Setpoint Case Guide (Analog Data Write, Modbus Function Code 10)

This section introduces various setpoint (analog write) scenarios. Each case's project files can be found in the provided "Example Project Files." Choose the closest match and modify as needed.

Case 1: Setpoint for one device with 4 analog setpoints

(1) Modify `main.lua`

Configure `YT_List`

```lua
-- IEC61850 setpoint data point definitions
YT_List =
{
  -- Modbus setpoint control unit #1 (4 setpoints)
  {"RTU/GGIO1.SetPnt1", ".setMag.f"}, -- setpoint, 32-bit float
  {"RTU/GGIO1.SetPnt2", ".setMag.f"}, -- setpoint, 32-bit float
  {"RTU/GGIO1.SetPnt3", ".setMag.f"}, -- setpoint, 32-bit float
  {"RTU/GGIO1.SetPnt4", ".setMag.f"}  -- setpoint, 32-bit float
}
```

Configure `MB_List`

```lua
-- Modbus setpoint data point definitions
MB_List =
{
  -- Modbus setpoint control unit #1 (4 setpoints)
  {
    -- 9600 bps, no parity, 1 stop bit, function code "10", modbus address 0x02, max
response 100 ms, inter-packet 1000 ms
    com = {"BAUDRATE_9600","NoneParity","StopBit_1","10",0x02,100,1000},
    data =
    {
```

```lua
      {"RTU/GGIO1.SetPnt1",0x0000, "F_ABCD"}, -- write two 16-bit registers → 32-bit ABCD
float
      {"RTU/GGIO1.SetPnt2",0x0002, "F_ABCD"}, -- write two 16-bit registers → 32-bit ABCD
float
      {"RTU/GGIO1.SetPnt3",0x0004, "F_ABCD"}, -- write two 16-bit registers → 32-bit ABCD
float
      {"RTU/GGIO1.SetPnt4",0x0006, "F_ABCD"}  -- write two 16-bit registers → 32-bit ABCD
float
    }
  }
}
```

Note: Lua syntax — no trailing comma after the last `}`.

(2) Modify `rtu.cid`

Define logical node

```xml
<LNodeType desc="Setpoint device (4 ch)" id="GGIO_TYPE_4wr" lnClass="GGIO">
  <DO desc="Mode" name="Mod" type="ENC_Mod" />
  <DO desc="Behavior" name="Beh" type="ENS_Beh" />
  <DO desc="Health" name="Health" type="ENS_Health" />
  <DO desc="Nameplate" name="NamPlt" type="LPL_2_NamPlt" />
  <DO desc="Setpoint #1" name="SetPnt1" type="ASG_SetPnt" />
  <DO desc="Setpoint #2" name="SetPnt2" type="ASG_SetPnt" />
  <DO desc="Setpoint #3" name="SetPnt3" type="ASG_SetPnt" />
  <DO desc="Setpoint #4" name="SetPnt4" type="ASG_SetPnt" />
</LNodeType>
```

Add logical node instance

```xml
<LN desc="Setpoint device #1 (4 ch)" lnClass="GGIO" lnType="GGIO_TYPE_4wr" inst="1"
prefix="" />
```

Configure DataSet and ReportControl

```xml
<DataSet name="YT_RR" desc="YT_RR">
  <FCDA ldInst="RTU" lnClass="GGIO" fc="SP" lnInst="1" doName="SetPnt1" daName="setMag.f"
/>
  <FCDA ldInst="RTU" lnClass="GGIO" fc="SP" lnInst="1" doName="SetPnt2" daName="setMag.f"
/>
  <FCDA ldInst="RTU" lnClass="GGIO" fc="SP" lnInst="1" doName="SetPnt3" daName="setMag.f"
/>
  <FCDA ldInst="RTU" lnClass="GGIO" fc="SP" lnInst="1" doName="SetPnt4" daName="setMag.f"
/>
</DataSet>

<ReportControl name="YT_RR" confRev="1" datSet="YT_RR" rptID="YT_RR" buffered="true"
intgPd="1000" bufTime="50">
  <TrgOps period="true" />
  <OptFields seqNum="true" timeStamp="true" dataSet="true" reasonCode="true"
configRef="true" />
  <RptEnabled max="1" />
</ReportControl>
```

DataSet is for the Report service and is optional.

Case 2: Setpoints for multiple devices, each with 4 analog setpoints

Repeat `YT_List`/`MB_List` for devices #1, #2, and #3. Add LN instances `inst="1"`, `inst="2"`, `inst="3"`
and include their `SetPnt1..4` into the `YT_RR` DataSet as shown in the original examples.