Universidad de Costa Rica
Escuela de Ingeniería Eléctrica
IE0624 - Laboratorio de Microcontroladores

Proyecto de Microcontroladores

Sebastián Vargas Quesada, C18295

27 de noviembre del 2024, Ciclo II

1. Resumen

En el presente proyecto se desarrolló un sistema de seguridad capaz de detectar movimiento y notificar al usuario tanto de manera sonora, mediante un buzzer, como visual, obteniendo una fotografía del evento captado. Para ello, se utilizó el microcontrolador Arduino UNO junto con componentes electrónicos adicionales, como un sensor ultrasónico, una cámara, un buzzer, un keypad y la plataforma de IoT ThingsBoard.

Se emplearon los puertos GPIO del Arduino para gestionar las entradas y salidas, la comunicación serial para el intercambio de datos con un script en Python, y la memoria EEPROM para almacenar de forma segura el código necesario para activar el sistema. La lógica del programa fue desarrollada en lenguaje C utilizando el entorno de desarrollo Arduino IDE, mientras que el script en Python permitió la comunicación con ThingsBoard.

Uno de los principales desafíos durante el desarrollo fue la implementación del switch en el dashboard de ThingsBoard, ya que este enviaba su estado utilizando RPC (Remote Procedure Call), una tecnología con la que inicialmente no se estaba familiarizado. No obstante, se logró crear un método en el script de Python capaz de interpretar el valor del switch, permitiendo así controlar el sistema de forma efectiva y satisfactoria. El repositorio del proyecto se puede consultar en el siguiente enlace: https://github.com/sebasvq106/microcontroladores y el video demostrativo lo puede encontrar en el siguiente link: https://drive.google.com/file/d/1c4P0ss7VCj5axJkX7as_-Z3B_V-IPIxU/view?usp=drive_link

2. Objetivos

2.1. Objetivo General

- Diseñar e implementar un sistema de seguridad utilizando un Arduino y un sensor para la detección de movimiento (Ultrasónico) que, al detectar un movimiento dentro de un rango, alertará al usuario mediante una alarma sonora, enviará notificaciones, tomará una fotografía del evento y permitirá la activación y desactivación tanto remota como presencial del sistema.

2.2. Objetivos Específicos

- Configurar e integrar el sensor tipo ultrasónico capaz de detectar un movimiento dentro de un rango de distancia.
- Programar el Arduino para que, al recibir la señal del sensor ultrasónico, active una alarma sonora y tome una fotografía del evento.
- Implementar el Internet de las cosas (IoT) para crear una comunicación con microcontrolador y el usuario utilizando la plataforma Thingsboard.

3. Alcances

El alcance de este proyecto incluye el diseño, desarrollo, simulación e implementación de un sistema de seguridad basado en un microcontrolador Arduino. El sistema será capaz de detectar movimiento dentro de un rango a través de un sensor ultrasónico, generar una alarma sonora utilizando un buzzer, y capturar imágenes mediante una cámara tras la activación del sensor. Además, se integrará tecnología IoT para permitir el control remoto del sistema, incluyendo la activación, desactivación, recepción de notificaciones y la foto capturada. También se incorporará un keypad como método alternativo para la activación y desactivación de manera presencial. Antes de la construcción del sistema se realizarán simulaciones en herramientas como Simulide y Tinkercad, mientras que la plataforma Thingsboard será utilizada para la integración de capacidades IoT, garantizando una comunicación eficiente entre el usuario y el sistema. Este proyecto busca proporcionar una solución de seguridad versátil, efectiva y accesible, adaptada a las necesidades del usuario.

4. Justificación

Según un artículo de CRHoy [?], en 2023 se reportaron aproximadamente 5600 robos a viviendas, una cifra alarmante que resalta la creciente inseguridad en nuestras comunidades. La implementación de un sistema de seguridad eficaz podría jugar un papel crucial en la reducción de estos incidentes, ofreciendo a los propietarios una mayor tranquilidad. La capacidad de activar y desactivar el sistema de manera remota no solo aporta una flexibilidad significativa, sino que también brinda comodidad a los usuarios. Esto les permite gestionar la seguridad de sus hogares desde cualquier lugar, ya sea a través de un teléfono inteligente o una computadora, adaptándose a sus estilos de vida y necesidades diarias. Además, ofrecer una solución de seguridad accesible y económica es fundamental para que un mayor número de personas pueda proteger sus hogares. Al eliminar barreras económicas, se garantiza que la seguridad no sea un privilegio exclusivo, sino un derecho al que todos puedan acceder, fomentando así un entorno más seguro y cohesionado en la comunidad.

5. Nota Teorica

5.1. Información General del Microcontrolador

Para la realización del proyecto se utilizará el Arduino UNO, un microcontrolador de 8 bits basado en la plataforma ATmega328P, fabricado por Microchip Technology. Este microcontrolador es conocido por su facilidad de uso y su amplia comunidad de soporte, lo que lo convierte en una opción popular para proyectos de electrónica y programación.

En este proyecto, se enfocará en el uso de los pines de entrada/salida (GPIO) y las comunicaciones seriales que ofrece el Arduino UNO. Este microcontrolador cuenta con 14 pines de GPIO, que se pueden configurar para funcionar como entradas digitales o salidas, así como 6 pines de ADC, que permiten la lectura de señales analógicas. Además, el Arduino UNO facilita la comunicación serial, lo que permite la transmisión de datos entre el microcontrolador y la computadora.

5.2. Diagrama de Pines

A continuación se muestra el diagrama de pines del Arduino UNO:

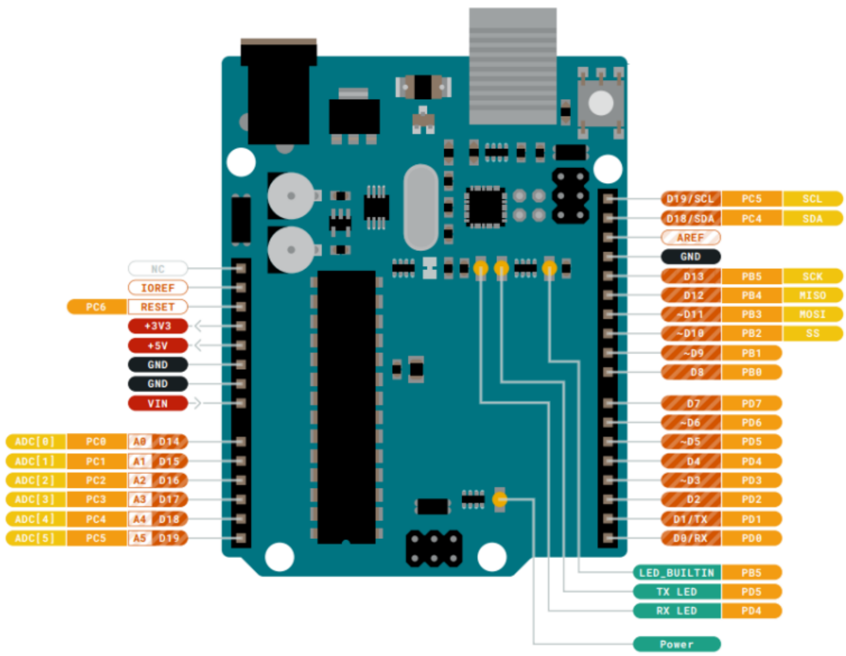


Figura 1: Diagrama de Pines del Arduino UNO [1]

5.3. Periféricos utilizados

Como se menciono anteriormente ara este laboratorio se utilizarán los diferentes pines GPIO de entrada y salida que posee el microcontrolador, la conexión serial y la memoria EEPROM.

5.3.1. GPIO

Los pines GPIO del Arduino pueden configurarse como entradas o salidas, dependiendo de los requerimientos específicos del proyecto. En el caso del sistema de seguridad, algunos pines se configurarán como entradas, como los utilizados para el keypad y el sensor ultrasónico, mientras que otros funcionarán como salidas, como el pin positivo del buzzer. Por otro lado, la cámara se conectará a los pines RX y TX del Arduino para establecer una comunicación serial, permitiendo la transmisión de datos entre los dos componetes.

5.3.2. Comunicación Serial

La comunicación serial, establecida a través de UART (Transmisor-Receptor Asíncrono Universal), es esencial para la interacción entre el microcontrolador y otros dispositivos. En este proyecto, desempeña un papel crucial al permitir la transferencia bidireccional de datos. Por un lado, facilita el envío de información desde el switch de ThingsBoard al Arduino para controlar el estado del sistema. Por otro lado, permite la obtención de imágenes capturadas por la cámara tras la detección de movimiento, las cuales se transmiten posteriormente a ThingsBoard.

5.3.3. Memoria EEPROM

La memoria EEPROM es suamente importante para el almacenamiento permanente de datos en el microcontrolador. En este proyecto, se utiliza para guardar de manera segura el código correcto necesario para activar y desactivar el sistema mediante el keypad. Al almacenar el código en la EEPROM, en lugar de escribirlo directamente en el código fuente del programa, se añade una medida adicional de seguridad, ya que evita que el código sea accesible o visible en caso de que el software sea inspeccionado. Esto asegura que el código permanezca protegido y que el sistema mantenga un nivel elevado de seguridad y confiabilidad.

5.4. Componentes Electrónicos Complementarios

Para la realización de este proyecto de utilizaron varios componentes electrónicos complementarios, a continuación se detalla cada uno:

5.4.1. HC-SR04

El sensor ultrasónico HC-SR04 es ampliamente utilizado para la detección de obstáculos y medición de distancias. Funciona emitiendo ondas ultrasónicas a través de su pin de transmisión (Trig) y midiendo el tiempo que tarda el eco en regresar al pin de recepción (Echo) [2]. En este proyecto, el HC-SR04 se configura como una entrada digital al microcontrolador, permitiendo medir la distancia de un objeto y detectar movimiento si este supera un rango establecido.

Para conectarlo al Arduino, se debe alimentar con 5V a través del pin VCC, conectar su pin GND a tierra, asignar el pin Trig a un pin configurado como OUTPUT en el Arduino, y conectar el pin Echo a un pin configurado como INPUT. Esta configuración asegura optener una lectura la cual se puede procesar en el microcontrolador.

5.4.2. VC0706

La cámara Adafruit VC0706 se conecta al microcontrolador Arduino mediante comunicación serial UART, por los puertos TX y RX, lo que facilita la transferencia de imágenes capturadas tras la detección de movimiento. Para proteger el pin RX de la cámara y garantizar una operación segura, se implementó un divisor de voltaje utilizando dos resistencias, adaptando los niveles de voltaje del Arduino a los requeridos por la cámara. Además, por cuestiones de velocidad y eficiencia, se configurará el tamaño de la imagen en 160x120, optimizando tanto la captura como la transferencia de datos [3].

5.4.3. Buzzer

El buzzer es un componente utilizado para generar alertas sonoras en diversos sistemas electrónicos. En este proyecto, se usará para la notificación sonora cuando se detecta movimiento.

Para conectarlo al Arduino, el buzzer debe alimentarse mediante su pin positivo (VCC), que se conecta a un pin configurado como OUTPUT en el microcontrolador, mientras que su pin negativo (GND) se conecta a tierra. Esta configuración permite al Arduino activar o desactivar el buzzer de forma programada, generando la señal sonora solo cuando sea necesario [4].

5.4.4. Keypad

El keypad es fundamental para controlar el sistema de manera presencial, ya que el usuario, al tener que ingresar un código, añade una capa de seguridad adicional al sistema, evitando activaciones o desactivaciones no autorizadas. Esta funcionalidad asegura que solo las personas que conocen el código puedan interactuar con el sistema, reforzando su confiabilidad y protección.

Para conectarlo al Arduino, el keypad requiere la asignación de pines tanto para sus filas como para sus columnas. Estos pines se configuran como entradas y salidas digitales en el microcontrolador, permitiendo detectar las combinaciones de teclas presionadas y validar el código ingresado contra los parámetros definidos en el sistema [5].

5.4.5. Plataforma Thingsboard

Para el apartado de IoT, se utilizará la plataforma ThingsBoard, proporcionada por la Escuela de Ingeniería Eléctrica de la Universidad de Costa Rica. Esta plataforma destaca por su interfaz sencilla y amigable, que facilita la configuración de dispositivos y la creación de dashboards personalizados. Además, incluye una amplia variedad de widgets que permiten desplegar la información de forma clara y efectiva.

Para establecer la comunicación con esta plataforma, se desarrolló un script en Python que procesa los datos y utiliza el protocolo MQTT (Message Queuing Telemetry Transport) para garantizar una transmisión eficiente y ligera de mensajes en tiempo real. Es importante mencionar que, para la transmisión de contenido HTML, como la imagen capturada por la cámara, se recurrió al protocolo HTTP, complementando así la funcionalidad de la plataforma. Esta combinación asegura una integración fluida entre el Arduino y ThingsBoard, habilitando un monitoreo y control eficiente del sistema en tiempo real.

5.4.6. Presupuesto

Otro apartado importante contar con un presupuesto para realizar el proyecto, por lo que en la siguiente tabla se las los precios aproximados de los componentes a utilizar:

Componentes	**Precio aproximado**
Arduino UNO	$28 US
HC-SR04	$4 US
VC0706	$70 US
Buzzer	$1 US
Keypad	$5 US

6. Desarrollo

6.1. Simulasión

El primer paso en el desarrollo del proyecto consistió en crear una simulación en la plataforma Tinkercad para verificar la viabilidad del sistema y definir el flujo de trabajo que seguirá. Esta etapa inicial permitió probar y ajustar las conexiones necesarias entre los componentes, asegurando que el diseño fuese funcional antes de proceder con la implementación física. En Tinkercad se recrearon todas las conexiones requeridas, como se ilustra en la siguiente imagen:

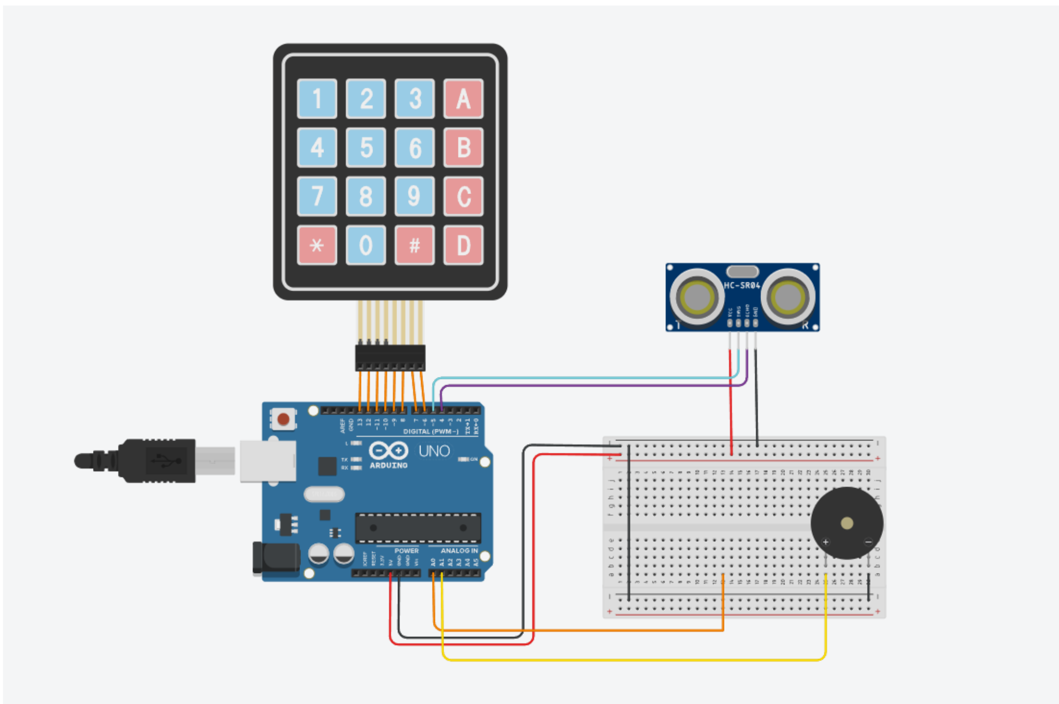


Figura 2: Circuito a simular

Debido a la disponibilidad limitada de componentes en Tinkercad, solo fue posible simular el uso del keypad, el sensor ultrasónico y el buzzer. Componentes clave como la cámara VC0706 y el apartado de IoT no están disponibles en esta plataforma, por lo que no pudieron ser incluidos en la simulación.

En cuanto a los resultados obtenidos de esta simulación, se destaca el correcto funcionamiento del sensor ultrasónico, que entra en operación únicamente tras ingresar el código correcto en el keypad. Una vez activo, si el sensor detecta una lectura por debajo de 6 cm, el buzzer se enciende como medida de alerta. Este permanecerá activado hasta que se vuelva a ingresar el código correcto en el keypad.

6.2. Implementación de la camara VC0706

Seguidamente, se procedió con la implementación de la cámara VC0706. Para ello, se descargó la librería *Adafruit-VC0706-Serial-Camera-Library* y se realizaron las conexiones necesarias con el Arduino UNO. Como se mencionó anteriormente, se implementó un divisor de tensión en el pin

RX de la VC0706 para proteger el componente de posibles picos de voltaje durante la recepción de información.

En cuanto al código, se utilizó el ejemplo proporcionado por la librería para tomar fotografías. Sin embargo, se realizaron ciertas modificaciones para enviar las imágenes a través del puerto serial y guardarlas en la computadora. Además, se configuró el tamaño de la imagen a 160x120 píxeles, lo que permitió acelerar el proceso de transferencia de las capturas a la computadora. A continuación, se muestra el código utilizado y una imagen capturada por la cámara. Cabe destacar que la imagen aparece volteada debido a la posición en la que se encuentra la cámara:

```
#include <Adafruit_VC0706.h>
#include <SPI.h>

#if defined(__AVR__) || defined(ESP8266)
#include <SoftwareSerial.h>
SoftwareSerial cameraconnection(2, 3); // Conexion de la c mara: TX en pin 2, RX en pin 3
#else
#define cameraconnection Serial1
#endif

Adafruit_VC0706 cam = Adafruit_VC0706(&cameraconnection);

void setup() {
  Serial.begin(9600);
  Serial.println("VC0706 Camera Snapshot to Serial");

  // Inicializa la c mara
  if (cam.begin()) {
    Serial.println("Camera Found:");
  } else {
    Serial.println("No camera found?");
    return;
  }

  // Configura el tama o de la imagen a la resoluci n m s peque a para reducir el tama o
  cam.setImageSize(VC0706_160x120); // Tama o peque o 160x120
  uint8_t imgsize = cam.getImageSize();
  Serial.print("Image size: ");
  if (imgsize == VC0706_640x480) Serial.println("640x480");
  if (imgsize == VC0706_320x240) Serial.println("320x240");
  if (imgsize == VC0706_160x120) Serial.println("160x120");

  Serial.println("Listo para tomar foto. Env a el comando 'snapshot' desde el script de Python.");
}

void loop() {
  // Espera el comando 'snapshot' desde Python
```

```
  if (Serial.available() > 0) {
    String command = Serial.readStringUntil('\n');
    command.trim();

    if (command == "snapshot") {
      Serial.println("Taking␣picture...");

      if (!cam.takePicture()) {
        Serial.println("Failed␣to␣snap!");
        return;
      }
      Serial.println("Picture␣taken!");

      // Env a el tama o de la imagen a Python
      uint32_t jpglen = cam.frameLength();
      Serial.print("Storing␣");
      Serial.print(jpglen);
      Serial.println("␣byte␣image.");

      // Env a la imagen en bloques de 32 bytes
      while (jpglen > 0) {
        uint8_t bytesToRead = min((uint32_t)32, jpglen);
        uint8_t *buffer = cam.readPicture(bytesToRead);
        Serial.write(buffer, bytesToRead);  // Enviar el bloque de datos
        jpglen -= bytesToRead;
      }

      Serial.println("done!");  // Indica a Python que la imagen ha sido enviada completamente

      cam.resumeVideo();
    }
  }
}
```

Figura 3: Fotografía obtenida con la cámara VC0706

6.3. Implementación del IoT, Keypad y buzzer

Para el apartado de IoT se usará la plaforma Thinsboard, esta paltaforma presenta diferentes widgets para mostrar información o para activar mecanismos. Dado que ThingsBoard no cuenta con un widget específico para mostrar imágenes, se optó por utilizar un widget que permite recibir y mostrar contenido en formato HTML. Para ello, se desarrolló una función en el script de Python que genera un archivo HTML con la imagen capturada por el Arduino y lo envía a ThingsBoard a través del protocolo HTTP.

En cuanto a las notificaciones, se añadió un widget que muestra un mensaje de texto según el estado del sistema. Si se detecta movimiento, el widget informa al usuario que se ha detectado actividad y se le recomienda revisar la imagen capturada. Por otro lado, si no se detecta movimiento, el widget muestra un mensaje indicando que el sistema no ha encontrado movimiento. Estas notificaciones se envían mediante el protocolo MQTT, gestionado desde el script de Python.

Adicionalmente, se incluyó un widget tipo switch para activar o desactivar el sistema de seguridad. El valor del switch se obtiene mediante una llamada RPC (Remote Procedure Call) utilizando la función get_Value1. Cada vez que se modifica el estado del switch, ThingsBoard envía una respuesta al script de Python con un método y un parámetro. En el script, se verifica que el método coincida con el esperado y se lee el parámetro, que puede tener un valor de true o false.

Tras configurar el entorno de IoT, se procedió a realizar pruebas físicas tanto del keypad como del buzzer. Para ello, se desarrolló un código en Arduino que permite activar el buzzer ingresando el código correcto en el keypad o activando el switch en ThingsBoard. Toda la comunicación entre el script de Python y el Arduino se lleva a cabo mediante comunicación serial. Además, se verificó que el buzzer pueda desactivarse ingresando nuevamente el código en el keypad o apagando el switch en la plataforma ThingsBoard.

