Application Note

POV Display

AN-CM-239

Abstract

This application note describes how to design a Perception of Vision Display. A design was created to demonstrate how this human sight phenomenon works, using just only a single GreenPAK IC to show a message on the display.

This application note comes complete with design files which can be found in the References section.

Contents

Figures

Tables

1 Terms and Definitions

ASM	Asynchronous state machine
CNT	Counter
DLY	Delay
DM	Dynamic memory
GPO	General-purpose output
I2C	Inter-integrated circuit
LED	Light-emitting diode
LUT	Look up table
MUX	Multiplexer
POV	Persistence of vision
RPM	Revolutions per minute
SR	Set-reset

2 References

For related documents and software, please visit:

GreenPAK™ Programmable Mixed-Signal Products | Renesas

Download our free GreenPAK™ Designer software [1] to open the .gp files [2] and view the proposed circuit design. Use the GreenPAK development tools [3] to freeze the design into your own customized IC in a matter of minutes. Renesas Electronics provides a complete library of application notes [4] featuring design examples as well as explanations of features and blocks within the Renesas IC.

[1] GreenPAK Designer Software, Software Download and User Guide, Renesas Electronics

[2] AN-CM-239 POV Display.gp, GreenPAK Design File, Renesas Electronics

[3] GreenPAK Development Tools, GreenPAK Development Tools Webpage, Renesas Electronics

[4] GreenPAK Application Notes, GreenPAK Application Notes Webpage, Renesas Electronics

[5] SLG46880V, Datasheet, Renesas Electronics

[6] POV Display, video, Renesas Electronics

3 Introduction

The Perception of Vision (POV) or Persistence of Vision (it has several variations) is an interesting human sight phenomenon which occurs when visual perception of an object does not cease despite the object changing position. Human beings see an image in intervals of fractions of seconds; these images are saved into the brain for very short time (an instant). An example of this phenomenon is when you observe a lighting source such as LEDs or bulbs, turned on and spun around. Our vision is tricked into believing the spinning light is actually a continuous circle, much like the continuous circle formed from a rotating propeller on a plane. POV has been used for many years, beginning with the giphoscope, to make different kinds of illusions and animations to our vision; it is frequently used to show messages and animations on displays using LEDs, spinning them in 2D or 3D for different kinds of messages.

The objective of this app note is to design and demonstrate how Perception of Vision works by writing the word “SILEGO” on the display to be built, and give ideas to guide you through the process of making more complex designs in the future. For this project, we used GreenPAK SLG46880, with its socket kit that allows this prototype to be easy connected to all external components using cables. Using the larger GreenPAK to design general purpose POV Displays is very advantageous because of its robust components such as ASM subsystems, which will allow you to print any kind of pattern on the display. This application will show a final result using a SLG46880.

4 Schematics

This POV Display example targets a 2D type shown in Figure 1, which has an array of eleven LEDs (each one with resistors to regulate current) connected directly to different GPO pins on the GreenPAK IC. The circuit is prototyped and soldered into PCB breadboards. The power supply used for the Display is a 9 V 10 A L1022 Alkaline Battery, connected to a voltage regulator circuit using LM7805V that outputs 5 V.

In addition to making the display rotate, a DC Motor is needed with enough strength to move all the control circuitry attached to the customized stand. In this case a 12 V motor was used, connected to a main switch, and an off-the-shelf regulated power supply which outputs different voltage levels through a rotary switch, allowing the motor to rotate at several speeds.

