МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт искусственного интеллекта (ИИИ)

Кафедра проблем управления

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

по дисциплине «Программное обеспечение мехатронных и робототехнических систем»

Выполнили студенты группы КРБО-03-22 Филичкина М.Д.

Рабышев Д.А.

Рольнов М.Ю.

Сафиулин Д.Р.

Проверил Морозов А.А.

Москва 2025

Программное обеспечение системы управления одной степенью робота УРТК

Цель работы: получение навыков создания программного обеспечения систем управления одной степенью подвижности учебного робота.

Задание: создать функциональный блок, основанный на библиотеке Acp10sdc, который будет осуществлять управление одной осью УРТК (см. рисунок 1). Включая обработку концевых датчиков, реферирование к датчику, расположенному со стороны мотора (после реферирования ось переходит в состояние «отключено»). Управление реализовать из теста или с кнопок стенда.

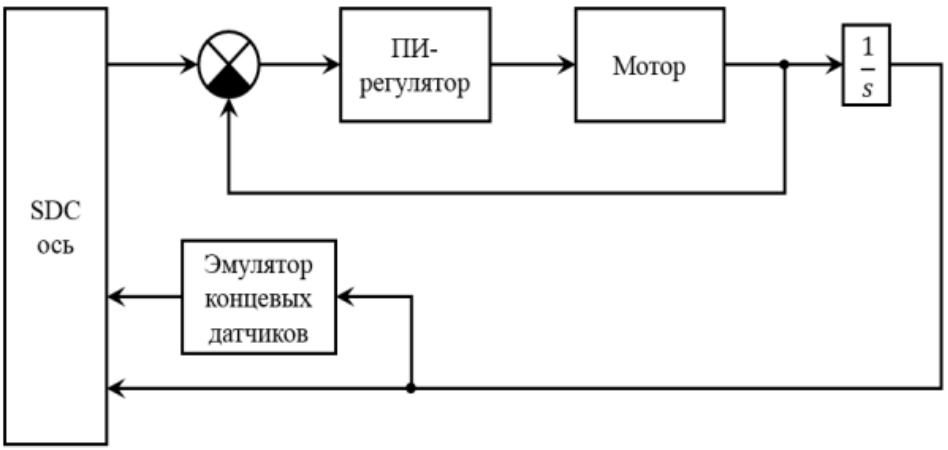

Рисунок 1. Структура системы управления одной степенью подвижности робота.

Ход работы

Запустили B&R Automation Studio и добавили следующие модули и интерфейсы:

- 5LS182.6-1;
- X20BC0083;
- X20MM4456
- X20DI9371;
- X20D09322;
- 80VD100PS.C02X-01.

Конфигурация при работе с контроллером стенда аналогична конфигурации в лабораторной работе №2 (рис.2).

