Машинно-зависимые языки программирования, лекция 8

Каф. ИУ7 МГТУ им. Н. Э. Баумана, 2022 г.

RISC-архитектура

Ранние архитектуры процессоров (комплексные, CISC (Complex instruction set computer)):

- большее количество команд
- разные способы адресации для упрощения написания программ на ассемблере
- поддержка конструкций языков высокого уровня

Недостатки: на практике многие возможности CISC используются компиляторами ЯВУ ограниченно, а их поддержка затратна.

RISC (reduced instruction set computer):

- сведение набора команд к простым типовым
- большее количество регистров (возможно за счёт общего упрощения архитектуры)
- стандартизация формата команд, упрощение конвейеризации

Семейство процессоров ARM

Свыше 90% рынка процессоров для мобильных устройств
ARMv1 – 1985 г.
Современные версии архитектуры - ARMv7, ARMv8.
Профайлы: Classic, Microcontroller, Real-time, Application (последняя буква в архитектуре)
Регистры общего назначения ARMv7:

- R0-R12
- R13 – SP
- R14 – LR (регистр связи)
- R15 – PC (счётчик команд)

Регистры R8–R12 существуют в двух экземплярах:

- для режима обработки быстрого прерывания
- для остальных режимов

Регистры LR и SP для каждого режима свои (6-7 пар)

Режимы ARM

- User mode — обычный режим выполнения программ. В этом режиме выполняется большинство программ.
- Fast Interrupt (FIQ) — режим быстрого прерывания (меньшее время срабатывания).
- Interrupt (IRQ) — основной режим прерывания.
- System mode — защищённый режим для использования операционной системой.
- Abort mode — режим, в который процессор переходит при возникновении ошибки доступа к памяти (доступ к данным или к инструкции на этапе prefetch конвейера).
- Supervisor mode — привилегированный пользовательский режим.
- Undefined mode — режим, в который процессор входит при попытке выполнить неизвестную ему инструкцию

Наборы команд ARM

- Базовый ARM
- Thumb (16-разрядные, более производительные)
- Thumb2 (32-разрядные)
- A64

Расширения

- VFP v1-v5
- SIMD, NEON, SVE
- AES, SHA

Current Program Status Register (CPSR)

Bits	Name	Function
[31]	N	Negative condition code flag
[30]	Z	Zero condition code flag
[29]	C	Carry condition code flag
[28]	V	Overflow condition code flag
[27]	Q	Cumulative saturation bit
[26:25]	IT[1:0]	If-Then execution state bits for the Thumb IT (If-Then) instruction
[24]	J	Jazelle bit
[19:16]	GE	Greater than or Equal flags
[15:10]	IT[7:2]	If-Then execution state bits for the Thumb IT (If-Then) instruction
[9]	E	Endianness execution state bit: 0 - Little-endian, 1 - Big-endian
[8]	A	Asynchronous abort mask bit
[7]	I	IRQ mask bit
[6]	F	FIRQ mask bit
[5]	T	Thumb execution state bit
[4:0]	M	**Mode field**

Быстрые (FIQ) и обычные (IRQ) прерывания

Fast interrupt - режим для получения данных от оборудования, минимизирующий задержки:

- скорость обработки выше;
- допустима работа только одного обработчика единовременно.

Standart interrupt - все прочие прерывания.

Команды ветвления B, BL, BLX

- B (Branch) - переход
- BL (Branch with link) - переход с сохранением адреса возврата в LR
- BLX - переход с переключением системы команд

Вызов программного прерывания

SWI immed_8 (0..255)

Переводит процессор в Supervisor mode, CPSR сохраняется в Supervisor Mode SPSR, управление передаётся обработчику прерывания по вектору.

Архитектура VLIW. Эльбрус-8С

VLIW (very large instruction word) - продолжение идей RISC для многопроцессорных систем. В каждой инструкции явно указывается, что должно делать каждое ядро процессора.
Эльбрус-8С:

- 8 ядер
- в каждом ядре - 6 арифметико-логических каналов со своими АЛУ и FPU, до 24 операций за такт

Широкая команда Эльбруса

Широкая команда - набор элементарных операций, которые могут быть запущены на исполнение в одном такте.
Доступны:

- 6 АЛУ (возможности различны)
- Устройство передачи управления
- 3 устройства для операций над предикатами
- 6 квалифицирующих предикатов
- 4 устройства асинхронного для команд чтения данных
- 4 32-разрядных литерала для констант

Определяющие свойства архитектуры "Эльбрус"

- Регистровый файл (рабочие регистры) - 256 регистров (32 для глобальных данных и 224 для стека процедур)
  - механизм регистровых окон: вызывающая подпрограмма выделяет вызываемой область в своём регистровом окне; на начало указывает регистр WD
  - пространство регистров подвижной базы - пространство в текущем окне, на начало указывает регистр BR
- Предикатный файл - 32 регистра со значениями true/false
- Подготовка передачи управления (disp) - подготовка к переходам при ветвлении для исключения задержек
- Асинхронный доступ к массивам

Java. Java virtual machine (JVM)

Java - объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems.
Приложения Java обычно транслируются в специальный байт-код, поэтому они могут работать на любой компьютерной архитектуре, для которой существует реализация виртуальной Java-машины.
Байт-код Java — набор инструкций, исполняемых виртуальной машиной Java. Каждый код операции байт-кода — один байт.
Группы инструкций:

- загрузка и сохранение (например, ALOAD_0, ISTORE),
- арифметические и логические операции (например, IADD, FCMPL),
- преобразование типов (например, I2B, D2I),
- создание и преобразование объекта (например, NEW, PUTFIELD),
- управление стеком (например, DUP, POP),
- операторы перехода (например, GOTO, IFEQ),
- вызовы методов и возврат (например, INVOKESTATIC, IRETURN).

javap - дизассемблер файлов классов Java

Платформа .NET. CLR, CIL

.NET (2002) - платформа, основанная на CLR (Common Language Runtime, общеязыковая исполняющая среда).
CLR — исполняющая среда для байт-кода CIL (MSIL), в которой компилируются программы, написанные на .NET-совместимых языках программирования.
CIL (Common Intermediate Language) — «высокоуровневый ассемблер» виртуальной машины .NET, основанный на работе со стеком.

```
ldloc.0     // push local variable 0 onto stack
ldloc.1     // push local variable 1 onto stack
add         // pop and add the top two stack items then push the result
            // onto the stack
stloc.0     // pop and store the top stack item to local variable 0
```

ildasm, ilasm - дизассемблер/ассемблер промежуточного языка (intermediate language)

WebAssembly (wasm)

WebAssembly — это бинарный формат инструкций для стековой виртуальной машины, предназначенной для компиляции программ на ЯВУ (C, C++, C#, Go, TypeScript/AssemblyScript, Kotlin, Pascal, Rust, D, Erlang) для WEB.

Исходный код на C	«линейный ассемблерный байт-код»	бинарный код WASM

`int factorial(int n) {`
`  if (n == 0)`
`  return 1;`
`  else`
`  return n * factorial(n-1);`
`}`
	`get_local 0`
	`i64.eqz`
	`if i64`
	`  i64.const 1`
	`else`
	`  get_local 0`
	`  get_local 0`
	`  i64.const 1`
	`  i64.sub`
	`  call 0`
	`  i64.mul`
	`end`
		`20 00`
		`50`
		`04 7e`
		`42 01`
		`05`
		`20 00`
		`20 00`
		`42 01`
		`7d`
		`10 00`
		`7e`
		`0b`