Práctica 5:

Comunicación Hardware – Software: desde CoOs a una aplicación en .Net

Antonio Portillo Raya

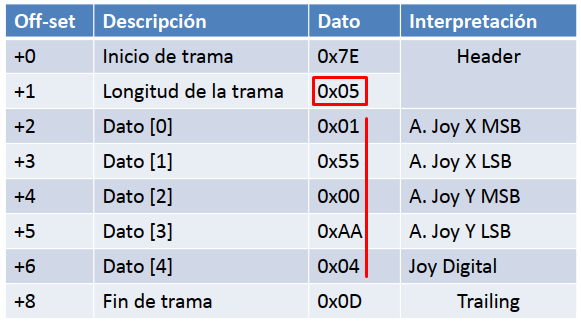
Alejandro Vega Vaquero

## Objetivos

* Ilustrar el manejo de los puertos series asíncronos de un PC desde C#
* Implementar y desarrollar un mecanismo de comunicación de alto nivel.
* Empaquetado de la información:
  + Enviar al PC paquetes con el estado de los sensores:
    - Joystick digital y analógico.
  + Recibir comandos desde el PC s:
    - Mover servo / motor.
    - Lanzar animaciones de Leds.
* Implementar una aplicación en C# usando Windows Form:
  + Mostrar la información recibida a través del puerto serie al usuario.
  + Generar y enviar comandos por el puerto serie al sistema.

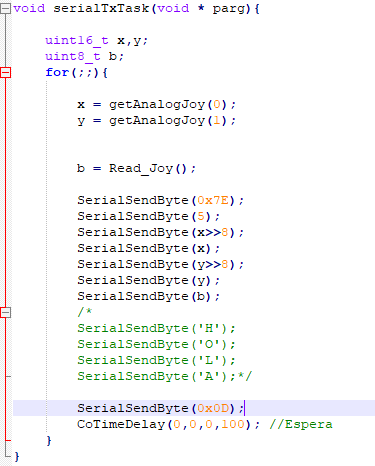
## Empaquetado de la información

En esta práctica vamos a usar mecanismos de empaquetamiento de la información para la comunicación ya que es lo común en sistemas a un alto nivel.



## Transmitiendo desde el STM32 – Recibiendo en C#

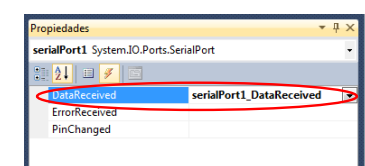
Con la función serialTxTask vamos a leer cada 100ms los valores de los dos ejes del joystick analógico y el digital. Además, vamos a enviar un paquete con la estructura arriba expuesta con los datos.



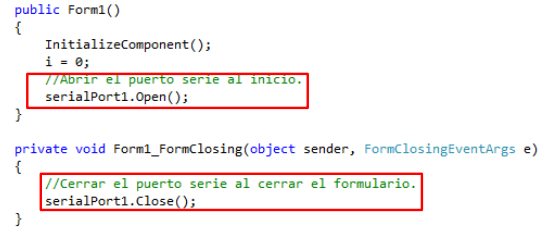
Para cerciorarnos de que enviamos correctamente usamos la herramienta X-CTU (vista hexadecimal).

## Formulario en el Visual Studio

Creamos un proyecto en Visual C# (Aplicación de Windows forms) en nuestro Visual Studio. En nuestro diseño iremos añadiendo diferentes elementos desde la barra de herramientas como botones o cuadros de texto.

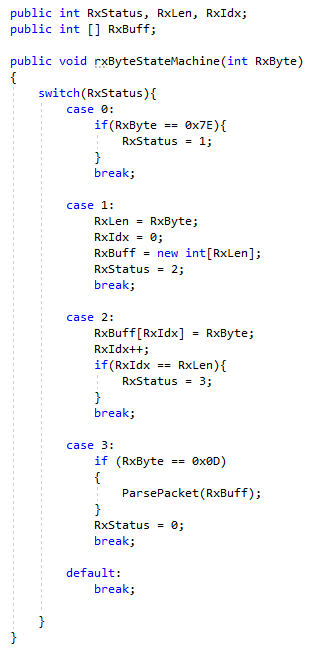
Para empezar, debemos añadir un serialPort para nuestra comunicación. Lo configuraremos a un BaudRate de 115200 y le cambiaremos el PortName al puerto que nos diga el X-CTU. Es importante que le añadamos un evento de datos recibidos. 

Añadiremos tambien un evento FormClosing donde posteriormente cerraremos el puerto serie.



