

Στο αρχείο `matmul-normal.c` δεσμεύουμε δυναμικά τρεις δισδιάστατους πίνακες $N \times N$ και τους αναθέτουμε αρχικές τιμές. Στη συνέχεια κάνουμε πολλαπλασιασμό πινάκων ανάμεσα στον A και στον B και αποθηκεύουμε το αποτέλεσμα στον πίνακα C . Ο πολλαπλασιασμός των πινάκων δίνεται από τον παρακάτω τύπο :

$$[\mathbf{AB}]_{i,j} = A_{i,1}B_{1,j} + A_{i,2}B_{2,j} + \dots + A_{i,n}B_{n,j} = \sum_{r=1}^n A_{i,r}B_{r,j}.$$

Ο πολλαπλασιασμός γίνεται τμηματικά κάθε φορά για κάθε στοιχείο του Πινάκα C μέσα στο τρίτο `for loop`.

Σε αντίθεση με το αρχείο `matmul-sse.c` όπου σε κάθε `loop` μέσα στο τρίτο `for` γίνονται τέσσερις πράξεις παράλληλα. Και τα δυο αρχεία παράγουν το ίδιο αποτέλεσμα άλλα με διαφορετική υλοποίηση.

N	With out SSE	With SSE
4	mflops: inf	mflops: inf
40	mflops: 24.947533	mflops: 84.947929
400	mflops: 1.741784	mflops: 6.753365
4000	mflops: 0.151587	mflops: 0.054111

Όπως βλέπουμε από τον πίνακα ο κώδικας με το SSE αποδίδει πολύ καλύτερα έχοντας πόλη καλύτερη απόδοση. Αυτό οφείλετε στο γεγονός ότι γίνονται τέσσερις πράξεις σε κάθε επανάληψη.

Αυτό που δεν μπορώ να εξηγήσω είναι γιατί έχει χαμηλότερη απόδοση το SSE με $N = 4000$. Πιστεύω ότι οφείλετε στο γεγονός ότι έτρεξα τα πειράματα πάνω στην πλατφόρμα Cloud9.

Δεν χρησιμοποίησα έλεγχο στο τέλος διότι στην αρχικοποίηση με σταθερές τιμές μου έκανε απαλοιφή τα `for loop`.

Η version του compiler που χρησιμοποίησα είναι : gcc version 4.8.4 (Ubuntu 4.8.4-2ubuntu1~14.04.3)

Η αρχιτεκτονική του επεξεργαστή είναι η παρακάτω:

```
processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 63
model name    : Intel(R) Xeon(R) CPU @ 2.30GHz
stepping      : 0
microcode     : 0x1
cpu MHz       : 2300.000
cache size    : 46080 KB
```

physical id : 0
siblings : 8
core id : 0
cpu cores : 4
apicid : 0
initial apicid : 0
fpu : yes
fpu_exception : yes
cpuid level : 13
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx
fxsr sse sse2 ss ht syscall nx pdpe1gb rdtscp lm constant_tsc rep_good nopl xtopology nonstop_tsc pni
pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4
_2 x2apic movbe popcnt aes xsave avx f16c rdrand hypervisor lahf_lm abm invpcid_single ssbd ibrs
ibpb stibp kaiser fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid xsaveopt arat
arch_capabilities
bugs : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf
bogomips : 4600.00
clflush size : 64
cache_alignment : 64
address sizes : 46 bits physical, 48 bits virtual
power management:

Πηγές : <https://software.intel.com/sites/landingpage/IntrinsicsGuide/#expand=3966&techs=SSE>