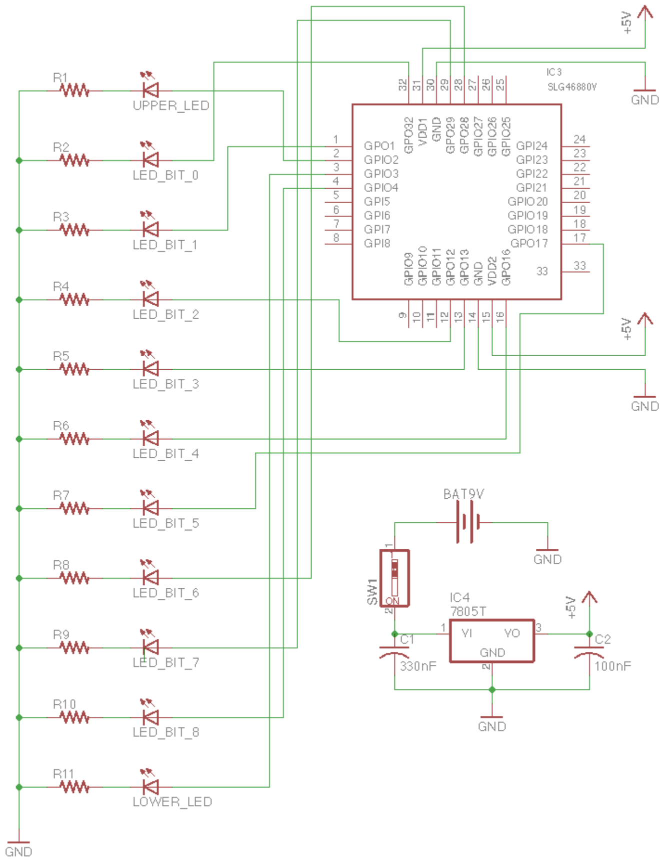


Figure 1: LEDs Control Schematic

5 GreenPAK Design

When designing different kind of messages and animations for a POV Display using GreenPAK, we should know both the tools and limitations of the chip. In this way we can create a proficient design, using the fewest electronics components to achieve the POV display. This design uses the new advantages offered by the SLG46880 GreenPAK IC, with a focus on the Asynchronous State Machine Subsystems component. The SLG46880 ASM Subsystem tool can be more advantageous than previous GreenPAK ASM tools because of its new features, which allow more complex State Machine designs. Some of the relevant ASM Subsystems internal components used are:

- 12-States ASM Macrocell
- Dynamic Memory (DM) Macrocell
- F(1) Computation Macrocell
- State Independent Components

The more state machine macrocells the chip allows to create and configure, the more numerous the design possibilities. Each of the twelve states were used to write different fractions of the word to be shown, turning on/off distinct combinations of the LEDs, some of which were repeated twice or more times, and in some cases the repeated states timing are changed, because the same pattern could be used for different letters at diverse times. States are structured in Table 1.

Table 1: States Structure

States	Belongs to
Silego [0]	“S”
Silego [1]	“S”, “G” & “O”
Silego [2]	“S”
Silego [3]	“I”
Silego [4]	“I”
Silego [5]	“L”
Silego [6]	“E”
Silego [7]	“G” & “O”
Silego [8]	“G”
Silego [9]	“O”
Silego [10]	“L” & “E”
No Message	Blank Spaces

Table 1, shows how each of the existing states in the design are related to the letters in the word “SILEGO”. This correlates to the LED configuration shown in Figure 2.

