

## IE-0624 Laboratorio de Microcontroladores

Jorge Adán Mora Soto, B95222 Jafet David Gutiérrez Guevara, B73558  
[jorgeadan.mora@ucr.ac.cr](mailto:jorgeadan.mora@ucr.ac.cr) [jafet.gutierrez@ucr.ac.cr](mailto:jafet.gutierrez@ucr.ac.cr)

8 de diciembre de 2022

---

### Proyecto Final Sistema de Control de Iluminación Gestual

---

#### Resumen

En este proyecto se implementa un sistema que controle el estado de un conjunto de bombillos a través de gestos. La primera funcionalidad consiste en que el sistema permitirte elevar y disminuir la intensidad de la luminaria desde completamente apagado, hasta la intensidad máxima. Por otra parte, se implementa un algoritmo que opera por defecto, encendiendo y cambiando la intensidad de las luces en función de la hora del día. El flujo bajo el cual se modela dicho sistema se muestra en la figura 1. El primer elemento que se puede observar de izquierda a derecha es una cámara, la cual enviará su transmisión al segundo componente: una computadora. La PC ejecutará un programa de reconocimiento de imágenes, el cual procesará la transmisión de la cámara. Si el programa detecta alguno de los gestos definidos, se enviará una respuesta al tercer componente: un Arduino UNO. Con base en el estímulo recibido, este microcontrolador generará la combinación de salidas correspondiente para producir el resultado deseado, ya sea encender, apagar, atenuar o aumentar el brillo del bombillo. Finalmente, el cambio en las salidas se propagará por un circuito de control que manipula la tensión suministrada al bombillo, y consecuentemente variará su intensidad lumínosa. El proyecto entrega resultados satisfactorios en todo los puntos deseados.

Repositorio en GitHub: <https://github.com/Jams1001/IE0624/tree/main/Proyecto>

ID del último commit de interés: [ca7bff7494111287325cac4007a7dbf291258807](https://github.com/Jams1001/IE0624/commit/ca7bff7494111287325cac4007a7dbf291258807)

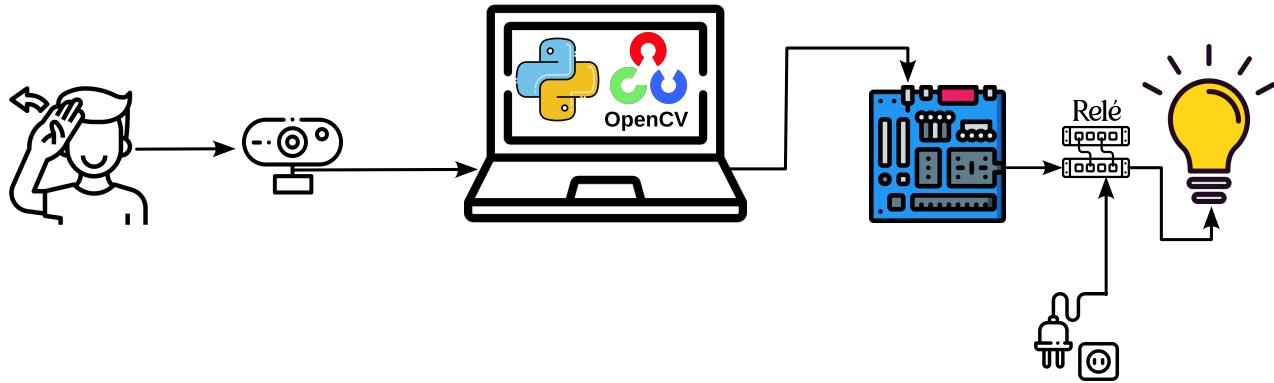


Figura 1: Diagrama ilustrativo (Autoría Propia).

## 1. Nota Teórica

### 1.1. Información general del MCU

En la siguiente práctica de laboratorio se utiliza el microcontrolador de ATmel ATmega328P como elemento central de la práctica. El mismo posee las siguientes características [1]:

- Microcontrolador AVR de 8 bits.
- Arquitectura RISC/Harvard.
- 4/8/16/64Kb Flash, 512b/1/2k bytes de SRAM y 256/512/1k bytes de EEPROM.
- 23 GPIOs.
- Timer/Counters de 8 y 16 bits e interrupciones.
- 8 canales PWM y comparador analógico y 6 canales 10-bit ADC.

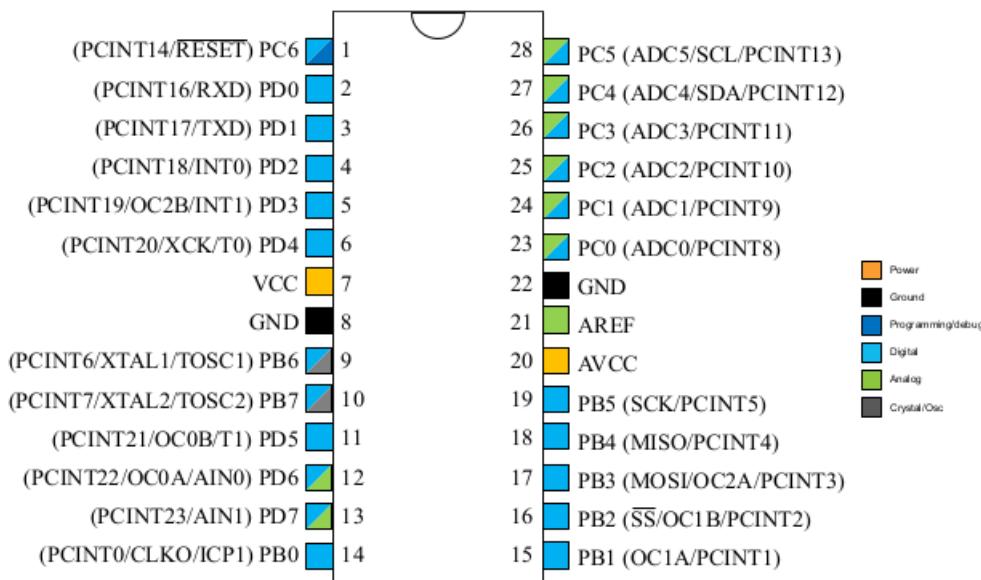


Figura 2: Diagrama de pines ATmega328P [2].

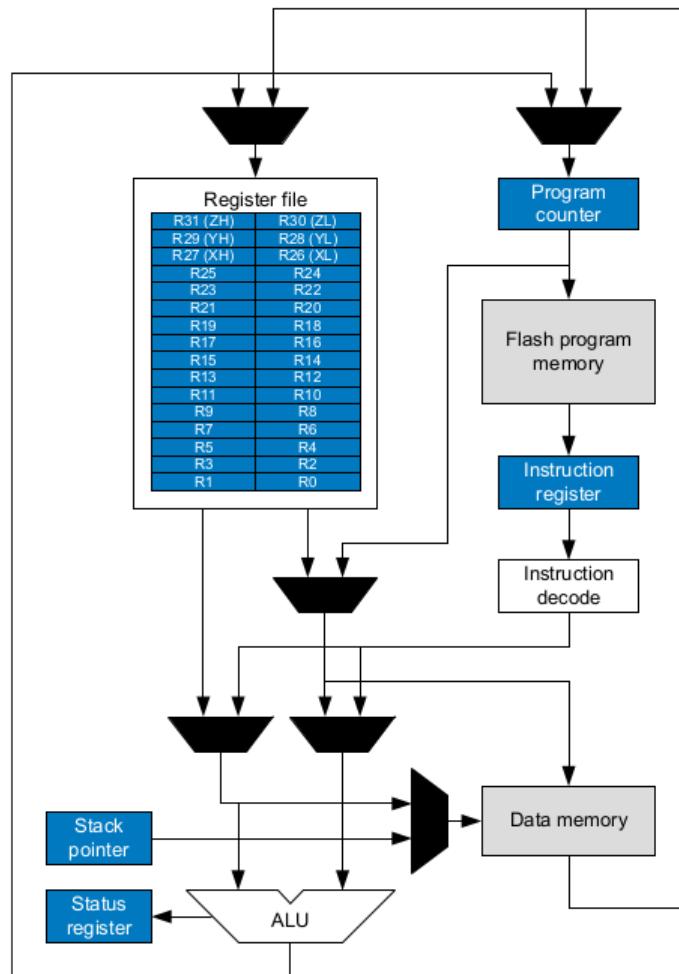


Figura 3: Diagrama de bloques ATmega328P [2].

### 1.1.1. Sobre la placa Arduino UNO Rev3

Arduino UNO es una excelente opción para iniciarse en la electrónica y la codificación, siendo la más utilizada de la familia de Arduinos.

Consta de 14 pines de entrada/salida digital (de los cuales 6 se pueden usar como salidas PWM), 6 entradas analógicas, un resonador cerámico de 16 MHz (CSTCE16M0V53-R0), una conexión USB, un conector de alimentación, un cabezal ICSP y un botón de reinicio. [3]

Adicionalmente Arduino proporciona la siguiente información

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) de los cuales 0.5 kB para bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13

Tabla 1: Características eléctricas y generales de la tarjeta Arduino UNO [3].

