

Die Anleitung ist vorerst fertig (Stand 12.04.2015 13:20)

Schritt 1:

Die Dateien herunterladen und in /home/username/mips\_gcc speichern:

Schritt 2:

Folgendes in die Konsole eintippen um die PATH Variable zu aktualisieren(Ubuntu):

Code: [Alles auswählen](#)

```
sudo gedit /etc/environment
```

In Debian:

Code: [Alles auswählen](#)

```
sudo gedit /etc/profile
```

Dann öffnet sich gedit und die PATH Variable die so aussehen kann:

Code: [Alles auswählen](#)

```
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games"
```

.. muss erweitert werden, dass auch der spätere Ort des neuen GCC gesucht wird:

Code: [Alles auswählen](#)

```
PATH="/home/username/mips_gcc/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games"
```

Danach mit gedit speichern, den Benutzer abmelden und wieder anmelden.

Somit ist die neue PATH Variable angelegt.

Schritt 3:

Konsole öffnen und in den Ordner /home/username/mips\_gcc dirigieren.

Folgendes eintippen und enter:

Code: [Alles auswählen](#)

```
export TARGET=mips-elf
export PREFIX=/home/username/mips_gcc
```

**Ab jetzt die Konsole nicht mehr schließen oder wechseln!**

Schritt 4:

Nun müssen zuerst die Binutils compiliert werden, dazu in den Ordner

/home/username/mips\_gcc dirigieren.

Folgendes in die Konsole eintippen:

Code: [Alles auswählen](#)

```
tar xzfv binutils-2.25.tar.gz
mkdir build-binutils
cd build-binutils
../binutils-2.25/configure --target=$TARGET --prefix=$PREFIX
make all
make install
```

Schritt 5:

Jetzt wird der MIPS GCC compiliert mit dem x86 GCC, dazu in den Ordner

/home/username/mips\_gcc dirigieren.

Folgendes in die Konsole eintippen:

Code: [Alles auswählen](#)

```
tar xjfv gcc-4.8.0.tar.bz2
mkdir build-gcc
cd build-gcc
../gcc-4.8.0/configure --target=$TARGET --prefix=$PREFIX --without-headers --with-newlib --enable-languages=c,c++
make all-gcc
make install-gcc
```

Dabei schonmal Kaffee machen, das dauert jetzt.

So jetzt haben wir einen MIPS GCC mit dem wir schon compilieren können, er ist nur total langweilig!

Denn es fehlen die Librarys für Stringverarbeitung und andere lustige Sachen.

Schritt 6:

Nun müssen wir mit dem eben compilierten MIPS GCC die newlib compilieren.

Mit der Knsole in den Ordner /home/username/mips\_gcc dirigieren.

Folgendes in die Konsole eintippen:

Code: [Alles auswählen](#)

```
tar xzfv newlib-2.1.0.tar.gz
mkdir build-newlib
cd build-newlib
../newlib-2.1.0/configure --target=$TARGET --prefix=$PREFIX
make all
make install
```

Schritt 7:

Jetzt haben wir eine fertig compilierte newlib und einen MIPS GCC der keine Ahnung hat von dieser newlib.

Wie ändern wir das? Den MIPS GCC einfach NOCHMAL compilieren.

Klingt komisch, is aber so!

Mit der Konsole in den Ordner /home/username/mips\_gcc dirigieren.

Folgendes in die Konsole eintippen:

Code: [Alles auswählen](#)

```
cd build-gcc
../gcc-4.8.0/configure --target=$TARGET --prefix=$PREFIX --with-newlib --disable-shared --disable-libssp --enable-languages=c,c++
make all
make install
```

Das dauert jetzt richtig lange, also ab zum Laden und schon mal den Kaffeevorrat aufstocken!

Schritt 8:

Mein kleinen Checker installieren, dieser überprüft ob auch alle Delayslots aus nops bestehen und ob es keine illegalen Befehle in den Code geschafft haben.

Code: [Alles auswählen](#)

```
tar xjfv slotcheck-1.1.1.tar.bz2
cd slotcheck-1.1.1
make all
```

Danach die erstellte "mips-elf-slotcheck" per Hand in den Ordner /home/username/mips\_gcc/bin kopieren

Schritt 9:

Den Code Dump zum RAM preloaden in VHDL installieren, dieser erstellt die 4 .mif Dateien fuer die VHDL Synthese und Tests.

Code: [Alles auswählen](#)

```
tar xjfv vhdldump-1.1.tar.bz2
cd vhdldump-1.1
make all
```

Danach die erstellte "mips-elf-vhdldump" per Hand in den Ordner /home/username/mips\_gcc/bin kopieren

Somit fedddich und läuft.

Änderungshistory:

19.1.2015 14:00

slotcheck-1.1.0

Zeigt nun auch die Funktion an in welcher der Fehler aufgetreten ist, nicht nur die Zeile.  
Weiterhin wird auch eine Statistik angefertigt wie häufig Befehle genutzt werden.

14.2.2015 12:30

slotcheck-1.1.1

Läuft jetzt auch auf 64Bit und erkennt den Pseudobefehl "bal"

08.03.2015 14:20

vhdldump-1.0

Erzeugt .mif Dateien für VHDL

12.04.2015 13:20

vhdldump-1.1

Bugbehebung, füllte zu früh mit Nullen auf