УДК 004.942

Рыбалёв Андрей Николаевич
Амурский государственный университет
г. Благовещенск, Россия
E-mail: amgu_appe@mail.ru
Rybalev Andrey Nikolaevich
Amur State University
Blagoveshchensk, Russia
E-mail: amgu_appe@mail.ru

ПРОЕКТИРОВАНИЕ СИСТЕМ УПРАВЛЕНИЯ ДИСКРЕТНЫМИ ПРОЦЕССАМИ НА ОСНОВЕ ИМИТАЦИОННЫХ МОДЕЛЕЙ

DISCRETE PROCESS CONTROL SYSTEMS DESIGNING BASED ON SIMULATION MODELS

Аннотация. Рассмотрен подход к разработке программного и частично аппаратного обеспечения для систем управления дискретными процессами, основанный на использовании имитационных моделей. Предложенная методика может быть применена на практике и при обучении.

Abstract. The article considers an approach to developing software and partially hardware for discrete process control systems based on the use of simulation models. The proposed methodology can be applied in practice and in training.

Ключевые слова: программируемый логический контроллер, имитационные модели, межпрограммный обмен, конечный автомат.

Key words: programmable logic controller, simulation models, interprogram exchange, finite state machine.

Введение

Проектирование любой системы управления начинается с разработки концептуальной модели, которая определяет режимы работы системы, задачи, решаемые системой в этих режимах, способы взаимодействия с человеком-оператором и т.д. В случае небольших локальных объектов целесообразно такую модель «оформить» не в виде текстового документа, а в виде компьютерной имитационной системы управления [1]. Такие системы дают наглядное представление о принятых решениях, поскольку позволяют оценить их в ходе «интерактивного общения». Дополнительным и очень ценным преимуществом имитационных систем является возможность проектирования прототипов электрических схем, программного кода, элементов человеко-машинного интерфейса без привязки к конкретным техническим устройствам. В то же время разработка моделей может проводиться с привлечением промышленных программных комплексов, которые, возможно, в будущем будут задействованы в реальных системах. В таком случае мы получаем почти готовое программное обеспечение на ранних стадиях проектирования.

Создание имитационных моделей систем управления является незаменимой частью

процесса подготовки специалистов по автоматизации производства. Выполняя проекты подобного рода, студенты получают навыки моделирования объекта управления, разработки программ для управляющих машин и операторских станций. Главное состоит в том, что система «реально» работает и выполняет свои функции во взаимодействии с человеком. Студенты АмГУ, обучающиеся по направлению 15.03.04 «Автоматизация технологических процессов и производств» в рамках дисциплины «Программное обеспечение систем управления» выполняют соответствующие индивидуальные задания. Одновременно в курсовой работе по дисциплине «Средства автоматизации и управления» разрабатывается проект технической реализации системы.

Рассмотрим один из возможных подходов к построению имитационной системы управления дискретным технологическим процессом, уже применяемый в учебном процессе. В качестве примера взята достаточно простая модель, построение которой используется в качестве образца.

Объект управления

Объектом управления является система дозирования сыпучего материала, состоящая из подающего и разгрузочного транспортеров и мерного бункера (рис. 1).

