Report – GP03-SWE3005-DR 01SEP-2017

Group 3, Moderator : 여혁수(2016312761)

Group members: 김경민, 유제영, 박정훈, 장민서

* Executive summary of what we learned from your group discussions

In our discussion, we discussed about control signals of MIPS processor, and executing procedures of many MIPS instructions. We first learned together about MIPS control signals. When the execution stage, There are a signal which do add, subtract or change the input of ALU on a basis of 2bits. Also when the mem stage, processor sometimes update the program counter, or branch the target to next instruction. Sometimes also accessing memory space to read memory and write something to other space. In write back stage, memread and memwrite signals make the load instruction go well. After learning about MIPS control signals, we talked about the way how MIPS instructions are executed utilizing various control signals. One of three kinds of instructions, there is R format instructions. In this cases, almost all of instructions use ALU controls and get output in ALU result. The cases of I format, many kinds of signals need very intricately. And the last cases called J format, it does not use some signals actively. For jumping, Just concatenating some bits of program counter to make address be specified. That’s all about our simple discussion summary because group discussion have quite short time. We can finally get some feeling that the simple instructions such as adding, subtracting, and shifting were also have complexed processes hidden in the processor when we looked deep into the processor.

Topic 1. Explain the 9 control signals required to implement the instruction cycle of MIPS processor.

1. Execution stage

1)RegDst: when deasserted, the register file destination number for the Write register comes from the rt field. When asserted, The register file destination number for the Write register comes from the rd field.

2)ALUOp: the value 00; The ALU performs an add operation. The value 01; The ALU performs a subtract operation. The value 10; The function field of the instruction determines the ALU operation.

3)ALUSrc: ALUSrcA; when deasserted, The first ALU operand is the PC. When asserted, The first ALU operand comes from the A register. ALUSrcB, the value 00, The second input to the ALU comes from the B register. The value 01, The second input to the ALU is the constant 4. The value 10, The second input to the ALU is the sign-extended, lower 16bits of the IR. The value 11, The second input to the ALU is the sign-extend, lower 16 bits of the IR shifted left 2bits.

2. MEM stage

1. Jump

: It is used to update program counter. This control signal is asserted for jump instructions to select the jump target address. That is, it enables loading the jump target address into the PC

1. Branch

: It asserts for branch instructions to select the branch target address as the next instruction address. Roughly similar with the Jump signal. That is, it is combined with a condition test boolean to enable loading the branch target address into the PC

1. MemRead

: Typically, MemRead and MemWrite are used to access the memory space and is used for memory read and memory write. It enables a memory read for ‘load’ instructions.

1. MemWrite

: It is used for ‘store’ or ‘sw’ implementation. That is, it enables a memory write for store instructions.

3. Write Back stage

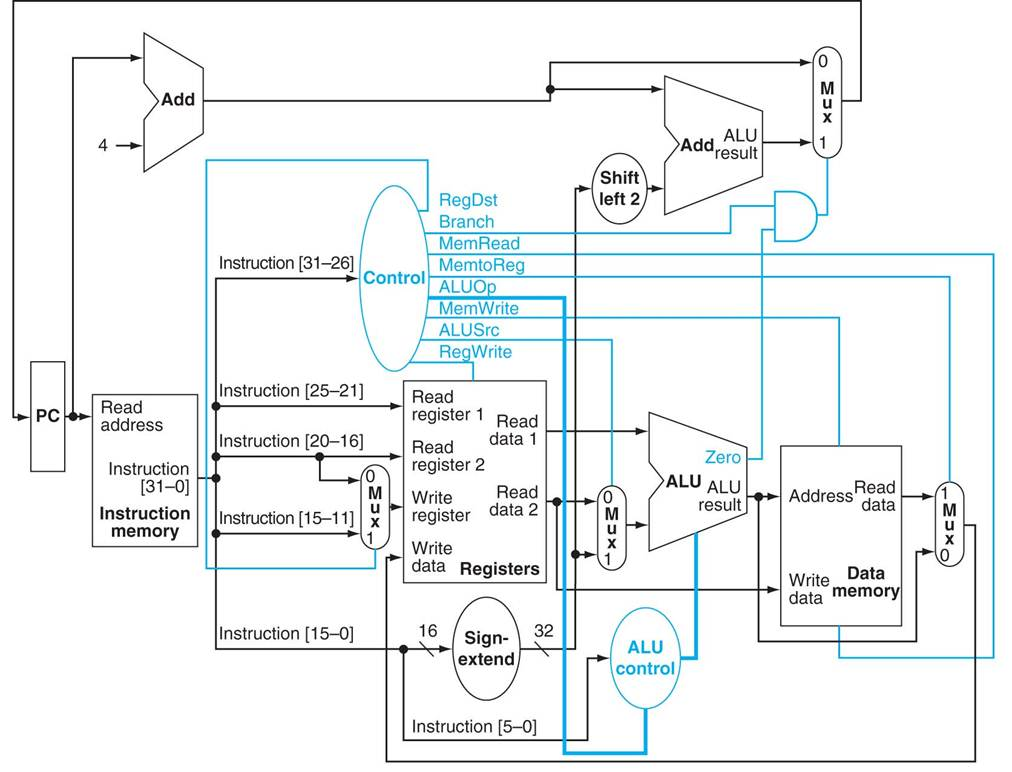
1)RegWrite: when asserted, The general-purpose register selected by the Write register number is written with the value of the Write data input.

