Start CLAR | C - Linux - Arduino - Raspberry

Inhaltsverzeichnis

Projekt c/

Homepage und Downloads

Homepage und Downloads: www.projektc.at

Projekt c/ stellt Werkzeuge und Informationen zum Entwickeln von C-Programmen mit Linux auf PC's, Raspberry und Arduino bereit.

- ▷ Alle Dateien befinden sich in Unterverzeichnissen des Projektordners c/ . Optional können weiter Ordner eingebunden werden.
- ▷ Der Name des Projektordners ist zwingend 'c'. Zur Identifizierung des Ordners dient die Lizenzdatei c/GNU_Licence.
- ▷ Die Lage des Projektordners c/ ist beliebig. Er kann auf internen oder externen Dateisystemen (z.B USB) eingerichtet werden.
- ▷ Es wird nichts im Linux-System installiert oder geändert!

Die Anleitung beschreibt das Einrichten von Projekt c/ auf Raspberry Pi und Linux PC.

Allgemeine Dokumentationen

Die allgemeinen Dokumentationen findet man unter c/1_Dokus oder im Internet:

Vorwort:	www.projektc.at/programs/c/clar_vorwort.pdf	c/1_Dokus/clar_vorwort.pdf
Projekt c/ einrichten. Die ersten Schritte:	www.projektc.at/programs/c/clar_start.pdf	c/1_Dokus/clar_start.pdf
Projekthilfe und Projektmanager:	www.projektc.at/programs/c/clar_chelp.pdf	c/1_Dokus/clar_chelp.pdf
Ein neues C Programm erstellen:	www.projektc.at/programs/c/clar_projekt.pdf	c/1_Dokus/clar_projekt.pdf
Basisobjekte ohne Terminal In/Ouput:	www.projektc.at/programs/c/clar_objekte1.pdf	c/1_Dokus/clar_objekte1.pdf
Terminalsteuerung Box-Objekte für In/Ouput:	www.projektc.at/programs/c/clar_objekte2.pdf	c/1_Dokus/clar_objekte2.pdf
Datenspeicherung:	www.projektc.at/programs/c/clar_objekte3.pdf	c/1_Dokus/clar_objekte3.pdf

Dokumentation: C-Programme und Bibliotheken

- ▷ Die ersten Infos zu den C-Programmen oder Bibliotheken findet man in der Hilfedatei '1_read.me' im jeweiligen Programmordner.
- ▷ Für aufwendige Programme gibt es Beschreibungen im Format *.odt oder *.pdf im Ordner ProgName/bin/_ProgName/
- ▷ Der Programmcodes ist in den C-Headern der Programme oder Bibliotheken dokumentiert.
- ▷ Hilfe zu den fertigen Programmen liefert immer die Programm-Startoption '-h'.

Einstiege:

Programm chelp	Verwaltung von Projekt c/. Menügesteuerter Zugriff auf alle Dokus
Projekt c/ Einstieg und Übersicht	c/1_read.me
Header/Dokus für Bibliotheksfunktionen	c/lib/1_read.me
Testprogramme für Bibliotheksfunktionen	c/libtest/1_read.me

A| Projekt c/ auf PC oder PI einrichten

1. Voraussetzungen

- ▷ Linux System
- ▷ Datenträger mit symbolischen Links!
- ▷ Der Name des Projektordners ist zwingend 'c'
  Die Lage des Ordners im Dateisystem ist beliebig!
  Im Projektordner wird die Lizenzdatei c/GNU_Licence abgelegt
- ▷ Option: Startlinks für die Programme im Homeverzeichnis ~/bin
- ▷ X-Software: geany Entwicklungsumgebung, Textverarbeitung mit Option goto Zeilennummer
  und ein Starter für Standard-Programme des Benutzers z.B. xdg-open

2. Download

Homepage:	www.projektc.at/programs/c	Projekt /c Werkstatt
Download 1:	c.tar.gz	Projektarchiv in ~/Downloads
Download 2:	shasum.txt	Prüfsumme mit Anleitung in ~/Downloads

3. Archiv entpacken

▷ Beispiel: Zielordner /home/user/test/c/
Im Terminal ausführen:

```
cd ~                                  # In das Homeverzeichnis wechseln
mkdir -pv ~/test/c                    # Ordner und Projektordner c anlegen
cd test/c                             # In den Ordner c wechseln
mv ~/Downloads/c.tar.gz ./            # Archiv in den Ordner test/c kopieren
```

```
cat ~/Downloads/shasum.txt            # Prüfsumme und Befehl auslesen
 Datum : Mo 22 Sep 2025 22:51:12 CEST
 Befehl: shasum -a 256 c.tar.gz
 42b0ef5b891dacc5d6833d0cd66ae074413a99745812b2627d22954d32d4f3e5  c.tar.gz
```

```
shasum -a 256 c.tar.gz                # Prüfsumme des Archivs ermitteln
 42b0ef5b891dacc5d6833d0cd66ae074413a99745812b2627d22954d32d4f3e5  c.tar.gz
```

```
tar -xzf c.tar.gz -C ~/test           # Archiv in Ordner Test entpacken
tree | less                           # Ordner c prüfen
```

```
.
├── 1_Dokus
│   ├── clar_chelp.odt
│   ├── clar_chelp.pdf
│   ├── clar_objekte1.odt
.   .
├── bin
│   ├── 1_read.me
│   ├── archivfoto
│   │   ├── archivfoto.c
│   │   ├── archivfoto.h
│   │   ├── bin
│   │   │   └── _archivfoto
.   .   .       ├── 1_read.me
```

4. chelp einrichten

Mit cd /home/user/test/c in den Projektordner wechseln:

▷ Befehl make

Fehlerfall: make: makefile: Keine Berechtigung
make: *** Keine Ziele. Schluss.
Zugriffsrechte überprüfen/setzen

Erfolg: Es werden die möglichen Make-Optionen angezeigt. Auszug:

```
# ═══════════════════════════════════════════════════════
#
# Make Aufrufe für Projekt /c auf PC und Pi
# 2023-04-05
# ──────────────────────────────────────────────────────
# Im Projektordner c/
#
# make            | Diese Anleitung anzeigen
# make chelp     ▶| Startpunkt: Programmhilfe 'chelp' und
#                 | alle Bibliotheken neu compilieren.
#                 | 'chelp' starten und Einstellungen prüfen
#
# make infosys   ▶| Systemeinstellungen anzeigen und verwalten
# make newprg    ▶| Programm zum Anlegen neuer Projekte
#
# make picamctl  ▶| Pi: Camera-Steuerung picamctl
# make allpi      | Pi: Programme mit GPIO aus c/pi/
#
# make clean      | Alle Programme und Bibliotheken löschen
# make libs       | Nur Bibliotheken erzeugen
# make allbin     | Bibliotheken und Programme für PC
#
# make all        | Alles: Libs, Bin, Tests, Demo
# ...
# ═══════════════════════════════════════════════════════
```

▷ Befehl make clean löscht zur Sicherheit alle Programmreste des Projekts.

▷ Befehl make chelp
- Compiliert zuerst alle Bibliotheken neu und dann das Programm chelp.
- Danach wird das Programm gestartet und führt einen ersten Startcheck durch.
- Im Startcheck können fehlenden Einstellungen vorgenommen werden.

