МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт кибернетики
Кафедра проблем управления

Лабораторная работа №1

Тема: «Отладка программного обеспечения робототехнических систем с использованием виртуального моделирования»

Выполнил:
Студент 4-го курса
группы КРБО-01-17
Эсаулов И. Д.

Преподаватель:
Морозов А.А

г. Москва,
2020

Цель работы

Получение навыков моделирования объекта управления в промышленных системах автоматического управления и создание функциональных блоков.

Задание

Создать виртуальную систему управления (рис. 1 .1), включающую: модель объекта управления (рис. 1 .2), ПИ-регулятор (рис. 1 .3), сумматор и обратную связь. Передаточная функция объекта:

□=1 / (ke*(Tm*s+1))

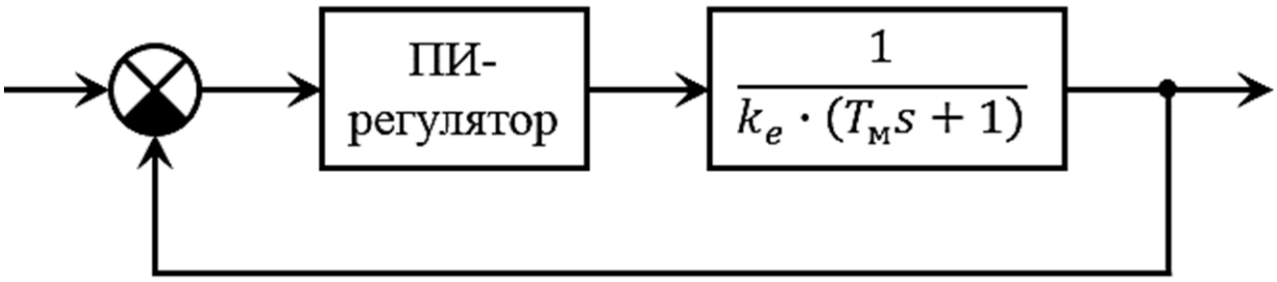

Рис. 1 - Структура системы управления

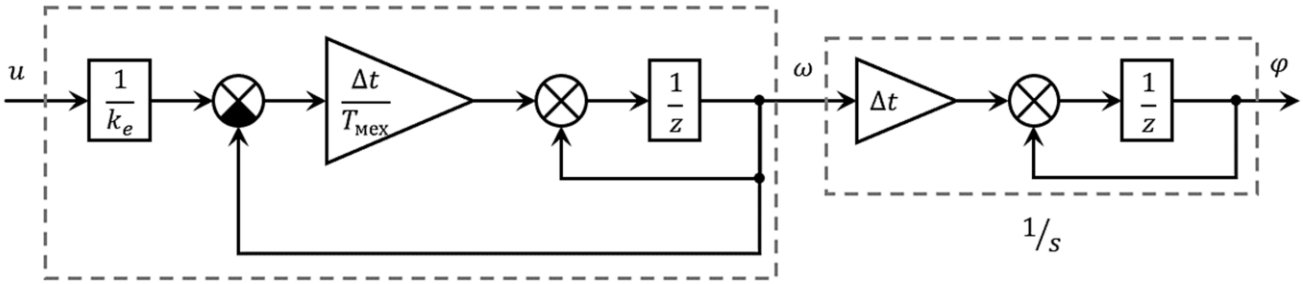

Рис. 2 - Структура объекта управления

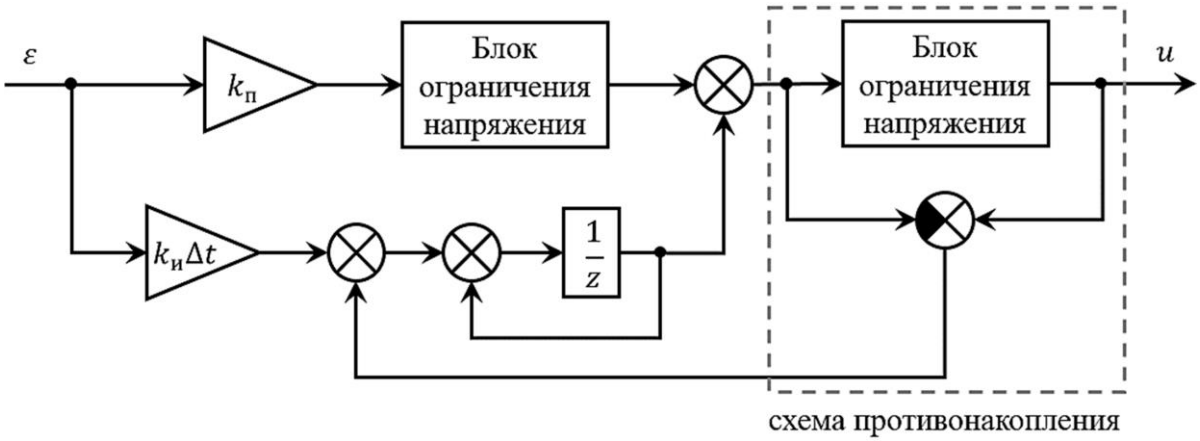

Рис. 3 - Структура ПИ-регулятора

Отчет по выполненному заданию

1 Часть

Создается проект в AS. Собирается схема из стандартных модулей для работы с учебным стендом. Конфигурация оборудования представлена на рисунке 4.

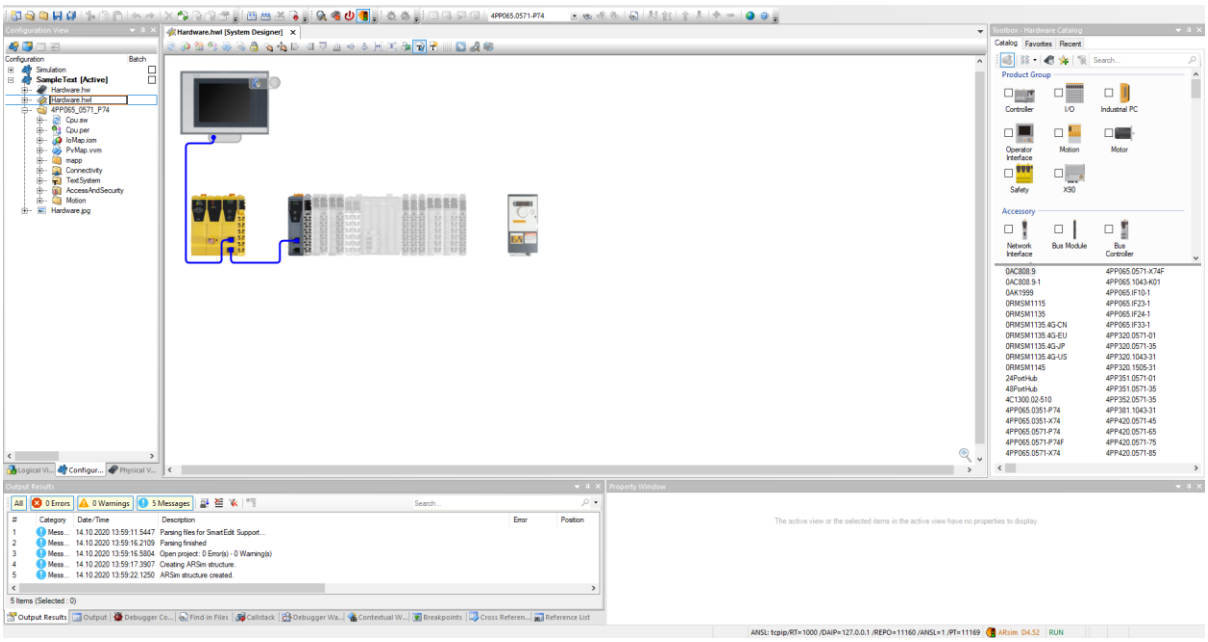

Рис. 4 - Конфигурация оборудования

В библиотеке инициализируются 3 функциональных блока с названиями FB_Motor, FB_Regulator и FB_Integrator (рис. 5).

