Экранированный проводник

Оглавление

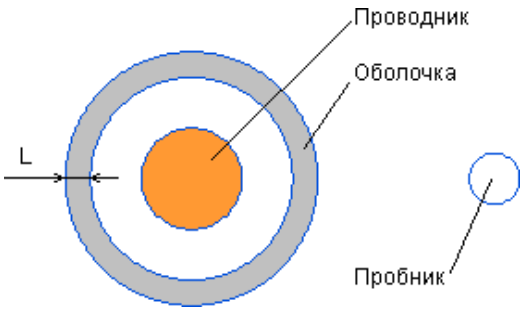
Введение.

В этом примере шаг за шагом описывается процесс создания документа в Microsoft Word, который взаимодействует с сервером ELCUT и отображает результаты вычислений. На каждом шаге описывается один этап создания документа. После описания каждого шага есть кнопка, нажав которую Вы сможете посмотреть на работу только что описанного кода.

Данный пример подготовлен специально для студенческой версии программы ELCUT. Все, что нужно для работы этого примера, кроме установленного приложения Microsoft Word – это студенческая версия ELCUT.

Основное содержание документа – в макросах, поэтому для нормальной работы Вы должны разрешить запуск макросов.

Описание модели.



Круглый медный проводник радиуса 2 мм расположен в начале координат. По нему идет ток. Величину и частоту тока мы будем менять, но по умолчанию будем считать, что полный ток равен 100 А при частоте 50 Гц.

Вокруг проводника на некотором расстоянии находится цилиндрический экран. Внутренний радиус экрана – 4 мм. Мы будем исследовать зависимость поля, индуцируемого проводником от толщины экрана и от его физических свойств (электрической проводимости и магнитной проницаемости). Интервал изменения толщины экрана – от 0.5 мм до 5.5 мм.

Также мы сравним результаты, получаемые для экрана с замкнутыми концами (нулевое напряжение), экрана с разомкнутыми концами (нулевой полный ток) и вообще без экрана.

Для измерения характеристик индуцированного поля на некотором расстоянии от экрана мы поместим пробник (зонд) – стальной стержень, радиуса 1 мм. Расстояние между центрами проводника и пробника равно 12 мм.

Измерять мы будем три основные величины:

- Потенциал пробника
- Плотность магнитной индукции
- Мощность тепловыделения в проводнике и экране

Для измерения напряжения и магнитной индукции возьмем точку вблизи поверхности пробника (точку с координатами (0.0112, 0)).

Всю построенную модель ограничим окружностью радиуса 10 см. На этой окружности зададим граничное условие Дирихле (A=0). Все свободное пространство между проводником, экраном и пробником заполнено воздухом.

Описанная модель соответствует задаче магнитного поля переменных токов в ELCUT.

Для того, чтобы проблема могла быть решена с использованием студенческой версии ELCUT, нужно уменьшить количество узлов сетки конечных элементов. Для этого достаточно задать следующие интервалы разбиения: на внешней границе экрана – 2 мм; на внешней границе пробника – 1 мм; на ограничивающей модель окружности – 5 см.

Описанная модель изображена на рисунке выше (ее центральная часть, без ограничивающей окружности).

Первый шаг: Создание задачи ELCUT.

Это первый шаг создания документа MS Word, использующего технологию ActiveField. Удобно запускать решение задачи в ELCUT нажатием кнопки, поэтому можно начать с добавления кнопки в наш документ. Для этого отобразите панель инструментов “Control Toolbox” и нажмите в ней кнопку “Command button”. Кнопка вставится в Ваш документ. Можно изменить текст, который отображается на ней.

Запуск встроенного Visual Basic

Чтобы редактировать программу на VBA, которая запускается при нажатии на кнопку, щелкните дважды на ней, если Вы находитесь в Design Mode, либо нажмите Alt+F11 для вызова редактора Microsoft Visual Basic. При нажатии на кнопку, которую Вы добавили в документ, будет выполняться процедура

```
Private Sub CommandButton1_Click()

End Sub
```

Подключение библиотеки объектов ELCUT

Первое, что Вы должны сделать, это подключить ELCUT Object Library. Для этого выберите пункт меню Tools -> References и в появившемся диалоге пометьте пункт “ELCUT 5.1 Object Library”. Если данная строка отсутствует в списке, значит ELCUT не установлен, или неправильно установлен на Вашем компьютере.

