Benchmarks

Several of the leading manufacturers (Dec, HP, MIPS, Motorola, ...) have developed a set of **benchmark** programs to be used under **specified conditions**.

- ♦ known as:
  - SPECint92 and SPECfp92
  - for integer and floating point operations respectively
- ♦ both benchmarks are the geometric mean of the speedup achieved on each of N programs versus a VAX 11/780
- ♦ VAX 11/780 chosen for historical reasons:
  - it was one of the most popular minicomputers of the 1980s

These benchmarks are less ambiguous than the “measures” MIPS, MOPS or Mflops, **but** are based on a UNIX environment and thus:

- ♦ indicate potential performance of a processor
- ♦ may not be applicable to real-time systems

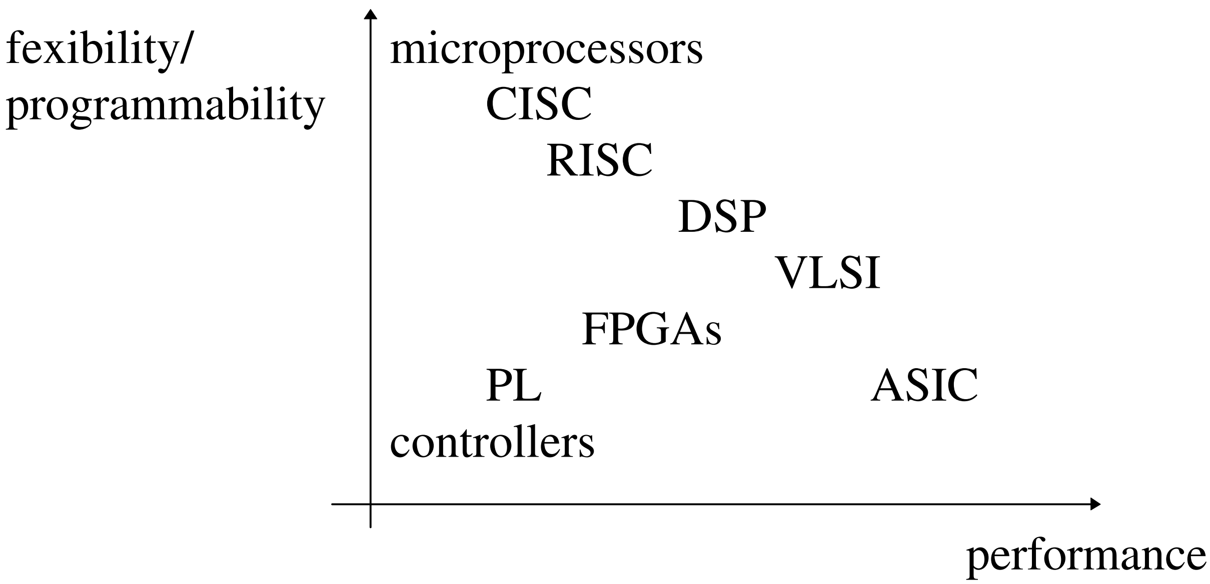
Processors

A large variety of devices is available, ranging from high performance *microprocessors* to VLSI devices and ASICs (application specific integrated circuits), **but**

- ♦ range of choices is not a linear spectrum as devices should be characterised in many ways, the most important of which are often:
  - performance
  - flexibility/programmability
  - cost
- ♦ for specific applications, other factors may be paramount, such as:
  - I/O interfaces
  - power supply/consumption
  - environmental factors
  - software support
  - development tools
  - re-programmability

Range of devices

The main choices are roughly:



Key

- CISC = conventional instruction set computers
  * M68000 etc
- RISC = reduced instruction set computers
  * Sparc
- DSP = digital signal processors
  * M56000, M96000, TMS320C30, TMS320C40
- FPGA = field programmable gate array
  * Xilinx, Altera, ...
- PC = programmable controller

System architecture

In addition to choosing the processor, consideration has to be given to the system architecture, in particular to:

- ♦ which resources should be dedicated and which are to be shared, including
    - processor
    - memory
    - storage (if applicable)
    - communication links
    - code

Dedicated systems are often the simplest and most attractive, but they may not be:

- cheap enough
- flexible enough
    - * eg, ability to introduce new processes after the system has been built
- reliable enough
    - * eg, tolerant to hardware faults
    - * redundant enough

In practice

- look at propaganda figures such as:
  - MIPS, MOPs, Mflops, etc, but treat with caution
- analyse spec. sheets carefully for:
  - clock rate, co-processor, pipelining, etc
- look at SPECmarks
- test and measure performance on one's *own* system on one's *own* code
  - eg: for a number of machines

Familiar Processors

- the Motorola families
  - cf other courses
  - includes
    - CISC - 68000, 68020, ...
    - RISC - 88000
    - DSP - 56000, 96000
- see eg: Heath, “Microprocessor architectures and systems”, RISC, CISC and DSP”, B H Newns, 1991.
- important features for real-time systems include:
  - speed
  - pipelining
  - co-processors
  - separate address/data-buses
  - support for interrupts
  - separate user and supervisor states
    - separate and protected resources
    - user normally executes code
    - switches to supervisor when an exception occurs (instead of failing)

CISC machines (M68000)

- instructions decoded internally into microcode and nanocode sequences (not accessible to programmers) that are pipelined by splitting fetch, decode and execution into independent stages, eg:

- but, pipeline stalls when any one stage cannot complete its task on time:

RISC machines

Why RISC? 80/20 (90/10) rule for CISC:

- ♦ 80% of the instructions generated and executed use only 20% of the available instruction set
- ♦ this 20% comprises the "simpler" instructions =>
  - implement only these, reduce complexity and increase performance by putting more of the computer system "on-chip"

RISC

- ♦ all instructions executed in single cycle
- ♦ memory accessed only via load and store instructions =>
  - instructions do not manipulate memory directly as this can take many cycles
  - all execution units are hardwired with no microcoding
  - Harvard architecture
- ♦ to cope with the code expansion => need
  - good compilers
  - fast memory

RISC machines: M88000 family

- ♦ concurrent:
  - integer
  - FP (addition and multiply)
  - data
  - instruction fetch
- ♦ pipelined
  - > 220 MIPS @ 20 Mhz claimed
- ♦ needs compiler capable of optimizing the pipeline
- ♦ separate USER and SUPERVISOR states
- ♦ designed for dual processor, fault tolerant operation

For image processing:

- ♦ all but the simplest tasks need:
  - > 20-50 MHz machines
    (often much more required)
  - RISC rather than CISC
  - several recent machines offer useful performance:
    - * MIPS, Dec alpha, pentium, power, ...

New 64 bit RISC machines

see: IEEE Spectrum, June 1995

DSPs: Digital Signal Processors

Recall FIR filtering:

$$y[n] = \sum_{i=0}^{m} g_i x[n-i]$$

=>

- repeated multiply-accumulate
- need rapid access to filter coefficients, g_i and to past data, x[n-i]
- need high-speed I/O

CISC: (eg 68000)

- multiply takes > 40 clock cycles
- single bus for all fetches

RISC

- single cycle arithmetic
- simultaneous data and instruction fetches
- more dependable loop timings

♦ **But:**

- at high cost (~general purpose)
- DSP architecture specifically enhanced for the task

Basic DSP architecture

- ♦ enhanced Harvard architecture internally
  - X,Y data & address buses
  - program & address buses
  - global data bus
- ♦ enhanced word length
  - eg: 24 bit in M56000
- ♦ multiply-accumulate (MAC) in a single machine cycle
- ♦ concurrent moves to update data for next cycle
- ♦ internal parallelism
  - single cycle takes ~ 97 ns for:
    - * 24 x 24 bit multiply
    - * 56 bit addition
    - * 2 data bus operations
    - * 2 address pointer updates
  - => > 60 MIPS
- ♦ special addressing and loop modes
  - circular buffer
- ♦ hardware "rep" statement