



Generative AI for Creating Test and Measurement Procedures

SweepMe!

We quickly combine instruments from different vendors!

Our users can create their own measurement procedures in a low-code/no-code environment based on modular contents.





Approach

Our modularized software allows for fast automation!

Ready-made modules can be quickly assembled to program sequences by drag & drop.

Interactive dashboard for fast visualization.

 **Free software + paid add-on modules**

 **Highly reusable & sustainable content**

 **Shared generic instrument drivers**

How to make use of generative AI?

What is the best way to add generative AI when users mainly work with a graphical user interface?

Documentation assistance

„Why creating documentation if people don't read it?“

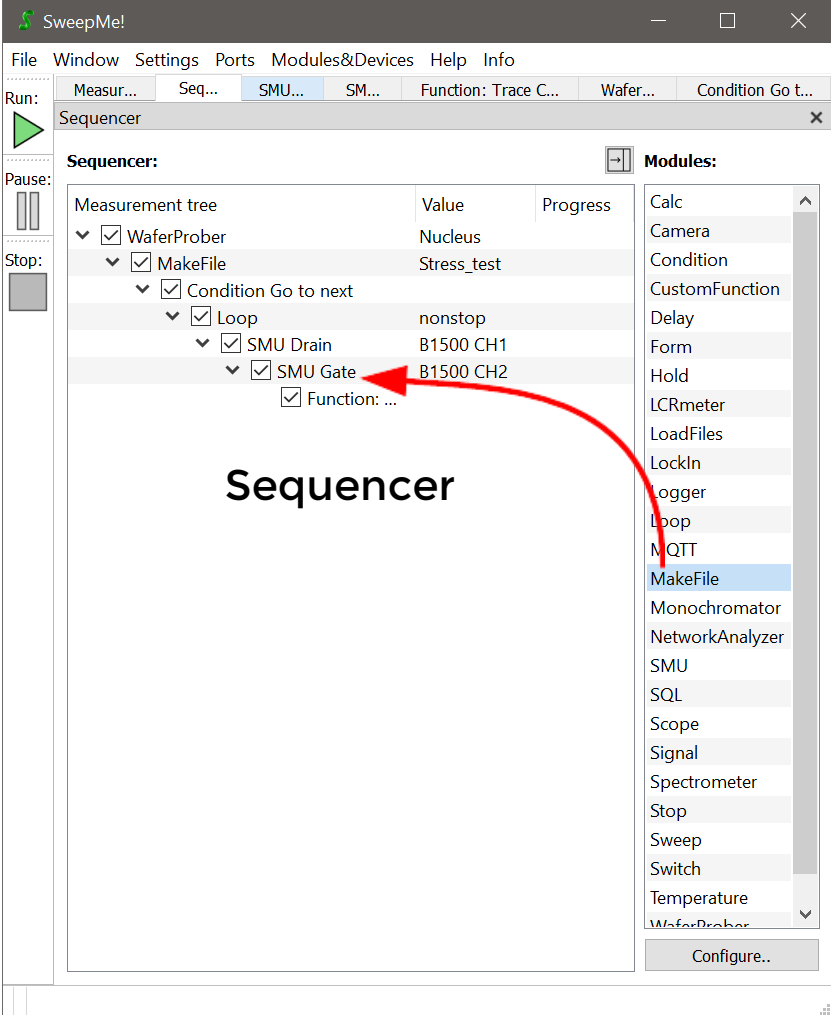
Instrument driver creation

„How to implement thousands of instruments?“

Sequencing

„How to create programs without testing every line of code?“

Gameplay



The screenshot shows the SweepMe! Sequencer interface. The main window is titled "SweepMe!" and has a menu bar with "File", "Window", "Settings", "Ports", "Modules&Devices", "Help", and "Info". Below the menu bar is a toolbar with buttons for "Measur...", "Seq...", "SMU...", "SM...", "Function: Trace C...", "Wafer...", and "Condition Go t...". The "Seq..." button is selected, and the "Sequencer" window is open.

