

Laboratorio di Sistemi Operativi  
Anno Accademico 2005-2006

Software Development with uMPS

Mauro Morsiani

Copyright © 2006 Mauro Morsiani  
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license can be found at:  
<http://www.gnu.org/licenses/fdl.html>

## uMPS simulator and user interface

### ? “Nothing is real”:

- ? What is uMPS?
  - ? it is a computer system
  - ? it is a simulator (that is, a set of programs)
- ? How to get it?
  - ? download from <http://mps.sourceforge.net>
  - ? compile and install (see INSTALL and README)
- ? Suggested configuration on your own PC:
  - ? recent Linux distribution on x86 (Debian, Ubuntu, Fedora...)
  - ? release 1.23-RC2 (RC1 is identical, but is better suited for non-root installations)

© 2006 Mauro Morsiani

2

## uMPS simulator and user interface

### ? uMPS simulator main commands:

- ? `umps`: the simulator itself
- ? `umps-elf2umps`: to convert the output of the compiler to files the simulator will understand
- ? `umps-objdump`: to analyze these files
- ? `umps-mkdev`: to build disks and tapes for the simulator

© 2006 Mauro Morsiani

3

## uMPS simulator and user interface

### ? uMPS simulator-related files:

- ? In the `support/` and `example*/` directory:
  - ? `*.rom.umps`: the ROM files
  - ? `*.core.umps`: the kernel to be loaded
  - ? `*.stab.umps`: the kernel *symbol table*
  - ? `*.aout.umps`: for programs other than kernel
  - ? other `*.umps` files (`term0.umps`, `printer0.umps`...): files associated to devices
  - ? `/etc/umpsrc` and `.umpsrc` (`ls -a` to see it): the simulator configuration file
  - ? `elf32*.x` files: configuration files for the cross-compiler

© 2006 Mauro Morsiani

4

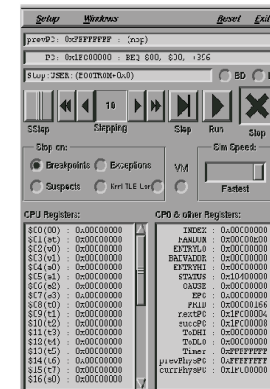
## uMPS simulator and user interface

### ? uMPS other essential components:

- ? some libraries (XForms, libelf) for building the simulator
- ? libumps.e (and libumps.o) under support/: uMPS library for interfacing with ROM services, **CP0** registers and issue TLB-related and SYSCALL instructions
- ? crtso.o and crti.o: kernel and program startup functions
- ? const.h and types.h under support/h: some useful types and constants (eg. processor state definition)
- ? a *cross-compiler* based on GNU gcc:
  - ? mipsel-linux-gcc for little-endian uMPS (on x86)
  - ? mips-linux-gcc for big-endian uMPS (on PPC)

## uMPS simulator and user interface

### ? "Out of the Matrix": uMPS main window



## uMPS simulator and user interface

### ? uMPS main window features:

- ? a *menu bar* with *Setup*, *Windows*, *Reset*, and *Exit*
- ? three *status lines* showing:
  - ? the last instruction executed
  - ? the instruction about to be executed
  - ? a status line with:
    - ? the current status of the simulator (running or stopped)
    - ? the reason the simulator stopped, if any (by user request or because some debugging event has happened)
    - ? the location of the instruction in a readable form
- ? Load and Branch Delay status flags

## uMPS simulator and user interface

### ? uMPS main window features (cont'd):

- ? a series of buttons for controlling the simulation progression (*SingleStep*, *Stepping*, *Step*, *Run*, *Stop*)
- ? a simulation speed slider (from *Slowest* to *Fastest*)
- ? a *Virtual Memory* indicator
- ? some *Stop on* buttons, to stop simulation on:
  - ? *Exceptions*
  - ? *Breakpoints*
  - ? *Suspects*
  - ? TLB-refill events in Kernel or User mode (requires also Exception to be selected)

