Open access Journal International Journal of Emerging Trends in Science and Technology

ATmega32 Controlled "Persistence of Vision" Display

Authors

Uday D. Wankar[1], Akhil H. Wankhede[2]

[1]Electrial Engineering Department, Government College of Engineering,
Ballarsha Bypass Road, Chandrapur - 442403
Maharashtra State., India
Email: *udaywankar@gmail.com*

[2]Mechanical Engineering Department, Government College of Engineering,
Ballarsha Bypass Road, Chandrapur - 442403
Maharashtra State., India
Email: *akhilwankhede@gmail.com*

Abstract

This paper explains the project which includes design and fabrication of a display based on Persistence of vision. The objective of the project is to create virtual display in air. A class of display device described as "POV" is one that composes an image by displaying one spatial portion at a time in rapid succession (for example, one column of pixels every few milliseconds). A two-dimensional POV display is often accomplished by means of rapidly moving a single row of LEDs along a linear or circular path. The effect is that the image is perceived as a whole by the viewer as long as the entire path is completed during the visual persistence time of the human eye. A further effect is often to give the illusion of the image floating in mid-air. For building this project, requirement is just a small 40 pin microcontroller, a position encoder, and SMD LEDs.

Keywords: *POV display, persistence of vision, Rotating display, ATMEGA32*

1. Introduction

Our project is a persistence of vision display (POV) that spins 360 degrees horizontally. The purpose of our POV display project is to create a small apparatus that will create a visual using only a small number of LEDs as it spins in a circle. When the LEDs rotate several times around a point in less than a second, the human eye reaches its limit of motion perception and creates an illusion of a continuous image. Therefore, our POV display demonstrates this phenomenon by creating a visual as the LEDs spin rapidly in a circle and the person watching will see one continuous image.

Our POV project includes a total of 48 single color LEDs that are stacked vertically and rotates in a circle on the horizontal plane. The pictures generated by the spinning LEDs are coordinated by an AVR ATMEGA32 microcontroller and programmed using the software AVR STUDIO 4. I R sensor is used in conjunction so that the microcontroller can receive a reference point as to when it should start outputting the visual during each rotation. The white Pease of paper is placed separately from the rotating platform underneath where the I R sensor is located. This way, the I R sensor will pass over the white paper during every rotation. We put the LED board, Atmega32 microcontroller, and a 6V battery (used to power the microcontroller) all onto a single 2x16 inch platform that is mounted to the top of a small 230V Exhaust fan. The Exhaust fan is plugged into the wall socket and can be held or placed firmly on any flat surface for the POV display to function properly.

2. Persistence of Vision Project Design

2.1 Picking LEDs

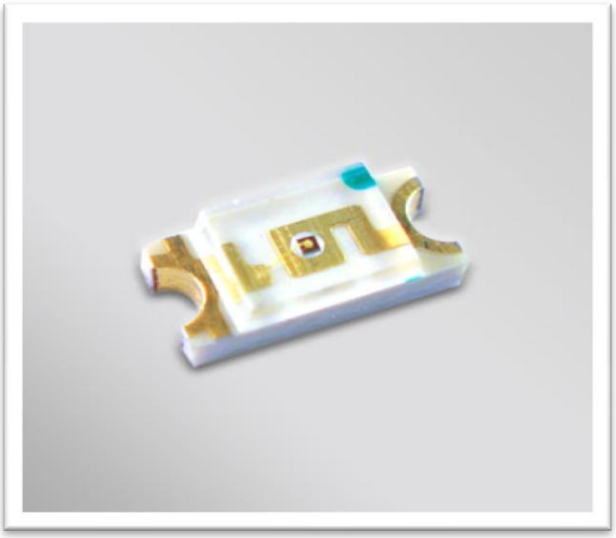


Figure 2.1: SMD LED

The first step to creating our POV display is putting into consideration how many LEDs we can implement and whether to use single color or RGB (red green and blue colored) LEDs. We established that eight LEDs stacked in a column will be enough to properly display images and texts without worries of horizontal and vertical space limits. Since the POV spins around 360 degrees, we can display visuals around that entire circumference. The problem we had to consider was whether to use RGB or single colored LEDs. Single colored LEDs require only an input and ground, whileRGB LEDs require three inputs (one for each color) and a ground connection. Since the Atmega32 microcontroller provides a total of 32 input and outputs, using single color LEDs was the best choice because there were enough ports to connect each of the eight LEDs. On the other hand, RGB LEDs would have required a total of 24 pins (8 green, 8blue, 8 red), which is possible through a multiplexing scheme using the 74HC595 chip connected to the Atmega32 microcontroller. It is also possible to use only four RGB LEDs that will take up 12 slots on the Atmega32 microcontroller. However, we concluded that only four LEDs will make the visual too small. Another possibility is to use a PIC microcontroller that includes 40 pins and a chip programmer to control the RGB LEDs. Since the 74HC595 chip and the PIC programmer were unavailable, we decided to use 48 single colored SMDs and addressing them by led matrix method so that 48 SMD LEDs can be controlled by 16 pins of Atmega32 microcontroller.