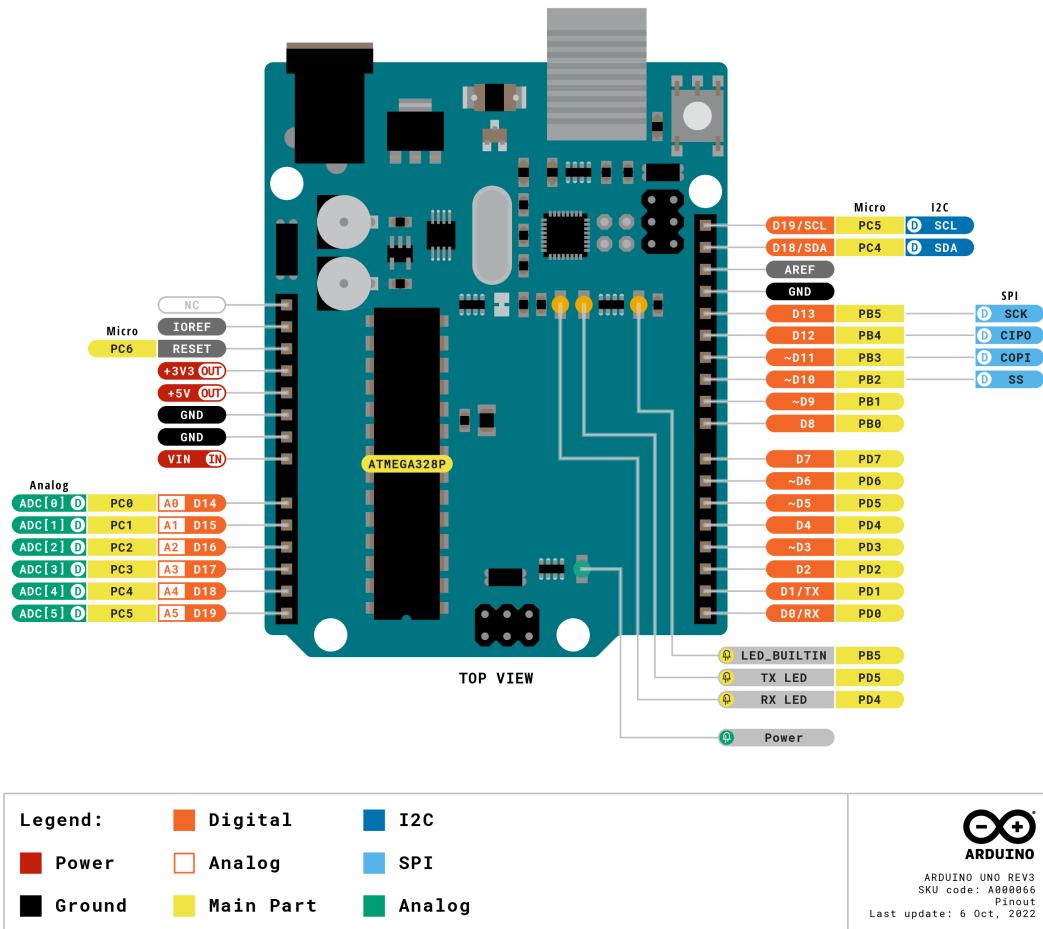


Figura 4: Diagrama pinout de Arduino UNO Rev3 [3].

## 1.2. Funciones de interés

- **pinMode()**: Configura el pin especificado para que se comporte como una entrada o una salida [3].
  - **analogRead()**: Lee el valor del pin analógico especificado. Las placas Arduino contienen un convertidor analógico a digital multicanal de 10 bits. Esto significa que mapeará los voltajes de entrada entre 0 y el voltaje de funcionamiento (5 V o 3,3 V) en valores enteros entre 0 y 1023. En un Arduino UNO, por ejemplo, esto produce una resolución entre lecturas de: 5 voltios / 1024 unidades o 0,0049 voltios (4,9 mV) por unidad [3].
  - **map()**: Vuelve a asignar un número de un rango a otro. Es decir, un valor de fromLow se asignaría a toLow, un valor de fromHigh a toHigh, valores intermedios a valores intermedios, etc. Reasigna la escala analógica a la escala digital en la resolución disponible [3].
  - **delay()**: Pausa el programa por la cantidad de tiempo (en milisegundos) especificado como parámetro. (Hay 1000 milisegundos en un segundo) [3].
  - **analogWrite()**: Escribe un valor analógico (onda PWM) en un pin. Se puede usar para encender un LED con diferentes brillos o impulsar un motor a varias velocidades. Después de una

llamada a `analogWrite()`, el pin generará una onda rectangular constante del ciclo de trabajo especificado hasta la próxima llamada a `analogWrite()` (o una llamada a `digitalRead()` o `digitalWrite()`) en el mismo pin. Recibe como parámetros `analogWrite(pin, value)`, en donde `pin` corresponde el pin de Arduino para escribir y `value` al ciclo de trabajo que va entre 0 (siempre apagado) y 255 (siempre encendido) [3].

### 1.3. Librerías de interés

- **mediapipe:** Esta librería, para el script de reconocimiento, consiste en una solución de detección de objetos 3D en tiempo real, de tal modo que detecta objetos en imágenes 2D y estima sus poses a través de un modelo, ya integrado, de machine learning.
- **LiquidCrystal:** Esta librería, para el sketch de arduino, permite controlar pantallas LCD que son compatibles con el controlador Hitachi HD44780. Hay muchos de ellos por ahí, y generalmente puede identificarlos por la interfaz de 16 pines.
- **TimerOne:** Esta librería, para el sketch de arduino, significa una solución para tener un contador de 16 bits que se puede configurar para realizar varias funciones diferentes, tales como fuentes de reloj. Esta librería puede usar un prescalar o incremento basado en la entrada de un pin de flanco ascendente/descendente. este prescalar divide el reloj de la CPU por: OFF, 1, 8, 64, 256, 1024.
- **RBDdimer:** Esta librería, para el sketch de arduino, se usa para trabajar con un dimmer. Brinda la capacidad de controlar una gran cantidad de dimmers; facilita la comunicación y el uso de variables en torno al dimmer. Funciona para los mcus de Mega, UNO, ESP8266, ESP32, Arduino M0, Arduino Zero, Arduino Due, STM32, entre otros.

### 1.4. Componentes complementarios

Dos elementos del circuito de control que valen la pena destacar son:

- Relé de 5V: Básicamente sirve como interruptor. Por un lado, se activa utilizando una señal de bajo voltaje, como por ejemplo una tensión continua de 5 V. Al activarse el relé, una bobina interna se encarga de abrir o cerrar el interruptor [4].
- Regulador de luz: Un regulador de intensidad puede conectarse a un Arduino para controlar la tensión de CA que recibe un dispositivo. El lado de bajo voltaje y el lado de alto voltaje están completamente aislados [5], lo cual asegura la protección del circuito de control y el microcontrolador.

Adicionalmente se presenta a continuación una tabla con todos los componentes electrónicos utilizados y el costo del proyecto:



a) Relé de 5V, 1 canal de gatillo de nivel alto o bajo opto aislado [6]

b) Regulador de luz Arduino de 1 canal [5]

Figura 5: Elementos principales del circuito de control

Componente	Cantidad	Precio (USD)
Relé	1	1.37
Regulador de luz	1	2.93
Pantalla LCD	1	5
Arduino UNO	1	28.5
<b>Total</b>		<b>37.8</b>

Tabla 2: Lista de cantidad y precios de los componentes (Autoría propia).

## 1.5. Diseño

En la figura 6 se presenta el esquemático completo del circuito implementado para este proyecto.

Con respecto a la pantalla LCD solo se trató de conectar los pines necesarios por la pantalla para cumplir con los requerimientos de lo que se desea imprimir; esto fue más seguir la conexión en base al firmware, que diseño de electrónica como tal. Esta pantalla, según las especificaciones técnicas, presenta un consumo de  $0.125mV$ .

Seguidamente, el consumo más significativo será por parte del bombillo; el cual, según las especificaciones técnicas tiene un consumo de  $60Watts$ . Tanto el dimmer, como el relé, tienen una caída de tensión no significativa; al igual que los cables por ser de poca longitud presentan una caída de tensión mínima con respecto al consumo principal. Por lo que el consumo total se podría reducir al bombillo. En base a este consumo se diseña.

El relé puede manejar corrientes de contacto de  $[10, 250]A$  AC o  $30VDC$ , con un voltaje de bobina de  $12V$  por canal. Posee voltaje de operación, en conjunto de  $[5, 12]V$ , y maneja  $[3, 5]V$  por cada canal.

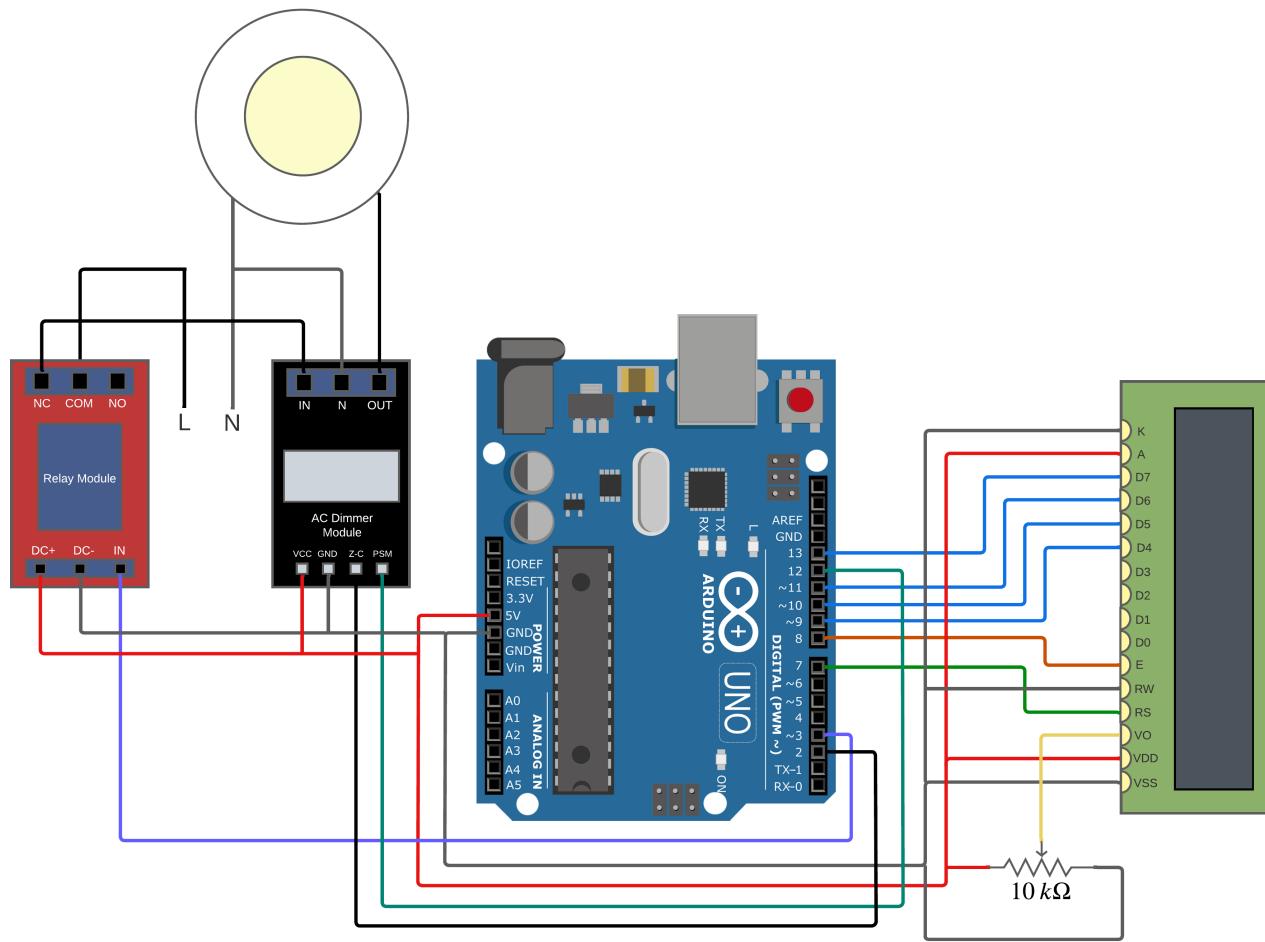


Figura 6: Esquemático a diseñado (Autoría propia).

