|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждениевысшего профессионального образования"МИРЭА – Российский технологический университет"РТУ МИРЭА |
| Факультет Кибернетики |
| Кафедра Проблем управления |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1** | |
| **по дисциплине** | |
| **«**Программное обеспечение мехатронных и робототехнических систем**»** | |
|  | |
| Выполнил студент группы КРБО-02-16 | Архипов В. |
| Принял | Морозов А.А. |

|  |  |  |
| --- | --- | --- |
| Лабораторные работы выполнены | «\_\_»\_\_\_\_\_\_\_201\_\_ г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*  *(подпись студента)* |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_201\_\_ г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*  *(подпись руководителя)* |

Москва 2019

# Лабораторная работа №1

**Отладка программного обеспечения роботехнических систем с использованием виртуального моделирования**

**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:

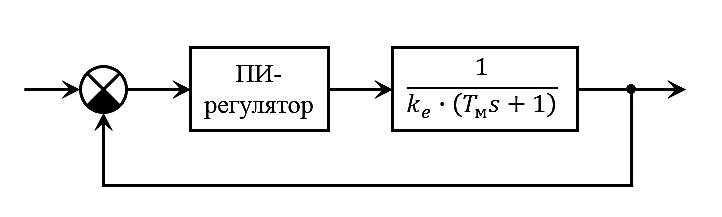


Рис. 1 – Структура системы управления

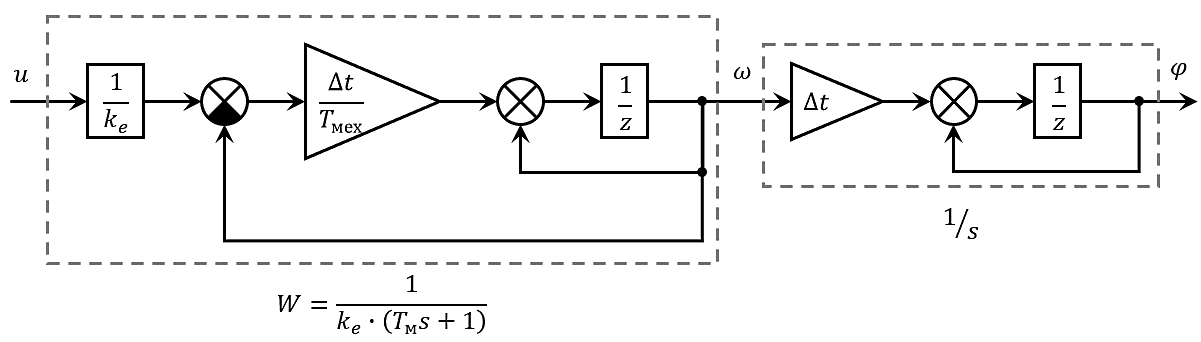


Рис. 2 – Структура объекта управления

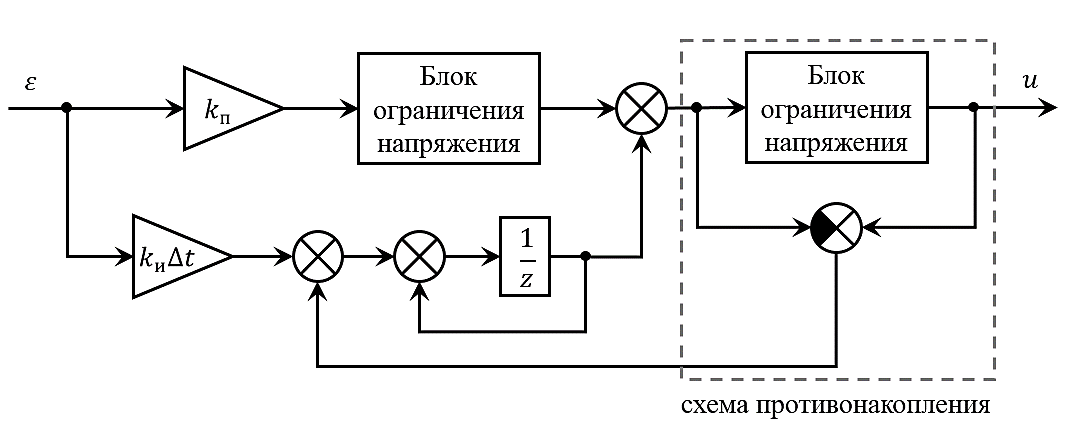


Рис. 3 – Структура ПИ-регулятора

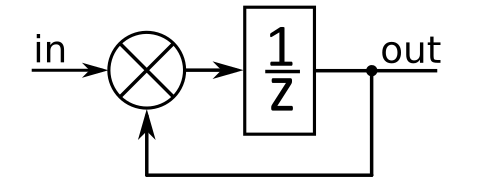
**Создание проекта и реализация моделей объектов:**

