Fix Spectre in Hardware!

Why and How

M. Anton Ertl, TU Wien

Why should I listen to this talk?

- Spectre: A security vulnerability of all processors with speculative execution
  Affected: all high-performance processors
- Known to processor manufacturers since June 2017
  Known to the public since January 2018
- New attacks found regularly since then
  e.g., DownFall and Inception published in August 2023
- Not fixed in hardware yet

What is architecture and microarchitecture?

```
        #r8=0x1000 r9=0xff8
0x2000  mov (%r9), %r10
0x2003  add 1, r10
0x2007  mov %r10, (%r8)
0x200a
```

registers

PC=0x200a

...

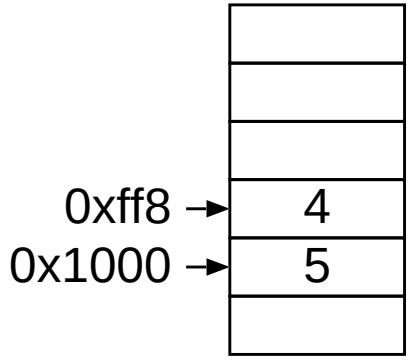
r8=0x1000

r9=0xff8

r10=5

...

memory



I-Cache

Branch Predictor

L2 cache

...

D-Cache

addr	data
0x1000	5
0xff8	4

What are side channels?

- Side channels are not designed for communicating data
- Reveal data through ancillary properties of processing

D-Cache

addr	data
0x1000	5
0xff8	4

Set-associative cache (example)

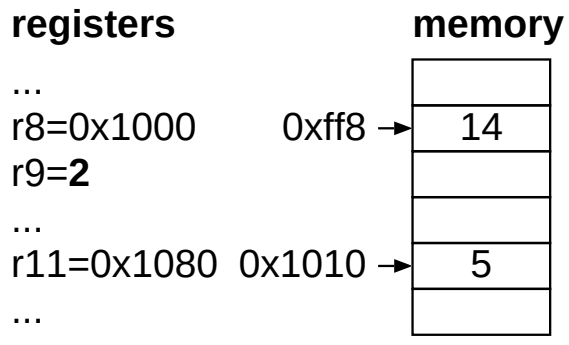
- 64 sets for D-cache
- Set n for addresses $4096m + 64n + (0...63)$
- access replaces a cache line
- timing reveals 6 bits of accessed address

Defense against classical timing attacks

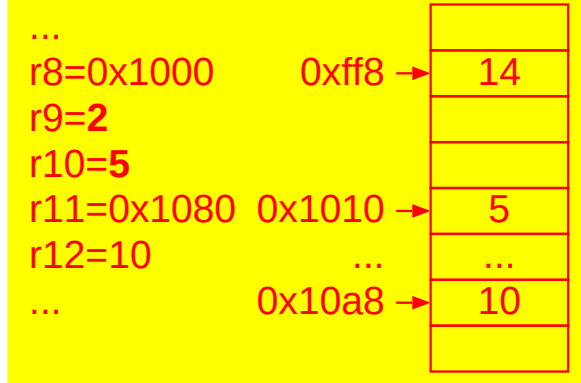
- Write constant-time code
- for cryptographic and password-handling code
- Not practical for most other code
  but then other code does not access passwords or keys
- Requires knowledge about instruction characteristics

What is speculative execution?

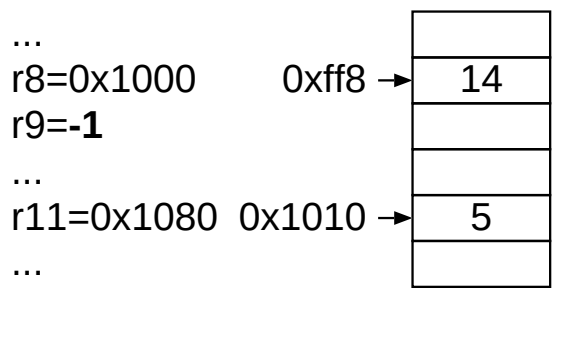
```
# r8=0x1000 r9=2 r11=0x1080
# m[0x1010]=5
# m[0x10a8]=10
cmp     15,%r9
ja      outofbounds
mov     (%r8,%r9,8),%r10
mov     (%r11,%r10,8),%r12
```



