МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт химии и энергетики
(наименование института полностью)

Кафедра Электроснабжение и электротехника
(наименование)

13.03.02. Электроэнергетика и электротехника
(код и наименование направления подготовки / специальности)

Цифровые технологии в электроэнергетике
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка цифровой модели дифференциальной защиты силового трансформатора

Обучающийся А.Р. Бочкарева
(Инициалы Фамилия) (личная подпись)

Руководитель к.т.н., Д.А. Кретов
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант А.В. Прошина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2025

Аннотация

В выпускной квалификационной работе разработана цифровая модель дифференциальной защиты силового двухобмоточного трансформатора. Модель разработана с использованием элементов библиотеки MATLAB/Simulink. Для выполнения поставленной цели проведен обзор и анализ принципов функционирования цифровых дифференциальных защит силовых трансформаторов, а также представлено описание принципов функционирования дифференциальной защиты трансформатора.

Представлено подробное описание последовательности этапов разработки цифровой модели дифференциальной защиты силового трансформатора с описанием всех элементов разработанных программных блоков, а также алгоритмов из работы. Подробное описание позволит адаптировать разработанную модель для двухобмоточных трансформаторов различной мощности и типа.

Разработанная модель дифференциальной защиты протестирована и верифицирована на тестовой схеме с использованием двухобмоточного трансформатора ТМН-6300/110/10. Результаты тестирования модели на различных типах коротких замыканий позволяют утверждать об адекватности работы модели.

Пояснительная записка представлена на 61 листе формата А4, содержит 28 рисунков и 3 таблицы. Список используемых источников и используемой литературы содержит двадцать пять наименований. В приложении А представлен программный код модели дифференциальной защиты силового трансформатора на языке «Си».

Abstract

The graduation thesis, a digital model of differential protection of a two-winding power transformer is developed. The model is developed with the elements of the MATLAB/Simulink basic library. To achieve the graduation thesis goal, a review and analysis of the operating principles of power transformer digital differential protection was carried out, and an operating principles description for differential protection of a transformer is presented.

Th detailed description for the stages sequence of the digital model of differential protection of a power transformer developing is presented. with a description of all elements of the developed software blocks, as well as its algorithms. A detailed description allows to adopt the developed model for simulation with two-winding power transformers of a various rate and type.

The developed model of differential protection is tested and verified on a test-circuit using a two-winding transformer TMN-6300/110/10. The results of model testing with various short circuits types allow to state the model adequacy.

The graduation thesis is presented on 61 A4 sheets, contains 28 figures and 3 tables. The list of sources and references contains twenty-five titles.

Содержание

Введение

В настоящее время наблюдается уверенный переход устройств релейной защиты и автоматики на современные микропроцессорные технологии. Периоды развития устройств релейной защиты и автоматики электроэнергетических систем могут быть разделены на:

- период использования электромеханических систем релейной защиты и автоматики;
- период использования аналоговых устройств релейной защиты и автоматики;
- период использования цифровых устройств релейной защиты;
- период использования интеллектуальных устройств релейной защиты и автоматики.

В настоящее время в мире и в России наблюдается тренд перехода на интеллектуальные устройства релейной защиты и автоматики. Интеллектуальные устройства релейной защиты имеют отличительную особенность, заключающуюся в возможности интеграции с действующими стандартами МЭК 61850, а также применением специализированных алгоритмов действия релейной защиты и автоматики позволяющих более быстро реагировать на развивающиеся аварийные ситуации и предотвращать их.

Для разработки быстродействующих интеллектуальных устройств и алгоритмов релейной защиты и автоматики необходимо на этапе разработки использовать современный модельно-ориентированный подход. Применение модельно-ориентированного подхода позволяет существенно сократить время разработки микропроцессорных систем и в том числе блоков релейной защиты и автоматики. В целом модельно-ориентированный подход позволяет создавать цифровые модели микропроцессорных устройств и алгоритмов их функционирования для исследования их работы в различных ситуациях. Применение цифровых моделей позволяет ускорить процесс разработки

интеллектуальных устройств релейной защиты, а также снизить конечную стоимость продукта за счет отсутствия долгих натурных тестирований и испытаний разрабатываемых устройств. Модельно-ориентированный подход (МОП) в проектировании микропроцессорных устройств включает в себя следующие виды:

- модель в контуре управления (Model-In-the-Loop, MIL);
- программа в контуре управления (Software-In-the-Loop, SIL);
- процессор в контуре управления (Processor-In-the-Loop, PIL);
- аппаратные средства в контуре управления (Hardware-In-the-Loop, HIL).

Релейная защита и автоматика является важным элементом, обеспечивающим функционирование электроэнергетических систем. Применение современных быстродействующих алгоритмов позволит существенно снизить риски развития аварийных ситуаций и повреждения оборудования [20]. В связи с этим тематика выпускной квалификационной работы, связанная с разработкой цифровой модели релейной защиты, является актуальной.

Целью выпускной квалификационной работы является повышение эффективности исследования режимов работы микропроцессорных устройств релейной защиты силового двухобмоточного трансформатора за счет использования цифровой модели.

Задачи требующие решения для достижения поставленной цели:

- провести анализ и описать особенности функционирования дифференциальной защиты силового двухобмоточного трансформатора;
- разработать цифровую модель дифференциальной защиты силового двухобмоточного трансформатора;
- разработать тестовую схему для верификации разработанной цифровой модели дифференциальной защиты силового двухобмоточного трансформатора;

– выполнить верификацию разработанной цифровой модели дифференциальной защиты силового двухобмоточного трансформатора.

Проведенный анализ литературных источников и научных публикаций показал, что вопросы моделирования дифференциальных защит трансформаторов рассматривались в научных работах [7], [6], [2], [21], [13], [14], [12].

При выполнении поставленных задач необходимо учитывать особенности функционирования микропроцессорных устройств дифференциальной защиты силовых двухобмоточных трансформаторов российского и зарубежного производства. При этом при использовании данных по устройствам зарубежного производства необходимо учитывать специфику интеграции их функций работу Единой энергетической системы России (ЕЭС России). Разработанная цифровая модель должна функционировать с учетом универсальных методик расчета параметров уставок ее срабатывания.

1 Функции дифференциальной защиты силового двухобмоточного трансформатора

К основным показателям релейной защиты, характеризующим ее основные функции относятся [1]:

- чувствительность;
- селективность.

Дифференциальная защита силового трансформатора является защитой, обеспечивающей абсолютную селективность, которая достигается установкой измерительных трансформаторов тока на каждой из обмоток защищаемого силового трансформатора [5].

Важным и необходимым условием правильного функционирования дифференциальной защиты силового трансформатора является баланс сравниваемых токов по каждой из обмоток для нагрузочного режима и режима повреждения вне области защищаемого объекта. Для обеспечения баланса сравниваемых токов дифференциальная защита силовых трансформаторов должна обеспечивать предварительное преобразование измеряемых токов на каждой из обмоток. Преобразование токов на каждой из обмоток силового трансформатора должно выполняться так как [4]:

- обмотки силового трансформатора выполняются на различные номинальные напряжения, а также могут быть рассчитаны на различные номинальные мощности, что справедливо для трехобмоточных и автотрансформаторов;
- номинальный ток защищаемого силового трансформатора не в полной мере соответствует номинальному току измерительного трансформатора тока устанавливаемого на каждой из обмоток силового трансформатора;
- схема и группа соединения обмоток защищаемого силового трансформатора обеспечивает появление фазового сдвига между токами в каждой из обмоток защищаемого силового трансформатора;

– при однофазном коротком замыкании на землю со стороны обмотки, соединенной по схеме «звезда с заземленной нейтралью» требуется обеспечить компенсацию составляющей тока нулевой последовательности для предотвращения ложных срабатываний.

Корректировка измеряемого тока на каждой из обмоток силового трансформатора выполняется для аналоговых релейных защит за счет:

– подбора коэффициентов передачи для каждого измерительного трансформатора тока;
– применения различных схем включения вторичных обмоток измерительных трансформаторов тока;
– установки выравнивающих трансформаторов или автотрансформаторов.

Для цифровых устройств релейной защиты корректировка токов выполняется за счет умножения рассчитанных векторов входных токов отдельных сторон на соответствующие комплексные коэффициенты. При этом стоит отметить, что комплексные коэффициенты определяются системой автоматически исходя из следующих данных:

– номинальной мощности защищаемого силового трансформатора;
– номинального напряжения защищаемого силового трансформатора;
– схемы и группы соединения обмоток защищаемого силового трансформатора.

Особенностью цифровых защит является практически полная корректировка коэффициентов передачи каждого измерительного трансформатора тока устанавливаемого на всех обмотках защищаемого силового трансформатора [18].

На рисунке 1 показана схема дифференциальной защиты силового двухобмоточного трансформатора при внешнем однофазным коротким замыканием на землю и отсутствием компенсации тока I_0.

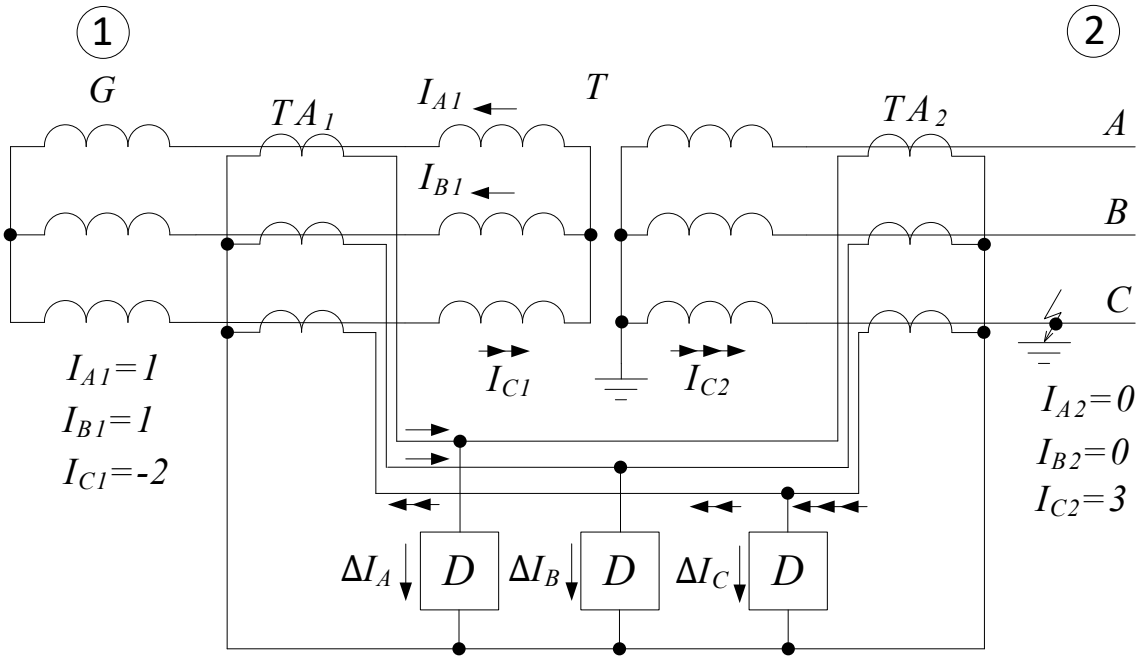

Рисунок 1 – Распределение токов в цепях дифференциальной защиты при внешнем однофазном коротком замыкании на землю без компенсации тока I_0

На схеме, представленной на рисунке 1 обмотки измерительных трансформаторов тока TA_1 и TA_2 включены по дифференциальной схеме образуя тем самым пофазную дифференциальную защиту силового трансформатора (T) с реагирующими элементами (D). Питание защищаемого двухобмоточного силового трансформатора выполняется от генератора (G) – узел 1 схемы (рисунок 1). В узле 2 схемы (рисунок 1) происходит короткое замыкание фазы C на землю. При этом со стороны узла 2 обмотки силового двухобмоточного трансформатора соединены в звезду с заземленной нейтралью. При возникновении на фазе C однофазного короткого замыкания вне зоны действия дифференциальной защиты силового трансформатора токи в относительных единицах в фазах A, B и C становятся равными: $I_{A2} = 0$, $I_{B2} = 0$, $I_{C2} = 3$ соответственно. Кроме того, токи на стороне силового трансформатора с схемой соединения «звезда с заземленной нейтралью» содержат составляющие нулевой последовательности I_0, которые не могут протекать в обмотке силового трансформатора со стороны генератора, так как отсутствует точка заземления. Токи в плечах дифференциальной защиты силового трансформатора со стороны генератора становятся равными: $I_{A1} = 1$,

$I_{B1} = 1$, $I_{C1} = -2$ соответственно. В результате того, что фазные токи в обмотках силового трансформатора не соответствуют в каждом фазном реагирующем элементе (D) возникают дифференциальные токи ΔI_A, ΔI_B и ΔI_C, а общий дифференциальный ток $\Delta I = 1$. Соответственно баланс токов нарушается и реагирующие устройства могут ложно отработать на отключение силового трансформатора, а это не соответствует принципу абсолютной селективности дифференциальной защиты. Небаланс токов при внешних однофазных коротких замыканиях на землю, в цифровых микропроцессорных устройства защиты корректируется программой, как показано на рисунке 2.

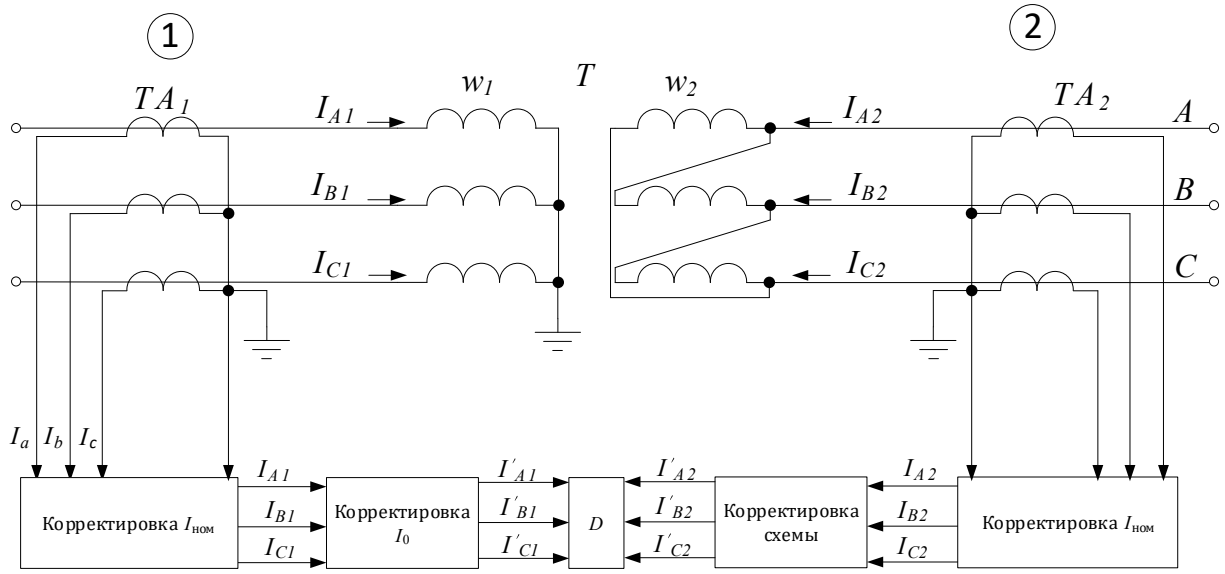

Рисунок 2 - Корректировка фазных токов в микропроцессорной дифференциальной защите

Перед сравнением токов в реагирующем элементе (D) (рисунок 2) вторичные токи от измерительных трансформаторов тока TA_1 и TA_2 умножаются на корректирующие коэффициенты $К_{ном.i}$, которые генерируются в блоках корректировки номинального тока «Корректировка $I_{ном}$» (рисунок 2) как со стороны обмотки w_1, $i = 1$, так и со стороны обмотки w_2, $i = 2$. Корректирующие действительные коэффициенты определяются выражениями [17]:

$$К_{ном.i} = \frac{I_{ном.TA_i}}{I_{ном.w_i}} \quad (1)$$

где $I_{ном.TA_i}$ – номинальное значение тока измерительного трансформатора тока установленного на i-й обмотке силового двухобмоточного трансформатора, А;

$I_{ном.w_i}$ - номинальное значение тока i-й обмотки силового двухобмоточного трансформатора, А.

Номинальные токи обмоток силового трансформатора для выражения (1) определяется для двухобмоточного трансформатора выражением [8]:

$$I_{ном.w_i} = \frac{S_{ном.T}}{\sqrt{3} \cdot U_{ном.w_i}} \quad (2)$$

где $S_{ном.T}$ – номинальная полная мощность силового двухобмоточного трансформатора, ВА;

$U_{ном.w_i}$ – номинальное значение напряжения i-й обмотки силового двухобмоточного трансформатора, В.

В результате ввода в микропроцессорный терминал дифференциальной защиты силового трансформатора его паспортных данных, а также паспортных данных измерительных трансформаторов тока происходит программное устранение небаланса токов вызванных разностью номинальных токов силового и измерительных трансформаторов по выражениям (1) и (2).

Программная корректировка небаланса дифференциальной цепи при внешних однофазных коротких замыканиях осуществляется в блоке «Корректировка I_0» (рисунок 2). Для этого микропроцессорный терминал выполняет измерение фазных токов и рассчитывает составляющие нулевой последовательности со стороны обмотки силового двухобмоточного трансформатора, соединенной по схеме «звезда с заземленной нейтралью».

Ток нулевой последовательности определяется в комплексной форме по выражению:

$$\underline{I}_0 = \frac{\left(\underline{I}_A + \underline{I}_B + \underline{I}_C\right)}{3} \quad (3)$$

где $\underline{I}_A$ – комплексное измеренное значение тока фазы A;

$\underline{I}_B$ – комплексное измеренное значение тока фазы B;

$\underline{I}_C$ – комплексное измеренное значение тока фазы C.

После определения комплексного значения тока нулевой последовательности по выражению (3) полученное значение $\underline{I}_0$ вычитается из фазных токов обмотки силового двухобмоточного трансформатора, соединенной по схеме «звезда с заземленной нейтралью» по выражению:

$$\underline{I}'_j = \underline{I}_j - \underline{I}_0 \quad (4)$$

где $\underline{I}_j$ - комплексное измеренное значение тока j-й фазы.

Применение корректировки токов представленной выражениями (1) - (4), согласно рисунка 2 позволяет устранить небаланс токов вызванный наличием токов нулевой последовательности (I_0).

На рисунке 2, в цепи корректировки токов также показан блок «Корректировка схемы». Данный блок корректирует значения токов со стороны обмотки силового двухобмоточного трансформатора, соединенной по схеме «треугольник». Блок «Корректировка схемы» учитывает сдвиг по фазе и изменения по модулю сравниваемых токов вызванных различием схем соединения обмоток защищаемого силового двухобмоточного трансформатора.

После корректировки токов согласно схеме, представленной на рисунке 2 алгоритм действия дифференциальной защиты в реагирующем элементе (D) выполняется согласно алгоритму представленного на рисунке 3.

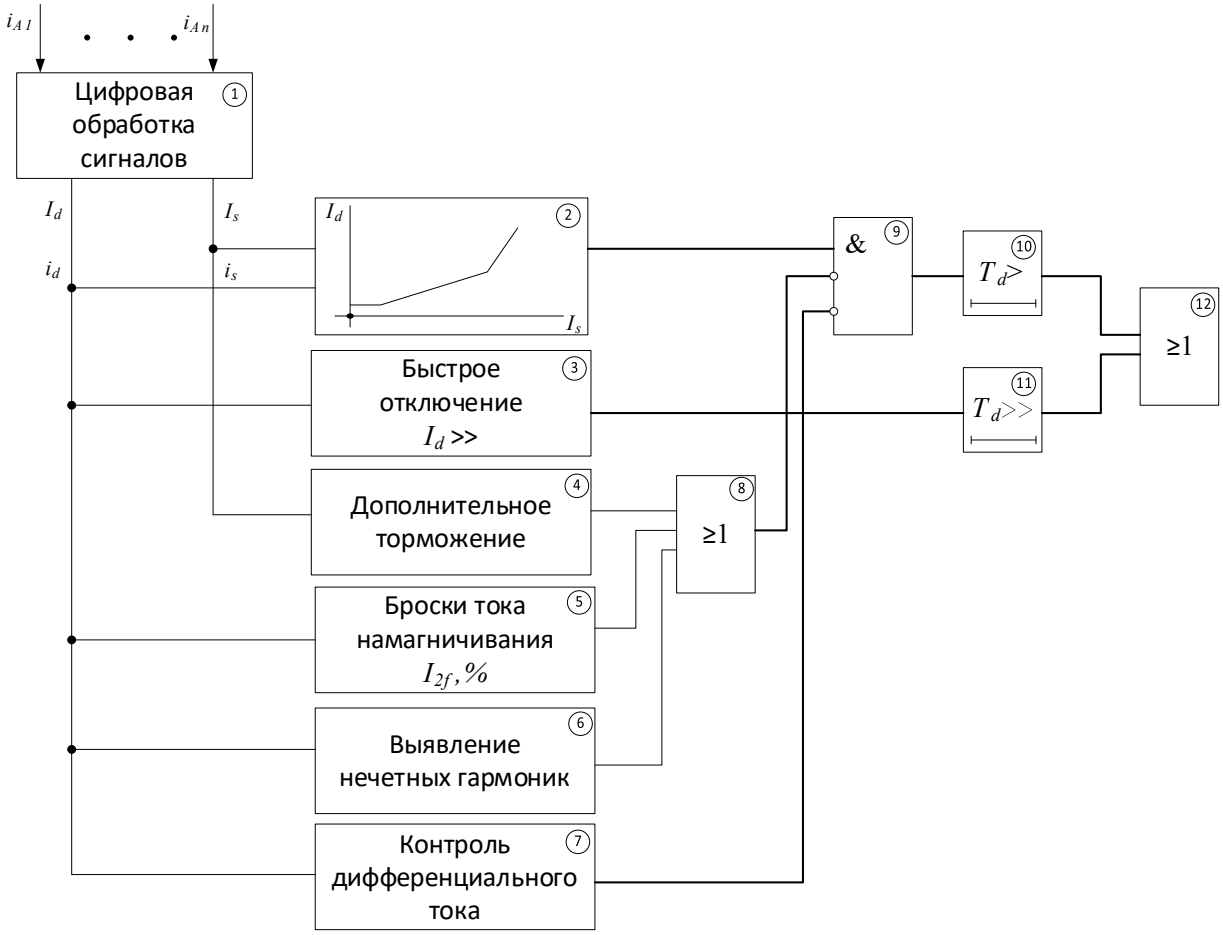

Рисунок 3 - Алгоритм действия дифференциальной защиты

Функциональные блоки алгоритма действия дифференциальной защиты силового двухобмоточного трансформатора (рисунок 3) выполняемого реагирующим элементом (D) согласно схеме, представленной на рисунке 2 представлены в таблице 1.

Таблица 1 - Описание блоков алгоритма действия дифференциальной защиты силового двухобмоточного трансформатора

Номер блока	Наименование блока	Описание блока
1	Блок цифровой обработки сигналов	Блок выполняет цифровую обработку сигналов с формированием цифровых значений дифференциального (I_d) и тормозного (I_s) токов
2	Блок характеристики отключения	В блоке формируется основная характеристика отключения $I_d = f(I_s)$ которая обеспечивает защиту с абсолютной селективностью с торможением от суммы абсолютных значений токов
3	Блок быстрого отключения	Блок выполняет быстрое отключение защищаемого объекта при выполнении критерия существенного превышения значения дифференциального тока $I_d \gg$. Данный критерий соответствует короткому замыканию в зоне действия дифференциальной защиты. Выполнение данного критерия не требует применения тормозной характеристики (блок 11).
4	Блок дополнительного торможения	Блок выполняет функцию формирования дополнительного торможения при внешних коротких замыканиях (не входящих в зону действия дифференциальной защиты). Блок фиксирует значения токов от измерительных трансформаторов тока в начальный момент возникновения короткого замыкания и оценивает входит ли короткое замыкание в защищаемую зону. При этом фиксация токов происходит в начальный момент времени возникновения короткого замыкания, т.е. до момента начала насыщения измерительных трансформаторов тока.
5	Блок определения броска тока намагничивания	В блоке выполняется гармонический анализ дифференциального тока (i_d) по результатам которого выявляется увеличение амплитуды второй гармонической составляющей (I_{2f}) с частотой $f_2 = 100$ (Гц) являющейся характеристикой процесса насыщения защищаемого силового трансформатора в момент его включения в сеть. При выявлении наличия процесса насыщения защищаемого силового трансформатора происходит блокировка действия дифференциальной защиты с помощью блока 9 «&».При этом быстрое отключение, т.е. действие блока 3 не блокируется.

