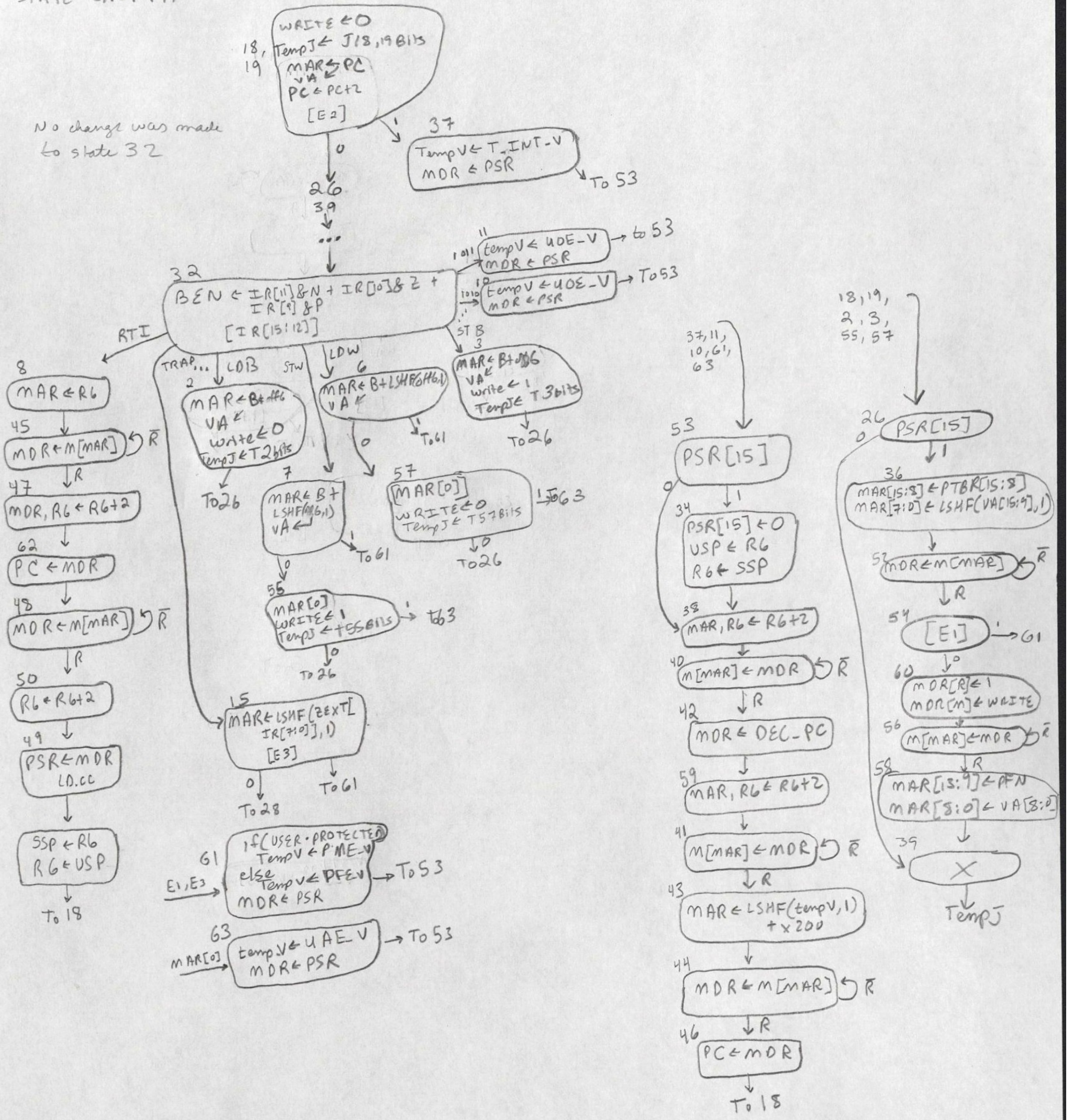
LAB 5 DOCUMENTATION
EE460N
11/17/2014

MODIFIED STATE GRAPH
ADDITIONAL DATA PATH STRUCTURES
MODIFIED MICROSEQUENCER
NEW CONTROL SIGNAL BITS

George Neal
gln276

STATE GRAPH

No change was made to state 32



DESCRIPTION OF NEW STATES IN STATE GRAPH

Included on the state graph are all the new states that enable timer interrupts and the 3 types of exception handling: memory protection, unaligned access, and unknown opcode. State 32 is included for clarity, and remains unchanged from the original LC3b state graph.

