UNIVERSIDAD de VALLADOLID

ESCUELA de INGENIERÍAS INDUSTRIALES

INGENIERO TÉCNICO INDUSTRIAL, ESPECIALIDAD EN ELECTRÓNICA

PROYECTO FIN DE CARRERA

SEGUIDOR LUMINICO

CON DOS

GRADOS DE LIBERTAD

Autores:

METE BOLOPO, PASTOR

CHICHAVA, JOAO ANTONIO

Tutor:

F. JAVIER GARCIA RUIZ

Ingeniería de Sistemas
Y
Automática

MAYO – 2013

Índice

1. INTRODUCCION

Un seguidor lumínico es un dispositivo mecánico capaz seguir un haz de luz
Existe una gran variedad de seguidores de luz en el mercado. La clasificación de éstos se puede hacer atendiendo a varios factores tales como el tipo de plataforma, el número de ejes o el tipo de unidad de control.

2. OBJETIVOS DEL PROYECTO

El objetivo de este proyecto, es construir y programar un seguidor lumínico con dos grados de libertad, que sea capaz seguir un haz de luz.

Para lograr este objetivo el sistema constará de una superficie en la que se posicionan 4 fotocélulas divididas en grupos de 2 apoyadas sobre una plataforma mecánica con 2 grados de libertad. Se deberán detectar umbrales de luz comparando los valores de dos fotocélulas de cada grupo a través del microcontrolador Arduino UNO. Como resultado de dicha comparación se generará un movimiento de giro o de inclinación a través de dos servomotores de forma que la superficie se mantenga perpendicular al haz lumínico y a través de una pantalla LCD en cada momento conoceremos el ángulo de rotación y el de inclinación de la superficie.

3. FUNCIONAMIENTO

Como sensor de luz usamos una LDR, Como esta funciona como una resistencia variable de manera que, cuanta más cantidad de luz reciba, menor será su resistencia

Vamos aprovechar esa característica de la LDR, si añadimos una resistencia en serie podemos utilizar la LDR para hacer el ya conocido divisor de tensión.

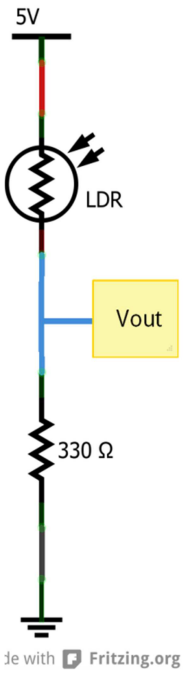


Figura 1. Modo de conexión de una LDR

Como muestra la figura 1, utilizamos la LDR como **resistencia superior** del divisor de tensión, así nos dará la tensión máxima cuando esté a plena luz, ya que se comportará prácticamente como un cortocircuito, con una resistencia de 50Ω o 100Ω.

A plena oscuridad nos dará la tensión mínima al aumentar considerablemente la resistencia de la LDR entorna a 1MΩ. Nuestra fotorresistencia configurada de este modo, nos va a dar **0v cuando esté completamente a oscuras**, y **+5v cuando esté completamente iluminada**, situaciones que se pueden conseguir dependiendo del entorno en el que trabaje, y por otra parte, el rango de 0v a 5v es pasado a un rango de **0 a 1023** que es como lo leerá el Arduino. Para poder llevar a cabo los movimientos de inclinación y rotación necesitamos de cuatro fotorresistencias que las dividimos en dos grupos de dos fotorresistencias cada grupo.

Grupo1: Ldr1 y Ldr2 →Rotación

Grupo2: Ldr3 y Ldr4 →Inclinación

Cada divisor de tensión de la figura 2 (V_0, V_1, V_2, V_3) la conectamos una entrada analógica del arduino (A_0, A_1, A_2, A_3) respectivamente.

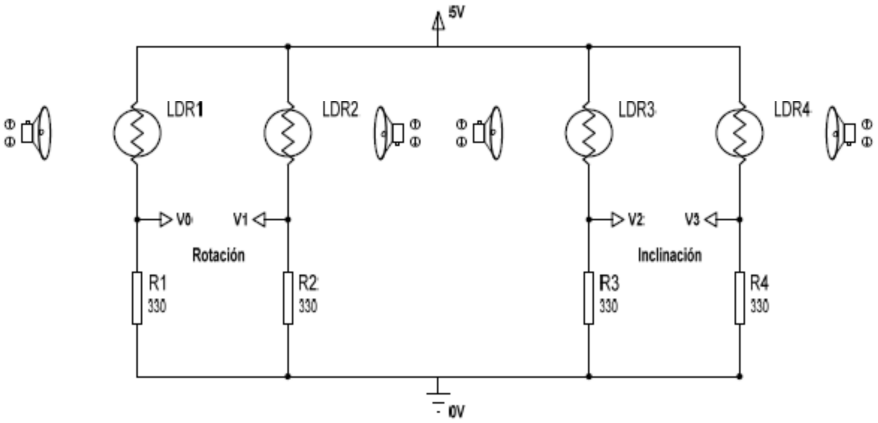


Figura 2. Divisor de tensión de las LDRs

Sensores	**Divisores de tensión**	**Entradas analógicas**
Ldr1 (LdrLeft)	v_0	A_0
Ldr2 (LdrRight)	v_1	A_1
Ldr3 (LdrTop)	v_2	A_2
Ldr4 (LdrDown)	v_3	A_3

Tabla 1. Pines sensores-divisores de tensión- entradas analógicas Arduino

Como hemos visto si la LDR está o no en presencia de luz su valor interpretado por el Arduino estará en un rango de 0 a 1023.

Tensión Analógica (divisor de tensión)	Tensión Digital (placa Arduino)	Intensidad de luz
0	0	Oscuridad
5	1023	Muchísima luz

Tabla 2. Señal analógica y digital en función de la intensidad de luz

El microcontrolador del Arduino siguiendo el código de programa se encargará de comparar las tensiones de los divisores de cada grupo.

Por tanto el microcontrolador Arduino emitirá una orden que culminará con un movimiento de inclinación y/o rotación según proceda y el Display LCD nos mostrará los grados inclinados y rotados.

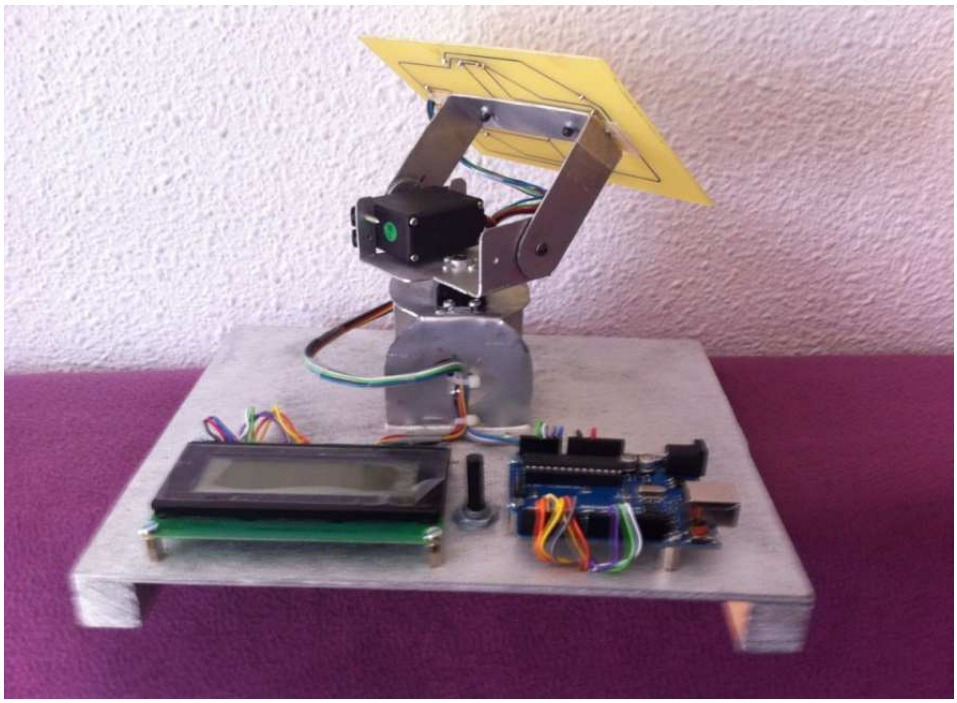


Figura 3. Seguidor lumínico

4. PRUEBAS Y RESULTADOS

Para llevar a cabo las pruebas reunimos varias fuentes de luz de diferente potencia.

$\boldsymbol{valref}$: Es un valor de referencia que nosotros elegimos que representa un margen a partir del cual el sistema de control considera que hay diferencia entre dos divisores de tensión.

4.1. PRIMER EXPERIMENTO

1. Con la fuente de 0.1w y $\boldsymbol{valref}$ **= 1,** hacemos incidir el haz de luz sobre la placa desde una distancia aproximada de 3 metros, nos vamos acercando progresivamente a la placa seguidora hasta que ésta siga el haz de luz, instante en el que medimos la distancia manualmente con el metro, desde la fuente al centro de la placa.

2. Repetimos el paso 1 pero variando valref (5, 10, 45, 65, y 85).
3. Repetimos los pasos 1 y 2 con las fuentes de (0.5, 40, 60 y 100) W.
Anotamos los valores obtenidos en la siguiente tabla.

valref	Fuentes de luz (w)				
	0.1	0.5	40	60	100
	Distancia de detección (cm)				
1	20	63	160	170	220
5	11	44	92	102	130
10	9	35	69	74	93
45	3	17	39	41	59
65	2	13	35	37	55
85	1	8	31	33	45

Tabla 3. Valores obtenidos del primer experimento

4.2. SEGUNDO EXPERIMENTO

Consiste en interrumpir de forma parcial o completa el haz de luz que incide sobre la placa seguidora, intercalando una superficie opaca (una cartulina negra que no deja pasar la luz o crea sombra parcialmente) y ver cómo responde el seguidor lumínico.

4.3. TERCE EXPERIMENTO

Consiste en ver el comportamiento del seguidor lumínico en ausencia de luz o cuando incide un haz de luz en el centro de la placa seguidora.

4.4. CUARTO EXPERIMENTO

Consiste en sacar el seguidor lumínico al exterior y ver cómo se comporta en presencia de los rayos solares.

La idea era tener el seguidor lumínico en un lugar donde los rayos solares puedan incidir sobre su placa seguidora durante unas 3 o 4 horas e ir observando los grados de inclinación y rotación cada 15 o 20 minutos, o sea a medida que el sol vaya describiendo su trayectoria.

En un principio ya sabíamos que este experimento sería el más complicado y el que más tiempo nos iba a llevar, pues la fuente de luz es el sol y no se trata de una fuente que nosotros podamos usarla en el momento que queramos, pues esta sólo está disponible en las horas del sol, es decir, desde el amanecer hasta el ocaso y dependiendo de los factores meteorológicos (cielo nublado, una nube tapando el sol, etc.).

Este seguimiento durante 3 o 4 horas no hemos podido llevar a cabo pues en los días que nos pusimos a realizar esta prueba tuvimos muy pocos minutos de sol. No obstante, en estos pocos minutos de sol que dispusimos cuando inicializábamos el seguidor, éste se posicionaba inmediatamente hacia la dirección de los rayos solares.

5. CONCLUSIONES

En ausencia de luz (0 lux), el seguidor lumínico permanece inmóvil.

En presencia de luz, si tenemos una única fuente de luz el seguidor seguirá el haz de luz de la fuente y si tenemos más de una fuente de luz el seguidor lumínico seguirá el haz de luz de la fuente de mayor intensidad.

La distancia de detección depende de la intensidad de luz de la fuente de modo que a mayor intensidad de luz tendremos mayor distancia de detección.

Durante este proyecto, hemos podido adquirir muchas competencias tales como:

- Entender cómo funciona un sistema mecatrónico interactivo, entendiendo sus partes principales: sensor, actuador y controlador en una estructura física.
- Entender cómo se calibra un sensor LDR para que de valores dentro de un rango de tensión. Esto es aplicable a otros sensores como por ejemplo un potenciómetro.
- Aprender los fundamentos del lenguaje de programación de Arduino: sus principales funciones y el uso de variables globales y locales.
- Entender cómo funciona un algoritmo de control en bucle.
- Familiarizarse con la soldadura de cables y conectores.

Nuestro seguidor lumínico es capaz de seguir haces de luz de fuentes naturales (el sol) y artificiales.

Hemos elaborado un prototipo “seguidor lumínico” experimental de bajo costo

Estamos satisfechos de haber hecho este proyecto, gracias a este proyecto hemos ampliado nuestros conocimientos acerca de Arduino, ya que tiene infinidades de aplicaciones. Descubrimos que su entorno de programación era fácil después de ver algunos ejemplos de códigos en red y haber recopilado mucha información.

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERÍAS INDUSTRIALES

SEGUIDOR LUMÍNICO CON

DOS

GRADOS DE LIBERTAD

METE BOLOPO, PASTOR

CHICHAVA, JOAO ANTONIO

MAYO - 2013

UNIVERSIDAD de VALLADOLID

ESCUELA de INGENIERÍAS INDUSTRIALES

INGENIERO TÉCNICO INDUSTRIAL, ESPECIALIDAD EN ELECTRÓNICA

PROYECTO FIN DE CARRERA

SEGUIDOR LUMINICO CON DOS GRADOS DE LIBERTAD

Autores:

METE BOLOPO, PASTOR

CHICHAVA, JOAO ANTONIO

Tutor:

F. JAVIER GARCIA RUIZ — Ingeniería de Sistemas Automática

MAYO – 2013

P. Mete Bolopo

A mi familia y en especial a mi padre por haberme inculcado la idea de tener estudios para afrontar los desafíos del milenio, al santo padre, por el ánimo y las ganas que me disteis, porque sin vosotros nada de esto hubiera sido posible. A todos mis amigos.

Gracias a todos.

J. A. Chichava

A mi familia y en especial a mi hermano José por toda la confianza que me brindasteis al largo de estos años. A ti princesa por haber estado ahí cuando más apoyo y ánimo necesitaba.

Muchísimas gracias a todos.

Ya podemos decir: ***¡Por Fin¡***

Va Por Vosotros

Índice

Índice de Figuras

Índice de Tablas

1. INTRODUCCION

Un seguidor lumínico es un dispositivo autónomo capaz seguir un haz de luz. Existe una gran variedad de seguidores de luz en el mercado. La clasificación de éstos se puede hacer atendiendo a varios factores, tales como el tipo de plataforma, el número de ejes o el tipo de unidad de control.

Según el tipo de plataforma, los seguidores pueden clasificarse en:

- *1. Plataformas fijas*

Están fijas a través de cimentación con excavación o sin excavación o a través de los tornillos de cimentación realizados con acero galvanizado.

- *2. Plataformas móviles*

Están ancladas a vehículos con desplazamiento, basados en carros o plataformas y dotadas de un sistema locomotor de tipo rodante. Siguen su camino por telemando o guiándose por la información recibida de su entorno a través de sus sensores.

Guiados mediante pistas materializadas a través de la radiación electromagnética de circuitos empotrados en el suelo, o a través de bandas detectadas fotoeléctricamente, pueden incluso llegar a sortear obstáculos y están dotados de un nivel relativamente de inteligencia.

Según el número de ejes pueden clasificarse en:

- ✓ *1. Dispositivos de un eje*

Estos dispositivos solo disponen de un eje de giro, por lo que no permiten alcanzar todas las posiciones angulares. Dentro de este tipo de seguidores, existen varios y diferentes clasificados en función de la configuración del eje de giro.

- ✓ *2. Dispositivos de dos ejes*

Éstos disponen de dos ejes de giro que permiten alcanzar cualquier orientación angular.

La diferencia principal entre los seguidores descritos anteriormente reside en la capacidad de conseguir la orientación adecuada que maximice la captación de la luz.

Los seguidores de un solo eje, permiten acercarse a la orientación óptima, pero no alcanzan, generalmente, una orientación precisa ya que su capacidad de movimiento está limitada a un solo eje. Finalmente, los seguidores de dos ejes permiten alcanzar cualquier orientación, por lo que la captación de luz es máxima.

Según el tipo de la unidad de control, los seguidores se pueden clasificar en:

- *1. Pasivos*

Estos no tienen una electrónica extensa ni actuadores para realizar los giros. Por el contrario, se utilizan sistemas pasivos el seguimiento.

- *2. Micro procesados*

Estos seguidores no necesitan sensores para la detección de la incidencia de la luz, sino que utilizan algoritmos matemáticos que calculan la trayectoria de la luz en función de la ubicación del dispositivo y la hora y fecha en la que se realiza el cálculo.

- *3. Electro-Ópticos*

Éstos utilizan algún tipo de sensor para determinar la posición angular real de la luz.

Los seguidores pasivos tienen una precisión muy limitada ya que no utilizan mecanismos activos de giro y dependen fundamentalmente de la variación de densidad de algún líquido provocada por incidencias diferentes en función de la orientación del seguidor. Los seguidores micro-procesados dependen de un correcto posicionado conseguido habitualmente mediante GPS y de una señal de reloj que indique la hora correcta. Al no tener sensores, estos sistemas son incapaces de determinar si existe algún tipo de perturbación externa. Finalmente, los seguidores electro-ópticos hacen uso de sensores, por lo que pueden determinar si alguna perturbación está afectándoles.

Normalmente los seguidores de luz forman parte de la estructura ya sea de los paneles solares, colector solar, robots o vehículos móviles, por lo que el movimiento de ambos es solidario. En el caso abordado en este proyecto, se diseña un seguidor lumínico de dos ejes que obtendrá la orientación de la luz a través de las LDRs y que posteriormente enviarán dicha información a un microcontrolador que actuará sobre los servomotores para su correcta orientación.

En función de la clasificación descrita anteriormente, el dispositivo objeto del proyecto estaría encuadrado, atendiendo al tipo de plataforma a los fijos, atendiendo a la capacidad de movimiento, en el grupo de seguidores de dos ejes; y atendiendo a la unidad o tipo de control, estaría incluido dentro del grupo de seguidores electro-ópticos.

1.1. Motivación para la realización del proyecto

Este proyecto nos da la posibilidad de comenzar a adquirir una metodología de trabajo en proyectos que involucren mecánica, electrónica y control informático. Por tanto se pone énfasis en la adquisición de competencias.

Visto que la unidad de control del seguidor lumínico será el microcontrolador de la placa Arduino Uno Rev3, podemos adquirir y ampliar nuestros conocimientos acerca de la plataforma Arduino, pues se trata de una plataforma con una infinidad de aplicaciones en el mundo de la electrónica en el control de dispositivos de bajo consumo como LEDs, servomotores, motores de corriente continua, relés, etc.

1.2. Objetivos del proyecto

El objetivo de este proyecto, es construir y programar un seguidor lumínico con dos grados de libertad, que sea capaz seguir un haz de luz.

Para lograr este objetivo el sistema constará de una superficie en la que se posicionan cuatro fotocélulas divididas en grupos de dos, apoyadas sobre una plataforma mecánica con dos grados de libertad. Se deberán detectar umbrales de luz comparando los valores de dos fotocélulas de cada grupo a través del microcontrolador Arduino. Como resultado de dicha comparación se generará un movimiento de rotación o de inclinación a través de dos servomotores de forma que la superficie se mantenga perpendicular al haz lumínico y a través de una pantalla LCD en cada momento conoceremos el ángulo de rotación y el de inclinación de la superficie seguidora.

1.3. Descripción del prototipo a desarrollar

El seguidor lumínico que se presenta en este documento es un dispositivo que se maneja en dos ejes. La base en que se fundamenta el funcionamiento del mismo corresponde al seguimiento de una fuente luminosa, ya sea artificial o solar. El dispositivo seguidor estará conformado por tres etapas básicas: Recepción y acondicionamiento de la señal de entrada. Procesamiento y control de la señal y etapa de salida.

En la etapa de recepción y acondicionamiento de la señal, se hace uso de cuatro sensores y la tarjeta adquisidora Arduino Uno mediante el módulo de conversión A/D del microcontrolador atmega328 de ATMEL. Los sensores son cuatro fotorresistencias divididas en dos grupos de dos dispuestas en la placa seguidora de tal forma que puedan captar los rayos luminosos que incidan de manera perpendicular sobre la superficie de la misma.

La etapa de procesamiento y control de la señal consiste básicamente en realizar una comparación entre los valores obtenidos de dos sensores (dos fotorresistencias para el eje vertical y dos fotorresistencias para el eje horizontal). Dependiendo del caso que se presente y de la forma en el que se haya programado el microcontrolador, se definirá un sentido de giro para el servomotor de eje horizontal; y también se definirá un sentido de giro para el servomotor de eje vertical.

En la etapa de salida el servomotor tanto el del eje vertical como el del eje horizontal es controlado mediante una salida PWM generada por el microcontrolador; dependiendo del ancho de pulso generado, éste girará en un sentido u en otro.

