INTRODUCTION TO DIGITAL SIGNAL PROCESSORS

Prof. Brian L. Evans

in collaboration with

Niranjan Damera-Venkata and

Magesh Valliappan

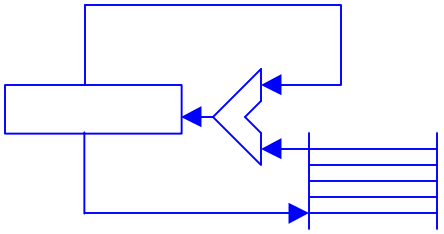
Embedded Signal Processing Laboratory

The University of Texas at Austin

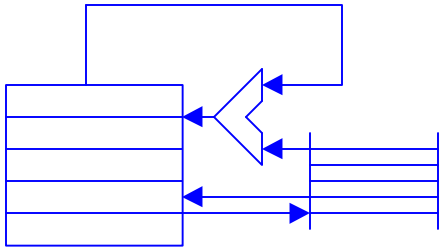
Austin, TX 78712-1084

http://signal.ece.utexas.edu/

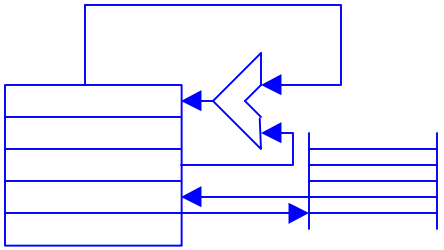
Accumulator architecture



Memory-register architecture



Load-store architecture



Outline

- Signal processing applications
- Conventional DSP architecture
- Pipelining in DSP processors
- RISC vs. DSP processor architectures
- TI TMS320C6x VLIW DSP architecture
- Signal and image processing applications
- Signal processing on general-purpose processors
- Conclusion

Signal Processing Applications

- Low-cost embedded systems
  - Modems, cellular telephones, disk drives, printers
- High-throughput applications
  - Halftoning, base stations, 3-D sonar, tomography
- PC based multimedia
  - Compression/decompression of audio, graphics, video
- Embedded processor requirements
  - Inexpensive with small area and volume
  - Deterministic interrupt service routine latency
  - Low power: ~50 mW (TMS320C54x uses 0.36 μA/MIP)

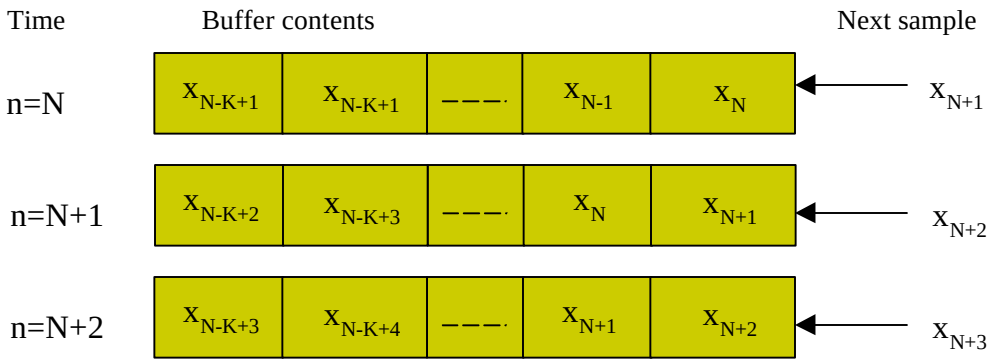
Conventional DSP Architecture

- Harvard architecture
  - Separate data memory/bus and program memory/bus
  - Three reads and one or two writes per instruction cycle
- Deterministic interrupt service routine latency
- Multiply-accumulate in single instruction cycle
- Special addressing modes supported in hardware
  - Modulo addressing for circular buffers (e.g. FIR filters)
  - Bit-reversed addressing (e.g. fast Fourier transforms)
- Instructions to keep the pipeline (3-4 stages) full
  - Zero-overhead looping (one pipeline flush to set up)
  - Delayed branches

Conventional DSP Architecture (con't)

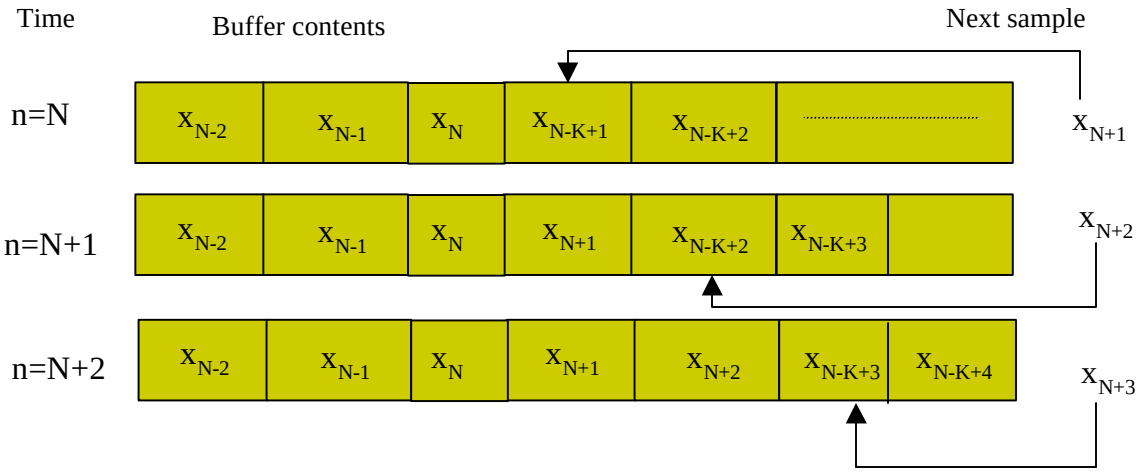
- Modulo addressing
  - implementing circular buffers and delay lines