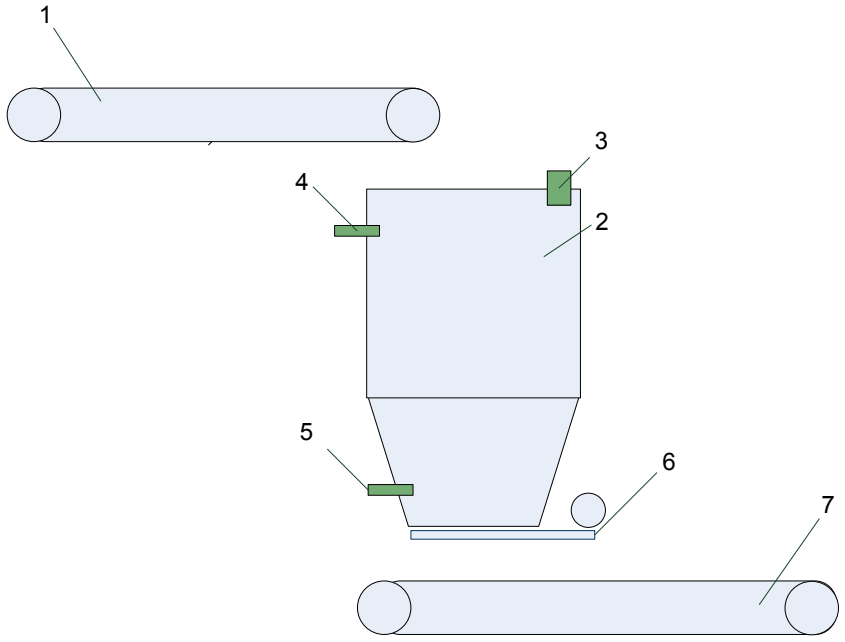

Рис. 1. Схема установки.

Обозначения к рис. 1: 1 – подающий транспортер, оснащенный датчиком движения ленты; 2 – мерный бункер; 3 – измерительный преобразователь уровня (например, ультразвуковой или микроволновый радарный); 4, 5 – контактные датчики верхнего и нижнего уровней, формирующие сигналы «бункер полон» и «бункер пуст»; 6 – шиберная задвижка с электроприводом, привод снабжен концевыми выключателями, срабатывающими в положениях «задвижка закрыта» и «задвижка открыта»; 7 – разгрузочный транспортер, оснащенный датчиком движения ленты.

Система должна допускать ручное управление всеми механизмами с защитой необходимыми блокировками от неправильного включения, а также автоматическое управление с возможностью задания отпускаемого объема/уровня в мерном баке.

Ручное управление производится с помощью органов щита (лицевой панели электротехнического шкафа). Шкаф установлен рядом с объектом управления, поэтому оператор может непосредственно наблюдать за ходом процесса.

Автоматическое управление осуществляет программируемый логический контроллер (ПЛК). Промышленной сетью контроллер связан с устройством человеко-машинного интерфейса (например, панелью оператора или компьютером), находящимся в помещении, отдаленном от объекта. Поэтому программа контроллера, помимо решения задачи управления, должна выявлять нештатные ситуации в процессе и аварии оборудования и сообщать о них оператору.

Структура имитационной системы

На рис. 2 показана структурная схема имитационной модели. Модель строится на базе двух программных единиц: Simulink-модели объекта управления (MathWorks® Matlab) и SoftLogic пакета CODESYS 3.5 (CODESYS Group®), в котором реализуется вся «управляющая часть».

Рис. 2. Структура имитационной системы.

«Реальная» часть этой подсистемы содержит программу ПЛК и экран визуализации, которые действительно будут присутствовать в системе управления, если она будет воплощена в жизнь (после доработки/переработки). В «виртуальной» части находятся программные модели электрической схемы, щита управления и экран ввода неисправностей оборудования.

Взаимодействие частей внутри программы CODESYS производится через общие переменные, обмен сигналами с Simulink-моделью – по технологии OPC DA [2].

Модель объекта управления

Simulink-диаграмма модели показана на рис. 3.

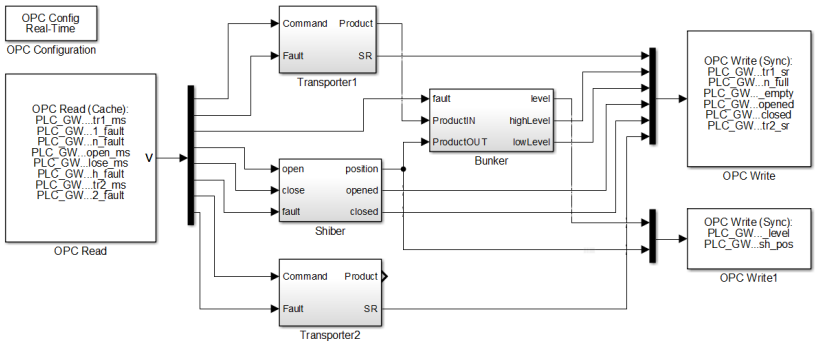

Рис. 3. Simulink-модель объекта управления.

