

Das TI-C-Board

Im Labor für Technische Informatik wird das TI-C-Board für die Praktika GTP und GSP verwendet. Das Board ist mit einem Microcontroller STM32F417ZG und einem TFT-Display EA-eDIPTFT32-a bestückt. Von den IO-Pins des Bausteins sind folgende herausgeführt:

- Port E Bits 0 ... 7 als Eingabe, auf diesen Eingängen liegen auch die Schalter S0 ... S7
- Port G Bits 0 .. 15 als Ausgabe, Bits 0 .. 7 auf Buchsen
- DAC1 und DAC2 als Digital Analog Wandler
- ADC 3 Kanäle 4 ... 8 als Analogwandler
- Timer 1 Kanäle 1 ... 4 als Zeitgeber
- Interrupteingänge 2 ... 5

Diese Signale stehen auf 2 mm Buchsen für die Beschaltung zur Verfügung. Außerdem sind herausgeführt:

- USART3 als RS232 Schnittstelle
- CAN1 als CAN-Bus
- I2C 2 als Ansteuerung für Sensoren
- Signale für das Lesen und Schreiben von SD-Karten

Software

Die Entwicklung der Software für das System wird mit der Keil-Entwicklungsumgebung uVision auf einem PC durchgeführt. Das ausführbare Programm wird in den Flash-Speicher des STM32F4 auf dem TI-C-Boards geladen. Die Anbindung des TI-C-Boards an den PC erfolgt über eine USB-Schnittstelle, die auch die Stromversorgung für das Board liefert.

Die Keil Entwicklungsumgebung organisiert die Softwareentwicklung in Projekten. Um das TI-Board zu initialisieren und zu betreiben wird nicht in der Keil Entwicklungsumgebung vorhandene Software benötigt, die das TI-Labor zur Verfügung stellt. Die Software muss in das Projekt integriert werden. Das **Aufsetzen** eines Projektes wird mit Batchkommandos durchgeführt. Die Batchkommandos liegen auf dem Desktop als Icon. Es gibt folgende Icon:

- **Keil C** Entwicklung von C Programmen
- **Keil ASM** Entwicklung von ASM-Programmen
- **Keil Simulator** Entwicklung von ASM-Programmen im Simulator

Der Name des Projekts wird im DOS-Fenster abgefragt. Ist das Projekt noch nicht vorhanden, so wird es angelegt. Ist das Projekt bereits angelegt, so wird die Entwicklungsumgebung mit dem Projekt aufgerufen.

Der Name des Projektes kann aus Buchstaben [A -Z, a - z], aus Ziffern [0 - 9] und aus dem Unterstrich [_] gebildet werden. Bitte keine Umlaute verwenden. Die Kommandos dienen nur dem Aufsetzen eines Projektes und **es kann kein bestehendes Projekt in ein Anderes überführt werden**. Die Projekte werden auf dem Laufwerk Z abgelegt.

Z:\TI_Labor\Keil\ *name_des_projektes*

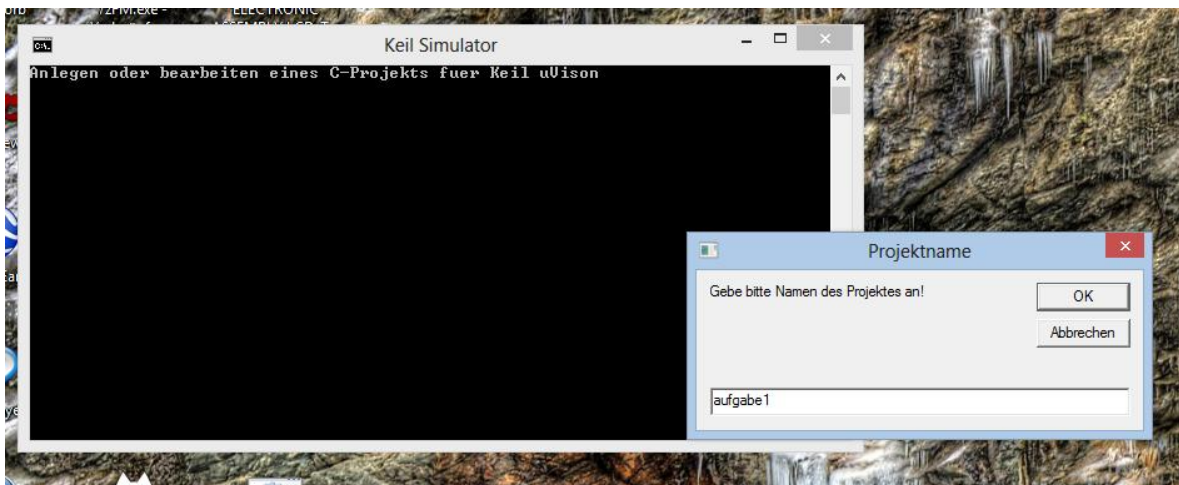
Die grundlegende Bedienung am Beispiel der Simulation

