Digital Design and Computer Organization

Memory Arrays - 2

Reetinder Sidhu

Department of Computer Science and Engineering

Digital Design and Computer Organization

Memory Arrays - 2

Reetinder Sidhu

Department of Computer Science and Engineering

Memory Arrays - 2

Course Outline

- Digital Design
  - Combinational logic design
  - Sequential logic design
    - ★ **Memory Arrays - 2**
- Computer Organization
  - Architecture (microprocessor instruction set)
  - Microarchitecure (microprocessor operation)

Concepts covered

- Memory Arrays

Memory Arrays - 2
Arrays in Software

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays
- Two operations are supported on arrays:

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays
- Two operations are supported on arrays:
  - Read an array location

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays
- Two operations are supported on arrays:
  - Read an array location
    - Ex: `x = a[1];`

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays
- Two operations are supported on arrays:
  - Read an array location
    - Ex: `x = a[1];`
  - Write an array location

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays
- Two operations are supported on arrays:
  - Read an array location
    - Ex: `x = a[1];`
  - Write an array location
    - Ex: `a[0] = 1;`

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays
- Two operations are supported on arrays:
  - Read an array location
    - Ex: `x = a[1];`
  - Write an array location
    - Ex: `a[0] = 1;`
- Can an array be implemented in hardware, as a logic circuit?

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays
- Two operations are supported on arrays:
  - Read an array location
    - Ex: `x = a[1];`
    - Input is array index
    - Output is the value stored at the index
  - Write an array location
    - Ex: `a[0] = 1;`

- Can an array be implemented in hardware, as a logic circuit?

Memory Arrays - 2
Arrays in Software

- Most high level languages (like C, C++ and Java) provide arrays
- Two operations are supported on arrays:
  - Read an array location
    - Ex: `x = a[1];`
    - Input is array index
    - Output is the value stored at the index
  - Write an array location
    - Ex: `a[0] = 1;`
    - Inputs are array index and the value to be stored at that index
    - Value stored at index location (no output)
- Can an array be implemented in hardware, as a logic circuit?

Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

- The term index for software arrays means the same as **address** for hardware arrays
  - Short form **addr** typically used

Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

- The term index for software arrays means the same as **address** for hardware arrays
  - Short form **addr** typically used
- For *n* memory locations, address should have $\lceil \log_2 n \rceil$ bits

Memory Arrays - 2

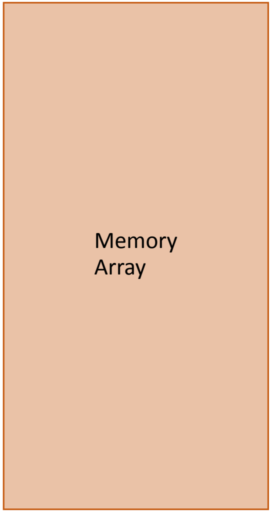
How to Construct a Memory Array Logic Circuit?

- The term index for software arrays means the same as **address** for hardware arrays
  - Short form **addr** typically used
- For n memory locations, address should have $\lceil \log_2 n \rceil$ bits
- Let start by constructing a simple memory that:
  - has eight locations
  - can store a single bit in each location
  - can perform a read or a write every clock cycle

Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

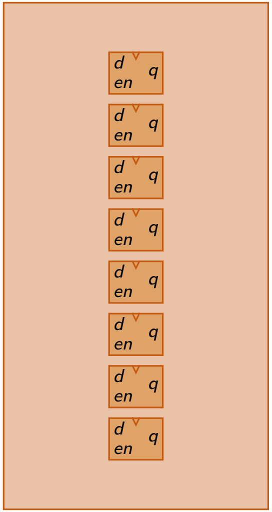
- The term index for software arrays means the same as **address** for hardware arrays
  - Short form **addr** typically used
- For n memory locations, address should have $\lceil \log_2 n \rceil$ bits
- Let start by constructing a simple memory that:
  - has eight locations
  - can store a single bit in each location
  - can perform a read or a write every clock cycle



