Midterm 2 for COSC330/PHYS306 - Fall 2021

Dr. Jordan Hanson - Whittier College Dept. of Physics and Astronomy

December 1, 2021

1 Chapter 6 - Functions of Combinational Logic

1. Consider Fig. 1, in which a 4-bit adder is depicted. Assuming a uniform worst-case delay of 8 ns from carry-in to carry-out for each stage, the total delay is 32 ns. (a) At what maximum frequency can this circuit perform additions, if the delay must be less than a clock period? (b) What would the result in (a) be if the adder was extended to 8 bits? (c) Create a timing diagram showing the addition of the numbers in Fig. 1. **Bonus:** include the timing delays, and indicate the clock period.

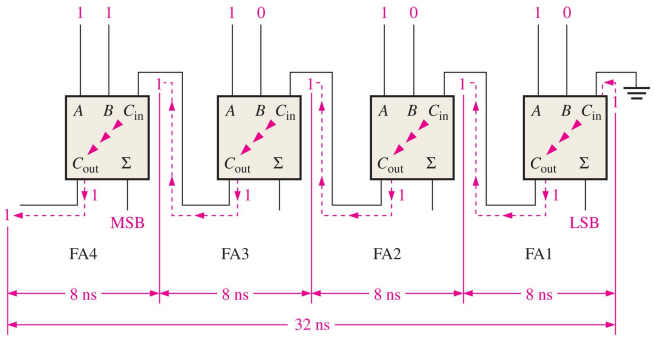

Figure 1: A schematic of a 4-bit adder with delays marked in nanoseconds.

- (a) If the total worst-case delay is 32 ns, then that implies a clock frequency of 1/32 GHz, or 31.25 MHz.
- (b) If the number of adder stages is doubled, then the maximum frequency scales down by a factor of two, to 15.625 MHz.
- (c) See below.

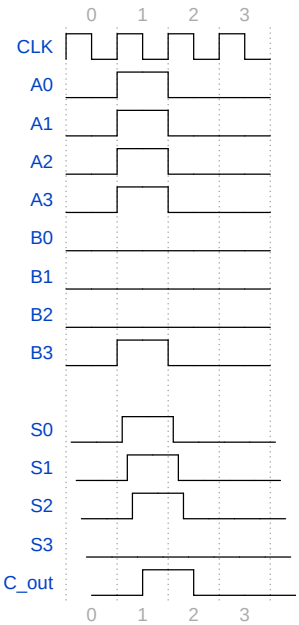

Figure 2: Solution to Exercise 1c (with simulated delays).

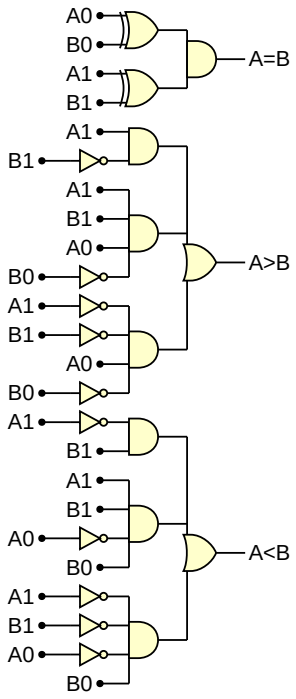

Figure 3: Solution to Exercise 2.

2. Using AND, OR, XNOR, and inverter gates, (a) create a 2-bit comparator that yields True if $A > B$, (b) True if $A = B$, and (c) True if $A < B$. For each circuit, assume both A and B are 2-bit binary positive numbers. (d) Wrap this all into one circuit with three outputs and 4 inputs. *Hint: use Karnaugh maps for parts (a)-(c) to simplify the gates. See Fig. 3.*

3. Generate the logic gates that convert 4-bit gray code to 4-bit binary code, and show that it works with a timing diagram. *See Fig. 4.*

