THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
JAPAN

17-19th, July, 2019

XENOMAI BASED REAL TIME MODEL WITHOUT USING RTOS

Pintu Kumar,
Software Architect,
Sony India Software Centre Pvt. Ltd., Bangalore
pintu.kumar@sony.com

THE LINUX FOUNDATION

AGENDA

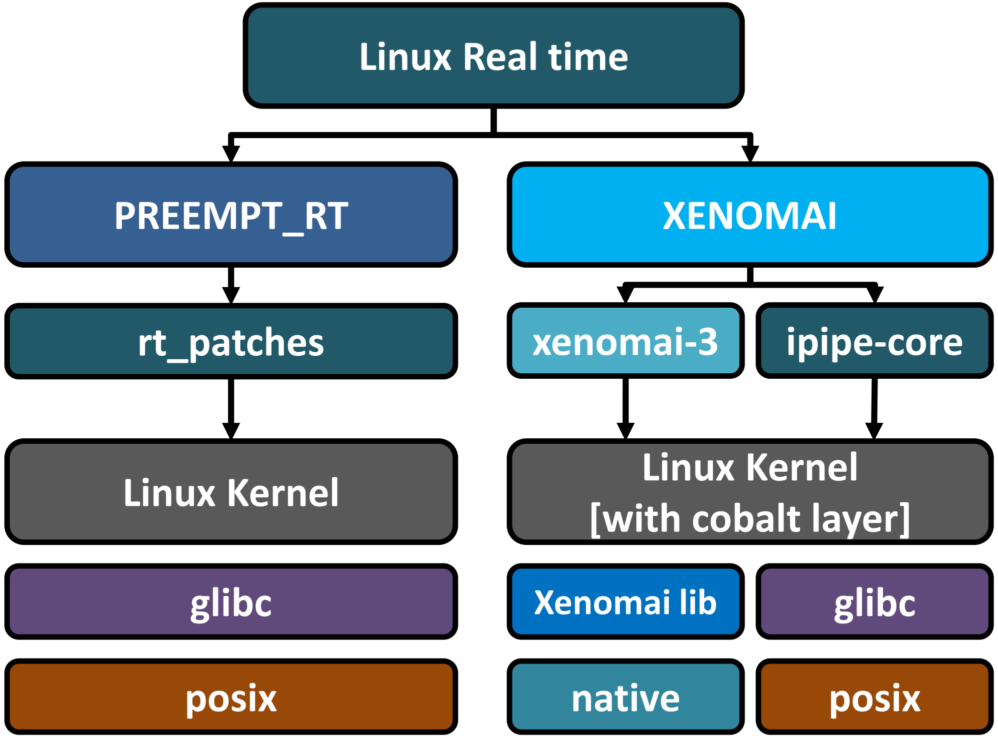
- INTRODUCTION
- XENOMAI INTEGRATION FOR RPI3
- OUR MODEL SETUP AND OVERVIEW
- FULL SYSTEM ARCHITECTURE
- EXPERIMENTATION RESULTS
- OBSERVATION AND IMPROVEMENT AREAS
- CONCLUSION
- REFERENCES

INTRODUCTION

- This talk is about designing a simple real-time model using Linux Kernel 4.9, Xenomai and Raspberry Pi 3 hardware.
- Demonstrate how this system behaves in a critical scenarios such as: sudden obstacle detection.

LINUX REAL TIME OPTIONS

- For Linux Real time we have 2 options:
  - ❑ PREEMPT_RT
  - ❑ XENOMAI
- Both requires Kernel changes. However some rt_patches already merged to mainline.
- Xenomai requires ipipe and cobalt layer.
- Both supports posix API. But Xenomai requires application to be re-build.
- Xenomai also supports alchemy layer for porting existing RTOS application.
- Currently Xenomai support is available only to limited SoC and Kernel version.


Linux Real time

PREEMPT_RT

XENOMAI

rt_patches

xenomai-3

ipipe-core

Linux Kernel

Linux Kernel
[with cobalt layer]

glibc

Xenomai lib

glibc

posix

native

posix

XENOMAI INTEGRATION

- **Download raspberry pi Kernel:**

```
git clone https://github.com/raspberrypi/linux    [branch: rpi-4.9.y]
```

- **Download ipipe patch for 4.9 Kernel from here:**

https://xenomai.org/downloads/ipipe/v4.x/arm/

[ipipe-core-4.9.51-arm-4.patch]

- **Download Xenomai-3 repository from here:**

```
git clone http://git.xenomai.org/xenomai-3.git
```

XENOMAI KERNEL INTEGRATION

- **Create Xenomai-3 Kernel patches:**

```
# cd xenomai-3
# ./scripts/prepare-kernel.sh --arch=arm --linux=<pi3 kernel path> --outpatch=rpi3-xenomai-3-kernel-4.9.80.patch
```

- **First apply ipipe-core patches to pi3 Kernel:**

```
# patch –p1 < ipipe-core-4.9.51-arm-4.patch
```

- **Then apply xenomai-3 Kernel patches:**

```
# patch –p1 < rpi3-xenomai-3-kernel-4.9.80.patch
```

- Disable CPU_IDLE, CPU_FREQ, KGDB, etc.
- Create a new config file for Xenomai.
- Build the raspberry-pi3 Kernel normally using the new config.
- Disable Desktop, Wi-Fi, Bluetooth.
- Enable ssh and remove all USB devices.

Note: You may need to fix hunk errors and build issues

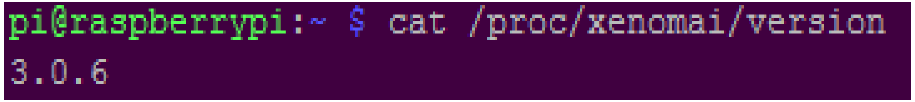
XENOMAI USER SPACE INTEGRATION

- **Build and install xenomai-3 on raspberry pi:**

```
# cd xenomai-3
# ./scripts/bootstrap
# ./configure --enable-smp
//To cross-compile we can use this:
# ./configure --enable-smp --host=arm-linux-gnueabihf
CFLAGS="-march=armv7-a" LDFLAGS="-march=armv7-a“
# make -j8
# make install
[Default installation path: /usr/xenomai/]
```

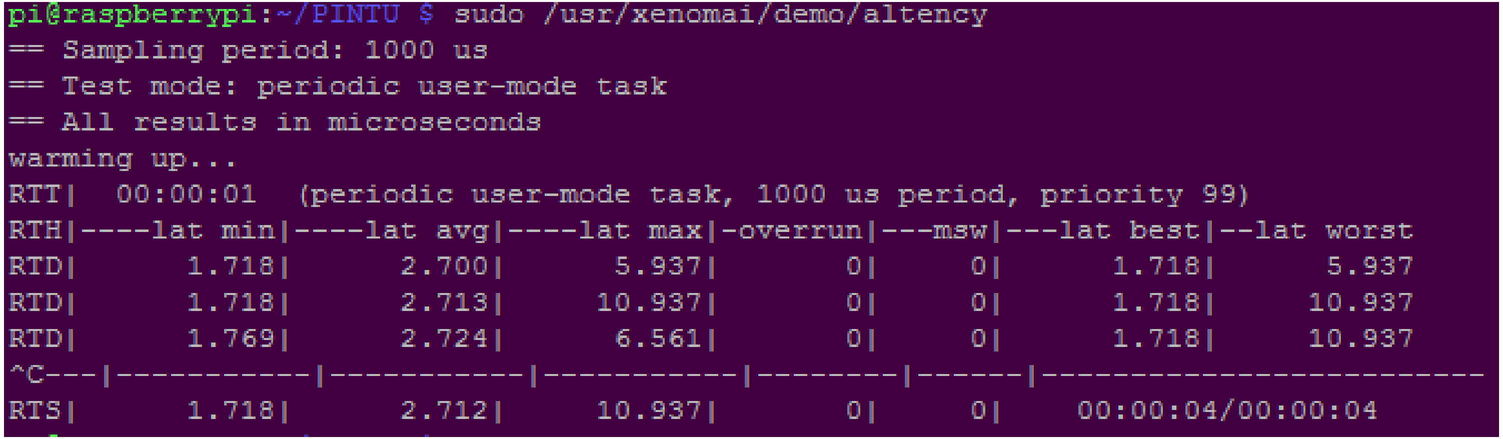
- **Verify that Xenomai is up and running on raspberry pi:**

