Digital Logic Circuits
'Shift Registers and Counters'

ELEC2200

Summer 2009

David J. Broderick
brodedj@auburn.edu
http://www.auburn.edu/~brodedj
Office: Broun 360

Outline

- Data transmission
- Shift registers
- Counters
  - Async
  - Sync
  - BCD
  - Modulo-A
- RAM

Serial vs. Parallel

- Data transmission of n bits
- Parallel
  - All bits at once
  - n connections between tx and rx
  - 1 time step to get all data
- Serial
  - One bit at a time
  - n time steps to get all data
  - 1 connection between tx and rx

Serial vs Parallel

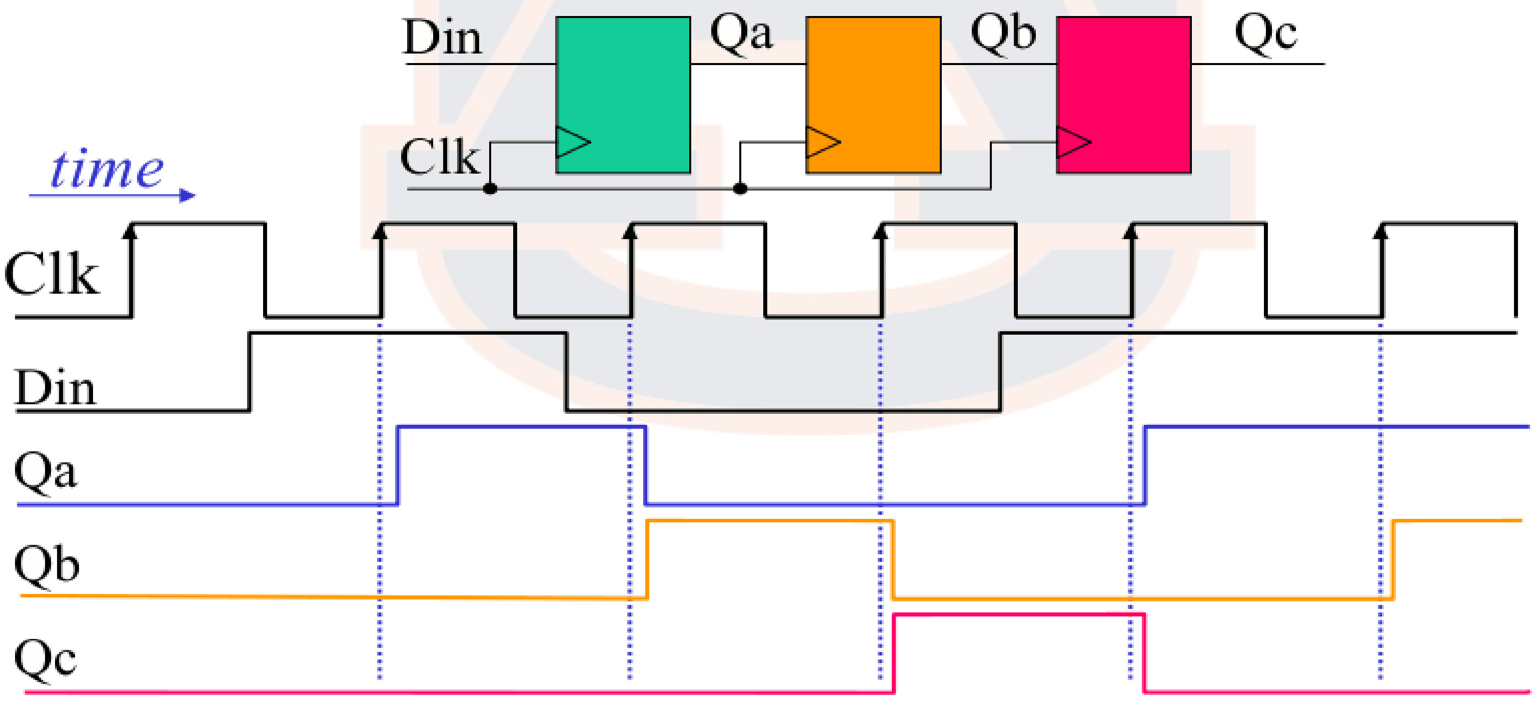
- Serial/Parallel
  - Both types of transmission
  - Large data set broken up into smaller 'words'
- Trade-offs:
  - # of inputs/outputs
  - Speed of transmission

Shift Registers

- Basis for storing binary values
  - To store n-bits, n memory elements required
  - We'll use D flip-flops here
- Also used for transmission of data
  - Values can be loaded either serially or in parallel
  - Values can be read out either serially or in parallel

