|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

**Институт кибернетики**

**Кафедра проблем управления**

**Лабораторная работа №1**

по дисциплине «Программное обеспечение мехатронных и робототехнических систем»

**Тема:** Отладка программного обеспечения робототехнических систем с использованием виртуального моделирования

Работу выполнили

студенты КРБО-01-17:

Бабанский А.А.

Болотских А.Ю.

Денисов Д.С.

Селезнев П.А.

Руководитель:

Морозов А.А.

**Цель работы**: получение навыков моделирования объекта управления в промышленных системах автоматического управления и создание функциональных блоков.

**Задание**: создать виртуальную систему управления (рис. 1), включающую: модель объекта управления (рис. 2), ПИ-регулятор (рис. 3), сумматор и обратную связь.

Передаточная функция объекта:

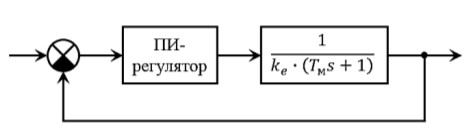


Рис. 1 – Структура системы управления

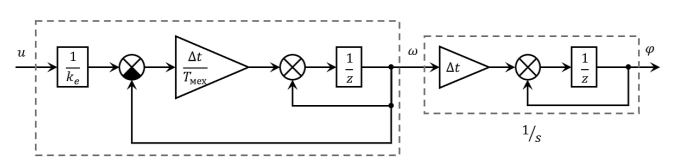


Рис. 2 – Структура объекта управления

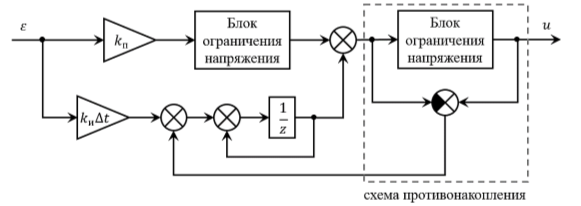


Рис. 3 – Структура ПИ-регулятора

**Ход работы**:

Был создан новый проект в среде Automation Studio без конфигурации оборудования.

В библиотеке «MotorCtrl» были созданы три функциональных блока:

1. «FB\_Motor». Затем была реализована модель двигателя постоянного тока (приложение 1). Описание структуры приведено в таблице 1.

**Таблица 1 – Параметры функционального блока 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] |

1. «FB\_Integrator». Работа интегратора заключается в накоплении сумм разностей входного и выходного значений в соответствии с шагом расчета. Листинг программы приведен в приложении 2, описание структуры - в таблице 2.

**Таблица 2 – Параметры функционального блока 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 [с] |

Структурная схема блока интегратора (рис. 4) имеет следующий вид:

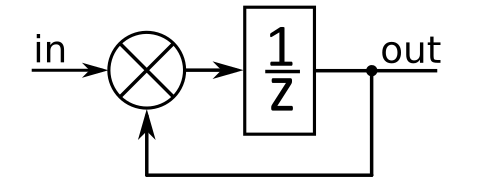


Рис.4 – Структурная схема интегратора

1. «FB\_Regulator». Расчет значения на выходе блока происходит в соответствии со схемой ПИ-регулятора (рис. 3). Схема противонакопления используется для ограничения напряжения, поступающего на двигатель, в целях обеспечения стабильности его работы.

Также в схеме присутствуют два ограничителя напряжения: после П-звена и выходного напряжения. Величина ограничения задается с помощью внутреннего состояния Umax.

Расчет коэффициентов произведен с помощью метода обратной задачи динамики:

Пусть *,* тогда, коэффициенты регулятора равны:

Листинг программы приведён в приложении 3, описание параметров структуры - в таблице 3.

**Таблица** **3 – Параметры функционального блока 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] |

Таким образом, в проект были добавлены функциональные блоки, параметры которых отображены на рис. 5.

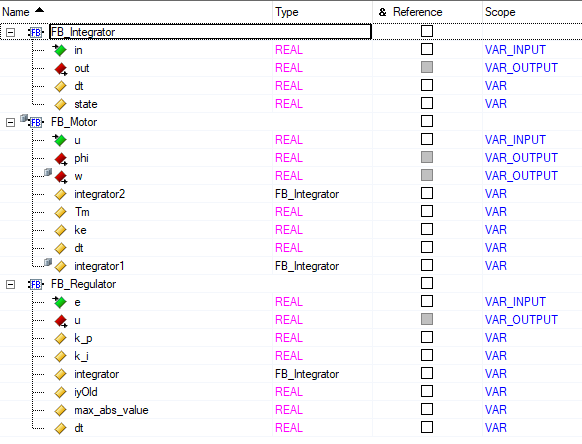


Рис. 5 – Описание функциональных блоков в среде Automation Studio

Далее была разработана программа Program1. Её алгоритм заключается в подаче на систему управления двигателя уставки по скорости равной 50 об/мин в течение длительного импульса.

