第18章

召集大会

本章介绍了RV32和RV64程序的C编译器标准和两种调用约定：基本ISA加标准通用扩展的约定（RV32G/RV64G），以及缺少浮点单元的实现（如RV32I/RV64I）的软浮点约定。

带有ISA扩展的实现可能需要扩展的调用约定。

18.1 C数据类型和对齐方式

表18.1总结了RISC-V
C程序原生支持的数据类型。在RV32和RV64的C语言编译器中，C语言类型int的宽度为32位。另一方面，long和指针的宽度都与整数寄存器一样，所以在RV32中，两者都是32位宽，而在RV64中，两者都是64位宽。等价地，RV32采用了ILP32整数模型，而RV64是LP64。在RV32和RV64中，C类型的long long是64位整数，float是32位IEEE 754-2008浮点数，double是64位IEEE 754-2008浮点数，long double是128位IEEE浮点数。

C类型的char和unsigned char是8位无符号整数，存储在RISC-V整数寄存器中时是零扩展。unsigned short是16位无符号整数，存储在RISC-V整数寄存器中时是零扩展。signed char是8位有符号整数，存储在RISC-V整数寄存器中时是符号扩展，即位（XLEN-1）...7都相等。short是16位有符号整数，存储在一个寄存器中时是符号扩展的。

在RV64中，32位类型，如int，被存储在整数寄存器中作为其32位值的适当符号扩展；也就是说，第63...31位都是相等的。这个限制甚至对无符号的32位类型也是如此。

RV32和RV64
C语言编译器和兼容软件在存储在内存中时保持所有上述数据类型的自然对齐。

C型	描述	RV32中的字节数	RV64中的字节数
炭	字符值/字节	1	1
短期	短整数	2	2
黝黝	整数	4	4
长	长整数	4	8
长长的	长长的整数	8	8
空白的*。	指针	4	8
浮动	单精度浮点数	4	4
双	双精度浮点数	8	8
长双倍	扩展精度的浮点数	16	16

表18.1:基本RISC-V ISA的C编译器数据类型。

18.2 RVG召集大会

RISC-V的调用惯例在可能的情况下用寄存器传递参数。最多八个整数寄存器a0-a7和最多八个浮点寄存器fa0-fa7被用于此目的。

如果函数的参数被概念化为C 结 构 的字段，每个字段都有指针对齐，那么参数寄存器就是该结 构前八个指针字的影子。如果参数$i < 8$是一个浮点类型，它将被传递到浮点寄存器fai中；否则，它将被传递到整数寄存器ai中。然而，作为结构的 联 合或数组字段的一部分的浮点参数是在整数寄存器中传递的。此外，变量函数的浮点参数（除了那些在参数列表中明确命名的参数）是在整数寄存器中传递的。

小于一个指针字的参数在参数寄存器的最小有效位中传递。相应地，在堆栈中传递的子指针字参数出现在指针字的 低位地址中，因为RISC-V有一个小字节的内存系统。

当两倍于指针字大小的原始参数被传递到堆栈中时，它们是自然对齐的。当它们被传递到整数寄存器中时，它们驻留在一个对齐的偶数寄存器对中，偶数寄存器持有最小的有效位。例如，在RV32中，函数void foo(int, long long)在a0中传递第一个参数，在a2和a3中传递第二个参数。在a1 中没有任何东西被传递。

超过指针字大小两倍的参数以引用方式传递。

概念 结 构中没有在参数寄存器中传递的部分被传递到堆栈中。堆栈指针sp指向未在寄存器中传递的第一个参数。

从函数中返回的值是整数寄存器a0和a1以及浮点寄存器fa0和fa1。只有当浮点值是基元或仅由一个或两个浮点值组成的 结 构的成员时，才在浮点寄存器中返回。其他适合两个指针字的返回值会在a0和a1中返回。较大的返回值完全在内存中传递；调用者分配这个内存区域，并将它的指针作为隐含的第一个参

数传递给被调用者。

在标准的RISC-V调用惯例中，堆栈向下增长，堆栈指针总是保持16字节对齐。

除了参数和返回值寄存器，7个整数寄存器t0-t6和12个浮点寄存器ft0-ft11是临时寄存器，在不同的调用中是不稳定的，如果以后使用必须由调用者保存。12个整数寄存器s0-s11和12个浮点寄存器fs0-fs11在不同的调用中是保留的，如果使用的话必须由被调用者保存。表18.2显示了每个整数和浮点寄存器在调用约定中的作用。

注册	ABI名称	描述	拯救者
x0	零	硬连接零	-
x1	筹集资金	返回地址	呼叫者
x2	ÃÃÃ	堆栈指针	卡利
x3	gp	全球指针	-
x4	tp	螺纹指针	-
x5-7	t0-2	暂时性的	呼叫者
x8	s0/fp	保存的寄存器/帧指针	卡利
x9	s1	已保存的登记簿	卡利
x10-11	a0-1	函数参数/返回值	呼叫者
x12-17	a2-7	功能论据	呼叫者
x18-27	s2-11	保存的登记簿	卡利
x28-31	t3-6	暂时性的	呼叫者
f0-7	ft0-7	FP临时工	呼叫者
f8-9	fs0-1	FP保存的寄存器	卡利
f10-11	fa0-1	FP参数/返回值	呼叫者
f12-17	fa2-7	FP论据	呼叫者
f18-27	fs2-11	FP保存的寄存器	卡利
f28-31	ft8-11	FP临时工	呼叫者

表18.2:RISC-V调用约定寄存器的使用。

18.3 软浮动呼叫公约

软浮点调用惯例用于缺乏浮点硬件的RV32和RV64实现。它避免了对F、D和Q标准扩展指令的所有使用，因此也避免了f寄存器的使用。

整数参数的传递和返回方式与RVG惯例相同，堆栈纪律也是如此。浮点参数在整数寄存器中传递和返回，使用相同大小的整数参数的规则。例如，在RV32中，函数double foo(int, double, long double)的第一个参数在a0中传递，第二个参数在a2和a3中传递，第三个参数通过a4引用；其结果在a0和a1中返回。在RV64中，参数在a0、a1和a2-a3对中传递，结果在a0中返回。

动态四舍五入模式和应计异常标志是通过提供的例程访问的

由C99头文件fenv.h提供。