6.4. Implementación del sensor Ultrasónico

Por ultimó, se implementó el sensor ultrasonico en el circuito fisico para poder detectar el movimiento de objeto dentro de un rango especificó. Para ello se utilizó el siguiente código:

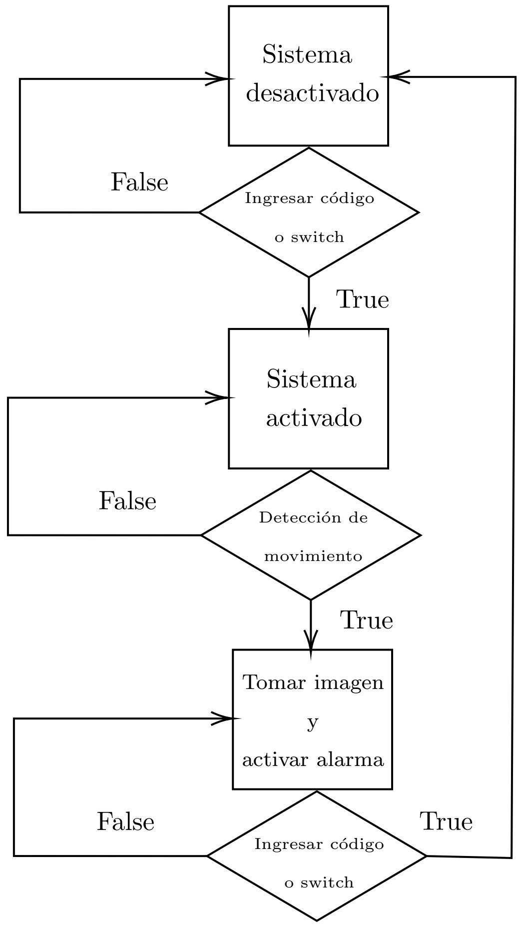
```
float measureDistance() {
  // Enviar un pulso de disparo
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Leer el tiempo de retorno del eco
  long duration = pulseIn(echoPin, HIGH);

  // Calcular la distancia en cm
  float distance = duration * 0.034 / 2;

  return distance;
}
```

Esta función retorna el valor de distancia medido el cual se utilizará para detectar ese movimiento. Luego de esto se unieron todas las implementaciones para crear el sistema final el cual presenta el siguiente flujo de trabajo:



A lo largo del proyecto, se desarrollaron diversas funciones en el script de Python para gestionar la comunicación tanto con el Arduino como con la plataforma ThingsBoard. Estas funciones permitieron enviar y recibir datos. Sin embargo, se presentó una complicación, ya que el script debía esperar simultáneamente dos eventos: el cambio de estado del switch en ThingsBoard y la detección de movimiento para procesar y recibir la imagen capturada.

Para resolver este desafío, se implementó la librería de Python threading, la cual permite crear y manejar múltiples hilos de ejecución de manera concurrente. Mediante esta solución, se configuraron dos hilos independientes: uno encargado de monitorear el estado del switch en ThingsBoard y otro dedicado a la detección de movimiento y la recepción de imágenes desde el Arduino.

6.5. Resultados y Análisis

A continuación se muestran los resultados obtenidos en el circuito:

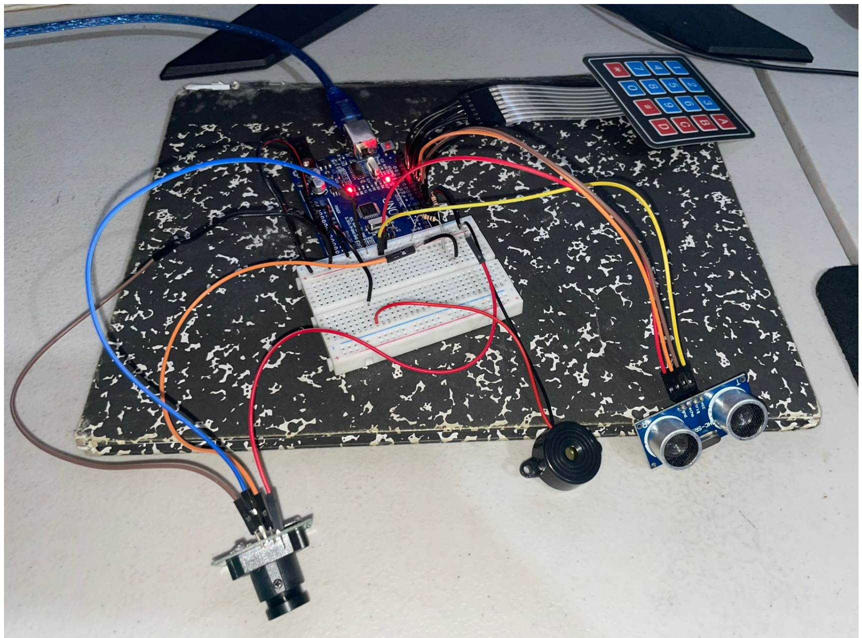


Figura 4: Circuito físico contruido

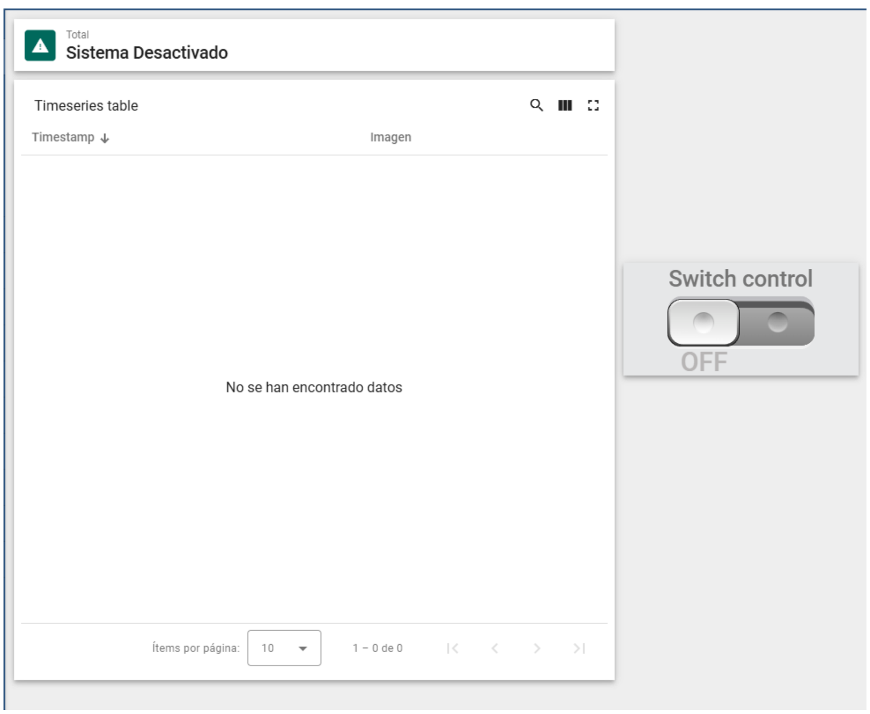


Figura 5: Dashboard de la plaforma IoT (Sistema Desactivado)

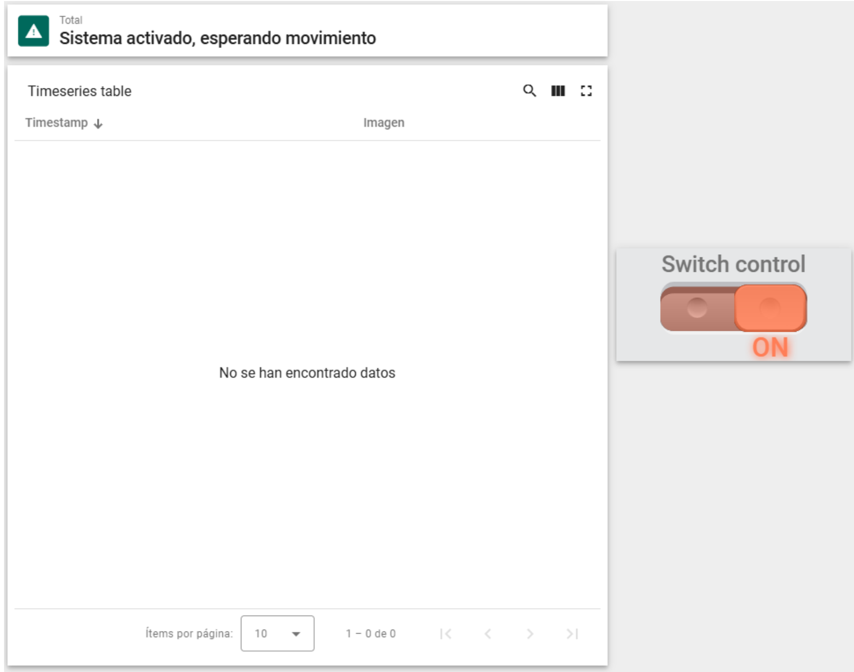


Figura 6: Dashboard de la plaforma IoT (Sistema Activado)

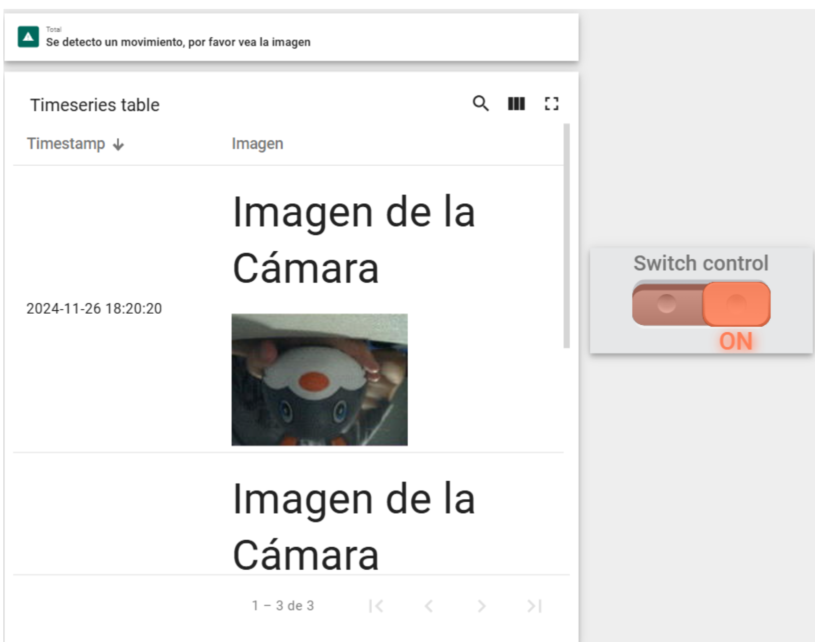


Figura 7: Dashboard de la plaforma IoT (Detección de movimiento)

Tras analizar los resultados obtenidos, se confirma que el circuito opera de acuerdo con lo esperado. En la figura 5, se observa cómo el dashboard de la plataforma ThingsBoard indica que el sistema de seguridad se encuentra desactivado, lo que significa que no está habilitado para detectar ningún movimiento.

Por otro lado, en la figura 6, se muestra el sistema activado tras accionar el switch en el dashboard.

Esto provoca que el Arduino inicie un monitoreo constante, realizando lecturas del sensor ultrasónico y permaneciendo a la espera de detectar un movimiento a una distancia inferior a los 6 centímetros.

Finalmente, en la figura 7, se presenta un ejemplo de detección de movimiento. En esta figura se puede observar cómo el sistema emite una notificación en el dashboard, informando al usuario sobre el suceso, y muestra la imagen capturada por la cámara. Es importante aclarar nuevamente que la imagen aparece girada debido a la posición invertida de la cámara, la cual fue instalada de esta manera para garantizar una conexión segura sin riesgo de que se desconecten los cables durante el funcionamiento.

7. Conclusiones y Recomendaciones

- Se logró diseñar y construir un circuito funcional que opera como un sistema de seguridad, capaz de detectar movimiento y notificar al usuario de manera efectiva. La notificación se realiza de forma sonora, a través de un buzzer, y visualmente, mostrando una imagen capturada del evento en la plataforma IoT configurada para el proyecto.

- Se incorporó un sensor ultrasónico, capaz de detectar el movimiento de un objeto dentro de un rango definido.

- Se implementó el internet de las cosas (IoT) utilizando la plataforma ThingsBoard, permitiendo activar y desactivar el sistema de seguridad de forma remota, recibir notificaciones en tiempo real y visualizar la imagen capturada del suceso.

- Se recomienda utilizar el protocolo HTTP para la transmisión del archivo HTML que contiene la imagen capturada, ya que el protocolo MQTT no funciona.

- Se recomienda utilizar los ejemplos proporcionados para la cámara VC0706, ya que estos sirven como una base sólida y simplifican significativamente el proceso de implementación.

Referencias

[1] Arduino. (2024). *A000066-datasheet.* Recuperado el 05 de octubre del 2024, de https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf.

[2] Elijah J. Morgan. (2014). *HCSR04 Ultrasonic Sensor.* Recuperado el 26 de noviembre del 2024, de https://www.alldatasheet.es/datasheet-pdf/pdf/1132203/ETC2/HC-SR04.html.

[3] Vimicro. (2007). *VC0706 Datasheet.* Recuperado el 26 de noviembre del 2024, de https://cdn-shop.adafruit.com/datasheets/VC0706%20Digital%20Video%20Processor%20Datasheet.pdf.

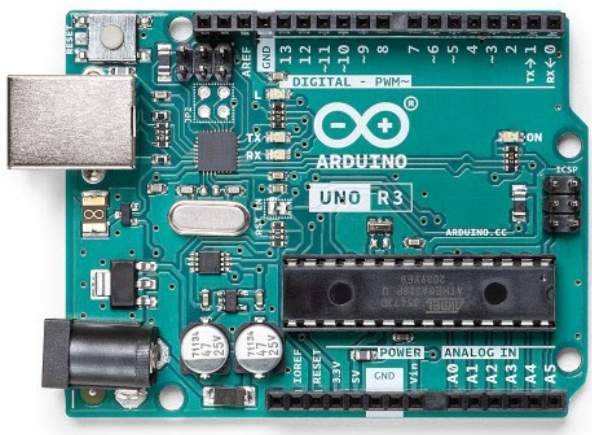
[4] pro-SIGNAL. *Buzzer.* Recuperado el 26 de noviembre del 2024, de https://www.farnell.com/datasheets/2171929.pdf.

[5] Parallax. *4x4 Matrix Membrane Keypad.* Recuperado el 26 de noviembre del 2024, de https://cdn.sparkfun.com/assets/f/f/a/5/0/DS-16038.pdf.

8. Apéndices

Se presentán los datasheets de los componetes utilizados

Product Reference Manual
SKU: A000066



Description

The Arduino® UNO R3 is the perfect board to get familiar with electronics and coding. This versatile development board is equipped with the well-known ATmega328P and the ATMega 16U2 Processor.

This board will give you a great first experience within the world of Arduino.

Target areas:

Maker, introduction, industries

Features

- **ATMega328P** Processor
  - **Memory**
    - AVR CPU at up to 16 MHz
    - 32 kB Flash
    - 2 kB SRAM
    - 1 kB EEPROM
  - **Security**
    - Power On Reset (POR)
    - Brown Out Detection (BOD)
  - **Peripherals**
    - 2x 8-bit Timer/Counter with a dedicated period register and compare channels
    - 1x 16-bit Timer/Counter with a dedicated period register, input capture and compare channels
    - 1x USART with fractional baud rate generator and start-of-frame detection
    - 1x controller/peripheral Serial Peripheral Interface (SPI)
    - 1x Dual mode controller/peripheral I2C
    - 1x Analog Comparator (AC) with a scalable reference input
    - Watchdog Timer with separate on-chip oscillator
    - Six PWM channels
    - Interrupt and wake-up on pin change
  - **ATMega16U2 Processor**
    - 8-bit AVR® RISC-based microcontroller
  - **Memory**
    - 16 kB ISP Flash
    - 512B EEPROM
    - 512B SRAM
    - debugWIRE interface for on-chip debugging and programming
  - **Power**
    - 2.7-5.5 volts

CONTENTS

1 The Board

1.1 Application Examples

The UNO board is the flagship product of Arduino. Regardless if you are new to the world of electronics or will use the UNO R3 as a tool for education purposes or industry-related tasks, the UNO R3 is likely to meet your needs.

First entry to electronics: If this is your first project within coding and electronics, get started with our most used and documented board; UNO. It is equipped with the well-known ATmega328P processor, 14 digital input/output pins, 6 analog inputs, USB connections, ICSP header and reset button. This board includes everything you will need for a great first experience with Arduino.

Industry-standard development board: Using the UNO R3 board in industries, there are a range of companies using the UNO R3 board as the brain for their PLC's.

Education purposes: Although the UNO R3 board has been with us for about ten years, it is still widely used for various education purposes and scientific projects. The board's high standard and top quality performance makes it a great resource to capture real time from sensors and to trigger complex laboratory equipment to mention a few examples.

1.2 Related Products

- Arduino Starter Kit
- Arduino UNO R4 Minima
- Arduino UNO R4 WiFi
- Tinkerkit Braccio Robot

2 Ratings

2.1 Recommended Operating Conditions

Symbol	Description	Min	Max
	Conservative thermal limits for the whole board:	-40 °C (-40 °F)	85 °C (185 °F)

NOTE: In extreme temperatures, EEPROM, voltage regulator, and the crystal oscillator, might not work as expected.

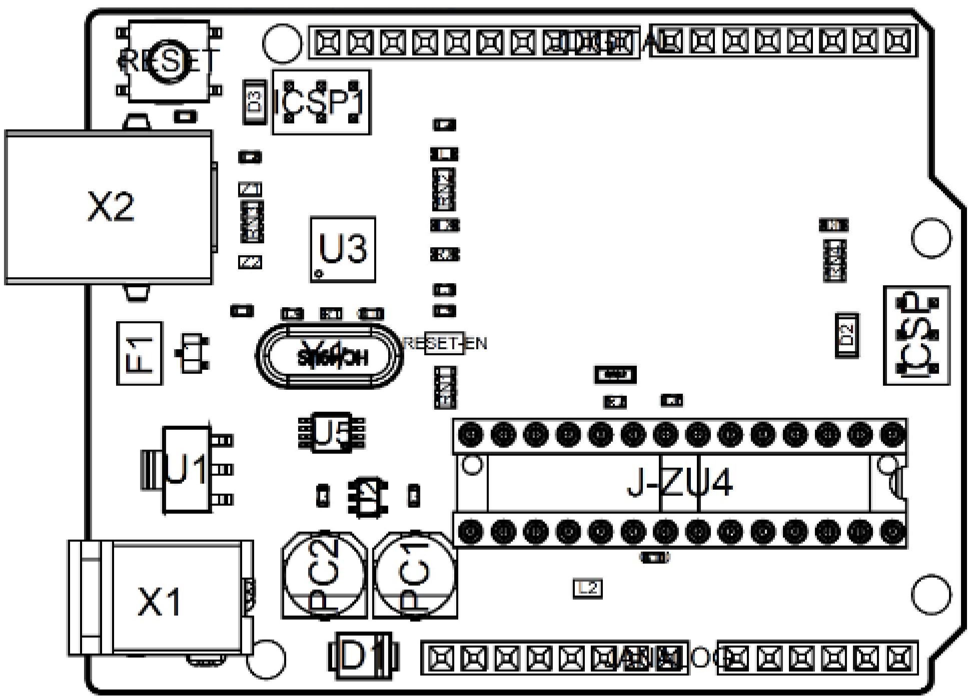
2.2 Power Consumption

Symbol	Description	Min	Typ	Max	Unit
VINMax	Maximum input voltage from VIN pad	6	-	20	V
VUSBMax	Maximum input voltage from USB connector		-	5.5	V
PMax	Maximum Power Consumption	-	-	xx	mA

3 Functional Overview

3.1 Board Topology

Top view



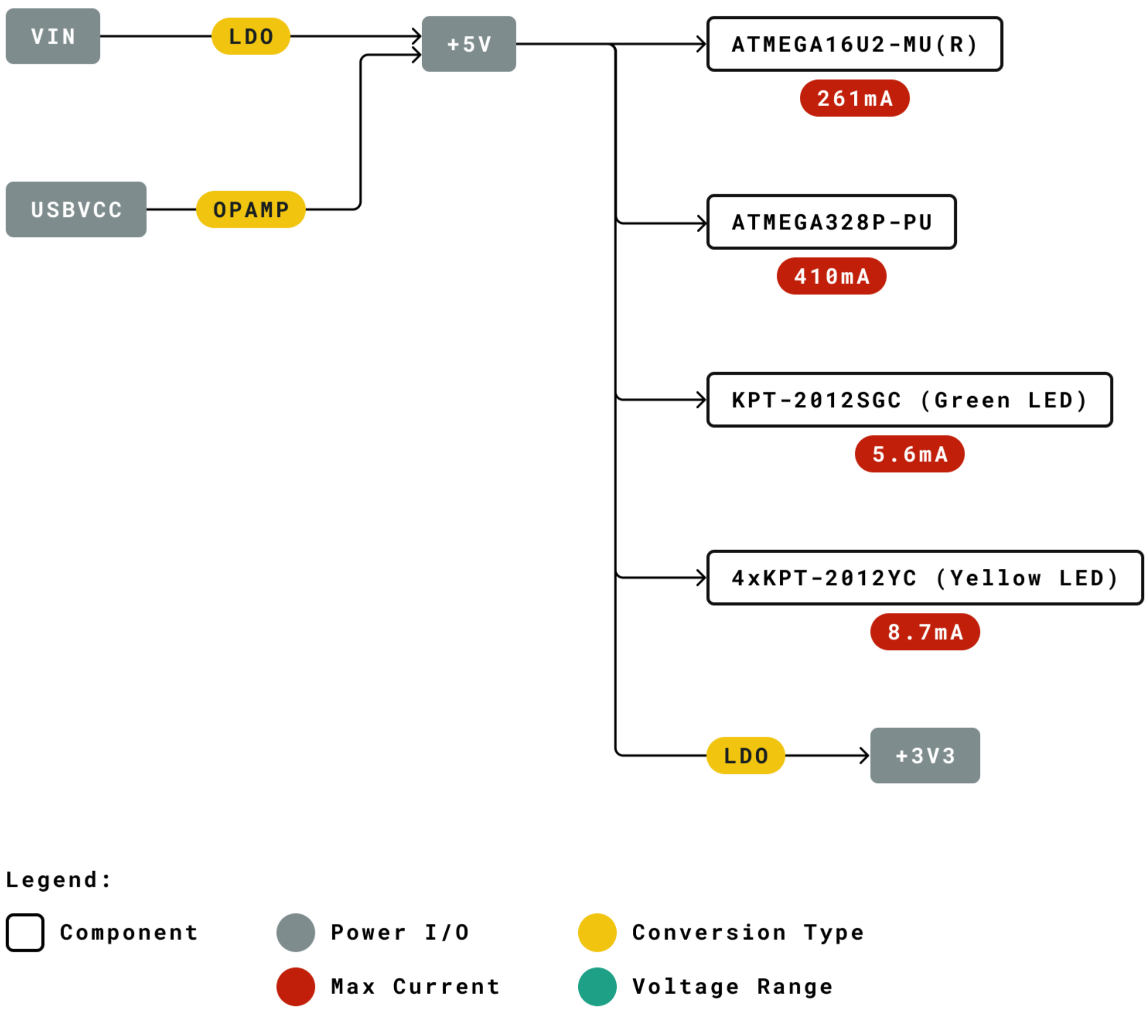
Board topology

Ref.	Description	Ref.	Description
X1	Power jack 2.1x5.5mm	U1	SPX1117M3-L-5 Regulator
X2	USB B Connector	U3	ATMEGA16U2 Module
PC1	EEE-1EA470WP 25V SMD Capacitor	U5	LMV358LIST-A.9 IC
PC2	EEE-1EA470WP 25V SMD Capacitor	F1	Chip Capacitor, High Density
D1	CGRA4007-G Rectifier	ICSP	Pin header connector (through hole 6)
J-ZU4	ATMEGA328P Module	ICSP1	Pin header connector (through hole 6)
Y1	ECS-160-20-4X-DU Oscillator		

3.2 Processor

The Main Processor is a ATmega328P running at up to 20 MHz. Most of its pins are connected to the external headers, however some are reserved for internal communication with the USB Bridge coprocessor.