Data-shifting



- Bit reversed addressing
  - used to implement the radix-2 FFT

Modulo addressing



Conventional DSP Architecture (con't)

	Fixed-Point	Floating-Point
Cost/Unit	$5 - $79	$5 - $381
Architecture	Accumulator	load-store or memory-register
Registers	2-4 data 8 address	8 or 16 data 8 or 16 address
Data Words	16 or 24 bit integer and fixed-point	32 bit integer and fixed/floating-point
On-Chip Memory	2-64 kwords data 2-64 kwords program	8-64 kwords data 8-64 kwords program
Address Space	16-128 kw data 16-64 kw program	16 Mw – 4Gw data 16 Mw – 4 Gw program
Compilers	C compilers; poor code generation	C, C++ compilers; better code generation
Examples	TI TMS320C5x; Motorola 56000	TI TMS320C3x; Analog Devices SHARC

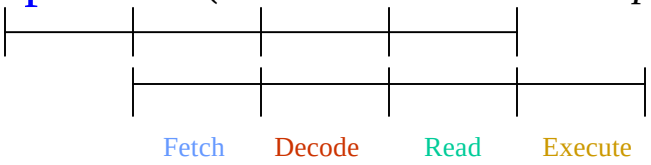
Conventional DSP Architecture (con't)

- Market share: 95% fixed-point, 5% floating-point
- Each processor family has dozens of members with different on-chip configurations
  - Size and map of data and program memory
  - A/D, input/output buffers, interfaces, timers, and D/A
- Drawbacks to conventional DSP processors
  - No byte addressing (needed for image and video)
  - Limited on-chip memory
  - Limited addressable memory on fixed-point DSPs, except Motorola 56300 (16 Mw data; 64 Mw program)
  - Non-standard C extensions to support fixed-point data

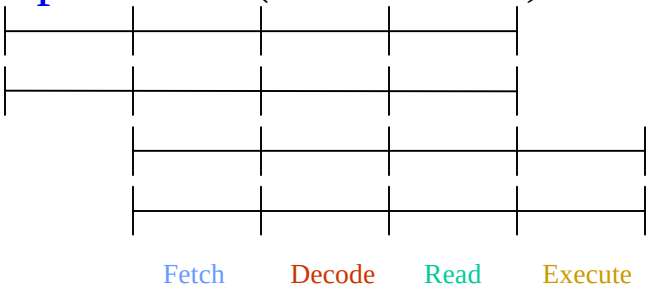
Pipelining

Sequential *(Motorola 56000)*

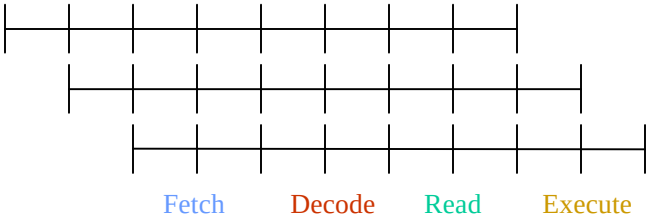
Pipelined *(Most conventional DSP processors)*



Superscalar *(Pentium, MIPS)*



Superpipelined *(CDC7600)*



Managing Pipelines

- compiler or programmer
- pipeline interlocking in the processor
- hardware instruction scheduling

Pipelining: Operation

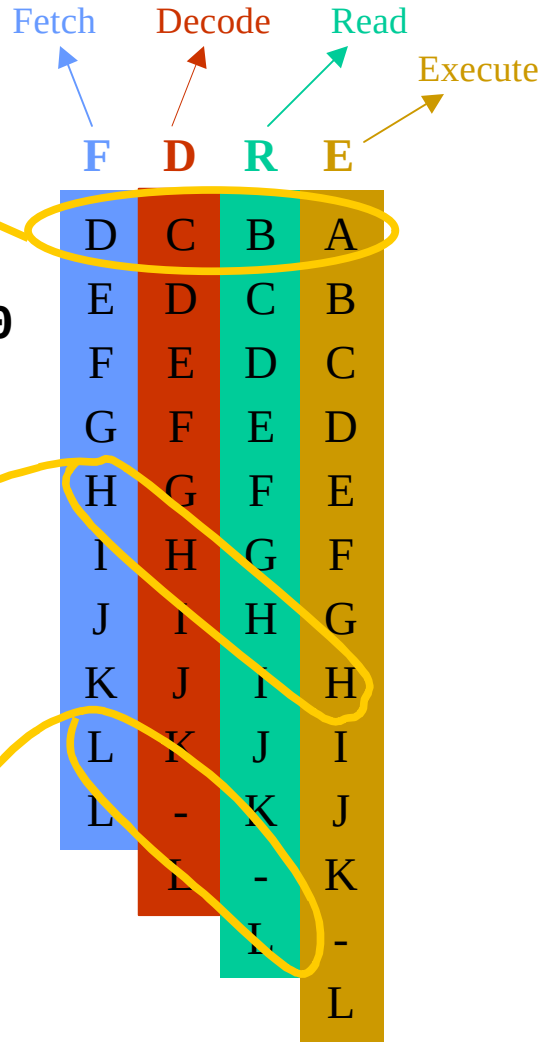
- Time-stationary pipeline model
  - Programmer controls each cycle
  - Motorola DSP56001

```
MAC X0,Y0,A  X:(R0)+,X0 Y:(R4)-,Y0
```

- Data-stationary pipeline model
  - Programmer specifies data operations
  - TMS320C30/40

```
MPYF *++AR0(1),*++AR1(IR0),R0
```

- Interlocked pipeline
  - Programmer is “protected” from pipeline effects

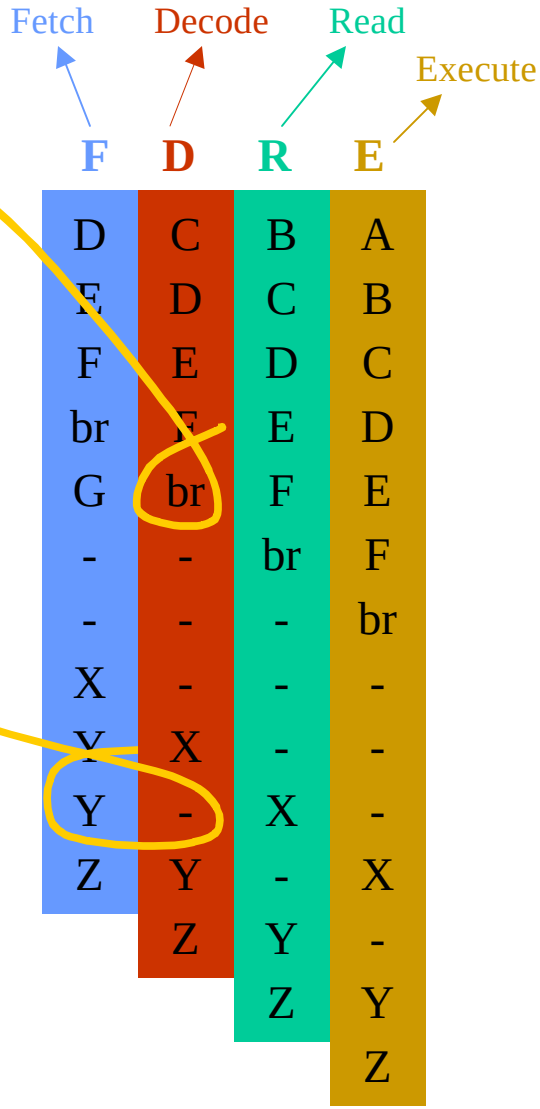


Pipelining: Hazards

- A control hazard occurs when a branch instruction is decoded
  - "Flush" the pipeline
  - or: Delayed branch (expose pipeline)
- A data hazard occurs because an operand cannot be read yet
  - Intended by programmer
  - or: Interlock hardware inserts "bubble"

TMS320C5x example

```
LAC #064h       LAR AR2, DATA
SAMM AR2        LACC *-
NOP
LACC *-
```

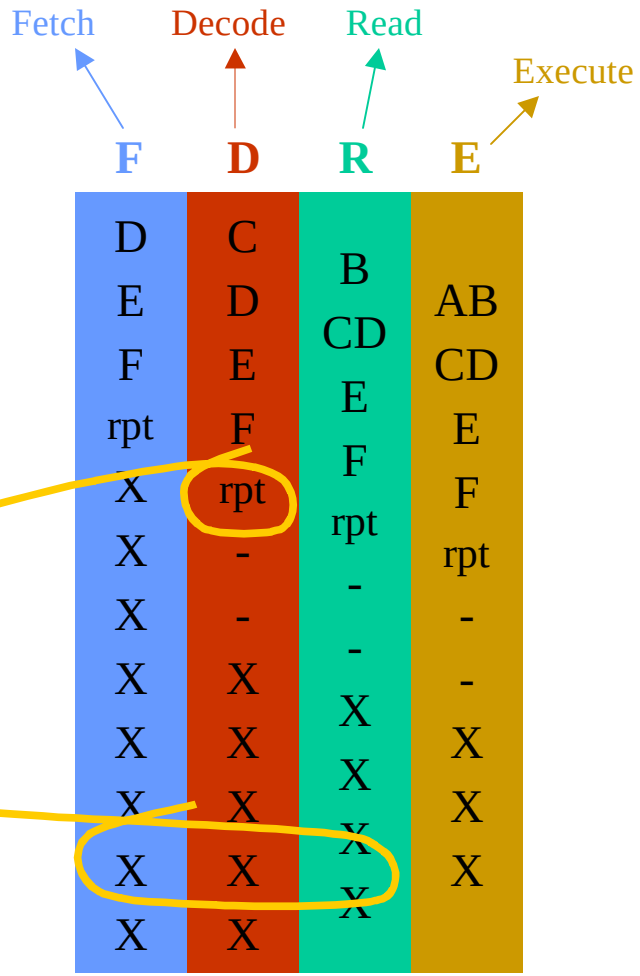


Pipelining: Avoiding Control Hazards

A key factor in the numeric performance of DSPs is the provision of special hardware to perform looping.