1. Anlegen eines Projekts zur Simulation des CORTEX Microcontroller :

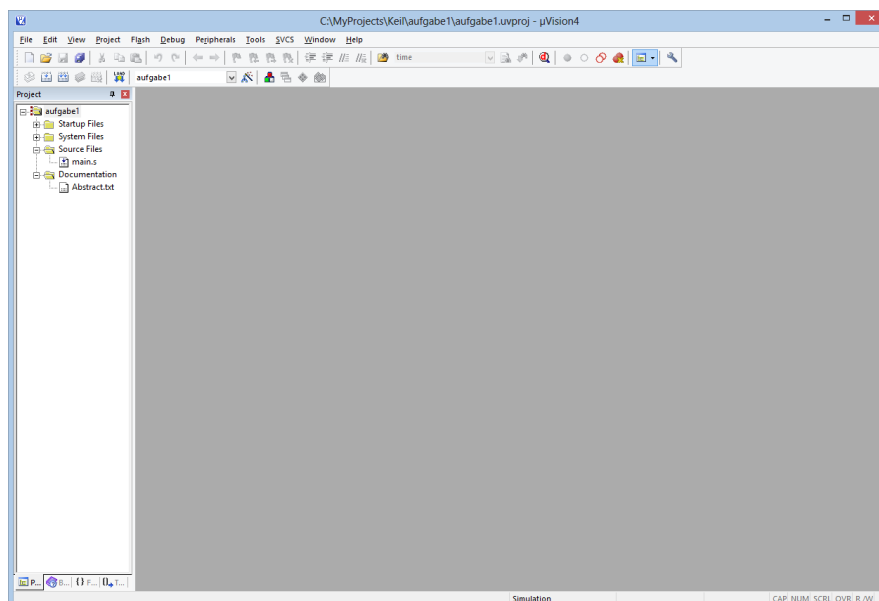
- Doppelt klicken des Icon **Keil Simulator**



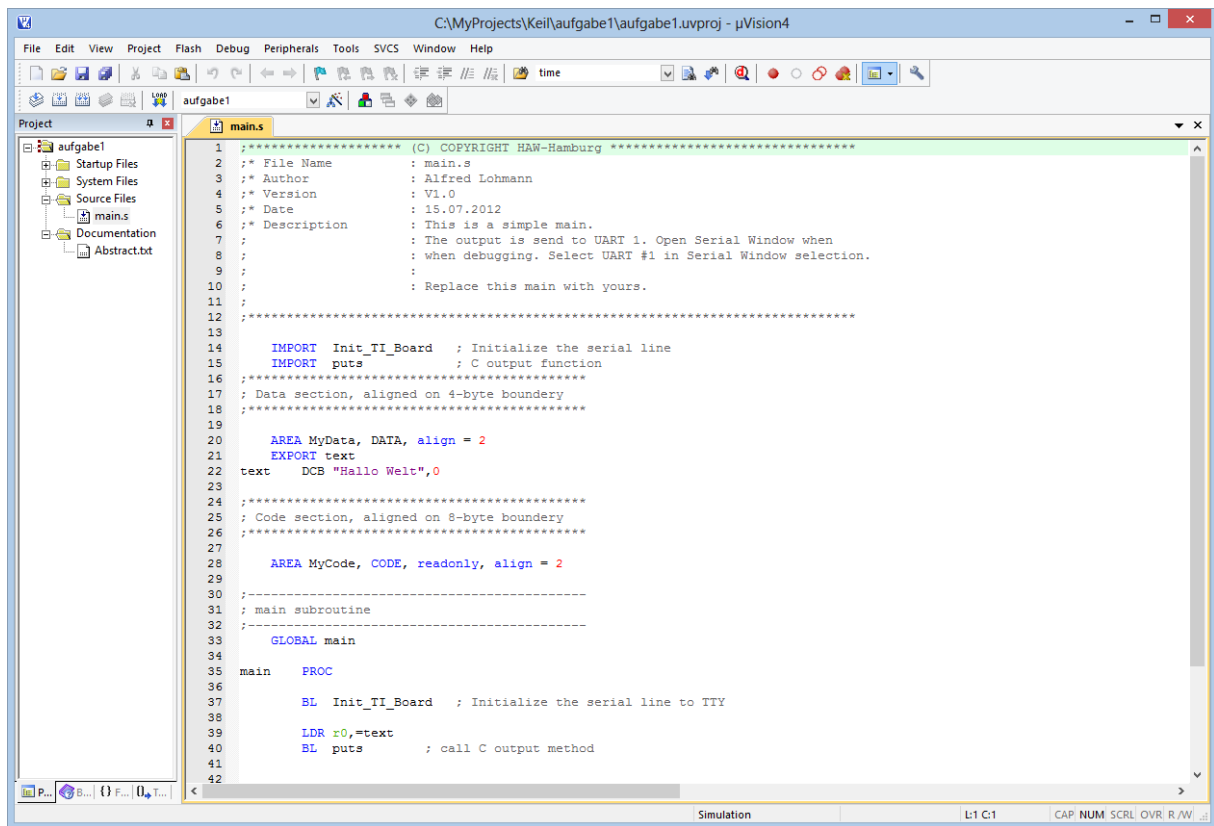
- Eingeben des Projektname im Fenster **Projektname**



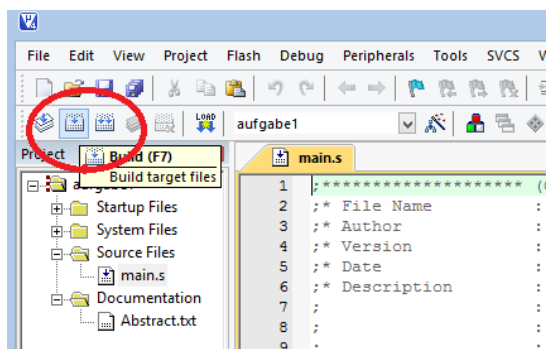
2. Es wird automatisch die Keil Entwicklungsumgebung aufgerufen.