Fehlerfall

Wenn sich chelp nicht compiliert lässt, sollte man die Zugriffsrechte prüfen und setzen:

cd /home/user/test/c	in den Projektordner c/ wechseln
id	liefert User-Name und User-Gruppe: Beispiel user:user
ls -l	zeigt Besitzer und Gruppe der Projektdateien
sudo chown -vR user:user *	Option: Besitzer und Gruppe neu einstellen. Achtung: nur im Projektordner aufrufen!

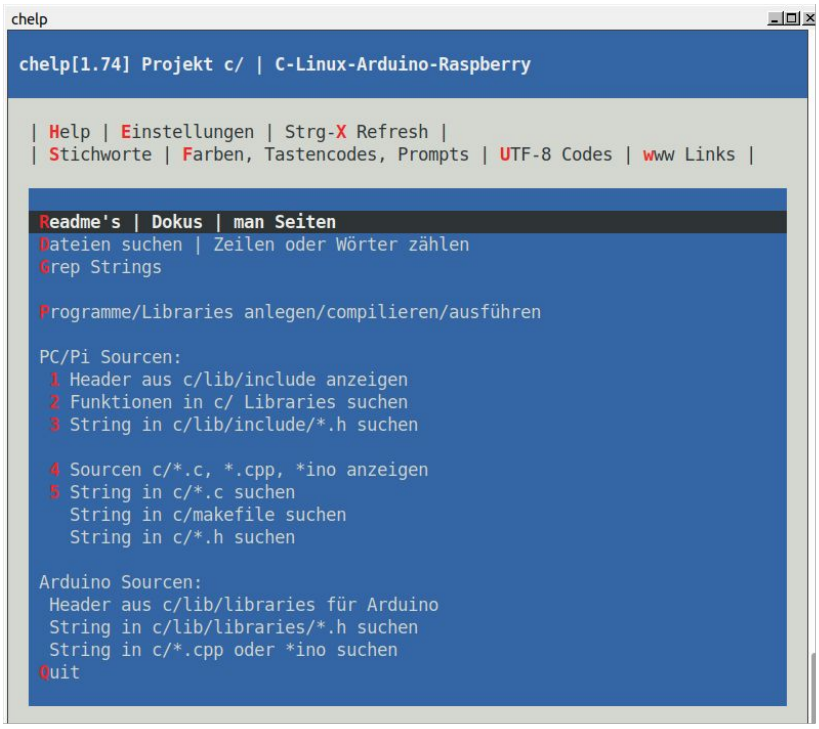
B| Projekthilfe chelp einrichten

Die Werkstatt chelp unterstützt die Entwicklung von C Programmen. Sie stellt vielfältige Informationen zum Projekt /c, zu den Bibliotheken, zum Linux und zu den eigenen Programmen zur Verfügung.

Die Programme können einzeln compiliert und im Ordner ~/bin können Startlinks erstellt werden.

Unter dem Menüpunkt Einstellungen können die Projekteinstellungen geprüft oder festgelegt werden.

Anleitung siehe Doku 'clar_chelp.pdf'



chelp kompilieren

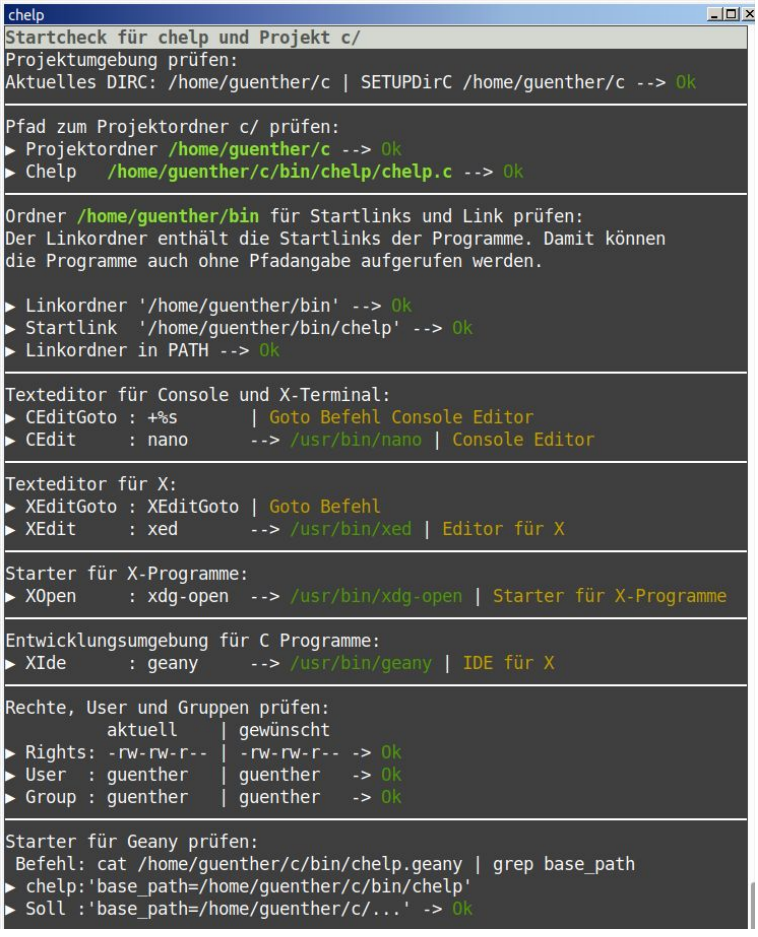
Ins Projektverzeichnis /home/user/c wechseln.

Mit Befehl make chelp werden zuerst die Bibliotheken und dann das Programm compiliert und gestartet.

Startcheck

Beim Start von chelp erfolgt automatisch ein Startcheck zu Überprüfung der Einstellungen.

Im Dialog Einstellungen kann dieser Startcheck manuell aufgerufen werden. Dort können die verschiedenen Einstellungen auch angepasst werden.



▷ Projektumgebung prüfen: Es wird der aktuelle Projektordner DIRC mit dem im Setup gespeicherten Ordner verglichen.

▷ Pfad zum Projektordner c/ prüfen: Schreibzugriff auf den Projektordner und Lesezugriff auf chelp.c prüfen.

▷ Startlinks: Der Linkordner ~/bin wird angelegt/geprüft und der Startlink für chelp wird angelegt/geprüft. Variable $PATH wird beim normalerweise beim Anmelden automatisch ergänzt. Mit dem angelegten Startlink kann das Programm nun in jedem Ordner mit chelp werden.

▷ Texteditor für Console und X-Terminal:
Die Verfügbarkeit des eingestellten Editors prüfen.

▷ Texteditor für X:
Die Verfügbarkeit des eingestellten Editors prüfen..

▷ Starter für X-Programme:
Die Verfügbarkeit des eingestellten Starters prüfen.

▷ Entwicklungsumgebung für C Programme:
Die Verfügbarkeit der eingestellten IDE prüfen.

▷ Rechte, User und Gruppen prüfen:
Einstellen unter Einstellungen > Rechte, User ...

