cs281: Introduction to Computer Systems

Lab03 – K-Map Simplification for an LED-based Circuit

Overview

In this lab, we will build a more complex combinational circuit than the sum bit of a full adder that we did in Lab02. In fact, since it will have three boolean outputs, it is really composed of three combinational circuits. But we will use simplification techniques that will allow us to "do more with less." The lab will also lead us through all the steps of the design process, from high level description of the objective through its realization.

The lab will use the analog input of a potentiometer (one of the dial knobs at the bottom of the breadboard), use the Arduino to convert that analog input into a binary value in three bits (ranging from decimal 0 to 5), and then build combinational circuits to control three LED's that will light up, based on how "high" the potentiometer is turned.

The figure below shows, for each of the decimal input values, the configuration of LEDs. An open circle indicates the LED is off. I chose LED colors of Red, Yellow, and Green in my implementation, but you can implement with other colors if you like. Also note that the LED lighting is "cumulative". So if the decimal input value is 3, for instance, both LED1 (Red) and LED2 (Yellow) are lit. Also note that the same result should occur in *both* cases that the input, interpreted as a decimal value, is either 1 *or* 2, and likewise for inputs 4 *or* 5. Also note that the LED result is not defined for *any other* value of input.

Decimal Input	LED Result
0	LED3 ○ LED2 ○ LED1 ○
1, 2	LED3 ○ LED2 ○ LED1 ●
3	LED3 ○ LED2 ● LED1 ●
4, 5	LED3 ● LED2 ● LED1 ●

In the following first section, we will focus on the three binary inputs and the combinational circuits for driving the LEDs. The second section will add in the dial knob (potentiometer) to Arduino and the Arduino providing the 3 bits of input needed for the circuits developed in the first section.

LED Control Circuits

Let's call our inputs A, B, and C, and start by filling in the truth table where binary output *LED1* indicates that the first LED should be lit, *LED2* indicates that the second LED should be lit, and *LED3* indicates that the third LED should be lit.

Q1 Based on the bit representation of the 0 to 5 input, you should fill in 0 or 1 values for the three outputs in the truth table. Since the state of the system is undefined for the last two rows of the truth table (i.e

the cases where the decimal input value is 6 or 7) we put an **X** as the output for all three LED output variables, indicating a **"Don't Care"** for that output.

row	A	B	C	*LED1*	*LED2*	*LED3*
0	0	0	0			
1	0	0	1			
2	0	1	0			
3	0	1	1			
4	1	0	0			
5	1	0	1			
6	1	1	0			
7	1	1	1			

Q2 Fill in the boolean expression defining *LED1*, in terms of A, B, and C in SOP form. Do not simplify:

LED1 $=$

Q3 Fill in the boolean expression defining *LED2*, in terms of A, B, and C in SOP form. Do not simplify:

LED2 $=$

Q4 Fill in the boolean expression defining *LED3*, in terms of A, B, and C in SOP form. Do not simplify:

LED3 $=$

Recall from the prelab that we can rewrite the truth table in the form of a K-map that will allow us to find simplifications. Each output variable will require its own K-map, so, unlike the above truth table, we cannot use a single K-map for all three outputs. Based on what you learned from the prelab, fill in each of the following K-maps based on the output values from the truth table above:

Q5 *LED1* K-map:

	BC			
A	0 0	0 1	1 1	1 0
0				
1				

Q6 *LED2* K-map:

	BC			
A	0 0	0 1	1 1	1 0
0				
1				

Q7 *LED3* K-map:

	BC			
A	0 0	0 1	1 1	1 0
0				
1				

Although the prelab provided some of the motivation and foundation on how a K-map can help provide simplified boolean expressions for an output, it stopped short of actually giving the full process of taking the K-map version of the truth table and determining the smallest set of simplified terms. Your instructor will now cover that missing piece before you proceed.

