December 2018

EMBEDDED SYSTEMS ENGINEERING powered by EECatalog

Guiding Embedded Designers on Systems and Technologies

Engineers' Guide to Android & Embedded Linux

Selecting an OS for Embedded Applications

www.eecatalog.com/embeddedlinux

Gold Sponsor

CONTENTS

EMBEDDED SYSTEMS ENGINEERING

ENGINEERS' GUIDE TO ANDROID & EMBEDDED LINUX 2018
www.eecatalog.com/embeddedlinux

Vice President & Publisher
Clair Bright

Editorial
Editor-in-Chief
Lynnette Reese | lreese@extensionmedia.com
Managing Editor
Anne Fisher | afisher@extensionmedia.com
Senior Editors
Caroline Hayes | chayes@extensionmedia.com
Dave Bursky | dbursky@extensionmedia.com
Pete Singer | psinger@extensionmedia.com
John Blyler | jblyler@extensionmedia.com

Creative / Production
Production Traffic Coordinator
Darla Rovetti
Production Manager
Nicky Jacobson
Graphic Designer
Simone Bradley
Senior Web Developer
Slava Dotsenko

Advertising / Reprint Sales
Vice President, Sales
Embedded Electronics Media Group
Clair Bright
cbright@extensionmedia.com
(415) 255-0390 ext. 15

Marketing/Circulation
Jenna Johnson
jjohnson@extensionmedia.com

To Subscribe
www.eecatalog.com

Extension MEDIA

Extension Media, LLC Corporate Office
President and Publisher
Vince Ridley
vridley@extensionmedia.com
(415) 255-0390 ext. 18
Vice President & Publisher
Clair Bright
cbright@extensionmedia.com
Human Resources / Administration
Darla Rovetti

Special Thanks to Our Sponsor

INTERNET OF THINGS DEVELOPERS CONFERENCE

WWW.IOT-DEVCON.COM

JUNE 5-6, 2019

SANTA CLARA CONVENTION CENTER

The Premier Conference Devoted To Implementing IoT Technology in Embedded Systems - PLAN NOW TO ATTEND!

Selecting an Operating System for Embedded Applications

Here are the questions to ask when adopting a rational approach that brings confidence to the selection process.

By Colin Walls, Mentor, a Siemens Business

With all but the most minimal of embedded systems, some kind of kernel or embedded operating system (OS) is likely to be required. And this requirement means that a choice has to be made.

There is a huge choice. At the highest level, the choice is between "heavyweight" OSes—those based on desktop OSes, like Linux—and real-time operating systems (RTOS), of which there are around 200 on the market. You have the choice of commercial products and various open-source options. In-house OS development is also an option, but this is very rarely a viable approach.

There have been attempts to produce a flowchart or a decision tree to help with OS selection, but this approach is too simplistic. There are too many parameters: technical factors, commercial concerns, past experience, word of mouth ... This article aims to to rationalize that decision-making process.

DO YOU NEED AN OS AT ALL?

It is rare nowadays to find an embedded system without an operating system (OS). Only the simplest kind of device can be built efficiently without a kernel of some kind. But this possibility should not be dismissed. The whole spectrum of embedded devices can be represented by a chart (Figure 1), which is roughly divided into four quadrants showing CPU complexity—broadly data bus width—against software complexity.

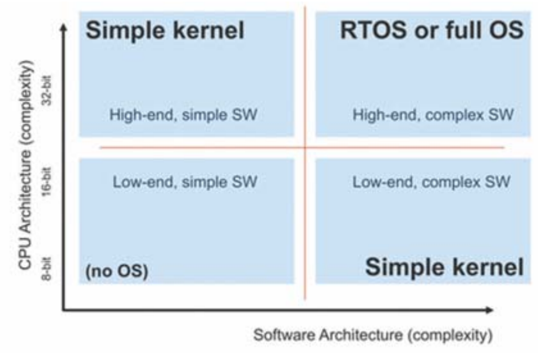


Figure 1: The four classes of embedded system (image: Mentor, a Siemens business)

The top right quadrant—complex software on a high-end processor—is the traditional province of real-time operating systems (RTOSes) and other operating systems. On a less powerful CPU, it may be useful to deploy a basic kernel, if the software is reasonably complex. Sometimes, a powerful chip is used to run quite simple software, where the required execution speed demands a certain level of CPU performance. In this case, a kernel may not strictly be required, but using one may be prudent as it improves the software architecture scalability and accommodates a future increase in complexity. It is really only when simple software is running on a low-end device that no kernel of any kind is necessary.

OS SELECTION CRITERIA

A series of inter-related, key questions should drive the decision-making process once you have opted to acquire an OS:

Is the application real-time?

