Máster en Ingeniería Mecatrónica
Computadores y Programación

Examen Ordinario – Mayo 2021

Ejercicio 1):Realizar una función que, dada una tabla de valores reales, y un valor real de comparación, calcule el mínimo y máximo de los datos de la tabla y devuelva:
2 puntos

- Un valor entero con todos sus bits a 0 si tanto el mínimo como el máximo se encuentran en el intervalo alrededor del 1% del valor de comparación.

Si no se cumple lo anterior:

- Un valor entero con únicamente el bit de peso 2 a 1, si el mínimo y/o el máximo se encuentran en el intervalo alrededor del 10% del valor de comparación.

Si no se cumple lo anterior:

- Un valor entero con únicamente el bit de peso 5 a 1.

Ejemplos:

cmp=10.0 → Intervalo 1% =[9.9 a 10.1] , intervalo 10% = [9.0 a 11.0]
t = [9.97, 10.02 , 10.01] , cmp=10.0 → Devuelve binario 00…000000
t = [9.97, 10.12 , 10.01] , cmp=10.0 → Devuelve binario 00…000100
t = [9.97, 11.12 , 10.01] , cmp=10.0 → Devuelve binario 00…100000

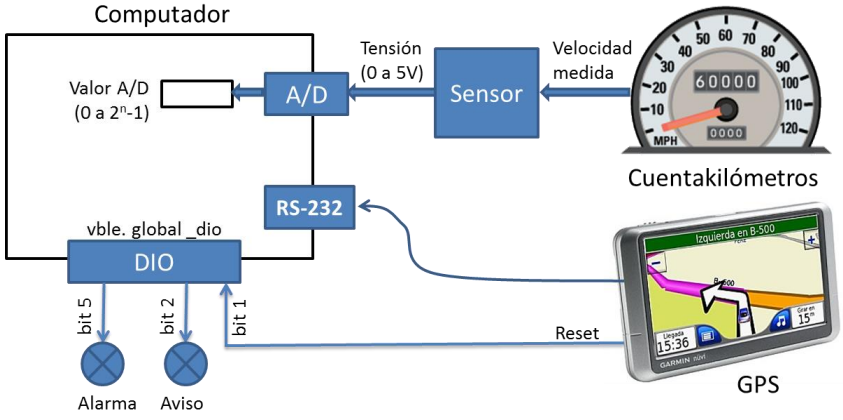
Ejercicio 2): Realizar una función que, dada una cadena de caracteres y un valor entero, devuelva el número real que se encuentra dentro de la cadena en el ítem indicado por el valor entero (comenzando la cuenta en 0). Los ítems de la cadena de caracteres estarán separados por el carácter ':'. La función devolverá 0 si no hay un número real en la posición indicada, o no hay dicha posición en la cadena.
2 puntos

Ejemplos:

MiFuncionEj2("3.5 : 4.7 : a.2 : 5.4 : 6.3",0); → Devuelve el nº real 3.5
MiFuncionEj1("3.5 : 4.7 : a.2 : 5.4 : 6.3",3); → Devuelve el nº real 5.4
MiFuncionEj2("3.5 : 4.7 : a.2 : 5.4 : 6.3",2); → Devuelve el nº real 0.0
MiFuncionEj2("3.5 : 4.7 : a.2 : 5.4 : 6.3",5); → Devuelve el nº real 0.0

Ejercicio 3): Se desea realizar un sistema de control para comprobar la exactitud de la medición de distancia del cuentakilómetros de un vehículo. Para ello, se dispone del montaje siguiente:

5 puntos



Máster en Ingeniería Mecatrónica
Computadores y Programación

Examen Ordinario – Mayo 2021

La velocidad medida por el cuentakilómetros es captada por un sensor que entrega una tensión entre 1V (para 0Km/h) y 4V (para 180Km/h), y adquirida con un conversor A/D 0-5V de 12 bits.

El GPS está conectado por RS-232 a la entrada estándar stdin (equivalente al teclado), y envía cada segundo una cadena de caracteres (longitud máxima 50) con el siguiente formato:

campo0 : campo1 : campo2 : campo3 : campo4 : campo5 \n

Dónde:

campo4 es la posición en Km desde el comienzo del viaje (ej: 75.2).

y el nº de caracteres de cada campo puede variar.

La comprobación de funcionamiento se realizará de la siguiente manera:

- La obtención de datos del GPS se realiza mediante un bucle llamando a la función gets() en main(). Una vez obtenida la cadena, se debe extraer la posición en Km a partir del campo 4 (función ejercicio 2).

