Eng-DB-2

A light-weight engineering database

Felix Bertram, fbertram@users.sf.net
Jörn Henneberg, jhenneberg@users.sf.net

GPL

Table of Content

Overview

A key to success in developing consumer products is to properly maintain a library with engineering information throughout the development cycle starting with early concepts and going all the way through to shipping the product. This library should hold:

- components
- approved-vendor-lists
- datasheets and other relevant documentation
- quotations
- design files
- bills of material

There are lots of commercial off-the-shelf PLM solutions available, including but not limited to Arena, Omnify or Windchill. However, these solutions are typically much more aimed at Operations than Engineering.

Engineering's requirements often differ from Operation's requirements; mostly in how to deal with incomplete, preliminary and inaccurate information. In order to maintain productivity in Engineering, the following key requirements have been identified:

- fast and light-weight process for creating parts
- distinction between global library and product-specific parts
- ability to deal with placeholder part-numbers
- fast method for adjusting part information including AVLs and cost
- fast method for importing quotations received from vendors
- fast method for comparing quotations
- full-text search for parts
- simple 'where-used' search

It is well understood, that ultimately the data will probably need to go into a database that is maintained by Operations. This leads to another requirement:

- straightforward way of exporting the data to hand-off to Operations

Prerequisites/ Installing

To run Eng-DB-2 you will need to have the following tools installed:

- Perl
- Spreadsheet::WriteExcel
- Text::CSV

Eng-DB-2 was successfully tested under the following operating systems:

- Mac OS X 10.6.5 … 10.9.5
  with Perl 5.10.0 and Spreadsheet::WriteExcel 2.37
- Windows XP/ SP2 and Windows 7
  with ActivePerl 5.12.2.1202 and Spreadsheet::WriteExcel 2.37

Probably, Eng-DB-2 runs under different systems as well, however this was not tested.

Perl comes bundled with Mac OS X. On Windows, you can download the Community Edition of ActivePerl from the ActiveState website (http://www.activestate.com).

Spreadsheet::WriteExcel and Text::CSV will always need to be installed separately. On Mac OS X, the following sequence of commands will do so:

> *sudo cpan*
> *install Spreadsheet::WriteExcel*
> *install Text::CSV*

On Windows use the following command:

> *perl -MCPAN -e "install 'Spreadsheet::WriteExcel'"*
> *perl -MCPAN -e "install 'Text:CSV'"*

There are multiple ways to install and use the database:

- single install on local storage
  To do so, simply pick a convenient location on your local drive and unpack the database.
- install on shared network drive
  Again, simply pick a convenient location on your local drive and unpack the database.
- install using revision-control system
  Here, the first user picks a location on the local drive and unpacks the database. Next, this folder needs to be committed to the revision-control system. Additional users can then check-out the database to local storage.

First Launch

To launch the database, simply double-click the launcher scripts provided in the “src” directory. You might want to create a shortcut to these launchers in the Windows Start Menu or OSX’s Dock.

Once you have launched the database, you will see a prompt similar to this:

> *=== Eng-DB-2 - a light-weight engineering database ===*
> *version 1.0 (2010nov23)*
>
> *INFO: reading library at 'library'*
> *INFO: loaded 29 parts*
>
> *Eng-DB-2 >*

From this prompt, you can execute various commands. For a list of available commands, either type “help” or any illegal command.

Database structure

Directory structure

The database has the following basic structure:

/eng_db	your database location might differ
/eng_db/doc	folder for documentation
/eng_db/src	folder for source files and launchers
/end_db/library	folder for library. You might for example have your RC4558 op-amp stored at this location: /eng_db/library/electrical/IC/CU-4558-0/
/eng_db/products	folder for products. Your product “A” would have its home folder at /eng_db/products/A and your product “B” would go to /eng_db/products/B

Regarding the structure of the database, there are only very few rules:

- all library parts reside inside the “library” folder. You may create an arbitrary structure inside this folder.
- all product home folders reside directly in the “products” folder, no further nesting is allowed.
- all product-specific parts reside inside the product. You may create an arbitrary directory structure inside the product’s home folder.

Components/ BOMs/ data

The database is created from a large structure of nested folders. Most of these folders have arbitrary names and will only be used to structure your data.