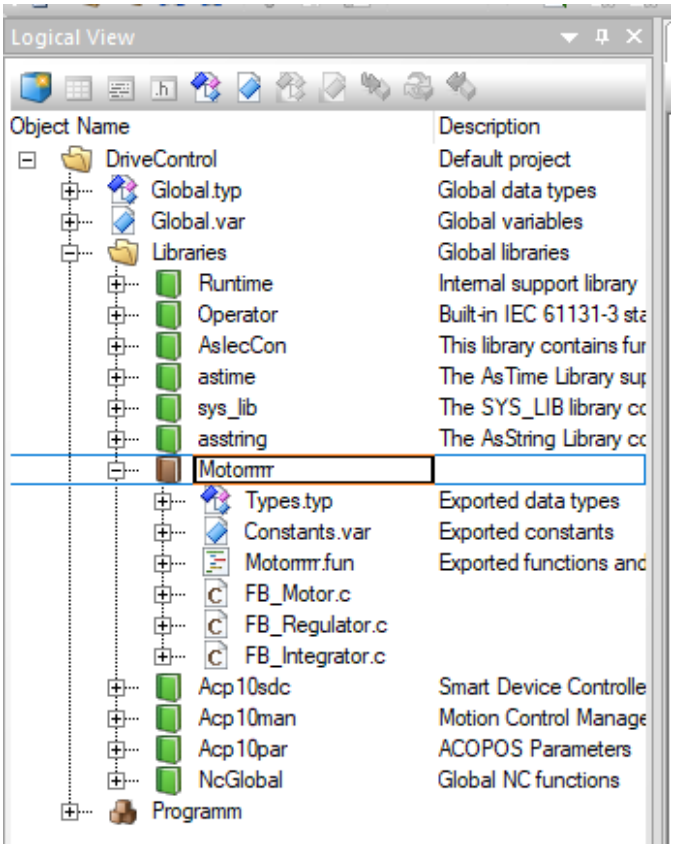

Рис. 5 - Функциональные блоки

Функциональный блок FB_Integrator моделирует поведение Интегратора. Его основная задача заключается в накоплении сумм разностей входного и выходного значений в соответствии с шагом расчета. На вход принимается значения интегрирующего звена, роль которого выполняет блок FB_Motor.

FB_Motor является основным элементом в системе, который имитирует работу двигателя постоянного тока. На вход модели подается напряжение, на выходе блок выдает частоту вращения двигателя.

Блок FB_Regulator соответствует модели ПИ-регулятора. Он обеспечивает ограничение напряжения, поступающего на двигатель, в целях обеспечения стабильности его работы. На вход подается значение рассогласования между задающим воздействием и реальной скоростью вращения вала ДПТ. На выходе значение напряжения, подаваемое на вход ДПТ.

2 Часть

В созданных функциональных блоках инициализируются переменные для дальнейшего использования последних в программном коде, который фактически имитирует поведение каждого из задуманных по заданию элемента двигателя постоянного тока (рис. 6).