- En una interrupción temporizada, cada 200 ms:
  - Se comprueba el bit de peso 1 de _dio. Cuando el usuario pone a cero el contador de distancia del GPS se activa el bit de peso 1 de _dio. En ese instante, se debe reiniciar la medida de posición a cero, así como el tiempo; sólo si ha habido esta activación en algún momento, se realizan los pasos siguientes de comprobación:
  - Se adquiere la velocidad actual del cuentakilómetros (en Km/h) a través de la lectura del canal A/D.
  - Se integra la velocidad para obtener la posición medida por el cuentakilómetros, mediante integración trapezoidal (ver anexo).
  - Se calcula el mínimo y máximo de las últimas 5 medidas de posición a partir del cuentakilómetros, y se compara con la medida más reciente obtenida del GPS (función ejercicio 1). Se utiliza el valor devuelto por esta función para activar los bits correspondientes de alarma y aviso de la variable global _dio.
  - Se escribe en pantalla una línea con la información siguiente:

    pos_min , pos_max , pos_GPS , AVISO ON/OFF, ALARMA ON/OFF

- El resto de bits de _dio no debe modificarse en ningún caso.

Máster en Ingeniería Mecatrónica
Computadores y Programación

Examen Ordinario – Mayo 2021

ANEXO

Librería auxiliar:

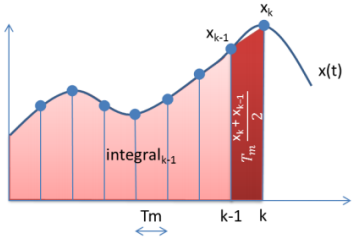
Se dispone de las siguientes funciones y declaraciones dentro de una librería FnAux (archivos FnAux.h y FnAux.lib):

```
// Inicialización de un temporizador que llame a la función indicada en
// FnCallback cada cierto intervalo (en ms)
void InitTemporizador(int tiempo_ms,void (*FnCallback)();

// Obtención del valor de conversion A/D del canal deseado utilizando
// nBits : resultado de 0 a 2^nBits-1, para tensión de entrada 0 a 5V.
int ConversionAD(int nCanal,int nBits);

// Variable global que almacena los contenidos de las salidas digitales
int _dio;
```

Algunas funciones de C:

```
int atoi(const char* cad);              // Devuelve entero equivalente a cadena
double atof(const char* cad);           // Devuelve real equivalente a cadena
double strtod(const char* cad,char** next);   // Id. a atof() y guarda en next puntero
                                              // a final de conversión
int strlen(const char* cadena);         // Devuelve longitud de cadena
char* strcpy(char* dst,const char* src);      // Copia cadena fuente en destino
char* strncpy(char* dst,const char* src,int n);      // Id. Máximo n caracteres
char* strcat(char* dst,const char* src);      // Concatena cadena Fuente a destino
char* strncat(char* dst,const char* src,int n);      // Id. Máximo n caracteres
char* strchr(const char* cad,char c);   // Busca caracter en cadena, devuelve puntero
                                        // a la primera ocurrencia o NULL si no está
char* strstr(const char* cad,const char* busca);     // Id. buscando cadena
int strcmp(const char* c1,const char* c2);    // Compara cadenas, devuelve 0 si iguales
char* gets(char* destino);              // Lee cadena de consola, almacena en destino
void* malloc(int n_bytes);              // Asigna memoria para n bytes
void free(void* ptr);                   // Libera memoria asignada
```

Integración trapezoidal:



$I_k = I_{k-1} + (x_k + x_{k-1})/2 * T_m$

Máster en Ingeniería Mecatrónica
Computadores y Programación

Examen Ordinario – Mayo 2021

APELLIDOS Y NOMBRE: _____

Ejercicio 4 (responder aquí):

1 punto (0.25 pto cuestión acertada, -0.1 pto cuestión fallada, 0 pto cuestión no contestada

a) ¿Cuánto vale la variable *x* tras ejecutar el código siguiente?

```
int Suma(const int* t,int n);                /* Suma los valores de una tabla
...                                          de n enteros */
int datos[4]={0,2,4,6};
int x;
x=Suma(datos+2,2);
```

- ☐ *x* vale 12 si la función `Suma( )` está correctamente realizada
- ☐ *x* vale 10 si la función `Suma( )` está correctamente realizada
- ☐ El valor de *x* depende del valor del puntero *datos*, es indeterminado
- ☐ *x* vale 2 si la función `Suma( )` está correctamente realizada

b) ¿ Qué ocurre con el código siguiente para convertir los valores de una tabla en sus cuadrados ?

```
void Cuadrados(const int* t,int n)
{
   int i;
   for (i=0;i<n;i++)
      t[i] *= t[i];
}
...
int datos[4]={0,2,4,6};
Cuadrados(datos[],4);
```

- ☐ No se puede compilar, la declaración del puntero *t* no puede ser const
- ☐ No se puede compilar, la llamada a `Cuadrados( )` es inválida
- ☐ Las dos anteriores son ciertas
- ☐ Ninguna de las anteriores es cierta

c) ¿ Cómo se sabe si el bit de peso P de la variable entera V está inactivo (a cero) ?

- ☐ `if (!(v & (1 <<p)) )`
- ☐ `if ~(v =& 1<<p)`
- ☐ `if (p[v] << ~1)`
- ☐ `if (v & (0 <<p) )`

d) ¿ Qué es falso 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