## 1.6. Conceptos

### 1.6.1. IoT

Por sus siglas del inglés «*Internet of Things*», el Internet de las cosas, si de alguna forma se puede englobar en una definición, consiste en la interconexión de objetos por medio de una red, generalmente inalámbrica, donde existe una interacción entre los mismos sin la necesaria intervención del factor humano.

Podría tratarse de sensores, calzado, vestuario, o hasta un lápiz! Esta idea involucra comunicación entre dos máquinas o M2M (machine to machine) y por ende, machine learning. Al utilizar un microcontrolador que se conecte a internet y que cuando exista un estímulo de movimiento, como por ejemplo un sismo, envíe sus aceleraciones reales en forma de datos a internet para ser visualizadas por otro dispositivo en cualquier otra parte del mundo, se hace uso de IoT.

### 1.6.2. AI

La inteligencia artificial ( o por sus siglas en inglés «*AI*»), la capacidad de una computadora digital o un robot controlado por computadora para realizar tareas comúnmente asociadas con seres inteligentes. El término se aplica con frecuencia al proyecto de desarrollar sistemas dotados de los procesos

intelectuales característicos de los humanos, como la capacidad de razonar, descubrir significado, generalizar o aprender de experiencias pasadas [7].

### 1.6.3. Red Neuronal artificial

Consiste en una metodología de inteligencia artificial que enseña a las computadoras a procesar datos de una manera inspirada en el cerebro humano, con patrones o modelos previamente cargados de esos mismos eventos. Es un tipo de proceso de aprendizaje automático, llamado aprendizaje profundo, que utiliza nodos o neuronas interconectados en una estructura en capas que se asemeja al cerebro humano. Crea un sistema adaptativo que las computadoras usan para aprender de sus errores y mejorar continuamente. De esta forma las redes neuronales artificiales intentan resolver problemas complicados, como resumir documentos o reconocer rostros, reconocer movimientos, y todo esto en búsqueda de una mayor precisión para conquistar los límites de las capacidades humanas [8].

### 1.6.4. Machine learning

El aprendizaje automático, mejor conocido como *machine learning*, es una rama de la inteligencia artificial dirigida al desarrollo de herramientas que permiten que las máquinas aprendan sin estar explícitamente programadas para ello. Con dichas herramientas, se puede hacer que distintos dispositivos sean capaces identificar patrones entre un conjunto de datos, y a partir de ellos, realizar predicciones. El *machine learning* puede utilizarse para crear tecnologías inteligentes que faciliten la vida de sus usuarios, como Google Assistant, para dar un ejemplo [9].

## 2. Desarrollo / Análisis de Resultados

### 2.1. Análisis de SW

#### 2.1.1. Firmware

De forma general, se puede afirmar que el firmware desarrollado genera el comportamiento deseado cuando se carga en el Arduino UNO. La intensidad luminosa de la bombilla cambia correctamente en función de la posición de una mano ante la cámara de la computadora. Además, el lazo de ejecución modifica la intensidad luminosa a un valor predeterminado cuando el reloj configurado llega a cuatro horas específicas:

- **7 : 00** define un 40 % de intensidad lumínica.
- **8 : 30** define un 80 % de intensidad lumínica.
- **20 : 45** define un 45 % de intensidad lumínica.
- **22 : 15** define un 20 % de intensidad lumínica.

Por otro lado, tanto la hora como el porcentaje de luminosidad se despliegan en la pantalla LCD.

Sin embargo, durante el desarrollo del proyecto se encontraron otros resultados y efectos de interés. En primer lugar, aunque la hora de referencia que toma el proyecto, para que tenga sentido, debería ser la hora presente, para efectos demostrativos se configura una hora desde el script de reconocimiento. Esto para poder comprobar de manera dinámica el funcionamiento a del Arduino UNO a diferentes horas sin tener que esperar largas jornadas de tiempo para que el microcontrolador llegue a las horas de interés. Dicho esto, se escribió el firmware para que inicialice el reloj a partir del dato recibido con la primera lectura que realiza del puerto serial. Este primer dato corresponde a la hora que se definió desde el script de reconocimiento explicado posteriormente. Desde este punto el microcontrolador lleva la cuenta del reloj ininterrumpidamente, a menos de que se utilice el botón reset integrado en la tableta de desarrollo. En este último caso, el Arduino entra en un estado de inactividad, en espera de recibir nuevamente una hora desde el puerto serial. El funcionamiento de esta sección se comprobó a través de la pantalla LCD externa, en la cual se despliega el reloj. Sin embargo, en esta parte se detectó que en ocasiones aparentemente aleatorias, la cuenta desplegada en la pantalla permanece en una cuenta más tiempo del esperado, para después saltar a la cuenta correcta. Se determinó que dicho bug puede deberse a que el Arduino UNO ejecuta los ciclos de control a una frecuencia más alta de lo que la pantalla LCD puede procesar adecuadamente. Para solucionar este error se recomienda utilizar un módulo adaptador de LCD a I2C para poder controlar la pantalla LCD con solo dos pines del Arduino UNO. De esta manera se podría tener un mejor control de los paquetes de datos que se envían del Arduino a la pantalla.

Por otro lado, uno de los retos que se tuvo, a la hora debuggear el firmware, fue determinar el rango de valores a la entrada del módulo Dimmer AC que proporcionan su funcionamiento adecuado para una bombilla. Para integrar el módulo Dimmer AC al proyecto, se empleó la librería `RBDdimmer.h`, la cual provee una función llamada `setPower()` para configurar la intensidad luminosa que se quiere para la bombilla en un momento dado. Dicha función recibe números enteros en un rango entre 0 y 100. Dicho rango se estableció de esa manera para tener una referencia a nivel de porcentaje de la intensidad luminosa deseada. Sin embargo, los rangos de operación óptimos varían dependiendo de las características de la bombilla. Por esta razón, se tuvo que determinar dicho rango de manera

experimental. Los valores de entrada de la función `setPower()` y los porcentajes de intensidad luminosa estimados que generan se presentan en la siguiente tabla:

Entrada de <code>setPower()</code>	Porcentaje de intensidad luminosa estimado
20	10 %
27	20 %
35	30 %
37	40 %
40	50 %
43	60 %
50	70 %
56	80 %
67	90 %

Tabla 3: Entradas de `setPower()` y sus porcentajes de intensidad luminosa estimados

Como se muestra en la tabla 3, el rango de entradas adecuado para la bombilla utilizada es [20, 67]. Además, cabe mencionar que para proteger la resistencia de la bombilla incandescente se decidió no exceder el 90 % de intensidad luminosa.

### 2.1.2. Script de reconocimiento de imágenes de Python

Este script entregado en /IE0624/Proyecto/src/hand\_recognizer.py es el encargado de registrar en coordenadas de la mano de derecha a izquierda con una resolución de 0 a 100 puntos de longitud. Adicionalmente también se encarga de solicitar 3 parámetros para definir una hora de inicio. Idealmente, registrar la hora de inicio sería para que el control de luz en base a la hora del día, tendría sentido con la hora real; pero para efectos demostrativos se solicitan los parámetros de hora para poder probar diferentes horas a voluntad y comprobar la funcionalidad del proyecto.

En figura 7 se ilustra el funcionamiento del script.

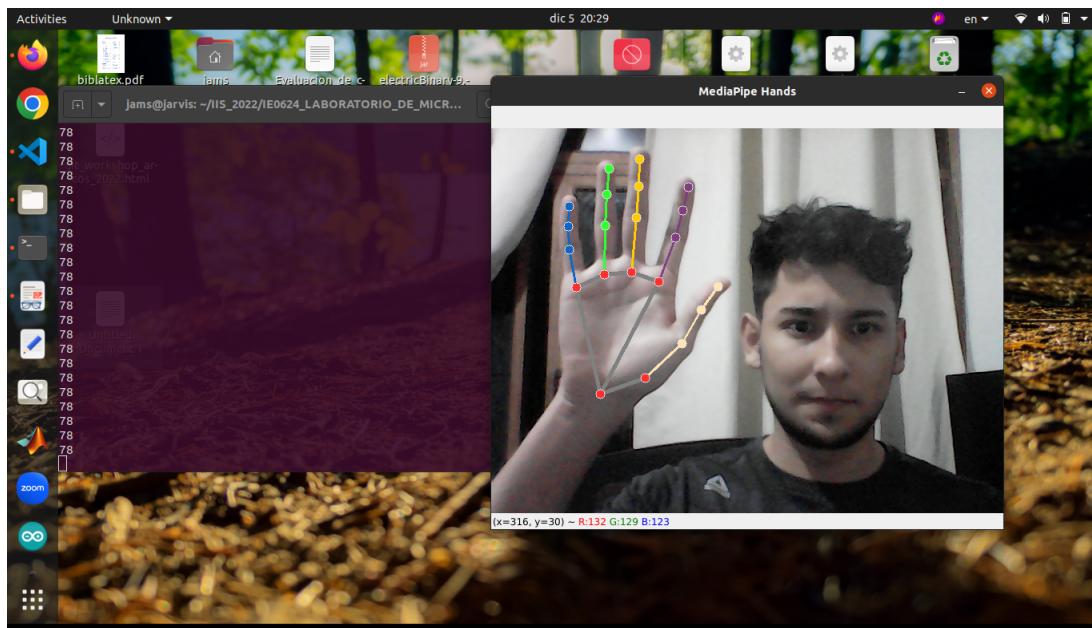


Figura 7: Funcionalidad del script `hand_recognizer.py` (Autoría propia).

El funcionamiento de este script es acertado y no hay mucho más por analizar, pues su comprobación es simple. Registra aproximadamente 20 muestras por segundo y es un programa sólido.

Consiste primeramente en la integración de las librerías mencionadas anteriormente. Seguidamente se registran los parámetros deseados de la hora con en el orden [hour, minute, second] para posteriormente llegar al grueso del programa de visión por computador.

