USE YOUR IMAGINATION™

Blue Book

EXAMINATION BOOK

BOX NO. ____

NAME Yu Cheng yc4845

SUBJECT CSA ECE 6913

CLASS ____

SECTION B

INSTRUCTOR ____

DATE 10/01/2024

11" x 8.5" 8 LEAVES 16 PAGES

ROARING SPRING® PAPER PRODUCTS
ROARING SPRING, PA 16673

for example,
enting 6 to t ... a shift and add instead of an actual ... ulate 9×6 by shifting the bit string

Yu Cheng   yc4845   1

problem 1:

a) CPU ~~Time~~ performance equation

Execution Time = IC * CPI * Tcycle

CPU ~~Time~~ clock cycles = (Instruction number * CPI) / clock frequency

$IC_{P_1} * CPI_{P_1} * Tcycle_{P_1} = IC_{P_2} * CPI_{P_2} * Tcycle_{P_2} = IC_{P_3} \times CPI_{P_3} \times Tcycle_{P_3}$

$\frac{IN_{P_1} * 1}{5 \times 10^3} = \frac{IN_{P_2} \times 2}{5 \times 10^3} = \frac{IN_{P_3} \times 3}{5 \times 10^3}$

∴ The ratio should be $P_1 : P_2 : P_3 = 1 : 2 : 3$.

b)

$\frac{IN_{P_1} \times 1}{CF_1} = \frac{IN_{P_2} \times 2}{CF_2} = \frac{IN_{P_3} \times 3}{CF_3} = \frac{IN_{P_1} \times 1}{5 \times 10^3}$

$\frac{1.5 \times 1}{CF_1} = \frac{1.25 \times 2}{CF_2} = \frac{1 \times 3}{CF_3}$

∴ $CF_1 = 7.5 \times 10^3$ Hz

$CF_2 = 6.025 \times 10^3$ Hz

$CF_3 = 5 \times 10^3$ Hz.

yc4895

Problem 2:

a) ~~add x7 x10~~

~~addi x8, x10, x7~~

~~addi~~

```
slli x10, x10, 16
srli x10, x10, 16
```

~~srli t1, t0, 32~~

~~sll x10, x10, 32~~

b)

```
          mv    t0, x10
          mv    t1, x10
          mv    t2  x11
          li    t3, 0
sum-loop: add   t3, t3, t1
          addi  t1, t1, 1
          blt   t1, t2, sum-loop
          srai  t3  t3  2
          mv    x12, t3
```

for example ... performance enhance... 4C[32i+32]

and add instead of an actu... we can calculate 9×6 by ... that result]. Show the best ... 8-bit unsigned i...

... = g - A[C[16] + B[32]] 5 . 6 . 7

(iii) A[i] = 4B[8i-81] + 4C[32i+32] 4 5

16×× x

4. One possible performance enhancement is to do a shift and add instead of an actual multiplication. [Since 9×6, for example, can be written (2×2×2+1)×6, we can calculate 9×6 by shifting the bit string ...ting 6 to the left three times and then adding 6 to that result]. Show the best way to calculate ... using shifts and adds/subtracts. Assume both in...

fgi 8, 9, 10.

Yu Cheng yc48x5

Problem 3:

i) ~~add x?, x6, x7.~~
~~add~~
~~# Load C[128]~~
~~lw x7 512(x7)~~
~~sll x7, 2.~~
~~add x4 x6 x7~~

li x7, 128
slli x7, x7, 1
add x6, x7, x6
mul x5, x6, x5
sub x8, x9, x5

~~lw x~~
~~sub x8, x9, x4~~

ii) ~~lw x6 128(x6)~~
~~lw x7 64(x7)~~
~~add x24, x6, x7~~
~~slli x24, x24, 2~~
~~add x4, x5, x24~~

li x7, 16
li x6, 32
add x7, x7, x6
mul x5, x7, x5
sub, x8, x9, x5

iii) add x7, x10, x10 # x7 = 8i
addi x7, x7, -81 # x7 = 8i-81
slli x7, x7, 2
add x7, x7, x6 # x7 = address of B[8i-81]

slli x7, x10, 2
add x7, x7, x5 # x7 = address of A[i]
~~lw x24 0(x24)~~

```
add  x7, x12, x12   # x7 = 32i
addi x7, x7, 32     # x7 = 32i+32
slli x7, x7, 2
add  x7, x7, x5     # x5 = address of C[32i+32]

slli x4, x8, 2
add  x4, x4, x5     # x4 = address of A[i]
```

Yu Cheng yc98xx

Problem 4:

~~0x2Ehex =~~

~~0x3Dhex~~ =

0x2Ehex = 46 ⇒ 001 01 1110 in binary

0x3Dex = 61 ⇒ 001 1101 in binary

46x61 = 2806 = 0xAF616

61 = (32+16+8+4+1)

46x61 = 46x(32+16+8+4+1)

~~li t0 0x2E~~

~~li t1 0x3D~~

li t2

slli t3 t0 4

add t2 t2 t3

slli t3 t0 3

add t2 t2 t3

slli t3 t0 2

add t2 t2 t3

li x10 46

slli x11, x10, 5

slli x12, x10, 4

add x11, x11, x12

slli x12, x10, 3

add x11, x11 ~~x12~~

slli x12 x10 2

add x11, x11, x12

add x11, x11, x12

Problem 5:

(1) 142 instructions ⇒ 8 bits (128 < 142 < 256)

32 registers ⇒ 5 bits ; 16 bit immediate

1 reg in, 1 reg out : 8+5+5 = 18 bits ⇒ 24 bits

2 reg in, 1 reg out : 8+5+5+5 = 23 bits ⇒ 24 bits

1 reg in, 1 reg out, 1 imm : 8+5+5+16 = 34 bits ⇒ 40 bits

1 imm in, 1 reg out : 8+16+5 = 29 bits ⇒ 32 bits

(2)

Assuming it has 100 instruction

Since the largest instruction type requires 40-bit instructions

Thus memory used by fixed-length encoding = 40 * 100 = 4000 bits

where as, each instruction type in the variable encoding will use the number of bits from 1:

100 * (0.2 * 24 + 0.3 * 24 + 0.25 * 40 + 0.25 * 32) = 3000 bits

Created with Scanner Pro