UNIVERSIDAD DE ALCALÁ ESCUELA POLITÉCNICA SUPERIOR DEPARTAMENTO DE ELECTRÓNICA		GRADO EN INGENIERÍA ELECTRÓNICA DE COMUNICACIONES	
ASIGNATURA	SISTEMAS ELECTRÓNICOS DIGITALES AVANZADOS	**FECHA**	**MAYO 2016**
APELLIDOS, NOMBRE		**GRUPO**	

PRUEBA DE EVALUACIÓN FINAL (PRUEBA 2)

CUESTIÓN 1

Suponiendo que un sistema basado en el LPC1768 tiene cuatro tareas con los parámetros que se indican en la tabla y teniendo en cuenta que existe una **región crítica** en la Tarea 3 de **5ms**, y otra en el programa principal de **2ms**

Tarea	Prioridad	Subprioridad	C	T	D
Tarea 1	0	0	3	10	10
Tarea 2	1	0	10	40	40
Tarea 3	1	1	10	70	40
Tarea 4	1	2	15	100	80

Nota: Unidades en ms

Se pide:

a) Analice ejecutabilidad de las tareas Tarea 1 y Tarea 4

CUESTIÓN 2.
Explique qué es un DSP y sus características fundamentales.

CUESTIÓN 3.
Se desea realizar el control de la calefacción remota de una vivienda con un sistema basado en el LPC1768 para lo que se utilizan las librerías RL-ARM TPCnet de Keil. Escriba las funciones necesarias para que, al acceer a la página "temperatura.cgi":

a) visualice remotamente la temperatura ambiente almacenada en el microcontrolador en la variable *int ambiente*
b) permita introducir la temperatura de consigna que será almacenada en la variable *int consigna*

Al pulsar "Enviar consigna" se envía la información junto con la URL con el formato "*temperatura.cgi?consigna=25*", por ejemplo.

Escriba el contenido del fichero "temperatura.cgi" y las funciones del fichero HTTP_CGI.c necesarias.

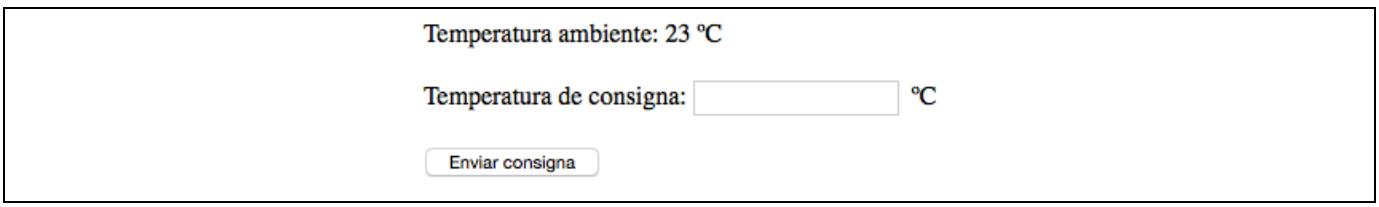

Figura 1: Página que se viasualiza al acceder a *temeperatura.cgi*

```
<!DOCTYPE html>
<html>
  <head>
    <meta content="text/html; charset=windows-1252" http-equiv="content-type">
  </head>
  <body>
    Temperatura ambiente: 23 &#186;C<br>
    <br>
    <form action="temperatura.cgi" method="get">
      Temperatura de consigna: <input name="consigna" type="text"> 
      &#186;C<br>
      <br>
      <input value="Enviar consigna" type="submit">
    </form>
  </body>
</html>
```
Figura 2: Código HTML de la página "temperatura.cgi" una vez cargado

CUESTIÓN 4.

El código del Anexo I corresponde a la programación de un StateChart.

a) Represente el StateChart
b) Ponga en diagrama asociado a las transiciones que corresponden con temporizaciones, el tiempo en milisegundos (póngalo entre paréntesis).