1. Создаем проект, куда добавляем объекты ANSI C Program; ANSI C Library. Работать будем с функциональными блоками – особыми структурами (подпрограммами), напоминающие классы в ООП. В ANSI C Library добавляем функциональные блоки:

* FB\_Motor – для реализации модели двигателя постоянного тока. На входе – напряжение. На выходе – частота вращения и положение.
* FB\_Regulator – данный функциональный блок будет выполнять роль ПИ-регулятора. На входе – значение рассогласования между задающим воздействием и реальной скоростью вала ДПТ. На выходе – напряжение, подаваемое на вход ДПТ.
* FB\_Integrator – блок, представляющий собой модель интегратора. На вход принимает значения с интегрирующего звена. Выход – соответственное значение с интегрирующего звена.

1. Создание моделей объектов. Для создания функциональных блоков мотора и регулятора нужен интегратор.

На структурных схемах блок интегратора представляет собой следующую структуру (Рис. 4):

*Рис. 4 – Структура блока интегратора*

**Таблица 1 – 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] |

Логика работы интегратора заключается в накоплении сумм разностей входного и выходного значений в соответствии с шагом расчета. Это надо реализовать в программном коде данного функционального блока. Расчет значения на выходе данного функционального блока можно получить из передаточной функции интегратора путем Z-преобразования.

Структурная схема блока мотора представлена на Рис. 2.

**Таблица** **2 – Параметры функционального блока FBMotor**

|  |  |  |  |
| --- | --- | --- | --- |
| **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] |

Структурная схема блока регулятора представлена на Рис. 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] |

1. Расчет коэффициентов регулятора производится с помощью метода обратной задачи динамики (ОЗД). Примем .

Следовательно, коэффициенты регулятора:

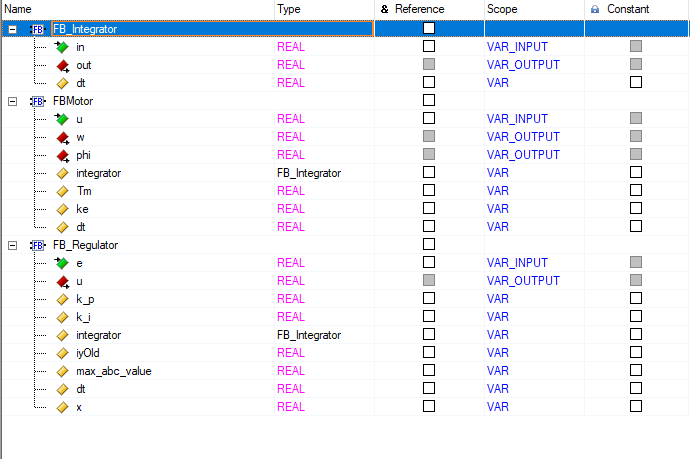
Для дальнейшего расчета примем следующие коэффициенты:

* шаг расчета [с] ДПТ dt = 0.002;
* постоянная ЭДС двигателя [В•мин/об] Ke =2;
* электромеханическая постоянная времени [с] Tm = 0.4;
* желаемая постоянная времени [с] T\_ж = dt \* 3 = 0.006;
* значение integrator для FB\_Motor dt = 0.002;
* шаг расчета [с] регулятора =0.01с;
* пропорциональный коэффициент регулятора k\_p=0.0064;
* интегральный коэффициент регулятора k\_i=0.16;
* граница блока ограничения [В] (максимальное напряжение) max\_abs\_value = 240;
* уставка по скорости меняется с 0 на 45 и наоборот каждые 6с;
* cnt = 0;
* speed 0-45.

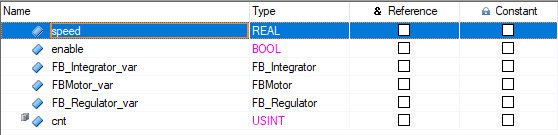
1. После реализации функциональных блоков объединяем объект и регулятор в систему управления (Рис. 1) в основной программе с применением разработанных функциональных блоков;

**Таблица 4 – Переменные основной программы**

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип данных** | **Описание** |
| fb\_controller | FB\_Controller | рассогласование между задающим воздействием и реальной скоростью вращения вала ДПТ [об/мин] |
| fb\_motor | FB\_Motor | напряжение, подаваемое на вход ДПТ [В] |
| speed | REAL | уставка по скорости |
| enable | BOOL | интегральный коэффициент регулятора |

