

Глава 4 Ввод в действие

Введение

В данной главе описана функциональность коммуникационного процессора CP 342S-CAN для шины SPEED-Bus производства VIPA. Модуль можно использовать только на шине SPEED-Bus, в одном из слотов слева от центрального процессора (CPU).

Содержание

Раздел	Страница
Глава 4 Ввод в действие	4-1
Сведения о CANopen.....	4-2
Адрессация на шине SPEED-Bus	4-4
Разработка проекта	4-5
Режимы работы.....	4-14
Образ данных процесса	4-15
Структура сообщения.....	4-17
Каталог объектов	4-22
Диагностика	4-44
Чтение SZL	4-50
(Де-)Активация станции.....	Ошибка! Закладка не определена.

Сведения о CANopen

Введение

CANopen (**C**ontrol **A**rea **N**etwork) – это международный стандарт открытых полевых шин, предназначенных для строительства, производства и автоматизации технологических процессов. Первоначально он был разработан для автомобилестроения.

Благодаря расширенным возможностям обнаружения ошибок, шина CAN признана наиболее безопасной. Вероятность появления нераспознанной ошибки оценивается менее, чем в 4.7×10^{-11} . Сбойные сообщения помечаются и передаются повторно.

В отличие от шин Profibus и Interbus, для CAN определён седьмой уровень взаимодействия «CAN-level-7-protocol» (CAN=CAN application layer), описывающий различные пользовательские профили уровня 7 для шины CAN. Один стандартный пользовательский профиль определён организацией CiA (**C**AN in **A**utomation) и называется CANopen.

Профиль CANopen

Профиль CANopen – это пользовательский профиль для промышленных приложений реального времени, который поддерживается большим числом производителей. CANopen опубликован организацией CAN in Automation association (CiA) под названием DS-301. Спецификация DS-301 определяет стандарты для устройств CAN. Эта спецификация имеет целью сделать оборудование различных производителей взаимозаменяемым. Понятие совместимости оборудования в дальнейшем было развито в спецификации DS-401, которая определяет стандарты на технические данные оборудования и данные процесса. Спецификация DS-401 содержит стандарты на дискретные и аналоговые модули ввода-вывода.

Профиль CANopen - это коммуникационный профиль, определяющий объекты, используемые для передачи данных, а также профили устройств, в которых описаны типы данных для передачи посредством объектов.

Профиль CANopen базируется на каталоге объектов и похож на профиль, используемый для Profibus. Профиль DS-301 определяет два стандартных объекта и несколько специальных объектов:

- Объекты данных процесса (Process data objects - PDO)
PDO используются для передачи данных в реальном времени
- Объекты сервисных данных (Service data objects - SDO)
SDO предоставляют доступ к каталогу объектов для чтения и записи

Носитель данных	Шина CAN имеет линейную топологию. Для построения сети Вы можете применять маршрутизаторы. Количество устройств в сети ограничивается только производительностью шинных модулей. Максимальная длина сети определяется скоростью линии. При скорости 1 МБод (Мбит/сек) длина ветви сети до 40 м, а при скорости 80 кБод (кбит/сек) расстояние ограничено 1000 м. Носитель информации для шины CAN представляет собой экранированный 3-жильный кабель (возможен 5-жильный). Сигналы по шине CAN передаются разностными уровнями напряжений, поэтому шина CAN менее чувствительна к внешним помехам, чем чистый потенциальный или токовый сигнал. С обеих сторон отрезок сети должен заканчиваться терминаторами – резисторами номиналом около 120Ω. Модуль VIPA для шины CAN имеет 9-контактный разъём. При помощи этого разъёма модуль подключается непосредственно к сети CAN. Все устройства сети должны иметь одинаковую скорость обмена. Благодаря шинной топологии можно подключать и отключать устройства без разрыва линии. Это также делает систему более ремонтнопригодной. Расширение сети никак не влияет на работающее оборудование. Неисправные или новые станции распознаются автоматически.
Метод доступа к шине	Методы доступа к шине, в основном, бывают двух типов: контролируемый (детерминированный) и неконтролируемый (случайный). Для шины CAN применяется метод множественного доступа с контролем несущей частоты (Carrier-Sense Multiple Access - CSMA). Все станции имеют одинаковые права доступа к шине, пока шина свободна (случайный доступ к сети). Обмен данными базируется на идентификаторах сообщений, а не идентификаторах станций. Каждое сообщение содержит уникальный идентификатор, в состав которого входит приоритет сообщения. В любой момент времени только одна станция может занять шину для передачи сообщения. Управление доступом к шине CAN производится по алгоритму контроля передаваемых битов (бит-арбитраж - bit-based arbitration), что обеспечивает отсутствие коллизий в линии. Под отсутствием коллизий подразумевается, что станция, выигравшая бит-арбитраж, не нуждается в повторной пересылке сообщения. Если несколько станций пытаются занять шину одновременно, то автоматически выбирается станция с наивысшим приоритетом сообщения. Станция, готовая к передаче, обязательно ждёт, когда шина будет свободна.

Адрессация на шине SPEED-Bus

Вступление

Для доступа к периферийным модулям, в модуле CPU должны быть выделены соответствующие адреса. При отсутствии конфигурации, модуль CPU, во время загрузки, автоматически назначает адреса ввода-вывода периферийным модулям, в зависимости от места их установки. Адреса получают и модули на шине SPEED-Bus.

Максимальное количество подключаемых модулей

Конфигуратор аппаратуры фирмы Siemens позволят устанавливать на шину не более 8 модулей. Модули CPU серии SPEED7 могут управлять 32 модулями на стандартной шине и дополнительно 10 модулями на шине SPEED-Bus. Встроенные коммуникационные процессора (Ethernet и Profibus), конфигурируемые как виртуальные модули на стандартной шине, входят в общее число модулей на стандартной шине.

Если в проекте более 8 модулей на стандартной шине, то необходимо указывать в конфигурации дополнительные **виртуальные** стойки расширения, подключенные к главной. Для этого в конфигурации аппаратуры нужно установить модуль IM 360 фирмы Siemens в слот 3 главной стойки. После этого можно добавить до 3 дополнительных профильных рельсов, каждый из которых должен содержать модуль IM 361 фирмы Siemens в слоте 3.

Задание адресов в аппаратной конфигурации

Доступ к модулям на чтение или запись осуществляется через периферийные адреса или образ процесса.

Адреса можно задавать в конфигураторе аппаратуры через виртуальный Profibus. Для этого необходимо подключить файл SPEEDBUS.GSD. Адреса указываются в диалоге свойств модулей можно.

Автоматическое распределение адресов

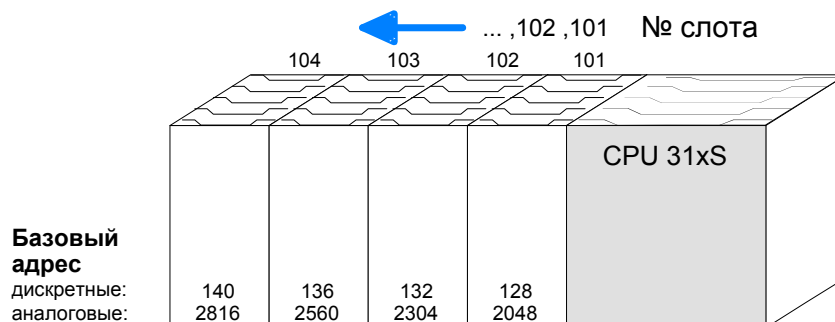
Если Вам не нравится конфигуратор аппаратуры, полезным оказывается автоматическое распределение адресов.

Для дискретных модулей (DIO) адреса выделяются в зависимости от места с шагом 4 байта, для аналоговых (AIO), функциональных (FM) и коммуникационных (CP) модулей – с шагом 256 байтов.

В зависимости от номера слота, базовый адрес модуля вычисляется по формулам:

DIO: $\text{Базовый адрес} = 4 \cdot (\text{№ слота} - 101) + 128$

AIO, FM, CP: $\text{Базовый адрес} = 256 \cdot (\text{№ слота} - 101) + 2048$



Разработка проекта

Введение

Разработка проекта для коммуникационного процессора CANopen на шину SPEED-Bus ведётся при помощи программы WinCoCT (**Windows CANopen Configuration Tool**) фирмы VIPA. Проект экспортируется из WinCoCT как файл wld, который затем можно импортировать в конфигуратор аппаратуры фирмы Siemens.

Для подключения модуля CAN к CPU серии SPEED7, нужно указать его в конфигурации оборудования как ведомое устройство DP типа VIPA_SPEEDBUS виртуальной сети PROFIBUS.

Краткий обзор

Для использования в проекте модулей серии 300S, в т.ч. ведущего модуля шины CAN на SPEED-Bus, нужно добавить модули серии 300S в каталог аппаратуры при помощи GSD-файла фирмы VIPA. Разработка проекта включает в себя следующие шаги:

- Запустите WinCoCT и создать проект сети CANopen.
- Создайте группу кнопкой  и установите в неё ведущее устройство CANopen на SPEED-Bus кнопкой . Имейте в виду, что номер узла невозможно будет изменить позже.
- Активируйте ведущее устройство в диалоге **Node** > *CANopen Manager* > "Device is NMT Master" и подтвердите кнопкой [Close].
- Задайте такие параметры, как средства диагностики и область адресов CPU в разделе **Node** > *PLC Parameters*".
- Создайте группу кнопкой  и добавьте в неё подчинённые устройства CANopen при помощи кнопки .
- Добавьте модули к вашим подчинённым устройствам при помощи кнопки "Modules" и опишите их параметры, если необходимо.
- Подключите данные процесса в таблице "Connections" и уточните их расположение в образе процесса ведущего устройства, если необходимо.
- Сохраните проект и экспортируйте его как wld-файл, выбрав **File** > *Export*.
- Переключитесь в программу SIMATIC manager фирмы Siemens и скопируйте блоки данных CAN из wld-файла в каталог блоков.
- Запустите конфигуратор аппаратуры фирмы Siemens и импортируйте файл SPEEDBUS.GSD для изделий серии SPEED7 фирмы VIPA.
- Начните проект для модуля CPU 318-2DP (6ES7 318-2AJ00-0AB0/V3.0) фирмы Siemens.
- Начиная со слота 4, установите модули серии 300 на стандартную шину в порядке подключения.
- Шина SPEED-Bus параметрируется как виртуальная сеть Profibus, для чего в конфигурации указывается модуль CP 342-5 (342-5DA02 V5.0), в режиме ведущего устройства, на последнем месте. К этой виртуальной сети подключаются модули SPEED-Bus, как ведомые устройства типа VIPA_SPEEDBUS. Номер узла сети соответствует номеру слота, начиная с № 100 для CPU. В слот 0 каждого ведомого устройства добавьте необходимый модуль и опишите его параметры, если необходимо.

Начальные сведения

Конфигуратор аппаратуры – это часть пакета «SIMATIC Manager» фирмы Siemens. В нём ведётся разработка проекта. Модули, используемые для параметрирования, расположены в каталоге аппаратуры.

Для ввода в действие модулей серии 300S необходимо включить описание модулей в каталог аппаратуры. Это делается при помощи GSD-файла SPEEDBUS.GSD фирмы VIPA.

**Внимание!**

Для разработки проекта необходимы знания программ SIMATIC Manager и конфигуратора аппаратуры фирмы Siemens!

Подключение GSD-файла

- Скопируйте поставляемый фирмой VIPA GSD-файл SPEEDBUS.GSD в каталог ...\\siemens\\step7\\s7data\\gsd
- Запустите конфигуратор оборудования фирмы Siemens
- Закройте все проекты
- Выберите пункт **Options** > *Install new GSD-file*
- Найдите и выберите файл "SPEEDBUS.GSD"

После этого модули серии 300S фирмы VIPA появятся в каталоге аппаратуры, и их можно будет использовать в проектах.

WinCoCT

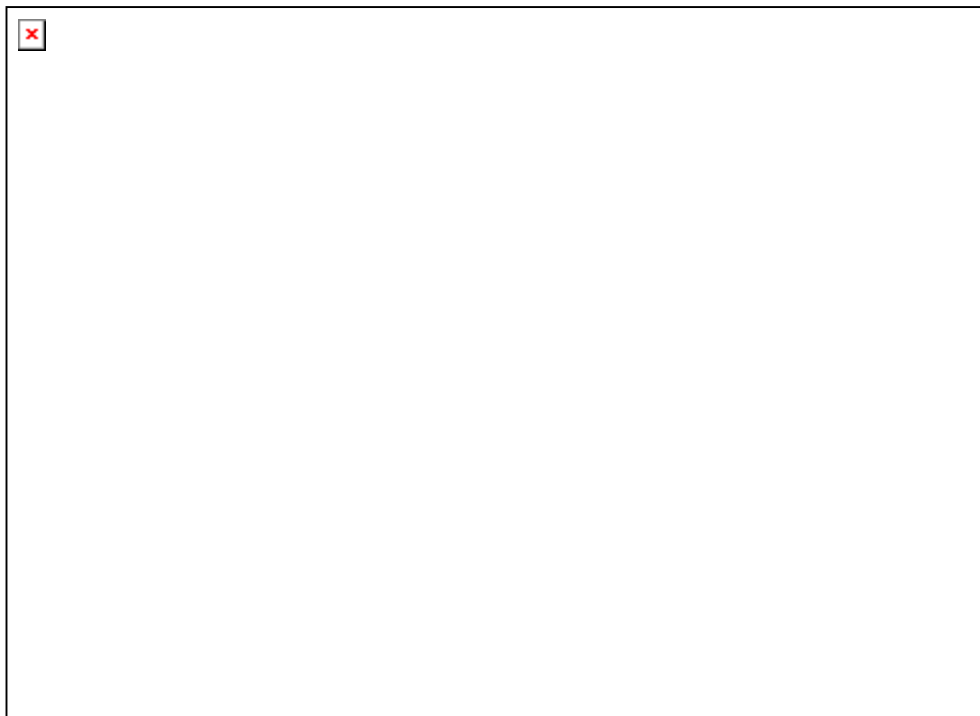
WinCoCT (**W**indows **CAN**open **C**onfiguration **T**ool) – это инструмент для проектирования сетей CANopen, разработанный фирмой VIPA.

