



## Domain specific programmable processors: architectures and design methodologies.

Ingrid Verbauwhede  
Electrical Engineering Department, UCLA  
7440B Boelter Hall  
ingrid@ee.ucla.edu

VLSI CAD seminar March 5, 2001

Ingrid Verbauwhede

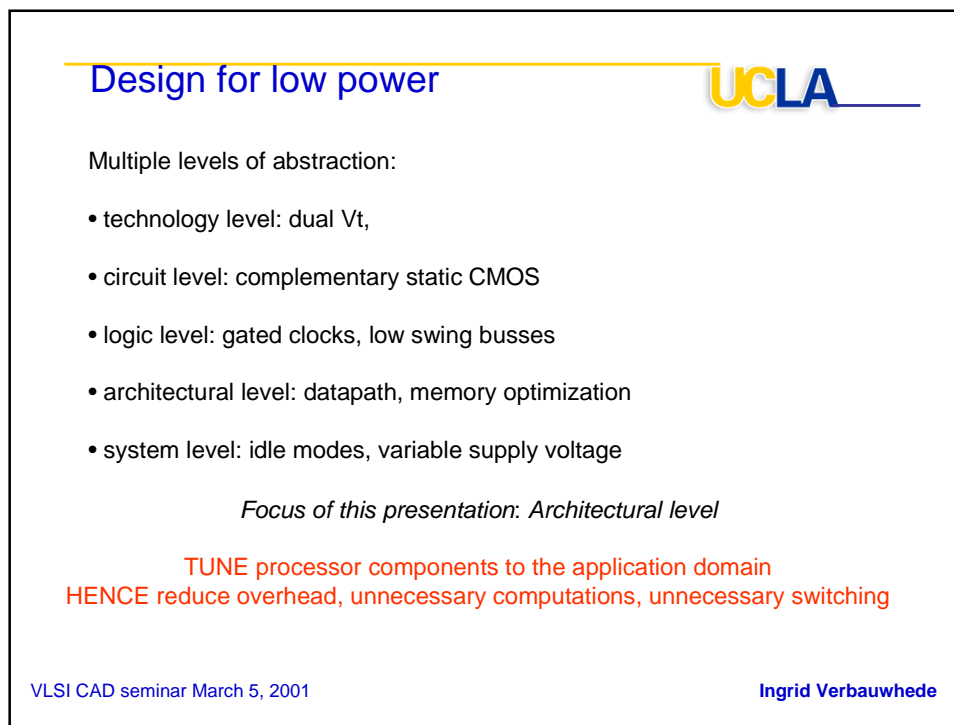
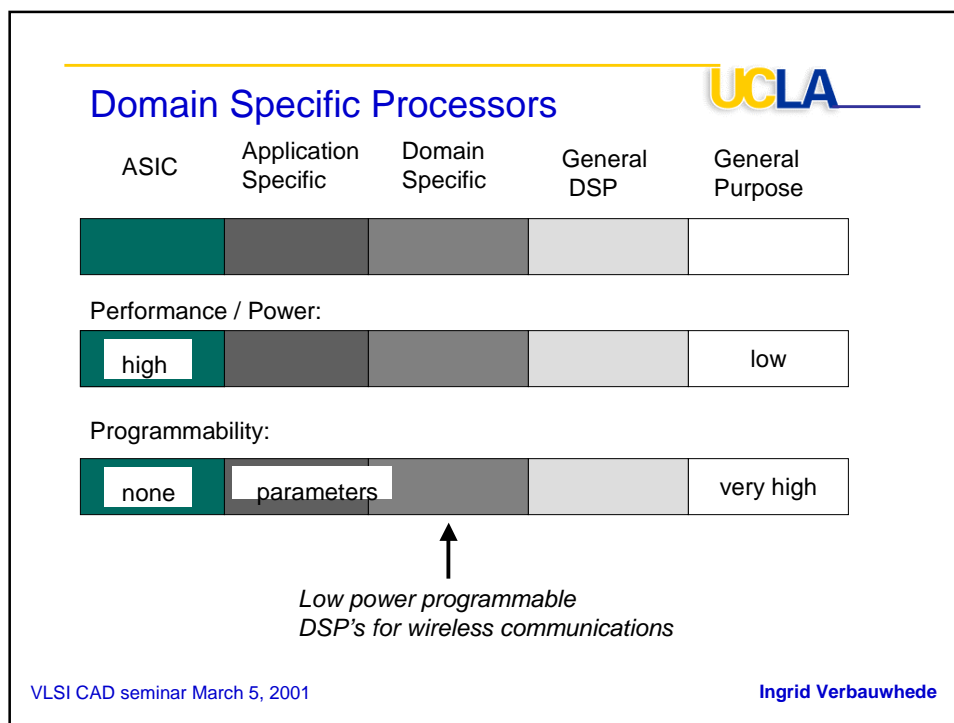