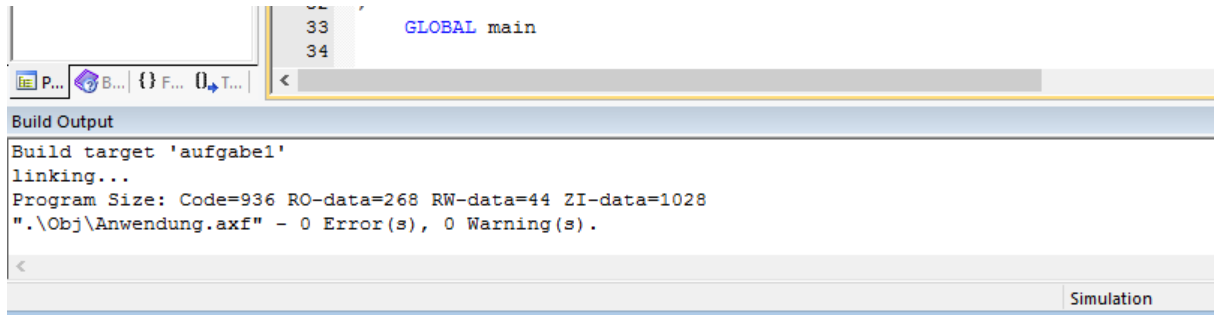
3. Die Datei **main.s** durch anklicken öffnen. Der Code eines einfachen Hauptprogramms erscheint im Editorfenster. Dieser Code wird durch das eigene Programm ersetzt.



4. Das Programm Übersetzen und Linken. Entweder mit der Taste F7, mit dem Menüpunkt **Project/Build Target** oder mit dem Icon in der Kommandozeile



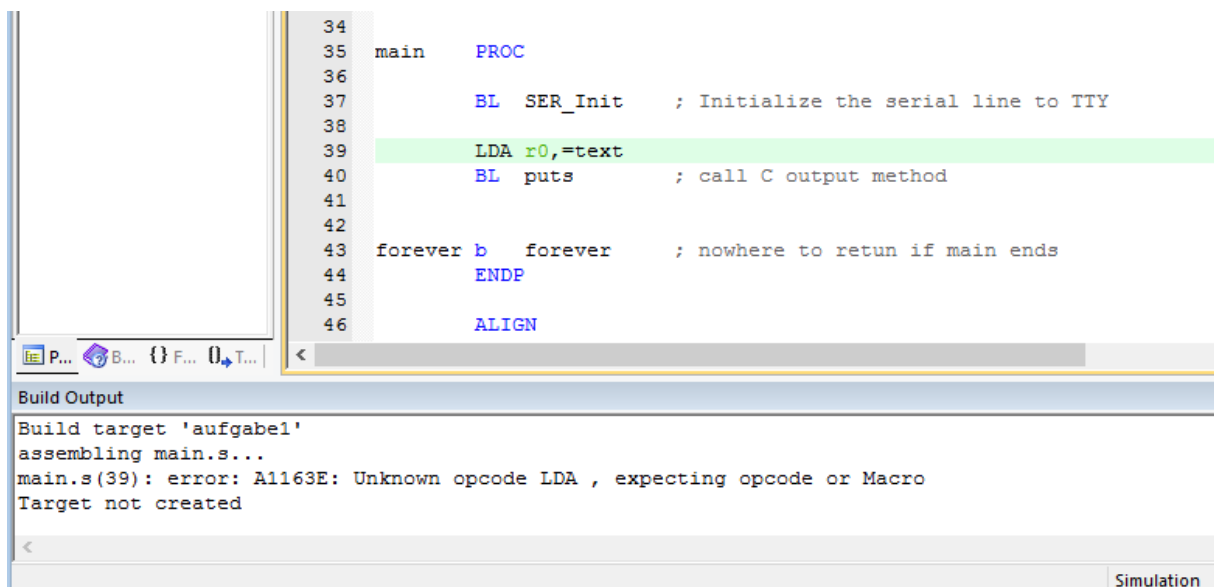
5. Im Fenster **Build Output** werden die Fehlermeldung des Vorgangs ausgegeben. Bei erfolgreicher Durchführung wird ausgegeben, dass die Datei **Anwendung.axf** angelegt wurde. Ohne Diese Datei kann das Programm nicht ausgeführt werden oder eine Fehlersuche durchgeführt werden.



