Организация / держатель: _____

СОГЛАСОВАНО

Должность _____

Подпись _____ ФИО _____

Дата _____

УТВЕРЖДАЮ

Должность _____

Подпись _____ ФИО _____

Дата _____

KNV DemoBlocks

Библиотека функциональных блоков

Версия 0.1.0.0

Руководство программиста

Лист утверждения

КНВ.00002-01 33 01-ЛУ

Электронный документ PDF

Разработал _____

Подпись _____ ФИО _____

Нормоконтроль _____

Подпись _____ Дата _____

Проект для утверждения. Подписи не оформлены.

2026

К утверждению

КНВ.00002-01 33 01-ЛУ

KNV DemoBlocks

Библиотека функциональных блоков

Версия 0.1.0.0

Руководство программиста

КНВ.00002-01 33 01

Электронный документ PDF

Листов 12

Редакция 01 от 27.11.2026

Образец комплектной библиотечной поставки

Проект для утверждения

2026

АННОТАЦИЯ

Пример руководства программиста к библиотеке KNV DemoBlocks версии 0.1.0.0. Документ описывает два функциональных блока из проекта конвейера: FB_Motor и FB_BatchCounter. Приведены интерфейсы, алгоритмы, граничные случаи, подключение и ST-пример совместного применения.

Это образец комплектной поставки: файл .library, исходные тексты, PDF и результаты проверки в CODESYS V3.5 SP17 Patch 3. Логика блоков сохранена; к объявлениям добавлены комментарии и явные ссылки Standard.TON / Standard.R_TRIG.

Структура руководства принята по ГОСТ 19.504-79; вид документа - руководство программиста, код 33. Редакция 01 от 27.11.2026. Статус - проект для утверждения. Первый физический лист PDF является отдельным листом утверждения и не входит в нумерованный объём.

СОДЕРЖАНИЕ

Раздел	Лист
1 Назначение и условия применения; 2 Характеристика	3
3 Обращение к библиотеке	4
4 Входные и выходные данные. FB_Motor	5-6
4.2 FB_BatchCounter	7-8
5 Сообщения и диагностика	9
Приложение А. Пример совместного применения	10
Приложение Б. Проверки	11
Нормативные ссылки; регистрация изменений	12

1 НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ

KNV DemoBlocks - учебная библиотека повторно используемых блоков для циклической программы ПЛК. FB_Motor формирует команду привода и контролирует обратную связь. FB_BatchCounter считает фронты сигнала и выдаёт признак достижения задания. Визуализация, рецепты, журнал и текстовые аварии в библиотеку не входят.

Параметр поставки	**Значение**
Наименование / версия	KNV DemoBlocks / 0.1.0.0
Пространство имён	KNV
Поставщик в свойствах	Industrial Demo - условное имя демонстрации
Формат	KNV_DemoBlocks_0_1_0_0.library - исходная редактируемая библиотека
Зависимость	Standard 3.5.15.0 (System)
Проверенная среда	CODESYS V3.5 SP17 Patch 3 (3.5.17.30), 32 bit; локальная эмуляция

Вызов одного экземпляра выполнять один раз за цикл из одной задачи. В испытании использована задача 20 мс. Обмен с физическими I/O отсутствует: входы и выходы связывает вызывающее приложение. Потребление памяти отдельным экземпляром и максимальная частота сигналов на физическом ПЛК не измерялись.

2 ХАРАКТЕРИСТИКА БИБЛИОТЕКИ

Язык реализации - ST. Блоки сохраняют своё состояние между вызовами, но не объявляют RETAIN или PERSISTENT. Пропуск вызовов меняет обработку фронтов и таймера. Импульсы датчика должны гарантированно считываться задачей: необходимы различимые состояния FALSE и TRUE между принимаемыми фронтами.

FB_Motor принимает запрос уровня; механизма «Пуск/Стоп» с защёлкой внутри нет. FB_BatchCounter не выключает привод самостоятельно. За последовательность вызовов, останов по xDone и исключение самопуска отвечает программа приложения.

Блоки предназначены для примера промышленной логики; перенос на конкретный ПЛК требует проверки циклограммы, сигналов и блокировок оборудования. CODESYS 2.3 и физический контроллер для этой библиотеки не проверялись.

