



# *Dateiverwaltung & Versionskontrolle mit Git*

## *Training Teil 1*



Inh. Dipl. Ing. Mario Blunk

Buchfinkenweg 3  
99097 Erfurt / Deutschland

Telefon 0361 6022 5184

Email [info@blunk-electronic.de](mailto:info@blunk-electronic.de)

Internet [www.blunk-electronic.de](http://www.blunk-electronic.de)

# Voraussetzungen

1. Kurzinfo

[http://www.blunk-electronic.de/pdf/git\\_einfuehrung.pdf](http://www.blunk-electronic.de/pdf/git_einfuehrung.pdf)

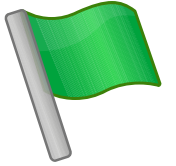
2. Grundkenntnisse zur Arbeit mit Kommandozeile

3. Mut zu etwas Neuem



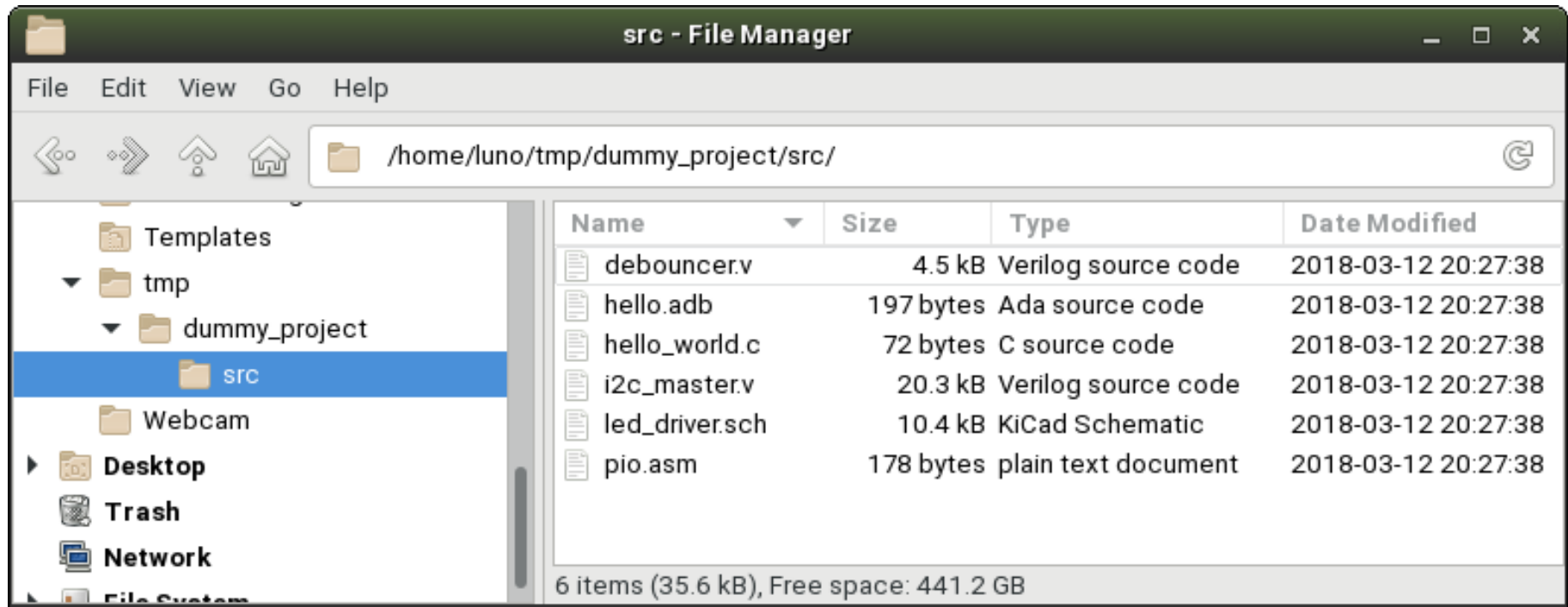
8h

# ***Teil A - Überblick***



1. Repository anlegen (init)
2. Dateien auf Index setzen (add)
3. Änderungen einchecken (commit)
4. Projektverlauf einsehen (log)
5. beliebigen Arbeitsstand einsehen (checkout)
6. Dateien ignorieren
7. Dateien löschen/umbenennen (rm/mv)