3.3 Power Tree



Power tree

4 Board Operation

4.1 Getting Started - IDE

If you want to program your UNO R3 while offline you need to install the Arduino Desktop IDE [1] To connect the UNO R3 to your computer, you'll need a USB-B cable. This also provides power to the board, as indicated by the LED.

4.2 Getting Started - Arduino Cloud Editor

All Arduino boards, including this one, work out-of-the-box on the Arduino Cloud Editor [2], by just installing a simple plugin.

The Arduino Cloud Editor is hosted online, therefore it will always be up-to-date with the latest features and support for all boards. Follow **[3]** to start coding on the browser and upload your sketches onto your board.

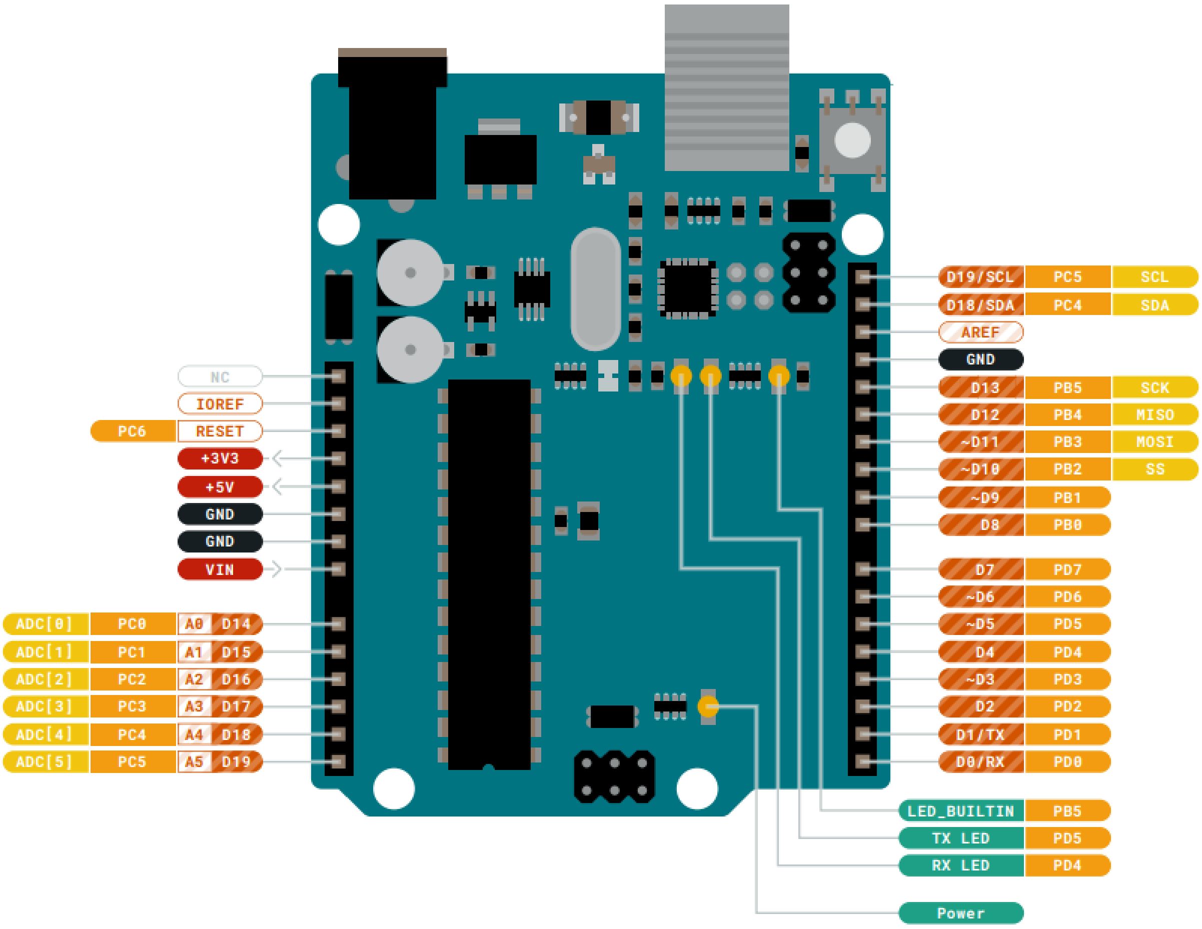
4.3 Sample Sketches

Sample sketches for the UNO R3 can be found either in the "Examples" menu in the Arduino IDE or in the "Documentation" section of the Arduino website [4].

4.4 Online Resources

Now that you have gone through the basics of what you can do with the board you can explore the endless possibilities it provides by checking exciting projects on Arduino Project Hub [5], the Arduino Library Reference [6] and the online Arduino store [7] where you will be able to complement your board with sensors, actuators and more.

5 Connector Pinouts



Pinout

5.1 JANALOG

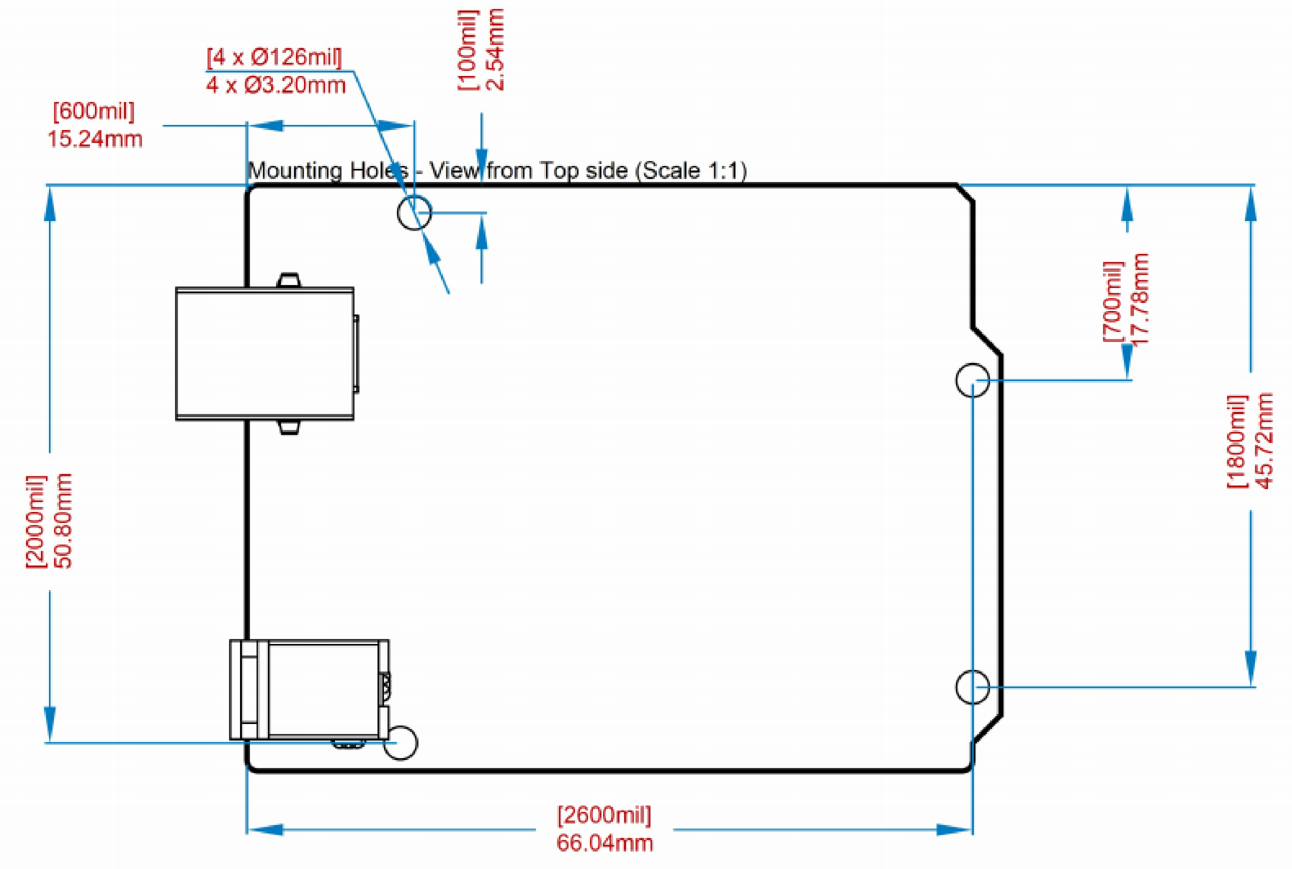
Pin	Function	Type	Description
1	NC	NC	Not connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog/GPIO	Analog input 0 /GPIO
10	A1	Analog/GPIO	Analog input 1 /GPIO
11	A2	Analog/GPIO	Analog input 2 /GPIO
12	A3	Analog/GPIO	Analog input 3 /GPIO
13	A4/SDA	Analog input/I2C	Analog input 4/I2C Data line
14	A5/SCL	Analog input/I2C	Analog input 5/I2C Clock line

5.2 JDIGITAL

Pin	Function	Type	Description
1	D0	Digital/GPIO	Digital pin 0/GPIO
2	D1	Digital/GPIO	Digital pin 1/GPIO
3	D2	Digital/GPIO	Digital pin 2/GPIO
4	D3	Digital/GPIO	Digital pin 3/GPIO
5	D4	Digital/GPIO	Digital pin 4/GPIO
6	D5	Digital/GPIO	Digital pin 5/GPIO
7	D6	Digital/GPIO	Digital pin 6/GPIO
8	D7	Digital/GPIO	Digital pin 7/GPIO
9	D8	Digital/GPIO	Digital pin 8/GPIO
10	D9	Digital/GPIO	Digital pin 9/GPIO
11	SS	Digital	SPI Chip Select
12	MOSI	Digital	SPI1 Main Out Secondary In
13	MISO	Digital	SPI Main In Secondary Out
14	SCK	Digital	SPI serial clock output
15	GND	Power	Ground
16	AREF	Digital	Analog reference voltage
17	A4/SD4	Digital	Analog input 4/I2C Data line (duplicated)
18	A5/SD5	Digital	Analog input 5/I2C Clock line (duplicated)

5.3 Mechanical Information

5.4 Board Outline & Mounting Holes



Board outline

6 Certifications

6.1 Declaration of Conformity CE DoC (EU)

We declare under our sole responsibility that the products above are in conformity with the essential requirements of the following EU Directives and therefore qualify for free movement within markets comprising the European Union (EU) and European Economic Area (EEA).

ROHS 2 Directive 2011/65/EU	
Conforms to:	EN50581:2012
Directive 2014/35/EU. (LVD)	
Conforms to:	EN 60950-1:2006/A11:2009/A1:2010/A12:2011/AC:2011
Directive 2004/40/EC & 2008/46/EC & 2013/35/EU, EMF	
Conforms to:	EN 62311:2008

6.2 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021

Arduino boards are in compliance with RoHS 2 Directive 2011/65/EU of the European Parliament and RoHS 3 Directive 2015/863/EU of the Council of 4 June 2015 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Substance	Maximum limit (ppm)
Lead (Pb)	1000
Cadmium (Cd)	100
Mercury (Hg)	1000
Hexavalent Chromium (Cr6+)	1000
Poly Brominated Biphenyls (PBB)	1000
Poly Brominated Diphenyl ethers (PBDE)	1000
Bis(2-Ethylhexyl} phthalate (DEHP)	1000
Benzyl butyl phthalate (BBP)	1000
Dibutyl phthalate (DBP)	1000
Diisobutyl phthalate (DIBP)	1000

Exemptions: No exemptions are claimed.

Arduino Boards are fully compliant with the related requirements of European Union Regulation (EC) 1907 /2006 concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH). We declare none of the SVHCs (https://echa.europa.eu/web/guest/candidate-list-table), the Candidate List of Substances of Very High Concern for authorization currently released by ECHA, is present in all products (and also package) in quantities totaling in a concentration equal or above 0.1%. To the best of our knowledge, we also declare that our products do not contain any of the substances listed on the "Authorization List" (Annex XIV of the REACH regulations) and Substances of Very High Concern (SVHC) in any significant amounts as specified by the Annex XVII of Candidate list published by ECHA (European Chemical Agency) 1907 /2006/EC.

6.3 Conflict Minerals Declaration

As a global supplier of electronic and electrical components, Arduino is aware of our obligations with regards to laws and regulations regarding Conflict Minerals, specifically the Dodd-Frank Wall Street Reform and Consumer Protection Act, Section 1502. Arduino does not directly source or process conflict minerals such as Tin, Tantalum, Tungsten, or Gold. Conflict minerals are contained in our products in the form of solder, or as a component in metal alloys. As part of our reasonable due diligence Arduino has contacted component suppliers within our supply chain to verify their continued compliance with the regulations. Based on the information received thus far we declare that our products contain Conflict Minerals sourced from conflict-free areas.

7 FCC Caution

Any Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference

(2) this device must accept any interference received, including interference that may cause undesired operation.

FCC RF Radiation Exposure Statement:

1. This Transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2. This equipment complies with RF radiation exposure limits set forth for an uncontrolled environment.
3. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

English: User manuals for license-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada license-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR Warning:

English This equipment should be installed and operated with minimum distance 20 cm between the radiator and your body.

French: Lors de l’ installation et de l’ exploitation de ce dispositif, la distance entre le radiateur et le corps est d ’au moins 20 cm.

Important: The operating temperature of the EUT can’t exceed 85℃ and shouldn’t be lower than -40℃.

Hereby, Arduino S.r.l. declares that this product is in compliance with essential requirements and other relevant provisions of Directive 2014/53/EU. This product is allowed to be used in all EU member states.

8 Company Information

Company name	Arduino S.r.l
Company Address	Via Andrea Appiani 25 20900 MONZA Italy

9 Reference Documentation

Reference	Link
Arduino IDE (Desktop)	https://www.arduino.cc/en/Main/Software
Arduino Cloud Editor	https://create.arduino.cc/editor
Arduino Cloud Editor - Getting Started	https://docs.arduino.cc/arduino-cloud/guides/editor/
Arduino Website	https://www.arduino.cc/
Arduino Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
Library Reference	https://www.arduino.cc/reference/en/
Arduino Store	https://store.arduino.cc/

10 Revision History

Date	Revision	Changes
25/04/2024	3	Updated link to new Cloud Editor
26/07/2023	2	General Update
06/2021	1	Datasheet release

中文（ZH）

描述

Arduino UNO R3 是熟悉电子技术和编码的完美开发板。这款多功能开发板配备了著名的 ATmega328P 和 ATMega 16U2 处理器。该开发板将为您带来 Arduino 世界绝佳的初次体验。

目标领域：

创客、介绍、工业领域

特点

- **ATMega328P** 处理器
    - **内存**
        - AVR CPU 频率高达 16 MHz
        - 32KB 闪存
        - 2KB SRAM
        - 1KB EEPROM
    - **安全性**
        - 上电复位 (POR)
        - 欠压检测 (BOD)
    - **外设**
        - 2x 8 位定时器/计数器，带专用周期寄存器和比较通道
        - 1x 16 位定时器/计数器，带专用周期寄存器、输入捕获和比较通道
        - 1x USART，带分数波特率发生器和起始帧信号检测功能
        - 1x 控制器/外设串行外设接口 (SPI)
        - 1x 双模控制器/外设 I2C
        - 1 个模拟比较器 (AC)，带可扩展参考输入
        - 看门狗定时器，带独立的片上振荡器
        - 6 通道 PWM
        - 引脚变化时的中断和唤醒
    - **ATMega16U2 处理器**
        - 基于 AVR® RISC 的 8 位微控制器
    - **内存**
        - 16 KB ISP 闪存
        - 512B EEPROM
        - 512B SRAM

    - 用于片上调试和编程的 debugWIRE 接口
- **电源**
    - 2.7-5.5 伏特

目录

11 电路板简介

11.1 应用示例

UNO 电路板是 Arduino 的旗舰产品。无论您是初次接触电路板产品，还是将 UNO 用作教育或工业相关任务的工具，UNO 都能满足您的需求。

初次接触电子技术: 如果这是您第一次参与编码和电子技术项目，那么就从我们最常用、记录最多的电路板 Arduino UNO 开始吧。它配备了著名的 ATmega328P 处理器、14 个数字输入/输出引脚、6 个模拟输入、USB 连接、ICSP 接头和复位按钮。该电路板包含了您获得良好的 Arduino 初次体验所需的一切。

** 行业标准开发板:** 在工业领域使用 Arduino UNO R3 开发板，有许多公司使用 UNO 开发板作为其 PLC 的大脑。

教育用途: 尽管我们推出 UNO R3 电路板已有大约十年之久，但它仍被广泛用于各种教育用途和科学项目。该电路板的高标准和一流性能使其成为从传感器采集实时数据和触发复杂实验室设备等各种应用场合的绝佳资源。

11.2 相关产品

- Starter Kit
- Arduino UNO R4 Minima
- Arduino UNO R4 WiFi
- Tinkerkit Braccio Robot

12 额定值

12.1 建议运行条件

符号	描述	最小值	最大值
	整个电路板的保守温度极限值:	-40 °C (-40°F)	85 °C (185°F)

注意： 在极端温度下，EEPROM、电压调节器和晶体振荡器可能无法正常工作。

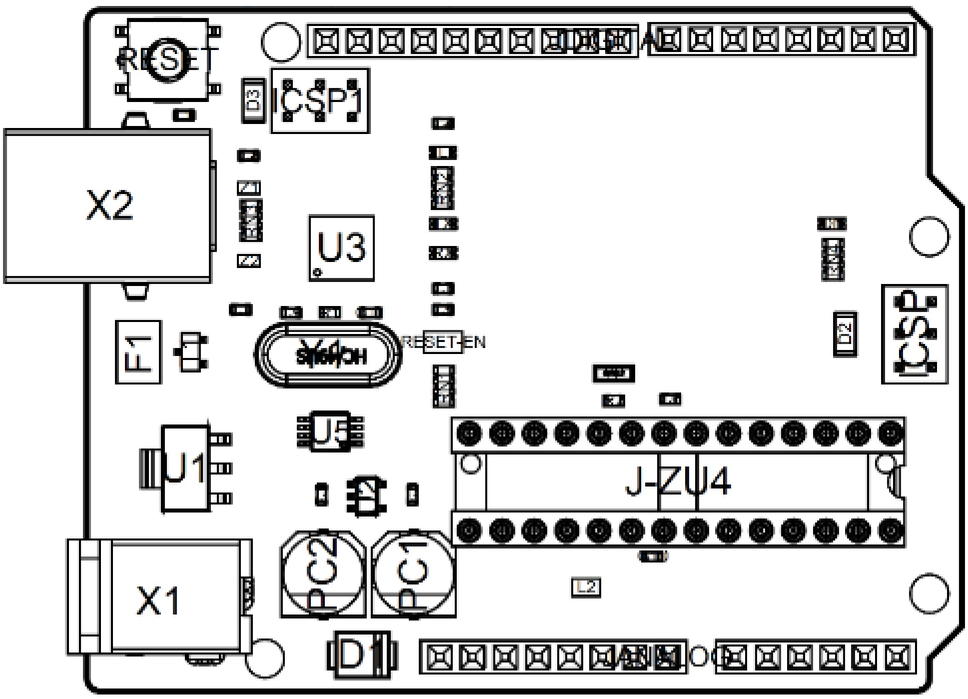
12.2 功耗

符号	描述	最小值	典型值	最大值	单位
VINMax	来自 VIN 焊盘的最大输入电压	6	-	20	V
VUSBMax	来自 USB 连接器的最大输入电压		-	5.5	V
PMax	最大功耗	-	-	xx	mA

