



Universitat Autònoma de Barcelona

Escola Tècnica Superior d'Enginyeria
Secció d'Enginyeria Informàtica

Benchmarks i mètriques d'avaluació de plataformes encastades multimèdia

Memòria del Projecte Fi de Carrera
D'Enginyeria en Informàtica
presentada per Joan Carles Chak Ma
i dirigida per Marc Moreno Berengué,
Guillermo Talavera Velilla i
Jordi Carrabina Bordoll

Bellaterra, 16 de juny de 2008

RESUM

Aquest projecte compara les possibilitats per a aplicacions multimèdia d'algunes de les arquitectures de processador que podem trobar en sistemes encastats.

Per fer-ho s'ha seleccionat una sèrie de benchmarks que inclouen una mostra d'aplicacions multimèdia, així com un conjunt de benchmarks que ens permet mesurar aspectes d'un sistema operatiu GNU/Linux.

També s'ha determinat quines haurien de ser les principals mètriques a considerar en el context dels sistemes encastats.

RESUMEN

Este proyecto compara las posibilidades para aplicaciones multimedia de algunas de las arquitecturas de procesador que podemos encontrarnos en sistemas embebidos.

Se ha seleccionado una serie de benchmarks que incluyen una muestra de aplicaciones multimedia, además de un conjunto de benchmarks que nos permiten medir aspectos de un sistema operativo GNU/Linux.

También se ha determinado cuales serían las principales métricas a considerar en el contexto de los sistemas embebidos.

SUMMARY

This project shows the possibilities in multimedia applications of some processor's architectures that can be found on embedded systems.

To achieve this goal, it has been selected a battery of benchmarks that contains a sample of multimedia applications, and also some benchmarks that allows us to measure some features of a GNU/Linux operative system.

Furthermore, it had been determined which metrics should we consider under a embedded system context.

Els signants, en Guillermo Talavera Velilla i Jordi Carrabina Bordoll, professors del Departament de Microelectrònica i Sistemes Electrònics de la Universitat Autònoma de Barcelona

Certifica

Que el treball corresponent a la present memòria ha estat realitzat sota la nostra direcció per Joan Carles Chak Ma. I perquè així consti ho signo

Guillermo Talavera Velilla i Jordi Carrabina Bordoll

Bellaterra, a 16 de juny de 2008

There is an end to this begin.

AGRAÏMENTS

A la meva família, pel seu suport i infinita paciència.

Als meus companys de pis per tots els anys de convivència pacífica en un entorn poc saludable però al menys molt amistós.

A tots els amics i companys que m'han acompanyat durant aquests darrers anys, en especial els que m'han hagut de suportar.

A Marc Moreno (malgrat els seus acudits dolents sobre xinesos), pel seu suport moral i tècnic. A Guillem Talavera per la seva ajuda, santa paciència, i per recordar-se de mi per a fer el projecte amb Mitsubishi. A David Castells, per els seus intents de conèixer els continguts d'aquest PFC, i que al final han resultat útils per extreure algun que altre consell útil. Als companys de Tecnològics per suportar les tribulacions electròniques d'un informàtic. I a Jordi Carrabina per la oportunitat que em va oferir ara fa més d'un any.

A tots els companys de feina de CEPHIS per la vostra ajuda, suport, amiat, i bons moments. La meva procrastinació amb el PFC ha acabat essent un infern gràcies a vosaltres.

Finalment, a la gent de Mitsubishi per confiar en nosaltres per el projecte IT1000 i per cedir-nos les seves plataformes encastades, sense les quals aquest projecte no hagués estat possible.

Índex

Objectiu.....	1
Capítol 1. Introducció i anàlisi.....	2
1.1. Introducció als sistemes encastrats.....	2
1.2. Processadors de sistemes encastrats.....	4
1.2.1. ARM.....	4
1.2.2. X86.....	6
1.3. Introducció als benchmarking	8
1.4. Mètriques de caracterització.....	10
1.5. Justificació de l'elecció.....	11
1.5.1. MediaBench 1.0.....	12
1.5.2. Unixbench 4.1.0.....	13
1.5.3. MiBench 1.0.....	13
Capítol 2. Entorn de Desenvolupament i Metodologia de Treball.....	15
2.1. Plataformes seleccionades.....	15
2.2. Eines i entorns utilitzats.....	17
2.3. Adaptació de components / mòduls	18
Capítol 3. Test i Resultats.....	20
3.1. Entorn de test.....	20
3.2. Tests i resultats.....	21
3.2.1. Benchmarks multimèdia.....	21
3.2.1. Consums.....	24
3.2.2. Preu.....	27
3.2.3. Mida.....	27
3.3. Altres resultats.....	28

3.3.1. Unixbench.....	28
3.3.2. QEMU.....	29
Capítol 4. Conclusions.....	32
4.1. Conclusions.....	32
4.2. Experiència personal i professional.....	33
4.3. Evolució futura.....	34
Referències.....	37
Bibliografia.....	39
Annexos.....	40
Unixbench: components i mòduls.....	40

Llista d'acrònims

ADPCM:	<i>Adaptive Differential Pulse Code Modulation</i>	MPEG:	<i>Moving Picture Experts Group</i>
AMD:	<i>Advanced Micro Devices</i>	TIFF:	<i>Tagged Image File Format</i>
ARM:	<i>Advanced RISC Machine</i>	OMAP:	<i>Open Multimedia Application Platform</i>
ATM:	<i>Automated Teller Machine</i>	PATA:	<i>Parallel AT Attachment</i>
CISC:	<i>Complex Instruction Set Computer</i>	PCM:	<i>Pulse Code Modulation</i>
CF:	<i>Compact Flash</i>	PDA:	<i>Personal Digital Assistant</i>
CPU:	<i>Control Processor Unit</i>	QoS:	<i>Quality of Service</i>
DDR:	<i>Double Data Rate</i>	RAM:	<i>Random Access Memory</i>
DSP:	<i>Digital Signal Processor</i>	ROM:	<i>Read Only Memory</i>
EEMBC:	<i>Embedded Microprocessor Benchmark Consortium</i>	RISC:	<i>Reduced Instruction Set Computer</i>
GCC:	<i>GNU Compiler Collection</i>	SATA:	<i>Serial AT Attachment</i>
GNU:	<i>GNU's Not Unix</i>	SBC:	<i>Single-Board Computer</i>
JPEG:	<i>Joint Photographic Experts Group</i>	SDK:	<i>Software Development Kit</i>
LVDS:	<i>Low-voltage differential signaling</i>	TI:	<i>Texas Instruments</i>
MID:	<i>Mobile Internet Device</i>	ULV:	<i>Ultra Low-Voltage</i>
MIPS:	<i>Millions Instructions Per Second</i>	UMPC:	<i>Ultra Mobile PC</i>
MP3:	<i>MPEG-1 Audio Layer 3</i>	USB:	<i>Universal Serial Bus</i>
		VGA:	<i>Video Graphics Array</i>
		WAV:	<i>Waveform (audio format)</i>

Llista de figures

Figura 1: Sistemes encastats en un cotxe.....	2
Figura 2: Evolució del número de processadors RISC distribuïts.....	5
Figura 3: Lincroft serà un SoC que integrarà un processador basat en x86.....	8
Figura 4: Plataformes seleccionades.....	16
Figura 5: Emulador ARM en QEMU.....	18
Figura 6: MiBench: compressió i descompressió JPEG, i descompressió MP3.....	21
Figura 7: Operacions amb imatges TIFF.....	22
Figura 8: Operacions amb imatges TIFF i tractament de text.....	22
Figura 9: Conversió d'ADPCM a PCM i descompressió de fitxers MPEG.....	23
Figura 10: Consum de les plataformes.....	25
Figura 11: Relació rendiment/consum de les plataformes	26
Figura 12: Resultats del benchmarking al sistema de fitxers.....	29
Figura 13: Resultats MiBench amb QEMU.....	30
Figura 14: Resultats MiBench i MediaBench amb QEMU.....	30

Llista de taules

Taula 1: Consums màxims i típics de les plataformes.....	24
Taula 2: Preus de les plataformes.....	27
Taula 3: Mida de les plataformes.....	28

Objectiu

Gràcies als diferents avenços que s'han aconseguit en els darrers anys, avui en dia no resulta estrany trobar-se sistemes encastats amb la capacitat de processament necessària per a reproduir gairebé tot tipus de continguts multimèdia.

L'objectiu principal d'aquest projecte consistirà en mostrar quines són les possibilitats per a aplicacions multimèdia de diferents plataformes encastades de les que disposem, i que són una petita mostra de les diverses arquitectures de processador que ens podem trobar normalment en una plataforma avui en dia.

Per arribar a aquesta fita, s'ha seleccionat dos conjunts de benchmarks: el primer emmarcat en l'àmbit multimèdia (compressió i/o descompressió de continguts en format JPEG, TIFF, MPEG i ADPCM); i el segon conjunt per mesurar diversos aspectes relacionats amb el sistema operatiu, que per aquest projecte s'ha limitat només a distribucions GNU/Linux.

Finalment, s'ha realitzat una comparació entre les diferents plataformes amb els resultats obtinguts.

Capítol 1. Introducció i anàlisi

1.1. Introducció als sistemes encastats

Dels 9.000 milions de processadors que es varen fabricar a l'any 2005, només un 2% d'aquests estaven destinats a ordinadors personals, estacions de treball o servidors. El 98% restant (o el que és el mateix, 8.800 milions de processadors) varen tenir com a destí un sistema encastat [1].

Aquesta dada ja ens dóna una idea prou significativa de la forta presència, tot i que potser no siguem plenament conscients, dels sistemes encastats en el nostre dia a dia, ja sigui en aplicacions tan diferents entre si com ho poden ser per exemple l'automoció (integrats en sistemes antibloquejants de frens, navegació GPS, control d'airbags), les comunicacions (telèfons mòbils, routers, mòdems), l'electrònica de la llar (integrats en rentadores, rentaplats, microones) o multimèdia (videoconsols, reproductors de vídeo/música, quioscs de revelatge de fotos...).

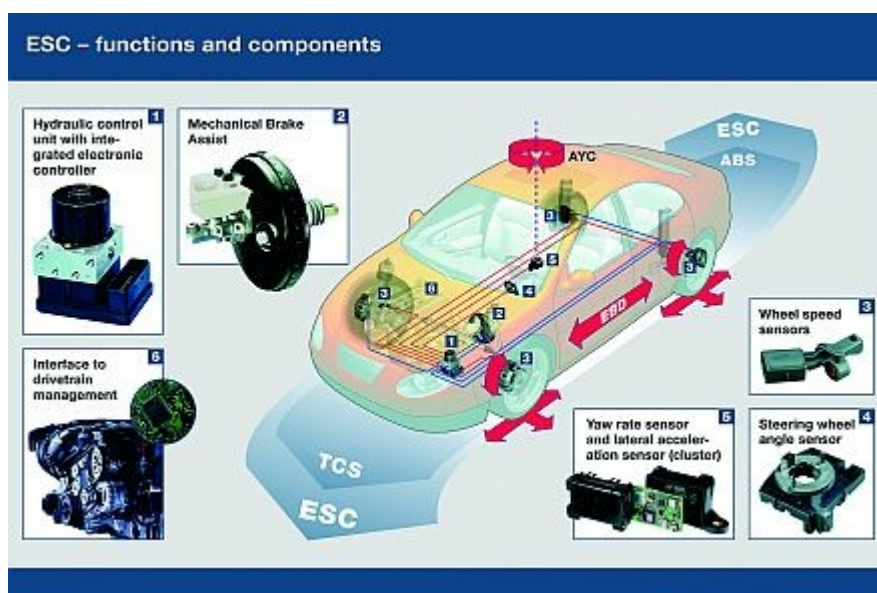


Figura 1: Sistemes encastats en un cotxe. (Font: www.conti-online.com)

Dit això, és important diferenciar el que és un sistema encastat d'un sistema de propòsit general: totes dues són plataformes de computació, però en el cas de l'encastat

la seva finalitat està limitada a realitzar una tasca en concret, o un conjunt molt limitat de tasques. Per altra banda, un sistema de propòsit general (com podria ser per exemple un ordinador personal, o un servidor) està preparat per realitzar diverses tasques segons el programari del que disposi.

