Национальный исследовательский университет «МЭИ»

Отчет по курсовому проекту.

«Системы управления электроприводов»

Выполнил:	Одозиободо Ифеаньи
Проверил:	Шпак Дмитрий Михайлович
Группа:	ЭЛ-01м-24
Оценка:	
Подпись преподавателя:	

Москва 2023

Оглавление

1. **Задание на курсовой проект.**

Вариант №1 (Асинхронный двигатель)

Вариант системы управления: 1

Исходные данные:

Число пар полюсов	1
Сопротивление резистора измерительной цепи датчика тока	82
Число первичных витков при коэффициенте датчика тока 1000:1	5
Коэффициент преобразования сигнала ТГ, (об/мин)/АЦП	2
Количество импульсов на механический оборот квадратурного датчика положения ротора	10000

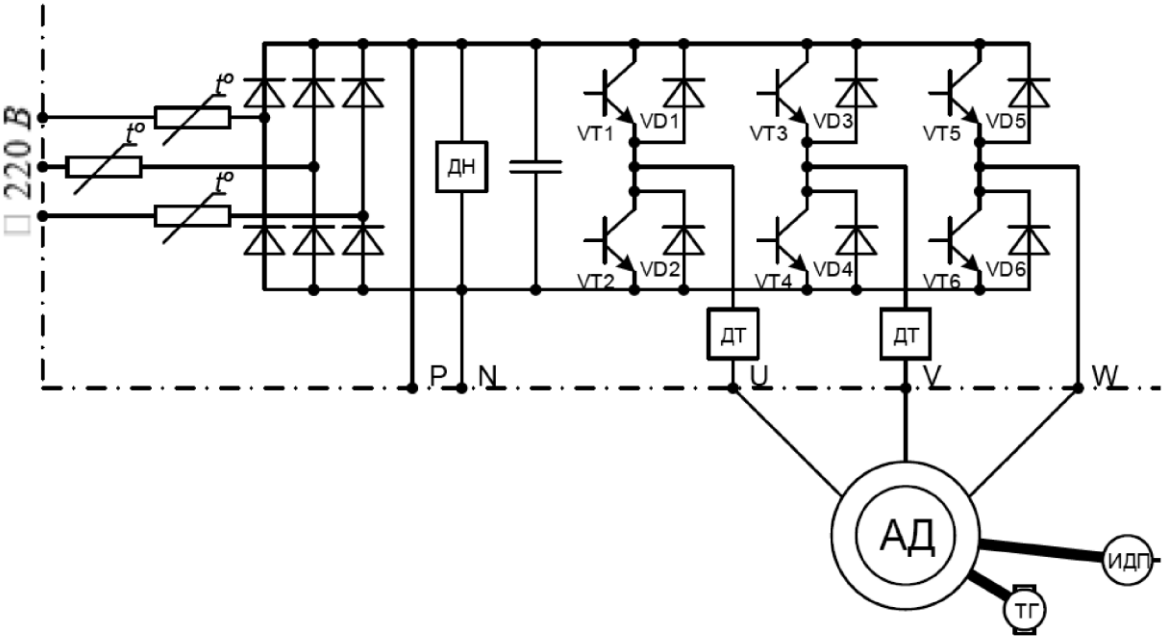

Рисунок 1 Схема исследуемого асинхронного двигателя, подключенного к преобразователю.

2. Перевод показаний АЦП в физические единицы.

1.1 Определение тока в фазах.

Для измерения токов в фазах асинхронного двигателя используется два трансформатора тока в фазах A и B, ток в фазе C вычисляется из условия $i_A + i_B + i_c = 0$.

Ток во вторичной обмотке трансформатора определяется по формуле:

$$i_2 = i_1 \frac{w_1}{w_2}$$

Тогда, напряжение на входе АЦП можно вычислить по формуле:

$$u_2 = R_{\text{изм}} \cdot i_2 = R_{\text{изм}} \cdot i_1 \frac{w_1}{w_2}$$

Т.к. АЦП используемого микроконтроллера имеет 12 разрядов, то его кодовый диапазон 0 – 4095. Напряжение на входе АЦП может изменяться в диапазоне -5 – 5 В.

Таким образом, можно определить дискретность измерения АЦП.

$$u_{2\text{дискр}} = \frac{u_{\text{изм.макс}}}{n_{\text{АЦП}}} = \frac{10}{4095} = 0{,}002442 \text{ В/ед}$$

С учетом трансформатора тока получаем:

$$i_{1\text{дискр}} = \frac{u_2}{R_{\text{изм}}} \cdot \frac{w_2}{w_1} = \frac{0{,}002442}{82} \cdot \frac{1000}{5} = 0{,}0059561 \text{ А}$$

Таким образом, единице кода АЦП соответствует 0,0059561 А в фазе двигателя.

Формула для преобразования кода АЦП в ток, т.к. нулевому току соответствует код АЦП 2048, то необходимо произвести смещение:

$$i_{\text{ф}} = (i_{\text{АЦП}} - 2048) \cdot i_{1\text{дискр}} = (i_{\text{АЦП}} - 2048) \cdot \frac{1}{R_{\text{изм}}} \cdot \frac{w_2}{w_1} \cdot \frac{u_{\text{изм.макс}}}{n_{\text{АЦП}}}$$

Листинг 1 – Код для преобразования кода АЦП в ток.

```
ia = (int) (drive.iA - 2048) * 0.0059561;
ib = (int) (drive.iB - 2048) * 0.0059561;
ic = -ia - ib;
```

1.2 Определение скорости двигателя.

Для измерения скорости двигателя используется тахогенератор с коэффициентом преобразование равным 2. Нулевой скорости соответствует код АЦП равный 2048. Тогда для получения скорости можно воспользоваться формулой:

$$n = (n_{\text{АЦП}} - 2048) \cdot k_{\text{тг}}$$

Листинг 2 – Код для преобразования кода АЦП в скорость АД.

```
speed = (int) (drive.adcSpeed - 2048) * ktg; // об/мин
```

3. Определение параметров асинхронного двигателя

3.1 Определение сопротивления якорной цепи.

Для определения сопротивления подадим постоянное напряжение на обмотки статора. Зададим следующие значения уставок ШИМ.

Cmpr1 = 200

Cmpr2 = 0

Cmpr3 =0

Определим потенциалы фаз асинхронного двигателя при заданных уставках.

$$\varphi_{\text{A}} = \frac{cmpr1}{tpr} U_{DC} = \frac{200}{6000} 540 = 18 \text{ В}$$

$$\varphi_B = \frac{cmpr1}{tpr} U_{DC} = 0 \text{ В}$$

$$\varphi_C = \frac{cmpr1}{tpr} U_{DC} = 0 \text{ В}$$

Определение потенциала средней точки производим по формуле:

$$\varphi_N = \frac{\varphi_\text{A} + \varphi_B + \varphi_C}{3} = \frac{18}{3} = 6 \text{ В}$$

Производим расчет потенциалов фаз:

$$U_a = \varphi_\text{A} - \varphi_N = 18 - 6 = 12 \text{ В}$$

$$U_b = \varphi_B - \varphi_N = 0 - 6 = -6\text{В}$$

$$U_c = \varphi_C - \varphi_N = 0 - 6 = -6 \text{ В}$$

Определяем установившееся значение тока по рисунку 3.

$$I_a = 4{,}1335 \text{ А}$$

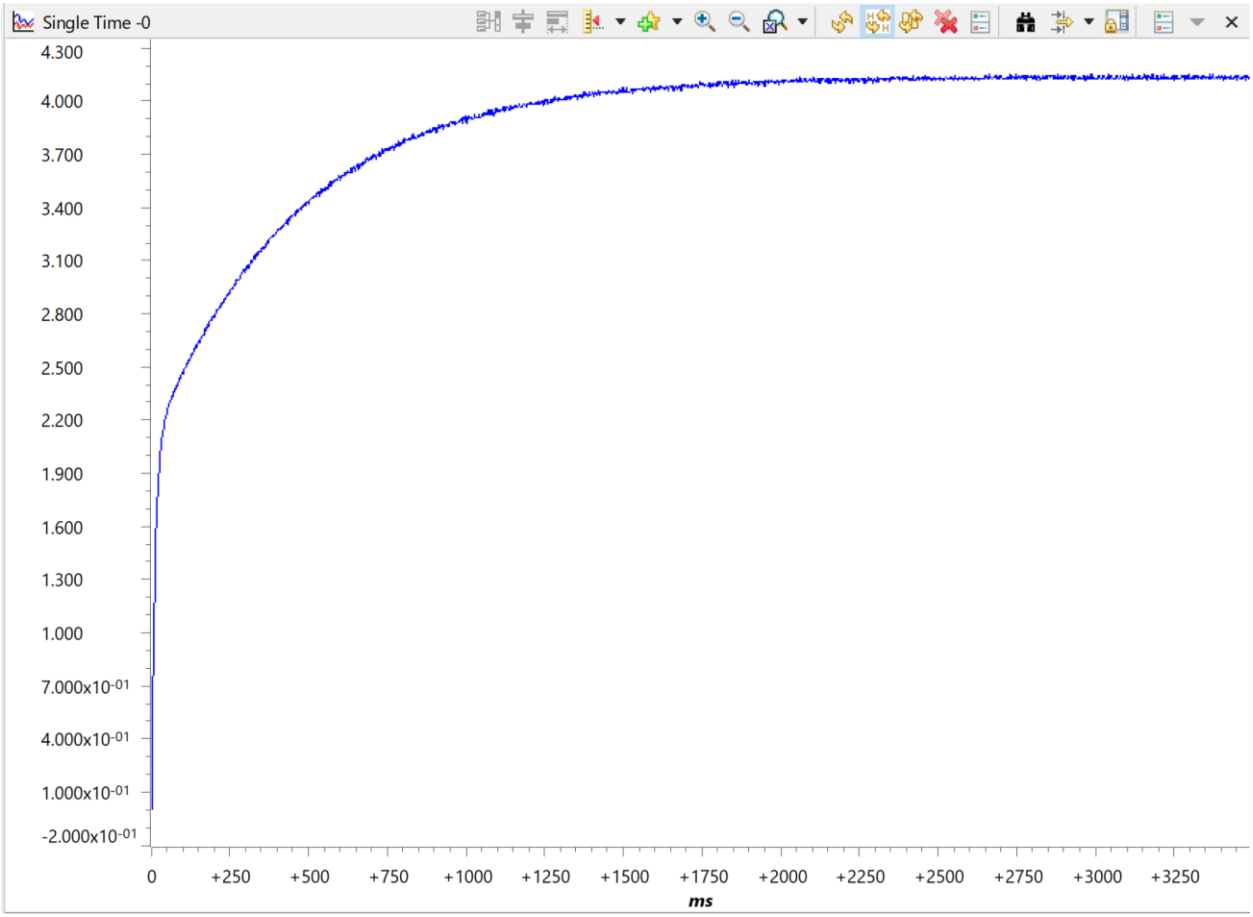

Рисунок 2 Осциллограмма тока в фазе А при подаче напряжения.

Определение сопротивления обмотки статора

$$R_s = \frac{U_a}{I_a} = \frac{12}{4{,}1335} = 2{,}9\ \text{Ом}$$

3.2 Определение индуктивностей двигателя

Определение индуктивности статора проводим по опыту холостого хода. Для подачи напряжения на статор двигателя воспользуемся синусоидальной ШИМ. В листинге 3 представлен код, позволяющий реализовать синусоидальную ШИМ, задавая Umax можно изменять амплитуду прикладываемого напряжения.

Листинг 3 – Синусоидальная ШИМ.

```
drive.cmpr1 = (unsigned int) (Umax * sin(teta) + 3000);
drive.cmpr2 = (unsigned int) (Umax * sin(teta - 2.0944) + 3000);
drive.cmpr3 = (unsigned int) (Umax * sin(teta + 2.0944) + 3000);
 teta += 2 * PI * 50 * 0.0002;
  if (teta >= 2 * PI)
        teta = 0;
```

Задаем Umax равным 1000, тогда амплитудное значение фазного напряжения будет равно:

$$U_{\text{А.ампл}} = \frac{cmpr1}{tpr} U_{DC} = \frac{1000}{6000} 540 = 90\text{В}$$

На рисунке 4 показаны параметры АД при холостом ходе. Определение тока холостого хода двигателя проводим по осциллограмме тока в фазе А (Рисунок 5). Максимальное значение тока, определенное по осциллограмме равно 0,2086 А.

Expression	Value	Type	Address
drive.cmpr1	2007	unsigned int	0x00009551@Data
drive.cmpr2	3387	unsigned int	0x00009552@Data
drive.cmpr3	3604	unsigned int	0x00009553@Data
drive.iA	2035	unsigned int	0x00009559@Data
drive.iB	2112	unsigned int	0x0000955A@Data
drive.fault	0	int	0x0000955C@Data
drive.adcSpeed	3548	unsigned int	0x00009556@Data
speed	3000	int	0x00009541@Data
ia	-0.07742929	float	0x0000954E@Data
ib	0.3811904	float	0x0000954C@Data
ic	-0.3037611	float	0x00009548@Data

Рисунок 3 Параметры асинхронного двигателя при холостом ходе.

