Fall 2003 CS150 - Problem Set 2: Solutions

Question 1.
a. Many possibilities. Here is one using 3 LSB of input as select lines for Mux.

D_2	D_1	D_0	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	D_3
1	0	0	0
1	0	1	D_3
1	1	0	D_3
1	1	1	1

b. Use D_1 and D_0 to select mux inputs.

D_1	D_0	Y
0	0	0
0	1	$D_3 \cdot D_2$
1	0	$D_3 \cdot D_2$
1	1	$D_3 + D_2$

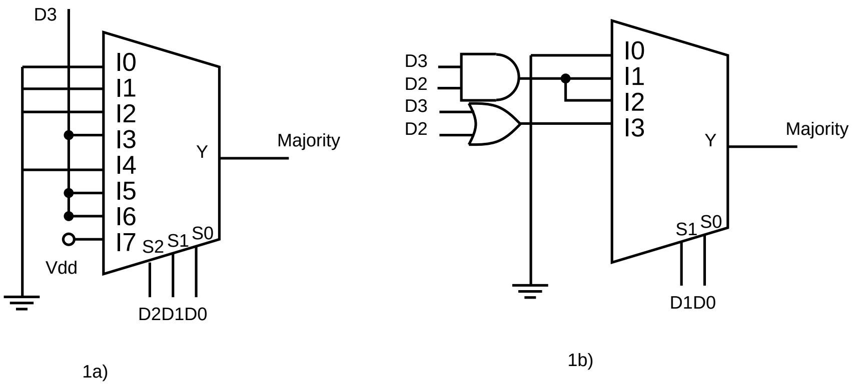


Figure 1: Mux implementation of majority logic.

Question 2.
a) Three 3-input LUTs can be connected to make a 4-input LUT. Let the 4 input LUT be $L(a, b, c, d)$. We will create two 3-input LUTS $f(a, b, c)$ and $g(a, b, c)$, where $f(a, b, c) = L(a, b, c, 0)$ and $g(a, b, c) = L(a, b, c, 1)$. So f will calculate the same thing as L given that $d = 0$ and g will calculate the same thing as given that $d = 1$. Finally, we will hook both of these 3-input LUTs into a third 3-input LUT $h(f, g, d)$ which will return f if $d = 0$ and g otherwise. Hence the function for it will be $h = f\bar{d} + g \cdot d$. As a result, and 4 input lookup function can be implemented.
b) Given the 4 input LUT we created in part a), we can build a 5-input LUT $M(a, b, c, d, e)$ by combining two 4 input LUTs, $L_1(a, b, c, d)$ and $L_2(a, b, c, d)$ where $L_1() = M(a, b, c, d, 0)$ and $L_2() = M(a, b, c, d, 1)$ and again using a 3 input LUT as a 2:1 MUX such that $M = L_1 \cdot \bar{e} + L_1 \cdot e$.
Question 3.
Denote inputs to this bit A_i and B_i, outputs from this bit L_i $(A < B)$ and E_i $(A = B)$, and inputs from the higher bit L_{i+1} and E_{i+1}.

To determine L_i:

- If $L_{i+1} = 1$, then $L_i = 1$ (the most significant bits of A are are less than B).
- If $E_{i+1} = 1$, then $L_i = \overline{A} \cdot B$ ($L_{i+1} = 0$ as A and B bits were equal).
- Otherwise $B < A$ and $L_i = 0$.
- Hence, $L_i = L_{i+1} + E_{i+1} \cdot \overline{A} \cdot B$.

To determine E_i:

- If $E_{i+1} = 1$, then $E_i = A \cdot B + \overline{A} \cdot \overline{B}$.
- Otherwise, $A < B$ or $B < A$, and $E_i = 0$.
- Hence, $\begin{aligned} E_i &= E_{i+1}(A \cdot B + \overline{A} \cdot \overline{B}) \\ &= E_{i+1}\overline{(A \oplus B)} \end{aligned}$

To determine $\min_i(A, B)$:

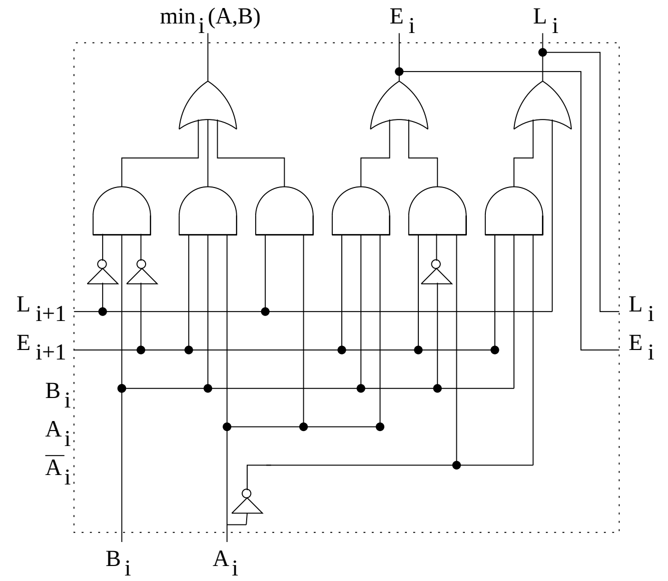
- If $L_{i+1} = 1$ then $\min_i(A, B) = A_i$.
- If $E_{i+1} = 1$ then $\min_i(A, B) = A_i \cdot B_i$. Either $A_i = B_i = 1$, or the minimum is zero.
- Otherwise $B < A$ and $\min_i(A, B) = B_i$.
- Hence, $\min_i(A, B) = L_{i+1} \cdot A_i + E_{i+1} \cdot A_i \cdot B_i + \overline{L}_{i+1} \cdot \overline{E}_{i+1} \cdot B$.

