



Computer Organization and Architecture

Branch Prediction

Jeongseob Ahn

Department of Software & Computer Engineering
Ajou University



Branch Prediction

- Longer pipelines can't readily determine branch outcome early
 - Stall penalty becomes unacceptable
- Predict outcome of branch
 - Only stall if prediction is wrong
- In MIPS pipeline
 - Can *statically* predict branches **not taken**
 - Fetch instruction after branch, with no delay

Security Issue

The screenshot shows a web browser with two tabs. The background tab is titled "Meltdown and Spectre" and shows a page with the heading "Meltdown and Spectre" and "Vulnerabilities in". The foreground tab is titled "Spectre (security vulnerability)" and shows the Wikipedia article for "Spectre (security vulnerability)".

The Wikipedia article includes the following text:

Spectre is a **vulnerability** that affects modern **microprocessors** that perform **branch prediction**.^{[1][2][3]} On most processors, the **speculative execution** resulting from a **branch misprediction** may leave observable side effects that may reveal private data to attackers. For example, if the pattern of memory accesses performed by such speculative execution depends on private data, the resulting state of the data cache constitutes a **side channel** through which an attacker may be able to extract information about the private data using a **timing attack**.^{[4][5][6]}

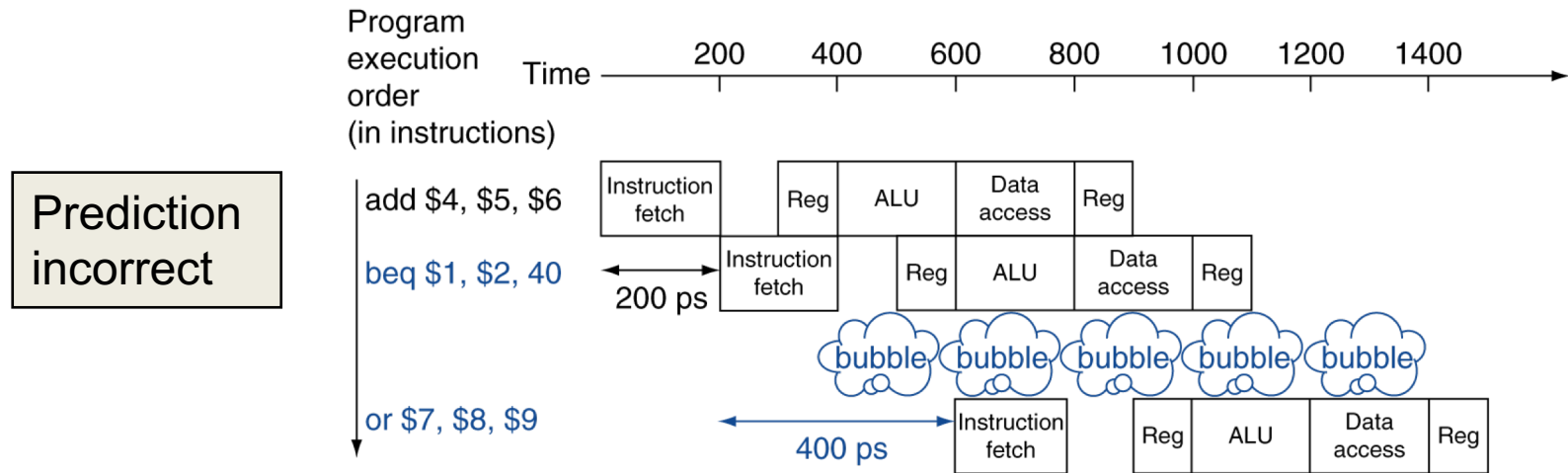
Two **Common Vulnerabilities and Exposures** IDs related to Spectre, **CVE-2017-5753**^[7] (bounds check bypass, Spectre-V1, Spectre 1.0) and **CVE-2017-5715**^[8] (branch target injection, Spectre-V2), have been issued.^[7] **JIT engines** used for **JavaScript** were found to be vulnerable. A website can read data stored in the browser for another website, or the browser's memory itself.^[8]