La sección que muestrea registra únicamente las coordenadas horizontales de la posición de la mano. En el entregable estas coordenadas no se imprimen, esto se implementa únicamente para verificar su funcionamiento. Seguidamente se entra a la parte de interacción serial con el arduino. Es muy importante resaltar que las muestras capturadas se envian como **string** al arduino por el anterior análisis dado en la sección del correspondiente al firmware. La comunicación serial presentó la dificultad de ser demasiada información para procesar por el arduino, pues el muestreo era mayor, con mayor cantidad de decimales, y con más precisión; dado esto, y debido a que para efectos de percepción humana en una señal analógica basta con al menos unos 10 puntos de luminosidad para cumplir con el objetivo del proyecto, se parametrizó las muestras en 10 estados. Los puntos de «posición» que tienen la capacidad de salir de la computadora hacia el arduino son 10: [0, 10, 20, 30, 40, 50, 60, 70, 80, 90]

La función **arduino** que es la encargada de enviar una parametrización de las muestras por el puerto serial, recibe un valor, dígase que este valor se encuentra entre 30 y 40, seguidamente ese valor se parametriza en en 30, y esta es la «señal» que se envía a través del puerto serial. Esto logró resolver el problema de las muestras excesivas y permitió evitar que el programa fuera más lento o se detuviera **«crasheando»** la computadora.

Por último se crea la imagen que se ilustra en figura 7 para poder ver con facilidad los puntos específicos coloridos que registran la posición de la mano.

## 2.2. Análisis de HW

El circuito de prueba construido para este proyecto se subdivide en dos secciones importantes: el circuito de control y el de potencia. El primero de estos dos se puede observar en la figura 8.

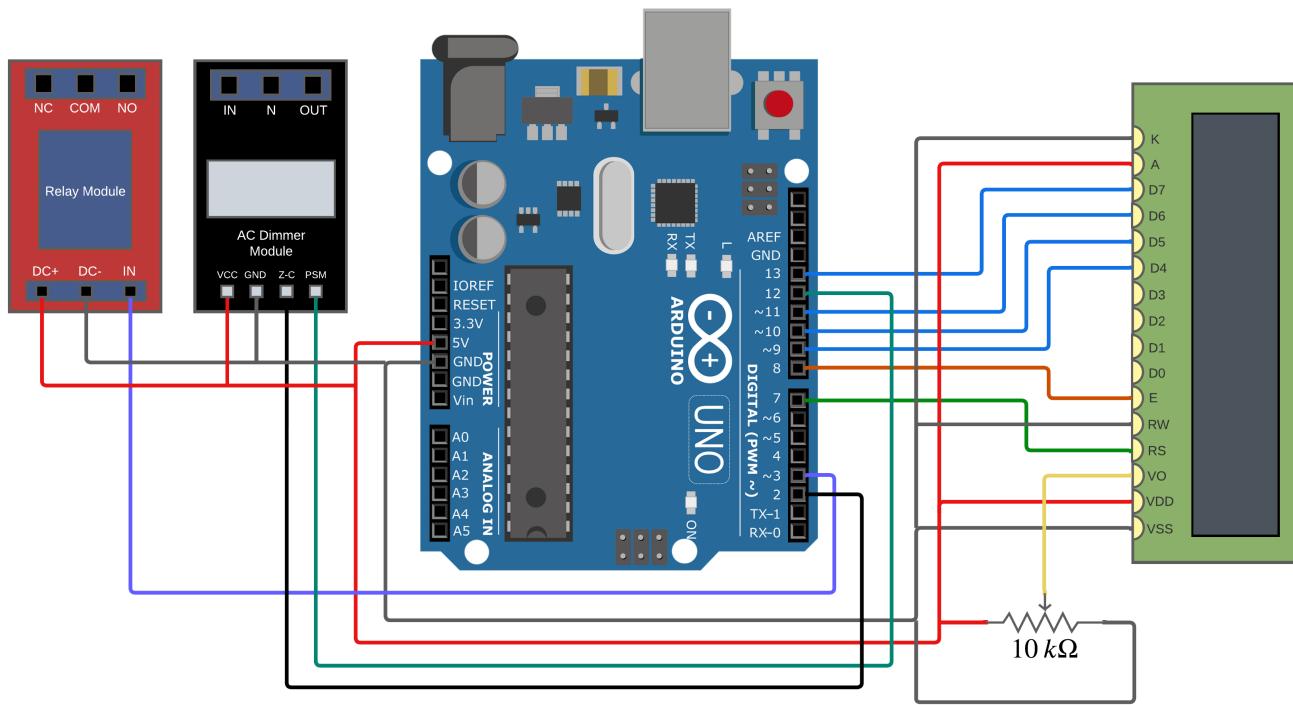


Figura 8: Diagrama del circuito de control

Por otra lado, la parte correspondiente al circuito de potencia se presenta en la figura 9.

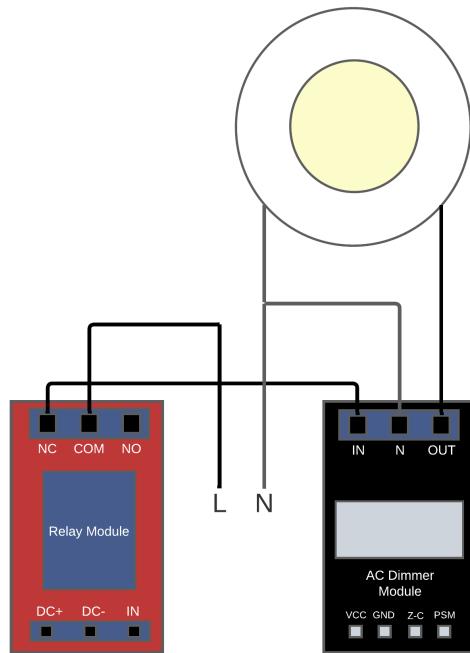


Figura 9: Diagrama del circuito de potencia

Con respecto al circuito de control, primeramente se sabe que la bobina del relé consume  $90\text{ mA}$  y necesita estar alimentada a  $5\text{ V}$  para operar correctamente. Al multiplicar estos últimos dos datos se puede determinar que la potencia consumida por el relé es de  $450\text{ mW}$ . Por otra parte, la pantalla LCD 16x2 presenta un consumo máximo de  $25\text{ mA}$  y requiere de  $5\text{ V}$  para funcionar, por lo tanto, este componente consume aproximadamente  $125\text{ mW}$ . En cuanto al módulo dimmer AC, se sabe que sus periféricos pueden operar adecuadamente a cualquier corriente menor a  $10\text{ mA}$ . Además, este componente también está conectado a la fuente de tensión de  $5\text{ V}$  del Arduino UNO. Tomando en cuenta estos datos, se tiene que el módulo dimmer AC puede llegar a consumir hasta  $50\text{ mW}$ . Al sumar todos los resultados anteriores, se tiene un consumo de potencia de  $625\text{ mA}$  por parte del circuito de control.

Por el lado del circuito de potencia, el único el único elemento que presenta consumo de potencia es la resistencia de la bombilla incandescente. El valor de dicha potencia corresponde a un máximo de  $100\text{ W}$ .

**A continuación se presenta un link para accesar y ver un video de su correcto funcionamiento:**

<https://drive.google.com/drive/folders/12DI6IFieF0CJ9Nj7iPxgw47PLW-znKQv?usp=sharing>

### 3. Conclusiones y Recomendaciones

- Se recomienda exportar el script que muestrea las coordenadas a un modelo procesable por el microcontrolador. Esto a través de TensorFlow con una red neuronal entrenada para el reconocimiento y predicción de gestos manuales; esto para generar un sistema lo más autónomo posible, integrando una cámara adicional y sin la necesidad de un computador. Aprovechando la capacidad de un mcu para producir un sistema embebido. Esta potencial implementación probablemente implique exportar el proyecto a un microcontrolador más potente como el Arduino Nano BLE sense.
- El objetivo del proyecto se cumplió exitosamente. El controlador de luz es correcto y la comunicación serial de igual forma. Lo que deja al Arduino UNO como una plataforma que le permite a los usuarios, no necesariamente únicamente del campo técnico-científico, una manera simple para crear objetos interactivos que pueden tener entradas de interruptores y sensores, para controlar salidas tanto físicas como digitales.
- Se recomienda adaptar otro entorno de desarrollo integrado para programar y compilar “Arduino”, pues aunque el utilizado está basado en C/C++, en primera instancia solo permite trabajarse a sí mismo bajo su propio IDE; el cual es bastante limitado. Por esto, a través de por ejemplo Visual estudio code, se pueden descargar extensiones e instalar librerías para utilizar un entorno más profesional.