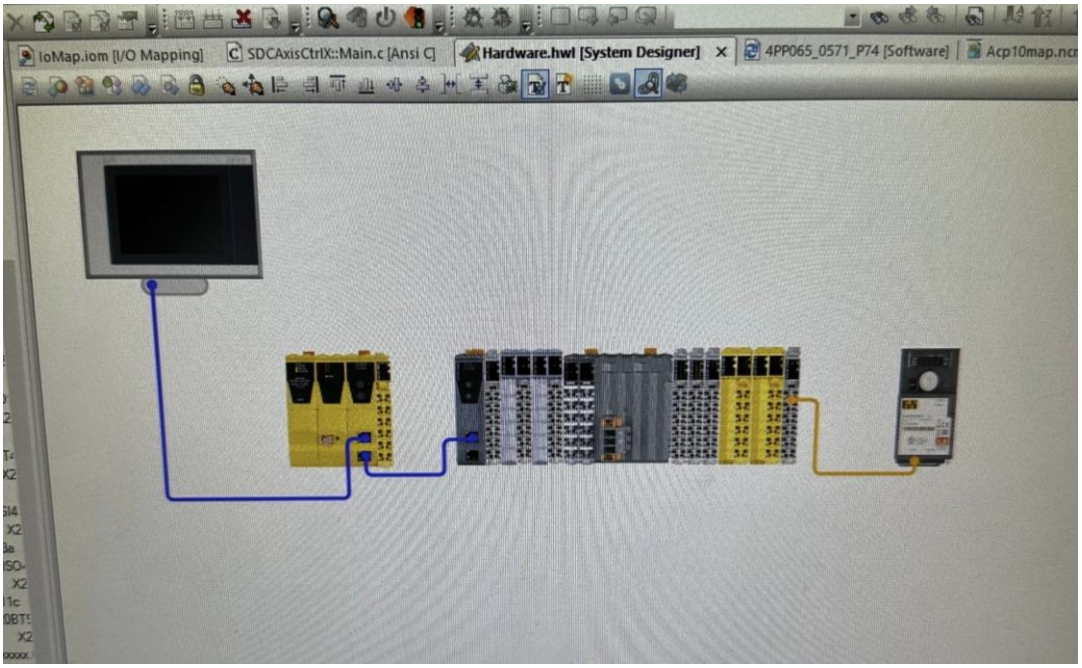

Рисунок 2. Конфигурация оборудования

Для создания SDC-оси необходимо добавить библиотеки Acp10, а также создать в конфигурации NC-мэпинг. После этого в проекте, помимо стандартного набора, появится следующий набор библиотек (рис. 3), также необходимо добавить библиотеку AsIOTime.

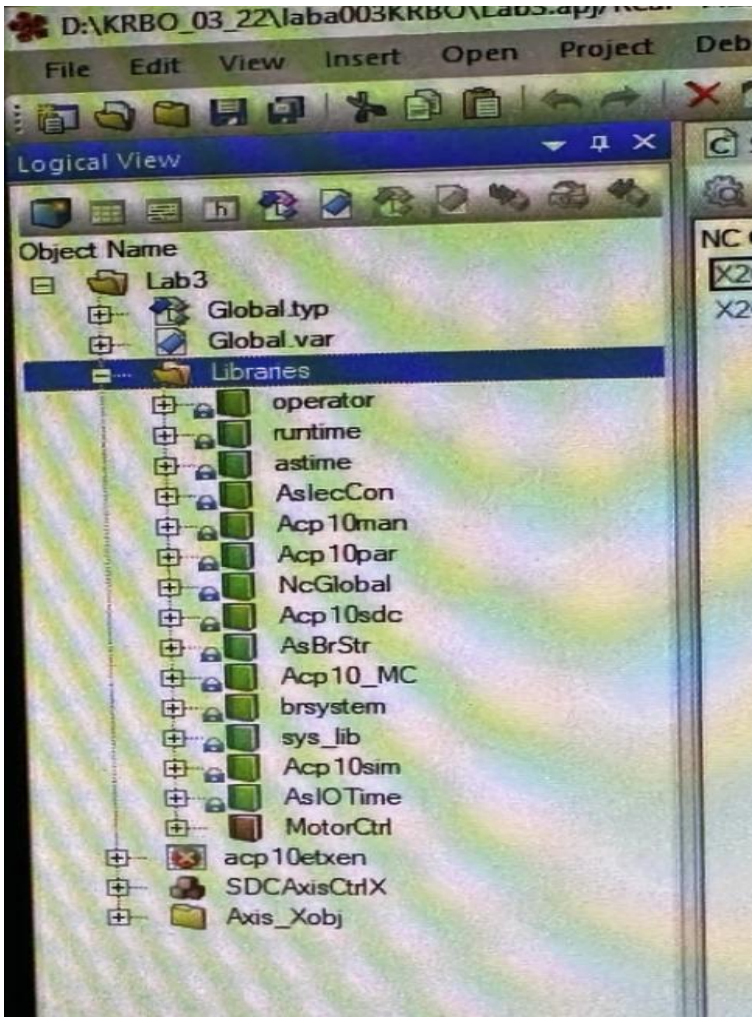

Рисунок 3. Библиотеки проекта

В Global variables автоматически созданы переменные и структуры, необходимые для работы SDC. Переименовали название оси, заменив его на Axis_X. Набор глобальных переменных должен иметь следующий вид (см. таблицу 1 и рисунок 4).

Таблица 1. Глобальные переменные взаимодействия SDC-оси.

