Cátedra de Sistemas Operativos II

Trabajo Práctico N°IV

Soriano, Juan

22 de junio del 2019

Introducción

En esta sección se encuentra la especificación de los requerimientos del desarrollo de un programa con tres tareas que corre en el sistema operativo de tiempo real FreeRTOS correspondiente al trabajo N° 4 de la Sistemas Operativos II.

Propósito

El propósito del Trabajo Práctico n° 4 de Sistemas Operativos II es el de diseñar e implementar un software con diversas tareas que se ejecute sobre un sistema operativo en tiempo real (RTOS). Se realizaron 3 tareas principales:

1. Instalar y configurar FreeRTOS en el sistema embebido seleccionado.
2. Crear un programa con dos tareas simples, un productor y un consumidor y realizar un análisis completo del sistema con tracealyzer.
3. Diseñar e implementar una aplicación con una tarea periódica y una aperiódica que generan datos. La tarea períodica genera datos de longitud fija mientras que la tarea aperiódica de longitud variable. Además implementar una tarea consumidor que envía el valor recibido a la terminal de una computadora por puerto seria (UART).

   A continuación se muestra un diagrama esquemático de lo realizado

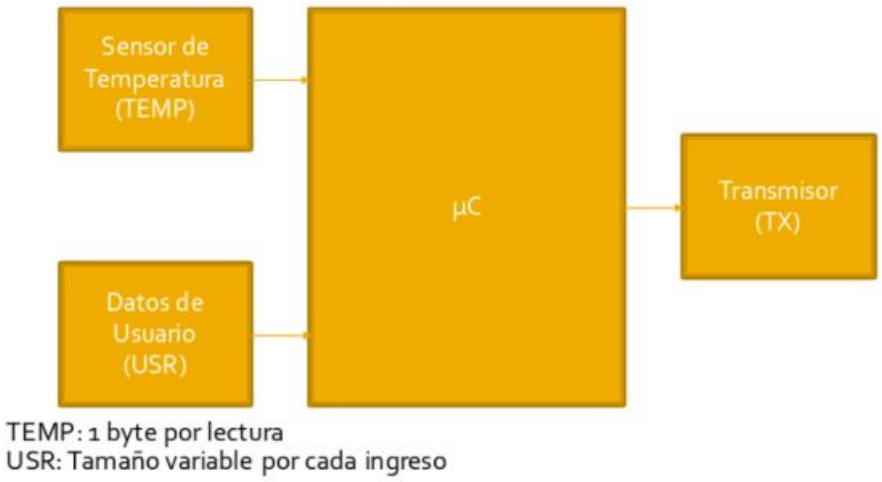

Figura 1. Diagrama del diseño de la aplicación para el inciso 3.

Referencias

1. https://www.freertos.org/
2. https://percepio.com/tracealyzer/

Descripción General

En la presente sección se explicarán los distintos aspectos del trabajo. Las funciones que ofrece el producto, la perspectiva del mismo, las características de los usuarios, requisitos para futuras iteraciones, entre otras cosas.

Perspectiva del Producto

El producto es una implementación de un servicio liviano hosteado en un webserver (lighttpd) que debe poder ofrecer funcionalidades a nivel de sistema operativo al usuario por medio de CGI.

Características de los Usuarios

El producto se destina a usuarios con conocimientos básicos del manejo de sistemas operativos Linux, sistemas embebidos, eclipse y hardware para conexión e utilización del puerto serie.

Restricciones

Se necesita de una computadora con requerimientos de hardware mínimos capaz de comunicarse por puerto serie con la placa. Además se necesita una placa LPC176X capaz de correr el programa, freeRTOS y el tracealyzer. Por otra parte es necesario un conversor USB a UART.

Requisitos Específicos

En la presente sección se explicará con mayor lujo de detalle las entradas y salidas del sistema como así también una descripción de las funciones y requerimientos funcionales del producto.

Funciones

En esta sección se listan los requerimientos funcionales de la aplicación para la versión 1 del release del producto con fecha 22 de Junio de 2019:

El producto realiza principalmente las siguientes tareas:

1. Recolección de información del contexto de las tareas utilizando el tracealyzer.
2. Devolución de los datos por puerto serie de la placa.
3. Simulación de tres tareas, dos generadoras y una consumidora de datos.

