

Introduction

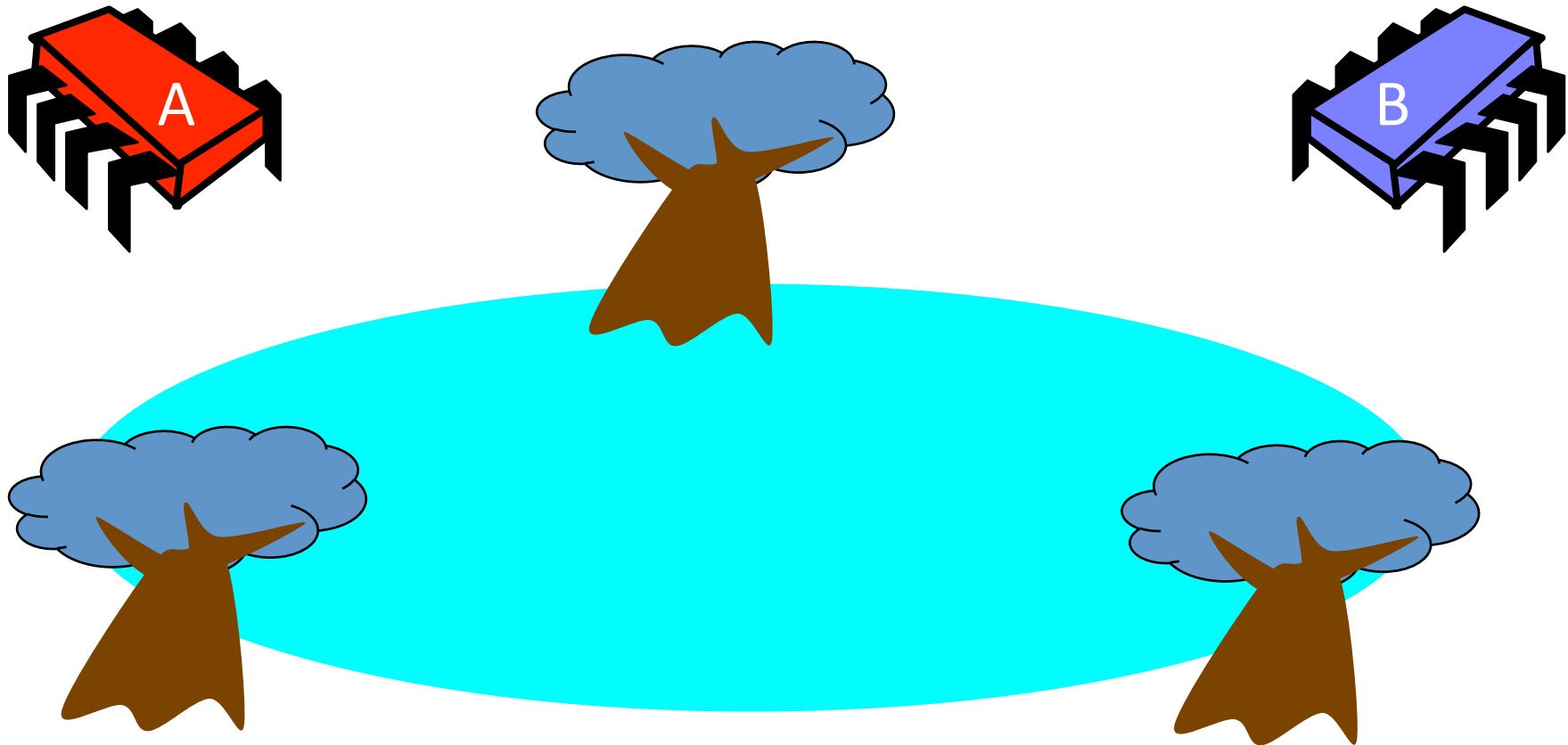
Companion slides for
The Art of Multiprocessor Programming
by Maurice Herlihy & Nir Shavit