Malgrat que els diferents àmbits on podem trobar-nos sistemes encastats tinguin poc o res a veure entre ells, els sistemes encastats en general comparteixen una sèrie de trets comuns[2]:

- Són d'aplicació específica: en principi només poden realitzar la tasca o tasques per les quals han estat fabricats.
- Uns requeriments específics. Per exemple, alguns sistemes encastats han de ser en temps real (sincronitzats amb les peticions de l'entorn), o complir unes limitacions de qualitat de servei (QoS). Altres requeriments poden estar relacionats amb l'entrada/sortida de dades.
- Un funcionament predefinit. Els sistemes encastats tenen definits uns fluxos concrets a l'hora de realitzar les tasques per les quals han estat construïts i no permeten fer un ús del sistema encastat per el qual no ha estat definit.
- En principi, no són programables per l'usuari. La part software d'un sistema encastat (que normalment s'anomena firmware) pot estar emmagatzemada en una memòria de tipus només-lectura o en memòries Flash. En aquest últim cas, tot i que pot existir alguna possibilitat de sobreescriure el contingut de la memòria, no sempre es faciliten a l'usuari les eines necessàries per crear-se un firmware propi, o programar el dispositiu per “personalitzar-lo” o donar-li un ús diferent del que té.
- Quan parlem de computació mòbil (com per exemple telèfons mòbils, videoconsoles, etc.) l'autonomia dels dispositius també és un factor molt important a tenir en compte. Això fa que els sistemes encastats que es dissenyen en aquests casos tinguin com a requeriment addicional un consum energètic que sigui el més baix possible per tal d'allargar la vida de la bateria.

1.2. Processadors de sistemes encastrats

A diferència del que succeeix en els sistemes de propòsit general (ordinadors personals, estacions de treball, servidors, etc.), on quasi tota la quota de mercat se la reparteixen majoritàriament dos fabricants (Intel i AMD) i hi ha una arquitectura de processador predominant (la x86), en els sistemes encastrats ens podem trobar amb una gran varietat de processadors de diferents arquitectures i fabricants.

Per posar una sèrie d'exemples, algunes de les arquitectures de processador (i els seus respectius fabricants/dissenyadors) que hi poden haver en un sistema encastrat són: 65816, 65C02 (Western Design Center); 68HC08, 68HC11, 68k, ColdFire (Freescale/Motorola); ARM (ARM Ltd.); AVR (Atmel); C167, Tricore (Infineon); eZ8, eZ80, z80 (Zilog); H8 (Renesas Technology); MIPS (MIPS Technologies); PIC (Microchip Technology); PowerPC (Apple, IBM, Motorola); SHARC (Analog Devices); SuperH (Hitachi/Renesas Technology); TLCS-47, TLCS-870, TLCS-900 (Toshiba); V850 (NEC Electronics Corporation); x86 (variïts).

Les raons d'aquesta diversitat les podríem trobar en diversos motius, però fent una mica de síntesi i des d'un punt de vista “d'enginyeria” podríem dir que bàsicament hi ha tanta varietat degut a que cadascuna d'aquestes arquitectures es pot adaptar millor a alguna necessitat o requeriment concret (mida reduïda, baix consum, sense necessitats especials de refrigeració, etc).

Donat el context en que es desenvoluparà aquest projecte, només s'ha treballat sobre ARM i x86, ja que són dues arquitectures que podem trobar-nos en un sistema encastrat (sobre tot la primera) i que tenen en principi prou capacitat de processament per realitzar tasques multimèdia.

1.2.1. ARM

ARM (Advanced RISC Machine, anteriorment Acorn RISC Machine) és una arquitectura basada en processadors RISC de 32 bits i desenvolupada en l'actualitat per ARM Limited (companyia que es va crear al 1990 com a *joint-venture* entre Acorn Computers, Apple Computer i VLSI Technology).

Actualment l'arquitectura ARM domina el mercat de processadors RISC de 32 bits per a sistemes encastats: a l'any 2005 es va estimar que de tots els processadors RISC de 32 bits que hi havia al mercat, el 75% pertanyien a alguna família de processadors ARM[3]. O dit amb altres dades, segons la mateixa ARM Limited els seus *partners* varen distribuir fins aquelles dates més de 2.500 milions de processadors ARM.

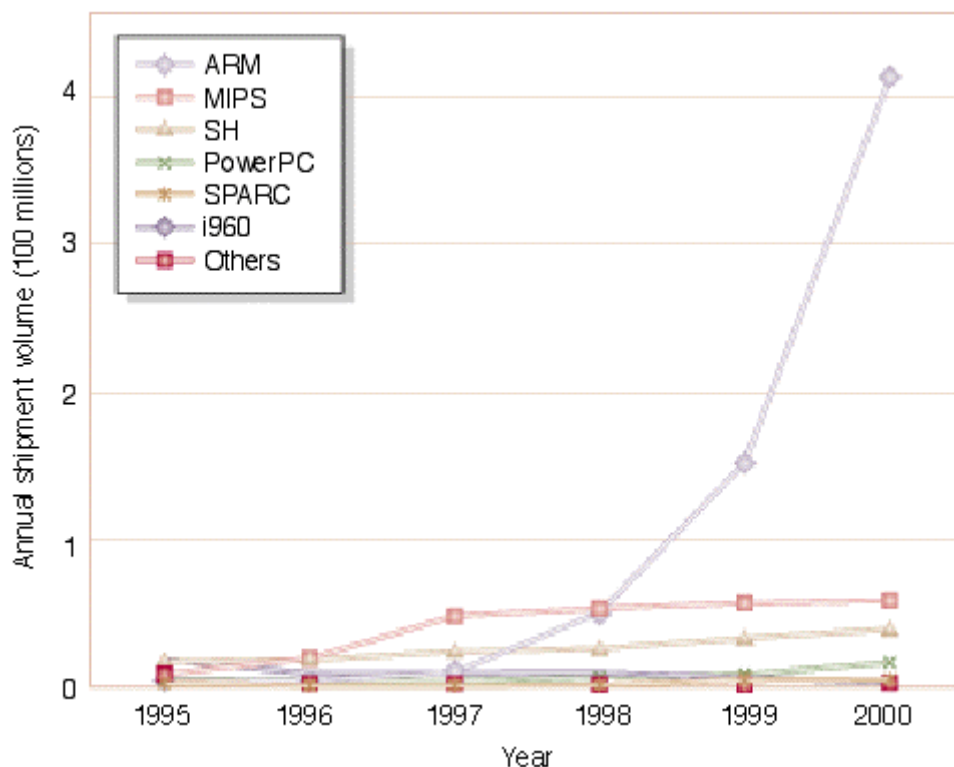


Figura 2: Evolució del número de processadors RISC distribuïts. Font: (<http://techon.nikkeibp.co.jp/>)

Part d'aquest èxit s'ha degut a que els processadors ARM tenen un baix consum i un rendiment raonable per aquest consum (MIPS/watt). Això els ha fet fermes candidats per a dispositius electrònics mòbils que requereixen certa capacitat de càlcul (com poden ser telèfons mòbils, calculadores, PDA's, reproductors de vídeo/àudio, o videoconsoles) i on és molt important aconseguir un consum d'energia el més baix possible. A més a més, en els últims anys els processadors ARM han anat incorporant noves tecnologies, com per exemple Thumb (un mode que habilita al processador executar instruccions de 16 bits, i amb el qual s'aconsegueix un codi compilat més dens)

o Jazelle DBX (que permet a un ARM executar bytecode de Java) per millorar la seva eficiència.

Els processadors ARM es poden classificar per “famílies” que agrupen diferents nuclis. En l'actualitat les famílies que més s'utilitzen són l'ARM7 (present en dispositius com la Nintendo GBA o la Nintendo DS), l'ARM9 (GP32, GP2X, les calculadores HP-49/50), l'ARM9E (telèfons mòbils, OMAP's de Texas Instruments), l'ARM10, l'ARM11 (Apple iPhone, Apple iPod touch), i la Cortex (Pandora).

Cal esmentar que ARM Limited no fabrica cap dels seus processadors, sinó que utilitza un sistema de llicències per a que les parts interessades ho puguin fer. Per aquest motiu ens podem trobar amb multitud d'empreses de semiconductors que fabriquen processadors ARM.

Menció especial mereixen, donat que es volia fer servir un per aquest projecte, els OMAP (Open Multimedia Application Platform) de Texas Instruments, que són uns processadors per a encastats basats en un ARM i pensats per aplicacions multimèdia. Alguns d'aquests processadors són heterogenis (format per més d'una arquitectura), ja que contenen dos nuclis diferents (un ARM i un DSP de Texas Instruments)

1.2.2. X86

L'arquitectura x86 també ha arribat als sistemes encastats, tot i que avui en dia en comparació a l'arquitectura ARM el seu rang d'usos ha estat més limitat, al menys en lo que es refereix a plataformes encastades portables/mòbils. Algunes de les raons podrien ser, tal i com comenta Edwards en el seu article *Migrating from x86 to PowerPC*[5]:

La família (de processadors i plataformes) compatible amb x86 és (llevat d'unes poques excepcions que podem ignorar de forma segura) utilitzat com a part d'una arquitectura hardware més o menys compatible amb un PC. Aquests processadors són grans i extremadament famolencs d'energia; la refrigeració activa (ventiladors) és gairebé un requeriment universal en dissenys de x86 (...). Un disseny basat en un x86 implica una “herència” que es requereix per mantenir la compatibilitat cap enrere (tant en el nucli de la CPU com en la circuiteria de suport de la placa base), i molta gent és conscient d'això. Hi han pocs SoC “de veritat” basats en un nucli compatible

amb x86, perquè una quantitat significativa de circuiteria sempre és necessària en un sistema x86. A més a més, a diferència d'altres dispositius encastats, els xips x86 no tenen JTAG “on-chip” o alguna altre interfície per debuggar el hardware.

Fent un breu repàs a les característiques d'alguns dels processadors o sistemes compatibles amb x86 que ens podem trobar en l'actualitat per sistemes encastats[6], podem comprovar que efectivament, fins ara no s'ha tingut com a prioritat reduir la mida que ocupen els processadors (que en alguns casos és la mateixa que els processadors x86 d'escriptori en els que estaven basats) o les plataformes que les contenen, sinó que més aviat s'ha intentat reduir al màxim el consum energètic del processador i/o fet els esforços necessaris per donar la possibilitat de refrigerar els processadors mitjançant solucions passives (només amb dissipadors). Però tot i això, sembla que “l'herència” que provoca una arquitectura x86 encara té un pes molt important en el consum energètic d'una plataforma basat en aquesta arquitectura.

Això ja ens dóna una idea de perquè avui en dia no ens trobem amb dispositius *handheld* (portables a la mà) que facin servir un processador x86 encastat, però sí que els trobem en dispositius de control industrial, quioscs interactius, ATM's, terminals punt de venda, servidors, etc.