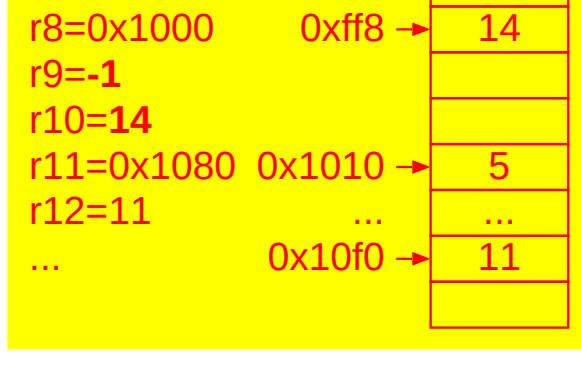
speculative architectural state




```
# r8=0x1000 r9=-1 r11=0x1080
# m[0xff8]=14
# m[0x10f0]=11
cmp     15,%r9
ja      outofbounds
mov     (%r8,%r9,8),%r10
mov     (%r11,%r10,8),%r12
```

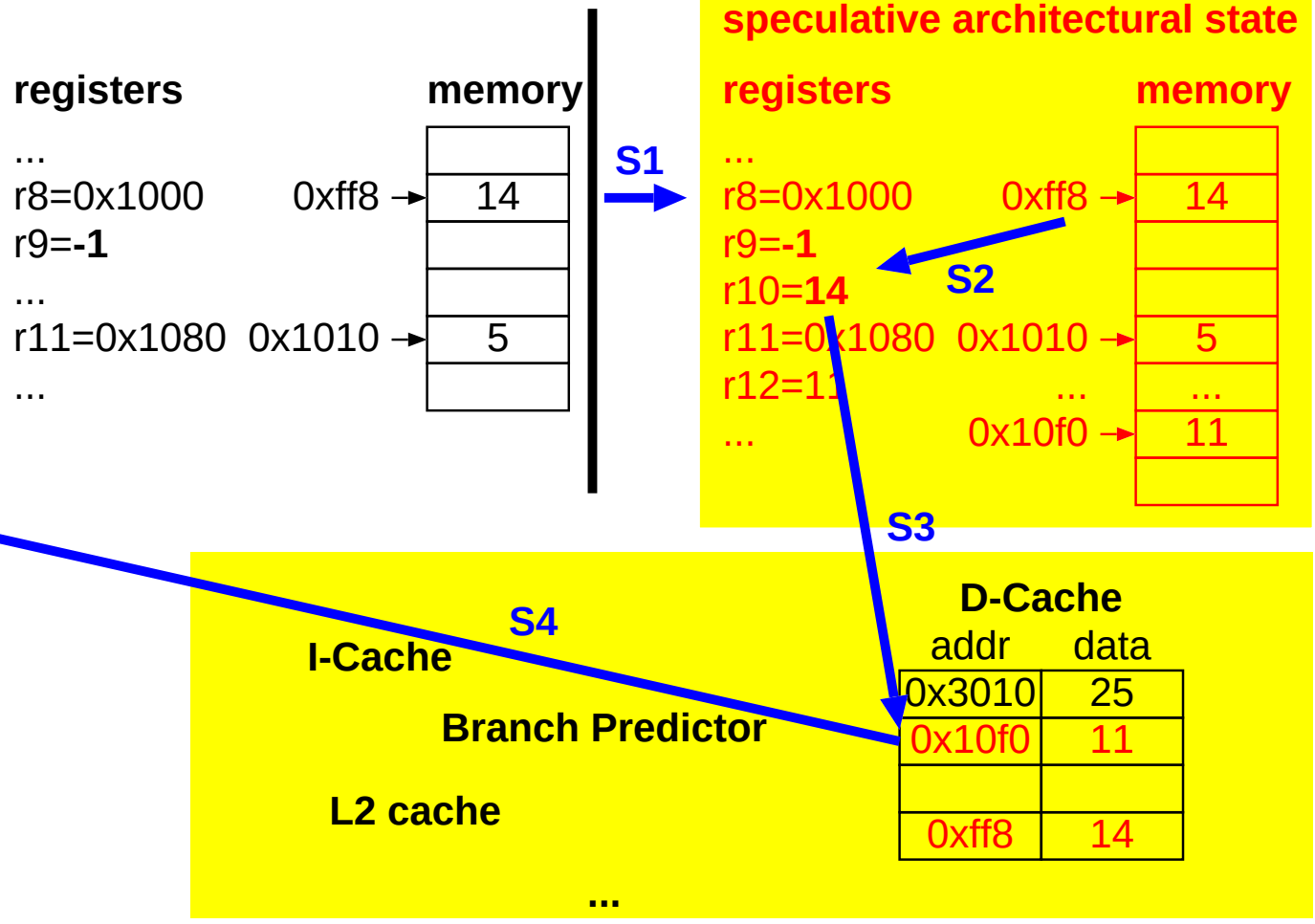


speculative architectural state



What is Spectre?

```
# r8=0x1000 r9=-1 r11=0x1080
#m[0xff8]=14
# m[0x10f0]=11
cmp     15,%r9
ja      outofbounds
mov     (%r8,%r9,8),%r10
mov     (%r11,%r10,8),%r12
```



How relevant is Spectre?

- No known attack in the wild
  But then, how would you know?
- Working exploit (reading unaccessible files in Linux) is available
- What about the large number of software vulnerabilities?
