目录

术语

ISCS	integrated supervision and control system
RTDB	real time database
ACII	analogue combined internal input
DCII	discrete combined internal input
DIO	Discrete internal output
AIO	Analogue internal output

综合监控实时数据库点表定义

1. 实时数据库数据结构

实时数据库采用层次化结构对数据进行组织和管理。

1.1. 数据点表

实时数据点表由节点和设备构成，见图 1 实时数据库点表结构。

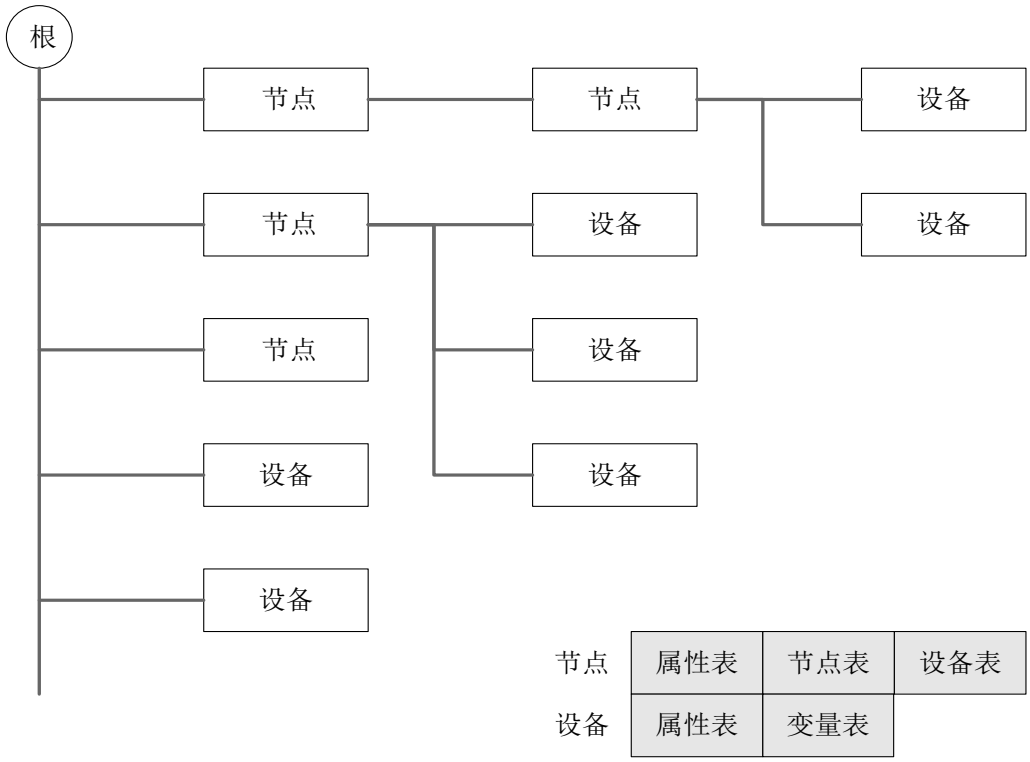

图 1 实时数据库点表结构

节点由节点属性、子节点和设备构成。其中，属性数量为 0 或多个；子节点数量为 0 或多个；设备数量为 0 或多个。

设备由属性和变量构成。其中，属性数量为 0 或多个；内部变量数量为 0 或多个。

1.2. 变量

实时数据库中的变量为内部变量，内部变量分为输入变量和输出变量。

1.2.1. 输入变量

输入变量为两级结构，第一级，变量属性。它定义变量拥有的全部属性。第二级，变量数据。它定义了变量的有效数据源。变量数据有五类它们分别是采集数据、计算数据、强制数据、估算数据和告警数据。其中，采集数据和计算数据只能有一个有效。

输入变量有模拟输入变量（描述浮点类型数据）、离散输入变量（描述整数类型数据）、原型输入变量（描述字符串类型数据）和结构化输入变量（描述自定义数据）。

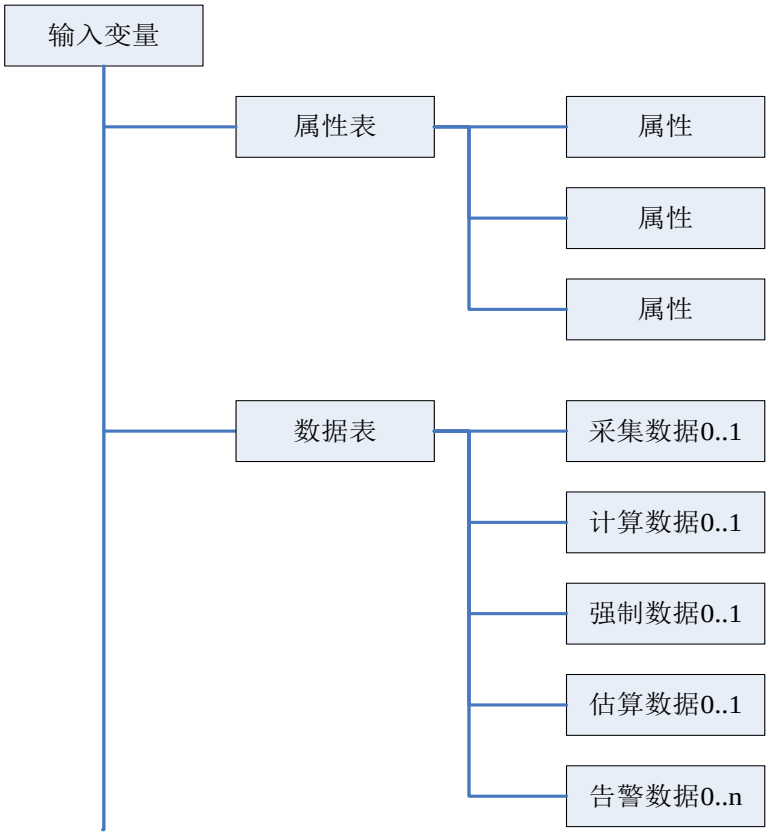

图 2 输入变量

1.2.2. 输出变量

输出变量分为两类：离散输出变量和模拟输出变量。

离散输出变量有两级。第一级，属性。它定义了命令需要的属性。第二级，数据。它定义了命令行为需要的和发送给设备的数据。

模拟输出变量，只有一级。它定义了发送命令需要的属性和数据。

2. 变量路径

2.1. 路径定义

路径指的是在 RTDB 中查找节点的路由（类似文件系统路径）。路径的格式如下：

$节点名:[...]:节点名.属性键

具体的含义为：

- $

  路径的根符号，路径的开始标记。路径必须以'$'开头。

- :

  节点分隔符。

- .

  节点与属性的分隔符

- 节点名

  节点 ID，构成符号有：

  - 字母[a-z]和[A-Z]
  - 数字[0-9]
  - 下划线‘_’和连接线‘-’

路径搜索形式有四种：当前路径搜索、绝对路径搜索、当前位置前向搜索和当前位置后向搜索。

2.1.1. 当前路径搜索

路径形式：

.	-- 当前节点（变量节点）
.key	-- 当前节点的属性
.key.childkey	-- 当前节点的属性的子属性

2.1.2. 绝对路径搜索

路径形式：

$	-- 根
$node	-- 节点
$node.key	-- 节点的属性
$node.key.childkey	-- 节点属性的子属性
$node1:...:nodeN	-- 节点 N
$node1:...:nodeN.key	-- 节点 N 的属性
$node1:...:nodeN.key.childkey	-- 节点 N 的属性的子属性

2.1.3. 当前位置前向搜索

路径形式：

:node	-- 当前节点的直接孩子节点
:node.key	-- 当前节点的直接孩子节点的属性
:node.key.childkey	-- 当前节点的直接孩子节点的属性的子属性
:node1:...:nodeN	-- 当前节点的第 N 级孩子节点
:node1:...:nodeN.key	-- 当前节点的第 N 代子孙节点的属性
:node1:...:nodeN.key.childkey	-- 当前节点的第 N 代子孙节点的属性的子属性

2.1.4. 当前位置后向搜索

路径形式：

^	-- 当前节点的直接父亲节点
^.key	-- 当前节点直接父亲节点属性
^.key.childkey	-- 当前节点直接父亲节点属性的子属性
^:...:^	-- 当前节点的直接 N 代祖先
^:...:^.key	-- 当前节点的直接 N 代祖先属性
^:...:^.key.childkey	-- 当前节点的直接 N 代祖先属性的子属性
^:node	-- 当前节点的兄弟节点
^:node.key	-- 当前节点兄弟节点的属性
^:node.key.childkey	-- 当前节点兄弟节点的属性的子属性
^:^:node	-- 当前节点的父亲的兄弟节点
^:^:node.key	-- 当前节点的父亲的兄弟节点的属性
^:^:node.key.childkey	-- 当前节点的父亲的兄弟节点的属性子属性

备注：

（1）路径搜索模式由路径的开始字符决定。“.”点号标识当前路径搜索，“$”美元符号标识绝对路径搜索，“:”冒号标识当前路径前向搜索（搜索当前节点子孙节点），“^”异或符号标识当前路径后向搜索（搜索当前节点的祖先节点）。

（2）“.”点号只能出现在所有其它标识符号的后面，出现次数为 0、1 或 2。

（3）“$”美元符号只能出现在路径开头，且仅出现一次。

（4）“^”异或符号若有多个，必须连续出现，且必须在任何具体节点之前出现。

3. 变量定义

综合监控系统定义两种类型的变量：内部变量、外部变量。

外部变量，用于数据采集、命令控制和告警。作用范围协议插件、FEP、告警服务和 HMI。

内部变量，用于数据分析。作用范围实时数据就和 HMI。

一个内部变量至少需要有一个外部变量与之对应。

3.1. 外部变量

略

3.2. 内部变量

实时数据库定义的内部变量有：EQP、ACI、DCI、SCI、AIO、DIO 和 SIO。

HOST，环境变量。表示服务器、应用系统等环境设备。

EQP，设备变量。表示被监控的具体设备。例如空调、自动扶梯等。

ACI，模拟组合输入（analogue combined input data）。表示模拟类型输入数据。

DCI，离散组合输入（discrete combined input data）。表示离散类型输入数据。

SCI，结构化组合输入（structured combined input data）。表示复合类型输入数据。

AIO，模拟内部输出（analogue interval output data）。表示内部输出数据。

DIO，离散内部输出（discrete interval output data）。表示离散内部输出数据。

SIO，结构化内部输出（structured interval output data）。表示复合类型内部输出数据。

3.3. 内部输入变量

3.3.1. EQP（设备）

名称	类型	必要	修改者	描述
name	String	否	配置	设备名
id	String	否	配置	设备编号
label	String	否	配置	设备描述
manufacturer	String	否	配置	设备制造商信息
vendor	String	否	配置	设备供应商信息
fnCategory	Int32	否	配置	设备功能类型（functional category）。 一个设备属于一种功能类型。功能类型由项目定义。
geoCategory	Int32	否	配置	设备地理位置类型（geographical category）。 一个设备属于一个地理位置类型。地理位置类型由项目定义。
isControlable	Int32	是	系统	设备是否可以控制： 0：禁止控制 1：允许控制 备注： 设备禁止状态为禁止控制设为 1，否则设置为 0。 设备在收到命令时，需要检查本数据项。
inhibit	Int32	是	系统	设备禁用状态掩码： 0x00000000：允许控制 0x00000001：控制禁止（禁止挂牌） 0x00000002：控制禁止（检修挂牌） 0x00000004：控制禁止（接地挂牌） 0x00000010：告警禁止 0x00000020：监视禁止 备注： HMI 操作员设置

3.3.2. ACI（模拟组合输入）

3.3.2.1. ACI

名称	类型	必要	修改者	描述
name	String	否	配置	变量名称
label	String	否	配置	变量描述

isStateChanged	Int32	是	系统	组合值的状态是否发生变化。 0：状态无变化 1：状态发生变化 有变化设置 1；HMI 操作员确认时设置 0
source	Int32	是	系统	标识组合的数据来自哪里，可选值： 1：来自 acquired 或 computed 2：来自 forced 3：来自 estimated 可选计算函数： ISCS_SOURCE
status	Int32	是	系统	值的组合状态，描述值的状态： 0：非计算的/非强制的/非获取的/非估计的 1：计算的/强制的/获取的/估计的 可选计算函数： ISCS_STATUS
validity	Int32	是	系统	标识值的有效性： 0：无效 1：有效 当值是强制值时，有效性设为 1
date	Int64	是	系统	组合的 ACI 日期，单位毫秒。 可选计算函数： ISCS_DATE
value	Double	是	系统	组合的 ACI 值。 可选的计算函数： ISCS_VALUE
previousStatus	Int32	否	系统	上次组合值的状态。 可选计算函数： ISCS_SYNSTATUS
previousValidity	Int32	否	系统	上次组合值的有效性。
previousDate	Int64	否	系统	上次组合的时间。 可选计算函数： ISCS_SYNDATE
previousValue	Double	否	系统	上次组合值。 可选计算函数： ISCS_SYNVALUE 计算得来
inhibit	Int32	是	系统	数据点禁用掩码： 0x00000000：无禁止 0x00000001：刷新禁止 0x00000002：告警禁止 0x00000004：预留 0x00000010：预留 备注： HMI 操作员设置
computed	Object	否	系统	计算数据。

				用于定义具体的计算数据。当 acquired 发生变化时进行数据计算。 只有在 acquired 的数据是转换后的数据才配置计算数据。
value	Double	是	系统	计算的结果值。 可用的函数：
date	Int64	是	系统	结果值的日期，单位毫秒。 可用的函数：
status	Int32	是	系统	结果值的状态： 0：非计算的值 1：计算的值 可用的函数：
estimated	Object			估计数据。 acquired/computed 无效时使用。
value	Double	是	配置	估计值
date	Int64	是	系统	估计值的日期
status	Int32	是	配置	估计值状态： 0：非估算的值 1：估算的值
forced	Object			强制数据。 HMI 操作员和系统修改。
value	Double	是	系统	强制的值。 HMI 操作员设置。
date	Date	是	系统	日期，单位毫秒。 RTDB 系统修改，强制时的时间戳。
status	Int32	是	系统	强制状态： 0：非强制的值 1：强制的值 由 HMI 操作员执行强制/取消强制。RTDB 修改相关数据。
acquired	Object			获取的数据。 与 input 相关联，与 computed 互斥。
value	Double	是	函数	获取的值。 可用的函数： ISCS_ACQ_VALUE
date	Int64	是	函数	日期，单位毫秒。 可用的函数： ISCS_ACQ_DATE
status	Int32	是	函数	值的状态： 0：非获取的值 1：获取的值 可用的函数： ISCS_ACQ_STATUS
input	Object	是		外部输入的获取到的数据。

					anchor 和 name 用于关联外部和内部变量
	anchor	String	否	配置	外部变量的环境（例如插件 ID）
	name	String	否	配置	外部变量名
	value	Double	是	系统	采集到的值
	date	Int64	是	系统	采集日期（毫秒）
	status	Int32	是	系统	采集值状态： 0：非采集的值 1：采集的值

3.3.2.2. AAL（模拟输入告警）

名称		类型	必要	修改者	描述
dssEventType		String	否	配置	关联的决策支持系统的事件类型。该类型必须在决策支持系统中存在。
confirmed		Int32	是	系统	告警确认状态。
counter		Int32	是	系统	告警计数器。保存连续发生的告警数量。告警恢复后复位为 0。
valueLimits		Double[]	是	配置	值属性的边界值（最多 5）。这些边界值与 valueTable 属性必须对应。边界值与区间是固定的： V0：容差值 V1：边界值 1，定义区间 1，(-∞，V1] V2：边界值 2，定义区间 2，(V1, V2] V3：边界值 3，定义区间 3，(V2, V3] V4：边界值 4，定义区间 4，(V3, V4] V5：边界值 5，定义区间 5，(V4, V5]
valueTable		Object[]	是	配置	定义边界值的处理区间，必须与 valueLimits 对应。
	index	Int32	是	配置	区间索引从 1 到 5： 区间 1，(-∞，V1] 区间 2，(V1, V2] 区间 3，(V2, V3] 区间 4，(V3, V4] 区间 5，(V4, V5] 区间 6，(V5, +∞) V1，V2，V3，V4，V5 在 valueLimits 中定义
	label	String	是	配置	当前区间描述
	state	String	是	配置	当前区间的告警状态，可能值如下： ：Normal ：Warning ：Alarm ：Transitory
	severity	Int32	是	配置	告警严重等级（1-8），等级程度有项目定义

3.3.2.3. 两个边界值

假设两个边界值由低到高分别是 limit1 和 limit2。两个边界值把告警划分为三个区间，区间 1 对应值范围(-∞, limit1]，本区间为告警；区间 2 对应值范围(limit1, limit2]，本区间为正常；区间 3 对应值范围(limit2, ∞]，本区间为告警。

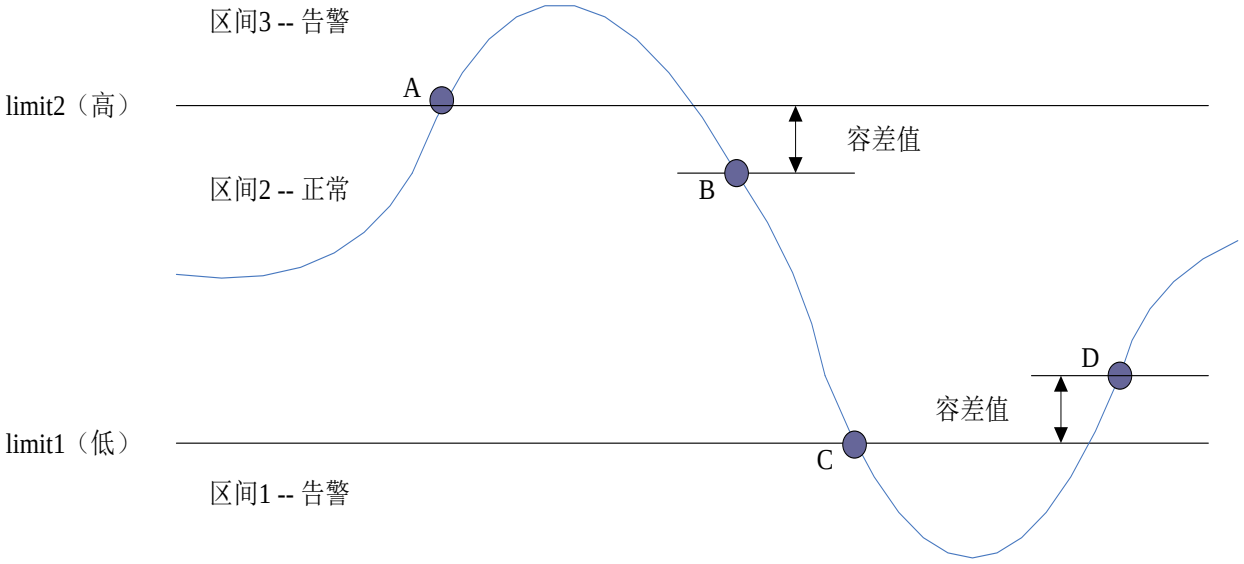

图 4 两边界值的 AALD

当前值低于或等于 limit1 时（C 点），区间 1 是当前区间。当前值高于 limit2 时（A 点），区间 3 是当前区间。当前值高于 limit1 但低于或等于 limit2 时，并且当前值与 limt1 的差和 limit2 与当前值的差都大于等于容差值的时候（B 点和 D 点），当前区间是区间 2。

3.3.2.4. 三个边界值

假设三个边界值由低到高分别是 limit1，limit2，limit3。三个边界值把告警划分为四个区间，区间 1 对应值范围(-∞, limit1]，本区间为告警；区间 2 对应值范围(limit1, limit2]，本区间为正常；区间 3 对应值范围(limit2, limit3]，本区间为告警；区间 4 对应值范围(limit3, ∞)，本区间为告警。

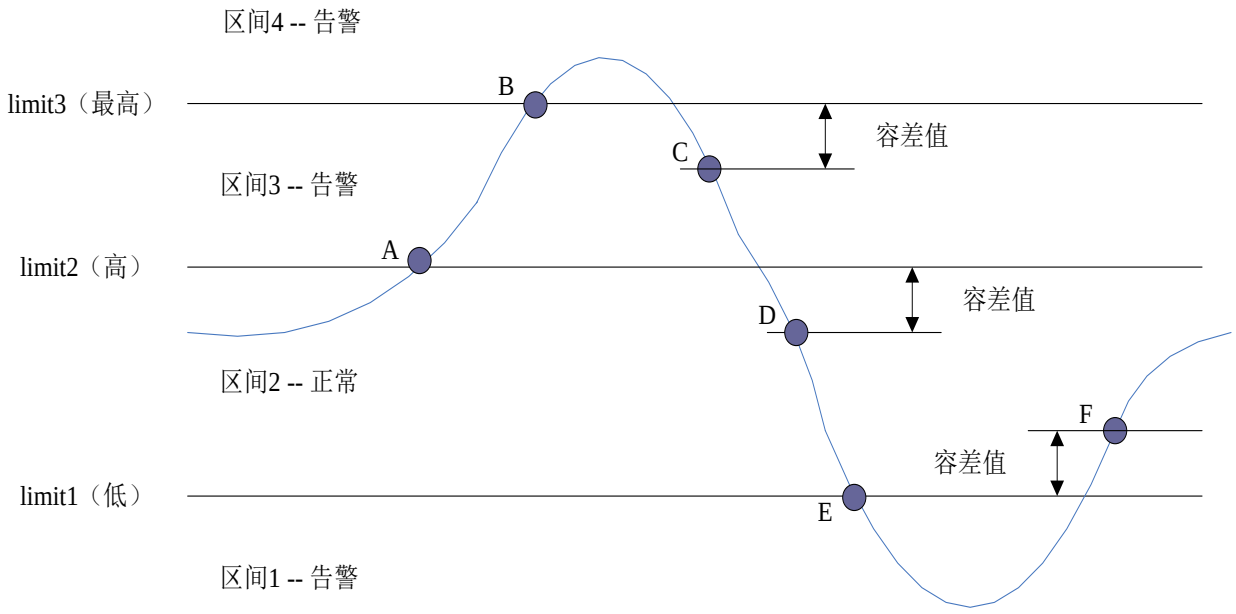

图 5 三边界值的 AALD

当前值低于或等于 limit1 时（E 点），区间 1 是当前区间。当前值高于 limit1 但低于或等于 limit2 时，如果当前值与 limit1 的差和 limit2 与当前值的差都大于或等于容差时（F 点和 D 点），区间 2 为当前区间。当前值高于 limit2（A 点）但低于或等于 limit3 时，并且 limit3 与当前值的差大于或等于容差值（C 点）时，区间 3 为当前区间。当前值高于 limit3 时（B 点），区间 4 为当前区间。

3.3.3. DCI（离散组合输入）

3.3.3.1. DCI

名称	类型	必要	修改者	描述
name	String	否	配置	变量名称
label	String	否	配置	变量描述
isStateChanged	Int32	是	系统	状态是否发生变化： 0：状态无变化 1：状态发生变化 当设备的值有变化设置 1；HMI 操作员确认时设置 0
source	Int32	是	系统	标识组合的数据来自哪里： 1：来自 acquired 或 computed 2：来自 forced 3：来自 estimated

				可选计算函数： ISCS_SOURCE
status	Int32	是	系统	值的组合状态，描述值的状态： 0：非计算的/非强制的/非获取的/非估计的 1：计算的/强制的/获取的/估计的 可选计算函数： ISCS_STATUS
validity	Int32	是	系统	标识值的有效性： 0：无效 1：有效 当值是强制值时，有效性设为 1
date	Int64	是	系统	组合的 DCI 日期，单位毫秒。 可选计算函数： ISCS_DATE
value	Int32	是	系统	组合的 DCI 值。 可选的计算函数： ISCS_VALUE
previousStatus	Int32	否	系统	上次组合值的状态。 可选计算函数： ISCS_SYNSTATUS
previousValidity	Int32	否	系统	上次组合值的有效性。
previousDate	Int64	否	系统	上次组合的时间。 可选计算函数： ISCS_SYNDATE
previousValue	Int32	否	系统	上次组合值。 可选计算函数： ISCS_SYNVALUE 计算得来
inhibit	Int32	是	系统	数据点禁用掩码： 0x00000000：无禁止 0x00000001：刷新禁止 0x00000002：告警禁止 0x00000004：预留 0x00000010：预留 备注： HMI 操作员设置
computed	Object			计算数据。 用于定义具体的计算数据。当 acquired 发生变化时进行数据计算。 只有在 acquired 的数据是转换后的数据才配置计算数据。
value	Int32	是	函数	计算的结果值。 可用的函数：
date	Int64	是	函数	结果值的日期，单位毫秒。 可用的函数：

	status	Int32	是	函数	结果值的状态： 0：非计算的值 1：计算的值 可用的函数：
estimated		Object			估计数据。 acquired/computed 无效时使用。
	value	Int32	是	函数	估计值
	date	Int64	是	函数	估计值的日期
	status	Int32	是	函数	估计值状态： 0：非估算的值 1：估算的值
forced		Object			强制数据。 HMI 操作员和系统修改。
	value	Int32	是	系统	强制的值。 HMI 操作员设置。
	date	Int64	是	系统	日期，单位毫秒。 RTDB 系统修改，强制时的时间戳。
	status	Int32	是	系统	强制状态： 0：非强制的值 1：强制的值 由 HMI 操作员执行强制/取消强制。RTDB 修改相关数据。
acquired		Object			获取的数据。 与 input 相关联，与 computed 互斥。
	value	Int32	是	函数	获取的值。 可用的函数： ISCS_ACQ_VALUE ISCS_ACQ_COMBINE2_VALUE ISCS_ACQ_COMBINE3_VALUE ISCS_ACQ_SELECT3_VALUE
	date	Int64	是	函数	日期，单位毫秒。 可用的函数： ISCS_ACQ_DATE
	status	Int32	是	函数	值的状态： 0：非获取的值 1：获取的值 可用的函数： ISCS_ACQ_STATUS
input		Object			外部输入的获取到的数据，其中 anchor 和 name 把外部变量关联到内部变量
	anchor	String	是	配置	外部变量的环境（例如插件 ID）
	name	String	是	配置	外部变量名
	value	Int32	是	系统	采集到的值
	date	Int64	是	系统	采集日期（毫秒）

	status	Int32	是	系统	采集值状态： 0：非采集的值 1：采集的值

3.3.3.2. DAL

名称		类型	必要	修改者	描述
dssEventType		String	否	配置	关联的决策支持系统的事件类型。该类型必须在决策支持系统中存在。
confirmed		Int32	是	系统	告警确认状态。
counter		Int32	是	系统	告警计数器。保存连续发生的告警数量。告警恢复后复位为 0。
valueLimits		Int32 []	是	配置	值属性的边界值（最多 5）。这些边界值与 valueTable 属性必须对应。边界值与区间是固定的： V0：未定义 V1：边界值 1，定义区间 1 V2：边界值 2，定义区间 2 V3：边界值 3，定义区间 3 V4：边界值 4，定义区间 4 V5：边界值 5，定义区间 5
valueTable		Object[]	是	配置	定义边界值的处理区间，必须与 valueLimits 对应。
	index	Int32	是	配置	区间索引从 1 到 5： 区间 1 区间 2 区间 3 区间 4 区间 5 V1，V2，V3，V4，V5 在 valueLimits 中定义
	label	String	是	配置	当前区间描述
	state	String	是	配置	当前区间的告警状态，可能值如下： ：Normal ：Warning ：Alarm ：Transitory
	severity	Int32	是	配置	告警严重等级（1-8），等级程度有项目定义

3.3.4. SCI

名称	类型	必要	修改者	描述
name	String	否	配置	变量名称

label	String	否	配置	变量描述
isStateChanged	Int32	是	系统	状态是否发生变化： 0：状态无变化 1：状态发生变化 当设备的值有变化设置 1；HMI 操作员确认时设置 0
status	Int32	是	系统	值的组合状态，描述值的状态： 0：非获取的 1：获取的
validity	Int32	是	系统	标识值的有效性： 0：无效 1：有效
date	Int64	是	系统	组合的 SCI 日期，单位毫秒。 可选计算函数： ISCS_DATE
value	String	是	系统	组合的 DCI 值。 可选的计算函数： ISCS_VALUE
inhibit	Int32	是	系统	数据点禁用掩码： 0x00000000：无禁止 0x00000001：刷新禁止 0x00000002：告警禁止 0x00000004：预留 0x00000010：预留 备注： HMI 操作员设置
computed	Object			计算数据。 用于定义具体的计算数据。当 acquired 发生变化时进行数据计算。 只有在 acquired 的数据是转换后的数据才配置计算数据。
value	Int32*string/bitfield	是	函数	计算的结果值。 可用的函数：
date	Int64	是	函数	结果值的日期，单位毫秒。 可用的函数：
status	Int32	是	函数	结果值的状态： 0：非计算的值 1：计算的值 可用的函数：
forced	Object			强制数据。 HMI 操作员和系统修改。
value	String	是	系统	强制的值。 HMI 操作员设置。

	date	Int64	是	系统	日期，单位毫秒。 RTDB 系统修改，强制时的时间戳。
	status	Int32	是	系统	强制状态： 0：非强制的值 1：强制的值 由 HMI 操作员执行强制/取消强制。RTDB 修改相关数据。
acquired		Object			获取的数据。 与 input 相关联，与 computed 互斥。
	value	Int32	是	函数	获取的值。
	date	Int64	是	函数	日期，单位毫秒。 可用的函数： ISCS_ACQ_DATE
	status	Int32	是	函数	值的状态： 0：非获取的值 1：获取的值 可用的函数： ISCS_ACQ_STATUS
input		Object			外部输入的获取到的数据，其中 anchor 和 name 把外部变量关联到内部变量
	anchor	String	是	配置	外部变量的环境（例如插件 ID）
	name	String	是	配置	外部变量名
	value	Int32	是	系统	采集到的值
	date	Int64	是	系统	采集日期（毫秒）
	status	Int32	是	系统	采集值状态： 0：非采集的值 1：采集的值

3.4. 内部输出变量

3.4.1. AIO

名称	类型	必要	修改者	描述
name	String	是	配置	名称
label	String	否	配置	变量描述
isLocal	Int32	否	配置	本地命令标识： 0：远程命令 1：本地命令 本地命令仅在 RTDB 内部执行，不会下发给下级系统。远程命令需要下发给下级系统执行，RTDB 不执行。
execStatus	Int32	是	命令	命令的当前执行状态：

				0：完成（返回条件验证完毕） 1：执行中（执行条件验证中） 2：执行中（执行条件验证完毕） 3：执行中（返回条件验证中） 4：执行中（返回条件验证完毕） 5：错误（执行条件验证失败） 6：错误（返回条件验证失败）
initCond<n>	Int32	否	函数	命令初始化条件，n 表示有 n 个初始化条件。它们分别是 initCond1，……，initCondn。 值： 0：失败 1：成功 可用的函数： ISCS_IF
initCondGL	Int32	否	函数	综合的初始化条件，initCond<n>逻辑与。 值： 0：失败 1：成功 可用的函数 ISCS_IF
retCond<n>	Int32	否	函数	返回条件，n 表示有 n 个返回条件。它们分别是 retCond1，……，retCondn。 值： 0：失败 1：成功 可用的函数： ISCS_IF
retCondGL	Int32	否	函数	综合的返回条件，retCond<n>逻辑与。 值： 0：失败 1：成功 可用的函数 ISCS_IF
source	String	否	命令	命令源，项目自己定义。
status	Int32	是	命令	当前状态掩码： 0x0001：有效 0x0002：无效（设备被禁止） 0x0004：无效（变量被禁止）
value	Double	是	命令	命令的当前值（当前正在执行的命令）
date	Int64	是	命令	命令的下发时间（单位：毫秒）
valueLimits	Double[2]	否	配置	命令值的边界： [0]：下限值 [1]：上限值 当命令的值不在边界范围内，命令不发送。

output		Object			命令内容
	anchor	String	是	配置	外部变量环境（例如 path）。
	name	String	是	配置	命令名称。详情4见系统命令。
	value	Double	是	命令	命令值

3.4.2. DIO

名称	类型	必要	修改者	描述
name	String	是	配置	名称
label	String	否	配置	变量描述
isLocal	Int32	否	配置	本地命令标识： 0：远程命令 1：本地命令 本地命令仅在 RTDB 内部执行，不会下发给下级系统。远程命令需要下发给下级系统执行，RTDB 不执行。
execStatus	Int32	是	命令	命令的当前执行状态： 0：完成（返回条件验证完毕） 1：执行中（执行条件验证中） 2：执行中（执行条件验证完毕） 3：执行中（返回条件验证中） 4：执行中（返回条件验证完毕） 5：错误（执行条件验证失败） 6：错误（返回条件验证失败）
initCond<n>	Int32	否	函数	命令初始化条件，n 表示有 n 个初始化条件。它们分别是 initCond1，……，initCondn。 值： 0：失败 1：成功 可用的函数： ISCS_IF
initCondGL	Int32	否	函数	综合的初始化条件，initCond<n>逻辑与。 值： 0：失败 1：成功 可用的函数 ISCS_IF
retCond<n>	Int32	否	函数	返回条件，n 表示有 n 个返回条件。它们分别是 retCond1，……，retCondn。 值： 0：失败 1：成功

				可用的函数： ISCS_IF
retCondGL	Int32	否	函数	综合的返回条件，retCond<n>逻辑与。 值： 0：失败 1：成功 可用的函数 ISCS_IF
source	String	否	命令	命令源，项目自己定义。
status	Int32	是	命令	当前状态掩码： 0x0001：有效 0x0002：无效（设备被禁止） 0x0004：无效（变量被禁止）
value	Int32	是	命令	命令的当前值（当前正在执行的命令）
date	Int64	是	命令	命令的下发时间（单位：毫秒）
output	Object			命令内容
anchor	String	是	配置	外部变量环境（例如 path）
name	String	是	配置	命令名称。详情4见系统命令。
value	Int32	是	命令	命令值

3.4.3. SIO

名称	类型	必要	修改者	描述
name	String	是	配置	名称
label	String	否	配置	变量描述
isLocal	Int32	否	配置	本地命令标识： 0：远程命令 1：本地命令 本地命令仅在 RTDB 内部执行，不会下发给下级系统。远程命令需要下发给下级系统执行，RTDB 不执行。
execStatus	Int32	是	命令	命令的当前执行状态： 0：完成（返回条件验证完毕） 1：执行中（执行条件验证中） 2：执行中（执行条件验证完毕） 3：执行中（返回条件验证中） 4：执行中（返回条件验证完毕） 5：错误（执行条件验证失败） 6：错误（返回条件验证失败）
initCond<n>	Int32	否	函数	命令初始化条件，n 表示有 n 个初始化条件。它们分别是 initCond1，……，initCondn。 值： 0：失败

				1：成功 可用的函数： ISCS_IF
initCondGL	Int32	否	函数	综合的初始化条件，initCond<n>逻辑与。 值： 0：失败 1：成功 可用的函数 ISCS_IF
retCond<n>	Int32	否	函数	返回条件，n 表示有 n 个返回条件。它们分别是 retCond1，……，retCondn。 值： 0：失败 1：成功 可用的函数： ISCS_IF
retCondGL	Int32	否	函数	综合的返回条件，retCond<n>逻辑与。 值： 0：失败 1：成功 可用的函数 ISCS_IF
status	Int32	是	命令	当前状态掩码： 0x0001：有效 0x0002：无效（设备被禁止） 0x0004：无效（变量被禁止）
value	String	是	命令	命令的当前值（当前正在执行的命令）
date	Int64	是	命令	命令的下发时间（单位：毫秒）
output	Object			命令内容
anchor	String	是	配置	外部变量环境（例如 path）
name	String	是	配置	命令名称。详情4见系统命令。
value	String	是	命令	命令值

4. 系统命令

系统命令分就地命令和远程命令。

就地命令指的是 RTDB 本地需要处理的命令。

远程命令指的是 RTDB 本地不处理，需要下级系统处理的命令。

就地命令和远程命令的设置，把 RTDB 配置的输出点的”isLocal”属性设为 1 时，该输出点就是就地命令，设为 0 时，该输出的就是远程命令。

4.1. 就地命令

RTDB 支持的就地命令，有强制值命令、取消强制命令、挂牌命令、摘牌命令。

表 4-1-1 就地命令

命令	说明
LC_SET_FORCE	强制值，需要输入强制的值作为参数
LC_CLS_FORCE	取消强制，无参数
LC_SET_JZGP	挂牌（禁止挂牌），无参数
LC_CLS_JZGP	摘牌（禁止挂牌），无参数
LC_SET_JZGPJX	挂牌（禁止挂牌检修），无参数
LC_CLS_JZGPJX	摘牌（禁止挂牌检修），无参数
LC_SET_JZGPJD	挂牌（禁止挂牌接地），无参数
LC_CLS_JZGPJD	摘牌（禁止挂牌接地），无参数
LC_CLS_STATECHANGE	复位状态变化（值 0），无参数

4.2. 远程命令

5. 计算引擎（CE）

计算引擎（CE：Calculation Engine）为综合监控提供事件处理计算。它由表达式和函数构成。

5.1. 表达式

实时数据库表达式支持两种类型的表达式：函数表达式、逻辑运算表达式和算术运算表达式。

注意，不支持表达式嵌套。函数表达式不能嵌套函数表达式或算术运算表达式；算数运算表达式不能嵌套函数表达式。

函数表达式命名规则，函数表达式必须以“ISCS_”作为表达式前缀，后面跟表达式功能名称。函数名称最大长度不能超过 128 个字符。

实时数据库支持函数表达式和计算表达式，函数表达式如下：

ISCS_XXX()

ISCS_XXX([arg1],,[argN])

计算表达式如下（算数运算表达式）：

[arg1]+...+[argN]

[arg1]-...-[argN]

[arg1]*2 或 2*[arg1]

[arg]/2 huo 2/[arg]

[arg]%3

5.1.1. 运算符

表达式 IF 表达式，逻辑运算符和算术运算符。具体如下：

- IF 语句

- ✓ IF
- ✓ IF　ELSE

➢ 逻辑运算符

- ✓ and
- ✓ or
- ✓ not

➢ 比较运算符

- ✓ $gt

  大于，相当于“>”。

- ✓ $gte

  大于等于，相当于“>=”。

- ✓ $e

  等于，相当于“==”。

- ✓ $ne

  不等于，相当于“!=”。

- ✓ $lt

  小于，相当于“<”。

- ✓ $lte

  小于等于，相当于“<=”。

➢ 算术运算符

加‘+’、减‘-’、乘‘*’，除‘/’。

5.2. 属性参数

表达式和函数可以访问任何节点的属性。属性的访问方式如下：

- [.attributeName[position]]

  当前节点的属性。

- [^.attributeName[position]]

  当前节点的父节点的属性。

- [^:nodeName.attributeName[position]]

  当前节点的兄弟节点的属性。

- [:nodeName.attriubteName[position]]

  当前节点的孩子节点的属性。

  备注：“position”为属性的第几个值（属性可以拥有多个值）。

6. 表达式

6.1. 函数表达式

6.1.1. ISCS_ACQ_STATUS

形式：

ISCS_ACQ_STATUS([status1],...,[statusN])

参数：

[status1] -- RTDB 的 status 值

.....

[statusN] -- RTDB 的 status 值

返回值：

RTDB 的 status 值。

功能描述：

计算 ACI、DCI 和 SCI 类型数据点的 acquired 属性的 status 属性值。

[statatus1]，……，[statusN]中，任何一个无效，则返回无效。

用法实例：

ISCS_ACQ_STATUS([.input.status[0]])

ISCS_ACQ_STATUS([.input.status[0]],.input.status[1]])

ISCS_ACQ_STATUS([.input.status[0]],...,.input.status[n]])

6.1.2. ISCS_ACQ_DATE

形式：

ISCS_ACQ_DATE([date])

参数：

[date] -- RTDB 的 date 值

返回值：

RTDB 的 date 值

功能描述：

计算 ACI、DCI 和 SCI 类型数据点的 acquired 属性的 date 属性值。

用法实例：

ISCS_ACQ_DATE([.input.date[0]])

6.1.3. ISCS_ACQ_VALUE

形式：

ISCS_ACQ_VALUE([value])

参数：

[value] -- RTDB 的 value 值

返回值：

RTDB 的 value 值

功能描述：

计算 ACI、DCI 和 SCI 类型数据点的 acquired 属性的 value 属性值。

用法实例：

ISCS_ACQ_VALUE([.input.value[0]])

6.1.4. ISCS_ACQ_COMBINE2_VALUE

形式：

ISCS_ACQ_COMBINE2_VALUE([value1],[value2])

参数：

[value1] -- RTDB 的 value 值
[value2] -- RTDB 的 value 值

返回值：

RTDB 的 value 值

功能描述：

计算 DCI 类型数据点的 acquired 属性的 value 属性值。
用于两个值的包含组合计算。

[value1]	[value2]	[result]
0	0	0
0	1	1
1	0	2
1	1	3

用法实例：

ISCS_ACQ_COMBINE2_VALUE([.input.value[0]],[.input.value[1]])

6.1.5. ISCS_ACQ_COMBINE3_VALUE

形式：

ISCS_ACQ_COMBINE3_VALUE([value1],[value2],[value3])

参数：

[value1] -- RTDB 的 value 值
[value2] -- RTDB 的 value 值
[value3] -- RTDB 的 value 值

返回值：

RTDB 的 value 值

功能描述：

计算 DCI 类型数据点的 acquired 属性的 value 属性值。
用于两个值的包含组合计算。

[value1]	[value2]	[value3]	[result]
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

用法实例：

ISCS_ACQ_COMBINE3_VALUE([.input.value[0]],[.input.value[1]],[.input.value[2]])

6.1.6. ISCS_ACQ_SELECT3_VALUE

形式：

ISCS_ACQ_SELECT3_VALUE([value1],[value2],[value3])

参数：

[value1] -- RTDB 的 value 值
[value2] -- RTDB 的 value 值
[value3] -- RTDB 的 value 值

返回值：

RTDB 的 value 值

功能描述：

计算 DCI 类型数据点的 acquired 属性的 value 属性值。
用于三个值的互斥组合计算。

[value1]	[value2]	[value3]	[result]
0	0	0	0
1	0	0	1
0	1	0	2
0	0	1	3

其他情况 7

用法实例：
ISCS_ACQ_VALUE([.input.value[0]],[.input.value[1]],[.input.value[2]])

6.1.7. ISCS_SOURCE

形式：
ISCS_SOURCE([inputStatus],[forcedStatus],[estimatedStatus])

参数：
[inputStatus] -- RTDB 的 status 值，详情参照 acquired/comupted 的 status。
[forcedStatus] -- RTDB 的 status 值，详情参照 forced 的 status。
[estimatedStatus] -- RTDB 的 status 值，详情参照 estimated 的 status。

返回值：
RTDB 的 source 值。

功能描述：
计算 ACI、DCI 和 SCI 类型数据点的 source 属性值。

[inputStatus]	[forcedStatus]	[estimatedStatus]	[result]
x	未指定或 0	未指定或 0	1
x	1	x	2
0	0	1	3

用法实例：
ISCS_SOURCE([acquired.status[0]])
ISCS_SOURCE([acquired.status[0]],[forced.status[0]])
ISCS_SOURCE([acquired.status[0]],[forced.status[0]],[estimated.status[0]])
ISCS_SOURCE([computed.status[0]])
ISCS_SOURCE([computed.status[0]],[forced.status[0]])
ISCS_SOURCE([computed.status[0]],[forced.status[0]],[estimated.status[0]])

6.1.8. ISCS_STATUS

形式：
ISCS_STATUS([source],[inputStatus],[forcedStatus],[estimatedStatus])

参数：
[source] -- RTDB 的 source 值。
[inputStatus] -- RTDB 的 status 值，详情参照 acquired/comupted 的 status。
[forcedStatus] -- RTDB 的 status 值，详情参照 forced 的 status。
[estimatedStatus] -- RTDB 的 status 值，详情参照 estimated 的 status。

返回值：

RTDB 的 status 值。

功能描述：

计算 ACI、DCI 和 SCI 类型数据点的 status 属性值。

[source]	[result]
1	inputStatus
2	forcedStatus
3	estimatedStatus

用法实例：

ISCS_STATUS([.source[0]],[acquired.status[0]],[forced.status[0]])
ISCS_STATUS([.source[0]],[computed.status[0]],[forced.status[0]])
ISCS_STATUS([.source[0]],[acquired.status[0]],[forced.status[0]],[estimated.status[0]])
ISCS_STATUS([.source[0]],[computed.status[0]],[forced.status[0]],[estimated.status[0]])

6.1.9. ISCS_DATE

形式：

ISCS_DATE([source],[inputDate],[forcedDate],[estimatedDate])

参数：

[source] -- RTDB 的 source 值。
[inputDate] -- RTDB 的 date 值，详情参照 acquired/comupted 的 date。
[forcedDate] -- RTDB 的 date 值，详情参照 forced 的 date。
[estimatedDate] -- RTDB 的 date 值，详情参照 estimated 的 date。

返回值：

RTDB 的 date 值。

功能描述：

计算 ACI、DCI 和 SCI 类型数据点的 date 属性值。

[source]	[result]
1	inputDate
2	forcedDate
3	estimatedDate

用法实例：

ISCS_DATE([.source[0]],[acquired.date[0]],[forced.date[0]])
ISCS_DATE([.source[0]],[computed.date[0]],[forced.date[0]])
ISCS_DATE([.source[0]],[acquired.date[0]],[forced.date[0]],[estimated.date[0]])
ISCS_DATE([.source[0]],[computed.date[0]],[forced.date[0]],[estimated.date[0]])

6.1.10. ISCS_VALUE

形式：

ISCS_VALUE([source],[status],[inputValue],[forcedValue],[estimatedValue])

参数：

[source] -- RTDB 的 source 值。
[status] -- RTDB 的 status 值。
[inputValue] -- RTDB 的 value 值，详情参照 acquired/comupted 的 value。
[forcedValue] -- RTDB 的 value 值，详情参照 forced 的 value。
[estimatedValue] -- RTDB 的 value 值（可選項），详情参照 estimated 的 value。
SCI 点没有本数据项。

返回值：

RTDB 的 value 值。

功能描述：

计算 ACI、DCI 和 SCI 的 value 属性值。

[source]	[result]
1	inputValue
2	forcedValue
3	estimatedValue（SCI 类型点没有该数据）

当 status 为有效时（1），返回选择的结果值（result）；否则返回值为当前值。

用法实例：

```
ISCS_VALUE([.source[0]],[.status[0]],[acquired.value[0]],[forced.value[0]])
ISCS_VALUE([.source[0]],[.status[0]],[computed.value[0]],[forced.value[0]])
ISCS_VALUE([.source[0]],[.status[0]],[acquired.value[0]],[forced.value[0]],[estimated.value[0]])
ISCS_DATE([.source[0]],[.status[0]],[computed.value[0]],[forced.value[0]],[estimated.value[0]])
```

6.1.11. ISCS_SYNSTATUS

形式：

```
ISCS_SYNSTATUS([status])
```

参数：

[status] -- RTDB 的 status 值

返回值：

RTDB 的 status 值

功能描述：

同步 ACI/DCI/SCI 类型的（上次的）组合的 status。

用法实例：

```
ISCS_SYNSTATUS([.status])
```

6.1.12. ISCS_SYNDATE

形式：

ISCS_SYNDATE([date])

参数：

[date] -- RTDB 的 date 值

返回值：

RTDB 的 date 值

功能描述：

同步 ACI/DCI/SCI 类型的（上次的）组合值的 date。

用法实例：

ISCS_SYNDATE([.date])

6.1.13. ISCS_SYNVALUE

形式：

ISCS_SYNVALUE([value])

参数：

[value] -- RTDB 的 value 值

返回值：

RTDB 的 value 值

功能描述：

同步 ACI/DCI/SCI 类型的（上次的）组合值的 value。

用法实例：

ISCS_SYNVALUE([.value])

6.2. 函数表达式优先级

综合监控实时数据库系统的函数表达式有执行优先级（priority）。表达式的执行顺序按照优先级由高到低顺序执行。Priority 的值越小表示执行优先级越高。

函数表达式执行组，综合监控实时数据库按照监控点对表达式进行分组，即一个监控点是一个独立分组。在一个分组内，表达式的执行优先权必须是从零开始的连续的正整数（优先级相同的则需要强行设置不同的优先级，只要符合优先级从零一次递增即可）。

表达式优先级范围仅限于分组内的各个表达式，不同分组之间不存在优先级。

表 5-5-1 函数表达式优先级

优先权	表达式
0	ISCS_ACQ_STATUS, ISCS_ACQ_DATE, ISCS_ACQ_VALUE, ISCS_ACQ_COMBINE2_VALUE, ISCS_ACQ_COMBINE3_VALUE, ISCS_ACQ_SELECT3_VALUE
1	ISCS_SYNSTATUS, ISCS_SYNDATE, ISCS_SYNVALUE
2	ISCS_SOURCE
3	ISCS_STATUS
4	ISCS_DATE
5	ISCS_VALUE

实例 1，表达式分组内包含的函数表达式有 ISCS_ACQ_STATUS、ISCS_SYNSTATUS 和 ISCS_STATUS。它们的表达式优先权定义如下：

ISCS_ACQ_STATUS:0
ISCS_SYNSATUS:1
ISCS_STATUS:2

实例 2，表达式分组内包含的函数表达式有 ISCS_SYNSTATUS、ISCS_SYNDATE、ISCS_STATUS 和 ISCS_DATE。分组存在相同级别的函数表达式，它们的表达式优先权定义如下：

ISCS_SYNSTATUS:0
ISCS_SYNDATE:1
ISCS_DATE:2
或
ISCS_SYNDATE:0
ISCS_SYNSTATUS:1
ISCS_DATE:2

6.3. 逻辑运算表达式

6.4. 算数运算表达式

7. 变量处理（待进一步更新）

RTDB 变量处理分为输入变量的处理和输出变量处理。输入变量处理指对从设备采集到的设备状态数据进行计算分析，并生成对应的计算结果（告警信息等）。输出变量处理指对发送给监控系统或设备的命令进行环境检查、命令执行控制和结果回收等。

7.1. 输入变量处理

RTDB 处理的输入变量有模拟输入变量、离散输入变量、原型输入变量和结构化输入变量。

输入变量的处理流程，首先，采集系统从设备采集设备的运行状态等数据，生成监控外部变量（与内部变量对应）数据。然后，采集系统把外部变量数据上传给 RTDB 的内部变量。最后，RTDB 对内部变量进行分析处理，并生成处理结果（告警等数据），然后把结果上传给 Cache 系统（图 6）。

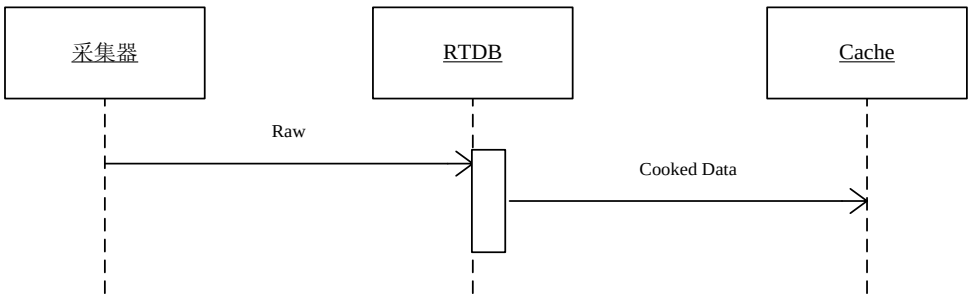

图 6 输入变量处理流程

7.1.1. 输入变量更新

RTDB 的内部输入变量的更新通过内部获取变量（AAC/DAC/RAC/SAC）实现。内部获取变量提供一个外部变量表来存放外部变量数据（veTable：External Variable Table）。采集器采集的数据上传到 RTDB 之后，就存放在 veTable 中（图 7）。

veTable				
vepole	vename	value	date	status
	iscs/bas/elevator	1	2001/01/13 ...	1

图 7 veTable 结构

一个内部获取变量可以关联多个外部变量。外部变量的值通过外部变量名进行关联，一个外部变量名称关联一组值<value,date,status>，每一组< value,date,status >输入变量都对应一个值和表达式。表达式是一个 CE 函数，用来组合 veTable 中存储的数据，计算出内部获取变量真正的值。在本系统中，内部输入获取变量只关联一个外部变量。

7.1.2. 输入变量处理

内部输入变量（ACI、DCI、RCI 和 SCI）的数据处理过程就是执行 CE 函数序列的过程。当内部获取变量的 veTable 被更新后，与当前内部输入变量关联的一系列 CE 函数就会被触发去执行数据的分析处理（CE 函数按照固定的顺序执行）。

所有内部输入变量的处理流程是一样的，不同之处是不同的变量配置的 CE 函数和 CE 函数的执行顺序。这里以模拟内部输入变量为例进行说明（图 8）。

RTDB 收到上传的模拟外部变量 AEV（AEV: Analogue External Variable）后，用 AEV 更新内部变量 ACI 的 AAC 的 veTable，并触发 ACI 的 CE 函数处理流程。首先，执行 AAC 的 CE 函数处理流程（图 8 函数处理流程 1），来更新 AAC 的 acqStatus、acqValue 和 acqDate 属性。然后，ACI 的 CE 函数处理流程（图 8 函数处理流程 2），来更新 ACI 的 previousStatus、previousValue 等属性值。最后，执行 AAL 的 CE 函数处理流程（图 8 函数处理流程 3），进行告警的运算和通知处理。

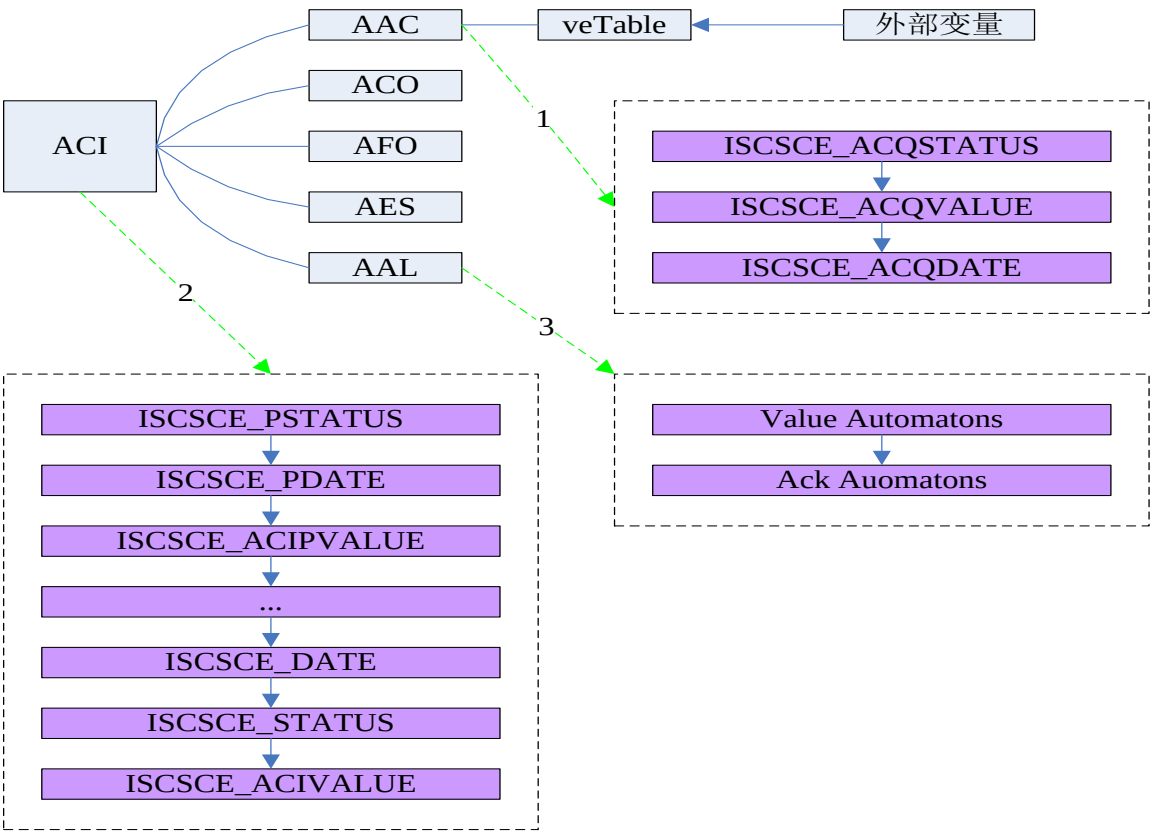

图 8 ACI 数据更新处理过程

如果一个内部输入变量没有告警数据变量（AL），在执行 CE 函数处理流程时，就没有函数处理流程 3 这一过程。

7.2. 输出变量处理

RTDB 的内部输出变量有 AIO、DIO 和 SIO。

内部输出变量的处理流程，RTDB 把 HMI 的命令映射到设备的内部输出变量 IO（AIO、DIO 或 SIO）。然后查询 RTDB 中变量的相关数据（变量控制状态、设备的状态等），检测是否允许执行命令，并把检查结果更新到 IO 变量的 execStatus 属性。最后，根据 execStatus 的状态值，终止或执行命令，并检测命令执行完成后的相关环境变量，然后更新到 IO 的对应属性，并返回结果给 HMI（图 9）。

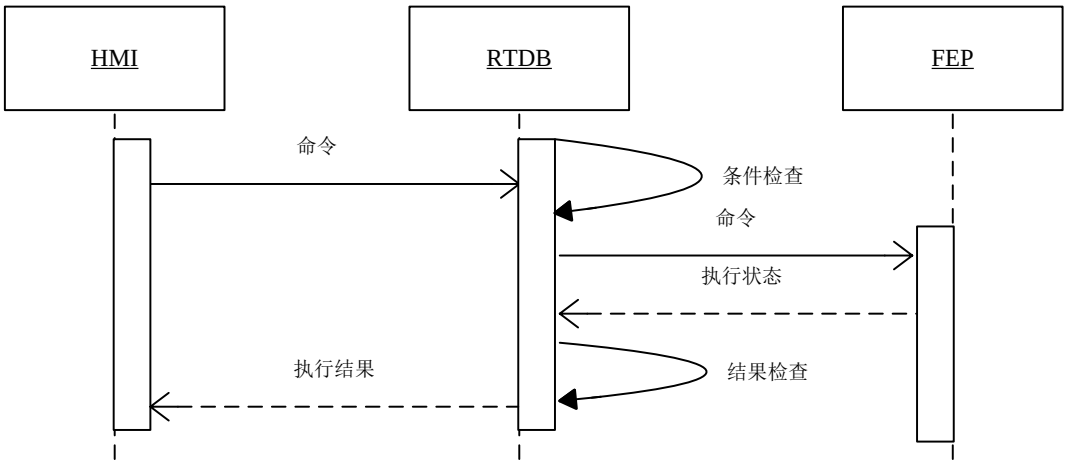

图 9 内部输出变量处理流程

RTDB 仅负责命令的初始化、命令执行环境检查（初始环境和执行后环境）和命令的转发，FEP 负责命令的执行。FEP 负责把命令发送给正确的协议插件，协议插件负责把命令发送给具体的设备（通过插件管理的外部输出变量数据库查找设备），由设备执行最终的控制命令。

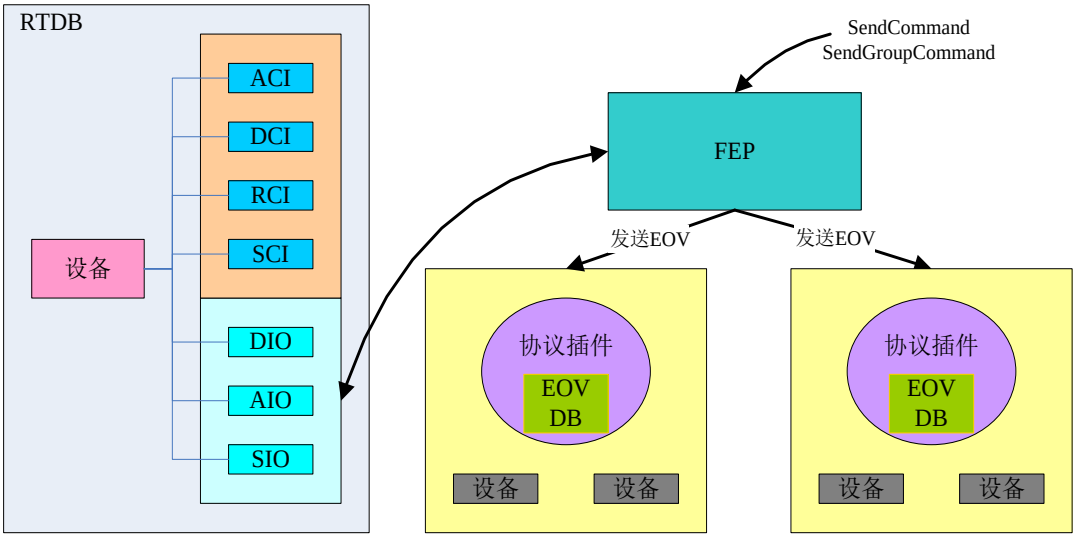

图 10 子系统间的命令执行

7.2.1. 离散输出变量处理

DIO 用于值为离散类型的命令，一个设备变量只有一个 DIO 变量。DIO 通过属性 valueTable 定义控制命令（最多 8 个命令），DOV 通过属性 veCoord 定义要发送的外部变量坐标（外部环境、外部变量名和变量值）。DIO 的 valueTable 中的命令与 DOV 是一一对应的，通过 valueTable 的 dovname 进行关联（图 11）。

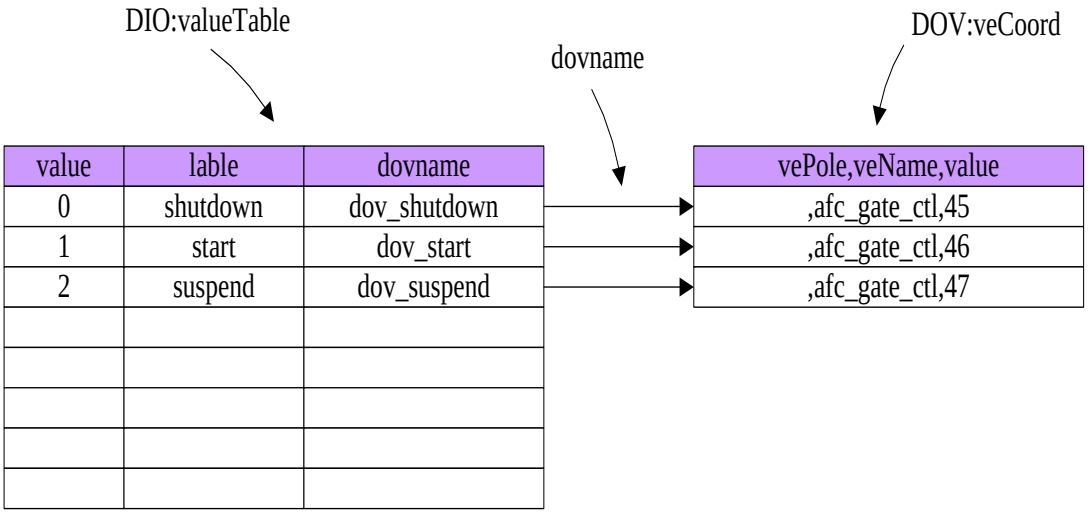

图 11 DIO-DOV 值

RTDB 从 HMI 接收离散类型的控制命令后，根据命令从 RTDB 找到目标设备变量的 DIO。首先，设置 DIO 属性 execStatus=1（等待命令初始化）。然后，使用命令对应的 DOV 变量，执行命令环境的初始化检测；如果 DOV 的 initCondTO ≠0，在 initCondTO 时间 initCondGL 仍为 FALSE，则设置 execStatus=5（初始化超时不能发送），否则设置 execStatus=0（命令初始化完成）。然后，执行命令发送，把 DOV 的 veCoord 的外部变量参数发送给 FEP 执行命令。最后，执行返回值检测；如果 returnCondTO ≠0，那么 returnCondGL 必须在 returnCondTO 时间内完成校验，并设置 execStatus=3（命令完成），否则设置 execStatus=5（返回条件超时），命令结束（图 12）。

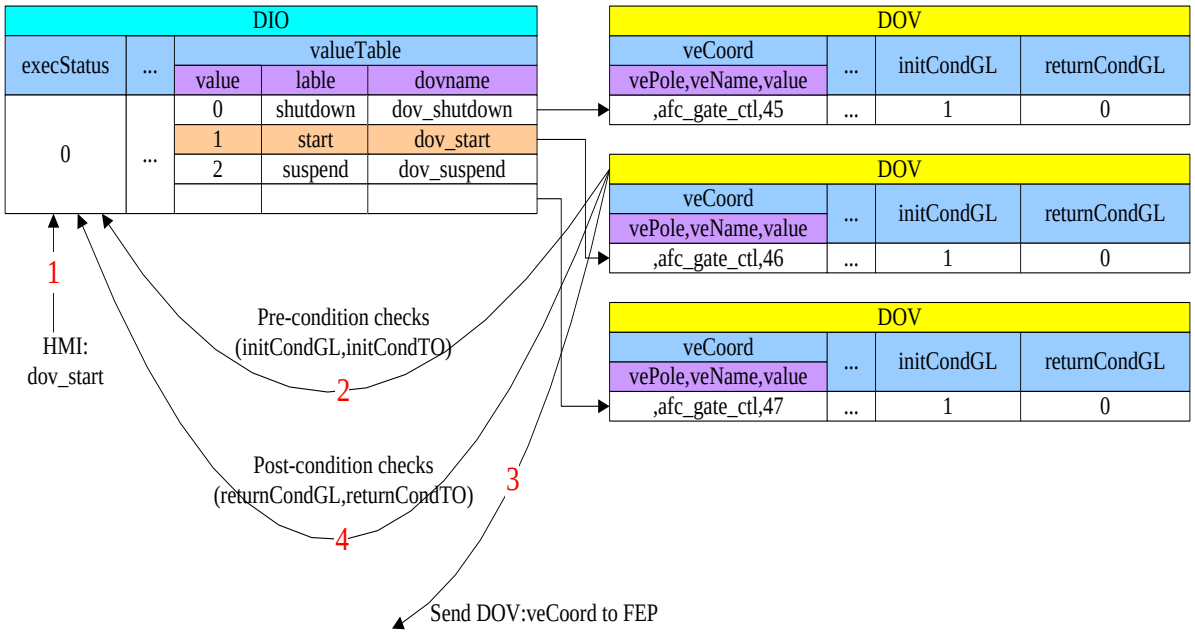

图 12 离散命令处理

7.2.2. 模拟输出变量处理

AIO 用于值为模拟类型的命令，一个设备变量可能有多个 AIO 变量。AIO 的属性 value 存放命令的当前值，属性 valueLimits 存放 value 的校验值，属性 veCoord 存放要发送的外部变量坐标（外部环境、外部变量名和变量值）（图 13）。

DIO					
execStatus	value	valueLimits		...	veCoord(vePole,veName,value)
0	3.12	2.11	5.46	...	,bas_mopeset,

图 13 AIO 的值

命令的处理过程同 DIO，不同之处在于，AIO 需要对命令值进行校验，只有在 valueLimits 范围内的 value 才是有效的命令值。value 的校验条件 valueLimits[0] <= value <= valueLimits[1]。

7.2.3. 结构化输出变量处理

目前的系统版本规划中，暂时不处理结构化输出变量。