EMIoT Lab 3 Report:
Energy storage, generation and conversion

Yuqi Li s336721   Xuxin Zhou s337564

Group 6

February 26, 2026

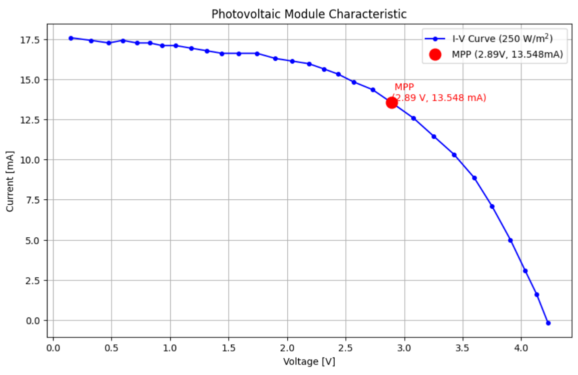
Contents

1 Introduction

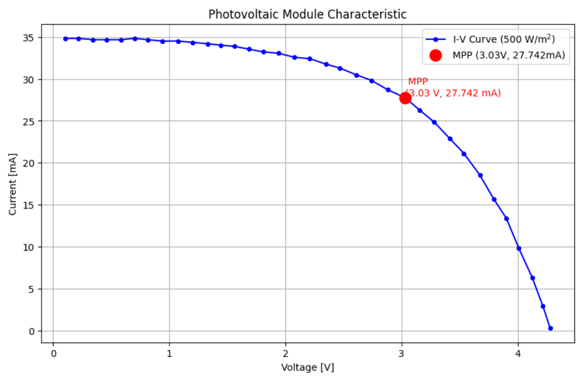
2 Part 1: Model of the photovoltaic module

2.1 Extract Data Using PlotDigitizer

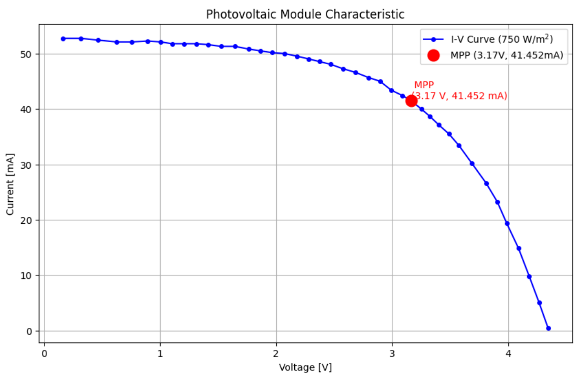
We use the PlotDigitizer and load the required figure of curves(MP3-37 in PVdatasheet.pdf), by pointing on the curves, finally we got the fitted relationship and exported them on different txt files (Different light intensities correspond to different txt files.).
For these exported data(**250w.txt, 500w.txt, 750w.txt, 1000w.txt**), we implement the Python script and perform fitting and reconstruction on these discrete points, and find their corresponding maximum power point. The analyzed result is shown below. and we fill them into the **inc/config_pv.h**



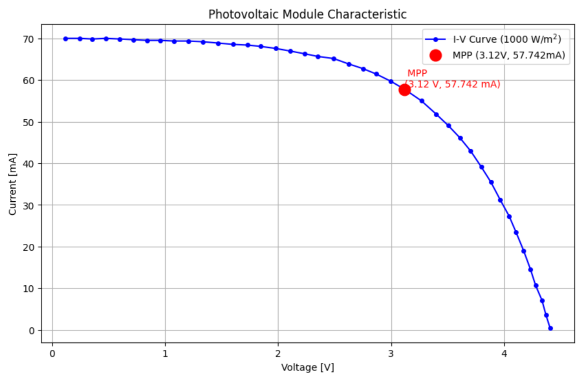
(a) 250w



(b) 500w



(c) 750w



(d) 1000w

Figure 1: Results from different Irradiance

Table 1: Photovoltaic Module Characteristics at Different Irradiance Levels

Irradiance (W/m^2)	V_{MPP} (V)	I_{MPP} (mA)	P_{MPP} (mW)
250	2.89	13.548	39.15
500	3.03	27.742	84.06
750	3.17	41.452	131.40
1000	3.12	57.742	180.16

Analysis: The results show that the MPP current increases linearly with irradiance. The MPP voltage rises up to 750 W/m^2 but drops slightly at 1000 W/m^2. It might be due to manual measurement errors.

2.2 Irradiance Trace Visualization

We implement a Python script to read the **gmonths.mat** file to make the Irradiance situations visible. The result is shown below(here we only display the first 6 days due to layout constraints). The script is attached in **lab3.ipynb**.

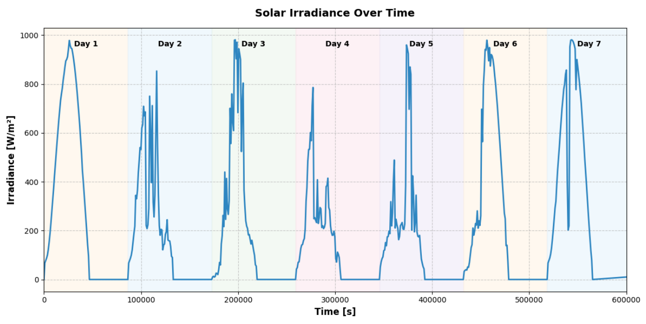


Figure 2: Irradiance Trace Visualization

3 Part 2: Model of DC-DC converters and battery

In this section, we also use the PlotDigitizer to extract the points, then fit and reconstruct them using the Python script.

