

# DSP

# Microarchitctures

Wen-meï Hwu

ECE

University of Illinois,  
Urbana-Champaign

# What is DSP

- Embedded microprocessors that are designed to handle digital signal processing applications in a very cost effective manner
- Current market leaders
  - TI, Motorola, Lucent
- 1998 market - \$3.5Billions

# Nature of DSPs

- DSPs utilize special hardware to meet performance, power, and price points
  - Sacrifice orthogonality and ease-of-use to meet goals
  - Assume hand-assembly or libraries used for core algorithms
  - Compiler mostly used for control and glue logic

# Common DSP Features

- Circular addressing
  - address register automatically wraps when reaching a pre-specified bound
  - bound or block size registers selected on an instruction basis
  - bounds not necessarily powers of 2
  - works for both increment and decrement, zero always lower bound

# Common DSP Features

- Bit reversed addressing
  - first specify a bit position  $B$
  - reverse order of  $[B, \dots, 0]$  into  $[0, \dots, B]$
  - designed for some FFT algorithms where final results must be derived from a scrambled order
  - requires padding if data set size not power of 2

# Common DSP Features

- Multiple on-chip memory banks (bandwidth, RAM/ROM)
  - X, Y memory
- Fractional integer types
  - implied exponent, must be managed by the programmer
  - just integer arithmetic
  - need explicit post-shift after mul/div

# Common DSP Features

- Saturating arithmetic
  - values stay at max if incremented past max
  - values stay at min if decremented past min
  - minimize effect of overflow, clamping effect on real signals

# Common DSP Features

- Hardware loops
  - loop instruction to specify
    - loop iteration count
    - range of instructions in the loop
  - no need for explicit instructions for incrementing loop counters
  - no need for loop back branches, delay slots, branch prediction

