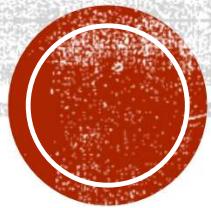


RAPID DEVELOPMENT OF EMBEDDED SYSTEMS FOR DEVELOPERS WITH OPEN-SOURCE TOOLS



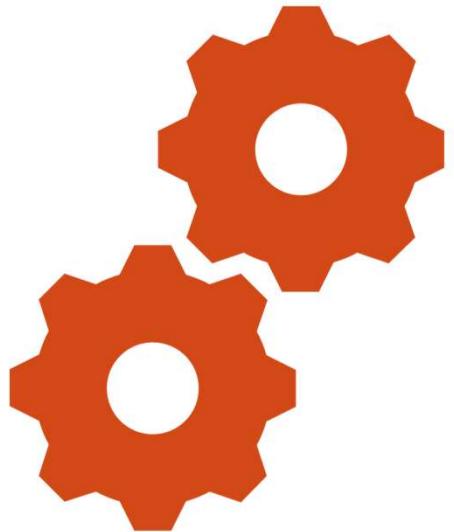
BHARATH G



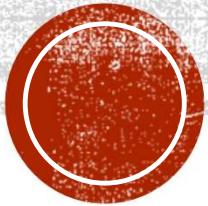
Senior Lead Engineer | Collins Aerospace

- 9+ Years of Experience in Firmware Development
- Medical, Aerospace and IoT device Design
- Embedded C++ | RTOS | Bluetooth Low Energy | Embedded Linux | Technical Training | Career Guidance





EMBEDDED SYSTEM DEVELOPMENT LIFE CYCLE



THE PRIMARY STAGES OF THE EMBEDDED PRODUCT DEVELOPMENT LIFE CYCLE

- The need phase
- Conceptualization phase
- Analysis phase
- Design phase
- Development and testing phase
- Deployment phase
- Support phase
- Upgrades phase
- Retirement/disposal



Source: [Building Embedded System Design & Software Development \(qt.io\)](https://www.qt.io/building-embedded-system-design-software-development)



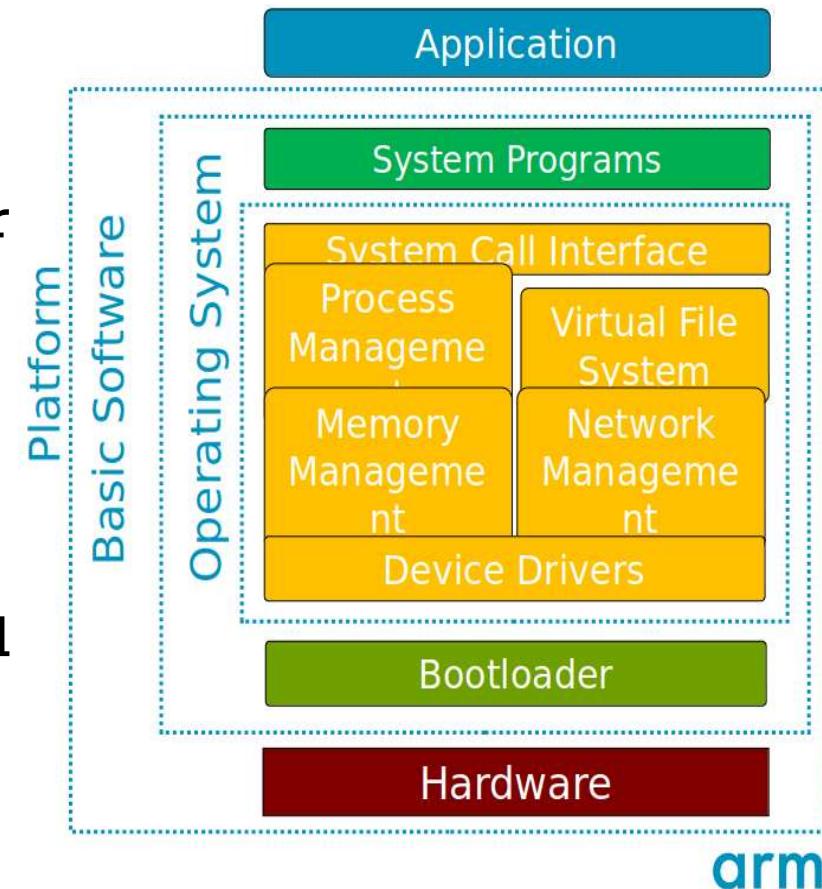
Embedded System Components

▪ Application

- Software that implements the functionalities for which the embedded system is intended (e.g., to control an Internal Combustion Engine)

▪ Platform

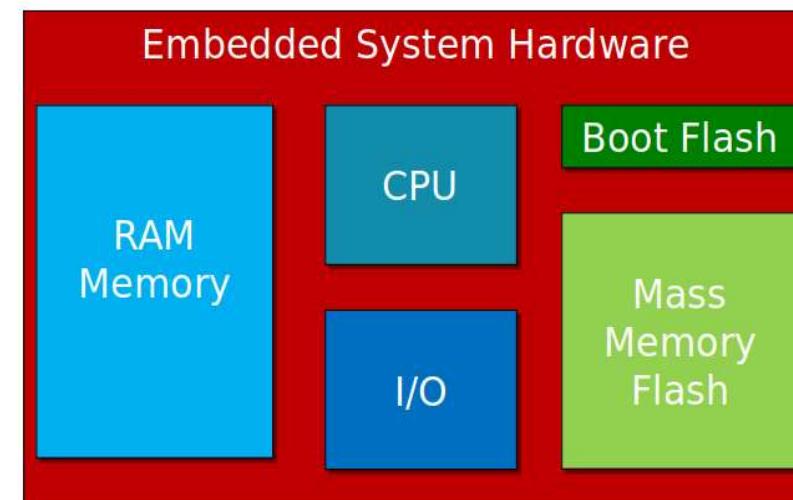
- Combination of hardware and basic software components that provides the services needed for the application to run
- Basic software may include system programs, operating system, bootloader



Source: [Embedded Linux – Arm®](#)

