Máster en Ingeniería Mecatrónica
Computadores y Programación

Examen Ordinario –Junio-Julio 2022

1) *1.5 puntos*

Realizar una función que, dada una tabla de valores reales, un límite inferior (valor real ***lim_inf***) por debajo del cual los datos no son válidos, y un nº entero (***nc***) con el número de datos contiguos a considerar, calcule:

- Se obtiene el índice del 1er elemento de la tabla que es menor a lim_inf (si no hay ninguno, usar como índice el nº de elementos de la tabla).
- Si el índice es mayor o igual a **nc**, se calcula y devuelve la media de los valores entre 0 y nc.
- Si el índice es menor a **nc**, se devuelve 0.

Ejemplo:

ind ≡ 0, 1, 4, 5, 6, , 10

tabla ≡ **84, 88, 95, 99, 84, 99**, 50, 36, 32, 30, 31

lim_inf = 80

nc = 2

MiFuncionEj1(tabla,80,2) → devuelve: **91.5** (media de los valores entre 0 y 5 incluidos).

MiFuncionEj1(tabla,90,2) → devuelve: **0** (menos de 2 elementos al principio mayores que lim_inf).

2) *1.5 puntos*

Realizar una función que, dada una cadena de caracteres con el formato indicado en el recuadro siguiente, y un valor de altura (número real), escriba en pantalla el texto correspondiente al 1er elemento que cumple que la altura recibido como argumento es superior al dato de dicho elemento, y devuelva un valor true/false indicando si alguna lo ha cumplido.

Formato de cadena:

Tipo=altura,importe / Tipo=altura,importe / Tipo=altura,importe /….

Camion=2500,60.40 / Furgoneta=1500,32.75 / Coche=1000,25.50

Ejemplos de resultado de la función:

MiFuncionEj2(cadena,1200); → Escribe Coche 25.50€ , devuelve true

MiFuncionEj2(cadena, 1800); → Escribe Furgoneta 32.75€, devuelve true

MiFuncionEj2(cadena,500); → No escribe nada, devuelve false

3) *1 puntos*

Realizar una función que, dado un entero que refleja el estado de un puerto de entrada, y dos enteros adicionales que indican los pesos de 2 bits a chequear, devuelva el valor entero:

0: si ambos bits del puerto son iguales

1: si el bit indicado por el 1er peso está a 1, y el segundo a 0

-1: si el bit indicado por el 1er peso está a 0, y el segundo a 1

Ejemplos:

valor_entero

B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
1	0	1	1	0	1	0	1

MiFuncionEj3(valor_entero,2,4) → devuelve 0 (B_2 y B_4 son iguales)

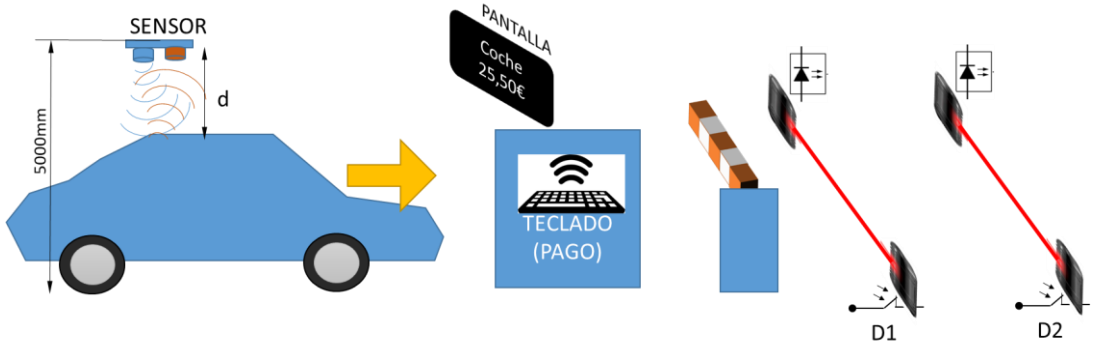
MiFuncionEj3(valor_entero,6,4) → devuelve -1 (B_6=0 y B_4=1)

