2. Link and Memory Architectures and Technologies

2.1 Links, Thruput/Buffering, Multi-Access Ovrhds

2.2 Memories: On-chip / Off-chip SRAM, DRAM

2.A Appendix: Elastic Buffers for Cross-Clock Commun.

Manolis Katevenis

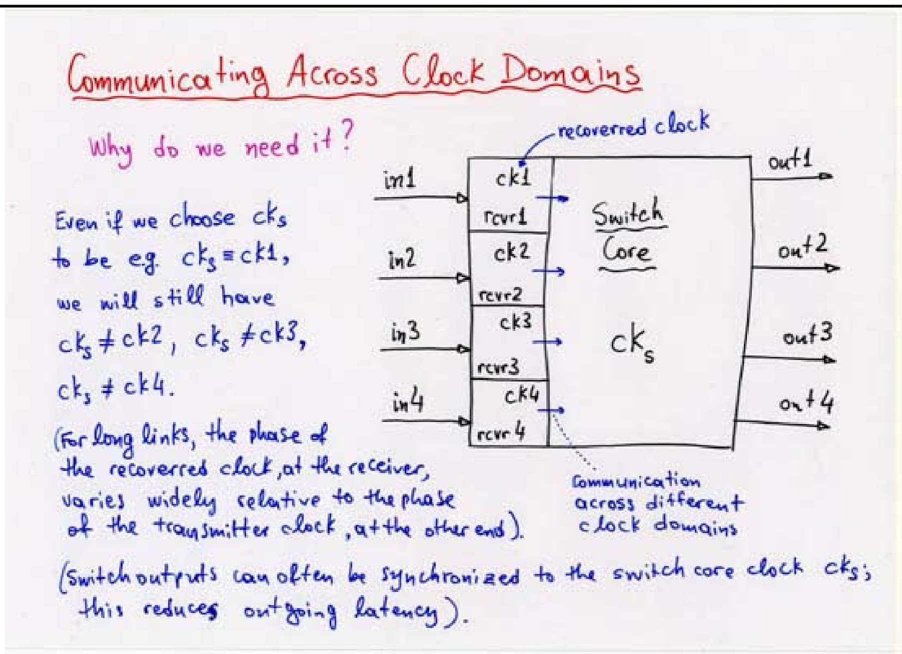
CS-534 – Univ. of Crete and FORTH, Greece

`http://archvlsi.ics.forth.gr/~kateveni/534`

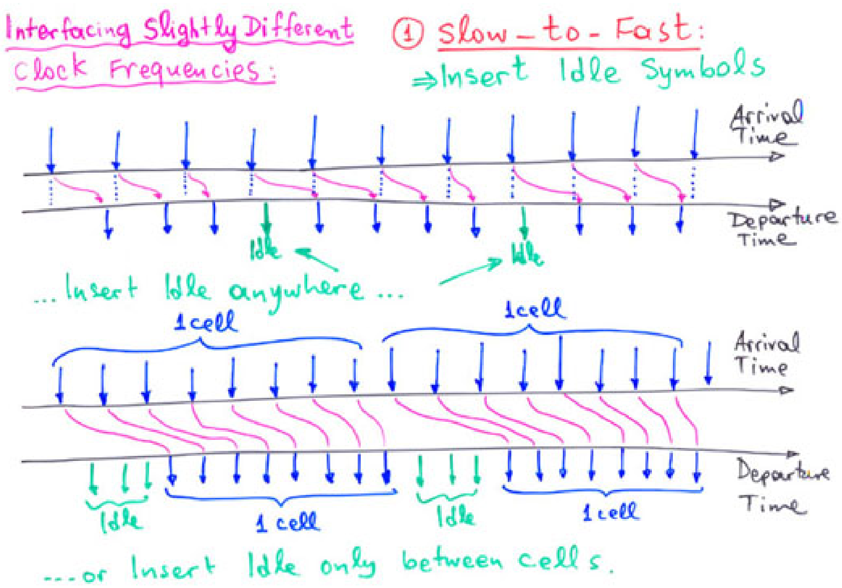
2.A Elastic Buffers for Cross-Clock Communication