A real-time system is not necessarily fast but responds to external events in a predictable and reproducible way. An OS that exhibits a high degree of determinism is termed a real-time operating system (RTOS). Determinism can vary. A "hard real-time" system is one where there are very tight time criteria; "soft real-time" is more relaxed.

For a hard real-time application, a true RTOS is really the only option. For a soft real-time application, an RTOS may still be used, but Linux (running on a CPU that is fast enough) may be acceptable.

Is the memory size-constrained?

All embedded devices have some kind of limitations on memory size but can still be quite large. However, unless there are many megabytes of RAM, the "heavyweight" OSes are not an option.

How much CPU power is available?

If the available microprocessor/microcontroller/core is only just fast enough for the application, the additional overhead of a heavyweight OS may be a problem. An RTOS is likely to make much more efficient use of the available CPU power.

2ew19P
Your e-code for free admission
embedded-world.de/voucher

Nürnberg, Germany

February 26 – 28, 2019

embeddedworld

Exhibition&Conference

... it´s a smarter world

TODAY, TOMORROW, AND BEYOND

Your one-stop resource for the entire spectrum of embedded systems: discover more than 1,000 companies and get inspired by the latest trends and product developments, by renowned speakers and exciting shows.

Keep up to date:
embedded-world.de

Media partners

Markt&Technik
DIE UNABHÄNGIGE WOCHENZEITUNG FÜR ELEKTRONIK

Computer & AUTOMATION
Fachmedium der Automatisierungstechnik

DESIGN& ELEKTRONIK
KNOW-HOW FÜR ENTWICKLER

SmarterWorld
Solutions for a Smarter World

Elektronik
Fachmedium für industrielle Anwender und Entwickler

MEDIZIN+elektronik
Fachmedium für Elektronik in der Medizintechnik

Elektronik automotive
Fachmedium für professionelle Automobilelektronik

elektroniknet.de

Exhibition organizer
NürnbergMesse GmbH
T +49 9 11 86 06-49 12
F +49 9 11 86 06-49 13
visitorservice@nuernbergmesse.de

Conference organizer
WEKA FACHMEDIEN GmbH
T +49 89 2 55 56-13 49
F +49 89 2 55 56-03 49
info@embedded-world.eu

NÜRNBERG MESSE

Is device power consumption a concern?
Power consumption is almost always a design parameter in modern devices, particularly for portable, battery-driven equipment. Software has a significant bearing on the power efficiency of a device. Factors like memory and CPU usage efficiency are important, but an OS may also include built- n power management facilities.

Does the design include unique or obscure peripherals?
Although many RTOS products offer very large ranges of middleware and drivers, Linux is likely to provide even more. So, for an obscure device, there may well be a Linux driver. For custom peripheral hardware, a driver would need to be written. This is a specialized activity. There is a plethora of Linux driver writing expertise available.

Does the design feature a memory management unit (MMU)?
If there is no MMU, the heavyweight OSes are unlikely to be an option; most RTOSes are fine without an MMU. If there is an MMU, Linux becomes a possibility. Many RTOSes can take some advantage of an MMU, if one is available.

Does the application need high security?
Specifically, is it necessary for tasks to be protected from one another?

If so, a process model OS is ideal, as each task can be a process; the OS uses the MMU to provide protection. This has a significant CPU time overhead and is offered by the heavyweight OSes.

Another option, supported by some RTOS products, is the implementation of "thread protected mode" or "lightweight process model" using an MMU. This has a lower overhead than a process model, but most of the security advantages.

Does the application require certification?
Some devices need to pass specific safety and quality standards and receive certification before they can be sold. Notably, this is the case for aerospace and medical applications. Such a certification process is expensive and requires access to all the source code (including the OS). The cost is significantly affected by the volume of source code to be analyzed, so a compact OS is advantageous.

In general, only a whole application can be certified. So, an OS cannot be certified by itself. However, selecting an OS that has a track record of successful certifications makes sense.

What is the end cost and shipping volume of the device?
Any OS has some costs associated with it, and these have an influence on the development and manufacturing costs of a device. A royalty bearing OS has a modest upfront cost and a charge for each device shipped, which may change according to volume. This is clearly good for low volume applications. A royalty free OS may have a slightly larger initial cost, but no ongoing license fees. Open source OSes sound as if they are "free," but in reality there are support costs which may be quite significant.

What's the team's experience?
Acquiring expertise is expensive, so leveraging existing knowledge and expertise is vital. An example is the API used by Linux: POSIX. There is a very large body of experience with POSIX, which is also supported by many of the available RTOS products on the market.

Naturally the availability, quality, and cost of documentation and support are critical factors to consider when selecting an OS.

