EECS150 - Digital Design

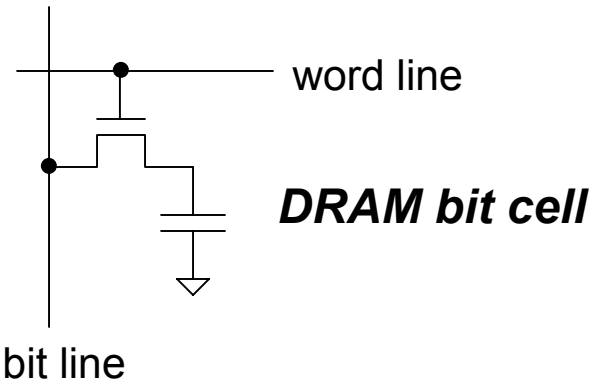
Lecture 17 – Memory 2

October 22, 2002

John Wawrzynek

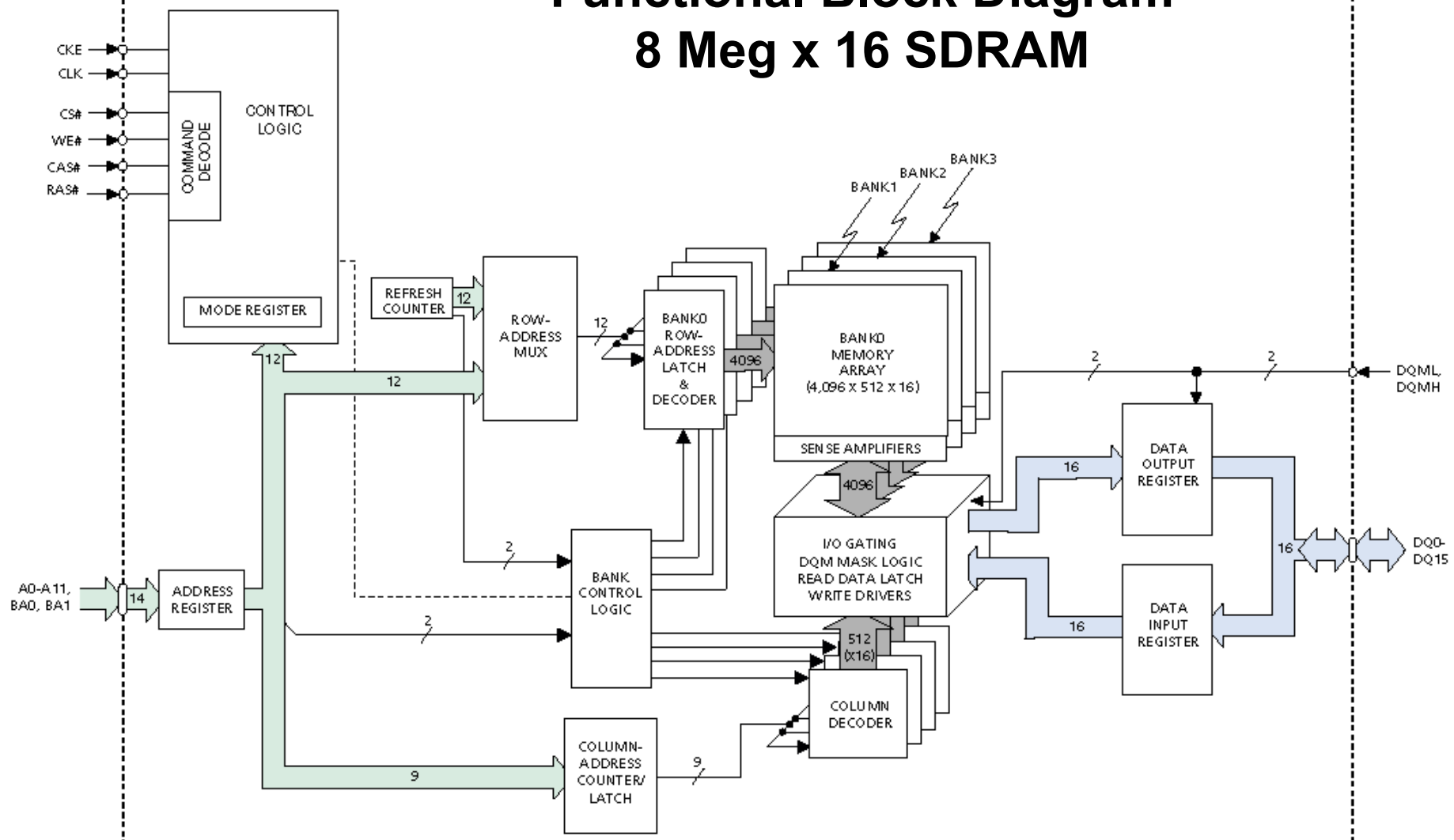
SDRAM Recap

- General Characteristics
  - Optimized for high density and therefore low cost/bit
  - Special fabrication process – usually on a separate chip from processor
  - Needs periodic refresh (in most applications)
  - Relatively slow because:
    - High capacity leads to large cell arrays with high word- and bit-line capacitance
    - Complex read/write cycle. Read needs "precharge" and write-back



  - Multiple clock cycles per read or write access
  - Multiple reads and writes are often grouped together to amortize overhead. Referred to as "bursting".

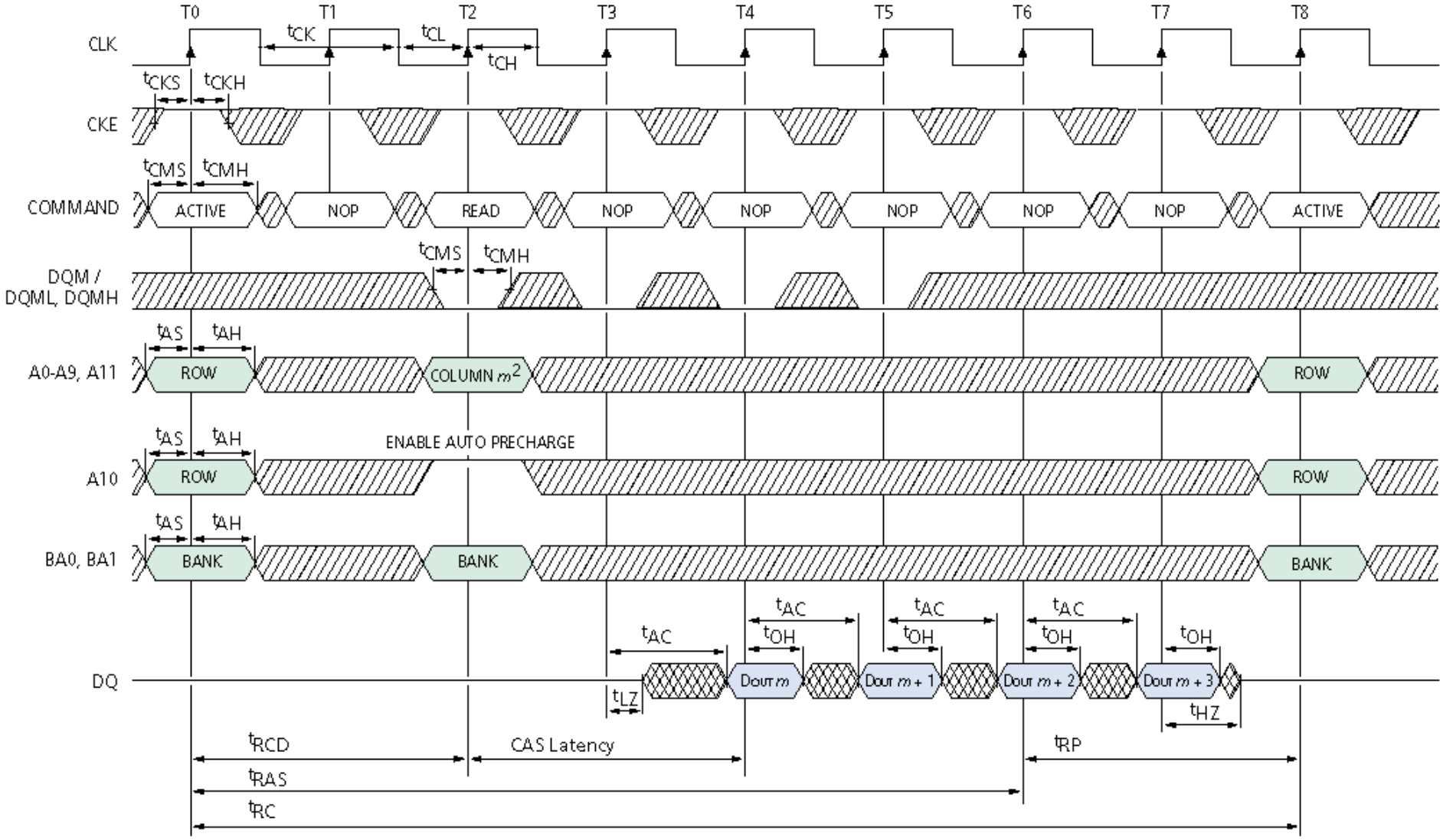
Functional Block Diagram
8 Meg x 16 SDRAM



