

## Universal software for the real-time control of sequential processing techniques

Julian Pilz, Martin Tazreiter, and Anna Maria Coclite

Citation: *Journal of Vacuum Science & Technology A* **37**, 063201 (2019); doi: 10.1116/1.5125052

View online: <https://doi.org/10.1116/1.5125052>

View Table of Contents: <https://avs.scitation.org/toc/jva/37/6>

Published by the [American Vacuum Society](http://www.avs.org)

---

### ARTICLES YOU MAY BE INTERESTED IN

[Atomic layer deposition of silicon-based dielectrics for semiconductor manufacturing: Current status and future outlook](#)

*Journal of Vacuum Science & Technology A* **37**, 060904 (2019); <https://doi.org/10.1116/1.5113631>

[Role of plasma properties in controlling crystallinity and phase in oxide films grown by plasma-enhanced atomic layer epitaxy](#)

*Journal of Vacuum Science & Technology A* **37**, 060909 (2019); <https://doi.org/10.1116/1.5128208>

[Paradigm shift in thin-film growth by magnetron sputtering: From gas-ion to metal-ion irradiation of the growing film](#)

*Journal of Vacuum Science & Technology A* **37**, 060801 (2019); <https://doi.org/10.1116/1.5121226>

[Status and prospects of plasma-assisted atomic layer deposition](#)

*Journal of Vacuum Science & Technology A* **37**, 030902 (2019); <https://doi.org/10.1116/1.5088582>


[Conformality in atomic layer deposition: Current status overview of analysis and modelling](#)

*Applied Physics Reviews* **6**, 021302 (2019); <https://doi.org/10.1063/1.5060967>

[Review of high-throughput approaches to search for piezoelectric nitrides](#)

*Journal of Vacuum Science & Technology A* **37**, 060803 (2019); <https://doi.org/10.1116/1.5125648>


---



## Instruments for Advanced Science


Contact Hiden Analytical for further details:  
[www.HidenAnalytical.com](http://www.HidenAnalytical.com)  
[info@hiden.co.uk](mailto:info@hiden.co.uk)

**CLICK TO VIEW** our product catalogue



### Gas Analysis

- dynamic measurement of reaction gas streams
- catalysis and thermal analysis
- molecular beam studies
- dissolved species probes
- fermentation, environmental and ecological studies



### Surface Science

- UHV-TPD
- SIMS
- end point detection in ion beam etch
- elemental imaging - surface mapping



### Plasma Diagnostics

- plasma source characterization
- etch and deposition process reaction kinetic studies
- analysis of neutral and radical species



### Vacuum Analysis

- partial pressure measurement and control of process gases
- reactive sputter process control
- vacuum diagnostics
- vacuum coating process monitoring



# Universal software for the real-time control of sequential processing techniques

Julian Pilz, Martin Tazreiter, and Anna Maria Coclite<sup>a)</sup>

*Institute of Solid State Physics, NAWI Graz, Graz University of Technology, 8010 Graz, Austria*

(Received 20 August 2019; accepted 19 September 2019; published 22 October 2019)

<https://doi.org/10.1116/1.5125052>

## I. INTRODUCTION

Sequential process control is essential for many thin film processing techniques such as atomic<sup>1</sup> and molecular layer deposition<sup>2</sup> (ALD and MLD), molecular beam epitaxy,<sup>3</sup> and atomic layer etching.<sup>4</sup> In these processes, valves or shutters are used to dose precursors in a vacuum reactor. The dosing has to happen in a fully automated and reproducible way and often has to be quite short (e.g., around ~10 ms for metalorganic precursors in ALD). Furthermore, novel hybrid materials require exact control of the composition and grading of their various components to obtain the desired properties.<sup>5</sup>

Implementations of control programs for thin film deposition have been reported in the literature. Selvaraj and Takoudis<sup>6</sup> have reported on a program that is able to deliver four different precursors as well as oxidant and purging gases in a sequential fashion. The number of supercycles and the number of layers of a material within one supercycle can be chosen in the program. This makes it possible to deposit general sequences like  $[A]v[B]w[C]x[D]Yz$ , where A, B, C, and D are the depositions of the individual materials; v, w, x, and y are the number of cycles of the specific materials; and z is the number of supercycles. The program has been implemented in LABVIEW. Piercy and Losego<sup>7</sup> have presented a more versatile program. Their tree-based approach offers the possibility to implement more complex structures such as graded laminates and sandwich structures. The program is not limited to the number of precursors and the order of complexity (the sequence above could be called a second order process as there is a loop over the supercycles and a loop over the individual materials) and has also been implemented in LABVIEW. In neither of the mentioned publications has a hardware integration (i.e., the transfer of sequence signals to valves, etc.) been presented.