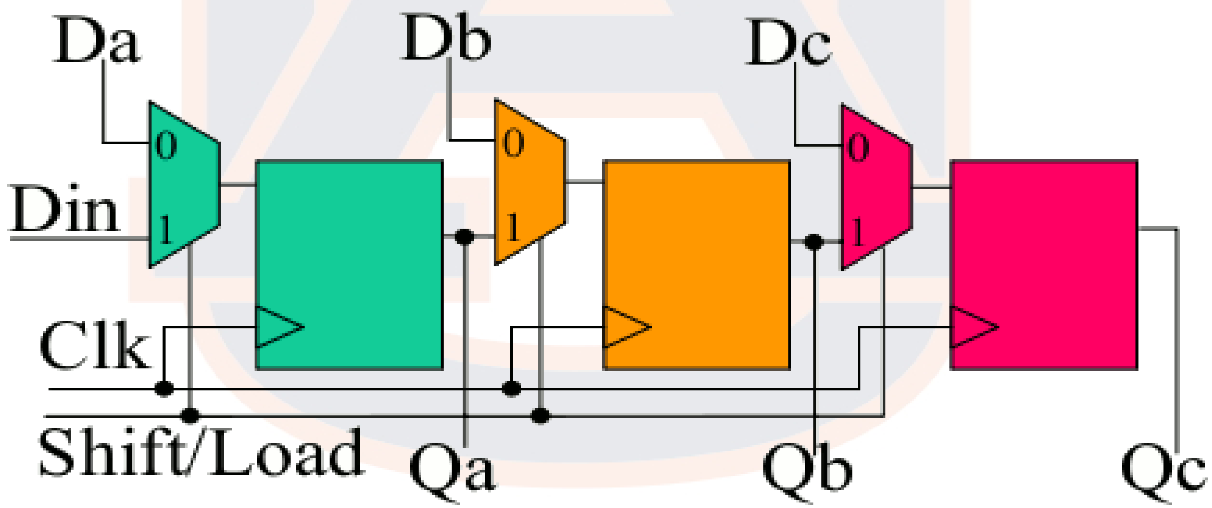
Shift Register

- A series of D flip-flops with outputs connected to the input of the next flip-flop
  - Serial-in, serial-out = data in on Din; data out on Qc
  - Serial-in, parallel-out = data in on Din; data out on Qa, Qb, Qc



Shift Register

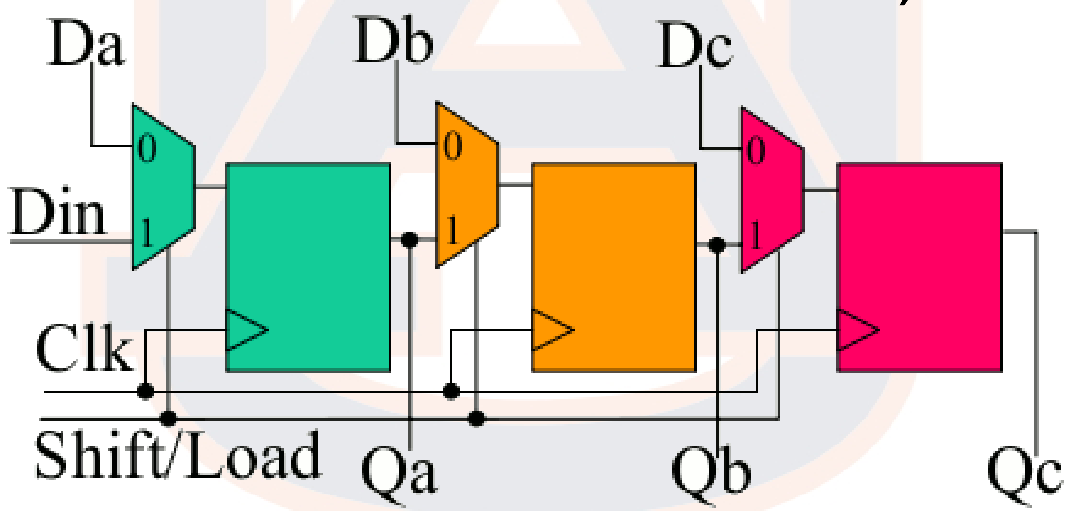
- Adding multiplexers to the inputs of the flip-flops allows data to be loaded in parallel
- Load/Shift signal line allows control over which operation is being performed