13 功能概述

13.1 电路板拓扑结构

俯视图



电路板拓扑结构

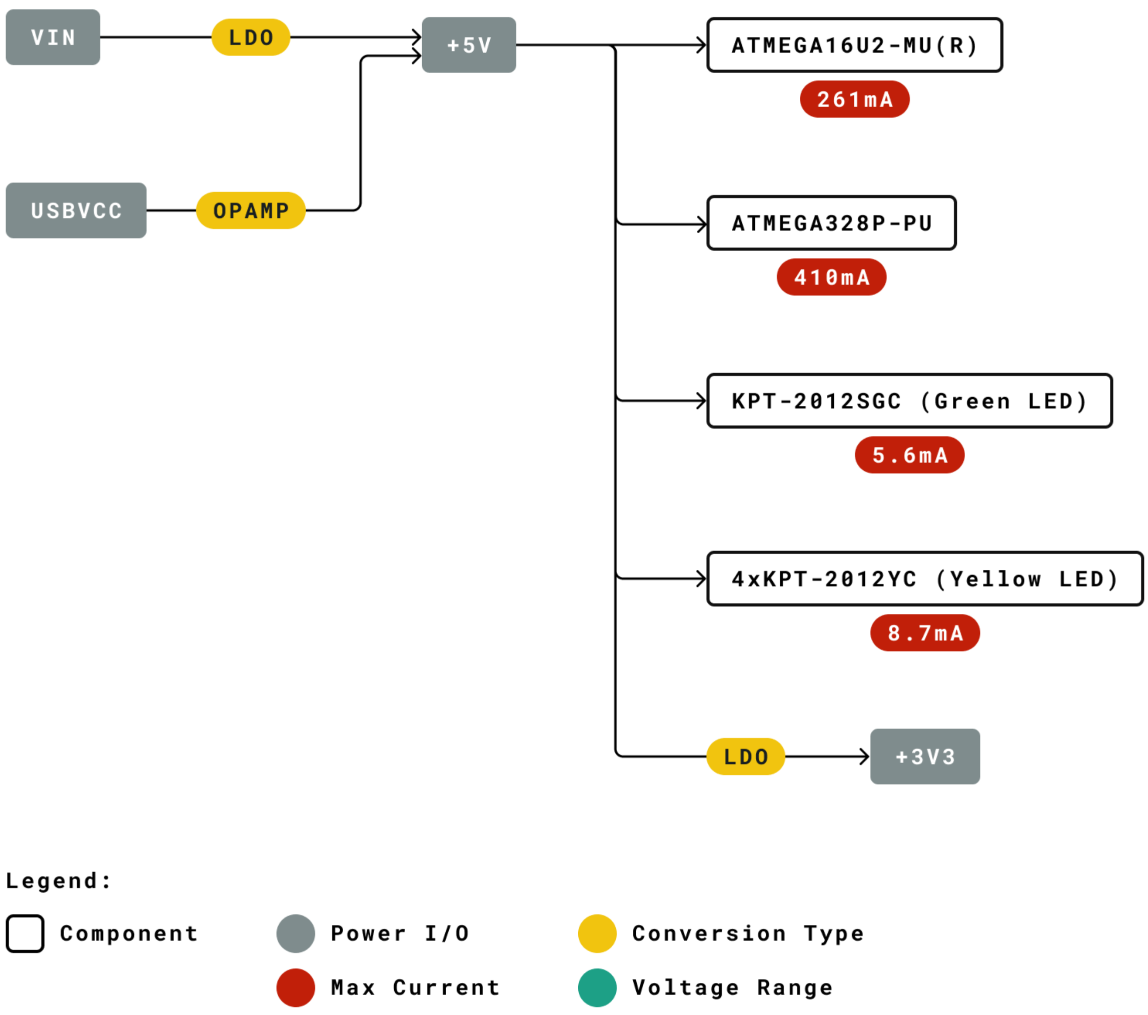
编号	描述	编号	描述
X1	电源插孔 2.1x5.5 毫米	U1	SPX1117M3-L-5 调节器

编号	描述	编号	描述
X2	USB B 连接器	U3	ATMEGA16U2 模块
PC1	EEE-1EA470WP 25V SMD 电容器	U5	LMV358LIST-A.9 IC
PC2	EEE-1EA470WP 25V SMD 电容器	F1	片式电容器，高密度
D1	CGRA4007-G 整流器	ICSP	引脚接头连接器（通过 6 号孔）
J-ZU4	ATMEGA328P 模块	ICSP1	引脚接头连接器（通过 6 号孔）
Y1	ECS-160-20-4X-DU 振荡器		

13.2 处理器

主处理器是 ATmega328P，运行频率高达 20 MHz。它的大部分引脚都与外部接头相连，但也有一些引脚用于与 USB 桥协处理器进行内部通信。

13.3 电源树



电源树

14 电路板操作

14.1 入门指南 - IDE

如需在离线状态下对 Arduino UNO R3 进行编程，则需要安装 Arduino Desktop IDE [1] 若要将 Arduino UNO 连接到计算机，需要使用 USB-B 电缆。如 LED 指示灯所示，该电缆还可以为电路板供电。

14.2 入门指南 - Arduino Cloud Editor

包括本电路板在内的所有 Arduino 电路板，都可以在 Arduino Cloud Editor [2] 上开箱即用，只需安装一个简单的插件即可。

Arduino Cloud Editor 是在线托管的，因此它将始终提供最新功能并支持所有电路板。接下来**[3]**开始在浏览器上编码并将程序上传到您的电路板上。

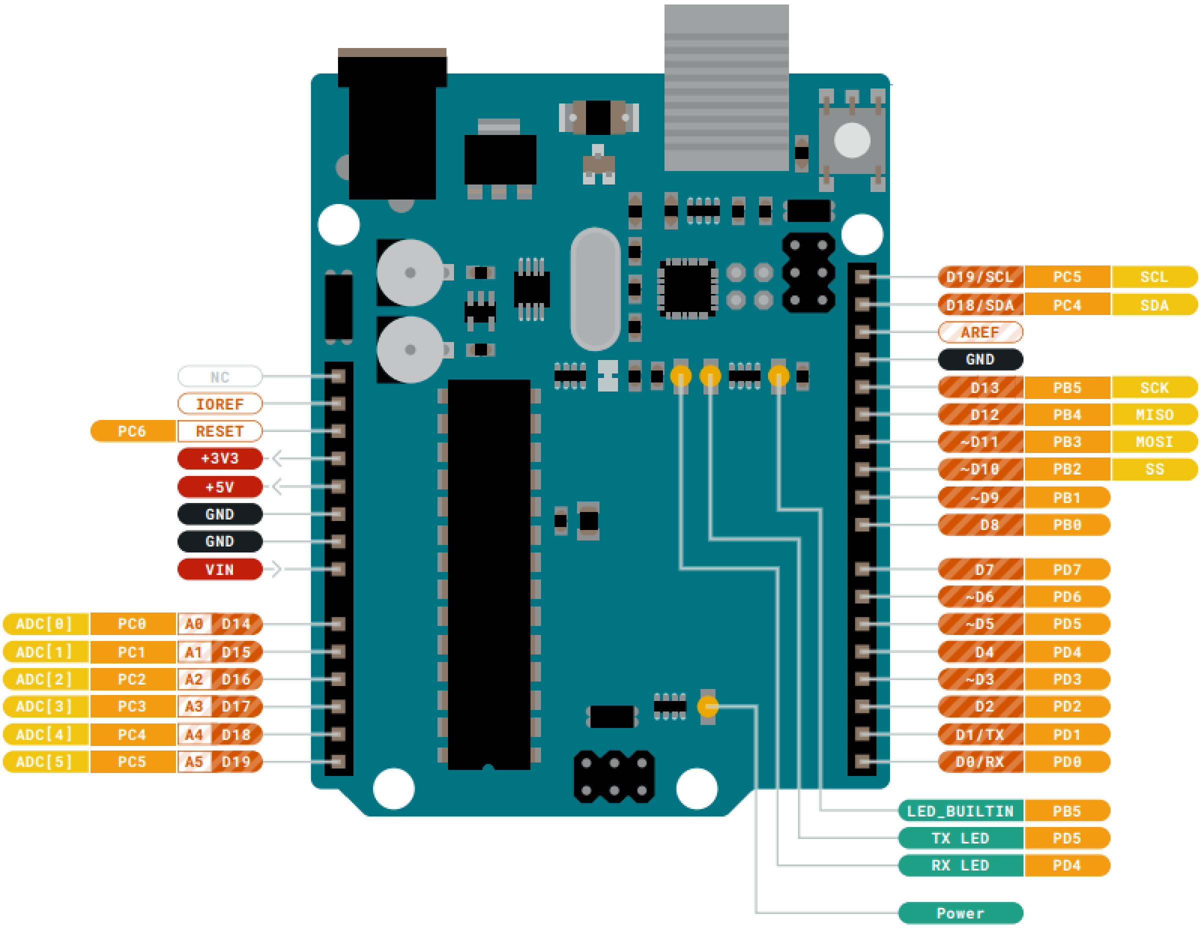
14.3 示例程序

Arduino UNO R3 的示例程序可以在 Arduino IDE 的"示例"菜单或 Arduino 网站 [4] 的"文档"部分找到

14.4 在线资源

现在，您已经了解该电路板的基本功能，就可以通过查看 Arduino Project Hub **[5]**、Arduino Library Reference **[6]** 以及在线 Arduino 商店 **[7]**上的精彩项目来探索它所提供的无限可能性；在这些项目中，您可以为电路板配备传感器、执行器等。

15 连接器引脚布局



布局

15.1 JANALOG

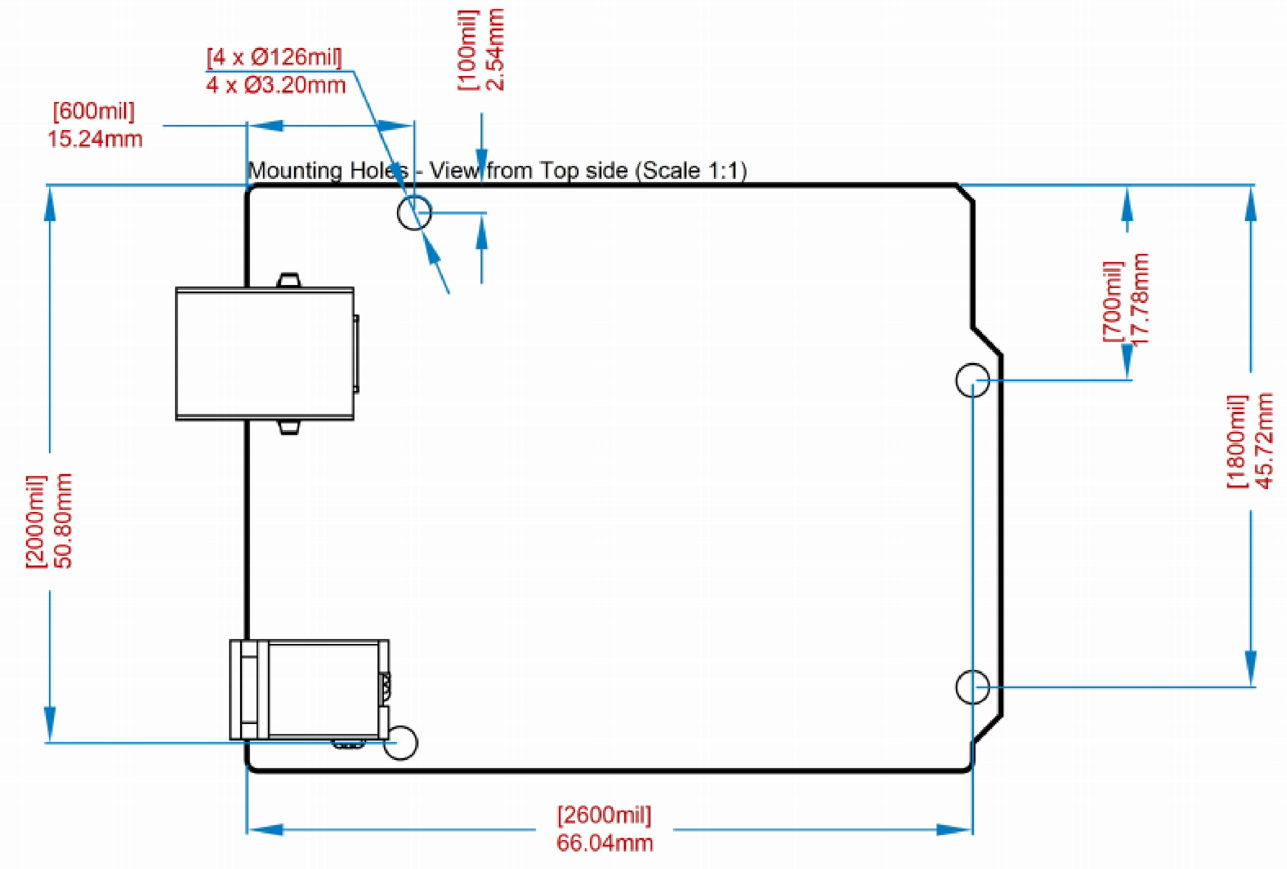
引脚	功能	类型	描述
1	NC	NC	未连接
2	IOREF	IOREF	数字逻辑参考电压 V - 连接至 5V
3	复位	复位	复位
4	+3V3	电源	+3V3 电源轨
5	+5V	电源	+5V 电源轨
6	GND	电源	接地
7	GND	电源	接地
8	VIN	电源	电压输入
9	A0	模拟/GPIO	模拟输入0 / GPIO
10	A1	模拟/GPIO	模拟输入1 / GPIO
11	A2	模拟/GPIO	模拟输入2 / GPIO
12	A3	模拟/GPIO	模拟输入3 / GPIO
13	A4/SDA	模拟输入/I2C	模拟输入 4/I2C 数据线
14	A5/SCL	模拟输入/I2C	模拟输入 5/I2C 时钟线

15.2 JDIGITAL

引脚	功能	类型	描述
1	D0	数字引脚/GPIO	数字引脚 0/GPIO
2	D1	数字引脚/GPIO	数字引脚 1/GPIO
3	D2	数字引脚/GPIO	数字引脚 2/GPIO
4	D3	数字引脚/GPIO	数字引脚 3/GPIO
5	D4	数字引脚/GPIO	数字引脚 4/GPIO
6	D5	数字引脚/GPIO	数字引脚 5/GPIO
7	D6	数字引脚/GPIO	数字引脚 6/GPIO
8	D7	数字引脚/GPIO	数字引脚 7/GPIO
9	D8	数字引脚/GPIO	数字引脚 8/GPIO
10	D9	数字引脚/GPIO	数字引脚 9/GPIO
11	SS	数字	SPI 芯片选择
12	MOSI	数字	SPI1 主输出副输入
13	MISO	数字	SPI 主输入副输出
14	SCK	数字	SPI 串行时钟输出
15	GND	电源	接地
16	AREF	数字	模拟参考电压
17	A4/SD4	数字	模拟输入 4/I2C 数据线（重复）
18	A5/SD5	数字	模拟输入 5/I2C 时钟线（重复）

15.3 机械层信息

15.4 电路板外形图和安装孔



电路板外形图

16 认证

16.1 符合性声明 CE DoC（欧盟）

我们在此郑重声明，上述产品符合以下欧盟指令的基本要求，因此有资格在包括欧盟（EU）和欧洲经济区（EEA）在内的市场内自由流通。

RoHS 2 指令 2011/65/EU	
符合：	EN50581:2012
指令 2014/35/EU。（LVD）	
符合：	EN 60950-1:2006/A11:2009/A1:2010/A12:2011/AC:2011
指令 2004/40/EC & 2008/46/EC & 2013/35/EU, EMF	
符合：	EN 62311:2008

16.2 声明符合欧盟 RoHS 和 REACH 211 01/19/2021

Arduino 电路板符合欧洲议会关于限制在电子电气设备中使用某些有害物质的 RoHS 2 指令 2011/65/EU 和欧盟理事会于 2015 年 6 月 4 日颁布的关于限制在电子电气设备中使用某些有害物质的 RoHS 3 指令 2015/863/EU。

物质	**最大限值 (ppm)**
铅 (Pb)	1000
镉 (Cd)	100
汞 (Hg)	1000
六价铬（Cr6+）	1000
多溴联苯（PBB）	1000
多溴联苯醚（PBDE）	1000
邻苯二甲酸二(2-乙基己)酯 (DEHP)	1000
邻苯二甲酸丁苄酯 (BBP)	1000
邻苯二甲酸二丁酯（DBP）	1000
邻苯二甲酸二异丁酯（DIBP）	1000

豁免：未申请任何豁免。

Arduino 电路板完全符合欧盟法规 (EC) 1907/2006 中关于化学品注册、评估、许可和限制 (REACH) 的相关要求。我们声明，所有产品（包括包装）中的 SVHC (https://echa.europa.eu/web/guest/candidate-list-table),（欧洲化学品管理局目前发布的《高度关注物质候选授权清单》）含量总浓度均未超过 0.1%。据我们所知，我们还声明，我们的产品不含 ECHA（欧洲化学品管理局）1907/2006/EC 公布的候选清单附件 XVII 中规定的"授权清单"（REACH 法规附件 XIV）和高度关注物质 (SVHC) 所列的任何物质。

16.3 冲突矿产声明

作为电子和电气元件的全球供应商，Arduino 意识到我们有义务遵守有关冲突矿产的法律法规，特别是《多德-弗兰克华尔街改革与消费者保护法案》第 1502 条。Arduino 不直接采购或加工锡、钽、钨或金等冲突矿物。冲突矿物以焊料的形式或作为金属合金的组成部分存在于我们的产品中。作为我们合理尽职调查的一部分，Arduino 已联系供应链中的元件供应商，以核实他们是否始终遵守法规的相关规定。根据迄今收到的信息，我们声明我们的产品中含有来自非冲突地区的冲突矿物。

17 FCC 警告

任何未经合规性负责方明确批准的更改或修改都可能导致用户无权操作设备。

本设备符合 FCC 规则第 15 部分的规定。操作须满足以下两个条件：

(1) 此设备不会造成有害干扰

(2) 此设备必须接受接收到的任何干扰，包括可能导致不良操作的干扰。

FCC 射频辐射暴露声明:

1. 此发射器不得与任何其他天线或发射器放置在同一位置或同时运行。
2. 此设备符合为非受控环境规定的射频辐射暴露限值。
3. 安装和操作本设备时，辐射源与您的身体之间至少应保持 20 厘米的距离。

English: User manuals for license-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada license-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR警告:

English This equipment should be installed and operated with a minimum distance of 20 cm between the radiator and your body.

French: Lors de l' installation et de l' exploitation de ce dispositif, la distance entre le radiateur et le corps est d 'au moins 20 cm.

重要提示： EUT 的工作温度不能超过 85°C，也不能低于 -40°C。

Arduino S.r.l. 特此声明，本产品符合 2014/53/EU 指令的基本要求和其他相关规定。本产品允许在所有欧盟成员国使用。

18 公司信息

公司名称	Arduino S.r.l
公司地址	Via Andrea Appiani 25 20900 MONZA Italy

19 参考资料

参考资料	链接
Arduino IDE (Desktop)	https://www.arduino.cc/en/Main/Software
Arduino IDE (Cloud)	https://create.arduino.cc/editor
Cloud IDE 入门指南	https://create.arduino.cc/projecthub/Arduino_Genuino/getting-started-with-arduino-web-editor-4b3e4a
Arduino 网站	https://www.arduino.cc/
Arduino Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
库参考	https://www.arduino.cc/reference/en/
在线商店	https://store.arduino.cc/

20 修订记录

日期	版次	变更
2023/07/26	2	一般更新
2021/06	1	数据表发布

HC-SR04 Ultrasonic Sensor

Elijah J. Morgan

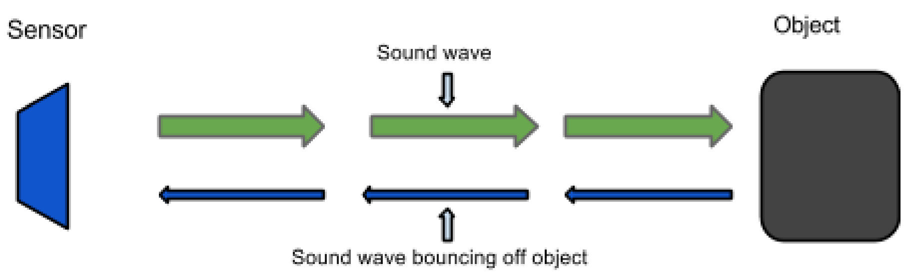
Nov. 16 2014

The purpose of this file is to explain how the HC-SR04 works. It will give a brief explanation of how ultrasonic sensors work in general. It will also explain how to wire the sensor up to a microcontroller and how to take/interpret readings. It will also discuss some sources of errors and bad readings.

1. **How Ultrasonic Sensors Work**
2. **HC-SR04 Specifications**
3. **Timing chart, Pin explanations and Taking Distance Measurements**
4. **Wiring HC-SR04 with a microcontroller**
5. **Errors and Bad Readings**

1. How Ultrasonic Sensors Work

Ultrasonic sensors use sound to determine the distance between the sensor and the closest object in its path. How do ultrasonic sensors do this? Ultrasonic sensors are essentially sound sensors, but they operate at a frequency above human hearing.



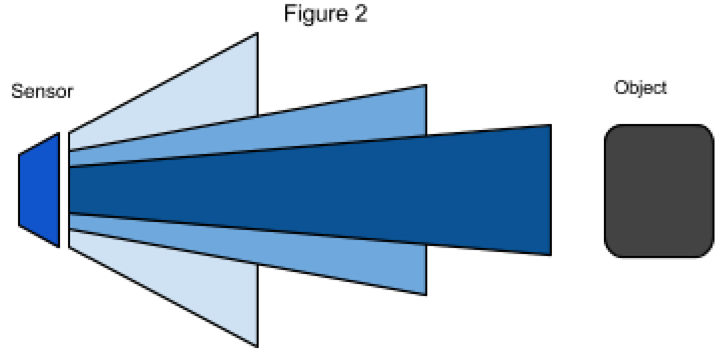
The sensor sends out a sound wave at a specific frequency. It then listens for that specific sound wave to bounce off of an object and come back (Figure 1). The sensor keeps track of the time between sending the sound wave and the sound wave returning. If you know how fast something is going and how long it is traveling you can find the distance traveled with equation 1.

Equation 1. $d = v \times t$

The speed of sound can be calculated based on the a variety of atmospheric conditions, including temperature, humidity and pressure. Actually calculating the distance will be shown later on in this document.

It should be noted that ultrasonic sensors have a cone of detection, the angle of this cone varies with distance, Figure 2 show this relation. The ability of a sensor to detect an object also depends on the objects orientation to the sensor. If an object doesn't present a flat surface to the sensor then it is possible the sound wave will bounce off the object in a way that it does not return to the sensor.

