CS203 Winter '16 - Project 2

Due: Thursday, February 4 before class.
All projects will be turned in to iLearn.

1. Objective

The goal of project 2 is to explore pipeline design choices and branch predictors.

2. Exploring Pipeline Design

Wide Pipelines

a) Make a backup of your processor configuration file
`cp simu.conf.apache simu.conf.apache.backup`
b) Edit `simu.conf.apache`
Change `issueWidth` and `retireWidth` to `1`
This configuration represents a pipeline that only issues a single instruction
c) What is the IPC for the four workloads (crafty, sjeng, dhry, blackscholes)?
d) Now change the `issueWidth` and `retireWidth` to 2. This configuration represents a pipeline that can issue two instructions per cycle.
e) What is the IPC? How is a wider pipeline able to improve IPC?
f) Now change the `issueWidth` and `retireWidth` to 4.
g) What is the IPC? Are there still significant IPC improvements? Why or why not?

Deep Pipelines

In a 5-stage pipeline design, the frequency is 1GHz with a pipeline latch delay of 500ps. As a computer architect, your goal is to increase the pipeline depth in order to increase the processor frequency.

a) Plot pipeline stages vs frequency
b) What is the maximum frequency achievable?

Let's now take into account the area overhead of a latch. Assume that our 5-stage pipeline design takes up 64mm^2, with each latch taking up 4mm^2.

c) What is the optimum number of pipeline stages (round up)? What processor frequency would this design run at?

3. Exploring Branch Predictors

In this section, we will explore the effect of various basic branch prediction policies.

a) Reset your `simu.conf.apache` configuration file.
`cp simu.conf.apache.backup simu.conf.apache`

b) Edit simu.conf.apache.
Comment out `bpred2 = 'BPredIssueX2'` as we'll use a simple single level predictor

c) You can specify the branch predictor type by editing the following line:
```
[BPredIssueX]
type                    = "2bit" #2bit, NotTaken, Taken
```

d) The branch prediction rate is given by the following highlighted in blue:

```
*****************************************************************************************************
# File  : esesc_noname.M4fWnD  : Mon Jan 11 02:19:05 2016
*****************************************************************************************************
Sampler 0 (Procs  0 1)
        Rabbit     Warmup    Detail    Timing    Total  KIPS
  KIPS   44359       1760       757       765      3794
  Time     5.0%      83.3%      1.6%     10.1%              : Sim Time (s) 131.789 Exe   4.739 ms Sim (1700MHz)
  Inst    59.0%      38.7%      0.3%      2.0%            : Approx Total Time 288.533 ms Sim (1700MHz)
*****************************************************************************************************
Proc : Avg.Time :  BPType          :  Total    :         RAS        :  BPred  :         BTB         :  BTAC   : WasteRatio
   0 :   24.993 :  2bit            :  61.11%  : (100.00% of 9.58%) :  68.11% : ( 54.37% of 29.30%) :  33.74% :  38.41%
-----------------------------------------------------------------------------------------------------------------------------
Proc :  nCommit   :   nInst    :  AALU   :  BALU   :  CALU   :  LALU   :  SALU   :  LD Fwd :    Replay    : Worst Unit  (clk)
   0 :  10139922  :  10139922 :  48.38% :  18.22% :   0.05% :  25.76% :   7.60% :  0.00%  :      N/A     : MUNIT_AALU  0.22
-----------------------------------------------------------------------------------------------------------------------------
--------
Proc  IPC     uIPC    Active        Cycles    Busy   LDQ   STQ  IWin   ROB  Regs    IO  maxBr  MisBr Br4Clk brDelay
   0   1.26   1.26   1.00         8056756     62.9   0.2   1.8   1.8   0.1   0.7   0.0   0.0   0.0    0.0   13.1
*****************************************************************************************************
Cache           Occ  AvgMemLat MemAccesses  MissRate  (  RD ,    WR,    BUS)
IL1(0)          0.0  2.2       3608033       0.18%    ( 99.8%,  0.0%,  0.0%)
-----------------------------------------------------------------------------------------------------
DL1(0)          0.0  4.9       3347212       0.23%    ( 99.8%, 97.2%,  0.0%)
-----------------------------------------------------------------------------------------------------
ITLB(0)         0.0  2.2       3607979       0.00%    (100.0%,  0.0%,  0.0%)
L2(0)           0.0  43.7      14186        24.61%    ( 75.4%,  0.0%,  0.0%)
Memory(0)       0.0  0.0       0             0.00%    (100.0%,  0.0%,  0.0%)
PTLB(0)         0.0  4.9       3339493       0.16%    ( 99.8%,  0.0%,  0.0%)
*****************************************************************************************************
```

e) Fill out the following table with the branch prediction rate:

Benchmark	2bit	Taken	NotTaken
blackscholes			
crafty			
kernel_dhry			
sjeng			

f) Let's take the NotTaken case as the reference machine, what are the speedups of the following:

	2bit	Taken
blackscholes		
crafty		
kernel_dhry		
sjeng		
Geometric Mean		

g) Why do different applications behave differently based on the branch predictor?

h) How can branch predictors be designed to incorporate more information in order to better predict branches?