LSU EE 4720 **Homework 7** **Due: 27 April 2026**

Solution *Formatted 19:16, 1 May 2026*

Late submissions will be accepted without penalty until the morning of 30 April 2026.

Collaboration Rules

Each student is expected to complete his or her own assignment. It is okay to work with other students and to ask questions in order to get ideas on how to solve the problems or how to overcome some obstacle (be it a question of MIPS, RISC-V, or assembler syntax, interpreting error messages, how a part of the problem might be solved, etc.) It is also acceptable to seek out resources for help on MIPS, RISC-V, etc. It is okay to make use of AI LLM tools such as ChatGPT, Claude, and Copilot to generate sample code. (Do not assume LLM output is correct. Treat LLM output the same way one might treat legal advice given by a lawyer character in a movie: it may sound impressive, but it can range from sage advice to utter nonsense.)

After availing oneself to these resources **each student is expected to be able to complete the assignment alone**. Test questions will be based on homework questions and **the assumed time needed to complete the question will be for a student who had solved the homework assignment on which it was based**.

Student Expectations

To solve this assignment you are expected to avail yourself of references provided in class and on the Web site, and to learn how to handle references that are at first hard to understand, and to keep looking (and asking) when the answer isn't in the first place you look. Some of the problems require thought, and you are expected to persevere until you find a solution. It is each student's duty to him or herself to resolve frustrations and roadblocks, helped along by the satisfaction of making progress. There are plenty of old problems and solutions to look at. One way to resolve issues is to ask Dr. Koppelman or others for help.

Resources

Many past final exams and homework assignments have problems on branch prediction.

Problem 1: Solve 2025 Final Exam Problem 3a, which asks for the prediction accuracy of bimodal and local predictors on two branches.

See the posted solution, at `https://www.ece.lsu.edu/ee4720/2025/fe_sol.pdf`.

Problem 2: Consider the predictor from Problem 3b of the 2025 Final Exam. Assume that the size of the target field is 16 bits and that $h = 8$.

☑ Determine the size of the BHT in bits. ☑ Determine the size of the PHT in bits.

The BHT is indexed by $16 - 2 = 14$ bits and so there are 2^{14} entries. Each BHT entry consists of a 1-bit `Is Branch` field (size assumed), a 16-bit `Target` field (size given), and an $h = 8$-bit local history field (also given) for a total of $1 + 16 + 8 = 25$ bits. The total size of the BHT is then $25 \times 2^{14}\,\mathrm{b} = 409600\,\mathrm{b}$.

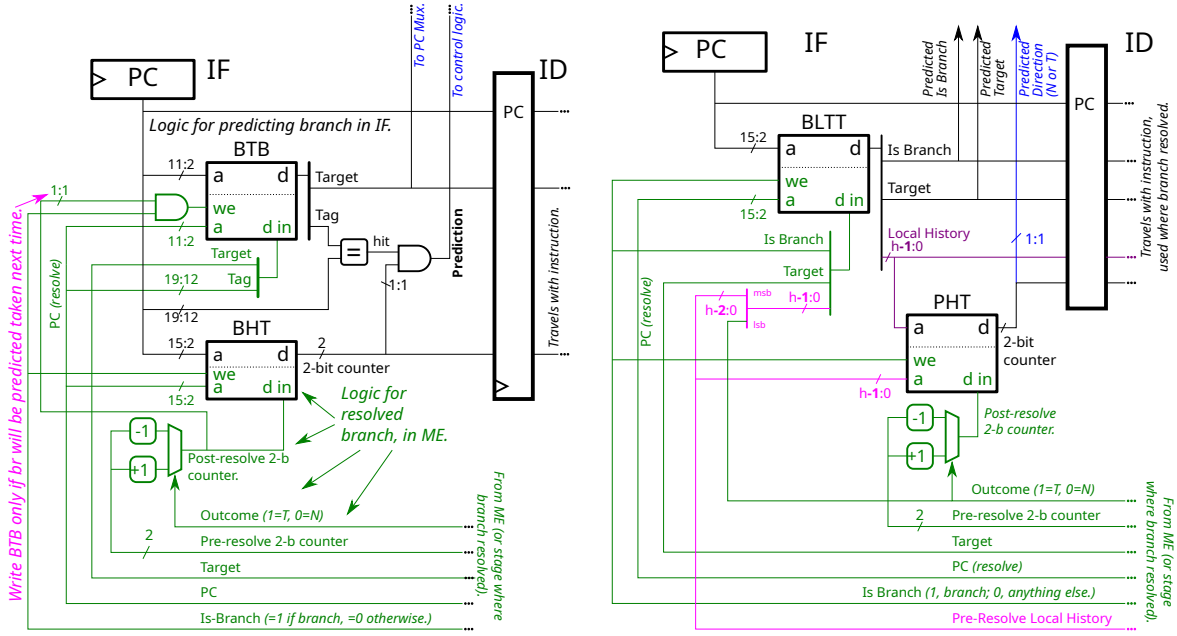
Since the local history size is 8 bits there are $2^8 = 256$ entries. Each is two bits, for a total size of 512 bits.