In this publication, we present both a graphical user interface (GUI) for the development of universal recipes for sequential processes and the firmware for the execution of such a recipe on an external microcontroller. The GUI was developed in PYTHON, an *Arduino* Uno was used as the microcontroller, and its firmware was written in C/C++ using the *Arduino* IDE. In our opinion, this approach has several advantages compared to the ones in the literature:

- (1) Only open-source software and hardware are used.

- (2) The cost for the necessary hardware control (*Arduino* + relay/MOSFET module) is low.
- (3) The *Arduino* platform offers many possibilities to expand the functionalities of the program such as sensor inputs.
- (4) The firmware on an external microcontroller offers real-time control of the process that is not straightforward to implement for software running on a multi-thread operating system (*Windows*, *Mac OS*, *Linux*, ...).
- (5) The firmware works independently of the GUI. The microcontroller processes any recipe sent to it via a serial interface.
- (6) Recipes are created on a mere text basis. This offers a fast way to create, save, and load recipe files but could be more difficult to read than a more graphic representation.

In the following, the concept of the process control is introduced as well as the setup and the graphical user interface. Exemplary material structures are presented, which could be deposited using the program. Furthermore, the realization of the firmware is discussed. We believe that the presented software is useful for many research groups dealing with thin film processing techniques as it enables the development of versatile processes at low costs.

## II. CONCEPT, SETUP, AND GRAPHICAL USER INTERFACE

The main concept of the program relies on the definition of a recipe through a sequence that comprises sequence steps and their repetition. Sequence steps are depicted as successive capital letters (starting from A) and consist of a combination of three basic commands that are as follows:

- (1) open o[*Arduino port*].
- (2) close c[*Arduino port*].
- (3) wait w[*time in s*].

Box brackets followed by a positive integer number *n* define a repetition, whereas *n* is the number of repetitions of the enclosed sequence steps.

Examples of sequences:

- $[AB]2 = ABAB$
- $[ [AB]2C ]2 = ABABCABABC$

A full definition of a recipe including the commands of the sequence steps would, for example, be  $[AB]2; A = o2, w0.5, c2; B = o3, w2, c3$ . Here ports 2 and 3 are opened sequentially for 0.5 and 2 s, respectively, twice. Throughout the paper, an *Arduino* port being open or closed refers to a

Note: This paper is part of the 2020 Special Topic Collection on Atomic Layer Deposition (ALD).

<sup>a)</sup>Electronic mail: [anna.coclite@tugraz.at](mailto:anna.coclite@tugraz.at)



voltage of +5 V or 0 V at the port, respectively. Depending on the connection to the port (see Fig. 1), this leads to the opening or closing of, e.g., a valve.

Figure 1 shows a sketch of the setup. The GUI runs on a PC and allows the user to develop, save, and load recipes. During the modification, the total duration of the recipe is automatically calculated. After developing a recipe with the right syntax, the user can execute the recipe. The software will then automatically connect to the Arduino and facilitate a serial connection. The sequence to be executed is sent via this serial connection to the Arduino, and the Arduino confirms the receipt of the sequence, decodes it (i.e., breaks up all the brackets via a recursive function), executes it [i.e., opens (+5 V) or closes (0 V) its ports], and sends a message to the PC upon starting a new sequence step. The firmware of the Arduino works independently of the GUI, i.e., any software can make a serial connection to the Arduino and send a recipe to be executed. The output ports of the Arduino are connected to any kind of (fast) switches such as relays or MOSFETs that connect or disconnect the loads to a power supply. Loads for deposition processes include ALD and general gas valves, plasma generators, and gate valves for *in situ* spectroscopic ellipsometry.<sup>8</sup>

A flowchart with the main process steps upon execution of a recipe is shown in Fig. 2. The user starts by editing or opening a recipe. The total duration is automatically calculated and displayed. The recipe is checked for syntax errors (such as unclosed brackets), and upon execution, a serial connection to the Arduino is opened. The recipe is sent to the Arduino and the Arduino confirms the start of the recipe. Every time a new sequence step is started, the letter corresponding to this sequence step is sent to the PC, and the

