Table of Contents

Slave Control API

API that connects the master and slave device by using the RS-485 network. The v2 devices will now work as a dummy reader when set as a slave device. It will not store any kind of information for user and all will be stored inside the master device. The slave device will send the information scanned, and the matching and access rule check will be done from the master device. The slave device will only receive the result.

- BS2_GetSlaveDevice: Searches a slave device from the RS-485 network.
- BS2_SetSlaveDevice: Add/Modify/Delete a slave device from the master device.
- BS2_GetSlaveExDevice: In case of CoreStation, searches a slave device from the RS-485 network.
- BS2_SetSlaveExDevice: In case of CoreStation, Add/Modify/Delete a slave device from the master device.
- BS2_SearchDevicesCoreStation: Searches CoreStation devices from the current network.
- BS2_SearchDevicesCoreStationEx: [+ 2.6.3] Searches CoreStation devices from the current network with host IP.
- BS2_GetDevicesCoreStation: Gets searched CoreStation devices.

> **CAUTION**
> When the SDK is initialized, there are no information about the slave devices. Therefore, a slave device must be searched or added before controlling.

Structure

BS2Rs485SlaveDevice

```
typedef struct {
    uint32_t deviceID;
    uint16_t deviceType;
    uint8_t enableOSDP;
    uint8_t connected;
} BS2Rs485SlaveDevice;
```

1. *deviceID*
Device ID.

2. *deviceType*
Device type.

3. *enableOSDP*
Decides whether to use a slave device.

4. *connected*
Displays whether a slave device is connected to the master device.

BS2Rs485SlaveDeviceEX

```
typedef struct {
    uint32_t deviceID;
    uint16_t deviceType;
    uint8_t enableOSDP;
    uint8_t connected;
    uint8_t channelInfo;
    uint8_t reserved[3];
} BS2Rs485SlaveDeviceEX;
```

1. *deviceID*
Device ID.

2. *deviceType*
Device type.

3. *enableOSDP*
Decides whether to use a slave device.

4. *connected*
Displays whether a slave device is connected to the master device.

5. *channelInfo*
Channel value of slave device.

6. *reserved*
Reserved space.

BS2OsdpStandardDevice

```
typedef struct {
    BS2_DEVICE_ID    deviceID;        ///< 4 bytes
    BS2_DEVICE_TYPE    deviceType;        ///< 2 bytes
    BS2_BOOL        enableOSDP;        ///< 1 byte
    BS2_BOOL        connected;        ///< 1 byte

    uint8_t            channelInfo;    ///< 1 byte
    uint8_t            osdpID;            ///< 1 byte
    BS2_BOOL        supremaSearch;    ///< 1 byte
    BS2_BOOL        activate;        ///< 1 byte

    BS2_BOOL        useSecure;        ///< 1 byte
    uint8_t            vendorCode[3];    ///< 3 bytes
```

```
    BS2_VERSION        fwVersion;        ///< 4 bytes

    uint8_t            modelNumber;    ///< 1 byte
    uint8_t            modelVersion;    ///< 1 byte
    BS2_BOOL        readInfo;        ///< 1 byte
    uint8_t            reserved[25];    ///< 25 byte (packing)
} BS2OsdpStandardDevice;         ///< 48 bytes
```

1. *deviceID*
OSDP Device Identifier.

2. *deviceType*
Device type. Usually gets BS2_DEVICE_TYPE_3RD_OSDP_DEVICE.

3. *enableOSDP*
Always set to true.

4. *connected*
If true, an OSDP device is currently connected.

5. *channelInfo*
Connected channel information. `CoreStation40` has 5 channels from 0 to 4, it has channel values within this range.

6. *osdpID*
OSDP Identifier.

7. *supremaSearch*
Information that is distinct from existing RS485 that does not support OSDP, and always set to false.

8. *activate*
Regardless of the connection status of the device, you can set whether the connected device is operating, which indicates the activation status of the operation.

9. *useSecure*
Indicates whether encrypted communication is enabled.
If a separate key is not set, encryption communication is used as the default key, and can be changed with BS2_SetOsdpStandardDeviceSecurityKey.

10. *vendorCode*
Unique value of Vendor.

11. *fwVersion*
FW version information of OSDP device.

12. *modelNumber*
Model number of the OSDP device.

13. *modelVersion*
Model version of the OSDP device.

14. *readInfo*
Indicates whether OSDP device information such as vendorCode, fwVersion, or model has. If it has the OSDP device information, it means that the device has ever been connected to the master device.

15. *reserved*
Reserved Space.

BS2OsdpStandardDeviceAvailable

```
typedef struct {
    uint8_t                   channelIndex;              ///< 1 byte
    BS2_OSDP_CHANNEL_TYPE    channelType;               ///< 1 byte
    uint8_t                   maxOsdpDevice;             ///< 1 byte
    uint8_t                   numOsdpAvailibleDevice;    ///< 1 byte
    BS2_DEVICE_ID            deviceIDs[8];              ///< 4 x 8 = 32
bytes
} BS2OsdpStandardChannelInfo;                            ///< 36 bytes

typedef struct {
    uint8_t                 numOfChannel;            ///< 1 byte
    uint8_t                 reserved[3];              ///< 3 bytes
    BS2OsdpStandardChannelInfo channels[BS2_RS485_MAX_CHANNELS_EX]; ///< 36
x 8 = 288 bytes
    uint8_t                 reserved1[32];           ///< 32 bytes
} BS2OsdpStandardDeviceAvailable;                     ///< 288 bytes + 36
```

1. *channelIndex*
The communication channel number to which the OSDP device is connected.

2. *channelType*
Indicates the type to which the device communicating RS485 is connected.
Based on CoreStation40, there are 5 assignable channels from 0 to 4, and Suprema devices and OSDP devices cannot be mixed and operated within each channel.
If no device is connected to a particular channel, it has a 0 indicating that it can be connected even if it is a Suprema device or an OSDP device.
If a Suprema device is connected to a specific channel, only Suprema devices are allowed to connect to that channel, and channelType has a value of 1. The OSDP device is ignored even if it is connected.
If an OSDP device is connected to a specific channel, only OSDP devices are allowed to connect to that channel, and channelType has a value of 2. The Suprema device is ignored even if it is connected.
Each channel of CoreStation40 can be mixed and operated as Suprema device channel and OSDP device channel.
The maximum number of OSDP devices allowed to connect to a channel is limited to 2, and if the channel is already maxed out, the channelType will be 3, indicating that no more connections are allowed.

Value	Description
0	Normal

Value	Description
1	Suprema Device
2	OSDP Device
3	OSDP Device FULL

3. *maxOsdpDevice*
Indicates the maximum number of devices that can be connected in that channel.
If the channelType is 1, it will get 32, if 2 or 3, it will get 2.

4. *numOsdpAvailibleDevice*
Indicates the number of devices currently available for connection in that channel.

5. *deviceIDs*
The list of Device Identifier that is connected or can be connected in that channel.

6. *numOfChannel*
Number of channel. CoreStation40 has a total of 5 channels.

7. *reserved*
Reserved Space.

8. *channels*
OSDP device information of each channel.
You can have up to 8 channel information, but since CoreStation40 has 5 channels, only numbers 0 to 4 are valid.

9. *reserved1*
Reserved Space.

BS2OsdpStandardNotify

```
typedef struct {
    BS2_DEVICE_ID    deviceID;        ///< 4 bytes
    BS2_DEVICE_TYPE    deviceType;        ///< 2 bytes
    BS2_BOOL        enableOSDP;        ///< 1 byte
    BS2_BOOL        connected;        ///< 1 byte

    uint8_t            channelInfo;    ///< 1 byte
    uint8_t            osdpID;            ///< 1 byte
    BS2_BOOL        supremaSearch;    ///< 1 byte
    BS2_BOOL        activate;        ///< 1 byte

    BS2_BOOL        useSecure;        ///< 1 byte
    uint8_t            vendorCode[3];    ///< 3 bytes

    BS2_VERSION        fwVersion;        ///< 4 bytes

    uint8_t            modelNumber;    ///< 1 byte
```

```
    uint8_t             modelVersion;    ///< 1 byte
    BS2_BOOL         readInfo;        ///< 1 byte
    uint8_t             reserved[5];    ///< 5 bytes (packing)
} BS2OsdpStandardNotify;         ///< 48 bytes
```

1. *deviceID*
OSDP Device Identifier.

2. *deviceType*
Device type. Usually gets BS2_DEVICE_TYPE_3RD_OSDP_DEVICE.

3. *enableOSDP*
Always set to true.

4. *connected*
If true, an OSDP device is currently connected.

5. *channelInfo*
Connected channel information. `CoreStation40` has 5 channels from 0 to 4, it has channel values within this range.

6. *osdpID*
OSDP Identifier.

7. *supremaSearch*
Information that is distinct from existing RS485 that does not support OSDP, and always set to false.

8. *activate*
Regardless of the connection status of the device, you can set whether the connected device is operating, which indicates the activation status of the operation.

9. *useSecure*
Indicates whether encrypted communication is enabled.
If a separate key is not set, encryption communication is used as the default key, and can be changed with BS2_SetOsdpStandardDeviceSecurityKey.

10. *vendorCode*
Unique value of Vendor.

11. *fwVersion*
FW version information of OSDP device.

12. *modelNumber*
Model number of the OSDP device.

13. *modelVersion*
Model version of the OSDP device.

14. *readInfo*
Indicates whether OSDP device information such as vendorCode, fwVersion, or model has. If it has the OSDP device information, it means that the device has ever been connected to the master device.

15. *reserved*
Reserved Space.

BS2OsdpStandardDeviceAdd

```
typedef struct {
    uint8_t         osdpID;                         ///< 1 byte
    uint8_t            activate;                       ///< 1 byte
    uint8_t            useSecureSession;               ///< 1 byte
    uint8_t            deviceType;                     ///< 1 byte
    BS2_DEVICE_ID    deviceID;                        ///< 4 bytes
} BS2OsdpStandardDeviceAdd;                          ///< 8 bytes
```

1. *osdpID*
OSDP Identifier. The identifier must be set to a random value between 0 and 126 by the user.
This value does not allow duplication within the same channel and may raise an error if duplicated or out-of-range values are set.
If the channels are different within the master device, you can add devices by setting them to the same identifier.

2. *activate*
Specifies the device activation state.
Regardless of the connection state of the device, if set to false, the operation of the device is ignored even if it is successfully connected.

3. *useSecureSession*
Specifies whether to encrypt communication.
If a separate key is not set, encryption communication is used as the default key, and can be changed with BS2_SetOsdpStandardDeviceSecurityKey.

4. *deviceType*
Device type. Should be set to BS2_DEVICE_TYPE_3RD_OSDP_DEVICE.

5. *deviceID*
Device Identifier. When set to 0, the master device automatically allocates.

BS2OsdpStandardDeviceUpdate

```
typedef struct {
    uint8_t         osdpID;                         ///< 1 byte
    uint8_t            activate;                       ///< 1 byte
    uint8_t            useSecureSession;               ///< 1 byte
    uint8_t            deviceType;                     ///< 1 byte
    BS2_DEVICE_ID    deviceID;                        ///< 4 bytes
} BS2OsdpStandardDeviceUpdate;                       ///< 8 bytes
```

1. *osdpID*
OSDP Identifier. The identifier must be set to a random value between 0 and 126 by the user.
This value does not allow duplication within the same channel and may raise an error if duplicated or out-of-range values are set.
If the channels are different within the master device, you can add devices by setting them to the same identifier.

2. *activate*
Specifies the device activation state.
Regardless of the connection state of the device, if set to false, the operation of the device is ignored even if it is successfully connected.

3. *useSecureSession*
Specifies whether to encrypt communication.
If a separate key is not set, encryption communication is used as the default key, and can be changed with BS2_SetOsdpStandardDeviceSecurityKey.

4. *deviceType*
Device type. Should be set to BS2_DEVICE_TYPE_3RD_OSDP_DEVICE.

5. *deviceID*
Device Identifier.

BS2OsdpStandardDeviceCapability

```
typedef struct {
    uint8_t                     compliance;
    uint8_t                     count;
} BS2OsdpStandardDeviceCapabilityItem;

typedef struct {
    BS2OsdpStandardDeviceCapabilityItem input;            ///< 2 bytes
    BS2OsdpStandardDeviceCapabilityItem    output;           ///< 2 bytes
    BS2OsdpStandardDeviceCapabilityItem    led;            ///< 2 bytes
    BS2OsdpStandardDeviceCapabilityItem    audio;            ///< 2 bytes
    BS2OsdpStandardDeviceCapabilityItem    textOutput;       ///< 2 bytes
    BS2OsdpStandardDeviceCapabilityItem    reader;           ///< 2 bytes

    uint16_t                recvBufferSize;              ///< 2 bytes
    uint16_t                largeMsgSize;               ///< 2 bytes

    uint8_t                    osdpVersion;              ///< 1 byte
    uint8_t                    cardFormat;                  ///< 1 byte
    uint8_t                    timeKeeping;              ///< 1 byte
    uint8_t                    canCommSecure;               ///< 1 byte

    BS2_BOOL                crcSupport;                  ///< 1 byte
    BS2_BOOL                smartCardSupport;           ///< 1 byte
    BS2_BOOL                biometricSupport;           ///< 1 byte
```

```
    BS2_BOOL                securePinEntrySupport;        ///< 1 byte

    uint8_t                    reserved[4];                ///< 4 bytes
} BS2OsdpStandardDeviceCapability;                        ///< 28 bytes
```

1. *compliance*
Indicates the compliance level according to the function of the PD.
Functions include input, output, led, audio, textOutput, etc. The compliance level is different for each function, so refer to the relevant OSDP document.

2. *count*
It refers to the number of objects according to the function of the PD, and the meaning of the number is different for each function, so refer to the relevant OSDP document.

3. *input*
Define the input (monitoring) function.

4. *output*
Define the output (monitoring) function.

5. *led*
Define the LED function.

6. *audio*
Define the Buzzer function.

7. *textOutput*
Define the text output function.

8. *reader*
Indicates the number of supported devices, only count information has meaning.

9. *recvBufferSize*
Indicates the short message size the PD can receive.

10. *largeMsgSize*
Indicates the maximum size of a long message that the PD can process.

11. *osdpVersion*
OSDP version.

12. *cardFormat*
Defines the card data format function and gets a value of 01, 02, or 03. Please refer to the compliance level related to the card data format of the OSDP document.

13. *timeKeeping*
Indicates the date and time type of the PD and what to keep it for. In OSDP 2.2, this feature is not used.

14. *canCommSecure*
Indicates whether secure communication is supported.

15. *crcSupport*
Indicates whether checksums are supported.

16. *smartCardSupport*
Indicates whether smart cards are supported.

17. *biometricSupport*
Indicates whether biometric processing is supported.

18. *securePinEntrySupport*
Indicates whether Secure PIN Entry (SPE) is supported.

19. *reserved*
Reserved Space.

BS2OsdpStandardDeviceResult

```
typedef struct {
    BS2_DEVICE_ID    deviceID;
    BS2_OSDP_RESULT  result;
} BS2OsdpStandardDeviceResult;
```

1. *deviceID*
Device Identifier.

2. *result*
Gets OSDP device command result value.

Value	Description
0	Success
1	Fail
2	Not available

BS2OsdpStandardDeviceSecurityKey

```
typedef struct {
    uint8_t    key[BS2_OSDP_STANDARD_KEY_SIZE];
    uint8_t    reserved[32];
} BS2OsdpStandardDeviceSecurityKey;
```

1. *key*
16-byte security key used in OSDP device.

2. *reserved*
Reserved Space.