

# Wie benutze ich die Keil-Simulation und Stimuli

## Einführung

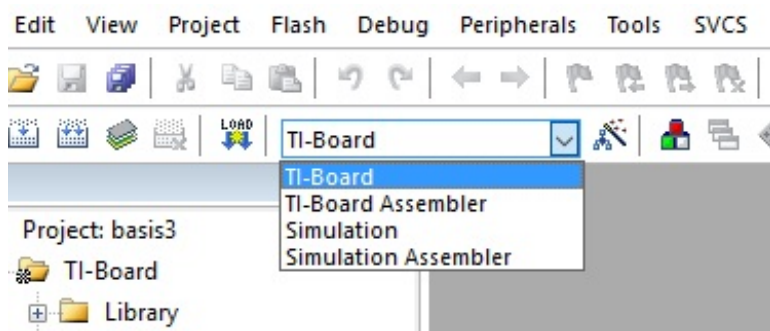
Die Simulation versucht das Verhalten des TI-Boards mit Einschränkungen nachzustellen. So soll den Studenten die Möglichkeit gegeben werden, ihren Code auch ohne TI-Board zu testen. Für Assembler-Projekte sind die Abschnitte **Target Auswahl**, **Simulation TFT-Display und LEDS**, **Toolbox** und **GTP stimuli** relevant.

## Target Auswahl

Das Basisprojekt hat vier Targets. Die Ziele **TI-Board** und **Simulation** beziehen sich auf C-Projekte. Zwischen diesen Targets kann gewechselt werden, um den gleichen C-Code entweder auf das TI-Board zu spielen oder zu simulieren.

Die Ziele **TI-Board Assembler** und **Simulation Assembler** beziehen sich auf Assembler-Projekte. Zwischen diesen Targets kann gewechselt werden, um den gleichen Assembler-Code entweder auf das TI-Board zu spielen oder zu simulieren.

Um das Target zu wechseln, klickt man einfach auf das Dropdown-Menü rechts neben dem Download-Button und wählt das entsprechende Target aus.



**Wichtig** Bei jedem Wechsel zwischen TI-Board und Simulation sollte ein "Rebuild" gemacht werden (nicht nur "Build"). Danach wird die Simulation über den Debug-Session-Button gestartet (genau so, wie man auch das TI-Board debuggen würde).

**Wenn man die Simulation resetten möchte, sollte die Debug-Session erneut gestartet werden, um zu gewährleisten, dass auch die simulierte Umgebung korrekt resettet wurde.**

## Der Code

Damit dein Code sowohl mit dem TI-Board als auch der Simulation kompatibel ist, sollte man statt der üblichen `stm32f4xx.h` das Header-File `TI_memory_map.h` includen. Nun hat man den Zugriff auf z.B. GPIOG und GPIOE, wie man es gewohnt ist, jedoch funktioniert es auch in der Simulation ohne den Code bei jedem Wechsel zu verändern.

Die Simulation läuft nicht in Echtzeit, daher wird empfohlen bei langen Delays oder ähnlichem den Code mit einem If-Then-Else-Block im Präprozessor etwas zu beschleunigen.

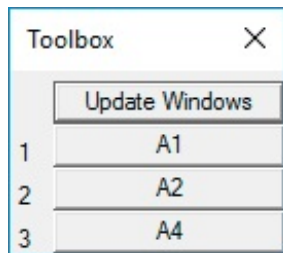
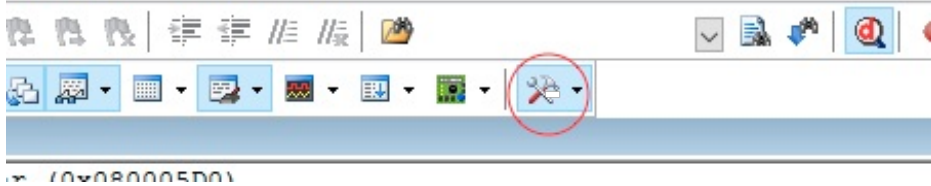
```
1  #ifdef SIMULATION
2      #define ONE_SEC_DELAY_IN_uS 1000 //hier nur 1ms fuer die Simulation
3  #else
4      #define ONE_SEC_DELAY_IN_uS 1000000 //hier wirklich 1s
5  #endif
```

## Toolbox

In der Toolbox stehen die **Stimuli** für die Praktika 1, 2 und 4 zur Verfügung. Falls das Toolbox-Fenster nicht automatisch geöffnet wird findet man es unter folgendem Button:

- µVision

ools SVCS Window Help



So sieht die Toolbox beim Starten der Simulation aus.

## Stimuli

Immer als ERSTES in der Toolbox die Aufgabe wählen! Erst danach sollte auf "Run" geklickt werden.

