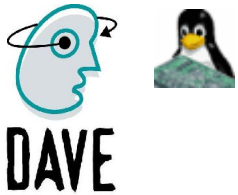


Pordenone, 27 Novembre 2004

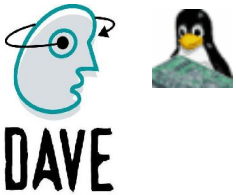
Linux Day 2004

**Esempi applicativi di Linux  
Embedded su architetture  
non x86: alcuni casi reali**



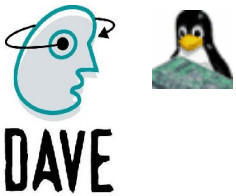
# Chi siamo

- DAVE S.R.L.
- via Forniz 2 , 33080 Porcia
- tel. +39.0434.921215
- fax +39.0434.591631
- e-mail: [info@dave-tech.it](mailto:info@dave-tech.it)
- web: [www.dave-tech.it](http://www.dave-tech.it)
- fondata nel 1998
- presentazione disponibile online:  
[www.dave-tech.it/download/misc/dave-ld04.pdf](http://www.dave-tech.it/download/misc/dave-ld04.pdf)



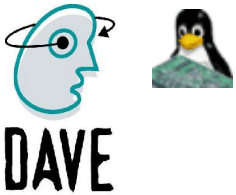
# Cosa intendiamo per sistema embedded

- classificazione (opinabile) dei processori (costo decrescente, volumi di produzione crescenti):
  1. general purpose: Pentiums II/III/IV, PowerPC, SPARC, Athlon ecc)
    - software general purpose (da applicazioni da ufficio a simulazioni di sistemi biologici)
    - s.o. “pesanti” (Unix, Linux, Windows NT ecc.)
    - applicazioni: Personal Computer, workstation ecc.
  2. processori embedded: ARM, x86 (AMD520, Geode), Hitachi SH-3/4, MIPS, PowerPC
    - singolo programma
    - s.o. estremamente ridotto, spesso real-time
    - supporto funzionalita' DSP
    - applicazioni: telefonia cellulare, elettronica di consumo, controllo industriale ecc.
  3. microcontrollori
    - il costo ridotto e' l'obiettivo fondamentale
    - parallelismo ridotto (tipicamente 8 bit)
    - volumi di produzione enormi
    - applicazioni: automobili, termostati, telecomandi ecc.



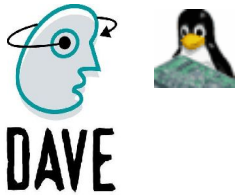
## Cosa intendiamo per sistema embedded (2)

- molte definizioni in letteratura; non c'e' una definizione universalmente riconosciuta
- sistema embedded:
  - è un computer specializzato, incorporato in un dispositivo fisico in modo tale che possa controllarne le funzioni tramite un apposito programma software dedicato
  - tipicamente è dotato delle risorse hardware minime indispensabili per espletare le funzioni per cui è preposto
- differenze rispetto ad un sistema PC classico:
  - frequenze di lavoro (potenza di calcolo) tipicamente di molto inferiori
  - tagli di memoria notevolmente inferiori
  - dispositivi di I/O spesso molto piu' primitivi o addirittura assenti
- molte architetture non x86 (ARM, PowerPC, MIPS, SH-4 ecc.) profondamente incompatibili tra loro
  - diversa endianness
  - diverso set di istruzioni
  - diversa organizzazione della memoria
  - ...



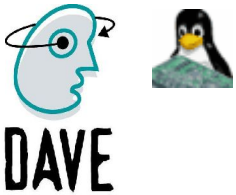
# Cosa intendiamo per sistema embedded (3)

- classificazione (opinabile) dei processori (costo decrescente, volumi di produzione crescenti):
  1. general purpose: Pentiums II/III/IV, PowerPC, SPARC, Athlon ecc)
    - software general purpose (da applicazioni da ufficio a simulazioni di sistemi biologici)
    - s.o. “pesanti” (Unix, Linux, Windows NT ecc.)
    - applicazioni: Personal Computer, workstation ecc.
  2. processori embedded: ARM, x86 (AMD520, Geode), Hitachi SH-3/4, MIPS, PowerPC
    - singolo programma
    - s.o. estremamente ridotto, spesso real-time
    - supporto funzionalita' DSP
    - applicazioni: telefonia cellulare, elettronica di consumo, controllo industriale ecc.
  3. microcontrollori
    - il costo ridotto e' l'obiettivo fondamentale
    - parallelismo ridotto (tipicamente 8 bit)
    - volumi di produzione enormi
    - applicazioni: automobili, termostati, telecomandi ecc.



