Processeurs de traitement du signal (DSP)

État de l'art
Critères de choix
Perspectives

Olivier Sentieys
ENSSAT - Université de Rennes 1

sentieys@enssat.fr

http://archi.enssat.fr/

PLAN

I. Architecture des DSPs conventionnels

II. Critères de choix

III. Évolution des architectures de DSP

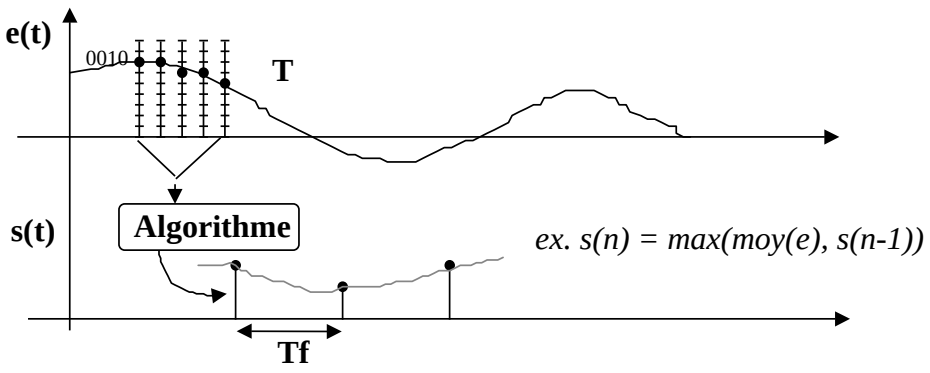
IV. Méthode de programmation des DSP

I. Architecture des DSP

1. Algorithmes de traitement du signal
2. Solutions architecturales
3. Architecture Harvard
4. DSPs conventionnels
5. Modélisation
6. Panorama

Système de TNS

- Systèmes DSP exécutent algorithmes temps réel sur des signaux numériques
  - Signaux numériques : flot de données
  - Temps Réel
    - Temps de l'exécution de l'algorithme Tex guidé par acquisition I/O
    - Période d'échantillonnage T
    - Période des sorties Tf (frame period) > Tex
    - Ni plus vite ... ni plus lentement (not faster ... not slower)



Fonctions typiques de TS

- Convolution, filtrage
  - y = y + x.h : MAC
- Adaptation
  - $y_n = y_{n-1} + x.h$: MAD
- FFT, multiplication complexe
  - xr=xr.wr-xi.wi; xi=xr.wi+xi.wr
- Viterbi
  - a1 = x1 + x2; a2 = y1 + y2; y = (a1>a2) ? a1 : a2 : ACS

Caractéristiques algorithmiques

- Quantité importante de données
  - scalaires, vectorielles, matricielles, multidimensionnelles
  - opérations I/O intensives par DMA
- Charge de calcul importante
  - multiplications-accumulations (convolution)
  - multiplications-additions (FFT, DCT, adaptation, distances,...)
- Virgule Fixe ou Flottante
  - problèmes liés à la quantification !!!
- Calculs d'adressage complexes
  - bit-reverse ou similaire (FFT), vieillissement des données, ...
- Boucles de traitement courtes
  - les instructions peuvent être placées en interne
  - pas besoin de grande taille de cache (données ou instructions)

Benchmarks algorithmiques

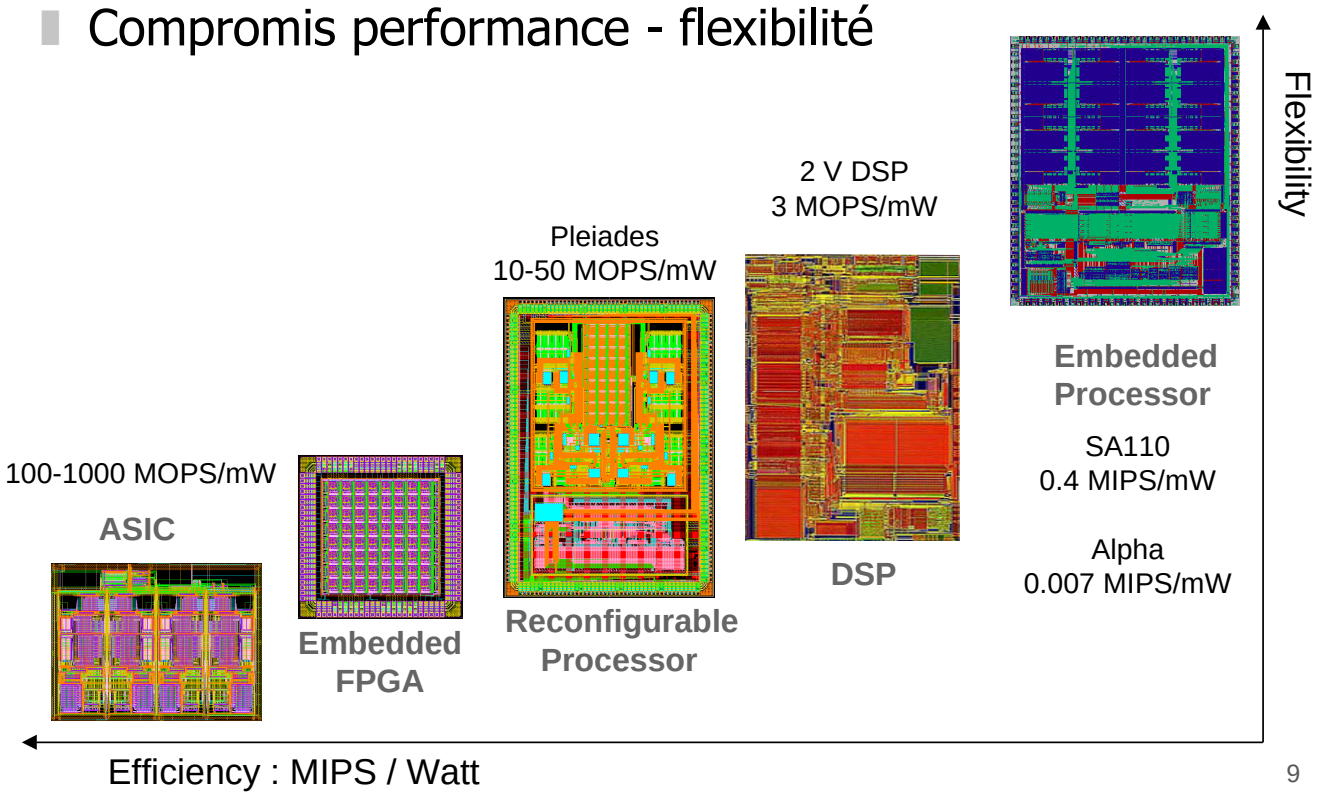
Function	Description	Application Examples
Real block finite impulse response (FIR) filter	FIR filter that operates on a block of real (not complex) data	G.728 speech encoding, other speech processing
Complex block FIR filter	FIR filter that operates on a block of complex data	Modem channel equalisation
Real single-sample FIR filter	FIR filter that operates on a single sample of real data	Speech processing, general filtering
Least-mean-square adaptive FIR filter	LMS adaptive FIR filter that operates on a single sample of real data	Channel equalisation, servo control, linear predictive encoding
Infinite inpulse response (IIR) filter	IIR filter that operates on a single sample of real data	Audio processing, general filtering
Vector dot product	Sum of the pointwise multiplication of two vectors	Convolution, correlation, matrix multiplication, multidimensional signal processing
Vector add	Pointwise addition of two vectors producing a third vector	Graphics, combining audio signals or images, vector search
Vector maximum	Discovery of the value and location of a vector's maximum value	Error-control coding, algorithms using block floating-point arithmetic
Convolutional encoder	Application of convolutional forward error-correction code to a block of bits	North American digital cellular telephone equipment (IS-54 standard)
Finite-state machine (FSM)	A contrived series of control operations (test, branch, push, pop) and bit manipulations	Control operations appear in nearly all digital signal processing applications
256-point, radix-2, in-place fast Fourier transform (FFT)	FFt conversion of a normal time-domain signal into the frequency domain	Radar, MPEG audio compression, spectral analysis

I. Architecture des DSP

1. Algorithmes de traitement du signal
2. Solutions architecturales
3. Architecture Harvard
4. DSPs conventionnels
5. Modélisation
6. Panorama

Solutions architecturales

- Compromis performance - flexibilité

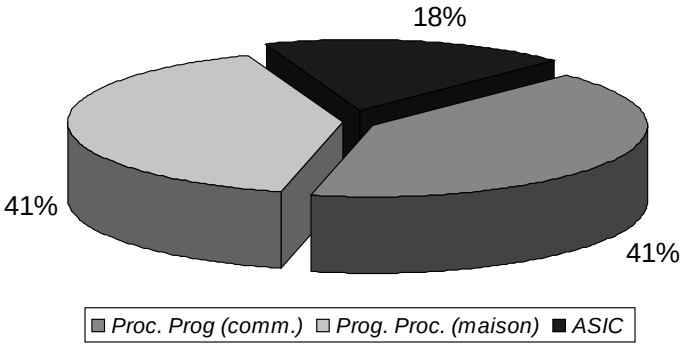


Solutions architecturales

- Processeurs programmables du commerce (ISP)
  - Processeurs généraux ***RISC, VLIW***
  - Processeurs de Traitement du Signal (DSP)
  - Processeurs Multimédia
  - Microcontrôleurs
- Processeurs programmables "maison" (ASIP)
  - De type DSP ou µCtrl
- Processeurs et logique reconfigurables
  - FPGA enfouis, Processeur reconfigurable
- Coprocesseurs ASIC

Solutions architecturales

- Utilisation dans un cadre industriel



[Après P. Paulin]

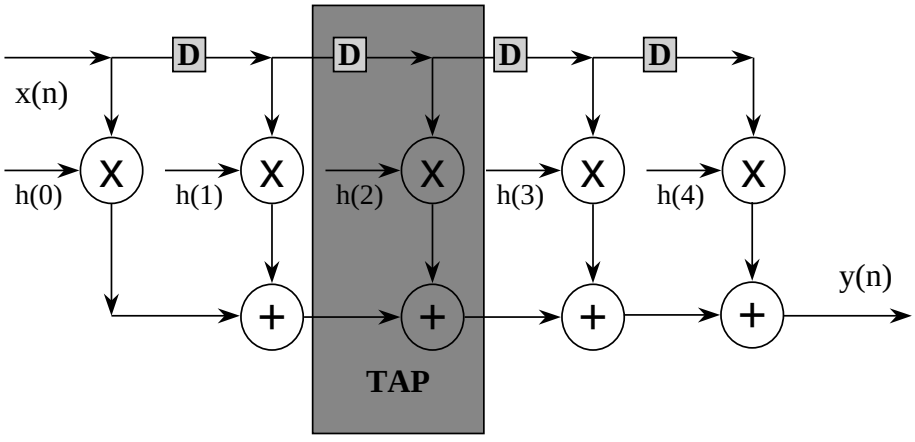
- Performances

Processeur	MIPS	Vdd	Pmoy	MIPS/Watt
CoolRisc	14	3V	2.8 mW	5000
TMS C54x	30-200	1.8V	460 mW	1500-3000
TMS C6x	1600	2.5V	2W	800
Dec α	500	1.8-3V	50 W	7-10

Exemple *Fil Rouge*

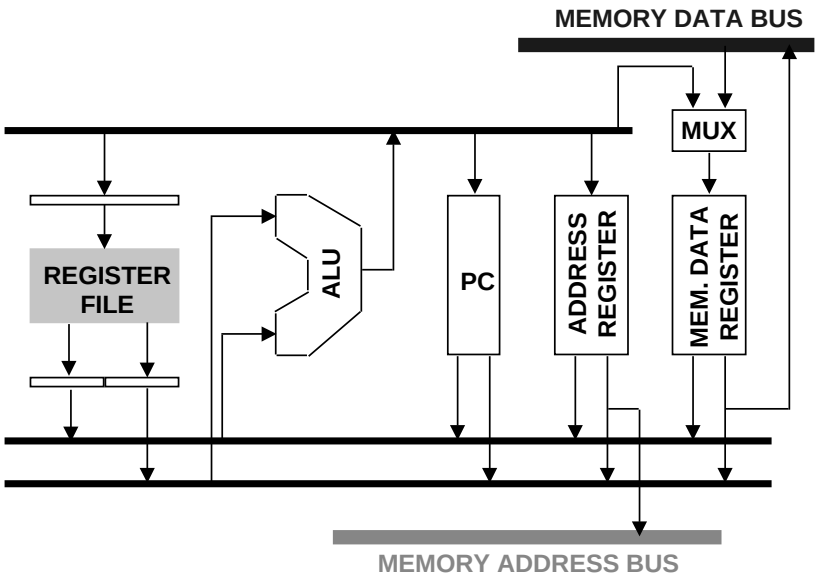
- Filtre Numérique RIF sur N points

$$y(n) = \sum_{i=0}^{N-1} h(i).x(n-i) = x(n) * h(n)$$



Architecture Von Neumann

- Les processeurs RISC



Code example:
multiply & accumulate
*r2 = r2 + #imm * r1*

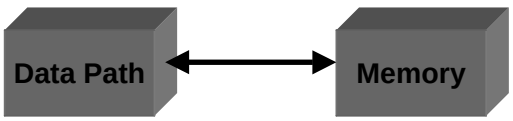
```
mov #imm,r3
mul r1,r3
add r3,r2
```

⇒ 3 instructions, >3 cycles d'horloge

FIR sur machine Von Neumann

- Problèmes
  - Bande passante avec la mémoire
  - Code pour le contrôle et la gestion de l'adressage
  - Multiplication lente

```
loop:
   mov *r0,x0
   mov *r1,x1
   mpy x0,y0,a
   add a,b
   mov y0,*r2
   inc r0
   inc r1
   inc r2
   dec ctr
   tst ctr
   jnz loop
```



Exécution en 15 à 20 cycles

I. Architecture des DSP

1. Algorithmes de traitement du signal
2. Solutions architecturales
3. **Architecture Harvard**
4. **DSPs conventionnels**
5. **Modélisation**
6. **Panorama**

Généralités sur les DSPs

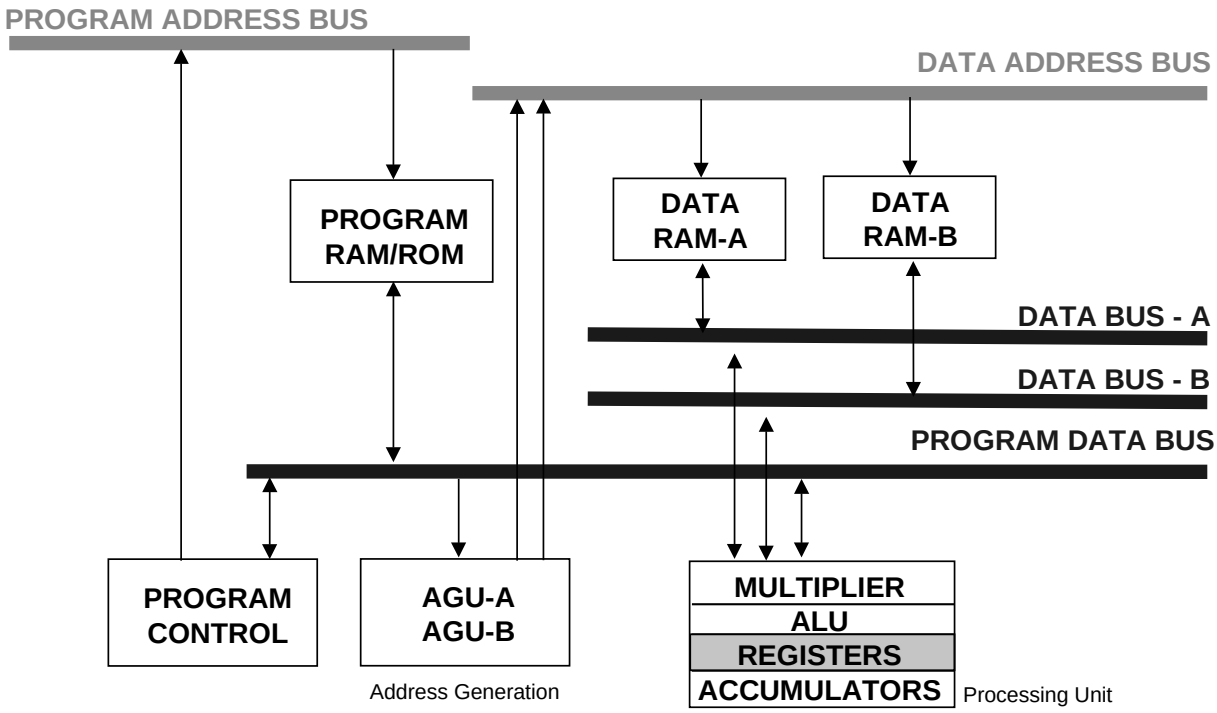
http://archi.enssat.fr

Propriétés du traitement numérique du signal	Conséquences sur les architectures
Calculs intensifs et répétitifs	■ Fonctionnement pipeline ■ Architecture Harvard ■ Structures de contrôle évoluées
Primitives simples	■ Unités de traitement spécialisées câblées ■ Gestion d'adressage complexes

Le tout dans un environnement temps réel

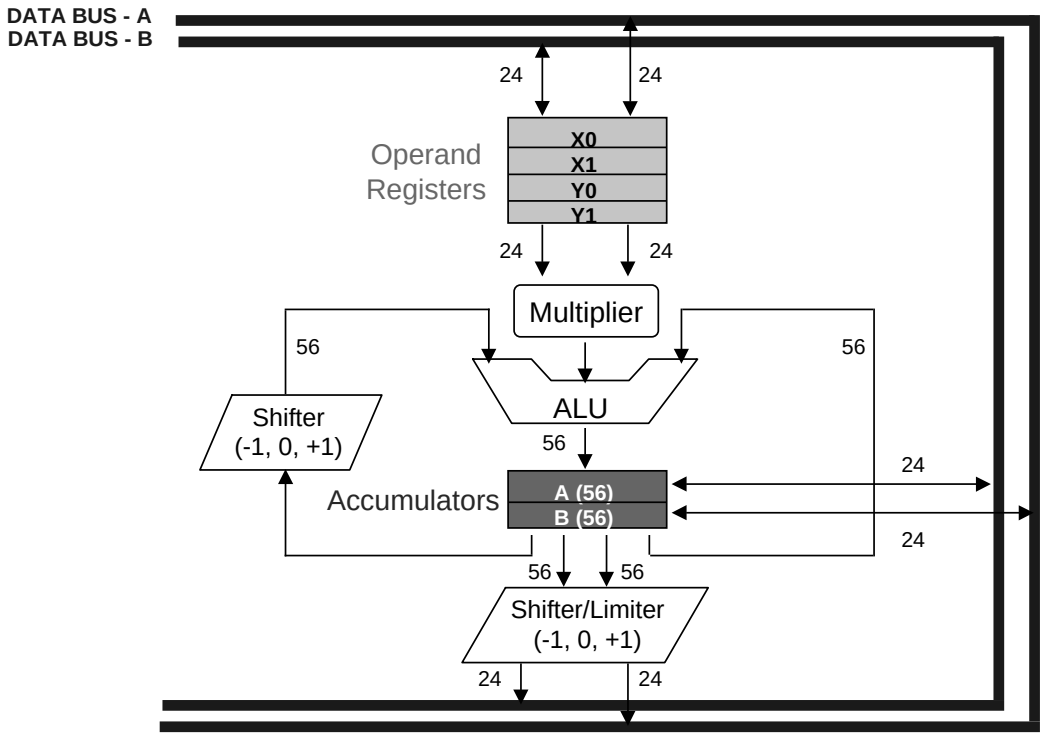
Architecture Harvard

http://archi.enssat.fr



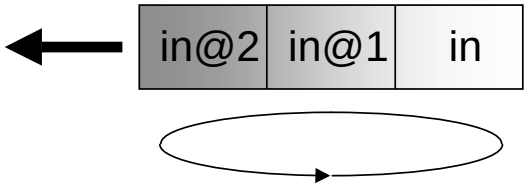
Architecture Harvard

http://archi.enssat.fr



Architecture Harvard

- Bus et mémoires données/instructions séparées
- Unité de traitement de type mpy-acc
- Registres distribués (≠ RISC register file)
  - Chaque module (ALU) possède ses propres registres locaux
- Génération adresses efficaces (AGUs)
  - Modes d'adressage spéciaux : auto incr-decr, circular buffering (delay line) ...

