Jurnal Riset Sistem Informasi dan Aplikasi Komputer (JRSIKOM)
Vol. 1, No. 4, Desember 2025
SK.ISSN: 3090076X/II.7.4/SK.ISSN/05/2025 e-ISSN: 3090-076X

PENGARUH TEKNOLOGI PIPELINE TERHADAP PERFORMA PROSESOR SUPERSKALAR DALAM SIKLUS DATA

Suci Maharani[1], Elinnah Qori[2], Febri Irwansyah[3], Indra Gunawan, M.Kom[4]

[1,2,3,4]STIKOM Tunas Bangsa Pematangsiantar

Email: [1]maharanisucii262@gmail.com,[2]elinnahqori5@gmail.com, [3]febriirwansyah12@gmail.com, [4]indra@amiktunasbangsa.ac.id

Abstrak

Penelitian ini membahas pengaruh penerapan teknologi *pipeline* terhadap performa prosesor *superskalar* dalam siklus data dengan meninjau hubungan antara kedalaman *pipeline*, lebar *issue width*, dan mekanisme *branch prediction* terhadap efisiensi eksekusi instruksi. Penelitian dilakukan menggunakan pendekatan simulasi kuantitatif berbasis perangkat lunak arsitektur prosesor seperti *SimpleScalar* dan *gem5*, dengan beberapa variasi konfigurasi tahap *pipeline* (5, 10, dan 15) dan lebar *issue* (1, 2, dan 4). Hasil simulasi menunjukkan bahwa peningkatan kedalaman *pipeline* dan perluasan *issue width* secara signifikan meningkatkan nilai *Instruction per Cycle (IPC)* serta efisiensi prosesor, di mana kombinasi *pipeline* 15 tahap dengan *issue width* empat dan *hybrid branch prediction* menghasilkan IPC tertinggi sebesar 2,3 dengan efisiensi 85%. Namun, semakin dalam *pipeline*, kompleksitas kontrol dan potensi *hazard* juga meningkat, sehingga keseimbangan antara kedalaman *pipeline*, akurasi *branch prediction*, dan efisiensi daya menjadi faktor penting dalam desain prosesor modern. Penelitian ini menegaskan bahwa penerapan *pipeline* yang optimal mampu meningkatkan *throughput* dan stabilitas kinerja prosesor *superskalar* tanpa mengorbankan efisiensi energi.

Kata kunci: *Pipeline, Superskalar, Branch Prediction, IPC, Hazard.*

THE INFLUENCE OF PIPELINE TECHNOLOGY ON THE PERFORMANCE OF SUPERSCALAR PROCESSORS IN DATA CYCLES

Abstract

This study examines the influence of pipeline technology on the performance of superscalar processors in data cycles by analyzing the relationship between pipeline depth, issue width, and branch prediction mechanisms on instruction execution efficiency. The research applies a quantitative simulation approach using processor architecture tools such as SimpleScalar and gem5, with several configuration variations in pipeline stages (5, 10, and 15) and issue widths (1, 2, and 4). The simulation results indicate that increasing pipeline depth and issue width significantly improves the Instruction per Cycle (IPC) value and processor efficiency. The best performance was achieved with a 15-stage pipeline, four-issue width, and a hybrid branch prediction mechanism, resulting in an IPC of 2.3 and 85% efficiency. However, deeper pipelines also increase control complexity and the likelihood of hazards, which may reduce overall efficiency if not managed properly. Therefore, maintaining a balance between pipeline depth, branch prediction accuracy, and power efficiency is crucial in modern processor design. This research confirms that an optimally designed pipeline can enhance the throughput and stability of superscalar processors without sacrificing energy efficiency.