Shift Register

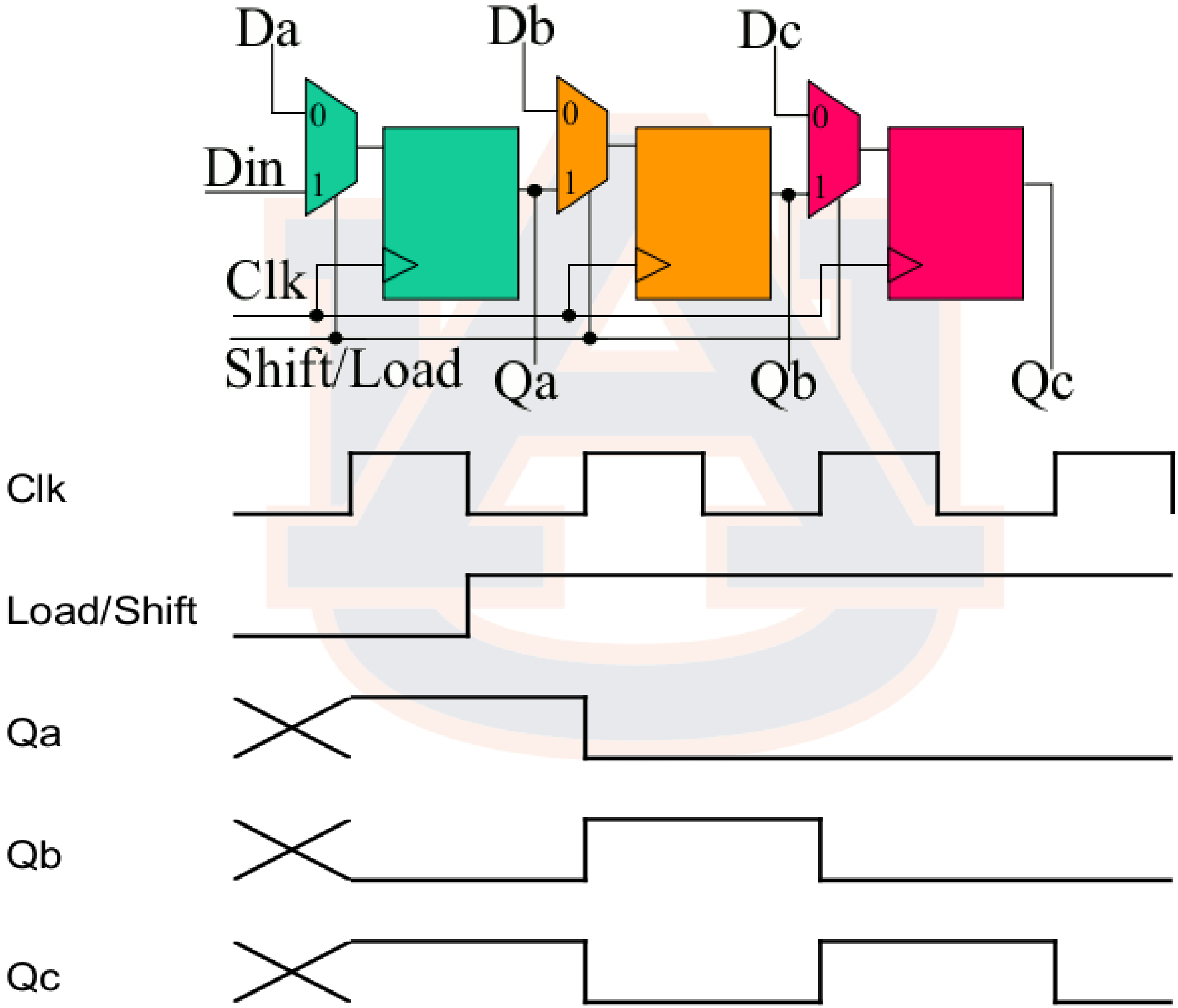
- Parallel-in, parallel-out = data in on Da, Db, Dc; data out on Qa, Qb, Qc ($\overline{\text{Load}}$/Shift=0)

- Parallel-in, serial-out = data in on Da, Db, Dc; data out on Qc ($\overline{\text{Load}}$/Shift=0, then $\overline{\text{Load}}$/Shift=1)



