Gate, Truth Tables, and Logic Equations
- the electronics inside a modern computer are digital
- a binary system matches the underlying abstraction inherent in the electronics
  - asserted, logically true or 1
  - deasserted, logically false or 0
- combinational: blocks without memory, depends only on the current input
- sequential logic: blocks with memory, depend on both the inputs and the value stored in memory
- truth table: defines the values of the outputs for each possible set of input values; for a logic block with \( n \) inputs, there are \( 2^n \) entries in the truth table; truth tables can completely describe any combinational logic function; we can use shorthand of specifying only the truth table entries for the nonzero outputs
- Boolean algebra: express logic function with logic equations using 3 operations:
  - the OR (or logical sum) operator +: the result of an OR operator is 1 if either of the variables is 1
  - the AND (or logical product) operator •: the result of an AND operator is 1 only if both inputs are 1
  - the unary operator NOT (or inversion or negation) \( \bar{A} \) or \( A\# \): the result of a NOT operator is 1 only if the input is 0
- laws for Boolean algebra
  - identity law: \( A + 0 = A \) and \( A \cdot 1 = A \)
  - zero and one laws: \( A + 1 = 1 \) and \( A \cdot 0 = 0 \)
  - inverse laws: \( A + A\# = 1 \) and \( A \cdot A\# = 0 \)
  - commutative laws: \( A + B = B + A \) and \( A \cdot B = B \cdot A \)
  - association laws: \( A + (B + C) = (A + B) + C \) and \( A \cdot (B \cdot C) = (A \cdot B) \cdot C \)
  - distributive laws: \( A \cdot (B + C) = (A \cdot B) + (A \cdot C) \) and \( A + (B \cdot C) = (A + B) \cdot (A + C) \)
  - gate: the basic building block of logic; any logical function can be constructed using AND gate, OR gate, and NOT gate (or inverter, or using bubbles); in fact, all logic functions can be constructed with only a single universal gate type, if that gate is inverting; the two common inverting gates are called NOR and NAND gates

Combinational Logic
- a decoder has an \( n \)-bit input and \( 2^n \) outputs, where only one output is asserted for each input combination; a decoder translates the \( n \)-bit input into a signal that corresponds to the binary value of the \( n \)-bit input; an encoder performs the inverse function of a decoder, taking \( 2^n \) inputs and producing an \( n \)-bit output
- a multiplexor or selector: the output is one of the inputs that is selected by a control; if there are \( n \) data inputs, there will need to be \([\lg n]\) selector inputs; a multiplexor consists of 3 parts:
  - a decoder that generates \( n \) signals, each indicating a different input value
  - an array of \( n \) AND gates, each combining one of the inputs with a signal form the decoder
  - a single large OR gate that incorporates the outputs of the AND gates
- a two-level logic and PLAs
  - two-level representation: any logic function can be written in a canonical form, where every input is either a true or complemented variable and there are only two levels of gates – one being AND and the other OR – with a possible inversion on the final output
    - product of sums: a logical product (AND) of sums (OR)
    - sum of products (more common) or programmable logic array (PLA): a logical sum of products (or minterms)
  - each truth table entry for which the function is true corresponds to a product term; we can use this relationship between a truth table and a two-level representation to generate a gate-level implementation of any set of logic functions
  - a PLA can directly implemented the truth table of a set of logic functions with multiple inputs and outputs; the total size of a PLA is equal to the sum of the size of the AND gate array (the AND plane) and the size of the OR gate array (the OR plane); the AND gate array is equal to the number of inputs is the number of outputs times the number of product terms
  - to efficiently implement a set of logic functions, first, only the truth table entries that produce a true value for at least one output have any logic gates associated with them; second, each different product will have only one entry in the PLA, even if the product term is used in multiple outputs
  - the contents of a PLA are fixed when the PLA is created, but PALs can be programmed electronically when a designer is ready to use them
- read-only memory (ROM)
  - a ROM’s contents are fixed, usually at the time the ROM is created
  - a programmable ROM (or PROM) can be programmed electronically
  - an erasable PROM (or EPROM) needs a slow erasure process using ultraviolet light
  - a ROM has \( n \) input address lines and \( 2^n \) (the height) of addressable entries, the number of bits (the width) in each addressable entry is
Clocks

- a clock is simply a free-running signal with a fixed cycle time (or clock period); the clock frequency is imply the inverse of the cycle time
- edge-triggered clocking: all state changes occur on a clock edge, either the rising edge or the falling edge of the clock is active
- a synchronous system: signals that are written into state elements must be valid when the active clock edge occurs; a signal is valid if it is stable (i.e., not changing) and the value will not change again until the inputs change;
- combinational circuits cannot have feedback, if the inputs to a combinational logic unit are not changed, the outputs will eventually become valid
- in sequential circuits, the inputs to a combinational logic block come from a state elements, and the output are written into another or the same state element; outputs changes only on the clock edge; to ensure that the values written into the state elements on the active clock edge are valid, the clock must have a long enough period so that all the signals in the combinational logic block stabilize; this constraints sets a lower bound on the length of the clock period

