The Design Complexity of Program Undo Support in a General Purpose Processor

Radu Teodorescu and Josep Torrellas

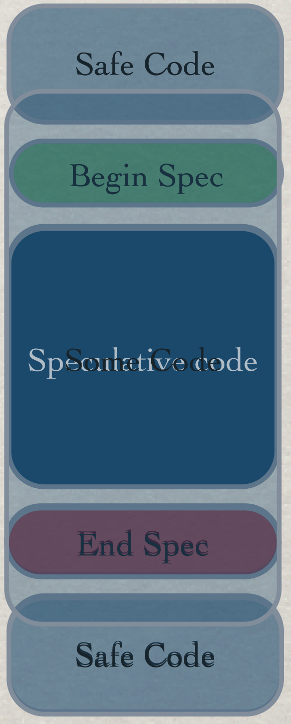
University of Illinois at Urbana-Champaign
http://iacoma.cs.uiuc.edu

Processor with program undo support

Safe

Speculative

- Speculative execution over large code sections
  - Rollback/re-play of program execution
  - Very low overhead
  - Speculation exposed to the software



Applications of program undo

- Safety net for speculating over:
  - correctness of aggressive optimizations:
    - thread level speculation, value prediction, speculative synchronization
  - system reliability:
    - software - primitive for software debugging

How complex is program undo support?

- Determined the hardware needed
- Implemented it in a simple processor
- Prototyped it using FPGA technology
- Estimated complexity using three metrics:
  - Hardware overhead, development time, VHDL code size

Implementation of program undo support

Safe

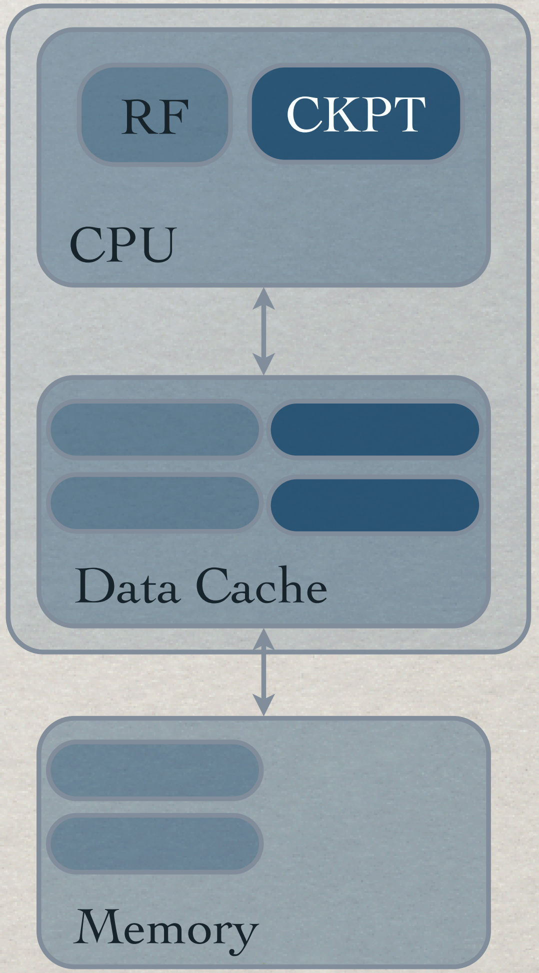
Speculative

- Save/restore processor state, buffer speculative data, control transitions:

1. Register checkpointing and restoration

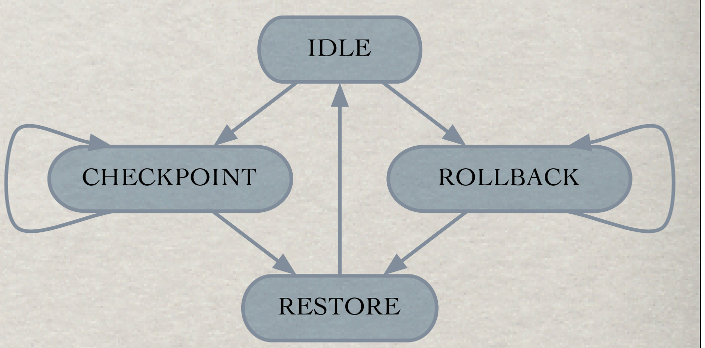
2. Data cache that buffers speculative data