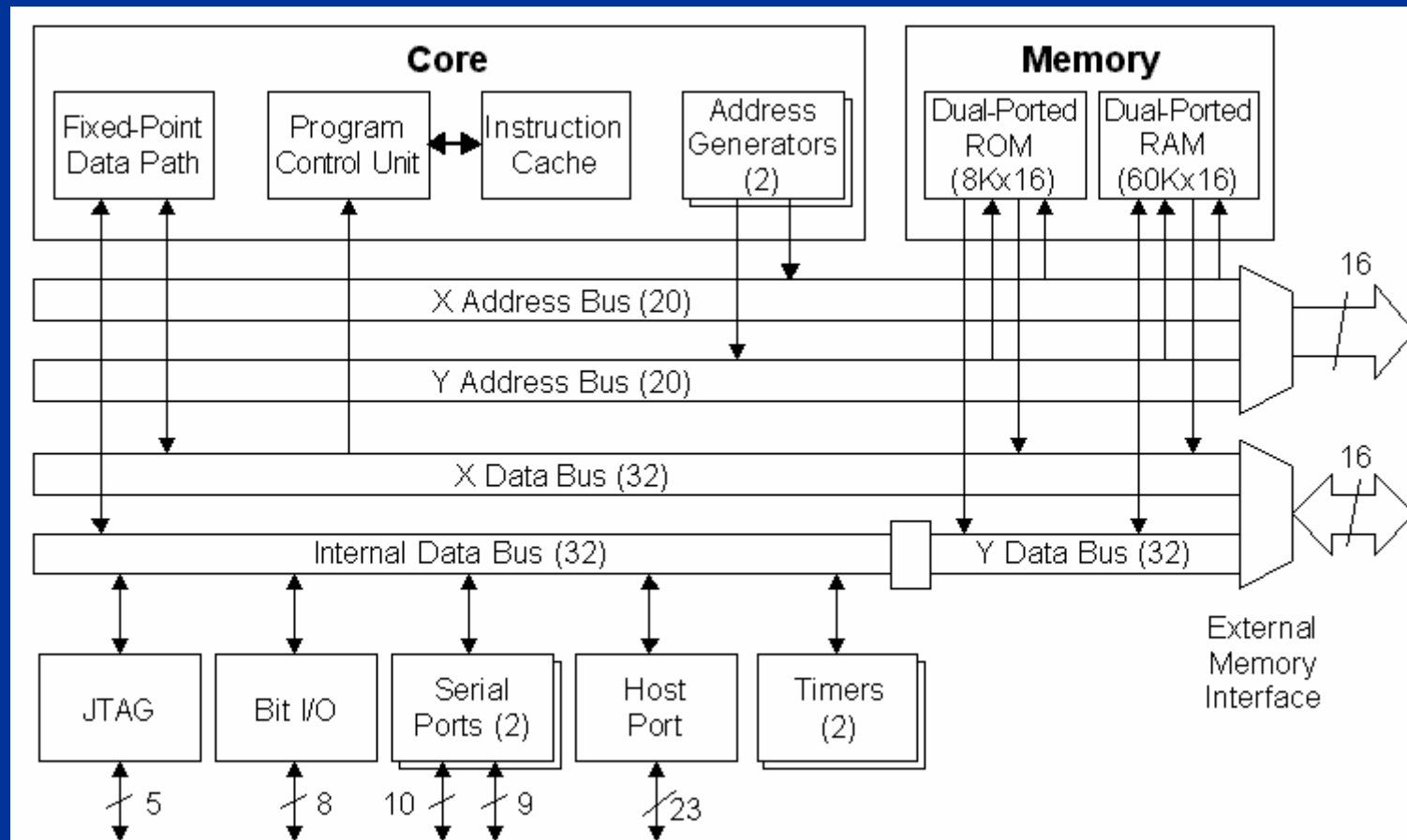
# Common DSP Features

- Special, highly parallel op instructions
  - target specific algorithms such as FIR filters
  - e.g., parallel memory accesses, address increments, with multiply-accumulate, store all in one instruction

# Common DSP Features

- Integrated I/O, special hardware
  - standard embedded processor peripherals
  - DSP specific accelerators such as Viterbi decoders

# Lucent DSP16210



# Closing the Compiler Gap

- Language extensions
  - Abstractions for special addressing modes, saturating arithmetic
  - Alignment and memory bank specifiers in variable declarations
  - Fractional integer types
  - Inline assembly