ANEXO I (Código de StateChart)

```
typedef enum {N1_DEFAULT, N1_1, N1_2, N1_3} TipoState_N_1;
typedef enum {N1_3_DEFAULT, N1_3_1, N1_3_2, N1_3_3} TipoState_N_1_3;
typedef enum {N1_3_3_DEFAULT, N1_3_3_1, N1_3_3_2} TipoState_N_1_3_3;
typedef enum {N1_3_3_1_DEFAULT, N1_3_3_1_1, N1_3_3_1_2} TipoState_N_1_3_3_1;
typedef enum {N1_3_3_2_DEFAULT, N1_3_3_2_1, N1_3_3_2_2} TipoState_N_1_3_3_2;

TipoState_N_1       State_N_1;
TipoState_N_1_3     State_N_1_3;
TipoState_N_1_3_3   State_N_1_3_3;
TipoState_N_1_3_3_1 State_N_1_3_3_1;
TipoState_N_1_3_3_2 State_N_1_3_3_2;

int Entry;

unsigned int T1, T2;

void periodic_interrupt_5ms(void) {
   if (T1 > 0) T1--;
   if (T2 > 0) T2--;
}

void Init_StateChart(void) {
   State_N_1 = N1_DEFAULT;
   State_N_1_3 = N1_3_DEFAULT;
   State_N_1_3_3 = N1_3_3_DEFAULT;
   State_N_1_3_3_1 = N1_3_3_1_DEFAULT;
   State_N_1_3_3_2 = N1_3_3_2_DEFAULT;
   Entry = 1;
}

void Exec_N1_3_3_1(void)
{
   if (State_N_1_3_3_1 == N1_3_3_1_DEFAULT)
            State_N_1_3_3_1 = N1_3_3_1_2;

   if (Entry) {
      Entry = 0;
   }

   if (Input_1 > 0) {
      State_N_1_3_3 = N1_3_3_2;
      State_N_1_3_3_2 = N1_3_3_2_DEFAULT;
      Entry = 1;
      return;
   }

   switch (State_N_1_3_3_1) {
      case N1_3_3_1_1:
         if (Entry) {
            T1 = 10000;
            Action_1();
            Entry = 0;
         }
         if (T1 == 0) {
            State_N_1_3_3_1 = N1_3_3_1_2;
            Entry = 1;
         }
         break;
      case N1_3_3_1_2:
         if (Entry) {
            T1 = 5000;
            Action_2();
            Entry = 0;
         }
         if (T1 == 0) {
            State_N_1_3_3_1 = N1_3_3_1_1;
            Entry = 1;
         }
         break;
   }
}

void Exec_N1_3_3_2(void)
{
   if (State_N_1_3_3_2 == N1_3_3_2_DEFAULT)
      State_N_1_3_3_2 = N1_3_3_2_1;

   if (Entry) {
      Entry = 0;
   }

   switch (State_N_1_3_3_2) {
      case N1_3_3_2_1:
         if (Entry) {
            T1 = 200;
            Entry = 0;
         }
         Retrocede();
         if (T1 == 0) {
            State_N_1_3_3_2 = N1_3_3_2_2;
            Entry = 1;
         }
         break;

      case N1_3_3_2_2:
         if (Entry) {
            T1 = 200;
            Entry = 0;
         }
         Action_3();
         if (T1 == 0) {
            State_N_1_3_3 = N1_3_3_1;
            State_N_1_3_3_1 = N1_3_3_1_2;

            Entry = 1;
         }
         break;
   }
}
```

```
void Exec_N1_3_3(void)
{
   if (State_N_1_3_3 == N1_3_3_DEFAULT)
      State_N_1_3_3 = N1_3_3_1;

   if (Entry) {
      Entry = 0;
   }

   if (Input_2 < 20) {
      State_N_1_3 = N1_3_2;
      Entry = 1;
      return;
   }

   switch (State_N_1_3_3) {
      case N1_3_3_1:
         Exec_N1_3_3_1();
         break;
      case N1_3_3_ESCAPE:
         Exec_N1_3_3_2();
         break;
   }
}

void Exec_N1_3(void) {

   if (State_N_1_3 == N1_3_DEFAULT)
      State_N_1_3 = N1_3_1;

   if (Entry) {
      T2 = 3000;
      Entry = 0;
    }

    if (T2 == 0) {
      State_N_1 = N1_2;
      Entry = 1;
      return;
   }

   switch (State_N_1_3) {
      case N1_3_1:
         if (Entry) {
            T1 = 500;
            Entry = 0;
         }
         Action_4();
         if (T1 == 0) {
            State_N_1_3 = N1_3_3;
            State_N_1_3_3 = N1_3_3_DEFAULT;
            Entry = 1;
            break;
         }
         break;
      case N1_3_2:
         if (Entry) {
            Entry = 0;
         }
         Action_1();
         if (Input_2 >= 20) {
            State_N_1_3 = N1_3_3;
            State_N_1_3_3 = N1_3_3_1;
            State_N_1_3_3_1 = N1_3_3_1_1;
            Entry = 1;
            break;
         }
         break;
      case N1_3_3:
         Exec_N1_3_3();
         break;
   }
}

void Exec_Nivel_1(void) {
   if (State_N_1 == N1_DEFAULT)
      State_N_1 = N1_1;

   switch (State_N_1) {
      case N1_1:
         Action_4();
         if (Input_3 == 1) {
            State_N_1 = N1_3;
            State_N_1_3 = N1_3_DEFAULT;
            Entry = 1;
            break;
         }
         break;
      case N1_2:
         Action_4();
         break;
      case N1_3:
         Exec_N1_3();
         break;
   }
}

int main (void)
{
   Init_IO();
   Init_StateChart();

   while(1) {
      ReadInput();
      Exec_Nivel_1();
      UpdateOutput();
   }
}
```