## Bibliografía

- [1] M. M. V. Fallas, *Arduino UNO: PID, GPIO, ADC y comunicacione*, IE-0624 Laboratorio de Microcontroladores, ago. de 2022.
- [2] Atmel, “ATmega328/P DATASHEET COMPLETE”, Atmel, 1600 Technology Drive, San Jose, CA 95110 USA, Datasheet 06, jun. de 2016.
- [3] Arduino. “Arduino Uno Rev3”. Visitado en Diciembre 4 de 2022. (mayo de 2022), dirección: <https://store.arduino.cc/products/arduino-uno-rev3>.
- [4] D. Miguel, *Práctica 16. Encender un bombillo de 110V usando un relé*, <http://mecabot-ula.org/tutoriales/arduino/practica-16-encender-un-bombillo-de-110v-usando-un-rele/>, 2019.
- [5] RobotDyn, *Regulador de luz Arduino de 1 canal, módulo de atenuación Arduino, controlador de atenuación de luz CA, módulo de atenuación de CA para Arduino, STM32, ARM, AVR, lógica de 3.3 V/5 V, CA 50/60 Hz, 220 V/110 V*, [https://www.amazon.com/Regulador-Arduino-m%C3%B3dulo-atenuaci%C3%B3n-controlador/dp/B072K9P7KH/ref=sr\\_1\\_1\\_sspa?keywords=ac+dimmer+arduino&qid=1664350984&qu=eyJxc2MiOiIxLjU5IiwicXNhIjoiMC4wMCIsInFzcCI6IjAuMDAifQ%3D%3D&sprefix=rel%C3%A9%2B5vdc.%2Caps%2C292&sr=8-1&th=1](https://www.amazon.com/Regulador-Arduino-m%C3%B3dulo-atenuaci%C3%B3n-controlador/dp/B072K9P7KH/ref=sr_1_1_sspa?keywords=ac+dimmer+arduino&qid=1664350984&qu=eyJxc2MiOiIxLjU5IiwicXNhIjoiMC4wMCIsInFzcCI6IjAuMDAifQ%3D%3D&sprefix=rel%C3%A9%2B5vdc.%2Caps%2C292&sr=8-1&th=1), 2022.
- [6] AZKO, *Módulo de relé de placa de relé de 5 V, 1 canal de gatillo de nivel alto o bajo opto aislado*, [https://www.amazon.com/-/es/M%C3%B3dulo-placa-canal-gatillo-aislado/dp/B09G6H7JDT/ref=sr\\_1\\_1?\\_\\_mk\\_es\\_US=%C3%85M%C3%85%C5%BD%C3%95%C3%91&qid=3OCEGXEGN7AP&keywords=relay%2B5vdc.&qid=1664353286&qu=eyJxc2MiOiIxLjU5IiwicXNhIjoiMC4wMCIsInFzcCI6IjAuMDAifQ%3D%3D&sprefix=rel%C3%A9%2B5vdc.%2Caps%2C292&sr=8-1&th=1](https://www.amazon.com/-/es/M%C3%B3dulo-placa-canal-gatillo-aislado/dp/B09G6H7JDT/ref=sr_1_1?__mk_es_US=%C3%85M%C3%85%C5%BD%C3%95%C3%91&qid=3OCEGXEGN7AP&keywords=relay%2B5vdc.&qid=1664353286&qu=eyJxc2MiOiIxLjU5IiwicXNhIjoiMC4wMCIsInFzcCI6IjAuMDAifQ%3D%3D&sprefix=rel%C3%A9%2B5vdc.%2Caps%2C292&sr=8-1&th=1), 2022.
- [7] B. Copeland. “artificial intelligence”. Visitado en noviembre 29 de 2022. (nov. de 2011), dirección: <https://www.britannica.com/technology/artificial-intelligence>.
- [8] AWS. “What Is A Neural Network?” Visitado en noviembre 29 de 2022. (nov. de 2022), dirección: <https://aws.amazon.com/what-is/neural-network/>.
- [9] M. Natraj. “AI Magic Wand with TensorFlow Lite for Microcontrollers and Arduino”. Visitado en noviembre 23 de 2022. (mar. de 2022), dirección: <https://codelabs.developers.google.com/magicwand#0>.

## 4. Apéndices

23. SPI – Serial Peripheral Interface.....	215
23.1. Features.....	215
23.2. Overview.....	215
23.3. SS Pin Functionality.....	219
23.4. Data Modes.....	219
23.5. Register Description.....	220
24. USART - Universal Synchronous Asynchronous Receiver Transceiver.....	225
24.1. Features.....	225
24.2. Overview.....	225
24.3. Block Diagram.....	225
24.4. Clock Generation.....	226
24.5. Frame Formats.....	229
24.6. USART Initialization.....	230
24.7. Data Transmission – The USART Transmitter.....	231
24.8. Data Reception – The USART Receiver.....	233
24.9. Asynchronous Data Reception.....	237
24.10. Multi-Processor Communication Mode.....	239
24.11. Examples of Baud Rate Setting.....	240
24.12. Register Description.....	243
25. USARTSPI - USART in SPI Mode.....	254
25.1. Features.....	254
25.2. Overview.....	254
25.3. Clock Generation.....	254
25.4. SPI Data Modes and Timing.....	255
25.5. Frame Formats.....	255
25.6. Data Transfer.....	257
25.7. AVR USART MSPIM vs. AVR SPI.....	258
25.8. Register Description.....	259
26. TWI - 2-wire Serial Interface.....	260
26.1. Features.....	260
26.2. Two-Wire Serial Interface Bus Definition.....	260
26.3. Data Transfer and Frame Format.....	261
26.4. Multi-master Bus Systems, Arbitration and Synchronization.....	264
26.5. Overview of the TWI Module.....	266
26.6. Using the TWI.....	268
26.7. Transmission Modes.....	271
26.8. Multi-master Systems and Arbitration.....	289
26.9. Register Description.....	291
27. AC - Analog Comparator.....	299
27.1. Overview.....	299
27.2. Analog Comparator Multiplexed Input.....	299
27.3. Register Description.....	300
28. ADC - Analog to Digital Converter.....	305

28.1. Features.....	305
28.2. Overview.....	305
28.3. Starting a Conversion.....	307
28.4. Prescaling and Conversion Timing.....	308
28.5. Changing Channel or Reference Selection.....	310
28.6. ADC Noise Canceler.....	312
28.7. ADC Conversion Result.....	315
28.8. Temperature Measurement.....	316
28.9. Register Description.....	316
<b>29. DBG - debugWIRE On-chip Debug System.....</b>	<b>327</b>
29.1. Features.....	327
29.2. Overview.....	327
29.3. Physical Interface.....	327
29.4. Software Break Points.....	328
29.5. Limitations of debugWIRE.....	328
29.6. Register Description.....	328
<b>30. BTLDR - Boot Loader Support – Read-While-Write Self-Programming.....</b>	<b>330</b>
30.1. Features.....	330
30.2. Overview.....	330
30.3. Application and Boot Loader Flash Sections.....	330
30.4. Read-While-Write and No Read-While-Write Flash Sections.....	331
30.5. Boot Loader Lock Bits.....	333
30.6. Entering the Boot Loader Program.....	334
30.7. Addressing the Flash During Self-Programming.....	335
30.8. Self-Programming the Flash.....	336
30.9. Register Description.....	344
<b>31. MEMPROG- Memory Programming.....</b>	<b>347</b>
31.1. Program And Data Memory Lock Bits.....	347
31.2. Fuse Bits.....	348
31.3. Signature Bytes.....	350
31.4. Calibration Byte.....	351
31.5. Page Size.....	351
31.6. Parallel Programming Parameters, Pin Mapping, and Commands.....	351
31.7. Parallel Programming.....	353
31.8. Serial Downloading.....	360
<b>32. Electrical Characteristics.....</b>	<b>365</b>
32.1. Absolute Maximum Ratings.....	365
32.2. Common DC Characteristics.....	365
32.3. Speed Grades.....	368
32.4. Clock Characteristics.....	369
32.5. System and Reset Characteristics.....	370
32.6. SPI Timing Characteristics.....	371
32.7. Two-wire Serial Interface Characteristics.....	372
32.8. ADC Characteristics.....	374
32.9. Parallel Programming Characteristics.....	375

33. Typical Characteristics ( $T_A = -40^\circ\text{C}$ to $85^\circ\text{C}$ ).....	378
33.1. ATmega328 Typical Characteristics.....	378
34. Typical Characteristics ( $T_A = -40^\circ\text{C}$ to $105^\circ\text{C}$ ).....	403
34.1. ATmega328P Typical Characteristics.....	403
35. Register Summary.....	428
35.1. Note.....	430
36. Instruction Set Summary.....	432
37. Packaging Information.....	436
37.1. 32-pin 32A.....	436
37.2. 32-pin 32M1-A.....	437
37.3. 28-pin 28M1.....	438
37.4. 28-pin 28P3.....	439
38. Errata.....	440
38.1. Errata ATmega328/P.....	440
39. Datasheet Revision History.....	443
39.1. Rev. A – 06/2016.....	443

## 1. Description

The Atmel AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in a single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega328/P provides the following features: 32Kbytes of In-System Programmable Flash with Read-While-Write capabilities, 1Kbytes EEPROM, 2Kbytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), three flexible Timer/Counters with compare modes and PWM, 1 serial programmable USARTs , 1 byte-oriented 2-wire Serial Interface (I2C), a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages) , a programmable Watchdog Timer with internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main oscillator and the asynchronous timer continue to run.

Atmel offers the QTouch® library for embedding capacitive touch buttons, sliders and wheels functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS™) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.

The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega328/P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega328/P is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

## 2. Configuration Summary

Features	ATmega328/P
Pin Count	28/32
Flash (Bytes)	32K
SRAM (Bytes)	2K
EEPROM (Bytes)	1K
General Purpose I/O Lines	23
SPI	2
TWI (I <sup>2</sup> C)	1
USART	1
ADC	10-bit 15kSPS
ADC Channels	8
8-bit Timer/Counters	2
16-bit Timer/Counters	1

### 3. Ordering Information

#### 3.1. ATmega328

Speed [MHz] <sup>(3)</sup>	Power Supply [V]	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20	1.8 - 5.5	ATmega328-AU ATmega328-AUR <sup>(5)</sup> ATmega328-MMH <sup>(4)</sup> ATmega328-MMHR <sup>(4)(5)</sup> ATmega328-MU ATmega328-MUR <sup>(5)</sup> ATmega328-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

**Note:**

1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
3. Please refer to *Speed Grades* for Speed vs. V<sub>CC</sub>
4. Tape & Reel.
5. NiPdAu Lead Finish.

Package Type	
28M1	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
28P3	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
32A	32-lead, Thin (1.0mm) Plastic Quad Flat Package (TQFP)

### 3.2. ATmega328P

Speed [MHz] <sup>(3)</sup>	Power Supply [V]	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20	1.8 - 5.5	ATmega328P-AU ATmega328P-AUR <sup>(4)</sup> ATmega328P-MMH <sup>(4)</sup> ATmega328P-MMHR <sup>(4)(5)</sup> ATmega328P-MU ATmega328P-MUR <sup>(5)</sup> ATmega328P-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)
		ATmega328P-AN ATmega328P-ANR <sup>(5)</sup> ATmega328P-MN ATmega328P-MNR <sup>(5)</sup> ATmega328P-PN	32A 32A 32M1-A 32M1-A 28P3	Industrial (-40°C to 105°C)

