Ciencia Latina
Revista Multidisciplinar

DOI: https://doi.org/10.37811/cl_rcm.v7i1.4998

Comparativa y análisis de tiempos de respuesta y latencia, linux CNC vs. Rasberry Pi OS RT

Benjamín Vázquez Martínez
bvazquezm2100@alumno.ipn.mx
https://orcid.org/0000-0002-1345-5069
Instituto Politécnico Nacional - CDMX

Pedro Guevara López
pguevara@ipn.mx
https://orcid.org/0000-0001-5373-1403
Instituto Politécnico Nacional - CDMX

Leobardo Hernández González
lhernandezg@ipn.mx
https://orcid.org/0000-0002-4555-8695
Instituto Politécnico Nacional - CDMX

María Teresa Zagaceta Álvarez
mzagaceta@ipn.mx
https://orcid.org/0000-0003-1084-8569
Instituto Politécnico Nacional - CDMX

José David Muñiz Elizalde
jose.me@tlahuac.tecnm.mx
https://orcid.org/0000-0001-8407-4418
Tecnológico Nacional de México – CDMX

RESUMEN

En un sistema en tiempo real no crítico, se emplean utilizando computadoras de placa reducida como son las SBC Raspberry PI por su tamaño compacto y gran desempeño en sus distribuciones de SO Linux. Esto permite utilizar diferentes tipos de sistemas operativos con kernel en tiempo real dependiendo de sus características propias. En este artículo se comparan 2 tipos de kernel utilizando Linux CNC y Raspberry PI OS con PREEMPT RT, utilizando como banco de pruebas la herramienta Cyclictest para la comparativa de latencia y un proceso computacional de complejidad O^3 para la comparativa de tiempos de respuesta.

Palabras clave: *STR; kernel; SBC; latencia; complejidad*

Correspondencia: bvazquezm2100@alumno.ipn.mx
Artículo recibido 19 enero 2023 Aceptado para publicación: 19 febrero 2023
Conflictos de Interés: Ninguna que declarar
Todo el contenido de **Ciencia Latina Revista Científica Multidisciplinar**, publicados en este sitio están disponibles bajo Licencia Creative Commons.
Cómo citar: Vázquez Martínez, B., Guevara López, P., Hernández González , L., Zagaceta Álvarez, M. T., & Muñiz Elizalde, J. D. (2023). Comparativa y análisis de tiempos de respuesta y latencia, linux CNC vs. Rasberry Pi OS RT. Ciencia Latina Revista Científica Multidisciplinar, 7(1), 7673-7685. https://doi.org/10.37811/cl_rcm.v7i1.4998

Comparison and analysis of response times and latency, Linux CNC vs. Rasberry Pi OS RT

ABSTRACT

In a non-critical real-time system, they are used using small board computers such as the SBC Raspberry PI for its compact size and great performance in their Linux OS distributions. This allows the use of different types of real-time kernel operating systems depending on their own characteristics. In this article we compare 2 types of kernel using Linux CNC and Raspberry PI OS with PREEMPT RT, using as test bench the Cyclictest tool for the latency comparison and a computational process of O^3 complexity for the response time comparison.

Keywords: *STR, kernel; SBC latency; complexity*

INTRODUCCIÓN

Una SBC Raspberry PI modelo 4b es una computadora de placa reducida con las características de silencio al no utilizar un ventilador, conexión por medios inalámbricos integrados, con puertos USB y una variedad en la selección de la memoria RAM con la disponibilidad de 1, 2, 4 y 8 GB. Además, cuenta con 2 puertos micro-HDMI que soporta 4K. Mientras que el procesador es un ARM Cortex A72 con 4 núcleos y 1.5 Ghz. (Raspberry Pi, 2022), (SEGURA, 2018)

Un kernel (Duany, Muñoz, & Morera, 2007) en tiempo real es el encargado de administrar las tereas mediante prioridades, colocando las tareas que deben ser procesadas primero con una prioridad alta, mientras que las tareas menos importantes para el STR les proporciona una prioridad menor. (Ramos, 2004)

Los sistemas en tiempo real (STR) críticos y no críticos tienen ciertas restricciones temporales para la ejecución de sus procesos, esto permite un alto tiempo de respuesta para los procesos que realiza con sus entradas y salidas de la SBC Raspberry PI. Permitiendo que su kernel administre los procesos por su prioridad asignada.

El SO de Raspberry Pi es un sistema gratuito basado en debían con una optimización para los dispositivos montados en la SBC, oficialmente el SO recomendado para un buen funcionamiento del hatware de la Raspberry. Además, de contar con 35,000 para fácil instalación en la SBC de las herramientas disponibles. (Raspberry Pi Documentation, 2022)

De acuerdo con (LinuxCNC Documents, 2022) Linux CNC es un Sistema operativo de código abierto que permite el acceso a herramientas libres y disponibles, ya sea un pc de escritorio o una SBC las herramientas pueden ejecutarse en tiempo real con ayuda de su kernel RT ya instalado en el SO.

