**Register Reuse**

Part 1 & 2

dgemm0

𝐋𝐨𝐚𝐝 𝐚𝐧𝐝 𝐬𝐭𝐨𝐫𝐞 𝐭𝐢𝐦𝐞 = 𝟒 ∗ 𝒏𝟑 ∗ 𝟏𝟎𝟎/ 𝟐𝐆𝐡𝐳 = 𝟒 ∗ (𝟏𝟎𝟎𝟎)𝟑 /𝟐𝑮𝒉𝒛 ⁄ = 𝟐𝟎𝟎𝐬（ A, B, C matrices load and C store）this version is not using registers, so the load and store time is just the wasting time.

𝐎𝐩𝐞𝐫𝐚𝐭𝐢𝐧𝐠 𝐭𝐢𝐦𝐞 is 2/4 ∗ 𝒏𝟑 ∗ /𝟐𝐆𝐡𝐳 = 0.25 seconds total time is 200.25 s

(200.25 - 0.25) / 200.25 = 0.9988 so 99.88% time is wasted.

dgemm1

this version has two load operations( a && b loading) in (i,j) loop and in k loop (a && b loading)

so the total loading time is (2n3 + 2n2)\*(Cycle Time)\*(Additional Delay)

= (2\*1000\*1000 + 2\*1000\*1000)\*0.5\*(10-9)\*100

= 100.1 seconds

𝐎𝐩𝐞𝐫𝐚𝐭𝐢𝐧𝐠 𝐭𝐢𝐦𝐞 is same as dge0 2/4 ∗ 𝒏𝟑 ∗ /𝟐𝐆𝐡𝐳 = 0.25 seconds

So the total time is = 100.1 + 0.25 = 100.35 seconds

(100.35 - 0.25) / 100.35 = 0.9975

= 99.75% time is wasted

Part 3

We can use 3\*3 size to improve, however 3\*3 will have like 27 registers to use, so I put all C into registers and 3 registers for B & A each.

**Cache Reuse**

n=10000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Element Cache Miss | | | Cache Read | | | Miss Rate |
| A | B | C | A | B | C |
| ijk  jik | n for a[,k] | k%10==0 | n | 1 | n^3 | n^3 | n^2 | about 55% |
| Ikj  kij | 1 | n for b[,j] | j%10==0 | n for c[,j] | j%10==0 | n^2 | n^3 | n^3 | about 10% |
| Jki  kji | n | 1 | n | n^3 | n^2 | n^3 | 100% |

n=10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Element Cache Miss | | | Cache Read | | | Miss Rate |
| A | B | C | A | B | C |
| Ijk  jik | 1 if k%10==0 | 1 if k%10==0 | 1 if k%10==0 | n^3 | n^3 | n^2 | 1.43% |
| Ikj  kij | 1 if j%10==0 | 1 if j%10==0 | 1 if j%10==0 | n^2 | n^3 | n^3 |
| Jki  kji | 1 if i%10==0 | 1 if i%10==0 | 1 if i%10==0 | n^3 | n^2 | n^3 |