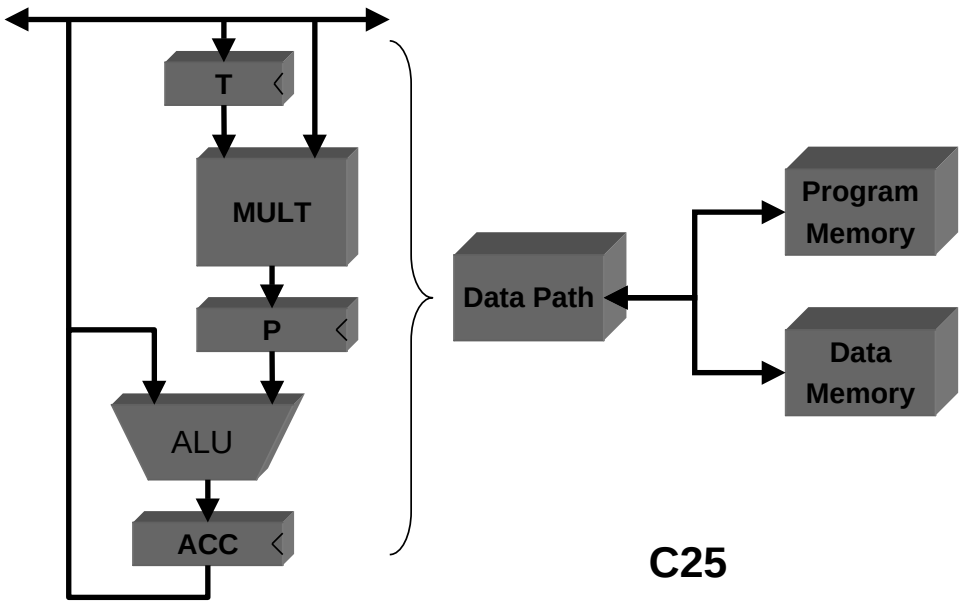


FIR sur DSP conventionnel

- Jeu d'instructions complexe

```
loop:
   acc=acc+P
// P=T*PM(*r0+)
// T=DM(*r1+)
   jnz loop
```

Exécution en 1 cycle



DSP conventionnel

- 16-24 bits en virgule fixe, accumulation 40-60 bits
- 32 bits en virgule flottante
- 16 à 32 bits d'instructions
- Une instruction par cycle, jeu d'instructions complexe
- Architecture non orthogonale, fortement contrainte
- Mémoire à accès multiples "on-chip"
- Unités dédiées de gestion des adresses
- Matériel dédié pour la gestion des boucles

100 millions de produits intègrent ces DSP

I. Architecture des DSP

DSP : modélisation

- IP : *Instruction Processor* : FU qui interprète les instructions et les passe au DP
- DP : *Data Processor* : FU qui modifie ou transforme les données
- IM : *Instruction Memory* : stocke les instructions
- DM : *Data Memory* : stocke les données traitées par le DP
- EIU : *External Interface Unit* : contrôle les accès aux données ou instructions externes, ou à d'autres processeurs

Diagramme d'état d'un IP

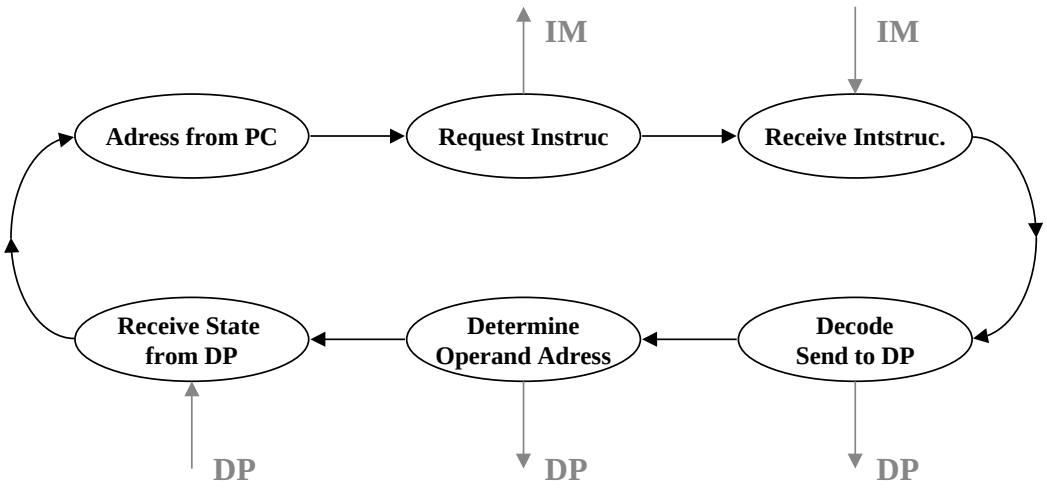
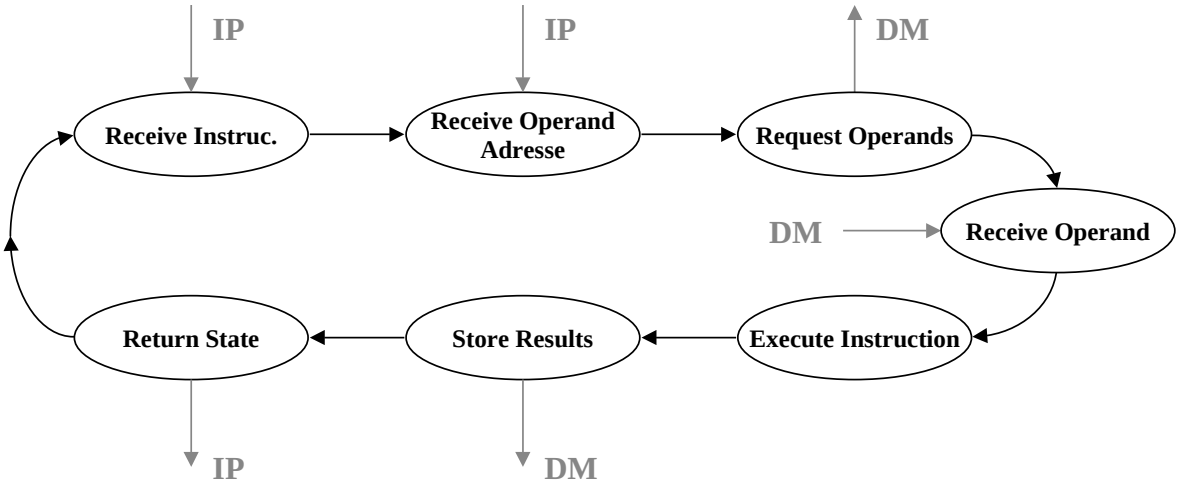
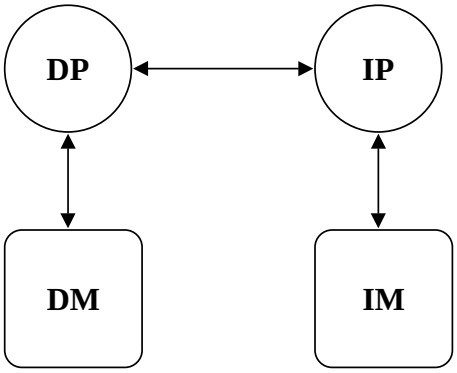


Diagramme d'état d'un DP



Architecture Harvard de base

- Une mémoire données et une mémoire programme
- *Fetch* d'instruction pipeliné avec *fetch* opérande



Optimisations des performances

Comment améliorer l'architecture Harvard de base ?

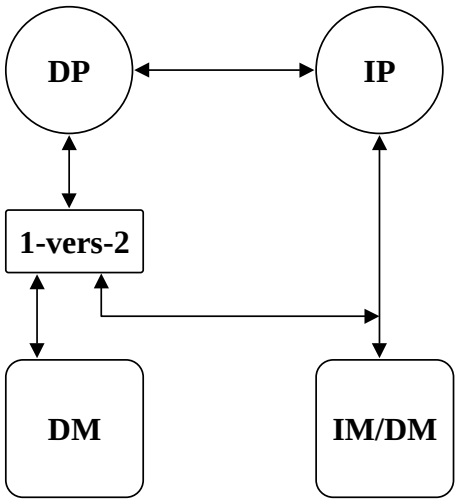
- Transformations du diagramme d'état
  - réarrangement des états pour augmenter le temps de cycle
  - exécution en parallèle d'états (ex. Store Results et Return State)
- Pipeline de l'exécution des états
  - mise en parallèle d'états appartenant à des instructions successives
- Augmentation du nombre de FU
  - plusieurs DP peuvent exécuter des instructions en parallèle sur des données indépendantes

Interconnexions entre FUs

- Les interconnexions suivantes sont valables pour toutes les FUs précédentes (DP, IP, IM, DM).
  - 1 - vers - 1
    - une FU est connectée à une autre FU
  - n - vers - n
    - une FUi d'un ensemble de FUs est connectée à une autre FUi
  - 1 - vers - n
    - une FU est connectée à n autres FUs d'un ensemble de FUs
  - n - par - n
    - toute FUi d'un ensemble est connectée à chaque autre FUj

Modification 1

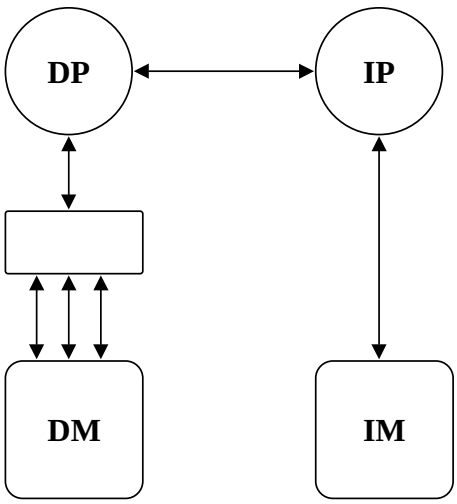
- Autorisation de mémorisation de données dans l'IM
- En un cycle : *fetch* deux opérandes de la mémoire, exécute MAC, écriture du résultat en I/O ou mémoire



AT&T DSP32 et DSP32C

Modification 2

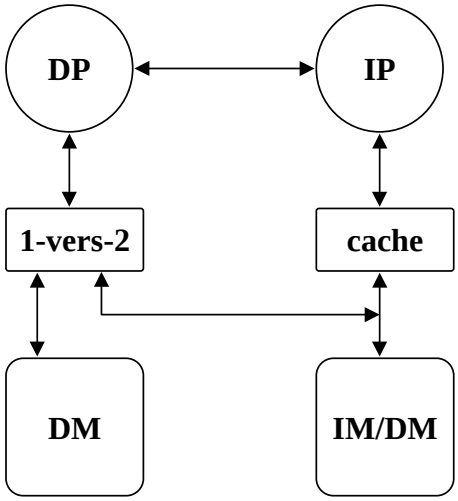
- DM est une mémoire multi-ports, plusieurs accès aux données par cycle
- Utilisable pour des mémoires internes au CI



Fujitsu MB86232 (3 ports en mémoire interne)

Modification 3

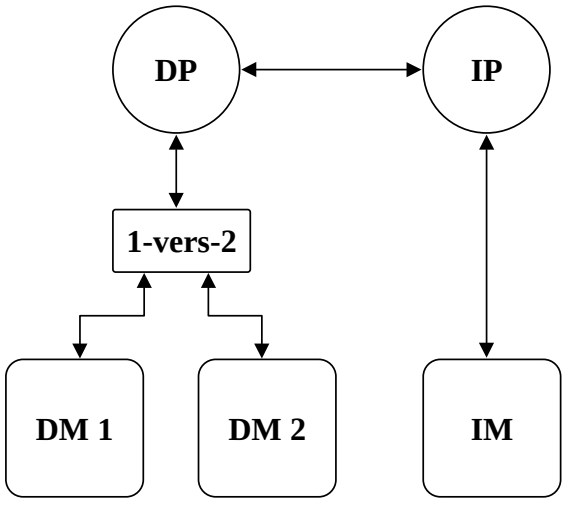
- Cache pour charger les instructions fréquentes
- Évite les conflits d'accès données et instructions de la modification 1



TMS320C25 : cache 1 instruction pour les boucles
DSP16 : cache 15 instructions
ADSP-2100 : cache 16 instructions

Modification 4

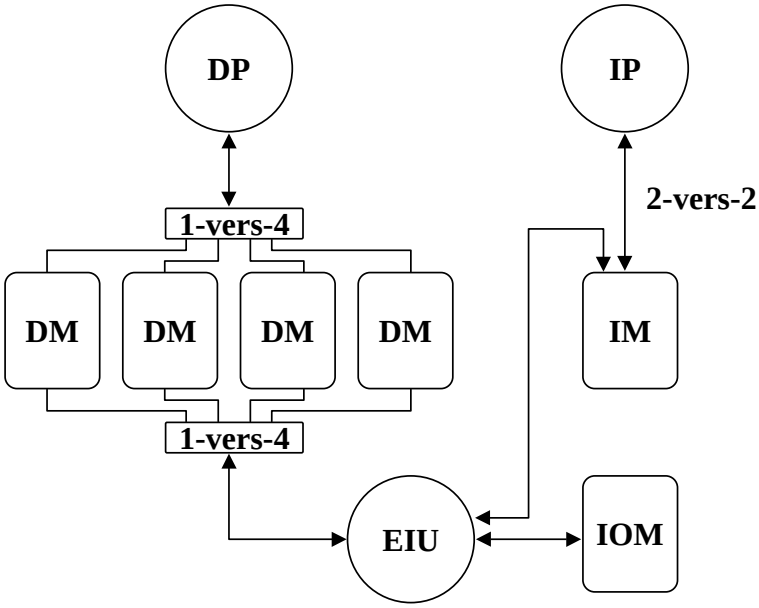
- Deux mémoires données DM séparées
- En un cycle : *fetch* deux opérandes et une instruction si le cycle mémoire est le même que le cycle d'instruction de base



Motorola DSP 56001 et 96002
TMS320C30 et C40

Modification 5

- Bancs mémoire multiples
- Instruction multi-opérandes en parallèle avec accès I/O
- Problème de partition des données sur un grand nombre de bancs (FFT)

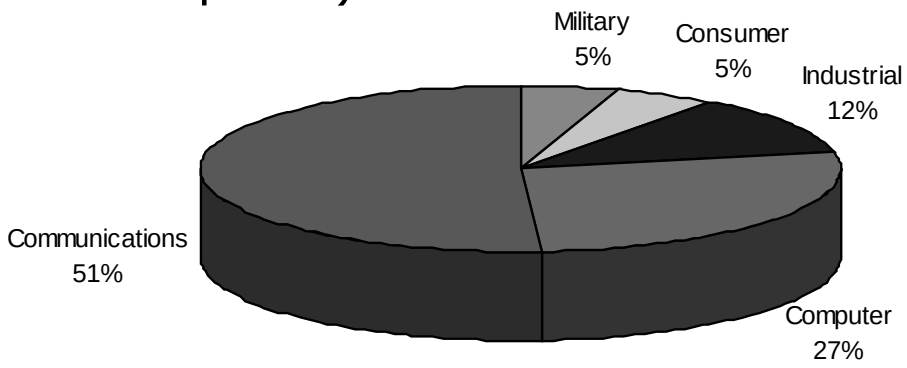


Hitachi DSPi (6 bancs mémoire)

I. Architecture des DSP

Le marché des processeurs DSP

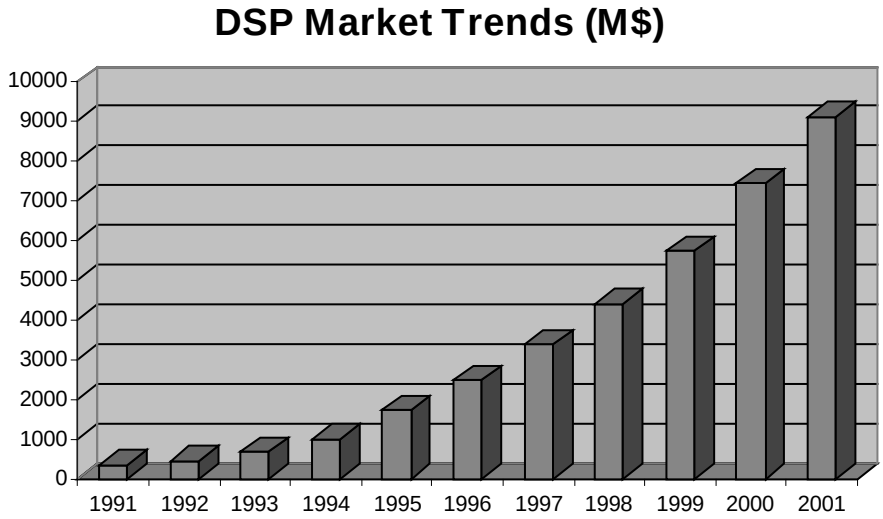
- Les communications dominent
- Ordinateurs sont seconds (e.g. compression d'image, reconnaissance de parole, conversion texte-vers-parole)



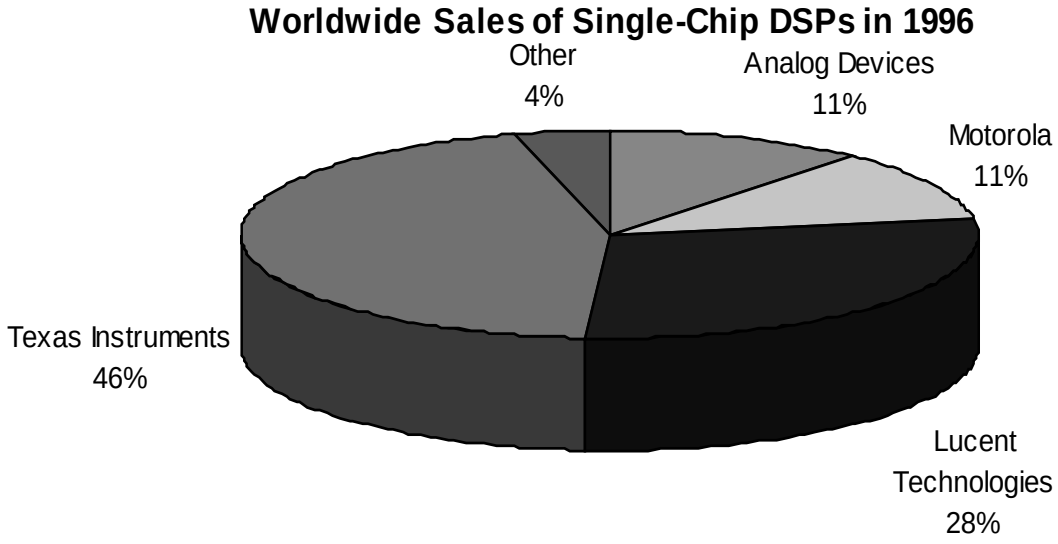
[ICE97] *B. McCleanICE, "Status 1997: A Report on the Integrated Circuit Industry", Integrated Circuit Engineering Corporation (ICE), Scottsdale, 1997*

Le marché des processeurs DSP

- En haut de la liste des plus fortes croissances du marché de l'industrie des semi-conducteurs.

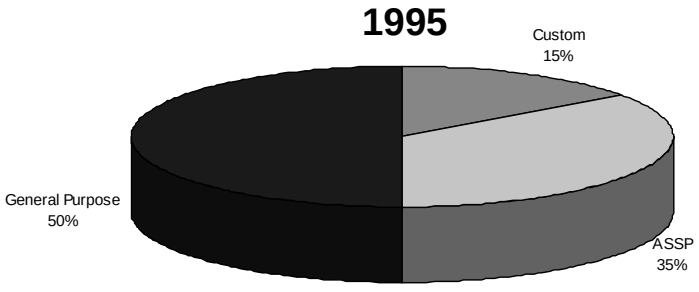


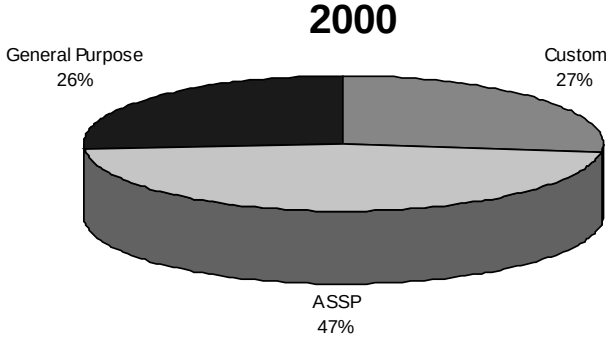
Le marché des processeurs *DSP*



Le marché des processeurs *DSP*

- *Application-specific standard products* (ASSPs) dominent
- Solutions à intégration de *DSP core*
  - e.g. DSP Group Pine and OAK DSP-cores for integration into Atmel's cell and gate array library

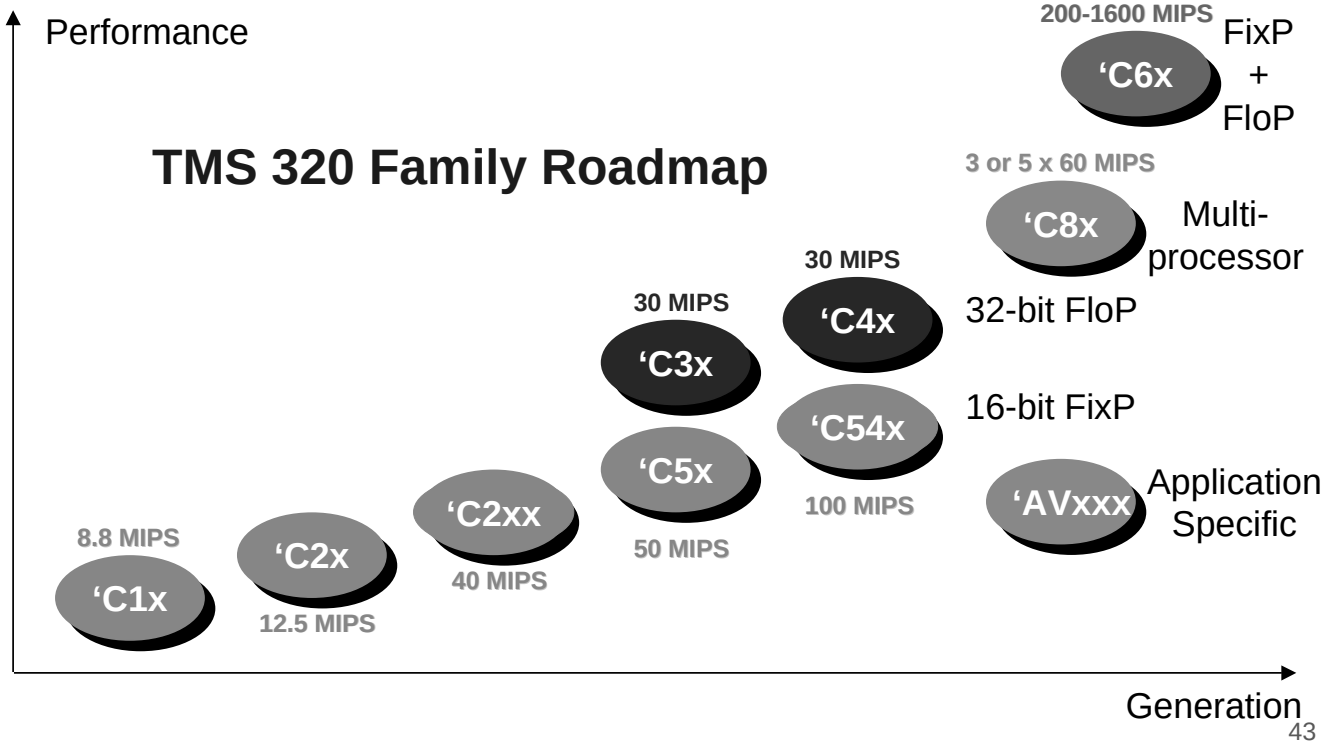




Cœurs de DSPs

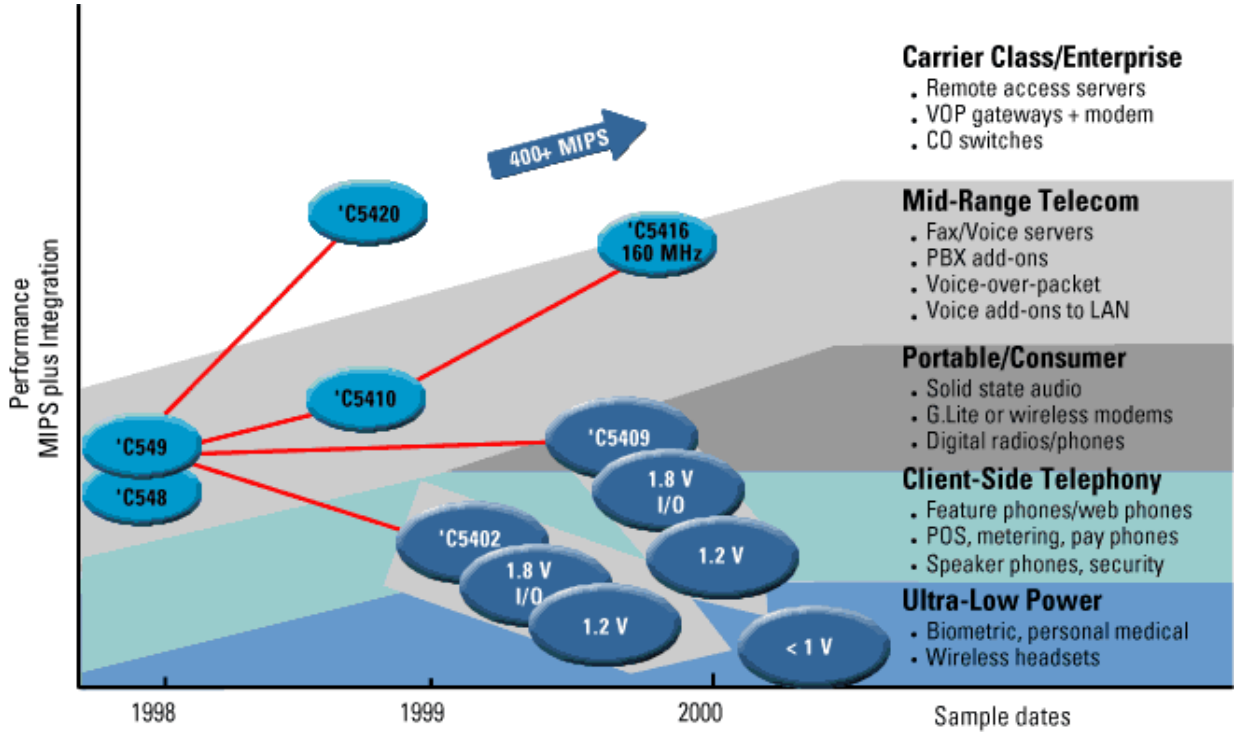
- Délivré sous licence, modulaire, bloc IP
- Caractérisation d'un cœur
  - foundry-captive, licenciable
- Contenu du cœur
  - cœur (+ mémoire (+ périphériques))
- Exemples
  - Infineon Carmel
  - Infineon TriCore
  - ARM
  - DSP Group OAK/PINE
  - ST D950, ST Lx

