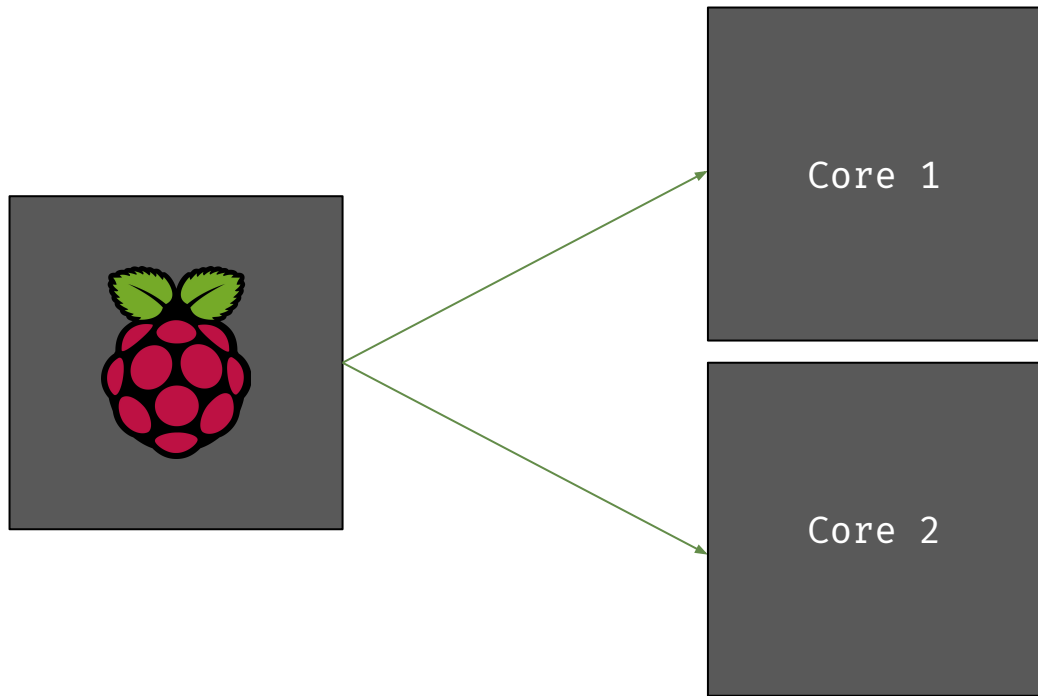
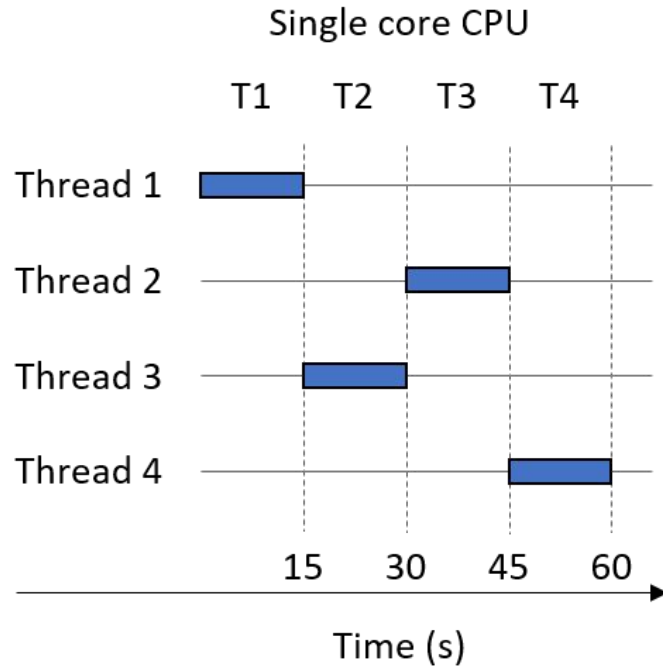




