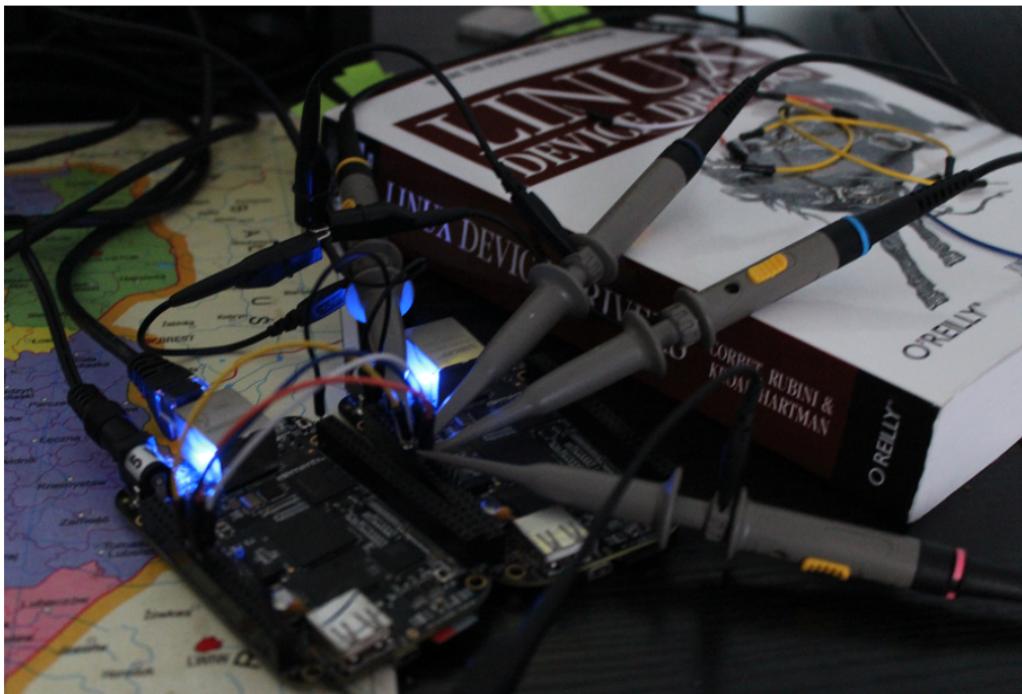


IoT Without a Net: A Practical Guide To Working With Microcontrollers The Open Source Way



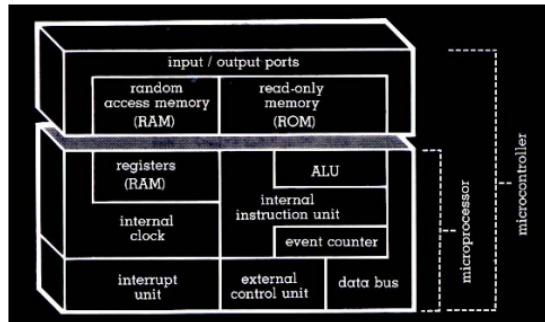
Presentation Outline

Exactly What Kind of Hardware Are We Talking About?	1
Example Reference Architecture	2
Example Boards	3
Microcontroller Architecture / CPU Families	4
Hybrid and "Combo" Boards	5
What About Software Tools?	6
Toolchains, SDKs, and Architectures	7
Bare-Metal and Firmware	7
Vendor vs. Open Source Tools	8
Example: ESP8266, Adafruit Feather HUZZAH	9
ESP8266 Cont.	10
Example: PRU-ICSS, TI BeagleBoneBlack	11
PRU-ICSS Cont.	12
Example: Atmel SAM3X8E, Udo / Udo Neo	13
SAM3X8E Cont.	14
How To Choose?	15
Where to Go Next	16
License and Thanks!	17

Exactly What Kind of Hardware Are We Talking About?

