

Assemblerprogrammierung Cheat Sheet

Arbeit mit Visual Studio

Projekt und Datei anlegen:

1. Datei | Neu | Projekt

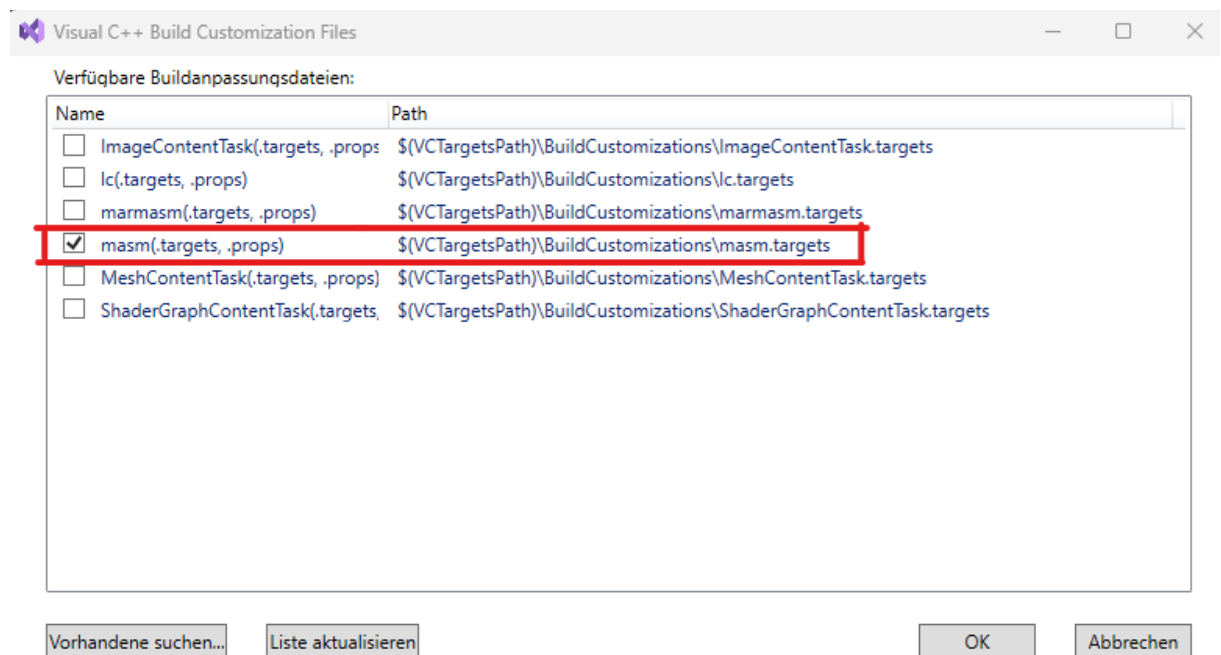
- Unter "Installierte Vorlagen" - "Andere Projekttypen" | Visual Studio - Projektmappen - Leere Projektmappe und Namen eintragen.
- Evtl. noch bei "Ort" den Ordner für den Workspace angeben (z.B. Home-Laufwerk bei Desktops des Fachbereiches, bei eigenen Rechnern ist das egal)
- mit "OK" bestätigen.

2. Datei | Hinzufügen | Neues Projekt | Installierte Vorlagen | VC++ | Allgemein | **Leeres Projekt**

- Unten den Namen eintragen.
- Mit "OK" bestätigen.

3. Rechtsklick auf das Projekt (im Projektmappen-Explorer) und „**Buildabhängigkeiten**“ und „**Buildanpassungen**“ wählen.

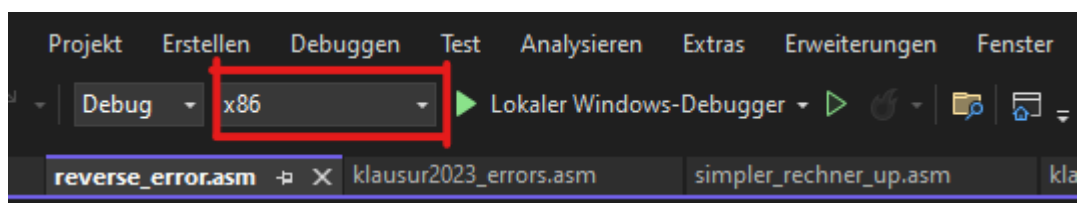
- Dort einen Haken bei "**masm**" setzen und mit "OK" bestätigen.



4. Rechtsklick auf Projekt - Hinzufügen | neues Element | Visual C++ | Hilfsprogramm | Textdatei.

- Bei "Namen" den gewünschten Dateinamen angeben, aber unbedingt mit der Erweiterung ".asm". Mit "Hinzufügen" beenden.

5. Für die Debug-Konfiguration (Ausführung Ihres Programms) „**x86**“ festlegen



Arbeit mit mehreren .asm Dateien:

1. Rechtsklick auf Projekt | Eigenschaften.

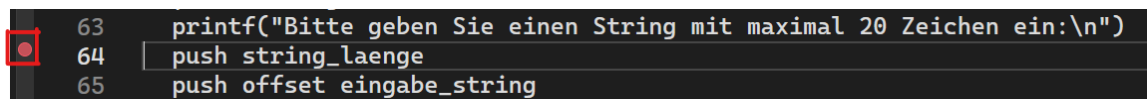
- Konfigurationseigenschaften | **Linker | Erweitert**
- auf **Einstiegspunkt** klicken. Hier den Namen des Einstiegspunktes für das auszuführende Assemblerprogramm eintragen (z.B. "main"). Mit "Übernehmen" und "OK" Fenster schließen.
- Das Assemblerprogramm muss dann natürlich ein Label "main" besitzen.

