**Лабораторная работа № 1**

**Реализация алгоритма управления нагревателем и насосом для контроллера ПЛК-160 на языке функциональных блоков CFC пакета Codesys v2.3**

**Цель работы**

Изучение принципов и реализация составления прикладных программ для промышленных логических контроллеров (ПЛК) на языке CFC пакета Codesys v2.3.

**Теоретическая часть**

Основой комплекса CODESYS является среда разработки прикладных программ для [программируемых логических контроллеров](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BB%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D0%BB%D0%B5%D1%80). Она распространяется бесплатно и может быть без ограничений установлена на нескольких рабочих местах.

В CODESYS для программирования доступны все пять определяемых стандартом [IEC 61131-3](https://ru.wikipedia.org/wiki/IEC61131-3) (МЭК 61131-3) языков:

– [*IL*](https://ru.wikipedia.org/wiki/Instruction_List) (*Instruction List*) – ассемблер-подобный язык;

– [*ST*](https://ru.wikipedia.org/wiki/Structured_Text) (*Structured Text*) – Pascal-подобный язык;

– [*LD*](https://ru.wikipedia.org/wiki/Ladder_Diagram) (*Ladder Diagram*) – язык релейных схем;

– [*FBD*](https://ru.wikipedia.org/wiki/FBD) (*Function Block Diagram*) – язык функциональных блоков;

– [*SFC*](https://ru.wikipedia.org/wiki/SFC) (*Sequential Function Chart*) – язык диаграмм состояний.

В дополнение к [*FBD*](https://ru.wikipedia.org/wiki/FBD) поддержан язык программирования *CFC*

(*Continuous Function Chart*) с произвольным размещением блоков и расстановкой порядка их выполнения.

В CODESYS реализован ряд других расширений спецификации стандарта [IEC 61131-3](https://ru.wikipedia.org/wiki/IEC61131-3). Самой существенной из них является поддержка [объектно-ориентированного программирования (ООП)](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5). Встроенные компиляторы CODESYS генерируют [машинный код](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) (двоичный код), который загружается в контроллер. Поддерживаются основные 16- и 32-разрядные [процессоры](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BD%D1%82%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80): [Infineon C166, TriCore](https://ru.wikipedia.org/wiki/Infineon_Technologies), [80×86](https://ru.wikipedia.org/wiki/X86), [ARM (архитектура)](https://ru.wikipedia.org/wiki/ARM_(%D0%B0%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0)), [PowerPC](https://ru.wikipedia.org/wiki/PowerPC" \o "PowerPC), [SH](https://ru.wikipedia.org/wiki/SuperH), [MIPS (архитектура)](https://ru.wikipedia.org/wiki/MIPS_(%D0%B0%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0)), [Analog Devices Blackfin](https://ru.wikipedia.org/wiki/Analog_Devices" \o "Analog Devices), [TI C2000/28x](https://ru.wikipedia.org/wiki/Texas_Instruments) и другие. При подключении к контроллеру среда программирования переходит в [режим отладки](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BB%D0%B0%D0%B4%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D1%8B). В нем доступны мониторинг/изменение/фиксация значений переменных, [точки останова](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D1%87%D0%BA%D0%B0_%D0%BE%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%B0), [контроль потока выполнения](https://ru.wikipedia.org/w/index.php?title=%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C_%D0%BF%D0%BE%D1%82%D0%BE%D0%BA%D0%B0_%D0%B2%D1%8B%D0%BF%D0%BE%D0%BB%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F&action=edit&redlink=1), горячее обновление кода, графическая трассировка в [реальном времени](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%B2%D1%80%D0%B5%D0%BC%D1%8F) и другие отладочные инструменты.

CODESYS версии v.2.3 построен на базе так называемой платформы автоматизации: CODESYS Automation Platform. Она позволяет изготовителям оборудования развивать комплекс путём подключения собственных [плагинов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%B3%D0%B8%D0%BD).

Расширенная профессиональная версия среды разработки носит название CODESYS Professional Developer Edition. Она включает поддержу [*UML*](https://ru.wikipedia.org/wiki/UML)-диаграмм (*Unified Modeling Language* – унифицированный язык моделирования) классов и состояний, подключение [системы контроля версий](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B9) [Subversion](https://ru.wikipedia.org/wiki/Subversion" \o "Subversion), [статический анализатор](https://ru.wikipedia.org/w/index.php?title=%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%82%D0%BE%D1%80_%D0%BA%D0%BE%D0%B4%D0%B0&action=edit&redlink=1) и [профилировщик](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%84%D0%B8%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) кода. Распространяется по лицензии.

Инструмент CODESYS Application Composer позволяет перейти от программирования практических приложений к их быстрому составлению. Пользователь продумывает собственную базу объектов, соответствующих определенным приборам, механическим узлам машины и т. п. Каждый объект включает программную реализацию и визуальное представление. Законченное приложение составляется из необходимых объектов, конфигурируется и автоматически генерируется программа на языках МЭК 61131-3.

Для программирования контроллера в среде CODESYS в него должна быть встроена система исполнения (Control Runtime System). Она устанавливается в контроллер в процессе его изготовления. Существует специальный инструмент ([Software development kit](https://ru.wikipedia.org/wiki/Software_development_kit" \o "Software development kit)), позволяющий адаптировать её к различным аппаратным и программным платформам.

**Рабочее Задание**

1. Ознакомьтесь с программным комплексом CoDeSys, с этапами работы в нем, рассмотрите устройство ПЛК, его архитектуру и типы.

2. Ознакомьтесь с языком функциональных блоков CFC.

3. По методике, приведенной ниже, создайте алгоритм простейшей программы на языке функциональных блоков CFC.

4. Реализуйте на основе созданного алгоритма логику управления нагревателем и насосом для контроллера ПЛК-160 на языке CFC пакета Codesys 2.3.

5. Сделайте отчет о выполненном задании.

**Методика проведения работы**

1. Разберите пример.

Имеется водонагревательный котел и насос (рис. 1), который прокачивает воду через него, а также кнопки «Пуск» - на запуск работы схемы и «Стоп» - на остановку работы схемы. Когда оператор нажимает на кнопку «Пуск» сначала в системе включается насос, котел подключается после с задержкой 3 сек. Соответственно, когда оператор нажимает кнопку «Стоп» котел выключается сразу, а насос некоторое время продолжает работу – 5 сек. После этого выключается. Итак, насос включается сразу, а котел с задержкой, при отключении котел отключается сразу, а насос с задержкой.

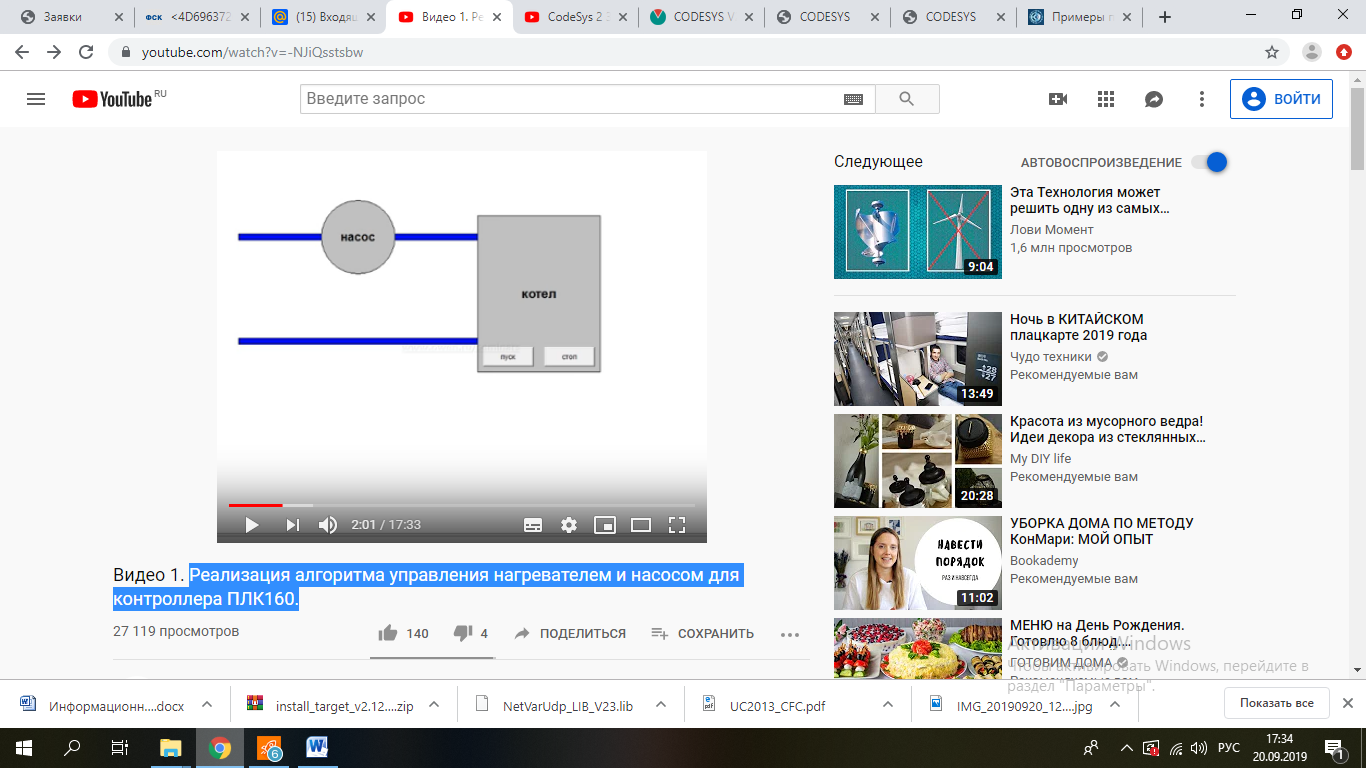
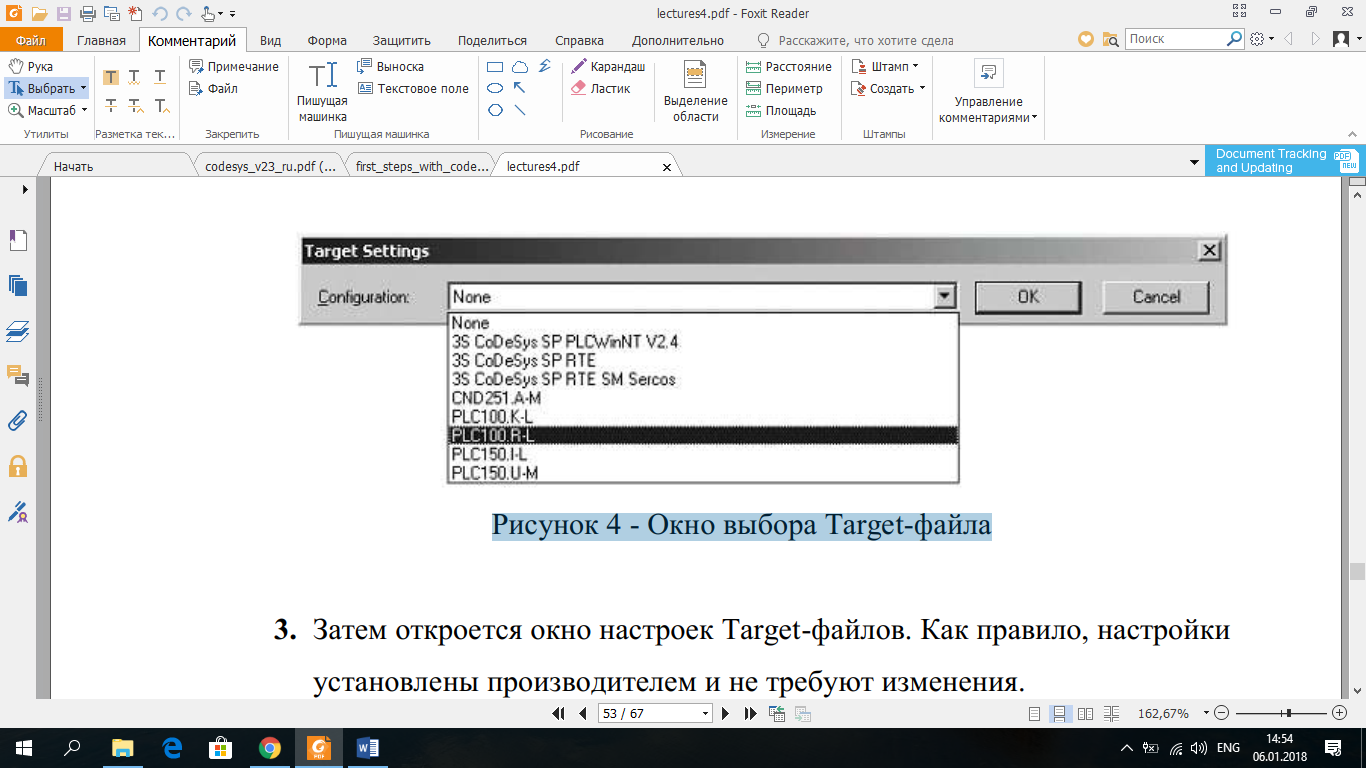