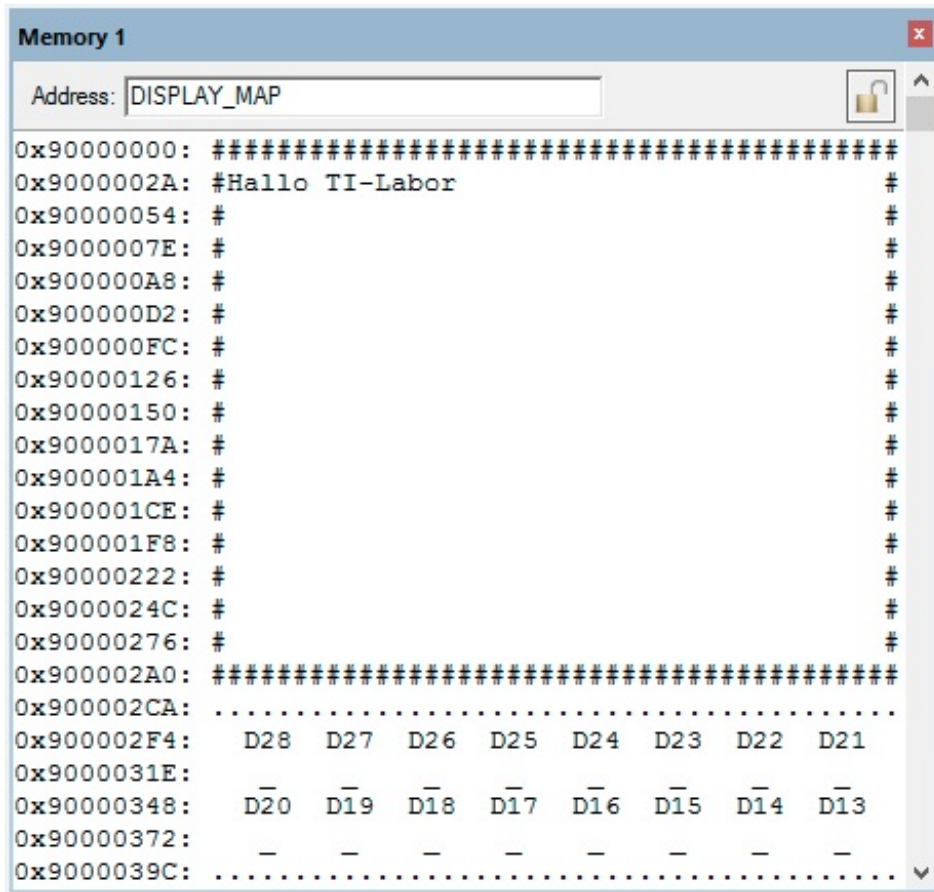
## Terminal

In der Simulation finden sie den "Terminal" bzw. die Serielle Schnittstelle (printf) in uVision im Menu unter: \***View** -> **Serial Windows** -> **UART #1**

## Simulation TFT-Display und LEDs

Das TFT-Display des TI-Boards wird im Speicher simuliert. Hierfür nimmt man einfach ein Memory-Fenster und gibt als Adresse "DISPLAY\_MAP" an. Die Darstellung kann man mit einem Rechtsklick auf ASCII umstellen. Als Umrandung des Displays sind Rauten dargestellt, so kann die richtige Breite für das Fenster gewählt werden. Unterhalb des Displays sind auch die Blauen LEDs dargestellt, welche am GPIOG liegen. Ein "\*" unterhalb der LED-Bezeichnung (z.B. D13) zeigt an, dass diese LED an ist. Ein "\_" steht für eine LED, welche aus ist.

**ES IST MÖGLICH, DASS DAS MEMORY-FENSTER NICHT GANZ AKTUELL IST. UM SICHER ZU GEHEN, DASS DIE ANZEIGE MIT AKTUELLEN WERTEN DARGESTELLT WIRD, SOLLTE DER ADRESSBEREICH NEUGELADEN WERDEN!**



## Logic Analyzer

Der Logic Analyzer ist unter **View -> Analysis Windows -> Logic Analyzer** zu finden. Klicke auf **Setup...** und anschließend auf **Import Signal Definitions...** um eine Einstellung zu laden. Die Files liegen in: */TIBRDLIB/Logic Analyzer \* Drehgeber.uvl \** OneWire.uvl

# Einschränkungen

## GPIO

Es sollte nur auf **GPIOE** und **GPIOG** zugegriffen werden. **GPIOG** unterstützt auf den unteren acht Bits alle GPIO Operationen (*ODR*, *IDR*, *MODER*, *OTYPER*...). Die oberen acht Bits des GPIOG sind als Outputs (LEDs D21 bis D28) initialisiert und können nicht über MODER oder OTYPER umkonfiguriert werden. **GPIOE** kann nicht umkonfiguriert werden und ist als Input Register initialisiert.

## Timer

Die Simulation unterstützt den Timer TIM2. Der Timer in der Simulation funktioniert auch ohne korrekte Initialisierung! Er läuft ab Beginn der Simulation. Somit kann man die Initialisierung des Timers in der Simulation nicht testen.

## Performance

In der Simulation wird ein anderer Prozessor verwendet, als auf dem TI-Board daher kann es Abweichungen in der Performance geben. In der Tendenz sollte das TI-Board schneller sein als das simulierte Board. Besonders wenn viele Floating-Point-Berechnungen gemacht werden.