Table of Contents:

- Oscillator frequency deviation
  - have *idle* symbols in the flow, for removal/insertion at interfaces
- Single-bit communication
  - either 0 or 1 is OK, but *not* intermediate – *not metastable*
- Multi-bit word communication, at almost 1 word/cycle rate
  - can*not* sample asynchronously to the source clock
  - need 2-port FIFO
  - read-pointer to write-pointer comparison problem
  - synchronizing the empty/full flags leads to low access rate
  - 1-hot (or 2-hot) encoding; compare to an old, conservative, synchronized version of the other-domain's pointer



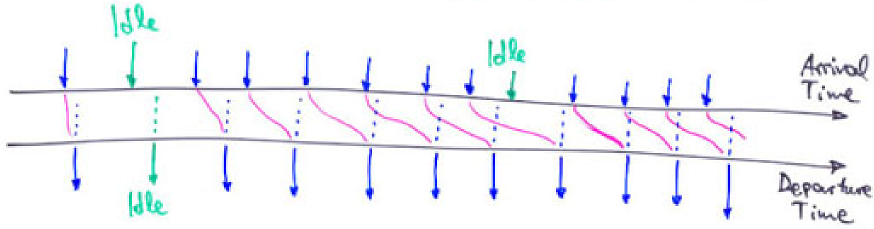
CS-534, Copyright Univ. of Crete



Interfacing Slightly Different Clock Frequencies:

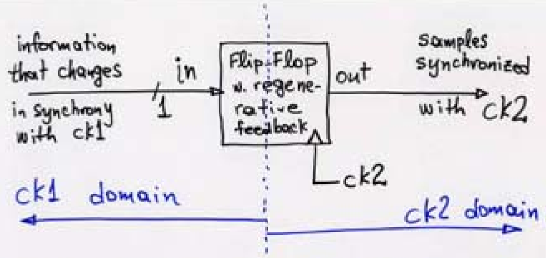
② Fast-to-Slow:

(α) ⇒ Remove Idle Symbols



and/or (β) ⇒ notify the transmitter, by feedback ("backpressure") to slow-down or wait, i.e. to insert more idle's (so that they can be removed) ...

Metastability, Synchronization Delay



Q: What happens if "in" changes at exactly the time when the active edge of ck2 comes?

A: "out" may get into "metastable" state (intermediate voltage) and stay in it for an undetermined duration t_{meta}.

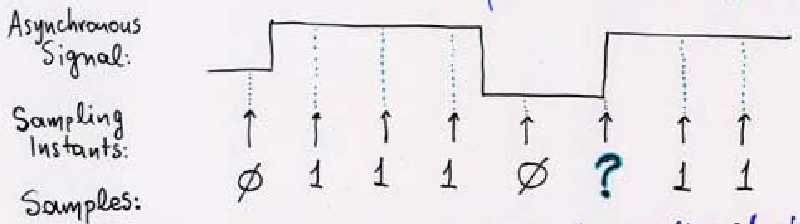
$$\text{Probability}\,(t_{meta} \geq \delta) \sim e^{-\frac{\delta}{\text{flip-flop feedback loop delay}}}$$

P(failure)...also ~ frequency of edges in "in" × frequency of ck2

Solution: do NOT use "out" for a time δ on the order of tens or hundreds of FF feedback loop delays (synchronization failure frequency < 1 in e.g. 100 years...)

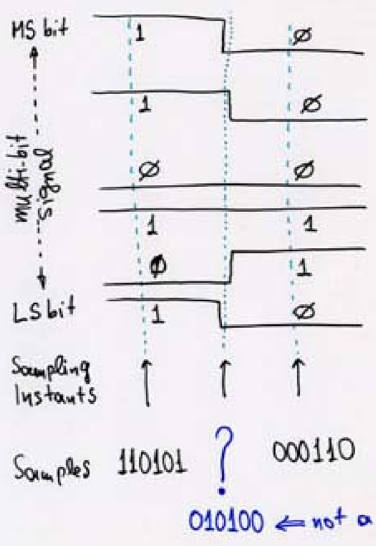
Was the Signal Sampled before or after its Change?

