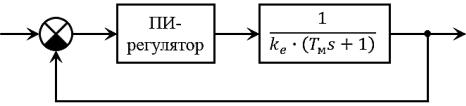
**Лабораторная работа № 1.**

**Отладка программного обеспечения робототехнических систем с**

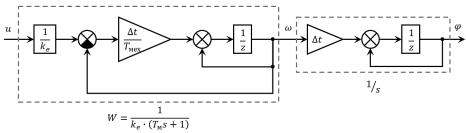
**использованием виртуального моделирования.**

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

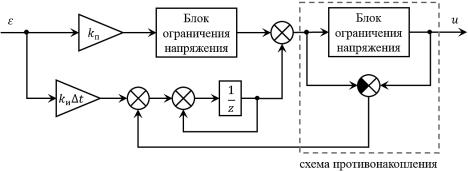
**Задание**:создать виртуальную систему управления(рис. 1.1), включающую: модель объекта управления (рис. 1.2), ПИ-регулятор (рис. 1.3), сумматор и обратную связь. Передаточная функция объекта:



**Рис. 1.1 Структура системы управления.**



**Рис. 1.2 Структура объекта управления.**



**Рис. 1.3 Структура ПИ-регулятора.**

**Описание действий и использованного аппарата:**

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:

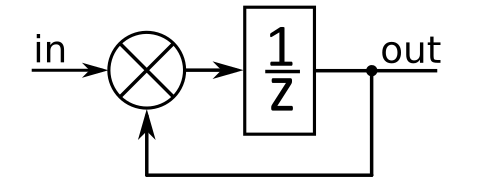
FB\_Motor – для реализации модели двигателя постоянного тока. На входе – напряжение. На выходе – частота вращения и положение.

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 схема блок интегратора представляет собой следующую структуру:

**

**Рисунок** **1.4 – Структурная схема интегратора**

**Таблица 1. Параметры функционального блока FB\_Integrator**

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

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.

**Таблица 2. 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] |

|  |  |  |  |
| --- | --- | --- | --- |
| **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] |
| внутреннее состояние | e\_kp | REAL | промежуточное значение вычислений |

**Таблица 3. Параметры функционального блока FB\_Regulator**

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

Следовательно, коэффициенты регулятора:

=20

Для дальнейшего расчета примем следующие коэффициенты:

шаг расчета [с] ДПТ dt = 0.01c

постоянная ЭДС двигателя [В•мин/об] Ke =2

электромеханическая постоянная времени [с] Tm = 0.4

значение integrator для FB\_Motor dt = 0.01

шаг расчета [с] регулятора =0.01с

пропорциональный коэффициент регулятора k\_p=8

интегральный коэффициент регулятора k\_i=20

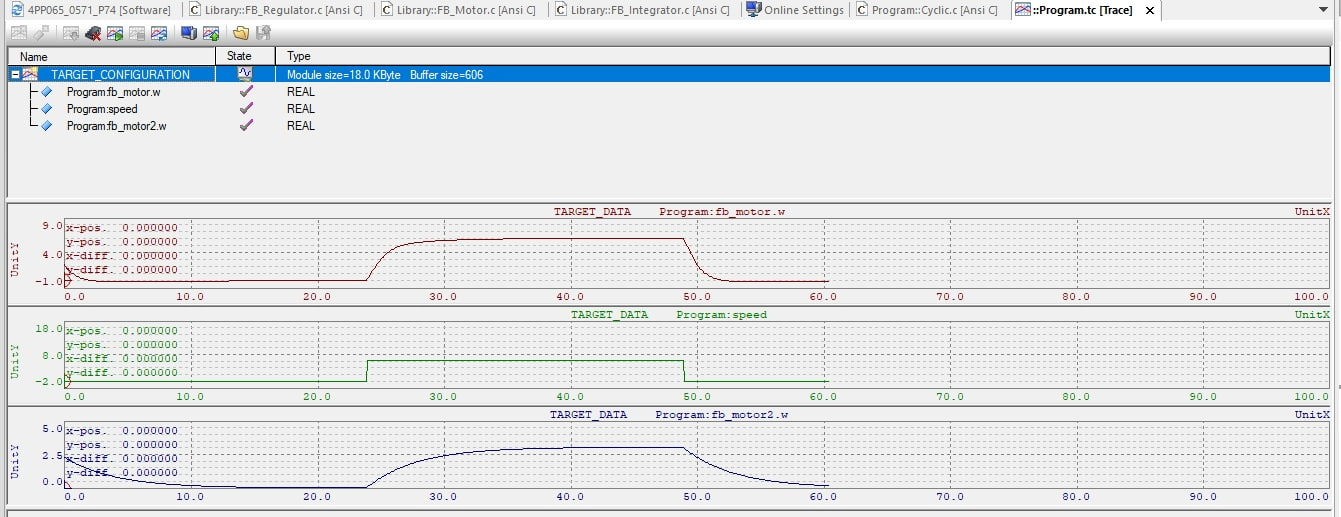
граница блока ограничения [В] (максимальное напряжение) max\_abs\_value = 24

