Ismael Fernández de la Cotera

Mateo Agustoni Goycoa

Descripción del trabajo

Nuestro proyecto consiste en un marcador de baloncesto con las funciones básicas que poseería uno oficial. Entre ellas encontramos el tanteo de puntos, el número de faltas de cada equipo, control del tiempo (a reloj parado), bocina y un registro donde quedan guardados los resultados de cada partido disputado. Este podrá ser alimentado bien desde un ordenador, mediante la conexión serial, o mediante una toma de 230V con enchufe tipo C o F.

Para sus funcionalidades hemos decidido emplear unos sencillos controles usando pulsadores. Consta de:

- Botones centrales (VERDE):
  Son los encargados de parar, reanudar y resetear el tiempo, además de hacer sonar la bocina a petición del usuario.

- Botones laterales (ROJO):
  Se encuentran repartidos entre los dos equipos por igual, permiten llevar el tanteo (+1 y -1) y el número de faltas de cada equipo.

- Botón inferior (ROJO)
  Solo se podrá activar al final de un partido. Al hacerlo, mostrará en el display los resultados de los partidos ya jugados, que han quedado guardados en una estructura dinámica.

- Interruptor ON/OFF
  Situado en la esquina superior derecha. Regula el paso de la corriente eléctrica desde un enchufe hasta el aparato.

Toda la información necesaria para el correcto desarrollo de un partido se mostrará según la siguiente distribución:

El tiempo restante del actual cuarto (4 cuartos por partido), junto con el periodo de juego en el que se encuentra se encontrará en la parte central del display superior mientras que los puntos obtenidos estarán en los laterales del mismo, justo debajo de la letra identificativa de cada equipo (A/B). Por último, las faltas cometidas por cada equipo en un mismo cuarto se expondrán en los LEDs situados justo bajo las inscripciones “Team A” y “Team B”, refiriéndose evidentemente al equipo correspondiente. El número de infracciones se representará según la numeración ascendente 0-5 sobre cada LED independiente. Al finalizar un partido, será necesario pulsar el botón CONT (continue), tras lo que se mostrarán los resultados registrados en la memoria uno a uno.

Como se puede observar, el proyecto está construido a partir de una placa Az-Delivery UNO (equivalente a Arduino UNO).

Resumen y explicación del código

En primer lugar, se incluyen únicamente las librerías necesarias para el control del display, pues las librerías básicas como puede ser “Malloc.h” están implícitas en la propia placa.

Tras esto se define la estructura “partido”, que será la encargada de guardar los resultados de partidos ya jugados mediante asignación dinámica de memoria, y las funciones que se utilizarán para dar vida al proyecto, cuyo cuerpo se encuentra al final:

- Bocina: Hará sonar el zumbador incorporado el tiempo definido TBOCINA
- Encender y Apagar: Proveerán o privarán, respectivamente, de corriente al pin seleccionado en la llamada a la función.
- FinCuarto y FinPartido: Se encargan de llevar a cabo todas las operaciones necesarias al terminar un cuarto o el partido, como pueden ser reiniciar puntos, faltas, tiempo…
- Save: La responsable de guardar los resultados de cada partido al final del mismo, además de mostrarlos tras ello.

Inmediatamente después encontramos la función setup, aquella que se ejecuta una única vez y en la que se declara lo necesario para el funcionamiento del programa, en este caso el modo de cada pin utilizado.

Lo siguiente será la función loop, que se ejecuta en bucle mientras la placa esté funcionando. En ella encontramos, primeramente, algunas asignaciones necesarias estrictamente dentro de esta función, donde se van a usar. La mayoría de las variables son estáticas (static), para conservar su valor entre repeticiones del bucle loop. A partir de aquí veremos principalmente 4 partes bien diferenciadas: Tiempo, Puntos, Faltas y Display. Dentro de cada uno se llevan a cabo las labores básicas correspondientes a la función que su propio nombre indica, como pueden ser la lectura de los botones necesarios o el encendido de pines.