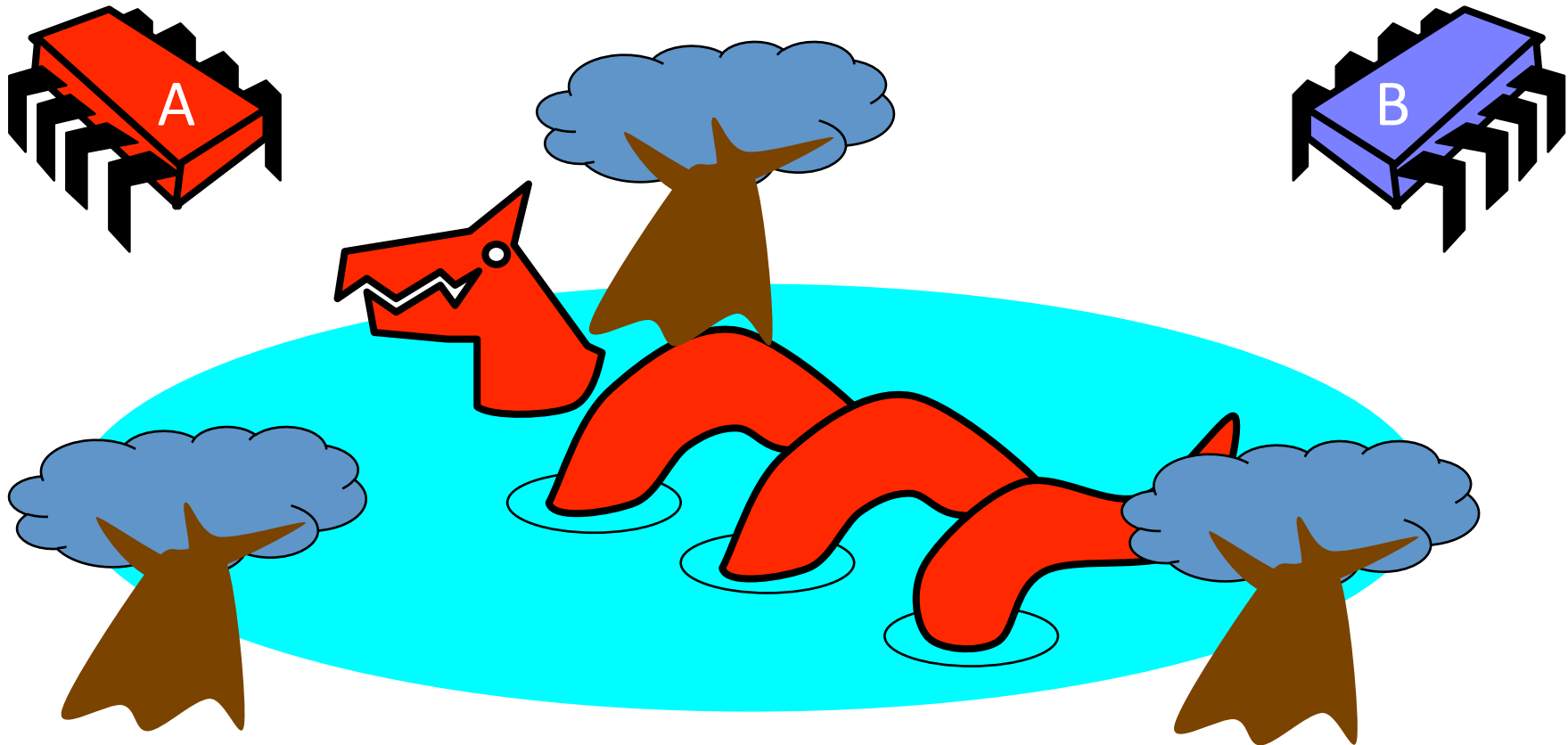
This work is licensed under a
[Creative Commons Attribution-ShareAlike 2.5 License](https://creativecommons.org/licenses/by-sa/2.5/).

- You are free:
 - to Share — to copy, distribute and transmit the work
 - to Remix — to adapt the work
- Under the following conditions:
 - Attribution. You must attribute the work to “The Art of Multiprocessor Programming” (but not in any way that suggests that the authors endorse you or your use of the work).
 - Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.
- For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to
 - <http://creativecommons.org/licenses/by-sa/3.0/>.
- Any of the above conditions can be waived if you get permission from the copyright holder.
- Nothing in this license impairs or restricts the author's moral rights.

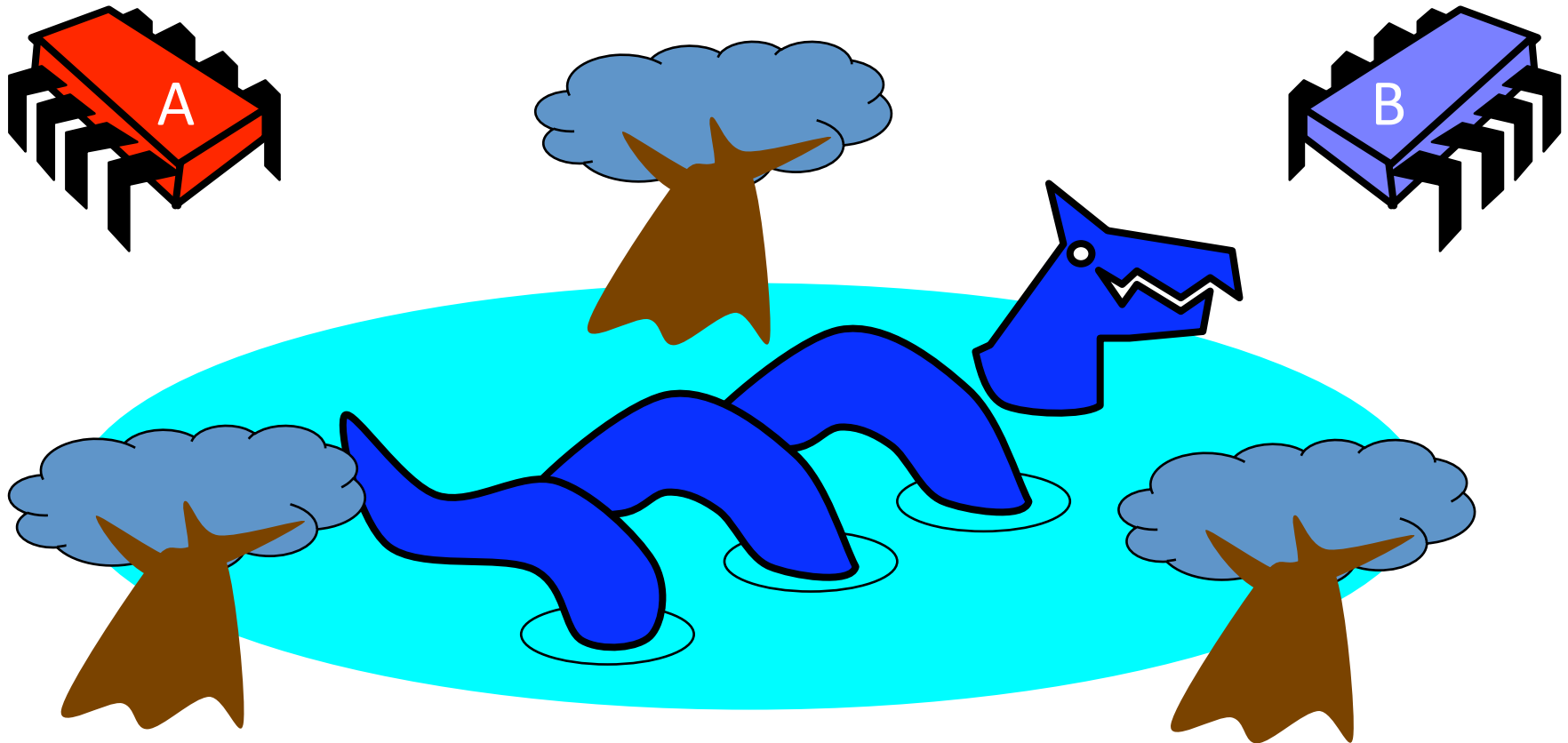
Mutual Exclusion or “Alice & Bob share a pond”



