Q3

Implement a Gshare Predictor: In this part, you are required to implement Gshare branch predictor. There are several branch predictors implemented in gem5, including 2-bit local predictor, bi-modal predictor, and tournament predictor. However, gem5 does not expose any branch prediction configuration in the command-line interface. src/cpu/o3/O3CPU.py is the file where the particular branch prediction is selected, whereas src/cpu/pred/BranchPredictor.py specifies the configuration of each predictor. All the branch prediction implementation can be found in src/cpu/pred as well. The branch predictor in gem5 inherit the base class, BPredUnit, from src/cpu/pred/bpred unit.hh. The different behaviors of predictors are distinguished in the 5 virtual functions - lookup, update, uncondBranch, btbUpdate, and squash. To implement a new branch predictor, you only have to fill up these 5 function with your own design. Here are the descriptions and notes of each virtual function.

• lookup: This function returns whether a branch with a given PC is predicted as taken or not taken. The counter data of branch history (Like the counter to store the global history and the counter for local history for the specific PC) are also backup in this stage, and can be used to restore those counter if necessary.

• update: Update those counter value of history in any mean for branch prediction by a given outcome of a branch instruction. Note that the function argument, squashed, is to indicate whether this is a misprediction or not.

• uncondBranch: This function is called when the target branch instruction is an unconditional branch. Typically, a branch predictor will do nothing but update global history with taken, as a unconditional branch is always taken.

• btbUpdate: This function is called only if there is a miss in Branch Target Buffer (BTB). It means the branch prediction does not know where to jump even though the predictor can accurately predict the branch outcome. In this case, you will predict a branch as not taken so that the target branch address is not required. Thus, this function is typically to correct what you did in the lookup function into a result that you predict a branch as non-taken.

• squash: If there is a branch misprediction, everything changed (counters, registers, cache, memory, etc) due to the outstanding instructions after this branch need to be reverted back into the state before the branch instruction was issued. This includes those victim branch instructions following the branch misprediction. This function is primarily called to erase what the outstanding branches did to those history counters by restoring them with the backup value you stored
in update function.

The following table contains the parameters you need to set for the three branch predictors. Run using the se.py script provided in gem5. You only enable the following flags and leave all others as the default --cpu-type=DerivO3CPU --caches --l2cache --bpred= TournamentBP(), LocalBP() and GshareBP().

2-bit local	Tournament	Gshare
16384 entries * 2 bits	Local History table - 1024 * 11 bits Local predictor size - 2048 * 2 bits One 12-bit Global history register Global predictor size - 4096 * 2 bits Choice predictor size - 4096 * 2 bits	16384 entries * 2 bit one 16-bit Global history register

Solution

The majority of the inspiration for GShare Predictor was obtained through the tournament predictor as it has both the GShare as well as PShare predictors and through other online resources.

Steps

1. Write the gshare.c and gshare.h files using the other implemented predictors code for tournament predictor comes very handy.
2. Now we need to instantiate the predictor in the python script which is named as BranchPredictor.py, depending on the parameters we require for our class.
3. Further, we need to make sure the Sconscript is also edit such that the parser generates the corresponding related files during the build process for this just edit the Sconscript file in the same directory.

The parameters according to the table above can be populated using the BranchPredictor.py. Once you are done with all the initialisation build again using the following command.

```
scons build/X86/gem5.opt -j5
```