WinCoCT отображает топологию сетей CANopen в удобной графической форме. Он позволяет устанавливать, настраивать, группировать полевые устройства, а также настраивать соединения.

Устройства добавляются в проект из списка, который можно расширять при помощи EDS-файла (**E**lectronic **D**ata **S**heet) в любое время.

Нажатием правой кнопки мыши на устройстве открывается контекстное меню, состоящее частично из постоянных и частично из меняющихся пунктов.

Обмен данными (PDO) настраивается в таблице, в которой TxPDO представлены как строки и RxPDO как столбцы.

**Установка параметров проекта**

В диалоге **Tools** > *Project options* вы можете задать параметры сети CAN, такие как скорость, выбор ведущего устройства и т.д.

Более подробную информацию можно найти в руководстве на WinCoCT.

Параметры модуля ведущего устройства CAN на SPEED-Bus

Программа WinCoCT позволяет задавать параметры для модуля ведущего устройства CAN фирмы VIPA на SPEED-Bus. При нажатии правой кнопкой мыши на устройстве вызывается диалог «PLC-Parameters»:



Slot number (номер слота)	Номер слота на шине 101 ... 110: Адрес модуля на шине SPEED-Bus, значение 101 соответствует маркировке «SLOT 1» на шине SPEED-Bus
CANopen DeviceProfileNumber (номер профиля)	Всегда 0x195
Behavior at PLC-STOP (Поведение при остановке PLC)	Здесь задаётся поведение выходных сигналов при переходе CPU в режим STOP. Возможны следующие значения: <i>Switch substitute value 0</i>: Установить в значение 0. Подчинённые устройства в рабочем состоянии. <i>Keep last value</i>: Сохранять последнее значение. Подчинённые устройства в рабочем состоянии. <i>Pre-operational</i>: Подчинённые устройства переводятся в пред-рабочее состояние (выводятся из рабочего). При переходе модуля CPU из режима STOP в режим RUN все подчинённые устройства переводятся в рабочее состояние. <i>Pre-operational + switch substitute value</i>: Все выходы устанавливаются в 0, затем каждое подчинённое устройство, указанное в конфигурации, переводится в пред-рабочее состояние. При переходе модуля CPU из режима STOP в режим RUN все подчинённые устройства переводятся в рабочее состояние.
Diagnostic (Диагностика)	В этом разделе определяется реакция на появление диагностических событий. <i>Diagnostic</i>: Активирует диагностические функции
NMT-Slave (как ведомое)	<i>CANopen state (состояние модуля CANopen)</i>: Если выбрано, модуль CAN может передать своё состояние "preoperational" (предрабочее) или "operational" (рабочее) в модуль CPU. Это состояние можно запросить при помощи SFC 13.
NMT-Master (как ведущее)	<i>Slave failure/recovery (сбой/восстановления подчинённых устройств)</i>: Если выбрано, то блок OB 86 вызывается в CPU в случаях сбоев или перезапусков подчинённых устройств. <i>Error control (наблюдение за ошибками)</i>: Если выбрано, то ведущий модуль CAN посылает всю информацию об ошибках в модуль CPU, где вызывается прерывание OB 82. <i>Emergency Telegram (телеграмма об аварии)</i>: Если выбрано, ведущий модуль CAN посылает все телеграммы о неисправностях подчинённых устройств в модуль CPU, где вызывается прерывание OB 82.
Диапазон адресов в CPU	Следующие поля позволяют задавать диапазоны периферийных адресов модуля CANopen в CPU. Блок информации ввода или вывода состоит из 4 байтов. <i>Input addr. 6000, Input blocks (Адреса входов 6000, входные блоки)</i> Базовый периферийный адрес PI входных данных CAN в CPU, с 6000h, до 80 входных блоков (320 байтов). <i>Output addr. 6000, Output blocks (Адреса выходов 6000, выходн. блоки)</i> Базовый периферийный адрес PIQ выходных данных CAN в CPU, с 6000h, до 80 выходных блоков (320 байтов). <i>Input addr. A000, Input blocks (Адреса входов 6000, входные блоки)</i>

Базовый периферийный адрес PII входных сетевых переменных CAN в CPU, с A000h, до 80 входных блоков (320 байтов).

Output addr. A000, Output blocks (Адреса выходов A000, выходн. блоки)

Базовый периферийный адрес PIQ выходных сетевых переменных CAN в CPU, с 6000h, до 80 выходных блоков (320 байтов).

Manufacturer
Specific Interrupt
(OB 57)
(специфическое
прерывание
производителя)

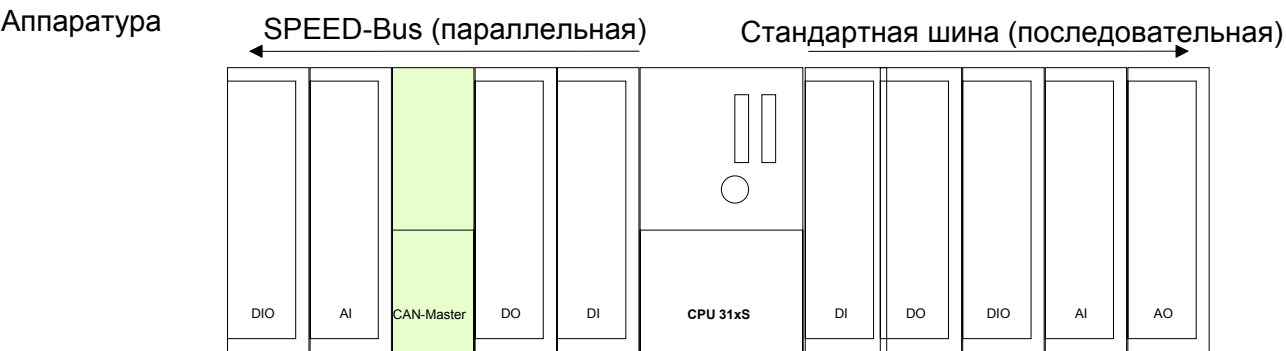
Activate (активировать): Разрешает использование специфического прерывания производителя OB 57.

Number of Messages (количество сообщений): Количество сообщений, которые должны быть приняты для генерации прерывания OB 57.

Дополнительно нужно проинициализировать индекс 2000h в каталоге CANopen.

- Пошаговая
инструкция по
разработке
проекта
- Далее приведена пошаговая инструкция по разработке проекта. Разработка проекта ведётся в 4 этапа:

 - Разработка проекта сети CAN в WinCoCT и экспорт его в виде wld-файла
 - Импорт проекта сети CAN
 - Проектирование модулей на стандартной шине
 - Проектирование шины SPEED-Bus, как виртуальной сети Profibus, и модулей на шину SPEED-Bus. Необходим файл SPEEDBUS.GSD.



Начальные
условия

Для проектирования сети CANopen, необходимо скопировать в каталог EDS программы WinCoCT обновлённый EDS-файл.

Для ввода в действие модулей серии 300S необходимо добавить их в каталог аппаратуры при помощи GSD-файла SPEEDBUS.GSD фирмы VIPA.

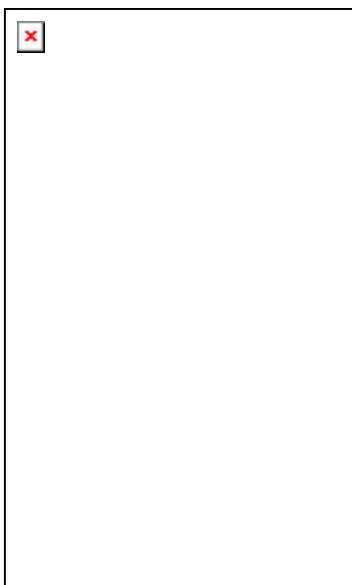
Проектирование сети CAN в WinCoCT



- Скопируйте необходимые EDS-файлы в EDS-каталог и запустите WinCoCT.
- Создайте группу "master" кнопкой  и добавьте ведущее устройство CANopen кнопкой .
- Создайте группу "slave" кнопкой  и добавьте подчинённые устройства CANopen кнопкой .
- Правой кнопкой мыши нажмите на нужные подчинённые устройства и добавьте необходимые модули через диалог "Modules".
- Задайте параметры модулям в диалоге [Parameter], либо через соответствующий каталог объектов.



- Правой кнопкой мыши нажмите на ведущем устройстве, откроется диалог "Set PLC Parameters". Здесь задаются диагностические свойства и адресные пространства в CPU. В пункте "Slot number" (номер слота) введите № слота модуля CAN на шине SPEED-Bus + 100 (101...110). При экспортировании проекта, WinCoCT создаёт соответствующие блоки данных DB, добавляя к номеру слота 2000.



- Перейдите в таблицу "Connections" (подключения) в главном окне. Здесь данные процесса показаны в виде матрицы как входа (1-ая колонка) и как выхода (1-ая строка). Для просмотра данных соответствующего устройства нажмите на знак "+" возле него.
- Для удобства, разрешённые соединения помечены зелёным цветом в перекрестье. Выберите соответствующую ячейку мышкой и нажмите на неё. → теперь можно описать соединение в соответствующем окне PDO. Соединение можно проверить, переключившись в окно "Layout", нажав мышкой на ведущее устройство и просмотрев его диаграмму "Process Picture".
- Сохраните проект.
- При помощи **File > Export** экспортируйте проект сети CANopen в wld-файл. Имя файла будет состоять из имени проекта + номер узла + идентификатор ID **Master/Slave**.
- Из полученного wld-файла необходимые блоки данных могут быть перенесены в программу ПЛК. Подробнее об этом – на следующей странице.

На этом проектирование сети CANopen в WinCoCT окончено.

Перенос конфигурации в программу ПЛК

- Запустите SIMATIC Manager фирмы Siemens с новым проектом. Откройте конфигуратор аппаратуры, установите профильный рельс из каталога.
- Добавьте следующий модуль ЦПУ фирмы Siemens в слот 2: CPU 318-2DP (6ES7 318-2AJ00-0AB0/V3.0)
- Откройте wld-файл в диалоге **File > Memory Card File > open**
- Скопируйте блок DB 2xxx в каталог программных блоков

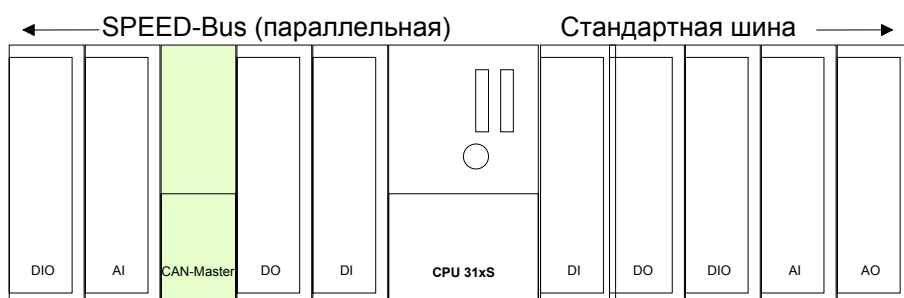
После того, как вы загрузите блоки в модуль SPEED7-CPU, они опознаются модулем CPU, и параметры, содержащиеся в них, передаются в модуль CAN.

Это происходит, если модуль CAN указан в конфигурации шины SPEED-Bus. Далее приведена пошаговая инструкция для конфигурирования.

Проектирование для модулей на стандартную шину

Модули, расположенные справа от модуля CPU, на стандартной шине, конфигурируются следующим образом:

- Запустите конфигуратор оборудования фирмы Siemens в новом проекте и установите профильный рельс из каталога аппаратуры.
- Установите следующий модуль центрального процессора в слот №2: CPU 318-2DP (6ES7 318-2AJ00-0AB0/V3.0)
- Установите модули серии 300 на шину в порядке подключения, начиная со слота 4.
- Задайте для модулей необходимые параметры. Диалог изменения параметров открывается после двойного нажатия на модуле.
- Для расширения шины можете использовать модули IM 360 фирмы Siemens. С их помощью можно добавить до 3 виртуальных корзин расширения с модулем IM 361 в каждой. Модули расширения шины добавляются всегда в слот 3.
- Сохраните проект



Standard bus

Slot	Module
1	
2	CPU 318-2
X2	DP
X1	MPI/DP
3	
4	DI
5	DO
6	DIO
7	AI
8	AO

Разработка проекта
шины SPEED-Bus
как виртуальной
сети Profibus

При разработке проекта шина SPEED-Bus представляется как виртуальная шина с ведущим устройством Profibus DP. Для этого последним модулем на стандартной шине в проекте нужно указать ведущее устройство Profibus DP (342-5DA02 V5.0).

Для проектирования систем на базе модулей 300S необходимо включить модули серии 300S в каталог аппаратуры, установив (импортировав) файл SPEEDBUS.GSD от VIPA.

После импортирования файла SPEEDBUS.GSD модули серии System 300S от VIPA будут включены в каталог аппаратуры: *Profibus-DP / Additional field devices / I/O / VIPA_SPEEDBUS*.

