Aplicación: Menú con enconder y pantallas LCD

Equipo 3

Abad Dolores Lázaro (Diseño de simulación y Código de Botones)

Rodríguez Hernández Erick Abimael (Desarrollo de código y Prueba física)

Introducción

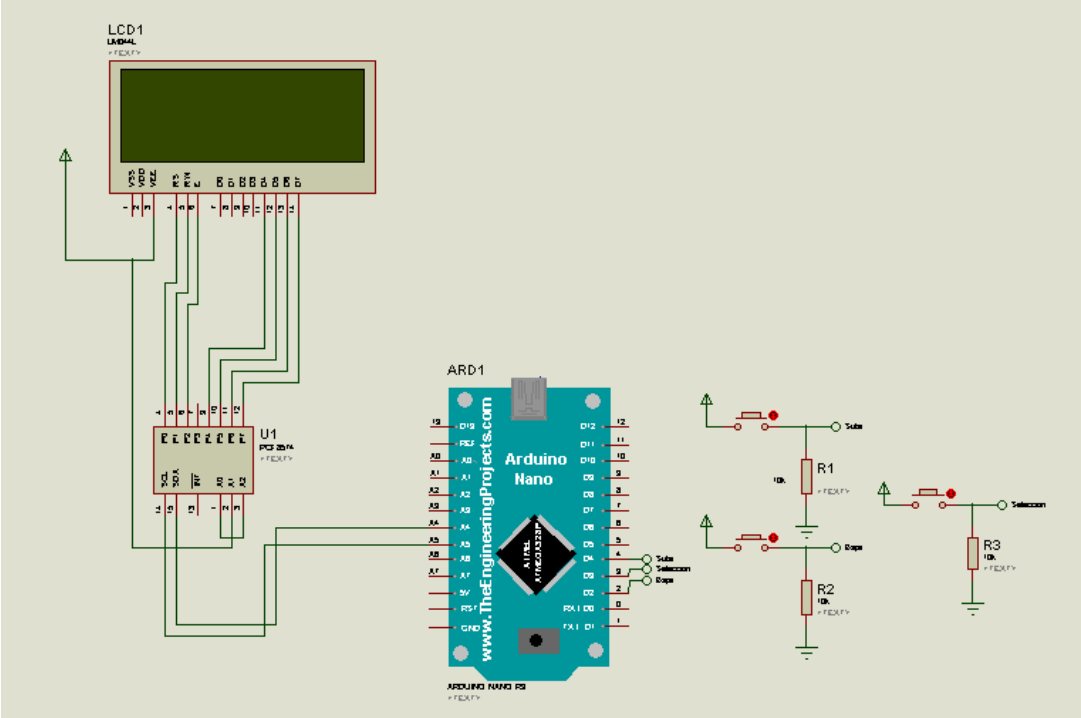
En un sistema embebido suele ser necesaria una interfaz o medio por el cual el usuario pueda establecer una comunicación con el sistema durante la operación, esta interfaz debe ser transparente y precisa. Esta interfaz usualmente suele ser un menú desplegado en una pantalla. Este menú debe dotarnos de la capacidad de navegar fácilmente por una colección de opciones y submenús los cuales puedan ser anidados de manera ordenada.

Componentes requeridos

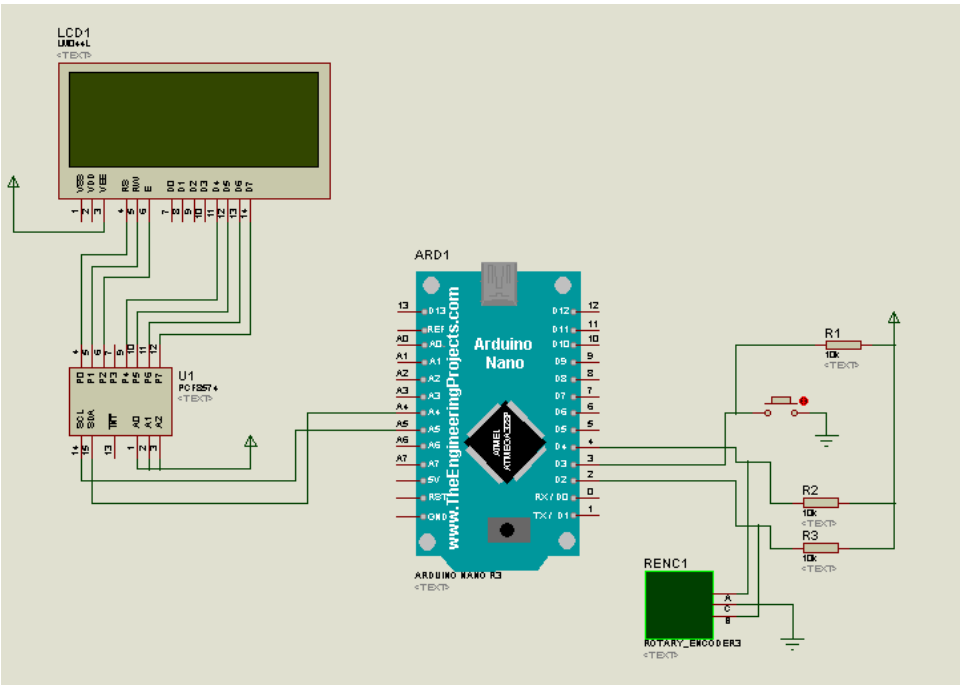
1. Arduino (En este caso UNO)
2. LCD de 20x4
3. Extensor PCF8574
4. Encoder Rotativo
5. -Botones
6. -3 Resistencias (1KΩ)

Diagrama Del Circuito.

El primer circuito base será constituido por un microprocesador (en este caso un Arduino NANO) y tres botones que nos ayudarán a mandar señales para navegar en el menú; para visualizar la información y el menú en sí mismo utilizaremos una pantalla LCD de 4x20 comunicada mediante un extensor PCF8574 el cual permite una transferencia de información por protocolo I2C. Esto nos ayuda a reducir el número de pines utilizados en el Arduino.



El segundo circuito base será constituido por un microprocesador (en este caso un Arduino NANO); un enconder el cual nos servirá como transductor y medio de interacción entre el sistema y el usuario. este enconder brinda dos tipos de señales, una señal de sentido de jiro y una señal de pulso gracias a un botón integrado en el mismo y Para visualizar la información y el menú en si mismo utilizaremos una pantalla LCD de 4x20 comunicada mediante un extensor PCF8574 el cual permite una transferencia de información por protocolo I2C.



Programación y estructuración de código

La organización y navegación por los menús se desarrolló en 4 funciones dentro del código de Arduino.

Void Actualizacion_menu();

Esta función anida los arreglos correspondientes a los diferentes menús o submenús, el acceso a los mismos esta determinado por una dirección guardada en la variable "Estado_menu". Según sea la dirección guardada en esta variable la función determinara que menú o submenú debe desplegar en la pantalla. La organización propuesta para los menús es la siguiente:

- 0 ->menú principal (INICIO)
  - Opción 1
    - Atrás ->te direcciona un nivel más arriba en jerarquía (En este caso menú principal)
    - Opcion11
      - Atrás -> te direcciona un nivel más arriba en jerarquía (Opción 1)
      - Opcion111
      - Opcion112
    - Opcion12
      - Atrás
      - Opcion121
      - Opcion122
    - Opcion13
  - Opción 2
    - Atras
    - Opcion21
    - Opcion22
    - Opcion33
  - Opción 3
  - Opción 4

En la siguiente imagen se observa como se desarrollan los menus y como se anidan dentro de un if el cual decidirá la dirección a la que ingresar.
Las opciones que se desplegaran dentro de un menú serán los componentes del vector "arrayMenu[]" del mismo, vemos como en la dirección 0 ó (Estado_menu == 0) lo que se desplegaran serán una serie de opciones de la 1 a la 6. El tamaño del menú no esta definido o limitado, si es necesario se pueden agregar o eliminar opciones con solo modificar dicho vector.

```
void Actualizacion_menu() {
  if (Bandera_menu == 1) {
    Bandera_menu = 0;
   // wdt_reset();
    if (Estado_menu == 0) {
      String arrayMenu[] = {"Opcion1", "Opcion2", "Opcion3", "Opcion4", "Opcion5", "Opcion6"};
      int size = sizeof(arrayMenu) / sizeof(arrayMenu[0]); //determinacion del tamaño del menu
      Impresion(arrayMenu, size); //Llamada a funcion de impresion

    } else if (Estado_menu == 1) {
      String arrayMenu[] = {"Atras", "Opcion11"};
      int size = sizeof(arrayMenu) / sizeof(arrayMenu[0]);
      Impresion(arrayMenu, size);
    } else if (Estado_menu == 11) {
      String arrayMenu[] = {"Atras", "Opcion111"};
      int size = sizeof(arrayMenu) / sizeof(arrayMenu[0]);
      Impresion(arrayMenu, size);
    } else if (Estado_menu == 111) {
      String arrayMenu[] = {"Atras"};
      int size = sizeof(arrayMenu) / sizeof(arrayMenu[0]);
      Impresion(arrayMenu, size);
```

Del mismo modo si es necesario agregar mas menús basta con replicar o dentro de la función el siguiente segmento de código:

```
else if (Estado_menu == ???) {
  String arrayMenu[] = {"Atras", "OpcionA"};
  int size = sizeof(arrayMenu) / sizeof(arrayMenu[0]);
  Impresion(arrayMenu, size);
}
```

A esta rutina solo le es necesario modificar la dirección apuntada en el Estado_menu siendo precavidos de no repetir una dirección anterior (se sugiere seguir la organización propuesta para evitar confusiones) y modificar el array a conveniencia.

void encoder_boton()

Esta función regula el estado de la variable Estado_menu, es decir, regula los menús que se desplegaran y como se coordinaran unos con otros. Esto lo logrará a partir de considera dos cosas, que menú se encuentra desplegado actualmente en la pantalla y en que opción tenemos colocado el cursor al momento presionar el botón.