Restricciones de Diseño

En esta sección se listan las restricciones de diseño de la aplicación para la versión 1 del release del producto con fecha 22 de Junio de 2019:

1. Utilizar una placa con un cortex M.
2. Utilizar tracealyzer.

Atributos del Sistema

En esta sección se listan los atributos del sistema de la aplicación para la versión 1 del release del producto con fecha 6 de Junio de 2019:

1. Sistema portable, es decir, deberá ser capaz de correr en cualquier sistema operativo.

Guía “how to” para instalar el entorno de prueba

El primer paso consiste en instalar el LPXExpresso, el IDE para programar la LPC1769 utilizada en este trabajo práctico. Luego instalar FreeRTOS del siguiente enlace: https://www.freertos.org/. Una vez descargadas ambas herramientas, en https://percepio.com/tracealyzer/ se puede descargar el tracealyzer que viene con una licencia de prueba. Una vez instaladas todas las herramientas, se deberán configurar los headers de trace para trabajar con la LPCExpresso, en la carpeta RecTrace que genera el tracealyzer, se encuentran los headers para trabajar con FreeRTOS y deberán ser configurados para trabajar con la placa que se utilice. Una vez configuradas e instaladas todas las herramientas, se puede instalar el plugin de tracealyzer para trabajar con eclipse según el tutorial https://mcuoneclipse.com/2017/03/08/percepio-freertos-tracealyzer-plugin-for-eclipse/. Una vez instalado, solamente con inicializar tracealyzer y hacer un save snapshot, la ventana de tracealyzer se abre con la captura del trace correspondiente.

Implementación y resultados obtenidos

Productor y consumidor simple

Esta etapa del trabajo se hizo para familiarizarse con el funcionamiento del sistema y de la herramienta tracealyzer. Utilizando el ejemplo de la bibliografía, se implementó un programa que consiste en dos tareas simples y una cola. Una de las tareas emite mensajes y la otra los va recibiendo. A continuación se muestran los datos obtenidos con la herramienta tracealyzer.

