Organisasi Sistem Komputer

Bab 1. Arsitektur Komputer dan Representasi Bilangan

1.1 Sejarah Komputer

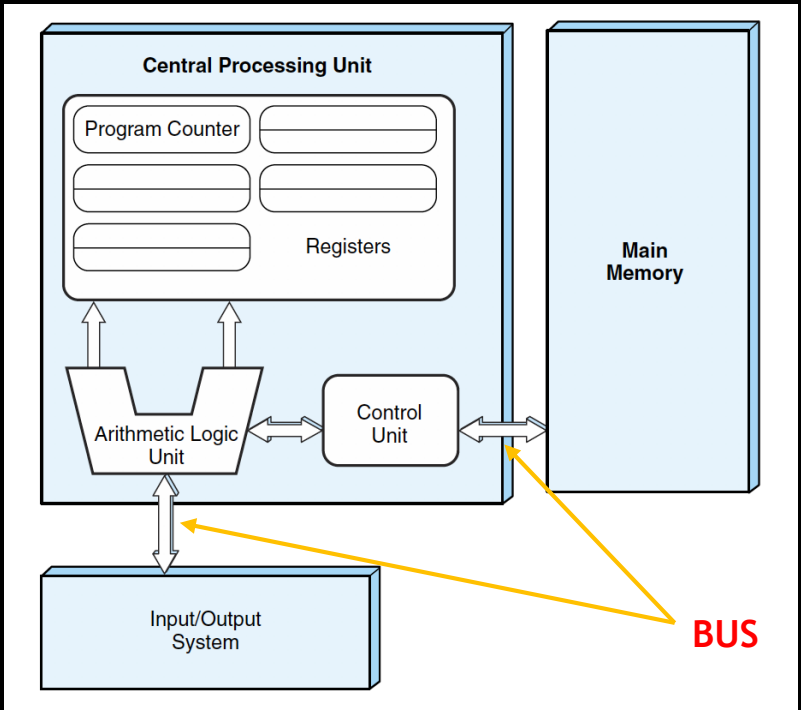
1.2 Arsitektur Komputer

1.3 Representasi Bilangan

Pembahasan:

- Model von Neumann
- Memori dan CPU
- Siklus Fetch-Decode-Execute
- Clock
- Instruction Set Architecture

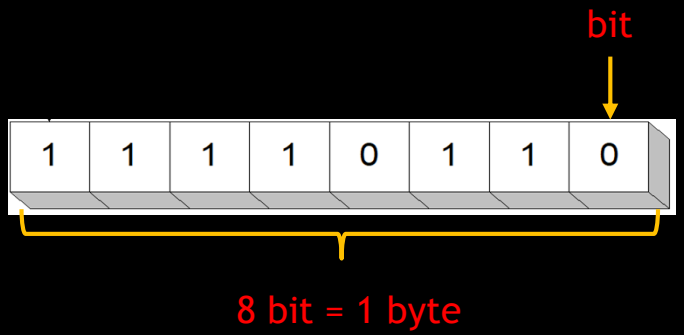
Model von Neumann



- Arsitektur komputer yang digunakan hingga sekarang hampir semuanya berdasarkan pada model von Neumann
- Tiga komponen utama komputer:
  - **CPU** yang melakukan pekerjaan kalkulasi dan mengatur semua yang terjadi di komputer
  - **Memori** yang berisi kode dan data
  - **Input dan Ouput** untuk menerima informasi dan mengirimkan informasi
- Tiga komponen ini berkomunikasi satu sama lain melalui **bus**

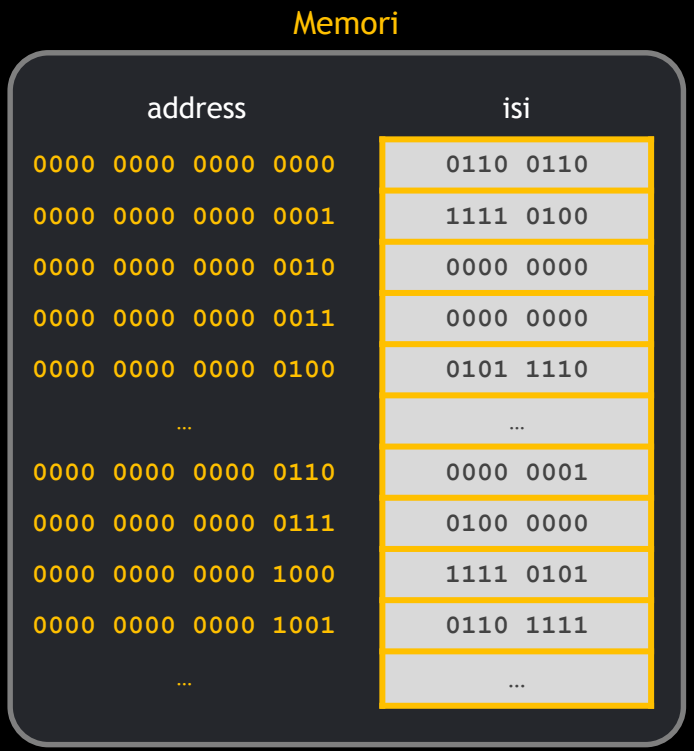
Informasi dalam Komputer

- Semua “informasi” dalam komputer berbentuk **biner** (*binary*)
  - 0: voltase 0
  - 1: voltase positif (5V)
- Unit informasi terkecil yang disimpan dalam memori disebut dengan **byte** yang merupakan pengelompokkan 8 bit