  These usually get fixed soon
- Spectre has not been fixed in > 6 years
  A hardware fix will not affect existing processors
  Software mitigations may prevent attacks. Or not

What about software mitigations for Spectre?

Speculative Load Hardening (simplified)

```
      cmp     15,%r9
      ja      end
      mov     $0x0,%rax
      cmova   %rax, %r9
      mov     (%r8,%r9,8),%r10
      mov     (%r11,%r10,8),%r12
end:
```

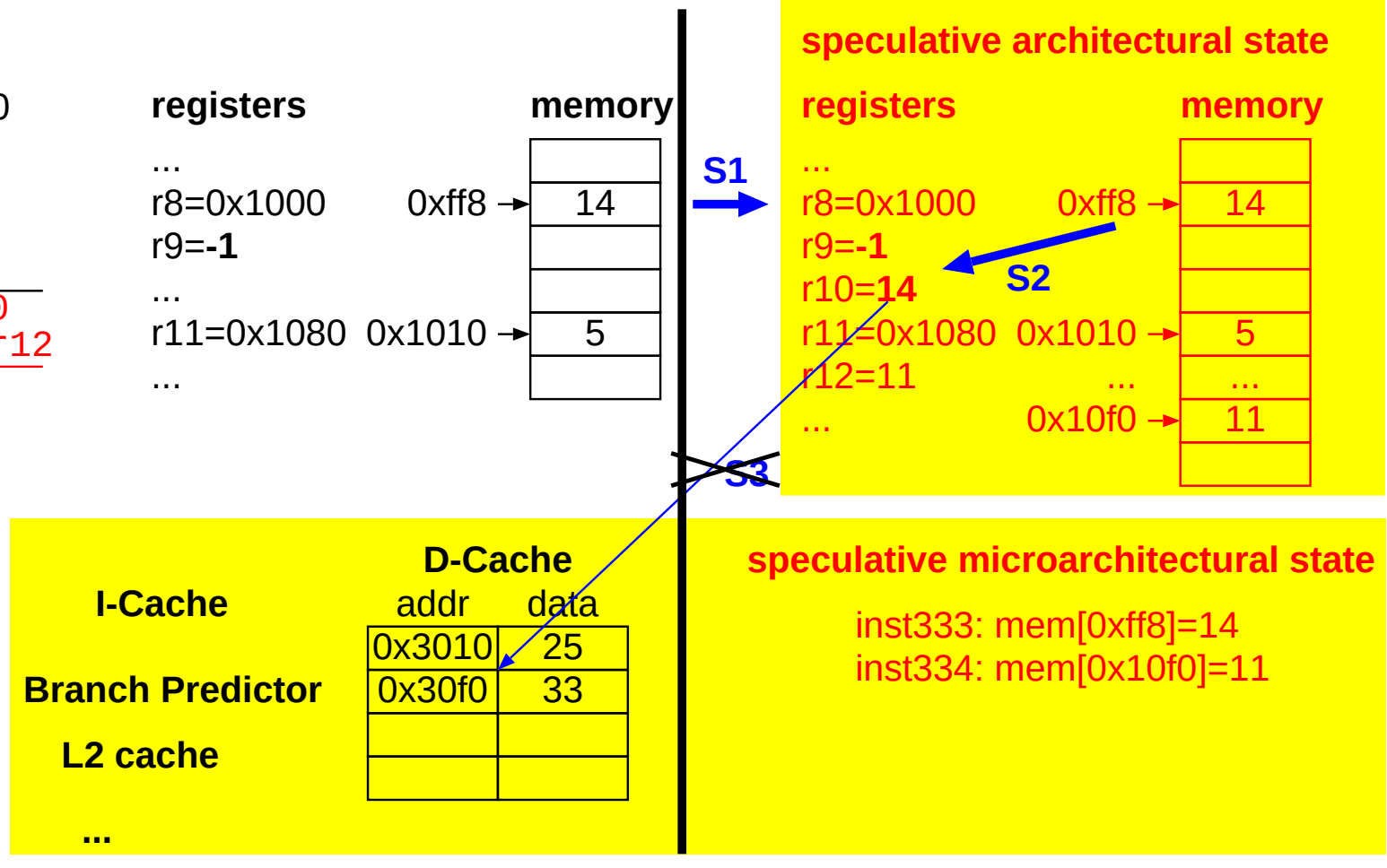
- SLH fixes only Spectre v1
- Slowdown 2.5× (for Ultimate SLH)
- Slowdown 2×–9.5× of Gforth from retpolines (Spectre v2)
- Selective application?
  Lots of effort
  Error prone
  Repeat effort on next Spectre variant