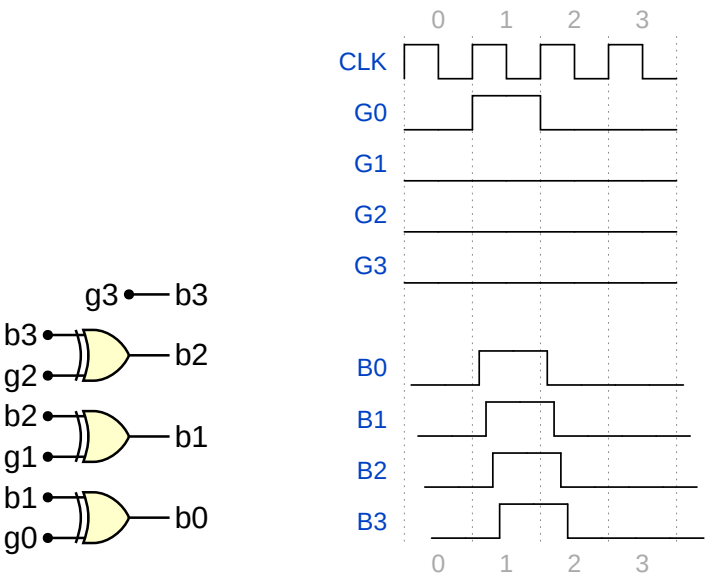

Figure 4: Solution to Exercise 3.

4. Consider Fig. 5, in which a decimal to binary conversion circuit is shown. (a) Add logic to the circuit such that it becomes a hexadecimal to binary converter. (b) Draw a logic symbol for this circuit with the correct number of inputs and outputs. (c) Connect two hex-to-bin converters to a symbolic 2-to-1 multiplexer with the correct number of data select line(s) to form a system that can send two-digit hex numbers over one output line.

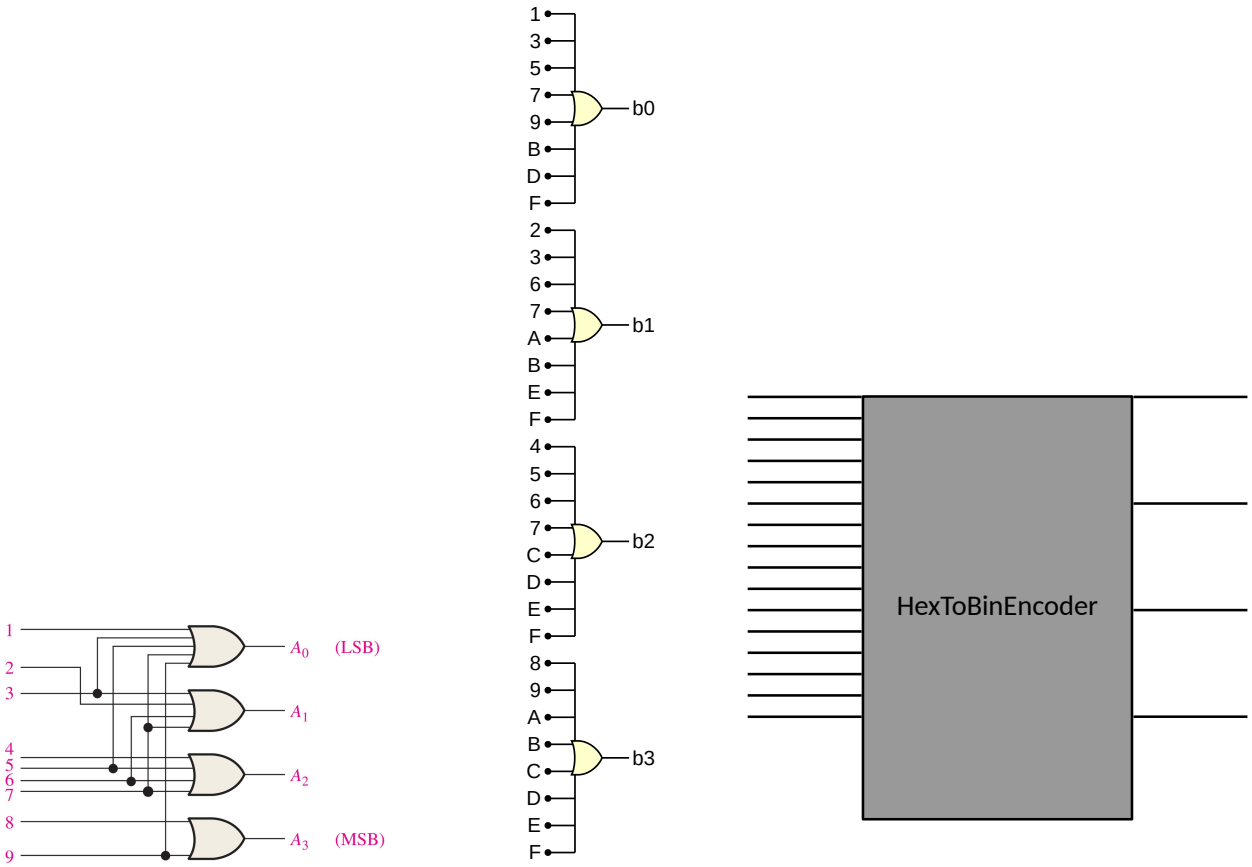

Figure 5: (Left) A decimal to binary encoder circuit. (Middle and Right) Solution to Exercise 4.