▷ Starter für Geany prüfen:
Unter Einstellungen > Geany Startdateien können die Startdateien für Geany der aktuellen Umgebung angepasste werden.

Der Startcheck sollte fehlerfrei ablaufen!

Fehler unter Einstellungen oder in der Konfiguration beheben und den Befehl Startcheck nochmals aufrufen.

Eine genaue Anleitung findet man in der Doku chelp:

c/1_Dokus/clar_chelp.pdf
oder www.projektc.at/programs/c/clar_chelp.pdf

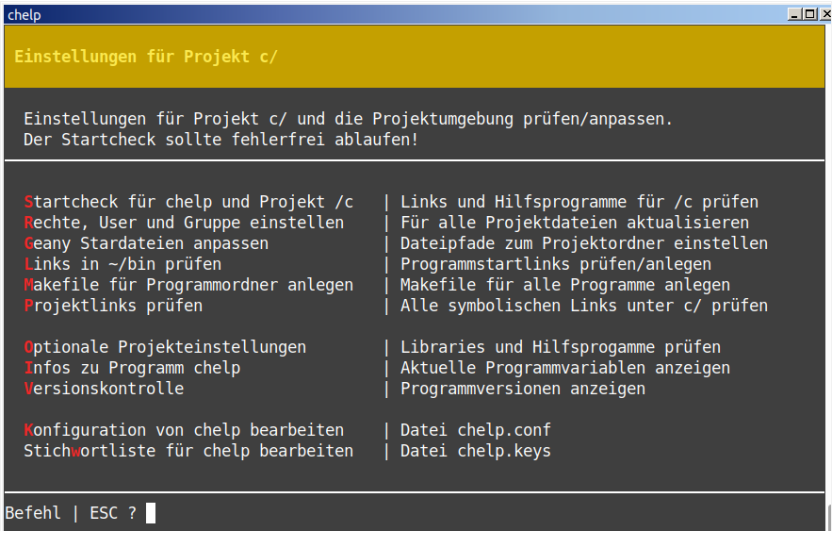
Einstellungen

Einstellungen für Projekt c/ und die Projektumgebung prüfen/anpassen.

Befehl: Einstellungen

Die genaue Anleitung findet man in Doku chelp:

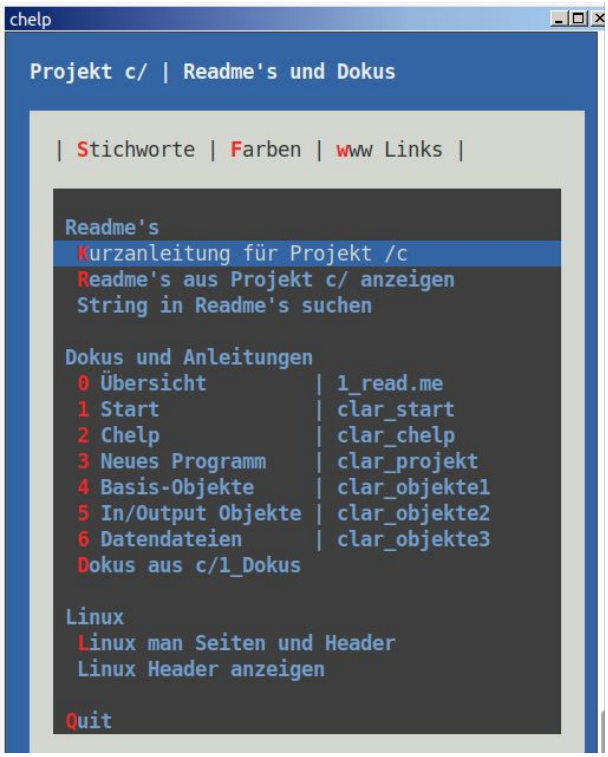
c/1_Dokus/clar_chelp.pdf
oder
www.projektc.at/programs/c/clar_chelp.pdf



Hilfe zum Projekt c/

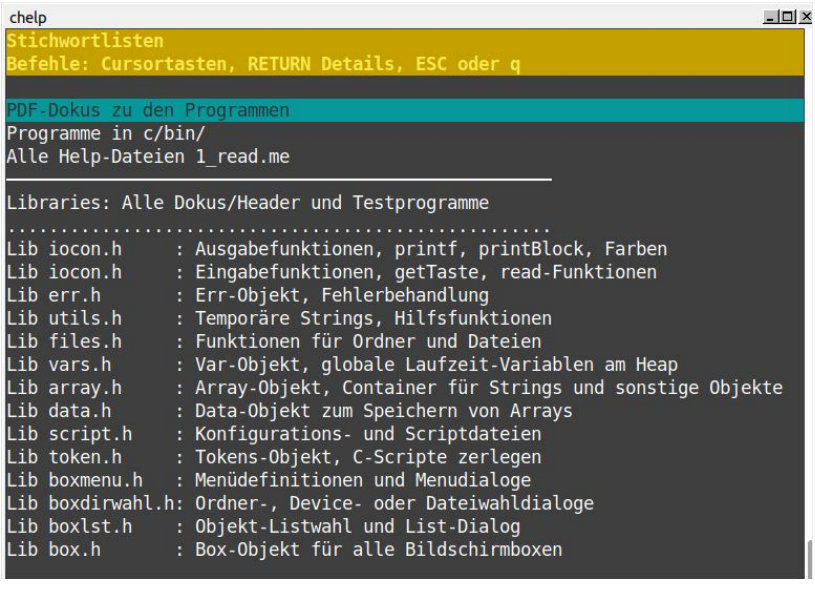
Infos und Dokumentationen können über die Befehle Readme's und Stichworte abgerufen werden.

Befehl: Readme's | Dokus | man Seiten



Befehl: Stichworte

Über selbst definierte Stichwortlisten können Informationen abgerufen und Programme gestartet werden.



Die genaue Anleitung findet man in Doku chelp:

c/1_Dokus/clar_chelp.pdf
oder www.projektc.at/programs/c/clar_chelp.pdf

C| Programm infosys

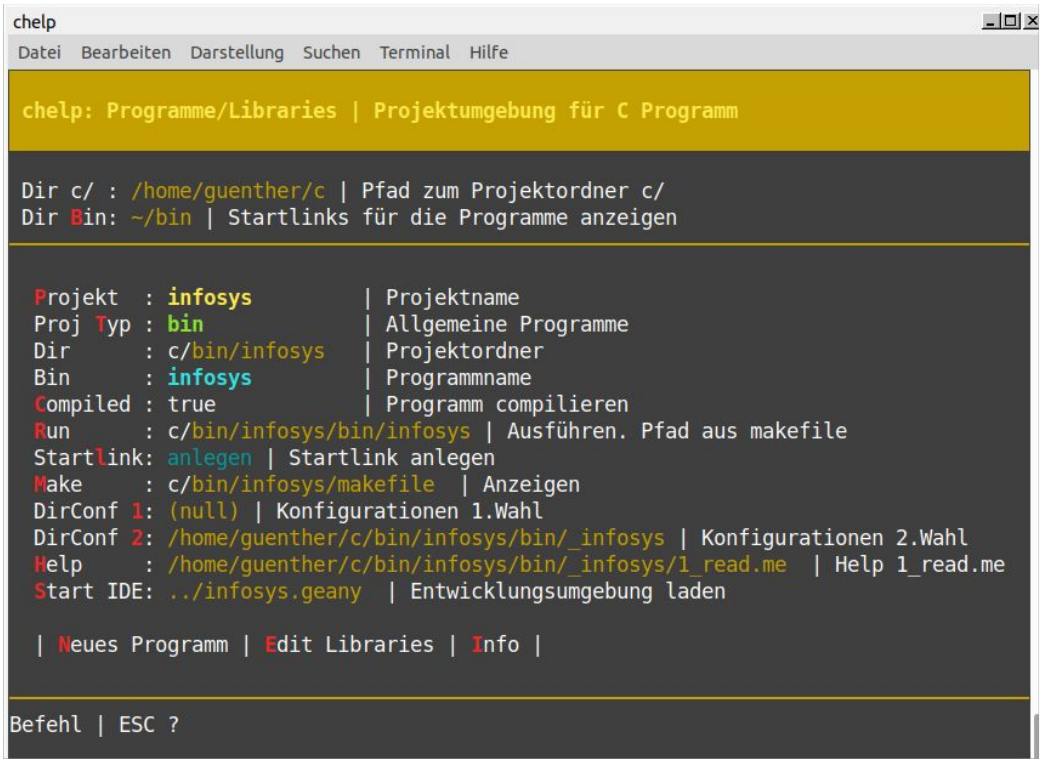
Das Programm infosys liefert umfangreiche Systeminformationen zum Rechner, zu Linux und zur Systemwartung.
Das Programm ist immer 'Work In Progress'.

Programm 'infosys' mit chelp compilieren

Programm infosys mit chelp compilieren und einrichten.

Im Menu von chelp: Programme/Libraries anlegen/compilieren/ausführen

Projektname infosys wählen: Projekt | 1 bin | infosys
Programm compilieren: Compiled
Startlink anlegen: Startlink Programm kann dann einfach mit infosys gestartet werden.

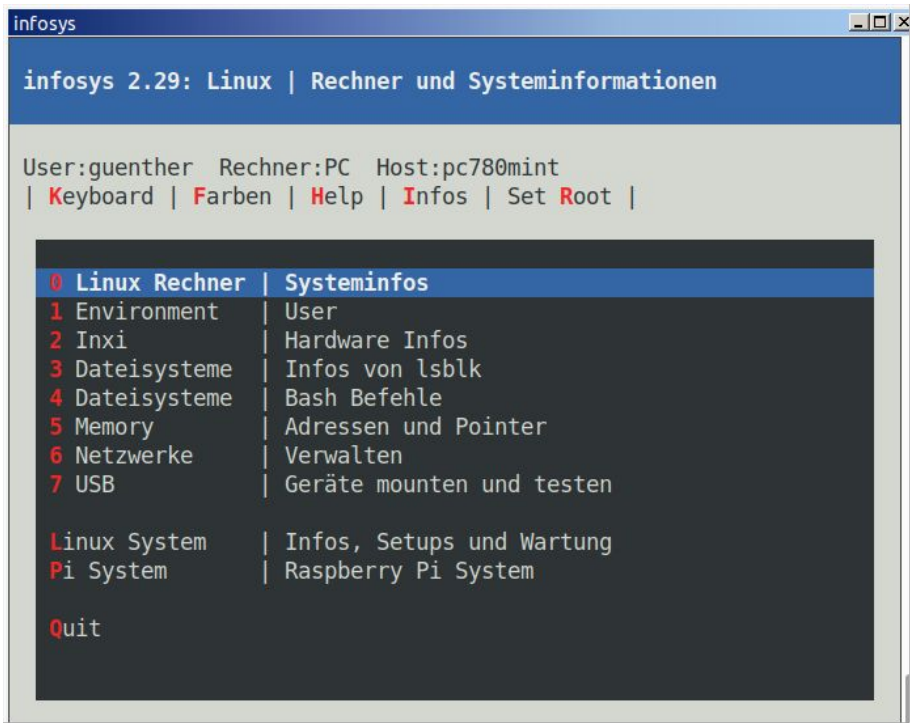


Entwicklungsumgebung: Start IDE Entwicklungsumgebung starten
Neues Programm: Neues Programm Programmgerüst für neues Programm anlegen

Anleitung für chelp siehe Doku chelp: www.projektc.at/programs/c/clar_chelp.pdf

Programm infosys

Liefert ausführliche Informationen zum Rechner, zu Linux, zur Systemwartung und Systeminfos für den C Programmierer.

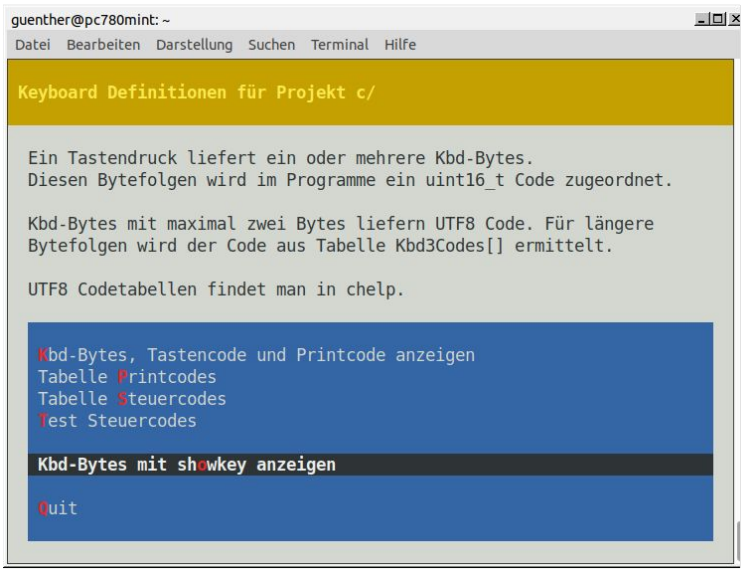


Infos zum Keyboard

Programm infosys kann auch zum Überprüfen der Tastatureingaben verwendet werden.

Funktionierende Cursortasten sind für die Eingabefunktionen unbedingt notwendig.

Programm: infosys | Keyboard



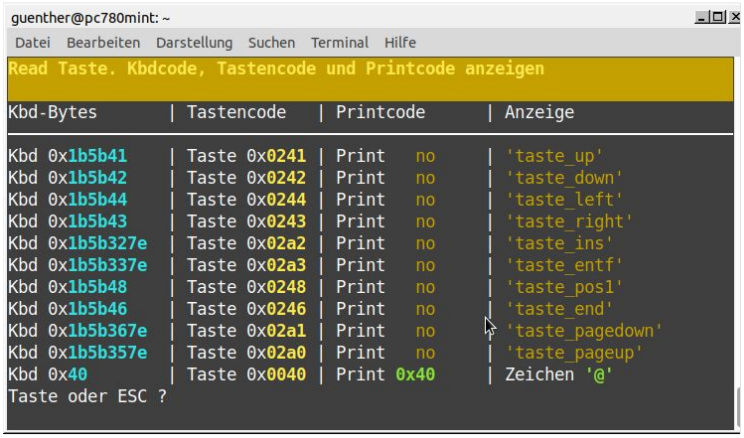
Befehl: infosys | Kbd-Bytes, Tastencode und Printcode anzeigen

Alle Cursortasten prüfen!