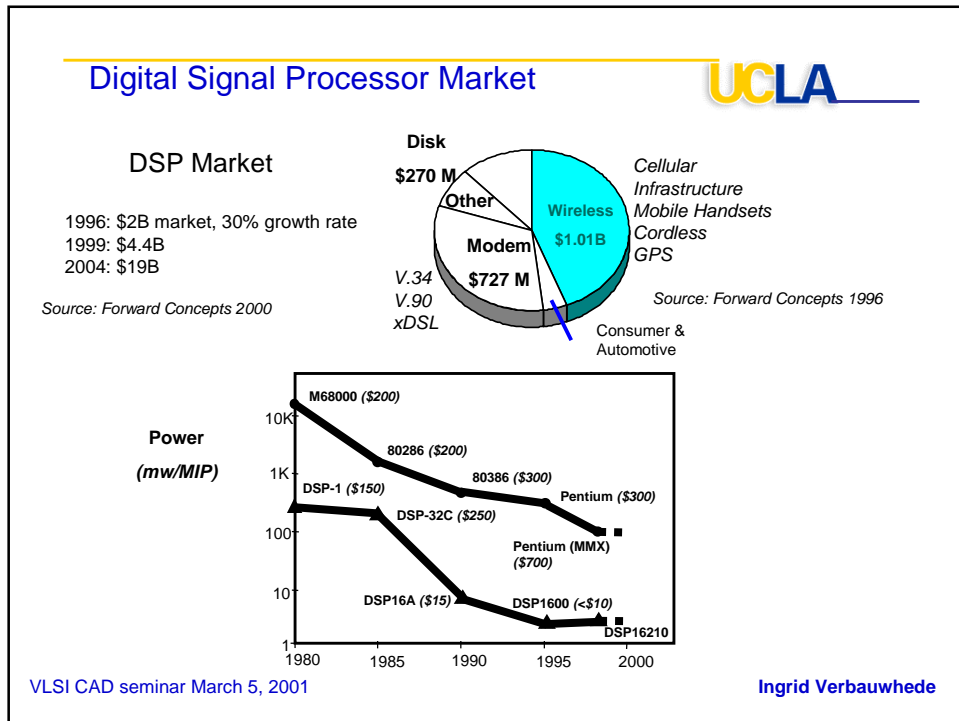
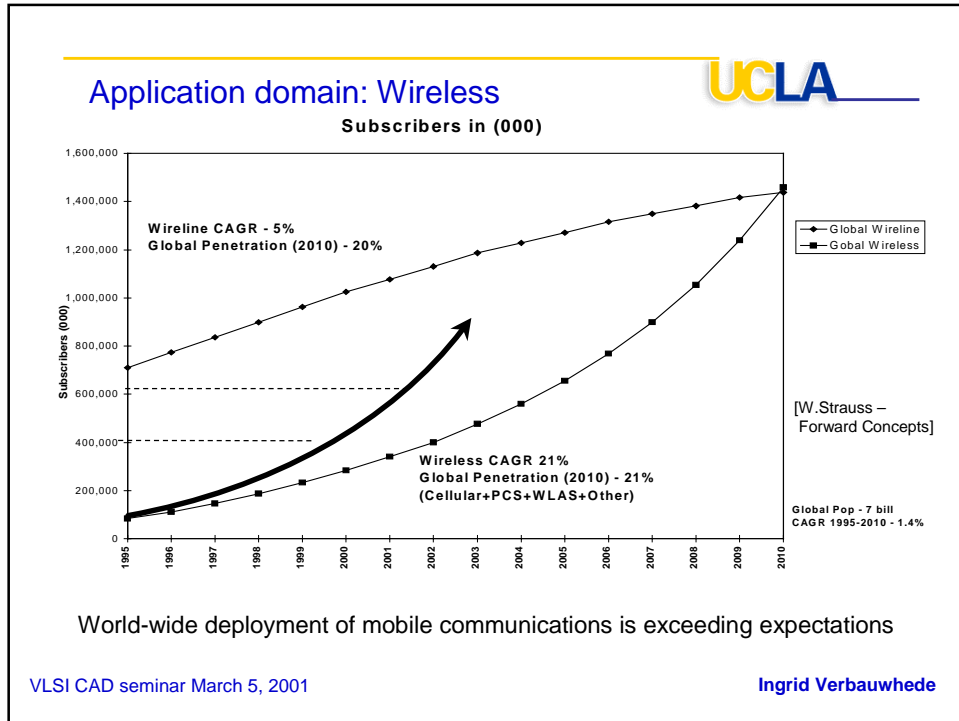
## Outline

- Definition, motivation for domain specific processors
- DSP processors
  - FIR's
  - Viterbi
  - Turbo coding
- Crypto processors
  - secret key AES
  - public key: elliptic curve
- Design methodology
- Conclusion

VLSI CAD seminar March 5, 2001

Ingrid Verbauwhede





## Domain Specific Processors



Domain specific processors: to combine

- High performance
- Low Power
- High degree of programmability

Application domains that need it:

- Wireless communications (baseband processing)
- Video processors
- Embedded micro controllers
- Etc.

*Application domain is narrower, hence need high volume to compensate development cost.*

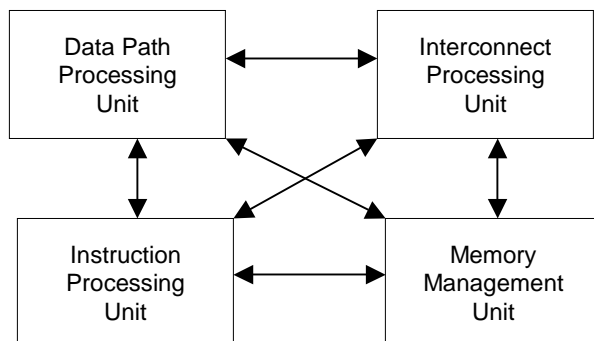
VLSI CAD seminar March 5, 2001

Ingrid Verbauwhede

## DSP Processor Fundamentals



Processor Components [Skillikorn-88]



Adapt **ALL** components to the application domain!