# cat /proc/xenomai/version



```
pi@raspberrypi:~ $ cat /proc/xenomai/version
3.0.6
```

# /usr/xenomai/demo/altency

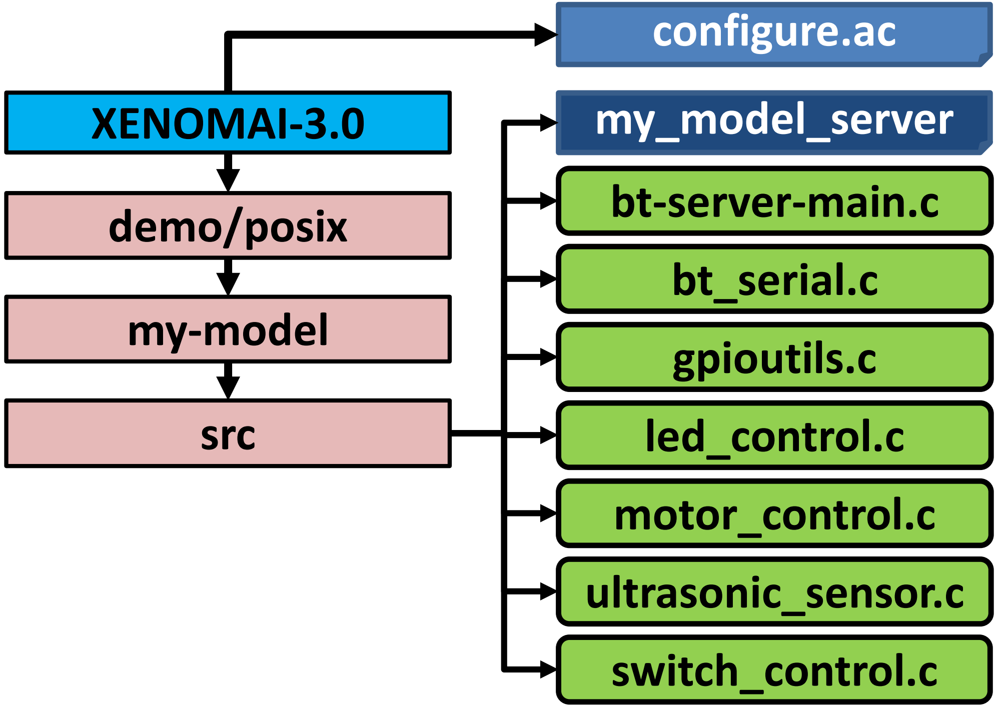


```
pi@raspberrypi:~/PINTU $ sudo /usr/xenomai/demo/altency
== Sampling period: 1000 us
== Test mode: periodic user-mode task
== All results in microseconds
warming up...
RTT|  00:00:01  (periodic user-mode task, 1000 us period, priority 99)
RTH|----lat min|----lat avg|----lat max|-overrun|---msw|---lat best|--lat worst
RTD|      1.718|      2.700|      5.937|       0|     0|      1.718|      5.937
RTD|      1.718|      2.713|     10.937|       0|     0|      1.718|     10.937
RTD|      1.769|      2.724|      6.561|       0|     0|      1.718|     10.937
^C---|-----------|-----------|-----------|--------|------|-------------------------
RTS|      1.718|      2.712|     10.937|       0|     0|    00:00:04/00:00:04
```

SETTING UP OUR MODEL

- **Integrating and building our model repo:**

# Edit configure.ac to include our model
# demo/posix/my-model/Makefile
# Update Makefile.am in each folder
# build xenomai normally as shown earlier
# This should create:
my_model_server as final output for our model