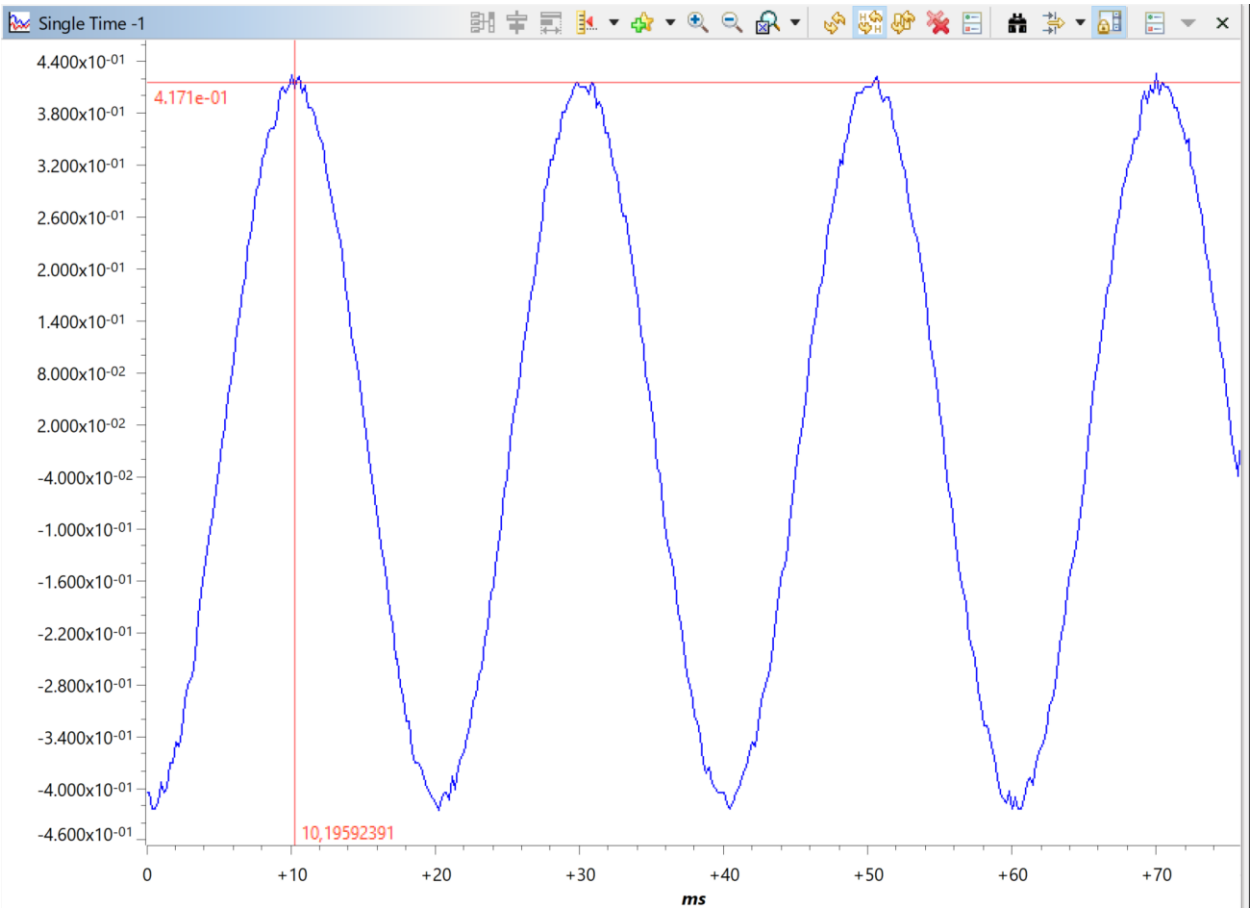

Рисунок 4 Ток фазы А при холостом ходе.

Определение полной индуктивности статора проводим по формуле:

$$L_s = \sqrt{(\frac{U_{\text{А.ампл}}}{I_{A.\text{ампл}}})^2 - R_s^2} \cdot \frac{1}{2\pi f_1} = \sqrt{\left(\frac{90}{0{,}417}\right)^2 - 2{,}9^2} \cdot \frac{1}{2\pi \cdot 50} = 0{,}687 \text{ Гн}$$

Для определения сопротивлений рассеяния статора и ротора определим критический момент двигателя. Для его определения будем увеличивать момент на валу двигателя пока скорость не начнет быстро падать до нуля, т.е. произойдет опрокидывание двигателя. На рисунке 6 показана осциллограмма изменения нагрузки на валу двигателя, при моменте равном 0,8 происходит быстрое падение скорости двигателя, что говорит о моменте на валу двигателя выше критического, тогда принимаем критический момент двигателя равным:

$$M_{кр} = \frac{0,75 + 0,8}{2} = 0,775 \text{ Н} \cdot \text{м}$$

Рисунок 5 Изменение момента на валу двигателя.

Определение сопротивления индуктивностей рассеяния проводим по формуле:

$$U_{ампл} = 90 \text{ В}$$

$$U = \frac{U_{ампл}}{\sqrt{2}}$$

$$X_1 + X_2' = \frac{3U^2}{2\omega_0 M_k} = \frac{3 \cdot (\frac{90}{\sqrt{2}})^2}{2 \cdot 314,16 \cdot 0,775} = 24,95 \text{ Ом}$$

Принимаем индуктивности рассеяния статора и ротора равными, поэтому делим сумму их сопротивлений пополам и на частоту питающего напряжения.

$$L_{s\sigma} = L_{r\sigma} = \frac{X_1 + X_2'}{2 \cdot 2\pi \cdot f} = \frac{24{,}95}{4 \cdot \pi \cdot 50} = 0{,}0397 \text{ Гн}$$

4. Разработка скалярной системы управления с S-образным задатчиком интенсивности в программе SimInTech.

S – Образный задатчик интенсивности

Рассмотрим принцип работы S – образного задатчика интенсивности, показанного на рисунке 6.

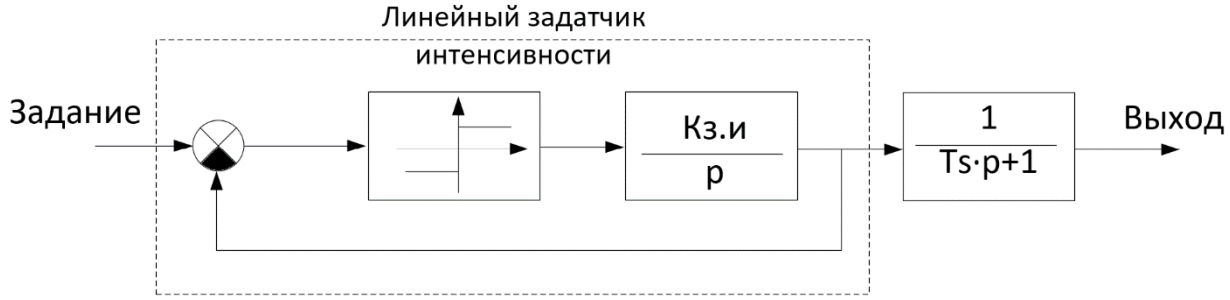

Рисунок 6 Структурная схема S-образного задатчика интенсивности

На входе задатчика интенсивности задается требуемая скорость асинхронного двигателя, которая приходит на сумматор с отрицательной обратной связью из интегрального звена. После сумматора находится реле, которое при наличии входного сигнала больше нуля выдает единицу на интегральное звено, наличие реле необходимо для обеспечения линейного нарастания выхода интегрального звена,$k_{\text{з.и.}}$определяет темп нарастания выхода линейного задатчика интенсивности. Для обеспечения S – образного задания скорости после интегратора расположено инерциальное звено 1-го порядка, в котором Ts отвечает за начальное и конечное поведение задатчика интенсивности.

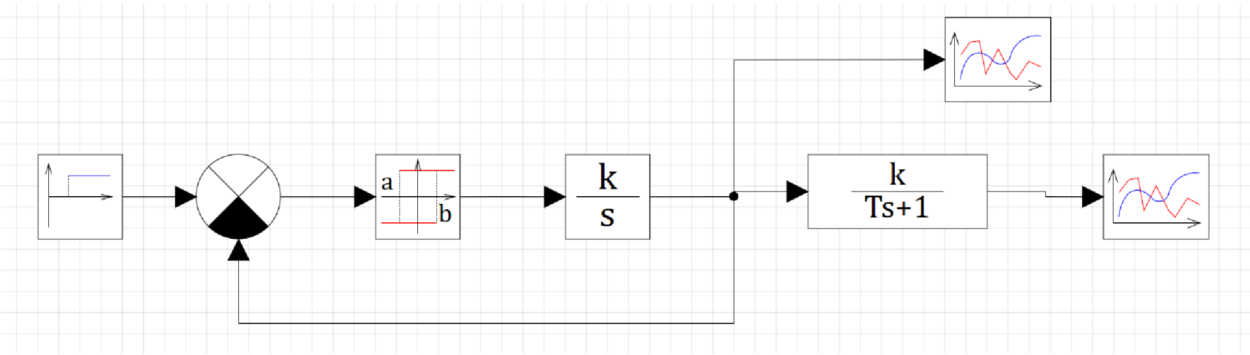

Рисунок 7 S - образный задатчик интенсивности в SimInTech

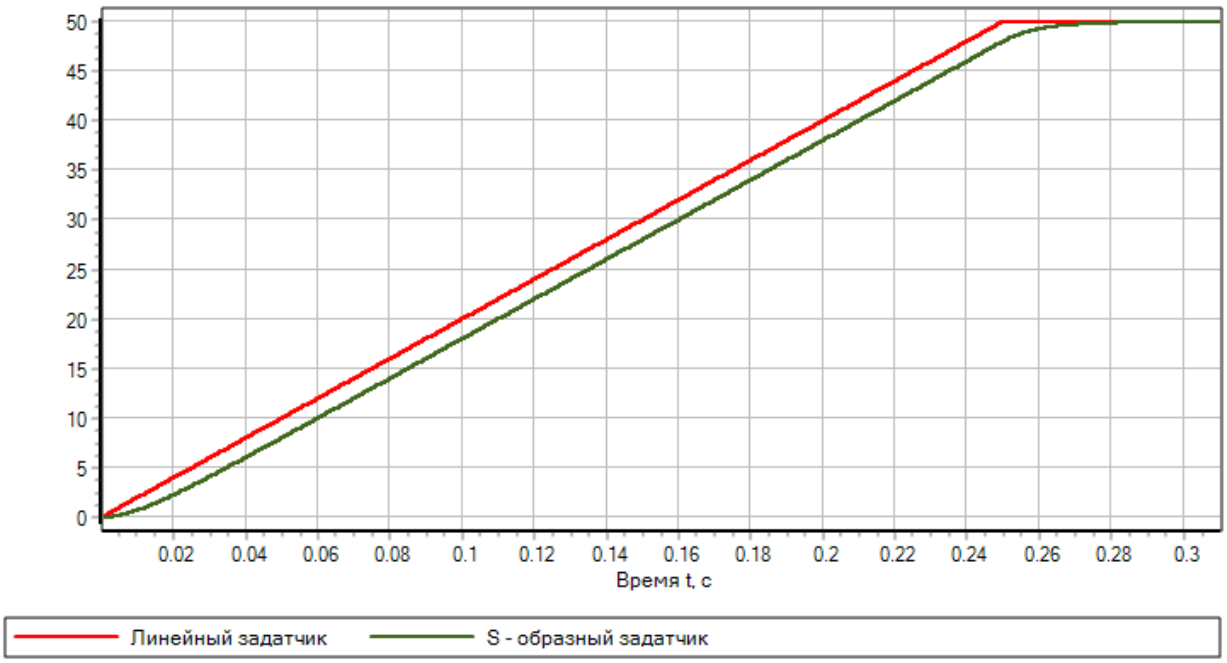

Рисунок 8 Выход линейного задатчика интенсивности и S – образного задатчика

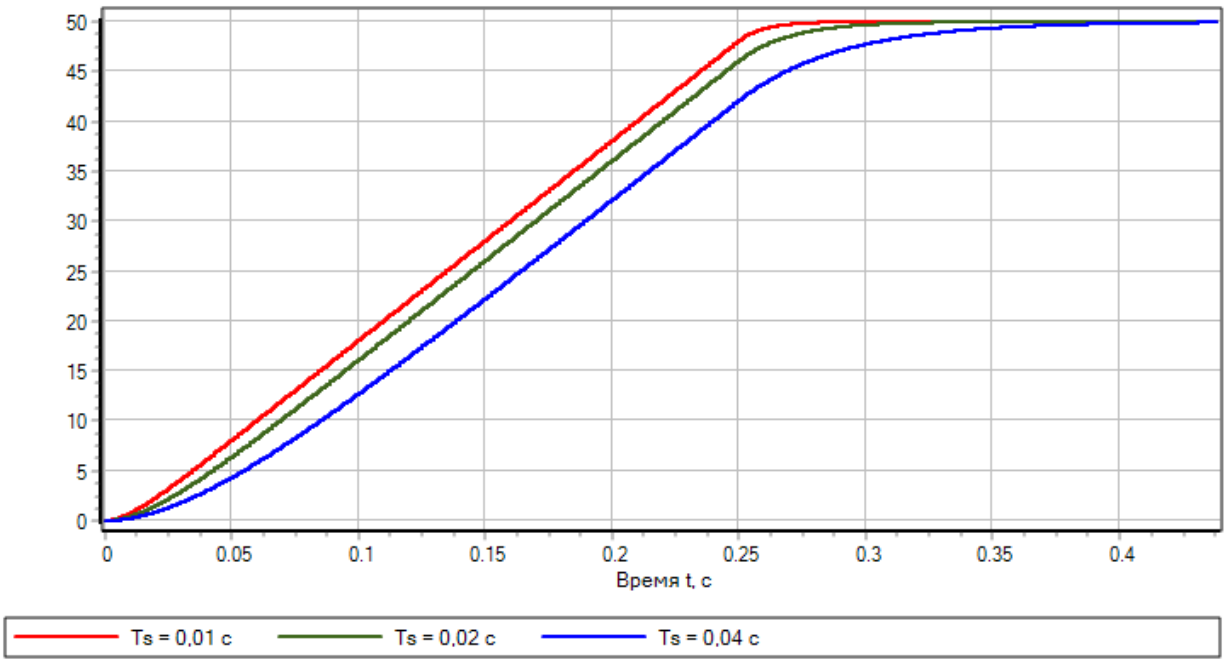

Рисунок 9 Выход S – образного задатчика интенсивности при различных Ts

Разработка модели двигателя и преобразователя частоты в SimInTech

Для симуляции асинхронного двигателя используется блок АД, на который подаются напряжения с блока инвертора. Напряжение звена постоянного тока задаем 540 В. Инвертором управляет блок ШИМ, частоту ШИМ задаем 5 кГц. Блок ФСМ производит преобразование задаваемых синусоидальных фазных напряжений A,B,C в форму напряжений для использования максимального напряжения звена постоянного тока.