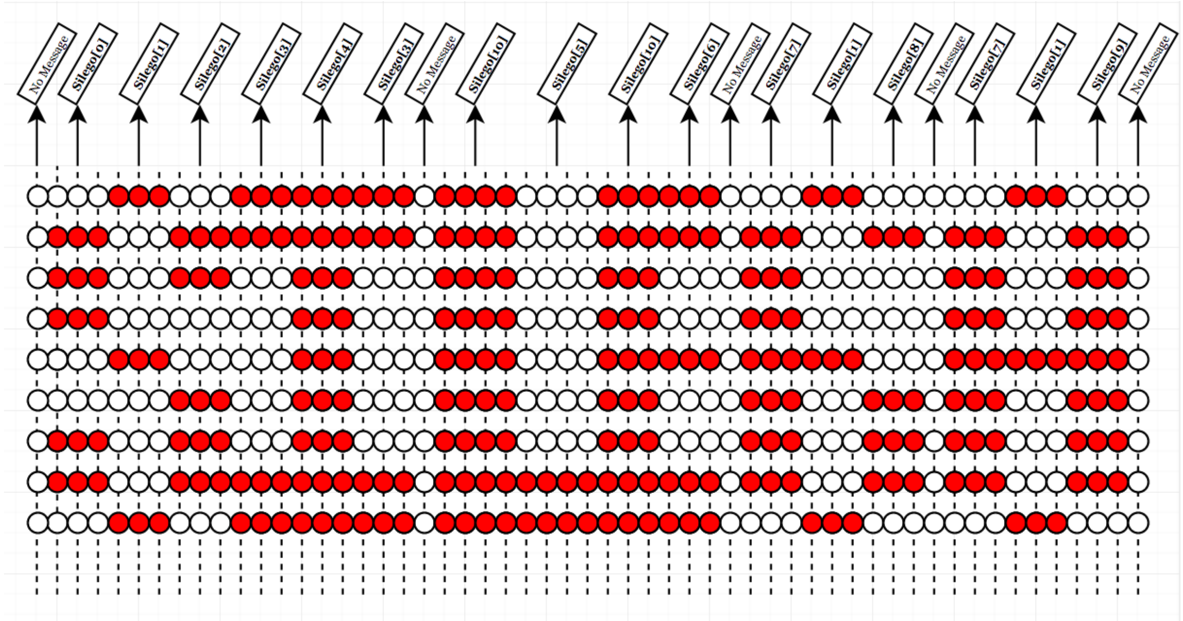


Figure 2: LEDs States and Timing Diagram

As you can observe, all the states together executed at different timing achieves the complete build of the word, Figure 3 shows how states are connected/related. All state’s transitions are in the order of milliseconds, and each of the columns in the diagram of Figure 2 represents one millisecond (1 ms). Some of the states last 3 ms, 4 ms and others, long enough with the minimum speed of the motor used for the video demonstration at approximately 460 RPM.

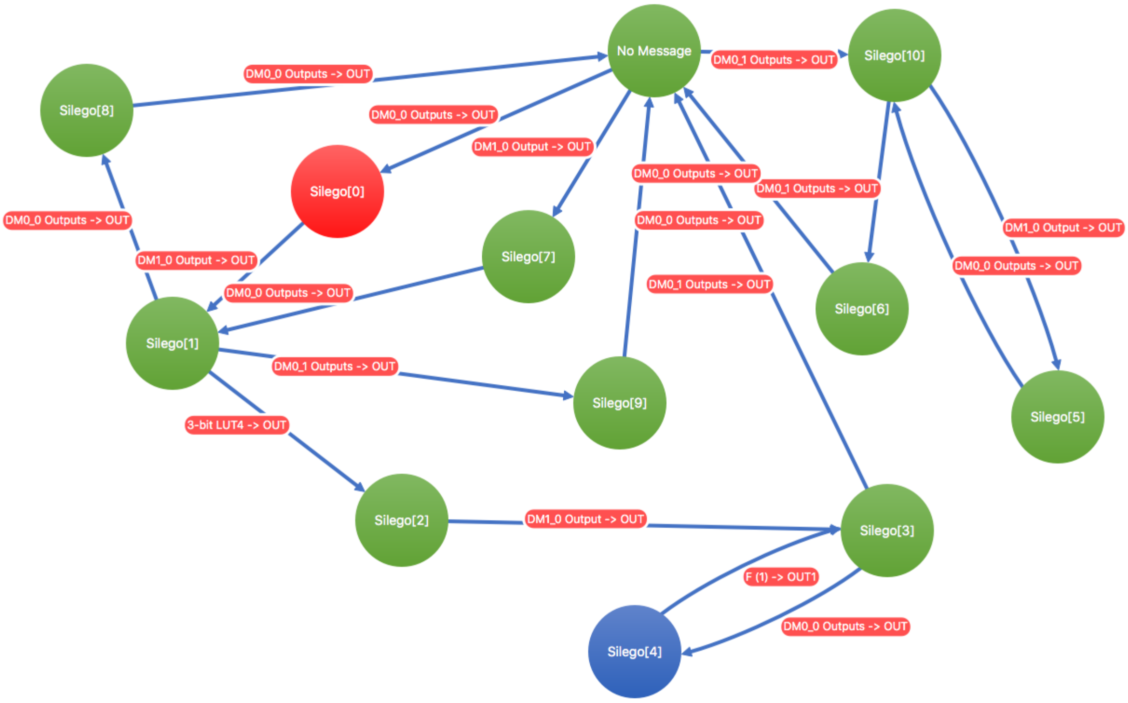


Figure 3: ASM States Diagram

It is important to consider and measure the speed of the motor to know and calculate the timing on a general-purpose design. In this way the message can be synced with the motor speed, thereby visible to the human eye. Another consideration to make the transition of states less imperceptible, and clearer to our vision, is to increase the speed of the motor to more than 1000 RPM, and the states timing set in the order of microseconds so the message could be seen smoothly.

