PONG IMPLEMENTATION IN 8 X 8 DOT MATRIX DISPLAY USING MSP430

Aadith V Menon,Abdul Fathaah Shamsuddin,Abhijith P
Electronics And Communication Department,
National Institute Of Technology, Calicut

1 Abstract

Microcontrollers omnipresent around us from basic household appliances like washing machines to advanced military equipments. This project demonstrates techniques to interface a microcontroller with the interactive world through a game pong which is one of the earliest arcade games released in 1972. The state of the game is controlled by user input and the output which is displayed on 8 x 8 dot matrix display is interfaced with the microcontroller serially.

2 Introduction

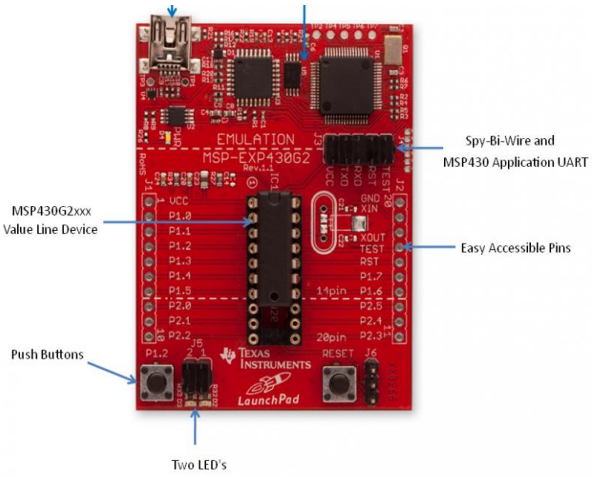
Numerous applications around us work with the integration of realtime input, which then is processed and a decision is taken by a microcontroller or microprocessor. Timely action is a necessity in many of the applications. In this project we demonstrate the feasibility of such scenario in fast paced game. This is also demonstrates how you can utilise a system to process and act in various states of the game/environment.

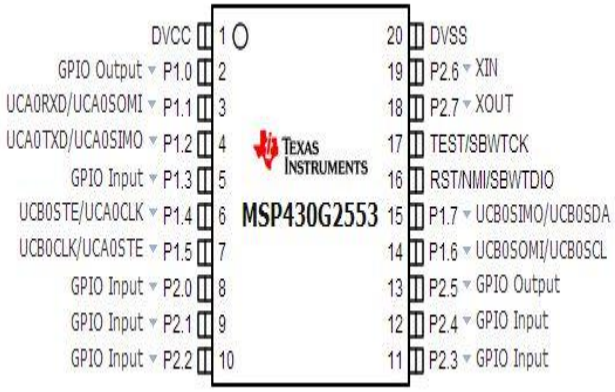
3 Major Components

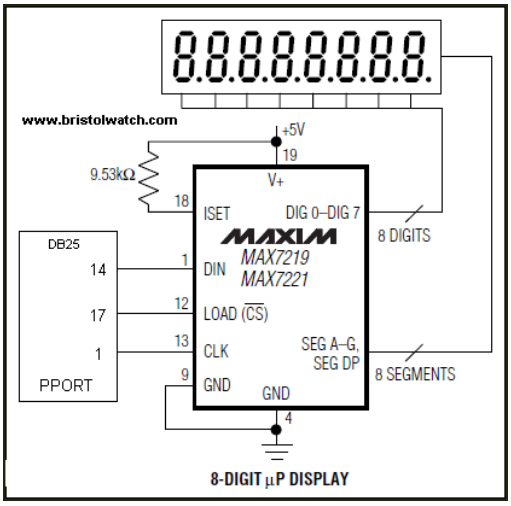
3.1 MSP430

The MSP430 can be used for low powered embedded devices. The current drawn in idle mode can be less than 1 µA. The top CPU speed is 25 MHz. It can be throttled back for lower power consumption. The MSP430 also uses six different low-power modes, which can disable unneeded clocks and CPU. Additionally, the MSP430 is capable of wake-up times below 1 microsecond, allowing the microcontroller to stay in sleep mode longer, minimizing its average current consumption. The device comes in a variety of configurations featuring the usual peripherals:
internal oscillator, timer including PWM, watchdog, USART, SPI, I²C, 10/12/14/16/24-bit ADCs,
and brownoutreset circuitry. Some less usual peripheral options include comparators (that can be used with the timers to do simple ADC), on-chip op-amps for signal conditioning, 12-bit DAC, LCD driver, hardware multiplier, USB, and DMA for ADC results. Apart from some older EPROM(MSP430E3xx) and high volume mask ROM (MSP430Cxxx) versions, all of the devices are in-system programmable via JTAG (full four-wire or Spy-Bi-Wire) or a built in bootstrap loader (BSL) using UART such as RS232, or USB on devices with USB support.

