EE 460N Lab 5
Raiyan Chowdhury
rac4444

This README describes my implementation of the Lab 5 assignment.

The State Diagram

The changes made to the state diagram are attached in a picture at the end of this document. In addition to the changes made in Lab 4, six states have been added for the purpose of virtual address translation. Firstly, states 18, 19, 2, 3, 6, and 7 have all been changed so that the virtual address is stored in the new VA register instead of the MAR. After this, there is a state that checks for an unaligned exception. Then, the state in which memory access would normally occur has its state number saved in a new register called “Saved_J” and the state machine branches to state 51, which marks the beginning of address translation.

In state 51, the address of the PTE that we want is loaded into the MAR, using the page number of the virtual address stored in VA and the PTBR. Then the PTE is read into the MDR, and the state machine waits to see whether there is a protection or page fault exception. If there isn’t, then bit 0 of the MDR (the reference bit) is set. If the memory access in question is a write (STB or STW) then bit 1 (modified) is also set. The state machine checks whether or not this is a write access by checking the value of a register called “is_W,” which is set prior to the address translation. Next, the PTE is written back to the page table. Finally, the frame number from the PTE and the offset from the virtual address is written into the MAR, and the state machine branches back to the appropriate state based on the value inside Saved_J and memory access occurs normally from here.

In order for this design to work, some changes had to be made to my implementation of Lab 4. In Lab 4, when executing a STB or STW instruction, the state machine checked for an unaligned exception in the same state in which the MDR received the contents of the source register. However, the MDR is used in the address translation, but the unaligned exception still has to be checked prior to said translation (as it has priority over protection or page fault exceptions). So, the state prior to the translation is changed to only check for an unaligned exception, and the MDR receives the source register’s contents in a separate state after the translation. This means two more states were added in addition to the previously mentioned six, for a total of eight additional

states. This leaves one state left unused, as my Lab 4 implementation left nine states unused.

The Data Path and Control Signals

Changes to the data path are included in a diagram at the end of this document. Three new registers were added--VA, PTBR, and is_W. VA is simply used to store virtual addresses prior to address translation. The PTBR register contains the PTBR value used in translations. The is_W register, as described previously, is used as a way for the state machine to know whether or not the M bit of the PTE needs to be set.

Two new tristate drivers have been added to drive the bus with values stored in the new structures. GatePTBR is used to put the value of PTE's address onto the bus. Bits [15:8] of the PTBR and bits [15:9] of the VA are combined and put on the bus via GatePTBR. VA[15:9] is right shifted 8 times before this--this places the relevant bits where we want them, as they represent the page number and we want the page number to be left shifted by 1 (or multiplied by 2). In addition, GateVA is used to put the translated physical address onto the bus. VA bits [8:0] and MDR bits [13:9] are added together before being put on the bus.

Two new control signals have been added, Has_PTE and SET_MR. Has_PTE is simply a way for the machine to know that the MDR currently contains the PTE, which then triggers the checks for protection and page faults. This is done since we don't want the state machine to check for these exceptions every single time the MDR gets any value. The SET_MR control signal is used as an input into the MUX that feeds into the MDR, in addition to MIO.EN. When SET_MR is set, the control mux selects the modified MDR as input, where the reference and, if appropriate, modified bits have been set.