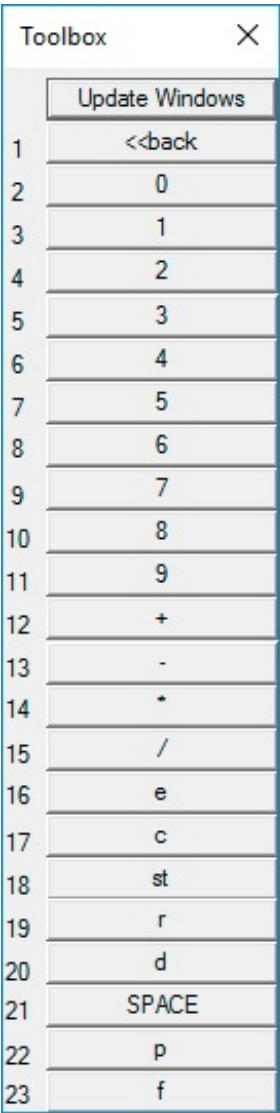
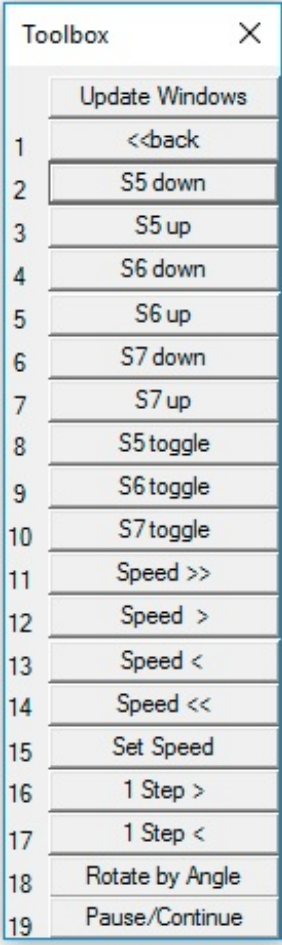
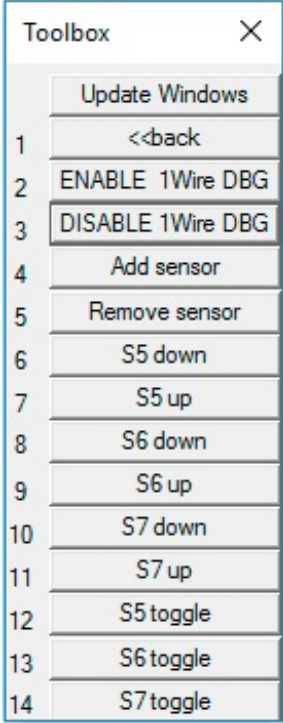
## Display

Das simulierte Display unterstützt nicht alle Funktionen des TFT auf dem TI-Board. Alle nicht unterstützten Funktionen werden einfach ignoriert. Somit ist auch das Keypad auf dem Display nicht sichtbar. Implementierte Funktionen: - gotoXY - setWindow - putc - puts

## Interrupts

Interrupts werden in der Simulation nicht unterstützt. Diese können nur auf dem TI-Board verwendet werden.

## Stimuli für das GSP

A1	A2	A4
		

### A1: Taschenrechner RPN

Bezieht sich auf den Taschenrechner. Die hier angezeigten Buttons simulieren die Touch-Eingabe auf dem TFT. Wenn man auf einen Button klickt, wird dieses Zeichen bis zur nächsten Abfrage am Display gespeichert und dann übertragen. Wenn die Abfrage am Display noch nicht erfolgt ist und ein weiterer Button gedrückt wird, so wird das Zeichen überschrieben und nur das zuletzt gedrückte Zeichen wird übertragen.

### A2: Drehgeber

Bezieht sich auf den Drehgeber. Zum Einen können die Buttons S5, S6 und S7 gedrückt oder getoggelt werden. Zum Anderen kann die Rotation des Drehgebers in Einer- oder Zehnerschritten gewählt werden. Alternativ kann auch im Command-Window `rotarySpeed = ###` mit entsprechender Geschwindigkeit statt `###` in deg/s eingegeben werden.

### A4: OneWire

Bezieht sich auf den One-Wire Temperatur-Sensor. TODO: Anzahl der Sensoren wählen

## Eigene Stimuli

In der Datei "**userSIM.ini**" können eigene Buttons erstellt werden oder automatisch Module für eine Aufgabe geladen werden.

Weitere Informationen zu der Simulationsumgebung findet man unter:

[http://www.keil.com/support/man/docs/uv4/uv4\\_debug\\_commands.htm](http://www.keil.com/support/man/docs/uv4/uv4_debug_commands.htm)

[http://www.keil.com/support/man/docs/uv4/uv4\\_debug\\_functions.htm](http://www.keil.com/support/man/docs/uv4/uv4_debug_functions.htm)

[http://www.keil.com/support/man/docs/uv4/uv4\\_simulation.htm](http://www.keil.com/support/man/docs/uv4/uv4_simulation.htm)