

Zuma Version 2.2 Upgrade notes:

Things to do/look out for in upgrading from Zuma 2.0 to version 2.1

These changes have been made in the M44 project, which can be used as a template project from which to copy these changes.

Device Driver

1. Modify the driver of all PCI cards to support shareable IRQs under 95 and NT.

Host DLL Mods

1. Modify the DLL to accommodate the new shareable interrupt driver, if needed.
2. Add the Pro Essentials graphics DLLs (PEGRAP32.DLL, PEGRAPHS.DLL) to the target's lib\host directory.
3. Create a Borland import library for the board.
4. Add the Hasp DLL (IIHASPDL.DLL) to the target's lib\host directory.
5. Add the Borland I/O DLL (BORIO.DLL) to the target's lib\host directory.

Applet Mods

1. Generate a version of Viewer for the target. This is a useful diagnostic aid. This utility should not be placed into the distribution.

Peripheral Library Mods

1. Add the Portable Interrupt Library functions to the Misc area of the PLIBs. Modify all example programs to use the new Portable library functions.
- 2.

Target Examples

1. Eliminate the COMPAND, SNAPDMA, ACQUIRE, BM_XFER, CFFT, IACK, PASSFLT, RAM and ROM examples and other non-essential examples.
2. Updates to the board TEST program
 - Make subtests menu-accessible
 - Assimilate the BM_XFER test into the standard board TEST program.
 - Embellish the digital test to incorporate loopback and self-test
 - Add a sync serial loopback test, if possible
 - Add a RAM and Flash ROM test, if possible

Host Examples

1. Port the Builder version of SCOPE. Eliminate the MSVC version.
2. Delete the RECORDER example

Documentation

1. Check out PLIB.DOC and HOST32.DOC from VSS and document all added/modified PLIB and DLL functions.
2. Email a change notice to webmaster who must re-release SW manual., CD distribution and web site libraries.

Zuma Version 2.1 Upgrade notes:

Things to do/look out for in upgrading from Zuma 2.0 to version 2.1

These changes have been made in the M44 project, which can be used as a template project from which to copy these changes.

Device Driver

2. Modify the driver to the new unidriver format.

Host DLL Mods

6. Modify the DLL to accommodate the unidriver.
7. Modify the DLL to add the mailbox_interrupt(), mailbox_interrupt_ack(), mailbox_interrupt_enable(), mailbox_interrupt_disable(), mailbox_interrupt_install() and mailbox_interrupt_deinstall() mailbox functions.
8. Add a private version resource to the DLL. Eliminate the DLL's version #define
9. Add the graphics sever files into the target's lib\host directory.

Applet Mods

2. Share in missing graphing components into the TERMINAL applet. Recompile all applets to make use of version resource information and the new COFF loader, which has been added to all of the applets.
3. Verify that hardware-keyed applets fail to run without a key.
4. Generate a version of Viewer for the target. This is a useful diagnostic aid.

Peripheral Library Mods

3. Upgrade boot.asm to new version from TI C 5.0. Share from M44cc\periph\rts. Write a custom iiboot..c which will perform all board-level initialization prior to main().
4. Modify all .mki files to eliminate references to -e c_int00
5. Rename board-specific peripheral bus support directory to periph\bus (i.e. on SBCs change from \PERIPH\SERIAL to \PERIPH\BUS).
6. Break all platform specific peripheral library functions out of PERIPH.H and place in separate header files. Include all of these .H files at bottom of PERIPH.H.
7. Make inline versions of all time-critical peripheral library functions, placing the inline code in subdirectories of the include\target directory, so that they can be readily located. Name the inline analog and digital I/O functions according to the new peripheral function naming convention in the M44 project. See the M44 header files for examples.
Note1: Retain the module site parameter in peripheral I/O functions even on boards without a modular I/O architecture.
Note2: Add paired and individual analog I/O channel and utility routines plus digital I/O support ala the M44 libraries.
8. Modify the peripheral libraries to add the mailbox_interrupt(), mailbox_interrupt_ack(), mailbox_interrupt_enable(), mailbox_interrupt_disable(), mailbox_interrupt_install() and mailbox_interrupt_deinstall() mailbox functions.
9. Code and add a cpu_speed() function, if possible. Use it to set MHZ in iiboot.c.
10. The ultimate goal is to have a single peripheral library function set which allows high-performance access access to all of the target's peripherals without any explicit memory accesses. If functions are needed to achieve this, invent them and add to the general peripheral library function pool and we'll try to write a version of the function for all targets. Similarly, rename any renegade plib functions to conform to documented plib functions wherever possible.
11. Add structures to header files describing hardware peripherals as appropriate and recode associated peripheral functions to improve code readability and performance.

12. Move VECTORS.ASM from root into PERIPH\RTS and .INCLUDE it in BOOT.ASM to force it's inclusion in each project.

Target Examples

3. Eliminate VECTORS.ASM from all projects as it's included in BOOT.ASM now.
4. Delete all Code Composer workspace-related files.
5. Be sure to enclose all Innovative-authored #include dependencies in quotes rather than brackets so that the makefile generator picks them up.
6. Recode the example programs to use the inline-able peripheral library functions instead of direct hardware accesses. Verify proper example operation.
7. Modify the CFFT, FFT, SNAP, SNAPDMA examples to use the new terminal graphing features. This will improve production testability. Consider other candidate examples as appropriate.
8. Add the timeplot.c and mailxrpt.c examples.
9. Recode example programs to allow sharing with architecturally-similar boards within VSS.
10. Attempt to share the Host code for SCOPE.C from the M44 to improve maintainability.
11. Modify the production test programs to streamline board testing wherever possible.
12. Build a Codewright example for every target example program.

Host Examples

3. Add the XRPT example which illustrates bi-directional interrupt usage.
4. Add the RECORDER example which illustrates high-speed data logging
5. Add the SCOPE example, which illustrates general DLL usage.

Documentation

3. Check out PLIB.DOC and HOST32.DOC from VSS and document all added/modified PLIB and DLL functions.
4. Email a change notice to webmaster who must re-release SW manual., CD distribution and web site libraries.