1.4. Estructura de la memoria

La estructura de la memoria de este documento contiene lo siguiente:

- En el **Capítulo 1**, se realiza una introducción al proyecto que se va a realizar, explicando la motivación que nos lleva a realizarlo, los objetivos marcados así como una breve descripción del mismo.

- En el **Capítulo 2**, se expone los antecedentes en las aplicaciones del proyecto, tanto en la mejora en la captación de energía solar bien como en la rama de robótica.

- En el **Capítulo 3**, se realiza una descripción de los componentes integrantes del proyecto, haciendo una brevemente introducción al mundo del Arduino, del Display LCD, pasando por el principio básico del funcionamiento de una LDR bien como controlar un servomotor.

- En el **Capítulo 4**, se muestra paso a paso el desarrollo del proyecto desde el algoritmo del seguidor pasando por el diseño del circuito electrónico, diseño y fabricación de las piezas de aluminio hasta el montaje y el conexionado del seguir lumínico.

- En el **capítulo 5**, se detallan las pruebas y resultados, realizando experimentos para ver el comportamiento del seguidor en ausencia de luz (oscuridad) y en presencia de luz con fuentes de luz de diferentes intensidades.

- En el **Capítulo 6**, se dedica a las conclusiones. Se analiza si se han cumplido los objetivos propuestos en el inicio del proyecto. Aquí se explica también sobre las dificultades y problemas surgidos durante la realización del proyecto y se indica la solución los mismos.

- En el Capítulo 7, se muestran los anexos de programación, del LCD, de los servomotores y bibliografía.

2. ANTECEDENTES

2.1. Energía solar

Los combustibles fósiles (petróleo, carbón, gas natural) también llamados fuentes de energía no renovables, en los últimos años sus reservas han ido disminuyendo a un gran ritmo. Teniendo en cuenta que estas fuentes representan el 80% de la energía comercial empleada en el mundo. Esta energía se obtiene al quemar los combustibles fósiles, proceso en el cual se forman grandes cantidades de anhídrido carbónico y otros gases contaminantes que se emiten a la atmosfera eso ocasiona un gran daño a nuestro ambiente y por lo tanto a nosotros mismos. En pocas palabras, el problema es que nuestras fuentes de energía se están agotando y no hay forma de renovarlas además destruyen nuestro ecosistema.

Una de las alternativas a esta problemática es el uso de fuentes de energía renovables, las cuales no consumen un recurso finito como un combustible fósil, además de causar menos impactos ambientales. Alguna de estas fuentes son: energía hidroeléctrica, energía solar, energía de la biomasa, energía eólica, energía geotermal, entre otras.

Los seguidores solares actuales basan su algoritmo de seguimiento del sol en variables del tiempo: segundos, minutos, horas, día, día de la semana, día del mes, mes, año, huso horario, latitud, hora de la salida del sol, de la puesta del sol, con una compensación de los meses con 31 días y del año bisiesto hasta el año 2100. Esto quiere decir, se basan en el conocimiento de la trayectoria del sol independientemente de que sus rayos incidan o no en la tierra (cielo cubierto, eclipse,…).

2.1.1. Aplicaciones en la rama de la energía solar

Para mejorar la captación de la radiación solar se puede utilizar un seguidor lumínico que es un dispositivo mecánico capaz seguir un haz de luz que orientará los paneles solares o el colector solar de manera que su posición siempre se encuentre perpendicular a los rayos solares, siguiendo la trayectoria del sol desde el amanecer hasta el ocaso basándose en los rayos incidentes.

La **energía solar**, se puede aprovechar para generar electricidad mediante celdas fotovoltaicas, las cuales son dispositivos formados por metales sensibles a la luz que desprenden electrones cuando los fotones inciden sobre ellos. Otra manera de aprovechar la energía solar es mediante un colector solar, el cual es un dispositivo que utiliza el calor del sol para calentar alguna sustancia (agua, aceite, etc.) para la

climatización de piscinas, calefacción, o generación de energía mediante el movimiento de una turbina.

2.2. Robótica

La robótica es la ciencia y la tecnología de los robots. Se ocupa del diseño, manufactura y aplicaciones de los robots. La robótica combina diversas disciplinas como son: la mecánica, la electrónica, la informática, la inteligencia artificial y la ingeniería de control.

La robótica en un concepto de predominio mundial, la mayor parte de la gente tiene una idea de lo que es robótica sus aplicaciones y el potencial que tiene.

La robótica es la rama de la tecnología y es un instrumento importante en el desarrollo del ser humano y es la que actualmente sustituye al hombre en algunas tareas como el avance en la tecnología donde se incluye las poderosas computadoras electrónicas los actuadores de control retroalimentado transmisión de potencia a través de engranes y la tecnología en sensores ha contribuido a flexibilizar los mecanismos automáticos en diferentes tareas.

Un robot es una entidad virtual o mecánica artificial. En la práctica, esto es por lo general un sistema electromecánico que, por su apariencia o sus movimientos, ofrece la sensación de tener un propósito propio.

La historia de la robótica ha estado unida a la construcción de "artefactos", que trataban de materializar el deseo humano de crear seres a su semejanza y que lo descargasen del trabajo.

El ingeniero español Leonardo Torres Quevedo (GAP) que construyó el primer mando a distancia para su automóvil mediante telegrafía sin hilo, el ajedrecista automático, el primer transbordador aéreo y otros muchos ingenios; acuñaron el término "automática" en relación con la teoría de la automatización de tareas tradicionalmente asociadas a los humanos.

El robot seguidor de luz se clasifica en el campo de robótica móvil. La tarea fundamental de un robot móvil es el desplazamiento, en un entorno conocido o desconocido.

El término robótica es acuñado por Isaac Asimov, definiendo a la ciencia que estudia a los robots. Asimov creó también las Tres Leyes de la Robótica.

En la ciencia ficción el hombre ha imaginado a los robots visitando nuevos mundos, haciéndose con el poder, o simplemente aliviando de las labores caseras.

2.2.1. Aplicaciones en la rama de la robótica

Podemos aplicar también el seguidor lumínico en la fabricación de robots y vehículos móviles utilizados en casos de:

- Emergencia o desastres naturales: llevar agua, alimentos y oxígeno a personas atrapadas dentro de escombros o a diferentes lugares en los cuales el acceso a las personas es nulo o sumamente peligroso.

- Exploración u otras actividades que impliquen salvar vidas, proteger o prevenir riesgos a las personas (industria espacial, industria mineras, industria química,…)

3. DESCRIPCION DE LOS COMPONENTES

Para poder llevar a cabo los objetivos antes mencionados se van a utilizar los componentes siguientes:

- Cuatro fotocélulas situadas en una placa que actuará como superficie de seguimiento.

- Un microcontrolador para comparar las señales de las fotocélulas y calcular la acción de control necesaria. Se ha elegido una placa Arduino Uno Rev3, debido a su gran versatilidad, su fácil manejo y su reducido precio.

- Una pantalla LCD para visualizar el ángulo rotado o inclinado.

- Se construirá una plataforma que tenga dos grados de libertad, permitiendo la rotación y la inclinación de la superficie seguidora.

- Dos servomotores que serán los encargados de mover la plataforma de acuerdo con las órdenes del programa del controlador.

3.1. Arduino Uno Rev3

- "**Arduino Uno**" es el nombre que ha sido elegido para el lanzamiento de Arduino 1.0. Uno es la última de la serie de placas Arduino USB y el modelo de referencia más actualizado sucesora de la Arduino Duemilanove para la plataforma Arduino.

- **Uso del Arduino Uno. ¿Qué es el Arduino?**

Arduino es una herramienta para hacer que los ordenadores puedan sentir y controlar el mundo físico.

Es una plataforma de desarrollo de computación física de código abierto, basada en una placa con un sencillo microcontrolador y un entorno de desarrollo para crear software (programas) para la placa.

Se puede usar Arduino para crear objetos interactivos, leyendo datos de una gran variedad de interruptores y sensores y controlar multitud de tipos de luces, motores y otros actuadores físicos.

Los proyectos de Arduino pueden ser autónomos o comunicarse con programa o software que se ejecute en tu ordenador como (Macromedia Flash, Processing, Max/MSP).

La placa se puede montar a mano o adquirirse y lista para usar, el software de desarrollo es abierto y se puede descargar gratis. Por otro lado el software consiste en un entorno de desarrollo que implementa el lenguaje de programación Processing/Wiring y el cargador de arranque (*bootloader*) que corre en la placa

.¿**Por qué utilizar Arduino?**

Pues lo especial del Arduino UNO consiste en dos aspectos:

- ✓ El hardware es muy fácil de entender y de manejar. No se requiere ser un "iniciado" en electrónica ni un especialista en microcontroladores para poder sacarle "jugo" a esta tarjeta. Como interface de entrada/salida el Arduino Uno usa simplemente conectores "Headers" hembra. Lo más bonito está en que puedes agregar encima de la tarjeta el hardware deseado que necesite tu aplicación específica.

- ✓ El software se construye a base de bloques, igual que el hardware. Mucha gente lo aprende solamente practicando a base de ejemplos o bien con algunos de los numerosos tutoriales en la red.

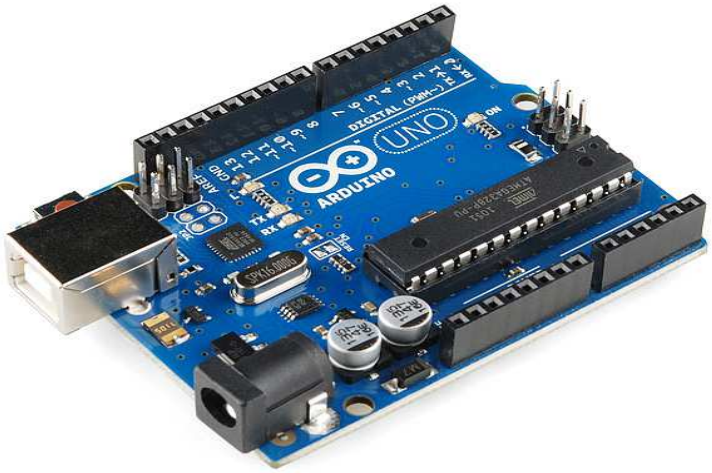


Figura 1. Placa Arduino

Hay otros microcontroladores y plataformas con microcontroladores disponibles para la computación física, como Parallax basic stamp, Bx-24 de Netmedia, phidgets, handyboard del MIT, y muchos otros que ofrecen funcionalidades similares.

Todas estas herramientas organizan el complicado trabajo de programar un microcontrolador en paquetes fáciles de usar.

Pero Arduino, además de simplificar el proceso de trabajar con microcontroladores, ofrece algunas ventajas respecto a otros sistemas a profesores, estudiantes y amateurs:

- Asequible - Las placas Arduino son más asequibles comparadas con otras plataformas de microcontroladores. La versión más cara de un módulo de Arduino puede ser montada a mano abaratando así aún más su coste.

- Multi-Plataforma - El software de Arduino funciona en los sistemas operativos Windows, Macintosh y Linux. La mayoría de los entornos para microcontroladores están limitados a Windows.

- Entorno de programación simple y directo - El entorno de programación de Arduino es fácil de usar para principiantes y lo suficientemente flexible para los usuarios avanzados. Pensando en los profesores, Arduino está basado en el entorno de programación de Processing con lo que el estudiante que aprenda a programar en este entorno se sentirá familiarizado con el entorno de desarrollo Arduino.

- Software ampliable y de código abierto- El software Arduino está publicado bajo una licencia libre y preparado para ser ampliado por programadores experimentados. El lenguaje puede ampliarse a través de librerías de C++, y si se está interesado en profundizar en los detalles técnicos, se puede dar el salto a la programación en el lenguaje AVR C en el que está basado. De igual modo se puede añadir directamente código en AVR C en tus programas si así lo deseas.

- Hardware ampliable y de Código abierto - Arduino está basado en los microcontroladores ATMEGA168, ATMEGA328 y ATMEGA1280. Los planos de los módulos están publicados bajo licencia Creative Commons, por lo que diseñadores de circuitos con experiencia pueden hacer su propia versión del módulo, ampliándolo u optimizándolo.

¿Qué son los microcontroladores?

Un microcontrolador (abreviado µC, UC o MCU) es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica.

Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

Algunos microcontroladores pueden utilizar palabras de cuatro bits y funcionan a velocidad de reloj con frecuencias tan bajas como 4KHz, con un consumo de baja potencia (mW o microvatios). Por lo general, tendrá la capacidad para mantener la funcionalidad a la espera de un evento como pulsar un botón o de otra interrupción, el consumo de energía durante el sueño (reloj de la CPU y la mayoría de los periféricos) puede ser sólo nanovatios, lo que hace que muchos de ellos sean muy adecuados para aplicaciones con batería de larga duración.

Otros microcontroladores pueden servir para controles de rendimiento crítico, donde sea necesario actuar más como un procesador digital de señal (DSP), con velocidad de reloj y consumo de energía más altos.

Al ser fabricados, la memoria ROM del microcontrolador no posee datos. Para que pueda controlar algún proceso es necesario generar o crear y luego grabar en la EEPROM del microcontrolador algún programa, el cual puede ser escrito en lenguaje ensamblador u otro lenguaje para microcontroladores; sin embargo, para que el programa pueda ser grabado en la memoria del microcontrolador debe ser codificado en sistema numérico hexadecimal que es finalmente el sistema que hace trabajar al microcontrolador cuando éste es alimentado con el voltaje adecuado y asociado a dispositivos analógicos y discretos para su funcionamiento.

Algunas aplicaciones de los microcontroladores:

- Sistemas de comunicación: en grandes automatismos como centrales y en teléfonos fijos, móviles, fax, etc.

- Electrodomésticos: lavadoras, hornos, frigoríficos, lavavajillas, batidoras, televisores, vídeos, reproductores DVD, equipos de música, mandos a distancia, consolas, etc.

- Industria informática: se encuentra casi todos los periféricos; ratones, teclados, impresoras, escáner, etc.

- Automoción: climatización, seguridad, ABS, etc.

- Industria: autómatas, control de procesos, etc.

- Sistemas de supervisión, vigilancia y alarmas: ascensores, calefacción, aire acondicionado, alarma de incendio, robo, etc.

3.1.1. Hardware

Especificaciones:

Microcontrolador	ATmega328
Voltaje de operación	5V
Voltaje de entrada(recomendada)	7-12V
Voltaje de entrada (rango)	6-20V
E/S digitales	14 (6 de las cuales provistas de PWM)
Entradas analógicas	6
Imax dc E/S para 5V	40 mA
Imax dc E/S para 3.3V	50 mA
Memoria Flash	32 KB (ATmega328) de la cual 0.5 KB es usada para bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Velocidad de reloj	16 MHz

Alimentación del sistema

La placa Arduino puede ser alimentada vía USB o con una fuente de alimentación externa.

La fuente de alimentación externa puede consistir en un adaptador AC-DC o una batería. En caso de conectar el cable USB y una fuente externa, Arduino seleccionará automáticamente la procedencia de la alimentación, además de una protección para cortocircuitos. El adaptador se puede conectar en el Jack de conexión que trae la placa.

También se puede conectar directamente mediante las líneas marcadas como GND y Vin en las conexiones de potencia con la etiqueta POWER.

VIN. Línea de entrada de tensión a la placa del Arduino cuando se usa una fuente de alimentación externa. Se puede aplicar tensión a través de este pin, o, si se está alimentando la placa mediante el Jack de conexión, acceder a ella a través de él.

5V. La fuente de alimentación regulada que alimenta al microcontrolador y otros componentes de la placa.

3V3. Alimentación de 3,3v generado por el chip FTDI de la placa. La máxima corriente de salida es 50mA.

GND. Tierra. Uno de los objetivos de este trabajo está directamente relacionado con la forma de alimentar el sistema. El dispositivo final debe poder alimentarse de la misma red en la que se está midiendo, o mediante un cable USB.

3.1.2. Software

¿Cómo conectar la placa Arduino al ordenador y volcar el código del primer programa?

Para programar la tarjeta se necesita el ambiente de desarrollo Arduino, el cual se puede descargar del siguiente link **http://www.arduino.cc**

- 1 | Consigue un Arduino y un cable USB
- 2 | Descarga el IDE de Arduino
- 3 | Conecta la placa
- 4 | Instala los *drivers*
- 5 | Ejecuta la Aplicación Arduino
- 6 | Abre el ejemplo *Blink*
- 7 | Selecciona tu placa
- 8 | Selecciona tu puerto serie
- 9 | Sube el *sketch* a la placa
- **1 | Consigue un Arduino y un cable USB**

Asumimos que estamos usando una placa Arduino Rev3. Si tienes cualquier otra placa necesitas leer las instrucciones correspondientes a tu placa.

También necesitarás un cable estándar USB (conexión A - conexión B), como los que se usan para conectar, por ejemplo, una impresora USB.

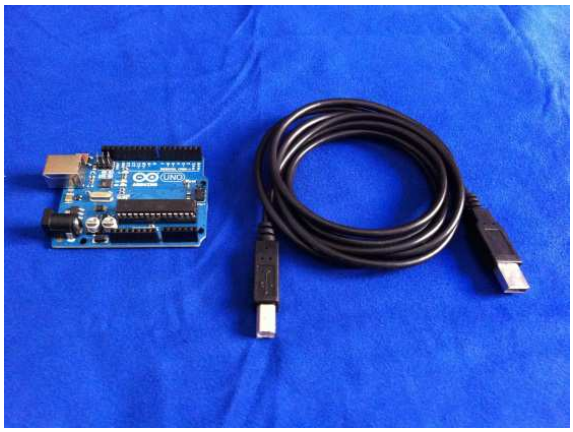


Figura 2. Placa Arduino y Cable USB

- **2 | Descarga el IDE de Arduino**

Descarga la última versión de la página de descargas de www.arduino.cc. Cuando la descarga finalice, descomprime el fichero. Asegúrate de mantener la estructura de directorios. Haz doble click en la carpeta *arduino-00XX* para abrirla. Deberías ver una serie de ficheros y carpetas ahí dentro.

Ponemos el ratón sobre el fichero "arduino.exe" y cliqueamos con el botón derecho del ratón y seleccionamos "Ejecutar como administrador" (no es necesario, pero así evitamos posibles problemas de limitaciones de Microsoft Windows 7).

- **3 | Conecta la placa**

Conecta la placa Arduino a tu ordenador usando el cable USB. El LED verde indicador de la alimentación (nombrado como PWR en la placa) debería quedar encendido a partir de ese momento.

- **4 | Instala los *drivers***

Cuando conectas la placa, Windows debería inicializar la instalación de los *drivers* (siempre y cuando no hayas utilizado ese ordenador con una placa Arduino anteriormente).

En Windows Vista y Windows 7, los drivers deberían descargarse e instalarse automáticamente.

En Windows XP, se abrirá el diálogo de instalación de Nuevo Hardware:

- Cuando te pregunten: ¿Puede Windows conectarse a Windows Update para buscar el software? selecciona No, no esta vez. Haz click en *Siguiente*.

- Selecciona Instalar desde una lista o localización específica (Avanzado) haz click en *Siguiente*.

- Asegúrate que Buscar los mejores drivers en estas localizaciones esté seleccionado; deselecciona Buscar en medios removibles; selecciona Incluye esta localización en la búsqueda y navega al directorio drivers/FTDI USB Drivers dentro de la carpeta de Arduino que has descomprimido previamente. (La versión más reciente de los *drivers* se puede encontrar en la página web del fabricante del chip FTDI.) Haz click en *Siguiente*.

- El asistente de instalación buscará los *drivers* y te anunciará que encontró un "USB Serial Converter" (se traduce por *Conversor USB-Serie*). Haz click en *Finalizar*.

- El asistente de instalación de hardware volverá a iniciarse. Repite los mismos pasos que antes y selecciona la misma carpeta de instalación de los *drivers*. Esta vez el sistema encontrará un "USB Serial Port" (o *Puerto USB-Serie*).

Puedes comprobar que los *drivers* se han instalado correctamente abriendo la carpeta del Administrador del Dispositivos, en el grupo *Dispositivos* del panel de control del sistema. Busca "USB Serial Port" (o *Puerto USB-Serie*) en la sección de puertos; esa es tu placa Arduino.

- **5 | Ejecuta la Aplicación Arduino**

Haz doble click en la aplicación Arduino.

- **6 | Abre el ejemplo *Blink***

Abre el programa de ejemplo para hacer parpadear un LED ("LED blink"):
File > Examples > Digital > Blink.

