ACTIVIDADES PROGRAMACIÓN CON ARDUINO

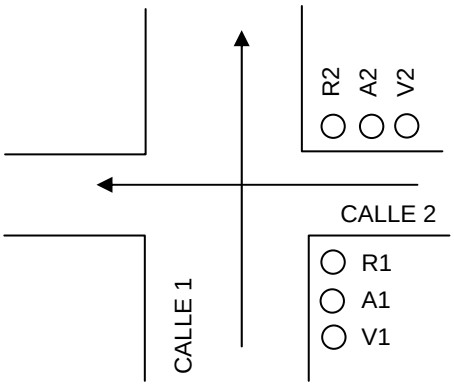
X.1.- A continuación se presenta el programa de ejemplo "Blink", que hace titilar un LED conectado en el pin 0 de la tarjeta Arduino, con una cadencia de 1 segundo encendido y 1 apagado. Edítalo en Arduino, cárgalo en la tarjeta y comprueba que funciona.

```
void setup() {
        pinMode(0,OUTPUT);
}
void loop() {
        digitalWrite(0,HIGH);
        delay(1000);
        digitalWrite(0,LOW);
        delay(1000);
}
```

Tomándolo como referencia, elaborar los programas correspondientes a las siguientes variantes:

a) Hacer que el LED esté permanentemente encendido.
b) Que el LED esté 1 segundo encendido y 0,5 segundos apagado.
c) Disponer dos LEDs, uno verde conectado al pin 0 y otro rojo al 1. El verde debe estar siempre encendido y el rojo 0,8 segundos encendido y 0,5 segundos apagado.
d) Con los dos LED anteriores, que cada uno esté 0,5 segundos encendido y 0,5 segundos apagado pero que cuando uno esté encendido el otro esté apagado.
e) Con tres LED, verde, rojo y amarillo, conectados en los pines 0, 1 y 2, programar el funcionamiento de un semáforo, de forma que el rojo esté encendido 5 segundos, el amarillo 1,5 segundos y el verde 4 segundos.

X.2.- Realizar el programa que controla los dos semáforos de un cruce de calles (ver figura). La calle 1 es la principal por lo que su semáforo verde durará 15 s y el verde de la calle 2 durará 10 s. El ámbar durará 4 s. Habrá un intervalo de seguridad de 2 s desde que se pone en rojo en una calle hasta que se pone en verde en la otra.
Nota: los LEDs rojo, ámbar y verde de la calle 1 se conectarán en los pines 0, 1 y 2 respectivamente. Los LEDs rojo, ámbar y verde de la calle 2 se conectarán en los pines 3, 4 y 12 respectivamente. Tener en cuenta, que los pines 3 y 12 de la placa anexa a la tarjeta Arduino, no llevan conectadas resistencias en serie para proteger los LEDs, por lo que habrá que conectar resistencias de unos 150 a 250 Ω.



X.3.- A continuación se presenta el ejemplo "Button", que hace que un LED verde conectado en el pin 0 se encienda cuando un interruptor conectado al pin 7 se cierra y el LED se apague cuando dicho interruptor se abre. Edítalo, cárgalo en la tarjeta y comprueba que funciona.

```
int pinLED = 0;
int pinInterruptor = 7;
int estadointerruptor = 0;

void setup() {
        pinMode(pinLED, OUTPUT);
        pinMode(pinInterruptor, INPUT);
}
void loop() {
        estadointerruptor = digitalRead(pinInterruptor);
        if (estadointerruptor == HIGH) {
```

```
            digitalWrite(pinLED, HIGH);
        }
        else {
            digitalWrite(pinLED, LOW);
        }
}
```

