In [computer science](https://en.wikipedia.org/wiki/Computer_science), **instruction pipelining** is a technique for implementing [instruction-level parallelism](https://en.wikipedia.org/wiki/Instruction-level_parallelism) within a single processor. Pipelining attempts to keep every part of the processor busy with some instruction by dividing incoming [instructions](https://en.wikipedia.org/wiki/Machine_code) into a series of sequential steps (the eponymous "[pipeline](https://en.wikipedia.org/wiki/Pipeline_(computing))") performed by different [processor units](https://en.wikipedia.org/wiki/Central_processing_unit#Structure_and_implementation) with different parts of instructions processed in parallel.

a **branch target predictor** is the part of a processor that predicts the target of a taken [conditional branch](https://en.wikipedia.org/wiki/Conditional_branch) or an unconditional branch instruction before the [target of the branch instruction](https://en.wikipedia.org/wiki/Jump_target_(computing)) is computed by the execution unit of the processor

**branch predictor**[[1]](https://en.wikipedia.org/wiki/Branch_predictor#cite_note-dbp-class-report-1)[[2]](https://en.wikipedia.org/wiki/Branch_predictor#cite_note-schemes-and-performances-2)[[3]](https://en.wikipedia.org/wiki/Branch_predictor#cite_note-3)[[4]](https://en.wikipedia.org/wiki/Branch_predictor#cite_note-4)[[5]](https://en.wikipedia.org/wiki/Branch_predictor#cite_note-5) is a [digital circuit](https://en.wikipedia.org/wiki/Digital_electronics) that tries to guess which way a [branch](https://en.wikipedia.org/wiki/Branch_(computer_science)) (e.g., an [if–then–else structure](https://en.wikipedia.org/wiki/Conditional_(programming))) will go before this is known definitively. The purpose of the branch predictor is to improve the flow in the [instruction pipeline](https://en.wikipedia.org/wiki/Instruction_pipeline).

Without branch prediction, the processor would have to wait until the conditional jump instruction has passed the execute stage before the next instruction can enter the fetch stage in the pipeline. The branch predictor attempts to avoid this waste of time by trying to guess whether the conditional jump is most likely to be taken or not taken. The branch that is guessed to be the most likely is then fetched and [speculatively executed](https://en.wikipedia.org/wiki/Speculative_execution). If it is later detected that the guess was wrong, then the speculatively executed or partially executed instructions are discarded and the pipeline starts over with the correct branch, incurring a delay.

The time that is wasted in case of a **branch misprediction** is equal to the number of stages in the pipeline from the fetch stage to the execute stage.