**Лекция 7**

**Работа с Simulink Coder**

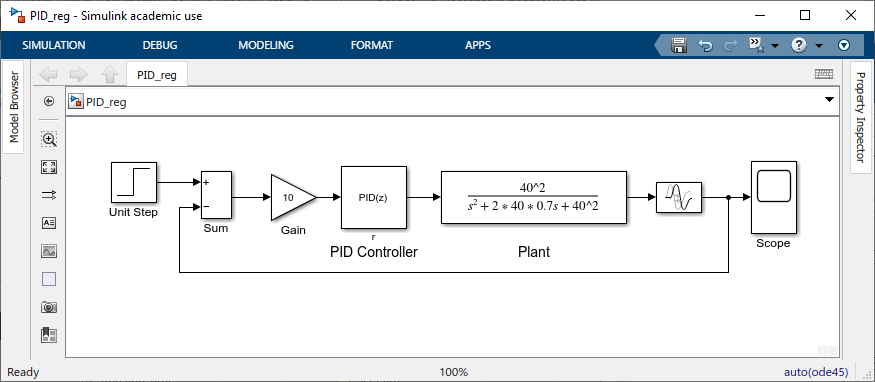
**Разработка программ для программируемых логических контроллеров (PLC) и микропроцессоров**

* 1. ***Подготовка ПИД-регулятора для реализации в PLC***

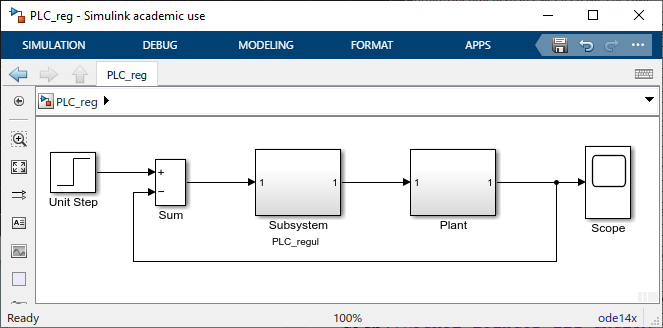
***7.2. Реализация ПИД-регулятора на языке C***

**7.1. Подготовка ПИД-регулятора для реализации в PLC**

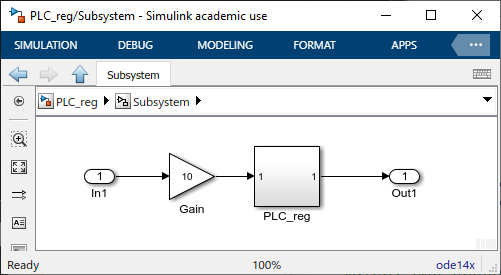
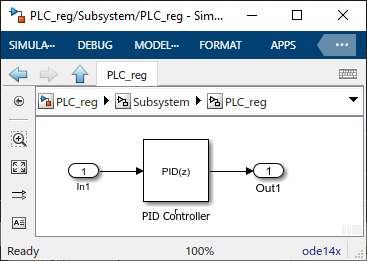
На первом этапе создадим Simulink-модель системы управления, показанную на рис.1 (файл **PID\_reg.slx**). Скопируем эту модель и создадим новую. Выделим блоки регулятора (Gain и PID Controller), которые будут реализованы программно в PLC. С помощью контекстного меню создадим из выделенных блоков подсистему. Также в виде подсистемы представим объект управления, хотя это не обязательно. Получившаяся при этом модель представлена на рис. 2 (файл **PLC\_reg.slx**), а состав ее блоков на рис. 3. На рис. 1 представлен элемент транспортной задержки. Он моделирует задержку передачи сигнала на выход модели и в канал обратной связи на заданное время.