3.1 DC-DC Converter Datasheet Digitalization for PV-module

For curve in **PV_DCDCconverter.pdf**, we implement a python script to reconstruction the exported txt file(**trace_conv_pv.txt**) by PlotDigitzer and generate the Sorted data(**Since the exported data in the original txt file may be disorganized, it is necessary to sort the raw data in ascending order to ensure that the correct data is entered into the lookup tables of the SystemC simulator.**) for two required arrays to put into the **inc/config_converter_pv.h**. The reconstructed curve and data is shown below.

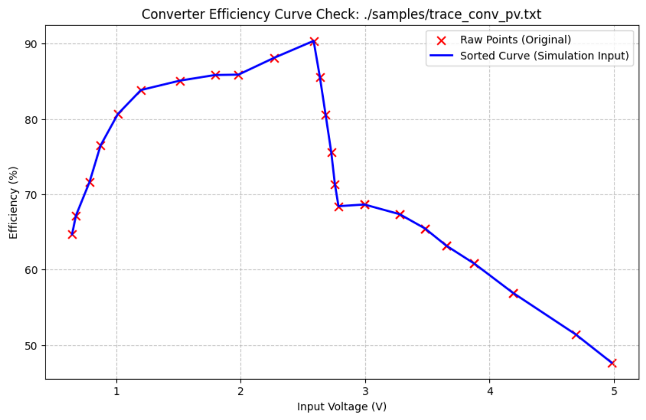


Figure 3: Reconstructed curve for DC-DC converter

Listing 1: Configuration parameters for PV Converter (inc/config_converter_pv.h)

```
#define SIZE_CONV_PV 24
```

```
static const double V_CONV_PV[SIZE_CONV_PV] = {
    0.6459, 0.6773, 0.7864, 0.8749, 1.0136, 1.1997,
    1.5119, 1.7951, 1.9824, 2.2697, 2.5861, 2.6361,
    2.6819, 2.7277, 2.7569, 2.7860, 2.9937, 3.2765,
    3.4801, 3.6527, 3.8761, 4.1871, 4.6905, 4.9790
};

static const double ETA_CONV_PV[SIZE_CONV_PV] = {
    64.6824, 67.1739, 71.6304, 76.4808, 80.6522, 83.8485,
    85.0931, 85.8398, 85.8896, 88.1298, 90.3700, 85.5909,
    80.6127, 75.5847, 71.3532, 68.4161, 68.6417, 67.3568,
    65.4633, 63.1640, 60.7971, 56.9086, 51.3971, 47.6439
};
```

3.2 DC-DC Converter Datasheet Digitalization for Battery

Using the same method as previously, we reconstruct the exported txt file **trace_conv_dcdc.txt**, and the curve we selected for Digitalization in **BATT_DCDCconv.pdf** is the Vin = 2.4V one. But the original data of **x axis** is in logarithmic scale, thus we point more explicit points in PlotDigitizer in log mode and reconstruct it. and we also script to export the sorted data, and set these parameters in **inc/config_converter_battery.h**, The result is shown below.

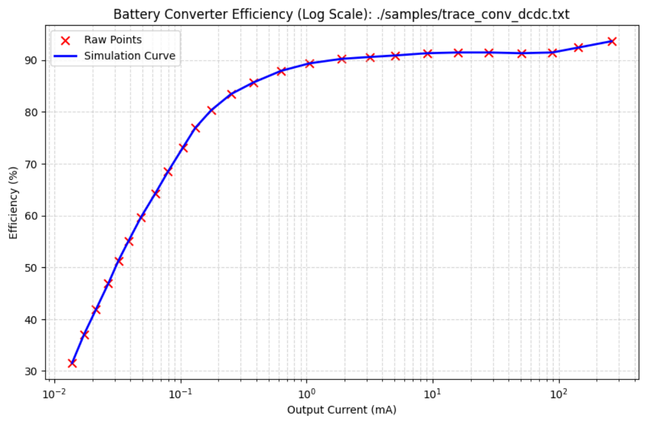
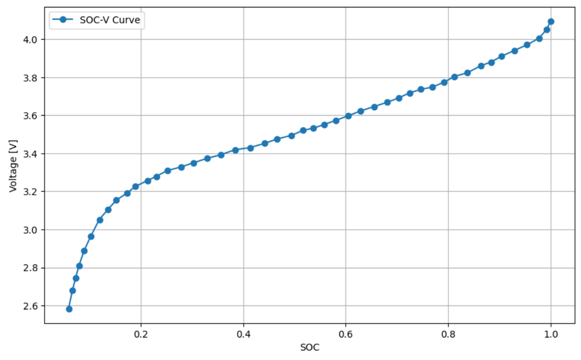