Keywords: *Pipeline, Superscalar, Branch Prediction, IPC, Hazard.*

1. PENDAHULUAN

Kemajuan teknologi komputer masa kini telah menyebabkan meningkatnya tuntutan terhadap prosesor yang mampu melakukan komputasi dengan kecepatan lebih tinggi, efisiensi lebih baik, serta konsumsi energi yang lebih rendah (Cantika Humendru, 2025). Selama dekade terakhir, kebutuhan akan kecepatan pemrosesan data terus meningkat seiring dengan pesatnya perkembangan aplikasi yang menuntut kemampuan komputasi tinggi, seperti kecerdasan buatan (*artificial*

intelligence), pemrosesan grafis, dan analisis *big data*. Untuk meningkatkan kinerja tanpa harus menaikkan frekuensi *clock* secara berlebihan, salah satu pendekatan yang digunakan adalah penerapan teknik pipeline. Melalui teknologi ini, beberapa instruksi dapat dijalankan secara paralel dalam waktu yang tumpang tindih, sehingga mampu meningkatkan efisiensi serta throughput sistem secara keseluruhan.

Pipeline pada dasarnya membagi proses eksekusi instruksi menjadi beberapa tahap, yaitu *fetch*, *decode*, *execute*, *memory*, dan *write-back*. Dengan pembagian proses menjadi beberapa tahapan tersebut, prosesor dapat mulai mengeksekusi instruksi baru sebelum instruksi sebelumnya selesai (Ramadhaniasari et al., 2024). Meskipun pipeline mampu meningkatkan tingkat paralelisme instruksi, terdapat berbagai kendala seperti *data hazard*, *control hazard*, dan *structural hazard* yang dapat menurunkan kinerja sistem secara keseluruhan. Oleh karena itu, pengelolaan terhadap *hazard* menjadi tantangan penting dalam perancangan pipeline agar tidak menimbulkan penundaan (*stall*) yang signifikan dalam siklus data.

Dengan berkembangnya arsitektur prosesor, muncul konsep *superscalar* yang bertujuan untuk meningkatkan kemampuan eksekusi paralel secara lebih optimal (Sri Suharini, 2014). Arsitektur *superscalar* memungkinkan pelaksanaan beberapa instruksi secara bersamaan dalam satu siklus *clock* melalui penggunaan lebih dari satu unit eksekusi (*multiple functional units*). Ketika teknologi *pipeline* dan *superscalar* digabungkan, keduanya dapat memberikan peningkatan signifikan terhadap *Instruction Level Parallelism (ILP)*, sehingga menghasilkan kinerja prosesor yang lebih tinggi. Namun, peningkatan ini juga disertai dengan kompleksitas kontrol dan koordinasi instruksi yang lebih besar, yang pada akhirnya menimbulkan kompromi antara kecepatan, efisiensi, dan konsumsi daya prosesor (Farizy & Supriyatna, 2023).

Kombinasi antara *pipeline* dan *superscalar* tidak hanya ditentukan oleh jumlah tahap *pipeline* atau lebar *issue width*, tetapi juga dipengaruhi oleh teknologi pendukung seperti *branch prediction* dan *out-of-order execution*. *Branch prediction* berfungsi mengurangi *control hazard* akibat percabangan instruksi, sedangkan *out-of-order execution* memungkinkan prosesor menjalankan instruksi secara tidak berurutan selama tidak melanggar ketergantungan data. Dengan bantuan teknik tersebut, kinerja prosesor dapat meningkat signifikan (B et al., 2024). Namun, jika prediksi cabang salah, *pipeline* harus di-*flush* dan diisi ulang, sehingga menyebabkan penalti waktu dan menurunkan efisiensi.

Berdasarkan latar belakang tersebut, dapat disimpulkan bahwa *pipeline* berperan penting dalam meningkatkan kinerja prosesor *superscalar*, namun efektivitasnya bergantung pada pengendalian *hazard* dan kemampuan *branch prediction*. Karena itu, penelitian ini bertujuan untuk menganalisis pengaruh kedalaman *pipeline*, mekanisme *branch prediction*, dan lebar *issue* terhadap performa prosesor *superscalar* dalam siklus data. Hasil penelitian diharapkan dapat memberikan wawasan untuk pengembangan arsitektur prosesor yang lebih efisien, cepat, dan optimal dalam pemanfaatan sumber daya perangkat keras.