Máster en Ingeniería Mecatrónica
Computadores y Programación

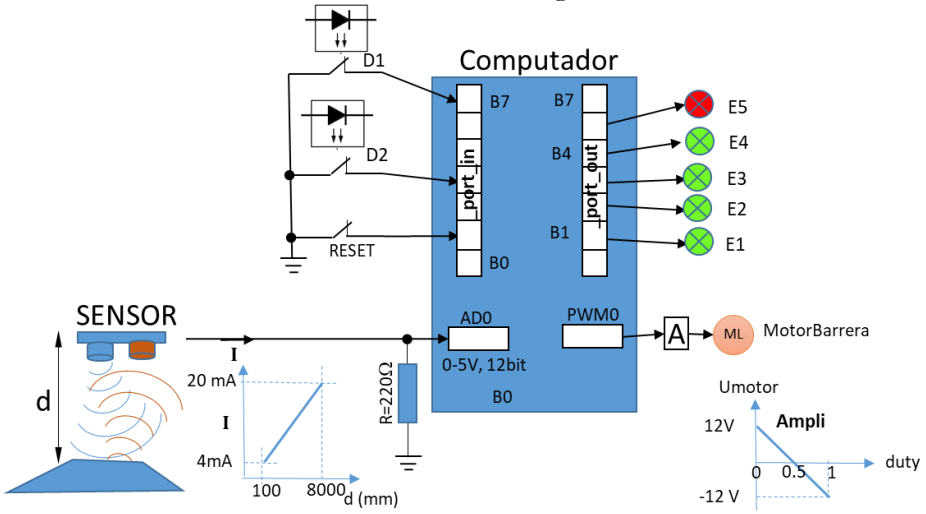
Examen Ordinario –Junio-Julio 2022

4) *5 puntos*

Se desea presentar de forma automática a los vehículos que acceden a un peaje la información del importe a pagar, y activar la barrera y el semáforo cuando el usuario ha abonado el importe. Para ello, se dispone del siguiente montaje que permite medir la altura mediante un sensor ultrasónico; con el cual se estima el tipo de vehículo de que se trata. Cuando el usuario efectúa el pago (que equivale a la introducción por teclado del valor del importe), si el importe satisfecho es mayor o igual al estimado, se abre la barrera hasta que el coche haya pasado completamente.



El conexionado con el computador será el siguiente:



Se debe realizar un programa para la gestión como se indica a continuación:

El estado se reflejará en los LED de visualización, y puede ser:

- ***E1***: Sin vehículo detectado: la pantalla estará en blanco, y la barrera bajada
- ***E2***: Vehículo detectado, importe no pagado: la pantalla muestra el tipo de vehículo y el importe, la barrera permanece bajada.
- ***E3***: Vehículo detectado, importe pagado: la pantalla muestra el tipo de vehículo y el importe, la barrera se sube.
- ***E4***: Vehículo ha superado la barrera (D2 activo y D1 inactivo) se borra la pantalla y se vuelve al estado ***E1***.
- ***E5***: ERROR. Tiempo vencido de apertura de barrera sin que pase el vehículo. Se regresa a ***E1*** cuando un operario pulse el botón RESET.

El programa principal inicializa una cadena de texto con la información de alturas e importes, inicializa una tabla de ***n***=10 medidas de altura, inicializa el resto de datos necesarios, establece

el estado ***E1***, y arranca una interrupción temporizada (10ms) para la obtención de datos del sensor, y se queda en un bucle a la espera de la entrada por teclado del importe abonado (scanf); si el estado es ***E2*** y el importe es igual al solicitado, se inicializa un contador de tiempo, se abre la barrera (aplicando 9V al motor) y se pasa al estado ***E3***.