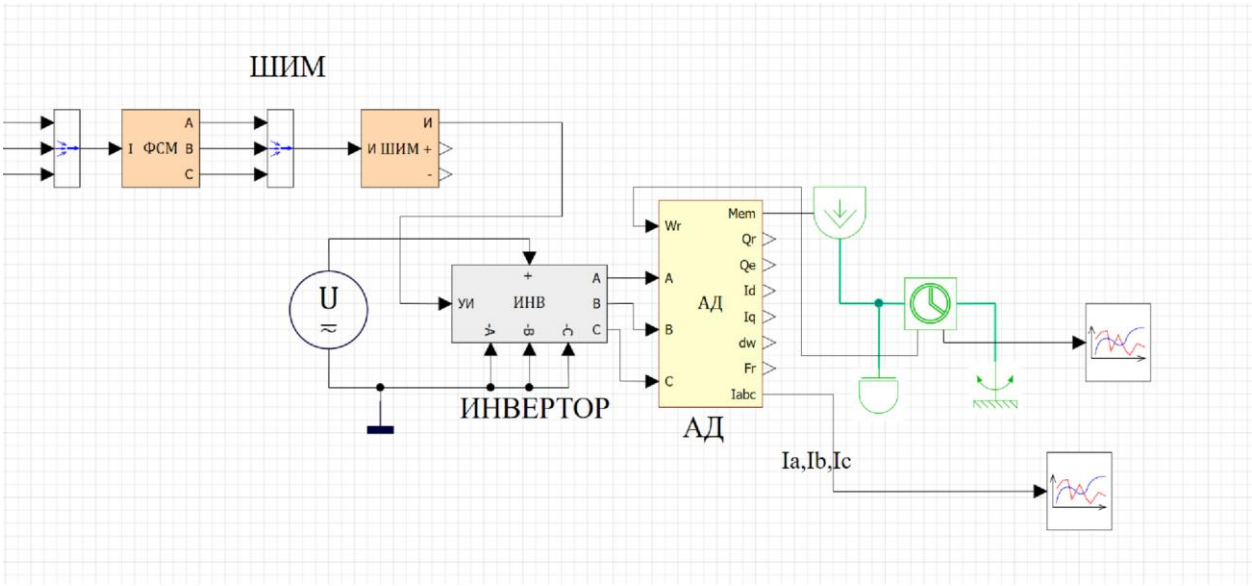

Рисунок 10 Модель АД и преобразователя частоты

Расчет зависимости U(f) для асинхронного двигателя и ее реализация.

Для нахождения зависимости напряжения двигателя от частоты воспользуемся формулами:

$$M_{\text{к.ном}} = \frac{3 \cdot U_{\text{ном}}^2}{2 \cdot \omega_{0\text{ном}} \cdot (R_1 + \sqrt{R_1^2 + (X_1 + X_2)^2}}$$

$$U(f) = \sqrt{\frac{M_{\text{к.ном}} \cdot 2 \cdot \omega_{0\text{ном}} \cdot \frac{f}{f_{\text{ном}}} \cdot (R_1 + \sqrt{R_1^2 + \left((X_1 + X_2)\frac{f}{f_{\text{ном}}}\right)^2})}{3}}$$

Данные асинхронного двигателя:

$$X_1 + X_2' = 24{,}95 \text{ Ом}$$

$$R_s = 2{,}9 \text{ Ом}$$

$$\omega_0 = 314{,}16 \text{ рад/с}$$

Таблица 1 –частотный закон с IR компенсацией.

U1	14,21	34,09	75,25	137,15	220
f1	0	5	15	30	50

U1 – действующее фазное напряжение

Для изменения задания напряжения в зависимости от задаваемой частоты используем блок ломаной статической характеристики, схема показана на рисунке 11.

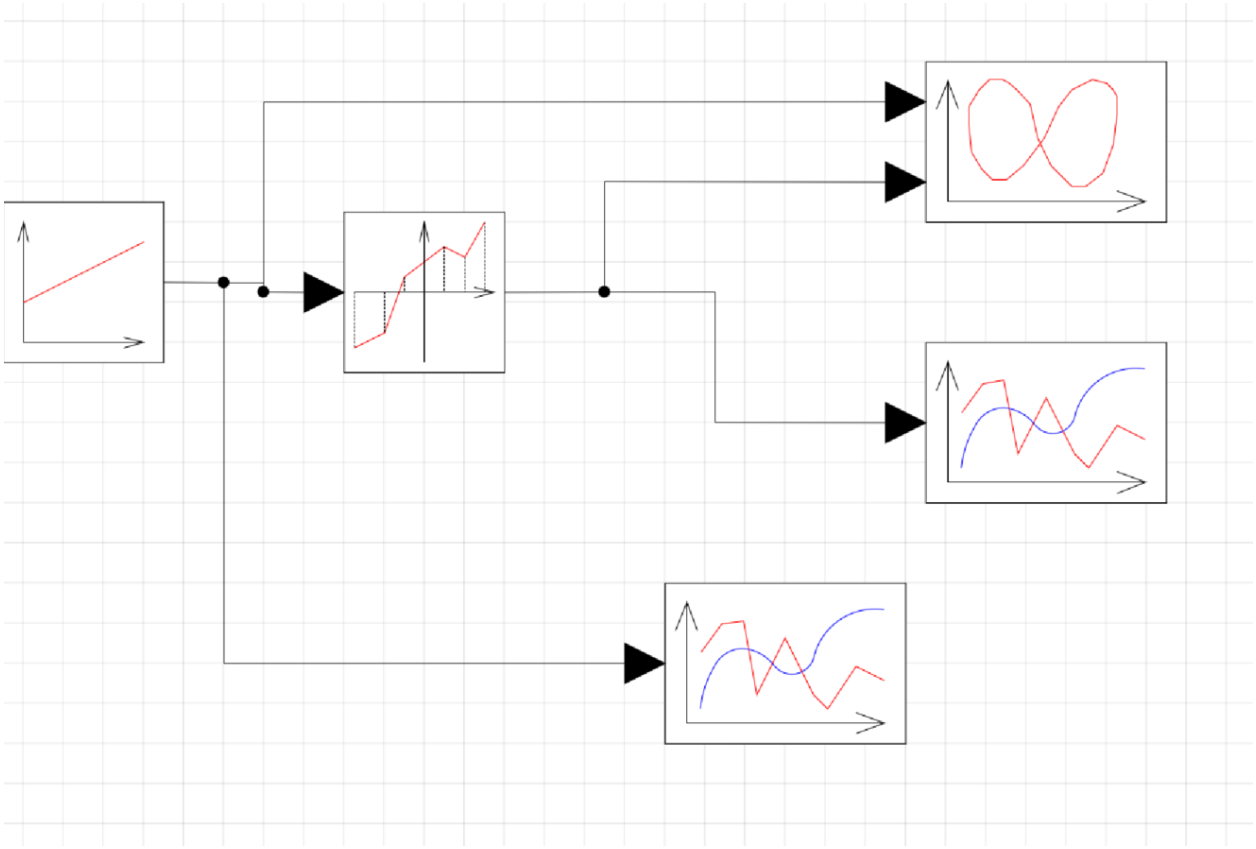

Рисунок 11 Блок задания напряжения двигателя от частоты

На рисунке 12 показана полученная зависимость напряжения от частоты при использовании блока ломаной статической характеристики.

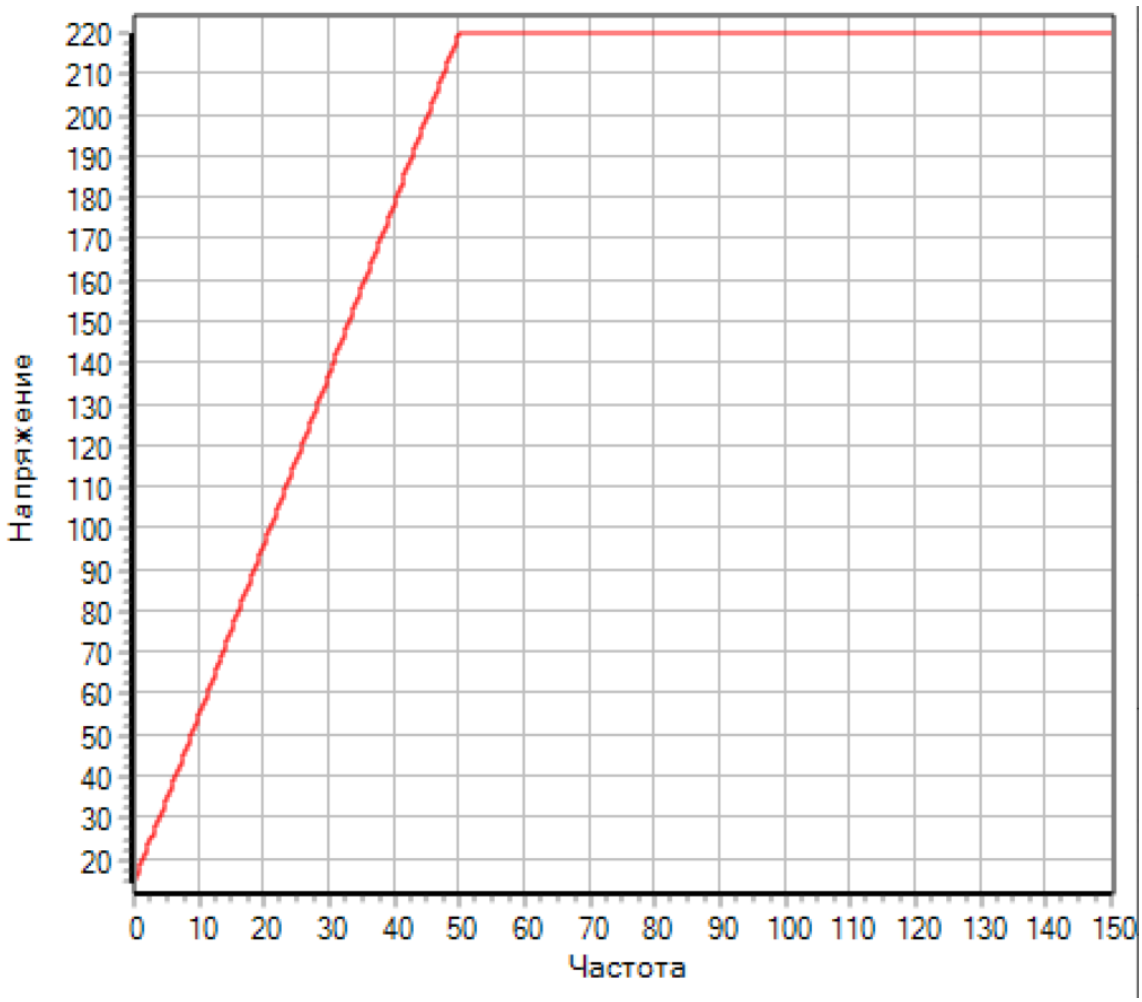

Рисунок 12 Зависимость напряжения от частоты.

Разработка скалярной системы управления

На рисунке 13 показана разработанная скалярная система управления, она состоит из задатчика интенсивности, выход которого подается на задание частоты синусоидального напряжения и на блок зависимости напряжения от частоты, который задает амплитуду фазного напряжения.

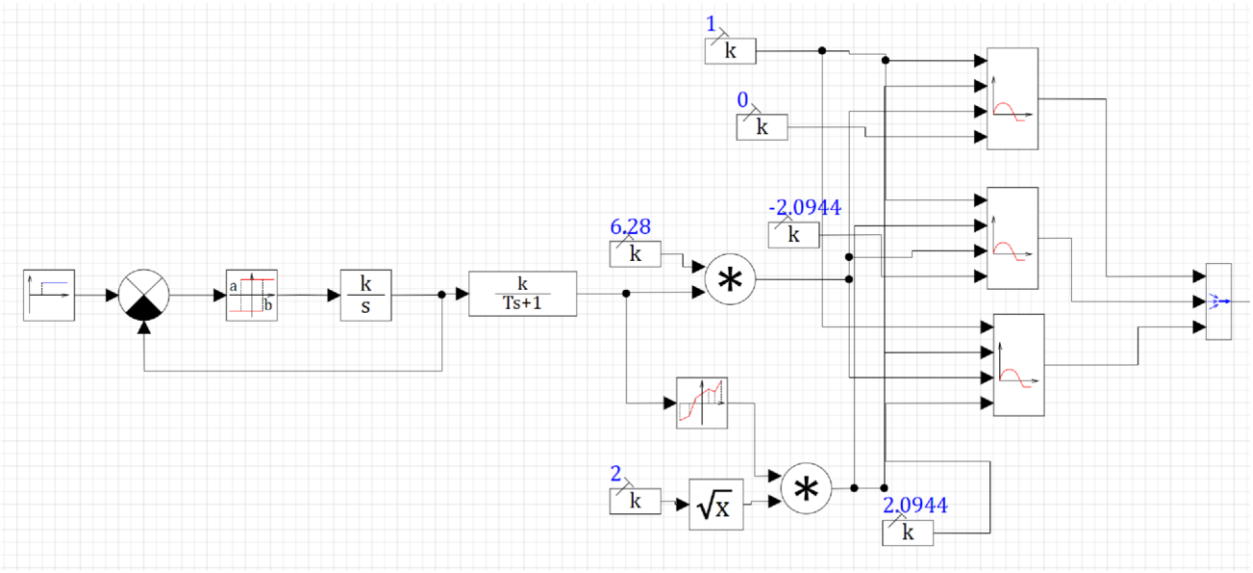

Рисунок 13 Схема скалярной системы управления, реализованная в SimInTech

На рисунке 14 показаны выходные напряжения системы управления, поступающие на ШИМ. На рисунке 16 показан установившийся режим, можно убедиться что частота задаваемого напряжения равна 50 Гц, синусоидальное напряжение формируется правильно.