Magic Happens Here

Q8 Annotate the K-map for *LED1*, circling a minimal set of prime implicants, and labeling each circled prime implicant with its associated boolean expression. Then fill in the boolean expression defining *LED1* based on the K-map:

LED1 =

Q9 Annotate the K-map for *LED2*, circling a minimal set of prime implicants, and labeling each circled prime implicant with its associated boolean expression. Then fill in the boolean expression defining *LED2* based on the K-map:

LED2 =

Q10 Annotate the K-map for *LED3*, circling a minimal set of prime implicants, and labeling each circled prime implicant with its associated boolean expression. Then fill in the boolean expression defining *LED3* based on the K-map:

LED3 =

Breadboard Setup

We will use A, B, and C to refer to the 3 inputs to the circuit, and *LED1*, *LED2*, *LED3* to refer to the three outputs. We also need to refer to elements of the breadboard, so S_1 through S_8 will refer to the breadboard's *Logic Switches* at the lower left side of the board.

1. Pick three of the 5 pin "rows" on the breadboard to be designated for the three inputs A, B, and C. Wire Logic Switch S_1 to A, S_2 to B, and S_3 to C.
2. Wire A to Logic Monitor/Logic Indicator 1, B to Logic Indicator 2, and S to Logic Indicator 3. This will give us the ability to visualize the inputs to the circuit.
3. Designate another three rows on the breadboard to be binary outputs *LED1*, *LED2*, *LED3*, and wire to Logic Indicators 6, 7, and 8.
4. Wire from the LED1 row to the positive input for an LED circuit. Recall from Lab01 that an LED circuit starts with a row for the input, then we wire a 330 Ohm resistor from that row to another row, the positive (longer) lead of an LED is wired onto that second row, and the negative lead to another row, and that final row is wired to GND.
5. Repeat with LED2 and LED3 to two additional (independent) LED circuits. You should test each of your LED circuits by temporarily wiring +5V to each LED circuit, but then removing +5V after testing.

Circuit Realization

Based on your equations from Q8, Q9, and Q10, and obtaining necessary AND, OR, and NOT chips, implement the three circuits for LED1, LED2, and LED3, wiring the outputs to the rows designated for these outputs as defined above. Test and Debug your circuit.

Q11 Demonstrate your circuit for your instructor or student assistant:

Q12 Describe what happens when you use the manual Logic Switches to encode the Don't Care inputs for decimal input 6 or 7. Are the results what you expect?

Analog Input to Binary Encoding

Imagine we want to use a physical knob to "turn up the volume" to drive our cumulative three LED circuit implemented above. To do so, we will use the knob on the left at the bottom of the breadboard, labeled ***10K POT***. The potentiometer is, in effect, a way to provide a variable amount of resistance, depending on how "far" you have turned the potentiometer. The 10K refers to the the range of resistance possible with this particular potentiometer.

Fortunately, we do not have to learn a lot about the analog world to accomplish our goal for this lab. The Arduino will do the "heavy lifting" both in terms of taking the result of the potentiometer and obtaining a digital value, and then covering the digital value into our needed range of 0 to 5_10 ($000_2 \ldots 101_2$). The Arduino circuit board and the `analogRead()` library function take care of the first, and the sketch that we write will take care of the second.

The Wiring

1. Wire the outer left column of the pins for the 10K POT to GND.
2. Wire the outer right column of the pins for the 10K POT to +5V.
3. Wire one of the middle columns of the pins for the 10K POT to *Analog In* pin 0 on the Arduino.
4. Wire +5V and GND on the Arduino to the breadboard, as done in previous labs.
5. Remove the wires from breadboard Logic Switches S_1, S_2, and S_3 to A, B, C, respectively.
6. Wire Digital pin 11 on the Arduino to A, pin 12 to B, and pin 13 to C.

Arduino Sketch

Type in the following sketch, correcting any compilation errors and then loading to the Arduino.

```
const int potpin = 0;
const int WAIT = 100;
void setup(){
  Serial.begin(9600);

  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);

  pinMode(potpin, INPUT);
}

void loop(){
  int val;
  int dval;

  val = analogRead(potpin);
  dval = val/171;

  Serial.print("From POT: ");
  Serial.println(val);
  Serial.print("Decimal value conversion: ");
  Serial.println(dval);

  int bitval = dval & 1;
  digitalWrite(13, bitval);
  dval = dval >> 1;
  bitval = dval & 1;
  digitalWrite(12, bitval);
  dval = dval >> 1;
  bitval = dval & 1;
  digitalWrite(11, bitval);

  delay(WAIT);
}
```

Q13 Demonstrate your completed system for the instructor or student assistant.