- Primary Characteristics
 - Stand-alone or separate real-time CPU core(s)
 - Can also come in combination or hybrid configurations
 - Requires firmware loaded at runtime to do something
 - Includes both hard and soft processor cores
 - Interfaces for debug and/or communication with Linux runtime
- Typical Applications
 - IoT, industrial, automotive, consumer, hobbyist
 - Machine control (3D printing, milling, process control)
 - Car navigation, entertainment, communication
 - Door/entry control, locks, power, LEDs, appliances
 - Autopilots (drones, rovers, UAVs) and robotics
 - Wearables, instrumentation, mesh networks

Example Reference Architecture

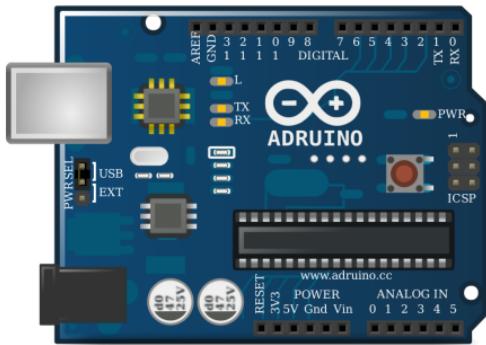


"A microcontroller (or MCU, short for microcontroller unit) is a small computer or System on Chip (SoC) in a single integrated circuit containing a processor core, memory, and programmable I/O peripherals (may also include program memory)."

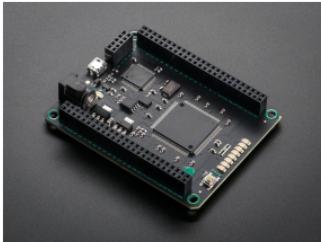
[1]

<https://en.wikipedia.org/wiki/Microcontroller>

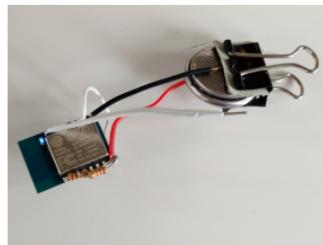
Example Boards



AVR/ARM: Arduinos



Spartan 6 XC6SLX9 FPGA



Xtensa LX-106: ESP8266



P8X32A microcontroller

Microcontroller Architecture / CPU Families

- 4 most "common" architecture families in DIY and FOSS
 - 8051, PIC and AVR are [Harvard architecture](#), which uses separate memory spaces for RAM and programs, while ARM is [von Neumann architecture](#) (program and RAM share the same memory space)
 - ARM is a 16 or 32 bit architecture, others are byte (8-bit) architecture
 - 8051 and PIC have limited stack space - limited to 128 bytes for the 8051, 8 words or less for PIC
 - 8051, AVR and ARM can directly address all available RAM, while PIC can only directly address 256 bytes
 - 8051 and PIC need multiple clock cycles per instruction, while AVR and ARM can execute most instructions in a single clock cycle
 - AVR and ARM have great open source compilers, libs, examples
- Still around: PowerPC, MIPS, STM, TI, Toshiba, Freescale/NXP, etc
- Combination and Hybrid Architectures/Implementations
 - [PRU-ICSS / PRUSSv2](#) - Programmable Real-Time Unit and Industrial Communication Subsystem
 - DSP - Digital Signal Processor
 - FPGA - Field Programmable Gate Array
 - Massively Parallel - Parallax Cog/Hub, ParallelA, Transputer

Hybrid and "Combo" Boards

What About Software Tools?

Toolchains, SDKs, and Architectures

Bare-Metal and Firmware

Each micro-controller/real-time core has an architecture and instruction set, which any toolchain must support (or at least a viable subset). Most have both vendor support and a gcc port, although in some cases the gcc port might be new and essentially still under development (eg, the TI PRUSS toolchain).

The toolchain itself is mainly the compiler/linker/loader and a standard C library (of which there are several). In order to build anything "fancy" you also need a Software Development Kit (or SDK). Sometimes you can live with either the compiler runtime and/or libc, but a good set of libraries with documented interfaces is always a Good Thing. Depending on the target arch there is almost always a vendor SDK and probably open source alternatives.

You should always do your homework (for a real project) or just pick one and try it and see...

- [AVR-Ada Blog](#) by Tero Koskinen
 - [tkoskine bitbucket repos](#) - Tools for AVR and ARM in multiple languages

Vendor vs. Open Source Tools

Example: ESP8266, Adafruit Feather HUZZAH

ESP8266 Cont.

Example: PRU-ICSS, TI BeagleBoneBlack

PRU-ICSS Cont.

Example: Atmel SAM3X8E, Udo0 / Udo0 Neo

SAM3X8E Cont.

How To Choose?

Where to Go Next

License and Thanks!

Author: Stephen L Arnold
FOSS Hat: Gentoo Linux Developer
Contact: nerdboy@gentoo.org
Revision: 0.1
Date: 2017-02-01, 19:47 PST8PDT
License: CC-Attribution-ShareAlike
Copyright: 2016 [Stephen Arnold](#)

