type instruction and JReg depends on the function field only. When JReg is equal to '1', PCSrc (PC control unit output control signal) will be '10' to select the value of register Rs as input to PC.

Also, we need to store PC+4 in register Rd. To accomplish this, we need another multiplexer to select between the incremented PC, the ALU result and data memory out, to be placed on BusW. Also, we need to add a path from the output of the incremented PC to the input of this new multiplexer. A control signal 'RA' (Return Address) is needed to select between the incremented PC and the ALU result. The MemtoReg multiplexer selects between the output of the 'RA' multiplexer and the Data Memory output to place on BusW.

b) Show the values of the control signals to control the execution of the **jalr** instruction. If you need add a new control signal, please add it along with its value to the table below. Use the following table for ALUCtrl.

ALU function	4-bit ALU Control
AND	0001
OR	0010
XOR	0011
ADD	0100
SUB	0101
SLT	0110

The main control signals for the JALR instruction are the same for other R-type instructions, such as ADD and SUB. The ALU Control signals for the JALR instruction require JReg = 1, RA = 0 and ALUCtrl is a don't care. These control signals are shown in the table below:

RegDst	RegWrite	ExtOp	ALUSrc	MemRead	MemWrite	MemtoReg	ALUCtrl	J	Beq	Bne	RA	JReg
Rd = 1	1	X	X	0	0	0	XXXX	0	0	0	0	1

Q2. Processor Performance

Suppose we add the multiply and divide instructions. The operation times are as follows:

Instruction memory access time = 190 ps, Data memory access time = 190 ps,
Register file read access time = 150 ps, Register file write access = 150 ps
ALU delay for basic instructions = 190 ps, ALU delay for multiply or divide = 550 ps
Ignore the other delays in the multiplexers, control unit, sign-extension, etc.

Assume the following instruction mix: 30% ALU, 15% multiply & divide, 20% load, 10% store, 15% branch, and 10% jump.

a) What is the total delay for each instruction class and the clock cycle for the single-cycle CPU design?

Instruction Class	Instruction Memory	Register Read	ALU	Data Memory	Register Write	Total Delay
Basic ALU	190 ps	150 ps	190 ps		150 ps	680 ps
Mul & Div	190 ps	150 ps	550 ps		150 ps	1040 ps
Load	190 ps	150 ps	190 ps	190 ps	150 ps	870 ps
Store	190 ps	150 ps	190 ps	190 ps		720 ps
Branch	190 ps	150 ps	190 ps			530 ps
Jump	190 ps	150 ps				340 ps

Clock cycle = max delay = 1040 ps.

b) Assume we fix the clock cycle to 200 ps for a multi-cycle CPU, what is the CPI for each instruction class and the speedup over a fixed-length clock cycle?

Solution:
CPI for Basic ALU = 4 cycles
CPI for Multiply & Divide = 6 cycles
CPI for Load = 5 cycles
CPI for Store = 4 cycles
CPI for Branch = 3 cycles
CPI for Jump = 2 cycles

Average CPI = 0.3 * 4 + 0.15 * 6 + 0.2* 5 + 0.1 * 4 + 0.15 * 3 + 0.1 * 2 = 4.15

Speedup of multi-cycle over single-cycle = (1040 * 1) / (200 * 4.15) = 1.253

Q3. (10 pts) Consider the following MIPS code sequence:

```
a: add $t0, $s0, $s1
b: sub $t1, $s2, $t0
c: xor $t0, $s0, $s1
d: or  $t2, $t1, $t0
```

a) (5 pts) Identify all the RAW dependencies between pairs of instructions.

Instruction b is dependent on instruction a ($t0)
Instruction d is dependent on instruction b ($t1)
Instruction d is dependent on instruction c ($t0)