KiCad 5: Projektportierung von KiCad4 (in Bearbeitung)

Die neue Version von KiCad bringt einige grundlegende Änderungen mit sich, was leider den Aufwand erfordert, dass ein KiCad4-Projekt portiert werden muss, nachdem es kein zurück mehr gibt.

Der wesentliche Unterschied ist nun, dass ein Symbol bisher den z.B. den Namen 'R' gehabt hat, und nun 'Device:R' genannt werden muss.

Der zugehörige Blogeintrag auf der KiCad Website ist hier zu finden:
http://kicad-pcb.org/post/symbol-lib-table/
Diese Anleitung hab ich bei der Druchführung nicht immer ganz verständlich bzw. eindeutig empfunden, daher mein Artikel zum Thema.

Systemweite Vorbereitungen

Config Verzeichnis

Ich verwende derzeit noch KiCad 4 und 5 parallel. Daher habe ich 2 Ordner für die config eingerichtet:

```
/home/karl/.config/kicad4
/home/karl/.config/kicad5
```

, wobei einer der beiden per symlink immer auf

```
/home/karl/.config/kicad
```

verlinkt wird, je nachdem, welche Version ich gerade verwende.

Template Ordner

Im config ordner für KiCad5 habe ich auch den Template Ordner reinkopiert (von /usr/share/kicad/template), in dem auch das KiCad Projekt-Template sitzt: **kicad.pro**

Diese Datei darf keine Einträge mehr in der Gruppe [eeschema/libraries] haben:

kicad.pro

```
update=Mon 18 Apr 2018 22:56:56 MDT
version=1
last_client=kicad
[general]
version=1
RootSch=
```

```
BoardNm=
[pcbnew]
version=1
LastNetListRead=
UseCmpFile=1
PadDrill=0.600000000000
PadDrillOvalY=0.600000000000
PadSizeH=1.500000000000
PadSizeV=1.500000000000
PcbTextSizeV=1.500000000000
PcbTextSizeH=1.500000000000
PcbTextThickness=0.300000000000
ModuleTextSizeV=1.000000000000
ModuleTextSizeH=1.000000000000
ModuleTextSizeThickness=0.150000000000
SolderMaskClearance=0.050000000000
SolderMaskMinWidth=0.10000000000
DrawSegmentWidth=0.200000000000
BoardOutlineThickness=0.100000000000
ModuleOutlineThickness=0.150000000000
[cvpcb]
version=1
NetIExt=net
[eeschema]
version=1
LibDir=
[eeschema/libraries]
# must be empty for KiCad5
# refer to ~/.config/kicad/sym-lib-table
```

Bibliotheken

Ab KiCad5 gibt es eine neue Art der Bibliotheksverwaltung. Der automatische Download von GitHub wurde deaktiviert, man darf sich nun selber drum kümmern, wann und ob man Updates von GitHub in seine eigene Installation übernimmt - ausgezeichnet!

GitHub

Dazu habe ich die 3 Repositories von GitHub geklont und systematisch erreichbar abgelegt:

```
/home/karl/Team14/git/kicad5-footprints
/home/karl/Team14/git/kicad5-symbols
/home/karl/Team14/git/kicad5-packages3D
```

Das jeweilige Repo hat den 5er ursprünglich nicht im Namen, aber er sorgt hier für Klarheit.

sym-lib-table

Die Datei **sym-lib-table** muss von /home/karl/Team14/git/kicad5-symbols nach /home/karl/.config/kicad5 kopiert werden. Somit hat man alle GitHub Libs in die eigene Installation Systemweit eingebunden, ähnlich wie die schon bekannte **fp-lib-table**, die für die Footprints zuständig ist.