Продолжение таблицы 1

Номер блока	Наименование блока	Описание блока
6	Блок выявления нечетных гармоник	В блоке выполняется гармонический анализ дифференциального тока для выявления наличия и амплитуды третьей гармоники (I_{3f}) с частотой $f_3 = 150$ (Гц) и пятой гармоники (I_{5f}) с частотой $f_5 = 250$ (Гц). Данный блок, также как и блок 5 выполняет функцию оценки насыщения защищаемого силового трансформатора так как при увеличение мгновенного значения напряжения, подводимого к защищаемому силовому трансформатору, вызывает рост индукции с симметричным насыщением магнитопровода. Именно присутствие третьей и пятой гармоник в дифференциальном токе определяет наличие данного процесса. Данный блок, также как и блок 5 блокирует действие основного канала отключения дифференциальной защиты формируемый блоком 2.
7	Блок контроля дифференциального тока	Блок контролирует исправность токовых цепей за счет контроля дифференциального тока (I_d). Блок блокирует отключение по основному каналу (блок 2). Блокировка отключения по основному каналу при условии превышения значения дифференциального тока выполняется на основании превышения установленного значения тока небаланса $I_{d.\text{н.}}$ превышающего уставку по току начального срабатывания $I_{d.\text{уст.}}$ в течении заданного времени, обычно $t \leq 2$ с. Возрастание тока небаланса $I_{d.\text{н.}}$ характеризует неисправность дифференциальных токовых цепей защиты. Данный блок необходим для предотвращения неселективного отключения защищаемого силового трансформатора при возникновении внешних коротких замыканий.
8	Блок сравнения	Блок выполняет сравнение сигналов от блоков 4, 5 и 6 и выдает наличие превышение хотя бы от одного из блоков значения «1».
9	Логический блок «И»	Блок определяет необходимость блокировки действия основного канали дифференциальной защиты
10	Блок тормозной характеристики	Блок формирует выдержку времени в зависимости от сигнала формируемого основным каналом срабатывания дифференциальной защиты (блок 2) на основании используемой тормозной характеристики.

Окончание таблицы 1

Номер блока	Наименование блока	Описание блока
11	Блок ускорения срабатывания	Блок используется для ускорения срабатывания дифференциальной защиты при условии выполнения уставки в блоке 3.
12	Блок сравнения	Блок предназначен для формирования сигнала отключения защищаемого объекта. Сигнал отключение соответствует выходным значениям выше «1».

Выводы по разделу.

Рассмотрены особенности функционирования дифференциальной защиты силового двухобмоточного трансформатора. Рассмотрены особенности действия аналоговой защиты и современной микропроцессорной защиты.

Рассмотрены схемы корректировки значений тока в микропроцессорном устройстве дифференциальной защиты для обеспечения правильного срабатывания защиты при внутренних коротких замыканиях и блокировки срабатывания при внешних коротких замыканиях.

Рассмотрен алгоритм действия цифровой дифференциальной защиты силового трансформатора, который будет использован при разработке цифровой модели дифференциальной защиты силового двухобмоточного трансформатора.

2 Выбор объекта для цифровой модели дифференциальной защиты силового трансформатора

Для выполнения поставленной цели выпускной квалификационной работы необходимо выбрать объект – силовой двухобмоточный трансформатор, на котором будет тестироваться и верифицироваться разработанная модель цифровой дифференциальной защиты.

Для разработки модели цифровой дифференциальной защиты силового двухобмоточного трансформатора необходимо определить уставки срабатывания микропроцессорного терминала. Для повышения точности и достоверности разрабатываемой модели цифровой дифференциальной защиты в качестве силового трансформатора примем трансформатор марки ТМН 6300/110/10 кВ, а также примем расчетные уставки, приведенные в методических рекомендациях отечественного производителя микропроцессорных устройств релейной защиты и автоматики ООО «НТЦ «Механотроника» [10]. Уставки микропроцессорного терминала дифференциальной защиты БМРЗ-ТД силового трансформатора ТМН 6300/110/10 представлены в таблице 2.

Таблица 2 - Значения уставок терминала БМРЗ-ТД для ТМН-6300/110/10

Наименование уставки	Единица измерения	Обозначение уставки	Величина уставки
Ток срабатывания дифференциальной токовой отсечки (ДТО)	о.е.	$I_{ДТО}$	6,41
Начальный ток срабатывания дифференциальной защиты трансформатора	о.е.	$I_{ДЗТ.нач.}$	0,3
Коэффициент торможения второго участка дифференциальной защиты трансформатора	-	$К_{торм.2}$	0,44
Коэффициент торможения третьего участка дифференциальной защиты трансформатора	-	$К_{торм.3}$	0,65
Блокировка броска тока намагничивания	о.е.	$К_{гарм.2}$	0,15
Длительность блокировки броска тока намагничивания	с	$t_{гарм.2}$	1

Схема распределительного устройства с трансформатором ТМН-6300/110/10 и микропроцессорными терминалами БМРЗ-ТД, согласно [10], [9], представлена на рисунке 4.

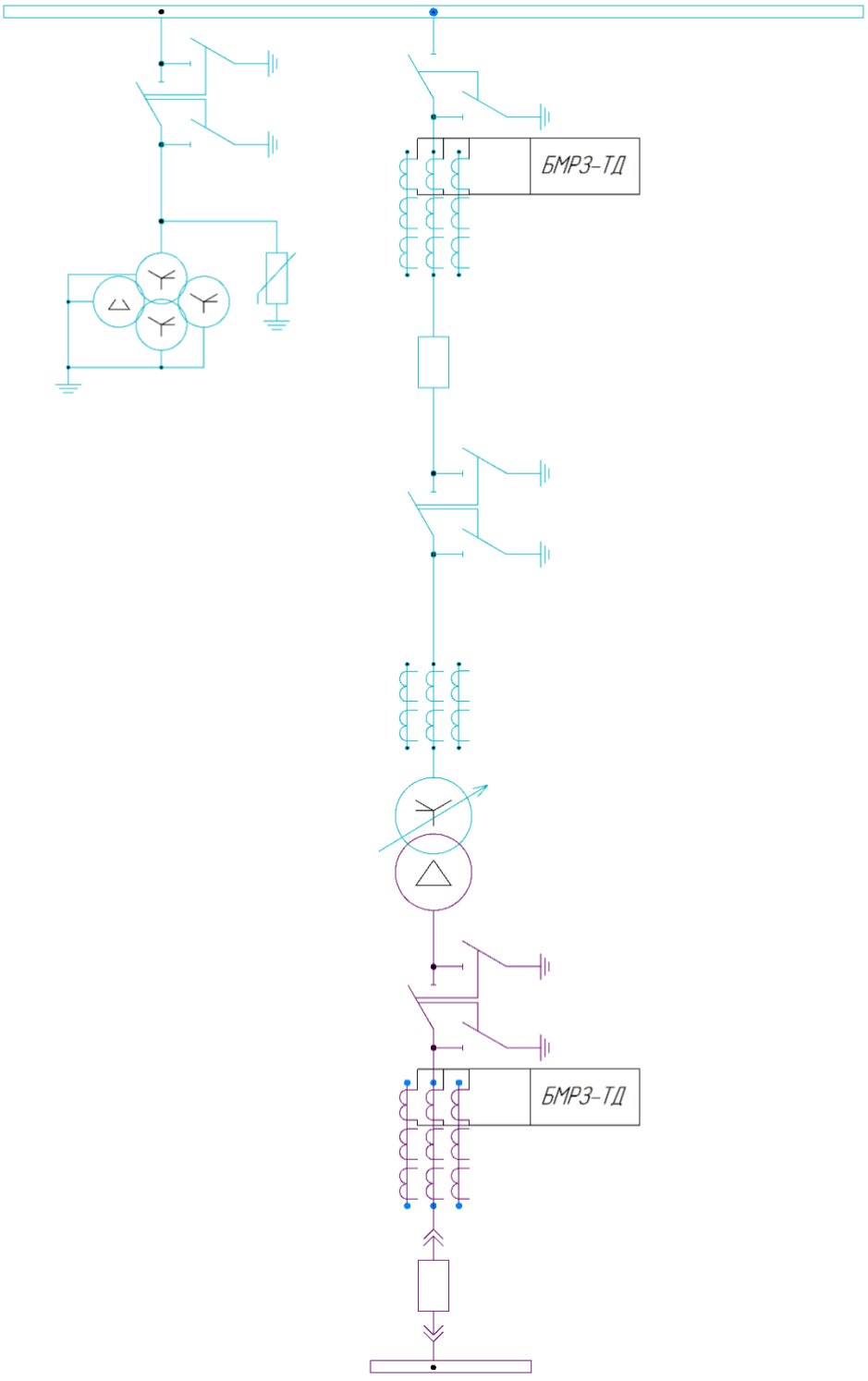

Рисунок 4 - Схема распределительного устройства с трансформатором ТМН-6300/110/10 и терминалами БМРЗ-ТД

Паспортные параметры трансформатора ТМН-6300/110/10 представлены вы таблице 3 [15].

Таблица 3 - Паспортные данные трансформатора ТМН-6300/110/10

Наименование параметра	Единица измерения параметра	Обозначение параметра	Значение параметра
Марка трансформатора	-	-	ТМН
Полная мощность	кВА	$S_{\text{ном.транс}}$	6300
Напряжение первичной обмотки	кВ	$U_{\text{ном.ВН}}$	115
Напряжение вторичной обмотки	кВ	$U_{\text{ном.ВН}}$	11
Схема соединения первичной обмотки	-	-	Ун
Схема соединения вторичной обмотки	-	-	D
Группа соединения обмоток	-	-	11
Напряжение короткого замыкания	%	$u_{\text{кз.\%}}$	10,5
Ток холостого хода	%	$i_{\text{хх,\%}}$	0,56
Потери короткого замыкания	кВт	$P_{\text{кз}}$	35
Потери холостого хода	кВт	$P_{\text{хх}}$	6,5
Частота сети	Гц	f	50

Выводы по разделу.

Для разработки, тестирования и верификации модели цифровой дифференциальной защиты силового двухобмоточного трансформатора выбран трансформатор марки ТМН-6300/110/10 кВ. Питание трансформатора согласно схемы распределительного устройства одностороннее со стороны 110 кВ. Нагрузка подключается со стороны 10 кВ.

Для выбранного трансформатора определены паспортные данные уставки микропроцессорного терминала БМРЗ-ТД.

Уставки микропроцессорного терминала БМРЗ-ТД производства ООО «НТЦ «Механотроника» будут использованы при разработке модели цифровой дифференциальной защиты силового трансформатора.

3 Разработка программных алгоритмов дифференциальной защиты силового трансформатора

Для разработки программных алгоритмов дифференциальной защиты силовых двухобмоточных трансформаторов необходимо определить объект защиты – силовой двухобмоточный трансформатор. Разработка программных алгоритмов работы дифференциальной защиты силового двухобмоточного трансформатора проводится с использованием программного продукта MATLAB с программной средой динамического моделирования Simulink [23]. При разработке программных алгоритмов работы дифференциальной защиты силового двухобмоточного трансформатора учитываются результаты, полученные в первом и третьем разделах выпускной квалификационной работы.

3.1 Разработка блока измерения токов

Блок измерения токов – является программным аналогом измерительных трансформаторов тока устанавливаемых со стороны первичной и вторичной обмоток двухобмоточного силового трансформатора. Блок измерения токов (БИТ) позволит моделировать коэффициент трансформации измерительных трансформаторов тока. Стоит отметить, что разработанный БИТ не учитывает характеристики насыщения, которые являются неотъемлемой частью реальных измерительных трансформаторов тока. В целом БИТ может быть представлен как аналог цифровых датчиков тока устанавливаемых со стороны первичной и вторичной обмоток.

Для разработки БИТ используются модели элементов электроэнергетической системы из библиотеки элементов MATLAB/Simulink [22]:

- блок «Current Measurement» использован для измерения мгновенных значений фазного тока;

– блок «Gain» использован для моделирования коэффициента трансформации реального измерительного трансформатора тока.

Блок схема алгоритма функционирования БИТ представлена на рисунке 5.

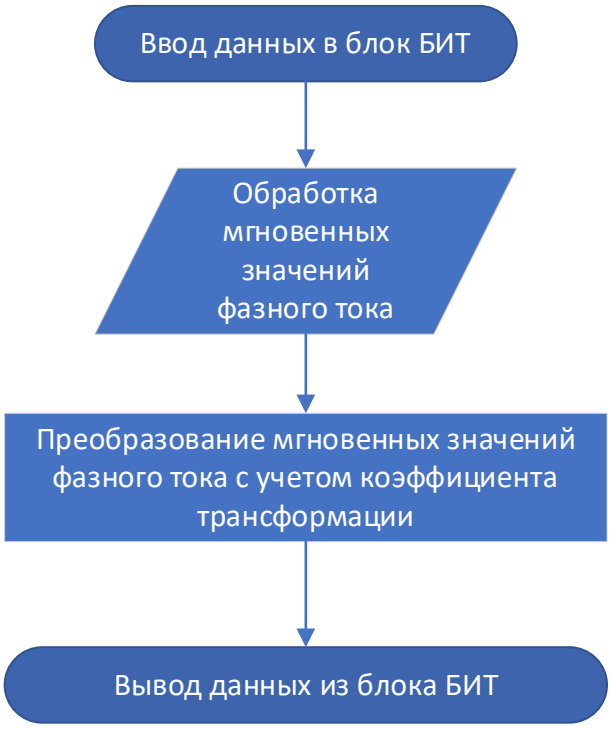

Рисунок 5 - Алгоритм функционирования блока измерения тока

Модель блока измерения токов, разработанная в MATLAB/Simulink [16] представлена на рисунке 6. Переменные модели БИТ:

– входные переменные БИТ: A1_HV, B1_HV и C1_HV – являются переменными, используемыми для передачи сигналов тока фаз A, B и C соответственно;
– выходные переменные БИТ: A2_HV, B2_HV и C2_HV – являются переменными, используемыми для передачи сигналов тока фаз A, B и C соответственно;
– выходные переменные БИТ: I_A_HV, I_B_HV и I_C_HV – являются переменными сигнала измеренных мгновенных значений тока в фазах A, B и C соответственно.

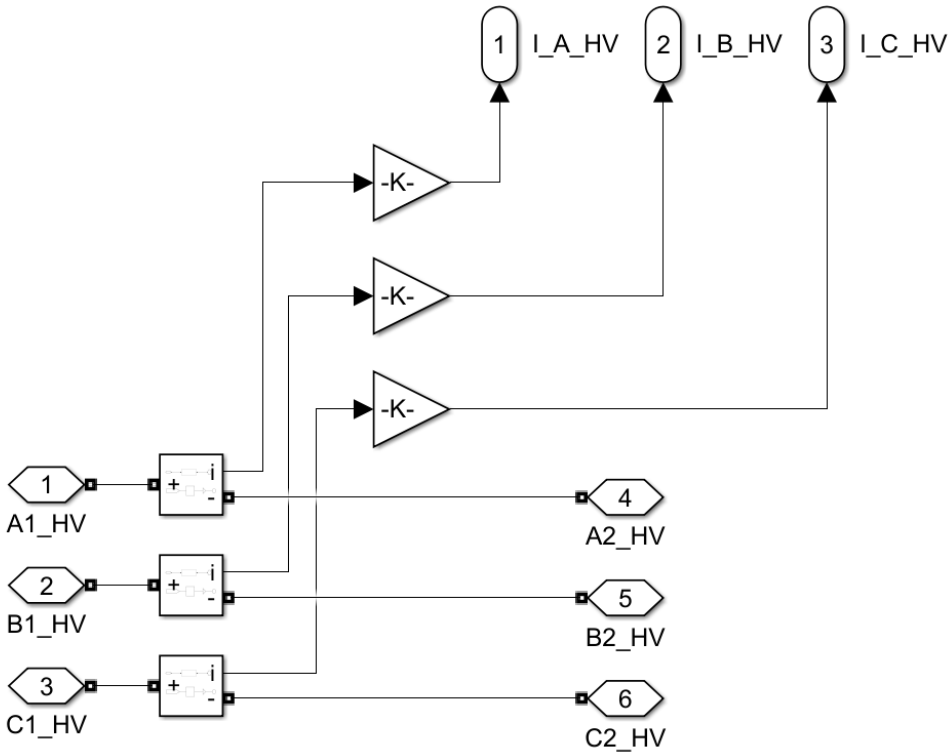

Рисунок 6 - Модель блока измерения токов со стороны обмотки высокого напряжения двухобмоточного трансформатора

При разработке БИТ не учитывается схема соединения трансформаторов тока. Ввод значений в блок «Gain» выполняется как:

$$\text{К} = \frac{1}{I_{\text{ном}.TA_i}}$$

Для использования сигналов мгновенных значений фазных токов в модели дифференциальной защиты силового двухобмоточного трансформатора (сигналы переменных I_A_HV, I_B_HV и I_C_HV, рисунок 6) необходимо предварительно выполнить их корректировку.

3.2 Разработка блока корректировки токов

Корректировка сигналов переменных I_A_HV, I_B_HV и I_C_HV, рисунок 6) выполняется согласно выражениям (1) и (2). Блок схема блока корректировки токов (БКТ) представлена на рисунке 7.

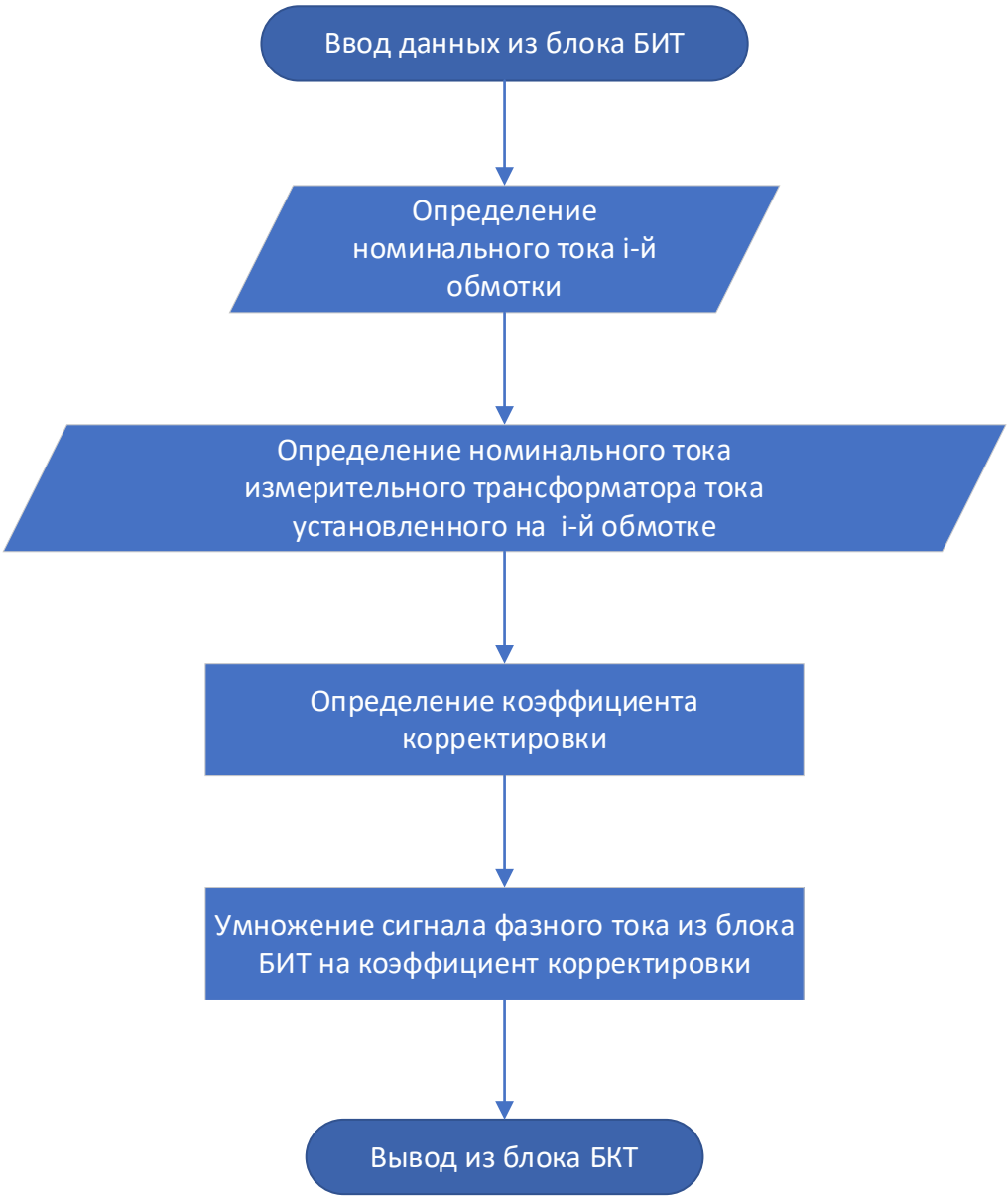

Рисунок 7 - Алгоритм функционирования блока корректировки тока

Модель блока корректировки токов, разработанная в MATLAB/Simulink представлена на рисунке 8.

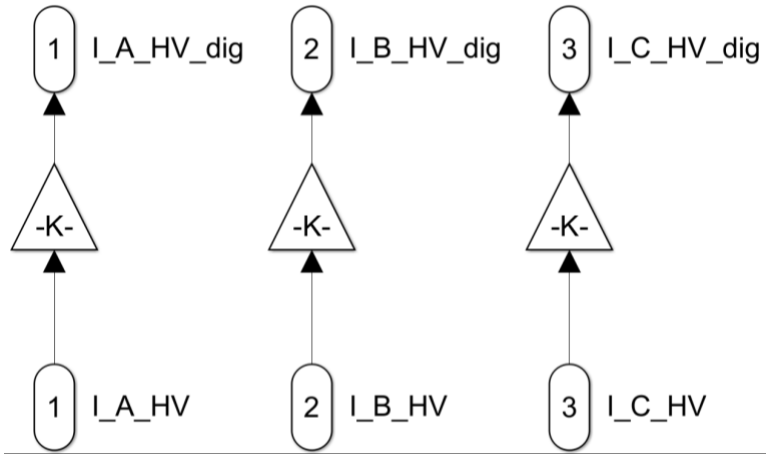

Рисунок 8 - Модель блока измерения токов со стороны обмотки высокого напряжения двухобмоточного трансформатора

Переменные модели БКТ (рисунок 8):

- входные переменные БКТ: I_A_HV, I_B_HV и I_C_HV – являются выходными переменными блока измерения токов (БИТ), используемыми для передачи сигналов тока фаз A, B и C соответственно;
- выходные переменные БКТ: A2_HV_dig, B2_HV_dig и C2_HV_dig – являются переменными, используемыми для передачи скорректированных сигналов тока фаз A, B и C соответственно.

Ввод значений в блок «Gain» БИТ выполняется как:

$$\text{К} = \frac{I_{\text{ном}.TA_i}}{I_{\text{ном}.w_i}}$$

Выходные сигналы блока корректировки тока используются в модели дифференциальной защиты силового двухобмоточного трансформатора.

3.3 Разработка блока измерения гармоник тока

Функционирование цифровых микропроцессорных терминалов релейной защиты основано на предварительном выделении основной гармоники тока или напряжения, в зависимости от типа и вида микропроцессорной релейной защиты, поэтому при разработке модели цифровой дифференциальной защиты силового двухобмоточного трансформатора в рамках выполнения выпускной квалификационной работы необходимо разработать блок измерения действующих значений гармоник тока – блок измерения гармоник (БИГ).

Модель БИГ для фазы А обмотки высокого напряжения силового двухобмоточного трансформатора разработанная в MATДAB/Simulink представлена на рисунке 9.

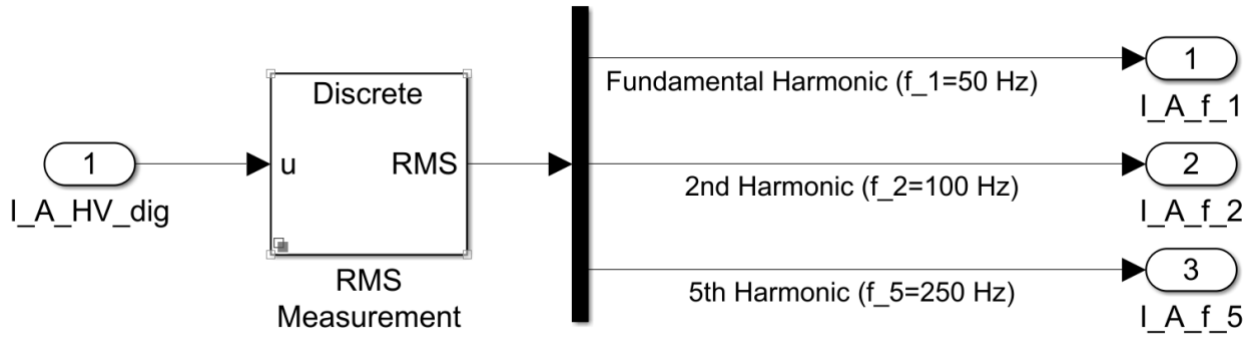

Рисунок 9 - Модель блока измерения действующих значений первой, второй и пятой гармоник тока

К функциям БИГ относится измерение действующих значений первой – основной гармоники тока ($f_1 = 50$ Гц), второй гармоники тока ($f_2 = 100$ Гц) и пятой гармоники тока ($f_5 = 250$ Гц). Действующие значений второй и пятой гармоник тока позволят реализовать в модели цифровой дифференциальной защиты силового двухобмоточного трансформатора алгоритм блокировки действия при появлении броска тока намагничивания (БТН) в момент подключения трансформатора к сети.

3.4 Разработка блока первой ступени дифференциальной защиты двухобмоточного трансформатора

Первая ступень дифференциальной защиты трансформатора является дифференциальной токовой отсечкой, т.е. при появлении дифференциального тока происходит срабатывание ступени и поврежденный участок отключается. При этом необходимо учитывать ток небаланса, который может возникать в дифференциальной цепи защиты. Для снижения ложного срабатывания первой ступени необходимо ввести уставку тока небаланса. Согласно методическим указаниям производителей рекомендуется принимать величину уставки равную 0,3. С учетом этого значения уставки тока небаланса алгоритм работы первой ступени токовой защиты представлен на рисунке 10.