3. Instructions enable/disable speculation on-the-fly



Register checkpointing

- Beginning of the speculative section
  - RF saved into a Shadow RF
  - PC, state registers are saved
- End of speculative section
  - Commit: discard checkpoint
  - Rollback: restore RF & PC

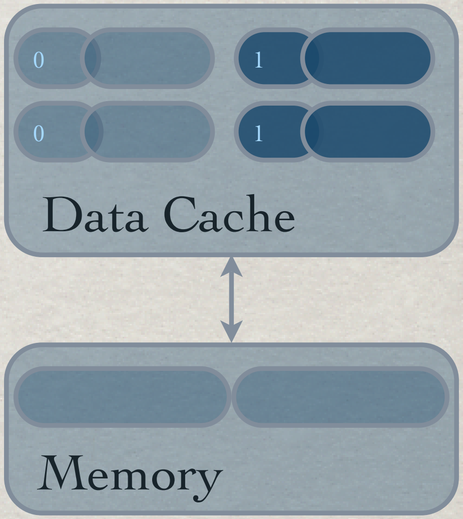


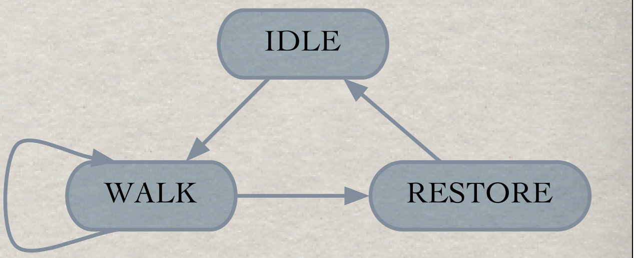
Safe

Speculative

Data cache extensions

- Holds both speculative and non-speculative data
- Speculative lines will not be evicted to memory
- Cache walk state machine:
  - Commit: merge lines
  - Rollback: invalidate speculative lines