Запуск ELCUT

Теперь можно написать простой код, который запускает ELCUT в качестве сервера и отображает на экране его окно.

```
    Dim QF As New ELCUT.Application
    ELC.MainWindow.Visible = True
```

Создание новой задачи и геометрической модели

Далее создайте новую проблему и модель и проверьте, что они действительно создаются.

```
    Dim Prb As ELCUT.Problem
    Dim Mdl As ELCUT.Model
    Set Prb = ELC.Problems.Add
    Set Mdl = ELC.Models.Add
```

Если теперь нажать кнопку дважды, не закрывая ELCUT, то Вы получите по два экземпляра проблемы и модели. Чтобы этого избежать, можно сначала закрыть все открытые в ELCUT документы, а уже потом создавать новые. Для этого добавьте в программу следующие строки.

```
    ELC.Problems.Close
    ELC.Models.Close
```

Настройка свойств задачи

Теперь для завершения создания проблемы осталось инициализировать ее свойства и сохранить файл проблемы. Обратите внимание, что все файлы будут сохраняться в текущем каталоге, в котором находится данный документ. Вы можете указать полный путь, в этом случае не забудьте указывать полный путь и далее (и в свойствах проблемы, и при сохранении модели и данных).

```
    ' Define problem parameters
    Prb.Class = qfPlaneParallel
    Prb.ProblemType = qfTimeHarmonicMagnetics
    Prb.Frequency = 50
    Prb.Coordinates = qfCartesian
    Prb.LengthUnits = qfMeters
    Prb.ReferencedFile(qfModelFile) = Path & "\Conduct.mod"
    Prb.ReferencedFile(qfDataFile) = Path & "\Conduct.dhe"
    ' Save the problem
    Prb.SaveAs Path & "\Conduct.pbm"
```

Здесь мы установили тип задачи – «магнитное поле переменных токов» в плоскопараллельной постановке, использовать будем декартовы координаты и единицы измерения длины – метр. Пока мы не будем изменять частоту тока в проводнике, поэтому определили фиксированную величину частоты – 50 Гц.

Скопируем создание проблемы в отдельную процедуру `CreateProblem`, чтобы ее удобно было вызывать и на следующих шагах примера. Переменные же `ELC`, `Prb` и `Mdl` удобно сделать глобальными.

Второй шаг: Рисуем геометрическую модель.

Создание модели также перенесем в отдельную процедуру `DrawModel`.

Добавление геометрических объектов

Начнем рисовать модель.

```
    ' рисуем проводник
    Mdl.Shapes.AddEdge ELC.PointXY(0, 0.002), _
        ELC.PointXY(0, -0.002), 3.1416
    Mdl.Shapes.AddEdge ELC.PointXY(0, -0.002), _
        ELC.PointXY(0, 0.002), 3.1416
```

Последние строки добавляют к модели две дуги по $\pi \approx 3.1416$ радиан, то есть рисуют целую окружность радиуса 0.002 м.

Функции `AddEdge` требуют в качестве аргумента объекты класса `Point`. Правильный способ получения этого объекта – методы `PointXY` или `PointRA` объекта `Application`. Не забывайте указывать этот объект (в нашем примере он имеет имя `ELC`.) при вызове методов.

Аналогично нарисуем окружности внутреннего и внешнего радиуса экрана, пробника и ограничивающую окружность.

```
    ' рисуем экран
    Mdl.Shapes.AddEdge ELC.PointXY(0, 0.004), _
        ELC.PointXY(0, -0.004), 3.1416
    Mdl.Shapes.AddEdge ELC.PointXY(0, -0.004), _
        ELC.PointXY(0, 0.004), 3.1416
    Mdl.Shapes.AddEdge ELC.PointXY(0, 0.005), _
        ELC.PointXY(0, -0.005), 3.1416
    Mdl.Shapes.AddEdge ELC.PointXY(0, -0.005), _
        ELC.PointXY(0, 0.005), 3.1416
    ' рисуем пробник
    Mdl.Shapes.AddEdge ELC.PointXY(0.012, 0.001), _
        ELC.PointXY(0.012, -0.001), 3.1416
    Mdl.Shapes.AddEdge ELC.PointXY(0.012, -0.001), _
        ELC.PointXY(0.012, 0.001), 3.1416
    ' рисуем внешнюю границу
    Mdl.Shapes.AddEdge ELC.PointXY(0, 0.1),
        ELC.PointXY(0, -0.1), 3.1416
    Mdl.Shapes.AddEdge ELC.PointXY(0, -0.1), _
        ELC.PointXY(0, 0.1), 3.1416
```

Настройка масштаба в окне модели

Чтобы посмотреть на получившуюся модель, нужно отобразить окно модели. Можно также распахнуть это окно.

```
    Dim Win As ELCUT.ModelWindow

    ' рыскрываем окно до максимального размера
    Set Win = Mdl.Windows(1)
    Win.WindowState = qfMaximized
```

Теперь можно выбрать удобный масштаб для отображения модели. Для этого существует метод `Zoom`. Обратите внимание, что метода `Zoom` нет в классе `Window`, поэтому объект `Win` должен быть именно класса `ModelWindow`.

