ВСТРОЕННОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ КОНТРОЛЛЕРА ЗАРЯДА CCS

Описание функциональных характеристик

ООО «Силовая электроника»

СОДЕРЖАНИЕ

1. Общие сведения

Встроенное программное обеспечение (ПО) контроллера заряда CCS предназначено для управления зарядной сессией в соответствии со стандартами IEC61851, ISO 15118 и DIN 70121 в составе электрозарядной станции.

Встроенное программное обеспечение контроллера заряда CCS обеспечивает:

- прием параметров зарядной сессии и управляющих команд по интерфейсам CAN и Ethernet;
- выдачу информации о ходе и параметрах текущей зарядной сессии по интерфейсам CAN и Ethernet;
- цифровой обмен данными с электромобилем через встроенный PLC модем;
- контроль температуры контактов зарядного разъема по электрическому сопротивлению встроенных в него термодатчиков;
- управления светодиодной индикацией состояния зарядной сессии.

2. Описание программного интерфейса контроллера CCS при взаимодействии по RPC

Интерфейс взаимодействия с потребителем – Ethernet. Программный интерфейс взаимодействия основан на использовании удаленного вызова процедур (RPC) по сетевому протоколу TCP IP v4. Используемый формат сообщений msgpack-rpc. Для данного протокола взаимодействия существуют библиотеки с открытым исходным кодом под разные платформы и языки программирования.

Контроллер CCS реализует одновременно сервер RPC для приема команд от управляющего контроллера и клиент RPC для передачи команд на управляющий контроллер (рисунок 1).

Рисунок 1 – Взаимодействие управляющего контроллера и контроллера CCS

Контроллер сконфигурирован на получение параметров (IP адрес, маска подсети, шлюз), необходимых для работы в сети TCP/IP, по протоколу DHCP при загрузке. Также по протоколу DHCP контроллеру может быть передан адрес NTP сервера для синхронизации времени.

При отсутствии в сети DHCP сервера контроллеру присваивается статический IP адрес 192.168.3.221/24. Порт сервера RPC контроллера CCS: 9000.

Обмен информацией между управляющим контроллером и контроллером CCS осуществляется по схеме Master-Slave, при этом контроллер CCS выступает в роли Slave. Для инициализации соединения клиент RPC управляющего контроллера подключается к серверу RPC контроллера CCS и выполняет метод «rpcConnectRequest»:

string rpcConnectRequest(string interfaceId, string remoteAddress, int remotePort, time_t connectionTimeoutMsec, time_t pingPeriodMsec,uint32_t pingCheckCount);

interfaceId – идентификатор интерфейса RPC, должен быть равен "IID_SECC_CCS_1.1";

remoteAddress – собственный IP адрес управляющего контроллера;

remotePort – порт сервера RPC управляющего контроллера;

connectionTimeoutMsec – таймаут соединения TCP, мсек;

pingPeriodMsec – требуемый период передачи пингов, мсек;

pingCheckCount – количество пропущенных пингов, после которого соединение считается разорванным.

После получения вызова «rpcConnectRequest» клиент RPC контроллера CCS выполняет обратное подключение к серверу RPC управляющего контроллера по указанному IP адресу и порту и начинает с периодичностью pingPeriodMsec вызывать метод «rpcPing» для поддержания соединения:

void rpcPing(int selfConnectionInputState, int selfConnectionOutputState);

selfConnectionInputState – принимаются ли пинги с удаленной стороны (1 – пингов нет, 2 – пинги есть);

selfConnectionOutputState – отправляются ли пинги к удаленной стороне (1 – ошибка отправки пингов, 2 – пинги отправляются).

Клиент RPC управляющего контроллера после установки соединения также должен с периодичностью pingPeriodMsec осуществлять вызов метода «rpcPing» для поддержания соединения и передавать актуальные значения selfConnectionInputState, selfConnectionOutputState.

Если сервер RPC контроллера CCS после установки соединения не получает пинги в течение времени pingPeriodMsec * pingCheckCount или происходит таймаут соединения TCP, то соединение клиента RPC контроллера CCS к серверу RPC управляющего контроллера разрывается и устанавливается заново.