2.2 Constructing the LED Board

Soldering everything that is connected to the Atmega32 microcontroller was straightforward. We used eight output pins on the Atmega32 microcontroller and connected each to a 1k ohm resistor that is connected to a single LED and then to the ground pin of the microcontroller. The resistor before the LED is necessary to stabilize the current that it's drawing from the microcontroller. We picked 1k ohm resistors because it made the LED bright enough to be seen under normal room lighting and not too bright in the dark.

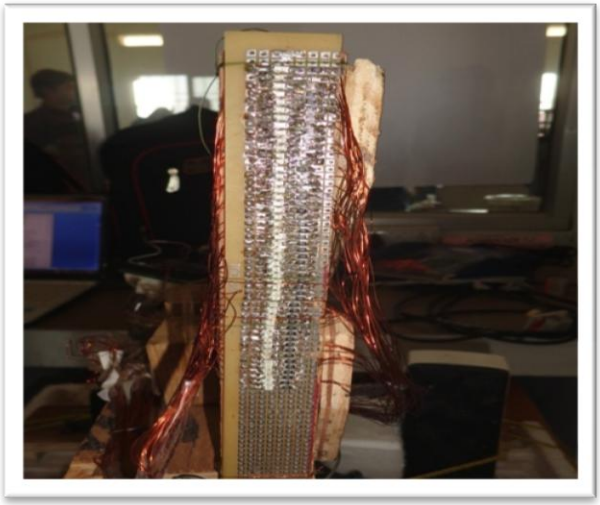


Figure 2.2: LED Board

2.3 Atmega32 Microcontroller

We use Atmega32 Microcontrollerto controller LEDs so that message can be displayed in digital format. Atmega32 is high performance, low-power Atmel 8 bit AVR RISC-based microcontroller 32kb of programmable flash memory. This small sized IC is used, mainly because of its reduced weight. This improves the performance of the display, because reduced weight gives advantage of increased RPM.

Table 2.3: Atmega32 specification

Microcontroller	Atmega32
CPU	8-bit AVR
Operating voltage (V)	4.5 - 5.5
Max operating Freq.(MHz)	16
Max IO pins	32
Flash (Kbytes)	32Kb
EEPROM	1Kb
SRAM	2Kb
DC current per IO pin	40mA

2.4 Interrupter Module

Next was the IR sensor that is placed facing downwards towards where the white piece of paper will be. We included a 10k ohm pull-up resistor from the input to the output so that when the sensor sense white paper, it will act as though it is disconnected. Since the IR sensor only needed about five volts for it to run, we connected a wire from VDD to the 5V output pin on the Atmega32 microcontroller. The output of the IR sensor is connected to one of the pins on the microcontroller and then declared as an input in our software program. This way, whenever the IR sensor is not over the white piece of paper, it will continue to run at 0 volts and the microcontroller will read that input as "LOW".When the white piece of paper passes under the sensor, the sensor will output a "HIGH" signal, telling the microcontroller to start displaying a frame.