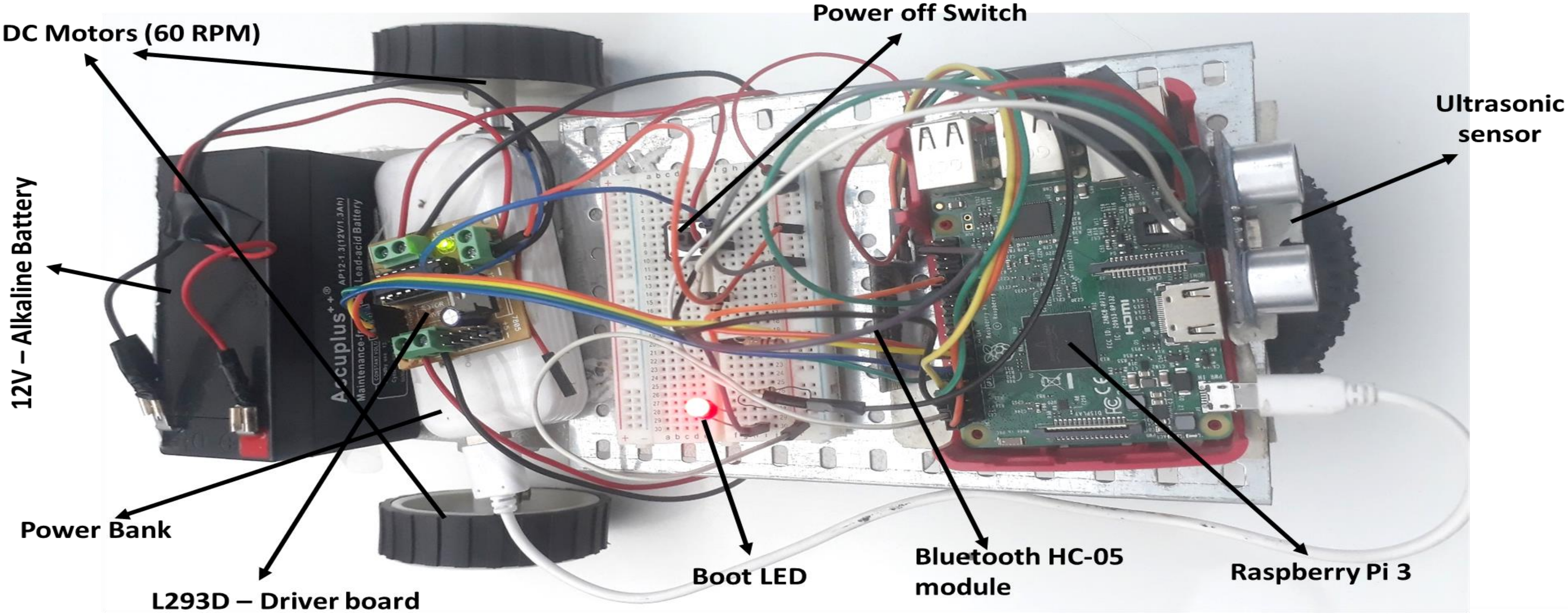


- **Setting up our model on target:**

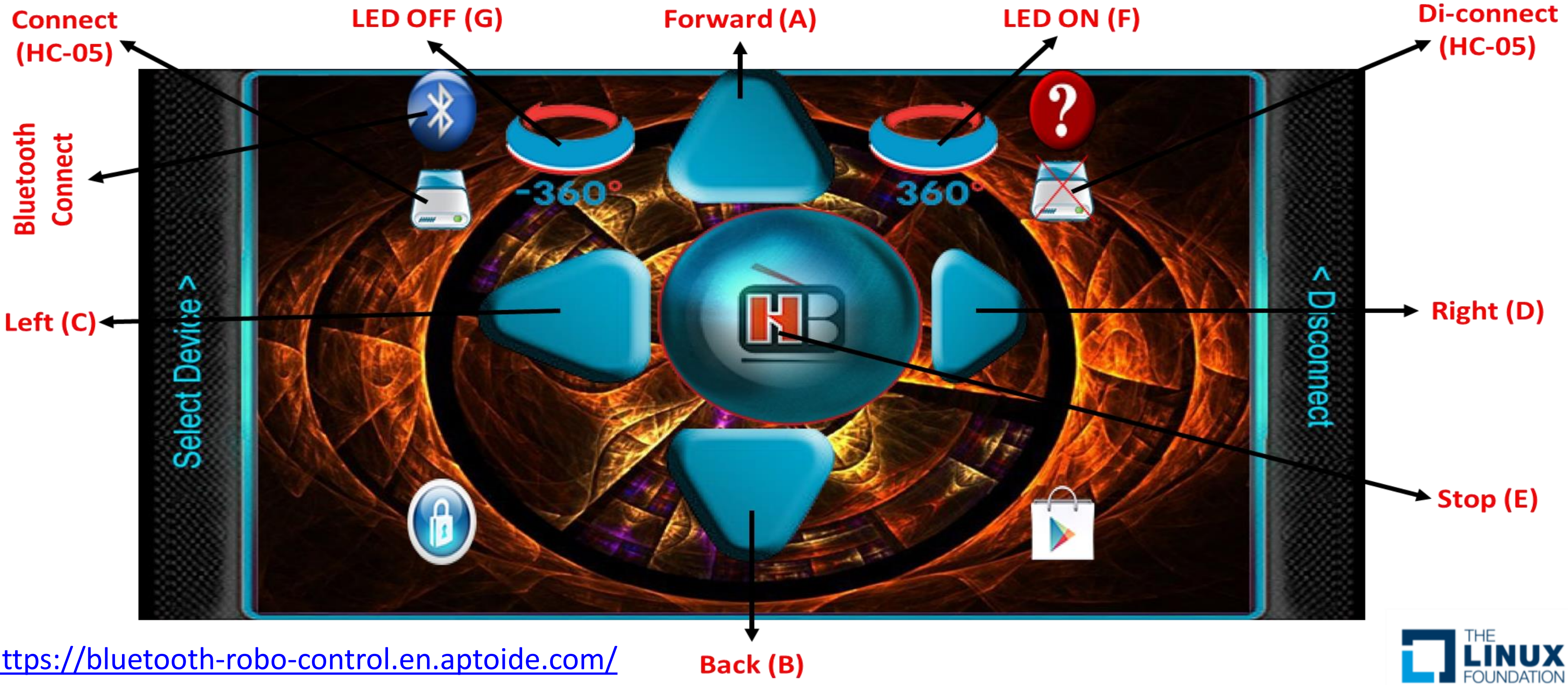
# copy **my_model_server** to /usr/xenomai/bin
# Comment this line in: /lib/systemd/system/serial-getty@.service
# ExecStart=-/sbin/agetty --keep-baud 115200,38400,9600 %I $TERM

# Create new systemd service (**bt-model-service**) and add our executable:
# ExecStart=/usr/xenomai/bin/ **my_model_server**

HARDWARE MODEL OVERVIEW

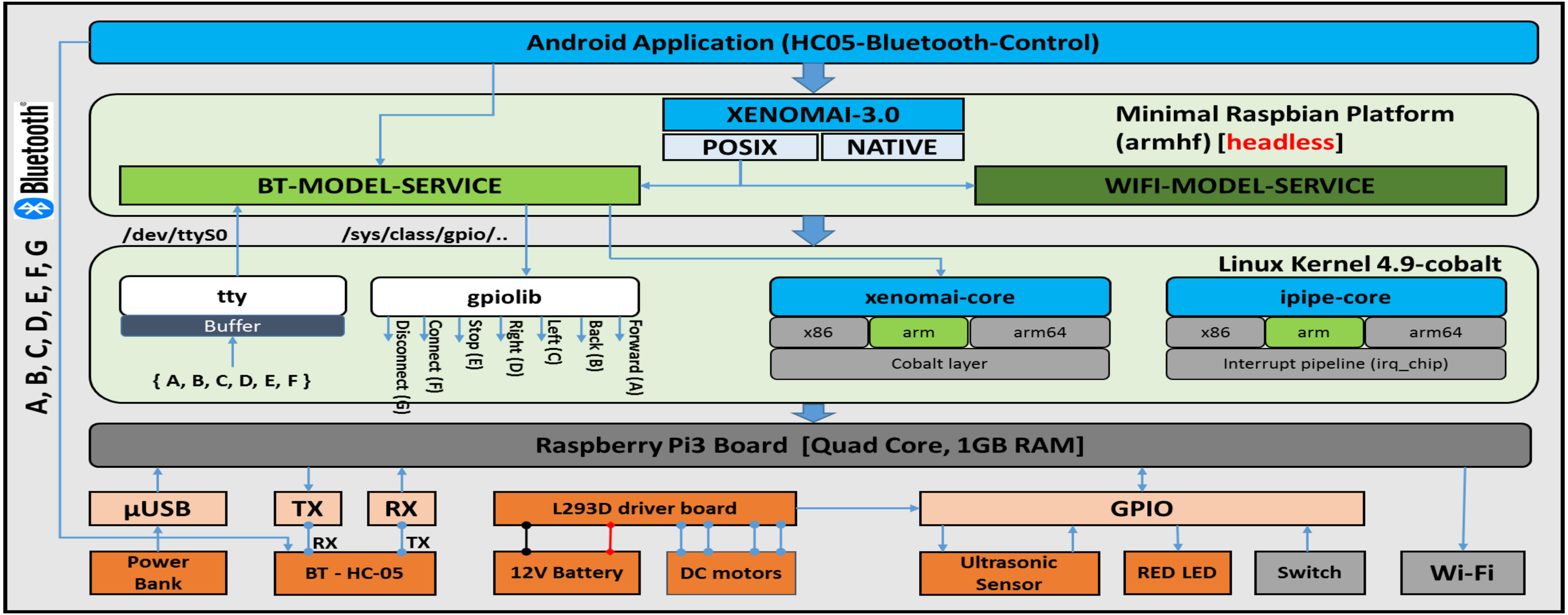


ANDROID APPLICATION OVERVIEW



https://bluetooth-robo-control.en.aptoide.com/

FULL SYSTEM ARCHITECTURE



Android Application (HC05-Bluetooth-Control)

Bluetooth

A, B, C, D, E, F, G

XENOMAI-3.0

POSIX

NATIVE

Minimal Raspbian Platform (armhf) [headless]

BT-MODEL-SERVICE

WIFI-MODEL-SERVICE

/dev/ttyS0

/sys/class/gpio/..

Linux Kernel 4.9-cobalt

tty

Buffer

{ A, B, C, D, E, F }

gpiolib

Disconnect (G)

Connect (F)

Stop (E)

Right (D)

Left (C)

Back (B)

Forward (A)

xenomai-core

x86

arm

arm64

Cobalt layer

ipipe-core

x86

arm

arm64

Interrupt pipeline (irq_chip)

Raspberry Pi3 Board [Quad Core, 1GB RAM]

