L8: Intertask/Interthread Communication

- Information – data
- Source and Destinations for the exchanges
- Control and Synchronization

Shared Variables
- Global Variables
- Shared Buffer
  - Producer vs. Consumer
  - Double Buffer
  - Circular Buffer
- Mailbox
- Pipes, Named Pipes

The producer task “produces” information that is “consumed” by a consumer task. A buffer is used to hold data between the two tasks.

The producer & consumer must be synchronized; that is, a producer must wait if it attempts to put data into a full buffer whereas a consumer must wait if it attempts to extract data from an empty buffer.
T1: Producer

```
Initialize

do forever
  if B0 isEmpty()
    repeat
      produce info in B0
    until B1 isEmpty()
    mark B0 as Full
  endif
  
  if B1 isEmpty()
    repeat
      produce info in B1
    until B0 isEmpty()
    mark B1 as Full
  endif
end loop
```
L8: Circular (Ring) Buffer

Conceptual view of a circular queue

(a) Head Pointer
(b) Tail Pointer
(c) Head Pointer
L8: Circular (Ring) Buffer

T1: Producer

Initialize
do forever
if cirbuf(prd_indx).isFull
   ERROR
endif
produce info in cirbuf(prd_indx)
mark cirbuf(prd_indx) as Full
prd_indx = next()
end loop

T2: Consumer

Initialize
do forever
   while cirbuf(cns_indx).isEmpty
do nothing
   end while
   consume info from cirbuf(cns_indx)
   mark cirbuf(cns_indx) as Empty
cns_indx = next()
end loop
L8: Mailbox

“n:1” \(\rightarrow\) multiple producers, one single consumer

- Data can have any size (pre-defined or arranged on-the-fly)
- Data may have a priority (e.g. associated with the task’s priority)
L8: Mailbox

**Ti: Producer(s)**

Initialize

do forever

allocate data

produce info in data

send(TC_mailbox, data)

end loop

“send”, a.k.a. “post”

**TC: Consumer**

Initialize

do forever

data = wait(TC_mailbox)

consume info in data

destroy data

end loop

“wait”, a.k.a. “pend” (blocking function, releases CPU)
L8: Data with Return Mailbox

Data must include the return mailbox.
L8: Data with Return Mailbox

**Ti: Producer(s)**

Initialize

do forever

\[\text{data.returnMailbox} = \text{Ti\_mailbox}\]

produce info in data

send(\text{TC\_mailbox}, \text{data})

ack = wait(\text{Ti\_mailbox})

end loop

**TC: Consumer**

Initialize

do forever

\[\text{data} = \text{wait(\text{TC\_mailbox})}\]

consume info in data

send(\text{data.returnMailbox}, \text{ack})

end loop
Assignment

• Readings on
  – File descriptors
  – Pipes

You can find the links on Blackboard