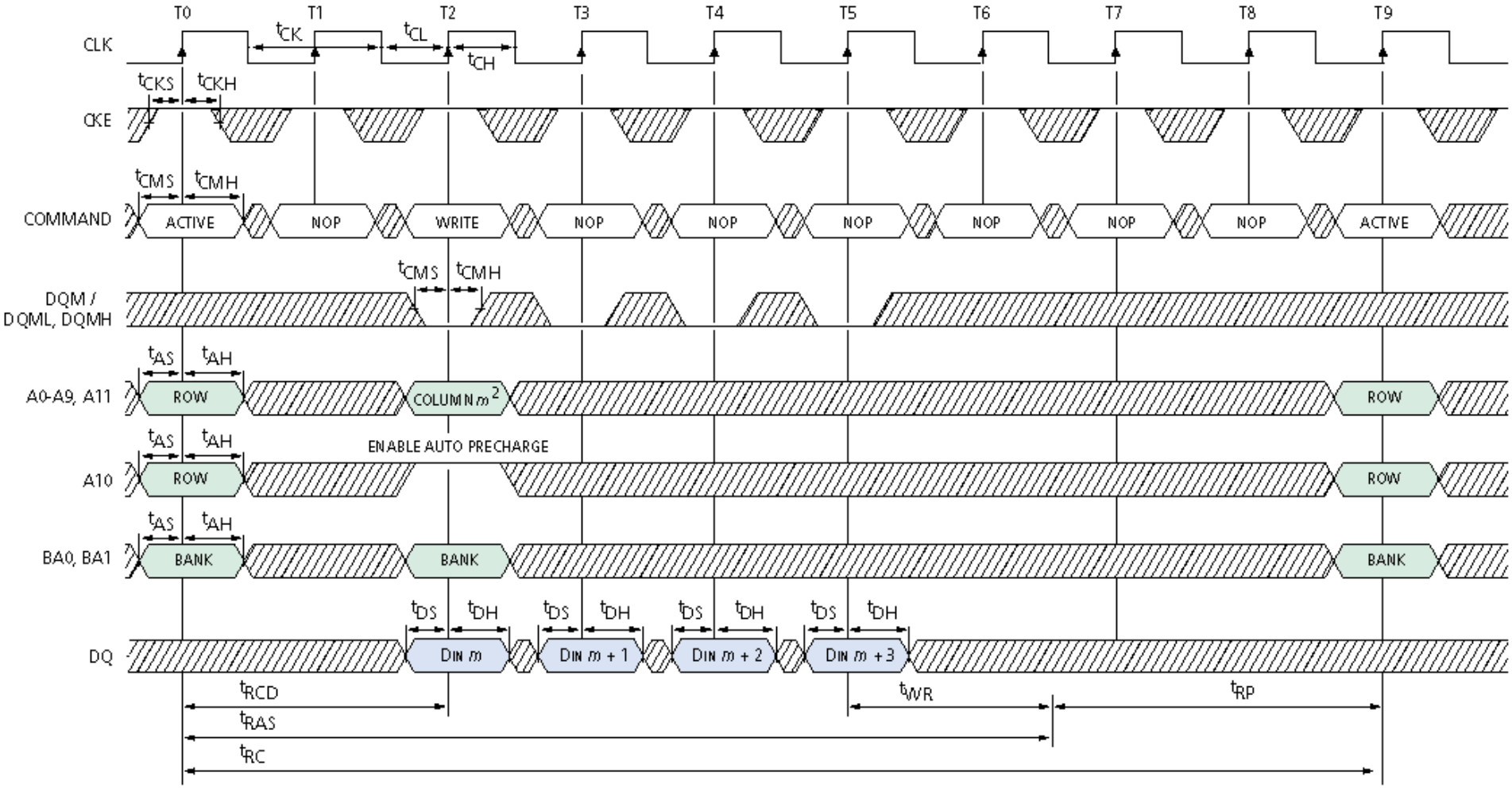
SDRAM Details

- Multiple "banks" of cell arrays are used to reduce access time:
  - Each bank is 4K rows by 512 "columns" by 16 bits (for our part)
- Read and Write operations as split into RAS (row access) followed by CAS (column access)
- These operations are controlled by sending commands
  - Commands are sent using the RAS, CAS, CS, & WE pins.
- Address pins are "time multiplexed"
  - During RAS operation, address lines select the bank and row
  - During CAS operation, address lines select the column.
- "ACTIVE" command "opens" a row for operation
  - transfers the contents of the entire to a row buffer
- Subsequent "READ" or "WRITE" commands modify the contents of the row buffer.
- For burst reads and writes during "READ" or "WRITE" the starting address of the block is supplied.
  - Burst length is programmable as 1, 2, 4, 8 or a "full page" (entire row) with a burst terminate option.
- Special commands are used for initialization (burst options etc.)
