Universidad Autónoma de Tamaulipas

Facultad de ingeniería Arturo Narro Siller

Materia: Diseño Electrónico Basado en Sistemas Embebidos

Nombre: Gonzales Saldívar Luis Roberto

Guerrero Gamez Francisco Javier

Martínez Reyes Fernando

Sánchez Ramírez Alan Ariel

Villalobos de León Juan Carlos

Grupo: 8-I

Maestro: García Ruiz Alejandro Humberto

Tareas Unidad 2

Índice

Tabla de contenido

1.- Sensor y tipos de sensores

Es un dispositivo diseñado para captar un estímulo de su entorno y traducir esa información que recibe. Esa información recibida es normalmente convertida a un impulso eléctrico que posteriormente es procesado por una serie de circuitos que generan una acción predeterminada en un aparato, sistema o máquina. Es un artefacto que en algunas aplicaciones transforma una clase de información en otra que se quiere medir o controlar.

Los sensores reaccionan a los cambios de las condiciones físicas alterando sus propiedades eléctricas. Por lo tanto, la mayoría de estos dispositivos industriales dependen de sistemas electrónicos para capturar, analizar y transmitir información sobre el medio ambiente.

Estos sistemas electrónicos se basan en los mismos principios que los circuitos eléctricos para funcionar, por lo que la capacidad de controlar el flujo de energía eléctrica es muy importante. Es decir, un sensor convierte los estímulos como el calor, la luz, el sonido y el movimiento en señales eléctricas. Estas señales se pasan a través de una interfaz que las convierte en un código binario y lo pasa a una computadora para ser procesado.

Las características principales de los sensores son:

Rango. Es el valor mínimo y máximo de la variable física que el sensor puede percibir o medir.

Amplitud. Es la diferencia entre los valores máximos y mínimos de entrada.

Exactitud. El error en la medición se especifica en términos de precisión. Se define como la diferencia entre el valor medido y el valor real. Se define en términos de % de la escala completa o % de la lectura.

Precisión. Se define como la cercanía entre un conjunto de valores y es diferente de la exactitud.

Sensibilidad. Es la relación entre el valor de la salida y el valor de la entrada.

La alineación. Es la máxima desviación entre los valores medidos de un sensor de la curva ideal.

Histéresis. Es la diferencia en la salida cuando la entrada varía de dos maneras, aumentando y disminuyendo.

Resolución. Es el cambio mínimo en la entrada que puede ser detectado por el sensor.

Reproducibilidad. Se define como la capacidad del sensor de producir la misma salida cuando se aplica la misma entrada.

Repetibilidad. Capacidad del sensor de producir la misma salida cada vez que se aplica la misma entrada y todas las condiciones físicas y de medición se mantienen iguales, incluyendo el operador, el instrumento, las condiciones ambientales, etc.

Tiempo de respuesta. Se expresa generalmente como el tiempo en que la salida alcanza un cierto porcentaje de su valor final, en respuesta a un cambio de paso de la entrada.

Tipos de sensores

Sensores de proximidad

Son transductores que detectan objetos o señales que se encuentran cerca del elemento sensor. Existen varios tipos de sensores de proximidad según el principio físico que utilizan.

Inductivos

Han sido diseñados para trabajar generando un campo magnético y detectando las pérdidas de corriente de dicho campo generadas al introducirse en él los objetos de detección férricos. El sensor consiste en una bobina con núcleo de ferrita, un oscilador, un sensor de nivel de disparo de la señal y un circuito de salida.

Al aproximarse un objeto metálico, se inducen corrientes de histéresis en el objeto, debido a ello hay una pérdida de energía y una menor amplitud de oscilación. El circuito sensor reconoce entonces un cambio específico de amplitud y genera una señal que conmuta la salida de estado sólido o la posición on y off.

Magnéticos

Son caracterizados por la posibilidad de distancias grandes de la conmutación, disponible de los sensores con dimensiones pequeñas. Detectan los objetos magnéticos (imanes generalmente permanentes) que se utilizan para accionar el proceso de la conmutación.

Los campos magnéticos pueden pasar a través de muchos materiales no magnéticos, el proceso de la conmutación se puede también accionar sin la necesidad de la exposición directa al objeto. Usando los conductores magnéticos, por ejemplo el hierro; el campo magnético se puede transmitir a mayores distancias para poder llevarse la señal de áreas de alta temperatura.

Capacitivos

Detectan objetos metálicos, o no metálicos, midiendo el cambio en la capacitancia, la cual depende de la constante dieléctrica del material a detectar, su masa, tamaño, y distancia hasta la superficie sensible del detector. Debido a la influencia del objeto a detectar, y del cambio de capacitancia, la amplificación se incrementa haciendo entrar en oscilación el oscilador.

Cuando un objeto conductor se acerca a la cara activa del detector, el objeto actúa como un condensador. El cambio de la capacitancia es significativo durante una larga distancia, si se aproxima un objeto no conductor, (>1) solamente se produce un cambio pequeño en la constante dieléctrica, y el incremento en su capacitancia es muy pequeño comparado con los materiales conductores.

Ultrasónicos

Trabajan libres de roces mecánicos y detectan objetos a distancias de hasta 8 m y emiten impulsos ultrasónicos. Estos se reflejan en un objeto, el sensor recibe el eco producido y lo convierte en señales eléctricas, las cuales son elaboradas en el aparato de valoración.

Trabajan solamente en el aire, y pueden detectar objetos con diferentes formas, superficies y de diferentes materiales. Los materiales pueden ser sólidos, líquidos o polvorientos, sin embargo, han de ser deflectores de sonido. Los sensores trabajan según el tiempo de transcurso del eco, es decir, se valora la distancia temporal entre el impulso de emisión y el impulso del eco.

Sensores fotoeléctricos

Responden al cambio en la intensidad de la luz, requieren de un componente emisor que genera la luz, y un componente receptor que percibe la luz generada por el emisor. Están diseñados especialmente para la detección, clasificación y posicionamiento de objetos; la detección de formas, colores y diferencias de superficie, incluso bajo condiciones ambientales extremas.

Sensores de área

Se emplean en numerosas soluciones como el registro de objetos, personas, vehículos, y el control de presencia y sobredimensionamiento de objetos. Utilizan multi haces de luz para la detección de objetos en movimiento en áreas específicas.

Sensores de presión

Su objetivo es transformar una magnitud física en una eléctrica, en este caso transforman una fuerza por unidad de superficie en un voltaje equivalente a esa presión ejercida. Aunque los formatos son diferentes, destacan en general por su robustez, ya que, en procesos industriales están sometidos a todo tipo de líquidos, existiendo así sensores de presión para agua, de presión para aceite, líquido de frenos, etc.

