



—

Simulationsumgebung für digitale und analoge Ein- und Ausgänge

—

Benutzerdokumentation

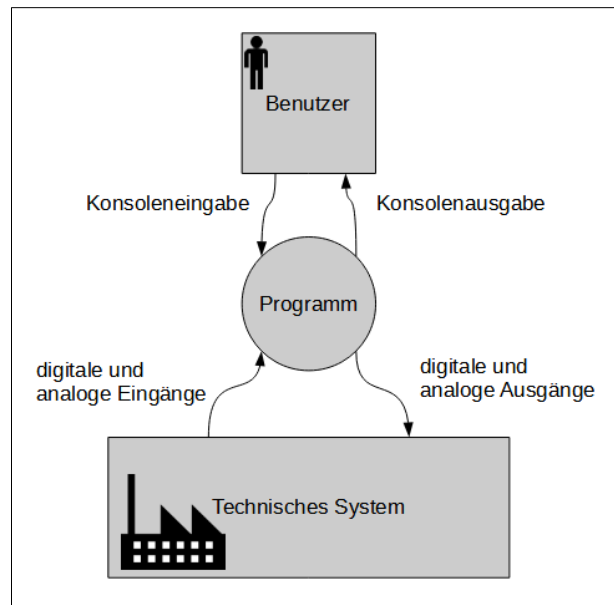
| | |
|---------|---------------|
| Autor | Markus Breuer |
| Version | 0.6.4a |
| Datum | 29.03.2026 |

Inhaltsverzeichnis

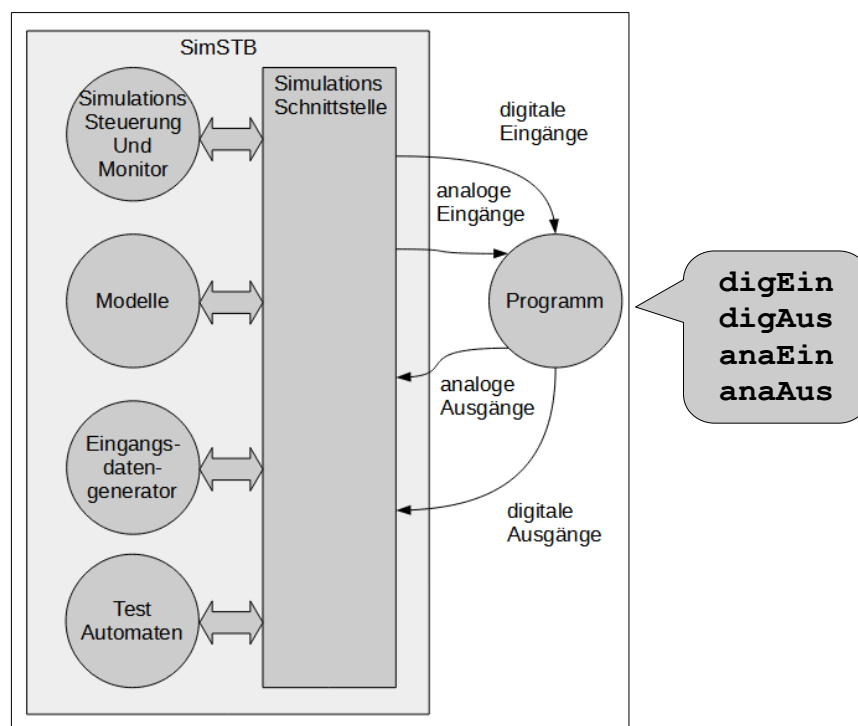
| | |
|--|----|
| 1 Allgemeines..... | 2 |
| 2 Lokale Installation der Simulationsumgebung..... | 4 |
| 3 Steuerung der Simulationsumgebung SimSTB..... | 5 |
| 4 Erstellung eigener Programme mit Python..... | 8 |
| 5 SimSTB Ein- und Ausgangsbelegung..... | 10 |

1 Allgemeines

Oft muss ein Programm nicht nur über die Konsole oder eine graphische Benutzeroberfläche mit dem Benutzer kommunizieren, sondern auch über analoge und digitale Schnittstellen mit einem technischen System.



Die Simulationsumgebung **SimSTB** erlaubt es, dies für Schulungszwecke auch ohne zusätzliche Hardware mittels Simulation durchzuführen.



SimSTB besteht aus zwei Teilen:

1. Der eigentlichen **Simulationsumgebung** und den **Steuerprogrammen** für diese. Die Steuerprogramme sind in Kapitel 3 beschrieben. Die Installation in Kapitel 2 beschrieben.
2. Einer **Programmier-Schnittstelle**, um aus eigenen Programmen die Simulationsumgebung zu nutzen. Es ist eine **Python-Schnittstelle** vorhanden. In Kapitel 4 ist beschrieben, wie Sie eigene Programme erstellen können.