Path from state 8 back to 18 (RTI Handling):

This sequence of states restores the machine back to its state before the occurrence of the interrupt/exception that changed the process state from USER to SUPERVISOR. It does so by popping both the old PC and old PSR off of the Supervisor Stack, and latching these values to their respective registers. The sequence also restores the stack pointer (R6) to the User Stack Pointer (value before the interrupt/exception occurred).

NOTE: This implementation makes the following assumptions:

1. RTI is only called when the machine is in SUPERVISOR mode, and was previously running in USER mode.
    1. I.e., nested interrupts and exceptions are not supported.
2. The stack pointer (R6) is pointing to the same location that it was when the machine began handling the interrupt/exception.
3. Interrupt service routines and exception handlers save registers that are clobbered within routines, and restore them at the end of the routines.

Path from state 53 back to 18 (Interrupt/Exception preparation)

This sequence of states switches the state of the machine from USER mode to SUPERVISOR mode (or remain in SUPERVISOR mode if such is the case). The switch from USER to SUPERVISOR mode occurs by switching PSR[15] from a 1 to a 0, storing the Stack pointer to the USP register, and latching the SSP register value to R6. Next, the rest of the current machine state information is saved so that it may be retrieved when the interrupt/exception is serviced. This is done by pushing the decremented PC and PSR onto the Supervisor Stack. Finally, the interrupt/exception vector is left shifted, and added to the base address of the interrupt/exception vector table (x200), and this value is latched to the PC. The machine then returns to state 18, and instruction execution proceeds normally in SUPERVISOR mode.

Path from state 26 to TempJ (Virtual Address Translation)

This sequence of states translates the virtual address located in the MAR into a physical address using a single layer translation scheme. It first reads the PTE of the page containing the VA information by loading the PTBR + indexed page number offset into the MAR, and loads it into the MDR. The M bit is set if the access to the virtual address was a write, and cleared if it was a read. The R bit is always set on an access. The PTE is checked for access violation conditions, as well as page frame validity (residence). If successful, the MAR is loaded with the physical address, and the path uses the TempJ register as a vector pointing back to the proper state as specified by the state that branched to this path.

STATE GRAPH (CONTINUED)

NOTE: States 2, 3, 6, 7, 18, 19, 55, 57, and 61 have been modified to implement Virtual Memory:

1. States 2, 3, 6, 7, 18, 19, now load the VA Register with the same address of the MAR.

2. States 2, 3, 55, 57, 18, 19, now load the WRITE register with the appropriate value (used to set/clear M bit of PTE), and load the TempJ register with the appropriate state vector.

3. If the system is in SUPERVISOR mode, then the translation process is skipped (microbranch at state 26). This prohibits the ISR, and other SUPERVISOR level code, from using USER space memory locations.

