コンピュータ設計論2006年度試験問題(A)

2007.2.8

問題1

パイプライン化された MIPS マシンが次のプログラムを実行するときの様子を、解答用紙の図上に示せ

アドレス	命令			
0	lw	\$1,	0(\$8)	
4	lw	\$2,	0(\$12)	
8	sub	\$3,	\$2,	\$1
12	beq	\$3,	\$4,	1
16	lw	\$5,	0(\$3)	
20	lw	\$6,	0(\$16)	
24	add	\$6,	\$5,	\$6
28	sw	\$6,	0(\$20)	

プログラム開始時のマシンの状態は次の通りである。

1. レジスタ PC, IF/ID, ID/EX, EX/MEM, MEM/WB: 0
2. レジスタファイル Reg: Reg[r]= r
   (r 番目のレジスタには値 r が入っている。)
3. データメモリ DMem: Mem[$address$]= $address$ ($address$ は 4 の倍数。)
   (アドレス a のメモリには値 a が入っている。)

1クロック周期に1枚の図を使い、第3クロックから第7クロック周期まで、5クロック周期分の状態を示せ。第0クロックはリセット状態とする。(第0クロック周期では、アドレス0の命令がフェッチされている図となる。以後、第1クロック周期ではアドレス4の命令のフェッチ、等々。)

各図の上部に、クロック周期の番号と各ステージに入っている命令を、コメントとして書き加えよ。

問題2

アドレス 28 の命令を完了した時、初期状態と異なる値をもつレジスタおよびデータメモリとその内容を、10 進数で書け。

問題3

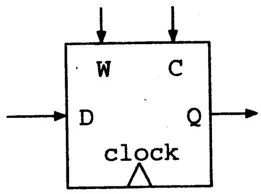
20 番地の命令を

20 sw \$5, 0(\$16)

に置き換えたとき、第7クロックの様子を説明せよ。このとき正しい実行を保証するためにはデータパスをどのように変更したらよいかを述べよ。

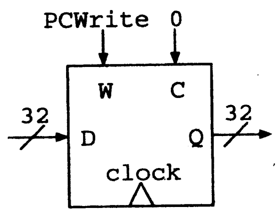
図の回路要素のうち、動作が自明でないものについて、以下に説明する。

1. レジスタ (PC, IF/ID, ID/EX, EX/MEM, MEM/WG)

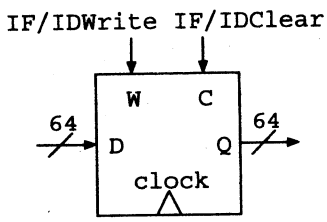


W	C	動作	(clock に同期して)
0	0	記憶	(Q=Q)
1	0	書き込み	(Q=D)
×	1	クリヤ	(Q=0)

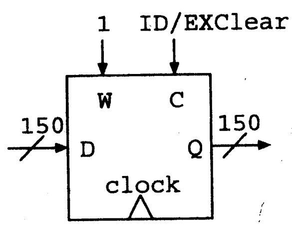
(a) PC



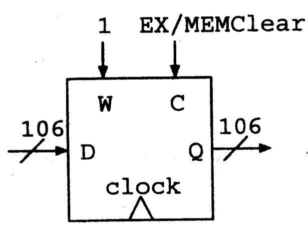
(b) IF/ID



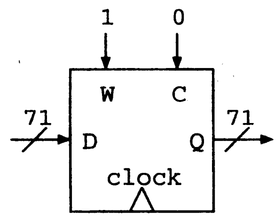
(c) ID/EX



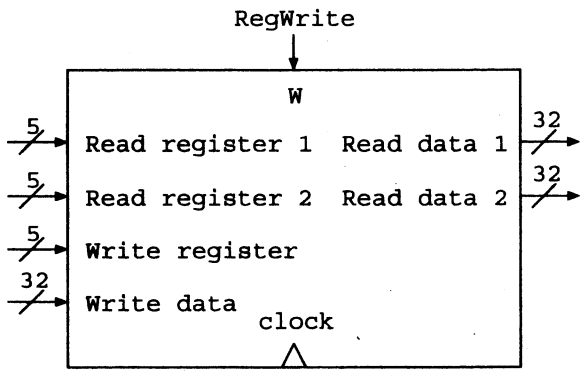
(d) EX/MEM



(e) MEM/WB



2. Reg



- 読み出し: 下の Forwarding の条件が成り立たないとき、RegWrite の値によらず常に

  Read data 1 = Reg[Read register 1]