Cortex M0+



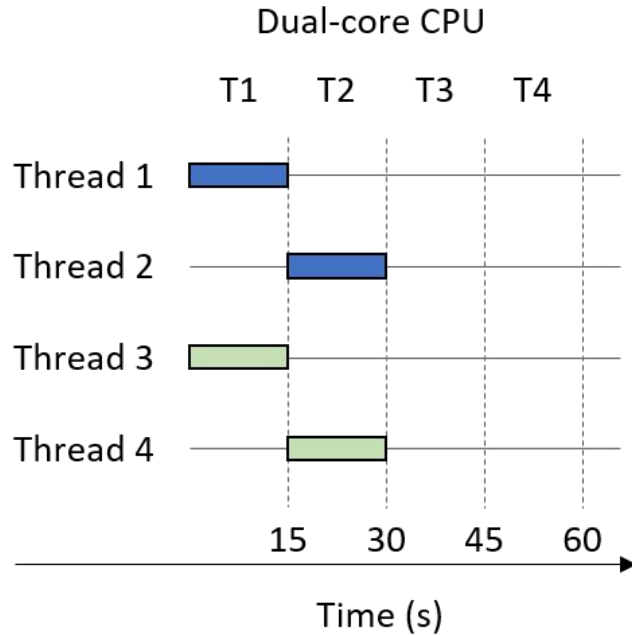
- Both have equal compute power
- Both run simultaneously