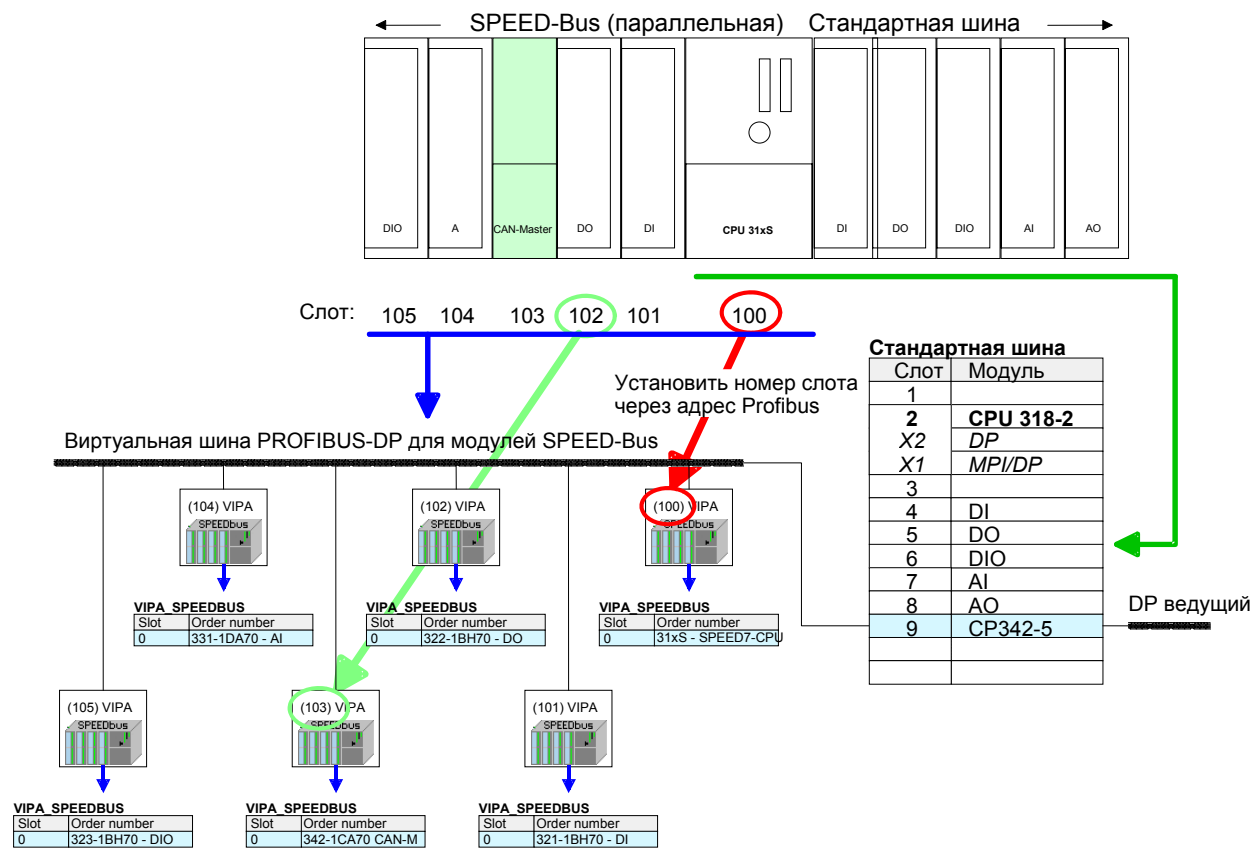
Теперь для модуля CPU и каждого модуля шины SPEED-Bus установите на виртуальный интерфейс PROFIBUS ведомые устройства "VIPA_SPEEDBUS".

В качестве номера слота укажите для них адреса PROFIBUS 100...110 и установите в каждый из них, в слот 0, соответствующий модуль из подкаталога аппаратуры VIPA_SPEEDBUS.



Внимание!

Будьте внимательны, не задавайте неоднозначные адреса DP при проектировании шины SPEED-Bus! Конфигуратор аппаратуры фирмы Siemens не проверяет адреса внешних устройств DP!

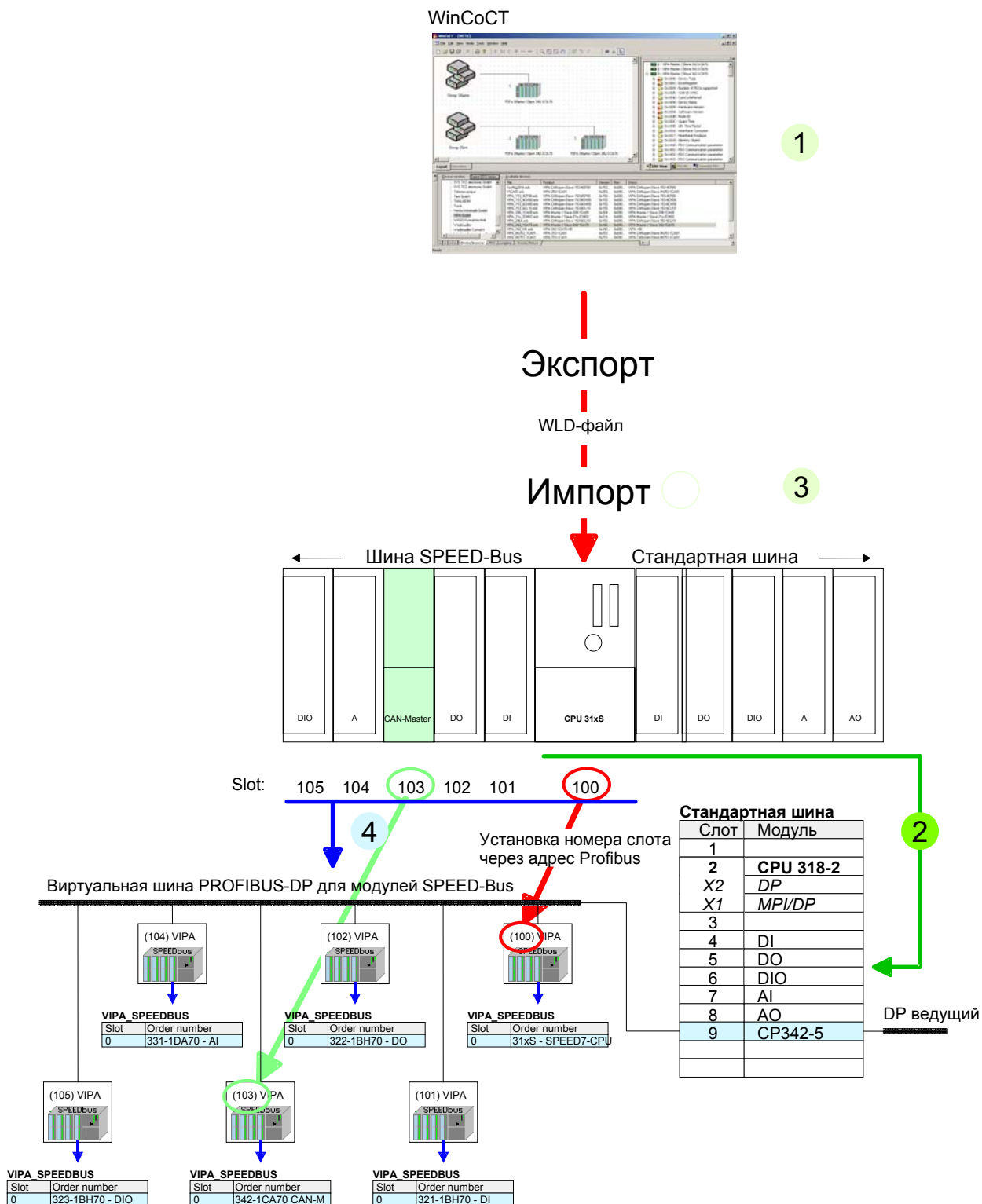


Необходимый модуль устанавливается из подкаталога аппаратуры VIPA_SPEEDBUS в слот 0.

При загрузке аппаратной конфигурации в модуль CPU, необходимые параметры передаются также и в модуль CAN.

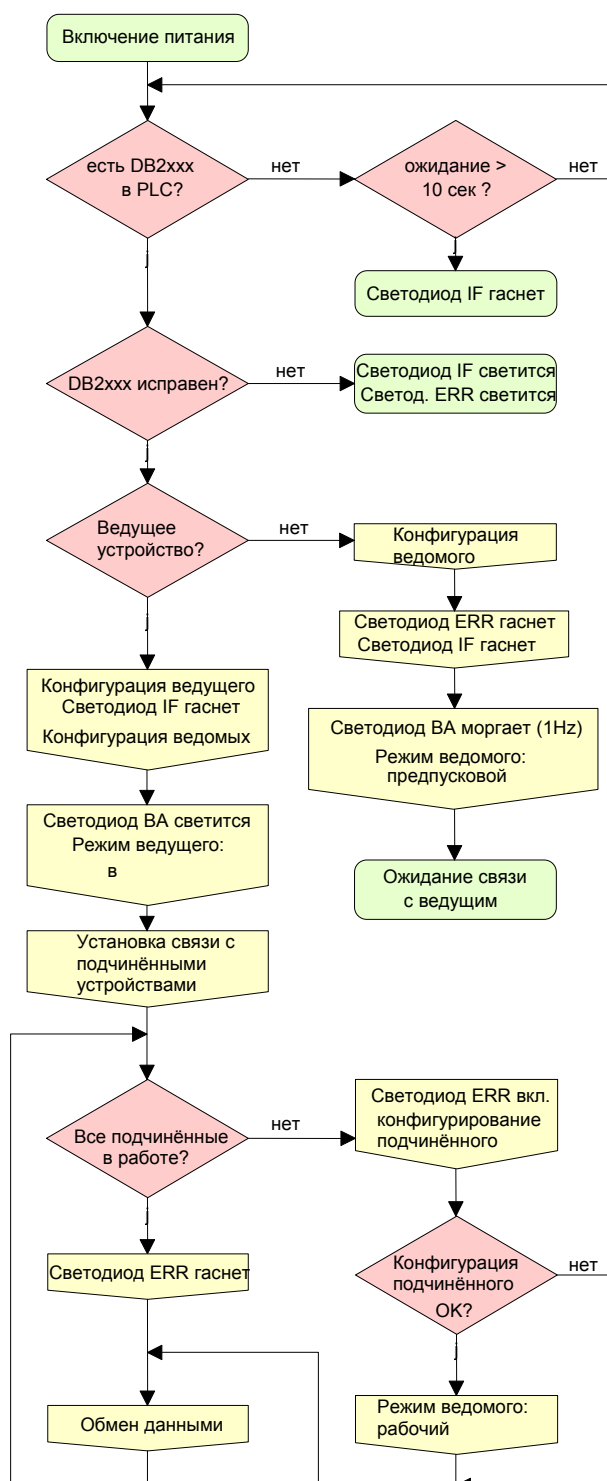
Резюме

Следующий рисунок иллюстрирует все шаги по разработке проекта:



Необходимый модуль устанавливается из подкаталога аппаратуры VIPA_SPEEDBUS в слот 0.

Режимы работы



Переход в рабочее состояние «STOP → RUN» (автоматически)

После подачи питания и при наличии действительного проекта в модуле CPU, ведущее устройство автоматически переключается в режим RUN. Модуль не имеет переключателя режимов работы.

После подачи питания конфигурация автоматически передаётся в модуль CAN из CPU. После этого устанавливается связь с подчинёнными устройствами.

Если есть активные связи и параметры шины верны, ведущее устройство CAN переходит в режим "работа". Индикаторы RUN и BA зажигаются.

Если параметры модуля CAN заданы неверно, он остаётся в состоянии STOP и индицирует ошибку параметрирования индикатором IF.

Рабочее состояние «RUN»

В рабочем состоянии (RUN), светодиодные индикаторы RUN и BA включены. В этом состоянии возможен обмен данными.

В состоянии ошибки (например, сбой подчинённого устройства) индикатор ERR включен и в CPU передаётся сигнал аварии.

Образ данных процесса

Образ данных процесса состоит из следующих частей:

- Образ входных данных (PII) для объектов RPDO
- Образ выходных данных (PIQ) для объектов TPDO

Каждая часть состоит из 320 байтов цифровых данных ("Digital Data") и 320 байтов сетевых переменных ("Network Variables").

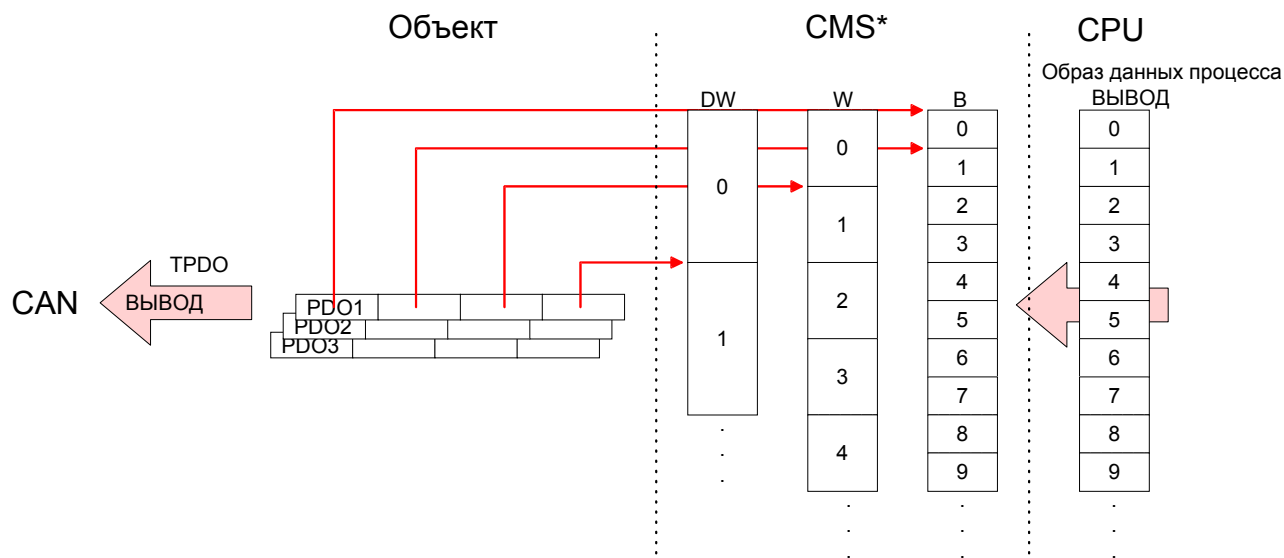
Входные данные Для входных данных применяются следующие типы объектов:

- цифровые входа 8 бит (объект 0x6000)
- цифровые входа 16 бит (объект 0x6100)
- цифровые входа 32 бит (объект 0x6120)
- входные сетевые переменные 8 бит (объект 0xA040)
- входные сетевые переменные 16 бит (объект 0xA100)
- входные сетевые переменные 32 бит (объект 0xA200)
- входные сетевые переменные 64 бит (объект 0xA440)

Как видно из следующей иллюстрации, объекты цифровых входов могут делить между собой пересекающиеся области памяти в CPU.

