Arhitectura Sistemelor de Calcul November, 2021

Seminar 0x02

Cristian Rusu

1 Seminar objectives

In this seminar, we will solve problems involving:

- combinational logic and circuits;
- sequential logic and circuits;
- state machines.

2 Exercises

1. Construct the truth tables for the following logical expressions:
   (a) $X = A + BC$
   (b) $X = A(B + D)$
   (c) $X = (A + B)(A + C)$
   (d) $X = \bar{A}(B + C)D$
   (e) $X = A\bar{B}(C + D)$
   (f) $X = AB(C \oplus D)$
   (g) $X = A\bar{B}C\bar{D}$
   (h) $X = (A\bar{B} + \bar{A}B) + (C\bar{D} + \bar{C}D)$
   (i) $X = (\bar{A}\bar{B} + AB)(\bar{C}\bar{D} + CD)$
   (j) $X = A(A \oplus B)(B \oplus C)D$

2. Draw the digital circuits for the following logical expressions:
   (a) $X = (A + B)C$
   (b) $X = A + BC + \bar{D}$
   (c) $X = AB + \bar{A}\bar{C}$
   (d) $X = !(A + B)(C + D)\bar{D}$
   (e) $X = A \oplus B \oplus CD$
   (f) $X = A\bar{B} \oplus CB$

3. De Morgan's laws are: $\overline{A + B} = \bar{A}\bar{B}$ and $\overline{AB} = \bar{A} + \bar{B}$. Tasks:
   (a) verify both laws;
   (b) draw the circuits for: $\overline{A + B}$, $\bar{A}\bar{B}$, $\overline{AB}$ and $\bar{A} + \bar{B}$;
   (c) also using these two laws, evaluate/simplify the following expressions:
      i. $\overline{\bar{A} + \bar{B}}$
      ii. $\overline{\bar{A}\bar{B}}$
      iii. $\overline{A + B + C}$, generalize to N binary variables A_i
      iv. $\overline{ABC}$, generalize to N binary variables A_i
      v. $\overline{(A + B)}\bar{A}\bar{B}$
      vi. $\overline{AB}(\bar{A} + \bar{B})$
      vii. $\overline{(A + B)}(\bar{A} + \bar{B})$
      viii. $\bar{A}\bar{B}\overline{(AB)}$
      ix. $C + \overline{CB}$
      x. $\overline{AB}(\bar{A} + B)(\bar{B} + \bar{B})$
      xi. $\overline{(\overline{A\bar{B}})(\bar{B} + C)}$
      xii. $\overline{(\overline{A + \bar{B}})(\bar{B}C)}$
      xiii. $\overline{(\bar{A} + C)(\overline{AB})}$

4. Simplify the following logical expressions:

   (a) $A + 0$
   (b) $\bar{A}0$
   (c) $A + \bar{A}$
   (d) $A + A$
   (e) $A + AB$
   (f) $A + \bar{A}B$
   (g) $A(\bar{A} + B)$
   (h) $AB + \bar{A}B$
   (i) $(\bar{A} + \bar{B})(\bar{A} + B)$
   (j) $A(A + B + C + \dots)$
   (k) let $f(A, B, C) = A + B + C$
      i. $f(A, B, AB)$
      ii. $f(A, B, \bar{A}\bar{B})$
      iii. $f(A, B, \overline{AB})$
   (l) $A + A\bar{A}$
   (m) $AB + A\bar{B}$
   (n) $\bar{A} + B\bar{A}$
   (o) $(D + \bar{A} + B + \bar{C})B$
   (p) $(A + \bar{B})(A + B)$
   (q) $C(C + CD)$
   (r) $A(A + AB)$
   (s) $\overline{\bar{A} + \bar{A}}$
   (t) $\overline{A + \bar{A}}$
   (u) $D + (D\bar{C}BA)$
   (v) $\bar{D}\overline{(DBCA)}$
   (w) $AC + \bar{A}B + BC$
   (x) $(A + C)(\bar{A} + B)(B + C)$
   (y) $\bar{A} + \bar{B} + AB\bar{C}$
   (z) $(A + C)(AD + A\bar{D}) + AC + C$
   ($\aleph_0$) $\bar{A}(A + B) + (B + AA)(A + \bar{B})$
   ($\aleph_1$) $(A + B)^2 + (A + B)^3 + A + 3\bar{A} + A^3$