2)MemtoReg: when deasserted, The value fed to the register file Write data input comes from ALUOut. When asserted, The value fed to the register file Write data input comes from the MDR.

Topic 2. Explain how MIPS instructions are executed on a block diagram of MIPS processor

1. R format

1. add & mul

: Basically, the data should pass the ALU unit to implement addi operation. Before solving, let’s take a look at the organization of the R-format instruction. First 6bits are for opcode, 5 bits for each ‘rs’, ‘rs’ and ‘rd’ register. In the righthand side, shamt(5bits) and func(6bits) are assigned. Data from the first two registers rs and rt are passed to register file (Read register1 and 2). And they are passed to ALU unit. In this case, the ‘func’ field is decoded and passes ‘ALUop’ signal to the ALU control to determine which calculation will be held. The result of the calculation gets back to the register file and then write into the rd register.

2) and & slt

: Like “add” and “mul” instructions, “and” and “slt” are also R format instructions, so the underlying data path is the same. It sends the information (rs, rt) of the two source registers are read from the instruction decode and register fetch stage to the input of the ALU.

After the operation is determined by the funct code, the ALU computes and outputs the two input data.

The output data is stored in the destination register. At this time, the number of the destination register is input to the write register, the result of the operation is input to the write data, and a control signal called RegWrite is allowed to be written.

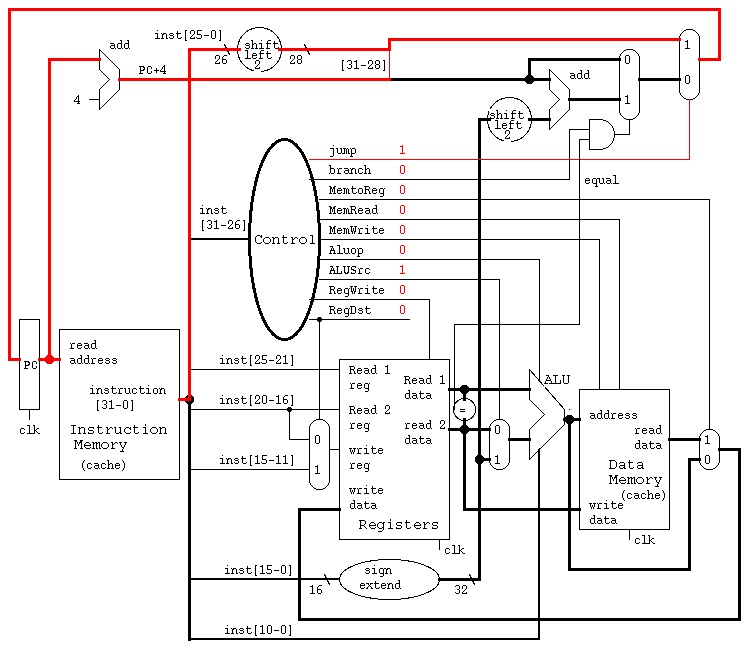
2. I format

Lw, sw, and bne are I format instructions, and the field bit positions of load or store instruction correspond to 31:26, 25:21, 20:16, 15: 0 in order of op, rs, rt, and address. The register rs is the base register plus the 16-bit address field to create the memory address. For the Load command, rt is the destination register for the loaded value. For the store command, rt is the source register, and the value of that register is stored in memory.

First, the lw command operates in five steps. The instruction is fetched from the instruction memory, the pc value is increased, and the register value is read from the register file. The ALU obtains the sum of the value read from the register file and the sign extended value of the lower 16 bits of the instruction (ALUSrc). This sum is used as an address for data memory access (MemRead) and the destination register is specified by bits 20:16 of the instruction. Finally, the data from the memory unit is written to the register file (MemtoReg, RegWrite). In the datapath, the lw command uses ALUSrc, MemtoReg, RegWrite, and MemRead as control signals.

The sw command uses ALUSrc and MemWrite as the control signals and has RegDst and MemtoReg as don'care terms. The major difference is that the memory control is writing (ALUSrc, MemWrite), not memory read, that the value of the second register read is used to store the data, and the data memory value is written to the register file Does not happen.

The branch command bne uses ALUOp and Branch as control signals. Unlike the previous two commands, ALUSrc is not used. After comparing the data read from the register with ALU, it moves to branch control logic by using ZERO output. Then use zero output to select one of the two candidates for the next program counter value (to branch target).

3. J format

1) jr

: The jr instruction is a J type instruction. So first we concatenate the 4 bits of the program counter to the address specified by 26 bits to obtain the address. Then you do not need to read or write the values of the other registers. Just jump to that address. So just set the Program Counter to this address via the MUX. I will jump right away. By using Opcode, We don’t need to touch any other part because only the jump part is activated.