Aquesta tendència però, sembla que pot canviar en els pròxims anys[7], ja que Intel ha fet diversos esforços per a que l'arquitectura x86 entri finalment en el mercat dels dispositius *handheld* de baix consum, actualment dominat per ARM. Un dels fruits d'aquest esforços són els processadors Intel Atom (Silverthorne) i la plataforma Intel Centrino Atom, que en un principi s'utilitzaran per UMPC o dispositius semblants, però que en el futur s'espera que es comenci a integrar en plataformes encastades com per exemple smartphones o PDA's:

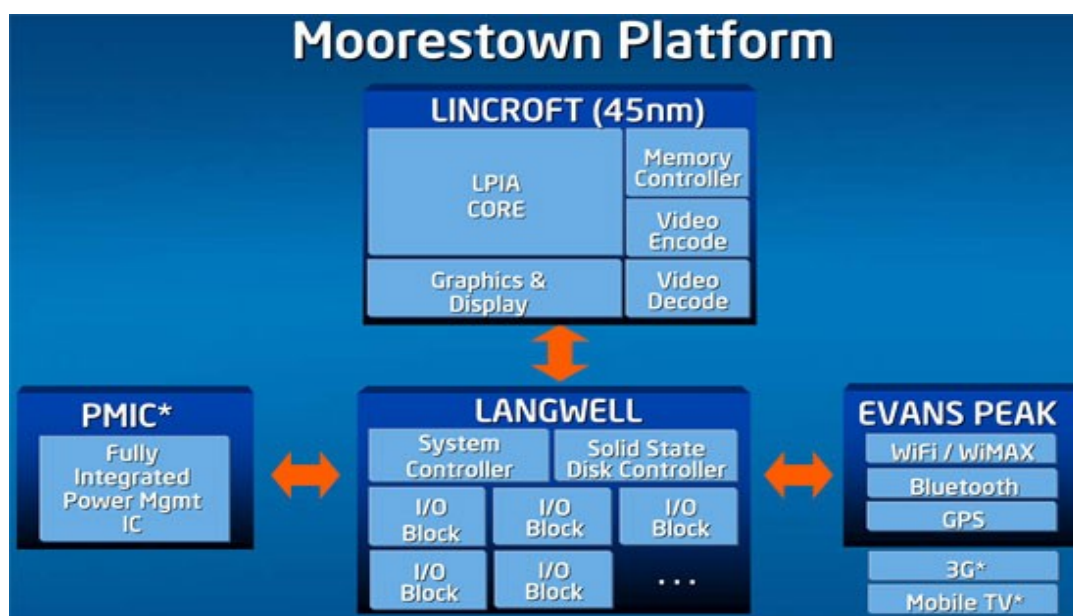


Figura 3: Lincroft serà un SoC que integrarà un processador basat en x86. (Font: www.anandtech.com)

1.3.Introducció als benchmarking

El benchmarking és un mètode que consisteix en executar un programa (o conjunt d'aquests), o realitzar altres operacions per tal d'obtenir el rendiment d'un sistema o component d'aquest. Normalment, el terme benchmark s'utilitza per aquells programes que han estat dissenyats per realitzats per fer benchmarking.

La necessitat d'aquests benchmarks ve donada per la incapacitat de conèixer el rendiment “real” d'un objecte (per exemple l'unitat de coma flotant de la CPU) fixant-nos només en les especificacions del hardware. Un clar exemple d'això el tenim amb els processadors dels sistemes de propòsit general, on fi fa uns anys se li va donar excessiva importància a la freqüència a la que funcionaven. Per altre banda, els benchmarks també ens permet comparar el rendiment amb altres màquines.

Segons una classificació proposada per Weiss[8], els benchmarks els podríem caracteritzar en tres tipus diferents:

- Els benchmarks **sintètics** són creats amb l'intenció de mesurar una o més característiques d'un processador, compilador, o sistema global.

Els sintètics poden imitar una barreja d'instruccions d'aplicacions de “món real”, o bé ser completament artificials. Per aquest motiu, aquests benchmarks no són representatius del que seria una càrrega de treball real.

Els benchmarks sintètic tendeixen a ser de mida reduïda (tot i que no sempre es així) degut a que són útils per debuggar o aïllar una funcionalitat en concret.

Exemples de benchmarks sintètics els trobem en el Dhrystone, o el Whetstone.

- Els benchmarks de “**món real**” o d'aplicació són aquells que utilitzen aplicacions del sistema operatiu o de nivell d'usuari per mesurar el rendiment a nivell de sistema.

Normalment es tracten de benchmarks que tenen requeriments grans en quant emmagatzemament de dades i codi. Comprovar el temps que triga en compilar un programa en C o la mitjana de frames per segon que obtenim en un cert temps en un videojoc es poden considerar benchmarks d'aplicació.

Els benchmarks que utilitzen per exemple un processador de text, una base de dades, o un videojoc, pertanyen a aquesta categoria.

- Per últim, els benchmarks **derivats o algorítmics** (també anomenats de *kernel*) es troben a mig camí entre els sintètics i els de món reals, ja que son desenvolupats buscant un compromís entre aquests dos tipus de benchmarks (és a dir, ni massa petits, ni massa grans). Com el seu nom indica, els benchmarks derivats “extreuen” els algoritmes principals d'una aplicació i generen conjunts de dades que podríem trobar en aplicacions del món real.

Els benchmarks derivats ens permeten evitar l'execució d'un programa sencer, i poden ser utilitzats tant per debuggar com per fer un anàlisi comparatiu entre plataformes.

En aquest grup hi trobaríem per exemple els conjunts de benchmarks desenvolupats pel consorci EEMBC o els que formen part de MiBench.

En el àmbit de les plataformes encastades, els benchmarks ens poden resultar útils principalment per dos motius:

- Ens permeten conèixer si els components compleixen els requeriments que té la plataforma per la seva aplicació.
- Gràcies als resultats que obtenim, podem establir diverses mètriques (com per exemple Preu/Rendiment, o Rendiment/Potència) i comparar-les amb altres sistemes.

Per últim, la funcionalitat d'un benchmark no només radica en trobar el rendiment d'un sistema o component en concret, sinó que també es poden utilitzar per conèixer el gast energètic (com per exemple el EEMBC's EnergyBench) o inclús la robustesa d'un sistema operatiu.

1.4. Mètriques de caracterització

A l'hora d'escollir o desenvolupar una plataforma encastada, normalment es tenen en consideració tot una sèrie de mètriques[2]:

- Com que la plataforma estarà orientada a l'aplicació o a un domini de aplicacions, s'ha de assegurar que aquesta compleix els requeriments que s'estableixen inicialment tant en el hardware com en el software.
- Suport hardware i software. Abans de decidir-nos per un entorn o altres, també és important comprovar quin és el suport que ens ofereixen, tant en el hardware (recursos, certificacions, documentació, personalització d'algun component en concret...) com en el software (entorn de programació, compatibilitat de les llibreries amb el hardware seleccionat,...).
- Enfoc de la plataforma: depenent de l'aplicació, podem trobar-nos amb un sistema reactiu (respecte d'events i comportaments no deterministes), orientat a processament de flux de dades (data flow) o un sistema mixt.
- Orientada o no a baix consum. En cas que un dels requisits de la plataforma encastada sigui que aquesta consumeixi el menys possible, llavors ens trobarem amb una restricció important. S'haurà de fer la selecció de components tenint en compte no només el rendiment que necessitem, sinó també el consum que tenen aquests, si disposen de modes d'estalvi d'energia, etc., així com intentar reduir al màxim la circuiteria addicional.

- Dimensions de la plataforma hardware. Segons els requeriments inicials, també ens podem trobar amb limitacions en la mida que ocupi la plataforma hardware.
- Portabilitat o mobilitat. En un sistema mòbil tindríem dues fites: per un costat ens interessaria que aquesta tingués el menor consum possible, i per l'altre, la mida també hauria de ser el més reduït possible.
- Segons els requeriments de la plataforma encastada que es vol desenvolupar tindrem unes necessitats de connectivitat de dades: connectivitat (port serie, port paral·lel, USB, ...), comunicacions (Ethernet, Firewire,...), visualització de dades (LVDS, VGA,...).
- Compatibilitat del hardware/software amb anteriors desenvolupaments. Si es vol evolucionar una plataforma, per minimitzar costos a vegades és interessant veure quin és el grau de compatibilitat del nou hardware/software amb desenvolupaments anteriors.
- Cost i disponibilitat del hardware. Per últim, una altra de les mètriques a tenir en compte a l'hora de seleccionar una plataforma o una altre és que també podem tenir limitacions en el cost total del hardware. A més, si es vol fabricar grans quantitats d'aquestes plataformes, també és necessari tenir garanties de que en el futur hi haurà prou disponibilitat del hardware.

1.5. Justificació de l'elecció

A l'hora de seleccionar els benchmarks que formarien part d'aquest projecte, es va tenir en compte vàries restriccions.

La primera d'elles va ser que, donat que volien fer *benchmarking* per conèixer les possibilitats multimèdia de vàries plataformes, els benchmarks seleccionats havien de realitzar alguna tasca relacionada amb aquest context i que fes servir un conjunt de dades multimèdia.

Es varen intentar evitar al màxim els benchmarks sintètics com el Dhrystone o l'SpecInt, ja que aquests no ens donarien una idea gaire precisa de les possibilitats per a aplicacions multimèdia d'una plataforma encastada. Com ja s'ha comentat anteriorment,

els benchmarks sintètics normalment avaluen parts concretes d'un sistema, o les càrregues de treball poden ser poc representatives de les que ens trobem normalment en plataformes encastades[9].

Per altra banda, com les plataformes de les que disposàvem tenien arquitectures diferents, també es va veure que era necessari disposar del codi font dels benchmarks per poder generar els executables per cada plataforma. Preferiblement el codi font havia d'estar en C, ja que això ens facilitaria el treball de portar els benchmarks a altres plataformes i arquitectures. Per aquest motiu, els benchmarks escrits en codi ensamblador van quedar descartats, ja que la seva portabilitat era molt limitada.

Un altre dels motius per els quals es volia disposar del codi font era perquè ens donaria la possibilitat de modificar (o recodificar) el codi font per adaptar-los a les nostres necessitats, com es va donar el cas.

Per últim, donat que per aquest projecte s'utilitzaria distribucions GNU/Linux, els benchmarks havien de poder executar-se en aquest sistema operatiu.

Tenint en compte totes aquestes restriccions, finalment es va fer la següent selecció de benchmarks:

1.5.1. MediaBench 1.0

MediaBench 1.0 és un conjunt d'aplicacions codificades en llenguatge d'alt nivell (C). El conjunt d'aplicacions el formen 19 programes que realitzen tasques de processats d'imatges, comunicacions i aplicacions per a DSP's, però per aquest projecte finalment només es van utilitzar algunes de les aplicacions que estan orientades a la banda de consum:

- ADPCM: aquesta aplicació realitza la conversió d'un arxiu d'àudio en format ADPCM (Adaptive Differential Pulse Code Modulation) a format PCM.
- MPEG: al conjunt d'aplicacions de MediaBench s'han inclòs els programes que realitzen la descompressió i compressió de vídeo en format MPEG2. En el nostre cas només utilitzarem el que realitza la descompressió de vídeo.

1.5.2. Unixbench 4.1.0

Aquest conjunt de benchmarks semblava en principi que era el més complert de tots: una part dels benchmarks tenia un enfoc més computacional (Dhrystone, Whetstone, etc), mentre que la resta es centrava en diversos aspectes del sistema operatiu.

Per desgràcia, a mesura que es va desenvolupar aquest projecte, es va comprovar que els benchmarks més computacionals es van haver de descartar per diversos motius: per una banda eren massa sensibles a “optimitzacions”[10] del compilador; i per l'altre, també es va comprovar que els benchmarks eren massa petits i hi havia la possibilitat de que capiguessin completament en la cache del processador (amb lo que a vegades no fariem benchmarking del sistema memòria principal-processador).

Per aquest motiu, es va decidir que Unixbench només s'utilitzaria per fer benchmarking d'alguns aspectes del sistema operatiu: és a dir, únicament es van considerar els resultats dels benchmarks que pertanyen a la categoria System (més detalls al Annex). Malauradament, experimentalment (consultar Figura 12) es va poder comprovar que aquests benchmarks són depenents de la CPU on s'executin, i que per tant no només estem fent benchmarking del sistema operatiu (el que es pretenia amb Unixbench) sinó també de la CPU. Per aquest motiu, finalment no s'han inclòs els resultats obtinguts d'Unixbench.

1.5.3. MiBench 1.0

Per últim, MiBench és un conjunt de benchmarks que es presenta com l'alternativa lliure dels benchmarks EEMBC. Al igual que aquests, MiBench intenta ser uns benchmarks que siguin representatius de les tasques que pot realitzar un sistema encastat, i que es troben dividits en diferents categories segons l'àmbit que es vulgui fer benchmarking (automoció i control industrial, consumidor, ofimàtica, xarxa, seguretat, i telecomunicacions).

Per el cas que ens ocupa, la categoria que més ens interessa és la de consumidor, on s'agrupen els benchmarks amb un enfoc més multimèdia:

- Compresió i descompresió de fitxers JPEG

- Descompressió de fitxers MP3 a format WAV.
- Operacions amb imatges TIFF (de color a b/n, de tiff a bitmap, etc).
- Renderizat de text.

Dintre de MiBench podem trobar dos conjunts de dades de major o menor mida. Per la realització d'aquest projecte només es va fer servir el conjunt de dades grans, ja que són més representatives de les que càrregues de treball d'un sistema encastat actual.

Capítol 2. Entorn de Desenvolupament i Metodologia de Treball

2.1. Plataformes seleccionades

A l'hora de seleccionar les plataformes per realitzar aquest projecte es va intentar que aquestes fossin més o menys representatives del que ens podem trobar avui en dia en una plataforma encastada orientada a consum. En un principi no es va tenir en compte si aquestes eren portables o mòbils.

Donat que sembla que s'aproxima una “batalla” entre ARM i x86 pel mercat de sistemes encastats orientats a consum (UMPC's, MID, PDA's, smartphones, etc) , vaig trobar interessant centrar-me únicament en aquestes dues arquitectures. Per desgràcia, no es va poder comptar amb plataformes que tinguessin alguns dels últims oferiments tant de x86 (Intel Atom) com d'ARM, però sí amb algunes que es poden adquirir en l'actualitat.

Tenint en compte tots aquests criteris, la selecció inicial va quedar de la següent manera:

- Un SBC d'Albatron en format mini-ITX i que té un processador Intel Celeron M a 1.200 Mhz (512 KiB L2, FSB a 400 Mhz), 512 MiB DDR266 de RAM, i una Compact Flash industrial de Sandisk de 2 GiB. El sistema operatiu d'aquesta plataforma era una Debian 4.0r3.
- Un SBC d'Aaeon amb un processador Intel Ultra Low-Voltage Celeron a 400 Mhz, 512 MiB DDR266 de RAM i una CF de Sandisk de 2 GiB. El sistema operatiu que es va fer servir també era una Debian 4.0r3.
- Una PDA Compaq iPAQ 3630. Aquest dispositiu té un processador Intel StrongARM SA1110 a 206 Mhz, disposa de 32 MiB de RAM i 16 MiB de ROM, i també es va fer servir una Compact Flash de Sandisk de 2 GiB. En

aquest cas el sistema operatiu GNU/Linux està basat en OpenEmbedded (GPE).

- La plataforma OMAP 5912 de Texas Instruments, que conté un processador ARM9 i un DSP TMS320C55xx que funciona a 192 Mhz. Entre les característiques de la placa de prototipatge trobem que disposa de 32 MiB de RAM i 32 MiB de Flash ROM.

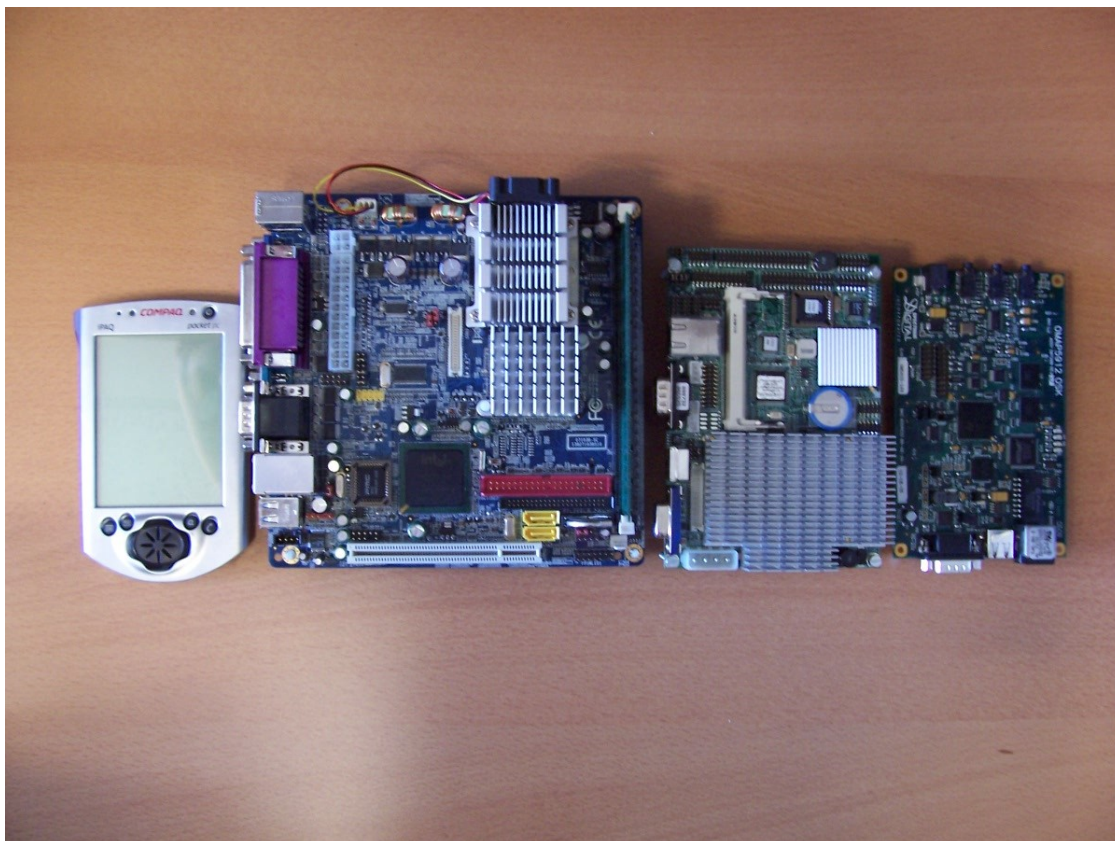


Figura 4: D'esquerra a dreta: Compaq iPAQ, SBC d'Albatron, SBC d'Aeon, OMAP 5912.

Com es pot comprovar a la figura 4, les plataformes x86 que s'han seleccionat s'han dissenyat tenint en compte un requeriments específics en quan connectivitat (ports VGA, sèrie, Ethernet, etc.) i on s'ha evitat l'ús d'elements de dissipació activa (ventiladors). La plataforma basada en OMAP és la que normalment s'utilitza per a prototipatge, mentre que l'ARM ja forma part d'un producte comercial.

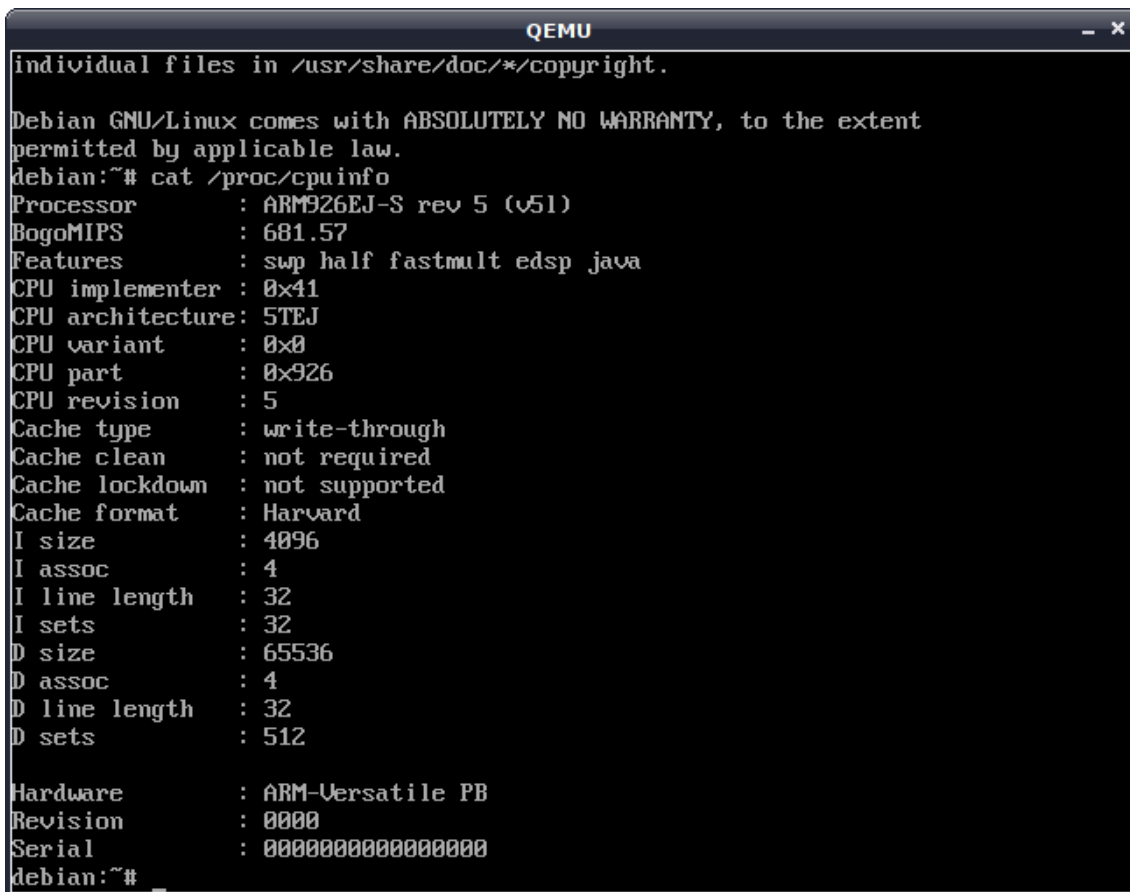
Per altra banda, també es varen fer servir dos PC's estàndard. Un d'ells es va fer servir per comparar les seves prestacions amb els de les plataformes encastades, mentre que l'altre es va fer per verificar el rendiment del emulador QEMU:

- El PC de referència disposava d'un processador Intel Core 2 Duo E6750 (2.66 Ghz, 64 KiB L1, 4 MiB L2, i FSB a 1333 Mhz), 2 GiB DDR2 a 800 Mhz de RAM, i un disc dur SATA300 de Western Digital de 300 GiB. El sistema operatiu on es va realitzar el benchmarking era una Ubuntu 8.04.
- El PC que es va fer servir per comparar el resultats amb QEMU disposava d'un processador AMD Athlon XP 2400+ (1.8 Ghz, 128 KiB L1, 256 KiB L2, FSB a 400 Mhz), 512 MiB DDR266 de RAM, i un disc dur PATA100 de Western Digital de 80 GiB. El sistema operatiu de la màquina era una Ubuntu 8.04.

2.2.Eines i entorns utilitzats

En un principi, per generar els executables per a cada arquitectura (x86, ARM) es va decidir utilitzar una màquina x86 amb un sistema operatiu GNU/Linux i els compiladors GCC necessaris per a les dues arquitectures (un seria natiu per a x86 i l'altre un compilador creuat per a ARM). No es va fer servir cap IDE a l'hora d'adaptar els benchmarks, ja que les parts que s'havien de modificar estaven força localitzades en fitxers concrets.

Per altra banda, també vaig trobar interessant crear un entorn basat en QEMU emulant un processador ARM, donat que Google està fent servir en el seu SDK d'Android (que és una plataforma software i sistema operatiu per a dispositius mòbils) un entorn semblant per fer les simulacions de les aplicacions. Amb aquest entorn es volia verificar si es pot utilitzar QEMU com si es tractés d'una plataforma ARM “real”, és a dir, que els resultats obtinguts no són pas dependents de la màquina hoste on s'executa.



```

QEMU
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
debian:~# cat /proc/cpuinfo
Processor       : ARM926EJ-S rev 5 (v5l)
BogoMIPS       : 681.57
Features        : swp half fastmult edsp java
CPU implementer : 0x41
CPU architecture: 5TEJ
CPU variant     : 0x0
CPU part       : 0x926
CPU revision    : 5
Cache type      : write-through
Cache clean     : not required
Cache lockdown  : not supported
Cache format    : Harvard
I size         : 4096
I assoc        : 4
I line length   : 32
I sets         : 32
D size         : 65536
D assoc        : 4
D line length   : 32
D sets         : 512

Hardware       : ARM-Versatile PB
Revision      : 0000
Serial        : 0000000000000000
debian:~#

```

Figura 5: Emulador ARM en QEMU

El processador que s'emula sota QEMU és un ARM926EJ-S, un processador força habitual en telèfons mòbils. Al igual que les plataformes x86 seleccionades per aquest projecte, l'entorn que es va muntar feia servir com a sistema operatiu una Debian 4.0r3.

2.3. Adaptació de components / mòduls

Arribat a aquest punt, vaig trobar-me amb un “problema” a l'hora de treballar amb la plataforma OMAP. El DSP que conté l'OMAP 5912 no es feia servir en cap moment, degut a que les aplicacions per a fer benchmarking seleccionades estaven programades de tal manera que només feien servir els recursos hardware disponibles del processador.

Això requeria per tant una recodificació complerta de totes les aplicacions que composaven els benchmarks, identificant primer aquelles parts que s'havien de processar en el DSP (realitzant perfilats de cadascuna de les aplicacions), afegint parts de codi per fer la comunicació entre l'ARM i el DSP (tenint cura de que el model de

transferència de dades que s'implementes no tingués un efecte negatiu al rendiment de l'aplicació). En definitiva, que incloure l'OMAP 5912 en els benchmarkings suposava tenir benchmarks que no eren idèntics, i per tant, difícilment comparables.

Per aquest motiu es va decidir excloure l'OMAP de les plataformes seleccionades, quedant únicament les x86 i l'ARM.

Per aquestes arquitectures es va haver d'adaptar les aplicacions per tenir una mesura del rendiment de les plataformes, ja que llevat d'Unixbench, la resta de aplicacions no disposava de cap indicador que el poguéssim fer servir com a benchmarks.

Les solucions que es podien adoptar podien ser dos: o incloure la funció de C `gettimeofday()`, o bé incloure la funció `clock()` al inici i al final de cadascuna de les aplicacions.

En el primer cas, el que es fa és implementar un “*wall clock time*”, que ens permet obtenir el temps real que cal per completar una tasca, incloent-hi els temps que poden passar per endarreriments programats (`sleep()`) o deguts a que s'espera a que estigui disponible un recurs determinat. En altres paraules, el “*wall clock time*” és un temps en que es suma el temps que necessita la CPU, el temps per a E/S, i temps provocats pels canals de comunicació. La funció `gettimeofday()` té una precisió de microsegons.

Per altra banda, amb la funció `clock()` podem conèixer quin és el temps de CPU que cal per realitzar una tasca. El problema que em vaig trobar amb aquesta implementació és que les finestres de temps que s'utilitzen normalment per comptar el número de clocks era massa gran per els processadors i les aplicacions/dades que es feien servir, per lo que en alguns casos el resultat era simplement 0. Per aquest motiu es va descartar aquesta implementació, ja que en principi semblava que n'hi havia prou tenint el “*wall clock time*”, però si s'hagués volgut conèixer el temps de CPU, llavors s'hauria d'haver implementat un “*high resolution clock*”.

Capítol 3. Test i Resultats

3.1. Entorn de test

Per a realitzar els tests es varen fer servir els conjunts de dades que es proporcionaven amb els benchmarks. En el cas de MiBench es varen utilitzar únicament el conjunt de dades gran, ja que com s'ha comentat anteriorment és el que més s'aproxima als requeriments que actualment se li pot demanar a un sistema encastat multimèdia:

- Conjunt de dades de MiBench:
 1. Compresió JPEG: Imatge en format PPM de 512x512 pixels.
 2. Descompresió JPEG: Imatge en format JPEG de 512x512 pixels.
 3. MAD: Arxiu MP3 monocanal amb un bitrate de 96 Kib, i freqüència de mostreig de 44.100 Khz.
 4. TIFF2BW: Arxiu TIFF en color de 3069x3100 pixels.
 5. TIFF2RGBA: Arxiu TIFF en color de 3069x3100 pixels.
 6. TIFFDITHER: Arxiu TIFF en color de 3069x3100 pixels.
 7. TIFFMEDIAN: Arxiu TIFF en blanc i negre de 3069x3100 pixels.
 8. TYPESET: Arxiu de text de 8.0 KiB.
- Conjunt de dades de MediaBench:
 1. ADPCM: arxiu de so en format ADPCM
 2. MPEG2DEC: Arxiu MPEG de 720x576, framerate de 25 fps, i 1 segon de durada.

3.2. Tests i resultats

3.2.1. Benchmarks multimèdia

Tots els benchmarks de MiBench i MediaBench es varen executar tres vegades en cadascuna de les plataformes i es va fer la mitjana dels temps obtinguts per tal de tenir una mesura més o menys fiable (s'ha de recordar que s'està fent servir un “*wall clock time*” i que si en algun moment donat es produeix un temps d'espera pel motiu que sigui, també l'estarem mesurant).

Els resultats obtinguts en les plataformes encastades i el PC de referència varen ser els que mostren les següents figures:

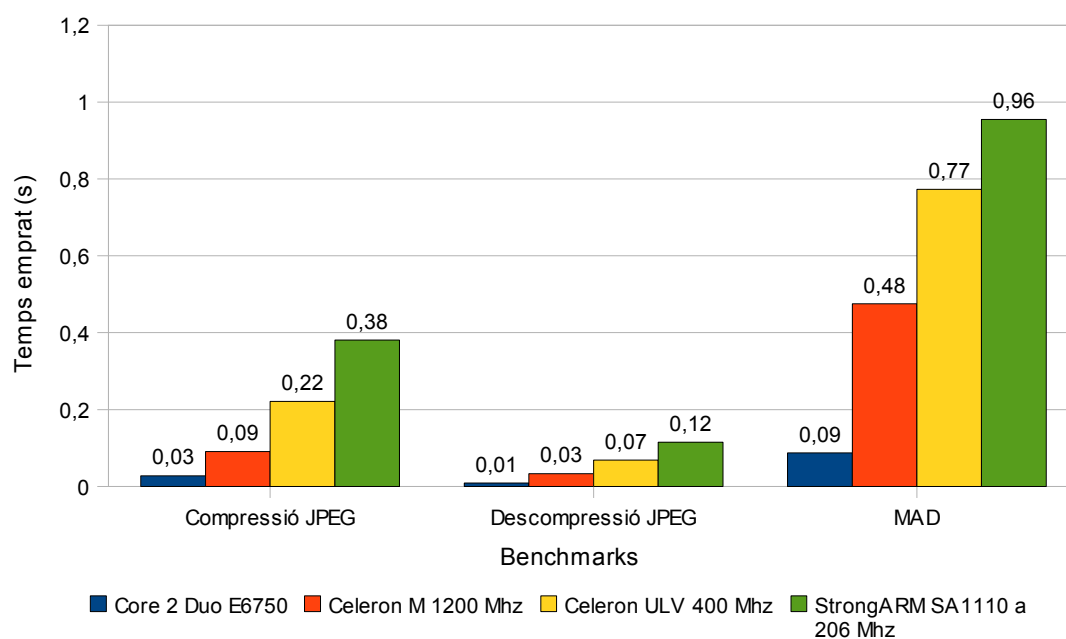


Figura 6: MiBench: compressió i descompressió JPEG, i descompressió MP3.

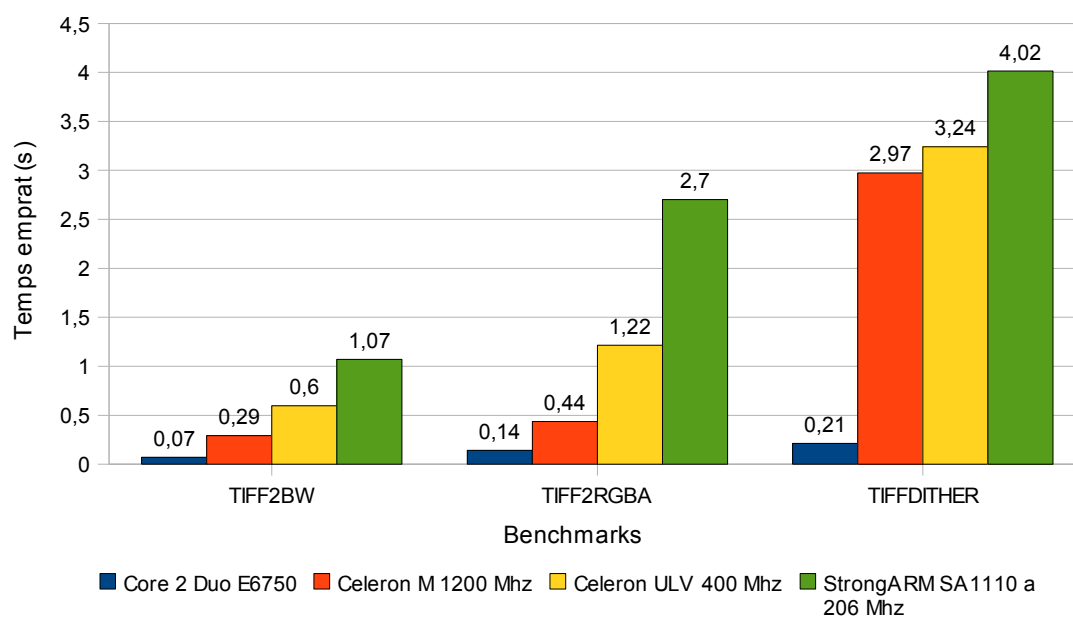


Figura 7: Operacions amb imatges TIFF.

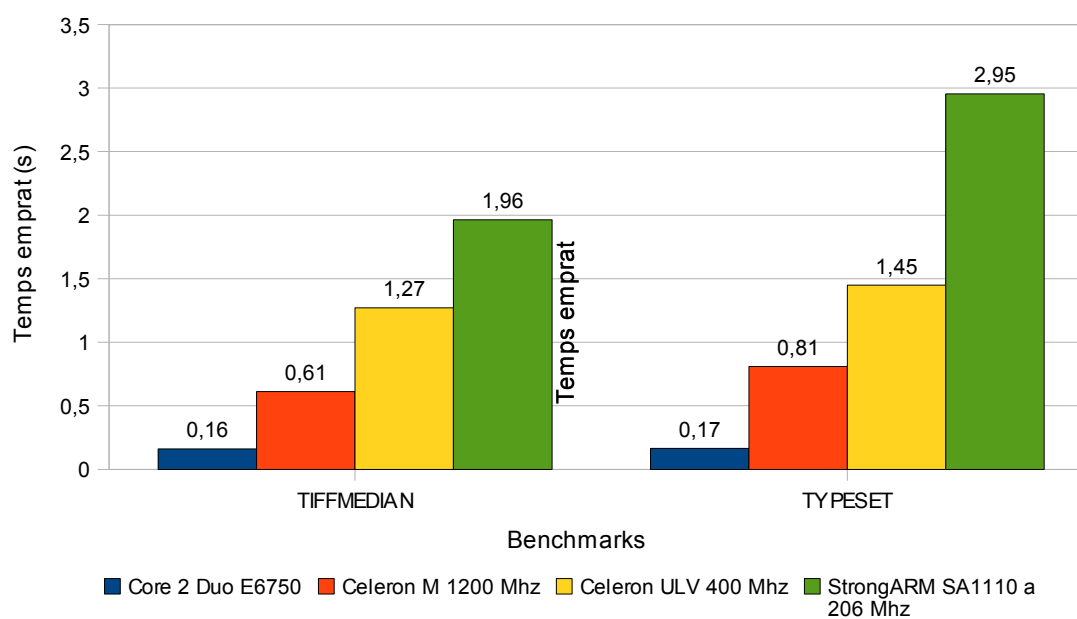


Figura 8: Operacions amb imatges TIFF i tractament de text.

