University of BRISTOL

Systems & Software Security

COMSM0050

2020/2021

bristol.ac.uk

University of BRISTOL

Speculative execution

bristol.ac.uk

How the CPU work

- FETCH
- DECODE
- EXECUTE
- WRITE BACK

How the CPU work

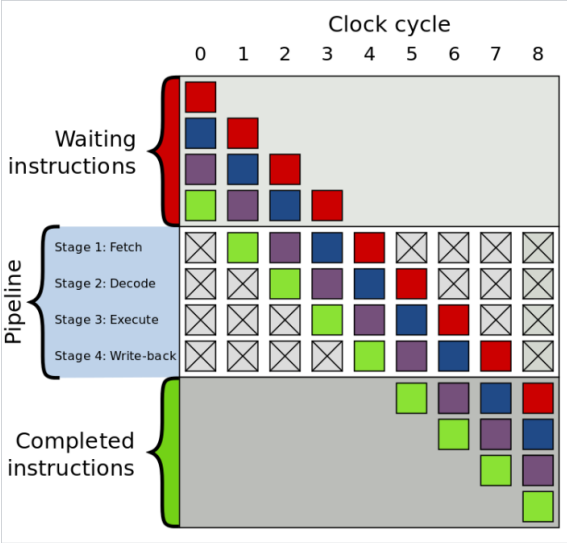

Figure 1: Example of 4-stage pipeline. The colored boxes represent instructions independent of each other

- FETCH
- DECODE
- EXECUTE
- WRITE BACK

What happened on branch?

- You evaluate the branch condition…
- … but you need to wait to know what will be the next instruction
- Your pipeline remains empty
- That create a “bubble”

How can we avoid bubble?

- We need branches…
- We can guess what is likely to be executed next
  - branch predictor
- If wrong roll back

How can we avoid bubble?

- We need branches…
- We can guess what is likely to be executed next
  - branch predictor
- If wrong roll back

- If we are not too bad at guessing we should be able to improve performance.

Static predictor

- Simplest approach
- Predict always taken
- Predict always not taken
- Backward branch predict taken; forward branch predict not taken

Can also be informed by the compiler (through a bit in the opcode)

Dynamic branch prediction

- Prediction based on the execution history of a program
  - e.g. branch in a loop more likely to execute the loop code
- During startup information is gathered
  - Can use static prediction
- Once information has been gathered dynamic branch prediction become effective
- Better Performance
- More hardware complexity

Implementing dynamic prediction (simple)

- Table associating branching address to prediction
- 1 bit
  - 0 branch was not taken
  - 1 branch was taken
- If 0 predict not taken
- If 1 predict taken

Implementing dynamic prediction (simple)

- Table associating branching address to prediction
- 1 bit
  - 0 branch was not taken
  - 1 branch was taken
- If 0 predict not taken
- If 1 predict taken

In other word, predict the same branch will be taken as the branching was seen

Implementing dynamic prediction (simple)

- Table associating branching address to prediction
- 1 bit
  - 0 branch was not taken
  - 1 branch was taken
- If 0 predict not taken
- If 1 predict taken
- Let's take a loop with 9 iterations
  - It will be correct in 80% of the time
  - First time predict exit instead of looping
  - Last iteration would predict continue instead of exit

More complex predictor

- This is the simplest predictor
- Obviously much more complex now
  - … but beyond the scope of this course
- Take an architecture unit to know more!