3 ОБРАЩЕНИЕ К БИБЛИОТЕКЕ

3.1 Подключение в CODESYS 3.5

1. В среде SP17 Patch 3 открыть «Инструменты → Репозиторий библиотек». Установить KNV_DemoBlocks_0_1_0_0.library. В списке должна появиться KNV DemoBlocks, версия 0.1.0.0, поставщик Industrial Demo.

2. В Application открыть менеджер библиотек и добавить этот выпуск. Убедиться, что зависимость Standard 3.5.15.0 доступна, а пространство имён подключённой библиотеки - KNV.

3. Объявить собственные экземпляры. Для каждого привода и каждого независимого счётчика нужен отдельный экземпляр. Вызовы разместить в одной циклической задаче.

3.2 Минимальный пример вызова

```
VAR
    fbM1 : KNV.FB_Motor;
    fbBatch : KNV.FB_BatchCounter;
END_VAR

fbM1(
    xRequest := xRunRequest,
    xPermit := xPermit,
    xDriveFault := xDriveFault,
    xFeedback := xFeedback,
    xReset := xReset,
    tStartTimeout := T#2s);

fbBatch(
    xPulse := xSensor,
    xEnable := fbM1.xRunning,
    xReset := xNewBatch,
    udiTarget := 25);
```

Здесь xRunRequest, xPermit, xDriveFault, xFeedback, xReset, xSensor и xNewBatch - BOOL-переменные вызывающей программы. Фрагмент показывает интерфейс; управление пуском и остановом нужно добавить в приложении. Полный пример приведён в приложении А.

После вызова доступны fbM1.xCommand, fbM1.xRunning, fbM1.xFault, fbBatch.udiCount, fbBatch.xDone и fbBatch.xAccepted. Значение xAccepted действительно только до следующего вызова данного экземпляра.

4 ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

4.1 FB_Motor - команда привода и контроль ОС

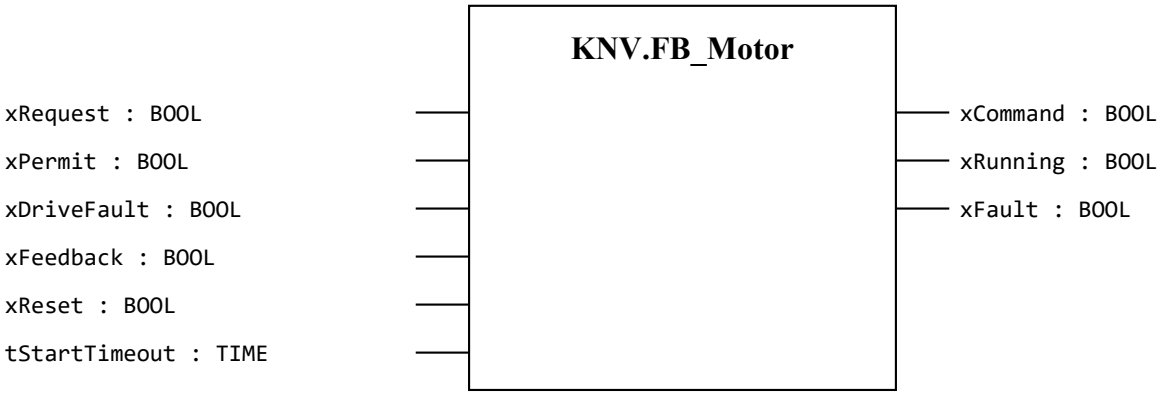


Рисунок 1 - Интерфейс функционального блока FB_Motor

Вход	**Тип / начальное значение**	**Назначение**
xRequest	BOOL / FALSE	Запрос движения уровня. TRUE требует команду; FALSE снимает её.
xPermit	BOOL / TRUE	Разрешение работы. FALSE немедленно снимает команду без фиксации аварии.
xDriveFault	BOOL / FALSE	Внешняя неисправность. TRUE фиксирует xFault даже при отсутствии запроса.
xFeedback	BOOL / FALSE	Подтверждение работы. Контролируется при действующей команде.
xReset	BOOL / FALSE	Сброс уровня; действует только при xRequest=FALSE и xDriveFault=FALSE.
tStartTimeout	TIME / T#2s	Допустимое непрерывное отсутствие ОС при команде. Задавать положительное время.

Выход	**Тип**	**Значение**
xCommand	BOOL	Команда после проверки запроса, разрешения и аварии.
xRunning	BOOL	Команда активна, есть ОС, аварии нет.
xFault	BOOL	Защёлка внешней неисправности или тайм-аута отсутствия ОС.