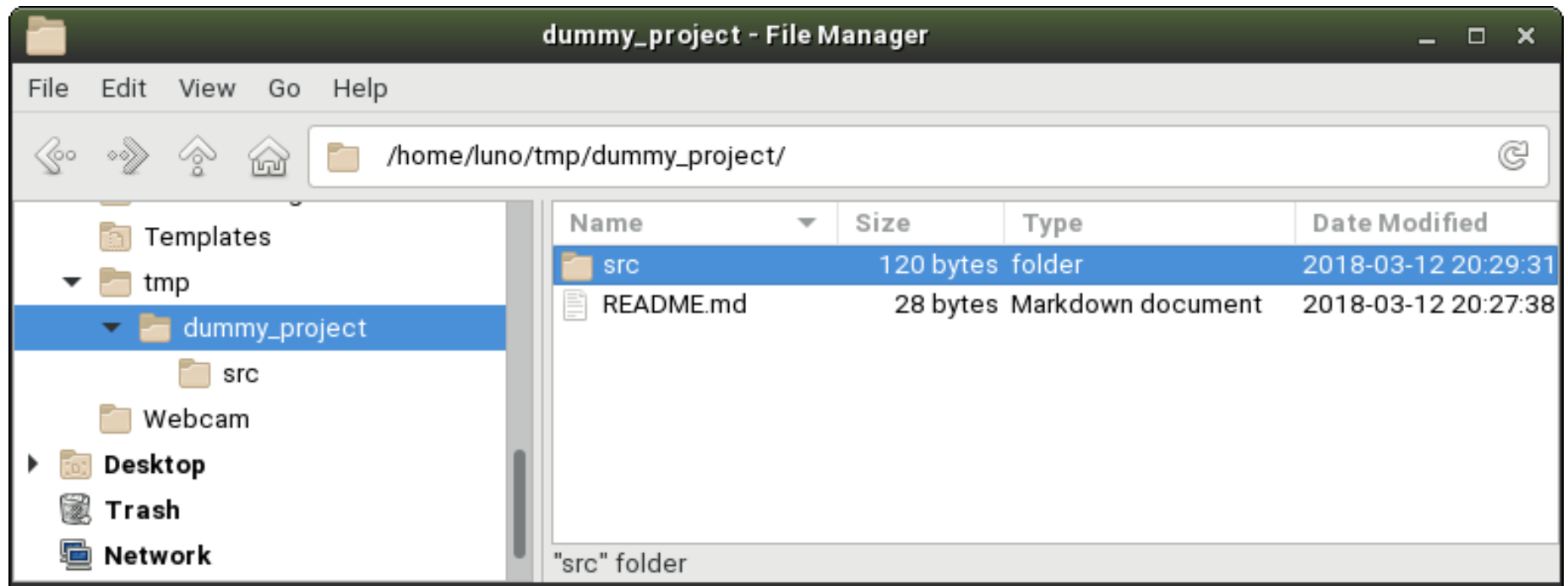
# Unser Projektverzeichnis



Beliebige Unterverzeichnisse möglich

- Mechanik-Konstruktion (FreeCad, ...)
- ECAD (KiCad, EAGLE, ...)
- Dokumentation (LibreOffice, Sphinx, ...)

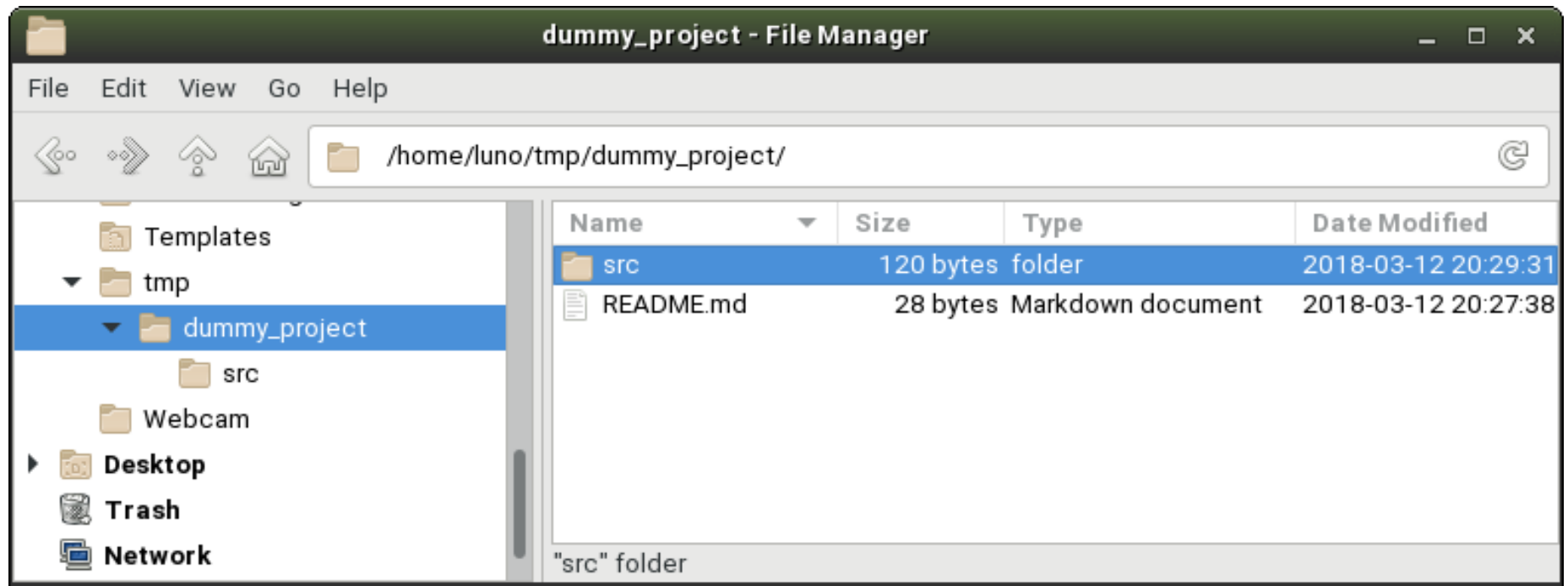
# Repository anlegen #1



in Wurzelverzeichnis des Projektes wechseln:

```
$ cd tmp/dummy_project/
```

# Repository anlegen #2



Repository initialisieren:

```
$ git init
```

```
Initialized empty Git repository in  
/home/luno/tmp/dummy_project/.git/
```

# *Repository anlegen #3*

Status abfragen:

```
$ git status
```

```
On branch master
```

```
No commits yet
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will  
be committed)
```

```
README.md
```

```
src/
```

```
nothing added to commit but untracked files  
present (use "git add" to track)
```



# *Datei auf Index setzen*

```
$ git add README.md
```

```
$ git status
```

```
On branch master
```

```
No commits yet
```

```
Changes to be committed:
```

```
(use "git rm --cached <file>..." to unstage)
```

```
new file:   README.md
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will  
be committed)
```

```
src/
```

# *Datei einchecken*

```
$ git commit -m "first commit"
```

```
[master (root-commit) 1d154f5] first commit  
1 file changed, 2 insertions(+)  
create mode 100644 README.md
```

```
$ git status
```

```
On branch master
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will  
be committed)
```

```
src/
```

```
nothing added to commit but untracked files  
present (use "git add" to track)
```

# *Weitere Datei auf Index*

```
$ git add src/pio.asm
```

```
$ git status
```

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

**new file: src/pio.asm**

Untracked files:

(use "git add <file>..." to include in what will be committed)

**src/debouncer.v**

**src/hello.adb**

**src/hello\_world.c**

**src/i2c\_master.v**

**src/led\_driver.sch**

# *Weitere Datei einchecken*

```
$ git commit -m "pio configuration"
```

```
$ git status
```

```
On branch master
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will  
be committed)
```

```
src/debouncer.v
```

```
src/hello.adb
```

```
src/hello_world.c
```

```
src/i2c_master.v
```

```
src/led_driver.sch
```

```
nothing added to commit but untracked files  
present (use "git add" to track)
```