How about disabling speculation?

- Eliminates S1
- Very slow
  A55 on Rock5b (no speculation) 3.3× slower than A76
  ... 7.8× slower than Firestorm (Apple M1)

How to fix Spectre in hardware?

```
# r8=0x1000 r9=-1 r11=0x1080
#m[0xff8]=14
# m[0x10f0]=11
cmp     15,%r9
ja      outofbounds
mov     (%r8,%r9,8),%r10
mov     (%r11,%r10,8),%r12
```



How to fix Spectre in hardware? (cont.)

Resource contention as side channel

- execution ports, functional units, cache ports, ...
- fixed partitioning between SMT threads
  time-division multiplexing (fixed slots) for unique resources
- Within thread:
  older instructions have priority
  front-end resources independent of speculation

Power side channel

- Meltdown-Power prevented by fix
- Other speculative attacks with power side channels?
  Imaginable, but what is the bandwidth?

How much does the fix cost?

- Design complexity
- Some chip area, but not huge
  E.g., maybe 30 cache lines compared to 224 physical ZMM registers
- Performance for MuonTrap (cache-only part of fix)
  1.05× speedup for Parsec
  1.04× slowdown for SPEC CPU 2006
  compared to vulnerable hardware with no software mitigations applied

What should I do?

- **Computer customer**
  Ask CPU manufacturers when they will fix Spectre in hardware
  When one of them does, buy from them
- **Researcher**
  Design and evaluate efficient ways of fixing Spectre
  Work on proving that a fix closes the vulnerability
- **CPU manufacturer**
  Go ahead and fix Spectre in hardware!
  Competetive advantage for the first mover
  Avoid the constant stream of new Spectre variants

Conclusion

- Spectre:
  S1: misspeculation
  S2: access secret
  S3: send secret through side channel
  S4: receive secret from side channel
- Software mitigations cause big slowdown
  Selective application usually impractical
- Solution: Fix it in hardware!
  Keep speculative microarchitectural state separate from commited state
  Also eliminate resource contention side channel from speculation
- Cost: Complexity, some chip area, some performance

Paper: http://www.euroforth.org/ef23/papers/ertl.pdf