The screenshot shows the 'Build Output' window of an IDE. The text inside reads: 'Build target 'aufgabe1'', 'linking...', 'Program Size: Code=936 RO-data=268 RW-data=44 ZI-data=1028', and '".\Obj\Anwendung.axf" - 0 Error(s), 0 Warning(s)'. The window has a toolbar with icons for Project, Build, Find, and Run, and a 'Simulation' button at the bottom right.

```
Build target 'aufgabe1'
linking...
Program Size: Code=936 RO-data=268 RW-data=44 ZI-data=1028
".\Obj\Anwendung.axf" - 0 Error(s), 0 Warning(s).
```

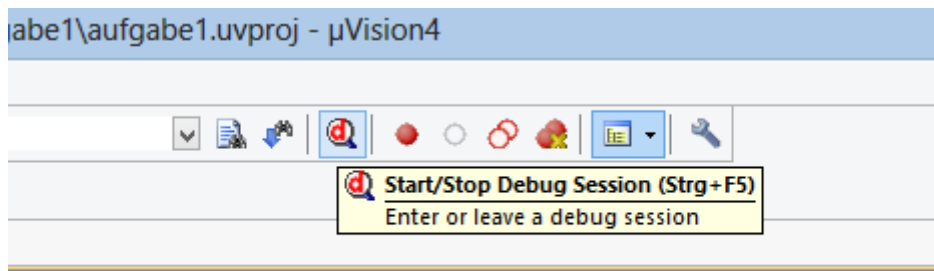
6. Sollten Fehler aufgetreten sein werden diese im **Build Output** Fenster angezeigt. Die fehlerhafte Zeile wird im Editor-Fenster durch anklicken der Fehlermeldung angezeigt. Das ausführbare Programm (Target) wird nicht angelegt.



The screenshot shows the 'Build Output' window with an error message: 'main.s(39): error: A1163E: Unknown opcode LDA , expecting opcode or Macro' and 'Target not created'. In the background, the assembly code is visible, with line 39 highlighted in green: 'LDA r0,=text'. The window has the same toolbar and 'Simulation' button as the previous screenshot.

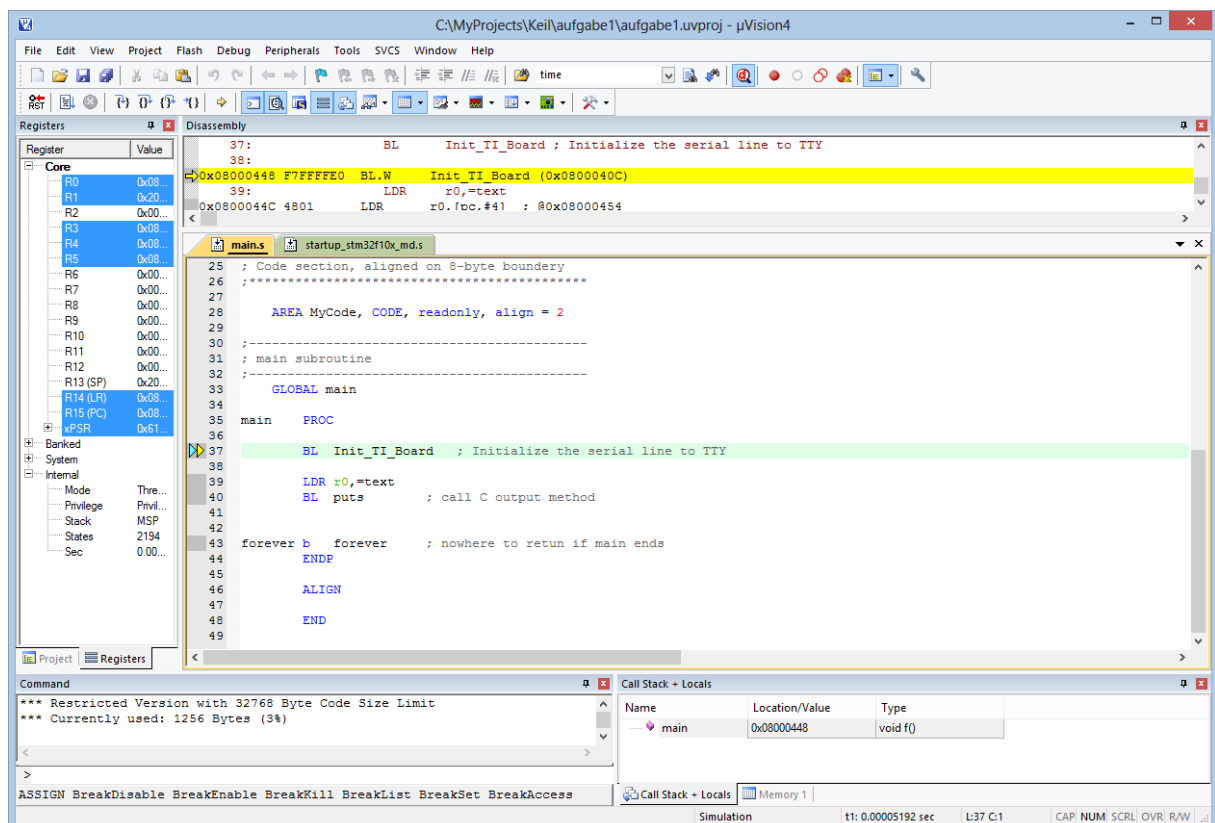
```
Build target 'aufgabe1'
assembling main.s...
main.s(39): error: A1163E: Unknown opcode LDA , expecting opcode or Macro
Target not created
```

7. Nach erfolgreicher Durchführung des „Build“-Vorgangs kann das Programm ausgeführt werden. Dazu den Button **Start/Stop Debug Session** klicken.