METODOLOGÍA

En esta investigación, las restricciones temporales por comparar son el tiempo de respuesta y la latencia: El tiempo de respuesta o tiempo de ejecución, de acuerdo con (medel, 2004), es el tiempo en que una tarea de tiempo real cumple su cometido sin interrupciones, que depende del hardware y software. Por otra parte, la latencia, conforme a (medel, 2004), es el tiempo que el sistema operativo tarda en dar el control a la rutina de interrupción a partir de ésta. La comparación de estos factores es de gran importancia, ya que permite determinar la capacidad del sistema operativo para

inicializar variables y generar el entorno necesario para la ejecución de código. Por esta razón, se consideraron ambas restricciones temporales para comparar el desempeño de los dos sistemas operativos.

Antes de comenzar con la preparación de las pruebas en la comparación de estos dos sistemas operativos en tiempo real, se muestra la Tabla 1 en la cual se observan las versiones, kernel, precio, compatibilidad de plataformas, tipo y dificultad de instalación de los dos sistemas operativos utilizados.

Tabla 1.- *Comparación de sistemas operativos*

	Linux CNC 2.8.1 para Raspberry Pi 4	Raspberry Pi OS (10 abril 2022 con parche PREEMPT_RT)
Sistema operativo base	Debian 10 Buster	Debian 11 Bullseye
Versión de kernel	4.19.71-rt24-v71+	5.4.81-rt45-v8+
Precio:	Gratuito	Gratuito
Plataformas compartibles:	32 bit,64 bit, ARM x64	32 bits, ARM x64
Instalación:	Descompresión de imagen IMG en memoria microSD mediante Raspberry Pi Imager	Descompresión de imagen IMG en memoria microSD mediante Raspberry Pi Imager y parcheo de kernel RT
Dificultad de instalación	Baja	Media

Preparación de LINUX CNC

Para iniciar el proceso de instalación de Linux CNC se tienen que seguir el siguiente procedimiento:

Paso 1:

Descargar la imagen que tiene como nombre, LinuxCNC “2.8.1 Raspberry Pi OS based on Debian 10 Buster “en la página oficial de Linux CNC: (Linux CNC, 2022)

Paso 2:

Montar la imagen en la memoria SD clase 10, cualquier herramienta que realice el procedimiento lo puede realizar, en este caso, se ocupó la herramienta de Rasberry Pi Imager, el cual está disponible para Windows, macOS y GNU/Linux.

Paso 3:

Instalar la memoria en la placa de SBC y encenderla.

Paso 4:

Abrir consola y escribir el siguiente comando, la placa debe responder con lo que se muestra en la Figura 1

Figura 1.- *Ejecución de comando para obtener información del SO CNC*

```
pi@linuxcnc:~ $ uname -a
Linux linuxcnc 4.19.71-rt24-v7l+ #1 SMP PREEMPT RT Fri Jan
21:15:16 GMT 2021 armv7l GNU/Linux
```

Con lo que muestra en consola se puede observar que ya cuenta con un kernel en el sistema operativo en tiempo real.

Preparación de RT-LINUX con PREEMP_RT

Para iniciar el proceso de instalación de RT-LINUX con PREEMP RT se tienen que seguir el siguiente procedimiento:

Paso 1:

Es necesario realizar la carga del sistema operativo Raspberry Pi OS a una microSD de al menos 8 GB de capacidad usando el software Raspberry Pi Imager. En este paso es necesario elegir Raspberry Pi OS (32-bit) o Raspberry Pi OS (64-bit), en cualquiera de sus variantes.

Paso 2:

Una vez que esté preparada la imagen se procede a encender la SBC Raspberry Pi apoyándose de un monitor o de una sesión SSH.

Paso 3:

Se procede a iniciar y a ejecutar en la terminal la instrucción “uname –a”, para conocer el estado del kernel del sistema operativo tal como se muestra en la Figura 2.

Figura 2.- Ejecución de comando para obtener información del SO Raspberry Pi

```
bcasillas@raspberrypi:~ $ uname -a
Linux raspberrypi 5.10.92-v8+ #1514 PREEMPT Mon Jan 17
17:39:38 GMT 2022 aarch GNU/Linux
```

Paso 4:

Posteriormente se requieren ingresar los siguientes comandos que aparecen en la Figura 3 en la terminal de la SBC Raspberry Pi. A continuación, se aprecia en la Figura 3 los comandos ingresados en la terminal de la SBC Raspberry Pi.