µUSB

TX

RX

L293D driver board

GPIO

RX

TX

Power Bank

BT - HC-05

12V Battery

DC motors

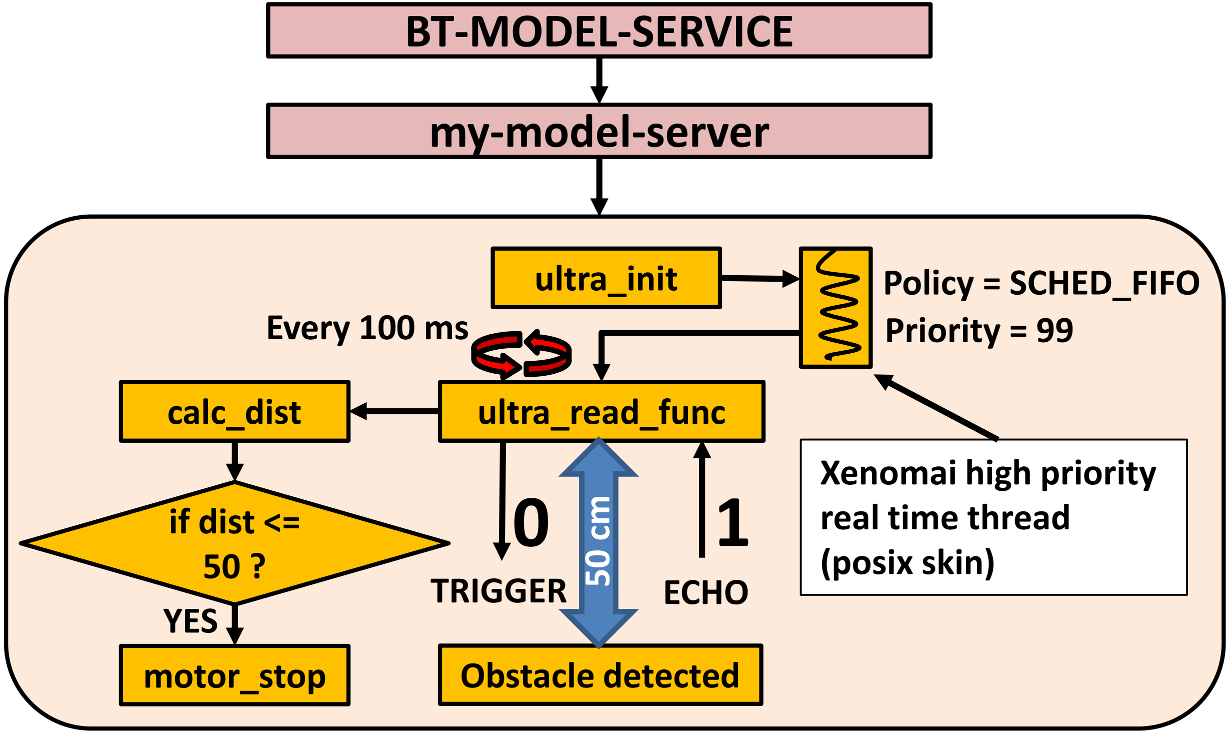
Ultrasonic Sensor

RED LED

Switch

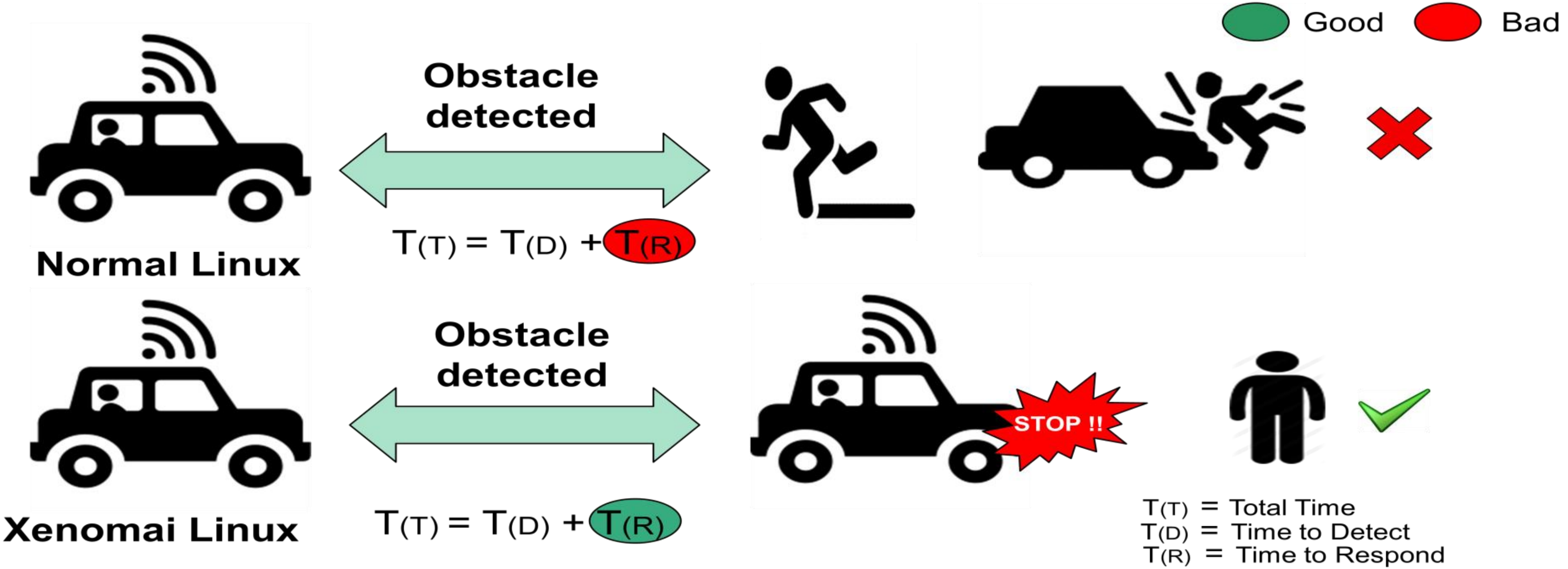
Wi-Fi

SUDDEN OBSTACLE HANDLING



- Not everything needs to be real time in a system.
- Identify the most critical part of your system.
- Convert only that portion to high real time.
- Here only the ultrasonic sensor handling thread should be high priority.
- motor_stop is just an example about action taken.

EXAMPLE USECASE SCENARIO

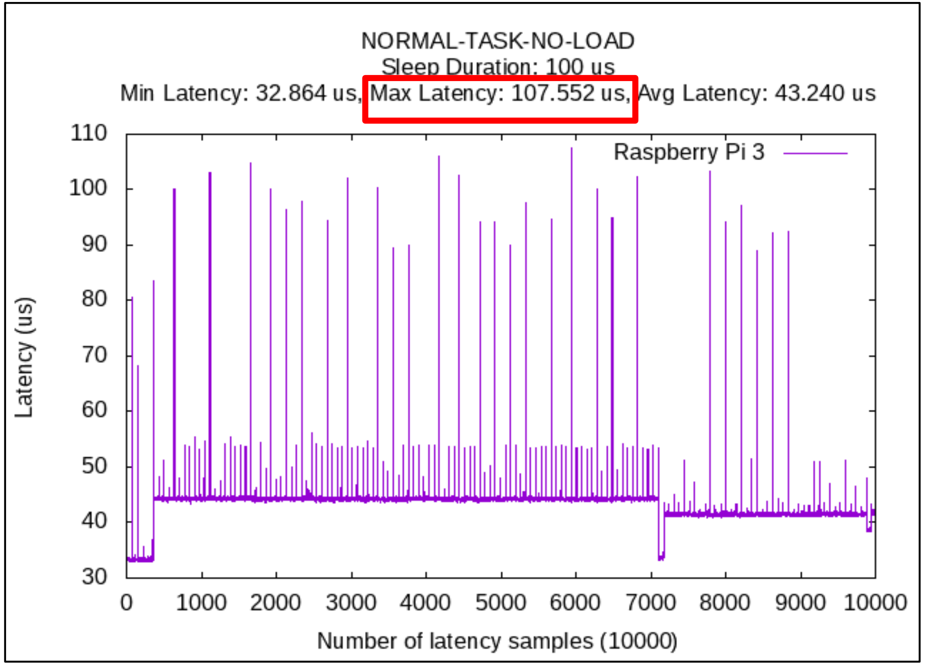


Note: This is just a hypothetical use case and not a real scenario.

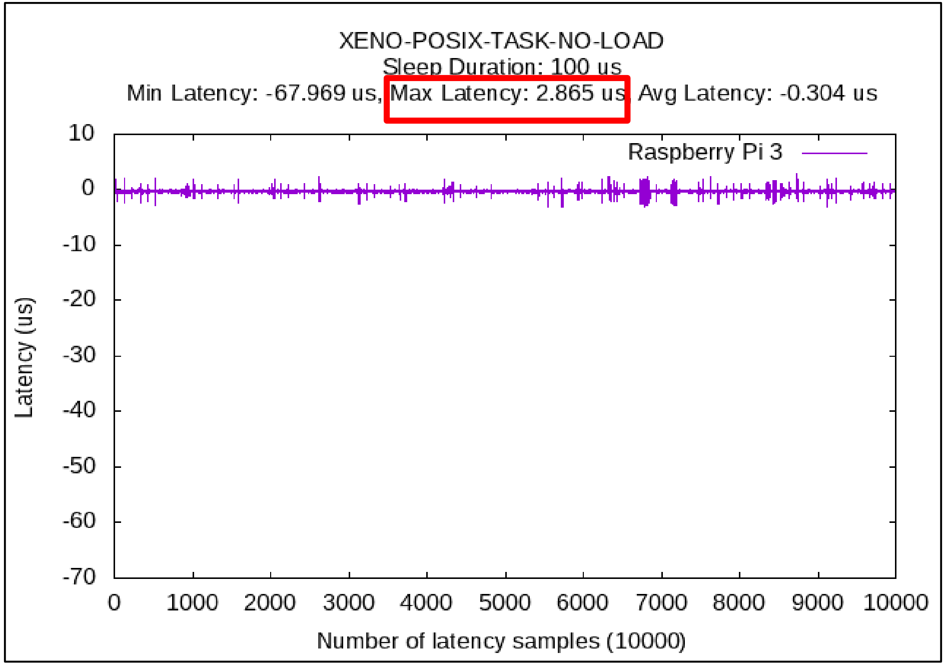
There could be many other use cases.