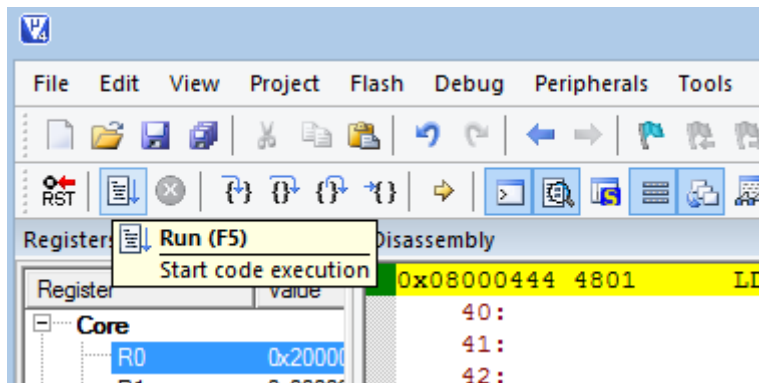


serial line
ion

8. Das Programm wird in den Simulationsspeicher geladen und das Programm wird am Anfang des Hauptprogramms angehalten. Es werden jetzt folgende Fenster angezeigt:
- Register des CORTEX Prozessors
 - Assemblercode, wie er aus dem Speicher gelesen wird
 - Editorfenster
 - Command Fenster
 - Fenster mit dem Stack und den lokalen Variablen, umschaltbar auf Darstellung des Speichers (RAM)



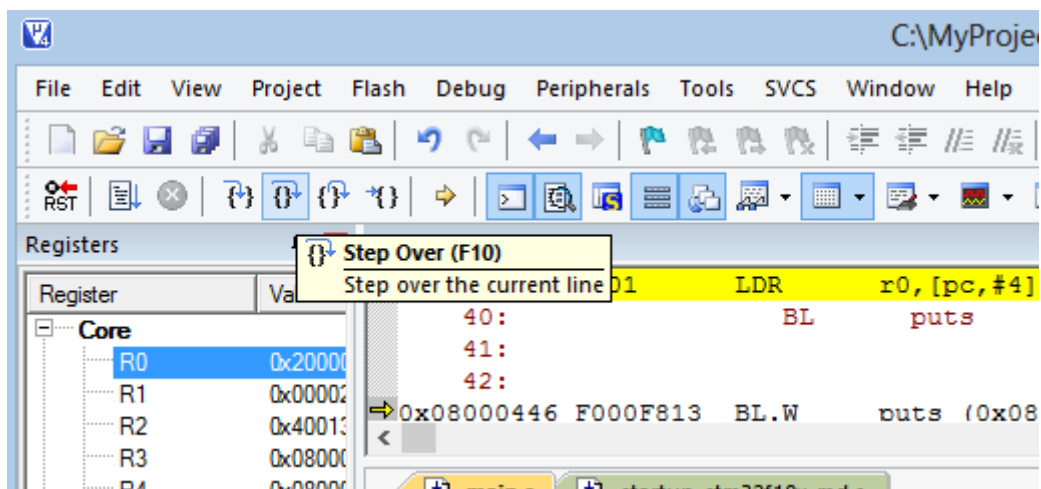
9. Das Programm kann nun mit dem Kommando **Run** ausgeführt werden.



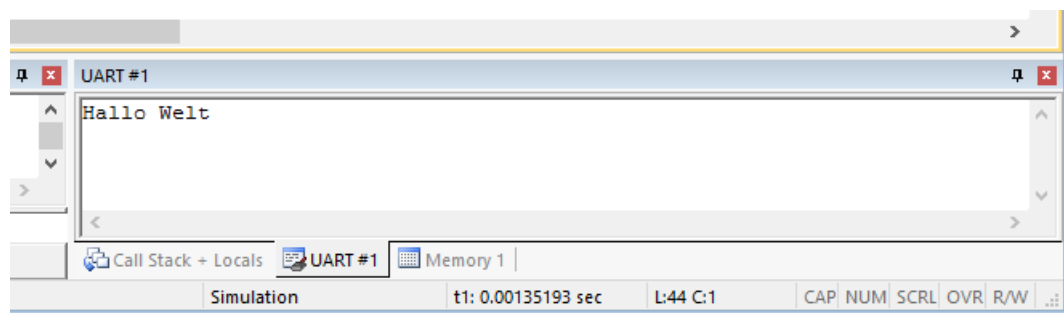
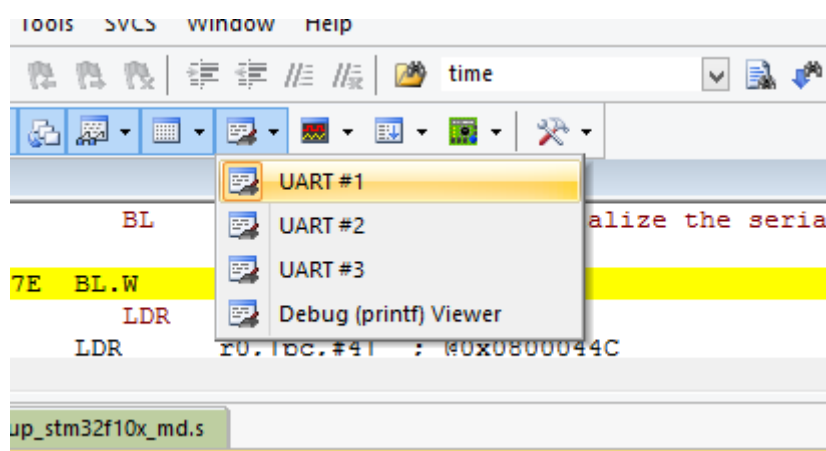
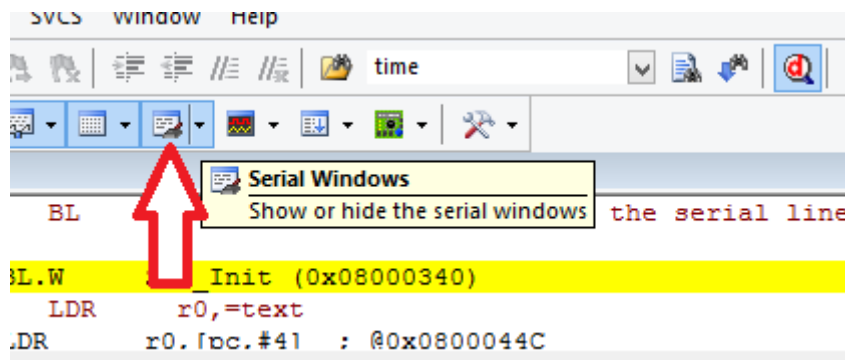
10. Eine Fehlersuche kann mit den Kommandos