On March 15, 2018, **Intel** reported that it will redesign its **CPUs** (performance losses to be determined) to help protect against the Spectre and related **Meltdown** vulnerabilities (especially, Spectre variant 2 and Meltdown, but not Spectre variant 1), and expects to release the newly redesigned processors later in 2018.^{[9][10][11][12]} On October 8, 2018, Intel is reported to have added hardware and firmware mitigations regarding Spectre and Meltdown vulnerabilities to its latest processors.^[13] On October 18, 2018, MIT researchers suggested a new mitigation approach, called **DAWG** (Dynamically Allocated Way Guard), which may promise better security without compromising performance.^[14]

The article also features a sidebar with navigation links, a "Wikipedia Asian Month" banner, and a "SPECTRE" logo featuring a ghost with a branch.

https://meta.wikimedia.org/wiki/Wikipedia_Asian_Month_2018

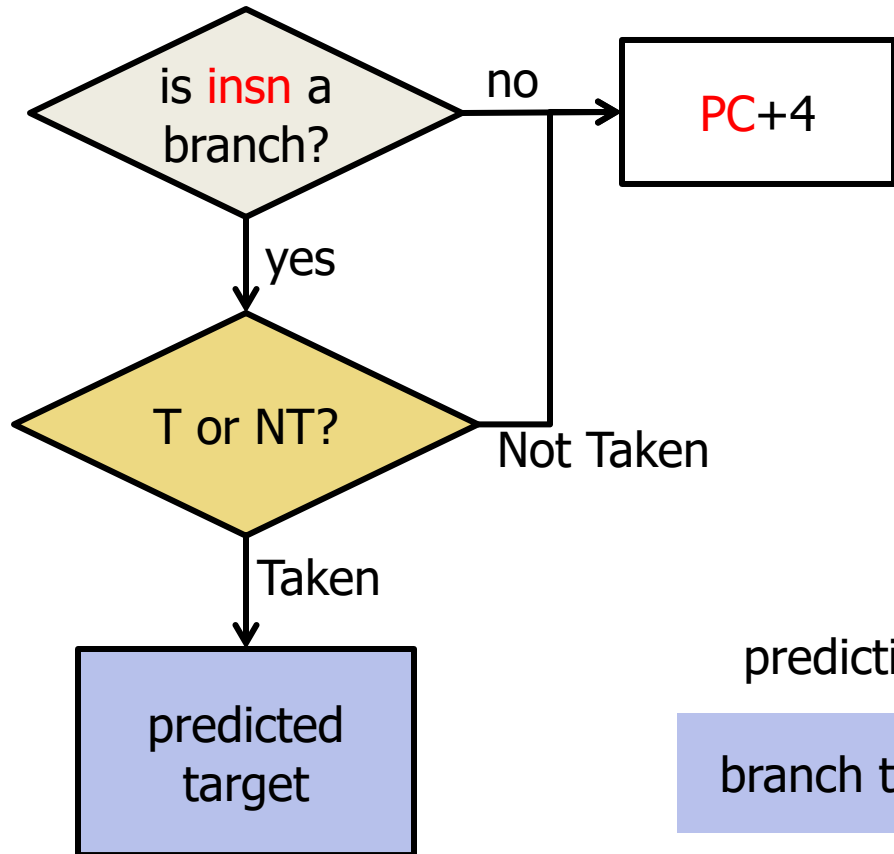
MIPS with Predict Not Taken



Dynamic Branch Prediction

- Hardware measures actual branch behavior
 - Record recent history of each branch
- Assume future behavior will continue the trend
 - When wrong, stall while re-fetching, and update history

Branch Prediction Steps



- Which instructions behavior are we trying to predict?
- Where does **PC** come from?