Имя	Тип	Описание
Axis_X	ACP10AXIS_typ	Инициализация SDC-оси в главной программе
Axis_X _HW	SdcHwCfg_typ	Переменная конфигурации аппаратных средств оси
Axis_X _DrvIf	SdcDrvIf16_typ	Переменная для контроля интерфейса привода
Axis_X _DiDoIf	SdcDiDoIf_typ	Переменная для контроля цифровых входов/выходов интерфейса привода
Axis_X _EncIf	SdcEncIf16_typ	Переменная для контроля датчиков двигателя

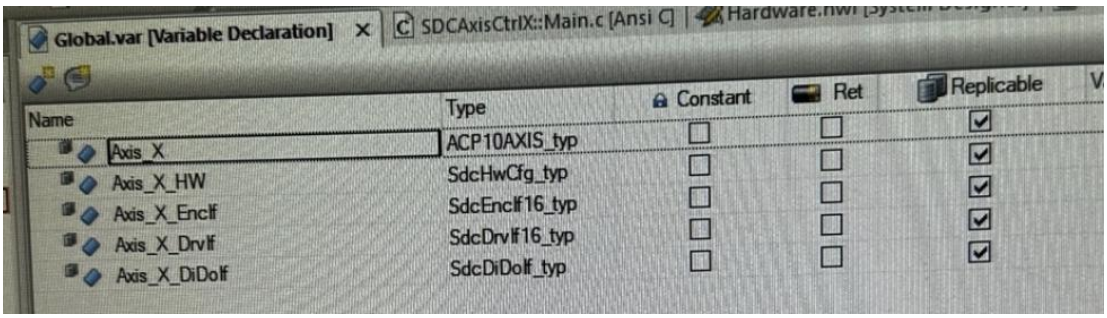

Рисунок 4. Глобальные переменные

Для создания SDC-оси необходимо добавить в папку проекта, средством Toolbox, модуль инициализации оси «ACP10 Axis» (рис.5).

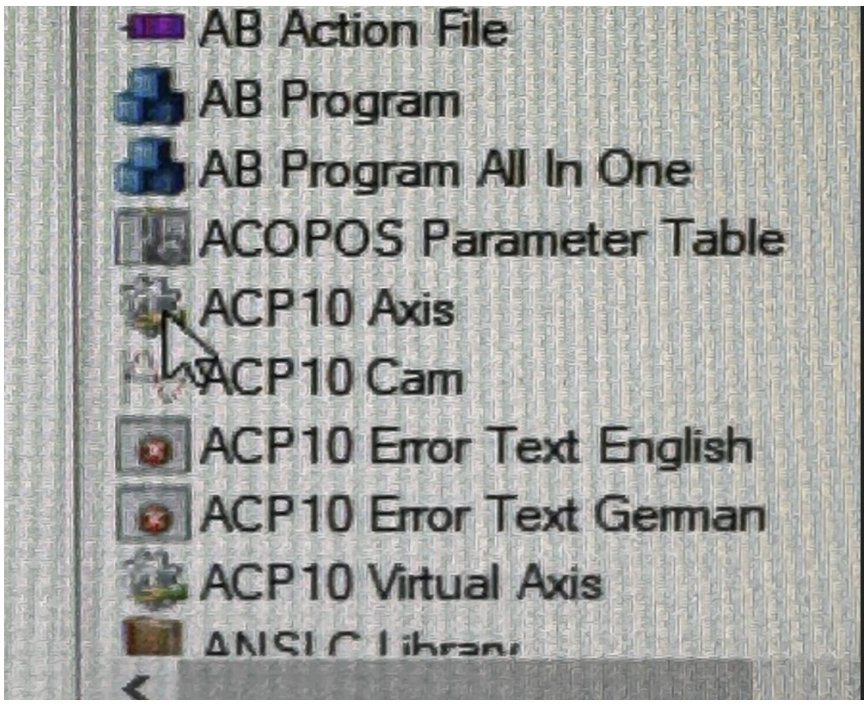

Рисунок 5. Модуль инициализации оси

Далее открываем его и в параметрах инициализации указываем реальные параметры оси УРТК (см. таблицу 2 и рисунок 6).

Таблица 2. Параметры инициализации SDC-оси

Параметр	Значение	Описание
pos_hw_end	ncACTIV_HI	Состояние концевого датчика в положительном направлении
neg_hw_end	ncACTIV_HI	Состояние концевого датчика в отрицательном направлении
Units	3000	Количество юнитов на оборот [unit]
a1_pos	10000	Ускорение в положительном направлении [unit/sl]
a2_pos	10000	Замедление в положительном направлении [unit/sl]
a1_neg	10000	Ускорение в отрицательном направлении [unit/sl]
a2_neg	10000	Замедление в отрицательном направлении [unit/sl]
ds_stop	50000.0	Ошибка по положению ведущая к остановке [unit]
ds_warning	500.0	Ошибка по положению ведущая к предупреждению [unit]
Kv	10	Коэффициент П-регулятора положения [1/s]
v_switch	6000	Начальная скорость движения к концевому датчику [unit/s]
v_trigger	2000	Скорость движения вокруг концевого датчика [unit/s]
a	10000	Ускорение [unit/sl]
Mode	ncEND_SWITCH	Режим реферирования
edge_sw	ncNEGATIVE	Режим подъезда к датчику
trigg_dir	ncNEGATIVE	
fix_dir	ncOFF	