Tomándolo como referencia, elaborar los programas correspondientes a las siguientes variantes:

a) Se coloca otro LED rojo conectado en el pin 1 que se enciende cuando el verde se apaga y se apaga cuando el verde se enciende.
b) Se conectan, además del LED verde y el interruptor iniciales, un zumbador en el pin 3 y otro interruptor en el pin 8, de forma que con un interruptor se encienda y apague el LED y con el otro se active y desactive el zumbador.
c) Se conecta un motor entre los pines 5 y 6 y dos interruptores en los pines 7 y 8, de forma que con el interruptor del pin 7 se controle la marcha o la parada del motor y con el interruptor del pin 8 el sentido de giro del motor cuando este está en marcha.
d) Se conectan tres LED de diferentes colores en los pines 0, 1 y 2 y un interruptor en el pin 7. Los LED deben funcionar encendiéndose uno tras otro de forma cíclica. Con el interruptor abierto cada LED estará encendido 1 segundo, mientras que con el interruptor cerrado la secuencia será más rápida, estando cada LED encendido sólo 0,5 segundos.
e) Se conecta un LED en el pin 0, un pulsador en el pin 7 y otro en el pin 8. Debe funcionar de modo que cuando se pulse el pulsador del pin 7 el LED se encienda y cuando se pulse el pulsador del pin 8 el LED se apague.
f) Se utilizan un LED conectado en el pin 0 y un único pulsador en el pin 7. Debe funcionar de modo que cada vez que pulsemos el pulsador el LED cambie de estado (se encienda si está apagado y se apague si está encendido). Nota: si se mantiene el pulsador pulsado permanentemente, el LED se quedará estable en el último estado adquirido.

X.4.- Ejemplo resuelto: tenemos un LED, L1, y dos pulsadores, P1 y P2. Al pulsar el pulsador P1 el LED se enciende. Una vez encendido, al cabo de 10 s el LED se apaga solo, pero si antes de que transcurran los 10 segundos pulso el pulsador P2 se apaga el LED. Si antes de pasar los 10 segundos se pulsa el pulsador P1 vuelve a empezar a contar el tiempo. Edítalo y carga el programa en la tarjeta y comprueba que funciona. Trata de comprender lo que hace.
Conexiones: L1 en pin 0, P1 en pin 7 y P2 en pin 8.

```
int pinL1 = 0;
int pinP1 = 7;
int pinP2 = 8;
void setup() {
        pinMode(pinL1, OUTPUT);
        pinMode(pinP1, INPUT);
        pinMode(pinP2, INPUT);
        digitalWrite(pinL1, LOW);
}
void loop() {
        while (digitalRead(pinP1)==LOW) { }
        digitalWrite(pinL1,HIGH);
        //divido los 10 s en 50 trozos de 0,2 s para escasear los pulsadores, no vale delay()
        for (int i=0;i<=50;i++) {
            if(digitalRead(pinP1)==HIGH) i=0;
            if(digitalRead(pinP2)==HIGH) break;
            delay(200);
        }
        digitalWrite(pinL1,LOW);
}
```

X.5.- Diseñar un programa con Arduino que permita controlar un vehículo con dos ruedas motrices, de forma que un interruptor A permita el encendido o apagado de un motor situado en la rueda izquierda, otro interruptor B haga que dicho motor gire en un sentido o en otro. Lo mismo debe ocurrir con otros dos interruptores C y D que controlarán la rueda derecha.

Conexiones: El motor de la rueda izquierda MI, irá conectado en los pines 5 y 6, el motor de la rueda derecha MD en los pines 10 y 11. Los interruptores A y el B en los pines 7 y 8 respectivamente y los interruptores C y D en los pines 9 y 12 respectivamente.

X.6. - Al programa de la actividad 3, hacerle los siguientes añadidos:

- Cuando los dos motores hagan que el vehículo avance hacia delante, deberá encenderse el LED frontal del vehículo.
- Cuando los dos motores hagan que el vehículo avance hacia atrás, deberá encenderse el LED trasero del vehículo.
- Cuando uno de los motores esté activo y el otro inactivo, se encenderá el LED lateral contrario al motor activo del vehículo.

X.7.- Elaborar el programa de control del apagado automático de las luces de escalera de un bloque de viviendas. Al pulsar un pulsador conectado al pin 7, se encenderán dos LEDs (simulan a las luces) conectados en los pines 0 y 1. El LED A (que se supone que simula a la luz de la entrada del bloque) se apagará automáticamente a los 10 s y el LED B (que se supone que simula a las luces de las escaleras) se apagará a los 20 s. En este caso, para simplificar, supondremos que no se vuelve a pulsar el pulsador hasta que no ha terminado el ciclo completo.