# *Weitere Dateien auf Index*

```
$ git add src/*.v
```

```
$ git status
```

```
On branch master
```

```
Changes to be committed:
```

```
(use "git reset HEAD <file>..." to unstage)
```

```
new file:   src/debouncer.v
```

```
new file:   src/i2c_master.v
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will  
be committed)
```

```
src/hello.adb
```

```
src/hello_world.c
```

```
src/led_driver.sch
```

# *Weitere Dateien einchecken*

```
$ git commit -m "alle Verilog-Dateien"
```

```
$ git status
```

```
On branch master
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will  
be committed)
```

```
src/hello.adb
```

```
src/hello_world.c
```

```
src/led_driver.sch
```

```
nothing added to commit but untracked files  
present (use "git add" to track)
```

# *Weitere Dateien ...*

```
$ git add src/*
```

```
$ git commit -m "der Rest"
```

```
$ git status
```

```
On branch master
```

```
nothing to commit, working tree clean
```

# Projektverlauf #1

\$ gitk

The screenshot shows the gitk GUI for a project named 'dummy\_project'. The window title is 'dummy\_project: All files - gitk'. The interface is divided into several sections:

- Commit History:** A list of commits on the left side, with the 'master' branch highlighted. The commits are:
  - den Rest
  - alle Verilog-Dateien
  - pio configuration
  - first commit
- Commit Details:** A table showing the author and timestamp for the selected commit (SHA1 ID: 753eb0fc82a7926902efc0a321168b154e438e60).

Author	Timestamp
Mario Blunk <mario.blunk@blunk-electronic.de>	2018-03-12 21:38:49
Mario Blunk <mario.blunk@blunk-electronic.de>	2018-03-12 21:34:24
Mario Blunk <mario.blunk@blunk-electronic.de>	2018-03-12 21:25:13
Mario Blunk <mario.blunk@blunk-electronic.de>	2018-03-12 21:14:12
- Search and Navigation:** A search bar with 'commit containing:' and a 'Find' button. Below it, a 'Search' input field and a 'Patch' button.
- Diff View:** The main area shows a diff for the file 'src/debouncer.v'. The diff shows a new file mode 100644 and the following code:

```
new file mode 100644
index 0000000..eb753b2
@@ -0,0 +1,91 @@
+// outputs a H-pulse after input has been constant high for
+
+module debouncer (clk, in, out, reset_n);
+
+    output reg    out; // debounced output
+    input         in; // bouncing input signal
+    input         clk;
+    input         reset_n;
```
- Comments:** A section on the right side showing comments for the files 'src/debouncer.v' and 'src/i2c\_master.v'.



# Projektverlauf #2

```
$ git log
```

```
commit eb376971479bbe207df68e26f95a2069213e0196 (HEAD -> master)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:38:49 2018 +0100
```

der Rest

```
commit 753eb0fc82a7926902efc0a321168b154e438e60
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:34:24 2018 +0100
```

alle Verilog-Dateien

- 
- 
- 

```
commit 1d154f59ccc91dd6eddc232570d5d357c3628411
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:14:12 2018 +0100
```

first commit

# ***Arbeitsstand einsehen #1***

```
$ git checkout 753e
```

```
Note: checking out '753e'.
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using `-b` with the checkout command again. Example:

```
git checkout -b <new-branch-name>
```

```
HEAD is now at 753eb0f alle Verilog-Dateien
```

# Arbeitsstand einsehen #2

```
$ git log
```

```
commit 753eb0fc82a7926902efc0a321168b154e438e60 (HEAD)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:34:24 2018 +0100
```

alle Verilog-Dateien

```
commit 35da10d4228e60b5592836c97ddb853a81b2928a
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:25:13 2018 +0100
```

pio configuration

```
commit 1d154f59ccc91dd6eddc232570d5d357c3628411
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:14:12 2018 +0100
```

first commit

# Arbeitsstand einsehen #3

```
$ git checkout master
```

```
Previous HEAD position was 753eb0f alle Verilog-Dateien  
Switched to branch 'master'
```

```
$ git log
```

```
commit eb376971479bbe207df68e26f95a2069213e0196 (HEAD -> master)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:38:49 2018 +0100
```

den Rest

```
commit 753eb0fc82a7926902efc0a321168b154e438e60
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:34:24 2018 +0100
```

alle Verilog-Dateien

- 
- 
-

# *Dateien ignorieren #1*

Manche Dateien sollen von Git ignoriert werden:

- Binärdateien
- PDF
- Log-Dateien von Editoren (z.B. readme.md~)
- ...

**\$ git status**

On branch master

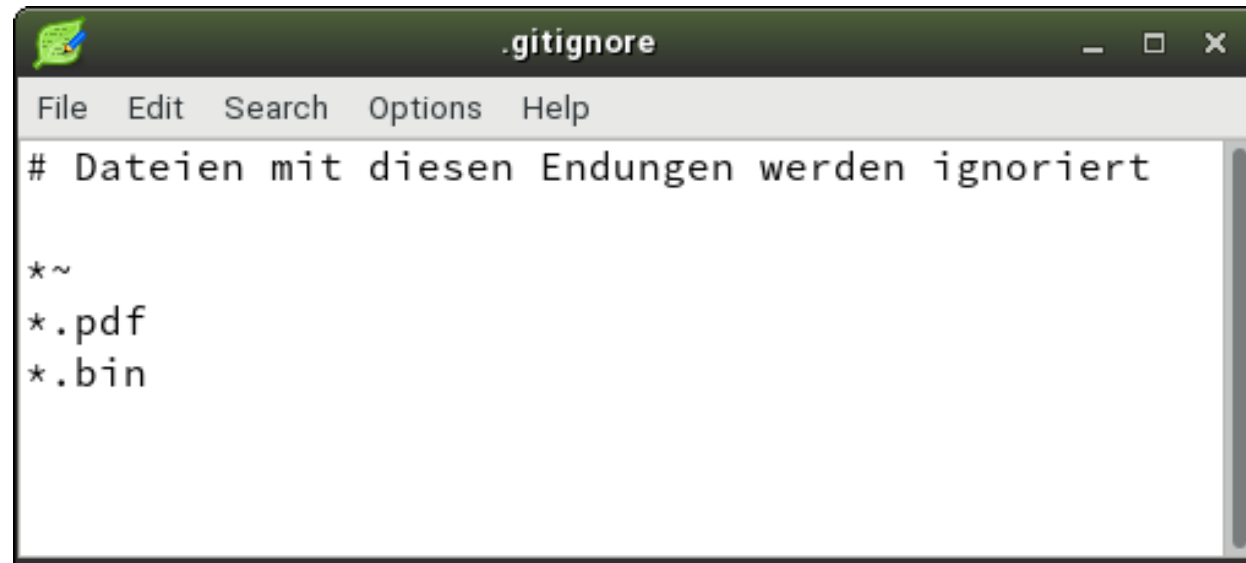
Untracked files:

(use "git add <file>..." to include in what will be committed)

**README.md~**

# Dateien ignorieren #2

Mit Texteditor im Wurzelverzeichnis des Projektes Datei mit Name `.gitignore` anlegen:



```
.gitignore
File Edit Search Options Help
# Dateien mit diesen Endungen werden ignoriert
*~
*.pdf
*.bin
```

**\$ git status**

On branch master

Untracked files:

(use "git add <file>..." to include in what will be committed)

**.gitignore**

# ***Dateien ignorieren #3***

Datei .gitignore auf Index und Einchecken:

```
$ git add .gitignore
```

```
$ git commit -m "Ignorierfilter eingebaut"
```

```
$ git status
```

```
On branch master
```

```
nothing to commit, working tree clean
```

# *Dateien/Verzeichnisse umbenennen*

```
$ git mv hello.adb hello_world.adb
```

```
$ git status
```

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

```
renamed:    hello.adb -> hello_world.adb
```

```
$ git commit -m "namen geaendert"
```

```
$ git mv src/ quellen
```

```
$ git status
```

```
$ git commit -m "quellverzeichnis geaendert"
```



# *Dateien/Verzeichnisse löschen*

```
$ git rm hello.adb
```

```
$ git status
```

```
git status
```

```
On branch master
```

```
Changes to be committed:
```

```
(use "git reset HEAD <file>..." to unstage)
```

```
deleted:    hello.adb
```

```
$ git commit -m "aufgeraeumt"
```

```
$ git rm -r src/
```

```
rm 'src/debouncer.v'
```

```
rm 'src/hello.adb'
```

```
rm 'src/hello_world.c'
```

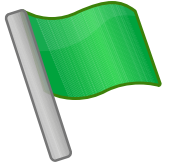
```
rm 'src/i2c_master.v'
```

```
rm 'src/led_driver.sch'
```

```
rm 'src/pio.asm'
```

```
$ git commit -m "quellen geloescht"
```

# ***Teil B - Überblick***



1. Datei zurücksetzen (checkout)
2. Datei aus Commit wiederherstellen (checkout)
3. einen Commit verwerfen
4. Marken setzen (tag)
5. Verzweigungen starten (branch)
6. Verzweigungen vereinen (merge)

# Datei zurücksetzen #1

Datei src/pio.asm mit Texteditor bearbeiten:

*ursprünglicher Code*

```
ld    A,0CFh
out   (PIO_A_C),A    ;set PIO A to bit mode
ld    A,0F1h
out   (PIO_A_C),A    ;set io configuration: A[2:0] are outputs
```

*geänderter Code, aber mit vielen Fehlern*

```
ld    B,0CFh
in    (PIO_A_C),A    ;set PIO A to bit mode
ld    A,045h
out   (PIO_A_C),A    ;set io configuration: A[2:0] are outputs
```

# *Datei zurücksetzen #2*

Feststellung: pio.asm ist Murks !  
Ausgangslage zügig wieder herstellen !

```
$ git status
```

```
On branch master
```

```
Changes not staged for commit:
```

```
(use "git add <file>..." to update what will be committed)
```

```
(use "git checkout -- <file>..." to discard changes in working directory)
```

```
modified:   pio.asm
```

```
$ git checkout src/pio.asm
```

```
$ git status
```

```
On branch master
```

```
nothing to commit, working tree clean
```

# *Datei aus Commit wiederherstellen #1*

*ursprünglicher Code*

```
ld    A,0CFh
out   (PIO_A_C),A    ;set PIO A to bit mode
ld    A,0F1h
out   (PIO_A_C),A    ;set io configuration: A[2:0] are outputs
```

*geänderter Code*

```
ld    A,0CFh
out   (PIO_B_C),A    ;set PIO B to bit mode
ld    A,0F1h
out   (PIO_B_C),A    ;set io configuration: B[2:0] are outputs
```

```
$ git add src/pio.asm
```

```
$ git commit -m "port B verwenden"
```

# *Datei aus Commit wiederherstellen #2*

```
$ git log
```

```
commit 8825874094ca8c72eba8d532ae80bcc740799594 (HEAD -> master)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Wed Mar 14 21:31:15 2018 +0100
```

```
port B verwenden
```

```
·  
·  
·
```

```
commit 35da10d4228e60b5592836c97ddb853a81b2928a
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:25:13 2018 +0100
```

```
pio configuration
```

```
·  
·  
·
```

# *Datei aus Commit wiederherstellen #3*



```
$ git checkout 35da1 src/pio.asm
```

```
$ git status
```

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

**modified: src/pio.asm**

```
ld    A,0CFh
out   (PIO_A_C),A ;set PIO A to bit mode
ld    A,0F1h
out   (PIO_A_C),A ;set io configuration: A[2:0] are outputs
```

```
$ git commit -m "zurück nach Port B"
```

```
$ git status
```

# *Datei aus Commit wiederherstellen #4*

**\$ git log**

commit 0304235af20ea17f6d42078de9fbba8e6e1d54a5 (HEAD -> master)

Author: Mario Blunk <mario.blunk@blunk-electronic.de>

Date: Wed Mar 14 21:44:24 2018 +0100

zurück nach Port A

commit 8825874094ca8c72eba8d532ae80bcc740799594

Author: Mario Blunk <mario.blunk@blunk-electronic.de>

Date: Wed Mar 14 21:31:15 2018 +0100

port B verwenden

•  
•  
•



# Commit verwerfen #1



```
$ git reset --hard HEAD^
```

```
HEAD is now at 8825874 port B verwenden
```

```
$ git log
```

```
commit 8825874094ca8c72eba8d532ae80bcc740799594 (HEAD -> master)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Wed Mar 14 21:31:15 2018 +0100
```