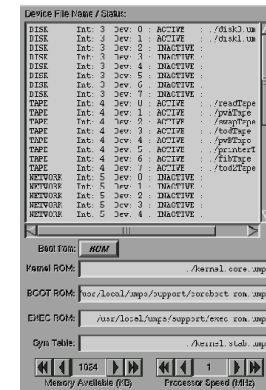
## uMPS simulator and user interface

### ? uMPS main window features (cont'd):

- ? main CPU registers
- ? CP0 registers
- ? some useful "meta-registers":
  - ? **nextPC**, **succPC**: next PC locations computed from the pipeline
  - ? **prevPhysPC** and **currPhysPC**: previous and current PC locations in physical memory (useful when VM is on)
- ? register values may be changed just by double-clicking on them

## uMPS simulator and user interface

### ? uMPS Setup Window



## uMPS simulator and user interface

### ? uMPS setup window features:

- ? the file list of `.umps` files associated to various devices
- ? the "Load core?" flag: to decide if the kernel file is loaded in RAM before the start of the simulation or not
- ? the file list of `.umps` files used for:
  - ? ROMs (`.rom.umps`)
  - ? kernel (`.core.umps` and `.stab.umps`)
- ? the RAM size selector (64KB – 2 MB)
- ? the processor speed selector (1-99 MHz)

## uMPS simulator and user interface

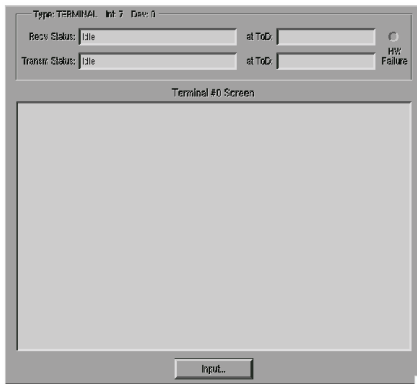
### ? uMPS setup menu contains:

- ? the *Setup Window* option
- ? the *Reset to Setup* option: to reset the simulator to the state described in the *Setup Window*, reloading all the files
- ? the *Reset to Defaults* option: to reset the simulator to the state described in the `.umpsrc` file, reloading all the files
- ? Some options may be configured only in the `.umpsrc` file:
  - ? TLB size
  - ? *Expert Mode*: no confirmation for operations
  - ? initial GUI setup (which buttons are on or off, etc.)



## uMPS simulator and user interface

### ? uMPS Terminal Status Window



17

## uMPS simulator and user interface

### ? uMPS Device Status Window details:

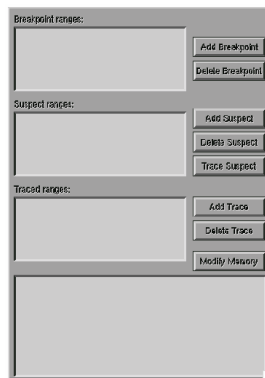
- ? for each device:
  - ? a status line describes what the device is currently doing
  - ? a TOD field tells when the current operation (if any) will end
  - ? a *HW Failure* button allows to set the device as "faulty" (all operations will not be performed and will return an error status)
- ? Terminal devices have also:
  - ? two device status lines (one for receiver, one for transmitter)
  - ? a screen to show the output
  - ? an *Input* button to allow line input

© 2006 Mauro Morsiani

18

## uMPS simulator and user interface

### ? uMPS Memory Browser Window



19

## uMPS simulator and user interface

### ? uMPS Memory Browser Window details:

- ? Breakpoint list and *Add / Delete* buttons: to show, set and remove Breakpoints
- ? Suspect list, and *Add / Delete / Trace* buttons: to show, set and remove *Suspect areas*
- ? Trace list and *Add / Delete / Modify* buttons: to show, set and remove memory traced ranges and to modify RAM contents
- ? memory content window: to show memory contents

© 2006 Mauro Morsiani

20