- Example, Assume:
    - Da,Db,Dc=1,0,1
    - Din=0

Shift Register



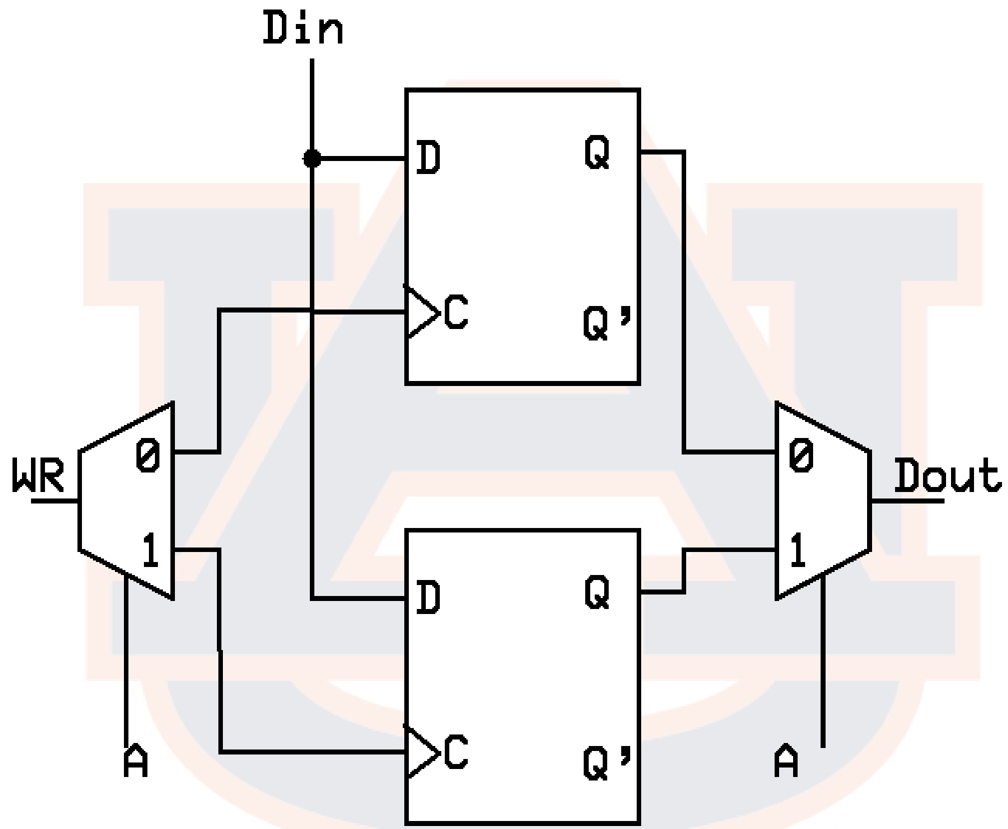
Serial Transmission

- The shift register forms the basis of serial communication
- Examples of serial communication:
  - PC serial port (RS-232) and its industrial counterparts RS-422, RS-485
  - Universal Serial Bus (USB)
  - Ethernet
  - Serial ATA
  - I^2C

Random Access Memory (RAM)

- Signals needed:
  - Data – what value being stored
  - Address – which element to store data in
  - Write Enable – when the other signals are valid, used to cause the element to remember the data value
- The simplest RAM example
  - 1 bit of data at each address
  - 2 addresses (1 bit address)