Die von den Tasten gesendeten Kbd-Bytes können je nach System oder Terminal (Console oder X-Terminal) variieren.

In der Bibliothek iocon.a können verschiedene Bytefolgen für die Tasten definiert werden.

Siehe c/lib/include/iocon.h

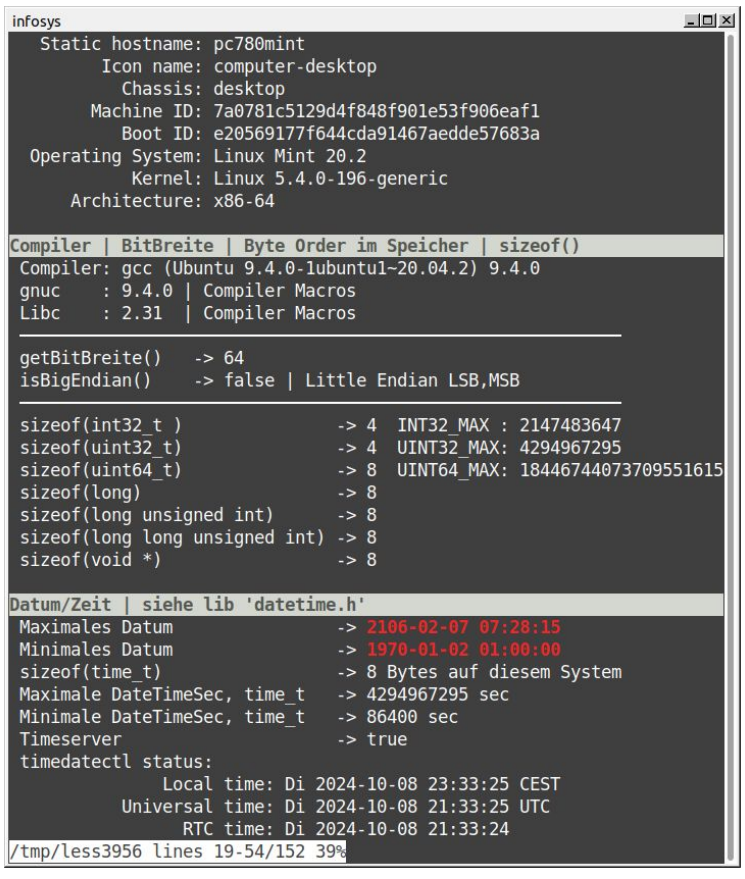


Infos zum Rechner

Auszug der Systeminfos.

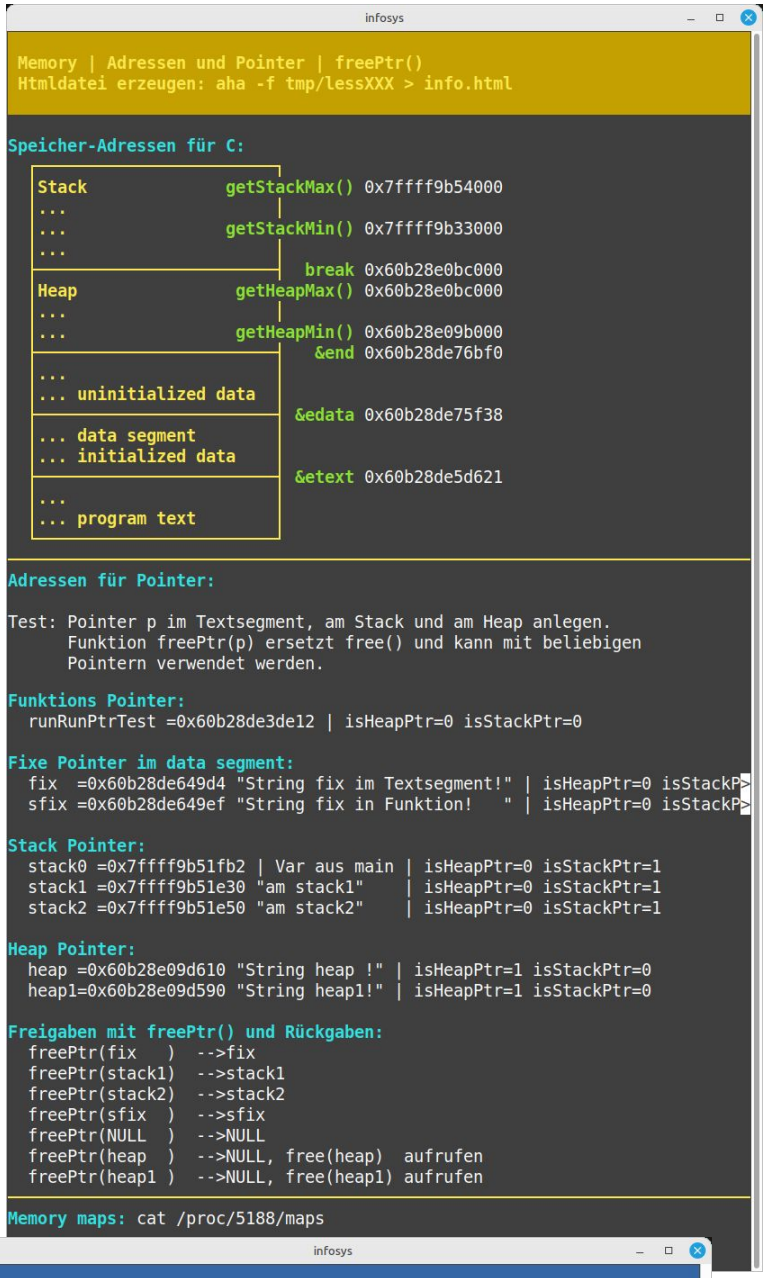
Wichtige Infos für Programmierer.

Diese Infos werden immer den aktuellen Erfordernissen entsprechend erweitert.