5. We discussed multiplexing in the lecture. Tasks:

   (a) suppose we have a MUX with two binary inputs and a one-bit select signal. Write the input-output relation for this MUX. Also draw the circuit that implements this MUX;

   (b) a MUX with two binary inputs and a binary select signal (1 bit) can simulate NOT, OR, and AND gates. Draw the MUX configurations (and write the truth table) for each gate;

   (c) we discussed in the lecture how a MUX can implement an "if" statement. Describe a circuit that implements: Y = S ? foo(A) : bar(B), where foo() and bar() are two functions (logical functions in our case). Is there an important difference between this circuit and the way we evaluate an "if" statement in a programming language?

   (d) in a programming language, if we have a vector (array) $\mathbf{x}$ of size N containing binary values, we can access the element at position i using x_i. How do we implement this access using a MUX? How many inputs does this MUX have? What is the size (number of bits) of the select signal?

   (e) we need to implement a 4-input MUX, but only 2-input MUX circuits are available. How do we implement the 4-input MUX? How many 2-input MUXes do we need? If we want to implement a MUX with N inputs, how many 2-input MUXes do we need?

6. Appendix 1 shows a single-digit display that uses 7 segments (*7-segment display*). You have 10 buttons available. Design a combinational circuit (give only the truth table) that displays the digit corresponding to the button pressed.

7. We discussed logic gates in the lecture and how they can be used to implement adder circuits. Another very important bitwise operation that we have not discussed is the shift operation. Tasks:

   (a) given a 4-bit number x, shift it to the right by 2 bits using a logical shift, an arithmetic shift, and a circular shift. Repeat the operations by shifting to the left;

(b) implement a circuit that shifts a variable x consisting of N bits to the right by d positions using a single MUX (with as many inputs as needed);

(c) can you improve the implementation in the previous problem? (hint: shift by $5 \equiv$ shift by 4, then by 1)

8. Appendices 2 and 3 show two adder circuits (2-bit and 4-bit, respectively). For each circuit, write the logical expression relating the output c_{out} to the input c_{in}. Refer to the lecture notes and use the binary variables p and g in the logical expressions you derive.

9. Consider an N-bit adder circuit for signed integers (similar to the one in Appendix 3, but generalized to N). Write the truth table for a circuit that detects *overflow*/*underflow*.

10. A digital signal A can take values in $\{0, 1\}$ over time, and we want to detect whether the sequence 010 occurs in the signal. If this bit sequence is detected in A, a variable Y is set to 1; otherwise, this variable is 0. Tasks:

    (a) draw the state diagram and the transitions between the states;

    (b) what does the sequential circuit corresponding to the diagram drawn above look like?

    (c) construct the state table for the sequential circuit;

    (d) write the expressions for the combinational logic.

11. Appendix 4 shows a sequential circuit and a timing diagram of its signals (here, C is CLK). Determine how the output Z changes over time. (hint: first redraw the circuit using the basic NOT, OR, and AND gates - this will make the circuit easier to read)

12. We discussed a 2-bit sequential counter circuit in the lecture and wrote its state representation / truth table. Tasks:

    (a) use logical functions to write the relation between the "previous state" and the "next state". Also draw the sequential circuit that implements the 2-bit counter. (hint: each state consists of 2 bits; denote these bits by q_1 and q_0);

    (b) extend the counter to operate on 3 bits;

    (c) Gray code is a binary code in which consecutive symbols differ by exactly one bit (for example, the 3-bit Gray code is 000, 001, 011, 010, 110, 111, 101, and 100). Modify the 3-bit counter created above so that it counts in Gray code.