La variable Estado_anterior_menu contiene la dirección del menú que se encuentra desplegado en el momento en que se pulso el menú. Por otro lado, la posición del cursor viene dada por la variable posición_encoder, esta variable es el acumulado del contador del encoder.

La función está estructurada mediante un compendio de switch. Un switch funciona con la variable Estado_anterior_menu y te direcciona al case correspondiente a la dirección del menú que tengas desplegado en ese instante. Estando en el case correspondiente entraremos a un nuevo switch, esta vez con la variable posicion_encoder como referencia para saber en que posicion tenemos el cursor, este switch tendrá tantos case

como opciones tenga su menu y cada case nos actualizara la variable Estado_menu con la dirección del nuevo menu a desplegar según la opción seleccionada.

Al final de esta función actualizaremos la variable Estado_anterior_menu con el nuevo menu recién desplegado, reniciaremos el contador del encoder (posicion_encoder = 0) y daremos un 1 en Bandera_menu para notificar que debemos actualizar la pantalla con la función Actualizacion_menu()

```
void encoder_boton() {
  static unsigned long ultimaInterrupcion_B = 0;  // variable static con ultimo valor de
  unsigned long tiempoInterrupcion_B = millis();  // variable almacena valor de func. millis
  if (tiempoInterrupcion_B - ultimaInterrupcion_B > 5) {
  // Serial.println(" INTERRUPCION ENCODER");
    switch (Estado_anterior_menu) {
      case 0:
        switch (posicion_encoder) {
          case 0:
            Estado_menu = 1;
            break;
          case 1:
            Estado_menu = 2;
            break;
          case 2:
            Estado_menu = 3;
            break;
          case 5:
            Estado_menu = 6;
            break;
        }
        break;

      case 1:
        switch (posicion_encoder) {
          case 0:
            Estado_menu = 0;
            break;
          case 1:
            Estado_menu = 11;
            break;
```

```
    }
    break;
  case 11:
    switch (posicion_encoder) {
      case 0:
        Estado_menu = 1;
        break;
      case 1:
        Estado_menu = 111;
        break;
    }
    break;
  case 111:
    switch (posicion_encoder) {
      case 0:
        Estado_menu = 11;
        break;
    }
    break;

  case 2:
    switch (posicion_encoder) {
      case 0:
        Estado_menu = 0;
        break;
      case 1:
        Estado_menu = 21;
        break;
    }
    break;
```

void Impresion(String *arrayMenu, int size);

Esta rutina es llamada cada que es necesaria la actualización de la pantalla, sea por una modificación en el cursor o por la actualización de menús.

La función despliega los componentes de un array en los diferentes renglones de la pantalla, al mismo tiempo es capaz de desplazar el menú en caso de tener el cursor en una opción mayor a las que sean posibles imprimir en la pantalla, en dicha situación recorrerá todo el menú n posiciones para ajustarse a lo solicitado. No es necesaria la modificación de la función al modificar los menús.

```
//FUNCION QUE IMPRIME EL MENU CORRESPONDIENTE
void Impresion( String *arrayMenu, int size) {
  lcd.clear();
  lcd.setCursor(0, 0);
  posicion_encoder = min(size, max(0, posicion_encoder));
  if (posicion_encoder <= 3) {      //Despliega las primeras  opciones del menu
    for (int x = 0; x < size && x <= 3 ; x++) //
    {
      lcd.setCursor(2, x);
      lcd.print(arrayMenu[x]);
    }
    lcd.setCursor(0, posicion_encoder); //ubica el cursor en la posicion correspondiente al encoder
    lcd.print(">");     //imprime el cursor
  } else {
    int menu_Extra = posicion_encoder - 3;     // si la opcion buscada es superior a la 5 despliega las correspondientes
    for (int x = menu_Extra; x < size && x <= (3 + menu_Extra) ; x++) //
    {
      lcd.setCursor(2, x - menu_Extra);
      lcd.print(arrayMenu[x]);
    }
    lcd.setCursor(0, 3);
    lcd.print(">");
  }
}
```

void encoder();

Esta interrupción corresponde a la lectura del jiro del encoder, internamente lleva un contador acumulado el cual aumenta o decrementa según el sentido de jiro y la cantidad de pasos aplicados. La variable posicion_encoder almacena este contador.
Este contador es utilizado como referencia para saber la posición del cursor en el menú correspondiente y cada que ingresamos a un menú diferente reiniciamos la cuenta para entrar con el cursor posicionado al inicio del menú. Dentro de la función contamos con una bandera que se disparará cada que se entre a la función, esta bandera regula la entrada a la función de actualización del menú. Es decir, cada que se dispare la bandera se actualizara el menú.