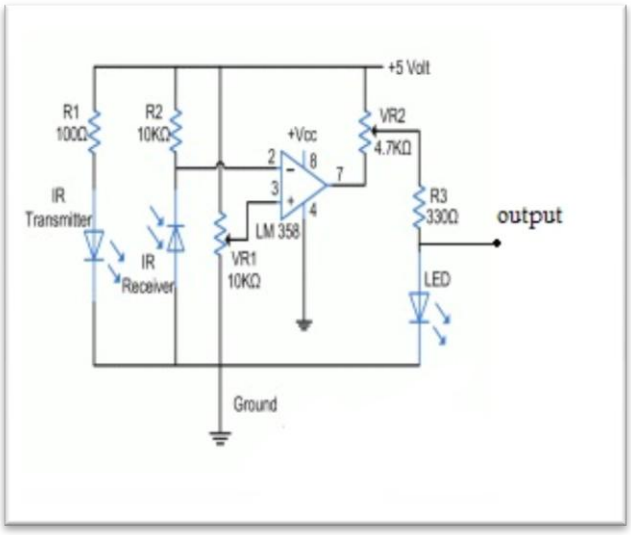


Figure 2.4: IR sensor schematic

2.5 DC Power Supply

One other thing we made ourselves was a battery powered source that is mounted on top of the fan along with the rest of the components. Originally, we planned to use an external power supply that will be continuously touching a PCB board that is connected to the microcontroller. The reason is because we cannot use a regular wire from a power source that is not on the fan to power the microcontroller while it spins, or else the wires will get tangled. The challenge to this method is finding a way for the wires to be always in contact with the PCB board, otherwise, the microcontroller will lose power and cease to function. Since we decided to use a heavy duty fan, it became possible to mount a 6V battery on top of the spinning platform along with the other components to power the microcontroller. The battery is a bit heavy and added weight and imbalance to the platform, so we decided to place it in the exact middle. We also soldered together the battery wires with a round socket plug in head so that is able to connect to the power input of the Atmega32 microcontroller. In terms of power consumption, the Atmega32 microcontroller receives 6V from the battery and provides power to the LEDs and IR sensor, and the small Exhaust fan draws 230V from the power outlet. It is not the most efficient fan, but it works nicely to create a POV display.

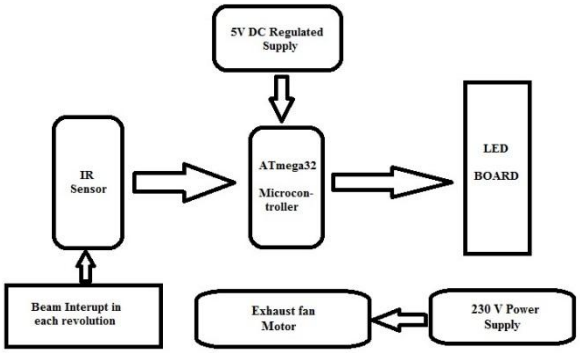


Figure 2: Block diagram.

2.6 Picking the Motor

The next task, that turned out to be more difficult than we had originally thought, was picking out the right motor to be used to spin the display. The first thing that is necessary for the POV display to work

properly is a fast spinning motor that has more than 2000 rotations per minute. A motor having 2000 RPMs is equivalent to 33 full rotations every second. That is more than enough to reach a persistence of vision image that is comfortable to a human eye. We originally used a small one inch diameter motor that had 2300 RPMs, however, the weight and balancing issues made it very hard for the motor to pick up speed. We then decided to use a small table fan and a medium sized standing fan. These fans were heavier duty and can be placed firmly on any surface. The Exhaust fan had about 400 RPMs and produced an ok visual that seemed a little choppy because the platform was not perfectly balanced. So the small desk fan with the component mounted on top of it slowed the rotation speed to about 350 RPMs and that gave us a visual at about 23 frames per second. The medium sized standing fan had a very strong torque and a rotational speed of about 3000 or more RPMs. The problem with this fan was that it rotated too fast (even at the lowest rotation setting) and started throwing the components off the platform even after applying a lot of hot glue and tape. However, it did display a very smooth image at about 50 frames per second. We decided to use the small Exhaust fan instead of the medium standing fan due to safety concerns.

3. Programming the Microcontroller

We used the AVR STUDIO 4 software provided by the Atmel Corporation to program the microcontroller. In the program, we setup PORT A 8 pins by declaring each pin as an output. This will allow the microcontroller to provide power to the 8 LEDs whenever it is programmed to. We then declared pin 1 of PORT D as the input so that the IR sensor can send a signal to the microcontroller whenever it detects a white piece of paper. Next is a loop function that will consecutively loop the program for it display our image very quickly each time the sensor passes over white piece of paper. It uses an “if” and “else” statement to establish when to display the visual. We set it so that if the sensor reports a low signal (when there is less than 2 volts present at the pin), meaning that it just passed over white piece of paper, then it will start displaying the visual very quickly. Otherwise, it will just keep the LEDs off.

The next step is to create a function that will show a set image whenever it is called up in the loop function. We used a binary formatter method by creating our own data that corresponds to each letter in the alphabet and some simple pictures. Since a byte can store an 8-bit unsigned number from 0 to 255, we used that to create an on and off mark for each of our LEDs. For example, if we wanted to display the letter “N”, we would map out the letter using “0” and “1” like the shape below. Where “1” will turn the LED on and “0” keeps the LED off.

