第四章：

4.1

4.1.1 The values of the signals are as follows:

RegWrite	MemRead	ALUMux	MemWrite	ALUop	RegMux	Branch
1	0	0	0	10	0	0

图 4-2 中一共有 7 个信号，信号名称可部分参考图 4-17。

信号说明：

RegWrite：寄存器写； Memread：存储器读 Memwrite：存储器写

ALUop：ALU 操作控制，10 表示 R 型指令。

ALUMux ： 0 (from Reg)/ 1 (from Imm) .

RegMux: 0 (from ALU)/ 1 (from Mem).

Branch：1 表示在执行分支指令,0 表示执行非分支指令。

4.1.2 除了分支目标地址加法器 Branch Add 和存储器 Memory 未用到，其他部件都用到。

4.1.3 有输出但未用到: Branch Add；

没有输出: Memory。

4.5

4.5.1 The data memory is used by LW and SW instructions, so the answer is:

25% + 10% = 35%

4.5.2 The sign-extend circuit is actually computing a result in every cycle, but its output is ignored for ADD and NOT instructions. The input of the sign-extend circuit is needed for ADDI (to provide the immediate ALU operand), BEQ (to provide the PC-relative offset), and LW and SW (to provide the offset used in addressing memory) so the answer is:

20% + 25% + 25% + 10% = 80%

4.8

4.8.1

Pipelined	Single-cycle
350 ps	1250 ps

4.8.2

Pipelined	Single-cycle
1750 ps	1250 ps

4.8.3

Stage to split	New clock cycle time
ID	300 ps

4.8.4

a.	35%

4.8.5

a.	65%

4.8.6　单周期：采用长周期，1Cycle0=1250ps，指令总延迟1250ps；

Pipeline采用5个周期：1Cycle1=350ps，指令总延迟1750ps；

多周期采用3-5个周期：1Cycle2=350ps；平均周期数：0.2*5（LW指令）+0.6*4（SW、ALU指令）+0.2*3（BEQ指令）=4.0 Cycle2.

平均指令延迟1400ps。

4.9

4.9.1

Instruction sequence	Dependences

`I1: OR R1,R2,R3`
`I2: OR R2,R1,R4`
`I3: OR R1,R1,R2`
	RAW on R1 from I1 to I2 and I3
	RAW on R2 from I2 to I3
	WAR on R2 from I1 to I2
	WAR on R1 from I2 to I3
	WAW on R1 from I1 to I3

4.9.2 In the basic five-stage pipeline WAR and WAW dependences do not cause any hazards. Without forwarding, any RAW dependence between an instruction and the next two instructions (if register read happens in the second half of the clock cycle and the register write happens in the first half). The code that eliminates these hazards by inserting NOP instructions is:

Instruction sequence	
OR R1,R2,R3	
NOP	Delay I2 to avoid RAW hazard on R1 from I1
NOP	
OR R2,R1,R4	
NOP	Delay I3 to avoid RAW hazard on R2 from I2
NOP	
OR R1,R1,R2	

4.9.3 With full forwarding, an ALU instruction can forward a value to EX stage of the next instruction without a hazard. However, a load cannot forward to the EX stage of the next instruction (by can to the instruction after that). The code that eliminates these hazards by inserting NOP instructions is:

Instruction sequence	
OR R1,R2,R3	
OR R2,R1,R4	No RAW hazard on R1 from I1 (forwarded)
OR R1,R1,R2	No RAW hazard on R2 from I2 (forwarded)

4.9.4

有旁路时，无阻塞，3条指令执行时间为5+2=7cycle,时间为

7*300=2100ps;

无旁路时，3条指令执行时间为7+4(stall)=11cycle，时间为

11*250=2750ps.

加速比：2750/2100=1.31

4.9.5　* 需在I2与I3之间插入2个NOP。

4.9.6

仅有ALU-ALU旁路时，阻塞2次，3条指令执行时间为5+4=9cycle,时间

为9*290=2610ps;

无旁路时，3条指令执行时间为7+4(stall)=11cycle，时间为

11*250=2750ps.

加速比：2750/2610=1.05