ARE WE PLM YET?

A beginners introduction to product lifecycle management for KiCad

KiCon // 2025 // North America // v1

01

INTRODUCTION

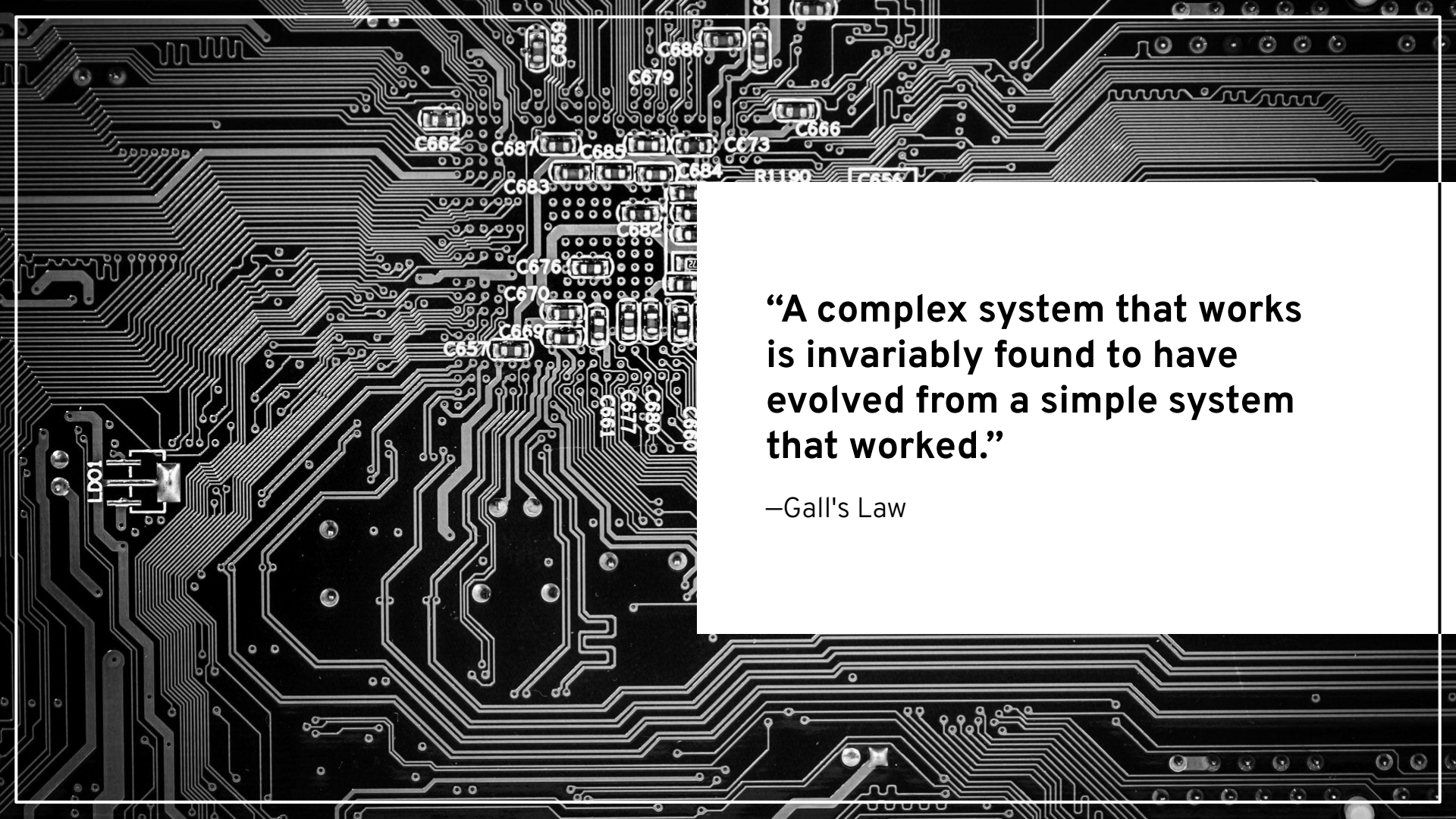
ABOUT ME

CHRIS WILSON

hardware design engineer
PM in PCBA manufacturing

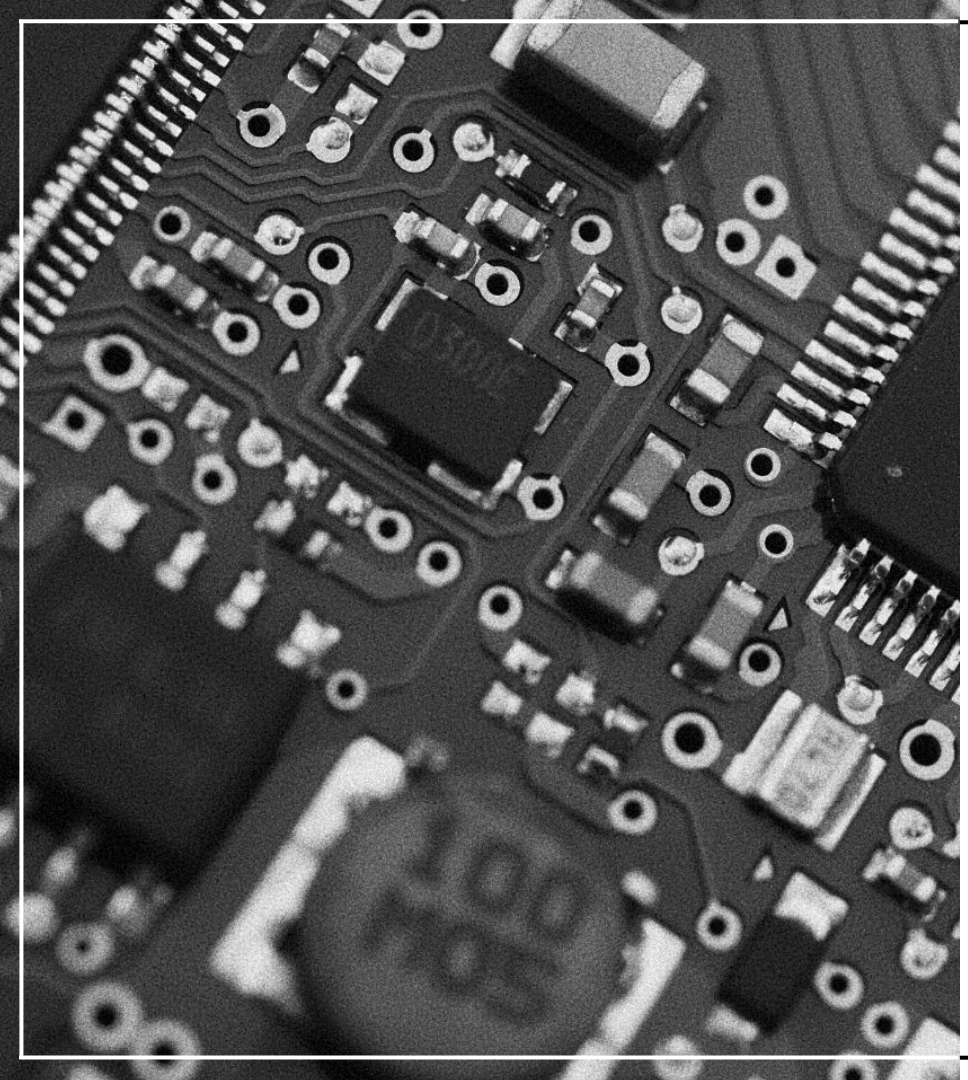
COMMON GROUND ELECTRONICS

embedded systems
engineering services



“A complex system that works is invariably found to have evolved from a simple system that worked.”

—Gall's Law



A SIMPLE SYSTEM THAT WORKS

This talk is about my attempt at setting up a simple PLM workflow for KiCad that works for open-source designs.

GOALS (AND NON-GOALS) FOR THIS TALK

PLM can get pretty complex, we only have 40 min!

Goals

- Enough info to get started with PLM in a weekend
- Walk through PLM integration with KiCad + Aligni

Non-goals

- Cover every aspect of PLM
- Compare / contrast multiple PLM systems

02

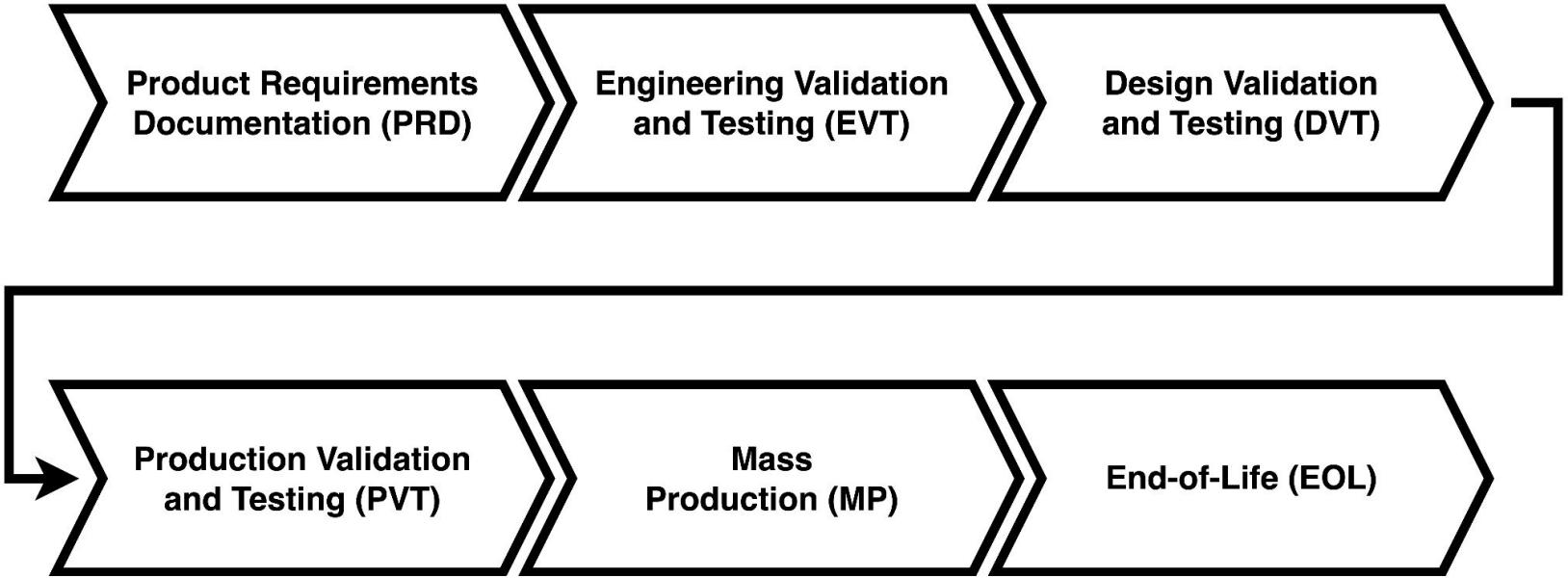
WHAT IS PLM?

A basic introduction to product lifecycle management

WHAT IS PLM?

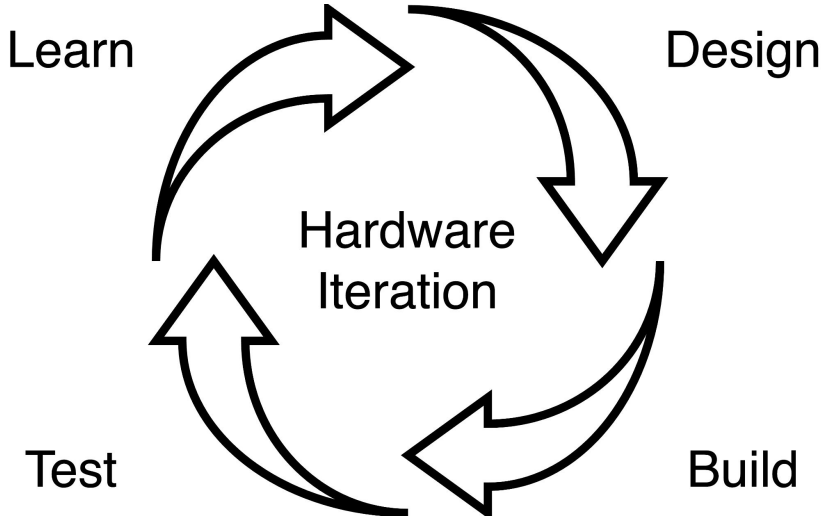
Product Lifecycle Management is a process that combines best practices and software tools to **centralize and structure product information** over the course of the product development lifecycle.

PRODUCT DEVELOPMENT LIFECYCLE



MULTIPLE HARDWARE ITERATIONS

Over the lifecycle of a product, the hardware design goes through multiple iterations.



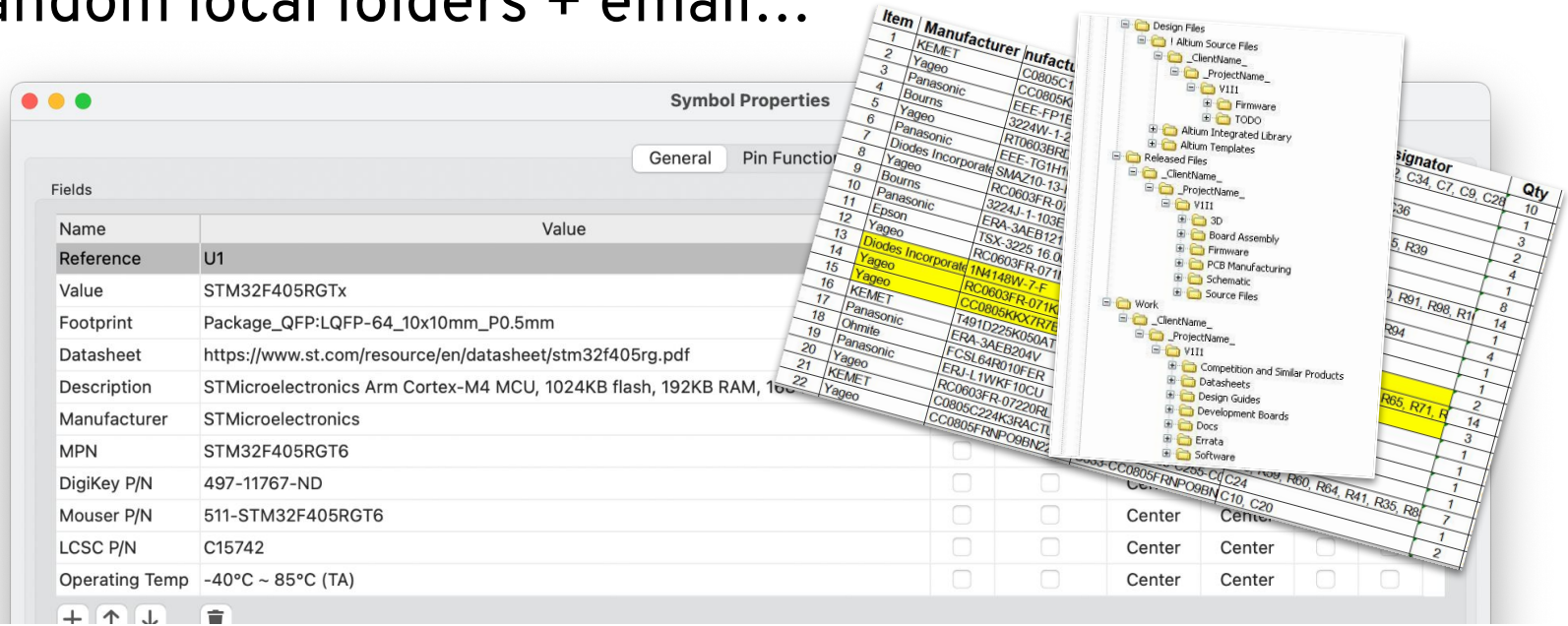
LOTS OF PRODUCT DATA TO MANAGE...

Each hardware iteration:

- Part numbers and revisions
- Parameters & specifications
- CAD files, drawings, manufacturing documentation
- Bill of Materials (BOM)
- Suppliers
- Regulatory compliance documentation
- etc...

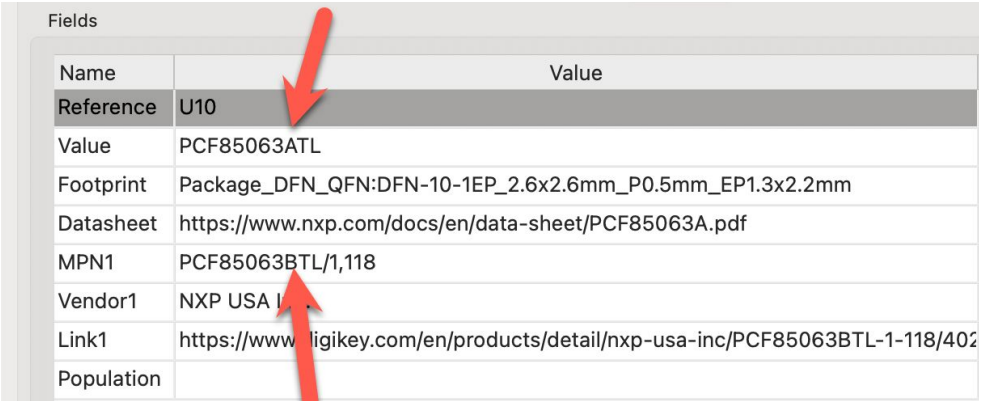
WHERE IS THIS DATA USUALLY STORED?

Sometimes in KiCad + spreadsheets + Google Drive + random local folders + email...