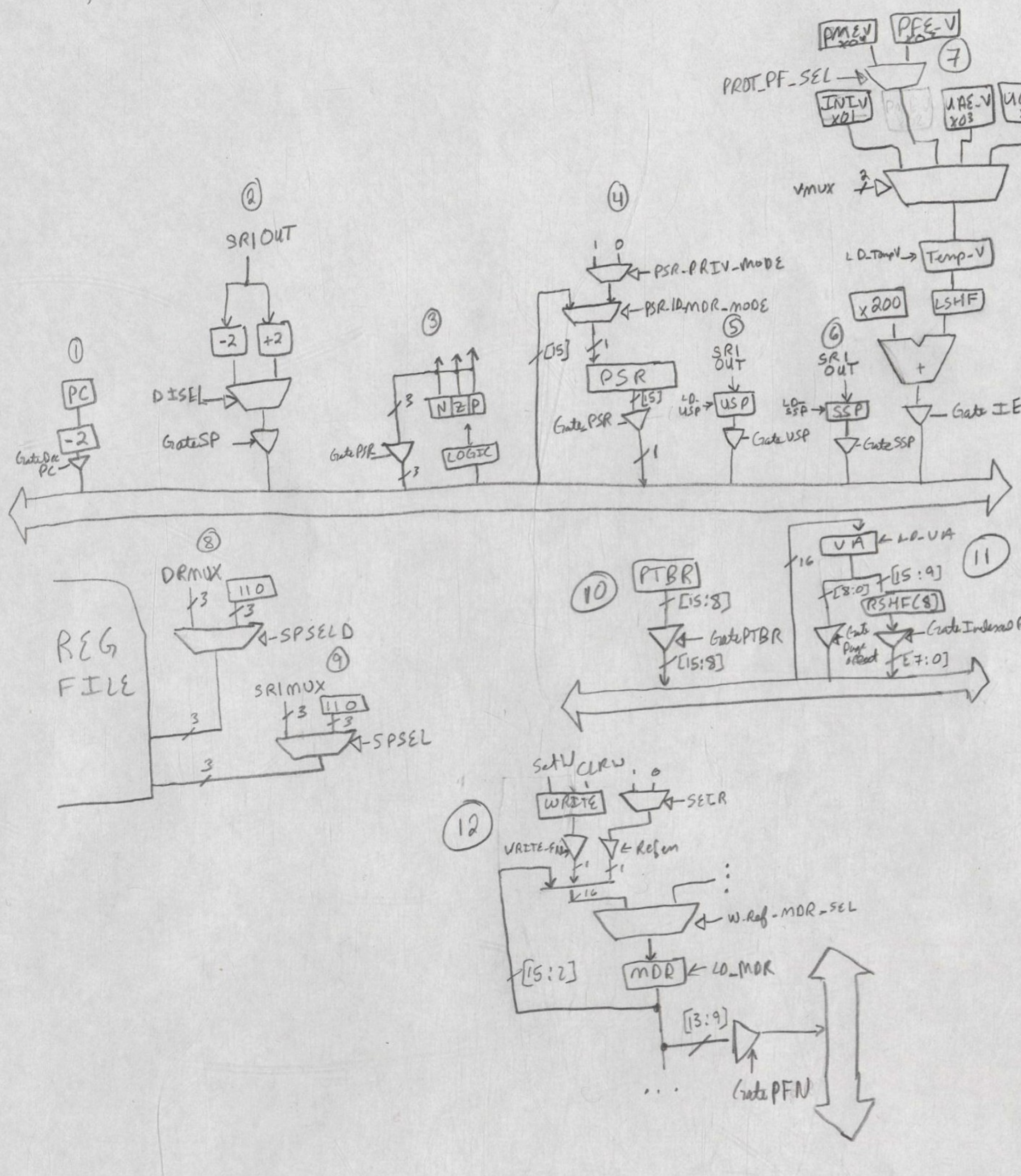
New interrupt/exception initialization states (10, 11, 37, 61, 63):

These states are microbranched to when the conditions for a timer interrupt or exception are met. Each of them loads the Temp_V register with the appropriate interrupt/exception offset to be later added to the base of the interrupt/exception vector table (x200). Each state also loads the MDR with the current PSR (PSR[15] and Condition Codes).

New Branching states (55, 57):

These states simply check MAR[0] and determine whether or not an unaligned access is occurring. They are new states because the previous states check for protected memory exceptions (protected exceptions have priority to unaligned access exceptions), and must occur in a different state. They could not be placed in the subsequent state either though, because memory is accessed in the subsequent state.