Sensores de temperatura

Recogen información sobre la temperatura de una fuente y la cambian a una forma que pueda ser comprendida por otro dispositivo. Se trata de una categoría de sensores de uso común que detectan la temperatura o el calor y también mide la temperatura de un medio.

Sensores de flujo

Permiten medir y monitorear el flujo de los medios de proceso, como lubricante o agua de enfriamiento, en una amplia gama de aplicaciones. Cuando reciben una alerta de que el flujo se ha ralentizado o detenido, pueden responder rápidamente y evitar un tiempo de paro imprevisto de la máquina o incluso la detención del sistema en su totalidad.

Sensores de corriente

Detectan la corriente de forma rápida y exacta para controlar con precisión sistemas electrónicos de potencia tales como convertidores de frecuencia, convertidores de tracción, sistemas de alimentación eléctrica ininterrumpida o sistemas de soldadura.

2.- Actuador y tipos de actuadores

Un actuador es un dispositivo que convierte la energía en movimiento o que se utiliza para aplicar fuerza. El dispositivo toma energía de una determinada fuente (que puede ser energía creada por aire, líquido o electricidad) y la convierte en el movimiento deseado. Los dos tipos de movimiento básico deseados son lineal y rotativo, pero también es común el movimiento oscilatorio.

Los actuadores lineales trabajan convirtiendo energía en movimientos lineales rectos, los cuales sirven para empujar o tirar. Los actuadores rotativos, por otro lado, convierten la energía en movimientos oscilatorios y se utilizan, en general, en distintas válvulas, como las de mariposa o de bola.

Los actuadores se utilizan típicamente en aplicaciones industriales y de manufactura. Dispositivos como válvulas, motores, interruptores y bombas dependen ampliamente de ellos. Cada tipo de actuador cuenta con distintas versiones y se ofrece en diferentes tamaños, estilos y modos de operación, de acuerdo con cada aplicación en específico.

Tipos de actuadores

Los actuadores se categorizan según la fuente de energía que utilizan para generar el movimiento.

Actuadores mecánicos

Los actuadores mecánicos trabajan mediante la conversión de un tipo de movimiento (por ejemplo, movimiento rotativo) a otro tipo de movimiento (por ejemplo, un movimiento lineal). Estos actuadores emplean una combinación de componentes para operar, incluidos engranes, poleas, cadenas, resortes y rieles, entre otros.

Actuadores neumáticos

Los actuadores neumáticos son quizás el tipo de actuador más común. Su fuente de energía es el aire comprimido, el cual se utiliza para mover un pistón cuando el aire se libera o descomprime. Los actuadores neumáticos de pistón se usan comúnmente para la operación de válvulas mariposa.

Los actuadores neumáticos son deseables en muchas aplicaciones, debido a que pueden responder con rapidez a operaciones de arranque y paro, así como debido a que no necesitan una fuente de energía para operar. Asimismo, son más baratos, seguros, más poderosos y confiables que otros actuadores.

Actuadores hidráulicos

Los actuadores hidráulicos utilizan líquidos como aceite para generar movimientos lineales, rotativos u oscilatorios. A diferencia del aire, los líquidos son prácticamente imposibles de comprimir, por lo que los actuadores hidráulicos se utilizan en aplicaciones donde se requiere de una fuerza inmensa. Se utilizan en todos los sistemas que manejan cargas grandes, como maquinaria pesada de construcción y barcos.

Actuador eléctrico

Los actuadores eléctricos son de los más limpios, fáciles de usar y de disponibilidad inmediata, debido a que no usan aceite ni requieren aire comprimido para operar. En cambio, dependen de la energía de una fuente externa, como una batería, para conducir un motor y convertir la energía eléctrica en fuerza mecánica. Los actuadores eléctricos operados por motor se utilizan en líneas de tubería de grandes diámetros.

3.- FPGA Fundamentos y Características

Una FPGA (Field Programable Gate Array) es un complejo circuito integrado digital programable compuesto por bloques lógicos configurables (CLB) y puertos de entrada/salida (IOB), cuya interconexión y funcionalidad puede ser programada mediante un lenguaje de descripción especializado.

Su principal característica y ventaja es que pueden reprogramarse para un trabajo específico o cambiar sus requisitos después de haberse producido. Esto también implica que en muchos casos se pueden hacer cambios físicos sin hacer modificaciones costosas en la placa que lo soporta.

Básicamente, una FPGA es un conjunto de múltiples circuitos (lógicos y de otros tipos) dispuestos matricialmente, cuyas interconexiones son programables por el usuario para la aplicación requerida. En una FPGA se programa su hardware, a diferencia de los microcontroladores/microprocesadores, en los que solo existe un hardware fijo y se programa su software (firmware).

Aplicaciones de las FPGA

Tienen un amplio espectro de aplicaciones: como industria automotriz, reconocimiento de voz, controladores de dispositivos, sistemas militares, radiotransmisores/receptores gestionados por software , computación de alto rendimiento, sistemas de emulación de hardware , equipos de investigación en medicina, etc. tienen un papel notable en el desarrollo de sistemas integrados

debido a su capacidad para iniciar el desarrollo de software (SW) del sistema simultáneamente con el hardware (HW), permitir simulaciones de rendimiento del sistema en una fase muy temprana del desarrollo, pudiendo realizar varias particiones del sistema (SW y HW), ensayos e iteraciones antes de la congelación final de la arquitectura del sistema.

arquitectura de los circuitos

Una FPGA se compone de una gran cantidad de elementos denominados Bloques Lógicos Configurables (CLB) -también llamados Logic Cells (LC)- que están interconectados por una serie de uniones programables llamadas Fabric (tejido), que sirven de buses (rutas) para el intercambio de señales entre CLBs. La interfaz que controla la comunicación entre la FPGA y los dispositivos externos está compuesta por los bloques de entrada/salida (I/O).

Los tipos de Bloques Lógicos que constituyen una FPGA, y que son configurables cuando esta es programada, son:

Consultar tablas (LUT). Almacena una lista, predefinida por el usuario, de salidas y entradas lógicas. Se carga con valores relevantes para la FPGA según sus instrucciones específicas. Una LUT es como una RAM que se carga cada vez que se conecta la FPGA. Cuando se configura una FPGA, los bits de la LUT se cargan con unos o ceros, en función de la configuración de la tabla de verdad deseada. En lugar de conectar numerosas puertas lógicas, para crear la tabla de verdad, esto se simula en un tipo especial de RAM. Las LUT con cuatro a seis bits de entrada son ampliamente utilizados.