Figura 3.- *Comandos para la instalación del kernel PREEMPT-RT en SO raspberry Pi*

```
1.    cd /tmp
2.    wget
      https://github.com/kdoren/linux/releases/download/5
      .4.81-rt45/kernel8-5.4.81-rt.tgz
3.    tar xzf kernel8-5.4.81-rt.tgz
4.    cd boot
5.    sudo cp -rd * /boot/
6.    cd ../lib/
7.    sudo cp -dr * /lib/
8.    cd ../overlays
9.    sudo cp -dr * /boot/overlays
10.   cd ../broadcom
11.   sudo cp -dr bcm* /boot/
12.   cd /boot/
13.   sudo nano config.txt
14.   bcasillas@raspberrypi:~ $ cd /tmp
15.   bcasillas@raspberrypi:/tmp $
16.   wget  https://github.com/kdoren/linux/releases/download/5.4.81-
      rt45/kernel8-5.4.81-rt.tgz
17.   bcasillas@raspberrypi:/tmp $ tar xzf kernel8-5.4.81-rt.tgz
18.   bcasillas@raspberrypi:/tmp $ cd boot
19.   bcasillas@raspberrypi:/tmp/boot $ sudo cp -rd * /boot/
20.   bcasillas@raspberrypi:/tmp/boot $ cd ../lib/
21.   bcasillas@raspberrypi:/tmp/lib $ sudo cp -dr * /lib/
22.   bcasillas@raspberrypi:/tmp/lib $ cd ../overlays
23.   bcasillas@raspberrypi:/tmp/overlays   $   sudo   cp   -dr   *
      /boot/overlays
24.   bcasillas@raspberrypi:/tmp/overlays $ cd ../broadcom
25.   bcasillas@raspberrypi:/tmp/broadcom $ sudo cp -dr bcm* /boot/
26.   bcasillas@raspberrypi:/tmp/broadcom $ cd /boot/
27.   bcasillas@raspberrypi:/boot $ sudo nano config.txt
```

De la instrucción 1 a 3 se realiza la descarga del kernel en una carpeta especificada y su extracción, de la instrucción 4 a 11 se realiza la copia recursiva de los archivos extraídos a las carpetas destino preservando los enlaces simbólicos. En la línea 12 se entra en la carpeta /boot/ del sistema operativo y en la línea 13 se edita el archivo de configuración.

Paso 5:

Al ingresar el último comando (sudo nano config.txt), recorremos el archivo de texto hasta encontrar el código mostrado en la Figura 4

Figura 4.- *Texto información en config.txt*

```
[pi4]
#Run as fast as firmware / board allows
arm boost=1
```

En el párrafo mostrado en la Figura 5 se añaden las siguientes líneas de instrucciones

Figura 5.- *Texto de información a escribir dentro de config.txt*

```
arm_64bit=1
kernel=kernel8_rt.img
```

El párrafo completo debe apreciarse como se muestra en la Figura 6:

Figura 6.- *Texto final dentro de config.txt*

```
[pi4]
#Run as fast as firmware / board allows
arm_boost=1
arm 64bit=1
```

Se guardan los cambios y se procede a reiniciar Raspberry Pi OS mediante el comando sudo reboot, tras volver a ingresar las credenciales se introduce el comando uname -a y se aprecia el resultado de la Figura 7.

Figura 7.- *Reinicio del sistema y obtención de la información del SO*

```
bcasillas@raspberrypi:/boot $ sudo reboot
bcasillas@raspberrypi:~ $ uname -a
Linux raspberrypi 5.4.81-rt45-v8+ #9 SMP
PREEMPT_RT Mon Dec 28 00:13:29 PST 2020
aarch64
```

Ahora es posible observar que el estado del kernel incluye el sufijo RT, con lo cual, se confirma la presencia de un kernel de tiempo real en el sistema operativo Raspberry Pi OS.

Instalación de Cyclictest (linuxfoundation, 2022) Para la instalación de esta prueba se ejecutan los siguientes comandos que muestra la figura 7.

Figura 7.- *Comandos de instalación Cyclictest*

```
git                                    clone
git://git.kernel.org/pub/scm/utils/rt-
tests/rt-tests.git
cd rt-tests
git checkout stable/v1.0
make all
make install
```

El último paso es opcional (ver figura 8).

