भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Assignment - 1

CS2323

Kartik Srinivas - ES20BTECH11015
September 18, 2022

Index

1 Problem 1 **2**
1.1 Solution . . . 2
1.2 Julia code . . . 2

2 Problem 2 **3**
2.1 Solution . . . 3
2.2 Julia Code . . . 3

3 Problem 3 **4**
3.1 Solution . . . 4
3.2 Julia code . . . 4

1 Problem 1

1.1 Solution

$$T = IC \times CPI \times T_c$$

Substituting the values we get

$$T_1 = 40\ ns$$
$$T_2 = 51\ ns$$

So the first one is faster and he is faster by **11 nanoseconds** In terms of the factors of speed , the first one actually looses

$$F = \frac{CPI_1 \times T_{c1}}{CPI_2 \times T_{c2}}$$

$$\boxed{F = 1.1764}$$

In general T1 would be slower but since the number of instructions for T1 are less, it becomes faster in this case. For the new CPU to have the same speed with a smaller CPI of 1.5 , it will take more time, or it will have lower frequency

$$T_{c3} = \frac{T_1}{IC_1 \times CPI_3} \tag{1}$$

$$\tag{2}$$

$$\boxed{f_{c3} = 1/T_{c3} = 0.75Ghz} \tag{3}$$

1.2 Julia code

```
IC_1  = 2e10 ; IC_2 = 3e10
f_1 = 1e9 ; f_2 = 2e9
cpi_1 = 2 ; cpi_2 = 3.4
t1  = IC_1 * cpi_1 /f_1
t2 = IC_2 * cpi_2 /f_2
println("t1 = ", t1 , " t2 = ", t2)
t_diff = (t1 - t2)* 1e-9
function calc_time(IC, CPI, time)
    return IC * CPI * time
ecpi_3 = 1.5
t3  = t1/(cpi_3 * IC_1)
print(1/t3)
```

2 Problem 2

2.1 Solution

The average cpi needs to be calculated

$$CPI_{av} = \frac{\sum_i CPI_i \times n_i}{\sum_i n_i} \tag{4}$$

This value for say (x number of instructions) is equal to the following

$$CPI^1_{av} = 1.8$$
$$CPI^2_{av} = 2.2$$

Now to calculate the time for them usually using the code expansion factor for compiler 1 = 1.2,

$$T_1 = CPI^1_{av} \times 1.2x = 2.16x < 2.20x = CPI^2_{av} \times x = T_2 \tag{5}$$

$$\boxed{T_1 < T_2} \tag{6}$$

Therefore the **first one is usually faster**, despite having more instructions. Now to calculate the value of Time for 1 in the case of 1.2e7 instructions :-

$$\boxed{T_1 = 2.16 \times 1e7 \times \frac{1}{f} = 0.108\ s} \tag{7}$$

2.2 Julia Code

```
using LinearAlgebra
cpi = [1, 3, 4]
function calc_time(IC, CPI, time)
    return IC * CPI * time
end
c1 = [0.7 , 0.1, 0.2]  ; c2 = [0.5 , 0.3, 0.2]
av_1 = dot(c1, cpi)
av_2 = dot(c2, cpi)
println("average cpi of first = ", av_1)
println("average cpi of second = ", av_2)
IC_2 = 1e7
factor = 1.2
IC_1 = IC_2 * factor
freq = 2e9
t1  = calc_time(IC_1, av_1, 1/freq)
println("Time taken by CPU_1 = ", t1)
```

3 Problem 3

3.1 Solution

Yield is calculable

$$yield = \frac{1}{(1 + \frac{defects \times die-area}{2})^2} = 0.949219$$

Dies per wafer is the floor of the ratio of the areas

$$DPW = floor(\frac{A_w = \pi * d^2/4}{A_d = l \times b}) = 192.0 \ \ dies$$

Cost per die is total cost by the useful wafers

$$CPD = \frac{CPW = 200}{DPW \times yield} = 1.097\$ \tag{8}$$

3.2 Julia code

```
d = 18e-2
a_d = 11e-3 * 12e-3
defect_rate = 0.04*1e4
cost = 200
a_w =  * (d^2)/4
n_d = floor(a_w/(a_d))
yield = 1/((1 + ((defect_rate*a_d)/2)))^2
cost_per_die = cost/(n_d * yield)
```