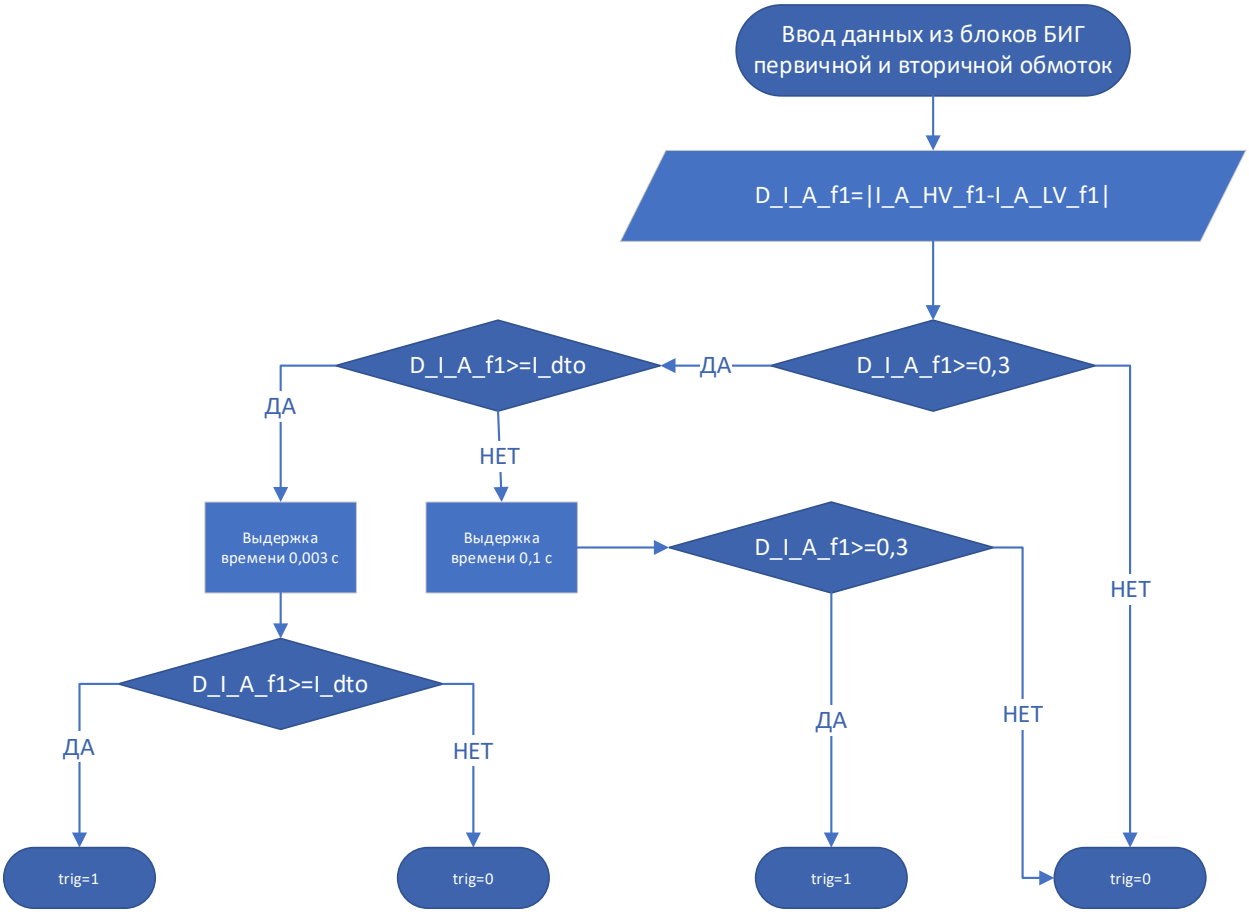

Рисунок 10 - Блок схема алгоритма работы первой ступени дифференциальной токовой защиты трансформатора

Модель первой ступени дифференциальной токовой защиты силового двухобмоточного трансформатора для фазы А разработанная в MATLAB/Simulink представлена на рисунке 11.

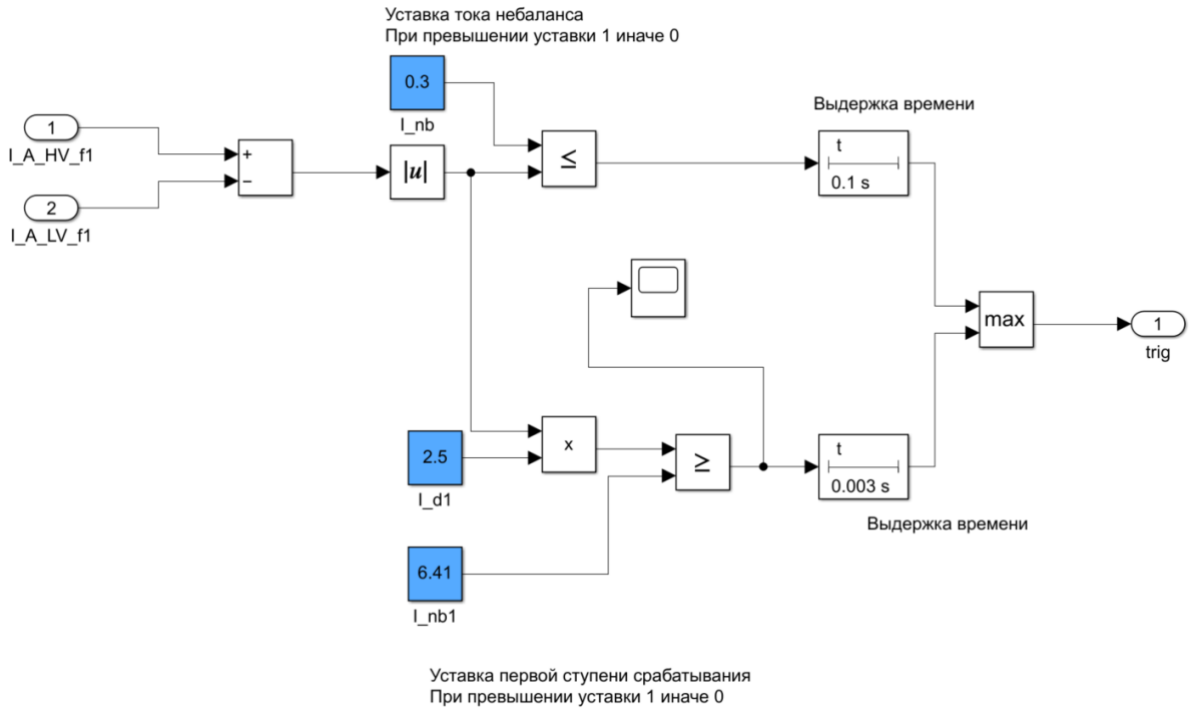

Рисунок 11 – Модель первой ступени дифференциальной токовой защиты силового двухобмоточного трансформатора по фазе А

В модели первой ступени дифференциальной токовой защиты силового двухобмоточного трансформатора (рисунок 11) используются:

- блок «Minus» для определения величины дифференциального фазного тока;
- блок «Abs» для получения модуля дифференциального фазного тока;
- блок «Constant» для ввода уставки тока небаланса;
- блок «Relational Operator» для сравнения модуля дифференциального фазного тока с величиной уставки. Выходной сигнал имеет тип Boolean, т.е. при превышении уставки тока небаланса дифференциальным фазным током выходной сигнал равен 1, при отсутствии превышения сигнал имеет значение равное 0;

– блок «On/Off Delay» моделирует выдержку времени. Если в течении заданного для блока времени сигнал типа Boolean не изменятся, то на выходе блок формируется логический сигнал аналогичный по значению входному. В противном случае сигнал меняется на 0;
– блок «MinMax» используется в модели блоков для определения наличия логических сигналов после блоков выдержки времени. Если с обоих блоков выдержки времени поступает сигнал 1, то выходной сигнал также равен 1. Если сигнал 1 поступает только с одного из блоков, то выходной сигнал также равен 1. Это позволяет сформировать управляющий сигнал срабатывания первой ступени дифференциальной токовой отсечки по двум критериям.

3.5 Разработка блока определения тормозного тока

Для разработки адекватной модели дифференциальной защиты силового двухобмоточного трансформатора необходимо обязательный учет в модели значений тормозного тока. Значение тормозного тока вычисляется для каждой фазы силового двухобмоточного трансформатора. При разработке блока определения тормозного тока (БОТТ) необходимо учесть, что существуют различные алгоритмы его определения, однако для реализации функций БОТТ необходимо учитывать инвариантность к токам коротких замыканий, возникающих вне зоны действия дифференциальной защиты вне зависимости от числа контролируемых сторон [3]. Применение данного допущения позволит повысить универсальность разработанной модели дифференциальной защиты и адаптировать ее к работе с другими типа силовых трансформаторов.

Первый алгоритм определения тормозного тока инвариантный к количеству подключений основан на определении максимального действующего значения основной гармоники тока после измерительного трансформатора тока.

Для двухобмоточного трансформатора данный алгоритм можно записать в виде:

$$I_{\text{торм.вар}-1} = max\{|I_{A1}|, |I_{A2}|, |I_{B1}|, |I_{B2}|, |I_{C1}|, |I_{C2}|\} \quad (5)$$

где I_{A1}, I_{B1}, I_{C1} – модули действующих значений основной гармоники тока первичной обмотки силового трансформатора после измерительного трансформатора тока для фаз А, В и С соответственно;

I_{A1}, I_{B1}, I_{C1} – модули действующих значений основной гармоники тока вторичной обмотки силового трансформатора после измерительного трансформатора тока для фаз А, В и С соответственно.

Второй алгоритм определения тормозного тока инвариантный к количеству подключений основан на определении полу суммы действующих значений основной гармоники тока после измерительного трансформатора тока.

Для двухобмоточного трансформатора данный алгоритм можно записать в виде:

$$I_{\text{торм.вар}-2} = \frac{|I_{A1}| + |I_{A2}| + |I_{B1}| + |I_{B2}| + |I_{C1}| + |I_{C2}|}{2} \quad (6)$$

Первый алгоритм определения тормозного тока (5) характерен для микропроцессорных терминалов зарубежных производителей, например производителей ABB, в цифровых микропроцессорных терминалах релейной защиты типа RET, [19] и General Electric, в цифровых микропроцессорных терминалах релейной защиты типа T60 [21].

Второй алгоритм определения тормозного тока характерен для микропроцессорных терминалов отечественного производства, например терминалов ООО НПП «ЭКРА» [11].

Для разрабатываемой модели дифференциальной защиты примем алгоритм определения тормозного тока как комбинированный из алгоритмов (5) и (6), при этом алгоритмы (5) и (6) модифицируем с учетом расчета тормозного тока для каждой фазы двухобмоточного силового трансформатора.

Итоговый алгоритм определения тормозного тока разрабатываемой модели цифровой дифференциальной защиты силового двухобмоточного трансформатора представим на рисунке 12.

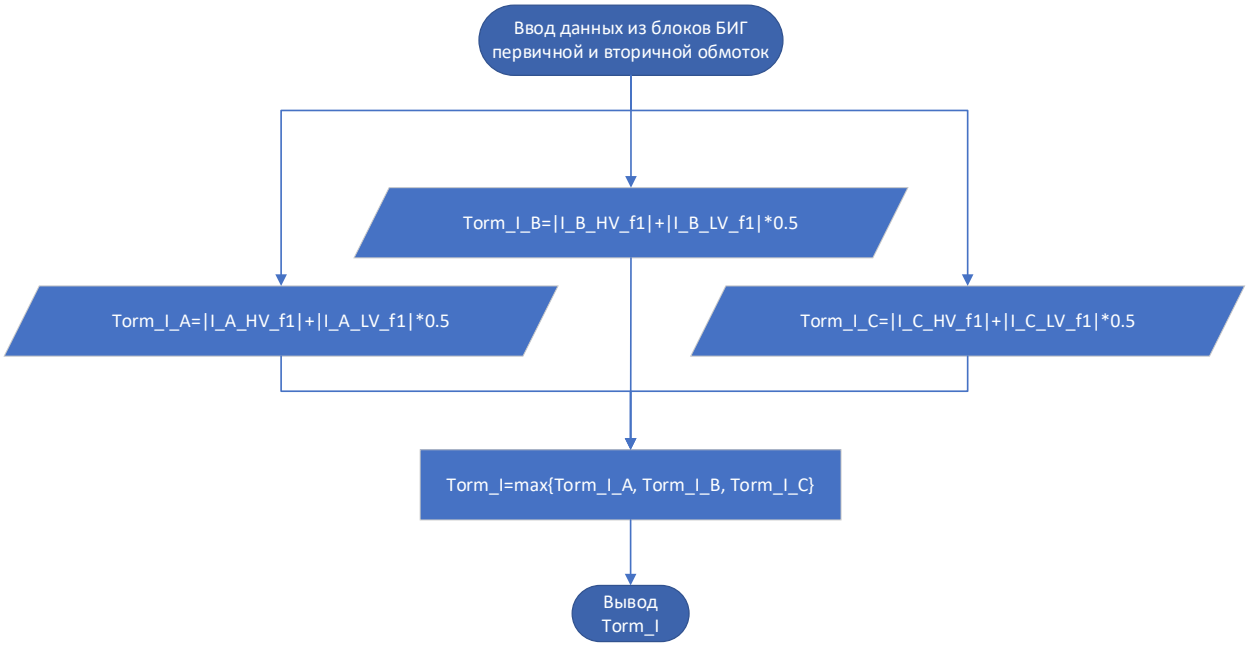

Рисунок 12 - Алгоритм определения тормозного тока

Модель БОТТ, разработанная с использованием элементов библиотеки MATLAB/Simulink представлена на рисунке 13.

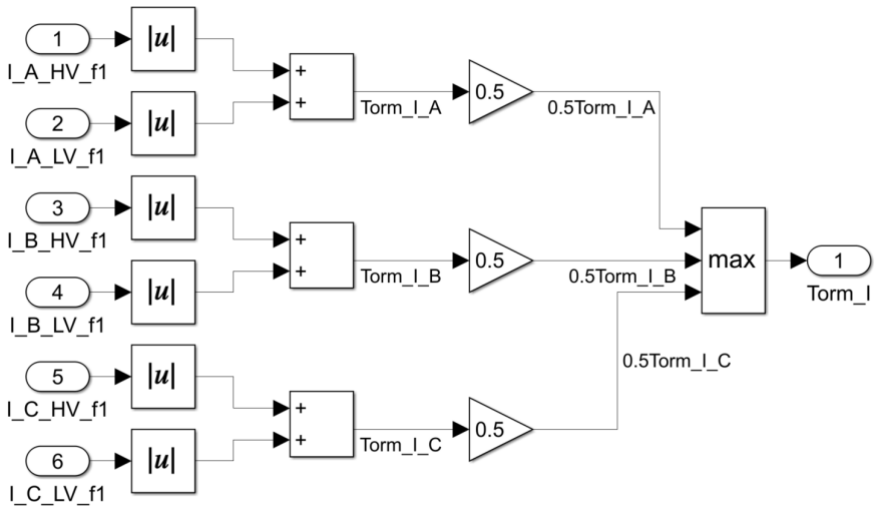

Рисунок 13 – Модель блока определения тормозного тока

Модель БОТТ (рисунок 13) состоит из следующих элементов:

- блок «Abs» для получения модулей токов основной гармоники токов первичной и вторичной обмоток;
- Блок «Sum» применяется для суммирования модулей токов основной гармоники токов первичной и вторичной обмоток. Блок используется для токов каждой из фаз обмоток силового двухобмоточного трансформатора;
- блок «MinMax» реализует функцию определения максимального значения из тормозных токов по каждой из фаз силового двухобмоточного трансформатора;
- блок «Gain» используется для определения полу суммы модулей фазных токов обмоток каждой из фаз.

3.6 Разработка блока второй ступени дифференциальной защиты двухобмоточного трансформатора

Вторая ступень дифференциальной защиты силового двухобмоточного трансформатора является ступенью с торможением, а алгоритм ее срабатывания определяется тормозной характеристикой дифференциальной защиты.

Вторая ступень позволяет отстроить защиту на блокировку срабатывания от повышения нагрузки, а также позволяет реагировать защите на внутренние повреждения силового двухобмоточного трансформатора которые не сопровождаются высокими значениями токов короткого замыкания.

Тормозная характеристика дифференциальной защиты трансформатора представляется ломаной линией с тремя участками которая делит плоскость возможных значений дифференциального и тормозного токов на зону срабатывания и зону несрабатывания защиты.

Угол наклона каждого участка тормозной характеристики определен в уставках дифференциальной защиты (таблице 2). При этом угол наклона определяется коэффициентом торможения для каждого из участков по выражению:

$$К_{торм.i} = \operatorname{tg} \alpha_i \quad (7)$$

где i – номер участка тормозной характеристики, принимает значения от $i = 1 \dots 3$;

α_i – угол наклона соответствующего i-го участка тормозной характеристики второй ступени дифференциальной защиты.

Для заданных в таблице 2 значений коэффициентов торможения по (7) получим:

– угол наклона второго участка тормозной характеристики:

$$К_{торм.2} = \operatorname{tg} \alpha_2$$

$$\operatorname{tg} \alpha_2 = 0{,}44$$

$$\alpha_2 = \operatorname{arctg}(0{,}44) = 23{,}75\,^\circ$$

– угол наклона третьего участка тормозной характеристики:

$$К_{торм.з} = \operatorname{tg}\alpha_з$$

$$\operatorname{tg}\alpha_з = 0{,}65$$

$$\alpha_з = \operatorname{arctg}(0{,}65) = 33{,}02\,°$$

Первый участок тормозной характеристики соответствует работе защиты без торможения и в целом моделируется блоком первой ступени дифференциальной защиты. Данный участок характеризуется продолжительностью и отсутствием наклона.

Точки перегиба ломаной линии – тормозной характеристики дифференциальной защиты трансформатора определяются соответствующими значениями тормозного и дифференциального токов к номинальному току трансформатора и определяются уставками срабатывания релейной защиты. Для модели дифференциальной защиты силового двухобмоточного трансформатора тормозная характеристика представлена на рисунке 14.

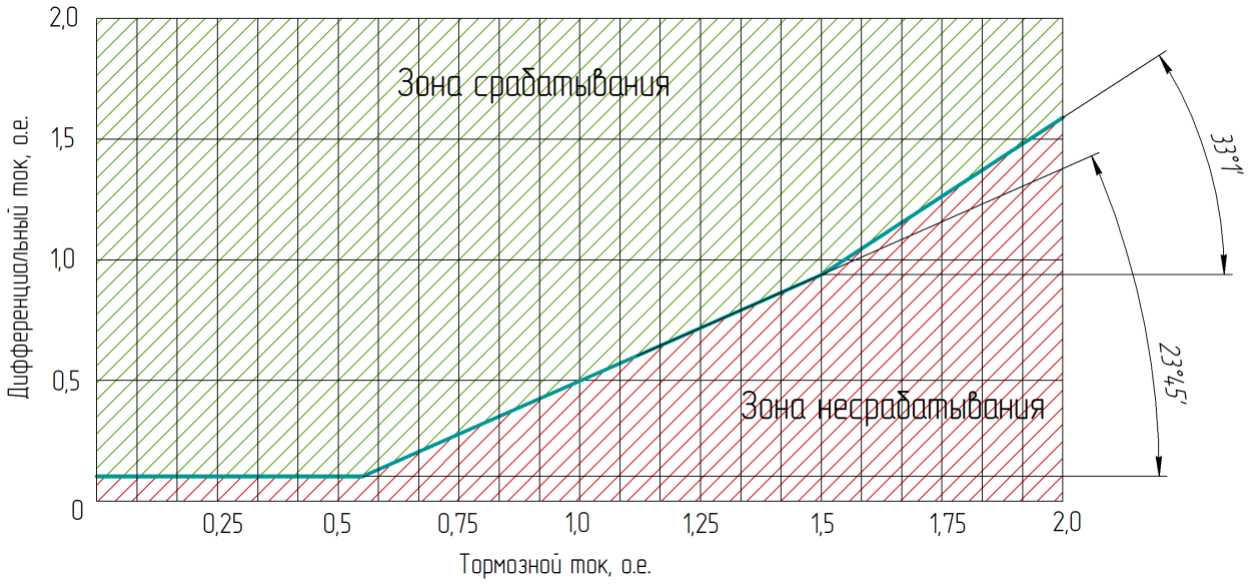

Рисунок 14 - Тормозная характеристика дифференциальной защиты для ТМН-6300/110/10

Алгоритм работы первого участка тормозной характеристики представлен на рисунке 15. В разработанном алгоритме производится

сравнение отношений дифференциального и тормозного токов с уставками с учетом тормозных коэффициентов.

При этом в программном алгоритме тормозной характеристики тормозной коэффициент первого участка – участка без наклона принят равным $К_{торм.1} = 1$. Несмотря на то, что данный коэффициент ($К_{торм.1} = 1$) не попадает под действие выражения (7), он позволяет на снижать дополнительно дифференциальный и тормозной токи, и реагировать дифференциальной защите на повреждения в зоне ее действия. Согласно программному алгоритму (рисунок 15) разработана модель с использованием стандартных элементов MATLAB/Simulink которая представлена на рисунке 16.

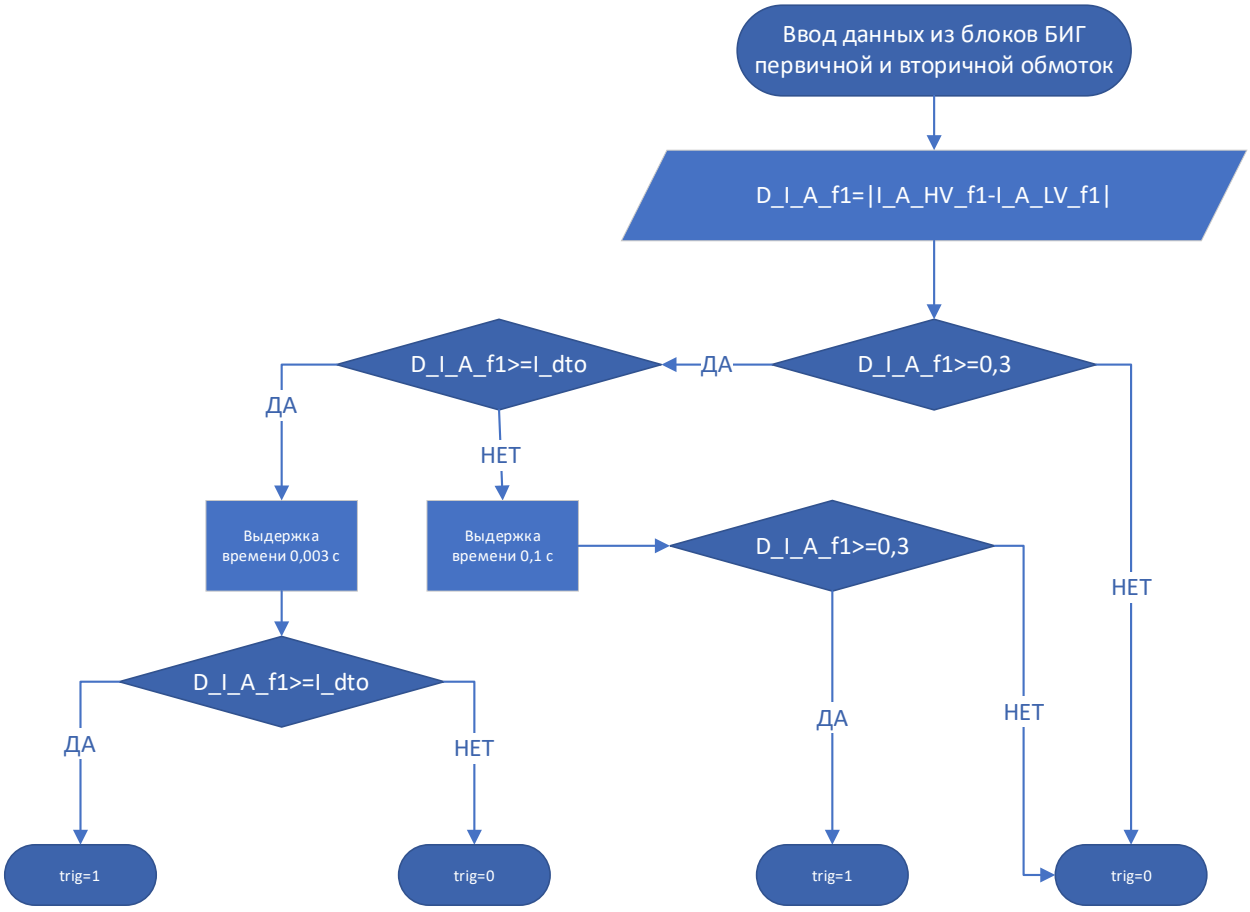

Рисунок 15 - Алгоритм первого участка тормозной характеристики

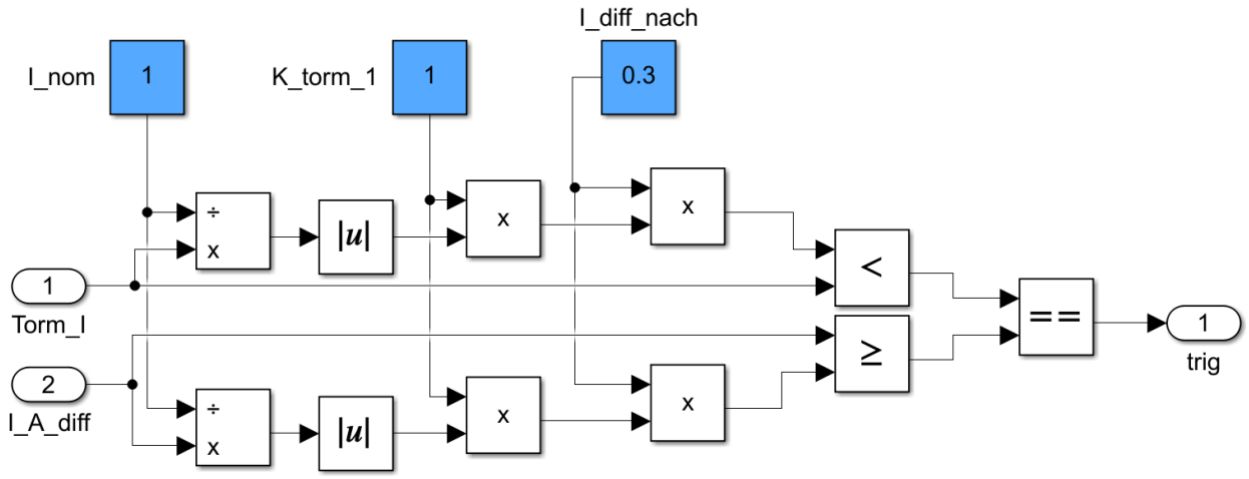

Рисунок 16 - Модель первого участка тормозной характеристики дифференциальной защиты по фазе А

В модели блока первого участка тормозной характеристики использованы:

– блок «Constant» используется для ввода в модель номинального тока трансформатора (I_nom), тормозного коэффициента (K_torm_1), а также уставки начального дифференциального тока (I_diff_nach);
– блок «Divide» используется для определения отношения тормозного тока и дифференциального токов к номинальному току трансформатора;
– блок «Abs» определяет модуль значения отношений тормозного и дифференциального токов к номинальному току трансформатора;
– блок «Product» используется для умножения отношений тормозного и дифференциального токов к номинальному току трансформатора на значение тормозного коэффициента (K_torm1) и уставки дифференциального тока;
– блоки «Relational Operator» используются для определения превышения или снижения значений уставок и определяют срабатывание дифференциальной защиты на первом участке тормозной характеристики.

