1. Pendahuluan

1.1. Latar Belakang

Masalah pada CC-NUMA salah satunya adalah performansi sistem CC-NUMA itu sendiri serta masalah yang berkaitan dengan *cache* dan memori. Definisi CC-NUMA sendiri adalah Sistem NUMA yang telah mendukung *cache coherence* [11]. *Cache coherence* pada sistem CC-NUMA dilakukan dengan mengimplementasikan protokol koherensi. Protokol koherensi cache dapat menggunakan kebijakan yang berbeda untuk menetapkan hak-hak (membaca atau menulis) dan tugas (memasok data ke *cache* lain) bahwa node tertentu memiliki lebih dari setiap blok[11]. Pada multiprosesor terutama CC-NUMA dengan *shared-memory* dan *cache* yang terpisah untuk tiap prosesor, sangat mungkin untuk prosesor satu dengan yang lain merujuk ke satu lokasi *cache block* yang sama namun mempunyai informasi yang berbeda. Hal ini menyatakan bahwa *cache* belum koheren, dan disinilah peran protokol *cache coherence*. Salah satu protocol *cache coherence* adalah MOESI. MOESI adalah pengembangan dari protokol MESI yang mempunyai empat *state*[11]. MOESI dapat digabungkan dengan bersamaToken *coherence.* Token akan berperan untuk menspesifikasikan setiap blok memori dengan nomor tetap, kemudian dikombinasikan dalam setiap *state* pada MOESI[4]. Dalam Gem5 *simulator* protokol yang mendukung kombinasi MOESI dan Token adalah MOESI CMP Token.

Permasalahan lain yang berkaitan dengan CC-NUMA adalah performansi *cache* itu sendiri. Performansi *cache* secara umum dapat dilihat dari *cache miss rate*. *Cache miss rate* merupakan perbandingan antara *cache miss* dan total pengaksesan ke *cache* yang dilakukan oleh prosesor. *Cache miss* adalah kegagalan untuk membaca atau menulis data pada *cache*. *Cache miss* mempengaruhi performasi *cache* yang dapat membuat kinerja *cache* menjadi lebih lambat karena harus mengakses level *cache* atau memori yang lebih lambat [12]. Beberapa hal yang dapat mempengaruhi *cache miss rate* adalah ukuran *cache block,* jumlah *core* cpu, *associativity,* dan *ukuran cache.* Ukuran *cache block* menjadi faktor yang penting ketika dibandingkan dengan ukuran jumlah *core* cpu, *associativity*, dan *ukuran cache* yang sama, dikarenakan *cache block* merupakan bagian dari *cache* yang berisi lokasi untuk menyimpan informasi yang akan dirujuk oleh prosesor.

Dikarenakan masalah *cache coherence* telah dapat diselesaikan pada sistem CC-NUMA, masalah yang perlu diteliti adalah performansi *cache* dan memori terhadap pada CC-NUMA. Dengan penerapan protokol MOESI CMP Token sebagai protokol *cache coherence* pada CC-NUMA , dan salah satu faktor yang mempengaruhi performansi *cache* dapat ditentukan dengan ukuran *cache block*-nya. Maka dalam tugas akhir ini akan meneliti pengaruh penambahan ukuran *cache block* pada sistem CC-NUMA yang menerapkan protokol MOESI CMP Token sebagai protokol *cache coherence*.

1.2. Perumusan Masalah

Berdasarkan latar belakang, adapun perumusan masalah dalam Tugas Akhir ini antara lain :

1. Bagaimana performansi sistem CC-NUMA dengan penambahan ukuran *cache block*?
2. Bagaimana performansi *cache* dan memori pada sistem CC-NUMA dengan penambahan *cache block*?

Batasan masalah pada tugas akhir ini adalah sebagai berikut :

1. Arsitektur sistem yang digunakan adalah x86.
2. Tidak menangani perubahan parameter berkaitan dengan penambahan jumlah prosesor, *associativity* dan ukuran *cache*.
3. Tidak menangani konfigurasi *cache block* secara terpisah pada L1 dan L2 *cache*.
4. Simulator yang digunakan Gem5
5. Ukuran *cache block* yang digunakan adalah 16, 32, 64 dan 128 *bytes.*

1.3. Tujuan

Tujuan tugas akhir ini adalah :

1. Menganalisis performansi sistem CC-NUMA dengan penambahan ukuran *cache block.*
2. Menganalisis performansi *cache* dan memori pada sistem CC-NUMA dengan penambahan ukuran *cache block.*

1.4. Metodologi Penyelesaian Masalah

Metodologi yang digunakan dalam menyelesaikan masalah pada tugas akhir ini adalah dengan menggunkan langkah-langkah sebagai berikut :

1. Studi literatur

   Tahap ini merupakan tahap mengumpulkan berbagai teori maupun konsep yang mendukung tugas akhir diantaranya meliputi masalah sistem NUMA dan *cache* coherence protocol.

2. Perancangan Sistem

   Perancangan sistem pada simulasi ini akan dilakukan dengan mengidentifikasi sistem CC-NUMA yang akan dijadikan objek penelitian adan protocol *cache coherence* yang akan diimplementasikan.

3. Implementasi Sistem

   Pembangunan simulasi ini akan menerapkan protokol MOESI CMP Token pada CC-NUMA dengan menggunakan simulator Gem5 yang berjalan di sistem operasi Ubuntu 12.04 64bit.

4. Pengujian dan Analisa Hasil

   Tahap ini merupakan tahap untuk menguji sistem yang telah menerapkan protokol MOESI CMP Token pada CC-NUMA dengan untuk mengukur koherensi *cache* dan performansi memori pada arsitektur NUMA tersebut.

5. Penyusuanan Laporan Tugas Akhir

   Tahap ini merupakan tahap untuk membuat kesimpulan dari hasil analisis dan pegujian yang dilakukan. Kemudian dilakukan dokumentasi semua tahapan proses diatas berupa laporan yang berisi tentang dasar teori dan hasil tugas akhir ini ke dalam sebuah buku tugas akhir.