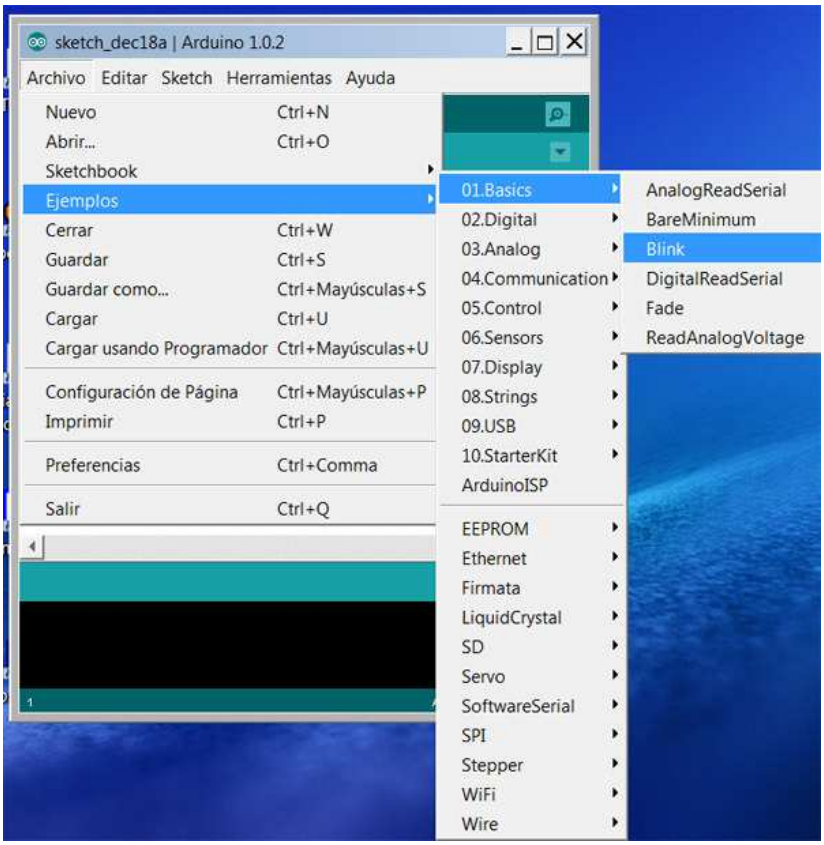


Figura 3. Ejemplo Blink

- **7 | Selecciona tu placa**

Necesitarás seleccionar el tipo de placa de tu Arduino en el menú Tools > Board. Para las nuevas placas Arduino con el chip ATmega 328 (comprueba el texto escrito en el chip de la placa), selecciona la opción Arduino Uno del menú desplegable. Anteriormente las placas Arduino incluían un chip ATmega 168.

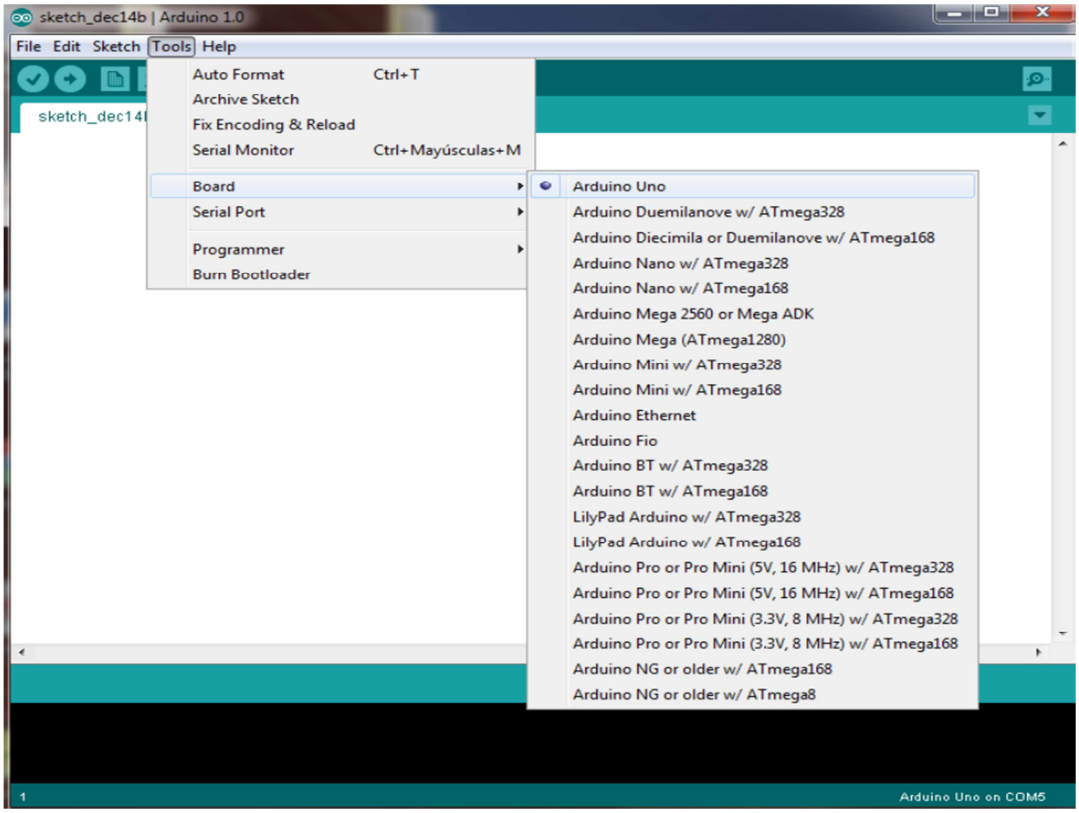


Figura 4. Selección de la placa Arduino

- **8 | Selecciona tu puerto serie**

Selecciona el dispositivo serie de la placa Arduino en el menú Tools | Serial Port (*Herramientas* | *Puertos Serie*). Lo más probable es que sea COM3 o mayor (COM1 y COM2 se reservan, por regla general para puertos serie de hardware).

Para asegurarte de cual es, puedes desconectar la placa y volver a mirar el menú; el puerto de la placa habrá desaparecido de la lista. Reconecta la placa y selecciona el puerto apropiado.

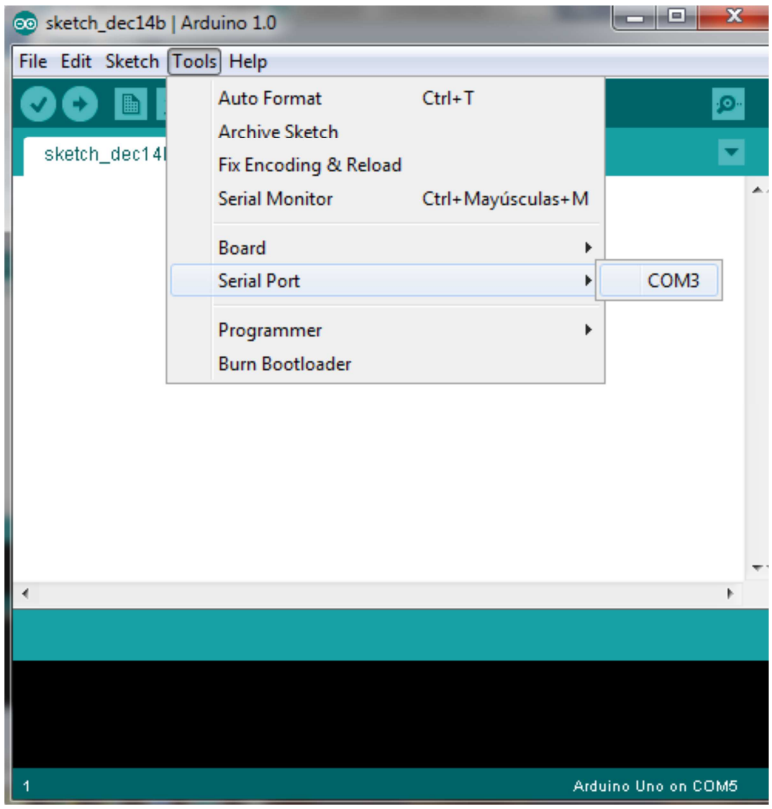


Figura 5. Selección del puerto Serie

- **9 | Sube el *sketch* a la placa**

Ahora simplemente pulsa sobre el botón "Upload" en el Entorno Arduino. Espera unos pocos segundos - deberías ver parpadear los LED RX y TX de la placa. Si el volcado del código es exitoso verás aparecer el mensaje "Done uploading" en la barra de estado.

(*Aviso:* Si tienes una placa Arduino Mini, NG, u otras placas, necesitarás presionar el botón de reseteo de la placa inmediatamente antes de presionar el botón "Upload" el Entorno de programación Arduino.)

Unos pocos segundos después de finalizar el volcado del programa deberás ver cómo el LED de la placa conectado al pin 13 (L) comienza a parpadear (con un color naranja). Con esto ya tendríamos configurada la placa.

3.2. Pantalla LCD

Las pantallas de cristal Líquido o pantallas LCD para mensajes, tienen la capacidad de mostrar cualquier carácter alfanumérico, permitiendo representar la información que genera cualquier equipo electrónico generalmente las pantallas suelen constar de una matriz 5x7 o 5x8 puntos distribuidos en dos, tres, o cuatro líneas de 16 hasta 40 caracteres cada línea. El proceso de visualización es gobernado por un microcontrolador incorporado a la pantalla, siendo Hitachi 44780 el módulo de controlador más utilizado.

En este proyecto, utilizaremos el modelo 4x20 compuesto por 4 líneas de 20 caracteres, **LiquidCrystal** – Es una biblioteca que nos permite controlar la pantalla LCD que son compatibles con el Hitachi 44780 conductor.

Figura 6. Pantalla LCD 20x4

El LCD tiene un interfaz en paralelo, lo que significa que el microcontrolador tiene que manipular varios pines a la vez para controlar la visualización. La interfaz se compone de los siguientes pines:

- RS: selección de registro, pin que controla el lugar de la memoria de la pantalla LCD que está escribiendo datos. Puede seleccionar el registro de datos, que contiene lo que sucede en la pantalla, o un registro de instrucción, que es donde el controlador de la pantalla LCD se ve para obtener instrucciones sobre qué hacer a continuación.

- R/W: es el pin de lectura y escritura, selecciona el modo lectura o escritura. Siguiendo con la descripción, destacar que tiene :

- Ocho pines de datos que van de (D0÷D7). Los estados de dichos pines que pueden ser (altos o bajos) son los bites que se están escribiendo en un registro cuando se escribe, o los valores que estamos leyendo.

- VO: para el contraste, pines de alimentación (+5 y GND) y retroalimentación del LED (BKLT+ Y BKLT-) pines que se pueden utilizar para alimentar el LCD, controlar el contraste de la pantalla, y encender y apagar el LED luz de fondo, respectivamente.

El proceso de control de visualización consiste en colocar los datos que forma la carga de lo que se desea mostrar en los registros de datos, a continuación poner las instrucciones en el registro de instrucciones. Cabe señalar que la biblioteca LiquidCrystal se encarga de todo este proceso.

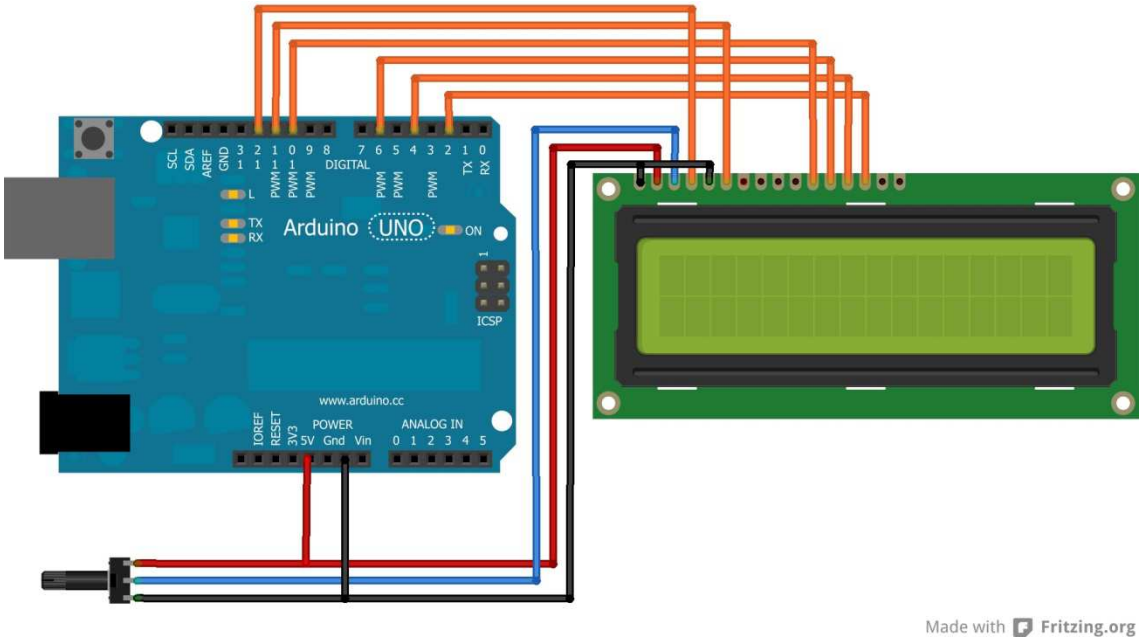
El Hitachi compatible con las pantallas LCD se puede controlar en dos modos: 4 bites u 8 bites.

El modo 4 bites requiere siete pines I/O del Arduino, para mostrar textos en la pantalla.
El modo de 8 bites requiere once pines I/O del Arduino, para mostrar texto en la pantalla.
No obstante se puede hacer casi todo en modo 4 bites con el LCD 4x20.

3.2.1. Conexiones de la pantalla LCD a la placa Arduino

Decir que hay varias formas de conectar la pantalla LCD a la placa Arduino dependiendo del objetivo que se persigue. Mostramos a continuación un ejemplo con las siguientes conexiones:

LCD (RS) pin, va conectado al pin digital 12
LCD (ENABLE) pin, va conectado al pin digital 11
LCD (D4) pin, va conectado al pin digital 10
LCD (D5) pin, va conectado al pin digital 6
LCD (D6) pin, va conectado al pin digital 4
LCD (D7) pin, va conectado al pin digital 2



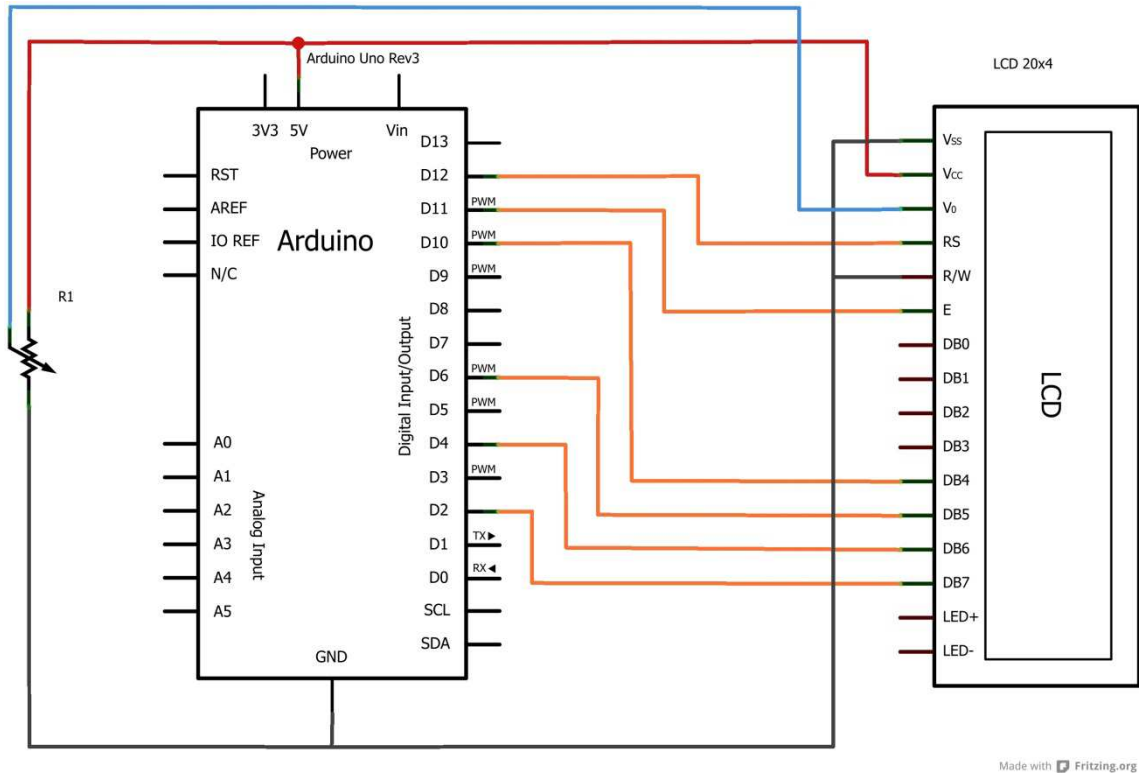


Figura 7. Esquema de conexión Arduino - LCD

Además, conectamos un potenciómetro de 10k a +5V y GND, con la patilla de variación conectada al pin3 pantalla LCD pin v_0.

Un **potenciómetro** es un resistor cuyo valor de resistencia es variable.

Figura 8. Potenciómetro

De esta manera, indirectamente, se puede controlar la intensidad de corriente que fluye por un circuito si se conecta en paralelo, o la diferencia de potencial al conectarlo en serie.

Normalmente, los potenciómetros se utilizan en circuitos de poca corriente. El objetivo de este potenciómetro es regular el contraste de la pantalla del Display LCD.

3.3. Fotorresistencias LDR y resistencias

Los sensores de luz más comunes son las resistencias dependientes de la luz o LDR (Light Dependent Resistor).

Un LDR es un resistor que varía su valor de resistencia eléctrica dependiendo de la cantidad de luz que incide sobre él. Se le llama, también, fotorresistor o fotorresistencia

Este tipo de sensor es muy común debido a su sencillez y bajo coste y consiste básicamente en un dispositivo que cambia su resistencia cuando cambia la intensidad de la luz. Este componente electrónico disminuye su resistencia con el aumento de luz incidente, pudiendo llegar a decenas de ohmios. Por el contrario, cuando se encuentra a oscuras, la resistencia entre sus dos patillas puede llegar a ser de varios megaohmios.

Una LDR está compuesta de una célula y dos patillas. La célula está formada por un semiconductor de alta resistencia como el sulfuro de cadmio (CdS).

Cuando la luz que incide sobre el dispositivo tiene energía suficiente para superar la banda de conducción, los electrones quedan libres y conducen electricidad, por lo que la resistencia total del dispositivo disminuye.

Las células de sulfuro cadmio se basan, por tanto, en la capacidad que tiene el cadmio de variar su resistencia en función de la cantidad de luz que incide en la célula.

Cuanto mayor es la cantidad de luz que incide sobre la superficie de la célula, menor es la resistencia que presenta ésta al paso de corriente. Este tipo de células son sensibles a una alta gama de frecuencias entre las que se encuentran las frecuencias infrarrojas, la luz visible y la ultravioleta. Esta gama implica que los sensores no sólo reaccionarán ante la luz solar, sino a otro gran número de interferencias. Esto es importante tenerlo en cuenta a la hora de diseñar el dispositivo para evitar una influencia excesiva de diferentes perturbaciones a la de la propia luz solar, que es la única que interesa.

La variación del valor de resistencia de este tipo de sensores tiene un cierto retardo. Por este motivo, estos sensores no son apropiados para usos en los que la variación lumínica es muy rápida. El tiempo de respuesta de las LDR suele ser del orden de una décima de segundo, por lo que, para el uso que se pretende dar en este proyecto, no existirá ningún tipo de problema siempre y cuando el periodo de muestreo sea superior al tiempo de respuesta de los sensores.

La variación de valor resistivo de un LDR tiene cierto retardo, que es diferente si se pasa de oscuro a iluminado o de iluminado a oscuro.

Por esta razón un LDR no se puede utilizar en algunas aplicaciones, en especial en aquellas en que la señal luminosa varía con rapidez.

La lentitud relativa del cambio es una ventaja en algunos casos, porque así se filtran variaciones rápidas de iluminación que podrían hacer inestable un sensor (por ejemplo cuando está iluminado por un tubo fluorescente alimentado por corriente alterna). En otras aplicaciones (como la detección de luminosidad para saber si es de día o es de noche) la lentitud de la detección no es importante.

Estos sensores se construyen a partir de unos componentes de selenio, sulfuro de cadmio o de plomo que se caracterizan por variar su resistencia en función de la luz que reciben.

En las figura 9 vemos su símbolo y su aspecto físico.