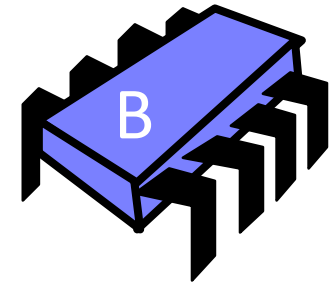
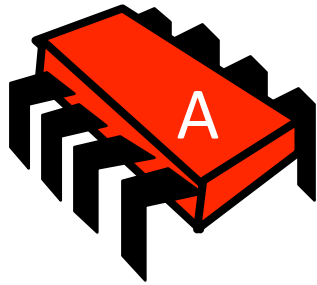
Alice has a pet



Bob has a pet



The Problem



Formalizing the Problem

- Two types of formal properties in asynchronous computation:
- Safety Properties
 - Nothing bad happens ever
- Liveness Properties
 - Something good happens eventually

Formalizing our Problem

- Mutual Exclusion
 - Both pets never in pond simultaneously
 - This is a *safety* property
- No Deadlock
 - if only one wants in, it gets in
 - if both want in, one gets in.
 - This is a *liveness* property

Simple Protocol

- Idea
 - Just look at the pond
- Gotcha
 - Trees obscure the view

Interpretation

- Threads can't "see" what other threads are doing
- Explicit communication required for coordination

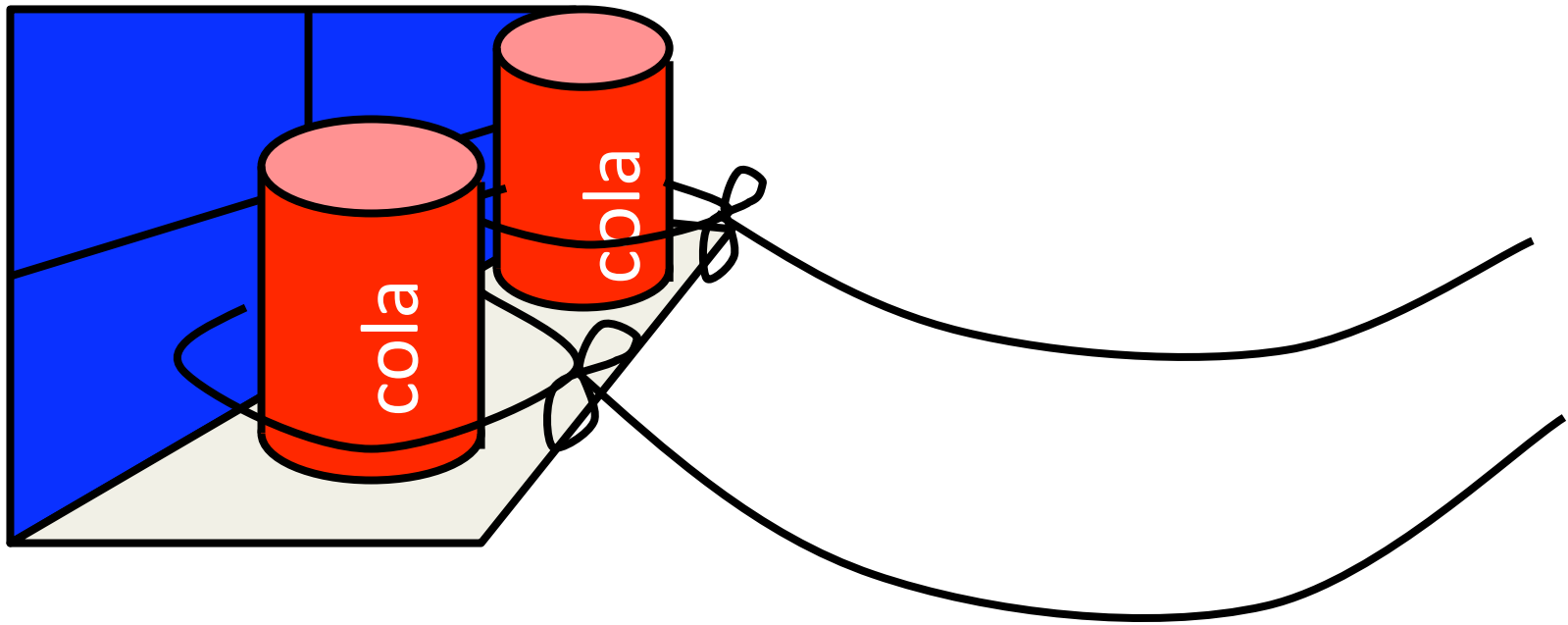
Cell Phone Protocol

- Idea
 - Bob calls Alice (or vice-versa)
- Gotcha
 - Bob takes shower
 - Alice recharges battery
 - Bob out shopping for pet food ...

