ARSITEKTUR REDUCED INSTRUCTION SET COMPUTERS (RISC)

Oleh :

LaOde Abdul Jumar

1. ARSITEKTUR REDUCED INSTRUCTION SET COMPUTERS (RISC)

Perkembangan inovasi komputer sejak 1960 menambah satu daftar penemuan yang sangat menarik dan paling penting , yaitu Arsitektur *Reduced Instruction Set computers (RISC*). Walaupun sistem RISC telah ditentukan dan dirancang dengan berbagai cara berdasarkan komunitasnya, elemen penting yang digunakan sebagian rancangan umumnya adalah sebagai berikut :

1. Set instruksi yang terbatas dan sederhana
2. Register general purpose berjumlah banyak atau penggunaaan teknologi kompiler untuk mengoptimalklan penggunaan register.
3. Penekanan pada pengoptimalan pipeline instruksi.

1. 1. Karakteristik-Karakteristik Eksekusi Instruksi

Salah satu evolusi komputer yang besar adalah evolusi bahasa pemprograman. Bahasa pemprograman memungkinkan programmer dapat mengekspresikan algoritma lebih singkat, lebih memperhatikan rincian, dan mendukung penggunaan pemprograman terstruktur, tetapi ternyata muncul masalah lain yaitu *semantic gap,* yaitu perbedaan antara operasi-operasi yang disediakan oleh HLL dengan yang disediakan oleh arsitektur komputer, ini

ditandai dengan ketidakefisienan eksekusi, program mesin yang berukuran besar, dan kompleksitas kompiler. Untuk mengurangi kesenjangan ini para perancang menjawabnya dengan arsitektur. Fitur-fiturnya meliputi set-set instruksi yang banyak, lusinan mode pengalamatan, dan statemen –statemen HLL yang diimplementasikan pada perangkat keras. Set-set instruksi yang kompleks tersebut dimaksudkan untuk :

1. Memudahkan pekerjaan kompiler
2. Meningkatkan efisiensi eksekusi, karena operasi yang kompleks dapat diimplementasikan didalam mikrokode.
3. Memberikan dukungan bagi HLL yang lebih kompleks dan canggih.

Oleh karena itu untuk memahami RISC perlu memperhatikan karakteristik eksekusi instruksi. Adapun aspek-aspek komputasinya adalah :

1. *Operasi-operasi yang dilakukan* ,.
2. *Operand-operand yang digunakan*,
3. *Pengurutan eksekusi*,.

1. Operasi

Beberapa penelitian telah menganalisis tingkah laku program HLL (High Level Language). Assignment Statement sangat menonjol yang menyatakan bahwa perpindahan sederhana merupakan satu hal yang penting. Hasil penelitian ini merupakan hal yang penting bagi perancang set instruksi mesin yang mengindikasikan jenis instruksi mana yang sering terjadi karena harus didukung optimal.

2. Operand

Penelitian Paterson telah memperhatikan [PATT82a] frekuensi dinamik terjadinya kelas-kelas variabel. Hasil yang konsisten diantara program pascal dan C menunjukkan mayoritas referensi menunjuk ke variable scalar. Penelitian ini telah menguji tingkah laku dinamik program HLL yang tidak tergantung pada arsitektur tertentu. Penelitian [LUND77] menguji instruksi DEC-10 dan secara dinamik menemukan setiap instruksi rata-rata mereferensi 0,5 operand dalam memori dan rata-rata mereferensi 1,4 register. Tentu saja angka ini tergantung pada arsitektur dan kompiler namun sudah cukup menjelaskan frekuensi pengaksesan operand sehingga menyatakan pentingnya sebuah arsitektur.

3. Procedure Calls

Dalam HLL procedure call dan return merupakan aspek penting karena merupakan operasi yang membutuhkan banyak waktu dalam program yang dikompalasi sehingga banyak berguna untuk memperhatikan cara implementasi opperasi ini secara efisien. Adapun aspeknya yang penting adalah *jumlah parameter* dan variabel yang berkaitan dengan prosedur dan kedalaman pensarangan (nesting).