4.1 FB_Motor (продолжение)

4.1.1 Порядок выполнения

Сначала при разрешённых условиях xReset снимает защёлку xFault. Затем xDriveFault снова устанавливает её, если внешняя неисправность активна. Формируется команда: xRequest AND xPermit AND NOT xFault.

Таймер Standard.TON работает при xCommand=TRUE и xFeedback=FALSE. По достижении tStartTimeout фиксируется xFault, а команда снимается в этом же вызове. Выход xRunning равен xCommand AND xFeedback AND NOT xFault.

4.1.2 Характерные ситуации

Условие	Результат
Запрос и разрешение TRUE, ОС TRUE	xCommand=TRUE, xRunning=TRUE, если ранее не было аварии.
Команда есть, ОС непрерывно нет дольше выдержки	xFault фиксируется; команда и признак работы снимаются.
ОС пропала во время уже начавшейся работы	Действует тот же таймер. Это контроль не только пуска, но и потери ОС при работе.
xPermit стал FALSE	Команда снята; сама потеря разрешения не фиксирует xFault.
xPermit вернулся, запрос остался TRUE	При отсутствии xFault команда появится снова. Для запрета самопуска приложение должно заранее снять запрос.
Причина ушла, xReset=FALSE	Защёлка аварии остаётся; одного восстановления ОС недостаточно.
xReset=TRUE, но запрос или внешний отказ TRUE	Сброс не разрешён.
xReset удерживается; запрос и отказ стали FALSE	Защёлка очищается без нового фронта xReset. Сброс обрабатывается по уровню.

4.1.3 Ограничения

Не выдаёт код причины аварии и текст сообщения. Не контролирует неожиданную ОС при снятой команде. Не фильтрует дребезг. Не проверяет допустимость tStartTimeout: нулевой или отрицательный параметр не следует использовать. Фактическое время реакции зависит от цикла задачи и реализации стандартного TON.

4.2 FB_BatchCounter - счётчик партии

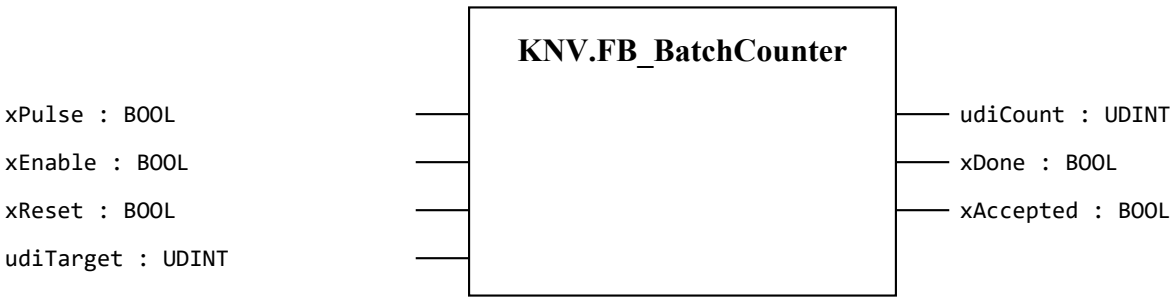


Рисунок 2 - Интерфейс функционального блока FB_BatchCounter

Вход	**Тип / начальное значение**	**Назначение**
xPulse	BOOL / FALSE	Учитывается передний фронт FALSE → TRUE. Длительный уровень не добавляет повторные изделия.
xEnable	BOOL / FALSE	Разрешение приёма нового фронта. Внутренний детектор фронта продолжает работу и при FALSE.
xReset	BOOL / FALSE	Обнуление счёта по уровню, с приоритетом перед счётом.
udiTarget	UDINT / 25	Задание партии. Значение 0 блокирует счёт и выдачу xDone. Держать постоянным в пределах партии.

Выход	**Тип**	**Значение**
udiCount	UDINT	Принятое число импульсов. Начальное значение 0.
xDone	BOOL	udiTarget > 0 и udiCount >= udiTarget. Не зависит от xEnable.
xAccepted	BOOL	TRUE ровно на один вызов, в котором принято изделие; при сбросе FALSE.

Диапазон UDINT: 0...4294967295. Защита от переполнения при постоянном задании обеспечивается условием udiCount < udiTarget перед приращением; после достижения задания счёт блокируется.