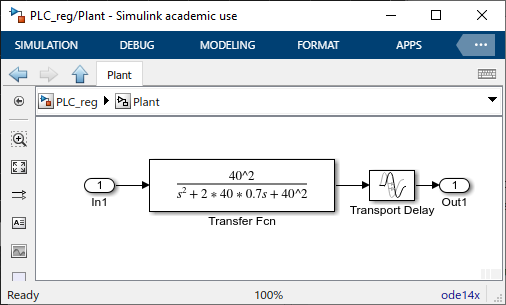


*Рис. 1. Исходная модель системы*



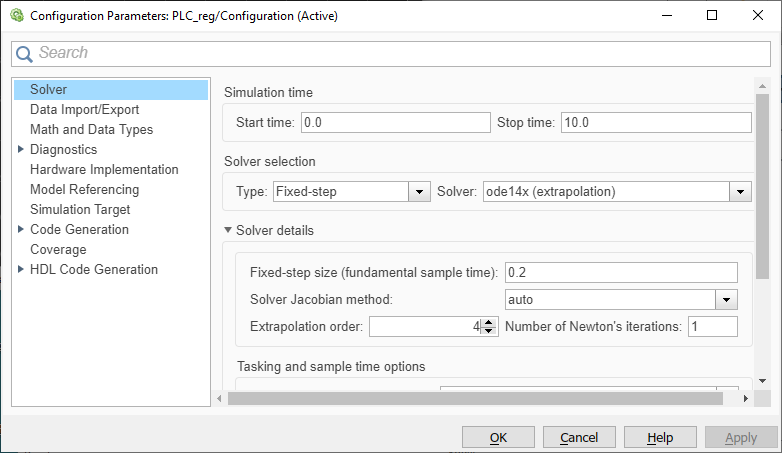
*Рис. 2. Модель системы управления*

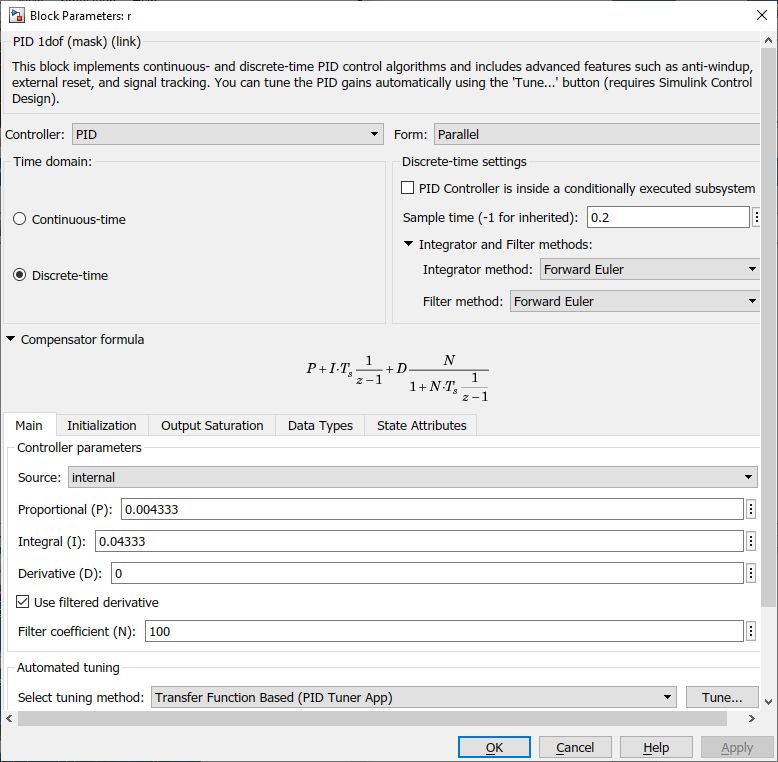


*Рис. 3. Состав блоков модели*

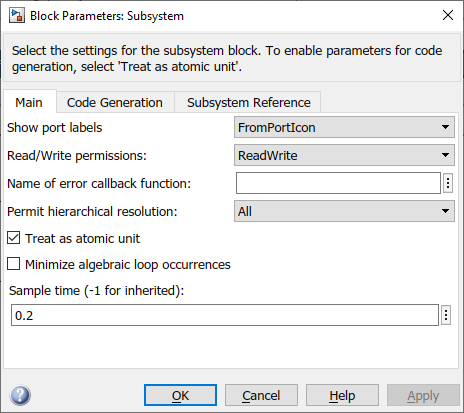
Произведем настройку системы. Сначала откроем окно конфигуратора и настроим Solver, как это показано на рис. 4. Затем откроем окно настройки блока PID(z), выполнив двойной щелчок на нем (рис. 5). Проверим настройку блока PID(z) в соответствии с рис. 5.



*Рис. 4. Настройка решателя Solver*

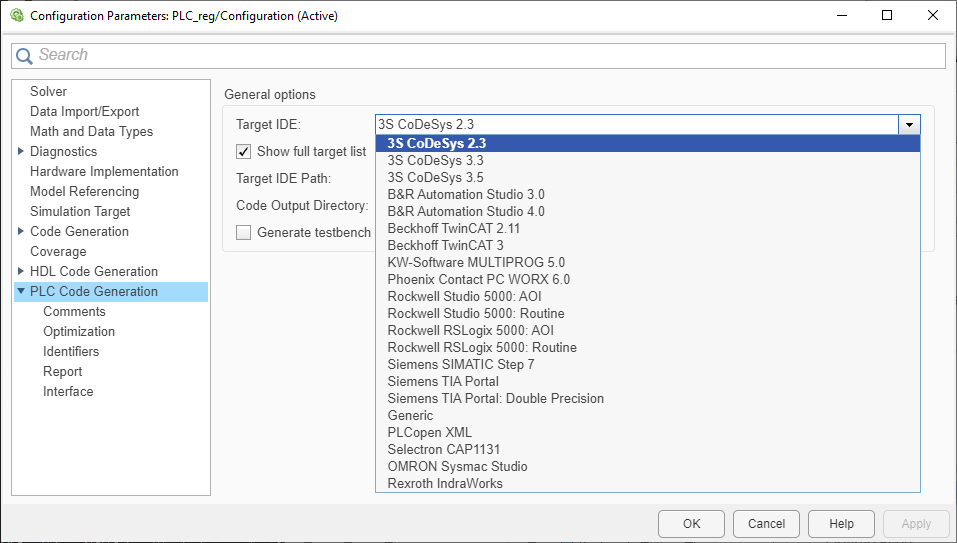


*Рис. 5. Окно настройки блока PID(z)*



*Рис. 6. Результат настройки*

Вернемся к основной модели (см. рис. 2) и с помощью контекстного меню блока регулятора **PLC\_regul** в появившемся меню выберем **Block Parameters (Subsystem)**. В появившемся окне (см. рис. 6) настроим параметры блока, не забыв отметить галочкой пункт **Treat as atomic unit**. Вновь выберем контекстное меню блока PLC\_regul и перейдем в окно конфигурации параметров PLC Code/Options…, показанное на рис. 7. Здесь нужно выбрать среду, для которой мы генерируем код (в данном случае 3S CoDeSys 2.3) и подтвердить свой выбор нажатием **OK**. После этого опять вызвать контекстное меню блока **PLC\_regul** и выбрать **PLC Code/Generate Code for Subsystem**. После сообщения об успешной генерации кода появляется окно отчета, в котором можно посмотреть файл кода PLC\_reg.exp на языке ST (табл. 1).