MULTICORE
It is increasingly common for embedded designs to be implemented using multiple CPU cores. This may be multiple identical cores (homogeneous multicore) or the cores may be of different architectures (heterogeneous multicore).

Homogeneous multicore is often selected in order to increase the available CPU power in an energy efficient way. Harnessing this power effectively requires an OS that supports symmetrical multiprocessing (SMP), where a single instance of the OS runs on all the cores and distributes work between them. Most heavyweight OSes have an SMP variant available, and a number of RTOS products also feature an SMP support option.

An alternative software architecture, which is a possibility with homogeneous multicore, but the only option with heterogeneous, is asymmetric multiprocessing (AMP). In this case, each core runs its own OS instance. With an AMP design, the OS for each core may be selected individually. For example, cores performing real-time functions might use an RTOS, whereas others may run Linux. In an AMP design, communication between the cores is very likely, so the selection of OSes must take into account the availability of inter-core communications support.

CONCLUSIONS
The key thing that differentiates embedded systems and desktop computers is variability. To the first approximation, all Windows computers are the same. But every embedded system has unique characteristics, both technical and commercial. As a result, no single OS can satisfy the requirements of every design.

The selection process is complex, as there are many variables. A very wide choice of products on the market is inevitable but complicates the selection process.

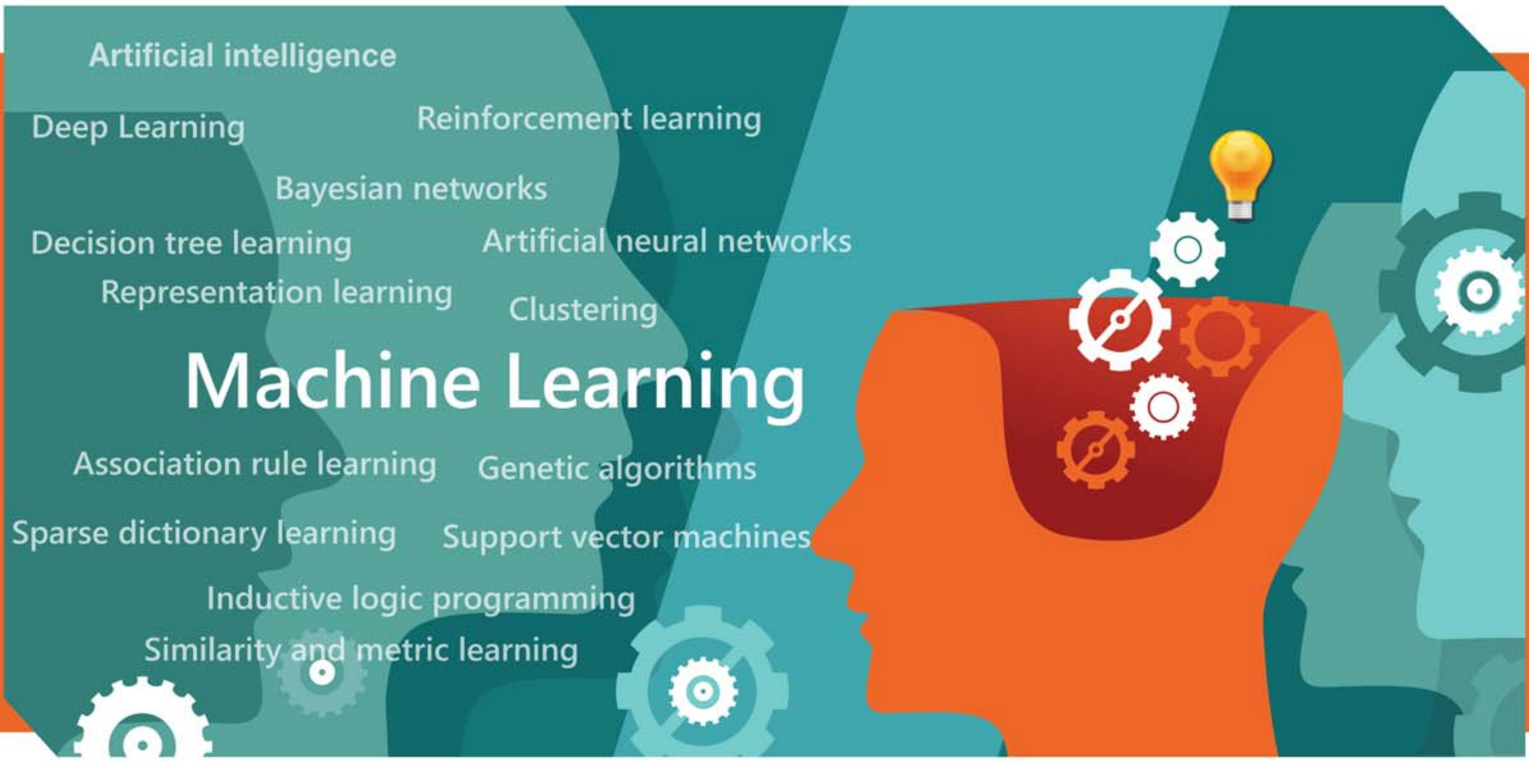
By methodically considering the requirements of a specific embedded system, a rational selection of OS may be made with confidence.