Алгоритмы работы второго и третьего участка тормозной характеристики аналогичны первому участку, однако в блоках модели второго

и третьего участка изменяются значения уставок дифференциального тока и уставки тормозного коэффициента, которые вводятся в блоки «Constant».

Блоки моделей второго и третьего участков тормозной характеристики, разработанные с использованием элементов MATLAB/Simulink представлены на рисунках 17 и 18 соответственно.

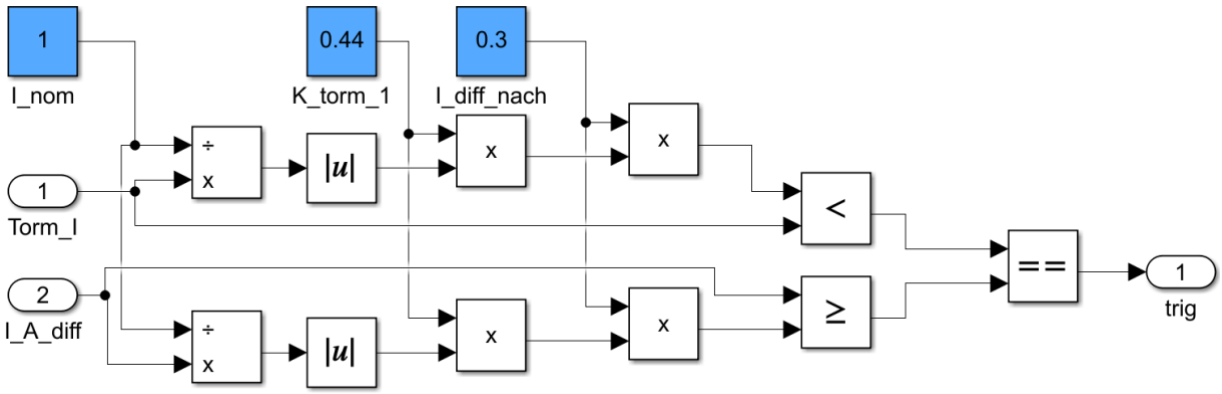

Рисунок 17 - Модель второго (наклонного) участка тормозной характеристики дифференциальной защиты по фазе А

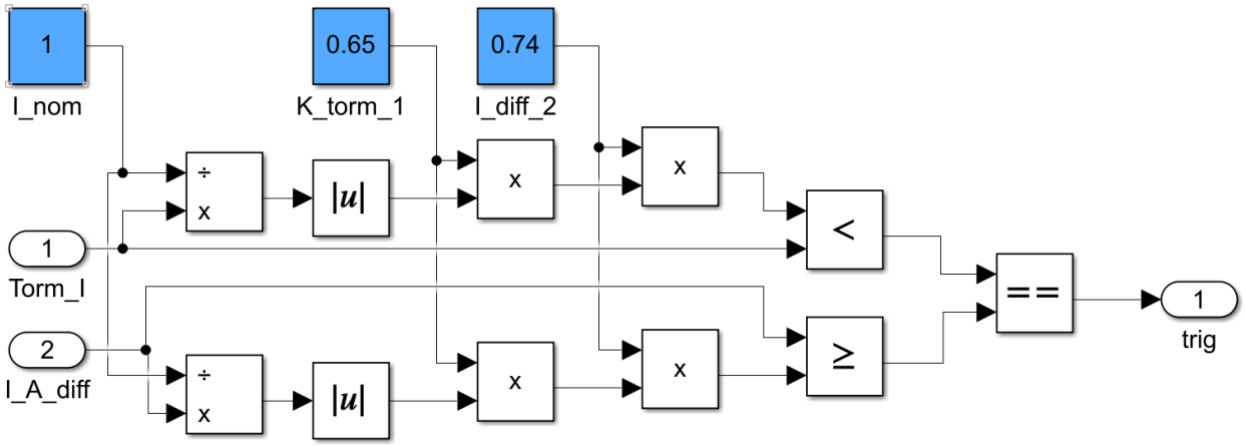

Рисунок 18 - Модель третьего (наклонного) участка тормозной характеристики дифференциальной защиты по фазе А

В результате обработки сигналов в блоках, моделирующих участки тормозной характеристики, выходные сигналы (trig) сравниваются для определения максимального значения. Алгоритм представлен на рисунке 19.

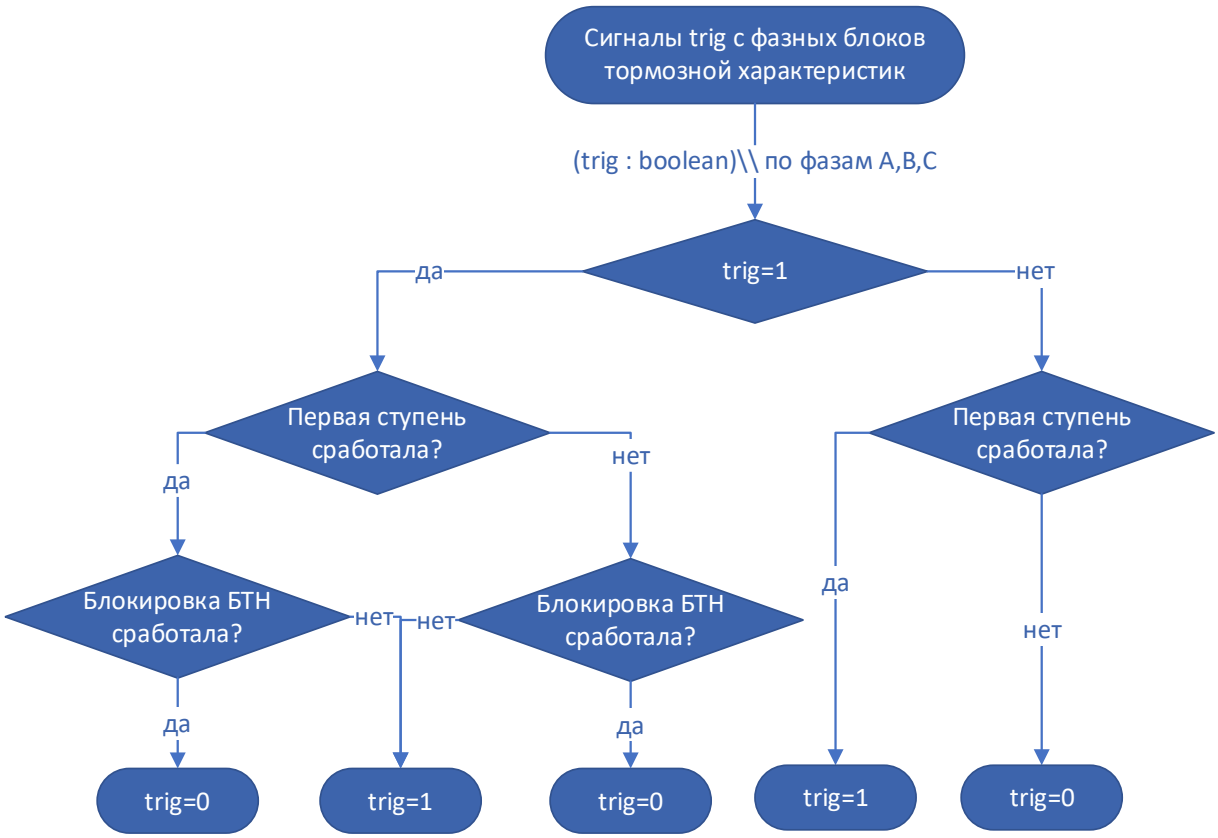

Рисунок 19 - Алгоритм оценки срабатывания ступеней и участков второй ступени дифференциальной защиты

Блоки каждого участка тормозной характеристики выполняются для каждой фазы. Выходные сигналы (trig) с каждого блока участков тормозной характеристики сравниваются между собой с цель. Определения максимального значения. Далее оценивается срабатывание первой ступени и наличие блокировки дифференциальной защиты от бросков тока намагничивания. Результатом является обобщенный сигнал trig который направляется на действие выключателя устанавливаемого на стороне высокого напряжения силового двухобмоточного трансформатора.

3.7 Разработка блока блокировки дифференциальной защиты

При подключении трансформатора к сети, в начальный момент времени возникает бросок тока намагничивания, который может привести к

срабатыванию дифференциальной защиты трансформатора. При этом в трансформаторе может не возникнуть никаких повреждений.

Вывод работы защиты на момент включения трансформатора не допускается так как в момент включения могут возникнуть внутренние повреждения трансформатора и может произойти аварийная ситуация. Для того чтобы повысить надежность и эффективность работы дифференциальной защиты необходимо вводить блокировку от броска тока намагничивания.

Блокировка срабатывает при превышении второй и/или пятой гармоник тока над основной гармоникой тока и блокирует действие всех степеней защиты. Однако блокировка по второй и/или пятой гармоникам тока не может действовать продолжительное время и должна быть отстроена от длительности броска тока намагничивания трансформатора.

Длительность и амплитуда броска тока намагничивания определяется моментом включения трансформатора, а также его мощностью. В [25] представлена зависимость броска тока намагничивания от мощности трансформатора. Для силового трансформатора ТМН-6300/110/10 кВ во втором разделе выпускной квалификационной работы выдержка времени для блокировки от броска тока намагничивания $t_{\text{гарм.2}} = 1$ (с). При этом уставка блокировки определена только для второй гармоники тока на уровне 15%. Алгоритм работы блокировки от броска тока намагничивания представлен на рисунке 20.

Принципиально, гармоническое разложение тока для выявления присутствия процесса насыщения силового трансформатора возможно выполнить только со стороны питания, в случае рассмотренном во втором разделе выпускной квалификационной работы (рисунок 4) питание приходит со стороны обмотки высокого напряжения.

Однако для повышения универсальности работы разрабатываемой модели дифференциальной защиты выполним учет возможности блокировки как стороны первичной, так и со стороны вторичной обмоток силового трансформатора – это позволит использовать разработанную модель

дифференциальной защиты в системах электроснабжения с двухсторонним питанием.

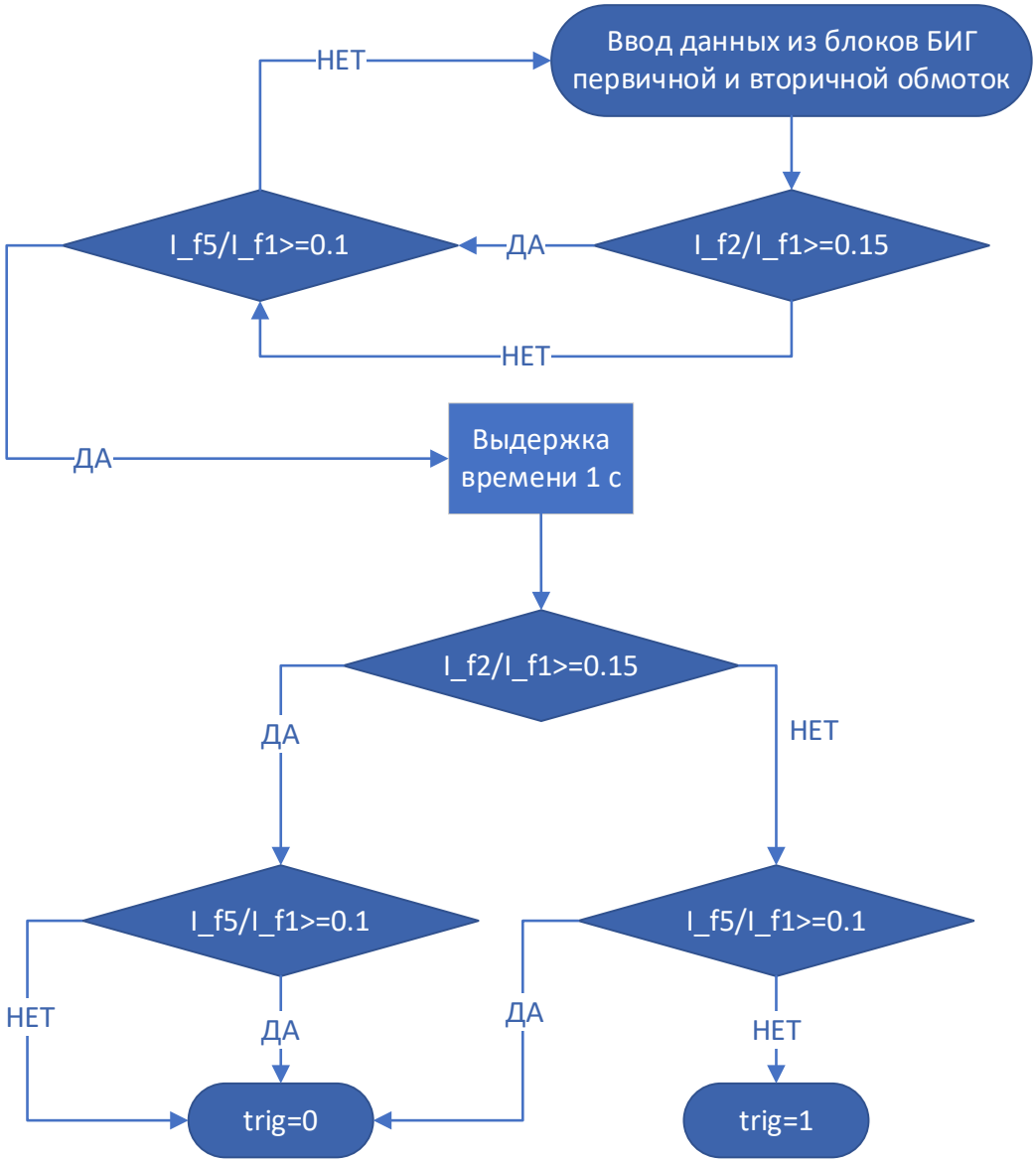

Рисунок 20 - Алгоритм работы блокировки от броска тока намагничивания

Разработанная модель блока определения блокировки от броска тока намагничивания для первичной обмотки фазы А представлена на рисунке 21.

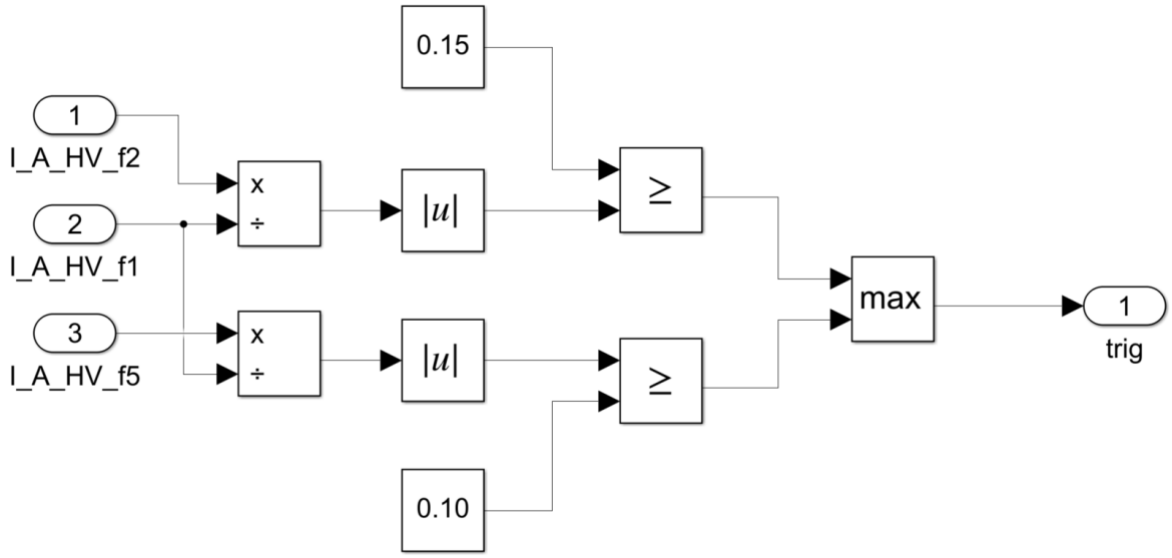

Рисунок 21 - Модель блокировки при броске тока намагничивания по фазе А со стороны обмотки высокого напряжения

Модель блокировки при броске тока намагничивания состоит из:

- блоков «Divide» определяющих отношение второй и пятой гармоник к основного гармонике тока;
- блоки «Abs» определяют модуль отношений второй гармоники тока к первой и пятой гармонике тока к пятой гармонике;
- блоки «Constant» задают уставки соответствующие броску тока намагничивания;
- блок «MinMax» определяет какая из блокировок – по второй или/или пятой гармонике тока сработала и формирует выходной логический сигнал – trig.

3.8 Компоновка модели дифференциальной защиты

Все разработанные в подразделах 3.1 – 3.7 блоки необходимо скомпоновать в общую модель дифференциальной защиты силового двухобмоточного трансформатора согласно общей принципиальной логике ее работы. Блок схема скомпонованной модели дифференциальной защиты силового двухобмоточного трансформатора представлена на рисунке 22.

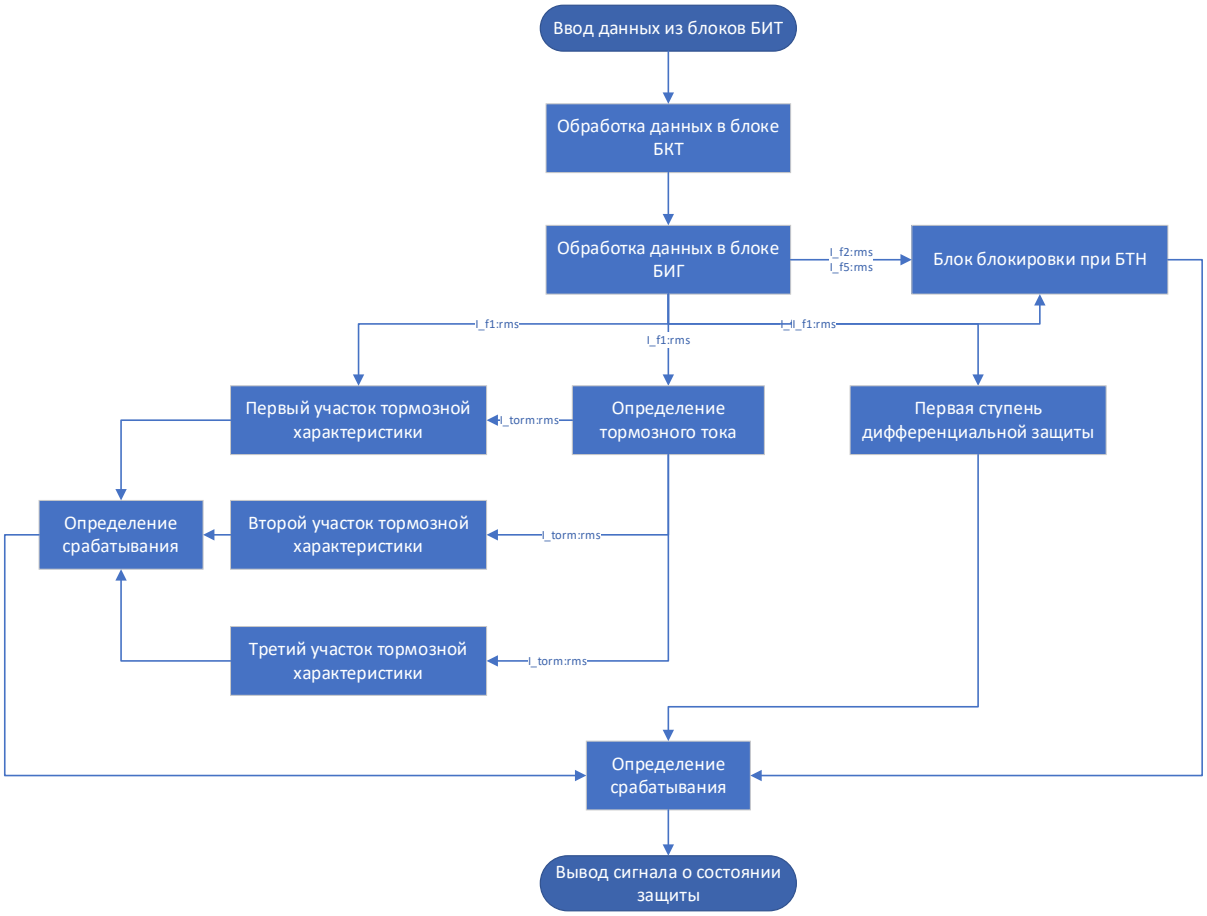

Рисунок 22 - Обобщенная блок схема модели дифференциальной защиты

Общий вид цифровой модели дифференциальной защиты двухобмоточного трансформатора, разработанный с учетом блоков представленных в разделах 3.1 – 3.7 выпускной квалификационной работы представлен на рисунке 23.

Дополнительно для проведения тестирования и верификации разработанной цифровой модели дифференциальной защиты силового двухобмоточного трансформатора составлен программный код на языке «Си».

В приложении А представлен программный код модели дифференциальной защиты двухобмоточного трансформатора выполненный на языке «Си».

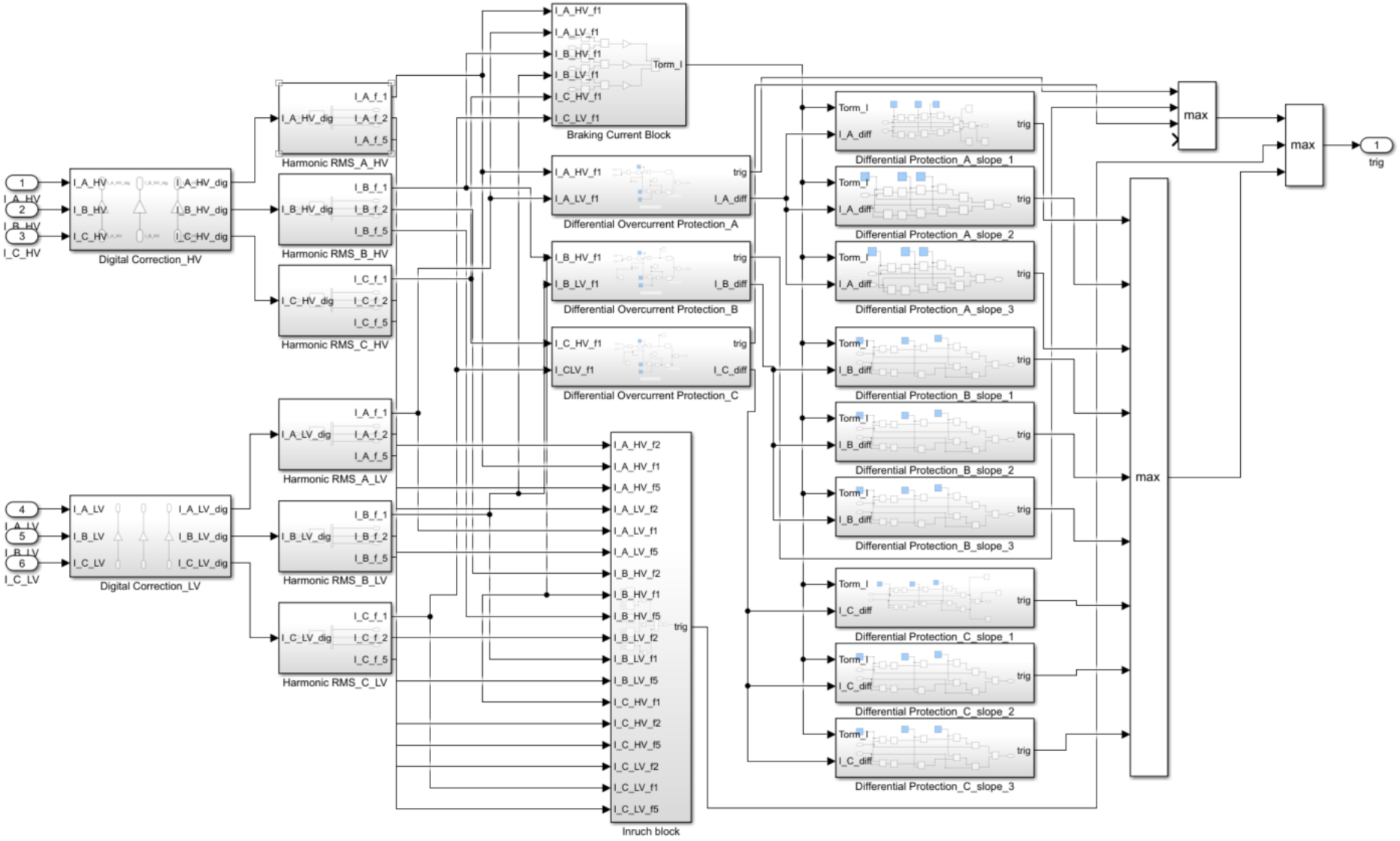

Рисунок 23 - Полная модель цифровой дифференциальной защиты силового двухобмоточного трансформатора

Выводы по разделу.