Однако, нас интересует не вся модель, а ее центральная часть с проводником и пробником. Для выбора удобного масштаба можно указать параметры в методе `Zoom`, но проще вызвать его без параметров перед добавлением ограничивающей

окружности. Соответственно, переменная `Win` должна быть объявлена и инициализирована выше этого места.

```
    ' установка масштаба
    Win.Zoom
```

Сохранение файла модели

Наконец, сохраним файл модели.

```
    Mdl.SaveAs Path & "\Conductor.mod"
```

В данном документе после каждого шага есть кнопка, позволяющая запустить код, созданный на этом шаге. Например, кнопка после этого, второго, шага имеет следующую процедуру.

```
Private Sub CommandButton2_Click()
    CreateProblem
    DrawModel
End Sub
```

Если Вы следуете инструкциям в этом документе и создаете свой собственный документ, взаимодействующий с сервером ELCUT, Вы можете не добавлять новых кнопок, а исправить код у существующей кнопки.

Третий шаг: Генерируем сетку конечных элементов и помечаем геометрические объекты.

Чтобы решить проблему в ELCUT, сперва нужно построить сетку конечных элементов. Вы можете запустить код, созданный на втором шаге примера, и затем в окне ELCUT выполнить команду "Построить сетку во всех блоках". Если у Вас установлена профессиональная версия ELCUT, сетка должна построиться. Если же Вы обладаете студенческой версией, построить сетку Вы не сможете, так как будет превышено ограничение на максимальное число узлов (200).

Управление густотой сетки

Уменьшить количество узлов в данной задаче нетрудно. Приведенный ниже код устанавливает необходимые размеры треугольников в сетке. Без задания этих интервалов ELCUT по умолчанию построит в данной задаче сетку из 3049 элементов, при установке интервалов в сетке будет лишь 125 элементов.

```
    ' Увеличиваем шаг сетки, чтобы вписаться в 200-узловое
ограничение
    Mdl.Shapes.Vertices.Nearest _
        (ELC.PointXY(0, 0.1)).Spacing = 0.05
    Mdl.Shapes.Vertices.Nearest _
        (ELC.PointXY(0, -0.1)).Spacing = 0.05
    Mdl.Shapes.Vertices.Nearest _
        (ELC.PointXY(0, 0.005)).Spacing = 0.002
    Mdl.Shapes.Vertices.Nearest _
        (ELC.PointXY(0, -0.005)).Spacing = 0.002
    Mdl.Shapes.Vertices.Nearest _
        (ELC.PointXY(0.012, 0.001)).Spacing = 0.001
    Mdl.Shapes.Vertices.Nearest _
        (ELC.PointXY(0.012, -0.001)).Spacing = 0.001
```

Можно устанавливать интервал разбиения не для вершин, а для ребер или блоков. В этом случае код будет немного короче.

Построение сетки

Теперь и в студенческой версии ELCUT можно построить сетку.

```
    ' построение сетки
    Mdl.Shapes.BuildMesh
```

Наконец, еще одна операция, которую осталось проделать с геометрической моделью, – это присвоить метки тем блокам, ребрам и вершинам, у которых мы хотим задать физические свойства.

Присвоение меток

Пометить надо все блоки.

```
    Mdl.Shapes.Blocks.Nearest(ELC.PointXY(0, 0)).Label = _
        "Conductor"
    Mdl.Shapes.Blocks.Nearest(ELC.PointXY(0, 0.003)).Label = "Air"
    Mdl.Shapes.Blocks.Nearest(ELC.PointXY(0, 0.0045)).Label = _
        "Shield"
    Mdl.Shapes.Blocks.Nearest(ELC.PointXY(0, 0.01)).Label = "Air"
    Mdl.Shapes.Blocks.Nearest(ELC.PointXY(0.012, 0)).Label = _
        "Tester"
```

И еще пометим два ребра – ограничивающую окружность, чтобы можно было задать на ней нулевое граничное условие Дирихле.

```
    Mdl.Shapes.Edges.Nearest(ELC.PointXY(0.1, 0)).Label = "Bound"
    Mdl.Shapes.Edges.Nearest(ELC.PointXY(-0.1, 0)).Label = "Bound"
```

Теперь можно сохранить модель. (Имя ей было присвоено на предыдущем шаге.)

```
    Mdl.Save
```

Третий шаг

Четвертый шаг: Создаем и метки и устанавливаем физические свойства.