Solución: Resolvemos el problema utilizando el **método de asignación de estados**.

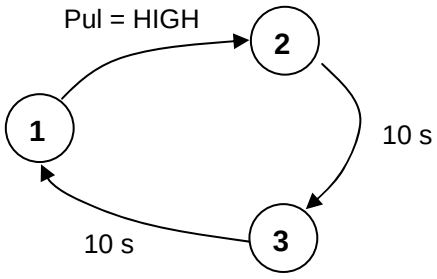
Asignación de estados:

- Estado 1: LedA = LedB = LOW; con Pul = HIGH pasa a estado 2
- Estado 2: LedA = LedB = HIGH; a los 10 s pasa a estado 3
- Estado 3; LedA = LOW, LedB=HIGH; a los 10 s pasa a estado 1

Asignación de estado inicial:

Si Pul = LOW, asigno estado 1, si Pul = HIGH estado 2.

```
int pinLedA = 0;
int pinLedB = 1;
int pinPul = 7;
int estado;
void setup() {
        pinMode(pinLedA, OUTPUT);
        pinMode(pinLedB, OUTPUT);
        pinMode(pinPul, INPUT);
        //leemos entradas para asignar estado inicial
        if(digitalRead(pinPul)==LOW)estado=1;
        else estado=2;
}
void loop() {
        if(estado==1) estado1();
        if(estado==2) estado2();
        if(estado==3) estado3();
}
void estado1(){
        digitalWrite(pinLedA, LOW);
        digitalWrite(pinLedB, LOW);
```



```
        while(digitalRead(pinPul)==LOW) { }
        estado=2;
}
void estado2(){
        digitalWrite(pinLedA, HIGH);
        digitalWrite(pinLedB, HIGH);
        delay(10000);
        estado=3;
}
void estado3(){
        digitalWrite(pinLedA, LOW);
        digitalWrite(pinLedB, HIGH);
        delay(10000);
        estado=1;
}
```

X.8.- Resolver el problema anterior sin utilizar el método de la asignación de estados.

X.9.- Resolver el mismo problema anterior pero teniendo en cuenta que si antes de terminar el ciclo se vuelve a pulsar el pulsador, se reinicia de nuevo. Usar el método de la asignación de estados.

X.10. – Diseñar un programa en Arduino que permita que un vehículo avance hasta que uno de los finales de carrera de los dos que tiene frontales, FCI en el lado izquierdo y FCD en el lado derecho, detecte un obstáculo. En ese momento, el vehículo dará marcha atrás primero un poco con ambos motores, MI y MD y después otro poco con el motor del lado contrario al final de carrera que se haya activado y cuando haya girado unos 90º proseguirá su marcha hacia adelante, respondiendo ante los obstáculos de la manera descrita. EL vehículo dispondrá de un interruptor, INT, para parar o poner en marcha el vehículo. Nota: por facilitar, el interruptor sólo parará el vehículo cuando este esté avanzando, no actuará cuando esté retrocediendo.

Conexiones: FCI en pin 7, FCD en pin 8, INT en pin 9, MI en pines 5 y 6, MD en pines 10 y 11.

X.11. – Realizar un programa en Arduino que permita que un vehículo con dos LDR pueda ir siguiendo una línea negra pintada en el suelo. Incluirá un interruptor de forma que en posición "off" el vehículo se pare y en posición "on" se ponga en funcionamiento

Conexiones: La LDR izquierda se conectará en el pin analógico 0 y la LDR derecha en el pin analógico 1. El motor de la rueda izquierda MI, irá conectado en los pines 5 y 6, el motor de la rueda derecha MD en los pines 10 y 11. El interruptor irá conectado en el pin 7.