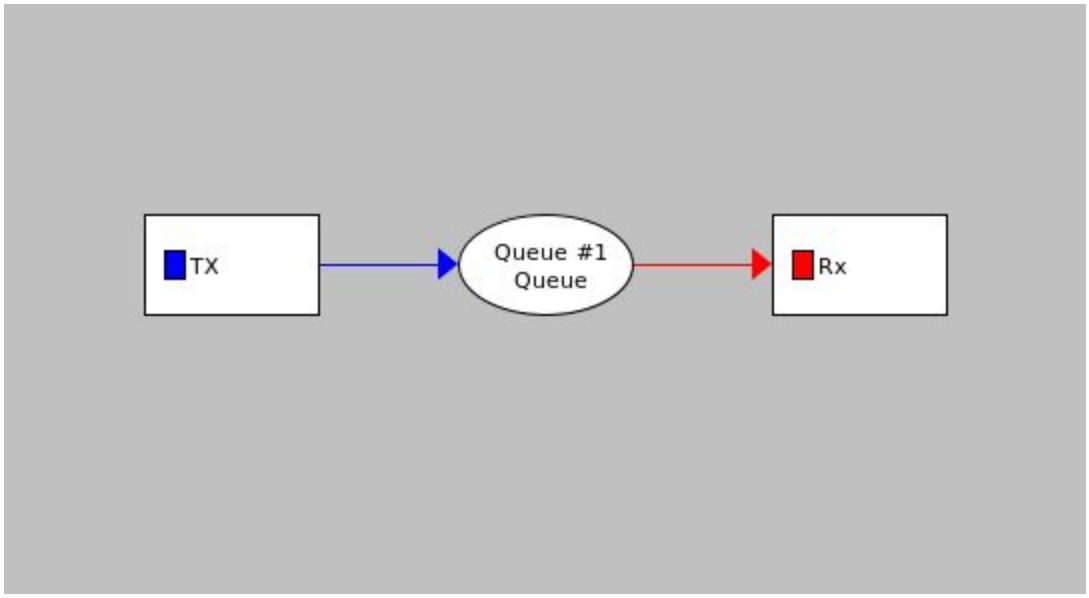

Figura 2. Actores

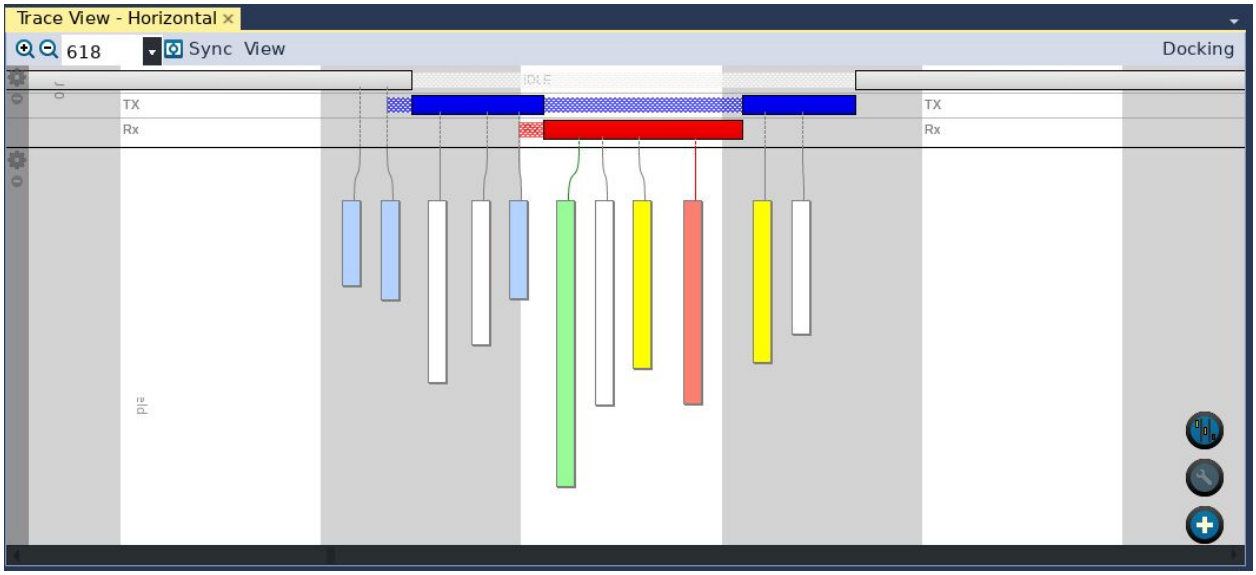

Figura 3. Traza capturada

Figura 4. Tiempo de ejecución de las tareas

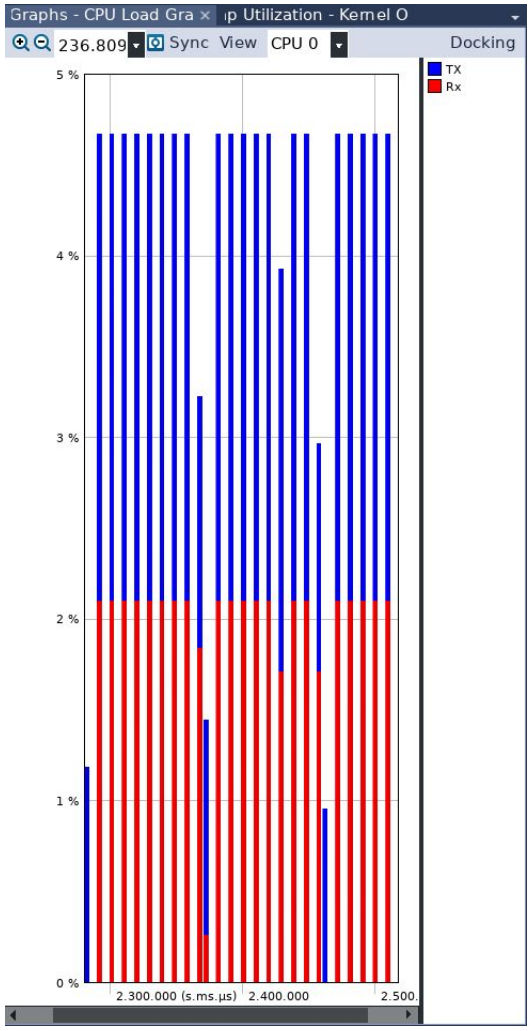

Figura 5. Utilización del CPU

Programa con tres tareas

En esta versión, se utilizan tres actores como se detalló previamente. La tarea receptora envía mediante puerto serie los datos leídos. Se asignó con mayor prioridad a la tarea consumidora para que sea preemptive respecto a las otras.