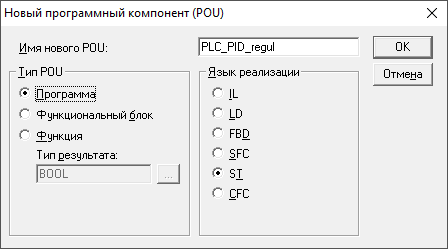
*Рис. 7. Окно настройки параметров генератора кода PLC Code Generation*

*Таблица 1*

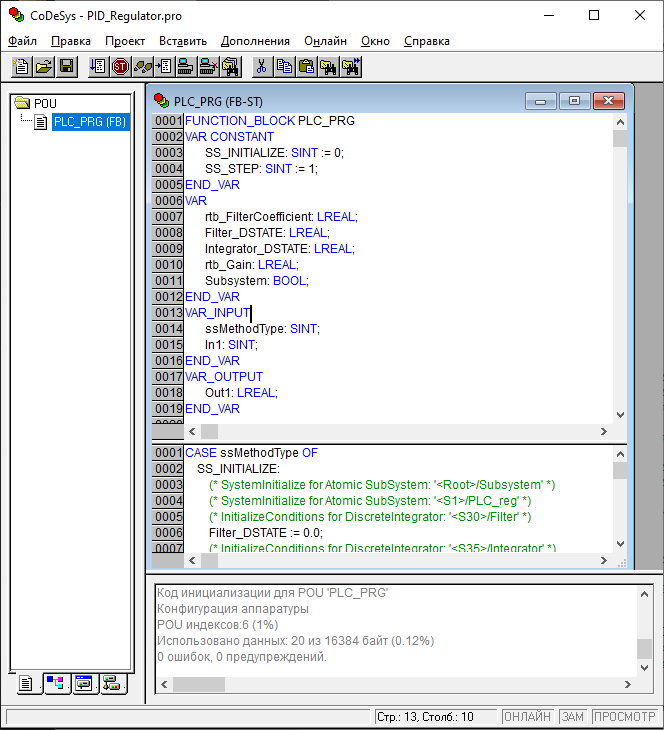
**Сгенерированный ST-код**

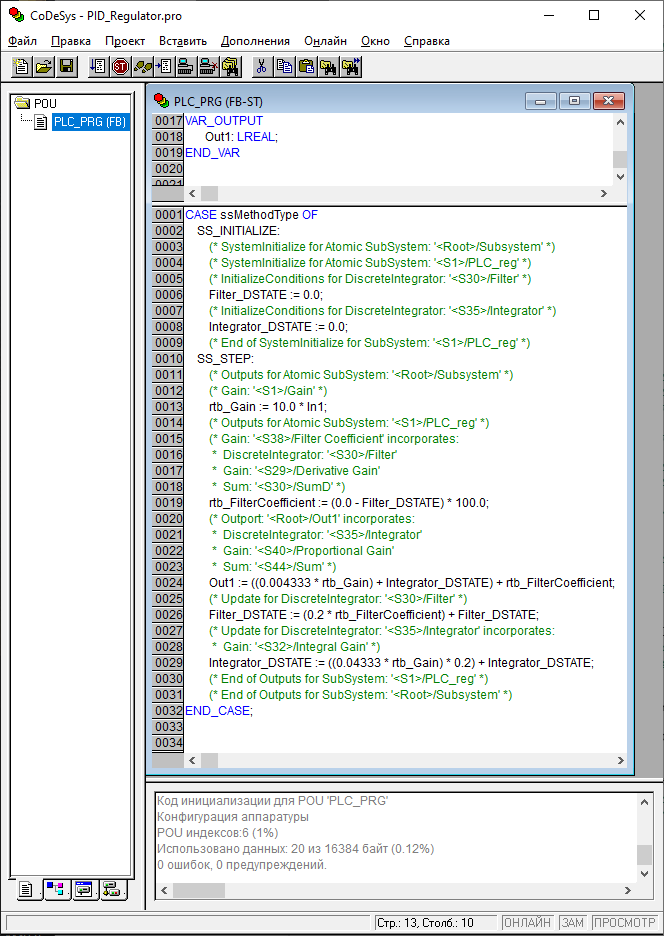
|  |
| --- |
| 1 (\*  2 \*  3 \* File: PLC\_reg.exp  4 \*  5 \* IEC 61131-3 Structured Text (ST) code generated for subsystem "PLC\_reg/Subsystem"  6 \*  7 \* Model name : PLC\_reg  8 \* Model version : 8.8  9 \* Model creator : Alex  10 \* Model last modified by : Admin  11 \* Model last modified on : Mon Apr 25 20:59:50 2022  12 \* Model sample time : 0.2s  13 \* Subsystem name : PLC\_reg/Subsystem  14 \* Subsystem sample time : 0.2s  15 \* Simulink PLC Coder version : 3.4 (R2021a) 14-Nov-2020  16 \* ST code generated on : Mon Apr 25 22:57:25 2022  17 \*  18 \* Target IDE selection : 3S CoDeSys 2.3  19 \* Test Bench included : No  20 \*  21 \*)  22 FUNCTION\_BLOCK Subsystem  23 VAR\_INPUT  24 ssMethodType: SINT;  25 In1: LREAL;  26 END\_VAR  27 VAR\_OUTPUT  28 Out1: LREAL;  29 END\_VAR  30 VAR  31 Filter\_DSTATE: LREAL;  32 Integrator\_DSTATE: LREAL;  33 END\_VAR  34 VAR\_TEMP  35 rtb\_Gain: LREAL;  36 rtb\_FilterCoefficient: LREAL;  37 END\_VAR  38 CASE ssMethodType OF  39 SS\_INITIALIZE:  40 (\* SystemInitialize for Atomic SubSystem: '<Root>/Subsystem' \*)  41 (\* SystemInitialize for Atomic SubSystem: '[<S1>/PLC\_reg](matlab:coder.internal.code2model('PLC_reg:10'))' \*)  42 (\* InitializeConditions for DiscreteIntegrator: '[<S30>/Filter](matlab:coder.internal.code2model('PLC_reg:1:737'))' \*)  **43 Filter\_DSTATE := 0.0;**  44 (\* InitializeConditions for DiscreteIntegrator: '[<S35>/Integrator](matlab:coder.internal.code2model('PLC_reg:1:843'))' \*)  **45 Integrator\_DSTATE := 0.0;**  46 (\* End of SystemInitialize for SubSystem: '[<S1>/PLC\_reg](matlab:coder.internal.code2model('PLC_reg:10'))' \*)  47 (\* End of SystemInitialize for SubSystem: '<Root>/Subsystem' \*)  48 SS\_STEP:  49 (\* Outputs for Atomic SubSystem: '<Root>/Subsystem' \*)  50 (\* Gain: '[<S1>/Gain](matlab:coder.internal.code2model('PLC_reg:13'))' \*)  **51 rtb\_Gain := 10.0 \* In1;**  52 (\* Outputs for Atomic SubSystem: '[<S1>/PLC\_reg](matlab:coder.internal.code2model('PLC_reg:10'))' \*)  53 (\* Gain: '[<S38>/Filter Coefficient](matlab:coder.internal.code2model('PLC_reg:1:882'))' incorporates:  54 \* DiscreteIntegrator: '[<S30>/Filter](matlab:coder.internal.code2model('PLC_reg:1:737'))'  55 \* Gain: '[<S29>/Derivative Gain](matlab:coder.internal.code2model('PLC_reg:1:688'))'  56 \* Sum: '[<S30>/SumD](matlab:coder.internal.code2model('PLC_reg:1:738'))' \*)  **57 rtb\_FilterCoefficient := (0.0 - Filter\_DSTATE) \* 100.0;**  58 (\* Outport: '<Root>/Out1' incorporates:  59 \* DiscreteIntegrator: '[<S35>/Integrator](matlab:coder.internal.code2model('PLC_reg:1:843'))'  60 \* Gain: '[<S40>/Proportional Gain](matlab:coder.internal.code2model('PLC_reg:1:913'))'  61 \* Sum: '[<S44>/Sum](matlab:coder.internal.code2model('PLC_reg:1:3526'))' \*)  **62 Out1 := ((0.004333 \* rtb\_Gain) + Integrator\_DSTATE) + rtb\_FilterCoefficient;**  63 (\* Update for DiscreteIntegrator: '[<S30>/Filter](matlab:coder.internal.code2model('PLC_reg:1:737'))' \*)  64 **Filter\_DSTATE := (0.2 \* rtb\_FilterCoefficient) + Filter\_DSTATE;**  65 (\* Update for DiscreteIntegrator: '[<S35>/Integrator](matlab:coder.internal.code2model('PLC_reg:1:843'))' incorporates:  66 \* Gain: '[<S32>/Integral Gain](matlab:coder.internal.code2model('PLC_reg:1:791'))' \*)  **67 Integrator\_DSTATE := ((0.04333 \* rtb\_Gain) \* 0.2) + Integrator\_DSTATE;**  68 (\* End of Outputs for SubSystem: '[<S1>/PLC\_reg](matlab:coder.internal.code2model('PLC_reg:10'))' \*)  69 (\* End of Outputs for SubSystem: '<Root>/Subsystem' \*)  70 END\_CASE;  71 END\_FUNCTION\_BLOCK  72 VAR\_GLOBAL CONSTANT  73 SS\_INITIALIZE: SINT := 0;  74 SS\_STEP: SINT := 1;  75 END\_VAR |

После этого запускаем среду разработки CoDeSys и создаем новый программный компонент ̶ Функциональный блок или Программа (см. рис. 8). Копируем программный код в среду CoDeSys и выполняем его компиляцию.