4.2 FB_BatchCounter (продолжение)

4.2.1 Обработка фронтов и приоритеты

В каждом вызове сначала выполняется Standard.R_TRIG. xAccepted сбрасывается в FALSE. При xReset=TRUE счёт обнуляется; иначе новый фронт принимается только при xEnable=TRUE и udiCount < udiTarget. После этого вычисляется xDone.

Вызов	xPulse	xEnable	xReset	udiCount	xAccepted
1	FALSE	TRUE	FALSE	0	FALSE
2	TRUE	TRUE	FALSE	1	TRUE
3	TRUE	TRUE	FALSE	1	FALSE
4	FALSE	TRUE	FALSE	1	FALSE
5	TRUE	TRUE	FALSE	2	TRUE

Таблица 1 - Пример при udiTarget=3 и исходном счётчике 0

4.2.2 Граничные случаи

Ситуация	Поведение
Фронт пришёл при xEnable=FALSE	Не учитывается и не ставится в очередь. Включение разрешения при удержанном TRUE не добавит изделие.
Одновременно сброс и фронт	Сброс имеет приоритет. Этот фронт уже обработан детектором и повторно не учитывается.
xReset удерживается TRUE	Счёт остаётся 0; xAccepted остаётся FALSE.
udiTarget=0	Счёт не увеличивается; xDone=FALSE. Старое значение счёта автоматически не обнуляется.
Задание уменьшено ниже текущего счёта	Счёт не урезается, xDone становится TRUE. Менять задание следует после сброса партии.
Первый вызов при xPulse=TRUE	Начальный переход детектора может быть принят за фронт. Перед разрешением счёта привести датчик к FALSE или выполнить управляемую инициализацию.

Для остановки по достижении задания приложение должно использовать xDone. Сам блок счётчика не управляет приводом и не выдаёт аварии отсутствия деталей.

5 СООБЩЕНИЯ И ДИАГНОСТИКА

Блоки не создают текстовые сообщения. Приложение интерпретирует логические выходы и формирует собственные коды и надписи. Показанные ниже тексты - пример сообщений верхнего уровня, а не встроенные строки библиотеки.

Признак	**Сообщение приложения / действие**
fbMotor.xFault=TRUE	«Неисправность привода или отсутствует обратная связь». Снять запрос, проверить сигнал отказа и ОС; затем подать сброс.
Сброс не снимает xFault	«Сброс не разрешён». Проверить xRequest=FALSE и xDriveFault=FALSE.
fbCounter.xDone=TRUE	«Партия завершена». Снять запрос движения и подготовить следующую партию.
fbCounter.xAccepted=TRUE	Событие приёма изделия. Увеличить суммарный счётчик один раз в этом же цикле при необходимости.
Счёт не растёт	Проверить xEnable, отсутствие xReset, положительное задание, наличие новых фронтов и период вызова.
Ошибка типа KNV.FB_Motor при компиляции	Проверить добавление библиотеки и её пространство имён в менеджере библиотек.
Не найден Standard.TON или Standard.R_TRIG	Проверить доступность Standard 3.5.15.0 и корректность зависимости.

5.1 Что входит в передаваемый комплект

Редактируемая .library; этот PDF; ST-исходники двух блоков; ST-пример PRG_Example; протокол проверок и контрольные суммы. Установка библиотеки в репозиторий и её добавление в приложение - две отдельные операции.

Для коммерческого выпуска дополнительно согласуются право использования и изменения, поставщик, номер версии и порядок поддержки. В данном образце используется условное имя Industrial Demo; условия коммерческого лицензирования не устанавливались.

PDF является внешним руководством. Комментарии добавлены в объявления блоков. Полная встроенная справка F1 / CHM и отдельный пакет установки в этом образце не создавались. Их можно формировать на следующем этапе средствами LibDoc.