Рис. 1.1. Схема технологической операции

Создайте алгоритм с помощью языка функциональных блоков – CFC.Для этого запустите систему Codesys 2.3. Создайте проект и выберите нужную модификацию контроллера Овен (если ее нет скачайте таргет-файл нужной модификации). В данном проекте выберите PLC 160M из списка. После создания проекта сохраните его.

*1. Для создания нового проекта в среде CoDeSys вызовите  
команду меню File/New или воспользуйтесь одноименной кнопкой на  
панели инструментов.*

*2. После создания проекта выберите Target-файл, соответствующий названию контроллера. Окно выбора Target-файла представлено на рис. 1.2.*

**

*Рис. 1.2. Окно выбора Target-файла*

*3. Затем откройти окно настроек Target-файлов. Настройки  
установлены производителем и не требуют изменения.  
 4. После задания настроек Target-файла создайте основной  
POU (главную программу проекта). Она всегда должна иметь тип Program и имя PLC\_PRG. Поэтому в данном диалоге разрешается менять только язык программирования (Language of the POU).*

1. Необходимо определиться, какие переменные, какие элементы внутри алгоритма вы будете использовать с точки зрения условия задачи.

Создание переменных происходит в верхнем окне программы, которая называется – область объявления переменных (рис. 1.3).

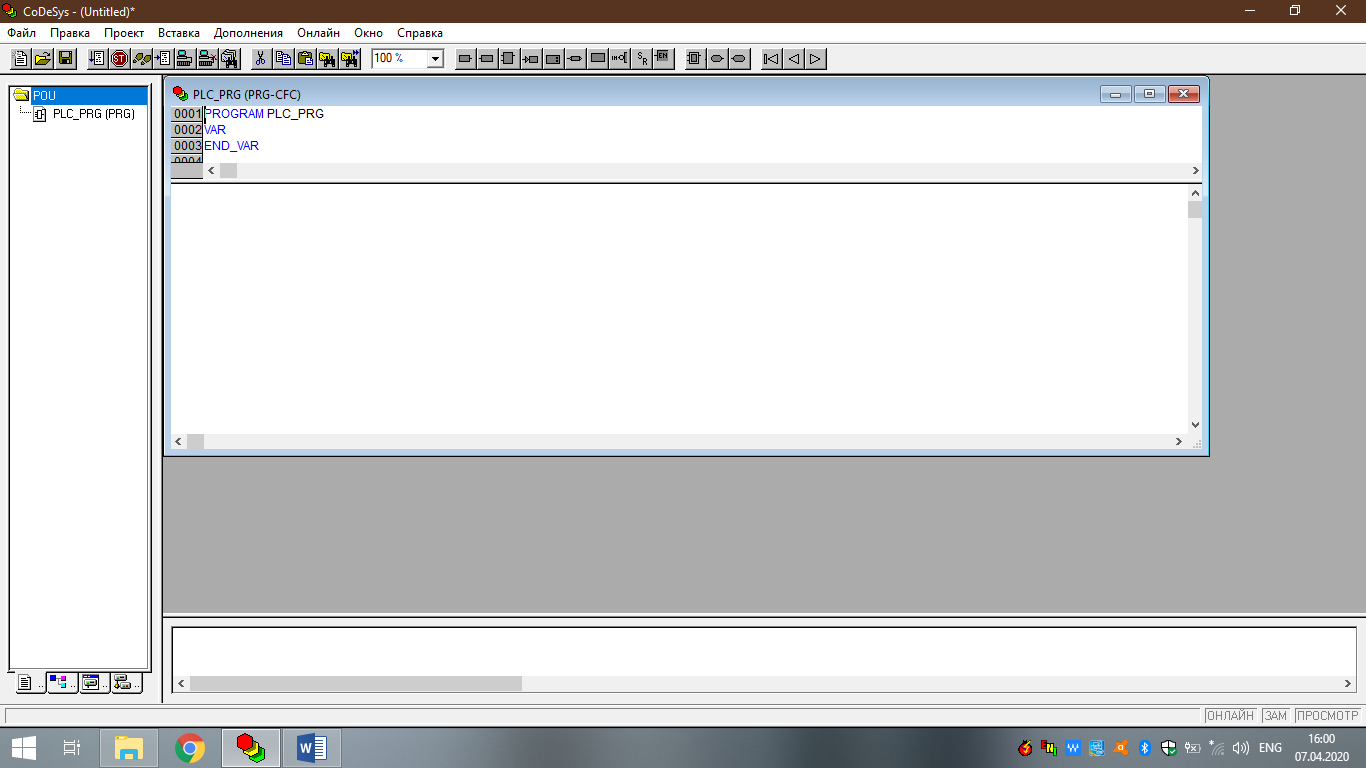


Рис. 1.3. Окно создания программы

1. Между ключевыми словами VAD и END\_VAR последовательно запишите все те сигналы, необходимые, чтоб реализовать алгоритм работы. Это, во-первых, кнопки «ПУСК» и «СТОП». С именами, соответственно, «pusk» и «stop».

Еще два элемента – это насос и котел: «nasos» и «kotel», которые мы будем включать и отключать в программе (рис. 1.4).

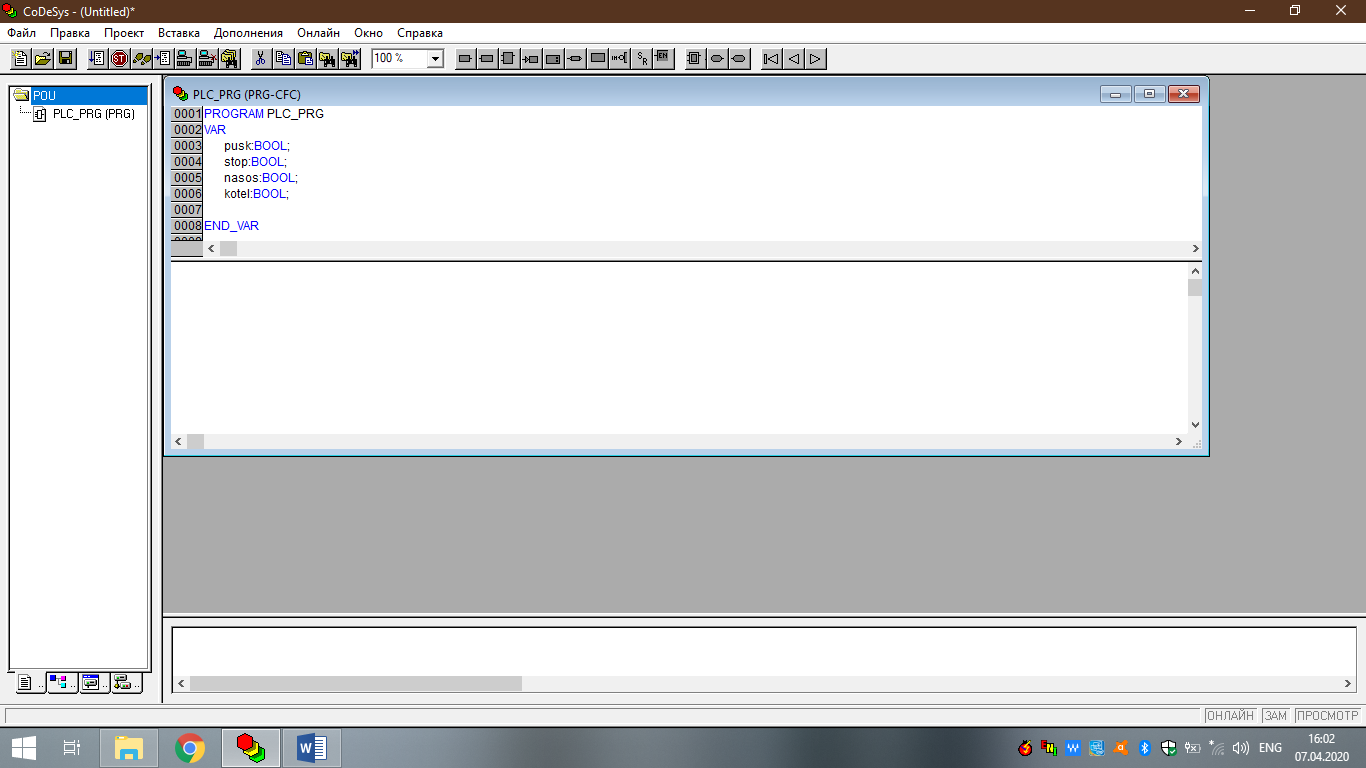


Рис. 1.4. Сигналы и переменные программы.