You may be asking yourself, how would you synchronize the speed of the motor with the speed of message or animation? This is accomplished by a few simple formulas. If you have a motor speed of 1000 RPM, to know how long the DC motor takes per revolution in seconds, then:

Frequency = 1000 RPM / 60 = 16.67 Hz

Period = 1 / 16.67 Hz = 59.99 ms

By knowing the period, you know how long the motor takes in a turn. If you want to print a message like “Hello World”, once you know the period of each turn, it is just matter of how big do you want the message to be on the display. To print the desired message at the desired size, follow this rule of thumb:

If, for example, you wish that the message covers 40 % of the display’s space, then:

Message Size = (Period * 40 %) / 100 % = (59.99 ms * 40 %) / 100 % = 24 ms

That means the message will be shown in 24 ms for each turn, so the blank space or the rest of the space in a turn (if you are not showing something after the message), should be:

Blank Space = Period – Message Size = 59.99 ms - 24 ms = 35.99 ms

Finally, if you need to show the message at that 40% of the period, you need to know how many states and transitions the message will need to write the expected message, for example if the message has twenty (20) transitions, then:

Single State Period = Message Size / 20 = 24 ms / 20 = 1.2 ms

So each state should last 1.2 ms to display the message correctly. Of course, you would notice that most of the first designs are not perfect, so you may be changing some parameters during physical testing to improve the design.

We used Dynamic Memory (DM) Macrocells to facilitate state transitions. Two of the four DM blocks have matrix connections so that they can interact with blocks outside the ASM subsystem. Each DM Macrocell can have up to 6 different configurations that can be used in different states.

DM blocks are used in this design to trigger the ASM to transition from one state to another. For example the Silego [3] state is repeated twice over the transitions; it needs to write the beginning and the end of the uppercase “I” letter which has the same pattern, but it first needs to go to Silego [4] to write the pattern of the middle of uppercase “I”, and then when Silego [3] is executed for a second time, it needs to go to No Message state, continuing the rest of transitions. How is it possible to prevent Silego [3] from falling into an infinite loop with Silego [4]? It’s simple, there are some LUTs configured as SR Flip Flops that tells Silego [3] to not choose Silego [4] over and over again, but choose No Message state on the second time.

Using SR Flip Flops to prevent infinite loops when any of the states are repeated is a great way to solve this problem, and only requires a 3-bit LUT configured as shown in Figure 4 and Figure 5. This process happens at the same time as the ASM output makes Silego [3] to go to Silego [4], so the next time the state machine executes Silego [3], it will be notified to choose No Message state to continue the process.

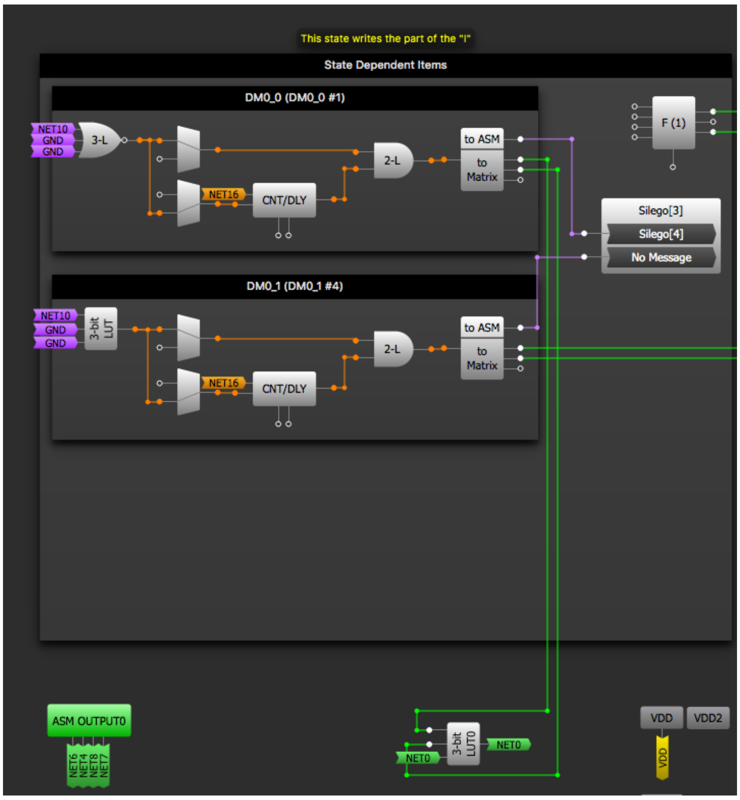
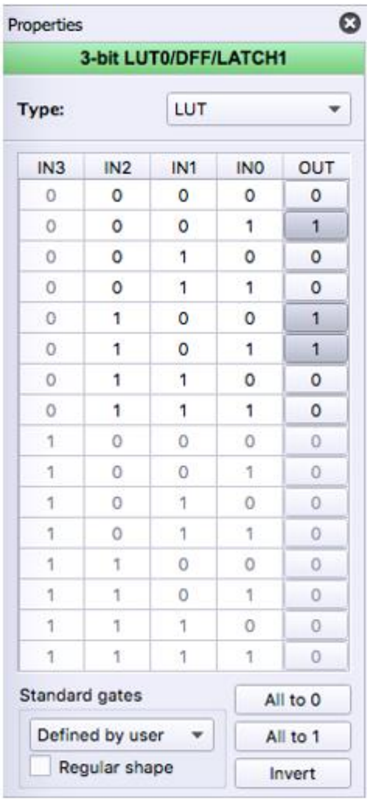