```
port B verwenden
```

```
commit ff48f9df119b382e49a380a46fcf36b1831d2828
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Tue Mar 13 22:04:05 2018 +0100
```

```
ignore filter wieder hergestellt
```

```
·  
·
```

# *einfache Marke setzen #1*

```
$ git log
```

```
commit 5cdcdeb444c4a3dcc9d35f0ac2800894bd963373 (HEAD -> master)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Tue Mar 13 21:37:31 2018 +0100
```

```
Ignorierfilter eingebaut
```

```
commit eb376971479bbe207df68e26f95a2069213e0196
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:38:49 2018 +0100
```

```
den Rest
```

```
commit 753eb0fc82a7926902efc0a321168b154e438e60
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:34:24 2018 +0100
```

```
alle Verilog-Dateien
```

```
·  
·
```

# *einfache Marke setzen #2*

engl. lightweight tag

```
$ git tag Konfiguration_fertig
```

```
$ git log
```

```
commit 5cdcdeb....800894bd963373 (HEAD -> master, tag: Konfiguration_fertig)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

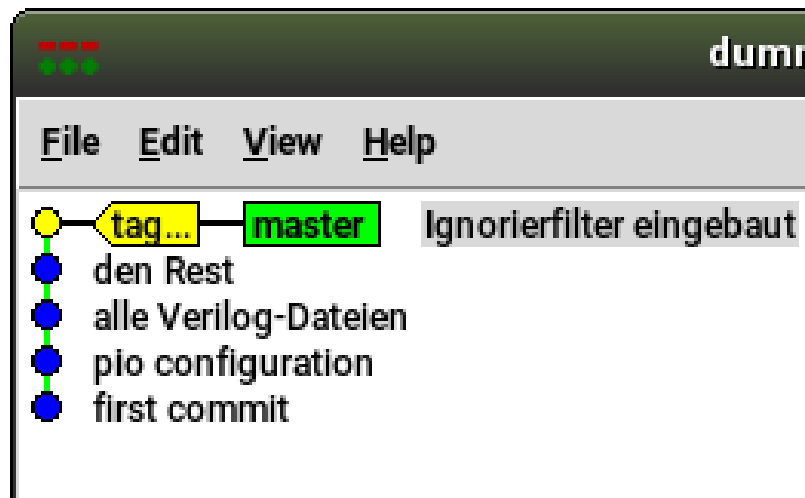
```
Date: Tue Mar 13 21:37:31 2018 +0100
```

Ignorierfilter eingebaut

.

.

```
$ gitk
```



# *einfache Marke setzen #3*

```
$ git tag Verilog 753eb0fc82
```

```
$ git log
```

```
commit 5cdcdeb....800894bd963373 (HEAD -> master, tag: Konfiguration_fertig)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Tue Mar 13 21:37:31 2018 +0100
```

Ignorierfilter eingebaut

```
commit eb376971479bbe207df68e26f95a2069213e0196
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:38:49 2018 +0100
```

den Rest

```
commit 753eb0fc82a7926902efc0a321168b154e438e60 (tag: Verilog)
```

```
Author: Mario Blunk <mario.blunk@blunk-electronic.de>
```

```
Date: Mon Mar 12 21:34:24 2018 +0100
```

alle Verilog-Dateien

- 
- 
-

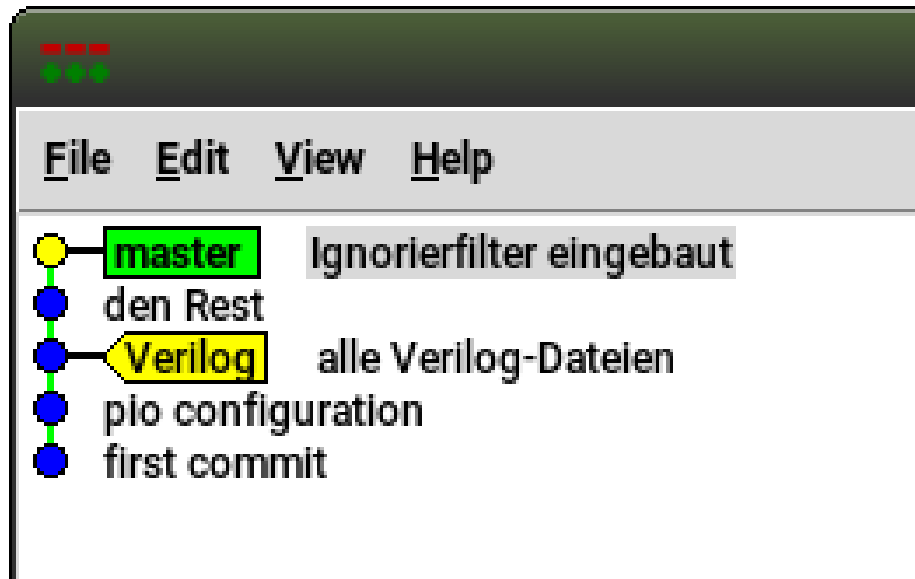
# *einfache Marke löschen #3*

```
$ git tag -d Konfiguration_fertig
```

```
Deleted tag 'Konfiguration_fertig' (was 5cdcdeb)
```

```
$ git log
```

```
$ gitk
```



# Release-Marke setzen #1

```
$ git tag -a V1.0
```

```
Das ist Version 1.0 unseres Dummy-Projektes
```

```
#  
# Write a message for tag:  
# V1.0  
# Lines starting with '#' will be ignored,  
# and can be used to set options for the future
```

```
$ git log
```

```
$ gitk
```

The screenshot shows the gitk GUI for a project named 'dummy\_project'. The main window displays a commit history with a yellow arrow pointing to a commit labeled 'V1.0' on the 'master' branch. The commit message is 'Ignorierfilter eingebaut'. The commit details pane on the right shows the commit message and the committer's name, 'Mario Blunk <mario.blunk@blunk-electronic.de>'. The bottom pane shows the commit details for the selected tag 'V1.0', including the SHA1 ID '5cdcddeb444c4a3dcc9d35f0ac2800894bd963373', the commit type 'commit', and the committer information. The commit message 'Das ist Version 1.0 unseres Dummy-Projektes' is displayed at the bottom of the commit details pane.