The "Sequencer" window has a "Run:" button (a green play icon) and a "Pause:" button (two vertical bars). Below these is a "Stop:" button (a grey square). The main area is divided into two panes: "Sequencer:" and "Modules:".

The "Sequencer:" pane contains a "Measurement tree" table with columns "Value" and "Progress". The tree is expanded to show the following structure:

| Measurement tree | Value | Progress |
|--|-------------|----------|
| <input checked="" type="checkbox"/> WaferProber | Nucleus | |
| <input checked="" type="checkbox"/> MakeFile | Stress_test | |
| <input checked="" type="checkbox"/> Condition Go to next | | |
| <input checked="" type="checkbox"/> Loop | nonstop | |
| <input checked="" type="checkbox"/> SMU Drain | B1500 CH1 | |
| <input checked="" type="checkbox"/> SMU Gate | B1500 CH2 | |
| <input checked="" type="checkbox"/> Function: ... | | |

A red arrow points from the "SMU Gate" row in the table to the "MakeFile" module in the "Modules:" list. The "MakeFile" module is highlighted in blue. The word "Sequencer" is written in large black text in the center of the window.

The "Modules:" pane contains a list of modules: Calc, Camera, Condition, CustomFunction, Delay, Form, Hold, LCRmeter, LoadFiles, LockIn, logger, Loop, MQTT, MakeFile (highlighted), Monochromator, NetworkAnalyzer, SMU, SQL, Scope, Signal, Spectrometer, Stop, Sweep, Switch, Temperature, WaferProber. A "Configure.." button is at the bottom of the list.

Gameplay

The screenshot displays the SweepMe! application window. The main menu includes File, Window, Settings, Ports, Modules&Devices, Help, and Info. Below the menu is a toolbar with buttons for Measur..., Seq..., SMU..., SM..., Function: Trace C..., Wafer..., and Condition Go t... The Sequencer window is active, showing a measurement tree and a list of modules.

Sequencer:

| Measurement tree | Value | Progress |
|--|-------------|----------|
| <input checked="" type="checkbox"/> WaferProber | Nucleus | |
| <input checked="" type="checkbox"/> MakeFile | Stress_test | |
| <input checked="" type="checkbox"/> Condition Go to next | | |
| <input checked="" type="checkbox"/> Loop | nonstop | |
| <input checked="" type="checkbox"/> SMU Drain | B1500 CH1 | |
| <input checked="" type="checkbox"/> SMU Gate | B1500 CH2 | |
| <input checked="" type="checkbox"/> Function: ... | | |

Modules:

- Calc
- Camera
- Condition
- CustomFunction
- Delay
- Form
- Hold
- LCRmeter
- LoadFiles
- LockIn
- logger
- Loop
- MQTT
- MakeFile
- Monochromator
- NetworkAnalyzer
- SMU
- SQL
- Scope
- Signal
- Spectrometer
- Stop
- Sweep
- Switch
- Temperature
- WaferProber

Buttons: Run (green play icon), Pause (vertical bars), Stop (grey square), and Configure..

The word "Sequencer" is written in large black text in the center of the measurement tree area.

Two red arrows point from the "MakeFile" module in the Modules list to the "Function: Trace C..." button in the toolbar and the "Function: ..." entry in the Sequencer measurement tree.

Gameplay

The main interface of SweepMe! shows a menu bar (File, Window, Settings, Ports, Modules&Devices, Help, Info) and a toolbar with buttons for Run, Pause, and Stop. The Sequencer panel displays a measurement tree with the following structure:

- Measurement tree
 - WaferProber
 - MakeFile
 - Condition Go to next
 - Loop
 - SMU Drain (B1500 CH1)
 - SMU Gate (B1500 CH2)
 - Function: ...

The Modules panel on the right lists various modules, with 'WaferProber' selected and highlighted in blue. A 'Configure..' button is located at the bottom of the Modules list.