Special text files direct Eng-DB-2 to recognize parts, BOMs or data. The following special text files exist:

- info.txt to hold part information
- data.txt to hold various additional data
- quote.txt to hold quotations
- xref.txt to hold cross-references

Part Information

Whenever an “info.txt” file is encountered, Eng-DB-2 will interpret this folder to hold part information. The folder name will be the part number and the contents of info.txt holds additional information about the part. The following tokens are recognized:

descr	the textural description of a part
rev	the revision number of the part
sku	an entry to the product SKUs; multiple sku lines are supported
avl	an entry for the approved vendor list; multiple avl lines are supported
price	the price for the part. This field is only supported for assemblies.
pricesrc	TODO
rohs	indicates rohs status for the part
bom	indicator that the BOM begins in the next row. This field distinguishes a component from an assembly.
alias	assigns a part number different from the folder name. See section “Dealing with forbidden characters in part numbers”.
flag	TODO

The following example shows the info.txt file for a 555 timer IC. The file is stored at “/eng_db/library/electrical/ICs/CU-555-0”. The use of the “price” and “avl” fields should be self-explanatory. The “rohs” field should be used when sufficient RoHS information for the part exists; this will be indicated in the Excel BOMs later.

descr=IC, Single Precision Timer, SOIC-8
rev=1
price=0.15
pricesrc=
rohs

alias=
avl="Texas Instruments", "NE555D"
avl="Fairchild Semiconductor", "LM555CM"

The price can be either specified in the 'home' currency or in a specific currency. If the price is specified with a currency, then the price will be converted to the home currency by use of a data entry (see section on Data). Examples:

price=1.00	price specified in home currency (which would be US dollar for me)
price=USD 1.00	price specified in US dollar, will be converted to home currency (which is also US dollar for me)
price=$1.00	'$' is equivalent to USD
price=RMB 6.50	price specified in RMB, will be converted to home currency

The next example shows the use of the SKU fields for a product. The file is stored at "/eng_db/products/Timer555"; resulting in the product name "Timer555" being assigned. The product features two separate SKUs that are referred to by two separate "sku" entries. Each SKU will result in a separate tab in the Excel BOM.

descr=Timer 555 circuits
rev=3
sku=Monostable
sku=Astable

In case you had only a single SKU named "Monostable", you could also use the "bom" field to start with the top-level BOM instead of specifying SKUs:

descr=Timer 555 circuits
rev=3
bom
Monostable 1

In this format, a product looks just like any other assembly. After the "bom" field, every line is expected to hold a part number, the quantity and the reference designator. In case there are no reference designators, you may use this field for comments.

Data

Whenever a "data.txt" file is found, Eng-DB-2 will interpret this as a datum that might be used later on. The following tokens are recognized:

descr	the textural description for the datum
value	the numerical value for the datum

Right now, Eng-DB-2 uses these data for currency conversions. In the future, this concept will probably be used for other purposes as well.

This is how RMB are specified, the file is stored at "/eng_db/library/_data_/RMB":

descr=exchange rate RMB -> USD
value=0.149253731343284

Please note that while the description specifies this as the exchange rate from RMB to USD, this is not the full story. It is actually the exchange rate from RMB to the 'home' currency- which happens to be USD for me. For this reason, the following entry at "/eng_db/library/_data_/USD/" is not too surprising:

descr=exchange rate USD -> USD
value=1.0

Quotations

Whenever a "quote.txt" file is found, Eng-DB-2 will interpret this as a list of quotations that will be used when creating BOMs. The following tokens are recognized:

descr	the textural description of the quotation

After the optional descr token come zero or more lines of quotations. Quotations can be given in multiple different formats:

- Without an equal sign and without currency information: this format works best if you cut & paste from an Excel sheet.
- With equal sign and with currency: this format can be used to specify quotes explicitly with the currency.
- With equal sign but without currency

The following example shows how multiple formats can be used:

descr=DigiKey
CC-103-0-06-0 0.06300
CC-105-A-TR-0 = $ 0.03540
CR-102-06-0 = USD 0.04300
CU-555-0 = 0.49000

Cross-references

Whenever an "xref.txt" file is found, Eng-DB-2 will interpret this as a list of cross-references that will be used when creating BOMs. Cross-references can be used for various purposes:

- to hold part numbers from a distributor, e.g. DigiKey. This helps to resolve ambiguities with the part and to procure sample quantities for prototype builds
- to hold part numbers from a CM. This increases the efficiency in communicating with the vendor.