1. Далее для работы нужно используйте временные значения, так как котел и насос включались и отключались с задержкой. Для этого задайте переменную, например, *T*вкл – время включения – 3 с и *Т*откл – время отключения – 5 с. (рис. 1.5)

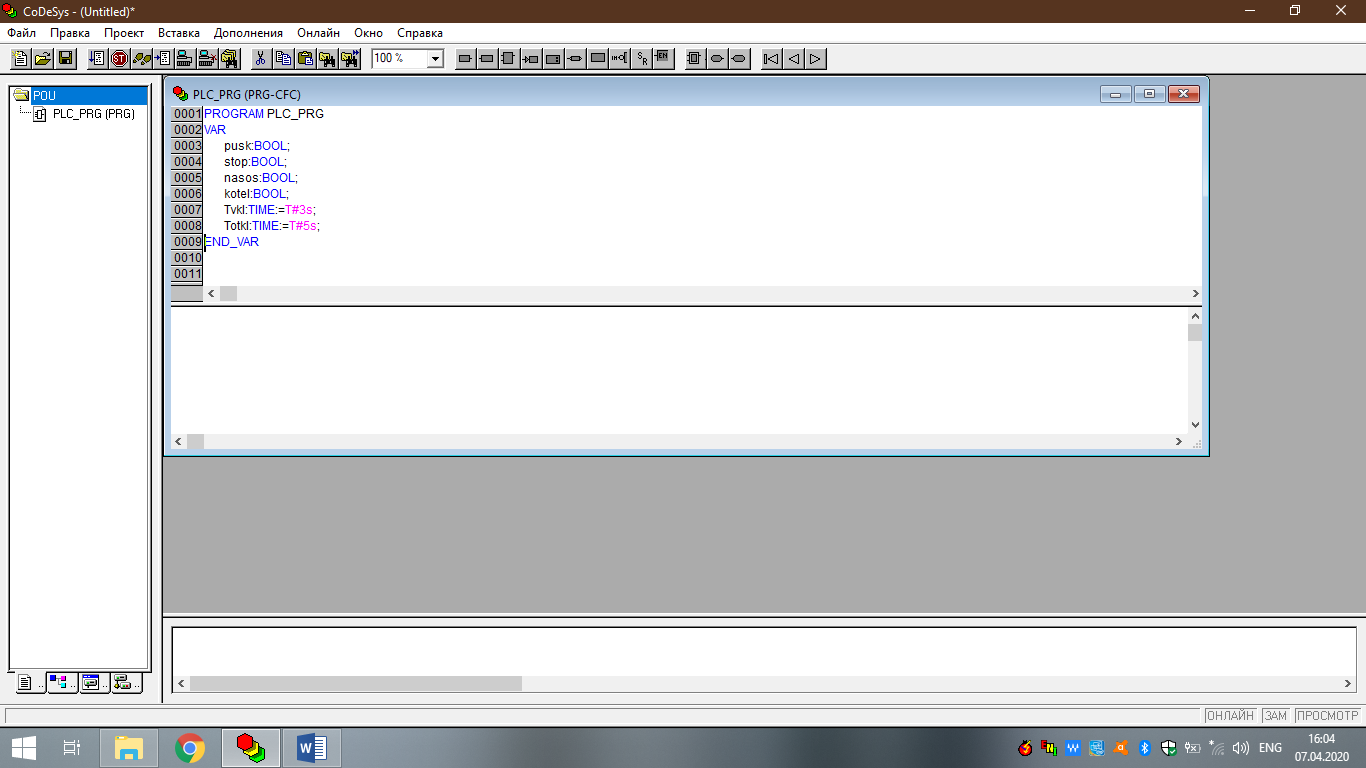


Рис. 1.5. Переменные времени включения и отключения

1. Для реализации времени включения и отключения, задайте два стандартных таймера. С именами их также в программе – ton1 в Codesys – это ТОN; tof 2 в Codesys – это TOF (рис. 1.6).

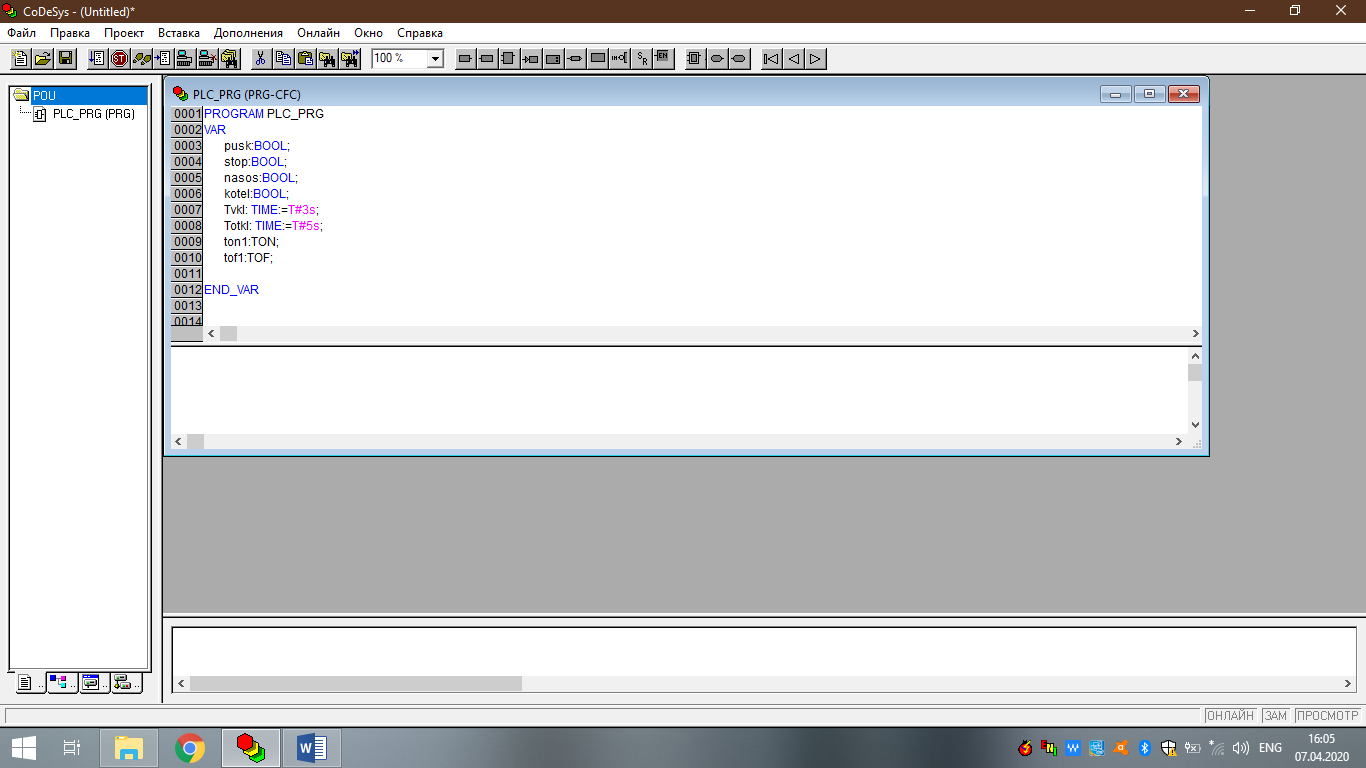


Рис. 1.6. Таймеры в алгоритме

В итоге получаются переменные под две кнопки (пуск и стоп), переменные под насос с котлом; две переменные, связанные с временными задержками; два таймера, которые эти задержки будут реализовывать.

5. Посмотрите как будет выглядеть алгоритм, используя язык функциональных блоков.

6. Для наглядности работы кнопок «Пуск » и «Стоп» введите еще одну переменную «WORK» (рис. 1.7).

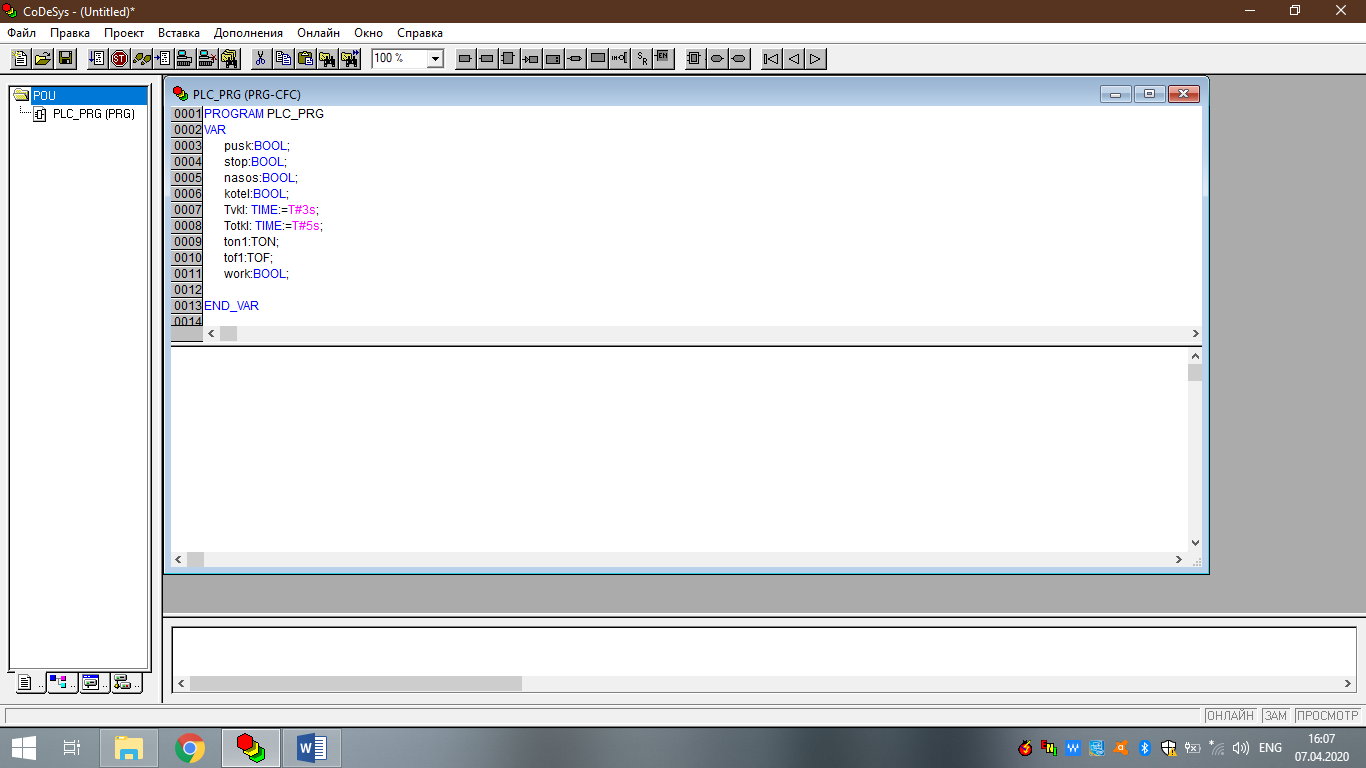


Рис. 1.7. Ввод переменной WORK

7. Составьте алгоритм: как только появляется значение «Истина» в переменной «Пуск», т.е. оператор нажимает соответствующую кнопку, то сделайт так, чтоб работа началась , т.е. значение «Истина» из переменной «Пуск» передалось в переменную «Work» (рис. 1.8).