- So far, we've only used 1



“Synchronous”

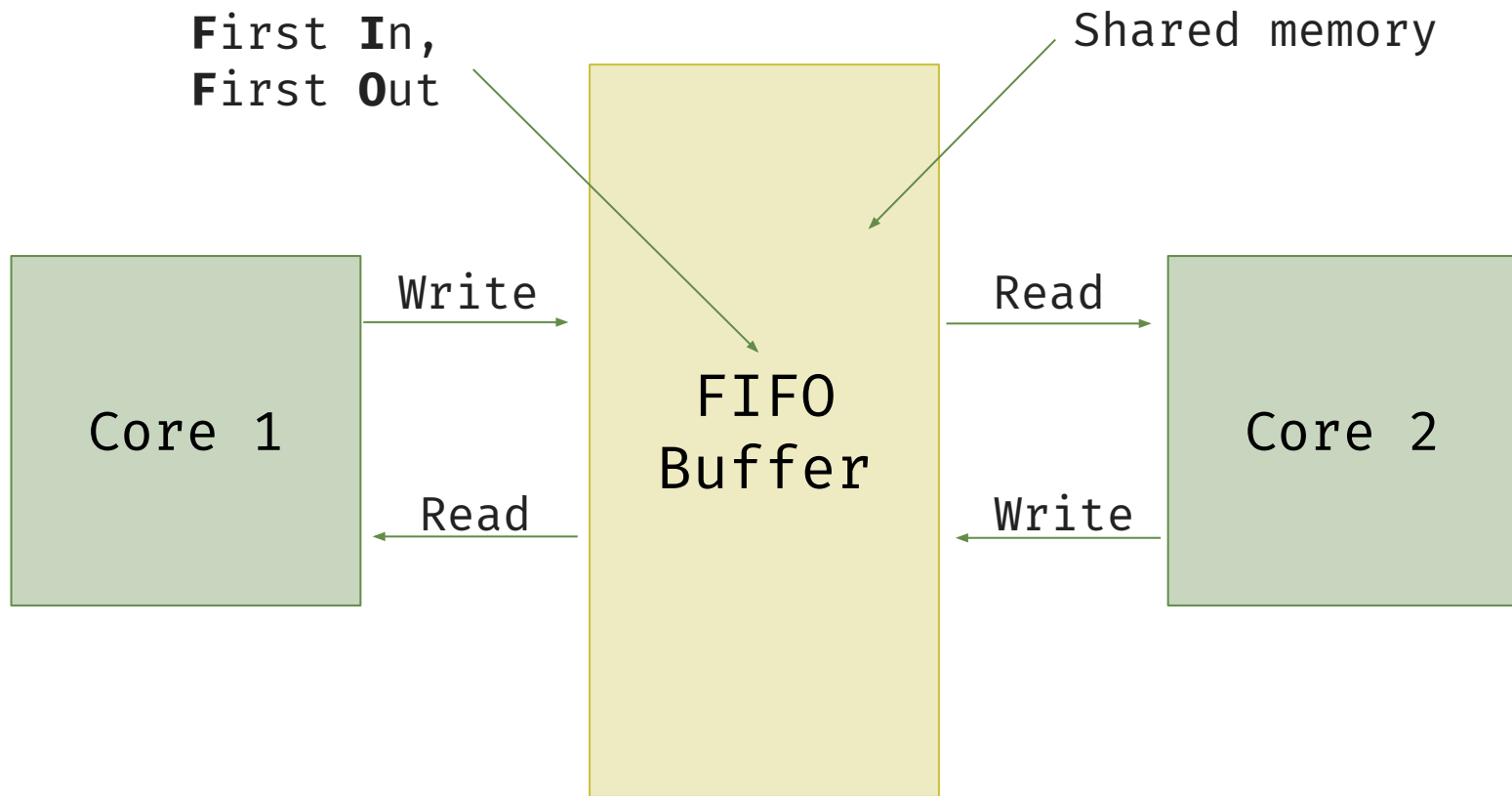
Each process has to finish before another can begin

Compute time distributed on a single-core processor



Each core is **synchronous** in itself, but performance is effectively *asynchronous*.

Compute time distributed on a two-core processor





FIFO Buffer

“Buffer”

Location that temporarily stores data in transit from one place to another

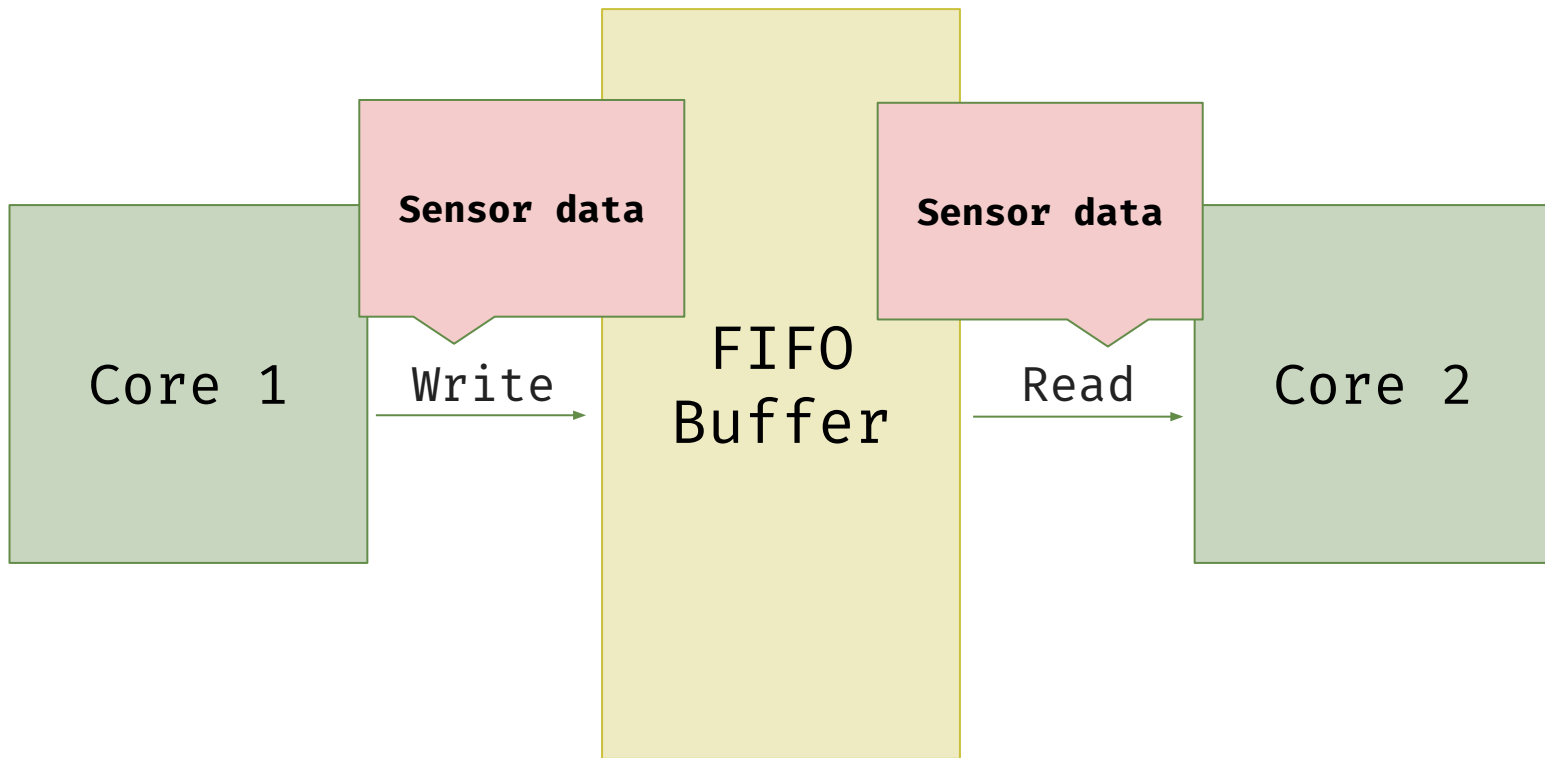
- This buffer can only hold **integers**