Объект представлен четырьмя подсистемами, имитирующими транспортеры (два одинаковых блока), мерный бункер и его шибер.

На входы блоков из подсистемы управления приходят управляющие сигналы (включить/ выключить), формируемые «магнитными пускателями» «электрической схемы», и сигналы ввода ошибок от виртуального экрана ввода неисправностей (перечень переменных приведен в приложении 1). Блок, имитирующий бункер, получает сигналы о расходе входящего и выходящего продукта (0-100%).

Блоки выдают сигналы датчиков движения лент транспортеров, концевых выключателей шибера, датчиков предельных уровней в бункере, а также непрерывные сигналы по уровню в бункере и положению шибера (последний сигнал в реальной системе управления не используется). Блоки, имитирующие транспортеры, помимо этого, имеют выходы, информирующие о расходе продукта.

Подсистемы реализованы в упрощенном виде (рис. 4). Во всех подсистемах поступление сигнала об ошибке приводит к переключению функциональных блоков Switch, в результате на входы передаточной функции и интеграторов подаются нулевые сигналы.

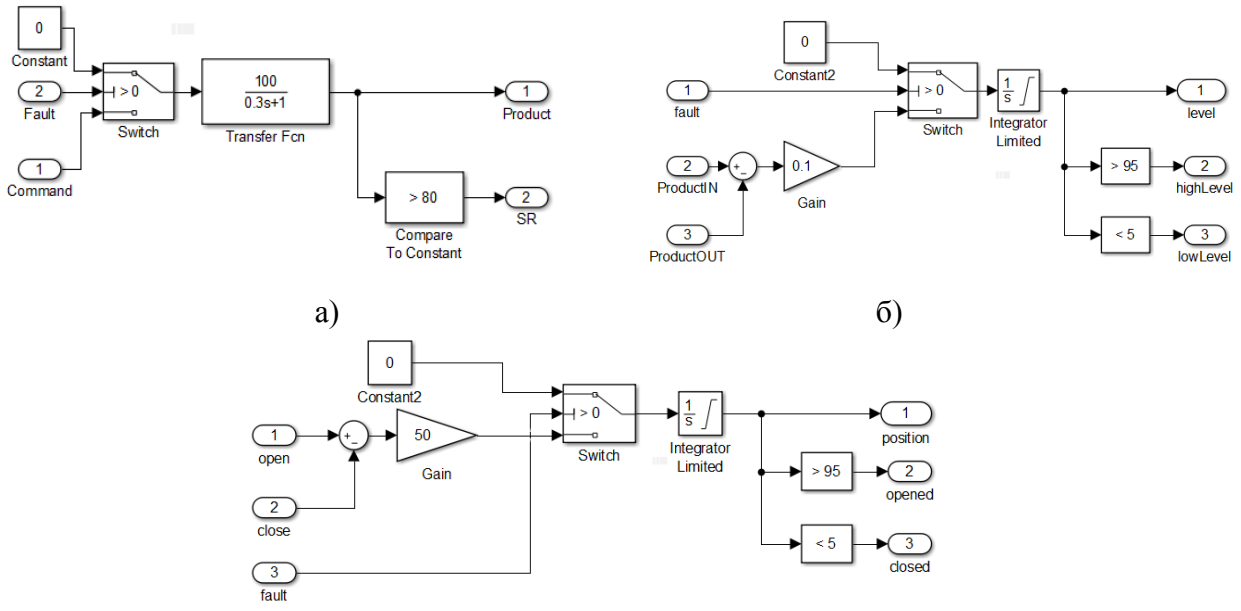

Рис. 4. Блоки модели Transporter (а), Bunker (б), Shiber (в).

Имитация электрической схемы

Программная модель электрической схемы (программа Electric), составленная на языке LD (Ladder Logic), показана на рис. 5. Она действительно похожа на релейно-контактную схему, поскольку состоит из «контактов» и «катушек». Ее цель – сформировать сигналы срабатывания «магнитных пускателей» транспортеров и привода шибера, а также «реле» предупредительной и аварийной сигнализации (катушки). В цепях находятся:

контакты «кнопок» щита управления (*_bt);

контакты переключателя в автоматический режим управления (auto), также находящегося на щите;

контакты датчиков на объекте (реле движения транспортеров (*_sr), верхнего предельного уровня в бункере (bun_full), концевых выключателей шибера);

выходные контакты ПЛК (*_on);

контакты самих катушек.

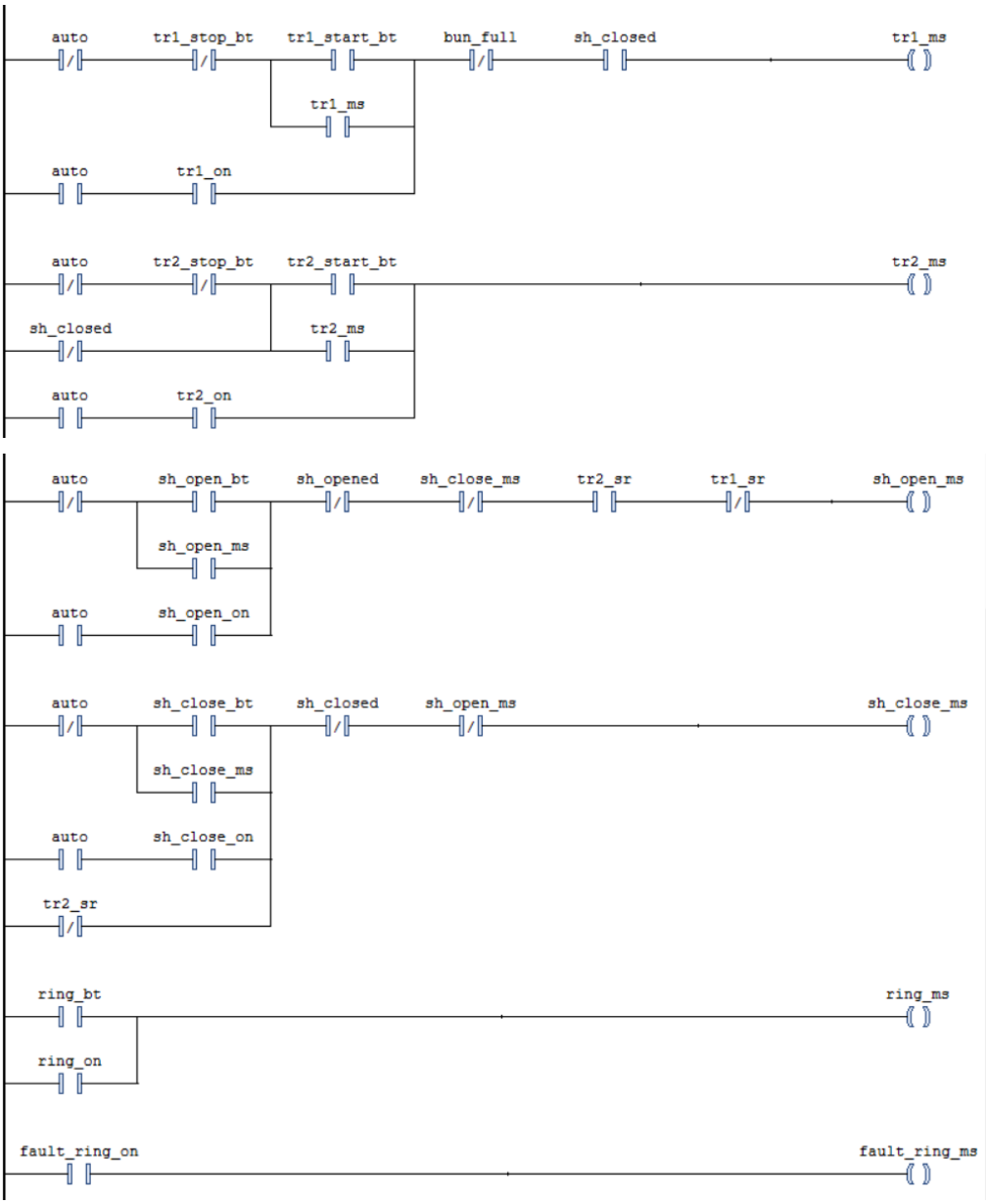

Рис. 5. Программная модель электрической схемы.

Схема в основном реализует простейшие цепи управления, дополненные следующими

блокировками: запрет движения подающего транспортера при заполненном бункере; запрет открытия шибера при запущенном подающем транспортере и неработающем разгрузочном; запрет одновременного включения «пускателей» открытия и закрытия шибера; принудительное закрытие шибера при неработающем разгрузочном транспортере.