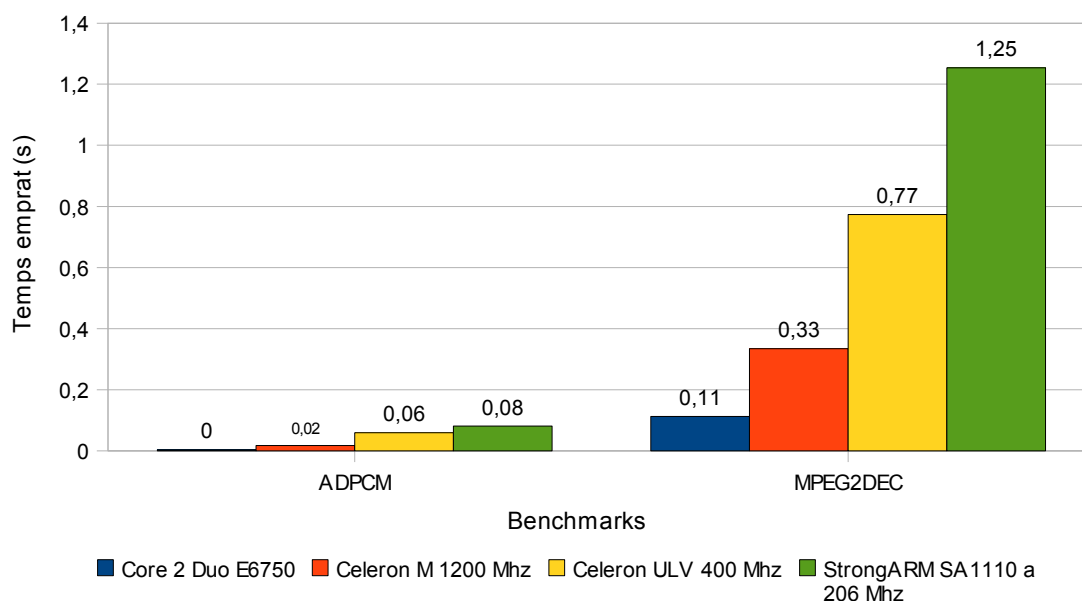


Figura 9: Conversió d'ADPCM a PCM i descompressió de fitxers MPEG.

Arribats a aquest punt, caldria preguntar-nos si les prestacions que ens ofereixen les plataformes compleixen els requisits de QoS (Quality of Service) fixats en els requeriments.

Per exemple, podem veure que l'StrongARM a 206 Mhz triga 1.25 segons en descomprimir l'arxiu de vídeo MPEG, per la qual cosa en principi hauria de quedar descartat per qualsevol aplicació que requereix descompressió de vídeo en temps real. Amb la mateixa plataforma hem obtingut un temps força alts al fer les operacions amb imatges TIFF, però en aquest cas podria no descartar-se la plataforma si dintre del QoS es consideressin aquests temps com acceptables.

Per altra banda, queda força clar amb aquests resultats que el Core 2 Duo és la plataforma on s'obtenen els millors resultats en tots els benchmarkings, però s'ha de tenir en compte que en un sistema encastat també són importants altres factors com ho poden ser el consum energètic, el preu de la plataforma, la mida o la seva portabilitat.

Per aquest motiu, no podem afirmar “a la lleugera” que una plataforma és millor que l'altre tenint només en compte els resultats obtinguts, si no que s'ha de fer un barem que tingui en compte els factors mencionats anteriorment.

3.2.1. Consums

Llevat del SBC basat en un Celeron ULV a 400 Mhz, malauradament per les altres plataformes no s'ha disposat de les eines necessàries per mesurar aquest consum energètic amb precisió. Per aquest motiu s'han fet estimacions del consum a partir de configuracions semblants (Core 2 Duo E6750[11], SBC Celeron a 1200 Mhz [12]), o bé s'ha calculat el consum a partir de la capacitat i voltatge de la bateria i els temps que es necessita per descarregar-la completament (Compaq iPaq 3630).

La següent taula mostra els consums de les diferents plataformes. S'ha diferenciat els consums en dues columnes. El consum màxim indica el consum de les plataformes quan se'ls sotmet a una alta càrrega de treball (full charge), com per exemple succeïx al executar els benchmarks. Per altra banda, el consum típic ens indica el consum del sistema quan aquest es troba en repòs (idle).

Plataforma	Consum màxim	Consum típic
PC Core 2 Duo E6750	171 W	144 W
SBC Celeron M 1200 Mhz	24 W	16 W
SBC Celeron ULV 400 Mhz	13 W	9.45 W
PDA Compaq iPAQ 3630 *	0,7 W	0.26 W

Taula 1: Consums màxims i típics de les plataformes

* Estimació feta a partir d'una bateria de 970mAh i 3.7 V que triga a descarregar-se 5 o 14 h.

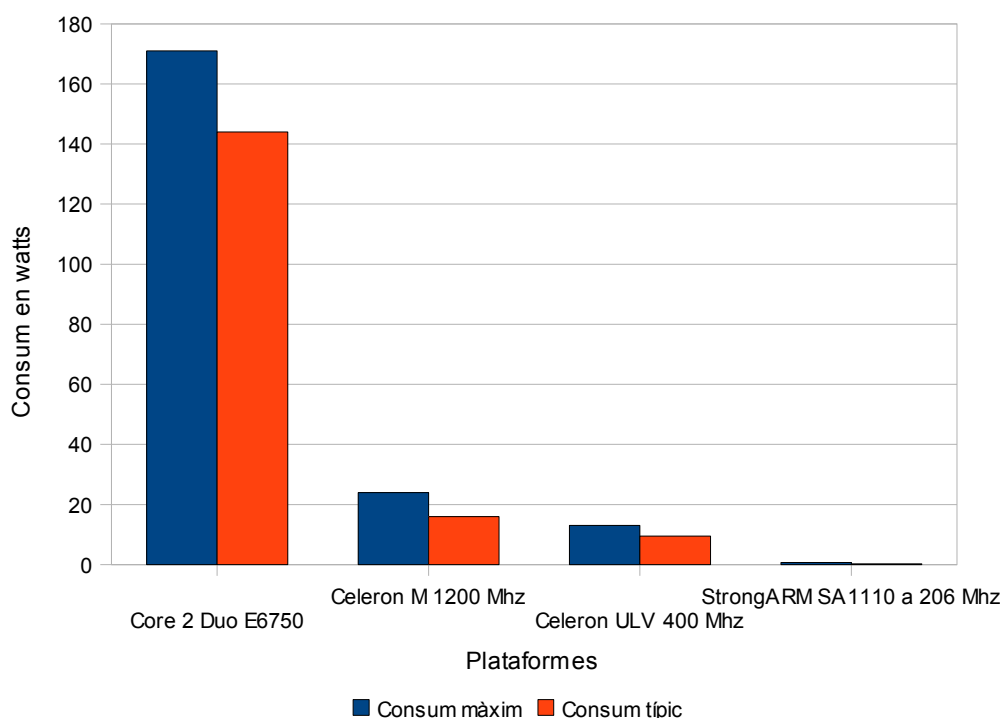


Figura 10: Consum de les plataformes

Com es pot veure a la gràfica anterior, el consum d'un Core 2 Duo E6750 es troba molt per sobre de qualsevol plataforma encastada que s'ha fet servir en aquest projecte. Però el Core 2 Duo també ha sigut la plataforma més ràpida amb diferència. Per aquesta raó resulta interessant veure la relació que hi ha entre el rendiment i el consum de les plataformes. Una de les mesures que es fa servir en aquests casos és normalment la de MIPS/watt, però donat que tenim dues arquitectures de processador diferents (i que a més a més són totalment contraries en quant a concepte, ja que l'ARM és RISC i mentre que la x86 és CISC) no podem fer servir aquesta mesura, ja que la teoria ens diu que en el cas dels processadors RISC es faran més instruccions per realitzar una mateixa tasca.

En el següent diagrama de Pareto, en el que s'ha utilitzat el temps emprat per a realitzar la tasca TIFFDITHER i el consum màxim de cadascuna de les plataformes, es pot veure aquesta relació:

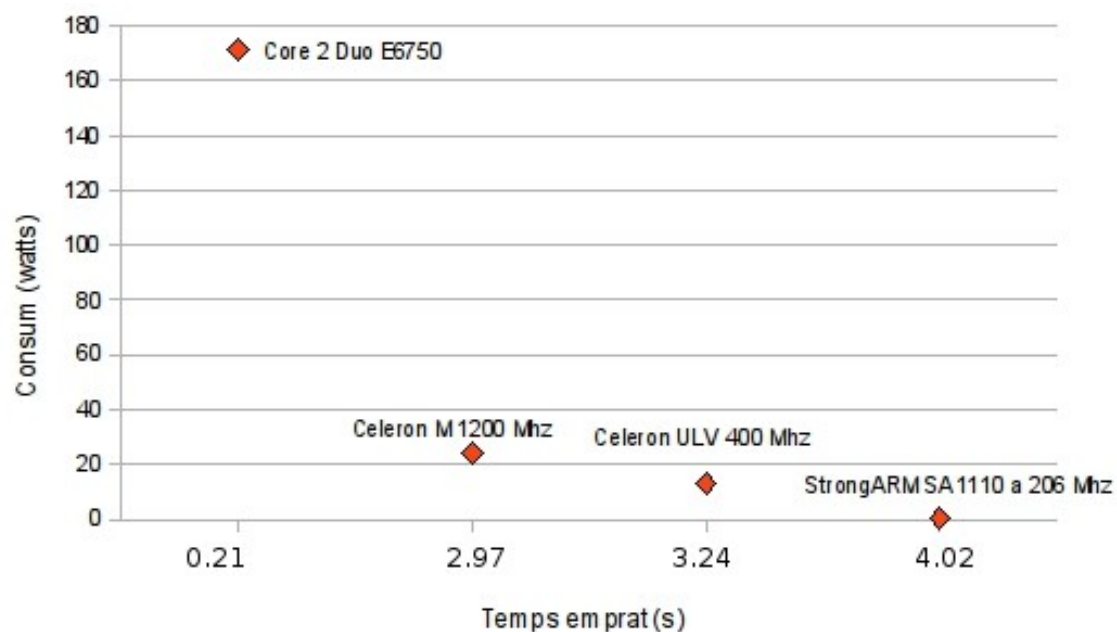


Figura 11: Relació rendiment/consum de les plataformes

Tenint en compte aquests resultats, podem veure que segons les condicions que tinguem, una sol·lució o una altre serà més adient. Quan el temps d'execució no està limitat, una solució basada en ARM es idònea ja que consumeix menys i el baix rendiment no és cap *handicap*. En l'altre extrem, quan el temps és una restricció forta i el consum no és crític, la sol·lució basada en el Core 2 Duo assolirà la tasca molt ràpidament ja que és la que ofereix millors prestacions, però amb un consum elevat. Entre aquests dos punts de treball, els dos SBC basats en un Celeron proporcionen solucions intermitges en el ratio consum/velocitat.

3.2.2. Preu

La llista de preus de les plataformes que s'han fet servir per realitzar *benchmarks* és la que es pot veure a la taula que es mostra a continuació:

Plataforma	Preu
PC Core 2 Duo E6750	600 €
SBC Celeron M 1200 Mhz	350 €
PDA Compaq iPAQ 3630	(descatalogat) 150 €

Taula 2: preus de les plataformes.

No s'ha pogut incloure el preu del SBC amb un Celeron ULV a 400 Mhz, ja que aquest tipus de producte es sol vendre només en grans volums i és necessari contactar amb comercials per a poder obtenir un preu. Per altra banda, per ser una mica equitatiu amb la PDA, en el cas del PC i el SBC també s'hauria d'afegir al preu el cost d'un monitor i els perifèrics d'entrada de dades.

Fixant-nos en els preus, es pot comprovar l'elevat cost que té el SBC amb un Celeron a 1200 Mhz, que costa més de la meitat que un Core 2 Duo E6750, sobre tot tenint en compte que el seu rendiment no és ni molt menys la meitat que el del C2D.

3.2.3. Mida

La mida també és un factor a tenir en compte si parlem de aplicacions encastades portàtils. En la següent taula es pot veure quins eren le mides de les plataformes que es varen fer servir. Per mesurar l'alçada també s'ha tingut en compte l'espai ocupat pels dissipadors i el ventilador que pot portar aquest:

Plataforma	Amplada	Llargada	Alçada
PC Core 2 Duo E6750	305 mm	244 mm	700 mm
SBC Celeron M 1200 Mhz	170 mm	170 mm	350 mm
SBC Celeron ULV 400 Mhz	100 mm	140 mm	200 mm
PDA Compaq iPAQ 3630	75 mm	130 mm	150 mm

Taula 3: Mida de les plataformes

Tant el PC basat en un Core 2 Duo com el SBC amb un Celeron M a 1200 Mhz no es poden considerar com candidats per aplicacions encastades portàtils, ja que a més de tenir una mida considerablement gran, també s'ha de comptar amb que tenen altres necessitats com fonts d'alimentació externa.

El SBC amb un Celeron ULV a 400 Mhz és una mica més dubtós, ja que en aquest cas la mida és bastant més reduïda, es pot alimentar amb un simple connector molex, i té un bus LVDS intern per connectar-hi un LCD. En aquest cas potser si que es podria fer servir com una plataforma encastada portable.