RESULT#1: 100us TASK (no-load)

NORMAL-LINUX

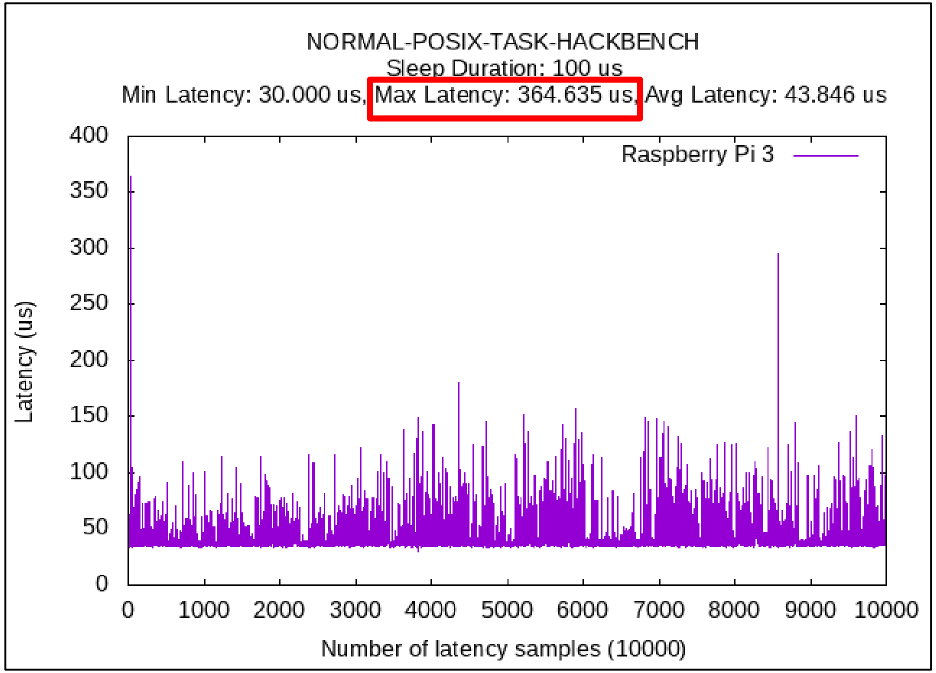


XENOMAI-POSIX

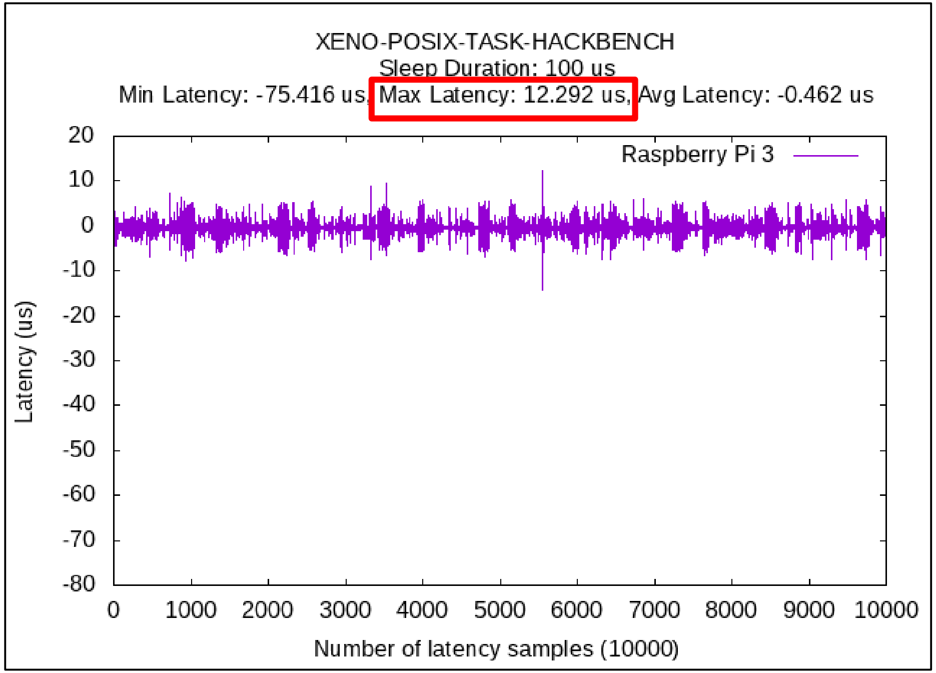


RESULT#2: 100us TASK (with-load)