The WaferProber configuration dialog box contains the following fields and controls:

- WaferProber label: WaferProber
- Device: WaferProber-Test_template (with Open/Modify button)
- Port: 0 (with Find Ports button)
- Sweep value die: Die table
- Sweep value subsite: Subsite table
- Actual values:
 - Die: [Empty text box]
 - Subsite: [Empty text box]
 - x in mm: [Empty text box]
 - y in mm: [Empty text box]
- Probe plan: (with Update button)

| Die table | | Subsite table | |
|-----------|--|---------------|---|
| 1 | <input checked="" type="checkbox"/> Test index 1 | 1 | <input checked="" type="checkbox"/> 1 (Label) |
| 2 | <input checked="" type="checkbox"/> Test index 2 | 2 | <input checked="" type="checkbox"/> 2 (Label) |
| 3 | <input checked="" type="checkbox"/> Test index 3 | 3 | <input checked="" type="checkbox"/> 3 (Label) |
| 4 | <input checked="" type="checkbox"/> Test index 4 | 4 | <input checked="" type="checkbox"/> 4 (Label) |
| 5 | <input checked="" type="checkbox"/> Test index 5 | 5 | <input checked="" type="checkbox"/> 5 (Label) |
| 6 | <input checked="" type="checkbox"/> Test index 6 | 6 | <input checked="" type="checkbox"/> 6 (Label) |

- Buttons: Go to next die, Go to next subsite
- Footer: Right-click on a table item to find further actions

Module

Gameplay

The image displays the SweepMe! software interface, which is used for controlling measurement equipment. It is divided into several main sections:

- Sequencer:** A tree view showing a sequence of operations. The 'SMU Gate' module is selected, with a value of 'B1500 CH2'. Other modules include 'WaferProber', 'MakeFile', 'Condition Go to next', and 'Loop'.
- Modules:** A list of available modules such as 'Calc', 'Camera', 'Condition', 'CustomFunction', 'Delay', 'Form', 'Hold', 'LCRmeter', 'LoadFiles', 'LockIn', 'logger', 'Loop', 'MQTT', 'MakeFile', 'Monochrom', 'NetworkAn', 'SMU', 'SQL', 'Scope', 'Signal', 'Spectromet', 'Stop', 'Sweep', 'Switch', 'Temperatur', 'WaferProber', and 'Config'.
- WaferProber Configuration:** A dialog box for configuring the 'WaferProber' module. It includes fields for 'WaferProber label', 'Device' (set to 'WaferProber-Test_template'), 'Port' (set to '0'), 'Sweep value die' (set to 'Die table'), and 'Sweep value subsite' (set to 'Subsite table'). There are also input fields for 'Actual values' (Die, Subsite, x in mm, y in mm) and a 'Go to next subsite' button.
- Python Instrument Driver:** A Notepad window showing the Python code for the 'WaferProber-Cascade_Nucleus/main.py' file. The code defines 'configure(self)' and 'unconfigure(self)' methods that interact with the device via a serial port.

Red arrows indicate the flow of information: from the 'SMU Gate' module in the Sequencer to the 'WaferProber' configuration dialog, and from the 'WaferProber' configuration dialog to the Python code editor.