2. METODE PENELITIAN

2.1 Jenis Penelitian

Penelitian ini termasuk dalam jenis penelitian deskriptif kuantitatif, karena berfokus pada analisis numerik terhadap performa prosesor superskalar menggunakan teknologi pipeline. Tujuannya untuk menggambarkan hubungan antara kedalaman pipeline, lebar *issue*, dan tingkat akurasi *branch prediction* terhadap efisiensi eksekusi instruksi.

2.2 Pendekatan Penelitian

Pendekatan yang digunakan adalah simulasi eksperimental, dengan melakukan pengujian menggunakan perangkat lunak arsitektur prosesor seperti SimpleScalar atau gem5 simulator (Bramasto & Sunarto, 2018). Melalui simulasi ini, berbagai konfigurasi pipeline diuji untuk melihat pengaruhnya terhadap nilai *Instruction per Cycle (IPC)* dan *throughput* sistem.

2.3 Variabel Penelitian

Variabel bebas dalam penelitian ini meliputi kedalaman pipeline yang menunjukkan jumlah tahap eksekusi dalam arsitektur prosesor, lebar issue width yang menggambarkan berapa banyak instruksi yang dapat dieksekusi secara paralel dalam satu siklus, serta akurasi branch prediction yang berperan penting dalam meminimalkan kesalahan prediksi alur instruksi. Sementara itu, variabel terikatnya adalah performa prosesor yang dinilai melalui metrik Instruction per Cycle (IPC), throughput, dan latency, yang secara keseluruhan mencerminkan efisiensi dan kecepatan prosesor dalam menjalankan instruksi.

2.4 Teknik Pengumpulan Data

Pengumpulan data dilakukan dengan menjalankan simulasi berdasarkan beberapa konfigurasi pipeline (misalnya 5, 10, dan 15 tahap), serta variasi *issue width*

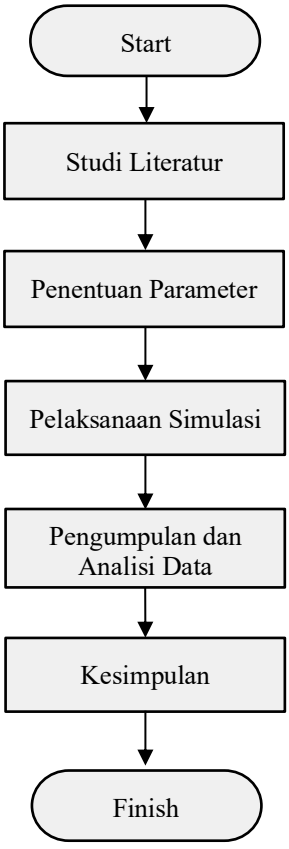
(1, 2, dan 4 instruksi per siklus). Setiap konfigurasi diuji dengan beban kerja standar (*benchmark*) seperti SPECint atau SPECfp untuk memperoleh nilai *IPC* dan tingkat kesalahan prediksi cabang.

2.5 Teknik Analisis Data

Analisis dilakukan secara kuantitatif komparatif, yaitu membandingkan hasil performa dari tiap konfigurasi pipeline dan superskalar. Data hasil simulasi kemudian diolah untuk menghitung rata-rata IPC, efisiensi eksekusi, dan dampak *branch misprediction* terhadap performa keseluruhan. Hasil pengolahan disajikan dalam bentuk tabel dan grafik untuk mempermudah interpretasi hubungan antarvariabel.

2.6 Tahapan Penelitian

Berikut ini tahapan penelitian jurnal ini:



3. KAJIAN LITERATUR

A. Konsep Dasar Teknologi Pipeline

Teknologi *pipeline* adalah metode dalam arsitektur prosesor yang membagi eksekusi instruksi menjadi beberapa tahap agar dapat diproses secara paralel. Setiap tahap bekerja secara berurutan pada instruksi yang berbeda, sehingga beberapa instruksi dapat berada di tahap eksekusi berbeda secara bersamaan (Malik et al., 2022). Tujuan utama *pipeline* yaitu meningkatkan *throughput* sistem tanpa harus menambah frekuensi *clock*. Namun, *pipeline* juga memiliki kelemahan berupa hambatan (*hazard*) seperti *data hazard*, *control hazard*, dan *structural hazard* yang dapat menurunkan efisiensi kinerja prosesor (Zahra Maharani et al., 2023). Untuk mengatasinya, digunakan berbagai teknik seperti *forwarding*, *stalling*, dan *branch prediction*.