NORMAL-LINUX-HACKBENCH



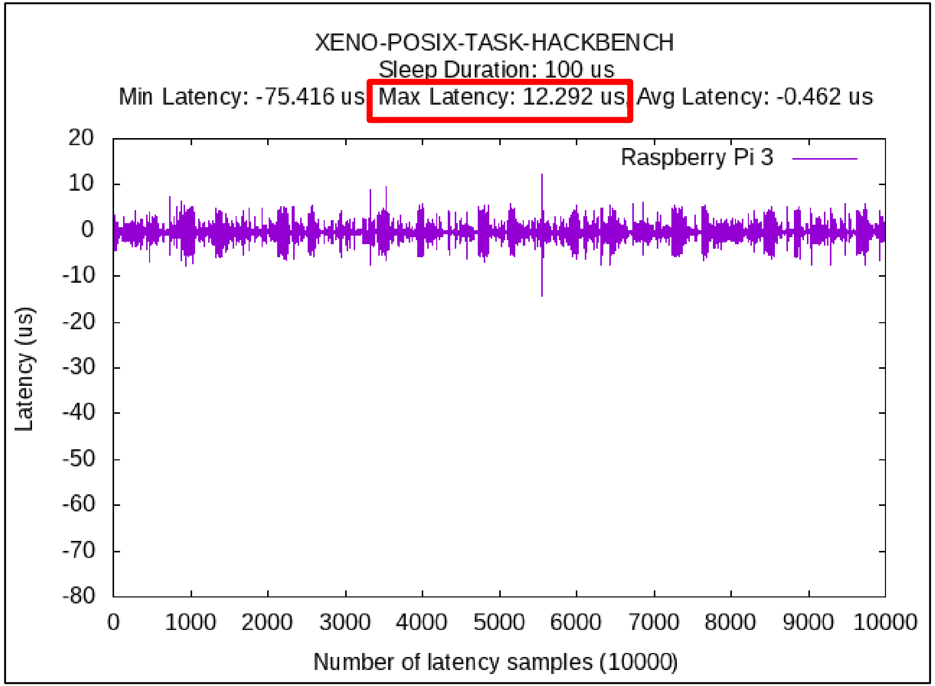
XENOMAI-POSIX-HACKBENCH



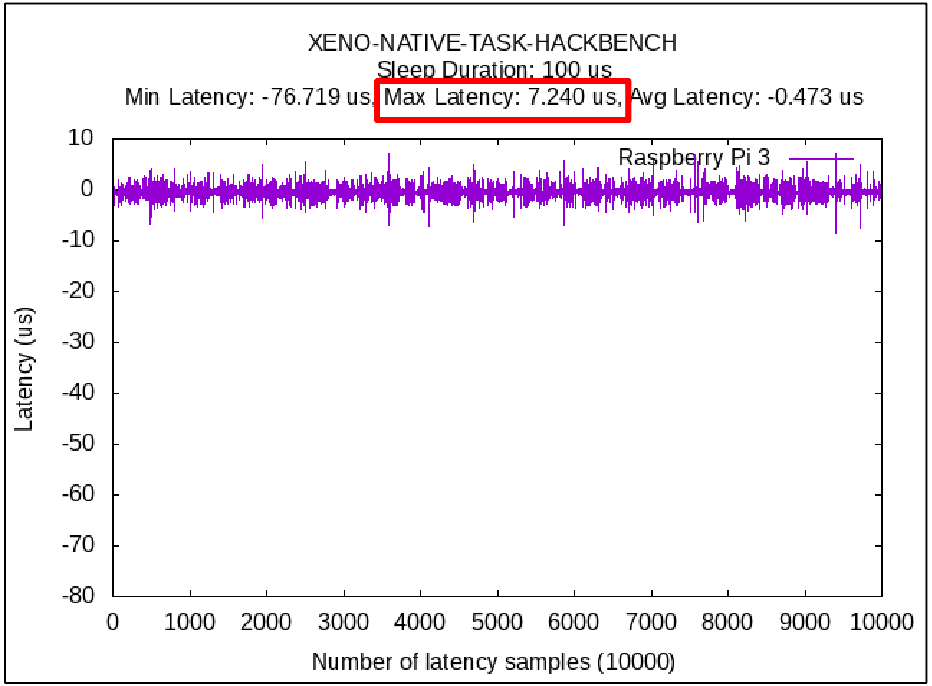
```
# hackbench --pipe 100 -s 100 --process 10000 -l 100000 &
```

RESULT#3: 100us TASK (using native API)

XENOMAI-POSIX-HACKBENCH

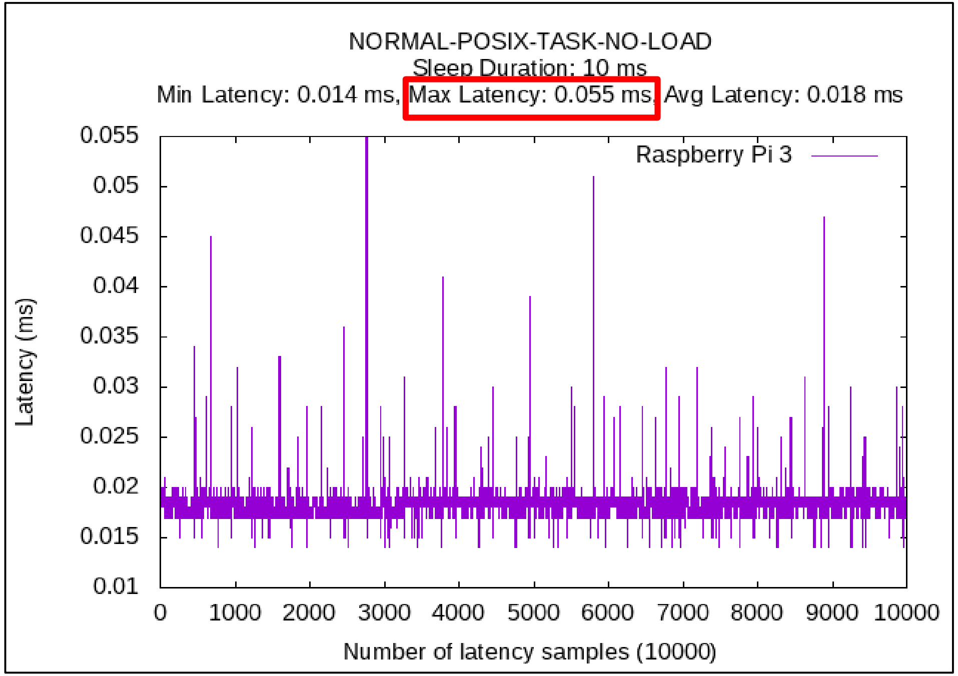


XENOMAI-NATIVE-HACKBENCH

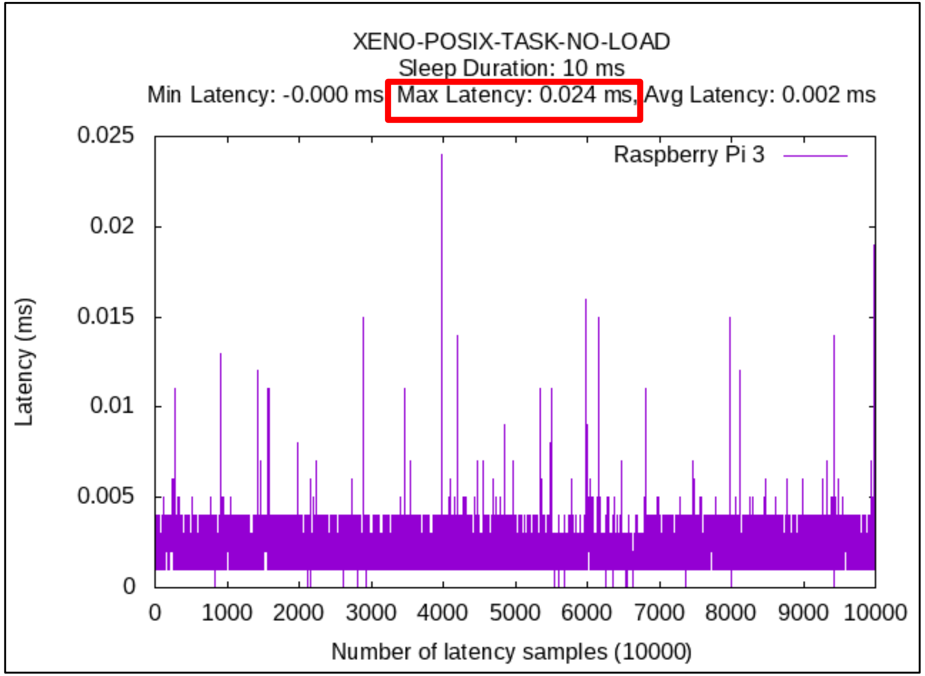


RESULT#4: 10ms TASK (no-load)