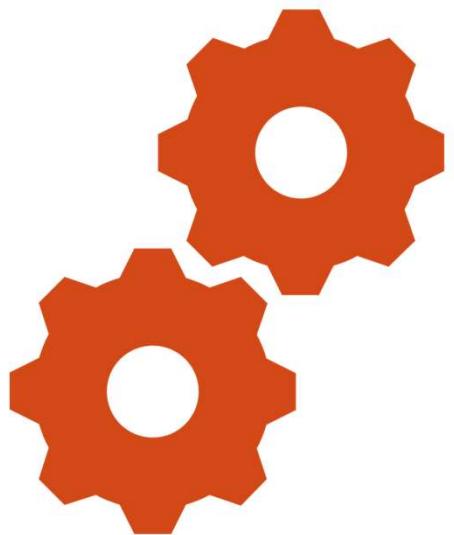
Hardware platform of Embedded Systems

- **RAM memory:** volatile memory storing data/code
- **CPU:** processor running software
- **I/O:** peripherals to get inputs from the user, and to provide outputs to the user
- **Boot Flash:** small non-volatile memory needed at power-up
- **Mass Memory Flash:** large non-volatile memory

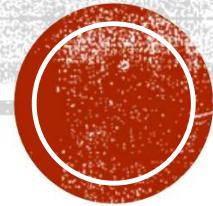


Source: [Embedded Linux – Arm®](#)





PROTOTYPE EMBEDDED SYSTEMS WITHOUT HARDWARE

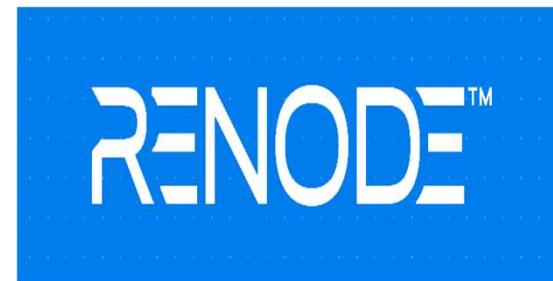


HOW TO PROTOTYPE EMBEDDED SYSTEMS WITHOUT HARDWARE

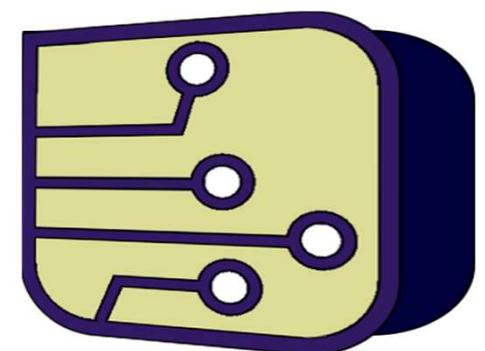
- Emulators
- Simulators



Linux Kernel,
Linux Application

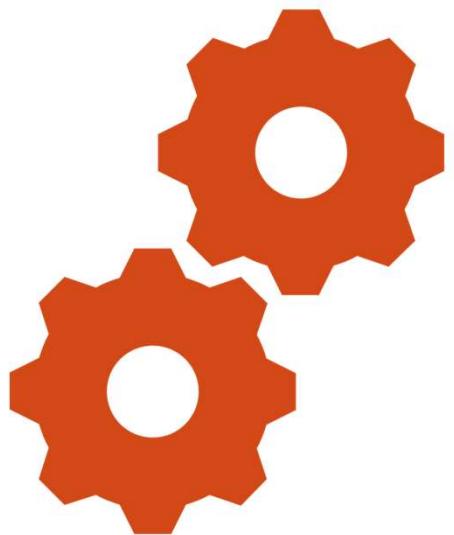


Boards & SoC's
Zephyr RTOS

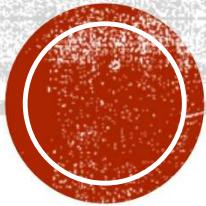


Electronic Circuits
AVR MCU's





EMBEDDED LINUX APPLICATION ON QEMU



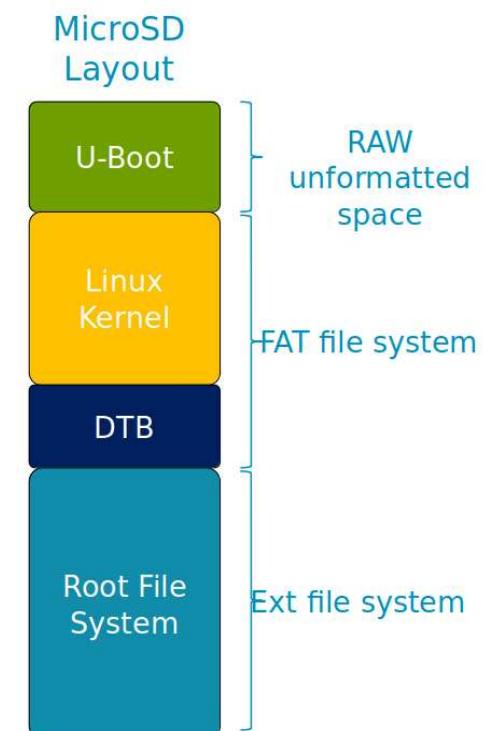
Components required to build Embedded Linux on target

- **An embedded Linux system requires the following components to operate:**

- The bootloader
- The Linux Kernel
- The device tree blob
- The Root File System

- **All these components shall be:**

- Configured for the embedded system hardware platform
- Compiled and linked into an executable format
- Deployed into the embedded system persistent storage for booting and operations