- A burst operation takes $\approx 4 + n$ cycles (for n words)

READ burst (with auto precharge)



WRITE burst (with auto precharge)

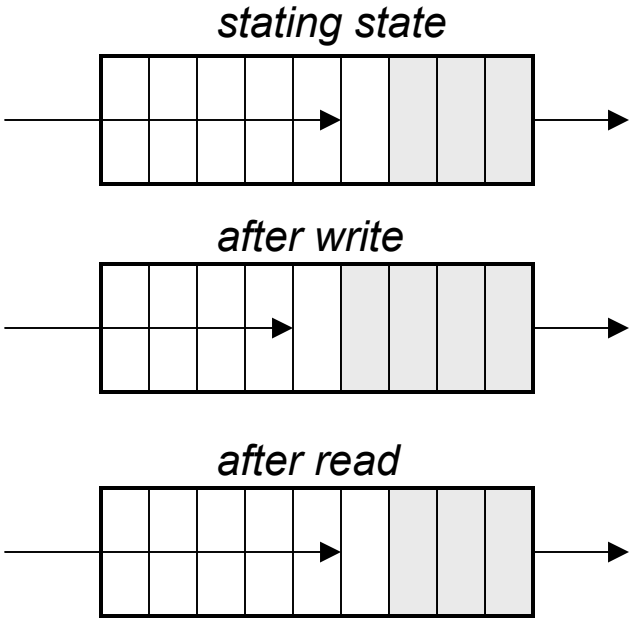


See datasheet for more details.

Verilog simulation models available.

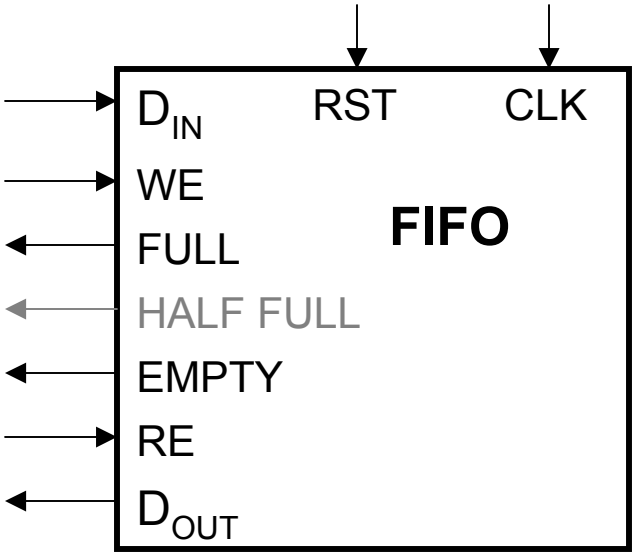
First-in-first-out (FIFO) Memory

- Used to implement *queues*.
- These find common use in computers and communication circuits.
- Generally, used for rate matching data producer and consumer:



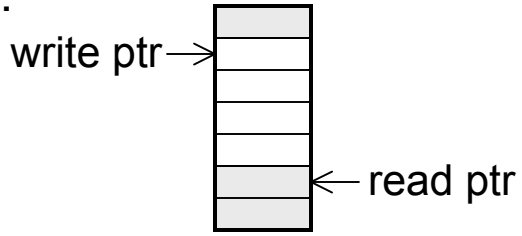
- Producer can perform many writes without consumer performing any reads (or vis versa). However, because of finite buffer size, on average, need equal number of reads and writes.
- Typical uses:
  - interfacing I/O devices. Example network interface. Data bursts from network, then processor bursts to memory buffer (or reads one word at a time from interface). Operations not synchronized.
  - Example: Audio output. Processor produces output samples in bursts (during process swap-in time). Audio DAC clocks it out at constant sample rate.