Разработаны блоки модели дифференциальной защиты силового двухобмоточного трансформатора.

Разработан блок измерения токов, для получения мгновенных значений токов приведенных к вторичным токам измерительных трансформаторов. Для корректировки токов разработан блок позволяющий снизить отклонение токов, тем самым повысив точность срабатывания дифференциальной защиты.

Разработан блок определения уровня первой, второй и пятой гармоник тока. Первая гармоника тока предназначена для функционирования первой и второй ступеней дифференциальной защиты, а по значениям второй и пятой гармоник тока определяется присутствие режима насыщения трансформатора, в том числе в момент его включения.

Разработан блок первой ступени дифференциальной защиты, моделирующий работу дифференциальной отсечки с учетом корректировки по току небаланса.

Вторая ступень дифференциальной защиты реализована тремя блоками, моделирующими первый, второй и третий участок тормозной характеристики с учетом тормозного тока определяемого в блоке расчета тормозного тока.

При срабатывании одной из степеней на вывод подается сигнал 1, при отсутствии сигнала 1 от блока блокировки от БТН. Если сигнал от защиты и от блока определения насыщения не совпадают, то происходит срабатывание защиты.

4 Верификация алгоритмов работы и модели цифровой дифференциальной защиты трансформатора

Для выполнения тестирования разработанных программных алгоритмов цифровой дифференциальной защиты силового двухобмоточного трансформатора необходимо смоделировать участок сети с использованием моделей элементов электроэнергетической системы из библиотеки элементов MATLAB/Simulink [23]. Однолинейная схема тестовой модели, а также ее реализация в MATLAB/Simulink представлены на рисунке 24.

Тестовая модель состоит из следующих элементов (рисунок 24) [22]:

- внешняя электроэнергетическая система (External Power Grid) – моделируется блоком Simplified Synchronous Machine PU из библиотеки элементов MATLAB/Simulink;
- измерительный трансформатор тока на напряжение 110 кВ (HV_Current transformer) – моделируется тремя блоками Current Measurement и тремя блоками Gain из библиотеки элементов MATLAB/Simulink;
- трехфазный управляемый выключатель на напряжение 110 кВ (HV_Switchgear) – моделируется стандартным блоком Breaker из библиотеки элементов MATLAB/Simulink;
- трехфазный двухобмоточный силовой трансформатор (Power Transformer) – моделируется стандартным блоком Three-Phase Transformer из библиотеки элементов MATLAB/Simulink;
- измерительный трансформатор тока на напряжение 10 кВ (LV_Current transformer) – моделируется тремя блоками Current Measurement и тремя блоками Gain из библиотеки элементов MATLAB/Simulink;

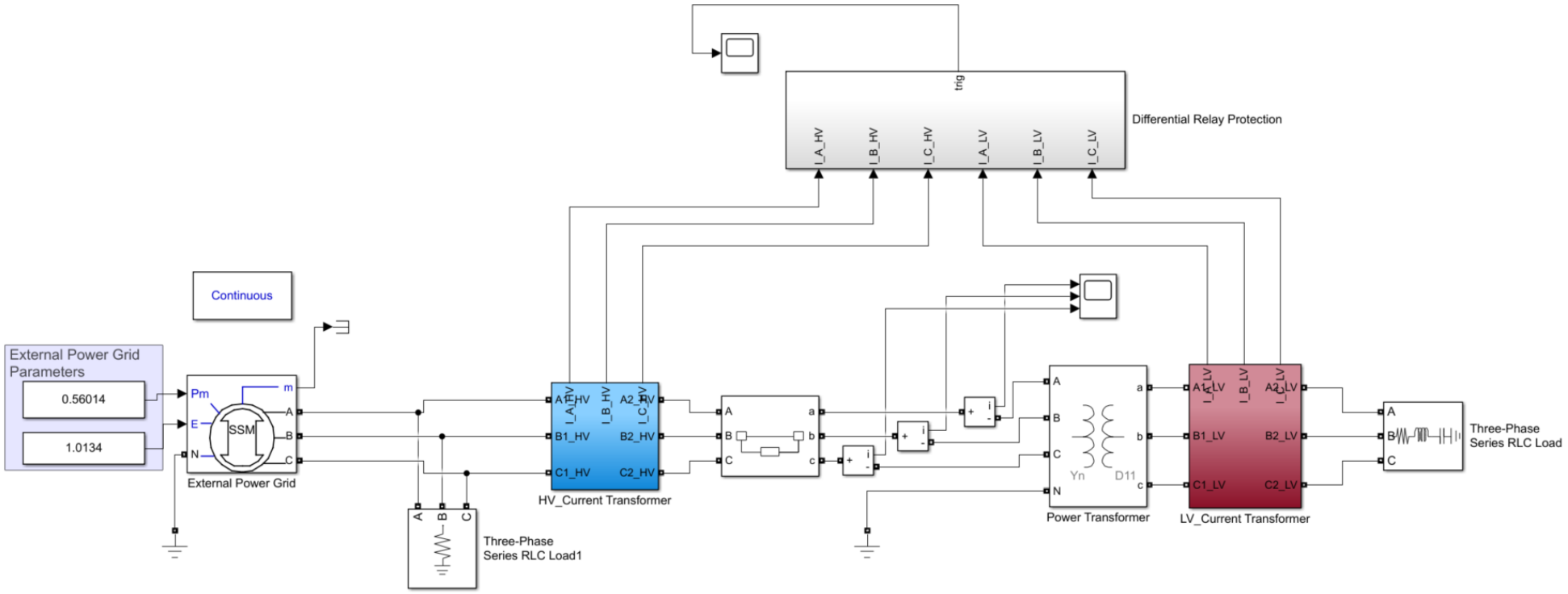

Рисунок 24 - Схема модели для верификации программных алгоритмов цифровой дифференциальной защиты трансформатора

– потребитель электрической энергии на напряжении 10 кВ – моделируется блоком трехфазной комплексной нагрузки Three-Phase Series RLC Load из библиотеки элементов MATLAB/Simulink.

Для всех блоков тестовой модели необходимо выполнить расчет параметров моделей. Расчет параметров моделей элементов тестовой схемы выполняется на основании паспортных характеристик используемого оборудования.

Выполним расчет параметров модели блока моделирующего двухобмоточный трехфазный трансформатор.

Выполним расчет параметров модели двухобмоточного трехстержневого трансформатора марки ТМН-6300/110/10 кВ используя паспортные данные, представленные в таблице 3.

Номинальный ток:

$$I_{\text{транс.ном}} = \frac{S_{\text{ном.транс}}}{\sqrt{3} \cdot U_{\text{ном.ВН}}} = \frac{6300}{\sqrt{3} \cdot 115} = 31{,}63 \text{ (А)}$$

Фазное напряжение первичной обмотки:

$$U_{\text{ф.ВН}} = \frac{U_{\text{ном.ВН}}}{\sqrt{3}} = \frac{110}{\sqrt{3}} = 6{,}64 \cdot 10^4 \text{ (В)}$$

Фазное напряжение вторичной обмотки:

$$U_{\text{ф.НН}} = \frac{U_{\text{ном.НН}}}{\sqrt{3}} = \frac{11}{\sqrt{3}} = 6{,}35 \cdot 10^3 \text{ (В)}$$

Фазный ток холостого хода:

$$I_{\text{ф.хх}} = \frac{I_{\text{транс.ном}} \cdot i_{\text{хх,\%}}}{100} = \frac{31{,}63 \cdot 0{,}56}{100} = 0{,}18 \text{ (А)}$$

Фазные потери холостого хода:

$$P_{\text{ф.хх}} = \frac{P_{\text{хх}}}{\sqrt{3}} = \frac{6{,}5}{\sqrt{3}} = 3753 \text{ (Вт)}$$

Активное сопротивление ветви намагничивания:

$$R_m = \frac{U_{\text{ф.ВН}}^2}{P_{\text{ф.хх}}} = \frac{(6{,}64 \cdot 10^4)^2}{3{,}753} = 1{,}17 \cdot 10^6 \text{ (Ом)}$$

Индуктивное сопротивление ветви намагничивания:

$$X_m = \frac{1}{\sqrt{\left(\frac{I_{\text{ф.хх}}}{U_{\text{ф.ВН}}}\right)^2 - \left(\frac{1}{R_m}\right)^2}} = \frac{1}{\sqrt{\left(\frac{0{,}185}{63{,}51}\right)^2 - \left(\frac{1}{1{,}17 \cdot 10^6}\right)^2}} = 3{,}96 \cdot 10^5 \text{ (Ом)}$$

Индуктивность ветви намагничивания:

$$L_m = \frac{X_m}{2 \cdot \pi \cdot f} = \frac{3{,}96 \cdot 10^5}{2 \cdot \pi \cdot 50} = 1{,}26 \cdot 10^3 \text{ (Гн)}$$

Коэффициент трансформации:

$$K_{\text{Т}} = \frac{U_{\text{ф.ВН}}}{U_{\text{ф.НН}}} = \frac{6{,}64 \cdot 10^4}{6{,}35 \cdot 10^3} = 10{,}45$$

Фазное напряжение короткого замыкания:

$$U_{\text{ф.кз}} = \frac{U_{\text{ф.ВН}} \cdot u_{\text{кз.\%}}}{100} = \frac{6{,}64 \cdot 10^4 \cdot 10{,}5}{100} = 6{,}97 \cdot 10^3 \text{ (В)}$$

Полное сопротивление короткого замыкания:

$$Z_{\text{кз}} = \frac{U_{\text{ф.кз}}}{I_{\text{транс.ном}}} = \frac{6{,}97 \cdot 10^3}{31{,}63} = 2{,}22 \cdot 10^2 \text{ (Ом)}$$

Фазные потери короткого замыкания:

$$P_{\text{ф.кз}} = \frac{1}{3} P_{\text{кз}} = \frac{1}{3} \cdot 35 = 1{,}17 \cdot 10^4 \text{ (Вт)}$$

Активное сопротивление короткого замыкания:

$$R_{\text{кз}} = \frac{P_{\text{ф.кз}}}{I^2_{\text{транс.ном}}} = \frac{1{,}17 \cdot 10^4}{31{,}63^2} = 11{,}66 \text{ (Ом)}$$

Индуктивное сопротивление короткого замыкания:

$$X_{\text{кз}} = \sqrt{Z^2_{\text{кз}} - R^2_{\text{кз}}} = \sqrt{(2{,}22 \cdot 10^2)^2 - 11{,}66^2} = 2{,}2 \cdot 10^2 \text{ (Ом)}$$

Активное сопротивление первичной обмотки:

$$R_1 = 0{,}5 \cdot R_{\text{кз}} = 0{,}5 \cdot 11{,}66 = 5{,}83 \text{ (Ом)}$$

Индуктивное сопротивление первичной обмотки:

$$X_1 = 0{,}5 \cdot X_{\text{кз}} = 0{,}5 \cdot 2{,}2 \cdot 10^2 = 1{,}1 \cdot 10^2 \text{ (Ом)}$$

Индуктивность первичной обмотки:

$$L_1 = \frac{X_1}{2 \cdot \pi \cdot f} = \frac{1{,}1 \cdot 10^2}{2 \cdot \pi \cdot 50} = 0{,}35 \text{ (Гн)}$$

Активное сопротивление вторичной обмотки:

$$R_2 = \frac{R_1}{K_{\text{Т}}^2} = \frac{5{,}83}{10{,}45^2} = 5{,}34 \cdot 10^{-2}\ (\text{Ом})$$

Индуктивное сопротивление вторичной обмотки:

$$X_2 = \frac{X_1}{K_{\text{Т}}^2} = \frac{1{,}1 \cdot 10^2}{10{,}45^2} = 1{,}01\ (\text{Ом})$$

Индуктивность вторичной обмотки:

$$L_2 = \frac{X_2}{2 \cdot \pi \cdot f} = \frac{1{,}01}{2 \cdot \pi \cdot 50} = 3{,}21 \cdot 10^{-3}\ (\text{Гн})$$

Индуктивность нулевой последовательности замыкания магнитного потока:

$$L_0 = \frac{L_m}{1000} = \frac{1{,}26 \cdot 10^3}{1000} = 1{,}26\ (\text{Гн})$$

Результаты расчета параметров блока модели необходимо занести в блок модели двухобмоточного силового трансформатора в MATLAB/Simulink.

Для верификации и тестирования разработанной модели дифференциальной защиты силового трансформатора проведем тестирование по следующим вариантам:

– проверка нормального функционирования разработанного блока дифференциальной защиты силового двухобмоточного трансформатора. Данный вариант тестирования позволит определить отсутствие срабатывания защиты в нормальном режиме, а также проверить наличие блокировки по броску тока намагничивания;

– проверка срабатывания защиты при трехфазном коротком замыкании в зоне действия защиты;
– проверка несрабатывания защиты при коротком замыкании вне зоны действия защиты.

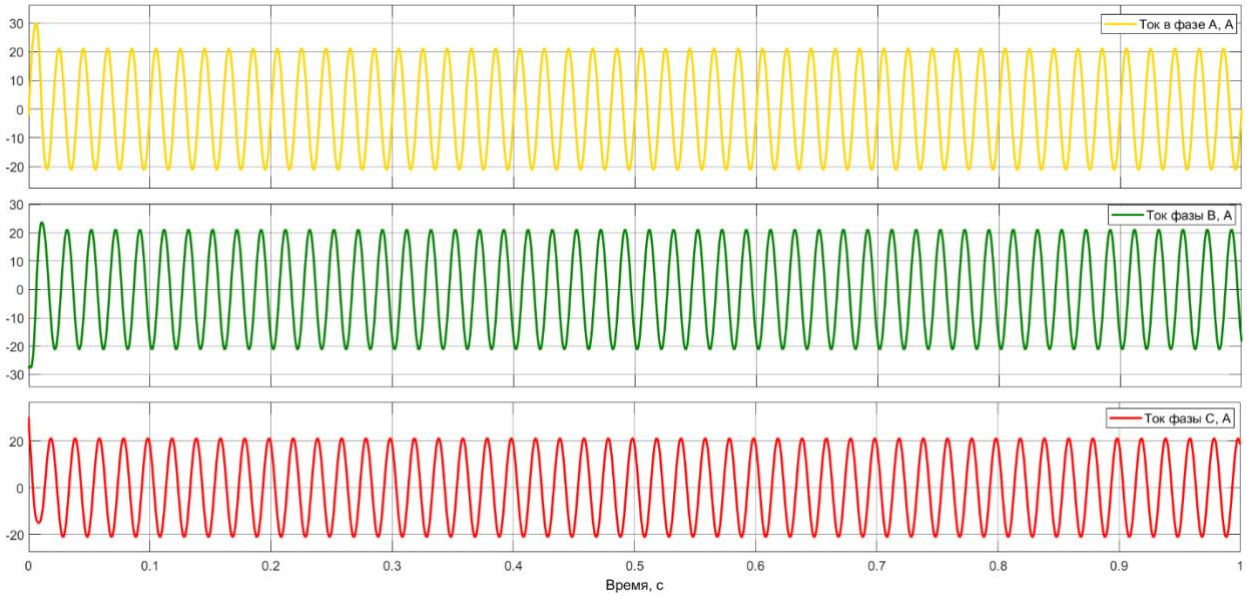

Рисунок 25 - Мгновенные значения токов со стороны 110 кВ трансформатора в установившемся режиме

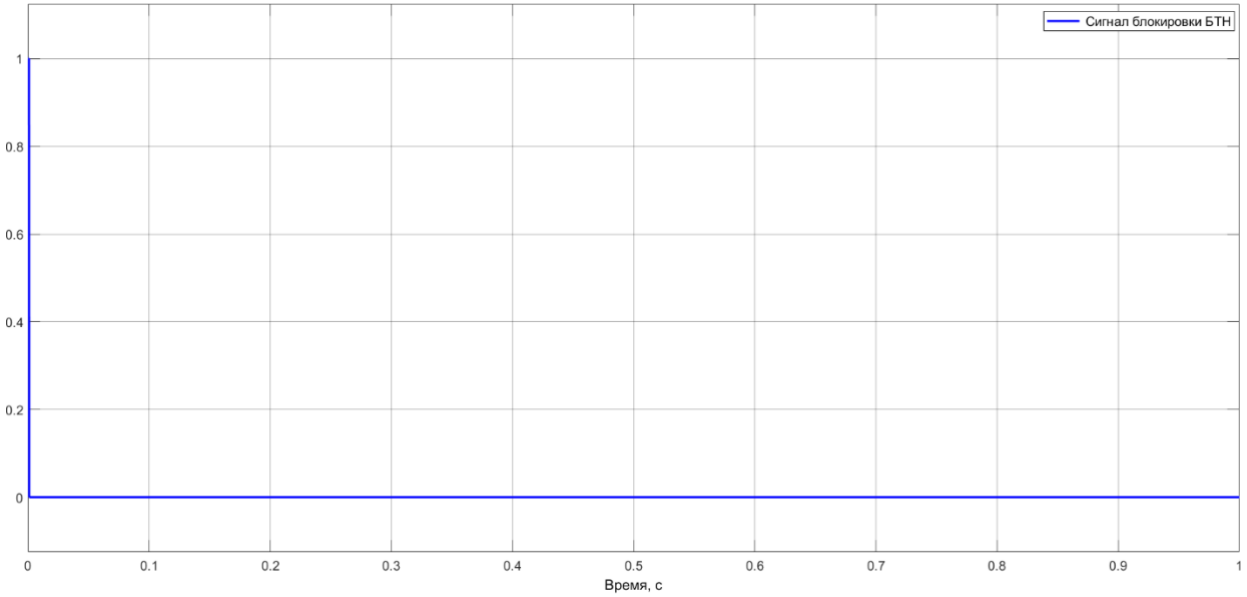

Рисунок 26 - Сигнал блокировки от БТН в установившемся режиме

Из рисунков 25 и 26 видно, что в отсутствии действия коротких замыкания модель дифференциальной защиты отрабатывает в штатном режиме, т.е. защита не срабатывает, при защита блокируется, так как в начальный момент модельного времени, момент включения, происходит небольшой бросок тока, продолжительность действия блокировки $1 \cdot 10^{-3}$ с.

При моделировании короткого замыкания в зоне действия дифференциальной защиты, сигнал положения выключателя записывался отдельно от действия выключателя, выключатель в модели не выполнял отключение в момент действия релейной защиты, для повышения скорости расчета модели и снижения влияния коммутационных перенапряжений.

Осциллограммы мгновенных значений тока при моделировании короткого замыкания в зоне действия защиты показано на рисунке 27. Короткое замыкание моделируется в 0,5 с модельного времени.

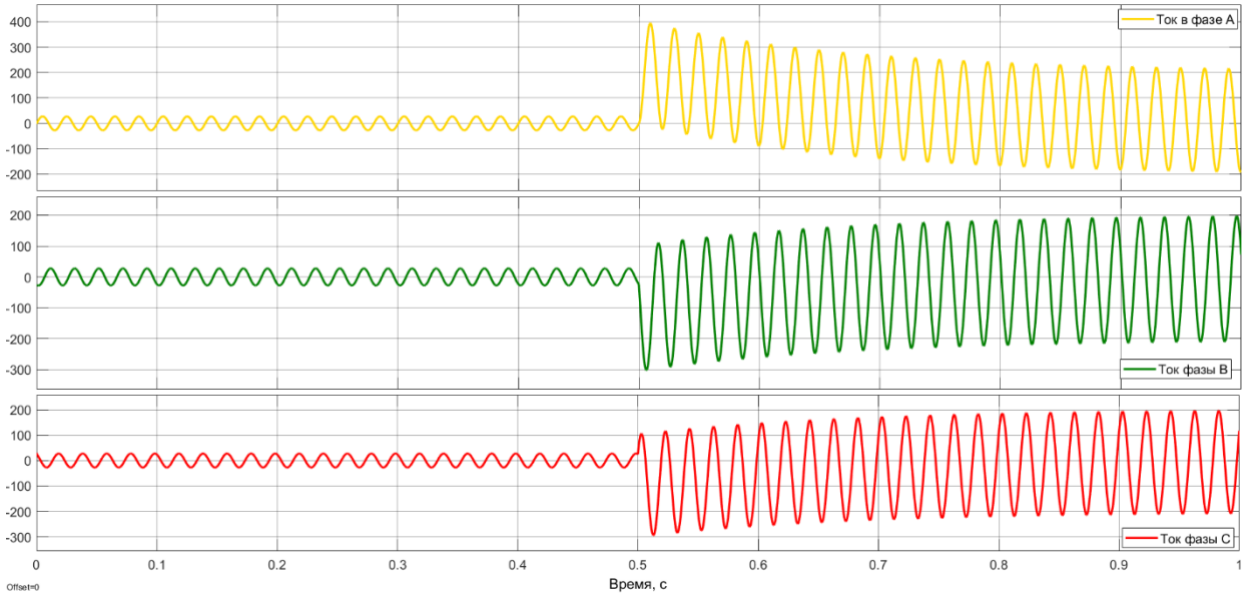

Рисунок 27 - Мгновенные значения токов со стороны 110 кВ трансформатора при моделировании короткого замыкания в зоне действия защиты

Логический сигнал, отражающий положение выключателя, а следовательно, и срабатывания защиты показан на рисунке 28. Значение сигнала 1 соответствует замкнутым контактам выключателя, состояние 0 разомкнутым контактам выключателя.

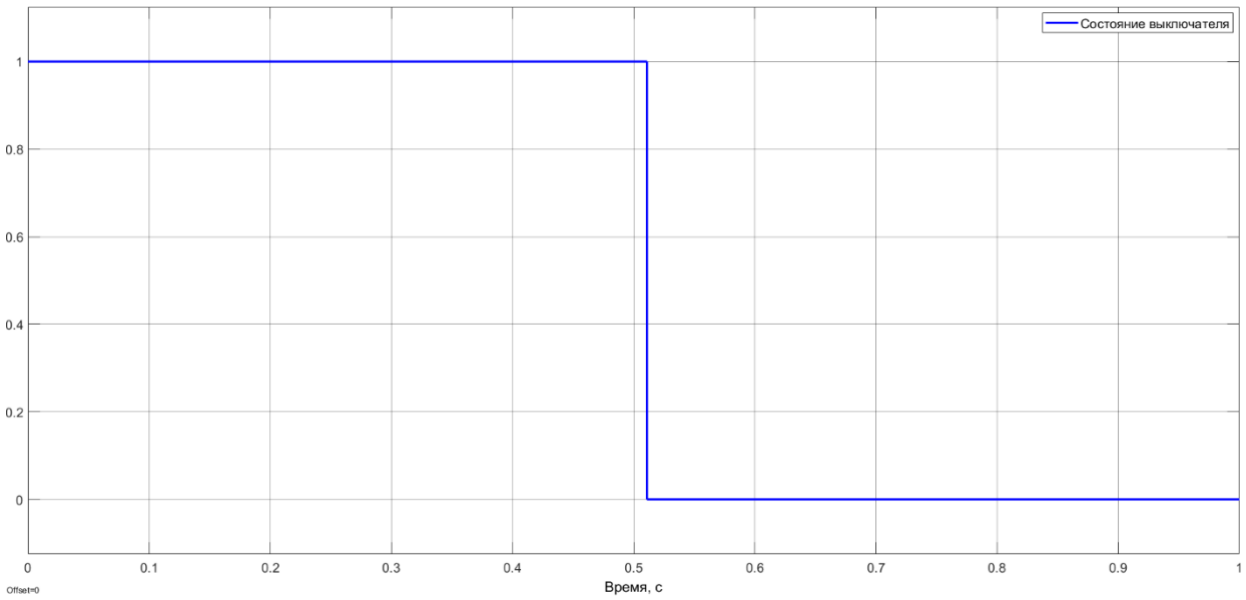

Рисунок 28 – Сигнал состояния выключателя при моделировании короткого замыкания в зоне действия защиты

Согласно графику, представленному на рисунке 28 отключение производится в 0,511 с, т.е. через 0,011 с после возникновения короткого замыкания, что говорит о срабатывании дифференциальной токовой отсечки.

На рисунке 29 показаны осциллограммы мгновенных значений токов при моделировании короткого замыкания вне зоны действия дифференциальной защиты.