  Read data 2 = Reg[Read register 2]

- 書き込み: RegWrite=1 のとき、clock に同期して

  Reg[Write register] = Write data

- Forwading:

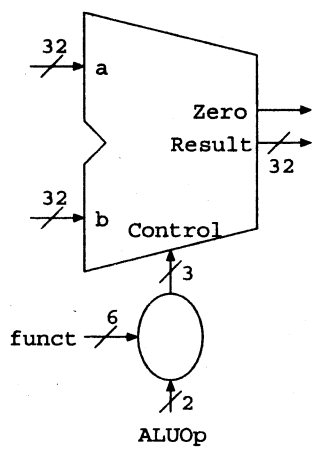
  RegWrite=1 かつ Read register 1 = WriteRegister のとき

  Read data 1 = Write data

  RegWrite=1 かつ Read register 2 = WriteRegister のとき

  Read data 2 = Write data

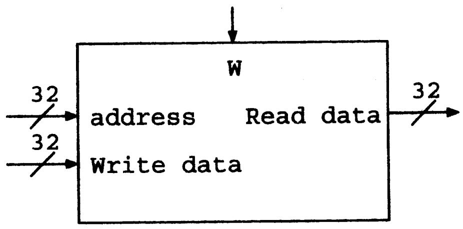
3. ALU



ALUOp	動作
00	Result = a+b
01	Result = a−b
10	Result = a *funct* b
11	使わない

Zero = (Result = 0)

4. Mem(IMem, DMem)



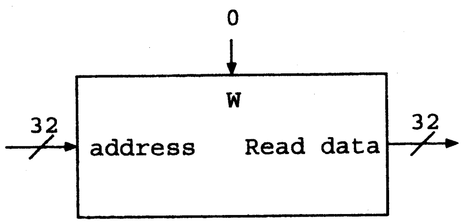
- 読み出し: W の値によらず、常に

  Read data = Mem[address]

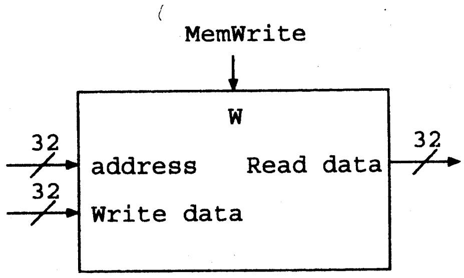
- 書き込み: W が 1 のとき、クロックに同期して[1]

  Mem[address] = Write data

(a) IMem



(b) DMem



[1]実際は、$\neg(W \wedge \neg clock)$ が 0 から 1 に変わるとき。

5. Control

	op	ALUSrc	ALUOp	RegDst	Branch	MemWrite	RegWrite	MemtoReg
add など R-type	0	0	10	1	0	0	1	0
lw	35	1	00	0	0	0	1	1
sw	43	1	00	×	0	1	0	×
beq	4	0	01	×	1	0	0	×

6. Hazard Detection and Forwarding Unit

- ハザードのない場合 (通常動作)

ALUSelA'	ALUSelB'	PCWrite	IF/IDWrite	ID/EXClear
0	0	1	1	0

- EX ハザード

  ID/EX.RegWrite=1 かつ下の場合

  case A: IF/ID.rs=ID/EX.rd かつ ID/EX.RegDst=1
  case A': IF/ID.rs=ID/EX.rt かつ ID/EX.RegDst=0
  case B: IF/ID.rt=ID/EX.rd かつ ID/EX.RegDst=1
  case B': IF/ID.rt=ID/EX.rt かつ ID/EX.RegDst=0

case	ALUSelA'	ALUSelB'	PCWrite	IF/IDWrite	ID/EXClear	
A	1	0	1	1	0	(Forwarding)
A'	×	×	0	0	1	(Stall)
B	0	1	1	1	0	(Forwarding)
B'	×	×	0	0	1	(Stall)
A と B	1	1	1	1	0	(Forwarding)