The following tokens are recognized:

descr	the textural description of the cross-reference

After the optional descr token come zero or more lines of cross-references. Each cross-reference consists of a part number followed by the cross-reference part number.

The following example shows how to use this feature:

```
descr=DigiKey
CC-103-0-06-0          445-2664-2-ND
CC-105-A-TR-0          ECA-2AM010B-ND
CR-102-06-0            P1.0KDBTR-ND
CU-555-0               296-6501-1-ND
```

Work flows

Selecting products

The database distinguishes between the global library and the product-specific items. Information residing in the library will be shared across all products and can be looked up at any time. Information residing in a product folder will only be available to that specific product and can only be looked up if that product is the currently selected product. This is useful in order to avoid cluttering the library with temporary information such as placeholder parts or modules.

In order to select a product use the “prod” command, followed by the name of the product. The following example shows how to select the product “IPH”:

```
Eng-DB-2 > prod iph
INFO: reading library at 'library'
INFO: reading library at 'products/IPH'
Eng-DB-2 (IPH) >
```

There are a few things worth noting:

- the product name is not case-sensitive
- the product name is displayed in parentheses after the prompt
- you can unselect the current product by issuing the “prod” command without a product name

When selecting a product, the library will be re-loaded to reflect the part changes. An additional “update” is not required.

Creating parts or assemblies

Creating a part is a very simple process:

- pick a part number. You should have some part numbering scheme, see section on part numbering for a suggestion
- decide if this is a global part or a product-specific part

- for a global part, create a folder inside the library tree and make sure the folder name matches the desired part number
- for a product-specific part, create a folder inside the specific product subtree and make sure the folder name matches the desired part number
- create an info.txt file to hold the part information
- add additional information, e.g. datasheets to the folder created
- if desired, add cross-references to your product-specific xref.txt file
- alternatively, you can create a CSV file, enter the required part information and use the 'import' command to batch-create parts. For more on this see section 'Importing pre-existing data'

Looking up parts

Before you look up parts, make sure the correct product is selected. Then, you can issue the "find" command followed by one or more words or partial words you want to search for. The search is not case sensitive and spans the following fields:

- part number
- part description
- AVL
- cross-reference

A few examples:

Eng-DB-2 (TIMER555) > find res
--- 1 ----------
Part Number: CR-102-06-0
Description: Resistor, 1k Ohm, 0.1W, Metal Film, SMD 0603
Revision: 1
Price: 0.0020
Quote: 0.0430
AVL: Panasonic - ECG [ERA-3AEB102V]
Crossref: P1.0KDBTR-ND

Eng-DB-2 (TIMER555) > find panasonic cap
--- 1 ----------
Part Number: CC-105-A-TR-0
Description: Capacitor, 1uF, 100V, 20%, Electrolytic, through-hole, radial
Revision: 1
Price: 0.0100
Quote: 0.0354
AVL: Panasonic - ECG [ECA-2AM010B]

Crossref: ECA-2AM010B-ND

Opening parts

Often, when you found a part you will want to review the additional information provided for it. As this information is stored in folders in the file system, a method to quickly locate and open that folder is useful. The “open” command does exactly that. It works just like the “find” command, however instead of printing the result of the search, it opens the associated folders.

The following example will open the folder for part CR-102-06-0:

Eng-DB-2 (TIMER555) > open res 1k
INFO: 'open "/Users/felix/Desktop/eng_db/library/electrical/resistors/CR-102-06-0"'

Assembling BOMs

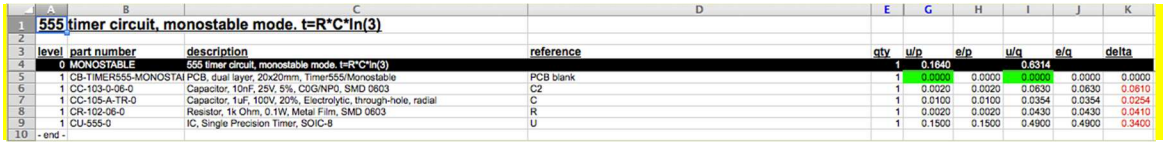
Once all parts and assemblies are created, finished-goods BOMs can be assembled using the “bldbom” command. The command does not take any parameters. While assembling the BOM, the console will show the hierarchy of assemblies. Example:

Eng-DB-2 (TIMER555) > bldbom
MONOSTABLE
ASTABLE

The command creates the following output:

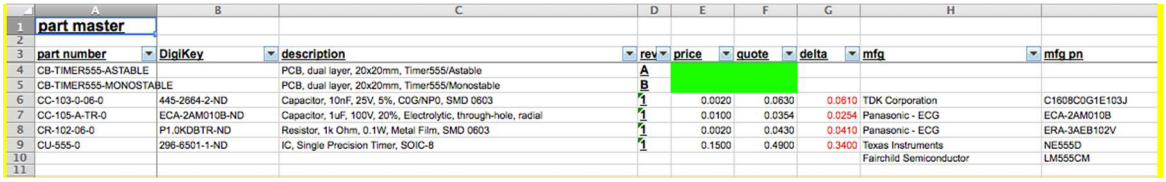
- Excel BOMs (both costed and sanitized) with separate tabs for each SKU and a part-master tab for detailed part information.
- Excel part lists (both costed and sanitized) with separate tabs for each SKU and a part-master tab for detailed part information.
- Text file with the simplified BOM information to allow easy comparison of BOMs.

The following image shows an example for a costed BOM. Every assembly will be colored. The lime-green fields indicate issues, in this case there is neither a price nor a quote given for the PCB fab. The delta column shows what the difference between your target pricing and the current quotation is.



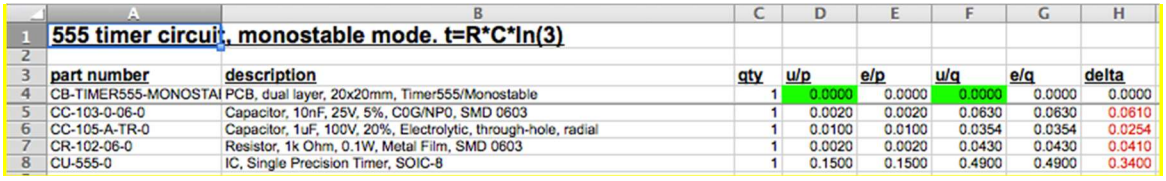
	A	B	C	D	E	G	H	I	J	K
1	555 timer circuit, monostable mode. t=R*C*ln(3)									
2										
3	level	part number	description	reference	qty	u/p	e/p	u/q	e/q	delta
4	0	MONOSTABLE	555 timer circuit, monostable mode. t=R*C*ln(3)		1	0.1640		0.6314		
5	1	CB-TIMER555-MONOSTAI	PCB, dual layer, 20x20mm, Timer555/Monostable	PCB blank	1	0.0000	0.0000	0.0000	0.0000	0.0000
6	1	CC-103-0-06-0	Capacitor, 10nF, 25V, 5%, C0G/NP0, SMD 0603	C2	1	0.0020	0.0020	0.0630	0.0630	0.0610
7	1	CC-105-A-TR-0	Capacitor, 1uF, 100V, 20%, Electrolytic, through-hole, radial	C	1	0.0100	0.0100	0.0354	0.0354	0.0254
8	1	CR-102-06-0	Resistor, 1k Ohm, 0.1W, Metal Film, SMD 0603	R	1	0.0020	0.0020	0.0430	0.0430	0.0410
9	1	CU-555-0	IC, Single Precision Timer, SOIC-8	U	1	0.1500	0.1500	0.4900	0.4900	0.3400
10	- end -									

This is an example of the part-master tab. This tab lists all the parts, their AVLs and cross-references:



	A	B	C	D	E	F	G	H	
1	part master								
2									
3	part number	DigiKey	description	rev	price	quote	delta	mfg	mfg pn
4	CB-TIMER555-ASTABLE		PCB, dual layer, 20x20mm, Timer555/Astable	A					
5	CB-TIMER555-MONOSTABLE		PCB, dual layer, 20x20mm, Timer555/Monostable	B					
6	CC-103-0-06-0	445-2664-2-ND	Capacitor, 10nF, 25V, 5%, C0G/NP0, SMD 0603	1	0.0020	0.0630	0.0610	TDK Corporation	C1608C0G1E103J
7	CC-105-A-TR-0	ECA-2AM010B-ND	Capacitor, 1uF, 100V, 20%, Electrolytic, through-hole, radial	1	0.0100	0.0354	0.0254	Panasonic - ECG	ECA-2AM010B
8	CR-102-06-0	P1.0KDBTR-ND	Resistor, 1k Ohm, 0.1W, Metal Film, SMD 0603	1	0.0020	0.0430	0.0410	Panasonic - ECG	ERA-3AEB102V
9	CU-555-0	296-6501-1-ND	IC, Single Precision Timer, SOIC-8	1	0.1500	0.4900	0.3400	Texas Instruments	NE555D
10								Fairchild Semiconductor	LM555CM
11									

And here is an example of the parts list. In contrast to the BOM which is structured by assemblies, the parts list simply lists all parts and their aggregated quantities for the product. This is exactly what you need to order parts:



	A	B	C	D	E	F	G	H
1	555 timer circuit, monostable mode. t=R*C*ln(3)							
2								
3	part number	description	qty	u/p	e/p	u/q	e/q	delta
4	CB-TIMER555-MONOSTAI	PCB, dual layer, 20x20mm, Timer555/Monostable	1	0.0000	0.0000	0.0000	0.0000	0.0000
5	CC-103-0-06-0	Capacitor, 10nF, 25V, 5%, C0G/NP0, SMD 0603	1	0.0020	0.0020	0.0630	0.0630	0.0610
6	CC-105-A-TR-0	Capacitor, 1uF, 100V, 20%, Electrolytic, through-hole, radial	1	0.0100	0.0100	0.0354	0.0354	0.0254
7	CR-102-06-0	Resistor, 1k Ohm, 0.1W, Metal Film, SMD 0603	1	0.0020	0.0020	0.0430	0.0430	0.0410
8	CU-555-0	IC, Single Precision Timer, SOIC-8	1	0.1500	0.1500	0.4900	0.4900	0.3400

Dealing with prices and quotations

Eng-DB-2 can help you in managing costs by annotating BOMs with costs, see the paragraph about creating BOMs for sample output. Here is a recommended way to work with quotations:

1. prepare a BOM with Eng-DB-2 and create an Excel BOM (bldbom command)
2. send the sanitized Excel BOM to your contract manufacturer and have them quote
3. when the quote comes back, open the sheet in Excel
4. Delete everything but the following columns: part number and unit price
5. Copy the sheet (CTRL-C)
6. Open your quote.txt file; if you don't have one create one next to your product's info.txt
7. Paste your data from above (CTRL-V) into quote.txt
8. Re-run the Excel BOM (bldbom command). Your Excel BOM is now annotated with the quotation!

Typically, you will use the quotation to estimate the total material cost for your product while you are still in development. Therefore, your most-recent BOM will often be out-of-sync with what the CM has quoted on. As the quotations are held in a separate file and keyed by the part number, this is not an issue.

Eng-DB-2 distinguishes between prices and quotations: A price is your target that you reasonably expect to pay for a component. A quotation is what the CM has quoted for that component. Both will often not match. However, it is good practice to have pricing attached to all your components in the database and then drive the quotations towards that target. Often in this process, you will be missing quotes or prices for certain components. Eng-DB-2 deals with this the following way:

- if a component does not have a price, that field will be marked lime green and Eng-DB-2 will carry over the number from the quotation
- if a component does not have a quotation, that field will be marked lime green and Eng-DB-2 will carry over the number from your pricing information

Updating price information

The updprice command allows to update library pricing information from local quotations (quote.txt) or price updates (price.txt).

The updprice command takes the following options:

q	transfer quotations (quote.txt) back to library

p	transfer pricing (price.txt) back to library
h	transfer only those prices which are higher than the library pricing
l	transfer only those prices which are lower than the library pricing
n	transfer pricing for those parts that have no library pricing yet
a	transfer all pricing information regardless of being higher or lower than library pricing
c	ask for confirmation of each price transferred

Here is an example:

```
Eng-DB-2 (CCS) > updprice qac
Part Number: CY-24M576-0
Price:       0.2500
Update:      0.0650
*** update this price? y/n/c:
```

Where-Used search

With eng-db-2, you can create a report where a certain part is used. You can search for part numbers or key words.