*Рис. 8. Создание нового программного компонента*





*Рис. 9. Окна раздела переменных и раздела кода CoDeSys после успешной компиляции кода ST-программы*

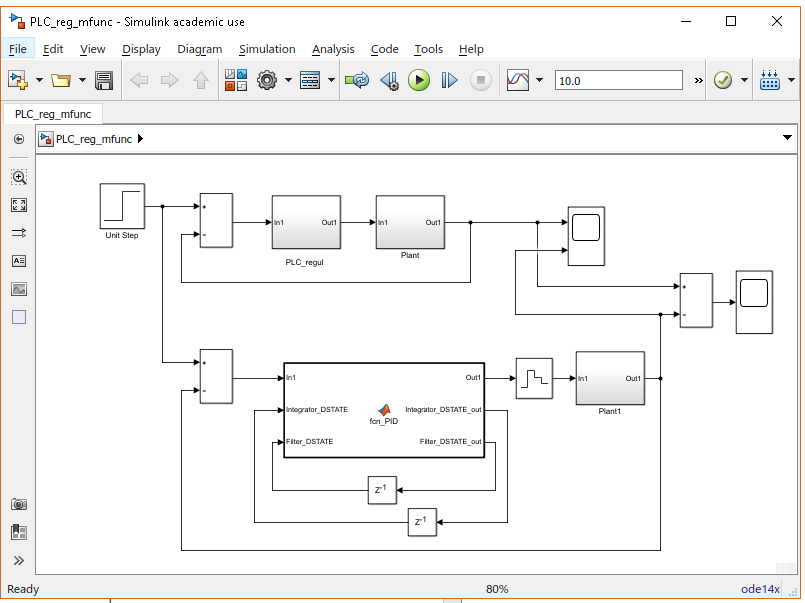
Окно среды CoDeSys после успешной компиляции программы показано на рис. 9. Здесь показаны два окна с полным содержимым раздела переменных (вверху) и раздела кода программы на языке ST (внизу).

Дальнейшие действия зависят от состава имеющегося оборудования и программной среды в которой мы предполагаем работать. Так, например, в среде CoDeSys можно использовать встроенный в среду виртуальный PLC в режиме симуляции (то есть, не подключая реальное оборудование) и отладить подготовленную выше программу. Здесь мы поступим по-другому.

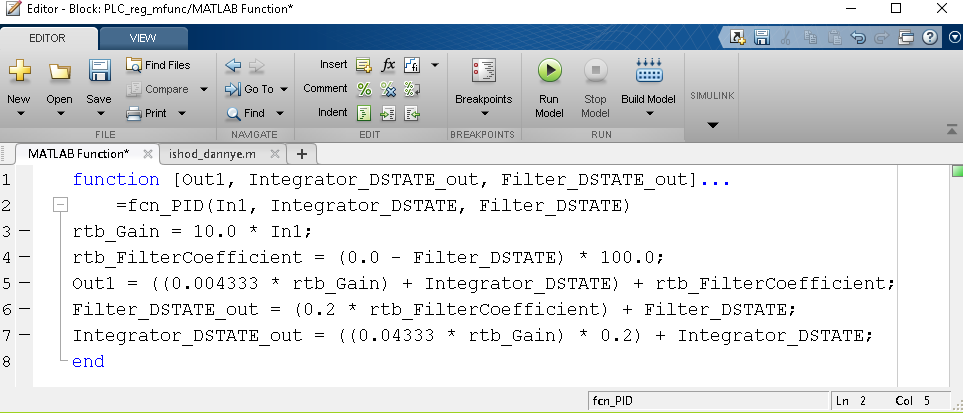
***Проверка PLC кода в модели Simulink***

Мы получили PLC код. Перед загрузкой его в PLC проверим правильность его работы в режиме моделирования в среде Simulink. Как это сделать?

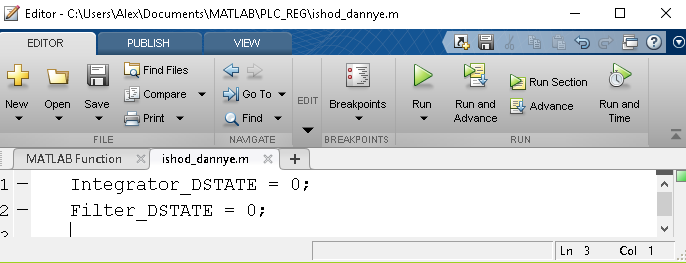
Создадим модель, показанную на рис. 10 (файл **PLC\_reg\_mfunc**). В качестве регулятора будем использовать блок MATLAB Function, в который запишем m-функцию, составленную из строк вычислительных операций кода ST-функции (это строки текста, выделенные фиолетовым цветом в табл. 1). Текст этой функции приведен на рис. 11 (при этом следует заменить символы присваивания ":=" на "="). Не забудем также проинициализировать начальные значения переменных (нужно запустить файл **ishod\_dannye.m**, см. рис. 12).