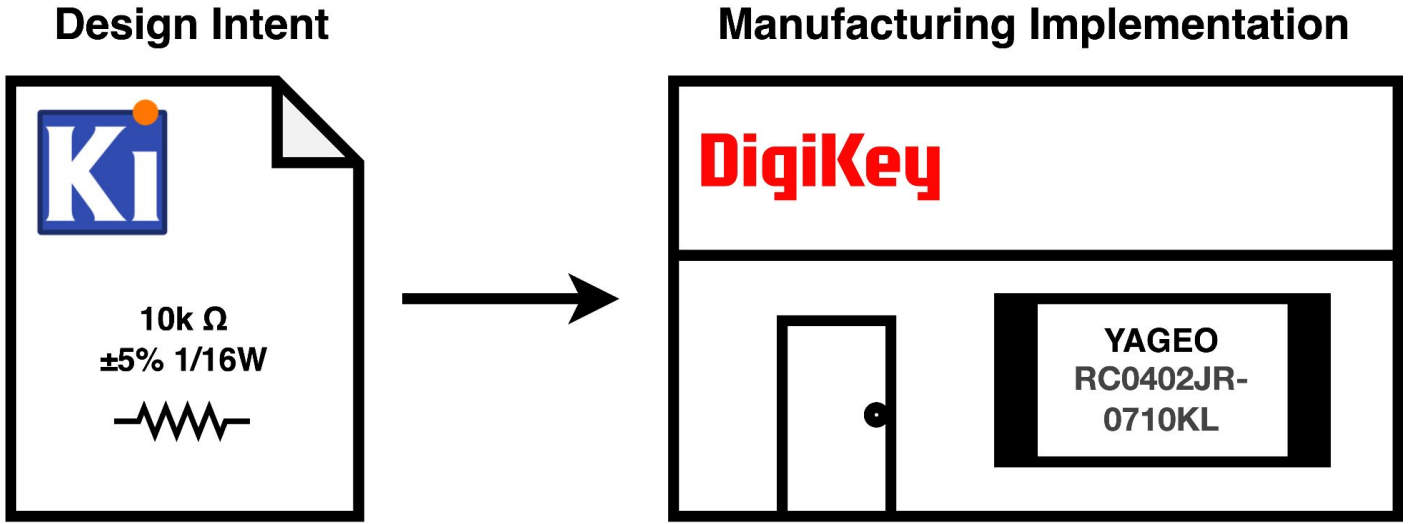
THIS IS NOT IDEAL

- Product data is siloed in KiCad
- Product data can become outdated
  - e.g. acquisitions (Fairchild → On Semi, etc)
- It's much easier to make mistakes with manual input!



DECOUPLE DESIGN FROM MANUFACTURING

Ideally CAD should reflect **design intent**, not a snapshot of today's supply chain data.



THERE'S GOTTA BE A BETTER WAY!

This darn data is so flingin' flangin' hard to manage!

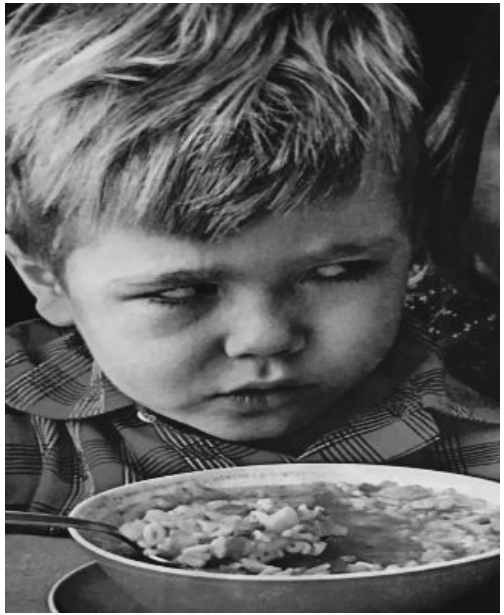
PLM SOFTWARE

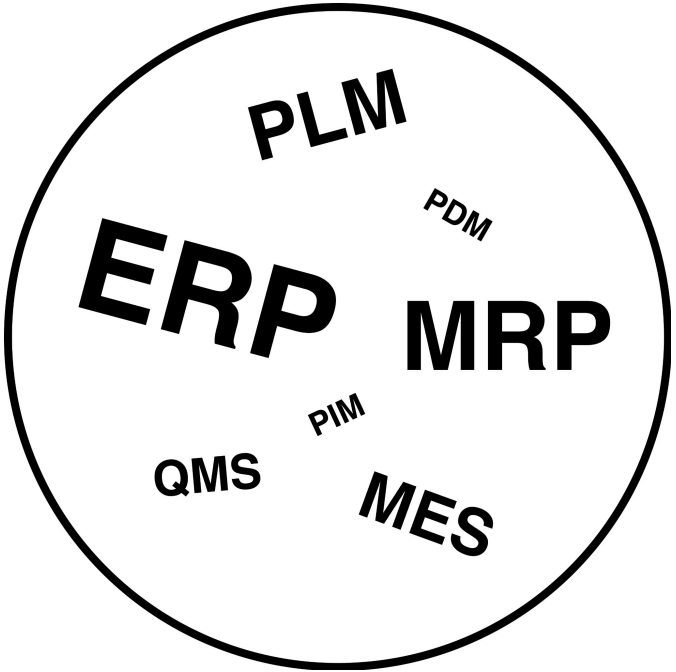
People realized this and made…PLM software!

WHAT DOES PLM SOFTWARE DO?

- Centralized “single source of truth” for product data
  - Part numbers, revisions, BOMs, documents, etc.
- Enables access to product data using role based access controls (internal teams & external vendors)
- Change management (ECR/ECO) and quality workflows
- Auditable change history
- Integrates with other systems (PDM ↔ CAD, MRP/ERP, MES, PIM, etc)

DECIPHERING THE “ALPHABET SOUP”





DECIPHERING THE “ALPHABET SOUP”

Product Data Management (PDM)

- Engineering tool to manage/version design files

Product Lifecycle Management (PLM)

- Central hub for product data, approvals, and lifecycle

Material Requirements Planning (MRP)

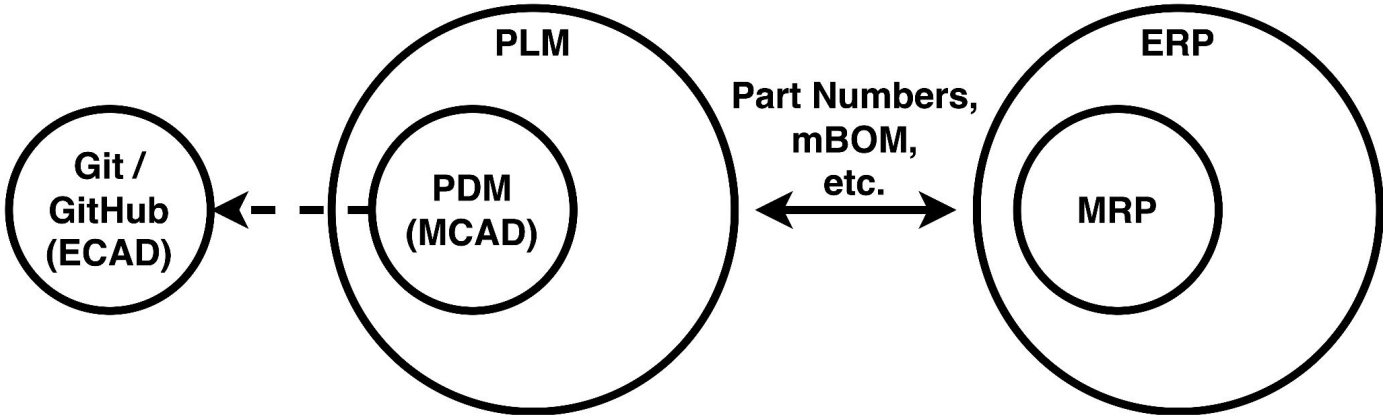
- Inventory, purchasing, scheduling, demand planning

Enterprise Resource Planning (ERP)

- Business and supply chain logic across departments

DECIPHERING THE "ALPHABET SOUP"

- PDM is typically a subset of PLM (MCAD)
  - More recently, ECAD tools using Git for PDM
- MRP is typically a "module" in a larger ERP system
- Product data flows from PLM to ERP (e.g. mBOM)



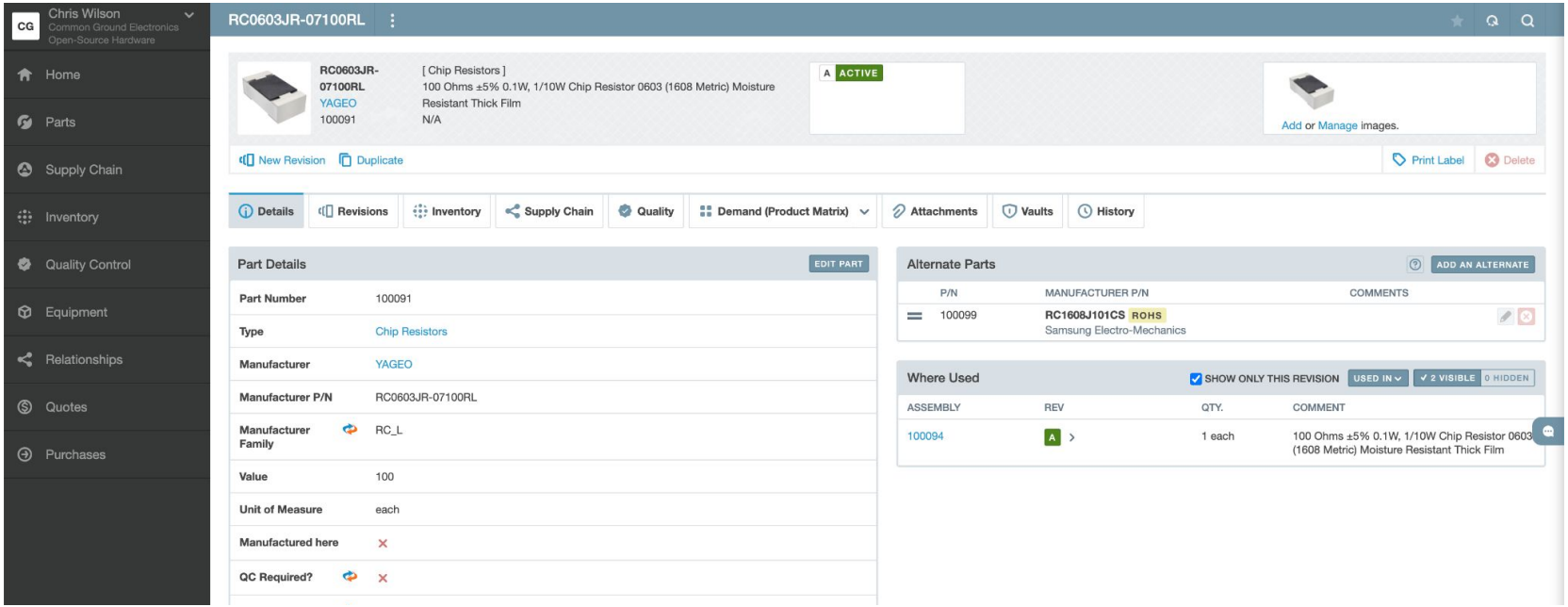
03

KICAD + ALIGNI

KiCad database library integration
with Aligni PLM

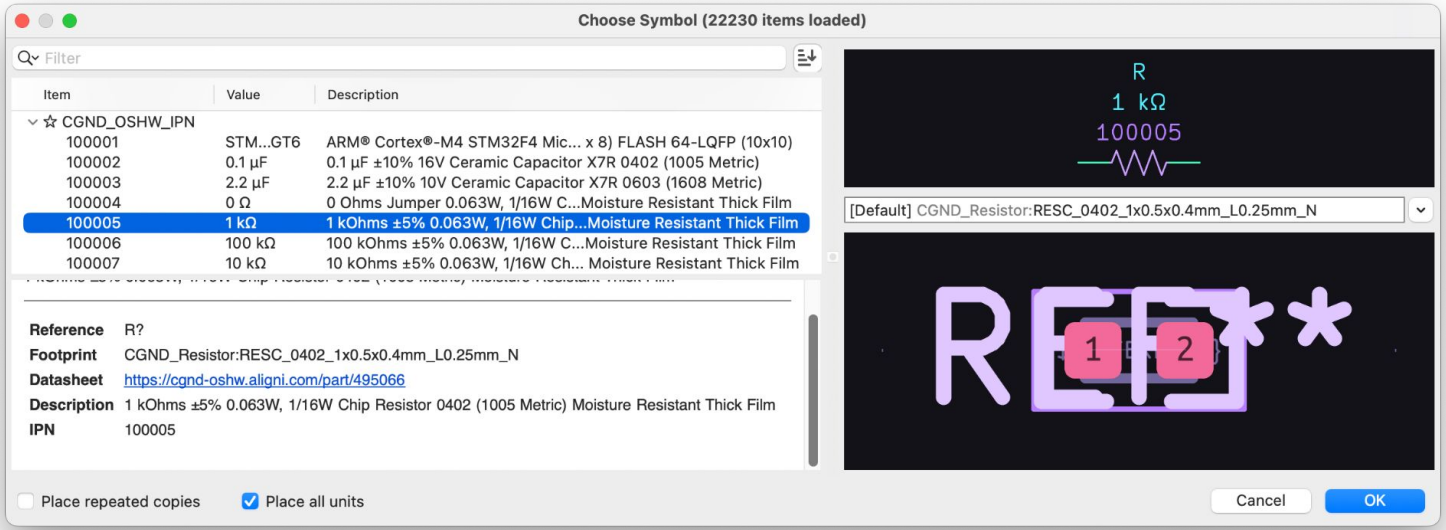
KICAD + ALIGNI // INTEGRATION GOALS

Part & supply chain data stored in Aligni (not KiCad)



KICAD + ALIGNI // INTEGRATION GOALS

Auto-generated library of fully-defined (“atomic”) parts



KICAD + ALIGNI // INTEGRATION GOALS

Import assembly BOM in Aligni directly from KiCad schematic BOM export

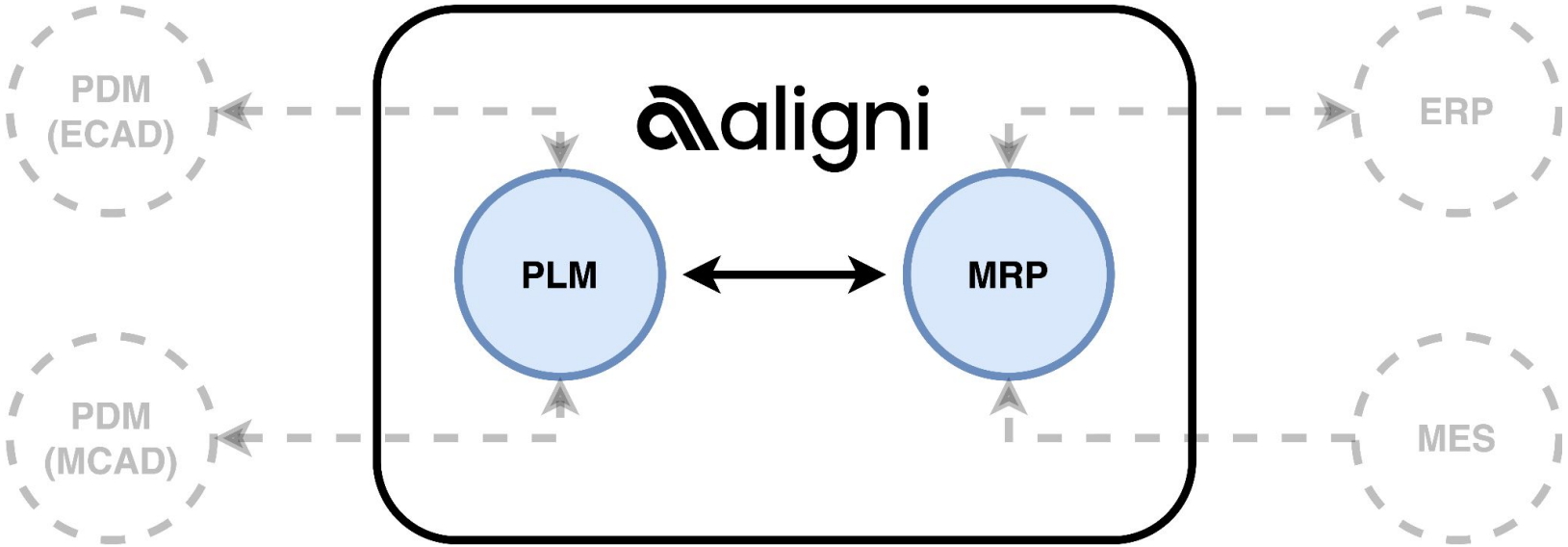
RPi_Pico_SAO_Host_v2_E...

#	#	Qty	Reference	#	IPN	Value	Description	Datasheet	DNP
	1	1	A1		100017	Raspberry Pi Pico	Raspberry Pi Pico, Microc	https://cgnd-oshw.aligr	
	2	1	D1		100089	Green	Green 570nm LED Indicati	https://cgnd-oshw.aligr	
	3	1	DOC1		100093	Pico SAO Host v2 Schematic	Raspberry Pi Pico SAO Ho	https://cgnd-oshw.aligr	
	4	2	J1,J2		100083	SFH11	6 Position Header Connec	https://cgnd-oshw.aligr	
	5	1	PCB1		100092	Pico SAO Host v2 PCB	Raspberry Pi Pico SAO Ho	https://cgnd-oshw.aligr	
	6	1	R1		100091	100 Ω	100 Ohms ±5% 0.1W, 1/1C	https://cgnd-oshw.aligr	
	7	1	R2		100090	560 Ω	560 Ohms ±5% 0.1W, 1/1C	https://cgnd-oshw.aligr	
	8	1	SW1		100069	PTS810SJM250SMTRLFS	Tactile Switch SPST-NO Tc	https://cgnd-oshw.aligr	
	9	1	SW2		100051	JS102011SAQN	Slide Switch SPDT Surfac	https://cgnd-oshw.aligr	

ALIGNI // INTRODUCTION

ALIGNI // PLM + MRP

Aligni combines PLM & MRP functionality



ALIGNI // PLM + MRP

- Part database (“Item Master”)
- Engineering change management (ECR/ECO)
- Quality control workflows
- Inventory management
- Planning & build management
- Supply chain management
  - Quoting / purchasing
  - Manufacturers / Vendors / Customers

ALIGNI // PLM + MRP

- **Part database (“Item Master”) ← this talk**
- Engineering change management (ECR/ECO)
- Quality control workflows
- Inventory management
- Planning & build management
- Supply chain management
  - Quoting / purchasing
  - Manufacturers / Vendors / Customers

ALIGNI // WHY ALIGNI?