There are, however, limitations that preclude its use in more complex embedded systems. The MSP430 does not have an external memory bus, so it is limited to on-chip memory (up to 512 KB flash memory and 66 KB RAM) which may be too small for applications that require large buffers or data tables. Also, although it has a DMA controller, it is very difficult to use it to move data off the chip due to a lack of a DMA output strobe.[2]





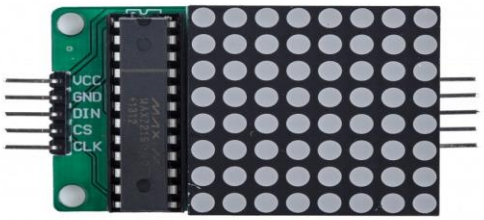


The image on the left shows a detailed diagram of MAX7219

Below is an image of dot matrix display

3.2 MAX7219

The MAX7219/MAX7221 are compact, serial input/output common-cathode display drivers that interface microprocessors (µPs) to 7-segment numeric LED displays of up to 8 digits, bar-graph displays, or 64 individual LEDs. Included on-chip are a BCD code-B decoder, multiplex scan circuitry, segment and digit drivers, and an 8x8 static RAM that stores each digit. Only one external resistor is required to set the segment current for all LEDs. The MAX7221 is compatible with SPI™, QSPI™, and MICROWIRE™, and has slewrate-limited segment drivers to reduce EMI. A convenient 4-wire serial interface connects to all common µPs. Individual digits may be addressed and updated without rewriting the entire display. The MAX7219/MAX7221 also allow the user to select codeB decoding or no-decode for each digit. The devices include a 150µA low-power shutdown mode, analog and digital brightness control, a scanlimit register that allows the user to display from 1 to 8 digits, and a test mode that forces all LEDs on.



4 Objective

The game contains two paddle(player and computer) which is movable to the left and write by one bit and a ball which can travel in both x and y direction constraint to the walls. The ball will only be kept in motion if the player/computer has its paddle directly below the ball.The player can control the paddle using two push buttons which is connected to the microcontroller.

5 Methodology

The input from the buttons is used formulate the next state of the game, which is displayed on the dot matrix display using serially using UART.

5.1 Design

The two push button is connected to the PORT 2 bit 0 and 2, it also have pull up resistors for better sensitivity. The display using had three connection data in, chip select and clock which was connected to P1.2,P1.3,P1.4 respectively. Ground and VCC was also supplied to the display and the buttons.

Push Button

A push button is a momentary or non-latching switch which causes a temporary change in the state of an electrical circuit only while the switch is physically actuated. An automatic mechanism (i.e. a spring) returns the switch to its default position immediately afterwards, restoring the initial circuit condition. There are two types:

A push to make switch allows electricity to flow between its two contacts when held in. When the button is released, the circuit is broken. This type of switch is also known as a Normally Open (NO) Switch

8 x 8 LED

A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. This effect is called electroluminescence. The color of the light (corresponding to the energy of the photons) is determined by the energy required for electrons to cross the band gap of the semiconductor.[1] White light is obtained by using multiple semiconductors or a layer of light-emitting phosphor on the semiconductor device.

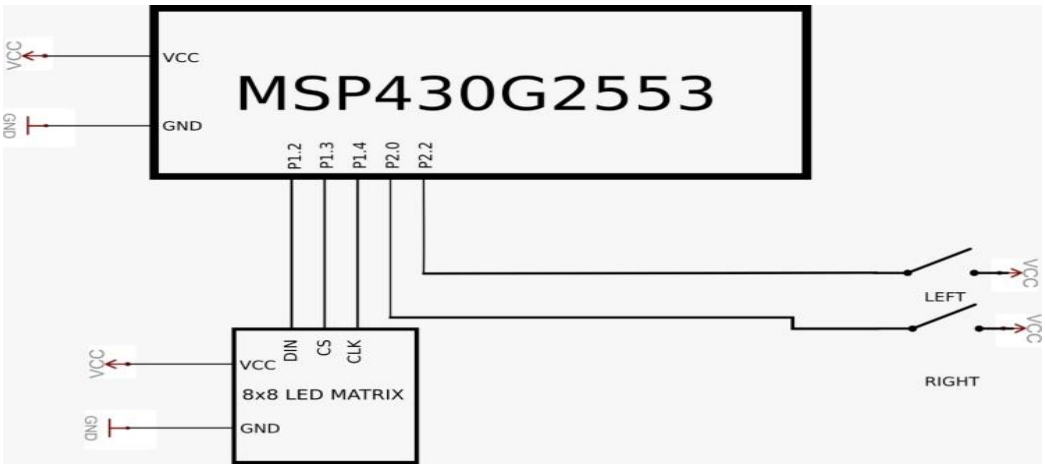
Appearing as practical electronic components in 1962, the earliest LEDs emitted low-intensity infrared light. Infrared LEDs are used in remote-control circuits, such as those used with a wide variety of consumer electronics. The first visible-light LEDs were of low intensity and limited to red. Modern LEDs are available across the visible, ultraviolet, and infrared wavelengths, with high light output.