Figura 8.- *Comando opcional en la instalación de Cyclictest*

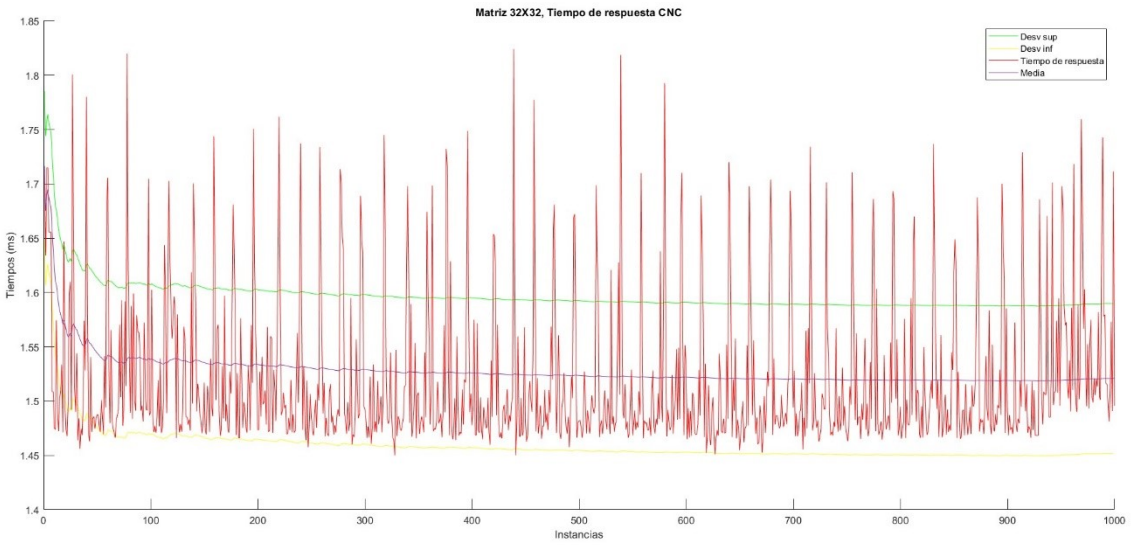
```
make cyclictest
```

En algunos SO RT no es necesario instalar ya que viene por defecto.

RESULTADOS Y DISCUSIÓN

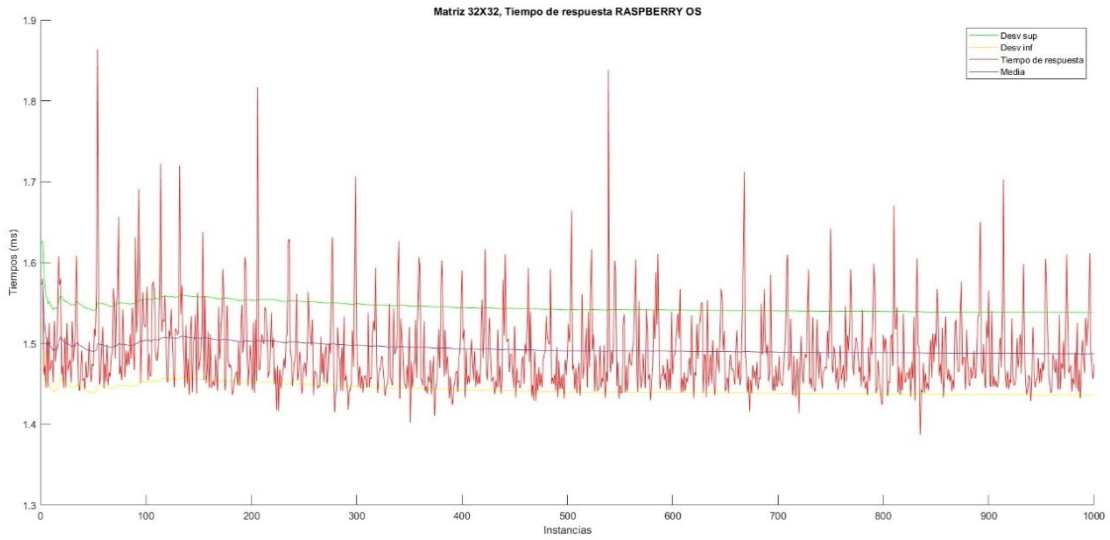
Para Linux CNC, en la realización de una matriz con tamaño 32x32, se puede apreciar que los experimentos arrojan instancias de 1.85ms aproximadamente mientras que en las instancias de menor valor se aproximan a 1.45ms, en la Figura 8 se aprecia el tiempo de respuesta de la matriz de tamaño de 32X32.