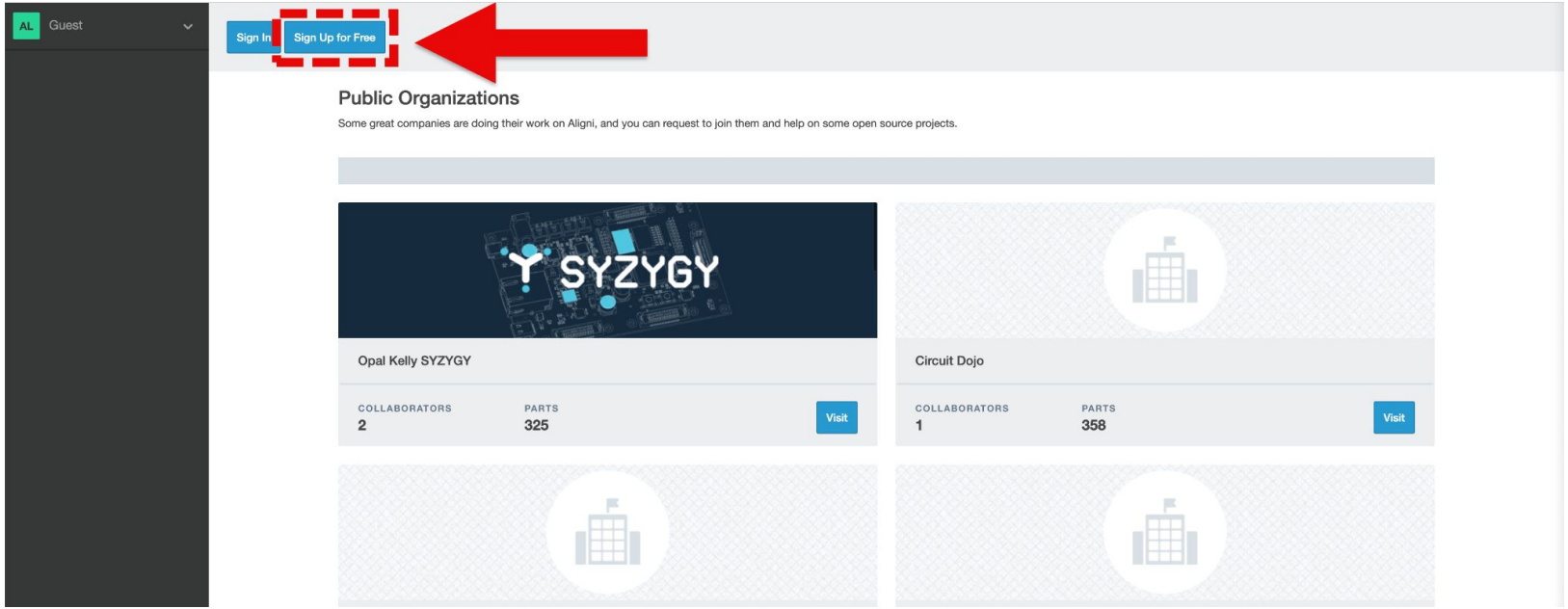
- Hosted solution with simple setup / no development
- Support for KiCad database (or HTTP) libraries
- ECAD agnostic (e.g. supports Altium also)
- Free tier with no time limit
- Public access for open-source projects (Open Aligni)
- Automatically managed internal part numbers

Aligni was the only solution that meet these criteria*

*In the last couple weeks, PartsBox added support for KiCad HTTP libraries

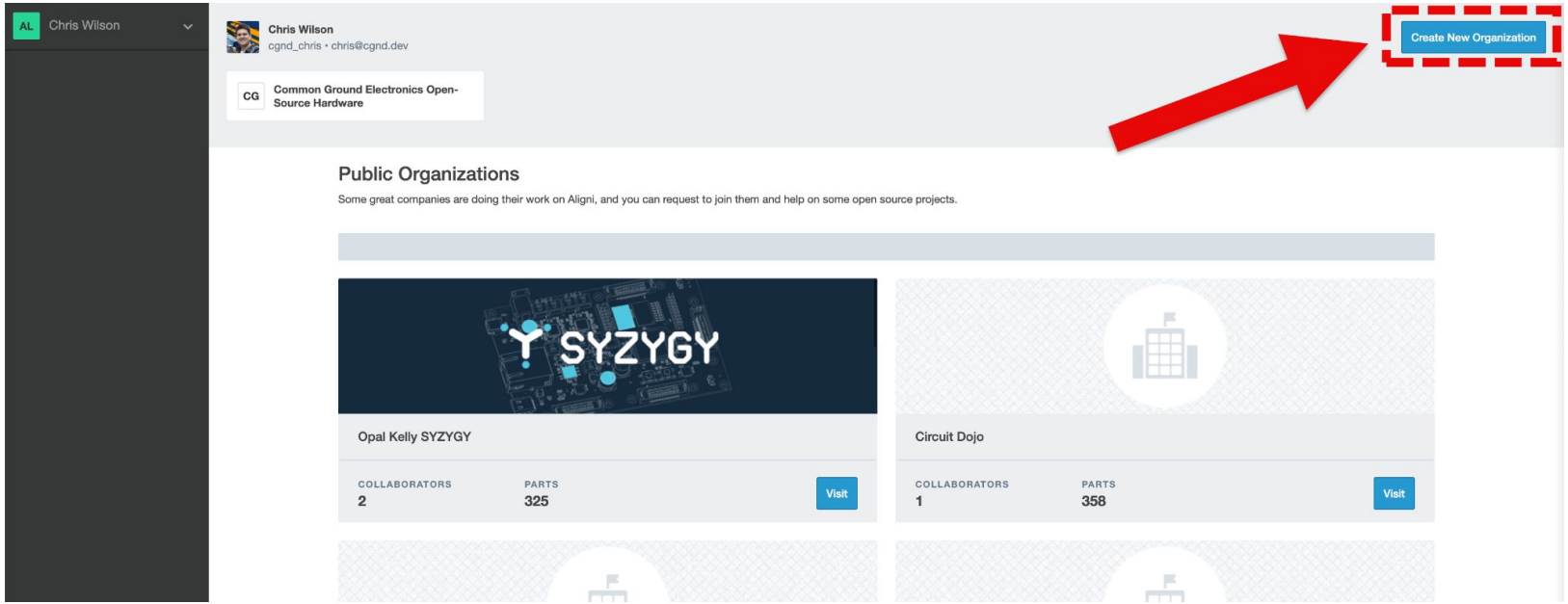
ALIGNI // SETUP // SIGN UP

Sign up for an account: https://app.aligni.com/catalog



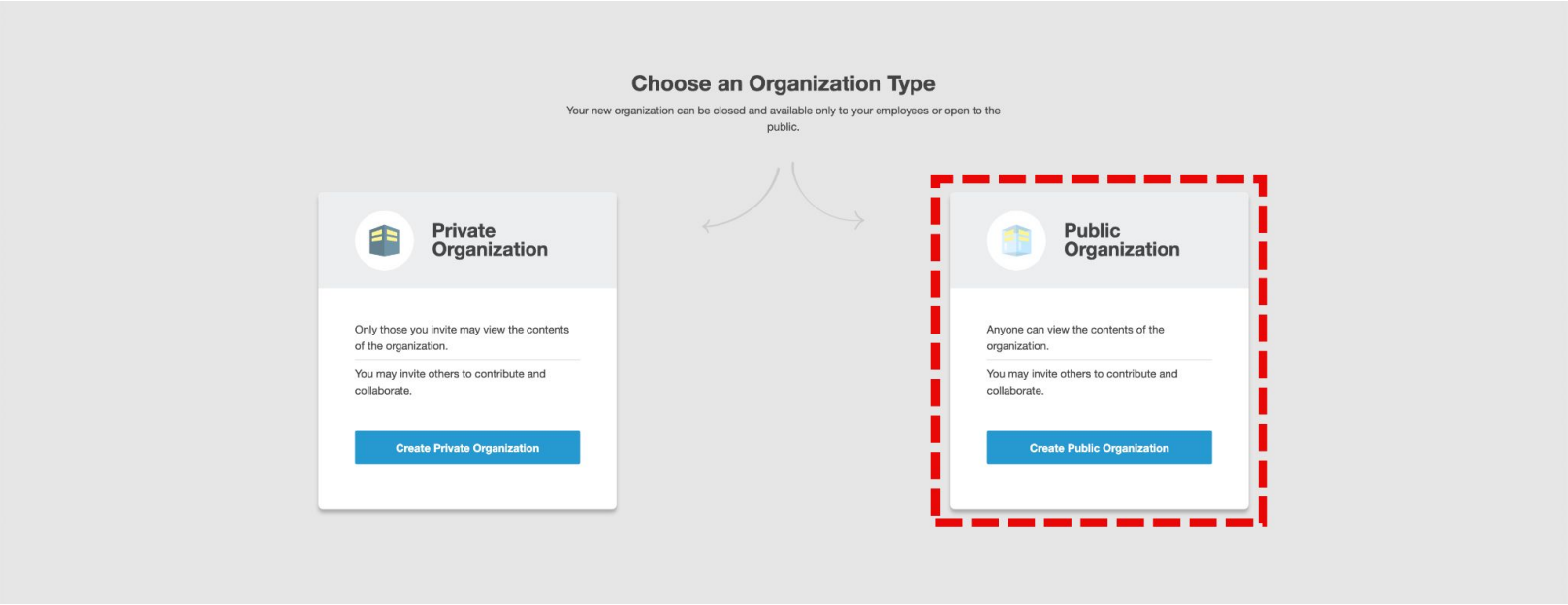
ALIGNI // SETUP // CREATE A NEW ORG

Create a new organization

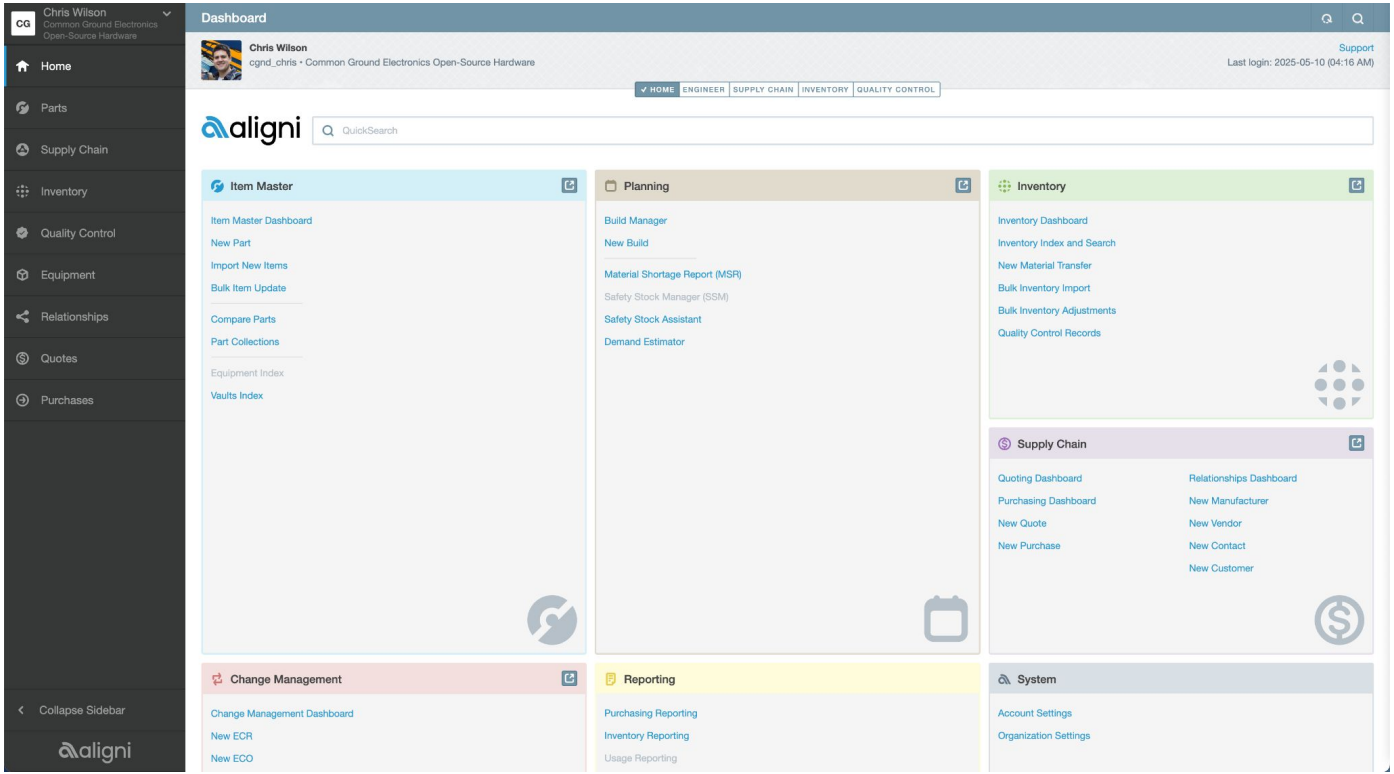


ALIGNI // SETUP // CREATE A NEW ORG

Choose “Public Organization” (free for open-source HW)

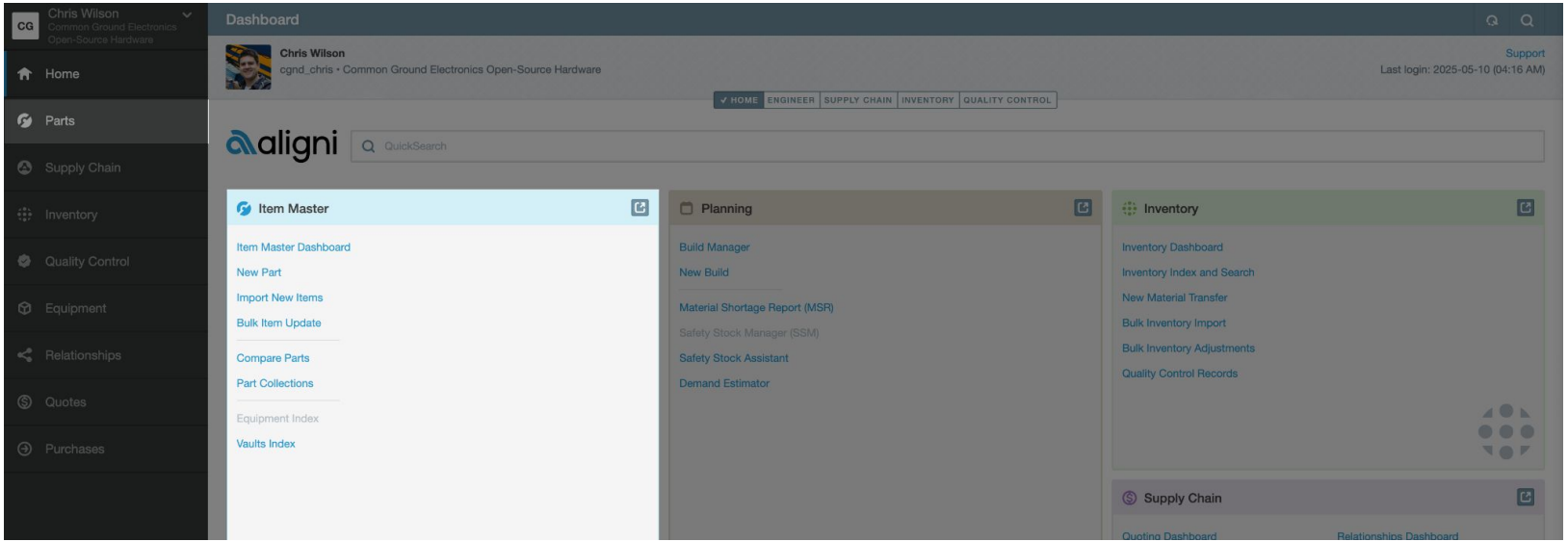


ALIGNI // DASHBOARD



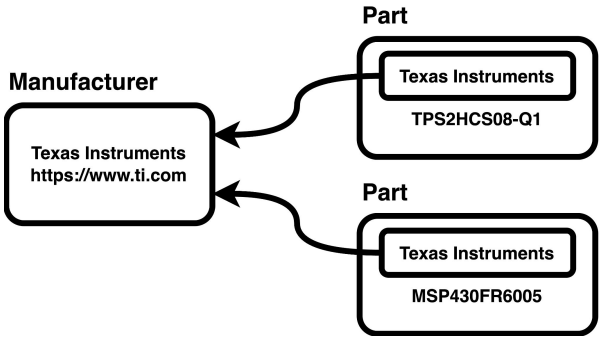
ALIGNI // ITEM MASTER

Focus of this talk is setting up the **Item Master** database as the source of part data for KiCad.