Multiplexores (MUX). Son circuitos combinacionales con varias entradas y una única salida de datos. Disponen de entradas de control capaces de seleccionar una sola de las entradas de datos existentes. La FPGA los emplea como dispositivo para recibir varias entradas y transmitirlas por un medio de transmisión compartida, dividiendo el medio de transmisión en múltiples canales.

Chanclas tipo D (DFF). Son registros binarios que se utilizan para sincronizar la lógica y guardar estados lógicos entre ciclos de reloj dentro de una FPGA. En cada flanco de reloj, un flip-flop bloquea el valor 1 o 0 en su entrada y mantiene ese valor constante hasta el siguiente flanco de reloj. Se utiliza para mantener el estado dentro del chip.

Sumadores completos (FA). Realiza operaciones aritméticas en la FPGA, sumando o restando dígitos binarios.

4.- PLC Fundamentos y Características

El PLC (Control Lógico Programable) es un equipo comúnmente utilizado por aquellas industrias que buscan dar un salto significativo en la automatización de todos sus procesos. Estos dispositivos se encuentran inmersos en la vida de la sociedad de distintas formas y maneras.

En sí, un PLC es una computadora industrial que usa la ingeniería para la automatización de procesos y tiene como finalidad, que las máquinas desarrollen efectivamente todos los sistemas que la componen. Gracias a estas bondades los PLC se han convertido en una herramienta fundamental para el desarrollo tecnológico de las industrias y todo el entorno social.

La operatividad del PLC está basada en procesos periódicos y de sucesión. Algunas secuencias de estos aparatos son:

Autodiagnóstico: Es la revisión de todos los circuitos. En caso de presentarse un inconveniente, el dispositivo indica una señal.

Lectura de entrada y grabación: Evalúa cada entrada para diagnosticar si está en estado de prendido o apagado y graba estos procesos en la memoria, instaurando una imagen.

Lectura y realización del programa: Utilizando la imagen que se encuentra en la memoria, el ordenador realiza el programa instruido por el usuario.

Registro y actualización de salidas: En este paso se restaura de manera coetánea todas las salidas

Algunas características de los PLC son:

- Controlan las entradas y salidas de manera segura
- Poseen una programación compatible con distintos lenguajes
- Interfaz amigable que facilita la comunicación con el usuario
- Conexión a sistemas de supervisión
- Ejecutan la programación de forma continuada
- Memorias divididas en dos partes

En rasgos generales estos autómatas de clasifican en compactos, modular, banda estrecha, banda baja, montaje en rack, ordenador industrial, software y de ranura.

En el mercado actual existe una gran gama de PLC disponibles para cada necesidad de automatización, siendo los más reconocidos por su calidad y seguridad los fabricados por las

marcas Siemens y Allen Bradley. El primero goza del dominio en el mercado asiático, así como el europeo y el segundo tiene la hegemonía en Estados Unidos. Con relación a los otros continentes ambas marcas tienen receptividad de sus productos.

5.- Raspberry Fundamentos y Características

La Raspberry Pi es una computadora de bajo costo y con un tamaño compacto, del porte de una tarjeta de crédito, puede ser conectada a un monitor de computador o un TV, y usarse con un mouse y teclado estándar. Es un pequeño computador que correo un sistema operativo linux capaz de permitirle a las personas de todas las edades explorar la computación y aprender a programar lenguajes como Scratch y Python. Es capaz de hacer la mayoría de las tareas típicas de un computador de escritorio, desde navegar en internet, reproducir videos en alta resolución, manipular documentos de ofimática, hasta reproducir juegos.

Además la Raspberry Pi tiene la habilidad de interactuar con el mundo exterior, puede ser usada en una amplia variedad de proyectos digitales, desde reproductores de música y video, detectores de padres, estaciones meteorológicas hasta cajas de aves con cámaras infrarrojas.

Consiste en una placa base que soporta distintos componentes de un ordenador como un procesador ARM de hasta 1500 MHz, un chip gráfico y una memoria RAM de hasta 8 GB.

Además, tiene otras muchas otras posibilidades.

- Sus puertos y entradas, permite conectar dispositivos periféricos. Por ejemplo, una pantalla táctil, un teclado e incluso un televisor.
- Contiene un procesador gráfico VideoCoreIV, con lo que permite la reproducción de vídeo -incluso en alta definición-.
- Permite la conexión a la red a través del puerto de Ethernet, y algunos modelos permiten conexión Wifi y Bluetooth.
- Consta de una ranura SD que permite instalar, a través de una tarjeta microSD, sistemas operativos libres.

Limitaciones de la Raspberry PI

En primer lugar, si bien es un microrordenador, su velocidad no es extremadamente rápida. Hablando rápido, es similar a la de un ordenador de principios de siglo.

Por ello, a veces no es la opción adecuada para navegar por la red o redactar documentos. Tampoco cargará un videojuego actual, ni es la adecuada para ejecutar programas de retoque fotográfico.

Ni qué decir tiene que, como servidor, solo es válido para webs en entorno de pruebas. En caso contrario, siempre se debe optar por un hosting profesional.

Por otro lado, se debe saber que la Raspberry PI no tiene un circuito de encendido y apagado. Este microordenador tiene un consumo bajo, pero debe ser enchufado a la corriente a través de un cargador. Para ello se recomienda uno de unos 2,5 amperios y 5 voltios.4

Han existido múltiples modelos:

Raspberry Pi 1 modelo A

Raspberry Pi 1 modelo B

Raspberry Pi 2 modelo B

Raspberry Pi 3 modelo B

Raspberry Pi 3 modelo B+

Raspberry Pi 3 modelo A+

Raspberry Pi 4 modelo B

Raspberry Pi 400

6.- Arduino Fundamentos y Características

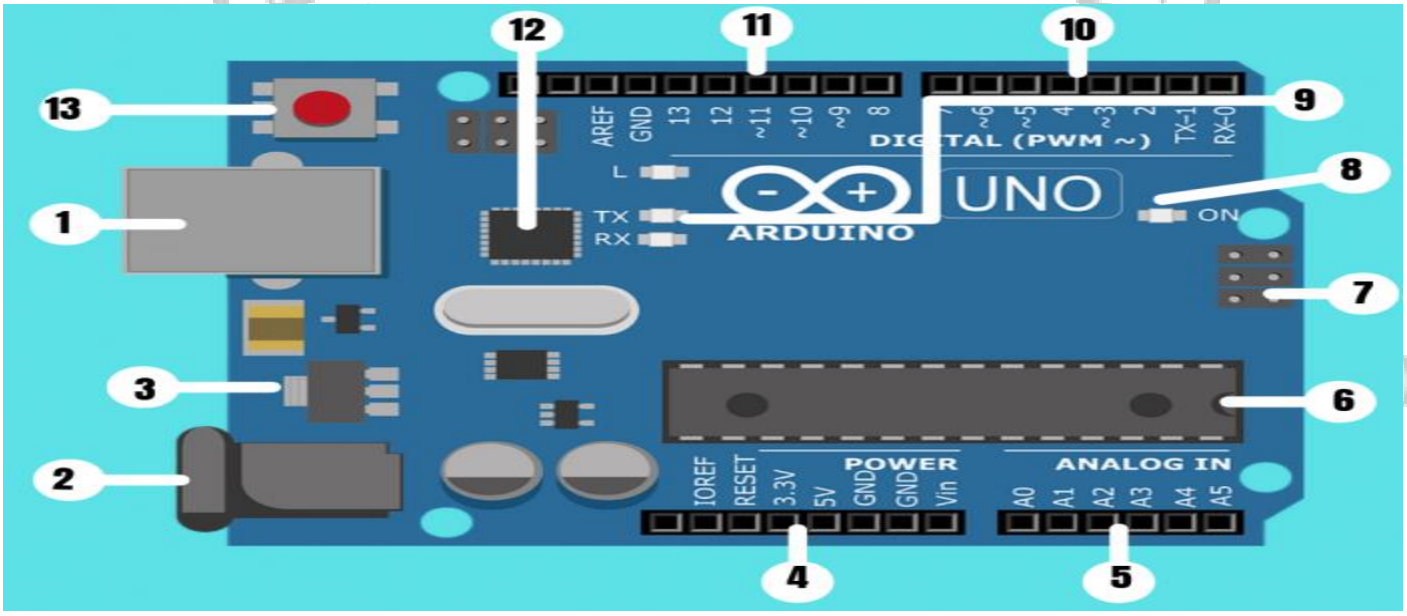
Arduino es una plataforma electrónica de código abierto basada en hardware y software de fácil manejo que se utiliza para la construcción de proyectos electrónicos. El mismo, está formado por una tarjeta o placa física de circuito programable (normalmente denominada micro-controlador) y un software, o IDE (Integrated Development Environment) que se instala en tu ordenador, y que se utiliza para picar y cargar código del ordenador a la tarjeta física.

Las placas de Arduino se caracterizan por leer entradas – la luz de un sensor, pulsar un botón, o un mensaje de texto enviado a una Red Social, para convertirla en una salida, activando un motor, encendiendo un LED, publicando algo on-line, permite indicar a la placa qué hacer enviando un conjunto de instrucciones al micro-controlador de la placa. Para ello utilizas el lenguaje de programación de Arduino, basado en Wiring y el software de Arduino (IDE), basado en Processing.

La placa Arduino funciona una vez se conecta a un ordenador a través de un USB, donde se conecta con el entorno de desarrollo Arduino (IDE). El usuario escribe el código de Arduino en el IDE, y luego lo sube al microcontrolador que ejecuta el código, interactuando con las entradas y salidas como sensores, motores y luces.

El código abierto de Arduino es particularmente amigable para los usuarios nuevos y experimentados. Hay miles de ejemplos de código de Arduino disponibles en Internet.

Partes de una placa Arduino



Alimentación USB/5VDC (1 y 2):

El Arduino UNO puede ser alimentado desde un cable USB de tipo B o mini procedente de tu ordenador o desde una fuente de alimentación entre 6V y 18V. En la imagen de arriba, la conexión USB está etiquetada (1) y el conector de la fuente de alimentación (2).

Ademas, la conexión USB sirve para cargar código en la placa de Arduino desde donde se pueden enviar datos de la programación e instrucciones a la placa.

Regulador de voltaje (3):

El regulador de voltaje controla la cantidad de voltaje que se deja entrar en la placa de Arduino; por lo que no dejará pasar un voltaje superior al establecido que podría dañar el circuito.

Conexiones (4):

Los pines o conexiones de Arduino se utilizan para conectar los cables que se van a necesitar para construir un circuito. Este tipo de conexiones tiene varios pines, cada uno de los cuales está impreso en la placa y se utilizan para diferentes funciones:

Reset: Permite el reseteo del micro controlador.

5V y 3.3V: la clavija de 5V suministra 5 voltios de energía, y la clavija de 3.3V suministra 3.3 voltios de energía. La mayoría de los componentes simples usados con el Arduino funcionan bien con 5 o 3.3 voltios.

GND: Hay varios pines GND en Arduino, se usan para conectar a tierra el circuito.

VIN: Se usa para conectar la alimentación de la placa con una fuente externa de entre 6 y 12VDC.

Puertos de entrada Analógicos (5):

El área de pines bajo la etiqueta ‘Analog In’ (A0 a A5 en la UNO) son los pines de entrada analógica. Estos pines pueden leer la señal de un sensor analógico y convertirla en un valor digital que podemos leer e interpretar.

Micro-controlador Atmega 328 (6):

Esta zona de la placa es el circuito integrado que actúa como cerebro/procesador de la placa de Arduino sobre el que vamos a implementar la programación.

Entrada ICSP (In Chip Serial Programmer) (7):

Esta entrada realiza la función de acceso directo para grabar, desde el PC al circuito, cualquier programa sin necesidad de utilizar el puerto USB.

Indicador LED de alimentación (8):

LED de encendido de la placa de Arduino que indica si el microprocesador esta activo.

LED TX RX (9):

TX es la abreviatura de transmisión de datos y RX es la abreviatura de recepción de datos. Estas marcas comunes aparecen con regularidad en la electrónica para indicar los pines responsables de la comunicación serie. Así mismo, Estos LED se activan visualmente cuando la placa esta recibiendo o transmitiendo datos.

Puertos Digitales (10):

Estos pines se pueden utilizar tanto para la entrada digital (como para indicar si se pulsa un botón) como para la salida digital (como para alimentar un LED).

Puerto de conexiones (11):

5 entradas o salidas auxiliares (de la 8 a la 12).

3 salidas 9, 10 y 11 que permiten la modulación por ancho o de pulso.

Salida 13 que sirve para conectar un led directamente a tierra.

Salida a tierra GND.

Pin AREF que se utiliza para fijar una tensión de referencia externa (entre 0 y 5 voltios) como límite superior de las clavijas de entrada analógica.

Chip de Arduino (12):

Permite identificar un dispositivo USB por el ordenador, es como su tarjeta de identificación o D.N.I. personal

Botón de RESET (13):

Al presionarlo conectará temporalmente el pin de reset a tierra y reiniciará cualquier código que esté cargado en el micro-controlador de Arduino.

7.- Inst: Delay propósito, sintaxis, ejemplos, etc.