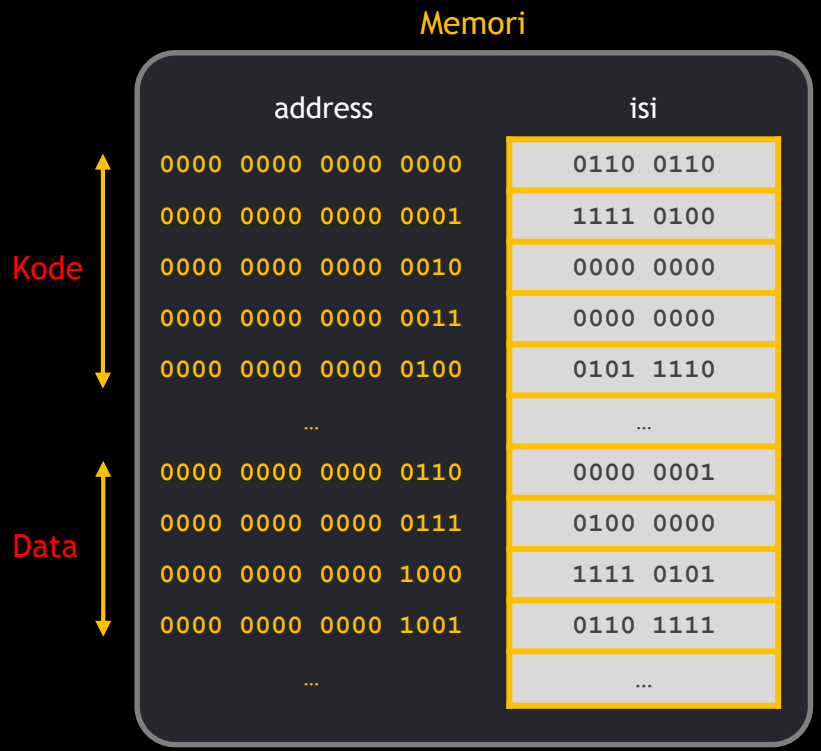
Memori

- Memori dapat dibayangkan sebagai kotak-kotak penyimpanan informasi yang disusun berderet
- Setiap kotak berkapasitas satu byte dan dilabeli dengan *address* (alamat) unik
- Address memori direpresentasikan dalam biner dengan jumlah bit sesuai dengan arsitektur CPU:
  - Pada CPU 32-bit, address memori direpresentasikan dalam 32-bit biner (ini kenapa sistem 32 bit mempunyai kapasitas memori maksimum 4 GB)
  - Pada CPU 64-bit, address memori direpresentasikan dalam 64-bit biner



Kode dan Data dalam Memori

- Sebuah program terdiri dari dua bagian: ***data*** dan ***kode***
  - **Data**: nilai-nilai
  - **Kode**: instruksi-instruksi
- Ketika program dijalankan, kode dan data program dimuat ke dalam memori
- CPU tidak dapat membedakan kode dan data
- Program yang harus memberitahu di alamat mana tersimpan data dan di alamat mana tersimpan kode



CPU

Program Counter

- Berisi alamat memori dari instruksi yang akan dijalankan
- Di-inkrementasi setelah setiap instruksi, dan dapat juga diubah nilainya untuk merubah alur program

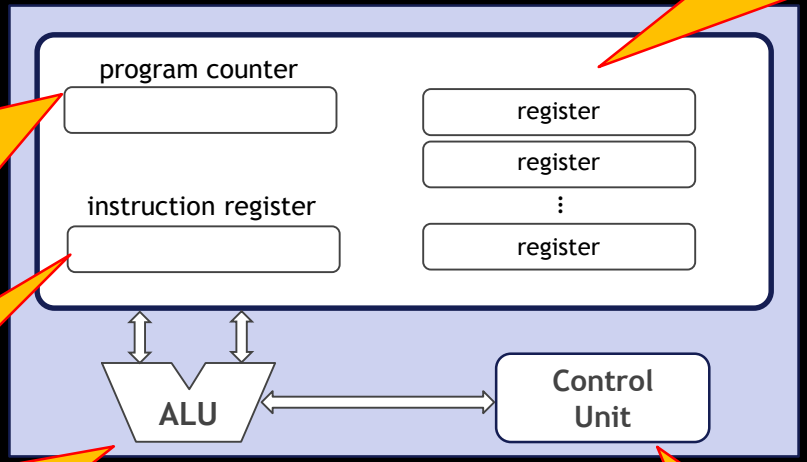
Instruction Register

Menyimpan kode instruksi yang akan dieksekusi

Arithmetic and Logic Unit

Melakukan komputasi aritmatika dan logika

+, -, *, /, OR, AND, XOR, dsb.