Um die Simulationsumgebung SimSTB aus eigenen Programmen zu nutzen, stehen 4 Funktionen zur Verfügung.

| Schnittstelle | Funktion | Kanäle/id | Typ | Richtung |
|-------------------|----------|-----------|---------|----------|
| Digitaler Eingang | digEin | 0 .. 15 | digital | Eingang |
| Digitaler Ausgang | digAus | 0 .. 15 | digital | Ausgang |
| Analoger Eingang | anaEin | 0 .. 7 | analog | Eingang |
| Analoger Ausgang | anaAus | 0 .. 7 | analog | Ausgang |

Die Kanäle bestimmen die Anzahl der jeweiligen Schnittstellen.

2 Lokale Installation der Simulationsumgebung

2.1 Voraussetzungen

- Python 3.10 und neuer, sowie pip müssen auf dem Rechner vorhanden sein.

2.2 Installation

1. Installieren Sie das Simulator-Paket mit dem Befehl `pip install simstb`.
2. Prüfen Sie mit `pip list`, ob das Paket installiert wurde.
3. Prüfen Sie mit `simstb_cli --version`, ob das Kommandozeilenwerkzeug korrekt installiert wurde.
4. Bauen Sie mit `simstb_cli --init` die Laufzeitumgebung auf. Hierzu wird ein Ordner `sim` mit der Laufzeitumgebung im aktuellen Arbeitsverzeichnis angelegt. Achten Sie darauf, sich beim Aufruf im richtigen Verzeichnis zu befinden.
5. Kontrollieren Sie, ob folgende Verzeichnis-Struktur und Dateien vorhanden sind.

```

sim
├── config.toml
├── CONTRIBUTING.md
├── data
│   ├── anaaus.txt
│   ├── anaein.txt
│   ├── digaus.txt
│   └── digein.txt
├── doc
│   ├── beispiel.py
│   ├── bilder
│   │   ├── band_links.gif
│   │   ├── band_rechts.gif
│   │   └── ...
│   └── SimSTB-Benutzerdokumentation.pdf
├── LICENSE.txt
├── modelle.json
└── README.md

```

6. Erstellen Sie eine Umgebungsvariable namens `SIMSTB_WURZEL`, welches auf das Simulationsverzeichnis `sim` zeigt.

2.3 Quelldateien

Bei der Simulationsumgebung SimSTB handelt es sich um Open Source Software. Diese steht über GitHub (<https://markusbreuer1964.github.io/SimSTB/>) zur Verfügung.

3 Steuerung der Simulationsumgebung SimSTB

1. Simulations Steuerung und Monitoring

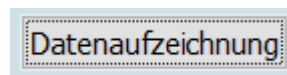
Mit Hilfe des Programms `simstb_gui` können Sie digitalen und analogen Ein- und Ausgänge überwachen und die Eingänge setzen. Die Werte werden im Sekundentakt aktualisiert. Starten Sie `simstb_gui` im Terminal.

- Durch Mausklick auf die digitalen Eingänge können Sie bei der Eingangsbelegung zwischen 0 und 1 wechseln.
- Nach Auswahl eines analogen Eingangs können Sie dort einen analogen Zahlenwert

- eingeben. Als Dezimaltrennzeichen müssen Sie einen Punkt benutzen.
- Mit Hilfe der nebenstehenden Knöpfe können Sie ganze Gruppen von Ein- bzw. Ausgängen zusammen setzen.



- Mit Hilfe es Knopfs Datenaufzeichnung können Sie eine Datenaufzeichnung starten.
-



Mit Hilfe der Knöpfe **Starten** und **Stoppen** kontrollieren Sie die Datenaufzeichnung. Solange die Aufzeichnung läuft werden jeweils alle digitalen und analogen Ein- und Ausgabewerte in eine CSV-Datei geschrieben

- Mit Hilfe des Knopfs **Funktionsgenerator** können Sie einen analogen Funktionsgenerator starten.

Funktionsgenerator

| Kanal | An/Aus | Signalform | Amplitude | P.Dauer (s) |
|-------|--------------------------|------------|-----------|-------------|
| AE0 | <input type="checkbox"/> | Zufall | 0 | 0 |
| AE1 | <input type="checkbox"/> | Zufall | 0 | 0 |
| AE2 | <input type="checkbox"/> | Zufall | 0 | 0 |
| AE3 | <input type="checkbox"/> | Zufall | 0 | 0 |
| AE4 | <input type="checkbox"/> | Zufall | 0 | 0 |
| AE5 | <input type="checkbox"/> | Zufall | 0 | 0 |
| AE6 | <input type="checkbox"/> | Zufall | 0 | 0 |
| AE7 | <input type="checkbox"/> | Zufall | 0 | 0 |

Funktionsgenerator Steuerung

Starten Stoppen Generator nicht aktiv

Schließen

Hier können Sie einzelnen analogen Eingangskanäle aktivieren, die jeweilige Signalform auswählen und Parameter zur Signalform einstellen. Mit Hilfe der Knöpfe **Starten** und **Stoppen** kontrollieren Sie den Funktionsgenerator.