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

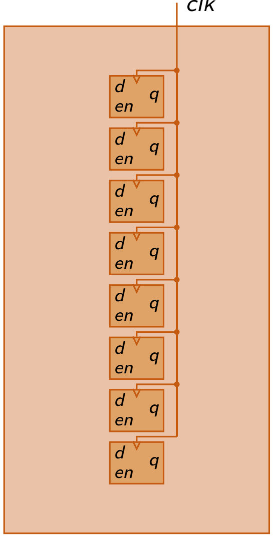
- The term index for software arrays means the same as **address** for hardware arrays
  - Short form **addr** typically used
- For n memory locations, address should have $\lceil \log_2 n \rceil$ bits
- Let start by constructing a simple memory that:
  - has eight locations
  - can store a single bit in each location
  - can perform a read or a write every clock cycle
- Eight flip-flops required for storage



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

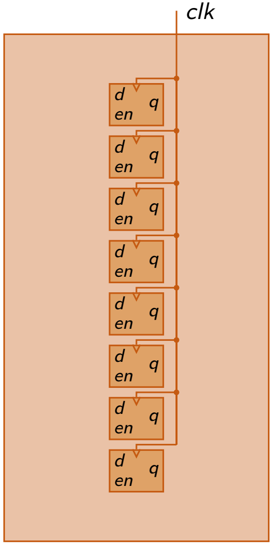
- The term index for software arrays means the same as **address** for hardware arrays
  - Short form **addr** typically used
- For n memory locations, address should have $\lceil \log_2 n \rceil$ bits
- Let start by constructing a simple memory that:
  - has eight locations
  - can store a single bit in each location
  - can perform a read or a write every clock cycle
- Eight flip-flops required for storage



Memory Arrays - 2

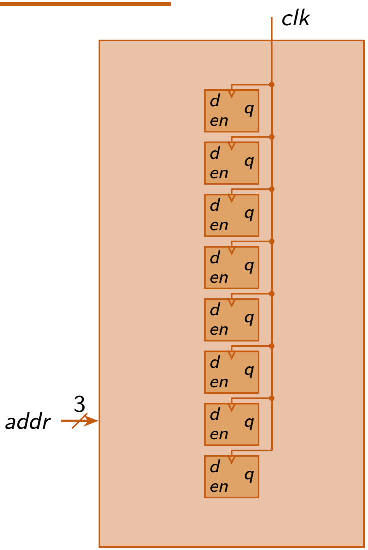
How to Construct a Memory Array Logic Circuit?

- Read an array location



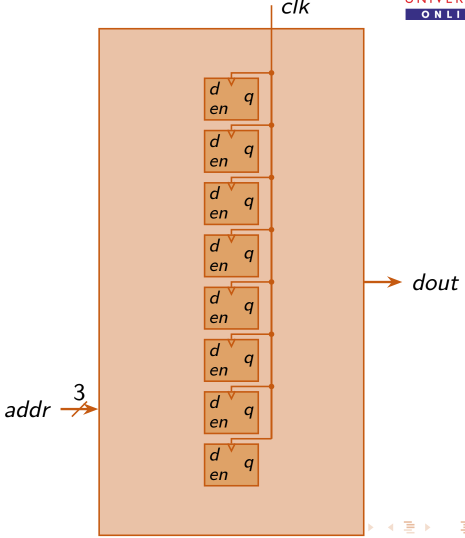
Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

- Read an array location
  - Input is address to be read (*addr*)



Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

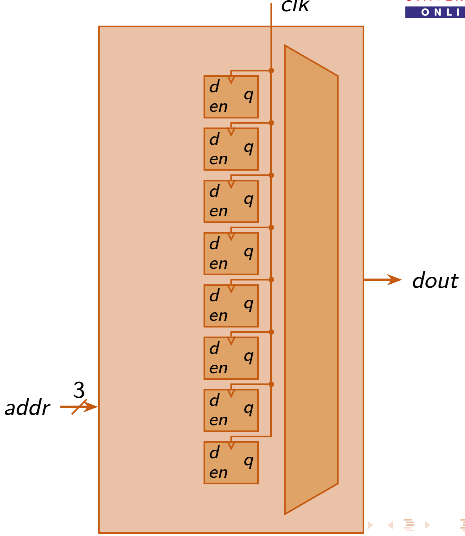
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)