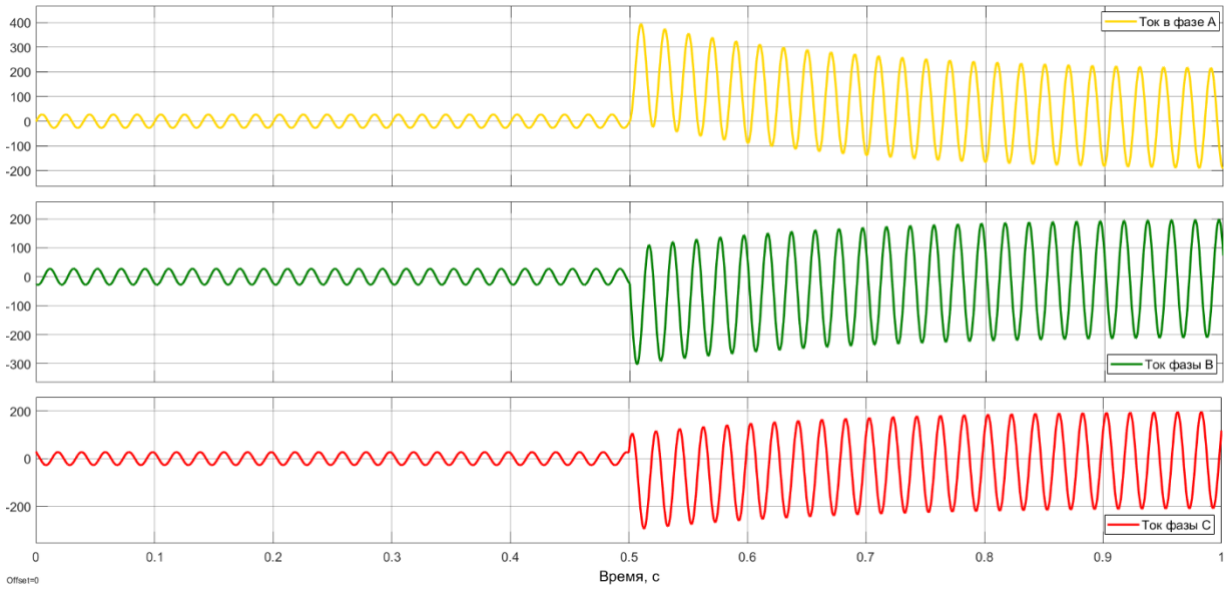

Рисунок 29 - Мгновенные значения токов со стороны 110 кВ трансформатора при моделировании короткого замыкания вне зоны действия защиты

На рисунке 30 показан сигнал состояния выключателя при моделировании короткого замыкания вне зоны действия дифференциальной защиты силового трансформатора.

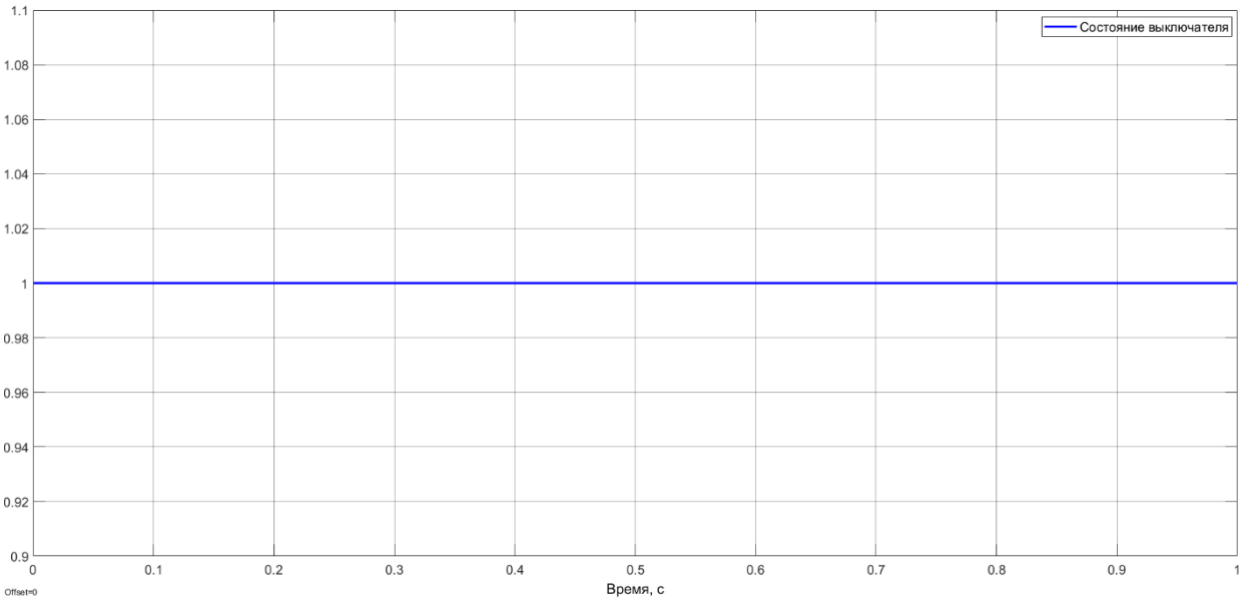

Рисунок 30 – Сигнал состояния выключателя при моделировании короткого замыкания в зоне действия защиты

Как видно из графика состояния выключателя (рисунок 30) короткое замыкание не вызывает отключение выключателя, что говорит об адекватности разработанной цифровой модели дифференциальной защиты силового трансформатора.

Выводы по разделу.

Для верификации и тестирования цифровой модели дифференциальной защиты силового трансформатора разработана тестовая схема, состоящая из источника питания, выключателя, трансформатора и нагрузки.

Для верификации алгоритмов работы выполнено тестирование в трех режимах [24]:

– нормальный устранившийся режим без короткого замыкания с подключением трансформатора к сети. В рамках данного варианта тестирования выполнена проверка срабатывания блокировки при

броске тока намагничивания, а также не срабатывание защиты в нормальном режиме;

- аварийный режим с трехфазным коротким замыканием в зоне действия дифференциальной защиты. Данный режим показал срабатывание дифференциальной токовой отсечки за 0,011 с от момента возникновения короткого замыкания (момент времени 0,5 с);
- аварийный режим с трехфазным коротким замыканием вне зоны действия дифференциальной защиты. Данный режим показал, что при возникновении короткого замыкания в момент времени 0,5 с дифференциальная защита не срабатывает и положение выключателя остается ез изменений (сигнал равен 1).

Проведенное тестирование разработанной модели показала адекватность работы алгоритмов. Разработанная модель дифференциальной защиты может быть использована в динамических моделях систем электроснабжения, а также при изучении принципов работы дифференциальной защиты в образовательном процессе.

Заключение

В выпускной квалификационной работе разработана цифровая модель дифференциальной защиты силового двухобмоточного трансформатора.

Во введение определена актуальность выбранной темы, поставлена цель и задачи для ее достижения, а также представлены ссылки на научные работы, выполненные на смежную тематику.

В первом разделе рассмотрены и проанализированы особенности функционирования дифференциальной защиты силового двухобмоточного трансформатора. Рассмотрены особенности действия аналоговой защиты и современной микропроцессорной (цифровой) защиты, Рассмотрены и проанализированы схемы корректировки значений тока в микропроцессорном устройстве дифференциальной защиты для обеспечения правильного срабатывания защиты при внутренних коротких замыканиях и блокировки срабатывания при внешних коротких замыканиях. Представлено описание алгоритма действия цифровой дифференциальной защиты силового трансформатора, который будет использован при разработке цифровой модели дифференциальной защиты силового двухобмоточного трансформатора в рамках достижения поставленной цели.

Во втором разделе, для разработки, тестирования и верификации модели цифровой дифференциальной защиты силового двухобмоточного трансформатора выбран трансформатор марки ТМН-6300/110/10 кВ. Для выбранного трансформатора определены уставки микропроцессорного терминала БМРЗ-ТД. Уставки микропроцессорного терминала БМРЗ-ТД производства ООО «НТЦ «Механотроника» использованы при разработке модели цифровой дифференциальной защиты силового трансформатора.

Третий раздел посвящен разработке и описанию алгоритмов и блоков цифровой модели дифференциальной защиты силового двухобмоточного трансформатора.

Разработан блок измерения токов, для получения мгновенных значений токов приведенных к вторичным токам измерительных трансформаторов. Для корректировки токов разработан блок позволяющий снизить отклонение токов, тем самым повысив точность срабатывания дифференциальной защиты.

Разработан блок определения уровня первой, второй и пятой гармоник тока. Первая гармоника тока предназначена для функционирования первой и второй ступеней дифференциальной защиты, а по значениям второй и пятой гармоник тока определяется присутствие режима насыщения трансформатора, в том числе в момент его включения.

Разработан блок первой ступени дифференциальной защиты, моделирующий работу дифференциальной отсечки с учетом корректировки по току небаланса.

Вторая ступень дифференциальной защиты реализована тремя блоками, моделирующими первый, второй и третий участок тормозной характеристики с учетом тормозного тока определяемого в блоке расчета тормозного тока.

При срабатывании одной из степеней на вывод подается сигнал 1, при отсутствии сигнала 1 от блока блокировки от БТН. Если сигнал от защиты и от блока определения насыщения не совпадают, то происходит срабатывание защиты.

В четвертом разделе выполнена верификация и тестирование цифровой модели дифференциальной защиты силового двухобмоточного трансформатора. Для верификации и тестирования цифровой модели разработана тестовая схема, состоящая из источника питания, выключателя, трансформаторов тока, силового трансформатора и нагрузки.

Для верификации алгоритмов работы выполнено тестирование в трех режимах:

- нормальный устранившийся режим без короткого замыкания с подключением трансформатора к сети. В рамках данного варианта тестирования выполнена проверка срабатывания блокировки при

броске тока намагничивания, а также не срабатывание защиты в нормальном режиме;

- аварийный режим с трехфазным коротким замыканием в зоне действия дифференциальной защиты. Данный режим показал срабатывание дифференциальной токовой отсечки за 0,011 с от момента возникновения короткого замыкания (момент времени 0,5 с);
- аварийный режим с трехфазным коротким замыканием вне зоны действия дифференциальной защиты. Данный режим показал, что при возникновении короткого замыкания в момент времени 0,5 с дифференциальная защита не срабатывает и положение выключателя остается ез изменений (сигнал равен 1).

Проведенное тестирование разработанной цифровой модели показало адекватность работы алгоритмов и цифровой модели в целом. Разработанная модель цифровой дифференциальной защиты может быть использована при динамическом моделировании режимов работы систем электроснабжения, а также при изучении принципов работы дифференциальной защиты в образовательном процессе.

В результате все поставленные задачи выполнены, цель выпускной квалификационной работы достигнута.

Список используемой литературы и используемых источников

1 Андреев М.В., Рубан Н.Ю., Суворов А.А. Математическое моделирование цифровой дифференциальной защиты трансформатора в среде MATLAB Simulink // Вестник Иркутского государственного технического университета. 2018. Т. 1. № 22. С. 134-150.

2 Антишкин А.Б. Адаптивные модификации алгоритма дифференциальной защиты трансформатора, Чувашский государственный университет имени И.Н. Ульянова, Чебоксары, Диссертация на соискание ученой степени кандидата технических наук 2018.

3 Дмитренко А.М., Журавлев Д.П. Анализ и исследование способов торможения цифровых дифференциальных защит блочных трансформаторов // Электрические станции, Т. 11, 2014. С. 36-41.

4 Дьяконов В.П. Математическое моделирование. 2-е изд. М.: ДМК Пресс, 2023. 783 с.

5 Евминов Л.И., Селиверстов Г.И. Релейная защита и автоматика электроэнергетических систем. Гомель: ГГТУим. П. О. Сухого, 2016. 531 с.

6 Ершов Ю.А., Малеев А.В. Моделирование микропроцессорных релейных защит в среде MATLAB // Журнал Сибирского федерального университета. Техника и технологии., Т. 3, № 2, 2010. С. 220-228.

7 Каппес А.Д., Апросин К.И. Алгоритмы защиты трансформатора для выявления внутренних коротких замыканий // Вестник ЮУрГУ. Серия "Энергетика", Т. 2, 2023. С. 18-29.

8 Киреева Э.А., Цырук С.А. Релейная защита и автоматика электроэнергетических систем. М.: Академия, 2010. 288 с.

9 ОАО «ФСК ЕЭС». СТО 56947007-25.040.70.101-2011 Правила графического отображения информации посредством ПТК и АСУ ТП (с изменениями от 18.07.2017) // Группа "Россети". 2011. URL: https://www.rosseti.ru/upload/iblock/0ae/9zbsr7sol90jcheztzqwxtb7ru8kmpk1.pdf (дата обращения: 03.03.2025).

10 ООО «НТЦ «Механотроника». Методические указания по расчету уставок и заданию параметров дифференциальной защиты с торможением 6 - 220 кВ // Официальный сайт ООО «НТЦ «Механотроника». 2024. URL: https://www.mtrele.ru/files/project/my_po_vybory_ystavok_dzt_dzo_dzsh_tznpver%203.10.pdf (дата обращения: 03.03.2025).

11 ООО НПП «ЭКРА». Терминал основных и резервных защит, автоматики, управления и сигнализации двухобмоточного трансформатора // Официальный сайт ООО НПП «ЭКРА». 2024. URL: https://ekra.ru/product/docs/rza-ps-6-35/t-rza/be2502a/%D0%A0%D0%AD%20%D0%BD%D0%B0%20%D0%91%D0%AD2502%D0%901804.pdf (дата обращения: 03.03.2025).

12 Пусенкова О.Е., Андреев М.В. Интеллектуальные энергосистемы : труды III Международного молодёжного форума, 28 сентября - 2 октября 2015 г. // Моделирование цифровой дифференциальной защиты силового трансформатора с использованием MATLAB. Томск. 2015. Vol. 3. pp. 82-85.

13 Романюк Ф.А., Ломан М.С. Исследование микропроцессорной дифференциальной защиты понижающего трансформатора // Наука и техника, № 1, 2014. С. 81-86.

14 Румянцев Ю.В. Методические вопросы исследования наджености больших систем энергетики // Исследование надежности срабатывания цифровой дифференциальной защиты трансформатора в системе динамического моделирования MATLAB-Simulink. Минск. 2015. Т. 66. С. 390-396.

15 Трансформаторы силовые масляные класса напряжения 110 кВ // Официальный сайт производителя ООО "Тольяттинский трансформатор". 2021. URL: https://www.transformator.com.ru/ttproduction/transform/145/1641/ (дата обращения: 8.04.2025).

16 Тугашова Л.Г., Затонский А.В. Моделирование объектов управления в MatLab. М.: Лань, 2020. 144 с.

17 Чернобровов Н.В. Релейная защита. 5-е изд. М.: Энергия, 1974. 680 с.

18 Шнеерсон Э.М. Цифровая релейная защита. М.: Энергоатомиздат, 2007. 549 с.

19 ABB Group. Technical Description Protection Terminals and Relay REF 54, RET 54, REX 521 // ABB Library. URL: https://library.e.abb.com/public/1bf244f72971c05fc2257036003d8c28/ref54_dnpprotmanENe.pdf (дата обращения: 03.03.2025).

20 Bernardon D.P., Garcia V.J. Smart Operation for Power Distribution Systems. New-York: Springer, 2018. 170 pp.

21 General Electric. Multilin T60. Transformer Protection System // GE Vernova. URL: https://www.gevernova.com/grid-solutions/automation/protection-control-metering/transformer-protection/multilin-t60 (дата обращения: 03.03.2025).

22 Kezunovic M., Ren J., Lotfifard S. Design, Modeling and Evaluation of Protective Relays for Power Systems. New-York: Springer, 2016. 297 C.

23 Lee H.H. Programming and Engineering Computing with MATLAB 2023. SDC Publications, 2023. 530 C.

24 Raja A.K., Srivastava A.P., Dwivedi M. Power Plant Engineering. New Age International (P) Ltd. New Delhi, 2006. 491 C

25 Ziegler G. Numerical Differential Protection: Principles and Applications. 2nd ed. Hoboken: John Wiley & Sons, 2012. 287 C.