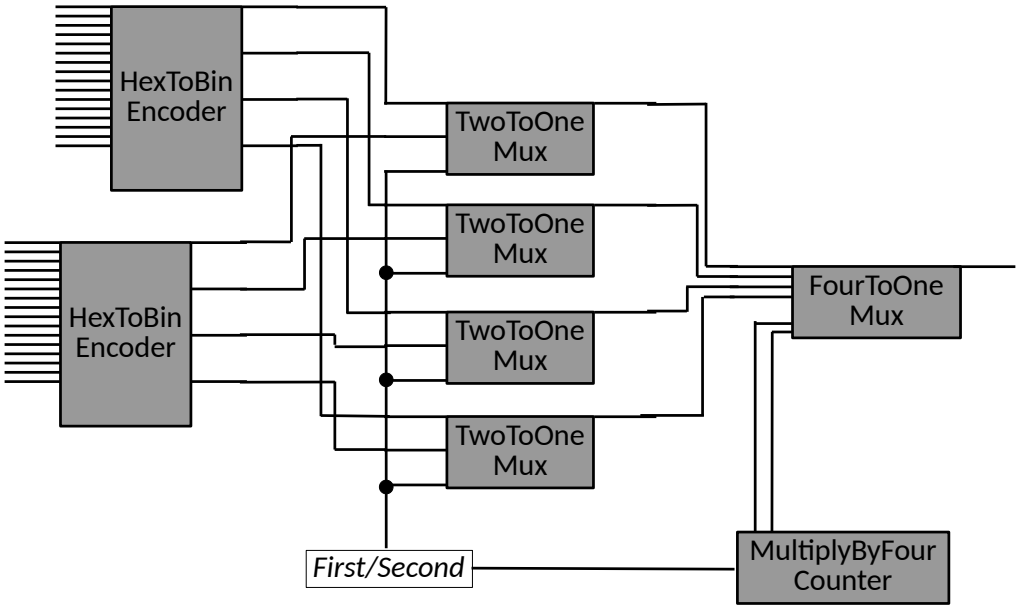

Figure 6: Solution to Exercise 4c. There are two hexadecimal to binary converters, each feeding into four 2-to-1 multiplexers, to switch between the upper number and the lower number. Each 2-to-1 multiplexer only needs one data-select line, controlled by the signal *First/Second*. If this signal is HIGH or LOW, the first or second number is selected, respectively. This signal should oscillate at some frequency, f. Finally, a counter that oscillates at $4f$ is needed to change the bits from 4-bit parallel to 4-bit serial structure. The 4 parallel-bits are fed to a 4-to-1 multiplexer, where they are converted to a serial signal.

2 Chapter 7 - Latches, Flip-flops, and Timers

1. Consider Fig. 7, in which a divide-by-four frequency divider is depicted with D flip-flops and a CLK signal. (a) Elaborate on this circuit to create a circuit that can divide the clock frequency by 2, 4, or 8. Show the flip-flops explicitly. (b) Develop a symbol for the 2-4-8-16 divider, with CLK signal input and four ouputs (one each for $f/2$, $f/4$, $f/8$, and $f/8$, where f is the clock frequency). (c) Connect the symbol for the new part to a symbolic 4-to-1 multiplexer with the appropriate number of data select lines to form a clock division system. (d) Show with a timing diagram the output if the user changes the data select lines at least once. For parts (b)-(d), it is only necessary to show symbols rather than individual flip-flops.