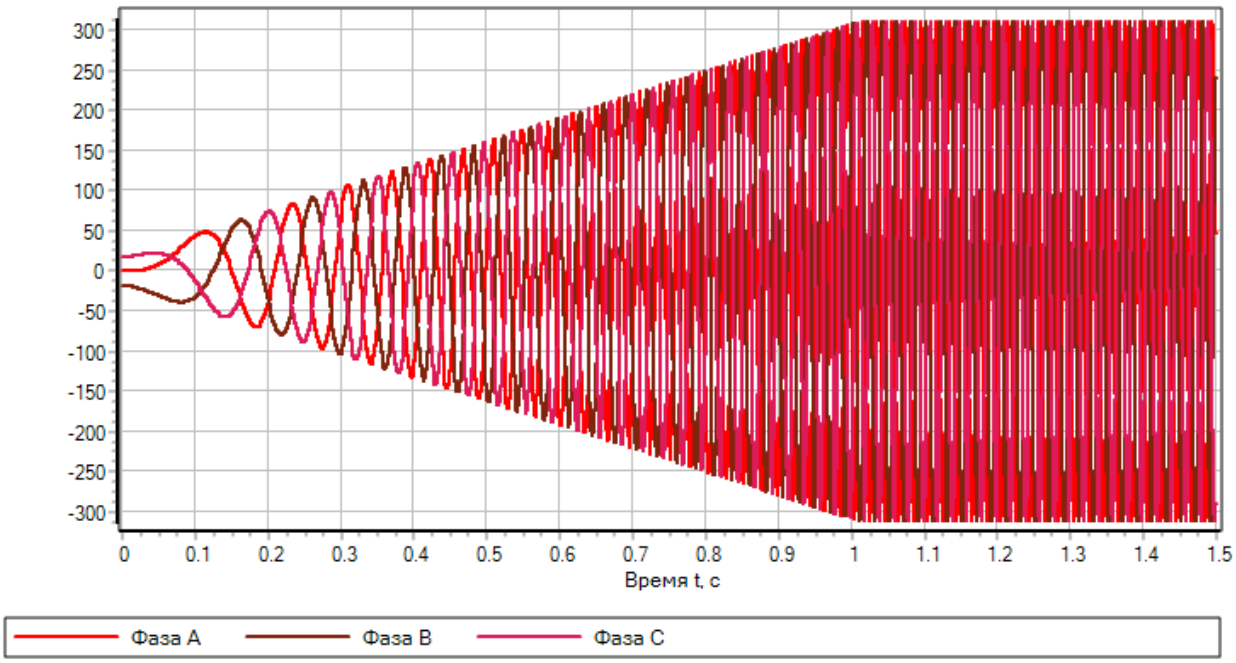

Рисунок 14 Задание напряжений на выходе системы управления.

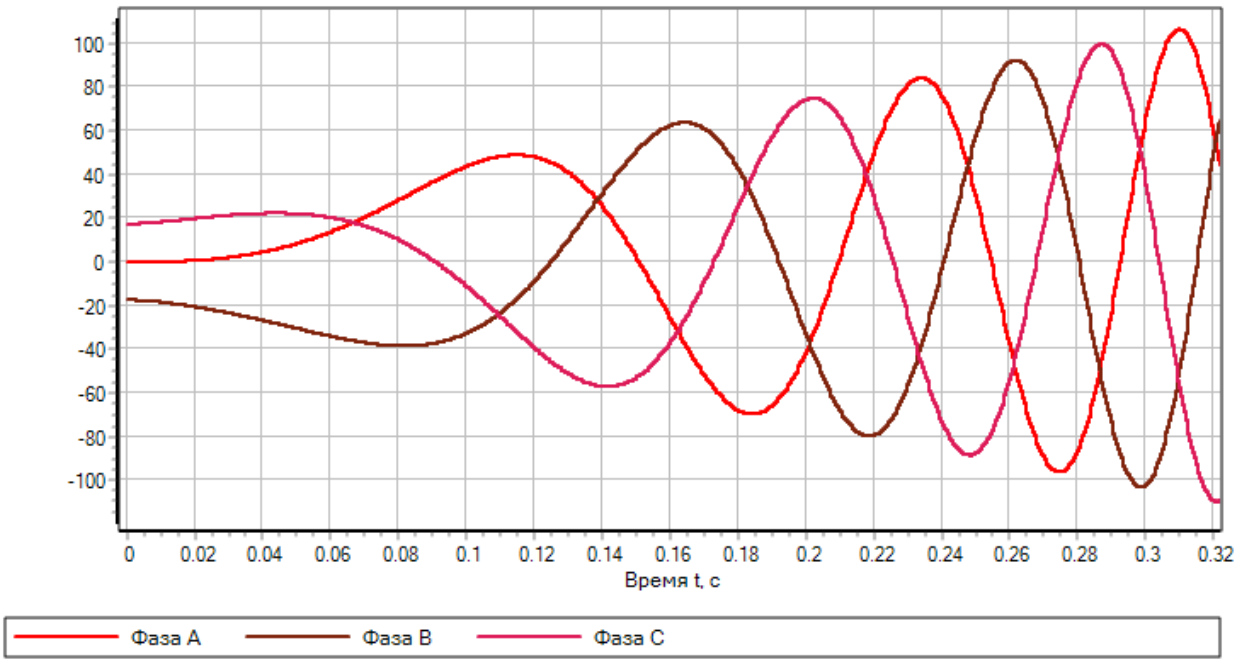

Рисунок 15 Начальный участок рисунка 14

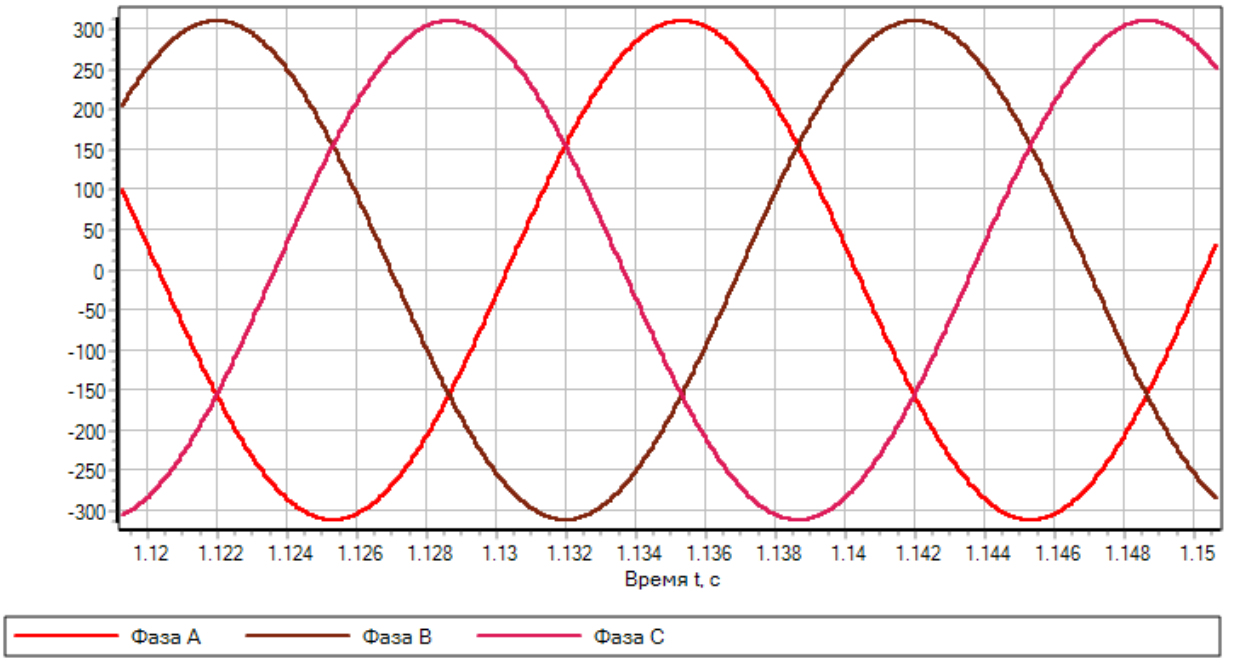

Рисунок 16 Установившийся режим рисунка 14

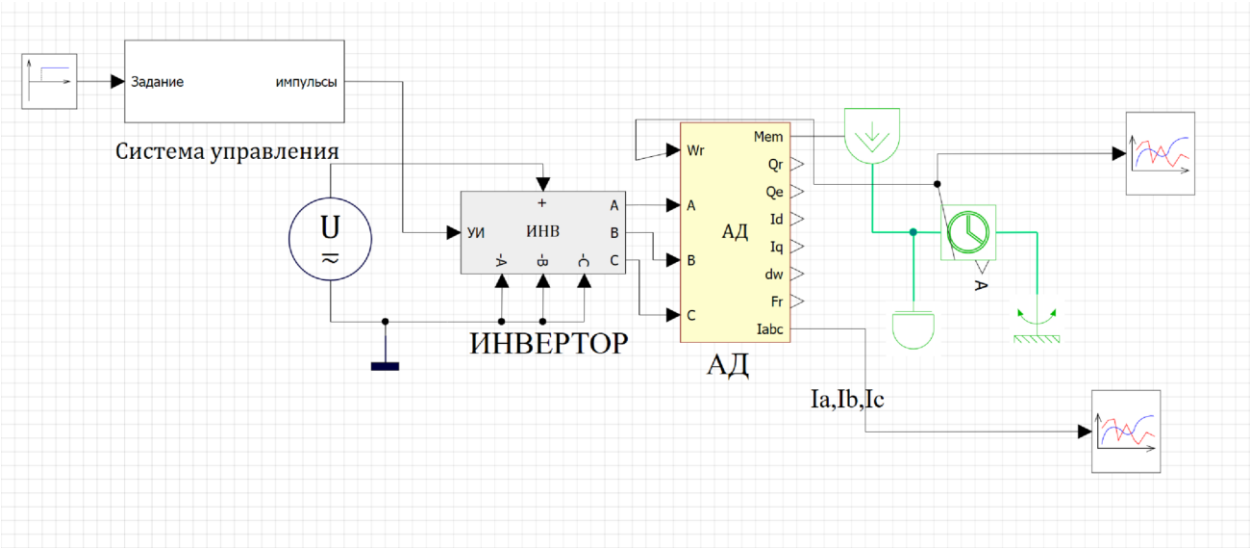

Рисунок 17 Система управления, подключенная к силовой части электропривода.

На рисунке 15 показан пуск двигателя за 1 секунду с использованием S – образного задатчика интенсивности. Колебания скорости двигателя связаны с наличием индуктивности в двигателе, что приводит к обмену энергией между индуктивным накопителем и механическим, соответственно появляются колебания. Данные колебания можно уменьшить, увеличив момент инерции вала двигателя. На рисунке 16 показаны токи двигателя при пуске, можно увидеть что плавный пуск позволяет уменьшить пусковой ток.