- Step (F11)
- Step over (F10)
- Step out (Strg+F11)

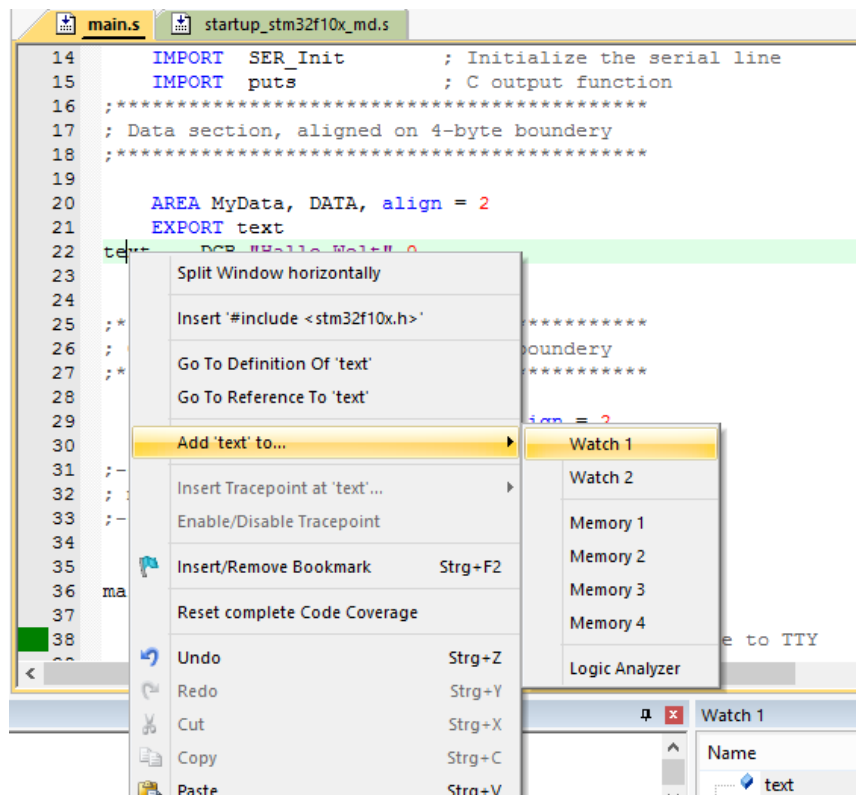
im Einzelschritt ausgeführt werden.



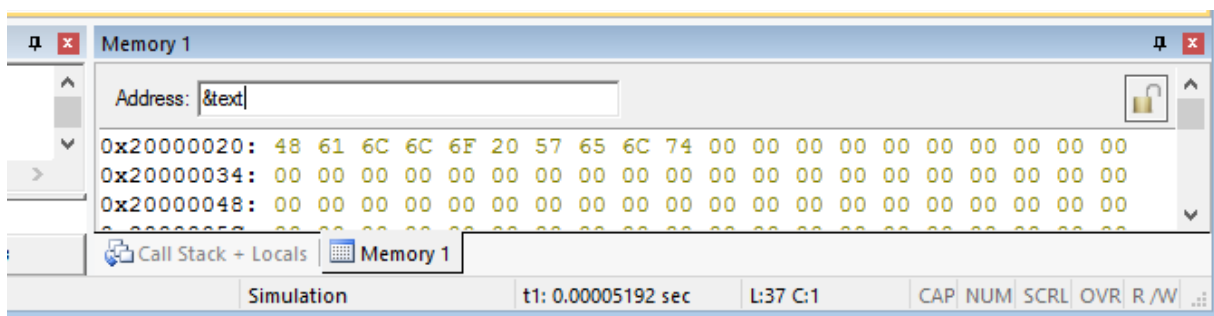
11. Eine Textausgabe kann im Simulator z.B. mit der C-Methode **puts()** erfolgen. Es muss ein Terminalfenster geöffnet werden, das mit dem UART1 verbunden ist. Die Eingabe von Daten erfolgt mit der Methode **getline()**



