Single Board Computer

Rev. Stan Lewis

Overview – Single Board Computers

- Introduction
- What is an SBC?
- Examples
- How are they different from “other” architectures?
- Advantages and Uses

First a bit of introduction

- BS. Computer Science 1986
  - Light years ago, in computer world
  - Corporate Data Systems – “main frames”
- Area of Expertise
  - Storage Management - hardware
  - System Performance and Capacity Planning – statistics
  - Usually in the “Operating Systems Group”
- Saw Evolution of IBM Systems beyond the 360 Architecture
- Each iteration underlying system became more complex especially in the basic instruction set

Introduction (cont.)

- Transitioned to ministry in 1995
  - More attuned to desktop systems
  - Later SBCs and microcontrollers
- SBC (Single Board Computers) and MCU (micro control units)
  - hobby
  - Computer Controlled Machines (first one in graduate school/seminary)
  - 3d Printers
- Not an “expert” in SBCs
- Lastly, MUCH overlap as speeds and form-factors have evolved.

From the past

- Each iteration of IBM from 360, 370, to ESA
  - CISC – Complex Instruction Set Computers
  - Evolution of instruction set
  - Features added – reduce downtime, increase capacity
- Somewhat pejorative – Big Iron name
  - Systems became bigger – physically
  - Systems became faster – more chips, more storage
- Key take away – “Complex”
- Pressure to make things smaller physically
- Moore’s Law – transistors double every two years - observation

SBC Defined

Complicated because of Overlap with “All in Ones,” tablets, etc.

- Fully functional computer on one board
- Meaning CPU, GPU, Storage, Peripherals, and other supporting hardware
- Obviously limits upgradeability
- Supports normal network interfaces such as ethernet, Wifi, Bluetooth

SBC Defined (cont.)

Major difference – RISC Architecture

- X86 (and beyond) processors are CISC
  - CISC – Complex Instruction Set Computers
  - Started by Intel in 1970s
  - Basically, one instruction does more work under the covers
  - Hardware intensive – more transistors, etc.
- SBC are typically RISC – Reduce Instruction Set Computers
  - Started in 1980s – David Patterson & John Hennessy
  - Deal with "complexity" issues of CISC
  - Basically, one instruction does one action
  - Register-driven rather than memory driven

Examples

- Raspberry Pi (released in 2012)
- Clones and derivatives
- Designed to be more “open source”
- Science labs
- Now in the “wild” – robotics, automation (home & industrial), hobbyists
- Different from Microcontrollers such as Arduino et. al.

SBC vs Microcontrollers

- SBCs and MCUs are similar in that they are RISC
- MCUs require additional supporting electronics beyond power
  - Interface to sensors, motors, relays, etc.
  - Often need driver chips
- MCU programs typically written into the actual hardware (eeprom)
- MCU programs typically more C++ like compiled into machine code

SBC vs Microcontrollers

- SBCs can run with displays, keyboards
- Often have USB and HDMI support - cross lines towards “desktops”
- Things rapidly get fuzzy
- MCU typically run their code directly
- SBC typically run their code on top of an Operating System - LINUX
- Code is often written in Python or others

How are they “different”?

- Already talked about some of this
- Different underlying architecture and organization
- More “open source” – even that is fuzzy
- Most phones, tablets, other smart devices use RISC technology
- Cost and complexity of CISC development continues to increase

Biggest Difference

- CISC
  - Hardware focused
  - Hard and complex to develop
  - Cost prohibitive to change once the “die is cast”
- RISC
  - Software focused
  - Easier to develop
  - Easier to change the program than the hardware!
  - Development time on application side

Biggest Difference

- CISC
  - Hardware focused
  - Hard and complex to develop
  - Cost prohibitive to change once the “die is cast”
- RISC
  - Software focused
  - Easier to develop
  - Easier to change the program than the hardware!
  - Development time on application side

Circle Back MVS, VM, TPF

- Three Major IBM Operating Systems in 80s and 90s
- Each One very good at what they did
- Airline – Used all three
- MVS – Commercial & Business side – large amounts of data
- TPF – Airline Operations side – transaction oriented
- VM – Development side – safe & stable work platform for other two
- PCs were beginning to show up

Not about one better or worse

- Right one for the job!
- Each One very good at what they did – still the case across the spectrum
- IoT – Internet of Things
- Home Automation & Industrial Automation
- My favorite 3D printing.

Pros and Cons

- Gets hard here
- No clear winners
- Intel x86, AMD, and compatible processors are changing
  - Intel i7 is both on the same chip
  - Weight of Intel, Microsoft, and others
- Power consumption – SBC and RISC

3d Printing

- All three platforms – Desktop (x64), SBC (Raspberry Pi), Arduino
- Desktop – part design and prep
- SBC – handles the control/monitoring of the printer
- Arduino – fading out
  - Less hardware centric – configure, compile, burn, test, repeat
  - More software centric – configure and reboot
  - Still great many use Arduino for 3d printing
- CNC Plasma table – Arduino-like

Other Uses

- NAS boxes – SBC and a USB hard drive
  - Competes with “cloud” based
- Gaming Consoles - Retro Gaming
- Weather Stations
- IoT!
- Point of Sale and Kiosk
- Anything that requires “specialized functionality”

Blurred Lines…

- Both RISC and CISC hardware can accomplish same end use today
- Miniaturization packed more hardware on a chip inc. speed and memory
- Cost of hardware/performance has plummeted, eg SSD vs HDD, memory
- Core processing speed
- Programming language and environment advances
- Operating Systems – cross hardware

Specifically Raspberry Pi (rPi)…

- Run variety of OS especially Linux (other Unix derivatives)
  - Linux (raspberry pi OS, and derivatives) ++
  - Windows 10, Windows 10 IoT Core
- Run either mode
  - Desktop (monitor, keyboard, mouse etc)
  - Headless (no monitor, keyboard)
- Internet of Things -> Internet of Everything!