ECE411 MP3 CP2 Report

Group: ZerotoOne

TA meeting:
We met our TA, Ahmed, during this checkpoint period. The following are some of the useful advice and design recommendations that we received:

1. How to choose the size of the L2 cache: first assign it to 1KB. If this slows the frequency, then reduce the size.
2. Split the data array in the L1 caches into 8-bit width
3. Recommend the size of the victim buffer to be 4-cache-line
4. If the clock frequency is not promising in the L2 cache design, recommend pipeline the L2 cache or add additional stage/latch in the arbiter to increase the frequency.
5. Preload the next-cache-line data as the prefetching algorithm.

Progress Report:
By checkpoint 2, we successfully replaced the magic memory with L1 caches and a arbiter. Its functionality was thoroughly tested by our own test program in addition to the given tests.
A critical problem we solved is how to deal with two consecutive stalls caused by reading instruction and data:

> Our original design would cause the stall controller to ignore the mem_resp signal corresponding to the first memory access of the two. The solution was to build a memory access proxy, which keeps the mem_resp signal high until the pipe starts moving.

In addition to the required components in checkpoint 2, we finished implementing the majority of the data forwarding components. We also found the following design deficiency:

> Our branch predictor would face the same data dependency hazard as the alu unit. We should either consider the data forwarding from EX, MEM, and WB to branch predictor, stall the pipes if the dependant data is in the EX, or put branch predictor to EX stage.

Besides the implemented functionalities, the design of the following parts are completed and ready to be implemented:

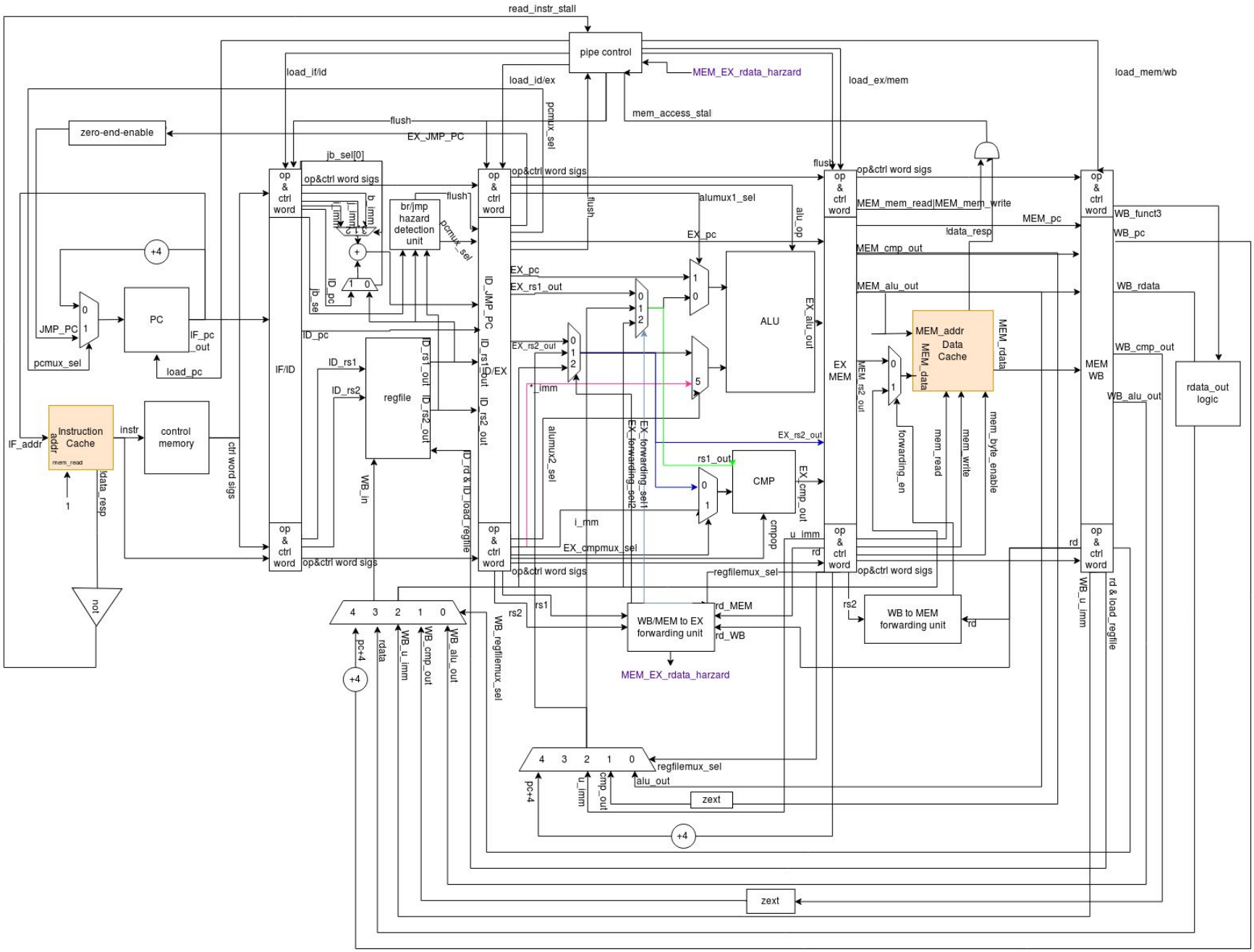
1. The rest components of data forwarding
2. Static branch predictor
3. L2 cache design

Contribution:
Haichuan Xu: L1 instruction cache and data cache implementation and debugging
Haiyang Zhang: Implementation of the arbiter, memory access proxy and debugging
Yuan Ma: Improvement of the forwarding design and hazard detection unit and debugging

Roadmap:
By checkpoint 3, we expect:

1. Finishing all the requirements on the manual: both the control hazard detection and the data forwarding

2. Finishing the design of some advanced cache features: eviction buffer and victim cache


read_instr_stall
pipe control
load_if/id
load_id/ex
load_ex/mem
load_mem/wb
MEM_EX_rdata_harzard
mem_access_stal
flush
EX_JMP_PC
zero-end-enable
pcmux_sel
jb_sel[0]
op & ctrl word
op&ctrl word sigs
br/jmp hazard detection unit
+4
JMP_PC
PC
IF_pc_out
load_pc
IF_addr
Instruction Cache
instr
control memory
ctrl word sigs
IF/ID
ID_pc
ID_rs1
ID_rs2
regfile
ID_rs1_out
ID_rs2_out
ID_JMP_PC
ID/EX
i_imm
alumux1_sel
alumux2_sel
alu_op
EX_pc
EX_rs1_out
EX_rs2_out
ALU
EX_alu_out
CMP
EX_cmp_out
cmpop
EX_cmpmux_sel
u_imm
rd
rs1_out
EX_forwarding_sel1
EX_forwarding_sel2
ID_rd & ID_load_regfile
WB_in
EX MEM
MEM_mem_read|MEM_mem_write
MEM_pc
ldata_resp
MEM_cmp_out
MEM_alu_out
MEM_addr
MEM_data
Data Cache
MEM_rdata
MEM_rs2_out
forwarding_en
mem_read
mem_write
mem_byte_enable
MEM WB
WB_funct3
WB_pc
WB_rdata
WB_cmp_out
WB_alu_out
rdata_out logic
WB_u_imm
rd & load_regfile
ldata_resp
not
1
4 3 2 1 0
pc-4
rdata
WB_u_imm
WB_cmp_out
WB_alu_out
WB_regfilemux_sel
rs1
rs2
WB/MEM to EX forwarding unit
rd_MEM
rd_WB
WB to MEM forwarding unit
regfilemux_sel
zext