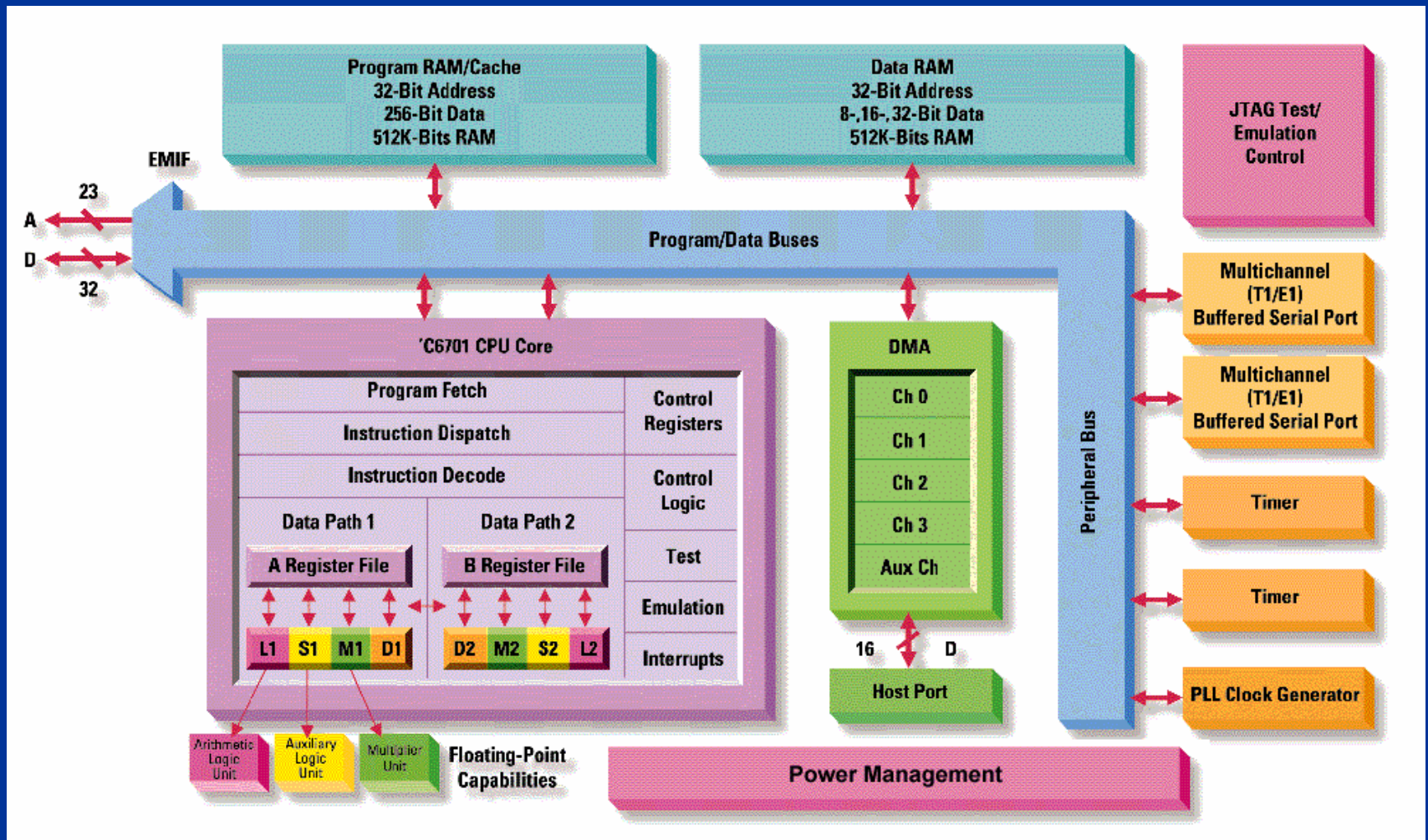
# Desired Compiler Support

- Make extensions first-class citizens (including inline assembly)
  - Tuned and targeted optimizations that utilize all the DSP features
  - Do all the tedious things well so the programmer doesn't have to
    - Interprocedural analysis, register allocation, scheduling

# Closing the Compiler Gap

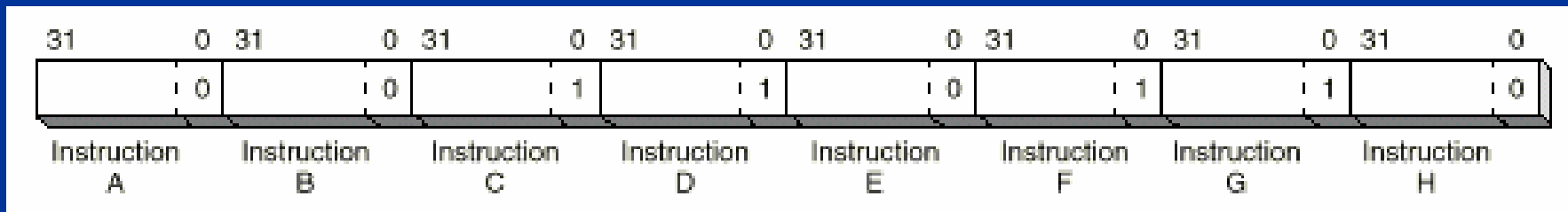
- More versatile architectures (promising trend for compilers)
  - VLIW, orthogonal instruction set, predication, speculation
  - Puts more tools in compilers hands with less restrictions
  - Trades off memory, power, and cost

# TI TMS320C67x



# TMS320C62x/C67x

- VLIW architecture with eight operations/cycle
  - Eight RISC operations per “fetch packet” (aligned on 256-bit boundary)
  - Multiple “execution packets” possible per fetch packet (operation’s p-bit)



