Shadi Jiha #40131284 25/01/2020

Winter 2020 Assignment 1

COMP 228 Assignment 1 Solution

Question 1:

a)

	A	B	Time
Serial	$x\%$	$(100-x)\%$	100%
Parallel	$x \times 2$	$\frac{(100-x)}{p}$	z
Max ($P = \infty$)	$x \times 2$	$\frac{(100-x)}{\infty}$	250 times

Solve for x first, and then find p for 75% of 250 and then 90% of 250.

Find x:

$$2x + 0 = 250$$

$$x = 125$$

1- How much processors do we need to speed up by 187.5 (75% of 250)?

$$2x + \frac{(100-x)}{p} = 187.5$$

$$\frac{-25}{p} = -62.5$$

$$\frac{25}{62.5} = p$$

$p = 0.4 \approx 1$ processors

2- How much processors do we need to speed up by 225 (90% of 250)?

$$2x + \frac{100-x}{p} = 225$$

$$250 + -\frac{25}{p} = 225$$

$1 = p$ processors

b)

	A	B	Time
Serial	x	y	100
Parallel	x	$\frac{(100-x)}{5\frac{2}{3}}$	$\frac{s}{2}$
Max ($P=\infty$)	x	$\frac{(100-x)}{\infty}$	S seconds

Find the value of s:

$$x+\frac{100-x}{5\frac{2}{3}}=\frac{s}{2}$$

$$2x+\frac{200-2x}{5\frac{2}{3}}=s$$

Find the value of x:

$$x+\frac{100-x}{\infty}=2x+\frac{200-2x}{5\frac{2}{3}}$$

$$x=2x+\frac{200-2x}{5\frac{2}{3}}$$

$$-x=\frac{200}{\frac{17}{3}}-\frac{2x}{\frac{17}{3}}$$

$$-x+\frac{6x}{17}=\frac{600}{17}$$

$$-\frac{11}{17}x=\frac{600}{17}$$

$$x=-54\frac{6}{11}$$

Find the value of y:

$$x+y=100$$

$$53+y=100$$

$$y=47$$

Solution: $x=53, y=47$

Question 2:

We have:

$$\frac{12a+b}{2a+0.1b} \geq 6\frac{2}{3} \text{ and } a+b=500$$

$$b = 500 - a$$

$$\rightarrow \frac{12a + 500 - a}{2a + 0.1(500 - a)} \geq 6\frac{2}{3}$$

$$\frac{11a + 500}{1.9a + 50} \geq 6\frac{2}{3}$$

$$a = 100$$

Find b:

$$b = 500 - a = 400$$

Power usage:

$$\alpha: 100(12) = 1200\frac{GFs}{s} = 1.2\frac{TFs}{s}$$

$$100(2) = 200\,Watt$$

$$\beta: 400(1) = 400\frac{GFs}{s} = 0.4\frac{TFs}{s}$$

$$400(0.1) = 40\,Watt$$

SOLUTION:

a) We need 400 beta and 100 alpha
b) Total performance: 1.6 TFs/s and power dissipation 240 $Watt$

Question 3:

Consider f6 and f7 as registers. The Code will be:

```
loop:   l.d     f6, 0(r1)   // Load the content of the correspoding index in 'a' to register
        mul.d   f6, f6, f2  // Multiply the data in even index 'a' by f2
        s.d     f6, 0(r2)   // Store the result in the addres of the correspoding index of 'b'
        addi    r1, r1, 8   // Add 8 bytes to address r1 of 'a' (This is like doing i++ in a loop)
        addi    r2, r2, 8   // Add 8 bytes to address r2 of 'b'

        l.d     f7, 0(r1)   // Load the content of the correspoding index in 'a' to register
        mul.d   f7, f7, f0  // Multiply the data in odd index 'a' by f0
        s.d     f7, 0(r3)   // Store the result in the addres of the correspoding index of 'c'
        addi    r1, r1, 8   // Add 8 bytes to address r1 of 'a'
        subi    r3, r3, 8   // Substract 8 bytes to address r3 of 'c' because we want in reverse order
        bne     r1, r4, loop    // While we havn't reached the end of 'a' keep going
```

Question 4:

The datastore (also called pipeline) is the unit that executes the instructions. All of fetch unit and branch unit require access to it because the need to know which instruction to execute next in a program.

Question 5:

a) 8 GBs = $8 \times 2^{30} = 858993459\ bytes \times 8 \frac{bits}{byte} = 68719476736\ bits$
b) 35 TBs = $35 \times 20^{40} = 3.8482907 \cdot 10^{13} bytes \times 8 = 3.0786326 \cdot 10^{14} bits$
c) 1.05 EBs = $1.05 \times 20^{60} = 1.2105676 \cdot 10^{18} \times 8 = 9.6845406 \cdot 10^{18} bits$
d) 0.5 EBs = $0.5 \times 20^{60} = 5.7646075 \cdot 10^{17} \times 8 = 4.611686 \cdot 10^{18} bits$

Question 6:

a) $\frac{GFs}{s} = \min\,(AI \cdot BW, peak\ GFs/s)$

$$AI \cdot BW > peak\ GFs/s$$
$$AI \cdot 162 > 102.4$$
$$AI > 0.632$$

So AI = 0.632 flops/byte

b) At peak:

$$\frac{GFs}{s} = \min(0.632 \cdot 162, 102.4)$$
$$= 102.4$$
$$\rightarrow \frac{102.4}{95} = 1.0778 \frac{\frac{GFs}{s}}{W}$$

c) Double the BW:

$$AI \cdot 324 > 102.4$$
$$AI > 0.316049$$

AI = 0.316 flops/byte

d) Power efficiency:

$$\frac{324}{95} = 3.4105 \frac{\frac{GFs}{s}}{W}$$

e) Depending on his need, if he wants to process more data and power usage isn't an issue, the costumer can ignore the power efficiency.