MEMORY ARRAYS - 2

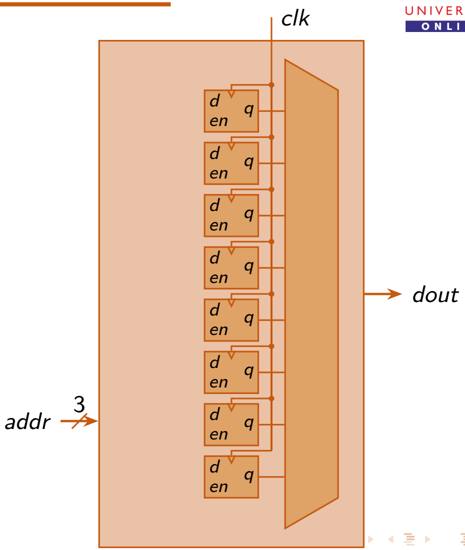
How to Construct a Memory Array Logic Circuit?

- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required



Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

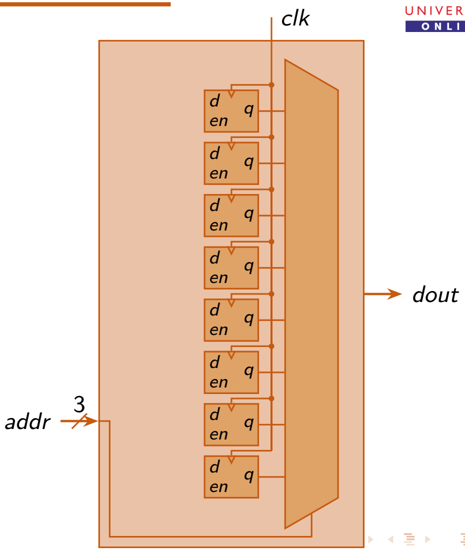
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

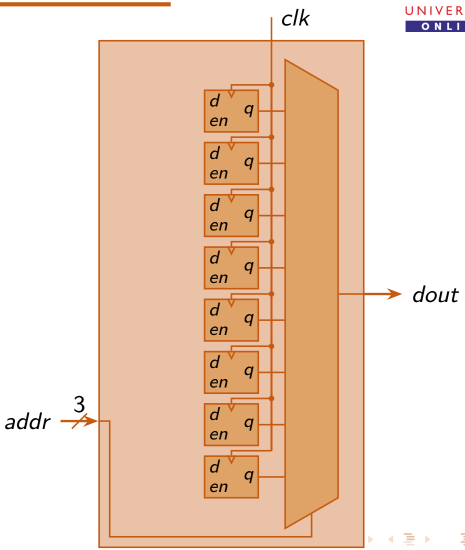
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required



Memory Arrays - 2

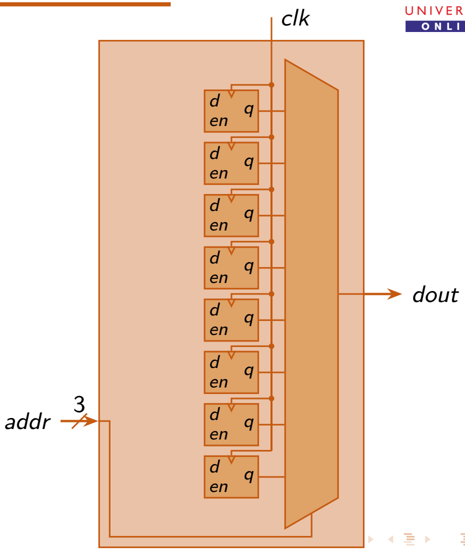
How to Construct a Memory Array Logic Circuit?

- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required



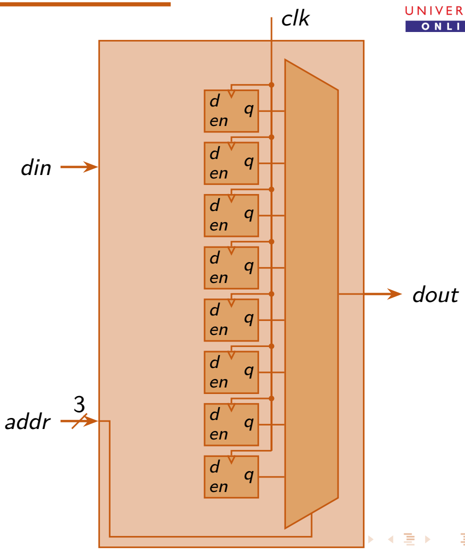
Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location



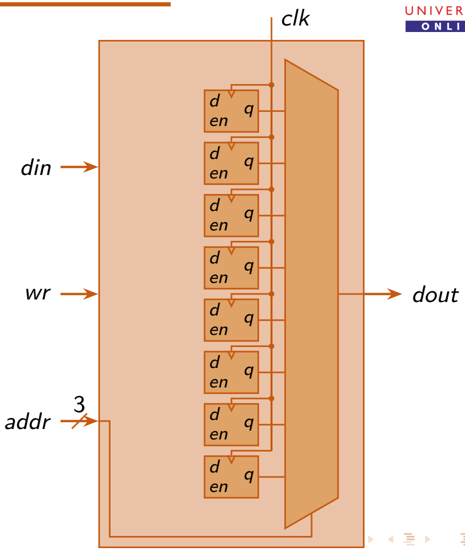
Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location
  - Inputs are address to be written (*addr*) and the value to be stored (*din*) at that address



Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

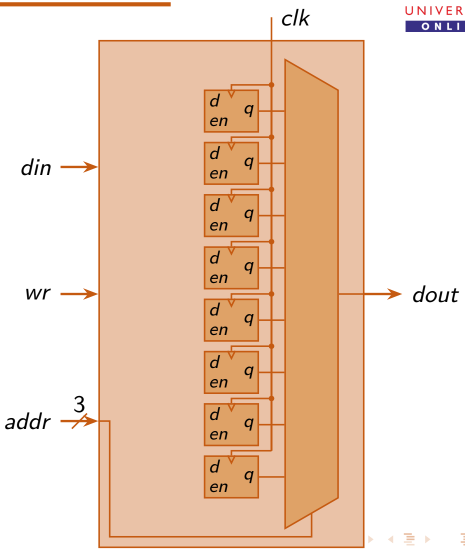
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location
  - Inputs are address to be written (*addr*) and the value to be stored (*din*) at that address
  - Another input is write signal (*wr*) which if 1 indicates a write operation



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

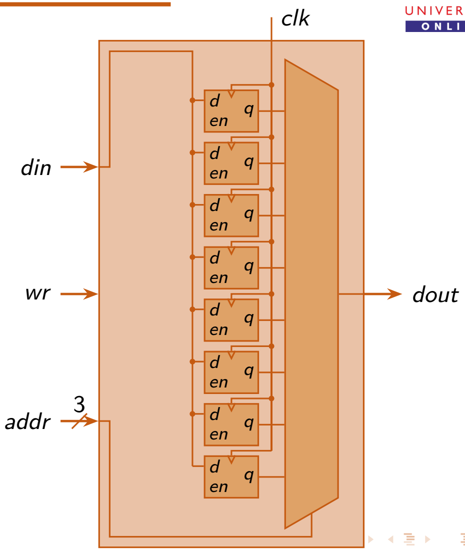
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location
  - Inputs are address to be written (*addr*) and the value to be stored (*din*) at that address
  - Another input is write signal (*wr*) which if 1 indicates a write operation
  - Value stored at index location (no output)



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

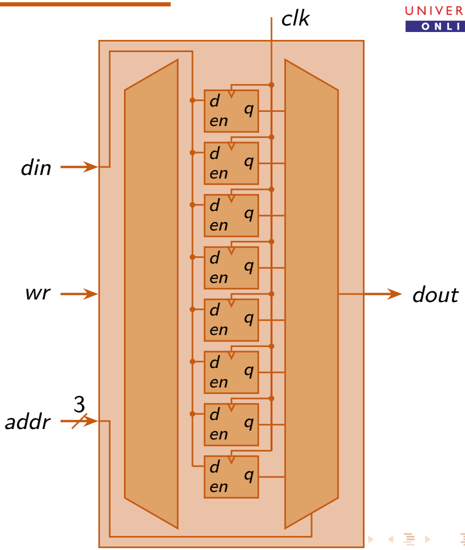
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location
  - Inputs are address to be written (*addr*) and the value to be stored (*din*) at that address
  - Another input is write signal (*wr*) which if 1 indicates a write operation
  - Value stored at index location (no output)