Problem 3: Solve Problem 3b of the 2025 Final Exam. *Note: In the original assignment I suggested that one look at the tagged bimodal predictor. That was unintentionally misleading because the use of a tag would not solve the final exam problem.*

☑ Solve Problem 3b of the 2025 Final Exam.

See the posted solution, at `https://www.ece.lsu.edu/ee4720/2025/fe_sol.pdf`.

Problem 4: The BTB/BHT bimodal predictor from the notes (and on the lower left) avoids wasting a BTB entry on a branch by using an oversized BHT. Below on the right is the local predictor used in the 2025 Final Exam and the class slides, but with the BHT (branch history table) renamed BLTT (branch local history and target table). (The table was renamed to avoid confusion with the bimodal predictor's BHT.) Larger versions of these appear on the following pages, and their SVG sources can be found at `https://www.ece.lsu.edu/ee4720/2026/lsli12-bimodal-btb-tag.svg` and `https://www.ece.lsu.edu/ee4720/2026/hw07-local-pred.svg`, and a collection of gates can be found at `https://www.ece.lsu.edu/ee4720/2026/c.svg`.



The local predictor BLTT entries are even larger than the BTB in a bimodal predictor, so we don't want to waste the BLTT entries either. Modify the local predictor so that its BLTT entries are not wasted on branches that are mostly not taken. Do so using a third table, something like the BHT in a bimodal predictor.