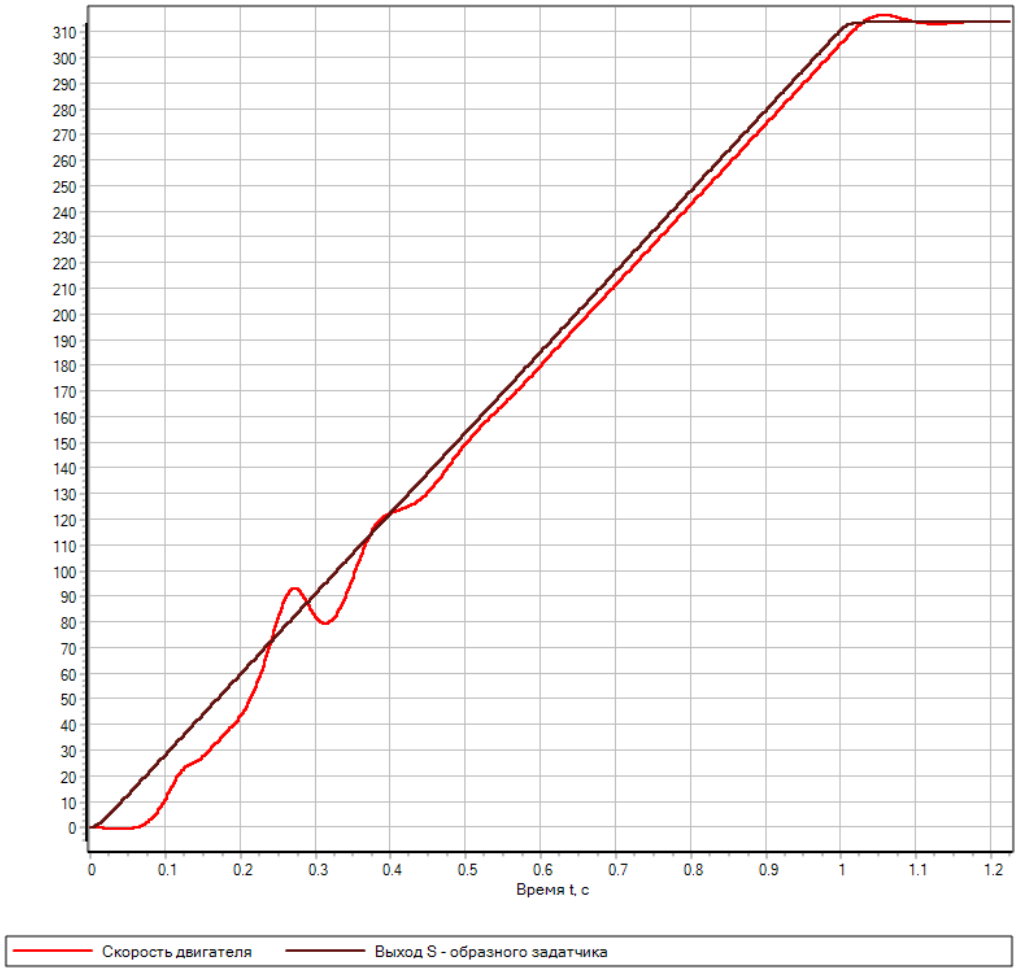

Рисунок 18 Пуск за 1 секунду

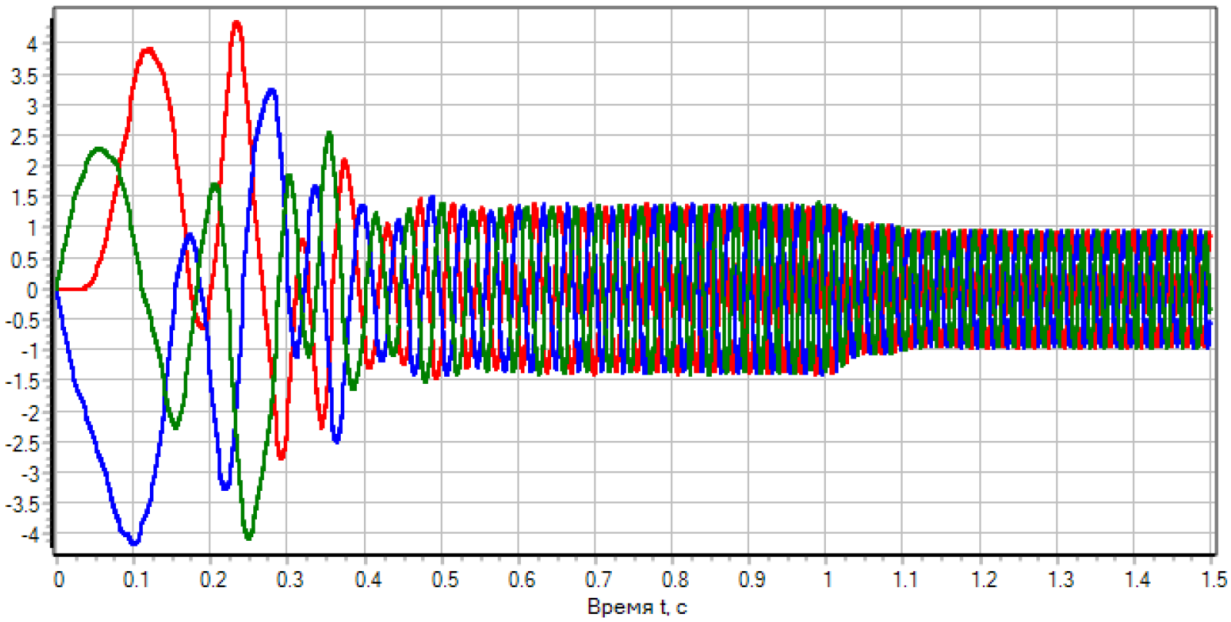

Рисунок 19 Токи двигателя при пуске за 1 секунду

5. Разработка скалярной системы управления с S-образным задатчиком в программе Code Composer Studio на языке программирования С.

Реализация S – образного задатчика интенсивности и зависимости U(f)

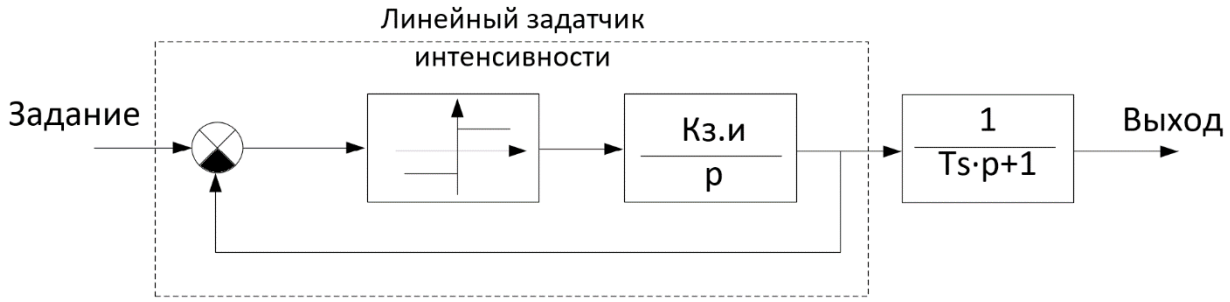

Рисунок 20 Структурная схема S – образного задатчика интенсивности

Для реализации задатчика, показанного на рисунке 20, необходимо получить разностные уравнения для интегрального и инерциального звеньев.

Разностное уравнение для интегрального звена

$$\frac{Y(p)}{X(p)} = \frac{k_{з.и}}{p}$$

$$yp = k_{з.и} \cdot x$$

$$yp = \frac{y_k - y_{k-1}}{T_{шим}}$$

$$\frac{y_k - y_{k-1}}{T_{шим}} = k_{з.и} \cdot x$$

$$y_k = y_{k-1} + T_{шим} \cdot k_{з.и} \cdot x$$

Разностное уравнение для инерциального звена

$$\frac{Y(p)}{X(p)} = \frac{1}{T_s p + 1}$$

$$T_s p y + y = x$$

$$T_s \frac{y_k - y_{k-1}}{T_{\text{шим}}} + y_k = x_k$$

$$\left(1 + \frac{T_s}{T_{\text{шим}}}\right) y_k = \left(\frac{T_s}{T_{\text{шим}}}\right) y_{k-1} + x_k$$

$$y_k = \frac{\frac{T_s}{T_{\text{шим}}}}{1 + \frac{T_s}{T_{\text{шим}}}} y_{k-1} + \frac{1}{1 + \frac{T_s}{T_{\text{шим}}}} x_k$$

$$k_1 = \frac{\frac{T_s}{T_{\text{шим}}}}{1 + \frac{T_s}{T_{\text{шим}}}}$$

$$k_2 = \frac{1}{1 + \frac{T_s}{T_{\text{шим}}}}$$

$$y_k = k_1 \cdot y_{k-1} + k_2 \cdot x_k$$

В листинге 4 показано объявление структуры с переменными, которые необходимы для реализации задатчика интенсивности и объявление прототипов требуемых функций. В структуре используются следующие переменные:

Input – заданное значение частоты на входе задатчика

Output – выходное задание частоты задатчика интенсивности

k – Интегральный коэффициент интегрального звена

Ts – постоянная времени инерционного звена

Int_out – значение выхода интегрального звена

Max – максимальное значение выхода задатчика

Min – минимальное значение выхода задатчика

tpwm – период ШИМ

k1 – коэффициент для разностного уравнения инерционного звена

k2 – коэффициент для разностного уравнения инерционного звена

Листинг 4 – Создание структуры и объявление прототипов функций.

```
#ifndef RAMP_GEN_H_
#define RAMP_GEN_H_

  struct Sramp {
        float input;
        float output;
        float k;
        float Ts;
        float int_out;
        float max;
        float min;
        float tpwm;
        float k1;
        float k2;


        void (*Init)(struct Sramp *);
        void (*Calc)(struct Sramp *);
 };

#define Sramp_Default {0, 0, 50, 0.2, 0, 50, -50, 0.0002, 0, 0,
Sramp_init,Sramp_Calc}


  void Sramp_init (struct Sramp  *);
  void Sramp_Calc (struct Sramp *);
  float voltfreq (float freq);

#endif /* RAMP_GEN_H_ */
```

В листинге 5 показана реализация S – образного задатчика интенсивности с использованием полученных разностных уравнений. Также была реализована зависимость U (f), соответствующая таблице 1. Зависимость получена линейным соединением заданных точек.

Листинг 5 – Функции инициализации и расчета задатчика интенсивности, функция расчета вольт-частотного закона.

```
#include "ramp_gen.h"
#include "math.h"

  void Sramp_init (struct Sramp  *ramp) { // расчет коэффициентов
        float h;
        h = ramp->tpwm;
        ramp->k1 =( ramp -> Ts) /(ramp->Ts +h ) ;
```

```
       ramp->k2 =( h) /(ramp->Ts +h ) ;
 }

  void Sramp_Calc (struct Sramp *ramp) { // расчет задатчика
        float delta;
        if (((ramp->input >0) && (ramp->output < ramp->max ) )  || ( (ramp->input
<0) && (ramp->output> ramp->min) ) ){
               delta = ramp -> input - ramp -> int_out;

      if (delta >0) ramp-> int_out = ramp -> int_out +( ramp->tpwm * ramp->k);
      if (delta<0) ramp-> int_out = ramp -> int_out -( ramp->tpwm * ramp->k);

               ramp-> output = ramp->output * ramp->k1 + ramp->k2 * ramp->int_out;
        }
        if ((ramp->input >0) && (ramp->output > ramp->max ) ) ramp->output = ramp-
>max;
        if ( (ramp->input <0) && (ramp->output < ramp->min) ) ramp->output = ramp
->min;
  }

  float voltfreq (float freq){ // зависимость напряжения от частоты
        float freqabs;
        float U;
        freqabs = abs(freq) ;
         if (freqabs <= 5.0){
               U = 14.21 + 19.88 * 0.2 * freqabs;
         } else if (freqabs <= 15.0 ){
               U = 34.09 + 41.16 * 0.1 * (freqabs - 5.0);
         } else if (freqabs <= 30.0){
               U = 75.25 + 61.9 * 0.066667 * (freqabs - 15);
         } else if (freqabs <= 50.0 ){
               U = 137.15 + 82.85 * 0.05 * (freqabs - 30);
         } else U = 220;

        return U;
  }
```

Проверка работы S – образного задатчика интенсивности.

Проверим работу задатчика интенивности при разных постоянных времени.

Expression	Value
(x)= input	50.0
(x)= output	50.0
(x)= k	50.0
(x)= Ts	0.0
(x)= int_out	50.00761
(x)= max	50.0
(x)= min	-50.0
(x)= tpwm	0.0002
(x)= k1	0.0
(x)= k2	1.0

Рисунок 21 Заданные параметры задатчика интенсивности

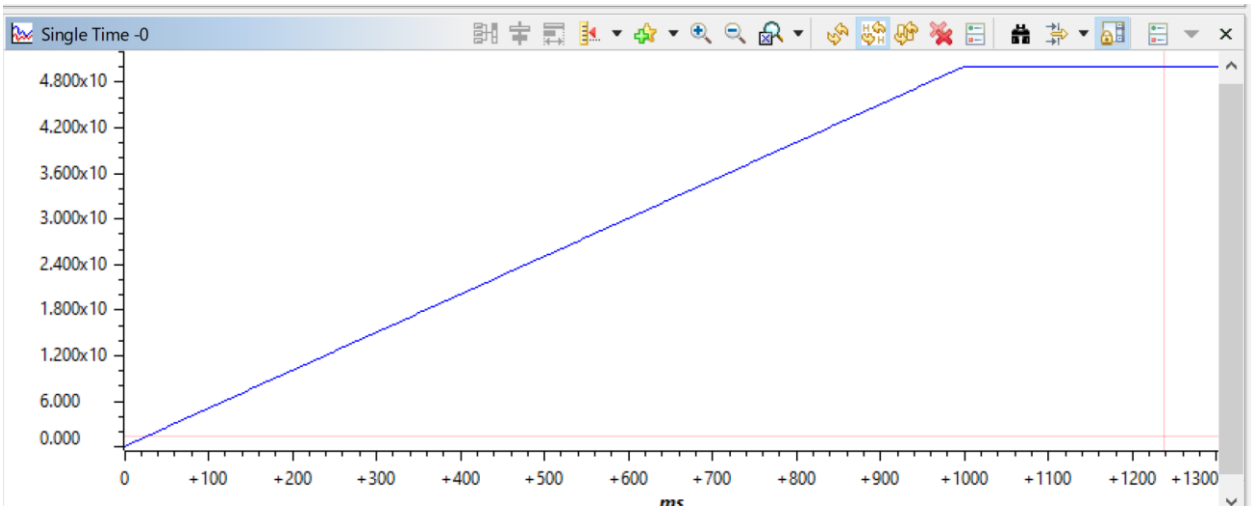

Рисунок 22 Выход задатчика интенсивности при Кз.и = 50 и Ts = 0.

ramp	{...}
input	50.0
output	50.0
k	50.0
Ts	0.1
int_out	50.00761
max	50.0
min	-50.0
tpwm	0.0002
k1	0.998004
k2	0.001996008

Рисунок 23 Заданные параметры задатчика интенсивности

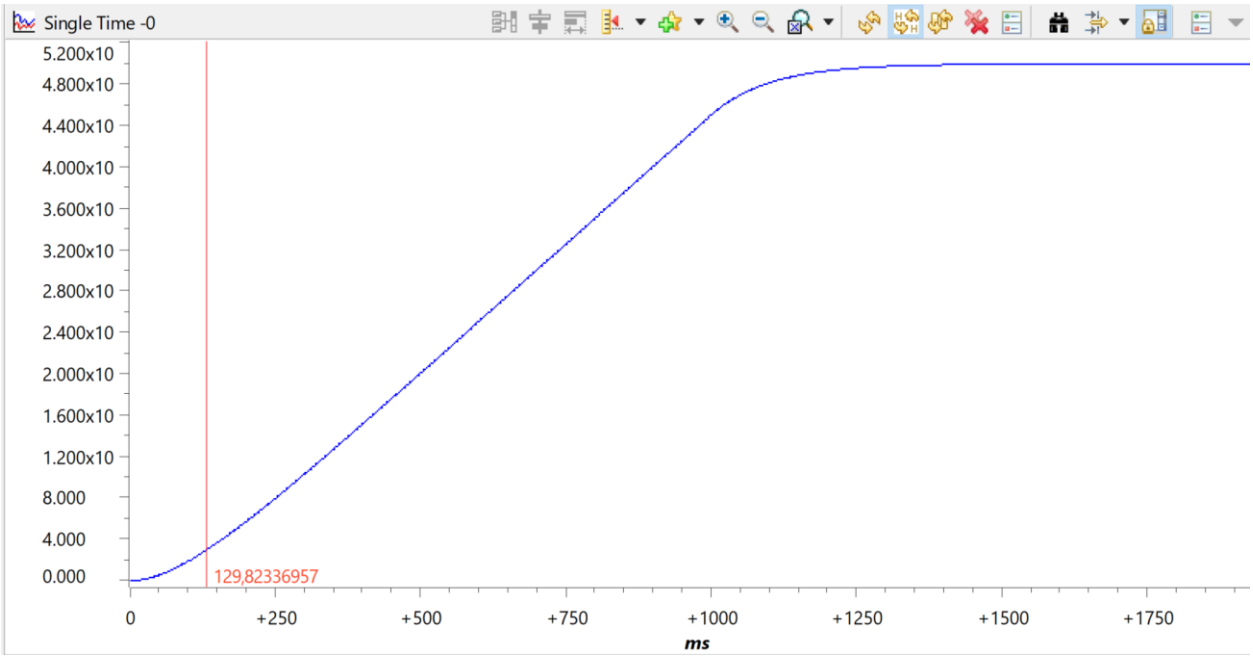

Рисунок 24 Выход задатчика интенсивности при Кз.и = 50 и Ts = 0,1

freq	0.116544
ramp	{...}
input	50.0
output	6.168311
k	50.0
Ts	0.2
int_out	13.60022
max	50.0
min	-50.0
tpwm	0.0002
k1	0.999001
k2	0.000999001