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

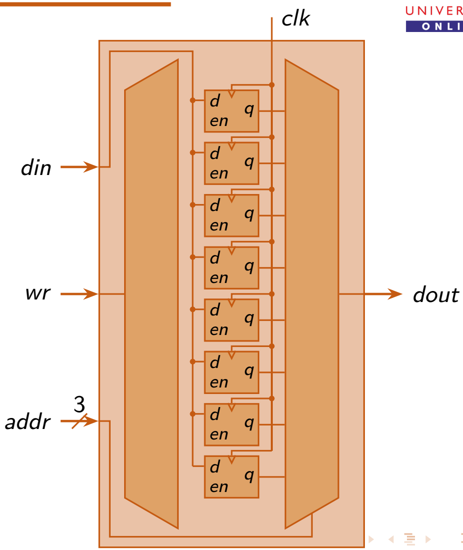
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location
  - Inputs are address to be written (*addr*) and the value to be stored (*din*) at that address
  - Another input is write signal (*wr*) which if 1 indicates a write operation
  - Value stored at index location (no output)



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

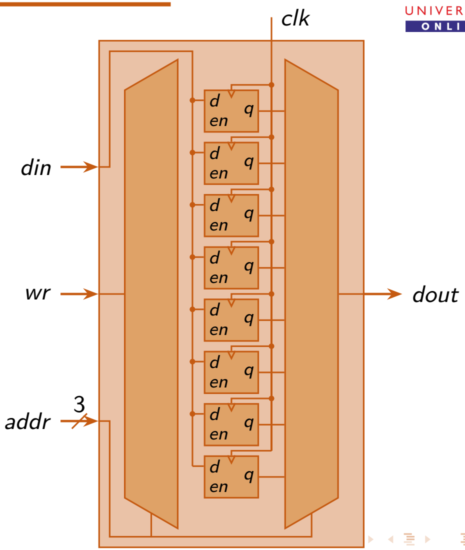
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location
  - Inputs are address to be written (*addr*) and the value to be stored (*din*) at that address
  - Another input is write signal (*wr*) which if 1 indicates a write operation
  - Value stored at index location (no output)



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

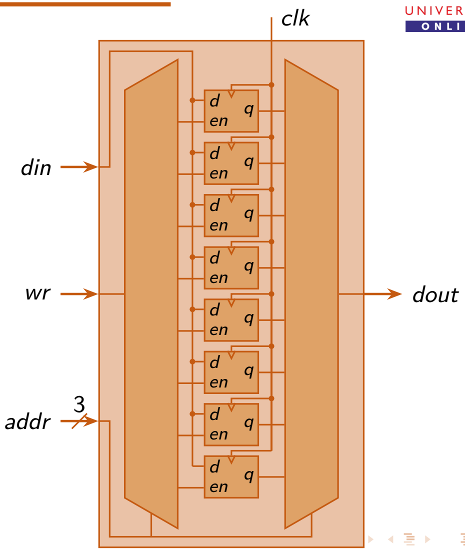
- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location
  - Inputs are address to be written (*addr*) and the value to be stored (*din*) at that address
  - Another input is write signal (*wr*) which if 1 indicates a write operation
  - Value stored at index location (no output)



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

- Read an array location
  - Input is address to be read (*addr*)
  - Output is the value stored at the address (*dout*)
  - So 8:1 mux required
- Write an array location
  - Inputs are address to be written (*addr*) and the value to be stored (*din*) at that address
  - Another input is write signal (*wr*) which if 1 indicates a write operation
  - Value stored at index location (no output)



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

- We extend our construction to a memory array of eight locations each of which can store 16 bits

Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

- We extend our construction to a memory array of eight locations each of which can store 16 bits
  - Above array size is specified as 8-word $\times$ 16-bit or simply 8 $\times$ 16

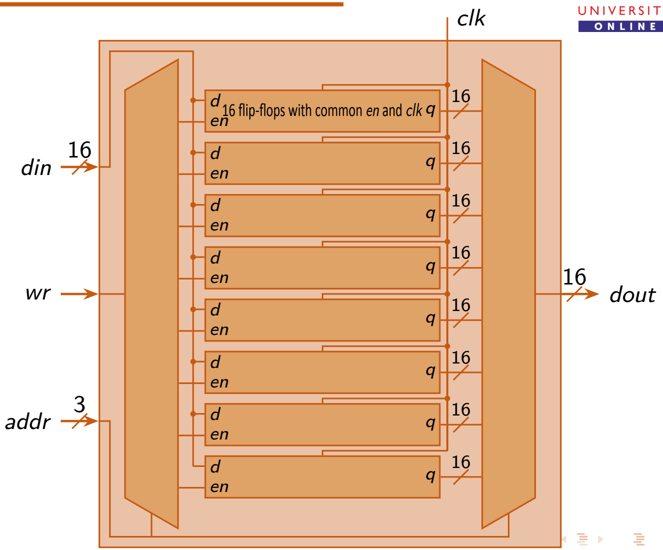
Memory Arrays - 2
How to Construct a Memory Array Logic Circuit?

- We extend our construction to a memory array of eight locations each of which can store 16 bits
  - Above array size is specified as 8-word $\times$ 16-bit or simply 8 $\times$ 16
- Each row now contains 16 flip-flops with common *en* and *clk* inputs
- The 8:1 mux now has 16-bit data inputs and output

Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

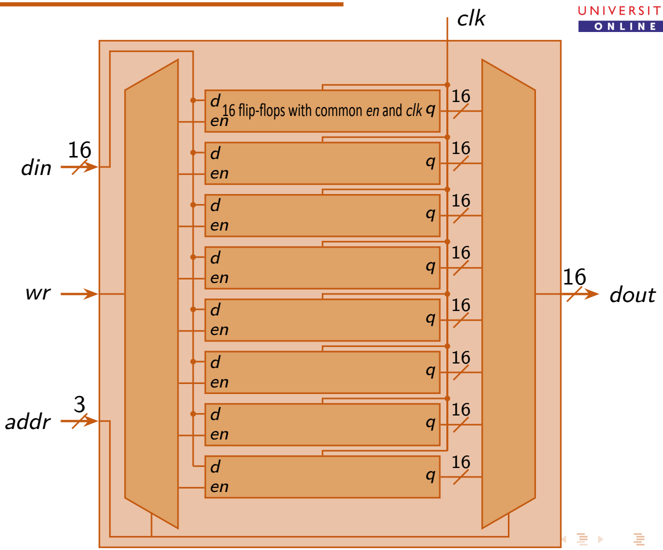
- We extend our construction to a memory array of eight locations each of which can store 16 bits
  - Above array size is specified as 8-word $\times$ 16-bit or simply 8×16
- Each row now contains 16 flip-flops with common *en* and *clk* inputs
- The 8:1 mux now has 16-bit data inputs and output



Memory Arrays - 2

How to Construct a Memory Array Logic Circuit?

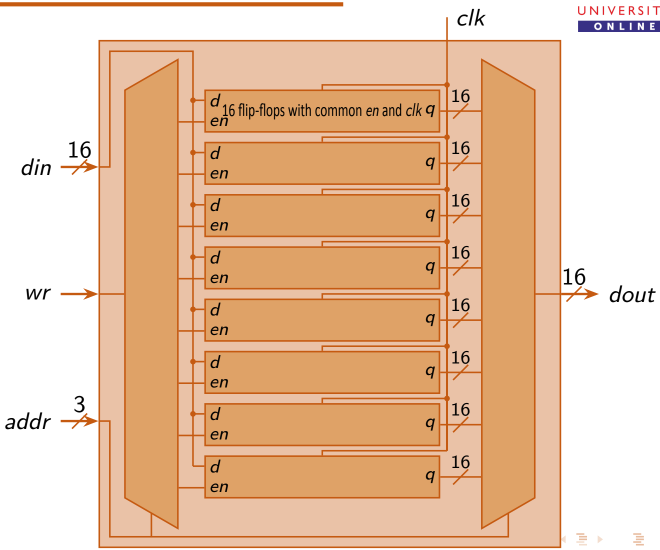
- We extend our construction to a memory array of eight locations each of which can store 16 bits
  - Above array size is specified as 8-word $\times$ 16-bit or simply 8 $\times$ 16
- Each row now contains 16 flip-flops with common *en* and *clk* inputs
- The 8:1 mux now has 16-bit data inputs and output
- In general, an $n \times m$ array contains n **words** of m bits each



