

# **PRADIS**

## **ОПИСАНИЕ ПАСПОРТА ЭЛЕМЕНТА PRADIS**

**ПРОГРАММНЫЙ КОМПЛЕКС ДЛЯ АВТОМАТИЗАЦИИ  
МОДЕЛИРОВАНИЯ НЕСТАЦИОНАРНЫХ ПРОЦЕССОВ В  
МЕХАНИЧЕСКИХ СИСТЕМАХ И СИСТЕМАХ ИНОЙ ФИЗИЧЕСКОЙ  
ПРИРОДЫ**

**ВЕРСИЯ 4.4**

## Содержание

1. Введение.....	3
2. Общий формат паспорта.....	3
3. Общие теги.....	3
3.1. Ter description.....	3
3.2. Ter field.....	4
3.3. Ter parameter.....	4
3.4. Ter parameterlist.....	4
3.5. Ter worklist.....	4
3.6. Ter statelist.....	5
3.7. Ter node.....	5
3.8. Ter nodelist .....	5
3.9. Ter image2 .....	5
4. Описание элементов.....	6
4.1. Описание моделей.....	6
4.2. Описание ПРВП.....	7
4.3. Описание ПГО.....	7
4.4. Описание объекта.....	8
4.5. Описание модуля.....	8
5. Создание обозначения элемента.....	8
5.1 Создание обозначения в препроцессоре.....	8

## 1. Введение

При создании нового элемента (модели, ПРВП, ПГО) необходимо описать данный элемент с помощью паспорта элемента. Данный паспорт будет использоваться решателем при формировании топологии схемы, препроцессором для описания элемента, по паспорту генерируется документация пользователя. С помощью функций системного каталога всегда можно получить доступ к паспорту включенных элементов. Поэтому паспорт должен выполняться тщательно и не содержать ошибок.

В данном документе описывается только формат паспорта. Включение паспорта в системный каталог, работа с моделями описывается в документах по утилитам armdoc, parm, arm.

Важно следующее: утилита parm (для моделей на Python) вызывает утилиту armdoc автоматически, утилита arm этого не делает. Таким образом, модели на языках Фортран или C++ требуют ручного добавления паспорта в каталог.

Для параметров, узлов, ПГО и объектов паспорт добавляется также вручную пользователем.

## 2. Общий формат паспорта

Для описания паспорта используется язык XML. Таким образом, формат паспорта подчиняется правилам этого языка. Элементы описания представлены в виде тегов с атрибутами. Используются вложенные теги. Все теги имеют открывающий тег и закрывающий.

Любой паспорт начинается с одинакового заголовка:

```
<?xml version="1.0" encoding="Windows-1251"?>
```

Далее следует тег описания элемента.

Тег должен иметь закрывающий тег. А именно, если открывающий тег <тег>, то закрывающий будет </тег>.

Для проверки правильности формата паспорта с точки зрения XML достаточно загрузить его в Internet Explorer. Если формат некорректен, то IE укажет строку и столбец ошибки.

В случае простого тега, т.е. внутри тега нет никаких дополнительных тегов, можно использовать форму, в которой тег сразу закрывается: <тег/>.

## 3. Общие теги

Во всех паспортах используются ряд тегов для описания узлов, параметров, полей и т.д.. Рассмотрим подробно эти теги.

### 3.1. Ter description.

Тег description используется для описания элемента паспорта (узла, параметра, поля, модуля, модели и т.д.) на русском и английском языках. Его вид:

```
<description>
  <russian> Здесь описание на русском      </russian>
  <english> Here is English description    </english>
</description>
```

В дальнейшем мы будем показывать этот тег в сокращенном виде: <описание элемента>. Но это надо понимать как сложный тег, показанный выше. Данный тег используется при генерации документации для описания элементов и генерации диалогов в препроцессоре.

### 3.2. Ter field

Используется для описания полей параметров, узлов или объектов. Формат:

```
<field name = "имя" type = "тип">  
  <описание поля>  
</field>
```

#### Ter fieldlist

Набор полей описывается тегом fieldlist. Его формат:

```
< fieldlist >  
  <поле1>  
  <полеN>  
</ fieldlist >
```

### 3.3. Ter parameter

Тег используется для описания параметра. Его вид:

```
<parameter name="имя параметра" type="тип параметра"  
default="значение по умолчанию">  
  <описание параметра>  
</parameter>
```

В теге задается имя параметра, его тип и значение по умолчанию. Значение по умолчанию – необязательный параметр.

В дальнейшем будем использовать сокращенное описание параметра: <параметр>.

Тег parameter может использоваться в двух значениях: при описании параметра какого-либо элемента и при описании структуры параметра. Данное описание тега показано для первого случая. Причем тип параметра должен содержать имя модуля, т.е. формат: “модуль.параметр”. Если параметр из модуля base, то указывать модуль необязательно.

При описании структуры параметра, т.е. нового типа параметра, тег parameter следует сразу за заголовочным тегом паспорта. Формат:

```
<parameter name = "имя типа" module = "имя модуля">  
  <описание типа параметра>  
  <список полей>  
</parameter>
```

Пример описания параметра смотрите в файле examples/armdocnew/base/pXYZ.xml.

### 3.4. Ter parameterlist

С помощью тега описывается список параметров. Формат:

```
<parameterlist>  
  <параметр1>  
  <параметрN>  
</parameterlist>
```

Никаких атрибутов у тега нет.

### 3.5. Ter worklist

С помощью тега описывается список рабочих переменных. Формат:

```
< worklist >  
  <параметр1>  
  <параметрN>
```

```
</ worklist >
```

Никаких атрибутов у тега нет. В описании рабочей переменной (тег параметр) не требуется задавать значение по умолчанию.

### 3.6.Ter statelist

С помощью тега описывается список рабочих переменных. Формат:

```
< statelist >  
  <параметр1>  
  <параметрN>  
</ statelist >
```

Никаких атрибутов у тега нет. В описании рабочей переменной (тег параметр) не требуется задавать значение по умолчанию.

### 3.7.Ter node

Тег описывает узел элемента. Его формат:

```
<node name = "имя узла" type = "тип узла" >  
  <описание узла>  
</node>
```

Тег node может использоваться как для описания узла в элементе, так и для описания нового типа узла. Выше приведено описание для узла для описания в элементе. Причем тип узла должен содержать имя модуля, т.е. формат: "модуль.узел". Если узел из модуля base, то указывать модуль необязательно.

При описании нового типа узла формат следующий:

```
< node name = "имя типа" module = "имя модуля">  
  <описание типа узла>  
  <список полей>  
</ node >
```

Пример описания узла смотрите в файле /examples/armdocnew/base/XYZ.xml.

### 3.8.Ter nodelist

Тег описывает список узлов в элементе. Его формат:

```
< nodelist >  
  <узел1>  
  <узелN>  
</ nodelist >
```

### 3.9.Ter image2

С помощью данного тега описывается схемное обозначение элемента. Его формат:

```
<image2 icon = "имя иконки" symbol = "описание символа" />
```

Здесь icon – имя файла иконки без расширения. Формат имени иконки: "имя модуля.имя элемента". При добавлении с помощью утилиты armdoc и createimage созданная иконка автоматически запишется в папку /docs/HTML/sysarm/image под именем имя модуля.имя элемента.png. Поэтому важно корректно указать данные параметры. В дальнейшем иконка используется при отображении документации и в препроцессоре.

Атрибут `symbol` содержит описание графических примитивов, которые рисуют обозначение. Все графические примитивы разделяются между собой знаком “:”. По данному описанию утилита `createimage` создает иконку и препроцессор рисует элемент в схеме. Возможны следующие графические примитивы:

- линия
- текст
- дуга
- эллипс
- прямоугольник
- эллипс заполненный
- прямоугольник заполненный
- стрелка
- порт

Формат графических примитивов здесь не приводится, поскольку описание обозначения можно создавать в редакторе и генерировать в файл.

Каким образом можно генерировать текст для атрибута `symbol` рассмотрим ниже в главе Создание обозначения элемента.

## 4. Описание элементов

### 4.1. Описание моделей

Тег `model` описывает заголовок модели и следует сразу за заголовком XML файла:

```
<model name = "имя" module = "имя модуля" alias = "псевдоним"
image = "имя ПГО по умолчанию" ext= "" ent = "" par = "" ign = ""
adr = "" wrk = "" str="" vpr = "" stp = "" wrp = "" >
  <описание модели>
  <список узлов>
  <список параметров>
  <список рабочих переменных>
  <список переменных состояния>
  <схемное обозначение>
</model>
```

Описание атрибутов:

- `name` – имя модели. Данное имя будет использоваться в препроцессоре;
- `module` – имя модуля модели;
- `alias` – псевдоним. Данное имя будет использоваться в PSL задании. Здесь требуется написать имя модели в старом каталоге PRADIS;
- `image` – имя ПГО по умолчанию для данной модели;
- `ext` – число внешних степеней свободы (не узлов!);
- `ent` – число внутренних степеней свободы (не узлов!);
- `par` – число параметров модели (одночисловых параметров, а не тех которые описаны в списке параметров!, т.е уже число развернутых полей параметров);
- `ign` – признак игнорирования якобиана;
- `adr` – признак начала якобиана;
- `wrk` – число рабочих переменных;
- `str` – число переменных состояния;
- `vpr` - признак переменного числа параметров;
- `stp` – коэффициент соответствия переменных состояния числу параметров;
- `wrp` – коэффициент соответствия рабочих переменных числу параметров.

Пример модели можно увидеть в файле `/examples/armdocnew/base/FSIN.xml`.

## 4.2. Описание ПРВП

Тег `ovr` описывает заголовок ПРВП и следует сразу за заголовком XML файла:

```
<ovr name = "имя" module = "имя модуля" alias = "псевдоним" out=
"" sys= "" par = "" wrk = "" vpr = "" vps = "" wrp = "" wrs = "">
  <описание модели>
  <список узлов>
  <список параметров>
  <список рабочих переменных>
  <схемное обозначение>
</ovr>
```

Описание атрибутов:

- `name` – имя модели. Данное имя будет использоваться в препроцессоре;
- `module` – имя модуля модели;
- `alias` – псевдоним. Данное имя будет использоваться в PSL задании. Здесь требуется написать имя модели в старом каталоге PRADIS;
- `out` – число выходных переменных;
- `sys` – число входных переменных;
- `par` – число параметров модели (одночисловых параметров, а не тех которые описаны в списке параметров!, т.е уже число развернутых полей параметров);
- `wrk` – число рабочих переменных;
- `vpr` - признак переменного числа параметров;
- `vps` – коэффициент соответствия рабочих переменных числу входных переменных;
- `wrp` – коэффициент соответствия рабочих переменных числу параметров.

Пример ПРВП можно увидеть в файле `/examples/armdocnew/base/V.xml`.

## 4.3. Описание ПГО

Тег `image` описывает заголовок ПГО и следует сразу за заголовком XML файла:

```
<image name = "имя" module = "имя модуля" alias = "псевдоним"
ext= "" par = "" wrk = "" vpr = "" wrp = "" unv = "" wrs = "" vps =
"">
  <описание модели>
  <список узлов>
  <список параметров>
  <схемное обозначение>
</image>
```

Описание атрибутов:

- `name` – имя модели. Данное имя будет использоваться в препроцессоре;
- `module` – имя модуля модели;
- `alias` – псевдоним. Данное имя будет использоваться в PSL задании. Здесь требуется написать имя модели в старом каталоге PRADIS;
- `ext` – число степеней свободы (не узлов!);
- `par` – число параметров модели (одночисловых параметров, а не тех которые описаны в списке параметров!, т.е уже число развернутых полей параметров);
- `wrk` – число рабочих переменных;
- `unv` –
- `vpr` - признак переменного числа параметров;
- `vps` –
- `wrs` –
- `wrp` – коэффициент соответствия рабочих переменных числу параметров.

Пример ПГО можно увидеть в файле `/examples/armdocnew/Images/ELP3D.xml`.

#### 4.4. Описание объекта

Тег `object` описывает объект и следует сразу за заголовком XML файла:

```
<object name = "имя" module = "имя модуля" >
  <описание модели>
  <список узлов>
  <список параметров>
  <список полей>
  <схемное обозначение>
</ object >
```

Описание атрибутов:

- `name` – имя модели. Данное имя будет использоваться в препроцессоре;
- `module` – имя модуля модели.

Пример объекта можно увидеть в файле `/examples/armdocnew/base/SHTERM.xml`.

#### 4.5. Описание модуля

Тег `<module>` описывает модуль и следует сразу за заголовком XML файла:

```
<module name = "имя модуля" >
  <описание модуля>
  <список моделей>
  <список ПРВП>
  <список ПГО>
  <список объектов>
  <список типов полей>
  <список типов параметров>
  <список типов узлов>
</module>
```

Впрочем, для пользователя, создающего новый модуль, нет необходимости создавать списки моделей, ПРВП, ПГО и т.д. Все дополнительные списки будут создаваться по ходу добавления в модуль элементов с помощью утилиты `armdoc`. Т.е. начальный файл пустого модуля должен быть следующим:

```
<module name = "имя модуля" >
  <описание модуля>
</module>
```

Важно, что в препроцессоре модули будут располагаться в порядке добавления в системный каталог. То же самое происходит с элементами модуля.

### 5. Создание обозначения элемента

Самый простой способ создать обозначения элемента – это воспользоваться режимом редактирования обозначения схемы в препроцессоре. Общий порядок создания обозначения можно описать следующим образом:

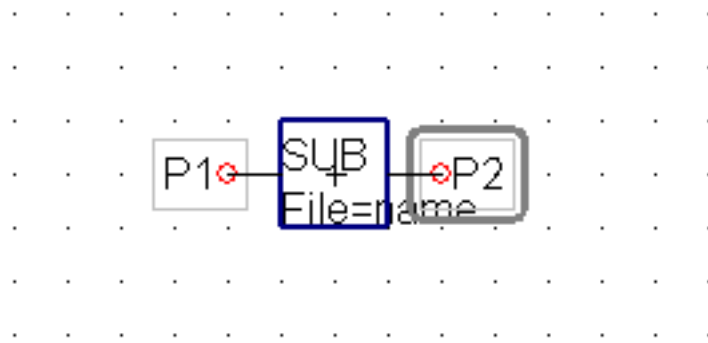
- создание обозначения с помощью графических примитивов в препроцессоре для какой-либо схемы;
- сохранение схемы в файл;
- копирование из файла схемы содержимого тега `Symbol`;
- замена угловых скобок на двоеточия;
- вставка полученного текста в паспорт элемента в атрибут `symbol` тега `image2`.

#### 5.1 Создание обозначения в препроцессоре

1. Создадим пустую новую схему.



2. Добавим в нее порты по числу внешних узлов модели.
3. Введем имена для каждого порта и его номер.
4. Перейдем в режим изменения обозначения схемы.



5. С помощью графических примитивов рисуем требуемое обозначение
6. Сохраняем его в файл
7. Открываем файл в текстовом редакторе
8. Находим тег <Symbol> и </Symbol>. Например:

```
<Symbol>
  <.ID -20 -16 SUB>
  <Line -20 20 40 0 #000080 2 1>
  <Line 20 20 0 -40 #000080 2 1>
  <Line -20 -20 40 0 #000080 2 1>
  <Line -20 20 0 -40 #000080 2 1>
  <.PortSym -40 0 1 0>
  <.PortSym 40 0 2 180>
  <Line -40 0 20 0 #000000 0 1>
  <Line 20 0 20 0 #000000 0 1>
</Symbol>
```

9. Копируем отдельно строки между тегами Symbol
10. Удаляем все открывающие угловые скобки "<"
11. Заменяем все закрывающие угловые скобки ">" на символ ":". Получили текст вида:

```
.ID -20 -16 SUB:
Line -20 20 40 0 #000080 2 1:
Line 20 20 0 -40 #000080 2 1:
Line -20 -20 40 0 #000080 2 1:
Line -20 20 0 -40 #000080 2 1:
.PortSym -40 0 1 0:
.PortSym 40 0 2 180:
Line -40 0 20 0 #000000 0 1:
Line 20 0 20 0 #000000 0 1
```

12. Копируем данный текст и вставляем его в значение атрибута symbol тега image2 элемента
13. Сохраняем паспорт элемента
14. Добавляем паспорт в каталог с помощью команды "armdoc -a имя\_файла\_паспорта"
15. Если все в порядке, то в папке /docs/HTML/sysarm/image/ появится файл с именем формата модуль.имя.png, который будет содержать иконку символа.
16. Можно проверить что получилось, открыв помощь по данной модели. После псевдонима элемента должна быть полученная картинка.

Для лучшего визуального восприятия типа узла необходимо при создании элементов маркировать цветом выводы элементов согласно следующему стандарту:

- DOF1 - темно-синий #000080
- Point - зеленый #00FF00
- Point2d – синий #0000FF
- XY - светло-синий #0080FF
- XYZ - светло-зеленый #00FF80