4 Erstellung eigener Programme mit Python

Benötigte Python Pakete:

Damit die Python-Schnittstelle genutzt werden kann, muss das Paket `simstb` installiert sein. Details sind dem Kapitel „2 Lokale Installation der Simulationsumgebung“ zu entnehmen.

```
pip install simstb
```

Voraussetzungen:

Damit das eigene Programm die Schnittstellenfunktionen nutzen kann muss man die Datei `simulator.py` importieren. Hierzu dient die folgende Anweisung:

```
import sim_schnittstelle.simulator as sim
```

Schnittstellenfunktionen:

```
DIGMAXLAENGE = 16
ANAMAXLAENGE = 8

def dig_ein( id)
def dig_aus( id, wert)

def ana_ein( id)
def ana_aus( id, wert)
```

- Der Parameter `id` ist vom Datentyp `int`.
- `id` ist zwischen 0 und 15 bei den beiden Funktionen für die digitale Ein- und Ausgabe.
- `id` ist zwischen 0 und 7 bei den beiden Funktionen für die analoge Ein- und Ausgabe.
- Der Parameter `wert` ist vom Datentyp `bool` bei der digitalen Ausgabe, vom Datentyp `float` bei der analogen Ausgabe.
- Die Funktion `dig_ein` gibt einen Wert vom Datentyp `bool` zurück. Die Funktion `ana_ein` gibt einen Wert vom Datentyp `float` zurück.
- Wenn ein ungültiger Index genutzt wird, geben `dig_ein` und `ana_ein` `None` zurück.
- Wenn ein ungültiger Index genutzt wird, speichern `dig_aus` und `ana_aus` nichts ab.

Log-Datei:

- Im Wurzelverzeichnis `sim` befindet sich eine Log-Datei namens `simstb.log`, welche Hinweise über aufgetretene Fehlsituationen gibt

Beispiel-Code (Auszug; vollständig in Unterordner `sim/doc`):

```
...
import sim_schnittstelle.simulator as sim
...
def test():
    """ Testfunktion """
    ende = False
    ...
    while ende != True:
        wert = sim.ana_ein( 0)
        print(wert)
        time.sleep( 1)
        ende = sim.dig_ein( 0)

    sim.dig_aus( 15, 1)
    sim.ana_aus( 7, -123.456)
    ...


test()
```

Analoger Eingang einlesen

Digitaler Eingang einlesen

Digitaler und
Analoger Ausgang setzen

5 SimSTB Ein- und Ausgangsbelegung

| SinSTB Ein- und Ausgangsbelegung  | | |
|--|----------------------------|--|
| Digital Eingänge | | |
| | <input type="radio"/> DE0 | |
| | <input type="radio"/> DE1 | |
| | <input type="radio"/> DE2 | |
| | <input type="radio"/> DE3 | |
| | <input type="radio"/> DE4 | |
| | <input type="radio"/> DE5 | |
| | <input type="radio"/> DE6 | |
| | <input type="radio"/> DE7 | |
| | <input type="radio"/> DE8 | |
| | <input type="radio"/> DE9 | |
| | <input type="radio"/> DE10 | |
| | <input type="radio"/> DE11 | |
| | <input type="radio"/> DE12 | |
| | <input type="radio"/> DE13 | |
| | <input type="radio"/> DE14 | |
| | <input type="radio"/> DE15 | |
| Analoge Eingänge | | |
| | <input type="radio"/> AE0 | |
| | <input type="radio"/> AE1 | |
| | <input type="radio"/> AE2 | |
| | <input type="radio"/> AE3 | |
| | <input type="radio"/> AE4 | |
| | <input type="radio"/> AE5 | |
| | <input type="radio"/> AE6 | |
| | <input type="radio"/> AE7 | |
| Digitale Ausgänge | | |
| DA0 | <input type="radio"/> | |
| DA1 | <input type="radio"/> | |
| DA2 | <input type="radio"/> | |
| DA3 | <input type="radio"/> | |
| DA4 | <input type="radio"/> | |
| DA5 | <input type="radio"/> | |
| DA6 | <input type="radio"/> | |
| DA7 | <input type="radio"/> | |
| DA8 | <input type="radio"/> | |
| DA9 | <input type="radio"/> | |
| DA10 | <input type="radio"/> | |
| DA11 | <input type="radio"/> | |
| DA12 | <input type="radio"/> | |
| DA13 | <input type="radio"/> | |
| DA14 | <input type="radio"/> | |
| DA15 | <input type="radio"/> | |
| Analoge Ausgänge | | |
| AA0 | <input type="radio"/> | |
| AA1 | <input type="radio"/> | |
| AA2 | <input type="radio"/> | |
| AA3 | <input type="radio"/> | |
| AA4 | <input type="radio"/> | |
| AA5 | <input type="radio"/> | |
| AA6 | <input type="radio"/> | |
| AA7 | <input type="radio"/> | |