Например, объект с индексом 0x6000, с подиндексом 2 пересекается с объектом 0x6100, с подиндексом 1. Они оба занимают одну и ту же ячейку памяти в CPU.

Учтите, что входные сетевые переменные также используют общую область памяти.



*) CMS = CANopen Master/Slave (Ведущий/Ведомый)

Выходные данные

Выходные данные назначаются похожим образом.

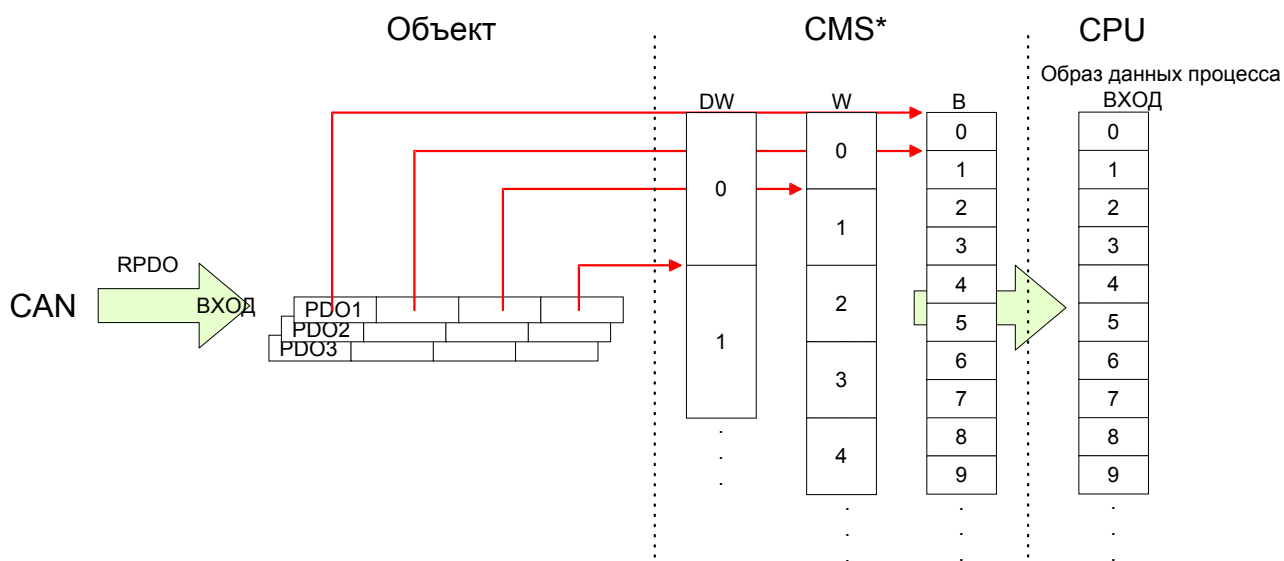
Для выходных данных применяются следующие типы объектов:

- цифровые выхода 8 бит (объект 0x6200)
- цифровые выхода 16 бит (объект 0x6300)
- цифровые выхода 32 бит (объект 0x6320)
- выходные сетевые переменные 8 бит (объект 0xA400)
- выходные сетевые переменные 16 бит (объект 0xA580)
- выходные сетевые переменные 32 бит (объект 0xA680)
- выходные сетевые переменные 64 бит (объект 0xA8C0)

Как видно из следующей иллюстрации, объекты цифровых выходов могут делить между собой пересекающиеся области памяти в CPU.

Например, объект с индексом 0x6200, с подиндексом 2 пересекается с объектом 0x6300, с подиндексом 1. Они оба занимают одну и ту же ячейку памяти в CPU.

Учтите, что выходные сетевые переменные также используют общую область памяти.



*) CMS = CANopen **M**aster/**S**lave (Ведущий/Ведомый)

Структура сообщения

Идентификатор

Все сообщения CANopen имеют структуру, соответствующую IEC 61158-2:

Идентификатор

Байт	Биты 7 ... 0
1	Биты 3 ... 0: старшие 4 бита идентификатора модуля Биты 7 ... 4: код функции CANopen
2	Биты 3 ... 0: код длины данных (DLC) Бит 4 (RTR-бит): 0: нет данных (запрос) 1: данные присутствуют в сообщении Биты 7 ... 5: Младшие 3 бита идентификатора модуля

Данные

Данные

Байт	Биты 7 ... 0
3 ... 10	Данные

Включение в двухбайтовый идентификатор кода функции и идентификатора модуля эффективно разделять сообщения по приоритетам. Код функции определяет тип сообщения (объект), а идентификатор модуля определяет получателя.

Устройства CANopen обмениваются данными в форме объектов. Профиль CANopen описывает два типа объектов, а также набор специальных объектов.

Модуль CAN фирмы VIPA для шины SPEED-Bus поддерживает следующие объекты:

- 40 PDO для передачи (связывание и отображение PDO)
- 40 PDO для приёма (связывание и отображение PDO)
- 2 стандартных SDO (1 сервер, 127 клиентов)
- 1 аварийный объект
- 1 объект управления сетью (NMT)
- Контроль узлов сети
- Сетевые импульсы (Heartbeat)

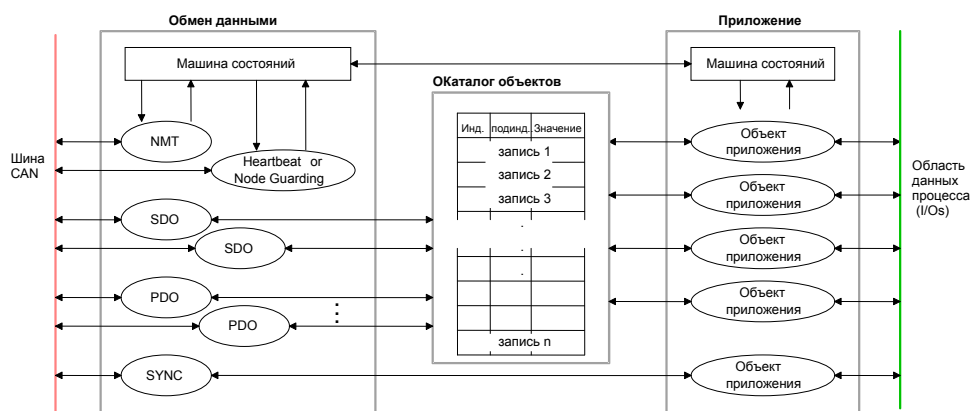


Внимание!

Полностью структура и набор данных описаны в документах (профилях) IEC 61158-2, IEC 61158-3, IEC 61158-4 and IEC 61158-5.

Структура модели устройства

Устройство CANopen может иметь следующую структуру:



Обмен данными

Обслуживаются объекты для обмена данными, такие как SDO, PDO, и дополнительные функции связи через сеть CANopen.

Приложение

Объекты приложения могут содержать входные и выходные данные. В случае ошибки машина состояний переводит выхода в безопасное состояние.

Каталог объектов организован как 2-мерная таблица. Данные выбираются по индексу и подиндексу.

Каталог объектов

Каталог объектов содержит все объекты данных (данные приложения и параметры), которые доступны или влияют на обмен, приложение или машину состояний.

PDO

Во многих полевых шинах устройства обмениваются полным образом входов-выходов, более или менее циклически. CANopen не ограничен только этим принципом обмена. Шина CAN даёт больше возможностей, являясь многомастерной шиной.

По шине CANopen данные передаются сегментами до 8 байтов. Эти сегменты называются объектами данных (**process data objects**), или PDO. Каждый PDO передаётся одной телеграммой CAN, которая имеет уникальный идентификатор, определяющий её приоритет.

Для обмена данными процесса модуль CANopen фирмы VIPA имеет 80 PDO. Каждый PDO содержит до 8 байтов данных. Передача PDO не подтверждается программно, т.к. протокол CAN гарантирует передачу.

Имеется 40 PDO для передачи данных и 40 PDO для приёма данных. Объекты PDO с точки зрения ведущего устройства CAN подразделяются следующим образом:

Принимаемые PDO (RxPDO) (ведущий модуль CAN принимает) содержат входные данные и сохраняются в области входных данных процесса PII (**process image of the inputs**).

Передаваемые PDO (TxPDO) (ведущий модуль CAN передаёт) содержат данные, сформированные в области выходных данных процесса PIQ (**process image of the outputs**).

Привязка данных ко входным или выходным PDO производится в программе WinCoCT автоматически.

SDO

Для доступа к каталогу объектов используются служебные объекты (**Service-Data-Object**), или SDO. SDO позволяют получить доступ на чтение или запись в каталог объектов. В описании протокола CAL-Layer-7 имеется спецификация на мультиплексированный протокол обмена данными (**Multiplexed-Domain-Transfer-Protocol**), применяемый для обмена объектами SDO. Этот протокол позволяет обмениваться данными любой длины. При необходимости сообщение делится на последовательность телеграмм (сегментов) CAN с одним и тем же идентификатором. При передаче SDO применяется дополнительно программное подтверждение, т.е. подтверждается каждый приём сообщения.



Внимание!

Подробное описание телеграмм SDO имеется в документе CiA DS-301. Далее описаны только сообщения об ошибках, возникающих вследствие ошибок параметров обмена.

SFC 219 CAN_TLGR Запрос SDO у ведущего модуля CAN

Каждый модуль CPU содержит встроенную функцию SFC 219. Она позволяет читать или писать SDO в модуле CAN из программы пользователя.

Ведущий модуль CAN выбирается по месту подключения, а подчинённое устройство – по его адресу CAN. Данные процесса выбираются при помощи индекса (*INDEX*) и подиндекса (*SUBINDEX*). Через SDO передаётся максимум 1 слово данных процесса. SFC 219 имеет следующие параметры:

Параметр

Параметр	Вход/Выход	Тип	Описание
REQUEST	ВХОД	BOOL	1 = Запуск
SLOT_MASTER	ВХОД	BYTE	SPEED-Bus слот (101...116)
NODEID	ВХОД	BYTE	CAN адрес (1...127)
TRANSFERTYP	ВХОД	BYTE	Тип передачи
INDEX	ВХОД	DWORD	CANopen индекс
SUBINDEX	ВХОД	DWORD	CANopen подиндекс
CANOPENERROR	ВЫХОД	DWORD	CANopen ошибка
RETVAL	ВЫХОД	WORD	Код ошибки (0=ОК, нет ошибки)
BUSY	ВЫХОД	BOOL	Занятость, функция выполняется
DATABUFFER	ВХ/ВЫХОД	ANY	Буфер для хранения данных SFC

REQUEST Управляющий параметр: 1: Запустить запрос

SLOT MASTER 101...115: Номер слота 1...15 на шині of SPEED-Bus

NODEID Адрес узла CANopen (1...127)

TRANSFERTYP	40h: Чтение SDO	23h: Запись SDO (1 DWORD, 32 бита)
		2Bh: Запись SDO (1 WORD, 16 бит)
		2Fh: Запись SDO (1 BYTE, 8 бит)

INDEX CANopen индекс

SUBINDEX CANopen подиндекс

CANOPENERROR Если ошибок нет, *CANOPENERROR* возвращает 0.
В случае ошибки *CANOPENERROR* содержит коды ошибок, генерируемые модулем CAN:

Код	Описание
0x05030000	Переключающийся бит не изменился
0x05040000	Превышен интервал времени для протокола SDO
0x05040001	Команда клиента/сервера не допустима или не известна
0x05040002	Неверный размер блока (только для блочного режима)
0x05040003	Неверный номер последовательности (только для блочного режима)
0x05040004	Ошибка контрольной суммы CRC (только для блочного режима)
0x05040005	Нет памяти
0x06010000	Неподдерживаемый режим доступа к объекту
0x06010001	Попытка чтения объекта, предназначенного только для записи
0x06010002	Попытка записи объекта, предназначенного только для чтения
0x06020000	Объект отсутствует в каталоге объектов
0x06040041	Объект не может быть сопоставлен с PDO
0x06040042	Количество и длина объектов для размещения в PDO превышает размер PDO
0x06040043	Общая ошибка совместимости параметров
0x06040047	Общая внутренняя ошибка совместимости в устройстве
0x06060000	Ошибка доступа вследствие аппаратного сбоя
0x06070010	Ошибка типа данных, длина служебного параметра не соответствует
0x06070012	Ошибка типа данных, длина служебного параметра слишком велика
0x06070013	Ошибка типа данных, длина служебного параметра слишком мала
0x06090011	Такой подиндекс отсутствует
0x06090030	Значение параметра выходит за границы (при записи)
0x06090031	Значение параметра при записи слишком велико
0x06090032	Значение параметра при записи слишком мало
0x06090036	Максимальное значение меньше минимального
0x08000000	Общая ошибка
0x08000020	Данные невозможно передать или сохранить в приложении
0x08000021	Данные невозможно передать или сохранить в приложении из-за локальной ошибки
0x08000022	Данные невозможно передать или сохранить в приложении вследствие неверного состояния устройства
0x08000023	Сбой при динамическом формировании словаря объектов или нет словаря объектов (например, например, словарь генерировался из файла и произошёл сбой доступа к вайлу)
0x08000024	Запрошенная операция не поддерживается.

RETVAL

Если функция выполнялась успешно, возвращаемое значение содержит длину полученных данных: 1: байт, 2: слово, 4: двойное слово.

Если при выполнении функции возникла ошибка, возвращаемое значение содержит код ошибки.