X.12.- Elaborar el programa de control del encendido y apagado automático de luces de escalera. Al abrir la puerta de casa (se supone que esto pisa un pulsador conectado a pin 7) se enciende automáticamente un LED conectado en el pin 0 que simula a la luz de la escalera. El LED se mantiene encendido en tanto tenga la puerta abierta (pulsador pisado). Cuando cierro la puerta, el LED permanece encendido 10 s más y después se apaga. Para simplificar, supondremos que no se interrumpe el periodo de espera, es decir, no se vuelve a abrir y cerrar la puerta en el periodo de espera de 10 s.

X.13.- Resolver el mismo caso anterior pero teniendo en cuenta que si antes de terminar el tiempo de espera de 10 s, se vuelve a abrir la puerta, se reinicia de nuevo.

X.14.- Elaborar el programa de control de un sistema formado por dos LEDs, LA y LB, conectados en los pines 0 y 1 y dos pulsadores, P1 y P2, conectados en los pines 7 y 8. Al pulsar P1 se enciende LA, al pulsar P2 se enciende LB y se apagan ambos al cabo de 6 segundos de la segunda

pulsación. Si antes de pulsar el P1 se pulsa el P2 no ocurre nada. Para simplificar, lo haremos de forma que si se vuelve a pulsar el P1 antes de que se hayan apagado los LEDs se hace caso omiso.

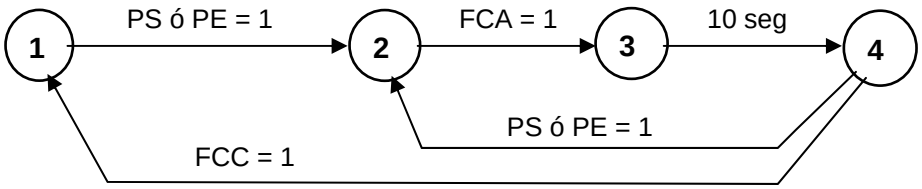
X.15.- Resolver el mismo caso anterior pero teniendo en cuenta que si antes de terminar el tiempo de espera de 6 s, se vuelve a pulsar el pulsador P1, se inicia de nuevo el ciclo, es decir, se enciende sólo LA y queda a la espera de que se pulse P2 para encender LB.

X.16.- Elabora el programa de control de una puerta de garaje de apertura y cierre automático por medio de un motor que gira en ambos sentidos. La puerta es de entrada y salida. Tendrá un pulsador a la entrada y otro a la salida. Al pulsar cualquiera de los pulsadores la puerta se abrirá hasta pisar el final de carrera de "puerta abierta" y al cabo de 10 s abierta se cerrará automáticamente hasta pisar el final de carrera de "puerta cerrada". Los pulsadores de entrada (PE) y salida (PS) irán conectados en los pines 7 y 8, los finales de carrera de "puerta abierta" (FCA) y "puerta cerrada" (FCC) en los pines 9 y 12, y el motor (M) en los pines 5 y 6.

Solución: Utilizamos el método de los estados.

Asignación de estados

- Estado 1: Puerta cerrada. Motor parado, espera PS=1 ó PE = 1 para pasar a estado 2.
- Estado 2: Puerta abriendo. M gira en el sentido de abrir, espera FCA = 1 para pasar a estado 3.
- Estado 3: Puerta abierta. Motor parado, espera 10 seg. para pasar a estado 4.
- Estado 4: Puerta cerrando. M gira en el sentido de cerrar. Espera FCC = 1 para pasar a estado 1 o bien, que PS ó PE se hagan 1 para pasar al estado 2.