Figure 4: DC-DC Converter for battery

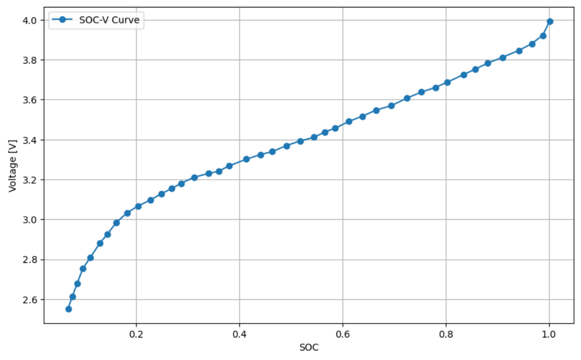
3.3 Battery Modeling

3.3.1 Extract data from the figure

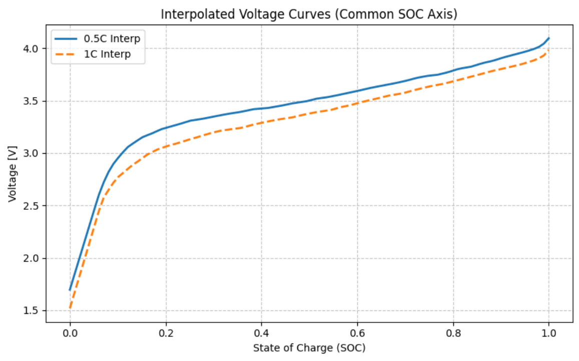
By pointing to the data of discharge characteristic of **battery.pdf**, and normalizing the capacity at different discharge rates to obtain the relationship between SOC and voltage. We select the curve of discharge rate in 0.5c and 1c, reconstruct them, the resluts is shown in Figure 5, and we interpolate the two data sets.



(a) 0.5c discharge relationship between Voltage and SOC



(b) 1c discharge relationship between Voltage and SOC



(c) The interpolated result

Figure 5: Extracted data from **battery.pdf**

3.3.2 Derive Voc and R

We implemented the script(in **.ipynb** file) to read the extracted data from the txt file(0.5c.txt and 1c.txt), and to do the data alignment and Interpolation as the previous step, then, according to the formula, to solve the equations.

$$V_{load} = V_{OC} - I \cdot R_{series} \tag{1}$$

$$R_{series}(SOC) = \frac{V_{0.5C}(SOC) - V_{1C}(SOC)}{I_{1C} - I_{0.5C}} \tag{2}$$

Here, I_{1C} and $I_{0.5C}$ are constants, equal to 3200 mA and 1600 mA. The values of $V_{0.5C}(SOC)$ and $V_{1C}(SOC)$ vary, corresponding to the normalized discrete voltage points in the SOC curve. And we got the result for the R-series and the changing Voc value. The result is shown in Figure 6.

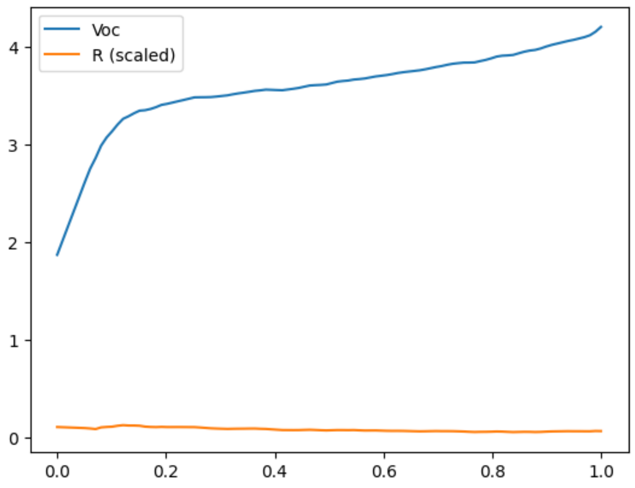


Figure 6: Voc and R

3.3.3 Curve Fitting

To model the non-linear behavior of the battery, a 4th-degree polynomial function was chosen. The fitting process was performed using the **scipy.optimize.curve_fit** library in Python, which utilizes the non-linear least squares method to minimize the residuals between the calculated discrete data and the polynomial curve. The target function is defined as the equation below, and the data put into the **curve_fit** function is **v_open_circuit** and **r_series** calculated in the previous step, and also scaling in the SOC range.

$$f(SOC) = a \cdot SOC^4 + b \cdot SOC^3 + c \cdot SOC^2 + d \cdot SOC + e \tag{3}$$

Where $SOC \in [0, 1]$.
Results: The obtained coefficients are presented below:

Table 2: Battery Model Parameter Fitting Results (4th-Degree Polynomial Coefficients)

Parameter	**a** (x^4)	**b** (x^3)	**c** (x^2)	**d** (x^1)	**e (const)**
V_{OC}(SOC)	-18.6484	44.6949	-36.6004	12.5585	2.0546
R(SOC)	-4.55×10^{-4}	1.16×10^{-3}	-9.18×10^{-4}	1.77×10^{-4}	1.03×10^{-4}

We also perform the check for the fitted curve and original data, the result is shown below, and fill the calculated value in **src/battery_voc.cpp**.