Код ошибки	Описание
0x F021	Неверный адрес устройства CAN (адрес = 0 или больше 127)
0x F022	Неверный тип передачи (Значение не равно 60h, 61h)
0x F023	Неверная длина данных (буфер слишком мал, для чтения SDO он должен быть не менее 4 байтов, для записи SDO - 1, 2 или 4 байта).
0x F024	SFC не поддерживается
0x F025	Буфер передачи модуля CANopen заполнен, задание не может быть выполнено в настоящее время.
0x F026	Буфер чтения модуля CANopen заполнен, задание не может быть выполнено в настоящее время.
0x F027	Чтение или запись SDO завершились с ошибкой, смотри коды ошибок CANopen.
0x F028	Превышение времени обмена SDO (нет улов CANopen с указанным номером NodeID).

BUSY

BUSY = 1: Запрос чтения/записи ещё не завершился.

DATABUFFER

Адрес буфера данных в PLC.

Чтение SDO: Область для получения прочитанных данных SDO.

Запись SDO: Область данных SDO, которые будут переданы.



Внимание!

Если запрос SDO завершён без ошибок, *RETVAL* содержит длину ответных данных (1, 2 или 4 байта), а *CANOPENERROR* = 0.

Каталог объектов

Структура

Каталог объектов CANopen содержит объекты CANopen для сетевого модуля. Каждая запись в каталоге объектов помечена 16-ти битовым индексом.

Если объект состоит из нескольких компонентов (например, объекты структурного типа и массивы), то эти компоненты помечены 8-битовым подиндексом.

Имя объекта указывает на его функцию. Атрибут типа записи соответствует типу данных объекта.

Атрибут доступа указывает, могут ли данные быть прочитаны, записаны или и то и другое.

Каталог объектов (профиль) делится на три части:

Область профиля параметров связи (0x1000 – 0x1FFF)

Эта область профиля описывает параметры связи.

0x1000 – 0x1018	Общие параметры (например, наименование устройства)
0x1400 – 0x1427	Параметры обмена (например, идентификаторы) для принимаемых объектов PDO
0x1600 – 0x1627	Карта принимаемых объектов PDO Карта содержит перекрёстные ссылки на объекты приложения, из которых сформированы PDO и размер данных зависимых.
0x1800 – 0x1827 0x1A00 – 0x1A27	Параметры обмена и карта для передаваемых объектов PDO

Область профиля производителя оборудования (0x2000 – 0x5FFF)

Здесь находятся данные производителя оборудования. Модуль CAN фирмы VIPA не имеет данных производителя.

Область профиля стандартных устройств (0x6000 – 0x9FFF)

Эта область содержит объекты стандартного профиля устройства DS-401.

**Обзор каталога
объектов**

Индекс	Содержание объекта
1000h	Тип устройства
1001h	Регистр ошибки
1005h	Распознавание COB-ID SYNC
1006h	Время ожидания SYNC
1007h	Продолжительность окна синхронизации
1008h	Наименование устройства
1009h	Версия аппаратуры
100Ah	Версия программного обеспечения
100Ch	Период контроля сети
100Dh	Параметр времени жизни
1016h	Время ожидания импульсов Heartbeat
1017h	Период посылки импульсов Heartbeat
1018h	Идентификационный объект
1400h to 1427h	Параметры обмена для RxPDO
1600h to 1627h	Карта RxPDO
1800h to 1827h	Параметры обмена TxPDO
1A00h to 1A27h	Карта TxPDO
1F22h	Краткая конфигурация устройства (DCF)
1F25h	Реконфигурация
1F80h	Запуск NMT
1F81h	Назначение подчинённых устройств
1F82h	Запрос сервиса управления NMT
1F83h	Запрос сервиса защиты Guarding
2000h	Идентификатор Rx-COB-ID для запуска OB57
2001h	Номера узлов, реагирующих на PLC-STOP
2002h	Номера узлов, реагирующих на PLC-Run
2003h	Начальный адрес счётчиков RxPDO
2004h	Начальный адрес переключателей NG/HB
2005h	Начальный адрес области сообщений L2
2016h	Контроль узла для устройств Lenze
2100h	Сообщение PLC-RUN
2101h	Сообщение PLC-STOP
2200h	J1939: Мультипакетная передача PGN
3000h	Специальные режимы CAN
6000h	8-битовые цифровые входа (см. DS 401)
6100h	16-битовые цифровые входа (см. DS 401)
6120h	32-битовые цифровые входа (см. DS 401)
6200h	8-битовые цифровые выхода (см. DS 401)
6300h	16-битовые цифровые выхода (см. DS 401)
6320h	32-битовые цифровые выхода (см. DS 401)
A040h	8-битовые сетевые входные переменные
A100h	16-битовые сетевые входные переменные
A200h	32-битовые сетевые входные переменные
A440h	64-битовые сетевые входные переменные
A4C0h	8-битовые сетевые выходные переменные
A580h	16-битовые сетевые выходные переменные
A680h	32-битовые сетевые выходные переменные
A8C0h	64-битовые сетевые выходные переменные

Тип устройства

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1000	0	Device type	Unsigned32	ro	N	0x00050191	Объявление типа устройства

32-битовое значение разделяется на 2 16-битовых поля:

Старшее слово	Младшее слово
Дополнительная информация об устройстве	Номер профиля
0000 0000 0000 wxyz (биты)	405 дес. = 0x0195

"Дополнительная информация" Содержит данные о типах сигналов данного устройства:

z=1 дискретные входа

y=1 дискретные выхода

x=1 аналоговые входа

w=1 аналоговые выхода

Регистр ошибки

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1001	0	Error Register	Unsigned8	ro	Y	0x00	Регистр ошибки

Бит 7							Бит 0
Произв.	reserved	reserved	Связь	reserved	reserved	reserved	Общее

Произв.: Ошибка, описанная производителем, см. объект 0x1003.

Связь: Ошибка обмена (переполнение CAN)

Общее: Случилась какая-либо ошибка (флаг установлен при любой ошибке)

Распознавание SYNC

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1005	0	COB-Id sync message	Unsigned32	ro	N	0x80000080	Распознавание сообщений SYNC

Младшие 11 битов 32-битного значения содержат значение (0x80=128 дес.), в котором старший бит (MSB) показывает, принимает ли устройство телеграммы SYNC (1) или нет (0).

Внимание: В противоположность идентификаторам PDO, MSB, будучи установлен, показывает, что этот идентификатор имеет значение.

Время ожидания SYNC

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x1006	0	Communi- cation cycle period	Unsigned32	rw	N	0x00000000	Максимальная продолжительность интервала SYNC, в μ s.

Если указано ненулевое значение, то ведущее устройство переходит в состояние ошибки при отсутствии SYNC-телеграммы дольше указанного времени при синхронном запросе PDO.

Продолжитель- ность окна синхронизации

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x1007	0	Synchronous window length	Unsigned32	rw	N	0x00000000	Продолжительность временного окна для синхронных PDO, в μ s.

Наименование устройства

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x1008	0	Manufacturer device name	Текстовая строка	ro	N		Имя устройства CANopen фирмы VIPA

VIPA IM 342-1CA70 = VIPA CANopen Master/Slave 342-1CA70

Поскольку возвращаемое значение больше 4 байтов, для передачи используется сегментированный протокол SDO.

Версия аппаратуры

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x1009	0	Manufacturer Hardware version	Текстовая строка	ro	N	1.00	Номер версии аппаратной реализации контроллера шины

VIPA 342-1CA70 = 1.00

Поскольку возвращаемое значение больше 4 байтов, для передачи используется сегментированный протокол SDO.

Версия программного обеспечения

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x100A	0	Manufacturer Software version	Текстовая строка	ro	N		Номер версии программного обеспечения модуля CANopen

VIPA 342-1CA70 = 1.07

Поскольку возвращаемое значение больше 4 байтов, для передачи используется сегментированный протокол SDO.

Период телеграмм защиты узла

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x100C	0	Guard time	Unsigned16	rw	N	0x0000	Интервал между двумя телеграммами защиты узла, в мс. Устанавливается ведущим NMT или при конфигурировании.

Параметр времени жизни узла

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x100D	0	Life time factor	Unsigned8	rw	N	0x00	Параметр времени жизни x период защиты узла = время жизни (сторожевой таймер для контроля работы сети)

Если телеграмма защиты узла не поступает в течение времени жизни, узел переходит в состояние ошибки. Если параметр времени жизни и/или период контроля сети равны нулю 0, узел не следит за телеграммами контроля сети, но отвечает на них ведущему устройству (контроль узла сети).

Время ожидания импульсов Heartbeat

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1016	0	Consumer heartbeat time	Unsigned8	ro	N	0x05	Количество структур
	1...127		Unsigned32	rw	N	0x00000000	Структура контроля импульса Heartbeat

Структура для контроля импульса Heartbeat:

Биты	31-24	23-16	15-0
Назначение	Reserved	Номер узла, подающего импульсы	Время ожидания импульсов
Тип данных	Unsigned8	Unsigned8	Unsigned16

Если в конфигурации узла установить время ожидания импульсов от него самого не равным 0, передача SDO прервётся и узел выставит код ошибки 06040043h.

Период посылки импульсов Heartbeat

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1017	0	Producer heartbeat time	Unsigned16	rw	N	0x0000	Задаёт время подачи импульсов heartbeat в мс

Идентификационный объект

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1018	0	Identity Object	Unsigned8	ro	N	0x04	Общая информация об устройстве (количество записей)
	1	Vendor ID	Unsigned32	ro	N	0xAFFEAFFE	Идентификатор производителя
	2	Product Code	Unsigned32	ro	N	0x3421CA70	Код изделия
	3	Revision Number	Unsigned32	ro	N		Номер версии
	4	Serial Number	Unsigned32	ro	N		Серийный номер

Параметры обмена RxPDO

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1400 ... 0x1427	0	Количество элементов	Unsigned8	ro	N	0x02	Параметры связи для принимаемых PDO. Подиндекс 0: количество следующих параметров
	1	COB-ID	Unsigned32	rw	N	0xC0000200 + Node-ID	Идентификатор COB-ID для RxPDO
	2	Transmission type	Unsigned8	rw	N	0xFF	Способ передачи для RxPDO

Подиндекс 1 (COB-ID): Младшие 11 битов 32-битового значения (биты 0-10) содержат идентификатор CAN, старший бит (MSBit, бит 31) показывает, активен PDO (1) или нет (0), бит 30 показывает, разрешён ли запрос RTR к данному PDO (0) или нет (1).

Подиндекс 2 содержит способ передачи.

Карта RxPDO

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1600 ... 0x1627	0	Number of elements	Unsigned8	rw	N	0x01	Отображение параметров в принимаемые PDO. Подиндекс 0: количество отображённых объектов
	1	1. mapped object	Unsigned32	rw	N	0x62000108	1-ый объект: (2 байта - индекс, 1 байт - подиндекс, 1 байт - размер в битах)
	2	2. mapped object	Unsigned32	rw	N	0x62000208	2-ой объект: (2 байта - индекс, 1 байт - подиндекс, 1 байт - размер в битах)
	...	...	...	...	...	...	...
	8	8. mapped object	Unsigned32	rw	N	0x62000808	8-ой объект: (2 байта - индекс, 1 байт - подиндекс, 1 байт - размер в битах)

Отображение данных в PDO автоматически поступает от ведущего устройства в зависимости от подключенных модулей.

Параметры обмена TxPDO

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1800 ... 0x1827	0	Количество элементов	Unsigned8	ro	N	0x05	Параметры связи для передаваемых PDO, Подиндекс 0: количество следующих параметров
	1	COB-ID	Unsigned32	rw	N	0x80000180 + Node-ID	Идентификатор COB-ID для TxPDO
	2	Transmission type	Unsigned8	rw	N	0xFF	Способ передачи для PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Задержка повторных посылок одинаковых PDO [значение x 100 мкс]
	5	Event time	Unsigned16	rw	N	0x0000	Таймер события [значение x 1 мс]

Подиндекс 1 (COB-ID): Младшие 11 битов 32-битового значения (биты 0-10) содержат идентификатор CAN, старший бит (MSBit, бит 31) показывает, активен PDO (1) или нет (0), бит 30 показывает, разрешён ли запрос RTR к данному PDO (0) или нет (1). Подиндекс 2 содержит способ передачи, подиндекс 3 – время задержки между посылками одного и того же PDO. Если таймеру события (подиндекс 5) присвоено ненулевое значение, PDO передаётся по истечении таймера. Если задана задержка (подиндекс 3), событие задерживается на это время.

Карта TxPDO

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1A00 ...	0	Number of elements	Unsigned8	rw	N	depending on the components fitted	Отображение параметров в передаваемые PDO.; Подиндекс 0: количество отображённых объектов
0x1A27	1	1. mapped object	Unsigned32	rw	N	0x60000108	1-ый объект: (2 байта - индекс, 1 байт - подиндекс, 1 байт - размер в битах)
	2	2. mapped object	Unsigned32	rw	N	0x60000208	2-ой объект: (2 байта - индекс, 1 байт - подиндекс, 1 байт - размер в битах)
	...	...	...	...	...	...	...
	8	8. mapped object	Unsigned32	rw	N	0x60000808	8-ой объект: (2 байта - индекс, 1 байт - подиндекс, 1 байт - размер в битах)

Отображение данных в PDO автоматически поступает от ведущего устройства в зависимости от подключенных модулей.

**Краткая
конфигурация
DCF**

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1F22	Массив	Concise DCF	Domain	rw	N		

Этот объект требуется для менеджера конфигурации. Краткий DCF – это сокращённая форма файла конфигурации устройства (**DCF** = **D**evice **C**onfiguration **F**ile).

Реконфигурация

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x1F25	Массив	ConfigureSlave	Unsigned32	rw	N	0x00000000	

Через этот объект конфигуратор может инициировать передачу сохранённой конфигурации в сеть. Реконфигурацию можно выполнить в любое время через индекс 0x1F25.

Подиндекс 0 содержит значение 128.

Подиндекс x (где x = 1...127): запускает реконфигурацию узла с номером x.

Подиндекс 128 запускает реконфигурацию всех узлов.