- MEM ハザード

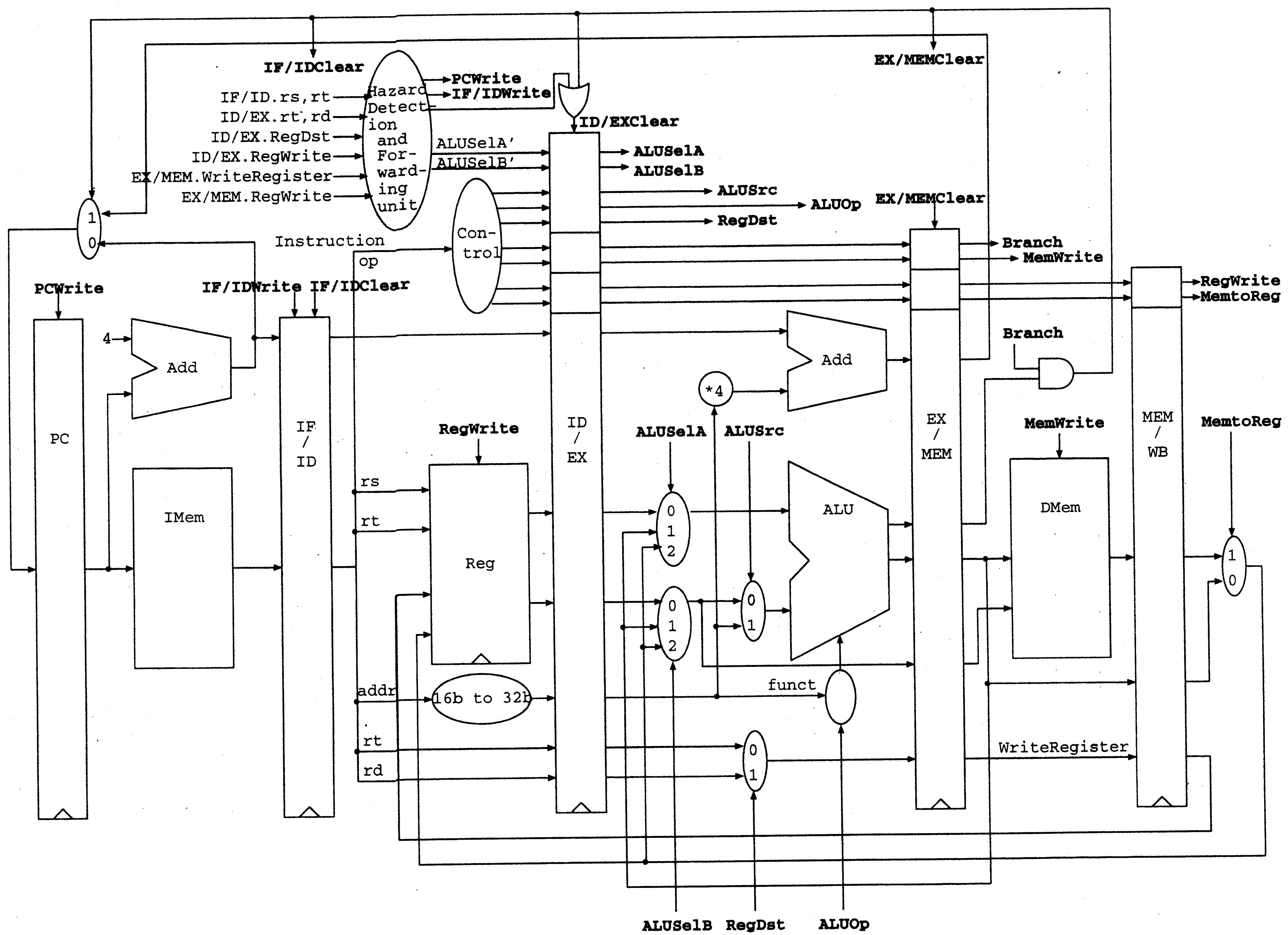
  EX/MEM.RegWrite=1 かつ下の場合

  case C: IF/ID.rs=EX/MEM.WriteRegister かつ EX ハザード case A, A' でない
  case D: IF/ID.rt=EX/MEM.WriteRegister かつ EX ハザード case B, B' でない

case	ALUSelA'	ALUSelB'	PCWrite	IF/IDWrite	ID/EXClear	
C	2	0	1	1	0	(Forwarding)
D	0	2	1	1	0	(Forwarding)
C と D	2	2	1	1	0	(Forwarding)