Figure 2



2. HC-SR04 Specifications

The sensor chosen for the Firefighting Drone Project was the HC-SR04. This section contains the specifications and why they are important to the sensor module. The sensor modules requirements are as follows.

- Cost
- Weight
- Community of hobbyists and support
- Accuracy of object detection
- Probability of working in a smoky environment
- Ease of use

The HC-SR04 Specifications are listed below. These specifications are from the Cytron Technologies HC-SR04 User's Manual (source 1).

- Power Supply: +5V DC
- Quiescent Current: <2mA
- Working current: 15mA
- Effectual Angle: <15º
- Ranging Distance: 2-400 cm
- Resolution: 0.3 cm
- Measuring Angle: 30º
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm
- Weight: approx. 10 g

The HC-SR04's best selling point is its price; it can be purchased at around $2 per unit.

3. Timing Chart and Pin Explanations

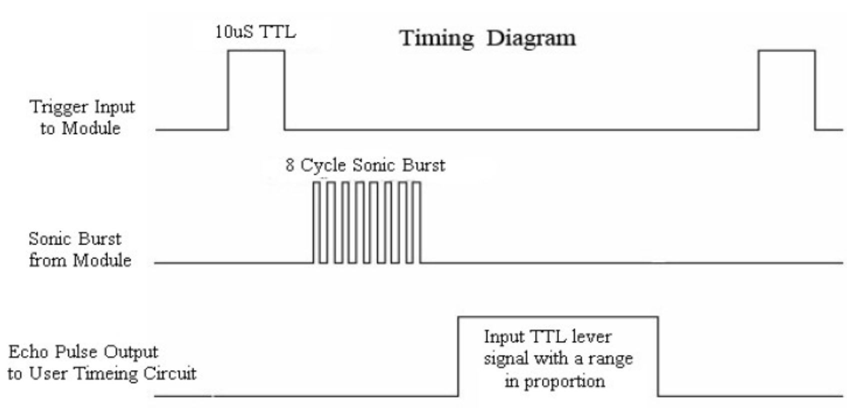
The HC-SR04 has four pins, VCC, GND, TRIG and ECHO; these pins all have different functions. The VCC and GND pins are the simplest -- they power the HC-SR04. These pins need to be attached to a +5 volt source and ground respectively. There is a single control pin: the TRIG pin. The TRIG pin is responsible for sending the ultrasonic burst. This pin should be set to HIGH for 10 μs, at which point the HC-SR04 will send out an eight cycle sonic burst at 40 kHZ. After a sonic burst has been sent the ECHO pin will go HIGH. The ECHO pin is the data pin -- it is used in taking distance measurements. After an ultrasonic burst is sent the pin will go HIGH, it will stay high until an ultrasonic burst is detected back, at which point it will go LOW.

Taking Distance Measurements

The HC-SR04 can be triggered to send out an ultrasonic burst by setting the TRIG pin to HIGH. Once the burst is sent the ECHO pin will automatically go HIGH. This pin will remain HIGH until the the burst hits the sensor again. You can calculate the distance to the object by keeping track of how long the ECHO pin stays HIGH. The time ECHO stays HIGH is the time the burst spent traveling. Using this measurement in equation 1 along with the speed of sound will yield the distance travelled. A summary of this is listed below, along with a visual representation in Figure 2.

1. Set TRIG to HIGH
2. Set a timer when ECHO goes to HIGH
3. Keep the timer running until ECHO goes to LOW
4. Save that time
5. Use equation 1 to determine the distance travelled

Figure 3
Source 2



Source 2

To interpret the time reading into a distance you need to change equation 1. The clock on the device you are using will probably count in microseconds or smaller. To use equation 1 the speed of sound needs to determined,which is 343 meters per second at standard temperature and pressure. To convert this into more useful form use equation 2 to change from meters per second to microseconds per centimeter. Then equation 3 can be used to easily compute the distance in centimeters.

Equation 2. $Distance\ =\ \frac{Speed}{170.15\ m} \times \frac{Meters}{100\ cm} \times \frac{1e6\ \mu S}{170.15\ m} \times \frac{58.772\ \mu S}{cm}$

Equation 3. $Distance\ =\ \frac{time}{58} = \frac{\mu s}{\mu s/cm} = cm$

4. Wiring the HC-SR04 to a Microcontroller

This section only covers the hardware side. For information on how to integrate the software side, look at one of the links below or look into the specific microcontroller you are using.

The HC-SR04 has 4 pins: VCC, GND, TRIG and ECHO.

1. VCC is a 5v power supply. This should come from the microcontroller
2. GND is a ground pin. Attach to ground on the microcontroller.
3. TRIG should be attached to a GPIO pin that can be set to HIGH
4. ECHO is a little more difficult. The HC-SR04 outputs 5v, which could destroy many microcontroller GPIO pins (the maximum allowed voltage varies). In order to step down the voltage use a single resistor or a voltage divider circuit. Once again this depends on the specific microcontroller you are using, you will need to find out its GPIO maximum voltage and make sure you are below that.

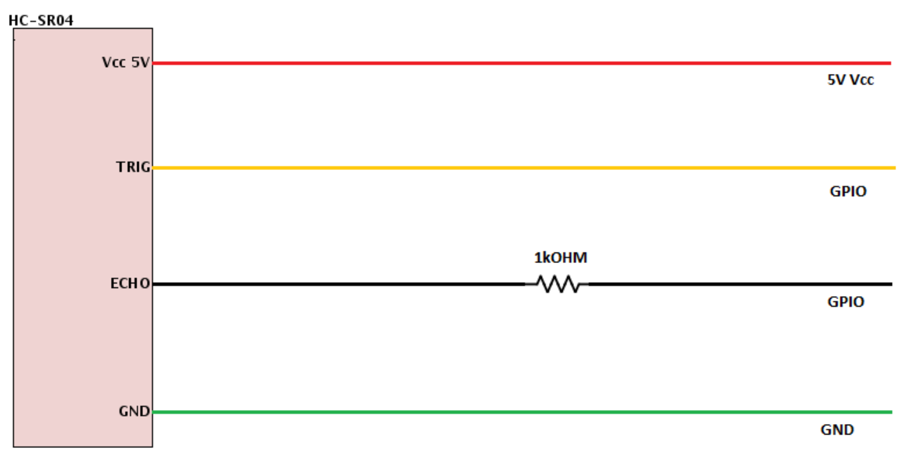
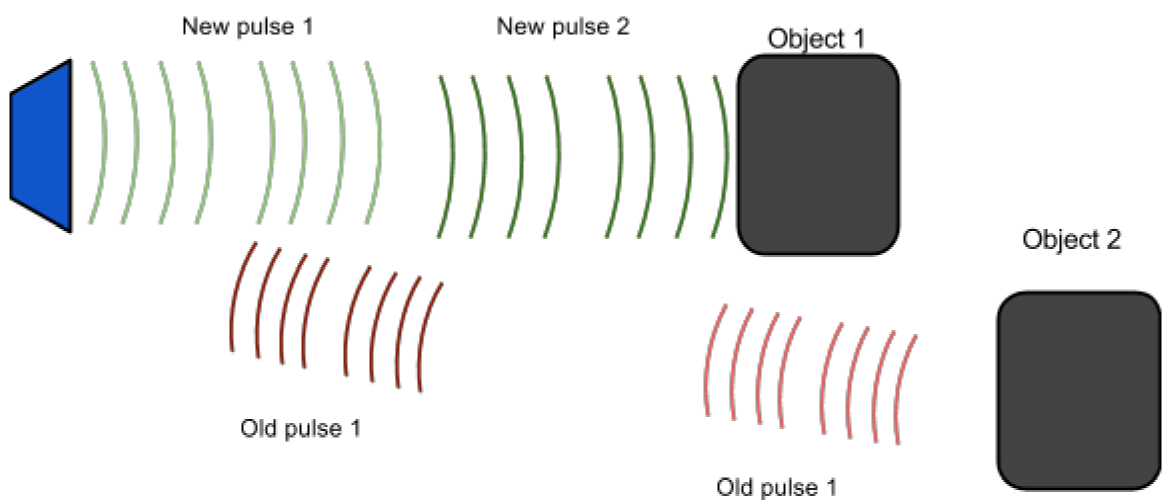


Figure 4

5. Errors and Bad Readings

Ultrasonic sensors are great sensors -- they work well for many applications where other types of sensors fall short. Unfortunately, they do have weaknesses. These weaknesses can be mitigated and worked around, but first they must be understood. The first weakness is that they use sound. There is a limit to how fast ultrasonic sensors can get distance measurements. The longer the distance, the slower they are at reporting the distance. The second weakness comes from the way sound bounces off of objects. In enclosed spaces it is possible, if not probable that there will be unintended echos. The echos can very easily cause false short readings. In Figure 2 a pulse was sent out. It bounced off of object 1 and returned to the sensor. The distance was recorded and then a new pulse was sent. There was another object farther away, so that when the new pulse reaches object 1, the first signal will reach the sensor. This will cause the sensor to think that there is an object closer than is actually true. The old pulse is smaller than the new pulse because it has grown weaker. The longer the pulse exists the weaker it grows until it is negligible. If multiple sensors are being used, the number of echos will increase along with the number of errors. There are two main ways to reduce the number of errors. The first is to provide shielding around the sensor. This prevents echos coming in from angle outside what the sensor should actually pick up. The second is to reduce the frequency at which pulses are sent out. This gives more time for the echos to dissipate.



Works Cited

Source 1.
"HC-SR04 User's_Manual." *docs.google.* Cytron Technologies, May 2013 Web. 5 Dec. 2009.
<https://docs.google.com/document/d/1Y-yZnNhMYy7rwhAgyL_pfa39RsB-x2qR4vP8saG73rE/edit>

Source 2.
"Attiny2313 Ultrasonic distance (HR-SR04) example." *CircuitDB*. n.a. 7 Sept. 2014 Web. 5 Dec. 2014. <http://www.circuitdb.com/?p=1162>

Links

These are not formatted; you will need to copy and paste them into your web browser.

Want to learn about Ultrasonic Sensors in general?
http://www.sensorsmag.com/sensors/acoustic-ultrasound/choosing-ultrasonic-sensor-proximity-or-distance-measurement-825

All about the HC-SR04

- http://www.circuitdb.com/?p=1162
- http://www.micropik.com/PDF/HCSR04.pdf
- http://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/
- http://www.ezdenki.com/ultrasonic.php
  (^fantastic tutorial, explains a lot of stuff)
- http://www.elecrow.com/hcsr04-ultrasonic-ranging-sensor-p-316.html
  (^ this one has some cool charts)

VC0706

Digital Video Processor

Datasheet

Version 1.0

September 28, 2007

Notes1: The information is subject to change without notice. Before using this document, please confirm that this is the latest version. Notes2: Not all products and/or types are available in every country. Please check with a Vimicro sales representative for availability and additional information.

Important Notice

Vimicro and its subsidiaries reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. Customers should contact Vimicro's sales department before purchasing the product described in this document. All products are sold subject to Vimicro's terms and conditions of sale supplied at the time of order acknowledgment.

Vimicro does not warrant or represent that any license, either explicit or implied, is granted under any Vimicro patent right, copyright, mask work right, or other Vimicro intellectual property right relating to any combination, machine, or process in which Vimicro products or services are used. Information published by Vimicro regarding third-party products or services does not constitute a license from Vimicro to use such products or service or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from Vimicro under the patents or other intellectual property of Vimicro.

Vimicro semiconductor devices are intended for standard uses (such as office equipment, computers, industrial/communications/measuring equipment, and personal/home equipment). Customers using semiconductor devices for special applications (including aerospace, nuclear, military and medical applications) in which a failure or malfunction might endanger life or limb and which require extremely high reliability must contact our Sales Department first. If damage is caused by such use of our semiconductor devices without first consulting our Sales Department, Vimicro will not assume any responsibility for the loss.

The contents of this document must not be reprinted or duplicated without written permission of Vimicro. Information and circuit diagrams in this document are only examples of device application. They are not intended to be used in actual equipment. Vimicro accepts no responsibility for infringement of patents or other rights owned by third parties caused by use of the information and circuit diagrams in this document.

Reproduction of information in Vimicro data books or data sheets is permissible only if preproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. Vimicro is not responsible or liable for such altered documentation. Resale of Vimicro products or services with statements different from or beyond the parameters stated by Vimicro for that product or service voids all explicit and any implied warranties for the associated Vimicro product or service and is an unfair and deceptive business practice. Vimicro is not responsible or liable for any such statements.

Table of Content

1. GENERAL DESCRIPTION

VC0706 is a high performance camera processor with enhanced image processing functions. This SOC chip has CMOS sensor interface and digital video input interface, can capture the video stream from CMOS sensor or external TV decoder, implement video enhancement, OSD overlay and motion detection, before output the digital video through the CCIR656 output interface, make VC0706 can easily to be connected to external DSP/host processor for further video processing, such as the compression and broadcast. External host processors can control the VC0706 through the flexible SPI/UART interface.

VC0706 embedded the hard-wired JPEG codec, supports up to 30fps encoding or decoding with the VGA resolution. VC0706 can compress the captured video stream to M-JPEG stream and output through the SPI/UART interface, or receive M-JPEG stream written by external host, before decoding and output.

VC0706 embedded TV encoder and video DAC, make it can output NTSC/PAL video to TV monitors or other 75ohm display devices directly.

VC0706 can be easily to be applied in security camera, automotive camera or and embedded camera require image pre-processing & M-JPEG compression.

Below are VC0706's key parameters:

PARAMETERS	VALUES	UNIT
Sensor Type Supported	VGA/CIF CMOS	
NTSC (M) output	712 x 486	Pixel
PAL (B) output	704 x 576	Pixel
Maximum Frame Rate	60 (@ 27MHz in NTSC) 50 (@ 27MHz in PAL)	fps
Maximum SPI Interface Clock Frequency	18	Mhz
Maximum HS-UART Interface BPS	921.6	kbps
Power consumption	65	mA
Crystal input Frequency	6.0	MHz
Power Supply	3.3 & 1.2	V
Operating Temperature	-40 ~ +85	ºC

VC0706 has two variations listed in the table below:

PRODUCT NUMBER	PACKAGE
VC0706PREA	100-Pin LQFP (14mm x 14mm x 1.4mm)
VC0706PTNA	144-Pin LQFP (20mm x 20mm x 1.4mm)

2. FULL FEATURE LIST

- Video Input
  - Support up to VGA CMOS sensor
  - Support CCIR656 input from external TV decoder
- Video Out
  - Digital video output
    - CCIR656 digital video output interface
  - Analog video output
    - BT.470 compatible analog composite video output: NTSC or PAL
    - Support 8-bit key protect video encryption output mode
    - Support NTSC-M/J/4.43 and PAL-/B/D/G/H/M/N/I/Nc TV video
  - Compressed video output
    - JPEG format
    - Resolution and compression ratio configurable
- Video steam frame rate control
- Image process pipeline
  - Black level correction with G1/G2 filter
  - Auto Exposure (AE)
  - Auto White Balance (AWB)
  - Auto Gain Control (AGC)
  - Auto defect pixel detection and cancellation
  - Auto lens shading compensation
  - Auto edge extraction and sharpness processing
  - Auto false color suppression
  - Auto backlight detection and wide dynamic range compensation
  - Advanced noise reduction for high image quality under low light
  - Auto UV suppression under low light
  - Configurable gamma and color correction
  - Configurable brightness, contrast, hue, saturation control
  - Edge-adaptive CFA interpolation
  - Up to 16-window for statistical collection, exposure
  - Color/BW video image auto/manual conversion
- Motion detection
  - 4x4 window, sensitivity programmable
  - Support both horizontal and vertical calculation
  - The window size, location can be programmable
  - The motion detection function can be enabled and disable by window
  - The threshold of each motion detection window can be set separately
  - Slight illumination changes alarm cancellation when no motion
  - Support separate motion detection threshold to adjust post process frame rate

- Flexible image scalar
  - ✧ Support 1:1, 2:1, 4:1 horizontally and vertically down-scalar of input image
  - ✧ Support scale for D1 resolution output
    - ■ Arbitrary ratio scale-up
    - ■ Maximum 2 times scale-down
- On Screen Display
  - ✧ Up to four OSD channels supported with separate control
    - ■ One character channel support up to 4 lines
    - ■ One static graphic channel for the usage of user programmable logo
    - ■ One mosaic channel for the usage of private area protection
    - ■ One bitmap channel support real time display the bitmap
  - ✧ Rich control features for each channel
  - ✧ Built-in character library with totally 80 characters
- Build-in 8-bit MCU
  - ✧ High performance MCU core with rich peripherals
  - ✧ On-chip SRAM & program code ROM
  - ✧ Support disable mask ROM and run program code from external flash
    - ■ Separate code memory and data memory chip select signal
    - ■ Separate address bus and data bus
    - ■ Support the encryption of external ROM code to protect customers intellectual property
    - ■ Address space extension via page mode
- High speed SPI Interface
  - ✧ Master/Slave mode selectable
  - ✧ Programmable bit clock supported up to 18MHz
  - ✧ Support both host and DMA work mode
  - ✧ DMA channel between OSD bitmap channel and SPI
  - ✧ Bi-direction DMA channel between FBUF and SPI
- High speed UART Interface
  - ✧ Programmable bit rate supported up to 921.6 kbps
  - ✧ Bi-direction DMA channel between FBUF and UART
- Flexible GPIO function configuration mechanism support rich features
  - ✧ Backlight compensation enable
  - ✧ Mirror control
  - ✧ AE indoor/outdoor switch
  - ✧ Motion enable
  - ✧ Motion output
  - ✧ Color or Black & White output selection
- One input clock: 6MHz crystal
- Configurable I2C EEPROM, SPI EEPROM/FLASH

3. CHIP BLOCK DIAGRAM

VC0706 integrates SIF (sensor interface), ISP (image signal processor), frame rate control unit, Video Enhancement Engine, JPEG codec, Motion Detection Engine, IPP (image post-processor), OSD unit, TV encoder, Video DAC and 8051 compatible MCU with built-in code-ROM and data-RAM.

The CCIR656 output interface enable VC0706 output enhanced digital video stream to external DSP or host CPU for compression or broadcast.

The bi-direction DMA path between FBUF and SPI/HS-UART enable the external processor read out the captured JPEG image or write JPEG pictures for display.

With the DMA channel between SPI interface and OSD unit, VC0706 can be easily used to implement static or dynamic bitmap overlay with real time video data from sensor.

The behaviors of VC0706 can be programmed through I2C EEPROM, SPI EEPROM/FLASH , and external host processor . Multiple GPIO channels support typical control functions.

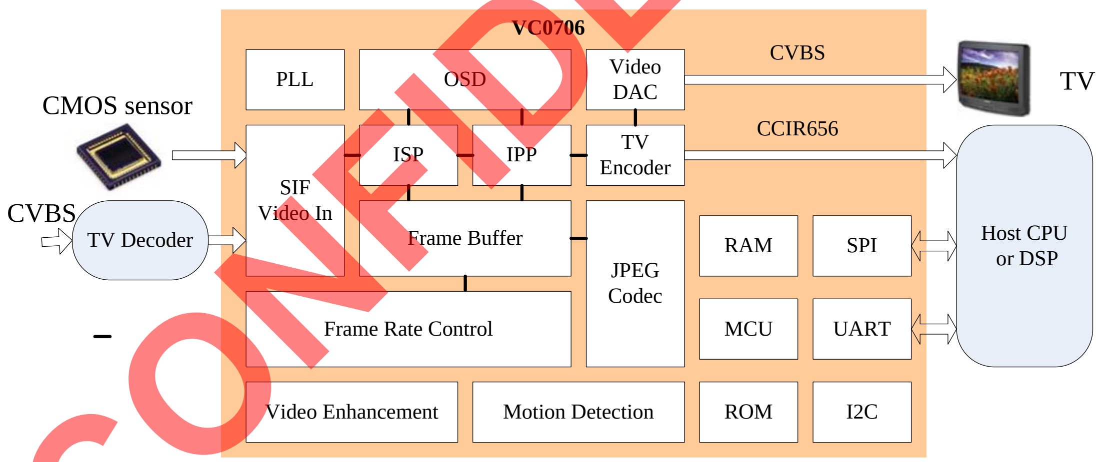


Figure 1: VC0706 Chip Block Diagram

4. PIN ASSIGNMENTS

4.1 VC0706PREA Pin Assignments

PIN NUMBER VC0706PRSA	NAME	TYPE	DESCRIPTION
1	CS_SDA	I/O, OD , S	Sensor I2C bus SDA pin
2	GPIO13	I/O, PD	GPIO 13
3	CS_SCK	I/O, OD , S	Sensor I2C bus SCK pin
4	GPIO12	I/O , PD	GPIO 12
5	CS_HSYNC	I, PD	Sensor Horizontal sync input
6	MCU_WEN	O	MCU NOR flash interface write enable signal "0" indicate write operation; "1" indicate read operation
7	CS_VSYNC	I, PD	Sensor vertical sync input
8	MCU_A15	O	MCU NOR flash interface address bit 15
9	MCU_A14	O	MCU NOR flash interface address bit 14
10	CS_PCLK	I, PD	Sensor pixel clock input
11	CS_CLK	O	Sensor CLK output
12	GPIO9	I/O , PD	GPIO 9
	OSD_RDY	O	Ready to accept new OSD data from host through SPI interface
13	CS_D0	I, PD	Sensor data input bit 0
	CS_PCLK	I, PD	Sensor pixel clock input
14	MCU_DCEN	O	MCU NOR flash interface data memory chip select , "0" valid
15	CS_D1	I, PD	Sensor data input bit 1
16	CS_D2	I, PD	Sensor data input bit 2
17	MCU_A13	O	MCU NOR flash interface address bit 13
18	MCU_A12	O	MCU NOR flash interface address bit 12
19	CS_D3	I, PD	Sensor data input bit 3
20	CS_D4	I, PD	Sensor data input bit 4
21	MCU_A11	O	MCU NOR flash interface address bit 11
22	MCU_A10	O	MCU NOR flash interface address bit 10
23	CS_D5	I, PD	Sensor data input bit 5