# Release-Marke setzen #2


```
$ git tag -a V0.8 753eb0fc82
```

```
Das ist V0.8 des Dummy-Projektes
```

```
#  
# Write a message for tag:  
#   V0.8  
# Lines starting with '#' will be ignore
```

```
$ git log
```

```
$ gitk
```



The screenshot shows the gitk GUI for the 'dummy' repository. The top menu bar includes 'File', 'Edit', 'View', and 'Help'. The main area displays a commit graph with a yellow dot for 'master' (Ignorierfilter eingebaut), blue dots for 'den Rest', 'pio configuration', and 'first commit', and a yellow box labeled '2 tags...' pointing to the current commit. The commit details section shows the SHA1 ID: 753eb0fc82a7926902efc0a321168b154e438e6. Below this is a search bar with 'Find' and 'Search' buttons, and a dropdown menu set to 'commit containing:'. At the bottom, there are buttons for 'Diff', 'Old version', and 'New version', along with a 'Lines of context: 3' field. The commit message is displayed as follows:

```
Tag: V0.8  
object 753eb0fc82a7926902efc0a321168b154e438e60  
type commit  
tag V0.8  
tagger Mario Blunk <mario.blunk@blunk-electronic.de>  
Das ist V0.8 des Dummy-Projektes
```

# Verzweigung starten #1

```
$ git checkout 753eb0fc82
```

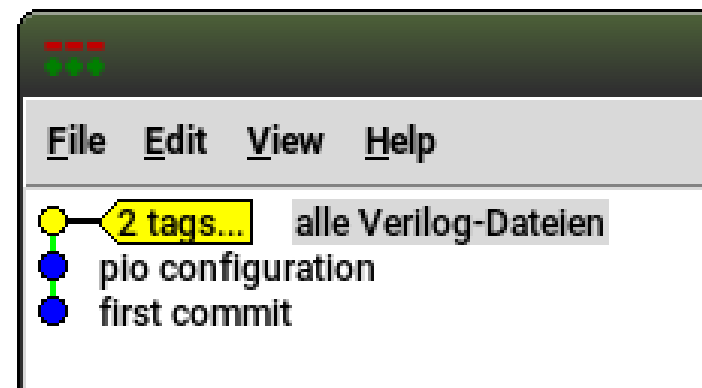
```
Note: checking out '753eb0fc82'.
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using `-b` with the checkout command again. Example:

```
git checkout -b <new-branch-name>
```

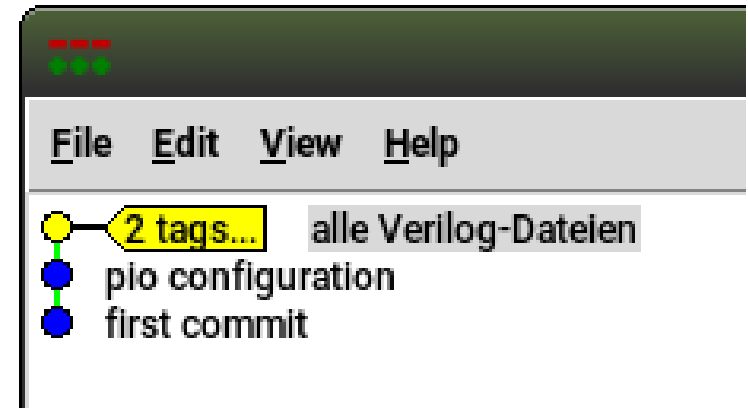
HEAD is now at 753eb0f alle Verilog-Dateien





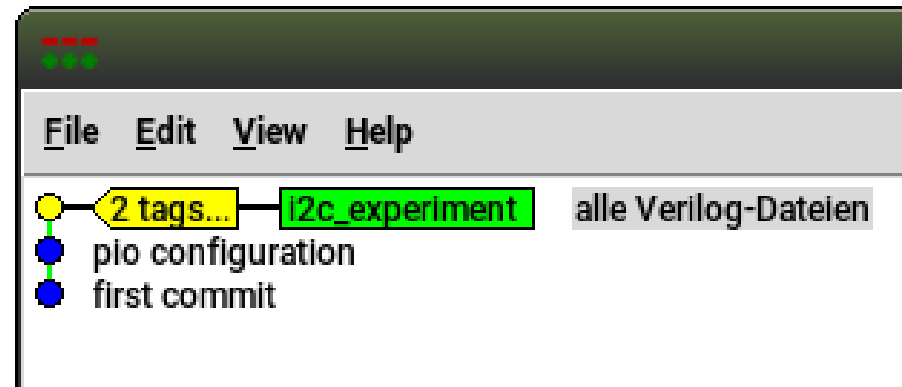
# Verzweigung starten #2

```
$ git branch
* (HEAD detached at 753eb0f)
master
```



```
$ git branch i2c_experiment
```

```
$ git branch
* (HEAD detached at 753eb0f)
i2c_experiment
master
```



# *zwischen Zweigen wechseln*

```
$ git checkout i2c_experiment
```

```
Switched to branch 'i2c_experiment'
```

```
$ git branch
```

```
* i2c_experiment
```

```
master
```

*im Zweig i2c\_experiment weiterarbeiten, add, commit, tag ...*

```
$ git checkout master
```

```
Switched to branch 'master'
```

```
$ git branch
```

```
* master
```

```
i2c_experiment
```

*im Zweig master weiterarbeiten, add, commit, add, commit, tag ...*

# *in Zweig weiterarbeiten*

```
$ git status
```

```
git status
```

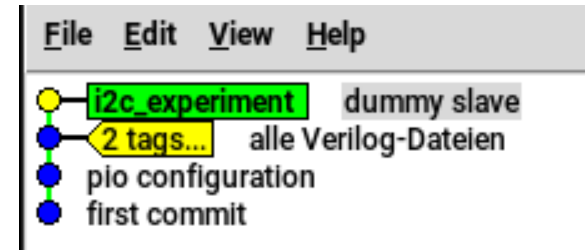
```
On branch i2c_experiment
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be committed)
```

**i2c\_slave.v**