# Cosa intendiamo per Linux Embedded

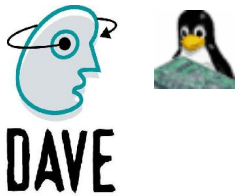
- sistema Linux Embedded: sistema embedded basato sul kernel di Linux, opportunamente configurato per svolgere lo scopo per cui e' stato progettato
  - root file system ridotto, spesso read-only
  - esclusione di sottosistemi/funzionalità non necessari (stack TCP/IP, PCI, USB, swapping ecc.)
- distribuzioni
  - “custom” costruite partendo dal software standard disponibile su Internet
  - distribuzioni commerciali



DAVE

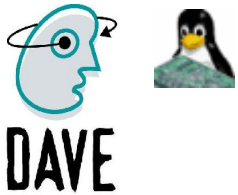
# Linux Embedded: i vantaggi (molti!)

- gratuita' del codice (sia del kernel che di moltissime applicazioni)
- affidabilita' dovuta all'enorme comunita' di sviluppatori/utilizzatori
- enorme quantita' di software disponibile, in buona parte rilasciato con licenza GPL o simile
- modularita' (filosofia UNIX)
- possibilita' di entrare in contatto diretto con gli autori del software (mailing list pubbliche)
- portabilita' del codice (POSIX)
- possibilita' di debuggare tutto o parte del codice applicativo sull'host (PC/workstation)
- numerose architetture supportate, tra cui anche quelle prive di MMU (uClinux - [www.uclinux.org](http://www.uclinux.org))
- supporto di moltissimi protocolli di comunicazione standard
- tool chain (GCC e binutils) gratuita (solitamente i compilatori per sistemi embedded costano migliaia/decine di migliaia di euro)



# Linux Embedded: gli svantaggi (pochi!)

- requisiti hardware abbastanza pesanti in confronto ai sistemi embedded tradizionali (in parte non più vero grazie ai costi dei microcontrollori a 32 bit di recente concezione)
- difficoltà nell'implementare sistemi real time o comunque con sufficiente reattività nei confronti degli eventi esterni (latenza agli interrupt)
  - soluzione #1: si ricorre a kernel real time che convivono con linux (es. RTAI)
  - soluzione #2: si può adottare il kernel 2.6 che ha migliorato notevolmente (un ordine di grandezza) la latenza agli interrupt ed il tempo di switch di contesto