3. Процедуры программного интерфейса контроллера

3.1 Процедуры, исполняемые контроллером (RPC сервер)

- перезагрузка контролера void RESET();
- разрешение зарядной сессии void AUTHORIZE();

- остановка зарядной сессии void USER_STOP();
- установка лимитов станции для зарядной сессии (максимальные выходные мощность, напряжение, ток; минимальные напряжение и ток)

void SET_INVERTOR_LIMITS(float maximumPowerLimitW, float maximumVoltage-LimitV, float maximumCurrentLimitA, float minimumVoltageLimitV, float minimumCurrentLim-itA, float peakCurrentRippleA);

- установка счетчика оставшегося времени заряда; при равенстве 0 заряд прекращается

void SET_CHARGE_TIME_LIMIT(float chargeTimeLimitSec);

- установка текущего статуса силовой части (на силовые преобразователи подается питающее напряжение; подается напряжение на выход зарядной станции; имеется ли любая ошибка в силовых преобразователях; отсутствие обмена информацией с силовыми преобразователями)

void SET_INVERTOR_STATE(bool isPowerOn, bool isInvertorOn, bool isInver-torError, bool isInterfaceError);

- установка текущих параметров зарядной сессии (текущее напряжение и ток на выходе зарядной станции)

void SET_INVERTOR_PRESENT_PARAMS(float presentVoltageV, float presentCur-rentA);

- установка текущих параметров контроля изоляции (активность устройства контроля изоляции; идет самотестирование устройства контроля изоляции; статус изоляции (INVALID, VALID, WARNING, FAULT, NO_IMD))

void SET_ISOLATION_STATE(bool isIsolationMonitoring, bool isImdTest, string isola-tionLevel);

3.2 Процедуры, вызываемые контроллером (rpc клиент)

- передача текущей версии ПО контроллера void SETVERSION(string version);
- передача идентификатора контроллера void SET_SECC_ID(string seccId);
- передача текущей стадии зарядной сессии (DISCONNECTED, CONNECTED, HANDSHAKE, SESSION_SETUP, SERVICE_DISCOVERY, SERVICE_DETAIL, PAYMENT_SERVICE_SELECTION, AUTHORIZATION, CHARGE_PARAMETER_DISCOVERY, CABLE_CHECK, PRECHARGE, POWER_DELIVERY, CURRENT_DEMAND, METERING_RECEIPT, WELDING_DETECTION, SESSION_STOP, ERROR, STOP)

void SET_SECC_CURRENT_STATE(string currentState);

- передача используемого V2G протокола (UNKNOWN, ISO_15118_2_2010, ISO_15118_2_2013, DIN_70121_2012)

void SET_V2G_PROTOCOL(string protocol);

- передача максимальных зарядных лимитов электромобиля

void SET_EV_LIMITS(float maximumPowerLimitW, float maximumVoltageLimitV, float maximumCurrentLimitA);

- управление силовыми преобразователями (режим работы силовых преобразователей (3 = выключены, 1 = режим ожидания, 2 = включены, на выход подается напряжение), требуемое напряжение и ток);

void SET_EV_TARGET_PARAMS(uint32_t To_Inv_State, float targetVoltageV, float targetCurrentA);

- передача параметров электромобиля

void SET_EV_PARAMS(string evId, float departureTime, float energyCapacity, float energyRequest);

- передача готовности и ошибок электромобиля (NO_ERROR, RESS_TEMPERATURE_INHIBIT, EV_SHIFT_POSITION, CHARGE_CONNECTOR_LOCK_FAULT, EV_RESS_MALFUNCTION, CHARGING_CURRENT_DIFFERENTIAL, CHARGING_VOLTAGE_OUT_OF_RANGE, CHARGING_SYSTEM_INCOMPATIBILITY, UNKNOWN_ERROR)

void SET_EV_STATE(bool evReady, string evErrorCode);

- передача состояния электромобиля

void SET_EV_SOC(float evSoc, bool bulkChargingComplete, bool chargingComplete, float bulkSoc, float fullSoc, float remainingTimeToBulkSocSec, float remainingTimeToFullSocSec);

- передача состояния линии CP (состояние линии CP: STATE_A, STATE_B1, STATE_B2, STATE_C, STATE_F, измеренные значения напряжения на линии CP). Функция вызывается при изменении состояния линии CP или при изменении напряжения на 0.2В

void SET_CP_STATE(string cpState, float cpVoltagePos, float cpVoltageNeg);

- передача ошибок контроллера (на данный момент не реализовано)

void SET_ERROR_CODE(uint32_t errorCode, string errorCode_str);

Процедуры вызываются контроллером при каждом изменении одного из параметров.

4. Последовательность обмена сообщениями с контроллером

После установки/переподключения RPC соединения с контроллером со стороны контроллера вызываются следующие функции для передачи актуальных параметров:

SET_EV_LIMITS;

SET_EV_TARGET_PARAMS;

SET_EV_PARAMS;

SET_EV_STATE;

SET_EV_SOC;

SET_SECC_CURRENT_STATE;

SET_SECC_ID;

SET_ERROR_CODE;

SET_V2G_PROTOCOL;

SETVERSION.