Cabe destacar dentro de este loop el uso de la técnica denominada “Charlieplexing”, por la cual hemos sido capaces de manejar los 12 LEDs utilizados para indicar las faltas de equipo mediante tan solo 6 pines. La técnica consiste en montar un circuito con 3 pines y 6 LEDs conectando cada uno de los pines entre sí y en ambas direcciones. Uno de los pines se colocara en modo “INPUT”, que define un estado de alta impedancia, por lo que la rama que concierne a dicho pin quedará inactiva. Los dos restantes estarán en modo “OUTPUT” dando 5V y 0V respectivamente, generando así la diferencia de potencial que hará al LED correspondiente iluminarse. Este proceso ha sido repetido para los dos equipos, encendiéndose así tan solo uno de los LEDs en cada lado, el indicador del número de faltas cometidas entre 0 y 5 (bonus).

Concluiré con una breve explicación de la función Save, la más compleja de todas las funciones propias utilizadas. En ella se diferencian 2 partes, una primera en la que se guarda el resultado de los partidos jugados, y una segunda donde se hace que estos salgan por el display, para lo que ha sido necesario pasar por referencia la clase "display".

Para poder guardar dichos resultados se utilizan dos vectores de estructuras cuya dimensión aumenta a cada partido que se juega: "log", el principal, y "aux", un vector auxiliar donde se guardarán los datos ya obtenidos para poder utilizar la función realloc con el vector principal y guardar así el nuevo resultado.

Al acabar un partido y finalizar el proceso de guardado, el marcador esperará a que el usuario pulse el botón CONT y pasará a mostrar los datos de dicho vector de estructuras, "log", componente a componente tras cada nueva pulsación del botón. Para acabar, se reiniciará el contenido del display, las faltas, los puntos y todo lo relativo al juego.

*Normalmente el tiempo de cada cuarto comenzará en los 10 minutos reglamentarios (600s) disminuyendo hasta 0 y haciendo sonar la bocina cuando el crono esté activado, sin embargo, hemos situado el tiempo en 5 segundos y el número de cuarto en 4 para poder realizar la presentación de una forma rápida y comprobar que todo funciona como se espera. *