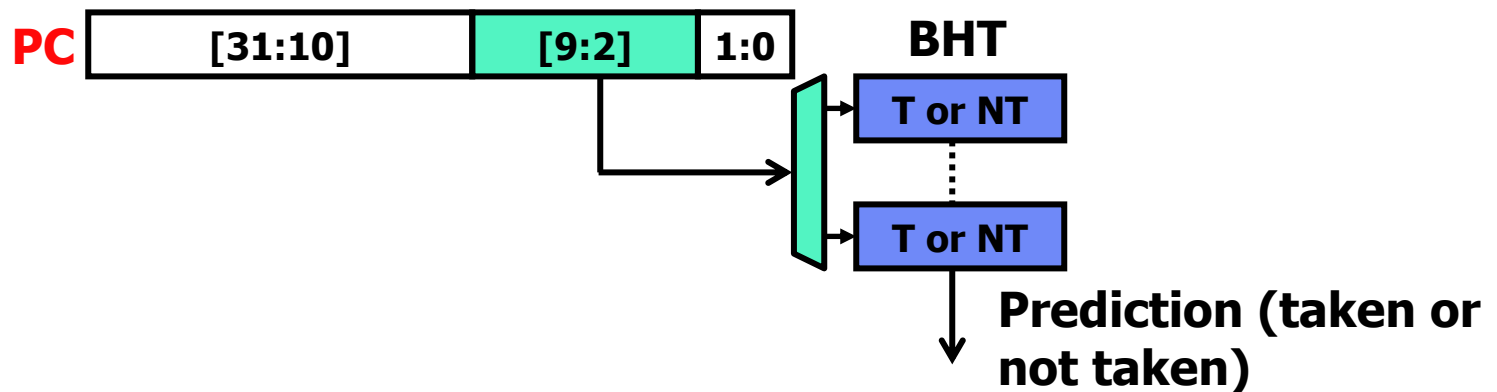
prediction source:

branch target buffer

direction predictor

Branch Direction Prediction

- **Direction predictor (DIRP)**
 - Map conditional-branch PC to taken/not-taken (T/N) decision
- **Branch history table (BHT):** simplest predictor
 - PC indexes table of bits (0 = N, 1 = T), no tags
 - Essentially: branch will go same way it went last time



Branch History Table (BHT)

- **Branch history table (BHT):**
simplest direction predictor
 - PC indexes table of bits (0 = N, 1 = T), no tags
 - Essentially: branch will go same way it went last time
 - Problem: **inner loop branch** below

```
for (i=0;i<100;i++)
    for (j=0;j<3;j++)
        // loop body
```

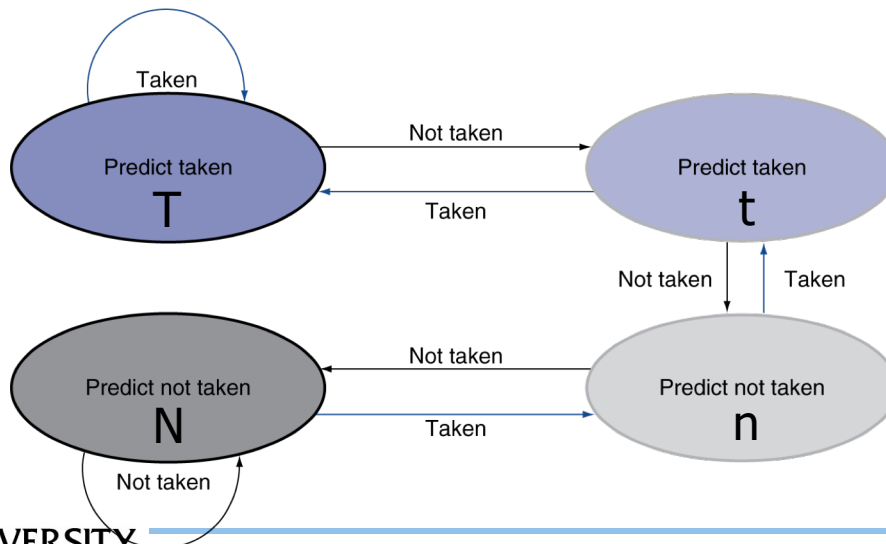
 - Two “built-in” mis-predictions per inner loop iteration
 - Branch predictor “changes its mind too quickly”

Time	State	Prediction	Outcome	Result?
1	N	N	T	Wrong
2	T	T	T	Correct
3	T	T	T	Correct
4	T	T	N	Wrong
5	N	N	T	Wrong
6	T	T	T	Correct
7	T	T	T	Correct
8	T	T	N	Wrong
9	N	N	T	Wrong
10	T	T	T	Correct
11	T	T	T	Correct
12	T	T	N	Wrong

Two-Bit Saturating Counters (2bc)