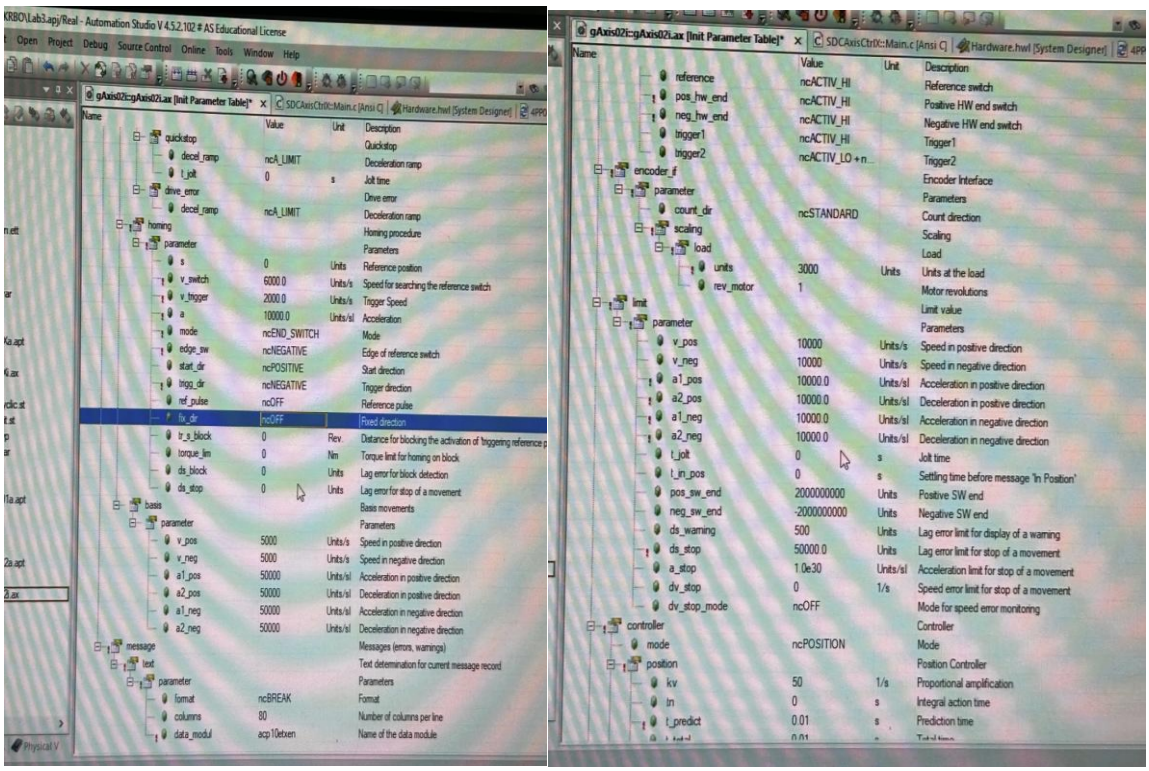

Рисунок 6. Инициализация SDC-оси

Далее средством Toolbox добавляем таблицу инициализации оси «ACOPOS Parameter table» в таблице 3 указаны все необходимые параметры, а на рисунке 7 представлена их реализация в программе.