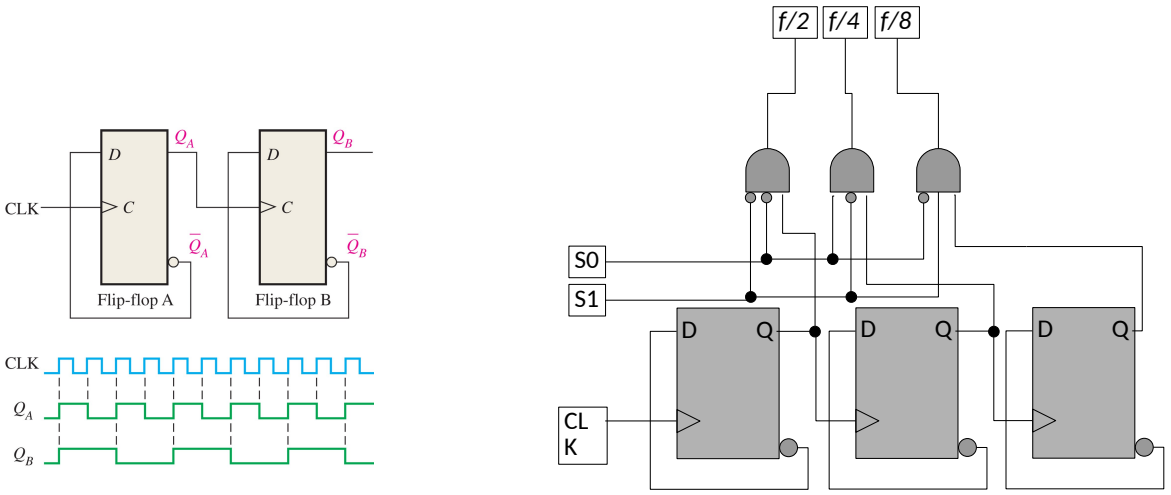

Figure 7: (Left) Two D flip-flops forming a frequency divider. (Right) Solution to Exercise 1 (a).

- (a) See Fig. 7 (right). The data-select lines enable the frequency outputs. These outputs could be fed into a final OR gate to limit the outputs to one line.
- (b) See Fig. 8 (left).
- (c) See Fig. 8 (right).
- (d) See Fig. 9.

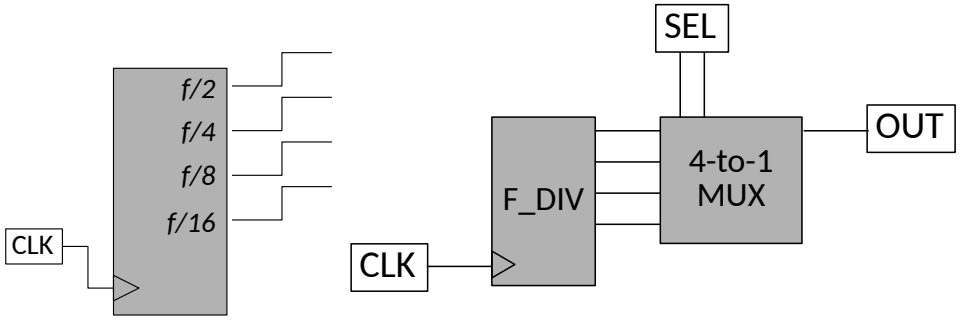

Figure 8: (Left) Solution to Exercise 1 (b). (Right) Solution to Exercise 1 (c).

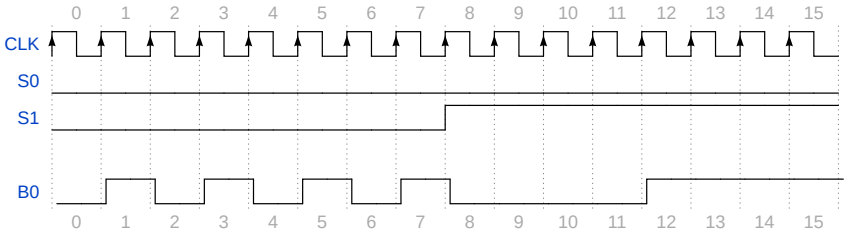

Figure 9: Solution to Exercise 1 (d). The user switches the data select signal at clock cycle 8, changing the frequency from 1/2 CLK to 1/8 CLK.