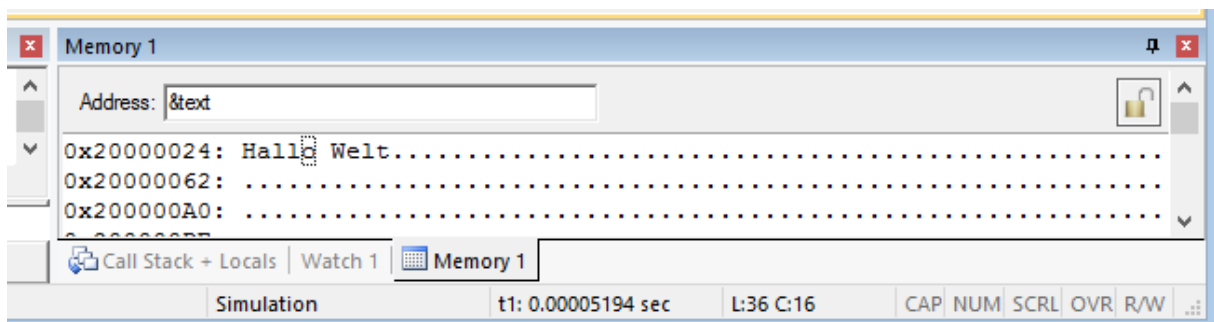
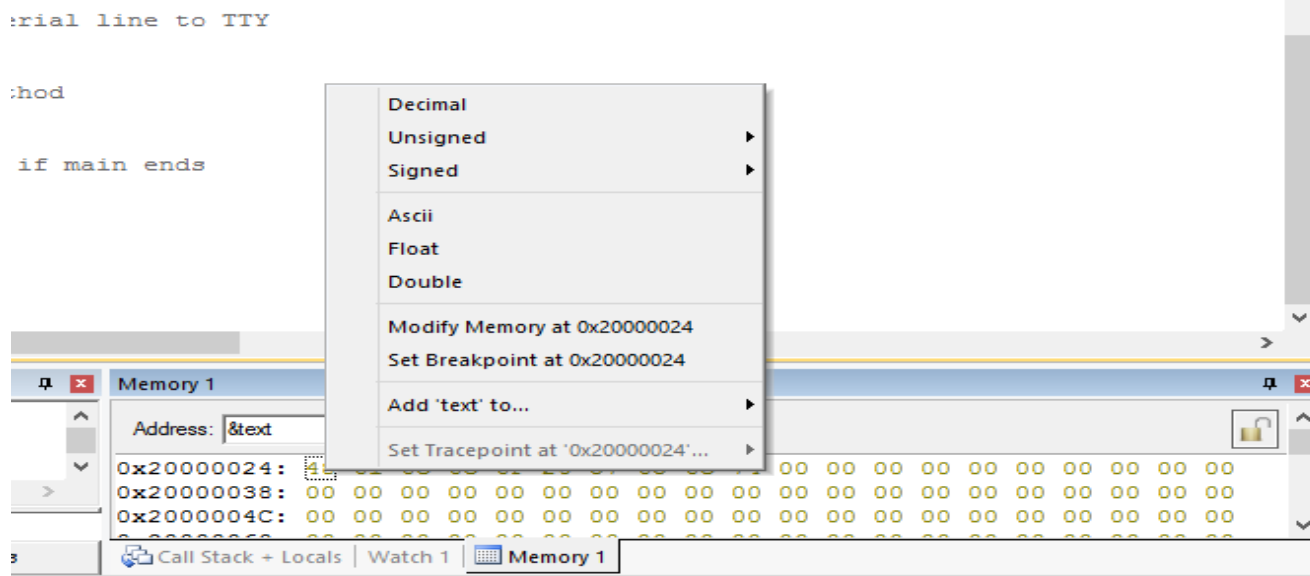
12. Eine Variable kann im **Watch**-Fenster angezeigt werden. Dazu wird im Editorfenster der Mauszeiger auf die Variable gesetzt und die rechte Maustaste betätigt. Mit **Add xx to Watch1** wird die Variable im Watch-Fenster angezeigt.



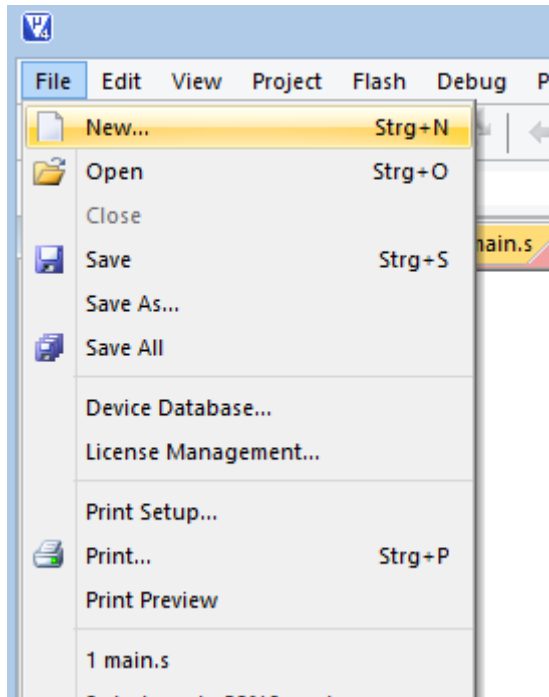
13. Eine Variable, die im Speicher liegt, kann im Memory-Fenster angesehen werden. Dem Namen der Variablen ist im Adressfeld ein &-Zeichen voranzustellen.

