

CSE 2021 COMPUTER ORGANIZATION

HUGH CHESSER
CSE B 1012U

Example from last time....

Activity 2: Consider the C instruction

$$A[300] = h + A[300]$$

- A. Write the equivalent MIPS code for the above C instruction assuming \$t1 contains the base address of array A (i.e., address of A[0]) and \$s2 contains the value of h
- B. Write the binary machine language code for the result in part A.

```
[00400024] 8d2a04b0 lw $t0, 1200($t1) ; 1: lw $t2, 1200($t1)
[00400028] 024a5820 add $t1, $t0, $s2 ; 2: add $t3, $s2, $t2
[0040002c] ad2b04b0 sw $t1, 1200($t1) ; 3: sw $t3, 1200($t1)
```

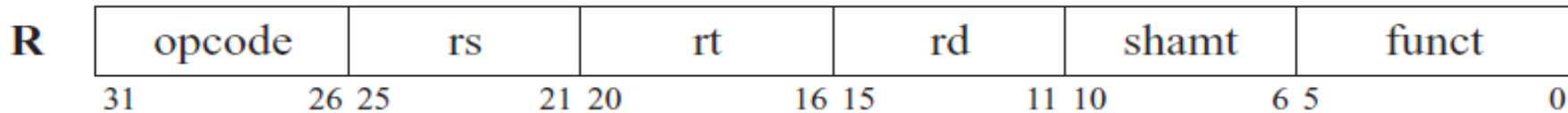
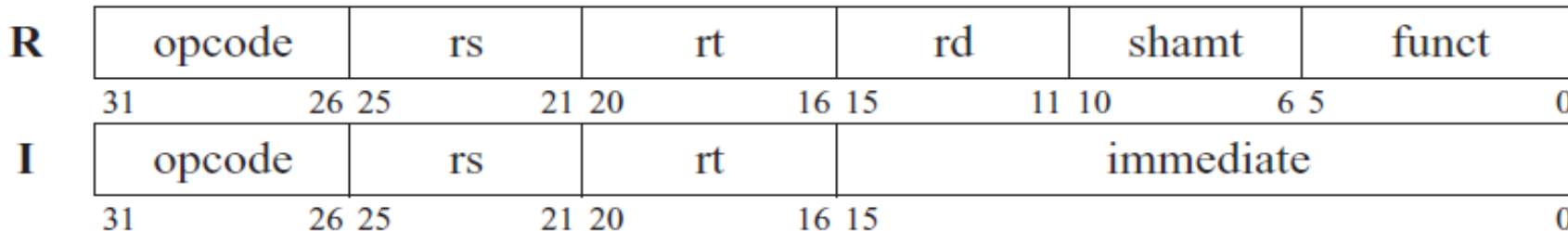
Assembler

NAME	NUMBER
\$zero	0
\$at	1
\$v0-\$v1	2-3
\$a0-\$a3	4-7
\$t0-\$t7	8-15
\$s0-\$s7	16-23
\$t8-\$t9	24-25

CORE INSTRUCTION SET

NAME, MNEMONIC	FOR-MAT	OPERATION (in Verilog)	OPCODE / FUNCT (Hex)
Add	R	$R[rd] = R[rs] + R[rt]$	(1) $0 / 20_{hex}$
Load Word	I	$R[rt] = M[R[rs]+SignExtImm]$	(2) 23_{hex}
Store Word	I	$M[R[rs]+SignExtImm] = R[rt]$	(2) $2b_{hex}$

BASIC INSTRUCTION FORMATS



Add



Agenda for Today

Number representations

MIPS Logical Instructions

- Today: Patterson: Sections 2.4, 2.6
- Wednesday: Sections 2.8 – 2.13 and Appendix B.9

Decimal to Binary Conversion

Any integer, N can be represented as follows:

$$N = \sum_{i=0}^n b_i 2^i \quad \text{where } n = \text{floor}(\log_2 N) = \text{floor}\left(\frac{\log N}{\log 2}\right)$$

If N is odd, then $b_0 = 1$, otherwise is $b_0 = 0$

$$N_e = \begin{cases} N - 1 \times 2^0 = \sum_{i=1}^n b_i 2^i & \text{for N odd, } b_0 = 1 \\ N = \sum_{i=1}^n b_i 2^i & \text{for N even, } b_0 = 0 \end{cases}$$

Divide even integer by 2

$$N_e / 2 = M = \sum_{i=1}^n b_i 2^{i-1}$$

Repeat until left with integer of 0

$$M_e = \begin{cases} M - 1 \times 2^{1-1} = \sum_{i=1}^n b_{i-1} 2^{i-1} & \text{for M odd, } b_1 = 1 \\ M = \sum_{i=1}^n b_i 2^{i-1} & \text{for M even, } b_1 = 0 \end{cases}$$

Example Decimal to Binary

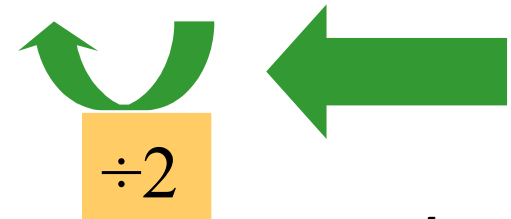
Case 1: Convert 445_{ten} to binary representation.