4. Implikasi

Secara umum penelitian menyatakan terdapat tiga buah elemen yang menentukan karakter arsitektur RISC :

1. Penggunaan register dalam jumlah besar yang ditunjukan untuk mengotimalkan pereferensian operand.

2. Diperlukan perhatian bagi perancangan pipelaine instruksi karena tingginya proporsi instruksi pencabangan bersyarat dan procedure call, pipeline instruksi yang bersifat langsung dan ringkas menjadi tidak efisien
3. Terdapat set instruksi yang disederhanakan

1. 2. Karakteristik Arsitektur Reduced Instruction Set Computers (RISC)

Arsitektur RISC memiliki beberapa karakteristik diantaranya :

1. *Siklus mesin* ditentukan oleh waktu yang digunakan untuk mengambil dua buah operand dari register, melakukan operasi ALU, dan menyimpan hasil operasinya kedalam register, dengan demikian instruksi mesin RISC tidak boleh lebih kompleks dan harus dapat mengeksekusi secepat mikroinstruksi pada mesin-mesin CISC. Dengan menggunakan instruksi sederhana atau instruksi satu siklus hanya dibutuhkan satu mikrokode atau tidak sama sekali, instruksi mesin dapat dihardwired. Instruksi seperti itu akan dieksekusi lebih cepat dibanding yang sejenis pada yang lain karena tidak perlu mengakses penyimapanan kontrol mikroprogram saat eksekusi instruksi berlangsung.
2. Operasi berbentuk dari register-ke register yang hanya terdiri dari operasi **load** dan **store** yang mengakses memori . Fitur rancangan ini menyederhanakan set instruksi sehingga menyederhanakan pula unit control. Keuntungan lainnya memungkinkan optimasi pemakaian register sehingga operand yang sering diakses akan tetap ada di penyimpan berkecepatan

tinggi. Penekanan pada operasi register ke register merupakan hal yang unik bagi perancangan RISC.

3. Penggunaan mode pengalamatan sederhana, hampir sama dengan instruksi menggunakan pengalamatan register,. Beberapa mode tambahan seperti pergeseran dan pe-relatif dapat dimasukkan selain itu banyak mode kompleks dapat disintesis pada perangkat lunak dibanding yang sederhana, selain dapat menyederhanakan sel instruksi dan unit kontrol.
4. penggunaan format-format instruksi sederhana, panjang instruksinya tetap dan disesuaikan dengan panjang word. Fitur ini memiliki beberapa kelebihan karena dengan menggunakan field yang tetap pendekodean opcode dan pengaksesan operand register dapat dilakukan secara bersama-sama

2. Ciri-Ciri RISC

1. Instruksi berukuran tunggal
2. Ukuran yang umum adalah 4 byte
3. Jumlah pengalamatan data sedikit, biasanya kurang dari 5 buah.
4. Tidak terdapat pengalamatan tak langsung yang mengharuskan melakukan sebuah akses memori agar memperoleh alamat operand lainnya dalam memori
5. Tidak terdapat operasi yang menggabungkan operasi load/store dengan operasi aritmatika, seperti penambahan ke memori dan penambahan dari memori.
6. Tidak terdapat lebih dari satu operand beralamat memori per instruksi

7. Tidak mendukung perataan sembarang bagi data untuk operasi load/ store
8. Jumlah maksimum pemakaian memori manajemen bagi suatu alamat data adalah sebuah instruksi .
9. Jumlah bit bagi integer register spesifier sama dengan 5 atau lebih, artinya sedikitnya 32 buah register integer dapat direferensikan sekaligus secara eksplisit.
10. Jumlah bit floating point register spesifier sama dengan 4 atau lebih, artinya sedikitnya 16 register floating point dapat direferensikan sekaligus secara eksplisit.

Beberapa prosesor implementasi dari arsiteketur RISC adalah AMD 29000, MIPS R2000, SPARC, MC 88000, HP PA, IBM RT/TC, IBM RS/6000, intel i860, Motorola 88000 (keluarga Motorola), PowerPC G5.

2. PROSESSOR YANG MENGGUNAKAN SISTEM RISC

2.1. PowerPC dibangun dengan arsitektur RISC

Proyek mini komputer 801 di IBM pada tahun 1975 mengawali banyak konsep arsitektur yang digunakan dalam sistem RISC. 801 bersama dengan prosessor RISC I Berkeley, meluncurkan gerakan RISC, namun 801 hanya merupakan prototipe yang ditujukan untuk mengenalkan konsep disain. Keberhasilan memperkenalkan 801 menyebabkan IBM membangun produk workstation RISC komersial yaitu PC RT pada tahun 1986, dengan mengadaptasi konsep arsitektural 801 kedalam kinerja yang sebanding atau yang

lebih baik. IBM RISC System/6000 merupakan mesin RISC superscalar[1][3] yang dipasarkan sebagai workstation berunjuk kerja tinggi, tidak lama kemudian IBM mengkaitkan mesin ini sebagai arsitektur POWER.