(A) Serial Signal Sampling: (Need $f_{sampling} > f_{signal\ change}$ in order to "see" all signal changes)



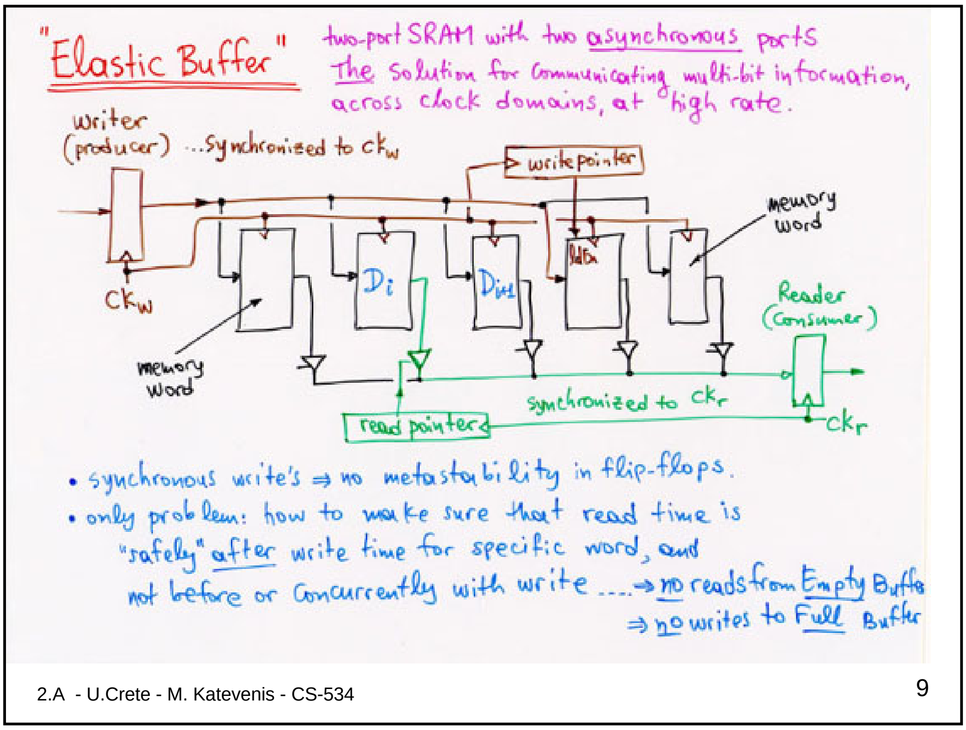
Since the signal changes asynchronously to the sampling clock, the sampling point "?" could have been a little before or a little after the signal changes, yielding either Ø or 1. Hence, it does not matter if the FF metastability eventually yields Ø or 1 – all that matters is that it becomes a valid binary value, that all of its receivers interpret in the same way.