VLSI CAD seminar March 5, 2001

Ingrid Verbauwhede

UCLA  
 Compute intensive functions: evolution of DSP's

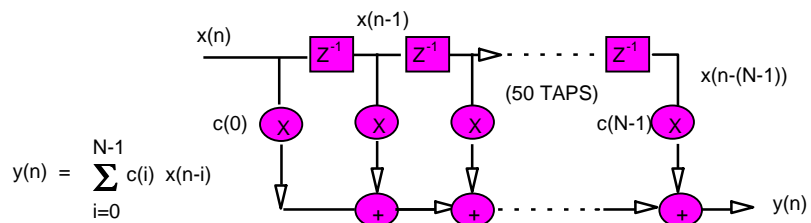
Illustrate with examples used in wireless communications.

- Simple FIR example
- Speed-up of FIR example
- Viterbi acceleration
- Square distance
- Turbo coding

VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide

UCLA  
 FIR implementation



$$\begin{aligned}
 y(0) &= c(0)x(0) + c(1)x(-1) + c(2)x(-2) + \dots + c(N-1)x(1-N); \\
 y(1) &= c(0)x(1) + c(1)x(0) + c(2)x(-1) + \dots + c(N-1)x(2-N); \\
 y(2) &= c(0)x(2) + c(1)x(1) + c(2)x(0) + \dots + c(N-1)x(3-N); \\
 &\dots \\
 y(n) &= c(0)x(n) + c(1)x(n-1) + c(2)x(n-2) + \dots + c(N-1)x(n-(N-1));
 \end{aligned}$$

Execute row by row

VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide

**UCLA**

## Von Neumann machine

- One memory space

Processor Core

mpy ALU

Address Bus

Data Bus

Memory

VLSI CAD seminar March 5, 2001 Ingrid Verbauwhede

**UCLA**

## FIR on Von Neumann


Assume Von Neumann has multiply and accumulate instruction (not necessarily the case)  
Assume also that pipelining allows to execute the multiply and accumulate in parallel with the read or write operations.  
Then one tap needs 4 cycles:

1. read multiply-accumulate instruction
2. read data value from memory
3. read coefficient from memory
4. write data value to the next location in the delay line (because for the next sample, all values are shifted by one location)

***Memory bandwidth is crucial !!!***

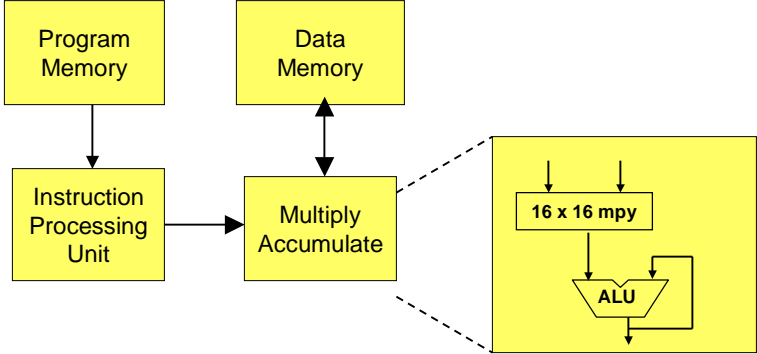
50 taps filter = 200 cycles!

VLSI CAD seminar March 5, 2001 Ingrid Verbauwhede




## Basic Harvard Architecture

- Separate data memory from program memory!

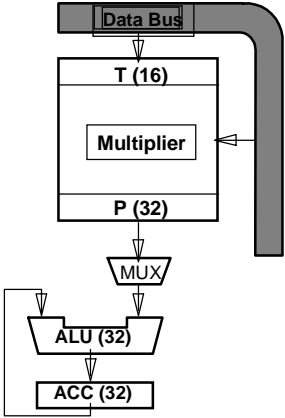


- Different from Von Neumann machine:  
one address bus - one data bus - one memory space