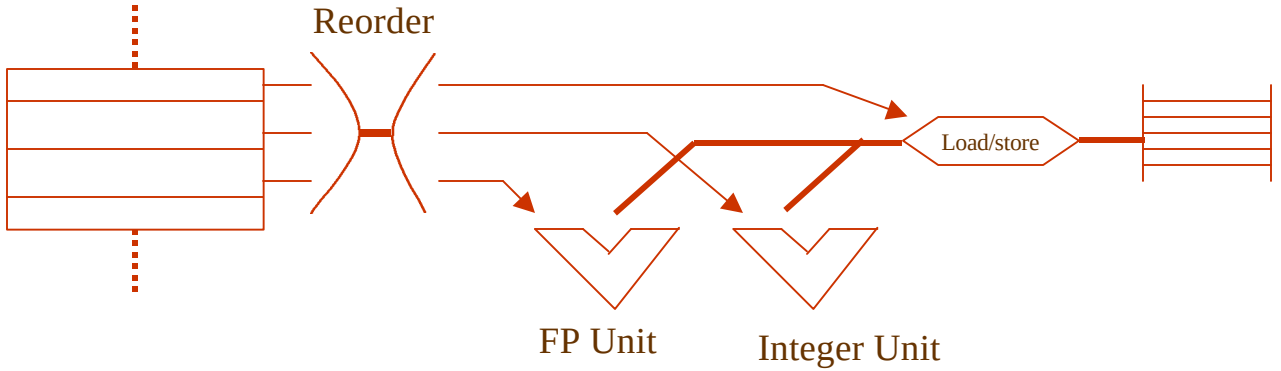
```
RPT COUNT
TBLR *+
```

- A repeat instruction repeats one instruction or a block of instructions after repeat
- The pipeline is filled with repeated instruction (or block of instructions)
- Cost: one pipeline flush only