Создание новых меток

Физические свойства материалов хранятся в ELCUT в отдельном файле данных. Проблема содержит ссылку на этот файл. На первом шаге в свойствах проблемы мы записали, что файл данных у нас называется "Conductor.dhe". Метод `Labels` у класса `Problem` позволяет получить коллекцию меток из файла данных, связанного с задачей, не открывая файл данных отдельно.

```
Prb.Labels(qfBlock).Add.Name = "Air"
Prb.Labels(qfBlock).Add.Name = "Conductor"
Prb.Labels(qfBlock).Add.Name = "Shield"
Prb.Labels(qfBlock).Add.Name = "Tester"
Prb.Labels(qfBlock).Add.Name = "Bound"

Prb.Save
```

Определение физических свойств метки

При изменении физических свойств материалов не обойтись без дополнительных переменных.

```
Dim Lab As ELCUT.Label
Dim LaBlHE As ELCUT.LabelBlockHE
Dim LaEdHE As ELCUT.LabelEdgeHE
```

Для каждой метки блока теперь нужно написать примерно следующий фрагмент кода.

```
' свойства проводника
Set Lab = Data.Labels(qfBlock)("Conductor")
Set LaBlHE = Lab.Content
LaBlHE.Kxx = 1
LaBlHE.Kyy = 1
LaBlHE.Conductivity = 60000000
LaBlHE.Loading = 100
LaBlHE.TotalCurrent = True
Lab.Content = LaBlHE
```

Не забудьте выполнить последнее присваивание, обновляя содержимое объекта `Lab`.

Физические свойства метки ребра

Для метки ребер внешней границы напишем следующий код.

```
' свойства ребра внешней границы
Set Lab = Data.Labels(qfEdge)("Bound")
Set LaEdHE = Lab.Content
LaEdHE.Dirichlet = 0
Lab.Content = LaEdHE
```

Четвертый шаг

Пятый шаг: Решение задачи и анализ результатов.

Решение задачи

Метод `SolveProblem` решает проблему, а `AnalyzeResults` – открывает окно картины поля.

```
' решаем задачи
Prb.SolveProblem
If Prb.Solved Then
    ' открываем окно картины поля
    Prb.AnalyzeResults
End If
```

Манипулирование картиной поля

Первое, что мы сделаем – установим интересующие нас параметры отображения картины поля. В качестве параметров методу `Zoom` передаем границы интересующего нас прямоугольника, содержащего проводник с экраном и пробник. А наблюдать будем цветовую картину напряжения.

Так же, как и при редактировании физических свойств, не забудьте выполнить последнее присваивание, применяя определенные параметры к окну `Win`.

```
Dim Win As ELCUT.FieldWindow
Dim PS As ELCUT.FieldPicture

Set Win = Prb.Result.Windows(1)
Win.Zoom ELC.PointXY(-0.006, -0.005), _
    ELC.PointXY(0.013, 0.005)
Set PS = Win.PictureSettings
PS.ColorMap = qfVoltage
PS.Equilines = False
Win.PictureSettings = PS
```

Вычисление локальных полевых величин

Объявим переменные, в которых будем сохранять вычисленные значения напряжения, индукции и мощности.

```
Dim V As Double, B As Double, P As Double
```

Далее, получим интересующие нас локальные параметры в точке у поверхности пробника. Для этого создается объект класса `FieldPointHE`, связанный с определенной точкой и затем, уже у этого объекта получаем нужные локальные характеристики поля в точке – напряжение и плотность магнитной индукции.

```
Dim FP As ELCUT.FieldPointHE

Set FP = Prb.Result.GetLocalValues(ELC.PointXY(0.0112, 0))
V = FP.ElectroPotential.R
B = FP.FluxDensity.RMS
```

Вычисление интегральных характеристик поля

Еще нас интересует мощность тепловыделения в проводнике и экране. Это интегральная характеристика. Для вычисления интеграла мы должны создать контур интегрирования.