RAM Example



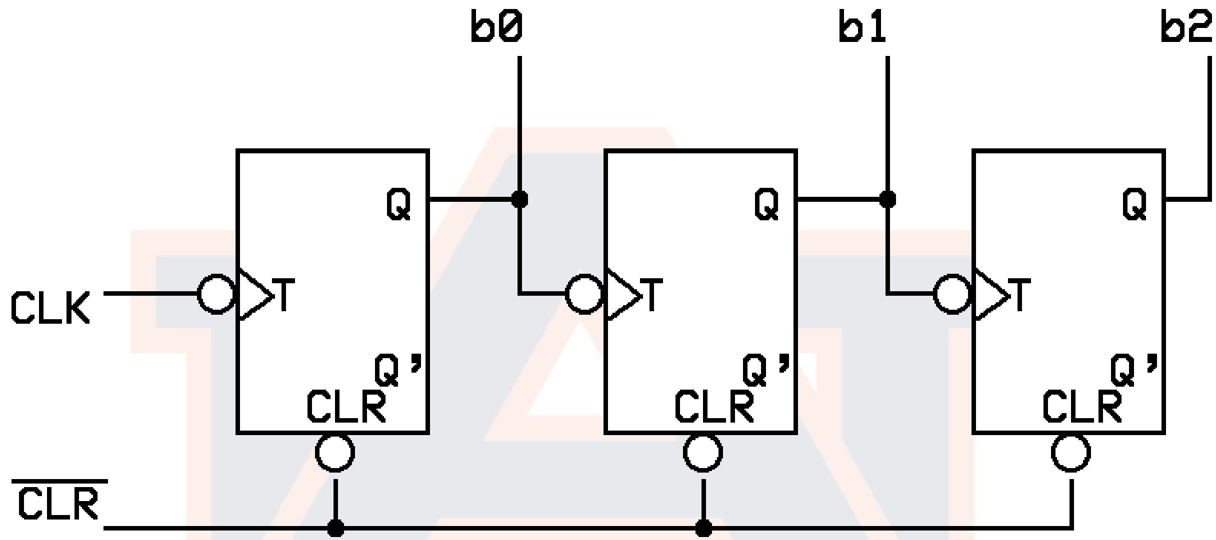
Counters

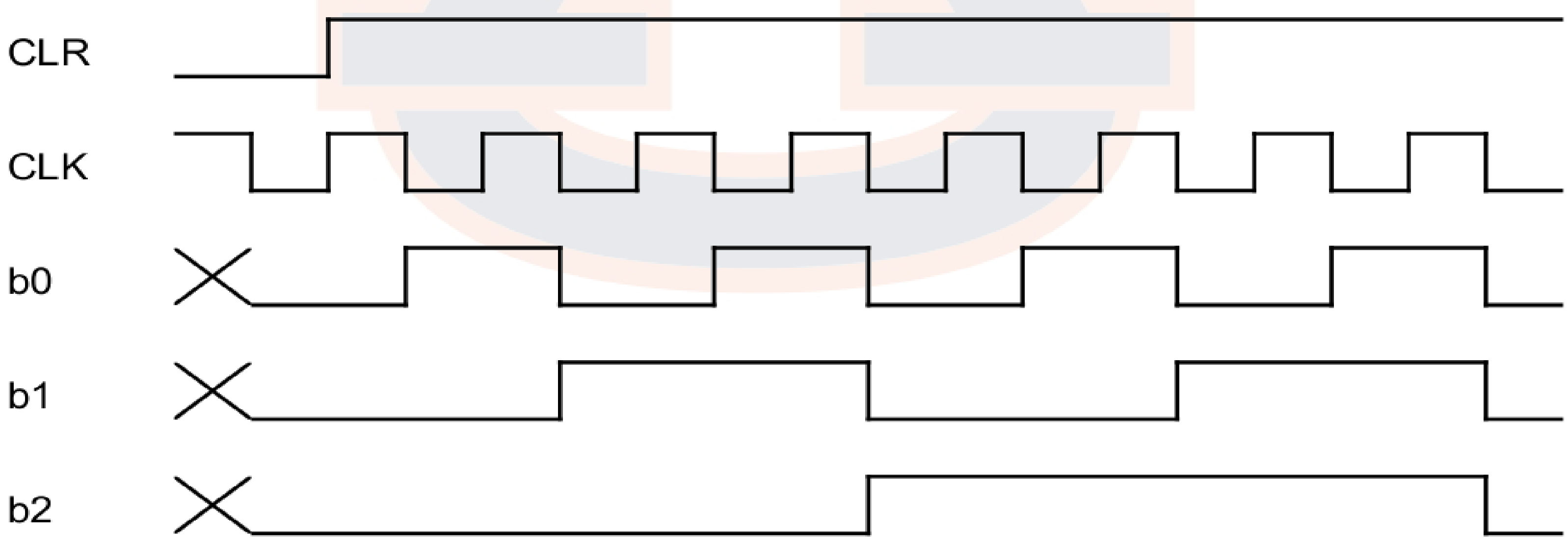
- Storing and transmitting numbers are both important tasks
- Numbers should have some meaning in the real-world
- Counters allow a series of pulses to be enumerated
- Some things the pulses can represent:
    - A unit of time (clock)
    - Distance (odometer)
    - Coins (vending machine)
    - Pharmaceuticals (pill counting)
    - Attendance (turnstile)
- Pulse can occur both regularly and irregularly in time

Asynchronous Counters

- Asynchronous counters are simple but slow
- Outputs don't change at the same time, each memory element (bit) is dependent on the output of the element before it
- As number of bits increase, so does worst case time for an update
    - Limits the frequency of pulses being counted
    - Can result in lost counts

