seL4 IPC Emulation Internals

To implement the seL4 IPC emulation, we need to:

- Pass the **emulated register set values** + the message registers.
- Follow the architecture specific calling conventions and the seL4 semantics.

For example to emulate **`x64_sys_send`**: on x86_64:

```
x64_sys_send(seL4_Word sys, seL4_Word dest, seL4_Word info,
             seL4_Word msg0, seL4_Word msg1, seL4_Word msg2,
             seL4_Word msg3)
{
        register seL4_Word mr0 asm("r10") = msg0;
        register seL4_Word mr1 asm("r8")  = msg1;
        register seL4_Word mr2 asm("r9")  = msg2;
        register seL4_Word mr3 asm("r15") = msg3;
        asm volatile(
          "movq   %%rsp, %%rbx        \n"
          "syscall                    \n"
          "movq   %%rbx, %%rsp        \n"
          :
          : "d"(sys),
          "D"(dest),
          "S"(info),
          "r"(mr0),
          "r"(mr1),
          "r"(mr2),
          "r"(mr3)
          : "%rbx", "%rcx", "%r11"
        );
}
```

`RDI`	stores	**`syscall number`**
`RSI`	stores	**`message info`**
`RDX`	stores	**`capability pointer`**
`R10`	stores	**`message reigister 0`**
`R8`	stores	**`message reigister 1`**
`R9`	stores	**`message reigister 2`**
`R15`	stores	**`message reigister 3`**
`R12`	stores	**`reply`** (only used in MCS configuration)
`IPC Buffer`	stores	**`Other message registers`**