2. Statt mehreren Dateien in einem Projekt, jeweils ein neues Projekt anlegen

- Weitere Dateien (Makros, Unterprogramme, etc.) können über **include** eingebunden werden

Debuggen

- Einstellungen zum Debuggen können nur beim Debuggen vorgenommen werden!
- Dazu einen roten **Haltepunkt** setzen und das Programm ausführen



```
63  printf("Bitte geben Sie einen String mit maximal 20 Zeichen ein:\n")
64  push string_laenge
65  push offset eingabe_string
```

- im Menü Debuggen | Fenster | Register um die wichtigsten Register zu überwachen
- im Fenster Register: Rechtsklick; Haken setzen bei CPU-Register und Kennzeichen (das sind die Flags)
- Da die Flags hier teilweise anders bezeichnet werden bitte folgende Tabelle beachten:

| Flag | Wert |
|-----------------|--------|
| Overflow | OV = 1 |
| Direction | UP = 1 |
| Interrupt | EI = 1 |
| Sign | PL = 1 |
| Zero | ZR = 1 |
| Auxiliary Carry | AC = 1 |
| Parity | PE = 1 |
| Carry | CY = 1 |

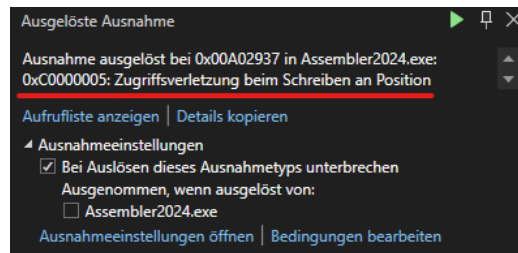
- Außerdem unter Debuggen | Fenster | **Arbeitsspeicher** | Arbeitsspeicher 1 anwählen um den Speicher während des Debuggens zu sehen
- Im Speicher können Variablen mit „&<Variablenname>“ gefunden werden
- Alternativ gibt es das Fenster: Debuggen | Fenster | **Überwachen** | Überwachen 1 – dort können Variablen sowie Registerinhalte direkt beobachtet werden
- Mit der Taste **F11** führt man einen **Einzelschritt** aus (eine Programmcodezeile wird ausgeführt) – zu beachten: Makros werden dabei übersprungen!

Printf

- Printf ist eines der zugelassenen Makros für die Prüfung und erlaubt die Ausgabe formatierter Strings, ähnlich wie die Methode mit den gleichen Namen in C.
- Mögliche Platzhalter für die Strings und deren Verwendung:
 - o printf("Ausgabe von Strings: %s", offset string_var)
 - o printf("Ausgabe von Zahlen: %d", number_var)
 - o printf("Ausgabe von Zeichen: %c", char_var)
- Auch mehrere Platzhalter sind möglich (durch weitere Kommata getrennt)
 - o printf("%d ist groesser als %d", zahl_var, eax)

Häufige Probleme

1. Beim Ausführen tritt der Fehler „**Zugriffsverletzung beim Schreiben**“ auf



- Passiert häufig bei größeren Projekten, nach Änderungen am Quellcode
- Erst sicherstellen, dass es nicht am Programmcode selbst liegt
- Zum Beheben des Problems im Menü: **Erstellen | Projektmappe neu erstellen** auswählen

2. Wo finde ich meine **.exe Datei**?

- Zunächst mit Rechtsklick auf die .asm Datei **oberhalb** des Code-Editors: **Enthaltenden Ordner öffnen**
- Liegen Projekt und Projektmappe im selben Verzeichnis?
 - o Die .exe liegt in dem **Debug**-Ordner der im **selben Verzeichnis** wie die .asm Datei liegt
- Projekt und Projektmappe nicht im selben Verzeichnis?
 - o Die .exe liegt in dem **Debug**-Ordner **der Projektmappe**, welches sich eine Ebene höher befindet

3. Makros mit Labels mehrfach aufrufen

- Wenn Sie Labels in ihrem Makro verwenden und es mehrfach aufrufen wollen
- Verwenden Sie das Schlüsselwort „**LOCAL**“ gefolgt von allen Labels in ihrem Makro direkt **nach der Makrodefinition!**

```
1 zahl_makro macro zahl_len, zahl_ergebnis
2     LOCAL set_max, set_min, starte_eingabe, eingabe_schleife, ende_eingabe, zahl_eingabe
```

4. **Eingabe Buffer bei StdIn**

- Alle Eingaben in der Konsole werden in einem Buffer gespeichert
- Beim Aufruf von StdIn wird dieser Buffer direkt von der Funktion genutzt, falls vorhanden
- Das kann dazu führen, dass ein **mehrfaches Aufrufen von StdIn** die **Nutzereingabe überspringt**, da bereits ein Buffer vorhanden ist
- Meist sind das entweder **0 (String-Ende)** oder **10 (Neue Zeile)** – da beide Werte nach dem Drücken der Eingabetaste an den Buffer übergeben werden.
- Für ein **Einlesen von Daten über StdIn** empfiehlt sich daher folgende Struktur um zu verhindern, dass diese beiden Zeichen aus dem Buffer übernommen werden:

```
38 | wiederhole_eingabe:
39 |     push text_laenge
40 |     push offset eingabe_text
41 |     call StdIn
42 |     mov al, eingabe_text[0]
43 |     cmp al, 0
44 |     je wiederhole_eingabe
45 |     cmp al, 10
46 |     je wiederhole_eingabe
```