13. Implement Euclid's algorithm for the GCD using a sequential circuit.

14. Implement an algorithm that checks whether a number is prime using a sequential circuit.

15. Consider two binary variables A and B, each consisting of 3 bits. Write logical functions that take the value "1" only if:

    (a) A contains exactly one "0" bit;

    (b) A contains an odd number of "0" bits;

    (c) A is a number less than 4 in decimal;

    (d) A == B;

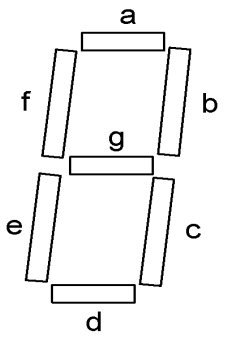
    (e) A < B, if A and B are nonnegative integers;

    (f) A < B, if A and B are signed integers represented in two's complement.

16. Show that NOR and NAND gates are each universal (that is, either type of gate can be used on its own to implement the 3 basic gates NOT, OR, and AND).

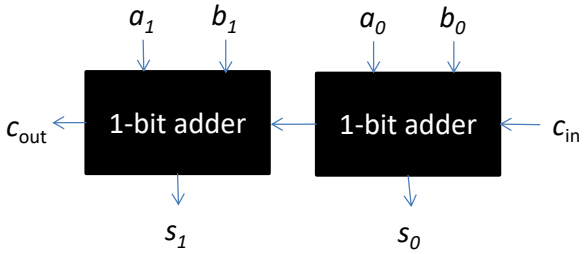
17. We have a vending machine that sells snacks. It accepts coins of 10 bani, 50 bani, and 1 leu. When a coin is inserted into the machine, a digital signal is activated according to the coin (A, B, and C, respectively), and a general digital signal (G) is activated (a coin has been inserted). There is also a reset signal (RST) that resets the inserted amount to zero. We want a circuit that activates a digital output X when the inserted amount exceeds 60 bani.

Appendix 1



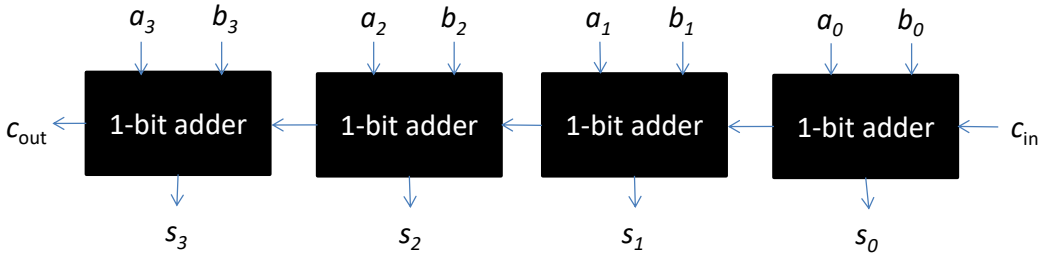
7-segment digit display.

Appendix 2



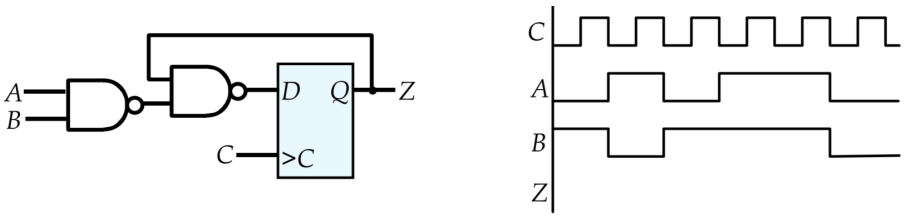
2-bit adder circuit.

Appendix 3



4-bit adder circuit.

Appendix 4



Timing diagram for a sequential circuit.