3 Chapters 8 and 9 - Shift Registers and Counters

1. Consider the timing diagram in Fig. 10. Using any combination of *shift registers* and supporting gates, create a circuit with 8-outputs that produces this timing diagram.
   Fig. 11 contains a timing diagram corresponding to Fig. 10 (right). Two bi-directional shift registers are OR'd

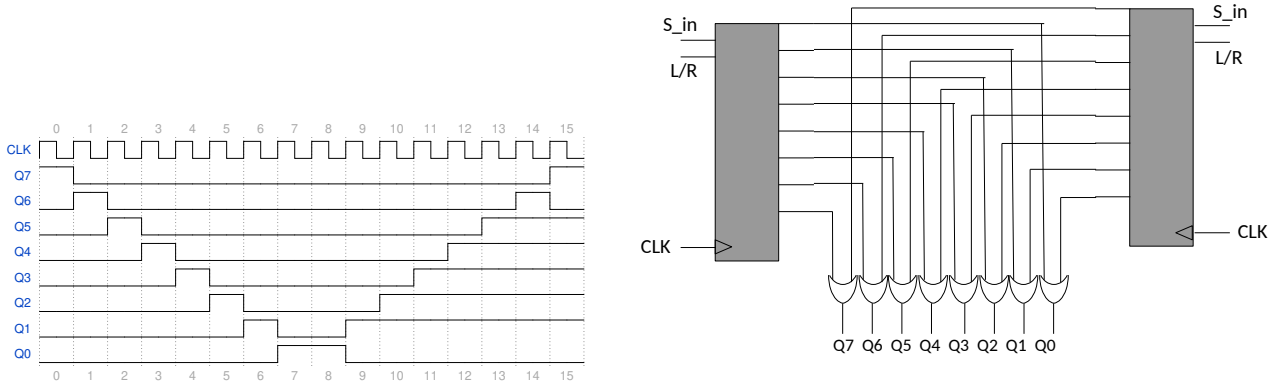

Figure 10: (Left) A waveform that repeats every 16 clock periods. (Right) One possible solution to Exercise 1.

in parallel, and initially are cleared. One bit is loaded into one shift-register and shifted across the OR array. At the right moment, that bit is lost and a new bit is added to the other register. That bit travels in the register in the opposite direction. A solution could be devised using only one bi-directional shift register, however the shifting would have to be "paused" for one clock cycle (cycles 7 and 8). That is, with one register, the timing would be off when the bit turns around. Alternatively, we could just OR one register with itself, and cleverly connect the outputs to avoid the timing issue.

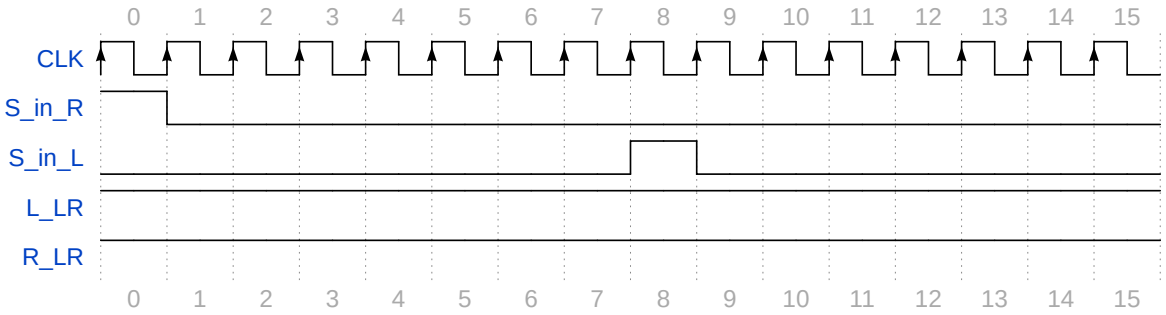

Figure 11: A timing diagram corresponding to Fig. 10 (right).

2. Consider Fig. 12, in which 4 and 5-bit Johnson counters are depicted. (a) Create a circuit based on Fig. 12 called a *combinatoric trigger*. Imagine four digital channels A, B, C, and D, as inputs. The output of the circuit should be True if *any two* of the four channels is True *within 100 ns of each other*. Develop any necessary logic symbols, and use the appropriate clock frequency. The Johnson counter(s) can have any number of bits. Fig.