Texas Instruments



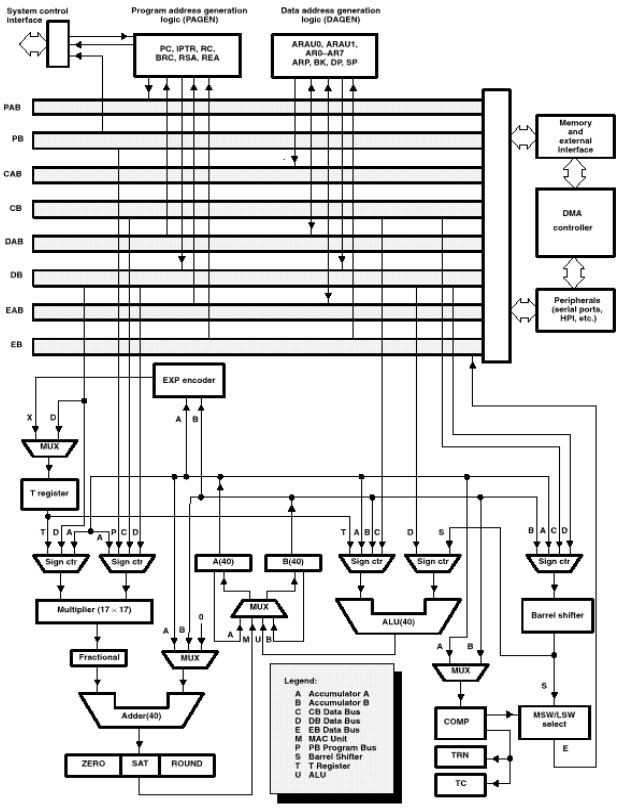
Famille C5x

- Evolution

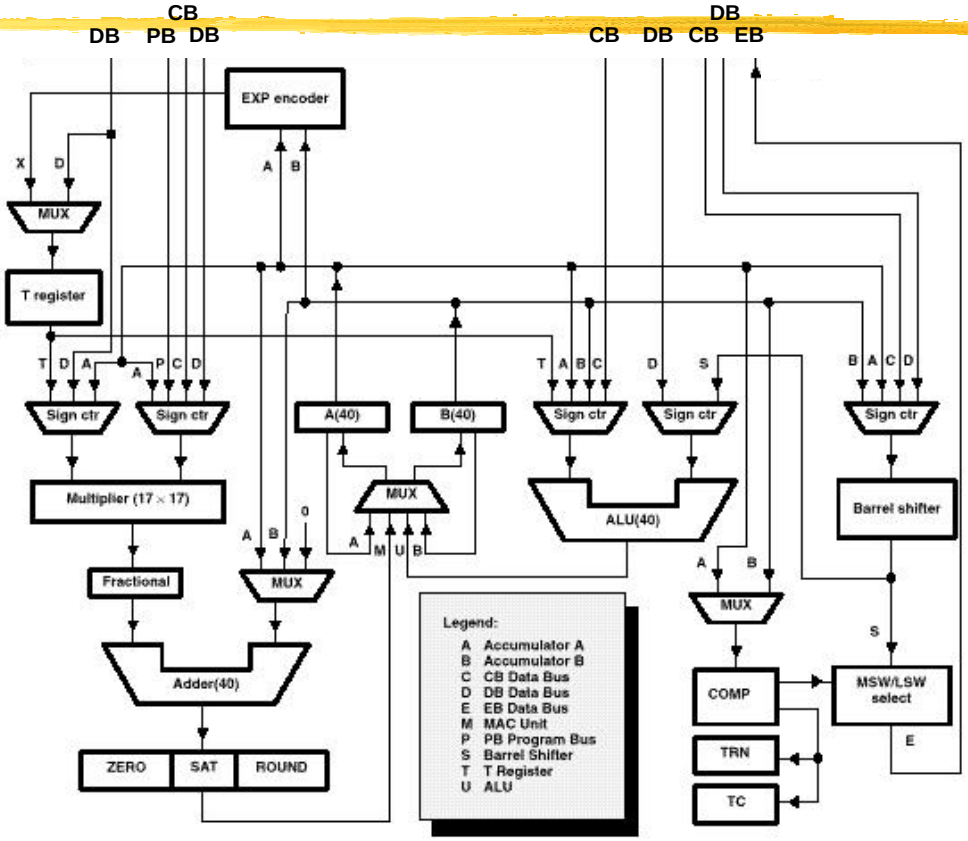


C5x Architecture

- C54x
  - 40-100-160 MIPS
  - 1000-2000-3000 MIPS/W
  - 17x17b multiplier, 40b ALU, 40b adder, ACS unit
  - Dual cores (C5420)
  - 60% of cellular handsets
  - $5 for C5402 100MIPS - $75
- C55x
  - 160 MHz, 320 MIPS, 80mW
  - 4000 MIPS/W
  - Dual MAC
  - 160 KW SRAM
  - 400 MIPS in 2001, 20 MIPS/mW



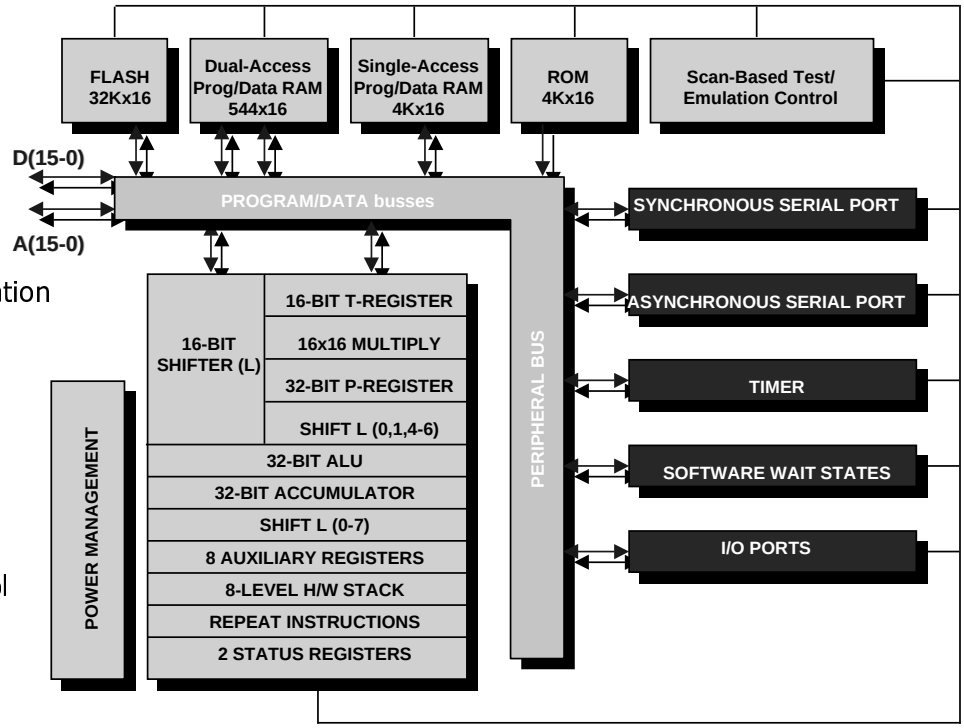
C5x Architecture



Texas Instruments

TMS 320C2xx Series - Architecture/Features

- **Key specifications:**
  - FixP DSP (20-40 MIPS)
  - 3V and 5V versions
  - Typical active power:
    1.9 mA/MIPS@5V
    1.1 mA/MIPS@3V
  - 100-pin TQFP
  - JTAG scan-based emulation
  - Price/1000: $6-$23
- **Key applications:**
  - Set-top boxes
  - Feature phones
  - Security systems
  - Telecom
  - Servo and motor control
  - Radar Detectors
  - Digital Cameras
  - Modems
  - CD ROMs ...



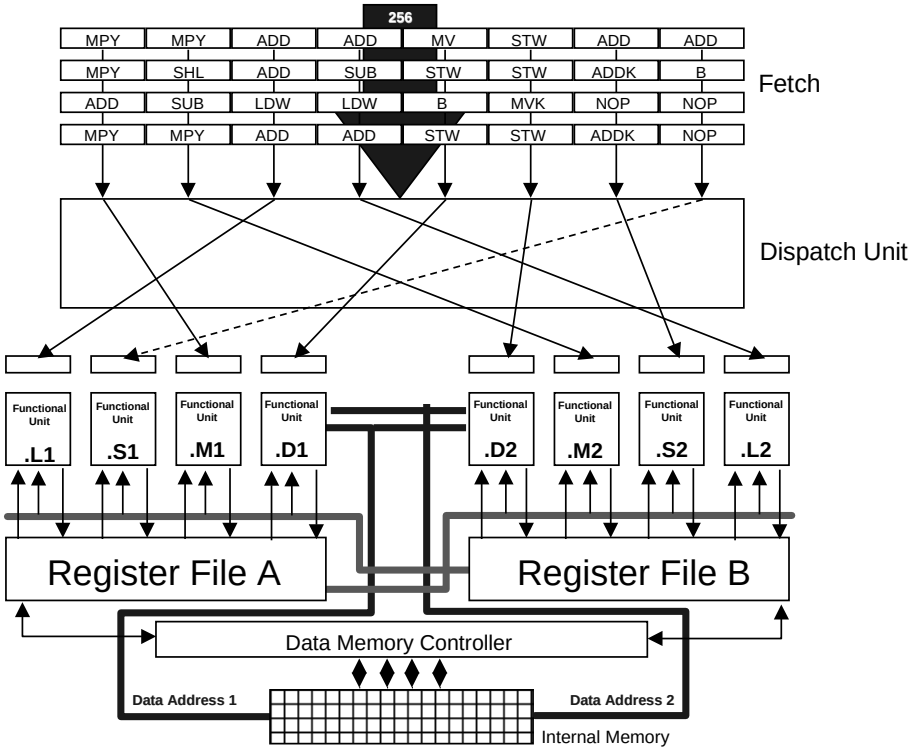
Texas Instruments ASSP's

TMS 320 AV Series

- DSP products aimed specifically at the set-top box market
- TMS320AV120 MPEG Audio Decoder
  - Input MPEG audio stream; Output: 16- or 18-bit serial PCM.
  - Usable as stand-alone MPEG audio decoder, eliminating the need for a host processor.
  - Cost: $10.00 (Quantity: 1000 units).
- TMS320AV220 Video CD MPEG Decoder
  - Integrates MPEG-1 system decoder, audio buffer and on-chip microcode.
  - Input: MPEG stream; Output: 704x288 @ 50 Hz.
  - Synchronises audio and video outputs without external host.
  - Integrates Video CD control functions (fast forward, reverse and freeze frame).
  - Cost: $29.00 (Quantity: 1000 units).
- TMS320AV420 Digital NTSC Video Decoder
  - Converts RGB or YUV bit-streams output from the 'AV220 to an analog NTSC signal for TV.
  - Vertically interpolates scan lines to create a smooth picture by reducing the noise and flicker that is associated with line-doubled implementations of MPEG-1 Video.
  - Generates synchronisation timing signals for the 'AV220 Video Decoder.
  - Supports overlays, providing the ability to superimpose both text and graphics on the video display.
  - Cost: $7.50 (Quantity: 1000 units).

Texas Instruments

TMS 320C6x Series - VelociTI 'C6200 CPU



Texas Instruments

TMS 320C6x Series - Features

- Advanced VLIW CPU with eight functional units (RISC-like code)
- 1600 MIPS @ 200MHz, 0.25µ CMOS
- 1M-bit on-chip memory (512kbit data / 512kbit program)
- 32-bit external memory interface
- Two enhanced-buffered serial ports (high bandwith telecom, interprocessor com)
- 16-bit host access port (Host processor access to on-chip data memory)
- Flexible PLL clock generator (Multiplies ext. clock rate for 2 or 4 for maximum CPU performance)
- 352-lead ball grid array package
- Price/1000: $135

Texas Instruments

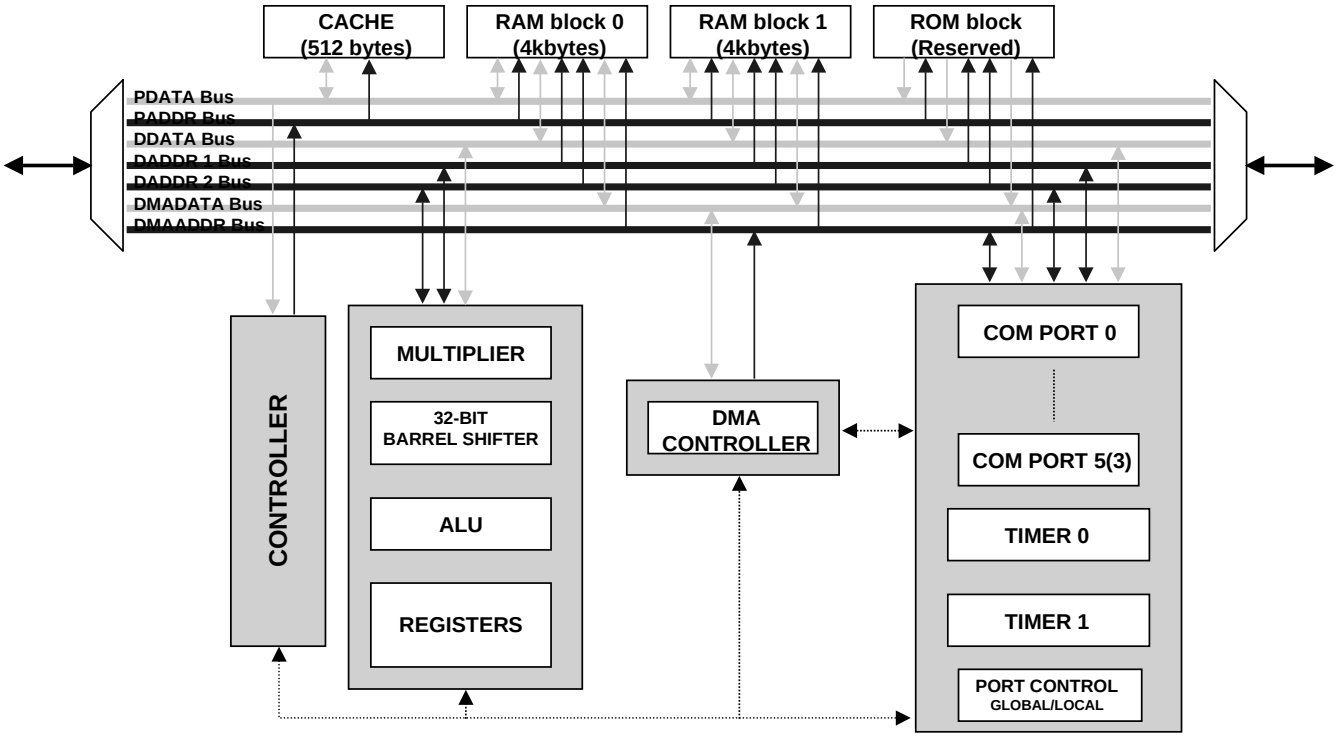
TMS 320C6x Series - CPU

- Two sets of functional units including:
  - Two multipliers
  - Six aritmetich logic units (ALUs)
  - 32 registers with 32-bit wordlength each
  - data-addressing units .D1 and .D2 exclusively responsible for data transfers between memory and the register files
- 8-/16-/32-bit data support
- 40-bit arithmetic options (extra precision for vocoders...)
- Saturation and normalisation
- Bit-field manipulation and instruction: extract, set, clear, bit counting.

Texas Instruments

TMS 320C4x Series - Architecture

http://archi.enssat.fr



Texas Instruments

TMS 320C4x Series - Features

http://archi.enssat.fr

- 40/50/60 MHz 40-bit FloP DSP
- 32-bit integer data format
- 4 or 6 20 MBytes/s communication ports
- Dual data and address busses
- 8 kBytes internal SRAM + Boot ROM
- JTAG analysis & emulation
- 50 MFLOPS, 25 MIPS, 275 MOPS @ 50 MHz
- Price/1000: $130-$196

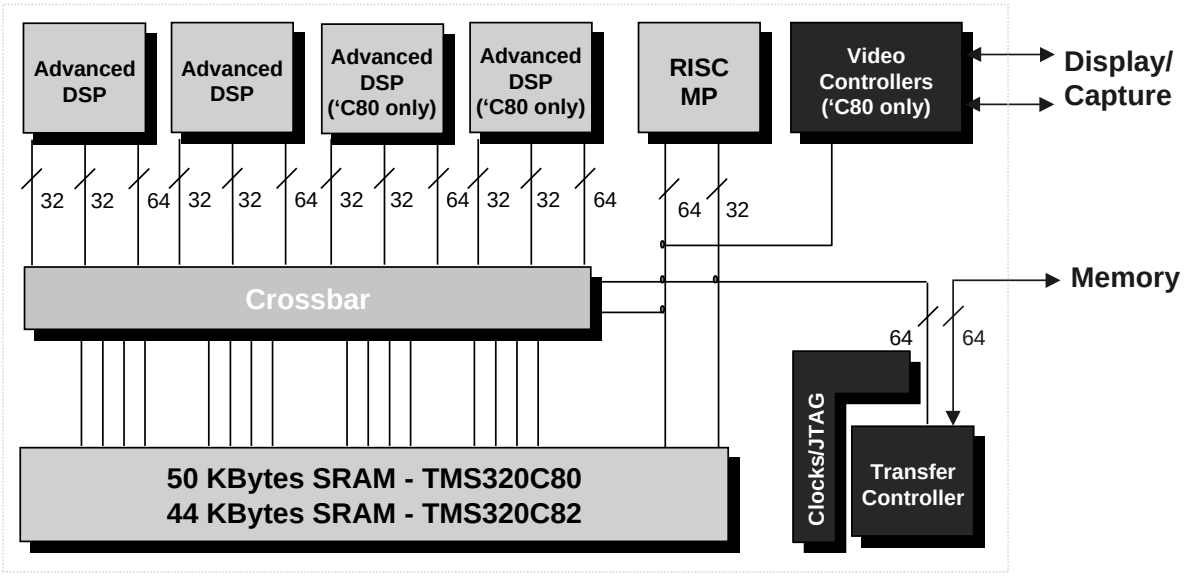
Texas Instruments

TMS 320C8x Series - Features

- Key Specifications:
  - Fully programmable MIMD architecture
  - 4(2) parallel 32-bit data/64-bit instruction advanced FixP DSPs
  - 32-bit RISC master processor
  - 400 MByte/s 4 GByte address space
  - 20/25 ns cycle time
  - Video controller
  - 50 (44) kBytes on-chip SRAM
  - 2 (1.5) BOPS performance
  - 305-pin PGA
  - Price/1000: $134-$331
- Key features:
  - Intelligent on-chip transfer controller and on-board memory (SRAM, DRAM and VRAM)
  - Multiple 32-bit parallel processing, advanced DSP
  - 32-bit ALU, can be divided into two 16-bit ALUs or four 8-bit ALUs for parallel processing on lower precision data
  - 8-, 16-, 32-, 64-bit dynamic external bus interface
- Key applications:
  - Video conferencing, video phones, high speed telecommunications;
  - Image and video processing, multimedia workstations, 2D and 3D graphics accelerators, virtual reality;
  - Security; Radar and sonar systems.