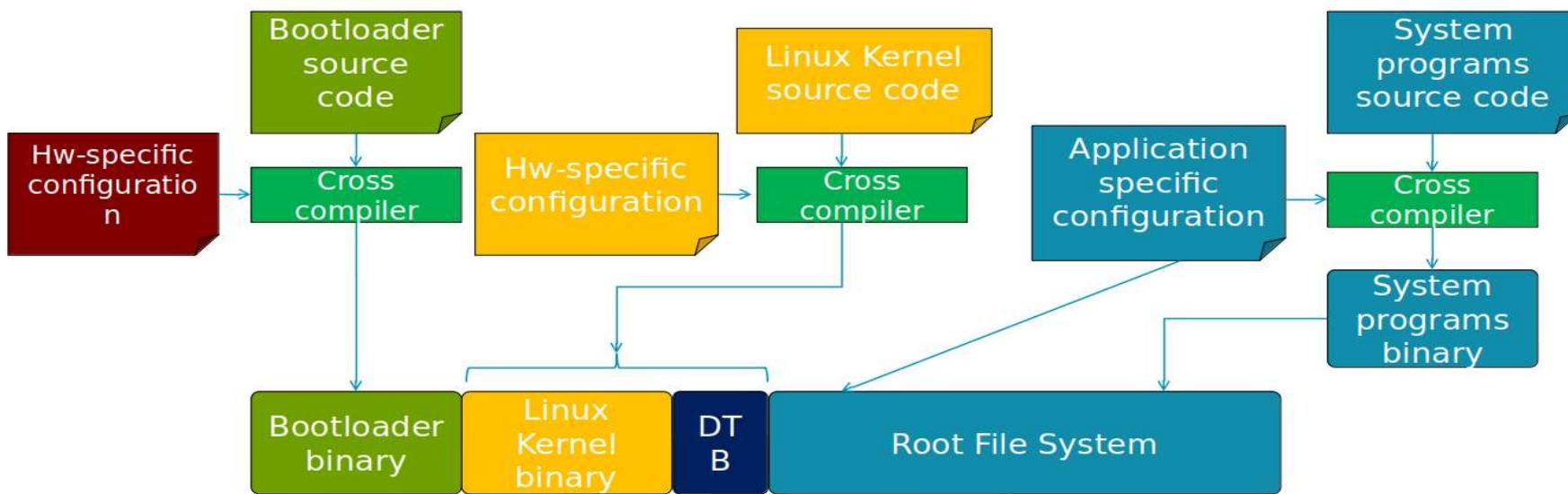


arm

Source: [Embedded Linux – Arm®](#)

Embedded Linux Workflow

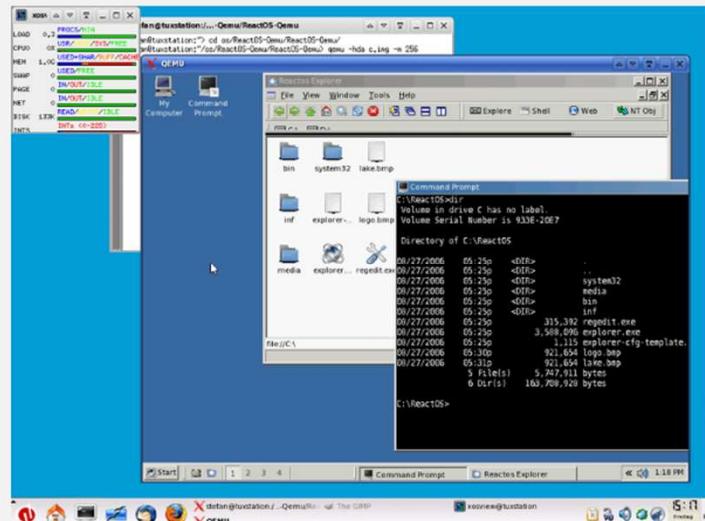
The Workflow



QEMU – Quick Emulator



Generic and open-source machine emulator and virtualizer



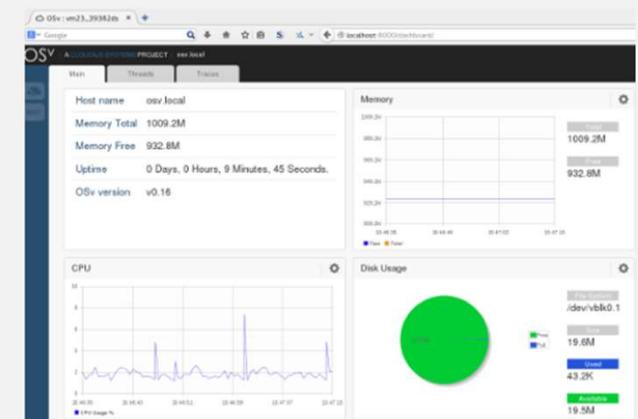
Full-system emulation

Run operating systems for any machine, on any supported architecture

```
[test@donizetti ~]$ qemu-arm ./ls --color /
bin etc lib64 mnt root srv system-upgrade-root var
boot home lost+found opt run sys tmp
dev lib media proc sbin system-upgrade usr
[test@donizetti ~]$ uname -a
Linux donizetti 4.6.7-300.fc24.x86_64 #1 SMP Wed Aug 17 18:48:43 UTC 2016 x86_64
x86_64 x86_64 GNU/Linux
[test@donizetti ~]$ file ./ls
./ls: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked,
 interpreter /lib/ld-linux-armhf.so.3, for GNU/Linux 3.0.0, stripped
[test@donizetti ~]$
```

