LED Grid with 2 Buttons

Project Description: Using two buttons (left and right). When the left button is pressed, the row of LEDs fill up from left to right. When the right button is pressed, the rows fill from right to left. When all the rows are filled, a short animation is displayed and the program resets.

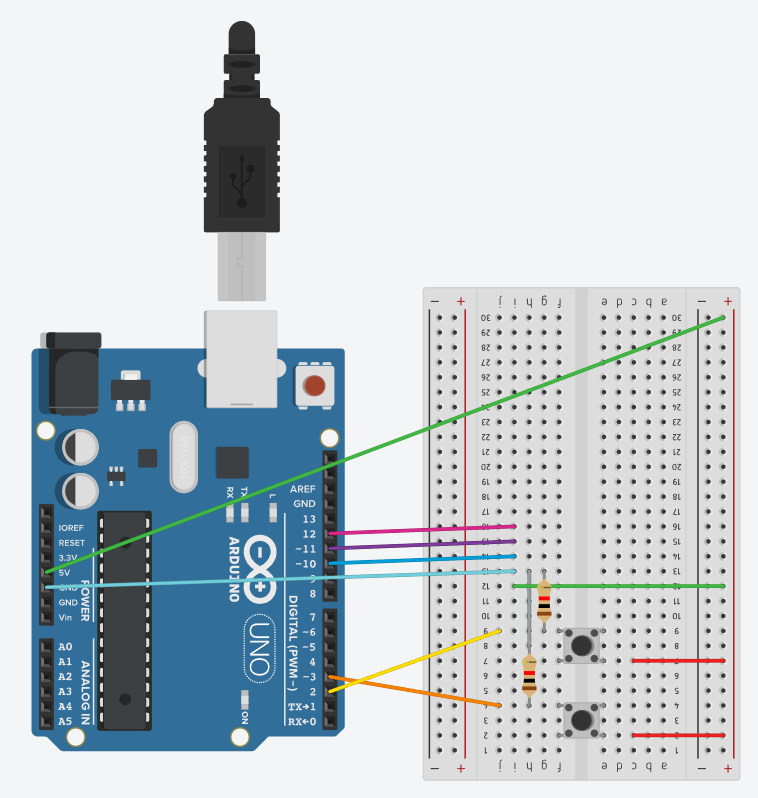
Video of a sample project:

[LED Grid - 2 Buttons DEMO](https://drive.google.com/file/d/1OBrycrL7BmQDyLa8ZbezD_q5-DElXdaw/view?usp=sharing)

Circuit Diagram:

Materials Needed:

* 2: Buttons
* 10: Wires
* LED Matrix
* 2: 1k Resistors
* Arduino Uno



This circuit diagram does NOT include the LED grid. The LED grid should be placed in row j.

With GND (row 13), Vin (5V), DIN (row 14 breadboard), CS (row 15), CLK (row 17)

Sample Code:

| #include "LedControl.h"  // Set the LED grid pins to specific ports on the Arduino int DIN = 10; int CS = 11; int CLK = 12;  // Keep track of the state of the button. int leftButtonState; int rightButtonState;  // Number of rows on the Arduino int rows = 8; int columns = 8; int counter = 0;  // The address is the LED we are currently on - since we only have 1 grid, this will be 0 int address = 0;  // The LED grid object used LedControl lc=LedControl(DIN,CLK,CS,1);   /\*\*  \* The set up for our grid and buttons.  \*/ void setup() {  /\*  The MAX72XX is in power-saving mode on startup,  we have to do a wakeup call  \*/  lc.shutdown(address,false);    // Sets the light intensity of the display  lc.setIntensity(address,1);    // Clears the display  lc.clearDisplay(address);   // Setting the button pins to input pins  pinMode(2, INPUT);  pinMode(3, INPUT);  }  /\*\*  \* Continuously checks for button presses.  \*/ void loop() {  // Read the button states  leftButtonState = digitalRead(3);  rightButtonState = digitalRead(2);   // Checks if left button pressed - left animation  if(leftButtonState == HIGH){  writeLeft(counter);  counter++;   // Right button - right animation  } else if(rightButtonState == HIGH){  writeRight(counter);  counter++;   // When the Grid is full - clear  } else if(counter > rows){  eraseAll();  counter = 0;  }   /\*\*  \* BONUS CHALLENGE: How can we change the code such that each time we   \* restart the grid fills the opposite way than previous.   \* For example: first go from 0 -> 7, then 7-> 0 and repeat  \*/ }  /\*\*  \* Fills a row from left to right.  \*/ void writeLeft(int rowToFill){  // Initially all LEDs are blank.  byte x = B00000000;  /\*\*  \* Basic principle for left shifts  \* 0000 >> 0001 >> 0010 >> 0110 >> 1110 >> 1111 // The shifted value  \* + 0001 0001 0001 0001 // Add 1 to the end  \* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \* 0001 0011 0111 1111  \*/   // Iterates through all the rows.  for(int i = 0; i < rows; i++){  // Shifts a bit left one  x << 1;    // Adds to the end  x++;    // Sets a row to the corresponding value  lc.setRow(address, rowToFill, x);   // Delays so we can see the animation occur  delay(100);  } }  /\*\*  \* Fills an input row from right to left  \* rowToFill: the row filled  \*/ void writeRight(int rowToFill){  /\*   \* Since we are filling from right to left we need to get from  \* 0000 -> 1000 -> 1100 -> 1110 -> 1111  \*   \* Bit shifting pads a number with 0s. Thus we need to add 1s to the left when bit shifting  \* \*Basic Principle\*  \* 0000 >> 1000 >> 0100 >> 0110 >> 0111 our shifted bit: x  \* + 1000 1000 1000 1000 our added value: add  \* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \* 1000 1000 1100 1110 1111  \*   \*/   // Added value  byte add = B10000000;   // Initial value  byte x = B00000000;   // Iterates through the row inputted  for(int i = 0; i < rows; i++){   // Shifts x right  x = x >> 1;   // Adds the added number  x += add;    // Sets the corresponding row to match the bit shifted values  lc.setRow(address, rowToFill, x);   // Delays so the image can be seen  delay(100);  } }  /\*\*  \* Once the LED fills up, we create a small animation to clear the screen.  \*/ void eraseAll(){  // We want to iterate through all the rows to clear each row.  for(int i = 0; i < rows; i++){   // The initial state of the LEDs (all on)  byte x = B11111111;   // Iterate through each column of the LED and clears one LED at a time.  for(int j = 0; j < columns; j++){    // Checks if the row is even or odd  if(i % 2 == 0){     // even: bit shift right  x = x >> 1;    }else {     //odd: bit shift left  x = x << 1;    }    // Sets the current row the the state of x.  lc.setRow(address, i, x);   // Delays so we can see the changes on the LED grid  delay(50);    }  } } |
| --- |