- EX ハザードと MEM ハザードが同時に起こる場合 (多少省略がある)

case	ALUSelA'	ALUSelB'	PCWrite	IF/IDWrite	ID/EXClear	
A と D	1	2	1	1	0	(Forwarding)
B と C	2	1	1	1	0	(Forwarding)


IF/IDClear
IF/ID.rs,rt
ID/EX.rt,rd
ID/EX.RegDst
ID/EX.RegWrite
EX/MEM.WriteRegister
EX/MEM.RegWrite
Hazard Detection and Forwarding unit
PCWrite
IF/IDWrite
ID/EXClear
ALUSelA'
ALUSelB'
ALUSelA
ALUSelB
ALUSrc
ALUOp
RegDst
EX/MEMClear
EX/MEMClear
Con-trol
Instruction
op
PCWrite
IF/IDWrite
IF/IDClear
4
Add
PC
IMem
IF / ID
rs
rt
addr
rt
rd
RegWrite
Reg
16b to 32b
ID / EX
*4
Add
ALUSelA
ALUSrc
ALU
funct
ALUSelB
RegDst
ALUOp
EX / MEM
Branch
MemWrite
Branch
MemWrite
DMem
WriteRegister
MEM / WB
RegWrite
MemtoReg
MemtoReg

MIPS operands

Name	Example	Comments
32 registers	$0, $1, $2, . . . , $31, Hi, Lo	Fast locations for data. In MIPS, data must be in registers to perform arithmetic. MIPS register $0 always equals 0. Register $1 is reserved for the assembler to handle pseudoinstructions and large constants. Hi and Lo are 32-bit registers containing the results of multiply and divide.
2^{30} memory words	Memory[0],Memory[4], . . . , Memory[4294967292]	Accessed only by data transfer instructions. MIPS uses byte addresses, so sequential words differ by 4. Memory holds data structures, such as arrays, and spilled registers, such as those saved on procedure calls.

MIPS assembly language

Category	Instruction	Example	Meaning	Comments
Arithmetic	add	add $1,$2,$3	$1 = $2 + $3	3 operands; exception possible
	subtract	sub $1,$2,$3	$1 = $2 – $3	3 operands; exception possible
	add immediate	addi $1,$2,100	$1 = $2 + 100	+ constant; exception possible
	add unsigned	addu $1,$2,$3	$1 = $2 + $3	3 operands; no exceptions
	subtract unsigned	subu $1,$2,$3	$1 = $2 – $3	3 operands; no exceptions
	add imm. unsign.	addiu $1,$2,100	$1 = $2 + 100	+ constant; no exceptions
	Move fr. copr. reg.	mfc0 $1,$epc	$1 = $epc	Used to get exception PC
	multiply	mult $2,$3	Hi, Lo = $2 ¥ $3	64-bit signed product in Hi, Lo
	multiply unsigned	multu $2,$3	Hi, Lo = $2 ¥ $3	64-bit unsigned product in Hi, Lo
	divide	div $2,$3	Lo = $2 ÷ $3, Hi = $2 mod $3	Lo = quotient, Hi = remainder
	divide unsigned	divu $2,$3	Lo = $2 ÷ $3, Hi = $2 mod $3	Unsigned quotient and remainder
	Move from Hi	mfhi $1	$1 = Hi	Used to get copy of Hi
	Move from Lo	mflo $1	$1 = Lo	Use to get copy of Lo
Logical	and	and $1,$2,$3	$1 = $2 & $3	3 register operands; logical AND
	or	or $1,$2,$3	$1 = $2 | $3	3 register operands; logical OR
	and immediate	andi $1,$2,100	$1 = $2 & 100	Logical AND register, constant
	or immediate	ori $1,$2,100	$1 = $2 | 100	Logical OR register, constant
	shift left logical	sll $1,$2,10	$1 = $2 << 10	Shift left by constant
	shift right logical	srl $1,$2,10	$1 = $2 >> 10	Shift right by constant
Data transfer	load word	lw $1,100($2)	$1 = Memory[$2+100]	Data from memory to register
	store word	sw $1,100($2)	Memory[$2+100] = $1	Data from register to memory
	load upper imm.	lui $1,100	$1 = 100 x 2^{16}	Loads constant in upper 16 bits
Conditional branch	branch on equal	beq $1,$2,100	if ($1 == $2) go to PC+4+100	Equal test; PC relative branch
	branch on not eq.	bne $1,$2,100	if ($1!= $2) go to PC+4+100	Not equal test; PC relative
	set on less than	slt $1,$2,$3	if ($2 < $3) $1=1; else $1=0	Compare less than; 2's complement
	set less than imm.	slti $1,$2,100	if ($2 < 100) $1=1; else $1=0	Compare < constant; 2's comp.
	set less than uns.	sltu $1,$2,$3	if ($2 < $3) $1=1; else $1=0	Compare less than; natural number
	set l.t. imm. uns.	sltiu $1,$2,100	if ($2 < 100) $1=1; else $1=0	Compare < constant; natural
Unconditional jump	jump	j 10000	go to 10000	Jump to target address
	jump register	jr $31	go to $31	For switch, procedure return
	jump and link	jal 10000	$31 = PC + 4; go to 10000	For procedure call