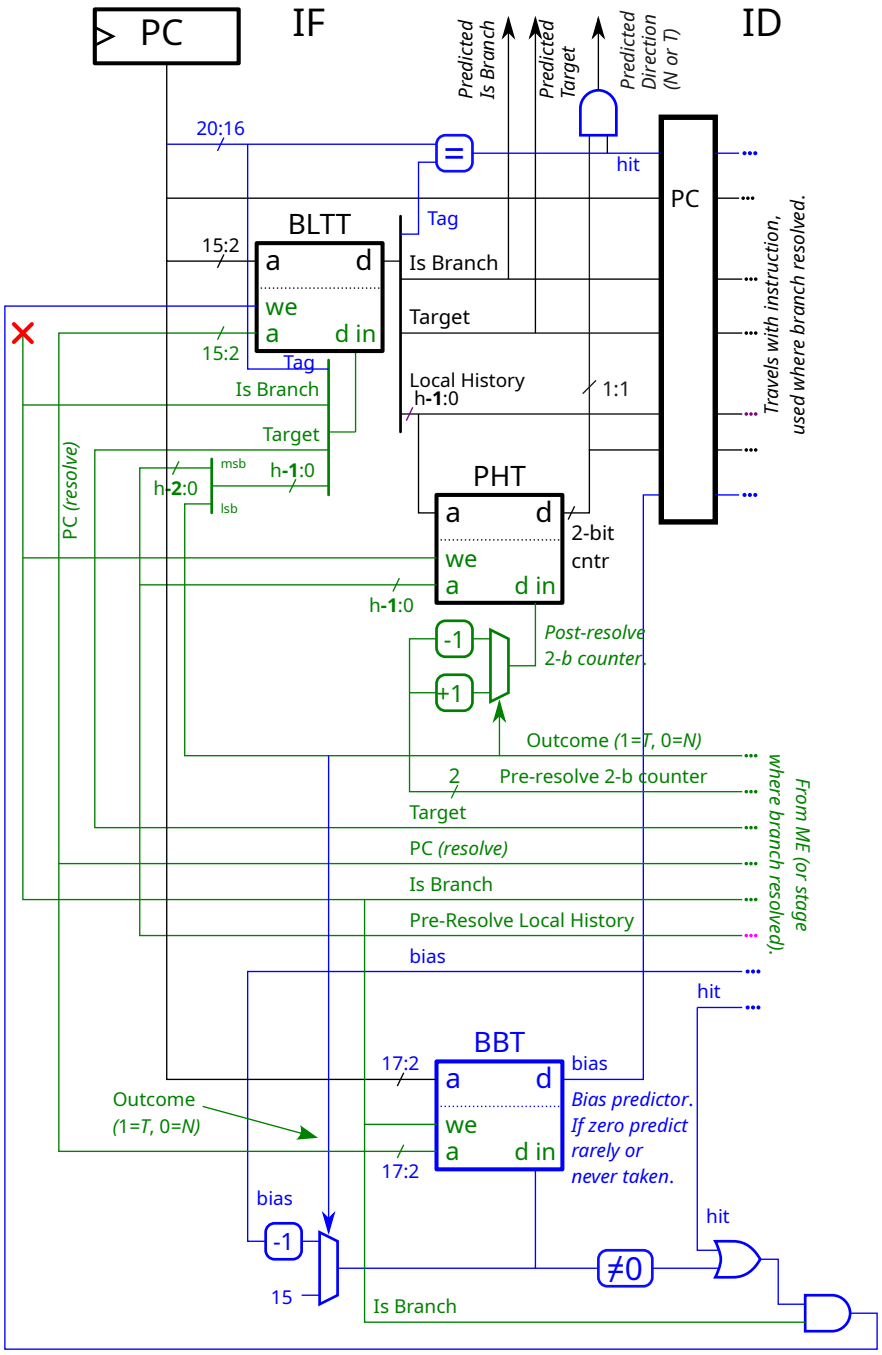
☑ Modify the local predictor to avoid wasting BLTT entries on mostly not-taken branches.par

☑ The local predictor should still be able to predict sequences like `N N N T T N N N T T ...`

☑ Note that it's important to keep the local history in the BLTT consistently updated.