Infos zum Speicher

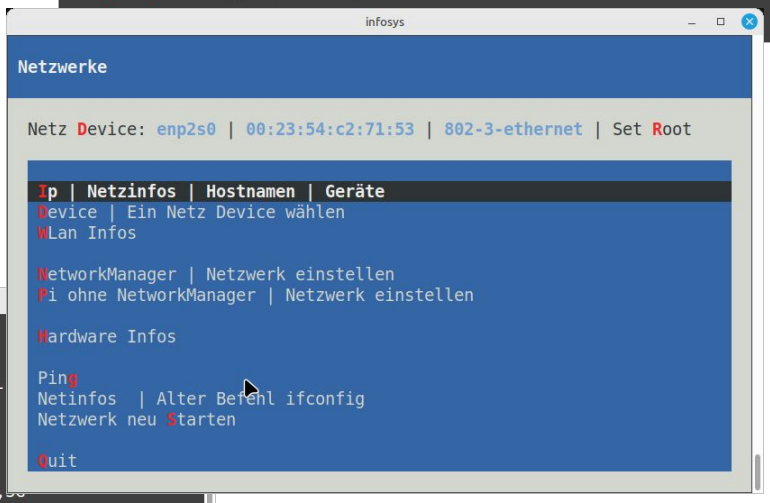
Memory-Map für die Speicher-Adressen von: C

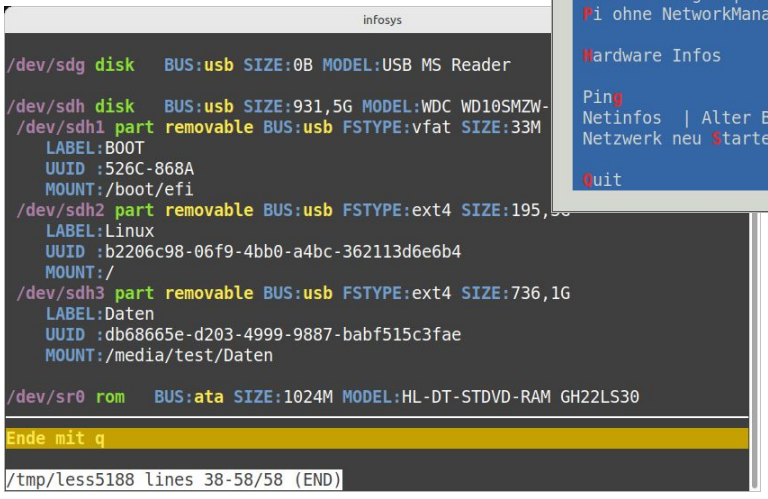


Pointer anlegen und freigeben mit freePtr():

Netzwerke:

Diese Befehle werden immer den aktuellen Erfordernissen entsprechend erweitert.





Infos zu den Dateisysteme

D| Ein eigenes C Programme erstellen

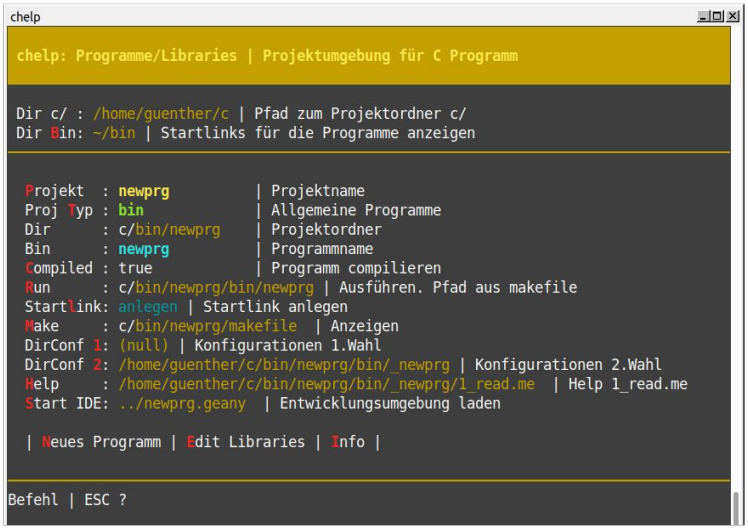
Zum Testen von Projekt c/ sollte man dann probeweise ein eigenes Programm mit newprg aus einer Vorlage erstellen.
Das Erstellen eines neuen Programms/Projekts aus Vorlagen wird in clar_projekt.pdf genau beschrieben.

Eine Kurzfassung:

Hilfs-Programm 'newprg' compilieren

chelp starten. Im Menu von chelp: Programme/Libraries anlegen/compilieren/ausführen

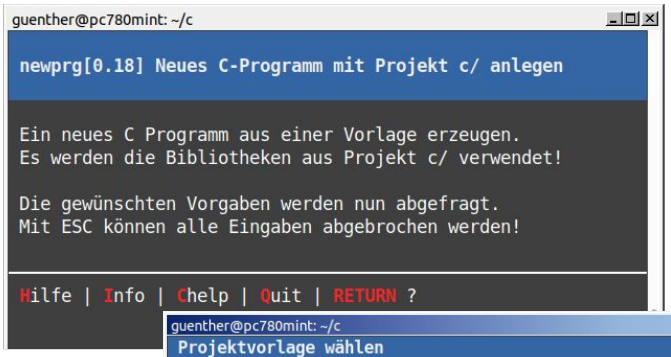
Projektname infosys wählen: Projekt | 1 bin | Projektordner: newprog



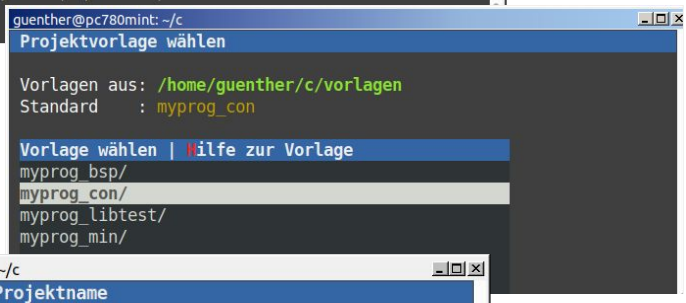
Programm compilieren: Compiled
Startlink anlegen: Startlink

Neues Programm 'newprg' erstellen

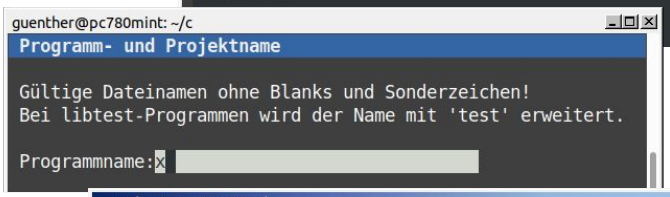
Mit Befehl Neues Programm kann newprg gestartet werden. Programm chelp wird dabei durch newprg ersetzt



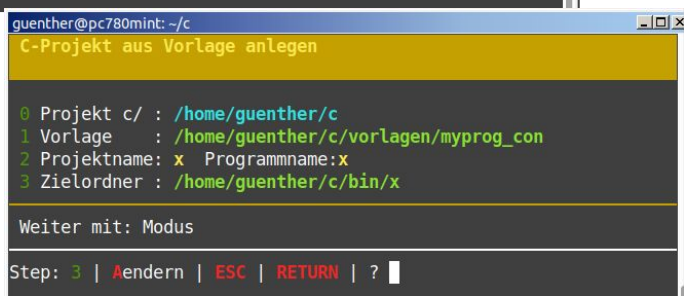
Die Vorlage myprog_con erzeugt die Grundlage für ein Terminal-Programm



Programmname festlegen



Der Modus: automatisch oder Weiter mit Taste



Danach landet man wieder in chelp. Mit Befehl Programme/Libraries anlegen/compilieren/ausführen kann das Programm x dann compiliert und gestartet werden.

Für die weitere Bearbeitung sollte die Entwicklungsumgebung geany installiert sein. Eine genaue Beschreibung zum Erstellen neuer Programmen aus Vorlagen findet man in clar_projekt.pdf .

Programme compilieren

Für das Compilieren von Programmen gibt es mehrere Möglichkeiten.