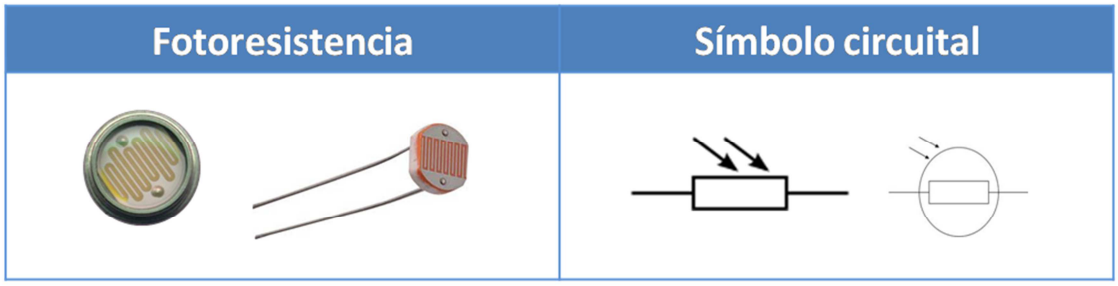


Figura 9. Símbolo y aspecto físico de una LDR

Las dimensiones de las células pueden ser variadas, pero los modelos comerciales más comunes son dos, las de 10-12 mm de diámetro y las de 5 mm. Para el proyecto serán necesarios cuatro sensores, por lo que, si se eligen los sensores de un diámetro superior, el tamaño total del dispositivo puede ser excesivo. Por este motivo, para este proyecto, el tamaño óptimo de los sensores es 5 mm de diámetro.

Las dimensiones de estos sensores son las siguientes:

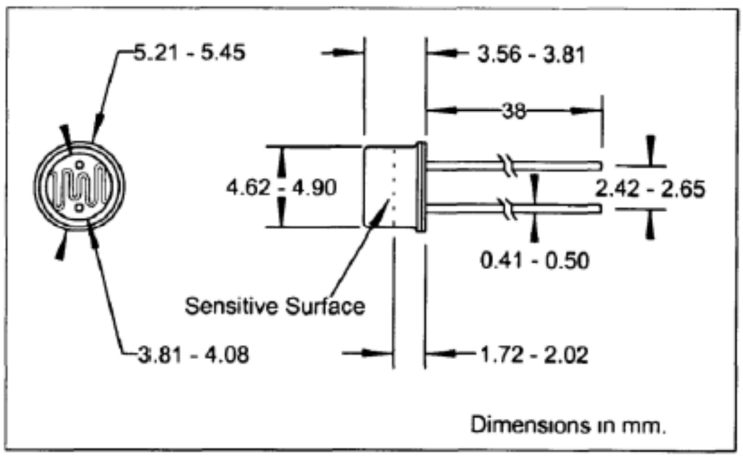


Figura 10. Dimensiones de una LDR

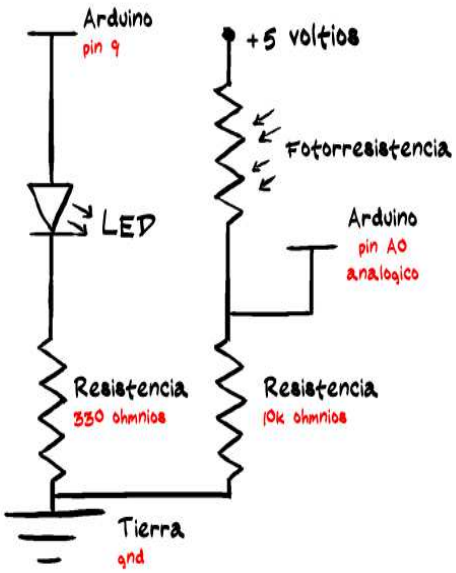


Figura 11. LDR conectada a un pin analógico de Arduino

La variación de la intensidad luminosa recibida por una LDR puede estar provocada por varios motivos. Uno de éstos puede ser una variación en la intensidad del foco que genera la fuente luminosa. En el caso de la luz solar, esta variación puede ser provocada por las condiciones climáticas o ambientales. Otro de los motivos que pueden provocar una variación en la intensidad luminosa que incide sobre un dispositivo detector es la sombra provocada por un objeto opaco que se interponga entre la fuente de luz y el dispositivo detector. Este último motivo será el utilizado para el desarrollo, diseño e implementación del seguidor solar objeto del proyecto.

Una LDR expuesta a una fuente de luz constante sin ningún objeto que se interponga entre su superficie y la fuente luminosa (en este caso es el sol) presentará una resistencia mínima, ya que toda su superficie estará recibiendo la incidencia de los fotones que suministran la energía suficiente para provocar la circulación de corriente.

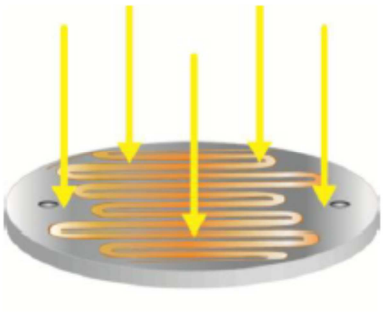


Figura 12. Sensor LDR expuesto a haces de luz

Si se coloca un objeto junto al sensor y los rayos que inciden de forma perpendicular sobre la superficie del sensor, el objeto en cuestión no genera ninguna sombra sobre el sensor, por lo que éste seguirá presentando un valor de resistencia mínimo.

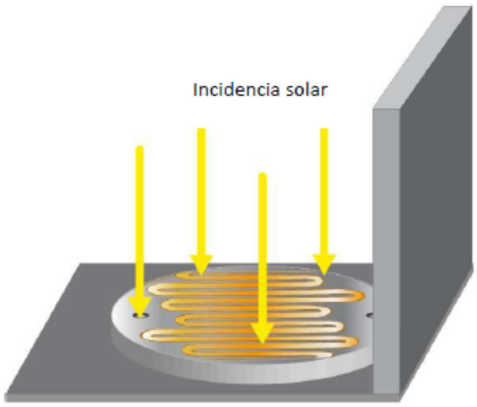


Figura 13. Sensor LDR expuesto a la luz solar con objeto

Por el contrario, si los rayos solares no inciden de manera perpendicular sobre la superficie del sensor, sino que lo hacen formando un ángulo cualquiera, este objeto podría provocar una sombra sobre dicha superficie y, por tanto, la intensidad luminosa incidente sobre el sensor sería menor, aumentando con ello la resistencia del dispositivo

Que se produzca una sombra sobre el sensor, dependerá del ángulo de los rayos solares con respecto al plano generado por la superficie del mismo. Si los rayos inciden desde el lado opuesto al que se encuentra ubicado el objeto, éste seguirá sin producir ningún tipo de sombra sobre el sensor. Si, por el contrario, los rayos inciden desde la misma zona donde está colocado el objeto, éste estará interpuesto entre el sensor y la fuente luminosa, por lo que proyectará una sombra sobre el sensor

La forma de la proyección dependerá fundamentalmente de la forma del objeto que la provoca y de la distorsión que provoque la fuente luminosa. Si la fuente luminosa es artificial, los rayos que parten de dicho foco, tienden generalmente a abrirse, por lo que la sombra proyectada cambiará de tamaño en función de la distancia existente entre el foco y el objeto, y entre ambos y el plano de proyección.

Figura 14. Sombra proyectada por un objeto sometido a luz artificial

El sol es una fuente de luz suficientemente lejana y sus rayos se pueden considerar paralelos entre sí, por lo que el tamaño y forma de la sombra no dependerá de la distancia. La forma y tamaño de la sombra solo dependerá de la forma y tamaño del objeto y del ángulo con el que incidan los rayos con respecto a éste y al plano de proyección.

De este modo, si el objeto utilizado es una placa plana de forma rectangular colocada junto al sensor, formando un ángulo de 90º con su superficie, la sombra proyectada por éste crecerá de manera inversamente proporcional al ángulo formado por los rayos y la superficie del sensor (figura 15), por lo que la cantidad de luz que incidirá sobre el sensor irá disminuyendo.

Se puede asegurar por tanto, que la resistencia del sensor crece a medida que los rayos solares se alejan de la normal a la superficie del sensor, obteniéndose de esta manera una relación directa entre el ángulo de los rayos solares y la resistencia eléctrica del sensor.

Utilizando este procedimiento, se puede convertir una variable difícil de medir, como es el ángulo con el que inciden los rayos solares sobre una superficie, en una variable eléctrica fácil de manejar.

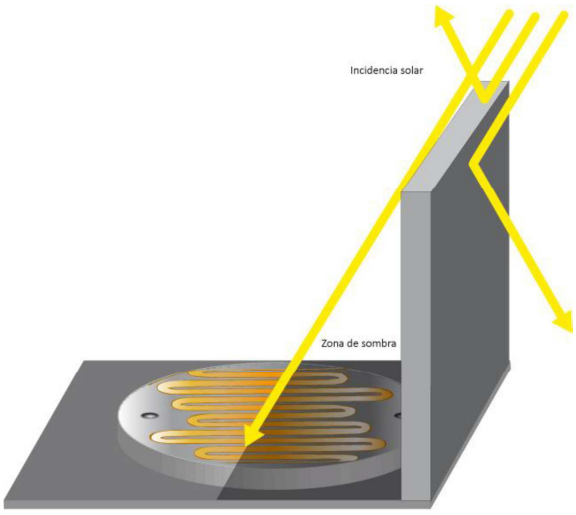


Figura 15. Sombra generada por un objeto sobre la superficie del sensor

Como se ha comentado anteriormente, si los rayos solares inciden desde el lado opuesto al representado en la figura 15, la superficie colocada junto el sensor proyectará su sombra hacia el otro lado, por lo que no influirá en la intensidad luminosa que incide sobre el sensor.

Por este motivo, si se quiere que, sea cual sea la orientación del sol, el sensor detecte el ángulo de incidencia de los rayos solares, es necesario colocar objetos en cada uno de

los laterales del sensor. Así, incidan desde donde incidan los rayos solares, los objetos proyectarán una sombra sobre el sensor y la resistencia de éste se verá incrementada

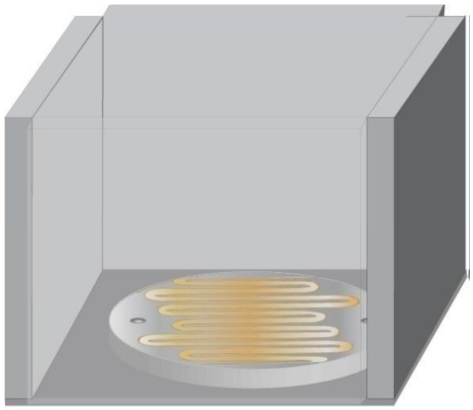


Figura 16. Sensor LDR rodeado por paneles

3.3.1. Funcionamiento de una LDR

Los LDR se fabrican con un cristal semiconductor fotosensible como el sulfuro de cadmio. Si la luz que incide es de alta frecuencia los fotones son absorbidos por la elasticidad del semiconductor dando a los electrones suficiente energía para saltar a la banda de conducción. El electrón libre que resulta (y su hueco asociado) conduce electricidad de tal modo que disminuye la resistencia.

En una LDR la resistencia es inversamente proporcional al nivel de luz que hay en su entorno, es decir, si aumenta la luz, baja la resistencia y viceversa.

Usamos una resistencia de 330Ω en serie con la fotorresistencia para limitar la corriente que circula por la fotorresistencia.

Figura 17. Resistencia de 330Ω

3.4. Servomotores

Un servomotor, es un dispositivo similar a un motor de corriente continua que tiene la capacidad de ubicarse en cualquier posición dentro de su rango de operación y mantenerse estable en dicha posición. Tiene la capacidad de ser controlado, tanto en velocidad como en posición.

Los servomotores se utilizan frecuentemente en radio control y en robótica, aunque su uso no se limita a éstos. Es posible modificar un servomotor para obtener motor continuo que, si bien ya no tiene la capacidad de control del servomotor, conserva la fuerza, velocidad y baja inercia que caracteriza a estos dispositivos.

Por lo general los servomotores suelen tener un rango de 180º aunque existen de 210º, 360º e incluso de rotación continua.

3.4.1. Conexión de un servo

Los servomotores constan de tres cables, dos de ellos son para alimentación, y uno para la señal de control como podemos ver en la siguiente imagen:

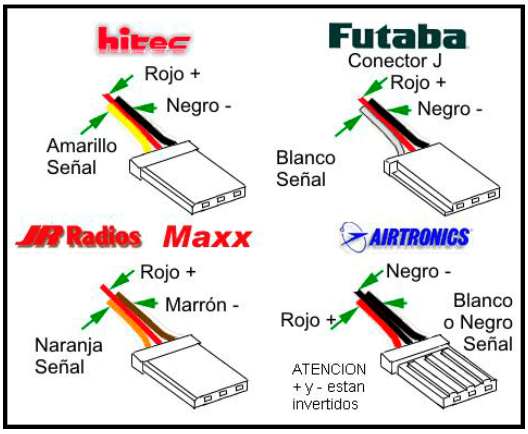


Figura 18. Conectores de algunos fabricantes

Fabricante	Voltaje positivo	Tierra	Señal de control
Futaba	Rojo	Negro	Blanco
Dong Yang	Rojo	Marrón	Naranja
Hobico	Rojo	Negro	Amarillo
Hitec	Rojo	Negro	Amarillo
JR	Rojo	Marrón	Naranja
Airtronics	Rojo	Negro	Naranja
Fleet	Rojo	Negro	Blanco
Krafr	Rojo	Negro	Naranja
E-Sky	Rojo	Negro	Blanco

Tabla 1. Código de colores de conectores de algunos fabricantes

La tensión de trabajo de los servomotores, suele estar comprendida entre 3 y los 7v, siendo 5v la tensión que se utiliza en la mayoría de las aplicaciones fijas donde interviene una fuente de alimentación conectada a la red de energía domiciliaria, y 6v para los casos de alimentación a batería cuando se trata de equipos móviles. En todos los casos se requiere una señal de control de 5v de amplitud.

En cuanto al número de servos que podemos conectar a nuestro Arduino, todo va a depender del consumo que tengan estos en sus especificaciones técnicas, pero se recomienda colocarlos de modo que no supere el consumo máximo del Arduino, recordemos que el consumo de lo que conectemos a los pines de nuestro Arduino no debe exceder de los 40mA.

El rango en el que pueden trabajar estaría (solo es una orientación, para asegurarse mirar especificaciones del fabricante) entre los 3.0 V y 7.0 V.

Cuando necesitemos utilizar más servos, deberemos utilizar un pack de pilas o una fuente de alimentación externa, recordando siempre unificar las masas con nuestro Arduino.

3.4.2. Características de los servomotores

Los servos están conformados por un motor, una caja reductora y un circuito de control.

Destacar que tienen una potencia proporcional para cargas mecánicas. Por consiguiente, los servos tienen un consumo de energía reducido. Aunque habitualmente el fabricante indica su corriente de consumo.

La corriente depende principalmente del par, puede ser superior a un amperio si el servo está enclavado, pero no es muy alto si el servo está libre moviéndose todo el tiempo.

Con esto, podemos decir que un servomotor es un motor especial al que se ha añadido un sistema de control (tarjeta electrónica), un potenciómetro y un conjunto de engranajes.

Con anterioridad los servomotores no permitían que el motor girara 360°, sólo aproximadamente 180°, sin embargo, hoy en día existen servomotores en los que puede ser contralada su posición y velocidad en los 360°.

Los servomotores son comúnmente usados en modelismo como aviones, barcos, helicópteros y trenes para controlar de manera eficaz los sistemas motores y los de dirección.

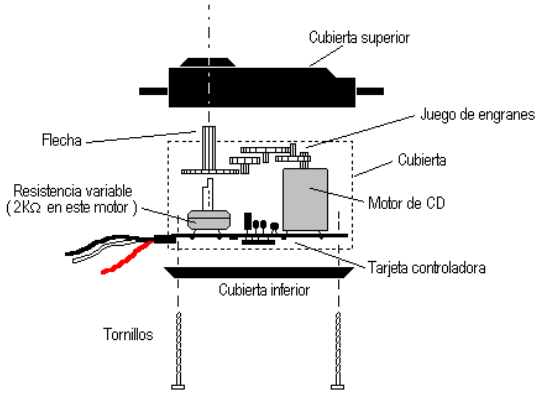


Figura 19. Interior de un servomotor

Los servomotores hacen uso de la modulación por ancho de pulsos (PWM) para controlar la dirección o posición de los motores de corriente continua. La mayoría trabajan en la frecuencia de cincuenta hercios.

3.4.3. Funcionamiento de los servomotores

Bien, los servomotores son en realidad motores de corriente continua con una serie de engranajes que transforman su velocidad, en torque (fuerza) y un sistema de control que utiliza un potenciómetro para saber constantemente la ubicación del eje, este sistema de control, además, responderá a una señal que nosotros le enviemos para establecer la posición del eje…

La señal que introducimos al servo, es una señal *parecida, que no igual*, a la del tipo **PWM** que ya conocemos, digo esto ya que la función PWM de Arduino, recordemos que la utilizábamos para simular una señal analógica con la función *analogWrite(),* de manera que dependiendo del ancho de los pulsos digitales emitidos con una frecuencia de 416Hz, nos emulaba una señal analógica, pero para los servos no queremos eso.

Vamos a emitir pulsos con una frecuencia mucho menor, 50Hz, y serán meramente digitales, sin intención de querer ser nada analógico y dependiendo del ancho del pulso que le enviemos (he ahí el parecido), nuestro servo se situará en una posición o en otra, los pulsos deben llevar una frecuencia de 20ms (los 50Hz) entre ellos para que el servomotor los interprete correctamente, aunque podrían funcionarnos igualmente en un intervalo entre 10ms y 30ms.

Con esto queremos decir que el servomotor va a leer cada 20ms su entrada de señal, y dependiendo de lo que dure el 1 lógico (ancho del pulso de +5v) calculará el grado del eje en el que se debe situar.

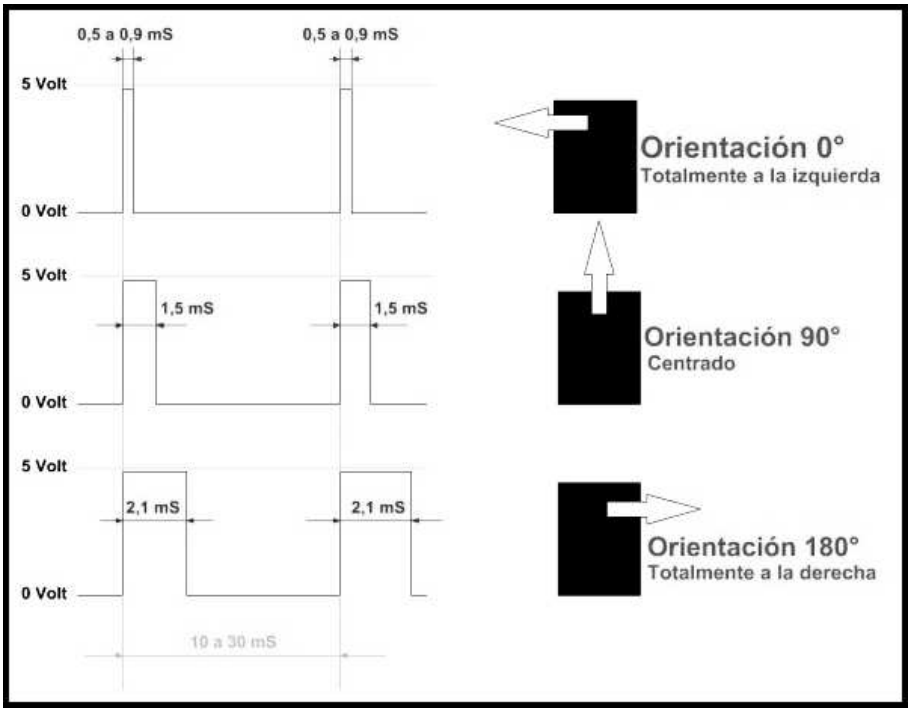


Figura 20. Duración de la señal de control de un servomotor

El Sketch del servomotor

A la hora de controlar el servomotor con nuestro Arduino, tenemos una librería "**Servo.h**" que nos hace la vida mucho más sencilla, pues calcula por nosotros el ancho de pulso que tiene que reproducir para un grado en concreto, esta maravillosa librería deberéis incluirla al inicio de vuestro programa de la siguiente manera:

```
#include <Servo.h>
```

A continuación deberemos crear un objeto del tipo "Servo" para cada uno de los servomotores que queramos controlar:

```
Servo motorInclinación;

Servo motorRotación;
```

Y en el Setup, deberemos asociar nuestros objetos, mediante la instrucción *objeto_x.*attach*(número de pin)* a los pines digitales que nos van a generar los pulsos para establecer la posición de nuestros servos:

```
motorInclinación.attach(3);

motorRotación.attach(5);
```

Y por último nos queda "escribir" en nuestro servo el grado al que queremos se dirija mediante la instrucción *objeto_x.*write*(número de grado o variable que almacene un número de grado)* como vemos, en el siguiente ejemplo, al motor izquierdo lo estamos mandado al grado 90 y al motor derecho al grado que esté almacenado en *posición 180.*

```
motorInclinación.write(90);

motorRotación.write(180;
```

El servomotor de 360° según sus características puede dar cuatro vueltas y media o sea 9π. Practicando con este servo descubrimos que la vuelta completa no correspondía a 360° sino que correspondía a 106 de ahí que si necesitamos de visualizar el valor de ángulo de este servomotor en el programa tenemos que utilizar la función map que hace la conversión de 0-106 a 0-360°.