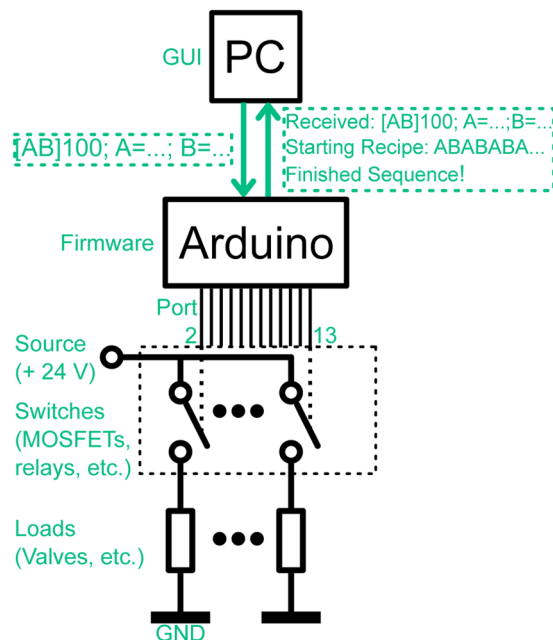


FIG. 1. Sketch of the overall setup. Communication between the PC and the Arduino is facilitated with the GUI. The digital outputs of the Arduino are connected to switches, which switch the corresponding loads according to the recipe.

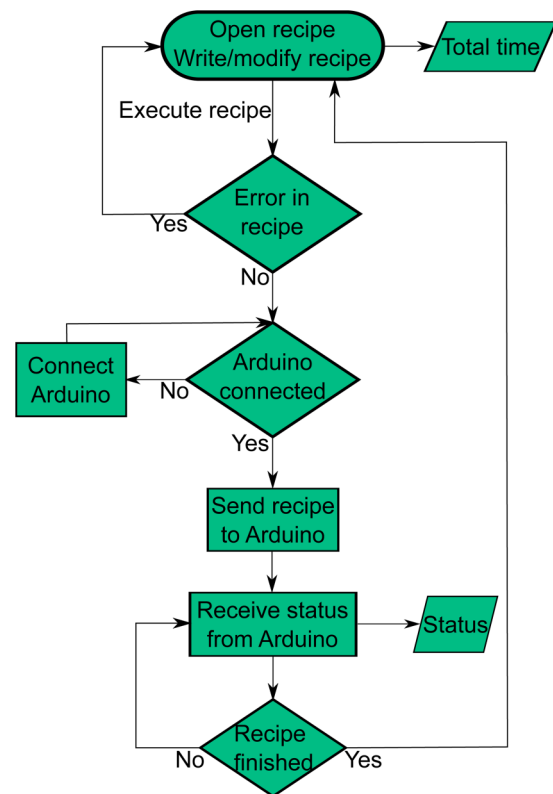


FIG. 2. Flowchart of the main steps in the recipe mode of the GUI.

current status of the recipe is updated. Upon completion of the recipe, the Arduino sends a message to PC, and the user can start a new recipe. The serial connection is terminated when the GUI is closed.

Figure 3 shows a snapshot of the GUI in the recipe mode, which was developed in PYTHON. (1) It marks the status of the serial connection to the Arduino. It will turn from disconnected to connected upon starting a recipe (and change its background color, accordingly). (2) It marks two textboxes including the strings sent to and received from the Arduino. (3) It marks three buttons for opening, saving, and executing a recipe. (4) It marks two textboxes to modify a recipe. In the upper one, the sequence is defined. In the lower textbox, the sequence steps are defined with the commands open (o[Arduino port]), close (c[Arduino port]), and wait (w[time in s]). The commands are separated by commas, and every sequence step has to be defined in a separate line. Comments can be written with #[comment]. (5) It marks the total time of the recipe, which is automatically calculated from the inputs in (4). When the recipe is started, the remaining time and the current status below the executed recipe are shown. The current sequence step is highlighted, and the counter besides a bracket shows how many times the sequence within this bracket has already been executed. Finally, at the bottom of the GUI at (6), a status bar shows information for the user such as errors in the recipe or connection problems to the Arduino.

Another feature of the program is the analysis mode, which can be selected in the tab bar. A screenshot is shown in Fig. 4. It consists of buttons referring to the Arduino