В эксперименте используются два двигателя: без регулятора и с регулятором.

Для снятия показаний использовалось средство диагностики «Trace». В качестве параметров слежения были взяты следующие переменные: «speed», «fb\_motor.w», хранящая скорость двигателя с регулятором, «fb\_motor2.w», хранящая скорость двигателя без регулятора. Результат моделирования приведён на рис. 6.

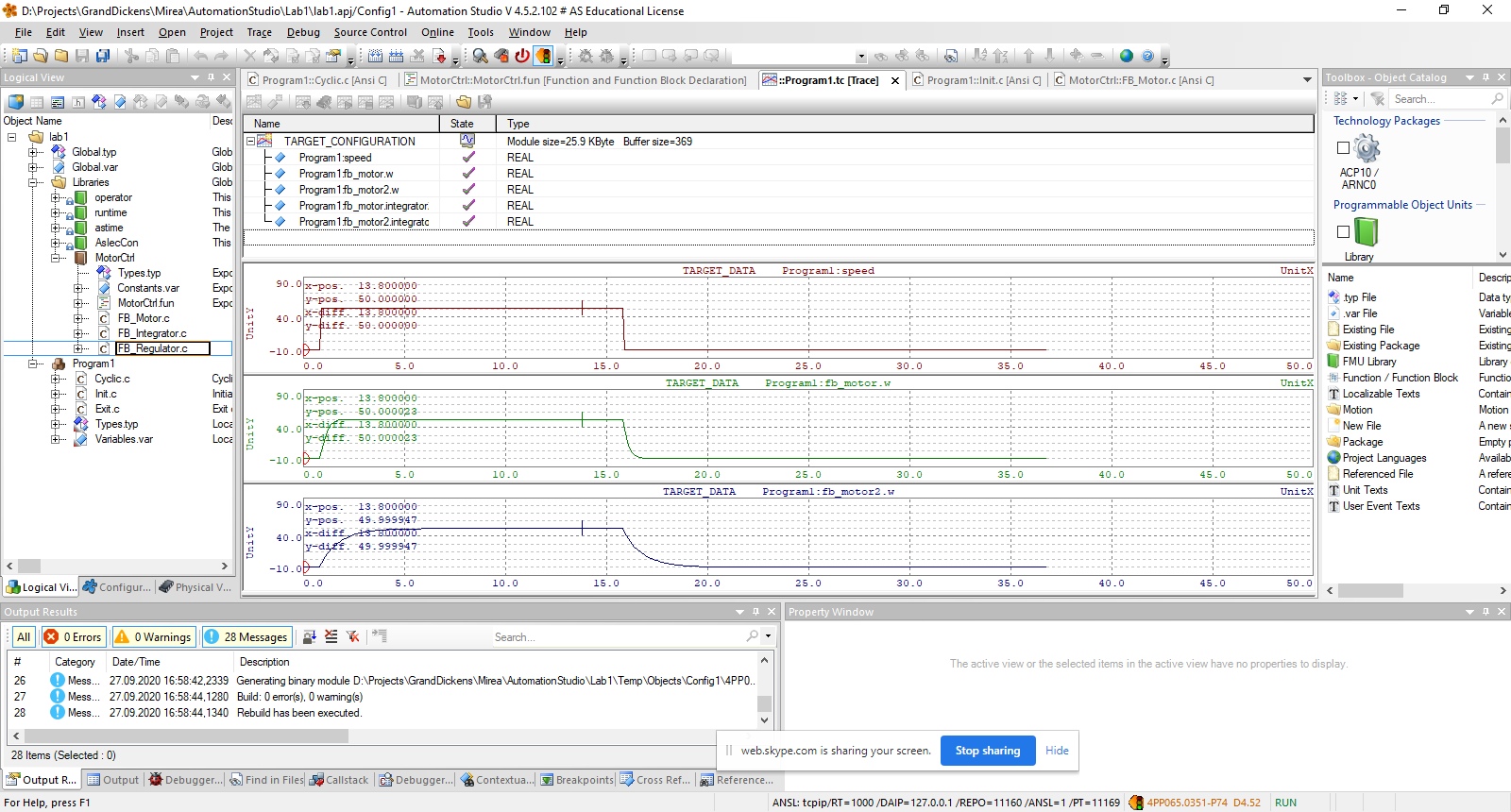


Рис. 6 – Результаты моделирования

**Вывод**: были получены навыки отладки во время моделирования объекта управления в промышленных системах автоматического управления и создание функциональных блоков. Использование регулятора повышает быстродействие системы управления.