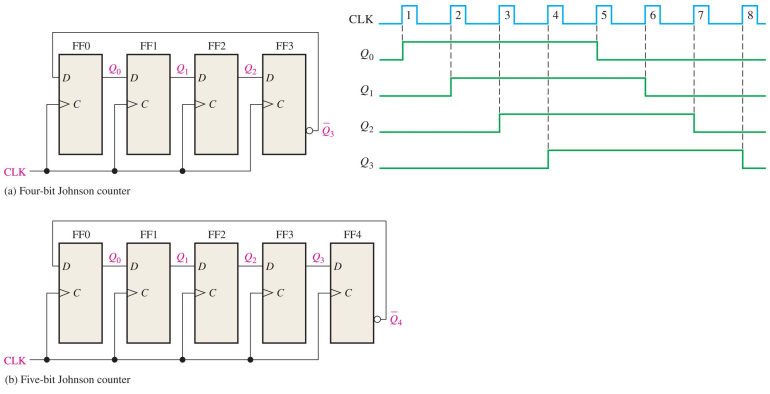

Figure 12: (Left) 4 and 5-bit Johnson counters. (Right) The timing diagram for the 4-bit case.

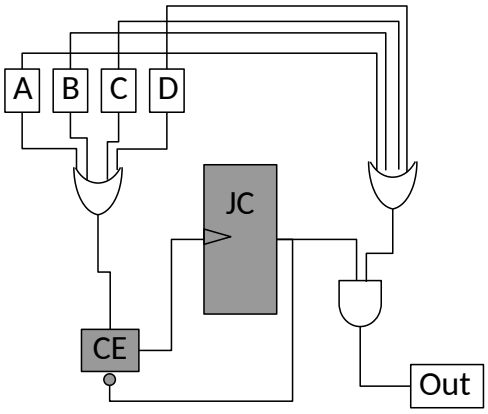

Figure 13: A solution to Exercise 2 based around the pulse output of a Johnson counter.

13 contains one possible design. The JC component stands for Johnson counter. Suppose it has 8 bits. If we clock it at 80 MHz, then the basic pulse output will have 100 ns width. We have $100/8 = 12.5$ ns per clock cycle, so $f = 1/12.5$ GHz, which is 0.08 GHz or 80 MHz. When any one of the four digital digital inputs goes high, it will trigger the Clock Enable (CE) via an OR gate. The clock enabler, or clock generator, is a counter that turns on at some frequency (80 MHz in this case) when enabled, and switches off when disabled. The digital inputs A-D are OR'd with the Johnson counter output. This means that if another digital signal goes high when the Johnson counter is high, then the AND gate will raise the output high. Whenever the Johnson counter recycles, the output will go low. The active low disable on the clock enable (CE) will shut off the clock, and stop the Johnson counter. Any new incoming signal A-D will override this and restart the Johnson counter. An example timing diagram is given in Fig. 14 below.

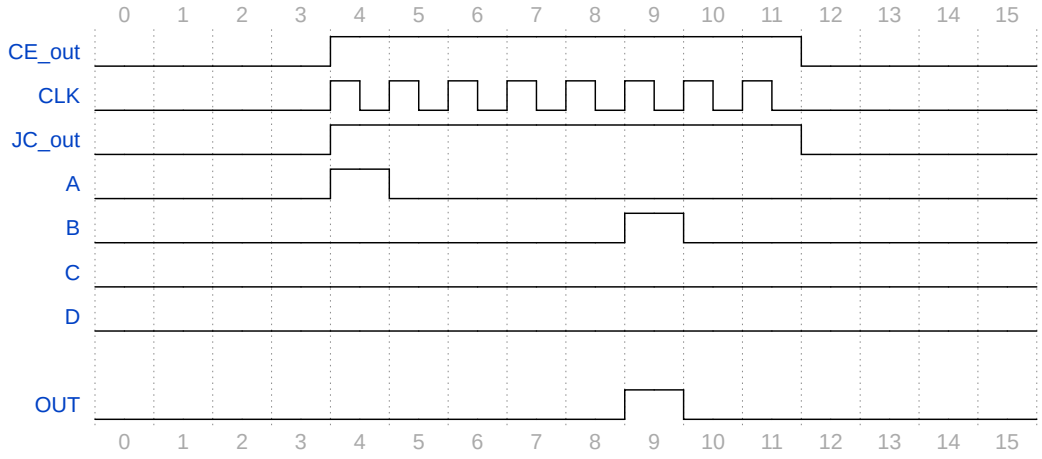

Figure 14: A timing diagram corresponding to Fig. 13.