Таблица 3. Параметры SDC-оси (AcpParTab.apt)

Имя	ID	Значение	Описание
SERVO_V_MAX_OUTPUT	64201	6500	Максимальная скорость вращения двигателя [unit/s]
SCALE_ENCOD_INCR	109	24	Число импульсов на оборот инкрементного датчика

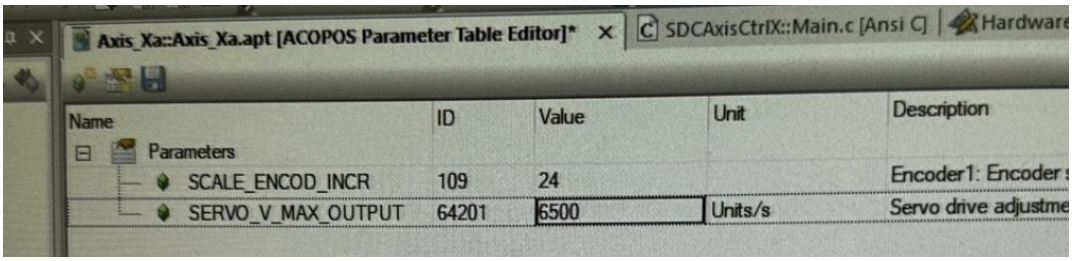

Рисунок 7. Реализация параметров SDC-оси (AcpParTab.apt)

Для мэпинга новой SDC оси рисунок 8, заполняем необходимые параметры представленные в таблице 4 для новой оси, а реализация представлена на рисунке 9.

Рисунок 8. Папка для добавления новой оси

Таблица 4 Параметры мэпинга создаваемой SDC-оси

Имя NC объекта (оси)	PLC адрес	Тип NC объекта	Таблица инициализации	Таблица параметров ACOPOS
Axis_X	SDC_IF1.ST2	ncAXIS	gAxis01i	gAxis01a

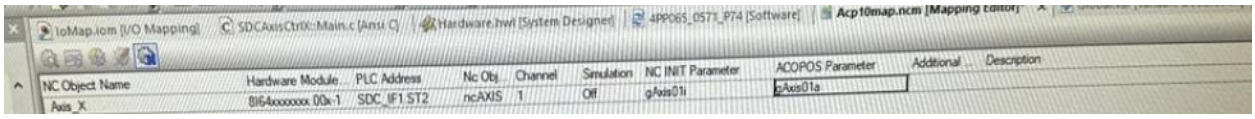

Рисунок 9. Параметры новой оси

Используя модель регулятора и объекта управления из лабораторной работы №1, разрабатываем управление одной осью робота. Функциональный блок «FB_Axis» отвечает за определение входного воздействия на ось двигателя путем задания ШИМ сигнала, параметры блока описаны в таблице 5.

Таблица 5. Параметры функционального блока FB_Axis

Конфигурация	Имя	Тип данных	Описание
вход	reset_error	BOOL	Сброс ошибок мотора
вход	endswitch_a_reached	BOOL	Состояние начального концевого датчика
вход	endswitch_b_reached	BOOL	Состояние конечного концевого датчика
выход	reset_counter	BOOL	Сброс счетчика
выход	pwm_value	INT	Время импульса ШИМ
выход	counter	INT	Счетчик импульсов
выход	speed	REAL	Скорость вращения оси двигателя
внутреннее состояние	last_counter	INT	Хранение предыдущего значения counter в процессе расчета

Реализация функциональных блоков представлена на рисунке 10.

