Embedded Linux Intro

Norman McEntire

Introduction

- Embedded Linux Systems Are All Around You!
  - Industrial Devices, Medical Devices, Consumer Devices, Networking Equipment, Transportation Systems, Power Grid Equipment, etc. etc. etc.
- Example from one of my clients
  - Itron Electric Power Meter

Two Large Categories of Embedded Systems

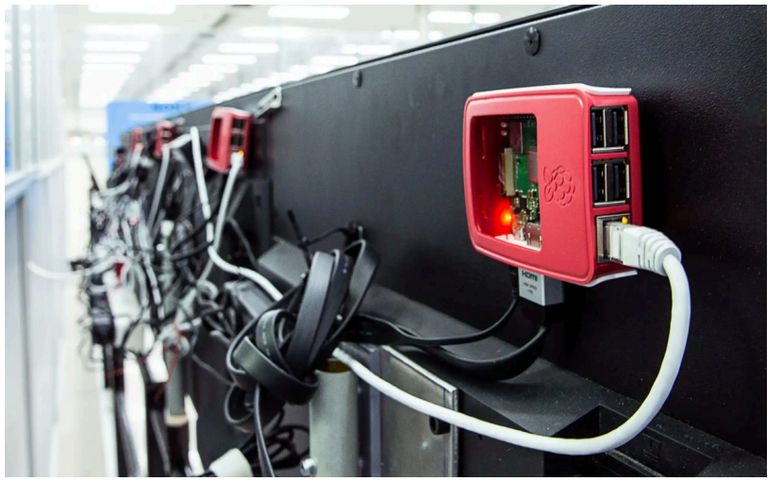
- Embedded Systems can be grouped into two large categories
    - Embedded Systems Based on **Microcontrollers**
        - Example Microcontroller: ARM Cortex M4
            - No Memory Management (physical memory addresses)
            - Example OS: FreeRTOS
            - Example Platform: **Arduino**
    - Embedded Systems Based on **Microprocessors**
        - Example Microprocessor: **ARM A7**
            - **Memory Management** (virtual to physical translation)
        - Example OS: **Embedded GNU/Linux**
        - Example Platform: **Raspberry Pi**

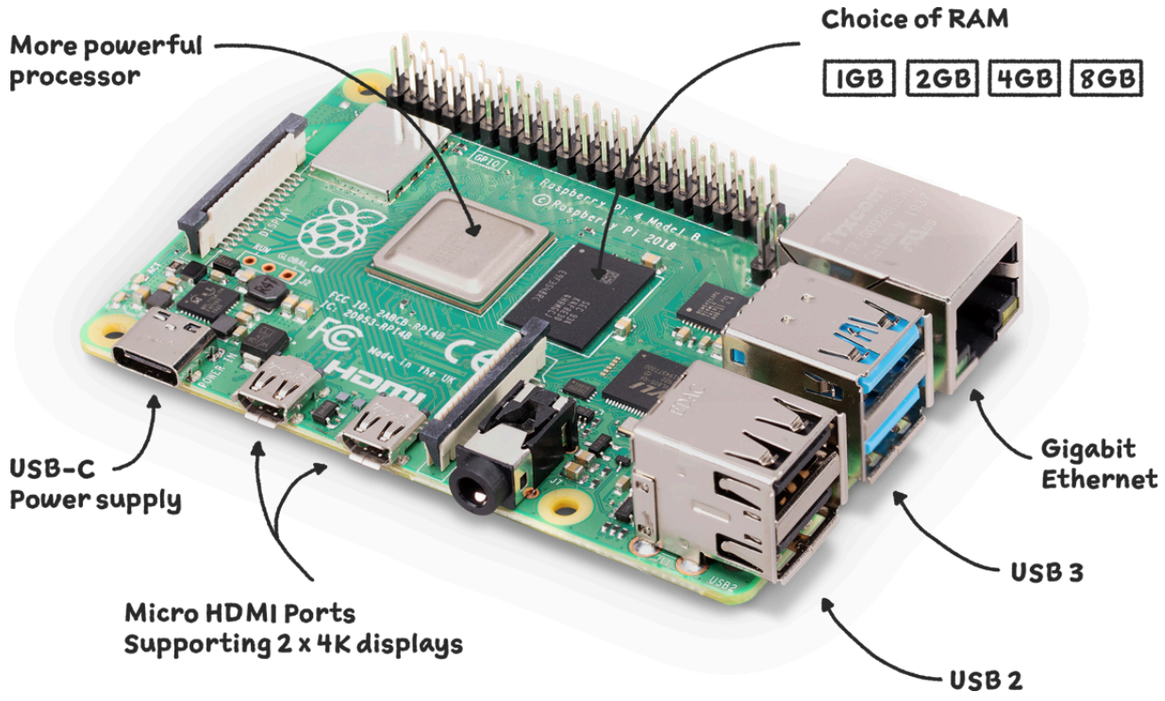
Three Categories of COPS Embedded Linux Systems

