Reloj digital sencillo.

Introducción

Hay sencillos aparatos que a veces te sorprenden su funcionamiento. Un reloj es como algo mágico, que parece dominar el tiempo. Sin duda, un invento imprescindible en al Historia de la Humanidad.

Este ejemplo de Arduino, fantástico como proyecto de fin de aprendizaje con la placa, es a la vez sencillo y efectivo. Sus resultados deben ser inmediatos y ofrecer una alta satisfacción al alumno.

Nota: este proyecto es simplemente el esbozo de una idea. No está testado ni montado físicamente en placa.

Competencias Básicas

1. Competencia matemática: medición del tiempo; reflexión sobre distintos ritmos y frecuencias.
2. Competencia lingüística: sintaxis de programación; expresión de ideas.
3. Competencia TIC: instalación de programas; procedimientos de programación.
4. Aprender a Aprender: búsqueda de referencias del lenguaje; imaginación para desarrollar circuitos.

Descripción del circuito electrónico

Como sabemos Arduino Uno tiene una capacidad de 14 entradas o salidas digitales, algunas de las cuales pueden ser moduladas analógicamente, más 6 entradas analógicas. Nuestros displays de 7 segmentos necesitan conectarse a 7 salidas digitales, numeradas de la “a” a la “g”, que activen cada led del display más un ánodo común que, conectado a través de una resistencia, lleve la intensidad a tierra.

Un reloj que muestre horas y minutos necesitaría cuatro displays de este tipo, con lo cual hablamos de 28 conexiones, el doble de las que puede ofrecer Arduino. Además, conectaríamos a través de una resistencia pequeña (unos 100Ω – 500Ω) cada ánodo a tierra.

Sin embargo si los ánodos no son conectados a tierra, sino a Vcc (tensión de alimentación), no circulará intensidad por el diodo y no se iluminarán. Este hecho puedo usarlo para reducir el conexionado de los displays a Arduino activando los cuatro displays en secuencia.

Efectivamente, conectando siete salidas digitales de Arduino a los displays, en paralelo, sólo se activará el display que en ese momento tenga el ánodo conectado a tierra. Así que, además de esas siete salidas digitales, necesito conectar cuatro salidas a cada ánodo de cada display y enviar en secuencia un cero lógico a la salida que quiero activar (podríamos decir que esas cuatro salidas son activas por lógica inversa). Dando un barrido del primer al cuarto dígito, del primer al cuarto display, si la velocidad de cambio es lo suficientemente lenta para activar los leds del display y lo suficientemente rápida para que el ojo humano no distinga el parpadeo podremos ofrecer los cuatros dígitos a la vez.

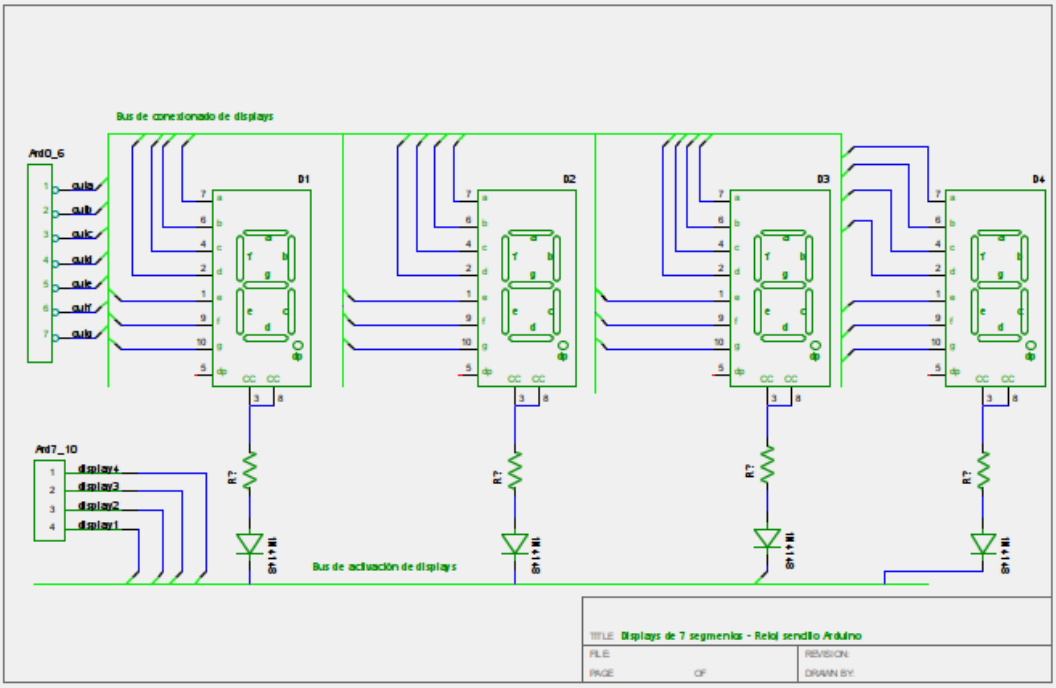
Y habremos usado, en total, 11 salidas digitales. Con dos pines más, esta vez configurados como entradas digitales, y conectados a pulsadores, podremos además modificar los valores ofrecidos por los dígitos de la hora y de los minutos. También puedo contemplar la posibilidad de usar dos potenciómetros conectados a entradas analógicas para el ajuste de la hora en el reloj.