3.3. Altres resultats

En aquest apartat s'ha volgut incloure tots aquells resultats que finalment han quedat exclosos perquè no s'han considerat com a vàlids.

3.3.1. Unixbench

Teòricament, els benchmarks d'Unixbench que es van seleccionar finalment ens servirien per poder mesurar el rendiment de diferents aspectes d'un sistema operatiu GNU/Linux o el sistema de fitxers utilitzat.

Això ens seria útil per exemple en el cas que no poguéssim utilitzar la mateixa distribució GNU/Linux o sistema de fitxers en totes les plataformes (com va ser el cas

en aquest projecte), i així poder comparar quin és l'impacte del sistema operatiu o del sistema de fitxers en el rendiment.

Per desgràcia, es va comprovar que tots aquests benchmarks eren depenents de CPU. Per exemple, les aplicacions que realitzen el benchmarking del sistema de fitxers, en dues plataformes de característiques molt similars en quant a busos i mida de memòria (els dos SBC), el rendiment era més gran quan més capacitat de processament tenia la CPU. Tenint en compte que per les proves es va utilitzar la mateixa Compact Flash amb la mateixa distribució GNU/Linux (una Debian 4.0r3), amb els resultats obtinguts semblava clar que la CPU de la plataforma també afectava els resultats:

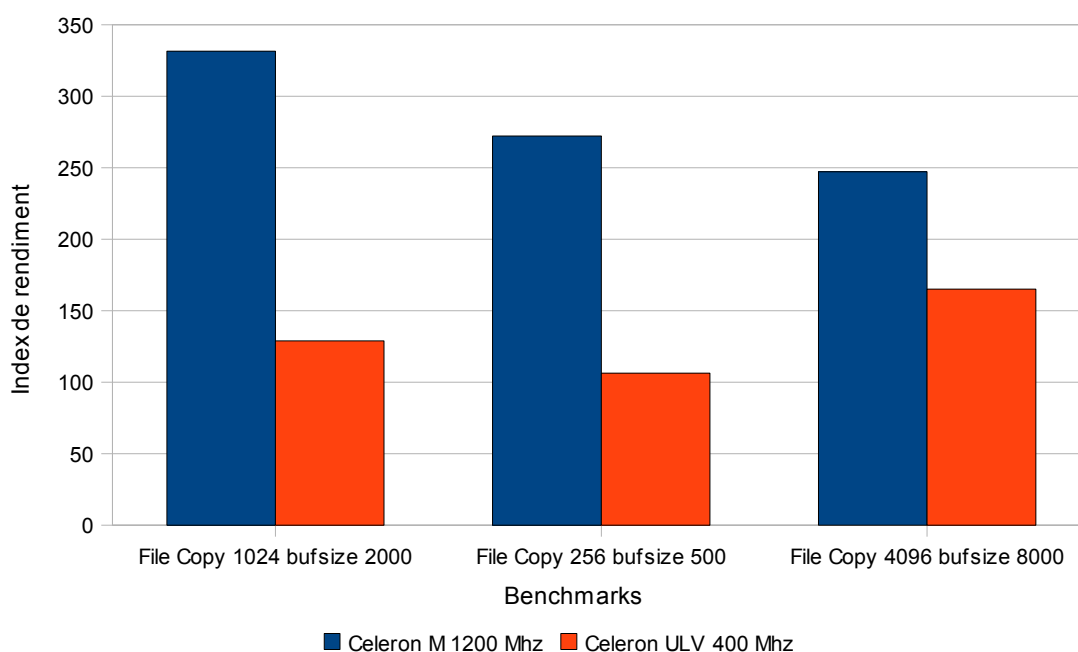


Figura 12: Resultats del benchmarking al sistema de fitxers.

Donat aquests resultats, es va decidir no tenir en compte els resultats d'Unixbench, ja que no es servien per mesurar l'eficiència d'un sistema operatiu.

3.3.2. QEMU

Uns altres resultats que també es van descartar van ser els obtinguts en l'entorn QEMU que simulava una arquitectura ARM.

En aquest entorn es van executar la mateixa bateria de benchmarks que en les plataformes encastades. El primer objectiu era comprovar si els resultats eren depenents de la màquina hoste on s'executava l'entorn QEMU, i per aquesta raó es va utilitzar dos PC's diferents. Els resultats que es van obtenir són els que es mostren en les següents figures:

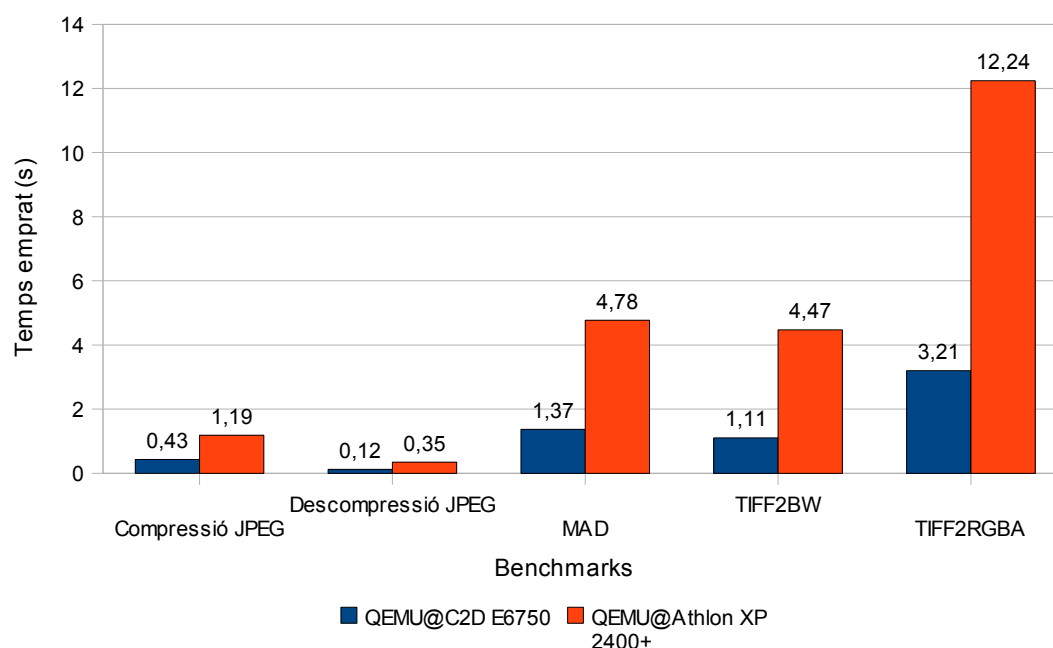


Figura 13: Resultats MiBench amb QEMU

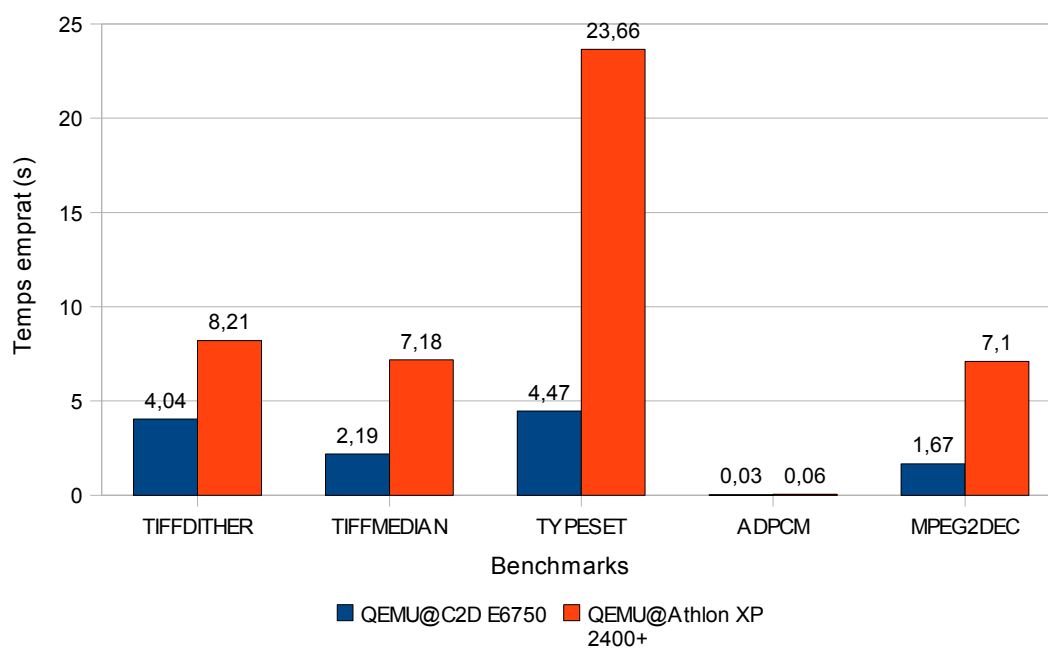


Figura 14: Resultats MiBench i MediaBench amb QEMU

Com es pot observar clarament, els resultats obtinguts són clarament depenents de la màquina on s'està executant QEMU. La conclusió és doncs, que no podem fer servir QEMU per obtenir mesures reals del rendiment d'un ARM, ja que tots els resultats són depenents de la màquina hoste on s'estigui executant QEMU.

Capítol 4. Conclusions

4.1. Conclusions

Les conclusions que he pogut extreure després d'haver realitzat aquest projecte són varies.

Des d'un punt de vista teòric, per un costat m'he pogut adonar de la dificultat de trobar benchmarks adequats per a plataformes encastades degut a que són d'aplicació específica i perquè el seu domini d'aplicació és molt ampli.

En el pitjor dels casos, crec que es podria optar per utilitzar benchmarks “genèrics”, com per exemple SpecInt. Amb això podríem obtenir diversos resultats d'un sistema encastat, però tenint en compte que no són benchmarks fets per plataformes encastades ni tampoc representen càrregues de treball típiques d'una aplicació encastada, no ens donaran una visió real de les possibilitats de la plataforma encastada per l'aplicació que hi volem posar.

Un altra solució seria fer servir els benchmarks del consorci EEMBC. L'avantatge en aquest cas el trobaríem en que els algorismes utilitzats i les càrregues de treball d'aquests benchmarks són bastant semblants als que podem trobar normalment en aplicacions d'un sistema encastat, i per tant tindríem una idea més real del rendiment d'aquest. L'únic problema que ens podem trobar és que no existeixi cap conjunt de benchmarks d'EEMBC que sigui més o menys semblant a l'aplicació en quant a algorismes o càrregues de treball, ja que en l'actualitat els conjunts de benchmarks estan limitats a set mercats (automoció, consumidor, multimèdia, JAVA, xarxes, tasques ofimàtiques i telecomunicacions).

Per això crec que la millor solució seria fer servir com a benchmark l'aplicació final o algorismes d'aquest. Aquesta solució ens donaria una visió més acertada del rendiment de la nostra plataforma encastada, però el problema que ens podem trobar en aquest cas és que simplement no disposen de l'aplicació en el moment en que es vulguin avaluar diferents plataformes, o que s'hagi de fer adaptacions per cadascuna de les plataformes.

En quant les conclusions que he pogut extreure de la part tècnica, durant la realització d'aquest projecte m'he adonat de la dificultat de comparar directament plataformes homogènies i heterogonies, ja que aquestes últimes requereixen una recodificació del software, i per tant ens trobem amb el problema de que no estariem comparant el mateix.

També he pogut comprovar que el rendiment que obtenim amb QEMU és dependent de la màquina on s'estigui executant. Per aquest motiu, s'ha de considerar que qualsevol rendiment que obtenim de QEMU no serà el que obtindrem en una plataforma basada en un ARM. En altres paraules, hem de considerar a QEMU com a un simulador d'una arquitectura ARM, i no pas com un emulador.

D'altra banda, s'hauria de descartar qualsevol ús dels benchmarks d'Unixbench, ja que s'ha comprovat que avui en dia té masses mancances com per utilitzar-los per avaluar qualsevol tipus de plataforma.

