LUMISSIL Microsystems
A Division of ISSI

Matrix Drivers
Make or Buy?

The increase of home appliance intelligence is presenting a challenge for ease of use leading to consumer's low acceptance of new functions. The appliance user interface is the consumer's first and only point of contact for controlling and modifying the appliance's functionality. Having an intuitive user interface to guide the consumer's interaction with the appliance enables maximum use of product features while keeping its operation simple.

LEDs found in a user interface not only provide appliance status feedback but also guide the programming of its functions. To accomplish this requires many LEDs in strategic locations, with different colors, brightness levels and animation capabilities. Addressing such large numbers of LEDs presents a challenge since every LED requires its own power and control method.

Matrix LED drivers are designed to power and control an array of LEDs configured in a row and column grid layout. Driver flexibility is accomplished by including programmable registers that are used for individual adjustment of both the LED current and animation, plus other advanced features. These drivers commonly interface with an MCU, enabling independent programming and control of each LED in the matrix. This individualized control is the preferred method for displaying human-machine (HMI) information on the appliance user interface. Lumissil's LED driver product line includes various families of matrix LED drivers that cover a range of matrix sizes and LED current levels. This diverse portfolio of matrix drivers provides a choice of driver solutions to meet a wide range of matrix LED requirements.

However, there are circumstances where design engineers are more comfortable designing their own discrete matrix driver solution. This article will discuss the benefits of using a dedicated matrix LED driver versus designing one using conventional discrete methods.

LED MATRIX BACKGROUND

An LED array is necessary when the LED count exceeds 36, as supporting a higher number of LEDs would require more IC package pins. This increases the package size requiring a larger PCB area and leads to higher power consumption and increased driver cost. An LED matrix is a two-dimensional array organized in rows and columns, such that any LED can be identified by its row and column position. Matrix LED drivers implement a row/column scanning method, reducing pin count and package size while effectively controlling numerous LEDs.

For instance, driving 64 LEDs would require 64 pins, one pin for each LED. However, by connecting all the LED cathodes together in rows (R1 ~ R8), and LED anodes in columns (C1 ~ C8), the resulting LED array would require only 16 pins to control it. In this array, each LED is addressed by referencing its row and column location. While only one column can be ON at a time, any or all rows can be ON simultaneously. In Figure 1, if C1 is ON and R4 is pulled low, then only the LED in the fourth row and first column will be turned ON, (R4,C1).

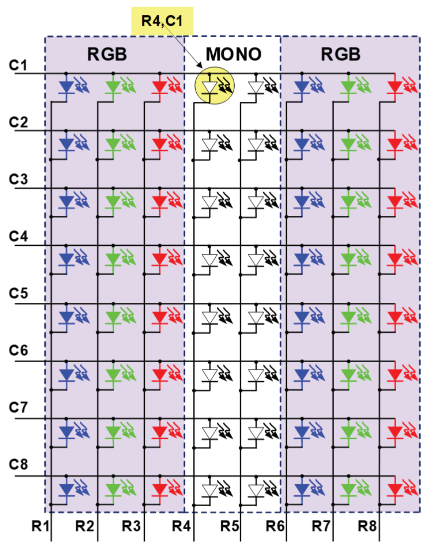

Figure 1 LED 8x8 LED Array

The entire LED array can be made to appear as fully ON by quickly scanning all the rows and columns in a multiplexed scanning sequence, Figure 2. In this approach, all rows are activated, while a single column is briefly powered ON before being turned OFF then the next column is briefly power ON. This sequence repeats for each column in the sequence. Once the last column (C8) has been powered ON for the specified time period, the cycle sequence reintitiates, starting with the first column (C1). When this cycle is done swiftly, it 'appears' that all the LEDs in the 8x8 array are continuously illuminated. This is due to the 'Persistence of Vision' or POV, where the human eye tends to blend the light emitted from a rapid sequence of lights.