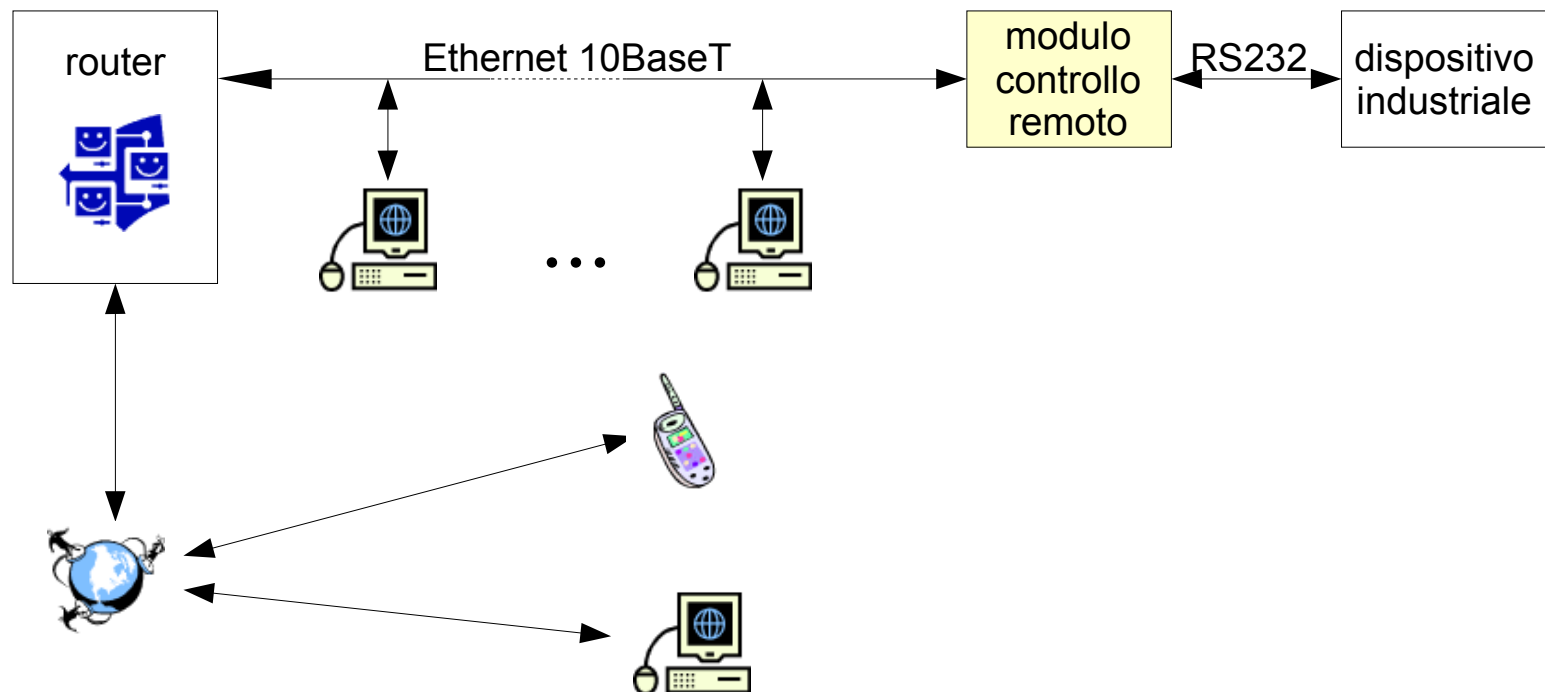


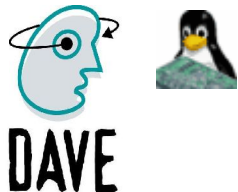
# Esempio applicativo #1

- Obiettivo: implementare il controllo/monitoraggio remoto via LAN di un dispositivo industriale non dotato di interfaccia di rete
- il controllo deve poter essere effettuato attraverso:
  - interfaccia web (web server integrato)
  - protocollo SNMP
  - protocolli proprietari da incapsulare in connessioni TCP

# Esempio applicativo #1 - soluzione

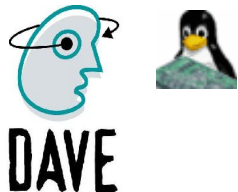
- CPU: ARM7 @ 75 MHz
- Memoria: 4 MB Flash / 16 MB RAM
- I/O: seriali (RS232), Ethernet 10BaseT
- software (uClinux): TCP/IP, web server open source, NET-SNMP, app. proprietaria