уставка по скорости меняется с 0 на 6 и наоборот каждые 2,5с

count = 2500; 5000;

speed 0-6

1. Графики



**Рисунок 1.5**

На рис.1.5 приведён результат в виде трёх графиков.

В верхнем графике приведен переходной процесс, который показывает зависимость скорость вращения без регулятора от времени. В среднем графике приведен переходной процесс, который показывает зависимость уставки от времени. В нижнем - приведен переходной процесс, который показывает зависимость скорость вращения с регулятором от времени.

**Вывод**:

В ходе выполнения лабораторной работы была создана модель системы управления двигателями с регулятором и без регулятора. По результату можно сказать, что с двигатель с регулятором не достигает установленной скорости, так как Tжел в регуляторе меньше чем время выполнения в двигателе. Также были получены навыки моделирования объектов управления в промышленных системах автоматического управления, а также навыки создания функциональных блоков.

**Приложение.**

Листинг кода блока интегратора.

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

{

inst->out+=inst->in;

}

Листинг кода блока двигателя.

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

{

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

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

FB\_Integrator(&(inst->integrator));

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

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

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

FB\_Integrator(&(inst->integrator));

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

}

Листинг кода блока ПИ-регулятора.

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

{

inst->integrator.in=inst->e \* inst->k\_i \* inst->dt + inst->iyOld; FB\_Integrator(&(inst->integrator)); inst->e\_kp=inst->e\*inst->k\_p;

inst->e\_kp=(inst->e\_kp > inst->max\_abs\_value || inst->e\_kp < - inst->max\_abs\_value)?((inst->e\_kp>0)?inst->max\_abs\_value:-inst->max\_abs\_value):inst->e\_kp;

inst->e\_kp+=inst->integrator.out;

inst->u=(inst->e\_kp > inst->max\_abs\_value || inst->e\_kp < - inst->max\_abs\_value)?((inst->e\_kp>0)?inst->max\_abs\_value:-inst->max\_abs\_value):inst->e\_kp;

inst->iyOld=inst->u-inst->e\_kp;

}

Листинг кода системы управления.

void \_INIT ProgramInit(void)

{

fb\_motor.dt=0.01;

fb\_motor.Ke=2;

fb\_motor.Tm=0.4;

fb\_motor.integrator.dt=0.01;

fb\_motor2.dt=0.01;

fb\_motor2.Ke=2;

fb\_motor2.Tm=0.4;

fb\_motor2.integrator.dt=0.01;

fb\_regulator.dt=0.01;

fb\_regulator.k\_p=8;

fb\_regulator.k\_i=20;

fb\_regulator.max\_abs\_value=24;

fb\_regulator.integrator.dt=0.01;

enable=1;

count=0;

}

Листинг кода для создания ступенчатого воздействия.

void \_CYCLIC ProgramCyclic(void)

{

if(enable)

{

count+=10;

if(count<=2500)

{

speed=0;

}

else

{

speed=6;

if(count>=5000)

{

count=0;

}

}

}

fb\_regulator.e=speed - fb\_motor.w;

fb\_motor2.u=speed;

FB\_Regulator(&fb\_regulator);

fb\_motor.u=fb\_regulator.u;

FB\_Motor(&fb\_motor);

FB\_Motor(&fb\_motor2);

}