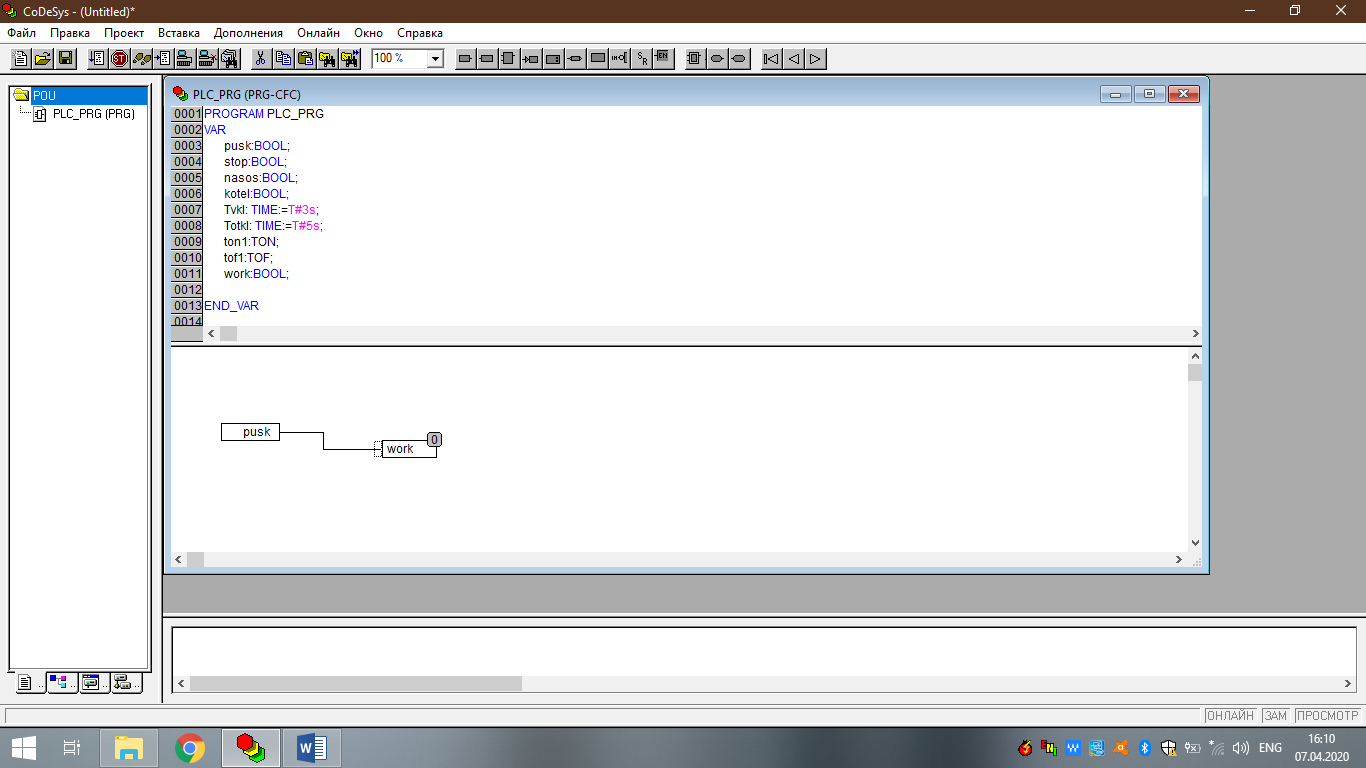


Рис. 1.8. Передача значения «Истина» из переменной «Пуск»,

передалась в переменную «Work»

Для фиксации этого значения выполните операцию SET (рис. 1.9).

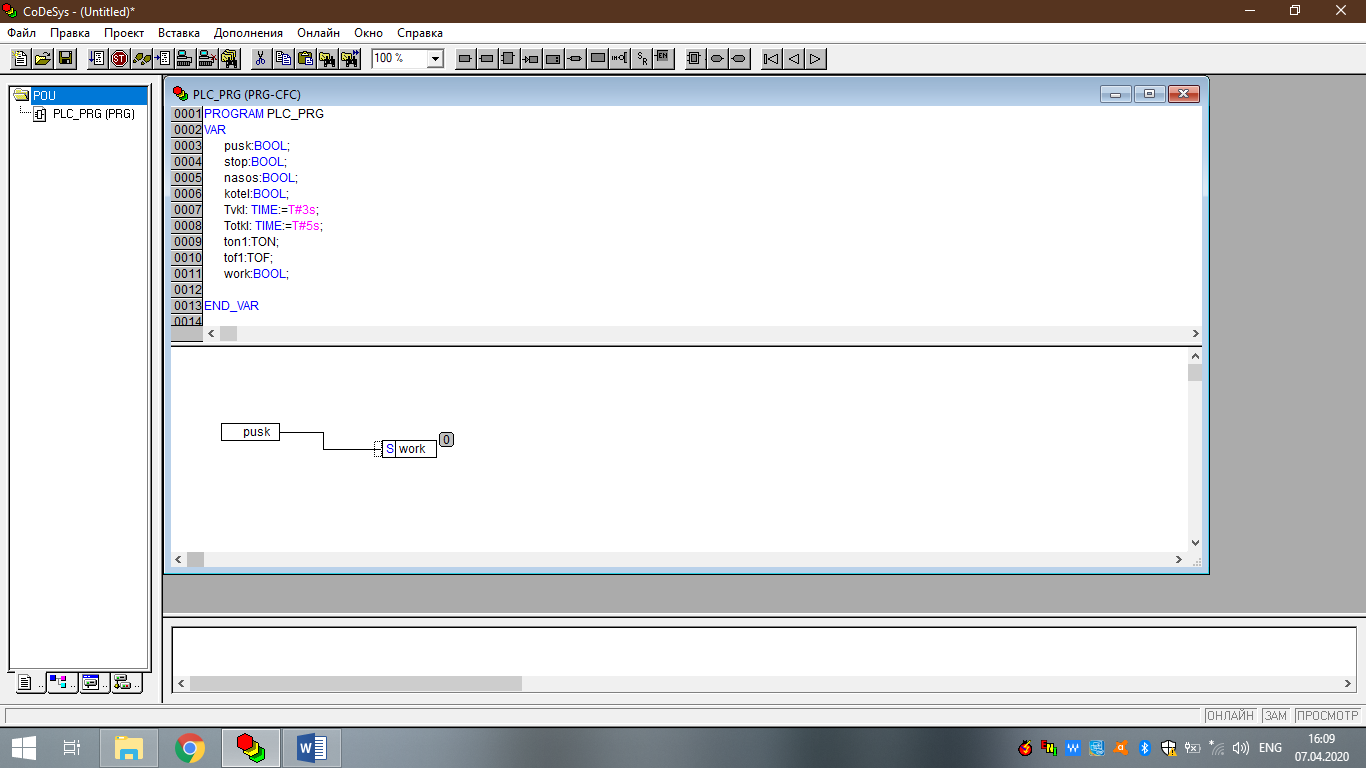


Рис. 1.9. Операция SET

Когда оператор будет нажимать кнопку «СТОП», в переменную «Work» запишите значение «Ложь», т.е. выключите установку целиком. Для этого на рабочей области пропишите «Work» с операцией сброса (RESET) – получается простейший RS - триггер с приоритетом включения (по «пуску» включается, по «стоп» - отключается) (рис. 1.10).

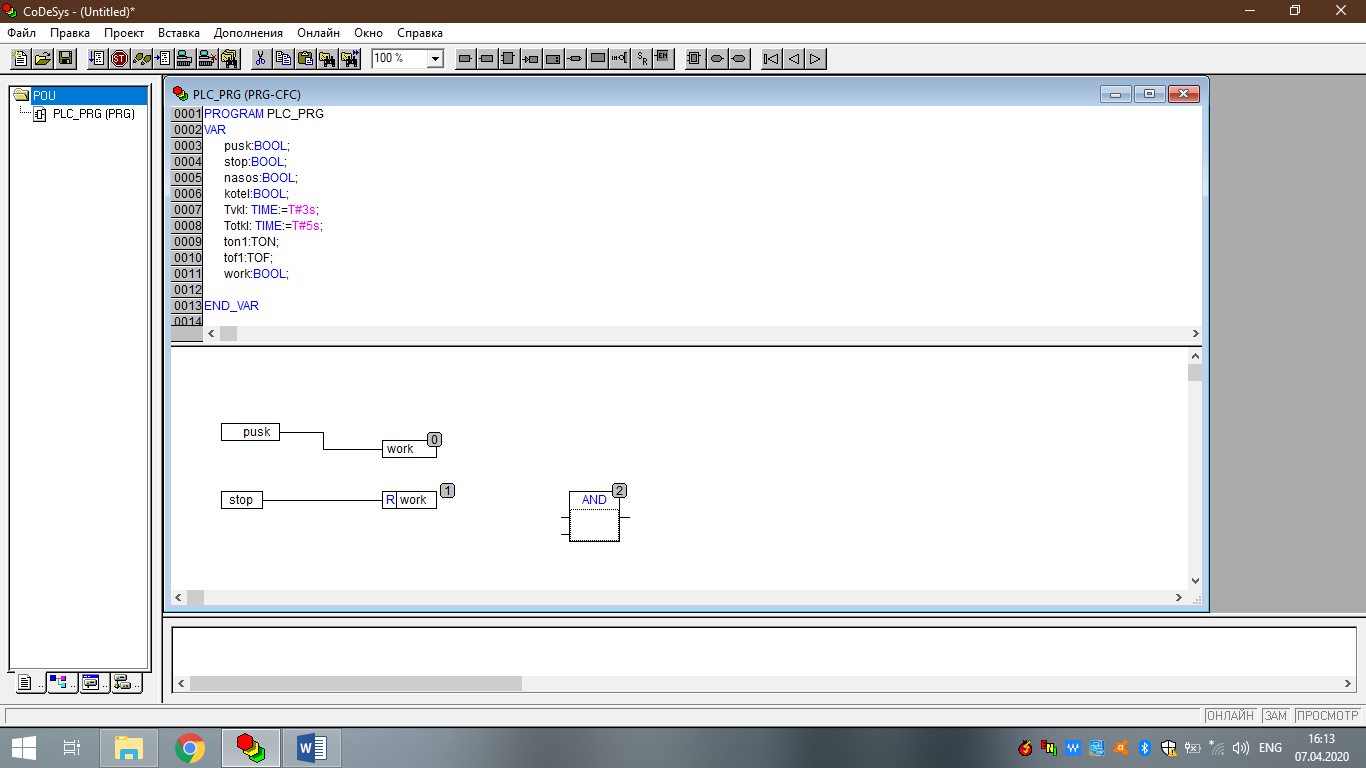


Рис. 1.10. RS- триггер

8. Для включения котла с задержкой используйте таймер (таймер TON), на рабочую область добавьте элемент