PIN NUMBER			
24	CS_D6	I, PD	Sensor data input bit 6
25	MCU_A9	O	MCU NOR flash interface address bit 9
26	GPIO8	I/O, PD	GPIO 8
	OSD_STOP	I, PD	Host asserted signal to stop OSD DMA
27	MCU_A8	O	MCU NOR flash interface address bit 8
28	CS_D7	I, PD	Sensor data input bit 7
29	CS_D8	I, PD	Sensor data input bit 8
30	CS_D9	I, PD	Sensor data input bit 9
31	MCU_A7	O	MCU NOR flash interface address bit 7
32	MCU_A6	O	MCU NOR flash interface address bit 6
33	CS_PWD	O	CMOS Sensor Power Down Control
	GPIO5	O	GPIO 5 , only can be used as output mode
	TV_HSYNC_O	O	TV encoder HSYNC output
34	CS_RSTB	O	Sensor Reset Output
	TV_MOD	I, PD	Hardware strap pin : TV_MOD
35	MCU_A5	O	MCU NOR flash interface address bit 5
36	MCU_A4	O	MCU NOR flash interface address bit 4
37	GPIO7	I/O , PD	GPIO 7
38	RSTN	I , PU	Power on reset
39	GPIO4	I/O, PD	GPIO 4
	TV_VSYNC_O	O	TV Vertical SYNC Output
	TV_CSYNC_O	O	TV Composite SYNC Output
40	GPIO11	I/O , PD	GPIO 11
41	MCU_A3	O	MCU NOR flash interface address bit 3
42	MCU_A2	O	MCU NOR flash interface address bit 2
43	MCU_A1	O	MCU NOR flash interface address bit 1
	OCODE_MOD	I , PD	Hardware strap pin : OCODE_MOD
44	MCU_A0	O	MCU NOR flash interface address bit 0
	OROM_MOD	I , PD	Hardware strap pin : OROM_MOD
45	VDD_IO	P	Digital IO Power 3.3V
46	VSS	P	Digital IO/core ground
47	VDD	P	Digital core power supply 1.2V
48	VSSA	P	Analog ground

PIN NUMBER			
49	MONA	A	Test signal , not connect
50	VDDA	P	Analog power supply (3.3v)
51	VLX	A	Test signal , not connect
52	VSSA	P	Analog ground
53	DAC_REXT	A	DAC external resistor pin
54	DAC_COMP	A	DAC Compensation pin
55	DAC_CVBS	A, O	CVBS channel signal analog output
56	VSSA_DAC	P	DAC analog ground
57	VDDA_DAC	P	DAC analog Power (3.3V)
58	VSSA_PLL	P	PLL Analog ground
59	VDDA_PLL	P	PLL Analog power (3.3V)
60	TV_D7	O	CCIR656 digital video output bit 7
61	TV_D6	O	CCIR656 digital video output bit 6
62	TV_D5	O	CCIR656 digital video output bit 5
63	TV_D4	O	CCIR656 digital video output bit 4
64	GPIO6	I/O , PD	GPIO 6
	TV_ODD_O	O	TV encoder ODD signal output , "Low" indicate even field , "High" indicate odd field
65	TV_D3	O	CCIR656 digital video output bit 3
66	TV_D2	O	CCIR656 digital video output bit 2
67	GPIO5	O	GPIO 5 , only can be used as output mode
	TV_HSYNC_O	O	TV encoder HSYNC output
	BOOT_MOD	I, PD	Hardware strap pin : BOOT_MOD
68	TV_D1	O	CCIR656 digital video output bit 1
69	TV_D0	O	CCIR656 digital video output bit 0
70	CLK_IN	I	Crystal input
71	CLK_OUT	O	Crystal output (loopback)
72	VSS	P	Digital IO/CORE ground
73	VDD_IO	P	Digital IO Power (3.3v)
74	TV_PCLK	O	Digital video pixel clock output
75	TEST	I, PD	Manufacture test mode, connect to GND
76	GPIO0	I/O, PD	GPIO 0
77	GPIO1	I/O, PD	GPIO 1
78	MCU_D0	I/O , PD	MCU NOR flash interface data bus bit 0

PIN NUMBER			
79	MCU_D1	I/O , PD	MCU NOR flash interface data bus bit 1
80	UART_TXD	O	High speed UART data output
	GPIO14	O	GPIO 14 , only can be used as output mode
	SPI_MOD	I, PD	Hardware strap pin : SPI_MOD
81	UART_RXD	I, PD	High speed UART data input
	GPIO15	I/O , PD	GPIO 15
82	MCU_D2	I/O , PD	MCU NOR flash interface data bus bit 2
83	MCU_D3	I/O , PD	MCU NOR flash interface data bus bit 3
84	GPIO2	I/O, PD	GPIO 2
85	GPIO3	I/O, PD	GPIO 3
	TV_CLK_O	O	TV 27MHZ CLK output
86	MCU_D4	I/O , PD	MCU NOR flash interface data bus bit 4
87	MCU_D5	I/O , PD	MCU NOR flash interface data bus bit 5
88	GPIO10	I/O , PD	GPIO 10
89	SPI_SS	I/O, PU	SPI master interface slave select output
	GPIO10	I/O, PU	GPIO 10
90	SPI_SCK	I/O, PD	SPI master interface serial clock output
	GPIO11	I/O, PD	GPIO 11
91	SPI_MOSI	I/O, PD	SPI master interface master output
	GPIO12	I/O, PD	GPIO 12
92	SPI_MISO	I/O, PD	SPI master interface slave input
	GPIO13	I/O, PD	GPIO 13
93	MCU_D6	I/O, PD	MCU NOR flash interface data bus bit 6
94	MCU_D7	I/O, PD	MCU NOR flash interface data bus bit 7
95	CS_ENB	O	Sensor enable pin, Output
	SIF_EDG	I, PD	Hardware strap pin: SIF_EDG
96	MCU_OEN	O	MCU NOR flash interface external memory output enable , "0" valid
97	MCU_PCEN	O	MCU NOR flash interface code memory chip select , "0" valid
98	VDD	P	Digital core Power 1.2V
99	VSS	P	Digital IO/core ground
100	VDD_IO	P	Digital IO Power 3.3V

4.2 VC0706PTNA Pin Assignments

PIN NUMBER VC0706PTNA	NAME	TYPE	DESCRIPTION
1	NC		No connect
2	NC		No connect
3	NC		No connect
4	CS_SDA	I/O, OD , S	Sensor I2C bus SDA pin
5	GPIO13	I/O, PD	GPIO 13
6	CS_SCK	I/O, OD , S	Sensor I2C bus SCK pin
7	GPIO12	I/O , PD	GPIO 12
8	CS_HSYNC	I, PD	Sensor Horizontal sync input
9	MCU_WEN	O	MCU NOR flash interface write enable signal "0" indicate write operation; "1" indicate read operation
10	CS_VSYNC	I, PD	Sensor vertical sync input
11	MCU_A15	O	MCU NOR flash interface address bit 15
12	MCU_A14	O	MCU NOR flash interface address bit 14
13	VDD	P	Digital Core Power, 1.2v
14	CS_PCLK	I, PD	Sensor pixel clock input
15	VSS	P	Digital IO/Core ground
16	CS_CLK	O	Sensor CLK output
17	VDD_IO	P	Digital IO power, 3.3v
18	GPIO9	I/O , PD	GPIO 9
	OSD_RDY	O	Ready to accept new OSD data from host through SPI interface
19	CS_D0	I, PD	Sensor data input bit 0
	CS_PCLK	I, PD	Sensor pixel clock input
20	MCU_DCEN	O	MCU NOR flash interface data memory chip select , "0" valid
21	CS_D1	I, PD	Sensor data input bit 1
22	CS_D2	I, PD	Sensor data input bit 2
23	MCU_A13	O	MCU NOR flash interface address bit 13
24	MCU_A12	O	MCU NOR flash interface address bit 12
25	VDD_IO	P	Digital IO power, 3.3v

PIN NUMBER			
26	VSS	P	Digital IO/core ground
27	VDD	P	Digital core power, 1.2v
28	CS_D3	I, PD	Sensor data input bit 3
29	CS_D4	I, PD	Sensor data input bit 4
30	MCU_A11	O	MCU NOR flash interface address bit 11
31	MCU_A10	O	MCU NOR flash interface address bit 10
32	CS_D5	I, PD	Sensor data input bit 5
33	CS_D6	I, PD	Sensor data input bit 6
34	MCU_A9	O	MCU NOR flash interface address bit 9
35	NC		Not connect
36	NC		Not connect
37	GPIO8	I/O, PD	GPIO 8
	OSD_STOP	I, PD	Host asserted signal to stop OSD DMA
38	MCU_A8	O	MCU NOR flash interface address bit 8
39	CS_D7	I, PD	Sensor data input bit 7
40	CS_D8	I, PD	Sensor data input bit 8
41	CS_D9	I, PD	Sensor data input bit 9
42	VDD_IO	P	Digital IO power , 3.3v
43	MCU_A7	O	MCU NOR flash interface address bit 7
44	VSS	P	Digital IO/Core ground
45	MCU_A6	O	MCU NOR flash interface address bit 6
46	VDD	P	Digital core power , 1.2v
47	CS_PWD	O	CMOS Sensor Power Down Control
	GPIO5	O	GPIO 5 , only can be used as output mode
	TV_HSYNC_O	O	TV encoder HSYNC output
48	CS_RSTB	O	Sensor Reset Output
	TV_MOD	I, PD	Hardware strap pin : TV_MOD
49	MCU_A5	O	MCU NOR flash interface address bit 5
50	MCU_A4	O	MCU NOR flash interface address bit 4
51	GPIO7	I/O , PD	GPIO 7
52	RSTN	I , PU	Power on reset
53	GPIO4	I/O, PD	GPIO 4
	TV_VSYNC_O	O	TV Vertical SYNC Output

PIN NUMBER			
	TV_CSYNC_O	O	TV Composite SYNC Output
54	GPIO11	I/O , PD	GPIO 11
55	MCU_A3	O	MCU NOR flash interface address bit 3
56	MCU_A2	O	MCU NOR flash interface address bit 2
57	MCU_A1	O	MCU NOR flash interface address bit 1
	OCODE_MOD	I , PD	Hardware strap pin : OCODE_MOD
58	MCU_A0	O	MCU NOR flash interface address bit 0
	OROM_MOD	I , PD	Hardware strap pin : OROM_MOD
59	VDD_IO	P	Digital IO Power 3.3V
60	VSS	P	Digital IO/core ground
61	VDD	P	Digital core power , 1.2v
62	VFB	A	Test signal , not connect
63	VOUT	A	Test signal , not connect
64	VSSA	P	Analog ground
65	VSSA	P	Analog ground
66	VSSA	P	Analog ground
67	MONA	A	Test signal , not connect
68	VDDA	P	Analog power supply ,3.3V
69	VDDA	P	Analog power supply ,3.3v
70	VDDA	P	Analog power supply ,3.3v
71	ONKEYZ	A	Test pin, need connect to VDDA
72	NC		No connect
73	VLX	A	Test signal , not connect
74	VLX	A	Test signal , not connect
75	VSSA	P	Analog ground
76	VSSA	P	Analog ground
77	ONKEYZ	A	Test pin, need connect to VDDA
78	DAC_REXT	A	DAC external resistor pin
79	DAC_COMP	A	DAC Compensation pin
80	DAC_CVBS	A, O	CVBS channel signal analog output
81	VSSA_SUB	P	DAC pad ground
82	VSSA_DAC	P	DAC analog ground
83	VDDA_DAC	P	DAC analog power ,3.3v
84	VDDA_WELL	P	DAC pad Power ,3.3V

PIN NUMBER			
85	VSS_PLL	P	Digital core ground for DAC/PLL
86	VDD_PLL	P	Digital core power for DAC/PLL ,1.2v
87	VSSA_PLL	P	PLL Analog ground
88	VDDA_PLL	P	PLL Analog power ,3.3V
89	TV_D7	O	CCIR656 digital video output bit 7
90	VDD	P	Digital core power ,1.2v
91	TV_D6	O	CCIR656 digital video output bit 6
92	VSS	P	Digital IO/core ground
93	TV_D5	O	CCIR656 digital video output bit 5
94	TV_D4	O	CCIR656 digital video output bit 4
95	GPIO6	I/O , PD	GPIO 6
	TV_ODD_O	O	TV encoder ODD signal output , "Low" indicate even field , "High" indicate odd field
96	TV_D3	O	CCIR656 digital video output bit 3
97	TV_D2	O	CCIR656 digital video output bit 2
98	GPIO5	O	GPIO 5 , only can be used as output mode
	TV_HSYNC_O	O	TV encoder HSYNC output
	BOOT_MOD	I, PD	Hardware strap pin : BOOT_MOD
99	TV_D1	O	CCIR656 digital video output bit 1
100	TV_D0	O	CCIR656 digital video output bit 0
101	TV_VSYNC	O	TV vsync output
102	TV_HSYNC	O	TV hsync output
103	VDD	P	Digital core power , 1.2v
104	CLK_IN	I	Crystal input
105	CLK_OUT	O	Crystal output (loopback)
106	VSS	P	Digital IO/CORE ground
107	VDD_IO	P	Digital IO Power ,3.3v
108	TV_PCLK	O	Digital video pixel clock output
109	TEST	I, PD	Manufacture test mode, connect to GND
110	NC		Not connect
111	NC		Not connect
112	NC		Not connect
113	GPIO0	I/O, PD	GPIO 0
114	VDD_IO	P	Digital IO power, 3.3v

PIN NUMBER			
115	GPIO1	I/O, PD	GPIO 1
116	MCU_D0	I/O , PD	MCU NOR flash interface data bus bit 0
117	VSS	P	Digital IO/core ground
118	MCU_D1	I/O , PD	MCU NOR flash interface data bus bit 1
119	VDD	P	Digital core power, 1.2v
120	UART_TXD	O	High speed UART data output
	GPIO14	O	GPIO 14 , only can be used as output mode
	SPI_MOD	I, PD	Hardware strap pin : SPI_MOD
121	UART_RXD	I, PD	High speed UART data input
	GPIO15	I/O , PD	GPIO 15
122	MCU_D2	I/O , PD	MCU NOR flash interface data bus bit 2
123	MCU_D3	I/O , PD	MCU NOR flash interface data bus bit 3
124	GPIO2	I/O, PD	GPIO 2
125	GPIO3	I/O, PD	GPIO 3
	TV_CLK_O	O	TV 27MHZ CLK output
126	MCU_D4	I/O , PD	MCU NOR flash interface data bus bit 4
127	MCU_D5	I/O , PD	MCU NOR flash interface data bus bit 5
128	GPIO10	I/O , PD	GPIO 10
129	SPI_SS	I/O, PU	SPI master interface slave select output
	GPIO10	I/O, PU	GPIO 10
130	VDD	P	Digital core power , 1.2v
131	SPI_SCK	I/O, PD	SPI master interface serial clock output
	GPIO11	I/O, PD	GPIO 11
132	VSS	P	Digital IO/core ground
133	SPI_MOSI	I/O, PD	SPI master interface master output
	GPIO12	I/O, PD	GPIO 12
134	VDD_IO	P	Digital IO power , 3.3v
135	SPI_MISO	I/O, PD	SPI master interface slave input
	GPIO13	I/O, PD	GPIO 13
136	MCU_D6	I/O, PD	MCU NOR flash interface data bus bit 6
137	MCU_D7	I/O, PD	MCU NOR flash interface data bus bit 7
138	CS_ENB	O	Sensor enable pin, Output
	SIF_EDG	I, PD	Hardware strap pin: SIF_EDG

PIN NUMBER			
139	MCU_OEN	O	MCU NOR flash interface external memory output enable , “0” valid
140	MCU_PCEN	O	MCU NOR flash interface code memory chip select , “0” valid
141	VDD	P	Digital core Power 1.2V
142	VSS	P	Digital IO/core ground
143	LDOM	I	Test pin, connect to GND
144	VDD_IO	P	Digital IO Power 3.3V

Note: I/O – Bidirectional Input/Output
I – Input
O – Output
P – Power
G – Ground
A – Analog
PD – Internal Pull Up
PU – Internal Pull Down
OD – Open Drain
S – Schmitt Trigger
A – Analog Links (to a resistor, capacitor etc.)

4.3 Hardware Strap Pin Descriptions

VC0706 has hardware strap pins. Those pins are multiplex usage pins. When the VC0706 had been reset, the pull-up or pull-down status of these pins will set the configuration of VC0706.

The table below lists the hardware strap pins and their functions:

In the table, 0 means pull-down, 1 means pull-up, the on-chip pull-down resisters will select the default setting if no external resistors used to pull-up the hardware strap pins.

SIGNAL NAME	DESCRIPTION
TV_MOD	Used for the selection of TV 525 lines or 625 lines output mode 0 (default): 525 lines, NTSC TV system 1: 625 lines, PAL TV system
SIF_EDGE	Used for the selection of the clock edge for sensor-interface sampling sensor’s input data 0(default): use default setting for sampling 1: use reverse edge for sampling
BOOT_MOD	Used for the selection of host boot enable or disable

	0(default): Host boot disable 1 : Host boot enable
SPI_MOD	Used for the selection of SPI interface work in master mode or slave mode 0(default) : SPI interface work in master mode 1 : SPI interface work in slave mode
OROM_MOD	Used for the selection of boot from internal ROM or external ROM 0(default) : MCU boot from internal ROM 1 : MCU boot from external ROM
OCODE_MOD	Used for the selection of external code ROM working under normal mode or encryption mode 0(default) : Normal mode 1 : Encryption mode