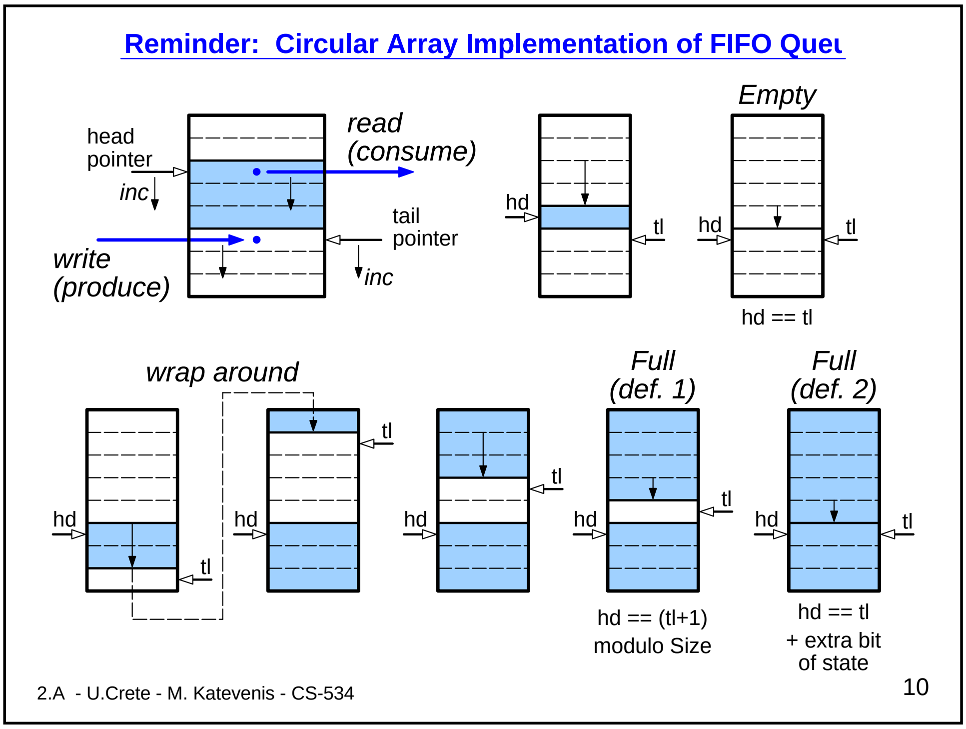
(B) Parallel Signal Sampling (Impossible w. asynchronous clock)



For multibit signals, asynchronous sampling is basically useless: there is no way to tell if the sampled word is a valid one (old or new) or an invalid word made by a random selection of bits from these two words.

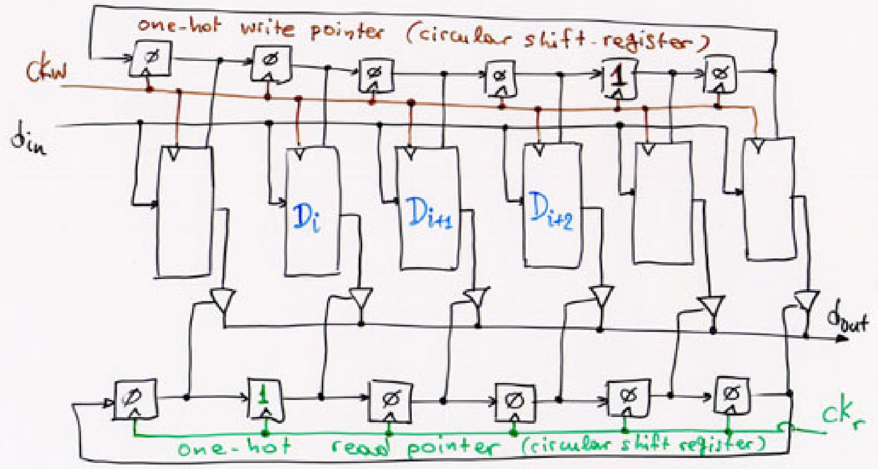
(Only possible solution: $f_{sampling} > 2 \times f_{signal\ change}$, then expect to see each word twice or more...)