Python instrument driver

Low code tools and Generative AI

Generative AI can be a great addition to Low code/No code tools

Documentation assistance

„Why creating documentation if people don't read it?“

Instrument driver creation

„How to implement thousands of instruments?“

Sequencing

„How to create programs without testing every line of code?“

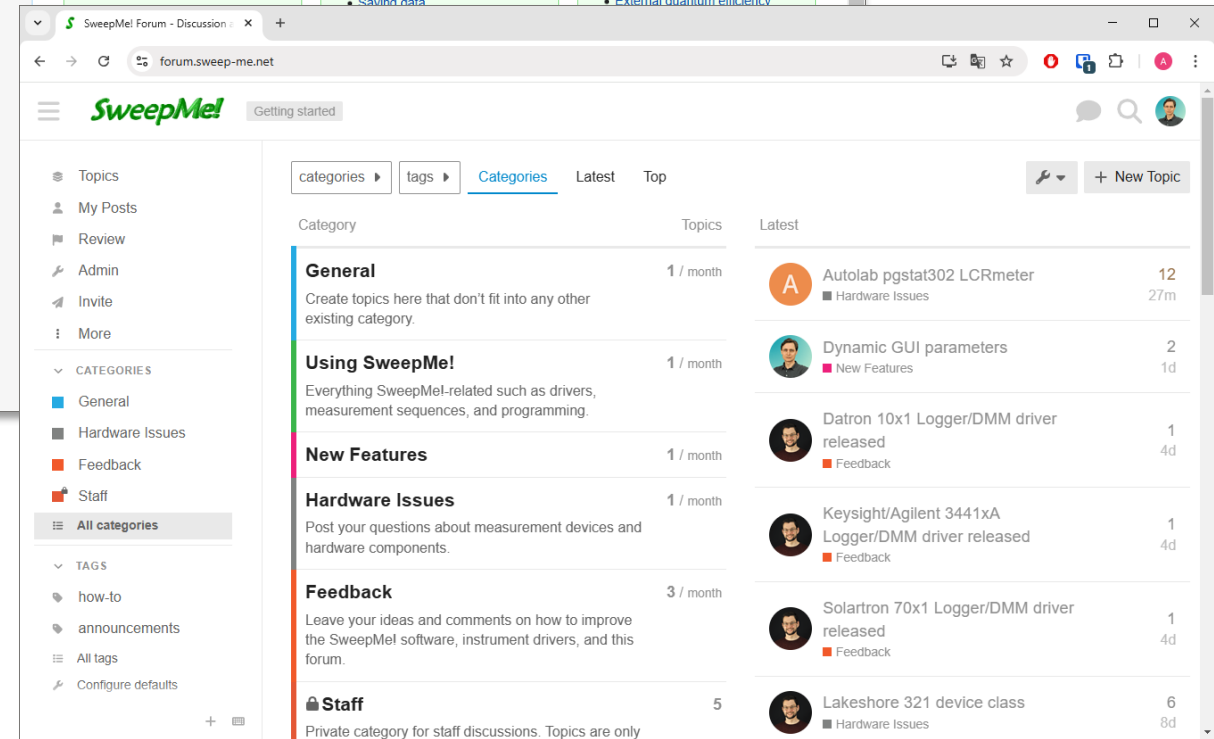
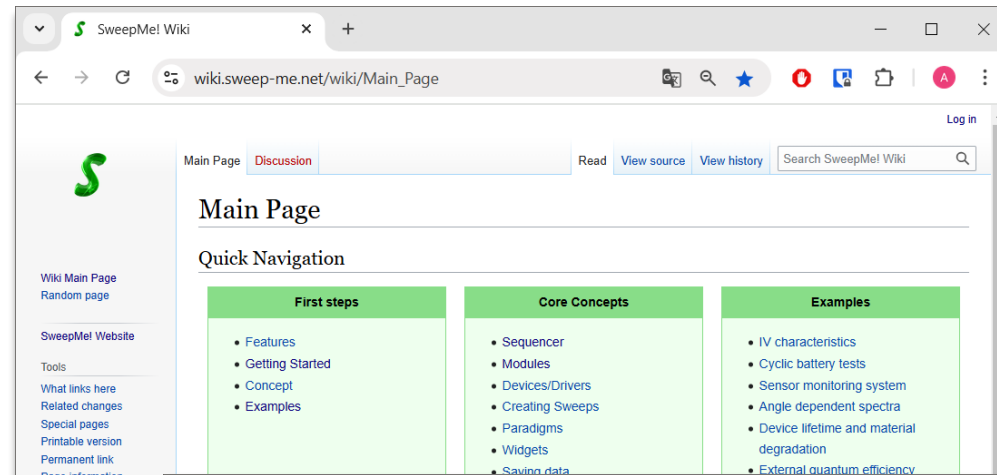
Ways to use LLMs