Figure 4: Silego [3] State



Properties

3-bit LUT0/DFF/LATCH1

Type: LUT

IN3	IN2	IN1	IN0	OUT
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Standard gates: Defined by user

□ Regular shape

Figure 5: LUT0 Properties

Another ASM block that was helpful for this project is the F(1) Computational Macrocell. The F(1) can perform a list of specific commands to read, store, process and output desired data. It is able to manipulate 1 bit at a time. In this project the F(1) block was used to read, delay and output bits to control some LUTs and enable states (such as in Silego [1] to enable Silego [2]).

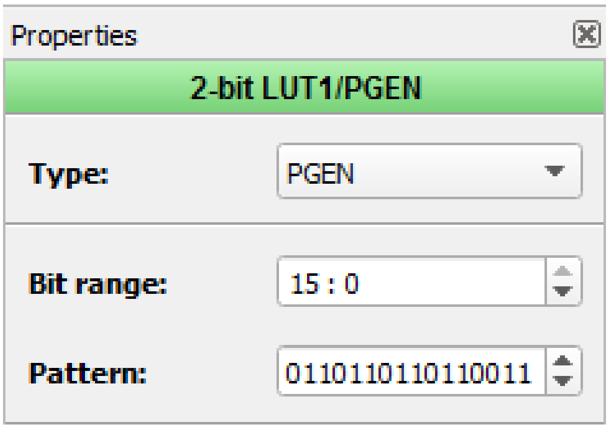
The table in Figure 1 explains how each of the LEDs are addressed to the GPO pins of the GreenPAK; the associated physical pins are addressed from the ASM Output RAM in the matrix, as it is shown in Table 2.

Table 2: Pin Address to ASM Outputs

Pin Name	Pin Number	ASM OUTPUT 0	ASM OUTPUT 1
LED_BIT_0	32	-	OUT 0
LED_BIT_1	1	-	OUT 1
LED_BIT_2	12	-	OUT 2
LED_BIT_3	13	-	OUT 3
LED_BIT_4	16	OUT 1	-
LED_BIT_5	17	-	OUT 5
LED_BIT_6	28	-	OUT 6
LED_BIT_7	29	-	OUT 7
LED_BIT_8	4	OUT 0	-

As you can see in Table 2, each pin of the chip was addressed to distinct ASM outputs; ASM OUTPUT 1 has eight (8) outputs all used directly connected to external GPOs except OUT 4. ASM OUTPUT 0 has four (4) outputs where OUT 0 and OUT 1 are directly connected to PIN 4 and PIN 16 respectively; OUT 2 is used to reset LUT5 and LUT6 at Silego [5] and Silego [9] states and finally OUT 3 is used to set LUT6 at Silego [4] and Silego [7]. ASM nRESET is not toggled in this design so it is just forced to HIGH being connected to VDD.

Upper and Lower LEDs were added to this project to make additional animation while “SILEGO” is being displayed. This animation is about a few lines that are circling over time with the motor’s movement. These lines are white LEDs, while the ones used to write the letters are red. To achieve this animation, we used the GreenPAK’s PGEN and CNT0. The PGEN is a pattern generator which will output the next bit in its array at every clock edge. We divided the motor’s turn period into 16 sections, and the result was set to the output period of CNT0. The pattern programmed into the PGEN is shown in Figure 6.