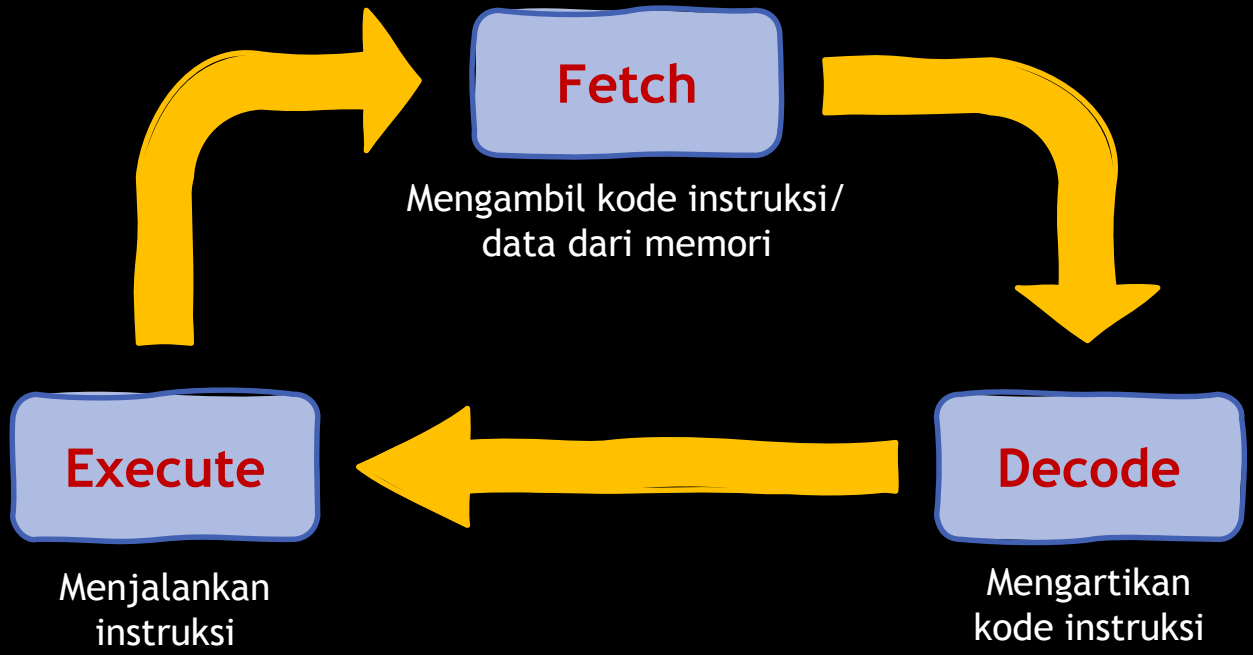
Register

Tempat penyimpanan data sementara sebelum data diproses

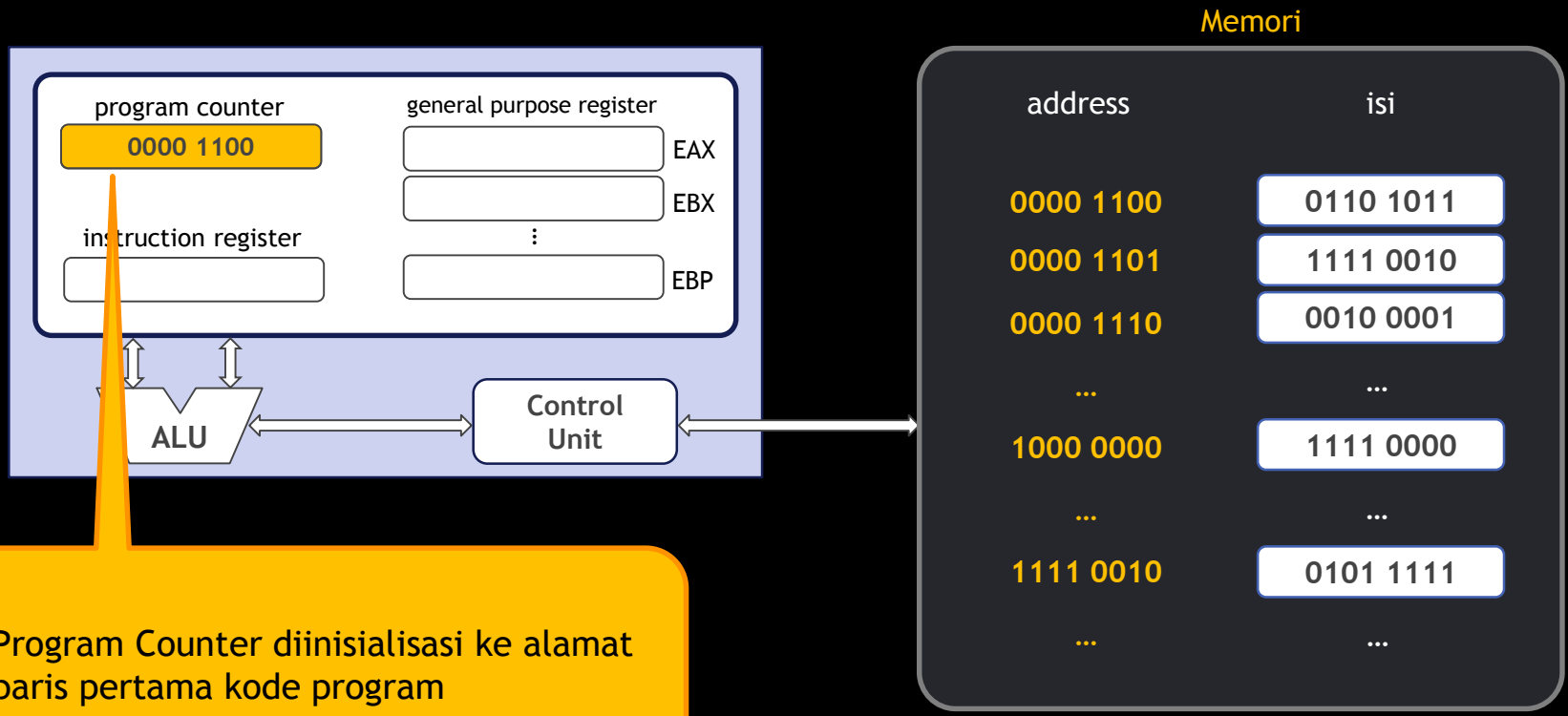
Control Unit

Menerjemahkan instruksi dan mengatur semua yang terjadi pada prosesor.

