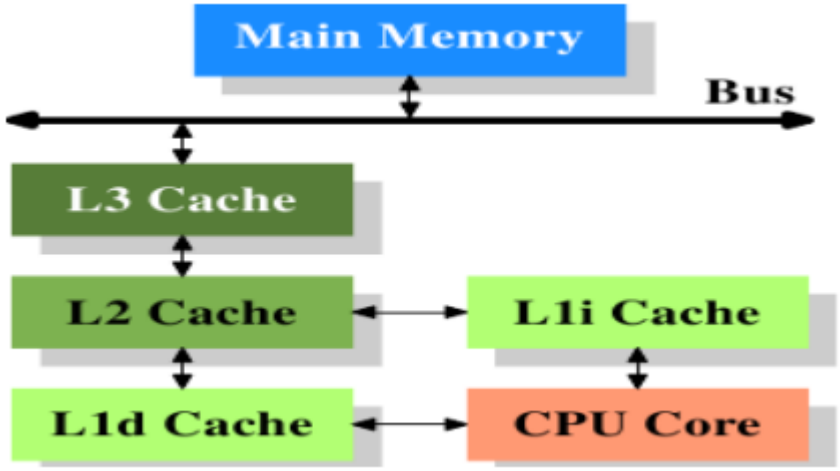
Isi Jurnal

Memori cache, kadang-kadang disebut penyimpanan cache atau cache RAM, pada dasarnya adalah bagian dari memori yang terdiri dari RAM statis (SRAM) berkecepatan tinggi, bukan RAM dinamis (DRAM) untuk memori utama. Cache memori efisien karena sebagian besar program mengakses data atau instruksi yang sama berulang-ulang. Lokasi utama memori cache dalam arsitektur komputer adalah antara memori utama dan inti CPU, sebagai konfigurasi memori cache yang sederhana dan minimal.

Memori cache juga dikembangkan ke berbagai level untuk meningkatkan kinerja prosesor secara keseluruhan. Level-level ini mencapai beberapa pengembangan untuk kinerja keseluruhan. Tingkat sederhana yang dikembangkan digambarkan pada gambar 1 untuk menggambarkan tiga tingkat memori cache dan memperkenalkan nomenklatur. L1d mendefinisikan cache data level 1, sedangkan, L1i mengekspresikan cache instruksi level 1, dll.

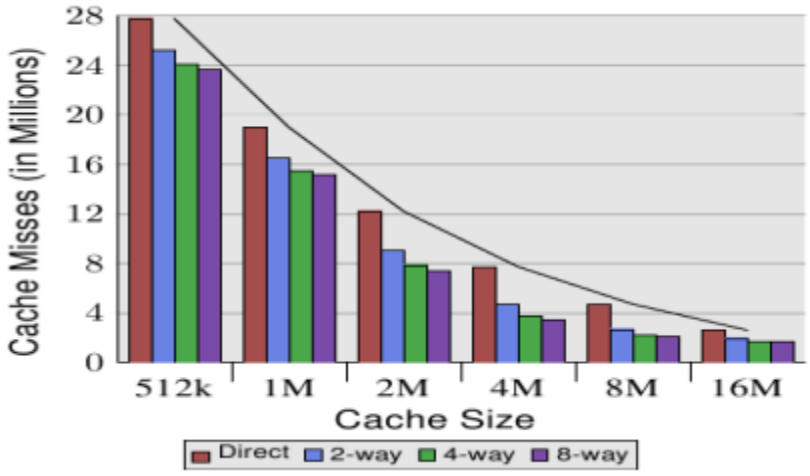


Gambar 1 – Prosesor dengan Level 3 Cache

Operasi Cache di Level Tinggi

Ketika konten memori dibutuhkan oleh prosesor, seluruh baris cache dimuat ke dalam L1d. Alamat memori untuk setiap baris cache dihitung dengan menutupi nilai alamat menurut ukuran baris cache. Untuk saluran cache 64 byte, ini berarti 6 bit paling rendah dinolkan. Bit yang dibuang digunakan sebagai offset ke saluran cache. Bit yang tersisa dalam beberapa kasus

digunakan untuk menemukan baris dalam cache dan sebagai tag. Dalam praktiknya, nilai alamat dibagi menjadi tiga bagian.



Gambar 2 – Ukuran Cache

Gambar 2 membuat data lebih mudah dipahami. Ini menunjukkan data untuk ukuran saluran cache tetap 32 byte. Melihat angka untuk ukuran cache yang diberikan, dapat dicatat bahwa asosiatif memang dapat membantu mengurangi jumlah cache yang hilang secara signifikan. Untuk cache 8MB yang beralih dari pemetaan langsung ke set 2-arah, cache asosiatif menghemat hampir 44% cache yang hilang. Prosesor dapat menyimpan lebih banyak set kerja dalam cache dengan cache asosiatif yang ditetapkan dibandingkan dengan cache yang dipetakan langsung.

