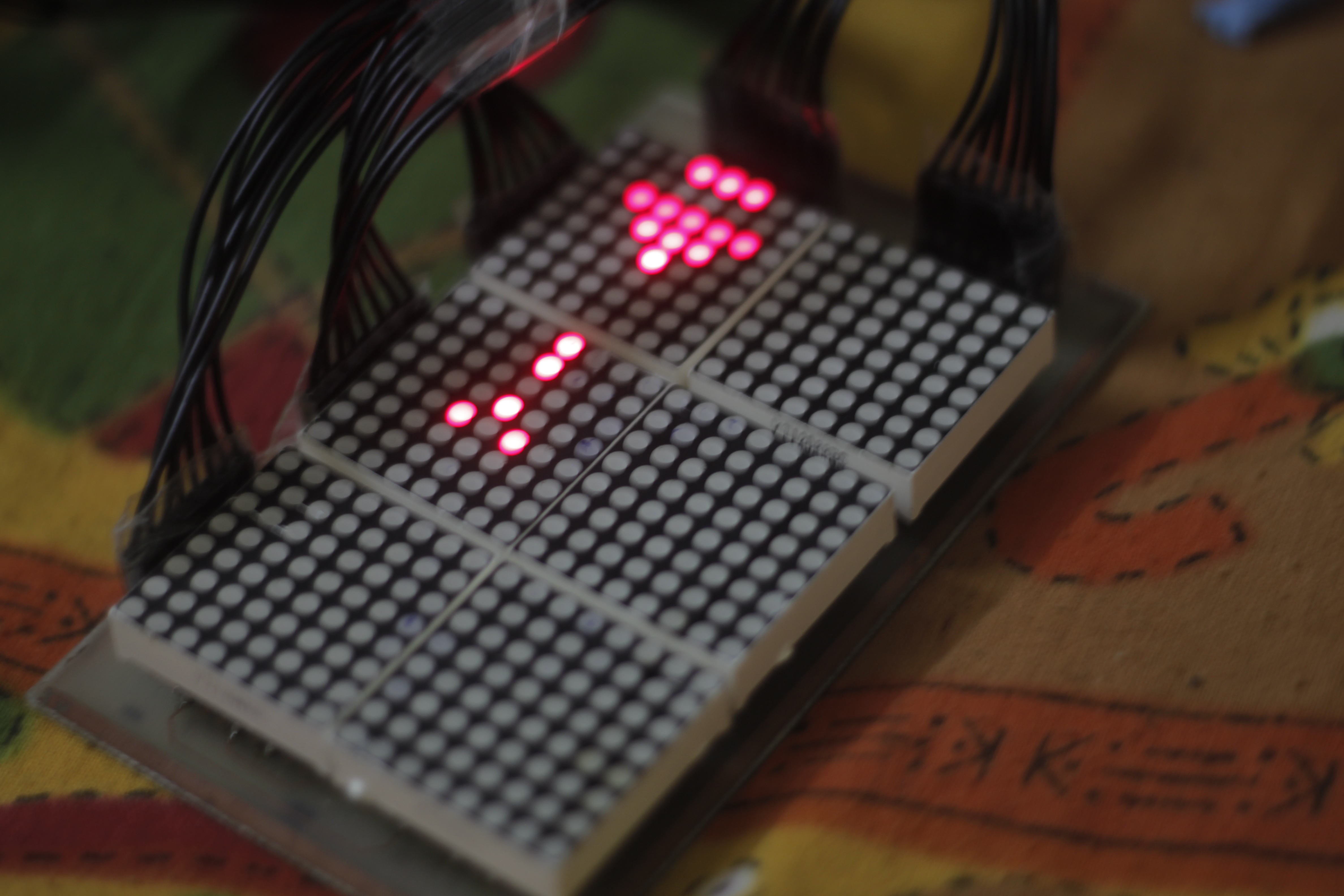
Ashish Goyal | Utkarsh Sharma | April 11, 2014

LED SPACE IMPACT

EE 214 PROJECT



# Abstract:

It is going to be a game where a spaceship has to dodge and finish off space invaders moving towards it at increasing speeds. It is to be made on a set of LED matrices. It draws inspiration from the old mobile game “Space Impact”. The space invaders will move horizontally towards the spaceship. The space ship would be given powers to move vertically and fire bullets back at the invaders. The longer a player can survive and the more number of invaders it shoots, the more points he/she gets. Lots of variations and increasing levels of difficulties can be introduced in the game to make it more interesting. We’ll use the Krypton board and a Verilog code to implement this idea.

## Introduction

It’s the very popular old mobile phone game, where our spaceship, on the left edge of the screen moves past all the invaders charging towards it. We plan to use four to six, 8x8 LED matrices and make specific LEDs glow to represent the Spaceship and invaders and the bullets fired by them, in distinguished shapes. The speed of the invaders moving towards the spaceship will increase with time and they’ll appear at discrete vertical levels randomly. They’ll move towards the spaceship to crash into it and the player will try to dodge or shoot them to score more points, which we can display using Seven-segment display units. We can make the computer play the game as well. And to increase difficulty we can allow vertical movement of invaders as well and we can also make them fire bullets at the spaceship.