En Arduino, el Delay es una función que hace que el procesador espere. Esta espera permite no hacer nada y esperar hasta la ejecución de la siguiente instrucción durante un retardo de tiempo definido. Entonces esta función tiene un parámetro de entrada del tipo entero, que es la espera en milisegundos. De hecho, esta es una de las instrucciones más usadas en el lenguaje Arduino.

La función de Arduino Delay está basada en el uso de timers. Por ejemplo, la idea es que el procesador espera a que el timer le notifique de su desbordamiento. Cuando esto ocurre se genera una interrupción por Hardware. Esta interrupción ocurre en un tiempo conocido lo cual permite detener al procesador durante un tiempo definido. Quizás la palabra detener no sea la más apropiada, es más una espera y mientras espera esta verificando o esperando a las interrupciones del timer.

Su sintaxis:

void delay(retardoMilisegundos); // Donde “retardoMilisegundos” es un numero entero que representa el valor del retardo en milisegundos.

Delay(1000);

Ejemplo:

En el siguiente ejemplo, se usará a la función de Arduino millis para generar un retardo de 1 segundos = 1000ms para enviar un mensaje por el puerto serial.

```
int contador = 0;
int pin1 = 13;
int pin2 = 12;
int pin3 = 11;
int pin4 = 10;
void setup() {
  pinMode(pin1, OUTPUT);
  pinMode(pin2, OUTPUT);
  pinMode(pin3, OUTPUT);
```

```
  pinMode(pin4, OUTPUT);
}
void loop() {
  digitalWrite(pin1, HIGH); digitalWrite(pin2, LOW); digitalWrite(pin3, LOW); digitalWrite(pin4, LOW);
  delay(1000);
  digitalWrite(pin1, LOW); digitalWrite(pin2, HIGH); digitalWrite(pin3, LOW); digitalWrite(pin4, LOW);
  delay(1000);
  digitalWrite(pin1, LOW); digitalWrite(pin2, LOW); digitalWrite(pin3, HIGH); digitalWrite(pin4, LOW);
  delay(1000);
  digitalWrite(pin1, LOW); digitalWrite(pin2, LOW); digitalWrite(pin3, LOW); digitalWrite(pin4, HIGH);
  delay(1000);
}
```

8.- Instrucciones que componen al paquete serial (readString, TimeOut, begin, etc.)

La comunicación serie es muy importante porque gran parte de los protocolos utilizados actualmente son serie y además muchos dispositivos de comunicación inalámbrica usan la comunicación serie para hablar con Arduino como los módulos bluetooth y los módulos Xbee. También la comunicación serie es la que se usa generalmente para comunicar el Arduino con el Ordenador.

Todas las placas Arduino tienen al menos un puerto serie disponible en los pines digitales 0 (RX) y 1 (TX) compartido con el USB. Por lo tanto no es posible usar estos pines como entradas/salidas digitales.

El Arduino mega dispone de tres puertos adicionales Serial1 on pins 19 (RX) and 18 (TX), Serial2 on pins 17 (RX) and 16 (TX), Serial3 on pins 15 (RX) and 14 (TX). Estos pines no están conectados al interfaz USB del Arduino.

Algunas funciones son:

Serial.readString()

Lee los caracteres del buffer serie en una cadena. La función se anula si el tiempo de espera se ha alcanzado. Esta función es parte de la clase Stream, y es llamada por cualquier clase que herede de ella (Wire, Serial, etc.).

Serial.setTimeout()

Establece el máximo de milisegundos para esperar datos en serie. El valor predeterminado es 1000 milisegundos.

Sintaxis: Serial.setTimeout(time)

Serial: objeto de puerto serie. Consulte la lista de puertos seriales disponibles para cada placa en la página principal Serial .

time: duración del tiempo de espera en milisegundos. Tipos de datos permitidos: long.

Begin

Establece la velocidad de datos en bits por segundo (baudios) para la transmisión de datos en serie. Para comunicarse con Serial Monitor, asegúrese de usar una de las velocidades en baudios enumeradas en el menú en la esquina inferior derecha de su pantalla. Sin embargo, puede especificar otras velocidades

Sintaxis: Serial.begin(speed, config)

Serial: objeto de puerto serie.

speed: en bits por segundo (baudios). Tipos de datos permitidos: long.

config: establece bits de datos, paridad y parada. Los valores válidos son: ERIAL_5N1, SERIAL_6N1, SERIAL_7N1, SERIAL_8N1(predeterminado), SERIAL_5N2, SERIAL_6N2, SERIAL_7N2, SERIAL_8N2, SERIAL_5E1: paridad, SERIAL_6E1, SERIAL_7E1, SERIAL_8E1, SERIAL_5E2, SERIAL_6E2, SERIAL_7E2, SERIAL_8E2, SERIAL_5O1par : paridad impar

Serie.leer()

Lee los datos seriales entrantes.

Sintaxis: Serial.read()

Serial: objeto de puerto serie.

Serial.print()

Imprime datos en el puerto serial como texto ASCII legible por humanos. Este comando puede tomar muchas formas. Los números se imprimen utilizando un carácter ASCII para cada dígito. Los flotantes se imprimen de manera similar como dígitos ASCII, por defecto con dos decimales. Los bytes se envían como un solo carácter. Los caracteres y las cadenas se envían tal cual.

Sintaxis: Serial.print(val)

Serial: objeto de puerto serie.

val: el valor a imprimir. Tipos de datos permitidos: cualquier tipo de datos.

Serial.readBytesUntil()

Serial.readBytesUntil() lee caracteres del búfer serial en una matriz. La función finaliza (las comprobaciones se realizan en este orden) si se ha leído la longitud determinada, si se agota el tiempo de espera o si se detecta el carácter de terminación (en cuyo caso, la función devuelve los caracteres hasta el último carácter antes del terminador proporcionado). El terminador en sí no se devuelve en el búfer.

Sintaxis: Serial.readBytesUntil(character, buffer, length)

Serial: objeto de puerto serie.

character: el carácter a buscar. Tipos de datos permitidos: char.

buffer: el búfer para almacenar los bytes. Tipos de datos permitidos: matriz de charo byte.

length: el número de bytes a leer. Tipos de datos permitidos: int.

9.- Arreglos en Arduino

Un arreglo o array es un conjunto de valores a los que se accede con un número índice. Cualquier valor puede ser recogido haciendo uso del nombre de la matriz y el número del índice. El primer valor de la matriz es el que está indicado con el índice 0, es decir el primer valor del conjunto es el de la posición 0. Un array tiene que ser declarado y opcionalmente asignados valores a cada posición antes de ser utilizado.