Memory Arrays - 2

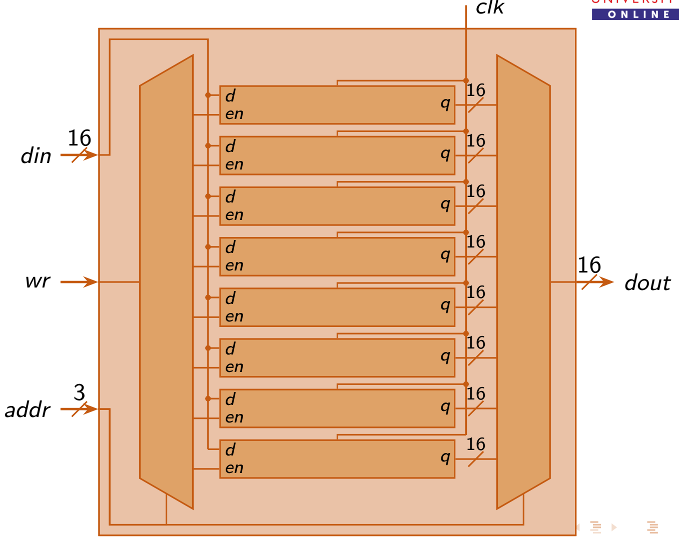
How to Construct a Memory Array Logic Circuit?

- We extend our construction to a memory array of eight locations each of which can store 16 bits
  - Above array size is specified as 8-word $\times$ 16-bit or simply 8 $\times$ 16
- Each row now contains 16 flip-flops with common *en* and *clk* inputs
- The 8:1 mux now has 16-bit data inputs and output
- In general, an $n \times m$ array contains n **words** of m bits each
  - Each memory element is called a **bit cell**



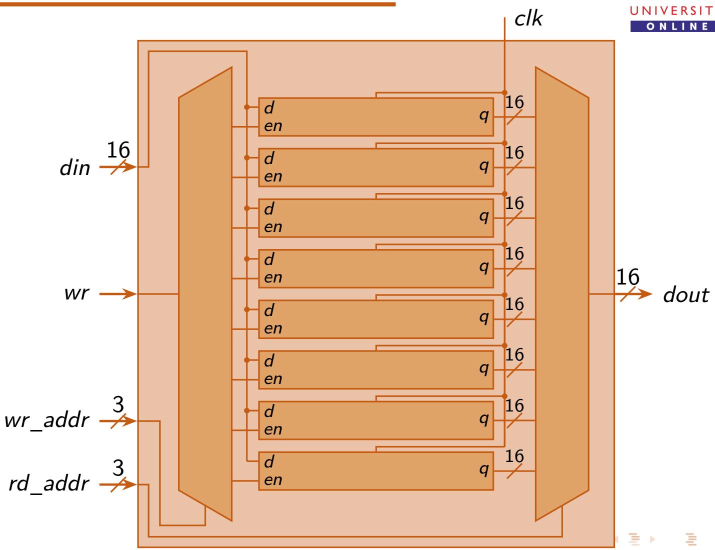
Memory Arrays - 2
Memory Ports

- A memory **port** is a set of signals that provide read and/or write access to a memory address in the array
- One read / write port:
  - *addr*, *wr*, *din* and *dout*