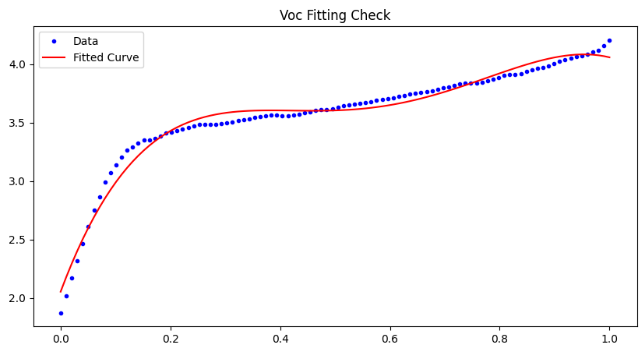


Figure 7: Check for Voc

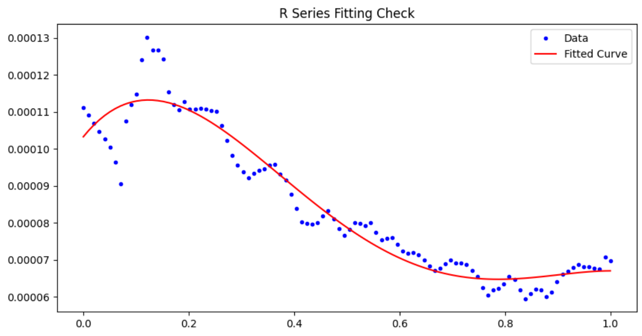


Figure 8: Check for Rseries

4 Part 3: Load modeling and scheduling

4.1 First analysis of the Lab:

We implement a script to analyze the produced simulation log (sim_trace.txt).
First, we analyze the load's power, photovoltaic power, battery power, and battery SOC's variation over time. The overall result is shown in Figure 9. We select the first seven days to do the analysis of the main feature of the simulation, because the whole simulation trace is too long and the first 7 days are enough to do the analysis.

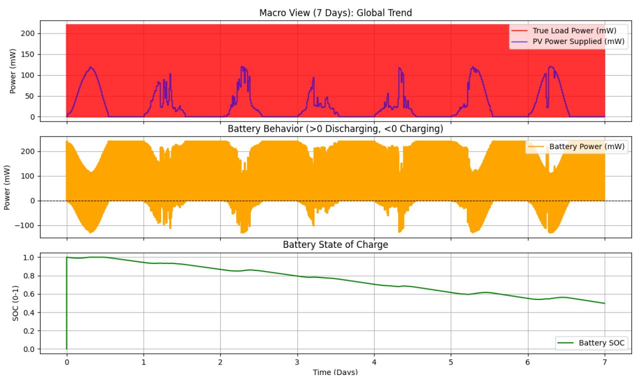


Figure 9: The overall feature for the simulation, including the PV power, Load power, Battery power, and the SOC.

And the overall feature is not quite clear for some specific requirements(the content and the curves in the figure are too dense and there is overlapping.). Thus, we also zoom in on some specific time durations to find the detailed result. The zoom in feature is shown in Figure 10.

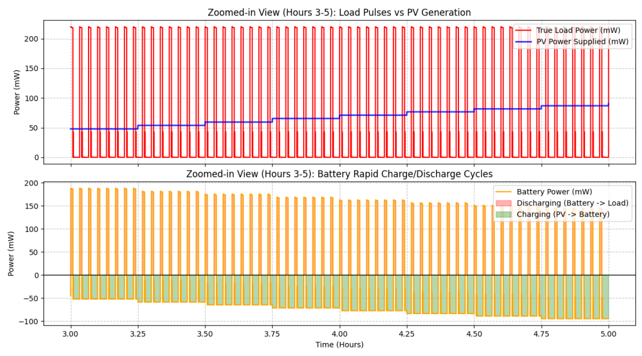


Figure 10: The detailed feature of simulation for the first hour 3 to hour 5.

Analysis:

1. As simulation time progresses, the battery SOC value gradually decreases, indicating that throughout the system's operation, the total energy harvested from PV is less than the energy consumed by the load, resulting in the decline of SOC.

2. Load power consumption exhibits periodic variations, because under parallel scheduling, multiple sensors are awakened simultaneously to perform tasks. The currents from these multiple loads combine to form a large power pulse. After completing their measurement tasks, the sensors enter sleep mode, causing the total load to drop sharply. And as defined in parallel.json, the load activation time and the time for in **on state** is a constant value.

3. The supplied PV power is slightly changing during the time variation. and its Waveform shape(in Figure 9) corresponds to the previous Irradiance Trace Visualization Figure 2, which shows the simulation is running correctly.

4. In Figure 10, the original load power is over 200mW, and in the positive part of the battery curve, it is less than 200mW, because of the Compensation from PV power. As the PV power slightly increased during the time variation, the battery power is correspondingly reduced, showing the accuracy of simulation result.

5. In Figure 10, the green part showing the Net Revenue for battery power, because the these green area corresponding to the off state for the load(sensors). In this stage, the Load power is zero. Thus, the battery power in the green area has the same numerical value as PV power.

Secondly, according to the value of sim_trace.txt, we calculate the efficiency of PV converter and Battery converter, and determine the distribution of them. The result is show in Figure 11. Here, we analysis the discharge efficiency first.