Figura 8.- *CNC matriz de tamaño 32X32*



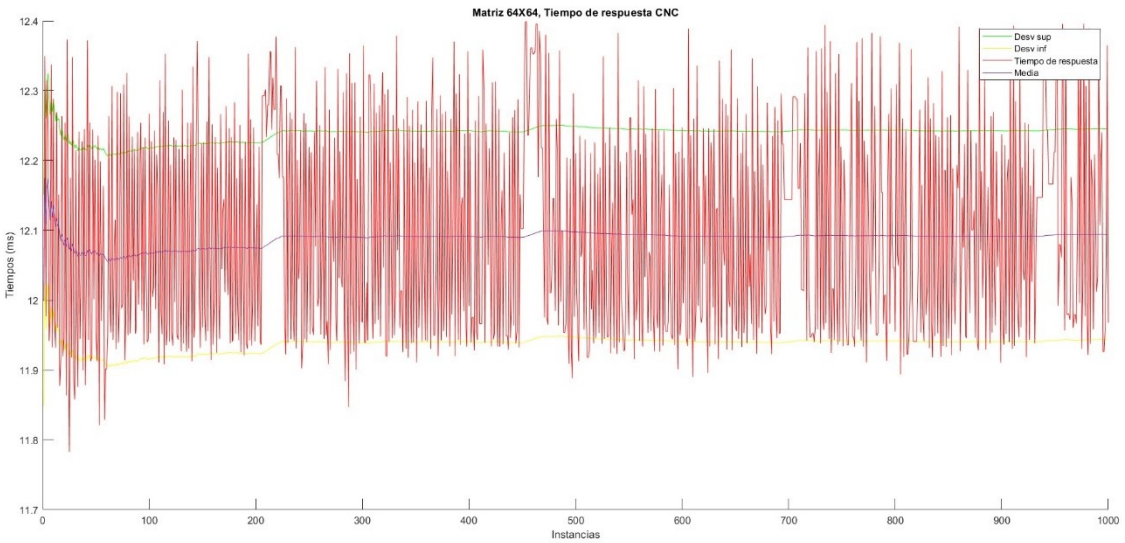
En una matriz de tamaño 32x32 aplicada a 4 experimentos se puede observar del algoritmo de inversión de matrices en Raspberry Pi OS con parche PREEMPT RT datos diferentes con respecto a Linux CNC, mostrando variaciones antes de la instancia 300, observar la Figura 9.

Figura 9.- *Raspberry OS matriz de tamaño 32X32*



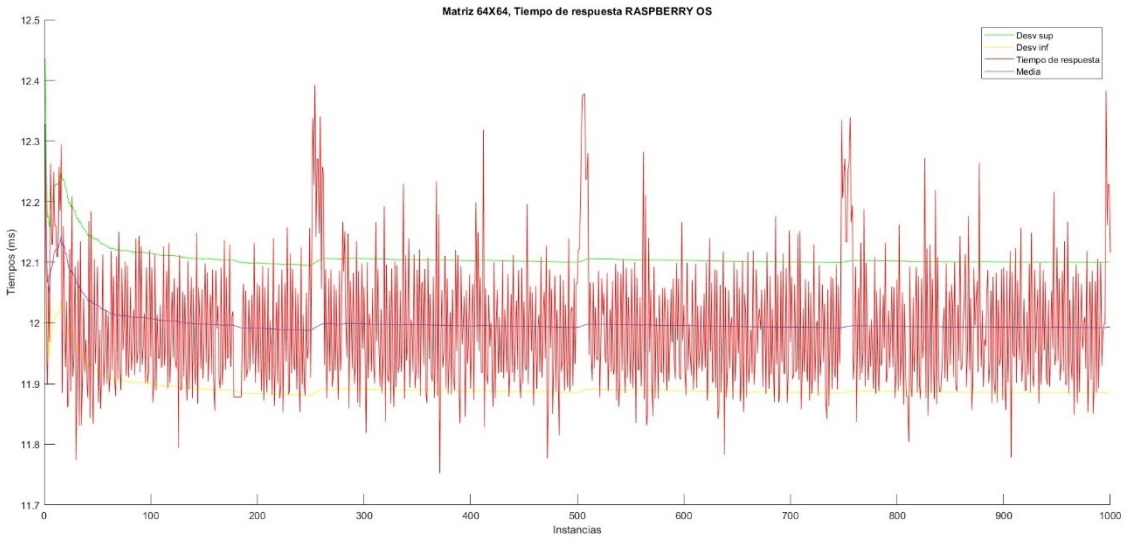
En el caso de la ejecución de una matriz de tamaño 64x64 se observa un tiempo de respuesta mayor, teniendo más instancias de aproximadamente 12.5, como se puede observar en la Figura 10. Mientras que para las instancias contrarias en menor tiempo de respuesta son de aproximadamente 11.9; cada uno de ellos se mantiene en un valor aproximado de 12.1ms.