Texas Instruments

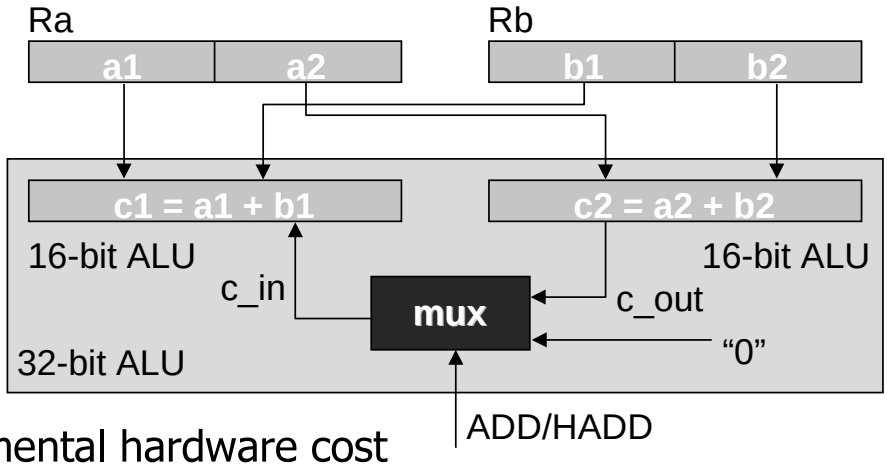
TMS 320C8x Series - Block Diagram



Texas Instruments

TMS 320C8x Series - Subword Parallelism

- Simultaneous processing of lower-precision data packed in word (e.g. 16/32-bit ADD)

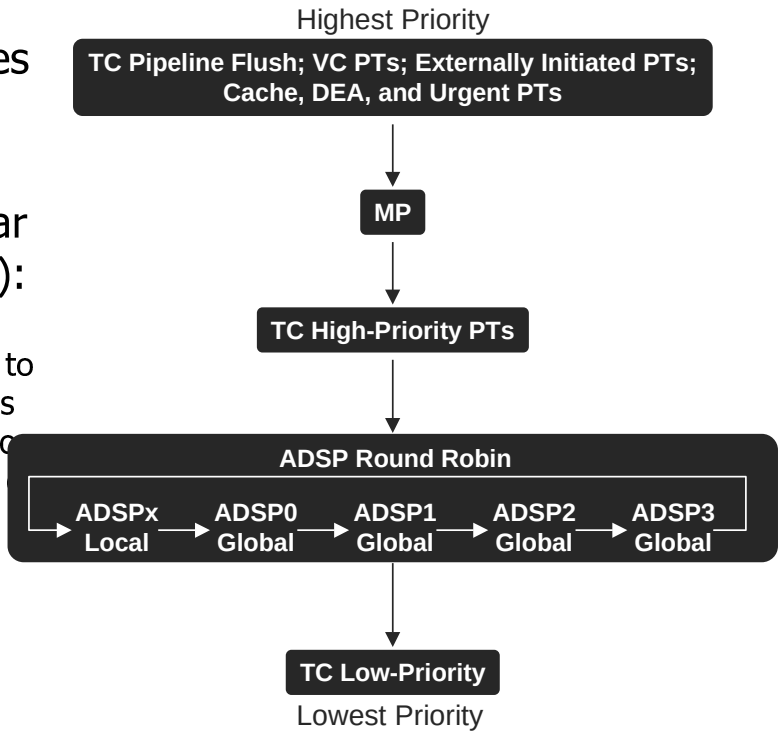


- Small incremental hardware cost
- Two prerequisites
  - lower precision sufficient
  - exploitable data parallelism
    => Low-level video processing tasks

Texas Instruments

TMS 320C8x Series - Crossbar

- Allows 5 instruction fetches and 10 parallel data accesses per cycle
  - => shared memory system
- Two-stage pipelined crossbar access (two machine cycles):
  - Stage 1: Request from processor/transfer controller(TC) to access a certain memory (address MSB transfer) + granting signal to one of the requesting processors / TC.
  - Stage 2: address LSB transfer + access of requested memory
- Access decisioning
  - Round robin
  - Priority



Texas Instruments

TMS 320C8x Series - Master Processor

- 32-bit RISC processor with IEEE-754 FloP unit.
- Special set of parallel multiplication, addition, load, and store instructions -> 100 MFLOPS
- FloP unit operations use the same register file as the integer and logic unit.
- 32-bit single precision and 64-bit double precision:
  - Full double precision add-unit;
  - FloP multiply unit is supported with microcode to provide single- and double-precision operations.
- MP Instructions:
  - Integer add and substract, logical, compare;
  - FloP arithmetic & conversion, vector FloP aritmetic, vector FloP multiply/add/subtract, vector FloP conversion, double precision FloP accumulators in vector instructions;
  - Branch unconditionally (+compare to zero, on bit), call functions, subroutines and return;

Texas Instruments

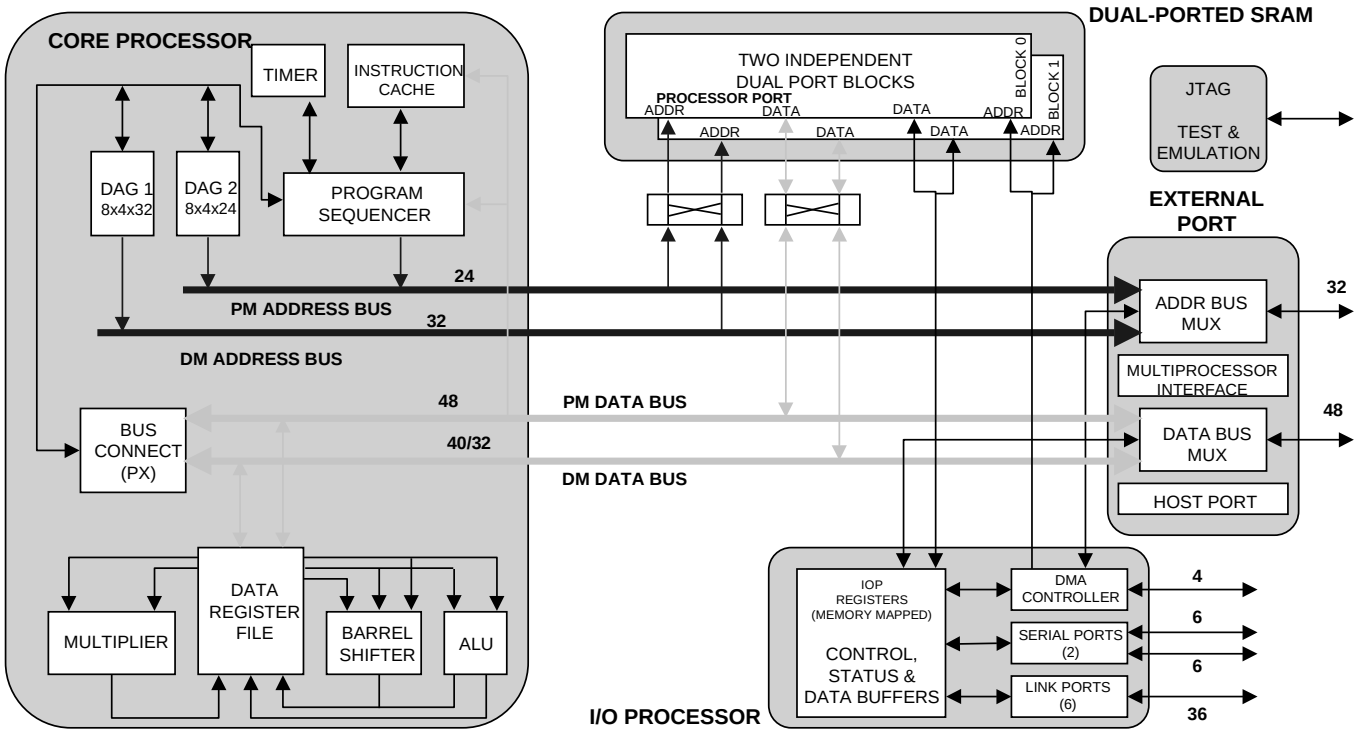
TMS 320C8x Series - ADSPs

- Basic DSP architecture
- Separate address units for global and local memory
- ALU data path
  - subword parallelism
  - 32-bit three-input ALU
  - barrel rotator, mask generator, multiple flags expander, leftmost-one, rightmost-one ...
- 16-by-16-bit multiplier
- However:
  - no AGUs/ACUs presented
  - no FIFO
- DSP link concept is not available: communication between DSPs must take place via memory.

Analog Devices

SHARC series - Architecture

http://archi.enssat.fr



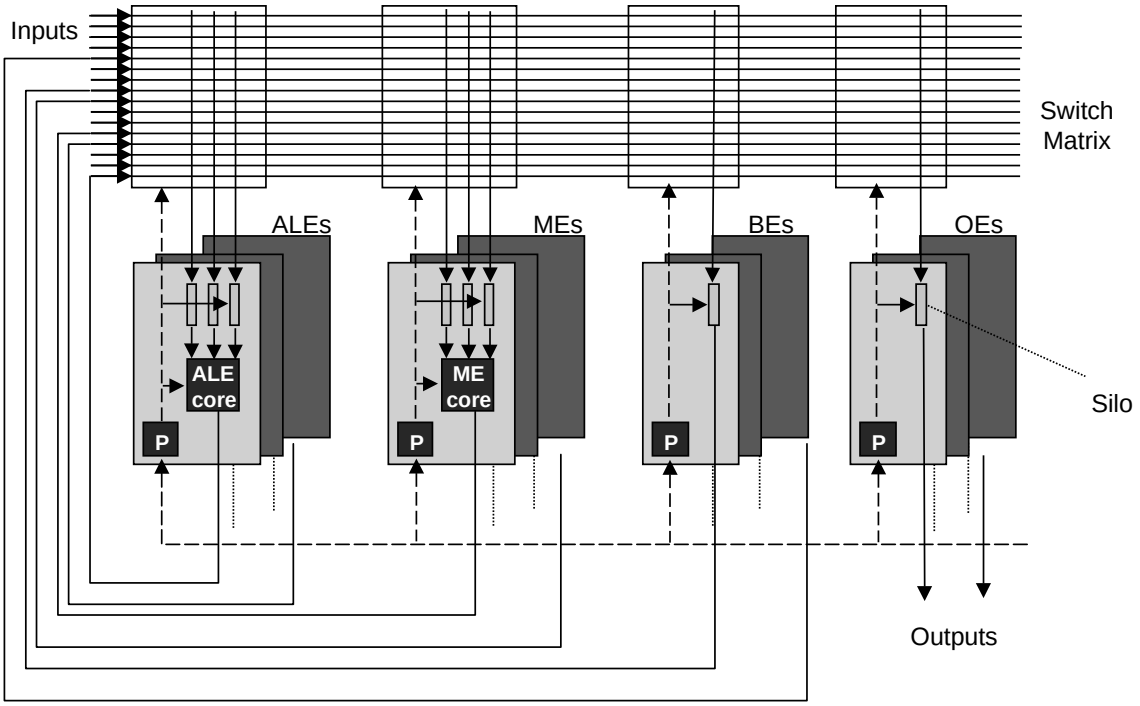
Analog Devices

SHARC series - Features

http://archi.enssat.fr

- Super Harvard ARChitecture: 32-bit FloP core
- 25ns instruction rate, 40 MIPS, 120 MFLOPS
- 4 Mbits on-chip SRAM: dual ported for independent core and I/O access
- 4 Gwords off-chip addressable
- Scalable multi-processing: glueless connection -> six 40 Mbytes/sec link ports
- Two 40 Mbits/s serial ports - asynchronous, full-duplex
- JTAG debugging support
- 3.3/5 V
- Price/1000: $80-$430

Philips VSP 1/2

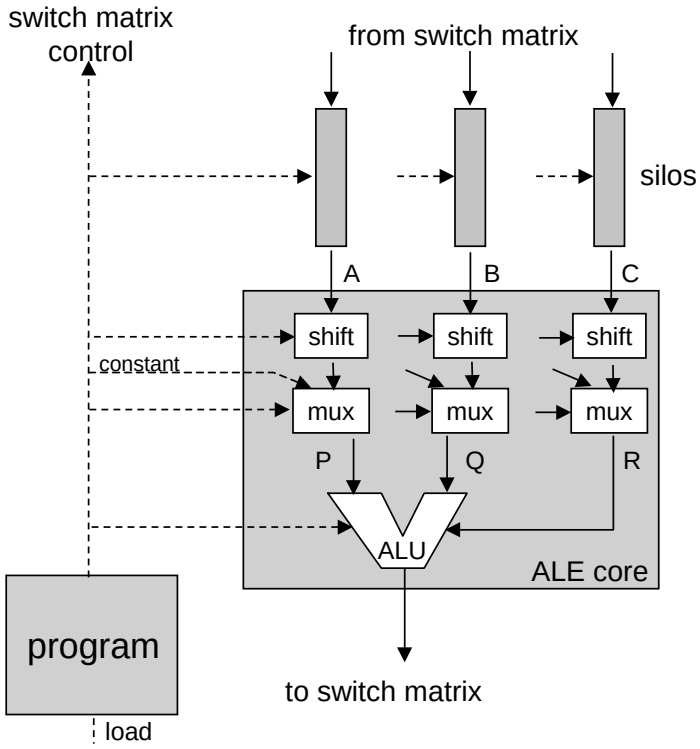


Philips VSP 1/2

- 12-bit Video Signal Processor (VSP)
- Processing Elements (PEs)
  - Arithmetic and logic element (ALE)
  - Memory element (ME)
  - Buffer element (BE)
  - Output element (OE)
- Local program memory for each PE
- Unconstrained programmable communication between all elements in a single cycle
- Silos = programmable delay elements used to store intermediate values (32 12-bit words)

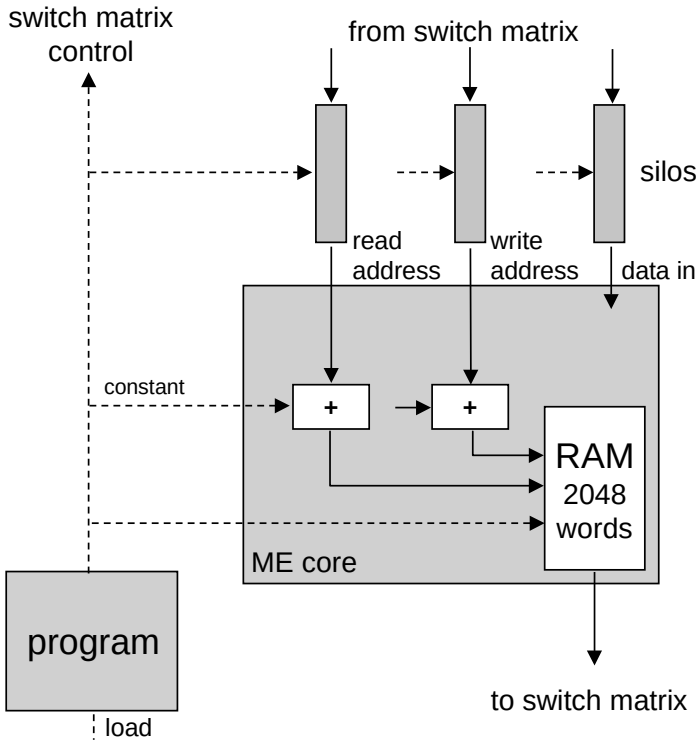
Philips VSP 1/2- ALE Core

- Barrel shifters
- Multiplexers for constant input
- ALU
- Booth encoded multiplication steps
  - allows multiplications without a costly fully parallel multiplier



Philips VSP 1/2- ME Core

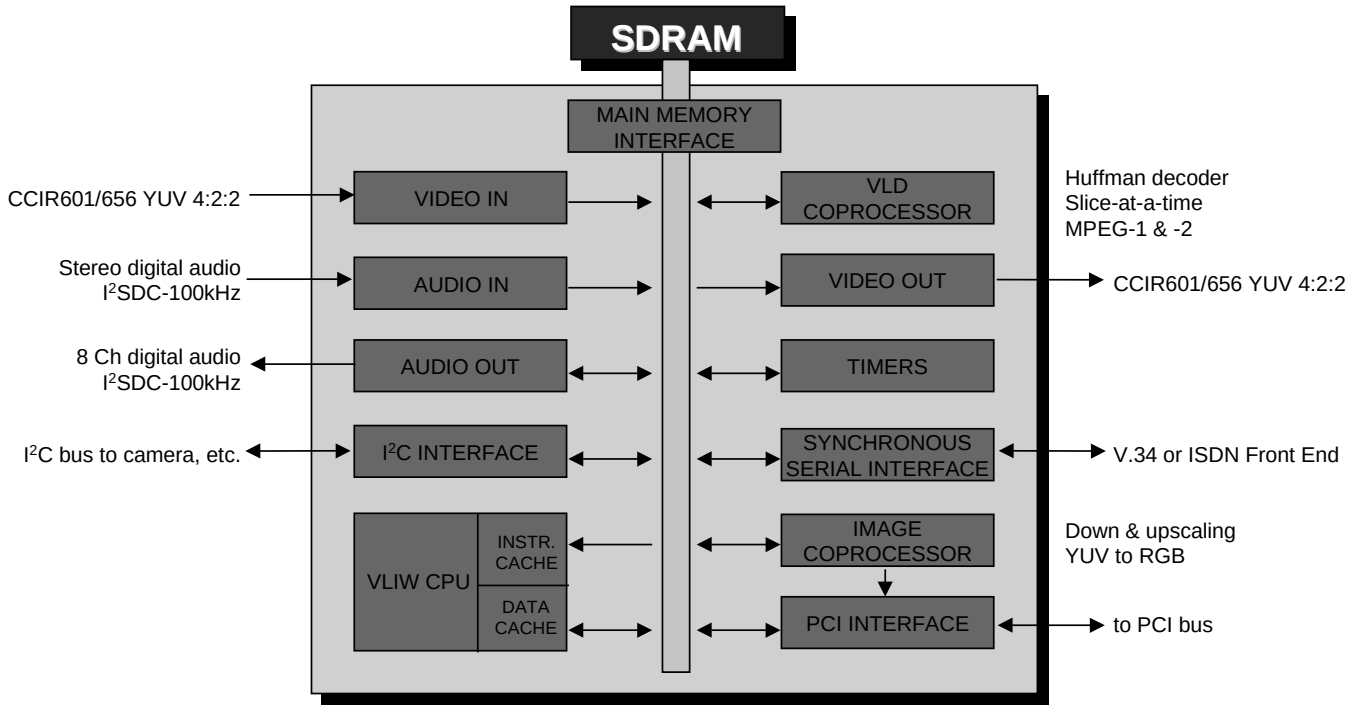
- 2048 12-words RAM
- Logic for address calculation
- Look Up Table (LUT)
- BEs
  - additional storage of intermediate results
- OEs
  - buffering
  - chip output



Philips VSP 1/2- Specs

	VSP1	VSP2
Technology	1,2µ CMOS	0,8µ CMOS
Chip size	90 mm^2	156 mm^2
Transistors	206 000	1 150 000
Package	QPF160, PGA 176	QPF208
Dissipation	1W	< 5W
Clock	27 MHz	54 MHz
Word size	12 bit	12 bit
ALEs	3	12
MEs	2	4
Size MEs	512 x 12	2k x 12
Memory style	single port	dual port
BEs	0	6
Inputs	5	6
Outputs	5	6

Philips TriMedia

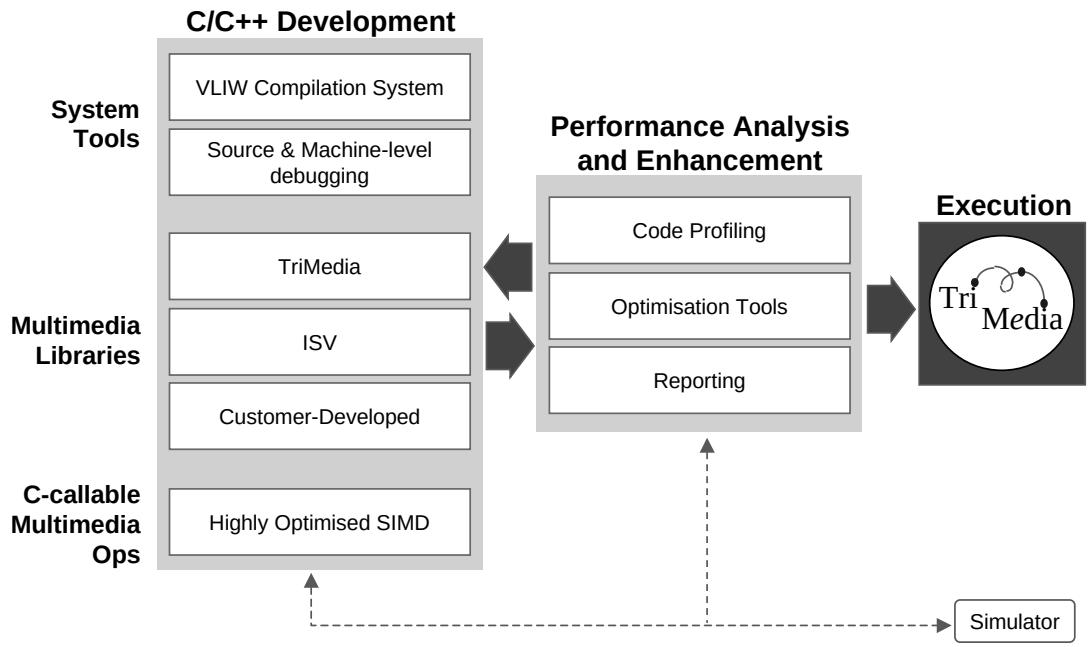


Philips TriMedia - Features

- Concurrently processes audio, video, graphics and communications data "for a true-to-life multimedia experience" (according to Philips ...)
- 4 GOPS VLIW CPU
  - 100 MHz clock rate with 5 ops/cycle
  - 32-bit lineair address space
  - 128 32-bit general purpose registers (15 read and 5 write ports) and 27 functional units
  - Guarded execution avoids branch operations
- On-chip audio, video and graphics I/O and accelerator peripherals
- 132 MB/second PCI bus interface
- Memory subsystem
  - Dedicated 32-bit, 400 MB/sec bus interface with SDRAM
  - 32k instruction cache, 16k dual-ported cache
  - 0.50 - 0.35µ CMOS, 3.3V, 4 Watt, 163 active pins, 4 million transistors (TM1000 series)

Philips TriMedia

Software Development Environment



Philips TriMedia - Applications

- MPEG-2, full DVD compliance: 74%
- Dolby Digital AC3
  - 6 Channel: 26 %
  - 2 Channel: 19%
- H.324 Videophone, full options, 15 fps (includes V.34 / V.80): 89 %

II. Critères de choix

1. **Format des données**
2. **Vitesse**
3. **Coût**
4. **Énergie dissipée**
5. **Outils de développement**
6. **Multiprocesseur**

Format Arithmétique

- Arithmétique virgule fixe
  - Les nombres sont représentés entre -1.0 et +1.0, les calculs sont identiques aux entiers.
  - Les signaux doivent être recadrés (*scaling*) pendant le traitement pour assurer que les nombres restent dans le rang dynamique.
  - La précision peut être aussi bonne qu'en flottant.
  - Les DSPs fixes peuvent émuler l'arithmétique flottante.
    - *Block Floating Point* (exposant fixe) plus efficace
- Arithmétique virgule flottante
  - Les nombres sont représentés comme mantisse * $2^{exponent}$, la mantisse est entre -1.0 et +1.0.
  - Pas de problème de rang dynamique et de précision.