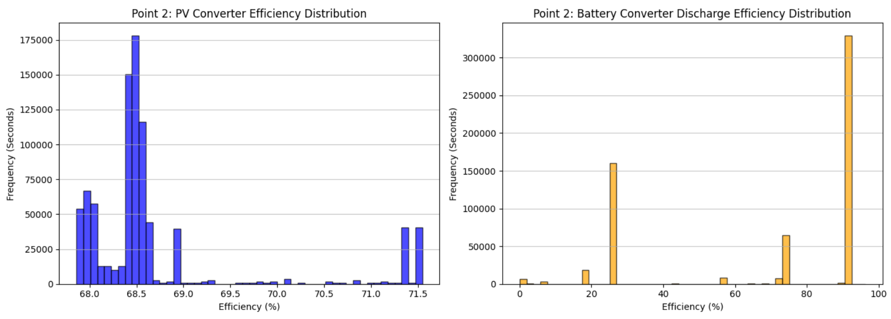


Figure 11: Discharge efficiency Distribution of converters

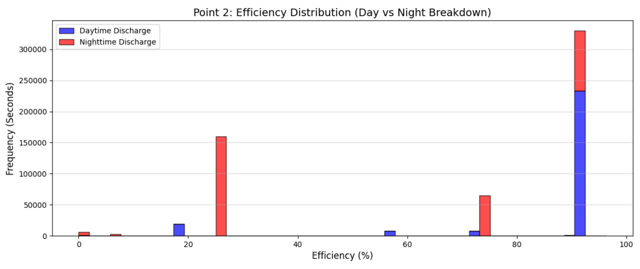


Figure 12: Detailed discharge efficiency Distribution for day and night

Table 3: Battery Converter Discharge Efficiency Statistics

Metric	**Value**	**Description**
Total Discharge Time	601,666 s	Total duration
- Night Discharge Time	331,677 s	Mostly in Low Efficiency Zone
- Day Discharge Time	269,989 s	Mostly in High Efficiency Zone
Average Efficiency		
- Night Average	54.45%	**Bad Area**
- Day Average	84.33%	**Good Area**
Low Efficiency (less 40%)		
- Duration	188,253 s	Time spent in wasteful state
- Ratio	31.29%	Percentage of discharge time

Analysis:

1. The most of the time, the efficiency of PV converter is around 68%, because the Vmpp of PV panel is from 2.89 V to 3.12 V (250 W/m^2 to 1000 W/m^2), and the corresponding efficiency is around from 68% to 72%(according to the efficency curve from datasheet), and according to the Irradiance Trace Visualization, most of Irradiance is larger than 250w/m2(for this sitiuation, Vmpp is mostly aroud 3.1V), thus the efficiency distribution is mostly around 68%.

2. For the efficiency distribution of Battery converter, the converter working on high efficiency(90%) or on low efficiency(less than 40%). That's because the load current, as we mentioned in previous, the load power is periodic duty cycled, when the loads in activation state, the power is over 200mW(the load current is high), and corresponding efficiency in battery converter datasheet is around 90%, and when the loads is on sleep state, the load current is almost close to zero, thus lead to the low efficiency for converter.

3. The datailed distribution of battery converter is because during the day time(system supported by PV), when the loads activated, the current will fall in high efficiency range, and when it is in sleep mode, the power from PV will Compensate for this portion of the current, battery converter didn't need to work at this stage. But during the night time, only when the loads activated will fall in high efficiency range, and when the loads in sleep mode, there is no PV power to Compensate the slightly sleep current, thus the converter will in low efficiency range, and according to the Figure 10 and parallel.json the sleep time is higher than activation time, thus in night time the more time will spend working in low efficiency.

Next, we analyzed the distribution of battery charging efficiency. The result is shown below:

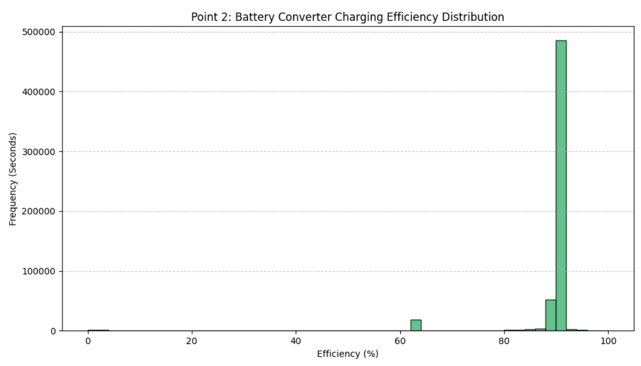


Figure 13: Battery Converter Charging Efficiency Distribution

Table 4: Battery Converter Charging Efficiency Statistics

Metric	**Value**	**Description**
Total Charge Time	584,403 s	Total duration of battery charging
Average Efficiency	88.45%	Overall performance in charging state
Low Efficiency (less 40%)		
- Duration	8,887 s	Time spent in wasteful state
- Ratio	1.52%	Percentage of total charge time

Analysis:

The distribution chart indicates that the majority of discharge time occurs within the high-efficiency range. According to the datasheet, the charging current typically falls within this high-efficiency range of approximately 1-100mA. During daytime hours, when loads are in sleep mode, only a minimal current is required to maintain their sleep state. The remaining current charges the battery. The current generated by the PV system also falls precisely within the converter's high-efficiency range.
The distribution chart also shows a lower efficiency range. This occurs because, based on the distribution of light intensity, there are periods of extremely low light intensity—potentially representing overcast or rainy conditions. Under such circumstances, the current supplied by the PV system barely meets the load demand, leaving only a minimal surplus to feed into the battery. This surplus falls precisely within the lower efficiency range of the battery converter.
Third, we implement the script to filter the data by find the **i_tot** larger than zero to determine the battery discharging state, and the rest is the charging state for battery. The result is shown in figure 13.