IBM kemudian menjalin kerjasama dengan Motorola, pembuat mikroprosessor seri 6800, dan Apple, yang menggunakan keping Motorola dalam komputer Macintoshnya dan hasilnya adalah seri mesin yang mengimplementasikan arsitektur PowerPC yang diturunkan dari arsitektur POWER dan merupakan sistem RISC superscalar. Sejauh ini diperkenalkan empat anggota kelompok PowerPC yaitu

1. **601**,merupakan mesin 32-bit yang ditujukan untuk membawa arsitektur PowerPC kepasar secepat mungkin.
2. **603**, merupakan mesin 32-bit yang ditujukan bagi low-end desktop dan komputer portable dengan implementasi yang lebih efesien.
3. **604**, merupakan mesin 32-bit yang ditujukan bagi low-end server dan desktop, dengan menggunakan teknik rancangan superscalar lanjutan guna mendapatkan kinerja yang lebih baik.
4. **620**, ditujukan bagi high-end server, sekaligus merupakan kelompok PowerPC pertama yang mengimplementasikan arsitektur 64 bit penuh, termasuk regiater 64-bit dan lintasan data.

2.2. Karakteristik dan Fungsi

1. Jenis-Jenis Data

1

PowerPC dapat beroperasi menggunakan data yang panjang 8 bit (byte), 16 bit (halfword), 32 bit (word), dan 64 bit (doubleword). Beberapa instruksi mengharuskan agar operand memori dijajarkan (aligned) pada batas 32-bit, walaupun secara umum tidak terlalu diperlukan. Salah satu ciri PowerPC yang menarik adalah dapat menggunakan cara little-endian maupun big-endian[2][6], dengan kata lain, byte yang paling kurang signifikan disimpan dalam alamat terendah atau tertinggi. Konsep ke-endianan pertama kali dibahas dalam literatur Cohen [COHE8]. Pada byte ke-endian-an harus melakukan pengurutan nilai-nilai skalar multibyte. Konsep ini terjadi apabila terdapat kebutuhan untuk memperlakukan entitas multiple-byte sebagai butir data tunggal, walaupun entitas ini terdiri dari unit-unit yang dapat dialamati yang lebih kecil. Beberapa mesin seperti intel 80x86, pentium, dan VAX, merupakan mesin-mesin litlle endian, sedangkan mesin-mesin seperti IBM S/370, Motorola 680x0, dan sebagian besar mesin-mesin RISC merupakn mesin-mesin big-endian. Sifat ke-endian-an tidak akan melampaui unit data. Dalam sembarang mesin, aggregate seperti file, struktur data, dan array terdiri dari beberapa unit data, yang masing-masing memakai ke-endian-an. Jadi konversi blok memori dari suatu jenis ke-endian-an kejenis lainnya memerlukan pemahaman struktur data.

Tidak terdapat konsensus umum tentang ke-endianan yang terbaik[3][7], PowerPC sendiri adalah jenis prosesor yang bi-endian, yang mendukung baik mode big-endian maupun litlle-endian. Arsitektur bi-endian memungkinkan

2

3

pembuat perangkat lunak untuk memilih mode yang mana saja ketika harus memindahkan sistem operasi dan aplikasi dari suatu mesin ke mesin lainnya.

Byte, halfword, word, doubleword merupakan jenis data umum. Prosesor mengiterpretasikan isi item data tertentu tergantung pada instruksi. Prosesor fixed point mengenal jenis data berikut :

Unsigned Byte : dapat digunakan bagi operasi logika atau aritmetika integer. Data ini dimuat dari memori ke register umum dengan zero-extending dsebelah kiri keukuran penuh register.

- *Unsigned Halfword* : seperti diatas namun dengan kuantitas 16-bit.
- *Signed Halfword* : digunakan untuk operasi aritmatika, dimuatkan kedalam memori dengan sign-extending pada sebelah kiri keukuran penuh register (yaitu, bit tanda disalinkan keposisi-posisi yang kosong).
- *Unsigned Word* : digunakan untuk operasi logika dan berfungsi sebagai pointer lokal.
- Signed Word : digunakan untuk operasi aritmatika.
- Unsigned Doubleword : digunakan sebagai pointer alamat.
- Byte String : panjangnya mulai 0 hingga 128 byte.

Selain itu PowerPC mendukung data floating poing presisi tunggal dan presisi ganda yang ditetapkan pada IEEE 754.