Virgule Fixe ou Flottante ?

- Virgule FIXE
  - Précision entre [-1 et 1] mais problèmes de débordement
  - Plus efficace (MIPS/W), plus rapide et moins cher
  - Consommation moins importante
  - Temps de développement plus long -> NRE
  - Plutôt application grand public, grande diffusion

C50 (50/35/25ns) (2 I/O série) (9K RAM) PQFP 132
50 ns : PU 327 FF; 233 FF (100 Pièces)
25 ns : PU 396 FF; 282 FF (100 Pièces)

- Virgule Flottante
  - Temps de développement plus rapide
  - Plus grande portabilité, compilateur plus efficace , pas de recadrage
  - Pas de débordement : dynamique de 3.4 1038 # 1500dB!
  - Plus cher, Consomme plus

C40 (50/40/33ns) (6 I/O parallèle 20Mbytes) (2K RAM) PGA 325
50 ns : PU 1795 FF; 1282 FF (100 Pièces)
33 ns : PU 1974 FF; 1410 FF (100 Pièces)

Data Width

- Basic Rule: **size increases** => number of package pins & size of external memory devices increase => **cost increases**.
- FloP DSPs use a 32-bit data word
  - = single precision
  - Some processors have a extended precision mode: e.g. 40 bit (ADI's SHARC series)
- FixP DSPs commonly use 16-bit words
  - Motorola DSP5600x and DSP563xx families: 24-bit data words;
  - Zoran's ZR3800x family: 20-bit data word.
  - 3 possibilities to meet demanded data width:
    - equal or less than the one of the available processor -> OK;
    - larger: use other processor or string together an appropriate combination of instructions to reach double precision
    - trade-off: speed falls down

[Bier97]

Choix du *bon* DSP : Vitesse

- Métrique : temps requis pour réaliser un certain nombre de traitements
  - MIPS (*Millions of Instructions Per Second*)
    - Attention : le traitement réalisé par une instruction peut varier fortement d'un processeur à un autre (instructions très spécialisées)
    - Valable si les architectures à comparer sont très similaires (RISC)
  - MOPS (*Millions of Operations Per Second*)
    - Problème identique
  - MACS (*Multiply-ACcumulate operations per Second*)
    - Plus réaliste dans le cas de DSP, il reste cependant à évaluer les mouvements mémoire et les traitements avant et après l'instruction MAC
- Solution : *Algorithm Kernel Benchmarking*

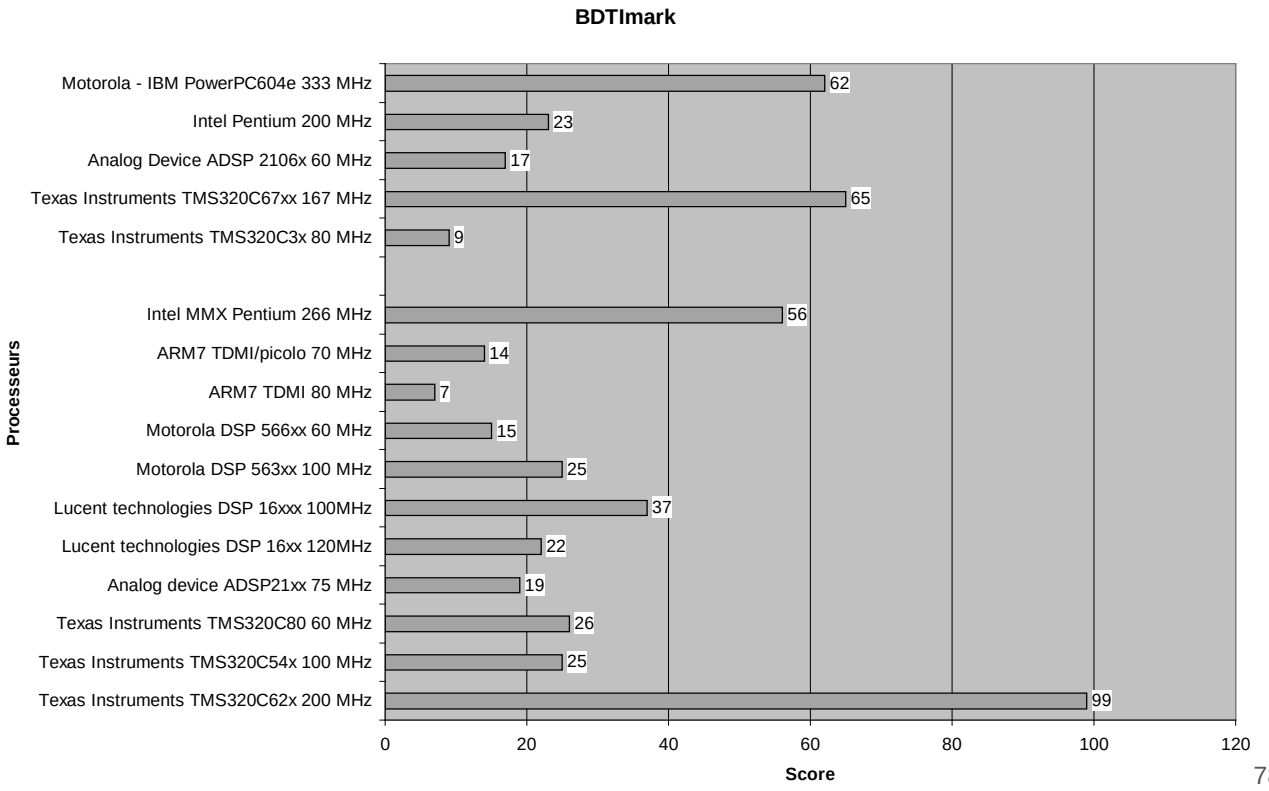
BDTIMARK

http://www.bdti.com

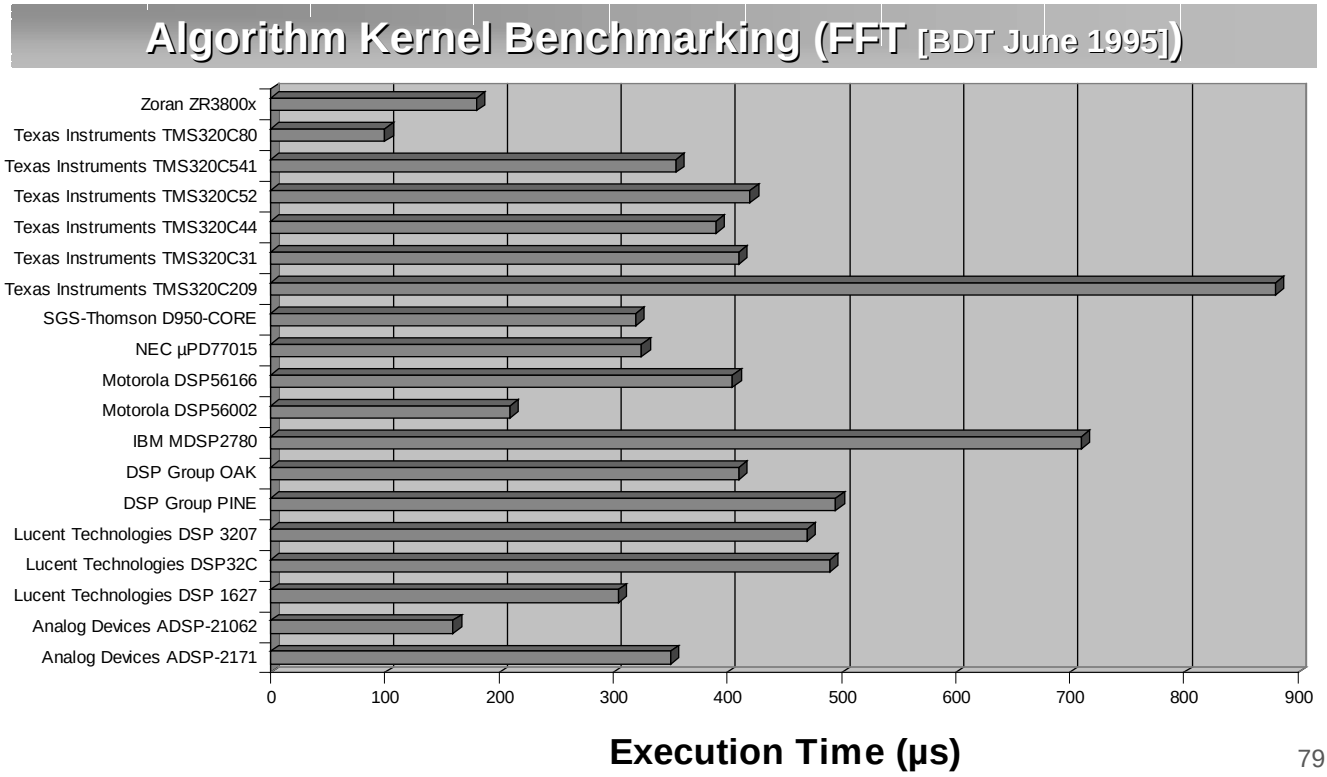
Function	Description	Application Examples
Real block finite impulse response (FIR) filter	FIR filter that operates on a block of real (not complex) data	G.728 speech encoding, other speech processing
Complex block FIR filter	FIR filter that operates on a block of complex data	Modem channel equalisation
Real single-sample FIR filter	FIR filter that operates on a single sample of real data	Speech processing, general filtering
Least-mean-square adaptive FIR filter	LMS adaptive FIR filter that operates on a single sample of real data	Channel equalisation, servo control, linear predictive encoding
Infinite inpulse response (IIR) filter	IIR filter that operates on a single sample of real data	Audio processing, general filtering
Vector dot product	Sum of the pointwise multiplication of two vectors	Convolution, correlation, matrix multiplication, multidimensional signal processing
Vector add	Pointwise addition of two vectors producing a third vector	Graphics, combining audio signals or images, vector search
Vector maximum	Discovery of the value and location of a vector's maximum value	Error-control coding, algorithms using block floating-point arithmetic
Convolutional encoder	Application of convolutional forward error-correction code to a block of bits	North American digital cellular telephone equipment (IS-54 standard)
Finite-state machine (FSM)	A contrived series of control operations (test, branch, push, pop) and bit manipulations	Control operations appear in nearly all digital signal processing applications
256-point, radix-2, in-place fast Fourier transform (FFT)	FFt conversion of a normal time-domain signal into the frequency domain	Radar, MPEG audio compression, spectral analysis

BDTIMARK

http://archi.enssat.fr

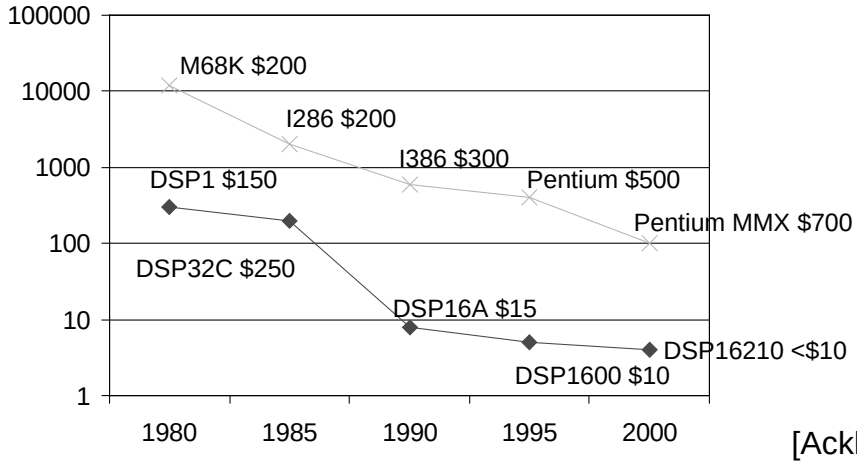


Choix du *bon* DSP : Vitesse



Choix du *bon* DSP : Coût

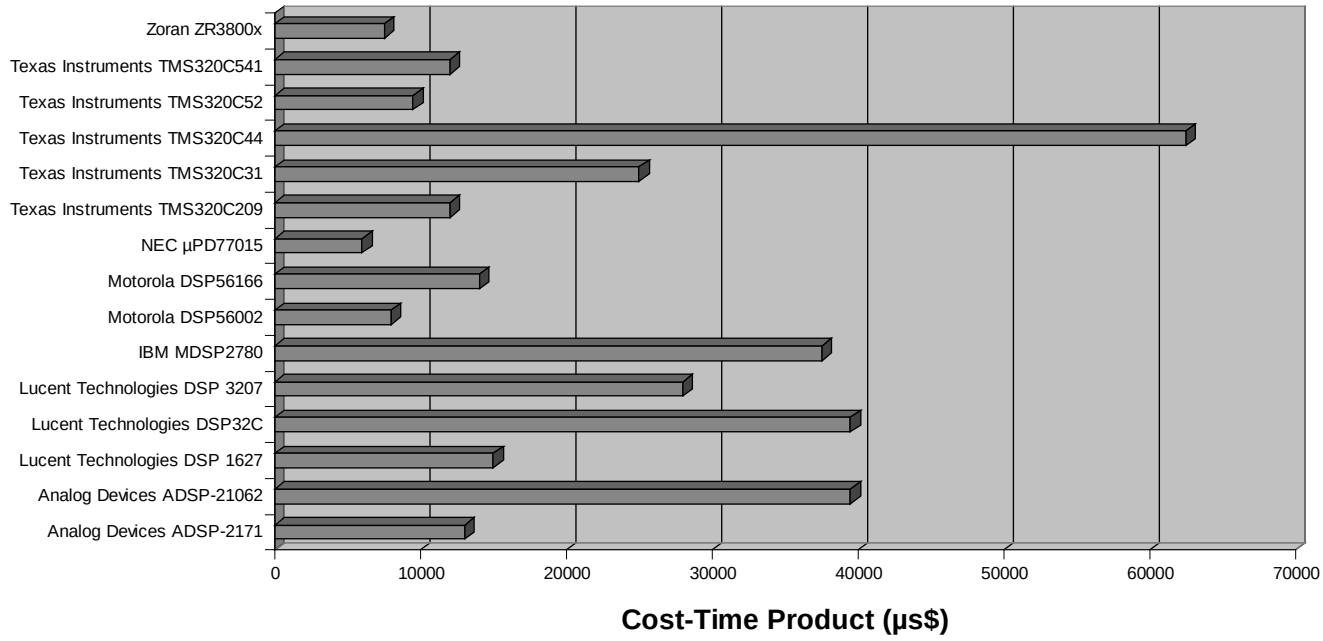
- Important pour une production a fort volume
  - Faible coût
    - Les DSPs sont efficaces en MIPS/mW et en MIPS/mm^2
    - Cela se paye sur le flexibilité et la programmabilité
    - *Packaging*, quantité, format arithmétique



[Ackland ISLPD98]

Choix du *bon* DSP : Coût

Algorithm Kernel Benchmarking (FFT [BDT June 1995])



Organisation de la mémoire

- Harvard architecture linked to memory bandwidth
- Size of supported on- and off-chip memory
  - FixP DSPs
    - small-to-medium on-chip memories (256 - 32K words)
    - small external data busses
      - e.g. AD, AT&T, Motorola and TI: 16bit or less address busses
  - FloP DSPs
    - little (or no) on-chip memory
    - large external busses
      - e.g. AD ADSP-21020: no on-chip memory, 32-bit external data address bus + 24-bit external program bus.
      - e.g. TI TMS320C30: 6K words on-chip, one 24-bit and one 13-bit external address bus
      - Both chips provide instruction caches
      - exception: ADSP-21060: 4Mbits on-chip memory, configurable in various combinations (e.g. 64K 32-bit words and 40K 48-bit words)
- Trade-off between application area and performance

Choix du bon DSP : énergie

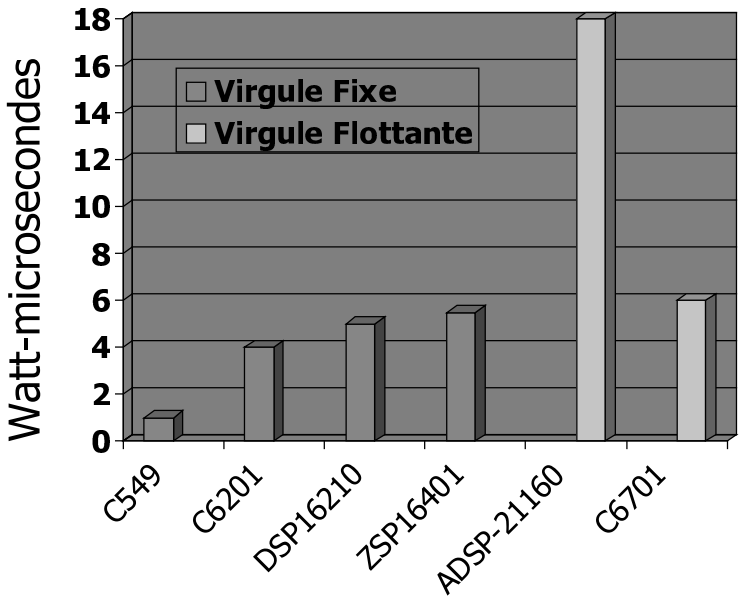
- Gestion de la puissance
  - Opération à basse tension d'alimentation
  - Modes "Sleep" ou "Idle"
  - Gestion de l'horloge programmable : PLL, *clock dividers*
  - Contrôle des périphériques
- Attention aux données *typiques* ou *maximales*

DSP	MACS	CMOS	Actif	Veille
16210	200	0.35u	325mW	5.4mW
1609	80	0.3u	265mW	0.5mW
RISC uC	-	0.35u	93mW	75uW

Lucent DSPs à 3.3 Volts [Ackland ISLPD98]

Énergie

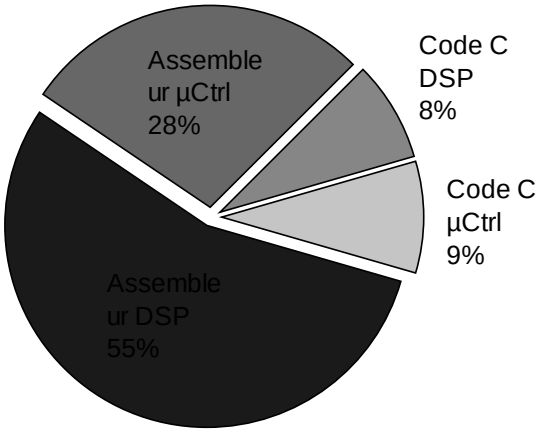
- Filtrage numérique RIF



- C549
  - 100 MHz, 2.5V
- C6201
  - 200 MHz, 1.8V
- DSP16210
  - 100 MHz, 3.3V
- ZSP16401
  - 200 MHz, 2.5V
- ADSP-21160
  - 100 MHz, 2.5V
- C6701
  - 167 MHz, 1.8V

Choix du bon DSP : Outils

- Compilateur C vs développement assembleur
  - Facteur 4 à 10 pour les DSP virgule fixe
  - Facteur 2 à 4 en virgule flottante
- Critères de sélection des outils et du langage
  - Compilateurs : taille et vitesse du code
  - Simulateurs : vitesse
  - 75-90% du code est écrit en assembleur



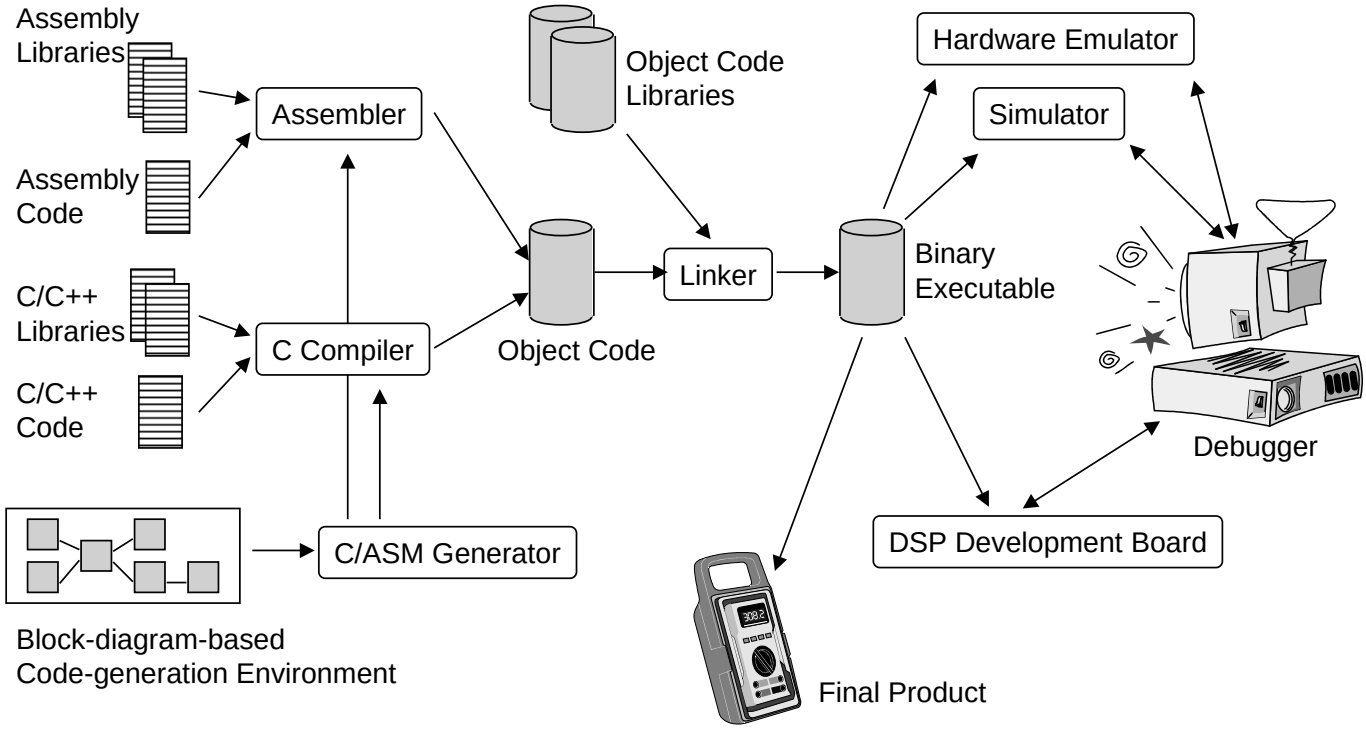
DSPstone : C54x

- *Overhead* de la version compilée sur celle optimisée à la main en assembleur

Benchmark	Overhead Clock cycles %	Program Memory %	Data Memory %
N Real Update	373	1306	53
N Compx Update	137	882	26
Dot Product	620	1309	680
Matrix	24	414	5
Convolution	1076	1545	100
FIR	1045	769	97
FIR 2D	427	743	158
IIR N Biquad	140	764	55
LMS	406	-35	78