Siklus Fetch-Decode-Execute

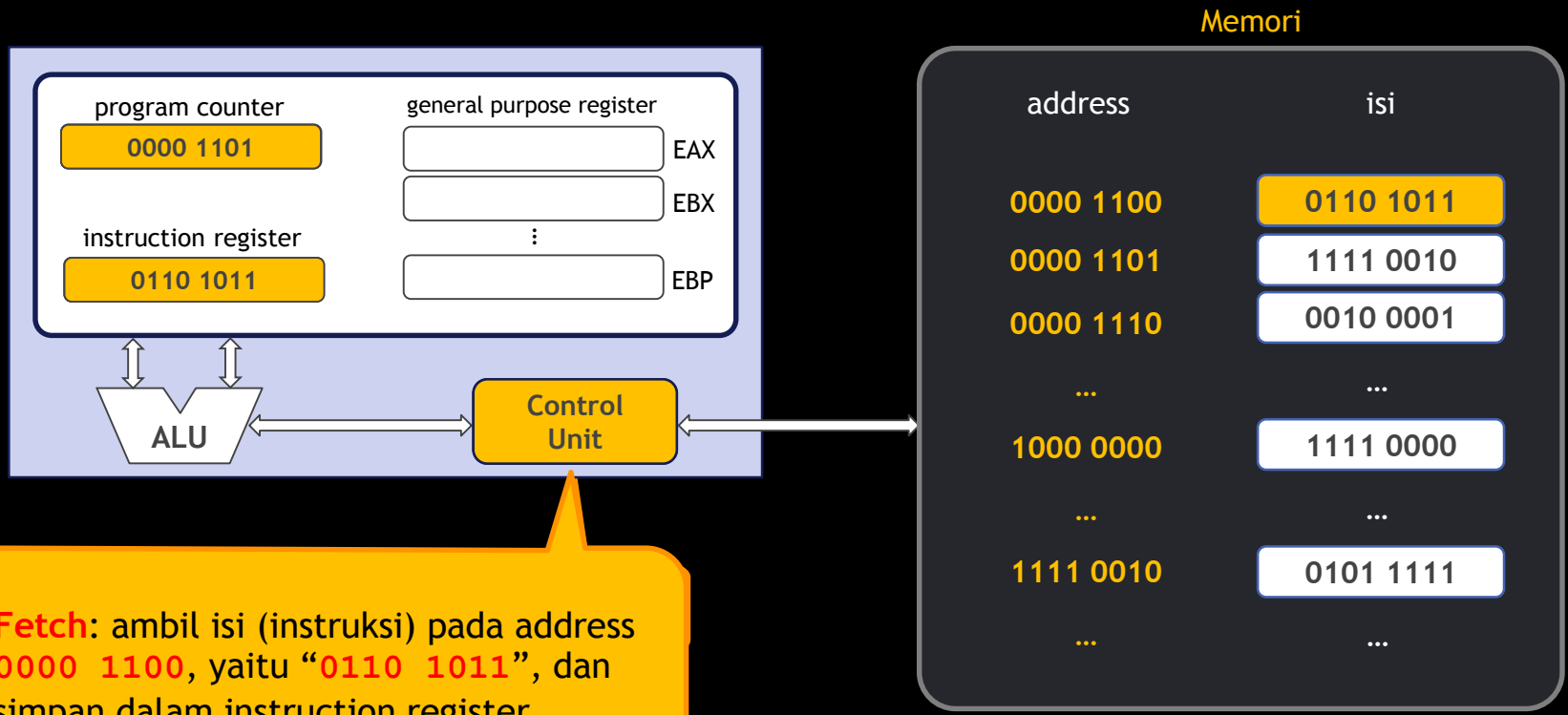


Inisialisasi



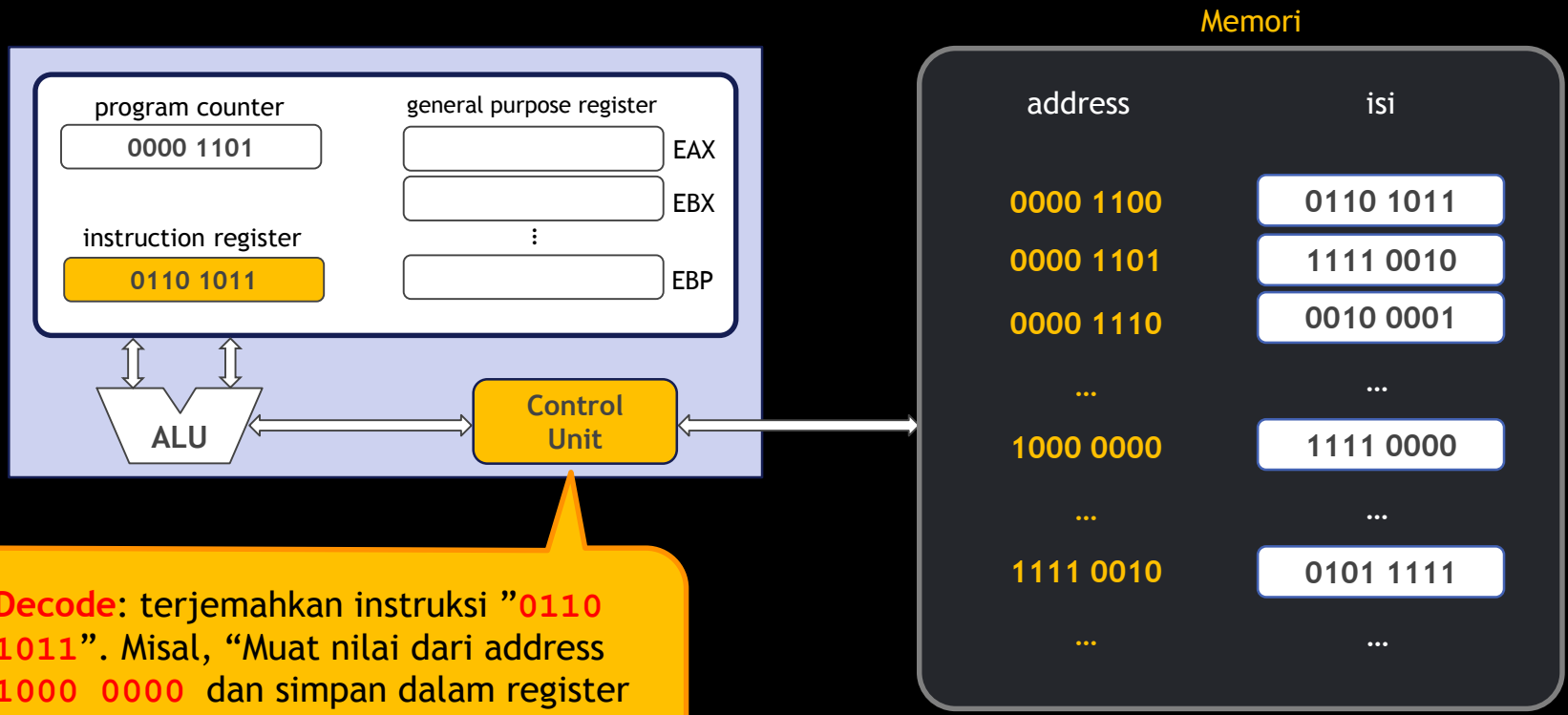
Program Counter diinisialisasi ke alamat baris pertama kode program