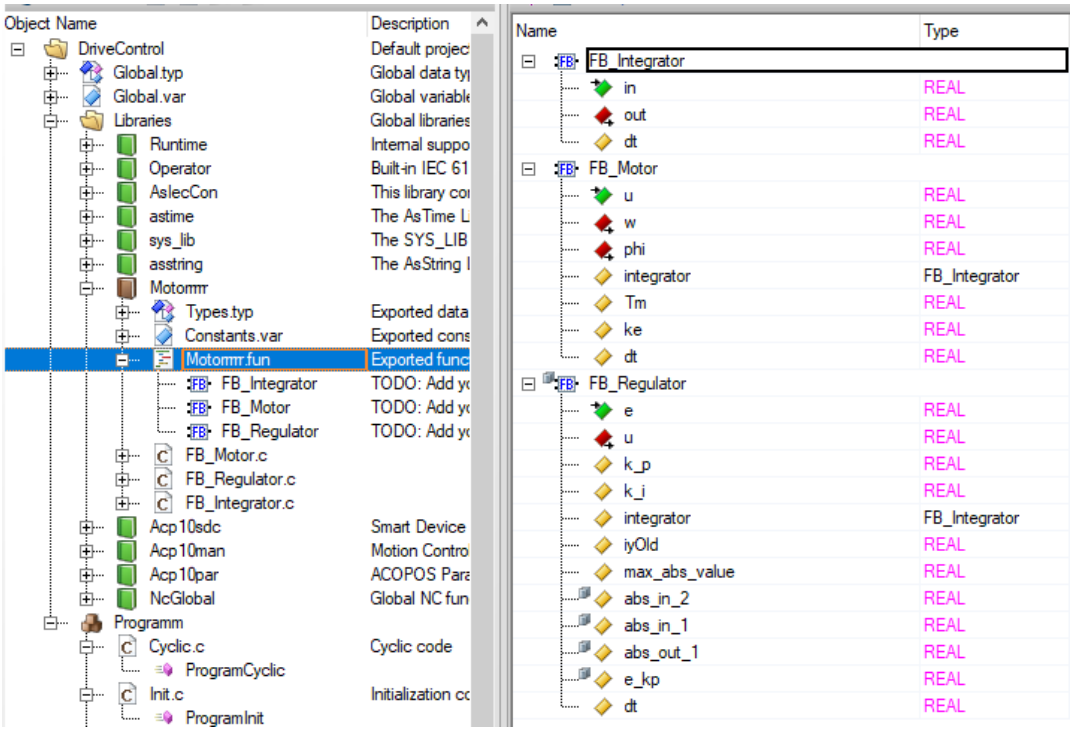

Рис. 6 - Переменные функциональных блоков

В таблицах 1,2 и 3 находится пояснение значений инициализированных переменных.

Таблица 1 - Переменные FB_Integrator

Конфигурация	**Имя**	**Тип данных**	**Описание**
вход	in	REAL	вход интегрирующего звена
выход	out	REAL	выход интегрирующего звена
внутреннее состояние	dt	REAL	шаг расчета [c]

Таблица 2 - Переменные FB_Motor

Конфигурация	**Имя**	**Тип данных**	**Описание**
вход	u	REAL	входное напряжение [В]
выход	w	REAL	частота вращения [об/мин]
выход	phi	REAL	положение [рад]
внутреннее состояние	integrator	FB _Integrator	интегратор
внутреннее состояние	Tm	REAL	электромеханическая постоянная времени [с]
внутреннее состояние	ke	REAL	постоянная ЭДС двигателя [В•мин/об]
внутреннее состояние	dt	REAL	шаг расчета [с]

Таблица 3 - Переменные FB_Regulator

Конфигурация	**Имя**	**Тип данных**	**Описание**
вход	e	REAL	рассогласование между задающим воздействием и реальной скоростью вращения вала ДПТ [об/мин]
выход	u	REAL	напряжение, подаваемое на вход ДПТ [В]
внутреннее состояние	k_p	REAL	пропорциональный коэффициент регулятора
внутреннее состояние	k_i	REAL	интегральный коэффициент регулятора
внутреннее состояние	integrator	FB_Integrator	интегратор
внутреннее состояние	iyOld	REAL	хранение предыдущего значения схемы противонакопления

внутреннее состояние	max_abs_value	REAL	граница блока ограничения [В]
внутреннее состояние	dt	REAL	шаг расчета [с]

С помощью формул (1) Метода Обратной Задачи Динамики, приведенных в методичке, производится расчет значения коэффициентов регулятора.

$$\Delta p=1/(T_ж*\Delta*\Delta o) \Longrightarrow \Delta p=(Tm*s+1)/(T_ж\cdot\Delta\cdot(1/ke)) =$$

$$ke*(\Delta|\Delta o|*\Delta*\Delta+1)/(T_ж*\Delta) = (ke*Tm)/T_ж+ke/(T_ж\cdot\Delta)$$

(1)

Из формул выше следует, что:

$$K_П = ke*Tm/T_ж, \quad (2)$$

$$\Delta и=ke*T_ж. \quad (3)$$

В расчете используются следующие коэффициенты:

dt = 0.002 [с];
Ke =2 [В*мин/об];
Tm = 0.4 [с];
T_ж = 0,1 [с];
шаг расчета регулятора =0.002 [с] ;
cnt = 0;
max_abs_value = 60.

Коэффициенты регулятора принимают следующие значения:

k_p = 8 (пропорциональный коэффициент регулятора);
k_i = 20 (интегральный коэффициент регулятора).