	L_{i+1}	E_{i+1}	L_i	E_i	$\min_i(A, B)$
$B < A$	0	0	0	0	B_i
$A = B$	0	1	$\overline{A}_i \cdot B_i$	$\overline{(A \oplus B)}$	$A_i \cdot B_i$
$A < B$	1	0	1	0	A_i
illegal	1	1			

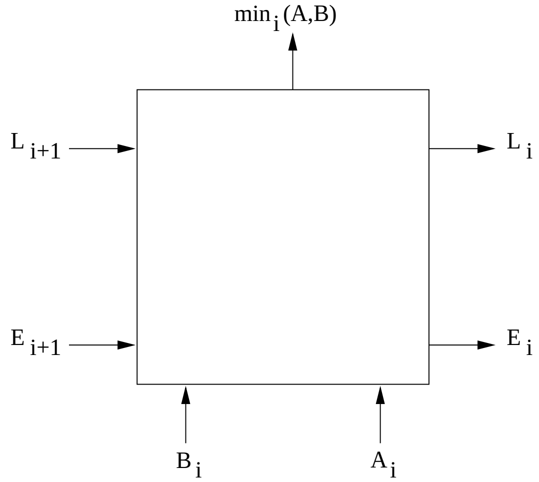
Table 1: Truth table for $\min_i$, L_i and E_i.

Shown below is a modular, combinatorial circuit which implements these functions and is easily cascaded.

In order for the combinatorial logic modules to correctly determine $\min(A, B)$, the most significant bit requires inputs of $E_8 = 1$ and $L_8 = 0$ (these are the inputs which would arrive from a higher significant bit to the module).



Modular logic diagram.



Block diagram of the inputs and outputs for the logic.

Figure 2: Logic and block diagrams for the modular circuit to compare A and B.

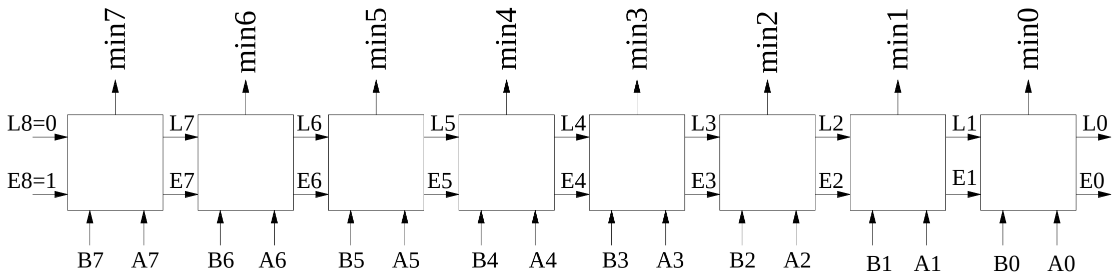


Figure 3: (b) Interconnection of 1 bit modules to implement 8 bit combinatorial logic, which computes the minimum of two 8 bit numbers A and B.

(c) To calculate the maximum delay, assuming unit delay for all gates and interters:

- Consider the most significant bit: E_8 and L_8 do not change, so the critical paths to consider are those for A_7 and B_7. To get the correct E_7 and L_7 outputs for the next stage: the longest paths for A_7 and B_7 are through an inverter, an AND gate, then an OR gate - taking **3 time units**.
- For lower significant bits, inputs A_i and B_i will be held constant until all the modules have finished - they have been stable for 3 units (or more) of time already. Whereas, the module cannot correctly compute E_i and L_i until E_{i+1} and L_{i+1} arrive - hence E_{i+1} and L_{i+1} are the critical signals to consider.

  To determine E_i and L_i, E_{i+1} and L_{i+1} signals must pass through one AND gate and one OR gate - a delay of 2 time units. There are six modules between module 7 and module 0, so $2 \times 6 =$ **12 time units**.
- Module 0 receives E_1 and L_1, what is the maximum delay to determine $\min_0(A, B)$? At worst, E_1 and L_1 have to propagate through an inverter, an AND gate, and an OR gate - a further **3 time units**.
- Thus, the worst case delay for calculating the 8 bit minimum is **18 time units**.