ПРИМЕР. Если Вы хотите запустить реконфигурацию узла 2 и имеется сохранённая конфигурация для этого узла, Вы должны записать значение 0x666E6F63 (ASCII = "conf") в объект 1F25h, подиндекс 2.

Запуск NMT

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x1F80	0x00	NMTStartup	Unsigned32	rw	N	0x00000000	

Определяет модуль как ведущее устройство управления сетью (NMT).

Бит	Назначение
Бит 0	0 Устройство не управляет сетью, все остальные биты игнорируются, список подчинённых устройств игнорируется. 1 Устройство управляет сетью. (NMT Master).
Бит 1	0 Работает только для назначенных подчинённых устройств. 1 После запуска подаёт команды NMT Start Remote Node (запуск удалённого узла) для всех узлов сети.
Биты 2-31	Зарезервировано CiA, всегда 0

Назначение подчинённых устройств

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x1F81	Мас-сив	SlaveAssignment	Unsigned32	rw	N	0x00000000	

Назначение ведомых устройств, которые управляются с ведущего.

Подиндекс 0 содержит 127. Остальные подиндексы отвечают за соответствующие номера узлов.

Байт	Бит	Описание
0	0	0 Устройство с этим номером не подчинённое 1 Устройство с этим номером подчинённое. После конфигурирования узел переводится в рабочее состояние (operational).
	1	0 Информировать приложение при появлении событий неисправности или загрузки подчинённых устройств. 1 Информировать приложение при появлении событий неисправности или загрузки подчинённых устройств, и автоматически запускать сервис управления ошибками.
	2	0 При появлении событий неисправности или загрузки подчинённых устройств не конфигурировать и не запускать подчинённые устройства автоматически. 1 При появлении событий неисправности или загрузки подчинённых устройств запускать подчинённые устройства автоматически.
	3..7	Зарезервировано CiA, всегда 0
1		8-ми битовое значение RetryFactor
2,3		16-ти битовое значение GuardTime

Запрос сервиса управления NMT

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x1F82	Мас-сив	RequestNMT	Unsigned8	rw	N	0x00000000	

Если полностью автоматический запуск сети не желателен, то функции управления сетью (NMT):

- смена состояний
- запуск сервиса защиты
- конфигурация через CMT

могут быть выполнены по запросу для каждого в отдельности узла.

Переключение состояний всех узлов сети производится через индекс 1F82h в каталоге объектов:

Подиндекс 0 содержит значение 128.

Подиндекс x (где x=1..127) иницирует сервис NMT для узла с номером x.

Подиндекс 128 иницирует запрос NMT для всех узлов сети.

Необходимое состояние записывается как число.

Состояние	Значение
Готовность	4
Рабочее	5
Сброс узла (ResetNode)	6
Сброс связи (ResetCommunication)	7
Пред-рабочее (PreOperational)	127

Запрос сервиса защиты Guarding

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x1F83	Мас-сив	RequestGuarding	Unsigned32	rw	N	0x00000000	

Подиндекс 0 содержит значение 128.

Подиндекс x (где x=1..127) иницирует/останавливает сервис защиты (guarding) для узла с номером x.

Подиндекс 128: иницирует/останавливает сервис защиты (guarding) для всех узлов.

Значе-ние	Запись	Чтение
1	Запуск защиты	Сервис защиты на устройстве запущен
0	Останов защиты	Сервис защиты на устройстве не запущен

Rx-COB-ID для запуска OB57

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x2000	0	Количество элементов	Unsigned8	ro	N	8	Количество доступных элементов в объекте
	1	1. COB-ID	Unsigned32	rw	N	0	COB-ID, запускающий OB57
	2	2. COB-ID	Unsigned32	rw	N	0	COB-ID, запускающий OB57
	...	...	...	...	...	...	...
	8	8. COB-ID	Unsigned32	rw	N	0	COB-ID, запускающий OB57

В данном индексе указываются идентификаторы объектов (COB-ID), которые при их приёме вызовут запуск OB57 в модуле CPU.

Структура идентификатора COB-ID

UNSIGNED32 – беззнаковое 32-битовое значение

MSB

LSB

Биты	31	30	29	28-11	10-0
11-битовый-ID	0/1	0/1	0	00000000000000000000	11-б. идентификатор
29-битовый-ID	0/1	0/1	1	29-битовый идентификатор	

Номер бита	Значение	Пояснения
31 (MSB)	0	PDO присутствует / действительный
	1	PDO отсутствует / не действительный
30	0	Запрос RTR разрешёт для этого PDO
	1	Запрос RTR запрещён для этого PDO
29	0	11-битовый идентификатор (CAN 2.0A)
	1	29-битовый идентификатор (CAN 2.0B)
28-11	0	Биты равны 0, если бит 29 равен 0
	X	Если бит 29=1: биты 28-11 = часть 29-битового идентификатора COB-ID
10-0 (LSB)	X	Биты 10-0 идентификатора COB-ID

Номера узлов PLC-STOP

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x2001	0x00	Количество элементов	Unsigned8	ro	N	0	Количество доступных элементов в объекте
	0x01	1. Node-ID for PLC-STOP	Unsigned8	rw	N	0	Номер останавливаемого узла (1...127)
	...	...	...	...	...	...	...
	0x10	16. Node-ID for PLC-STOP	Unsigned8	rw	N	0	Номер останавливаемого узла (1...127)

При смене состояния процессора RUN→STOP узлы сети CANopen, указанные в этом списке, будут переведены в предпусковое состояние командой NMT *Preoperational*.

Номера узлов PLC-RUN

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x2002	0x00	Количество элементов	Unsigned8	ro	N	0	Количество доступных элементов в объекте
	0x01	1. Node-ID for PLC-RUN	Unsigned8	rw	N	0	Номер запускаемого узла (1...127)
	...	...	...	...	...	...	...
	0x10	16. Node-ID for PLC-RUN	Unsigned8	rw	N	0	Номер запускаемого узла (1...127)

При смене состояния процессора STOP→RUN узлы сети CANopen, указанные в этом списке, будут переведены в рабочее состояние командой NMT *Operational*.

Начальный адрес счётчиков RxPDO

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x2003	0x00	Start address RxPDO-Counter	Unsigned8	rw	N	0	Начальный адрес счётчиков RxPDO

В этом индексе можно задать начальный адрес в области входных данных процесса (PII) массива счётчиков приёма RxPDO.

Для каждого объекта RxPDO имеется счётчик, увеличивающийся на 1 при каждом приёме PDO. При достижении значения 255 счётчик автоматически переходит на значение 1. Значение 0 счётчик принимает по умолчанию или при переходе CPU в состояние STOP.

Адрес PII	Назначение
X	Счётчик для RxPDO 1
X+1	Счётчик для RxPDO 2
X+2	Счётчик для RxPDO 3
X+3	Счётчик для RxPDO 4
X+4	Счётчик для RxPDO 5
X+5	Счётчик для RxPDO 6
...	
X+35	Счётчик для RxPDO 36
X+36	Счётчик для RxPDO 37
X+37	Счётчик для RxPDO 38
X+38	Счётчик для RxPDO 39
X+39	Счётчик для RxPDO 40
X+40	Счётчик для сообщения SYNC

Начальный адрес переключателей NG/HB

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x2003	0x00	Start address NG/HB –Toggle Bit	Unsigned8	rw	N	0	Начальный адрес переключателей NG/HB

В этом индексе можно задать начальный адрес в области входных данных процесса (PII) для массива переключателей NG/HB (NodeGuarding/Heartbeat). Для признаков NG/HB отводится 1 бит, который меняет своё состояние при приёме соответствующего объекта (COB-ID). По умолчанию и при переходе CPU в режим STOP биты NG/HB принимают значение 0.

Адрес PII	Назначение
X	Переключатель для COB-ID 0x701 .. 0x708
X+1	Переключатель для COB-ID 0x709 .. 0x710
X+2	Переключатель для COB-ID 0x711 .. 0x718
X+3	Переключатель для COB-ID 0x719 .. 0x720
X+4	Переключатель для COB-ID 0x721 .. 0x728
X+5	Переключатель для COB-ID 0x729 .. 0x730
X+6	Переключатель для COB-ID 0x731 .. 0x738
X+7	Переключатель для COB-ID 0x739 .. 0x740
X+8	Переключатель для COB-ID 0x741 .. 0x748
X+9	Переключатель для COB-ID 0x749 .. 0x750
X+10	Переключатель для COB-ID 0x751 .. 0x758
X+11	Переключатель для COB-ID 0x759 .. 0x760
X+12	Переключатель для COB-ID 0x761 .. 0x768
X+13	Переключатель для COB-ID 0x769 .. 0x770
X+14	Переключатель для COB-ID 0x771 .. 0x778
X+15	Переключатель для COB-ID 0x779 .. 0x77F

Начальный адрес области сообщений L2

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x2005	0x00	Start address L2-Message-Area	Unsigned8	rw	N	0	Начальный адрес области сообщений L2

В этом индексе можно задать начальный адрес в области входных/выходных данных процесса (PII/PIQ) для отправки телеграмм из программы пользователя через область сообщений. В области сообщений предусмотрено место для 5 различных телеграмм. Сообщения отправляются по команде, записываемой в байт команд.

По умолчанию и при переходе CPU в режим STOP все байты области сообщений принимают значение 0.

Адр. PIQ	Канал	Тип	Назначение	Адр. PII	Канал	Тип	Назначение
X	0	B	Команда	X	0	B	Состояние
X+1		B	Длина дан.	X+1	1	B	Состояние
X+2		DW	Идентификатор объекта COB-ID	X+2	2	B	Состояние
X+3				X+3	3	B	Состояние
X+4				X+5	4	B	Состояние
X+5							
X+6		B	Данные 0				
X+7		B	Данные 1				
X+8		B	Данные 2				
X+9		B	Данные 3				
X+10		B	Данные 4				
X+11		B	Данные 5				
X+12		B	Данные 6				
X+13		B	Данные 7				
X+14	1	B	Команда				
X+15		B	Длина дан.				
X+16		DW	Идентификатор объекта COB-ID				
X+17							
X+18							
X+19							
X+20		B	Данные 0				
X+21		B	Данные 1				
X+22		B	Данные 2				
X+23		B	Данные 3				
X+24		B	Данные 4				
X+25		B	Данные 5				
X+26		B	Данные 6				
X+27		B	Данные 7				
X+28	2	B	Команда				
X+29		B	Длина дан.				
X+30		DW	Идентификатор объекта COB-ID				
X+31							
X+32							
X+33							
X+34		B	Данные 0				
X+35		B	Данные 1				
X+36		B	Данные 2				
X+37		B	Данные 3				
X+38		B	Данные 4				
X+39		B	Данные 5				
X+40		B	Данные 6				
X+41		B	Данные 7				

продолжение следует ...

... продолжение

Адр. PIQ	Канал	Тип	Назначение	Адр. PII	Канал	Тип	Назначение
X+42	3	B	Команда				
X+43		B	Длина дан.				
X+44		DW	Идентификатор объекта COB-ID				
X+45							
X+46							
X+47							
X+48		B	Данные 0				
X+49		B	Данные 1				
X+50		B	Данные 2				
X+51		B	Данные 3				
X+52		B	Данные 4				
X+53		B	Данные 5				
X+54		B	Данные 6				
X+55		B	Данные 7				
X+56	4	B	Команда				
X+57		B	Длина дан.				
X+58		DW	Идентификатор объекта COB-ID				
X+59							
X+60							
X+61							
X+62		B	Данные 0				
X+63		B	Данные 1				
X+64		B	Данные 2				
X+65		B	Данные 3				
X+66		B	Данные 4				
X+67		B	Данные 5				
X+68		B	Данные 6				
X+69		B	Данные 7				

Модуль CANopen	Модуль CPU
Инициализация / Переход PLC в режим STOP	
Область сообщений L2: Данные заполняются нулями	OB100: Область PIQ отправки сообщений L2 заполняется нулями
Отправка телеграммы	
Статус PII не равен команде PIQ? → Поместить телеграмму в очередь для отправки → установить статус PII = команде PIQ	Статус PII равен команде PIQ? → Заполнить COB-ID и данные → увеличить команду в PIQ на 1

Контроль узла для устройств Lenze

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x2016	0x00	Количество элементов	Unsigned8	ro	N	127	Количество доступных элементов в объекте
	0x01	1. entry	Unsigned32	rw	N	0	Контроль узла номер 1
	0x02	2. entry	Unsigned32	rw	N	0	Контроль узла номер 2
	0x03	3. entry	Unsigned32	rw	N	0	Контроль узла номер 3
	...	...	...	...	...	...	...
	0x7F	127. entry	Unsigned32	rw	N	0	Контроль узла номер 127

Этот индекс разработан специально для устройств Lenze. Эти устройства не поддерживают протоколы Nodeguarding и Heartbeat для CANopen, описанные в спецификации DS301.

Вместо этого применяется протокол SDO. Запрос SDO посылается периодически модулем CAN к устройству Lenze (Период = $TimeoutValue * 100ms$). Если от устройства Lenze нет ответов в течение таймута в 1 сек., то в модуль CPU передаётся состояние ошибки (вызывается OB86).

Структура данных для контроля модуля Lenze

Биты	31-16	15-8	7-0
Значение	Индекс	Подиндекс	<i>TimeoutValue</i>
Представление	Unsigned16	Unsigned8	Unsigned8

Пример для Lenze:

entry = 0x5E980005

Индекс=0x5E98 (равнозначен коду Lenze C0359), подиндекс=0, *TimeoutValue*=5, Период=5*100=500ms

Сообщение PLC-RUN

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x2100	0x00	Количество элементов	Unsigned8	ro	N	10	Количество доступных элементов в объекте
	0x01	COB-ID	Unsigned32	rw	N	0	Идентификатор объекта COB-ID
	0x02	Data length	Unsigned8	rw	N	0	Длина данных
	0x03	Data 1	Unsigned8	rw	N	0	Данные 1
	0x04	Data 2	Unsigned8	rw	N	0	Данные 2
	0x05	Data 3	Unsigned8	rw	N	0	Данные 3

В этом индексе можно описать телеграмму CAN, которая будет отправлена в сеть при смене состояний модуля CPU STOP→RUN.

