ECE 2372 Modern Digital Design July 15

Reading assignment for today: Ch 3: section 3-5 Decoding

One common application of digital logic circuits is decoding binary patterns so they can be displayed in a more convenient form. Let us consider a very simple case. Suppose with had a two-bit system for encoding and decoding the wind direction. With a two-bit system, we can have 4 possible combinations, in this case: N, E, S, and W for the 4 directions. We might encode them according to the following table:

| Binary code | |
|---|---|
| A B | direction |
| 0 0 | N |
| 0 1 | E |
| 1 0 | S |
| 1 1 | W |

Another way to show this:

| Binary code | direction | | | |
|---|---|---|---|---|
| A B | N | E | S | W |
| 0 0 | 1 | 0 | 0 | 0 |
| 0 1 | 0 | 1 | 0 | 0 |
| 1 0 | 0 | 0 | 1 | 0 |
| 1 1 | 0 | 0 | 0 | 1 |

Note from the table above that since we have 4 different output functions, we need 4 different logic functions, one to decode each of the 4 cases.

N = F(A,B) = ~A~B

E = F(A,B) = ~AB

S= F(A,B) = A~B

W = F(A,B) = AB

An example SOP solution is shown in figure 3-13 of your textbook on page 129. This type of circuit is often called a 2-to-4-line decoder. Chips with this functionality built-in are readily available. Chips with extended decoding capability (e.g the 74LS138 3-to-8-line decoder) are also readily available. You are encouraged to visit www.mouser.com and enter the search term 74LS138, and take a look at the datasheet for that component.

We are going to consider part of the design of a 4-bit BCD to 7-sgement display decoder circuit. Recall earlier that we can represent a decimal number by using 4 bits to represent the decimal number as 0 thru 9. So, for example the BCD bit pattern 0000 = 0, and 1001 = 9. Numbers greater than 9 are not valid in BCD, so patterns 1010 thru to 1111 are not valid BCD numbers. It is convenient, for clocks, and decimal displays to store decimal numbers as 4-biut BCD numbers internally, but display them as conventional 7-segment display numbers. The 7-segment display pattern assignments were developed a very long time ago and are well standardized. The very top segment is the a segment, top right is the b

segment, etc. These are shown on page 3 of the attached 74LS47 decoder datasheet that is included as a separate file.

Since we need to decode 4 bit combinations for each segment, we need a total of 7 functions of the 4-bit variables ABCD.

| $2^3$ | $2^2$ | $2^1$ | $2^0$ | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 8's | 4's | 2's | 1's | Segment functions | | | | | | |
| A | B | C | D | a | b | c | d | e | f | g |
| 0 | 0 | 0 | 0 | 1 | 1 | | | | | |
| 0 | 0 | 0 | 1 | 0 | 1 | | | | | |
| 0 | 0 | 1 | 0 | 1 | 1 | | | | | |
| 0 | 0 | 1 | 1 | 1 | 1 | | | | | |
| 0 | 1 | 0 | 0 | 0 | 1 | | | | | |
| 0 | 1 | 0 | 1 | 1 | 0 | | | | | |
| 0 | 1 | 1 | 0 | 1 | 0 | | | | | |
| 0 | 1 | 1 | 1 | 1 | 1 | | | | | |
| 1 | 0 | 0 | 0 | 1 | 1 | | | | | |
| 1 | 0 | 0 | 1 | 1 | 1 | | | | | |
| 1 | 0 | 1 | 0 | x | x | | | | | |
| 1 | 0 | 1 | 1 | x | x | | | | | |
| 1 | 1 | 0 | 0 | x | x | | | | | |
| 1 | 1 | 0 | 1 | x | x | | | | | |
| 1 | 1 | 1 | 0 | x | x | | | | | |
| 1 | 1 | 1 | 1 | x | x | | | | | |

Part of the table has been filled in above. The table is complete for segments a and b. It is left as an exercise for the student to finish this table for the remaining segments c thru g. Notice that for patterns greater than 9 (1001), the output function is given as x. We could have used d in place of x, but d or x in such a location represents a don't care term. That is, since codes 1010 thru 1111 are not valid, we should not ever see those codes, and it may not be necessary to consider them. If you absolutely do NOT want codes from 1010 to 1111 to display anything, then you must use ZEROs in the decoded function to make sure no segments are ever turned on for those invalid cases.

| | AB | CD | | | |
|---|---|---|---|---|---|
| 00 | | 00 | 01 | 11 | 10 |
| | | 0000<br>1 | 0001<br>0 | 0011<br>1 | 0010<br>1 |
| | | 0100<br>0 | 0101<br>1 | 0111<br>1 | 0110<br>1 |
| | | 1100<br>x | 1101<br>x | 1111<br>x | 1110<br>x |
| | | 1000<br>1 | 1001<br>1 | 1011<br>x | 1010<br>x |

Due to making use of the don't care positions, we are able to reduce this K-map quite a bit. Noitce the groupings we obtained: C + A + BD + ~B~D. Notice that one of the groupings is the 4 corners which form the group ~B~D.

It is left as an exercise for the student to solve the remainder of this 4-bit BCD to 7-segment display decoder circuit. You will need to complete the work for segments; b,c,d,e,f, and g. You may want to look at the commercial component 74LS47 (datasheet provided separately in this same folder). Note that in the commercial component, some additional capability is provided which increases the complexity somewhat. A logic level diagram of the entire circuit is given on page 5. Consider the impact on your solution if you were not allowed to make use of the don't care option, that is if the display had to remain totally blank or provide some type of error code for any input patterns between 1010 and 1111. Consider further what would be involved to make a 4-bit to 7-segment display decoder that would properly display 0-F of the hexadecimal numbers for any of the 4-bit binary input patterns.

First exam will be tomorrow. It will be posted on BB in the subfolder exams, and will be open for a limited number of hours. It will open around mid day and close by 11:59 pm (midnight) tomorrow Thursday July 16th. Open notes and open book of course. Any calculator. Do your own work and don't collaborate with anyone. You will upload your answer sheet back to BB.