Lab 5 Readme Lab Report
Dhruv Sandesara (djs3967)

Note: Only the changes I made on top of lab 4 are mentioned in this document. The immediate next pages contain the final state diagram, micro-sequencer and the data-path for lab5 with the changes made highlighted. For reference the original Lab 4 readme has also been attached afterwards.

1 Changes made to State diagram:

- The biggest change that we made for memory access is that whenever we try to put a value in the MAR we save a return state number for where to go after the VA to PA translation is completed.
- We also load the MAR values to the VA and then we decide where to direct the state.
- The states with the aformentioned changes are states 8,26(new number), 2, 6, 7, 3, 50,46,37, 18,19.
- Then the redirection falls in 3 categories. In case of Word sized access we go to state 45 and 56 for the LDB and the STB instructions. There we check if the BUs value latched to the MAR had a 0 in bit 1 and then throws an exception. If not then the redirection happens to one of the 2 remaining redirections.
- One of them is state 51 which is the Common entry point for any load mem instruction. Or to state 57 which is the common entry point for the store mem instruction. The entry state end the 2 states following that do the same thing.
- They Load the MAR with the PTBR added with the Virtual page number left shifted by one as the PTE is 16 bits.
- Then we wait for the mdr to be loaded with the PTE. When ready we put pte entries in the MDR but at the same time check for a Mem_PROT or Page Fault.
- This logic is by lookin at bit 3 and the Priv and if Priv is 1 and Prot is 0 then we send the next control signal for mem prot as 1.
- We also check the bit 2 to get if the page entry is valid or not. And if not we set the next control signal for Page Fault to 1.
- Then we have the third state for Load and Store diverge as we change us the MRD ref bit and set it to 1 for both. But we only set the dirty/ modified bit to 1 in Store.
- In this state we also calculate the Excec vector incase the Prot or Page Fault control signals were detected.
- Then if an exception was detected we go to the common Exception state of 39.
- If not we store back our Modified PTE to the page table.
- This was done in state 41 which is again common for both the load and store which when ready goes to State 43 which loads up the MAR with the PFN with the VA offset bits in 8:0.
- Then in this special state we go back to the ret state number that was stored in the reg before.
- Another major change that I made to make sure the prog fit in 64 state was to consolidate the exception handler vector loading case to 1 common one instead of the 4 different one for lab 4.
- Thus my state 39 loads up the Vector with the EXCV and also loads the PSR to the Temp MDR as the MDR will be overwritten in the address translation.
- This happens in the Interrupt handler case state 59 as well. This temp mdr is popped off into MDR in the common int/exc handler case of state 43.
- One last change was for state 10 and 11 I kept the states vector go to a hard coded 0x05 value.
- Now including all the LDVA Setting Ret addresses we have covered all the state diagram changes

2&3 Changes to the data-path with each control signals role explained:

- The first thing I added was the RetJ register. This was to know which state number to go back to once address translation is done. This has with it associated the R bits which have the state. The LDRetJ which loads the value and the GotoRet which basically selects this value as the next state.
- The next 2 values which were modified and added were the Mem_Prot and Mem_Page_Fault. The way the page fault works is to basically look at bit 2 of MDR to see if it is resident in mem.
- The other is the Mem_PRot which checks bit 3 to check if it is a protected page and if it is checks if the Priv is in user mode. If it is a Prot error is thrown. This is to detect a mem PROT access
- The next logic is the PTBR which holds the PTBR values and has nothing associated with it.
- Next change was the adding of VA with the LDVA signal to load the VA so we can use it without writing over it with the physical address.
- Then we have a MAR SEL to select the value of the bus or to use the calculated values from the PTBR + VPN LSHF by1 or the PFN with the VA offset. The diagram shows 1 bit MUX but it actually is a 2 bit so select one of these 3 values.
- Next we also use a TempMDR so as that our original value is not overwritten by PTE entry. So we have a LD signal with it. We can select the MDR input from either the previous logic or the TempMDR.
- Next we also have SetMDR bit 0 and 1 signals to write back the modified and reference values in the page table
- Finally we have the EXCV replacing the hardcoded values of vectors 0x02 and 0x03. This also has a Ld signal with it which is unfortunately forgot to include on the paper but the code does it. It checks if and alignment error had occurred then we would write the 0x03 vector else if it was a prot error, then write vector 4 else if page error then vector 2. The INTV and unknown vector are hardcoded in their own states.
- That does it for the changes to the data-path and control signals.

4 Changes made to the MicroSeq:

- I removed whatever logic we had for the Mem Prot and Mem align. We still have the Interrupt signal changing bit 3 to 1 when condition code is 4 to either fo to state 51 or 59.
- We then check if it is condition code 5 and if alignment problems occurred. And we check if the Condition is 6 and a Prot or Page problem occurred. If they did we go to common state 39 and if not we just go to the J bits
- The finally we check if we want to go back to the Ret value. If so we load J with the RetJ Reg value and if not just go to the original J values.
- That does it for the changes to the MicroSeq.
-