1040609 并行升级方案

修订记录

版本号	更新时间	修改人	更新内容简述
V1.0.0	2024-7-11	王彦鹏	内容新增
V2.0.0	2024-7-17	王彦鹏	内容新增

1 概述

1.1 背景

随着车型架构越来越复杂，支持FOTA升级的ECU越来越多，为了提高刷写速率，满足客户需求， FOTA云端和车端应支持并行刷写策略。

1.2 目的

此文档的主要目的为明确并行升级的实现方案，详细描述并行升级功能逻辑等。以供相关人员参阅。

1.3 简介

并行升级主要是根据ECU的特性，通过对ECU的分组，使多个ECU可以同时升级，减少升级时间，给用户带来会更好的体验。

1.4 范围

此文档应用于支持并行刷写的车型项目开发工作。本文档主要读者为开发人员、测试人员、设计人员（UE、UI）等。

1.5 引用

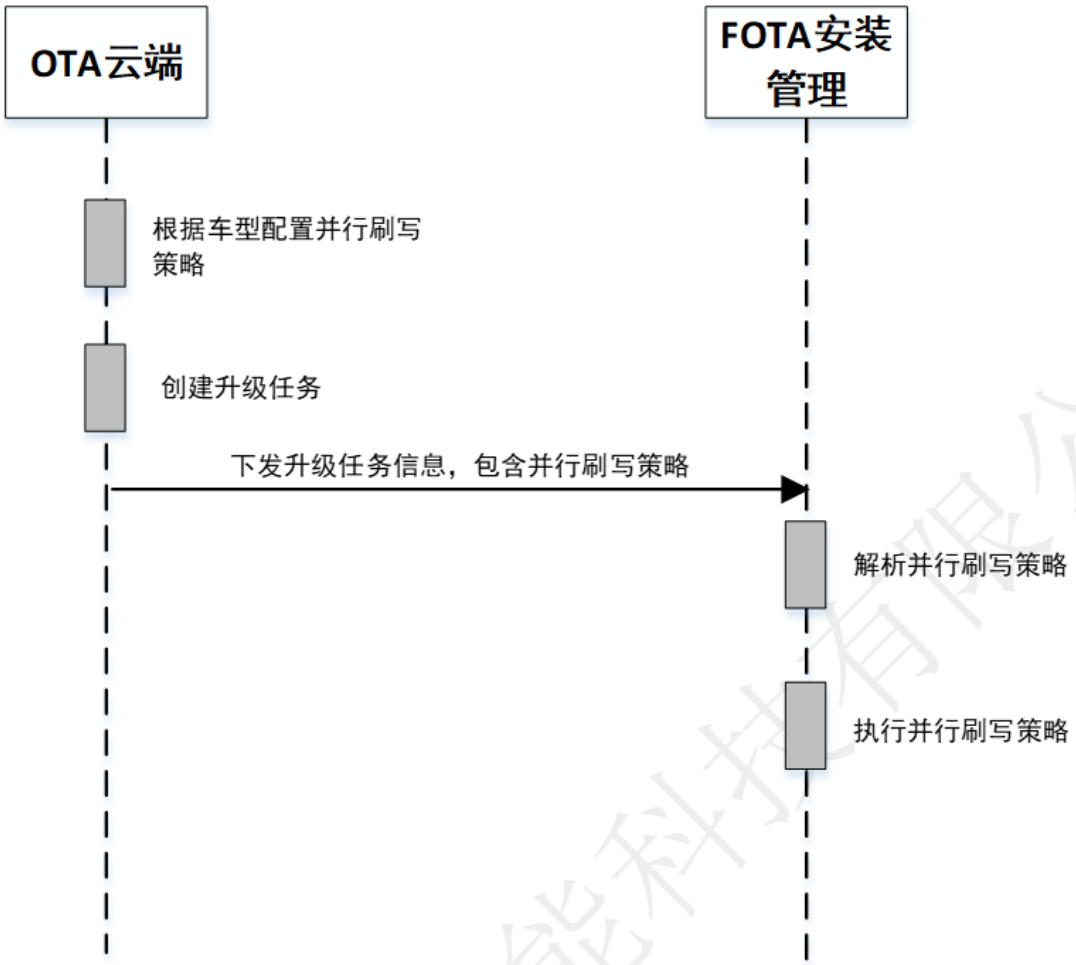
N/A

1.6 术语

缩写/术语	释义
FOTA	Firmware Over-The-Air/空中固件升级
OTA-Server	Firmware Over-The-Air Server/空中固件升级服务器端
OTA-Client	Firmware Over-The-Air Client/空中固件升级客户端
Installer	FOTA诊断刷写模块
UC-Master	车端FOTA升级控制主程序
零件组	指一组ECU，它们需要同时成功升级或回退。如果升级过程中任何一个ECU失败，整个零件组的升级都会失败，并且所有ECU将一起回退到之前的状态
并行组	指一组ECU，它们具有相同的升级顺序，并且在逻辑上构成一个集合。这些ECU可以同时开始升级，但它们的升级状态是相互独立的，一个ECU的升级成功或失败不会影响同一组内其他ECU的升级过程
ECU 升级序列	指根据ECU的特性和功能划分的升级优先级。用于标识各ECU的升级顺序、串行或并行关系。

2 产品架构

2.1 整体流程图



1. 云端按照不同车型拓扑结构以及ECU特性，配置每个车型的ECU的升级顺序；
2. 云端按照车型或ECU创建升级策略时，按照已配置的ECU升级顺序对应不同的车型进行保存；
3. 车辆前往云端检测新版本时，云端根据VIN所属的车型/车型配置查询到合适的升级顺序并跟随策略下发至OTA-Client；
4. UC-Master在升级开始时，解析云端下发的本次待升级ECU的升级顺序，通知Installer模块进行相关ECU的串行或并行安装；

2.2 功能列表

序号	功能模块	功能需求	功能描述	涉及ECU或SDK
1	**OTA云端**	配置并行刷写策略	OTA云端应支持根据车型配置并行刷写策略。	OTA-Server
2		FOTA升级任务信息包含并行刷写策略	在创建升级策略时，OTA云端应能根据升级策略自动生成升级目标ECU的并行刷写策略，升级策略中是否上高压升级会影响并行刷写策略。并行刷写策略应能够通过FOTA升级任务信息下发给车端。	OTA-Server
3	**FOTA安装管理**	从FOTA升级任务信息中解析和获取并行刷写策略	FOTA安装管理模块应能从FOTA升级任务信息中解析和获取并行刷写策略。	OTA-Client
4		支持并行刷写策略规则	FOTA安装管理模块应支持并行刷写策略的规则。	OTA-Client

5		执行并行刷写策略	在FOTA安装阶段，FOTA安装管理模块应根据OTA云端下发的并行刷写策略执行并行刷写。	OTA-Client; Installer;

FOTA云端和车端应支持并行刷写策略，以实现对升级任务中不同总线或同一总线上的目标ECU同时执行刷写流程，从而提高刷写速率。

3 规格说明

3.1 ECU升级顺序逻辑说明

3.1.1 ECU升级顺序配置规则

1. OTA-Server应支持根据车型配置并行刷写策略，并行刷写策略主要包括ECU升级顺序属性；
2. **ECU升级序列属性**：用于标识各ECU的升级顺序、串行或并行。ECU升级顺序属性通过upgradeOrder参数来表示，每个需要刷写的ECU必须分配且仅分配一个唯一的upgradeOrder参数值；

   （1） **最高位字节0xAA表示ECU所属的升级组。**对于相同0xAA的ECU，即为同一升级组，在同一升级组内进行串行或并行升级；对于不同0xAA的ECU，即为不同升级组，不能并行升级。不同升级组的升级顺序不同，0xAA值越小，升级优先级越高，前一0xAA升级全部完成后才可以继续进行后一0xAA的升级。0xAA升级组只表示ECU的升级顺序，不表示不同ECU的软件版本关联关系。不同ECU的软件版本关联关系通过在升级策略中配置零件组来表示；一个并行组应该是固定的高压或低压升级。

   （2） **中间字节0xBB表示ECU所属的0xAA升级组里面的升级小组。**对于不同0xBB的ECU，即为不同的小组，在不同的小组中的ECU可以进行并行升级；对于相同0xBB的ECU，即为同一小组，在同一小组的ECU也可以并行刷写，需要根据其并行能力进行升级。其中小组内升级的优先级需要根据最低字节0xCC进行判断，0xBB升级小组只表示0xAA升级组下面ECU的一个属性，没有优先级关系，具体根据车型架构定义；

   （3） **最低字节0xCC表示ECU 在整个升级方案的升级顺序。**在整个并行升级中，0xCC值越小，ECU升级优先级越高，按照整车的ECU升级逻辑给每个ECU赋予一个单独的0xCC值表示升级的优先级，用户也可以根据具体的需求更改CC的值来定义ECU的升级顺序；
3. 整个并行升级中可以同时**并行升级的ECU最大数目为M**：用于标识具体车型中，同一个0xAA升级组中可支持ECU并行刷写的能力，M默认为3，后续根据具体的项目需求在车端配置文件中进行配置；
4. **相同0xBB升级小组内可并行刷写的ECU最大数目为N**：用于标识具体车型中，同一个0xAA升级内可支持相同0xBB的ECU并行升级的能力，NA默认为1，后续根据具体的项目需求在车端配置文件中进行配置；
5. 在0xAA相同的升级组内，需要在优先保持整个并行升级中可以同时并行升级的ECU最大数目为M的前提下，根据N的数值采用轮询筛查的方法对该升级组下的ECU进行升级；

（1） 首先根据0xAA将升级的ECU分组，不同0xAA升级组之间是串行关系；

（2） 然后在相同的0xAA升级组内，在满足整个并行升级中可以同时并行升级的ECU最大数目为M的前提下，根据0xCC的先后顺序对要升级的ECU进行逐一筛选，筛选过程中需要根据 ECU的0xBB值判断该ECU是否和前面已经选入升级队列的ECU的0xBB相同，不相同则直接纳入队列；如果相同，需要判断加上该ECU之后,此时相同0xBB的ECU是否超过了N个，如果未超过将该ECU继续加入升级的队列；如果超过了则将该ECU暂时放回队列中，暂不升级，选择下一个ECU进行判断；依次逐一筛查，直到达到升级的最大数目M后暂时停止筛查，等待其中一个ECU升级完成后，从剩余ECU中根据CC的优先级再次进行筛查补充升级，直到所有ECU升级完毕；

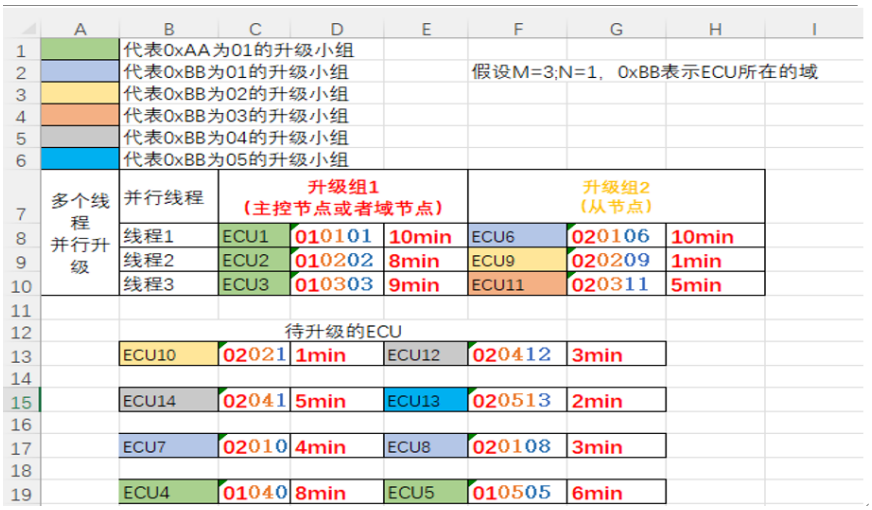
（3） 例如:某个车型的N=3,M=10,可以支持并行升级的0xBB升级小组有5个,则在同一个0xAA升级组内,根据0xCC的值的大小，按照相同0xBB的ECU不超过3个的原则优先筛选10个ECU进行升级；

（4） 结构示例：

	代表0xBB为01的升级小组				假设M=10;N=3，0xBB表示ECU所在的域									
	代表0xBB为02的升级小组													
	代表0xBB为03的升级小组													
	代表0xBB为04的升级小组													
	代表0xBB为05的升级小组													
	并行线程	升级组1（主控节点或者域节点）			升级组2（从节点）			待升级的ECU						
多个线程并行升级	线程1	DHU	010101	10min	ECU1	020106	10min	ECU15	020320	6min	ECU16	020421	3min	
	线程2	ADS	010202	8min	ECU2	020107	8min							
	线程3	VGM	010303	9min	ECU3	020108	9min	ECU17	020422	5min	ECU18	020523	2min	
	线程4	CEM	010404	1min	ECU6	020211	1min							
	线程5	VCU	010505	5min	ECU7	020312	5min	ECU19	020524	6min	ECU20	020525	7min	
	线程6	…			ECU8	020213	6min							
	线程7				ECU9	020314	7min	ECU4	020109	4min	ECU5	020110	1min	
	线程8				ECU10	020415	8min							
	线程9				ECU11	020316	9min	ECU12	020317	6min	ECU13	020318	6min	
	线程10				ECU14	020519	10min							

通过0xCC升级顺序，以轮询筛选的形式进行最多10个ecu,每个域不超过3个ecu同时升级的方式进行刷写
当遇到0xBB为01的ECU同时升级的个数变成4个时，则将该ECU放置队伍的末尾，进行下一个ECU的识别；
当10个ECU中其中一个升级完成时，按照顺序由下一个进行识别补充升级；

M=10,N=3示例图

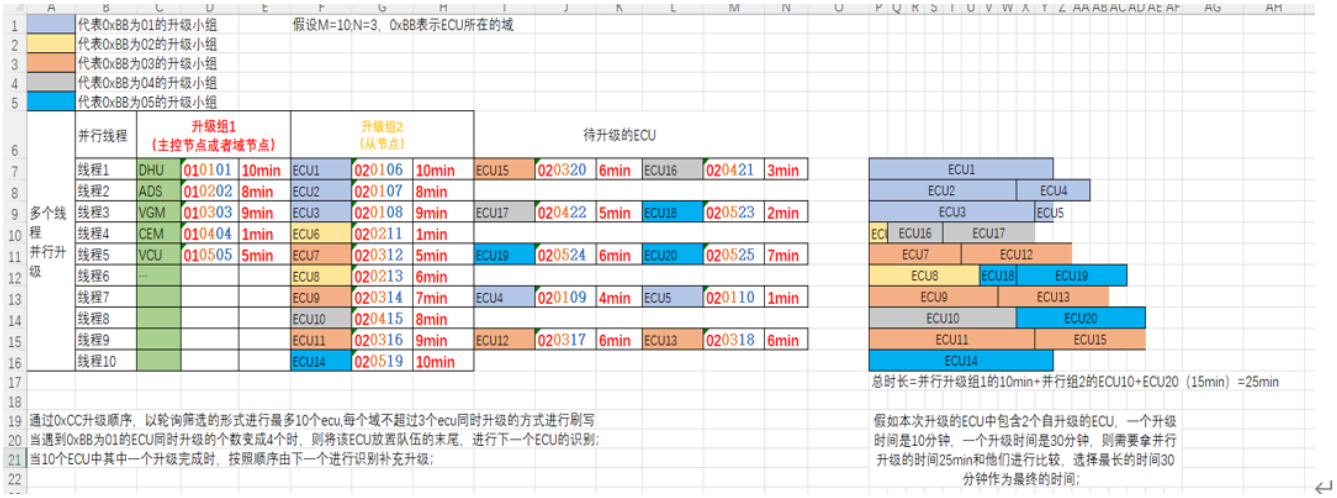


	代表0xAA为01的升级小组						
	代表0xBB为01的升级小组				假设M=3;N=1，0xBB表示ECU所在的域		
	代表0xBB为02的升级小组						
	代表0xBB为03的升级小组						
	代表0xBB为04的升级小组						
	代表0xBB为05的升级小组						
多个线程并行升级	并行线程	升级组1（主控节点或者域节点）			升级组2（从节点）		
	线程1	ECU1	010101	10min	ECU6	020106	10min
	线程2	ECU2	010202	8min	ECU9	020209	1min
	线程3	ECU3	010303	9min	ECU11	020311	5min
	待升级的ECU						
	ECU10	02021	1min		ECU12	020412	3min
	ECU14	02041	5min		ECU13	020513	2min
	ECU7	02010	4min		ECU8	020108	3min
	ECU4	01040	8min		ECU5	010505	6min

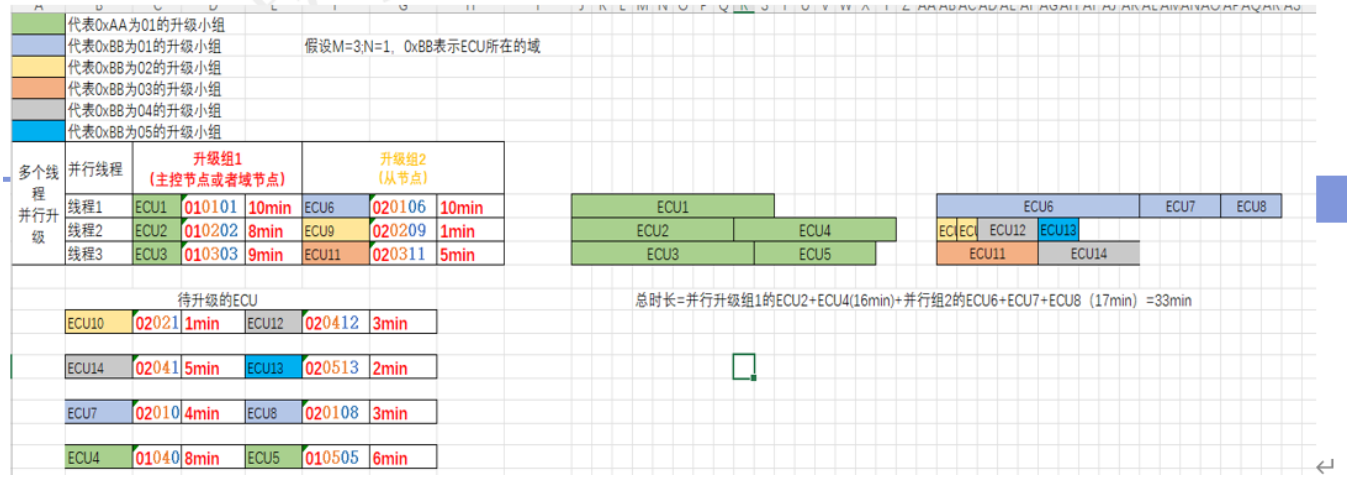
M=3,N=1示例图

- （5） 当N=1，M≠1时，表示相同0xBB升级小组内的ECU只能串行升级；当M=1时，表示该车型不支持并行升级并行升级时升级的所有ECU的预计耗时的时间；（本次并行升级最终结果为正常升级，并且全部升级，升级过程中无卡死等异常情况出现）

- （1） 云端根据上述并行升级的规则，在结合每个ECU的预计升级时长，模拟出本次升级的所有ECU的顺序；首先将ECU按照AA分成不同的升级组，不同升级组之前串行升级，升级时间需要相加；然后根据0xAA升级组内ECU的升级顺序，计算出M条线程各个线程的升级时间，取消耗最长的时间作为该0xAA升级组的预计升级时间；最后再将每个0xAA升级组预计升级时间相加作为本次升级的预计安装时间；
- （3） 云端需要根据每个车辆实际要升级的ECU进行预计升级时间计算；
- （4） 每个单独的ECU升级时长继承于软件版本配置的升级时长；
- （5） 结构示例：



M=10,N=3示例图

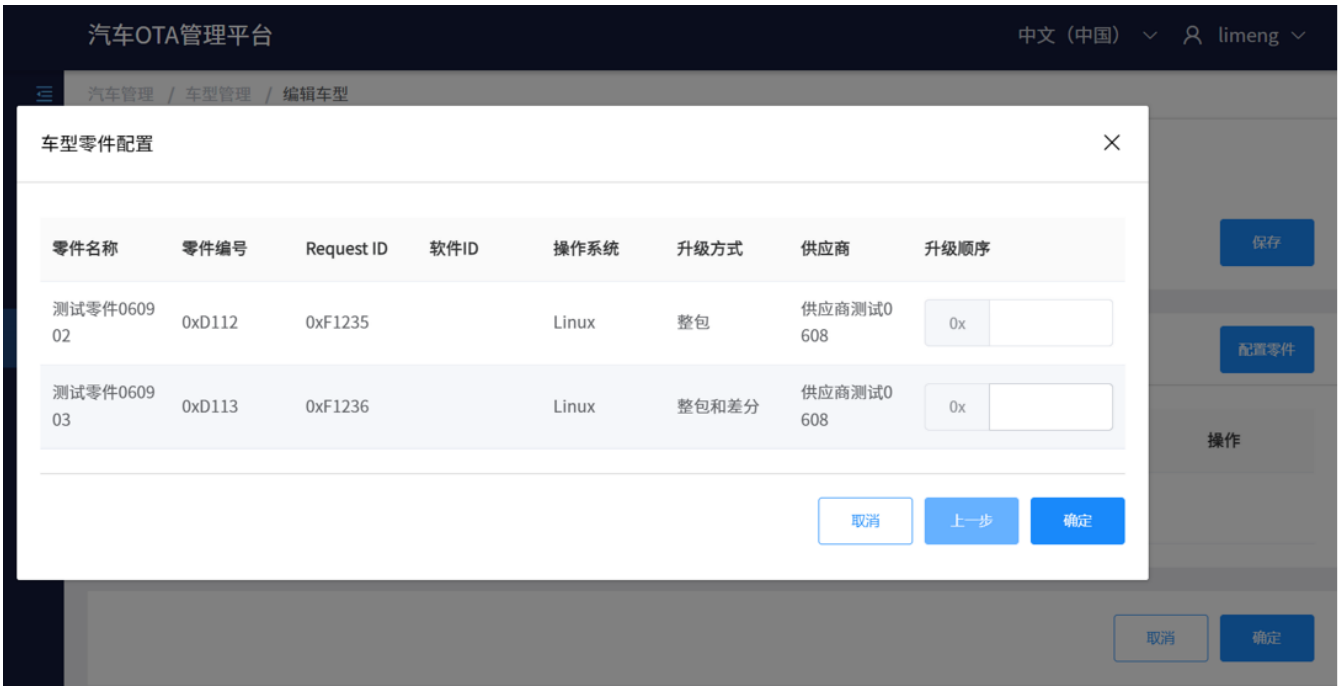


M=3,N=1示例图

3.1.2 原型设计

1. 平台在配置OTA配方时，填写“升级顺序”内容
    - 配置完升级顺序后，点击【确认】进行保存，并可以在列表中进行查看；
2. upgradeOrder参数由五个字节组成，即0xAABBCC；
    - 前端控制校验，仅允许填写0xAABBCC格式的内容，其中ABC均为0-9的正整数；
    - 0x应考虑在前端做成默认前缀，用户不用填写0x，仅需要填写AABBCC的数字；
        a. 存在写入数字长度校验6位；
        b. 每个ECU必须填写相应的升级顺序；
    - 点击【取消】，放弃本次升级顺序的配置，页面返回【配置零件】界面；
    - 点击【上一步】，放弃本次升级顺序的配置，页面返回【配置零件】界面；

点击【确定】，保存本次升级顺序的配置，，页面返回【车型零件信息】界面；



3.1.3 ECU升级顺序在车型和车型配置中的继承

1. 当为某车型配置所有ECU的升级顺序后，并为该车型下辖的车型配置添加了部分ECU，则该车型配置包含的所有ECU自动继承车型中对应ECU的升级顺序；
    - 举例：假定某车型中配置了5个ECU分别为ABCDE，且ABCDE的升级顺序依次为0x010101、0x010102、0x010103、0x010104、0x010105，并存在高配车型存在ABCD四个ECU、低配车型存在ABE三个ECU，则对于高配车型配置自动继承的升级顺序为0x010101、0x010102、0x010103、0x010104，低配车型配置自动继承的升级顺序为0x010101、0x010102、0x010105；

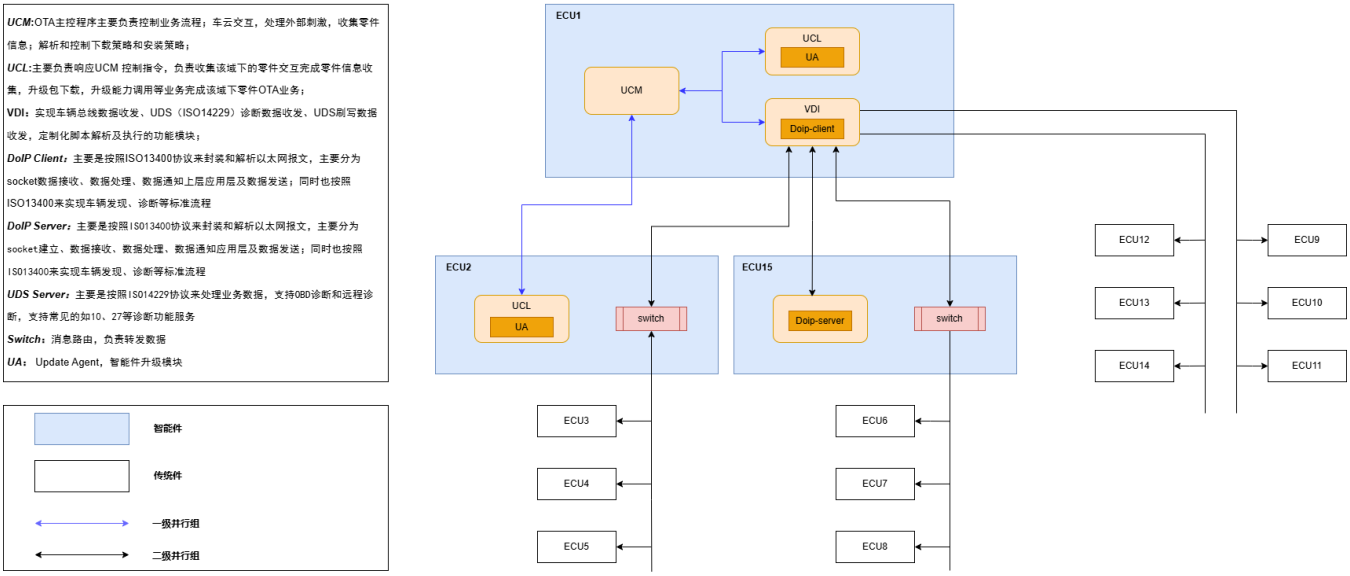
3.1.4 ECU升级顺序在策略配置中的使用

1. 对策略中配置的任意ECU升级，需要分别按照该ECU所属车型/车型配置进行保存；
    - 举例：如配置ABCDE等5个ECU同时归属于2个车型中，进行一次升级配置，则策略应分别针对2个车型对ABCDE的升级顺序分别进行保存；
    - 对车型一：ABCDE（顺序升级，0x010101，...），对车型二：EDCBA（顺序升级0x010101，...）；
2. 任务发布后，任意VIN前往OTA-Server进行检测时，车云接口处应先识别该VIN归属的车型，再提取由策略保存的对应的升级顺序返回；
    - 按上述描述，对于标准策略，存在仅检测到车型一ABD三个ECU，则应提取该车型的ABD的升级顺序进行下发；
3. OTA-Server下发给车端的FOTA升级任务信息中包含默认的并行刷写策略信息，包括ECU刷写顺序属性InstallationOrder参数值，本次升级的预计升级时间；

3.2 HMI显示规则

1. 总体原则：整车升级进度与单个ECU安装进度只允许前进不允许后退，当出现重试时，进度卡住；
2. 整体升级结果：
    - 本次升级中包含的所有ECU全部升级成功，则为升级成功；
    - 本次升级中存在ECU升级失败，则为升级失败；
3. 整体进度：
    - 当前正在进行中的若干个ECU的单个进度求和平均值；

4 并行升级设计规则



规则：

规则一：同一级并行组，零件可以并行升级（ECU1和ECU2可以并行升级）。
规则二：高并行组可以和相邻低并行组零件并行升级（ECU2和ECU15可并行）。
规则三：同一通道零件不并行（ECU3和ECU4不能并行升级，EUC6和ECU7不并行）。
规则四：OTA过程中零件会重启，为保障成功率，从属关系零件不并行（ECU1与ECU3 、ECU4、EUC15等不并行；EUC2与ECU3、ECU4、ECU5不并行）。

5 并行刷写需求约束

5.1 关于并行升级，对网关的需求

要求	详细说明	备注
支持DoIP（ISO-13400、ISO14229）全栈能力	1. 支持UDS Server，用于通过以太网来响应UDS Client的诊断请求； 2. 支持UDS Client，用于通过以太网来诊断其它UDS Server节点；	
支持UDS ON CAN全栈能力（ISO 15765、ISO14229）	1. 支持UDS Server，用于通过can/canfd来响应UDS Client的诊断请求； 2. 支持UDS Client，用于通过can/canfd来诊断其它UDS Server节点；	
支持将DOIP诊断数据路由至CAN/CAN-FD/LIN等类型总线		
支持将DOIP诊断数据路由至其他网段总线		
支持多路诊断数据同步执行	1. DOIP并行刷写； 2. 不同can路并行刷写；	并行刷写要求
支持透传基于SOCKET传输的数据		某两个ECU之间通过SOCKET进行数据传输，CGW需要进行数据透传

UDS仲裁机制	1. OBD、FOTA、DOTA优先级仲裁； 2. 支持诊断模式的查询；	
支持UDS会话维持功能寻址3E80发送		

并行升级对零件升级环境要求：**一个并行组应该是固定的高压或低压升级。**

5.2 并行刷写属性及策略

5.2.1 并行升级零件基础属性（获取配置信息返回的 零件配置信息中零件的属性）

字段	类型	说明
ecuReqId	String(64)	诊断请求id
ecuSwDid	String(64)	升级对象软件版本 DID（用于标识零件内的模块，如IVI中的MCU标识）
PartitionType	Int	零件分区类型 取值范围： 1：单分区 2：全A/B分区 3：部分A/B分区
instCondition	Int	安装约束,指该零件需要处于低压或高压条件下才可以安装 取值范围： 1: 高压下（上高压/上电） 2: 低压下（下高压/下电）

5.2.2 无感更新零件任务属性

字段	类型	说明
ecuReqId	String(64)	诊断请求id
ecuSwDid	String(64)	升级对象软件版本 DID（用于标识零件内的模块，如IVI中的MCU标识）
deployType	Int	是否采用无感升级 取值范围： 1：不采用无感升级 2：采用无感升级

5.2.3 并行升级策略信息

ecuGroupOrder 零件组、升级顺序信息对象：

字段	类型	说明
groupFlag	Int	零件分组标识，若多个零件的 groupFlag 若相同，则表明这多个零件属于同一个零件组
actionForFailed	Int	升级失败后动作 取值范围： 1：rollback 回滚 2：stop 停止升级 3：continue 继续升级

groupEcus	List	零件组内 Ecu 列表

groupEcus 零件组内ECU对象

字段	类型	说明
ecuReqId	String (64)	诊断请求id
ecuSwDid	String (64)	升级对象软件版本 DID（用于标识零件内的模块，如IVI中的MCU标识）
upgradeOrder	Int	零件升级顺序（依据车型配置中的零件顺序） 零件升级顺序，格式：0xAABBCC，AA、BB、CC 取值范围均为：[0-F] AB 表示升级组，升级组越小，升级优先级越高。 BB 表示网段，若升级组相同，网段不同，网段越小，升级优先级越高。 CC 表示刷写顺序，若升级组相同，网段相同，刷写顺序越小，升级优先级越高；若升级组相同，网段相同，刷写顺序相同，则升级优先级相同。
ecuTimeConsuming	Int	Ecu 预计升级耗时，单位：S
ecuTimeout	Int	Ecu 升级超时时间，单位：S

5.3 刷写失败异常处理策略

- ECU刷写失败后的回滚是立即回滚，还是需要等待所有ECU安装完成在进行回滚?
  - 需要根据安装策略进行对应处理（actionForFailed，回滚、停止升级、继续升级）

- 当前刷写失败的ECU会立即回滚，该零件组内已经刷写成功的ECU需要等到所有ECU（包括其他零件组的ECU）刷写完成后才执行回滚操作

- 当前ECU刷写失败后，若安装策略中配置回滚或停止，则该ECU所属的零件组内的尚未执行刷写的ECU不会进行后续刷写操作。

- ECU刷写失败后，同一个零件组内的ECU需要回滚(若零件组配置需要回滚)，那么不同零件组的已经刷写的ECU是否需要回滚?

— 不需要，不同零件组之间互不影响。

- 回滚有并行吗?

- 单零件立即回滚，所有零件刷写完成后的回滚是并行回滚。

5.4 零件升级接口要求（主要针对智能件）

升级接口：参数中必须携带升级类型（type），

— 无感升级，针对全A/B分区零件及部分A/B分区零件，执行A/B件分区刷写。

— 无感升级失败的有感升级，对全A/B分区零件，执行A/B件分区刷写。对部分A/B分区零件，A/B分区模块进行刷写及其他模块刷写。

— 无感升级成功的有感升级，对部分A/B分区零件，执行非A/B分区模块进行刷写。

激活接口：针对全A/B分区零件及部分A/B分区零件，

— 全A/B分区零件，执行A/B分区切面。

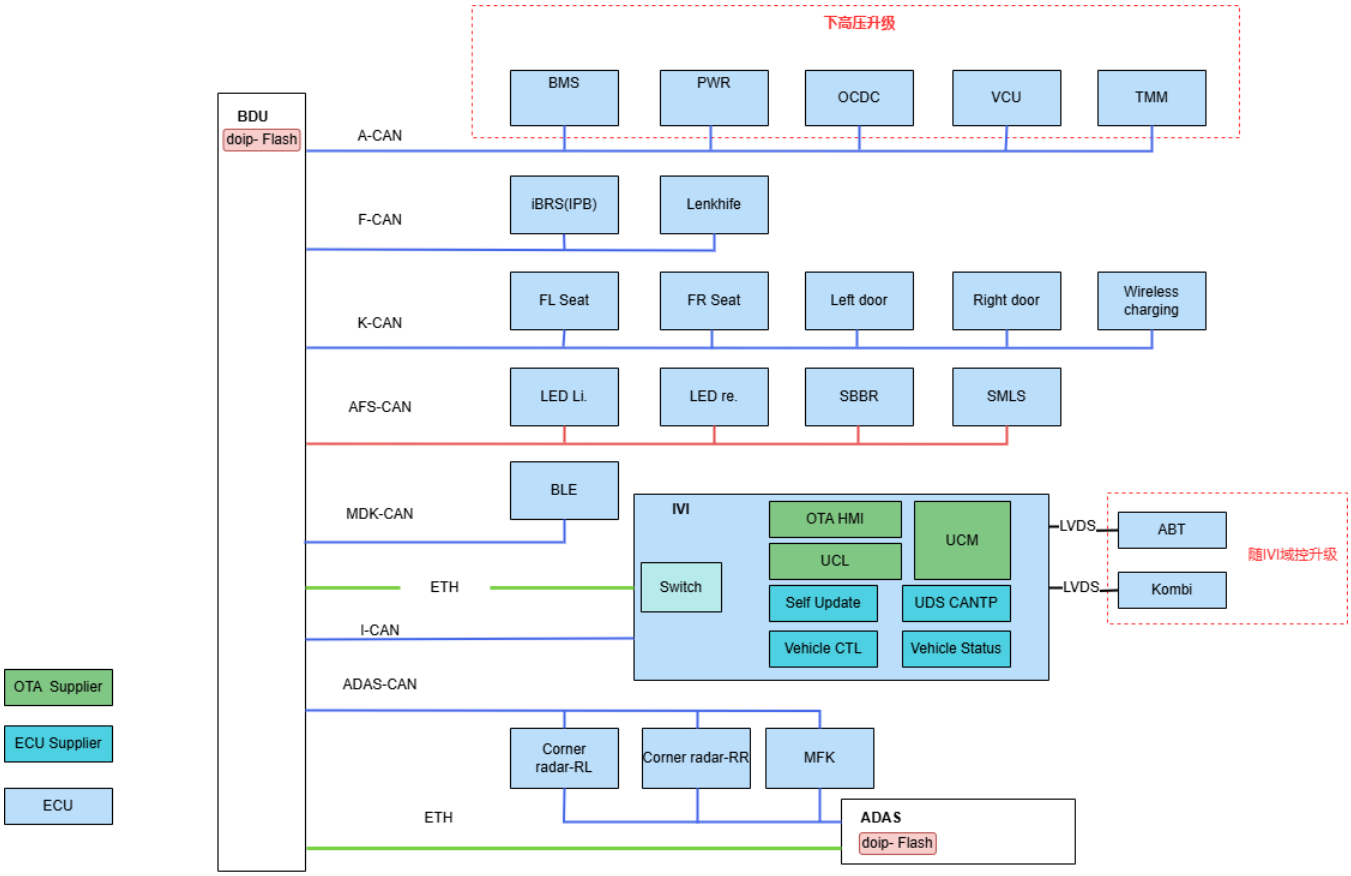
— 部分A/B分区零件，执行A/B分区模块切面。

回滚接口：

— 有对应升级包的，执行升级回滚。

— 无升级包的自己回滚（若自身支持回滚，如A/B分区零件，由A面还原到B面）。

6 一汽大众 J01 项目并行升级序列



Analysis:

1. **分析零件的特性:**
    a. IVI 集成 UCM，IVI具备A/B分区，故IVI可以与其他零件并行（除了 A-CAN下面的低压件升级不能并行）。
    b. BDU 下挂很多传统ECU，故BDU不能与下挂件并行刷写。（这里需要注意，ADAS依赖BDU的switch，故BDU与ADAS不能并行刷写）
    c. ADAS 下挂于 BDU，所以ADAS不能与BDU并行。
    d. A-CAN 下挂ECU都需要在车辆处于低压下刷写。
    e. ABT、Kombi 零件随 IVI 域控升级。
    f. 需要考虑零件之间的依赖，例如：ADAS-CAN下面的ECU，依赖ADAS升级成功后后才能刷，此时ADAS就需要与ADAS-CAN下挂零件独立成两个并行组。
2. **分析CAN总线能力:**
    a. 同一条CAN总线上最多只能并行一个，防止并行时CAN丢包问题。
    b. BDU下面的几条CAN线可以并行，即A-CAN、F-CAN、K-CAN、AFS-CAN、MDK-CAN、ADAS-CAN 可以并行执行刷写。
3. **OTA属性的影响:**
    a. 是否需要考虑 “OTA模式” 是否需要维持，以及有谁来维持，维持者需要考虑最后升级，如果维持者是AB分区件，则无影响。
4. **并行升级能力M、N的确定原则:**
    a. **最大并行升级能力 M，一般会从如下两个角度来衡量:**
        i. **刷写引擎的性能:** 刷写引擎（如uds-clien）每增加一路并行升级，都会对宿主设备的资源（如内存和CPU）产生影响。如果资源不足，会导致最大并行能力受限。
        ii. **网关转发能力:** 设备中的网关（如BDU）能够转发的最大并行消息数量。例如，本J01车型中BDU支持七路不同CAN总线及两路ETH总线并行消息转发，那么它的并行能力就是9路。
    b. **并行小组 N，一般会根据 BB 具体含义来定:**
        i. 本项目中 BB 表示一条 CAN/ETH，而一条 CAN 最多支持一路下挂件的刷写（主要考虑CAN总线的负载率，若并行多个下挂件可能会出现丢包问题）。

Summary:

并行组　0xAA，可以分为三组：低压件、BDU、其他传统件

并行小组 0xBB，可以按照CAN总线来区分，即，F-CAN、K-CAN、AFS-CAN、MDK-CAN、ADAS-CAN
升级顺序 0xCC，需要考虑零件之间是否有依赖，以及升级时长来排序。

下图为根据J01目前提供的零件信息及属性，根据上述分析给出的参考的ECU并行升级序列，具体实施时需要完善零件属性和车型能力进行定义并行升级序列：