Рисунок 25 Заданные параметры задатчика интенсивности

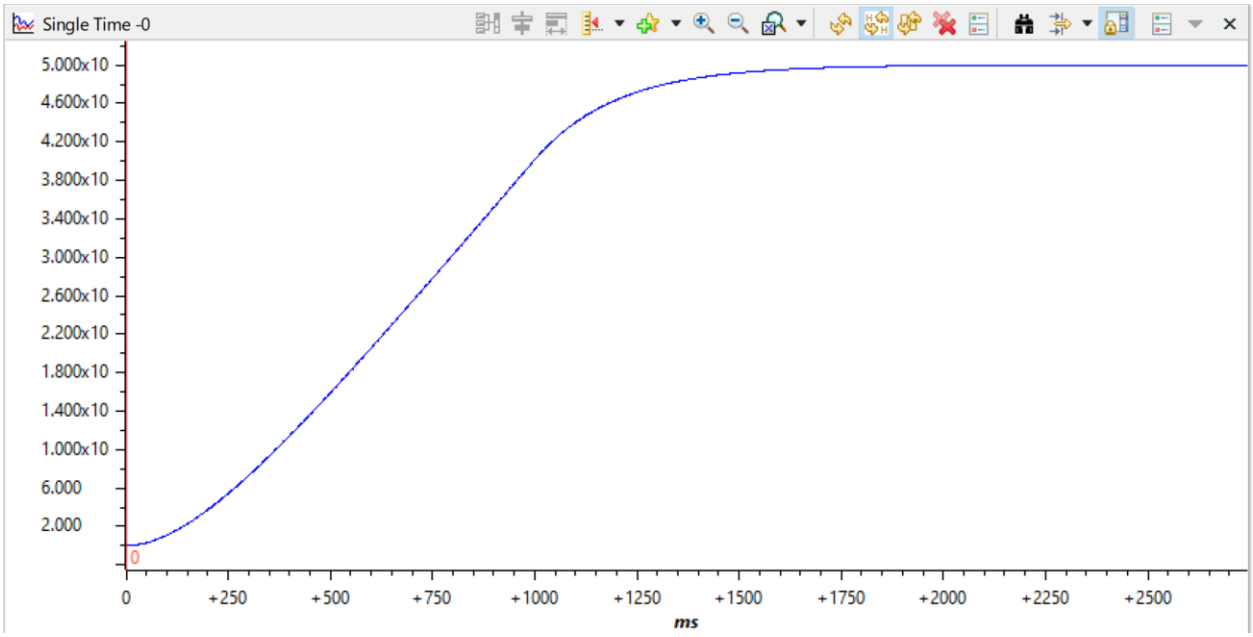

Рисунок 26 Выход задатчика интенсивности при Кз.и = 50 и Ts = 0,2

Выходы задатчика интенсивности при различных постоянных времени(Рисунок 22, Рисунок 24, Рисунок 26) совпадают с теоретическими графиками из SimInTech (Рисунок 8, Рисунок 9).

Реализация векторной ШИМ

Реализацию векторной ШИМ осуществляем с привязкой к нулевой шине. Реализованная векторная ШИМ показана в листинге 6.

Листинг 6 – Реализация векторной ШИМ в коде

```
U = voltfreq (freq) * 2.44949  *0.001851852* 0.866; // расчет длины вектора в относительных единицах

      if (teta < 1.0472){
            ym1 = 1.1547 * U * sin(teta);
            ym = U * cos(teta) - 0.57735 * U * sin(teta);
            sum = ym + ym1;
                  if (sum > 1){
                        ym = ym / (sum);
                        ym1 = ym1 / (sum);
                  }
            ya = ym + ym1;
            yb = ym1;
            yc = 0;

      } else if (teta < 2*1.0472){
            ym1 = 1.1547 * U * sin(teta - 1.0472);
          ym = U * cos(teta - 1.0472) - 0.57735 * U * sin(teta - 1.0472);
       sum = ym + ym1;
             if (sum > 1){
                        ym = ym / (sum);
                        ym1 = ym1 / (sum);
                  }
                        ya = ym ;
                        yb = ym + ym1 ;
                        yc = 0;
      } else if (teta < 3*1.0472){
                    ym1 = 1.1547 * U * sin(teta - 2*1.0472);
                        ym = U * cos(teta - 2*1.0472) - 0.57735 * U * sin(teta - 2*1.0472);
                     sum = ym + ym1;
                           if (sum > 1){
                                    ym = ym / (sum);
                                    ym1 = ym1 / (sum);
                              }
                                    ya = 0;
                                    yb =  ym +ym1 ;
                                    yc = ym1;
      } else if (teta < 4*1.0472){
            ym1 = 1.1547 * U * sin(teta - 3*1.0472);
                ym = U * cos(teta - 3*1.0472) - 0.57735 * U * sin(teta - 3*1.0472);
               sum = ym + ym1;
                        if (sum > 1){
                                    ym = ym / (sum);
                                    ym1 = ym1 / (sum);
                              }
                               ya = 0;
```

```
                                yb =  ym ;
                                yc = ym + ym1;
} else  if (teta < 5*1.0472){
    ym1 = 1.1547 * U * sin(teta -( 4*1.0472));
          ym = U * cos(teta - (4*1.0472)) - 0.57735 * U * sin(teta - (4*1.0472));
     sum = ym + ym1;
               if (sum > 1){
                          ym = ym / (sum);
                          ym1 = ym1 / (sum);
                    }
                      ya = ym1;
                          yb =  0 ;
                          yc = ym + ym1;
} else if (teta <= 6*1.0472){
    ym1 = 1.1547 * U * sin(teta - 5*1.0472);
          ym = U * cos(teta - 5*1.0472) - 0.57735 * U * sin(teta - 5*1.0472);
     sum = ym + ym1;
               if (sum > 1){
                          ym = ym / (sum);
                          ym1 = ym1 / (sum);
                    }
                      ya = ym1 +ym;
                          yb =  0 ;
                          yc = ym ;
}

           drive.cmpr1 =(unsigned int) (ya * 6000);
             drive.cmpr2 =(unsigned int) (yb * 6000);
             drive.cmpr3 = (unsigned int) (yc * 6000);

      teta += 2 * PI * freq* 0.0002;
      if (teta >= 2 * PI)
            teta = 0;
```

Для проверки правильности реализации векторной ШИМ снимем осциллограммы задаваемых скважностей и относительных линейных напряжений.

