ANTEPROYECTO DE “MÓDULO DE AUTOMATIZACIÓN DE UN SISTEMA DE BLACK-OUT”

Integrante 1: Natanael Robaina

Integrante 1: natanaelrobaina@impatrq.com

Integrante 2: Vogel Tomas

Integrante 2: tomasvogel@impatrq.com

Integrante 3: Rodriguez Juan Pablo

Integrante 3: juanpablorodriguez@impatrq.com

Integrante 4: Seta Vicente

Integrante 4: vicenteseta@impatrq.com

Integrante 5: Sturla Luca

Integrante 5: lucasturla@impatrq.com

1. RESULTADO FINAL

¿Alguna vez se preguntó por qué debe molestarse en bajar o subir su black-out? Sencillamente porque este sistema no había sido creado aún.

Logramos diseñar un sistema automatizado de black-out. Esto significa que nuestro sistema es capaz de decidir por sus propios medios si el black-out debe bajar, subir o parar. ¿Cómo lo hace? Lo hace en función a la luz que el sistema recibe. Cuando la luz es mayor al umbral establecido por el usuario, el black-out baja; y cuando la luz recibida es menor, el black-out sube.

Con el fin de mostrar con facilidad el funcionamiento de nuestro sistema, llevamos a cabo el diseño de una habitación con 2 ventanas, realizada a escala, a las cuales les instalamos el sistema automatizado de black-out. Vale la pena destacar que en cada una de las ventanas se instaló exactamente el mismo sistema sin ninguna modificación.

Lo que resulta interesante de nuestro proyecto es que, por las características del mismo, es posible adaptarlo a cualquier ventana cualesquiera sean sus dimensiones.

2. LISTA DE MATERIALES UTILIZADOS

- 2 LDR (uno por ventana)
- 4 sensores de luz infrarrojos (2 por ventana)
- 2 motores paso a paso 28BYJ 48 con sus respectivos módulos ULN2003 A
- Raspberry PI Pico
- 2 Resistencias de 10 Kohm
- Cables
- Placa PCB 7cm x 7 cm
- Fuente 5V
- Impresiones 3D(PLA)
- madera laminada
- Tela

3. FUNCIONAMIENTO

El sistema consta de distintas partes (software, hardware, diseño) que diferenciaremos más adelante, sin embargo, el correcto funcionamiento se logra con la comunicación exitosa entre cada una de las partes.

A través de la fotorresistencia utilizada como sensor de luz, se censan los datos que contienen la información con la cantidad de luz que está recibiendo el sistema. Estos datos no son más que diferencias de potencial en el LDR. La raspberry Pi Pico mide y compara estas diferencias de potencial con lo que llamamos umbral de luz (como ya mencionamos el usuario es capaz de establecer dicho umbral).

Cuando este umbral es superado se darán los procesos necesarios en la Raspberry para que el motor paso a paso se accione, en el caso de nuestra maqueta, en sentido anti horario

para que el black-out baje. En este punto, entra en juego los sensores de luz infrarroja. Se coloca uno por encima del black-out y otro por debajo del mismo. Estos cumplen la función de establecer los límites superior e inferior del recorrido del black-out. De esta forma, cuando el sensor de luz infrarroja ubicado en la parte inferior detecte la presencia de la tela blanca, la luz infrarroja rebotará en la misma y este será el parámetro que indicará que el black-out debe parar. El black-out habría llegado al límite inferior de su recorrido.

Por otro lado, cuando el umbral no es superado, y además el sensor de luz infrarroja en la parte superior detecte la presencia del black-out (por el rebote de la luz infrarroja), se accionará el motor paso a paso en sentido contrario haciendo que el black-out suba hasta el momento en cual el sensor de luz infrarroja deje de detectar la tela blanca (la luz infrarroja no rebotará), y de vuelta este parámetro actúa como un STOP, indicándole al motor que llegó al límite superior y que debe frenar.

