Laporan Praktikum

Sistem Operasi

Dosen pengampu : (*Iwan Lesmana, S.Kom., M.Kom.*)

Modul 1

Nama : Muhammad Rizal Nurfirdaus

NIM : 20230810088

Kelas : TINFC – 2023 – 04

Teknik Informatika

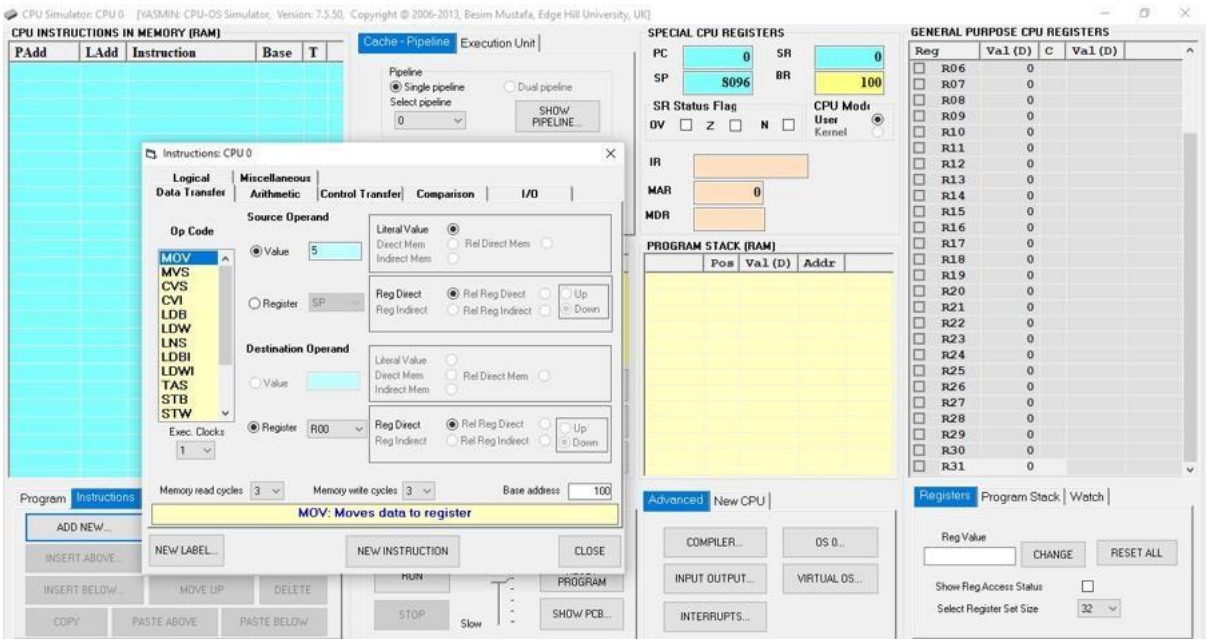
Fakultas Ilmu Komputer

Universitas Kuningan

Praktikum

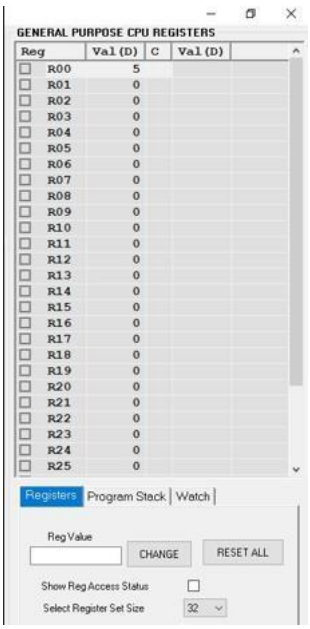
a. Transfer data

1. Buatlah instruksi yang memindahkan (move) angka 5 ke register R00.



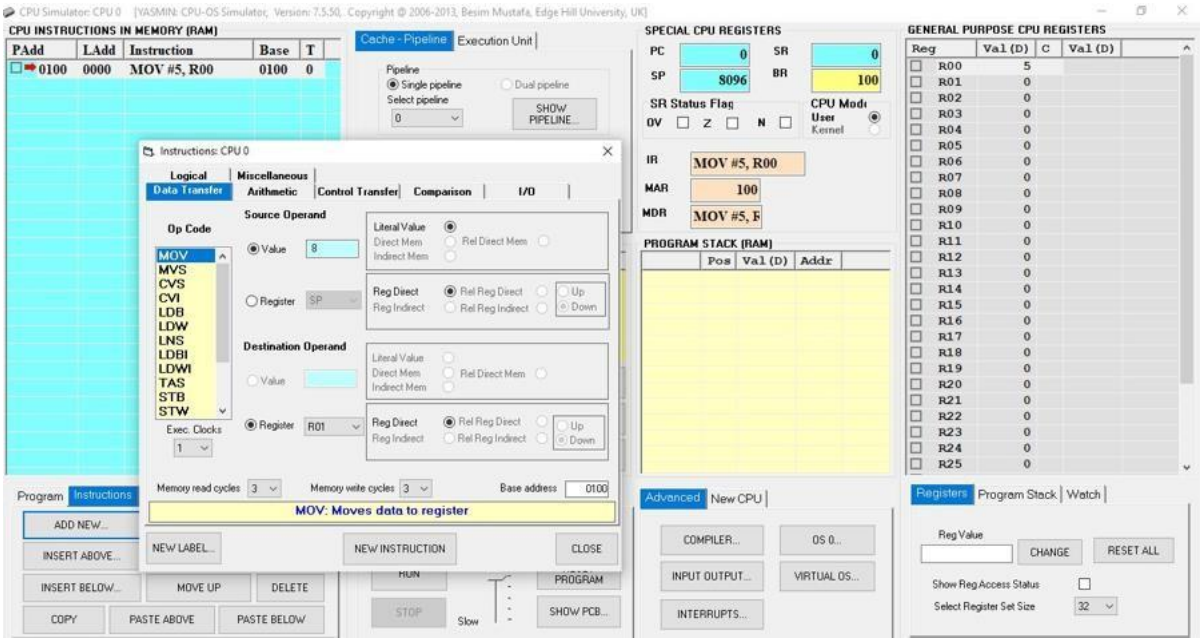
Langkah pertama yaitu ADD NEW dan memilih MOV, lalu masukkan fungsi Source Operand dengan Value 5 dan Destination Operand Register R00. Lalu juka sudah pilih NEW INSTRUCTION.

2. Eksekusi instruksi diatas (dengan klik dua kali pada tampilan instruction memory).



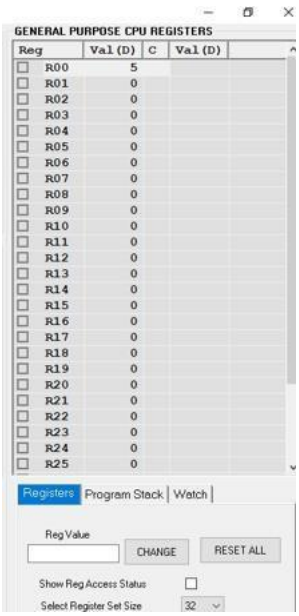
Setelah di eksekusi maka akan menampilkan data R00 = 5. Yang artinya data sudah masuk

3. Buatlah instruksi yang memindahkan angka 8 ke register R01



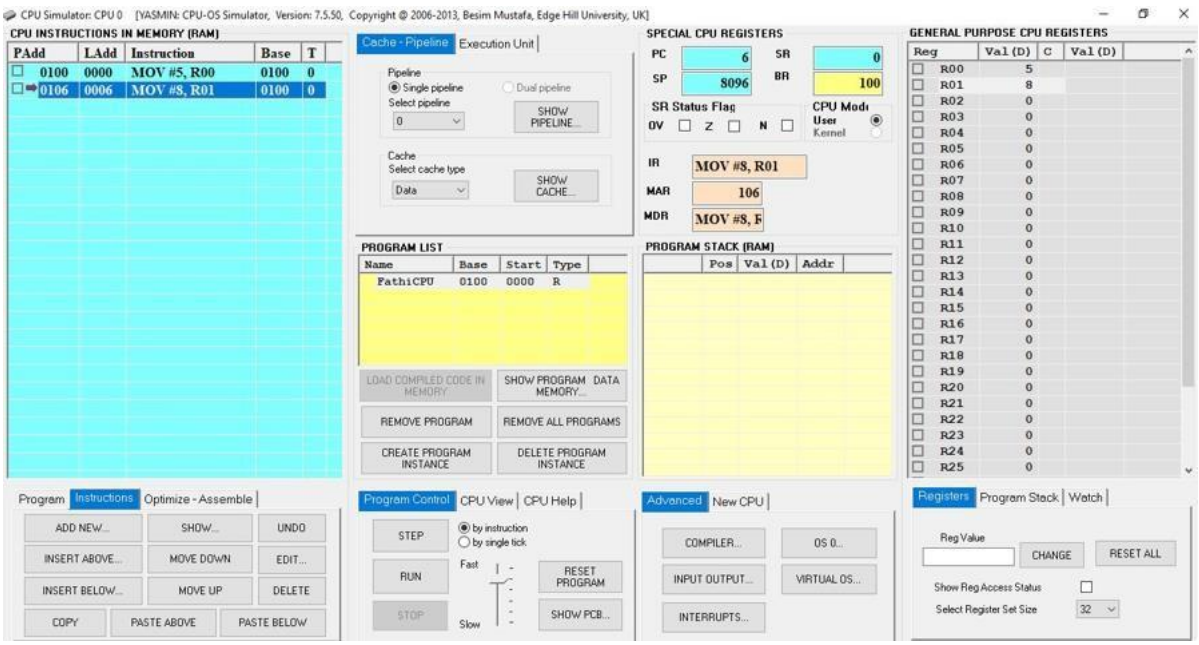
Setelah itu ADD NEW dan memilih MOV kembali, lalu masukkan fungsi Source Operand dengan Value 8 dan Destination Operand Register R01. Lalu juka sudah pilih NEW INSTRUCTION.

4. Eksekusilah.



Berikut tampilan ketika sudah di eksekusi, maka nilai yang masuk pada R01 adalah 8.

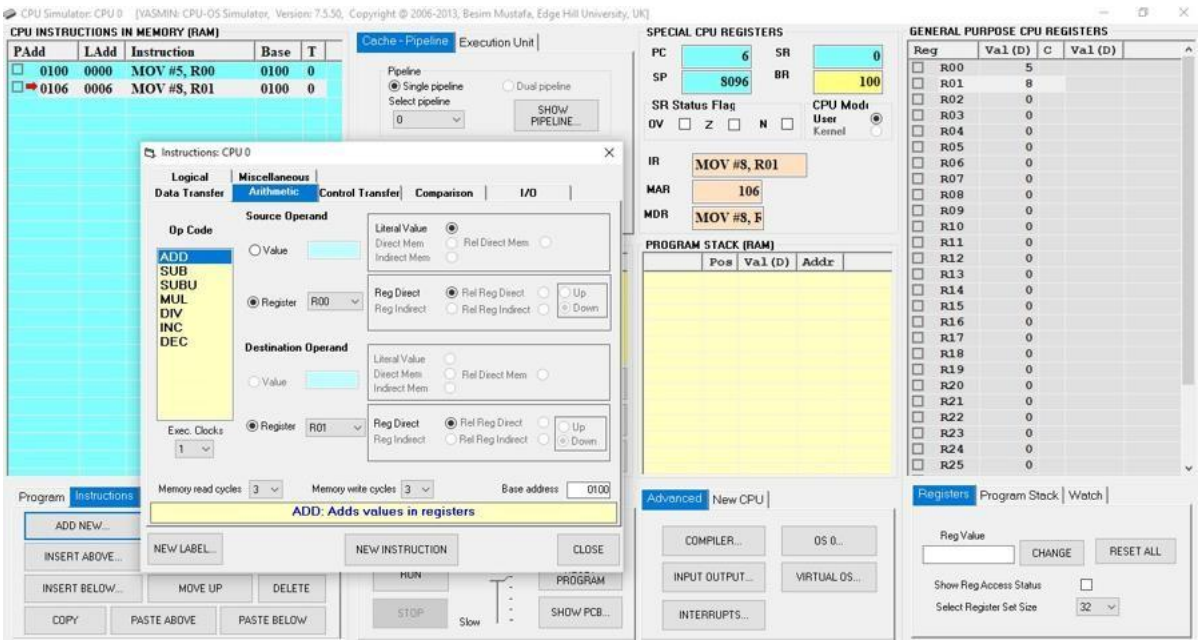
5. Amati isi R00 dan R01 pada tampilan Register Set.



Berikut isi dari R00 dan R01 pada tampilan Register Set.

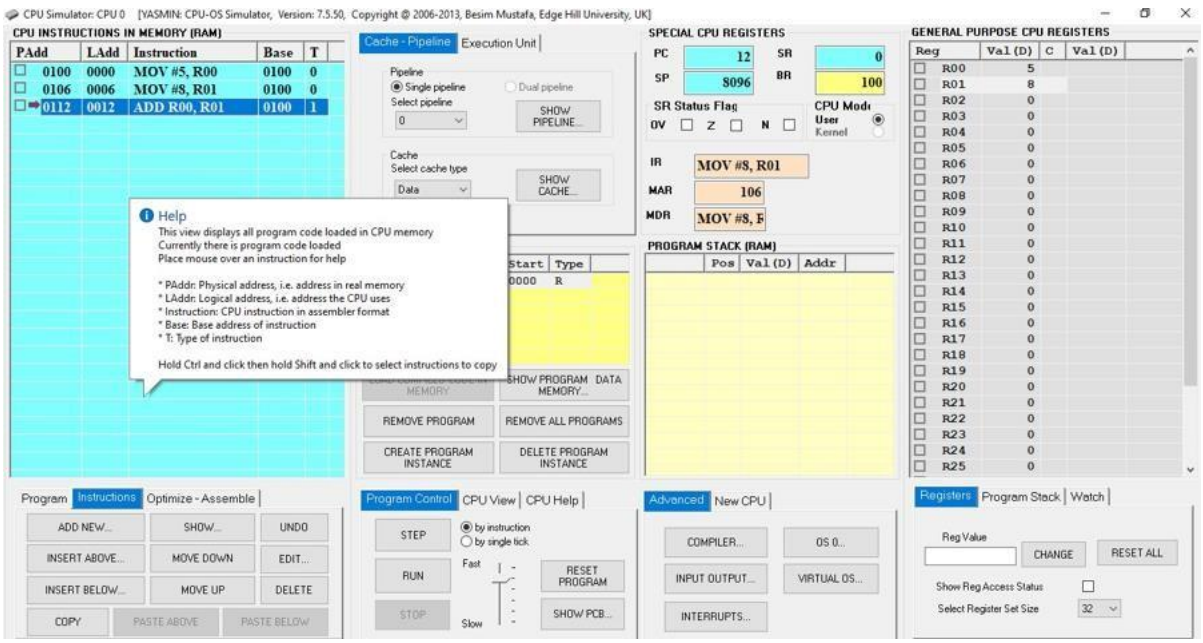
b. Aritmatika

6. Buatlah suatu instruksi yang menambahkan (add) isi R00 dan R01.



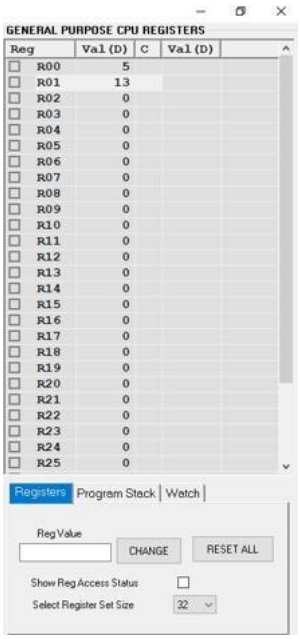
Setelah itu ADD NEW dan memilih Arimatic lalu pilih ADD, lalu masukkan fungsi Source Operand dengan Register R00 dan Destination Operand Register R01. Lalu juka sudah pilih NEW INSTRUCTION.

7. Eksekusilah.



Setelah itu eksekusi perintah yang telah dimasukan tadi, dengan menekan 2(dua) kali pada ADD R00,R01.

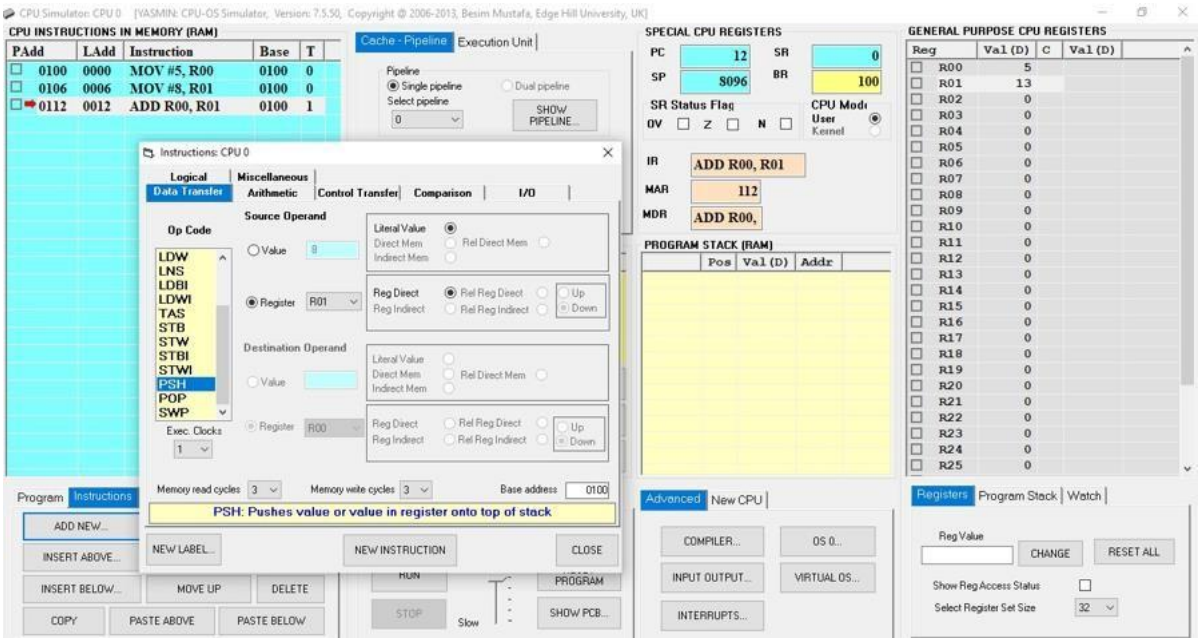
8. Amati dimana hasil penjumlahan tersebut disimpan.



Berikut hasil dari penambahan, dimana yang tadinya menampilkan angka 8 pada line ke 2. Menjadi angka 13, hal itu dikarenakan angka 8 akan ditambahkan pada angka 5 yang terdapat pada line 1.

c. Stack Pointer (SP)

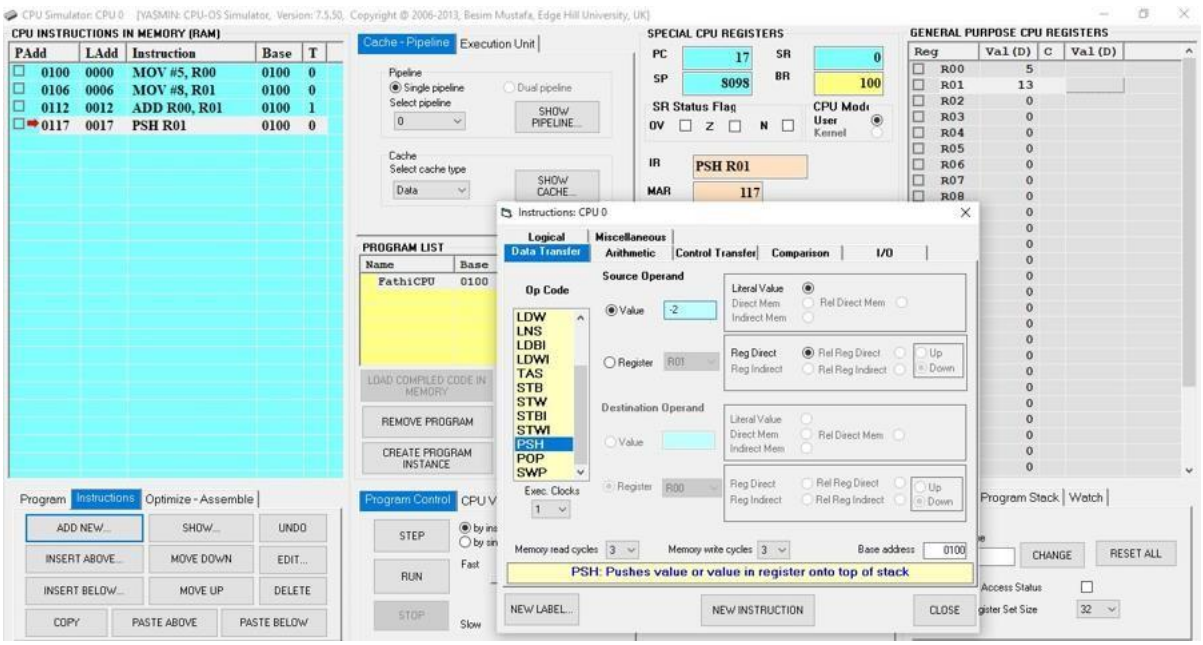
9. Buatlah instruksi yang menaruh hasil diatas pada program stack, kemudian eksekusilah.



Untuk menambahkan stuck, maka kita bisa menggunakan proses PSH yang dimana pada Source Operand pada bagian Register dipilih R01

Jika sudah di eksekusi maka akan menampilkan laporan program stack seperti diatas.

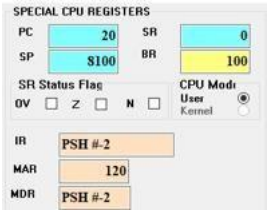
10. Buatlah instruksi untuk menaruh (push) angka -2 pada stack teratas dan eksekusilah.



Selanjutnya membuatkan perintah push. Disini kita kembali memilih PSH dan kemudian pada Source Operand di Values kita isikan sesuai perintah, yaitu -2. Kemudian Klik NEW INTRUCTION dan jika sudah pilih Close

Maka akan menampilkan program stack seperti gambar diatas, dimana angka -2 terdapat dibagian atas dari angka sebelumnya yaitu 13.

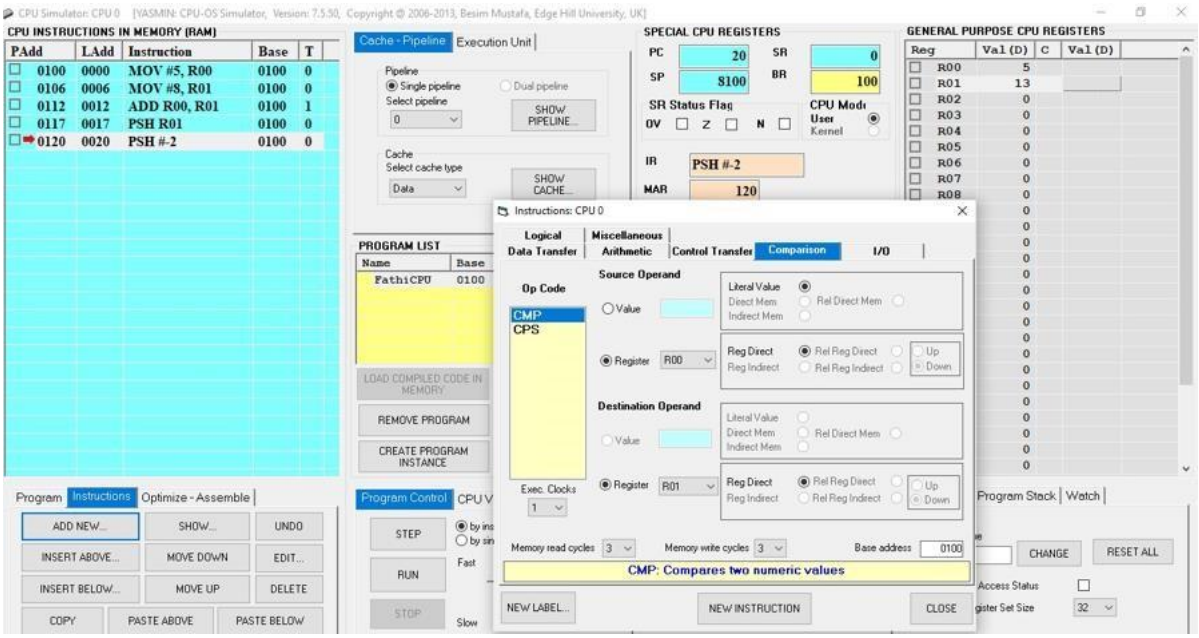
11. Amati nilai register SP.



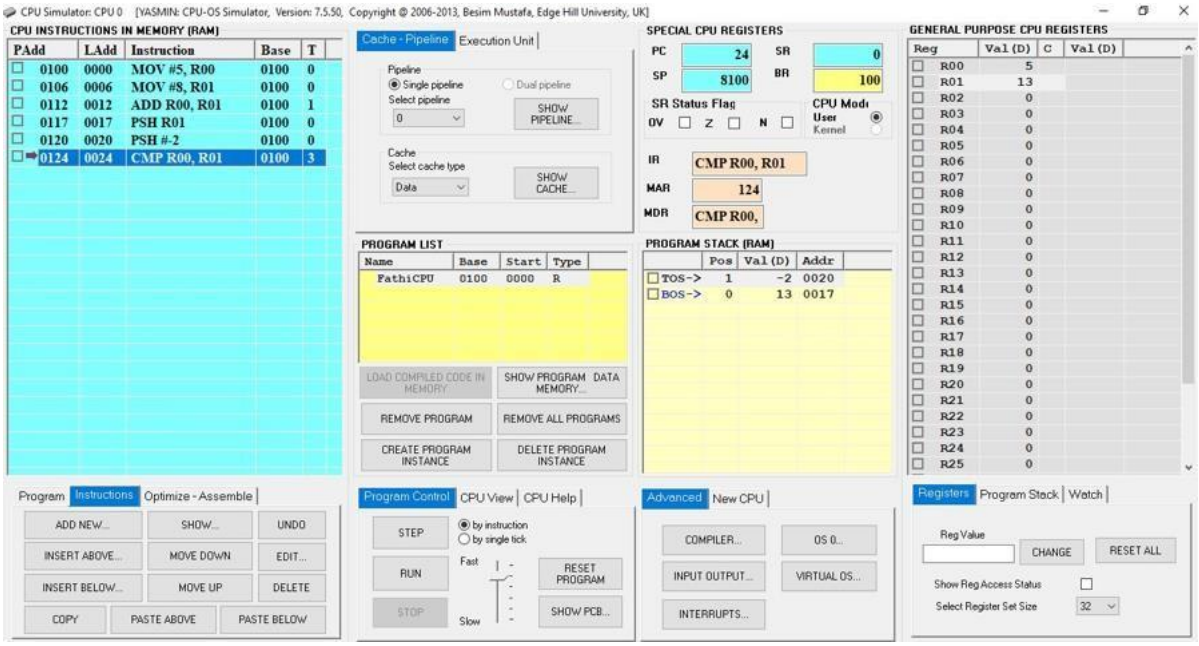
Setelah itu pada bagian Special CPU Register pada bagian SP menampilkan angka 8100.

d. Pembanding

12. Buatlah instruksi untuk membandingkan nilai register R00 dan R01.

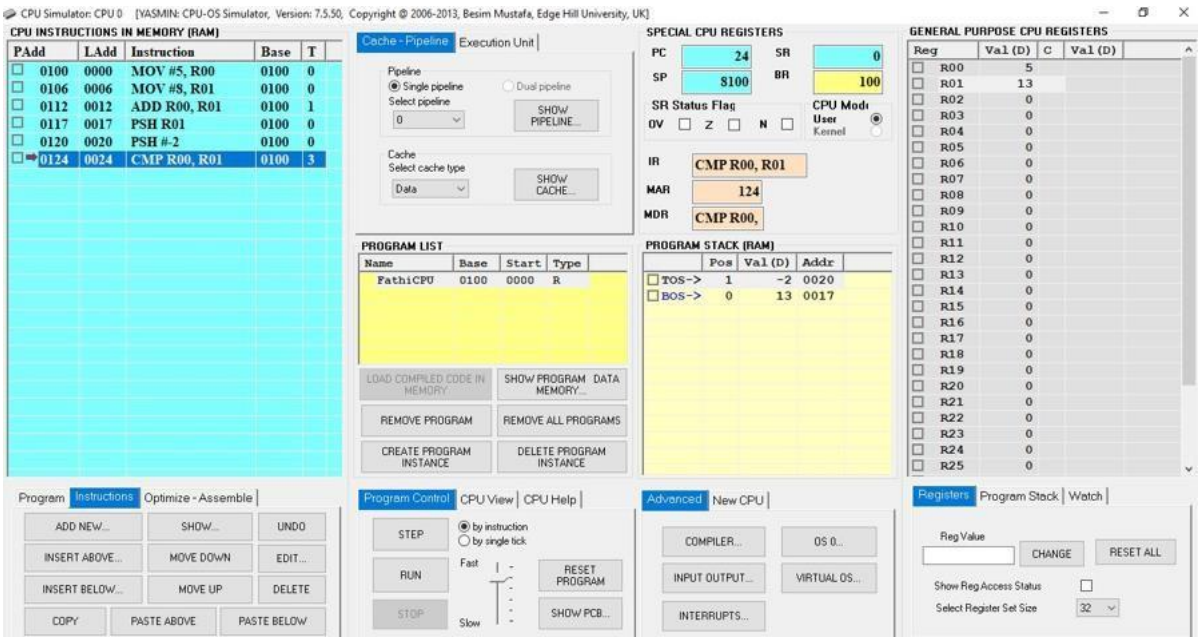


Selanjutnya, untuk membuat instruksi guna membandingkan nilai register R00 dan R01 dengan cara ADD NEW dan masuk pada Comparison dan memilih CMP untuk membandingkan. Lalu pada Source Operand di bagian register masukkan R00 dan pada bagian Destination Operand di Register masukan R01. Lalu NEW INSTRUCTION dan jika sudah tekan close 13. Eksekusilah.



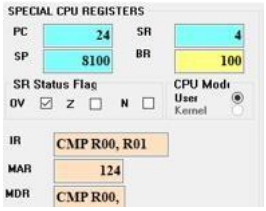
Lalu ketika sudah di eksekusi maka akan menampilkan informasi yaitu CMP R00,R01.

14. Amati nilai register SR.

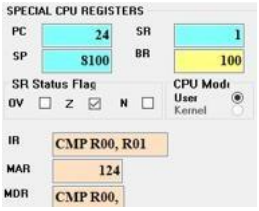


Setelah itu pada bagian SR akan menampilkan kode 0.

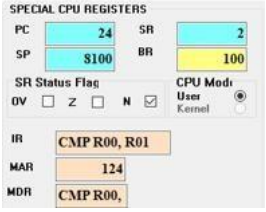
15. Amati bit status OV/Z/N pada status register.



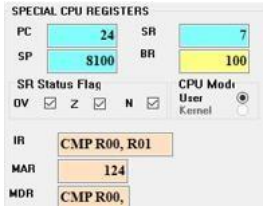
Bit Status OV yang menunjukkan perbedaan pada SR, pada OV bagian SR akan memunculkan angka 4



Bit Status Z yang menunjukkan perbedaan pada SR, pada Z bagian SR akan memunculkan angka 1

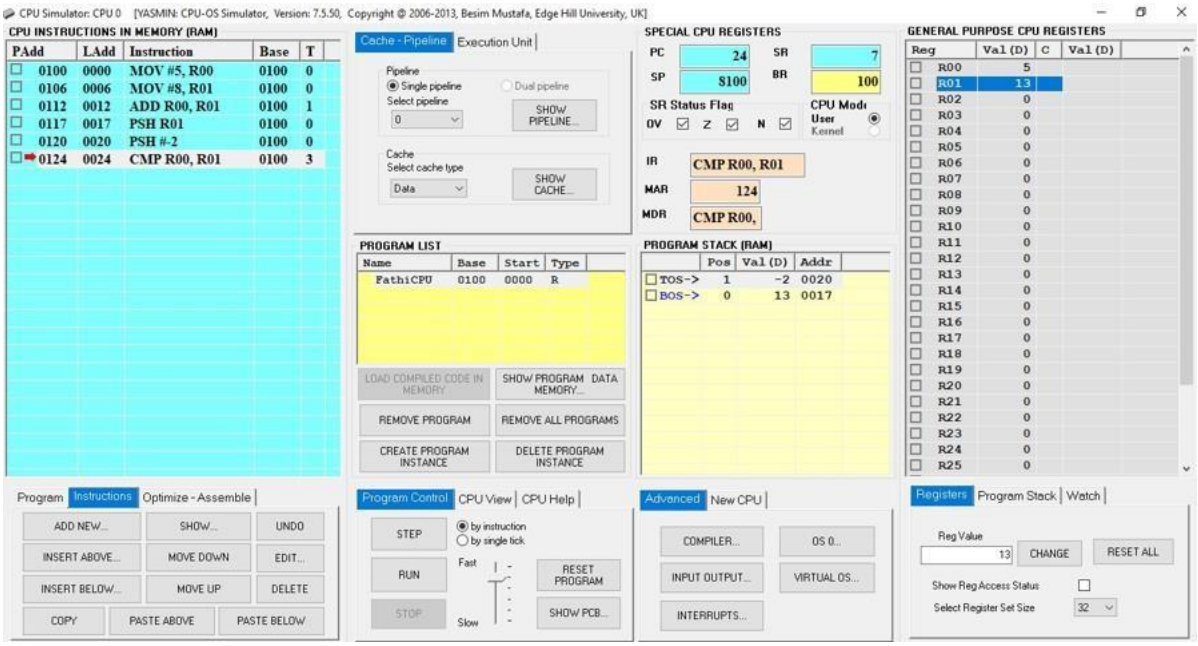


Bit Status N yang menunjukkan perbedaan pada SR, pada N bagian SR akan memunculkan angka 2



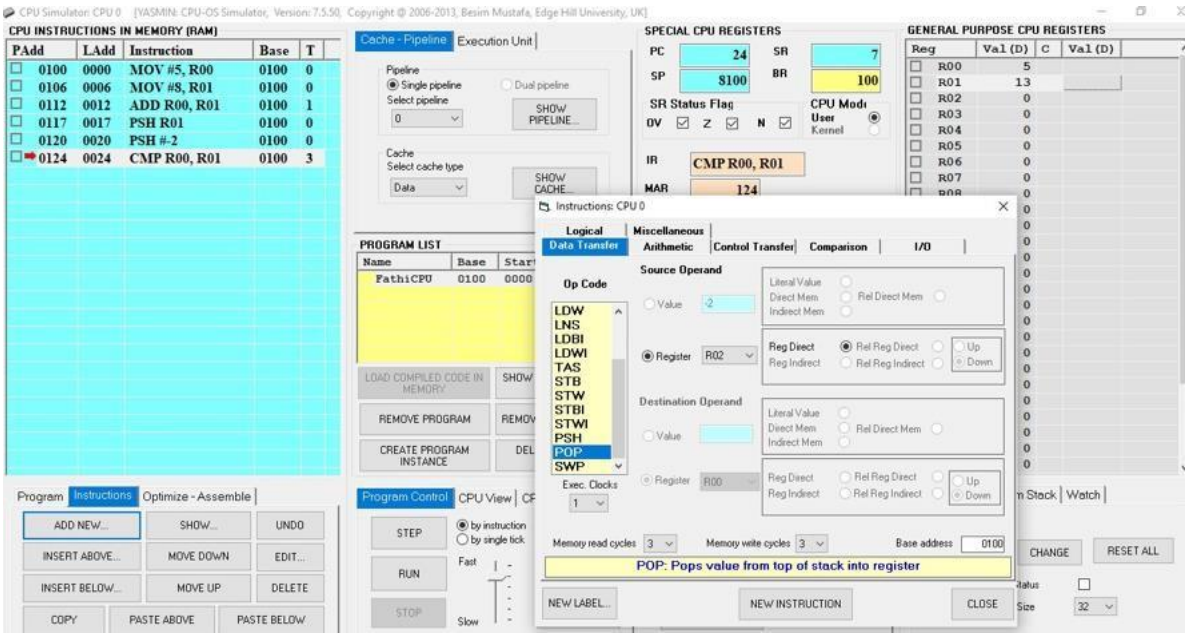
Bit Status jika memilih OV/Z/N maka pada SR yang muncul adalah angka 7 16.

Analisa status register tersebut.

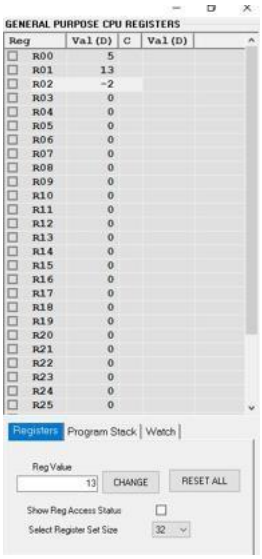


e. Stack Pointer (SP)

17. Buatlah instruksi untuk mengambil (pop) nilai teratas dari program stack ke register R02, kemudian eksekusilah.

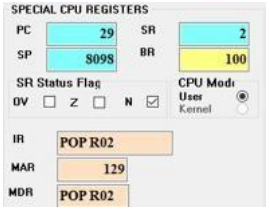


Pada instruksi untuk mengambil (pop) nilai teratas dari program stack ke register R02, dilakukanlah cara dimana kita memilih ADD NEW yang kemudian kita memilih POP, setelah itu pada bagian register di Sourch Operand kita masukan angka R02. Lalu Klik NEW INTRUCTION dan close.



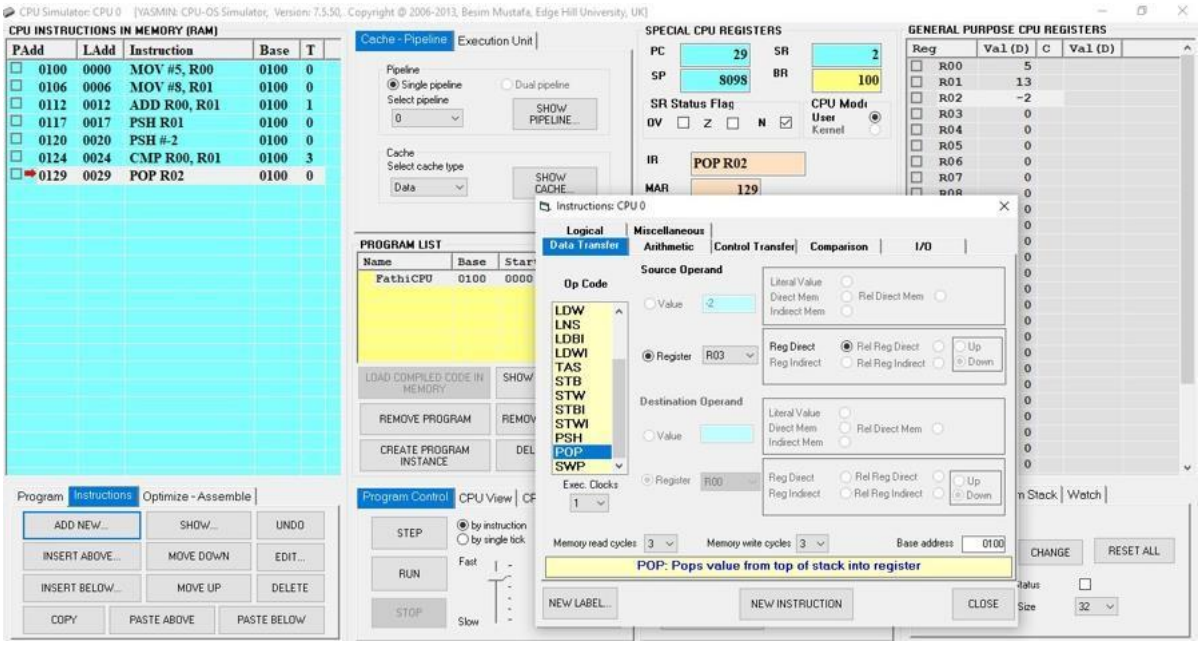
Berikut tampilan ketika telah di eksekusi, pada R02 akan menampilkan angka -2.

18. Amati nilai pada register SP.



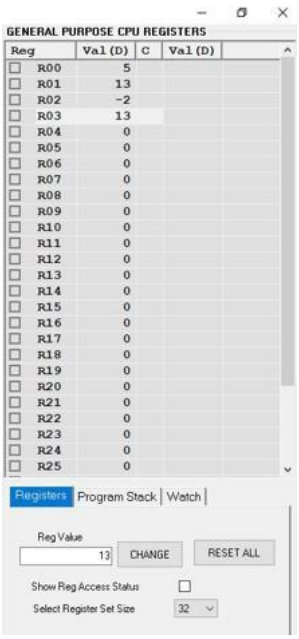
Berikut adalah tampilan yang memberikan informasi pada bagian Register SP berubah menjadi 8098

19. Buatlah suatu instruksi untuk mengambil nilai teratas dari program stack ke register R03.



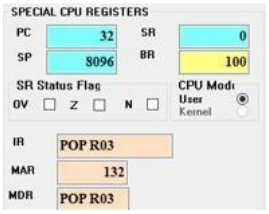
Untuk membuat instruksi untuk mengambil nilai teratas dari program stack ke register R03, maka yang perlu dilakukan adalah ADD NEW kemudian memilih menu POP. Pada bahgian sourch operand kita langsung memasuki register dan memilih R03. Setelah selesai pilih NEW INTRUCTION dan kemudian close.

20. Eksekusilah.



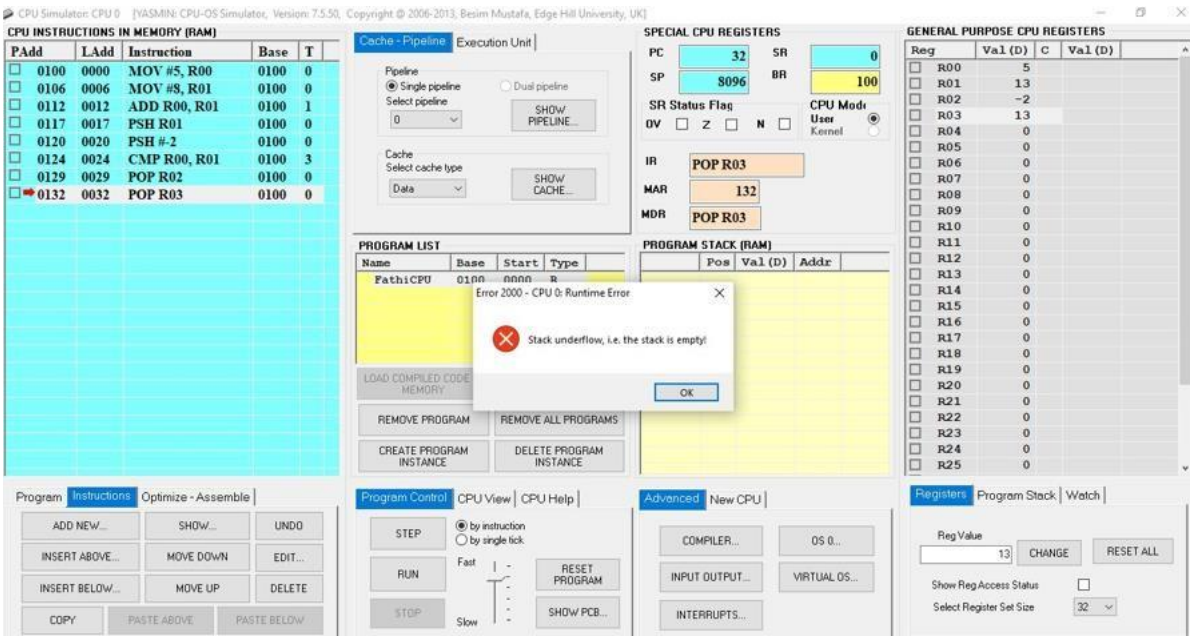
Berikut adalah hasil dari eksekusi pada perintah sebelumya. Dimana disini pada R03 menampilkan angka 13.

21. Amati nilai pada register SP.



Pada nilai register bagian SP maka akan berubah menjad 8096

22. Eksekusi lagi instruksi yang terakhir. Apa yang terjadi? Jelaskan.



Disini menampilkan pesan bahwa eksekusi tersebut eror. Hal ini dikarenakan sebelumnya telah di eksekusi terlebih dahulu. Jadi sistem ini hanya bisa satu kali eksekusi. **Pre test**

1. Apa itu CPU simulator?

CPU simulator adalah sebuah perangkat lunak yang berfungsi untuk meniru atau mensimulasikan cara kerja sebuah CPU. Dengan CPU simulator, kita bisa menjalankan instruksi-instruksi yang biasanya dijalankan oleh CPU asli. Jadi ini seperti kita sedang memahami tentang intruksi-intruksi yang bisa dijalankan CPU. 2. Sebutkan dan jelaskan dua instruksi yang dapat dieksekusi oleh CPU Simulator. Ada dua instruksi yang bida dieksekusi CPU Simulator pertama LOAD dan ADD berikut penjelasannya akan saya jelaskan:

- LOAD: Instruksi ini digunakan untuk mengambil data dari memori dan memasukkannya ke dalam register CPU. Misalnya, jika instruksi ditulis sebagai LOAD A, 10, artinya nilai dari alamat memori 10 akan diambil dan dimasukkan ke dalam register A.
- ADD: Instruksi ini digunakan untuk menambahkan nilai dari dua register atau menambahkan nilai dari register dengan sebuah nilai tertentu, kemudian menyimpan hasilnya. Contoh, ADD A, B akan menambahkan nilai di register A dengan nilai di register B, dan hasilnya disimpan kembali di register A.

3. Apa fungsi dari register PC, SR, dan SP dalam CPU? Jelaskan masing-masing. Berikut ini penjelasan dari ketiganya:
    - PC (Program Counter): Register ini berfungsi untuk menyimpan alamat instruksi berikutnya yang akan dieksekusi oleh CPU. Setiap kali CPU menyelesaikan sebuah instruksi, nilai di PC akan diperbarui untuk menunjuk ke instruksi berikutnya.
    - SR (Status Register): Register ini menyimpan informasi status hasil operasi sebelumnya, seperti apakah hasilnya nol, negatif, atau terjadi overflow. Informasi di SR digunakan oleh CPU untuk menentukan langkah eksekusi selanjutnya.
    - SP (Stack Pointer): Register ini melacak posisi stack, yang merupakan area memori sementara yang digunakan untuk menyimpan data seperti alamat pengembalian dan variabel lokal. Nilai SP akan berubah setiap kali ada data yang ditambahkan atau diambil dari stack.

Post Test

1. Jelaskan langkah-langkah untuk memindahkan data ke register menggunakan CPU simulator.

   Untuk memindahkan data ke register dalam CPU simulator, langkah-langkah yang perlu kita lakukan adalah sebagai berikut:
    - Pertama, gunakan instruksi LOAD untuk mengambil data dari lokasi memori tertentu.
    - Kedua, Tentukan alamat memori sumber data yang akan diambil.
    - Ketiga, Pilih register tujuan di mana data akan ditempatkan.
    - Keempat, Jalankan instruksi, dan data akan dipindahkan dari memori ke register yang dipilih.
2. Bagaimana cara menambahkan nilai dalam register menggunakan CPU simulator.
   Untuk menambahkan nilai dalam register, CPU simulator menyediakan instruksi seperti ADD. Berikut adalah langkah-langkahnya:
    - Pilih dua register yang ingin dijumlahkan nilainya atau satu register dan sebuah nilai tertentu.
    - Gunakan instruksi ADD dengan format ADD [Register1], [Register2] atau ADD [Register], [nilai].
    - Setelah instruksi dijalankan, hasil penjumlahan akan tersimpan di register pertama yang dipilih.
3. Jelaskan perbedaan antara register dan stack dalam konteks CPU.
   Register adalah tempat penyimpanan data yang sangat cepat dan terletak langsung di dalam CPU, digunakan untuk menyimpan data sementara yang diperlukan secara langsung untuk eksekusi instruksi. Stack, di sisi lain, adalah area memori khusus yang digunakan untuk menyimpan data sementara seperti variabel lokal atau alamat pengembalian fungsi, dengan sistem akses LIFO (Last In, First Out).

Register bekerja lebih cepat dibandingkan stack karena berada di dalam CPU, sedangkan stack berada di memori utama.

Tugas

1. Buatlah program CPU sesuai tahap-tahap yang telah dijelaskan. Jalankan serta buat kesimpulan.

   Berikut adalah contoh tahap-tahap untuk membuat program CPU sederhana menggunakan CPU simulator, serta kesimpulannya:

a. Langkah-langkah Pembuatan Program CPU Sederhana:

- Inisialisasi Memori dan Register: Pertama, tentukan nilai awal di beberapa alamat memori jika diperlukan. Siapkan register yang akan digunakan.
- Instruksi `LOAD`: Gunakan instruksi `LOAD` untuk memuat data dari alamat memori tertentu ke dalam register. Misalnya, `LOAD A, 10` akan memuat data dari alamat memori 10 ke register A.
- Instruksi `ADD`: Tambahkan data yang ada di dua register atau tambahkan data dalam register dengan nilai tertentu. Contohnya, `ADD A, B` akan menambahkan nilai di register A dan B, hasilnya disimpan di register A.
- Instruksi `STORE`: Gunakan instruksi `STORE` untuk menyimpan nilai dari register ke alamat memori tertentu. Misalnya, `STORE A, 20` akan menyimpan nilai di register A ke alamat memori 20.
- Loop atau Kondisi: Tambahkan instruksi untuk membuat perulangan atau kondisi (jika diperlukan) untuk memproses data secara berulang atau menentukan jalur eksekusi yang berbeda.
- Instruksi `HALT`: Gunakan instruksi `HALT` untuk menghentikan program setelah semua instruksi selesai dijalankan.

b. Menjalankan Program:

- Setelah program selesai ditulis, jalankan program pada CPU simulator. Amati setiap langkah eksekusi untuk memastikan instruksi bekerja sesuai dengan yang diinginkan.

  c. Kesimpulan:

- Dalam membuat program CPU, pemahaman tentang instruksi dasar seperti `LOAD`, `ADD`, `STORE`, dan `HALT` sangat penting karena instruksi tersebut merupakan dasar dari semua operasi yang dilakukan CPU.
- CPU simulator sangat berguna untuk memahami cara kerja CPU dan arsitektur komputer karena memungkinkan kita mengamati cara instruksi diproses dan data dipindahkan di dalam CPU tanpa menggunakan perangkat keras.
- Program sederhana ini membantu kita memahami bagaimana CPU mengambil data dari memori, melakukan operasi aritmatika, dan mengembalikan hasilnya ke memori.

Kesimpulan

Jadi, Kesimpulannya CPU simulator adalah alat yang sangat membantu dalam pembelajaran arsitektur komputer, khususnya untuk memahami operasi dasar CPU, fungsi register, dan pemanfaatan stack.

Penggunaan CPU simulator memungkinkan pengguna untuk melihat langsung bagaimana data diproses dan dimanipulasi dalam CPU, sehingga meningkatkan pemahaman tentang prinsip kerja perangkat keras tanpa memerlukan akses langsung ke komponen fisik.

Dengan memahami instruksi dasar, register, dan stack, pengguna dapat menulis program yang lebih efisien dan memahami cara kerja di dalam CPU, yang menjadi dasar penting dalam ilmu komputer dan pemrograman sistem.