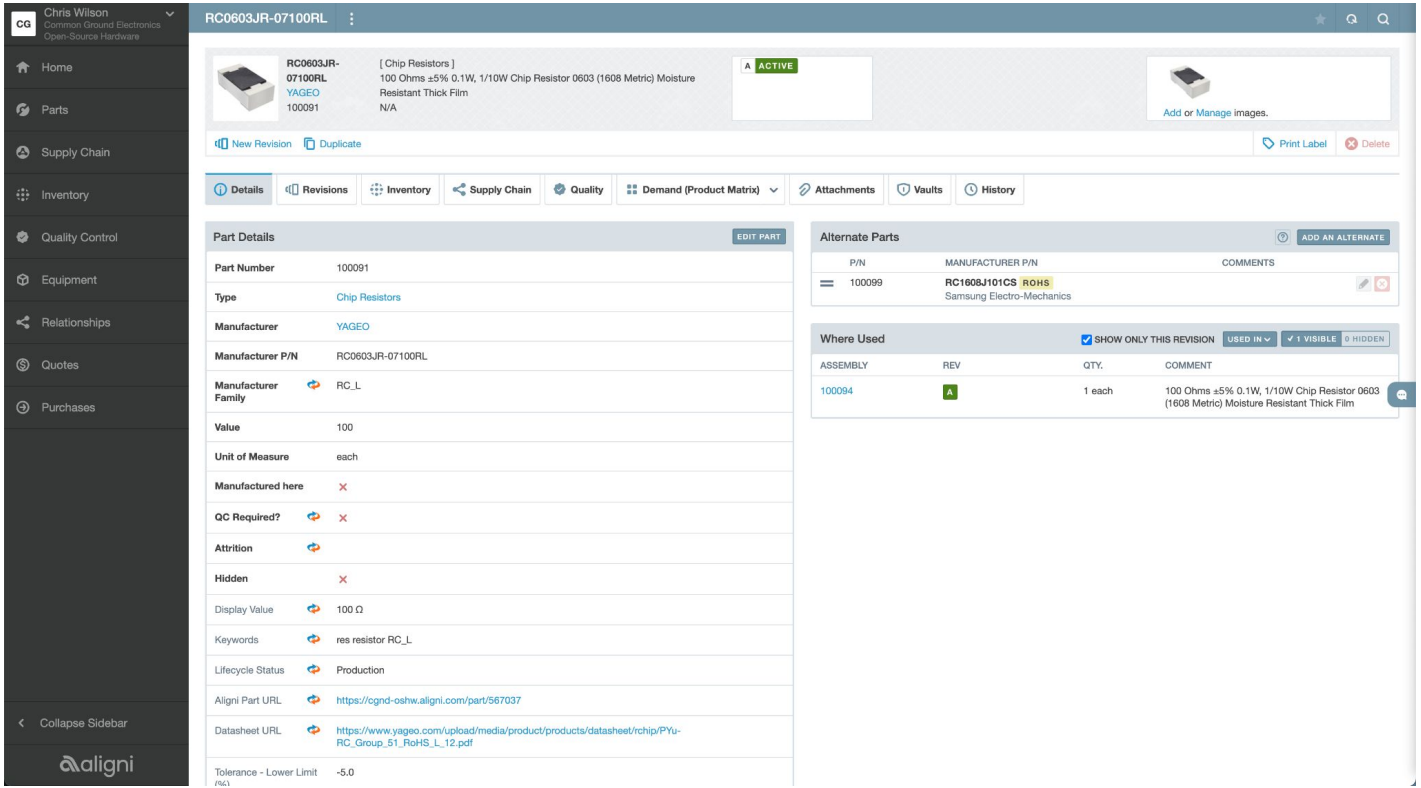


ALIGNI // ITEM MASTER

- "*Single source of truth*" central repository for part data that can be used in other systems
- Part numbers, descriptions, specifications, costs, suppliers, and other essential attributes
- Backed by a "normalized relational database"

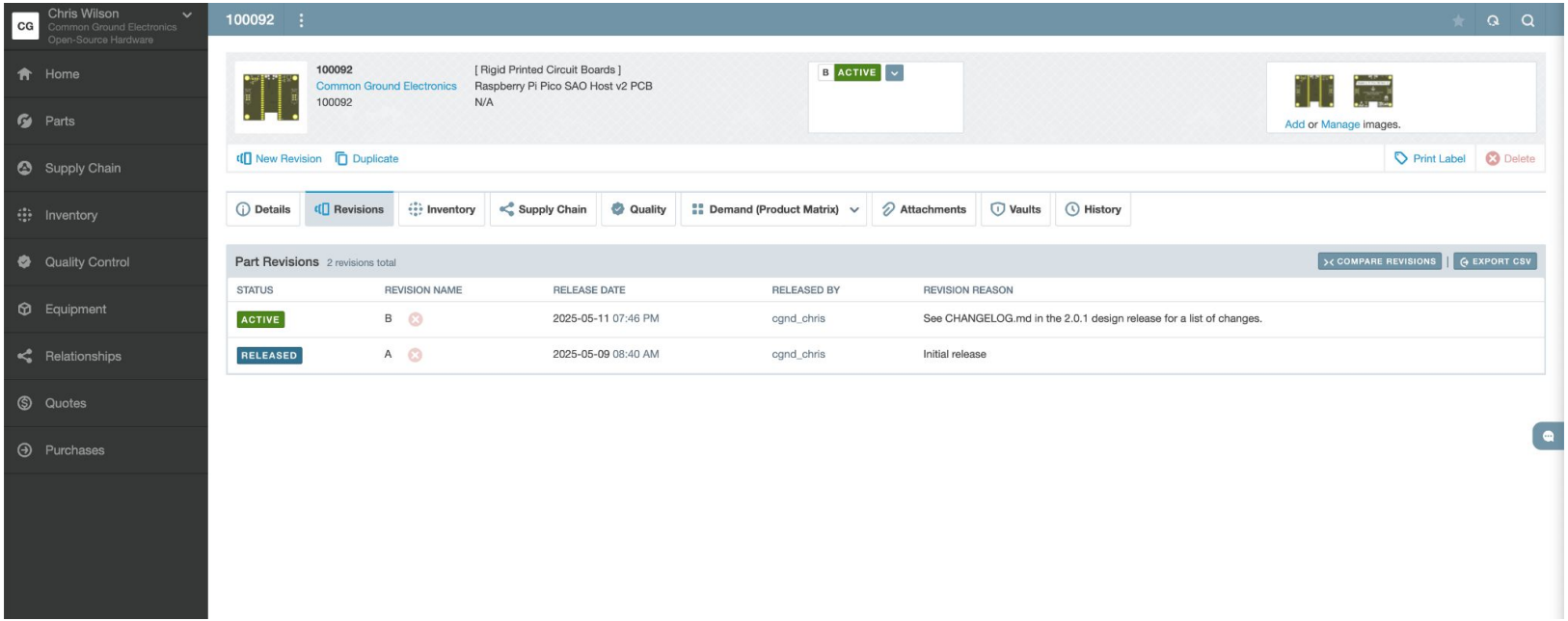


ALIGNI // PARTS



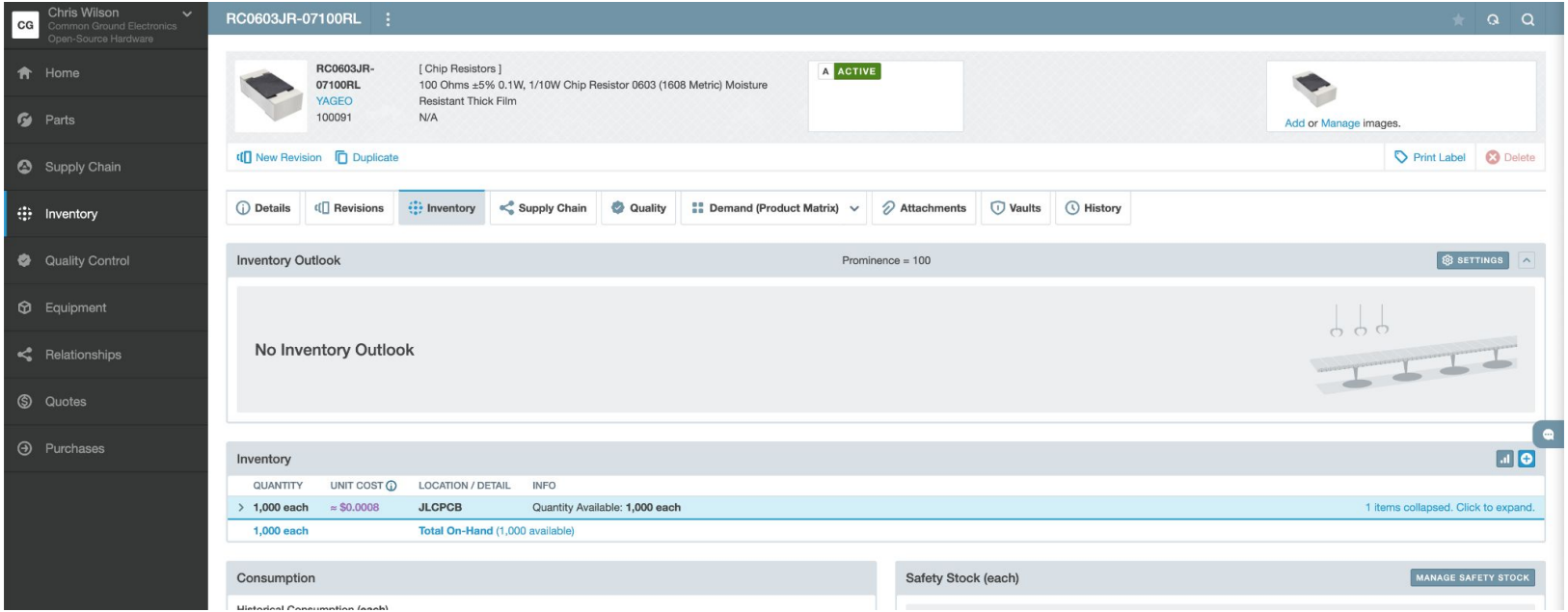
ALIGNI // PARTS // REVISIONS

Revisions: track changes to parts (more on this later)



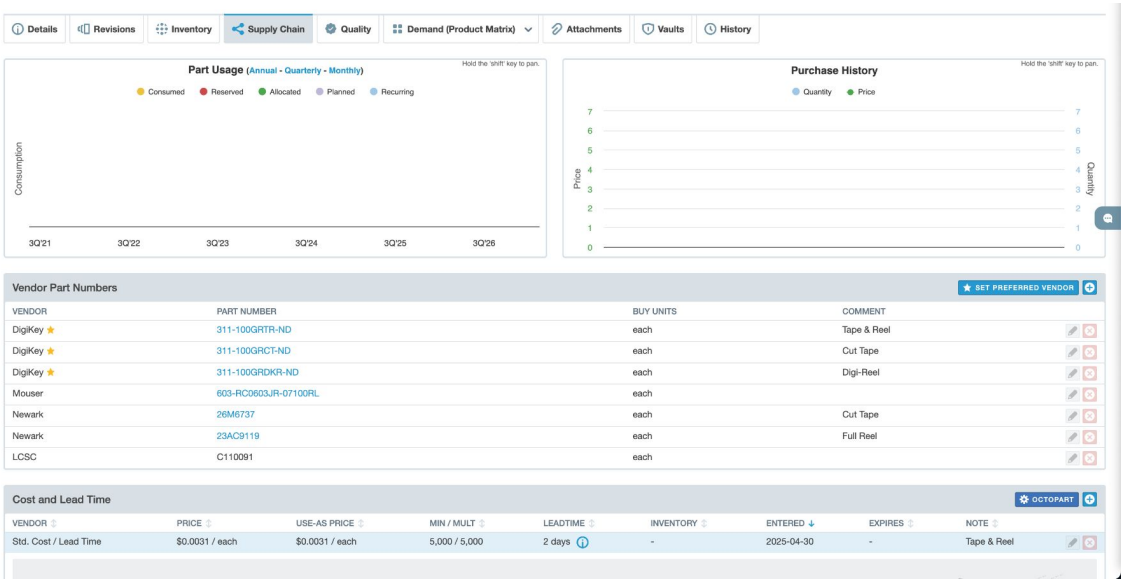
ALIGNI // PARTS // INVENTORY

Warehouse location and stock count of this part



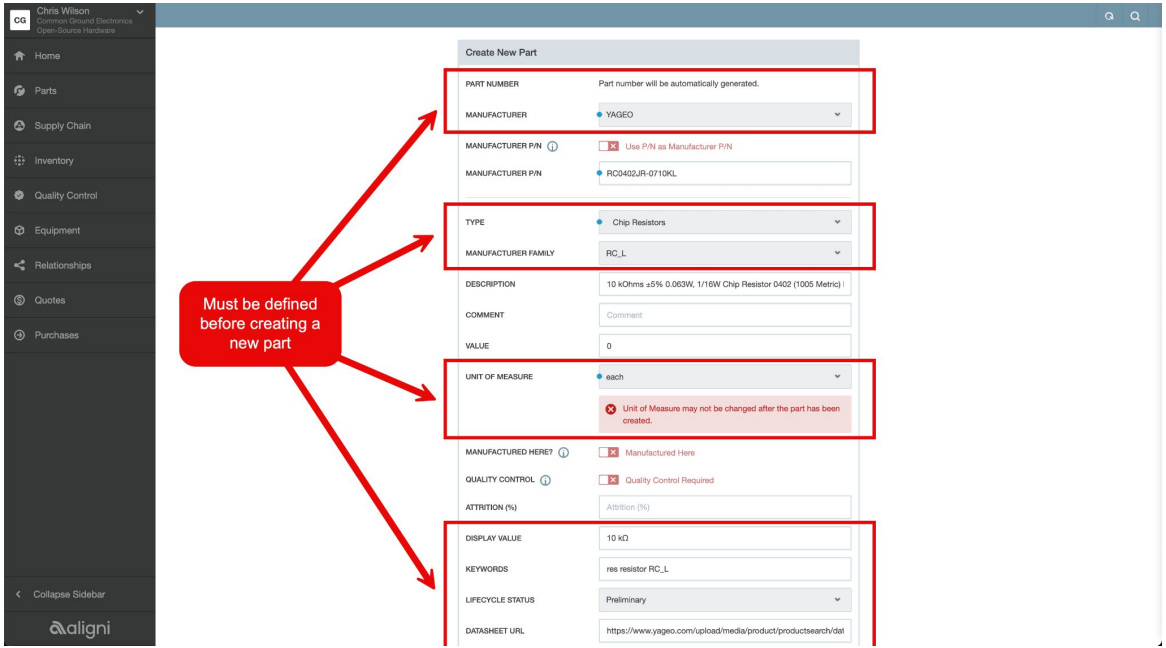
ALIGNI // PARTS // SUPPLY CHAIN

One-to-many relationship between the part in Aligni and supplier part number (SKU)



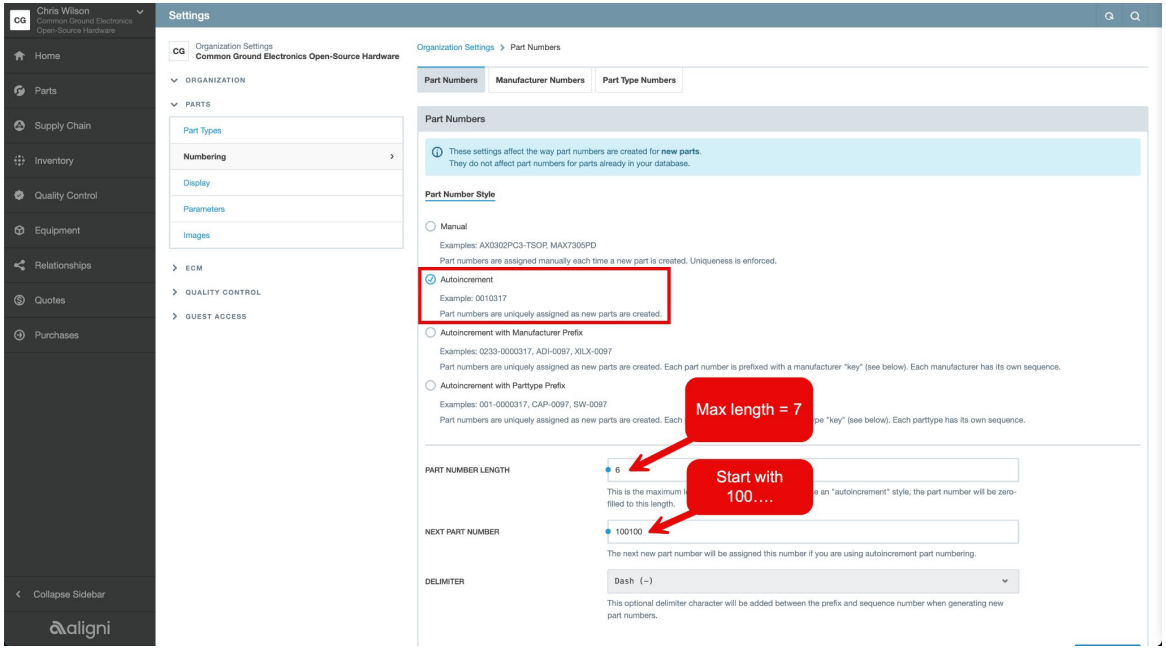
ALIGNI // PARTS // CREATING NEW PARTS

Some initial setup required before creating new parts



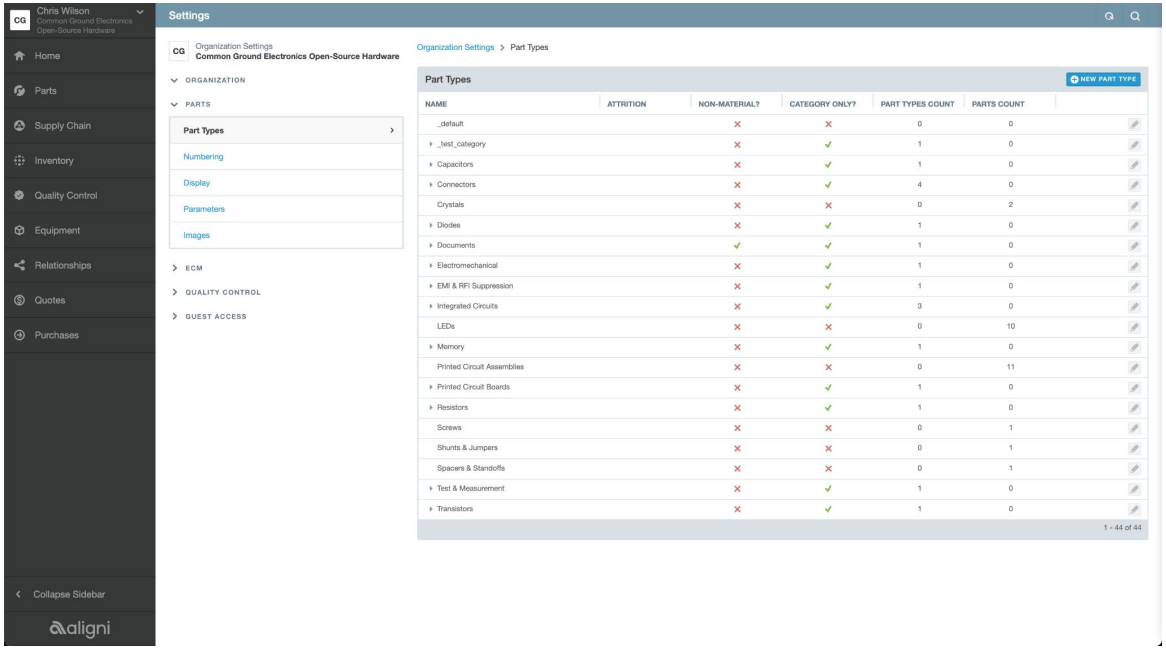
ALIGNI // SETUP // PART NUMBER SCHEME

Recommended internal part number scheme (`100001`):



