Telemetry Case Guide (Analog Acquisition, Modbus Function Code 03/04)

This section introduces various telemetry scenarios. The project files for each case can be found in the provided “Example Project Files.” You can choose the most similar reference according to your needs and modify it accordingly.

Case 1: Acquire one device, each with 2 decimal values (environment temperature & humidity sensor)

(1) Modify `main.lua`

Configure `YC_List`

```
-- IEC61850 telemetry data point definitions
YC_List =
{
  -- Sensor #1 (temperature & humidity)
  {"RTU/GGIO1.AnIn1", ".mag.f", "FLOAT32"}, -- floating point, humidity
  {"RTU/GGIO1.AnIn2", ".mag.f", "FLOAT32"}  -- floating point, temperature
}
```

Configure `MB_List`

```
-- Modbus telemetry data point definitions
MB_List =
{
  -- Sensor #1 (temperature & humidity)
  {
    -- 4800 bps, no parity, 1 stop bit, function code "03", address 0x01, max response
wait 100 ms, inter-packet interval 1000 ms
    com = {"BAUDRATE_4800","NoneParity","StopBit_1","03",0x01,100,1000},
    data =
    {
      {"RTU/GGIO1.AnIn1",0x0000,"S_AB",1}, -- humidity, 1 decimal (S_AB integer × 0.1)
      {"RTU/GGIO1.AnIn2",0x0001,"S_AB",1}  -- temperature, 1 decimal (S_AB integer × 0.1)
    }
  }
}
```

Note: The format above follows the Lua syntax. When adding/removing/editing, do not put a comma after the last closing `}` of any list, whether top-level or nested.

(2) Modify `rtu.cid`

Define logical node (temperature & humidity)

```
<LNodeType desc="Temperature & Humidity Sensor" id="GGIO_TYPE_WSD" 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="Humidity value" name="AnIn1" type="MV_AnIn_Float32" />
      <DO desc="Temperature value" name="AnIn2" type="MV_AnIn_Float32" />
</LNodeType>
```

Add logical node instance

```
<LN desc="Temperature & Humidity Sensor #1" lnClass="GGIO" lnType="GGIO_TYPE_WSD" inst="1"
prefix="" />
```

Configure DataSet

```
<DataSet name="YC_RM" desc="YC_RM">
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn1" daName="mag.f" />
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn2" daName="mag.f" />
</DataSet>
```

DataSet is for the Report service and is optional. Add items that need periodic reporting or reporting on change as required.

Case 2: Acquire multiple devices, each with 2 decimal values (environment temperature & humidity sensors)

(1) Modify `main.lua`

Configure `YC_List`

```
-- IEC61850 telemetry data point definitions
YC_List =
{
  -- Sensor #1 (temperature & humidity)
  {"RTU/GGIO1.AnIn1", ".mag.f", "FLOAT32"}, -- floating point, humidity
  {"RTU/GGIO1.AnIn2", ".mag.f", "FLOAT32"}, -- floating point, temperature
  -- Sensor #2 (temperature & humidity)
  {"RTU/GGIO2.AnIn1", ".mag.f", "FLOAT32"}, -- floating point, humidity
  {"RTU/GGIO2.AnIn2", ".mag.f", "FLOAT32"}, -- floating point, temperature
  -- Sensor #3 (temperature & humidity)
  {"RTU/GGIO3.AnIn1", ".mag.f", "FLOAT32"}, -- floating point, humidity
  {"RTU/GGIO3.AnIn2", ".mag.f", "FLOAT32"}  -- floating point, temperature
}
```

Configure `MB_List`

```
-- Modbus telemetry data point definitions
MB_List =
{
  -- Sensor #1
  {
    -- 4800 bps, no parity, 1 stop bit, function code "03", address 0x01, max response
wait 100 ms, inter-packet interval 1000 ms
    com = {"BAUDRATE_4800","NoneParity","StopBit_1","03",0x01,100,1000},
    data =
    {
      {"RTU/GGIO1.AnIn1",0x0000,"S_AB",1}, -- humidity, 1 decimal (S_AB × 0.1)
      {"RTU/GGIO1.AnIn2",0x0001,"S_AB",1}  -- temperature, 1 decimal (S_AB × 0.1)
    }
  },
  -- Sensor #2
  {
    -- 4800 bps, no parity, 1 stop bit, function code "03", address 0x02, max response
wait 100 ms, inter-packet interval 1000 ms
```

```
    com = {"BAUDRATE_4800","NoneParity","StopBit_1","03",0x02,100,1000},
    data =
    {
      {"RTU/GGIO2.AnIn1",0x0000,"S_AB",1}, -- humidity, 1 decimal (S_AB × 0.1)
      {"RTU/GGIO2.AnIn2",0x0001,"S_AB",1}  -- temperature, 1 decimal (S_AB × 0.1)
    }
  },
  -- Sensor #3
  {
    -- 4800 bps, no parity, 1 stop bit, function code "03", address 0x03, max response
wait 100 ms, inter-packet interval 1000 ms
    com = {"BAUDRATE_4800","NoneParity","StopBit_1","03",0x03,100,1000},
    data =
    {
      {"RTU/GGIO3.AnIn1",0x0000,"S_AB",1}, -- humidity, 1 decimal (S_AB × 0.1)
      {"RTU/GGIO3.AnIn2",0x0001,"S_AB",1}  -- temperature, 1 decimal (S_AB × 0.1)
    }
  }
}
```

Note: The format above follows the Lua syntax. Do not add a trailing comma after the last `}`.

(2) Modify `rtu.cid`

Define logical node (temperature & humidity)

```
<LNodeType desc="Temperature & Humidity Sensor" id="GGIO_TYPE_WSD" 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="Humidity value" name="AnIn1" type="MV_AnIn_Float32" />
      <DO desc="Temperature value" name="AnIn2" type="MV_AnIn_Float32" />
</LNodeType>
```

Add logical node instances

```
<LN desc="Temp/Humidity Sensor #1" lnClass="GGIO" lnType="GGIO_TYPE_WSD" inst="1"
prefix="" />
<LN desc="Temp/Humidity Sensor #2" lnClass="GGIO" lnType="GGIO_TYPE_WSD" inst="2"
prefix="" />
<LN desc="Temp/Humidity Sensor #3" lnClass="GGIO" lnType="GGIO_TYPE_WSD" inst="3"
prefix="" />
```

Configure DataSet

```
<DataSet name="YC_RM" desc="YC_RM">
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn1" daName="mag.f" />
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn2" daName="mag.f" />
</DataSet>
```

DataSet is for the Report service and is optional.

Case 3: Acquire one device, each with 3 decimals and 3 integers (transformer A/B/C phase temperatures)

(1) Modify `main.lua`

Configure `YC_List`

```
-- IEC61850 telemetry data point definitions
YC_List =
{
  -- Transformer #1
  {"RTU/GGIO1.AnIn1", ".mag.f", "FLOAT32"}, -- floating point, phase A temperature
  {"RTU/GGIO1.AnIn2", ".mag.f", "FLOAT32"}, -- floating point, phase B temperature
  {"RTU/GGIO1.AnIn3", ".mag.f", "FLOAT32"},  -- floating point, phase C temperature
  {"RTU/GGIO1.Inc1", ".stVal", "INT32"}, -- integer, phase A sensor signal strength
  {"RTU/GGIO1.Inc2", ".stVal", "INT32"}, -- integer, phase B sensor signal strength
  {"RTU/GGIO1.Inc3", ".stVal", "INT32"}  -- integer, phase C sensor signal strength

}
```

Configure `MB_List`

```
-- Modbus telemetry data point definitions
MB_List =
{
  -- Transformer #1
  {
    -- 115200 bps, no parity, 1 stop bit, function code "03", address 0x02, max response
wait 100 ms, inter-packet interval 1000 ms
    com = {"BAUDRATE_115200","NoneParity","StopBit_1","03",0x02,100,1000},
    data =
    {
      {"RTU/GGIO1.AnIn1",0x0030,"U_AB",2}, -- phase A temperature, 2 decimals (U_AB ×
0.01)
      {"RTU/GGIO1.AnIn2",0x0031,"U_AB",2}, -- phase B temperature, 2 decimals (U_AB ×
0.01)
      {"RTU/GGIO1.AnIn3",0x0032,"U_AB",2}, -- phase C temperature, 2 decimals (U_AB ×
0.01)
      {"RTU/GGIO1.Inc1",0x0060,"U_AB",2}, -- phase A signal strength, integer
      {"RTU/GGIO1.Inc2",0x0061,"U_AB",2}, -- phase B signal strength, integer
      {"RTU/GGIO1.Inc3",0x0062,"U_AB",2}  -- phase C signal strength, integer
    }
  }
}
```

Note: Follow Lua syntax; do not add a trailing comma after the last `}`.

(2) Modify `rtu.cid`

Define logical node (transformer A/B/C phase temperatures)

```
<LNodeType desc="Transformer Temperature Sensor" id="GGIO_TYPE_BYQ" 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="Phase A temperature" name="AnIn1" type="MV_AnIn_Float32" />
  <DO desc="Phase B temperature" name="AnIn2" type="MV_AnIn_Float32" />
  <DO desc="Phase C temperature" name="AnIn3" type="MV_AnIn_Float32" />
  <DO desc="Phase A sensor signal strength" name="Inc1" type="INS_INT32" />
  <DO desc="Phase B sensor signal strength" name="Inc2" type="INS_INT32" />
  <DO desc="Phase C sensor signal strength" name="Inc3" type="INS_INT32" />
</LNodeType>
```

Add logical node instance

```
<LN desc="Transformer #1 Temperature" lnClass="GGIO" lnType="GGIO_TYPE_BYQ" inst="1"
prefix="" />
```

Configure DataSet

```
<DataSet name="YC_RM" desc="YC_RM">
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn1" daName="mag.f" />
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn2" daName="mag.f" />
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn3" daName="mag.f" />
</DataSet>
```

DataSet is optional.

Case 4: Acquire one device with many decimal values (energy meter)

(1) Modify `main.lua`

Configure `YC_List`

```
-- IEC61850 telemetry data point definitions
YC_List =
{
    -- Energy meter #1 (model: TP613)
    {"RTU/GGIO1.AnIn1", ".mag.f", "FLOAT32"}, -- floating point, line voltage Uab
    {"RTU/GGIO1.AnIn2", ".mag.f", "FLOAT32"}, -- floating point, line voltage Ubc
    {"RTU/GGIO1.AnIn3", ".mag.f", "FLOAT32"}, -- floating point, line voltage Uca
    {"RTU/GGIO1.AnIn4", ".mag.f", "FLOAT32"}, -- floating point, line voltage average
ULLAvg
    {"RTU/GGIO1.AnIn5", ".mag.f", "FLOAT32"}, -- floating point, phase voltage Uan
    {"RTU/GGIO1.AnIn6", ".mag.f", "FLOAT32"}, -- floating point, phase voltage Ubn
    {"RTU/GGIO1.AnIn7", ".mag.f", "FLOAT32"}, -- floating point, phase voltage Ucn
```

```
    {"RTU/GGIO1.AnIn8", ".mag.f", "FLOAT32"}, -- floating point, phase voltage average
ULNavg
    {"RTU/GGIO1.AnIn9", ".mag.f", "FLOAT32"}, -- floating point, current Ia
    {"RTU/GGIO1.AnIn10", ".mag.f", "FLOAT32"}, -- floating point, current Ib
    {"RTU/GGIO1.AnIn11", ".mag.f", "FLOAT32"}, -- floating point, current Ic
    {"RTU/GGIO1.AnIn12", ".mag.f", "FLOAT32"}, -- floating point, three-phase current
average IAvg
    {"RTU/GGIO1.AnIn13", ".mag.f", "FLOAT32"}, -- floating point, zero-sequence current In
    {"RTU/GGIO1.AnIn14", ".mag.f", "FLOAT32"}, -- floating point, frequency F
    {"RTU/GGIO1.AnIn15", ".mag.f", "FLOAT32"}, -- floating point, total power factor PF
    {"RTU/GGIO1.AnIn16", ".mag.f", "FLOAT32"}, -- floating point, total active power P
    {"RTU/GGIO1.AnIn17", ".mag.f", "FLOAT32"}, -- floating point, total reactive power Q
    {"RTU/GGIO1.AnIn18", ".mag.f", "FLOAT32"}, -- floating point, total apparent power S
    {"RTU/GGIO1.AnIn19", ".mag.f", "FLOAT32"}, -- floating point, phase-A power factor PFa
    {"RTU/GGIO1.AnIn20", ".mag.f", "FLOAT32"}, -- floating point, phase-B power factor PFb
    {"RTU/GGIO1.AnIn21", ".mag.f", "FLOAT32"}, -- floating point, phase-C power factor PFc
    {"RTU/GGIO1.AnIn22", ".mag.f", "FLOAT32"}, -- floating point, phase-A active power Pa
    {"RTU/GGIO1.AnIn23", ".mag.f", "FLOAT32"}, -- floating point, phase-B active power Pb
    {"RTU/GGIO1.AnIn24", ".mag.f", "FLOAT32"}, -- floating point, phase-C active power Pc
    {"RTU/GGIO1.AnIn25", ".mag.f", "FLOAT32"}, -- floating point, phase-A reactive power
Qa
    {"RTU/GGIO1.AnIn26", ".mag.f", "FLOAT32"}, -- floating point, phase-B reactive power
Qb
    {"RTU/GGIO1.AnIn27", ".mag.f", "FLOAT32"}, -- floating point, phase-C reactive power
Qc
    {"RTU/GGIO1.AnIn28", ".mag.f", "FLOAT32"}, -- floating point, phase-A apparent power
Sa
    {"RTU/GGIO1.AnIn29", ".mag.f", "FLOAT32"}, -- floating point, phase-B apparent power
Sb
    {"RTU/GGIO1.AnIn30", ".mag.f", "FLOAT32"} -- floating point, phase-C apparent power Sc
}
```

Configure `MB_List`

```
-- Modbus telemetry data point definitions
MB_List =
{
  -- Energy meter #1 (TP613)
  {
    -- 115200 bps, no parity, 1 stop bit, function code "03", address 0xff, max response
wait 1000 ms, inter-packet 100 ms
    com = {"BAUDRATE_115200","NoneParity","StopBit_1","03",0xff,1000,100},
    data =
    {
      {"RTU/GGIO1.AnIn1", 0x0064, "F_ABCD", 3}, -- Uab, keep 3 decimals (F_ABCD float)
      {"RTU/GGIO1.AnIn2", 0x0066, "F_ABCD", 3}, -- Ubc, keep 3 decimals
      {"RTU/GGIO1.AnIn3", 0x0068, "F_ABCD", 3}, -- Uca, keep 3 decimals
      {"RTU/GGIO1.AnIn4", 0x006A, "F_ABCD", 3}, -- ULLAvg, keep 3 decimals
      {"RTU/GGIO1.AnIn5", 0x006C, "F_ABCD", 3}, -- Uan, keep 3 decimals
      {"RTU/GGIO1.AnIn6", 0x006E, "F_ABCD", 3}, -- Ubn, keep 3 decimals
      {"RTU/GGIO1.AnIn7", 0x0070, "F_ABCD", 3}, -- Ucn, keep 3 decimals
```

```
      {"RTU/GGIO1.AnIn8", 0x0072, "F_ABCD", 3}, -- ULNavg, keep 3 decimals
      {"RTU/GGIO1.AnIn9", 0x0074, "F_ABCD", 3}, -- Ia, keep 3 decimals
      {"RTU/GGIO1.AnIn10", 0x0076, "F_ABCD", 3}, -- Ib, keep 3 decimals
      {"RTU/GGIO1.AnIn11", 0x0078, "F_ABCD", 3}, -- Ic, keep 3 decimals
      {"RTU/GGIO1.AnIn12", 0x007A, "F_ABCD", 3}, -- IAvg, keep 3 decimals
      {"RTU/GGIO1.AnIn13", 0x007C, "F_ABCD", 3}, -- In, keep 3 decimals
      {"RTU/GGIO1.AnIn14", 0x007E, "F_ABCD", 3}, -- Frequency F, keep 3 decimals
      {"RTU/GGIO1.AnIn15", 0x0080, "F_ABCD", 3}, -- PF, keep 3 decimals
      {"RTU/GGIO1.AnIn16", 0x0082, "F_ABCD", 3}, -- P, keep 3 decimals
      {"RTU/GGIO1.AnIn17", 0x0084, "F_ABCD", 3}, -- Q, keep 3 decimals
      {"RTU/GGIO1.AnIn18", 0x0086, "F_ABCD", 3}, -- S, keep 3 decimals
      {"RTU/GGIO1.AnIn19", 0x0088, "F_ABCD", 3}, -- PFa, keep 3 decimals
      {"RTU/GGIO1.AnIn20", 0x008A, "F_ABCD", 3}, -- PFb, keep 3 decimals
      {"RTU/GGIO1.AnIn21", 0x008C, "F_ABCD", 3}, -- PFc, keep 3 decimals
      {"RTU/GGIO1.AnIn22", 0x008E, "F_ABCD", 3}, -- Pa, keep 3 decimals
      {"RTU/GGIO1.AnIn23", 0x0090, "F_ABCD", 3}, -- Pb, keep 3 decimals
      {"RTU/GGIO1.AnIn24", 0x0092, "F_ABCD", 3}, -- Pc, keep 3 decimals
      {"RTU/GGIO1.AnIn25", 0x0094, "F_ABCD", 3}, -- Qa, keep 3 decimals
      {"RTU/GGIO1.AnIn26", 0x0096, "F_ABCD", 3}, -- Qb, keep 3 decimals
      {"RTU/GGIO1.AnIn27", 0x0098, "F_ABCD", 3}, -- Qc, keep 3 decimals
      {"RTU/GGIO1.AnIn28", 0x009A, "F_ABCD", 3}, -- Sa, keep 3 decimals
      {"RTU/GGIO1.AnIn29", 0x009C, "F_ABCD", 3}, -- Sb, keep 3 decimals
      {"RTU/GGIO1.AnIn30", 0x009E, "F_ABCD", 3}  -- Sc, keep 3 decimals
    }
  }
}
```

(2) Modify `rtu.cid`

Define logical node (energy meter)

```
<LNodeType desc="Energy Meter" id="GGIO_TYPE_DB" 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="Line voltage Uab" name="AnIn1" type="MV_AnIn_Float32" />
  <DO desc="Line voltage Ubc" name="AnIn2" type="MV_AnIn_Float32" />
  <DO desc="Line voltage Uca" name="AnIn3" type="MV_AnIn_Float32" />
  <DO desc="Line voltage average ULLAvg" name="AnIn4" type="MV_AnIn_Float32" />
  <DO desc="Phase voltage Uan" name="AnIn5" type="MV_AnIn_Float32" />
  <DO desc="Phase voltage Ubn" name="AnIn6" type="MV_AnIn_Float32" />
  <DO desc="Phase voltage Ucn" name="AnIn7" type="MV_AnIn_Float32" />
  <DO desc="Phase voltage average ULNavg" name="AnIn8" type="MV_AnIn_Float32" />
  <DO desc="Current Ia" name="AnIn9" type="MV_AnIn_Float32" />
  <DO desc="Current Ib" name="AnIn10" type="MV_AnIn_Float32" />
  <DO desc="Current Ic" name="AnIn11" type="MV_AnIn_Float32" />
  <DO desc="Three-phase current average IAvg" name="AnIn12" type="MV_AnIn_Float32" />
  <DO desc="Zero-sequence current In" name="AnIn13" type="MV_AnIn_Float32" />
  <DO desc="Frequency F" name="AnIn14" type="MV_AnIn_Float32" />
  <DO desc="Total power factor PF" name="AnIn15" type="MV_AnIn_Float32" />
  <DO desc="Total active power P" name="AnIn16" type="MV_AnIn_Float32" />
```

```
  <DO desc="Total reactive power Q" name="AnIn17" type="MV_AnIn_Float32" />
  <DO desc="Total apparent power S" name="AnIn18" type="MV_AnIn_Float32" />
  <DO desc="Phase-A power factor PFa" name="AnIn19" type="MV_AnIn_Float32" />
  <DO desc="Phase-B power factor PFb" name="AnIn20" type="MV_AnIn_Float32" />
  <DO desc="Phase-C power factor PFc" name="AnIn21" type="MV_AnIn_Float32" />
  <DO desc="Phase-A active power Pa" name="AnIn22" type="MV_AnIn_Float32" />
  <DO desc="Phase-B active power Pb" name="AnIn23" type="MV_AnIn_Float32" />
  <DO desc="Phase-C active power Pc" name="AnIn24" type="MV_AnIn_Float32" />
  <DO desc="Phase-A reactive power Qa" name="AnIn25" type="MV_AnIn_Float32" />
  <DO desc="Phase-B reactive power Qb" name="AnIn26" type="MV_AnIn_Float32" />
  <DO desc="Phase-C reactive power Qc" name="AnIn27" type="MV_AnIn_Float32" />
  <DO desc="Phase-A apparent power Sa" name="AnIn28" type="MV_AnIn_Float32" />
  <DO desc="Phase-B apparent power Sb" name="AnIn29" type="MV_AnIn_Float32" />
  <DO desc="Phase-C apparent power Sc" name="AnIn30" type="MV_AnIn_Float32" />
</LNodeType>
```

Add logical node instance

```
<LN desc="Energy Meter #1" lnClass="GGIO" lnType="GGIO_TYPE_DB" inst="1" prefix="" />
```

Configure DataSet

```
<DataSet name="YC_RM" desc="YC_RM">
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn1" daName="mag.f" />
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn2" daName="mag.f" />
  <FCDA ldInst="RTU" lnClass="GGIO" fc="MX" lnInst="1" doName="AnIn3" daName="mag.f" />
</DataSet>
```

DataSet is optional.

Case 5: Acquire multiple devices, each with many decimals (energy meters)

The configuration mirrors Case 4, repeated for devices #1, #2, and #3 in both `YC_List` and `MB_List`, and corresponding LN/instances/DataSet in `rtu.cid`.

Case 6: Acquire mixed devices: 2 temperature/humidity sensors + 1 energy meter + 1 transformer (A/B/C phase temperature)

Provide the combined `YC_List`/`MB_List` and corresponding `rtu.cid` LN types/instances/DataSet entries as shown in the original examples, with comments translated to English.

Note: All the Lua snippets above follow the same no-trailing-comma rule after the last `}`.