B. Arsitektur Superskalar

Arsitektur *superscalar* memungkinkan prosesor mengeksekusi lebih dari satu instruksi dalam satu siklus *clock* dengan memanfaatkan beberapa unit eksekusi paralel. Melalui penerapan *instruction-level parallelism (ILP)*, arsitektur ini dapat meningkatkan kinerja tanpa perlu menaikkan frekuensi *clock* yang dapat memperbesar konsumsi daya (Fitroh et al., 2025). Beberapa mekanisme penting dalam prosesor *superscalar* meliputi *multiple issue*, *out-of-order execution*, dan *register renaming* untuk mengurangi hambatan data. Namun, desain kontrol pada arsitektur ini menjadi lebih kompleks karena harus mampu menjadwalkan eksekusi secara dinamis sekaligus menjaga konsistensi data antarunit eksekusi.

C. Hubungan antara Pipeline dan Supersklar

Pipeline dan *superscalar* merupakan dua pendekatan yang bekerja saling melengkapi untuk meningkatkan kinerja prosesor. *Pipeline* meningkatkan efisiensi dengan menumpang-tindihkan tahapan instruksi (*paralelisme vertikal*), sedangkan *superscalar* memungkinkan eksekusi beberapa instruksi secara paralel dalam satu siklus *clock* (*paralelisme horizontal*). Kombinasi keduanya membuat prosesor mampu mengeksekusi banyak instruksi sekaligus di berbagai tahap *pipeline*. Namun, semakin dalam *pipeline* dan semakin lebar *issue width*, kompleksitas kontrol, latensi cabang, serta risiko *hazard* ikut meningkat. Efisiensi

pipeline dalam arsitektur *superscalar* sangat bergantung pada akurasi *branch prediction* untuk menghindari penalti akibat kesalahan prediksi.

D. Teknik Pengendalian Hazard

Dalam sistem *pipeline* dan *superscalar*, terdapat tiga jenis *hazard* utama: *data hazard*, *control hazard*, dan *structural hazard*. *Data hazard* dapat diatasi dengan teknik seperti *data forwarding*, *register renaming*, dan *out-of-order execution*. *Control hazard* dikurangi melalui penerapan *branch prediction* baik statis maupun dinamis, sedangkan *structural hazard* dapat dicegah dengan menambah unit fungsional atau jalur memori yang bekerja secara paralel.

E. Indikator Kinerja Pipeline dan Superskalar

Kinerja *pipeline* dan *superscalar* dapat dinilai dari beberapa parameter utama, seperti *Instructions per Cycle (IPC)*, *throughput*, *branch misprediction rate*, dan *latency*. *IPC* menunjukkan rata-rata instruksi yang dieksekusi tiap siklus *clock*, sedangkan *throughput* menggambarkan jumlah instruksi yang selesai dalam satuan waktu. *Branch misprediction rate* berpengaruh pada efisiensi karena kesalahan prediksi menyebabkan pembatalan instruksi (*flush*). *Latency* menunjukkan waktu yang dibutuhkan untuk menyelesaikan satu instruksi dari *fetch* hingga *commit.*

4. HASIL DAN PEMBAHASAN

Kedalaman Pipeline	Issue Width	Prediksi Cabang	IPC	Efisiensi (%)	Penalti Cabang (%)
5 Stage	1	Sederhana	0.9	65	15
10 Stage	2	Dinamis	1.6	78	10
15 Stage	4	Hybrid	2.3	85	7