Сообщение PLC-STOP

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x2101	0x00	Количество элементов	Unsigned8	ro	N	10	Количество доступных элементов в объекте
	0x01	COB-ID	Unsigned32	rw	N	0	Идентификатор объекта COB-ID
	0x02	Data length	Unsigned8	rw	N	0	Длина данных
	0x03	Data 1	Unsigned8	rw	N	0	Данные 1
	0x04	Data 2	Unsigned8	rw	N	0	Данные 2
	0x05	Data 3	Unsigned8	rw	N	0	Данные 3
	0x06	Data 4	Unsigned8	rw	N	0	Данные 4
	0x07	Data 5	Unsigned8	rw	N	0	Данные 5
	0x08	Data 6	Unsigned8	rw	N	0	Данные 6
	0x09	Data 7	Unsigned8	rw	N	0	Данные 7
	0x0A	Data 8	Unsigned8	rw	N	0	Данные 8

В этом индексе можно описать телеграмму CAN, которая будет отправлена в сеть при смене состояний модуля CPU RUN→STOP.

J1939: Мультипакетная передача PGN

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x2200	0x00	Количество элементов	Unsigned8	ro	N	16	Количество доступных элементов в объекте
	0x01	1. PGN	Unsigned8	rw	N	0	PGN
	...	...	...	...	...	...	...
	0x10	16. PGN	Unsigned8	rw	N	0	PGN

Этот индекс предназначен для протокола J1939.

При помощи мультипакетного протокола J1939 можно передавать большие по объёму данные. В этом протоколе используются идентификаторы COB-ID: 20ECFF00h и 20EBFF00h.

Номер PGN и длина данных передаётся идентификатором COB-ID 20ECFF00h. Сегментированные данные передаются затем под идентификатором COB-ID 20EBFF00h. Для корректной работы с данными в конфигураторе WinCoСТ устанавливается параметр "Manufacturer Alarm (OB 57)" (сигнал производителя) и количество сообщений = 1. Дополнительно, в индексе 2000h должны быть указаны идентификаторы COB-ID 20ECFF00h и 20EBFF00h.

Количество вызовов OB 57 в CPU можно ограничить индексом 0x2200. Блок OB 57 вызывается только в ответ на PGN-телеграммы, номера которых указаны в этом индексе.

Специальные режимы CAN

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x3000	0x00	Special Settings for CAN	Unsigned8	rw	N	0	Специальные режимы CAN

Этот индекс меняет некоторые режимы работы CAN.

Бит 0: Включение/отключение проверки длины RxPDO

Бит = 0: Проверка длины включена

Бит = 1: Проверка длины отключена

Биты 6...1: Зарезервировано

Бит 7: Специальный бит для J1939

Бит = 0: Приоритет объектов COB-ID протокола J1939 проверяется

Бит = 1: Приоритет объектов COB-ID протокола J1939 не проверяется

8-битовые цифровые входа

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x6000	0x00	8bit digital input block	Unsigned8	ro	N	0x01	Количество доступных 8-битовых входных блоков
	0x01	1. input block	Unsigned8	ro	Y		1. блок цифровых входов
	... 0xFE	... 254. input block	... Unsigned8	... ro	... Y	...	... 64. блок цифровых входов

16-битовые цифровые входа

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x6100	0x00	16bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 16-битовых входных блоков
	0x01	1. input block	Unsigned16	ro	N		1. блок цифровых входов
	... 0xA0	... 160. input block	... Unsigned16	... ro	... N	...	... 32. блок цифровых входов

32-битовые цифровые входа

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0x6120	0x00	32bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 32-битовых входных блоков
	0x01	1. input block	Unsigned32	ro	N		1. блок цифровых входов
	... 0x50	... 80. input block	... Unsigned32	... ro	... N	...	... 16. блок цифровых входов

**8-битовые
цифровые
выхода**

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x6200	0x00	8bit digital output block	Unsigned8	ro	N	0x01	Количество доступных 8-битовых выходных блоков
	0x01	1. output block	Unsigned8	rw	Y		1. блок цифровых выходов
	...	...	...	...	...	...	...
	0xFE	254. output block	Unsigned8	rw	Y		64. блок цифровых выходов

**16-битовые
цифровые
выхода**

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x6300	0x00	16bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 16-битовых выходных блоков
	0x01	1. output block	Unsigned16	rw	N		1. блок цифровых выходов
	...	...	...	...	...	...	...
	0x0A	160. output block	Unsigned16	rw	N		32. блок цифровых выходов

**32-битовые
цифровые
выхода**

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0x6320	0x00	32bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 32-битовых выходных блоков
	0x01	1. output block	Unsigned32	rw	N		1. блок цифровых выходов
	...	...	...	...	...	...	...
	0x50	80. output block	Unsigned32	rw	N		16. блок цифровых выходов

8-битовые сетевые входные переменные

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0xA040	0x00	8bit digital input block	Unsigned8	ro	N	0x01	Количество доступных 8-битовых выходных блоков
	0x01	1. input block	Unsigned8	ro	Y		1. блок цифровых выходов
	... 0xFE	... 254. input block	... Unsigned8	... ro	... Y	...	... 320. блок цифровых выходов

16-битовые сетевые входные переменные

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0xA100	0x00	16bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 16-битовых выходных блоков
	0x01	1. input block	Unsigned16	ro	N		1. блок цифровых выходов
	... 0xA0	... 160. input block	... Unsigned16	... ro	... N	...	... 160. блок цифровых выходов

32-битовые сетевые входные переменные

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0xA200	0x00	32bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 32-битовых выходных блоков
	0x01	1. input block	Unsigned32	ro	N		1. блок цифровых выходов
	... 0x50	... 80. input block	... Unsigned32	... ro	... N	...	... 80. блок цифровых выходов

64-битовые сетевые входные переменные

Индекс	Под-индекс	Имя	Тип	Атрибут	Карта	Значение по умолчанию	Назначение
0xA440	0x00	64bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 64-битовых выходных блоков
	0x01	1. input block	Unsigned32	ro	N		1. блок цифровых выходов
	... 0x28	... 40. input block	... Unsigned32	... ro	... N	...	... 40. блок цифровых выходов

8-битовые сетевые выходные переменные

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0xA400	0x00	8bit digital output block	Unsigned8	ro	N	0x01	Количество доступных 8-битовых выходных блоков
	0x01	1. output block	Unsigned8	rw	Y		1. digital output block
	... 0xFE	... 254. output block	... Unsigned8	... rw	... Y	...	... 320. digital output block

16-битовые сетевые выходные переменные

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0xA580	0x00	16bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 16-битовых выходных блоков
	0x01	1. output block	Unsigned16	rw	N		1. блок цифровых выходов
	... 0xA0	... 160. output block	... Unsigned16	... rw	... N	...	... 160. блок цифровых выходов

32-битовые сетевые выходные переменные

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0xA680	0x00	32bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 32-битовых выходных блоков
	0x01	1. output block	Unsigned32	rw	N		1. блок цифровых выходов
	... 0x50	... 80. output block	... Unsigned32	... rw	... N	...	... 80. блок цифровых выходов

64-битовые сетевые выходные переменные

Индекс	Под-индекс	Имя	Тип	Атри-бут	Карта	Значение по умолчанию	Назначение
0xA8C0	0x00	64bit digital input block	Unsigned8	ro	N	в зависимости от установленных компонентов	Количество доступных 64-битовых выходных блоков
	0x01	1. output block	Unsigned32	rw	N		1. блок цифровых выходов
	... 0x50	... 40. output block	... Unsigned32	... rw	... N	...	... 40. блок цифровых выходов

Диагностика

Введение

Если пункт "Diagnostic" в разделе "PLC-Parameters" программы WinCoСТ выбран, то модуль CAN может вызывать события, при которых срабатывают соответствующие OB в программе пользователя CPU.

- Изменение состояния модуля CAN (OB 82)
- Сбой и восстановление ведомого устройства (OB 86)
- Ошибка защиты узла (OB 82)
- Телеграмма об ошибке (OB 82)

Чтение диагностических функций SFC 13

Из организационного блока (OB) можно прочитать диагностические данные с использованием функции SFC 13 DPNRM_DG.

Входной параметр *RECORD* задаёт область памяти, куда, при успешном завершении, помещаются считанные из ведомого устройства данные. Операция чтения запускается при переходе в 1 входного параметра *REQ*.

Параметры

Параметр	Тип параметра	Тип данных	Блок памяти	Описание
REQ	INPUT (вход)	BOOL	I, Q, M, D, L, константа	<i>REQ</i> = 1: Запрос на чтение
LADDR	INPUT	WORD	I, Q, M, D, L, константа	Адрес модуля CAN VIPA 342-1CA70: 6000h + (n·100h) + Номер-Узла, где n = (1...8) – номер слота SPEED-Bus
RET_VAL	OUTPUT (выход)	INT	I, Q, M, D, L	Возвращаемое значение
RECORD	OUTPUT	ANY	I, Q, M, D, L	Область памяти для сохранения прочитанных данных. Допускается только тип BYTE.
BUSY	OUTPUT	BOOL	I, Q, M, D, L	<i>BUSY</i> = 1: операция чтения выполняется <i>BUSY</i> = 0: операция чтения завершена

RECORD

Перед тем, как прочитать данные, CPU выясняет их размер. Возможны два варианта:

- Если длина области памяти в параметре *RECORD*, меньше, чем необходимо, то данные отклоняются и в *RET_VAL* возвращается соответствующий код ошибки.
- Если длина области памяти в параметре *RECORD* достаточна, то данные переносятся в память CPU и *RET_VAL* возвращает реальный размер данных как положительное число.



Внимание!

Очень важно использовать соответствующие параметры *RECORD* для каждой операции. Операция однозначно идентифицируется по двум параметрам - *LADDR* и *RECORD*.

Описание работы функции SFC13

Функция SFC 13 выполняется асинхронно, за время выполнения запроса происходит несколько вызовов функции. Выходные параметры *RET_VAL* и *BUSY* отображают статус выполнения запроса, как показано в следующей таблице.

Взаимосвязь между параметрами *REQ*, *RET_VAL* и *BUSY* при вызовах:

Порядковый номер вызова	Тип вызова	REQ	RET_VAL	BUSY
1	первый вызов	1	7001h или код ошибки	1 0
2 до (n-1)	промежуточные вызовы	не зависимо	7002h	1
n	last call	не зависимо	Если запрос завершился без ошибок, то возвращается количество считанных байтов как положительное число. В случае ошибки возвращается код ошибки.	0 0

RET_VAL
(Возвращаемое значение)

При возникновении ошибки выполнения функции *RET_VAL* содержит код ошибки. При удачном завершении функции, параметр *RET_VAL* размер прочитанных данных.

Информация об ошибках

Подробная информация об SFC 13 и возвращаемых кодах ошибок содержится в руководстве "Список инструкций" с заказным номером VIPA HB140E_OPList.

Структура данных диагностики

Обычно, длина диагностических данных 35 байтов. Если в поле *station state 1* (байт состояния 1) бит 3 "DiagExtDiag" = 0, то доступна только диагностика CAN длиной 6 байтов.

В следующей таблице показана структура диагностических данных:

	Байт	Описание
Диагностические данные CAN	0	Состояние 1
	1	Состояние 2
	2	Состояние 3
	3	Номер узла
	4	всегда 0
	5	Тип устройства
Расширенная диагностика	6 ... 34	Сообщение о состоянии

Состояние 1

Бит	Наименование	Описание
0	DiagStationNonExistent	1 = Станция отсутствует 0 = Станция присутствует (если поступает сообщение о загрузке или защите узла, бит устанавливается в 0)
1	DiagStationNotReady	1 = Состояние станции предпусковое 0 = Станция в работе
2	-	зарезервировано
3	DiagExtDiag	0 = Поступила только диагностика CAN 1 = Имеется расширенная диагностика
7...4	-	зарезервировано

Состояние 2

Бит	Наименование	Описание
0	DiagPrmReq	0 = Станция сконфигурирована успешно 1 = Требуется переконфигурация
1	-	зарезервировано
2	-	всегда 1
3	DiagWD_ON	0 = Протокол защиты не реализован 1 = Защита узла включена
7...4	-	зарезервировано

Состояние 3

Этот байт зарезервирован для будущих расширений.

Номер узла

Идентификатор станции, от которой поступила диагностика.

Тип устройства

Тип станции, от которой поступила диагностика.

0 = Ведомое устройство

1 = Ведущее устройство NMT

Сообщение о состоянии

Байт	Наименование	Описание
0	Header	Всегда 29
1	Type	Всегда 81h
2	SlotNr	Всегда 0
3	Specifier	Характеристика сообщения 0 = не известно 1 = Сообщение присутствует 2 = Сообщение отсутствует
4..7	VendorID	CANopen: индекс 1018 подиндекс 1
8..11	ProductCode	CANopen: индекс 1018 подиндекс 2
12..15	RevisionNr	CANopen: индекс 1018 подиндекс 3
16..19	SerialNr	CANopen: индекс 1018 подиндекс 4
20	DiagError	Код ошибки (10h ... 31h) 10h = DIAG_SLAVEBOOTUP 11h = DIAG_SLAVEGRDERROR 12h = DIAG_SLAVESDOERROR 13h = DIAG_SLAVEEMCYIND
21..28	DiagErrorData	Дополнительные сведения об ошибке

Обзор
DiagError
DiagErrorData

Длина дополнительных сведений об ошибке всегда 8 байт.

DIAG_SLAVE
BOOTUP (10h)

Это сообщение вырабатывается как только ведущее устройство получает сообщение о загрузке ведомого устройства.

Дополнительные сведения: Все 8 байтов заполнены 0.

