

*Laboratorio di Sistemi Operativi*  
*Anno Accademico 2006-2007*

**Software Development with uMPS**

Mauro Morsiani

Copyright © 2007 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#TOC1>

# 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)

# 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

# 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

# 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:

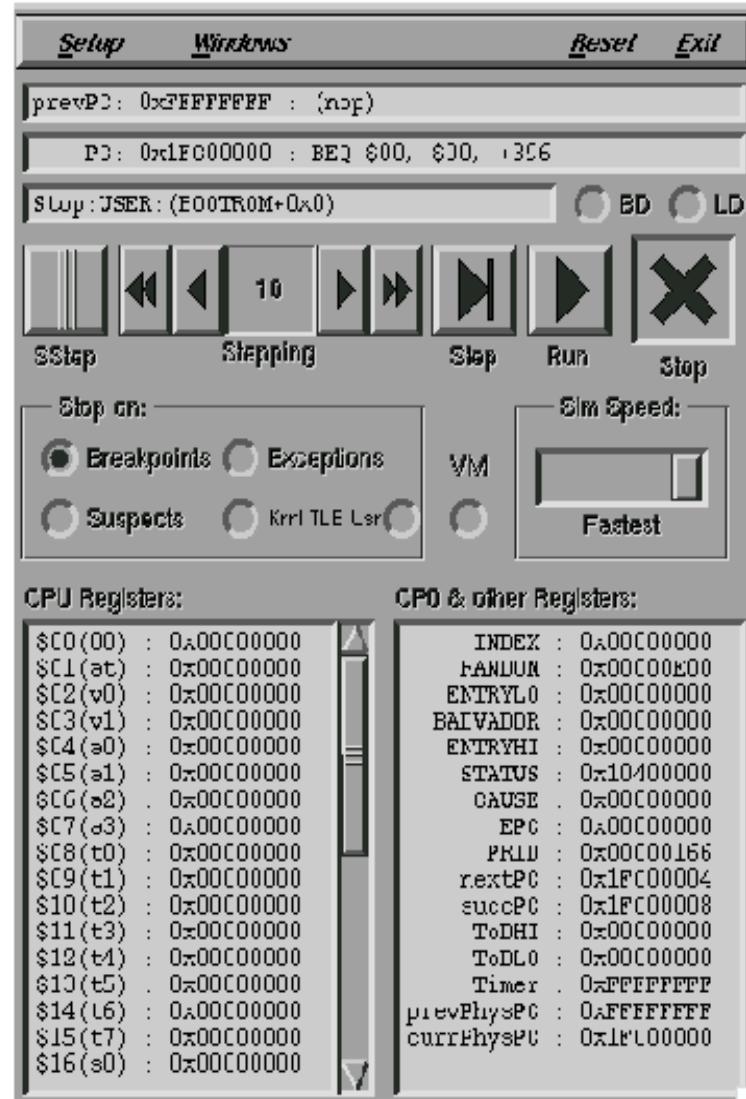
`mipsel-linux-gcc` for little-endian uMPS (on x86)

`mips-linux-gcc` for big-endian uMPS (on PPC)

the make utility

# 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

The screenshot displays the uMPS Setup Window with the following content:

**Device File Name / Status:**

DISK	Int: 3	Dev: 0	: ACTIVE	./disk0.un
DISK	Int: 3	Dev: 1	: ACTIVE	./disk1.un
DISK	Int: 3	Dev: 2	: INACTIVE	
DISK	Int: 3	Dev: 3	: INACTIVE	
DISK	Int: 3	Dev: 4	: INACTIVE	
DISK	Int: 3	Dev: 5	: INACTIVE	
DISK	Int: 3	Dev: 6	: INACTIVE	
DISK	Int: 3	Dev: 7	: INACTIVE	
TAPE	Int: 4	Dev: 0	: ACTIVE	./readTape
TAPE	Int: 4	Dev: 1	: ACTIVE	./pvATape
TAPE	Int: 4	Dev: 2	: ACTIVE	./swapTape
TAPE	Int: 4	Dev: 3	: ACTIVE	./todTape
TAPE	Int: 4	Dev: 4	: ACTIVE	./pvBTape
TAPE	Int: 4	Dev: 5	: ACTIVE	./printerT
TAPE	Int: 4	Dev: 6	: ACTIVE	./fibTape
TAPE	Int: 4	Dev: 7	: ACTIVE	./tod2Tape
NETWORK	Int: 5	Dev: 0	: INACTIVE	
NETWORK	Int: 5	Dev: 1	: INACTIVE	
NETWORK	Int: 5	Dev: 2	: INACTIVE	
NETWORK	Int: 5	Dev: 3	: INACTIVE	
NETWORK	Int: 5	Dev: 4	: INACTIVE	

**Boot from:**

**Kernel ROM:**

**BOOT ROM:**

**EXEC ROM:**

**Syn Table:**

**Memory Available (KB):**

**Processor Speed (MHz):**

# 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 required for most operations

initial GUI setup (which buttons are on or off, etc.)

# uMPS simulator and user interface

---

## **uMPS *Windows* menu contains:**

the *Symbol Table* option: to show the kernel Symbol Table

the *Memory Browser* option: to show memory contents, and to set Breakpoints and Suspects

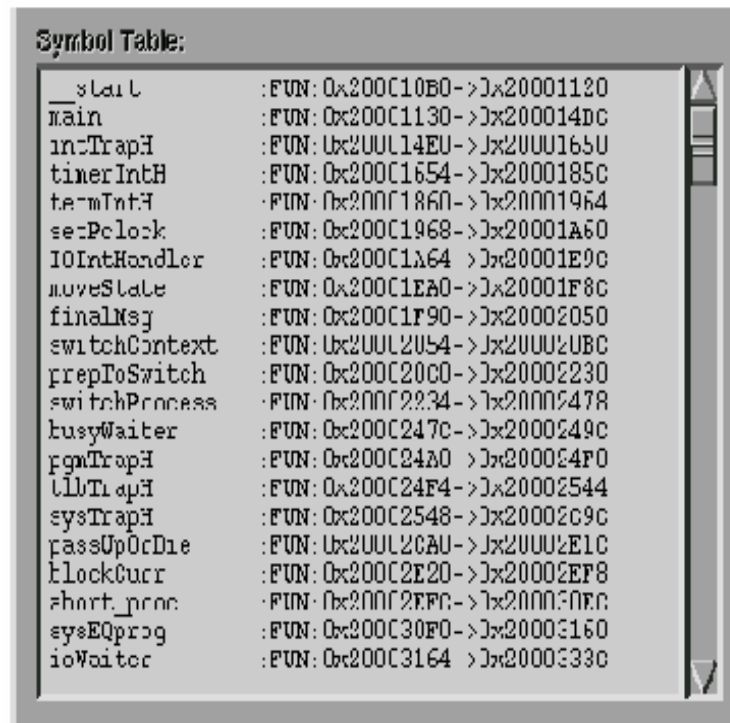
the *TLB Display* option: to show the TLB contents

*Int x Device Status* options: to show the devices' status

the *Terminal x* options: to show the terminals

# uMPS simulator and user interface

## uMPS Symbol Table Window



The screenshot shows a window titled "Symbol Table:" containing a list of symbols and their corresponding memory addresses. The symbols are listed on the left, and their addresses are listed on the right, separated by a colon. The addresses are in hexadecimal format, with some starting with "FUN:" and others with "Jx:". The symbols include: \_start, main, incTrapH, timerIntH, termIntH, secPclock, IOIntHandler, moveState, finalMsg, switchContext, prepToSwitch, switchProcess, busyWaiter, pgnTrapH, libTrapH, sysTrapH, passUpOrDie, blockCurr, short\_proc, sysEQprog, and ioWaiter.