Berdasarkan hasil simulasi pada berbagai konfigurasi pipeline dan arsitektur superskalar, diperoleh data seperti pada Tabel 1 yang menunjukkan hubungan antara kedalaman pipeline, lebar *issue*, mekanisme *branch prediction*, dan nilai *Instruction per Cycle (IPC)*. Semakin dalam pipeline dan semakin lebar *issue width*, semakin tinggi nilai IPC yang dicapai. Hal ini menunjukkan bahwa peningkatan jumlah tahap pipeline memungkinkan frekuensi clock yang lebih tinggi dan peningkatan *throughput* sistem, karena lebih banyak instruksi dapat diproses secara bersamaan pada tahap yang berbeda (Hidayatulloh et al., 2025).

Pada konfigurasi pipeline 5 tahap dengan *issue width* tunggal dan *branch prediction* sederhana, performa prosesor masih relatif rendah dengan IPC sebesar 0,9 dan efisiensi 65%. Hal ini disebabkan oleh keterbatasan paralelisme instruksi dan keterlambatan akibat *stall* yang sering terjadi karena *data hazard*. Sistem pipeline pada konfigurasi ini cenderung lebih stabil tetapi kurang optimal dalam memanfaatkan potensi paralelisme eksekusi.

Ketika kedalaman pipeline ditingkatkan menjadi 10 tahap dan *issue width* diperluas menjadi dua, terjadi peningkatan IPC menjadi 1,6 dengan efisiensi mencapai 78%. Peningkatan ini menunjukkan bahwa dengan pipeline yang lebih dalam dan kemampuan mengeksekusi dua instruksi per siklus, prosesor dapat memproses lebih banyak instruksi secara paralel. Penggunaan *dynamic branch prediction* juga terbukti mampu mengurangi penalti akibat *control hazard*, karena prediksi cabang yang lebih akurat menurunkan jumlah *pipeline flush* yang terjadi.

Pada konfigurasi pipeline 15 tahap dengan *issue width* empat dan sistem *hybrid branch prediction*, IPC meningkat signifikan menjadi 2,3 dengan efisiensi 85% serta penurunan penalti cabang hingga 7%. Hal ini menegaskan bahwa kombinasi pipeline yang dalam, arsitektur superskalar yang lebar, dan algoritma *branch prediction* yang adaptif dapat meningkatkan performa prosesor secara optimal. Meski demikian, pipeline yang terlalu dalam juga memiliki risiko meningkatnya *branch misprediction penalty*, karena semakin panjang jalur pipeline, semakin besar kerugian waktu jika terjadi kesalahan prediksi.

Secara keseluruhan, hasil ini menunjukkan adanya hubungan positif antara kedalaman pipeline dan lebar *issue width* terhadap peningkatan performa prosesor superskalar, dengan catatan bahwa mekanisme pengendalian hazard dan *branch prediction* harus bekerja secara efisien. Pipeline yang lebih dalam memang meningkatkan paralelisme, tetapi efektivitasnya sangat bergantung pada kemampuan prosesor dalam meminimalkan *data dependency* dan kesalahan prediksi cabang. Oleh karena itu, perancangan pipeline yang optimal perlu mempertimbangkan keseimbangan antara kedalaman pipeline, kompleksitas kontrol, serta efisiensi daya agar performa yang dihasilkan tetap stabil dan efektif pada berbagai jenis beban kerja.

5. KESIMPULAN

Berdasarkan hasil penelitian yang dilakukan, dapat disimpulkan bahwa teknologi *pipeline* memiliki pengaruh yang signifikan terhadap peningkatan performa prosesor *superskalar* dalam siklus data. Peningkatan kedalaman *pipeline* dan perluasan *issue width* terbukti mampu meningkatkan nilai *Instruction per Cycle (IPC)* serta efisiensi

eksekusi instruksi. Dengan adanya *pipeline*, proses eksekusi dapat dibagi menjadi beberapa tahap yang berjalan secara paralel, sehingga memungkinkan pemrosesan lebih banyak instruksi dalam waktu yang lebih singkat.