4.4 VC0706PREA (100-Pin LQFP) Pin Layout Diagram

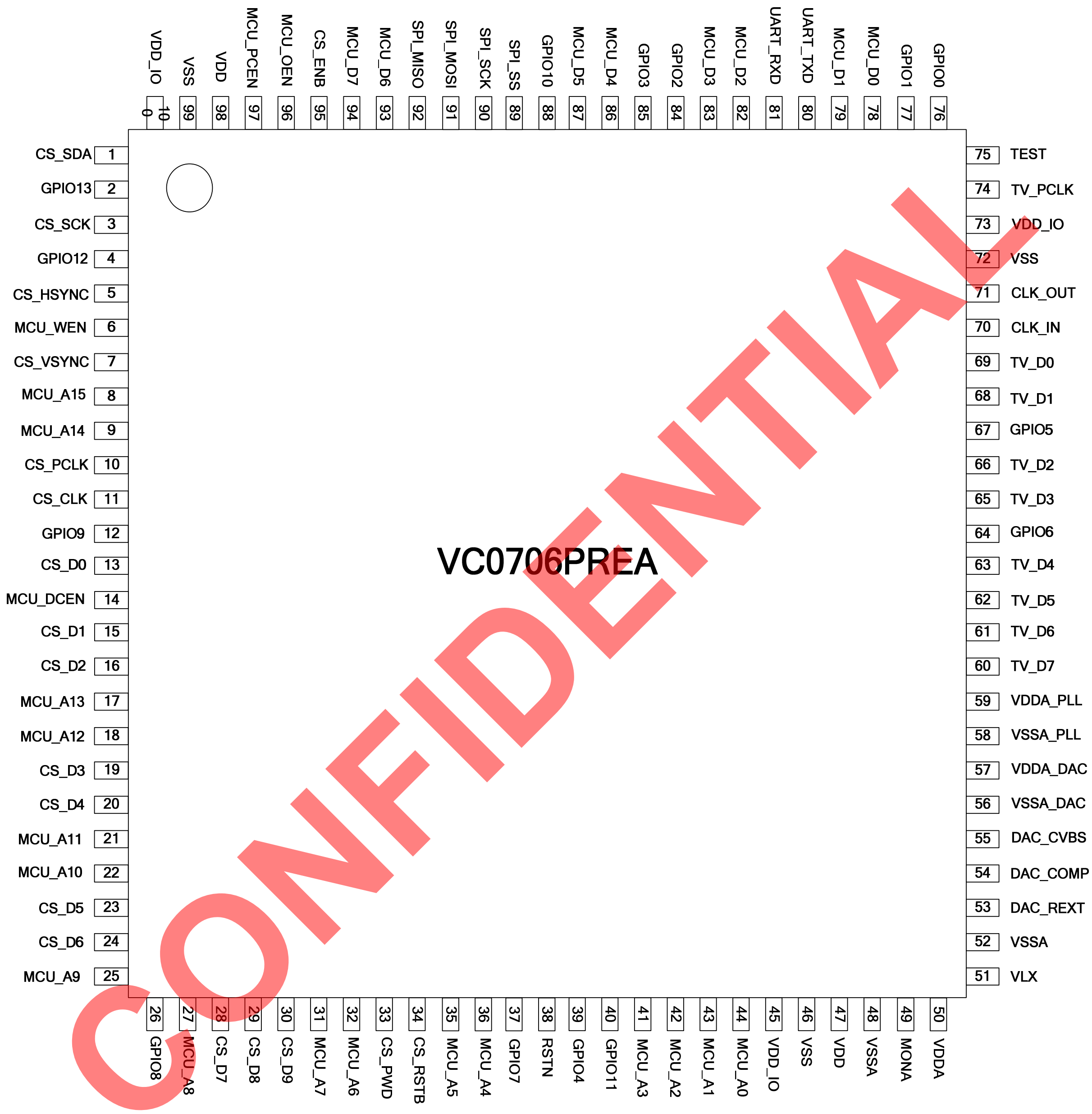


Figure 2: VC0706PREA Pin Layout Diagram

4.5 VC0706PTNA (144-Pin LQFP) Pin Layout Diagram

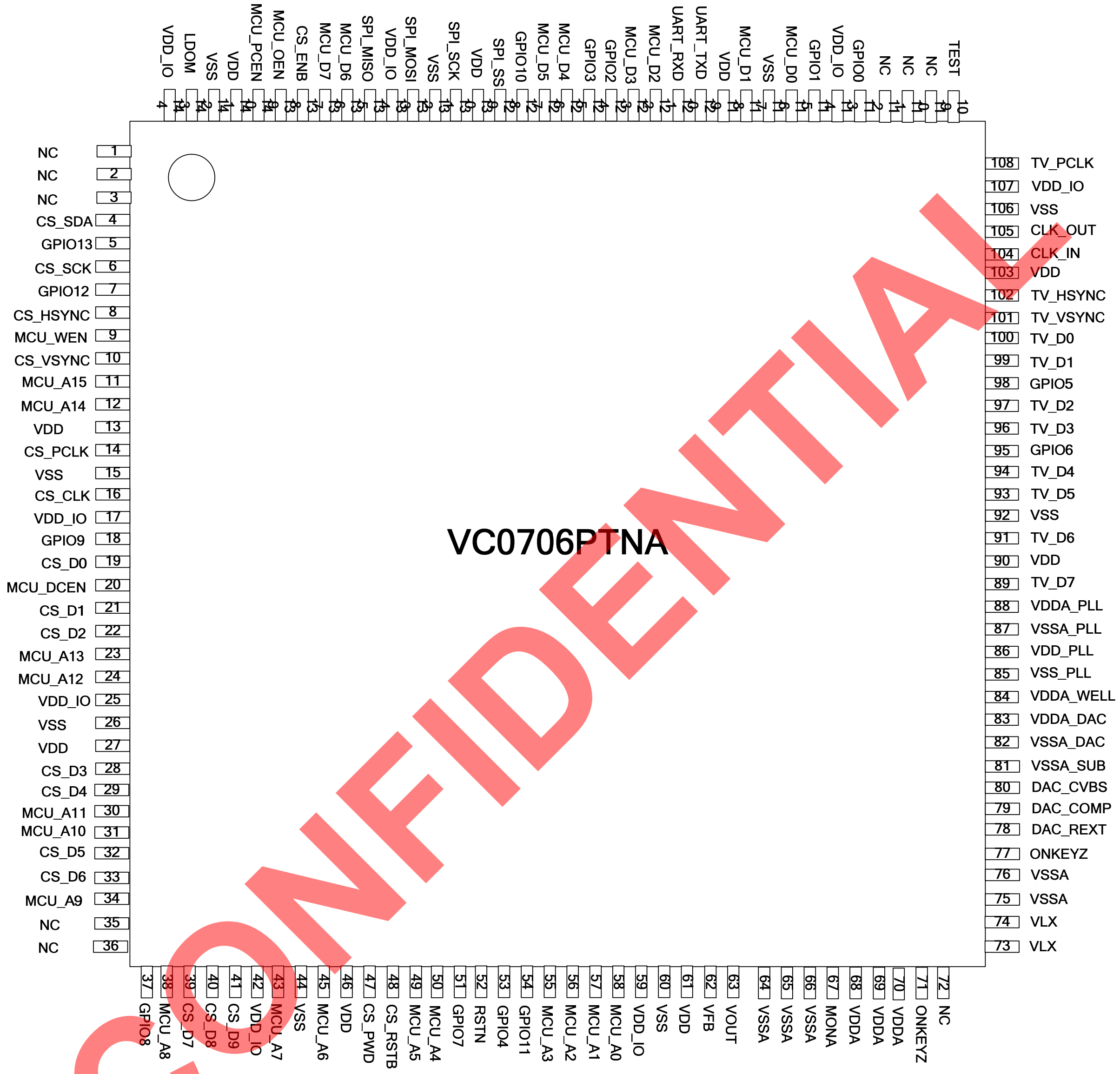


Figure 3: VC0706PTNA Pin Layout Diagram

5. ELECTRICAL CHARACTERISTICS

5.1. Recommended Operating Conditions

The recommended operating conditions are the recommended values to assure normal logic operation. As long as the device is used within the recommended operating conditions, the electrical characteristics (DC and AC characteristics) described below are assured.

SYMBOL	DESCRIPTION	MIN	TYP	MAX	UNIT
VDD_IO	I/O Digital Voltage (3.3V)	2.97	3.3	3.63	V
VDD_CORE	Core Digital Voltage	1.08	1.2	1.32	V
VDDA_DAC	Video DAC Voltage	2.97	3.3	3.63	V
VDDA_PLL	PLL Voltage	2.97	3.3	3.63	V
VDDA	Analog Voltage	2.97	3.3	3.63	V
T_O	Operating Temperature	-40	-	+85	ºC
T_S	Storage Temperature	-40	-	+125	ºC

5.2. Digital I/O Electrical Characteristics

For 3.3V I/O Application

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input low voltage	V_{IL}	-0.3	-	0.8	V
Input high voltage	V_{IH}	2.0	-	VDD_IO+0.3	V
Threshold point	V_T	1.30	1.41	1.53	V
Schmitt trig low to high threshold point	V_{T+}	1.54	1.65	1.74	V
Schmitt trig high to low threshold point	V_{T-}	0.95	1.02	1.09	V
Input leakage current	I_L	-	-	± 1	μA
Input pull-up resistor	R_{PU}	62	77	112	kΩ

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input pull-down resistor	R_{PD}	58	81	156	kΩ
Output low voltage	V_{OL}	-	-	0.4	V
Output high voltage	V_{OH}	2.4	-	-	V
Low level output current @ V_{OL}=0.4V	I_{OL}	4.2	6.6	8.1	mA
High level output current @ V_{OH}=2.4V	I_{OH}	4.7	9.6	14.9	mA

5.3. Video DAC Electrical Characteristics

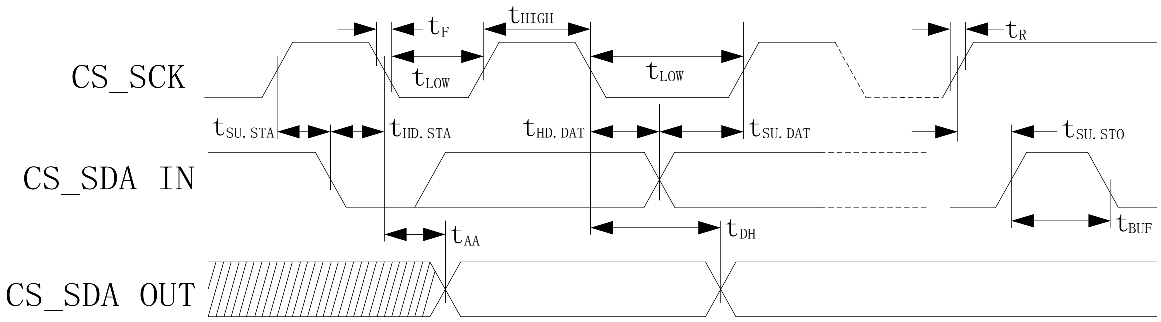
(VDDA_DAC = 3.3V , RL=37.5ohm , CL=10pF , VREFIN=1.20V; TEMP = 25°c)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Max per-channel output current	Ifs	-	34.8	-	mA
Max output voltage	Vfs	-	1.304	-	V
DAC resolution	-	-	10	-	bits
Integral non-linearity error	ERRinl	-	±1	±1.5	LSB
Differential non-linearity error	ERRdnl	-	±0.5	±1	LSB

5.4. AC Electrical Characteristics

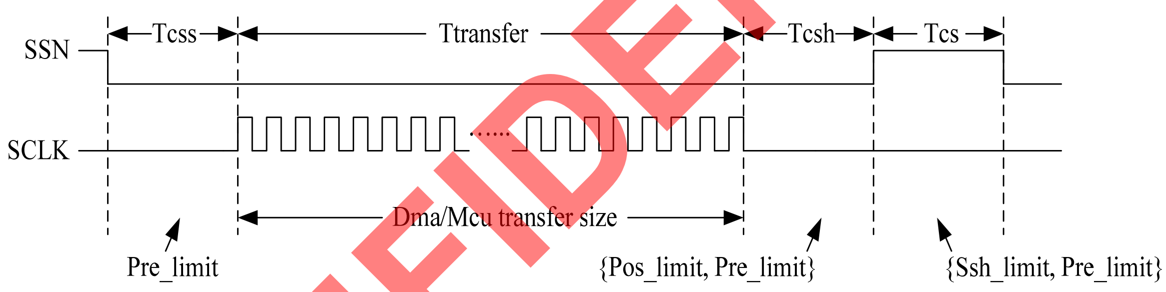
This section describes the AC characteristics of VC0706, including output delays, input setup, hold times, and all the interface timing.

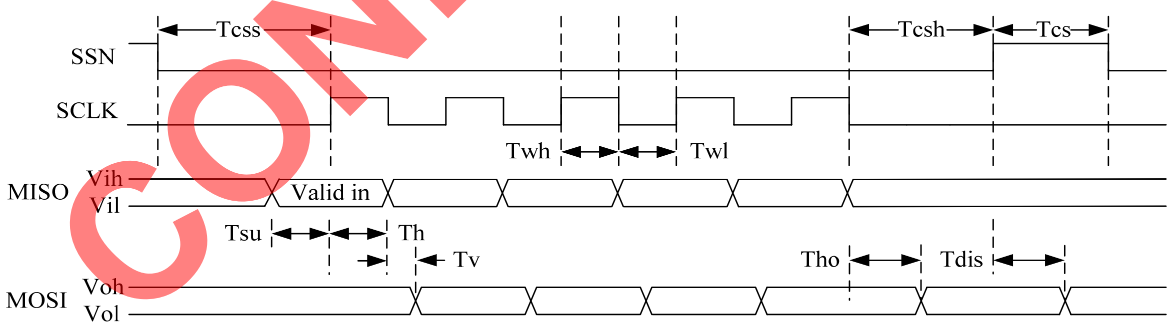
5.4.1. I^2C Interface Timing Specification



PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
CS_SCK clock frequency	f_{SCK}			400	kHz
Bus free time between a STOP and a START	t_{BUF}	4.7		-	us
Hold time for a START	$t_{HD;STA}$	4.0		-	us
Low period of CS_SCK	t_{LOW}	4.7		-	us
High period of CS_SCK	t_{HIGH}	4.0		-	us
Setup time for START	$t_{SU;STA}$	4.7		-	us
Data hold time	$t_{HD;DAT}$	0		-	us
Data setup time	$t_{SU;DAT}$	200		-	ns
Rise time of CS_SDA and CS_SCK	t_R	-		1	us
Fall time of CS_SDA and CS_SCK	t_F	-		300	ns
CS_SCK low to data out valid	t_{AA}	0.625			us
Data out hold time	t_{DH}	0.625			us
Setup time for STOP	$t_{SU;STO}$	4.7		-	us

5.4.2. SPI Master Interface Timing Specification

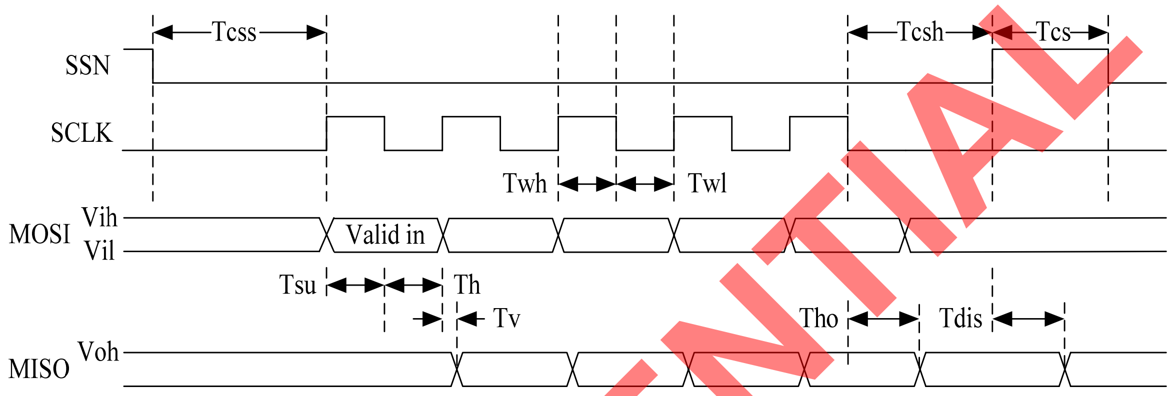




SYMBOL	PARAMETER	MIN	TYP	MAX	UNITS
Fsclk	SCLK clock frequency	0	-	18	MHz
Twh	SCLK high time	25	-	-	ns
Twl	SCLK low time	25	-	-	ns
Tcs	SSN high time	56	-	-	ns
Tcss	SSN setup time	56	-	-	ns
Tcsh	SSN hold time	56	-	-	ns

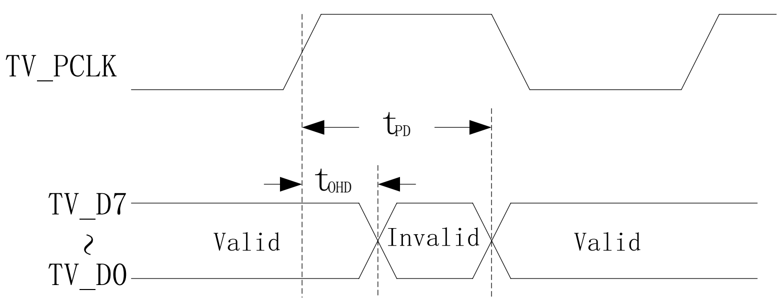
Tsu	Data in setup time	3	-	-	ns
Th	Data in hold time	3	-	-	ns
Tv	Output valid	-	-	3	ns
Tho	Output hold time	0	-	-	ns
Tdis	Output disable time	-	-	56	ns

5.4.3. SPI Slave Interface Timing Specification



SYMBOL	PARAMETER	MIN	TYP	MAX	UNITS
Fsclk	SCLK clock frequency	0	-	17	MHz
Twh	SCLK high time	28	-	-	ns
Twl	SCLK low time	28	-	-	ns
Tcs	SSN high time	56	-	-	ns
Tcss	SSN setup time	112	-	-	ns
Tcsh	SSN hold time	56	-	-	ns
Tsu	Data in setup time	3	-	-	ns
Th	Data in hold time	3	-	-	ns
Tv	Output valid	-	-	10	ns
Tho	Output hold time	32	-	-	ns
Tdis	Output disable time	-	-	56	ns

5.4.4. CCIR656 Timing Characteristics



Symbol	Parameter	Condition	Min.	Typ.	Max.	Unit
C_L	output load capacitance		15	---	40	pF
t_{OHD}	output hold time	C_L = 15pF	1.3	---	---	ns
t_{PD}	propagation delay	C_L = 25pF	---	---	7.9	ns

	Field Line Number		Active Line Number	
	525	625	525	625
Field 1 (F = 0)	4 ~ 265	1 ~312	20 ~ 263	23 ~ 310
Field 2 (F = 1)	266 ~ 3	313 ~ 625	283 ~ 525	336 ~ 623

Note: 1. When frame has 486 active lines for 525 system, line 263 is black.

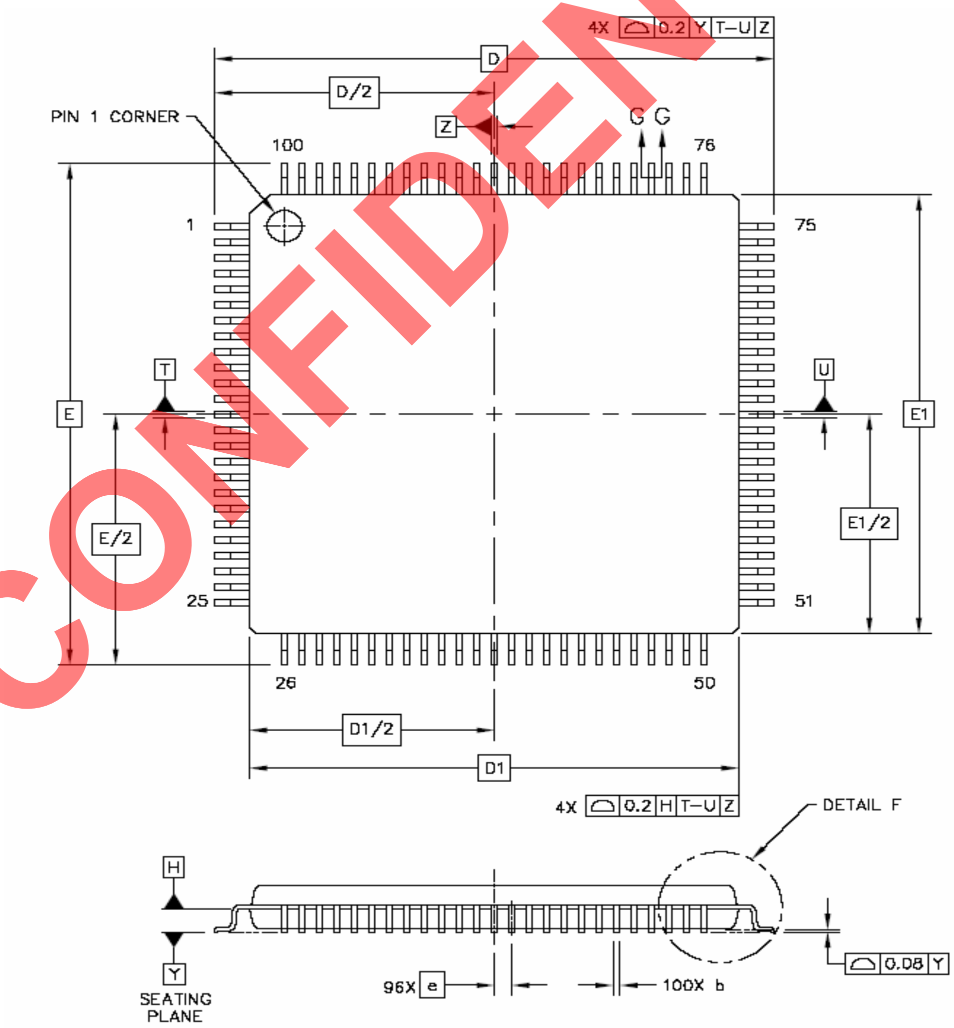
2. When frame has 480 active lines for 525 system, field 1 line 20, 21, 22, 263 are and field 2 line 283, 284, 285 are black.

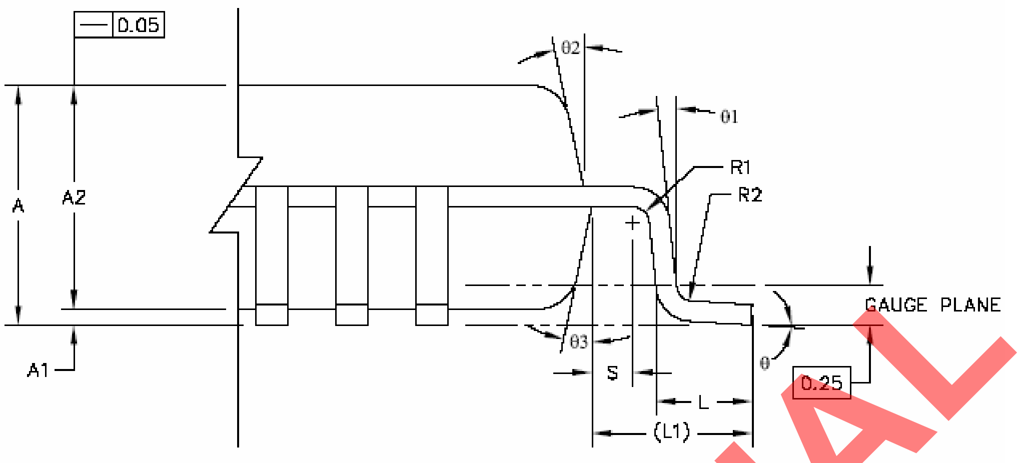
6. PACKAGE INFORMATION

6.1. Chip Marking Information

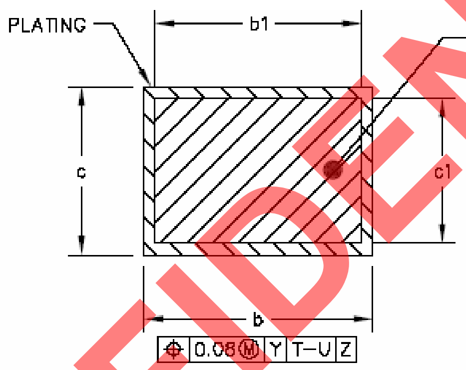
MARKING INFORMATION	LEAD-FREE PACKAGE
	Vimicro VC0706PREA Tracking No.
	Vimicro VC0706PTNA Tracking No.