[Ropers99]

Outils de développement



[Bier97]

Outils de développement

- Spécifications flots de données du TS, prototypage rapide, simulation, validation
  - Matlab/Simulink, Cadence SPW, Synopsys Cossap, Ptolemy
- Outils de profiling
- Compilateurs depuis des langages de *haut-niveau*
  - DSP en virgule fixe ou flottante
- Optimisations de l'assembleur

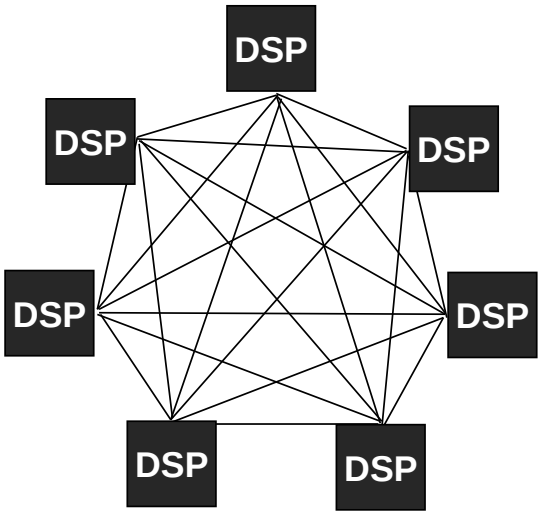
Application Development

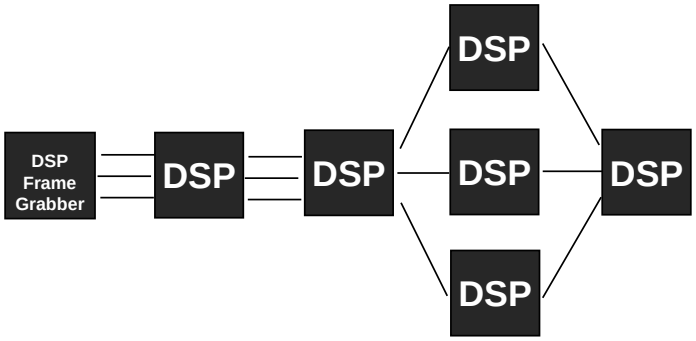
- Debugging
  - Instruction set simulators
    - Debugging before hardware is available
    - High-level languages: check compatibility with simulators
  - Hardware emulation
  - On-chip debugging/emuation
    - IEEE 1179.1 JTAG standard
      - scan-based emulation
      - be careful
  - DSP development boards
    - Real-time checking of algorithms
    - Low-production-volume systems

[Bier97]

Choix : multiprocesseur

- Beaucoup d'applications demandent l'utilisation de plusieurs DSP
  - Ports d'interconnexion, protocoles





III. Évolution des DSPs

1. DSPs conventionnels améliorés
2. Capacités SIMD
3. DSPs hybrides MCUs
4. VLIW
5. Superscalaire
6. Architectures clusterisées
7. Comparaison de performances

Évolution des DSPs

- Améliorer les performances au delà de l'augmentation liée à la vitesse d'horloge ?
- Augmenter le parallélisme
  - Augmentation du nombre d'opérations exécutées dans chaque instruction
  - Augmentation du nombre d'instructions exécutées par cycle d'horloge
- Contrôle

Nouvelles architectures DSP

- DSPs conventionnels améliorés
  - UT multiples, SIMD, coprocesseurs
  - Lucent DSP16xxx, ADI ADSP-2116x
- DSPs VLIW
  - TI C6xxx, Infineon Carmel
- DSPs superscalaires
  - ZSP 164xx
- Processeurs généralistes ou hybrides
  - GPP+SIMD, μC/DSP
  - PowerPC/AltiVec, Pentium/MMX
  - Infineon TriCore, SHx, ARM Piccolo, STx, SHx

Plus d'opérations par instruction

- Comment augmenter le nombre d'opérations qui peuvent être exécutées dans chaque instruction ?
  - Ajouter des unités d'exécution
    - multiplieur, additionneur, ...
    - jeu d'instruction à enrichir
    - taille de l'instruction à augmenter
    - bus de transfert mémoire à augmenter
  - Augmenter les capacités SIMD
- Dans le même ordre d'idées :
  - Utiliser du matériel très spécialisé
  - Utiliser un coprocesseur
  - Architectures hybrides DSP/MCU

Avantages et inconvénients

Méthode	Avantages	Inconvénients
Augmenter le nombre d'UE	Augmentation limitée de la consommation, du coût et de la densité de code Compatibilité maintenue	Augmentation importante de la complexité Difficulté à programmer et à compiler Perspectives limitées
Augmenter les capacités SIMD	Gain en performance très important pour des traitements par blocs Modifications architecturales limitées	Nécessité d'avoir un parallélisme au niveau des données important Consommation mémoire importante
Matériel dédié	Gain en performance important	Nécessite une bonne connaissance de l'application

Architecture hybride DSP/MCU

- Associer des fonctionnalités MCU
  - gestion efficace des ITs
  - manipulation de bits
  - exécution conditionnelle
- A des fonctionnalités DSP
  - calcul intensif
- Pour obtenir un code performant et efficace afin de limiter la taille du code et donc du circuit

Méthodes de couplage

- Utiliser plusieurs processeurs sur le même support
- Utiliser un coprocesseur
- Étendre les capacités DSP des microcontrôleurs
- Étendre les capacités de contrôle des DSPs
- Créer une architecture hybride DSP/Microcontrôleur

Suite

Architecture hybride DSP/MCU

- Méthodes de couplage

Méthode	Avantages	Inconvénients
Multiprocesseur	2 jeux d'instructions ≠ Les 2 cœurs travaillent en // Pas de conflits de ressources	Duplication de ressources Deux outils de développement
Coprocesseur	Les 2 cœurs travaillent en //	Modèle de programmation plus complexe Transferts de données
Extension	Modèle de programmation plus simple	Contraintes imposées par l'architecture initiale
Solution Hybride	Architecture plus "propre"	Développement plus risqué

Plus d'instructions par cycle

- L'objectif est ici de profiter du parallélisme au niveau instruction d'une application
- Deux techniques sont bien connues :
  - VLIW : empaquetage de plusieurs instructions de type RISC dans une seule "super-instruction"
  - Superscalaire : exécution parallèle de plusieurs instructions sélectionnées dynamiquement par le processeur

Avantages et inconvénients

Méthode	Avantages	Inconvénients
VLIW	Grand bon dans les performances Architectures plus orthogonales ⇒ meilleures cibles pour les compilateurs	Bande passante vers la mémoire importante Forte consommation Séquencement délicat Augmentation de la taille du code importante
Superscalaire	Grand bon dans les performances Architectures plus orthogonales ⇒ meilleures cibles pour les compilateurs Pas de problèmes de séquencement	Bande passante vers la mémoire importante Plus forte consommation Séquencement délicat Temps d'exécution difficilement prédictible

III. Évolution des DSPs

1. **DSPs conventionnels améliorés**
2. **Capacités SIMD**
3. **DSPs hybrides MCUs**
4. **VLIW**
5. **Superscalaire**
6. **Architectures clusterisées**
7. **Comparaison de performances**

DSPs conventionnels améliorés

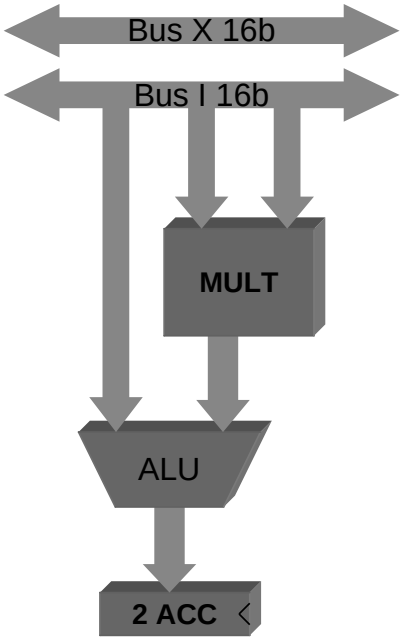
- Chemin de données multi-opérations
  - e.g. 2 multiplieurs-additionneurs
  - Instructions à redéfinir (jeu, taille)
  - Bande passante vers la mémoire
- Matériel spécialisé embarqué
  - *Application-oriented operations*
- Coprocesseurs
  - Décodage de Viterbi, codage de Huffman, filtrage FIR

Lucent DSP 16xx

- Filtrage FIR sur DSP conventionnel

```
DO N TIME:
   acc=acc+P
// P=X*Y
// Y=*r0++
// X=*r1++
```

Exécution en N cycles

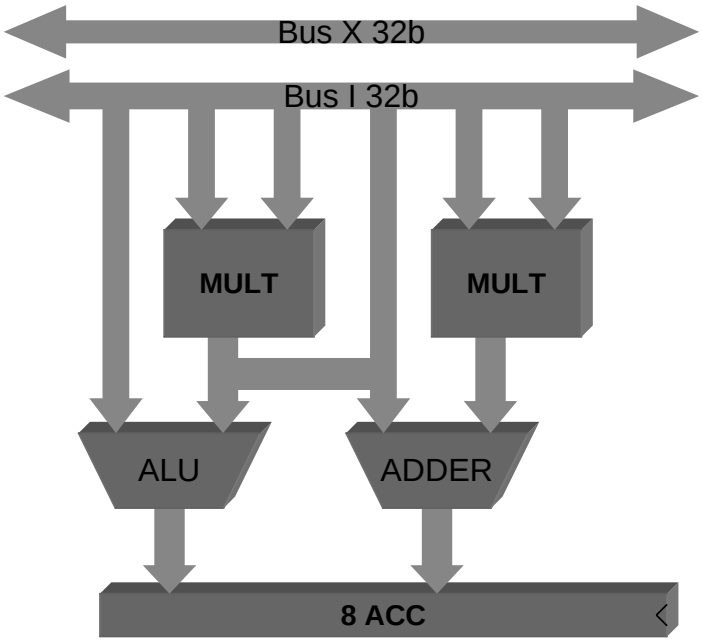


Lucent DSP 16xxx

- Filtrage FIR sur DSP conventionnel amélioré
  - Architecture Dual-MAC
    - 400 MIPS, 380mW
    - $5 - $80

```
DO N/2 TIME:
   acc=acc+P0+P1
// P0=X0*Y0
// P1=X1*Y1
// Y=*r0++
// X=*r1++
```

Exécution en N/2 cycles



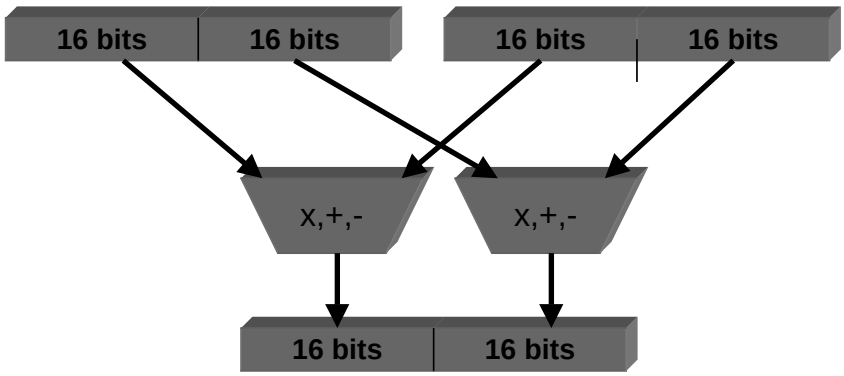
Retour

III. Évolution des DSPs

1. DSPs conventionnels améliorés
2. **Capacités SIMD**
3. **DSPs hybrides MCUs**
4. **VLIW**
5. **Superscalaire**
6. **Architectures clusterisées**
7. **Comparaison de performances**

Capacités SIMD

- Opérations parallèles sur différentes largeurs de chemins de données (16 bit, 8 bit, ...)
  - *Split* unités d'exécution
  - Unités d'exécution multiples
- Exemples
  - Lucent DSP16xxx, ADI ADSP-2116x, ADI TigerSHARC



Retour

III. Évolution des DSPs

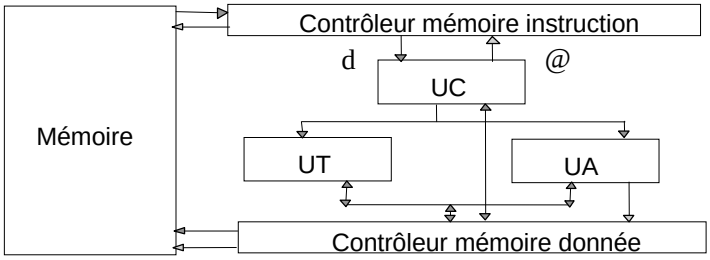
1. DSPs conventionnels améliorés
2. Capacités SIMD
3. **DSPs hybrides MCUs**
4. **VLIW**
5. **Superscalaire**
6. **Architectures clusterisées**
7. **Comparaison de performances**

Architecture Hybride DSP/MCU

http://archi.enssat.fr

- Objectif : associer
  - les hautes performances des DSPs
  - à l'efficacité du code microcontrôleur (MCU)
- ST100 (ST Microelectronics)
- TriCore (Infineon Technologies)
  - 32 bits MCU / 16 bits DSP
  - associe des fonctionnalités DSP :
    - MAC, architecture mémoire Harvard, modes d'adressages complexes
  - et microcontrôleur :
    - architecture load/store, large espace d'adressage, code compact

Points communs ST100/TriCore



- Mémoire de 4 Gø unifiée. Contrôleur de mémoire de donnée et d'instruction séparés pour autoriser l'emploi de mémoires cache.
- Seuls les instructions de *load* et de *store* accèdent à la mémoire, tous les traitements se font sur des données stockées en registre.
- Cœur constitué de 3 unités, pour le contrôle, le traitement et la génération d 'adresses.

Le Traitement

	ST100	TriCore
registres	N_1x 32 bits	16 x 32 bits
UAL	N_2x 40 bits = $2N_2$x16 bits	1 x 32 bits = 2 x 16 bits
Multiplieur	N_3 x 17 b	2 x 16 bits
SIMD	8, 16 bits	8, 16 bits
Autres	Registre en barillet 40 bits	saturation, support pour division, détection d'exposant

L'unité d'adresse

- Mémoire unifiée de 4Gø, possible intégration de mémoires cache de données et d'instructions
- Adressage modulo et bit-reverse
- Interruptions vectorisées et hiérarchisées
- Tricore
  - Mémoire segmentée (16 x 256 Mø)
  - 16 registres d'adresses
  - Bus de 128 bits entre le cœur et la mémoire intégrée
- ST100
  - Registres d'adresses 32 bits
  - Sauvegarde de contexte en pile

Le jeu d'instructions (1)

- Contrôle + calcul intensif
- Exécution conditionnelle
- TriCore
  - 2 jeux d'instructions : 32 bits et 16 bits, le second étant un sous-ensemble du premier
  - SIMD
  - Support Viterbi, division

Le jeu d'instructions (2)

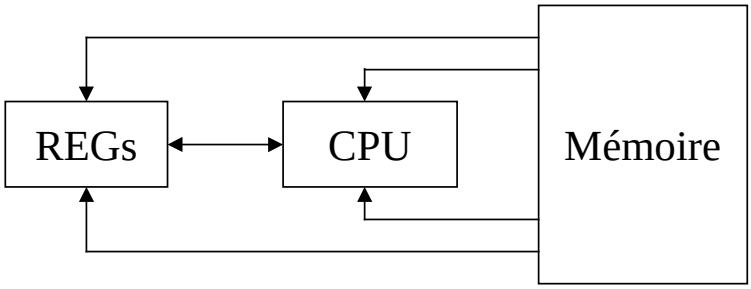
- ST100
  - 3 jeux d'instructions
  - GP16 :
    - de type RISC, très dense
  - GP32 :
    - extension du jeu 32 bits
    - divisé en trois classes : les références mémoires, les instructions arithmétiques et les instructions de contrôle
    - SIMD
  - SLIW :
    - 4 instructions GP32
    - fonction des unités fonctionnelles intégrées

Le contrôle

- Tricore
  - 64 bits d'instructions sont recherchés à chaque cycle et décodés puis exécutés dans l'un des trois pipelines (calcul d'adresse, traitement, gestion de boucle), au total 4 étages de pipeline.
  - Exécution superscalaire x 3.
- ST100
  - Selon l'architecture (ressources et bande passante), 128 bits d'instructions peuvent être lus et décodés simultanément. Chaque instruction décodée est ensuite dispatchée sur l'unité associée pour suivre le pipeline de chacune d 'elle.
  - Exécution superscalaire x2 en GP16 et GP32, VLIW x4 en SLIW

Le TMS320C27x

- DSP virgule fixe 16 bits
- Alternative à l'association DSP-Microcontrôleur, reprend l'architecture de la famille C2xx en cherchant à obtenir un code plus efficace.
- Architecture Harvard (modifiée) pour des raisons de compatibilité



Architecture du coeur

- Architecture Harvard modifiée organisée autour de quatre modules
  - Logique de contrôle
  - Génération d'adresse programme
  - Génération d'adresse donnée
  - Unité de traitement

- Connectés à la mémoire via six bus
  - 3 bus d'adresse
  - 3 bus de données

Le jeu d'instructions

- 90 % des instructions codées sur 16 bits
- Opération registre-registre
- Exécution conditionnelle
- Instruction *repeat*
- Instructions de calculs en un cycle
- Temps d'exécution des instructions de contrôle variable

Le contrôle

- Pipeline de 8 étages

Fetch1	Fetch2	Decod 1	Decod 2	Read 1	Read 2	Exec	Write-Back

- Lecture des instructions par 32 bits
- 32 ITs vectorisées et hiérarchisées, sauvegarde automatique de contexte par passage en pile

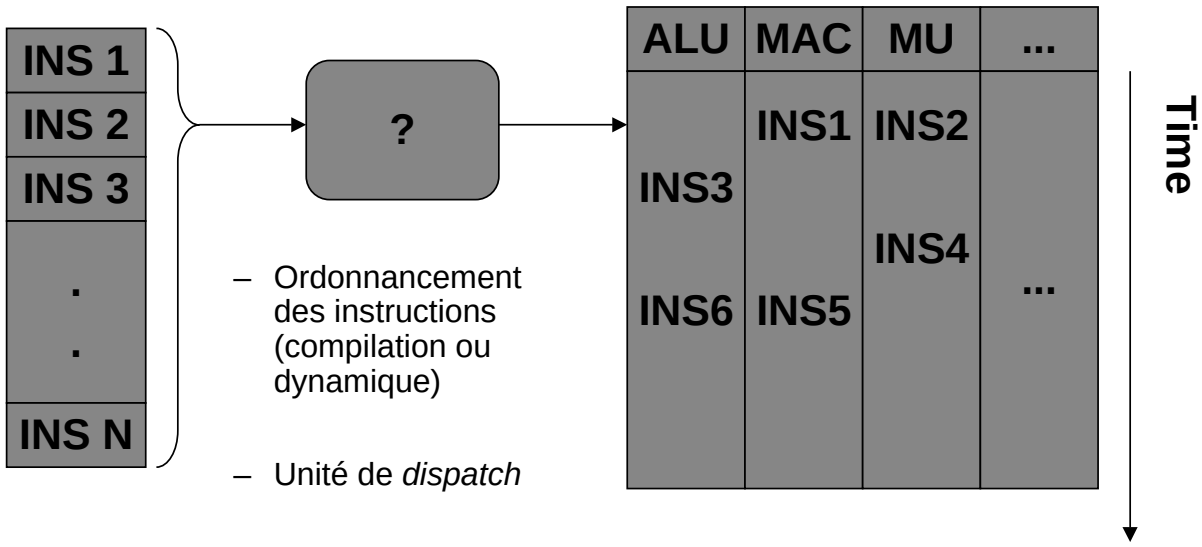
Retour

III. Évolution des DSPs

1. DSPs conventionnels améliorés
2. Capacités SIMD
3. DSPs hybrides MCUs
4. **VLIW**
5. **Superscalaire**
6. **Architectures clusterisées**
7. **Comparaison de performances**

Superscalaire vs VLIW

http://archi.enssat.fr



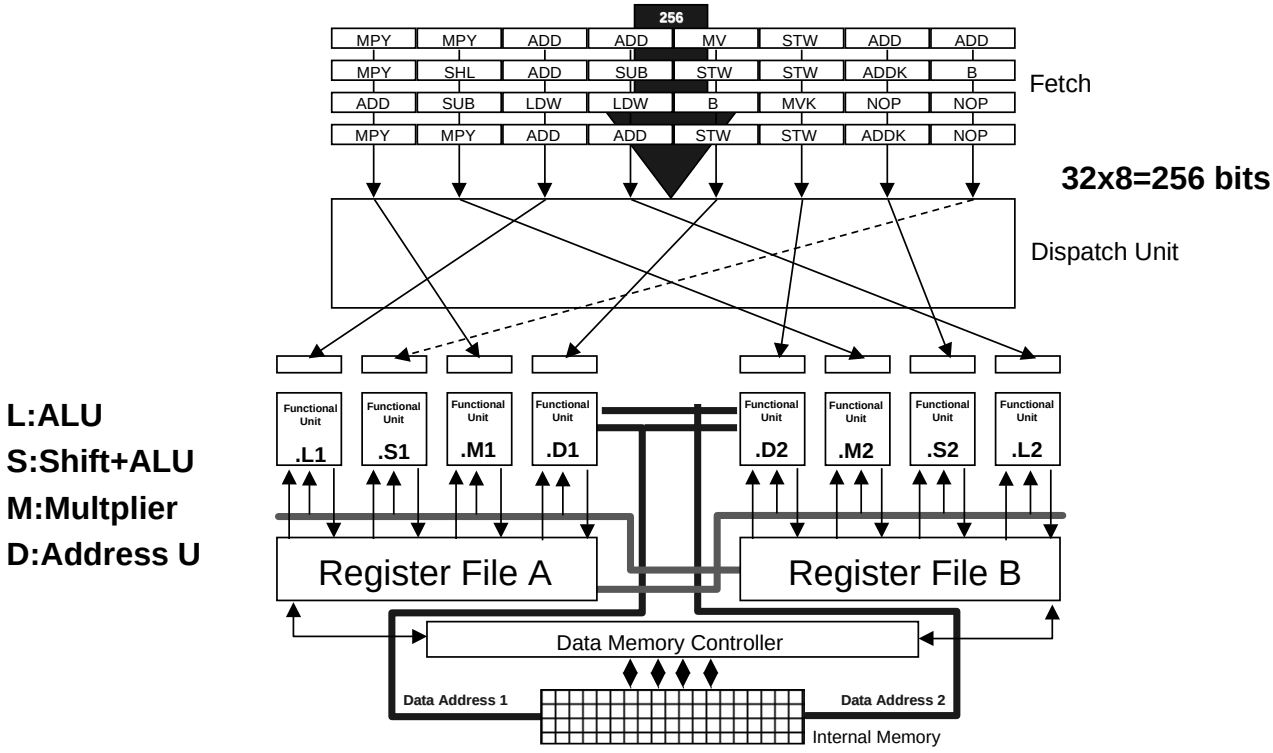
– Ordonnancement des instructions (compilation ou dynamique)