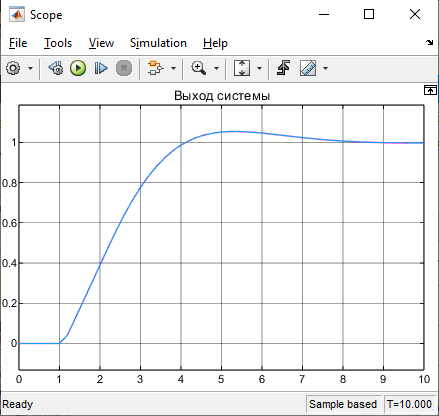
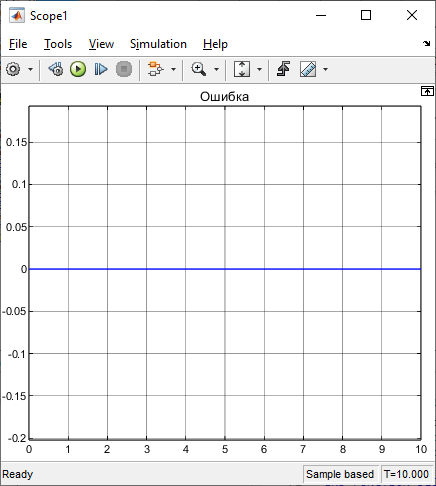
*Рис. 10. Модель системы с блоком регулятора в виде m-функции*



*Рис. 11. M-функция блока регулятора*



*Рис. 12. Инициализация переменных*

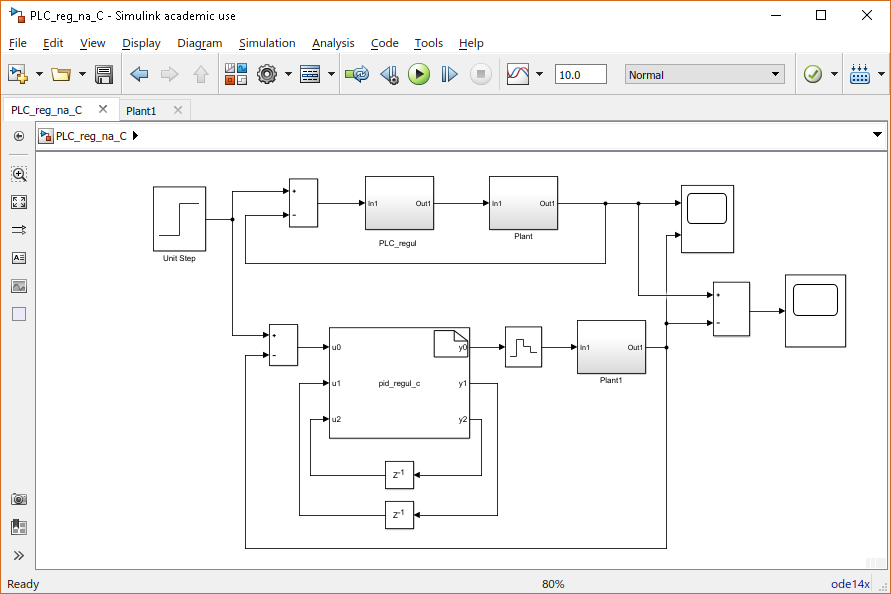
 

*Рис. 13. Результаты моделирования*

Теперь можно запустить нашу модель на симуляцию. Результаты моделирования, представленные на рис. 13, подтверждают правильность работы нашей модели. Задержка на 1 с вызвана настройкой блока транспортной задержки (см. выше).