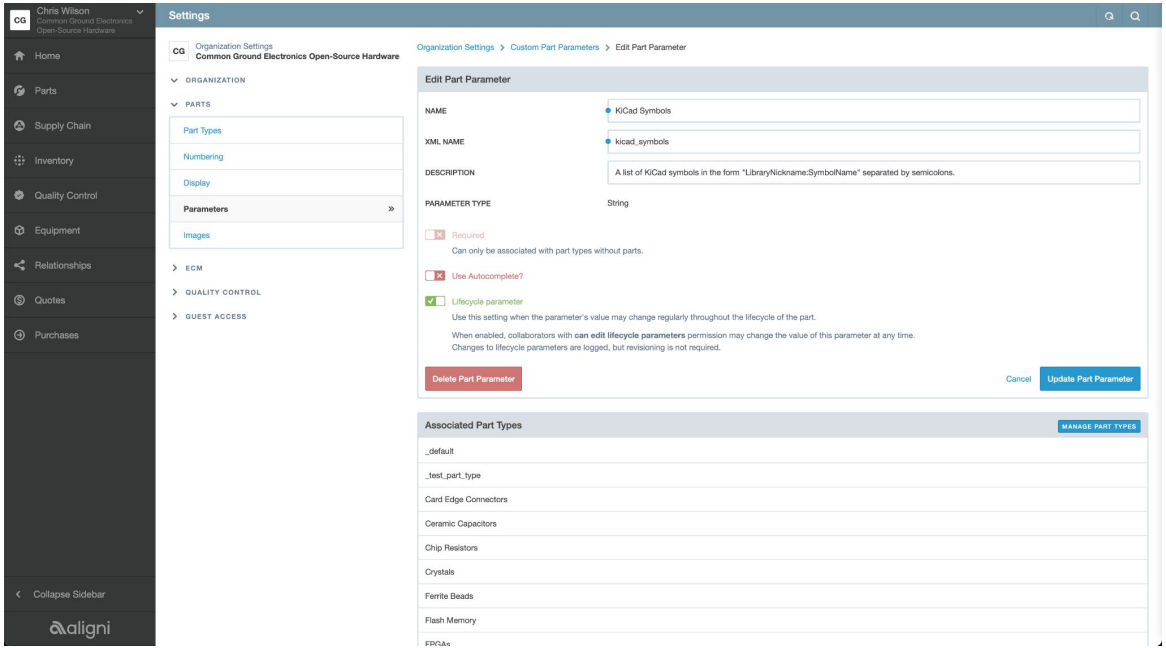
ALIGNI // SETUP // PART TYPES

Part Types associate parameters with groups of parts



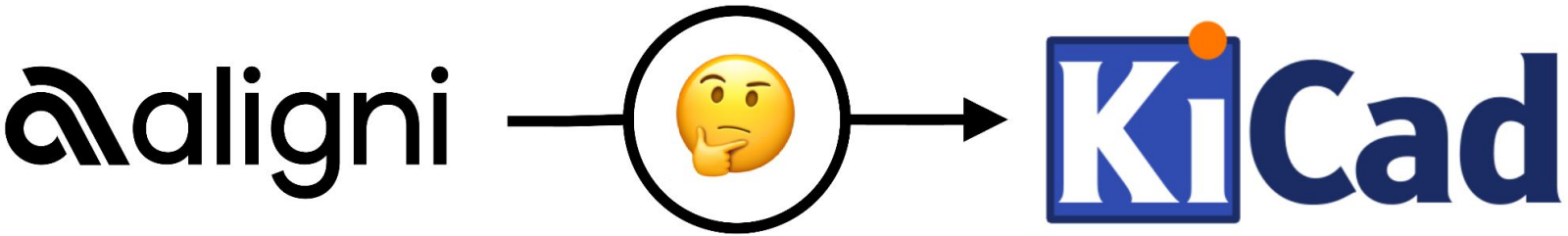
ALIGNI // SETUP // CUSTOM PARAMETERS

Example: “KiCad Symbols” custom parameter



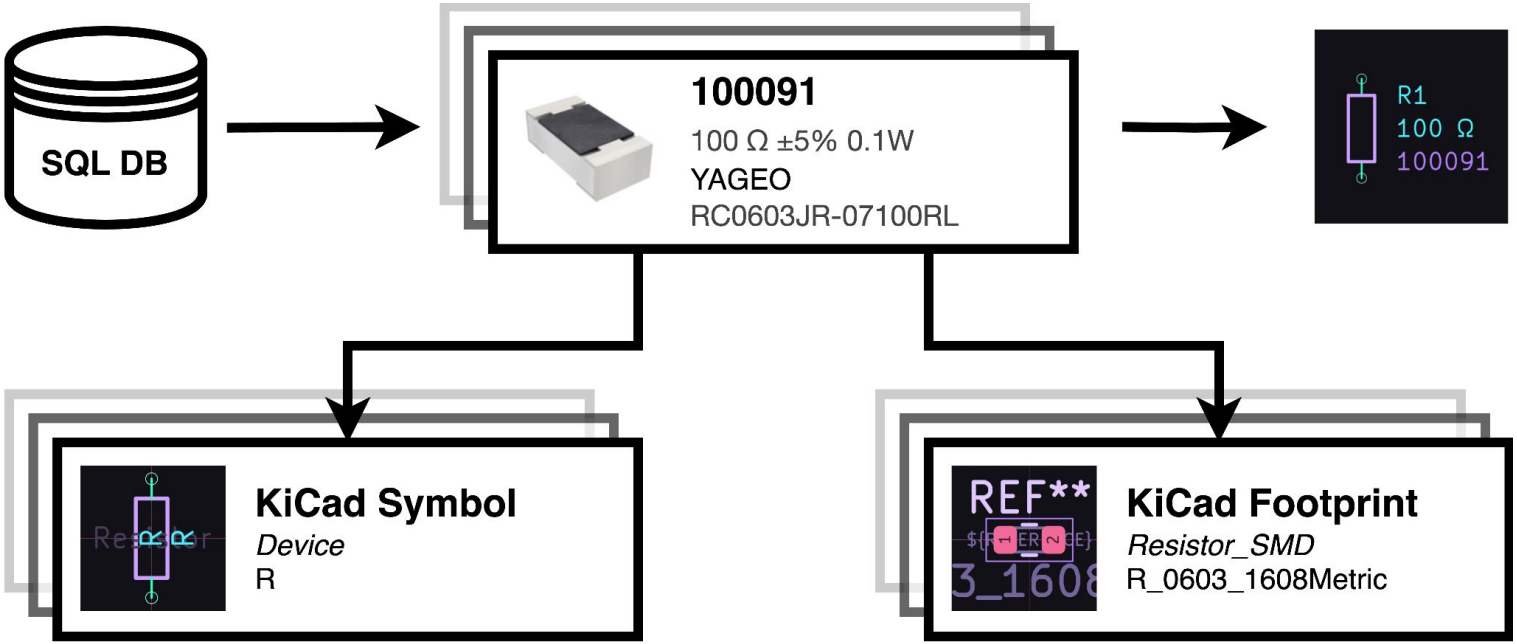
KICAD // DATABASE LIBRARY INTEGRATION

How to generate a KiCad library for all parts in Aligni?



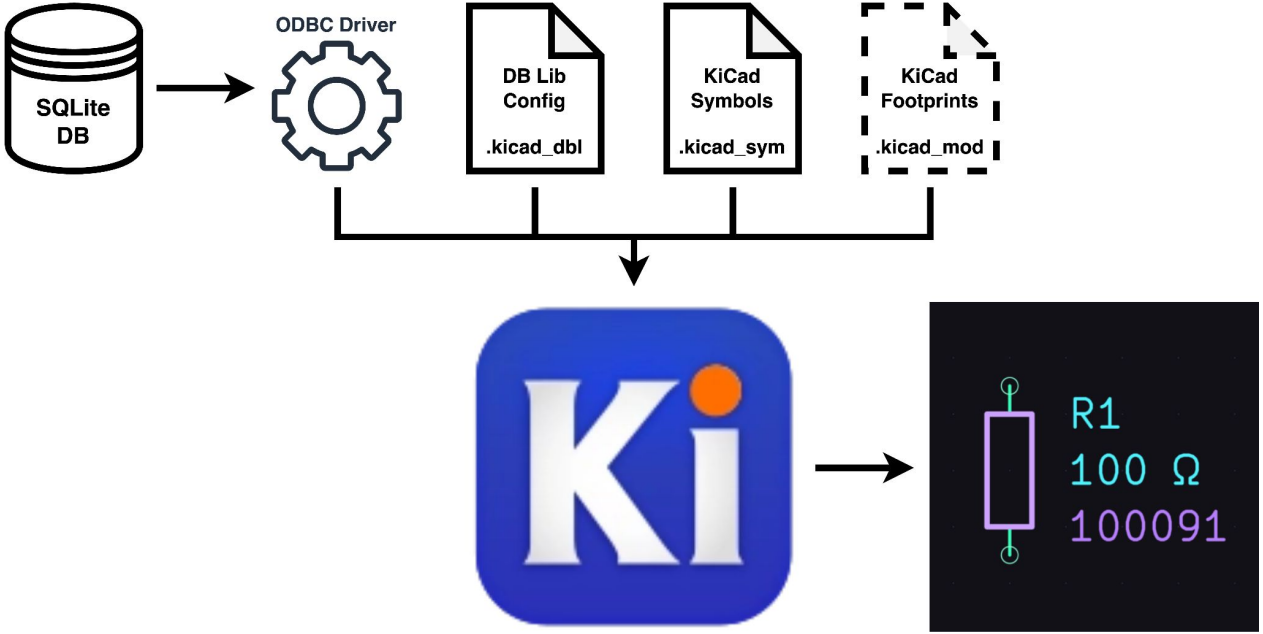
KICAD // DATABASE LIBRARY INTEGRATION

Database libraries are generated from SQL databases



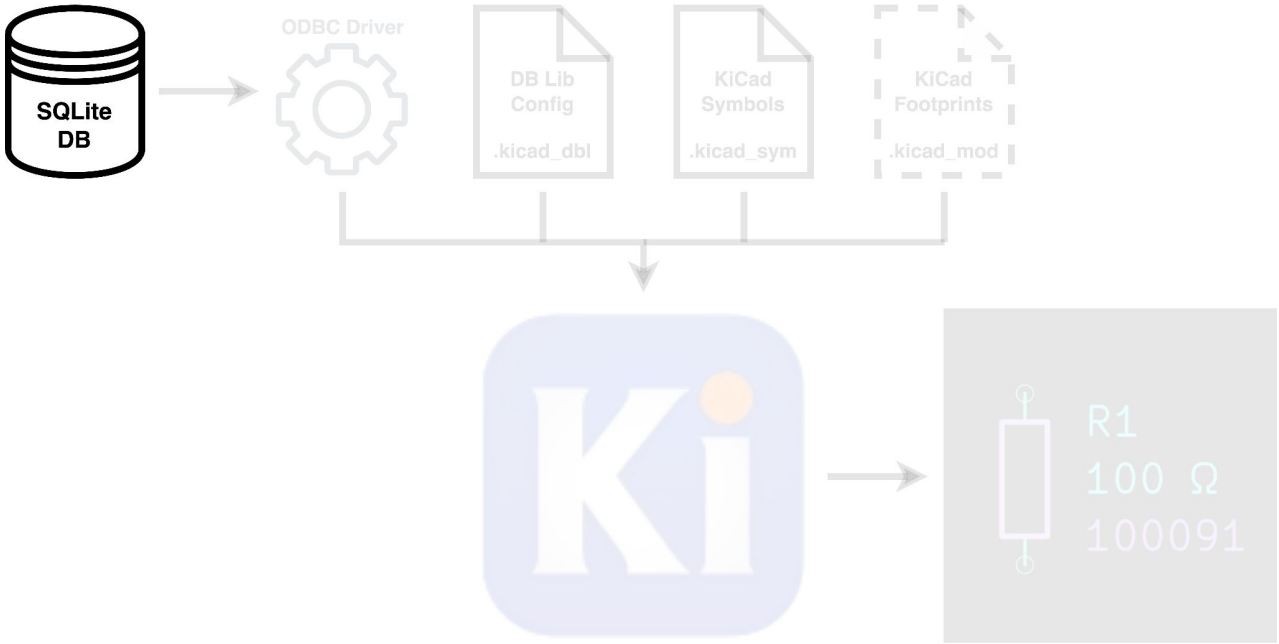
KICAD // DATABASE LIBRARY INTEGRATION

Database library inputs:



KICAD // DATABASE LIBRARY INTEGRATION

Where does the SQLite DB come from?



ALIGNI // REPLICATOR

Aligni Replicator is a Windows application* that generates a **local SQLite database** with all the parts in your online Aligni account.

*Replicator can be run in a VM but will not run in Wine.

https://docs.aligni.com/tools/replicator/

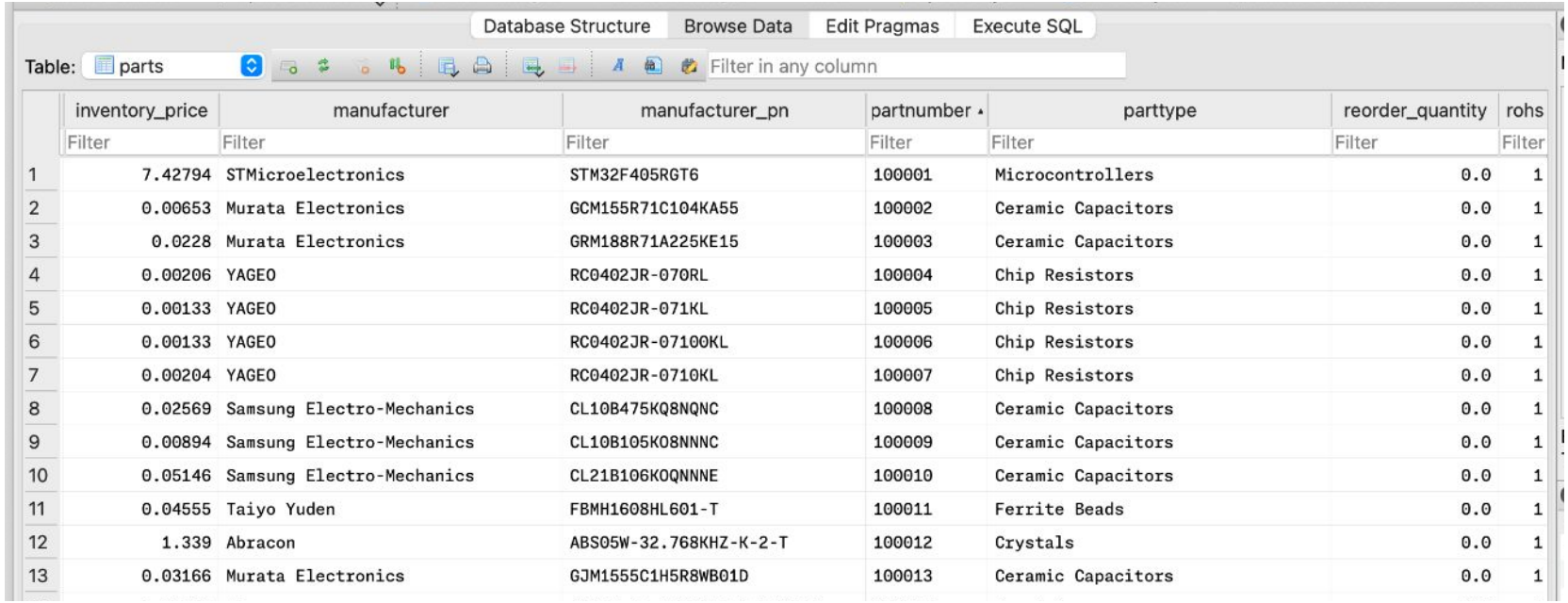
ALIGNI // REPLICATOR // DB SCHEMA

Database schema contains a `parts` table

Name	Type	Schema
Tables (1)		
parts		CREATE TABLE parts ('active' INT NULL, 'allow_fractional' INT NULL, 'comment' VARCHAR(4000), 'committed' INT
active	INT	"active" INT
allow_fractional	INT	"allow_fractional" INT
comment	VARCHAR(40...	"comment" VARCHAR(4000)
committed	INT	"committed" INT
created_on	DATETIME	"created_on" DATETIME
description	VARCHAR(40...	"description" VARCHAR(4000)
id	INT	"id" INT
inventory_price	FLOAT	"inventory_price" FLOAT
manufacturer	VARCHAR(40...	"manufacturer" VARCHAR(4000)
manufacturer_pn	VARCHAR(40...	"manufacturer_pn" VARCHAR(4000)
partnumber	VARCHAR(40...	"partnumber" VARCHAR(4000)
parttype	VARCHAR(40...	"parttype" VARCHAR(4000)
reorder_quantity	FLOAT	"reorder_quantity" FLOAT
rohs	INT	"rohs" INT
updated_on	DATETIME	"updated_on" DATETIME
value	FLOAT	"value" FLOAT
value_text	VARCHAR(40...	"value_text" VARCHAR(4000)

ALIGNI // REPLICATOR // DB DATA

`parts` table contains all parts from Aligni Item Master



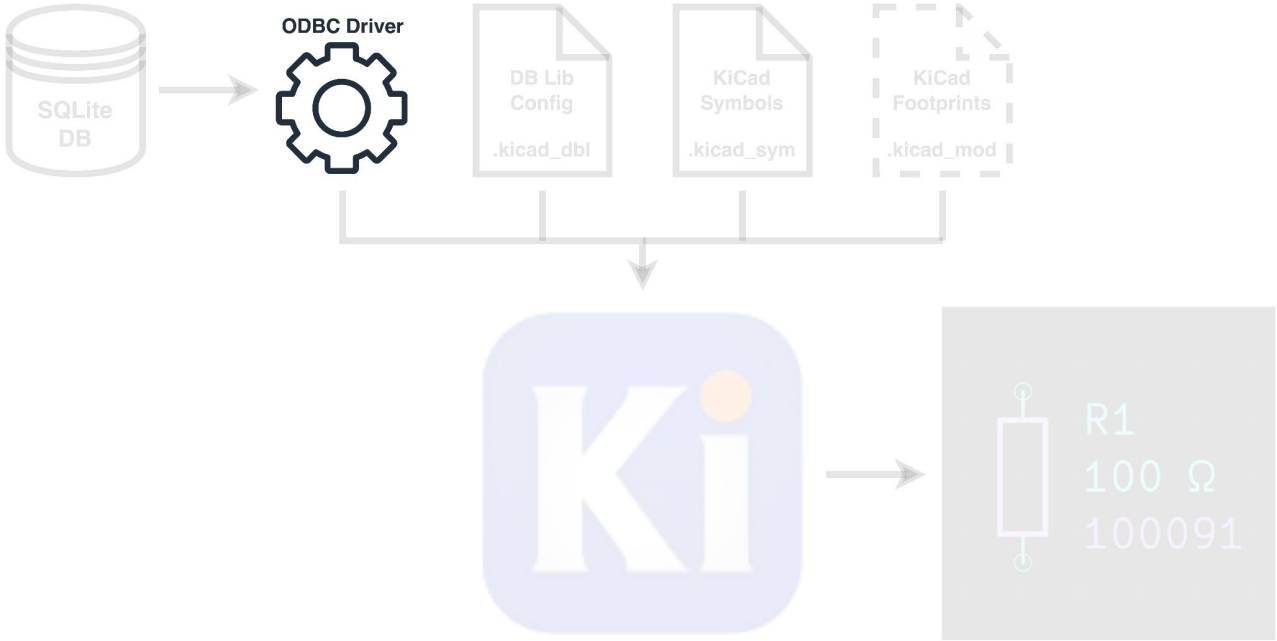
Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: parts — Filter in any column

	inventory_price	manufacturer	manufacturer_pn	partnumber	parttype	reorder_quantity	rohs
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	7.42794	STMicroelectronics	STM32F405RGT6	100001	Microcontrollers	0.0	1
2	0.00653	Murata Electronics	GCM155R71C104KA55	100002	Ceramic Capacitors	0.0	1
3	0.0228	Murata Electronics	GRM188R71A225KE15	100003	Ceramic Capacitors	0.0	1
4	0.00206	YAGEO	RC0402JR-070RL	100004	Chip Resistors	0.0	1
5	0.00133	YAGEO	RC0402JR-071KL	100005	Chip Resistors	0.0	1
6	0.00133	YAGEO	RC0402JR-07100KL	100006	Chip Resistors	0.0	1
7	0.00204	YAGEO	RC0402JR-0710KL	100007	Chip Resistors	0.0	1
8	0.02569	Samsung Electro-Mechanics	CL10B475KQ8NQNC	100008	Ceramic Capacitors	0.0	1
9	0.00894	Samsung Electro-Mechanics	CL10B105KO8NNNC	100009	Ceramic Capacitors	0.0	1
10	0.05146	Samsung Electro-Mechanics	CL21B106KOQNNNE	100010	Ceramic Capacitors	0.0	1
11	0.04555	Taiyo Yuden	FBMH1608HL601-T	100011	Ferrite Beads	0.0	1
12	1.339	Abracon	ABS05W-32.768KHZ-K-2-T	100012	Crystals	0.0	1
13	0.03166	Murata Electronics	GJM1555C1H5R8WB01D	100013	Ceramic Capacitors	0.0	1

KICAD // DATABASE LIBRARY INTEGRATION

What's this ODBC driver thing?