Приложение А

Программный код модели дифференциальной защиты силового трансформатора на языке Си

```
#include "Diff_protect_PT.h"
#include "rtwtypes.h"
#include <math.h>
#include <emmintrin.h>
#include <stddef.h>
#include "zero_crossing_types.h"
#define NumBitsPerChar                 8U
#ifndef rtmIsMajorTimeStep
#define rtmIsMajorTimeStep(rtm)        (((rtm)->Timing.simTimeStep) ==
MAJOR_TIME_STEP)
#endif
#ifndef rtmIsMinorTimeStep
#define rtmIsMinorTimeStep(rtm)        (((rtm)->Timing.simTimeStep) ==
MINOR_TIME_STEP)
#endif
#ifndef rtmSetTPtr
#define rtmSetTPtr(rtm, val)           ((rtm)->Timing.t = (val))
#endif
extern real_T rt_hypotd_snf(real_T u0, real_T u1);
static void NEGATIVEEdge_Disable(DW_NEGATIVEEdge *localDW);
static void NEGATIVEEdge(real_T rtu_Enable, boolean_T rtu_IN,
boolean_T
  rtu_INprevious, boolean_T *rty_OUT, DW_NEGATIVEEdge *localDW);
static void POSITIVEEdge_Disable(DW_POSITIVEEdge *localDW);
static void POSITIVEEdge(real_T rtu_Enable, boolean_T rtu_IN,
boolean_T
  rtu_INprevious, boolean_T *rty_OUT, DW_POSITIVEEdge *localDW);
static void OFFDelay_Init(boolean_T rtp_ic, DW_OFFDelay *localDW);
static void OFFDelay_Disable(DW_OFFDelay *localDW);
static void OFFDelay_Update(boolean_T rtu_IN, DW_OFFDelay *localDW);
static void OFFDelay(RT_MODEL * const rtM, boolean_T rtu_Enable,
boolean_T
                     rtu_IN, real_T rtu_clock, real_T rtu_DELAY,
boolean_T
                     *rty_OUT, const ConstB_OFFDelay *localC,
DW_OFFDelay
                     *localDW, ZCE_OFFDelay *localZCE);
static void ONDelay_Init(boolean_T rtp_ic, DW_ONDelay *localDW);
static void ONDelay_Disable(DW_ONDelay *localDW);
static void ONDelay_Update(boolean_T rtu_IN, DW_ONDelay *localDW);
static void ONDelay(RT_MODEL * const rtM, boolean_T rtu_Enable,
boolean_T rtu_IN,
                    real_T rtu_clock, real_T rtu_DELAY, boolean_T
*rty_OUT,
                    const ConstB_ONDelay *localC, DW_ONDelay *localDW,
                    ZCE_ONDelay *localZCE);
```

```
extern real_T rtInf;
extern real_T rtMinusInf;
extern real_T rtNaN;
extern real32_T rtInfF;
extern real32_T rtMinusInfF;
extern real32_T rtNaNF;
static void rt_InitInfAndNaN(size_t realSize);
static boolean_T rtIsInf(real_T value);
static boolean_T rtIsInfF(real32_T value);
static boolean_T rtIsNaN(real_T value);
static boolean_T rtIsNaNF(real32_T value);
typedef struct {
  struct {
    uint32_T wordH;
    uint32_T wordL;
  } words;
} BigEndianIEEEDouble;
typedef struct {
  struct {
    uint32_T wordL;
    uint32_T wordH;
  } words;
} LittleEndianIEEEDouble;
typedef struct {
  union {
    real32_T wordLreal;
    uint32_T wordLuint;
  } wordL;
} IEEESingle;
real_T rtInf;
real_T rtMinusInf;
real_T rtNaN;
real32_T rtInfF;
real32_T rtMinusInfF;
real32_T rtNaNF;
static real_T rtGetInf(void);
static real32_T rtGetInfF(void);
static real_T rtGetMinusInf(void);
static real32_T rtGetMinusInfF(void);
static real_T rtGetNaN(void);
static real32_T rtGetNaNF(void);
static void rt_InitInfAndNaN(size_t realSize)
{  (void) (realSize);
  rtNaN = rtGetNaN();
  rtNaNF = rtGetNaNF();
  rtInf = rtGetInf();
  rtInfF = rtGetInfF();
  rtMinusInf = rtGetMinusInf();
  rtMinusInfF = rtGetMinusInfF();}
static boolean_T rtIsInf(real_T value)
{
```

```
  return (boolean_T)((value==rtInf || value==rtMinusInf) ? 1U : 0U);
}
static boolean_T rtIsInfF(real32_T value)
{
  return (boolean_T)(((value)==rtInfF || (value)==rtMinusInfF) ? 1U :
0U);
}
static boolean_T rtIsNaN(real_T value)
{
  boolean_T result = (boolean_T) 0;
  size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
  if (bitsPerReal == 32U) {
    result = rtIsNaNF((real32_T)value);
  } else {
    union {
      LittleEndianIEEEDouble bitVal;
      real_T fltVal;
    } tmpVal;
    tmpVal.fltVal = value;
    result = (boolean_T)((tmpVal.bitVal.words.wordH & 0x7FF00000) ==
0x7FF00000 &&
                         ( (tmpVal.bitVal.words.wordH & 0x000FFFFF) !=
0 ||
                          (tmpVal.bitVal.words.wordL != 0) ));
  }
  return result;
}
static boolean_T rtIsNaNF(real32_T value)
{
  IEEESingle tmp;
  tmp.wordL.wordLreal = value;
  return (boolean_T)( (tmp.wordL.wordLuint & 0x7F800000) == 0x7F800000
&&
                     (tmp.wordL.wordLuint & 0x007FFFFF) != 0 );
}
static real_T rtGetInf(void)
{
  size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
  real_T inf = 0.0;
  if (bitsPerReal == 32U) {
    inf = rtGetInfF();
  } else {
    union {
      LittleEndianIEEEDouble bitVal;
      real_T fltVal;
    } tmpVal;
    tmpVal.bitVal.words.wordH = 0x7FF00000U;
    tmpVal.bitVal.words.wordL = 0x00000000U;
    inf = tmpVal.fltVal;
  }
  return inf;
```

```
}
static real32_T rtGetInfF(void)
{
  IEEESingle infF;
  infF.wordL.wordLuint = 0x7F800000U;
  return infF.wordL.wordLreal;
}
static real_T rtGetMinusInf(void)
{
  size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
  real_T minf = 0.0;
  if (bitsPerReal == 32U) {
    minf = rtGetMinusInfF();
  } else {
    union {
      LittleEndianIEEEDouble bitVal;
      real_T fltVal;
    } tmpVal;
    tmpVal.bitVal.words.wordH = 0xFFF00000U;
    tmpVal.bitVal.words.wordL = 0x00000000U;
    minf = tmpVal.fltVal;
  }
  return minf;
}
static real32_T rtGetMinusInfF(void)
{
  IEEESingle minfF;
  minfF.wordL.wordLuint = 0xFF800000U;
  return minfF.wordL.wordLreal;
}
static real_T rtGetNaN(void)
{
  size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
  real_T nan = 0.0;
  if (bitsPerReal == 32U) {
    nan = rtGetNaNF();
  } else {
    union {
      LittleEndianIEEEDouble bitVal;
      real_T fltVal;
    } tmpVal;
    tmpVal.bitVal.words.wordH = 0xFFF80000U;
    tmpVal.bitVal.words.wordL = 0x00000000U;
    nan = tmpVal.fltVal;
  }
  return nan;
}
static real32_T rtGetNaNF(void)
{
  IEEESingle nanF = { { 0.0F } };
  nanF.wordL.wordLuint = 0xFFC00000U;
```

```
  return nanF.wordL.wordLreal;
}
static void NEGATIVEEdge_Disable(DW_NEGATIVEEdge *localDW)
{
  localDW->NEGATIVEEdge_MODE = false;
}
static void NEGATIVEEdge(real_T rtu_Enable, boolean_T rtu_IN,
boolean_T
  rtu_INprevious, boolean_T *rty_OUT, DW_NEGATIVEEdge *localDW)
{
  if (rtu_Enable > 0.0) {
    localDW->NEGATIVEEdge_MODE = true;
  } else if (localDW->NEGATIVEEdge_MODE) {
    NEGATIVEEdge_Disable(localDW);
  }
  if (localDW->NEGATIVEEdge_MODE) {
    *rty_OUT = ((int32_T)rtu_INprevious > (int32_T)rtu_IN);
  }
}
static void POSITIVEEdge_Disable(DW_POSITIVEEdge *localDW)
{
  localDW->POSITIVEEdge_MODE = false;
}
static void POSITIVEEdge(real_T rtu_Enable, boolean_T rtu_IN,
boolean_T
  rtu_INprevious, boolean_T *rty_OUT, DW_POSITIVEEdge *localDW)
{
  if (rtu_Enable > 0.0) {
    localDW->POSITIVEEdge_MODE = true;
  } else if (localDW->POSITIVEEdge_MODE) {
    POSITIVEEdge_Disable(localDW);
  }
  if (localDW->POSITIVEEdge_MODE) {
    *rty_OUT = ((int32_T)rtu_INprevious < (int32_T)rtu_IN);
  }
}
static void OFFDelay_Init(boolean_T rtp_ic, DW_OFFDelay *localDW)
{
  localDW->Memory_PreviousInput = rtp_ic;
  localDW->ICic_PreviousInput = -1.0E+99;
}
static void OFFDelay_Disable(DW_OFFDelay *localDW)
{
  if (localDW->POSITIVEEdge_j.POSITIVEEdge_MODE) {
    POSITIVEEdge_Disable(&localDW->POSITIVEEdge_j);
  }
  if (localDW->NEGATIVEEdge_f.NEGATIVEEdge_MODE) {
    NEGATIVEEdge_Disable(&localDW->NEGATIVEEdge_f);
  }
  localDW->OFFDelay_MODE = false;}
```

```
static void OFFDelay(RT_MODEL * const rtM, boolean_T rtu_Enable,
boolean_T
                     rtu_IN, real_T rtu_clock, real_T rtu_DELAY,
boolean_T
                     *rty_OUT, const ConstB_OFFDelay *localC,
DW_OFFDelay
                     *localDW, ZCE_OFFDelay *localZCE)
{
  if (rtu_Enable) {
    localDW->OFFDelay_MODE = true;
  } else if (localDW->OFFDelay_MODE) {
    OFFDelay_Disable(localDW);
  }
  if (localDW->OFFDelay_MODE) {
    localDW->Memory = localDW->Memory_PreviousInput;
    POSITIVEEdge(localC->MultiportSwitch[0], rtu_IN, localDW->Memory,
                 &localDW->RelationalOperator1, &localDW-
>POSITIVEEdge_j);
    NEGATIVEEdge(localC->MultiportSwitch[1], rtu_IN, localDW->Memory,
                 &localDW->RelationalOperator1_d, &localDW-
>NEGATIVEEdge_f);
    localDW->LogicalOperator1 = (localDW->RelationalOperator1 ||
      localDW->RelationalOperator1_d);
    localDW->ICic = localDW->ICic_PreviousInput;
    if (localDW->LogicalOperator1) {
      localDW->Switch = rtu_clock;
    } else {
      localDW->Switch = localDW->ICic;
    }
    *rty_OUT = ((!(rtu_clock >= localDW->Switch + rtu_DELAY)) ||
rtu_IN);
    localDW->Clock = rtM->Timing.t[0];
    localZCE->TriggeredSubsystem_Trig_ZCE_k = localDW-
>LogicalOperator1;
  }
}
static void OFFDelay_Update(boolean_T rtu_IN, DW_OFFDelay *localDW)
{
  if (localDW->OFFDelay_MODE) {
    localDW->Memory_PreviousInput = rtu_IN;
    localDW->ICic_PreviousInput = localDW->Switch;
  }
}
static void ONDelay_Init(boolean_T rtp_ic, DW_ONDelay *localDW)
{
  localDW->Memory_PreviousInput = rtp_ic;
  localDW->ICic_PreviousInput = -1.0E+99;
}
static void ONDelay_Disable(DW_ONDelay *localDW)
{ if (localDW->POSITIVEEdge_l.POSITIVEEdge_MODE) {
    POSITIVEEdge_Disable(&localDW->POSITIVEEdge_l);}
```

```
  if (localDW->NEGATIVEEdge_b.NEGATIVEEdge_MODE) {
    NEGATIVEEdge_Disable(&localDW->NEGATIVEEdge_b);
  }
  localDW->ONDelay_MODE = false;
}
static void ONDelay(RT_MODEL * const rtM, boolean_T rtu_Enable,
boolean_T rtu_IN,
                    real_T rtu_clock, real_T rtu_DELAY, boolean_T
*rty_OUT,
                    const ConstB_ONDelay *localC, DW_ONDelay *localDW,
                    ZCE_ONDelay *localZCE)
{
  if (rtu_Enable) {
    localDW->ONDelay_MODE = true;
  } else if (localDW->ONDelay_MODE) {
    ONDelay_Disable(localDW);
  }
  if (localDW->ONDelay_MODE) {
    localDW->Memory = localDW->Memory_PreviousInput;
    POSITIVEEdge(localC->MultiportSwitch[0], rtu_IN, localDW->Memory,
                 &localDW->RelationalOperator1, &localDW-
>POSITIVEEdge_l);
    NEGATIVEEdge(localC->MultiportSwitch[1], rtu_IN, localDW->Memory,
                 &localDW->RelationalOperator1_a, &localDW-
>NEGATIVEEdge_b);
    localDW->LogicalOperator1 = (localDW->RelationalOperator1 ||
      localDW->RelationalOperator1_a);
    localDW->ICic = localDW->ICic_PreviousInput;
    if (localDW->LogicalOperator1) {
      localDW->Switch = rtu_clock;
    } else {
      localDW->Switch = localDW->ICic;
    }
    *rty_OUT = ((rtu_clock >= localDW->Switch + rtu_DELAY) && rtu_IN);
    localDW->Clock = rtM->Timing.t[0];
    localZCE->TriggeredSubsystem_Trig_ZCE = localDW->LogicalOperator1;
  }
}
static void ONDelay_Update(boolean_T rtu_IN, DW_ONDelay *localDW)
{
  if (localDW->ONDelay_MODE) {
    localDW->Memory_PreviousInput = rtu_IN;
    localDW->ICic_PreviousInput = localDW->Switch;
  }
}
real_T rt_hypotd_snf(real_T u0, real_T u1)
{
  real_T a;
  real_T y;
  a = fabs(u0);
  y = fabs(u1);
```

```
  if (a < y) {
    a /= y;
    y *= sqrt(a * a + 1.0);
  } else if (a > y) {
    y /= a;
    y = sqrt(y * y + 1.0) * a;
  } else if (!rtIsNaN(y)) {
    y = a * 1.4142135623730951;
  }
  return y;
}
void Diff_protect_PT_step(RT_MODEL *const rtM, real_T rtU_I_A_HV,
real_T
  rtU_I_B_HV, real_T rtU_I_C_HV, real_T rtU_I_A_LV, real_T rtU_I_B_LV,
real_T
  rtU_I_C_LV, boolean_T *rtY_trig)
{  DW *rtDW = rtM->dwork;
  PrevZCX *rtPrevZCX = rtM->prevZCSigState;
  real_T rtb_Integrator[6];
  real_T rtb_Delay[6];
  real_T rtb_Integrator_c[6];
  real_T rtb_Delay_k[6];
  real_T rtb_Integrator_n[6];
  real_T rtb_Delay_m[6];
  real_T rtb_Integrator_o[6];
  real_T rtb_Delay_ke[6];
  real_T rtb_Integrator_j[6];
  real_T rtb_Delay_f[6];
  real_T rtb_Integrator_d[6];
  real_T rtb_Delay_i[6];
  real_T rtb_Product_m[6];
  real_T rtb_Product_c[6];
  real_T rtb_Product_n[6];
  real_T rtb_Product_mz[6];
  real_T rtb_Product_au[6];
  real_T rtb_Product_o[6];
  { real_T rtb_Gain1_n[6];
    real_T rtb_Abs;
    real_T rtb_Gain1_py;
    real_T rtb_Gain2;
    real_T rtb_Gain2_j;
    real_T rtb_Integrator_ca_idx_0;
    real_T rtb_Integrator_ca_idx_1;
    real_T rtb_Integrator_dp_idx_0;
    real_T rtb_Integrator_dp_idx_1;
    real_T rtb_Integrator_dw_idx_0;
    real_T rtb_Integrator_f4_idx_0;
    real_T rtb_Integrator_f4_idx_1;
    real_T rtb_Integrator_f_idx_0;
    real_T rtb_Integrator_f_idx_1;
    real_T rtb_RealImagtoComplex_im;
```

```
    real_T rtb_RealImagtoComplex_im_a;
    real_T rtb_RealImagtoComplex_im_ag;
    real_T rtb_RealImagtoComplex_im_ag3;
    real_T rtb_RealImagtoComplex_im_ag3u;
    real_T rtb_RealImagtoComplex_re;
    real_T rtb_RealImagtoComplex_re_g;
    real_T rtb_RealImagtoComplex_re_gp;
    real_T rtb_RealImagtoComplex_re_gpd;
    real_T rtb_RealImagtoComplex_re_gpd3;
    real_T rtb_sincos_o1_idx_0;
    real_T rtb_sincos_o1_idx_1;
    int32_T i;
    for (i = 0; i < 6; i++) {
      if (rtDW->Integrator_PrevResetState != 0) {
        rtDW->Integrator_DSTATE[i] = 0.0;
      }
      rtb_Integrator[i] = rtDW->Integrator_DSTATE[i];
      rtb_Delay[i] = rtDW->Delay_DSTATE[i];
      rtb_Gain1_n[i] = ((rtb_Integrator[i] - ((rtb_Delay[i] -
        rtDW->UnitDelay2_DSTATE[i]) * 0.0 + rtDW-
>UnitDelay2_DSTATE[i])) +
                        (((rtb_Integrator[i] - rtDW-
>UnitDelay1_DSTATE[i]) +
                          rtDW->UnitDelay1_DSTATE[i]) - rtDW->
                         UnitDelay2_DSTATE[i])) * 0.5 * 100.0;
    }
    rtb_sincos_o1_idx_0 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[0],
      rtb_Gain1_n[3]);
    rtb_sincos_o1_idx_1 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[1],
      rtb_Gain1_n[4]);
    rtb_RealImagtoComplex_re_gpd3 = rtb_Gain1_n[2];
    rtb_RealImagtoComplex_im_ag3u = rtb_Gain1_n[5];
    for (i = 0; i < 6; i++) {
      if (rtDW->Integrator_PrevResetState_j != 0) {
        rtDW->Integrator_DSTATE_a[i] = 0.0;
      }
      rtb_Integrator_c[i] = rtDW->Integrator_DSTATE_a[i];
      rtb_Delay_k[i] = rtDW->Delay_DSTATE_n[i];
      rtb_Gain1_n[i] = ((rtb_Integrator_c[i] - ((rtb_Delay_k[i] -
        rtDW->UnitDelay2_DSTATE_g[i]) * 0.0 + rtDW-
>UnitDelay2_DSTATE_g[i])) +
                        (((rtb_Integrator_c[i] - rtDW-
>UnitDelay1_DSTATE_j[i]) +
                          rtDW->UnitDelay1_DSTATE_j[i]) -
                         rtDW->UnitDelay2_DSTATE_g[i])) * 0.5 * 100.0;
    }
    rtb_Integrator_ca_idx_0 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[0],
      rtb_Gain1_n[3]);
```

```
    rtb_Integrator_ca_idx_1 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[1],
      rtb_Gain1_n[4]);
    rtb_RealImagtoComplex_re_gpd = rtb_Gain1_n[2];
    rtb_RealImagtoComplex_im_ag3 = rtb_Gain1_n[5];
    rtb_Abs = fabs(rtb_sincos_o1_idx_0 - rtb_Integrator_ca_idx_0);
    rtDW->DataTypeConversion1 = (rtb_Abs > 0.3);
    rtb_Gain2_j = rtM->Timing.t[0];
    ONDelay(rtM, true, rtDW->DataTypeConversion1, rtb_Gain2_j, 0.1,
            &rtDW->LogicalOperator2_b, &rtConstB.ONDelay_kz, &rtDW-
>ONDelay_kz,
            &rtPrevZCX->ONDelay_kz);
    OFFDelay(rtM, rtConstB.LogicalOperator2_j, rtDW-
>DataTypeConversion1,
             rtb_Gain2_j, 0.1, &rtDW->LogicalOperator2_a,
&rtConstB.OFFDelay_c,
             &rtDW->OFFDelay_c, &rtPrevZCX->OFFDelay_c);
    rtDW->DataTypeConversion1_n = (rtb_Abs * 2.5 >= 6.41);
    ONDelay(rtM, true, rtDW->DataTypeConversion1_n, rtb_Gain2_j,
0.003,
            &rtDW->LogicalOperator2_ha, &rtConstB.ONDelay_k, &rtDW-
>ONDelay_k,
            &rtPrevZCX->ONDelay_k);
    OFFDelay(rtM, rtConstB.LogicalOperator2, rtDW-
>DataTypeConversion1_n,
             rtb_Gain2_j, 0.003, &rtDW->LogicalOperator2_p,
&rtConstB.OFFDelay_e,
             &rtDW->OFFDelay_e, &rtPrevZCX->OFFDelay_e);
    for (i = 0; i < 6; i++) {
      if (rtDW->Integrator_PrevResetState_i != 0) {
        rtDW->Integrator_DSTATE_k[i] = 0.0;
      }
      rtb_Integrator_n[i] = rtDW->Integrator_DSTATE_k[i];
      rtb_Delay_m[i] = rtDW->Delay_DSTATE_o[i];
      rtb_Gain1_n[i] = ((rtb_Integrator_n[i] - ((rtb_Delay_m[i] -
        rtDW->UnitDelay2_DSTATE_p[i]) * 0.0 + rtDW-
>UnitDelay2_DSTATE_p[i])) +
                        (((rtb_Integrator_n[i] - rtDW-
>UnitDelay1_DSTATE_m[i]) +
                          rtDW->UnitDelay1_DSTATE_m[i]) -
                         rtDW->UnitDelay2_DSTATE_p[i])) * 0.5 * 100.0;
    }
    rtb_Integrator_f4_idx_0 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[0],
      rtb_Gain1_n[3]);
    rtb_Integrator_f4_idx_1 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[1],
      rtb_Gain1_n[4]);
    rtb_RealImagtoComplex_re_gp = rtb_Gain1_n[2];
    rtb_RealImagtoComplex_im_ag = rtb_Gain1_n[5];
    for (i = 0; i < 6; i++) {
```

```
      if (rtDW->Integrator_PrevResetState_f != 0) {
        rtDW->Integrator_DSTATE_a0[i] = 0.0;
      }
      rtb_Integrator_o[i] = rtDW->Integrator_DSTATE_a0[i];
      rtb_Delay_ke[i] = rtDW->Delay_DSTATE_k[i];
      rtb_Gain1_n[i] = ((rtb_Integrator_o[i] - ((rtb_Delay_ke[i] -
        rtDW->UnitDelay2_DSTATE_j[i]) * 0.0 + rtDW-
>UnitDelay2_DSTATE_j[i])) +
                        (((rtb_Integrator_o[i] - rtDW-
>UnitDelay1_DSTATE_i[i]) +
                          rtDW->UnitDelay1_DSTATE_i[i]) -
                         rtDW->UnitDelay2_DSTATE_j[i])) * 0.5 * 100.0;
    }
    rtb_Integrator_dp_idx_0 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[0],
      rtb_Gain1_n[3]);
    rtb_Integrator_dp_idx_1 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[1],
      rtb_Gain1_n[4]);
    rtb_RealImagtoComplex_re_g = rtb_Gain1_n[2];
    rtb_RealImagtoComplex_im_a = rtb_Gain1_n[5];
    rtb_Gain2 = fabs(rtb_Integrator_f4_idx_0 -
rtb_Integrator_dp_idx_0);
    rtDW->DataTypeConversion1_g = (rtb_Gain2 > 0.3);
    ONDelay(rtM, true, rtDW->DataTypeConversion1_g, rtb_Gain2_j, 0.1,
            &rtDW->LogicalOperator2_c, &rtConstB.ONDelay_i, &rtDW-
>ONDelay_i,
            &rtPrevZCX->ONDelay_i);
    OFFDelay(rtM, rtConstB.LogicalOperator2_f, rtDW-
>DataTypeConversion1_g,
             rtb_Gain2_j, 0.1, &rtDW->LogicalOperator2_n,
&rtConstB.OFFDelay_c3,
             &rtDW->OFFDelay_c3, &rtPrevZCX->OFFDelay_c3);
    rtDW->DataTypeConversion1_gn = (rtb_Gain2 * 2.5 >= 6.41);
    ONDelay(rtM, true, rtDW->DataTypeConversion1_gn, rtb_Gain2_j,
0.003,
            &rtDW->LogicalOperator2_i, &rtConstB.ONDelay_e, &rtDW-
>ONDelay_e,
            &rtPrevZCX->ONDelay_e);
    OFFDelay(rtM, rtConstB.LogicalOperator2_a, rtDW-
>DataTypeConversion1_gn,
             rtb_Gain2_j, 0.003, &rtDW->LogicalOperator2_ni,
             &rtConstB.OFFDelay_k, &rtDW->OFFDelay_k, &rtPrevZCX-
>OFFDelay_k);
    for (i = 0; i < 6; i++) {
      if (rtDW->Integrator_PrevResetState_p != 0) {
        rtDW->Integrator_DSTATE_h[i] = 0.0;
      }
      rtb_Integrator_j[i] = rtDW->Integrator_DSTATE_h[i];
      rtb_Delay_f[i] = rtDW->Delay_DSTATE_g[i];
      rtb_Gain1_n[i] = ((rtb_Integrator_j[i] - ((rtb_Delay_f[i] -
```

```
        rtDW->UnitDelay2_DSTATE_c[i]) * 0.0 + rtDW-
>UnitDelay2_DSTATE_c[i])) +
                         (((rtb_Integrator_j[i] - rtDW-
>UnitDelay1_DSTATE_b[i]) +
                           rtDW->UnitDelay1_DSTATE_b[i]) -
                          rtDW->UnitDelay2_DSTATE_c[i])) * 0.5 * 100.0;
    }
    rtb_Integrator_f_idx_0 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[0],
      rtb_Gain1_n[3]);
    rtb_Integrator_f_idx_1 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[1],
      rtb_Gain1_n[4]);
    rtb_RealImagtoComplex_re = rtb_Gain1_n[2];
    rtb_RealImagtoComplex_im = rtb_Gain1_n[5];
    for (i = 0; i < 6; i++) {
      if (rtDW->Integrator_PrevResetState_o != 0) {
        rtDW->Integrator_DSTATE_i[i] = 0.0;
      }
      rtb_Integrator_d[i] = rtDW->Integrator_DSTATE_i[i];
      rtb_Delay_i[i] = rtDW->Delay_DSTATE_c[i];
      rtb_Gain1_n[i] = ((rtb_Integrator_d[i] - ((rtb_Delay_i[i] -
        rtDW->UnitDelay2_DSTATE_p5[i]) * 0.0 + rtDW-
>UnitDelay2_DSTATE_p5[i])) +
                        (((rtb_Integrator_d[i] - rtDW-
>UnitDelay1_DSTATE_h[i]) +
                          rtDW->UnitDelay1_DSTATE_h[i]) -
                         rtDW->UnitDelay2_DSTATE_p5[i])) * 0.5 *
100.0;
    }
    real_T trig_tmp;
    real_T trig_tmp_0;
    real_T trig_tmp_1;
    rtb_Integrator_dw_idx_0 = 0.70710678118654746 *
rt_hypotd_snf(rtb_Gain1_n[0],
      rtb_Gain1_n[3]);
    rtb_Gain1_py = fabs(rtb_Integrator_f_idx_0 -
rtb_Integrator_dw_idx_0);
    rtDW->DataTypeConversion1_e = (rtb_Gain1_py > 0.3);
    ONDelay(rtM, true, rtDW->DataTypeConversion1_e, rtb_Gain2_j, 0.1,
            &rtDW->LogicalOperator2, &rtConstB.ONDelay_l, &rtDW-
>ONDelay_l,
            &rtPrevZCX->ONDelay_l);
    OFFDelay(rtM, rtConstB.LogicalOperator2_g, rtDW-
>DataTypeConversion1_e,
             rtb_Gain2_j, 0.1, &rtDW->LogicalOperator2_l,
&rtConstB.OFFDelay_av,
             &rtDW->OFFDelay_av, &rtPrevZCX->OFFDelay_av);
    rtDW->DataTypeConversion1_i = (rtb_Gain1_py * 2.5 >= 6.41);
    ONDelay(rtM, true, rtDW->DataTypeConversion1_i, rtb_Gain2_j,
0.003,
```

```
            &rtDW->LogicalOperator2_h, &rtConstB.ONDelay_n, &rtDW-
>ONDelay_n,
            &rtPrevZCX->ONDelay_n);
    OFFDelay(rtM, rtConstB.LogicalOperator2_af, rtDW-
>DataTypeConversion1_i,
             rtb_Gain2_j, 0.003, &rtDW->LogicalOperator2_g,
&rtConstB.OFFDelay_a,
             &rtDW->OFFDelay_a, &rtPrevZCX->OFFDelay_a);
    rtb_Gain2_j = fmax(fmax((rtb_sincos_o1_idx_0 +
rtb_Integrator_ca_idx_0) *
      0.5, (rtb_Integrator_f4_idx_0 + rtb_Integrator_dp_idx_0) * 0.5),
                       (rtb_Integrator_f_idx_0 +
rtb_Integrator_dw_idx_0) * 0.5);
    rtb_Integrator_f_idx_0 = 1.25 * rtb_Gain2_j;
    trig_tmp = 1.25 * rtb_Gain2;
    trig_tmp_0 = 0.44 * rtb_Gain2_j * 0.3;
    trig_tmp_1 = 0.65 * rtb_Gain2_j * 0.74;
    *rtY_trig = (rtDW->LogicalOperator2_b || rtDW->LogicalOperator2_a
||
                 (rtDW->LogicalOperator2_ha || rtDW-
>LogicalOperator2_p) ||
                 (rtDW->LogicalOperator2_c || rtDW->LogicalOperator2_n