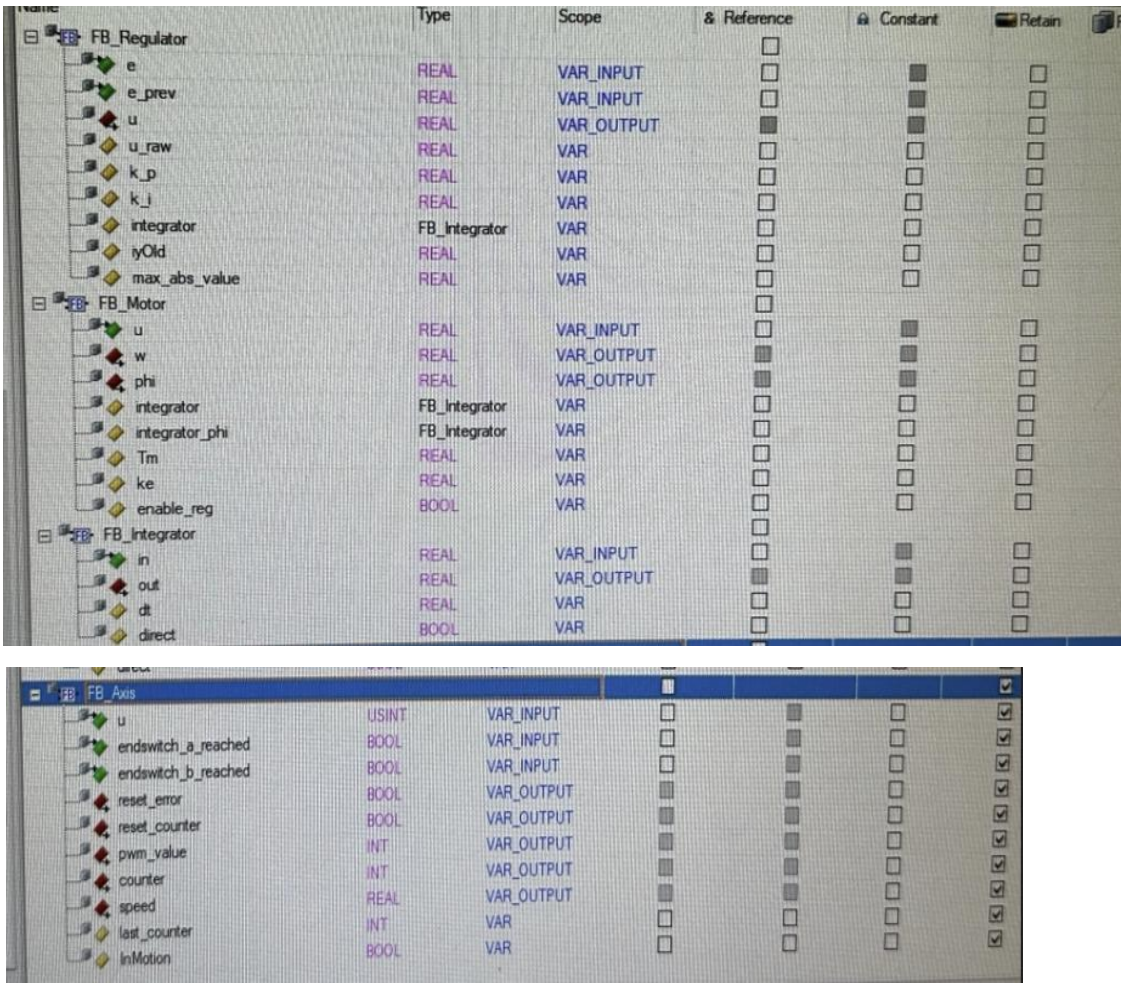

Рисунок 10. Параметры всех функциональных блоков

Создадим программы обработки одной оси с применением модернизированной библиотеки управления двигателем, для реализации этого необходимо создать ANSI C Program с названием «SDCAxisCtrlX» и добавить в нее необходимые переменные (таблица 6).

Таблица 6. Переменные программы SDCAxisCtrlX

Имя	Тип данных	Описание
axis_X	FB_Axis	Переменная оси
coil_powered	BOOL	Включение обмотки возбуждения
coli_pwm_value	INT	Время импульса ШИМ
fb_controller	FB_Controller	Переменная регулятора
pwm_period	UINT	Период ШИМ

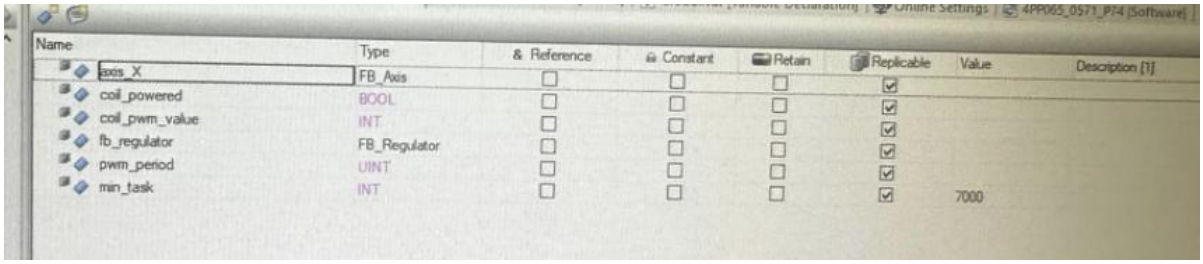

Рисунок 11. Реализация переменных программы SDCAxisCtrlX

Далее напишем программу, реализующая алгоритм, представленный на рисунке 12, где цикл программы 2мс.