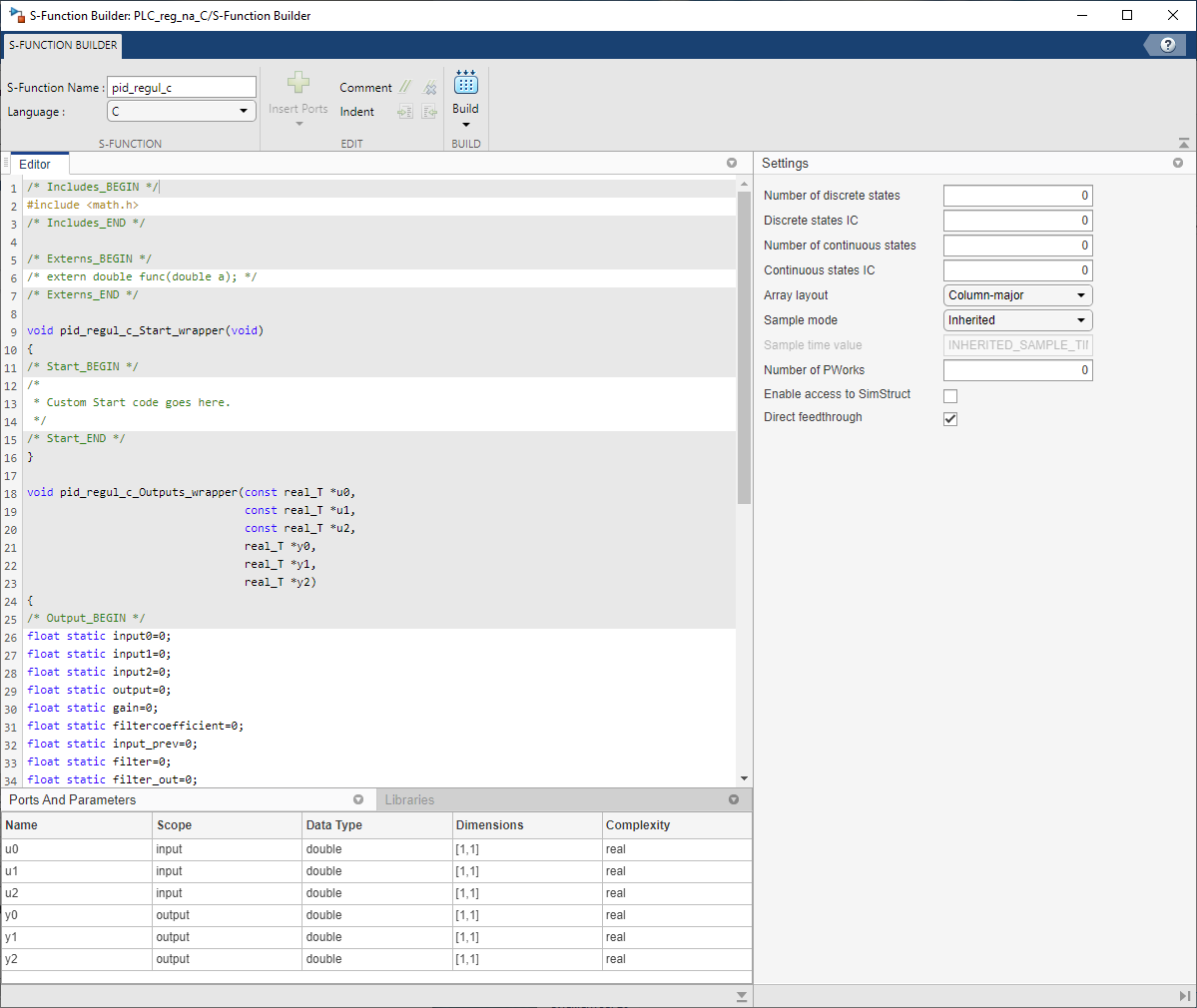
**7.2. Реализация ПИД-регулятора на языке C для использования в микроконтроллере**

В этом случае создадим блок S-Function Builder, размещенный в библиотеке User-Defined Function (см. рис. 14, файл **PLC\_reg\_na\_C.slx**).



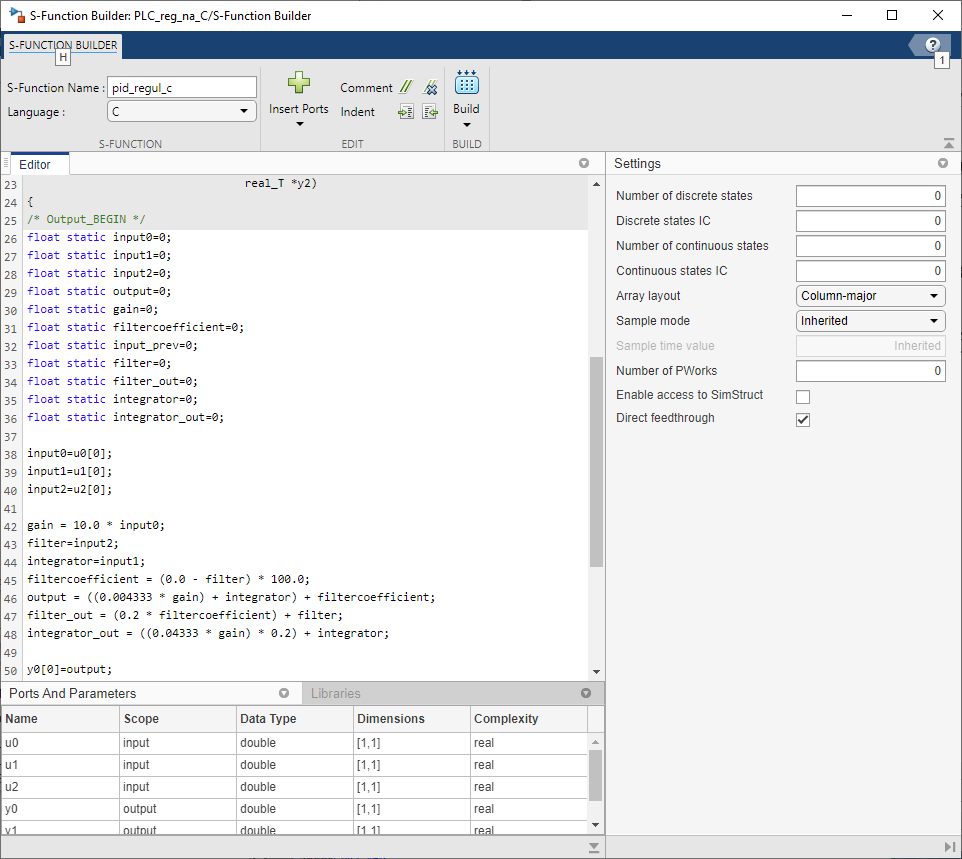
*Рис. 14. Модель системы с блоком регулятора в виде C-функции*

На рис. 14 видно, что в блоке регулятора имеются 3 входа и 3 выхода (как и в случае регулятора на языке MATLAB). Чтобы эти входы и выходы обозначились на схеме моделирования, необходимо открыть вкладку Insert Ports (при открытии она засвечивается) и затем с помощью кнопок Add Input и Add Output добавить входы () и выходы блока, присвоив им имена (см. рис. 15).