– Unité de *dispatch*

Very Long Instruction Word

- Caractéristiques
  - Plusieurs instructions par cycle, empaquetées dans une "super-instruction" large
  - Architecture plus régulière, plus orthogonale, plus proche du RISC
  - Jeu de registres uniforme, plus large
- Exemples
  - TI TMS320 C6xx
  - Infineon Carmel
  - ADI TigerSHARC
  - StarCore SC140 (Lucent + Motorola)

VLIW : C62xx

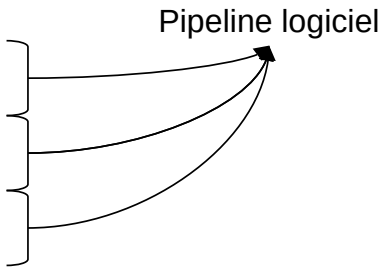


L:ALU
S:Shift+ALU
M:Multplier
D:Address U

VLIW C6x

- Filtrage RIF sur VLIW C6x

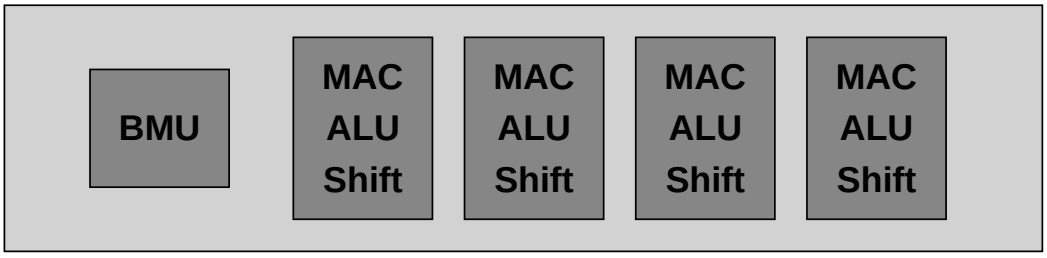
```
LOOP
   ADD .L1 A0,A3,A0
// ADD .L2 B1,B7,B1
// MPYLH .M1X A2,B2,A3
// MPYLH .M2X A2,B2,B7
// LDW .D2 *B4++,B2
// LDW .D1 *A7--,A2
// [B0] ADD .S2 -1,B0,B0
// [B0] B .S1 LOOP
```



Exécution en N/2 cycles

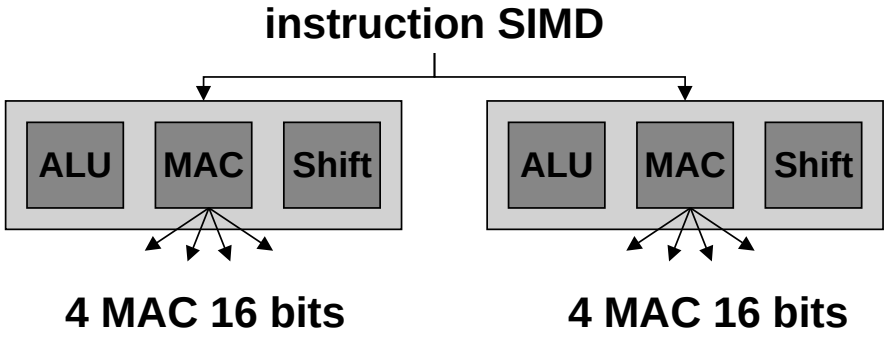
StarCore SC140 core

- Processeur VLIW 16 bits développé en commun par Lucent et Motorola
- 300 MHz, faible tension
- Optimisé pour faible consommation
  - Meilleure densité de code (16 bits) que C6x
  - Pipeline plus simple (5 étages contre 11), latences courtes



VLIW combiné au SIMD

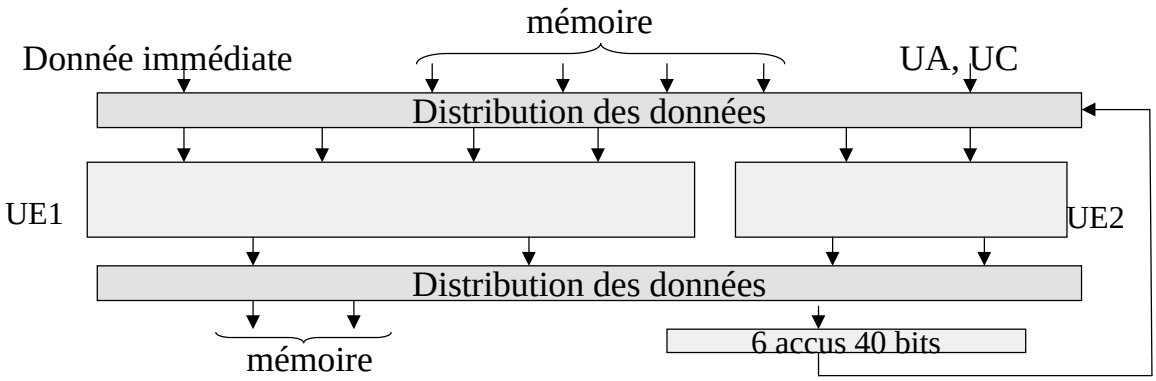
- ADI TigerSHARC
  - Combine le VLIW au SIMD afin d'atteindre un parallélisme massif
  - SIMD hiérarchique
    - Le TigerSHARC peut exécuter 8 multiplications 16x16 en virgule fixe par cycle (4x le C62xx)



Architectures VLIW

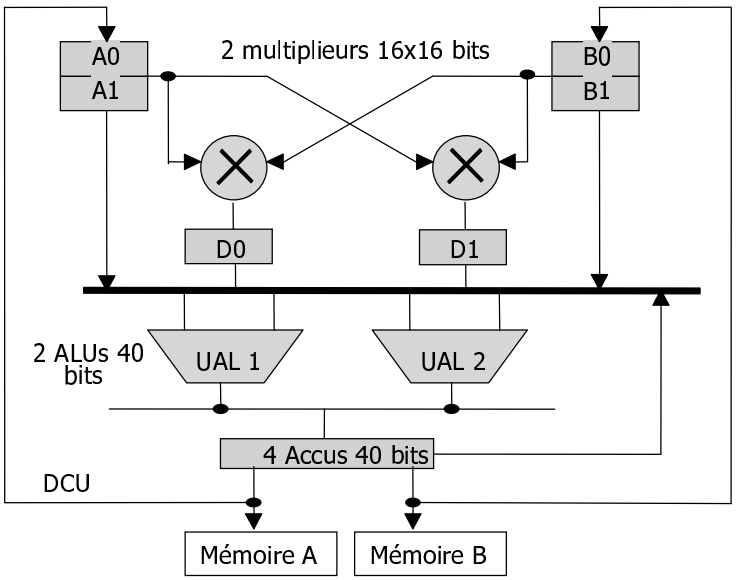
- R.E.A.L. DSP core (Philips Semi-conductor)
- Carmel DSP core (Infineon Technologies)
  - Licenciables, virgule fixe, 16 bits
  - Very Long Instruction Word : regroupement de plusieurs instructions élémentaires indépendantes dans une même instruction
  - Nombreuses unités d'exécution
  - 3 unités : UE, UA, UC

Les chemins de données (1)



- CARMEL
  - 2 UALS et 2 unités de MACs ainsi qu'un registre en barillet et une unité d'exposant
  - 6 Accumulateurs 40 bits

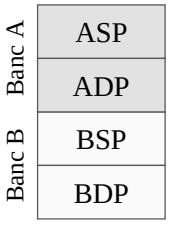
Les chemins de données (2)



- R.E.A.L.
  - 2 multiplieurs 16x16b
  - 2 UALs 40 bits / 4 UALs 16b
  - 4 Accumulateurs 40 b où 8 de 16b
  - L'utilisateur a la possibilité de définir des AXUs dédiées et de les placer n'importe où dans le chemin de données

La mémoire

- Pour alimenter le cœur en données, plusieurs bancs mémoire sont nécessaires.
- Le Carmel
  - Mémoire divisée en deux bancs A et B
  - Chaque banc divisé en simple et double port
  - Limites fixées par l'utilisateur pour une capacité totale de 64 kø



- Le REAL
  - De manière plus classique, il propose deux bancs mémoire de 64 kø

La génération d'adresses

- Un générateur d'adresses par banc mémoire : UAL + registres (base, offset, ...), adressage modulo, bit-reverse
- Carmel
  - 2 UALs par banc mémoire se partageant 4 registres de base et d'offset et 2 registres de limite et de longueur
  - Une 5ème UAL est utilisée pour la gestion de la pile
- R.E.A.L.
  - 2 UALs, chacune travaillant sur 8 registres organisés en 4 bancs de 2 pointeurs (base et offset)
  - Personnalisable à la manière du *datapath* (nouveaux modes d 'adressages, ...) avec des AXUs

Le contrôle (1)

- 3 jeux d 'instructions ≡ 3 comportements : normal / superscalaire / VLIW
- Intégration de matériel dédié à la gestion des boucles
- Exécution conditionnelle
- Carmel
  - Pipeline de 8 étages recherchant 48 bits de la mémoire instruction (8Møx48b); 48 bits correspondants à :
    - 2 instructions régulières de 24 bits
    - 1 instruction régulière de 48 bits
    - 1 appel à une instruction VLIW de 48 bits

Le contrôle (2)

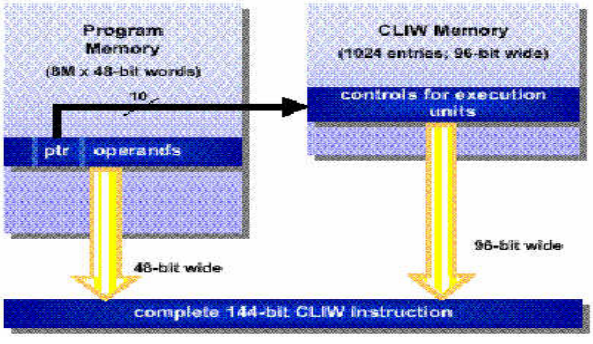
- Le R.E.A.L.
  - Pipeline de 3 étages simplifié en raison du caractère load/store de l'architecture.
  - Décomposition des jeux d 'instructions semblable au Carmel mais avec des instruction régulières codées sur 16 et 32 bits.
  - Possibilité de rajouter jusqu'à 1536 instructions pour la commande des AXUs.
  - Le traitement d'opérandes stockées en registres permet un jeu d'instruction plus orthogonal, plus simple à compiler.

Le VLIW (1)

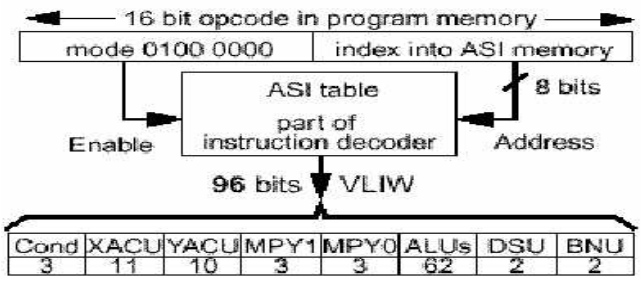
- Le R.E.A.L. et le Carmel l'utilisent de la même façon, bien que lui donnant des noms différents, respectivement ASI et CLIW.
- L'un comme l'autre appelle une instruction VLIW via un mot clé et a choisi de limiter le nombre d'entrée des mémoires VLIW à 256 pour le REAL et 1024 pour le Carmel.
- Les instructions VLIW sont à définir par l'utilisateur
- Exécution conditionnelle

Le VLIW (2)

- Carmel
  - Instruction de 144 bits dont les opérandes sont spécifiées par les 48 premiers et les opérations par les 96 restant
  - Instruction d'appel sur 48 bits



- R.E.A.L.
  - Instruction de 96 bits entièrement stockée dans la mémoire VLIW (opérations et opérandes)
  - Instruction d'appel sur 16 bits



Retour

III. Évolution des DSPs

1. DSPs conventionnels améliorés
2. Capacités SIMD
3. DSPs hybrides MCUs
4. VLIW
5. **Superscalaire**
6. **Architectures clusterisées**
7. **Comparaison de performances**

Superscalaire

- Techniques dérivées des processeurs généraux hautes-performances
  - Prédiction de branchement
  - Cache dynamique
- Plusieurs (2-4) instructions par cycle
- Jeu d'instructions de type RISC
- Parallélisme important

- Exemple
  - LSI Logic LSI401Z (ex ZSP ZSP164x)
  - Infineon TriCore

```
LOOP   // FIR on LSI401Z
   LDDU    R4, R14, 2
   LDDU    R8, R15, 2
   MAC2.A  R4,R8
   AGN0    LOOP
```

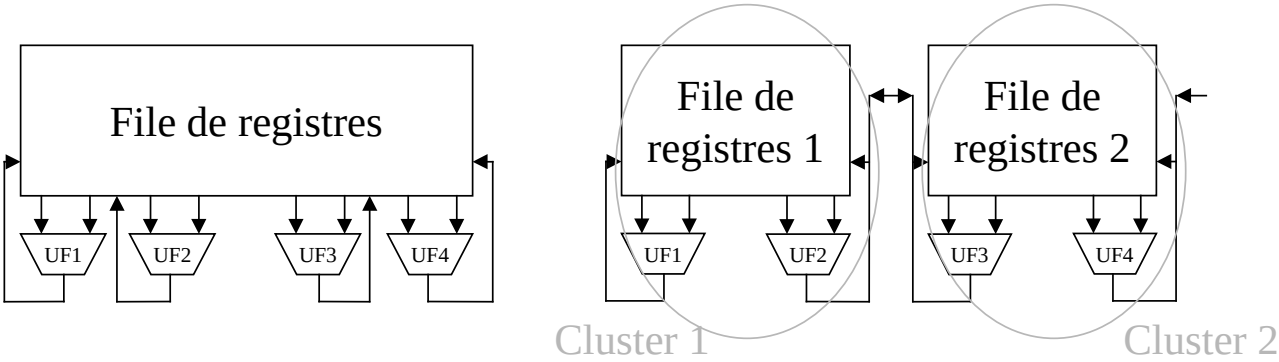
III. Évolution des DSPs

1. DSPs conventionnels améliorés
2. Capacités SIMD
3. DSPs hybrides MCUs
4. VLIW
5. Superscalaire
6. **Architectures clusterisées**
7. **Comparaison de performances**

Architectures «clusterisées»

- Objectif :
  - Apporter une solution aux problèmes posés par les architectures hautement parallèles
- L'augmentation du nombre de registres s'accompagne :
  - d'une augmentation du temps d'accès,
  - d'un décodage plus complexe,
  - d'instructions plus longues,
  - d'une augmentation de la distance entre le producteur et le consommateur.

Principe «clusterisé»

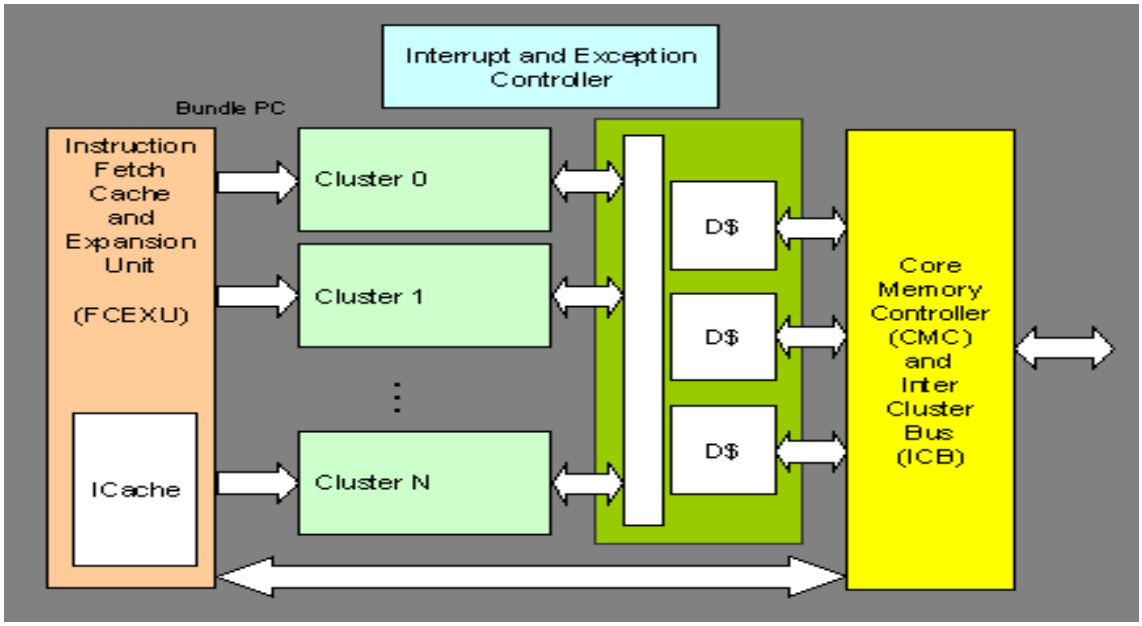


- Architecture clusterisée
  - Collection de clusters exécutant le même programme séquentiel dans le même pas de contrôle
- Cluster = banc de registres + unités fonctionnelles
  - Accès simple aux registres internes d'un cluster
  - Opérations de déplacement de données entre clusters

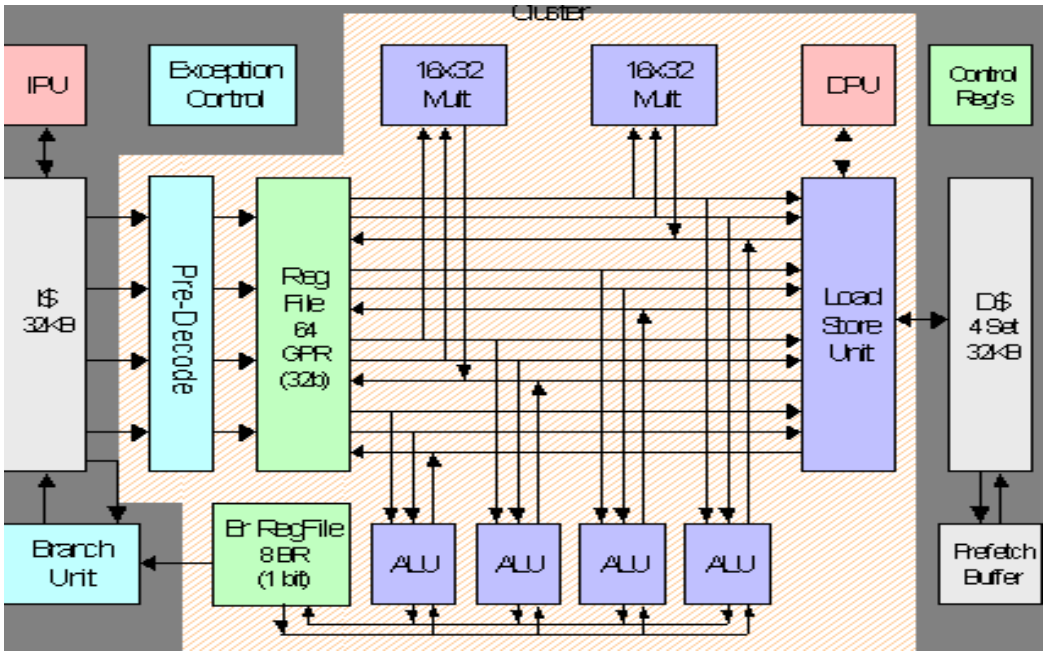
La famille Lx

- STmicroelectronics et Hewlett-Packard
- Famille Lx =
  - un cœur VLIW clusterisé,
  - un cœur configurable, personnalisable,
  - une chaîne de développement basée sur un compilateur ILP.
- Personnalisable au niveau
  - du nombre et des structures des Unités Fonctionnelles et des registres, des mémoires (dont cache),
  - du jeu d'instructions.
- Processus de développement hautement automatisé pour la *"customisation"*

Architecture du cœur



Architecture d'un cluster

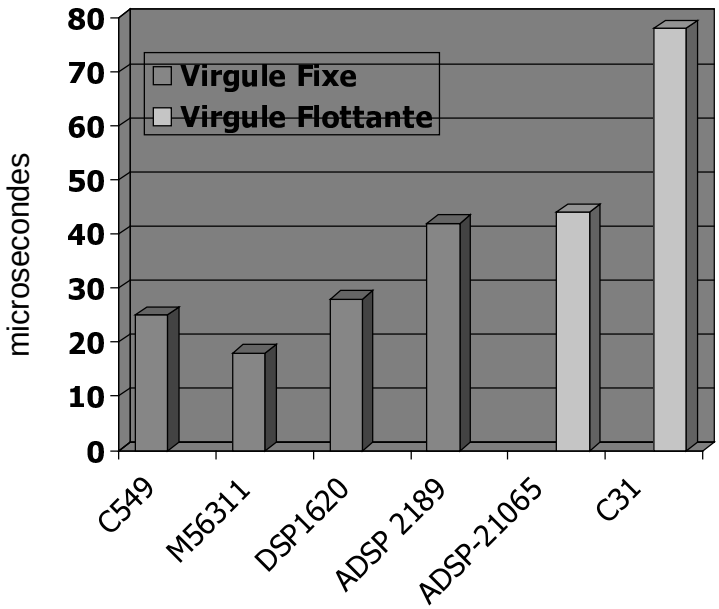


III. Évolution des DSPs

1. DSPs conventionnels améliorés
2. Capacités SIMD
3. DSPs hybrides MCUs
4. VLIW
5. Superscalaire
6. Architectures clusterisées
7. **Comparaison de performances**