Figura 10.- *CNC matriz de tamaño 64X64*



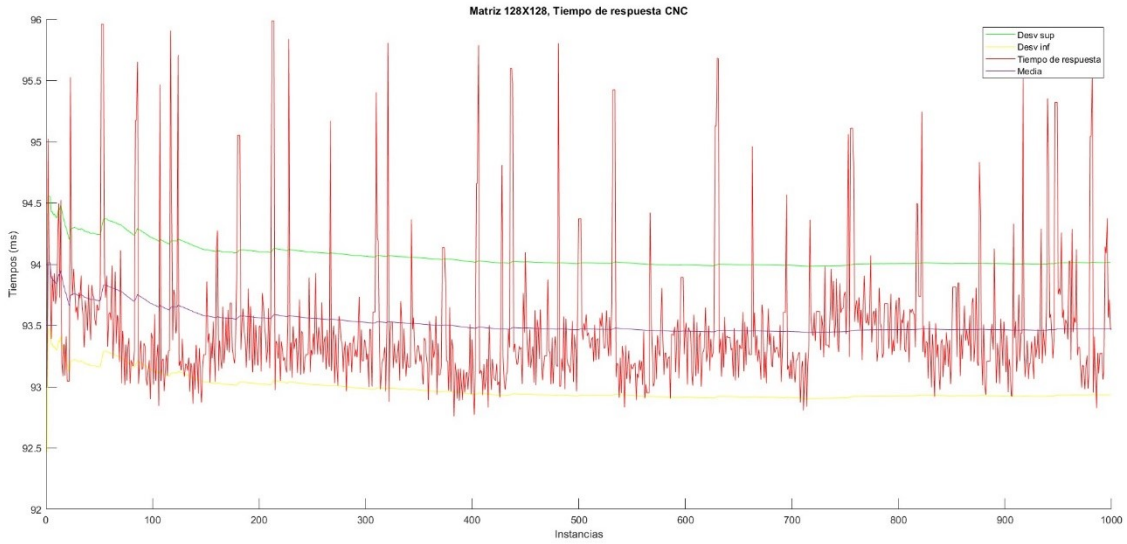
En la ejecución del algoritmo de 64x64 con los mismos experimentos en PREEMPT RT se ubican diferencias entre el experimento 1 y 2 con menores picos altos de tiempo con ello teniendo menor tiempo de respuesta como se muestra en la Figura 11. Estos mismos al final mostraron tener picos con la mayor altura de tiempo de 12.15 aproximadamente.

Figura 11.- *Raspberry OS matriz de tamaño 64X64*



Con la implementación del algoritmo inversión de matrices de tamaño 128x128, los datos obtenidos son variantes, instancias entre 96ms y 93ms. Se observa que el tiempo de respuesta es alto y la media recursiva muestra una región acotada en la instancia 300, se alcanza a notar como el tiempo de respuesta comienza muy variable al iniciar el algoritmo obteniendo valores muy bajos a comparación con las demás a partir de la instancia 100. Véase la Figura 12.

Figura 12.- *CNC matriz de tamaño 128X128*



Para la ejecución del siguiente algoritmo de una matriz de 128x128 con PREEMPT RT como puede observarse en la Figura 13. Arroja picos de instancias altos de hasta 96ms, sus instancias con menor tiempo son de 93ms, teniendo una media recursiva después de la instancia 200.

