

# DPF Version 7 Format Description

---

*DPF v7*

*DMS Reference US-DP7X-TB-01-EN-A*

*Ucamco Software  
March 2009*



Copyright

© Copyright Ucamco NV, Gent, Belgium

All rights reserved. This material, information and instructions for use contained herein are the property of Ucamco. The material, information and instructions are provided on an AS IS basis without warranty of any kind. There are no warranties granted or extended by this document. Furthermore Ucamco does not warrant, guarantee or make any representations regarding the use, or the results of the use of the software or the information contained herein. Ucamco shall not be liable for any direct, indirect, consequential or incidental damages arising out of the use or inability to use the software or the information contained herein.

The information contained herein is subject to change without prior notice. Revisions may be issued from time to time to advise of such changes and/or additions.

No part of this document may be reproduced, stored in a data base or retrieval system, or published, in any form or in any way, electronically, mechanically, by print, photoprint, microfilm or any other means without prior written permission from Ucamco.

This document supersedes all previous dated versions.

Trademarks

All product names cited are trademarks or registered trademarks of their respective owners. Correspondence regarding this publication should be forwarded to:

Ucamco NV  
 Bijenstraat 19  
 B-9051 Gent  
 Belgium

For more information

Our web site: <http://www.ucamco.com>

E-mail: [info@ucamco.com](mailto:info@ucamco.com)

Helpdesk

<p>Europe, Middle East &amp; Africa</p> <p>Customer Support for                  Plotters, Software and Mania/Maniabarco AOI</p> <p>Monday - Thursday:                  09.00 AM - 6.00 PM MET</p> <p>Friday:                  09.00 AM - 6.00 PM MET</p> <p>☎ + 32 9 216 99 00</p>	<p>Asia, Pacific, North, South &amp; Latin America</p> <p>Please contact our business partner for your country during support working hours or see <a href="#">contacts</a> page on our website</p>
<p>General support issues: <a href="mailto:support.eur@ucamco.com">support.eur@ucamco.com</a></p> <p>License issues: <a href="mailto:license.eur@ucamco.com">license.eur@ucamco.com</a></p> <p>Java™ HyperTool issues: <a href="mailto:hypertool@ucamco.com">hypertool@ucamco.com</a></p> <p>General information: <a href="mailto:info@ucamco.com">info@ucamco.com</a></p> <p>Sales issues: <a href="mailto:sales@ucamco.co">sales@ucamco.co</a></p>	

# Table of Contents

---

<b>Table of Contents</b> .....	<b>iv</b>
<b>1 Introduction</b> .....	<b>7</b>
1.1 Ucamco's DPF version 7 .....	7
1.2 About This Document.....	7
1.3 Other Available Documentation.....	8
<b>2 General Information</b> .....	<b>9</b>
2.1 DPF Syntax Properties.....	9
2.2 DPF File Structure.....	9
<b>3 HEADER</b> .....	<b>10</b>
3.1 Syntax .....	10
3.2 Description.....	10
3.3 Example.....	10
<b>4 UNIT &amp; DIMENSION</b> .....	<b>11</b>
4.1 Syntax .....	11
4.2 Description.....	11
4.3 Example.....	11
<b>5 IDENTIFICATION</b> .....	<b>12</b>
5.1 Syntax .....	12
5.2 Description.....	12
5.3 Example.....	12
<b>6 OBJECT ATTRIBUTES TABLE</b> .....	<b>13</b>
6.1 Syntax .....	13
6.2 Description.....	13
6.3 Example.....	13
<b>7 TRUE OBJECTS DEFINITION</b> .....	<b>15</b>
7.1 Syntax .....	15
7.2 Description.....	15
7.3 Example.....	15

<b>8 LAYER INFO .....</b>	<b>16</b>
8.1 Syntax .....	16
8.2 Description .....	16
8.3 Example .....	16
<b>9 REFERENCE POINTS .....</b>	<b>17</b>
9.1 Syntax .....	17
9.2 Description .....	17
9.3 Example .....	17
<b>10 NETLIST TABLE .....</b>	<b>18</b>
10.1 Syntax .....	18
10.2 Description .....	18
10.3 Example .....	18
<b>11 DATA .....</b>	<b>19</b>
11.1 Syntax .....	19
11.2 Example .....	19
<b>12 Unit 20</b>	
12.1 Syntax .....	20
12.2 Description .....	20
12.3 Example .....	20
<b>13 Aperture Definitions .....</b>	<b>21</b>
13.1 Syntax .....	21
13.2 Description .....	21
13.3 Aperture Shapes .....	21
13.3.1 Circle .....	21
13.3.2 Rectangle .....	22
13.3.3 Square .....	22
13.3.4 Donut .....	22
13.3.5 Box .....	23
13.3.6 Octagon .....	24
13.3.7 Thermal .....	24
13.3.8 Text .....	25
13.3.9 Contour .....	25
13.3.10 Complex .....	25
13.3.11 Block .....	26
<b>14 Aperture Options .....</b>	<b>27</b>

14.1 Mirroring.....	27
14.2 Rotation.....	27
14.3 Name.....	28
14.4 Aperture Attributes.....	28
14.5 Scaling.....	28
14.6 Reverse.....	28
14.7 Pattern.....	29
<b>15 Netlist Number .....</b>	<b>30</b>
15.1 Syntax.....	30
15.2 Description.....	30
15.3 Example.....	30
<b>16 True Object Reference .....</b>	<b>31</b>
16.1 Syntax.....	31
16.2 Description.....	31
16.3 Example.....	31
<b>17 Object Attribute Reference.....</b>	<b>32</b>
17.1 Syntax.....	32
17.2 Description.....	32
17.3 Example.....	32
<b>18 Plot Data.....</b>	<b>33</b>
18.1 Flash.....	33
18.2 Move.....	33
18.3 Draw.....	34
18.4 Arc.....	34
18.5 Vector Text.....	34
<b>19 DPF Example.....</b>	<b>37</b>
19.1 Layer in Ucam - Image.....	37
19.2 Layer in Ucam – DPF Syntax.....	37

# 1 Introduction

---

## 1.1 Ucamco's DPF version 7

This document describes the **DPF version 7** syntax. DPF stands for the **Dynamic Process Format**. The DPF information is part of Ucamco's JOB database structure. Each JOB contains references to one or more DPF files.

DPF is a data format, developed by Ucamco, to represent layer information of a **Printed Circuit Board**. This format not only describes the **image** of the layer such as pads, tracks, holes, power and ground planes but also **electrical netlist information** as well as additional **product information** via attributes.

Developed specially for the Electronics Manufacturing industry, DPF offers a variety of powerful features such as embedded aperture definitions, reverse objects, contour for outline description and block apertures to represent Step & Repeat items.





## 1.2 About This Document

This Guide is aimed at people who want to know more about the syntax of the DPF format. The following conventions are used in this document:

Character(s)	Usage
[ ]	<b>Square brackets:</b> indicate optional file elements. They are not part of the actual format.
< >	<b>Angular brackets:</b> indicate fixed file elements. They are not part of the actual format.
{ }	<b>Braces:</b> indicate that the options stated are not fixed and may be substituted for other formats. The options enclosed in braces can be separated by a vertical line  . Both the braces and the vertical line are not part of the actual format.
EOL	Indicates the <b>end of a line</b> , this is either a carriage return or a linefeed character. UCAM normally skips end of line characters, but when explicitly required, it is mentioned in the syntax description.
SPACE	Indicates an explicit space character. UCAM normally skips space characters, but when explicitly required, it is mentioned in the syntax description.

Character(s)	Usage
*	<b>Asterisk:</b> separates the fixed part from the optional part of a string of alphabetical characters. IN*CH means that you can use both IN and INCH. The asterisk is not part of the actual format.
<b>Bold text</b>	Quotes elements used in the syntax. Bold Text is normally used in the description and is not part of the actual format.

All other items should be taken as literal unless described otherwise.

 <b>Note:</b>	Provides essential extra information.
 <b>Tip:</b>	Provides useful extra information.
 <b>Example:</b>	Contains examples of file syntax, commands, settings, etc.
 <b>Warning:</b>	Contains an important warning.

## 1.3 Other Available Documentation


Ucamco offers its customers a wide range of documentation on all its software - including installation requirements and procedures, release information, detailed task descriptions and info on customer support availability.

If you need more info please mail us at [info@ucamco.com](mailto:info@ucamco.com)

## 2 General Information

---

### 2.1 DPF Syntax Properties

- ❑ **Coordinate System:** DPF uses a right hand coordinate system: 
- ❑ **Case sensitivity:** DPF is case insensitive.
- ❑ **Separators:** Separators are only required between two consecutive keywords.
- ❑ **Keyword delimiters:** Keywords stop at the first non-alphabetic character (space, line feed or carriage return) or at the end of a data block.

### 2.2 DPF File Structure

A DPF file consists of a number of sections

HEADER  
[OBJECT ATTRIBUTES TABLE]  
[TRUE OBJECTS DEFINITION]  
[LAYER INFO]  
[REFERENCE POINTS]  
[NETLIST TABLE]  
[DATA]

All sections will be described in detail in the following chapters

The OBJECT ATTRIBUTES TABLE section is handled in Chapter 7

The TRUE OBJECTS DEFINITION section is handled in Chapter 8

The LAYER INFO section is handled in Chapter 9

The REFERENCE POINTS section is handled in Chapter 10

The NETLIST TABLE section is handled in Chapter 11

The DATA section is handled in Chapter 12

# 3 HEADER

---

The HEADER section provides information about the file.  
Every program capable of saving DPF-files will generate a header.

## 3.1 Syntax

```
UNIT & DIMENSION  
[IDENTIFICATION]
```

## 3.2 Description

The UNIT & DIMENSION section is handled in Chapter 5  
The IDENTIFICATION section is handled in Chapter 6

## 3.3 Example

```
U=MIL X36.802,1791.338Y78.74,2854.075  
;;GeneratedBy : Ucam v7.1-4  
;;Hostname : bgws99  
;;User : wbi  
;;Date : Thu Nov 10 12:09:50 2005  
;;Digest :  
d2d1c9d7c1f3e7f6efe9e6f3e0e5ff89818c88848d86f6f0e6a9bfb8a1fdb3b  
1a6b7
```

## 4 UNIT & DIMENSION

---

This section sets the unit in which the coordinates of the DPF file are defined. The unit remains valid until overruled by a **Unit** sub-section. It also specifies the outside dimension of the data described in the DPF file.

### 4.1 Syntax

```
U={MM|IN*CH|MIL}<SPACE>Xminx,maxxYminy,maxy<EOL>
```

### 4.2 Description

**U** sets the unit in

- **MM**: millimeter, metric
- **INCH**: inch, imperial
- **MIL**: mil, thousandth of an inch (default)

**Xminx,maxx** and **Yminy,maxy** define the extents of the DPF data to be within

**minx** and **maxx** for the x-range and **miny** and **maxy** for the y-range.

### 4.3 Example

```
U=MIL X36.802,1791.338Y78.74,2854.075
```

# 5 IDENTIFICATION

---

As of Ucam version v7.1-1 an identification section has been introduced. In this section information is stored about the identity of the creator of the DPF job.

## 5.1 Syntax

```
;;GeneratedBy : Programe<EOL>  
;;Version : Version<EOL>  
;;Hostname : Hostname<EOL>  
;;User : UserID<EOL>  
;;Date : FullDate<EOL>  
;;Digest : Digest_String<EOL>
```

## 5.2 Description

- ❑ **Programe** is the name of the program used to generate this DPF file.
- ❑ **Version** is the version of this program.
- ❑ **Hostname** is the name of the used computer.
- ❑ **UserID** is the ID of the user who created or modified the DPF file.
- ❑ **FullDate** is the date and time when the job was created or modified.
- ❑ **Digest\_String** is an encoded string containing all identification parameters.

When loading a DPF file into Ucam the identification section will be ignored.

## 5.3 Example

```
;;GeneratedBy : Ucam v7.1-4  
;;Hostname : bgws99  
;;User : wbi  
;;Date : Thu Nov 10 12:09:50 2005  
;;Digest :  
d2d1c9d7c1f3e7f6efe9e6f3e0e5ff89818c88848d86f6f0e6a9bfb8a1fdb3b  
1a6b7
```

# 6 OBJECT ATTRIBUTES TABLE

---

This section lists all Object Attributes used in this DPF file in 2 consecutive parts:

- The first part contains all Object Attribute names
- The second part contains all possible Object Attribute values.

Check the **Object Attribute Reference** sub-section (Chapter 18) to find out how to assign attributes & values to data objects.

## 6.1 Syntax

```
;$name_index<SPACE>name<EOL>
[
;$name_index<SPACE>name<EOL>
[ ... ]
]
[
;$value_index<SPACE>"value"<EOL>
[
;$value_index<SPACE>"value"<EOL>
[ ... ]
]]
```

## 6.2 Description

Two tables, one which always links a **name\_index** with a **name** and one which always links a **value\_index** with a **value**.

- **name\_index** and **value\_index** are numbers larger then or equal to zero.
- **name**: defines an attribute name and should not contain any special characters.
- **value**: defines an attribute value.

## 6.3 Example

```
;$0 nonplated
;$1 plated
;$2 smd
;#0 "false"
;#1 "true"
```



# 7 TRUE OBJECTS DEFINITION

A True Object is the visible part of a positive object that is partially hidden by overlapping negative data. A flash scratched by negative data will show as one or more True Pads. A draw, an arc or a region scratched by negative data will show as one or more True Tracks. True objects are generated when building a Netlist. The TRUE OBJECTS DEFINITION section contains shape definitions of all True Objects.

## 7.1 Syntax

```
;@{f
;@index<SPACE><Aperture Definition><Netlist
Number> [<EOL>;@<SPACE><SPACE>]<Plot Data> [<Netlist
Number> [<EOL>;@<SPACE><SPACE>]<Plot Data> [...]]
[
;@index<SPACE><Aperture Definition><Netlist
Number> [<EOL>;@<SPACE><SPACE>]<Plot Data> [<Netlist
Number> [<EOL>;@<SPACE><SPACE>]<Plot Data> [...]]
[...]]
]
;@}
```

## 7.2 Description

Each shape definition starts with an index number and is followed by either:

- A flashed Complex aperture.
- A Region (Contour).

All True Objects with the same index are in fact parts of one particular object before it was scratched by negative data.

The **True Object Reference** sub-section describes how this index is used (see Chapter 17).

## 7.3 Example

```
;@{f
;@1 A1=COMPLEX, (M-80.332, 22.967C58.688, -116.053, -14.302, -
50.023, CW D-80.332, 22.967) N1F604.853, 1034.275
;@1 A1=COMPLEX, (M-47.572, 49.185C-
22.137, 115.215, 50.853, 49.185, CW D116.883, -23.805C-
47.572, 49.185, 50.853, 49.185, CW ) N2
;@ F539.698, 935.067
;@2 A6=CONTOUR
N7M787.4, 780.443D826.772, 741.072D, 669.291D898.552D951.857, 615.9
86C885.825, 590.55, 885.825, 688.975, CW D787.4D, 780.443N8
;@
M794.362, 787.4D984.251D, 688.975C958.817, 622.946, 885.825, 688.975
, CW D905.512, 676.251D, 748.031D833.731D794.362, 787.4
;@}
```

# 8 LAYER INFO

---

The LAYER INFO section is only informational and has no effect on the visual data. Information typed in the Info field of the Layer Modify dialog box will be inserted as comment in the DPF file.

## 8.1 Syntax

```
;comment<EOL>[;comment<EOL>[...]]
```

## 8.2 Description

- **comment** is a string containing any character.

The first character in the comment may not be **\$** or **#** as these have a special meaning (see OBJECT ATTRIBUTES TABLE). As described in the syntax, you can have multiple comments as long as each comment starts with a semicolon ( ; ) and ends with an EOL.

## 8.3 Example

```
;This info about a layer.  
;This is more info about this layer.
```

# 9 REFERENCE POINTS

---

This section contains a list of reference points. Reference points are similar to DPF Comments as they contain only informational data and have no effect on the visual data. However, some actions in UCAM depend on these reference points and yield different results when other reference points are defined.

Reference Points may be used for:

- Registration
- Output (e.g. Optical Inspection)

## 9.1 Syntax

```
Pn=refx,refy[ [<EOL>]Pn=refx,refy[ ... ] ]<EOL>
```

## 9.2 Description

Defines reference points with index **n** and position **refx**, **refy**.

- **n** is a positive number.
- **refx** and **refy** are values in the current unit.

## 9.3 Example

```
P0=295.275,2460.63P1=1771.653,2460.63P2=3248.031,2460.63
```

Defines 3 reference points P0, P1 and P2.

# 10 NETLIST TABLE

---

The NETLIST TABLE section is informational data and has no effect on the visual data. Netlist information is used within UCAM for functions that require electrical connection information or to create test fixtures for electrical test machines. While some of these functions do not specifically require netlist information, overall performance is always faster and more accurate when netlist information is available.

When importing IPC-D-356A Ucam will generate a Netlist Table inside the NETLIST TABLE section of the DPF file. Edit this section to add/modify net attributes if required.

## 10.1 Syntax

```
Nn,name="value" [,name="value" [...]]<EOL>
[
Nn,name="value" [,name="value" [...]]<EOL>
[... ]
]
```

## 10.2 Description

- **n**: is the net number for a name-value pair.
- **name**: is a normal ASCII string, no special characters are allowed.
- **value**: is a normal ASCII string. If the value contains special characters (like ; and , ) the value should be declared between double quotes.

For each net **n**, **name** or **value**, pairs can be defined. Within the NETLIST TABLE the net numbers must increment appropriately.

## 10.3 Example

```
N100,name="power",impedance="50Ohm"
N200,name="ground"
```

# 11 DATA

---

The DATA section gathers all the visual information of the DPF. Within this section shapes are used for plotting. Before a shape can be used for plotting it must be defined by means of an aperture definition.

## 11.1 Syntax

```
[<Unit>]<Aperture Definition>[<Netlist Number>][<Plot  
Data>][<Object Attribute Reference>][<True Object Reference>]  
[  
[<Unit>]<Aperture Definition>[<Netlist Number>][<Plot  
Data>][<Object Attribute Reference>][<True Object Reference>]  
[ ... ]  
]
```

## 11.2 Example

```
U=MM A1=CIRCLE,5,ATTR=(ape_attr="1") F15,25;$0=0  
;@1  
U=MIL A2=DONUT,196.85,118.11 N3F885.826,984.251  
A3=RECTANGLE,196.85,118.11,P=LR:19.685:3.937 N4F1181.102;$0=1  
A4=SQUARE,196.85 N5F1476.377
```

# 12 Unit

---

A **Unit** sub-section is used to set the unit in which the following coordinates are defined. Unit is modal, i.e. the unit remains valid until a new Unit is specified.

## 12.1 Syntax

`U={MM|IN*CH|MIL}<SPACE>`

## 12.2 Description

Sets the unit in

- ❑ **MM:** millimeter, metric
- ❑ **INCH:** inch, imperial
- ❑ **MIL:** mil, thousandth of an inch (default)

## 12.3 Example

`U=MM`

# 13 Aperture Definitions

---

An **Aperture Definition** sub-section contains the definition of a created Aperture. Apertures must be defined before they can be used. The definition of an aperture consists of its shape, size and extra options. Shapes and sizes are described in this Chapter, extra options are described in Chapter 15.

## 13.1 Syntax

`An=DEFINITION`

## 13.2 Description

Defines an aperture with aperture number **n**. This aperture number can be any value from zero to one billion. The aperture number does not have to be unique, meaning that apertures are distinguished based on their location in the file. All possibilities for **DEFINITION** are described below.

## 13.3 Aperture Shapes

Regular aperture shapes are:

- Circle
- Rectangle
- Square
- Donut
- Box
- Octagon
- Thermal
- Text
- Contour

Special **composed** aperture shapes are:

- Complex
- Block

### 13.3.1 Circle

Defines a circular aperture, this is the most common aperture used for pads and for drawing tracks with rounded edges. Circular flashes are also commonly used to represent drilling holes.

#### Syntax

`An=C*IRCLE, d[, <APERTURE OPTIONS>]`

#### Description

Defines a circle with:

- **n**: the aperture number

- **d**: the diameter

### Example

```
A10=C, 10  
A11=CIRCLE, 10
```

## 13.3.2 Rectangle

This defines a rectangular aperture, which can be used for placing SMDs or for drawing tracks with rectangular edges.

### Syntax

```
An=R*RECTANGLE, sx, sy[, <APERTURE OPTIONS>]
```

### Description

Defines a rectangle with:

- **n**: the aperture number
- **sx**: the size in the x-direction
- **sy**: the size in the y-direction

### Example

```
A12=R, 100, 50  
A13=RECTANGLE, 100, 50
```

## 13.3.3 Square

A square is a special case of a rectangular aperture, where the x-size is equal to the y-size. It is most commonly used for drawing tracks with square edges.

### Syntax

```
An=S*SQUARE, s[, <APERTURE OPTIONS>]
```

### Description

Defines a square with:

- **n**: the aperture number
- **s**: size of each side.

This is the same as a rectangular aperture where **sx** and **sy** are both equal to **s**.

### Example

```
A14=S, 20  
A15=SQUARE, 20
```

## 13.3.4 Donut

A donut can be used in combination with rectangular apertures to define targets. When used as a reversed flash, it can separate copper. The netlist of the separated copper is the netlist of the donut flash, and not the underlying copper.

### Syntax

```
An=D*ONUT, douter, dinner[, {RR|SS|SR}] [, <APERTURE OPTIONS>]
```

## Description

Defines a donut with:

- **n**: the aperture number
- **douter**: the outer diameter or size of the donut
- **dinner**: the inner diameter or size of the donut

3 different kinds of donut may be defined as follows:

- **RR**: circular outer and circular inner shape (round/round), this is the default
- **SS**: square outer and square inner shape (square/square)
- **SR**: square outer and circular inner shape (square/round)

Where **n** is the aperture number

## Example

```
A16=D,200,50,RR
A17=DONUT,200,50,SR
```

### 13.3.5 Box

A box is mainly used to define SMD-pads. It is a rectangle with featured corners.

## Syntax

```
An=BO*X, sx, sy, {R*OUNDED|S*STRAIGHT|A*ANTIQUUE|C*UT}=xcutoff[:ycutoff] [, <APERTURE OPTIONS>]
```

## Description





Defines a box with

- **n**: the aperture number
- **x**: the x size of the box
- **y**: the y size of the box
- **xcutoff**: defines the x size of the corner
- **ycutoff**: defines the y size of the corner
- **xcutoff**: should be larger than zero and less or equal to half **sx**
- **ycutoff**: should be larger than zero and less or equal to half **sy**



**Note**: When **ycutoff** is omitted, it is defaulted to the value of **xcutoff**.

The type of corner can be one of the following:

- **ROUNDED**: rounded corners e.g.  .  
When **xcutoff** is half **sx** and **ycutoff** is half **sy**, an ellipse is defined.
- **STRAIGHT**: a flattened corner where the edge consists of only one line e.g.  .  
When **xcutoff** is half **sx** and **ycutoff** is half **sy**, a diamond shape is defined.
- **ANTIQUUE**: inward rounded corners e.g. 
- **CUT**: two inwardly perpendicular lines as corners e.g. 

## Examples

A18=BO, 200, 100, S=50:25  
A19=BOX, 200, 100, R=50

### 13.3.6 Octagon

An octagon is a special case of a Box with STRAIGHT corners, where each of the 8 sides has an equal size.

#### Syntax

An=O\*CTAGON, size[, <APERTURE OPTIONS>]

#### Description

Defines an octagon, with

- **n**: the aperture number
- **size**: size of each side.

This is the same size that has to be used for **sx** and **sy** when using a BOX aperture to define the same shape.

#### Example

A20=O, 200  
A21=OCTAGON, 200

### 13.3.7 Thermal

A thermal is a shape that is used as a thermal isolation for pins that are connected to large copper areas. Thermals should always have the reverse polarity of the data used to define copper.

#### Syntax

An=THE\*ERMAL, outer, inner, gap, ngap, startangle, {RR|RS|SS|SR} [, <APERTURE OPTIONS>]

#### Description

Defines a thermal with:

- **n**: the aperture number
- **outer**: the outer diameter or size
- **inner**: the inner diameter or size
- **gap**: the size of the gap (or the size of the remaining copper when used as reverse flash as it should be)
- **ngap**: the number of gaps equally spaced around the thermal
- **startangle**: is the angle of the first gap. Zero means to the right side of the center point

The type of thermal can be one of the following:

- **RR**: round with rounded edges
- **RS**: round with square edges
- **SR**: square with rounded edges
- **SS**: square with square edges

## Example

```
A22=THE,200,180,20,4,45,rr
A23=THERMAL,200,180,20,3,0,rs
```

### 13.3.8 Text

Text is more than mere information, text also represents material (copper on signal layers).

#### Syntax

```
An=T*EXT, ("text_string"), [height[:width]] [, <APERTURE OPTIONS>]
```

#### Description

Defines a text aperture with:

- **n**: the aperture number.
- **text\_string**: the string of text characters that is displayed. As this string is enclosed between double quotes, a double quote inside the string has to be avoided by inserting it twice.
- **height**: the height of the text, when the height is omitted, then the height defaults to 100 mil.
- **width**: the width of the text, when no width is defined, then the width defaults to 84.21% of the text height.

#### Example

```
A24=T, ("Hello"),200:168.42
A25=TEXT, ("Say ""Hello"""),200:200
```

### 13.3.9 Contour

A contour aperture is used to describe large copper areas.

#### Syntax

```
An=CON*TOUR[, ST*ROKE=st] [, <APERTURE OPTIONS>]
```

#### Description

A contour aperture is considered as a pen with size = 0. This aperture is used to draw contour lines. After the selection of a contour aperture, you can add an enumeration of contour lines described by means of move, draw and arc. An overlap of two contour lines is not allowed.

You can add a stroke with size **st**, which results in a contour with a solid line on top of the contour. The stroke value is the width of this line.

#### Example

```
A26=CON
A27=CONTOUR, STROKE=10
```

### 13.3.10 Complex

A complex is used to define shapes that have no primitive in DPF. This way an aperture can be defined with a triangular shape.

#### Syntax

```
An=COM*PLEX, (<contour>[<contour>[ ... ]]) [, <APERTURE OPTIONS>]
```

```
An=COM*PLEX, Ac [, <APERTURE OPTIONS>]
contour:
<MOVE>{<DRAW>|<ARC>} [{<DRAW>|<ARC>} [ ... ]] {<DRAW>|<ARC>}
```

## Description

The first syntax defines a complex using an enumeration of contour descriptions. Each contour is described using the DPF plot commands for move, draw and arc. Each contour starts with a move command, is followed by at least one draw or arc command and ends with a draw or arc to close the contour.

The second syntax defines a complex using the same contour definition as the last complex defined with aperture number = **c**. This is useful for reusing a complex and applying different aperture options.

**n**: the aperture number of the newly created complex.

## Example

```
A28=COM, (M0, 0D100D, 100, D0D, 0)
A29=COMPLEX, A28
```

### 13.3.11 Block

A block contains DPF data that has to be repeated several times.

## Syntax

```
An=B*LOCK, ([COMMENT] [OBJECT ATTRIBUTE TABLE] [TRUE OBJECTS
TABLE] [PLOT DATA]) [, <APERTURE OPTIONS>]
An=B*LOCK, Ab [, <APERTURE OPTIONS>]
```

## Description

The first syntax defines the block that contains the DPF commands between round brackets. These DPF commands can also contain other block definitions.

The second syntax defines a block that has the same definition as the last block defined with aperture number = **b**. This is useful for defining a block with the same definitions but with different aperture options.

**n**: the aperture number of the newly created block.

## Example

```
A30=B, (U=MM A1=C, 0.1M0, 0D5D, 5D0D, 0)
A31=BLOCK, A30
```

# 14 Aperture Options

---

Every aperture in DPF can also have some extra options. These options are associated with the aperture definition and the available options are common for each kind of aperture. In DPF syntax the aperture options are separated by a comma.

- Mirroring
- Rotation
- Name
- Aperture attributes
- Scaling
- Reverse
- Pattern

## 14.1 Mirroring

### Syntax

`M*IRROR={X|Y|XY}`

### Description

- **X**: mirror along the X-axis
- **Y**: mirror along the Y-axis
- **XY**: mirror along the X and Y-axis

### Example

`A10=TEXT, "ABC", 10:8.42, M=X`

**Note:** UCAM Mirrors first and then Rotates the image.

## 14.2 Rotation

### Syntax

`R*OTATION=r`

### Description

**r**: is an angle specified in degrees. A positive value indicates a counterclockwise rotation while a negative value indicates a clockwise rotation.

### Example

`A10=RECTANGLE, 10, 20, R=90`

**Note:** UCAM Mirrors first and then Rotates the image.

## 14.3 Name

### Syntax

`NAME="name"`

### Description

**name**: is the optional name of an aperture

### Example

`A10=CIRCLE,10,NAME="special plated"`

## 14.4 Aperture Attributes

### Syntax

`A*ATTRIBUTES=(name="value" [, name="value" [...]])`

### Description

Associate **name-value** attribute pairs to the aperture.

**name**: may not contain any special characters.

### Example

`10=CIRCLE,10,ATTR=(attr="1",attr2="2")`

## 14.5 Scaling

### Syntax

`S*SCALE=s`

### Description

**s**: is the scaling factor.

Note: UCAM only uses scale factors for block and complex apertures, as they can refer to a previously defined definition. A scaling factor on any other aperture has no use. UCAM automatically adapts the dimensions of the aperture.

### Example

`A10=BLOCK,A9,SCALE=0.2`

## 14.6 Reverse

### Syntax

`RE*VERSE`

### Description

Whenever Reverse is used, the aperture is plotted in reverse, meaning white on black.

## Example

A10=CIRCLE, 10, REVERSE

# 14.7 Pattern









## Syntax

P\*ATTERN+{V|H|VH|L|R|LR|C45|C90}:[step[:width]]

## Description

- **step**: the distance between two pattern elements, the default is 10 mil.
- **width**: the width of a pattern element, the default is 5 mil.

The pattern type can be defined as follows:

-  **L**, Diagonal slant from left.
-  **R**, Diagonal slant from right.
-  **LR**, Diagonal slant from right and left.
-  **V**, Vertical.
-  **H**, Horizontal.
-  **VH**, Horizontal and Vertical.
-  **C45**, 45 degree circular dots (chessboard dot pattern).
-  **C90**, 90 degree circular dots (grid dot pattern).

## Example

A10=CIRCLE, 10, P=LR:1:0.5

# 15 Netlist Number

---

A **Netlist Number** sub-section is used to set the Net number of the Plot Data which follows. Netlist information is informational data and has no effect on the visual data. It is used within UCAM for functions that require electrical connection information or to create test fixtures for electrical test machines. While some of these functions do not specifically require netlist information, overall performance is always faster and more accurate when netlist information is available.

## 15.1 Syntax

<SPACE>Nn

## 15.2 Description

- n: is the net number.

## 15.3 Example

A100=CIRCLE, 50 N1F2300, 3700F1600F, 4200N2F2400, 3700F1700F, 4200

Defines a circle of 50 (current unit). 3 flashes made with this aperture belong to net 1 and 3 flashes made with this aperture belong to net 2.

# 16 True Object Reference

A **True Object Reference** sub-section is used to set a reference to a True Object in the TRUE OBJECTS DEFINITION section (see Chapter 8). True Object References and True Object Definitions are generated when building a Netlist.

## 16.1 Syntax

```
;@index<EOL>
```

## 16.2 Description

**Index** is a reference to an entry in the TRUE OBJECTS DEFINITION section.

## 16.3 Example

TRUE OBJECTS DEFINITION section

```
;@{f
;@1 A1=COMPLEX, (M-80.332,22.967C58.688,-116.053,-14.302,-
50.023,CW D-80.332,22.967) N1F604.853,1034.275
;@1 A1=COMPLEX, (M-47.572,49.185C-
22.137,115.215,50.853,49.185,CW D116.883,-23.805C-
47.572,49.185,50.853,49.185,CW ) N2
;@ F539.698,935.067
;@2 A6=CONTOUR
N7M787.4,780.443D826.772,741.072D,669.291D898.552D951.857,615.9
86C885.825,590.55,885.825,688.975,CW D787.4D,780.443N8
;@
M794.362,787.4D984.251D,688.975C958.817,622.946,885.825,688.975
,CW D905.512,676.251D,748.031D833.731D794.362,787.4
;@}
```

DATA section with True Objects References

```
U=MM A1=CIRCLE,5,ATTR=(ape_attr="1") F15,25;$0=0
;@1
A6=CONTOUR N0M885.825,590.55C984.251,688.975,885.825,688.975;@2
```

;@1 refers to 2 Complexes (Net 1 and Net 2)

;@2 refers to 2 Regions (Net 7 and Net 8)

# 17 Object Attribute Reference

---

Each object in the **Plot Data** can be given an attribute. An attribute always has a name, but the value of the attribute is optional. The name and value of the attribute are not inserted in the **Plot Data**. A reference to the attribute is used. This reference refers to the attribute list that was created in the OBJECT ATTRIBUTES TABLE section (see Chapter 7).

The **Object Attribute Reference** sub-section is used to create a reference to a pre-defined attribute (with or without value) in the OBJECT ATTRIBUTES TABLE.

## 17.1 Syntax

```
;$name_index[=value_index][,name_index[=value_index][  
...]]<EOL>
```

## 17.2 Description

Two tables, one which always links a **name\_index** with a **name** and one which always links a **value\_index** with a **value**.

- ❑ **name\_index** and **value\_index** are numbers larger than or equal to zero.
- ❑ **name**: defines an attribute name and should not contain any special characters.
- ❑ **value**: defines an attribute value.

## 17.3 Example

```
;$0 nonplated  
;$1 plated  
;$2 smd  
;#0 "false"  
;#1 "true"  
  
A100=CIRCLE,50 N25F2300,3700;$0=0,1,2=1  
; Assigns the nonplated object attribute with value "false"  
; Assign the plated object attribute without a value  
; Assigns the smd object attribute with value "true"
```

# 18 Plot Data


---

A **Plot Data** sub-section contains a list of actions performed with a defined aperture.

Possible actions are:

- Flash
- Move
- Draw
- Arc
- Vector Text

For each action a number of parameters and coordinates must be provided.

 **Note:** In DPF, the X and Y values are modal, and therefore they only need to be specified whenever their value actually changes.

## 18.1 Flash

### Syntax

```
[<UNIT>] [<NETLIST NUMBER>] F[x] [, y]
```

### Description

This command inserts a flash at position **x**, **y**, with an optional net number **n** and optional object attributes. The net number is modal and remains for all following plot commands unless modified.

## 18.2 Move

Although a move command is not directly available in UCAM, it constitutes part of a draw or an arc.

### Syntax

```
[<UNIT>] [<NETLIST NUMBER>] M[x] [, y]
```

### Description

Moves to position **x**, **y** without performing any plotting. This is done to mark the start point of a single arc or draw, or to mark the start of a series of draws and arcs.

There is no option for adding attributes here as object attributes are only useful when defined for whole objects.

It is however possible to assign an option net number **n**, and it counts for any draw or arc commands that follow, until modified in the next flash, move or vector text command.

## 18.3 Draw

### Syntax

```
[<UNIT>]D[x] [, y]
```

### Description

Draw from the current position (usually from the previous move, draw or arc command) to position **x**, **y**. There is no net number expected here, as it is only meaningful for the first move command.

It is possible to assign object attributes.

## 18.4 Arc

### Syntax

```
[<UNIT>]C[x] [, y] , mx, my, [{CW|CCW}]
```

### Description

Draws an arc from the current position to position **x**, **y** with center point **mx**, **my**.

**CW**: clockwise arc

**CCW**: counter-clockwise arc (default)

Arcs are usually drawn using a circle aperture. It is however possible to use a contour aperture. In that case the start and end point must be the same so that the arc describes a closed area (circle).

As object attributes are only meaningful on the whole object and this command is only a part of it, there is no option to add these attributes here.

It is possible to assign an optional net number **n**, and it counts for any following draw or arc command, until changed in the next flash, move or vector text command.

## 18.5 Vector Text

### Syntax

```
[<UNIT>]V[x] [, y] , ("text_string") , width[:space] , fontname , (<expanded text>) [VTX OPTIONS]
```

### Description

- ❑ **x / y**: are the coordinates that the text is positioned on. It is the position of the first character of the text defined in **text\_string**.
- ❑ **text\_string**: represents the string of text characters that should be displayed. As this string is enclosed between double quotes, a double quote inside the string has to be avoided by placing it twice.
- ❑ **width**: is the horizontal offset between the starting points of two consecutive characters. In this case the text is mono-spaced and can be used for tabular text.

When **width** is 0 then **space** defines the white space between two characters.

- **fontname**: is the name of the vector font. As no path names are allowed, the font is searched on \$UFNTDIR by default.
- **<expanded text>**: are the actual draws of the text, so that the font file is not required to visualize the DPF file. As vector text consist of multiple characters, it is not meaningful to assign a net number to it.

Optional object attributes can be assigned.

The possible VTX OPTIONS are mirroring, scaling and rotating and are the same as for the DPF Aperture Options.

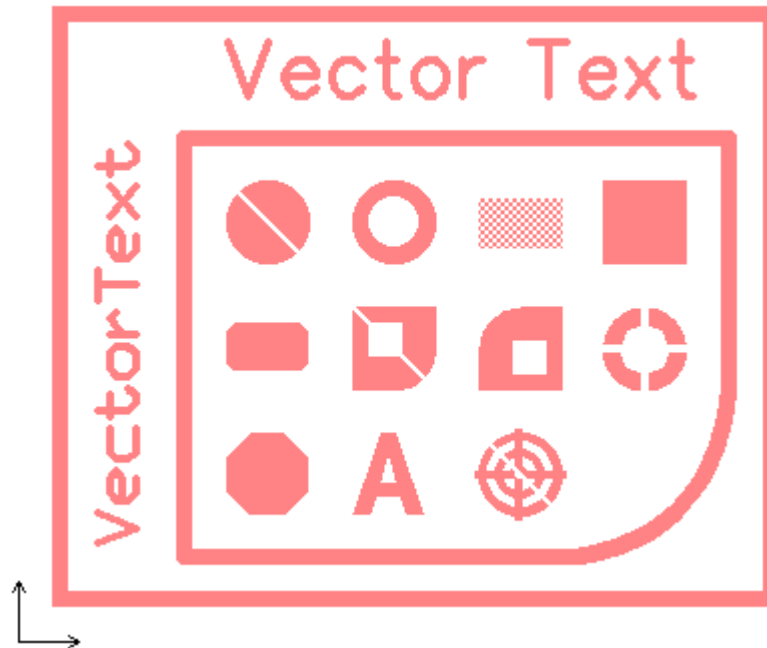
### Example

```
A9=CIRCLE,10 V400,100,("1"),800,vtx,(U=MM  
M0,75,21.75D,075),S=0.1
```



# 19 DPF Example

## 19.1 Layer in Ucam - Image



## 19.2 Layer in Ucam – DPF Syntax

```
U=MIL X36.802,1791.338Y78.74,2854.075
;;GeneratedBy : Ucam v7.1-4
;;Hostname : bgws99
;;User : wbi
;;Date : Thu Nov 10 12:09:50 2005
;;Digest :
d2d1c9d7c1f3e7f6efe9e6f3e0e5ff89818c88848d86f6f0e6a9bfb8a1fdb3b
1a6b7
;$0 my_object_attribute
;#0 "false"
;#1 "true"
;@{f
;@1 A1=COMPLEX, (M-80.332,22.967C58.688,-116.053,-14.302,-
50.023,CW D-80.332,22.967) N1F604.853,1034.275
;@1 A1=COMPLEX, (M-47.572,49.185C-
22.137,115.215,50.853,49.185,CW D116.883,-23.805C-
47.572,49.185,50.853,49.185,CW ) N2
;@ F539.698,935.067
;@2 A6=CONTOUR
N7M787.4,780.443D826.772,741.072D,669.291D898.552D951.857,615.9
86C885.825,590.55,885.825,688.975,CW D787.4D,780.443N8
;@
M794.362,787.4D984.251D,688.975C958.817,622.946,885.825,688.975
,CW D905.512,676.251D,748.031D833.731D794.362,787.4
```

```

;@3 A8=COMPLEX, (M-44.045,-44.045C44.045,44.045,53.887,-
53.887,CW D,4.342C-4.342,-44.045,53.887,-53.887D-44.045)
N10F1422.49,742.863
;@3 A8=COMPLEX, (M-21.094,12.66C27.292,61.046,-
30.937,70.889D66.995C-21.094,-27.043,-30.937,70.889,CW D,12.66)
N11F1507.313,618.087
;@3 A8=COMPLEX, (M-44.045,44.045C44.045,-44.045,-53.887,-
53.887,CW D4.342C-44.045,4.342,-53.887,-53.887D,44.045)
N12F1530.264,742.863
;@3 A8=COMPLEX, (M-32.799,19.787D6.904C55.291,-
28.6,65.133,29.629D,-68.302C-32.799,19.787,65.133,29.629,CW )
N13F1411.244,659.346
;@}
P0=295.275,2460.63P1=1771.653,2460.63P2=3248.031,2460.63
U=MM A1=CIRCLE,5,ATTR=(ape_attr="1") F15,25;$0=0
;@1
U=MIL A2=DONUT,196.85,118.11 N3F885.826,984.251
A3=RECTANGLE,196.85,118.11,P=LR:19.685:3.937 N4F1181.102;$0=1
A4=SQUARE,196.85 N5F1476.377
A5=BOX,196.85,118.11,ROUNDED=29.527:29.527 N6F590.551,688.976
A6=CONTOUR NOM885.825,590.55C984.251,688.975,885.825,688.975;@2
D,787.4D787.4D,590.55D885.825M826.772,748.031D,669.291D905.512D
,748.031D826.772
U=MM A7=COMPLEX, (M0,-2.5C2.5,0,0,0D,2.5D-2.5D,-2.5D0M-
1.5,1.5D,-0.5D0.5D,1.5D-1.5),R=180 N9F30,17.5
A8=THERMAL,5,3,0.5,4,0,RS U=MIL N0F1476.377;@3
U=MM A9=OCTAGON,5 U=MIL N14F590.55,393.7
A10=TEXT, ("A"),196.85:165.768 N15F787.401,295.274
A11=BLOCK, (
;$0 another_attribute
;#0 "123"
A50=CIRCLE,39.37 F0,0
A51=DONUT,118.11,78.74 F
A52=DONUT,196.85,157.48 F
A53=RECTANGLE,216.535,19.685 F
A54=RECTANGLE,19.685,216.535 F
A55=CIRCLE,19.685,REVERSE F;$0=0
) N16F1181.102,393.7
U=MM A20=CIRCLE,1 N17M10,30D,5M42.5,30D10U=MIL
M,196.85D1279.527M1673.228,590.551D,1181.102M1279.527,196.85
C1673.228,590.551,1279.527,590.551
U=MM A21=SQUARE,1 N18M2.5,37.5D,2.5D45D,37.5D2.5
A30=CIRCLE,0.5 U=MIL NOV492.126,1279.527, ("Vector
Text"),0:196.85,vtx, (U=MM
M0.75,21.75D8.75,0.75M16.75,21.75D8.75,0.75M23.25,8.75
D35.25D,10.75D34.25,12.75D33.25,13.75D31.25,14.75D28.25D26.25,1
3.75D24.25,11.75D23.25,8.75D,6.75D24.25,3.75D26.25,1.75D28.25,0
.75
D31.25D33.25,1.75D35.25,3.75M53.75,11.75D51.75,13.75D49.75,14.7
5D46.75D44.75,13.75D42.75,11.75D41.75,8.75D,6.75D42.75,3.75
D44.75,1.75D46.75,0.75D49.75D51.75,1.75D53.75,3.75M63.25,21.75D
,4.75D64.25,1.75D66.25,0.75D68.25M60.25,14.75D67.25M79.75D77.75
,13.75
D75.75,11.75D74.75,8.75D,6.75D75.75,3.75D77.75,1.75D79.75,0.75D
82.75D84.75,1.75D86.75,3.75D87.75,6.75D,8.75D86.75,11.75D84.75,
13.75
D82.75,14.75D79.75M94.25D,0.75M,8.75D95.25,11.75D97.25,13.75D99
.25,14.75D102.25M128.4499,21.75D,0.75M121.4499,21.75D135.4499
M141.9499,8.75D153.9499D,10.75D152.9499,12.75D151.9499,13.75D14
9.9499,14.75D146.9499D144.9499,13.75D142.9499,11.75D141.9499,8.
75

```

D, 6.75D142.9499, 3.75D144.9499, 1.75D146.9499, 0.75D149.9499D151.9  
499, 1.75D153.9499, 3.75M160.4499, 14.75D171.4499, 0.75M, 14.75  
D160.4499, 0.75M180.9499, 21.75D, 4.75D181.9499, 1.75D183.9499, 0.75  
D185.9499M177.9499, 14.75D184.9499), S=0.15 U=MIL V246.644, 266.15  
, ("Vector Text"), 0:68.898, ODBstandard, (M-100, 200D0, -  
100D100, 200M330.898, -100D230.898D180.898, -  
50D, 50D230.898, 100D330.898D380.898, 50  
D330.898, 0D180.898M661.796, 100D511.796D461.796, 50D, -  
50D511.796, -100D661.796M792.694, 200D, -50D842.694, -  
100D892.694D942.694, -50  
M742.694, 100D892.694M1023.592, -  
50D, 50D1073.592, 100D1173.592D1223.592, 50D, -50D1173.592, -  
100D1073.592D1023.592, -50M1304.49, 100D, -100  
M, 0D1404.49, 100D1454.49, 50M1535.388, 200D1735.388M1635.388D, -  
100M1966.286D1866.286D1816.286, -  
50D, 50D1866.286, 100D1966.286D2016.286, 50  
D1966.286, 0D1816.286M2097.184, 100D2297.184, -  
100M2097.184D2297.184, 100M2428.082, 200D, -50D2478.082, -  
100D2528.082D2578.082, -50  
M2378.082, 100D2528.082), R=90, S=0.3  
U=MM A40=CIRCLE, 0.25, REVERSE M12.5, 27.5D32.5, 7.5