2. Jenis Jenis Operasi

PowerPC banyak memiliki jenis operasi , berikut disajikan berbagai jenis operasi pada PowerPC :

Instruksi	Uraian
	Berorientasi Pencabangan
b	Pencabangan tidak bersyarat
bl	Bercabang kealamat sasaran dan menaruh alamat efektif instruksi yang berada setelah pencabangan kedalam link register
bc	Pencabangan bersyarat pada Count Register dan/atau pada bit dalkam Condition Register.
sc	System Call untuk membangkitkan layanan sistem operasi
trap	Memebandingkan dua buah operand dan membangkitkan system trap handler bila persyaratan tertentu dipenuhi.
	Load/Store
lwzu	Memuatkan word dan nol kesebelah kiri; mengupdate register sumber.
ld	Memuatkan dobleword.
lmw	Memuatkan word ganda; memuatkan word berurutan ke regiater yang berdekatan dari register sasaran melalui General Purpose Register 31.
lswx	memuatkan suatu untaian byte kedalam register yang dimulai dengan register sasaran; empat byte per-register; diambil semua dari register 31 hingga register 0.
	Arimatika Integer
add	Menjumlahkan isi dari dua buah integer dan menyimpannya dalam register ketiga
subf	Mengurangkan isi dua buah register dan menyimpannya dalam register ketiga.
mullw	Mengalikan isi dua buah register orde rendah 32-bit dan menyimpan hasil perkaliannya dalam register 64-bit ketiga.
divd	Membagi isi dua buah register 64-bit dan menyimpan kuosiennya dalam register ketiga.
	Logika dan Sift
cmp	Membandingkan dua buah operand dan menyetel empat buah bit kondisi dalam field register kondisi tertentu.
crand	Condition Register AND : dua bit Condition Register di-AND-kan dan hasilnya disimpan dalam salah satu dari kedua posisi tersebut.
and	Meng-AND-kan isi dua buah register dan menyimpannya dalam register ketiga

cntlzd	Mencacah jumlah bit 0 berturutan yang berawal pada bit nol dalam register sumber dan menempatkan hasil perhitungan dalam regiater tujuan.
rldic	Merotasikan ke kiri register doubleword, meng-AND-kannya dengan mask, dan menyimpannya dalam register tujuan.
sld	Menggeser kekiri dalam register sumber dan menyimpannya dalam register tujuan

Floating Point

lfs	Memuatkan bilangan floating point 32-bit dari memori, mengubahnya kedalam format 64 bit, dan menyimpannya dalam register floating point.
fadd	Menjumlahkan dua buah register floating point dan menyimpannya dalam register ketiga.
fmadd	Mengalikan isi dua buah register, menambahkan isi regiater ketiga, dan menyimpan hasilnya dalam regiater keempat.
fcmpu	Membandingkan dua buah operand floating point dan menyetel bit-bit kondisi.

Manajemen Cache

dcbf	Membersihkan (flush) blok data cache; melakukan lookup dalam cache yang terdapat pada alamat sasaran tertentu dan melakukan operasi pembersihan.
icbi	Menginvalidasikan instruksi blok cache

2.1. Instruksi-Instruksi berorientasi Pencabangan

PowerPC memiliki orientasi pencabangan tidak bersyarat dan pencabangan bersyarat. Instruksi-instruksi pencabangan bersyarat menguji suatu bit tunggal dari register kondisi apakah benar, salah, atau tidak peduli dan isi dari counter register apakah nol, bukan nol, atau tidak peduli. Dengan demikian terdapat sembilan macam kondisi instruksi pencabangan bersyarat yang terpisah. Apabila counter register diuji apakah nol atau bukan nol, maka sesudah pengujian register berkurang 1. Hal ini tentunya memudahkan penyiapan loop iterasi. Instruksi dapat juga mengindikasikan bahwa alamat dari pencabangan itu ditempatkan dalam link register, hal ini memungkinkan pengolahan call/return.

2.2. Instruksi-instruksi Load/Store

Dalam arsitektur PowerPC hanya instruksi load/store yang dapat mengakses lokasi memori, instruksi logika dan aritmetika hanya dilakukan terhadap register. Terdapat dua fitur yang membedakan instruksi-instruksi load/store :

1. Ukuran data, dimana data dapat dipindahkan dalam satu byte, halfword, word, atau doubleword. Instruksi-instruksi juga dapat digunakan untuk memuat atau menyimpan suatu untai byte ke dalam sejumlah register atau dari sejumlah register
2. Ekstensi Tanda, dimana pada pembuatan word dan halfword, bit-bit sebelah kiri register 64-bit tujuan yag tidak dipakai dapat diisi dengan bilangan-bilangan nol atau dengan bit tanda dari kuantitas yang dimuatkan.