Otro inconveniente que tuvimos con el mismo servomotor fue que perdía la referencia entre 0 y 15° o sea de 0 a 4 en la escala original de servomotor, pues cuando intentábamos posicionarle en un valor entre 0 y 15° no se quedaba parado en ese valor sino que seguía girando llegando a perder la referencia. Debido a este último problema nos vimos obligados a usar valores mínimos de ángulo por encima de los 15°.

4. DESARROLLO DEL PROYECTO

“Seguidor Lumínico”

Es una estructura que consta de una parte fija y otra móvil.

El mecanismo que vamos a desarrollar consta de dos servomotores, una placa Arduino, un Display LCD, tres piezas de aluminio, un potenciómetro, una placa de madera y una placa sensora (cuatro fotorresistencias, cuatro resistencias).

4.1. Planteamiento y resolución del problema

4.1.1. Planteamiento del problema

Pretendemos construir un seguidor lumínico con dos grados de libertad, o sea, dos movimientos independientes entre sí, un movimiento de inclinación (hacia arriba o hacia abajo) y otro de rotación (hacia la derecha o hacia la izquierda) y que podamos visualizar en todo momento los grados inclinados o rotados.

4.1.2. Resolución del problema

4.1.2.1. Algoritmo

El algoritmo de control toma los datos de los sensores de luz y calcula la posición del servomotor para que esté orientado hacia un máximo de luz dentro de su rango de movimiento.

Como vemos el algoritmo constará de tres partes importantes:

Entrada: Sensores de luz

Procedimiento: Sistema de control

Salida: Posicionamiento

A continuación vamos a definir los parámetros que usaremos en el diagrama de flujo:

S1 – Sensor de luz 1

S2 – Sensor de luz 2

AI – Angulo Inicial del servomotor

Am – Angulo mínimo

AM – Angulo Máximo

VR – Valor de Referencia

$\boldsymbol{valref}$: Es un valor de referencia que nosotros elegimos que representa un margen a partir del cual el sistema de control considera que hay diferencia entre dos divisores de tensión.

$$\boldsymbol{valref = VR} \qquad \in (0,1023)$$

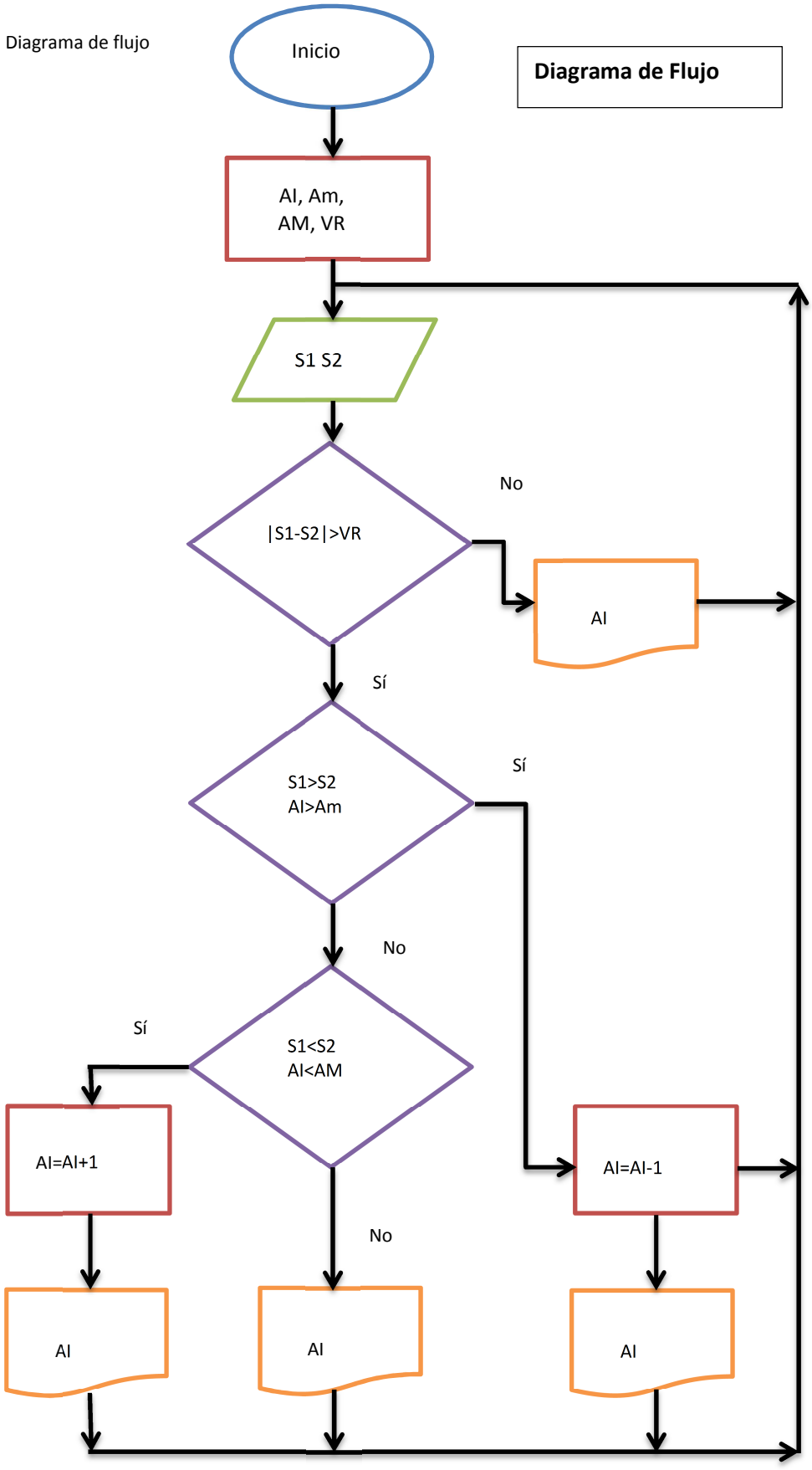
Respecto a la variable $\boldsymbol{valref}$, podemos decir que:

- Un valor pequeño de $\boldsymbol{valref}$ significa que la placa seguirá el haz de luz a la mínima diferencia de tensión que se detecte entre los divisores de tensión de las fotorresistencias.

- Un valor grande de $\boldsymbol{valref}$ implica que ha de haber grandes diferencias de tensión entre los divisores de tensión de las fotorresistencias para que la placa siga el haz de luz.

A continuación representamos el diagrama de flujo de un movimiento pero este diagrama de flujo vale tanto para el movimiento de rotación como el de inclinación. La única diferencia estará en el recorrido máximo de los servomotores ya que el servomotor de responsable del movimiento de inclinación tendrá un recorrido de 180° mientras que el encargado del movimiento de rotación tendrá un recorrido de 360°.

Figura 21. Diagrama de flujo



4.1.2.2. Funcionamiento

Basándonos en el algoritmo del apartado anterior como sensor de luz usaremos una fotorresistencia o LDR. Como esta funciona como una resistencia variable de manera que, cuanta más cantidad de luz reciba, menor será su resistencia, para que quede claro, si en un potenciómetro variábamos la resistencia deslizando un patín por la pista de material resistivo, aquí lo hará la cantidad de luz que reciba la fotorresistencia.

Vamos aprovechar esa característica de la LDR, si añadimos una resistencia en serie podemos utilizar la LDR para hacer el ya conocido divisor de tensión.

Podemos conectarlo de dos maneras diferentes como muestra la figura 22. Si utilizamos el LDR como resistencia inferior del divisor de tensión, nos dará la tensión máxima cuando tengamos el LDR en plena oscuridad, ya que estará oponiendo el máximo de su resistencia al paso de la corriente derivándose esta por v_{out} al completo, si lo utilizamos como resistencia superior, el resultado será el inverso.

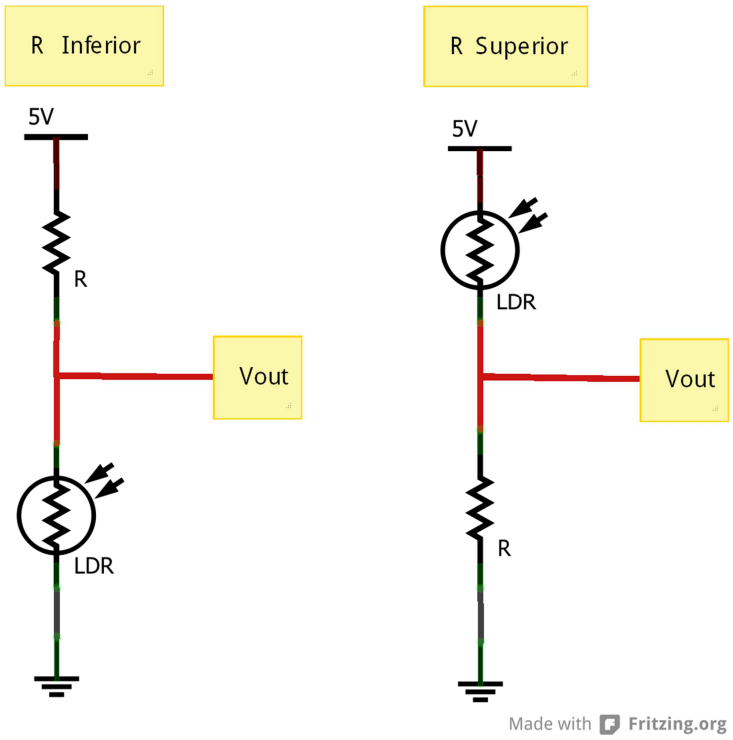


Figura 22. Modo de conexión de una LDR

Tendremos la tensión máxima cuando esté completamente iluminado, ya que se comportará prácticamente como un cortocircuito, con una resistencia de 50Ω o 100Ω.

En nuestro caso lo hemos utilizado como **"R Superior"**, de manera que cuanta más luz haya, más tensión tendremos a la salida de nuestro divisor de tensión.

Vemos que nuestra fotorresistencia configurada como divisor resistivo, nos va a dar **0v** cuando este **completamente a oscuras**, y **+5v** cuando esté **completamente iluminada**, situaciones que pueden ser difíciles de conseguir dependiendo del entorno en el que trabajemos, y por otra parte, ese rango de 0v a 5v es pasado a un rango de **0 a 1023** que es como lo leerá nuestro Arduino.

Para poder llevar a cabo los movimientos de inclinación y rotación necesitaremos de cuatro fotorresistencias divididas en dos grupos de dos fotorresistencias cada grupo.

Grupo1: Ldr1 y Ldr2 →Rotación

Grupo2: Ldr3 y Ldr4 →Inclinación

Cada divisor de tensión de la figura 23 (V_0, V_1, V_2, V_3) la conectamos una entrada analógica del arduino (A_0, A_1, A_2, A_3) respectivamente.

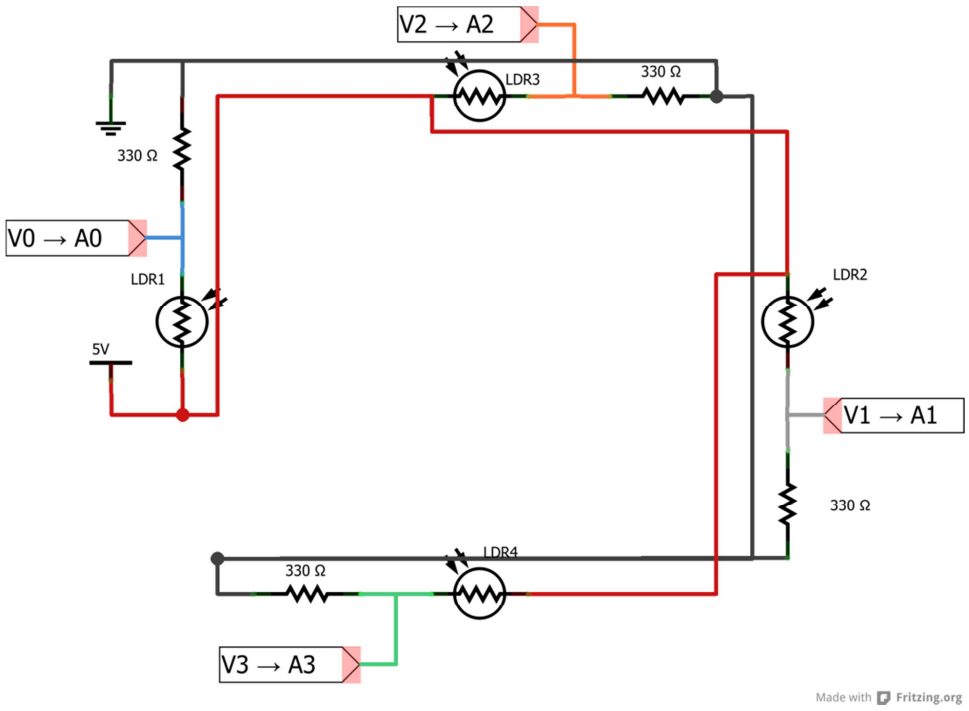


Figura 23. Divisor de tensión de las LDRs

Sensores	**Divisores de tensión**	**Entradas analógicas**
Ldr1 (LdrLeft)	v_0	A_0
Ldr2 (LdrRight)	v_1	A_1
Ldr3 (LdrTop)	v_2	A_2
Ldr4 (LdrDown)	v_3	A_3

Tabla 2. Pines sensores-divisores de tensión- entradas analógicas Arduino

Como hemos visto si la LDR está o no en presencia de luz su valor interpretado por el Arduino estará en un rango de 0 a 1023.

Tensión Analógica (divisor de tensión)	Tensión Digital (placa Arduino)	Intensidad de luz
0	0	Oscuridad
5	1023	Muchísima luz

Tabla 3. Señal analógica y digital en función de la intensidad de luz

El microcontrolador del Arduino siguiendo el código de programa que escribiremos se encargará de comparar las tensiones de los divisores de cada grupo de fotorresistencias.

$|\boldsymbol{A_0 - A_1}| > \boldsymbol{valref_1}$ $\quad valref_1 \in (0,1023)$

$|\boldsymbol{A_2 - A_3}| > \boldsymbol{valref_2}$ $\quad valref_2 \in (0,1023)$

Por tanto el microcontrolador Arduino emitirá una orden que culminará con un movimiento de inclinación y/o rotación según proceda y el Display LCD nos mostrará los grados inclinados y rotados.

4.2. Diseño del circuito electrónico

Hemos elegido el circuito electrónico de la figura 24 y la disposición de los componentes electrónicos de modo que el espacio central quede libre a fin de poderlo aprovechar para futuras aplicaciones. Como vemos se trata de un circuito sencillo con pocos elementos y de bajo coste.

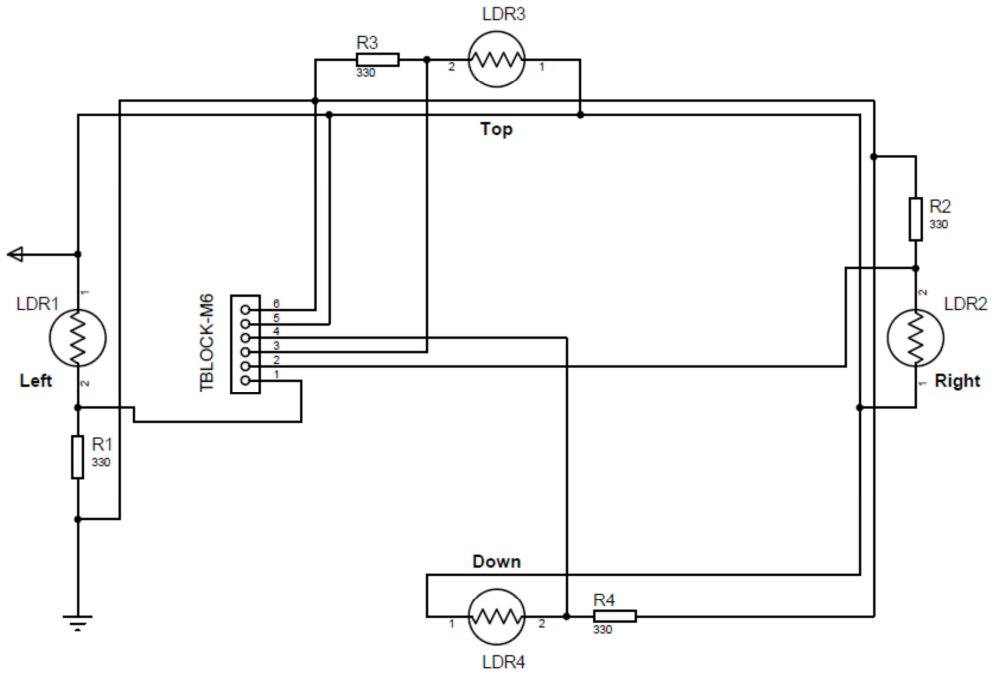


Figura 24. Circuito electrónico

En la red existen varios programas para diseño de circuitos y elaboración de PCB tales como:

Livewire, PCB Wizard, Eagle layout, Pspice, Proteus, Fritzing, etc.

En nuestro caso hemos usado el **PROTEUS 7.8**, desarrollado por Labcenter Electronics; es un entorno integrado diseñado para la realización completa de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño, simulación, depuración y construcción.

Este emulador consta de dos programas principales: Ares e Isis, y los módulos VSM y Electra.

- ISIS.- *Intelligent Schematic Input System* (Sistema de Enrutado de Esquemas Inteligente); es la herramienta que permite simular el esquema eléctrico del proyecto; incorporando una librería de más de 6.000 modelos de dispositivos digitales y analógicos como son: resistencias, microcontroladores, LEDs, relés, microprocesadores, incluyendo fuentes de alimentación, generadores de señales, etc.

- ARES.- AdvancedRouting and Editing Software (Software de Edición y Ruteo Avanzado); es la herramienta de enrutado, ubicación y edición de componentes, se utiliza para la fabricación de placas de circuito impreso (PCB's).

El PROTEUS 7.8 es una herramienta de fácil manejo y muy utilizado en el diseño de circuitos electrónicos y elaboración de PCB.

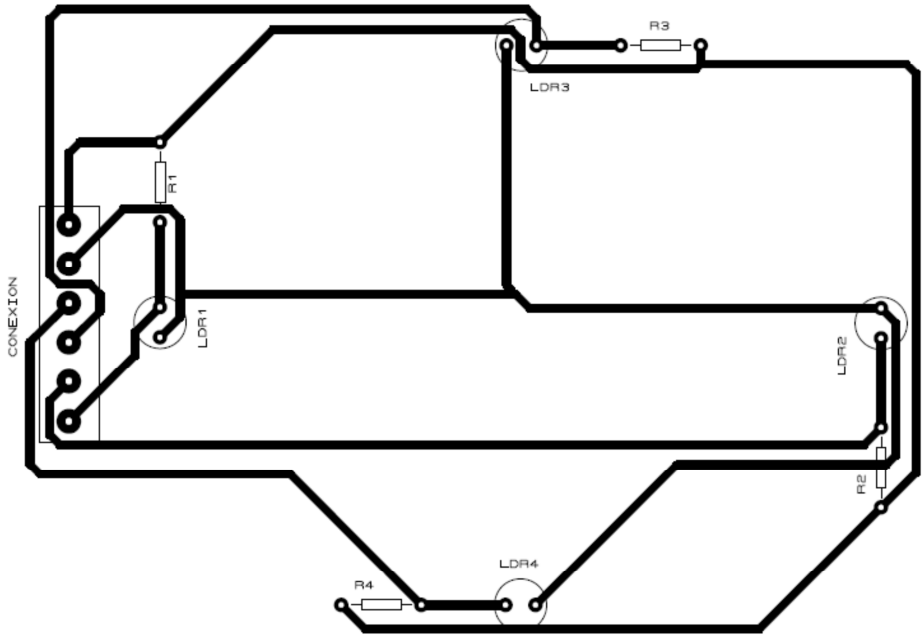


Figura 25. Circuito de pistas

4.3. Elaboración del Circuito Impreso

Primero preparamos los materiales:

1 placa de circuito impreso fotosensible 10x14cm

- 1 transparencia con las pistas dibujadas
- 1 insoladora
- 1 bolsa con revelador
- 1 atacador (ácido clorhídrico + perborato sódico)
- 2 bandejas de 2l
- 1 taladro
- 1 pinzas plásticas
- 1 par de guantes de látex
- 1 par de gafas protectoras
- agua

Figura 26. Materiales usados en la elaboración del circuito impreso

Nos ponemos los guantes, abrimos la insoladora, colocamos la transparencia, colocamos la placa del circuito impreso con la cara de cobre hacia abajo, cerramos la insoladora y la ponemos a insolar 4 minutos mientras vamos mezclando el revelador en 1 litro de agua.