Main MIPS assembly language instruction set. The floating-point instructions are shown in Figure 4.44 on page 241. Appendix A gives the full MIPS assembly language instruction set.

MIPS machine language

Name	Format	Example 6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	Comments
add	R	0	2	3	1	0	32	add $1,$2,$3
sub	R	0	2	3	1	0	34	sub $1,$2,$3
addi	I	8	2	1	100			addi $1,$2,100
addu	R	0	2	3	1	0	33	addu $1,$2,$3
subu	R	0	2	3	1	0	35	subu $1,$2,$3
addiu	I	9	2	1	100			addiu $1,$2,100
mfc0	R	16	0	1	14	0	0	mfc0 $1,$epc
mult	R	0	2	3	0	0	24	mult $2,$3
multu	R	0	2	3	0	0	25	multu $2,$3
div	R	0	2	3	0	0	26	div $2,$3
divu	R	0	2	3	0	0	27	divu $2,$3
mfhi	R	0	0	0	1	0	16	mfhi $1
mflo	R	0	0	0	1	0	18	mflo $1
and	R	0	2	3	1	0	36	and $1,$2,$3
or	R	0	2	3	1	0	37	or $1,$2,$3
andi	I	12	2	1	100			andi $1,$2,100
ori	I	13	2	1	100			ori $1,$2,100
sll	R	0	0	2	1	10	0	sll $1,$2,10
srl	R	0	0	2	1	10	2	srl $1,$2,10
lw	I	35	2	1	100			lw $1,100($2)
sw	I	43	2	1	100			sw $1,100($2)
lui	I	15	0	1	100			lui $1,100
beq	I	4	1	2	100			beq $1,$2,100
bne	I	5	1	2	100			bne $1,$2,100
slt	R	0	2	3	1	0	42	slt $1,$2,$3
slti	I	10	2	1	100			slti $1,$2,100
sltu	R	0	2	3	1	0	43	sltu $1,$2,$3
sltiu	I	11	2	1	100			sltiu $1,$2,100
j	J	2	10000					j 10000
jr	R	0	31	0	0	0	8	jr $31
jal	J	3	10000					jal 10000

MIPS machine language

Format R	R	op	rs	rt	rd	shamt	funct	Arithmetic instruction format
Format I	I	op	rs	rt	address/immediate			Transfer, branch, imm. format
Format J	J	op	target address					Jump instruction format
Field size		6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	All MIPS instructions 32 bits

Main MIPS machine language. Formats and examples are shown, with values in each field: op and funct fields form the opcode (each 6 bits), rs field gives a source register (5 bits), rt is also normally a source register (5 bits), rd is the destination register (5 bits), and shamt supplies the shift amount (5 bits). The field values are all in decimal. Floating-point machine language instructions are shown in Figure 4.44 on page 241. Appendix A gives the full MIPS machine language.