Trimaran
History and Motivation

Terminology

EPIC: Explicitly Parallel Instruction Computing

Architectural philosophy and technology:	RISC	EPIC	EPIC
Specific architecture and ISA:	PA-RISC	HPL-PD	IA-64
Implementation:	PA-8500	–	Merced™

The Motivation for EPIC

- In 1989, we at HPL believed that within the next 10 years:
  - a high ILP processor would fit on a chip
  - superscalar complexity would be an obstacle to sustaining Moore's Law
- Achieve high levels of ILP
  - the ability to issue over eight useful operations per cycle
- Retain hardware simplicity and short cycle times even at high levels of ILP
  - avoid schemes that force hardware to make complex decisions at run-time
- True general-purpose capability
  - "scientific" computations as well as "scalar" computations, i.e., code with a high frequency of conditional branches and pointer-based memory accesses

The EPIC philosophy

- Provide the facility to design the desired record of execution (ROE) at compile-time
  - Generalize VLIW's philosophy of compile-time scheduling and resource allocation: which operations? what time? which resources? which registers?
  - Features that provide greater program (compiler) control over microarchitectural capabilities
  - Features that assist in reducing the critical path through "scalar" computations
  - Features that permit one to "play the statistics"
- Provide the ability to communicate the desired ROE to the hardware
  - Maintain run-time transparency, i.e., "obedient" hardware
  - MultiOp, adequate architectural registers, rotating registers, non-unit assumed latencies (NUAL)
- Provide the ability to freeze virtual time during execution in response to unexpected dynamic events

Key features of HPL-PD

Features	Design Record of Execution	Communicate Record of Execution
MultiOp	x	x
Non-unit assumed latencies (NUAL), ELRs, latency stalling	x	x
Predication	x	
Compare-to-predicate	x	
Control speculative opcodes / exception tags	x	
Data speculation	x	
Prepared branches	x	
Long latency branches	x	x
Branch prediction control		x
Parallel multi-way branching	x	
Software pipelining branches	x	x
Rotating registers		x
Cache latency control	x	x
Cache hierarchy promotion control	x	x

New challenges in EPIC compilation

- Designing the desired ROE, exploiting the features of EPIC
- Managing the cache hierarchy
- The figure of merit is the schedule length, not the number of operations executed
  - Reduce the *length of the critical path* through the computation
  - Often, the critical path can be shortened by *increasing* the number of operations executed
- Statistical analysis, optimization and transformation
- Analysis of predicated code, i.e., code without a control flow graph
- Region-based compilation
- Machine description-driven ILP compilation

The Genesis of Trimaran

- Joint research partnership with the University of Illinois' IMPACT project [1991]
- Development of Elcor [Nov. 1993]
- Leveraging of the IMPACT compiler
- Injection of compiler ideas into IMPACT
- HPL-PD architecture specification published [Feb. 1994]
- The ReaCT-ILP project at NYU proposes the Trimaran project [Feb. 1996]
- Trimaran released [Aug. 1998]
  - HP Labs
  - The University of Illinois
  - New York University

This is a point of discontinuity

- EPIC represents a new philosophy of computing
  - Explicit parallelism
  - Unprecedented programmatic control over the resources of the machine
  - Architectural features that help in engineering the desired record-of-execution and in communicating it to the processor
  - The first architectural style to consciously focus on the reduction of the critical path through the computation
  - Capable of achieving high levels of ILP on a wide spectrum of applications
- Sophisticated architectures require sophisticated usage
  - EPIC uses advanced architectural features to exploit increasingly specialized properties of the workload
  - Sophisticated compilers are crucial for the effective use of EPIC
  - Trimaran and HPL-PD provide the ability to do EPIC compiler research