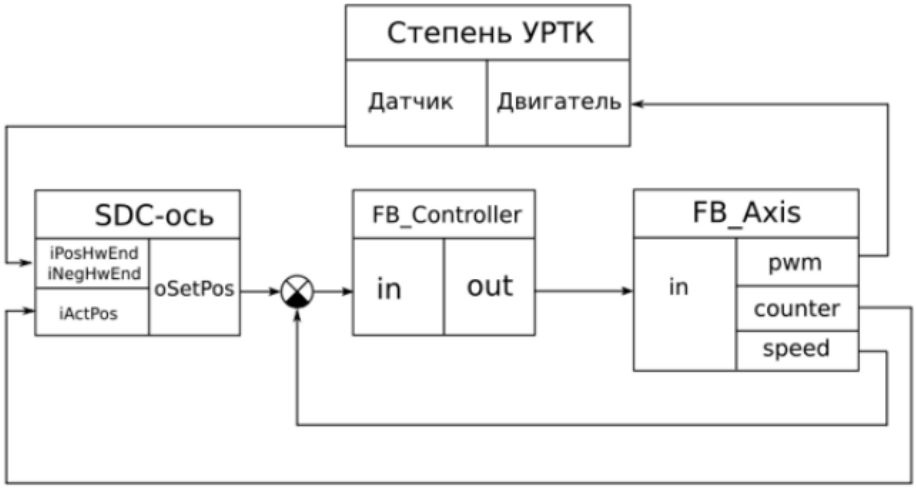

Рисунок 12. Структура программного обеспечения

Дополнительно, в части инициализации основной программы, необходимо:

- Установить параметры и входы готовности SDC-оси (листинг 1);
- Установить константы для FB_Control (аналогично ЛР №1).

Листинг 1

```
//Устанавливаем типы SDC модулей
Axis_X_HW.EncIf1_Typ = ncSDC_ENC16;
Axis_X_HW.DiDoIf_Typ = ncSDC_DIDO;
Axis_X_HW.DrvIf_Typ= ncSDC_DRVSERVO16;
//Устанавливаем имена переменных
strcpy(Axis_X_HW.EncIf1_Name, "Axis_X_EncIf");
strcpy(Axis_X_HW.DrvIf_Name, "Axis_X_DrvIf");
strcpy(Axis_X_HW.DiDoIf_Name,
"Axis_X_DiDoIf");
//Устанавливаем входы готовности и нормальной
работы
Axis_X_EncIf.iEncOK = 1;
```

```
Axis_X_DrvIf.iDrvOK = 1;
Axis_X_DrvIf.iStatusEnable = 1;
Axis_X_DiDoIf.iDriveReady = 1;
```

В основном цикле программы необходимо также:

- Инкрементировать LiveCounter-ы SDC оси (см. листинг 2), чтобы ось не падала в ошибку, указать iActTime как время начала цикла;

- Включать и отключать обмотку возбуждения в соответствии со значением переменной выключателя.

Листинг 2

```
Axis_X_EncIf.iLifeCnt++;
Axis_X_DiDoIf.iLifeCntDriveEnable++;
Axis_X_DiDoIf.iLifeCntDriveReady++;
Axis_X_DiDoIf.iLifeCntNegHwEnd++;
Axis_X_DiDoIf.iLifeCntPosHwEnd++;
Axis_X_DiDoIf.iLifeCntReference++;
Axis_X_DrvIf.iLifeCnt++;
Axis_X_EncIf.iActTime=
(INT)AsIOTimeCyclicStart();
```

Средством «Trace» 6ыли записаны графики перемещения оси (рис.13). На графиках изображены: состояние концевого датчика, текущее положение, желаемая скорость и оценка реальной текущей скорости. По графикам видно, что реальная скорость несколько оттает от уставки контроллера. Также видны погрешности, возникающие из-за дискретной природы инкрементного датчика.