ПРИЛОЖЕНИЕ А

Совместное применение с явным повторным пуском

Учебная обвязка: все BOOL-входы - переменные программы, без привязки к физическим I/O. Пуск принимается по фронту. При потере разрешения, останове, сбросе, аварии или завершении партии запрос снимается. После возврата разрешения нужен новый фронт ПУСК.

```
PROGRAM PRG_Example
VAR
    xStart, xStop, xReset, xNewBatch : BOOL;
    xPermit : BOOL := TRUE;
    xDriveFault, xFeedback, xSensor : BOOL;
    xRunRequest, xDriveCommand : BOOL;
    udiTarget : UDINT := 25;
    fbMotor : KNV.FB_Motor;
    fbCounter : KNV.FB_BatchCounter;
    rtStart : Standard.R_TRIG;
END_VAR

rtStart(CLK := xStart);

IF xStop OR xReset OR NOT xPermit OR xDriveFault
    OR fbMotor.xFault OR fbCounter.xDone THEN
    xRunRequest := FALSE;
END_IF;

IF rtStart.Q AND NOT xStop AND NOT xReset
    AND NOT xNewBatch AND xPermit AND NOT xDriveFault
    AND NOT fbMotor.xFault AND NOT fbCounter.xDone THEN
    xRunRequest := TRUE;
END_IF;

fbMotor(
    xRequest := xRunRequest,
    xPermit := xPermit,
    xDriveFault := xDriveFault,
    xFeedback := xFeedback,
    xReset := xReset,
    tStartTimeout := T#2s);

fbCounter(
    xPulse := xSensor,
    xEnable := fbMotor.xRunning AND xRunRequest,
    xReset := xNewBatch AND NOT xRunRequest,
    udiTarget := udiTarget);

IF fbCounter.xDone OR fbMotor.xFault THEN
    xRunRequest := FALSE;
END_IF;

xDriveCommand := fbMotor.xCommand AND xRunRequest;
```

ПРИЛОЖЕНИЕ Б

Проверка примера библиотеки

Проверено в локальном эмуляторе CODESYS SP17 Patch 3: 20 из 20 проверок пройдены. Физический ПЛК не подключался. Проверки выполнялись через экземпляры KNV.FB_Motor и KNV.FB_BatchCounter, подключённые из .library.

№	Проверка	Итог
1	Подключение KNV и компиляция потребителя без ошибок и предупреждений	Пройдено
2	Исходное состояние привода без команды и аварии	Пройдено
3	Тайм-аут ОС фиксирует аварию и снимает команду	Пройдено
4	Сброс тайм-аута при снятом запросе	Пройдено
5	Работа при запросе, разрешении и ОС	Пройдено
6	Потеря разрешения без аварийной защёлки	Пройдено
7	Возврат команды при удержанном запросе и возврате разрешения	Пройдено
8	Фиксация внешнего отказа	Пройдено
9	Запрет сброса при активном внешнем отказе	Пройдено
10	Обработка удержанного сброса по уровню	Пройдено
11	Контроль потери ОС во время работы	Пройдено
12	Запрет сброса при активном запросе	Пройдено
13	Приоритет сброса счётчика над фронтом	Пройдено
14	Отсутствие повторного фронта после сброса при TRUE	Пройдено
15	Один учёт при длительном TRUE	Пройдено
16	Длительность xAccepted - один цикл	Пройдено
17	Нет накопления фронтов при запрете счёта	Пройдено
18	Ограничение счёта заданием 3	Пройдено
19	Нулевое задание не считает и не завершает	Пройдено
20	Уменьшение задания не урезает текущий счёт	Пройдено

Исходный протокол - sample_library_tests.json. Пример PRG_Example включён в проект-потребитель и проверен компиляцией; отдельные проверки его последовательности команд в этот протокол не входят.

НОРМАТИВНЫЕ И ТЕХНИЧЕСКИЕ ССЫЛКИ

1. ГОСТ 19.504-79. ЕСПД. Руководство программиста. Требования к содержанию и оформлению. Структура разделов и примеры справочных карточек.

2. ГОСТ 19.101-2024, ГОСТ 19.104-78, ГОСТ 19.105-78, ГОСТ 19.106-78. Виды программных документов, титульная часть, структура и печатное представление. Актуальность нормативной базы проверена 27.11.2026.

3. CODESYS: Library Documentation. Документирование интерфейсов, комментарии и дальнейшая подготовка справки LibDoc.

4. CODESYS: Library Manager. Подключение библиотек и пространство имён. Дата обращения к электронным источникам: 27.11.2026.

Временное внутреннее обозначение руководства - КНВ.00002-01 33 01. До утверждения согласовать код разработчика, регистрационные реквизиты по ГОСТ 19.103-77 и подписи сторон. Внешний нормоконтроль и утверждение документа не выполнялись.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Редакция 01, 27.11.2026. Первичный образец документации к KNV DemoBlocks 0.1.0.0.

Изм.	Листы / основание изменения	Подпись	Дата

Версию библиотеки изменять при корректировке интерфейса или поведения; руководство, примеры и проверки обновлять одновременно. Хранить архив предыдущего выпуска.