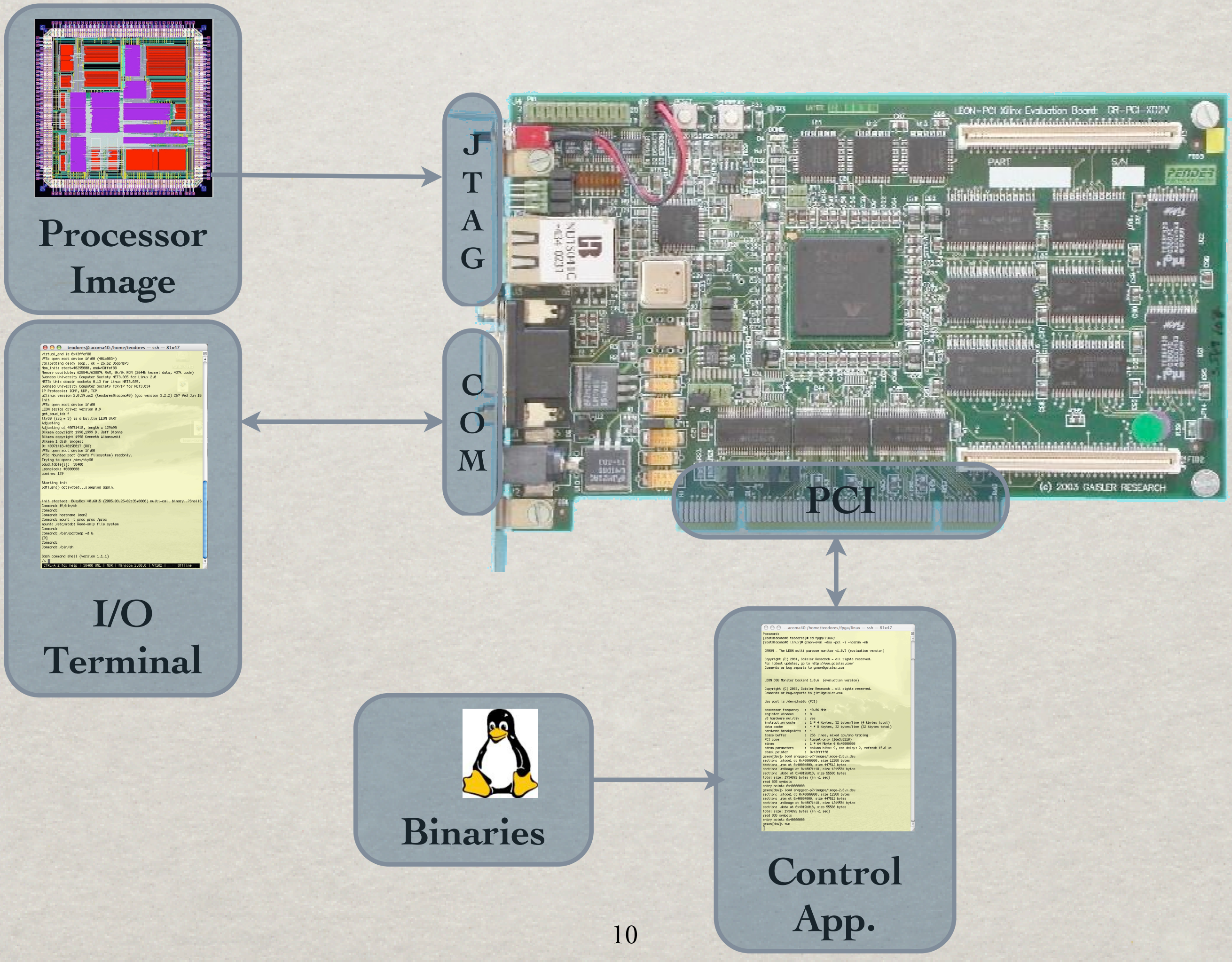
Software control

- Give the compiler control over speculative execution
- Added control instructions:
  - Begin speculation
  - End speculation (commit or rollback)
- We use SPARC's special access load
  - `LDA [r0] code, r1`

Hardware prototype

- LEON2 - SPARC V8 compliant processor
- In-order, single issue, 5-stage pipeline
- Windowed register file
- L1 instruction and data caches
- Synthesizable, open source VHDL code
- Fully functional, runs Linux embedded

System deployment


Processor Image
JTAG
I/O Terminal
COM
PCI
LEON-PCI Xilinx Evaluation Board
(c) 2003 GAISLER RESEARCH
Binaries
Control App.
teodores@iacoma40:/home/teodores — ssh — 81x47
...acoma40:/home/teodores/fpga/linux — ssh — 81x47