Figura 13.- *Raspberry OS matriz de tamaño 128X128*

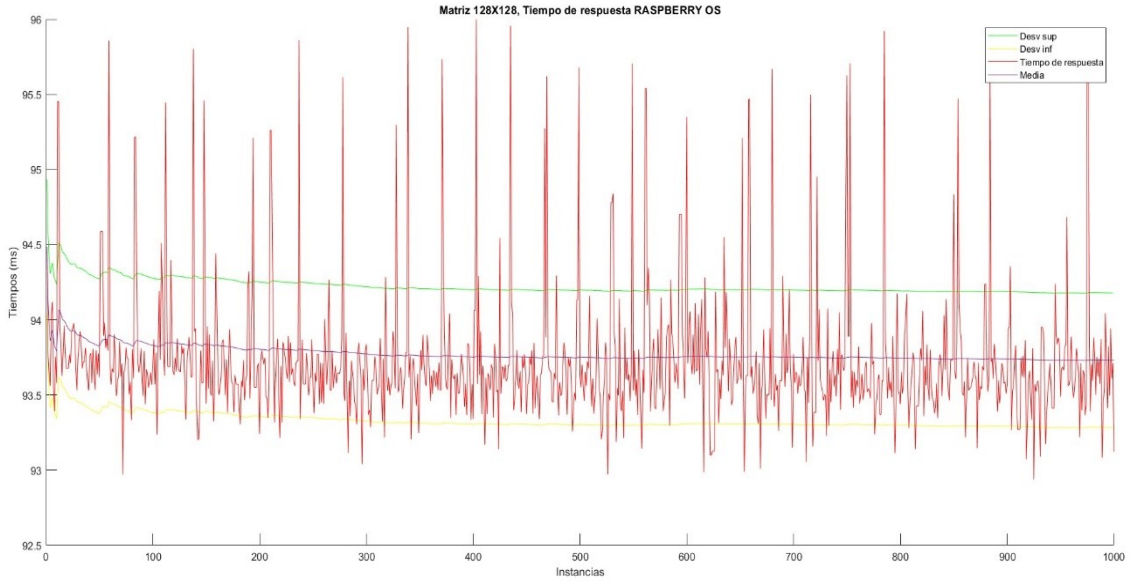
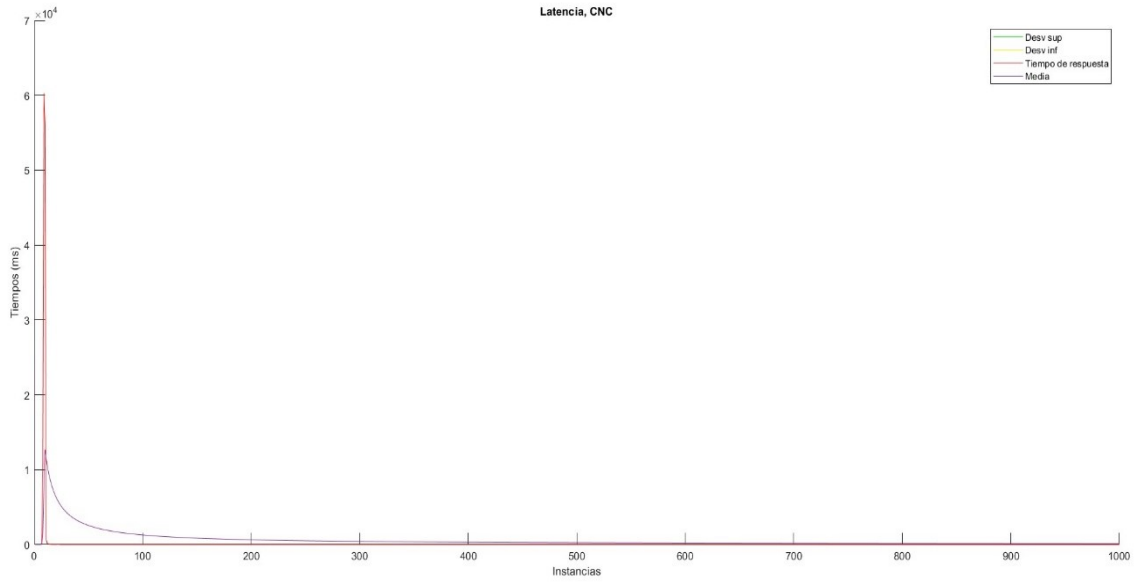
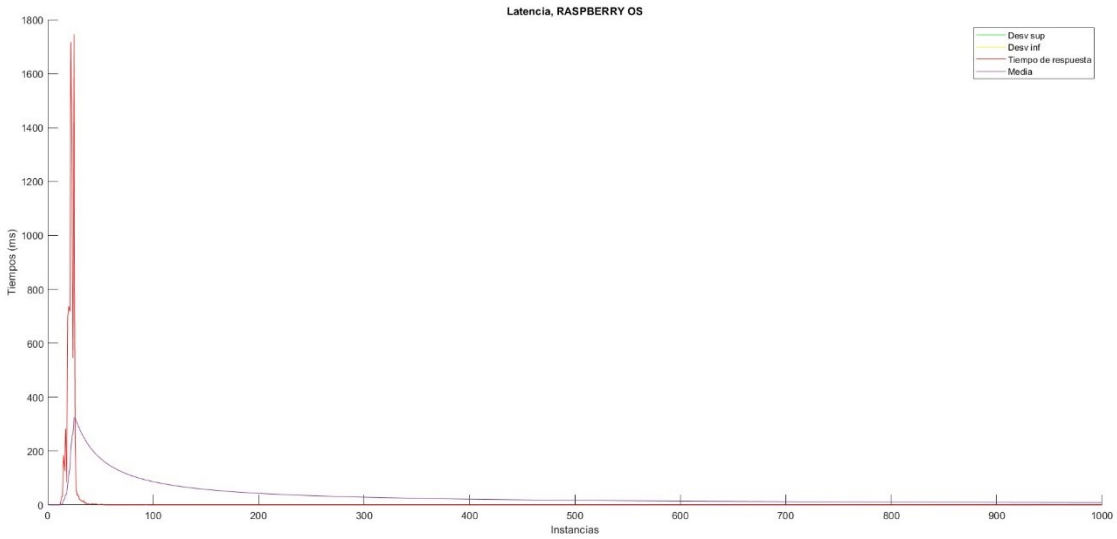


Figura 14.- *Latencia CNC con 1000 instancias*



Para la latencia en Linux CNC, se resalta un incremento muy alto sobrepasando los 60000ms, pero disminuye rápidamente en las instancias siguientes que muestran un valor de 0 o muy cercano a 0.