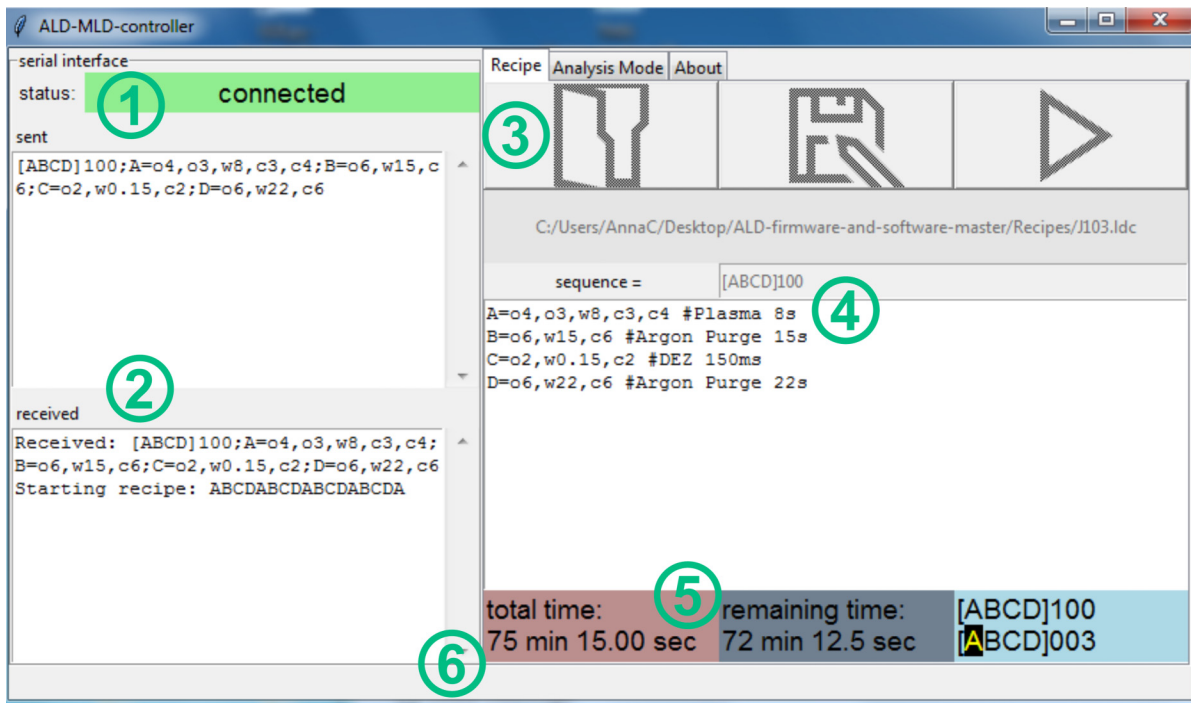


Fig. 3. Overview of the GUI in the recipe mode. Numbered labels show (1) status of the serial connection to the Arduino (connected/disconnected), (2) commands sent to and received from the Arduino via a serial interface, (3) buttons to load, save, and execute a recipe file, (4) textboxes to define the recipe sequence and sequence components, (5) total and remaining time of the running recipe and current status within the recipe, and (6) a status bar.

ports. By clicking on a button, the specific Arduino port will be inverted, thus turning from open to closed or from closed to open. The serial connection to the Arduino is automatically facilitated upon clicking on a button. The buttons are

deactivated when a recipe is started in the recipe mode to prevent communication issues.

After introducing the concept of the program, some exemplary processes are shown, which could be easily