Respecte les plataformes estudiades, hem vist que segons el temps disponible per realitzar la tasca una solució o una altre serà òptima respecte el consum emprat i en aquest cas, la tria del dispositiu final vindrà determinat per les condicions de treball habituals. En el cas d'escenaris de treball dinàmics, una plataforma que integrés més d'un dispositiu seria molt útil. Aquesta plataforma versàtil amb diversos dispositius (per exemple una plataforma que disposés d'un Core 2 Duo i un ARM) hauria de ser capaç de reconfigurar-se dinàmicament (és a dir, que fos capaç d'apagar o encendre els dispositius, distribuir les càrregues de treball de forma intel·ligent, etc) segons la càrrega de treball, o les limitacions del moment.

Per acabar, sobre els resultats obtinguts en la resta de benchmarks, crec que és important tenir una visió general de la problemàtica que ens trobem en el cas dels sistemes encastats. Es a dir, no en tenim prou en dir que la plataforma més ràpida és sempre la millor, sinó que també hem de considerar altres factors com el QoS, la relació rendiment/consum, el conjunt de dades a processar i el temps del que es disposa, etc, i plantejar-nos quines solucions hi podem aportar.

4.2. Experiència personal i professional

Aquest projecte m'ha permès treballar amb un dels meus interessos principals en el camp de l'informàtica (el hardware) des de una perspectiva diferent a la que havia fet fins ara, ja que amb els encastats m'he trobat moltes més limitacions a l'hora de treballar amb elles que no pas amb els sistemes de propòsit general.

Relacionat amb el punt anterior, gràcies a aquest projecte també he pogut veure part de la metodologia de treball que es segueix quan es tracten de sistemes encastats, i les diferències que hi ha amb els de propòsit general.

Per últim, comentar que professionalment m'agradaria treballar en el futur amb aquests sistemes encastats, especialment amb els que fan servir processadors heterogenis.

4.3. Evolució futura

Personalment crec que amb aquest projecte s'han deixat varies possibilitats obertes de cara un futur.

Per exemple, hauria estat interessant incloure l'OMAP del que es disposava, ja que és un processador força comú en sistemes encastats que tenen una aplicació multimèdia. Per desgràcia, per aprofitar el DSP d'aquest processador hagués estat necessari realitzar un recodificació total dels benchmarks utilitzats, tasca que quedava fora de l'àmbit d'aquest projecte.

Per altra banda també m'hauria agradat ampliar el ventall de resultats amb els benchmarks creats per el consorci EEMBC, especialment amb el conjunt de benchmarks DENBench que és el que està enfocat a l'àmbit multimèdia, ja que crec que s'hagués pogut comprovar millor la idoneïtat de les plataformes seleccionades en aplicacions multimèdia, donat que el conjunt de benchmarks és molt més gran que el que s'ha inclòs finalment per aquest projecte, i amb algorismes més complets. A més a més, els benchmarks de EEMBC també haurien donat la possibilitat de comprovar si les plataformes seleccionades per el projecte es podrien fer servir en altres àmbits totalment diferents, com per exemple l'automoció, telecomunicacions, en tasques ofimàtiques, com a plataforma per executar Java 2 Micro Edition, etc. El problema amb els

benchmarks d'EEMBC és que no són de domini públic, sinó que tenen un cost bastant elevat, i això els va descartar des d'un començament. A més, amb els benchmarks de DENBench crec

Relacionat amb l'entorn amb que s'ha hagut de treballar, s'ha trobat a faltar alguna eina que ens permetés mesurar el número d'instruccions que s'executen, així com poder conèixer els cicles de rellotge. Potser aquest últim punt s'hagués pogut conèixer fent servir un “rellotge d'altres prestacions”.

També crec que aquest projecte s'ha centrat excessivament en distribucions GNU/Linux, quan en un sistema encastrat ens podem trobar amb diversos sistemes operatius (Windows CE / Mobile, Windows XP Embedded, Symbian, etc.). Per això trobo que també hagués estat interessat executar els benchmarks en alguna de les plataformes amb algun sistema operatiu que no fos un GNU/Linux, per comprovar quin és l'impacte que pot tenir l'operatiu en el rendiment de les aplicacions.

Per últim, també trobo interessant el concepte de plataforma versàtil que s'ha introduït en les conclusions. Una plataforma versàtil seria una que integres diversos dispositius/s architectures i que fos capaç de reconfigurar-se dinàmicament segons la càrrega de treball o limitacions que es trobés. En aquest sentit crec que hi ha forces camins per explorar encara, i que són un repte prou interessants per capficar-s'hi en ells.

Referències

- [1] BARR, Michael. *Michael Barr's Embedded System Glossary*.
<http://www.netrino.com/Publications/Glossary/index.php>

- [2] MORENO BERENGUÉ, Marc. *Plataformes Embedded i Processador Heterogenis*.
Bellaterra: 2007.

- [3] VV.AA. *ARM Product Backgrounder*. Gener 2005. <http://www.arm.com/miscPDFs/3823.pdf>

- [4] EDA, Hiroki; SHINTOH, Tomonori. *ARM CPU Core Dominates Mobile Market*.
<http://techon.nikkeibp.co.jp/NEA/archive/200204/177680/>

- [5] EDWARDS, Lewin. *Migrating from x86 to PowerPC*. Gener 2005.
<http://www.ibm.com/developerworks/library/pa-migrate/>

- [6] VV.AA. *Embedded Processor and System-on-Chip Quick Reference Guide*.
<http://linuxdevices.com/articles/AT4313418436.html#x86>

- [7] CLARKE, Peter. *ARM vs. Intel? It's ARM for now*. Gener del 2008.
<http://www.eetimes.com/news/semi/showArticle.jhtml?articleID=205208036>

- [8] WEISS, Alan R. *Dhrystone Benchmark: History, Analysis, "Scores" and Recommendations*. Austin, novembre del 2002.

[9] ENGBLOM, Jakob. *Why SpecInt95 Should Not Be Used to Benchmark Embedded Systems Tools*. Uppsala. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=627311

[10] YORK, Richard. *Benchmarking in context: Dhrystone*.

[11] Intel Core 2 Duo E6750. <http://techreport.com/articles.x/12737/13>

[12] Advantech AIMB-240. http://www.advantech.com/products/Intel-Pentium-4-Processor-based-Mini-ITX-Motherboard/mod_1-2JKDYN.aspx

Bibliografia

VV.AA. *Embedded Processor and System-on-Chip Quick Reference Guide*. <http://linuxdevices.com/articles/AT4313418436.html>

WEISS, Alan R. *Dhrystone Benchmark: History, Analysis, "Scores" and Recommendations*. Austin, 1 de novembre 2002. <http://www.ebenchmarks.com/download/ECLDhrystoneWhitePaper2.pdf>

NG, Charles. *Non-Volatile Memory Requeriments for Automotive Systems*. Santa Clara, setembre de 2006. http://www.iec.org/newsletter/sept06_2/design_eng.html

LAL SHIMPI, Anand. *More Details on the Intel Atom Emerge*. Març de 2008. <http://www.anandtech.com/cpuchipsets/intel/showdoc.aspx?i=3254>

VV.AA. *Embedded system* http://en.wikipedia.org/wiki/Embedded_system

VV.AA. *ARM Architecture* http://en.wikipedia.org/wiki/ARM_architecture

VV.AA. *X86 Architecture* http://en.wikipedia.org/wiki/X86_architecture

VV.AA. *Benchmark* [http://en.wikipedia.org/wiki/Benchmark_\(computing\)](http://en.wikipedia.org/wiki/Benchmark_(computing))

VV.AA. *Texas Instruments OMAP* <<http://en.wikipedia.org/wiki/Omap>>

Annexos

Unixbench: components i mòduls

El conjunt de benchmarks que formen part d' UnixBench 4.10 es divideixen en cinc categories:

1. **Arithmetic:** en aquest grup hi podem trobar benchmark on la càrrega de treball principal es troba en una sèrie d'operacions aritmètiques:

- i. *whetstone-double*: implementació del benchmark Whetstone, utilitzant una precisió “double” per fer els càlculs en punt flotant.

Whetstone, al contrari que Dhrystone, fa sobretot èmfasi en les operacions aritmètiques, tant amb sencers com en punt flotant. Per altra part, aquest benchmark també inclou crides a funcions i procediments, condicionals, i operacions amb vectors.

El resultat es dona en MWIPS.

- ii. *register / short / int / long / float / double*: en aquests microbenchmarks es realitzen un mateix conjunt de operacions aritmètiques, però amb un tipus diferent de dades.

El resultat es dona en “lps” (loops per second), que en aquest cas ens indica el número de vegades que s'ha pogut realitzar tot el conjunt d'operacions en el interval de temps d'un segon.

- iii. *arithoh*: serveix per a tenir una referència del temps d'overhead que s'afegeix al realitzar les operacions aritmètiques dels microbenchmarks anteriors.

Al igual que en el cas anterior, el resultat es retorna en “lps”.

Cal destacar que els benchmarks (ii) i (iii) comparteixen el mateix codi font.

2. **System:** en aquest grup es troben els benchmarks que fan crides al sistema per realitzar alguna tasca determinada:

- i. *syscall*: evalua el temps necessari per fer les crides de les funcions de sistema `dup()`, `close()`, `getpid()`, `getuid()`, i `umask()`.
- ii. *pipe*: un procés crea un pipe connectat a ell mateix en el que s'envia i es rep 512 KiB. Serveix per evaluar el temps d'overhead que hi ha al enviar dades per un pipe.
- iii. *context1*: en aquest cas el programa principal crea un procés pare i fill connectats mitjançant un pipe. El bucle principal consisteix en que el procés pare envia el número d'iteracions lectura/escriptura que porten ambdós processos, i el fill únicament rep aquesta dada.
- iv. *spawn*: retorna el número de còpies que el sistema és capaç de crear i matar un procés fill en un temps determinat.
- v. *execl*: en aquest microbenchmark es mesura el número de vegades que aquest programa es pot cridar a si mateix en un temps determinat.
- vi. *fstime* / *fsbuffer* / *fsdisk*: retorna el throughput del sistema de fitxers en operacions de lectura, escriptura i còpia de dades utilitzant un buffer intermig. El resultat es dona en KiB/s.
- vii. *shell*: amb aquest benchmark s'intenta carregar el sistema amb un número determinat d'scripts (1, 8 o 16) funcionant de forma concurrent. Els scripts realitzen una tasca d'ordenació alfabètica.

El que s'acaba mesurant amb aquest benchmark és el nombre de vegades que els scripts finalitzen la tasca d'ordenació en el interval d'un minut.

3. **Misc**: en aquesta categoria trobem tres benchmarks amb propòsit general:

- i. *C*: retorna el número de còpies que es pot compilar i enllaçar un programa escrit en C en 60 segons.
- ii. *Dc*: retorna el número de còpies que el programa “dc” pot calcular l'arrel quadrada de 2 amb una precisió de 60 segons.
- iii. *hanoi*: algorisme recursiu de les Torres d'Hanoi. Es calcula quantes vegades el programa pot arribar a una solució en un temps determinat.

4. **Dhry:** amb el pas del temps, aquesta categoria ha quedat reduïda a només un benchmark: una implementació del Dhrystone que utilitza registres del processador. En anteriors versions es pot trobar una implementació que “a priori” no utilitza registres del processador, però donat que tots els compiladors moderns en fan ús es va decidir eliminar-ho d'aquest grup.
 - i. *dhry2reg*: implementació del conegut benchmark Dhrystone en la seva versió 2.1, i que utilitza com a màxim 2 registres del processador.

El Dhrystone intenta ser una representació de programació amb aritmètica d'enters (no hi ha operacions amb punt flotant) d'un sistema. La versió implementada a Unixbench 4.10, que està basada en les versions de Rick Richardson i de Reinhold Weicker, consta d'un conjunt de 103 instruccions que intenta representar un “escenari típic” que ens podem trobar al programar amb un llenguatge d'alt nivell (en aquest cas C).

5. **Index:** en aquest grup es troben els benchmarks amb els quals es genera un informe on hi ha una serie d'índexs que es creen a partir de comparar el resultat obtingut en la màquina local amb un resultat base (generat per un ordinador consistent en una SPARCstation 20-61, 128 MiB de RAM, i amb sistema operatiu Solaris 2.3). En aquest s'inclouen els benchmarks del grup System, el Dhrystone i el Whetstone.