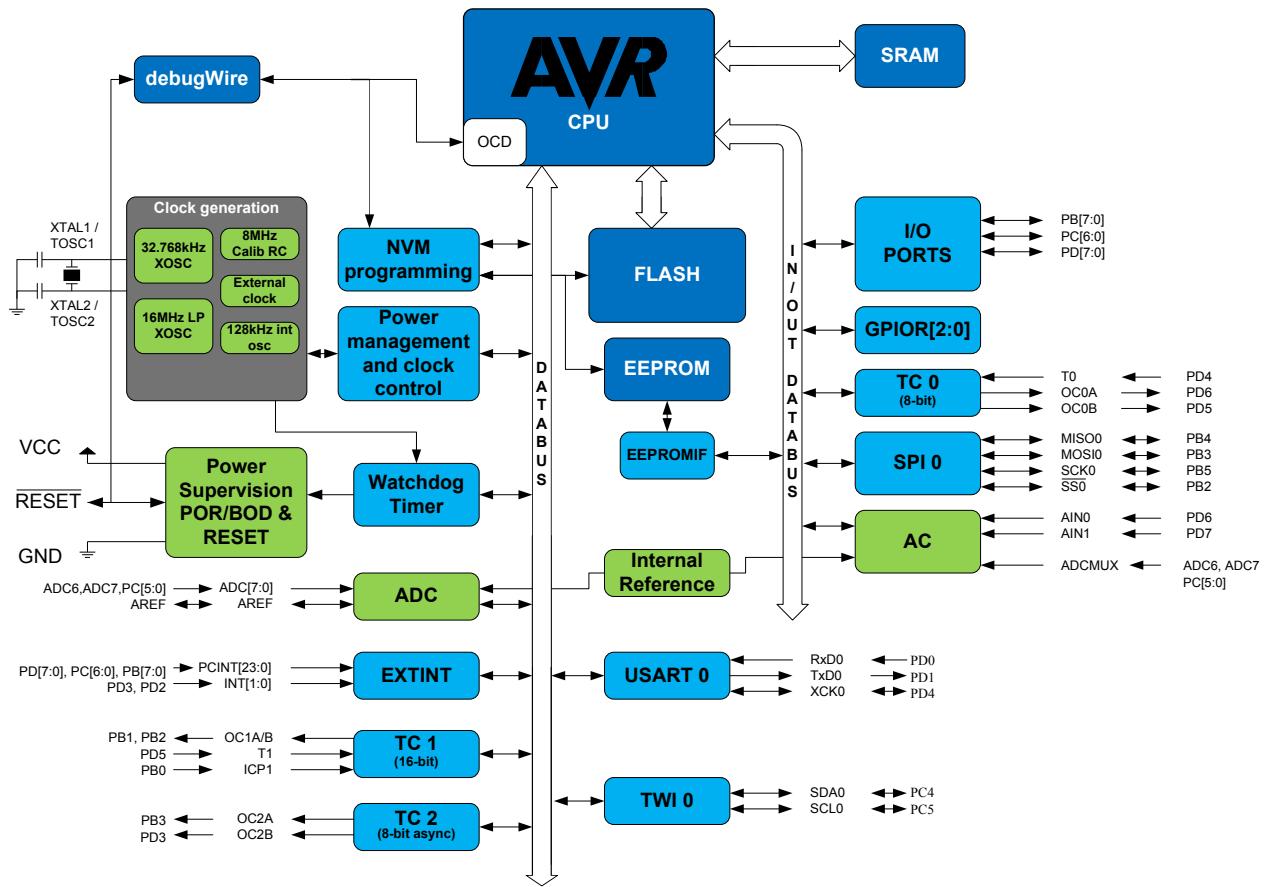
**Note:**

1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
3. Please refer to *Speed Grades* for Speed vs. V<sub>CC</sub>
4. Tape & Reel.
5. NiPdAu Lead Finish.

Package Type	
28M1	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
28P3	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
32A	32-lead, Thin (1.0mm) Plastic Quad Flat Package (TQFP)

## 4. Block Diagram

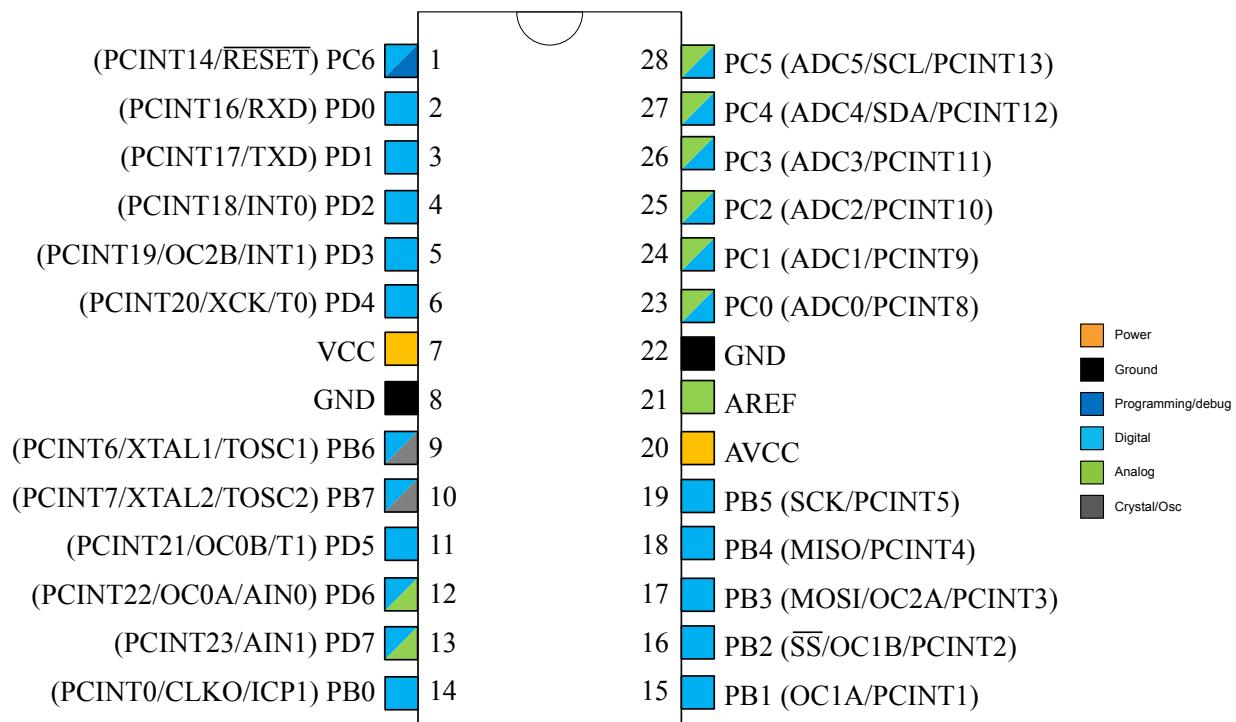
Figure 4-1. Block Diagram



## 5. Pin Configurations

### 5.1. Pin-out

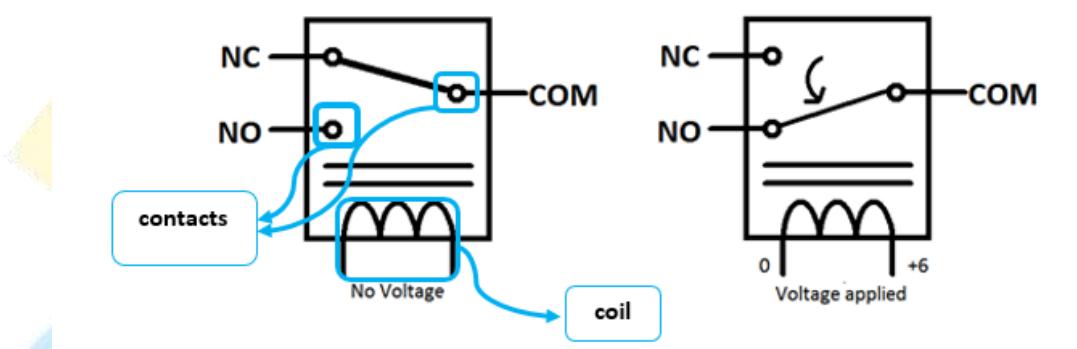
Figure 5-1. 28-pin PDIP



## RELAY MODULES

### RELAY WORKING IDEA

Relays consist of three pins normally open pin , normally closed pin, common pin and coil. When coil powered on magnetic field is generated the contacts connected to each other.

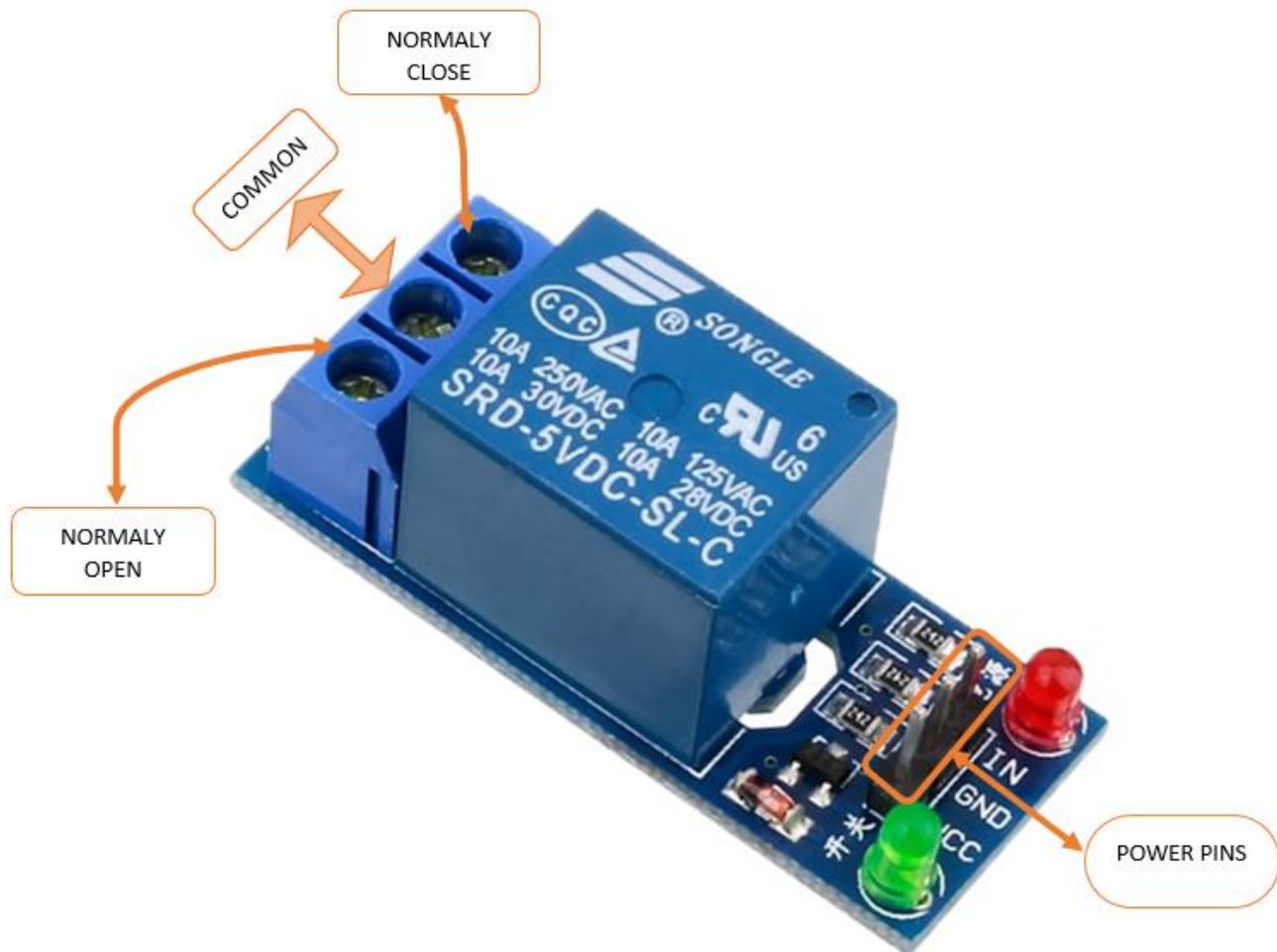


### Relay modules 1-channel features

- Contact current 10A and 250V AC or 30V DC.
- Each channel has indication LED.
- Coil voltage 12V per channel.
- Kit operating voltage 5-12 V
- Input signal 3-5 V for each channel.
- Three pins for normally open and closed for each channel.

