SBI (Supervisor Binary Interface)

This document explains the purpose and inner workings of the SBI (Supervisor Binary Interface) layer used by the Novix OS kernel to communicate with the underlying RISC-V machine mode firmware (OpenSBI).

Function: sbi_call()

```c
struct sbiret sbi_call(
    long arg0, long arg1, long arg2, long arg3,
    long arg4, long arg5, long fid, long eid
);
```

Purpose

Performs a low-level **ecall** from supervisor mode to machine mode.
This function is the core mechanism used by Novix OS to invoke SBI extensions such as setting timers, shutting down, or performing inter-mode communication.

Parameters

Parameter	Type	Description
`arg0` – `arg5`	`long`	General purpose arguments passed to the SBI call. Their meaning depends on the SBI function invoked.
`fid`	`long`	Function ID (within a given SBI extension). Identifies the specific operation to execute.
`eid`	`long`	Extension ID, identifying the SBI extension group (e.g., `SBI_EXT_TIMER`, `SBI_EXT_CONSOLE`).

Return Value

Returns a `struct sbiret`:

```c
struct sbiret {
    long error;   // SBI error code, or 0 if successful
    long value;   // Return value from the SBI function
};
```

Detailed Description

1. Each of the input parameters is placed into the corresponding **RISC-V register** (`a0`–`a7`).
   - Registers `a0`–`a5` carry the general arguments.
   - `a6` holds the Function ID (FID).
   - `a7` holds the Extension ID (EID).
2. The inline assembly instruction `ecall` is then executed.
   This switches from **Supervisor mode** to **Machine mode**, where OpenSBI processes the request.
3. After OpenSBI finishes, it returns two register values:
   - `a0` → `ret.error`
   - `a1` → `ret.value`
4. These are packaged into a `struct sbiret` and returned to the caller.

Use Cases

- `sbi_set_timer()` (see below)
- Console output via `SBI_EXT_CONSOLE`
- Power management or shutdown functions

Function: sbi_set_timer()

```c
void sbi_set_timer(uint64_t stime_value);
```

Purpose

Programs the next **machine timer interrupt** by setting the `stimecmp` register in machine mode through the SBI timer extension.

Parameters

Parameter	Type	Description
`stime_value`	`uint64_t`	The absolute time (in clock ticks) when the next timer interrupt should occur.

Return Value

This function does **not return** a value.
If successful, the system timer interrupt will be scheduled at the given timestamp.

Detailed Description

1. The function internally calls `sbi_call()` with:
   - `arg0 = stime_value`
   - `fid = SBI_SET_TIMER`
   - `eid = SBI_EXT_TIMER`
   - Other arguments (`arg1`–`arg5`) set to `0`.
2. The machine mode (OpenSBI) updates the system's timer comparator, causing a **timer interrupt** when the specified time is reached.
3. This interrupt is handled later by the **kernel's timer interrupt handler**, enabling multitasking and time-based events.

Example Usage

```c
// Schedule an interrupt 1 second from now (assuming 10 MHz clock)
uint64_t current_time = 10000000;
sbi_set_timer(current_time + 10000000);
```

Summary

The SBI interface abstracts away machine-level operations and provides a clean, architecture-defined API for supervisor kernels.
Through `sbi_call()` and helpers like `sbi_set_timer()`, Novix OS can perform privileged tasks such as scheduling, timing, or power control without directly touching machine registers.