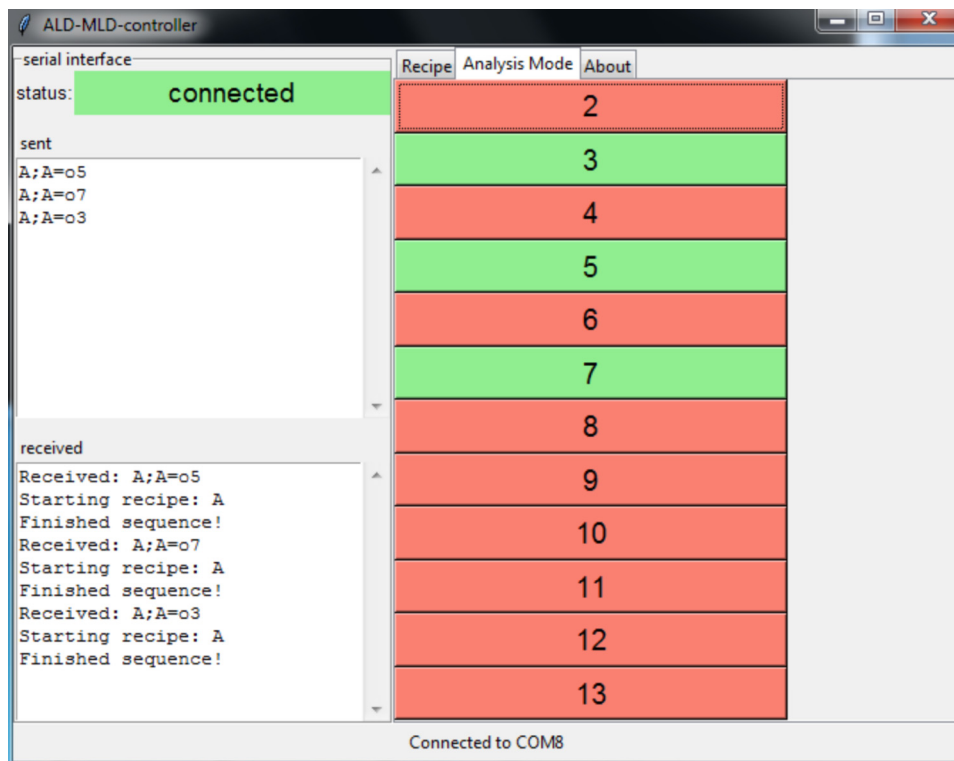


Fig. 4. Overview of the GUI in the analysis mode. Number of the buttons refers to the output channels of the Arduino and background color to the state of the channel. During the execution of a recipe, the buttons are disabled.



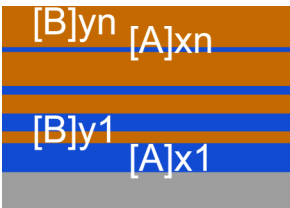
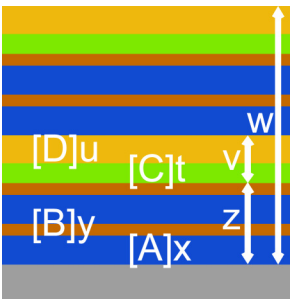
implemented. The program has already been successfully used for plasma-enhanced atomic layer deposition of ZnO<sup>9-11</sup> and molecular layer deposition of zirconium. Table I shows possible processes that could also benefit from the versatile opportunities of the program. The underlying syntax for the definition of a sequence makes it possible to implement complex processes such as multilevel laminates in a short and readable way. The table was inspired by the publication of Piercy and Losego.<sup>7</sup>

Furthermore, an exemplary recipe is given for 50 ALD cycles of Al<sub>2</sub>O<sub>3</sub> applying trimethylaluminum (TMA) and water, and argon as the purging gas. Let us also assume that for every five cycles of the process, the thickness is measured with *in situ* spectroscopic ellipsometry<sup>8</sup> (by opening

two gate valves in front of the source and the detector). We assume that the valves controlled by the Arduino outputs are connected in the following way: port 2/ALD valve TMA, port 3/ALD valve H<sub>2</sub>O, port 4/ALD valve Ar, port 5, and port 6/gate valves ellipsometry. The recipe then could look like that (assuming dose times of 0.01/10/0.1/20 s for TMA/purge/H<sub>2</sub>O/purge and measurement time of ellipsometry 10 s):

```
sequence = [ [ABCD] 5E] 10
A = o2, w0.01, c2 #TMA dose
B = o4, w10, c4 #Ar purge
C = o3, w0.1, c3 #H2O dose
D = o4, w20, c4 #Ar purge
E = o5, o6, w10, c5, c6 #ellipsometry
                                measurement
```

TABLE I. Examples of sequences for different material structures.

Description	Sequence and sketch	Experimental example
Single material	<ol style="list-style-type: none"> <li>1. [A]<sub>x</sub></li> <li>2. [AB]<sub>x</sub></li> <li>3. [ABCD]<sub>x</sub></li> </ol> <p>Depending on preferences, either of the three forms can be chosen. For 1, all dosing step and purge steps have to be included in the definition of A; for 2, the half cycles are splitted into two definitions; and for 3, all dosing and purging steps are separately defined. For better readability, form 1 is chosen for the following examples.</p> 	<p>Al<sub>2</sub>O<sub>3</sub><sup>13</sup></p> <ol style="list-style-type: none"> <li>1. A = TMA dose + purge + water dose + purge</li> <li>2. A = TMA dose + purge B = water dose + purge</li> <li>3. A = TMA dose B = purge C = water dose D = purge</li> </ol>
Basic laminate of two materials	<p>[[A]<sub>x</sub>[B]<sub>y</sub>]<sub>z</sub></p> 	<p>Al doped ZnO<sup>14</sup></p> <p>A = ZnO deposition B = Al<sub>2</sub>O<sub>3</sub> deposition</p>
Graded laminate	<p>[[A]<sub>x1</sub>[B]<sub>y1</sub>]<sub>z1</sub> [[A]<sub>x2</sub>[B]<sub>y2</sub>]<sub>z2</sub> ... [[A]<sub>xn</sub>[B]<sub>yn</sub>]<sub>zn</sub></p> 	<p>Graded laminates of Al<sub>2</sub>O<sub>3</sub> and Ta<sub>2</sub>O<sub>5</sub><sup>15</sup></p> <p>A = Al<sub>2</sub>O<sub>3</sub> deposition B = Ta<sub>2</sub>O<sub>5</sub> deposition</p>
Multilevel laminates	<p>[[[A]<sub>x</sub>[B]<sub>y</sub>]<sub>z</sub> [[C]<sub>t</sub>[D]<sub>u</sub>]<sub>v</sub>]<sub>w</sub></p> 	<p>(La<sub>x</sub>Sr<sub>1-x</sub>)MnO<sub>3</sub><sup>16</sup></p>