Соответствующий объект класса `Contour` можно получить через объект окна постпроцессора. Очищаем этот контур и добавляем к нему блоки проводника и экрана.

Далее вычисляем значение интеграла (типа `qfInt_Power`) по контуру.

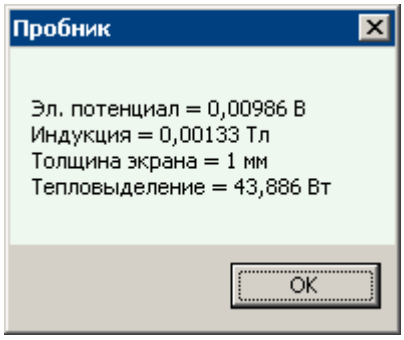
```
        Dim Cnt As ELCUT.Contour

        Set Cnt = Win.Contour
        Cnt.Delete
        Cnt.AddBlock "Conductor"
        Cnt.AddBlock "Shield"
        P = Prb.Result.GetIntegral(qfInt_Power, Cnt)
```

Выводим окно результатов

На следующем шаге мы вставим вычисленные значения прямо в этот документ, а пока для отображения вычисленных значений используем обычный Message Box.

```
    MsgBox "Potential = " & Format(V, "0.00000") & " V"
               & vbCrLf & _
               "Flux density = " & Format(B, "0.00000") & " T" _
               & vbCrLf & _
               "Shield thickness = 1 mm" & vbCrLf & _
              "Joule heat = " & Format(P, "0.000") & " W", , _
             "Conductor tester"
```

В результате выполнения получим следующее окно сообщения



.

Пятый шаг

Шестой шаг: Варьируем параметры задачи.

Мы хотим изучить, как зависит поле от размеров и физических свойств экрана. Также хотелось бы изменять величину и частоту тока в проводнике, использовать замкнутый или разомкнутый экран. Величина и частота тока будет задаваться в данном тексте, в процессе вычислений они меняться не будут.

Ввод параметров проводника

Величина тока в проводнике: 100 А.

Частота тока: 50 Гц.

Считываем данные из документа

Чтобы узнать (использовать в программе на VBA) значения, введенные в этих Edit Boxes, можно написать следующий нехитрый код.

```
Private Sub GetParameters()
    Dim I As Double, f As Double
    Dim TB As TextBox
    ' вводим значение тока
    Set TB = TextBox1
    If IsNumeric(TB.Value) Then
        I = CDbl(TB.Value)
    Else
        I = 100
    End If
    ' вводим значение частоты
    Set TB = TextBox2
    If IsNumeric(TB.Value) Then
        f = CDbl(TB.Value)
    Else
        f = 50
    End If
End Sub
```

Готовимся к изменению параметров

Результаты мы будем отображать в таблицах. Столбцы таблицы будут соответствовать замкнутому экрану, разомкнутому экрану и отсутствию экрана. Первая группа таблиц показывает зависимость поля от толщины экрана для медного и стального экранов, следующая группа таблиц – зависимость от проводимости экрана при фиксированном размере экрана, и последняя – от магнитной проницаемости экрана также при фиксированном размере.

Наблюдать мы будем три величины:

1. напряжение в пробнике – стальном проводнике, расположенном на некотором расстоянии от проводника с током, за экраном;
2. величину магнитной индукции в некоторой фиксированной точке пробника;
3. мощность тепловыделения проводника с экраном (тепловыделением пробника можно пренебречь).

На предыдущих шагах никаких параметров не было, поэтому скопируем процедуры `CreateProblem`, `DrawModel`, `MeshAndLabel`, `DefineData` и `SolveAndGetResult` и назовем новые копии соответственно.

```
Private Sub CreateProblem6()
Private Sub DrawModel6()
Private Sub MeshAndLabel6()
Private Sub DefineData6(Optional OpenEnd As Boolean = False, _
                        Optional Cond As Double = 7000000, _
                        Optional Perm As Double = 1000)
Private Sub SolveAndGetResult6(V As Double, B As Double, _
                               P As Double)
```

Варьируем параметры

В скопированных процедурах нужно сделать следующие изменения.