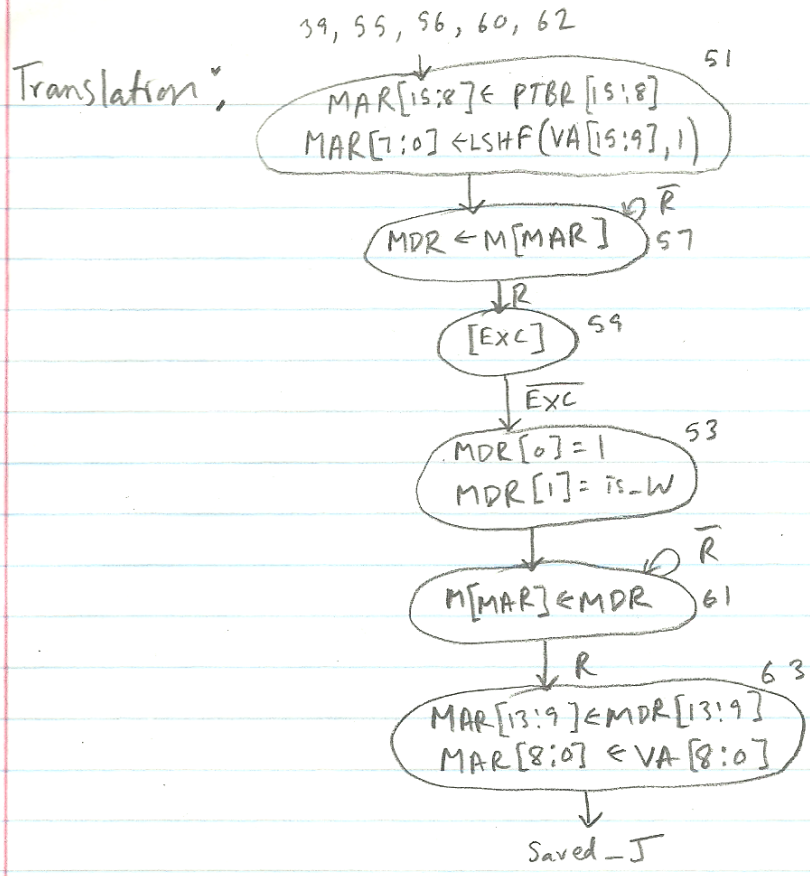
The Microsequencer

Changes to the microsequencer were made to accommodate the new address translation states. Several new control signals have been added, and the Saved_J register is also used here.

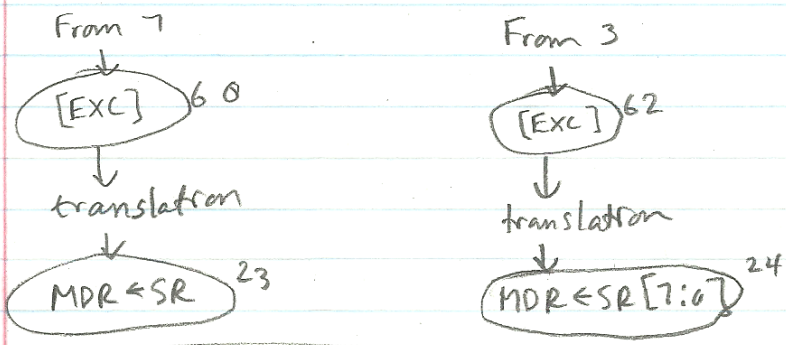
Two new input signals have been added to the mux--VAT and SJ. Although there is no need to add two new bits for only two additional input choices, I chose to add separate bits so that their meaning is distinct from each other and from the previously existing EXC and IRD bits, which have nothing to do with address translation.

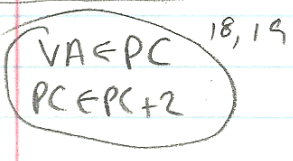
“VAT” essentially stands for “Virtual Address Translation,” and this bit is set in states 39, 55, 56, 60, and 62. When VAT is set, the state machine automatically branches to state 51, which is where address translation occurs. VAT also doubles as a load signal for Saved_J, and the “J” bits are saved in this register when VAT is set so that the state machine knows where to return after address translation. When “SJ” (“Saved J”) is set, the contents of Saved_J are chosen and the state machine returns to the appropriate state to continue memory access.

Translation:



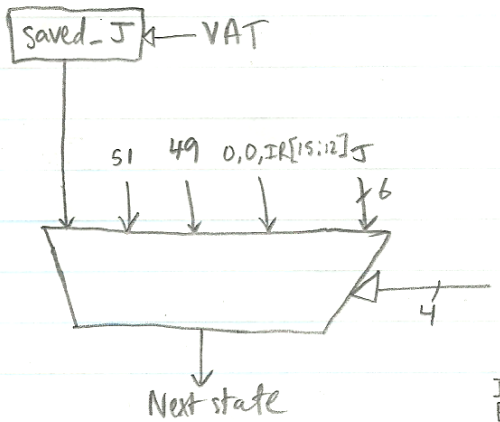
Changes from Lab 4:





States 2, 3, 6, and 7 also have MAR replaced with VA

Microsequencer:



all 0 → picks J
IRD = 1 → picks opcode
EXC = 1 → picks 49
VAT = 1 → picks 51, stores J in register
SJ = 1 → picks previously stored J

Data path: