

1、首先在后台打开文件 HL6805.xml，不在面板上显示（除非用特定菜单打开该文件，才在前端显示）。新生成文件保存到工程所在文件夹内名称为 new 的文件夹下。

2、寻找到<Descriptions> \<Devices> \<Device Physics="YY"> \<Profile> -> \<DataTypes> \<DataType>，然后所有<DataType>都是成对出现的，在每一对<DataType>里面都包含一个文件头（蓝色圈）和子模块（红色圈），找到

DT1601 数据类型，如下图：SB 输出名字不能改

```
<DataType>
  <Name>DT1601</Name>
  <BitSize>432</BitSize>
  <SubItem>
    <SubIdx>0</SubIdx>
    <Name>SubIndex 000</Name>
    <Type>USINT</Type>
    <BitSize>8</BitSize>
    <BitOffs>0</BitOffs>
    <Flags>
      <Access>ro</Access>
      <Category>o</Category>
    </Flags>
  </SubItem>
  <SubItem>
    <SubIdx>1</SubIdx>
    <Name>SubIndex 001</Name>
    <Type>UDINT</Type>
    <BitSize>8</BitSize>
    <BitOffs>16</BitOffs>
    <Flags>
      <Access>ro</Access>
    </Flags>
  </SubItem>
  <SubItem>
    <SubIdx>2</SubIdx>
    <Name>SubIndex 002</Name>
```

第一个固定的 offset 16

2.1 文件头只需要修改<BitSize>，基本值是 16；每添加一个 HL2001/2/3 模块，就加 8，每减少一个 HL2001 模块就减少 8；同理增加 HL4001 模块，就加 64，减少 HL4001 就减 64；增加 HL4002 也是加 64，减少也减 64；增加 HL5002 就增加 16，减少就减 16。

2.2 然后对于</SubItem>，第一个不要处理，从第二个开始：

每增加一个模块（HL2001/2/3、HL5002）就增加一个</SubItem>；如果增加一个 HL4001/2 模块，就增加 4 个</SubItem>。

然后</SubItem>的属性设置规则如下：

2.2.1 <SubIdx>1</SubIdx>按十进制递增

2.2.2 <Name>SubIndex 001</Name>按十进制递增

2.2.3 <Type>如果是 HL2001/2/3 则为 USINT，如果是 HL5002 则为 UDINT；如果是 HL4001/2 模块产生的 4 个</SubItem>，其<Type>也是 UINT。

2.2.4 <BitSize>如果是 HL2001/2/3 则为 8；如果是 HL5002 或 HL4001/2 模块产生的 4 个</SubItem>，则为 16

2.2.5 <BitOffs> 从第二个</SubItem>（SubIdx 为 1）开始，<BitOffs>为 16，然后第三个开始，<BitOffs> 等于上一个的<BitOffs> 加上<BitSize>。

2.2.6 <Flags>只包含一个属性<Access>，然后<Access>属性一直为 ro

3、找到 DT7010 数据类型，如下图：名字要用 SB 实际模块名称改，只是输出

```

<DataType>
  <Name>DT7010</Name>
  <BitSize>184</BitSize>
  <SubItem>
    <SubIdx>0</SubIdx>
    <Name>SubIndex 000</Name>
    <Type>USINT</Type>
    <BitSize>8</BitSize>
    <BitOffs>8</BitOffs>
    <Flags>
      <Access>ro</Access>
      <Category>o</Category>
    </Flags>
  </SubItem>
  <SubItem>
    <SubIdx>1</SubIdx>
    <Name>DO_1stByte</Name>
    <Type>UDINT</Type>
    <BitSize>8</BitSize>
    <BitOffs>16</BitOffs>
    <Flags>
      <Access>ro</Access>
      <Category>o</Category>
    </Flags>
  </SubItem>
  <SubItem>
    <SubIdx>2</SubIdx>

```

3.1 红色圈为文件头，只需要修改<BitSize>，基本值是 16；每添加一个 HL2001/2/3 模块，就加 8，每减少一个 HL2001 模块就减少 8；同理增加 HL4001 模块，就加 64，减少 HL4001 就减 64；增加 HL4002 也是加 64，减少也减 64；增加 HL5002 就增加 16，减少就减 16。

3.2 然后对于</SubItem>，第一个不要处理，从第二个开始：

每增加一个模块（HL2001/2/3、HL5002）就增加一个</SubItem>；如果增加一个 HL4001/2 模块，就增加 4 个</SubItem>。

然后</SubItem>的属性设置规则如下：

3.2.1 <SubIdx>1</SubIdx>按十进制递增

3.2.2 <Name> DO_05A_1stByte</Name>按实际组态的模块来命名，具体如下：

HL2001 的名字 DO_05A_nstByte(名字按照 05.08 定义的写)

HL2002 的名字 DO_15A_nstByte(n=1、2...实际插入的第几个 HL2002 模块)

HL2003 的名字 DO_20A_nstByte(n=1、2...实际插入的第几个 HL2003 模块)

HL4001 的 4 个 SubItem 名字 AOn_010V_Ch1、AOn_010V_Ch2、
AOn_010V_Ch3、AOn_010V_Ch4(n 表示插入的第几个 HL4001 模块)

HL4002 的 4 个 SubItem 名字 AOn_420mA_Ch1、AOn_420mA_Ch2、AOn_420mA_Ch3、AOn_420mA_Ch4(n 表示插入的第几个 HL4002 模块)

HL5002 的名字 Abs_Encode_n_Cmd(n 表示插入的第几个 HL5002 模块)

3.2.3 <Type>如果是 HL2001/2/3 则为 USINT，如果是 HL5002 则为 UINT；如果是 HL4001/2 模块产生的 4 个</SubItem>，其<Type>也是 UINT。

3.2.4 <BitSize>如果是 HL2001/2/3 则为 8；如果是 HL5002 和 HL4001/2 模块产生的 4 个</SubItem>，则为 16

3.2.5 <BitOffs> 从第二个</SubItem> (SubIdx 为 1) 开始，<BitOffs>为 16，然后第三个开始，<BitOffs> 等于上一个的<BitOffs> 加上<BitSize>。

3.2.6 <Flags>的 2 个属性<Access> (一直为 ro) 和<Category> (一直为 o)

4、寻找到<Descriptions> \<Devices> \<Device Physics="YY"> \<Profile> -> \<Objects>\<Object>，然后所有<Object>都是成对出现的，在每一对

<Object> 里面都包含一个文件头（蓝色圈）和子模块（红色圈），

4.1 找到**#x1601** 数据类型，如下图：

4.1.1 前三个属性不需要修改。

4.1.2 <BitSize>408</BitSize> 按实际添加的模块类型计算（有一个基本值 16）：
HL2001/2/3 则为 8；如果 HL5002 和 HL4001/2 模块产生的 4 个</SubItem>则为 16

4.1.3 第一个<SubItem>的<DefaultData>值等于后面所有<SubItem>的数量（十进制）。

4.1.4 从第二个<SubItem>开始，<Name>逐渐累加，<DefaultData>长度固定，每增加一个模块（HL2001/2/3、HL5002）就增加一个</SubItem>；如果增加一个 HL4001/2 模块，就增加 4 个</SubItem>。定义如下（全部按照**十六进制**）：

4.1.4.1 第一个字节就是模块大小：如果是 HL2001/2/3 则为 8；如果是 HL5002 和 HL4001/2 模块产生的 4 个</SubItem>，则为 16。（十六进制）

4.1.4.2 第二个字节从 01 开始累加；

4.1.4.3 第三第四字节不需要更改，固定为 1070 **第一个不改，**

```

<Object>
  <Index>#x1601</Index>
  <Name>Outputs</Name>
  <Type>DT1601</Type>
  <BitSize>408</BitSize>
  <Info>
    <SubItem>
      <Name>SubIndex 000</Name>
      <Info>
        <DefaultData>13</DefaultData>
      </Info>
    </SubItem>
    <SubItem>
      <Name>SubIndex 001</Name>
      <Info>
        <DefaultData>08011070</DefaultData>
        <!--Reference to 0x7010.1-->
      </Info>
    </SubItem>
  </Info>
</Object>

```

4.2 找到 **<Object>#x7010** 对象类型，如下图：

4.1 文件头只需要修改<BitSize>，基本值是 16；每添加一个 HL2001/2/3 模块，就加 8，每减少一个 HL2001 模块就减少 8；同理增加 HL4001 模块，就加 64，减少 HL4001 就减 64；增加 HL4002 也是加 64，减少也减 64；增加 HL5002 就增加 16，减少就减 16。

4.3 然后对于<Info>下的</SubItem>，第一个不要处理，从第二个开始：

第一个 SubItem 的 DefaultData 值等于后面的 SubItem 个数。

每增加一个模块（HL2001/2/3、HL5002）就增加一个</SubItem>；如果增加一个 HL4001/2 模块，就增加 4 个</SubItem>。

然后</SubItem>的属性设置规则如下：

4.3.1 <Name>的命名规则：

HL2001 的名字 DO_05A_1stByte

HL2002 的名字 DO_15A_1stByte

HL2003 的名字 DO_20A_1stByte

HL4001 的 4 个 SubItem 名字 AO_010V_Ch1、AO_010V_Ch2、AO_010V_Ch3、AO_010V_Ch4

HL4002 的 4 个 SubItem 名字 AO_420mA_Ch1、AO_420mA_Ch2、
AO_420mA_Ch3、AO_420mA_Ch4
HL5002 的名字 Abs_Encode_Cmd

4.3.2 另外一个属性保持如下：8 位两个 0，16 位四个 0，32 位八个 0

```
<Info>
  <DefaultData>00</DefaultData>
</Info>

<Object>
  <Index>#x7010</Index>
  <Name>RemoteIO_Output</Name>
  <Type>DT7010</Type>
  <BitSize>184</BitSize>
  <Info>
    <SubItem>
      <Name>SubIndex 000</Name>
      <Info>
        <DefaultData>06</DefaultData>
      </Info>
    </SubItem>
    <SubItem>
      <Name>DO_05A_1stByte</Name>
      <Info>
        <DefaultData>00</DefaultData>
      </Info>
    </SubItem>
    <SubItem>
      <Name>DO_15A_1stByte</Name>
      <Info>
        <DefaultData>00</DefaultData>
      </Info>
    </SubItem>
  </Info>
</Object>
```

5、寻找到<Descriptions>\<Devices>\<Device Physics="YY">\<Profile>->

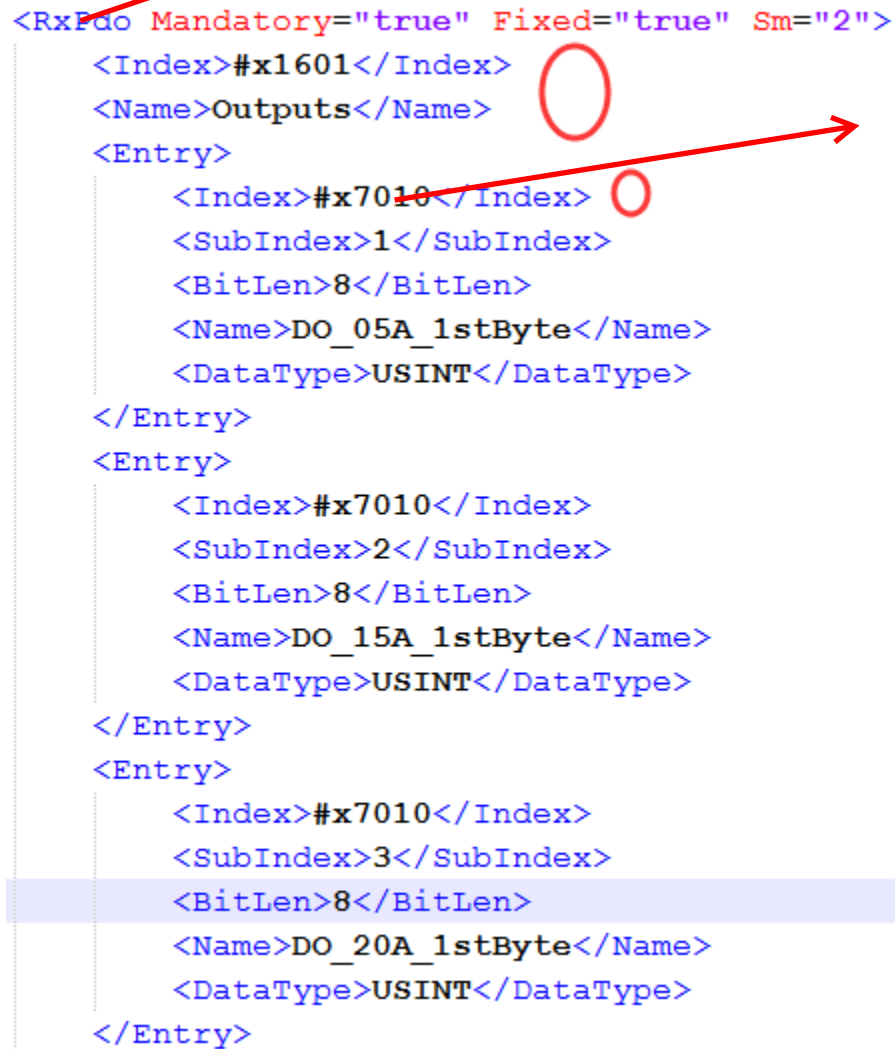
\<RxPdo Mandatory="true" Fixed="true" Sm="2">，然后所有<Object>都是成对出现的，在每一对<Object>里面都包含一个文件头和 ENTRY，找到#x1601，如下图：

5.1 红色圈不变

5.2 <SubIndex>从 1 开始逐渐递增

5.3 <BitLen>、<Name>、<DataType>根据实际插入模块来决定，具体参见前面的说明

5.4



```
<RxPdo Mandatory="true" Fixed="true" Sm="2">
  <Index>#x1601</Index>
  <Name>Outputs</Name>
  <Entry>
    <Index>#x7010</Index>
    <SubIndex>1</SubIndex>
    <BitLen>8</BitLen>
    <Name>DO_05A_1stByte</Name>
    <DataType>USINT</DataType>
  </Entry>
  <Entry>
    <Index>#x7010</Index>
    <SubIndex>2</SubIndex>
    <BitLen>8</BitLen>
    <Name>DO_15A_1stByte</Name>
    <DataType>USINT</DataType>
  </Entry>
  <Entry>
    <Index>#x7010</Index>
    <SubIndex>3</SubIndex>
    <BitLen>8</BitLen>
    <Name>DO_20A_1stByte</Name>
    <DataType>USINT</DataType>
  </Entry>
</RxPdo>
```

6、寻找到<Descriptions> \ <Devices> \ <Device Physics="YY"> \ <Profile> -> \ <DataTypes> \ <DataType>, 然后所有<DataType>都是成对出现的, 在每一对<DataType>里面都包含一个文件头 (蓝色圈) 和子模块 (红色圈),

找到 **DT1A00** 数据类型, 如下图:

```
<DataType>
  <Name>DT1A00</Name>
  <BitSize>336</BitSize>
  <SubItem>
    <SubIdx>0</SubIdx>
    <Name>SubIndex 000</Name>
    <Type>USINT</Type>
    <BitSize>8</BitSize>
    <BitOffs>0</BitOffs>
    <Flags>
      <Access>ro</Access>
      <Category>o</Category>
    </Flags>
  </SubItem>
  <SubItem>
    <SubIdx>1</SubIdx>
    <Name>SubIndex 001</Name>
    <Type>UDINT</Type>
    <BitSize>32</BitSize>
    <BitOffs>16</BitOffs>
    <Flags>
      <Access>ro</Access>
    </Flags>
  </SubItem>
  <SubItem>
    <SubIdx>2</SubIdx>
    <Name>SubIndex 002</Name>
```

6.1 文件头只需要修改<BitSize>, 基本值是 16; 每添加一个 HL1001 模块, 就加 8, 每减少一个 HL1001 模块就减少 8; 同理增加 HL3001 模块, 就加 64, 减少 HL3001 就减 64; 增加 HL3002 和 HL5001 也是加 64, 减少也减 64; 增加 HL5002 就增加 32, 减少就减 32。

6.2 然后对于</SubItem>, 第一个不要处理, 从第二个开始:

每增加一个模块 HL1001 和 HL5002 就增加一个</SubItem>; 如果增加一个 HL3001/2 模块, 就增加 4 个</SubItem>; 增加一个 HL5001 就增加 2 个</SubItem>;

然后</SubItem>的属性设置规则如下:

6.2.1 <SubIdx>**1**</SubIdx>按十进制递增

6.2.2 <Name>SubIndex **001**</Name>按十进制递增

6.2.3 <Type>如果是 HL1001 则为 USINT, 如果是 HL5002 则为 UDINT; 如果是 HL3001/2 模块产生的 4 个</SubItem>, 其<Type>是 UINT。