**Переменные функциональных блоков (Рис 5)

*Рис. 5 – Переменные функциональных блоков*

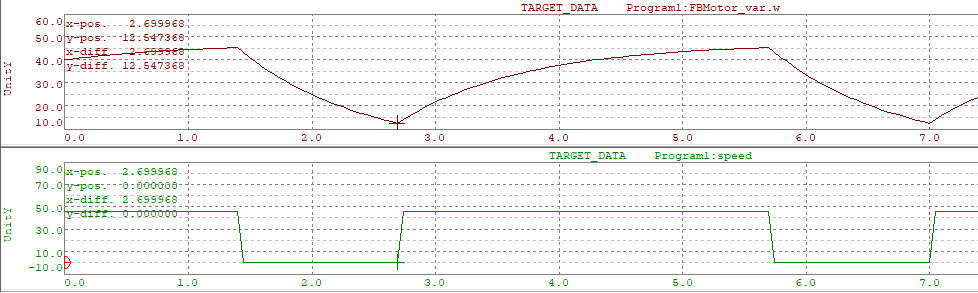
Переменные основной программы (Рис. 6)

*Рис. 6 – Переменные основной программы*

**Результаты тестирования:**

1. На Рис. 7 представлены графики переходного процесса системы без регулятора:

верхний – зависимость выходной частоты вращения от времени;

нижний – зависимость уставки от времени;

*Рис. 7 – Графики переходного процесса системы без регулятора*

1. На Рис. 8 представлены графики переходного процесса с регулятором с исходными параметрами k\_i и k\_p:

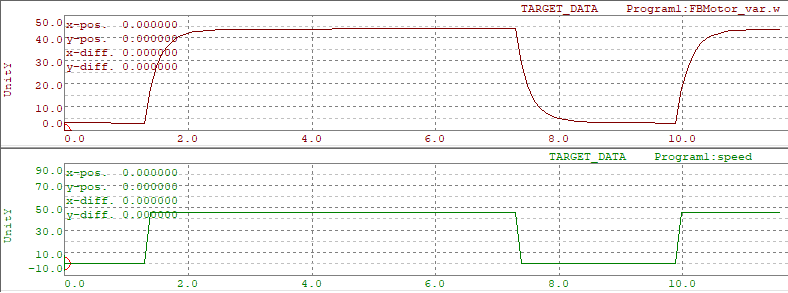
верхний – зависимость выходной частоты вращения от времени;

нижний – зависимость уставки от времени;

*Рис. 8 – Графики переходного процесса системы (исходные k\_i ; k\_p)*

1. На Рис. 9 представлены графики переходного процесса с регулятором с параметрами 0,2\*k\_i и k\_p:

верхний – зависимость выходной частоты вращения от времени;

нижний – зависимость уставки от времени;

*Рис. 9 – Графики переходного процесса системы (0,2\*k\_i ; k\_p)*

1. На Рис. 10 представлены графики переходного процесса с регулятором с параметрами 5\*k\_i и k\_p:

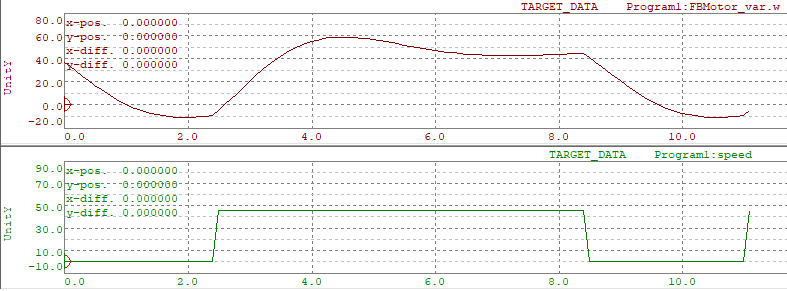
верхний – зависимость выходной частоты вращения от времени;

нижний – зависимость уставки от времени;

*Рис. 10 – Графики переходного процесса системы (5\*k\_i ; k\_p)*

1. На Рис. 11 представлены графики переходного процесса с регулятором с параметрами k\_i и 0,2\*k\_p:

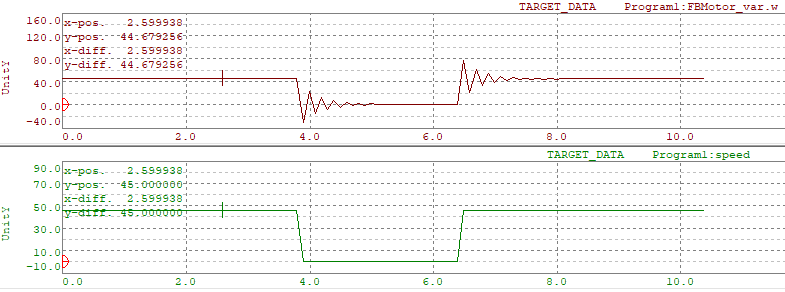
верхний – зависимость выходной частоты вращения от времени;