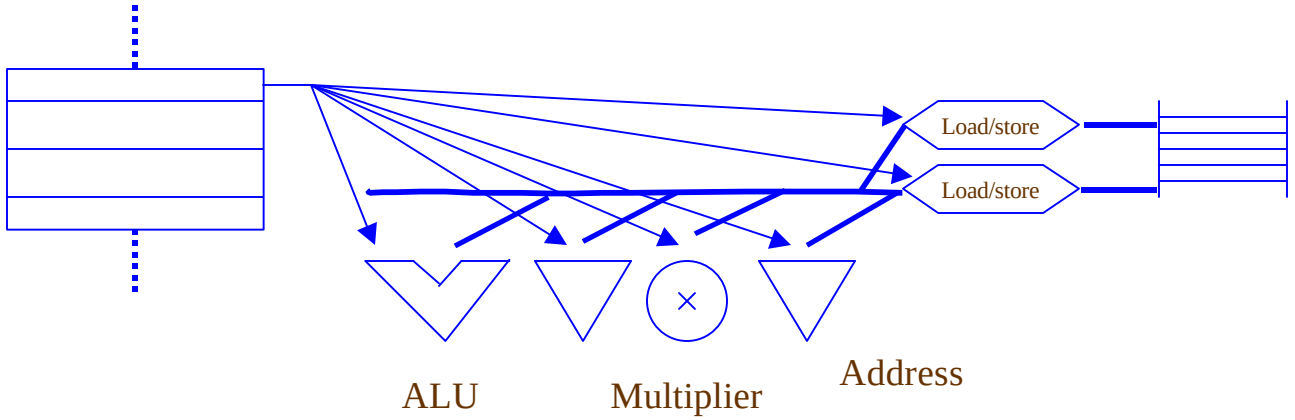


RISC vs. DSP: Instruction Encoding

- RISC: Superscalar

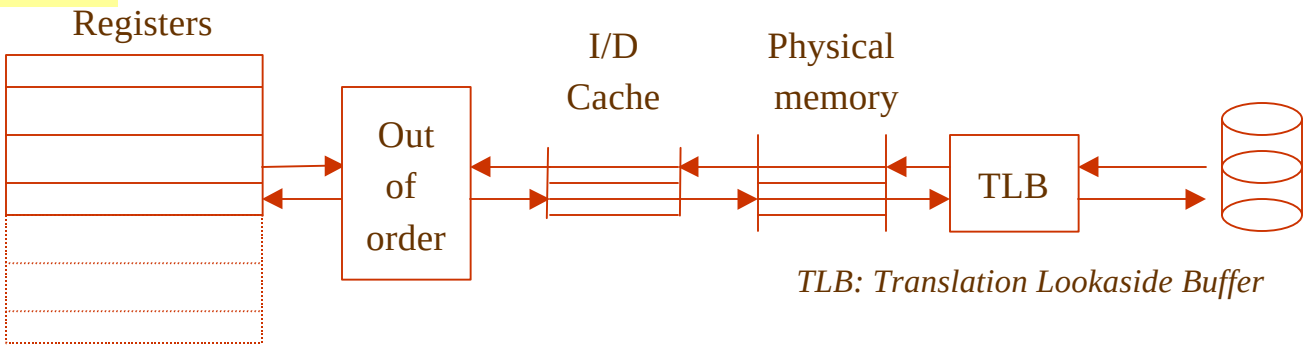


- DSP: Horizontal microcode



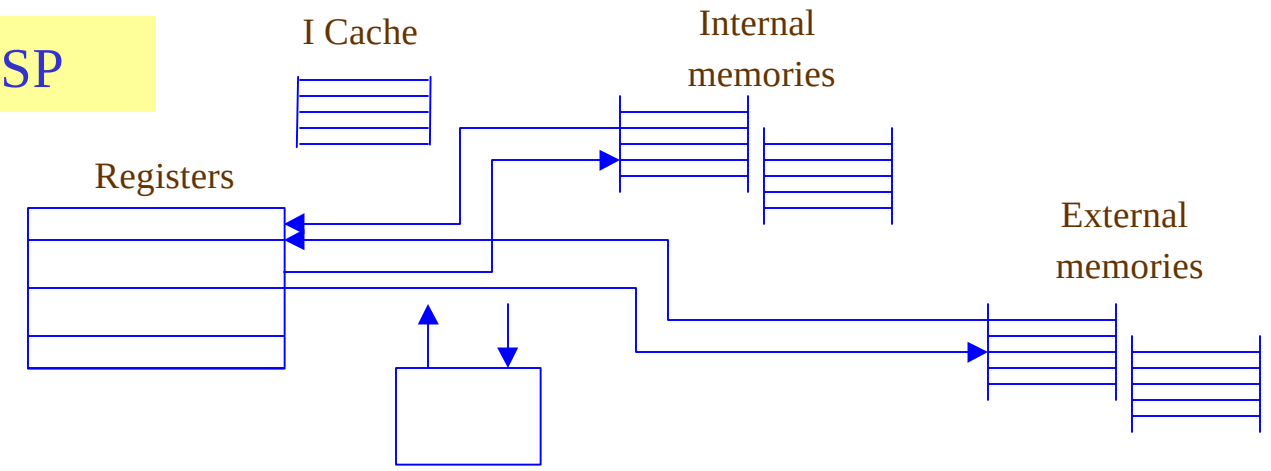
RISC vs. DSP: Memory Hierarchy

- RISC



TLB: Translation Lookaside Buffer

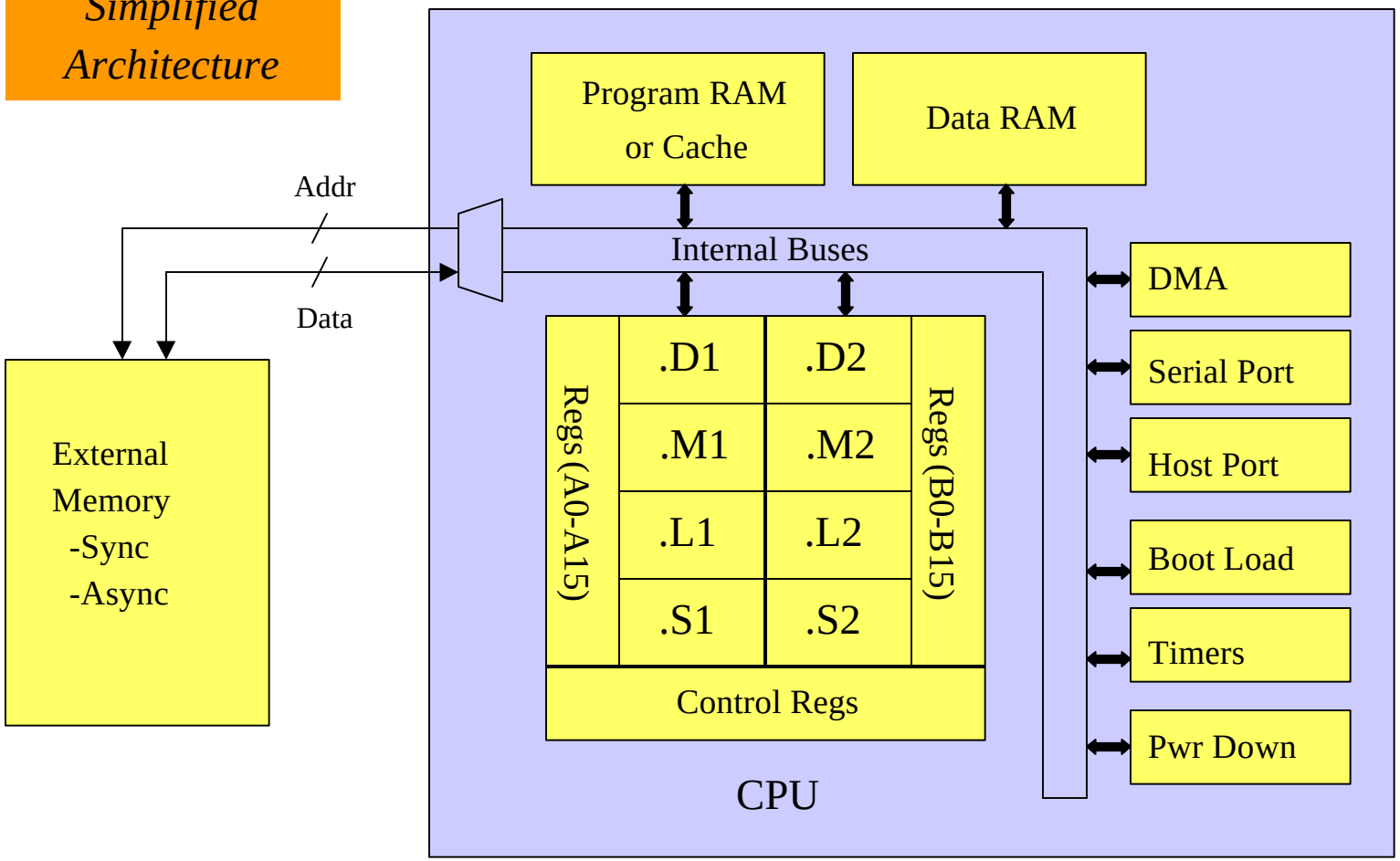
- DSP



DMA: Direct Memory Access

TI TMS320C6x VLIW DSP Architecture

Simplified Architecture



TI TMS320C6x VLIW DSP Architecture

- Two parallel data paths with single-cycle units:
  - Data unit - 32-bit address calculations (modulo, linear)
  - Multiplier unit - 16 bit x 16 bit with 32-bit result
  - Logical unit - 40-bit (saturation) arithmetic & compares
  - Shifter unit - 32-bit integer ALU and 40-bit shifter
- 16 32-bit registers in each data path
  - 40 bits can be stored in adjacent even/odd registers
- Fixed-point (C62x) and floating-point (C67x)
- TMS320C6201: $25 in volume
  - 150 MHz, 300 million MACs/sec, 1200 RISC MIPS
  - On-chip memory: 16 k x 32 program, 32 k x 16 data

TI TMS320C6x VLIW DSP Architecture

- One instruction cycle every clock cycle
- Deep pipeline
  - 7-11 stages in C62x: fetch 4, decode 2, execute 1-5