Declaración de un array:

```
int miArray[] = {valor0, valor1, valor2…}
```

Las matrices se utilizan a menudo con estamentos de tipo bucle, en los que la variable de incremento del contador del bucle se utiliza como índice o puntero del array. Utilizando un bucle tipo for, el contador comienza en cero 0 y escribe el valor que figura en la posición de índice 0 en la serie que realizada sigue escribiendo en las siguientes posiciones. Con un bucle for podremos recorrer un array ya sea para leerlo o para escribir en él.

```
int myPins[5];
for (int i = 0; i < 5; i = i + 1) {
  Serial.println(myPins[i]);
}
```

Un aspecto muy importante que se debe tener en cuenta, es que antes de nada, hemos de definir el tamaño, o el espacio máximo que puede ocupar el array que vamos a crear.

A los arrays que tienen un tamaño variable, se les denomina arrays dinámicos y aunque se pueden implementar , debemos aprender un poco más acerca del lenguaje y crear diferentes estrategias asociadas con punteros.

En el momento de crear una array, tenemos dos opciones de instanciarlo fuera del Setup o el loop.

- Inicializarlo en vacio estableciendo el tamaño máximo
- Inicializarlo con valores sin establecer tamaño máximo.

10.- Métodos con Arduino

Clases y Objetos

Un paradigma es el resultado de un proceso social en el cual un grupo de personas desarrolla nuevas ideas y crea principios y prácticas alrededor de estas ideas, resumiendo: un paradigma es una metodología de trabajo.

En programación, se trata de un enfoque concreto de desarrollar y estructurar el desarrollo de programas.

Paradigma imperativo:

Consiste en una secuencia de instrucciones que el ordenador debe ejecutar.

Los elementos más importantes en esta forma de programar son:

Variables, zonas de memoria donde guardamos información.

Tipos de datos, son los valores que se pueden almacenar.

Expresiones, corresponde a operaciones entre variables (del mismo o distinto tipo)

Paradigma funcional:

Consiste en el uso de funciones que realizan su tarea como si de una caja negra se tratase

Pese a que trabajamos con funciones, el modelo desarrollado hasta ahora con Arduino no verifica todos los requisitos del paradigma de programación funcional ya que, en nuestro caso existe el concepto de variable, que no se da en programación funcional.

Paradigma orientado a objetos:

Es el más popular en la actualidad.

Se fundamenta en la “fusión” de datos y funciones que operan sobre esos datos dentro de un nuevo tipo de dato.

Al nuevo tipo de dato se le llama CLASE.

A cada variable de una clase se le llama OBJETO.

Propiedades del paradigma orientado a objetos

Encapsulamiento

Significa que los datos pertenecen a un objeto (espacio de nombres del objeto).

Podemos ir más allá y ocultar los datos de un objeto a cualquier otro objeto o código que trate de hacer uso de ellos. Serían sólo accesibles al propio objeto y, en algunos casos, a objetos de sus clases descendientes.

Herencia:

Es la propiedad de crear nuevos datos a partir de un objeto

Una clase es un nuevo tipo de dato. Contiene :

otros datos (que pueden ser de cualquier tipo)

Funciones, que operan sobre esos datos.

Las variables incluidas en una clase se denominan **ATRIBUTOS.**

Las clases pueden contener funciones. A éstas se les denomina **MÉTODOS.**

11.- Técnicas de apoyo a la toma de decisiones

Pensamiento computacional

Es una habilidad que permite resolver problemas aprovechando la potencia de dispositivos con capacidad de cómputo, como los ordenadores o los teléfonos móviles inteligentes.

Otras definiciones de pensamiento computacional han ido surgiendo en la literatura científica desde entonces. Entre las más aceptadas se encuentran la de Aho y la de la Royal Society:

El pensamiento computacional es el proceso que permite formular problemas de forma que sus soluciones pueden ser representadas como secuencias de instrucciones y algoritmos.

El pensamiento computacional es el proceso de reconocimiento de aspectos de la informática en el mundo que nos rodea, y aplicar herramientas y técnicas de la informática para comprender y razonar sobre los sistemas y procesos tanto naturales como artificiales.

Teniendo en cuenta que vivimos en un mundo cada vez más digital, esta habilidad resulta fundamental para la vida en sociedad en el siglo XXI y, por tanto, para jóvenes en edad escolar.

Según esta definición operativa, el pensamiento computacional es un proceso de resolución de problemas que incluye las siguientes características:

Formular problemas de forma que se permita el uso de un ordenador y otras herramientas para ayudar a resolverlos.

Organizar y analizar lógicamente la información.

Representar la información a través de abstracciones como los modelos y las simulaciones.

Automatizar soluciones haciendo uso del pensamiento algorítmico (estableciendo una serie de pasos ordenados para llegar a la solución).

Identificar, analizar e implementar posibles soluciones con el objetivo de lograr la combinación más efectiva y eficiente de pasos y recursos.

Generalizar y transferir este proceso de resolución de problemas para ser capaz de resolver una gran variedad de familias de problemas.

12.- Inst: millis

La función millis() devuelve el número de milisegundos desde que la placa Arduino empezó a ejecutar, luego de un reinicio o el encendido. Este número se desbordará (volverá a cero), después de aproximadamente 50 días.

Retorna la cantidad de milisegundos en un valor long sin signo (unsigned long).

```
unsigned long tiempo;
void setup(){
Serial.begin(9600);
}
void loop(){
Serial.print("Tiempo: ");
tiempo = millis();
// imprime el tiempo transcurrido desde el inicio del programa
Serial.println(time);
```

```
// esperar un segundo para no enviar cantidades masivas de datos
delay(1000);
}
unsigned long tiempo;
void setup(){
Serial.begin(9600);
}
void loop(){
Serial.print("Tiempo: ");
tiempo = millis();
// imprime el tiempo transcurrido desde el inicio del programa
Serial.println(time);
// esperar un segundo para no enviar cantidades masivas de datos
delay(1000);
}
```

Se debe tener en cuenta que como el valor de retorno para millis() es un long sin signo (unsigned long), pueden producirse errores lógicos si un programador intenta hacer operaciones aritméticas con tipos de datos más pequeños, como de tipo int. Incluso con los long con signo se pueden producir errores ya que su valor máximo es la mitad que la de su contraparte sin signo.

¿Por qué usar millis() en lugar de delay()?

Presentaré dos ventajas al utilizar millis() para crear retardos, en comparación con el uso habitual de delay().

1. Cronometraje preciso

La primera ventaja que discutiremos es la exactitud en el tiempo.

Con millis() podemos garantizar que el bucle se ejecute tantas veces como queramos dentro del retardo sin afectar su extensión, independientemente del tiempo de ejecución (obviamente, siempre que el tiempo de ejecución de todas las instrucciones sea menor al retardo deseado).