KICAD // DATABASE LIBRARY INTEGRATION

ODBC (Open Database Connectivity)

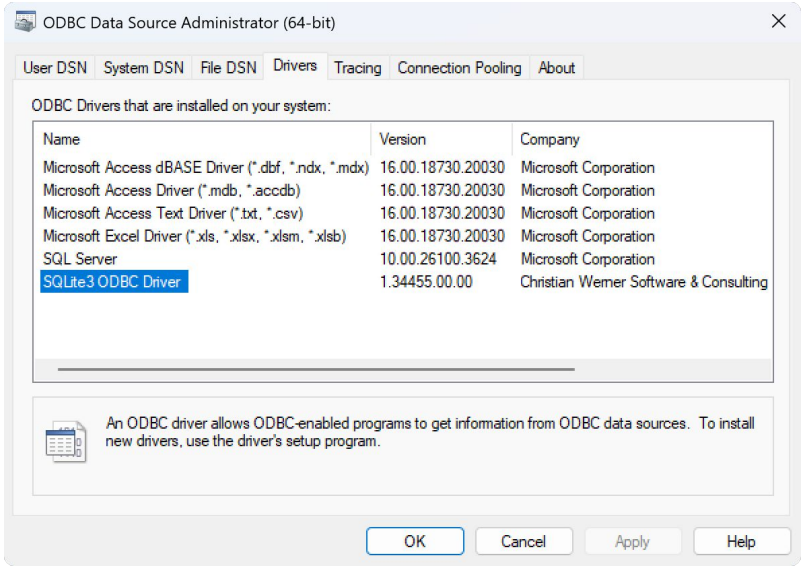
- Allows application to interact with different DBMS
- Industry-standard API

SQLite ODBC Driver
http://www.ch-werner.de/sqliteodbc/

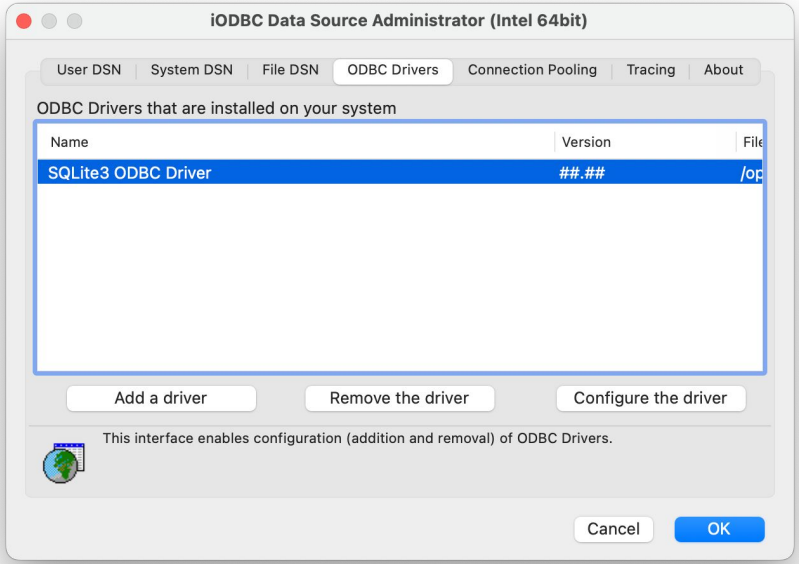
NOTE: On Windows, KiCad requires 64-bit ODBC driver, but Aligni Replicator requires 32-bit driver–install both!

KICAD // DATABASE LIBRARY INTEGRATION

Windows (64-bit)

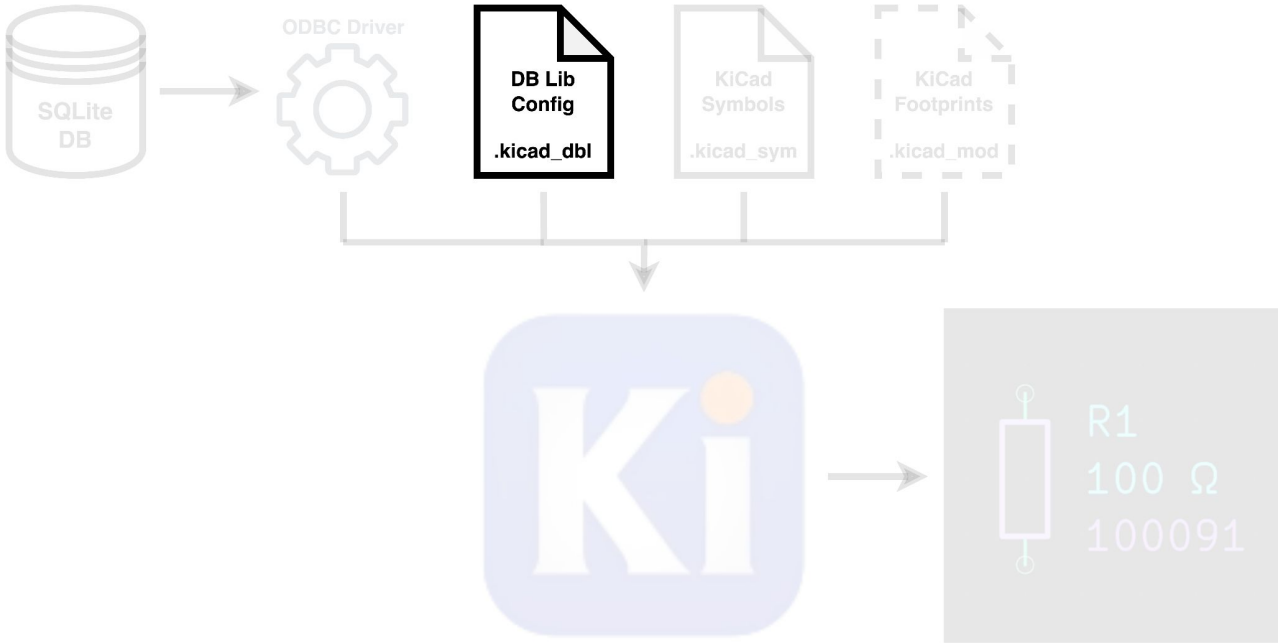


macOS



KiCad // Database Library Integration

Config file tells KiCad how to generate the DB library



KICAD // DATABASE LIBRARY INTEGRATION

Example: CGND_OSHW_IPN.kicad_dbl

- Database library configuration file
- Maps tables/fields from DB to KiCad symbols/fields

KICAD // DATABASE LIBRARY INTEGRATION

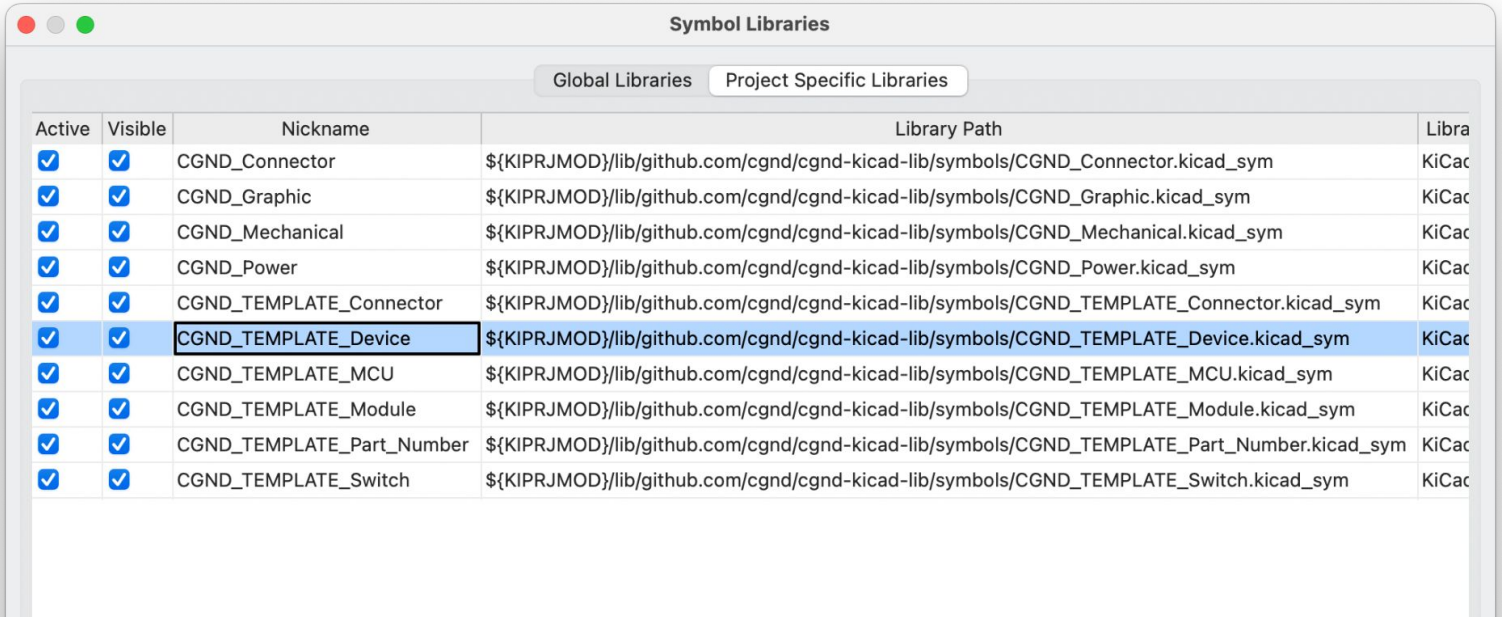
```
{
  "meta": {
    "version": 1
  },
  "name": "Common Ground Electronics OSHW IPN Library",
  "description": "A KiCad database library containing internal part number symbols",
  "source": {
    "type": "odbc",
    "dsn": "",
    "username": "",
    "password": "",
    "timeout_seconds": 2,
    "connection_string": "Driver={SQLite3 ODBC Driver};Database=${CWD}/../databases/CGND_OSHW_Aligni.sqlite"
  },
```

KICAD // DATABASE LIBRARY INTEGRATION

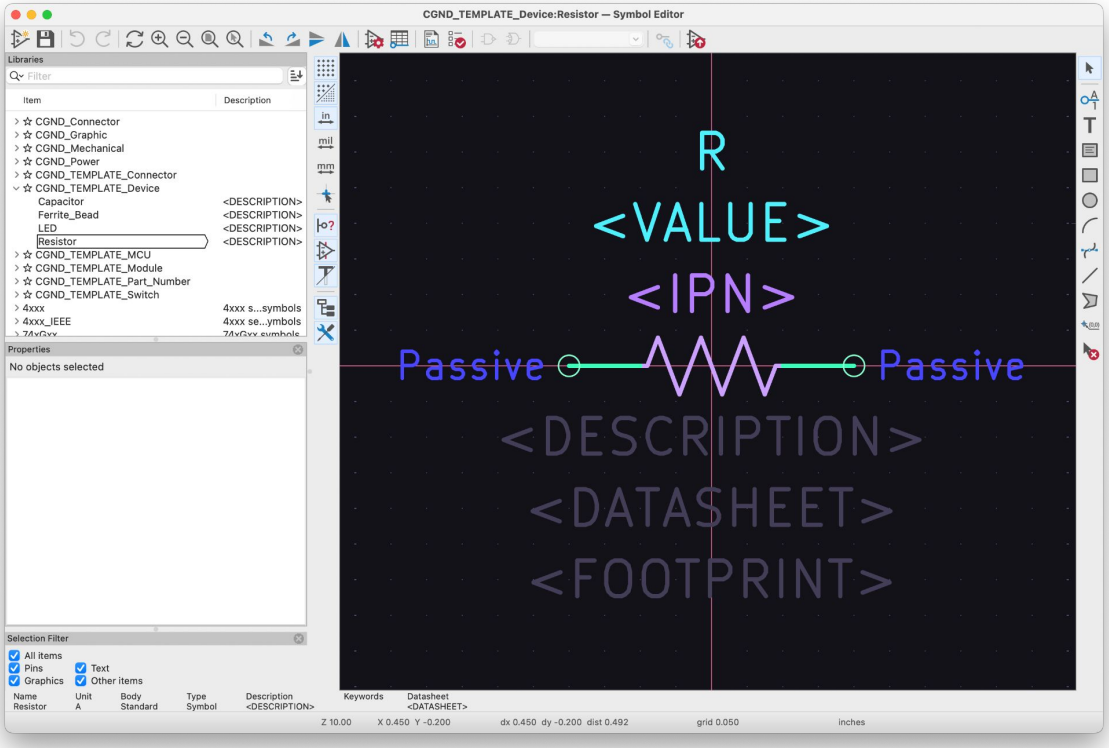
```
"libraries": [
  {
    "name": "",
    "table": "parts",
    "key": "partnumber",
    "symbols": "x_kicad_symbols",
    "footprints": "x_kicad_footprints",
    "fields": [
      ...
```

KICAD // DATABASE LIBRARY INTEGRATION

Referenced symbols must be defined in the library table



KICAD // DATABASE LIBRARY INTEGRATION



Fields/attributes populated from the Aligni DB.

No supply-chain details in the library symbol (only IPN and URL to Aligni part)

KICAD // DATABASE LIBRARY INTEGRATION

Map Aligni “Display Value” to symbol’s “Value” field

```
"fields": [
  {
    "column": "x_display_value",
    "name": "Value",
    "visible_on_add": false,
    "visible_in_chooser": false,
    "show_name": false,
    "inherit_properties": true
  },
```

KICAD // DATABASE LIBRARY INTEGRATION

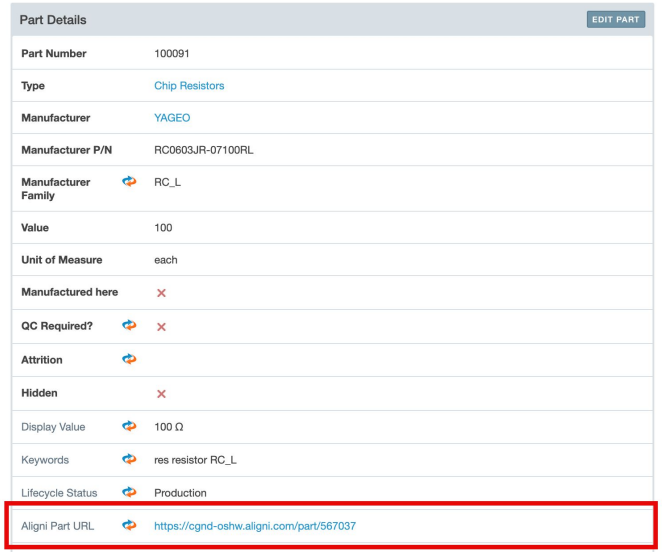
Map Aligni part URL to symbol's "Datasheet" field

ALIGNI // REPLICATOR IS MISSING PART URL

How to get the `x aligni part url` column in the Aligni DB to map to the KiCad “Datasheet” field?

Simplest solution is to add a custom part parameter with the Aligni part URL.