Por último, y como medida de protección, es conveniente acompañar la resistencia del ánodo con un diodo tipo 1N4148 de baja tensión de activación. Esto evitaría posibles corrientes inversas del ánodo (cuando se conecta a 5V) con los cátodos que puedan estar a cero. Teóricamente no deberían

activarse los tramos del display por esta circunstancia, ya que son leds y por tanto diodos, pero no está de más asegurar que esta posibilidad no ocurra.

Montaje

Esquema del montaje de los displays para arduino es el siguiente:



Programación

Una vez realizado el montaje tenemos que realizar el programa que cuente el tiempo, mande las señales correctas a los displays y realice la secuencia de activación. Un ejemplo de programa, que aún tiene que ser comprobado, sería el siguiente (con comentarios en el mismo):

```
// siete salidas digitales a cada tramo del display

int outa = 0;
int outb = 1;
int outc = 2;
int outd = 3;
int oute = 4;
int outf = 5;
int outg = 6;

// a los anodos de los displays. Actuan por logica inversa
int anodo[4]={7,8,9,10};
```

```
 // entradas para modificar la hora y minuto
 int inhoraria = 11;
 int inminutero = 12;

 // cuenta el tiempo
 long tiempo = 0; //maximo 86400000. Cuenta en milisegundos el tiempo transcurrido en un dia
 int hora = 0; // de 0 a 23.
 int minuto = 0; // de 0 a 60
 int horaria = 0; // variable para detectar si cambia la hora
 int minutero = 0; // variable para detectar si cambia el minutero

 // variables de los displays
 int displays[4]; // variable para cada display
 int cualactivo = 0 ; //variable para guardar cual se activa

 /* ================= */
 /* Funciones y subrutinas   */
 /* ================= */

 void mandasenal(int numdisplay, int valor) {
  // recibe el numero de display y el valor que tiene que entregar
  // a) activa cada tramo del display segun valor
  switch (valor) {
    case 1:
      digitalWrite(outa,LOW);
      digitalWrite(outb,HIGH);
      digitalWrite(outc,HIGH);
      digitalWrite(outd,LOW);
      digitalWrite(oute,LOW);
      digitalWrite(outf,LOW);
      digitalWrite(outg,LOW);
    break;
    case 2:
      digitalWrite(outa,HIGH);
      digitalWrite(outb,HIGH);
      digitalWrite(outc,LOW);
      digitalWrite(outd,HIGH);
      digitalWrite(oute,HIGH);
      digitalWrite(outf,LOW);
      digitalWrite(outg,HIGH);
    break;
    case 3:
      digitalWrite(outa,HIGH);
      digitalWrite(outb,HIGH);
      digitalWrite(outc,HIGH);
      digitalWrite(outd,HIGH);
      digitalWrite(oute,LOW);
      digitalWrite(outf,LOW);
      digitalWrite(outg,HIGH);
    break;
```

```
case 4:
  digitalWrite(outa,LOW);
  digitalWrite(outb,HIGH);
  digitalWrite(outc,HIGH);
  digitalWrite(outd,LOW);
  digitalWrite(oute,LOW);
  digitalWrite(outf,HIGH);
  digitalWrite(outg,HIGH);
break;
case 5:
  digitalWrite(outa,HIGH);
  digitalWrite(outb,LOW);
  digitalWrite(outc,HIGH);
  digitalWrite(outd,HIGH);
  digitalWrite(oute,LOW);
  digitalWrite(outf,HIGH);
  digitalWrite(outg,HIGH);
break;
case 6:
  digitalWrite(outa,LOW);
  digitalWrite(outb,LOW);
  digitalWrite(outc,HIGH);
  digitalWrite(outd,HIGH);
  digitalWrite(oute,HIGH);
  digitalWrite(outf,HIGH);
  digitalWrite(outg,HIGH);
break;
case 7:
  digitalWrite(outa,HIGH);
  digitalWrite(outb,HIGH);
  digitalWrite(outc,HIGH);
  digitalWrite(outd,LOW);
  digitalWrite(oute,LOW);
  digitalWrite(outf,LOW);
  digitalWrite(outg,LOW);
break;
case 8:
  digitalWrite(outa,HIGH);
  digitalWrite(outb,HIGH);
  digitalWrite(outc,HIGH);
  digitalWrite(outd,HIGH);
  digitalWrite(oute,HIGH);
  digitalWrite(outf,HIGH);
  digitalWrite(outg,HIGH);
break;
case 9:
  digitalWrite(outa,HIGH);
  digitalWrite(outb,HIGH);
  digitalWrite(outc,HIGH);
  digitalWrite(outd,HIGH);
  digitalWrite(oute,LOW);
  digitalWrite(outf,LOW);
```

```
      digitalWrite(outg,HIGH);
    break;
  }
  // b) activo un solo display cada vez
  for (int i=0;i<=3;i++) { // recorrido de 0 hasta 3
    if (i==numdisplay) {
       digitalWrite(anodo[i],LOW); // activo por logica inversa. En el pin almacenado por el array 'anodo' posicion 0 es donde se pone el cero
    } else {
      digitalWrite(anodo[i],HIGH); // desactivo por logica inversa
    } // fin del if
  } // fin del for
}

/* ============================================== */
/* Bucles principales del programa: loop y setup */