DATA PATH, NEW STRUCTURES

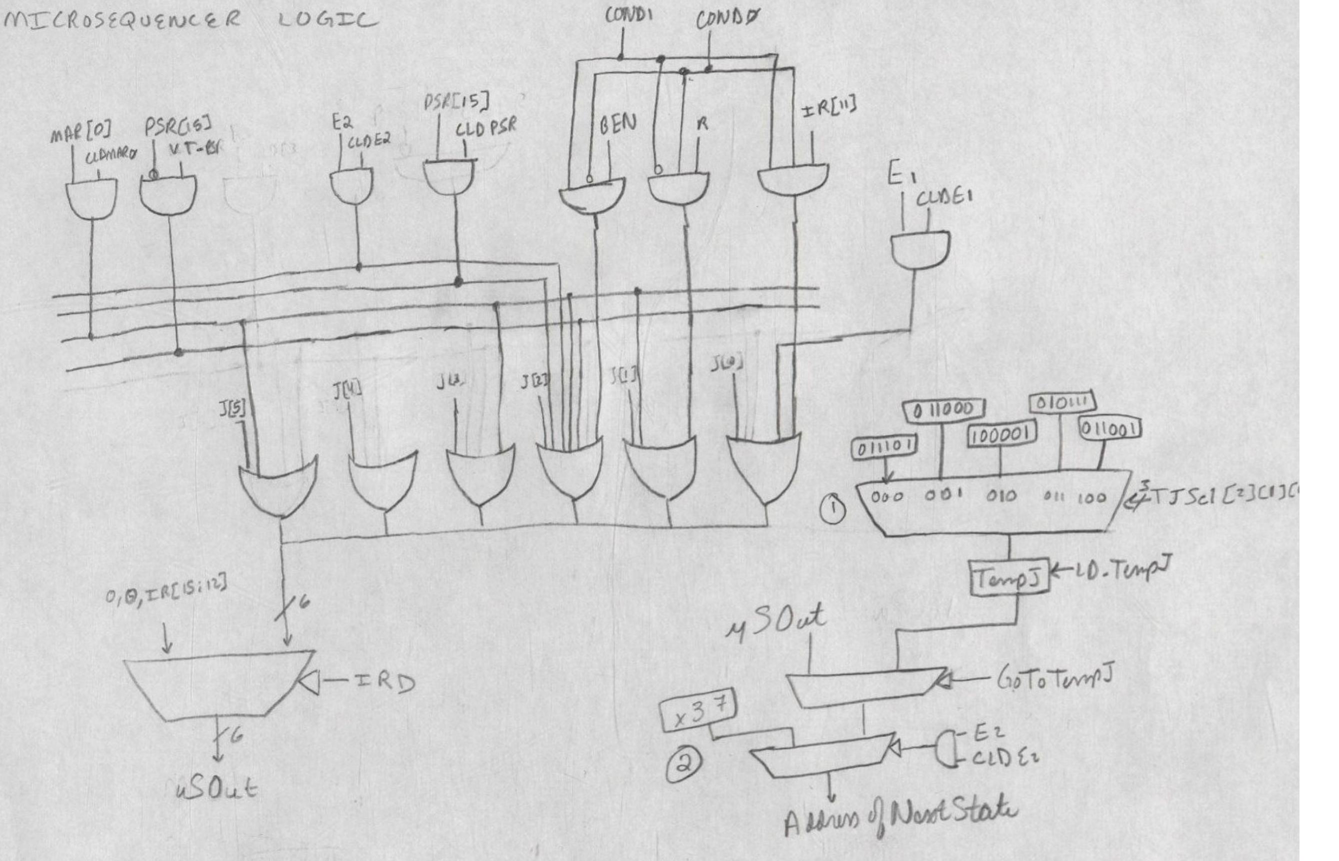


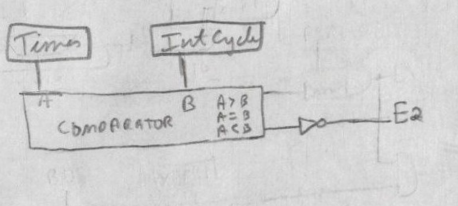
DESCRIPTION OF NEW DATA PATH STRUCTURES

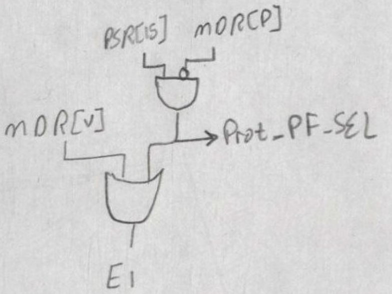
Included in the Data Path sheet are all the structures I added to implement timer interrupts and the 3 exceptions. I will describe the additional structures in order of left to right, top to bottom.

1. The decrementer and gate were added to allow the decremented PC to be pushed onto the Supervisor Stack in state 42.
2. The decrementer and adder are connected to the output of SR1 form the RegFile. The uCode must assert the appropriate signals to make sure the RegFile is outputting R6, the stack pointer. A MUX selects which value to gate to the bus, and the values are used when pushing/popping onto the Supervisor Stack in states 38, 59, 47, 50.
3. The tristate buffer connected to the CC bits was included to save the values when switching control of the machine from USER process to SUPERVISOR process.
4. From top to bottom: a MUX selects which privilege mode to write to the PSR register. Next, another MUX selects either this signal, or BUS[15] (to allow the PSR to be restored from memory in state state 49), and applies either signal to the input of the PSR register. The current PSR register is only 1 bit wide, but may easily be extended to support more features like privilege.
5. USP register saves the User Stack Pointer when switching from USER mode to SUPERVISOR mode.
6. SSP register saves the Supervisor Stack Pointer.
7. From top to bottom. First is a set of registers that include the offsets for the timer interrupt and 3 exceptions for the interrupt/exception vector table. The appropriate offset is chosen based on which event triggered a microbranch to state 10, 11, 37, 61, or 63. The offset is latched into a register, Temp_V in one of these states. Later in the control path (state 43), this value is left shifted and added to the interrupt/exception vector table, and latched to the BUS to be latched into the MDR.