Becomes important later

All in the timing

Cores	Approach	Speed Up	Efficiency
1	Synchronous		



Time on 1 core



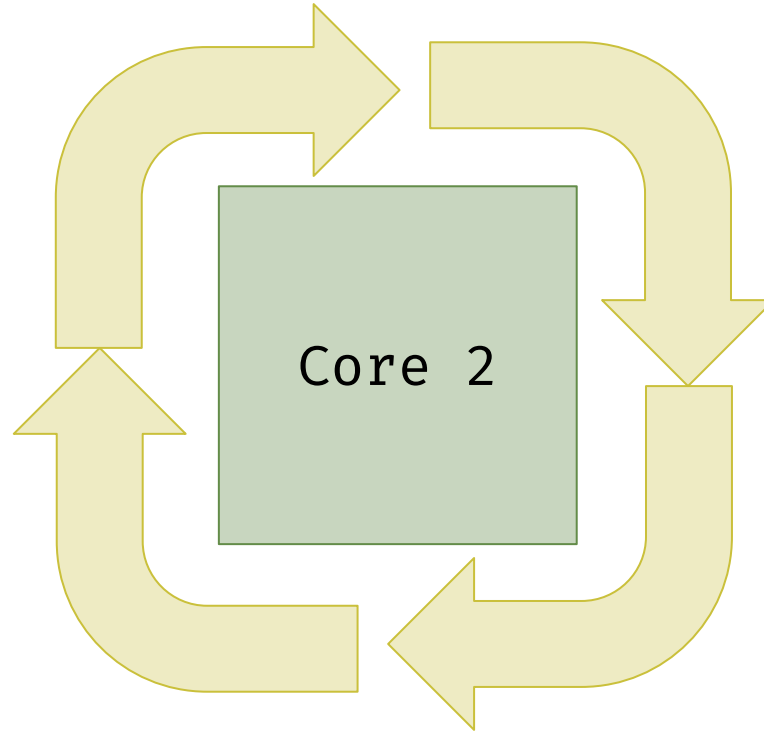
$$Speedup_c = \frac{T_1}{T_c}$$

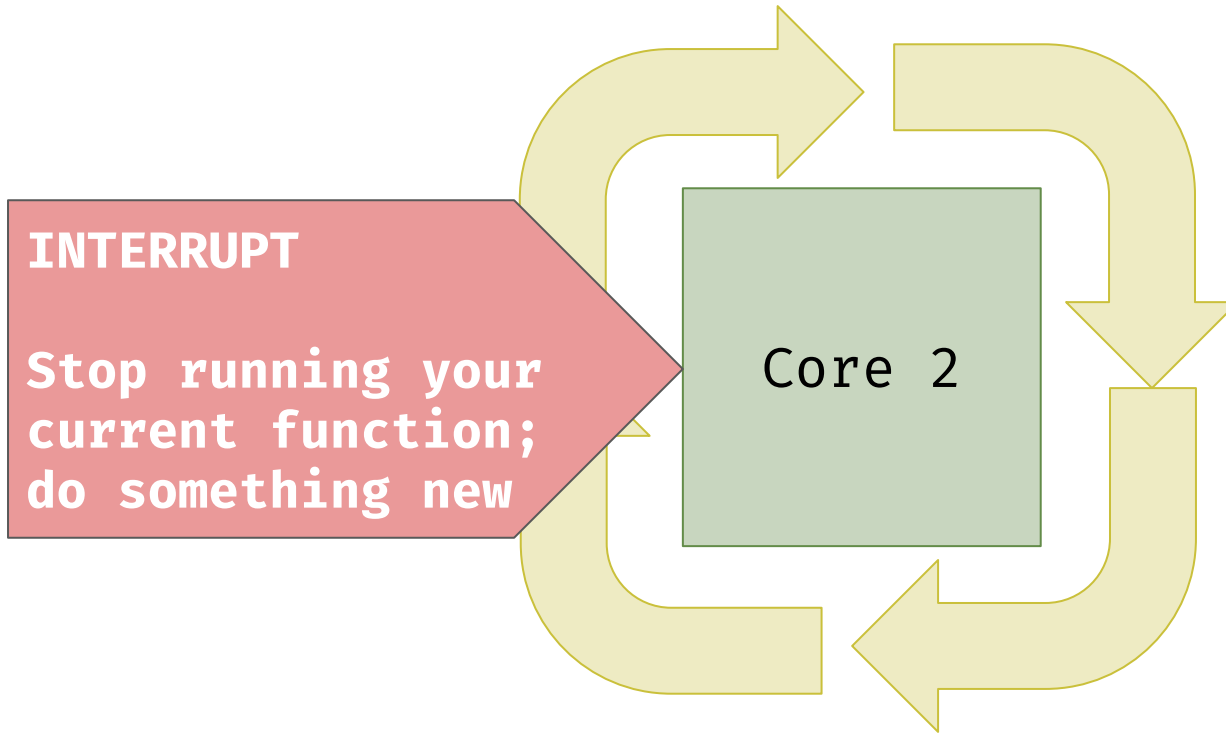


Time on n cores

$$Efficiency_c = \frac{T_1}{T_c \times c} = \frac{Speedup_c}{c}$$

The core is
otherwise
occupied until
we...





INTERRUPTS

irq

→ “**interrupt request**”

