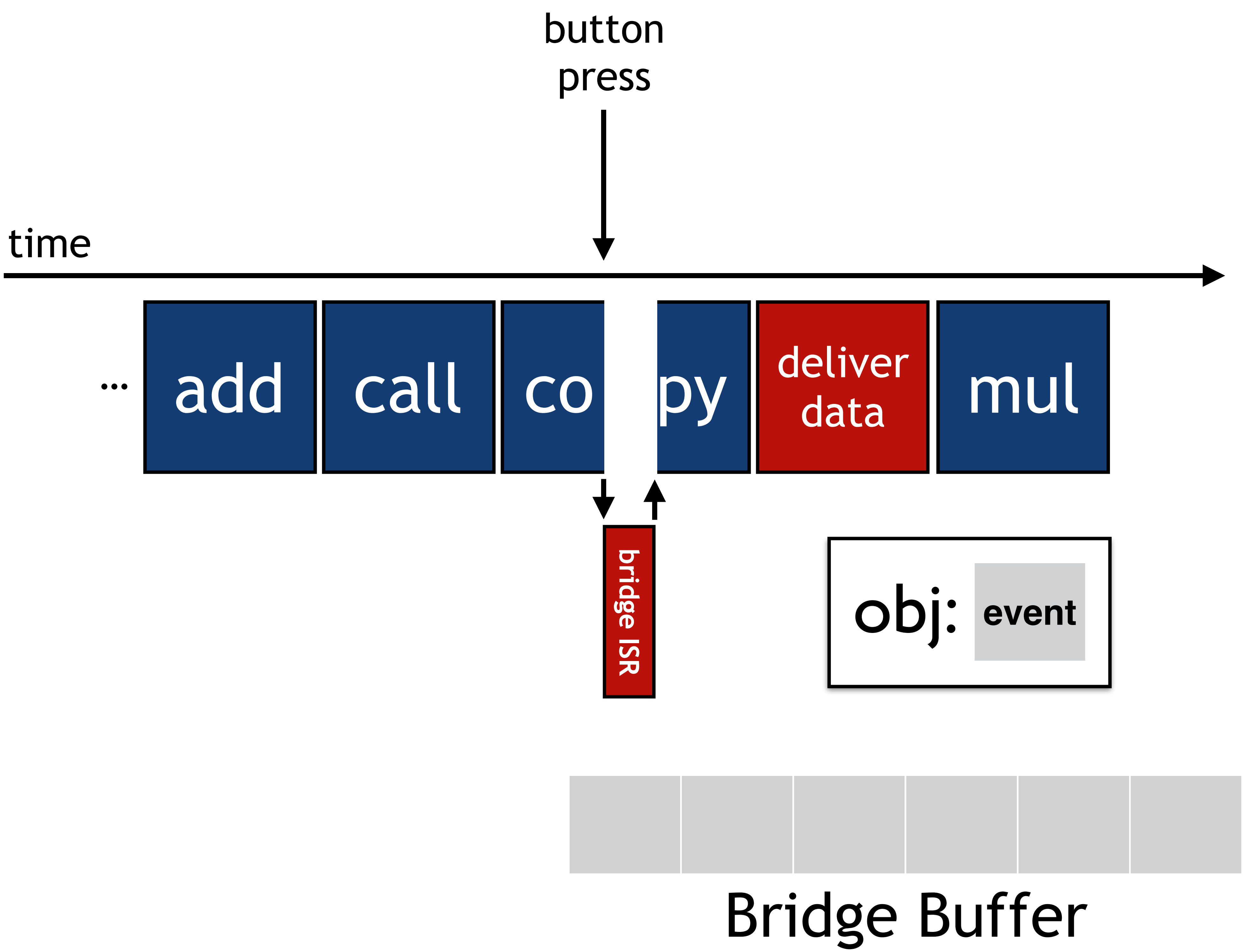
Message bridging



Message bridging



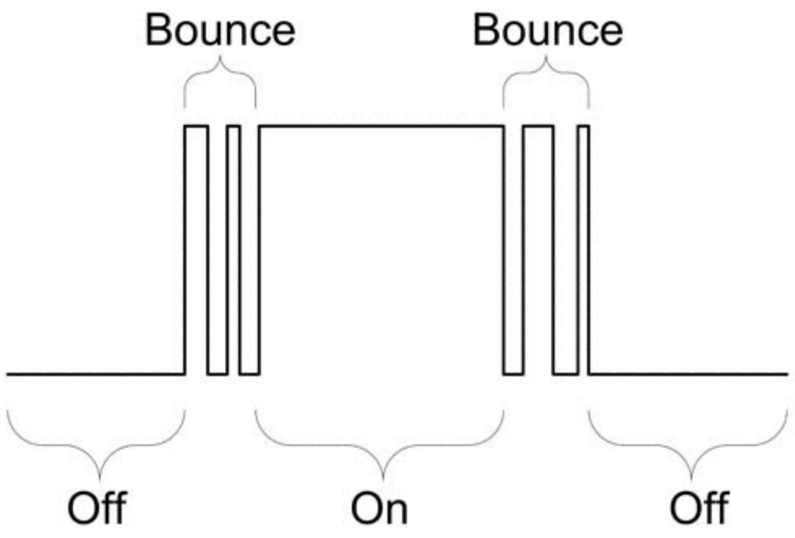
Message bridging



Message bridging

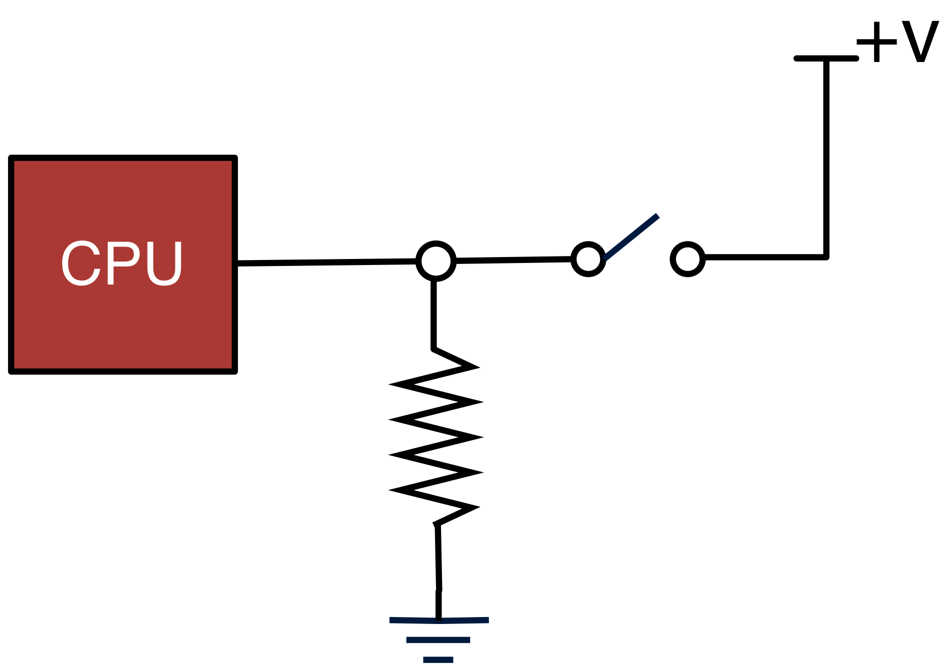
- **Two-phase bridging**
  - Copy data **into** bridge immediately
    - ~4 microseconds
    - No allocation
  - Copy data **from** bridge "later"
    - ~10s of milliseconds
    - Allocate long-term storage
      - When VM is *not* running

Message bridging



- **Debouncing:**
  - Polling: 17 lines of Python
  - Interrupts: 141 lines of C
  - Bridges: 33 lines of Medusa
    - **Zero impact on other threads**

Conclusions



- *All* **embedded systems are concurrent**
  - Current solutions are inadequate
- **Actor model well-suited to I/O**
  - Polling-like simplicity
  - Interrupt-like performance