"Elastic Buffer"
two-port SRAM with two asynchronous ports
The solution for communicating multi-bit information, across clock domains, at high rate.
writer (producer) ...synchronized to ckw
write pointer
memory word
Di
Di+1
ckw
Reader (Consumer)
memory word
read pointer
synchronized to ckr
ckr
• synchronous write's ⇒ no metastability in flip-flops.
• only problem: how to make sure that read time is "safely" after write time for specific word, and not before or concurrently with write ....⇒ no reads from Empty Buffer
⇒ no writes to Full Buffer
2.A - U.Crete - M. Katevenis - CS-534
9

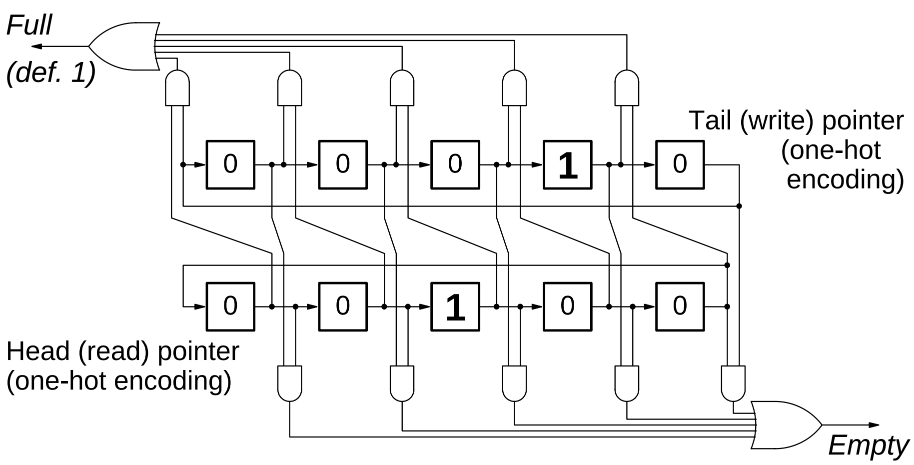

Reminder: Circular Array Implementation of FIFO Queu
head pointer
inc
read (consume)
write (produce)
tail pointer
inc
hd
tl
Empty
hd == tl
wrap around
Full (def. 1)
Full (def. 2)
hd == (tl+1) modulo Size
hd == tl + extra bit of state
2.A - U.Crete - M. Katevenis - CS-534
10

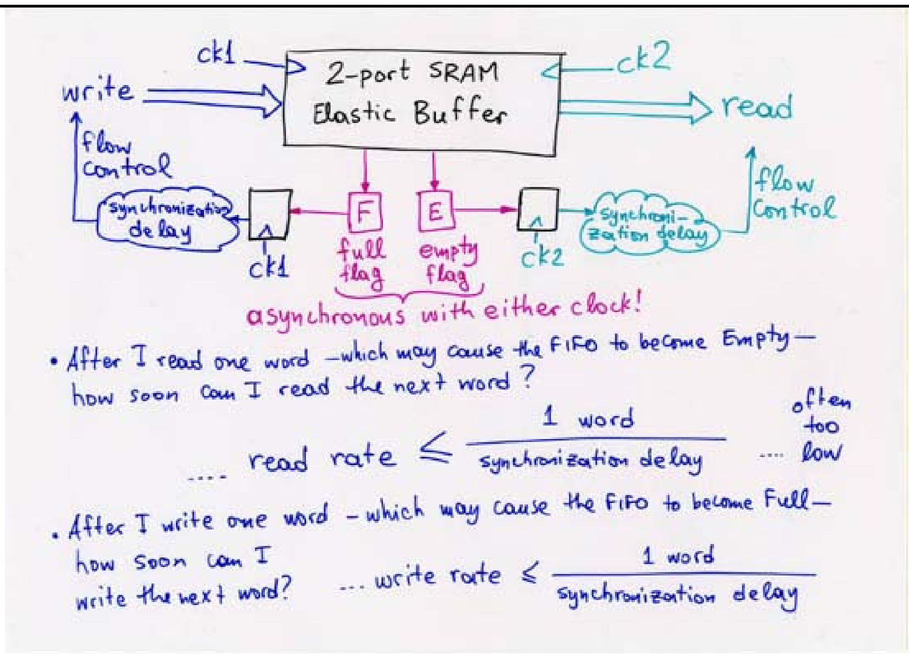
- synchronizing multi-bit values (read/write pointers) is very hard
- comparing asynchronous multi-bit values (pointers) is very hard