6.2. VC0706PREA Package Specification (14mm x 14mm)





DETAIL F
SCALE: 25/1



SECTION G−G
SCALE: 180/1

PACKAGE	LQFP 100L 14*14* 0.9 – 0.5		
DIM	MIN.	NOM.	MAX.
A	---	---	1.6
A1	0.05	---	0.15
A2	1.35	1.4	1.45
b	0.17	0.2	0.27
b1	0.17		0.23
c	0.09		0.2
c1	0.09		0.16
D	16 BSC		
D1	14 BSC		
e	0.5 BSC		
E	16 BSC		
E1	14 BSC		
L	0.45	0.6	0.75
L1	1 REF		
R1	0.08		---
R2	0.08		0.2

S	0.2		---
θ	0	3.5	7
θ1	0		---
θ2	11	12	13
θ3	11	12	13

NOTES:

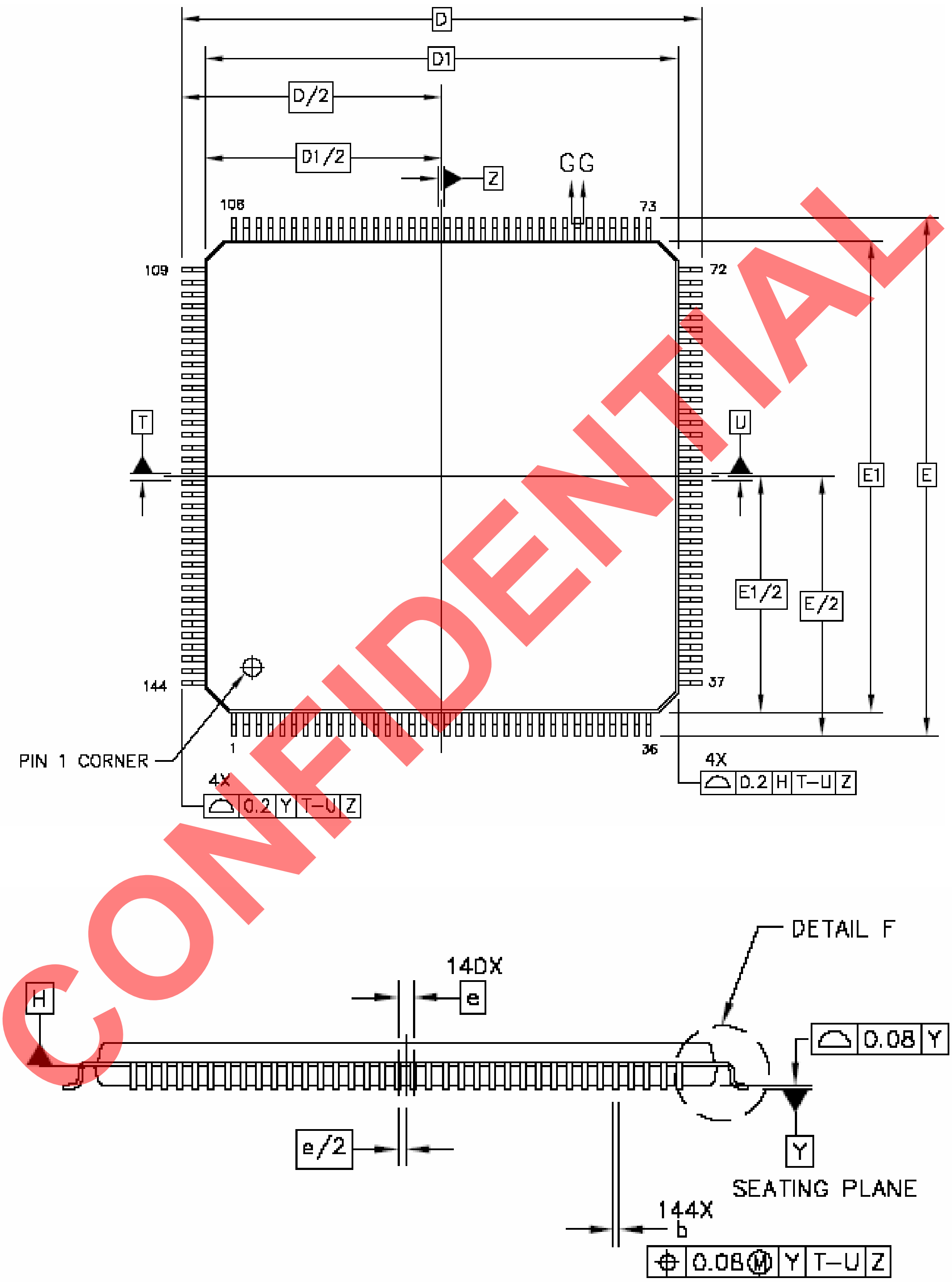
⌘ DIMENSIONS ARE IN MILLIMETERS .

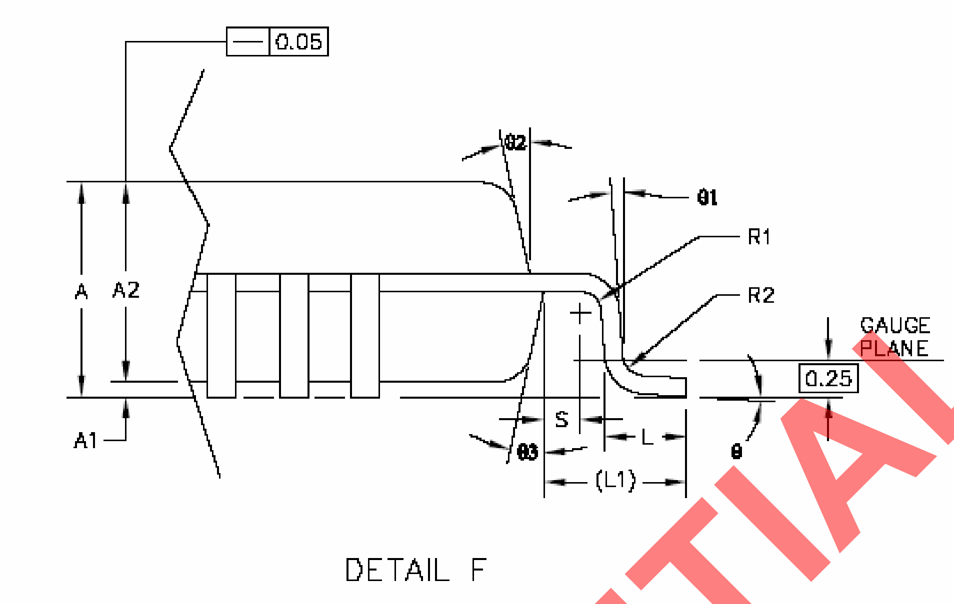
⌘ DIMENSIONS D1 AND E1 DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 mm PER SIDE. DIMENSIONS D1 AND E1 ARE MAXIMUM PLASTIC BODY SIZE DIMENSIONS INCLUDING MOLD MISMATCH .

⌘ INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M --- 1994.

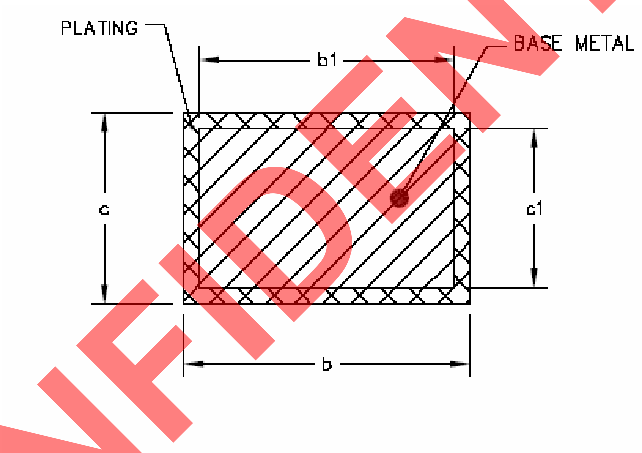
⌘ DIMENSION b DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM b DIMENSION BY MORE THAN 0.08mm . DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT . MINIMUM SPACE BETWEEN PROTRUSION AND AN ADJACENT LEAD IS 0.07mm AND 0.5mm PITCH PACKAGES.

6.3. VC0706PTNA Package Specification (20mm x 20mm)





DETAIL F



SECTION G–G

PACKAGE	LQFP 100L 14*14* 0.9 – 0.5		
DIM	MIN.	NOM.	MAX.
A	---	---	1.6
A1	0.05	---	0.15
A2	1.35	1.4	1.45
b	0.17	0.22	0.27
b1	0.17	0.2	0.23
c	0.09		0.2
c1	0.09		0.16
D	22 BSC		
D1	20 BSC		
e	0.5 BSC		
E	22 BSC		
E1	20 BSC		
L	0.5	0.6	0.75
L1	1 REF		

R1	0.08		---
R2	0.08		0.2
S	0.2		---
θ	0	3.5	7
θ1	0		---
θ2	11	12	13
θ3	11	12	13

NOTES:

⌘ DIMENSIONS ARE IN MILLIMETERS .

⌘ DIMENSIONS D1 AND E1 DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 mm PER SIDE. DIMENSIONS D1 AND E1 ARE MAXIMUM PLASTIC BODY SIZE DIMENSIONS INCLUDING MOLD MISMATCH .

⌘ INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M --- 1994.

⌘ DIMENSION b DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM b DIMENSION BY MORE THAN 0.08mm . DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT . MINIMUM SPACE BETWEEN PROTRUSION AND AN ADJACENT LEAD IS 0.07mm FOR 0.5mm PITCH PACKAGES .

7. CONTACT INFORMATION

Vimicro Corporation
15/F Shining Tower
No.35 Xueyuan Road
Haidian District, Beijing
P. R. China 100083
Tel: 86-10-68948888
Fax: 86-10-68944075
Web: www.vimicro.com

Vimicro Corporation – USA
1758 N. Shoreline Blvd.,
Mountain View, CA 94043
USA
Tel: 1-650-966-1882
Fax: 1-650-966-1885

8. REVISION HISTORY

Version No.	Remarks	Release Date
0.1	Initial draft released for review.	2007-8-15
0.9	Preliminary release.	2007-8-23
1.0	Merge VC0706PTNA and update block diagram, product description	2007-9-28

Buzzer

pro-SIGNAL

Features

- Black in colour
- With internal drive circuit
- Sealed structure
- Wave solderable and washable
- Housing material: Noryl

Applications

- Computer and peripherals
- Communications equipment
- Portable equipment
- Automobile electronics
- POS system
- Electronic cash register

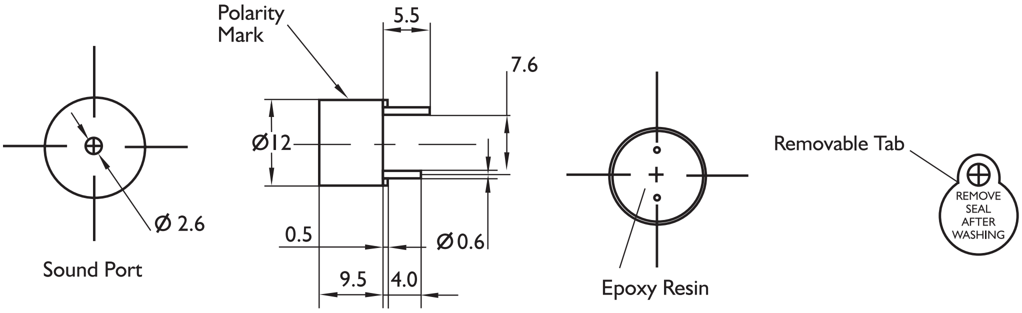
RoHS Compliant

Specifications:

Rated Voltage	: 6V DC
Operating Voltage	: 4 to 8V DC
Rated Current*	: ≤30mA
Sound Output at 10cm*	: ≥85dB
Resonant Frequency	: 2300 ±300Hz
Tone	: Continuous
Operating Temperature	: -25°C to +80°C
Storage Temperature	: -30°C to +85°C
Weight	: 2g

*Value applying at rated voltage (DC)

Diagram



Dimensions : Millimetres
Tolerance : ±0.5mm

Part Number Table

Description	Part Number
Buzzer, Electromech, 6V DC	ABI-009-RC

4x4 Matrix Membrane Keypad (#27899)

This 16-button keypad provides a useful human interface component for microcontroller projects. Convenient adhesive backing provides a simple way to mount the keypad in a variety of applications.

Features

- Ultra-thin design
- Adhesive backing
- Excellent price/performance ratio
- Easy interface to any microcontroller
- Example programs provided for the BASIC Stamp 2 and Propeller P8X32A microcontrollers

Key Specifications

- Maximum Rating: 24 VDC, 30 mA
- Interface: 8-pin access to 4x4 matrix
- Operating temperature: 32 to 122 °F (0 to 50°C)
- Dimensions:
  Keypad, 2.7 x 3.0 in (6.9 x 7.6 cm)
  Cable: 0.78 x 3.5 in (2.0 x 8.8 cm)

Application Ideas

- Security systems
- Menu selection
- Data entry for embedded systems

How it Works

Matrix keypads use a combination of four rows and four columns to provide button states to the host device, typically a microcontroller. Underneath each key is a pushbutton, with one end connected to one row, and the other end connected to one column. These connections are shown in Figure 1.

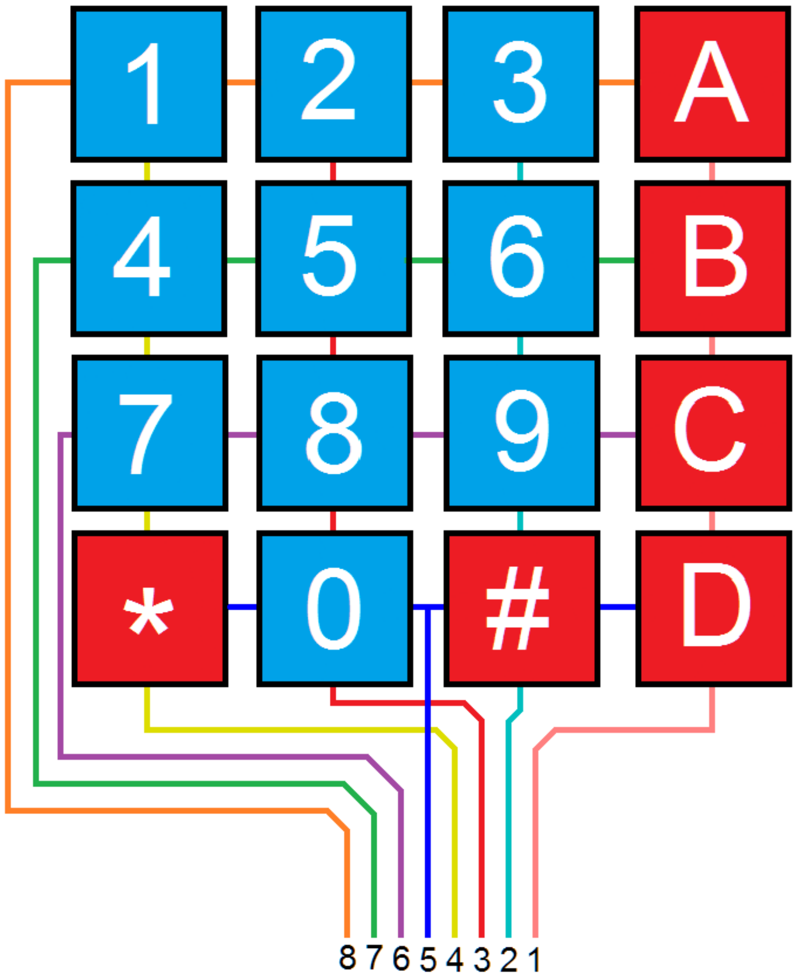


Figure 1: Matrix Keypad Connections

In order for the microcontroller to determine which button is pressed, it first needs to pull each of the four columns (pins 1-4) either low or high one at a time, and then poll the states of the four rows (pins 5-8). Depending on the states of the columns, the microcontroller can tell which button is pressed.

For example, say your program pulls all four columns low and then pulls the first row high. It then reads the input states of each column, and reads pin 1 high. This means that a contact has been made between column 4 and row 1, so button 'A' has been pressed.

Connection Diagrams

Figure 2

For use with the BASIC Stamp example program listed below.

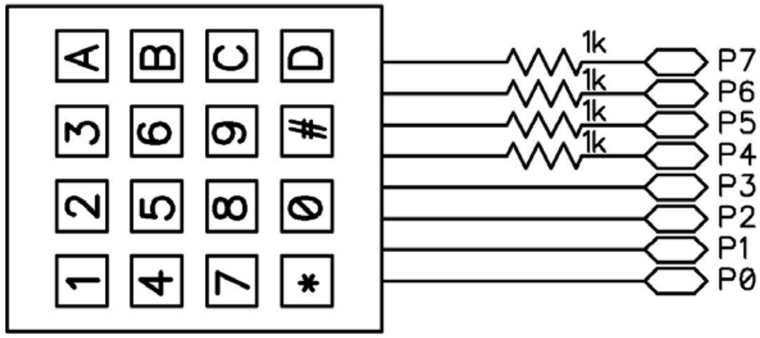
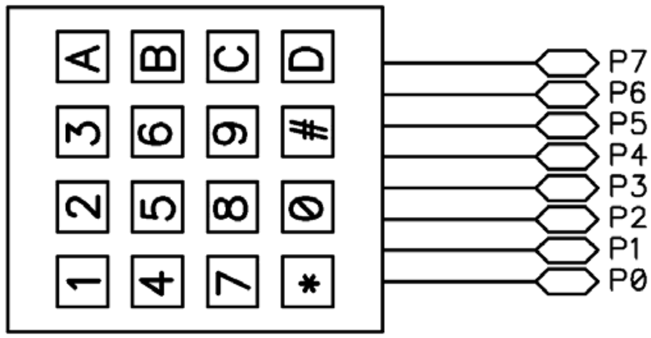


Figure 3

For use with the Propeller P8X32A example program listed below.



BASIC Stamp® Example Code

The example code below displays the button states of the 4x4 Matrix Membrane Keypad. It uses the Debug Terminal, which is built into the BASIC Stamp Editor software. The software is a free download from www.parallax.com/basicstampsoftware.

```
' 4x4MatrixKeypad_Demo.bs2
' Display buttons pressed on the 4x4 Matrix Membrane Keypad
' Author: Parallax HK Engineering

' {$STAMP BS2}
' {$PBASIC 2.5}

row         VAR  Nib                         ' Variable space for row counting
column      VAR  Nib                         ' Variable space for column counting
keypad      VAR  Word                        ' Variable space to store keypad output
keypadOld   VAR  Word                        ' Variable space to store old keypad output
temp        VAR  Nib                         ' Variable space for polling column states

DEBUG CLS                                    ' Clear Debug Terminal
GOSUB Update                                 ' Display keypad graphic

DO
  GOSUB ReadKeypad                           ' Read keypad button states
  DEBUG HOME, BIN16 keypad, CR, CR,          ' Display 16-bit keypad value
              BIN4 keypad >> 12,CR,          ' Display 1st row 4-bit keypad value
              BIN4 keypad >> 8, CR,          ' Display 2nd row 4-bit keypad value
              BIN4 keypad >> 4, CR,          ' Display 3rd row 4-bit keypad value
              BIN4 keypad                    ' Display 4th row 4-bit keypad value
```

```
  IF keypad <> keypadOld THEN                ' If different button is pressed,
    GOSUB Update                             ' update the keypad graphic to clear
  ENDIF                                      ' old display

  IF keypad THEN                             ' Display button pressed in graphic
    GOSUB display
  ENDIF

  keypadOld = keypad                         ' Store keypad value in variable keypadOld
LOOP

' -----[ Subroutine - ReadKeypad ]-------------------------------------------------
' Read keypad button states
ReadKeypad:
  keypad = 0
  OUTL   = %00000000                         ' Initialize IO
  DIRL   = %00000000

  FOR row = 0 TO 3
    DIRB = %1111                             ' Set columns (P7-P4) as outputs
    OUTB = %0000                             ' Pull columns low (act as pull down)
    OUTA = 1 << row                          ' Set rows high one by one
    DIRA = 1 << row

    temp = 0                                 ' Reset temp variable to 0
    FOR column = 0 TO 3
      INPUT (column + 4)                     ' Set columns as inputs
      temp = temp | (INB & (1 << column))    ' Poll column state and store in temp
    NEXT

    keypad = keypad << 4 | (Temp REV 4)      ' Store keypad value
  NEXT
RETURN

' -----[ Subroutine - Update ]-----------------------------------------------------
' Graphical depiction of keypad
Update:
  DEBUG CRSRXY,0,7,
    "+---+---+---+---+",CR,
    "|   |   |   |   |",CR,
    "+---+---+---+---+",CR,
    "|   |   |   |   |",CR,
    "+---+---+---+---+",CR,
    "|   |   |   |   |",CR,
    "+---+---+---+---+",CR,
    "|   |   |   |   |",CR,
    "+---+---+---+---+"
RETURN

' -----[ Subroutine - Display ]----------------------------------------------------
' Display button pressed in keypad graphic
Display:
  IF KeyPad.BIT15 THEN  DEBUG CRSRXY, 02,08,"1"
  IF Keypad.BIT14 THEN  DEBUG CRSRXY, 06,08,"2"
  IF KeyPad.BIT13 THEN  DEBUG CRSRXY, 10,08,"3"
  IF Keypad.BIT12 THEN  DEBUG CRSRXY, 14,08,"A"
  IF KeyPad.BIT11 THEN  DEBUG CRSRXY, 02,10,"4"
  IF Keypad.BIT10 THEN  DEBUG CRSRXY, 06,10,"5"
  IF KeyPad.BIT9  THEN  DEBUG CRSRXY, 10,10,"6"
  IF Keypad.BIT8  THEN  DEBUG CRSRXY, 14,10,"B"
  IF KeyPad.BIT7  THEN  DEBUG CRSRXY, 02,12,"7"
  IF Keypad.BIT6  THEN  DEBUG CRSRXY, 06,12,"8"
  IF KeyPad.BIT5  THEN  DEBUG CRSRXY, 10,12,"9"
```

```
  IF Keypad.BIT4  THEN  DEBUG CRSRXY, 14,12,"C"
  IF KeyPad.BIT3  THEN  DEBUG CRSRXY, 02,14,"*"
  IF Keypad.BIT2  THEN  DEBUG CRSRXY, 06,14,"0"
  IF KeyPad.BIT1  THEN  DEBUG CRSRXY, 10,14,"#"
  IF Keypad.BIT0  THEN  DEBUG CRSRXY, 14,14,"D"
RETURN
```

Propeller™ P8X32A Example Code

The example code below displays the button states of the 4x4 Matrix Membrane Keypad, and is a modified version of the 4x4 Keypad Reader DEMO object by Beau Schwabe.

Note: This application uses the 4x4 Keypad Reader.spin object. It also uses the Parallax Serial Terminal to display the device output. Both objects and the Parallax Serial Terminal itself are included with the with the Propeller Tool v1.2.7 or higher, which is available from the Downloads link at www.parallax.com/Propeller.

```
{{ 4x4 Keypad Reader PST.spin
Returns the entire 4x4 keypad matrix into a single WORD variable indicating which buttons are
pressed. }}

CON

  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000

OBJ
  text : "Parallax Serial Terminal"
  KP   : "4x4 Keypad Reader"

VAR
  word  keypad

PUB start
  'start term
  text.start(115200)
  text.str(string(13,"4x4 Keypad Demo..."))
  text.position(1, 7)
  text.str(string(13,"RAW keypad value 'word'"))

  text.position(1, 13)
  text.str(string(13,"Note: Try pressing multiple keys"))

  repeat
    keypad := KP.ReadKeyPad     '<-- One line command to read the 4x4 keypad
    text.position(5, 2)
    text.bin(keypad>>0, 4)      'Display 1st ROW
    text.position(5,3)
    text.bin(keypad>>4, 4)      'Display 2nd ROW
    text.position(5, 4)
    text.bin(keypad>>8, 4)      'Display 3rd ROW
    text.position(5, 5)
    text.bin(keypad>>12, 4)     'Display 4th ROW
    text.position(5, 9)
    text.bin(keypad, 16)        'Display RAW keypad value
```

Revision History

v1.0: original document
v1.1: Updated Figure 1 on page 2
v1.2: Updated Figure 1 on page 2 (again); updated BS2 comments