3.1 DETALLES TÉCNICOS

Si bien el LDR varía su resistencia en función a la luz que recibe, el Analog-To-Digital Converter de la Raspberry PI Pico mide tensión, por ello le conectamos una resistencia de 10 Kohm en serie a nuestro LDR y alimentamos con 3.3V.

Recapitulando, a mayor luz menor resistencia, por lo tanto, menor voltaje en nuestro LDR. Con esta especificación queremos dejar en claro que realmente para que nuestro black-out baje (motor paso a paso se accione en sentido anti horario), lo que sucede realmente es que la tensión en el LDR debe ser menor a la establecida en el umbral y viceversa.

El código de nuestro sistema es el que le indica a la Raspberry cuando o bajo qué condiciones se deben dar las órdenes de ejecución de cada uno de los procesos mencionados en el funcionamiento. Este código fue diseñado en lenguaje C.

4. ETAPAS

4.1 ANTE PROYECTO

En esta primera etapa, planteamos la idea buscando encontrar el diseño de un sistema que sea útil y práctico en cualquiera de nuestras casas. Como era de esperarse han habido modificaciones, cambios, problemas y soluciones a lo que sería el ante proyecto propuesto en abril de este año.

Si bien la idea original era la de realizar un sistema automatizado de persianas, al poco tiempo nos dimos cuenta de que las persianas como tales, se instalan para tener privacidad y realmente es uno el que decide en que momento abrir la persiana o bajarla dependiendo de lo que se desee en el momento. No obstante, la principal función del

black-out es la de evitar por completo el paso de la luz solar, es por ello que avanzamos por este lado, automatizando este proceso.

Otra de nuestras ideas propuestas en el ante proyecto fue la de utilizar un módulo bluetooth mediante el cual desde nuestro celular con una aplicación podríamos controlar para que el black-out baje o suba. Lamentablemente no pudimos llevar adelante esta idea, aunque realmente nos resulta interesante para llevarla a cabo el año que viene.

Mientras que en el ante proyecto comentamos que realizaríamos 3 entradas de luz, a la hora de llevarlo a la práctica realizamos 2 entradas de luz debido a que los costos de materiales (tanto componentes electrónicos como impresiones 3D) se elevaría demasiado.

4.2 SOFTWARE

Logramos diseñar un código en C funcional para nuestro proyecto. Fue de las etapas más complejas y sin dudas la más larga.

El software es lo que nos permite atribuirnos el hecho de que nuestro proyecto se adapta a cualquier tamaño de ventana, ya que en el mismo no se utilizaron parámetros de medidas en lo absoluto.

En el proceso de diseño del código aprendimos muchísimas cosas, destaco por ejemplo la programación de un motor paso a paso en ambos sentidos para que sus bobinas se polaricen de una u otra forma; la programación de sensores de luz infrarroja para los cuales debimos crear variables booleanas en donde se guarde su información; el entendimiento del funcionamiento del LDR como sensor de luz; y mismo el diseño de la lógica para que todo funcione correctamente.

4.3 HARDWARE

El hardware formó parte las etapas finales del proyecto y el mismo se basó en: conexión de circuitos electrónicos y diseño de maqueta.

La maqueta es un prisma en representación de una habitación cualquiera, al que 2 de sus paredes se le añadieron ventanas. Dichas 2 paredes y un conjunto de piezas de menor tamaño, cuyo montaje permite que el black-out baje y suba, fueron diseñadas a medida en la aplicación Blender y posteriormente impresas en PLC con una impresora 3D. El resto de las piezas que forman la maqueta, como lo son las 2 paredes restantes, el piso y el techo, fueron realizados con madera. Esto principalmente con el fin de no añadir más costos innecesarios de impresiones 3D.

Por otra parte, el conexionado entre los componentes electrónicos se llevó a cabio mediante cables macho-hembra. En nuestra

PCB, se soldó la Raspberry PI Pico y de esa misma se soldaron todos los cables necesarios a los determinados pines utilizados. La alimentación de cada componente se realizó con la ayuda de una Protoboard alimentada con 5V, con los que fue suficiente para la alimentación de todo el circuito.