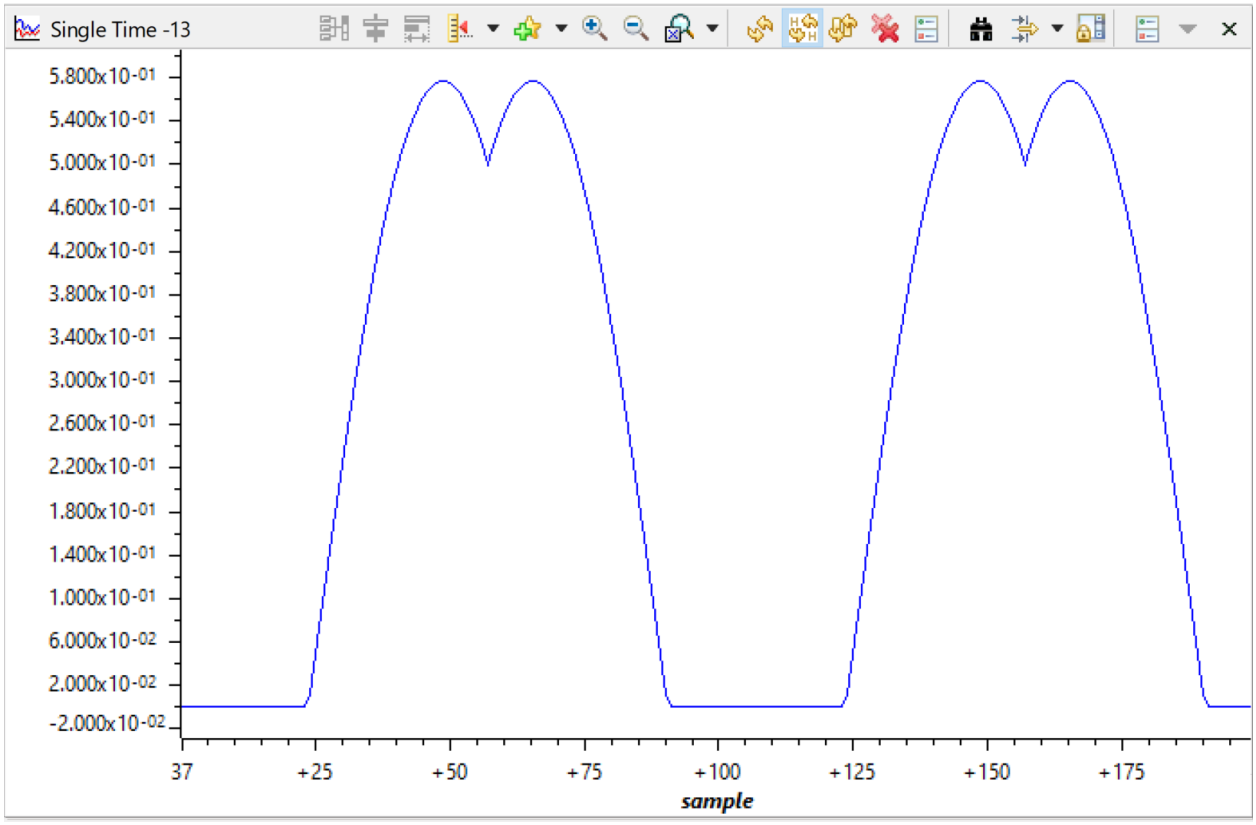

Рисунок 27 Скважности фазы А

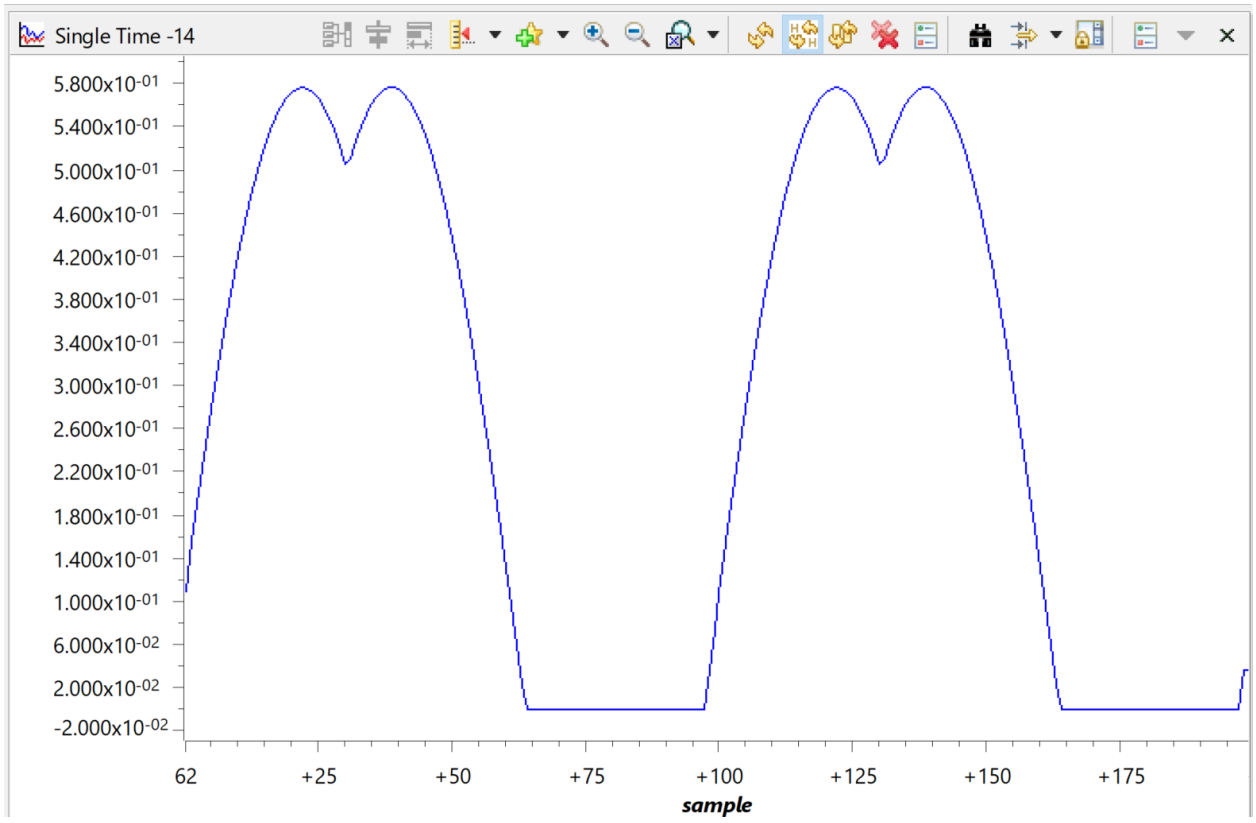

Рисунок 28 Скважности фазы В

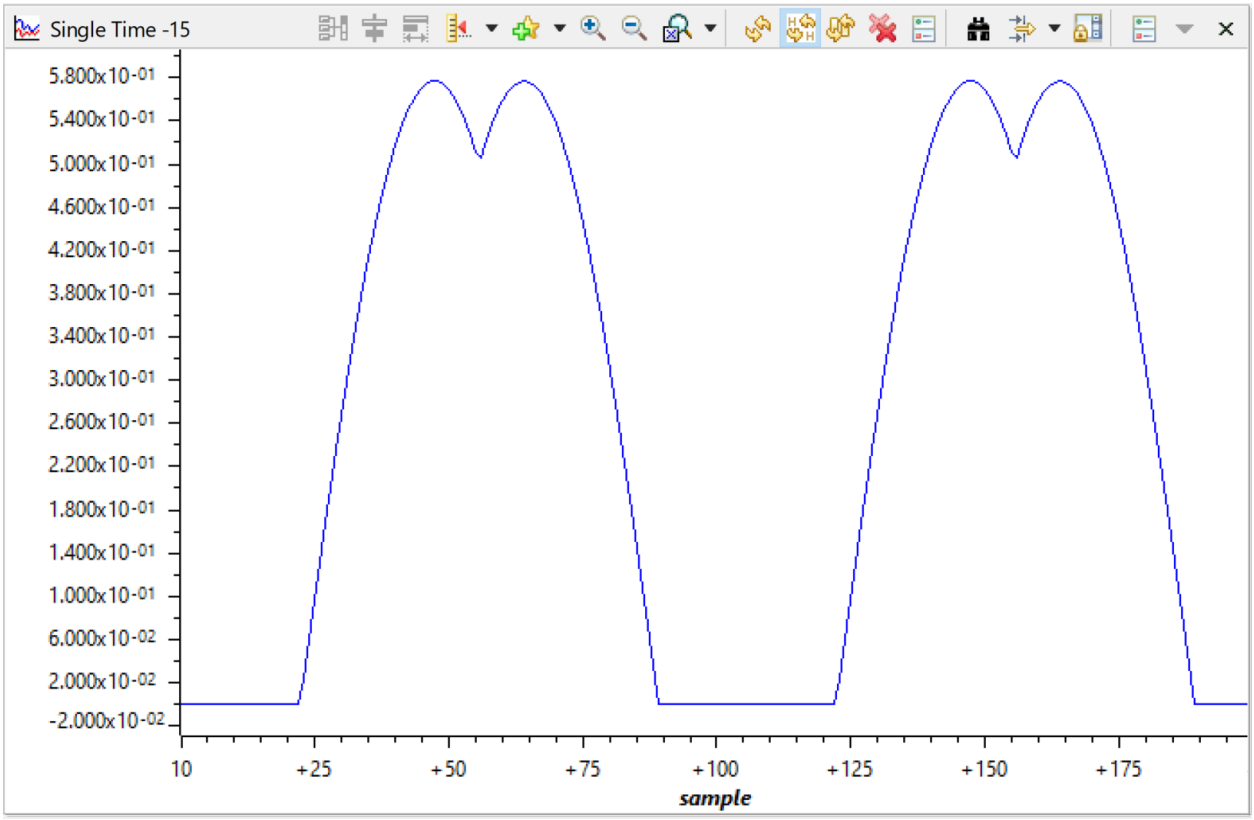

Рисунок 29 Скважности фазы С

Также убедимся в правильности линейных напряжений, линейные напряжения получаем вычитанием фазных скважностей.

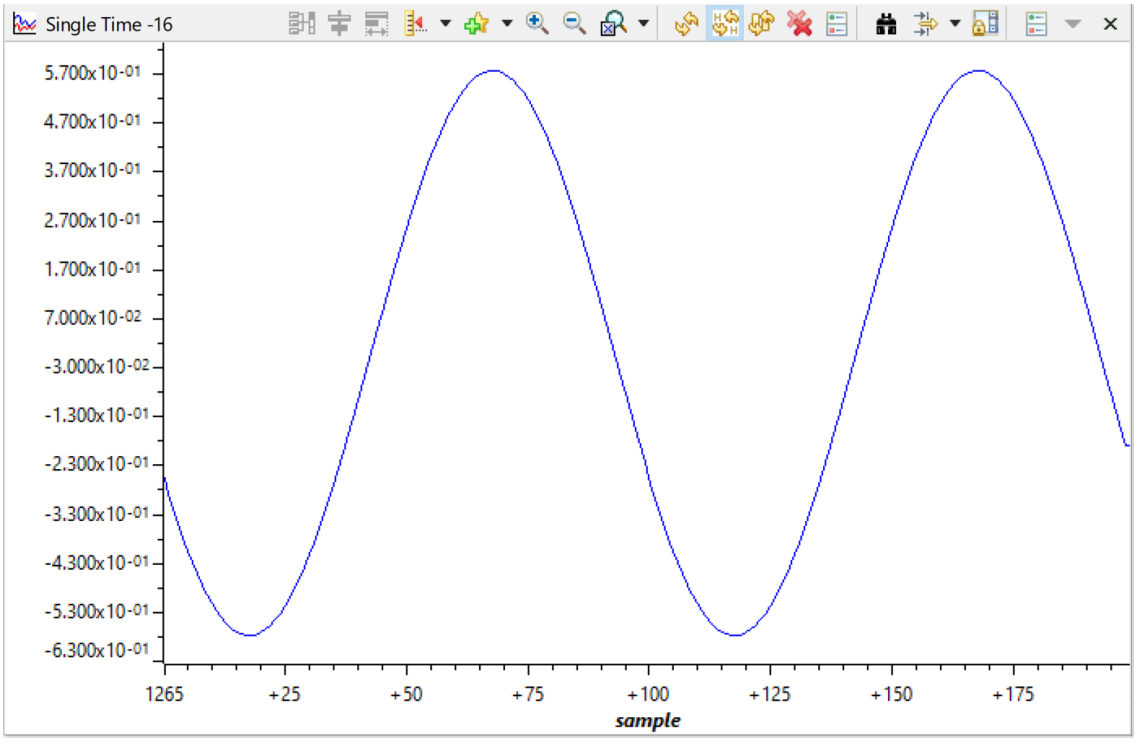

Рисунок 30 Относительное напряжение Uab

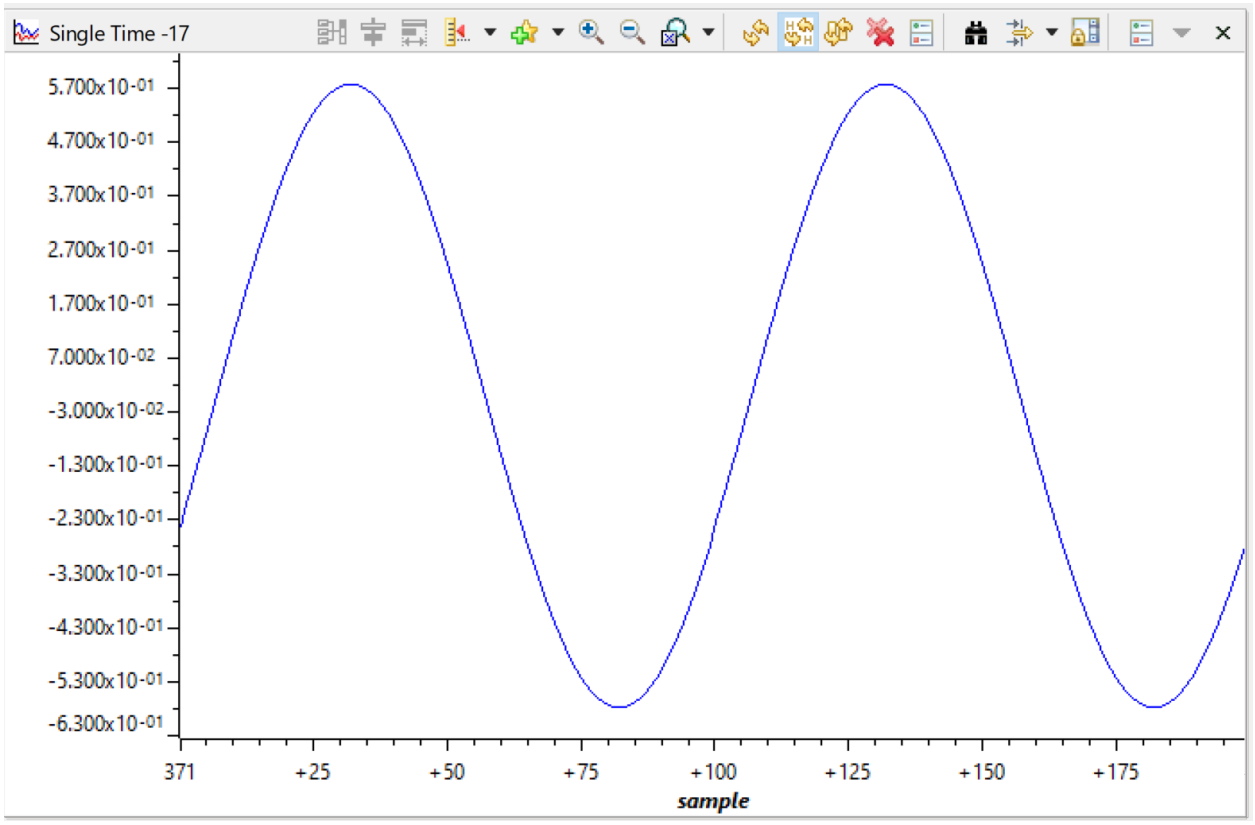

Рисунок 31 Относительное напряжение Ubc

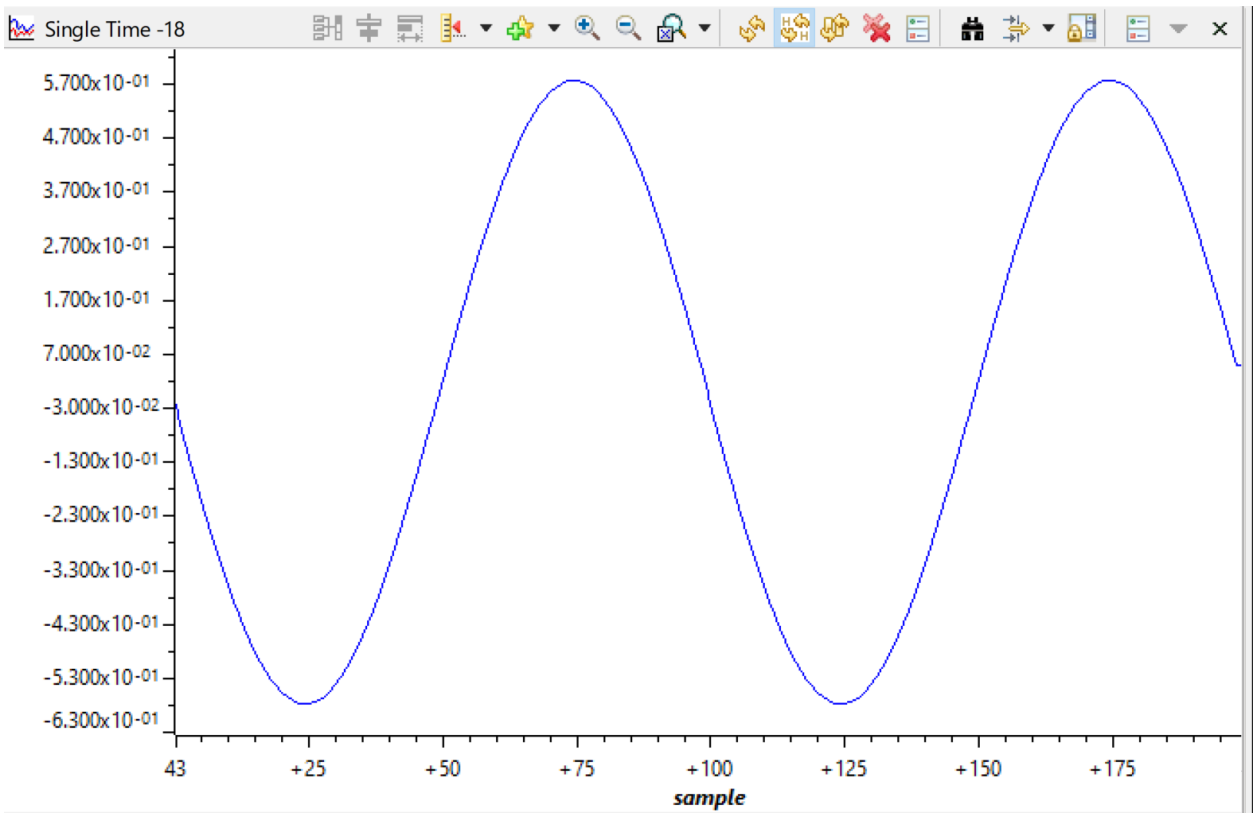

Рисунок 32 Относительное напряжение Uca

Полученные осциллограммы соответствуют теоретическим, значит векторная ШИМ разработана верно.

Пуск двигателя с разработанным S – образным задатчиком интенсивности.

Пуск двигателя показан на рисунке 33, осциллограммы пуска соответствуют теоретическим, значит система управления была разработана верно.