Для упрощения схема не реализует так называемый ремонтный режим, при котором большая часть блокировок может быть отключена.

Имитация человеко-машинного интерфейса

В системе присутствуют три экрана визуализации: виртуальный экран щита управления, экран панели оператора и виртуальный экран ввода неисправностей (рис. 6).

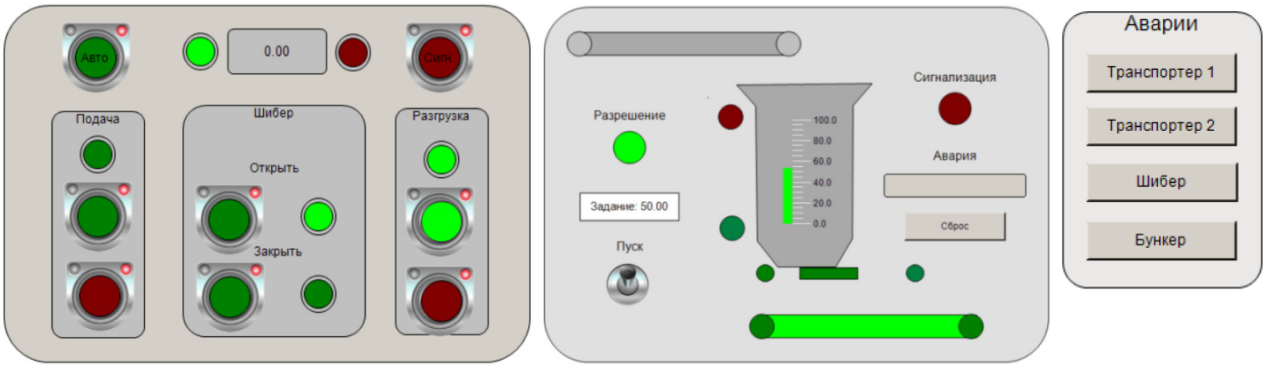

Рис. 6. Экраны визуализации.

На «щите» присутствуют кнопки запуска «пускателей» в ручном режиме, соответствующие сигнальные индикаторы, коммутируемые датчиками, кнопка/переключатель (с фиксацией) перевода в автоматический режим, кнопка включения предупредительной сигнализации перед запуском транспортеров, а также цифровой индикатор уровня и дискретные индикаторы его предельных значений. Кнопки типа «Пуск» подсвечиваются через контакты катушек.

Экран панели оператора содержит схему процесса, изменяющую цвета элементов при срабатывании «пускателей» и датчиков, индикатор «Разрешение», сигнализирующий о нахождении в автоматическом режиме, элемент ввода задания по уровню, выключатель запуска процесса, индикатор сигнализации, табло текстовых сообщений об ошибках и кнопку сброса системы после устранения ошибки.

Экран ввода неисправностей состоит из четырех кнопок с фиксацией, непосредственно устанавливающих значения соответствующих переменных.