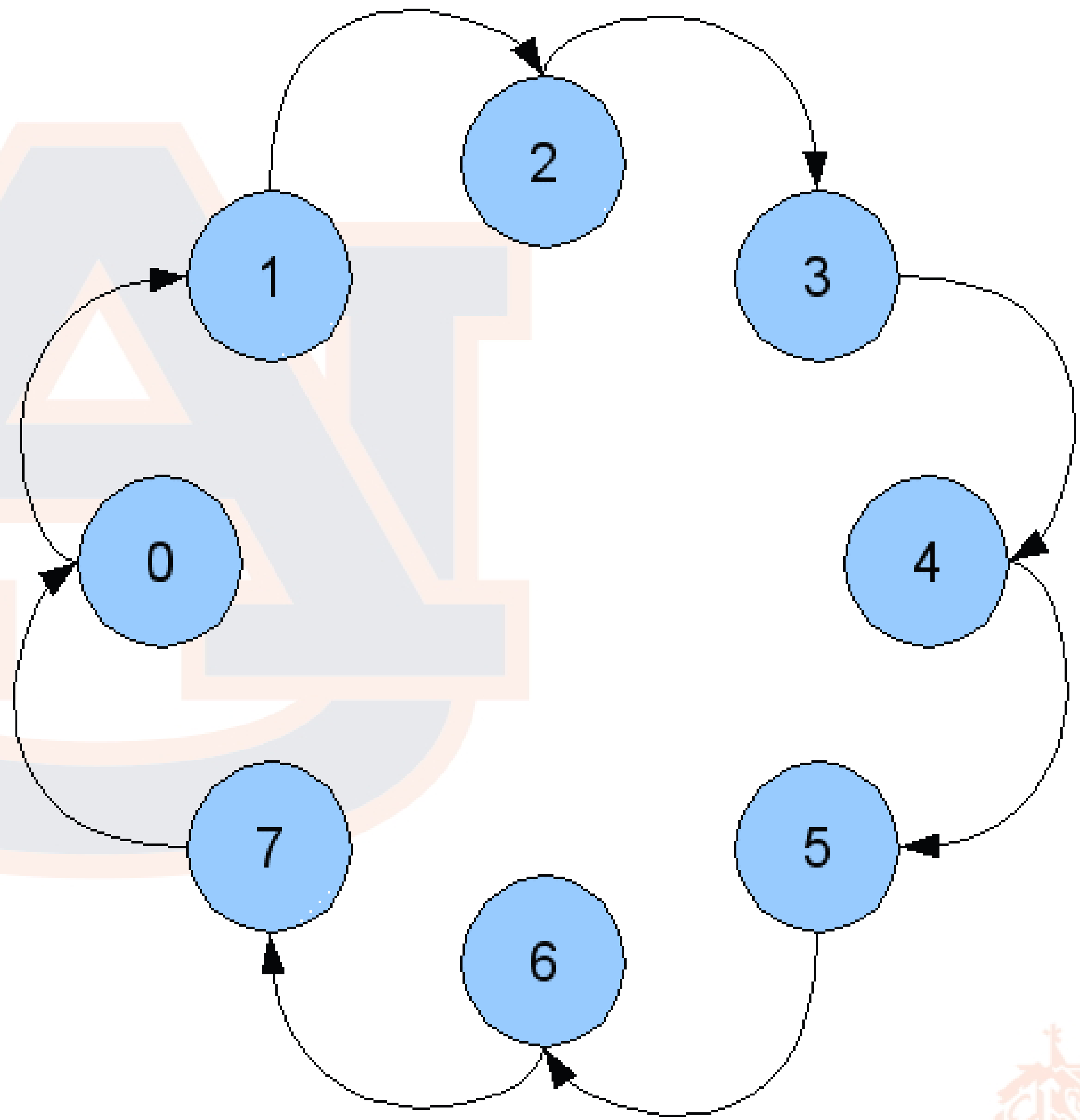
Asynchronous Ripple Counter





3-bit Counter State Diagram

- 3 bit state represented by decimal equivalent
- Negative edge of clock is only stimulus needed to advance state

