orphan:

Calling Convention Summary

Below is a summary of the calling conventions used on macOS and iOS.

The ABI stability manifesto gives more details on the use of the Swift error return and `self` registers, while The Swift Calling Convention covers the specifics in more details. (The Swift `self` register is known in other documents as the "Context register".)

x86-64

See Apple x86-64 Documentation, System V ABI AMD64 Processor Supplement.

Register usage

Register	Purpose	C++	ObjC	Swift
`rax`	Return value; also, for varargs, number of `xmm` registers used			
`rbx`	Callee-saved register			
`rdi`	Integer argument 1	`this`	`self`	
`rsi`	Integer argument 2		`_cmd`	
`rdx`	Integer argument 3 (2nd return value)			
`rcx`	Integer argument 4 (3rd return value)			
`r8`	Integer argument 5 (4th return value)			
`r9`	Integer argument 6			
`r12`	Callee-saved register			Error return
`r13`	Callee-saved register			`self`
`r14`	Callee-saved register			
`r15`	Callee-saved register (other platforms use as GOT ptr)			
`st0`	Used to return `long double` values			
`st1`	Used to return `long double` values			
`xmm0- xmm7`	Floating point arguments 1-8 (`xmm0-xmm3` also used for return)			
`rsp`	Stack pointer			
`rbp`	Callee-saved register, used as frame pointer			

Stack frame

On function entry, `rsp+8` is **16-byte aligned**, i.e. the start of the memory arguments is 16-byte aligned; the initial stack pointer is shown below as "entry `rsp`", but a typical non-leaf function will start by doing:

```
push %rbp
mov  %rsp, %rbp
sub  <local-size>, %rsp
```

Frameless leaf functions, however, will often not set up the frame pointer, `rbp`, in which case they may refer to arguments relative to `rsp` instead.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\swift-main\docs\ABI\(swift-main)(docs)(ABI)CallConvSummary.rst`, line 88)

Malformed table.

```
+---------------+---------------+-------------------------+
|               | ``rbp+8n+16`` | memory argument *n*     |
|               |               |                         |
|               | ...           | ...                     |
|               |               |                         |
|               | ``rbp+16``    | memory argument 0       |
+---------------+-----------+---+-------------------------+
| â†‘ Current Frame           |           â†“ Previous Frame |
+---------------+-----------+---+-------------------------+
|               | ``rbp+8``     | return address          |
|               |               |                         |
+---------------+---------------+-------------------------+
| entry ``rsp`` | ``rbp``       | previous ``rbp`` value  |
+---------------+---------------+-------------------------+
|               | ``rbp-8``     |                         |
|               |               |                         |
|               | ...           |      local storage      |
|               |               |                         |
|               | ``rsp``       |                         |
+---------------+---------------+-------------------------+
```

```
|               | ``rsp-8``     |                         |
|               |               |                         |
|               | ...           |        red zone         |
|               |               |                         |
|               | ``rsp-128``   |                         |
+---------------+---------------+-------------------------+
```

ARM64

See Apple ARM64 Documentation, Procedure Call Standard for the Arm 64-bit Architecture.

Register usage

Register	Special	Purpose	C++	ObjC	Swift
`x0`		Integer argument 1 (1st return value)	`this`	`self`	
`x1`		Integer argument 2 (2nd return value)		`_cmd`	
`x2`- `x7`		Integer arguments 3-8 (3rd-8th return values)			
`x8`		Indirect result location register			
`x16`	`ip0`	Scratch registers (used by dyld, can be used freely otherwise)			
`x17`	`ip1`				
`x18`		RESERVED **DO NOT USE**			
`x19`		Callee-saved register			
`x20`		Callee-saved register			`self`
`x21`		Callee-saved register			Error return
`x22`- `x28`		Callee-saved registers			
`x29`	`fp`	Frame pointer			
`x30`	`lr`	Link register			
`sp`		Stack pointer			
`v0`- `v7`		Floating point/SIMD arguments 1-8 (also for return)			
`v8`- `v15`		Callee-saved registers (**lower 64-bits only**)			

Stack frame

The stack pointer is **16-byte aligned**; on function entry, `sp` points at the location shown by "entry `sp`" below. As with x86, frameless leaf functions may not set up `fp`, in which case they will use `sp` relative accesses.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\swift-main\docs\ABI\(swift-main)(docs)(ABI)CallConvSummary.rst`, line 180)

Malformed table.

```
+--------------+---------------+-------------------------+
|              | ``fp+8n+16``  | last memory argument    |
|              |               |                         |
|              | ...           | ...                     |
|              |               |                         |
|              | ``fp+16``     | memory argument 0 [1]_ |
+--------------+------------+--+-------------------------+
| â†" Current Frame          |          â†' Previous Frame |
+--------------+------------+--+-------------------------+
| entry ``sp`` | ``fp+8``      | saved ``lr``            |
|              |               | (return address)        |
+--------------+---------------+-------------------------+
|              | ``fp``        | previous ``fp`` value   |
+--------------+---------------+-------------------------+
|              | ``fp-8``      |                         |
|              |               |                         |
|              | ...           |      local storage      |
|              |               |                         |
|              | ``sp``        |                         |
+--------------+---------------+-------------------------+
|              | ``sp-8``      |                         |
|              |               |                         |
|              | ...           |        red zone         |
|              |               |                         |
|              | ``sp-128``    |                         |
+--------------+---------------+-------------------------+
```

[1] See Apple documentation, however. Unlike the official ARM64 ABI, we pack arguments, so this might also hold argument 1, argument 2 and so on.