El código

```
#include <Adafruit_SSD1306.h>
#include <splash.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SPITFT.h>
#include <Adafruit_SPITFT_Macros.h>
#include <gfxfont.h>
#include <Wire.h>
#include <SPI.h>
#include <stdlib.h>

#define TCUARTO 5 //s
#define TBOCINA 2 //s
#define DELAY 350 //ms
#define CUARTO 4

typedef struct {
	int puntosA, puntosB;
}partido;

void Bocina();
void Encender(int);
void Apagar(int);
void FinCuarto(boolean *crono, byte *faltasA, byte *faltasB, float *time, byte
*cuarto, boolean *faltasAFl, boolean *faltasBFl, boolean *dispCuartoFl);
```

```
void FinPartido(byte *cuarto, int *puntosA, int*puntosB, boolean *dispPuntosFl,
boolean *dispCuartoFl, Adafruit_SSD1306 *display, boolean *displayFl);
void Save(int puntosA, int puntosB, Adafruit_SSD1306 *display, boolean
*displayFl, boolean *dispPuntosFl, boolean *dispCuartoFl);

void setup(){

	pinMode(2, INPUT); //Reset
	pinMode(3, INPUT); //Falta A
	pinMode(4, INPUT); //Falta B
	pinMode(11, OUTPUT); //Bocina auto
	pinMode(12, INPUT); //Continuar / mostrar siguiente partido
	pinMode(13, INPUT); //Play / pause
	pinMode(A0, INPUT); //Puntos A++
	pinMode(A1, INPUT); //Puntos A--
	pinMode(A2, INPUT); //Puntos B++
	pinMode(A3, INPUT); //Puntos B--
	Apagar(11);
}

void loop() {

	//Asignaciones iniciales
	static float time = 0, instI = 0, instF = 0;
	static boolean crono = false, displayFl = true, dispPuntosFl = true,
dispCuartoFl = true, faltasAFl = true, faltasBFl = true;
	static int puntosA = 0, puntosB = 0, tCuarto;
	static byte cuarto = CUARTO, faltasA = 0, faltasB = 0;
	static Adafruit_SSD1306 display;

	if (displayFl == true) { //Inicio del display

		delay(100);
		display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
		display.clearDisplay();
		display.display();
		display.setTextSize(2);
		display.setTextColor(WHITE);
		display.setRotation(0);
		display.setTextWrap(false);
		display.dim(0);
		display.setCursor(10, 10);
		display.print("A");
		display.setCursor(110, 10);
		display.print("B");
		displayFl = false;
	}

	//Tiempo
	if (digitalRead(13) == HIGH && crono == false) { //Play time

		crono = true;
		delay(DELAY);
	}
	if (digitalRead(13) == HIGH && crono == true) { //Pause time

		crono = false;
		delay(DELAY);
	}

	if (digitalRead(2) == HIGH) { //Reset time
```

```
        time = 0;
        crono = false;
}

instI = instF;
instF = millis() / 1000;
tCuarto = TCUARTO - time;
if (crono == true) //Cuenta tiempo (si activado)
        time += instF - instI;

//Puntos equipo A
if (digitalRead(A0) == HIGH) { //Puntos +1

        puntosA++;
        delay(DELAY);
        dispPuntosFl = true;
}
if (digitalRead(A1) == HIGH) { //Puntos -1

        puntosA--;
        delay(DELAY);
        dispPuntosFl = true;
}
if (puntosA < 0)
        puntosA = 0;

//Puntos equipo B
if (digitalRead(A2) == HIGH) { //Puntos +1

        puntosB++;
        delay(DELAY);
        dispPuntosFl = true;
}
if (digitalRead(A3) == HIGH) { //Puntos -1

        puntosB--;
        delay(DELAY);
        dispPuntosFl = true;
}
if (puntosB < 0)
        puntosB = 0;

//Faltas equipo A
if (digitalRead(3) == HIGH) { //Faltas +1

        faltasA++;
        delay(DELAY);
        faltasAFl = true;
}

if (faltasAFl == true) { //LEDs faltas (tecnica charlieplexing)
        if (faltasA == 0) {
                pinMode(5, OUTPUT);
                Encender(5);
                pinMode(6, OUTPUT);
                Apagar(6);
                pinMode(7, INPUT);
                delay(DELAY);
                faltasAFl = false;
        }
        if (faltasA == 1) {
                pinMode(7, OUTPUT);
```

```
                Encender(7);
                pinMode(5, OUTPUT);
                Apagar(5);
                pinMode(6, INPUT);
                delay(DELAY);
                faltasAFl = false;
        }
        if (faltasA == 2) {
                pinMode(6, OUTPUT);