В дальнейшем при установленном RPC соединении при изменении одного из параметров контроллер вызывает соответствующую функцию для передачи значений зарядной станции.

Для обновления текущих параметров со стороны зарядной станции после установки/переподключения RPC соединения рекомендуется вызвать следующие функции:

SET_INVERTOR_LIMITS;

SET_CHARGE_TIME_LIMIT;

SET_INVERTOR_STATE;

SET_INVERTOR_PRESENT_PARAMS;

SET_ISOLATION_STATE.

При установленном RPC соединении зарядная станция должна передавать текущие параметры путем вызова соответствующих функций или при изменении одного из параметров или периодически. При разрыве RPC соединения во время зарядной сессии, контроллер аварийно завершает сессию.

5. Последовательность проведения зарядной сессии

После инициализации контроллер находится в состоянии DISCONNECTED (SET_SECC_CURRENT_STATE).

Для разрешения проведения зарядной сессии станция должна вызвать функцию

AUTHORIZE, после чего контроллер перейдет в состояние ожидания подключения по линии CP.

После обнаружения подключения по линии CP (B1) контроллер включает ШИМ, переводит линию CP в состояние B2 и переходит в состояние ожидания рукопожатия с электромобилем HANDSHAKE.

После установки цифровой связи с электромобилем контроллер последовательно проходит стадии SESSION_SETUP, SERVICE_DISCOVERY, SERVICE_DETAIL, PAYMENT_SERVICE_SELECTION, AUTHORIZATION, CHARGE_PARAMETER_DISCOVERY.

К началу стадии CHARGE_PARAMETER_DISCOVERY контроллер должен иметь актуальные значения параметров от зарядной станции, установленные с помощью функции SET_INVERTOR_LIMITS.

На следующей стадии проводится проверка изоляции со стороны зарядной станции. Контроллер переходит в состояние CABLE_CHECK. Контроллер вызывает функцию SET_EV_TARGET_PARAMS для включения силовых преобразователей. Зарядная станция должна передавать текущие выходные параметры с помощью функций SET_INVERTOR_PRESENT_PARAMS, SET_INVERTOR_STATE. Минимальная длительность стадии проверки изоляции 10 секунд. По истечении этого времени анализируется флаг isImdTest. Контроллер перейдет к следующей стадии при равенстве этого флага FALSE. При равенстве значения TRU контроллер будет продолжать стадию проверки изоляции.

Если в процессе контроля изоляции станция обнаружит недопустимое значение сопротивления изоляции, она должна выставить флаг isInvertorError с помощью функции SET_INVERTOR_STATE. При этом контроллер прекратит зарядную сессию.

Значение isolationLevel, передаваемое функцией SET_ISOLATION_STATE, контроллер передает на электромобиль, не анализируя.

После успешного прохождения теста изоляции контроллер переходит в состояние PRECHARGE, происходит выравнивание напряжения на выходе зарядной станции с напряжением на АКБ электромобиля, после чего контроллер переходит в состояние CURRENT_DEMAND.

На стадии CURRENT_DEMAND происходит процесс зарядки электромобиля. Зарядная станция должна передавать оставшееся время зарядной сессии с помощью функции SET_CHARGE_TIME_LIMIT. При равенстве значения chargeTimeLimitSec нулю зарядная сессия будет прекращена. Допускается передавать любое значение, не равное 0, для поддержания зарядной сессии. Функцию нужно вызвать до перехода на стадию CURRENT_DEMAND.

Нормальное прекращение зарядной сессии возможно либо по инициативе электромобиля, либо по команде USER_STOP с зарядной станции. При нормальном завершении зарядной сессии контроллер переходит в состояние WELDING_DETECTION.

На стадии WELDING_DETECTION ожидается снижение напряжение на выходе зарядной станции ниже порогового уровня, затем контроллер переходит в состояние SESSION_STOP, после чего выключается ШИМ на линии CP (состояние B1) и контроллер переходит в начальное состояние (DISCONNECTED).

Завершение зарядной сессии с помощью функции USER_STOP допустимо на любой стадии зарядной сессии.

В ходе зарядной сессии контроллер проверяет состояние линии CP и цифровую связь с электромобилем. При возникновении ошибок осуществляется аварийное завершение зарядной сессии, контроллер переходит в состояние ERROR. Через несколько секунд происходит переход контроллера в начальное состояние (DISCONNECTED).

6. Требования к аппаратному обеспечению

Минимальные требования к аппаратному обеспечению для функционирования встроенного программного обеспечения контроллера CCS:

- процессорное ядро ARM Cortex-A7;
- тактовая частота не менее 500 МГц;
- объем Flash памяти – 256 МБ;
- объем ОЗУ – 256 МБ.