Fetch 1



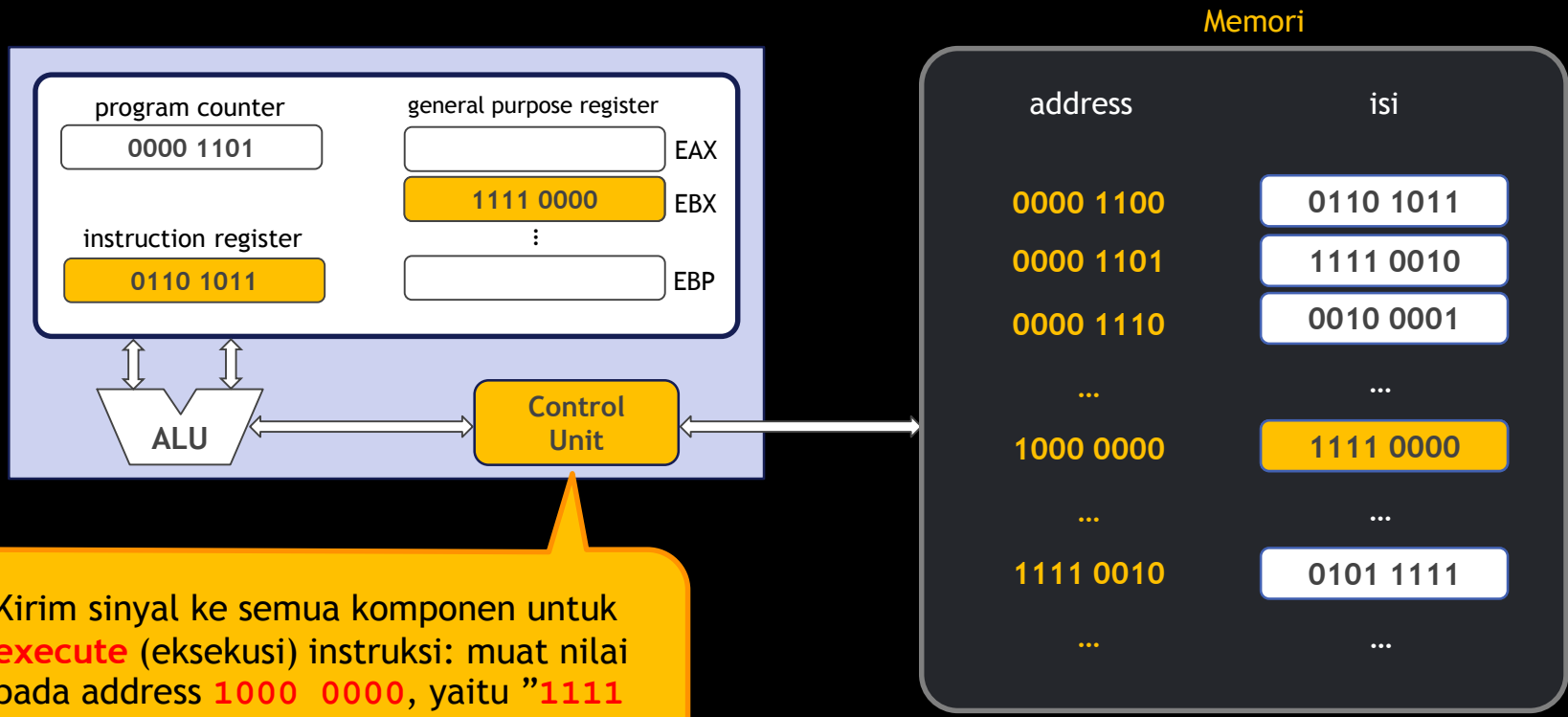
Fetch: ambil isi (instruksi) pada address `0000 1100`, yaitu "`0110 1011`", dan simpan dalam instruction register