  - 7-16 stages in C67x: fetch 4, decode 2, execute 1-10
  - If a branch is in the pipeline, interrupts are disabled (the latency of a branch is 5 cycles)
  - Avoid branches by using conditional execution
- No hardware protection against pipeline hazards
  - Compiler and assembler must prevent pipeline hazards
- C67x computes floating-point multiply in 4 cycles

C5x and C6x Addressing Modes

- Immediate
  - The operand is part of the instruction
- Register
  - The operand is specified in a register
- Direct
  - The address of the operand is part of the instruction (added to imply memory page)
- Indirect
  - The address of the operand is stored in a register

TMS320C5x	*TMS320C6x*
`ADD #0FFh`	`add .L1 -13,A1,A6`
`(implied)`	`add .L1 A7,A6,A7`
`ADD 010h`	`not supported`
`ADD *`	`ldw .L1 *A5++[8],A1`

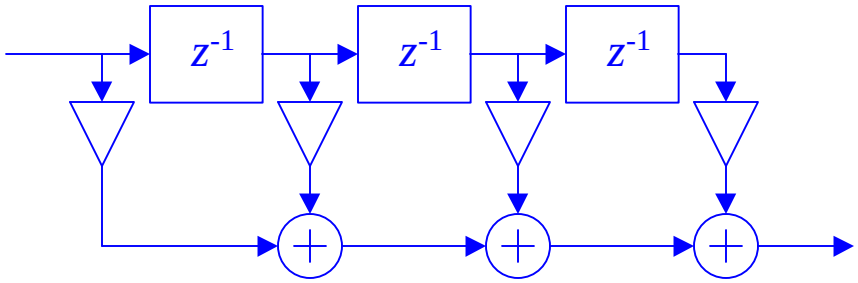
TMS320C6x vs. Pentium MMX

Processor	Peak MIPS	BDTI marks	ISR latency	Power	Unit Price	Area	Volume
Pentium MMX 233	466	49	1.14 μs	4.25 W	$213	5.5" x 2.5"	8.789 in^3
Pentium MMX 266	532	56	1.00 μs	4.85 W	$348	5.5" x 2.5"	8.789 in^3
C62x 150 MHz	1200	74	0.12 μs	1.45 W	$25	1.3" x 1.3"	0.118 in^3
C62x 200 MHz	1600	99	0.09 μs	1.94 W	$96	1.3" x 1.3"	0.118 in^3