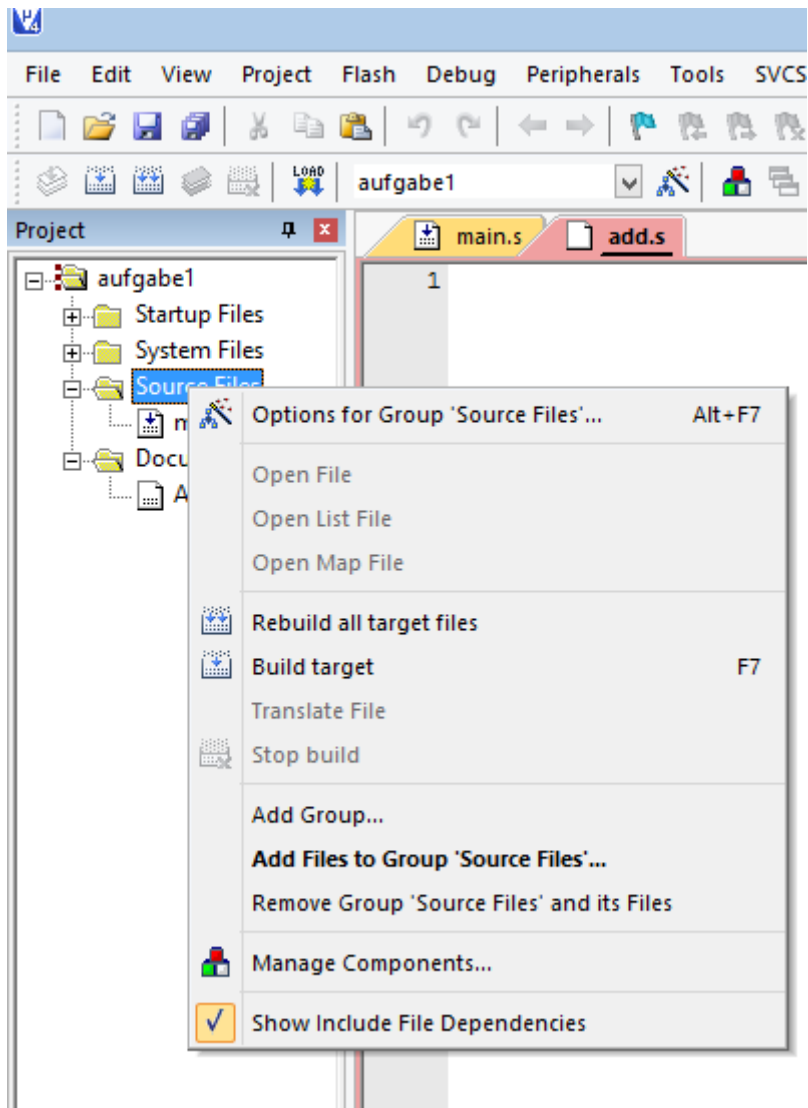


14. Die Darstellung der Werte im Memory-Fenster kann geändert werden, indem man den Mauszeiger auf einen Wert legt und dann die rechte Maustaste betätigt. Bei dem Feld **text** mit ASCII-Zeichen bietet sich die Auswahl **ASCII** an.



15. Das Anlegen einer neuen Datei erfolgt mit dem Kommando **File → New**. Es wird eine Datei mit dem Namen **Textn** angelegt. Diese Datei wird dann mit dem Kommando **File → Save As** als eine Quellcodedatei abgelegt, z.B. **add.s**. Diese Datei muss noch in das Projekt aufgenommen werden. Dazu den Mauszeiger im Project-Fenster auf **Source-Files** setzen und mit der rechten Maustaste klicken. Dann auswählen **Add Files to Group' Source Files'**. Danach wird die Datei dann beim nächsten „Build“-Vorgang mit übersetzt.





Die Entwicklung eines Programms für das TI-C-Board

Die Entwicklung eines C-Programms oder eines Assembler-Programms für das TI-C-Board ist identisch mit der eines Programms für die Simulation.

Ein Basisprojekt für die Entwicklung eines C-Programms wird durch anklicken des Icon **Keil C** angelegt.

Ein Basisprojekt für die Entwicklung eines Assembler-Programms mit dem Icon **Keil ASM**

Eine Textausgabe kann mit den normalen C-Methoden wie **printf()** oder **puts()** erfolgen. Die Eingabe mit Methoden wie **scanf()** oder **gets()**. Als Terminal steht das „**Serial Window**“ der Entwicklungsumgebung **nicht** zur Verfügung. Es muss das Terminal von Prof. Heitmann verwendet werden. Dieses Terminal steht als Icon **TI_Terminal** auf dem Desktop zur Verfügung.

Programme, die mit dem Simulator entwickelt worden sind, müssen in ein neues Projekt für das TI-C eingefügt werden. Es müssen die Quellcode Dateien in das neue Projekt übernommen werden. Eine direkte Umsetzung eines Projekts für den Simulator in ein Projekt für das TI-C-Board ist nicht möglich.