Symbol	Address
_start	:FUN: 0x200C10B0->Jx20001120
main	:FUN: 0x200C1130->Jx200014DC
incTrapH	:FUN: 0x200C14E0->Jx20001650
timerIntH	:FUN: 0x200C1654->Jx2000185C
termIntH	:FUN: 0x200C1860->Jx20001964
secPclock	:FUN: 0x200C1968->Jx20001A60
IOIntHandler	:FUN: 0x200C1A64->Jx20001E9C
moveState	:FUN: 0x200C1EA0->Jx20001F8C
finalMsg	:FUN: 0x200C1F90->Jx20002050
switchContext	:FUN: 0x200C2054->Jx200020BC
prepToSwitch	:FUN: 0x200C20C0->Jx20002230
switchProcess	:FUN: 0x200C2234->Jx20002478
busyWaiter	:FUN: 0x200C247C->Jx2000249C
pgnTrapH	:FUN: 0x200C24A0->Jx200024F0
libTrapH	:FUN: 0x200C24F4->Jx20002544
sysTrapH	:FUN: 0x200C2548->Jx20002C9C
passUpOrDie	:FUN: 0x200C2CA0->Jx20002E1C
blockCurr	:FUN: 0x200C2E20->Jx20002EF8
short_proc	:FUN: 0x200C2EFC->Jx200030FC
sysEQprog	:FUN: 0x200C30F0->Jx20003160
ioWaiter	:FUN: 0x200C3164->Jx2000333C

# uMPS simulator and user interface

## uMPS TLB Display Window

```
TLB Status:
TLE [00] : 0x000C0000 : 0x000C0000
TLE [01] : 0x000C0000 : 0x000C0000
TLE [02] : 0x000C0000 : 0x000C0000
TLE [03] : 0x000C0000 : 0x000C0000
TLE [04] : 0x000C0000 : 0x000C0000
TLE [05] : 0x000C0000 : 0x000C0000
TLE [06] : 0x000C0000 : 0x000C0000
TLE [07] : 0x000C0000 : 0x000C0000
TLE [08] : 0x000C0000 : 0x000C0000
TLE [09] : 0x000C0000 : 0x000C0000
TLE [10] : 0x000C0000 : 0x000C0000
TLE [11] : 0x000C0000 : 0x000C0000
TLE [12] : 0x000C0000 : 0x000C0000
TLE [13] : 0x000C0000 : 0x000C0000
TLE [14] : 0x000C0000 : 0x000C0000
TLE [15] : 0x000C0000 : 0x000C0000
```

# uMPS simulator and user interface

## uMPS Device Status Window

Type: DISK	Int: 0	Dev: 0
Status:	File Transfer SUCCEEDED	
Completion of ToD:	<input type="text"/>	<input type="radio"/> HW Failure
Type: DISK	Int: 0	Dev: 1
Status:	File	
Completion of ToD:	<input type="text"/>	<input type="radio"/> HW Failure
Type: NULLDEV	Int: 0	Dev: 2
Status:	Not operational	
Completion of ToD:	<input type="text"/>	<input type="radio"/> HW Failure
Type: NULLDEV	Int: 0	Dev: 3
Status:	Not operational	
Completion of ToD:	<input type="text"/>	<input type="radio"/> HW Failure
Type: NULLDEV	Int: 0	Dev: 4
Status:	Not operational	
Completion of ToD:	<input type="text"/>	<input type="radio"/> HW Failure
Type: NULLDEV	Int: 0	Dev: 5
Status:	Not operational	
Completion of ToD:	<input type="text"/>	<input type="radio"/> HW Failure
Type: NULLDEV	Int: 0	Dev: 6
Status:	Not operational	
Completion of ToD:	<input type="text"/>	<input type="radio"/> HW Failure
Type: NULLDEV	Int: 0	Dev: 7
Status:	Not operational	
Completion of ToD:	<input type="text"/>	<input type="radio"/> HW Failure

# uMPS simulator and user interface

## uMPS Terminal Status Window

The screenshot shows a window titled "Terminal #0 Screen". At the top, it displays "Type: TERMINAL Int: 7 Dev: 0". Below this, there are two rows of status information. The first row shows "Recv Status: Idle" and "at ToD: [ ]" with a radio button labeled "HW Failure". The second row shows "Transm Status: Idle" and "at ToD: [ ]". A large, empty rectangular area occupies the center of the window. At the bottom center, there is a button labeled "Input..".