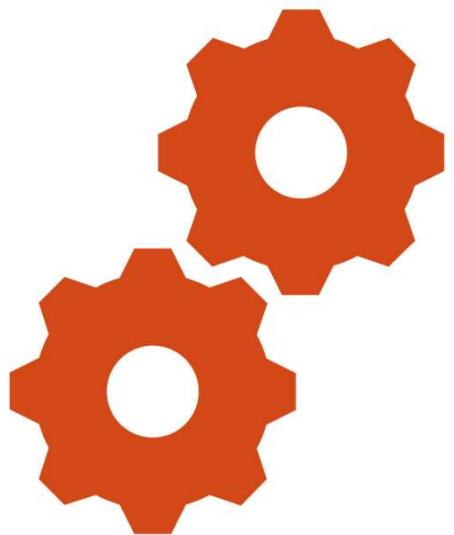
User-mode emulation

Run programs for another Linux/BSD target,
on any supported architecture



Virtualization

Run KVM and Xen virtual machines with near native performance

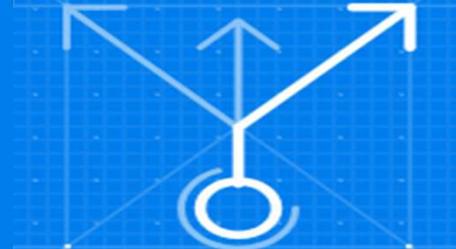


ZEPHYR RTOS APPLICATION ON RENODE



RENODE

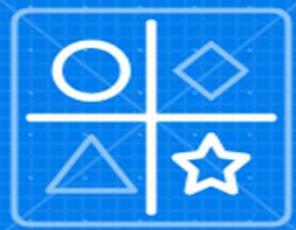
RENODE™



**Scalable simulation framework
with an IoT focus.**

**Vastly improves testing
capabilities.**

**Supports a modern, rapid
development workflow.**



**Lets you develop complex
multinode scenarios.**

**Easy to use, open and
extendible.**

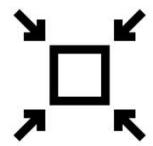
**Built for continuous
integration.**



**Enables security-hardened
devices and systems.**



ZEPHYR



SMALL

yet

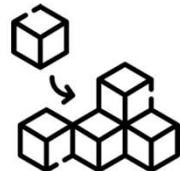


SCALABLE

< 8KB Flash

< 5KB RAM

from small sensor nodes
... to complex multi-core systems



FLEXIBLE

Heavily customizable

Out-of-the-box support for
450+ boards and 100s of sensors



SECURE

Built with safety & security in mind

Certification-ready

Long-term Support



Zephyr™



OPEN-SOURCE

Permissively licensed (Apache 2.0)

Vendor-neutral governance



ECOSYSTEM

Vibrant community

Supported by major silicon vendors

Supported Hardware Architectures



arm

Cortex-M, Cortex-R
& Cortex-A

intel.

x86 & x86_64

MIPS



Vibrant Ecosystem



Development Tools



Governing Board Technical Steering Committee



Contributors



Applications &
Middlewares



600+ supported boards... and growing



180+ Sensors Already Integrated



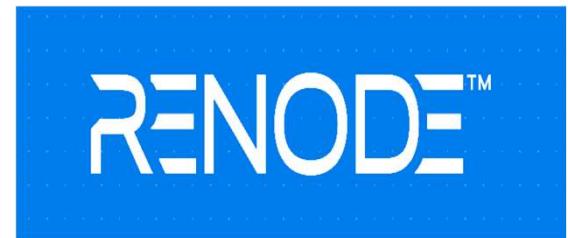
docs.zephyrproject.org/latest/hardware/index.html#hardware-support

github.com/zephyrproject-ztos/zephyr/tree/main/drivers/sensor

ZEPHYR ON RENODE

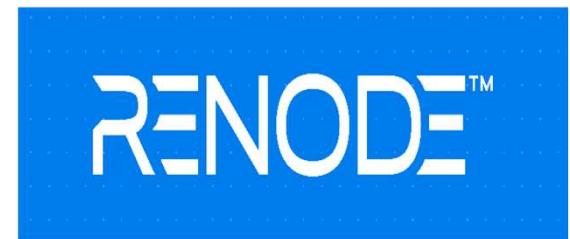
[Renode](#) and [The Zephyr Project](#)

- Zephyr on Renode – [Renodepedia](#)
- Supported boards- [Renodepedia – Boards](#)
- Example on STM32 - [Renodepedia - stm32f4 disco](#)
- Tutorials - [Renode Tutorials](#)



EXAMPLE ON RENODE

- Blinking LED on STM32F4 Disc - [Renodepedia - stm32f4_disco](#)
- BLE on nRF52840 - [Developing and testing Bluetooth Low Energy products on nRF52840 in Renode and Zephyr - Blogs - Nordic Blog - Nordic DevZone \(nordicsemi.com\)](#)
- Running your own Binaries - [Developing and testing BLE products on nRF52840 in Renode and Zephyr](#)



EXAMPLE ON RENODE

Renode



Renode, version 1.12.0.4346 (cf6a621e-202202280224)

(monitor) include @scripts/multi-node/nrf52840-ble-zephyr.resc
Script loaded. Now start with the 'start' command.

(peripheral) start
Starting emulation...
(peripheral)

Renode

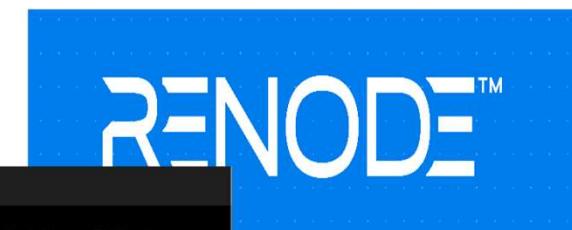
```
13:45:15.7935 [WARNING] central/radio: Unhandled write to offset 0x518. Unhandled bits: [25] when writing value 0x2000000. Tags: WHITEEN (0x1).
13:45:15.7935 [WARNING] central/radio: Unhandled write to offset 0x650. Unhandled bits: [0] when writing value 0x201. Tags: RU (0x1).
13:45:15.7936 [WARNING] central/radio: Unhandled write to offset 0x514. Unhandled bits: [24] when writing value 0x100008. Tags: PLEN (0x1).
13:45:15.7936 [WARNING] central/timer0: Unhandled write to offset 0x508, value 0x2.
13:45:15.7937 [WARNING] central/timer1: Unhandled write to offset 0x508, value 0x0.
13:45:15.8007 [WARNING] peripheral/radio: Unhandled read from offset 0x774.
13:45:15.8007 [WARNING] peripheral/radio: Unhandled write to offset 0x774, value 0x1000000.
13:45:15.8007 [WARNING] peripheral/radio: Unhandled write to offset 0x650. Unhandled bits: [0] when writing value 0x201. Tags: RU (0x1).
13:45:15.8007 [WARNING] peripheral/radio: Unhandled write to offset 0x514. Unhandled bits: [24] when writing value 0x100008. Tags: PLEN (0x1).
13:45:15.8008 [WARNING] peripheral/radio: Unhandled write to offset 0x554, value 0x21.
13:45:15.8008 [WARNING] peripheral/radio: Unhandled write to offset 0x518. Unhandled bits: [25] when writing value 0x20300018. Tags: WHITEEN (0x1).
13:45:15.8009 [WARNING] peripheral/timer0: Unhandled write to offset 0x508, value 0x2.
13:45:15.8093 [WARNING] peripheral/timer1: Unhandled write to offset 0x508, value 0x0.
13:45:15.8093 [WARNING] central/radio: Unhandled write to offset 0x650. Unhandled bits: [0] when writing value 0x201. Tags: RU (0x1).
13:45:15.8093 [WARNING] central/radio: Unhandled write to offset 0x514. Unhandled bits: [24] when writing value 0x100008. Tags: PLEN (0x1).
13:45:15.8097 [WARNING] peripheral/radio: Unhandled write to offset 0x650. Unhandled bits:
```