# TMS320C62x/C67x

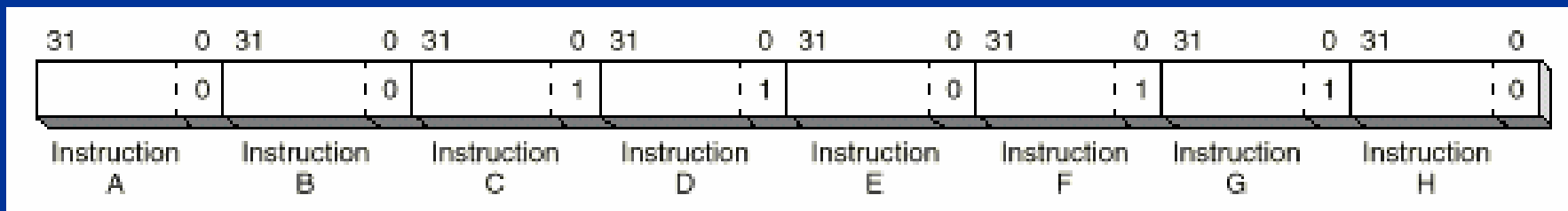
- May branch into middle of fetch packet or even execution packet
- Two clusters, each with 16 32-bit registers, 2 cross-cluster register bypass
- Units/datapaths explicitly specified (simple decode/issue)

# TMS320C62x/C67x

- EQs register model, no interlocking (stalls on cache miss/bank conflict)
  - Interruptible code portions must not have multiple assignments in flight
  - NOPs primarily used to provide latency (multicycle NOPs allowed)
- NOPs also used for fetch packet alignment before entering loop kernels

# Code Example

- Example fetch packet with four execution packets (A,B,CDF,FGH):



# Conditional Execution

- All instructions can be conditional on a register's value
  - Conditional branches are 'conditions' on an unconditional branch
  - Resources reserved even if instruction squashed (cannot send mutually exclusive operations to same function unit)

# Conditional Execution

- Tests all 32 of the register bits for zero or non-zero (based on z-bit)
- Three bits specify register, one bit (z) specifies zero/non-zero
  - 10 patterns for 5 registers assigned (B0, B1, B2, A1, A2)
  - 1 pattern for Unconditional execution specification

# Conditional Example

- 5 of 16 patterns (3+1 bits) reserved for future expansion
- Example (mutually exclusive parallel operations)
  - [B0] ADD .L1 A1, A2, A3  
|| [!B0] ADD .L2 B1, B2, B3
- Example, implementation of  $A1 = !(A1)$  (set A1 to 1 if non-zero)
  - [A1] MVK .S1 1, A1

# Burdens on Programmer Optimizer

- All latencies exposed and fully pipelined (including branches)
  - 6 cycle branches, 5 cycle loads, 4 cycle float add/multiply, 2 cycle int multiply, 1 cycle int add

# Modulo Scheduling

- Highly recommended for C60
  - Overlaps loop iterations for effective resource utilization
  - Unrolling inner loop can allow for fractional effective iteration intervals (II)

# Modulo Scheduling (cont.)

- Can take advantage of EQ register model to reduce register pressure and reduce need for modulo register renaming (or rotating register files)
- Significantly reduce time over straightforward execution of loop body

# Modulo Scheduling Example

- Dot product (two 100 element 16-bit arrays) from TI documentation
  - 1602 cycles for straightforward loop execution (16 cycles an iteration)
  - 58 cycles for optimized software pipelined loop (1/2 cycles an iteration)
- Prolog and epilogue can cause significant increase in code size

# C6x DSP-Specific Features

- Saturation arithmetic
  - enabled per register
- Circular addressing
  - two block sizes, must be power of 2
  - enabled per register
- Bit counting operations
- Bit-field extraction, set, clear

# DSP Specific Features

- Normalization support
  - make fractional number operations more like floating point
- 8, 16, and 32 bit data supported
- 8-bit overflow protection (40-bit integers using two registers)

# Missing DSP-specific Features

- (No reverse bit addressing)
- (No hardware loops)

# Where skilled designers are still needed

- Manipulation of algorithm and data structures to fit architecture features and system design constraints
  - Changing algorithm to use circular addressing
  - Choice of algorithm variant (radix-2, radix-4, split radix, etc.)

# Programmer Skills

- Optimizations that require utilization of special properties of algorithm (specialized butterflies to reduce multiplies and adds in FFT)
- Mapping algorithms to fixed point processors
- Speed/size/power tradeoffs