به نام خدا

۱) در زبان های سطح بالا دستور العمل ها توسط کامپایلر یا مفسر با دستور العمل های CPU جایگزین میشوند که عمدا این جایگزینی به صورت یک به چند است. زبان اسمبلی به سخت افزار نزدیک بوده و این جاگزینی که اینبار توسط اسمبلر انجام میشود یک به یک است. از این جهت است که برنامه های اسمبلی از برنامه هایی که با زبان های سطح بالا نوشته میشوند سریعتر است. در برنامه های اسمبلی کنترل بیشتری روی حافظه است.

۲) الف) از ثبات ها در CPU برای نگهداری اطلاعات و دسترسی سریع به آنها استفاده میشود.

ب) ثبات هایی که کاربر (برنامه نویس) اجازه دسترسی به آنها را داشته باشد ثبات های عام منظوره هستند و ثبات هایی که تنها خود CPU اجازه دسترسی به آنها را دارد ثبات های خاص منظوره هستند.

ج) در این پردازنده هر دو نوع ثبات های خاص منظوره و عام منظوره وجود دارند. ثبات هایی مثل AX,BX,CX,DX,CS,ES,DS,SS,... در این پردازنده عام منظوره هستند و ثبات هایی مثل IP,EFLAGS,... خاص منظوره اند.

د) ثبات ها در CPU های ARM و x64 مانند x86 شامل هر دو نوع عام منظوره و خاص منظوره اند. اما در MIPS تمامی ثبات ها مثل یکدیگر بوده و عام منظوره اند.

ه) ثبات EPI(x86) – RPI(x64) برای نگهداری خطی از برنامه که در حال اجرا است استفاده میشود.

ثبات EFLAGS برای نگهداری وضیعت CPU بعد از اجرای یک دستور العمل استفاده میشود.

ثبات های CR برای کنترل وضیعت CPU استفاده میشوند.

۳) در پردازنده های x86 از EFLAGS برای تصمیم گیری روند اجرا استفاده میشود. در پردازه هایی مانند MIPS وضیعت CPU در رجیستر هایی که از پیش تعیین شده (این رجیستر ها عام منظوره هستند) ذخیره میشوند. ایراد روش دوم این است که ممکن است برنامه نویس اشتباها باعث تغییر این رجیستر در زمانی که CPU آن را set کرده است بشود ولی از سویی استفاده از روش دوم این قابلیت را به برنامه نویس میدهد که با یک instruction در برنامه دو رجیستر را مقایسه کند و jump را انجام دهد. در حالی که میدانیم این عمل در x86 شامل دو instruction است.

۴) PF : Parity Flag : توازن زوج بوره و از آن برای تشخیص خطا استفاده میشود

AF : Axillary Carry Flag : carry داخلی بین ۲ , ۸ بیتی ثبات ها را مشخص میکند.

ZF : Zero Flag : مشخص میکند حاصل آخرین دستور برابر صفر بوده است یا خیر

TF : Trap Flag : از آن برای اجرای برنامه در حالت تک گام استفاده میشود.

SF : Sign Flag

IF : Interrupt Flag

DF : Direction Flag

OF : Overflow Flag

IOPL : IO Privilege

NT : Nested Tap

CF : Carry Flag

۵) ماشین های CISC دارای مجموعه دستورات بزرگی هستند که اعمالی مانند کارهای منطقی و محاسباتی و ... راا شامل میشود. در این ماشین ها برنامه های کامپایل شده دارای تعداد کمی دستور العمل خواهند بود. ماشین های RISC دارای مجموعه دستورات کوچکی هستند. در این ماشین ها به علت ساده بودن دستورالعمل ها اجرا و decode آنها سریعتر است.

۶) Stack Machine : ماشین هایی هستند که از یک پشته برای ارزیابی زیر دستور ها استفاده میکنند. در این ماشین ها برای اجرای دستوری مانند ADD توسط CPU برنامه میبایست operand ها را در stack , push کرده و بعد از اجرای دستور نتیجه را از stack , pop کند. نمونه ای از این ماشین ها JVM(Java Virtual Machine) میباشد.

Register Machine : ماشین هایی هستند که دارای تعدادی رجیستر برابر (رجیستر هایی با کارایی و نوع یکسان) هستند و از آنها برای ارزیابی زیر دستور ها استفاده میکنند.

امروزه ماشین هایی همچون اینتل دارای معماری بینابینی هستند.

۷) الف) در این روش بایت های پرارزش در خانه هایی با آدرس کوچکتر نگهداری میشوند.

ب) در این روش بایت های پرارزش در خانه هایی با آدرس بزرگتر نگهداری میشوند. این روش در CPU اینتل استفاده میشور. یکی از برتری های این روش آسان کردن Cast از نوع داده بزرگتر به کوچکتر است.

ج) به کم ارزش ترین بیت یک عدد گفته میشود.

د) به پرارزش ترین بیت یک عدد گفته میشود.

ه) به هر رقم باینری یک bit گفته میشود. به هر ۸ بیت یک byte گفته میشود و به هر دو بایت یک word گفته میشود.

۸) الف) pipelining در کامپیوتر به معنای قرار دادن تعدادی قطعه پردازشگر داده به صورت سری میباشد. در این CPU ها برای پردازش دستورالعمل ها به صورت سری انجام میشود.

ب) در این روش دستورالعمل های CPU به صورت موازی توسط Functional Unit های مختلف پردازش میشوند. این CPU ها به صورت فیزیکی دارای چند core هستند.

ج) در این روش هر CPU به اجرای دو Thread به صورت موازی مشغول میشود. در این روش CPU به صورت مجازی دارای چند core میشود.

د) در این CPU ها برنامه ها میتوانند دستور العمل هایی را به صورت همزمان روی CPU اجرا کنند.

۹) با استفاده از این روش اگر در تولید object file دچار مشکل شویم متوجه میشویم که error ها متعلق به کدام source فایل برنامه بوده یا اگر در link کردن دچار مشکل شویم متوجه میشویم که error از چه کتابخانه هایی میباشد. یکی دیگر از مزایای این روش این است portability است که object file ها دارا میباشند به این معنی که ممکن بتوان این فایل ها را روی سیستم دیگری با همان پردازنده ولی با سیستم عامل اندکی متفاوت link و استفاده کرد. این قابلیت عمدا در سیستم های Unix Like که از نسخه های متفاوتی از استاندارد POSIX-SUS استفاده میکنند کاربردی است.

۱۰) object فایلها برنامه ی خامی هستند که به زبان ماشین ترجمه شده است و هیچ یک از library های مورد نیازش به آن اضافه نشده است. Library ها مجموعه ای از توابع به زبان ماشین هستند که برنامه ها میتوانند از آنها استفاده کنند که در زمان لینک کردن در کنار برنامه قرار میگرند. DLL ها شبیه به کتابخانه ها میباشند با این تفاوت که در زمان اجرا به برنامه اضافه میشوند. معمولا چند برنامه میتوانند از یک DLL که در حافظه load شده است استفاده کنند. با این کار از حافظه به صورت بهینه استفاده میشود و یک کتابخانه چندباره بارگذاری نمیشود.

۱۱) به وسیله ی این ابزار ها میتوان فهمید که هر object file از چه قسمت هایی تشکیل شده (sections) , برای چه CPU (Little Endian – Big Endian; x64 – x86; vendor; …) تولید شده است , جدول نماد های آن چگونه است (symbol table) , داده هایی که در قسمت data نگهداری مشوند چیستند و header ای که OS برای این فایل گذاشته است چیست و .... و از این داده ها میتوان برای debug و هک و override کردن و .. استفاده کرد.