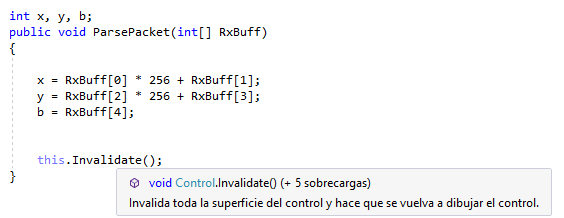
## Recibiendo datos

Para recibir los datos vamos a implementar una máquina de estados en la función RxByteStateMachine para recibir los datos de cada paquete. El método es invocado byte a byte. Al finalizar la recepción de un paquete deberemos descodificarlo.



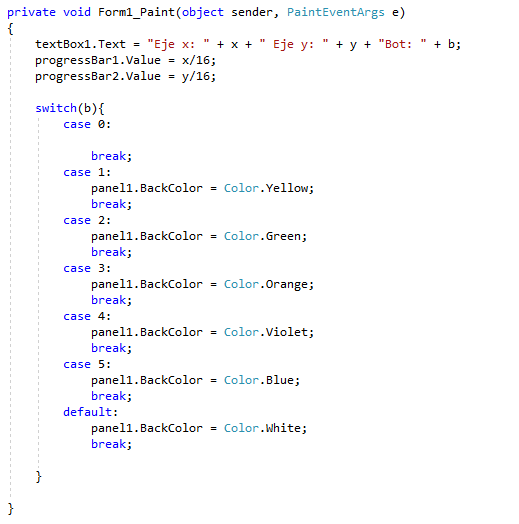
## Descodificando la información

Para decodificar el paquete usamos la siguiente función.



## Diseño del formulario

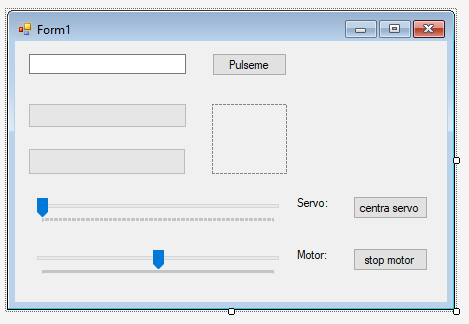
Usaremos un evento Paint para repintar el formulario.



Nuestro formulario constará de:

* + Un cuadro de texto y un botón.
  + Dos barras progresivas para pintar la posición de los joysticks analógicos.
  + Un panel para representar el valor del joystick digital.
  + Una barra para posicionar nuestro servo al gusto.
  + Una barra para hacer que nuestro motor gire en el sentido y velocidad que deseemos.
  + Dos botones para centrar el servo o parar el motor.

Quedando finalmente así:

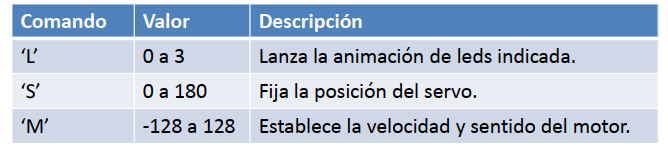


## Enviando comandos al CoOs

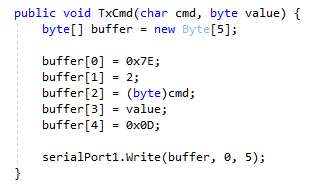
Vamos a enviar comandos al STM32 desde la aplicación para:

* + Lanzar animaciones LED.
  + Posicionar el servo.
  + Dar velocidad y dirección al motor.

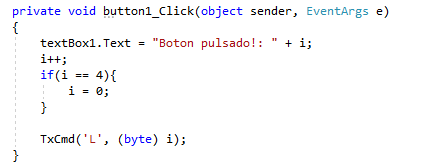
Los empaquetaremos en 2 bytes, comando y valor:



Para ello vamos a crear un método llamado TxCmd:

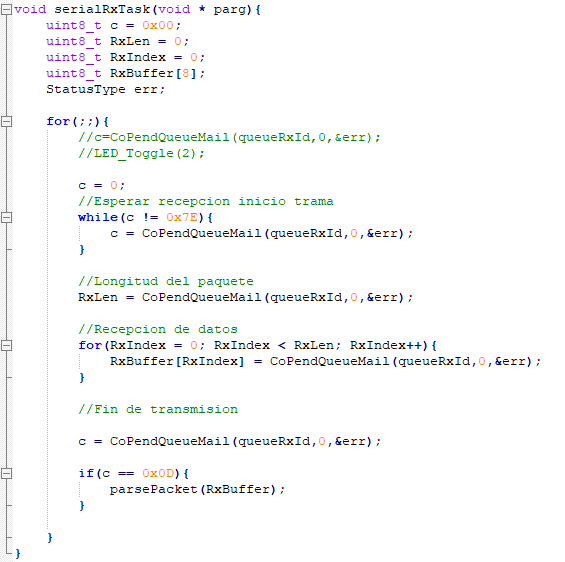


Para enviar comandos a los leds usaremos el primer botón.



## Recibiendo datos en el STM32

Modificamos la tarea serialRxTask para que quede así:



Al final de la tarea debemos decodificar el paquete:

