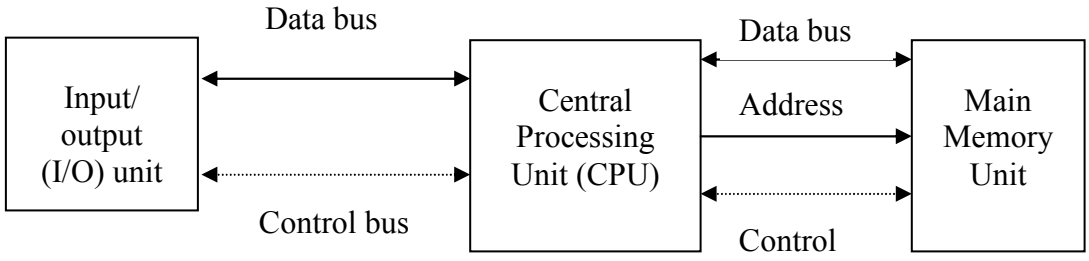
# CENTRAL PROCESSING UNIT (CPU)

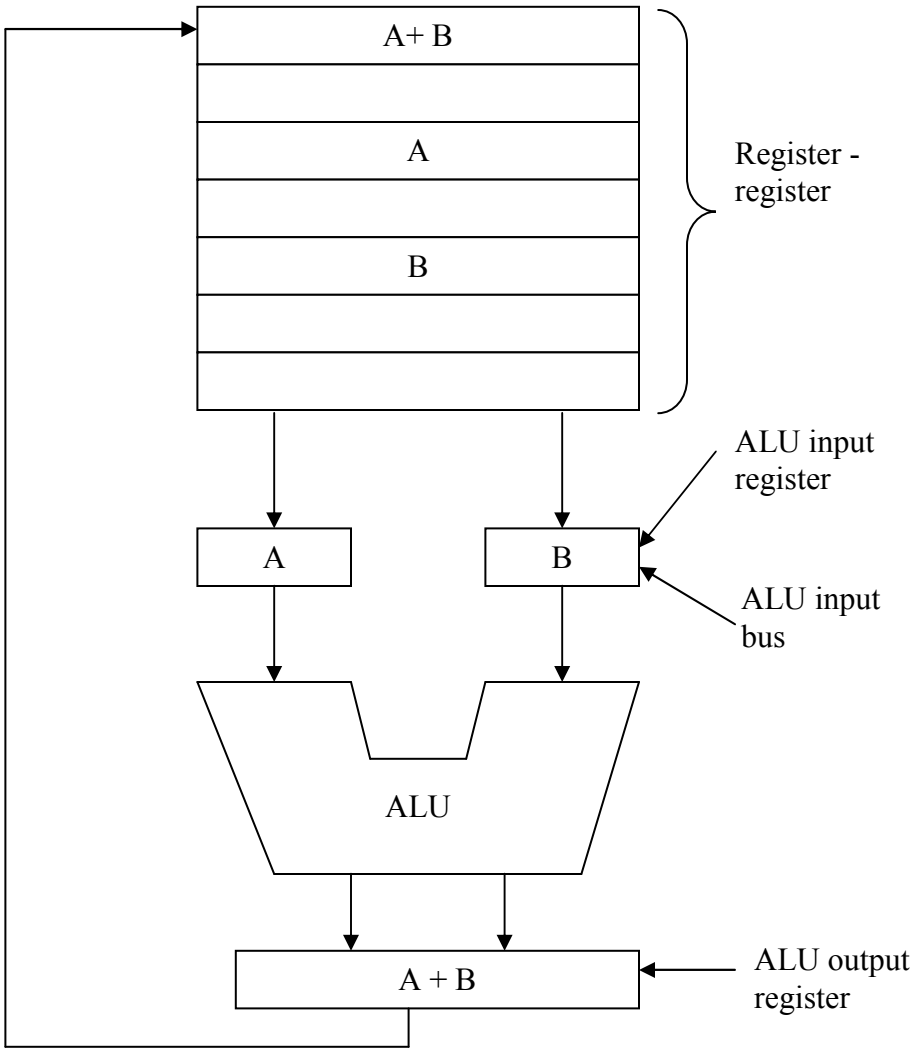
Arsitektur dasar mesin tipe von neumann menjadi kerangka referensi pada komputer digital umum (*general-purpose*) modern. 3 bagian fundamental tersebut adalah:




**Sebuah mesin tipe von neumann**

Program disimpan dalam unit memori utama yang berhadapan dengan piranti I/O melalui CPU. CPU membaca dari atau menulis ke memori, dengan mengirimkan alamat word ke unit memori melalui *bus address* kemudian menerima atau mengirimkan data melalui *bus data*. Data dipertukarkan antara CPU dan Unit I/O juga dengan menggunakan *bus data*. Operasi disinkronisasikan oleh dua *bus control* dengan sinyal kendali yang dikirimkan oleh CPU dan sinyal acknowledgment serta sinyal interupsi yang diterima oleh CPU.

**Organisasi CPU**




Gambar diatas disebut jalur data dan berisi register-register (terutama 1 sampai 32), ALU (Arithmetic Logic Unit) dan beberapa bus yang menghubungkan bagian-bagian tersebut. Register-register tersebut melengkapi dua register untuk input ALU, yang dalam gambar diberi label A dan B. Register-register ini menyimpan input ALU sementara ALU menjalankan fungsi perhitungan.

## KUMPULAN REGISTER

Register dari sebuah komputer secara kolektif disebut sebagai kumpulan register (*register set*).

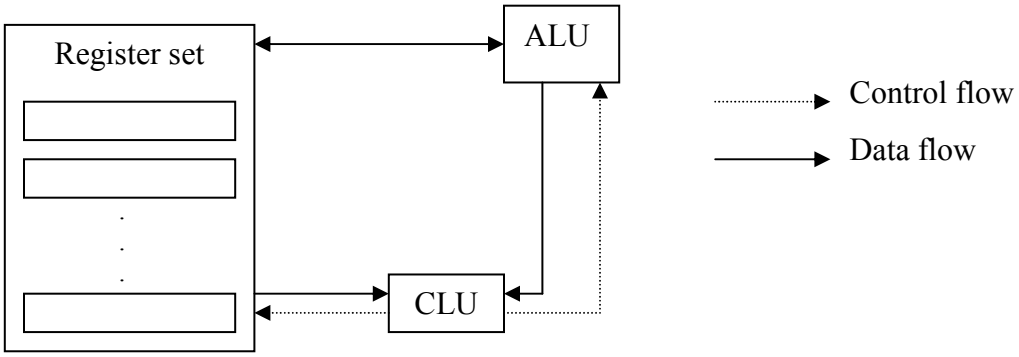
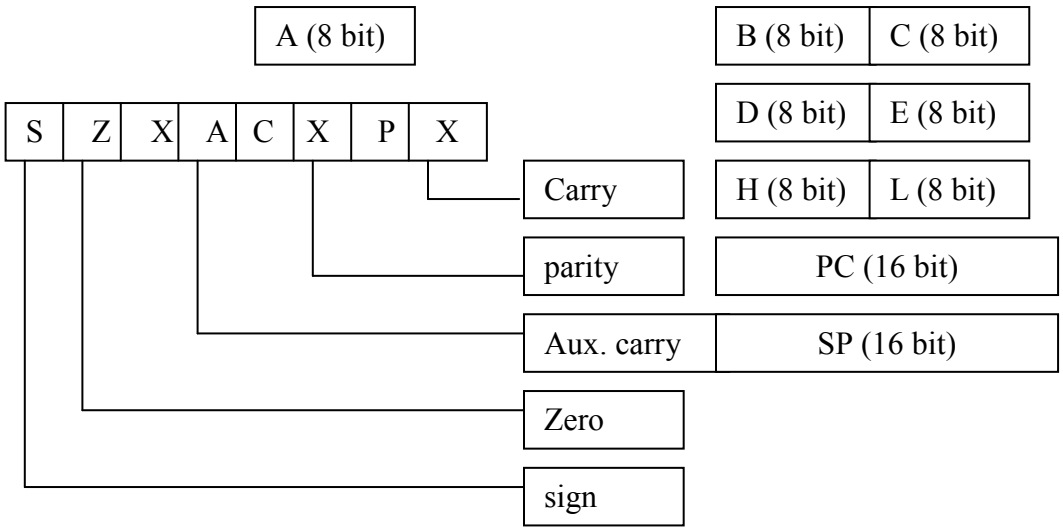



Diagram Blok Unit Pengolahan Pusat

Kumpulan register pada mikroprosesor intel 8085




(a.) Internal Registers


(b) Register pair organization

pada gambar (a) diatas menunjukkan kumpulan register pada mikroprosesor intel 8085. Pada CPU ini, register A berfungsi sebagai sebuah akumulator 8 bit. CPU juga mencakup sebuah *program counter* (PC), sebuah *stack pointer* (SP), sebuah *flag register* dan enam register pengalamatan 8 bit. Pada gambar (b) diatas, register 8 bit biasanya digunakan secara berpasangan. Register A bersama-sama dengan flag register, membentuk program status word (PSW). Tiga pasangan lainnya digunakan untuk tujuan pengalamatan, pasangan H merupakan pasangan yang sangat umum digunakan. Pasangan ini bisa dirujuk secara bersama-sama atau terpisah, yang menyebabkan tersedianya berbagai variasi intruksi.

**Format Instruksi**

Suatu Instruksi merupakan suatu tata cara yang digunakan oleh komputer untuk menyatakan operasi seperti ADD, STORE, LOAD, MOVE, dan BRANCH serta untuk menentukan lokasi data dimana suatu operasi akan dikerjakan. Kumpulan seluruh instruksi tersebut, disebut sebagai **kumpulan instruksi**.

**Format Alamat**

Pada salah satu komputer terdahulu, setiap instruksi terdiri atas sebuah opcode dan empat field alamat. Dimana :

| Opcode | A0 | A1 | A2 | A3 |
|---|---|---|---|---|

**Format empat alamat**

**A0 = Alamat operand pertama**

**A1 = Alamat operand kedua**

**A2 = Alamat dimana hasil operasi disimpan**

**A3 = Alamat dari instruksi berikutnya**

Karena komputer biasanya menjalankan intruksi secara berurutan, maka dapat memberi kode algoritma dengan cara tertentu dan menghilangkan kebutuhan akan A3. Jika dianggap bahwa panjang word memori tetap, maka dapat digunakan bit-bit yang memerinci A3 untuk sisa alamat yang ada dan dapat menggunakan ruang memori yang lebih besar tanpa meningkatkan ukuran word memori. Format ini dikenal dengan **format tiga-alamat** dimana:

A0 = alamat operand pertama

A1 = alamat operand kedua

A2 = alamat hasil

Format lain dikenal sebagai **format dua-alamat**, menghilangkan alamat A2 dan A3. Format ini merupakan format paling umum pada komputer komersial dan tergantung pada sistem tertentu, menggunakan salah satu dari akumulator A0 atau A1 untuk hasilnya. Bagaimanapun, cara termudah untuk mengorganisasikan sebuah komputer adalah dengan mempunyai sebuah register CPU tunggal dan kode instruksi dengan hanya dua bagian, format ini dikenal sebagai **format alamat-tunggal.** Disini akumulator menjalankan fungsi ganda: biasanya menjadi bagian alamat pada operand kedua dan juga lokasi dimana hasilnya disimpan.

**Mode Pengalamatan**

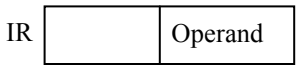
Suatu variasi mode pengalamatan (*addressing mode*) dapat digunakan untuk menentukan suatu alamat tempat untuk dimana operand akan di *fetch*. Beberapa teknik ini dapat meningkatkan kecepatan pelaksanaan instruksi dengan menurunkan jumlah referensi pada memori utama dan meningkatkan jumlah referensi

pada register kecepatan tinggi. Mode pengalamatan ini menjabarkan suatu aturan untuk menginterpretasikan atau memodifikasi field alamat dari instruksi sebelum operand direferensikan.

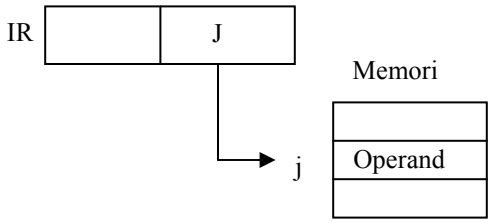
Beberapa mode pangalamatan umum diantaranya adalah :

| Mode | Nilai dari operand | Contoh transfer* |
|---|---|---|
| implied | Tidak ada operand dalam intruksi | |
| Immediate | Konstanta dalam field operand | OPR ← angka |
| Direct | Memori pada alamat | OPR ← M[ad] |
| Indirect | Memori pada alamat dalam alamat | OPR ← M(M[ad]) |
| Register | Register | OPR ← (R1) |
| Register-Indirect | Memori pada alamat register | OPR ←M[R1] |
| Autoincrement | Register, register increment | OPR ← (R1)<br>R1 ← (R1) +1 |
| Relative | Lokasi memori untuk PC juga alamat | OPR ← M[PC + ad] |
| Index | Lokasi memori untuk register indeks (XR) juga alamat | OPR ← M[XR + ad] |

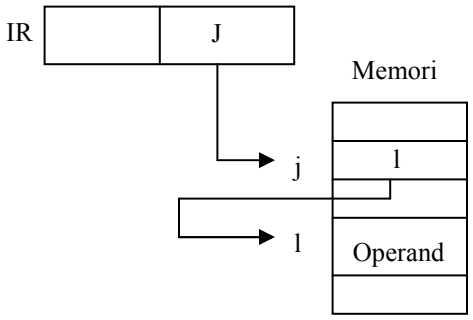
*OPR mewakili sebuah register untuk menyimpan operand yang akan digunakan sewaktu instruksi dijalankan.



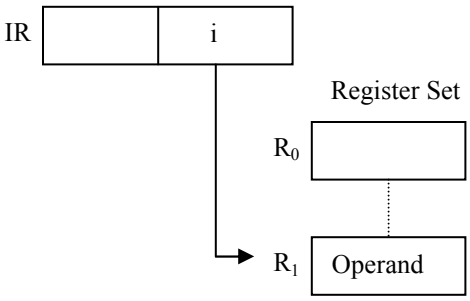

(a) immediate



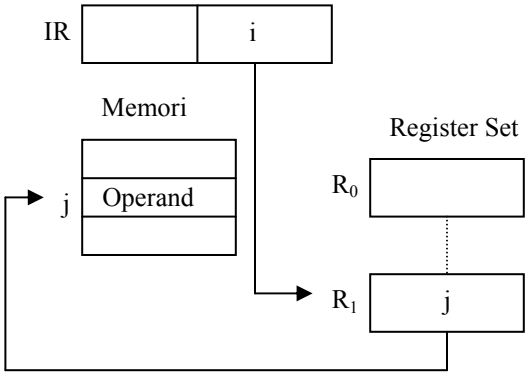

(b) Direct



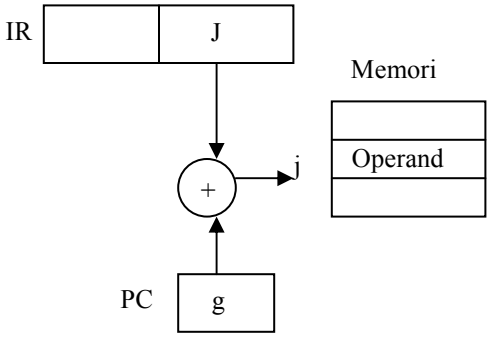

(c) Indirect




(d) Register




(e) Register Indirect

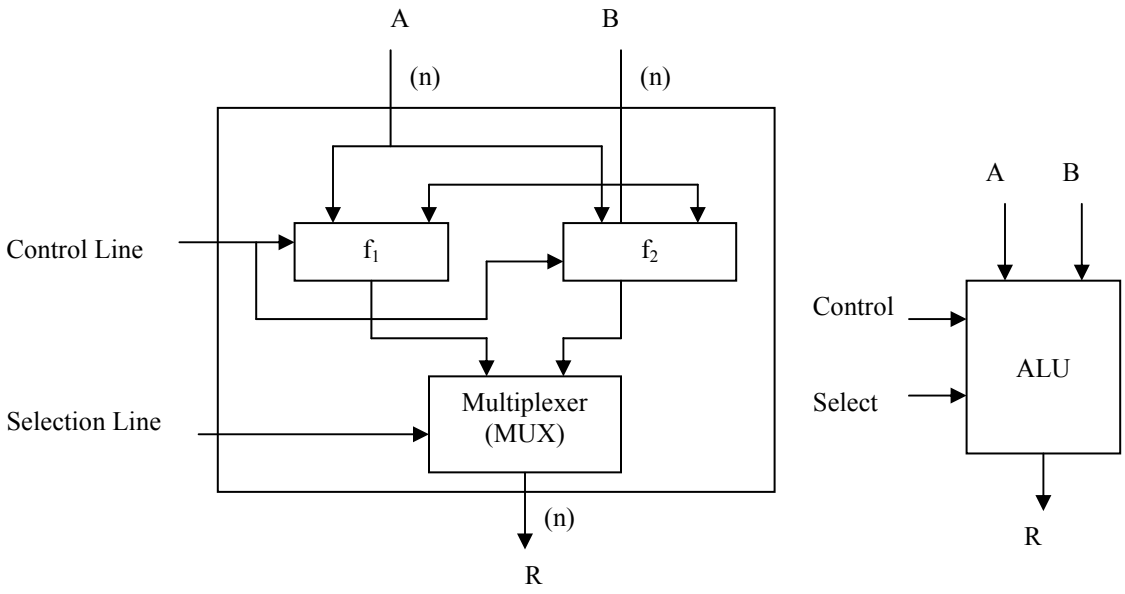



(f) Relative

**Gambar Mode alamat**

## ARITHMETIC AND LOGIC UNIT (ALU)

Ide mengenai satu adder umum yang mampu menambahkan dua register bersama-sama dan menyimpan hasilnya dalam register lainnya merupakan prinsip yang mendasar pada ALU. Sehingga ALU didefinisikan sebagai sebuah unit yang berisi sirkuit untuk menjalankan sekumpulan operasi mikro aritmatika dan logika. Sebuah contoh dari dua fungsi ALU ditunjukkan sebagai berikut,



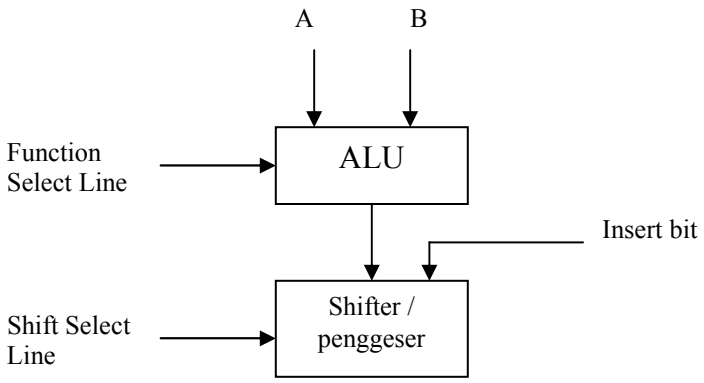

(a) internal organization (b) Block diagram

Sejumlah n baris input dari A dan B dihubungkan dengan blok fungsi $f_1$ dan $f_2$. Kemudian sejumlah n baris output pada blok tersebut dihubungkan dengan sejumlah n multiplexer (MUX). Tergantung dari operasi mikro tertentu yang harus dijalankan maka baris seleksi akan di-*set* untuk memilih baris output fungsi yang semestinya untuk sejumlah n baris dari R, yaitu hasil operasi ALU. Jumlah baris seleksi yang diperlukan tergantung pada jumlah fungsi di dalam ALU, pada bagian ini ada dua input n-bit, yaitu A dan B, dan sebuah output n-bit, yaitu R.

**Fungsi Aritmatika pada sebuah ALU** biasanya mencakup integer, floating-point (real) dan desimal berkode biner. Disini operasi yang terjadi adalah penambahan, pengurangan, perkalian dan pembagian.

**Fungsi Logika pada ALU** lebih sederhana. Untuk segala operasi logika yang ingin diterapkan, maka hanya perlu memuat sejumlah n gerbang logika tertentu untuk operasi tersebut (satu untuk setiap pasangan bit input).

Selain itu pula ALU dapat digunakan sebagai **Pergeseran**, dengan menerapkan sirkuit geser kombinasional yang dikenal sebagai skalar posisi. Karena kita ingin menjalankan pergeseran bersamaan dengan fungsi aritmatika atau logika, seperti pada perkalian atau pengepakan string, maka akan lebih efisien untuk men-*set* penggeser diluar ALU. Dengan cara ini dapat ditambahkan dua angka dan menggeser seluruh hasil dalam satu langkah daripada meneruskan hasilnya ke input ALU lagi dan kemudian men*setup* ALU untuk menggeser angka tersebut. Terlihat pada gambar dibawah ini:




**Gambar Konfigurasi ALU-Penggeser**

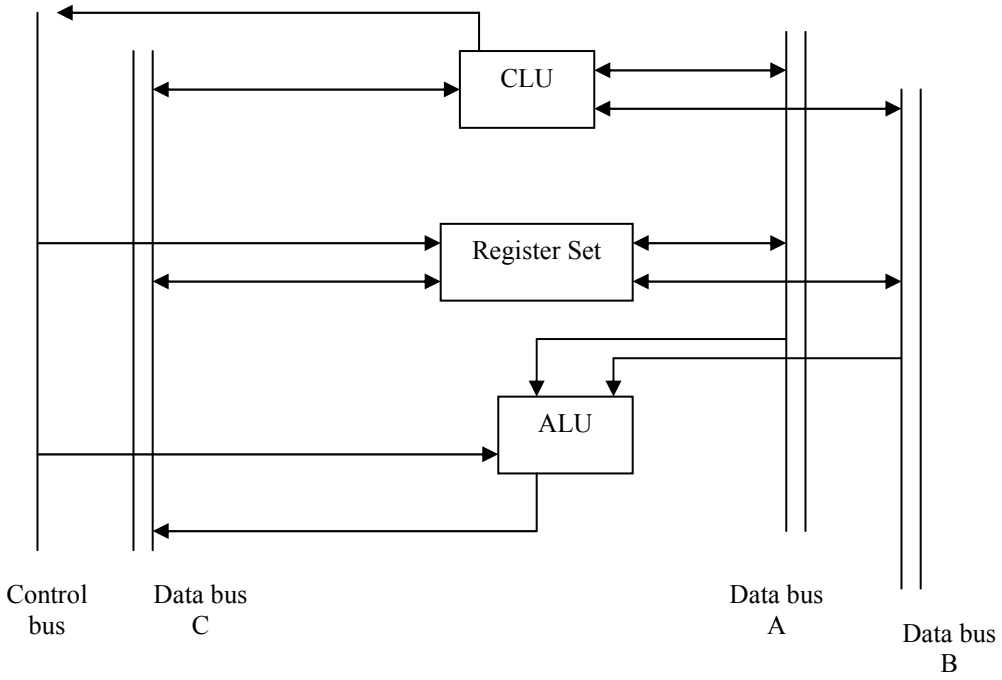
## CONTROL LOGIC UNIT (CLU)

CLU pada komputer memasukkan informasi tentang instruksi dan mengeluarkan baris kendali yang diperlukan untuk mengaktifkan operasi-mikro yang semestinya. CLU terbentuk atas sebuah prosesor instruksi (*IP* atau *instruction processor*) yang berfungsi untuk mengendalikan *fetch*, perhitungan alamat dan siklus interupsi, kemudian prosesor aritmatika (*AP* atau *arithmatic processor*) yang berfungsi untuk mengendalikan siklus eksekusi bagi operasi aritmatika dan logika.

### Konfigurasi CPU

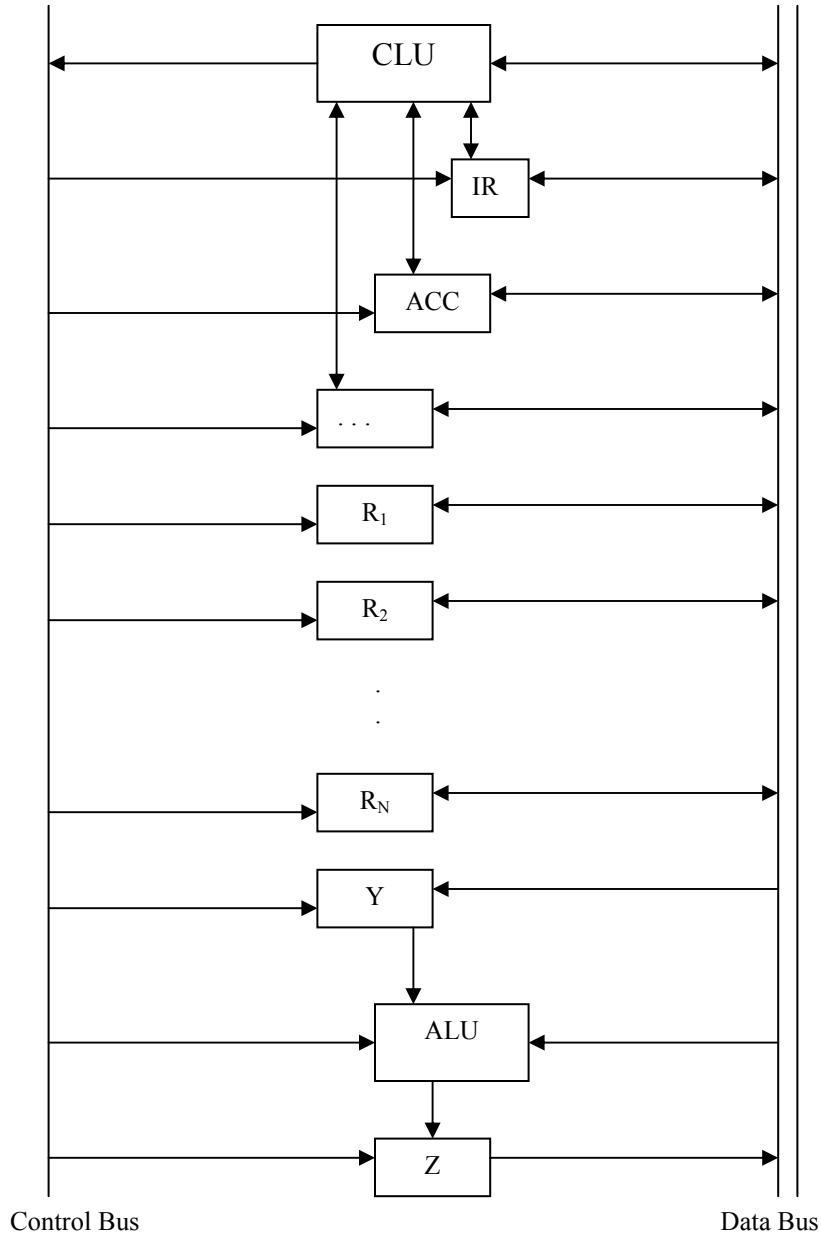
Komponen CPU dapat tersusun dalam berbagai cara, sangat tergantung pada jumlah bus data internal yang digunakan. Dua contoh diantaranya adalah Organisasi bus tunggal dan organisasi triple bus.

### Organisasi triple bus



Penggunaan tiga bus data internal, seperti gambar diatas akan melonggarkan beberapa batasan yang dibebankan oleh susunan bus-tunggal. Dalam hal ini, bus-bus yang terpisah dapat digunakan untuk dua input ALU termasuk juga untuk output ALU. Jika register dari kumpulan register adalah edge-trigerred, maka akan mungkin untuk menjalankan jenis operasi-mikro R1 ← (R2) + (R3) pada satu sinyal waktu.

**Organisasi Bus-Tunggal**.



ALU memerlukan input register Y dan register Z secara bersamaan. Dengan hanya sebuah bus data tunggal, sebuah operand akan disimpan dalam Y dan yang lainnya dapat disimpan dalam bus. Sewaktu ALU menghitung hasilnya, input tersebut harus tetap konstan pada bus. Karena itu, kadang-kadang hasilnya disimpan dalam Z sampai operasi selesai dan kemudian ditransfer melalui bus ke tempat dimana harus disimpan. Dalam hubungan yang sama, CLU memerlukan informasi dari register khusus (*special-purpose*) secara bersamaan untuk menghasilkan fungsi pengendalian yang tepat. Oleh karena itu, register-register tersebut dihubungkan secara langsung ke CLU selain dihubungkan dengan bus data untuk komunikasi umum.