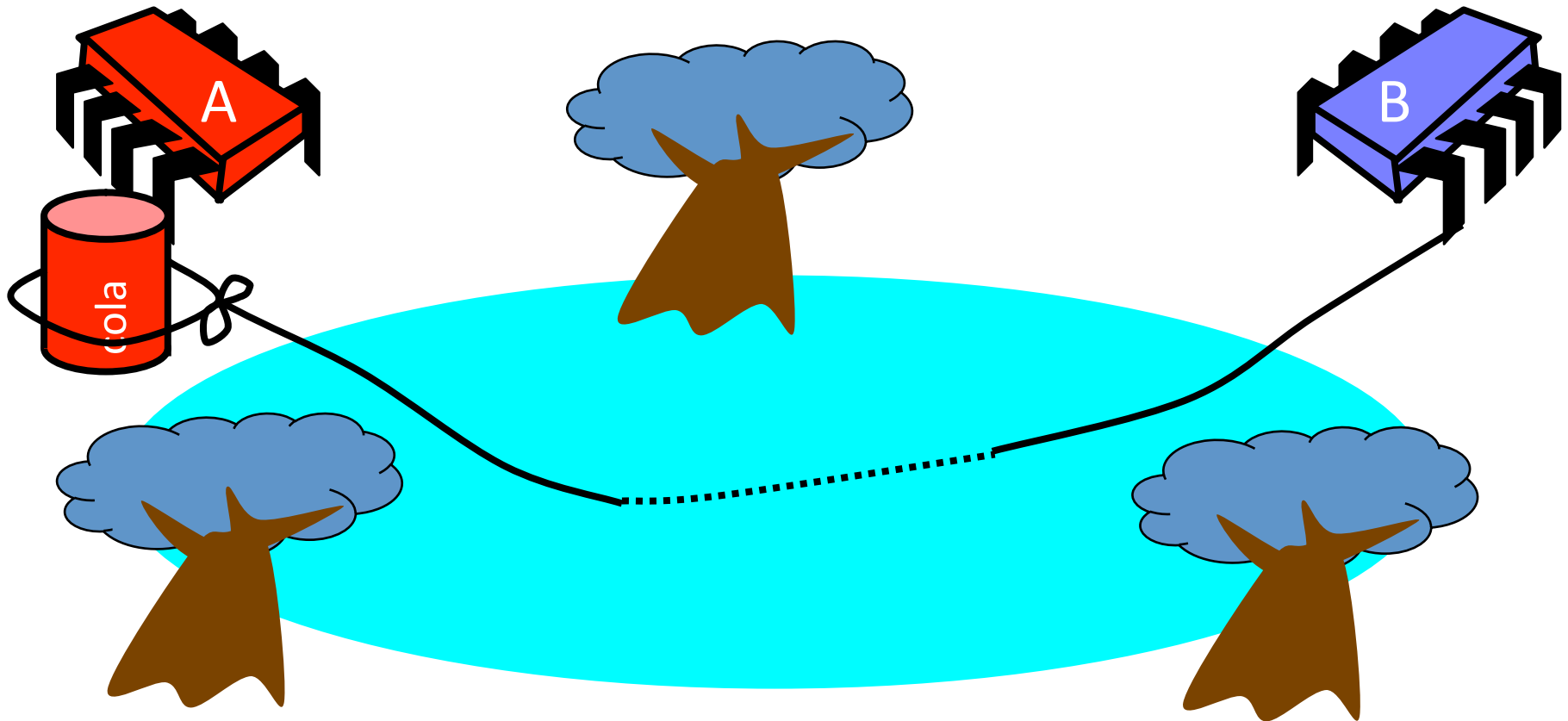
Interpretation

- Message-passing doesn't work
- Recipient might not be
 - Listening
 - There at all
- Communication must be
 - Persistent (like writing)
 - Not transient (like speaking)