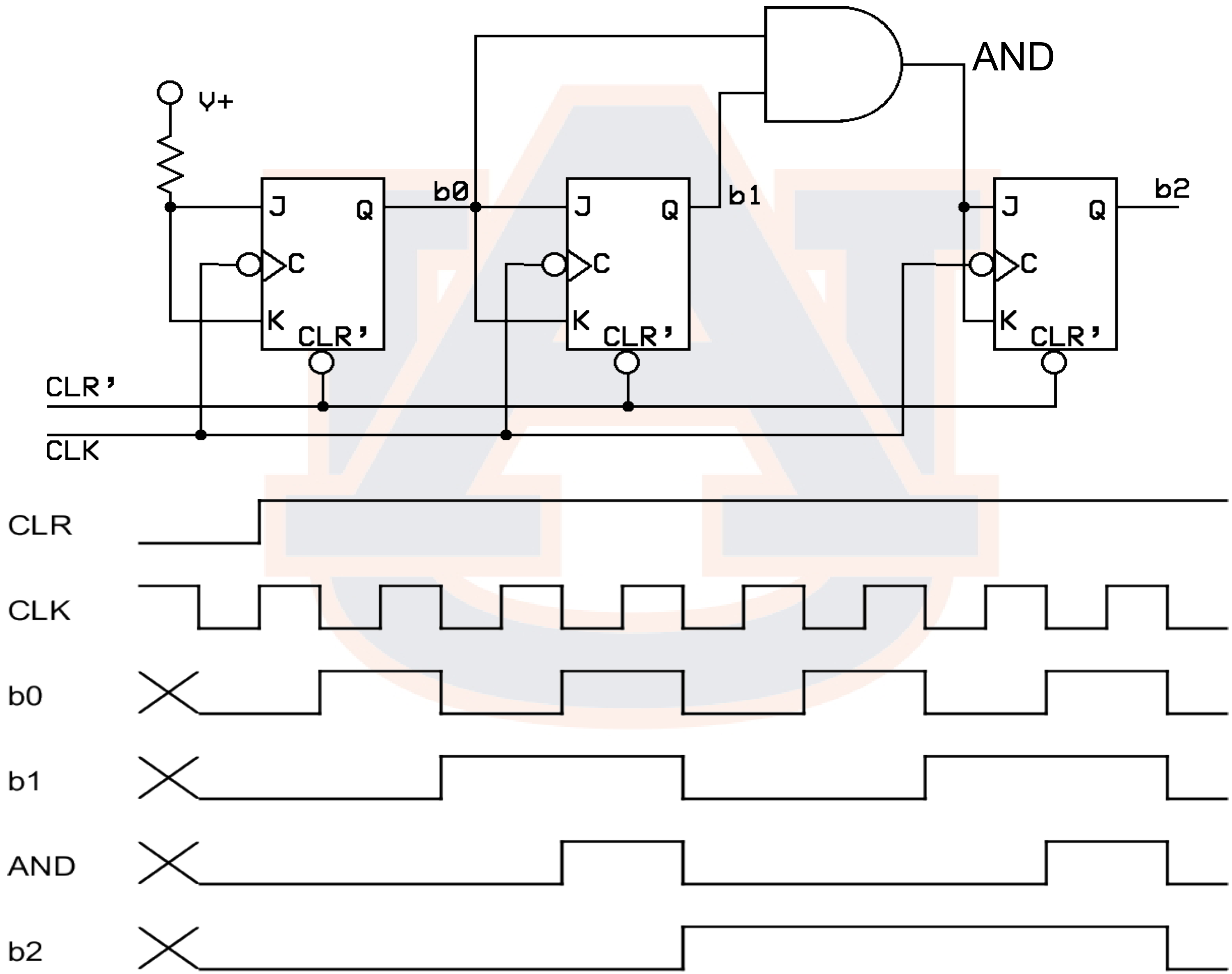


Synchronous Counter

- Built with either clocked TFFs or JK FFs
- All clock inputs tied together (each element changes state at the same time)
- Additional logic need to decide when to toggle a bit
- A bit should toggle if all bits of lesser significance are high

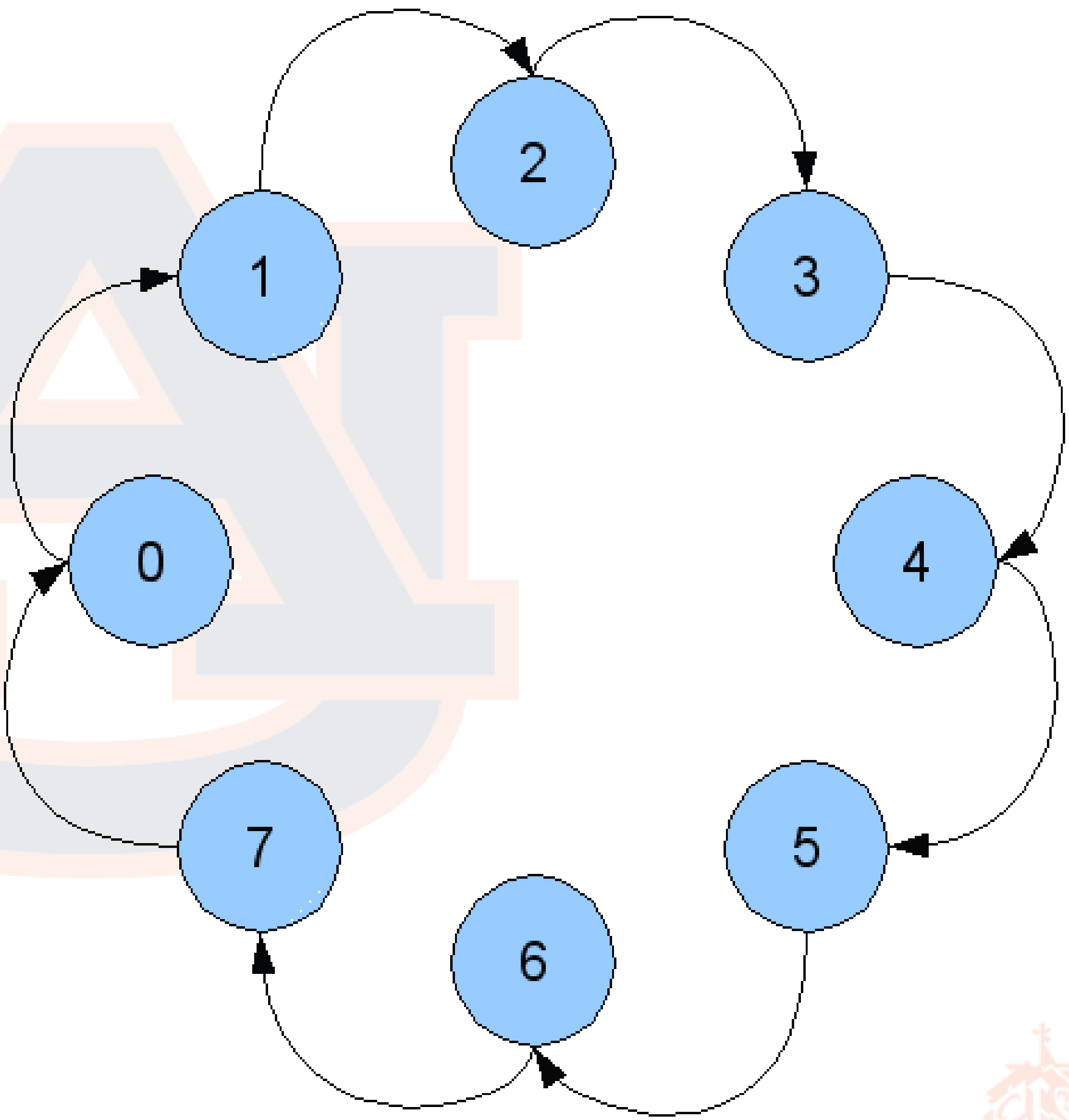
	b2	b1	b0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
0	0	0	0