- COPS: Common Off-The-Shelf
- Category 1: **MPU/Embedded Linux**
    - MPU: Microprocessor Unit (single or multicore)
    - 100% Focused on Embedded Linux
    - Example Platform: Raspberry Pi
- Category 2: **MPU/Embedded Linux + MCU/FreeRTOS**
    - MCU: Microcontroller Unit
    - Embedded Linux on MPU + FreeRTOS on MCU
    - Example Platform: STM
- Category 3: **MPU/Embedded Linux + FPGA**
    - Embedded Linux + FPGA (Field Programmable Gate Array)
    - Example Platform: Red Pitaya

Raspberry Pi

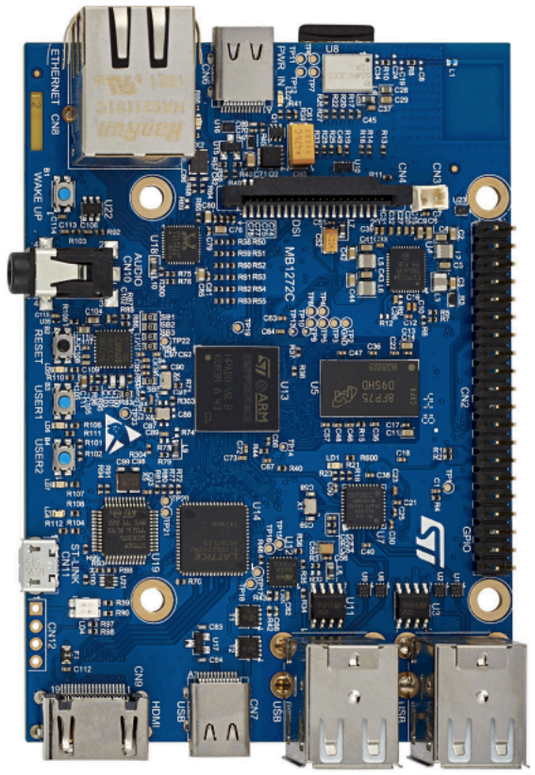
https://www.raspberrypi.com/





STM32MP1

https://www.st.com/resource/en/data_brief/stm32mp157f-dk2.pdf



STM32MP157F-DK2 top view with display removed. Picture is not contractual.