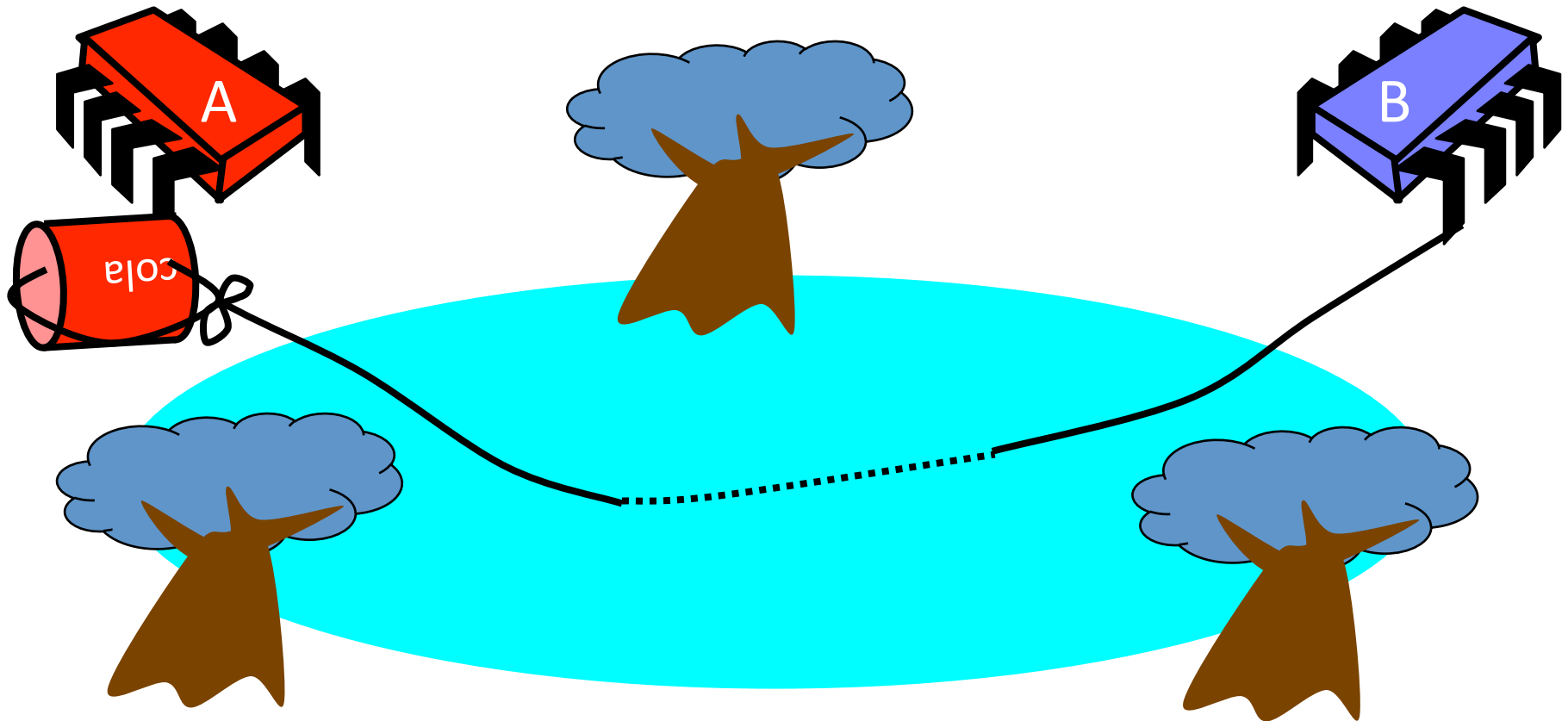
Can Protocol



Bob conveys a bit



Bob conveys a bit



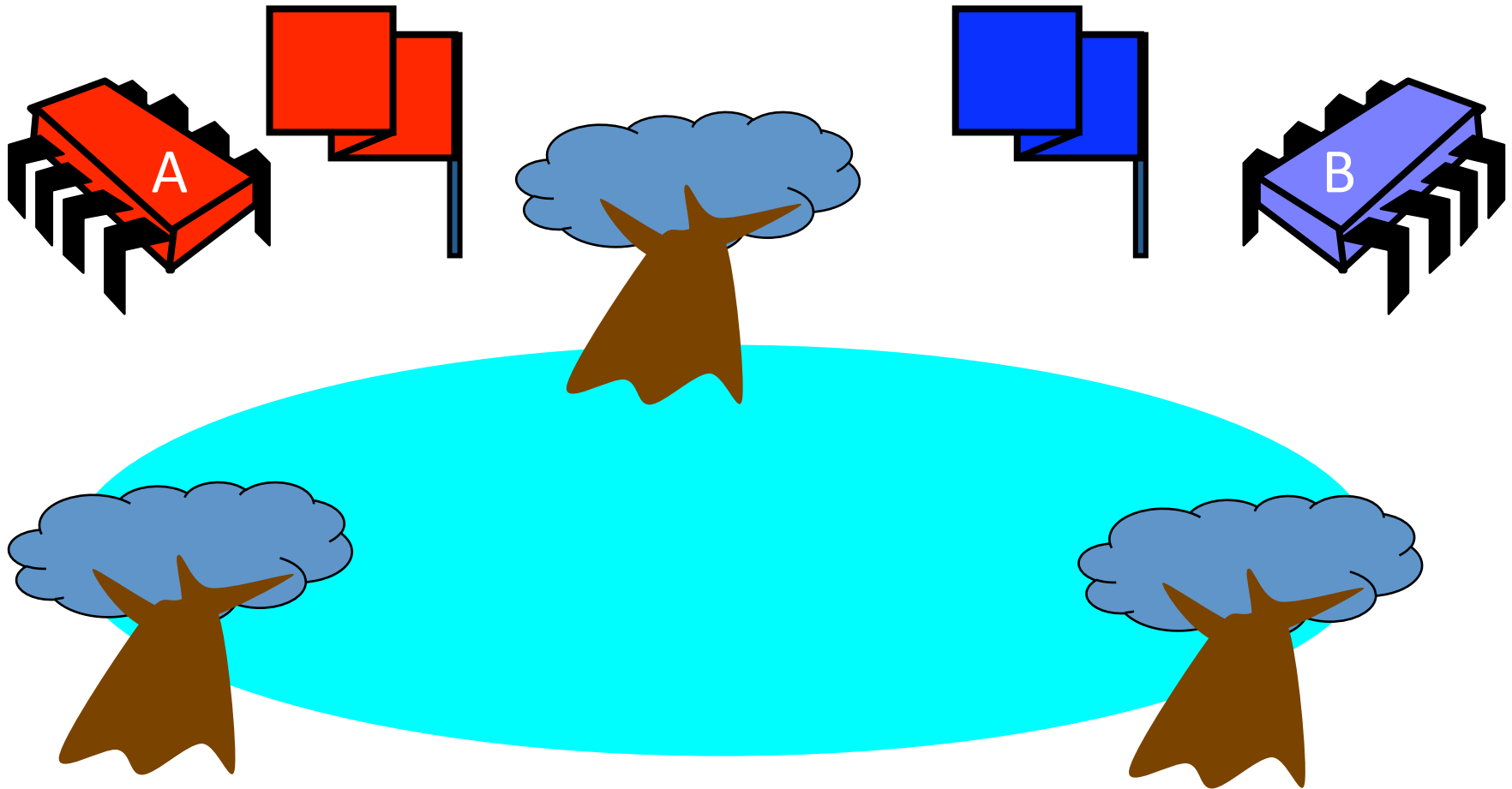
Can Protocol

- Idea
 - Cans on Alice's windowsill
 - Strings lead to Bob's house
 - Bob pulls strings, knocks over cans
- Gotcha
 - Cans cannot be reused
 - Bob runs out of cans