2-bit LUT1/PGEN

Type: PGEN

Bit range: 15 : 0

Pattern: 0110110110110011

Figure 6: PGEN Properties

6 Results

To test the design, we connected the SLG46880's socket to the PCB with a ribbon cable. Two external boards were connected to the circuit, one of which contained the voltage regulator and the other which contained the LED array. To start displaying the message for demonstration, we turned on the logic circuit which is controlled by the GreenPAK, and then turned on the DC motor. The speed may need to be adjusted for proper synchronization. The final result is shown in Figure 7. There is also an associated video with this application note.

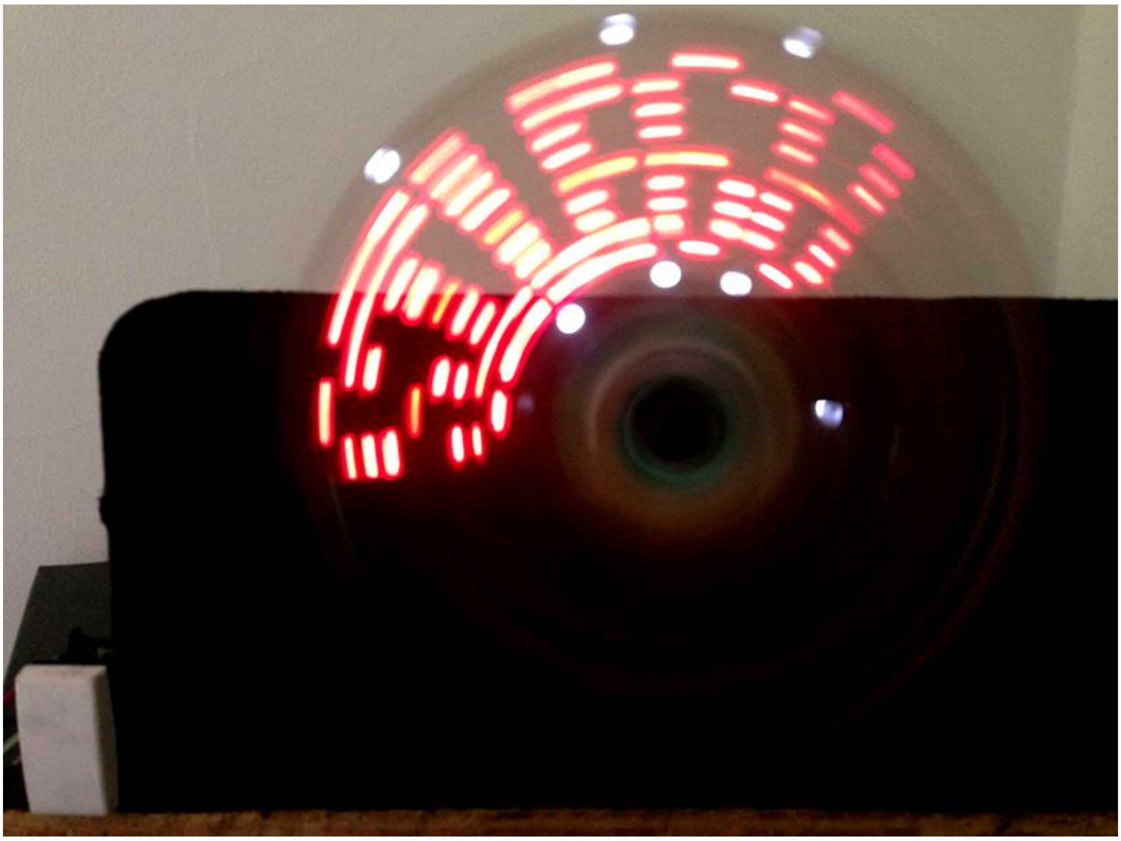


Figure 7: POV Display Output

7 Conclusion

The Perception of Vision Display presented in this project was designed using GreenPAK SLG46880 as the main controller. We demonstrated that the design works by writing the word "SILEGO" using LEDs.

Some improvements that could be made to the design include:

- Using multiple GreenPAKs to increase the amount of states possibilities to print a longer message or animation.
- Add more LEDs to the array. It may be helpful to use surface-mount LEDs rather than through-hole LEDs to decrease the mass of the spinning arm.
- Including a microcontroller could allow you to change the message displayed by using I^{2}C commands to reconfigure the GreenPAK design. This could be used to create a digital clock display that updates the digits to display the time accurately.

Revision History

Revision	Date	Description
1.0	21-Mar-2018	Initial Version

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.