# 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:

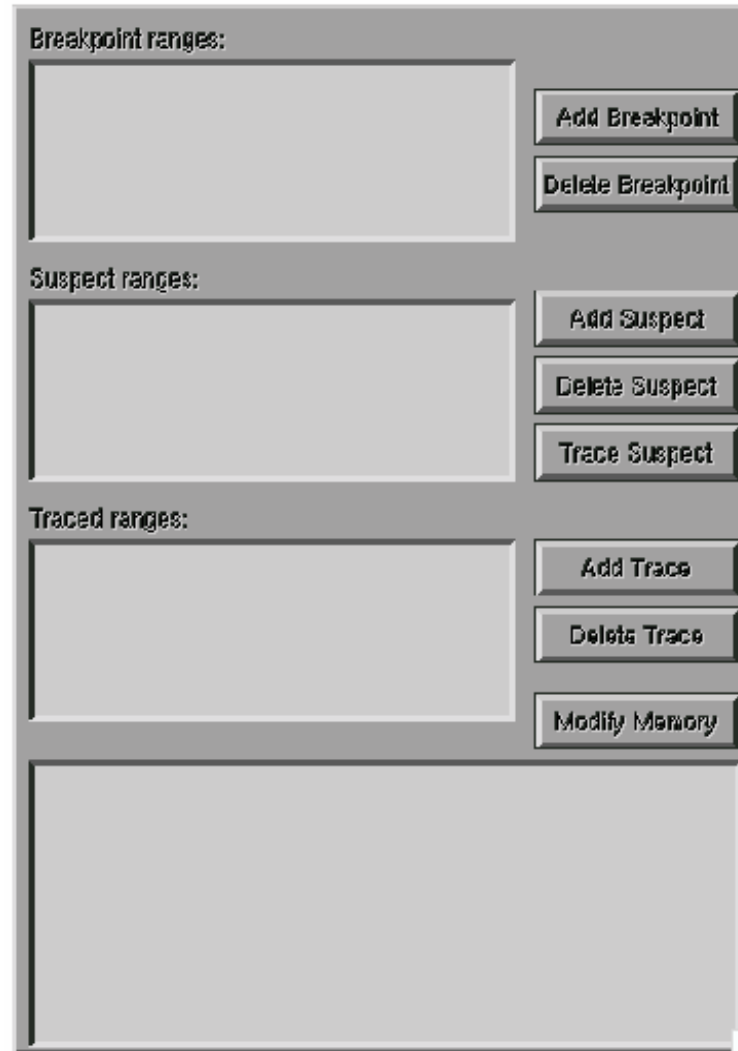
- two device status lines (one for receiver, one for transmitter)

- a screen to show the output

- an *Input* button to allow line input

# uMPS simulator and user interface

## uMPS Memory Browser Window



# 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

# uMPS simulator and user interface

---

## Breakpoint, Suspect and Trace: the debugger's tools of trade

*Breakpoint*: a position (an address) in the code; simulation stops when reaches it (may be referred to with a *symbol* + offset)

*Suspect area*: a memory range (a set of addresses) containing data (array, variables...) under exam; may be a *Read* suspect and/or a *Write* suspect (may be referred with a *symbol*)

*Suspect*: simulation stops when an access of the appropriate type (R, W) is made to the suspect area

*Traced range*: a range of memory addresses selected for showing

Addresses may be physical or virtual ones

Only physical addresses may be traced

# uMPS simulator and user interface

## uMPS: how to insert a Suspect range