Memory Elements

All memory elements store state: the output from any memory element depends both on the inputs and on the value that has been stored inside the memory element

- flip-flops and latches
  - the simplest type of memory elements are unclocked latches; a set-reset (S-R) latch built from a pair of NOR gates; if S is asserted then the output Q will be asserted and Q# will be deasserted; if R is asserted, then the output Q# will be asserted and Q will be deasserted; when both S and R are both asserted the last values of Q and Q# will continue to be stored in the cross-coupled structure; asserting S and R simultaneously can lead to incorrect operation: depend on how S and R are deasserted, the latch may oscillate or become metastable
  - a clocked D latch (or transparent latch) has two inputs (the data value to be stored D, and a clock signal C), and two complemented output Q and Q#; when C is asserted, the latch is open, and the value of Q becomes the value of D; when C is deasserted, the latch is closed, and the value of Q is whatever value was stored the last time the latch was open; the value of D must be stable when C changes
  - a D flip-flop is not transparent, its outputs change only on the (either rising or falling) clock edge; it is constructed from a pair of D latches: the first one is the master and the second one slave; the minimum time that the input must be valid before the clock edge is the set-up time, and the minimum time during which it must be valid after the clock edge is the hold time (usually 0 or very small); failure to meet timing requirement can result in a situation where the output of the flip-flop may not even be predictable
- register files
  - a register file consists of a set of registers that can be read and written by supplying a register number to be accessed
  - a register file can be implemented with a decoder for each read or write port and an array of registers built from D flip-flops
  - the read ports can be implemented with a pair of multiplexors, each of which is as wide as the number of bits in the register file; the register read number signal is used as the multiplexor selector signal
  - the write ports can be implemented by using a decoder to generate a signal that can be used to determine which register to write
  - if the same register is read and written during the same clock cycle, the write of the register file occurs on the clock edge, the register will be valid during the time it is read
- static random access memories (SRAMs)
  - larger amounts of memory are built using either SRAMs or DRAMs
  - in a SRAM the value stored in a cell is kept on a pair of inverting gates (which require 4 to 6 transistors per bit), and as long as the power is applied, the value can be kept indefinitely

The ROM is fully decoded: it contains a full output word for every possible input combination; a PLA is only partially decoded; this makes PLAs generally more efficient for implementing combinational logic functions; ROMs have the advantage of being able to implement any logic function with the matching number of inputs and outputs, make it easier to change the ROM contents if the logic function changes, since the size of the ROM need not change

- don’t cares
  - don’t cares make it easier to optimize the implementation of a logic function
  - output don’t cares arise when we don’t care about the value of an output for some input combination
  - input don’t cares arise when a output depends on only some of the inputs
- Karnaugh maps
- arrays of logic elements
- bus: a collection of data lines that is treated together as a single logical signal, a shared collection of lines with multiple sources and users

equal to the number of output bits; the total number of bits (or shape) the ROM is equal to the height times the width

- a ROM can encode a collection of logic functions directly from the truth table: if there are n functions with m inputs, we need a ROM with m addressable lines, with each entry being an bits wide; the entries in the input portion of the truth table represent the addresses of the entries in the ROM, while the contents of the output portion of the truth table constitute the contents of the ROM
- a ROM is fully decoded: it contains a full output word for every possible input combination; a PLA is only partially decoded; this makes PLAs generally more efficient for implementing combinational logic functions; ROMs have the advantage of being able to implement any logic function with the matching number of inputs and outputs, make it easier to change the ROM contents if the logic function changes, since the size of the ROM need not change

- don’t cares
  - don’t cares make it easier to optimize the implementation of a logic function
  - output don’t cares arise when we don’t care about the value of an output for some input combination
  - input don’t cares arise when a output depends on only some of the inputs
- Karnaugh maps
- arrays of logic elements
- bus: a collection of data lines that is treated together as a single logical signal, a shared collection of lines with multiple sources and users

Clocks

- a clock is simply a free-running signal with a fixed cycle time (or clock period); the clock frequency is imply the inverse of the cycle time
- edge-triggered clocking: all state changes occur on a clock edge, either the rising edge or the falling edge of the clock is active
- a synchronous system: signals that are written into state elements must be valid when the active clock edge occurs; a signal is valid if it is stable (i.e., not changing) and the value will not change again until the inputs change;
- combinational circuits cannot have feedback, if the inputs to a combinational logic unit are not changed, the outputs will eventually become valid
- in sequential circuits, the inputs to a combinational logic block come from a state elements, and the output are written into another or the same state element; outputs changes only on the clock edge; to ensure that the values written into the state elements on the active clock edge are valid, the clock must have a long enough period so that all the signals in the combinational logic block stabilize; this constraints sets a lower bound on the length of the clock period

Memory Elements

All memory elements store state: the output from any memory element depends both on the inputs and on the value that has been stored inside the memory element