/* ============================================== */

void setup () { // inicializo
  pinMode(outa, OUTPUT);
  pinMode(outb, OUTPUT);
  pinMode(outc, OUTPUT);
  pinMode(outd, OUTPUT);
  pinMode(oute, OUTPUT);
  pinMode(outf, OUTPUT);
  pinMode(outg, OUTPUT);
  pinMode(anodo[0], OUTPUT);
  pinMode(anodo[1], OUTPUT);
  pinMode(anodo[2], OUTPUT);
  pinMode(anodo[3], OUTPUT);
  pinMode(inhoraria, INPUT);
  pinMode(inminutero, INPUT);
}

void loop () { // bucle

 // 1) comprueba horaria
  horaria = digitalRead(inhoraria); //Lee la entrada digital
  if (horaria==true) { //compara
      delay(50); // tiempo de espera 50 milisegundos de pulsacion de boton. El boton debe mantenerse pulsado 50 milisegundos
      // hacerlo de esta manera evita que una pulsacion fisica sea reconocida como muchas pulsaciones
    // ajustar el retraso por cada boton. Se ha puesto un valor de 50 pero podria tener que ser mas
    if (horaria==digitalRead(inhoraria)) {
    tiempo = tiempo + 3600000; // suma  3600 segundos, o sea, una hora
      if (tiempo>=86400000) {tiempo=tiempo-86400000;} // si me paso del tiempo de un dia, resta 86400 seg
    }
  }
```

```
   // 2) comprueba minutero
   minutero = digitalRead(inminutero); //Lee la entrada digital
   if (minutero==true) { //compara
       delay(50); // tiempo de espera 50 milisegundos de pulsacion de boton. El boton debe mantenerse pulsado 50 milisegundos
     if (minutero==digitalRead(inminutero)) {
     tiempo = tiempo + 60000; // suma  60 segundos, o sea, un minuto
       if (tiempo>=86400000) {tiempo=tiempo-86400000;} // si me paso del tiempo de un dia, resta 86400 seg
     }
   }

   // 3) hora y minuto. Tiempo en los displays
  hora = tiempo / 3600000; // indica la hora
  minuto = (tiempo % 3600000) / 60000; // el resto del de la hora, son los segundos que se pasa de esta. Dividido entre 60 nos da los minutos
   // de izquierda a derecha, el primer display corresponde a las decenas de las horas, el segundo a las unidades de las horas,
  // el tercero a las decenas del minutero y el cuarto a las unidades del minutero
  displays[0]= hora/10;
  displays[1]= hora % 10; // resto de la division entre 10
  displays[2]= minuto / 10:
  displays[3]= minuto % 10: // resto de la division entre 10

  // 4) activa uno de los displays
  cualactivo=tiempo % 100; // el resto de la division entre 100 es un numero de 0 a 99. Cada 100 milisegundos se repite...
  // ...la secuencia de 0 a 99 y empieza de nuevo.
   cualactivo = cualactivo / 25; // en esos 100 milisegundos, la variable 'cualactivo' vale 0 los primeros 25 milisegundos, 1, 2 y 3 los
   // ultimos 25 milisegundos... Asi activare cada display en secuencia 25 milisegundos cada uno.
  mandasenal(cualactivo,displays[cualactivo]); // activa los pines del display elegido

  // comando de espera
  delay(1); // espera cada vez un milisegundo.

  // el bucle finaliza mandando un milisegundo mas al contador
  tiempo = (tiempo+1)*(tiempo<86400000) //si es mayor o igual que 86400000 se pone a cero.
 }
 // Fin del programa
```

Comentarios

El programa principal básicamente comprueba si se están pulsando los pulsadores de las horarias y el minutero. Si así es, suma una hora o un minuto al tiempo, restándose 86400000 milisegundos si me paso de esa cantidad. Esto ajustaría la hora del reloj.

Posteriormente calcula los valores que hay que poner en cada display, calculando previamente la variable hora y la variable minuto según los milisegundos recorridos.

Calcula después, cada milisegundo, a que display corresponde encenderse y pasa a la subrutina

“**mandasenal”** el display que toca encenderse y con qué valor. Cada display se enciende a intervalos de 25 milisegundos. Una de las cosas que habría que comprobar es si este régimen de cambio de display es suficiente para engañar al ojo humano o se necesita un régimen más rápido o más lento, para lo cual habría que modificar los valores en la programación.

Por último espera un milisegundo y añade 1 al contador general del tiempo, reinicializándolo a cero si se pasa de 8640000000 milisegundos.

La subrutina “**mandasenal”** se encarga de activar los tramos del display según el valor (código siete-segmentos) y elegir qué display es el que se activa mandando un “cero” al mismo.