LLMs can be customized and tailored to own products and services

- Chat (Q&A with additional prompts)
- CustomGPT (Interface with pre-configured prompt)
- AI Assistants (Pre-trained data files + API)
- Fine-tuned models (Model with re-fined parameters)

Documentation

- Crawling knowledge from Wiki, Forum, ...
- Uploading training data
- Problems with questions that need information from different sources



SweepMe! Driver Writer ▾



SweepMe! Driver Writer

Von SweepMe! GmbH

Parses PDF manuals to generate Python-based measurement device drivers.

I uploaded a PDF manual. Can you generate a Python...

Extract communication commands from...

Convert this device's SCPI commands into a...

How should I integrate this driver into my automatio...

Sende eine Nachricht an SweepMe! Driver Writer



ChatGPT kann Fehler machen. Überprüfe wichtige Informationen.



<https://chatgpt.com/g/g-679c9a16da048191be552b2246512077-sweepme-driver-writer>

- GitHub repo as examples
- Correct module import and class definition
- Defines variables and units
- Uses SweepMe!'s semantic driver functions
- Correct use of port communication
- Wrapped communication commands
- Continuous improvement through user feedback

Please write a driver for a Keithley 2400

Great choice! The Keithley 2400 SourceMeter uses SCPI commands for communication and supports GPIB, RS232, and USB (via VISA). I'll generate a SweepMe! driver based on the provided template.

Assumptions for the Driver

- Communication Protocol: GPIB, RS232, or USB via VISA. (I'll use `self.port.write()` and `self.port.read()`)
- Basic Operations: Setting voltage/current, measuring voltage/current.
- Units: Voltage (V), Current (A), Resistance (Ω).

SweepMe! Driver for Keithley 2400


```
python
from pysweepme.EmptyDeviceClass import EmptyDevice

class Device(EmptyDevice):
    def __init__(self) -> None:
        super().__init__()

        # Define measurable variables
        self.variables = ["Voltage", "Current", "Resistance"]
        self.units = ["V", "A", " $\Omega$ "]

    def connect(self) -> None:
        """Establishes connection with the Keithley 2400."""
        self.port.write("**RST") # Reset the device
        self.port.write("**IDN?") # Ask for ID
        idn = self.port.read()
```

Sende eine Nachricht an SweepMe! Driver Writer

+ 

Sequencer procedures

| Sequencer | |
|---|-------|
| Measurement tree | Value |
| <input checked="" type="checkbox"/> Loop1 | 4 |
| <input checked="" type="checkbox"/> MakeFile | ID |
| <input checked="" type="checkbox"/> SMU test Test Diode | |



ChatGPT



json representation

```
Code ▾
1 {
2   "child_items": [
3     {
4       "child_items": [
5         {
6           "child_items": [
7             {
8               "child_items": [],
9               "is_checked": true,
10              "is_expanded": true,
11              "module_details": {
12                "module_type": "SMU",
13                "Label": "SMU test",
14                "value": "Test Diode",
15                "Device": "SMU-Simulation_Diode",
16                "Channel": "",
17                "Port": "GPIB0::01::INSTR",
18                "SweepValue": "SweepEditor",
19                "SweepMode": "Voltage in V",
20                "RouteOut": "Rear",
21                "4wire": false,
22                "Compliance": "0.0001",
23                "SkipCompliance": false,
```

Conclusion

- **Generative AI** can create scripts, but cannot structure larger software.
- **Generative AI** is rapidly evolving, and own efforts quickly become obsolete.
- **Generative AI** can be a great enhancement to Low code tools with the right interfaces.
- **Generative AI** will have increasing impact on the use of software



Thank you

SweepMe! GmbH

contact@sweep-me.net

sweep-me.net

Bienertstr. 18, 01187 Dresden, Germany

Visit us on



<https://www.linkedin.com/company/sweepme/>