BDTImarks: Berkeley Design Technology Inc. DSP benchmark results (larger means better) http://www.bdti.com/bdtimark/results.htm

http://www.ece.utexas.edu/~bevans/courses/ee382c/lectures/processors.html

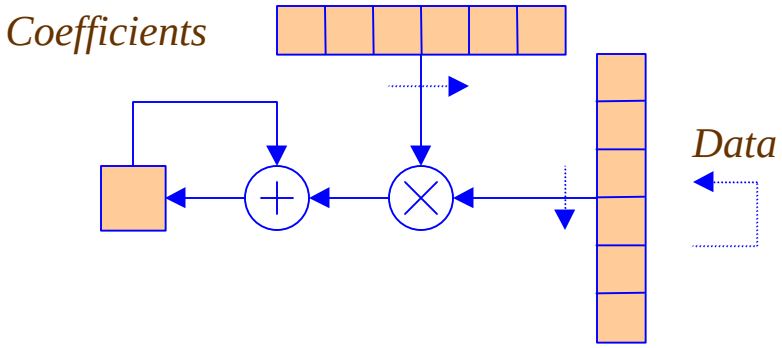
Application: FIR Filter



- Each tap requires
  - Fetching one data sample
  - Fetching one operand
  - Multiplying two numbers
  - Accumulating multiplication result
  - Shifting one sample in the delay line

- Computing an FIR tap in one instruction cycle
  - Three data memory accesses
  - Auto-increment or decrement addressing modes
  - Modulo addressing to implement delay line as circular buffer

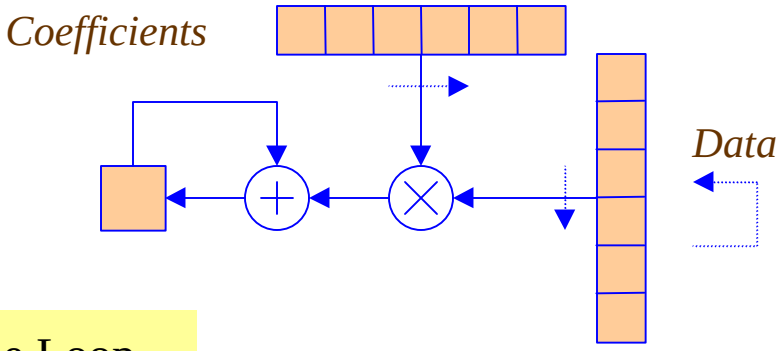
Application: FIR Filter on a TMS320C5x




```
COEFFP .set 02000h          ; Program mem address
X      .set 037Fh           ; Newest data sample
LASTAP .set 037FH           ; Oldest data sample

       …
       LAR AR3, #LASTAP     ; Point to oldest sample
       RPT #127
       MACD COEFFP, *-      ; Do the thing
       APAC
       SACH Y,1             ; Store result -- note shift
```

Application: FIR Filter on a TMS320C62x



Single-Cycle Loop

```
...

C7:       ldh .D1 *A1++,  A2   ; Read coefficient
 ||       ldh .D2 *B1++,  B2   ; Read data
 || [B0]  sub .L2  B0, 1, B0   ; Decrement counter
 || [B0]  B   .S2  c7          ; Branch if not zero
 ||       mpy .M1x A2, B2, A3  ; Form product
 ||       add .L1  A4, A3, A4  ; Accumulate result

...
```

Ordered Dithering on a TMS320C62x

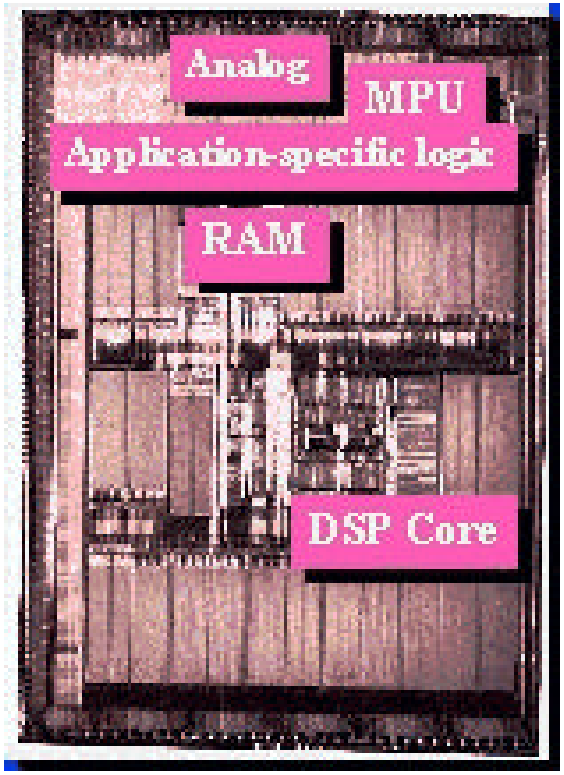
periodic array of thresholds

1/8	5/8	7/8	3/8
7/8	3/8	1/8	5/8

Throughput of two cycles

```
; remove next two lines if thresholds in linear array
        MVK    .S1 0x0001,AMR       ; modulo block size 2^2
        MVKH   .S1 0x4000,AMR       ; modulo addr reg B6
; initialize A6 and B6
        .trip  100                  ; minimum loop count
dith:   LDB    .D1 *A6++,A4         ; read pixel
||      LDB    .D2 *B6++,B4         ; read threshold
||      CMPGTU .L1x A4,B4,A1        ; threshold pixel
||      ZERO   .S1  A5              ; 0 if <= threshold
  [A1]  MVK    .S1  255,A5          ; 255 if > threshold
||      STB    .D1  A5,*A6++        ; store result
||[B0]  SUB    .L2  B0,1,B0         ; decrement counter
||[B0]  B      .S2  dith            ; branch if not zero
```