Программа ПЛК

Программа автоматического управления auto_prog составлена на языке ST (Structured Text), ее текст приведен в приложении 2. Она представляет собой конечный автомат [3], построенный на программной конструкции switch-case (в языке ST – case of-end_case), дополненный «схемой» обнаружения неисправностей. Состояния автомата: «ручное управление» (перевод всех выходов контроллера в состояние «выключено»); «ожидание» (включения выключателя на панели оператора); «загрузка» (бункера); «разгрузка» (бункера); «авария».

Переход из одного «нормального» состояния в другое производится по командам и сигналам датчиков и показаниям измерительного преобразователя уровня, в состояние «авария» – по сигналам специальных таймеров, отслеживающих время запуска механизмов.

При отсутствии аварий программа выполняет циклическое дозирование заданного количества сыпучего материала. Задание можно поменять в любой момент, но при этом оно вступит в действие только в новом цикле. Выключение процесса также осуществляется после окончания цикла.

Главная программа PLC_PRG в имитационной системе по очереди вызывает программы auto_prog и Electric.

В реальной системе в ПЛК останется только программа auto_prog, набор глобальных переменных будет соответствующим образом сокращен (часть переменных потребуется перенести конфигурацию ввода-вывода), а список визуализаций будет включать только экран оператора панели.

Заключение

Рассмотренные подход и последовательность действий по созданию имитационной системы управления дают возможность отладки каждого элемента на соответствующем этапе разработки. Модель объекта управления независимо апробируется путем ручного задания командных сигналов, передаваемых по OPC, и анализа откликов. Модель электрической схемы коммутации также может быть отлажена в ручном режиме, после чего она становится под управление органам визуализации щита. Наблюдать за процессом удобнее с помощью специального экрана визуализации, поэтому работа по его созданию проводится параллельно с остальными действиями. По мере развития системы повышается и функциональность экрана. В дальнейшем он превратится в основное средство человеко-машинного интерфейса. Заключительный этап – разработка и отладка программы автоматического управления. Наиболее подходящей технологией разработки программы для дискретных технологических процессов является технология конечных автоматов.

---

1 Рыбалев, А.Н. Имитационное моделирование АСУ ТП – Благовещенск: Амурский государственный университет, 2019. – 408 с.

2. Wolfgang Mahnke, Stefan-Helmut Leitner, Matthias Damm OPC Unified Architecture // Springer Science & Business Media, 2009. – 339 p.

3. Лупал, А.М. Теория автоматов: учеб. пособие – СПб.: СПбГУА, 2000. – 119 с.

Приложение 1.
Глобальные переменные проекта

```
VAR_GLOBAL
(*ЭЛЕМЕНТЫ ЭЕКТРИЧЕСКОЙ СХЕМЫ*)
tr1_ms, tr2_ms:BOOL; //пускатели транспортеров
sh_open_ms,sh_close_ms:BOOL; //пускатели шибера
ring_ms, fault_ring_ms:BOOL; //реле сигнализации

tr1_sr, tr2_sr:BOOL; //контроль скорости транспортеров,
bun_full, bun_empty:BOOL; //датчики бункер полон/пуст, входы контроллера
sh_closed, sh_opened:BOOL; //датчики шибер закрыт/открыт,входы контроллера

tr1_stop_bt, tr1_start_bt: BOOL; //кнопки управления первым транспортером
tr2_stop_bt, tr2_start_bt: BOOL; //кнопки управления вторым транспортером
sh_open_bt,  sh_close_bt: BOOL; //кнопки управления шибером
ring_bt:BOOL; //кнопка управления сигнализацией
auto:BOOL; //переключатель "Автоматическое управление", вход контроллера

bun_level:REAL; //уровень  в бункере, вход контроллера
(*Выходы ПЛК*)
tr1_on, tr2_on, sh_open_on, sh_close_on, ring_on, fault_ring_on:BOOL;

(*СИГНАЛЫ ПАНЕЛИ*)
ref:REAL; // задание по уровню
start_process:BOOL; // запуск процесса
reset:BOOL; //сброс системы после ошибок
fault_text:STRING(20); //строка с текстом сообщения

 (*ФИКТИВНЫЕ СИГНАЛЫ*)
tr1_fault,tr2_fault,sh_fault, bun_fault:BOOL; //сигналы ошибок элементов
sh_pos: REAL; //степень открытия шибера
END_VAR
```