Decode 1



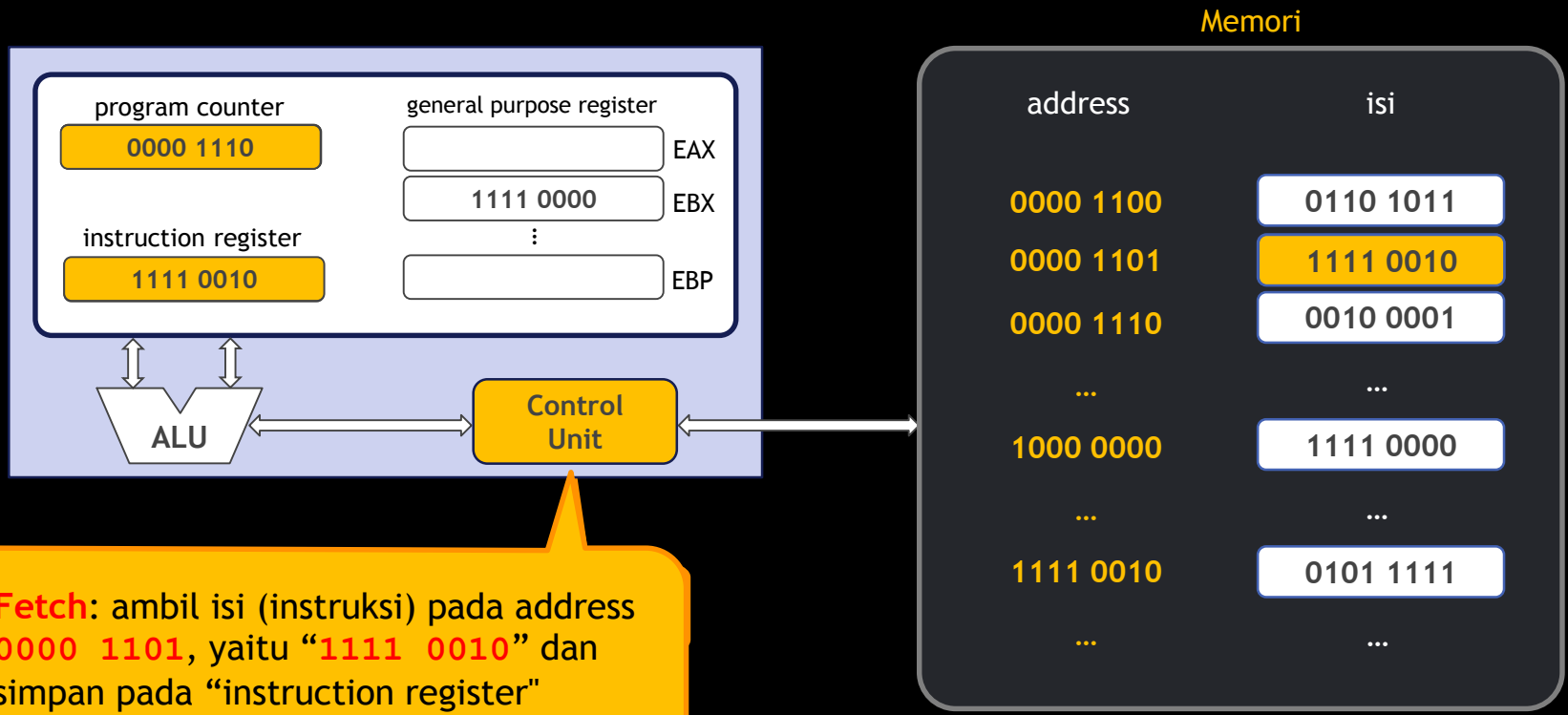
Decode: terjemahkan instruksi ”0110 1011”. Misal, “Muat nilai dari address 1000 0000 dan simpan dalam register EBX”

Execute 1



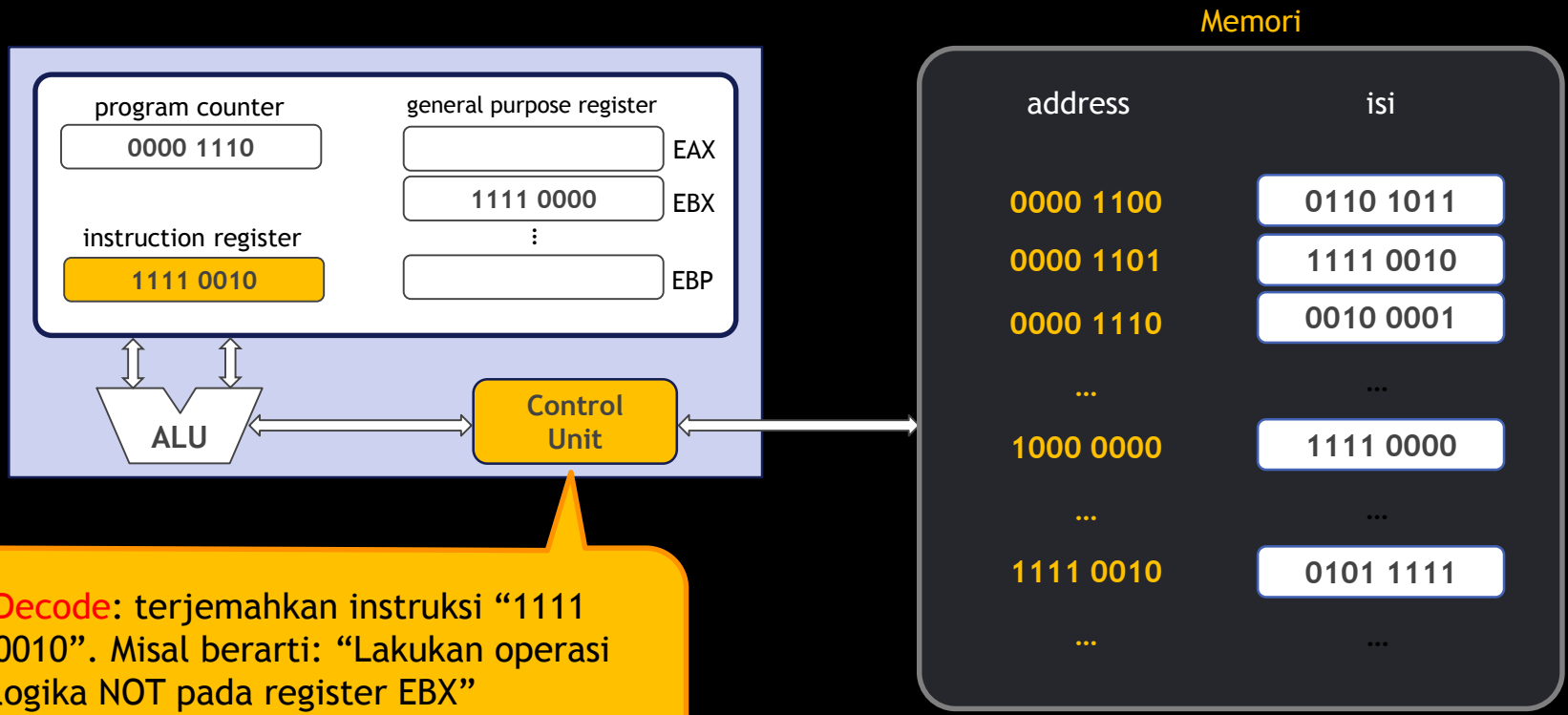
Kirim sinyal ke semua komponen untuk **execute** (eksekusi) instruksi: muat nilai pada address `1000 0000`, yaitu "`1111 0000`" dan simpan dalam register kedua