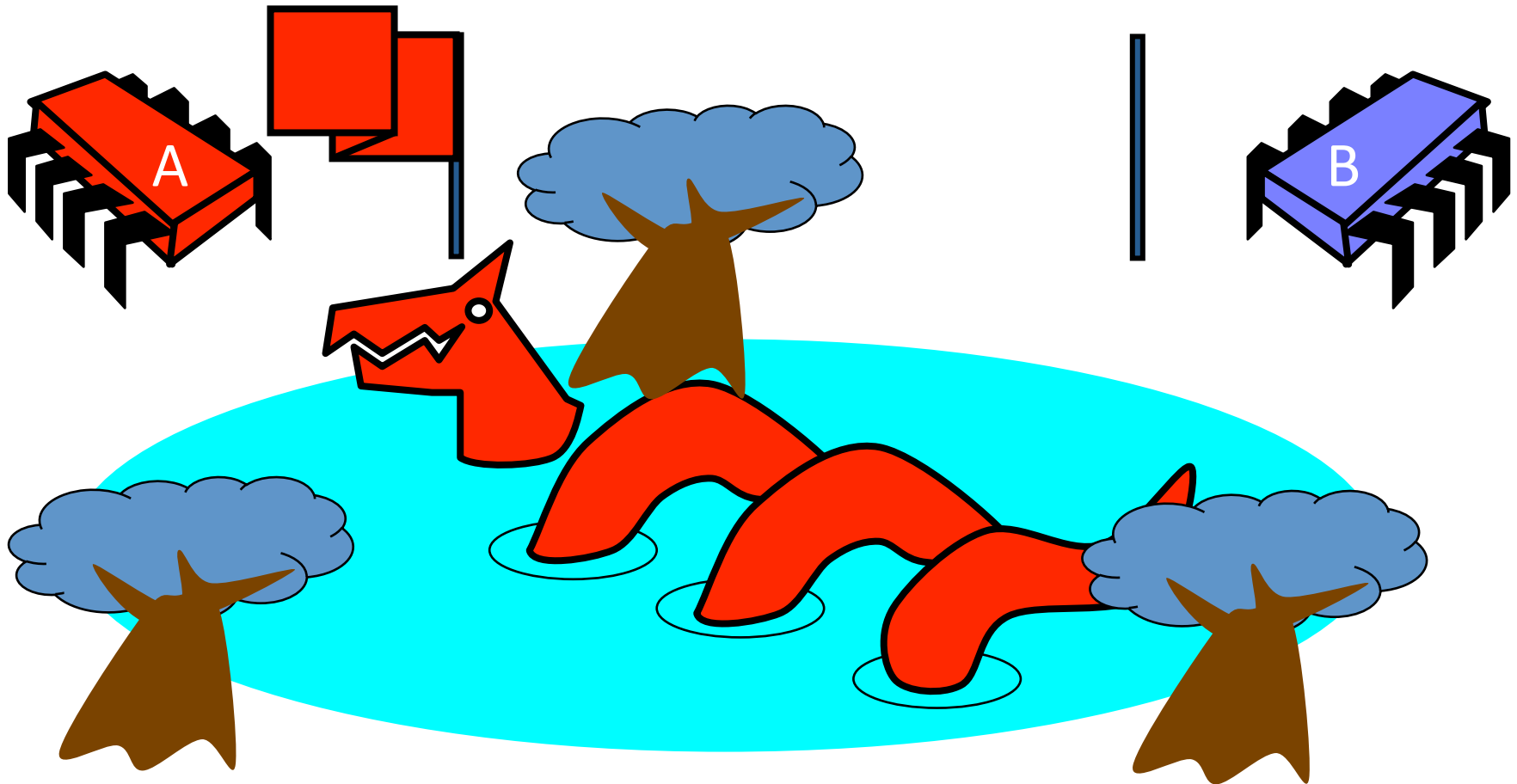
Interpretation

- Cannot solve mutual exclusion with interrupts
 - Sender sets fixed bit in receiver's space
 - Receiver resets bit when ready
 - Requires unbounded number of interrupt bits