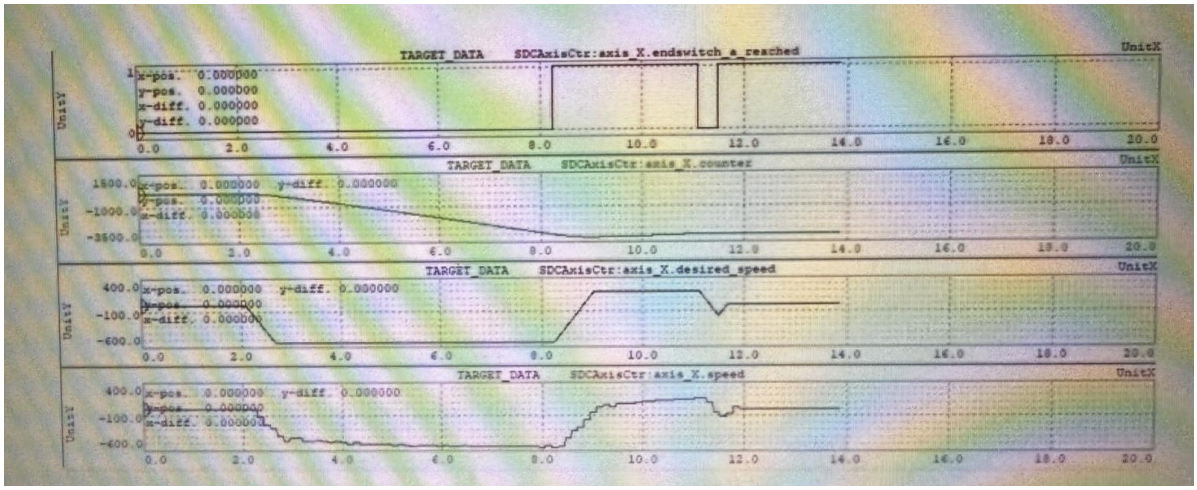

Рисунок 13. Вывод графиков

Вывод: в ходе выполнения лабораторной работы были успешно получены практические навыки создания программного обеспечения для системы управления одной степенью подвижности учебного робота на базе контроллера B&R. Была разработана и настроена SDC-ось с использованием библиотеки Acp10sdc, реализовано управление движением оси, включая обработку концевых датчиков и процедуру реферирования. Управление осуществлялось как программно, так и с помощью кнопок стенда.

В процессе работы были изучены и применены инструменты среды Automation Studio, включая конфигурирование оборудования, настройку глобальных переменных, параметров инициализации оси, а также создание функциональных блоков управления. Реализованный алгоритм управления позволил обеспечить движение оси в соответствии с заданными параметрами, однако в ходе тестирования были выявлены незначительные отклонения реальной скорости от заданной, обусловленные дискретностью инкрементного датчика и особенностями работы регулятора.

Приложение А (fb_axis)

```
#include <bur/plctypes.h>
#ifdef __cplusplus
 extern "C"
 {
#endif
 #include "Library.h"
#ifdef __cplusplus
 };
#endif
/* TODO: Add your comment here */
void FB_Axis(struct FB_Axis* inst)
{
 if (inst->endswitch_a_reached == 1 )
 {
  inst->dir = 1;
 }
 if (inst->endswitch_b_reached == 1 )
 {
  inst->dir = 0;
 }


 if(inst->dir == 1)
 {
  //inst->reset_counter = 0;
  inst->u = -inst->u;
  inst->pwm_value = (inst->u/24.0) * 32767;
 }
 if(inst->dir == 0)
 {
  //inst->reset_counter = 0;
  inst->pwm_value = (inst->u/24.0) * 32767;
 }
```

```
 if (inst->i == 1000)
 {
  inst->speed = (inst->counter - inst->last_counter)/2;
  inst->last_counter = inst->counter;
  inst->i = 0;
 }
 inst->spid = inst->speed;
 inst->i++;
}
```

Приложение B (fb_controller)

```
#include <bur/plctypes.h>
#ifdef __cplusplus
 extern "C"
 {
#endif
 #include "Library.h"
#ifdef __cplusplus
 };
#endif

void FB_Controller(struct FB_Controller* inst)
{

}
```

Приложение C (FB_endswitch)

```
#include <bur/plctypes.h>
#ifdef __cplusplus
 extern "C"
 {
#endif
 #include "Library.h"
#ifdef __cplusplus
 };
#endif
/* TODO: Add your comment here */
void FB_Endswitch(struct FB_Endswitch* inst)
{



}
```

Приложение D (FB_integrator)

```
#include <bur/plctypes.h>
#ifdef __cplusplus
 extern "C"
 {
#endif
 #include "Library.h"
#ifdef __cplusplus
 };
#endif
/* TODO: Add your comment here */
void FB_Integrator(struct FB_Integrator* inst)
{
 inst->out = inst->dt * inst->in + inst->state;
 inst->state = inst->out;
}
```

Приложение E (FB_regulator)

```
#include <bur/plctypes.h>
#ifdef __cplusplus
 extern "C"
 {
#endif
 #include "Library.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);
}
```