FIFO Interfaces

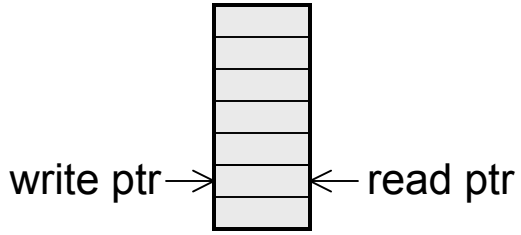


- After write or read operation, FULL and EMPTY indicate status of buffer.
- Used by external logic to control own reading from or writing to the buffer.
- FIFO resets to EMPTY state.
- HALF FULL (or other indicator of partial fullness) is optional.

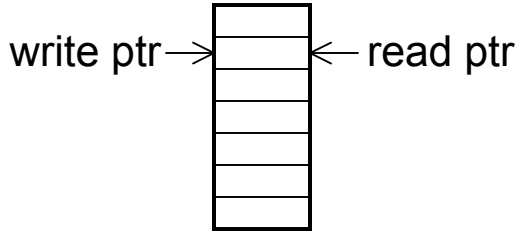
- Address pointers pointers are used internally to keep next write position and next read position into a dual-port memory.



- If pointers equal after write ⇒ FULL:



- If pointers equal after read ⇒ EMPTY:

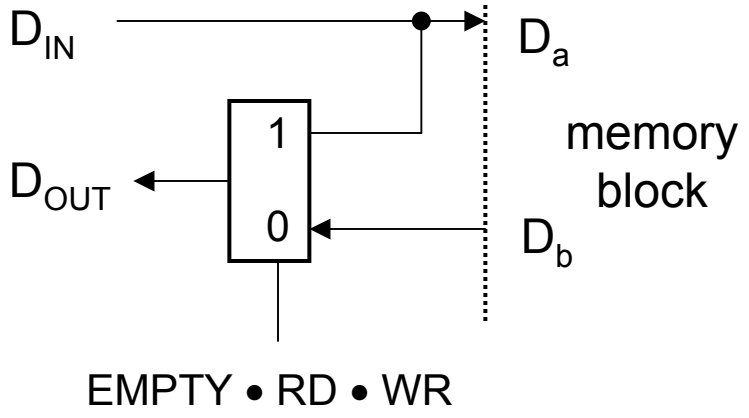


FIFO Implementation Details

- Assume, dual-port memory with asynchronous read, synchronous write.
- Binary counter for each of read and write address. CEs controlled by WE and RE.
- Equal comparator to see when pointers match.
- Flip-flop each for FULL and EMPTY flags:

WE	RD	equal	$EMPTY_i$	$FULL_i$
0	0	0	0	0
0	0	1	$EMPTY_{i-1}$	$FULL_{i-1}$
0	1	0	0	0
0	1	1	1	0
1	0	0	0	0
1	0	1	0	1
1	1	0	0	0
1	1	1	$EMPTY_{i-1}$	$FULL_{i-1}$

- With this memory, read happens before write. Therefore if WE and RD are asserted:
  - When FULL, correct operation will happen.
  - When EMPTY, would like to write before read.
  - Correct this case with a *bypass mux:*



Xilinx BlockRam Versions

- Our simple version has latency problems.
  - FULL and EMPTY signals are asserted near the end of the clock period.
- Xilinx version solves this by "predicting" when full or empty will be asserted on next write/read operation. Also uses BlockRam with synchronous reads.
- Available on Xilinx website along with "app note" – application note. These are linked to our page.
- Two versions available. Easy to modify to change the width if necessary.
- Will use the "cc" (common clock) version for checkpoint 2.
- Can use the "ic" (independent clock) version later for bridging network to video.