B11111111
B00000110
B00001100
B00011000
B00110000
B01100000
B11000000
B11111111

In the function that displays our image, we tell it which image to display and it will go through all of the data for all the columns in each frame for that image. Timer1 will tell the program to create a very short delay (in milliseconds) between each column that is read.This will shrinkorexpandthe width of each image.

After it goes through every column, the program will turn every LED off so that an empty space shows up between each image. In the loop function, we initialized another delay time that will increase or decrease the space between each image.

Figure 2.3: ATmega32 experimental board

3.1 Header Files

To display letters we have to give specific code output to LEDs via microcontroller. This code unique for a single letter or characters. If we have display a word which consist of double letter for example word 'LITTLE' having double T, so we have write code twice for letter T which consume time and memory, making program long.

So we write function of each alphabets with upper and lower case and characters and bind together in header file. It gives following advantages:

1. Makes program easy to understand.
2. Small program size.
3. Repetition of code avoided.
4. Consume less memory.
5. Easier to access each alphabets and characters.

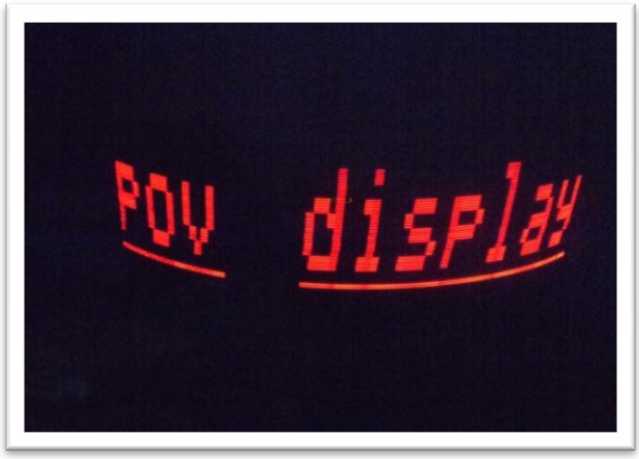
This header files are given below:

UPPERERALPHA.h file for upper case alphabets.

LOWERALPHA.h file for lower case alphabets.

DIGITS.h file for digits and characters.

4. Display of Generated Patterns



5. Problem Faced

5.1 Power Supply:

Power supply to the motor and IC was a bit of problem to us. We first used carbon brushes to give ground from the inner side of motor and VCC through axle, but the carbon brushes lowered the speed of motor. We then give ground from uppers side through carbon brushes, but it doesn't make 100% contacts with motor. So finally we used separate supply for both.

5.2 Balancing Setup:

It's a crucial problem since your setup will be running at high speed. So wobbling of setup will give distorted image. Your setup must be horizontal and center of mass of setup must be at the point about which setup is rotating. To resolve this we added weight through M-seal to the poor side to balance mass on ends.

5.3IR Detection at High Speed:

To call the interrupt at high speed, TSOP wasn't Detecting IR rays. To resolve this we use transistors to increase current through IR led and varies to increase sensitivity of TSOP.

6. Conclusion & Future Aspects

Our project is basically showing 2-D figure from1-D array of LEDs. The display is extremely attractive to look at and gives a sense of being a transparent display. By using a motor of higher RPM, one can achieve more clarity in the display.

1. Using multiple microcontrollers we can view POV of large sizes.
2. LED controllers can be used to change intensity of LEDs and showing HD Images.
3. Rotating a 2-D array of led setup, cool 3-D views can be made which can be viewed from all the directions.

References

1. Ramakant A. Gayakwad, *Op Amp and Linear Integrated Circuit*,2nded, Prentice Hall PTR, 2000
2. Mitchell's modular *LED x-y (horizontally and vertically digitally scanned array system)* was cited in the 29th International Science and Engineering Exposition "book of abstracts", page 97, published by the "Science Service", Washington D.C. May 1978.
3. Coltheart M. "*The persistence of vision*." *Philos Trans R SocLond B Biol Sci.* 1980 Jul 8; 290(1038):57–69.
4. Reference: http://www.engineersgarage.com/tutorials/avr-studio4-working
5. Reference: http://electronics.stackexchange.com/questions/18141/how-to-build-a-pov-display

Author Profile

Uday Wankar pursuing final year Engineering in Electrical Engineering from Government College of Engineering, Chandrapur, R. T. M. N. U.

Akhil Wankhede pursuing final year Engineering in Mechanical Engineering from Government College of Engineering, Chandrapur, R. T. M. N. U.