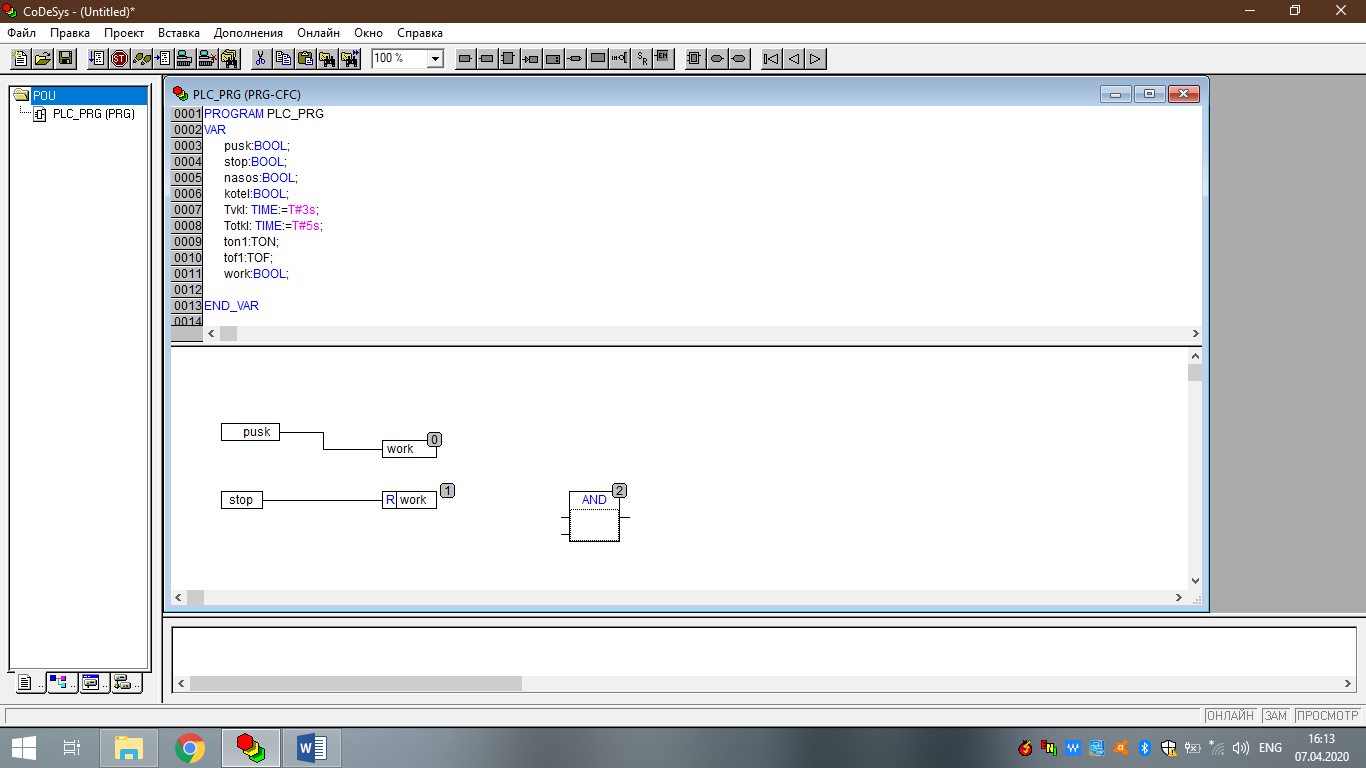


Рис. 1.11. Ввод таймера TON

Вместо названия AND добавьте название TON (тот таймер, который Вас интересует).

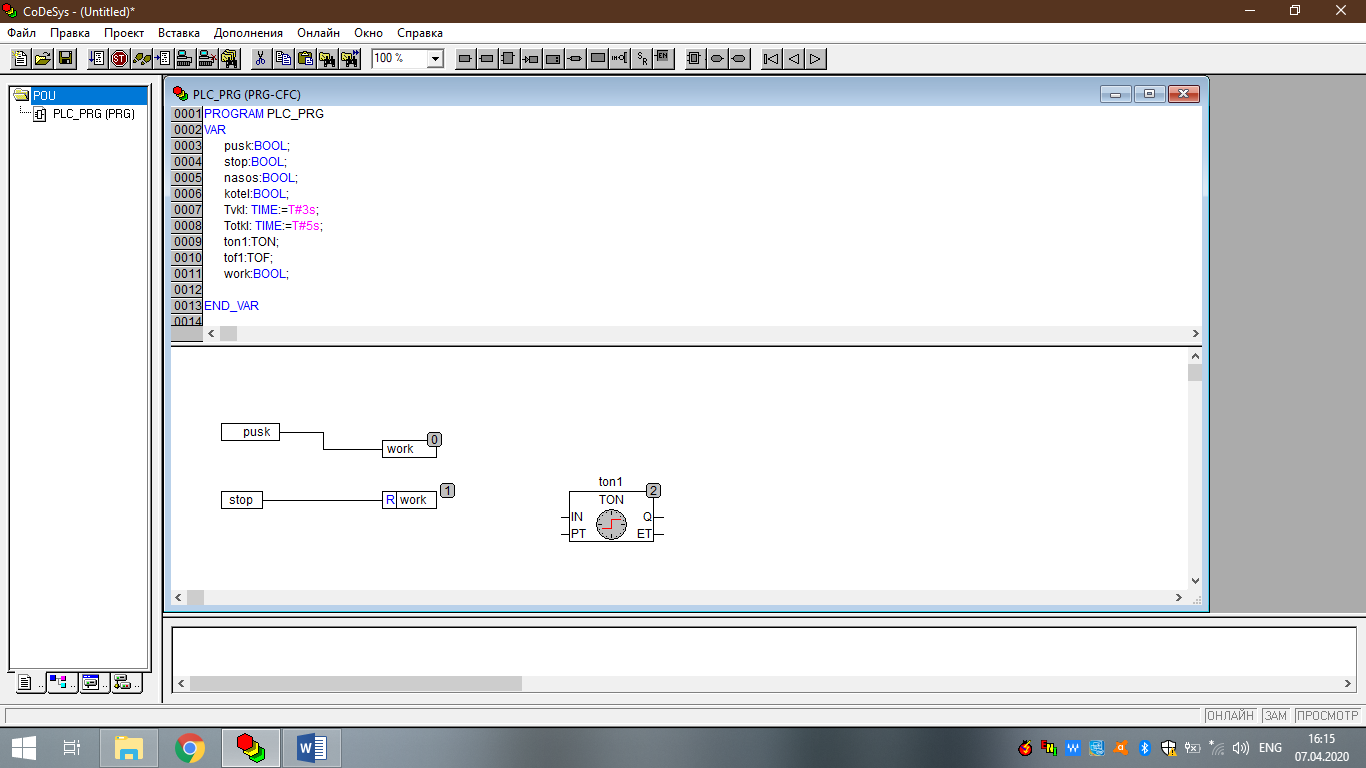


Рис. 1.12. Ввод таймера TON

9. Дайте название таймеру ton1, так как однотипных таймеров может быть несколько.

Сигналом для начала отсчета работы будет команда WORK (начало рабочего режима); задержку включения задайте через переменную *Tvkl* (по умолчанию имеет значение 3 c). На выходе таймера разместите переменную «kotel» (таким образом, дайте понять системе, что после нажатия клавиши «pusk» начинается работа «WORK», необходимо сделать отсчет задержки 3 сек и только после этого включите «kotel»). Когда значение «WORK» отпадает, котел отключается сразу без всяких задержек. Такова логика работы этого таймера (рис. 1.13).