3 Часть

В директории основной программы задаются постоянные основных созданных функциональных блоков. Также для реализации системы управления вводятся переменные speed (установка по скорости) и enable (активатор). Они используются в процессе управления моделью (рис. 7).

Name	Type	Reference	Constant	Retain	Replicable	Value
fb_controller	FB_Regulator	☐	☐	☐	☑	(0)
fb_motor	FB_Motor	☐	☐	☐	☑	(0)
fb_motor2	FB_Motor	☐	☐	☐	☑	(0)
speed	REAL	☐	☐	☐	☑	0.0
enable	BOOL	☐	☐	☐	☑	FALSE
count	REAL	☐	☐	☐	☑	0.0

Рис. 7 - Переменные основной программы

Создано 2 экземпляра мотора, так как необходимо сравнить 2 варианта изменения переменной двигателя с регулятором и без него.

4 Часть

Пишется программа с подачей ступенчатого воздействия на отлаженную систему. Добавляется счетчик counter для реализации алгоритма ступенчатого воздействия. Объединяются функциональные блоки в систему управления. Результаты показателей входного воздействия, или скорости вращения (speed) двигателя без регулятора (fb_motor2) и двигателя с регулятором (fb_motor), снятых средством Trace, представлены на рисунке 8:

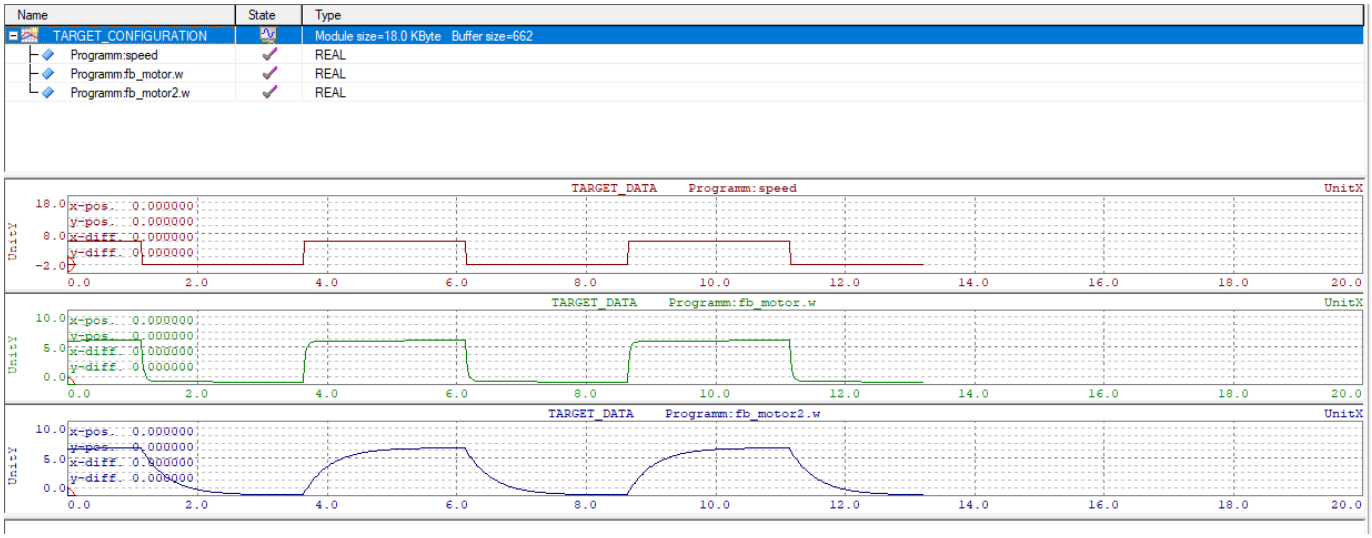

Рис. 8 - График изменения уставки по скорости (speed), скорости двигателя с регулятором (fb_motor) и двигателя без регулятора (fb_motor2)

Вывод

В ходе работы были получены навыки использования функциональных блоков и последующего построения объекта управления, моделирующего двигатель постоянного тока. Также были сопоставлены модели управления двигателя с регулятором и без регулятора. Как и ожидалось при использовании регулятора двигатель имеет лучшие параметры перерегулирования по сравнению с обычным прямым управлением.