Colin Walls has nearly forty years' experience in the electronics industry, largely dedicated to embedded software. A frequent presenter at conferences and seminars and author of numerous technical articles and two books on embedded software, Colin is an embedded software technologist with Mentor, a Siemens business, and is based in the UK. His regular blog is located at: http://blogs.mentor.com/colinwalls

SAVE THIS DATE!

MACHINE LEARNING & AI DEVELOPERS CONFERENCE

JUNE 5-6, 2019
SANTA CLARA CONVENTION CENTER
SANTA CLARA, CA



The Only Conference and Exhibition Dedicated to the Technical Challenges of Machine Learning and Artificial Intelligence in Embedded Systems

www.MLDevCon.com

EMAC Inc.

CutiPy™ ARM Industrial IoT

Compatible Operating Systems: MicroPython, Free RTOS
Supported Architectures: ARM

Designed and Manufactured in the USA the CutiPy™ Industrial IoT microcontroller was created to simplify connecting devices and machines to the multitude of systems found in an industrial environment. EMAC Inc. has designed an easy to use embedded solution that can be implemented anywhere from the factory floor to an offsite remote location.

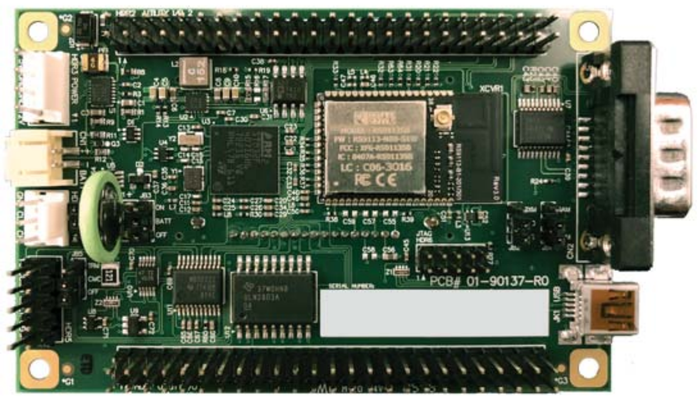
Based on the STMicroelectronics STM32F407IGH6, the CutiPy™ has an ARM Cortex-M4 processor running at 168MHz; with 192KB of SRAM, 1MB of internal flash and provides an SD card slot for additional storage. Standard IO interfaces are 2x USB 2.0 ports, 2x CAN 2.0B ports, 4x Serial ports, 2x SPI lines, 3x I2C connections, 24x GPIO connections (configurable as Timers, Counters, PWM, and GPIO), 8x High drive digital outputs, 13x 12-bit A/D ports, 2x 12-bit D/A ports and an onboard temperature sensor.

The CutiPy™ can be used with a rechargeable Lithium Ion battery connection for power, with built in charging circuitry from USB or a 5v power connector. Wireless module options provide connectivity on 802.11 a/b/g/n, Bluetooth, Thread and Zigbee networks. The CutiPy is ready for IoT success with simplified expansion.

Product URL:
http://www.emacinc.com/sales/cutipy

FEATURES

- Low Power Industrial IoT ARM microcontroller
- Graphic LCD and Push Buttons
- RTC with battery backup & Temperature Sensor
- 2x 50 pin Female Expansion Connectors
- Wifi, Bluetooth, Zigbee, Thread (Optional)



TECHNICAL SPECS

- STMicroelectronics ARM Cortex-M4 168 MHz w/Math Coprocessor
- 192 KB of SRAM, Up to 1MB of Flash, microSD card
- Up to 64x GPIO, 4x Serial Ports, 2x USB, SDIO, A/D, SPI, I2C & 2x CAN
- Rechargeable Battery Backup (option)
- Redpine RS9116 (BT/wifi/Zigbee) with on board antenna (optional)

APPLICATION AREAS

IoT, Agriculture, Industrial Automation, Intelligent Systems, Environmental Monitoring

AVAILABILITY

Now

CONTACT INFORMATION

EMAC Inc.
2390 EMAC Way
Carbondale, IL 62902
USA
Tel: 1 (618) 529-4525
Fax: 1 (618) 457-0110
info@emacinc.com
www.emacinc.com