kicad_common

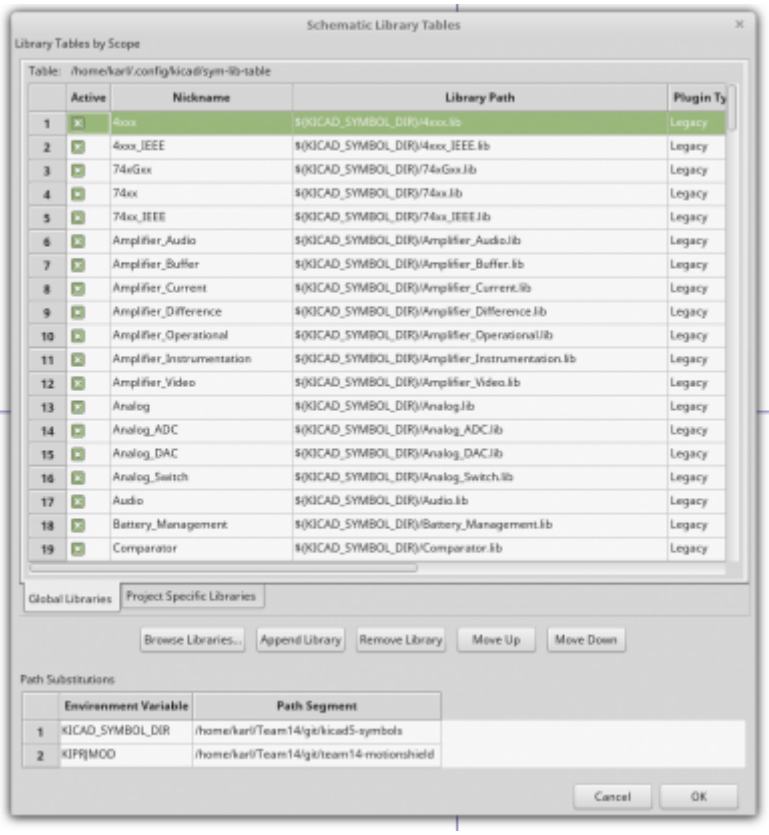
Die Datei **/home/karl/.config/kicad5/kicad_common** muss nun noch mit den neuen Pfaden geändert werden:

kicad_common

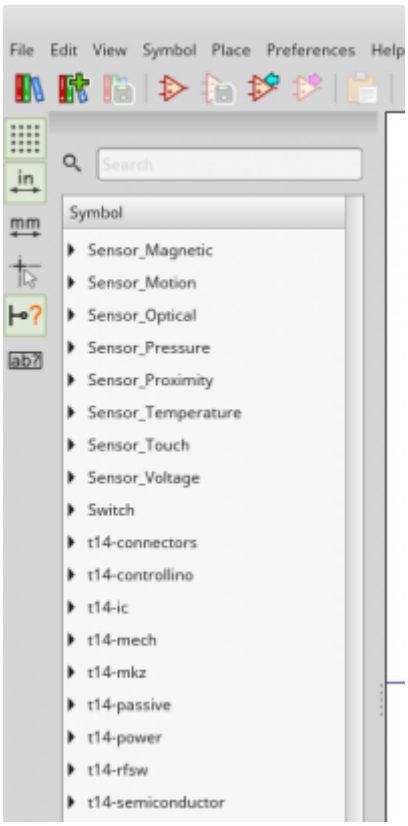
```
WorkingDir=/tmp/.mount_kicad5eesXIT/usr
ShowEnvVarWarningDialog=1
kicad_fplib_url=https://github.com/KiCad
Editor=/usr/bin/sublime-text
kicad_fplib_last_download_dir=/home/karl/MKZ/KiCad/github
UseIconsInMenus=1

[EnvironmentVariables]
KICAD_PTEMPLATES=/home/karl/.config/kicad/template
KIGITHUB=https://github.com/KiCad

# 3D Models:
KISYS3DMOD=/home/karl/Team14/git/kicad5-packages3D

# Footprints:
KISYSMOD=/home/karl/Team14/git/kicad5-footprints

# Schematic Symbols:
KICAD_SYMBOL_DIR=/home/karl/Team14/git/kicad5-symbols
```

Eigene Bibliotheken

Die eigenen Bibliotheken müssen nun auch noch in die systemweite sym-lib-table eingetragen werden, was am einfachsten über den Biblilotheks Manager geht. Dazu wird Kicad5 gestartet, dann der **Symbol Library Editor** (OPV-Symbol) gestartet und hier **Menu → Preferences → Manage Symbol Libraries…** klicken:



Hier kann man bequem seine eigenen Libs auswählen und eintragen. Ich hab unsere Team14 Libs gleich noch mit einem **t14-** Präfix ausgestattet, sodass diese in der langen Liste gut aufzufinden sind:



KiCad4 Projekt vorbereiten

Nachdem wir unser System für KiCad5 eingerichtet haben, geht es um das erste Projekt, das portiert werden soll. Hierbei sind einige Punkte zu beachten.