NORMAL-LINUX-NO-LOAD

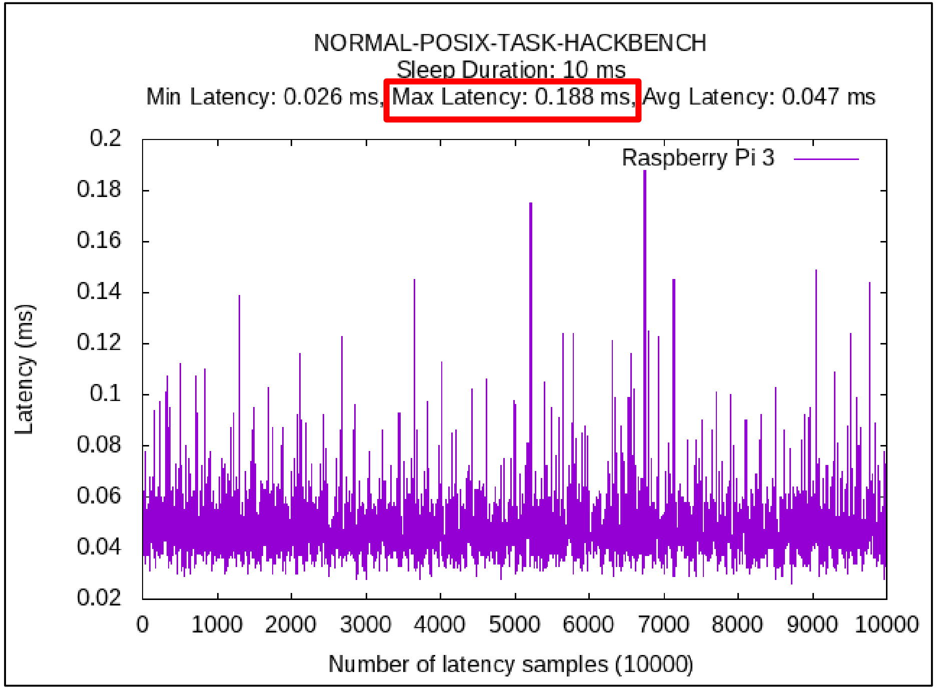


XENOMAI-POSIX-NO-LOAD

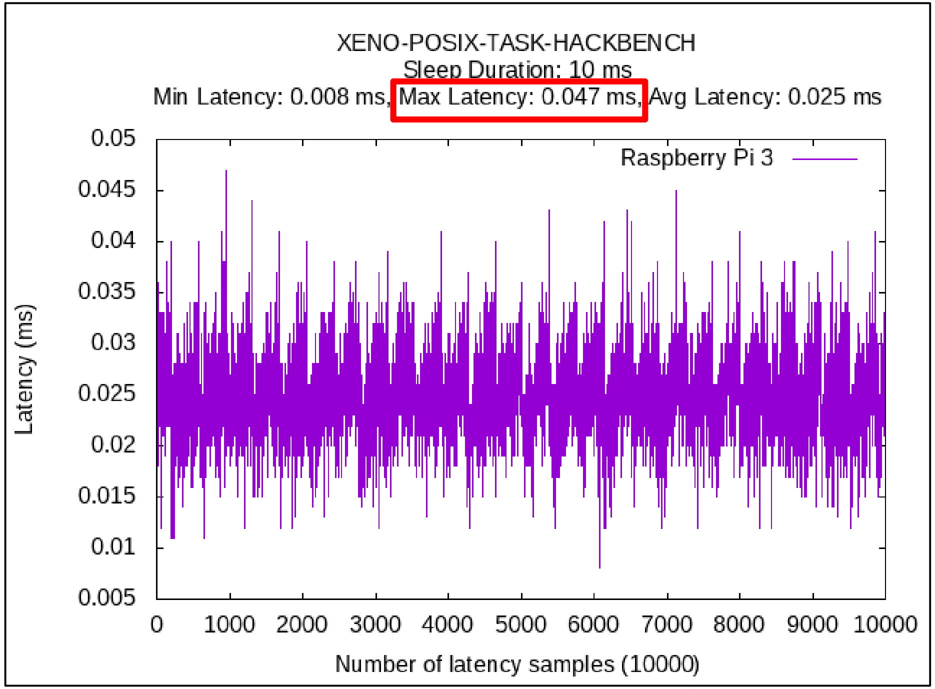


RESULT#5: 10ms TASK (with load)

NORMAL-LINUX-HACKBENCH



XENOMAI-POSIX-HACKBENCH



THE LINUX FOUNDATION

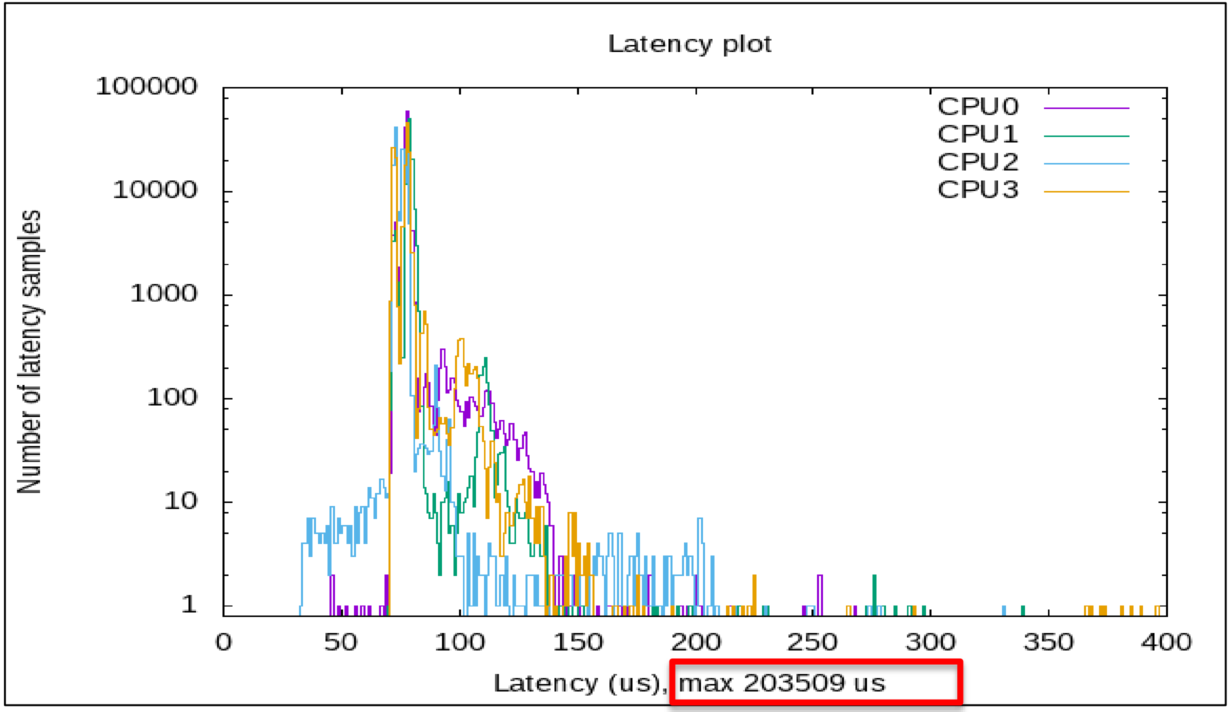
FURTHER KERNEL CONFIGS CHANGES

- **Disable CONFIG_NO_HZ (optional)**
- **Disable CONFIG_XENO_OPT_STATS**
- **Disable CONFIG_MIGRATION**
- **Disable CONFIG_PREEMPT_VOLUNTARY**
- **Enable CONFIG_PREEMPT**
- **Disable CONFIG_FTRACE**

Note: Some more Kernel configs can be cleaned up depending on your system

RESULT#6: CYCLICTEST OUTPUT (moving)

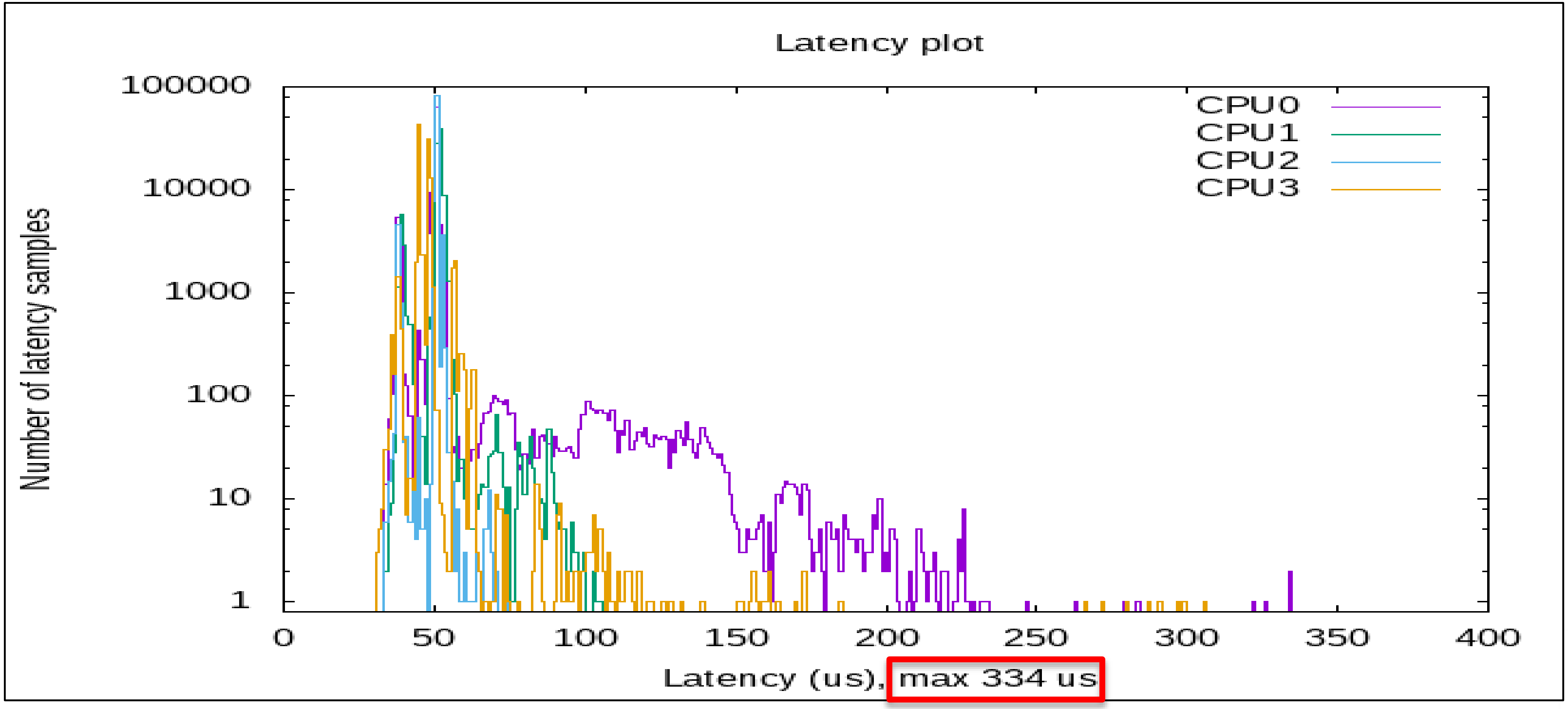
```
# sudo cyclictest -l1000000 --duration=5m -m -S p99 -i400 -h400 -q > output
```