⇒ use "decoded pointer" state... "One-Hot" Pointer Encoding

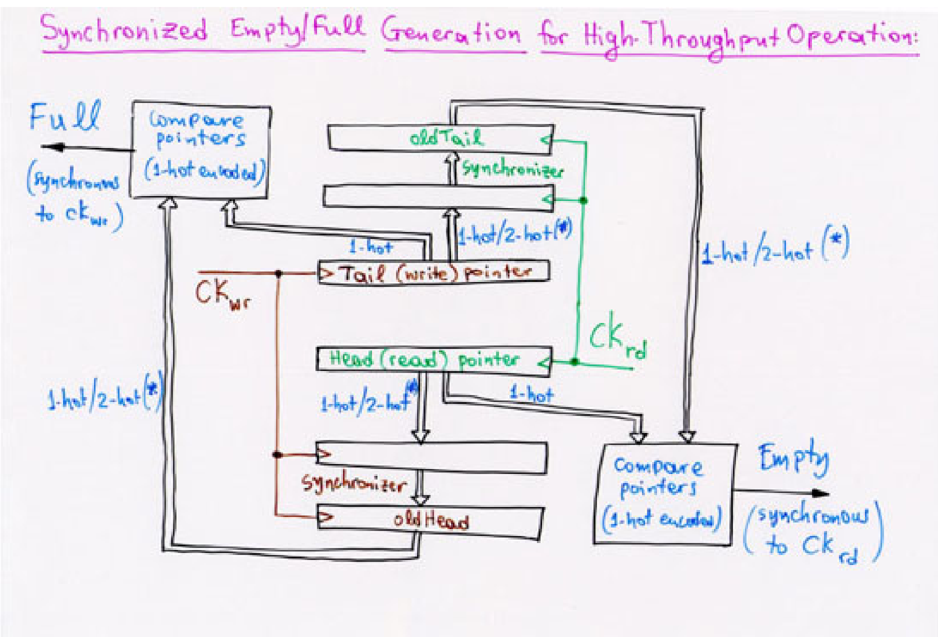


Empty/Full FIFO Detection using One-Hot Pointer Encodin





2.A - U.Crete - M. Katevenis - CS-534 13



2.A - U.Crete - M. Katevenis - CS-534 14

Timing & Synchronicity of Full & Empty Flags

- **Full** flag – Synchronous to ck_{wr}
  - asserted as soon as a write op. fills the FIFO up (def.1 "full")
  - negated after a word is read from the FIFO and the synchronization delay elapses
- **Empty** flag – Synchronous to ck_{rd}
  - asserted as soon as a read operation empties the FIFO
  - negated after a word is written into the FIFO and the synchronization delay elapses

- *Reference* on Synchronization and Elastic Buffers: *W. Dally, J. Poulton: "Digital Systems Engineering"*, Cambridge University Press, 1998, ISBN 0-521-59292-5 (sections 10.2 and 10.3 –especially 10.3.4.2).

2.A - U.Crete - M. Katevenis - CS-534 15

Sampling 1-hot pointers for synchronization purposes: 1-hot/2-hot versions

- A 1-hot encoded pointer is a multi-bit value.
- When sampling any such value with an asynchronous clock for synchronization purposes, there is always the possibility that *some bits are sampled* "before" and some "after" they transition.
- This may result in the sampled pointer containing 2 bits ON, or 1 bit ON, or no bit ON (2-hot, or 1-hot, or 0-hot).
- 2-hot is "OK": conservative!
- 1-hot is normal.
- 0-hot is bad: empty/full is not asserted even when the FIFO is in one of these states ➔ we have to ensure that 0-hot never happens!
- ⇒ Use a 1-hot/2-hot version of the pointer for synchronization purposes: make sure that the new "hot" bit is turned ON safely before the old "hot" bit is turned OFF (e.g. use appropriate OR function of master & slave flip-flops).

2.A - U.Crete - M. Katevenis - CS-534 16