6.2.4 <BitSize>如果是 HL1001 则为 8; 如果 HL5002 是 32; HL3001/2 模块产生的 4 个</SubItem>, 则为 16; 如果是 HL5001 产生的 2 个, 都是 32;

6.2.5 <BitOffs> 从第二个</SubItem> (SubIdx 为 1) 开始, <BitOffs>为 16, 然后第三个开始, <BitOffs> 等于上一个的<BitOffs> 加上<BitSize>。

6.2.6 <Flags>只包含一个属性<Access>, 然后<Access>属性一直为 ro

7、找到 **DT6001** 数据类型, 如下图:

```

<DataType>
  <Name>DT6001</Name>
  <BitSize>208</BitSize>
  <SubItem>
    <SubIdx>0</SubIdx>
    <Name>SubIndex 000</Name>
    <Type>USINT</Type>
    <BitSize>8</BitSize>
    <BitOffs>8</BitOffs>
    <Flags>
      <Access>ro</Access>
      <Category>o</Category>
    </Flags>
  </SubItem>
  <SubItem>
    <SubIdx>1</SubIdx>
    <Name>DI_1stByte</Name>
    <Type>UDINT</Type>
    <BitSize>32</BitSize>
    <BitOffs>16</BitOffs>
    <Flags>
      <Access>ro</Access>
      <Category>o</Category>
    </Flags>
  </SubItem>
  <SubItem>
    <SubIdx>2</SubIdx>

```

7.1 红色圈为文件头，只需要修改<BitSize>，基本值是 16；每添加一个 HL1001 模块，就加 8，每减少一个 HL1001 模块就减少 8；同理增加 HL3001 模块，就加 64，减少 HL3001 就减 64；增加 HL3002 也是加 64，减少也减 64；增加 HL5002 就增加 32，减少就减 32。如果是 HL5001，则是 64；

7.2 然后对于</SubItem>，第一个不要处理，从第二个开始：

每增加一个模块（HL1001、HL5002）就增加一个</SubItem>；如果增加一个 HL3001/2 模块，就增加 4 个</SubItem>；如果增加一个 HL5001，则增加 2 个。然后</SubItem>的属性设置规则如下：

7.2.1 <SubIdx>1</SubIdx>按十进制递增

7.2.2 <Name> DI_1stByte</Name>按实际组态的模块来命名，具体如下：

HL1001 的名字 DI_nstByte(n=1、2...实际插入的第几个 HL1001 模块)

HL5002 的名字 AbsEncodeSSI_n(n=1、2...实际插入的第几个 HL5002 模块)

HL5001 的名字 Counter600_1_n，Counter600_2_n(n=1、2...实际插入的第几个 HL5001 模块)

HL3001 的 4 个 SubItem 名字 AIn_010V_Ch1、AIn_010V_Ch2、AIn_010V_Ch3、AIn_010V_Ch4, (n=1、2...实际插入的第几个 HL3001 模块)

HL3002 的 4 个 SubItem 名字 AIn_420mA_Ch1、AIn_420mA_Ch2、AIn_420mA_Ch3、AIn_420mA_Ch4(n=1、2...实际插入的第几个 HL3002 模块)

7.2.3 <Type>如果是 HL1001 则为 USINT, 如果是 HL5002 则为 UDINT; 如果是 HL3001/2 模块产生的 4 个</SubItem>, 其<Type>也是 UINT; 如果是 HL5001 的两个模块, 则为 UDINT。

7.2.4 <BitSize>如果是 HL1001 则为 8; 如果 HL5002 是 32; HL3001/2 模块产生的 4 个</SubItem>, 则为 16; 如果是 HL5001 产生的 2 个, 都是 32;

7.2.5 <BitOffs> 从第二个</SubItem> (SubIdx 为 1) 开始, <BitOffs>为 16, 然后第三个开始, <BitOffs> 等于上一个的<BitOffs> 加上<BitSize>。

7.2.6 <Flags>的 2 个属性<Access> (一直为 ro) 和<Category> (一直为 o)

8、寻找到<Descriptions> \<Devices> \<Device Physics="YY"> \<Profile> ->

\<Objects>\<Object>, 然后所有<Object>都是成对出现的, 在每一对<Object>里面都包含一个文件头 (蓝色圈) 和子模块 (红色圈), 找到

#x1A00, 如下图:

8.1.1 前三个属性不需要修改。

8.1.2 <BitSize>408</BitSize> 按实际添加的模块类型计算 (有一个基本值 16): HL1001 则为 8; 如果 HL5002 则是 32; HL3001/2 模块产生的 4 个</SubItem>, 则为 16; HL5001 产生的 2 个都是 32。

8.1.3 第一个<SubItem>的<DefaultData>值等于后面所有<SubItem>的数量 (十进制)。

8.1.4 从第二个<SubItem>开始, <Name>逐渐累加, <DefaultData>长度固定, 每增加一个模块 (HL1001、HL5002) 就增加一个</SubItem>; 如果增加一个 HL3001/2 模块, 就增加 4 个</SubItem>。增加一个 HL5001 就增加 2 个。定义如下 (全部按照十六进制):

8.1.4.1 第一个字节就是模块大小: 如果是 HL1001 则为 8; 如果是 HL5002 和 HL5001 产生的 2 个</SubItem>, 则为 32。而 HL3001/2 模块产生的 4 个</SubItem>, 则为 16。

8.1.4.2 第二个字节从 01 开始累加;

8.1.4.3 第三第四字节不需要更改, 固定为 0160

注释直接不要!

```

<Object>
  <Index>#x1A00</Index>
  <Name>Inputs</Name>
  <Type>DT1A00</Type>
  <BitSize>336</BitSize>
  <Info>
    <SubItem>
      <Name>SubIndex 000</Name>
      <Info>
        <DefaultData>10</DefaultData>
      </Info>
    </SubItem>
    <SubItem>
      <Name>SubIndex 001</Name>
      <Info>
        <DefaultData>20010160</DefaultData>
        <!--Reference to 0x6001.0-->
      </Info>
    </SubItem>
    <SubItem>
      <Name>SubIndex 002</Name>
      <Info>
        <DefaultData>20020160</DefaultData>
        <!--Reference to 0x6001.0-->
      </Info>
    </SubItem>
  </Info>

```

8.2 找到#x6001 数据类型，如下图：

8.3 文件头只需要修改<BitSize>，基本值是 16；每添加一个 HL1001 模块，就加 8，每减少一个 HL1001 模块就减少 8；同理增加 HL3001 模块，就加 64，减少 HL3001 就减 64；增加 HL3002 也是加 64，减少也减 64；增加 HL5002 就增加 32，减少就减 32。如果增加 HL5001 增加 64，减少也是 64。

8.2 然后对于<Info>下的</SubItem>，第一个不要处理，从第二个开始：

每增加一个模块（HL1001 就增加一个</SubItem>；如果增加一个 HL3001/2 模块，就增加 4 个</SubItem>，如果增加一个 HL5001 就增加 2 个；如果增加 HL5002 就增加 1 个。

然后</SubItem>的属性设置规则如下：

8.2.1 <Name>的命名规则：

HL1001 的名字 DI_1stByte，第二个就是 DI_2stByte

HL5002 的名字 AbsEncodeSSI_1, 第二个同上

HL5001 的名字 Counter600_1_01, Counter600_1_02, 第二个同上

HL3001 的 4 个 SubItem 名字 AI1_010V_Ch1、AI1_010V_Ch2、
AI1_010V_Ch3、AI1_010V_Ch4, 第二个同上

HL3002 的 4 个 SubItem 名字 AI1_420mA_Ch1、AI1_420mA_Ch2、
AI1_420mA_Ch3、AI1_420mA_Ch4

8.2.2 另外一个属性保持如下: 注意 8 位 00, 16 位 0000, 32 位 00000000

```
<Info>
  <DefaultData>00</DefaultData>
</Info>

<Object>
  <Index>#x6001</Index>
  <Name>RemoteIO_Input</Name>
  <Type>DT6001</Type>
  <BitSize>208</BitSize>
  <Info>
    <SubItem>
      <Name>SubIndex 000</Name>
      <Info>
        <DefaultData>06</DefaultData>
      </Info>
    </SubItem>
    <SubItem>
      <Name>DI_1stByte</Name>
      <Info>
        <DefaultData>00</DefaultData>
      </Info>
    </SubItem>
    <SubItem>
      <Name>Joint2_Pos_Fb</Name>
      <Info>
        <DefaultData>00000000</DefaultData>
      </Info>
    </SubItem>
  </Info>
</Object>
```

9、寻找到<Descriptions> \<Devices> \<Device Physics="YY"> \<Profile> ->
<TxPdo Mandatory="true" Fixed="true" Sm="3">, 然后所有<Entry>都是成对出现

的, 在每一对<Entry>里面都包含一个文件头, 找到 **x1A00**, 如下图:

9.1 红色圈不变

9.2 <SubIndex>从 1 开始逐渐递增

9.3 <BitLen>、<Name>、<DataType>根据实际插入模块来决定, 具体参见前面说明

```
<TxPdo Mandatory="true" Fixed="true" Sm="3">  
  <Index>#x1A00</Index>  
  <Name>Inputs</Name>  
  <Entry>  
    <Index>#x6001</Index>  
    <SubIndex>1</SubIndex>  
    <BitLen>32</BitLen>  
    <Name>DI_1stByte</Name>  
    <DataType>USINT</DataType>  
  </Entry>  
  <Entry>  
    <Index>#x6001</Index>  
    <SubIndex>2</SubIndex>  
    <BitLen>32</BitLen>  
    <Name>Joint2_Pos_Fb</Name>  
    <DataType>UDINT</DataType>  
  </Entry>  
  <Entry>  
    <Index>#x6001</Index>  
    <SubIndex>3</SubIndex>  
    <BitLen>32</BitLen>  
    <Name>Joint3_Pos_Fb</Name>  
    <DataType>UDINT</DataType>  
  </Entry>  
  ...  
</TxPdo>
```

10、给一个例子

EtherCAT 耦合器，总线为 EtherCAT Slave，后面按顺序挂 4 个 HL1001，4 个 HL2001，2 个 HL2002，2 个 HL2003，2 个 HL3001，2 个 HL3002，2 个 HL4001，2 个 HL4002，2 个 HL5001，2 个 HL5002 模块，共计 24 个模块。经过组态软件组态后，生成的 xml 文件为 HL6805.xml

实际挂载模块	组态软件上显示	生成的 HL6805.xml 在 EC 主站软件上显示
HL1001	HL1001_1	DI_1stByte
HL1001	HL1001_2	DI_2stByte
HL1001	HL1001_3	DI_3stByte
HL1001	HL1001_4	DI_4stByte
HL2001	HL2001_1	DO_05A_1stByte
HL2001	HL2001_2	DO_05A_2stByte
HL2001	HL2001_3	DO_05A_3stByte
HL2001	HL2001_4	DO_05A_4stByte
HL2002	HL2002_1	DO_15A_1stByte
HL2002	HL2002_2	DO_15A_2stByte
HL2003	HL2003_1	DO_20A_1stByte
HL2003	HL2003_2	DO_20A_2stByte
HL3001	HL3001_1	AI1_010V_Ch1
		AI1_010V_Ch2
		AI1_010V_Ch3
		AI1_010V_Ch4
HL3001	HL3001_2	AI2_010V_Ch1
		AI2_010V_Ch2
		AI2_010V_Ch3
		AI2_010V_Ch4
HL3002	HL3002_1	AI1_420mA_Ch1
		AI1_420mA_Ch2
		AI1_420mA_Ch3
		AI1_420mA_Ch4
HL3002	HL3002_2	AI2_420mA_Ch1
		AI2_420mA_Ch2
		AI2_420mA_Ch3
		AI2_420mA_Ch4
HL4001	HL4001_1	AO1_010V_Ch1
		AO1_010V_Ch2
		AO1_010V_Ch3
		AO1_010V_Ch4
HL4001	HL4001_2	AO2_010V_Ch1
		AO2_010V_Ch2
		AO2_010V_Ch3
		AO2_010V_Ch4
HL4002	HL4002_1	AO1_420mA_Ch1
		AO1_420mA_Ch2

		AO1 420mA Ch3
		AO1 420mA Ch4
HL4002	HL4002_2	AO2 420mA Ch1
		AO2 420mA Ch2
		AO2 420mA Ch3
		AO2 420mA Ch4
HL5001	HL5001_1	Counter600 1 1
		Counter600 1 2
HL5001	HL5001_2	Counter600 2 1
		Counter600 2 2
HL5002	HL5002_1	AbsEncodeSSI 1
		Abs Encode 1 Cmd
HL5002	HL5002_2	AbsEncodeSSI 2
		Abs Encode 2 Cmd