Cross Development for the CoCo

John W. Linville

18th Annual “Last” Chicago CoCoFEST!

28-29 March 2009

Who am I?

Why is this interesting?

Why use a modern workstation to develop CoCo software?

- Enjoy modern creature comforts
- Spend less time dealing with vintage problems
- Enable use of modern software engineering practices

What is this about?

Making the most of modern tools!

- Source composition and management
- Build management
- Object code generation
- Target communication
- Binary execution
- Debugging

Editing

Use a modern (or at least familiar) editor!

- Simplest form of cross development
- Use a “standard” keyboard
    - Type faster
    - Make less mistakes
- Even works with BASIC!

Code Generators

Use tools to write source code for you...

- Take the drudgery out of data translation
- Avoid error prone encoding
- Reduce reluctance for code/data changes

Revision Control

Let the computer track source code changes!

- Revision control is Software Engineering 101
- Does the CoCo even have this?
- Even if it does, modern tools are much better

Build Object Code

Let the computer manage your build!

- Build just what you need
- Build everything you need
- Build it the same way every time

Build Other Bits Too

Don't just build code...

- ROM/Disk/Cassette images
- Graphics or other data files
- Documentation
    - Doxygen, javadoc, etc.
    - Presentation binaries (e.g. PDF)

Scripted Execution

Build results can drive other events

- Session initialization
- Automated testing

ToolShed

ToolShed provides several build-related tools

- Assembler, linker, rdump
- Filesystem manipulation
- Open source, CoCo community

Choosing Operating Environments

Operating environment influences choice of tools and options for execution and debugging

- Capabilities, users, developer skills
- Objects formats, coding requirements, etc.
- Available libraries

Available Operating Environments

A plethora of choices are available!

- Cassette, DECB, ROM pak
- Replacement DOS, bare metal
- Color DOS, FLEX, OS-9
- Others?

Languages

Language choice influences tool choices

- Assembly
    - Wide variety of assemblers
    - Various pseudo-ops, output formats, etc.
- C
    - Microware
    - Dunfield
    - gcc6809
    - Small C, etc.

Other Languages

Assembly and C are not the only options...

- BASIC
    - CoCo ROM
    - Ragin' BASIC
- Pascal
- Forth
- Java
- Etc...

Assemblers

Assembly language is always available, but assemblers vary...

- Syntax quirks (e.g. FCS vs. FCCZ)
- Macro languages
- OS support
- Output formats
- Reporting capabilities

Most assembler problems can be worked-around, so pick one that you like...

Compilers

Many compilers are at least somewhat retargetable

- 6809 code generation
- Startup code
- Library support
- Operating environment requirements

Physical Machine

Obvious choice, but...

- Painful to transfer code
- Slower to setup/recover
- Possible to damage hardware, ruin disks, etc.

Emulation

Emulation is a good alternative, but not perfect!

- Code may not run on real hardware
- Looks good on LCD, not too good on CM-8
- Project may require un-emulated hardware

Hybrid Setup

Possible best of both worlds?

- DriverWire and/or CoCoNet
- Cassette emulation
- ROM emulation
- DLOAD?

Host-based Tools

Lots of debugging is done offline

- Hex editor
- Disk image tools
- Object dump tools
- Disassemblers

Native Debuggers

Native debuggers are equally useful under emulation

- EDTASM+ ZBUG
- OS-9 debug

Monitor Programs

Monitor programs provide a window into the soul of the machine...

- Emulator monitors (Vavasour, others?)
- Monitor programs over debug port
- Remote debuggers (DriveWire3, NoICE, etc.)

Emulated Hardware

Take advantage of open source emulators...

- Simulate hardware in development
- Add “hardware” that connects to the workstation

Let's get started!

Got a project? Maybe I can help?

- Tools
- Drivers
- ????

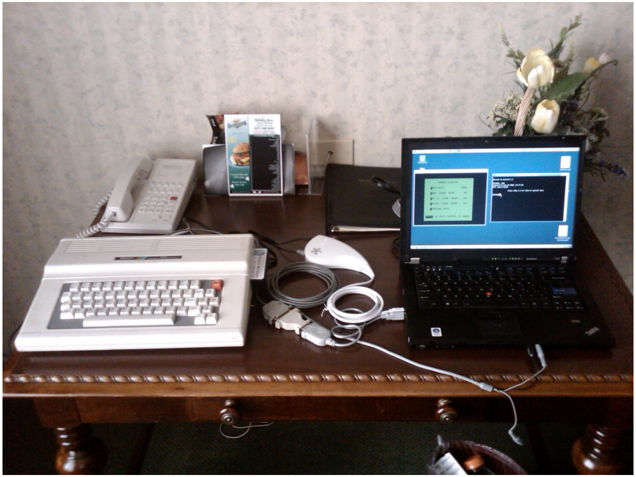
Demonstrations

(Semi-)prepared demonstration points...

- Revision control, host-based tools
- "Hello, world!" with absolute assemblers
- Generate BASIC loaders
- Upload code to the CoCo
- Debugging with a monitor program
- Verifying OS-9 modules

Impromptu demonstrations upon request!

Questions?



Contact

Feel free to contact me!

- Email linville@tuxdriver.com
  - ...@redhat.com
  - ...@gmail.com
  - ...@kernel.org
- IRC linville on FreeNode, OFTC, and LinuxNET
- Facebook as "John W. Linville"

Slides available:
http://www.kernel.org/pub/linux/kernel/people/linville/cocofest2009/