

```

// second instruction is in register fetch
    IDEXA <= Regs[IFIDIR[25:21]]; IDEXB <= Regs[IFIDIR[20:16]]; // get two registers
    IDEXIR <= IFIDIR; //pass along IR--can happen anywhere, since this affects next stage only!
// third instruction is doing address calculation or ALU operation
    if ((IDEXop==LW) |(IDEXop==SW)) // address calculation & copy B
EXMEMALUOut <= IDEXA +({16{IDEXIR[15]}}, IDEXIR[15:0]);
else if (IDEXop==ALUop) case (IDEXIR[5:0]) //case for the various R-type instructions
    32: EXMEMALUOut <= Ain + Bin; //add operation
    default: ; //other R-type operations: subtract, SLT, etc.
endcase
EXMEMIR <= IDEXIR; EXMEMB <= IDEXB; //pass along the IR & B register
//Mem stage of pipeline
    if (EXMEMop==ALUop) MEMWBValue <= EXMEMALUOut; //pass along ALU result
    else if (EXMEMop == LW) MEMWBValue <= DMemory[EXMEMALUOut>>2];
    else if (EXMEMop == SW) DMemory[EXMEMALUOut>>2] <=EXMEMB; //store
    MEMWBIR <= EXMEMIR; //pass along IR
// the WB stage
    if ((MEMWBop==ALUop) & (MEMWBrd != 0)) Regs[MEMWBrd] <= MEMWBValue; // ALU operation
    else if ((EXMEMop == LW)& (MEMWBrt != 0)) Regs[MEMWBrt] <= MEMWBValue;
end
endmodule

```

FIGURE e4.14.2 A behavioral definition of the five-stage MIPS pipeline with bypassing to ALU operations and address calculations.
(Continued)