||
                  (rtDW->LogicalOperator2_i || rtDW-
>LogicalOperator2_ni)) ||
                 (rtDW->LogicalOperator2 || rtDW->LogicalOperator2_l
||
                  (rtDW->LogicalOperator2_h || rtDW-
>LogicalOperator2_g)) ||
                 ((!(rtb_sincos_o1_idx_1 / rtb_sincos_o1_idx_0 <=
0.15)) &&
                  (!(0.70710678118654746 * rt_hypotd_snf
                     (rtb_RealImagtoComplex_re_gpd3,
                      rtb_RealImagtoComplex_im_ag3u) /
rtb_sincos_o1_idx_0 >=
                     0.1)) && ((!(rtb_Integrator_ca_idx_1 /
      rtb_Integrator_ca_idx_0 <= 0.15)) && (!(0.70710678118654746 *
      rt_hypotd_snf(rtb_RealImagtoComplex_re_gpd,
rtb_RealImagtoComplex_im_ag3) /
      rtb_Integrator_ca_idx_0 >= 0.1))) && ((!(rtb_Integrator_f4_idx_1
/
      rtb_Integrator_f4_idx_0 <= 0.15)) && (!(0.70710678118654746 *
      rt_hypotd_snf(rtb_RealImagtoComplex_re_gp,
rtb_RealImagtoComplex_im_ag) /
      rtb_Integrator_f4_idx_1 >= 0.1)) && ((!(rtb_Integrator_dp_idx_1
/
      rtb_Integrator_dp_idx_0 <= 0.15)) && (!(0.70710678118654746 *
      rt_hypotd_snf(rtb_RealImagtoComplex_re_g,
rtb_RealImagtoComplex_im_a) /
      rtb_Integrator_dp_idx_0 >= 0.1)))) &&
((!(rtb_Integrator_f4_idx_0 /
```

```
      rtb_Integrator_f_idx_1 <= 0.15)) && (!(rtb_Integrator_f4_idx_0 /
      (0.70710678118654746 * rt_hypotd_snf(rtb_RealImagtoComplex_re,
      rtb_RealImagtoComplex_im)) >= 0.1)) && ((!(0.70710678118654746 *
      rt_hypotd_snf(rtb_Gain1_n[1], rtb_Gain1_n[4]) /
rtb_Integrator_dw_idx_0 <=
      0.15)) && (!(0.70710678118654746 * rt_hypotd_snf(rtb_Gain1_n[2],
      rtb_Gain1_n[5]) / rtb_Integrator_dw_idx_0 >= 0.1))))) || (((0.3
*
      rtb_Gain2_j > rtb_Gain2_j) == (rtb_Abs >= 0.3 * rtb_Abs)) ||
((0.44 *
      rtb_Gain2_j * 0.4 > rtb_Gain2_j) == (rtb_Abs >= 0.44 * rtb_Abs *
0.4)) ||
      ((rtb_Abs >= 0.65 * rtb_Abs * 0.74) == (trig_tmp_1 >
rtb_Gain2_j)) ||
      ((0.3 * rtb_Integrator_f_idx_0 > rtb_Gain2_j) == (0.3 * trig_tmp
>=
      rtb_Gain2)) || ((rtb_Gain2 * 0.44 * 0.3 >= rtb_Gain2) ==
(trig_tmp_0 <
      rtb_Gain2_j)) || ((0.65 * rtb_Integrator_f_idx_0 * 0.74 >
rtb_Gain2_j) ==
                        (trig_tmp * 0.65 * 0.74 >= rtb_Gain2)) ||
((1.25 *
      rtb_Gain2_j * 0.3 < rtb_Gain2_j) == (1.25 * rtb_Gain1_py * 0.3
>=
      rtb_Gain1_py)) || ((rtb_Gain1_py * 0.44 * 0.3 >= rtb_Gain1_py)
==
                         (trig_tmp_0 < rtb_Gain2_j)) || ((rtb_Gain1_py
* 0.65 *
      0.74 >= rtb_Gain1_py) == (trig_tmp_1 < rtb_Gain2_j))));
    rtb_Abs = 9.484666455896301 * rtU_I_A_HV;
    rtb_Integrator_ca_idx_0 = 9.484666455896301 * rtU_I_B_HV;
    rtb_Gain2_j = 9.484666455896301 * rtU_I_C_HV;
    rtb_Integrator_dp_idx_0 = 1.8145527127563055 * rtU_I_A_LV;
    rtb_Gain1_py = 1.8145527127563055 * rtU_I_B_LV;
    rtb_Gain2 = 1.8145527127563055 * rtU_I_C_LV;
    if (rtDW->Initial_FirstOutputTime) {
      rtDW->Initial_FirstOutputTime = false;
      rtb_Integrator_dw_idx_0 = 0.0;
      rtb_Integrator_f_idx_1 = 0.0;
      rtb_Integrator_dp_idx_1 = 0.0;
    } else {
      rtb_Integrator_dw_idx_0 = rtDW->Integrator_DSTATE_m[0] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_m[0]) *
        6.2831853071795862;
      rtb_Integrator_f_idx_1 = rtDW->Integrator_DSTATE_m[1] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_m[1]) *
        6.2831853071795862;
      rtb_Integrator_dp_idx_1 = rtDW->Integrator_DSTATE_m[2] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_m[2]) *
        6.2831853071795862;
    }
```

```
    if ((rtDW->Integrator_DSTATE_m[0] < 0.0) || (rtDW-
>Integrator_DSTATE_m[0] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_m[0] = rtb_Integrator_dw_idx_0;
    }
    rtb_Product_m[0] = rtb_Abs * sin(rtDW->Integrator_DSTATE_m[0]);
    rtb_Product_m[3] = rtb_Abs * cos(rtDW->Integrator_DSTATE_m[0]);
    if ((rtDW->Integrator_DSTATE_m[1] < 0.0) || (rtDW-
>Integrator_DSTATE_m[1] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_m[1] = rtb_Integrator_f_idx_1;
    }
    rtb_Product_m[1] = rtb_Abs * sin(rtDW->Integrator_DSTATE_m[1]);
    rtb_Product_m[4] = rtb_Abs * cos(rtDW->Integrator_DSTATE_m[1]);
    if ((rtDW->Integrator_DSTATE_m[2] < 0.0) || (rtDW-
>Integrator_DSTATE_m[2] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_m[2] = rtb_Integrator_dp_idx_1;
    }
    rtb_Product_m[2] = rtb_Abs * sin(rtDW->Integrator_DSTATE_m[2]);
    rtb_Product_m[5] = rtb_Abs * cos(rtDW->Integrator_DSTATE_m[2]);
    if (rtDW->Initial_FirstOutputTime_o) {
      rtDW->Initial_FirstOutputTime_o = false;
      rtb_Integrator_f_idx_0 = 0.0;
      rtb_Integrator_f_idx_1 = 0.0;
      rtb_Abs = 0.0;
    } else {
      rtb_Integrator_f_idx_0 = rtDW->Integrator_DSTATE_l[0] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_l[0]) *
        6.2831853071795862;
      rtb_Integrator_f_idx_1 = rtDW->Integrator_DSTATE_l[1] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_l[1]) *
        6.2831853071795862;
      rtb_Abs = rtDW->Integrator_DSTATE_l[2] -
floor(0.15915494309189535 *
        rtDW->Integrator_DSTATE_l[2]) * 6.2831853071795862;
    }
    if ((rtDW->Integrator_DSTATE_l[0] < 0.0) || (rtDW-
>Integrator_DSTATE_l[0] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_l[0] = rtb_Integrator_f_idx_0;
    }
    rtb_Product_c[0] = rtb_Integrator_dp_idx_0 * sin(rtDW-
>Integrator_DSTATE_l[0]);
    rtb_Product_c[3] = rtb_Integrator_dp_idx_0 * cos(rtDW-
>Integrator_DSTATE_l[0]);
    if ((rtDW->Integrator_DSTATE_l[1] < 0.0) || (rtDW-
>Integrator_DSTATE_l[1] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_l[1] = rtb_Integrator_f_idx_1;
    }
```

```
    rtb_Product_c[1] = rtb_Integrator_dp_idx_0 * sin(rtDW-
>Integrator_DSTATE_l[1]);
    rtb_Product_c[4] = rtb_Integrator_dp_idx_0 * cos(rtDW-
>Integrator_DSTATE_l[1]);
    if ((rtDW->Integrator_DSTATE_l[2] < 0.0) || (rtDW-
>Integrator_DSTATE_l[2] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_l[2] = rtb_Abs;
    }
    rtb_Product_c[2] = rtb_Integrator_dp_idx_0 * sin(rtDW-
>Integrator_DSTATE_l[2]);
    rtb_Product_c[5] = rtb_Integrator_dp_idx_0 * cos(rtDW-
>Integrator_DSTATE_l[2]);
    if (rtDW->Initial_FirstOutputTime_m) {
      rtDW->Initial_FirstOutputTime_m = false;
      rtb_Integrator_dp_idx_0 = 0.0;
      rtb_Integrator_dp_idx_1 = 0.0;
      rtb_Abs = 0.0;
    } else {
      rtb_Integrator_dp_idx_0 = rtDW->Integrator_DSTATE_mx[0] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_mx[0]) *
        6.2831853071795862;
      rtb_Integrator_dp_idx_1 = rtDW->Integrator_DSTATE_mx[1] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_mx[1]) *
        6.2831853071795862;
      rtb_Abs = rtDW->Integrator_DSTATE_mx[2] -
floor(0.15915494309189535 *
        rtDW->Integrator_DSTATE_mx[2]) * 6.2831853071795862;
    }
    if ((rtDW->Integrator_DSTATE_mx[0] < 0.0) || (rtDW-
>Integrator_DSTATE_mx[0] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_mx[0] = rtb_Integrator_dp_idx_0;
    }
    rtb_Product_n[0] = rtb_Integrator_ca_idx_0 * sin(rtDW-
>Integrator_DSTATE_mx
      [0]);
    rtb_Product_n[3] = rtb_Integrator_ca_idx_0 * cos(rtDW-
>Integrator_DSTATE_mx
      [0]);
    if ((rtDW->Integrator_DSTATE_mx[1] < 0.0) || (rtDW-
>Integrator_DSTATE_mx[1] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_mx[1] = rtb_Integrator_dp_idx_1;
    }
    rtb_Product_n[1] = rtb_Integrator_ca_idx_0 * sin(rtDW-
>Integrator_DSTATE_mx
      [1]);
    rtb_Product_n[4] = rtb_Integrator_ca_idx_0 * cos(rtDW-
>Integrator_DSTATE_mx
      [1]);
```

```
    if ((rtDW->Integrator_DSTATE_mx[2] < 0.0) || (rtDW-
>Integrator_DSTATE_mx[2] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_mx[2] = rtb_Abs;
    }
    rtb_Product_n[2] = rtb_Integrator_ca_idx_0 * sin(rtDW-
>Integrator_DSTATE_mx
      [2]);
    rtb_Product_n[5] = rtb_Integrator_ca_idx_0 * cos(rtDW-
>Integrator_DSTATE_mx
      [2]);
    if (rtDW->Initial_FirstOutputTime_d) {
      rtDW->Initial_FirstOutputTime_d = false;
      rtb_Integrator_f4_idx_0 = 0.0;
      rtb_Integrator_f4_idx_1 = 0.0;
      rtb_Abs = 0.0;
    } else {
      rtb_Integrator_f4_idx_0 = rtDW->Integrator_DSTATE_p[0] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_p[0]) *
        6.2831853071795862;
      rtb_Integrator_f4_idx_1 = rtDW->Integrator_DSTATE_p[1] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_p[1]) *
        6.2831853071795862;
      rtb_Abs = rtDW->Integrator_DSTATE_p[2] -
floor(0.15915494309189535 *
        rtDW->Integrator_DSTATE_p[2]) * 6.2831853071795862;
    }
    if ((rtDW->Integrator_DSTATE_p[0] < 0.0) || (rtDW-
>Integrator_DSTATE_p[0] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_p[0] = rtb_Integrator_f4_idx_0;
    }
    rtb_Product_mz[0] = rtb_Gain1_py * sin(rtDW-
>Integrator_DSTATE_p[0]);
    rtb_Product_mz[3] = rtb_Gain1_py * cos(rtDW-
>Integrator_DSTATE_p[0]);
    if ((rtDW->Integrator_DSTATE_p[1] < 0.0) || (rtDW-
>Integrator_DSTATE_p[1] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_p[1] = rtb_Integrator_f4_idx_1;
    }
    rtb_Product_mz[1] = rtb_Gain1_py * sin(rtDW-
>Integrator_DSTATE_p[1]);
    rtb_Product_mz[4] = rtb_Gain1_py * cos(rtDW-
>Integrator_DSTATE_p[1]);
    if ((rtDW->Integrator_DSTATE_p[2] < 0.0) || (rtDW-
>Integrator_DSTATE_p[2] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_p[2] = rtb_Abs;
    }
```

```
    rtb_Product_mz[2] = rtb_Gain1_py * sin(rtDW-
>Integrator_DSTATE_p[2]);
    rtb_Product_mz[5] = rtb_Gain1_py * cos(rtDW-
>Integrator_DSTATE_p[2]);
    if (rtDW->Initial_FirstOutputTime_b) {
      rtDW->Initial_FirstOutputTime_b = false;
      rtb_Integrator_ca_idx_0 = 0.0;
      rtb_Integrator_ca_idx_1 = 0.0;
      rtb_Gain1_py = 0.0;
    } else {
      rtb_Integrator_ca_idx_0 = rtDW->Integrator_DSTATE_d[0] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_d[0]) *
        6.2831853071795862;
      rtb_Integrator_ca_idx_1 = rtDW->Integrator_DSTATE_d[1] - floor
        (0.15915494309189535 * rtDW->Integrator_DSTATE_d[1]) *
        6.2831853071795862;
      rtb_Gain1_py = rtDW->Integrator_DSTATE_d[2] -
floor(0.15915494309189535 *
        rtDW->Integrator_DSTATE_d[2]) * 6.2831853071795862;
    }
    if ((rtDW->Integrator_DSTATE_d[0] < 0.0) || (rtDW-
>Integrator_DSTATE_d[0] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_d[0] = rtb_Integrator_ca_idx_0;
    }
    rtb_Product_au[0] = rtb_Gain2_j * sin(rtDW-
>Integrator_DSTATE_d[0]);
    rtb_Product_au[3] = rtb_Gain2_j * cos(rtDW-
>Integrator_DSTATE_d[0]);
    if ((rtDW->Integrator_DSTATE_d[1] < 0.0) || (rtDW-
>Integrator_DSTATE_d[1] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_d[1] = rtb_Integrator_ca_idx_1;
    }
    rtb_Product_au[1] = rtb_Gain2_j * sin(rtDW-
>Integrator_DSTATE_d[1]);
    rtb_Product_au[4] = rtb_Gain2_j * cos(rtDW-
>Integrator_DSTATE_d[1]);
    if ((rtDW->Integrator_DSTATE_d[2] < 0.0) || (rtDW-
>Integrator_DSTATE_d[2] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_d[2] = rtb_Gain1_py;
    }
    rtb_Product_au[2] = rtb_Gain2_j * sin(rtDW-
>Integrator_DSTATE_d[2]);
    rtb_Product_au[5] = rtb_Gain2_j * cos(rtDW-
>Integrator_DSTATE_d[2]);
    if (rtDW->Initial_FirstOutputTime_f) {
      rtDW->Initial_FirstOutputTime_f = false;
      rtb_Gain2_j = 0.0;
      rtb_Gain1_py = 0.0;
```

```
      rtb_Abs = 0.0;
    } else {
      rtb_Gain2_j = rtDW->Integrator_DSTATE_mh[0] -
floor(0.15915494309189535 *
        rtDW->Integrator_DSTATE_mh[0]) * 6.2831853071795862;
      rtb_Gain1_py = rtDW->Integrator_DSTATE_mh[1] -
floor(0.15915494309189535 *
        rtDW->Integrator_DSTATE_mh[1]) * 6.2831853071795862;
      rtb_Abs = rtDW->Integrator_DSTATE_mh[2] -
floor(0.15915494309189535 *
        rtDW->Integrator_DSTATE_mh[2]) * 6.2831853071795862;
    }
    if ((rtDW->Integrator_DSTATE_mh[0] < 0.0) || (rtDW-
>Integrator_DSTATE_mh[0] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_mh[0] = rtb_Gain2_j;
    }
    rtb_Product_o[0] = rtb_Gain2 * sin(rtDW->Integrator_DSTATE_mh[0]);
    rtb_Product_o[3] = rtb_Gain2 * cos(rtDW->Integrator_DSTATE_mh[0]);
    if ((rtDW->Integrator_DSTATE_mh[1] < 0.0) || (rtDW-
>Integrator_DSTATE_mh[1] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_mh[1] = rtb_Gain1_py;
    }
    rtb_Product_o[1] = rtb_Gain2 * sin(rtDW->Integrator_DSTATE_mh[1]);
    rtb_Product_o[4] = rtb_Gain2 * cos(rtDW->Integrator_DSTATE_mh[1]);
    if ((rtDW->Integrator_DSTATE_mh[2] < 0.0) || (rtDW-
>Integrator_DSTATE_mh[2] >=
         6.2831853071795862)) {
      rtDW->Integrator_DSTATE_mh[2] = rtb_Abs;
    }
    rtb_Product_o[2] = rtb_Gain2 * sin(rtDW->Integrator_DSTATE_mh[2]);
    rtb_Product_o[5] = rtb_Gain2 * cos(rtDW->Integrator_DSTATE_mh[2]);
  }
  {
    __m128d tmp;
    __m128d tmp_0;
    int32_T i;
    int_T idxWidth;
    rtDW->Integrator_PrevResetState = 0;
    for (i = 0; i <= 4; i += 2) {
      tmp = _mm_loadu_pd(&rtb_Product_m[i]);
      tmp_0 = _mm_loadu_pd(&rtDW->Integrator_DSTATE[i]);
      _mm_storeu_pd(&rtDW->Integrator_DSTATE[i], _mm_add_pd(_mm_mul_pd
        (_mm_set1_pd(0.001), tmp), tmp_0));
      tmp = _mm_loadu_pd(&rtb_Integrator[i]);
      _mm_storeu_pd(&rtDW->UnitDelay1_DSTATE[i], tmp);
      tmp = _mm_loadu_pd(&rtb_Delay[i]);
      _mm_storeu_pd(&rtDW->UnitDelay2_DSTATE[i], tmp);
    }
    for (i = 0; i < 18; i++) {
```

```
      for (idxWidth = 0; idxWidth < 6; idxWidth++) {
        rtDW->Delay_DSTATE[i * 6 + idxWidth] = rtDW->Delay_DSTATE[(i +
1) * 6 +
          idxWidth];
      }
    }
    rtDW->Integrator_PrevResetState_j = 0;
    for (i = 0; i <= 4; i += 2) {
      tmp = _mm_loadu_pd(&rtb_Integrator[i]);
      _mm_storeu_pd(&rtDW->Delay_DSTATE[i + 108], tmp);
      tmp = _mm_loadu_pd(&rtb_Product_c[i]);
      tmp_0 = _mm_loadu_pd(&rtDW->Integrator_DSTATE_a[i]);
      _mm_storeu_pd(&rtDW->Integrator_DSTATE_a[i],
_mm_add_pd(_mm_mul_pd
        (_mm_set1_pd(0.001), tmp), tmp_0));
      tmp = _mm_loadu_pd(&rtb_Integrator_c[i]);
      _mm_storeu_pd(&rtDW->UnitDelay1_DSTATE_j[i], tmp);
      tmp = _mm_loadu_pd(&rtb_Delay_k[i]);
      _mm_storeu_pd(&rtDW->UnitDelay2_DSTATE_g[i], tmp);
    }
    for (i = 0; i < 18; i++) {
      for (idxWidth = 0; idxWidth < 6; idxWidth++) {
        rtDW->Delay_DSTATE_n[i * 6 + idxWidth] = rtDW-
>Delay_DSTATE_n[(i + 1) *
          6 + idxWidth];
      }
    }
    for (i = 0; i < 6; i++) {
      rtDW->Delay_DSTATE_n[i + 108] = rtb_Integrator_c[i];
    }
    ONDelay_Update(rtDW->DataTypeConversion1, &rtDW->ONDelay_kz);
    OFFDelay_Update(rtDW->DataTypeConversion1, &rtDW->OFFDelay_c);
    ONDelay_Update(rtDW->DataTypeConversion1_n, &rtDW->ONDelay_k);
    OFFDelay_Update(rtDW->DataTypeConversion1_n, &rtDW->OFFDelay_e);
    rtDW->Integrator_PrevResetState_i = 0;
    for (i = 0; i <= 4; i += 2) {
      tmp = _mm_loadu_pd(&rtb_Product_n[i]);
      tmp_0 = _mm_loadu_pd(&rtDW->Integrator_DSTATE_k[i]);
      _mm_storeu_pd(&rtDW->Integrator_DSTATE_k[i],
_mm_add_pd(_mm_mul_pd
        (_mm_set1_pd(0.001), tmp), tmp_0));
      tmp = _mm_loadu_pd(&rtb_Integrator_n[i]);
      _mm_storeu_pd(&rtDW->UnitDelay1_DSTATE_m[i], tmp);
      tmp = _mm_loadu_pd(&rtb_Delay_m[i]);
      _mm_storeu_pd(&rtDW->UnitDelay2_DSTATE_p[i], tmp);
    }
    for (i = 0; i < 18; i++) {
      for (idxWidth = 0; idxWidth < 6; idxWidth++) {
        rtDW->Delay_DSTATE_o[i * 6 + idxWidth] = rtDW-
>Delay_DSTATE_o[(i + 1) *
          6 + idxWidth];
```

```
      }
    }
    rtDW->Integrator_PrevResetState_f = 0;
    for (i = 0; i <= 4; i += 2) {
      tmp = _mm_loadu_pd(&rtb_Integrator_n[i]);
      _mm_storeu_pd(&rtDW->Delay_DSTATE_o[i + 108], tmp);
      tmp = _mm_loadu_pd(&rtb_Product_mz[i]);
      tmp_0 = _mm_loadu_pd(&rtDW->Integrator_DSTATE_a0[i]);
      _mm_storeu_pd(&rtDW->Integrator_DSTATE_a0[i],
_mm_add_pd(_mm_mul_pd
        (_mm_set1_pd(0.001), tmp), tmp_0));
      tmp = _mm_loadu_pd(&rtb_Integrator_o[i]);
      _mm_storeu_pd(&rtDW->UnitDelay1_DSTATE_i[i], tmp);
     tmp = _mm_loadu_pd(&rtb_Delay_ke[i]);
      _mm_storeu_pd(&rtDW->UnitDelay2_DSTATE_j[i], tmp);
    }
    for (i = 0; i < 18; i++) {
      for (idxWidth = 0; idxWidth < 6; idxWidth++) {
        rtDW->Delay_DSTATE_k[i * 6 + idxWidth] = rtDW-
>Delay_DSTATE_k[(i + 1) *
          6 + idxWidth];
      }
    }
    for (i = 0; i < 6; i++) {
      rtDW->Delay_DSTATE_k[i + 108] = rtb_Integrator_o[i];
    }
    ONDelay_Update(rtDW->DataTypeConversion1_g, &rtDW->ONDelay_i);
    OFFDelay_Update(rtDW->DataTypeConversion1_g, &rtDW->OFFDelay_c3);
    ONDelay_Update(rtDW->DataTypeConversion1_gn, &rtDW->ONDelay_e);
    OFFDelay_Update(rtDW->DataTypeConversion1_gn, &rtDW->OFFDelay_k);
    rtDW->Integrator_PrevResetState_p = 0;
    for (i = 0; i <= 4; i += 2) {
      tmp = _mm_loadu_pd(&rtb_Product_au[i]);
      tmp_0 = _mm_loadu_pd(&rtDW->Integrator_DSTATE_h[i]);
      _mm_storeu_pd(&rtDW->Integrator_DSTATE_h[i],
_mm_add_pd(_mm_mul_pd
        (_mm_set1_pd(0.001), tmp), tmp_0));
      tmp = _mm_loadu_pd(&rtb_Integrator_j[i]);
      _mm_storeu_pd(&rtDW->UnitDelay1_DSTATE_b[i], tmp);
      tmp = _mm_loadu_pd(&rtb_Delay_f[i]);
      _mm_storeu_pd(&rtDW->UnitDelay2_DSTATE_c[i], tmp);
    }
    for (i = 0; i < 18; i++) {
      for (idxWidth = 0; idxWidth < 6; idxWidth++) {
        rtDW->Delay_DSTATE_g[i * 6 + idxWidth] = rtDW-
>Delay_DSTATE_g[(i + 1) *
          6 + idxWidth];
      }
    }
    rtDW->Integrator_PrevResetState_o = 0;
    for (i = 0; i <= 4; i += 2) {
```

```
      tmp = _mm_loadu_pd(&rtb_Integrator_j[i]);
      _mm_storeu_pd(&rtDW->Delay_DSTATE_g[i + 108], tmp);
      tmp = _mm_loadu_pd(&rtb_Product_o[i]);
      tmp_0 = _mm_loadu_pd(&rtDW->Integrator_DSTATE_i[i]);
      _mm_storeu_pd(&rtDW->Integrator_DSTATE_i[i],
_mm_add_pd(_mm_mul_pd
        (_mm_set1_pd(0.001), tmp), tmp_0));
      tmp = _mm_loadu_pd(&rtb_Integrator_d[i]);
      _mm_storeu_pd(&rtDW->UnitDelay1_DSTATE_h[i], tmp);
      tmp = _mm_loadu_pd(&rtb_Delay_i[i]);
      _mm_storeu_pd(&rtDW->UnitDelay2_DSTATE_p5[i], tmp);
    }
    for (i = 0; i < 18; i++) {
      for (idxWidth = 0; idxWidth < 6; idxWidth++) {
        rtDW->Delay_DSTATE_c[i * 6 + idxWidth] = rtDW-
>Delay_DSTATE_c[(i + 1) *
          6 + idxWidth];
      }
    }
    for (i = 0; i < 6; i++) {
      rtDW->Delay_DSTATE_c[i + 108] = rtb_Integrator_d[i];
    }
    ONDelay_Update(rtDW->DataTypeConversion1_e, &rtDW->ONDelay_l);
    OFFDelay_Update(rtDW->DataTypeConversion1_e, &rtDW->OFFDelay_av);
    ONDelay_Update(rtDW->DataTypeConversion1_i, &rtDW->ONDelay_n);
    OFFDelay_Update(rtDW->DataTypeConversion1_i, &rtDW->OFFDelay_a);
    rtDW->Integrator_DSTATE_m[0] += 0.31415926535897931;
    rtDW->Integrator_DSTATE_l[0] += 0.31415926535897931;
    rtDW->Integrator_DSTATE_mx[0] += 0.31415926535897931;
    rtDW->Integrator_DSTATE_p[0] += 0.31415926535897931;
    rtDW->Integrator_DSTATE_d[0] += 0.31415926535897931;
    rtDW->Integrator_DSTATE_mh[0] += 0.31415926535897931;
    rtDW->Integrator_DSTATE_m[1] += 0.62831853071795862;
    rtDW->Integrator_DSTATE_l[1] += 0.62831853071795862;
    rtDW->Integrator_DSTATE_mx[1] += 0.62831853071795862;
    rtDW->Integrator_DSTATE_p[1] += 0.62831853071795862;
    rtDW->Integrator_DSTATE_d[1] += 0.62831853071795862;
    rtDW->Integrator_DSTATE_mh[1] += 0.62831853071795862;
    rtDW->Integrator_DSTATE_m[2] += 1.5707963267948968;
    rtDW->Integrator_DSTATE_l[2] += 1.5707963267948968;
    rtDW->Integrator_DSTATE_mx[2] += 1.5707963267948968;
    rtDW->Integrator_DSTATE_p[2] += 1.5707963267948968;
    rtDW->Integrator_DSTATE_d[2] += 1.5707963267948968;
    rtDW->Integrator_DSTATE_mh[2] += 1.5707963267948968;
  }
  rtM->Timing.t[0] =
    ((time_T)(++rtM->Timing.clockTick0)) * rtM->Timing.stepSize0;
  {
    rtM->Timing.clockTick1++;
  }
}
```

```
void Diff_protect_PT_initialize(RT_MODEL *const rtM)
{
  DW *rtDW = rtM->dwork;
  PrevZCX *rtPrevZCX = rtM->prevZCSigState;
  rt_InitInfAndNaN(sizeof(real_T));
  {
    rtsiSetSimTimeStepPtr(&rtM->solverInfo, &rtM->Timing.simTimeStep);
    rtsiSetTPtr(&rtM->solverInfo, &rtmGetTPtr(rtM));
    rtsiSetStepSizePtr(&rtM->solverInfo, &rtM->Timing.stepSize0);
    rtsiSetErrorStatusPtr(&rtM->solverInfo,
(&rtmGetErrorStatus(rtM)));
    rtsiSetRTModelPtr(&rtM->solverInfo, rtM);
  }
  rtsiSetSimTimeStep(&rtM->solverInfo, MAJOR_TIME_STEP);
  rtsiSetSolverName(&rtM->solverInfo,"FixedStepDiscrete");
  rtmSetTPtr(rtM, &rtM->Timing.tArray[0]);
  rtM->Timing.stepSize0 = 0.001;
  rtDW->Initial_FirstOutputTime = true;
  rtDW->Initial_FirstOutputTime_o = true;
  rtDW->Initial_FirstOutputTime_m = true;
  rtDW->Initial_FirstOutputTime_d = true;
  rtDW->Initial_FirstOutputTime_b = true;
  rtDW->Initial_FirstOutputTime_f = true;
  rtPrevZCX->ONDelay_l.TriggeredSubsystem_Trig_ZCE = POS_ZCSIG;
  rtPrevZCX->OFFDelay_av.TriggeredSubsystem_Trig_ZCE_k = POS_ZCSIG;
  rtPrevZCX->ONDelay_n.TriggeredSubsystem_Trig_ZCE = POS_ZCSIG;
  rtPrevZCX->OFFDelay_a.TriggeredSubsystem_Trig_ZCE_k = POS_ZCSIG;
  rtPrevZCX->ONDelay_i.TriggeredSubsystem_Trig_ZCE = POS_ZCSIG;
  rtPrevZCX->OFFDelay_c3.TriggeredSubsystem_Trig_ZCE_k = POS_ZCSIG;
  rtPrevZCX->ONDelay_e.TriggeredSubsystem_Trig_ZCE = POS_ZCSIG;
  rtPrevZCX->OFFDelay_k.TriggeredSubsystem_Trig_ZCE_k = POS_ZCSIG;
  rtPrevZCX->ONDelay_kz.TriggeredSubsystem_Trig_ZCE = POS_ZCSIG;
  rtPrevZCX->OFFDelay_c.TriggeredSubsystem_Trig_ZCE_k = POS_ZCSIG;
  rtPrevZCX->ONDelay_k.TriggeredSubsystem_Trig_ZCE = POS_ZCSIG;
  rtPrevZCX->OFFDelay_e.TriggeredSubsystem_Trig_ZCE_k = POS_ZCSIG;
  ONDelay_Init(false, &rtDW->ONDelay_kz);
  OFFDelay_Init(false, &rtDW->OFFDelay_c);
  ONDelay_Init(false, &rtDW->ONDelay_k);
  OFFDelay_Init(false, &rtDW->OFFDelay_e);
  ONDelay_Init(false, &rtDW->ONDelay_i);
  OFFDelay_Init(false, &rtDW->OFFDelay_c3);
  ONDelay_Init(false, &rtDW->ONDelay_e);
  OFFDelay_Init(false, &rtDW->OFFDelay_k);
  ONDelay_Init(false, &rtDW->ONDelay_l);
  OFFDelay_Init(false, &rtDW->OFFDelay_av);
  ONDelay_Init(false, &rtDW->ONDelay_n);
  OFFDelay_Init(false, &rtDW->OFFDelay_a);
}
```