8. This MUX specifies to the RegFile that the destination register is the Stack Pointer (R6).
9. This MUX specifies to the RegFile that the source register (SR1) is the Stack Pointer (R6).
10. The PTBR register holds the base address of the System Page Table. Its high 8 bits are loaded onto the bus when GatePTBR is asserted.
11. The VA register holds a copy of the MAR used to translate from virtual to physical addresses. Asserting GatePageOffset loads the low 8 bits of the register onto the bus. Asserting GateIndexedPageNo loads the high 7 bits that have been right shifted by 8 onto the bus (only BUS[7:0] are driven by this tristate bus.
12. This logic controls the WRITE register that writes to the M (modify) bit of the MDR (PTE) and the R (reference) bit. The information is multiplexed into the MDR using a select line. Also added is the GatePFN tristate buffer. This loads the PFN of the PTE (MDR[13:9]) onto the BUS.

MICROSEQUENCER LOGIC







DESCRIPTION OF NEW STRUCTURES FOR MICROSEQUENCER

Included in the microsequencer logic sheet is the new logic used to implement timer interrupts and the 3 exceptions. The new control bits E1, E2, E3, and MAR[0] and PSR[15] are used to take microbranches in the new states, and are included in the microsequencer.

E1:
The E1 bit has been slightly changed from lab4. Here, it signals that either the V bit of the PTE is 0, or that the system is in USER mode and the P bit of the PTE is 0 (protected space).

E2:
The E2 bit signals that the timer interrupt conditions have been met. When CLD_E2 is asserted and E2 is high, Jbit 4 is set to high, and will branch the current microinstruction to state 37, which prepares the machine to take the timer interrupt service routine. (2)
E2 is calculated by comparing the current Timer value to the Int_Cycle value (a constant which indicates which cycle to interrupt on). If the Timer>=Int_Cycle (not(Timer<Int_Cycle)), then E2 is set to high.