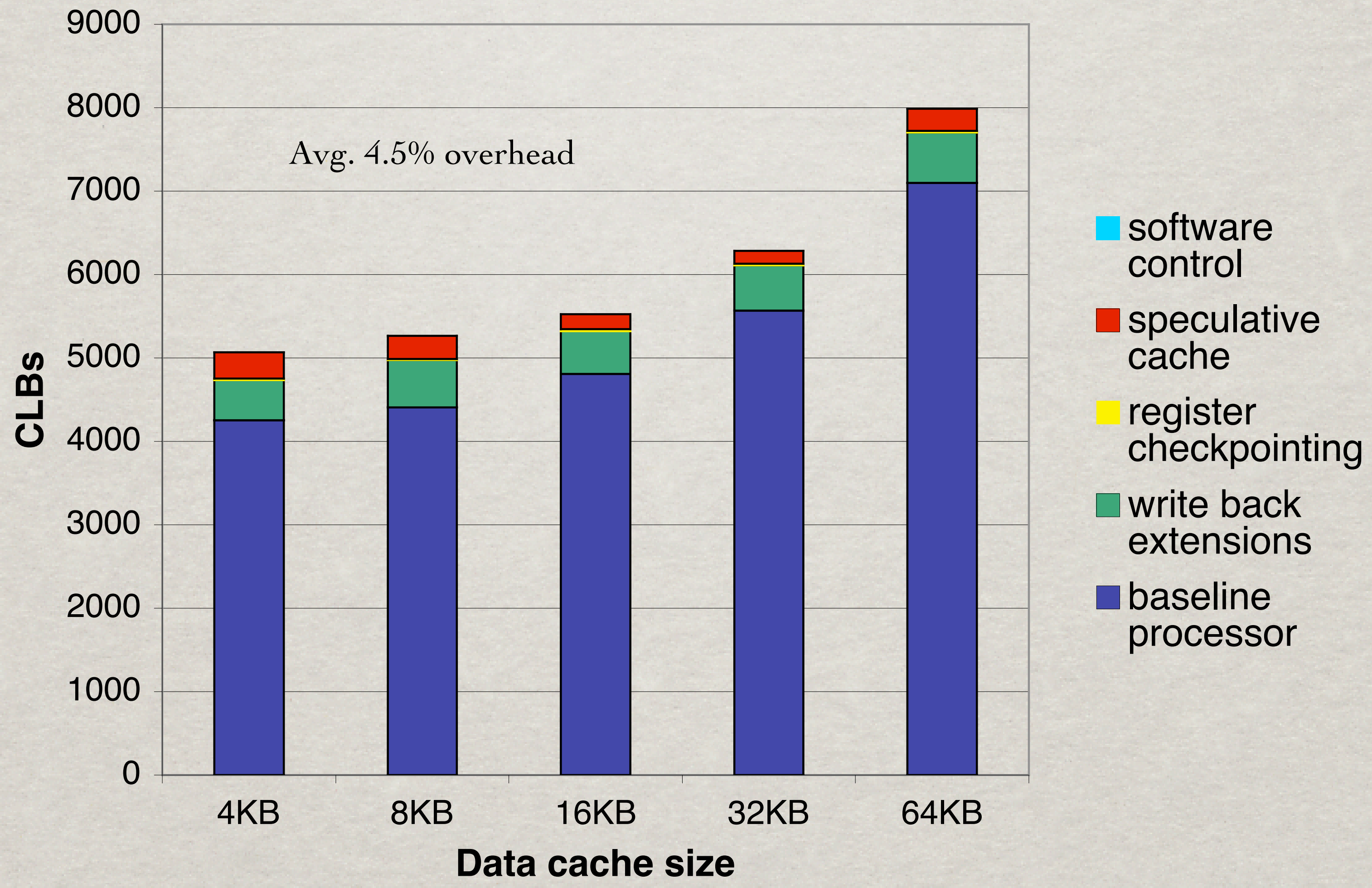
```
virtual_end is 0x43ffef88
VFS: open root device 1f:00 (401c0834)
Calibrating delay loop.. ok - 26.52 BogoMIPS
Mem_init: start=40295000, end=43ffef88
Memory available: 62884k/63887k RAM, 0k/0k ROM (2644k kernel data, 437k code)
Swansea University Computer Society NET3.035 for Linux 2.0
NET3: Unix domain sockets 0.13 for Linux NET3.035.
Swansea University Computer Society TCP/IP for NET3.034
IP Protocols: ICMP, UDP, TCP
uClinux version 2.0.39.uc2 (teodores@iacoma40) (gcc version 3.2.2) 267 Wed Jun 15
Init
VFS: open root device 1f:00
LEON serial driver version 0.9
get_baud_id: f
ttyS0 (irq = 3) is a builtin LEON UART
Adjusting
Adjusting at 40071418, length = 129b90
Blkmen copyright 1998,1999 D. Jeff Dionne
Blkmen copyright 1998 Kenneth Albanowski
Blkmen 1 disk images:
0: 40071418-4019B017 (RO)
VFS: open root device 1f:00
VFS: Mounted root (romfs filesystem) readonly.
Trying to open: /dev/ttyS0
baud_table[i]:  38400
Leonclock: 40000000
comine: 129

Starting init
bdflush() activated...sleeping again.


init started:  BusyBox v0.60.5 (2005.03.25-02:35+0000) multi-call binary..?ShellS
Command: #!/bin/sh
Command:
Command: hostname leon2
Command: mount -t proc proc /proc
mount: /etc/mtab: Read-only file system
Command:
Command: /bin/portmap -d &
[9]
Command:
Command: /bin/sh

Sash command shell (version 1.1.1)
/> 
 CTRL-A Z for help | 38400 8N1 | NOR | Minicom 2.00.0 | VT102 |      Offline
```

```
Password:
[root@iacoma40 teodores]# cd fpga/linux/
[root@iacoma40 linux]# grmon-eval -dsu -pci -i -nosram -nb