En la rutina de interrupción temporizada, se actúa en función del estado:

- Si el estado es ***E1***, se obtiene una medida nueva de distancia del sensor, y se añade la altura correspondiente a la tabla de medidas de altura. Se considera que hay un vehículo si la altura media retornada por la función del ejercicio 1, usando nº de datos contiguos igual a 7, es mayor que 1000mm. En tal caso, se escribe en pantalla el importe correspondiente a esta altura media mediante la función del ejercicio 2 y, si esta función devuelve true, se pasa al estado ***E2*** y se borran las medidas más recientes.
- Si el estado es ***E2*** no se hace nada (main pasará a ***E3*** cuando se introduzca el importe).
- Si el estado es ***E3***, se incrementa el contador de tiempo y se comprueban los sensores D1 y D2, usando la función del ejercicio 3. Si D1 está inactivo y D2 está activo, se pasa al estado ***E4***. Si, por el contrario, el contador de tiempo indica 20 seg, se pasa al estado ***E5*** (error).
- Si el estado es ***E4***, se cierra la barrera (aplicando -9V al motor), se borra la pantalla y se vuelve al estado ***E1***.
- Si el estado es ***E5***, se borra la pantalla y se comprueba el pulsador RESET. Si el pulsador está activo, se regresa al estado ***E1***.

- Se dispone de las siguientes declaraciones de funciones y variables de E/S en "io.h":

```
void InitTemporizador(int T_ms,void (*FnCallback)() );
// Lanza una temporización por callback.
// La función callback debe declararse como:    void MiFn();

int ValorAD(int n_canal,int n_bitsAD);
// Obtiene el valor de la conversión A/D del canal deseado con el n° de bits
// de digitalización indicado.

void EscribirPWM(float duty_0_a_1);
// Escribe en el PWM una señal con el duty deseado

int _port_in;        // Valor del puerto de entradas digitales
int _port_out;       // Valor del puerto de salidas digitales
```

Máster en Ingeniería Mecatrónica
Computadores y Programación

Examen Ordinario –Junio-Julio 2022

5) Cuestiones:

1.5 puntos (+0.5 pto cuestión acertada, -0.2 pto cuestión fallada, 0 pto cuestión no contestada

a) ¿ Cuál de estas afirmaciones es **falsa** respecto a una salida PWM ?

- ☐ El valor varía automáticamente entre 0 y 1 a impulsos de un reloj
- ☐ El valor promedio depende de la relación entre Ton y Ttotal del ciclo
- ☐ Se debe amplificar para conseguir la tensión y corriente necesaria para el accionamiento al que se conecte
- ☐ Nunca puede utilizarse con LED

b) ¿ Cuánto vale x tras el la invocación de la función BuscaValor( ) ?

```
float BuscaValor(const char* c)
{
   char* pt=strchr(c,'=');
   if (pt!=NULL)
      return atof(pt+1);
   return -1;
}
...
float x=BuscaValor("PRUEBA Y=45 X=22");
```

- ☐ x vale 45
- ☐ x vale 22
- ☐ x vale -1
- ☐ x vale 0

c) ¿Por qué el código siguiente para la comprobación del interruptor SW4 siempre devuelve falso?

```
#define TRUE 1
#define FALSE 0

if (_port_sw_in & (1<<4) == TRUE)
    printf("SW4 activado\n");
else
    printf("SW4 desactivado \n");
```

- ☐ Porque #define no se puede usar para comparaciones
- ☐ Porque 1<<4 es siempre distinto de TRUE
- ☐ Porque _port_sw_in debería ser una función
- ☐ Porque se debería comparar con !=

Máster en Ingeniería Mecatrónica
Computadores y Programación

Examen Ordinario –Junio-Julio 2022

Algunas funciones de C:

```
int atoi(const char* cadena);
double atof(const char* cadena);
double strtod(const char* cadena,char** final);
int strlen(const char* cadena);
char* strcpy(char* dst,const char* src);
char* strncpy(char* dst,const char* src,int n);
char* strcat(char* dst,const char* src);
char* strncat(char* dst,const char* src,int n);
int strcmp(const char* cad1,const char* cad2);
int strncmp(const char* cad1,const char* cad2,int n);
char* strchr(const char* cad,char c);
char* strstr(const char* cad,const char* busca);
int sprintf(char* dest,const char* fmt,...);
int sscanf(const char* src,const char* fmt,...);
FILE* fopen(const char* filename,const char* mode);
char* fgets(char* dest,int tam_max,FILE* f);
int fprintf(FILE* fid,const char* fmt,...);
int fscanf(FILE* fid,const char* fmt,...);
int fclose(FILE* fid);
void* malloc(int tam);
void free(void* pt);
```