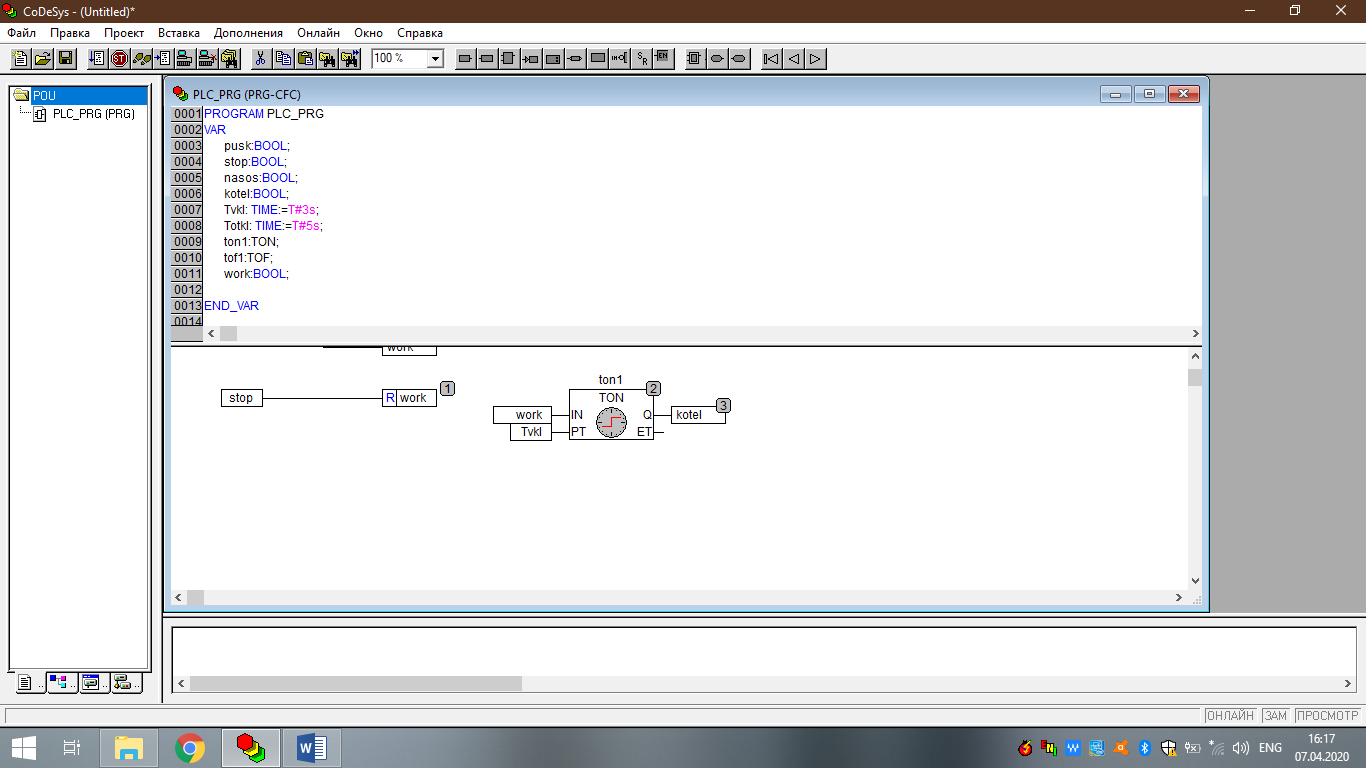
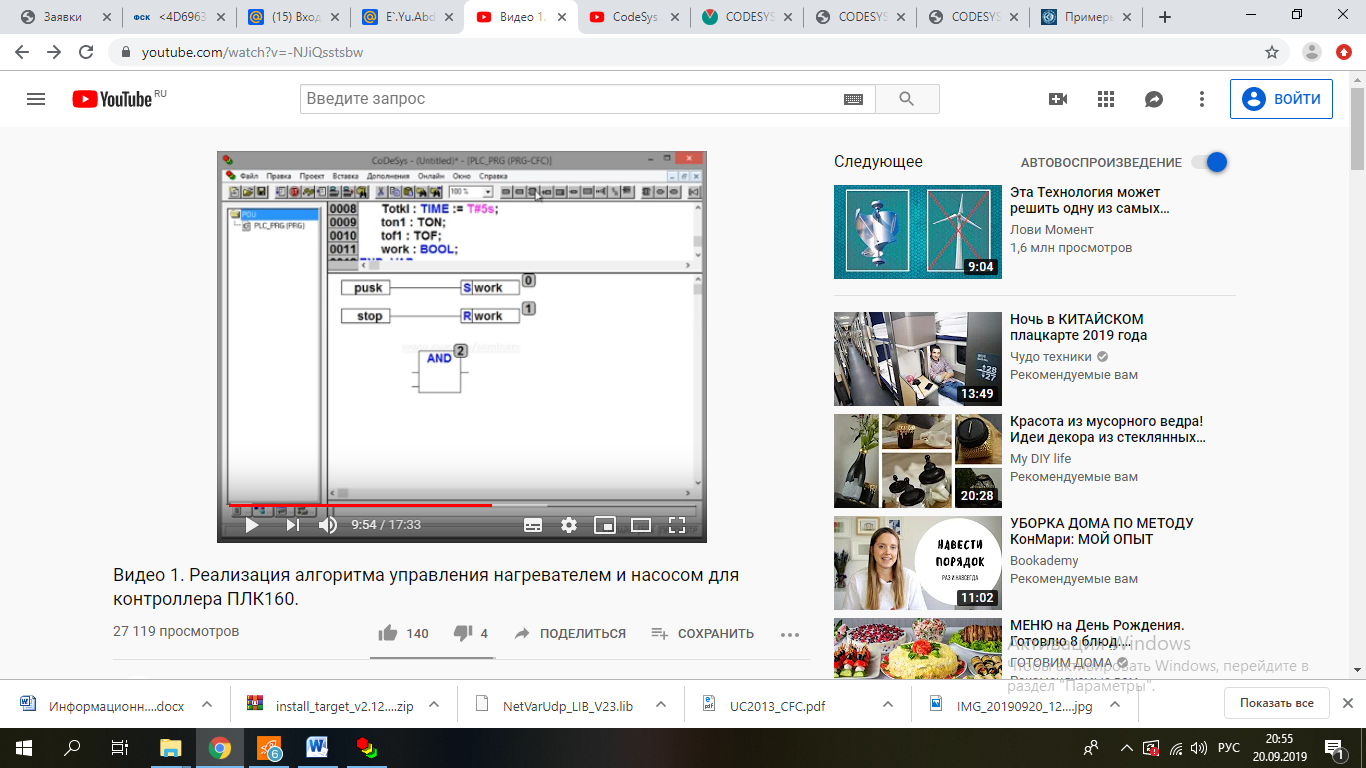


Рис. 1.13. Логика работы таймера TON

10. Далее по аналогии сделать задержку отключения для насоса.

Для этого опять разместите на рабочей плоскости блок-элемент

В заголовке пропишите нужный тип таймера TOF – это задержка отключения, и опять же присвойте ему имя, которое задали в программе – tof 1.

Соответственно, задержка отключения будет влиять на насос, подставьте на выход блока соответствующую переменную «nasos» (рис. 1.14).

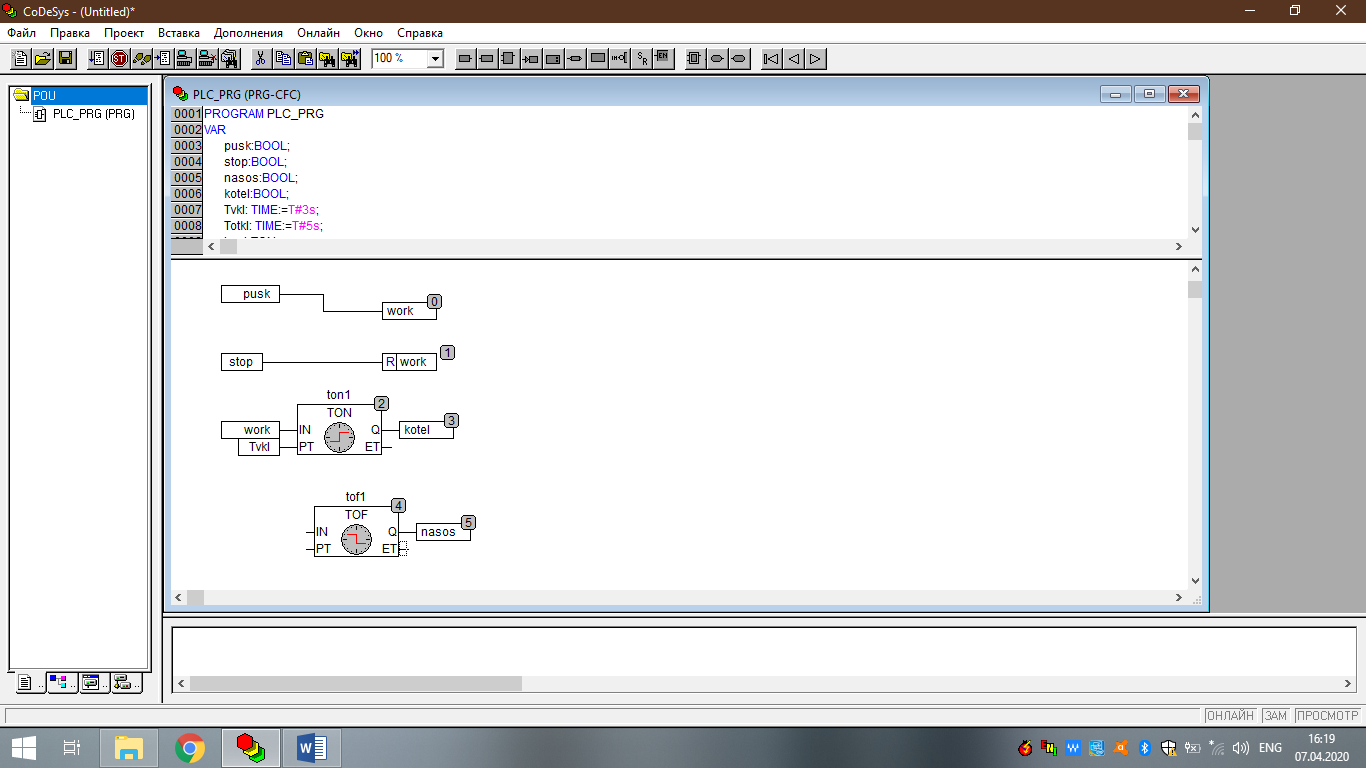


Рис. 1.15.