 GRMON - The LEON multi purpose monitor v1.0.7 (evaluation version)

 Copyright (C) 2004, Gaisler Research - all rights reserved.
 For latest updates, go to http://www.gaisler.com/
 Comments or bug-reports to grmon@gaisler.com


 LEON DSU Monitor backend 1.0.6  (evaluation version)

 Copyright (C) 2003, Gaisler Research - all rights reserved.
 Comments or bug-reports to jiri@gaisler.com

 dsu port is /dev/phob0a (PCI)

 processor frequency   :  40.06 MHz
 register windows      :  8
 v8 hardware mul/div   :  yes
 instruction cache     :  1 * 4 kbytes, 32 bytes/line (4 kbytes total)
 data cache            :  4 * 8 kbytes, 32 bytes/line (32 kbytes total)
 hardware breakpoints  :  4
 trace buffer          :  256 lines, mixed cpu/ahb tracing
 PCI core              :  target-only (16e3:0210)
 sdram                 :  1 * 64 Mbyte @ 0x40000000
 sdram parameters      :  column bits: 9, cas delay: 2, refresh 15.6 us
 stack pointer         :  0x43fffff0
grmon[dsu]> load snapgear-p7/images/image-2.0.x.dsu
section: .stage1 at 0x40000000, size 12288 bytes
section: .rom at 0x40004000, size 447512 bytes
section: .rdimage at 0x40071418, size 1219584 bytes
section: .data at 0x4019b018, size 55508 bytes
total size: 1734892 bytes (in <1 sec)
read 835 symbols
entry point: 0x40000000
grmon[dsu]> load snapgear-p7/images/image-2.0.x.dsu
section: .stage1 at 0x40000000, size 12288 bytes
section: .rom at 0x40004000, size 447512 bytes
section: .rdimage at 0x40071418, size 1219584 bytes
section: .data at 0x4019b018, size 55508 bytes
total size: 1734892 bytes (in <1 sec)
read 835 symbols
entry point: 0x40000000
grmon[dsu]> run
```