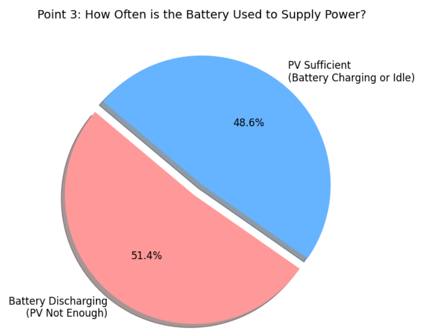


Figure 14: Distribution for charging and discharging state for battery

Analysis:

The discharging state proportion for battery is larger than charging state, and according to the previous analysis, the current for activation of parallel loads is far greater than the PV module can provide during the day time, and when the loads sleep, the charged power is also lower than the power consumed in activation state. In night timeit will lead to pure power consume for battery. This confirms why the battery SOC is decreased during time variation.

4.2 Second analysis of the Lab:

We developed the new configuration file named "sequential.json" in sim_setting. and re-run the simulation, the simulation result is recorded in **se_sim_trace.txt**. and we do the analysis for the SOC for battery to find the difference of result between original parallel configuration. The result is shown in Figure 14 and Figure 15:

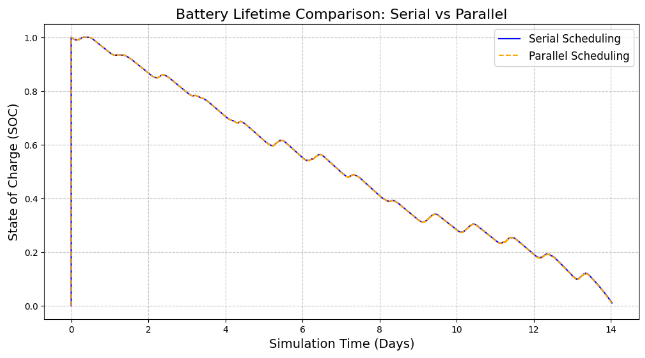


Figure 15: General Trends of serial config and parallel config, the time servived for serial is 14.03 days, and for parallel is 14.02 days.

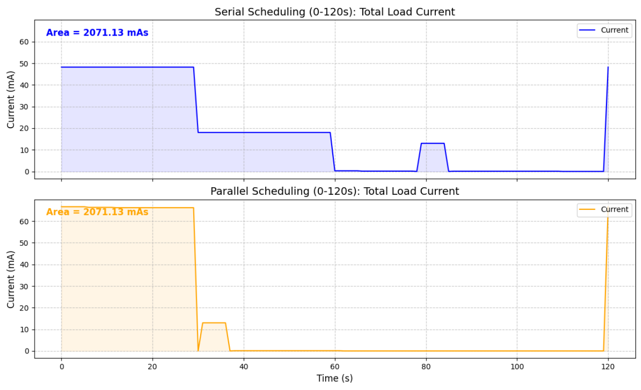


Figure 16: The detailed load curve for first 120 seconds

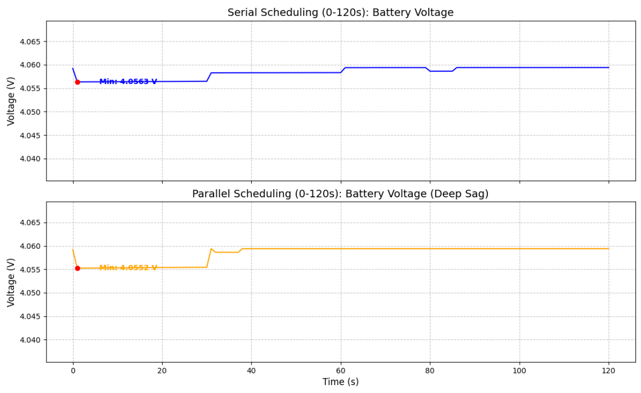


Figure 17: Batt_Vol affected by converter efficiency

Analysis:

From Figure 14, it's easy to found that the curves of serial configuration and parallel is almost overlap together, and the survival time differs only slightly between these two configurations.
In Figure 15, we select the first 120 seconds to do the analysis, because for both configurations, the period is the same(120s), and we calculate the area by time and current. We found that during one period, the area for the serial config and parallel config is the same. That's why the SOC curve for both

configurations is almost the same. And the slight difference in soc time between the two configurations is because of the different peak currents for each configuration, which will lead to different converter efficiency according to the datasheet, but according to the lookup table, the efficiency of the two configurations is similar, and we take battery internal resistance into account. Higher peak current demands lead to greater voltage drops, causing the voltage to decrease. To maintain the load power requirements, the converter draws higher currents from the battery, resulting in increased power consumption under the parallel strategy, and causes the final energy consumption to be different, made the different SOC times.
Also, according to the size of the idle current set in the JSON file, we can calculate that the total idle current is about 0.014 mA, and the corresponding conversion efficiency is 31.5%. The efficiency of serial and parallel is about 90% when there is a task. For serial, there is less free time, while parallel has more free time (about 84s vs 56s), so parallel has more heat loss. Therefore, the heat loss caused by low efficiency is also one of the reasons why the serial scheduling can last longer.
Notes: Here, we only analyze the **i_tot**,not the i_batt, because **i_tot** represent the real stress of whole system.

