

AMI CAN 报文说明

版本：AMI-2.7-2026-10-07。由 AMI.dbc 自动生成，13 条报文、53 个信号。

使用范围

CAN1 为 500 kbit/s，CAN2 为 250 kbit/s，均为 Classic CAN、87.5% 采样点。两路保留相同的接收 ID、解码与任务发送功能，MCU 不在两路之间转发原始报文。节点 AMI 为仪表 MCU，VCU 为任务请求的逻辑接收者，SDC_ACQ_DAC 和 BMS 为数据源。

端口	速率	RX / TX 引脚	DBC 文件
CAN1	500 kbit/s	PA11 / PA12	AMI.dbc
CAN2	250 kbit/s	PB0 / PB1	AMI_CAN2_250k.dbc

DBC 的 Baudrate / AMIPort 分别标注所属总线。CAN1 DBC 含 13 条报文、53 个信号；CAN2 DBC 含 12 条报文、49 个信号；仅 CAN1 包含 0x301 心跳，其余报文布局相同。CAN 工具必须按接入端口设置速率，导入 DBC 不一定自动改硬件速率。实际接法：SDC 500 kbit/s 接 CAN1，BMS 250 kbit/s 接 CAN2。

FDCCAN 时钟 64 MHz，时钟分频为 1；NominalPrescaler 为 CAN1=8、CAN2=16，两路均使用 Sync=1、TimeSeg1=13、TimeSeg2=2、SJW=1。速率 = 64 MHz / Prescaler / (1+13+2)；采样点 = (1+13)/16 = 87.5%。

本 DBC 包含 AMI 发送的 0x300 任务请求和 0x301 周期心跳，以及 AMI 当前识别的 SDC / BMS 和自定义显示输入。SDC 0x6A3 DAC 控制、0x6A5 ASSI 控制、BMS 其他电芯/故障字段不属于此接口子集。0x182 新增扭矩千分、电机温度和电机控制器 MCU 温度的仪表显示输入；发送者负责转换实际电机控制器协议。0x183 为新增 VCU 状态反馈，需 VCU 按此自定义接口发送；车速取 SDC 双前轮轮速平均值。旧 0x180 / 0x181 / 0x280 输入已停用。

SDC 字段依据 SDC_ACQ_DAC.dbc VERSION 1.1，核对远端 master 提交 7bc2d9f1eaaf19814957037bd097cd879cbf4cbf（2026-10-05），保持其信号名、位布局、缩放和单位；仅将输入接收节点设为 AMI。BMS 字段依据 BmsMonitor-CANTest 提交 e65c95203d1689df05343775982d7e85bd782646 的 BmsProtocolParser.cs。

DBC 为 ASCII 编码，注释和枚举使用英文，便于导入 CAN 工具；本文为 UTF-8 中文。物理值 = 原始值 × Factor + Offset；0x182 的三个数值为有符号 int16，其他数值按表定义。下表范围为接口允许范围，不表示车辆正常工作范围。

AMI 周期心跳 0x301

2026-10-04 新增：标准 ID 0x301、Classic CAN 数据帧、DLC 8。仅 CAN1（500 kbit/s）每 **100 ms** 尝试发送，CAN2 不发送心跳，不依赖接收活动或屏幕连接。首帧在启动约 100 ms 后发送。

字节	含义
0	协议版本，固定 1
1	保留，固定 0
2..3	uint16 小端心跳序号，首帧 0，每个调度周期加 1，65535 后回到 0
4..7	uint32 小端 HAL_GetTick() 运行时间，单位 ms，自然回绕

例如运行 100 ms 的首帧：01 00 00 00 64 00 00 00。任务请求先处理，心跳仅在发送 FIFO 至少有两个空位时入队，为任务保留一个空位；忙或入队失败时跳过本周期，不在软件中积压或补发。硬件自动重传仍启用，已入硬件 FIFO 的帧可能延迟到总线恢复后发出。100 ms 为主循环调度周期，实际总线时刻受仲裁/错误重传影响。序号按调度递增，因此丢帧时可以跳号。

仅表示 AMI 主循环运行，不表示屏幕、VCU 或车辆状态正常。2026-10-04 已烧录并全量回读一致；CAN1 抓取 121 帧，间隔 99.952~100.180 ms，序号连续，详见 [心跳测试报告](#)。CAN2 未接线抓帧。

报文总览

CAN ID	类型	DLC	报文名	方向和作用	周期 / 触发
0x182	标准	8	AMI_DrivetrainDisplay	VCU / 测试器 -> AMI: 扭矩千分、电机温度、MCU 温度	100 ms
0x183	标准	8	AMI_VCUStatus	VCU -> AMI: 实际当前任务、锁定状态、无人状态、EBS 状态	100 ms
0x300	标准	8	AMI_TaskRequest	AMI -> VCU: 用户确认后的任务选择请求	确认任务触发
0x301	标准	8	AMI_Heartbeat	AMI -> VCU (仅 CAN1): 版本、序号、运行时间	100 ms
0x6A0	标准	8	SDC_Heartbeat	SDC -> AMI: 板标识、版本、序号、上电时间	100 ms
0x6A1	标准	2	SDC_ADC_InputVoltage	SDC -> AMI: 低压电池电压, 精确 DLC=2	500 ms
0x6A2	标准	4	SDC_WheelSpeed	SDC -> AMI: 左前 PA0 / 右前 PA6 已标定轮速	20 ms
0x6A4	标准	4	SDC_DAC_Ack	SDC -> AMI: DAC 设置应答, 识别但不显示	DAC 设置后应答
0x6A6	标准	8	SDC_GPIO_Status	SDC -> AMI: PD0..PD14 原始电平和变化掩码	1000 ms
0x140	标准	8	BMS_Status	BMS -> AMI: 高压总电压、带偏置的总电流	源协议未指定
0x156	标准	8	BMS_Temperatures	BMS -> AMI: 八路实际温度, AMI 派生最高有效温度	源协议未指定
0x186040F3	扩展	8	BMS_Status_Extended	BMS -> AMI: 高压总电压、总电流扩展帧别名	源协议未指定
0x186140F3	扩展	8	BMS_Extremes_Extended	BMS -> AMI: 最高单体温度; 不使用 BMS 低压字段	源协议未指定

扩展帧在 DBC 的 B0_ 十进制 ID 中设置 bit31: 0x186040F3 -> 2556444915, 0x186140F3 -> 2556510451。CAN 工具中使用实际 29 位仲裁 ID 并选择扩展帧; bit31 不是总线上发送的仲裁 ID。

GenMsgCycleTime 填写已确认的 SDC 周期 100 / 500 / 20 / 1000 ms, 以及 0x182、0x183 和 0x301 的约定周期 100 ms。其余帧未填写周期, 默认 0 表示未指定, 不据此推断 BMS 周期。

信号布局

Start 为 DBC 起始位，位号从 0 开始。Intel @1 从最低有效位起算；Motorola @0 从最高有效位起算。例如 BMS 的 7|16@0+ 对应字节 0..1 大端无符号数，23|16@0+ 对应字节 2..3。单字节温度的字节序不改变实际解码。

0x182 - AMI_DrivetrainDisplay

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Torque_Permille	0	16	Intel	1	0	permille	-1000..1000
Motor_Temperature	16	16	Intel	0.1	0	degC	-3276.8..3276.7
MCU_Temperature	32	16	Intel	0.1	0	degC	-3276.8..3276.7
Torque_Valid	48	1	Intel	1	0	-	0..1
Motor_Temperature_Valid	49	1	Intel	1	0	-	0..1
MCU_Temperature_Valid	50	1	Intel	1	0	-	0..1
Protocol_Version	56	8	Intel	1	0	-	1..1

Torque_Valid: 0 = Invalid, 1 = Valid; Motor_Temperature_Valid: 0 = Invalid, 1 = Valid; MCU_Temperature_Valid: 0 = Invalid, 1 = Valid; Protocol_Version: 1 = Version_1。

0x183 - AMI_VCUStatus

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Current_Task	0	8	Intel	1	0	-	0..6
Locked	8	8	Intel	1	0	-	0..1
EBS_Normal	16	8	Intel	1	0	-	0..1
Task_Valid	24	1	Intel	1	0	-	0..1
Lock_Valid	25	1	Intel	1	0	-	0..1
EBS_Valid	26	1	Intel	1	0	-	0..1
AS_Valid	27	1	Intel	1	0	-	0..1
Feedback_Sequence	32	8	Intel	1	0	-	0..255
AS_State	40	8	Intel	1	0	-	0..5
Reserved	48	8	Intel	1	0	-	0 (固定)
Protocol_Version	56	8	Intel	1	0	-	1..2

Current_Task: 0 = None, 1 = Skidpad, 2 = Acceleration, 3 = Autocross, 4 = Inspection_Test, 5 = EBS_Test, 6 = Manual_Driving; Locked: 0 = Unlocked, 1 = Locked; EBS_Normal: 0 = Fault, 1 = Normal; Task_Valid: 0 = Unknown, 1 = Valid; Lock_Valid: 0 = Unknown, 1 = Valid; EBS_Valid: 0 = Unknown, 1 = Valid; AS_Valid: 0 = Unknown, 1 = Valid; AS_State: 0 = AS_OFF, 1 = AS_Ready, 2 = AS_Driving, 3 = AS_Finished, 4 = AS_Emergency, 5 = Manual_Driving; Protocol_Version: 1 = Legacy_Version_1, 2 = Version_2。

0x300 – AMI_TaskRequest

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Protocol_Version	0	8	Intel	1	0	–	1..1
Task_ID	8	8	Intel	1	0	–	1..6
Request_Sequence	16	8	Intel	1	0	–	0..255
Request_Source	24	8	Intel	1	0	–	1..1
MCU_Uptime	32	32	Intel	1	0	ms	0..4.29497e+09

Protocol_Version: 1 = Version_1; Task_ID: 1 = Skidpad, 2 = Acceleration, 3 = Autocross, 4 = Inspection_Test, 5 = EBS_Test, 6 = Manual_Driving; Request_Source: 1 = AMI_HMI。

0x301 – AMI_Heartbeat

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Protocol_Version	0	8	Intel	1	0	–	1..1
Reserved	8	8	Intel	1	0	–	0（固定）
Heartbeat_Sequence	16	16	Intel	1	0	–	0..65535
MCU_Uptime	32	32	Intel	1	0	ms	0..4.29497e+09

0x6A0 – SDC_Heartbeat

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Board_Identifier	0	8	Intel	1	0	–	0..255
Protocol_Version	8	8	Intel	1	0	–	0..255
Heartbeat_Sequence	16	16	Intel	1	0	–	0..65535
Uptime	32	32	Intel	0.001	0	s	0..4.29497e+06

Board_Identifier: 161 = A11_F407; Protocol_Version: 1 = Version_1。

0x6A1 – SDC_ADC_InputVoltage

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Input_Voltage	0	16	Intel	0.001	0	V	0..65.535

0x6A2 – SDC_WheelSpeed

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
WheelSpeed_LeftFront_PA0	0	16	Intel	0.001	0	m/s	0..65.535
WheelSpeed_RightFront_PA6	16	16	Intel	0.001	0	m/s	0..65.535

0x6A4 – SDC_DAC_Ack

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
DAC_Set_Result	0	8	Intel	1	0	–	0..1
DAC_Select	8	8	Intel	1	0	–	1..2
DAC_Output_Voltage	16	16	Intel	0.001	0	V	0..5

DAC_Set_Result: 0 = Failed, 1 = OK; DAC_Select: 1 = DAC1, 2 = DAC2。

0x6A6 – SDC_GPIO_Status

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
SDC_Input_State	0	15	Intel	1	0	–	0..32767
SDC_Changed_Mask	16	15	Intel	1	0	–	0..32767
Timestamp	32	32	Intel	0.001	0	s	0..4.29497e+06

0x140 – BMS_Status

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Pack_Voltage	7	16	Motorola	0.1	0	V	0..6553.5
Pack_Current	23	16	Motorola	0.1	–500	A	–500..6053.5

0x156 – BMS_Temperatures

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Temperature_1	7	8	Motorola	1	-40	degC	-40..215
Temperature_2	15	8	Motorola	1	-40	degC	-40..215
Temperature_3	23	8	Motorola	1	-40	degC	-40..215
Temperature_4	31	8	Motorola	1	-40	degC	-40..215
Temperature_5	39	8	Motorola	1	-40	degC	-40..215
Temperature_6	47	8	Motorola	1	-40	degC	-40..215
Temperature_7	55	8	Motorola	1	-40	degC	-40..215
Temperature_8	63	8	Motorola	1	-40	degC	-40..215

Temperature_1: 255 = Invalid; Temperature_2: 255 = Invalid; Temperature_3: 255 = Invalid; Temperature_4: 255 = Invalid; Temperature_5: 255 = Invalid;
 Temperature_6: 255 = Invalid; Temperature_7: 255 = Invalid; Temperature_8: 255 = Invalid。

0x186040F3 – BMS_Status_Extended

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Pack_Voltage	7	16	Motorola	0.1	0	V	0..6553.5
Pack_Current	23	16	Motorola	0.1	-500	A	-500..6053.5

0x186140F3 – BMS_Extremes_Extended

信号	Start	位数	字节序	Factor	Offset	单位	编码范围
Maximum_Cell_Temperature	39	8	Motorola	1	-40	degC	-40..215

Maximum_Cell_Temperature: 255 = Invalid。

SDC 轮速与车速显示 0x6A2

2026-10-05 的 SDC master 将旧四路 Hz 改为双前轮已标定轮速：标准 Classic CAN 数据帧，DLC 必须为 4，每 20 ms 一帧。字节 0..1 为左前 PA0，字节 2..3 为右前 PA6，均为小端 uint16 mm/s (DBC 0.001 m/s)。SDC 已使用每圈 49 脉冲、轮周长 1414 mm，AMI 不再次标定。

AMI 用双前轮平均轮速作为仪表车速，转换为 km/h 后一次四舍五入： $\text{dash.speed.val} = ((\text{left_raw} + \text{right_raw}) * 9 + 2500) / 5000$ ，全部采用 uint32 整数运算。有效时白色；超过 500 ms 没有合规轮速帧时保留历史值并变灰；新帧自动恢复。其他 SDC 心跳、GPIO、BMS 或 VCU 流量不延长车速有效期。旧 DLC 8、扩展、远程或 FD 帧拒绝。

该源未提供每轮有效标志，0 为有效停止读数，65535 为有效饱和值；不能据此判断传感器健康。SDC 超过 1 秒没有边沿会把对应轮速归零；AMI 仍会把持续收到的零作为有效数值。平均轮速不是 VCU 融合车速，单轮丢脉冲或打滑会影响显示。接收长度变更与旧固件不兼容，需配套 SDC 最新版本。

自定义动力显示输入 0x182

标准数据帧，DLC 必须为 8；CAN1 500K 或 CAN2 250K 均可接收，建议每 100 ms 发送。这是一条 AMI 仪表显示设置接口；扭矩千分以发送者定义的基准扭矩归一化，500 表示 500%。MCU 温度由发送者提供，表示电机控制器 MCU / 逆变器温度，不是 AMI 芯片自动测温。

字节	字段	编码与规则
0..1	扭矩千分	小端 int16，允许 -1000..1000，直接显示
2..3	电机温度	小端 int16 × 0.1 °C，仪表四舍五入到整数 °C
4..5	MCU 温度	小端 int16 × 0.1 °C，仪表四舍五入到整数 °C
6	有效标志	bit0 扭矩、bit1 电机温度、bit2 MCU 温度；1 有效，0 无效；bit3..7 必须为 0
7	帧版本	固定 1，与整体接口 v2.4 版本号独立

有效值更新并显示白色；标志为 0 时立即变灰并保留历史数字。扭矩超出 -1000..1000 时仅扭矩无效，两个温度仍按各自标志处理。温度的 int16 编码范围为 -3276.8..3276.7 °C；显示正负半值均向远离零方向取整。扩展帧、错误 DLC、版本非 1 或保留位非 0 均整帧拒绝，不覆盖数值、不刷新超时。连续 2000 ms 未收到合规帧后，三个值同时变灰；BMS / SDC 流量不能续期。

CAN 工具发送样例：标准 ID 182，DLC 8，数据 F4 01 C2 01 90 01 07 01，周期 100 ms。可用仓库脚本发送：

```
python -m pip install -r scripts/requirements-can.txt
python scripts/send_drivetrain_can.py --torque 500 --motor-temp 45 --mcu-temp 40 --dry-run
# PCAN 当前接 CAN2，速率 250000；其他适配器按 python-can 接口参数设置
python scripts/send_drivetrain_can.py --interface pcan --channel PCAN_USBBUS1 --bitrate 250000 --torque 500 --motor-temp 45 --mcu-temp 40 --duration 10
```

VCU 状态反馈 0x183

新增自定义标准数据帧，DLC 必须为 8，建议周期 100 ms；CAN1 500K 与 CAN2 250K 共用布局。v2.7 扩展此帧为版本 2，只有该帧更新当前任务、锁定、无人状态和 EBS 主页面显示。AMI 发出任务请求后保留 VCU 上次反馈，等待新的实际状态。

字节	字段	编码与规则
0	当前任务	0 无；1 八字绕环；2 直线加速；3 高速循迹；4 车检测试；5 EBS 测试；6 手动驾驶
1	锁定	0 已解锁（红）；1 已锁定（绿）
2	EBS 正常	0 故障（红底）；1 正常（绿底）
3	有效标志	bit0 当前任务，bit1 锁定，bit2 EBS，bit3 无人状态；1 有效，0 未知；bit4..7 为 0
4	反馈计数	0..255，供观察；不去重，不作为请求 ACK
5	无人状态	0 AS OFF；1 AS Ready；2 AS Driving；3 AS Finished；4 AS Emergency；5 Manual Driving
6	保留	必须为 0
7	帧版本	固定 2；旧版本 1 兼容规则见下文

例：02 01 01 0F 00 01 00 02 表示当前直线加速、已锁定、AS Ready、EBS 正常。00 00 00 0F 01 00 00 02 表示无任务、已解锁、AS OFF、EBS 故障。任务 1..6 为绿字，0 无任务为红字。各有效标志为 0 时，对应项立即灰色未知；连续超过 500 ms 无合规帧后四项均未知，其他 CAN 流量不能续期。错误 DLC、版本、枚举值、保留位或扩展/远程/FD 格式整帧拒绝，不改变状态、不续期。该反馈只驱动显示，EBS 安全动作及车辆任务裁决由 VCU 实现。

无人状态来源与兼容

状态语义依据用户提供 ASF 的 AMI 操作界面及 AS 状态描述。ASF 没有分配本帧的数值编码，上表 0..5 是 AMI 与 VCU 的自定义约定，VCU 需显式实现。Manual Driving 用于显示 ASF 的有人驾驶分支，不能由任务 6 自动推断。

主页面在已锁定/已解锁右侧显示独立状态：AS OFF 白色，AS Ready 黄色，AS Driving 黄色，AS Finished 蓝色，AS Emergency 红色，Manual Driving 绿色；未知灰色。文字不闪烁，颜色为仪表约定，不代替 ASSI 灯控制。

旧版本 1 (byte7=01) 仍接受原三项, 要求 byte3 bit3..7=0、byte5=0、byte6=0, 每次收到后无人状态立即置未知; 不能保留上一条版本 2 的 AS 状态。旧 MCU 不能识别版本 2; 新增状态需配套 v2.7 MCU 和包含 as_state 的屏幕工程。800×480 与 480×272 工程均已更新; 小屏 AS=5 显示“有人驾驶”。任务、锁定、EBS 与 AS 独立反馈, AMI 不推断安全状态。

发送工具默认版本 2; 指定 --as-state 时默认有效标志 15, 未指定时为 7 (AS 未知)。--valid 可单独清除各字段有效位; 兼容预览可用 --version 1, 不能同时指定 --as-state。

当前任务发送对照

下表第 1、2 字节均为 01, 并且 byte5=01, 只用于展示已锁定、AS Ready、EBS 正常; VCU 必须填写实际状态。每条合规帧同时更新四项及其有效性, 不能只改任务有效位而期望其他项继续保留。

任务号	当前任务	标准 0x183 / DLC 8 数据
0	无 (红)	00 01 01 0F 00 01 00 02
1	八字绕环	01 01 01 0F 00 01 00 02
2	直线加速	02 01 01 0F 00 01 00 02
3	高速循迹	03 01 01 0F 00 01 00 02
4	车检测试	04 01 01 0F 00 01 00 02
5	EBS 测试	05 01 01 0F 00 01 00 02
6	手动驾驶	06 01 01 0F 00 01 00 02

任务未知通过 byte3 的 bit0=0 编码。例如 00 01 01 0E 00 01 00 02显示任务未知、已锁定、AS Ready、EBS 正常; 不要发送任务号 7, 7 是屏内未知状态而不是 CAN 枚举。任务号、锁定、EBS、无人状态的数值范围始终校验, 有效标志为 0 时也不能发送越界值。

发送顺序: AMI 确认按钮 -> UART 任务帧 -> MCU CAN 0x300 请求 -> VCU 裁决 -> VCU 周期 CAN 0x183 实际状态 -> AMI 显示。确认发送后可继续显示此前实际任务, 直到 VCU 更新或反馈超时。

发送工具预览和实发示例:

```
python scripts/send_vcu_status_can.py --task 2 --locked 1 --ebs-normal 1 --as-state 1 --dry-run
python scripts/send_vcu_status_can.py --interface pcan --channel PCAN_USBBUS1 --bitrate 500000 --task 2 --locked 1 --ebs-normal 1 --as-state 1 --duration 10
# CAN2 改为 --bitrate 250000; Windows 适配器需要时添加 --dll-dir 指定 DLL 目录
```

屏内全局变量: current_task=0..6 (7 未知)、lock_state=0/1/2、ebs_state=0/1/2 (0 未知、1 绿、2 红); as_state=0..5, 6 未知。MCU 更新这些变量, dash 后初始化和 100 ms 定时器更新文案、颜色和任务按钮; 切页保留反馈。

AMI 任务请求语义

Task_ID	任务
1	八字绕环
2	直线加速
3	高速循迹
4	车检测试
5	EBS 测试
6	手动驾驶

用户确认时，屏幕通过串口发送 AA 55 01 NN 0D 0A，MCU 才产生 CAN 0x300 请求；取消不产生请求。Protocol_Version 和 Request_Source 固定为 1；Request_Sequence 从 MCU 启动后的 0 开始，在各目标口完成入队后加一，255 后回到 0。MCU_Uptime 为请求入队时的 HAL_GetTick() 毫秒值，小端 uint32，自然回绕。

优先发送到最近 500 ms 内收到过有效协议报文的 CAN 口；两路活跃则发送两路；没有活跃口时尝试两路。若未连接端口自动重发导致 TX FIFO 填满，现有队列可能等待。该策略沿用当前固件。任务表示操作员请求，VCU 负责裁决和安全互锁；当前实际任务由 0x183 反馈，不按请求立即显示；未定义逐请求 ACK 或运行完成报文，也不把任务号换成 ASSI 灯状态。

显示缩放与失效规则

显示项	输入和显示规则	失效条件
车速	0x6A2 / DLC 4 双前轮 uint16 mm/s 平均, ((左+右)*9+2500)/5000, 整数 km/h	500 ms 无有效轮速帧; 保留数字变灰
低压电池	0x6A1 uint16 mV; lvsbat.val=(mV+5)/10, 显示两位小数 V	1500 ms 无有效 ADC
高压电压	BMS 大端 uint16 × 0.1 V; 现有控件四舍五入到整数 V	2000 ms 无有效 BMS 状态
高压电流	BMS 大端 uint16 × 0.1 – 500 A; 四舍五入到整数 A	2000 ms 无有效 BMS 状态
最高单体温度	0x156 有效通道最大值, 或扩展帧字节 4 – 40	全无效或 2000 ms 无有效温度源
扭矩千分	0x182 小端 int16, 直接更新 dash.mct.val	bit0=0、超范围或 2000 ms 超时
电机温度	0x182 int16 × 0.1 °C, 整数显示 dash.djwd.val	bit1=0 或 2000 ms 超时
MCU 温度	0x182 int16 × 0.1 °C, 整数显示 dash.mcuwd.val	bit2=0 或 2000 ms 超时

电流字段是带偏置的 uint16, 不能按 int16 解码。温度原始 0xFF 为无效; DBC 枚举将其标为 Invalid。关闭枚举解码的工具可能显示 215, 聚合时必须先排除 原始 255, 不能把它当作 215 °C。有效实际温度范围为 -40..214 °C。第 9/10 路虚拟温度没有总线字段, 最高温度也属于 MCU 派生结果, 不新增虚构信号。