(See later slides for a SQL solution to add the URL)



KICAD // DATABASE LIBRARY INTEGRATION

Map Aligni part number to symbol's "IPN" field

```
{
  "column": "partnumber",
  "name": "IPN",
  "visible_on_add": true,
  "visible_in_chooser": true,
  "show_name": false,
  "inherit_properties": true
},
```

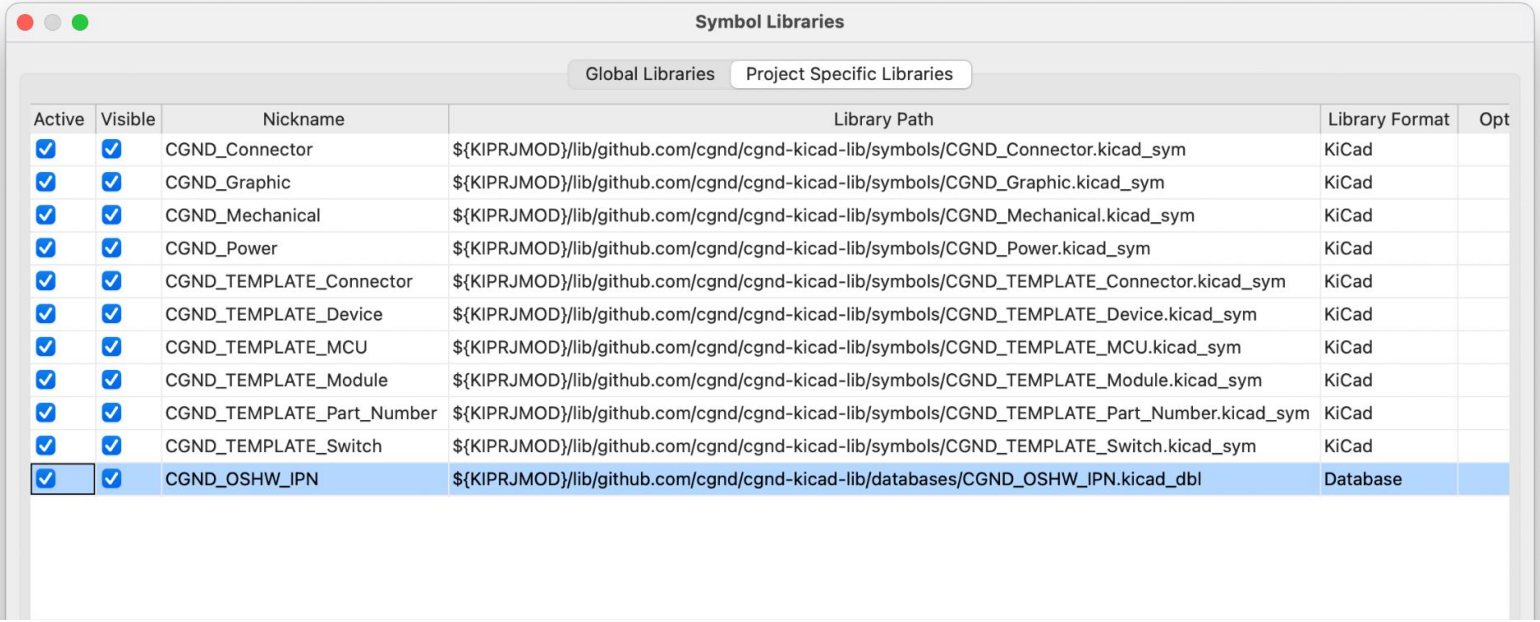
KICAD // DATABASE LIBRARY INTEGRATION

Map Aligni parameters to symbol properties

```
"properties": {
  "description": "description",
  "keywords": "x_keywords",
  "exclude_from_bom": "x_exclude_from_bom",
  "exclude_from_board": "x_exclude_from_board",
  "exclude_from_sim": "x_exclude_from_sim
}
```

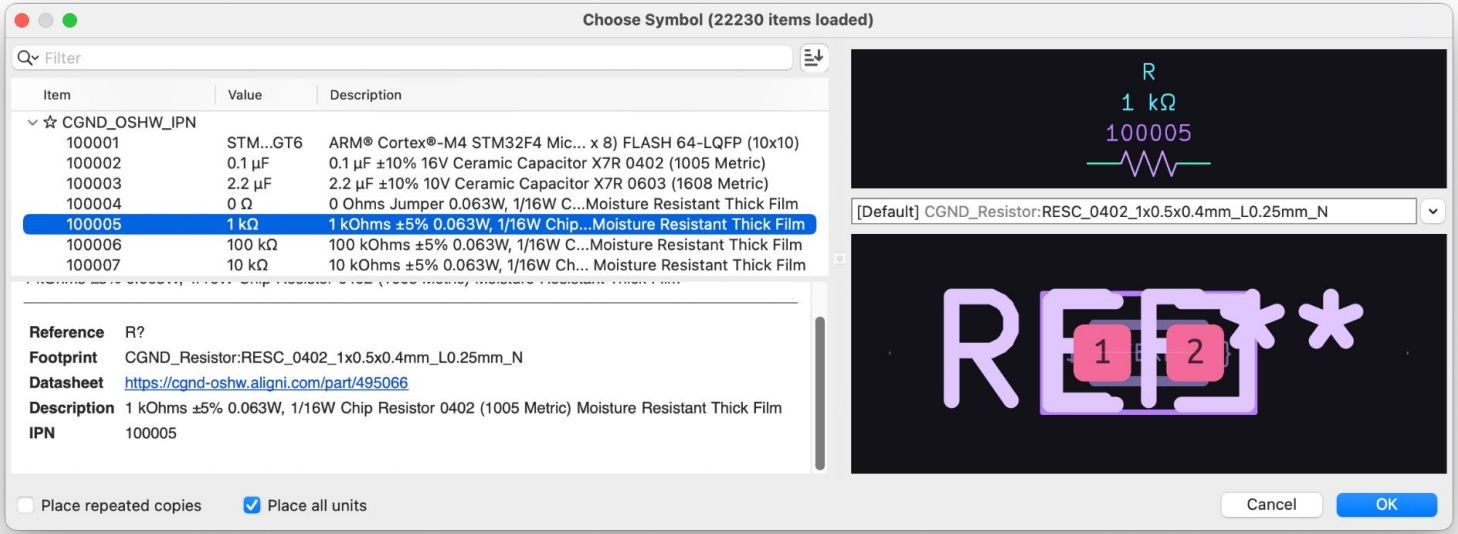
KICAD // DATABASE LIBRARY INTEGRATION

Add `*.kicad_dbl` config file to the symbol library table



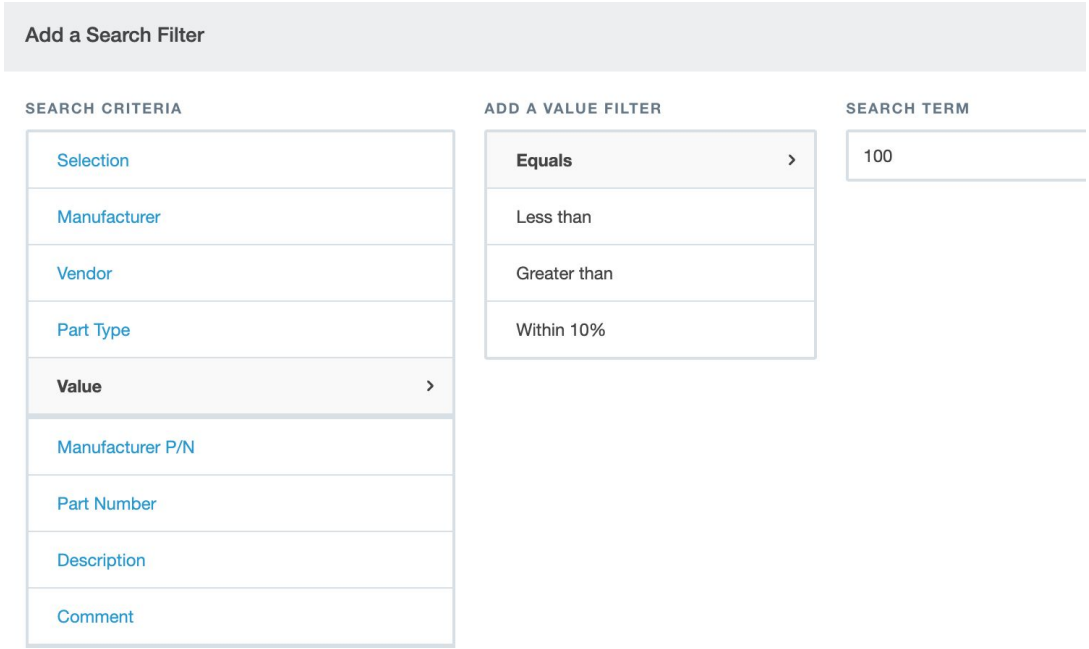
KICAD // DATABASE LIBRARY INTEGRATION

Now we can place Aligni parts in a KiCad schematic! 🎉



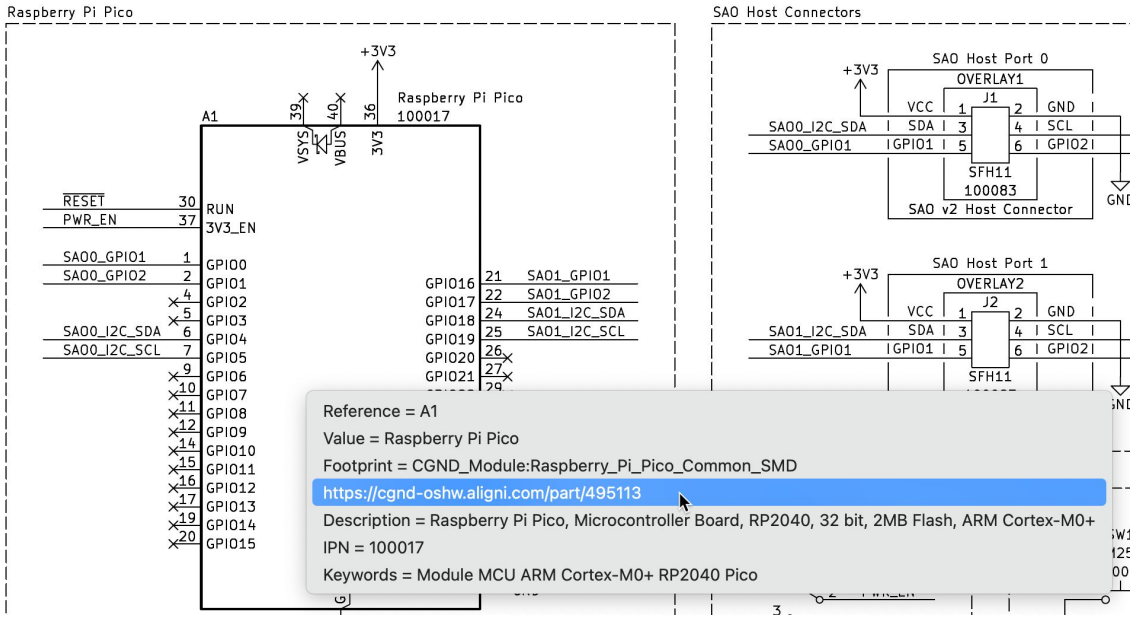
KICAD // DATABASE LIBRARY INTEGRATION

Parametric part search for library parts in Aligni



KICAD // DATABASE LIBRARY INTEGRATION

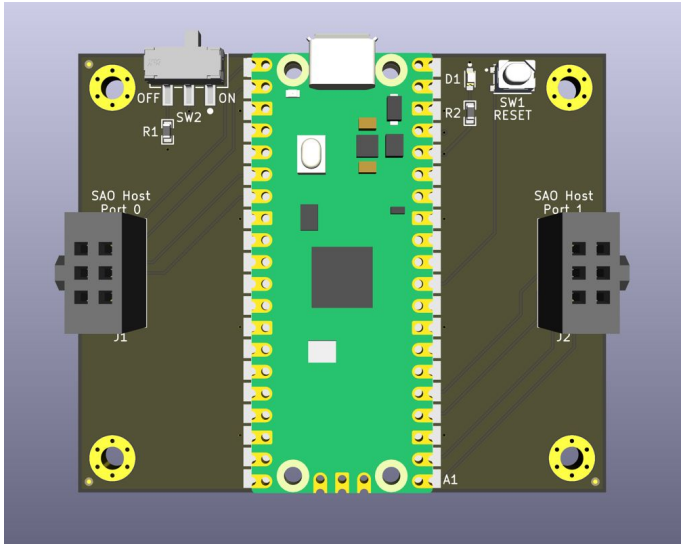
Links from schematic editor (“D”) & PDFs back to Aligni

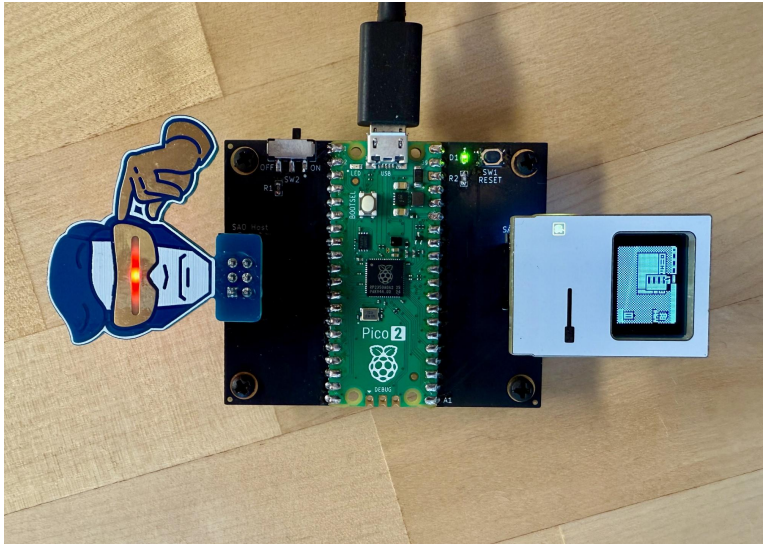



ALIGNI //
BOM IMPORT EXAMPLE
Solder
NinJa Pen
There is no cloud
Hacker.
DEFCON
BAY AREA

ALIGNI // BOM IMPORT EXAMPLE

Example assembly: Raspberry Pi Pico SAO Host board





https://github.com/cgnd/rpi-pico-sao-host

ALIGNI // ASSEMBLY BOM IMPORT

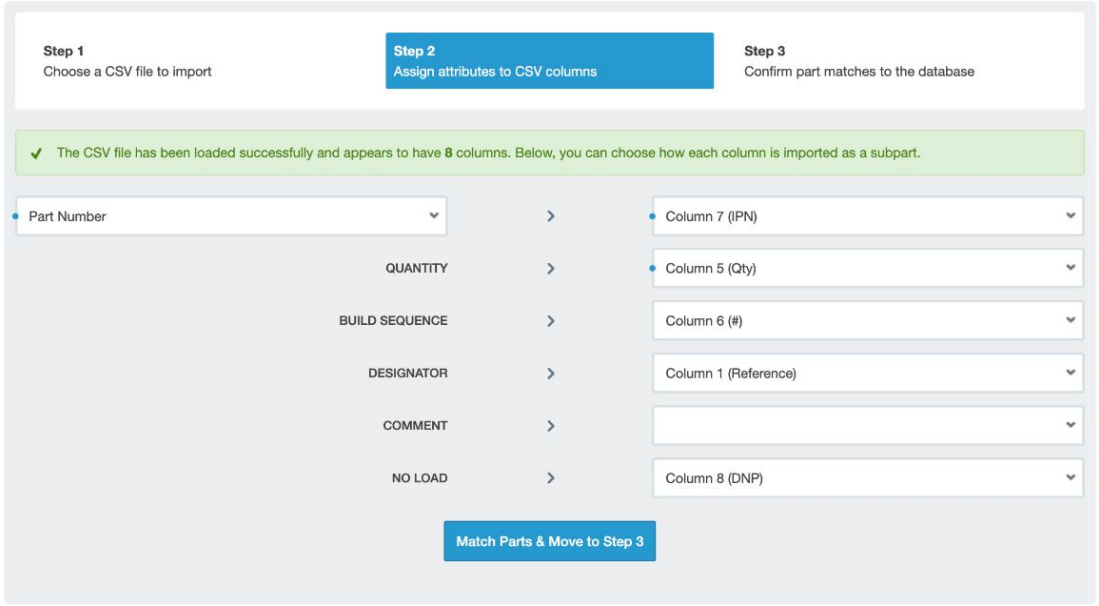
Export the KiCad schematic BOM as a CSV file

RPi_Pico_SAO_Host_v2_E...

# #	# Qty	Reference	# IPN	Value	Description	Datasheet	DNP
1	1	A1	100017	Raspberry Pi Pico	Raspberry Pi Pico, Microc	https://cgnd-oshw.aligı	
2	1	D1	100089	Green	Green 570nm LED Indicati	https://cgnd-oshw.aligı	
3	1	DOC1	100093	Pico SAO Host v2 Schematic	Raspberry Pi Pico SAO Ho	https://cgnd-oshw.aligı	
4	2	J1,J2	100083	SFH11	6 Position Header Connec	https://cgnd-oshw.aligı	
5	1	PCB1	100092	Pico SAO Host v2 PCB	Raspberry Pi Pico SAO Ho	https://cgnd-oshw.aligı	
6	1	R1	100091	100 Ω	100 Ohms ±5% 0.1W, 1/1C	https://cgnd-oshw.aligı	
7	1	R2	100090	560 Ω	560 Ohms ±5% 0.1W, 1/1C	https://cgnd-oshw.aligı	
8	1	SW1	100069	PTS810SJM250SMTRLFS	Tactile Switch SPST-NO Tc	https://cgnd-oshw.aligı	
9	1	SW2	100051	JS102011SAQN	Slide Switch SPDT Surfac	https://cgnd-oshw.aligı	