Answer:

$$n = \text{floor}\left(\frac{\log 445}{\log 2}\right) = 8$$

8	7	6	5	4	3	2	1	0	
1	1	0	1	1	1	1	0	1	=445
0	2	6	12	26	54	110	222	444	

Numbers in this row
MUST be even



Binary to Decimal Conversion

Case 2: Convert the following binary number to decimal representation: 110011001_{two}

$$\begin{aligned} & \underbrace{(1 \times 2^8)}_{256} + \underbrace{(1 \times 2^7)}_{128} + \underbrace{(0 \times 2^6)}_0 + \underbrace{(0 \times 2^5)}_0 + \underbrace{(1 \times 2^4)}_{16} + \underbrace{(1 \times 2^3)}_8 + \underbrace{(0 \times 2^2)}_0 + \underbrace{(0 \times 2^1)}_0 + \underbrace{(1 \times 2^0)}_1 \\ & = 409_{\text{ten}} \end{aligned}$$

Decimal to Hexadecimal Conversion

Any integer, N
can be
represented as
follows

$$N = \sum_{i=0}^n h_i 16^i$$

$$\text{where } n = \text{floor}(\log_{16} N) = \text{floor}\left(\frac{\log N}{\log 16}\right)$$

Divide integer
by 16

$$N/16 = I + \frac{h_0}{16}$$

$$I = \text{floor}(N/16)$$

$$h_0 = 16(N/16 - \text{floor}(N/16))$$

Repeat until left
with integer
less than 16

$$I = \sum_{i=1}^n h_i 16^{i-1} = J + \frac{h}{16} \dots$$

Hexa	Decimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
a	10
b	11
c	12
d	13
e	14
f	15

Hexadecimal Representation (1)

Case 3: Decimal to Hexadecimal Conversion

Example: Convert 445_{ten} into hexadecimal

$$n = \text{floor} \left(\frac{\log 445}{\log 16} \right) = 2$$

2 1 0
 1 b (11) d (13) =445
 0 1 27

$445_{\text{ten}} = 000001bd_{\text{hex}}$ in 1 word

Case 4: Hexadecimal to Decimal Conversion

Example: Convert $000001db_{\text{hex}}$ to decimal

$$\begin{aligned}
 & \underbrace{(1 \times 16^2)}_{256} + \underbrace{(d \times 16^1)}_{208} + \underbrace{(b \times 16^0)}_{11} \\
 & = 475_{\text{ten}}
 \end{aligned}$$

Hexa	Decimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
a	10
b	11
c	12
d	13
e	14
f	15

Binary to Hexadecimal Conversion

Any integer, N can be represented as follows

$$\begin{aligned} N &= \sum_{i=0}^n b_i 2^i = \sum_{k=0}^{\left\lfloor \frac{n}{4} \right\rfloor} (b_{4k+3} 2^3 + b_{4k+2} 2^2 + b_{4k+1} 2^1 + b_{4k}) 2^{4k} \\ &= \sum_{k=0}^m h_k 16^k \\ h_k &= b_{4k+3} 2^3 + b_{4k+2} 2^2 + b_{4k+1} 2^1 + b_{4k} \end{aligned}$$

Each group of 4 binary digits (starting from LSB) can be converted to a hexadecimal digit – represents a shortcut to working out the binary representation

Hexadecimal Representation (2)

Case 5: Hexadecimal to Binary Conversion

Example: Convert $000001bd_{\text{hex}}$ into binary

$$\underbrace{(0)}_{0000_{\text{two}}} + \underbrace{(0)}_{0000_{\text{two}}} + \underbrace{(0)}_{0000_{\text{two}}} + \underbrace{(0)}_{0000_{\text{two}}} + \underbrace{(0)}_{0000_{\text{two}}} + \underbrace{(1)}_{0001_{\text{two}}} + \underbrace{(b)}_{1011_{\text{two}}} + \underbrace{(d)}_{1101_{\text{two}}}$$