Fetch 2



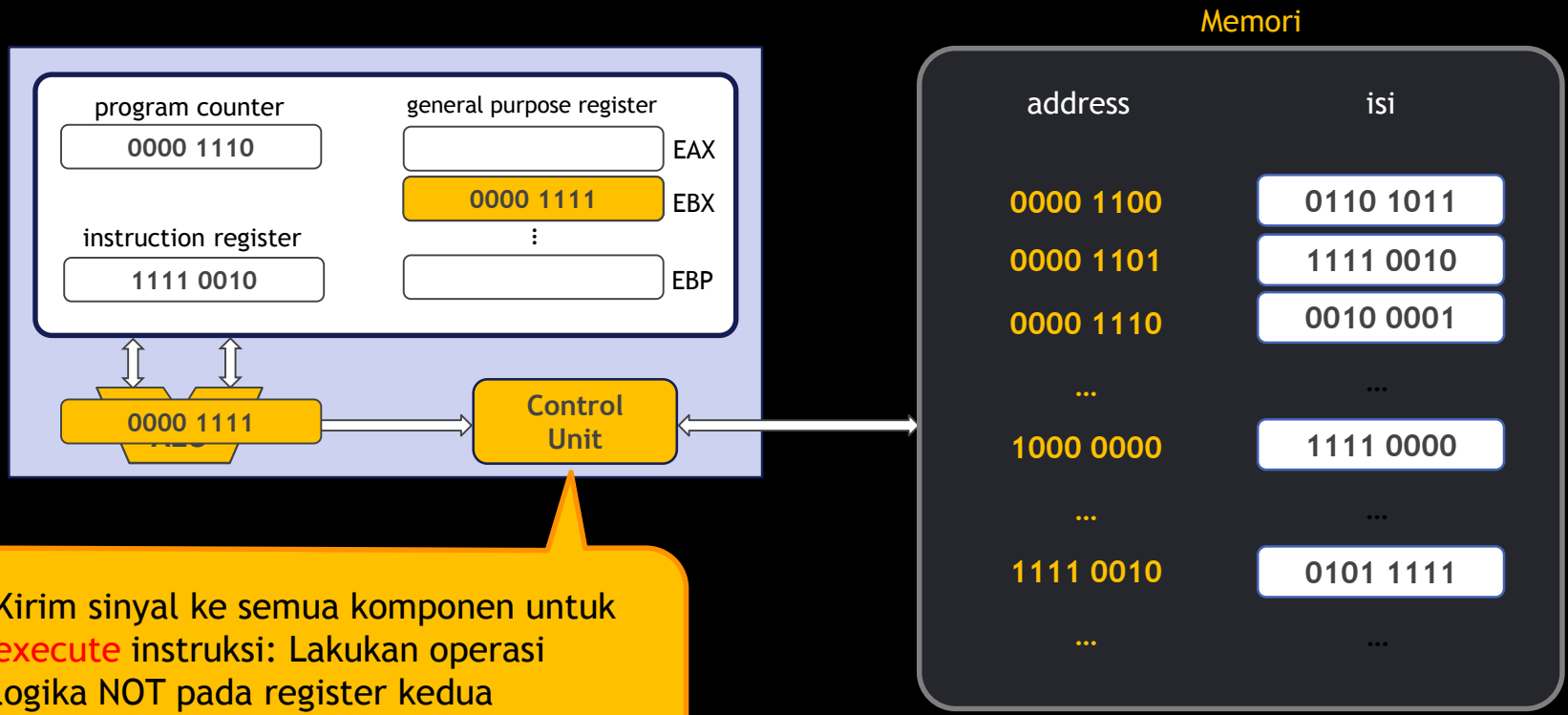
Fetch: ambil isi (instruksi) pada address `0000 1101`, yaitu "`1111 0010`" dan simpan pada "instruction register"