Приложение 2.
Программа автоматического управления

```
PROGRAM auto_prog
VAR
        state:BYTE;
        high_level, low_level:REAL;
        timer_ps:TON;
        timer_tr1: TON;
        timer_tr2: TON;
        timer_sh: TON;
        timer_bun: TON;
END_VAR

CASE state OF
0: (*ручное управление*)
        tr1_on:=tr2_on:=sh_open_on:=sh_close_on:=FALSE;
```

```
        fault_ring_on:=FALSE;
        fault_text:='';
        IF auto THEN
                start_process:=FALSE;
                state:=1;
        END_IF
1: (*ожидание*)
        tr1_on:=tr2_on:=sh_open_on:=FALSE;
        sh_close_on:=NOT sh_closed;
        fault_ring_on:=FALSE;
        fault_text:='';
        IF start_process THEN
                IF ref > bun_level THEN
                        high_level:=ref; low_level:=0;
                        state:=2;
                ELSE
                        high_level:=bun_level; low_level:=bun_level - ref;
                        state:=3;
                END_IF
        END_IF
        IF NOT auto THEN
                state:=0;
        END_IF
2: (*загрузка*)
        timer_ps(IN:=sh_closed, PT:=T#2S);
        ring_on:= NOT timer_ps.Q AND sh_closed;
        tr1_on:=sh_closed AND bun_level < high_level AND timer_ps.Q;
        tr2_on:=FALSE;
        sh_open_on:=FALSE;
        sh_close_on:= NOT sh_closed;
        fault_ring_on:=FALSE;
        fault_text:='';
        IF NOT auto THEN
                timer_ps(IN:=FALSE);
                state:=0;
        ELSIF bun_level > high_level THEN
                timer_ps(IN:=FALSE);
                state:=3;
        END_IF
3: (*разгрузка*)
        timer_ps(IN:=TRUE, PT:=t#2s);
        ring_on:= NOT timer_ps.Q;
        tr1_on:=FALSE;
        tr2_on:=timer_ps.Q;
        sh_open_on:= tr2_sr;
        sh_close_on:=FALSE;
```

```
        fault_ring_on:=FALSE;
        fault_text:='';
        IF NOT auto THEN
                timer_ps(IN:=FALSE);
                state:=0;
        ELSIF bun_level <= low_level THEN
                timer_ps(IN:=FALSE);
                state:=1;
        END_IF
4: (*АВАРИЯ*)
        tr1_on:=tr2_on:=sh_open_on:=sh_close_on:=FALSE;
        fault_ring_on:=TRUE;
        IF reset THEN
                state:=0;
        END_IF
END_CASE

(*Контроль загрузочного транспортера*)
timer_tr1(IN:=tr1_ms AND NOT tr1_sr,PT:= T#3S);
IF timer_tr1.Q THEN
        state:=4;
        fault_text:='Транспортер1';
END_IF

(*Контроль разгрузочного транспортера*)
timer_tr2(IN:=tr2_ms AND NOT tr2_sr,PT:= T#3S);
IF timer_tr2.Q THEN
        state:=4;
        fault_text:='Транспортер2';
END_IF

(*Контроль движения шибера*)
timer_sh(IN:=(sh_open_ms AND NOT sh_opened) OR
  (sh_close_ms AND NOT sh_closed),PT:= T#5S);
IF timer_sh.Q THEN
        state:=4;
        fault_text:='Шибер';
END_IF

(*Контроль бункера*)
timer_bun(IN:=auto AND (tr1_sr OR NOT sh_closed),PT:= T#10S);
IF timer_bun.Q THEN
        state:=4;
        fault_text:='Бункер';
END_IF
```