Sicherung des Projektes

All unsere KiCad-Projekte sind ohnehin mit git versioniert. Das kann ich nur dringend empfehlen. Hier stellen wir nur noch sicher, dass das Repo auch sauber ist, das heißt keine Änderungen seit dem letzen Commit vorhanden sind.

Alle Bibliotheken vorhanden?

Wird ein Schaltplan geöffnet, prüft KiCad, ob alle Bibliotheken auffindbar sind, die in der Projektdatei (z.B. motionshield.pro) eingetragen sind. Kommt hier eine Fehlermeldung, muss dies unbedingt bereinigt werden, indem die .pro-Datei angepasst wird.

Vorhandene Cache und Rescue Libs

Sollten im Projekt vorhandene *-cache.lib oder *-rescue.lib Bibliotheken vorhanden sein, können diese einfach so belassen werden [1)]

```
motionshield-cache.lib
motionshield-rescue.lib
```

Nochmals Sichern

Bevor es nun ernst wird die bisherige Arbeit sichern,

```
git add --all
git commit
```

damit man schnell per

```
git reset --hard
```

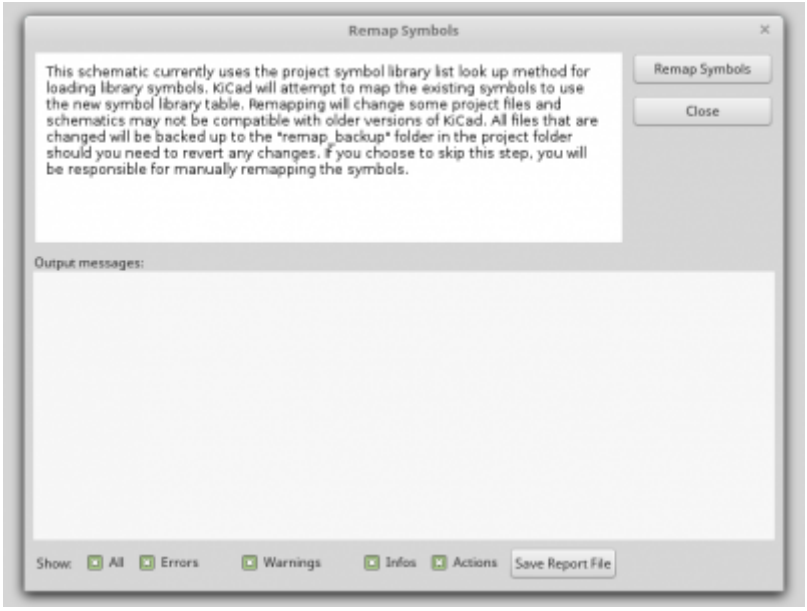
wieder alles zurückstellen kann.

Schaltplan neu zuordnen (remap)

```
WICHTIG: KiCad schließen, bevor wir weitermachen
```