Con delay() esto no es posible, ya que no sabemos cuánto tiempo durará el tiempo de ejecución de todas las instrucciones de programa que están dentro del ciclo.

2. Sin bloqueo

La ventaja principal de la función millis() es que no nos impide ejecutar otro código mientras estamos esperando.

Función micros() y desbordamiento

Al igual que delay() tiene una versión en microsegundos llamada delayMicroseconds(), la función millis() tiene como compañera para tiempos breves la función micros(). Si necesitamos una mejor resolución, micros() puede ser el camino a seguir. Sin embargo, tenga en cuenta que el valor devuelto por micros() se desbordará (volverá a cero) después de aproximadamente 70 minutos, en comparación con los 50 días de millis(). “Desbordamiento” significa que el conteo llega a su máximo, y entonces los valores de retorno de la función recomienzan desde cero

13.- Instrucciones para trabajar con escritura/lectura de señales digitales en arduino

Una señal digital es un tipo de señal generada por algún tipo de fenómeno electromagnético en que cada signo que codifica el contenido de la misma puede ser analizado en término de algunas magnitudes que representan valores discretos, en lugar de valores dentro de un cierto rango.

Los sistemas digitales, como por ejemplo un microcontrolador, usan la lógica de dos estados representados por dos niveles de tensión eléctrica, uno alto, H y otro bajo, L (de High y Low, respectivamente, en inglés). Por abstracción, dichos estados se sustituyen por ceros y unos, lo que facilita la aplicación de la lógica y la aritmética binaria. Si el nivel alto se representa por 1 y el bajo por 0, se habla de lógica positiva y en caso contrario de lógica negativa.

Se debe mencionar que, además de los niveles, en una señal digital están las transiciones de alto a bajo y de bajo a alto, denominadas flanco de bajada y de subida, respectivamente. En una señal digital, se denomina flanco a la transición del nivel bajo al alto (flanco de subida) o del nivel alto al bajo (flanco de bajada).

Ventajas de las señales digitales

- Ante la atenuación, puede ser amplificada y reconstruida al mismo tiempo, gracias a los sistemas de regeneración de señales.
- Cuenta con sistemas de detección y corrección de errores, en la recepción.
- Facilidad para el procesamiento de la señal. Cualquier operación es fácilmente realizable a través de cualquier software de edición o procesamiento de señal.
- Permite la generación infinita con pérdidas mínimas en la calidad. Esta ventaja solo es aplicable a los formatos de disco óptico; la cinta magnética digital, aunque en menor medida que la analógica (que solo soporta como mucho 4 o 5 generaciones), también va perdiendo información con la multigeneración.
- Las señales digitales se ven menos afectadas a causa del ruido ambiental en comparación con las señales analógicas y permite que haya menos interferencia sea una señal fluida o continua.

En arduino para tratar las entradas y salidas digitales se usan las siguientes funciones:

pinMode() – configura en el pin especificado si se va a comportar como una entrada o una salida.

digitalWrite() – Escribe un valor HIGH o LOW en el pin digital especificado. Si el pin está configurado como OUTPUT pone el voltaje correspondiente en el pin seleccionado. Si el pin está configurado como INPUT habilita o deshabilita la resistencia interna de pull up del correspondiente pin.

digitalRead() – lee el valor del pin correspondiente como HIGH o LOW.

14.- Instrucciones para trabajar con escritura/lectura de señales analógicas en arduino

Una señal analógica es una señal generada por algún tipo de fenómeno electromagnético; que es representable por una función matemática continua en la que es variable su amplitud y periodo (representando un dato de información) en función del tiempo. Algunas magnitudes físicas comúnmente portadoras de una señal de este tipo son eléctricas como la intensidad, la tensión y la potencia, pero también pueden ser hidráulicas como la presión y térmicas como la temperatura.

En la mayoría de los sistemas automatizados, utilizamos sensores para supervisor el mundo exterior. Estos sensores nos entregan una señal analógica. Como ejemplo mas practico podemos citar el caso de los sensores de:

- Temperatura
- Distancia
- Presión
- PH
- Intensidad de corriente en un circuito
- Caudal de agua en una tubería
- Velocidad de un coche
- % de Humedad

Para leer este tipo de señales continuas, necesitamos un convertidor analógico a digital (o ADC por sus siglas en ingles) y que nos permite leer el valor de una señal analógica en un momento dado.

Estos convertidores toman una muestra del valor actual de la señal y nos entregan su valor instantáneo, medido en Voltios.

Mediante la lectura repetida de muestras a lo largo del tiempo podemos reconstruir la señal original con mayor o menor precisión, dependiendo de la exactitud de nuestra medida y de la velocidad a la que pueda tomar esas muestras.

Las entradas analógicas del modelo Uno son las correspondientes a los pines de A0 a A5. Se caracterizan por leer valores de tensión de 0 a 5 Voltios con una resolución de 1024 (10 bits). Si dividimos 5 entre 1024 tenemos que ser capaz de detectar variaciones en el nivel de la señal de entrada de casi 5 mV.

El Arduino UNO dispone de seis convertidores analógico a digital, nominados de A0 hasta A5, rotuladas como ANALOG IN

Los convertidores de Arduino UNO y Mega son de 10 bits de resolución por lo que nos devolverá valores entre 0 y 210 = 1.024 para tensiones entre 0 y 5V.

Asegúrate de no usar sensores que puedan dar más de 5V máximo (con Arduino UNO y Mega), ya que dañarías el chip principal de Arduino.

Para usar las entradas analógicas, utilizaremos un potenciómetro en vez de un sensor, el cual conectaremos como se indica el siguiente circuito. Aplicaremos tensión de 5 Vdc a los extremos del potenciómetro y conectamos el pin central (cursor o variable) a la entrada del puerto A5 del Arduino.

15.- software: Proteus, uso, aplicación, funcionalidad, características, etc.

Proteus es una aplicación para la ejecución de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño del esquema electrónico, programación del software, construcción de la placa de circuito impreso, simulación de todo el conjunto, depuración de errores, documentación y construcción.

Sin la utilización de la suite Proteus, el proceso para construir un equipo electrónico basado en un microprocesador se compone de cinco etapas. Sólo al final del proceso somos capaces de detectar los errores y cualquier problema exige volver a ejecutar el ciclo completo, el cual se compone de:

- Diseño del circuito eléctrico
- Diseño del circuito impreso
- Construcción del prototipo físico
- Desarrollo del software
- Pruebas de funcionamiento

El depurado de errores puede convertirse en una labor ardúa en tiempo y recursos, lo que conlleva un alto coste económico. Sin embargo con la herramienta Proteus el proceso queda definido de la siguiente manera:

- Diseño del circuito electrónico
- Desarrollo del software
- Simulación del circuito y programa
- Diseño del circuito impreso
- Construcción del prototipo físico

Con Proteus las fases de prueba no suponen la necesidad de volver a construir nuevos prototipos, con el ahorro de costos y tiempo que ello supone.