1. Programme mit make compilieren

In jedem Programmordner befindet sich neben den Sourcedateien eine passende makefile Datei. Diese Datei wird mit den folgenden make Befehlen aufgerufen:

```
make clean   // löscht alle Objektdateien
make         // erzeugt das Programm im Ordner
```

Der Befehl make compiliert immer die Dateien im Ordner oder alle Programme in den darunter liegenden Ordnern.
Beim compilieren mit make werden keine Startlinks in /home/user/bin angelegt. Die Programme müssen dann mit dem Programm-Pfad gestartet werden. Die Startlinks können mit chelp angelegt werden. Siehe weiter Unten.

Im Startverzeichnis 'c/' liefert der Aufruf make eine Liste von einigen Möglichkeiten:

Achtung: Der Aufruf von make funktioniert nur, wenn alle externen Biblitheken für ein Programm installiert sind.
Diese Biblotheken können in chelp unter Einstellungen/Optionale Projekteinstellungen nachgeschlagen werden.
Beispiel: Wenn libjpeg-dev für jpg Bilder nicht installiert ist, bricht make allbin bei Programm pshow ab.

```
# ═══════════════════════════════════════════════════════════
#
# Make Aufrufe für Projekt /c auf PC und Pi
# 2023-04-05
# ──────────────────────────────────────────────────────────────
# Im Projektordner c/
#
# make            | Diese Anleitung anzeigen
# make chelp     ▶| Startpunkt: Programmhilfe 'chelp' und
#                 | alle Bibliotheken neu compilieren.
#                 | 'chelp' starten und Einstellungen prüfen
#
# make infosys   ▶| Systemeinstellungen anzeigen und verwalten
# make newprg    ▶| Programm zum Anlegen neuer Projekte
#
# make picamctl  ▶| Pi: Camera-Steuerung picamctl
# make allpi      | Pi: Programme mit GPIO aus c/pi/
#
# make clean      | Alle Programme und Bibliotheken löschen
# make libs       | Nur Bibliotheken erzeugen
# make allbin     | Bibliotheken und Programme für PC
#
# make all        | Alles: Libs, Bin, Tests, Demo
#
# make saveit     | Dateien und Ordner auf PC mit Pi synchronisieren
# make pshow      | Bilder und Videos auf der Console anzeigen mit fbp
# make kbdctl     | Songverwaltung für Yamaha PSR-S975
# make mtrainer   | ALSA-MIDI Test und Gehörtraining
# make mid2midraw | ALSA-MIDI Gehörtraining
#
# ──────────────────────────────────────────────────────────────
# Einzelne Programme compilieren
# In allen Unterordnern von c/ gibt es makefiles:
#
# make            | Programm(e) erzeugen
# make clean      | Programm(e) löschen
# ═══════════════════════════════════════════════════════════
```

2. Programme mit chelp compilieren

Programme können auch mit chelp compiliert werden. Dabei werden alle Projekteinstellungen angezeigt und es können auch die Startlinks erstellt werden.

Siehe Beispiel im nächsten Kapitel: Progamm 'infosys' mit chelp erstellen

Die genaue Anleitung für chelp findet man in Doku clar_chelp.pdf:

c/1_Dokus/clar_chelp.pdf
oder www.projektc.at/programs/c/clar_chelp.pdf

3. Programme mit Entwicklungsumgebung geany compilieren

Das Entwickeln und Compilieren von Programmen wird am besten über eine Entwicklungsumgebung (z.B geany) vorgenommen.

Die genaue Anleitung findet man in Doku clar_projekt.pdf

c/1_Dokus/projekt .pdf
oder www.projektc.at/programs/c/clar_projekt.pdf

Fehler beim Compilieren

Versuch 1:

1. Prüfen, ob das aktuelle Verzeichnis Projektordner c/ ist.
2. Alles bereinigen: Befehl make clean.
3. Bibliotheken make libs zuerst compilieren.
4. Befehl make wiederholen.

Versuch 2: Programme mit chelp einzeln compilieren

Versuch 3: Mit chelp die Entwicklungsumgebung geany mit dem Programmcode aufrufen und compilieren.
Dabei werden die Fehler in den Source-Texten angezeigt.

Datei- und Ordnerübersicht

```
Projekt c/: Datei- und Ordnerübersicht von c/.
            Alle Programmordner haben denselben Aufbau wie 'chelp'.

  c/                    Projektordner. Verwendet relative symbolische Links
  │
  ├── 1_read.me         Diese Datei. Einstiegshilfe  (obligatorisch!)
  ├── 2_todo.txt        Hinweise zu Projekten und nicht behobene Fehler
  │
  ├── makefile          Einstellungen für Befehl 'make'  (obligatorisch!)
  │                     make         ▶ Liefert Anleitung für mögliche make Aufrufe
  │                     make clean   ▶ Löscht alle Progamme von Projekt c/
  │                     make allbin  ▶ Alle Bibliotheken und PC-Programme erzeugen
  │                     make allpi   ▶ Alle Linkbibliotheken und Programme aus c/pi/ erzeugen
  │
  ├── GNU_Licence       GNU Lizenz (obligatorisch!)
  │
  │   ____________________________________________________________________
  ├── 1_Dokus/            Dokumentationen zu Projekt c/ (obligatorisch!)
  │   │                   siehe 1_Dokus/1_read.me
  │   │                   ____________________________________________
  │   ├── clar_vorwort.odt  Vorwort für Projekt c/
  │   ├── clar_vorwort.pdf
  │   ├── clar_start.odt    Anleitung für Projekt c/
  │   ├── clar_start.pdf
  │   │
  │   ├── clar_chelp.odt    Beschreibungen für chelp
  │   ├── clar_chelp.pdf
  │   │
  │   └── ...               Weitere Anleitungen
  │
  │   ____________________________________________________________________
  ├── arduino/          Arduino Programme und PC/Pi Programme mit
  │   │                 serieller Verbindung zum Arduino
  │   │                 ____________________________________________
  │   ├── 1_read.me     Beschreibungen für Arduino/
  │   │
  │   └── ...           siehe 1_read.me
  │
  │   ____________________________________________________________________
  ├── bin/              Allgemeine Programme für Pi/PC
  │   │                 Sourcen und Binaries (obligatorisch!)
  │   │
  │   ├── 1_read.me     Beschreibung der Programme im Ordner bin/
  │   ├── makefile      Compiliert alle Programme in bin/
  │   │
  │   │   ____________________________________________________________
  │   ├── chelp/        Aufbau der Programmordner am Beispiel chelp
  │   │   │            (obligatorisch)
  │   │   │
  │   │   ├── makefile  Befehl 'make' compiliert 'chelp'
  │   │   ├── chelp.h   Hauptheader : glob. Includes, Konstanten, Deklarationen
  │   │   ├── chelp.c   Hauptprogramm: Menu(), Init(), Main() und Exit()
  │   │   ├── run.c     Hauptfunktionen
  │   │   ├── ...       Weitere Header und Sourcen für chelp
  │   │   ├── ...       Hilfsmodule und Funktionen
  │   │   │   ________________________________________________________
  │   │   └── bin/                    Programm und Konfigurationen
  │   │       ├── _chelp/             Konfigurationsordner
  │   │       │   ├── 1_read.me       Kurzbeschreibung von chelp
  │   │       │   ├── chelp.conf      Konfigurationsdatei
  │   │       │   ├── chelp.keys.bak  Sicherung
  │   │       │   │
  │   │       │   ├── chelp.keys      Stichwortliste
  │   │       │   ├── chelp.keys.bak  Sicherung
  │   │       │   └── ...
  │   │       │
  │   │       └── chelp               Binary - fertiges Programm
  │   .
  │   │   ____________________________________________________________
  │   ├── ...           Alle weitere Programme haben denselben Aufbau wie von chelp/
  │   └── ...           ...
  │
  │   ____________________________________________________________________
  ├── bindemo/          Fertige Demo Programme (optional)
  │   │                 Sourcen und Binaries
  │   │
  │   ├── 1_read.me     Beschreibung der Programme im Ordner bindemo/
  │   ├── makefile      für alle Programme unter bindemo/
  │   │
  │   ├── ...           demo00
  │   └── ...           ...
  │
  │   ____________________________________________________________________
  ├── bsp/              Einfache C Beispiele zum Testen von speziellen
  │   │                 Fragestellungen. Sourcen und Binaries. (optional)
  │   │
  │   ├── 1_read.me     Dokumentationen für die C Beispiele/Testprogramme
  │   ├── makefile      für alle Programme unter bsp
  │   │
  │   ├── ...           weitere Beispiele siehe 1_read.me
  │   └── ...
  │
  .
```