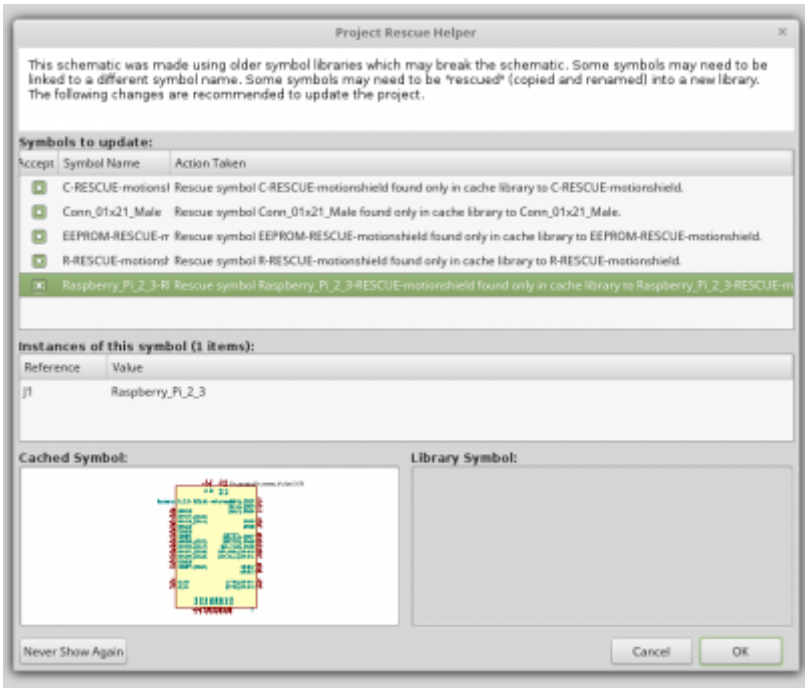
Dazu Starten wir KiCad und öffnen den Schaltplan Editor (Schematic Layout Editor).

Hier werden wir von einem Dialog begrüßt, der das Zuordnen der bisherigen Symbole nun auf die konkreten Bibliotheken übernehmen soll.



Hier können wir nun Dank der Sicherung bedenkenlos auf **Remap Symbols** klicken.

Sind Rescue-Symbole involviert, kommt der von KiCad4 bekannte Dialog, den man mit OK wieder schließt:



Der Dialog sollte dann weder Fehler noch Warnungen anzeigen. Mit Close wird der Dialog beendet und der „neue Schaltplan“ erscheint.

Speichern

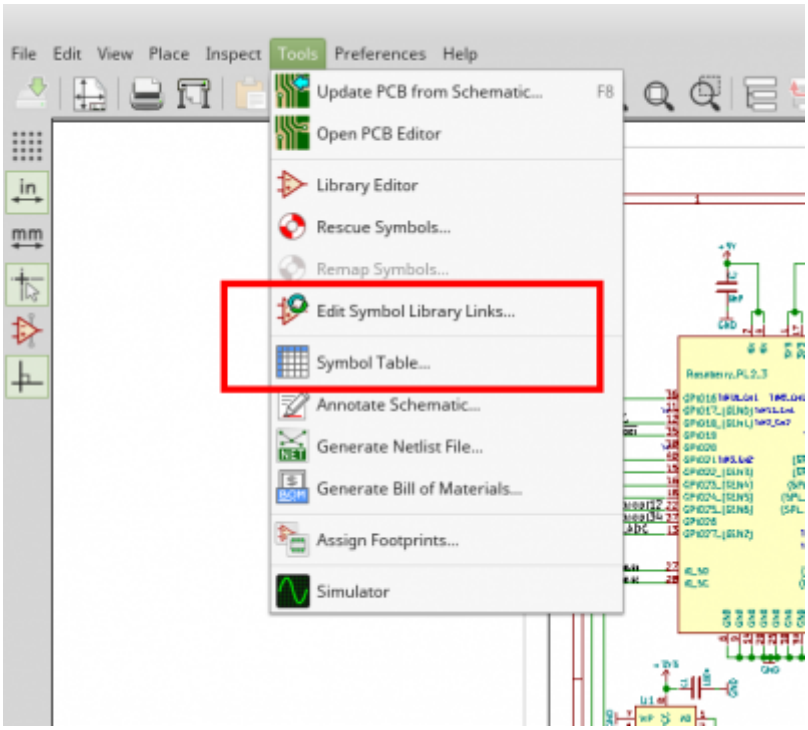
Klickt man auf Speichern, sollte das ohne Fehlermeldung gehen. Wenn ein Symbol mit alter Bezeichnung vorhanden ist, also keine Lib mit Doppelpunkt vorangestellt, gibt es beim Speichern eine Fehlermeldung, die darauf aufmerksam macht.

