ZigBee 3.0: Adding Endpoints

This document describes how to add additional endpoints to the Router application in the JN-AN-1217 ZigBee 3.0 Base Device Application Note.

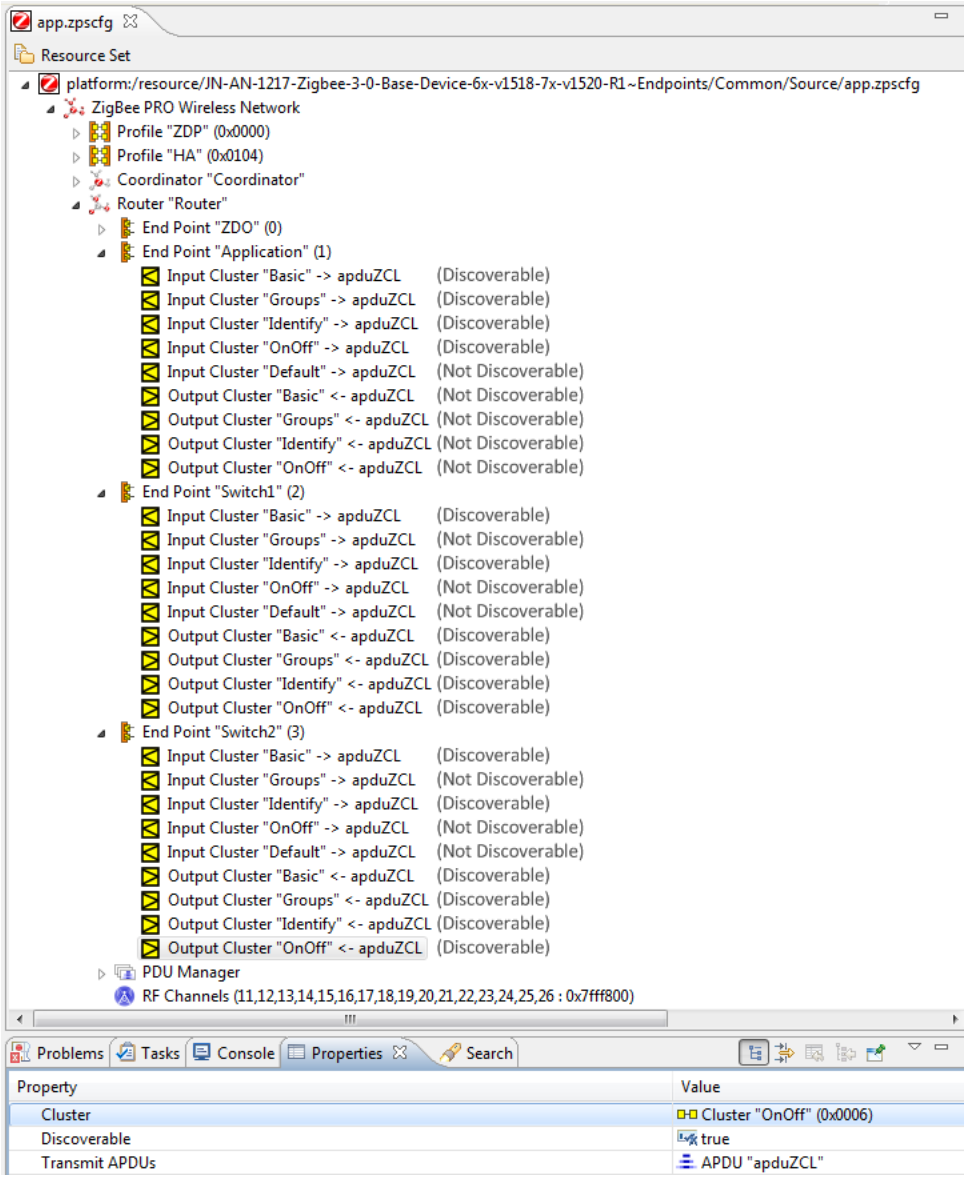
The Router application's main endpoint acts as a light controlled by the On/Off cluster acting as a Server. The steps below describe how to add two new endpoints with On/Off clusters acting as clients.

Note that these changes only go as far as making the new endpoints discoverable, no functionality has been added to read inputs and transmit commands from the new endpoints.

Common/Source/app.zpscfg

The first step is to update the ZigBee PRO Stack Configuration file to add the new endpoints (Switch1, Switch2) and their clusters to the Router application.

The End Device application's endpoint already contains clusters including an On/Off cluster acting as a client the same configuration was used for the Router's new endpoints as shown below:



Router/Source/zcl_options.h

This file is used to set the options used by the ZCL.

Number of Endpoints

The number of endpoints is increased from 1 to 3:

```
/* Number of endpoints supported by this device */
#define ZCL_NUMBER_OF_ENDPOINTS                                 3
```

Enable Client Clusters

The client cluster functionality for the new endpoints is enabled:

```
/****************************************************************************/
/*                            Enable Cluster                                */
/*                                                                          */
/* Add the following #define's to your zcl_options.h file to enable         */
/* cluster and their client or server instances                             */
/****************************************************************************/
#define CLD_BASIC
#define BASIC_SERVER
#define BASIC_CLIENT

#define CLD_IDENTIFY
#define IDENTIFY_SERVER
#define IDENTIFY_CLIENT

#define CLD_GROUPS
#define GROUPS_SERVER
#define GROUPS_CLIENT

#define CLD_ONOFF
#define ONOFF_SERVER
#define ONOFF_CLIENT
```

Router/Source/app_zcl_task.c

Base Device Data Structures

The structures that store data for the new Base Devices associated with the new endpoints are created:

```
/****************************************************************************/
/***        Exported Variables                                            ***/
/****************************************************************************/
tsZHA_BaseDevice sBaseDevice;
tsZHA_BaseDevice sBaseDeviceSwitch1;
tsZHA_BaseDevice sBaseDeviceSwitch2;
```

Register Base Device Endpoints - APP_ZCL_vInitialise()

The two new Base Devices and their endpoints are registered with the stack to make them available:

```
if (eZCL_Status != E_ZCL_SUCCESS)
{
          DBG_vPrintf(TRACE_ZCL, "Error:
eZHA_RegisterBaseDeviceEndPoint(Light): %02x\r\n", eZCL_Status);
}
```

```
/* Register Switch1 EndPoint */
eZCL_Status =  eZHA_RegisterBaseDeviceEndPoint(ROUTER_SWITCH1_ENDPOINT,
                                                &APP_ZCL_cbEndpointCallba
ck,
                                                &sBaseDeviceSwitch1);
if (eZCL_Status != E_ZCL_SUCCESS)
{
          DBG_vPrintf(TRACE_ZCL, "Error:
eZHA_RegisterBaseDeviceEndPoint(Switch1): %02x\r\n", eZCL_Status);
}
/* Register Switch2 EndPoint */
eZCL_Status =  eZHA_RegisterBaseDeviceEndPoint(ROUTER_SWITCH2_ENDPOINT,
                                                &APP_ZCL_cbEndpointCallba
ck,
                                                &sBaseDeviceSwitch2);
if (eZCL_Status != E_ZCL_SUCCESS)
{
          DBG_vPrintf(TRACE_ZCL, "Error:
eZHA_RegisterBaseDeviceEndPoint(Switch2): %02x\r\n", eZCL_Status);
}
```

Factory Reset Functionality - vHandleClusterCustomCommands()

The two new Base Devices are factory reset by re-registering them when the Reset To Factory Defaults command is received by the Basic cluster server:

```
case GENERAL_CLUSTER_ID_BASIC:
{
     tsCLD_BasicCallBackMessage *psCallBackMessage =
(tsCLD_BasicCallBackMessage*)psEvent->uMessage.sClusterCustomMessage.pvCustomData;
     if (psCallBackMessage->u8CommandId ==
E_CLD_BASIC_CMD_RESET_TO_FACTORY_DEFAULTS )
     {
          DBG_vPrintf(TRACE_ZCL, "Basic Factory Reset Received\n");
          memset(&sBaseDevice,0,sizeof(tsZHA_BaseDevice));
          APP_vZCL_DeviceSpecific_Init();
          eZHA_RegisterBaseDeviceEndPoint(ROUTER_APPLICATION_ENDPOINT,
                                          &APP_ZCL_cbEndpointCallback,
                                          &sBaseDevice);
          eZHA_RegisterBaseDeviceEndPoint(ROUTER_SWITCH1_ENDPOINT,
                                          &APP_ZCL_cbEndpointCallback,
                                          &sBaseDeviceSwitch1);
          eZHA_RegisterBaseDeviceEndPoint(ROUTER_SWITCH2_ENDPOINT,
                                          &APP_ZCL_cbEndpointCallback,
                                          &sBaseDeviceSwitch2);
     }
}
break;
```

Basic Server Cluster Data Initialisation -

APP_vZCL_DeviceSpecific_Init()

The default attribute values for the Basic clusters are initialised:

```
sBaseDevice.sOnOffServerCluster.bOnOff = FALSE;
memcpy(sBaseDevice.sBasicServerCluster.au8ManufacturerName, "NXP",
CLD_BAS_MANUF_NAME_SIZE);
memcpy(sBaseDevice.sBasicServerCluster.au8ModelIdentifier, "BDB-Router",
CLD_BAS_MODEL_ID_SIZE);
memcpy(sBaseDevice.sBasicServerCluster.au8DateCode, "20150212", CLD_BAS_DATE_SIZE);
memcpy(sBaseDevice.sBasicServerCluster.au8SWBuildID, "1000-0001",
CLD_BAS_SW_BUILD_SIZE);

sBaseDeviceSwitch1.sOnOffServerCluster.bOnOff = FALSE;
memcpy(sBaseDeviceSwitch1.sBasicServerCluster.au8ManufacturerName, "NXP",
CLD_BAS_MANUF_NAME_SIZE);
memcpy(sBaseDeviceSwitch1.sBasicServerCluster.au8ModelIdentifier, "BDB-Sw1",
CLD_BAS_MODEL_ID_SIZE);
memcpy(sBaseDeviceSwitch1.sBasicServerCluster.au8DateCode, "20170310",
CLD_BAS_DATE_SIZE);
memcpy(sBaseDeviceSwitch1.sBasicServerCluster.au8SWBuildID, "1000-0001",
CLD_BAS_SW_BUILD_SIZE);

sBaseDeviceSwitch2.sOnOffServerCluster.bOnOff = FALSE;
memcpy(sBaseDeviceSwitch2.sBasicServerCluster.au8ManufacturerName, "NXP",
CLD_BAS_MANUF_NAME_SIZE);
memcpy(sBaseDeviceSwitch2.sBasicServerCluster.au8ModelIdentifier, "BDB-Sw2",
CLD_BAS_MODEL_ID_SIZE);
memcpy(sBaseDeviceSwitch2.sBasicServerCluster.au8DateCode, "20170310",
CLD_BAS_DATE_SIZE);
memcpy(sBaseDeviceSwitch2.sBasicServerCluster.au8SWBuildID, "1000-0001",
CLD_BAS_SW_BUILD_SIZE);
```

Router/Source/app_zcl_task.h

The Base Device Data structures are made available to other modules:

```
/****************************************************************************/
/***        Exported Variables                                            ***/
/****************************************************************************/
extern tsZHA_BaseDevice sBaseDevice;
extern tsZHA_BaseDevice sBaseDeviceSwitch1;
extern tsZHA_BaseDevice sBaseDeviceSwitch2;
```

Router/Source/app_router_node.c

Enable ZCL Event Handler - vAppHandleAfEvent()

Data messages addressed to the two new endpoints are passed to the ZCL for processing:

```
if (psZpsAfEvent->u8EndPoint == ROUTER_APPLICATION_ENDPOINT
||  psZpsAfEvent->u8EndPoint == ROUTER_SWITCH1_ENDPOINT
||  psZpsAfEvent->u8EndPoint == ROUTER_SWITCH2_ENDPOINT)
{
     DBG_vPrintf(TRACE_APP, "Pass to ZCL\n");
     if ((psZpsAfEvent->sStackEvent.eType == ZPS_EVENT_APS_DATA_INDICATION) ||
          (psZpsAfEvent->sStackEvent.eType ==
ZPS_EVENT_APS_INTERPAN_DATA_INDICATION))
     {
          APP_ZCL_vEventHandler( &psZpsAfEvent->sStackEvent);
      }
}
```