- Two-bit saturating counters**

- Replace each single-bit prediction
 - $(0,1,2,3) = (N,n,t,T)$
- Adds “hysteresis”
 - Force predictor to mis-predict twice before “changing its mind”
- One misprediction each loop execution (rather than two)



Time	State	Prediction	Outcome	Result?
1	N	N	T	Wrong
2	n	N	T	Wrong
3	t	T	T	Correct
4	T	T	N	Wrong
5	t	T	T	Correct
6	T	T	T	Correct
7	T	T	T	Correct
8	T	T	N	Wrong
9	t	T	T	Correct
10	T	T	T	Correct
11	T	T	T	Correct
12	T	T	N	Wrong

Branch Target Address Prediction

- **Branch target buffer (BTB):**

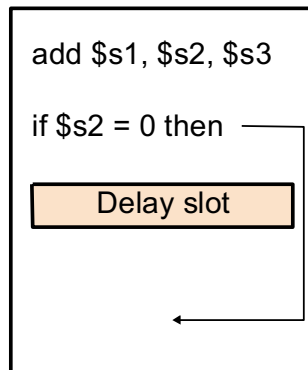
- A small hardware table
- is-a-branch = (BTB[hash(PC)].tag == PC) ? 1 : 0
- predicted-target = (BTB[hash(PC)].tag == PC) ? BTB[PC].target : 0

index	tag	target
0	0	0
1	0x4e3745	0x4738da
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	0	0

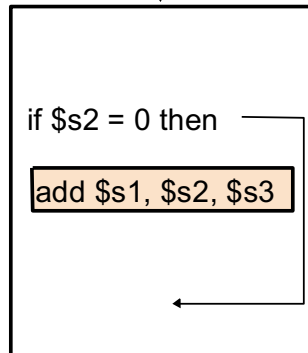
Compiler Approach: Delayed Branching

- Reordering data independent instructions does not change program semantics

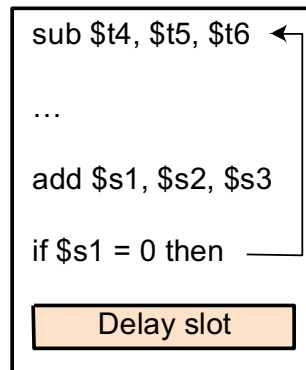
a. From before



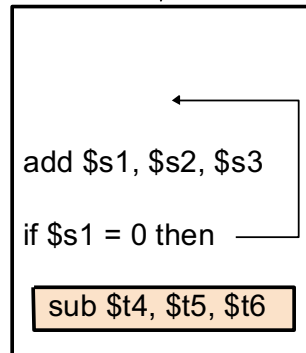
Becomes



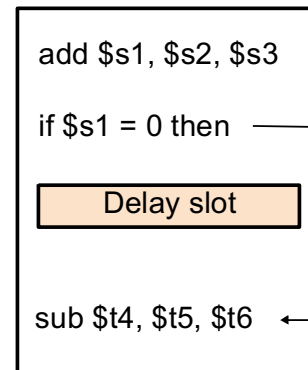
b. From target



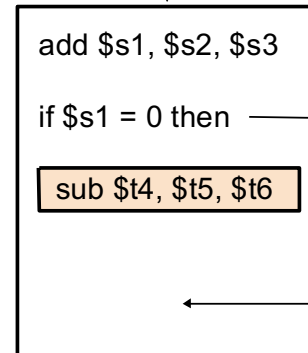
Becomes



c. From fall through



Becomes



Precise Exception with Pipeline

- Traps (invoked by instruction)
 - Squash all subsequent instructions
 - Set PC to exception handling routine
- External interrupts
 - Processor states updated in MEM and WB
 - Must finish the instructions in the middle of MEM and WB

Advanced Topics

- Superscalar
 - Fetching two instructions at the same time (2-way Superscalar)
- Out-of-order Execution
 - Processors dynamically change the order of instruction to maximize ILPs (Instruction Level Parallelism)
- Hardware Multithreading (a.k.a Intel's Hyper Threading)
 - Sharing the hardware resources except for PC and Register File to maximize TLPs (Thread Level Parallelism)
 - Processor is responsible for scheduling the hardware threads
 - It is not software multithreading technique