4.3 Third analysis of the Lab:

Considering the limitation of additional cost, we plan to test these combinations for exploratory purposes.
If we consider the real configuration of battery converter and PV converter(both using the step-up converters), we will have these setups available below, and we write a script to analyze the generated sim_trace.txt, as this log very long because of added power sources, we choose to Interval sample the system log, but no effect on final SOC trace results.

Add one battery in parallel:

By adding one battery in parallel, the nominal capacity is doubled, and the Internal resistance is half the original. By changing the corresponding parameters in **battery_voc.cpp**, the lifetime results are shown below in Figure 17:

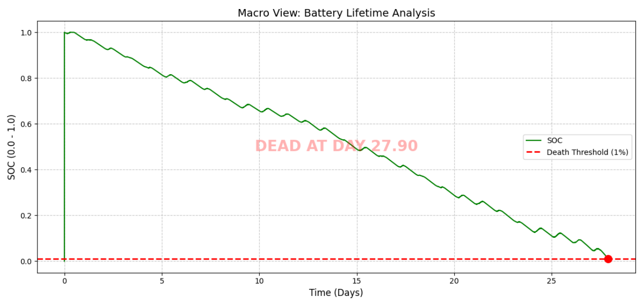


Figure 18: SOC trace by adding one battery in parallel

Add two batteries in parallel:

By adding two battery in parallel, the nominal capacity is tripled, and the Internal resistance is one-third of the original. By changing the corresponding parameters in **battery_voc.cpp**, the lifetime results are shown below in fthe igure :

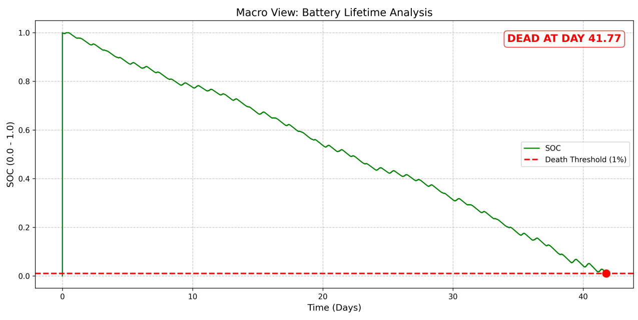


Figure 19: SOC trace of adding two battery in parallel

Add one PV in parallel:

By modifying the code in pv_panel.cpp, we change the current generated by PV, and keep the original voltage. The result is shown in Figure 19.

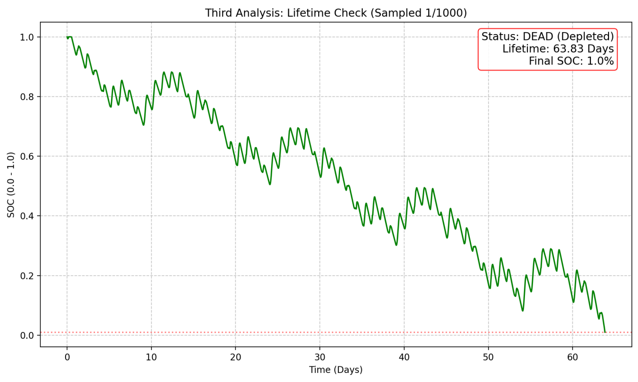


Figure 20: SOC trace of adding one PV in parallel

Add two PV in parallel:

As like previous modification made, we also change the current generated by PV. The result is shown in Figure 20.