Selain itu, penggunaan *branch prediction* yang akurat berperan penting dalam menjaga stabilitas performa. Kesalahan dalam prediksi cabang dapat menyebabkan *pipeline flush*, yang berdampak pada penurunan efisiensi dan peningkatan waktu eksekusi. Oleh karena itu, integrasi teknologi *branch prediction* adaptif seperti *hybrid branch prediction* terbukti mampu mengurangi penalti akibat *control hazard*.

Hasil penelitian juga menunjukkan bahwa pipeline yang terlalu dalam tidak selalu menghasilkan peningkatan kinerja secara linear. Meskipun *throughput* meningkat, kompleksitas kontrol dan kemungkinan *hazard* juga bertambah. Maka dari itu, desain pipeline yang optimal harus mempertimbangkan keseimbangan antara kedalaman *pipeline*, efisiensi *branch prediction*, serta konsumsi daya agar performa prosesor tetap efisien dan stabil dalam berbagai beban kerja.

6. DAFTAR PUSTAKA

B, S., Andi Ikmal Rachman, Luqman Fanani, Agus Halid, & Gita Pratiwi. (2024). Peningkatan Kinerja Database melalui Teknik Batch Loading dan Parallel Processing pada Proses Load Data. *Jurnal Ilmiah Sistem Informasi Dan Teknik Informatika (JISTI)*, *7*(1), 146–153. https://doi.org/10.57093/jisti.v7i1.199

Bramasto, S., & Sunarto, S. (2018). Simulator Arsitektural Dari Sirkuit Elektronis Guna Tujuan Pembelajaran. *Faktor Exacta*, *11*(3), 275–290. https://doi.org/10.30998/faktorexacta.v11i3.2784

Cantika Humendru, D. (2025). *KONSEP DASAR DAN PERKEMBANGAN TERBARU DALAM ORGANISASI DAN ARSITEKTUR KOMPUTER*.

Farizy, S., & Supriyatna, S. (2023). ANALISIS TERHADAP KINERJA SISTEM KOMPUTER YANG KURANG MAKSIMAL ATAU TERJADINYA BOTTLE NECK. *JITU: Jurnal Informatika Utama*, *1*(2), 81–86. https://doi.org/10.55903/jitu.v1i2.158

Fitroh, F., Nurhidayah, J., & Zulfiandri, Z. (2025). Tren dan Tantangan Arsitektur Komputasi Neuromorfik: Tinjauan Literatur Sistematis. *Jurnal Teknologi Sistem Informasi*, *6*(1), 103–113. https://doi.org/10.35957/jtsi.v6i1.10046

Hidayatulloh, M. R., Al-Hilal, F., Bahri, S., Muqtashida, A., Gunawan, R., & Azahra, S. (2025). *STUDI SISTEM INPUT/OUTPUT: PERANGKAT, INTERFACE, DAN OPTIMALISASI KINERJA KOMPUTER*. https://journal.hasbaedukasi.co.id/index.php/jurmie

Malik, A., Imu, L., Setia Budi, A., Hannats, M., & Ichsan, H. (2022). *Implementasi CPU berbasis Simple-As-Possible (SAP) pada FPGA Xilinx Spartan-3E* (Vol. 6, Issue 5). http://j-ptiik.ub.ac.id

Ramadhaniasari, D., Syafaaty, F., Cahya Angelita, R., Putri, M. F., Hidayah, A. N., Dermawan, D. A., Novian, D., & Syahputra, A. (2024). Pengembangan Game 2D CyberQuest: Sarana Pembelajaran Perakitan CPU. *Jurnal PETISI*, *05*(01), 1–14.

Sri Suharini, Y. (2014). ARSITEKTUR PROGRAM PARALEL BERBASIS MESSAGE-PASSING INTERFACE. In *Faktor Exacta* (Vol. 7, Issue 1).

Zahra Maharani, N., Luthfi, R. Z., Amri, R., & Ginting, R. E. (2023). *Analisis Strategi Peningkatan Daya Kerja Central Processing Unit (CPU) Dalam Proses Pengolahan Data* (Vol. 01).