### How to connect relay module with Arduino

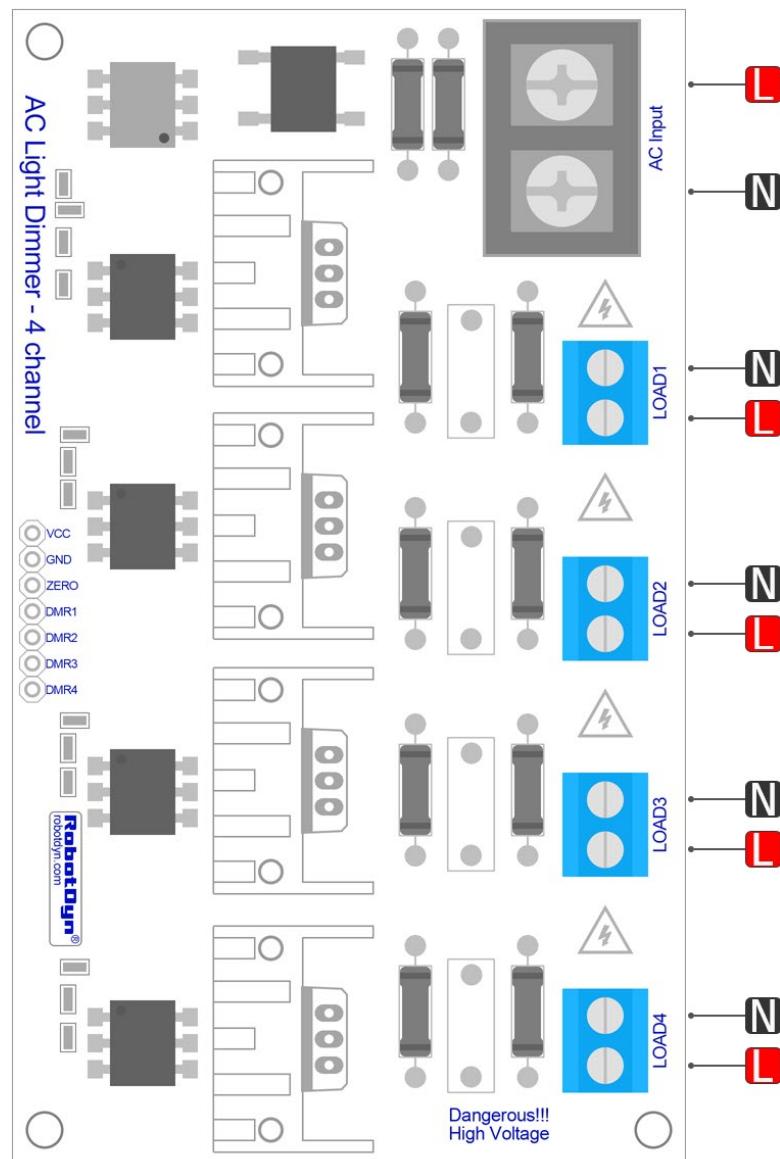
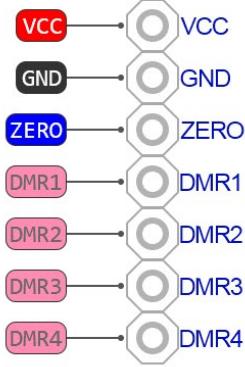
As shown in relay working idea it depends on magnetic field generated from the coil so there is power isolation between the coil and the switching pins so coils can be easily powered from Arduino by connecting VCC and GND pins from Arduino kit to the relay module kit after that we choose Arduino output pins depending on the number of relays needed in project designed and set these pins to output and make it out high (5 V) to control the coil that allow controlling of switching process.



**NOTE :** whatever was the relay channels number the pin configuration is the same for every channel except the power pins (VCC and GND) are for the board itself. The input signal (IN) pin for every relay.

**AC Light Dimmer Module,  
4 Channel, 3.3V/5V logic,  
AC 50/60hz, 220V/110V**

- Power
- Control
- GND
- PWM





## I2C Serial Interface 1602 LCD Module

This is I2C interface 16x2 LCD display module, a high-quality 2 line 16 character LCD module with on-board contrast control adjustment, backlight and I2C communication interface. For Arduino beginners, no more cumbersome and complex LCD driver circuit connection. The real significance advantages of this I2C Serial LCD module will simplify the circuit connection, save some I/O pins on Arduino board, simplified firmware development with widely available Arduino library.



**SKU:** [DSP-1182](#)

### **Brief Data:**

- Compatible with Arduino Board or other controller board with I2C bus.
- Display Type: Negative white on Blue backlight.
- I2C Address: 0x38-0x3F (0x3F default)
- Supply voltage: 5V
- Interface: I2C to 4bits LCD data and control lines.
- Contrast Adjustment: built-in Potentiometer.
- Backlight Control: Firmware or jumper wire.
- Board Size: 80x36 mm.

## Setting Up:

Hitachi's HD44780 based character LCD are very cheap and widely available, and is an essential part for any project that displays information. Using the LCD piggy-back board, desired data can be displayed on the LCD through the I2C bus. In principle, such backpacks are built around PCF8574 (from NXP) which is a general purpose bidirectional 8 bit I/O port expander that uses the I2C protocol. The PCF8574 is a silicon CMOS circuit provides general purpose remote I/O expansion (an 8-bit quasi-bidirectional) for most microcontroller families via the two-line bidirectional bus (I2C-bus). Note that most piggy-back modules are centered around PCF8574T (SO16 package of PCF8574 in DIP16 package) with a default slave address of 0x27. If your piggy-back board holds a PCF8574AT chip, then the default slave address will change to 0x3F. In short, if the piggy-back board is based on PCF8574T and the address connections (A0-A1-A2) are not bridged with solder it will have the slave address 0x27.



Address selection pads in the I2C-to-LCD piggy-back board.

**Table 5. PCF8574A address map**

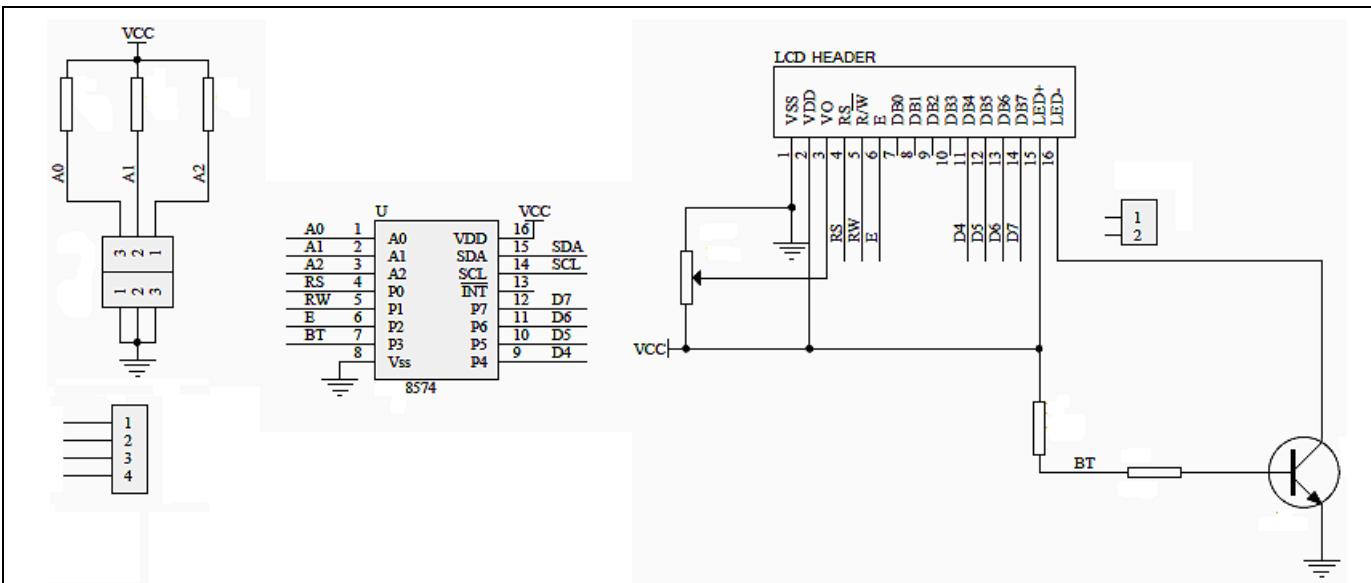
Pin connectivity			Address of PCF8574A									Address byte value		7-bit hexadecimal address without R/W
A2	A1	A0	A6	A5	A4	A3	A2	A1	A0	R/W	Write	Read		
V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	0	1	1	1	0	0	0	-	70h	71h	38h	
V <sub>SS</sub>	V <sub>SS</sub>	V <sub>DD</sub>	0	1	1	1	0	0	1	-	72h	73h	39h	
V <sub>SS</sub>	V <sub>DD</sub>	V <sub>SS</sub>	0	1	1	1	0	1	0	-	74h	75h	3Ah	
V <sub>SS</sub>	V <sub>DD</sub>	V <sub>DD</sub>	0	1	1	1	0	1	1	-	76h	77h	3Bh	
V <sub>DD</sub>	V <sub>SS</sub>	V <sub>SS</sub>	0	1	1	1	1	0	0	-	78h	79h	3Ch	
V <sub>DD</sub>	V <sub>SS</sub>	V <sub>DD</sub>	0	1	1	1	1	0	1	-	7Ah	7Bh	3Dh	
V <sub>DD</sub>	V <sub>DD</sub>	V <sub>SS</sub>	0	1	1	1	1	1	0	-	7Ch	7Dh	3Eh	
V <sub>DD</sub>	V <sub>DD</sub>	V <sub>DD</sub>	0	1	1	1	1	1	1	-	7Eh	7Fh	3Fh	

Address Setting of PCD8574A (extract from PCF8574A data specs).