### III. ARDUINO FIRMWARE

After introducing the concept and the implementation of the GUI, the implementation of the Arduino firmware is introduced. This is the part where the actual process control is facilitated. After a serial connection is facilitated with the Arduino, it waits to receive an input. Upon receiving a recipe string, it checks for the correctness of the sequence and the definition of the sequence steps. If everything is correct, an array consisting of the commands of the sequence steps is created and the evaluation of the sequence started. As the sequence should allow an arbitrary level of complexity (i.e., the number of loops over loops, defined in the syntax by box brackets `[ ]`), the evaluation of the sequence is implemented in a recursive fashion. The complexity of the sequence is hereby only limited by the memory of the microcontroller (which can be extended by switching to, e.g., an Arduino model with enhanced memory). A sketch of the evaluation is shown in Fig. 5. There is a for-loop that runs over all characters within the sequence string. If the character is an opening bracket “[,” the program looks for the corresponding closing bracket “]” and the number following it (i.e., the number of iterations of the subsequence). A for-loop runs over the number of iterations and calls the recursive evaluation function again, now with the subsequence string (i.e., the string between the opening “[” and closing bracket “]”) as an argument. If the character currently investigated in the for-loop over the sequence string is a top level sequence step letter, i.e., not within brackets, the sequence step is executed and the character is sent via the serial interface.

Considering the exemplary sequence from above, `[ [ABCD] 5E ] 10`, the first character is “[,” therefore, the

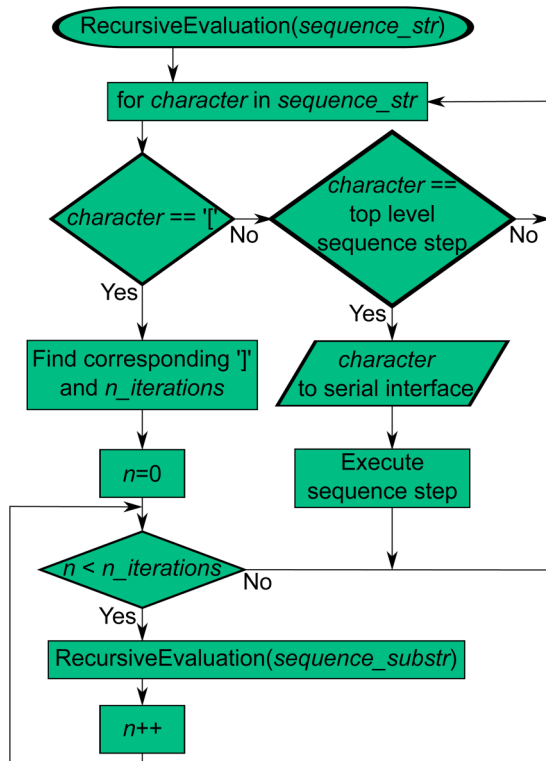


FIG. 5. Flowchart of the recursive evaluation of a sequence.