**Приложение 1**

**Листинг программы функционального блока FB\_Motor**

#include <bur/plctypes.h>  
#ifdef \_\_cplusplus  
 extern "C"  
 {  
#endif  
 #include "MotorCtrl.h"  
#ifdef \_\_cplusplus  
 };  
#endif  
  
void FB\_Motor(struct FB\_Motor\* inst)  
{  
  
 REAL a = inst->u / inst->ke - inst->w;  
   
 inst->integrator1.in = a;  
 FB\_Integrator(&inst->integrator1);  
 inst->w = inst->integrator1.out;  
   
 inst->integrator2.in = inst->w;  
 FB\_Integrator(&inst->integrator2);  
 inst->phi = inst->integrator2.out;  
}

**Приложение 2**

**Листинг программы функционального блока FB\_Integrator**

#include <bur/plctypes.h>  
#ifdef \_\_cplusplus  
 extern "C"  
 {  
#endif  
 #include "MotorCtrl.h"  
#ifdef \_\_cplusplus  
 };  
#endif

void FB\_Integrator(struct FB\_Integrator\* inst)  
{  
 inst->out = inst->dt \* inst->in + inst->state;  
 inst->state = inst->out;  
}

**Приложение 3**

**Листинг программы функционального блока FB\_Regulator**

#include <bur/plctypes.h>  
#ifdef \_\_cplusplus  
 extern "C"  
 {  
#endif  
 #include "MotorCtrl.h"  
#ifdef \_\_cplusplus  
 };  
#endif

void FB\_Regulator(struct FB\_Regulator\* inst)  
{  
 REAL a = inst->e \* inst->k\_p;  
 REAL b = inst->e \* inst->k\_i;  
 a = a > inst->max\_abs\_value ? inst->max\_abs\_value : a;  
 a = a < -inst->max\_abs\_value ? -inst->max\_abs\_value : a;  
 inst->integrator.in = b + inst->iyOld;  
 FB\_Integrator(&inst->integrator);  
 REAL sum = a + inst->integrator.out;  
 inst->u = sum;  
 inst->u = inst->u > inst->max\_abs\_value ? inst->max\_abs\_value : inst->u;  
 inst->u = inst->u < -inst->max\_abs\_value ? -inst->max\_abs\_value : inst->u;  
 inst->iyOld = inst->u - sum;  
}

**Приложение 4**

**Листинг основной программы**

#include <bur/plctypes.h>  
  
#ifdef \_DEFAULT\_INCLUDES  
 #include <AsDefault.h>  
#endif  
  
void \_INIT ProgramInit(void)  
{  
 fb\_motor.dt = 0.01;  
 fb\_motor.ke = 7.2;  
 fb\_motor.Tm = 0.1;  
 fb\_motor.u = 0;  
 fb\_motor.phi = 0;  
 fb\_motor.integrator1.dt = 0.01 / fb\_motor.Tm;  
 fb\_motor.integrator2.dt = 0.01;   
 fb\_motor2.dt = 0.01;  
 fb\_motor2.ke = 7.2;  
 fb\_motor2.Tm = 0.1;  
 fb\_motor2.u = 0;  
 fb\_motor2.phi = 0;  
 fb\_motor2.integrator1.dt = 0.01 / fb\_motor2.Tm;  
 fb\_motor2.integrator2.dt = 0.01;   
 fb\_regulator.dt = 0.01;  
 fb\_regulator.integrator.dt = 0.01;  
 fb\_regulator.k\_i = 7.2;  
 fb\_regulator.k\_p = 0.24 \* 3;  
 fb\_regulator.max\_abs\_value = 24.0;  
   
 speed = 0;  
 counter = 0;  
}

void \_CYCLIC ProgramCyclic(void)  
{  
 if (enable)   
 {  
 if (counter == 200)   
 {  
 speed = 0;  
 }  
 else if (counter == 50)   
 {  
 speed = 50;  
 }  
 if (counter < 201)  
 counter++;  
   
 fb\_regulator.e = speed-fb\_motor.w;  
 FB\_Regulator(&fb\_regulator);  
 fb\_motor.u = fb\_regulator.u;  
 fb\_motor2.u = speed \* fb\_motor2.ke;  
 FB\_Motor(&fb\_motor);  
 FB\_Motor(&fb\_motor2);  
 }   
}

void \_EXIT ProgramExit(void)  
{  
  
}