0

`multicore_fifo_clear_irq()`

1

`irq_set_exclusive_handler(...)`

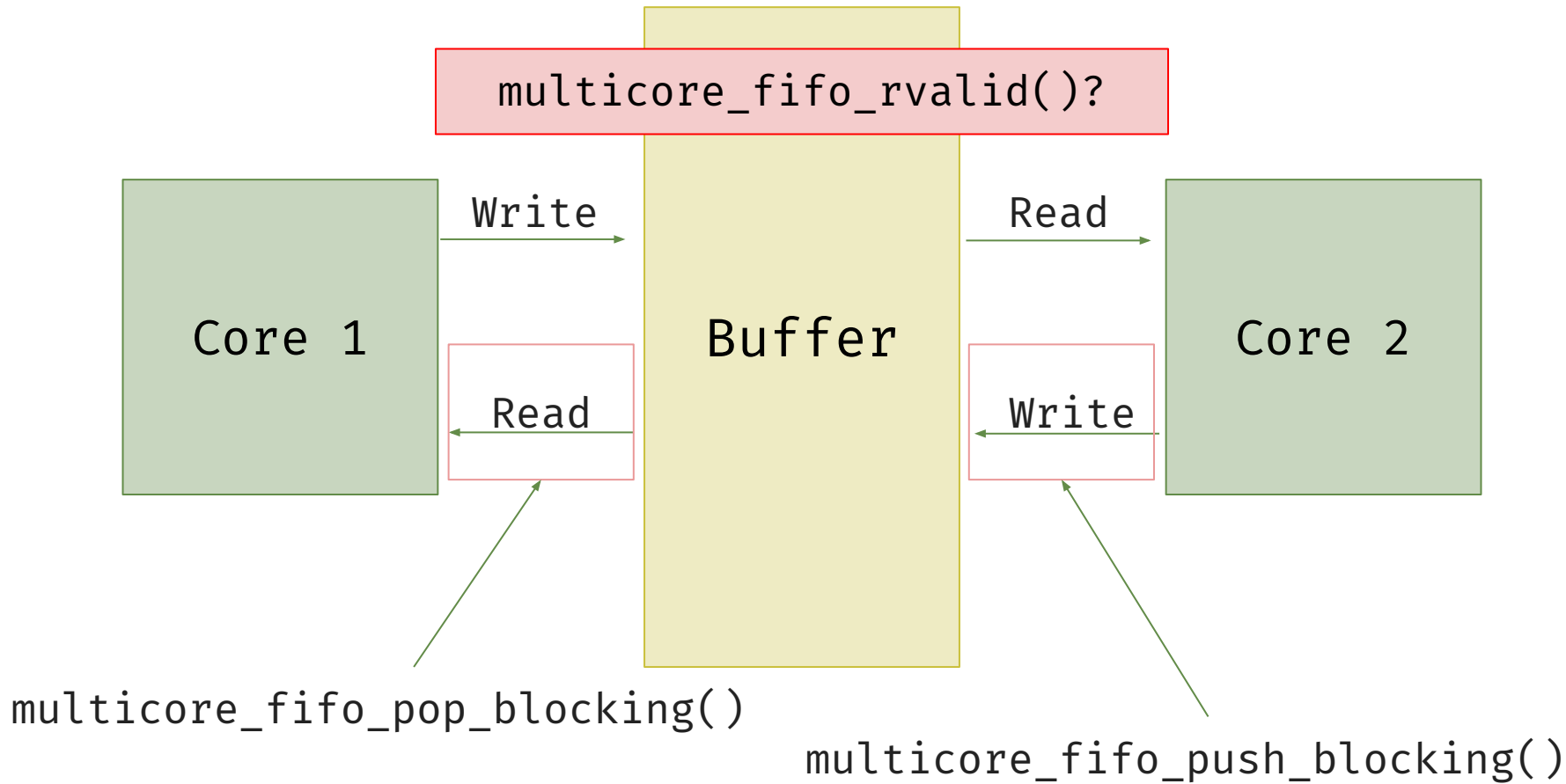
2

`irq_set_enabled(...)`

... when done ...