Backup löschen und per git sichern

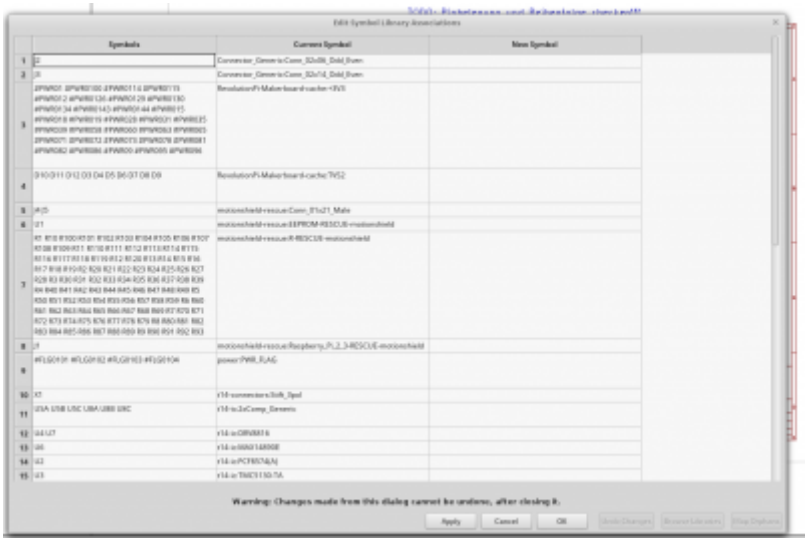
Nun kann man wieder alles schließen und den erstellten **rescue-backup** Ordner löschen wir und sichern das Projekt mit einem git-commit.

Nachbesserungen

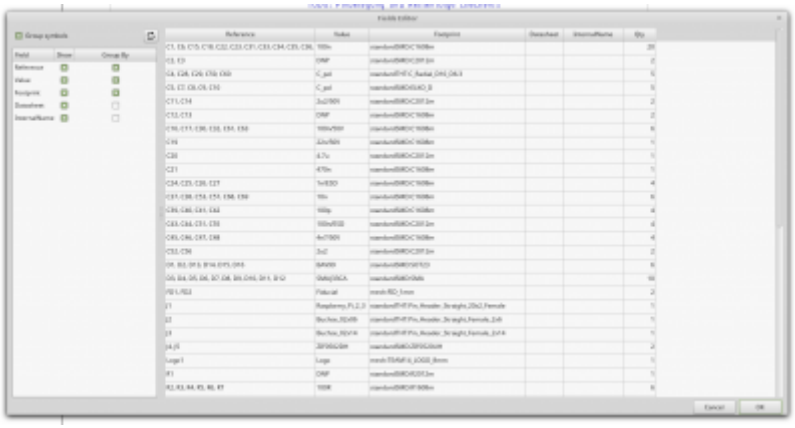
Möchte man sich einen Überblick über die Zuordnungen verschaffen gitb es 2 neue Werkzeuge dafür:



Symbol Library Associations Editor



Fields Editor



Vermutlich haben diese Werkzeuge die Fähigkeit, den von uns bisher verwendeten und weiterentwickelten KiCad Partslist Editor zu ersetzen. Dieser ist moementan für mit KiCad5 (noch) nicht kompatibel.

Footprints

Die Footprints sollten aufgrund der gleich bleibenden Einträge in der fp-lib-table ohne Probleme gefunden und übernommen werden.

3D Modelle

Hier gibt es leider massive Probleme. In den bisherigen Footprints wurde der Pfad zu einem 3D-Modell nur über eine bestimmte Umgebungsvariable (KISYS3DMOD) definiert, und diese wurde beim Link zum 3D-Modell erst gar nicht angeführt.

Beispiel anhand eines TQFP-48:

TQFP-48-EP_7x7mm_Pitch0.5mm.kicad_mod

```
  ...
  (model Housings_QFP.3dshapes/TQFP-48_7x7mm_Pitch0.5mm.wrl
    (at (xyz 0 0 0))
    (scale (xyz 1 1 1))
    (rotate (xyz 0 0 0))
  )
)
```

Was es hier bräuchte, wäre ein Skript, das in allen *.kicad_mod Dateien den Eintrag `(model` durch z.B. `(model ${KICAD4_3D_DIR}` ersetzt.

Links

https://forum.kicad.info/t/installing-both-kicad-nightly-and-stable-versions/9751/16

kicad, software, deutsch, artikel

1)

Versuche, diese vorher umzubenennen und in die .pro-Datei einzutragen scheiterten