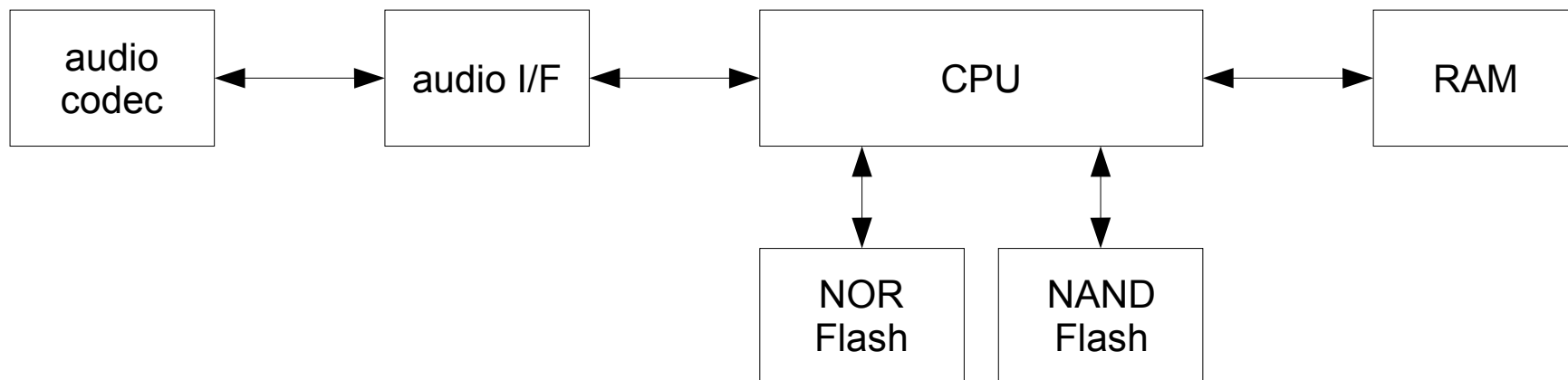
## Esempio applicativo #2

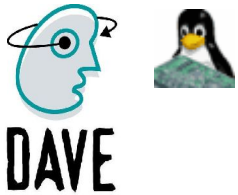
- Obiettivo: implementare un registratore/riproduttore audio digitale con compressione real-time
- la registrazione deve essere effettuata su un dispositivo di storage estremamente compatto e privo di organi meccanici (no hard disk!)
- possibilita' di disporre di piu' compressori contemporaneamente (MP3, G.722.2 ...) e di poterne aggiungere in futuro



## Esempio applicativo #2 - soluzione

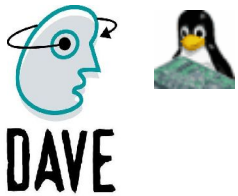
- CPU: PowerPC @ 333 MHz
- Memoria: 4 MB Flash (codice) / 32 MB NAND Flash (storage) / 32 MB RAM
- I/O: interfaccia audio proprietaria (ADC / DAC)
- software (Linux)
  - compressore/decompressore open source
  - MTD (Memory Technology Device) con file system opportuno (JFFS2, YAFFS ecc.)





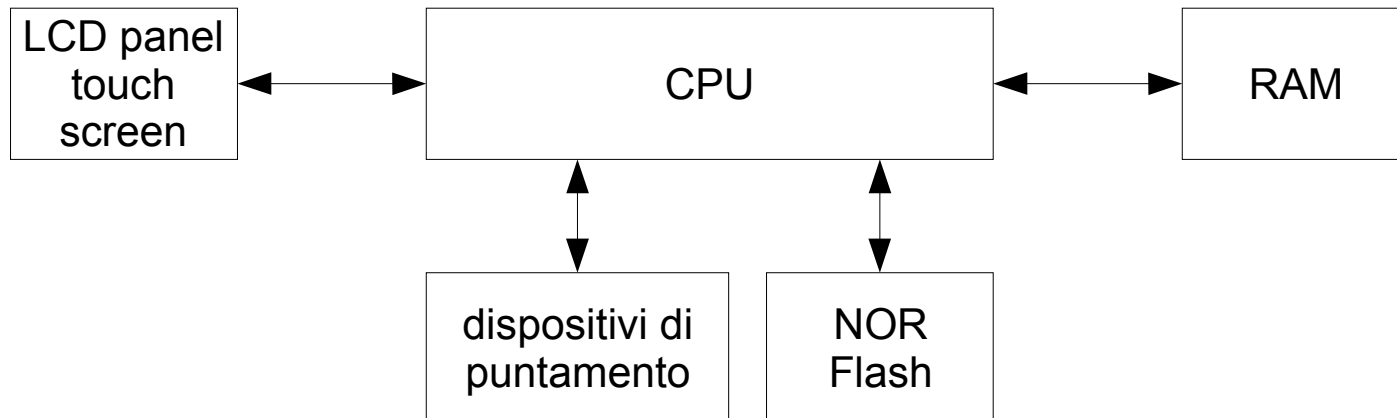
# Esempio applicativo #3

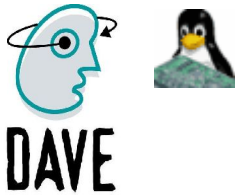
- Obiettivo: implementare un'interfaccia utente evoluta per dispositivi di elettronica di consumo
- l'interfaccia grafica deve supportare pannelli LCD di dimensioni variabili e touch screen
- lo sviluppo ed il debug dell'interfaccia grafica deve poter avvenire su PC