**Note:** When the pad A0~A2 is open, the pin is pull up to VDD. When the pin is solder shorted, it is pull down to VSS.

The default setting of this module is A0~A2 all open, so is pull up to VDD. The address is 3Fh in this case.

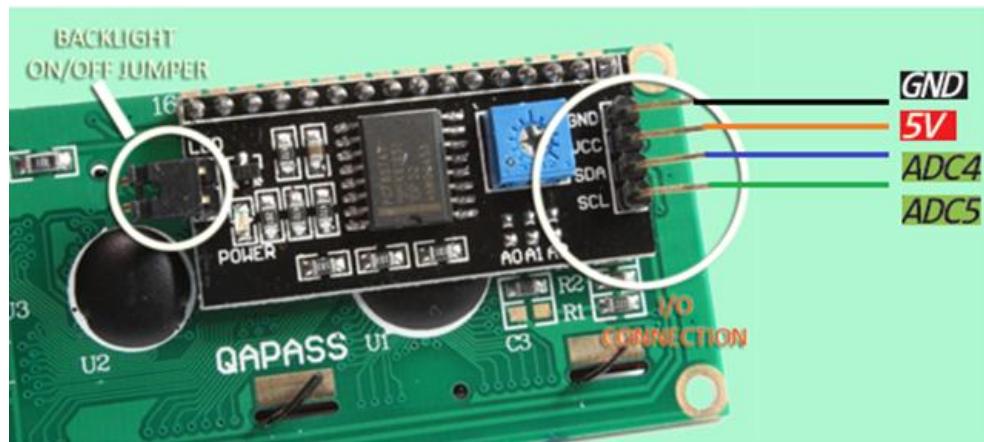
Reference circuit diagram of an Arduino-compatible LCD backpack is shown below. What follows next is information on how to use one of these inexpensive backpacks to interface with a microcontroller in ways it was exactly intended.



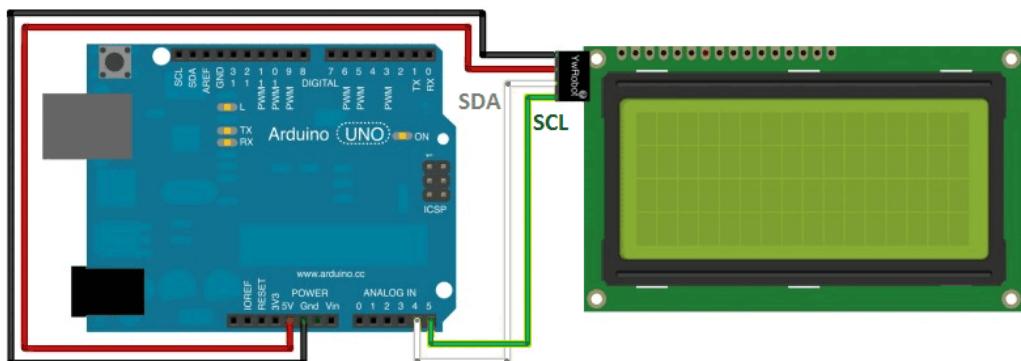
Reference circuit diagram of the I2C-to-LCD piggy-back board.

### I2C LCD Display.

At first you need to solder the I2C-to-LCD piggy-back board to the 16-pins LCD module. Ensure that the I2C-to-LCD piggy-back board pins are straight and fit in the LCD module, then solder in the first pin while keeping the I2C-to-LCD piggy-back board in the same plane with the LCD module. Once you have finished the soldering work, get four jumper wires and connect the LCD module to your Arduino as per the instruction given below.



LCD display to Arduino wiring.



## Arduino Setup

For this experiment it is necessary to download and install the “Arduino I2C LCD” library. First of all, rename the existing “LiquidCrystal” library folder in your Arduino libraries folder as a backup, and proceed to the rest of the process.

<https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads>

Next, copy-paste this example sketch Listing-1 for the experiment into the blank code window, verify, and then upload.

Arduino Sketch Listing-1:

```
/*=====
// Author      : Handson Technology
// Project     : I2C to LCD with Arduino Uno
// Description : LCD with I2C Interface.
// LiquidCrystal Library - I2C Serial to LCD
// Source-Code : I2C LCD.ino
=====*/
/*
-----( Import needed libraries )-----
#include <Wire.h> // Comes with Arduino IDE
// Get the LCD I2C Library here:
// https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads
// Move any other LCD libraries to another folder or delete them
// See Library "Docs" folder for possible commands etc.

#include <LiquidCrystal_I2C.h>
-----( Declare Constants )-----
// set the LCD address to 0x3F for PCF8574AT with A0,A1,A0 address line open, default
setting.
// Set the pins on the I2C chip used for LCD connections:
//           (addr, en,rw,rs,d4,d5,d6,d7,bl,blpol)
LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE); // Set the LCD I2C
address

-----( Declare Variables )-----

void setup() /*----( SETUP: RUNS ONCE )----*/
{
    Serial.begin(9600); // Used to type in characters

    lcd.begin(20,4); // initialize the lcd for 20 chars 4 lines, turn on
backlight

    // ----- Quick 3 blinks of backlight -----
    for(int i = 0; i< 3; i++)
    {
        lcd.backlight();
        delay(250);
        lcd.noBacklight();
        delay(250);
    }
    lcd.backlight(); // finish with backlight on

    //----- Write characters on the display -----
    // NOTE: Cursor Position: Lines and Characters start at 0
    lcd.setCursor(3,0); //Start at character 4 on line 0
    lcd.print("Hello, world!");
    delay(1000);
    lcd.setCursor(2,1);
    lcd.print("From Handsontec ");
}
```

```

delay(1000);
lcd.setCursor(0,2);
lcd.print("20 by 4 Line Display");
lcd.setCursor(0,3);
delay(2000);
lcd.print(" www.handsontec.com ");
delay(8000);
// Wait and then tell user they can start the Serial Monitor and type in characters
to
// Display. (Set Serial Monitor option to "No Line Ending")
lcd.setCursor(0,0); //Start at character 0 on line 0
lcd.print("Start Serial Monitor");
lcd.setCursor(0,1);
lcd.print("Type char to display");

}/*--(end setup )---*/



void loop() /*----( LOOP: RUNS CONSTANTLY )----*/
{
{
// when characters arrive over the serial port...
if (Serial.available()) {
    // wait a bit for the entire message to arrive
    delay(100);
    // clear the screen
    lcd.clear();
    // read all the available characters
    while (Serial.available() > 0) {
        // display each character to the LCD
        lcd.write(Serial.read());
    }
}
}

}/* --(end main loop )-- */

/* ( THE END ) */

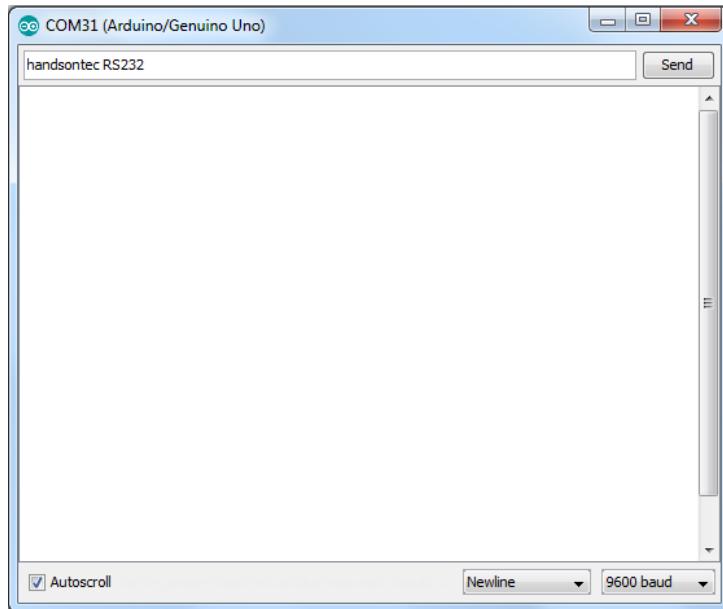
```

If you are 100% sure that everything is okay, but you don't see any characters on the display, try to adjust the contrast control pot of the backpack and set it a position where the characters are bright and the background does not have dirty boxes behind the characters. Following is a partial view of author's experiment with the above described code with 20x4 display module. Since the display used by the author is a very clear bright "black on yellow" type, it is very difficult to get a good catch due to polarization effects.



This sketch will also display character send from serial Monitor:

In Arduino IDE, go to “Tools” > “Serial Monitor”. Set the correct baud rate at 9600. Type the character on the top empty space and hit “SEND”.



The string of character will be displayed on the LCD module.



## **Resources:**

- [Handson Technology](#)
- [Complete Guide to Arduino LCD Interfacing \(PDF\)](#)



# Handsontec.com

*We have the parts for your ideas*

---

HandsOn Technology provides a multimedia and interactive platform for everyone interested in electronics. From beginner to diehard, from student to lecturer. Information, education, inspiration and entertainment. Analog and digital, practical and theoretical; software and hardware.



**HandsOn Technology support Open Source Hardware (OSHW)  
Development Platform.**

***Learn : Design : Share***

***[www.handsontec.com](http://www.handsontec.com)***



## The Face behind our product quality...

In a world of constant change and continuous technological development, a new or replacement product is never far away – and they all need to be tested.

Many vendors simply import and sell without checks and this cannot be the ultimate interests of anyone, particularly the customer. Every part sold on Handsotec is fully tested. So when buying from Handsontec products range, you can be confident you're getting outstanding quality and value.

We keep adding the new parts so that you can get rolling on your next project.



[Breakout Boards & Modules](#)



[Connectors](#)



[Electro-Mechanical Parts](#)



[Engineering Material](#)



[Mechanical Hardware](#)



[Electronics Components](#)



[Power Supply](#)



[Arduino Board & Shield](#)



[Tools & Accessory](#)