I. GRUNDLAGEN

Aufgaben der Hardware:

Ein- und Ausgabe von Daten
Verarbeiten von Daten
Speichern von Daten

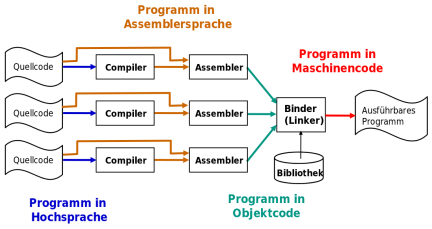
Klassische Hardwarekomponenten:

Ein- und Ausgabe
Hauptspeicher
Rechenwerk
Leitwerk

II. ANFORDERUNGEN HÖHERER PROGRAMMIERSPRACHEN

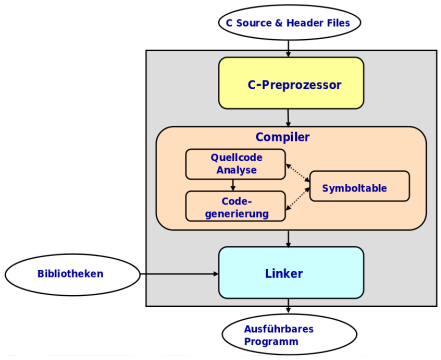
Begriffe:

Maschinensprache: Für Prozessor verständliche Anweisungsrepräsentation, z.B. 00101101001110101
Assemblersprache: Für Menschen verständliche Maaschinensprache, z.B. `add $s2, $s1, $s0`
Assembler: Übersetzt Assemblersprache eindeutig in Maschinensprache
Objektcode: Maschinenprogramm mit ungelösten externen Referenzen
Binder/Linker: Löst ungelöste Referenzen auf, verbindet alles zu einem ausführbaren Maschinenprogramm



Programmiersprache C:

Zwischenstellung zwischen Assembler und Hochsprache
hohe Portabilität trotz guter Architekturanpassung
einfache Programmierung
Datentypen: `char, int, float, double`
Kontrollstrukturen: Entscheidungen, Schleifen, Blöcke, Unterprogramme
Zeiger als Parameter möglich



C - Datentypen:

`char`: Ein Zeichen, meist 1 Byte
`int`: Integerzahl, 2 oder 4 Byte
`float`: Gleitkommazahl, meist 4 Byte
`double`: Gleitkommazahl, meist 8 Byte

C - Operatoren:

`*`: Multiplikation (`x*y`)
`/`: Division (`x/y`)
`%`: Modulo (`x%y`)
`+`: Addition (`x+y`)
`-`: Subtraktion (`x-y`)
`+` und `-` auch als Prä- und Postfix, alle auch als assign (`=` anhängen)

C - Bit-Operatoren:

`~`: Bitweise NOT (`~x`)
`<<`: links schieben (`x<<y`)
`>>`: rechts schieben (`x>>y`)
`&`: bitweise AND (`x&y`)
`^`: bitweise XOR (`x^y`)
`|`: bitweise OR (`xy|`)
alle auch als Assign (`=` anhängen)

C - Vergleichsoperatoren:

`>,<`: größer, kleiner als (`x>y, x<y`)
`>=,<=`: größergleich, kleinergleich als (`x>=y, x<=y`)
`==,!=`: gleich, ungleich (`x==y, x!=y`)

C - Spezialoperatoren:

Auswahloperator: `z = (a < b) ? a : b` (`z=a`, falls `a<b`, sonst `z=b`)