Fortsetzung auf der nächsten Seite.

Fortsetzung: Projekt c/: Datei- und Ordnerübersicht von c/.

```
.
│
├── lib/              Statische Bibliotheken von Projekt c/ für PC/Pi und
│                     Arduino. Sourcen und Binaries. (obligatorisch!)
│
│   ├── 1_read.me       Beschreibungen und Dokus für die statischen c/ Libraries
│   ├── makefile        für alle Bibliotheken unter lib
│
│   ├── libcomapi.a     Lib: ser. Kommunikation zwischen PC/Pi und Arduino
│   ├── libiocon.a      Lib: IO-Funktionen fürs Terminal
│   ├── libutils.a      Lib: Hilfsfunktionen für Strings, Shellbefehle, usw.
│   ├── libvars.a       Lib: Strukturen für globale Variablen
│
│   ├── libcomapi.geany Starter für IDE geany mit Library comapi
│   ├── libiocon.geany  Starter für IDE geany mit Library iocon
│   ├── libutils.geany  Starter für IDE geany mit Library utils
│   ├── libvars.geany   Starter für IDE geany mit Library vars
│
│   ──────────────────────────────────────────────────────────────────
│   Allgemeine statische c/ Bibliotheken für PC/Pi
│
│   ├── include/      Header *.h für alle Bibliotheken
│                     ──────────────────────────────────────
│   ├── iocon/        Lib: IO-Funktionen fürs Terminal
│   ├── utils/        Lib: Hilfsfunktionen für Strings, Shellbefehle, usw.
│   ├── vars/         Lib: Strukturen für globale Variablen
│                          Konfigurationsdateien lesen oder schreiben
│   ├── comapi/       Lib: ser. Kommunikation zwischen PC/Pi und Arduino
│   │   ├── ...
│   │   └── ...
│
│   └── libraries/    Bibliotheken für Arduino IDE
│                     ──────────────────────────────────────
│       ├── com/      Lib: Arduino/PC/Pi Header für comapi
│       ├── comapi/   Lib: ser. Kommunikation zwischen PC/Pi und Arduino
│
│       ├── taste/    Lib: Allgemeine Tastenabfrage
│       ├── kty81/    Lib: Temperatursensor kty81
│       ├── mpc9801/  Lib: I2C Temperatursensor
│       └── ...
│
│
│   ──────────────────────────────────────────────────────────────────
├── libtest/          Testprogramme und Dokumentation für die Funktionen
│                     in den Libraries. Sourcen und Binaries (optional)
│
│   ├── 1_read.me     Dokumentationen für Testprogramme
│   ├── makefile      für alle Programme unter libtest
│
│   ├── test_*/       Ordner für Doku- oder Testprogramm *
│   ├── test_*.geany  Starter für IDE geany mit Source *
│   ├── test*         Compiliertes Programm ohne '_'
│
│   └── ...
│
│
│   ──────────────────────────────────────────────────────────────────
├── pi/                 Programme für Pi mit GPIO Ansteuerungen
│                       Sourcen und Binaries (optional)
│
│   ├── 1_Dokus/        Entwicklungs-Dokus für Pi-Programme
│   ├── bin/            fertige Pi-Programme
│   │   ├── picamctl/   Camera Steuerung für Raspberry Pi
│   │   ├── piclock/    Echtzeituhr PCF8583 für Raspberry Pi
│   │   ├── ...
│   │   ├── picamctl.geany  Starter für IDE geany
│   │   ├── piclock.geany   Starter für IDE geany
│   │   └── ...
│
│   ├── bininc/         Linkbibliothek für Steuerungs-Module. Diese Module
│   │   │               werden über symbolische Links eingebunden.
│   │
│   │   ├── events      Verwaltung und Ansteuerung von GPIO-Geräten
│   │   ├── gpio        GPIO-Module für die GPIO-Hardware vom Pi
│   │   ├── bmp180      Druck und Temperatursensor bmp180. Module für
│   │   │               Schnittstelle gpioi2.c und SBus i2c-Interface
│   │   └── ...
│
│   ├── bsp/            Testbeispiele
│
│   ├── 1_read.me       Infos zu Pi-Programmen
│   └── makefile        für alle Pi-Programme
│
│
│   ──────────────────────────────────────────────────────────────────
├── vorlagen/           Programm-Vorlagen für neue Projekte.
│                       Verwendung mit 'newprg' (obligatorisch!)
│
│   ├── myprog_con      Standardvorlage mit Menu
│   ├── myprog_min      Minimalvorlage
│
│   ├── myprog_libtest  Vorlage für Testprogramme
│   ├── myprog_bsp      Vorlage für Testbeispiele
│
│   └── ...
.
```

GNU General Public License

```
/*

    Copyright 2022-2025 Günther Schardinger <schardinger@projektc.at>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <http://www.gnu.org/licenses/>.
    ____________________________________________________________________________

    Dieses Programm ist Freie Software: Sie können es unter den Bedingungen
    der GNU General Public License, wie von der Free Software Foundation,
    Version 3 der Lizenz oder (nach Ihrer Wahl) jeder neueren
    veröffentlichten Version, weiter verteilen und/oder modifizieren.

    Dieses Programm wird in der Hoffnung bereitgestellt, dass es nützlich sein wird,
    jedoch OHNE JEDE GEWÄHR,; sogar ohne die implizite
    Gewähr der MARKTFÄHIGKEIT oder EIGNUNG FÜR EINEN BESTIMMTEN ZWECK.
    Siehe die GNU General Public License für weitere Einzelheiten.

    Sie sollten eine Kopie der GNU General Public License zusammen mit diesem
    Programm erhalten haben. Wenn nicht, siehe <https://www.gnu.org/licenses/>.

*/
```