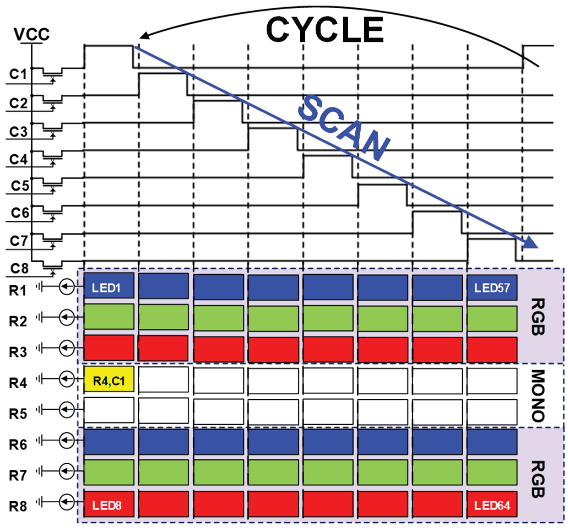

Figure 2 Matrix Multiplexed Scanning Sequence

The implementation of an LED matrix row/column scanning method using discrete components has been a conventional approach in controlling numerous LEDs in appliance user interfaces and control panels. While this method has proven effective over the years, it entails hidden performance costs.

To implement a discrete LED matrix scan, an MCU with scanning software is required. The MCU's GPIO outputs are responsible for driving external FETs, necessary for handling high LED currents, given that a microcontroller's GPIO has limited current-handling capability. This proves to be a cost-effective strategy when dealing with a small number of LEDs, and when the MCU possesses sufficient bandwidth to support ON/OFF control of the LEDs (refer to Figure 3).

However, if LED animation effects are desired, then a high bandwidth and more expensive MCU along with its associated software overhead is necessary. This is because the MCU must consistently refresh the GPIOs to achieve smooth dimming and animation.

In addition, columns (C1~C8) can only operate at a set current. This can be an issue if there is a mix of Red, Green, Blue or White LEDs that require different current levels. Variations in voltage, LED and resistor tolerances can lead to differences in current through different LEDs, resulting in perceived variation in brightness levels at the user interface.

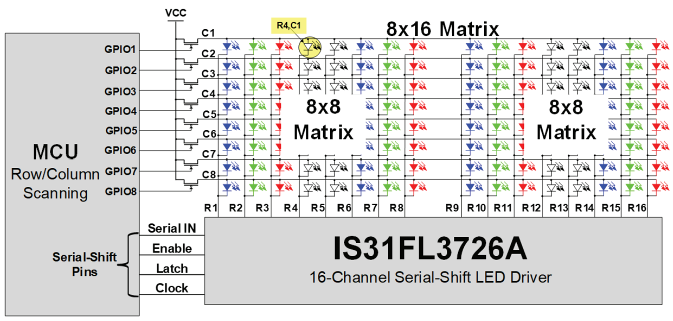

Figure 3. Discrete 8x16 LED Matrix

In the discrete matrix approach, the MCU has the responsibility for array scan timing by coordinating the column (C1~C8) timing with the rows (R1~R16) created by the IS31FL3726A LED serial shift driver. The MCU's involvement in scan timing and LED ON/OFF control demands a portion of its resources. This allocation restricts the MCU's ability to perform additional tasks, such as managing sensors or conducting calculations for the appliance. While a high-end MCU may have sufficient bandwidth to handle scan timing and other functions simultaneously, this inevitably raises the overall cost of the MCU.

A discrete implementation will provide basic functionality of turning ON/OFF the individual LEDs, but any RGB color mixing or LED dim effect will add to the MCU bandwidth requirements. Other functions missing with a discrete approach are LED de-ghosting, individual LED current adjust, LED error detection and EMC suppression capabilities.