VLSI CAD seminar March 5, 2001 Ingrid Verbauwheide



## TMS320C1x (1982)

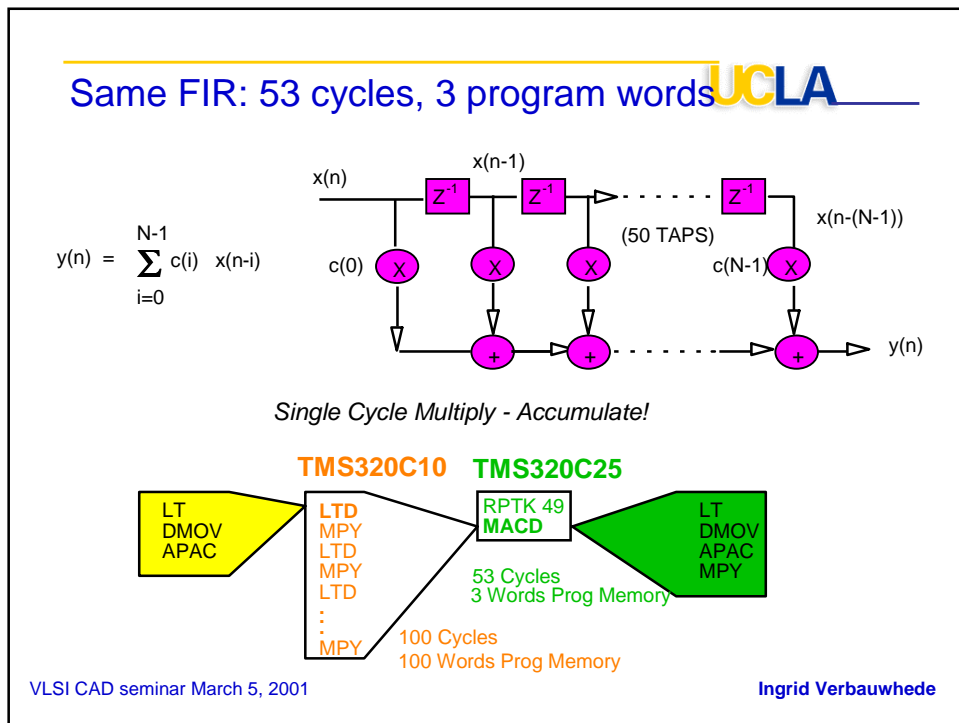
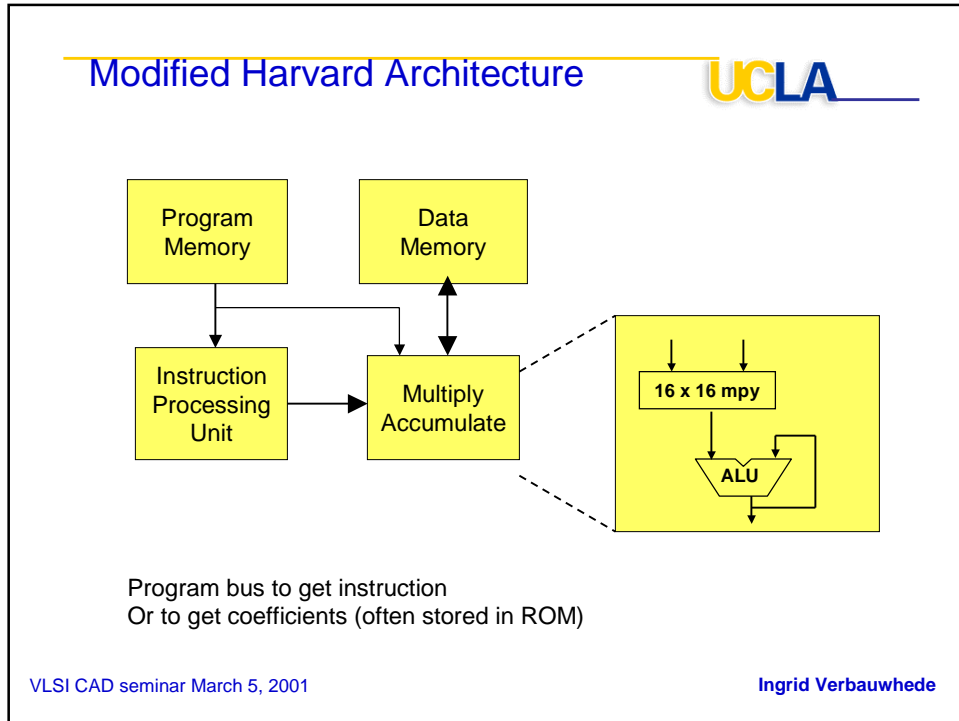


Compute  $Y = AX1 + BX2 + CX3 + DX4$

<b>ZAC</b>		ACC=0
<b>LT</b>	<b>X1</b>	T=X1
<b>MPY</b>	<b>A</b>	P=AX1
<b>LTA</b>	<b>X2</b>	ACC=AX1;T=X2
<b>MPY</b>	<b>B</b>	P=BX2
<b>LTA</b>	<b>X3</b>	ACC=AX1+BX2;T=X3
<b>MPY</b>	<b>C</b>	P=CX3
<b>LTA</b>	<b>X4</b>	ACC=AX1+BX2+CX3;T=X4
<b>MPY</b>	<b>D</b>	P=DX4
<b>APAC</b>		ACC=AX1+BX2+CX3+DX4
<b>SACH</b>	<b>Y1</b>	STORE 32-BIT RESULT
<b>SACH</b>	<b>Y2</b>	AT LOCATIONS Y1, Y2

- 50 taps = 104 cycles
- = Program ROM of 104 instructions

VLSI CAD seminar March 5, 2001 Ingrid Verbauwheide



TMS320C2x (1986) UCLA

### TMS320C2x Multiplier/ALU

- Single Cycle 16x16 bit Multiply yielding a 32-bit product
- Supports simultaneous Program and two Data Operand acquisition
- Supports simultaneous ALU and Multiplier operations
- 0-16 bit Left Post-Shifter

VLSI CAD seminar March 5, 2001 Ingrid Verbauwheide

### FIR speed-up UCLA

- One output = 2N reads, N MAC's, 1 write
- Two outputs = 4N reads, 2N MAC's, 2 writes

*Dual Mac Architecture with ONLY 2 data busses??*

- Read two 32-bit numbers instead of four 16-bit numbers  
*Solution by Lucent 16000 core with dual MAC*
- Run MAC at double frequency, read two 32-bit numbers  
*Solution by Matsushita*
- Insert delay register  
*Solution by Atmel's LODE (1996)*
- Use 3 busses  
*Solution by TI C55x (2000)*

VLSI CAD seminar March 5, 2001 Ingrid Verbauwheide

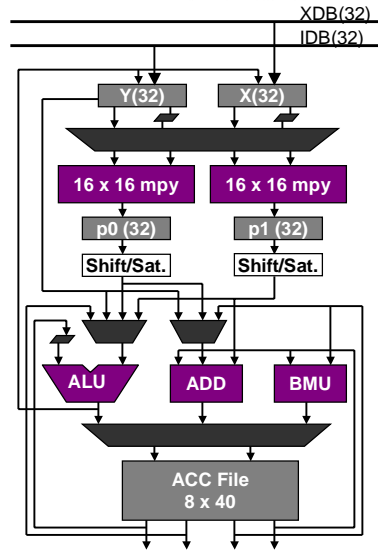
## FIR on Lucent DSP16210



### Inner loop of 32-tap FIR Filter

```
do 14 { //one instruction !
  a0=a0+p0+p1
  p0=xh*yh p1=xl*y1
  y=*r0++ x=*pt0++
}
```