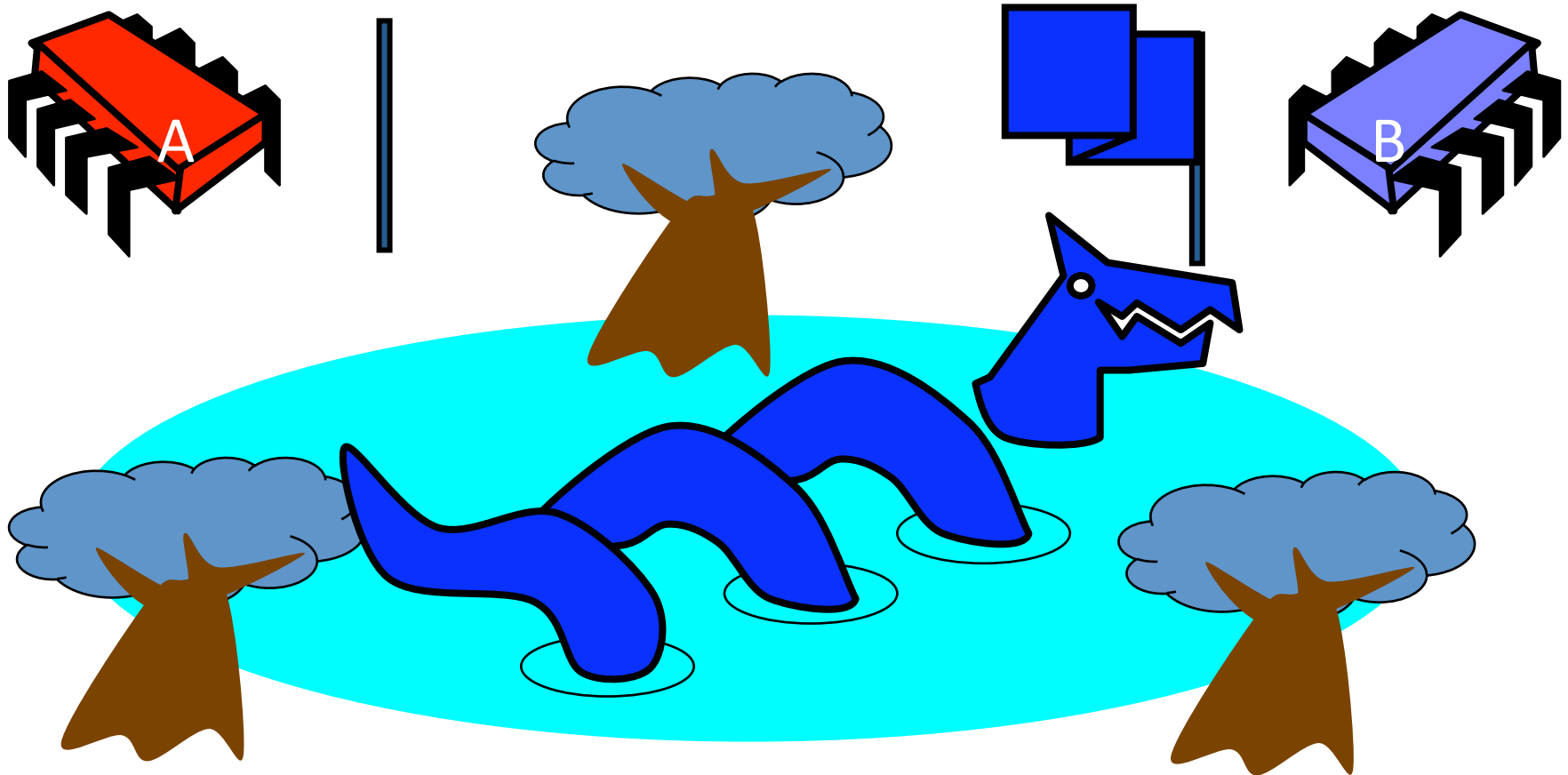
Flag Protocol



Alice's Protocol (sort of)



Bob's Protocol (sort of)

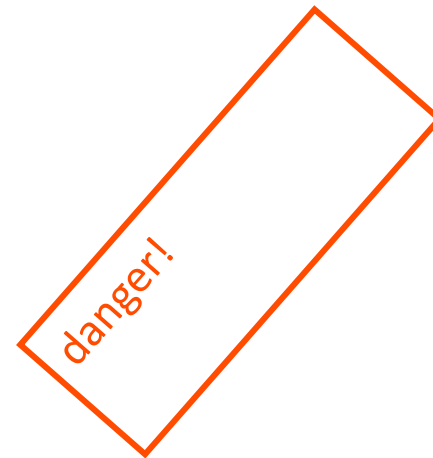


Alice's Protocol

- Raise flag
- Wait until Bob's flag is down
- Unleash pet
- Lower flag when pet returns

Bob's Protocol

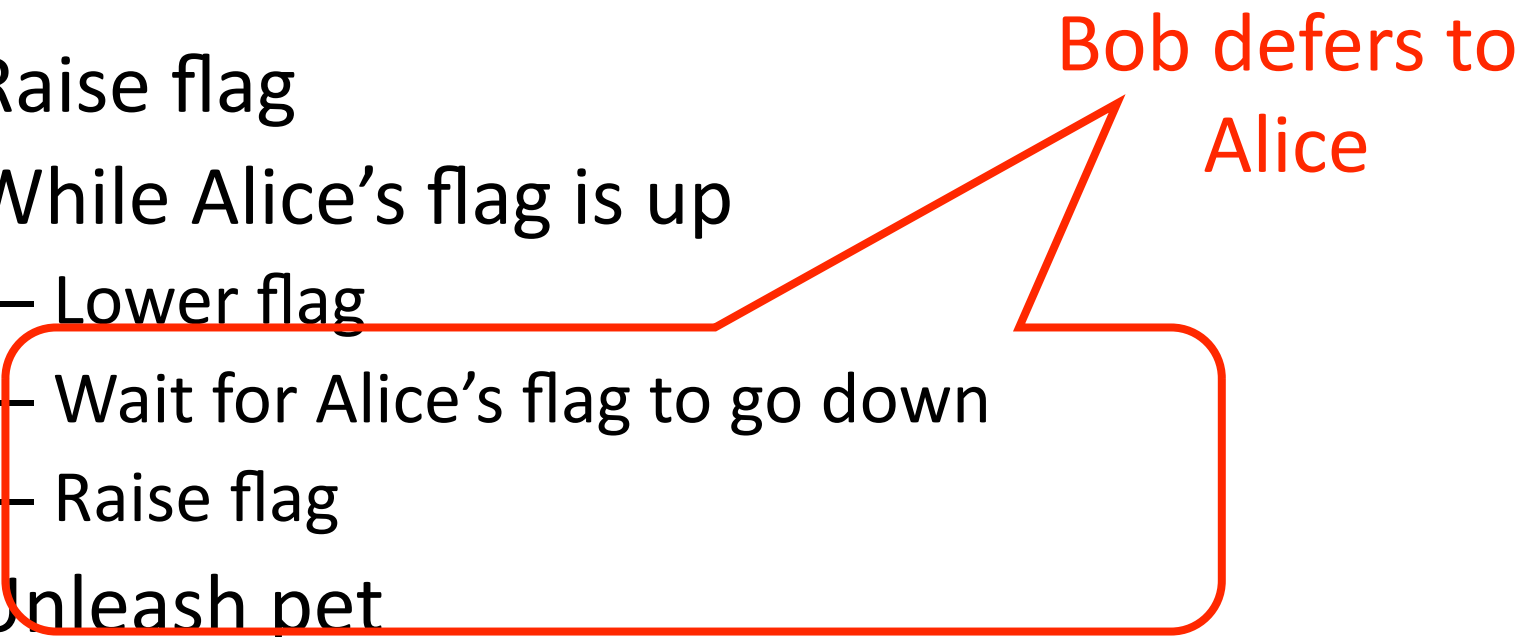
- Raise flag
- Wait until Alice's flag is down
- Unleash pet
- Lower flag when pet returns