central:sysbus:uart0

```
*** Booting Zephyr OS build zephyr-v2.6.0-2019-g30b4c29aa5d
Bluetooth initialized
Scanning successfully started
[DEVICE]: C0:00:AA:BB:CC:DD (random), AD evt type 0, AD data len 11, RSSI -10
[AD]: 1 data_len 1
[AD]: 3 data_len 6
Connected: C0:00:AA:BB:CC:DD (random)
[00:00:00.488] <inf> bt_hci_core: HW Platform: Nordic Semiconductor (0x0002)
[00:00:00.488] <inf> bt_hci_core: HW Variant: nRF52x (0x0002)
[00:00:00.488] <inf> bt_hci_core: Firmware: Standard Bluetooth controller (0x00)
Version 2.6 Build 99
[00:00:00.671] <inf> bt_hci_core: Identity: C0:00:AA:BB:CC:DD (random)
[00:00:00.671] <inf> bt_hci_core: HCI: version 5.2 (0x0b) revision 0x0000,
manufacturer 0x05f1
[00:00:00.671] <inf> bt_hci_core: LMP: version 5.2 (0x0b) subver 0xffff
[ATTRIBUTE] handle 25
[ATTRIBUTE] handle 26
[ATTRIBUTE] handle 28
[SUBSCRIBED]
[NOTIFICATION] data 0x20005a51 length 2
```

peripheral:sysbus.uart0

```
*** Booting Zephyr OS build zephyr-v2.6.0-2019-g30b4c29aa5d
Bluetooth initialized
Advertising successfully started
Connected
[00:00:00.000.488] <inf> bt_hci_core: HW Platform: Nordic Semiconductor (0x0002)
[00:00:00.488] <inf> bt_hci_core: HW Variant: nRF52x (0x0002)
[00:00:00.488] <inf> bt_hci_core: Firmware: Standard Bluetooth controller (0x00)
Version 2.6 Build 99
[00:00:00.671] <inf> bt_hci_core: Identity: C0:00:AA:BB:CC:DD (random)
[00:00:00.671] <inf> bt_hci_core: HCI: version 5.2 (0x0b) revision 0x0000,
manufacturer 0x05f1
[00:00:00.671] <inf> bt_hci_core: LMP: version 5.2 (0x0b) subver 0xffff
[00:00:00.462_006] <inf> hrs: HRS notifications enabled
```

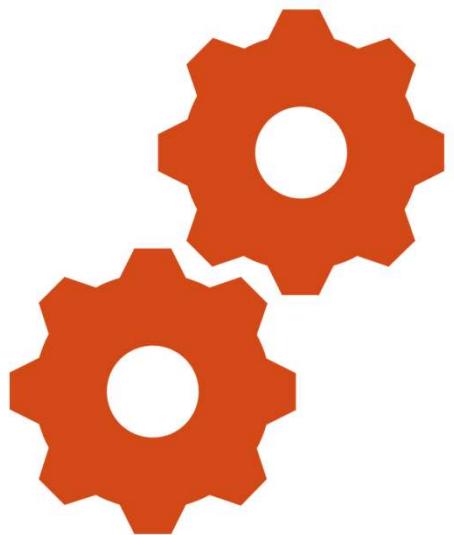


Zephyr™

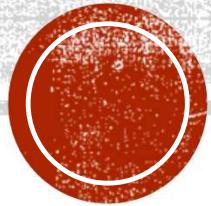
BLE on nRF52840 - [Developing and testing Bluetooth Low Energy products on nRF52840 in Renode and Zephyr](#)

Running your own Binaries - [Developing and testing BLE products on nRF52840 in Renode and Zephyr](#)