нижний – зависимость уставки от времени;

*Рис. 11 – Графики переходного процесса системы (k\_i ; 0,2\*k\_p)*

1. На Рис. 12 представлены графики переходного процесса с регулятором с параметрами k\_i и 5\*k\_p:

верхний – зависимость выходной частоты вращения от времени;

нижний – зависимость уставки от времени;

*Рис. 12 – Графики переходного процесса системы (k\_i ; 5\*k\_p)*

**Вывод:**

В ходе выполнения лабораторной работы была создана модель системы управления двигателями с регулятором и без регулятора. Преимущество системы управления с регулятором над системой управления без регулятора заключается в том, что регулятор позволяет контролировать некоторые параметры переходного процесса (например время прихода в установившийся режим) . Таким образом система с регулятором входит в установившийся за указанное время.

**Приложение**

Листинг кода блока интегратора:

#include <bur/plctypes.h>

#ifdef \_\_cplusplus

extern "C"

{

#endif

#include "Library1.h"

#ifdef \_\_cplusplus

};

#endif

/\* TODO: Add your comment here \*/

void FB\_Integrator(struct FB\_Integrator\* inst)

{

inst->out = inst->in + inst->out;

}

Листинг кода блока двигателя:

#include <bur/plctypes.h>

#ifdef \_\_cplusplus

extern "C"

{

#endif

#include "Library1.h"

#ifdef \_\_cplusplus

};

#endif

/\* TODO: Add your comment here \*/

void FBMotor(struct FBMotor\* inst)

{

inst->integrator.in = (inst->u/inst->ke - inst->w) \* (inst->dt/inst->Tm);

FB\_Integrator(&inst->integrator);

inst->w = inst->integrator.out;

inst->integrator.in = inst->w \* inst->dt;

FB\_Integrator(&inst->integrator);

inst->phi = inst->integrator.out;

}

Листинг кода блока ПИ-регулятора:

#include <bur/plctypes.h>

#ifdef \_\_cplusplus

extern "C"

{

#endif

#include "Library1.h"

#ifdef \_\_cplusplus

};

#endif

/\* TODO: Add your comment here \*/

void FB\_Regulator(struct FB\_Regulator\* inst)

{

float BON(float x)

{

if(x < 0)

{

if(x < -inst->max\_abc\_value)

return -inst->max\_abc\_value;

else

return x;

}

else

{

if(x > inst->max\_abc\_value)

return inst->max\_abc\_value;

else

return x;

}

}

inst->integrator.in = inst->e \* inst->k\_i \* inst->dt + inst->iyOld;

FB\_Integrator(&inst->integrator);

inst->u = BON((BON(inst->e \* inst->k\_p) + inst->integrator.out));

inst->iyOld = inst->u - (BON(inst->e \* inst->k\_p) + inst->integrator.out);

}

Листинг кода системы управления:

#include <bur/plctypes.h>

#ifdef \_DEFAULT\_INCLUDES

#include <AsDefault.h>

#endif

void \_INIT ProgramInit(void)

{

FB\_Integrator\_var.in = 0;

FB\_Integrator\_var.out = 0;

FB\_Integrator\_var.dt = 0.002;

FBMotor\_var.u = 0;

FBMotor\_var.w = 0 ;

FBMotor\_var.ke = 0.16\*FB\_Integrator\_var.dt\*3 ;

FBMotor\_var.Tm = (0.0064\*FB\_Integrator\_var.dt\*3)/FBMotor\_var.ke;

FBMotor\_var.phi = 0;

FBMotor\_var.dt = FB\_Integrator\_var.dt;

FB\_Regulator\_var.e = 0;

FB\_Regulator\_var.u = 0;

FB\_Regulator\_var.k\_p = 0.0064 ;

FB\_Regulator\_var.k\_i = 0.16;

FB\_Regulator\_var.dt = FB\_Integrator\_var.dt;

FB\_Regulator\_var.max\_abc\_value = 240;

FB\_Regulator\_var.iyOld = 0;

speed = 0;

enable = 1;

cnt=0;

}

void \_CYCLIC ProgramCyclic(void)

{

if(cnt> 25 && cnt<=85)

{

speed = 45;

}

else

speed = 0;

if(cnt>85)

cnt=0;

cnt++;

FB\_Regulator\_var.e = speed - FBMotor\_var.w;

FB\_Regulator(&FB\_Regulator\_var);

FBMotor\_var.u = FB\_Regulator\_var.u;

FBMotor(&FBMotor\_var);

/\*

FBMotor\_var.u = speed \* FBMotor\_var.ke;

FBMotor(&FBMotor\_var);\*/

}