Nama : Adi Nugroho
Nim : L200150120
Kelas : B

Set Instruksi (Instruction Set)

Set Instruksi (Instruction Set atau Instruction Set Architecture (ISA)) didefinisikan sebagai suatu aspek dalam arsitektur komputer yang dapat dilihat oleh para pemrogram. Secara umum, ISA ini mencakup jenis data yang didukung, jenis instruksi yang dipakai, jenis register, mode pengalamatan, arsitektur memori, penanganan interupsi, eksepsi, dan operasi I/O eksternalnya (jika ada). ISA merupakan sebuah spesifikasi dari kumpulan semua kode-kode biner (opcode) yang diimplementasikan dalam bentuk aslinya (native form) dalam sebuah desain prosesor tertentu. Kumpulan opcode tersebut, umumnya disebut sebagai bahasa mesin (machine language) untuk ISA yang bersangkutan. ISA yang populer digunakan adalah set instruksi untuk chip Intel x86, IA-64, IBM PowerPC, Motorola 68000, Sun SPARC, DEC Alpha, dan lain-lain.
ISA kadang-kadang digunakan untuk membedakan kumpulan karakteristik yang disebut di atas dengan mikroarsitektur prosesor, yang merupakan kumpulan teknik desain prosesor untuk mengimplementasikan set instruksi (mencakup microcode, pipeline, sistem cache, manajemen daya, dan lainnya).

Komputer-komputer dengan mikroarsitektur berbeda dapat saling berbagi set instruksi yang sama. Sebagai contoh, prosesor Intel Pentium dan prosesor AMD Athlon mengimplementasikan versi yang hampir identik dari set instruksi Intel x86, tetapi jika ditinjau dari desain internalnya, perbedaannya sangat radikal. Konsep ini dapat diperluas untuk ISA-ISA yang unik seperti TIMI yang terdapat dalam IBM System/38 dan IBM IAS/400. TIMI merupakan sebuah ISA yang diimplementasikan sebagai perangkat lunak level rendah yang berfungsi sebagai mesin virtual. TIMI didesain untuk meningkatkan masa hidup sebuah platform dan aplikasi yang ditulis untuknya, sehingga mengizinkan platform tersebut agar dapat dipindahkan ke perangkat keras yang sama sekali berbeda tanpa harus memodifikasi perangkat lunak (kecuali yang berkaitan dengan TIMI). Hal ini membuat IBM dapat memindahkan platform AS/400 dari arsitektur mikroprosesor CISC ke arsitektur mikroprosesor POWER tanpa harus menulis ulang bagian-bagian dari dalam sistem operasi atau perangkat lunak yang diasosiasikan dengannya. Ketika mendesain mikroarsitektur, para desainer menggunakan Register Transfer Language (RTL) untuk mendefinisikan operasi dari setiap instruksi yang terdapat dalam ISA.

Sebuah ISA juga dapat diemulasikan dalam bentuk perangkat lunak oleh sebuah interpreter. Karena terjadi translasi tambahan yang dibutuhkan untuk melakukan emulasi, hal ini memang menjadikannya lebih lambat jika dibandingkan dengan menjalankan program secara langsung di atas perangkat keras yang mengimplementasikan ISA tersebut. Akhir-akhir ini, banyak vendor ISA atau mikroarsitektur yang baru membuat perangkat lunak emulator yang dapat digunakan oleh para pengembang perangkat lunak sebelum implementasi dalam bentuk perangkat keras dirilis oleh vendor.
Daftar ISA di bawah ini tidak dapat dikatakan komprehensif, mengingat banyaknya arsitektur lama yang tidak digunakan lagi saat ini atau adanya ISA yang baru dibuat oleh para desainer.

