_BS2_AddOsdpStandardDevice_

Slave Control API > BS2_AddOsdpStandardDevice

BS2_AddOsdpStandardDevice

[+ 2.9.1] CoreStation40   OSDP   .
                              BS2_GetAvailableOsdpStandardDevice   BS2_GetOsdpStandardConfig
                    .
                                    OSDP   ,
BS2_GetRS485ConfigEx   .

```
#include "BS_API.h"

int BS2_AddOsdpStandardDevice(void* context, uint32_t deviceId, uint32_t
channelIndex, const BS2OsdpStandardDeviceAdd* osdpDevice, uint32_t*
outChannelIndex);
```

> BS2OsdpStandardDeviceAdd

- [In] *context* : Context
- [In] *deviceId* :
- [In] *channelIndex* :
- [In] *osdpDevice* :   OSDP
- [Out] *outChannelIndex* :   OSDP

                              BS_SDK_SUCCESS   ,
          .

BS2_AddOsdpStandardDevice
BS2_GetOsdpStandardDevice
BS2_GetAvailableOsdpStandardDevice
BS2_UpdateOsdpStandardDevice
BS2_RemoveOsdpStandardDevice
BS2_GetOsdpStandardDeviceCapability

[BS2_SetOsdpStandardDeviceSecurityKey](#)

(C++)

[sample_bs2_addosdpstandarddevice.cpp](#)

```cpp
BS2OsdpStandardDeviceAvailable availDevice = { , };
BS2OsdpStandardDeviceAdd addDevice = { , };

BS2_DEVICE_ID masterID = Utility::selectDeviceID(deviceList, false,
false);

cout << "Now add a OSDP device." << endl;
addDevice.deviceID = Utility::selectSlaveID();

uint32_t channelIndex = ;
for (int idx = ; idx < availDevice.numOfChannel; idx++)
{
    for (int didx = ; didx <
availDevice.channels[idx].numOsdpAvailableDevice; didx++)
    {
        if (availDevice.channels[idx].deviceIDs[didx] ==
addDevice.deviceID)
            channelIndex = availDevice.channels[idx].channelIndex;
    }
}

addDevice.osdpID = (uint8_t)Utility::getInput<uint32_t>("Please enter
the OSDP ID. [0 ~ 126]");
addDevice.useSecureSession = Utility::isYes("Does the OSDP device use
secure communication?");
addDevice.deviceType = BS2_DEVICE_TYPE_3RD_OSDP_DEVICE;
addDevice.activate = 1;

uint32_t outChannelIndex();
int sdkResult = BS2_AddOsdpStandardDevice(context_, masterID,
channelIndex, &addDevice, &outChannelIndex);
if (BS_SDK_SUCCESS != sdkResult)
    printf("BS2_AddOsdpStandardDevice call failed: %d", sdkResult);

return sdkResult;
```


(C#)

[sample_bs2_addosdpstandarddevice.cs](#)

```csharp
BS2OsdpStandardDeviceAdd addDevice =
```

```
Util.AllocateStructure<BS2OsdpStandardDeviceAdd>();

Util.HighlightLine(">>>> Now add a OSDP device.", "add");
addDevice.deviceID = Util.GetInputSlaveID();

UInt32 channelIndex = ;
for (int idx = ; idx < allDevice.numOfChannel; idx++)
{
    for (int didx = ; didx <
allDevice.channels[idx].numOsdpAvailableDevice; didx++)
    {
        if (allDevice.channels[idx].deviceIDs[didx] ==
addDevice.deviceID)
            channelIndex = allDevice.channels[idx].channelIndex;
    }
}

Util.HighlightLine(">>>> Please enter the OSDP ID. [0 ~ 126]", "OSDP
ID");
Console.Write(">>>> ");
addDevice.osdpID = Util.GetInput((byte));

Util.HighlightLine(">>>> Does the OSDP device use secure communication?
[Y/n]", "use secure communication");
Console.Write(">>>> ");
addDevice.useSecureSession = Convert.ToByte(Util.IsYes());

addDevice.deviceType =
Convert.ToByte(BS2DeviceTypeEnum.THIRD_OSDP_DEVICE);
addDevice.activate = Convert.ToByte(true);

UInt32 outChannelIndex = ;
BS2ErrorCode result =
(BS2ErrorCode)API.BS2_AddOsdpStandardDevice(sdkContext, deviceID,
channelIndex, ref addDevice, out outChannelIndex);
if (result != BS2ErrorCode.BS_SDK_SUCCESS)
    Console.WriteLine("Got error({0}).", result);
else
    Console.WriteLine("Call success.");

return result;
```