## Esempio applicativo #3 - soluzione

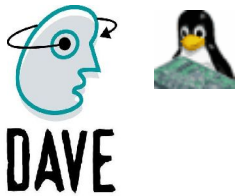
- CPU: ARM9 @ 200 MHz con controller LCD integrato
- Memoria: 8 MB Flash / 32 MB RAM
- I/O: interfaccia LCD/touch screen, dispositivi di puntamento/navigazione (pulsanti, manopole ecc.)
- software (Linux): microwindows, applicazione proprietaria





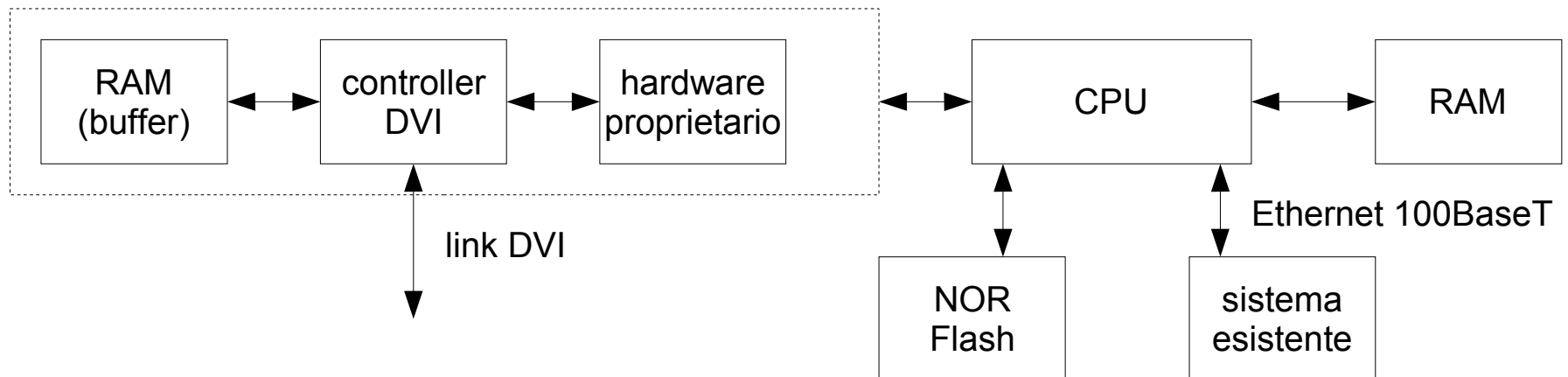
# Esempio applicativo #4

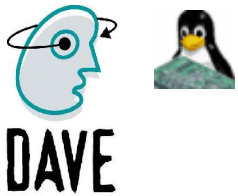
- Obiettivo: implementare un sistema per il trasferimento di immagini ad altissima risoluzione via Ethernet 100BaseT e conseguente trasmissione delle stesse su link DVI con cadenza regolare
- il sistema deve affiancarsi al sistema di supervisione esistente



## Esempio applicativo #4 - soluzione

- CPU: PowerPC @ 266 MHz
- Memoria: 4 MB Flash / 32 MB RAM (codice/kernel) / 128 MB RAM (buffering)
- I/O: interfaccia di rete 100BaseT, interfaccia DVI proprietaria
- software (Linux): stack TCP/IP, driver proprietari





# Conclusioni

- le esperienze “sul campo” hanno soddisfatto le aspettative riposte nell'uso di Linux nei sistemi embedded
- tutto fa supporre una notevole diffusione nei prossimi anni (indagini di mercato, sforzi fatti della casa di Redmond per non perdere fette del mercato embedded ecc.)