Early LEDs were often used as indicator lamps, replacing small incandescent bulbs, and in seven-segment displays. Recent developments have produced white-light LEDs suitable for room lighting. LEDs have led to new displays and sensors, while their high switching rates are useful in advanced communications technology.

LEDs have many advantages over incandescent light sources, including lower energy consumption, longer lifetime, improved physical robustness, smaller size, and faster switching. Light-emitting diodes are used in applications as diverse as aviation lighting, automotive headlamps, advertising, general lighting, traffic signals, camera flashes, lighted wallpaper and medical devices.

Unlike a laser, the color of light emitted from an LED is neither coherent nor monochromatic, but the spectrum is narrow with respect to human vision, and functionally monochromatic.

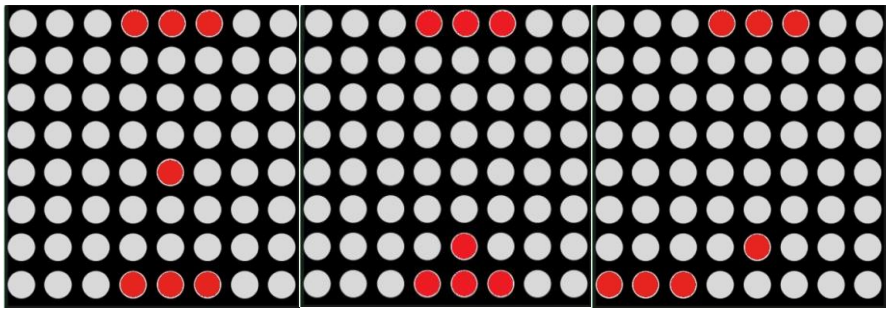
In this project we used 8 x 8 such LEDs to display the game. The LED were driven by the MAX7219.



Schematic of MSP430 and display module,

Inputs.

(a) Describes the initial state, (b) Describes the just before bouncing up (c) A Game Over state.



5.2 Logic

The program contains four fundamental variables for the pong namely

Pong_x ,Pong_y $\in (0,7)$

,x_dir ,y_dir $\in \{1.\text{-}1\}$

And paddle variables

Paddle_loc $\in$ *(0,7)*

And the display registers *disp[n]* contains 8 bytes.

The simplified logic is given below

---

x_dir = 1

y_dir = -1

paddle_loc = 3

While True:

Reset disp

Pong_y = Pong_y + y_dir

Pong_x = Pong_x + x_dir

ON disp[Pong_y] = BIT0 <<(left shift) Pong_x

Update Paddle_loc according to input

If pong_x is 0 or 7 change x_dir to - x_dir

Do the following if pong_y is:

Case 0: Game Over

Case 1:Check if paddle below if so y_dir = -y_dir

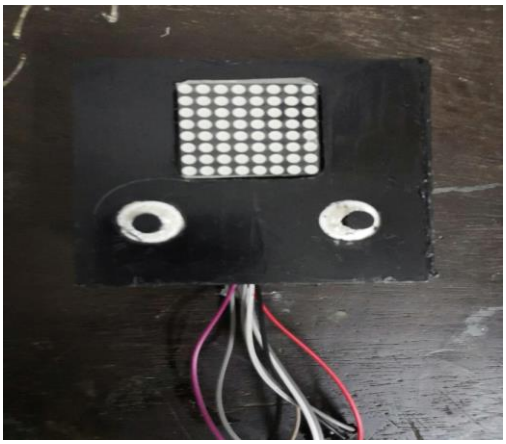
Case 7: Check if paddle if so y_dir = -y_dir

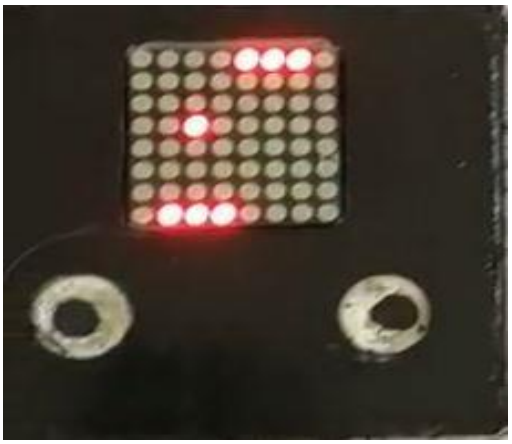
Repeat

---

6 Result

The circuit and logic was implemented, the system shows tactile feedbacks to the given inputs. Different test scenarios were tested and verified the working





7 Conclusion

This project shows how to implement a system which takes feedback from the world or a person and give real time outputs algorithmically.This can be further developed into real life problems accounting different circumstances.

8 Biblography

- MSP430 user guide and manual.
- MAX7219 user guide.
- Driver for 8 x 8 display github/evilbetty.