ISA yang diimplementasikan dalam bentuk perangkat keras :
Alpha AXP (DEC Alpha)
ARM (Acorn RISC Machine) (Advanced RISC Machine now ARM Ltd)
IA-64 (Itanium/Itanium 2)
MIPS
Motorola 68k
PA-RISC (HP Precision Architecture)
IBM POWER
IBM PowerPC
SPARC
SuperH (Hitachi)
System/360
Tricore (Infineon)
Transputer (STMicroelectronics)
VAX (Digital Equipment Corporation)
x86 (IA-32, Pentium, Athlon) (AMD64, EM64T)

ISA yang diimplementasikan dalam bentuk perangkat lunak lalu dibuat perangkat kerasnya :
p-Code (UCSD p-System Version III on Western Digital Pascal Micro-Engine)
Java virtual machine (ARM Jazelle, PicoJava)
FORTH

ISA yang tidak pernah diimplementasikan dalam bentuk perangkat keras :
SECD machine
ALGOL Object Code

Elemen - elemen dari instruction
Instruction terdiri dari beberapa elemen yaaitu :
- Operation Code (Op Code)
- kode perintah operasi
- Source Operand reference
- Operand penampung nilai yang akan diproses.
- Result Operand reference
- Operand penampung nilai hasil proses.
- Next Instruction Reference
- Penghubung ke instruksi berikutnya.

Operand merupakan obyek dari suatu Op code, operand biasanya ditampung pada salah satu tempat penyimpanan berikut:
- Main memory (or virtual memory or cache)
- CPU register
- I/O device

Mode Pengalamatan (Addressing Mode) untuk Operand
Terdapat beberapa mode pengalamatan operand, antara lain:
- Immediate
- Direct
- Indirect
- Register

Mode Immediate Addressing
Merupakan mode pengalamatan operand secara langsung, pada mode ini operand merupakan bagian dari instruction.
Operand merupakan area alamat (address field) dari suatu nilai yang akan diproses

Contoh: ADD 5
Keterangan: Tambahkan nilai 5 dengan nilai di register accumulator dan simpan hasilnya di register accumulator.

Karakteristik mode immediate:
- No memory reference to fetch data tidak memakai referensi memory untuk mengambil data
- Fast : cepat
- Limited range : terbatas dalam jangkauan nilai

Mode Direct Addressing
Merupakan mode pengalamatan operand dimana area alamat (address field) berisi alamat dari suatu nilai yang akan diproses.
Effective Address (EA) = Address field (A)
EA = A
Contoh: ADD A
Keterangan:
- Cari di memory pada alamat A untuk operand (Look in memory at address A for operand).
- Tambahkan isi yang ada pada alamat A dengan nilai di register accumulator dan simpan hasilnya di register accumulator. (Add contents of cell A to accumulator).
Karakteristik:
- Single memory reference to access data : Menggunakan memory untuk mengakses data
- No additional calculations to work out effective address : Tidak memerlukan kalkulasi untuk mendapatkan effective address
- Limited address space : Address space yang terbatas.

Mode Indirect Addressing
Merupakan mode pengalamatan operand dimana area alamat (address field) berisi alamat dari suatu alamat yang akan menunjukkan alamat dari suatu nilai yang akan diproses.
(Memory cell pointed to by address field contains the address of (pointer to) the operand)
EA = (A)
Keterangan:
- Cari di memory alamat A, cari alamat yang tertulis pada A untuk operand (Look in A, find address (A) and look there for operand).
Contoh: ADD (A)
Keterangan:
- Tambahkan isi dari cell yang alamatnya ditunjukkan oleh isi yang terdapat pada A dengan nilai yang ada di register accumulator dan simpan hasilnya di register accumulator. (Add contents of cell pointed to by contents of A to accumulator).
Karakteristik:
- Large address space : memerlukan space address yang besar.
- May be nested, multilevel, cascaded : Dapat dibuat nested (bersarang), multilevel dan cascade (bertumpuk).
- Multiple memory accesses to find operand : pengaksesan memory yang multiple untuk

mendapatkan operand sehingga mengakibatkan proses mode ini agak lebih lambat.

Mode Register Addressing
Merupakan mode pengalamatan operand dimana operand yang akan diproses ditampung/disimpan dalam register yang namanya ditulis di area alamat (address field).
(Operand is held in register named in address field)
EA = R
Karakteristik:
- Limited number of registers : terbatas pada jumlah register yang hanya sedikit
- Very small address field needed : karena addres yang kecil pada register, maka:
- Shorter instructions : instruksinya lebih pendek
- Faster instruction fetch : pemasukan data lebih cepat
- No memory access : tidak memerlukan akses ke memory
- Very fast execution : eksekusi sangat cepat
- Very limited address space : tetapi space alamat (address) sangat terbatas