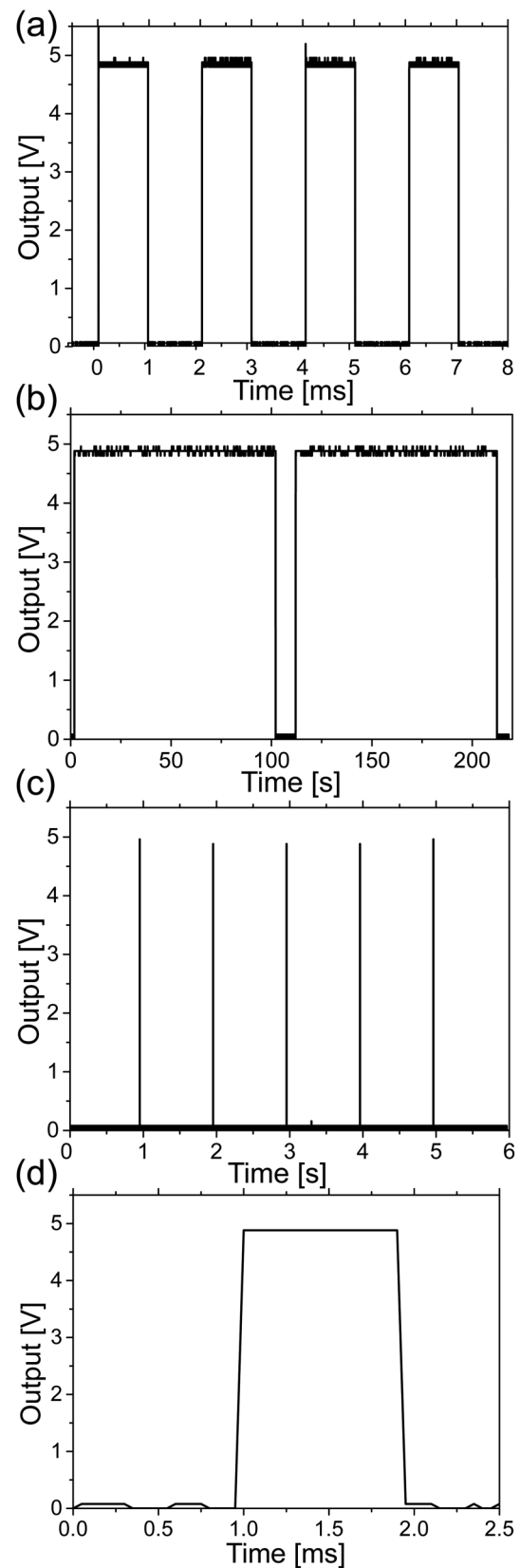


FIG. 6. Output voltage of an Arduino channel upon receiving different sequences from the GUI. The sent sequences are (a) channel 1 ms open, 1 ms closed; (b) channel 100 s open, 10 s closed; and (c) channel 1 ms open, 1 s closed. (d) Shows a magnification of the 1 ms pulse in (c). Note that for performing several thousand repetitions of the sequence in (a), sending of the sequence step via the serial connection had to be deactivated in the firmware.

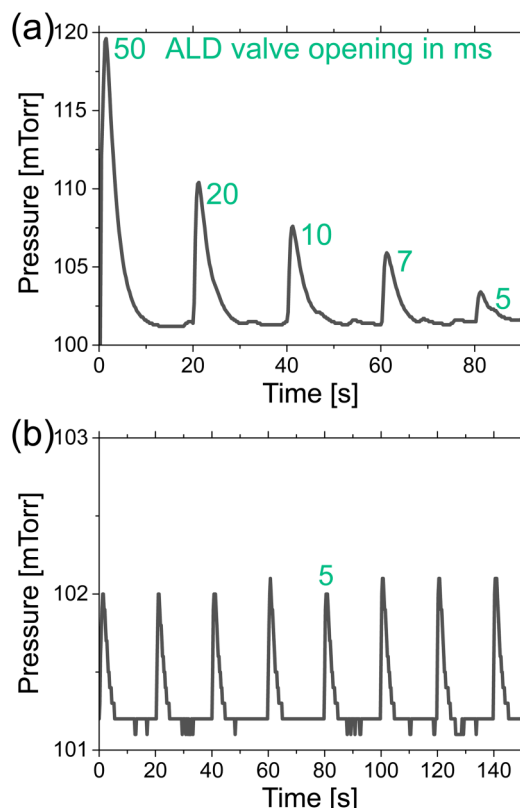


FIG. 7. Pressure increase during dosing of DEZ every 20 s with (a) different ALD valve opening times and (b) 5 ms ALD valve opening time. A constant flow of 20 sccm Ar was present for all doses and constituted the baseline pressure of around 101 mTorr. 5 ms was the shortest possible opening time; for shorter signals, the ALD valve did not open and no pressure increase was observable. Considering a readout period of the pressure of 0.25 s, the differences in peak heights in (b) are negligible.

program, currently at recursion depth 0, calls ten times the function with the substring “[ABCD]5E.” At recursion depth 1, the first character is again “[” and the function is called five times with the substring “ABCD.” Now at recursion depth 2, all characters are top level sequence steps that are executed. After these five repetitions, the program continues at recursion depth 1 at the character “A” of “[ABCD]5E,” which is within brackets and, therefore, is not executed. The next top level character to be executed is “E,” after which the program continues at recursion depth 0 at the character “A” of “[ [ABCD]5E]10,” which again is not top level.