1. Из процедуры `CreateProblem6` уберем строчку, отображающую основное окно ELCUT.
2. Переменные `I` и `f` из процедуры `GetParameters` сделаем глобальными и частоту тока в `CreateProblem6` установим равной не 50 Гц, а `f`.
3. Закрытие всех старых моделей перенесем из процедуры `CreateProblem6` в `DrawModel6`.
4. Добавим в `DrawModel6` код, который закрывает окно картины поля, если оно открыто.

```
    If Not Prb.Result Is Nothing Then Prb.Result.Close
```

5. Уберем из процедуры `DrawModel6` код, который распахивает окно и выбирает масштаб изображения.
6. Исправим построение геометрической модели так, чтобы внешний радиус экрана был равен не 0.005, а `-(0.004 + L)`, то есть толщина экрана равна `L`.

```
...
    Mdl.Shapes.AddEdge ELC.PointXY(0, 0.004 + L), _
        ELC.PointXY(0, -(0.004 + L)), 3.1416
    Mdl.Shapes.AddEdge ELC.PointXY(0, -(0.004 + L)), _
        ELC.PointXY(0, 0.004 + L), 3.1416
...
```

7. Исправим функцию `MeshAndLabel6`, чтобы она соответствовала новой геометрии модели.

```
    ' увеличиваем шаг сетки для сокращения числа узлов
    Mdl.Shapes.Edges.Nearest(ELC.PointXY(0.1, 0)).Spacing = 0.05
    Mdl.Shapes.Edges.Nearest(ELC.PointXY(0.004 + L, 0)).Spacing =
        0.002
    Mdl.Shapes.Edges.Nearest(ELC.PointXY(0.013, 0)).Spacing =
0.001
    ' помечаем геометрические объекты
...
    Mdl.Shapes.Blocks.Nearest _
        (ELC.PointXY(0, 0.004 + L / 2)).Label = "Shield"
...
```

8. У процедуры `DefineData` появилось три параметра – это параметры экрана. Код, устанавливающий физические свойства экрана будет тогда выглядеть так.

```
    Set Lab = Data.Labels(qfBlock)("Shield")
    Set LaBlHE = Lab.Content
    LaBlHE.Kxx = Perm
    LaBlHE.Kyy = Perm
    LaBlHE.Conductivity = Cond
    LaBlHE.Loading = 0
```

```
    LaBlHE.TotalCurrent = OpenEnd
    Lab.Content = LaBlHE
```

9. В процедуре `SolveAndGetResult6` локальные переменные `V`, `B` и `P` стали параметрами процедуры – выходными параметрами.

10. Уберем за ненадобностью из процедуры `SolveAndGetResult6` код, изменяющий настройки окна постпроцессора.

Заносим результаты расчета в таблицы

Теперь нужно итерировать параметры экрана и вставлять вычисляемые значения в таблицы. Удобнее не создавать таблицы программно, а подготовить их заранее. В этом примере таблицы уже подготовлены, и их можно увидеть в разделе Таблицы результатов расчета.

Процедура `SaveValues` вставляет в таблицы результат решения одной проблемы. Например, если `Table = 2` (вторая группа таблиц – 4, 5, 6), `Row = 4`, `Col = 3`, то в ячейку (4, 3) будет вставлено: в таблицу 4 – значение `V`, в таблицу 5 – `B`, в таблицу 6 – `P`.

```
Private Sub SaveValues(Table As Integer, _
                       Row As Integer, Col As Integer, _
                       V As Double, B As Double, P As Double)
    Tables((Table - 1) * 3 + 1).Cell(Row, Col).Select
    Selection.Text = Format(V, "0.0000")
    Tables((Table - 1) * 3 + 2).Cell(Row, Col).Select
    Selection.Text = Format(B, "0.0000")
    Tables((Table - 1) * 3 + 3).Cell(Row, Col).Select
    Selection.Text = Format(P, "0.0000")
End Sub
```

Процедура `SaveValuesAir` вставляет в таблицы посчитанные значения при отсутствии экрана.

Основная процедура `CommandButton6_Click` относительно длинная, но простая. В ней создается проблема, а затем в цикле – определяются параметры экрана, строится модель, сетка и данные, посчитанные значения вставляются в таблицы. Если Вы имеете минимальный опыт программирования на Visual Basic, то легко напишете ее самостоятельно.

Таблицы результатов расчета.

Зависимость от толщины экрана.

Таблица 1. Потенциал пробника в зависимости от толщины экрана, В.

Толщина экрана	Замкнутый экран		Разомкнутый экран		Без экрана
	Медь	Сталь	Медь	Сталь	
0.0005	0,0234	0,0225	0,0243	0,0246	0,0247
0.0015	0,0184	0,0139	0,0244	0,0248	
0.0025	0,0132	0,0139	0,0243	0,0248	
0.0035	0,0096	0,0044	0,0243	0,0248	
0.0045	0,0074	0,0045	0,0243	0,0249	
0.0055	0,0058	0,0013	0,0246	0,0250	

Таблица 2. Магнитная индукция на пробнике в зависимости от толщины экрана, Тл.