```
$ git add i2c_slave.v
```

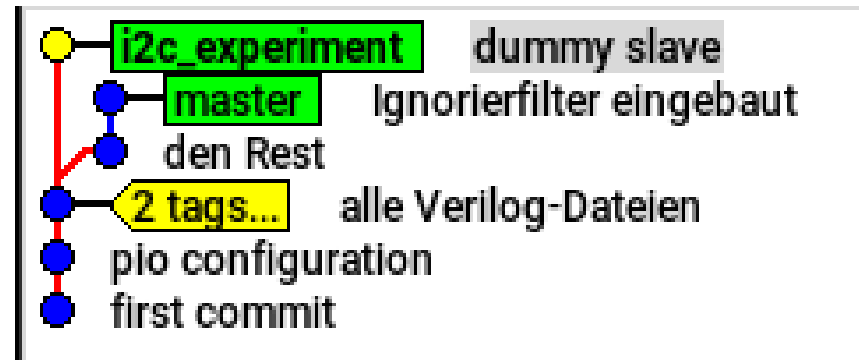


```
$ ls src/
```

```
debouncer.v i2c_master.v i2c_slave.v pio.asm
```

# Zweige vereinen (merge) #1

```
$ gitk -all
```



```
$ git checkout master
```

```
Switched to branch 'master'
```

```
$ ls src/
```

```
debouncer.v  hello.adb  hello_world.c  i2c_master.v  led_driver.sch  pio.asm
```

```
$ git merge i2c_experiment
```

# Zweige vereinen (merge) #2

```
$ git merge i2c_experiment
```

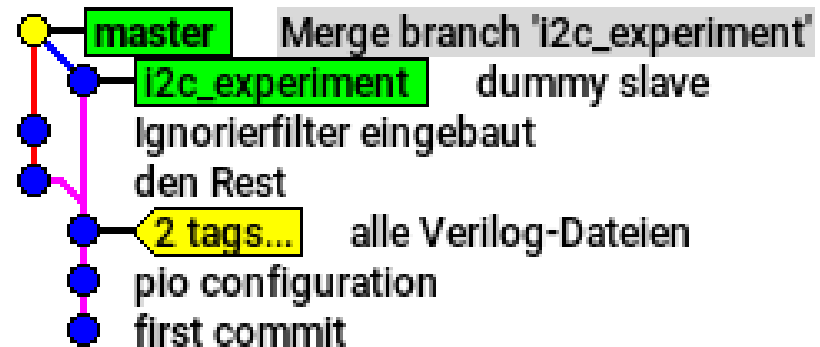
```
$ git status
```

```
On branch master
```

```
$ ls src/
```

```
debouncer.v  hello_world.c  i2c_slave.v  pio.asm  
hello.adb    i2c_master.v  led_driver.sch
```

```
$ gitk -all
```



# *Verzweigung starten #3*

```
$ git branch aufraeumen
```

```
$ git checkout aufraeumen
```

```
On branch 'aufraeumen'
```

```
$ git rm src/hello_world.c
```

```
$ ls src/
```

```
debouncer.v  hello.adb  i2c_master.v  i2c_slave.v  led_driver.sch  pio.asm
```

```
$ git commit -m "aufgeraeumt"
```

# Zweige vereinen (merge) #3

```
$ git checkout master
```

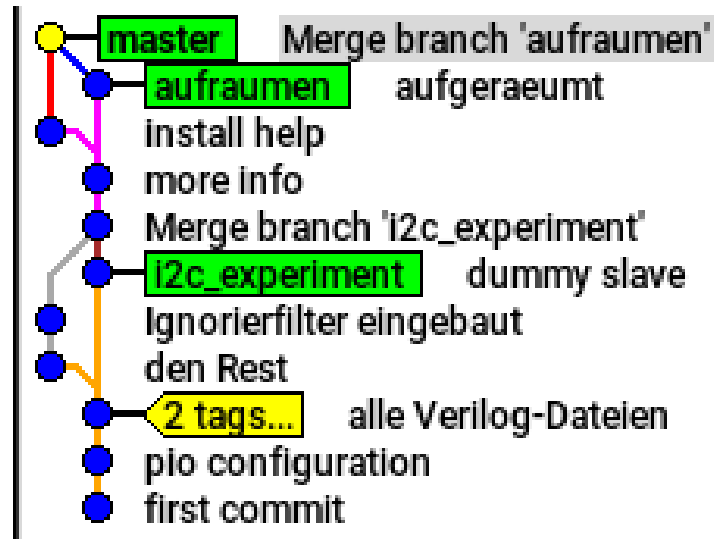
```
Switched to branch 'master'
```

```
$ git merge aufräumen
```

```
$ ls src/
```

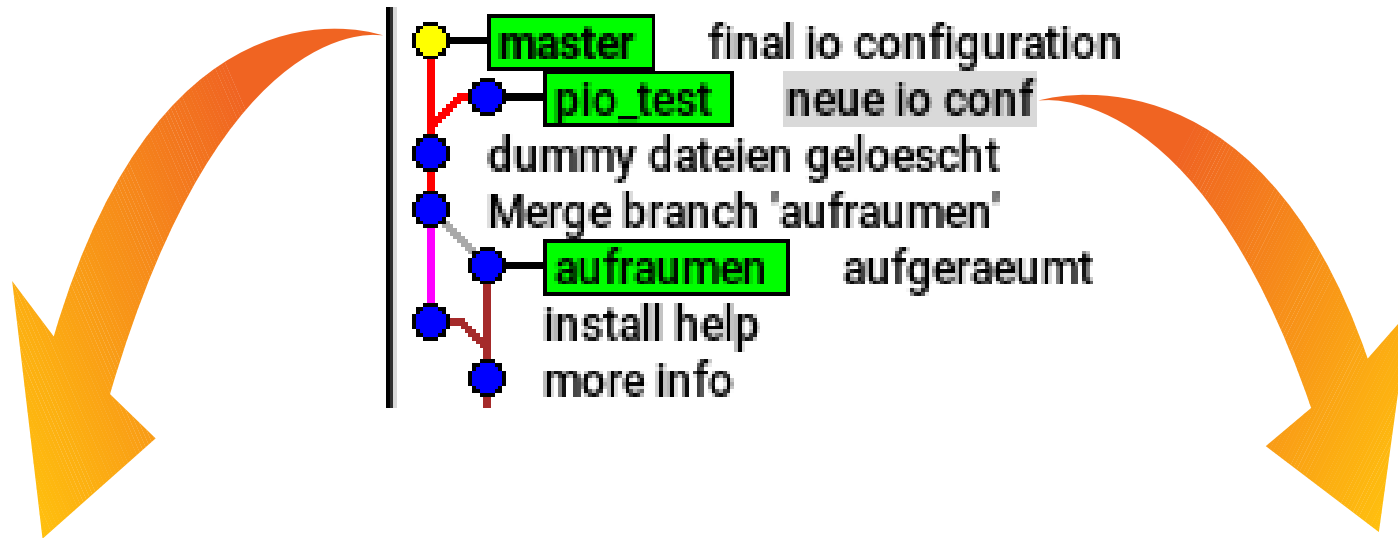
```
debouncer.v  hello.adb  i2c_master.v  led_driver.sch  pio.asm
```