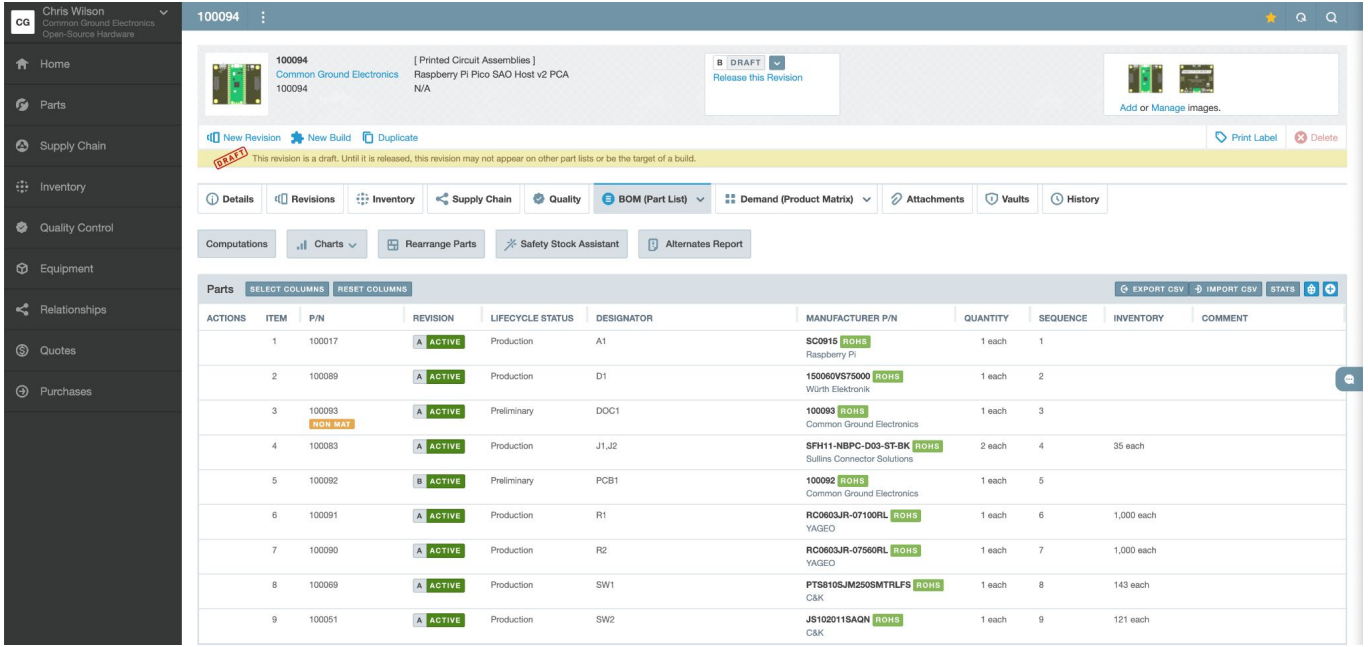
ALIGNI // ASSEMBLY BOM IMPORT

Aligni import: map CSV columns to Aligni BOM columns



ALIGNI // ASSEMBLY BOM IMPORT

Assembly BOM is populated based on parts from CSV



04

REVISIONS

How to use revisions in PLM

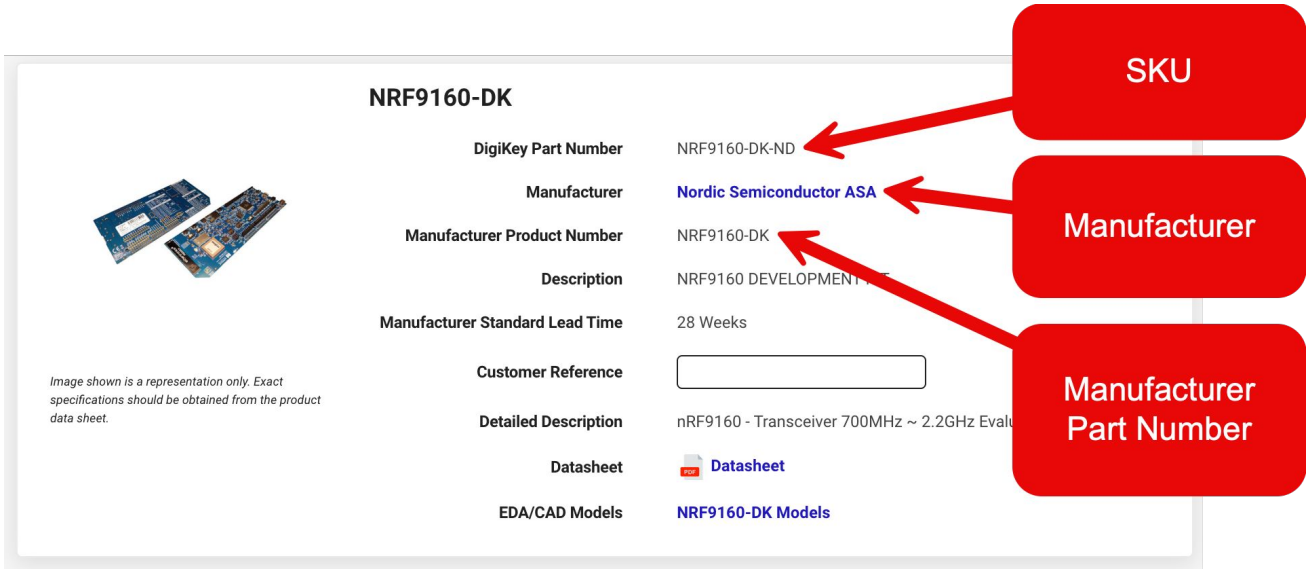
ALIGNI // PART REVISION BEST PRACTICES

- Parts in Aligni have “Revisions”
- Revisions allow tracking changes to a part over time
- **IMPORTANT:** many external systems (e.g. inventory management systems) **do NOT track part revisions**!
- Customers will typically buy from distributors using your **part number** only*

*Sometimes it’s possible to purchase a specific revision of a part when purchasing directly from the manufacturer

ALIGNI // PART REVISION BEST PRACTICES

- Example: DigiKey SKU = Manufacturer + MPN
- No way for customer to specify revision in order



ALIGNI // PART REVISION BEST PRACTICES

How should revisions be used with PLM part numbers?

Best Practice: different *revisions* of the same part number should be **interchangeable**.

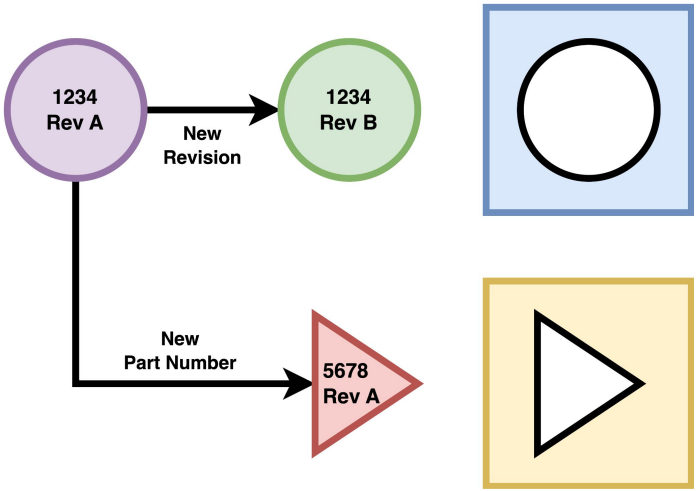
Use the "**Form, Fit, and Function**" (3F) rule:

> *If two parts have the same "form, fit, and function" (and sometimes "formulation") they can be substituted for one another.*

ALIGNI // PART REVISION BEST PRACTICES

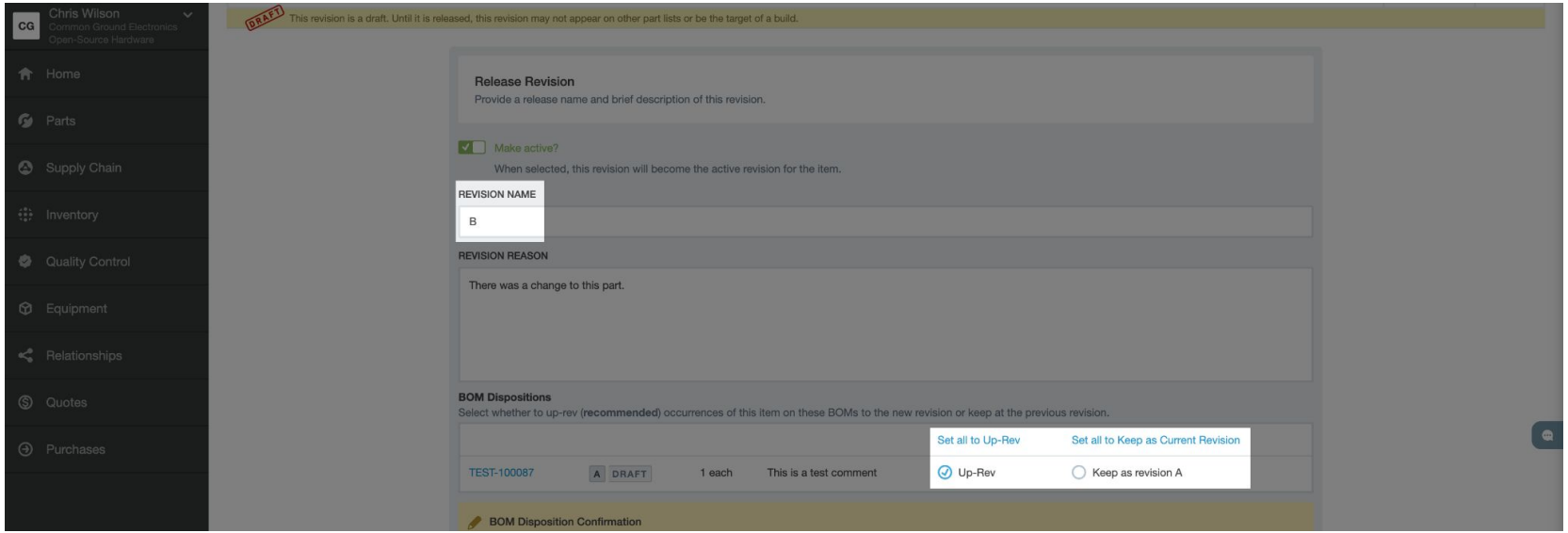
A general rule for dealing with changes to a part:

1. F/F/F **compatible** changes roll the **revision** of a part.
2. F/F/F **incompatible** changes **require a new part number**.



ALIGNI // PART REVISION BEST PRACTICES

Benefit: if revisions are following F/F/F methodology, PLM can “Up-Rev” all BOMs to latest part revision.

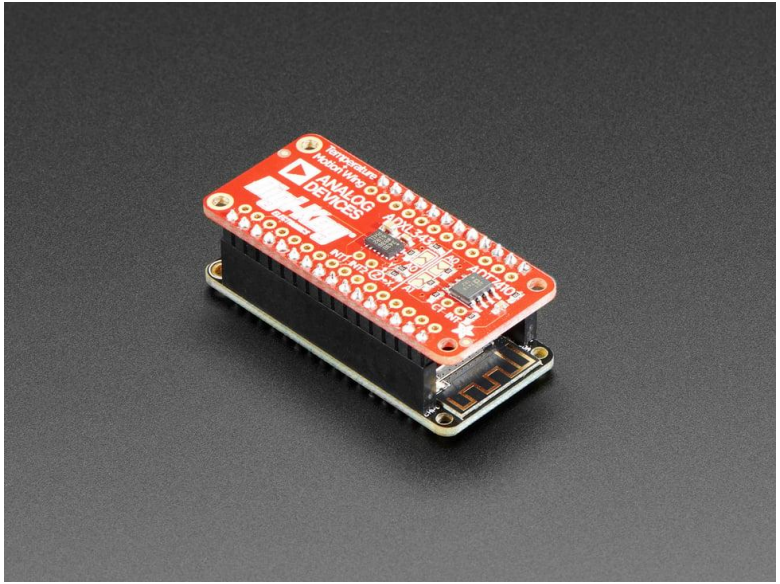


ALIGNI // PART REVISION BEST PRACTICES

This minimizes “churn” to assemblies when rev changes.

Example revision change:

```
SENSOR-NODE (Rev B)
├── SENSOR-BOARD (Rev C)
│   ├── RESISTOR (~~Rev A~~) → (Rev B)
│   ├── ACCELEROMETER (Rev A)
│   └── TEMP-SENSOR (Rev A)
└── FEATHER-BOARD (Rev A)
```



https://www.adafruit.com/product/4147

ALIGNI // PART REVISION BEST PRACTICES

❌ non-interchangeable: Triggers impact analysis for parent assemblies*

```
SENSOR-NODE (Rev B) → (???)
├── SENSOR-BOARD (Rev C) → (???)
│   ├── RESISTOR (~~Rev A~~) → (Rev B)
│   ├── ACCELEROMETER (Rev A)
│   └── TEMP-SENSOR (Rev A)
└── FEATHER-BOARD (Rev A)
```

✅ interchangeable: changes do not “roll up” the hierarchy

```
SENSOR-NODE (Rev B)
├── SENSOR-BOARD (Rev C)
│   ├── RESISTOR (~~Rev A~~) → (Rev B)
│   ├── ACCELEROMETER (Rev A)
│   └── TEMP-SENSOR (Rev A)
└── FEATHER-BOARD (Rev A)
```

*Repeat for *every* assembly BOM that uses the `RESISTOR` part...

Note: if you work in a highly regulated industry (medical, aerospace, etc), you most likely have a change management process that requires formally documenting all revision changes.

ALIGNI // PART REVISION BEST PRACTICES

A, B, C, … vs. 1, 2, 3, … vs. v1.0.0, v1.0.1, v1.1.0, v2.0.0, …

Lots of options! You can use any of these in Aligni.

ASME Y14.35 recommends revision letters A, B, C, …

“The revision letter is the identification of the revision level of the sheet or drawing. Upper case letters shall be used in sequence beginning with A and omitting letters “I,” “O,” “Q,” “S,” “X,” and “Z” …”

05

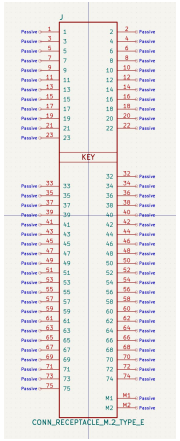
LIMITATIONS

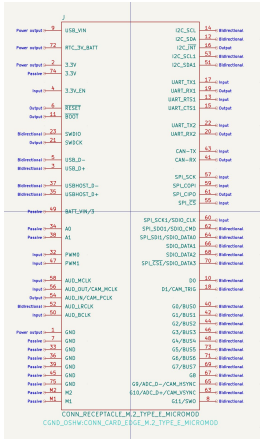
Issues with the current integration

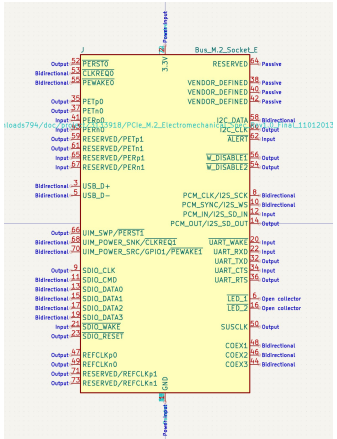
KICAD // ALTERNATE SYMBOLS

Currently no way to assign multiple symbols to a part:

Connector Generic MicroMod PCIe M.2



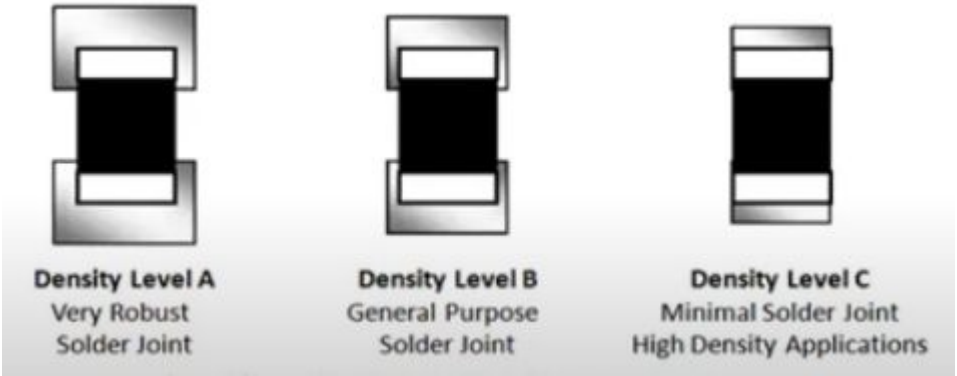




https://gitlab.com/kicad/code/kicad/-/issues/12380

KICAD // ALTERNATE FOOTPRINTS

Assigning multiple footprints to a part should be supported, but it wasn't working when I tried it:



https://gitlab.com/kicad/code/kicad/-/issues/13587

ALIGNI // USAGE LIMITATIONS

Free-tier usage limitations:

Usage		
	PUBLIC	ADD-ONS
Attachments	Unlimited	-
Builds	10	-
Collaborator Seats	0	0 / 5
Contacts	Unlimited	-
Customers	Unlimited	-
Demand Entries	Unavailable	-
ECO	5	-
ECR	5	-
Equipment	Unavailable	-
Inventory History	Unlimited	-
Inventory Sublocations	Unlimited	-
Inventory Units	Unlimited	-
Manufacturers	Unlimited	-
Material Transfers	Unlimited	-
Octopart Queries	100	-
Part Collections	Unlimited	-
Part Parameter Fields	Unlimited	-
Part Types	Unlimited	-
Parts	1000	-
Purchase Orders	Unlimited	-
Purchases	1	-
Quote Requests	1	-
Quote Responses	Unlimited	-
Quotes	Unlimited	-
UltraCart Integration	Unavailable	-
Units	Unlimited	-
Usage Reports	Unavailable	-
Vendors	Unlimited	-
Viewer Seats	Unavailable	-
Warehouses	3	-
WooCommerce Integration	Unavailable	-
Xero Integration	Unavailable	-

ALIGNI // ALIGNI DB ONLY HAS “PARTS” TABLE

KiCad database libraries feature supports creating multiple libraries, one for each table in the database.

However, Aligni Replicator software only generates a database with a single `parts` table.

As a result, it’s only possible to generate a single KiCad library for all parts in the Aligni Item Master. It’s not possible to have separate libraries based on part type.

THANKS

Do you have any questions?

chris@cgnd.dev

https://cgnd.dev

06

ADDITIONAL TIPS

Additional information that we
didn’t have time to cover in the talk

GIT // STORING SQLITE DATA IN A GIT REPO

- SQLite is a binary file format, not ideal for Git
- Hard to diff changes to a SQLite database (e.g. `sqldiff` won't show changes to internal metadata)
- Use https://github.com/simonw/sqlite-diffable to dump the database schema/data to JSON format
- JSON representation is formatted to be "diffable"
- Check the JSON representation into the Git repo instead of the SQLite database

ALIGNI // ADDING PART URL TO REPLICATOR DB

An `x_aligni_part_url` column can also be added to the Replicator database using SQL commands.

```sql
ALTER TABLE parts
ADD COLUMN x_aligni_part_url TEXT;

UPDATE parts
SET x_aligni_part_url = 'https://cgnd-oshw.aligni.com/part/' || id;
```

ALIGNI // ADDING PART URL TO REPLICATOR DB

This can be scripted in Python via `sqlite3` package:

```python
import sqlite3

conn = sqlite3.connect(db)
cursor = conn.cursor()
cursor.execute(f"""
    ALTER TABLE parts
    ADD COLUMN x_aligni_part_url TEXT
""")
cursor.execute(f"""
    UPDATE parts
    SET x_aligni_part_url = 'https://cgnd-oshw.aligni.com/part/' || id
""")
conn.commit()
conn.close()
```

07

RESOURCES

Additional resources

RESOURCES

- **Aligni Pubic Organizations Page**
  https://app.aligni.com/catalog
- **Aligni Documentation (highly recommended)**
  https://docs.aligni.com/
- **KiCad Database Libraries Documentation**
  https://docs.kicad.org/9.0/en/eeschema/eeschema.html#database-libraries

SQLITE ODBC DRIVER EXAMPLES

- **macOS**
  https://cdwilson.dev/articles/kicad-database-libraries-on-macos/
- **Windows**
  https://github.com/SumantKhalate/KiCad-libdb
- **Linux**
  https://datawookie.dev/blog/2015/09/setting-up-odbc-for-sqlite-on-ubuntu/

PLM RESOURCES

- **Part numbering system design**
  https://www.buyplm.com/plm-good-practice/part-numbering-system-software.aspx
- **Form/Fit/Function guide in Aligni**
  https://docs.aligni.com/guides/form-fit-function/
- **Part interchangeability best practices**
  https://www.buyplm.com/plm-good-practice/form-fit-function-interchangeable-parts.aspx

ALTERNATIVES TO CHECK OUT

Alternatives people mentioned but I haven't tried yet:

- https://partsbox.com/
- https://inventree.org/
- https://durolabs.co/
- https://github.com/git-plm/gitplm
- https://binner.io/

G2 PLM Reviews

- https://www.g2.com/categories/product-lifecycle-management-plm