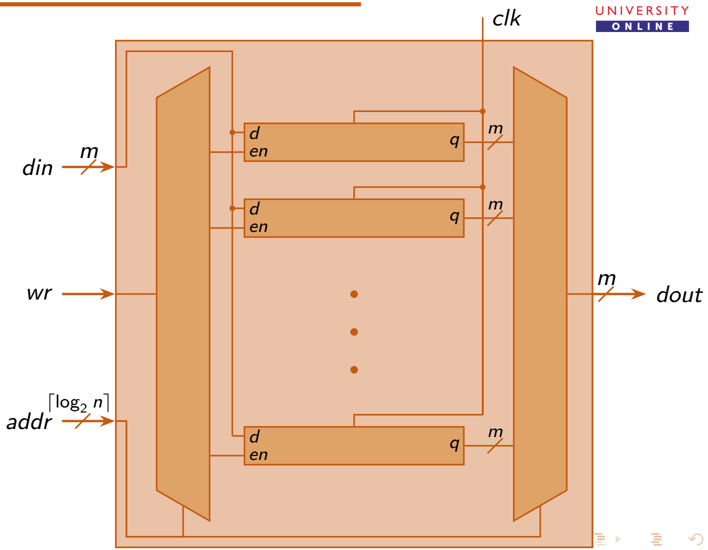
Memory Arrays - 2
Memory Ports

- A memory **port** is a set of signals that provide read and/or write access to a memory address in the array
- Two ports
- One read port:
  - *rd_addr* and *dout*
- One write port:
  - *wr_addr*, *wr* and *din*



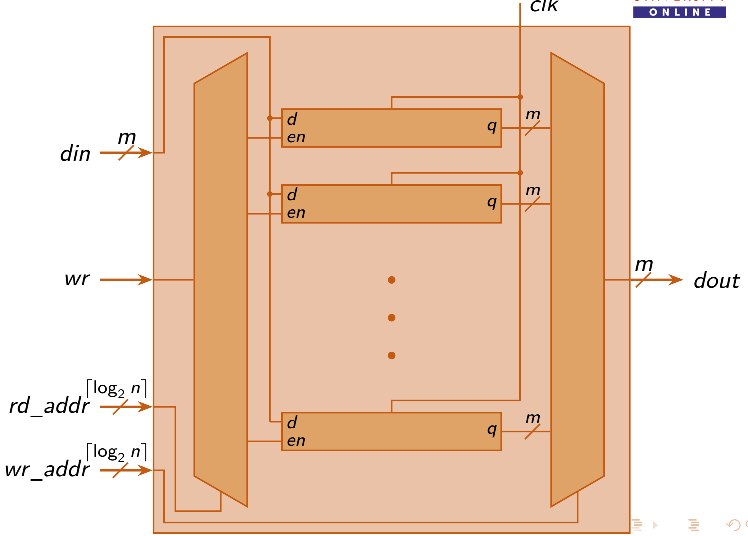
Memory Arrays - 2
Memory Array Organization

- Structure of a memory array of n-word $\times$ m-bit is shown
- Implementation of a small register file would have similar structure
- Other larger arrays may have different internal structures
  - Such as using a decoder with wordlines and bitlines
  - Latches instead of flip-flops
- However, the internal structures of (random access) memory arrays are functionally equivalent to the shown structure



Memory Arrays - 2
Memory Array Organization

- Structure of a memory array of n-word $\times$ m-bit is shown
- Implementation of a small register file would have similar structure
- Other larger arrays may have different internal structures
  - Such as using a decoder with wordlines and bitlines
  - Latches instead of flip-flops
- However, the internal structures of (random access) memory arrays are functionally equivalent to the shown structure



Memory Arrays - 2
Types of Memory Arrays

Memory type	Transistors per bit cell	Latency	Application
flip-flop	20	fast	Register file
SRAM[1]	6	medium	CPU cache
DRAM[2]	1	slow	Main memory

[1]Static Random Access Memory
[2]Dynamic Random Access Memory

Memory Arrays - 2

Think About It

- We have seen a memory array with one read port and one write port
- In a lab assignment, you implement a memory array with two read ports
- Can you design a memory array with two write ports?
    - What additional concerns, if any, would you need to handle?