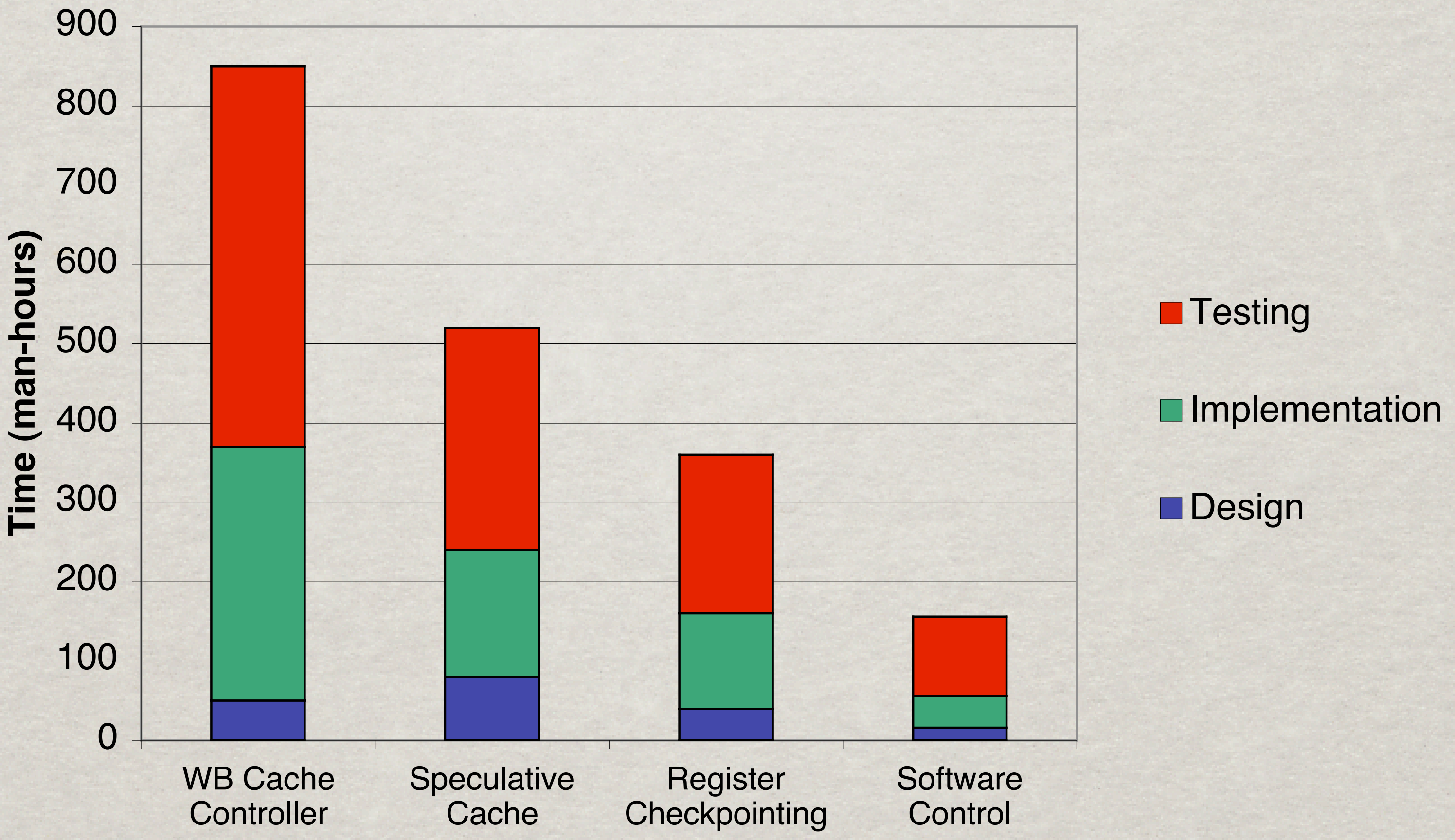
Evaluating design complexity

- Hardware overhead, development time, VHDL code size
- Major components:
  - register checkpointing
  - speculative cache
  - software control
- Comparison: write-back cache controller