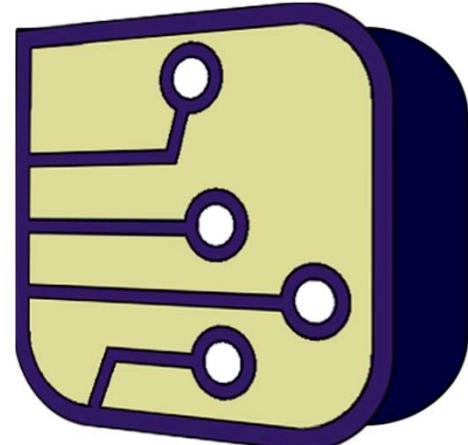
CIRCUITS WITH AVR ON SIMULIDE



CIRCUITS WITH AVR ON SIMULIDE

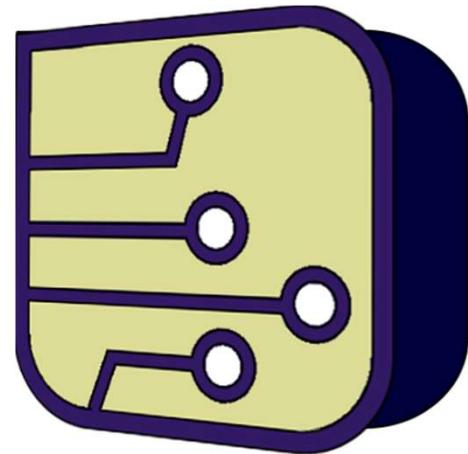
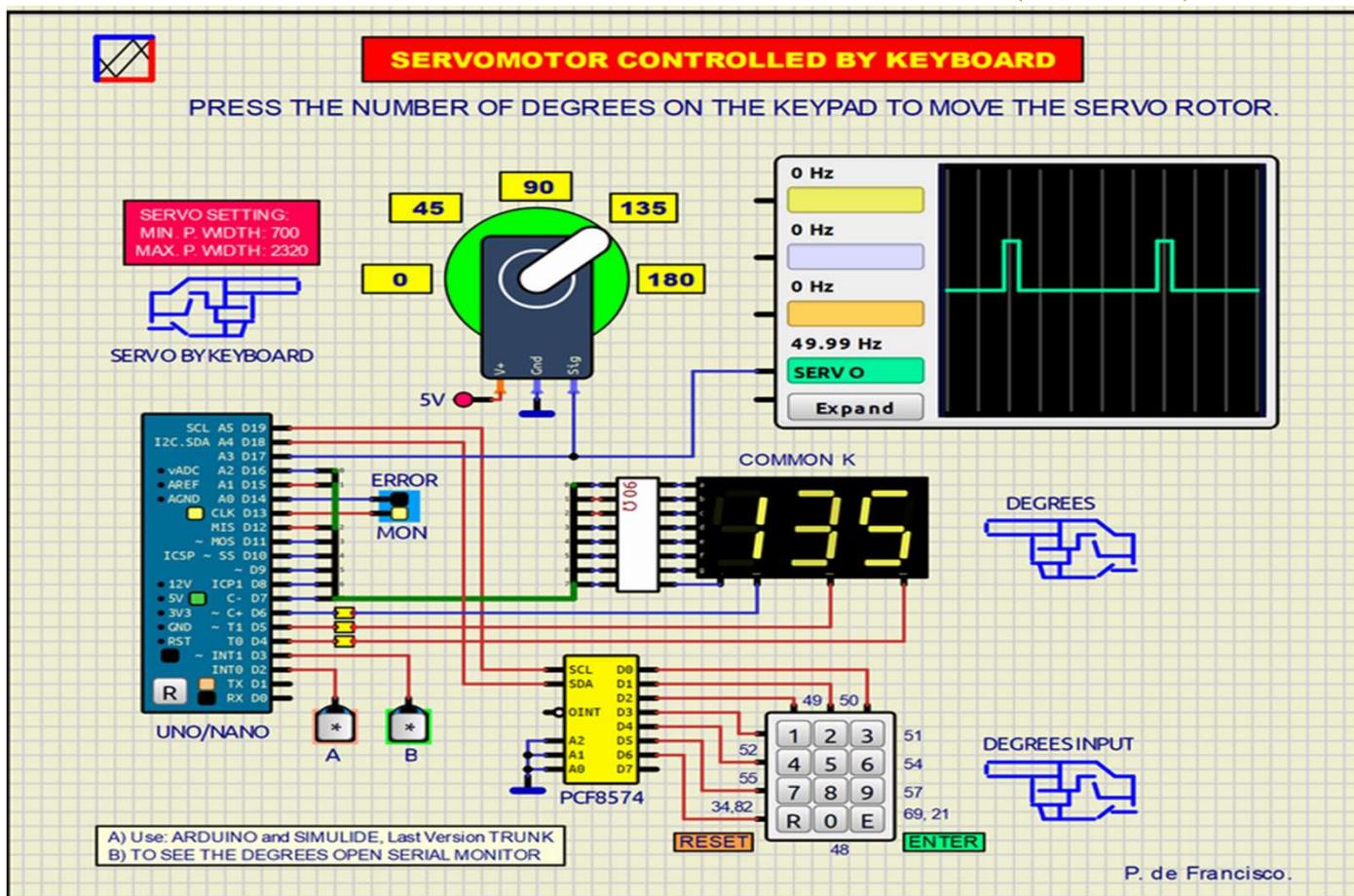
SimulIDE – Circuit Simulator

- Real time electronic circuit simulator
- Analog and digital electronic circuits and microcontrollers
- Supports PIC, AVR , Arduino and other MCUs and MPUs.
- Examples - [Projects – SimulIDE](#)
- Tutorials - <https://simulide.blogspot.com/p/tutorials-0415.html>
- Projects - [Projects – SimulIDE](#)



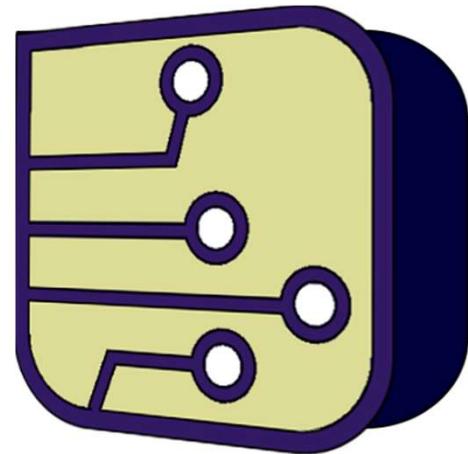
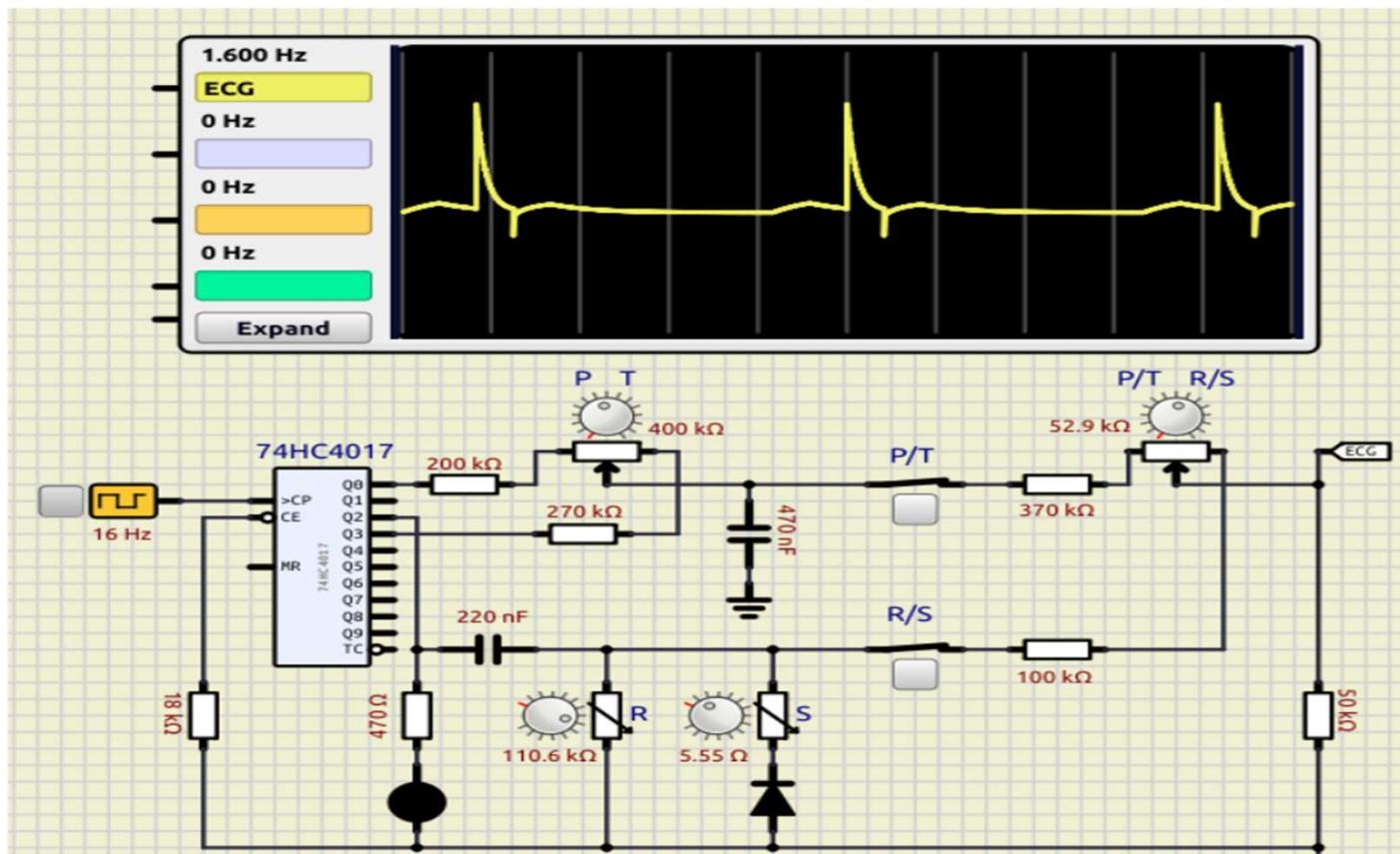
Controlling Servo With Keyboard