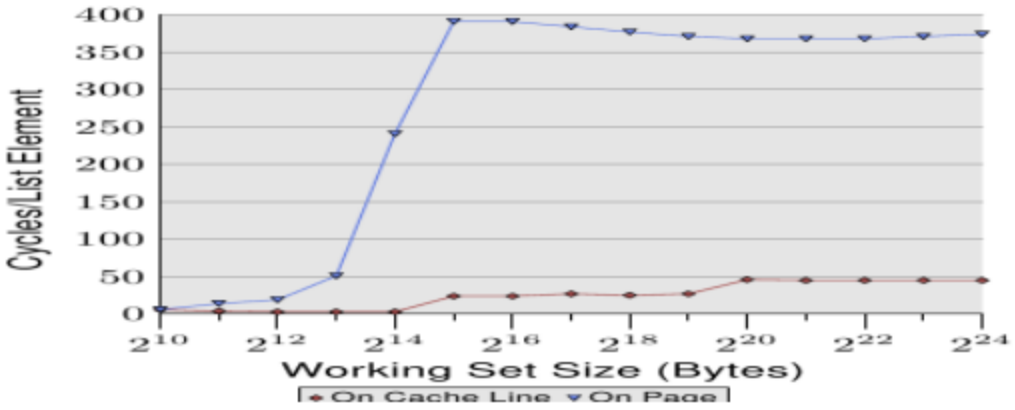
Metodologi

Metodologi untuk mencapai tujuan artikel ini didasarkan pada beberapa langkah:

1. Menyatakan transisi status dari beberapa proses berbeda sebagai program perangkat lunak untuk mensimulasikan pemrosesan dalam bahasa C.
2. Menggunakan keyboard sebagai perangkat keras input.
3. Menggunakan antarmuka port paralel, sambungkan port *output* ke PC.
4. Menggunakan IC SN74245 sebagai pelindung antara keyboard dan antarmuka port paralel.

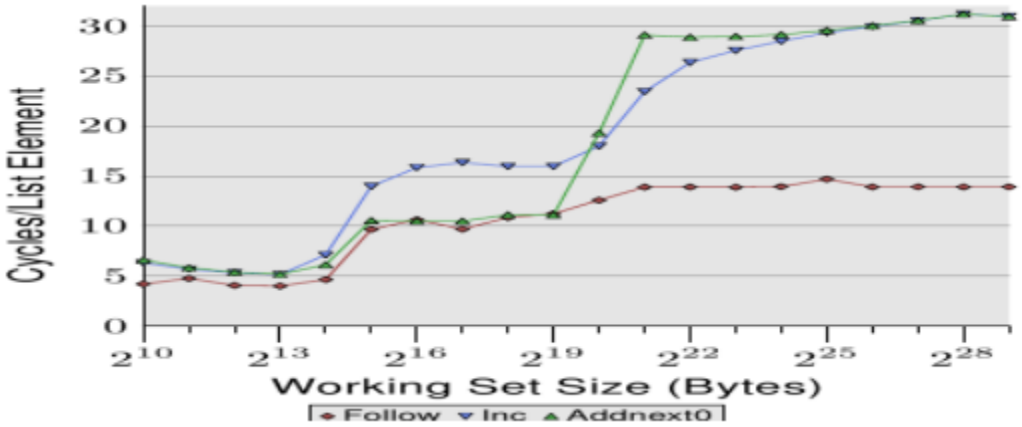
Hasil Penelitian

Proses simulasi diatur dan dilakukan sesuai dengan parameter normal dan ukuran RAM untuk mengevaluasi pengaruh berbagai tingkat cache pada kecepatan prosesor dan kinerja komputer secara keseluruhan. Karena keterbatasan RAM yang tersedia, ukuran set kerja harus dibatasi hingga 224 byte yang membutuhkan 1GB untuk menempatkan objek pada halaman terpisah. Hasil simulasi yang dinilai ditunjukkan pada Gambar 3.



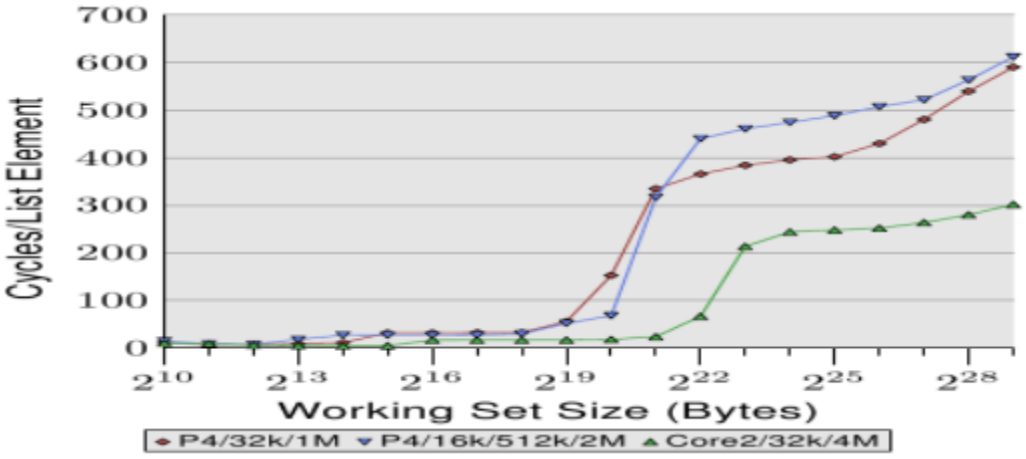
Gambar 3

Untuk evaluasi mendalam, beberapa detail lebih lanjut dari implementasi prefetch dengan melihat data uji coba di mana elemen daftar dimodifikasi. Gambar 4 menunjukkan tiga garis.



Gambar 4

Satu aspek terakhir dari penanganan cache yang berurutan dan efisien adalah ukuran cache. Ini harus jelas tetapi masih harus ditunjukkan. Gambar 5 menunjukkan waktu untuk benchmark Increment dengan elemen 128-byte (NPAD=15 pada mesin 64-bit). Di bawah ini adalah pengukuran yang menunjukkan peningkatan kinerja prosesor dari tiga mesin yang berbeda.



Gambar 5