DIAG_SLAVE
GRDERROR (11h)

Если ведомое устройство не отвечает на запросы протокола защиты узла или от него не поступают импульсы присутствия (heartbeat), ведущее устройство формирует следующее сообщение:

Дополнительные сведения:

Байт	Назначение
0	Код события
1	Текущий статус
2	Ожидаемый статус
3...7	Всегда 0

Код события	Описание
0	Протокол защиты узла не применяется.
1	Защита узла активирована (вновь). Данное сообщение поступает как при активации защиты, так и при восстановлении состояния защиты в норму.
2	Сбой протокола защиты: нет ответа от ведомого устройства в течение цикла протокола (<i>Guardtime</i>).
3	Авария протокола защиты: нет ответа от ведомого устройства в течение времени жизни станции: $Guardtime(t) * LifeTimeFactor(n)$. Авария срабатывает до того, как сообщение с кодом 2 придёт n раз.
4	Значение бита переключателя в сообщении от ведомого не соответствует ожидаемому значению. Ведущий запоминает новое значение переключателя, так что это разовая ошибка.
5	Ведомый объявляет о статусе, который не был запрошен ведущим. Эта ошибка возникает при произвольной смене состояния ведомым устройством. Данное сообщение приходит постоянно, пока несоответствие не будет устранено.
6	Сбой контроля присутствия ведомого устройства (протокол heartbeat). Нет импульсов присутствия в течение времени, указанного в таблице.

Текущий/ожидаемый статус	Описание
4	Готовность
5	Работа
6	Сброс
7	Сброс коммуникации
127	Предпусковой

DIAG_SLAVE
SDOERROR (12h)

Дополнительные сведения:

Byte	Code
0	Старший байт индекса SDO
1	Младший байт индекса SDO
2	Подиндекс SDO
3...6	Код ошибки CANOPENERROR
7	Всегда 0

Код	Описание
0x05030000	Переключающийся бит не изменился
0x05040000	Превышен интервал времени для протокола SDO
0x05040001	Команда клиента/сервера не допустима или не известна
0x05040002	Неверный размер блока (только для блочного режима)
0x05040003	Неверный номер последовательности (только для блочного режима)
0x05040004	Ошибка контрольной суммы CRC (только для блочного режима)
0x05040005	Нет памяти
0x06010000	Неподдерживаемый режим доступа к объекту
0x06010001	Попытка чтения объекта, предназначенного только для записи
0x06010002	Попытка записи объекта, предназначенного только для чтения
0x06020000	Объект отсутствует в каталоге объектов
0x06040041	Объект не может быть сопоставлен с PDO
0x06040042	Количество и длина объектов для размещения в PDO превышает размер PDO
0x06040043	Общая ошибка совместимости параметров
0x06040047	Общая внутренняя ошибка совместимости в устройстве
0x06060000	Ошибка доступа вследствие аппаратного сбоя
0x06070010	Ошибка типа данных, длина служебного параметра не соответствует
0x06070012	Ошибка типа данных, длина служебного параметра слишком велика
0x06070013	Ошибка типа данных, длина служебного параметра слишком мала
0x06090011	Такой подиндекс отсутствует
0x06090030	Значение параметра выходит за границы (при записи)
0x06090031	Значение параметра при записи слишком велико
0x06090032	Значение параметра при записи слишком мало
0x06090036	Максимальное значение меньше минимального
0x08000000	Общая ошибка
0x08000020	Данные невозможно передать или сохранить в приложении
0x08000021	Данные невозможно передать или сохранить в приложении из-за локальной ошибки
0x08000022	Данные невозможно передать или сохранить в приложении вследствие неверного состояния устройства
0x08000023	Сбой при динамическом формировании словаря объектов или нет словаря объектов (например, например, словарь генерировался из файла и произошёл сбой доступа к вайлу)
0x08000024	Запрошенная операция не поддерживается.

DIAG_SLAVE
EMCYIND (13h)

Дополнительные сведения: Телеграмма о чрезвычайной ситуации
Для передачи сообщений о внутренних сбоях устройств абонентам сети CAN-Bus с высоким приоритетом, сетевой модуль CAN фирмы VIPA поддерживает Чрезвычайный Объект.

Этот объект имеет высокий приоритет и сообщает важную информацию о состоянии устройства и сети.

Чрезвычайная телеграмма имеет всегда длину 8 байтов. Она начинается с 2-байтового кода ошибки, затем следует 1-байтовый регистр ошибки и в конце дополнительный код длиной 5 байтов.

Структура
телеграммы

Код ошибки		Регистр ошибки	Инф. байт 0	Инф. байт 1	Инф. байт 2	Инф. байт 3	Инф. байт 4
Младший байт	Старший байт	ErrorRegister, индекс 0x1001					

Код ошибки	Описание	Инф.байт 0	Инф.байт 1	Инф.байт 2	Инф.байт 3	Инф.байт 4
0x0000	Сброс ошибки	0x00	0x00	0x00	0x00	0x00
0x1000	Управление PDO	0xFF	0x10	Номер PDO	Млад.байт время	Стар.байт время
0x6200	Останов контроллера PLC-STOP	1=PLC-STOP	0x00	0x00	0x00	0x00
0x6363	Отображение PDO	Млад.байт: Парам. отображения	Стар.байт: Парам. отображения	Отображения	0x00	0x00
0x8100	Потребитель heartbeat	Номер узла	Млад.байт время	Стар.байт время	0x00	0x00
0x8100	Блочная передача SDO	0xF1	Млад.байт индекс	Стар.байт индекс	Подиндекс	0x00
0x8130	Ошибка защиты узла	Млад.байт GuardTime	Стар.байт GuardTime	LifeTime	0x00	0x00
0x8210	PDO не обраб. из-за ошибки длины	Номер PDO	Неверная длина	Длина PDO	0x00	0x00
0x8220	Превышение длины PDO	Номер PDO	Неверная длина	Длина PDO	0x00	0x00

Чтение SZL

Введение

Текущее состояние системы описывается списком состояний (SZL). Список SZL можно считывать только частями. Этот список строится в модуле CPU по требованию.

Для идентификации частей списка существует код SZL-ID.

Чтение SZL функцией SFC 51

Функция SFC 51 RDSYSST (чтение состояния системы) возвращает частичный список SZL или выдержку из частичного списка SZL.

Операция чтения запускается при установке параметра *REQ* в значение 1 при вызове SFC 51. Если операция чтения выполняется немедленно, то выходной параметр *BUSY* возвращает значение 0. *BUSY* возвращает значение 1 пока операция чтения не завершится.

Параметры

Параметр	Тип парам.	Тип данных	Операнд	Описание
REQ	INPUT (вход)	BOOL	I,Q,M,D,L, константа	<i>REQ</i> =1: запуск операции
SZL_ID	INPUT	WORD	I,Q,M,D,L, константа	<i>SZL-ID</i> идентификатор части списка или выдержки
INDEX	INPUT	WORD	I,Q,M,D,L, константа	Тип или размер объекта списка. Для модуля 342-1CA70 индекс = 6100h.
RET_VAL	OUTPUT (выход)	INT	I,Q,M,D,L	При возникновении ошибки выполнения возвращает код ошибки.
BUSY	OUTPUT	BOOL	I,Q,M,D,L	<i>BUSY</i> =1: операция чтения не завершена
SZL_HEADER	OUTPUT	STRUCT	D,L	Смотри ниже
DR	OUTPUT	ANY	I,Q,M,D,L	Область памяти для сохранения частичного списка SZL (выдержки): - В случае чтения только заголовка частичного списка SZL используется только <i>SZL_HEADFC</i>, <i>DR</i> не используется - В остальных случаях поля <i>LENGTHDR</i> и <i>N_DR</i> показывают размер данных, принятых в <i>DR</i>.

SZL_HEADER Параметр *SZL_HEADER* – это структура следующего содержания:

```
SZL_HEADER: STRUCT
    LENGTHDR: WORD
    N_DR:      WORD
END_STRUCT
```

Поле LENGTHDR показывает размер одной записи частичного списка SZL или выдержки из частичного списка SZL.

- Если Вы запрашиваете только заголовок частичного списка SZL, то поле N_DR содержит количество доступных записей.
- Иначе N_DR содержит количество записей, переданных в область памяти CPU.

RET_VAL
(Возвращаемое
значение)

RET_VAL содержит код ошибки, если во время выполнения функции возникла ошибочная ситуация. Иначе - *RET_VAL* содержит 0000h.

Дополнительная
информация

Дополнительная информация об SFC 51 и ошибках можно найти в руководстве "Список инструкций" с заказным номером: VIPA HB140E_OPList.

Списки SZL модуля CAN

Списки SZL имеют длину 8 слов.

Начиная с бита 0, каждый бит SZL соответствует номеру узла, в порядке возрастания. Бит 0 байта 0 соответствует номеру узла 1. Бит 3 байта 1 соответствует номеру узла 12.

Следующие идентификаторы SZL поддерживаются модулем CAN:

Идент. SZL-ID	Описание
0x92	Сконфигурированные ведомые станции: Бит=0: Станция не сконфигурирована Бит=1: Станция сконфигурирована
0x192	Активированные ведомые станции: Бит=0: Станция не запроектирована или не активирована Бит=1: Станция запроектирована и активирована
0x292	Текущее состояние ведомых устройств: Бит=0: Станция не запроектирована, не активирована или авария станции Бит=1: Станция запроектирована, активирована и в рабочем режиме
0x692	Диагностическая информация по станциям: Бит=0: Станция присутствует, доступна, исправна и активирована Бит=1: Станция не в норме или деактивирована

(Де-)Активация станции

Вступление

Существует возможность деактивировать и повторно активировать подключенное ведомое устройство (станцию), а также определить состояние при помощи функции SFC 12.

Если Вы описываете станции, которые временно отсутствуют или пока не нужны, CPU всё равно пытается периодически обращаться к ним. После того, как станции деактивированы, CPU прекращает попытки доступа к ним. В этом случае можно достичь наилучшего времени цикла и соответствующие сообщения об ошибках больше не будут поступать.



Внимание!

Пока функция SFC 12 выполняется, невозможно загрузить новую конфигурацию в CPU.

Также, CPU отвергает запросы SFC 12, если получает запрос на загрузку изменённой конфигурации.

Использование SFC 12

Функция SFC 12 выполняется асинхронно, за время выполнения запроса происходит несколько вызовов функции. Операция чтения запускается при переходе в 1 входного параметра *REQ*.

Выходные параметры *RET_VAL* и *BUSY* отображают статус выполнения запроса. Вызов SFC 12 из OB 100 не поддерживается.

Если Вы запускаете активацию или деактивацию станции при помощи вызова SFC 12, и предыдущий запрос ещё не завершён, то реакция зависит от того, что Вы собираетесь делать. Если параметр *LADDR* тот же, что был в предыдущий раз, то вызов интерпретируется как следующий.

Параметры

Параметр	Тип параметра	Тип данных	Аргумент	Описание
REQ	INPUT (вход)	BOOL	I,Q,M,D,L, константа	Запустить активацию или деактивацию при переходе <i>REQ</i> в значение 1
MODE	INPUT	BYTE	I,Q,M,D,L, константа	Идентификатор запроса Job-ID Возможные значения: 0: Запрос состояния 1: Активировать станцию 2: Деактивировать станцию
LADDR	INPUT	WORD	I,Q,M,D,L, константа	Адрес модуля CAN, VIPA 342-1CA70: 6000h + (n·100h) + Node-ID (где n=1...8, номер слота SPEED-Bus)
RET_VAL	OUTPUT (выход)	INT	I,Q,M,D,L	Код ошибки, если произошла ошибка во время выполнения функции
BUSY	OUTPUT	BOOL	I,Q,M,D,L	<i>BUSY</i> =1: Запрос выполняется <i>BUSY</i> =0: Запрос не выполнен

Информация об ошибках

Подробная информация об SFC 12 и кодах ошибок имеется в руководстве "Перечень инструкций" с заказным номером: VIPA HB140E_OPList.

Деактивация станции

При деактивации станции функцией SFC 12 состояние её меняется на предпусковое (безопасное).

Ведущее устройство, за которым закреплена станция, перестаёт обращаться к ней. Деактивированная станция не считается отсутствующей или аварийной и не приводит к аварийной индикации на ведущем устройстве или на CPU.

Образ входов деактивированной станции заполняется нулями, как и у аварийных устройств.

Если в программе пользователя используется непосредственный доступ к данным деактивированного устройства, будет вызван обработчик ошибок OB 122 и в диагностический буфер добавится соответствующее событие.

Деактивация станции не приводит к вызову обработчика ошибок OB 85, даже если её входа или выхода входят в образ данных процесса. Никаких сообщений не формируется в диагностическом буфере.

Деактивация станции не вызывает запуска обработчика аварии ведомого устройства OB 86, и операционная система не формирует никаких сообщений в диагностическом буфере.

Если нарушается работа станции, деактивированной при помощи SFC 12, операционная система не определяет аварии. В результате не происходит ни вызова OB 86, ни записи в диагностическом буфере. В этом случае авария станции определяется только после повторной активации и может быть обнаружена по значению *RET_VAL*.



Внимание!

Невозможно деактивировать все станции. Хотя бы одно подчинённое устройство должно остаться в работе.

Активация станции

Когда Вы вновь активируете станцию при помощи функции SFC 12, она заново параметрируется ведущим устройством, так же, как и в случае устранения неисправности на сбойной станции. Активация завершается когда станция способна обмениваться данными.

Активация станции не приводит к вызову обработчика ошибок OB 85, даже если её входа или выхода входят в образ данных процесса. Никаких сообщений не формируется в диагностическом буфере.

Деактивация станции не вызывает запуска обработчика аварии ведомого устройства OB 86, и операционная система не формирует никаких сообщений в диагностическом буфере.



Внимание!

Активация станции занимает некоторое время. Можно отменить активацию вызовом SFC 12 с тем же *LADDR* и *MODE* = 2. Вызов SFC 12 в этом случае нужно повторять до успешной отмены активации, когда *RET_VAL* = 0.