Pasados los 4 minutos retiramos la placa y la introducimos en la mezcla y con ayuda de las pinzas lavamos moviendo hasta que se vea completamente las pistas en la placa. Retiramos la placa, la lavamos con agua y la secamos con papel.

El siguiente paso se recomienda hacerlo en un lugar bien aireado y con las gafas puestas, pues estaremos trabajando con una mezcla que libera gases tóxicos.

Preparamos el atacador 250ml de ácido por 62,5g de perborato e introducimos la placa en la mezcla y vamos moviendo la placa con las pinzas hasta que el cobre haya desaparecido menos el de las pistas.

Retiramos la placa, la lavamos con abundante agua y la secamos con papel. Ojo, la mezcla anterior es corrosiva así que hay que tirarla en desagües de tuberías de PVC.

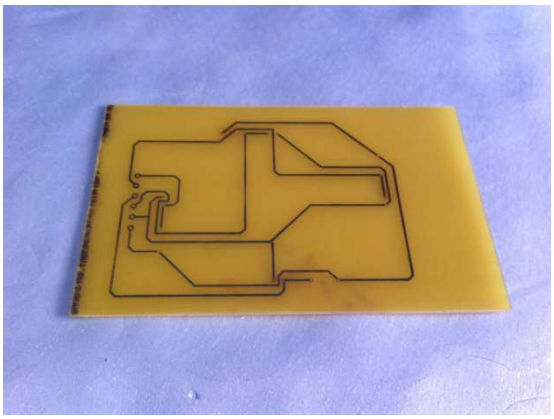


Figura 27. Placa del circuito impreso

Hacemos los agujeros en los puntos indicados con un taladro y ya tenemos lista la placa. Procedemos a soldar las fotorresistencias, resistencias, los cables de alimentación y de señales.

A la hora de soldar los componentes electrónicos hay que prestar mucha atención pues una mala soldadura puede provocar problemas graves en el funcionamiento de los mismos componentes principalmente en las LDRs, pues puede hacer variar mucho la resistencia de éstas.

4.4. Diseño y elaboración de las piezas de aluminio

En un primero momento usamos el prototipo provisional de la figura 28 para conseguir el movimiento de rotación y de inclinación.

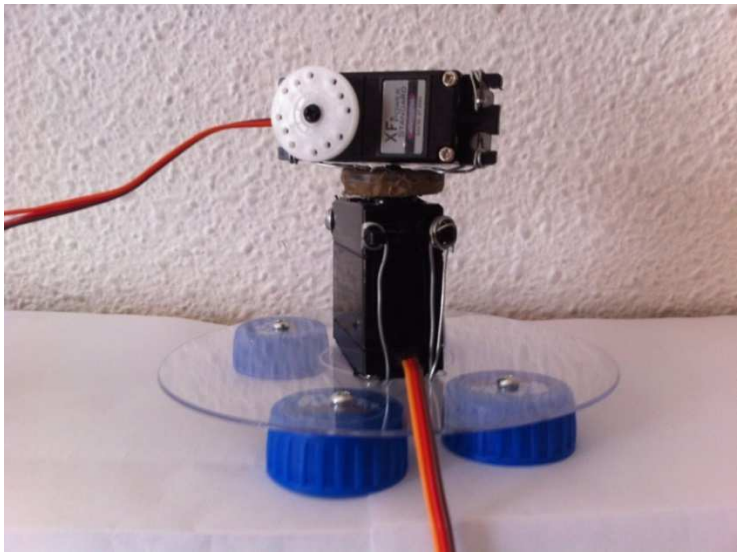


Figura 28. Acoplamiento inicial entre los dos servomotores

Visto que con la estructura en cuestión se conseguía lo deseado, procedemos a elaborar una estructura definitiva basada en esta misma configuración, guardando la funcionalidad de los servomotores de modo a conseguir un mecanismo que tenga dos movimientos independientes entre sí.

Diseñamos los bocetos de las figuras 29, 30 y 31 respectivamente.

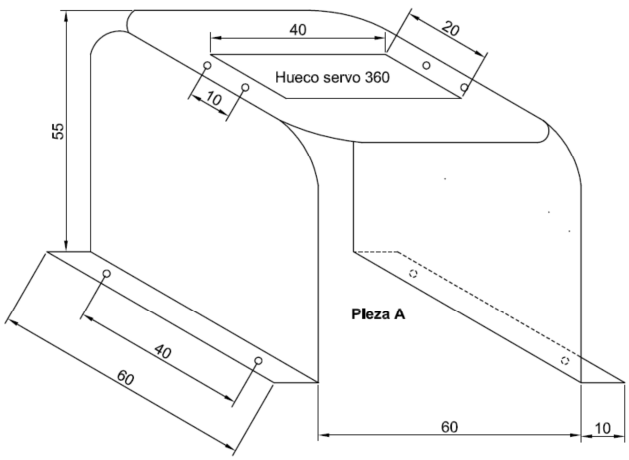


Figura 29. Pieza A

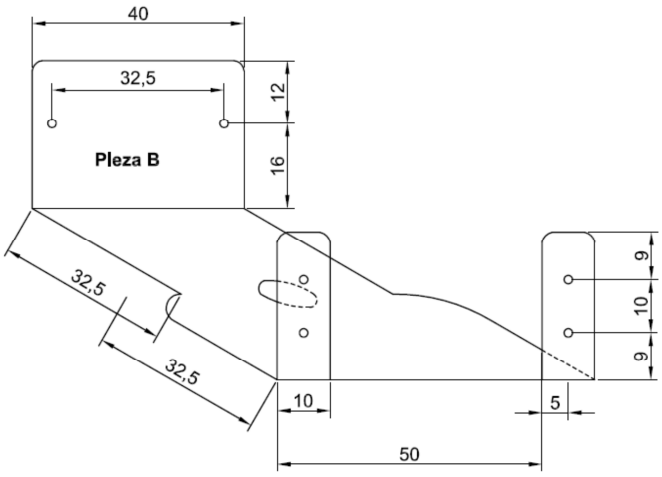


Figura 30. Pieza B

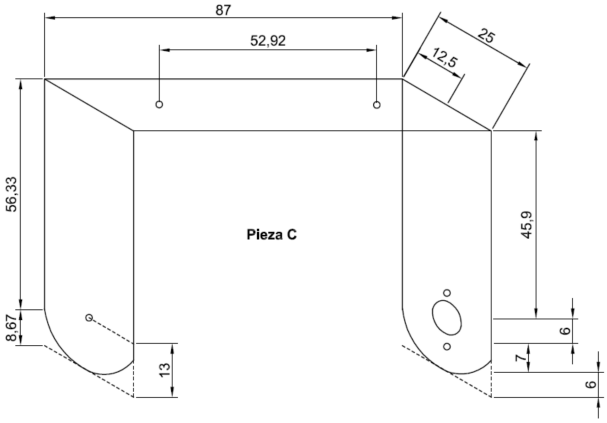


Figura 31. Pieza C

Acto seguido, usando como material una chapa de aluminio de 1mm de grosor, con ayuda de una tijeras para chapa cortamos las 3 láminas de aluminio con respectando sus medidas, después cogemos la lámina **A** en un torno y con ayuda de alicantes vamos doblándola hasta conseguir la forma de la pieza **A** y con un taladro los respectivos agujeros. Hacemos lo mismo y obtenemos las piezas las piezas **B y C**.

En el momento de construir las piezas de aluminio que van a formar la plataforma mecánica, hemos tenido en cuenta que ésta no sobrepase el peso que pueden accionar los servomotores, de ahí que hayamos elegido el aluminio de 1mm de grosor.

4.5. Montaje

En una base de madera de 20x24cm con ayuda de separadores, tornillos, tuercas y procedemos a unir los siguientes elementos formando así una única estructura.

- Pantalla LCD de 4x20
- Placa Arduino uno
- Potenciómetro
- Los servomotores
- Placa seguidora (LDRs y resistencias conectadas entre sí)
- Las piezas de aluminio

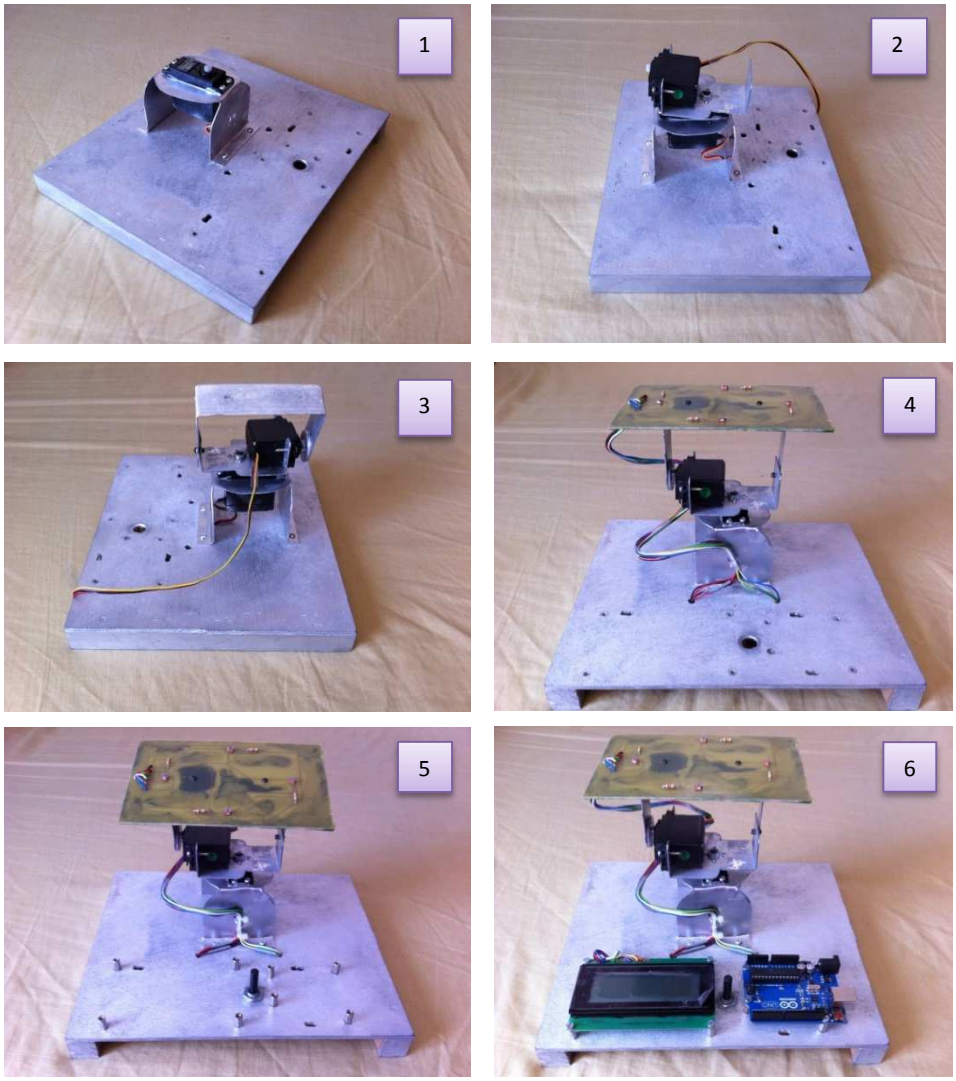


Figura 32. Fases del montaje

- 1. Sobre la base de madera fijamos la pieza A, en el centro de la pieza A sujetamos el servomotor de 360°.
- 2. Sobre el eje del servomotor de 360° sujetamos la pieza B, sobre el lateral de la cual fijamos el servo de 180°.
- 3. A uno de los laterales de la pieza C le unimos al eje del servo de 180° y el otro lateral le unimos a la pieza B por medio de un rodamiento.

- 4. Sujetamos la placa del circuito a la parte central de la pieza C.
- 5. Sujetamos los separadores y el potenciómetro de 10k en la placa.
- 6. Sobre los separadores fijamos el Display LCD y el Arduino.

Llegados a este punto ya tenemos una estructura con dos grados de libertad, o sea, tenemos un mecanismo capaz de realizar dos movimientos independientes aunque de momento no pueda realizar esos movimientos de manera automática ya que todavía no la tenemos conectada a la unidad de control que es la que definirá esos movimientos.

A continuación mostramos mediante tablas las conexiones existentes entre los diferentes componentes del seguidor lumínico.

Pines del LCD	**Pines del Arduino**
Vss	GND
Vcc	5V
V0	Patilla central del potenciómetro
RS	GND
R/W	D12
E	D11
DB4	D10
DB5	D6
DB6	D4
DB7	D2

Tabla 4. Conexiones Arduino - LCD

Divisor de tensión del LDR	**Pines del Arduino**
LDR1	A0
LDR2	A1
LDR3	A2
LDR4	A3

Tabla 5. Conexión Arduino - Divisor de tensión de LDR

Patillas del potenciómetro	**Pines del LCD**	**Pines del Arduino**
Izquierda	Vss	GND
Central	V0	
Derecho	Vcc	5V

Tabla 6. Conexión Potenciómetro - LCD - Arduino

Cables del servomotor	**Pines del Arduino**
Cable negro	GND
Cable marrón	GND
Cable rojo	5V
Cable naranja	D3 pin digital
Cable amarillo	D5

Tabla 7. Conexión servomotor – Arduino

Siguiendo las tablas 4 a 7 hacemos las respectivas conexiones quedando como el circuito de la figura 33.

Habrá que tener un especial cuidado con nuestro seguidor de modo que no tengamos muchos cruces de líneas por encima de los componentes para ello los cruces y conexiones de más de un cable las haremos por debajo del tablero de madera de modo que no sean visibles.

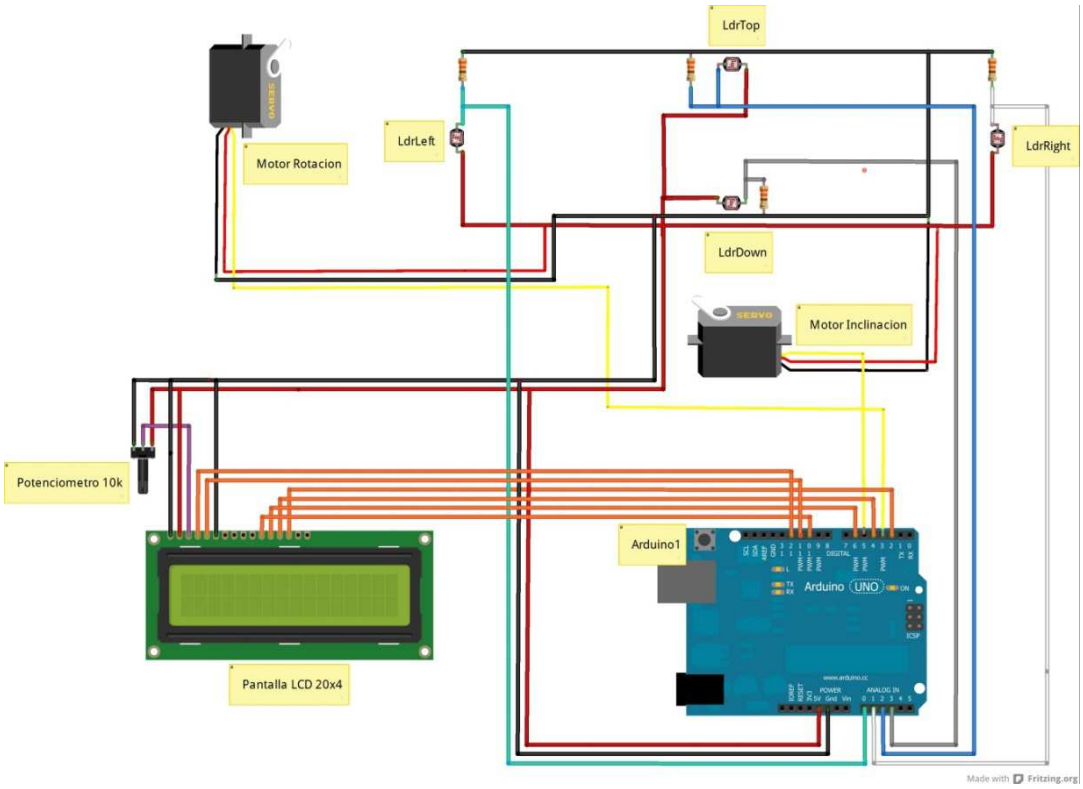


Figura 33. Esquema de conexión final

En nuestro seguidor lumínico tenemos dos tipos de movimiento:

- Movimiento de rotación, el servo de 360°.

- Movimiento de inclinación, el servo 180°.

También destacar que tanto el movimiento de rotación como el de inclinación están controlados cada uno por dos fotocélulas.

Para llevar a cabo estos estos movimientos en cada caso construimos un divisor de tensión entre la fotorresistencia y la resistencia en serie, esta señal la llevamos a la entrada analógica del Arduino Uno.

Comparamos las dos señales de las fotocélulas en cada caso estableciendo una proporción (ldr1-ldr2), la cual es ingresada en un circuito de potencia en la placa Arduino y según proceda, la placa del seguidor rotará en un sentido u otro (a izquierdas o a derechas), si es el caso de inclinación, se inclinará a un lado u otro (hacia arriba o hacia abajo).

En cuanto a la visualización del ángulo rotado o inclinado, lo efectuamos con el Display LCD de 4x20.

El potenciómetro de 10KΩ nos sirve para regular adecuadamente el contraste de la pantalla LCD.

Así queda el seguidor lumínico después de hacer todas las conexiones pertinentes.

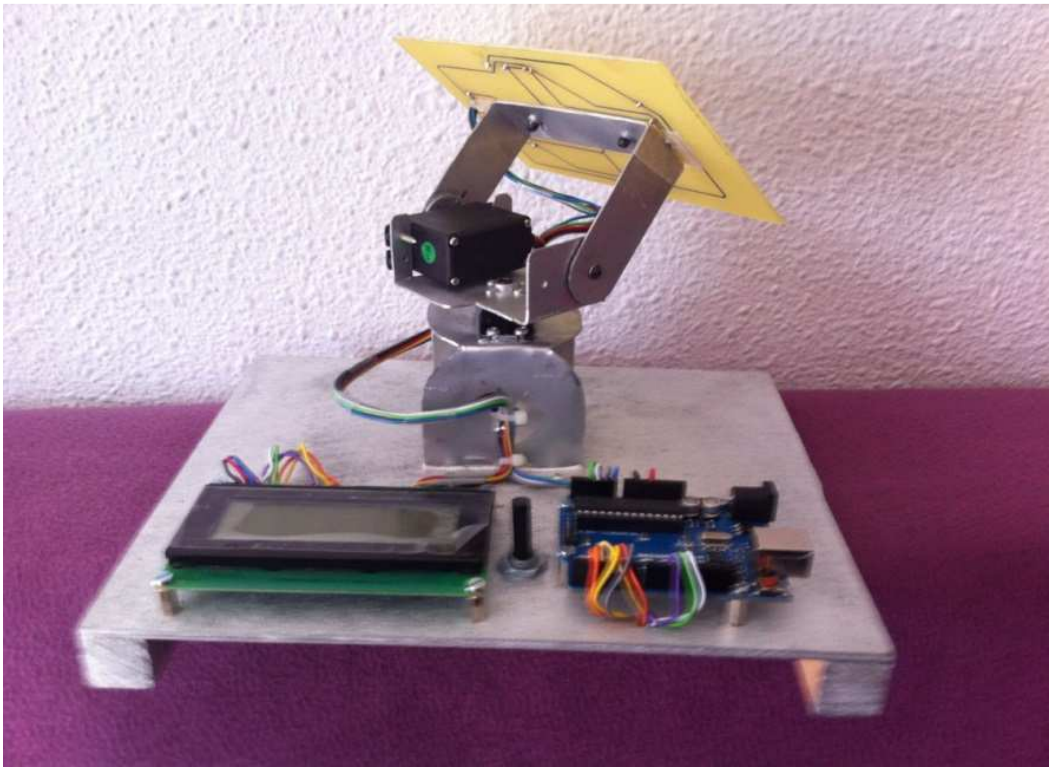


Figura 34. Seguidor Lumínico Resultante

5. PRUEBAS Y RESULTADOS

Recordemos que el objetivo propuesto es construir un seguidor lumínico con dos grados de libertad capaz de seguir un haz de luz. Pues una vez construido y programado el seguidor lumínico, procedemos a realizar varias pruebas para ver su funcionamiento. Las pruebas se realizan en una habitación de 390x290x240mm (larguraxanchuraxaltura) de 0 lux.