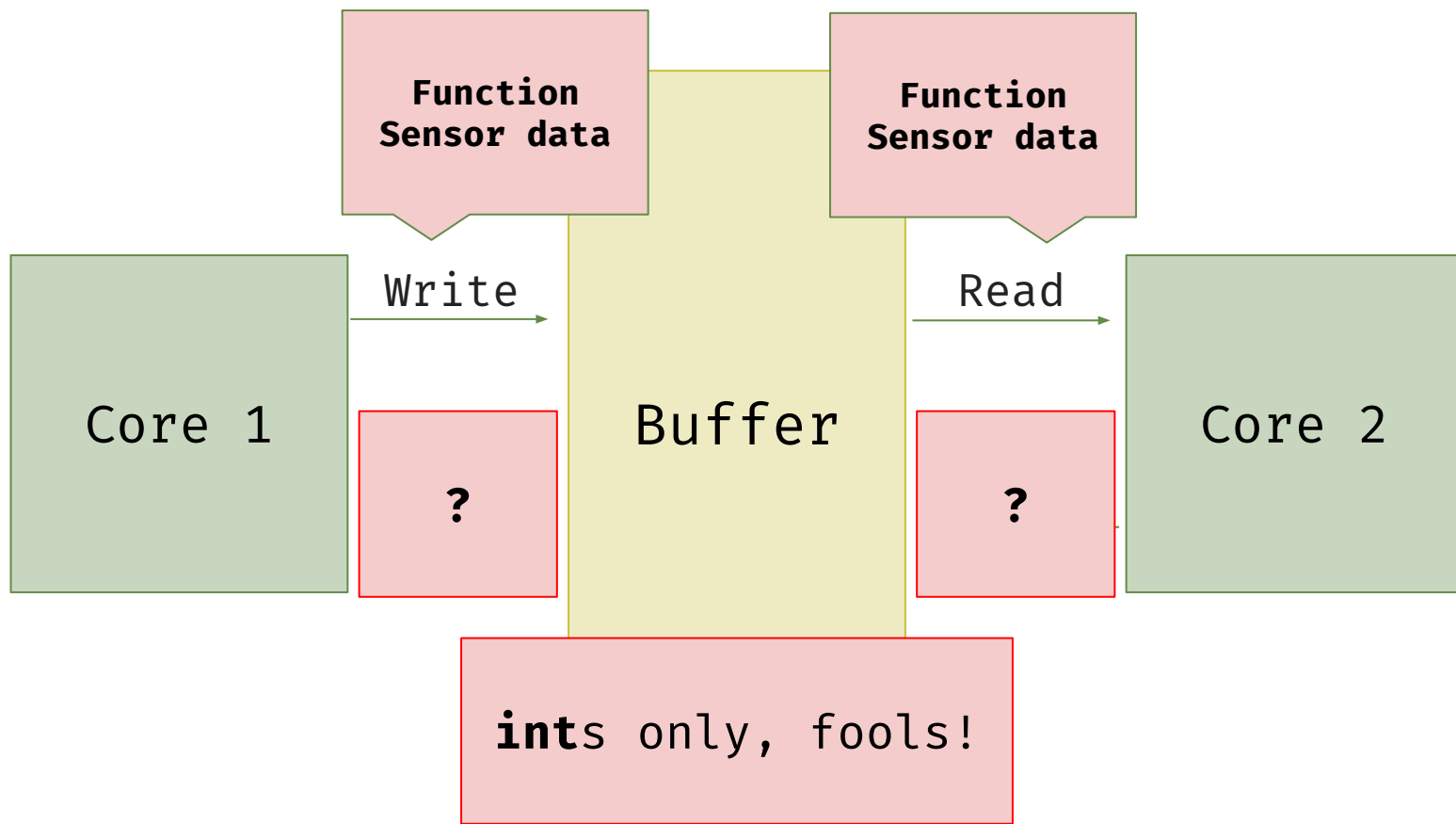
3

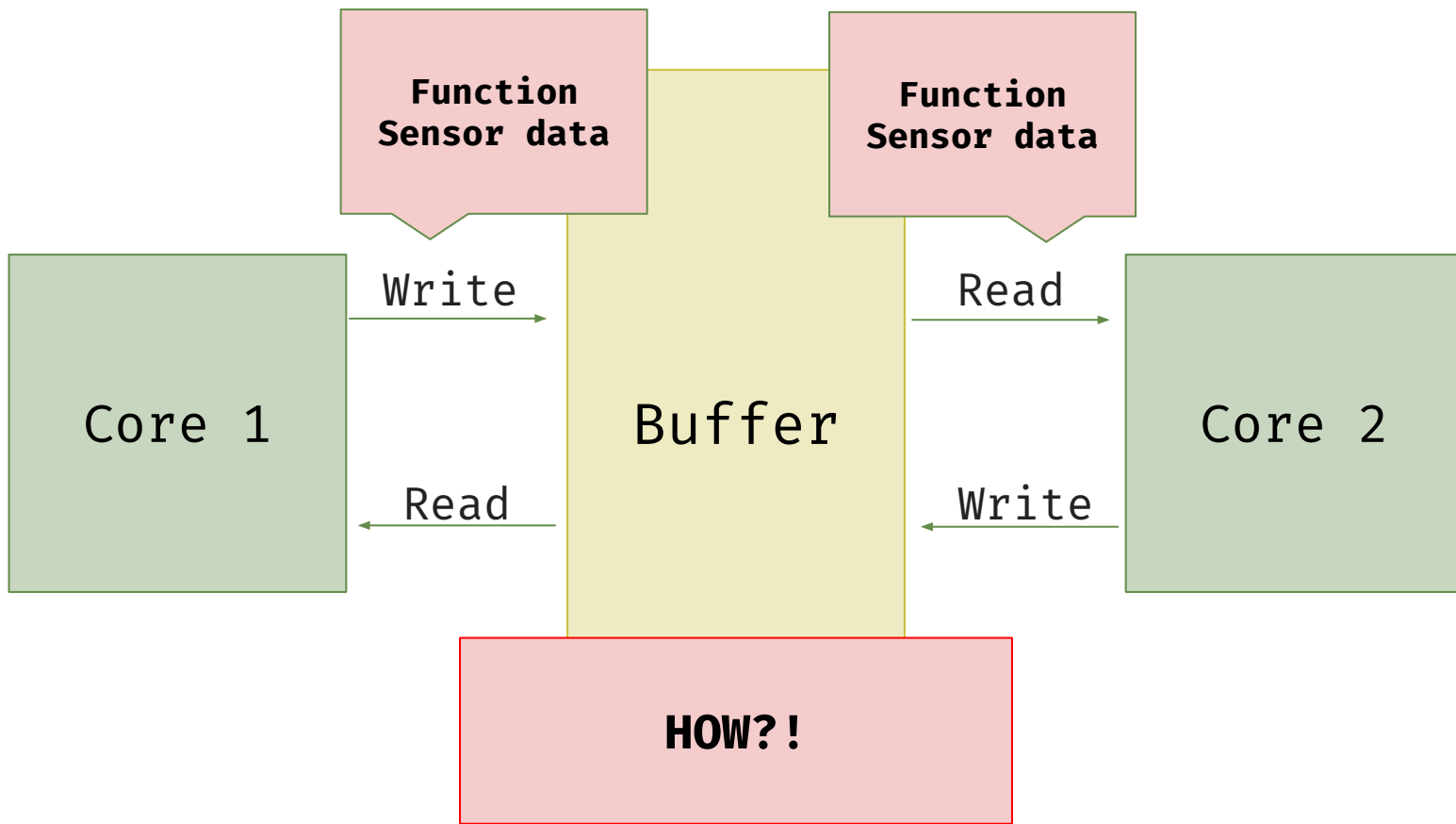
`multicore_fifo_clear_irq()`



All in the timing

Cores	Approach	Speed Up	Efficiency
1	Synchronous		
2	FIFO by value		





All in the timing

Cores	Approach	Speed Up	Efficiency
1	Synchronous		
2	FIFO by value		
2	FIFO by function		