Temps d'exécution

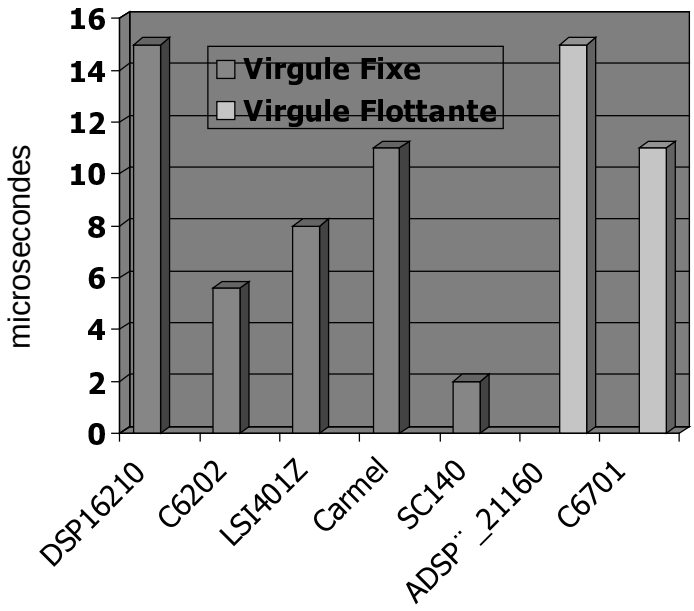
- Filtrage FIR sur DSPs conventionnels à faible coût



- C549
  - 120 MHz
- M56311
  - 150 MHz
- DSP1620
  - 120 MHz
- ADSP 2189
  - 75 MHz
- ADSP-21065
  - 60 MHz
- C31
  - 40 MHz

Temps d'exécution

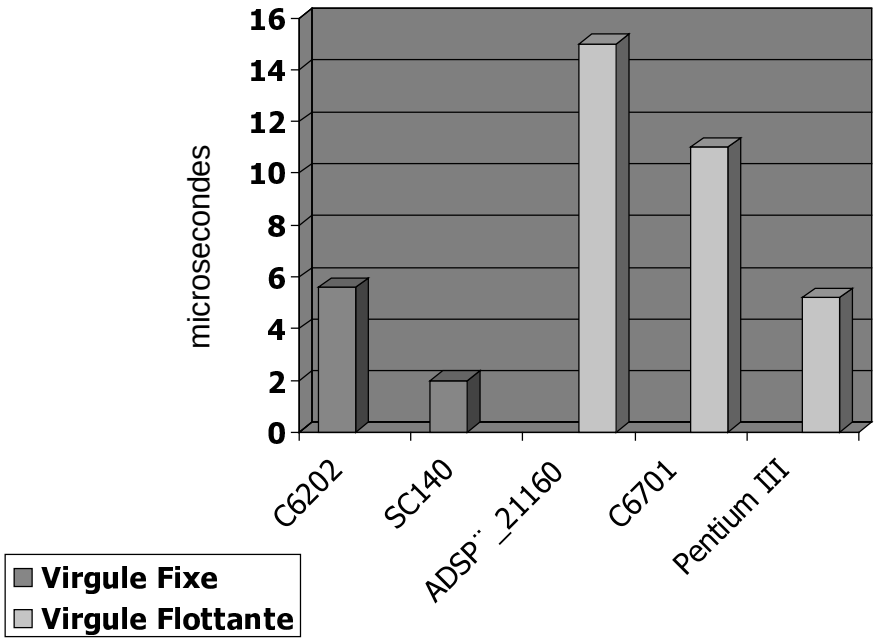
- Filtrage FIR sur DSPs hautes performances



- DSP16210
  - 120 MHz
- TMS 320C6202
  - 250 MHz
- LSI401Z
  - 200 MHz
- Carmel
  - 120 MHz
- SC-140
  - 300 MHz
- ADSP-21160
  - 100 MHz
- TMS 320C6701
  - 167 MHz

Temps d'exécution

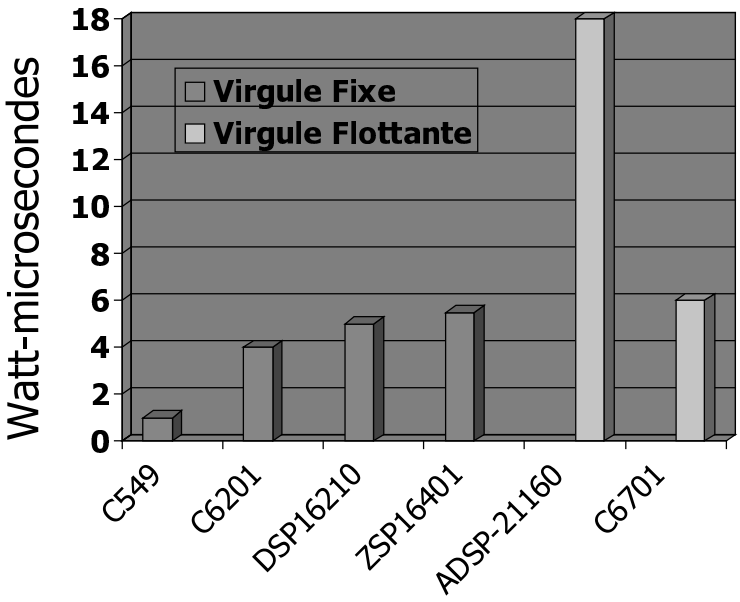
- Filtrage FIR sur DSPs et GPPs hautes performances



- TMS 320C6202
  - 250 MHz
- SC-140
  - 300 MHz
- ADSP-21160
  - 100 MHz
- TMS 320C6701
  - 167 MHz
- Pentium III
  - 1000 MHz

Consommation

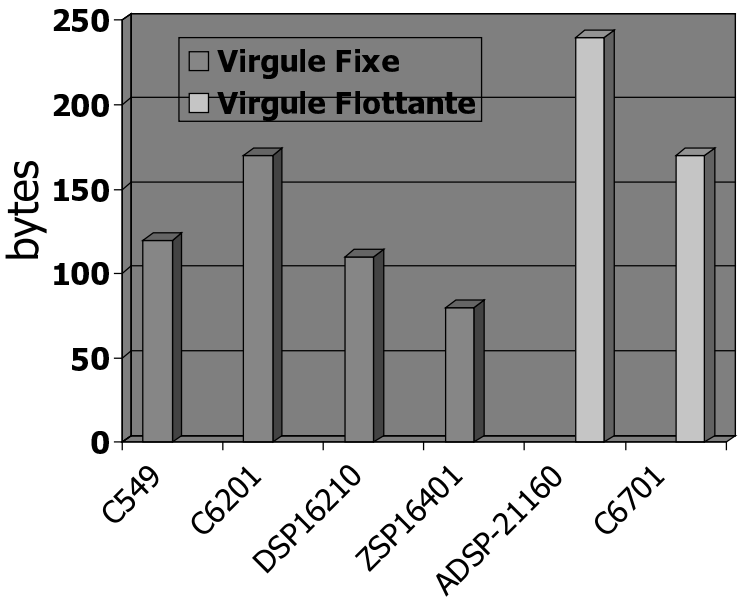
- Filtrage numérique RIF



- C549
  - 100 MHz, 2.5V
- C6201
  - 200 MHz, 1.8V
- DSP16210
  - 100 MHz, 3.3V
- ZSP16401
  - 200 MHz, 2.5V
- ADSP-21160
  - 100 MHz, 2.5V
- C6701
  - 167 MHz, 1.8V

Mémoire

- Machine d'état



- C549
  - 100 MHz, 2.5V
- C6201
  - 200 MHz, 1.8V
- DSP16210
  - 100 MHz, 3.3V
- ZSP16401
  - 200 MHz, 2.5V
- ADSP-21160
  - 100 MHz, 2.5V
- C6701
  - 167 MHz, 1.8V

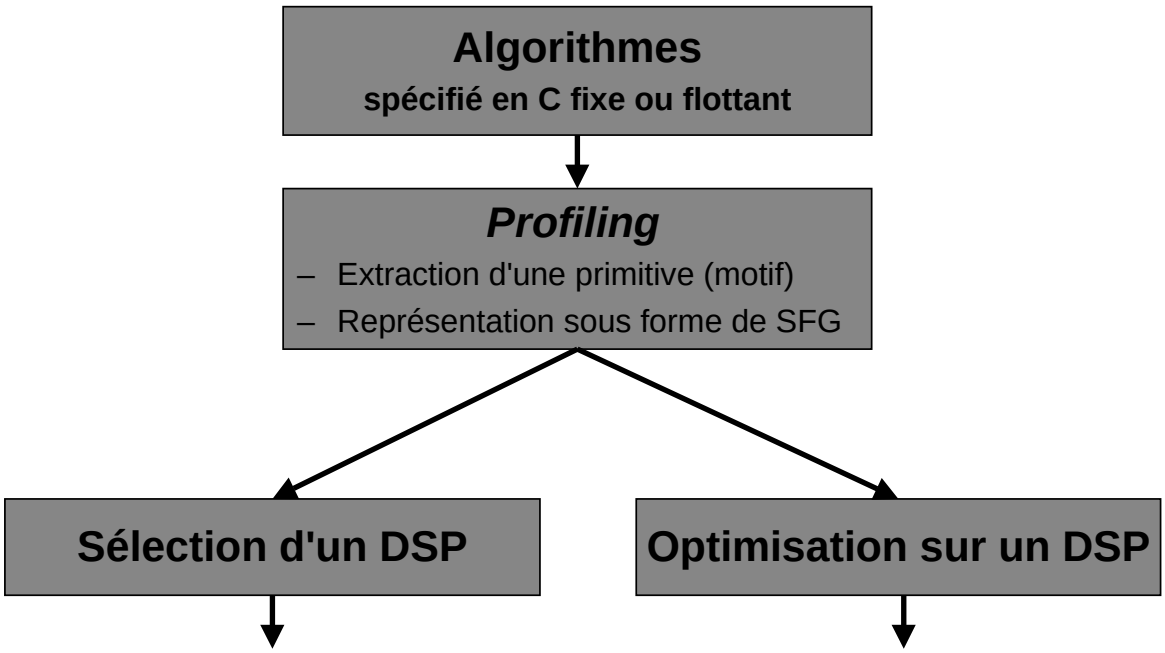
IV. Méthodes

1. **Méthodologie**
2. **Profiling**
3. **Ordonnancement et allocation**
4. **Exemples**

Optimisation.

- La connaissance de l'architecture interne est importante pour bien optimiser:
  - Placement des données et du programme pour perturber le moins possible le pipeline
- Effort sur la précision des calculs ou le temps de calcul
- Isoler la partie critique de l'algorithme qui est souvent à optimiser en assembleur
- Construire une bibliothèque optimisée de fonctions de base
- Ne pas oublier l'environnement matériel (mémoire lente, I/O, ...)

Méthode de programmation



Profiling

- Codeur JPEG

Function	Cycles/8x8 Block	Cycles/Frame
Color Conversion	192	589824
DCT	320	983040
Quantisation	64	19660
Encoding	160	491520

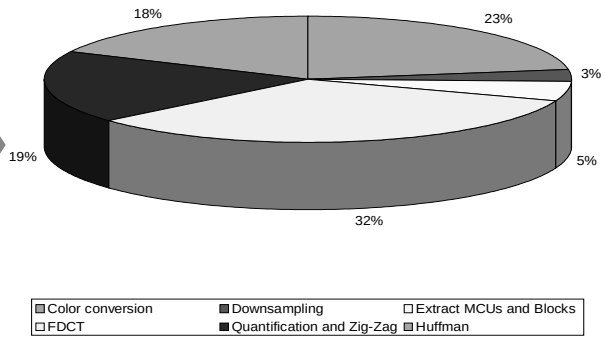
⇨ *Benchmark* applicatif
- ⇨ Lecture-écriture de l'image non pris en compte
- ⇨ Image supposée être dans la mémoire interne du DSP

Profiling de l'application

- Codeur JPEG

⇨ Résultats pour une image 256x256

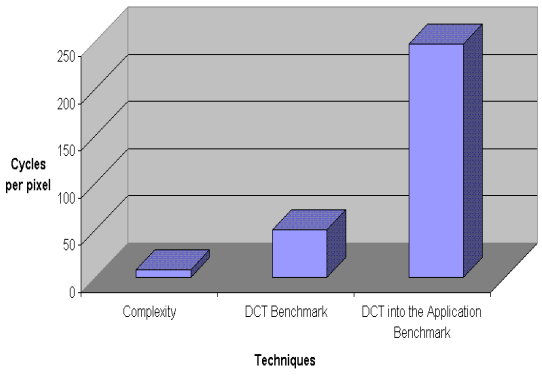
Initialisation of the board	11
Initialisation of the application	246765
Reading the input file	574751297
Color conversion	11753360
Downsampling	1592394
Extract MCUs and Blocks	2539624
FDCT	17306112
Quantification and Zig-Zag	9664396
Huffman	9226146
Write the output file	76719698
Others	61982607



- **Environ 800 cycles/pixel**

Benchmarking DCT

- *Kernel Benchmarking* => DCT
  - Assembleur => 2187 cycles
  - C => 10579 cycles
- Pour un bloc 8x8
  - Complexité => 10 cycles/pixel
  - *Benchmarking* en assembleur => 51 cycles/pixel
  - *Benchmarking* en C => 248 cycles/pixel
- Facteur entre l'assembleur et le compilateur C => 5



Méthode de programmation

Sélection d'un DSP
Modèle du DSP idéal

Optimisation sur un DSP
Modèle du DSP

Synthèse
- ordonnancement du SFG
- allocation des opérations sur les ressources
- allocation des variables sur les registres
- optimisation
  - **imbrication des calculs**
  - **pipeline logiciel**

Gestion de l'adressage
- organisation des mémoires
- table de transferts, gestion de la cohérence
- gestion des modes d'adressage

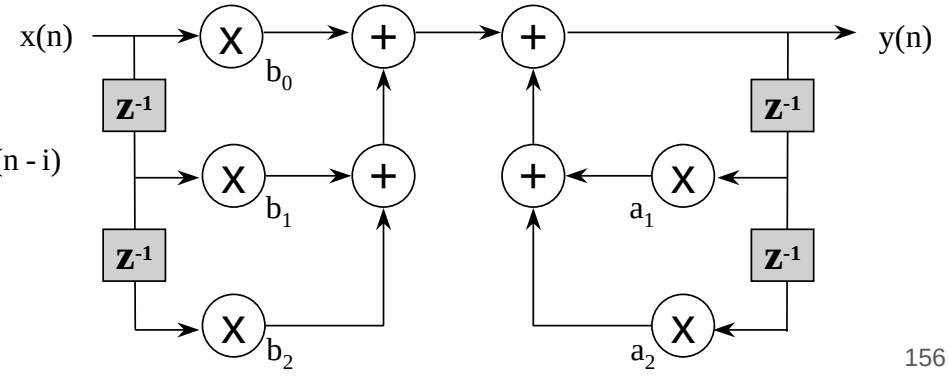
Sélection d'un DSP
Comparaison UT/jeu d'instructions

Optimisation sur un DSP
Code du DSP

Graphe Flot de Signal

- SFG : graphe cyclique G(N,A,D)
  - N : ensemble de nœuds n_i, A : ensemble d'arcs a_i
  - ni de N, représente une opération Oi
  - aij de A, représente un arc de ni vers nj si l'opération Oi est un prédécesseur de Oj dans les DFG
  - D : ensemble de délais entre itérations successives
- Le délai associé à un arc est généralement représenté par un retard (z^{-i} ou i.Te)

$$y(n) = \sum_{i=0}^{2} b_i.x(n-i) + \sum_{i=1}^{2} a_i.y(n-i)$$



[Madisetti98]

Métriques sur SFG (et DFG)

- Tbc :Temps de la (ou des) boucle critique d'un SFG

$$Tbc = \underset{b \in B}{Max}\left\lceil \frac{\sum_{j \in b} d_j}{n_b} \right\rceil$$

  - B est un ensemble de boucles b, n_b est le nombre de délais dans la boucle b, d_j est le temps d'exécution de l'opération Oj du noeud j
  - Tbc représente la latence minimale que l'on peut atteindre. C'est donc la borne min du temps d'exécution.
- Td : Temps de retard E/S d'un SFG

$$Td = \underset{p \in P}{Max}\left| \sum_{j \in p} d_j - n_p . Tbc \right|$$

  - P est un ensemble de chemin p, n_p est le nombre de délais dans le chemin p, d_j est le temps d'exécution de l'opération Oj du noeud j
  - Td représente le retard minimum entre l'entrée et la sortie d'une même itération. Définition plus générale que Tcc

Exemple

- Filtrage LMS

$$(1)\ y(n) = \sum_{i=0}^{N-1} h_n(i).x(n-i)$$

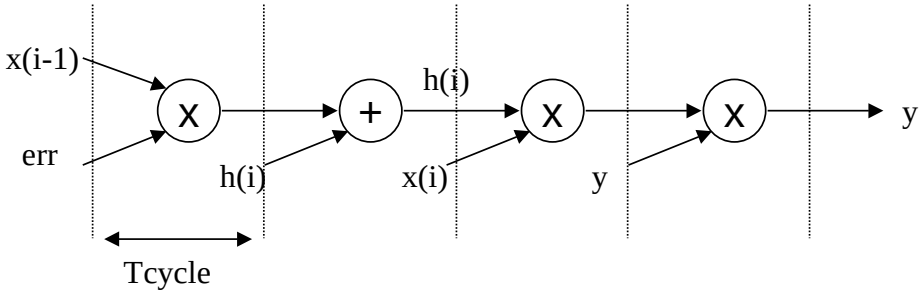
$$(2)\ e(n) = d(n) - y(n)$$

$$(3)\ h_n(i) = h_{n-1}(i) + \mu.e(n-1).x(n-i-1)$$

$$pour\ n = 0..N-1$$

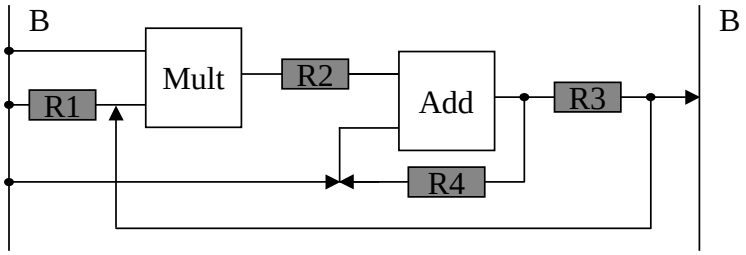
```
LIRE x(n)
y=0
POUR i de 0 à N-1 FAIRE
    h(i) = h(i) + err * x(n-i-1)
    y = y + h(i) * x(n-i)
FAIT
ECRIRE(y)
err = (d-y) * µ
```

- Motif répétitif



Exemple (suite)

- UT idéale



- Table d'occupation

Ressource	Cycle 1	Cycle 2	Cycle 3	Cycle 4
R1	err	err	err	err
R2	----	err.x(i-1)	----	h(i).x(i)
R3	----	----	h(i)	h(i)
R4	y	y	y	y
B	x(i-1)	h(i) (lecture)	x(i)	h(i) (écriture)

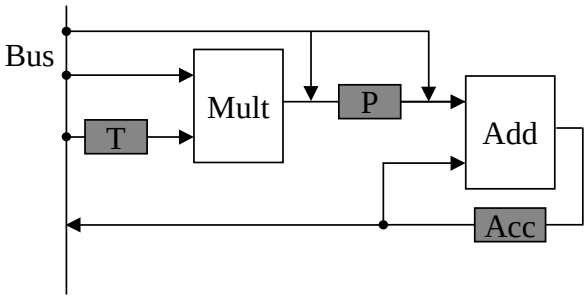
- Code

R2 := Bus B * R1 avec x(i-1) sur le bus B
R3 := R2 + Bus B avec h(i) sur le bus B
R2 := Bus B * R3 avec x(i) sur le bus B
R4 := R4 + R2 || Bus B := R3 h(i) est écrit sur le bus B

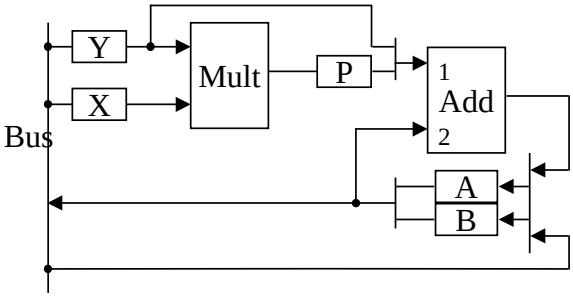
Exemple (suite)

- TMS 320 C25

T := err
P := T * bus
Acc := bus
Acc := Acc + P
bus := Acc sauve h(i)
T := bus
P := T * bus
Acc := bus charge y
Acc := Acc + P
bus := Acc sauve y



- DSP16x



Conclusions

- Les performances des processeurs DSPs ont augmenté de 150x sur 15 ans (40% par an).
- Les nouvelles architectures sont nombreuses et dominent l'offre actuelle.
  - Mais les DSPs conventionnels dominent toujours le volume de vente.
- Les processeurs généraux ont maintenant des performances qui concurrencent les DSPs.
  - Mais le prix...
- La facilité de compilation est un facteur important.
  - *time-to-market*...
- Choisir un DSP requiert une analyse fine
  - qui dépend de l'application...

Références

- [ICE97] B. McClean ICE, "Status 1997: A Report on the Integrated Circuit Industry", Integrated Circuit Engineering Corporation (ICE), Scottsdale, 1997
- [Bhaskaran95] V. Bhaskaran & K. Konstantinides, "Image and Video Compression Standards - Algorithms and Architectures", Kluwer Academic Publishers, Boston, 1995.
- [Bier97] J. Bier, P. Lapsley & G. Blalock, "Choosing a DSP Processor", Berkeley Design Technology (BDT), 1997.
- [DeMan97] H. De Man and al., "Language Based Design of Digital Systems", AI Course, KU Leuven/IMEC, april 1997.
- [Lapsley96] P. Lapsley and G. Blalock, "How to Estimate DSP Processor Performance", IEEE Spectrum, July 1996.
- [Pirsch97] P. Pirsch: "Video and Image Processing Architectures - Dedicated and Programmable Solutions", NFWO Symposium on Video Processing Architectures, January 1997.
- [Ackland98] B. Ackland and C. Nicol, "High Performance DSPs - What's Hot and What's Not?", IEEE Int. Symposium on Low Power Electronic Design ISLPED, 1998.
- [Ropers99] A. Ropers and al., "DSPstone : TI C54x" Report IISPS AAchen University of Technology, 1999.
- [Madisetti98] V. Madisetti, "VLSI Signal Processors" IEEE Press, 1998.