Asignación de estado inicial:
Lee FCC y FCA. Si FCC = 1 puerta cerrada, asigno estado 1, si FCA = 1 puerta abierta, asigno estado 3. Si FCC = FCA = 0, puede estar la puerta abriendo o cerrando, asigno estado 4 para que la puerta se cierre.

```
int pinPE = 7;
int pinPS = 8;
int pinFCA = 9;
int pinFCC = 12;
int pinMa = 5;
int pinMb = 6;
int estado;

void setup() {
        pinMode(pinPE, INPUT);
        pinMode(pinPS, INPUT);
        pinMode(pinFCA, INPUT);
        pinMode(pinFCC, INPUT);
        pinMode(pinMa, OUTPUT);
        pinMode(pinMb, OUTPUT);
```

```
        if(digitalRead(pinFCC)==HIGH) estado=1;
        else if(digitalRead(pinFCA)==HIGH) estado=3;
        else estado=4;
}
void loop() {
        if(estado==1) estado1();
        if(estado==2) estado2();
        if(estado==3) estado3();
        if(estado==4) estado4();
}
void estado1() {
        digitalWrite(pinMa, LOW);
        digitalWrite(pinMb, LOW);
        while(digitalRead(pinPE)==LOW && digitalRead(pinPS)==LOW) { }
        estado=2;
}
void estado2() {
        digitalWrite(pinMa, HIGH);
        digitalWrite(pinMb, LOW);
        while(digitalRead(pinFCA)==LOW) { }
        estado=3;
}
void estado3() {
        digitalWrite(pinMa, LOW);
        digitalWrite(pinMb, LOW);
        delay(10000);
        estado=4;
}
void estado4() {
        digitalWrite(pinMa, LOW);
        digitalWrite(pinMb, HIGH);
        while(estado==4) {
                if(digitalRead(pinFCC)==HIGH) estado=1;
                if(digitalRead(pinPE)==HIGH) estado=2;
                if(digitalRead(pinPS)==HIGH) estado=2;
        }
}
```

X.17.- Vamos a realizar una variante con respecto al problema anterior añadiendo una LDR que sea tapada por el coche mientras está pasando justo por la puerta. En este caso la puerta no debe empezar a cerrarse aunque hayan pasado los 10 s (para evitar golpear un coche que, por ejemplo, se haya calado justo al cruzar). La LDR se conectará en el pin analógico 0.

X.18.- Elaborar el programa de control de una carretilla automática que lleva mercancías de un lugar a otro en una fábrica. El funcionamiento será como sigue:
La carretilla inicia su recorrido en la zona de carga (posición que es detectada por el final de carrera FCC). La carretilla dispone de un sensor (P) que se activa al ponerle peso encima (lo simularemos por un final de carrera). Cuando se pone peso encima espera 5 seg. (para dar tiempo al operario que deposita la carga a retirarse de la carretilla para no ser arrollado) y emprende el camino hacia la zona de descarga (posición que es detectada por el final de carrera FCD) donde se para al llegar. Cuando un operario le retira la carga, vuelve a esperar 5 seg. y emprende el camino de regreso hacia la zona de carga. Como medida de seguridad, cuando la

carretilla esté en movimiento se encenderá un juego de pequeñas luces rojas (LR) a lo largo del recorrido y cuando esté parada un juego de luces verdes (LV).

Zona de carga Zona de descarga

Conexiones: El final de carrera P irá conectado al pin 7, los finales de carrera FCC Y FCD irán conectados en los pines 8 y 9 respectivamente. El LED LR irá en el pin 0 y el LED LV en el pin 1. El motor M irá conectado en los pines 5 y 6.

X.19.- Elaborar el programa de control de un sistema de alarma compuesto por una LDR, un LED verde, LV, un LED rojo, LR, un pulsador de activación, PA, un pulsador de desactivación, PD y un zumbador, Z, (en los ensayos de prueba se sustituirá por un LED ámbar, LA, hasta que el programa funcione bien para evitar el excesivo ruido en las pruebas del programa).
El funcionamiento será el siguiente: con la alarma desconectada se encenderá LV; la alarma se conectará pulsando el pulsador PA, encendiéndose LR. Una vez conectada la alarma, si se oscurece la LDR se activará la alarma haciendo sonar el zumbador Z. LR permanece también encendido mientras suena la alarma. La alarma se desconecta o se desactiva (en el caso de que haya saltado) pulsando PD, apagándose LR y el zumbador Z si estaba sonando y volviéndose a encender LV.
Conexiones: la LDR en el pin analógico 0, LR en el pin 1, LV en el pin 2, Z en el pin 3, PA en el pin 7 y PD en el pin 8.
Se recomienda usar el método de los estados.