MAR[0]:
MAR[0] is used to determine whether or not an unaligned access exception should occur. If, when the MAR is loaded for STW and LDW instructions, MAR[0] == 1, then the exception should occur. If CLD_MAR0 is high and MAR[0] == 1, then Jbits 5, 3, 2, and 1 are set to high, and the state machine branches to 63 assuming bits 4 and 0 are high.

PSR[15]:
PSR[15] is used in state 53 to determine whether or not the machine is already in SUPERVISOR mode, meaning that the PSR and USP (R6) don’t have to be staved, and that SSP doesn’t have to be loaded into the Stack Pointer (R6). This is vectored by asserting CLD_PSR
PSR[15] is also used to determine whether or not to translate the virtual address located in the MAR in state 26. If the system is in SUPERVISOR mode and PSR_VT is asserted, then the microbranch will vector to the end of the translation process, ignoring it.

GoToTempJ (1)
The GoToTempJ signal is asserted when TempJ is used to vector to the next state of the machine. The multiplexer above the TempJ register is used to load the appropriate vector into TempJ based on which state is loading it (2, 3, 18, 19, 55, 57).

NEW CONTROL SIGNAL BITS

```
Gate_PSR      - Gates PSR[15] and Condition Codes to BUS
Gate_USP      - Gates USP to BUS
Gate_SSP      - Gates SSP to BUS
Gate_SP       - Gates in/decremented stack pointer (R6)
Gate_IE_V     - Gate the interrupt vector table address to BUS
Gate_Dec_PC   - Gates the decremented (by 2) PC to BUS
LD_USP        - Enable USP to load the stack pointer (R6)
LD_SSP        - Enable SSP to load stack pointer (R6)
LD_PSR        - Enable PSR to load bit 15 of the BUS
              - Note: Control must also assert LD.CC when loading PSR
LD_TEMP_V     - Load temp register with interrupt/exception offset
SP_SEL        - Select stack pointer to be output from SR1OUT
SP_SELD       - Select stack pointer to be written to in REGFILE
DI_SEL        - Select whether to increment or decrement stack pointer by 2
VMUX1         - Select bits determining which INT/EXC offset to output
VMUX0         -
TM_COMP_MUX   - Selects whether to compare timer or MAR (BUS)
PSR_PRIV_MODE - Select whether to make PSR USER or SUPERVISOR mode
PSR_MDR_MODE  - SELECT whether to load PSR with data from BUS or MODE selecter
CLD_E1        - Load signal bit for Protected Exception to uSequencer
CLD_E2        - Load signal bit for Timer Interrupt to uSequencer
CLD_E3        - Load signal bit for Protected Exception (TRAP specific) to uSequencer
CLD_MAR0      - Load signal bit for Unaligned Access Exception to uSequencer
CLD_PSR       - Load signal bit for determining USER/SUPERVISOR mode
/* New For Lab 5*/
GATE_PTBR             - Gates PTBR to BUS (BITS 15:8)
GATE_VA_OFFSET        - Gates VA page offset (BITS 8:0)
GATE_VA_IND_OFFSET_PN - Gate indexed page number (BITS 15:9)
GATE_PFN              - Gate PFN (BITS 13:9 of PTE (MDR))
LD_VA                 - Load VA with BUS
SET_WRITE             - Set Write register bit
CLR_WRITE             - Clear Write register bit
WRITE_ENABLE          - Allow Write bit to be written to PTE (modify bit)
SET_REFBIT_SEL        - Set or clear reference bit
REF_BIT_ENABLE        - Allow reference bit to be written to PTE
WR_REF_MDR_SEL        - Select default MDR input or WR/REF input lines
TEMPJ_SEL             - Select which value to load into TempJ register
LD_TEMPJ              - Loads TempJ register with selected values
VT_PSR                - Load PSR signal to skip translation if in supervisor mode
GOTO_TEMPJ            - Set next state to state specified in TempJ
LD_WR                 - Load WR with set or cleared bit
```