Соответственно, когда на входе появится сигнал «WORK», насос будет включаться сразу без каких-либо задержек, когда же сигнал «WORK» пропадает, таймер будет отсчитывать задержку заданной переменной *T*otkl и по истечении 5 с насос будет отключаться. Включение насоса происходит без каких-либо задержек. В общем алгоритм готов. (рис. 1.16)

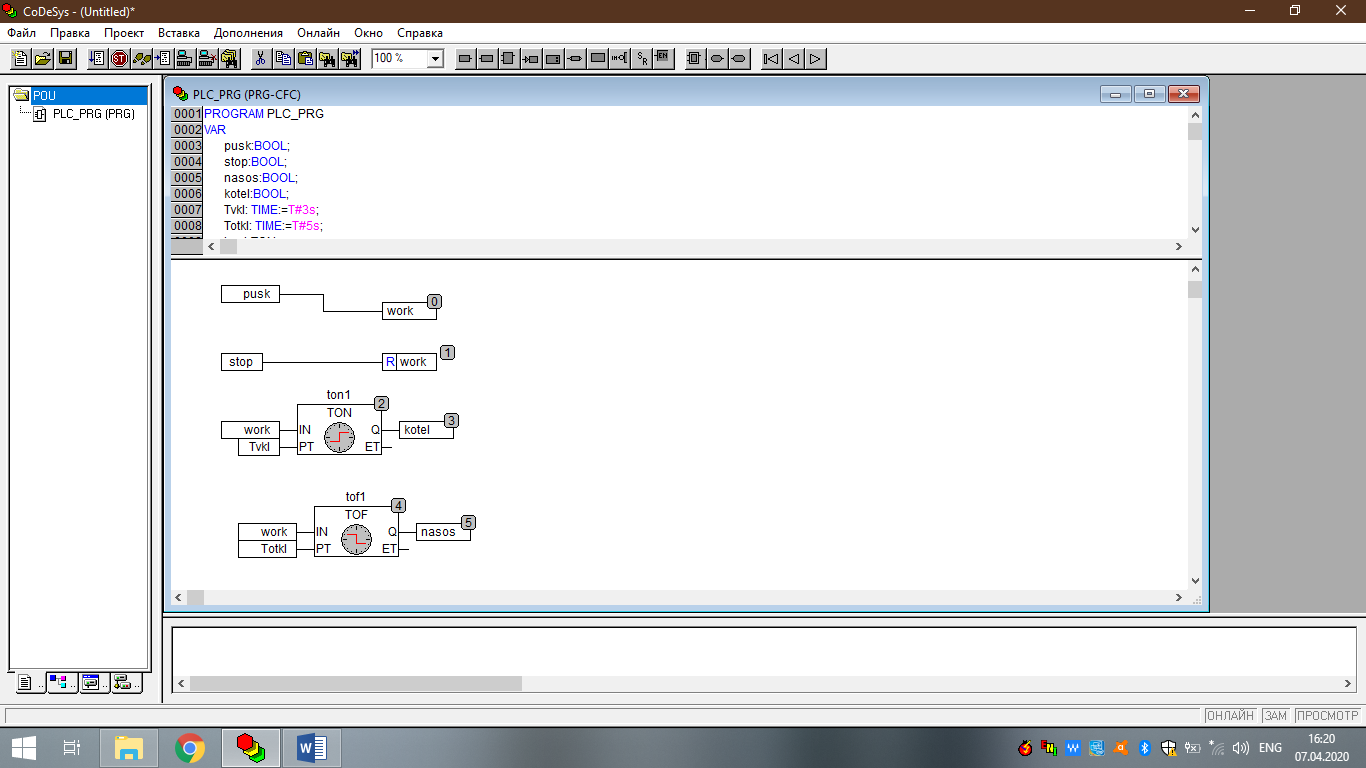


Рис. 1.16. Алгоритм включения работы котла

11. Далее сохраните алгоритм и расставьте порядок обработки блоков. Для этого правой кнопкой мыши нажмите на свободное место и выберите «Порядок» и далее «В соответствии с потоком данных» (рис. 1.17).

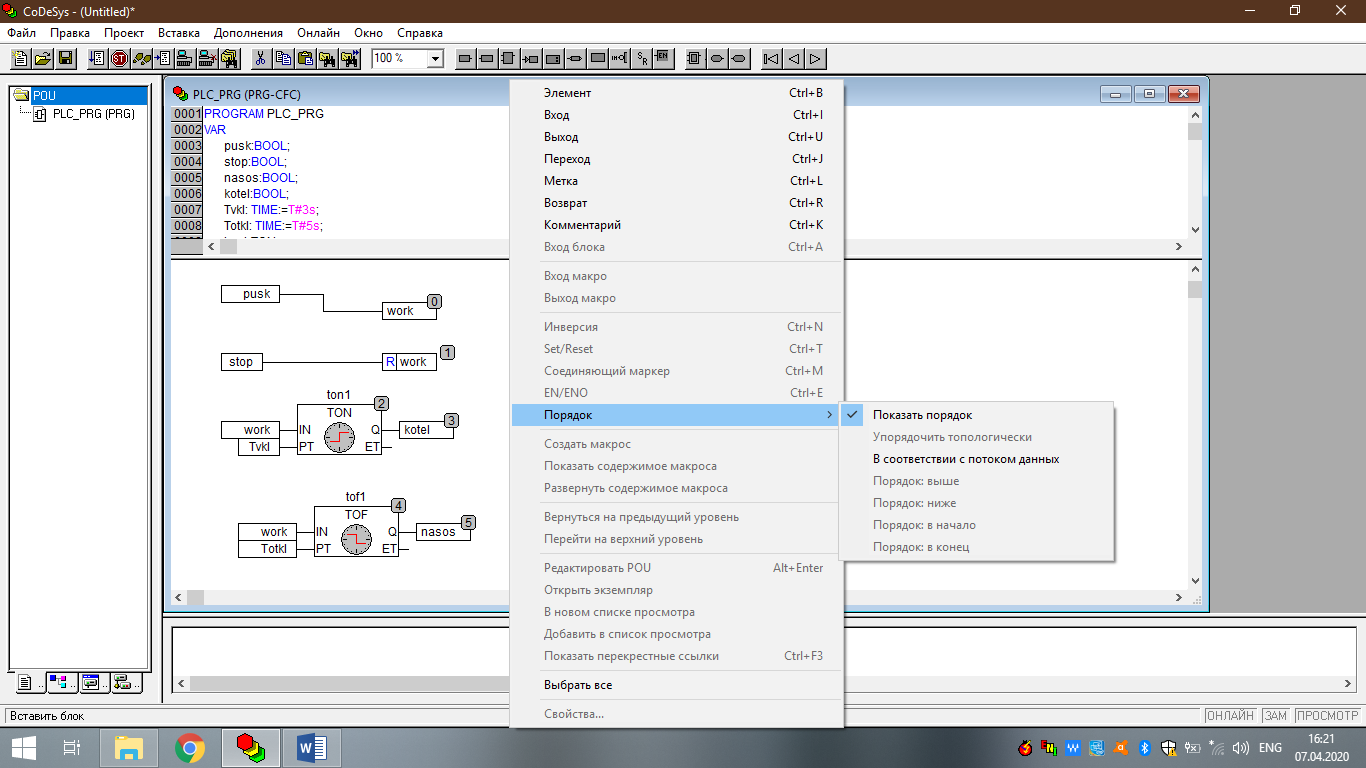


Рис. 1.17. Выбор функции «Порядок»

12. Для проверки работы алгоритма и программы воспользуйтесь режимом «Эмуляция» (рис. 1.18).

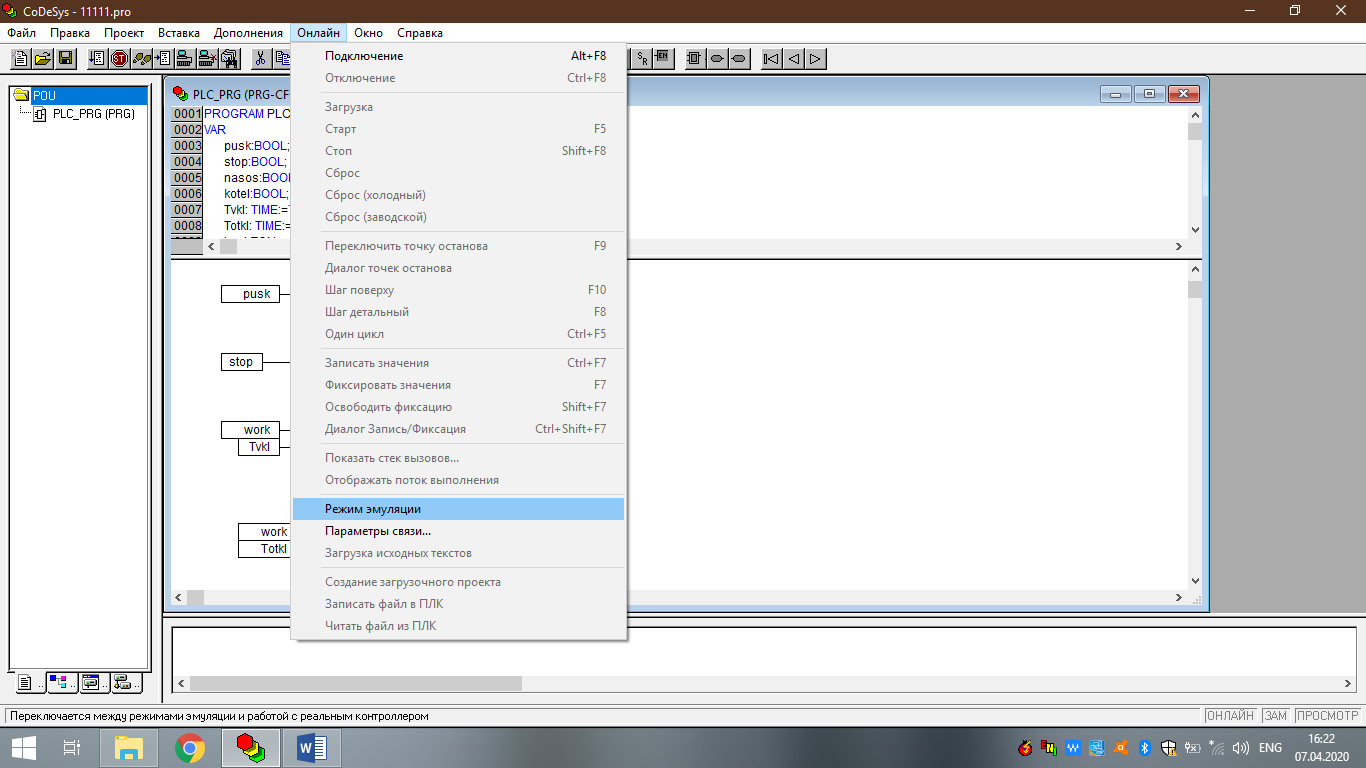


Рис. 1.18. Режим эмуляции

13. Выполнить «Подключение», далее меняется рабочая область (рис. 1.19).

