Writing Client Drivers

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, **line 3**)

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, **line 3**)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id2" refid="id1">
    :c:type:`struct ssam_controller <ssam_controller>`

.. |ssam_controller| replace:: :c:type:`struct ssam_controller <ssam_controller>`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, **line 4**)

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, **line 4**)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id4" refid="id3">
    :c:type:`struct ssam_device <ssam_device>`

.. |ssam_device| replace:: :c:type:`struct ssam_device <ssam_device>`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, **line 5**)

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, **line 5**)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id6" refid="id5">
    :c:type:`struct ssam_device_driver <ssam_device_driver>`

.. |ssam_device_driver| replace:: :c:type:`struct ssam_device_driver <ssam_device_driver>`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, **line 6**)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, **line 6**)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id8" refid="id7">
    :c:func:`ssam_client_bind`

.. |ssam_client_bind| replace:: :c:func:`ssam_client_bind`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 7)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 7)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id10" refid="id9">
    :c:func:`ssam_client_link`

.. |ssam_client_link| replace:: :c:func:`ssam_client_link`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 8)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 8)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id12" refid="id11">
    :c:func:`ssam_get_controller`

.. |ssam_get_controller| replace:: :c:func:`ssam_get_controller`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 9)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 9)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id14" refid="id13">
    :c:func:`ssam_controller_get`

.. |ssam_controller_get| replace:: :c:func:`ssam_controller_get`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 10)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 10)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id16" refid="id15">
    :c:func:`ssam_controller_put`

.. |ssam_controller_put| replace:: :c:func:`ssam_controller_put`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 11)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 11)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id18" refid="id17">
    :c:func:`ssam_device_alloc`

.. |ssam_device_alloc| replace:: :c:func:`ssam_device_alloc`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 12)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 12)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id20" refid="id19">
    :c:func:`ssam_device_add`

.. |ssam_device_add| replace:: :c:func:`ssam_device_add`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 13)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 13)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id22" refid="id21">
    :c:func:`ssam_device_remove`

.. |ssam_device_remove| replace:: :c:func:`ssam_device_remove`
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 14)

Unknown interpreted text role "c:func".

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation] [driver-api][surface_aggregator]client.rst, line 14)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id24" refid="id23">
    :c:func:`ssam_device_driver_register`

.. |ssam_device_driver_register| replace:: :c:func:`ssam_device_driver_register`
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 15)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 15)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id26" refid="id25">
    :c:func:`ssam_device_driver_unregister`

.. |ssam_device_driver_unregister| replace:: :c:func:`ssam_device_driver_unregister`
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 16)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 16)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id28" refid="id27">
    :c:func:`module_ssam_device_driver`

.. |module_ssam_device_driver| replace:: :c:func:`module_ssam_device_driver`
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 17)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 17)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id30" refid="id29">
    :c:func:`SSAM_DEVICE`

.. |SSAM_DEVICE| replace:: :c:func:`SSAM_DEVICE`
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 18)

Unknown interpreted text role "c:func".

**System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 18)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id32" refid="id31">
    :c:func:`ssam_notifier_register`

.. |ssam_notifier_register| replace:: :c:func:`ssam_notifier_register`
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 19)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 19)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id34" refid="id33">
    :c:func:`ssam_notifier_unregister`

.. |ssam_notifier_unregister| replace:: :c:func:`ssam_notifier_unregister`
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 20)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 20)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id36" refid="id35">
    :c:func:`ssam_request_sync`

.. |ssam_request_sync| replace:: :c:func:`ssam_request_sync`
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 21)

Unknown interpreted text role "c:type".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 21)

Substitution definition contains illegal element <problematic>:

```
<problematic ids="id38" refid="id37">
    :c:type:`enum ssam_event_mask <ssam_event_mask>`

.. |ssam_event_mask| replace:: :c:type:`enum ssam_event_mask <ssam_event_mask>`
```

For the API documentation, refer to:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst, line 30)

Unknown directive type "toctree".

```
.. toctree::
   :maxdepth: 2

   client-api
```

Overview

Client drivers can be set up in two main ways, depending on how the corresponding device is made available to the system. We specifically differentiate between devices that are presented to the system via one of the conventional ways, e.g. as platform devices via ACPI, and devices that are non-discoverable and instead need to be explicitly provided by some other mechanism, as discussed further below.

Non-SSAM Client Drivers

All communication with the SAM EC is handled via the |ssam_controller| representing that EC to the kernel. Drivers targeting a non-SSAM device (and thus not being a |ssam_device_driver|) need to explicitly establish a connection/relation to that controller. This can be done via the |ssam_client_bind| function. Said function returns a reference to the SSAM controller, but, more importantly, also establishes a device link between client device and controller (this can also be done separate via |ssam_client_link|). It is important to do this, as it, first, guarantees that the returned controller is valid for use in the client driver for as long as this driver is bound to its device, i.e. that the driver gets unbound before the controller ever becomes invalid, and, second, as it ensures correct suspend/resume ordering. This setup should be done in the driver's probe function, and may be used to defer probing in case the SSAM subsystem is not ready yet, for example:

```
static int client_driver_probe(struct platform_device *pdev)
{
        struct ssam_controller *ctrl;

        ctrl = ssam_client_bind(&pdev->dev);
        if (IS_ERR(ctrl))
                return PTR_ERR(ctrl) == -ENODEV ? -EPROBE_DEFER : PTR_ERR(ctrl);

        // ...

        return 0;
}
```

The controller may be separately obtained via |ssam_get_controller| and its lifetime be guaranteed via |ssam_controller_get| and |ssam_controller_put|. Note that none of these functions, however, guarantee that the controller will not be shut down or suspended. These functions essentially only operate on the reference, i.e. only guarantee a bare minimum of accessibility without any guarantees at all on practical operability.

Adding SSAM Devices

If a device does not already exist/is not already provided via conventional means, it should be provided as |ssam_device| via the SSAM client device hub. New devices can be added to this hub by entering their UID into the corresponding registry. SSAM devices can also be manually allocated via |ssam_device_alloc|, subsequently to which they have to be added via |ssam_device_add| and eventually removed via |ssam_device_remove|. By default, the parent of the device is set to the controller device provided for allocation, however this may be changed before the device is added. Note that, when changing the parent device, care must be taken to ensure that the controller lifetime and suspend/resume ordering guarantees, in the default setup provided through the parent-child relation, are preserved. If necessary, by use of |ssam_client_link| as is done for non-SSAM client drivers and described in more detail above.

A client device must always be removed by the party which added the respective device before the controller shuts down. Such removal can be guaranteed by linking the driver providing the SSAM device to the controller via |ssam_client_link|, causing it to unbind before the controller driver unbinds. Client devices registered with the controller as parent are automatically removed when the controller shuts down, but this should not be relied upon, especially as this does not extend to client devices with a different parent.

SSAM Client Drivers

SSAM client device drivers are, in essence, no different than other device driver types. They are represented via |ssam_device_driver| and bind to a |ssam_device| via its UID (:c:type:`struct ssam_device.uid <ssam_device>`) member and the match table (:c:type:`struct ssam_device_driver.match_table <ssam_device_driver>`), which should be set when declaring the driver struct instance. Refer to the |SSAM_DEVICE| macro documentation for more details on how to define members of the driver's match table.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 118); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 118); *backlink*

Unknown interpreted text role "c:type".

The UID for SSAM client devices consists of a `domain`, a `category`, a `target`, an `instance`, and a `function`. The `domain` is used differentiate between physical SAM devices (:c:type:`SSAM_DOMAIN_SERIALHUB <ssam_device_domain>`), i.e. devices that can be accessed via the Surface Serial Hub, and virtual ones (:c:type:`SSAM_DOMAIN_VIRTUAL <ssam_device_domain>`), such as client-device hubs, that have no real representation on the SAM EC and are solely used on the kernel/driver-side. For physical devices, `category` represents the target category, `target` the target ID, and `instance` the instance ID used to access the physical SAM device. In addition, `function` references a specific device functionality, but has no meaning to the SAM EC. The (default) name of a client device is generated based on its UID.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 127); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 127); *backlink*

Unknown interpreted text role "c:type".

A driver instance can be registered via |ssam_device_driver_register| and unregistered via |ssam_device_driver_unregister|. For convenience, the |module_ssam_device_driver| macro may be used to define module init- and exit-functions registering the driver.

The controller associated with a SSAM client device can be found in its :c:type:`struct ssam_device.ctrl <ssam_device>` member. This reference is guaranteed to be valid for at least as long as the client driver is bound, but should also be valid for as long as the client device exists. Note, however, that access outside of the bound client driver must ensure that the controller device is not suspended while making any requests or (un-)registering event notifiers (and thus should generally be avoided). This is guaranteed when the controller is accessed from inside the bound client driver.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 145); *backlink*

Unknown interpreted text role "c:type".

Making Synchronous Requests

Synchronous requests are (currently) the main form of host-initiated communication with the EC. There are a couple of ways to define and execute such requests, however, most of them boil down to something similar as shown in the example below. This example defines a write-read request, meaning that the caller provides an argument to the SAM EC and receives a response. The caller needs to know the (maximum) length of the response payload and provide a buffer for it.

Care must be taken to ensure that any command payload data passed to the SAM EC is provided in little-endian format and, similarly, any response payload data received from it is converted from little-endian to host endianness.

```
int perform_request(struct ssam_controller *ctrl, u32 arg, u32 *ret)
{
        struct ssam_request rqst;
        struct ssam_response resp;
        int status;

        /* Convert request argument to little-endian. */
        __le32 arg_le = cpu_to_le32(arg);
        __le32 ret_le = cpu_to_le32(0);

        /*
```

```
         * Initialize request specification. Replace this with your values.
         * The rqst.payload field may be NULL if rqst.length is zero,
         * indicating that the request does not have any argument.
         *
         * Note: The request parameters used here are not valid, i.e.
         *       they do not correspond to an actual SAM/EC request.
         */
        rqst.target_category = SSAM_SSH_TC_SAM;
        rqst.target_id = 0x01;
        rqst.command_id = 0x02;
        rqst.instance_id = 0x03;
        rqst.flags = SSAM_REQUEST_HAS_RESPONSE;
        rqst.length = sizeof(arg_le);
        rqst.payload = (u8 *)&arg_le;

        /* Initialize request response. */
        resp.capacity = sizeof(ret_le);
        resp.length = 0;
        resp.pointer = (u8 *)&ret_le;

        /*
         * Perform actual request. The response pointer may be null in case
         * the request does not have any response. This must be consistent
         * with the SSAM_REQUEST_HAS_RESPONSE flag set in the specification
         * above.
         */
        status = ssam_request_sync(ctrl, &rqst, &resp);

        /*
         * Alternatively use
         *
         *   ssam_request_sync_onstack(ctrl, &rqst, &resp, sizeof(arg_le));
         *
         * to perform the request, allocating the message buffer directly
         * on the stack as opposed to allocation via kzalloc().
         */

        /*
         * Convert request response back to native format. Note that in the
         * error case, this value is not touched by the SSAM core, i.e.
         * 'ret_le' will be zero as specified in its initialization.
         */
        *ret = le32_to_cpu(ret_le);

        return status;
}
```

Note that |ssam_request_sync| in its essence is a wrapper over lower-level request primitives, which may also be used to perform requests. Refer to its implementation and documentation for more details.

An arguably more user-friendly way of defining such functions is by using one of the generator macros, for example via:

```
SSAM_DEFINE_SYNC_REQUEST_W(__ssam_tmp_perf_mode_set, __le32, {
        .target_category = SSAM_SSH_TC_TMP,
        .target_id       = 0x01,
        .command_id      = 0x03,
        .instance_id     = 0x00,
});
```

This example defines a function

```
static int __ssam_tmp_perf_mode_set(struct ssam_controller *ctrl, const __le32 *arg);
```

executing the specified request, with the controller passed in when calling said function. In this example, the argument is provided via the `arg` pointer. Note that the generated function allocates the message buffer on the stack. Thus, if the argument provided via the request is large, these kinds of macros should be avoided. Also note that, in contrast to the previous non-macro example, this function does not do any endianness conversion, which has to be handled by the caller. Apart from those differences the function generated by the macro is similar to the one provided in the non-macro example above.

The full list of such function-generating macros is

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_N` for requests without return value and without argument.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, **line 265**); *backlink*

Unknown interpreted text role "c:func".

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_R` for requests with return value but no argument.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 267); *backlink*

  Unknown interpreted text role "c:func".

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_W` for requests without return value but with argument.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 269); *backlink*

  Unknown interpreted text role "c:func".

Refer to their respective documentation for more details. For each one of these macros, a special variant is provided, which targets request types applicable to multiple instances of the same device type:

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_MD_N`

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 276); *backlink*

  Unknown interpreted text role "c:func".

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_MD_R`

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 277); *backlink*

  Unknown interpreted text role "c:func".

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_MD_W`

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 278); *backlink*

  Unknown interpreted text role "c:func".

The difference of those macros to the previously mentioned versions is, that the device target and instance IDs are not fixed for the generated function, but instead have to be provided by the caller of said function.

Additionally, variants for direct use with client devices, i.e. |ssam_device|, are also provided. These can, for example, be used as follows:

```
SSAM_DEFINE_SYNC_REQUEST_CL_R(ssam_bat_get_sta, __le32, {
        .target_category = SSAM_SSH_TC_BAT,
        .command_id      = 0x01,
});
```

This invocation of the macro defines a function

```
static int ssam_bat_get_sta(struct ssam_device *sdev, __le32 *ret);
```

executing the specified request, using the device IDs and controller given in the client device. The full list of such macros for client devices is:

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_CL_N`

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 304); *backlink*

  Unknown interpreted text role "c:func".

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_CL_R`

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 305); *backlink*
>
> Unknown interpreted text role "c:func".

- :c:func:`SSAM_DEFINE_SYNC_REQUEST_CL_W`

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 306); *backlink*
>
> Unknown interpreted text role "c:func".

Handling Events

To receive events from the SAM EC, an event notifier must be registered for the desired event via |ssam_notifier_register|. The notifier must be unregistered via |ssam_notifier_unregister| once it is not required any more.

Event notifiers are registered by providing (at minimum) a callback to call in case an event has been received, the registry specifying how the event should be enabled, an event ID specifying for which target category and, optionally and depending on the registry used, for which instance ID events should be enabled, and finally, flags describing how the EC will send these events. If the specific registry does not enable events by instance ID, the instance ID must be set to zero. Additionally, a priority for the respective notifier may be specified, which determines its order in relation to any other notifier registered for the same target category.

By default, event notifiers will receive all events for the specific target category, regardless of the instance ID specified when registering the notifier. The core may be instructed to only call a notifier if the target ID or instance ID (or both) of the event match the ones implied by the notifier IDs (in case of target ID, the target ID of the registry), by providing an event mask (see |ssam_event_mask|).

In general, the target ID of the registry is also the target ID of the enabled event (with the notable exception being keyboard input events on the Surface Laptop 1 and 2, which are enabled via a registry with target ID 1, but provide events with target ID 2).

A full example for registering an event notifier and handling received events is provided below:

```
u32 notifier_callback(struct ssam_event_notifier *nf,
                      const struct ssam_event *event)
{
        int status = ...

        /* Handle the event here ... */

        /* Convert return value and indicate that we handled the event. */
        return ssam_notifier_from_errno(status) | SSAM_NOTIF_HANDLED;
}

int setup_notifier(struct ssam_device *sdev,
                   struct ssam_event_notifier *nf)
{
        /* Set priority wrt. other handlers of same target category. */
        nf->base.priority = 1;

        /* Set event/notifier callback. */
        nf->base.fn = notifier_callback;

        /* Specify event registry, i.e. how events get enabled/disabled. */
        nf->event.reg = SSAM_EVENT_REGISTRY_KIP;

        /* Specify which event to enable/disable */
        nf->event.id.target_category = sdev->uid.category;
        nf->event.id.instance = sdev->uid.instance;

        /*
         * Specify for which events the notifier callback gets executed.
         * This essentially tells the core if it can skip notifiers that
         * don't have target or instance IDs matching those of the event.
         */
        nf->event.mask = SSAM_EVENT_MASK_STRICT;

        /* Specify event flags. */
        nf->event.flags = SSAM_EVENT_SEQUENCED;

        return ssam_notifier_register(sdev->ctrl, nf);
}
```

Multiple event notifiers can be registered for the same event. The event handler core takes care of enabling and disabling events when notifiers are registered and unregistered, by keeping track of how many notifiers for a specific event (combination of registry, event target category, and event instance ID) are currently registered. This means that a specific event will be enabled when the first notifier for it is being registered and disabled when the last notifier for it is being unregistered. Note that the event flags are therefore only used on the first registered notifier, however, one should take care that notifiers for a specific event are always registered with the same flag and it is considered a bug to do otherwise.

Docutils System Messages

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 50); *backlink*

Undefined substitution referenced: "ssam_controller".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 50); *backlink*

Undefined substitution referenced: "ssam_device_driver".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 50); *backlink*

Undefined substitution referenced: "ssam_client_bind".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 50); *backlink*

Undefined substitution referenced: "ssam_client_link".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 80); *backlink*

Undefined substitution referenced: "ssam_get_controller".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 80); *backlink*

Undefined substitution referenced: "ssam_controller_get".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 80); *backlink*

Undefined substitution referenced: "ssam_controller_put".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 91); *backlink*

Undefined substitution referenced: "ssam_device".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, line 91); *backlink*

Undefined substitution referenced: "ssam_device_alloc".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 91); *backlink*

Undefined substitution referenced: "ssam_device_add".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 91); *backlink*

Undefined substitution referenced: "ssam_device_remove".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 91); *backlink*

Undefined substitution referenced: "ssam_client_link".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 105); *backlink*

Undefined substitution referenced: "ssam_client_link".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 118); *backlink*

Undefined substitution referenced: "ssam_device_driver".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 118); *backlink*

Undefined substitution referenced: "ssam_device".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 118); *backlink*

Undefined substitution referenced: "SSAM_DEVICE".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 140); *backlink*

Undefined substitution referenced: "ssam_device_driver_register".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 140); *backlink*

Undefined substitution referenced: "ssam_device_driver_unregister".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 140); *backlink*

Undefined substitution referenced: "module_ssam_device_driver".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master] [Documentation] [driver-api] [surface_aggregator]client.rst`, line 231); *backlink*

Undefined substitution referenced: "ssam_request_sync".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, **line 284**); *backlink*

Undefined substitution referenced: "ssam_device".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, **line 312**); *backlink*

Undefined substitution referenced: "ssam_notifier_register".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, **line 312**); *backlink*

Undefined substitution referenced: "ssam_notifier_unregister".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\surface_aggregator\[linux-master][Documentation][driver-api][surface_aggregator]client.rst`, **line 327**); *backlink*

Undefined substitution referenced: "ssam_event_mask".