Figura 15.- Latencia Raspberry OS con 1000 instancias



Para Raspberry OS se observa una variación de tiempo entre las primeras muestras con algunos valores sobrepasando los 2000ms, mientras que al trascurrir las instancias toman valores de 0 o muy cercanos a 0.

CONCLUSIONES

Para los cálculos en la inversión de matrices se refleja un comportamiento esperado con respecto a los SO RT, en la inversión de la matriz 32 x32 Raspberry PI OS muestra una media de 1.5 mientras que CNC tiene una elevación de 0.05 dejando una media de 1.55 en sus primeras instancias, mientras que al transcurrir las instancias se observa una disminución de tiempo.

Los SO muestran comportamientos muy parecidos, algo que resalta el comportamiento de Raspberry PI OS es que los procesos de inversión de matrices los dejaba en tránsito por momentos, haciendo que el tiempo aumentara. Pero aun que entrarán en tránsito el tiempo de respuesta es menor comparado con el tiempo de CNC, con una diferencia de 0.05ms a 0.5ms.

La latencia en los SO muestra un aumento y una diminución rápida de tiempo, pero Raspberry PI OS muestra que una variación en los tiempos más que CNC, mientras que CNC muestra un tiempo superior de 60s. al que Raspberry PI no supera los 2.5s.

LISTA DE REFERENCIAS

AGAMA, L. A. (2022). *ANÁLISIS DE LA LATENCIA DEL SERVICIO DE INTERNET Y SU IMPACTO EN LAS APLICACIONES DE INTERNET DE LAS COSAS (IoT).* Lima - Perú.

Alimenti, O., Friedrich, G., Reggiani, G., Tonietti, S., Velazquez, G., & Cofre, L. (2012). Análisis del Tiempo de Respuesta en entorno de Tiempo Real sobre el MAC 802.11e. *Argentine Symposium on Technology*, 11.

Arenas, J. S., & Alberto J. Álvares, a. (2016). IMPLEMENTACIÓN DE LA TÉCNICA RETROFITTING PARA EL ROBOT ASEA IRB6-S2 USANDO LINUXCNC. *CONEM 2016*, 10.

Duany, E., Muñoz, J., & Morera, M. (2007). Aplicación de RT-Linux en el control de motores de pasos. *Instituto Superior Politécnico José Antonio Echeverría*, 5.

González, J. S., & Martínez, R. d. (2015 - 2016). *Interfaz software/hardware entre Raspberry Pi y FPGA Spartan6 y su aplicación a simulación dirigida por eventos.* MADRID: UNIVERSIDAD COMPLUTENSE DE MADRID.

GUAMÁN, J. F., & DÍAZ, L. E. (2019). *DISEÑO Y CONSTRUCCIÓN DE UNA SMART TV USANDO RASPBERRY PI COMO SISTEMA EMBEBIDO PARA QUE FUNCIONE CON UNA TV CONVENCIONAL.* Quito.

Linux CNC. (20 de 12 de 2022). Obtenido de http://linuxcnc.org/downloads/

LinuxCNC Documents. (11 de 12 de 2022). Obtenido de http://linuxcnc.org/

linuxfoundation. (25 de 12 de 2022). Obtenido de https://wiki.linuxfoundation.org/realtime/documentation/howto/tools/rt-tests#compile-and-install

medel, g. y. (2004). Modelo Dinámico para una Tarea en Tiempo Real. *Computación y Sistemas Vol. 8*, 13.

Perez-Monte, C. F., Mercado, G. J., Taffernaberry, J. C., & Piccoli, M. F. (2014). *Protocolo de comunicaciones para renderización distribuida en tiempo real.* XXXIV Congresso da Sociedade Brasileira de Computação.

Ramos, R. R. (2004). *Diseño e implementación de un Kernel de tiempo real para un DSP 56827.* Ciudad de México.

Raspberry Pi. (02 de 11 de 2022). Obtenido de https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

Raspberry Pi Documentation. (11 de 12 de 2022). Obtenido de https://www.raspberrypi.com/documentation/computers/os.html

SEGURA, D. F. (2018). DISEÑO DE UN SOFT-PLC BASADO EN UN COMPUTADOR. *UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS*, 83.