## 2. Project description

#### Project Goal:

The goal is to make an enjoyable game. The major challenges we can have is synchronizing the motion of invaders and spaceship over all the LED matrices we use, as we are going to use more than one 8x8 LED matrix. All the LED matrices would be coded appropriately and we’ll have to identify when a bullet hits an invader and when an invader crashes into the spaceship and then change scores accordingly. Also if we want an entirely computer controlled game, we have to code the spaceship appropriately so that it can dodge and/or shoot the invaders properly. We’ll have to identify the position of each object on the LED module and then write appropriate codes in case they interact.

#### Technical Design

#### System level Overview:

Control variables for the led matrix control:

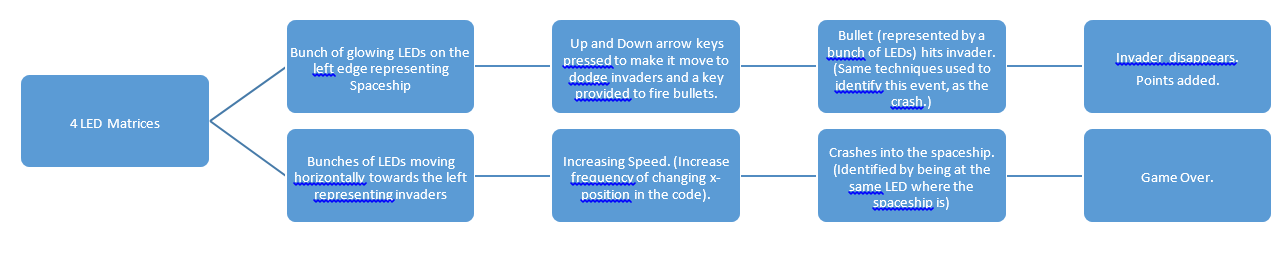
The led screen will be dependent upon the state of the machine as well as the control variables of the entities included in our project.

Following is the state diagram of our machine:

## if bullet

## invader crash

## if invader ship crash happens



Here is the link to a video of working of our project: [LED SPACE IMPACT](http://home.iitb.ac.in/~goyal.ag/resources/ledSpaceImpact.flv)

Some of the photos of our project, click this link: [Space Impact Photos](http://1drv.ms/1jqpSl3)

#### Software Architecture

Strictly speaking, there is a single module in our code which has reduced our work in terms of instantiating the modules and mapping port connections. Though there are lot of always blocks each of which has a completely different function in the code.

1. Always block to generate clock pulses for counterX and counterY in sync. 4 bit counterY runs from 0 to 15 and then increment in counterX is made.
2. A counter 321 which determines the state of the system immediately after its Start Screen state.
3. GameStart, firstScreen, are combinational function of reset button, counter321, gameOver.
4. Change in control variables of led matrix as described in a graphic above is essentially a sequential and finite state machine. The value of these variables depend upon the state of the machine, inputs given and the previous cycle values of these variables. For example, if bullet fired was 0 in previous state and fire button is pressed, bullet fired will be 1 now.
5. Now, since every led state (on or off) is determined by counterX, counterY, and the control variables, its value is a combinational function of the of counterX, counterY and control variables. Example: If a want to show a bullet, then ledState = counterX==bulletX & counterY==bulletY will turn on a particular led which corresponds to position of bullet. Taking or of all such conditions will give a collection of every single thing present on the led matrix.
6. Score is handled by a separate always block, displays on seven segment display. Score is also used to increase the speed of the game, in a way speed is a linear function of score.

#### Hardware interfacing

1. LED MATRIX: Heart and soul of our project. Display interface of our project. The USP of our project. As said earlier, two counters counterX and counterY were used to interface the led matrices. There were 6 of them three in length and two in breadth. They were time multiplexed. In one clock cycle only one of the column was displayed and in subsequent cycles the other ones. Thus giving an effect called persistence of vision. Whether a led will be on or off is combinational function as discussed earlier. Thus this is the rough process:

1. Seven Segment display for score: Another feature that we added in our project, to display the score. We have used two such display units to display a two digit score. One unit has 8 LEDs, for which seven inputs that have to be controlled to be able to display a certain number, one input for judging the orientation and two other input pins which are the cathode/anode for 4 LEDs each. Whether an LED (one of the seven segments) will be turned on or off depends on the score and that can be achieved by asserting the seven inputs high or low as required.

#### 2.2.4 Performance evaluation:

1. Space ship is moving perfectly by pressing up and down buttons.
2. Invaders are coming at increasing speed.
3. Bullets are fired when fire button is pressed
4. Bullet on hitting invaders increases scores. Both disappear too.
5. Invader on hitting spaceship forces game over condition. Score is being constantly displayed on the seven segment display.
6. On holding reset after game over or at any other instant restarts the game and resets the score.
7. Hence the projects meets all the requirements promised at the time of abstract submission.
8. At the top of everything, it’s fun.