• SERVO CONTROLLED BY KEYBOARD (DF330) – SimulIDE



ECG Signal Generator Circuit

- ECG signal generator. – SimulIDE



NEXT STEPS

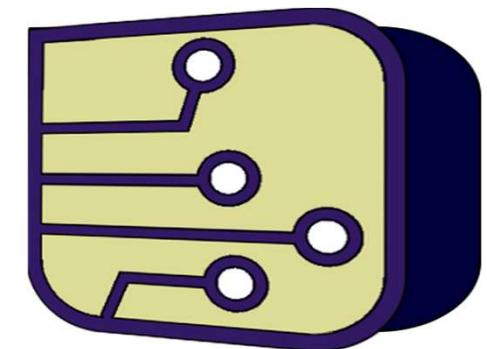
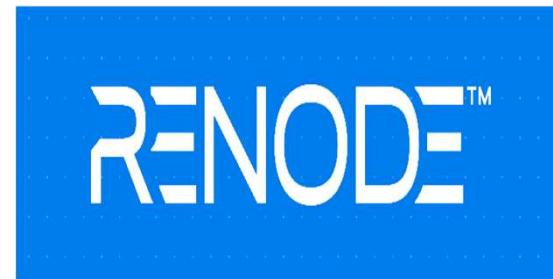
Resources to learn about QEMU, RENODE and SIMULIDE

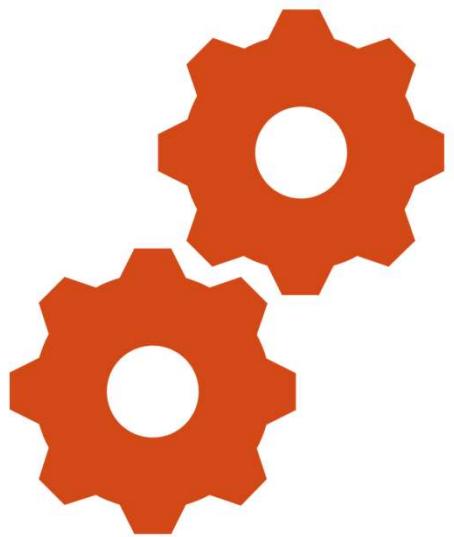
[Welcome to QEMU's documentation! — QEMU documentation](#)

[Compiling linux kernel for qemu arm emulator GitHub](#)

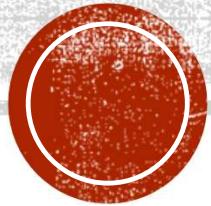
[Renode Tutorials](#)