F

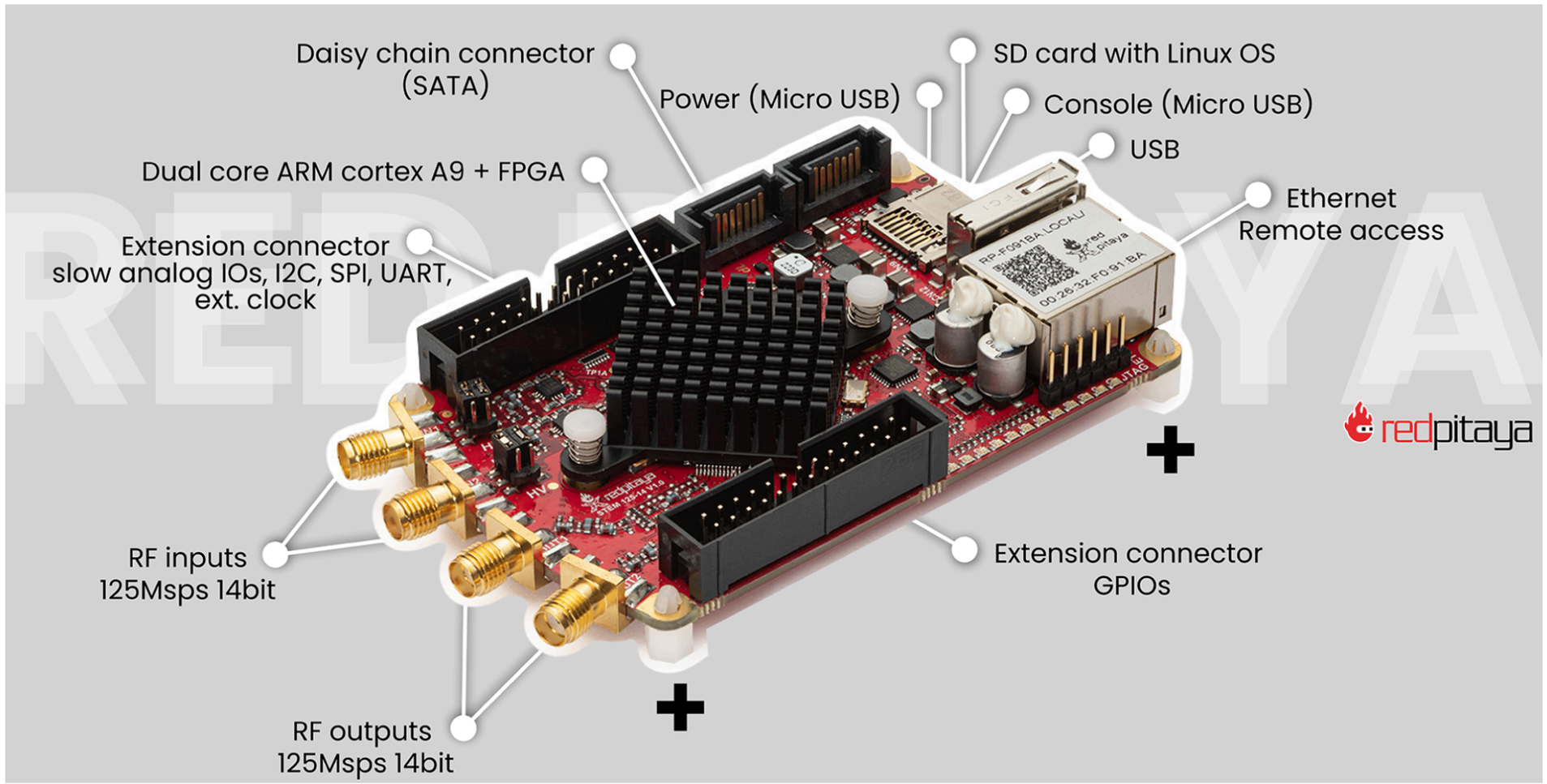
- The STM32MP157D-DK1 and STM32MP157F-DK2 Discovery kits leverage the capabilities of the increased-frequency 800 MHz microprocessors in the STM32MP157 product line to allow users to develop applications easily using STM32 MPU OpenSTLinux Distribution software for the main processor and STM32CubeMP1 software for the coprocessor.

STM32 MPU OpenSTLinux
(for MPU)

STM32CubeMP1
(for MCU)

Red Pitaya

https://redpitaya.com/stemlab-125-14/



What's Included in Embedded Linux?

- **Toolchain:** To build all the code shown below!
- **Libraries**: glib, uClibc, etc.
- **Bootloader**: Runs first when power applied
- Linux **Kernel:** Loaded by boot loader
- **Root File System, RFS**: mounted at end of kernel boot)
- **/sbin/init:** First process to run (PID 1)
- **Kernel Modules:** Dynamically loaded as needed and become part of kernel
- **Daemons**: Processes running in the background
- **CLI**: Command-Line Interface: Often Busybox
- Embedded Applications: As needed for specific embedded target

Embedded Linux “Software Stack”: Build vs Buy?

- From the previous slide, lots of software involved in Embedded Linux
  - Toolchain, Libraries, Bootloader, Kernel, Kernel Modules, Daemons, CLI, Applications, etc.
- Where to get it
  - Option 1: Comes with the board (e.g. RPi, etc.)
  - Option 2: Build your own (e.g. BuildRoot, etc)
  - Option 3: Hire a 3rd Party (e.g. Timesys, etc.)

Host and Target

- Embedded Linux Systems often developed as Host/Target
  - Host is the development system
    - Example: Host could be x86_64 PC
  - Target is the embedded system
    - Example: ARM A7 embedded system

Cross Tool Chain

- A Toolchain is used to develop on a Host and target the output to a Target
- Often requires a Cross-Tool chain
  - The toolchain runs on the HOST (e.g. x86_64 PC)
    - But produces output for the TARGET (e.g. armv7l)
- HOWEVER: Embedded Systems such as RPi may be both the Host and the Target - same toolchain for both

Fast Path To Mastering Embedded Linux

- We start with Raspberry Pi and explore what is already available
  - Toolchain, Libraries, Bootloader, Kernel, Kernel Modules, /sbin/init, Daemons, CLI, Applications
- Later we can look at ways to build/customize for your exact needs
  - Example: Buildroot and other build systems