


Sistemas en Tiempo Real – P2

Creación de un proyecto en ADA

- Seleccionar *Create new project from template*.
- Seleccionar *Basic→Simple Project*.
- Seleccionar el nombre del proyecto, del ejecutable y la ruta donde se creará el proyecto.
 - o No se crea automáticamente una carpeta para el proyecto, es decir, que si ponemos el escritorio como ruta, se crearán todos los ficheros directamente en el escritorio, por lo que se recomienda crear una carpeta específica.

El código estará en NombreProyecto/src/NombreProcedimientoPrincipal.adb

Para compilar el código, pulsar sobre el botón  de la barra de herramientas.

Para ejecutar, pulsar sobre el botón  de la barra de herramientas.

Abrir de un proyecto existente

- Seleccionar *Open existing project*: e indicar la ruta del fichero *.gpr* que contiene el proyecto.

Ejercicio 1.

Implementar el buffer acotado visto en teoría. Para ello haremos:

- Crear un paquete que implemente el objeto protegido *Buffer_Acotado*.
- Definir dos tareas tipo, una para los productores y otra para los consumidores.
- Crear 10 tareas productoras y 10 tareas consumidoras, cada una de las cuales insertará/leerá 50 elementos del buffer.

La práctica se comprimirá en un único archivo que será entregado a través del campus virtual usando la tarea creada para tal efecto.

ANEXO – Ayuda para ADA

Creación de paquetes

Se crearán dos ficheros en la carpeta src del proyecto:

- nombre_paquete.ads: definición del paquete.
- nombre_paquete.adb: implementación del paquete.

Paso de parámetros a tareas

Para poder pasar parámetros a las tareas, deben ser “punteros”, que en ADA se traduce en usar la palabra reservada *access* delante del tipo (sería similar al * en C++).

Sin embargo, como lo que queremos es pasar la dirección de una variable estática (lo que en C++ sería el &), usaremos *access all* en lugar de sólo *access* en la definición de los parámetros de la tarea. Se recomienda definir un nuevo tipo puntero.

Además, cuando definamos la variable que queremos pasar por declaración, usaremos la palabra *aliased* delante del nombre del tipo de la variable (esto permite acceder a la dirección de la misma) y, posteriormente usaremos el atributo 'Access para acceder a la dirección de la variable.

Resumiendo:

```
Variable: aliased TipoNormal;--Variable de tipo no puntero
TipoPuntero is access all TipoNormal;--Nuevo tipo puntero
task type TareaTipo(Parametro: TipoPuntero);--Parámetro de la
--tarea de tipo puntero
task body TareaTipo is begin ... end;--Cuerpo de la tarea
Tarea: TareaTipo(Variable'Access);--Creación de la tarea pasando
--por parámetro la dirección de la variable
```

Algunas funciones, atributos y operadores de utilidad

- Put (paquete GNAT.IO): Muestra por pantalla una cadena de caracteres o un entero.
- New_Line (paquete GNAT.IO): Imprime un salto de línea.
- Put_Line (paquete GNAT.IO): Imprime por pantalla una cadena de caracteres y la termina con un salto de línea.
- integer'Image(Dato): Devuelve el valor del entero *Dato* en forma de cadena de caracteres.
- integer'Value(Dato): Devuelve el valor de la cadena de caracteres *Dato* en forma de entero.
- &: Operador de concatenación.

Se recomienda poner los mensajes dentro de la región crítica ya que, si no, se pueden producir “interferencias” entre los diferentes mensajes de las distintas tareas.