X.20.- Elabora el programa de control de una puerta de garaje de apertura y cierre automático por medio de un motor que gira en ambos sentidos. El funcionamiento será:
La puerta es de entrada y salida. Tendrá un pulsador a cada lado de la puerta, uno para entrar y otro para salir. Tendrá dos semáforos, uno que controla la entrada y otro la salida, con LEDs rojos y verdes. Mientras no haya vehículos queriendo entrar o salir los semáforos estarán apagados. Cuando un vehículo pulse para salir, la puerta se abrirá y se encenderá el LED rojo del semáforo que controla la entrada (para avisar a los vehículos entrantes que no obstaculicen la salida). Cuando la puerta esté abierta del todo se encenderá el LED verde del semáforo que controla la salida. Al cabo de 10 s la puerta empezará a cerrarse poniéndose ambos semáforos en rojo. Cuando la puerta se cierre totalmente se apagarán los semáforos. Cuando un vehículo pulse para entrar el funcionamiento debe ser similar, es decir, se encenderá el LED rojo del semáforo que controla la salida (para avisar a los salientes que no obstaculicen la entrada), etc.,....
Nota: Los pulsadores de entrada (PE) y salida (PS) irán conectados en los pines 7 y 8, los finales de carrera de "puerta abierta" (FCA) y "puerta cerrada" (FCC) en los pines 9 y 12, el motor (M) en los pines 5 y 6, los LEDs rojo (LRE) y verde (LVE) del lado de entrada en los pines 0 y 1 y los LEDs rojo (LRS) y verde (LVS) del lado de salida en los pines 2 y 3.

X.21.- Elabora el programa de control de un puente levadizo sobre un río cuyo funcionamiento sea el siguiente:
El puente levadizo es movido por un motor que gira en los dos sentidos. Dispone de un semáforo con un LED rojo (LR), un LED verde (LV) y un LED ámbar (LA), que dirigen el paso de los vehículos. Tiene unos detectores, que simularemos por finales de carrera, que detectan la llegada de barcos por el lado izquierdo (FCI) y por el lado derecho (FCD). El puente cuenta también con dos finales de carrera que detectan su posición de bajado (FCB) y su posición de subido (FCS).
Cuando el puente está bajado, el semáforo estará en verde, indicando a los vehículos que pueden pasar por el puente. Cuando se detecte la llegada de un barco por la izquierda o por la derecha del río el semáforo pasará a ámbar y a los 5 s el semáforo pasará a rojo y el puente

empezará a subir. Cuando sea detectado el paso del barco por el otro lado del río (es decir, ha cruzado) el puente empezará a bajar. Cuando llegue abajo, el semáforo pasará a verde para los vehículos.
Hay que tener en cuenta al hacer el programa que los detectores de barcos situados a uno y otro lado del río, se utilizan tanto para detectar la entrada de barcos hacia el puente por su lado del río como la salida de barcos desde el puente procedentes del otro lado del río; o sea, detectan el paso de un barco por ellos pero no distinguen entre barco entrante o barco saliente.
¡Ojo!, tener en cuenta la posibilidad de que en el momento de iniciarse el control, el puente no esté bajado. En este caso, el motor bajará el puente como primera operación.
Para no complicar en exceso, supondremos que si ha entrado un barco por uno de los lados, no puede entrar un barco por el lado contrario antes de que salga el que ha entrado primero. Tampoco tendremos en cuenta la posibilidad de que si ha entrado un barco por uno de los lados, entre otro barco por el mismo lado antes de que el primer barco salga por el lado contrario.
Conexiones: Los finales de carrera FCI y FCD irán conectados en los pines 7 y 8 respectivamente. Los finales de carrera FCS y FCB en los pines 9 y 12 respectivamente, el motor (M) en los pines 5 y 6. El semáforo: LR en pin 0, LA en pin 1 y LV en pin 2.

X.22.- Elaborar el programa de control de un sistema con 4 LEDs (L1 a L4), dos pulsadores (P1 y P2) y un zumbador (Z). Empiezan los 4 LEDs apagados. Pulso n veces el pulsador P1; cuando pulso el pulsador P2 se encienden tantos LEDs como pulsaciones haya dado antes en el pulsador P1 (a partir de 4 pulsaciones se encienden sólo los 4 LEDs). Al pulsar de nuevo el pulsador 2 se apagan los LEDs, suena brevemente el zumbador (por ejemplo 0,5 segundos) y se empieza de nuevo.