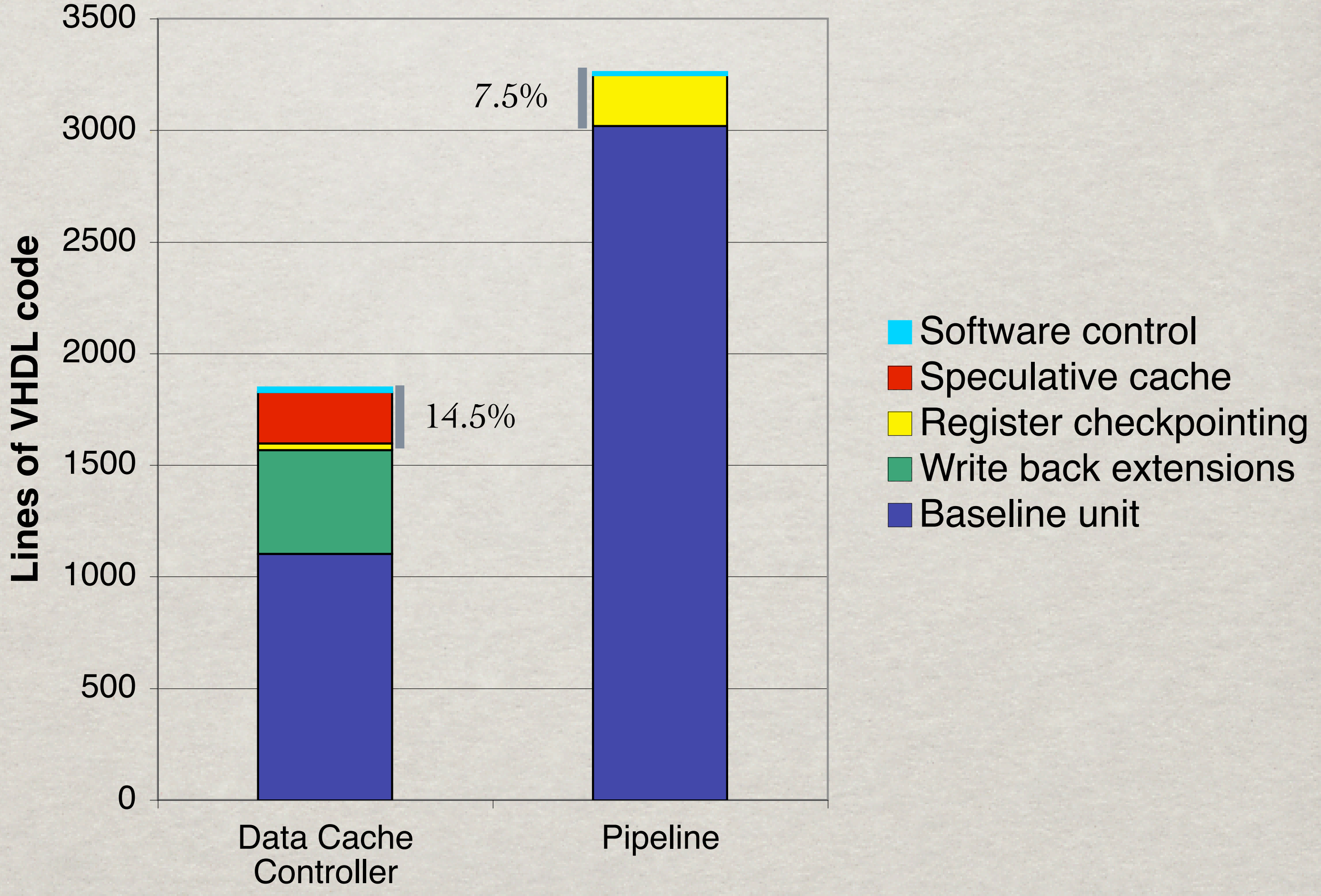
Hardware overhead



Development time



Lines of code



Conclusions

- Program undo support is reasonably easy to implement
- Complexity similar to adding write-back support to a write-through cache controller
- Qualifying factor:
    - Relatively simple processor

Thank you!

Short demo and questions...