- flip-flops and latches
  - the simplest type of memory elements are unclocked latches; a set-reset (S-R) latch built from a pair of NOR gates; if S is asserted then the output Q will be asserted and Q# will be deasserted; if R is asserted, then the output Q# will be asserted and Q will be deasserted; when both S and R are both asserted the last values of Q and Q# will continue to be stored in the cross-coupled structure; asserting S and R simultaneously can lead to incorrect operation: depend on how S and R are deasserted, the latch may oscillate or become metastable
  - a clocked D latch (or transparent latch) has two inputs (the data value to be stored D, and a clock signal C), and two complemented output Q and Q#; when C is asserted, the latch is open, and the value of Q becomes the value of D; when C is deasserted, the latch is closed, and the value of Q is whatever value was stored the last time the latch was open; the value of D must be stable when C changes
  - a D flip-flop is not transparent, its outputs change only on the (either rising or falling) clock edge; it is constructed from a pair of D latches: the first one is the master and the second one slave; the minimum time that the input must be valid before the clock edge is the set-up time, and the minimum time during which it must be valid after the clock edge is the hold time (usually 0 or very small); failure to meet timing requirement can result in a situation where the output of the flip-flop may not even be predictable
- register files
  - a register file consists of a set of registers that can be read and written by supplying a register number to be accessed
  - a register file can be implemented with a decoder for each read or write port and an array of registers built from D flip-flops
  - the read ports can be implemented with a pair of multiplexors, each of which is as wide as the number of bits in the register file; the register read number signal is used as the multiplexor selector signal
  - the write ports can be implemented by using a decoder to generate a signal that can be used to determine which register to write
  - if the same register is read and written during the same clock cycle, the write of the register file occurs on the clock edge, the register will be valid during the time it is read
- static random access memories (SRAMs)
  - larger amounts of memory are built using either SRAMs or DRAMs
  - in a SRAM the value stored in a cell is kept on a pair of inverting gates (which require 4 to 6 transistors per bit), and as long as the power is applied, the value can be kept indefinitely
configuration of an SRAM: an SRAM has \( n \) input address lines and \( 2^n \) (the height) of addressable locations, the number of bits in each addressable location is the width, which is usually narrow, and the data input and data output have the same width

- the Chip select signal must be active to initiate read or write access; the Output enable signal for read, and the Write enable for write
- SRAMs have a fixed access time; the SRAM read access time is the delay from the time that Output enable is true and the address lines are valid until the time that the data is on the output lines; the time to complete a write is specified by the contribution of the set-up times, the hold times, and the Write enable pulse width
- rather than using a giant multiplexor, large memories are implemented with a shared output line (the bit line), which multiple memory cells in the memory array can assert; to allow multiple sources to drive a single line, a tri-state buffer is used; the output of a tri-state buffer is equal to the asserted or deasserted input signal if the Output enable is asserted, and is otherwise in a high-impedance state that allows another tri-state buffer to determine the value of the shared output; it is critical that the Output enable of at most one the tri-state buffers be asserted
- it still requires a very large decoder and a correspondingly large number of word lines or address line; to circumvent this problem, large memories are organized as rectangular arrays and use a two-step decoding process
- the key capability provided by synchronous RAMs (either synchronous SRAMs or SSARAMs, or synchronous DRAMS or SDRAMs) is the ability to transfer a burst of data from a series of sequential addresses within an array or row, defined by a starting address and a burst length
- dynamic random access memories (DRAMs)
- in a DRAM, the value kept in a cell is stored as a charge in a capacitor, so it cannot be kept indefinitely and must periodically be refreshed; because DRAMs use only a single transistor per bit of storage, they are much denser and cheaper per bit
- the charge can be kept for several milliseconds, we can simply read its contents and write it back to refresh the cell; most large DRAMs use two-level decoding structure, and refresh an entire row (which shares a word line) with a read cycle followed immediately by a write cycle; typically, refresh operations consume 1% to 2% of the active cycle of the DRAMs
- the transistor inside the cell is a switch, called a pass transistor, the allows the value stored on the capacitor to be accessed for either reading or writing; when the signal on the word line is asserted, the switch is closed, connecting the capacitor to the bit line
- if the operation is write, then the value to be written is placed on the bit line; if the value is a 1, the capacitor will be charged; if the value is a 0, then the capacitor will be discharged
- when read, the DRAM must detect a very small charge stored in the capacitor; before activating the word line for a read, the bit line is charged to the voltage that is halfway between the low and high voltage; then, by activating the word line, the charge on the capacitor is read out onto the bit line; this causes the bit line to move slightly toward the high or low direction, and this change is detected with a sense amplifier, which can detect small changes in voltage
- DRAMs use a two-level decoder, consisting of a row access, followed by a column access
- the row access chooses one of a number of rows and activates the corresponding word line; the contents of all the columns in the active row are then stored in a set of latches; the column access then selects the data from the column latches
- to save pins and reduce the package cost, the same address lines are used for both the row and the column address
- error correction

**Finite State Machines**

**Timing Methodologies**
- level-sensitive timing
- asynchronous inputs and synchronizers