Los LEDs irán conectados en los pines 0, 1, 2 y 4. Los pulsadores irán conectados en los pines 7 y 8. El zumbador en el pin 3.

X.23.- Elaborar el programa de control de un sistema con 4 LEDs (L1 a L4) y dos pulsadores (P1 y P2). Empiezan dos LEDs encendidos. Por cada pulsación del pulsador P1 se enciende un nuevo LED y por cada pulsación del pulsador P2 se apaga un LED. Como máximo puede haber los 4 LEDs encendidos y como mínimo todos apagados.
Tener en cuenta que si se mantiene un pulsador pulsado no debe contarlo más que como una pulsación, es decir, hay que dejar de pulsar entre pulsación y pulsación.

Conexiones: Los LEDs irán conectados en los pines 0, 1, 2 y 4. Los pulsadores irán conectados en los pines 7 y 8.

X.24. – Realiza un programa en Arduino que haga que los cuatro LEDS del vehículo suministrado se vayan encendiendo progresivamente de izquierda a derecha. El encendido de los LEDs irá progresando desde una frecuencia lenta hasta una frecuencia más rápida. Luego se apagarán los LEDS y volverán a encenderse progresivamente pero de derecha a izquierda desde una frecuencia lenta hasta una más rápida. Finalmente se repetirá todo el proceso de manera indefinida.

Resolver la siguiente variante: que la frecuencia varíe en función de la luz que le de a una LDR.

X.25.- Supongamos una máquina provista de un motor que hace girar un disco. Supongamos que en cada vuelta el giro del disco hace que se pise un final de carrera FC. Elaborar un programa de control de manera que al pulsar un pulsador P el motor M empiece a girar y al dar 5 vueltas (o sea, pise 5 veces el final de carrera) se pare y se encienda un LED durante 5 segundos. El sistema empezará de nuevo cuando se vuelva a pulsar el pulsador P. Inicialmente o durante la espera a que se pulse el pulsador puede ocurrir que el disco no esté pisando el final de carrera FC, por lo que girará hasta que lo pise. A partir de este momento, esperará la pulsación del pulsador P.

Conexiones: el motor irá conectado en los pines 5 y 6, el pulsador de inicio P en el pin 7, el final de carrera FC que se pisa al girar en el pin 8 y el LED en el pin 0.
Para simplificar, supondremos que en el caso de que el pulsador se vuelva a pulsar antes de transcurrir los 5 segundos que permanece el LED encendido no se tendrá en cuenta.

X.26.- Sistema de alarma variante del anterior, compuesto por una LDR, un LED verde, LV, un LED rojo, LR, un pulsador de activación, PA, un pulsador de desactivación, PD, un pulsador auxiliar P2 y un zumbador, Z, (en los ensayos de prueba se sustituirá por un LED ámbar, LA, hasta que el programa funcione bien para evitar el excesivo ruido en las pruebas del programa).
El funcionamiento será el siguiente: con la alarma desconectada se enciende LV; la alarma se conectará pulsando el pulsador PA, encendiéndose LR. Una vez conectada la alarma, si se oscurece la LDR se activará la alarma haciendo sonar el zumbador Z. LR permanece también encendido mientras suena la alarma. La alarma se desconecta o se desactiva (en el caso de que haya saltado) mediante una determinada secuencia de pulsaciones de PA y P2 seguida de la pulsación de PD, apagándose LR y el zumbador Z si estaba sonando y volviéndose a encender LV. Si al pulsar PD no se ha introducido la secuencia correcta se puede empezar a introducir la secuencia de nuevo pero en el caso de que la alarma no estuviera activada se activará y empezará a sonar el zumbador. La secuencia de desactivación será, por ejemplo: PA-PA-P2-PA-P2, seguida de la pulsación de PD.
Conexiones: la LDR en el pin analógico 0, LR en el pin 1, LV en el pin 2, Z en el pin 3, PA en el pin 7, PD en el pin 8 y P2 en el pin 9.