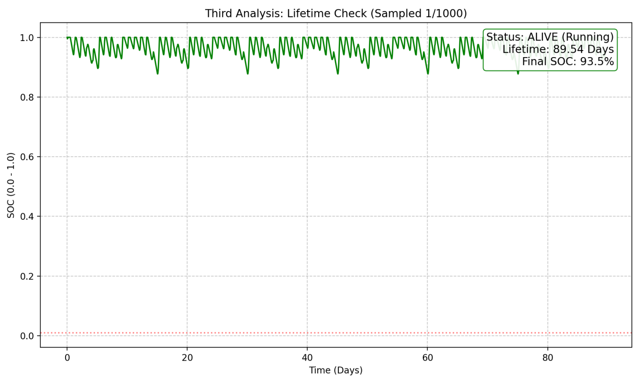


Figure 21: SOC trace of adding two PV in parallel

Hybrid: Add one PV in parallel and one battery in parallel

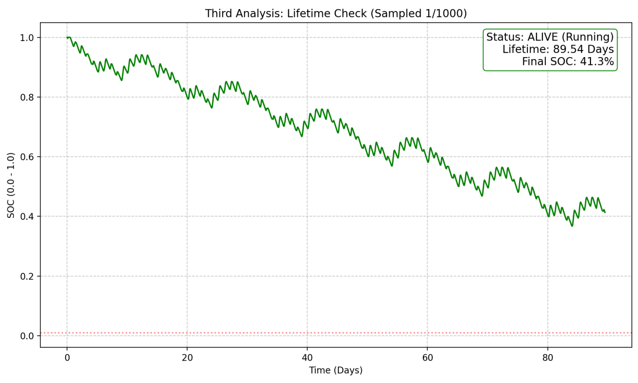


Figure 22: SOC trace of adding one PV in parallel and one battery in parallel

Analysis

For adding 1 battery (in parallel) the SOC life extended from the baseline (approximately 14 days) to 27.90 days. This is because parallel connection doubles the total capacity while halving the internal resistance. The SOC curve exhibits a smooth linear decline. Although the SOC life nearly doubled, the system ultimately ceased operation due to energy depletion.
For adding two batteries (in parallel), the SOC life further extends to 41.77 days. This is because the total battery capacity becomes three times greater. The SOC life improvement is linearly proportional to the increase in capacity. However, this merely delays the time of system failure without altering the fundamental energy deficit where consumption exceeds generation.
For Adding one PV panel(in parallel) significantly extends the SOC life to 63.83 days. At this point, the system has two PV panels. The SOC curve shows pronounced oscillations (charging during the day, discharging at night), indicating a substantial reduction in the energy deficit. Although the battery was eventually depleted, this outcome is significantly better than simply adding more batteries.
For Adding 2 PV panels (in parallel) is the optimal solution: The system remains ALIVE, with SOC reaching 93.5% at simulation end. With 3 PV panels, the system achieves Energy Autonomous status. Daily energy production fully covers and even exceeds consumption, maintaining SOC consistently above 90%. This is the most effective approach for achieving perpetual operation.
For the 1 PV + 1 Battery (parallel) The system remained ALIVE, with a SOC of 41.3% at simulation end. This configuration doubles both charging speed and storage depth. Although the system survived the 90-day simulation period, the SOC curve exhibits a gradual decline (from 100% to 41%). Compared to the “2 PV” configuration, the hybrid setup demonstrates weaker long-term stability. Under poorer sunlight conditions or extended operation, this system may eventually deplete its charge.

Serial configurations:

If we didn’t consider the real configuration of DC-DC converters(they are both **Boost converters**, if we add more power source in serial, boost converters may cannot provide the 3.3v to power BUS). But considering that the simulator did not modeling the specific circuit of the converters, we can try to add some serial configurations for power source.

Add two batteries in serial

We tripled the Voc and internal resistance of the batteries(by modifying the paremeters in battery_voc.cpp). The result is shown in Figure 22.

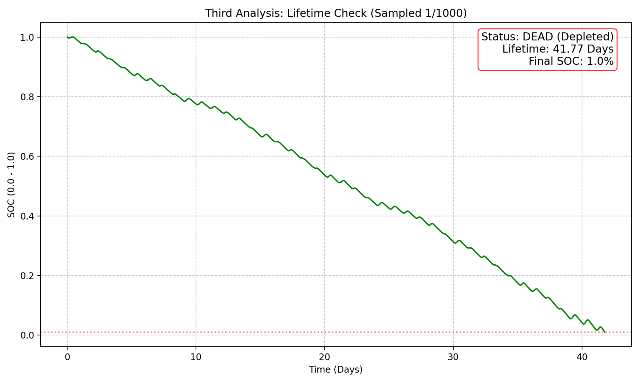


Figure 23: SOC trace of adding two batteries in serial

Add one PV in serial

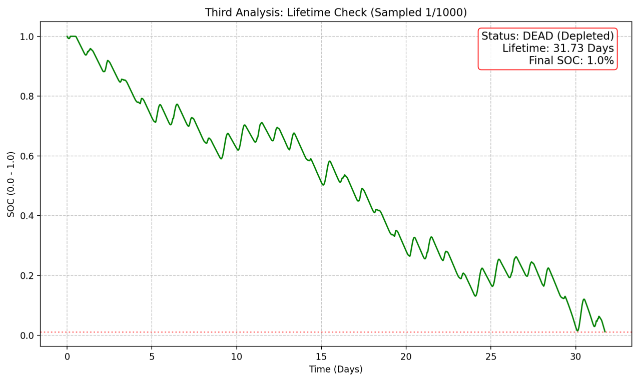


Figure 24: SOC trace of adding one PV in serial

Analysis:

Adding two batteries in series yields the same result as previously adding two batteries in parallel. This is because the total Wh capacity of the expanded battery pack remains unchanged. For the battery converter, we only performed efficiency-related modeling corresponding to the load current (the load itself did not change), which is independent of input voltage variations. Therefore, its SOC lifetime remains consistent with the previous configuration.
For adding a PV series connection, its SOC lifetime has decreased significantly compared to the previous parallel configuration—from 63 days to 31 days. This occurs because PV panel converter efficiency modeling is based on input voltage. Adding a new PV panel in series causes its output voltage to fall precisely within the converter's low-efficiency range, resulting in excessive energy loss. In contrast, the previous parallel configuration maintained the output voltage, potentially keeping it within the higher-efficiency range.