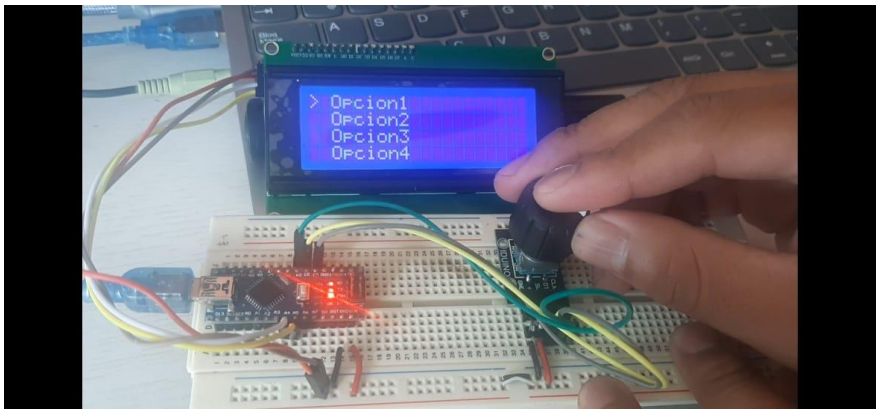
La función cuenta con un algoritmo anti-revote para aumentar la precisión del jiro.

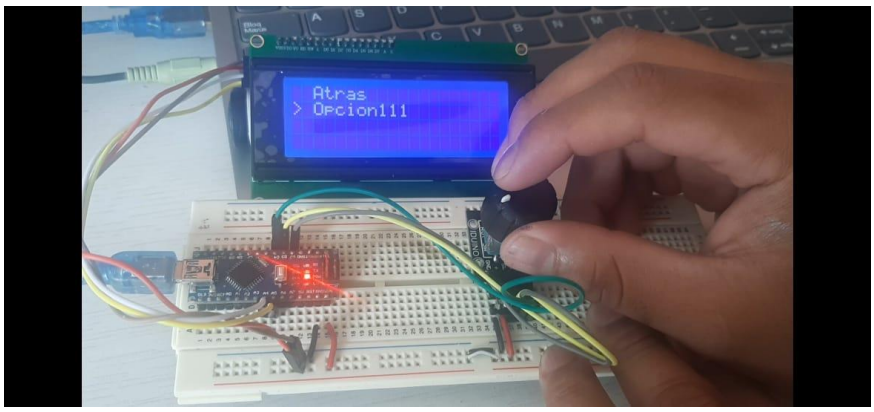
```
//FUNCION QUE CAPTURA EL GIRO DEL ENCODER
void encoder()  {                //Funcion que determina el sentido del giro del encoder
  static unsigned long ultimaInterrupcion = 0;  // variable static con ultimo valor de
  unsigned long tiempoInterrupcion = millis();  // variable almacena valor de func. millis

  if (tiempoInterrupcion - ultimaInterrupcion > 5) {  // No lee posibles rebotes en pulsos menores a 5 mseg.
    if (digitalRead(CLK) == HIGH)     // si CLK es HIGH, rotacion a la derecha
    {
      posicion_encoder-- ;        // incrementa posicion del menu en 1
    } else {                      // si CLK es LOW, rotacion a la izquierda
      posicion_encoder++ ;        // decrementa posicion del menu en 1
    }
    posicion_encoder = min(100, max(0, posicion_encoder));  // numero de opciones del Menu, inferior 0 superior en 5
    ultimaInterrupcion = tiempoInterrupcion;  //actualiza valor del tiempo tiempo
  }
  Bandera_menu = 1;  //bandera de actualizacion de menu

}
```

Funcionamiento en físico





Conclusión

Es de suma importancia para algunos dispositivos contar con un medio de comunicación directa con el usuario, por tales motivos la implementación de menús es algo necesario y debemos ser capases de desarrollarlos de manera óptima, intuitiva y confiable. Esta practica fue sumamente ilustrativa para comprender como se debe estructurar un menú cumpliendo todos estos requerimientos y nos abre la posibilidad de crear sistemas mas complejos con aplicaciones mas reales en un futuro cercano.