# Min Latencies:
00046 00070 00033 00071
# Avg Latencies:
00078 00078 00106 00076
# Max Latencies:
03630 01013 **203509** 01130
# Histogram Overflows:
00020 00006 00558 00022

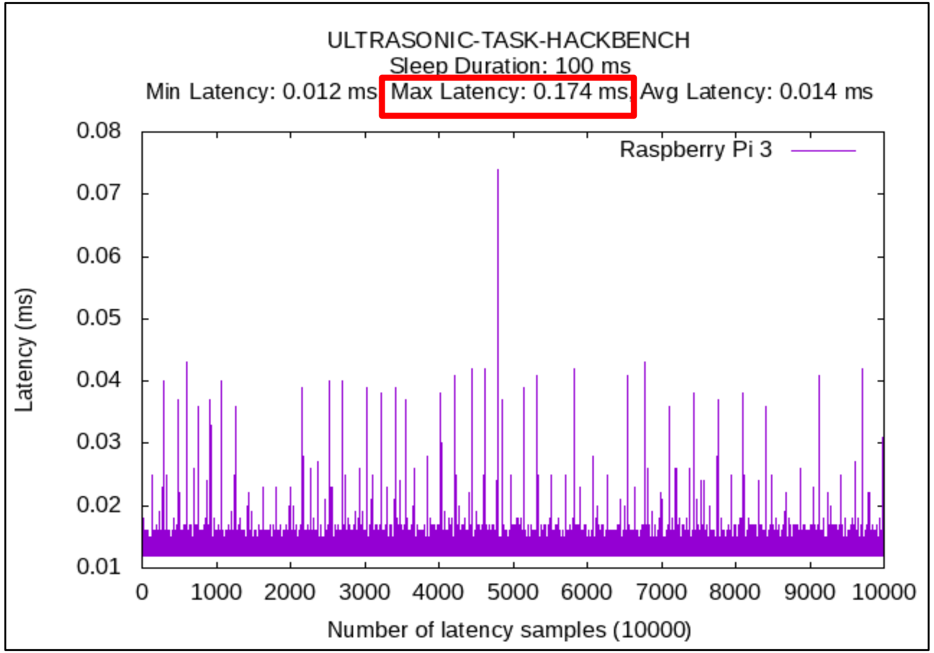
RESULT#7: CYCLICTEST OUTPUT (none)

```
# sudo cyclictest -l1000000 --duration=5m -m -S p99 -i400 -h400 -q > output
```

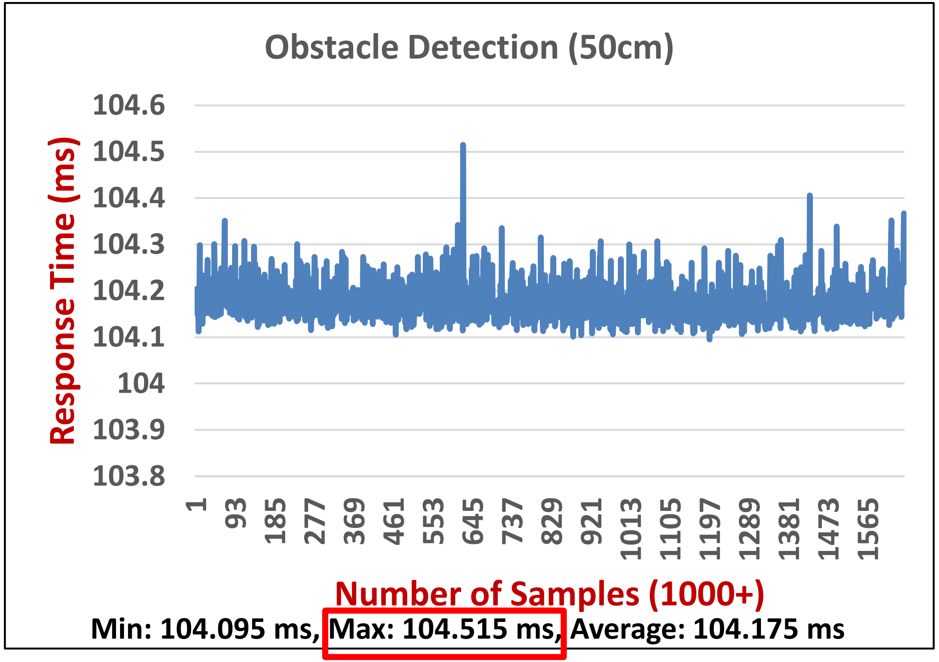


ACTUAL RESULT [Ultrasonic Task – 100ms]

XENOMAI-TASK-RESPONSE-HACKBENCH



RESPONSE-TO-ACTION-TAKEN-HACKBENCH



OBSERVATION & IMPROVEMENT AREAS

- **Xenomai maintains one bulky patches in separate repo. Once applied its difficult to upgrade.**
  - **xenomai-3/kernel/cobalt → linux/kernel/xenomai/**
  - **xenomai-3/kernel/drivers → linux/drivers/xenomai/**

Patches cannot be applied directly

- **Porting Xenomai on vendor Kernel is not easy without vendor and community support.**
- **Debugging real time issues could be challenge and time taking without expert knowledge.**
- **Unknowingly latencies could be negative or higher may be due to primary <-> secondary switching, system calls, etc.**
- **Some Kernel configs needs to be disabled to improve latency.**

CONCLUSION

- Although Xenomai shows some interesting results still there are scope for further improvements.
- Whether we use PREEMPT_RT or XENOMAI, its important to understand which portion of system needs real time capabilities.
- Measuring individual task latency and system tuning is important but could be tedious and painful.
- It is always better to start with bare minimal system and keep adding real time components which are optimized individually.
- Xenomai needs more people to improve its ecosystem. Interested people can join and contribute here:

  https://gitlab.denx.de/Xenomai/xenomai/wikis/How_To_Contribute

REFERENCES

- https://gitlab.denx.de/Xenomai/xenomai/wikis/home
- https://xenomai.org/documentation/xenomai-3/html/
- http://kth.diva-portal.org/smash/get/diva2:1251188/FULLTEXT01.pdf
- https://elinux.org/images/d/d7/Practical-Real-Time-Linux-ELCE15.pdf
- https://lemariva.com/blog/2018/07/raspberry-pi-xenomai-patching-tutorial-for-kernel-4-14-y
- http://wiki.csie.ncku.edu.tw/embedded/xenomai/rtlws_paper.pdf
- https://events.static.linuxfound.org/sites/events/files/slides/cyclictest.pdf
- https://play.google.com/store/apps/details?id=appinventor.ai_hobbyprojects_com.BluetoothRoboController&hl=en

THE LINUX FOUNDATION

OPEN SOURCE SUMMIT

JAPAN

THANK YOU!