```
$ gitk -all
```



# Konflikte (merge) #1

```
$ gitk -all
```



```
ld    A,0CFh
out   (PIO_A_C),A
ld    A,0F5h
out   (PIO_A_C),A
```

```
ld    A,0CFh
out   (PIO_A_C),A
ld    A,0F2h
out   (PIO_A_C),A
```



# ***Konflikte (merge) #2***

```
$ git checkout master
```

```
$ git merge pio_test
```

```
Auto-merging src/pio.asm
```

```
CONFLICT (content): Merge conflict in src/pio.asm
```

```
Automatic merge failed; fix conflicts and then commit the result.
```

```
$ git status
```

```
On branch master
```

```
You have unmerged paths.
```

```
  (fix conflicts and run "git commit")
```

```
  (use "git merge --abort" to abort the merge)
```

```
Unmerged paths:
```

```
  (use "git add <file>..." to mark resolution)
```

```
both modified:   src/pio.asm
```

# Konflikte (merge) #3

*mit Texteditor Konflikt beheben*

```
; Demo Assembly Code
```

```
; PIO initialization
```

```
    ld  A,0CFh
```

```
    out (PIO_A_C),A  ;set PIO A to bit mode
```

```
<<<<<<< HEAD
```

```
    ld  A,0F5h
```

```
=====
```

```
    ld  A,0F2h
```

```
>>>>>>> pio_test
```

```
    out (PIO_A_C),A  ;set io configuration: A[2:0] are
```

```
outputs
```

```
; end of file
```

# Konflikte (merge) #4

```
$ git add src/pio.asm
```

```
$ git status
```

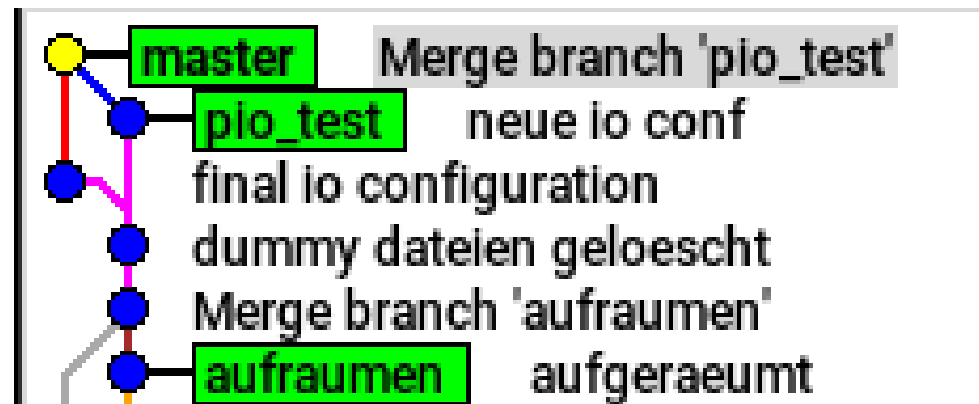
On branch master

All conflicts fixed but you are still merging.

(use "git commit" to conclude merge)

```
$ git commit
```

```
$ gitk --all
```



# *Links*

[http://www.blunk-electronic.de/pdf/git\\_einfuehrung.pdf](http://www.blunk-electronic.de/pdf/git_einfuehrung.pdf)



<https://de.wikipedia.org/wiki/Git>

<https://de.wikipedia.org/wiki/GitHub>

<https://de.wikipedia.org/wiki/GitLab>

<https://de.wikipedia.org/wiki/Bitbucket>

<https://git-scm.com/doc>

<https://github.com/Blunk-electronic>

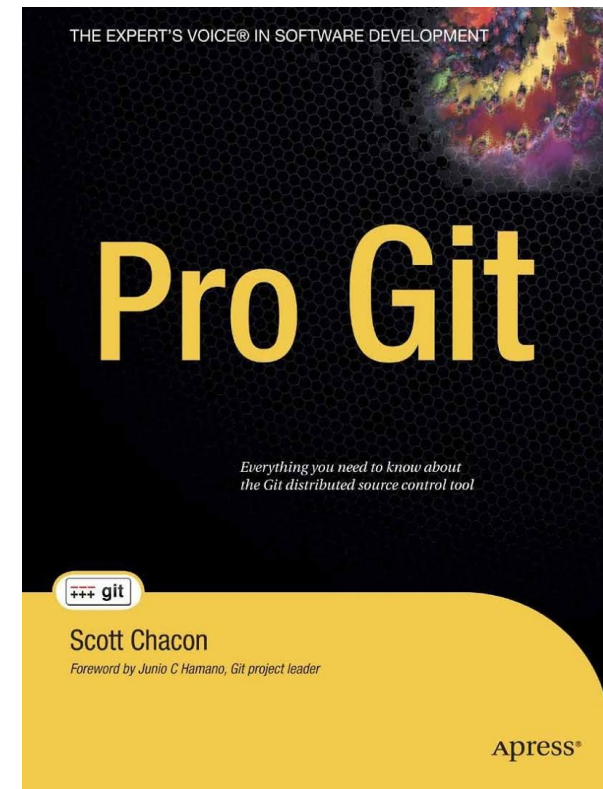
# Literatur

<http://gitbu.ch/>



<https://git-scm.com/book/de/v1>

<https://git-scm.com/book/en/v2>



<https://progit2.s3.amazonaws.com/en/2016-03-22-f3531/progit-en.1084.pdf>

***Danke für Ihre Aufmerksamkeit !***