                Encender(6);
                pinMode(7, OUTPUT);
                Apagar(7);
                pinMode(5, INPUT);
                delay(DELAY);
                faltasAFl = false;
        }
        if (faltasA == 3) {
                pinMode(5, OUTPUT);
                Encender(5);
                pinMode(7, OUTPUT);
                Apagar(7);
                pinMode(6, INPUT);
                delay(DELAY);
                faltasAFl = false;
        }
        if (faltasA == 4) {
                pinMode(6, OUTPUT);
                Encender(6);
                pinMode(5, OUTPUT);
                Apagar(5);
                pinMode(7, INPUT);
                delay(DELAY);
                faltasAFl = false;
        }
        if (faltasA == 5) {
                pinMode(7, OUTPUT);
                Encender(7);
                pinMode(6, OUTPUT);
                Apagar(6);
                pinMode(5, INPUT);
                delay(DELAY);
                faltasAFl = false;
        }
}
if (faltasA >5)
        faltasA = 5;

//Faltas equipo B
if (digitalRead(4) == HIGH) { //Faltas +1

        faltasB++;
        delay(DELAY);
        faltasBFl = true;
}

if (faltasBFl == true) { //LEDs faltas (tecnica charlieplexing)
        if (faltasB == 0) {
                pinMode(8, OUTPUT);
                Encender(8);
                pinMode(9, OUTPUT);
                Apagar(9);
                pinMode(10, INPUT);
```

```
                delay(DELAY);
                faltasBFl = false;
        }
        if (faltasB == 1) {
                pinMode(10, OUTPUT);
                Encender(10);
                pinMode(8, OUTPUT);
                Apagar(8);
                pinMode(9, INPUT);
                delay(DELAY);
                faltasBFl = false;
        }
        if (faltasB == 2) {
                pinMode(9, OUTPUT);
                Encender(9);
                pinMode(10, OUTPUT);
                Apagar(10);
                pinMode(8, INPUT);
                delay(DELAY);
                faltasBFl = false;
        }
        if (faltasB == 3) {
                pinMode(8, OUTPUT);
                Encender(8);
                pinMode(10, OUTPUT);
                Apagar(10);
                pinMode(9, INPUT);
                delay(DELAY);
                faltasBFl = false;
        }
        if (faltasB == 4) {
                pinMode(9, OUTPUT);
                Encender(9);
                pinMode(8, OUTPUT);
                Apagar(8);
                pinMode(10, INPUT);
                delay(DELAY);
                faltasBFl = false;
        }
        if (faltasB == 5) {
                pinMode(10, OUTPUT);
                Encender(10);
                pinMode(9, OUTPUT);
                Apagar(9);
                pinMode(8, INPUT);
                delay(DELAY);
                faltasBFl = false;
        }
}
if (faltasB > 5)
        faltasB = 5;

//Display
if (tCuarto == 600) {//Tiempo

        display.fillRect(35, 15, 70, 25, BLACK);
        display.setCursor(35, 25);
        display.print("10:00");
}
else {

        display.fillRect(35, 15, 70, 25, BLACK);
```

```
        display.setCursor(35, 25);
        display.setTextColor(WHITE);
        display.print("0");
        display.print((int)tCuarto / 60);
        display.print(":");
        if ((tCuarto % 60) < 10) {

                display.print("0");
                display.print((int)tCuarto % 60);
        }
        else {

                display.print((int)(tCuarto % 60));
        }
}

display.display();

if (dispCuartoFl == true) { //Cuarto

        display.fillRect(60, 40, 20, 25, BLACK);
        display.setCursor(60, 40);
        display.print(cuarto);
        display.display();
        dispCuartoFl = false;
}

if (dispPuntosFl == true) { //Puntos A y B

        display.fillRect(5, 45, 25, 25, BLACK);
        if (puntosA < 10) {
                display.setCursor(10, 45);
