Digital Systems Design and Laboratory Midterm

2017-02-23

- A **hardware description language (HDL)** is a specialized computer language used to describe the structure and behavior of digital logic circuits.
- One important difference between most programming languages and HDLs is that *HDLs explicitly include the notion of time*.
- **Logic synthesis** is the process of turning an abstract form of desired circuit behavior into an implementation of logic gates.
- **Forward design** reads a *finite-state machine* or a *finite automata* and generates the digital logic circuit.
- **Reverse engineering** reads a digital logic circuit and explains its function by a program.
- A **Moore machine** is a finite-state machine whose output values are determined solely by its current state.
- A **Mealy machine** is a finite-state machine whose output values are determined both by its current state and by the current inputs.

2017-03-02

- Two common types of circuit implementation are (1) **breadboards** or **field programmable gate arrays (FPGAs)**, and (2) **application specific integrated circuits (ASICs)**.
- A **logic analyzer** is designed to display and evaluate digital signals, and allows engineers to design, optimize, and debug the hardware in digital systems.
- The design goals for a product includes *correctness*, *acceptable cost*, and *fair performance*.
- The cost of a product can be approximated by $C_d/N + C_m + C_s$, where C_d/N is design cost shared by N products, C_m is manufacture cost, and C_s is support cost, including maintenance and customer service.
- **Combinational logic design** is the design and implementation of **logic functions** whose outputs depend solely on their inputs.
- A **logic funciton** takes in one or more Boolean inputs and gives a Boolean output.
    - Suppose n inputs are provided, there are 2^n input combinations, and 2^{2^n} possible logic functions in total.

    - A logic function may not care certain input combinations, or so-called "don't-cares", and give a random output (0 or 1).
- Basic **logical operators** include NOT ('), AND (·), NAND, OR (+), NOR, XOR, XNOR, etc.
- Axioms and theorems of **Boolean algebra** includes identity, annulment, idempotency, involution, complementarity, associativity (結合律), commutativity (交換律), distributivity (分配律), uniting, absorption, factoring, consensus, **DeMorgan's law**, and **duality**.
- Both NAND and NOR gates are so-called *universal gates*. Every logic function can be built with either NAND or NOR logic gates alone.
    - NOT is just a NAND or NOR with both inputs tied together.
    - NAND and NOR are duals, i.e. one can be implemented by using the other.
    - NOR is equivalent to AND with inputs complemented.
    - NAND is equivalent to OR with inputs complemented.
- A logic function can be written down in **canonical form**, or visualized in the form of (1) a combination of logic gates, or (2) a series of waveform propagation. Only *waveform view* is able to show **signal propagation delay**, and **glitches/hazards**.
- A logic function can have *many* gate realizations. The one with fewer levels of gates has several benefits: (1) smaller chip, (2) less signal propagation delay (better efficiency), (3) less error-prone, (4) more power-saving, (5) lower cost of production.
- **Canonical form** of a logic function:
    - A unique signature of a Boolean function.
    - Can be in the form of a **truth table**, **sum-of-products**, or **product-of-sums**.
- **Sum-of-products** canonical form = **Disjunctive normal form** = **Minterm expansion**:
    - Input combination for which output is true.
    - *NOT* a minimal form.
    - Denoted as $\sum m(X_1) + d(X_2)$, where $X_1, X_2 \subseteq S$; $X_1 \bigcap X_2 = \emptyset$, where X_1 is called **minterms**, and X_2 is called **don't-cares**.
- **Product-of-sums** canonical form = **Conjunctive normal form** = **Maxterm expansion**:
    - Input combination for which output is false.
    - *NOT* a minimal form.
    - Denoted as $\prod M(X_1) \cdot D(X_2)$, where $X_1, X_2 \subseteq S$; $X_1 \bigcap X_2 = \emptyset$, where X_1 is called **maxterms**, and X_2 is called **don't-cares**.
- **Logic optimization**
    - The process of finding an equivalent representation of the specified logic circuit under specified constraints, e.g. minimum chip, fewest levels of gates, etc.

    - Some of the optimization techniques are **Boolean cubes**, **Karnaugh maps**, and **Quine-McCluskey procedure**.
- **Boolean cube** is a way of visualizing when the uniting theorem can be applied.
    - n input variables = n-dimensional "cube"
- **Karnaugh map** can be seen as the flat map of a Boolean cube.
    - Hard to draw or visualize for more than 4 dimensions.
    - Numbering scheme based on **Gray-code**.
    - Only a single bit changes in code for adjacent map cells.

2017-03-09

- **Quine-McCluskey procedure** is a tabular procedure used to simplify logic functions of multiple variables.
    - *Step1*: All **implicants** (*both minterms and don't-cares*) are arranged in the increasing order of the number of 1s.
    - *Step2*: Combining implicants who vary by only a single digit changing (*Hamming distance = 1)*. That digit can be replaced with a "-" indicating that *the digit doesn't matter*.
    - *Step3*: Repeat *Step2* until no more implicants can be combined. The remaining implicants are called **prime implicants (PIs)**.
    - *Step4*: A **PI chart** is constructed, in which *each minterm occupies a column, and each PI occupies a row*. Note that only minterms form a column. Don't-cares are ignored.
    - *Step5*: Selecting a minimum subset of PIs that covers every minterm (column).
- 3 rules to select a minimum subset of PIs:
    - **Essential PIs**: If a minterm is convered by only one PI, that PI is called an essential one and must be selected.
    - *A row dominated by another row can be eliminated.*
    - *A column dominating another column can be eliminated.*

2017-03-16

2017-03-23

- **Programmable array logic (PAL)** and **programmable logic array (PLA)** are
    - Similar in that both are two level design.
    - Different in that PAL is programmable on only AND plane, while PLA is programmable on both

AND plane and OR plane.

