

uARM: a simple ARM virtual machine

Marco Melletti

1 dicembre 2014

Genealogia delle VM

- Chip: PDP-11, device ancora utilizzati (1983)
- MPS: MIPS, memoria virtuale sempre attiva (2004)
- uMPS: MIPS, memoria virtuale opzionale (2007)
- uMPS2: MIPS, supporto multicore, interfaccia grafica ristrutturata (2011)
- uARM

ARM: Advanced RISC Machine

- Architettura RISC: Reduced Instruction Set Computer
- Attuale e largamente utilizzata:
 - Embedded Systems
 - Smartphones
 - Nintendo DS
 - Raspberry Pi
 - Game Boy Advance
 - iPod
 - ...

uARM

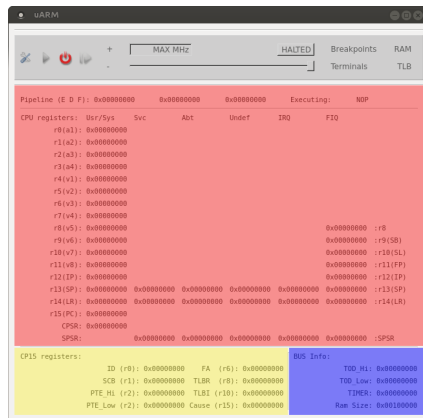
- Processore ARM7TDMI
- Memoria Little-Endian a dimensione variabile
- MMU con TLB a dimensione variabile
- 8 Device per tipo:
 - terminali
 - stampanti
 - schede di rete (VDE)
 - dischi fissi
 - nastri (dischi ottici)

uARM: GUI

The screenshot displays the uARM GUI with several windows open:

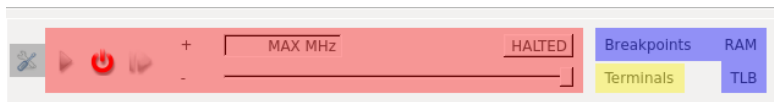
- Breakpoints:** Shows a table of breakpoints with columns for Function, Start, and End. The ASID is 0x7f and the Address is 0x0000.876c. The table lists breakpoints for main, prelinar..., setupTest, and TLBHandler.
- Machine Configuration:** Shows the General tab with hardware settings. Processors: 1, Default Clock Rate (MHz): 1, Views Refresh Rate: 600, RAM Size (Frames): 256, TLB Size (Entries): 16. BIOS settings include Execution ROM: uarm/facilities/BIOS.rom.uarm. Boot settings include Core file: test/dmatest.elf.core.uarm. Debugging Support includes Symbol Table: test/dmatest.elf.stab.uarm and Symbol Table ASID: 0x7f.
- uARM Terminal:** Shows the output of the uARM terminal, including the message "DMA transfer test.. preliminary test passed, setting up segment/page table s..".
- TLB:** Shows a table of TLB entries with columns for EntryHi and EntryLo. The table lists entries for main and TLBHandler.
- Ram Inspector:** Two instances of the Ram Inspector window are shown, displaying memory addresses and their corresponding values. The first instance shows addresses 0x07f0 to 0x0800, and the second instance shows addresses 0x0800 to 0x0800.

uARM: GUI



- Barra di controllo
- **Stato Processore**
- **Stato Coprocessore**
- **Informazioni di sistema**

uARM: Barra di controllo



- Configurazioni
- **Controllo esecuzione**
- **Terminali**
- **Funzioni di debug**

uARM: un esempio

Sorgente: foo/esempio.c

```
#include /usr/include/uarm/libuarm.h

int main(){
    tprint("Hellp World\n\0");
    HALT();
    tprint("");
    return 0;
}
```


Compilazione: un esempio

Sorgente: `foo/esempio.c`

Compiliamo il file:

```
$> arm-none-eabi-gcc -mcpu=arm7tdmi -c -o foo/esempio.o \  
foo/esempio.c
```

Linkiamo il file oggetto con la libreria di uARM e il file di inizializzazione:

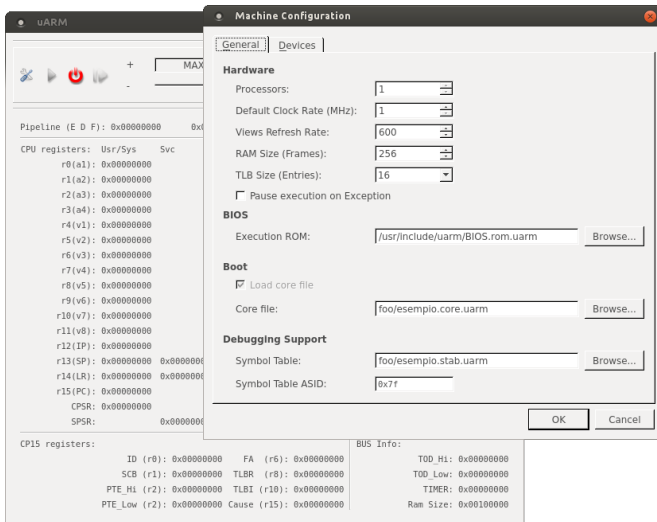
```
$> arm-none-eabi-ld -T \  
/usr/include/uarm/ldscripts/elf32ltsarm.h.uarmcore.x \  
-o esempio.elf /usr/include/uarm/crtso.o \  
/usr/include/uarm/libuarm.o esempio.o
```

Convertiamo l'eseguibile nel formato di uARM:

```
$> elf2uarm -k foo/esempio.elf
```

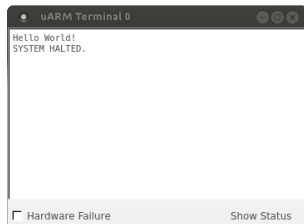
Esecuzione: un esempio

Impostiamo il core file generato per l'esecuzione:



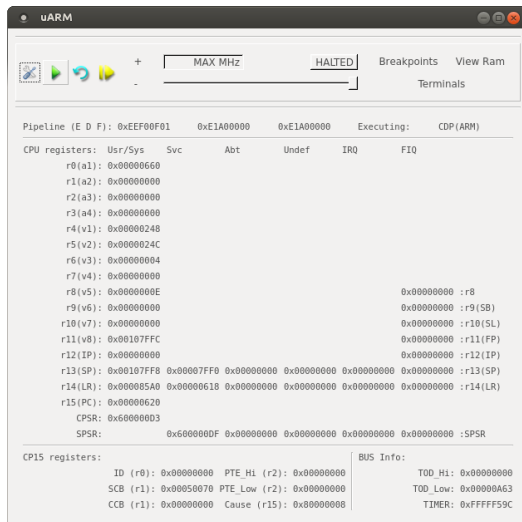
Esecuzione: un esempio

Avviamo la macchina e lanciamo l'esecuzione, il terminale 0 mostrerà l'output:



```
uARM Terminal 0
Hello World!
SYSTEM HALTED.

[ ] Hardware Failure      Show Status
```



uARM

MAX MHz HALTED Breakpoints View Ram Terminals

Pipeline (E D F): 0xEEF00F01 0xE1A00000 0xE1A00000 Executing: CDP(ARM)

CPU registers: Usr/Sys Svc Abt Under IRQ FIQ

r0(a1): 0x00000660
r1(a2): 0x00000000
r2(a3): 0x00000000
r3(a4): 0x00000000
r4(v1): 0x00000248
r5(v2): 0x0000024C
r6(v3): 0x00000004
r7(v4): 0x00000000
r8(v5): 0x0000000E 0x00000000 :r8
r9(v6): 0x00000000 0x00000000 :r9(SB)
r10(v7): 0x00000000 0x00000000 :r10(SL)
r11(v8): 0x00107FFC 0x00000000 :r11(FP)
r12(IP): 0x00000000 0x00000000 :r12(IP)
r13(SP): 0x00107FF8 0x00007FF0 0x00000000 0x00000000 0x00000000 :r13(SP)
r14(LR): 0x000085A0 0x00000618 0x00000000 0x00000000 0x00000000 :r14(LR)
r15(PC): 0x00000620
CPSR: 0x60000003
SPSR: 0x6000000F 0x00000000 0x00000000 0x00000000 0x00000000 :SPSR

CP15 registers:

ID (r0): 0x00000000	PTE_Hi (r2): 0x00000000	TOD_Hi: 0x00000000
SCB (r1): 0x00050070	PTE_Low (r2): 0x00000000	TOD_Low: 0x00000A63
CCB (r1): 0x00000000	Cause (r15): 0x80000000	TIMER: 0xFFFFF59C

uARM: un altro esempio

Proviamo davvero la macchina...

uARM

- Home page: <http://mellotanica.github.io/uARM/>
 - Repository ufficiale
 - Pacchetti per VirtLab
 - Questa introduzione
 - Specifiche della macchina
- Contattatemi per domande/problemi/richieste
 - marco.melletti@studio.unibo.it
 - mellotanica@hotmail.it

Grazie dell'attenzione

Domande?