Толщина экрана	Замкнутый экран		Разомкнутый экран		Без экрана
	Медь	Сталь	Медь	Сталь	
0.0005	0,0016	0,0015	0,0016	0,0017	0,0016
0.0015	0,0012	0,0009	0,0016	0,0016	
0.0025	0,0009	0,0010	0,0016	0,0017	
0.0035	0,0006	0,0003	0,0016	0,0017	
0.0045	0,0005	0,0003	0,0016	0,0017	
0.0055	0,0004	0,0001	0,0016	0,0017	

Таблица 3. Джоулевы потери в проводнике в зависимости от толщины экрана, Вт.

Толщина экрана	Замкнутый экран		Разомкнутый экран		Без экрана
	Медь	Сталь	Медь	Сталь	
0.0005	8,4760	38,5649	8,0201	16,4560	7,3669
0.0015	8,2271	54,9133	7,3692	107,0704	
0.0025	8,1569	36,1935	7,3753	101,8157	
0.0035	7,9831	52,5270	7,3865	78,2893	
0.0045	7,8583	44,4936	7,4038	60,5583	
0.0055	7,7671	54,8806	7,4269	73,4684	

Зависимость от проводимости экрана.

Таблица 4. Потенциал пробника в зависимости от электропроводности экрана, В.

Проводимость экрана	Замкнутый экран	Разомкнутый экран	Без экрана
0.7e+7	0,0157	0,0247	0,0247
1.7e+7	0,0130	0,0247	
2.7e+7	0,0125	0,0247	
3.7e+7	0,0123	0,0247	
4.7e+7	0,0122	0,0247	
5.7e+7	0,0122	0,0247	
6.7e+7	0,0121	0,0247	

Таблица 5. Магнитная индукция на пробнике в зависимости от электропроводности экрана, Тл.

Проводимость экрана	Замкнутый экран	Разомкнутый экран	Без экрана
0.7e+7	0,0011	0,0017	0,0016
1.7e+7	0,0009	0,0017	
2.7e+7	0,0008	0,0017	
3.7e+7	0,0008	0,0017	
4.7e+7	0,0008	0,0017	
5.7e+7	0,0008	0,0017	
6.7e+7	0,0008	0,0017	

Таблица 6. Джоулевы потери в проводнике в зависимости от электропроводности экрана, Вт.

Проводимость экрана	Замкнутый экран	Разомкнутый экран	Без экрана
0.7e+7	66,9930	63,1161	7,3669
1.7e+7	37,5861	78,5154	
2.7e+7	27,1636	67,9756	
3.7e+7	22,1261	57,0526	
4.7e+7	19,1839	48,7498	
5.7e+7	17,2615	42,5995	
6.7e+7	15,9094	37,9590	

Зависимость от магнитной проницаемости экрана.

Таблица 7. Потенциал пробника в зависимости от магнитной проницаемости экрана, В.

Магнитная проницаемость экрана	Замкнутый экран	Разомкнутый экран	Без экрана
1	0,0243	0,0244	0,0247
3.16	0,0244	0,0244	
10	0,0245	0,0245	
31.6	0,0244	0,0246	
100	0,0241	0,0246	
316	0,0217	0,0247	
1000	0,0157	0,0247	

Таблица 8. Магнитная индукция на пробнике в зависимости от магнитной проницаемости экрана, Т.

Магнитная проницаемость экрана	Замкнутый экран	Разомкнутый экран	Без экрана
1	0,0016	0,0016	0,0016
3.16	0,0016	0,0016	
10	0,0016	0,0016	
31.6	0,0017	0,0017	
100	0,0016	0,0017	
316	0,0015	0,0017	
1000	0,0011	0,0017	

Таблица 9. Джоулевы потери в проводнике в зависимости от магнитной проницаемости экрана, W.

Магнитная проницаемость экрана	Замкнутый экран	Разомкнутый экран	Без экрана
1	8,1386	8,0198	7,3669
3.16	8,1612	8,0204	
10	8,2483	8,0265	
31.6	8,6815	8,0872	
100	11,5158	8,6929	
316	29,0121	14,6183	
1000	66,9930	63,1161	

Дополнительный шаг: Картины поля и графики.