## Project Implementation

**Milestone 1:** Design Circuit

**Milestone 2:** Assemble circuit

**Milestone 3**: Complete Verilog code

**Milestone 4:** Complete testing and Debugging

*3.1 GANTT CHART*

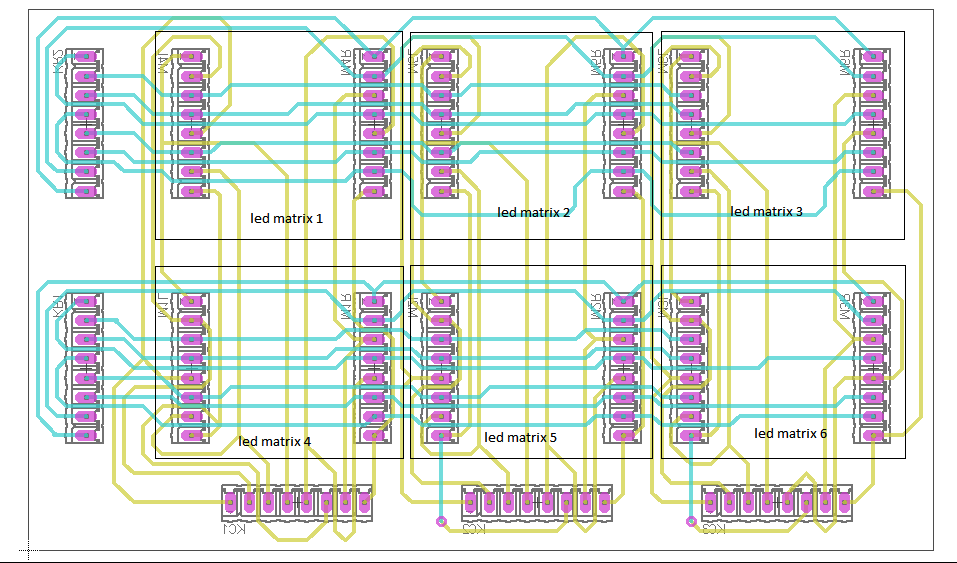
|  |  |  |  |
| --- | --- | --- | --- |
| **Task Name** | **Due Date** | **done by** | **Remark** |
| 1. **Writing Code** |  |  | On time |
| **1.1 Thread 1** | 10 March | Both | Done at the last. |
| **1.2 Thread 2** | 6 March | Ashish | On time |
| **1.3 Thread 3** | 8 March | Utkarsh |  |
| **1.4 Final Compilation** | 17 March | Ashish | Done at the last |
| **2. Circuit Design** | 18 March | Utkarsh | On time |
| **3. Assembling Circuit** |  |  |  |
| **3.1 Procuring Hardware** | 19 March | Both | Delayed by a week |
| **3.2 Keypad Interfacing** | 19 March | Ashish | Done at the start |
| **3.3 LED Matrix Interfacing** | 20 March | Both | Delayed by a week |
| **3.4 Final Circuit Integration** | 24 March | Utkarsh | Two week delay due to unviability of led matrix of required specifications. |
| **4. Testing and Debugging** |  |  |  |
| **4.1 Code testing, debugging and simulation** | 27 March | Ashish | Delayed by a week as circuit integration was delayed. |
| **4.2 Circuit testing and debugging** | 30 March | Utkarsh | Delayed by a week as circuit integration was delayed. |

## 3.2 List of components

* LED matrices – 6
* PCB – PCB lab
* Jumpers – 40
* Seven Segment display units – 2
* Krypton – CPLD

## 3.3 Circuit pcb design

The PCB was designed in order to reduce the number of wires and jumpers that we were going to use. We had decided that 40 inputs (16 rows + 24 columns) were required for our project. So, we designed a PCB, using Eagle, such that we were only required to use 40 jumper wires and rest all the connections were taken care of, on the PCB itself. We then soldered Male and Female bergstrips in order to provide inputs (to LED matrices) and mount the LED matrices on the board. The other 20 connections for the two seven segment display units were made directly.



## Result

We started out on this project knowing a bit about Verilog coding, thanks to our previous lab sessions, and very little knowledge of using the language to make interesting things with the help of various kinds of displays. This project helped us learn all of it.

At the end of this project, we were able to control the display on 6 LED matrices simultaneously. Almost the entire project is based on coding, and our code, on time multiplexing. We learnt how to use clocks for implementing time multiplexing. We also tried our hands at VGA display, after having completed our project on the LED matrix display, but had to cut that short due to lack of time. So the final product that we got after 4 weeks of learning was an interesting game, on an interesting kind of display. All the goals that we wanted to achieve from this project were successfully achieved and the game works just fine.

To summarize:

* We have used 6 LED matrices simultaneously, using time multiplexing.
* We have displayed various elements in the game, like a user-controlled spaceship, along with invaders and bullets.
* We have implemented different events in the game, like bullet-invader crash, and spaceship-invader crash.
* We have also implemented increasing difficulty with the progression of the game.
* We have a well-defined scoring pattern and the score is displayed separately as well.