KELEBIHAN DAN KEKURANGAN TEKNOLOGI RISC

Teknologi RISC relatif masih baru oleh karena itu tidak ada perdebatan dalam menggunakan RISC ataupun CISC, karena tekhnologi terus berkembang dan arsitektur berada dalam sebuah spektrum, bukannya berada dalam dua kategori yang jelas maka penilaian yang tegas akan sangat kecil kemungkinan untuk terjadi.

1. Kelebihan

1. Berkaitan dengan penyederhanaan kompiler, dimana tugas pembuat kompiler untuk menghasilkan rangkaian instruksi mesin bagi semua pernyataan HLL. Instruksi mesin yang kompleks seringkali sulit digunakan karena kompiler harus menemukan kasus-kasus yang sesuai dengan konsepnya. Pekerjaan mengoptimalkan kode yang dihasilkan untuk meminimalkan ukuran kode, mengurangi hitungan eksekusi instruksi, dan meningkatkan pipelining jauh lebih mudah apabila menggunakan RISC dibanding menggunakan CISC.
2. Arsitektur RISC yang mendasari PowerPC memiliki kecenderungan lebih menekankan pada referensi register dibanding referensi memori, dan referensi register memerlukan bit yang lebih sedikit sehingga memiliki akses eksekusi instruksi lebih cepat.
3. Kecenderungan operasi register ke register akan lebih menyederhanakan set instruksi dan menyederhanakan unit kontrol serta pengoptimasian register akan menyebabkan operand-operand yang sering diakses akan tetap berada dipenyimpan berkecepatan tinggi.
4. Penggunaan mode pengalamatan dan format instruksi yang lebih sederhana.

2. Kekurangan

1. Program yang dihasilkan dalam bahasa simbolik akan lebih panjang (instruksinya lebih banyak).

2. Program berukuran lebih besar sehingga membutuhkan memori yang lebih banyak, ini tentunya kurang menghemat sumber daya.
3. Program yang berukuran lebih besar akan menyebabkan
   b. Menurunnya kinerja, yaitu instruksi yang lebih banyak artinya akan lebih banyak byte-byte instruksi yang harus diambil.
   c. Pada lingkungan paging akan menyebabkan kemungkinan terjadinya page fault lebih besar.

KESIMPULAN

1. Arsitektur PowerPC merupakan pengembangan IBM 801, RT PC, dan RS/600 (dikenal juga dengan implementasi arsitektur POWER).

2. Implementasi pertama arsitektur power PC yaitu 601 memiliki rancangan yang sangat mirip dengan rancangan RS 6000, model PowerPC berikutnya mempunyai konsep superscalar.
3. Kelebihan arsitektur RISC yang berkaitan dengan kinerja dapat ditunjukan dengan sejumlah “Sircumstansial Evidence”.
   a. Optimasi kompiler yang lebih efektif dan dapat dikembangkan
   b. Sebagian besar instruksi yang dihasilkan oleh kompiler relatif sederhana.
   c. Berkaitan dengan penggunaan pipelining instruksi yang diterapkan secara lebih efektif terhadap RISC.
   d. Program-program RISC harus lebih responsife terhadap interrupt.

   Berkaitan dengan implementasi VLSI

   d. Apabila digunakan rancangan dan implementasi CPU akan berubah, artinya dimungkinkan untuk menaruh CPU keseluruhan pada keping tunggal.
   e. Waktu yang dibutuhkan untuk implementasi dan perancangan karena prosessor VLSI cukup sulit dibuat sehingga para perancang harus membuat rancangan rangkaian, tata letak dan pemodelan pada tingkat perangkat, dengan menggunakan pemodelan RISC proses tersebut akan lebih mudah selain apabila kinerja keping RISC ekuivalen dengan mikroprosessor CISC (Pentium) yang setara maka keuntungan dengan memakai pendekatan RISC akan terasa sekali.

DAFTAR PUSTAKA

Stalling, Williams, *Organisasi Dan Arsitektur Komputer: Perancangan Kinerja*. Jilid 1, Terj. Gurnita Priatna, Jakarta : Prenhallindo,1998.

Stalling, Williams, *Organisasi Dan Arsitektur Komputer: Perancangan Kinerja*. Jilid 2, Terj. Gurnita Priatna, Jakarta : Prenhallindo,1998.

PowerPC Siap Masuki 2-GHz, Komputek, Juni 2001, edisi 219 minggu II, hal 3.
http://www.lib.usm.my/Moro/GPI/bab10.html