Para llevar a cabo dichas pruebas reunimos varias fuentes de luz de diferentes potencias, a continuación vamos a llevar a cabo los siguientes experimentos:

5.1. Primer Experimento

Prueba 1

1. Empezamos con la fuente más pequeña de 0.1w, y el valor más pequeño de valref (***valref*** **= 1),** hacemos incidir el haz de luz sobre la placa desde una distancia aproximada de 3 metros, nos vamos acercando progresivamente a la placa seguidora hasta que ésta siga el haz de luz, instante en el que medimos la distancia manualmente con el metro, desde la fuente al centro de la placa (d = 20cm).

2. Con la misma fuente repetimos el punto (1) pero variando valref, mostramos los resultados obtenidos en la tabla 8.

valref	1	5	10	45	65	85
Dist. detección	20	11	9	3	2	1

Tabla 8. Distancia de detección con la fuente de 0.1W

Prueba 2

Cambiamos la fuente de luz por una de 0.5w y repetimos los puntos 1 y 2 de la prueba 1, los resultados obtenidos los mostramos en la tabla 9.

valref	1	5	10	45	65	85
Dist.detección (cm)	63	44	35	17	13	8

Tabla 9. Distancia de detección con la fuente de 0.5W

Aquí observamos que al haber utilizado una fuente de mayor potencia que la anterior, también lo son las distancias máximas de detección para el mismo valor de valref.

Prueba 3

Cambiamos la fuente de luz por una de 40w y repetimos los puntos 1 y 2 de la prueba 1, los resultados obtenidos los mostramos en la tabla 10.

valref	1	5	10	45	65	85
Dist. detección (cm)	160	92	69	39	35	31

Tabla 10. Distancia de detección con la fuente de 40W

De nuevo observamos que al utilizar una fuente de mayor potencia que la anterior la distancia máxima de detección es mayor a igual valor de valref.

Prueba 4

Cambiamos la fuente de luz por una de 60w y repetimos los puntos 1 y 2 de la prueba 1, los resultados obtenidos los mostramos en la tabla 11.

Valref	1	5	10	45	65	85
Dist. detección (cm)	170	102	74	41	37	33

Tabla 11. Distancia de detección con la fuente de 60W

Prueba 5

Cambiamos la fuente de luz por una de 100w y repetimos los puntos 1 y 2 de la prueba 1, los resultados obtenidos los mostramos en la tabla 12.

valref	1	5	10	45	65	85
Dist. detección (cm)	220	130	93	59	55	45

Tabla 12. Distancia de detección con la fuente de 100W

La observación hecha en la prueba 2 y haciendo una comparación en los resultados de las pruebas sucesivas con sus respectivas tablas, vemos que existe una clara evidencia de que para el mismo valor del parámetro valref la distancia de detección aumenta con la potencia de la fuente.

Cuando incidimos un haz de luz sobre la placa seguidora La máxima distancia a la que puede ser detectada el haz de luz depende de la potencia de la fuente utilizada, de modo que a mayor potencia de la fuente tendremos mayor distancia de detección.

A continuación mostramos los datos y los resultados obtenidos en la tabla 13, que no es nada más que la realización de más pruebas amen a las realizadas en los apartados anteriores.

$valref$	**Fuentes de luz (w)**				
	0.1	0.5	40	60	100
	Distancia de detección (cm)				
3	16.0	49.0	123.0	135.0	155.0
15	6.0	29.0	63.0	66.0	86.0
20	5.0	27.0	56.0	63.0	76.0
25	4.5	25.0	44.0	55.0	70.0
35	4.0	20.0	42.0	45.0	63.0
55	3.0	15.0	37.0	39.0	55.0
75	1.5	11.0	33.0	35.0	48.0
95	0.5	7.5	29.0	31.0	43.0
105	-	7.0	27.0	29.0	40.0
110	-	6.0	26.0	27.0	36.0
115	-	5.0	24.0	25.0	30.0
120	-	3.0	10.0	13.5	15.0

Tabla 13. Valores adicionales del primer experimento

5.2. Segundo Experimento

Esta prueba consiste en interrumpir de forma parcial o completa el haz de luz que incide sobre la placa seguidora y ver cómo responde el seguidor lumínico.

Hacemos incidir un haz de luz sobre la placa seguidora, ésta inicia un movimiento para posicionarse perpendicularmente al haz de luz.

Obstaculizamos parcialmente el haz de luz incidente intercalando una superficie opaca (una cartulina negra que no deja pasar la luz o crea sombra parcialmente) entre el haz de luz y la superficie seguidora, la placa seguidora inicia una maniobra orientándose hacia donde hay más luz.

Obstaculizamos completamente el haz incidente interponiendo una superficie opaca (una cartulina negra que no deja pasar la luz o crea sombra completamente) entre el haz de luz y la superficie seguidora provocando sombra sobre la placa seguidora completamente, el seguidor de luz se detiene.

5.3. Tercer Experimento

Consiste en ver el comportamiento del seguidor lumínico en ausencia de luz e incidiendo un haz de luz en el centro de la placa seguidora.

Vemos que en ausencia de luz la placa seguidora no se mueve. Al igual que incidiendo el haz de luz en el centro de la placa seguidora. A incidir el haz de luz en el centro de la placa, las fotorresistencias del mismo grupo (inclinación y rotación) las llega la misma intensidad de luz ya que son equidistantes, por tanto la placa seguidora no se mueve.

5.4. Cuarto Experimento

Este experimento consiste sacar nuestro seguidor lumínico al exterior y ver cómo se comporta en presencia de los rayos solares. Al llevar el seguidor lumínico al exterior en ausencia del sol (rayos salares), cuando le alimentamos a través del PC este se inicializa y se posiciona en la posición inicial definida por nosotros en el programa y cuando los rayos solares inciden sobre la placa seguidora esta se orienta en la dirección de los rayos solares. Esa orientación la conseguíamos cuando utilizábamos valores de referencia dentro del intervalo de 20 y 109.

La idea era dejar el seguidor en un lugar donde los rayos solares puedan incidir sobre la placa seguidora durante unas 3 o 4 horas e ir observando los grados de inclinación y rotación cada 15 o 20 minutos, o sea a medida que el sol vaya describiendo su trayectoria.

En un principio ya sabíamos que este experimento sería el más complicado y el que más tiempo nos iba a llevar, pues la fuente de luz es el sol y no se trata de una fuente que nosotros podamos usarla en el momento que queramos, pues esta sólo está disponible en las horas del sol, es decir, desde el amanecer hasta el ocaso y dependiendo de los factores meteorológicos (cielo nublado, una nube tapando el sol, etc.).

Este seguimiento durante 3 o 4 horas no hemos podido llevar a cabo pues en los días que nos pusimos a realizar esta prueba tuvimos muy pocos minutos de sol.

6. CONCLUSIONES

Tras reunir el material necesario para el proyecto, no hemos procedido de inmediato el montaje de los mismos, si no que los hemos estado probando en cuanto a su funcionamiento individualmente y con especial atención a los servos y el Display LCD.

De modo que descubrimos que el servo encargado del movimiento de rotación, la vuelta completa no correspondía a 360° sino que correspondía a 106 de ahí que en el programa hemos utilizado la función **map** que hace la conversión de 0-106 a 0-360°.

Otro problema que tuvimos fue que el mismo servo perdía la referencia entre 0 y 15° o sea de 0 a 4 en la escala original de servo, pues cuando le dábamos un valor entre 0 y 15° no se quedaba parado en ese valor sino que seguía girando llegando a perder la referencia. Como solución tomamos como ángulo mínimo o inicial 15°.

Con los resultados obtenidos de los experimentos llevados a cabo a fin de ver el funcionamiento de nuestro seguidor lumínico, deducimos lo siguiente:

En ausencia de luz (0 lux)

El seguidor lumínico si está en reposo permanece en reposo y si está en movimiento se detiene.

En presencia de luz artificial

Si tenemos una única fuente de luz el seguidor seguirá el haz de luz de la fuente y si tenemos más de una fuente de luz seguirá el haz de luz de la fuente de mayor intensidad.

La distancia de detección depende de la intensidad de luz de la fuente de modo que a mayor intensidad de luz tendremos mayor distancia de detección.

En presencia de luz natural (el sol)

Debido a las inclemencias del tiempo no hemos podido realizar debidamente las pruebas deseadas por la escasa presencia del sol que caracteriza esta época del año, ya que para realizar dichas pruebas necesitamos más horas de sol debido a que esta fuente no la podemos controlar ni dirigirla, pues el sol sigue una trayectoria predeterminada desde el amanecer hasta el ocaso. En otras palabras para observar el seguimiento de nuestro seguidor lumínico a los rayos solares necesitamos más tiempo. No obstante cuando inicializamos el seguidor lumínico vemos que este se posiciona inmediatamente en la dirección de los rayos solares.

Durante este proyecto, hemos podido adquirir muchas competencias tales como:

- Entender cómo funciona un sistema mecatrónico interactivo, entendiendo sus partes principales: sensor, actuador y controlador en una estructura física.
- Entender cómo se calibra un sensor LDR para que de valores dentro de un rango de tensión. Esto es aplicable a otros sensores como por ejemplo un potenciómetro.
- Aprender los fundamentos del lenguaje de programación de Arduino: sus principales funciones y el uso de variables globales y locales.
- Entender cómo funciona un algoritmo de control en bucle.
- Familiarizarse con la soldadura de cables y conectores.

A base de esfuerzo y dedicación hemos logrado los objetivos de nuestro proyecto.

Nuestro seguidor lumínico es capaz de seguir haces de luz de fuentes naturales y artificiales.

En cuanto a los costos de los materiales utilizados si bien todavía no se ha realizado un análisis detallado de éstos, los mismos han sido bajos y podemos afirmar que son inferiores a 100€, con lo cual podemos concluir que hemos elaborado un prototipo "seguidor lumínico" experimental de bajo costo

Estamos satisfechos de haber hecho este proyecto, gracias a este proyecto hemos ampliado nuestros conocimientos acerca de Arduino, ya que tiene infinidades de aplicaciones. Descubrimos que su entorno de programación es fácil después de ver algunos ejemplos de códigos en red y haber recopilado mucha información.

Por último, queremos agradecer a nuestro tutor Don Francisco Javier García y técnico de laboratorio Don Luis Almudí, por la ayuda que nos han dispensado.

7. ANEXOS

7.1. Programación

7.1.1. Código del Programa

```
#include <LiquidCrystal.h>

#include <Servo.h>

LiquidCrystal lcd(12, 11, 10, 6, 4, 2);

Servo servoInclinacion;

Servo servoRotacion;

int motorInclinacion=5;

int motorRotacion=3;

int ldrLeft = 0; / / Cable de Azul

int ldrRight = 1; / / Gris

int ldrTop = 2; / / Blanco

int ldrDown = 3; / / Verde

int valref = 5;

int valInclinacion = 135;

int valRotacion = 53; // equivalente a 180 grados

int valLeft;

int valRight;

int valTop;

int valDown;

void setup()

{

lcd.begin(20, 4);
```

```
lcd.setCursor(0, 0);

lcd.print("E.I.I. 2013");

lcd.setCursor(0,1);

lcd.print("Seguidor de luz");

servoInclinacion.attach(motorInclinacion);

servoRotacion.attach(motorRotacion);

}

void loop()

{

valLeft  = analogRead(ldrLeft);

valRight = analogRead(ldrRight);

valTop   = analogRead(ldrTop);

valDown  = analogRead(ldrDown);

// Parte de Inclinación

if(valTop-valDown>valref ||valTop-valDown<-valref)

{

if(valTop>valDown&&valInclinacion>90){

valInclinacion--;

}

else if(valDown>valTop&&valInclinacion<160){

valInclinacion++;

}

}

// parte de la rotación
```

```
if(valLeft-valRight>valref ||valLeft-valRight<-valref)
{
if(valRight>valLeft&&valRotacion>5){
valRotacion--;
}
else if(valLeft>valRight&&valRotacion<105){
valRotacion++;
}
}
servoInclinacion.write(valInclinacion);
lcd.setCursor(0,2);
lcd.print("Incl:");
lcd.print(servoInclinacion.read());
lcd.print(" grados ");
servoRotacion.write(valRotacion);
int Real = map(valRotacion,0,106,0,360);
lcd.setCursor(0,3);
lcd.print("Rot:");
lcd.print(Real);
lcd.print(" grados ");
delay(50);
}
```

7.1.2. Instrucciones usadas en el programa

Estructura de un programa

La estructura básica del lenguaje de programación de Arduino es bastante simple y se compone de al menos dos partes. Estas dos partes necesarias, o funciones, encierran bloques que contienen declaraciones, estamentos o instrucciones.

void setup(){

Estamentos;

}

void loop ()

{

Estamentos;

}

En donde **setup**() es la parte encargada de recoger la configuración y **loop**() es la que contiene el programa que se ejecutará cíclicamente (de ahí el termino loop –bucle-). Ambas funciones son necesarias para que el programa trabaje.

La función de configuración debe contener la declaración de las variables. Es la primera función a ejecutar en el programa, se ejecuta sólo una vez, y se utiliza para configurar o inicializar pinMode (modo de trabajo de las E/S), configuración de la comunicación en serie y otras.

Setup ()

La función **setup** () se invoca una sola vez cuando el programa empieza. Se utiliza para inicializar los modos de trabajo de los pines, o el puerto serie. Debe ser incluido en un programa aunque no haya declaración que ejecutar.

loop ()

Después de llamar a **setup**(), la función **loop**() hace precisamente lo que sugiere su nombre, se ejecuta de forma cíclica, lo que posibilita que el programa este respondiendo continuamente ante los eventos que se produzcan en la tarjeta.

{} Entre llaves

Las llaves sirven para definir el principio y el final de un bloque de instrucciones. Se utilizan para los bloques de programación **setup** (), **loop** (), **if**.., etc.

Una llave de apertura "{"siempre debe ir seguida de una llave de cierre "}", si no es así el programa dará errores.

El entorno de programación de Arduino incluye una herramienta de gran utilidad para comprobar el total de llaves. Sólo tienes que hacer click en el punto de inserción de una llave abierta e inmediatamente se marca el correspondiente cierre de ese bloque (llave cerrada).

; Punto y coma

El punto y coma "**;**" se utiliza para separar instrucciones en el lenguaje de programación de Arduino. También se utiliza para separar elementos en una instrucción de tipo "bucle **for** ".

Nota: Olvidarse de poner fin a una línea con un punto y coma se traducirá en un error de compilación. El texto de error puede ser obvio, y se referirá a la falta de una coma, o puede que no. Si se produce un error raro y de difícil detección lo primero que debemos hacer es comprobar que los puntos y comas están colocados al final de las instrucciones.

// línea de comentarios

Una línea de comentario empieza con / / y terminan con la siguiente línea de código. Al igual que los comentarios de bloque, los de línea son ignoradas por el programa y no ocupan espacio en la memoria.

Una línea de comentario se utiliza a menudo después de una instrucción, para proporcionar más información acerca de lo que hace esta o para recordarla más adelante.

Variables

Una variable es una manera de nombrar y almacenar un valor numérico para su uso posterior por el programa. Como su nombre indica, las variables son números que se pueden variar continuamente en contra de lo que ocurre con las constantes cuyo valor nunca cambia. Una variable debe ser declarada y, opcionalmente, asignarle un valor.

Una variable puede ser cualquier nombre o palabra que no sea una palabra reservada en el entorno de Arduino.

Declaración de variables

Todas las variables tienen que declararse antes de que puedan ser utilizadas. Para declarar una variable se comienza por definir su tipo como **int** (entero), **long** (largo), float (coma flotante), etc, asignándoles siempre un nombre, y, opcionalmente, un valor inicial. Esto sólo debe hacerse una vez en un programa, pero el valor se puede cambiar en cualquier momento usando aritmética y reasignaciones diversas. Una variable puede ser declarada en una serie de lugares del programa y en función del lugar en donde se lleve a cabo la definición esto determinará en que partes del programa se podrá hacer uso de ella.

Int;

Enteros son un tipo de datos primarios que almacenan valores numéricos de 16 bits sin decimales comprendidos en el rango 32,767 a -32,768.

Aritmética

Los operadores aritméticos que se incluyen en el entorno de programación son suma, resta, multiplicación y división. Estos devuelven la suma, diferencia, producto, o cociente (respectivamente) de dos operandos.

Asignaciones compuestas

Las asignaciones compuestas combinan una operación aritmética con una variable asignada. Estas son comúnmente utilizadas en los bucles tal como se describe más adelante. Estas asignaciones compuestas pueden ser:

x **++** // igual que x = x + 1, o incrementar x en + 1

x **--** // igual que x = x - 1, o decrementar x en -1

Operadores de comparación

Las comparaciones de una variable o constante con otra se utilizan con frecuencia en las estructuras condicionales del tipo if.. para testear si una condición es verdadera. En los ejemplos que siguen en las próximas páginas se verá su utilización práctica usando los siguientes tipo de condicionales:

x **==** y // x es igual a y

x **!=** y // x no es igual a y

x **<** y // x es menor que y

x **>** y // x es mayor que y

x **<=** y // x es menor o igual que y

x **>=** y // x es mayor o igual que y

Operadores lógicos

Los operadores lógicos son usualmente una forma de comparar dos expresiones y devolver un VERDADERO o FALSO dependiendo del operador. Existen tres operadores lógicos**, AND (&&), OR (||) y NOT (!),** que a menudo se utilizan en estamentos de tipo **if**…

if (si)

if es un estamento que se utiliza para probar si una determinada condición se ha alcanzado, como por ejemplo averiguar si un valor analógico está por encima de un cierto número, y ejecutar una serie de declaraciones (operaciones) que se escriben dentro de llaves, si es verdad. Si es falso (la condición no se cumple) el programa salta y no ejecuta las operaciones que están dentro de las llaves, El formato para if es el siguiente:

Dentro de las estructuras **if**, cuando se pregunte por un valor se debe poner el signo doble de igual "**==**".

if… else (si….. sino ..)

if… else viene a ser un estructura que se ejecuta en respuesta a la idea "si esto no se cumple haz esto otro". Por ejemplo, si se desea probar una entrada digital, y hacer una cosa si la entrada fue alto o hacer otra cosa si la entrada es baja, usted escribiría que de esta manera:

```
if (inputPin == HIGH) // si el valor de la entrada inputPin es alto)

{

instruccionesA; //ejecuta si se cumple la condición

}

else

{

instruccionesB; //ejecuta si no se cumple la condición

}
```

Nota: Un estamento de tipo if prueba simplemente si la condición dentro del paréntesis es verdadera o falsa. Esta declaración puede ser cualquier declaración válida.

analogRead(pin);

Lee el valor de un determinado pin definido como entrada analógica con una resolución de 10 bits. Esta instrucción sólo funciona en los pines (0-5). El rango de valor que podemos leer oscila de 0 a 1023.

delay(ms);

Detiene la ejecución del programa la cantidad de tiempo en ms que se indica en la propia instrucción. De tal manera que 1000 equivale a 1seg.

```
delay(1000); // espera 1 segundo
```

 # Include

Se utiliza para incluir bibliotecas externas en el boceto. Esto le da al programador acceso a un gran grupo de bibliotecas estándar de C (grupos de funciones pre- hechas), y también bibliotecas escrita especialmente para Arduino. Tenga en cuenta que **# include** no tiene un terminador de punto y coma.

servo.**attach**(pin);

Asocia la variable Servo a un pin.

servo: una variable de tipo Servo

pin: el número de pin con el que el servo está asociado

servo.**write**(ángulo);

Escribe un valor en el servo, controlando el eje en consecuencia. En un servo estándar ajustará el ángulo del eje (en grados), moviendo el eje hasta la orientación deseada.

servo: una variable tipo servo

ángulo: el valor a escribir en el servo, de 0° a 180° o 360°.

servo.**read**();

Lee el ángulo actual del servo (el valor que se ha pasado en la última llamada a **write**).

servo: una variable de tipo servo

angulo: el valor a escribir en el servo, de 0° a 180° o 360°.

LiquidCrystal(rs, enable, d3, d2, d1, d0);

Crea una variable de tipo LiquidCrystal. La pantalla se puede controlar por medio de 4 u 8 líneas de datos. En el primer caso, omitir los números de pines para d4 hasta d7 y dejar esos pines no conectados. El pin rw pueden ser conectado a masa en lugar de conectarse a un pin de Arduino, si es así, omítelo de los parámetros de esta función.

rs: número del pin de Arduino que está conectado al pin rs del lcd

rw: número del pin de Arduino que está conectado al pin rw del lcd (*opcional*)

enable: número del pin de Arduino que está conectado al pin enable del lcd.

lcd.**begin**(cols, rows);

Especifica las dimensiones (ancho y alto) del display lcd.

lcd: una variable de tipo LiquidCrystal

cols: número de columnas que tiene el display

rows: número de filas que tiene el display

lcd.**setCursor**(col, row);

lcd: una variable de tipo LiquidCrystal

col: la columna donde posicionar el cursor

row: la fila donde posicionar el cursor

lcd.**print**("data");

Imprime un texto en el lcd.

cd: una variable de tipo LiquidCrystal

data: los datos a imprimir (char, byte, int, long, or string)

lcd.**println**(val);

Imprime los datos en el lcd como texto legible ASCII seguido de un carácter de retorno de carro. Este comando tiene las mismas formas que **lcd.print**().

val: el valor a imprimir - cualquier tipo de datos

7.2. Display LCD 20x4

Systronix 20x4 LCD

Brief Technical Data

Here is brief data for the Systronix 20x4 character LCD. It is a DataVision part and uses the Samsung KS0066 LCD controller. It's a clone of the Hitachi HD44780. We're not aware of any incompatabilities between the two - at least we have never seen any in all the code and custom applications we have done.