The following example shows how to search for components containing the word texas:

```
Eng-DB-2 (CCS) > where texas
loading product 'CCS'...
loading product 'IPH'...
--- 1 ----------
Part Number: CU-4558-0
Description: IC, Operational Amplifier, Dual, GP, 3MHz
Used in:     qty = 2:  CCS [SKID-2.0-US-BRICK]
             qty = 2:  CCS [SKID-2.0-US-BRICK]
```

The report lists the products and SKUs that use the component along with the quantity.

Exporting a product database

Partially implemented – description t.b.d.

Dealing with forbidden characters in part numbers

If the engineering department did exist prior to using eng-db-2, chances are, there is an existing scheme for naming parts. Eng-db-2 stores part information in a folder named after the part. Name issues start to occur if the part name consists of invalid characters for folder names, such as ‘\’, ‘/’, ‘*’ etc.

Eng-db-2 deals with this kind of part names by creating an internal alias. Invalid characters are replaced with a dash (‘-‘) in folder names, and the ‘alias’ keyword is added to the info.txt file.

TODO: check if subst will do the alias functionality.

Importing pre-existing data

Often there will be a pre-existing parts database of some form that needs to be imported into eng-db-2. The data could stem from an operations database, or from another engineering tool that was previously used to manage parts. Another use-case scenario would be creating a number of parts at once for e.g. a new product.

Eng-db-2 can import part data from a CSV file. To do so, eng-db-2 must first be configured to import the correct CSV columns. The “import.txt” file is used to setup the CSV information. The file contains the following fields:

infoline=<line-number>
descr=<column name or number>
pn=<column name or number>
price=<column name or number>
avl=<vendor name column>, <vendor pn column>
default=<default sub-folder>
<sub-folder>=<pn-id>, <pn-id>, …, <pn-id>

The infoline identifier is optional and tells eng-db-2 the row number which holds the header titles. Both rows and columns are zero-based. Rows before this number are ignored. If infoline cannot be found in the file, it is assumed all other identifiers are numerical column numbers and that part records start at row 0.

All other column identifiers can either be numeric (zero-based as well) to indicate the column number, or contain the column name in the infoline. In the latter case, eng-db-2 will automatically extract the correct column number by searching for the column name. This is useful if the order of columns cannot be guaranteed over multiple imports. Both the pn and descr columns are required, all others are optional and will be ignored if not found in the CSV file.

The AVL identifier expects both a column for the vendor name followed by a column for the vendor part number, separated by a comma. Multiple AVL lines are supported.

If the imported data is supposed to be sorted into individual sub-folders like capacitors and resistors for the corresponding parts, eng-db-2 needs to know about the part number structure. The subfolder identifier is used for this purpose.

The subfolder identifier must start with a forward-slash (‘/’), and may contain multiple of them to support multiple folder levels.

The pn-id identifier is the part of the PN used for sorting the part into the correct folder structure. Several identifiers are supported for the same folder, separated by comma. Currently eng-db-2 only supports pn-ids at the beginning of the P/N.

The default identifier is required and used to have eng-db-2 create parts of unknown PN structure. If you do not want eng-db-2 to sort parts into different folders, this would be the only entry.

The following shows an example for the import.txt file format:

```
infoline=0
pn=item_number
descr=item_name
price=current_prod_cost
avl=vendor_1,vendor_item_number_1
avl=vendor_2,vendor_item_number_2
avl=vendor_3,vendor_item_number_3
# the 'default' definition is required
default=/undefined
/electrical/capacitors=CC
/electrical/diodes=CD
/electrical/resistors=CR
/electrical/oscillators=161,162,163
```

Appendix

Example part numbering scheme

You are completely free to choose your own part numbering scheme with Eng-DB-2. There are a few things to keep in mind though:

- apply some logic: it is good practice to design the part numbering scheme such that some basic information about the nature of a part can be retrieved from the part number.
- keep it simple: it is good practice to not try and encode every possible aspect of a part in the part number. Instead, part numbers should be relatively short and simple.
- allow for clashes: when using a simple scheme, sooner or later the situation will arise where you have multiple parts that would be assigned the same part number. It is good practice to allow for part number clashes of this kind by adding an index to the end of the part number.
- for localized parts, make sure you include some localization info, preferably following the ISO 3166 country codes: http://www.immigration-usa.com/country_digraphs.html

AC-xxx: Accessories/ Packout

AS-xxx: Assembly, set of items

This is used for a set of items, e.g. the packout BOM of a product

Cx-xxx: Electrical Components

CA-xxx: module/ electrical assembly/ cable assembly

Example: CA-AUBTM-23

CB-xxx: bare PCB boards

Example: CB-CCS-DSP

CC-xxx: Capacitors

CC-mme-d-ss-i

- mme: the mantissa of the capacitance with two digits plus the exponent with a single digit. Examples: 100=10pF; 103=10nF; 106=10uF.
- d: electrolyte. Use 0 for C0G/NP0, 5 for X5R, 7for X7R, Y for Y5V, A for aluminum.
- ss: the physical size of the component. Examples: 06=0603; 08=0805; TR=through-hole radial.
- i: an index to differentiate parts in case there are any clashes.

CD-xxx: Diodes

CD-xxx-i

- xxx: manufacturer part number. Care should be taken to use generic part numbers for parts that are available from multiple vendors. Examples: CU-4558-0 for a 4558 op-amp; CU-ADSP-BF592 for a Blackfin DSP from Analog Devices.
- I: index to differentiate parts in case of clashes.

CE-xxx: EMI material

E.g. ferrite cores attached to cables.

CF-xxx: Ferrites

CF-mme-ss-i

CJ-xxx: Connectors

CJ-xxx-ss-i

- xxx: this field holds the number/layout of pins.
- ss: the spacing of the pins in metric units.
- i: index to differentiate parts in case of clashes.

Examples:

- CJ-7x2-25-0: this is a dual-row header connector with 14 pins, 1/10” or 2.54 mm spacing
- CJ-TRS-3.5-0: this is a 3.5mm stereo audio jack

CL-xxx: Inductors

CL-mme-ss-i

- mme: the mantissa of the inductance with two digits plus the exponent with a single digit. Examples: 100=10uH, 470=47uH, 391=390uH
- ss: the physical size of the component. Examples: 10=1007
- i: an index to differentiate parts in case there are clashes
- mme: the mantissa of the resistance with two digits plus the exponent with a single digit. Examples: 310=31R, 471=470R
- ss: the physical size of the component, Examples: 08=0805

CQ-xxx: Transistors

See diodes for more details

CR-xxx: Resistors

CR-mme-ss-i for 5% or 10% tolerance

CR-mmme-ss-i for 1% tolerance

The fields are used as follows

- mme or mmme: the mantissa of the resistance with two/three digits plus the exponent with a single digit. Examples: 1001 = 1 kilo Ohm/1%; 330 = 33 Ohm/5%; 3R3 = 3.3 Ohm/5%.
- ss: the physical size of the component. Examples: 06= 0603; 08=0805.
- i: an index to differentiate parts in case there are any clashes.

CSW-xxx: Switches, encoders

CT-xxx: Transducers

CT-tvviiimmmr

- t: (T) Tweeter, (M) Midrange, (W) Woofer, (R) Passive Radiator, (S) Special, (H) Headphone, (C) MIC
- vv: 2 letter abbreviation for vendor
- iii: 3-digit nominal impedance value
- mmm: 3-character model number (based on vendor's model number)
- r: 1-character modifier to indicate model revision number (use hexadecimal)

Example: CT-TDM0062P61 for the most recent version of the tweeter used in Airdock.

CU-xxx: Integrated circuits

See diodes for more details

CVR-xxx: Potentiometers

CW-xxx: Cable/ Wire

CY-xxx: Crystals

Example: CY-24M576-0

Mx-xxx: Mechanical components

ME-xxx: electro-/mechanical module

MH-xxx: Hardware

MM-xxx: Metal parts

MO-xxx: Other mechanical items

MP-xxx: Plastic parts