<https://simulide.blogspot.com/p/tutorials-0415.html>



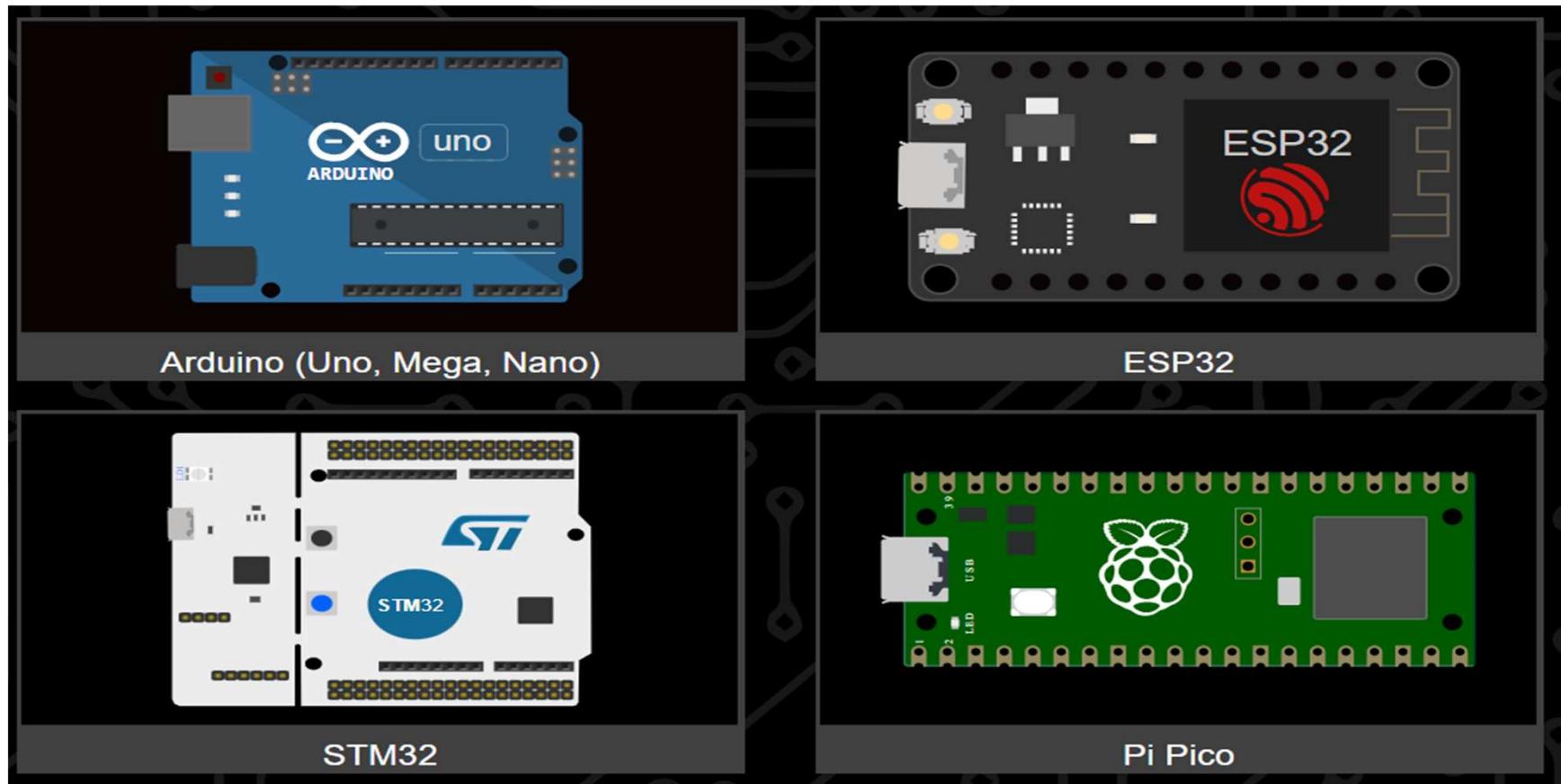


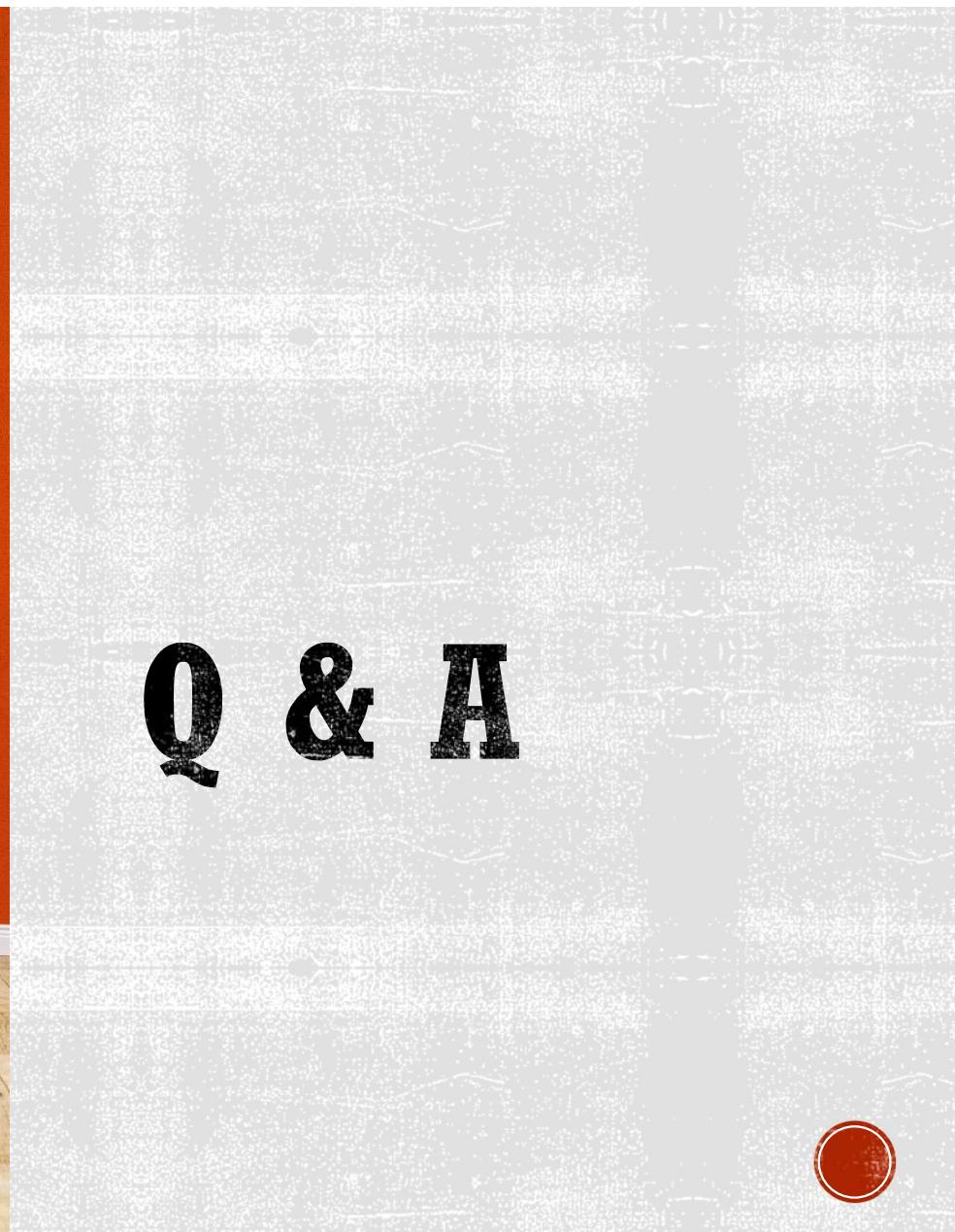
BONUS – WOKWI



Wokwi – Online Simulator + VSCode Offline

- Wokwi - Online ESP32, STM32, Arduino Simulator





STAY CONNECTED



- Bharathgopal175@gmail.com
- [Linkedin.com/in/Bharathgopal](https://www.linkedin.com/in/Bharathgopal)
- LinkedIn- QR Code