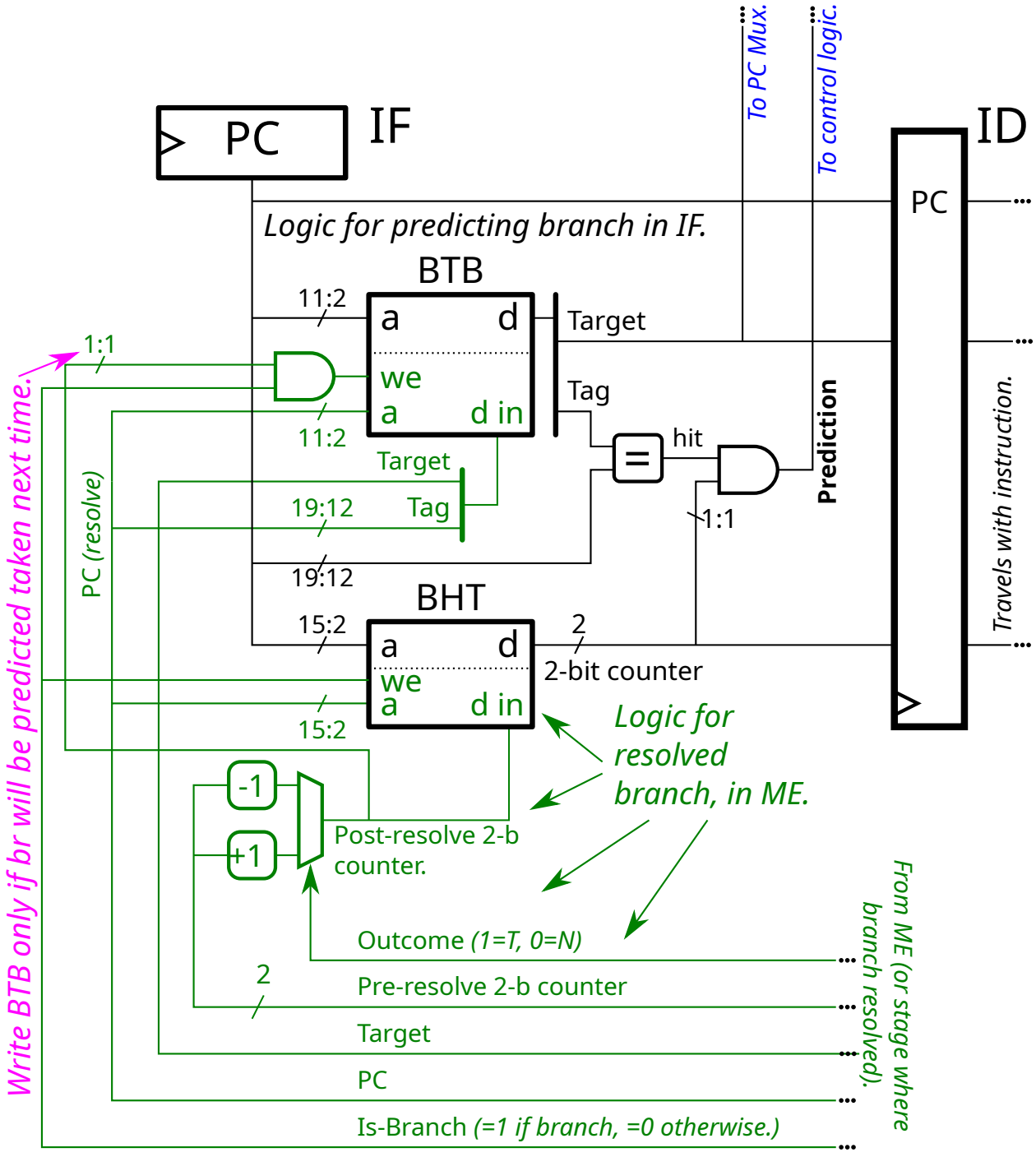
Solution appears below (or maybe the next page). A tag was added to the BLTT so that one could determine whether the branch being predicted matches the BLTT entry. Here the tag is five bits, which is a wild guess. A larger tag would have fewer false-positive hits (meaning `hit` is `1` when actually the branches are different), but of course requires more memory.

Also added was a BBT (branch bias table) which is used to detect branches that are mostly not taken. (Here mostly means almost never taken.) A BBT entry is set to 15 if a branch is taken and decremented if it's not taken. A branch needs 15 consecutive not-taken outcomes to reach zero. The BBT is indexed using two more bits than the BLTT so that the BBT can monitor four branches for each BBLT entry. The number of extra bits is also a wild guess. It must be greater than zero so that there is more than one BBT entry per BBLT entry. A BBLT entry is updated during ME (or when the branch resolves) if the entry hit (when it was in IF) or if the entry missed (the tag did not match) *and* the branch that did resolve is not highly biased not-taken. This way, branches that are highly biased not taken, and so don't need the BBLT, will not overwrite a BBLT entry.



Tagged BTB/BHT Bimodal Predictor

SVG source at `https://www.ece.lsu.edu/ee4720/2026/lsli12-bimodal-btb-tag.svg`.



Local Predictor:

SVG source at `https://www.ece.lsu.edu/ee4720/2026/hw07-local-pred.svg`.