- Outer Loop: 19 cycles, 38 bytes  
1 cycle in inner loop
- 5 exec units used in inner loop  
2 MACs per cycle
- Horizontal parallelism, one sample at a time
- 2G mobile wireless base-stations



Courtesy: Gareth Hughes, Bell Labs Australia  
VLSI CAD seminar March 5, 2001

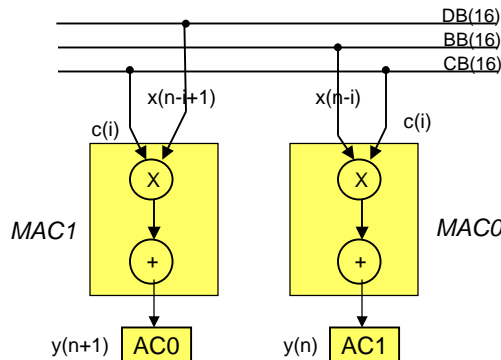
Ingrid Verbauwheide

## FIR on TI C55x (2000)



- FIR filter: two outputs in parallel with 3 busses

$$\begin{aligned}
 y(0) &= c(0)x(0) + c(1)x(-1) + c(2)x(-2) + \dots + c(N-1)x(1-N); \\
 y(1) &= c(0)x(1) + c(1)x(0) + c(2)x(-1) + \dots + c(N-1)x(2-N); \\
 y(2) &= c(0)x(2) + c(1)x(1) + c(2)x(0) + \dots + c(N-1)x(3-N); \\
 y(3) &= c(0)x(3) + c(1)x(2) + c(2)x(1) + \dots + c(N-1)x(4-N);
 \end{aligned}$$



VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide

## FIR on Lode (1996)

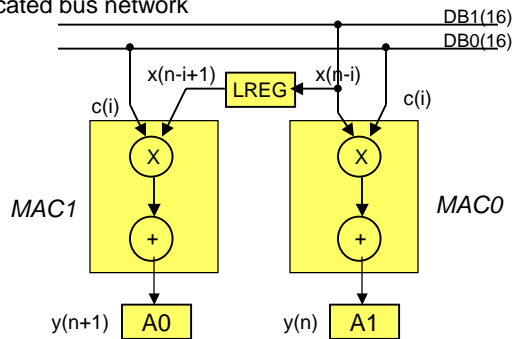


- FIR filter: two outputs in parallel with delay register

$$\begin{aligned}
 y(0) &= c(0)x(0) + c(1)x(-1) + c(2)x(-2) + \dots + c(N-1)x(1-N); \\
 y(1) &= c(0)x(1) + c(1)x(0) + c(2)x(-1) + \dots + c(N-1)x(2-N); \\
 y(2) &= c(0)x(2) + c(1)x(1) + c(2)x(0) + \dots + c(N-1)x(3-N); \\
 y(3) &= c(0)x(3) + c(1)x(2) + c(2)x(1) + \dots + c(N-1)x(4-N);
 \end{aligned}$$

- Two MAC units with dedicated bus network

- DB0 fetches coefficient
- DB1 fetches data
- LREG delays input data
- A0 stores  $y(n)$  output
- A1 stores  $y(n+1)$  output



VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide

## Energy comparison



- Total energy for one output sample:

Energy	Single MAC	Dual MAC	Dual MAC 3 busses	Dual MAC with REG
No. of MAC operations	N	N	N	N
No of Memory reads	2N	2N	1.5N	N
No of Instruction Cycles	N	N/2	N/2	N/2

Adaptation of the datapath: MAC, DMAC

Adaptation of the memory architecture and bus network

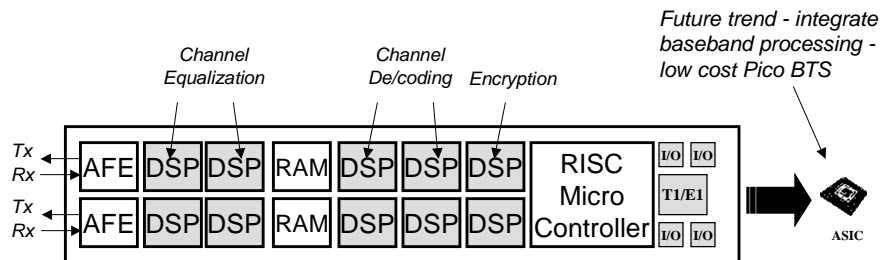
Adaptation of the instruction set

VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide

## 2G Basestation Baseband Processing

- Multiple DSPs used for baseband processing.
- RISC Microcontroller for timing, framing, I/O control
- Software upgradable over the network
- DSPs dominate cost and power consumption



Tx/Rx baseband processing board for 2-carrier GSM basestation

VLSI CAD seminar March 5, 2001

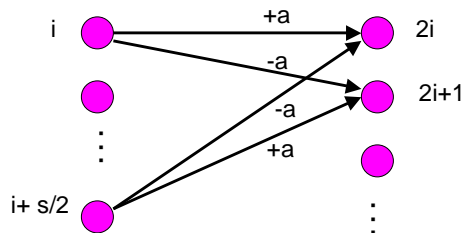
[C. Nicol, Bell Labs Australia]

Ingrid Verbauwheide

## Compute Intensive function 2: Viterbi

- Viterbi butterfly

$i$  = state index  
 $s$  = # of states =  $2^{k-1}$   
 $w$  = decoding window



- Basic equations:

$$d(2i) = \min \{ d(i) + a, d(i + s/2) - a \}$$


$$d(2i + 1) = \min \{ d(i) - a, d(i + s/2) + a \}$$

- Key operation: Add-Compare-Select (ACS)
- IS-95:  $k = 9$ , 256 states,  $w = 192$ , means  $2^8 \times 192 \times$  (cycles for one ACS)
- Basic algorithm in Viterbi channel decoders and MLSE based receivers, modified version in turbo decoders.