Возможно, у Вас уже возникло желание сделать более наглядное представление результатов. ELCUT позволяет посмотреть и импортировать картины поля, а также построить различные графики для одной проблемы.

На этом шаге мы вставим в этот документ две картины поля (напряженность и полный ток). Для нашей задачи интерес представляют графики зависимости физических величин от толщины экрана.

Удаление старых картинок из документа

Каждое вставленное в документ изображение будет являться объектом `InlineShape`. Перед тем, как вставлять новые картинки, нужно удалить старые. Чтобы не удалить лишнего (например, кнопки – тоже `InlineShapes`), каждой "нашей" картинке будем присваивать отличительный `AlternativeText`, по которому будем отличать ее от других.

```
    Dim IShp As InlineShape

    ' Delete old pictures
    For Each IShp In InlineShapes
        If Len(IShp.AlternativeText) > 9 Then
            If Left(IShp.AlternativeText, 10) = "My Picture" _
                Then IShp.Delete
        End If
    Next IShp
```

Готовимся к вставке картинки

Далее, установим удобный масштаб картины поля.

```
    Dim Win As ELCUT.FieldWindow

    ' Set properly postprocessor window scale
    Set Win = Prb.Result.Windows(1)
    Win.Zoom ELC.PointXY(-0.006, -0.005), ELC.PointXY(0.013,
0.005)
```

Наконец, устанавливаем параметры картины поля.

```
    Dim PS As ELCUT.FieldPicture

    ' Copy strength picture
    Set PS = Win.PictureSettings
    PS.ColorMap = qfKGrad
    PS.ColorGrades = 200
    PS.Equilines = False
    Win.PictureSettings = PS
```

Копируем изображение

Копируем изображение в буфер обмена. Для этого в классе `FieldWindow` существует метод `GetPicture`.

```
    Win.GetPicture
```

Указываем место для вставки

Чтобы вставить картинку из буфера обмена в документ Word, нужно установить курсор (выделение) в то место документа, куда нужно вставить картинку. Удобно заранее установить в нужном месте закладку и установить курсор по этой закладке.

```
    Selection.GoTo What:=wdGoToBookmark, Name:="Picture1"
```

Вставляем и форматируем изображение

Наконец, можно вставить изображение и изменить его размер.

```
    Selection.Paste
    Selection.MoveLeft Extend:=wdExtend
    With Selection.InlineShapes(1)
        .Width = .Width * 0.7
        .Height = .Height * 0.7
        .AlternativeText = "My Picture"
    End With
```

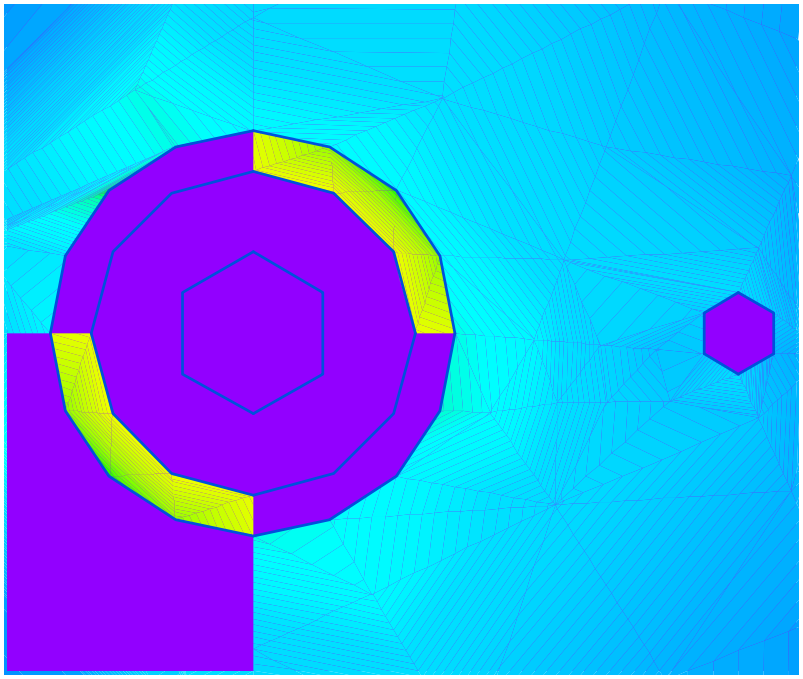
Вторая картинка вставляется аналогично.

.

Картинки поля

Рассчитанные картины поля.

Напряженность электрического поля



Плотность тока в проводниках