BMS 扩展帧的其他字段没有纳入本 DBC, 尤其字节 6..7 的低压电压不覆盖 SDC ADC。两类温度帧共用最高温度显示, 后收到的温度帧更新该显示及有效性。SDC GPIO PD0..PD14 按顺序映射到 d0、d5、d2、d6、d7、d1、d3、d4、d8..d14; 全部低电平正常、高电平故障, d15 无来源保持未知。心跳维持已有采样, 用户最新纠正 PD1 / Safe_S 为左电机互锁 (d5), 屏幕已删除 BSPD/TSMS。心跳和 GPIO 均停止超过 500 ms 后全部未知, 恢复需要新的 GPIO 采样; 当前 0x6A6 每 1000 ms 上报一次, 变化掩码累积期间变化; 心跳继续维持已有采样。

有效值显示白色; 未知、无效、超时显示深灰色, 历史数值保留但不作为有效测量。各源独立超时, 其他流量不续期。每 5 秒重新同步界面, 刚连接需等待首轮同步。

报文样例

ID / DLC	数据（十六进制）	解码及仪表显示
0x300 / 8	01 02 00 01 E8 03 00 00	直线加速请求、序号 0、来源 AMI、上电 1000 ms
0x183 / 8	02 01 01 0F 00 01 00 02	当前直线加速、已锁定、AS Ready、EBS 正常，四项有效
0x182 / 8	F4 01 C2 01 90 01 07 01	500‰、45.0 ℃、40.0 ℃，三个值有效
0x182 / 8	0C FE 9B FF 33 FF 07 01	-500‰、-10.1 ℃、-20.5 ℃；显示 -500 / -10 / -21
0x6A1 / 2	0C 30	ADC 12300 mV；低压 12.30 V
0x6A2 / 4	10 27 F8 2A	左前 10.000 / 右前 11.000 m/s；平均 37.8 km/h，显示 38
0x140 / 8	0F A0 14 03 00 00 00 05	400.0 V / +12.3 A；显示 400 / 12
0x186040F3 / 8 扩展	10 73 10 D1 00 00 00 05	421.1 V / -69.5 A；显示 421 / -70
0x156 / 8	42 FF 43 47 37 0A 50 41	第二路无效，有效最高温度 40 ℃
0x186140F3 / 8 扩展	10 04 0C E4 3F 17 00 7B	最高温度 23 ℃；BMS 低压字段不使用

验证和复现

在仓库根目录运行：

```
python -m pip install -r scripts/requirements.txt
python scripts/verify_dbc.py
python scripts/export_dbc_docs.py --check
# 若同时检出 SDC 仓库，可额外比较五条输入帧的原始布局：
python scripts/verify_dbc.py --sdc-dbc ../SDC_ACQ_DAC/SDC_ACQ_DAC.dbc
```

校验覆盖严格 DBC 解析（长度、信号越界与重叠）、13 个 ID 与固件宏对应、标准/扩展格式、53 个信号、发送/接收节点、SDC / 自定义帧周期、已知报文解码、六个任务号与计数边界编码、大端偏置电流、无效温度枚举、全负温度和截短 DLC。已安装并验证 cantools 44.1.0，接口参考 [cantools 官方文档](#)。

上述是静态及软件编解码验证，另校验两份 DBC 的速率、相同信号布局及源码 / CubeMX 位时序一致。2026-10-07 v2.7 已完成 MCU 烧录及 16672 字节回读一致；真实 CAN1 / MCU / COM6 到大小屏官方模拟器的无人状态与触摸回传验收通过，见 [无人状态报告](#) 与 [小屏报告](#)。v2.2 的烧录回读、新帧 CAN2 测试及串口显示核验状态见 [显示 CAN 测试报告](#)。v2.1 已完成烧录、完整 Flash 回读比对、CAN1 500K 与 CAN2 250K 到官方 HMI 调试器的实物测试，详见 [双 CAN 测试报告](#)。之前 v2.0 的 CAN 盒 -> 实物 AMI CAN1 / MCU -> USART2 -> COM4 -> 官方 HMI 调试器的实际更新和六个任务回传见 [联调报告](#)。物理串口屏、真实 SDC/BMS 节点、两路同时接入和车辆执行动作尚未实测；文档生成和 DBC 解析不替代这些测试。

修改 AMI.dbc 后运行 `python scripts/export_dbc_docs.py` 更新 CAN2 DBC 和本文；固件字节布局、DBC 与此文档需同时保持一致。