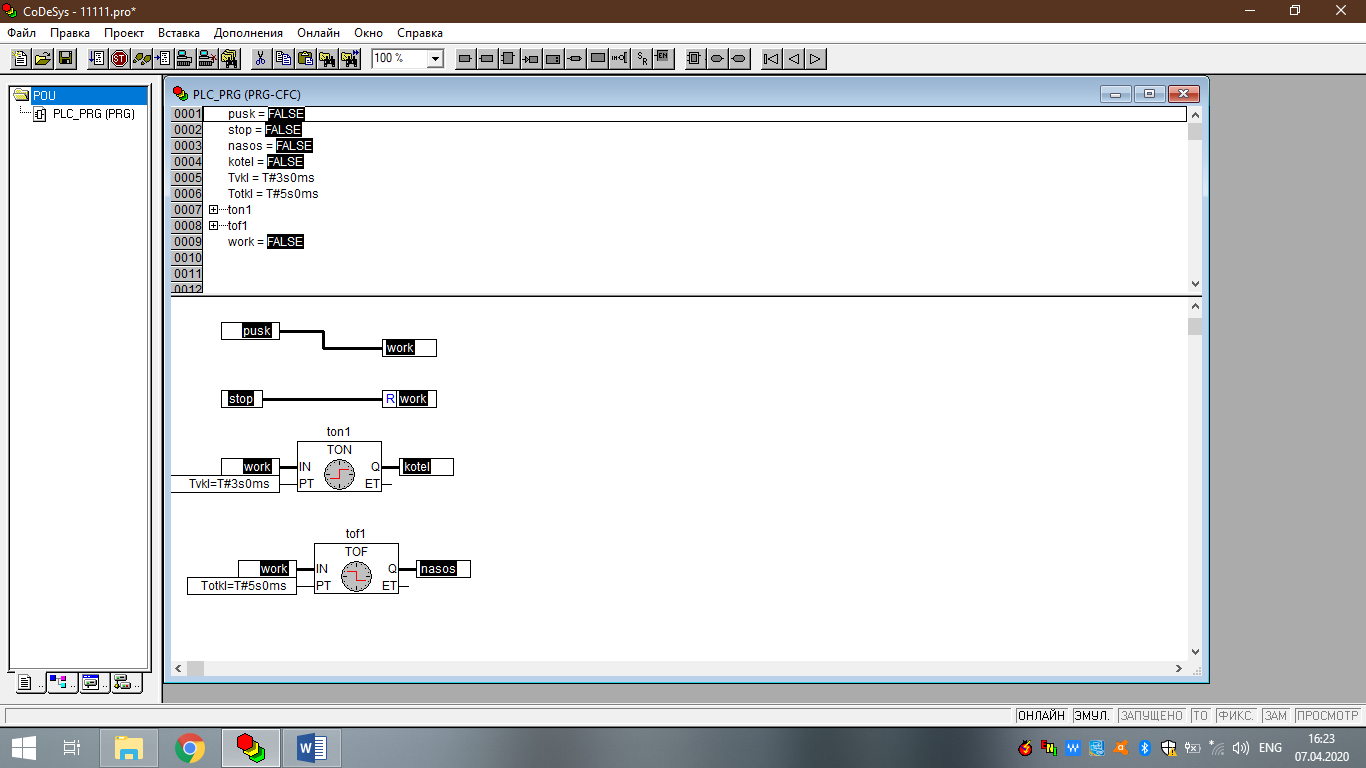


Рис. 1.19. Включение функции «Подключение»

14. Нажмите «Старт».

Для проверки без контроллера и отсутствия физических сигналов, дважды щелкните по переменной «pusk», которая по умолчанию имеет значение «Falce» и задайте вместо нее значение «TRUE». Далее в меню «Онлайн» выберите пункт «Записать значения» или используйте сочетание клавиш «Ctrl + F7».

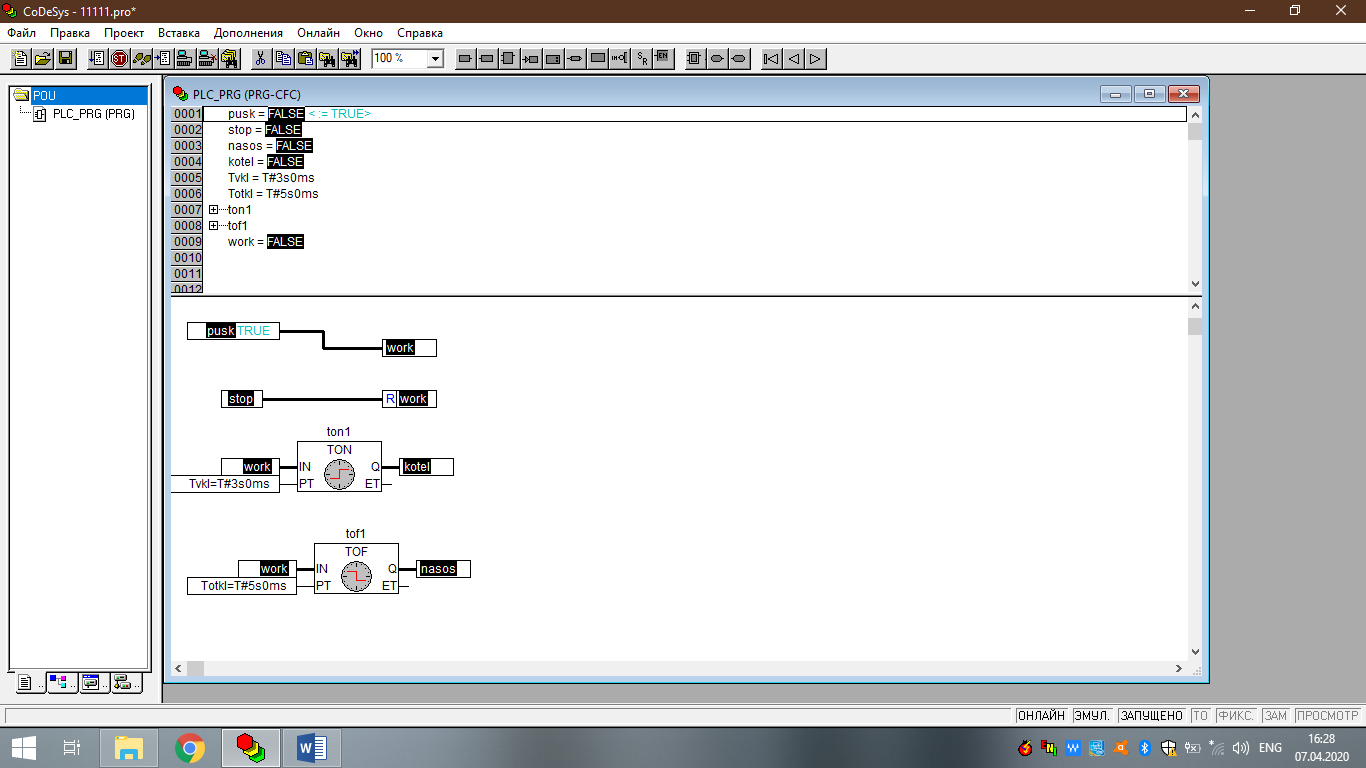


Рис. 1.20. Проверка работы программы

После этого значение добавляется в переменную «pusk», появляется «WORK», насос включился сразу, котел с задержкой времени. Далее через некоторое время сигнал «pusk» пропадает, появляется сигнал «stop» (рис. 1.21).

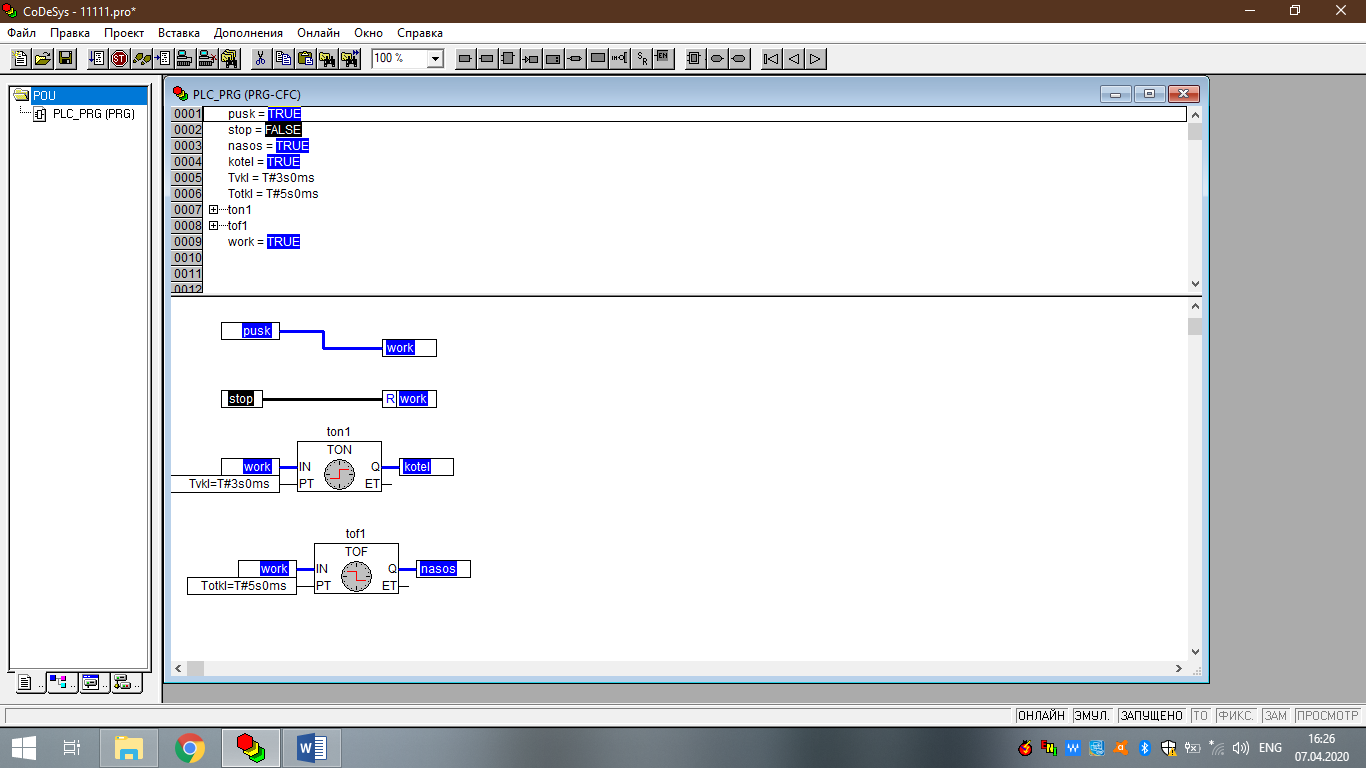


Рис. 1.21. Внешний вид программы с задержкой времени

**Требования к оформлению отчета**

Отчет о проделанной работе должен быть выполнен на листах формата А4 и содержать:

– название;

– цель работы;

– задание;

– алгоритм и графическое изображение программы;

– основные выводы;

– краткие ответы на контрольные вопросы.

**Контрольные вопросы**

1. Назовите пять языков, определяемых стандартом [IEC 61131-3](https://ru.wikipedia.org/wiki/IEC61131-3) (МЭК 61131-3)

2. Укажите назначение контроллера.

3. Назовите основные типы ПЛК.

4. Приведите типовой состав ПЛК.

5. Объясните, что позволяет выполнять язык функциональных блоков *CFC*.