DEDICATED LED DRIVERS FOR USER INTERFACES

An alternative approach is to use a dedicated matrix LED driver to offload a significant amount of MCU resources. Matrix LED drivers have programmable registers which facilitate the animation, color changing, brightness adjustment and de-ghosting of an LED array. The MCU only needs to update these registers over a standard I2C bus resulting in lower bandwidth requirements. This registerbased serial communication minimizes MCU hardware and software resources since there are no GPIOs and no timing synchronization that require a constant update.

Figure 4 shows a circuit implementation using the IS31FL3719 to replace and simplify the same 8x16 array that used discrete components, Figure 3. The IS31FL3719 has the key features necessary to assure a consistent LED output. It also has other features such as LED current adjust, LED ghost elimination, LED open/short detection and EMC suppression not readily available in a discrete implementation. The IS31FL3719 will replace the functionality of a discrete matrix design while adding performance features, requiring less MCU resources/pins and comes in a small 4x4mm package.

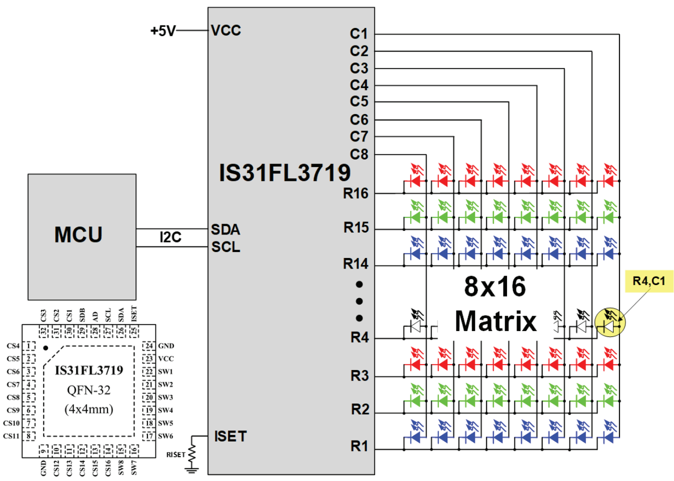

Figure 4. IS31FL3719 Matrix LED Driver Circuit and Package

As discussed earlier, accessing each LED in an array requires a scanning architecture as shown in Figure 2. Lumissil's matrix LED drivers such as the IS31FL3719 have an internal state machine that sequentially cycles through each C1~C8 and enables all 16 current sinks (R1~R16) for a short duration. Only those LEDs associated with the column and current sinks (rows) will turn ON for a specific time before the sequence continues to the next column.

Each LED in the 8x16 matrix has an associated programmable register for adjusting its operating PWM value and current level. As the state machine cycles from C1 to C8, specific register values are applied to the current sink rows (R1~R16). For example, when C1 is active, only the values associated with C1,R1~C1,R16 are updated and applied to the LEDs in C1 column. There is a short period between column activations called 'de-ghost' time used for discharging any parasitic capacitors which if not discharged might cause a false turn ON of an LED in the array. After this de-ghost time the state machine enables the next 'Cx' switch and updates its associated Rx LED registers. This scanning cycle continues indefinitely in a loop going from C1 to C8 then back to C1, etc. MCU overhead is minimized since the MCU only needs to perform register updates over a fast I2C bus.

MAKE OR BUY A MATRIX DRIVER?

Let's revisit the question should you design or buy a matrix driver? While it is possible to make a discrete matrix driver solution, it will lack many advanced functions. It will require a performance MCU, more components and more PCB space. A dedicated matrix driver from Lumissil has many advantages over a discrete solution. Among them is freeing up the MCU and it's software overhead so it can focus on other functions such as monitoring sensors, controlling motor or relays and performing algorithm calculations. A single chip matrix driver also facilitates the implementation of LED lighting effects and integrates many key functions such as LED de-ghosting and EMC suppression.

For the above reasons, it is advantageous to use Lumissil matrix drivers like the IS31FL3719 instead of designing a discrete matrix solution.

For additional information please see our Youtube video LED Matrix Drivers - Make or Buy? at https://www.youtube.com/watch?v=Y1dTbBM4M8g