This execution of the recipe with the microcontroller offers on the one hand independency from the software sending the recipe string via a serial connection (any rudimentary terminal software could be used to send recipes, and thus it is operating system independent). On the other hand, it allows quasi real-time execution of the sequence steps, often needed in the fast switching of valves in ALD processes. Figure 6 shows the switching of an output channel of the Arduino in different time scales occurring in sequential processes. In Figs. 6(a) and 6(b), a repetition of 1 ms and 100 s pulses is shown, respectively, demonstrating the possibility of the exact timing of very short and very long pulses. Figure 6(c), furthermore, shows 1 ms pulses with a 1 s delay in-between, and Fig 6(d) shows a magnification of the 1 ms pulse.

Figure 7 shows the application of short pulses for switching an ALD valve (Swagelok ALD3) to introduce diethylzinc (DEZ) into a vacuum reactor. Figure 7(a) shows the pressure increase in the reactor (measured by an MKS Baratron 262B pressure gauge and readout by an external software) with different valve opening times, offering the possibility to introduce discriminable low doses of DEZ. Figure 7(b) shows that the lowest dose possible with the setup (5 ms valve opening) can be introduced repetitively, without fluctuations in the dose.

#### IV. SUMMARY AND CONCLUSIONS

In this note, a GUI and firmware implementation of a control software for sequential processing techniques is presented. The GUI allows to create, modify, and save recipes, facilitates the communication with the processing hardware, and shows the current status of a running recipe. Recipes are based on the definition of a sequence including the repetition of sequence steps that are defined with combinations of three basic commands. The processing of recipes with the firmware in a recursive fashion allows an almost indefinite level of complexity of the recipes (only limited by the microcontroller memory) and a quasi real-time processing of such. Furthermore, only open-source software and hardware are used in this implementation. Thus, the versatility and low costs of our approach allow scientists to expand their research on creating complex material structures for a wide range of applications.

#### ACKNOWLEDGMENTS

This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation program (Grant Agreement No. 715403). The authors thank Peter Luidolt for his valuable inputs and suggestions.

- <sup>1</sup>S. M. George, *Chem. Rev.* **110**, 111 (2010).
- <sup>2</sup>P. Sundberg and M. Karppinen, *Beilstein J. Nanotechnol.* **5**, 1104 (2014).
- <sup>3</sup>A. Y. Cho and J. R. Arthur, *Prog. Solid State Chem.* **10**, 157 (1975).
- <sup>4</sup>G. S. Oehrlein, D. Metzler, and C. Li, *ECS J. Solid State Sci. Technol.* **4**, N5041 (2015).
- <sup>5</sup>K. Gregorczyk and M. Knez, *Prog. Mater. Sci.* **75**, 1 (2016).
- <sup>6</sup>S. K. Selvaraj and C. G. Takoudis, *J. Vac. Sci. Technol. A* **33**, 013201 (2015).
- <sup>7</sup>B. D. Piercy and M. D. Losego, *J. Vac. Sci. Technol. B* **33**, 043201 (2015).
- <sup>8</sup>E. Langereis, S. B. S. Heil, H. C. M. Knoops, W. Keuning, M. C. M. van de Sanden, and W. M. M. Kessels, *J. Phys. D Appl. Phys.* **42**, 073001 (2009).
- <sup>9</sup>A. Perrotta, J. Pilz, A. Milella, and A. M. Coclite, *Appl. Surf. Sci.* **483**, 10 (2019).
- <sup>10</sup>A. Perrotta, J. Pilz, S. Pachmajer, A. Milella, and A. M. Coclite, *Beilstein J. Nanotechnol.* **10**, 746 (2019).
- <sup>11</sup>J. Pilz, A. Perrotta, G. Leising, and A. M. Coclite, “ZnO thin films grown by plasma-enhanced atomic layer deposition: Material properties within and outside the ‘atomic layer deposition window,’” *Phys. Status Solidi* (published online).
- <sup>12</sup>A. Perrotta, R. Berger, F. Muralter, and A. M. Coclite, *Dalton Trans.* **48**, 14178 (2019).
- <sup>13</sup>M. D. Groner, F. H. Fabreguette, J. W. Elam, and S. M. George, *Chem. Mater.* **16**, 639 (2004).
- <sup>14</sup>P. Banerjee, W.-J. Lee, K.-R. Bae, S. B. Lee, and G. W. Rubloff, *J. Appl. Phys.* **108**, 043504 (2010).
- <sup>15</sup>A. Szeghalmi, S. Senz, M. Bretschneider, U. Gösele, and M. Knez, *Appl. Phys. Lett.* **94**, 133111 (2009).
- <sup>16</sup>T. P. Holme, C. Lee, and F. B. Prinz, *Solid State Ion.* **179**, 1540 (2008).