*Рис. 15. Добавление входных и выходных портов*

Теперь перейдем на вкладку Editor и введем код C-функции регулятора (см. рис. 16).



*Рис. 16. C-функция блока регулятора*

Отметим, что вид интерфейса может быть другим в различных версиях MATLAB/Simulink.

Полный текст программы блока ПИД-регулятора на языке C:

float static input\_prev=0;

float static filter=0;

float static filter\_out=0;

float static integrator=0;

float static integrator\_out=0;

input0=u0[0];

input1=u1[0];

input2=u2[0];

gain = 10.0 \* input0;

integrator=input1;

filter=input2;

filtercoefficient = (0.0 - filter) \* 100.0;

output = ((0.004333 \* gain) + integrator) + filtercoefficient;

filter\_out = (0.2 \* filtercoefficient) + filter;

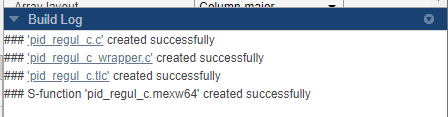
integrator\_out = ((0.04333 \* gain) \* 0.2) + integrator;

y0[0]=output;

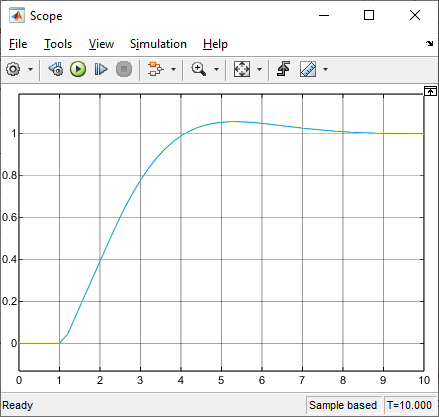
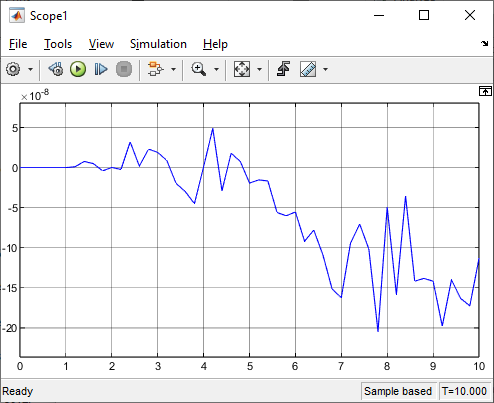
y1[0]=integrator\_out;

y2[0]=filter\_out;

После ввода текста функции, нажимаем кнопку Build, чтобы откомпилировать программу. После удачной компиляции программы будет создан двоичный загрузочный файл программы ПИД-регулятора и выведено сообщение в поле Build Log:



надпись ### **S-function 'pid\_regul\_c.mexw64' created successfully** свидетельствует об удачной компиляции и создании загрузочного файла **pid\_regul\_c.mexw64**

*Рис. 17. Результаты моделирования в режиме "программа в контуре управления"*

После этого запустим нашу модель на симуляцию. Результаты моделирования в режиме "Программа в контуре управления" представлены на рис. 17. Они свидетельствуют о высокой точности совпадения процессов симуляции исходной системы и системы в режиме "Программа в контуре управления".

***ЗАМЕЧАНИЕ: Описанным способом можно создать любой другой тип регулятора***.

**Практическая работа по материалу лекции 7**

Задание к практической работе по теме **«Разработка программ для программируемых логических контроллеров (PLC) и микропроцессоров»**

На основе рассмотренных в лекции методов разработки программ для программируемых логических контроллеров (PLC) и микропроцессоров и результатов предыдущих лабораторных и практических работ, выполнить разработку дискретных ПИД-регуляторов для своего варианта следящей системы управления с ДПТ независимого возбуждения. При этом должны быть выполнены все рассмотренные варианты построения регуляторов.

На компьютере должна быть установлена среда проектирования 3S CoDeSys 2.3 (3.3, 3.5), чтобы выполнить компиляцию и проверку полученной программы на языке ST. Для компиляции программы на языке C/C++ в среде MATLAB/Simulink должен быть установлен компилятор MinGW-w64 C/C++. Файлы установки **mingw.mlpkginstall и** CoDeSys\_v23941 имеются в папке к лекции (для установки CoDeSys могу направить ссылку). Для установки компилятора требуется регистрация на сайте <https://www.mathworks.com/> или личное обращение ко мне.

Сравнить результаты работы аналоговых и цифровых моделей системы управления в том числе с результатами, полученными в предыдущих практических работах.

Результаты работы оформить в виде отчета. Отчет должен содержать следующие разделы:

* Введение;
* Задание;
* Разработка моделей и программ;
* Результаты моделирования и их сравнительная оценка;
* Выводы по работе.

**Задание**

1. Используя изложенный в настоящей лекции материал в качестве методических указаний, выполнить все пункты разработки программ регуляторов для системы управления ДПТ в соответствии с вашим вариантом.
2. По результатам проектирования составить отчет о выполненной работе.