5. PROBLEMÁTICAS

En las etapas de desarrollo utilizamos una Raspberry PI Pico prestada la cual se terminó dañando por el uso intensivo de la misma. Más adelante tuvimos que comprar una Raspberry PI Pico para la muestra.

El primer error que notamos desarrollando el codigo fue cuando nos hayamos que al ejecutarlo el Motor no realizaba la función girar horario ni girar anti horario, mas tarde no dimos cuenta de que el problema fue no tener en cuenta la caja reductora del Motor de1/64(esto significa q para una vuelta del rotor se tienen q ejecutar un total de 4096 pasos), conociendo esta propiad se disminuyo el delay en la función programada q se le asigno al motor y este comezó a girar notablemente.

Como mencionamos en la descripción del hardware, diseñamos en Blender un conjunto de piezas de menor tamaño exclusivas para el armado del black-out. Una de estas piezas era un cilindro en el que uno de sus extremos se apoya sobre el eje del motor y el otro extremo sobre una plataforma (también diseñada e impresa en 3D). Resultó que hubo un error de medición y el largo del cilindro era menor que el largo de la ventana, haciendo que por los costados de la misma, siempre entre algo de luz a la parte interior. Para solucionarlo, cortamos maderas de una medida en específico y las colocamos por donde entraba luz. (En los costados de ambas ventanas).

Otro problema que surgió durante el diseño del código fue que para que se ejecute la orden de frenar el motor, básicamente utilizamos un parámetro que varía dependiendo de la ventana. Este parámetro eran las vueltas completas que realizaba el motor. Esto no podía ser así ya que dependiendo de la ventana se debería modificar el código, habría que incluso hacer cuentas para saber cuántas vueltas daría y realmente ni siquiera sería preciso o práctico. Es por ello que surgió la idea de utilizar sensores de luz infrarroja que serían utilizados como límites superior e inferior e indican cuando el motor debe frenar.

El error más importante que tuvimos fue el que nos dimos cuenta el propio día de la exposición. Resulta que no habíamos terminado de comprender la forma en que el ADC de la Raspberry funciona. Conectamos el LDR a uno de los pines de ADC, pero nunca iba a funcionar correctamente el sistema ya que no habíamos

conectado una resistencia al LDR ni estábamos alimentando dicho circuito. Por lo tanto, las tensiones que media el ADC hasta antes de resolver el problema no eran más que ruido eléctrico. De hecho, recuerdo perfectamente como cuando tocábamos el LDR con el dedo, nuestro proyecto comenzaba a funcionar. Como acabo de explicar, a través del diseño de un circuito simple con una resistencia conectada a 3.3V se solucionó este grave error.

Siendo crítico, considero que la falta de experiencia nos jugó bastante en contra, pero de eso se trata, de analizar los errores y problemas y aprender de ellos. Otro factor que debemos mejorar para el futuro es la organización de las conexiones entre cada componente.

6. DIVISIÓN DE TAREAS

Finalmente, fue un trabajo hecho en grupo, tuvimos que aprender a trabajar en equipo lo cual no siempre resulta sencillo. No llevamos a cabo una división exacta de tareas ya que entendimos que todos deben contribuir en todo y en lo que se necesite. Sin embargo, si deberíamos establecer una lista de que parte se encargó cada uno seria:

- Seta Vicente: software
- Robaina Natanael: soldaduras
- Vogel Tomas: montaje de hardware
- Rodriguez Juan Pablo: montaje de hardware
- Sturla Luca: diseño en Blender del hardware para las impresiones 3D

7. REFERENCIAS

Nuestras referencias fueron los consejos de distintos profesores y alguna recomendación de compañeros que ya trabajaron anteriormente con alguno de los componentes utilizados.

Especiales agradecimientos

Facundo Arguello y Fabrizio Carlassara