Case 6: Binary to Hexadecimal Conversion

Example: Convert $0000\ 0000\ 0000\ 0000\ 0000\ 0001\ 1011\ 1101_{\text{two}}$ to hexadecimal

$$\underbrace{(0000)}_{0_{\text{hex}}} + \underbrace{(0000)}_{0_{\text{hex}}} + \underbrace{(0000)}_{0_{\text{hex}}} + \underbrace{(0000)}_{0_{\text{hex}}} + \underbrace{(0000)}_{0_{\text{hex}}} + \underbrace{(0001)}_{1_{\text{hex}}} + \underbrace{(1011)}_{b_{\text{hex}}} + \underbrace{(1101)}_{d_{\text{hex}}}$$

Activity 1: Convert 1998_{ten} into binary using the hexadecimal shortcut.

2's Complement (1)

1. MIPS uses 2's complement to represent signed numbers
2. In 2's complement, a positive number is represented using a 31-bit binary number
 - Example: $+2_{\text{ten}}$ is represented as $0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0010_{\text{two}}$
or 00000002_{hex}
3. In 2's complement, a negative number $-X_{\text{two}}$ is represented by taking the complement of its magnitude X_{two} plus 1.

— Example: -2_{ten}

Represent the magnitude in binary format

2_{ten} is represented as $0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0010_{\text{two}}$

Take the complement of each digit

The results is $1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1101_{\text{two}}$

Add 1

-2_{ten} is represented as $1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1110_{\text{two}}$

or $ffffffe_{\text{hex}}$

2's Complement (2)

4. The MSB (32nd bit) is the sign bit.
5. To convert a 32-bit number in 2's complement to decimal

$$b_{31} \times (-2^{31}) + \sum_{i=0}^{30} b_i 2^i$$

— Example:

0000 0000 0000 0000 0000 0000 0000 0010_{two} is represented by 2

1111 1111 1111 1111 1111 1111 1111 1110_{two} is represented by

$$(1 \times -2^{31}) + (1 \times 2^{30}) + \dots + (1 \times 2^1) + (0 \times 2^0) = -2$$

Unsigned and Signed Arithmetic

MIPS has a separate format for unsigned and signed integers

1. Unsigned integers

- are saved as 32-bit words
- Example: Smallest unsigned integer is $00000000_{\text{hex}} = 0_{\text{ten}}$
Largest unsigned integer is $ffffff_{\text{hex}} = 4,294,967,295_{\text{ten}}$

2. Signed integers

- are saved as 32-bit words in 2's complement with the MSB reserved for sign
- If MSB = 1, then the number is negative
- If MSB = 0, then the number is positive

— Example:

Smallest signed integer: $1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000_{\text{two}}$
 $= -(2^{31})_{10} = -2,147,483,648_{10}$

Largest signed integer: $0111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111_{\text{two}}$
 $= (2^{31} - 1)_{10} = 2,147,483,647_{10}$

MIPS Logical Instructions

1. Shift logical left (shift right logical – srl):

```
sll $t2,$s0,4           # reg $s0 = reg $s0 << 4 bits
```

2. AND:

```
and $t0,$t1,$t2        # reg $t0 = reg $t1 & reg $t2
```

3. OR (NOR, XOR):

```
or  $t0,$t1,$t2       # reg $t0 = reg $t1 | reg $t2
```

```
nor $t1,$t1,$t3       # reg $t0 = ~ (reg $t1 | reg $t3)
```

MIPS Branch Instructions for *if*(1)

1. Branch if equal to:

```
    beq $s1,$s2,L1           # if $s1 == $s2, go to L1
```

2. Branch if not equal to:

```
    bne $s1,$s2,L2           # if $s1 != $s2, go to L2
```

3. Unconditional jump:

```
    j L3                     # go to L3
```

Example:

```
    if (i == j) go to L1;
```

```
    f = g + h;
```

```
L1:    f = f - i
```

Assume that the five variables f, g, h, i, and j are stored in the registers: \$s0 to \$s4

MIPS Code:

```
    beq $s3,$s4,L1           # go to L1 if i == j
```

```
    add $s0,$s1,$s2          # f = g + h
```

```
L1:    sub $s0,$s0,$s3        # f = f - i
```


MIPS Branch Instructions for *if* (2)

Example: C instructions

```
if (i == j)
    f = g + h;
else
    f = g - h;
```

Assume that the five variables f, g, h, i, and j are stored in the registers: \$s0 to \$s4

MIPS Code:

```
        bne $s3,$s4,L1          # go to L1 if i != j
        add $s0,$s1,$s2        # f = g + h
        j L2                    #
L1:     sub $s0,$s1,$s2        # f = g - h
L2:
```

Activity 3: Write the above code using “branch if equal to” statement?