This 20x4 LCD is electrically and mechanically interchangeable with 20x4 LCDs from several other vendors. The only differences we've seen among different 20x4 LCDs are:

1) LED backlight brightness, voltage and current vary widely, as does the quality of the display

2) There is a resistor “Rf” which sets the speed of the LCD interface by controlling the internal oscillator frequency. Several displays we have evaluated have a low resistor value. This makes the display too slow. Looking at the Hitachi data sheet page 56, it appears that perhaps the “incorrect” resistor is really intended for 3V use of the displays.

At 5V the resistor Rf should be 91 Kohms. At 3V it should be 75 Kohms. Using a 3V display at 5V is acceptable from a voltage standpoint (the display can operate on 3-5V) but the oscillator will then be running too slowly. One fix is to always check the busy flag and not use a fixed time delay in your code, then it will work regardless of the LCD speed. The other option is to always allow enough delay for the slower display.

All Systronix 20x4 LCDs have the 91 Kohm resistor and are intended for 5V operation.

ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Standard Value			Unit
		Min.	Typ.	Max.	
Supply Voltage for Logic	V_{DD}-	0	—	7.0	V
Supply Voltage for LCD Driver	V_{DD}-V_{EE}	—	—	13.5	V
Input Voltage	V_I	V_{SS}	—	V_{DD}	V
Operature Temp.	Topr	0	—	50	°C
Storage Temp.	Tstg	-20	—	70	°C

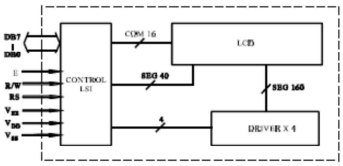
ELECTRICAL CHARACTERISTICS (REFLECTIVE TYPE)

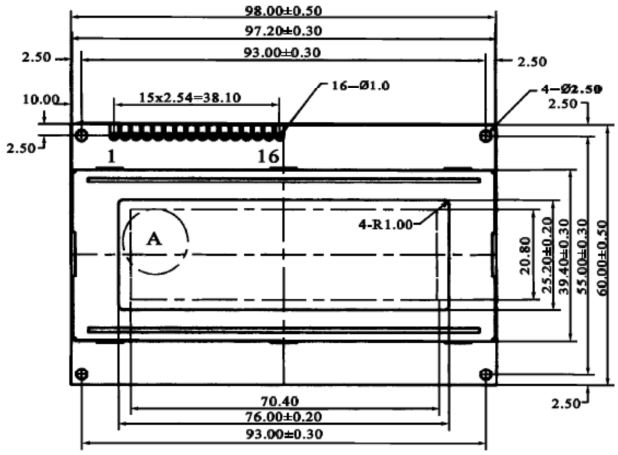
Item	Symbol	Test Condition	Standard Value			Unit
			Min.	Typ.	Max.	
Input "High" Voltage	V_{IH}	—	2.2	—	V_{EE}	V
Input "Low" Voltage	V_{IL}	—	—	—	0.6	V
Output "High" Voltage	V_{OH}	I_{OH}=0.2mA	2.2	—	—	V
Output "Low" Voltage	V_{OL}	I_{OL}=1.2mA	—	—	0.4	V
Supply Current	I_{DD}	V_{DD}=5.0A	—	2.5	4.0	mA

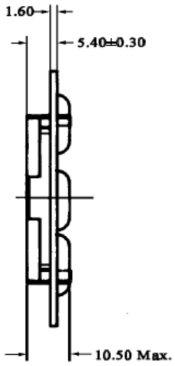
PIN FUNCTIONS

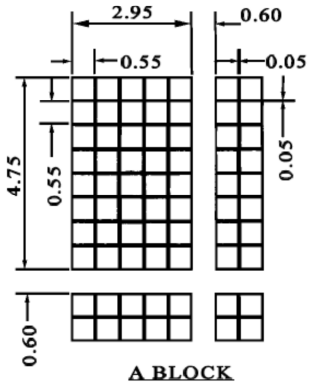
No	Symbol	Function	No	Symbol	Function
1	V_{SS}	GND, 0V	10	DB3	Data Bus
2	V_{DD}	+5V	11	DB4	—
3	V_{EE}	for LCD Drive	12	DB5	—
4	RS	Function Select	13	DB6	—
5	R/W	Read/Write	14	DB7	—
6	E	Enable Signal	15	LEDA	LED Power Supply
7-9	DB0-DB2	Data Bus Line	16	LEDA	

BLOCK DIAGRAM



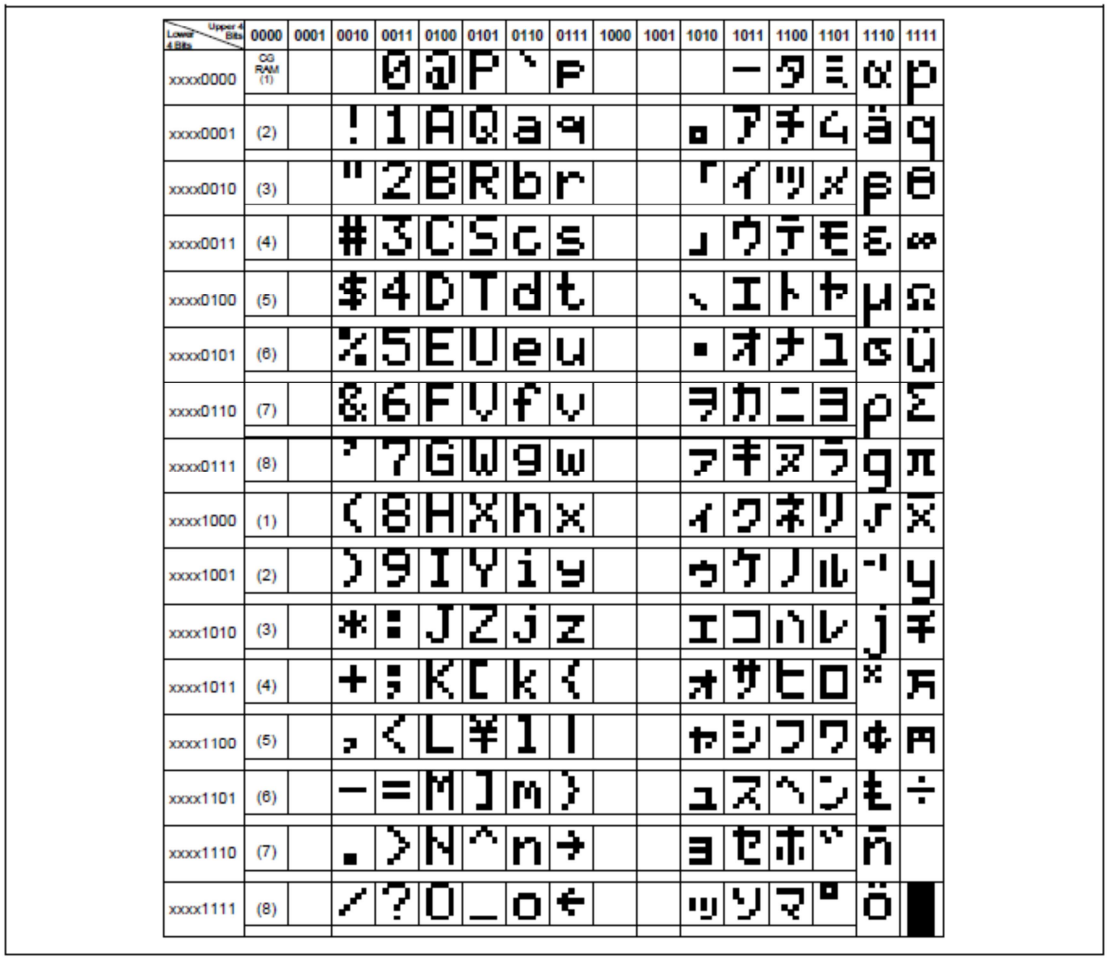






HD44780U

Table 4 **Correspondence between Character Codes and Character Patterns (ROM Code: A00)**



Note: The user can specify any pattern for character-generator RAM.

HD44780U

Initializing by Instruction

If the power supply conditions for correctly operating the internal reset circuit are not met, initialization by instructions becomes necessary.

Refer to Figures 25 and 26 for the procedures on 8-bit and 4-bit initializations, respectively.

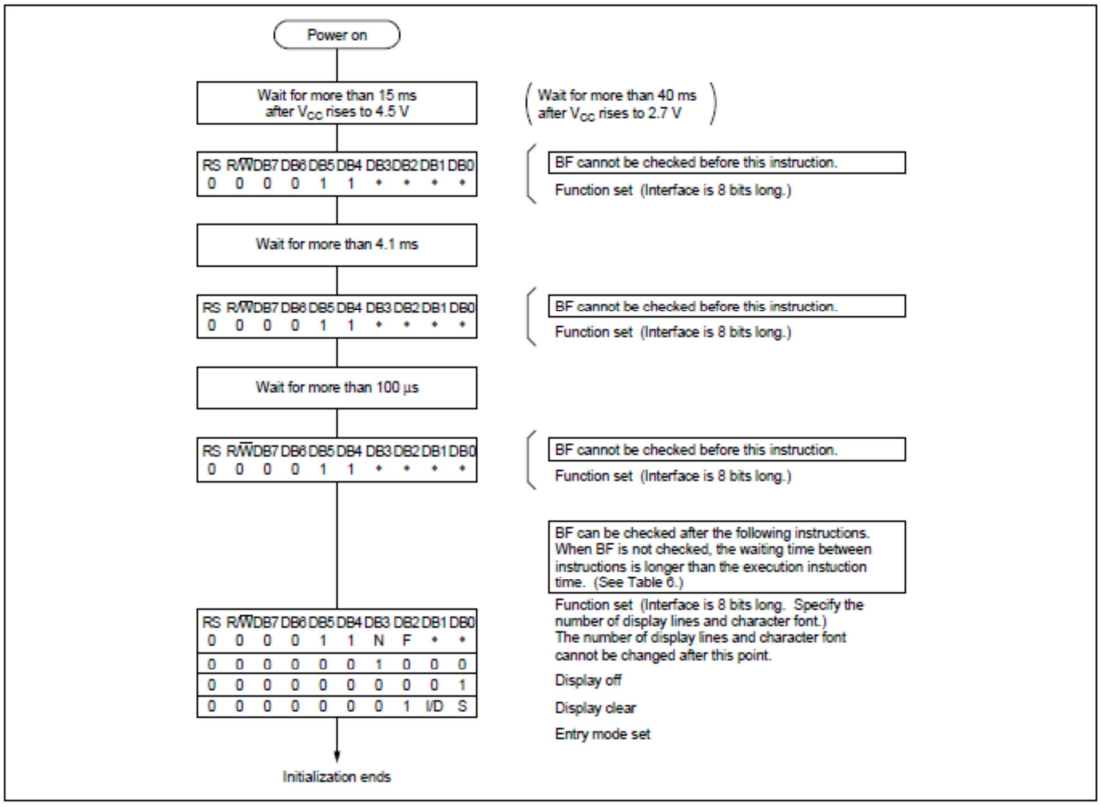


Figure 25 8-Bit Interface

HD44780U

Reset Function

Initializing by Internal Reset Circuit

An internal reset circuit automatically initializes the HD44780U when the power is turned on. The following instructions are executed during the initialization. The busy flag (BF) is kept in the busy state until the initialization ends (BF = 1). The busy state lasts for 10 ms after V_{CC} rises to 4.5 V.

1. Display clear
2. Function set:
   DL = 1; 8-bit interface data
   N = 0; 1-line display
   F = 0; 5 × 8 dot character font
3. Display on/off control:
   D = 0; Display off
   C = 0; Cursor off
   B = 0; Blinking off
4. Entry mode set:
   I/D = 1; Increment by 1
   S = 0; No shift

Note: If the electrical characteristics conditions listed under the table Power Supply Conditions Using Internal Reset Circuit are not met, the internal reset circuit will not operate normally and will fail to initialize the HD44780U. For such a case, initial-ization must be performed by the MPU as explained in the section, Initializing by Instruction.

Instructions

Outline

Only the instruction register (IR) and the data register (DR) of the HD44780U can be controlled by the MPU. Before starting the internal operation of the HD44780U, control information is temporarily stored into these registers to allow interfacing with various MPUs, which operate at different speeds, or various peripheral control devices. The internal operation of the HD44780U is determined by signals sent from the MPU. These signals, which include register selection signal (RS), read/

write signal (R/$\overline{W}$), and the data bus (DB0 to DB7), make up the HD44780U instructions (Table 6). There are four categories of instructions that:

- Designate HD44780U functions, such as display format, data length, etc.
- Set internal RAM addresses
- Perform data transfer with internal RAM
- Perform miscellaneous functions

Normally, instructions that perform data transfer with internal RAM are used the most. However, auto-incrementation by 1 (or auto-decrementation by 1) of internal HD44780U RAM addresses after each data write can lighten the program load of the MPU. Since the display shift instruction (Table 11) can perform concurrently with display data write, the user can minimize system development time with maximum programming efficiency.

When an instruction is being executed for internal operation, no instruction other than the busy flag/address read instruction can be executed.

Because the busy flag is set to 1 while an instruction is being executed, check it to make sure it is 0 before sending another instruction from the MPU.

Note: Be sure the HD44780U is not in the busy state (BF = 0) before sending an instruction from the MPU to the HD44780U. If an instruction is sent without checking the busy flag, the time between the first instruction and next instruction will take much longer than the instruction time itself. Refer to Table 6 for the list of each instruc-tion execution time.

Table 6 Instructions

	Code											Execution Time (max) (when f_{cp} or f_{osc} is 270 kHz)
Instruction	RS	R/$\overline{W}$	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description	
Clear display	0	0	0	0	0	0	0	0	0	1	Clears entire display and sets DDRAM address 0 in address counter.	
Return home	0	0	0	0	0	0	0	0	1	—	Sets DDRAM address 0 in address counter. Also returns display from being shifted to original position. DDRAM contents remain unchanged.	1.52 ms
Entry mode set	0	0	0	0	0	0	0	1	I/D	S	Sets cursor move direction and specifies display shift. These operations are performed during data write and read.	37 µs
Display on/off control	0	0	0	0	0	0	1	D	C	B	Sets entire display (D) on/off, cursor on/off (C), and blinking of cursor position character (B).	37 µs
Cursor or display shift	0	0	0	0	0	1	S/C	R/L	—	—	Moves cursor and shifts display without changing DDRAM contents.	37 µs
Function set	0	0	0	0	1	DL	N	F	—	—	Sets interface data length (DL), number of display lines (N), and character font (F).	37 µs
Set CGRAM address	0	0	0	1	ACG	ACG	ACG	ACG	ACG	ACG	Sets CGRAM address. CGRAM data is sent and received after this setting.	37 µs
Set DDRAM address	0	0	1	ADD	ADD	ADD	ADD	ADD	ADD	ADD	Sets DDRAM address. DDRAM data is sent and received after this setting.	37 µs
Read busy flag & address	0	1	BF	AC	AC	AC	AC	AC	AC	AC	Reads busy flag (BF) indicating internal operation is being performed and reads address counter contents.	0 µs

HD44780U

Table 6 Instructions (cont)

Instruction	RS	R/$\overline{W}$	Code DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description	Execution Time (max) (when f_{cp} or f_{OSC} is 270 kHz)
Write data to CG or DDRAM	1	0	Write data								Writes data into DDRAM or CGRAM.	37 µs t_{ADD} = 4 µs*
Read data from CG or DDRAM	1	1	Read data								Reads data from DDRAM or CGRAM.	37 µs t_{ADD} = 4 µs*
	I/D = 1: Increment I/D = 0: Decrement S = 1: Accompanies display shift S/C = 1: Display shift S/C = 0: Cursor move R/L = 1: Shift to the right R/L = 0: Shift to the left DL = 1: 8 bits, DL = 0: 4 bits N = 1: 2 lines, N = 0: 1 line F = 1: 5 × 10 dots, F = 0: 5 × 8 dots BF = 1: Internally operating BF = 0: Instructions acceptable										DDRAM: Display data RAM CGRAM: Character generator RAM ACG: CGRAM address ADD: DDRAM address (corresponds to cursor address) AC: Address counter used for both DD and CGRAM addresses	Execution time changes when frequency changes Example: When f_{cp} or f_{OSC} is 250 kHz, $37\ \mu s \times \frac{270}{250} = 40\ \mu s$

Note: — indicates no effect.

* After execution of the CGRAM/DDRAM data write or read instruction, the RAM address counter is incremented or decremented by 1. The RAM address counter is updated after the busy flag turns off. In Figure 10, t_{ADD} is the time elapsed after the busy flag turns off until the address counter is updated.

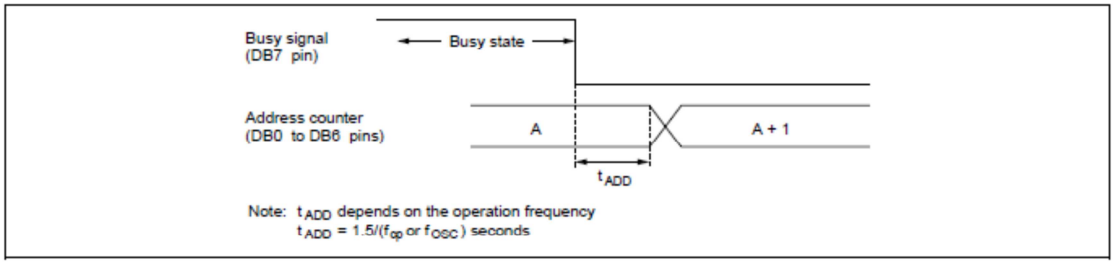


Figure 10 Address Counter Update

7.3. Servomotores

7.3.1. Hitec HS-311

User Reviews

Number of Reviews:	6
Average Rating:	2.2 / 5.0

Basic Information

Modulation:	Analog
Torque:	**4.8V:** 42.0 oz-in (3.02 kg-cm) **6.0V:** 48.6 oz-in (3.50 kg-cm)
Speed:	**4.8V:** 0.19 sec/60° **6.0V:** 0.15 sec/60°
Weight:	**1.52 oz (43.0 g)**
Dimensions:	**Length:** 1.57 in (39.9 mm) **Width:** 0.78 in (19.8 mm) **Height:** 1.43 in (36.3 mm)
Motor Type:	3-pole
Gear Type:	Plastic Bushing

Additional Specifications

Rotational Range:	0-9π
Pulse Cycle:	20 ms
Pulse Width:	900-2100 µs

7.3.2. GWS S125 1T 2BB

Description

The GWS Sail Winch servos are commonly used for radio control sail boats, but they are useful for any application where you want easy, inexpensive position control over more than 180°. We carry the 1/2 turn and 1 turn versions, which rotate 180° and 360°, respectively, when provided with the standard 1-2 ms servo pulse range output by many RC receivers.

These servos do not have physical end stops, so commanding them past their limits will cause them to rotate continuously until you change the commanded position back to something within its actual range. In our tests, this did not damage the servo, but we strongly recommend against it as there is no guarantee this will not damage the servo's feedback potentiometer.

The sail winch servos feature two ball bearings for improved performance, and they include a specialized servo horn that can be used as a winch. The 27 cm servo cable is terminated with either a female JR connector or a female Futaba connector. The Futaba connector has an extra polarizing tab that can be shaved off to make it compatible with male JR connectors.

Specifications

Dimensions

Size:	40.5 x 20 x 42 mm
Weight:	50 g

General specifications

Speed @ 6V:	0.21 sec/60°
Stall torque @ 6V:	7.6 kg·cm
Speed @ 4.8V:	0.26 sec/60°
Stall torque @ 4.8V:	6.6 kg·cm
Lead length:	270 mm

7.4. Bibliografía

http://arduino.cc
http://www.wiring.org.com
http://arduino.cc/en/booklet/homepage
http://eslibrary.stanford.edu/101/
http://www.youtube.com/watch?v=wuF0040paj9
www.ardumania.es
www.buenastareas.com
www.arduteka.com
www.opiron.com
www.tdrobotica.co/tutotiales/
www.pastebin.com/zPDKng6e
www.Earthshineelectronics.com
www.jayconsystems.com

Incluyendo materiales escritos por:
Massimo Banzi
Hernado Barragin
David Cuartielles
Tom Igoe
Todd Kurt
David Mellis