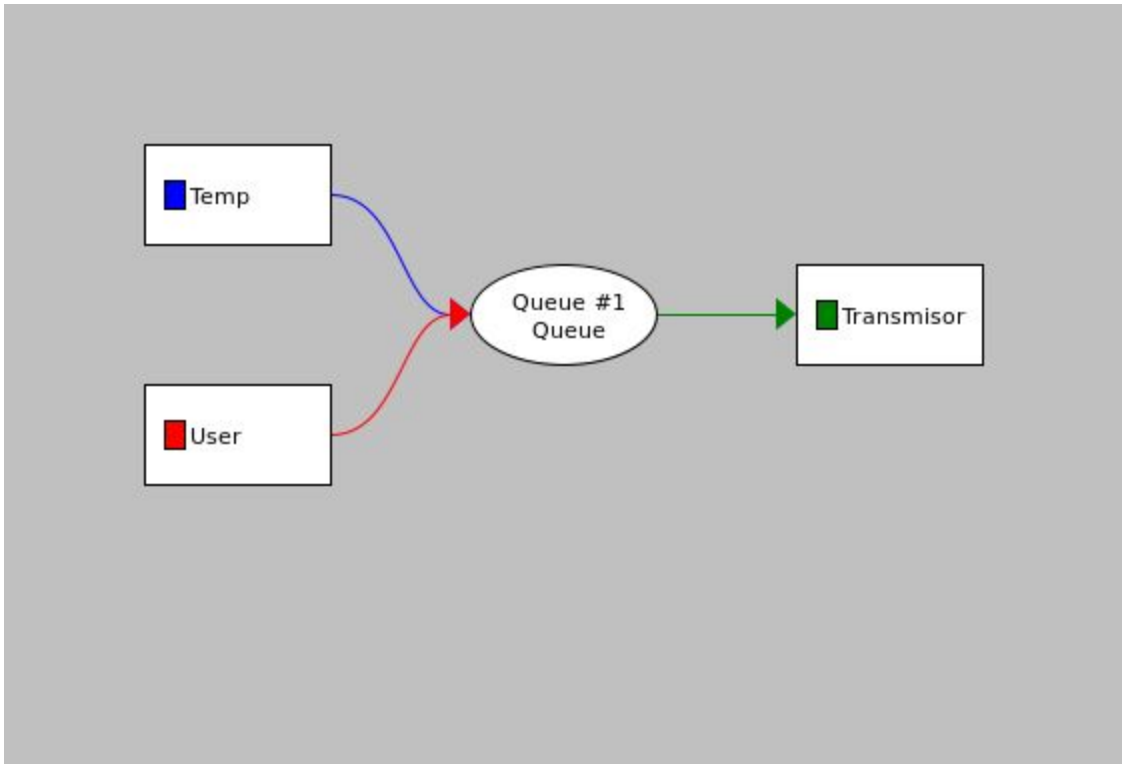

Figura 6. Actores

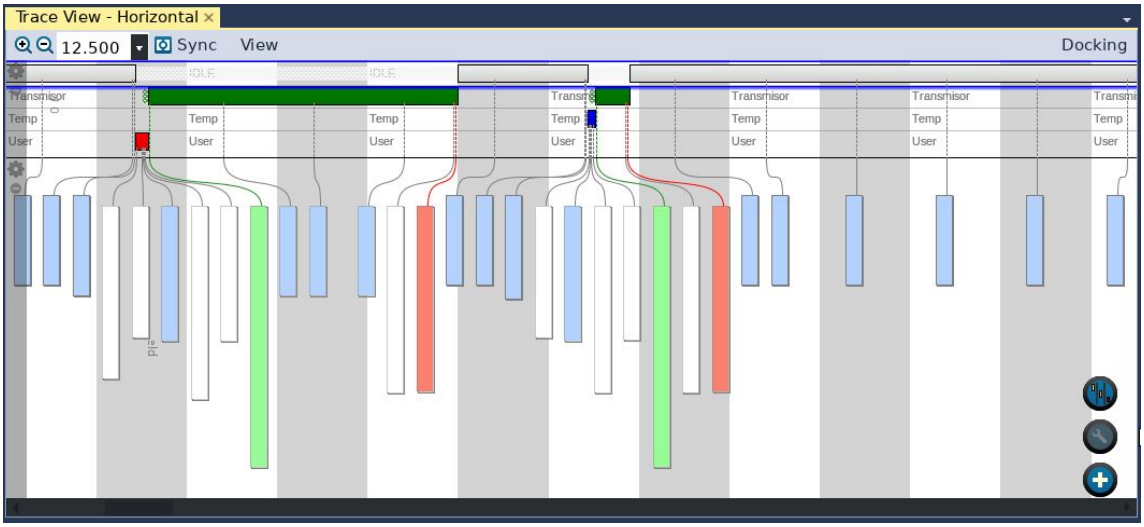

Figura 7. Captura de la traza

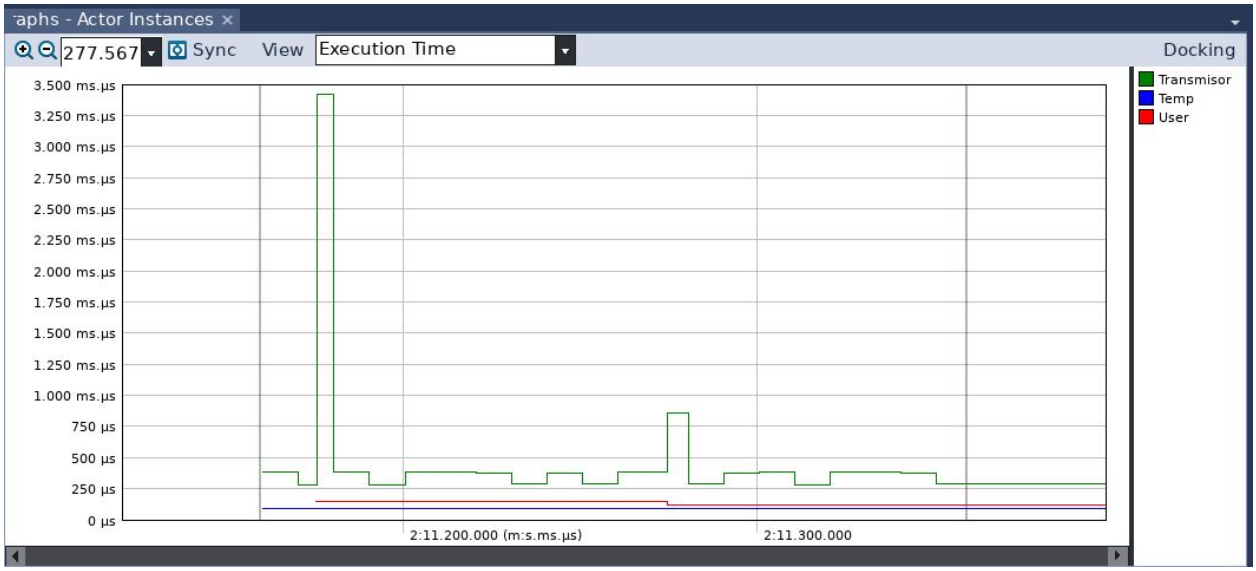

Figura 8. Tiempo de ejecución