Los diferentes módulos que componen Proteus se pueden adquirir de forma independiente añadiendo nuevas funcionalidades a medida que aumentan nuestras necesidades de desarrollo y

producción. Además, la capacidad de simular cada una de las familias de microprocesadores también es objeto de adquisición por separado. De esta manera podemos empezar adquiriendo unas funcionalidades básicas e ir adquiriendo progresivamente nuevas características aprovechando al máximo nuestras inversiones en la herramienta y asegurar al máximo los costes de inversión en el software.

características

Las principales características de Proteus son:

- Entorno de diseño gráfico de esquemas electrónicos (ISIS) extremadamente fácil de utilizar y dotado de poderosas herramientas para facilitar el trabajo del diseñador.
- Entorno de simulación prospice mixto entre el estandar SPICE3F5 y la tecnología exclusiva de Proteus de Modelación de Sistemas Virtuales (VSM)
- Entorno de diseño de placas de circuito impreso (ARES) de ultra-altas prestaciones con bases de datos de 32 bits, posicionador automático de elementos y generación automática de pistas con tecnologías de autocorte y regeneración.
- Moderno y atractivo interface de usuario estandarizado a lo largo de todos las herramientas que componen el entorno PROTEUS.
- La mayor parte de los módulos que componen PROTEUS han sido escritos por el mismo equipo, garantizando al máximo nivel posible la compatibilidad e inter-operatividad de todas las herramientas que componen el entorno PROTEUS, asegurando su estabilidad al máximo.

16.- software: LabVIEW

LabVIWE es una plataforma y entorno de desarrollo para diseñar sistemas, con un lenguaje de programación visual gráfico pensado para sistemas hardware y software de pruebas, control y diseño, simulado o real y embebido.

Los programas desarrollados con LabVIEW se llaman Instrumentos Virtuales, o VIs, y su origen provenía del control de instrumentos, aunque hoy en día se ha expandido ampliamente no solo al control de todo tipo de electrónica (Instrumentación electrónica) sino también a su programación embebida, comunicaciones, matemáticas, etc.

Caracteristicas

Su principal característica es la facilidad de uso, válido para programadores profesionales como para personas con pocos conocimientos en programación pueden hacer programas relativamente

complejos, imposibles para ellos de hacer con lenguajes tradicionales. También es muy rápido hacer programas con LabVIEW y cualquier programador, por experimentado que sea, puede beneficiarse de él. Los programas en LabView son llamados instrumentos virtuales (VIs) Para los amantes de lo complejo, con LabVIEW pueden crearse programas de miles de VIs (equivalente a millones de páginas de código texto) para aplicaciones complejas, programas de automatizaciones de decenas de miles de puntos de entradas/salidas, proyectos para combinar nuevos VIs con VIs ya creados, etc. Incluso existen buenas prácticas de programación para optimizar el rendimiento y la calidad de la programación. El labView 7.0 introduce un nuevo tipo de subVI llamado VIs Expreso (Express VIS). Estos son VIs interactivos que tienen una configuración de caja de diálogo que permite al usuario personalizar la funcionalidad del VI Expreso. El VIs estándar son VIs modulares y personalizables mediante cableado y funciones que son elementos fundamentales de operación de LabView.

Presenta facilidades para el manejo de:

Interfaces de comunicaciones:

- Puerto serie
- Puerto paralelo
- GPIB
- PXI
- VXI
- TCP/IP, UDP, DataSocket
- Irda
- Bluetooth
- USB
- OPC...

Capacidad de interactuar con otros lenguajes y aplicaciones:

- DLL: librerías de funciones
- .NET
- ActiveX
- Multisim
- Matlab/Simulink
- AutoCAD, SolidWorks, etc
- Herramientas gráficas y textuales para el procesado digital de señales.

- Visualización y manejo de gráficas con datos dinámicos.
- Adquisición y tratamiento de imágenes.
- Control de movimiento (combinado incluso con todo lo anterior).
- Tiempo Real estrictamente hablando.
- Programación de FPGAs para control o validación.
- Sincronización entre dispositivos.

Áreas de aplicación

- Análisis automatizado y plataformas de medida
- Test de fabricación
- Test de validación/medioambiental
- Test mecánico/estructural
- Test de fiabilidad en tiempo real
- Adquisición de datos
- Test de campo portátil
- Test de RF y comunicaciones
- Test en bancos de prueba
- Adquisición de imagen

Medidas industriales y plataformas de control

- Test y control integrado
- Automatización de máquinas
- Visión artificial
- Monitorización de condiciones de máquina
- Monitorización distribuida y control
- Monitorización de potencia

Diseño embebido y plataformas de prototipaje

- Diseño y análisis de sistemas empotrados
- Diseño de control
- Diseño de circuitos electrónicos
- Diseño mecánico
- Diseño de algoritmos

17.- software: Multisim

NI Multisim es un programa de simulación y captura de esquemas electrónicos que forma parte de un conjunto de programas de diseño de circuitos, junto con NI Ultiboard . Multisim es uno de los pocos programas de diseño de circuitos que emplea la simulación de software original basada en Berkeley SPICE. Multisim fue creado originalmente por una empresa llamada Electronics Workbench Group, que ahora es una división de National Instruments. Multisim incluye simulación de microcontrolador (anteriormente conocido como MultiMCU), así como funciones integradas de importación y exportación al software de diseño de placa de circuito impreso en la suite,NI Ultiboard.

Ventajas de NI Multisim

Enseñe una Variedad de Cursos

Utilice un entorno unificado para la enseñanza de electrónica de potencia, analógica y digital.

Comprenda Circuitos Más Rápido

Obtenga resultados más rápido con simulación gráfica e intuitiva.

Utilice una Herramienta Reconocida por la Industria

Estandarice en una herramienta de diseño utilizada en varias industrias.

Aplicaciones Académicas

Los estudiantes pueden usar Multisim para optimizar el rendimiento de su diseño de circuitos y reducir las iteraciones de prototipos en diferentes áreas de aplicación como diseño analógico, electrónica de potencia, energía renovable y diseños completos a nivel del sistema analógico/digital.

Aplicaciones de Diseño de Circuitos

Los ingenieros pueden usar Multisim para optimizar el rendimiento de su diseño de tarjetas de circuito impreso (PCB) y reducir las iteraciones de prototipos en diferentes áreas de aplicación como diseño de circuitos analógicos, electrónica de potencia, energía renovable y simulación completa de sistemas analógicos/digitales con fácil integración de hardware.