Decode 2



Decode: terjemahkan instruksi “1111 0010”. Misal berarti: “Lakukan operasi logika NOT pada register EBX”

Execute 2



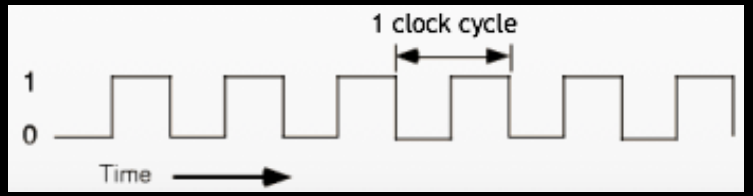
Kirim sinyal ke semua komponen untuk execute instruksi: Lakukan operasi logika NOT pada register kedua

Siklus Fetch-Decode-Execute

- Siklus Fetch-Decode-Execute hanya gambaran yang disederhanakan dari cara kerja computer
- Komputer modern saat ini:
  - CPU memiliki lebih dari satu Control Unit dan ALU
  - CPU memiliki cache untuk menyimpan beberapa instruksi
  - CPU juga terdiri lebih dari satu core (ini berarti beberapa CPU ditanam dalam satu chip)
  - Eksekusi instruksi di-*pipeline*: saat satu instruksi di-fetch, instruksi lain dieksekusi
- Namun, secara konsep cara kerja komputer hingga kini masih mengikuti siklus fetch-decode-execute

Clock

- Setiap komputer mempunyai clock internal yang mengatur seberapa cepat instruksi dapat dieksekusi dan mensinkronisasi komponen-komponen sistem



- Setiap “event” terjadi pada satu clock cycle, namun biasanya satu instruksi CPU memerlukan lebih dari satu clock cycle
- Frekuensi dari clock disebut dengan clock rate yang diukur dalam satuan Hz
- Clock cycle = 1 / clock rate
  - Clock rate = 2.4 GHz
  - Clock cycle = 1 / (2.4 * 1,000,000,000) = 0.416 ns

Clock

- Semakin tinggi clock rate, semakin pendek clock cycle
- Namun tidak berarti semakin tinggi clock rate semakin cepat komputer
- Kecepatan komputer ditentukan berapa banyak instruksi yang dikerjakan dalam clock cycle:
  - CPU Intel: clock rate 2 GHz dan satu perkalian membutuhkan 10 cycle
  - CPU AMD: clock rate 1.5 GHz dan satu perkalian membutuhkan 5 cycle
  - CPU AMD lebih cepat dari CPU Intel untuk menjalankan sebuah program yang melakukan banyak perkalian
- Jadi, clock rate tidak bisa dijadikan patokan untuk membandingkan kecepatan komputer terutama antar keluarga CPU berbeda

Instruction Set Architecture (ISA)

- Masing-masing arsitektur CPU memiliki set instruksi sendiri
  - CPU Intel mempunyai set instruksi berbeda dengan CPU ARM
- Set instruksi ini disebut dengan ISA: Instruction Set Architecture
- Spesifikasi ISA untuk Intel x86 dapat dilihat di http://ref.x86asm.net/

Instruction Set Architecture (ISA)

pf	0F	po	so	o	proc	st	m	rl	x	mnemonic	op1	op2	op3	op4	iext	tested f	modif f	def f	undef f	f values	description, notes
		00		r					L	ADD	**r/m8**	r8					o..szapc	o..szapc			Add
		01		r					L	ADD	**r/m16/32**	r16/32					o..szapc	o..szapc			Add
		02		r						ADD	**r8**	r/m8					o..szapc	o..szapc			Add
		03		r						ADD	**r16/32**	r/m16/32					o..szapc	o..szapc			Add
		04								ADD	**AL**	imm8					o..szapc	o..szapc			Add
		05								ADD	**eAX**	imm16/32					o..szapc	o..szapc			Add
		06								PUSH	ES										Push Word, Doubleword or Quadword Ont
		07								POP	**ES**										Pop a Value from the Stack
		08		r					L	OR	**r/m8**	r8					o..szapc	o..sz.pc	.....a..	o......c	Logical Inclusive OR
		09		r					L	OR	**r/m16/32**	r16/32					o..szapc	o..sz.pc	.....a..	o......c	Logical Inclusive OR
		0A		r						OR	**r8**	r/m8					o..szapc	o..sz.pc	.....a..	o......c	Logical Inclusive OR
		0B		r						OR	**r16/32**	r/m16/32					o..szapc	o..sz.pc	.....a..	o......c	Logical Inclusive OR
		0C								OR	**AL**	imm8					o..szapc	o..sz.pc	.....a..	o......c	Logical Inclusive OR
		0D								OR	**eAX**	imm16/32					o..szapc	o..sz.pc	.....a..	o......c	Logical Inclusive OR
		0E								PUSH	ES										Push Word, Doubleword or Quadword Ont
		0F			02+					*Two-byte instructions*											

operands

opcode dalam Hex

mnemonic

Ringkasan

- Komputer saat ini masih didesain dan bekerja berdasarkan prinsip model von Neuman
- Dalam model von Neumann, komputer terdiri dari tiga komponen utama: CPU, Memori, dan Input/Output
- Model von Neuman bekerja dalam siklus fetch-decode-execute:
  - Fetch: mengambil instruksi
  - Decode: menerjemahkan instruksi
  - Execute: mengeksekusi instruksi
- CPU mengerjakan sesuatu berdasarkan sinyal dari clock
- Setiap arsitektur CPU mempunyai set instruksi (ISA) tersendiri