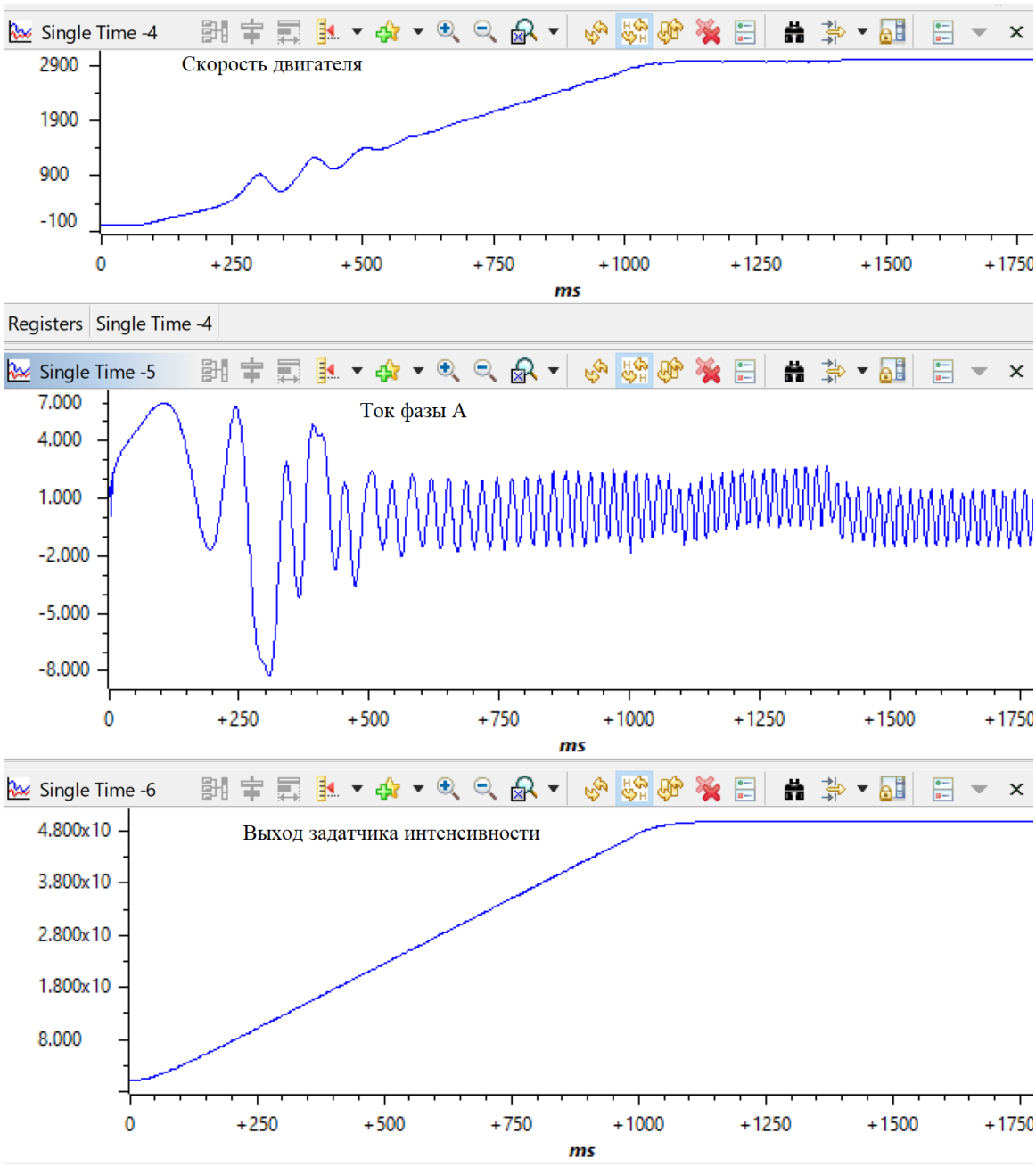

Рисунок 33 Плавный пуск двигателя на частоту 50 Гц

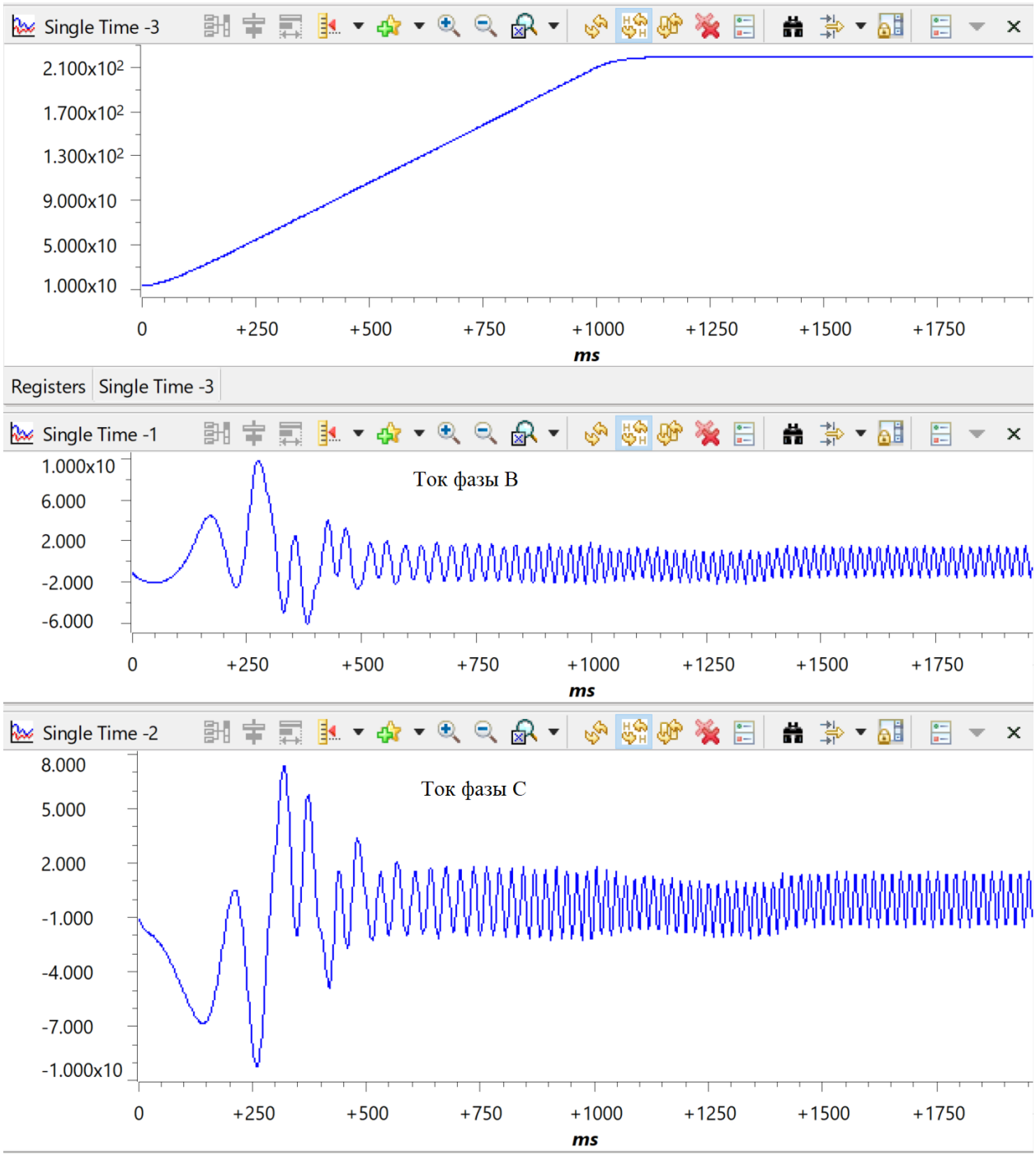

Рисунок 34 Плавный пуск двигателя на частоту 50 Гц

В листинге 7 приведен код главной программы, в которой происходит инициализация, объявление функции системы управления, вызов прерывания с функциями, обеспечивающими работу скалярной системы управления.

Листинг 7 –Главная программа

```
/* Основной файл работы с моделью двигателя.
 * В функции "main()" инициализируется периферия МК F28035 - настраивается тактирование,
 * инициализируется flash-память. Модуль ePWM1 настраивается на работу с частотой 5 кГц
 * при счёте "вверх-вниз", настраивается прерывание по достижении счётчиком нуля.
 *
 * Затем происходит инициализация модели двигателя, после чего разрешаются прерывания
 * и начинает выполняться бесконечный цикл.
 *
 * В функции-обработчике прерывания вызывается пустая функция "controlSystem()", в которой
 * пользователь может реализовать свою систему управления. Затем вызывается функция
 * расчёта модели двигателя.
 */

// Подключение необходимых заголовочных файлов
#include "DSP28x_Project.h"
#include "model.h"
#include "math.h"
#include "ramp_gen.h"
// Определение структуры модели двигателя
MODELDATA drive = MODEL_DEFAULTS;

float ia = 0;
float ib = 0;
float ic = 0;
float sin1 = 0;
float teta = 0;
int speed = 0;
int ktg = 2;
int Umax = 1000;
int skipBreakpoint = 10;
int skipCount = 0;

float freq = 0;
float U = 0;
float ym1 = 0;
float ym = 0;
float ya = 0;
float yb = 0;
float yc = 0;
float sum = 0;
float Uab = 0;
float Ubc = 0;
float Uca = 0;


void parameters() {
	ia = (int) (drive.iA - 2048) * 0.0059561;
	ib = (int) (drive.iB - 2048) * 0.0059561;
	ic = -ia - ib;
	speed = (int) (drive.adcSpeed - 2048) * ktg; // об/мин
}

void controlSystem() {
```

```
      U = voltfreq (freq) * 2.44949  *0.001851852* 0.866;

      if (teta < 1.0472){
            ym1 = 1.1547 * U * sin(teta);
            ym = U * cos(teta) - 0.57735 * U * sin(teta);
            sum = ym + ym1;
                  if (sum > 1){
                        ym = ym / (sum);
                        ym1 = ym1 / (sum);
                  }
            ya = ym + ym1;
            yb = ym1;
            yc = 0;

      } else if (teta < 2*1.0472){
            ym1 = 1.1547 * U * sin(teta - 1.0472);
          ym = U * cos(teta - 1.0472) - 0.57735 * U * sin(teta - 1.0472);
       sum = ym + ym1;
               if (sum > 1){
                        ym = ym / (sum);
                        ym1 = ym1 / (sum);
                  }
                    ya = ym ;
                        yb = ym + ym1 ;
                        yc = 0;
      } else if (teta < 3*1.0472){
                  ym1 = 1.1547 * U * sin(teta - 2*1.0472);
                      ym = U * cos(teta - 2*1.0472) - 0.57735 * U * sin(teta -
2*1.0472);
                   sum = ym + ym1;
                            if (sum > 1){
                                    ym = ym / (sum);
                                    ym1 = ym1 / (sum);
                                }
                                  ya = 0;
                                    yb =  ym +ym1 ;
                                    yc = ym1;
      } else if (teta < 4*1.0472){
            ym1 = 1.1547 * U * sin(teta - 3*1.0472);
                ym = U * cos(teta - 3*1.0472) - 0.57735 * U * sin(teta -
3*1.0472);
              sum = ym + ym1;
                      if (sum > 1){
                                    ym = ym / (sum);
                                    ym1 = ym1 / (sum);
                        }
                          ya = 0;
                                    yb =  ym ;
                                    yc = ym + ym1;
} else  if (teta < 5*1.0472){
    ym1 = 1.1547 * U * sin(teta -( 4*1.0472));
          ym = U * cos(teta - (4*1.0472)) - 0.57735 * U * sin(teta - (4*1.0472));
     sum = ym + ym1;
               if (sum > 1){
                        ym = ym / (sum);
                        ym1 = ym1 / (sum);
                  }
                    ya = ym1;
                        yb =  0 ;
                        yc = ym + ym1;
```

```
} else if (teta <= 6*1.0472){
    ym1 = 1.1547 * U * sin(teta - 5*1.0472);
          ym = U * cos(teta - 5*1.0472) - 0.57735 * U * sin(teta - 5*1.0472);
     sum = ym + ym1;
                if (sum > 1){
                          ym = ym / (sum);
                          ym1 = ym1 / (sum);
                    }
                      ya = ym1 +ym;
                          yb =  0 ;
                          yc = ym ;
}

           drive.cmpr1 =(unsigned int) (ya * 6000);
             drive.cmpr2 =(unsigned int) (yb * 6000);
             drive.cmpr3 = (unsigned int) (yc * 6000);

       teta += 2 * PI * freq* 0.0002;
       if (teta >= 2 * PI)
             teta = 0;



}

// Прототип функции-обработчика прерывания
interrupt void controlIsr(void);

struct Sramp ramp = Sramp_Default;

// Основная функция "main()", в которой происходит инициализация системы
int main(void) {
       // Настройка периферии микроконтроллера
       InitSysCtrl();
       memcpy(&RamfuncsRunStart, &RamfuncsLoadStart,
                    &RamfuncsLoadEnd - &RamfuncsLoadStart);
       InitFlash();
       InitPieCtrl();
       InitPieVectTable();

       // Настройка таймера для прерываний на частоте 5 кГц
       EALLOW;
       EPwm1Regs.TBPRD = 6000;
       EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
       EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
       EPwm1Regs.ETSEL.bit.INTEN = 1;
       EPwm1Regs.ETSEL.bit.INTSEL = TB_CTR_ZERO;
       EPwm1Regs.ETPS.bit.INTPRD = 1;

       PieVectTable.EPWM1_INT = controlIsr;
       PieCtrlRegs.PIEIER3.all = M_INT1;
       IER |= M_INT3;
       EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN;

       CpuTimer0Regs.PRD.all = 0xFFFFFFFF;
       CpuTimer0Regs.TCR.bit.TSS = 0;
       EDIS;

       // В первую очередь необходимо задать вариант в переменной "n"
       // и внутренний период таймера ШИМ 6000
       // TPR = Fcpu / Fpwm / 2 = 60000000 / 5000 / 2 = 6000
```

```
	// Доп. деление на 2 нужно, т.к. таймер считает вверх-вниз
	drive.n = 1;
	drive.tpr = 6000;
	ramp.Init(&ramp);

	// Теперь нужно вызвать функцию "Init" для инициализации модели
	drive.Init(&drive);

	// Глобально разрешаем прерывания
	EINT;

		drive.cmpr1 =0;
		     drive.cmpr2 =0;
		     drive.cmpr3 = 0;

	// Бесконечный цикл
	while (1) {

	}
}

// Прерывание для расчёта модели двигателя
interrupt void controlIsr(void) {
	// Пользовательская функция с расчётом системы управления,
	// в которой осуществляется расчёт и присвоение уставок сравнения cmpr1,
cmpr2, cmpr3
	ramp.Calc(&ramp);
	freq = ramp.output;

	controlSystem();

	// Выполнение модели двигателя
	drive.Execute(&drive);

	parameters();




	skipCount++;

	if (skipCount >= skipBreakpoint) {
	       skipCount = 0;



	}

	// Очистка флагов прерывания
	EPwm1Regs.ETCLR.bit.INT = 1;
	PieCtrlRegs.PIEACK.bit.ACK3 = 1;
}
```