- **Read-only memories (ROM)**
  - Two dimensional array of 1's and 0's.
  - Address is input, data in the memory is output.
  - ROM is slower than PAL and PLA in that ROM is not a two-level design.
- **Multiplexers/Selectors**
  - Choose from multiple inputs to pass one to the next level.
  - $2^{n-1} : 1$ multiplexer can implement any function of n variables.
  - Can serve as a *lookup table*.
- **Tri-State** gates: output values are 0, 1, and Z, where Z means *high impedance, infinite resistance, and no connection*. **Output enable (OE)** is an additional input for *tri-state gates.*
  - When OE is *high*, this gate is a *non-inverting buffer*.
  - When OE is *low*, it is as though the gate were disconnected from the output.
- Representation of negative numbers has 3 major schemes: **sign and magnitude**, **ones complement**, and **twos complement**.
- **Sign and magnitude** representation:
  - Negative number is obtained by inverting *highest order bit*.
  - Two representations of 0.
  - Must compare *magnitudes* to determine the sign of result.
- **Ones complement** representation:
  - Negative number is obtained by inverting *every bit*.
  - Two representations of 0.
  - **End around carry**: equivalent to subtracting 2^n and then adding
- **Twos complement** representation:
  - Negative number is obtained by inverting *every bit* and then *add 1*.
  - Only *one* representation for 0.
  - The *most common* choice for integer number systems.
- The **half adder** has 2 inputs: A, B; 2 outputs: S (*sum*), C (*carry*)
- The **full adder** has:
  - 3 inputs: A, B, C_{in} (*carry-in*)
  - 2 outputs: S (*sum*), C_{out} (*carry-out*)
  - $S = A \bigoplus B \bigoplus C_{\text{in}}$
  - $C_{\text{out}} = A \cdot B + B \cdot C_{\text{in}} + A \cdot C_{\text{in}} = A \cdot B + C_{\text{in}} \cdot (A + B) = A \cdot B + C_{\text{in}} \cdot (A \bigoplus B)$
- **Carry lookahead circuits**: each of the carry equations can be implemented in a two-level logic

network

    - **Carry generate** $G_i = A_i \cdot B_i$
    - **Carry propagate** $P_i = A_i \bigoplus B_i$
    - Therefore, $S_i = P_i \bigoplus C_i$; $C_{i+1} = G_i + C_i \cdot P_i$
- **Combinational multiplier**: partial product accumulation.

2017-03-30

- [x] Homework: Quine–McCluskey Algorithm
- Solutions: quine_mccluskey.js

2017-04-06

2017-04-13

2017-04-20

- *Combinational* v.s. *Sequential* logic:
    - **Combinational logic**: A function of only the present input.
    - **Sequential logic**: A type of logic circuit whose output depends not only on the present value of its input signals but on the sequence of past inputs. Has **state (memory)**.
- **Clock**: a periodic event that causes state of memory element to change. There is a timing "window" around the clocking event during which the input must remain stable and unchanged in order to be recognized.
- Terms about latches and flip-flops:
    - $Q(t)$ or Q: the input state at time t
    - $Q(t + \Delta)$ or Q^+: the output state at time $t + \Delta$
    - Δ: the propagation time of the circuit
- Cross-coupled NOR gates:
    - **Race condition**: When both S and R return from 1 to 0 simultaneously, Q and Q' would oscillate, therefore, leading to an unstable result.
- Latches and flip-flops:
    - Latches and flip-flops are the basic elements used to store information.
    - With two states and a feedback path that allows it to store a bit of information.
    - Flip-flops can be built from latches.

- **Latch** → asynchronous; level-triggered
    - Continuously checks its inputs and changes its output correspondingly.
    - Sensitive to the duration of the pulse.
    - Transfer data as long as the switch is on.
    - Asynchronous inputs are dangerous since they take effect immediately. Glitches can be disastrous.
- **Flip-flop** → synchronous; edge-triggered
    - Continuously checks its inputs and changes its output correspondingly only at times determined by clocking signal.
    - Sensitive to signal change.
    - Transfer data only at the single instant and data cannot be changed until next signal change.
    - Synchronous inputs are greatly preferred!
- Edge-triggered device:
    - *Positive* edge triggered: Inputs sampled on rising edge. Outputs change after rising edge.
    - *Negative* edge triggered: Inputs sampled on falling edge. Outputs change after falling edge.
- Characteristic equations of flip-flops:
    - **R-S**: $Q^+ = S + R'Q$
    - **J-K**: $Q^+ = JQ' + K'Q$
    - **D**: $Q^+ = D$
    - **T**: $Q^+ = TQ' + T'Q = T \oplus Q$
- [x] Lab 1: Arithmetic Logic Unit
- Solutions: dsdl_lab1.v

2017-04-27