                display.print(puntosA);
        }
        else {
                if (puntosA < 100) {
                        display.setCursor(5, 45);
                        display.print(puntosA);
                }
                else {
                        display.setCursor(0, 45);
                        display.print(puntosA);
                }
        }

        display.fillRect(105, 45, 25, 25, BLACK);
        if (puntosB < 10) {
                display.setCursor(110, 45);
                display.print(puntosB);
        }
        else {
                if (puntosB < 100) {
                        display.setCursor(105, 45);
                        display.print(puntosB);
                }
                else {
                        display.setCursor(100, 45);
                        display.print(puntosB);
                }
        }
        display.display();
        dispPuntosFl = false;
```

```
	}

	//Final
	if (tCuarto == 0 && crono == true)
		FinCuarto(&crono, &faltasA, &faltasB, &time, &cuarto, &faltasAFl,
&faltasBFl, &dispCuartoFl);
	if (cuarto > 4)
		FinPartido(&cuarto, &puntosA, &puntosB, &dispPuntosFl,
&dispCuartoFl, &display, &displayFl);
}

//Funciones
void Bocina() {
	Encender(11);
	delay(1000 * TBOCINA);
	Apagar(11);
}
void Encender(int Pin) {
	digitalWrite(Pin, HIGH);
}
void Apagar(int Pin) {
	digitalWrite(Pin, LOW);
}
void FinCuarto(boolean *crono, byte *faltasA, byte *faltasB, float *time, byte
*cuarto, boolean *faltasAFl, boolean *faltasBFl, boolean *dispCuartoFl){

	*crono = false;
	Bocina();
	*faltasA = 0;
	*faltasB = 0;
	*time = 0;
	*cuarto = *cuarto + 1;
	*faltasAFl = true;
	*faltasBFl = true;
  if(*cuarto < 5)
	  *dispCuartoFl = true;
}
void FinPartido(byte *cuarto, int *puntosA, int*puntosB, boolean *dispPuntosFl,
boolean *dispCuartoFl, Adafruit_SSD1306 *display, boolean *displayFl) {

	Save(*puntosA, *puntosB, display, displayFl, dispPuntosFl, dispCuartoFl);
	*cuarto = CUARTO;
	*puntosA = 0;
	*puntosB = 0;
	*dispPuntosFl = true;
	*dispCuartoFl = true;
}
void Save(int puntosA, int puntosB, Adafruit_SSD1306 *display, boolean
*displayFl, boolean *dispPuntosFl, boolean *dispCuartoFl) {

	static int cont = 0;
	cont++;

	//Guardar resultados
	partido *log, *aux;
	if (cont == 1) {
		log = (partido *)malloc(sizeof(partido));
	}
	if (cont == 2) {
		aux = (partido *)malloc(sizeof(partido));
		aux[0].puntosA = log[0].puntosA;
		aux[0].puntosB = log[0].puntosB;
```

```
		log = (partido *)realloc(log, 2 * sizeof(partido));
		log[0].puntosA = aux[0].puntosA;
		log[0].puntosB = aux[0].puntosB;
	}
	if(cont>2) {
		aux = (partido*)realloc(aux, (cont - 1) * sizeof(partido));
		for (int i = 0;i < cont - 1;i++) {
			aux[i].puntosA = log[i].puntosA;
			aux[i].puntosB = log[i].puntosB;
		}
		log = (partido *)realloc(log, cont * sizeof(partido));
		for (int i = 0;i < cont - 1;i++) {
			log[i].puntosA = aux[i].puntosA;
			log[i].puntosB = aux[i].puntosB;
		}
	}

	log[cont-1].puntosA = puntosA;
	log[cont-1].puntosB = puntosB;

	//Mostrar resultados guardados por pantalla
	(*display).fillRect(35, 15, 70, 40, BLACK);
	(*display).setCursor(36, 15);
	(*display).print("Pulse");
	(*display).setCursor(41, 35);
	(*display).print("CONT");
	(*display).display();

	int k = 0;
	while (k<cont) {

		if (digitalRead(12) == HIGH) {
			(*display).clearDisplay();
			(*display).setCursor(15,20);
			(*display).print("Partido ");
			(*display).print(k+1);
			if (log[k].puntosA > 9)
				if(log[k].puntosB > 9)
					(*display).setCursor(25, 40);
				else
					(*display).setCursor(30, 40);
			else
				(*display).setCursor(35, 40);
			(*display).print(log[k].puntosA);
			(*display).print(" - ");
			(*display).print(log[k].puntosB);
			(*display).display();
			delay(DELAY);
			k++;
			while (digitalRead(12) != HIGH) {}
		}
	}

	//Flags para reiniciar display
	*displayFl = true;
	*dispPuntosFl = true;
	*dispCuartoFl = true;
}
```