Листинг программы 1

```
#include &lt;bur/plctypes.h&gt;
#ifdef _DEFAULT_INCLUDES
#include &lt;AsDefault.h&gt;
#endif

void _INIT ProgramInit(void)
{
enable=1;
count=0;

fb_motor.dt = 0.01;
fb_motor.ke=2;
fb_motor.Tm=0.4;
fb_motor2.dt=0.01;
fb_motor2.ke=2;
fb_motor2.Tm=0.4;
fb_controller.dt=0.01;
fb_controller.k_p=8;
fb_controller.k_i=20;
fb_controller.max_abs_value=24;
}
```

Листинг программы 2

```
#include &lt;bur/plctypes.h&gt;
#ifdef _DEFAULT_INCLUDES
#include &lt;AsDefault.h&gt;
#endif

void _CYCLIC ProgramCyclic(void)
{
if(enable)
{
count += 10;
if(count&lt;=5000)
{
speed=0;
}
else
{
speed=6;
if(count&gt;=10000)
{
count=0;
}
}
}
fb_controller.e=speed-fb_motor.w;
fb_motor2.u=speed*fb_motor2.ke;
FB_Regulator(&amp;fb_controller);
fb_motor.u=fb_controller.u*fb_motor.ke;
FB_Motor(&amp;fb_motor);
FB_Motor(&amp;fb_motor2);
}
```

Листинг программы 3:

```
#include &lt;bur/plctypes.h&gt;
#ifdef __cplusplus
extern &quot;C&quot;
{
#endif
#include &quot;MotrContr.h&quot;
#ifdef __cplusplus
};
#endif
void FB_Integrator(struct FB_Integrator* inst)
{
inst-&gt;out += inst-&gt;in;
}

#include &lt;bur/plctypes.h&gt;
#ifdef __cplusplus
extern &quot;C&quot;
{
#endif
#include &quot;MotrContr.h&quot;
#ifdef __cplusplus
};
#endif
void FB_Motor(struct FB_Motor* inst)
{
inst-&gt;integrator.in = (inst-&gt;u / inst-&gt;ke - inst-&gt;w) * inst-&gt;dt /

inst-&gt;Tm;
FB_Integrator(&amp;(inst-&gt;integrator));
inst-&gt;w = inst-&gt;integrator.out;
inst-&gt;integrator.in = inst-&gt;w * inst-&gt;dt;
FB_Integrator(&amp;(inst-&gt;integrator));
inst-&gt;phi = inst-&gt;integrator.out;
}

#include &lt;bur/plctypes.h&gt;
#ifdef __cplusplus
extern &quot;C&quot;
{
#endif
#include &quot;MotrContr.h&quot;
#ifdef __cplusplus
};
#endif
void FB_Regulator(struct FB_Regulator* inst)
{
inst-&gt;integrator.in=inst-&gt;e * inst-&gt;k_i * inst-&gt;dt + inst-&gt;iyOld;

FB_Integrator(&amp;(inst-&gt;integrator));

inst-&gt;e_kp=inst-&gt;e*inst-&gt;k_p;
```

```
inst-&gt;e_kp=(inst-&gt;e_kp &gt; inst-&gt;max_abs_value || inst-&gt;e_kp &lt; - inst-
&gt;max_abs_value)?((inst-&gt;e_kp&gt;0)?inst-&gt;max_abs_value:-inst-
&gt;max_abs_value):inst-&gt;e_kp;

inst-&gt;e_kp+=inst-&gt;integrator.out;
inst-&gt;u=(inst-&gt;e_kp &gt; inst-&gt;max_abs_value || inst-&gt;e_kp &lt; - inst-
&gt;max_abs_value)?((inst-&gt;e_kp&gt;0)?inst-&gt;max_abs_value:-inst-
&gt;max_abs_value):inst-&gt;e_kp;

inst-&gt;iyOld=inst-&gt;u-inst-&gt;e_kp;
}
```

Ссылка на репозиторий

https://github.com/Elllijah/BR_lab1/tree/master