Synchronous Counter



3-bit Counter State Diagram

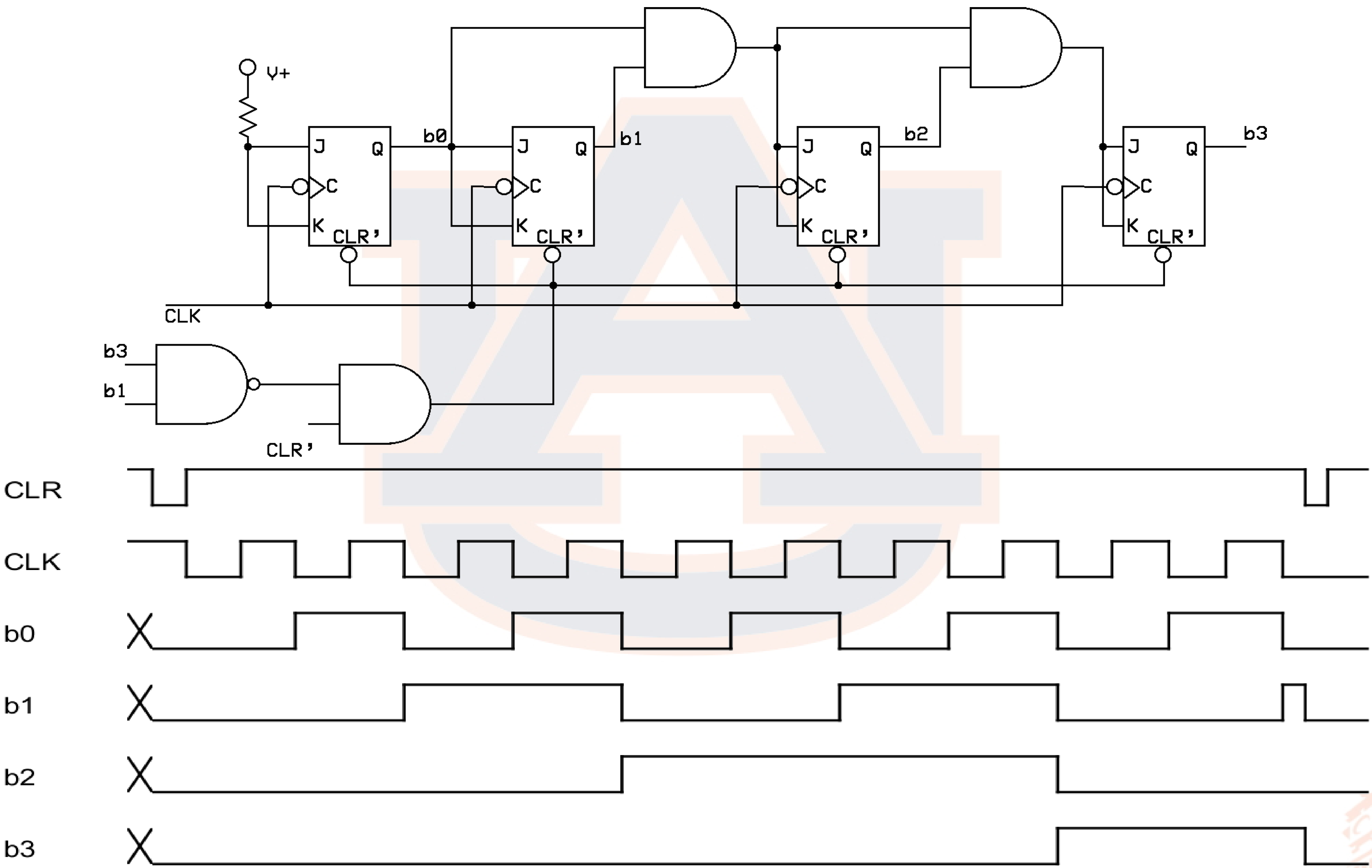
- What if I only want to count to 5?
- What if I need to count down?
- Can I count in a code (Gray code)?
- Why is France so far away?
- Some of these we'll answer today, some over the next week



BCD Counter

- We want to display a count in decimal
- 4 bits necessary to store digits 0,...,9
- We can detect when 10_{10} is present and reset the counter
- Clr' input is asynchronous in this example
- The time 10_{10} is present on the outputs is insignificant but not non-existent

BCD Counter



Modulo-A Counter

- This can be generalized to clear the counter when an arbitrary value A_0, A_1, A_2, …
- Recall the comparator circuit. (Equal-to or Not-equal-to)
- When the state of the flip-flops is equivalent to the value A, the state is reset to zero

Modulo-A Counter