Bob's Protocol (2nd try)

- Raise flag
- While Alice's flag is up
 - Lower flag
 - Wait for Alice's flag to go down
 - Raise flag
- Unleash pet
- Lower flag when pet returns

Bob's Protocol

- Raise flag
 - While Alice's flag is up
 - Lower flag
 - Wait for Alice's flag to go down
 - Raise flag
 - Unleash pet
 - Lower flag when pet returns
- Bob defers to Alice
- 

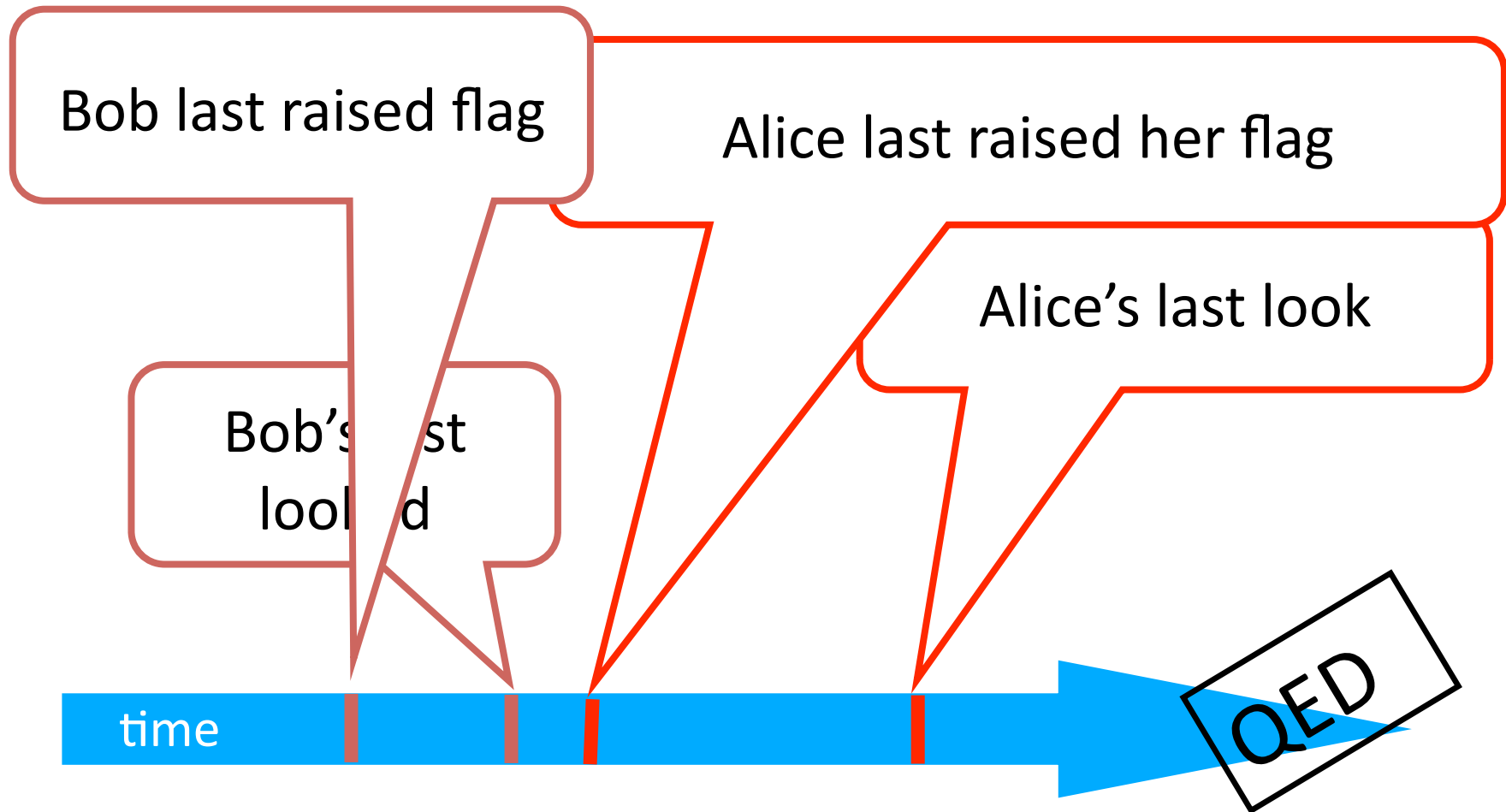
The Flag Principle

- Raise the flag
- Look at other's flag
- Flag Principle:
 - If each raises and looks, then
 - Last to look must see both flags up

Proof of Mutual Exclusion

- Assume both pets in pond
 - Derive a contradiction
 - By reasoning backwards
- Consider the last time Alice and Bob each looked before letting the pets in
- Without loss of generality assume Alice was the last to look...

Proof



Alice must have seen Bob's Flag. A Contradiction

Proof of No Deadlock

- If only one pet wants in, it gets in.

Proof of No Deadlock

- If only one pet wants in, it gets in.
- Deadlock requires both continually trying to get in.

Proof of No Deadlock

- If only one pet wants in, it gets in.
- Deadlock requires both continually trying to get in.
- If Bob sees Alice's flag, he gives her priority (a gentleman...)

QED

Remarks

- Protocol is *unfair*
 - Bob's pet might never get in
- Protocol uses *waiting*
 - If Bob is eaten by his pet, Alice's pet might never get in

Moral of Story

- Mutual Exclusion **cannot be solved** by
 - transient communication (cell phones)
 - interrupts (cans)
- It **can be solved** by
 - one-bit shared variables
 - that can be read or written