DSP Cores



- ASIC with:
  - Programmable DSP
  - RAM
  - ROM
  - Standard cells
  - Codec
  - Peripherals
  - Gate array
  - Microcontroller

DSP on General Purpose Processors

- Multimedia applications on PCs
  - Video, audio, graphics and animation
  - Repetitive parallel sequences of instructions
- Native signal processing examples
  - Sun Visual Instruction Set (UltraSPARC 1/2)
  - Intel MMX (Pentium I/II/III)
  - Intel Concurrent SIMD-FP (Pentium III)
- Single Instruction Multiple Data (SIMD)
  - One instruction acts on multiple data in parallel
  - Well-suited for graphics

DSP on General Purpose Processors (con't)

- Programming is considerably tougher
  - C/C++ compilers do not generate native signal processing code except Metrowerks CodeWarrior 5 gives MMX code
  - Libraries of routines using native signal processing
  - Hand code using in-line assembly for best performance
  - Pack/unpack data not aligned on SIMD word boundaries
  - 50-cycle penalty to switch to MMX; 0 penalty for VIS
  - Saturation arithmetic in MMX; not supported in VIS
  - Extended-precision accumulation in MMX; none in VIS
- Speedup for applications
  - Signal and image processing - 1.5:1 to 2:1
  - Graphics - 4:1 to 6:1 (no packing/unpacking)

Intel MMX Instruction Set

- 64-bit SIMD register (4 data types)
  - 64-bit quad word
  - Packed byte (8 bytes packed into 64 bits)
  - Packed word (4 16-bit words packed into 64 bits)
  - Packed double word (2 double words packed into 64 bits)
- 57 new instructions
  - Pack and unpack
  - Add, subtract, multiply, and multiply/accumulate
- Saturation and wraparound arithmetic
- Maximum parallelism possible
  - 8:1 for 8-bit additions
  - 4:1 for 8 x 16 multiplication or 16-bit additions

Concluding Remarks

- Conventional digital signal processors
  - High performance vs. power consumption/cost/volume
  - Excel at one-dimensional processing
  - Per cycle: 1 16x16 MAC & 4 16-bit RISC instructions
- TMS320C6x VLIW DSP
  - High performance vs. cost/volume
  - Excel at multidimensional signal processing
  - Per cycle: 2 16x16 MACs & 4 32-bit RISC instructions
- Native Signal Processing
  - Available on desktop computers
  - Excels at graphics
  - Per cycle: 2 8x16 MACs OR 8 8-bit RISC instructions
- In-line assembly code for best performance

Concluding Remarks

- Digital signal processor market
  - 40% annual growth rate since 1990
  - $3.5 billion revenue in 1998
  - 45% TI, 25% Lucent, 10% Motorola, 8% Analog Devices
- Independent benchmarking by industry
  - Berkeley Design Technology Inc. http://www.bdti.com
  - EDN Embedded Microprocessor Benchmark Consortium http://www.eembc.org
- Web resources
  - comp.dsp newsgroup: FAQ www.bdti.com/faq/dsp_faq.html
  - embedded processors and systems: www.eg3.com
  - on-line courses and DSP boards: www.techonline.com

References

1 G. E. Allen, B. L. Evans, and D. C. Schanbacher, "Real-Time Sonar Beamforming on a Unix Workstation," *Proc. IEEE Asilomar Conf. On Signals, Systems, and Computers*, pp. 764-768, 1998.
http://www.ece.utexas.edu/~bevans/papers/1998/beamforming/

2 R. Bhargava, R. Radhakrishnan, B. L. Evans, and L. K. John, "Evaluating MMX Technology Using DSP and Multimedia Applications," *Proc. IEEE Sym. On Microarchitecture*, pp. 37-46, 1998.
http://www.ece.utexas.edu/~ravib/mmxdsp/

3 W. Chen, H. J. Reekie, S. Bhave, and E. A. Lee, "Native Signal Processing on the UltraSPARC in the Ptolemy Environment," *Proc. IEEE Asilomar Conf. On Signals, Systems, and Computers*, 1996.
http://www.ece.utexas.edu/~bevans/courses/ee382c/lectures/21_nsp/vis/

4 B. L. Evans, "EE379K-17 Real-Time DSP Laboratory," UT Austin.
http://www.ece.utexas.edu/~bevans/courses/realtime/

5 B. L. Evans, "EE382C Embedded Software Systems," UT Austin.
http://www.ece.utexas.edu/~bevans/courses/ee382c/

6 A. Kulkarni and A. Dube, "Evaluation of the Code Generation Domain in Ptolemy," http://www.ece.utexas.edu/~bevans/talks/benchmarking97/sld001.htm

7 P. Lapsley, J. Bier, A. Shoham, and E. A. Lee, *DSP Processor Fundamentals*, IEEE Press, 1997.