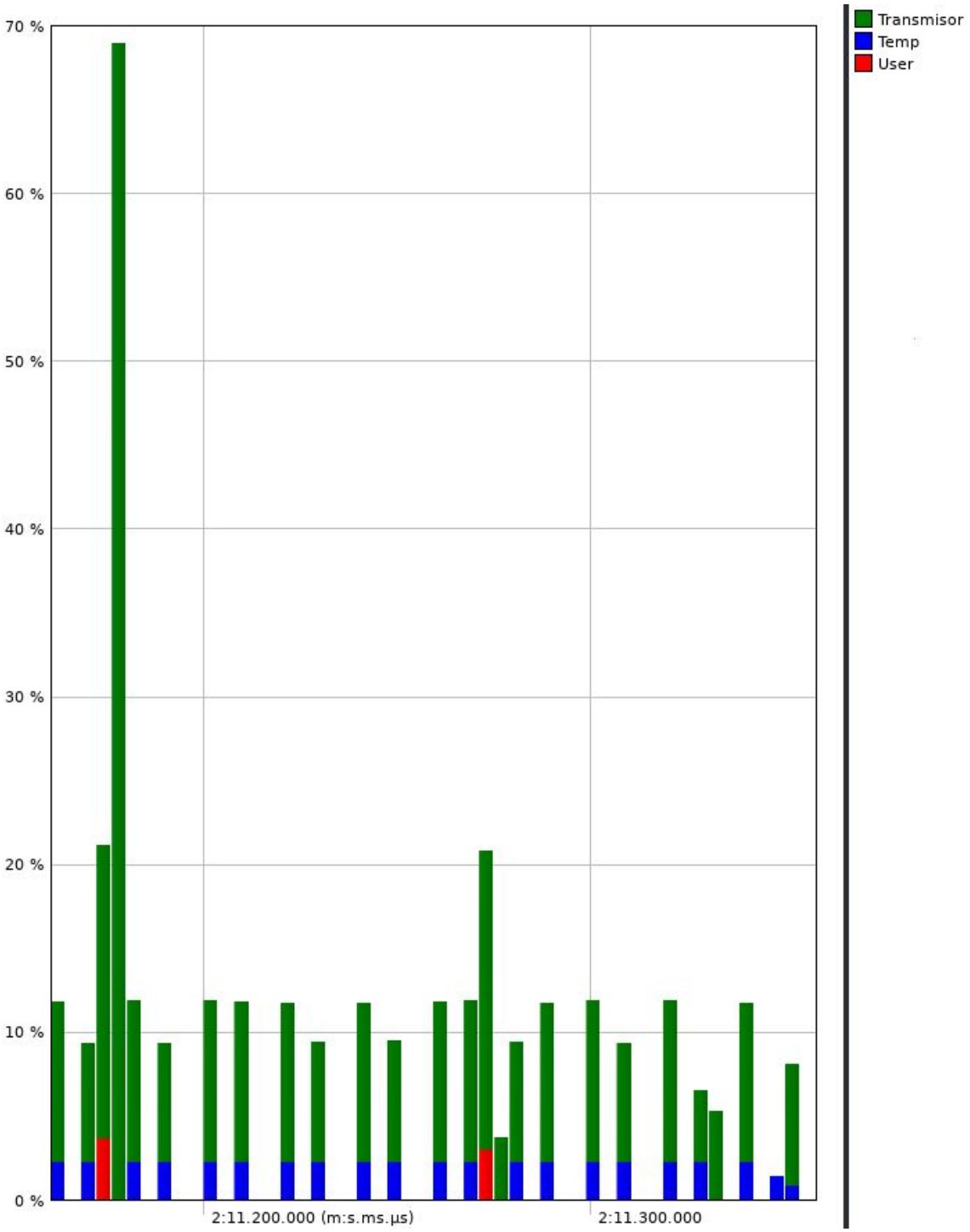

Figura 9. Tiempo de uso de CPU

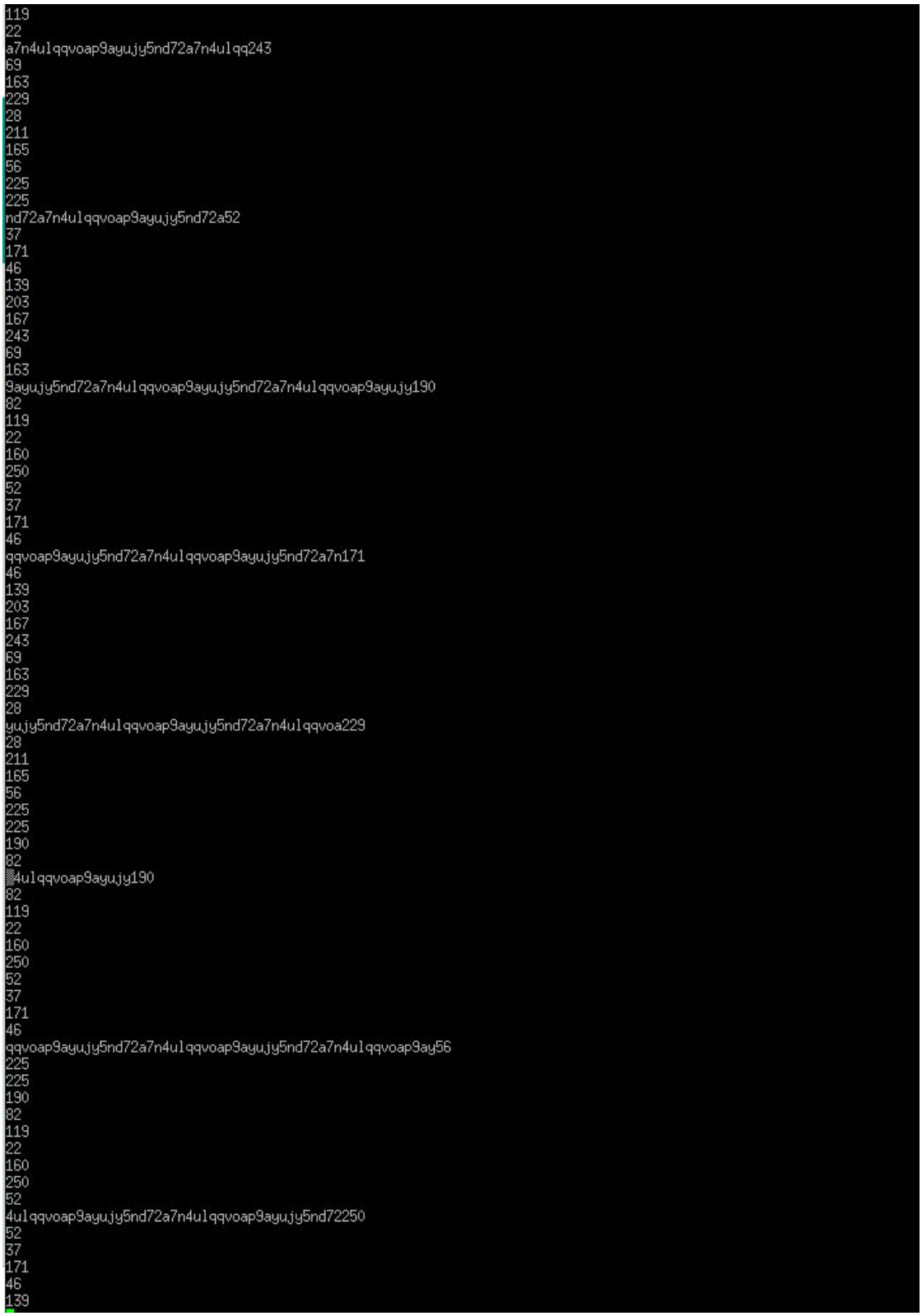

Figura 10. Recepción por puerto serie en la PC

Conclusiones

A modo de conclusión se puede decir que fue posible realizar el diseño e implementación de un software simple con tres tareas empleando freeRTOS y monitorear como el sistema operativo en tiempo real realiza el scheduling de las tareas instanciadas. Además fue de gran utilidad aprender a utilizar el software tracealyzer que permite debuggear cosas que de otra forma no sería posible