VLSI CAD seminar March 5, 2001

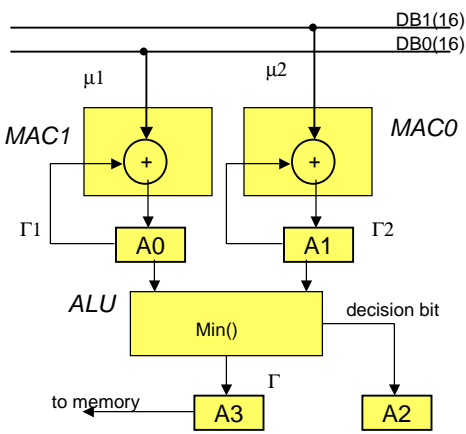
Ingrid Verbauwheide

$$\Gamma = \min [(\Gamma_1 + \mu_1), (\Gamma_2 + \mu_2)]$$




● Two MAC units & ALU: Add-Compare-Select

- DMAC operates as dual add/subtract unit
- ALU finds minimum
- Shortest distance saved
- Path indicator saved
- 4 cycles / butterfly



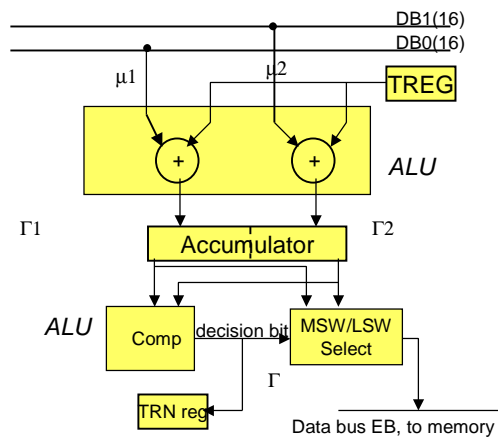
VLSI CAD seminar March 5, 2001 Ingrid Verbauwheide

$$\Gamma = \min [(\Gamma_1 + \mu_1), (\Gamma_2 + \mu_2)]$$



● ALU and CSSU: CMPS instruction


- ALU splits in 16 bit halves
- ACC splits in half
- Shortest distance saved
- CSSU compares halves
- Path indicator saved
- 4 cycles / butterfly



Source: TI Application Report, Viterbi Decoding in the TMS320C54x family, document SPRA071

VLSI CAD seminar March 5, 2001 Ingrid Verbauwheide

## Viterbi on LU DSP16210

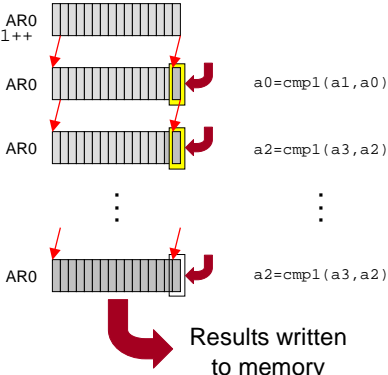


**GSM (K=5, 16 states)**

```

do 8 {
  a0=a4+y a1=a5-y *r3++=a0h
  a2=a4-y a3=a5+y *r5++=a2h
  a0=cmp1(a1,a0) yh=*r0 r0=r1+j j=k k=*pt1++
  a2=cmp1(a3,a2) a4_5h=*pt0++
}
    
```


- Comparison functions store ACS decision bits:
- Hardware support for Viterbi algorithm:
  - ACS calculations are efficient
  - Minimal overhead
- 4 cycles per butterfly
  - 32 cycles per GSM timeslot.



Courtesy: Gareth Hughes, Bell Labs Australia  
VLSI CAD seminar March 5, 2001

Ingrid Verbauwhede

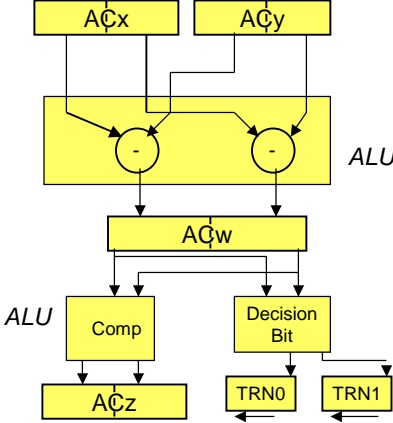
## Turbo coding on TIC55x



- ALU and CSSU: *max\_diff* instruction

Maxdiff ACx, ACy, ACz, ACw

- ALU splits in 16 bit halves
- 4 ACC's splits in half
- differences stored in ACw halves
- Max stored in ACz halves
- Path indicators saved in TRN's



Source: TMS320C55x DSP Mnemonic Instruction Set Ref. Guide, document SPRU374C  
VLSI CAD seminar March 5, 2001

Ingrid Verbauwhede

## Case study: LPC Speech Coder



Main computation modules:

- FIR's, autocorrelation, Levinson-Durbin algo

Platform	Clock Freq.	Cycles (MIPS)	Area/ Memory	Energy/ frame	Techno-logy	Power Supply
TI C5402	12 MHz	240K	8.7 KB	42.7 $\mu$ J	0.18 $\mu$ m <sup>b</sup>	1.8 V core 3.3 V I/O
TI C5510	5 MHz	120K	10.2 KB	3.2 $\mu$ J	0.15 $\mu$ m <sup>b</sup>	1.6 V core 3.3 V I/O
TI C6211	150 MHz <sup>a</sup>	30K	16 KB <sup>a</sup>	288 $\mu$ J	0.18 $\mu$ m	1.8 V core 3.3 V I/O
OCAPI	600 KHz	11K	1.4 mm <sup>2</sup>	2.1 $\mu$ J	0.25 $\mu$ m	2.5 V
ARJT	150KHz	3K	3.2 mm <sup>2</sup>	4.3 $\mu$ J	0.35 $\mu$ m	3.3 V

a: accommodates around 75 channels, ONLY program code

b: estimates

EE201A Class project!

VLSI CAD seminar March 5, 2001

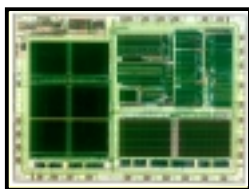
Ingrid Verbauwheide

## Low Power DSP's



### DSP 1600 Core

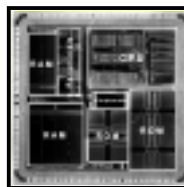
(Lucent - 1609 low cost consumer 16-bit)



- 0.35 $\mu$  3LM CMOS
- 80 M 16b MAC/s at 3.3V
- 1.4 mW/MHz at 3.3V
- 30  $\mu$ W stand-by power

### C54x 1V DSP

(Texas Instruments - ISSCC 1997)



- 0.25 $\mu$  3LM CMOS
- 63 M 16b MAC/s at 1.0V
- 0.21 mW/MHz at 1.0V
- 4.0 mW stand-by power
- Dual  $V_t$  process

VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide

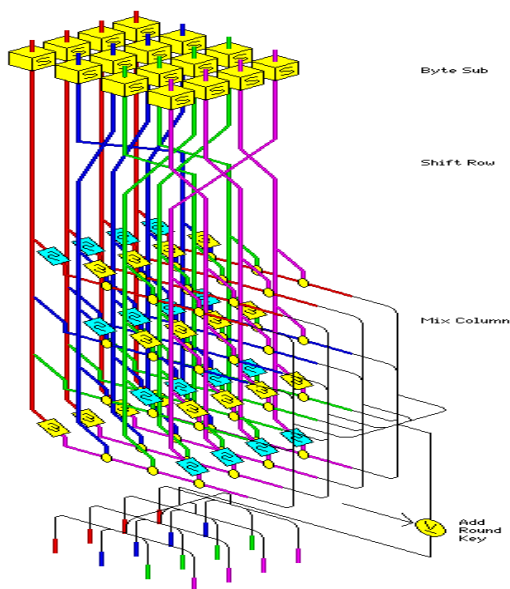
## Cryptography



- ❑ Unusual arithmetic, large word lengths
- ❑ High throughput requirements (e.g. routers)
- ❑ Low power (portable applications)
- ❑ Resistant against timing and power attacks (smart cards)
- ❑ Our work:
  - ❑ AES Rijndael co-processor ( 6Gb/sec encryption!)
  - ❑ Elliptic curve public key co-processor  
on Atmel FPSLIC (FPGA + AVR co-processor)
  - ❑ Circuit techniques resistant against power attacks

VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide



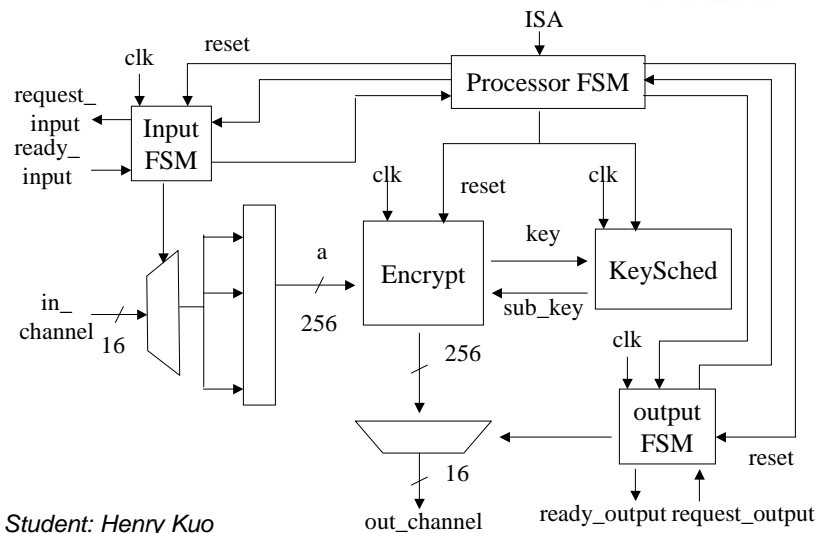
## AES Rijndael

[ref: <http://home.ecn.ab.ca/~javid/crypto>]

VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide

## AES Rijndael – 6Gb/s encryption rate

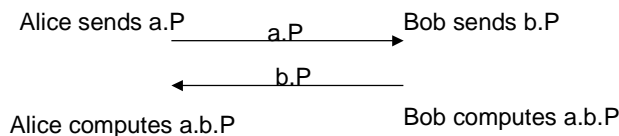


VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide

## Elliptic curve cryptography

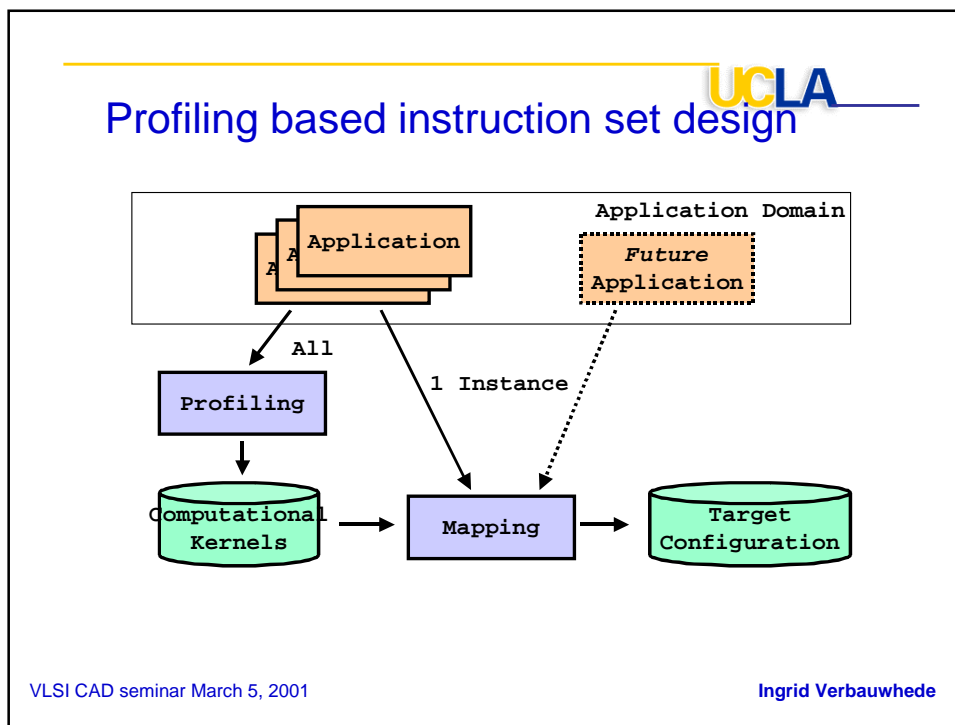
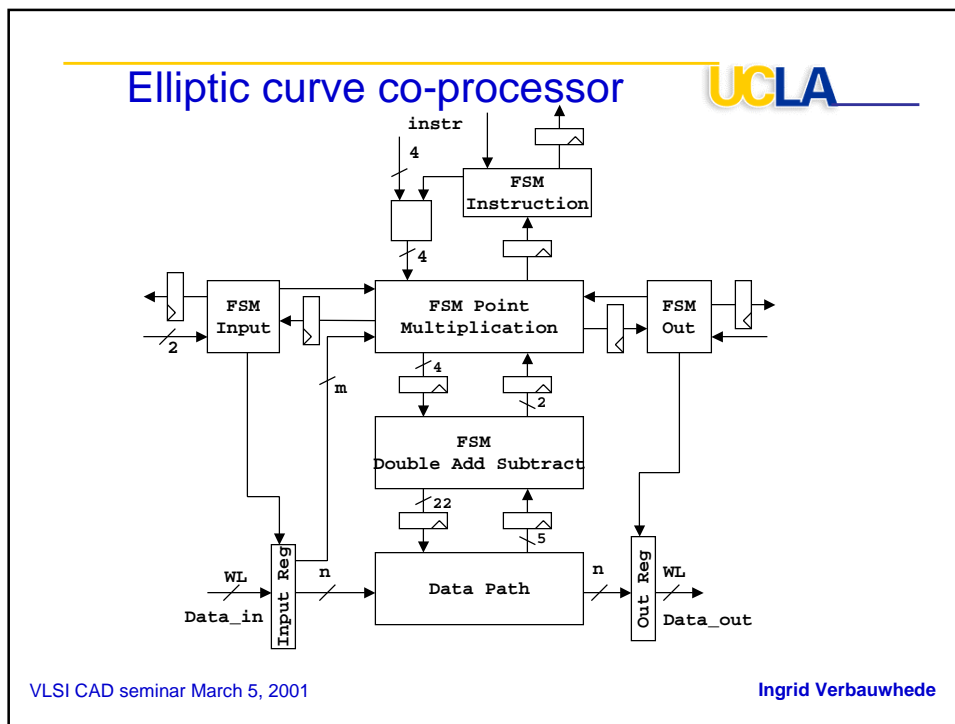
- Diffie Hellman key exchange protocol
- $P$  is point on the elliptic curve,  $a$  and  $b$  are secrets of Alice and Bob




- Secret key is  $a.b.P$
- Eavesdropper has  $a.P$  and  $b.P$ , but can not derive  $a$  nor  $b$ .
- Based on discrete logarithm problem is hard in elliptic curve

VLSI CAD seminar March 5, 2001

Ingrid Verbauwheide






## Example: L2, L3, L4 header parsing

Student: Maged Attia

VLSI CAD seminar March 5, 2001 Ingrid Verbauwheide



## Conclusion

- ❑ Domain specific processor
  - ❑ optimized datapaths
  - ❑ optimized controller/instruction set
  - ❑ optimized memory architecture and bandwidth
  - ❑ optimized interconnect
- ❑ Current research: processor architectures for
  - ❑ secret-key & public-key cryptography
  - ❑ wireless communications: turbo codes, Low density parity check codes
  - ❑ embedded routers
  - ❑ DESIGN METHODOLOGY to support this.
- ❑ Request for: floorplanning and routing for differential logic styles.

VLSI CAD seminar March 5, 2001 Ingrid Verbauwheide

---

## Need to run...



- EE215B – Advanced Digital Integrated Circuits
- Main project topic: noise & reliability in deep-submicron
  - interconnect – RLC effects on chip?
  - mixed-signal on chip
  - process variations
- Today: adiabatic computing
- NO CS students enrolled!