

Brief Overview of the Generator Script

The generator script (`generate_sa818s_at_commands.py`) creates a JSON file (`sa818s_at_commands.json`) containing:

- **Initialization Commands:** One-time setup commands for the transmitter and receiver, ensuring proper module operation.
- **Configurations:** 1,952 `AT+DMOSETGROUP` command pairs for the PMR446 band (16 frequencies × 122 CXCSS codes), with:
 - **Transmitter:** Transmit-only, using valid `TFV` (frequency) and `Tx_CXCSS`, with dummy `RFV` (400.0000 MHz) and `Rx_CXCSS` (0000).
 - **Receiver:** Receive-only, using valid `RFV`, `Rx_CXCSS`, and `SQ=4`, with dummy `TFV` and `Tx_CXCSS`.
 - Format per entry:

```
{
  "id": <1-1952>,
  "type": "NONE" | "CTCSS" | "CDCSS",
  "frequency": "<e.g., 446.0063>",
  "at_tx_cmd": "AT+DMOSETGROUP=0,<freq>,400.0000,
<cxcss>,4,0000\r\n",
  "at_rx_cmd": "AT+DMOSETGROUP=0,400.0000,<freq>,0000,4,
<cxcss>\r\n"
}
```

Purpose: Generates all possible PMR446 configurations for a transmit-only SA818S module (PTT pin low) and a receive-only module (PTT pin high), ensuring compliance with PMR446 standards (12.5 kHz bandwidth, 500 mW via H/L pin low).

Key Features:

- Covers 16 PMR446 frequencies (446.00625–446.19375 MHz).
- Includes 122 CXCSS codes (1 NONE, 38 CTCSS, 83 CDCSS).
- Uses fixed squelch (4) and dummy values for irrelevant parameters.
- Saves to a compact JSON (~500–600 KB).

Initialization Commands

The initialization commands are included in the `initialization` block of `sa818s_at_commands.json`. They are applied once to set up the modules before sending `AT+DMOSETGROUP` commands. Based on the generator script, they are:

Transmitter Initialization (Transmit-Only):

- **AT+DMOCONNECT:** Handshake to confirm module operation.
- **AT+SETFILTER=0,0,0:** Enables pre-emphasis/de-emphasis, high-pass, and low-pass filters for audio quality.

- **AT+SETTAIL=0:** Disables tail tone to avoid unnecessary tones after transmission.

JSON Representation:

```
"transmitter": [
  "AT+DMOCONNECT\r\n",
  "AT+SETFILTER=0,0,0\r\n",
  "AT+SETTAIL=0\r\n"
]
```

Note: PTT pin (Pin 5) must be set to low (0) via hardware to lock in transmit-only mode (SA818S datasheet, Page 10).

Receiver Initialization (Receive-Only):

- **AT+DMOCONNECT:** Handshake to confirm module operation.
- **AT+DMOSETVOLUME=4:** Sets volume to mid-level (1–8 scale, 4 chosen as default per your example).
- **AT+SETFILTER=0,0,0:** Enables all filters for optimal audio reception.

JSON Representation:

```
"receiver": [
  "AT+DMOCONNECT\r\n",
  "AT+DMOSETVOLUME=4\r\n",
  "AT+SETFILTER=0,0,0\r\n"
]
```

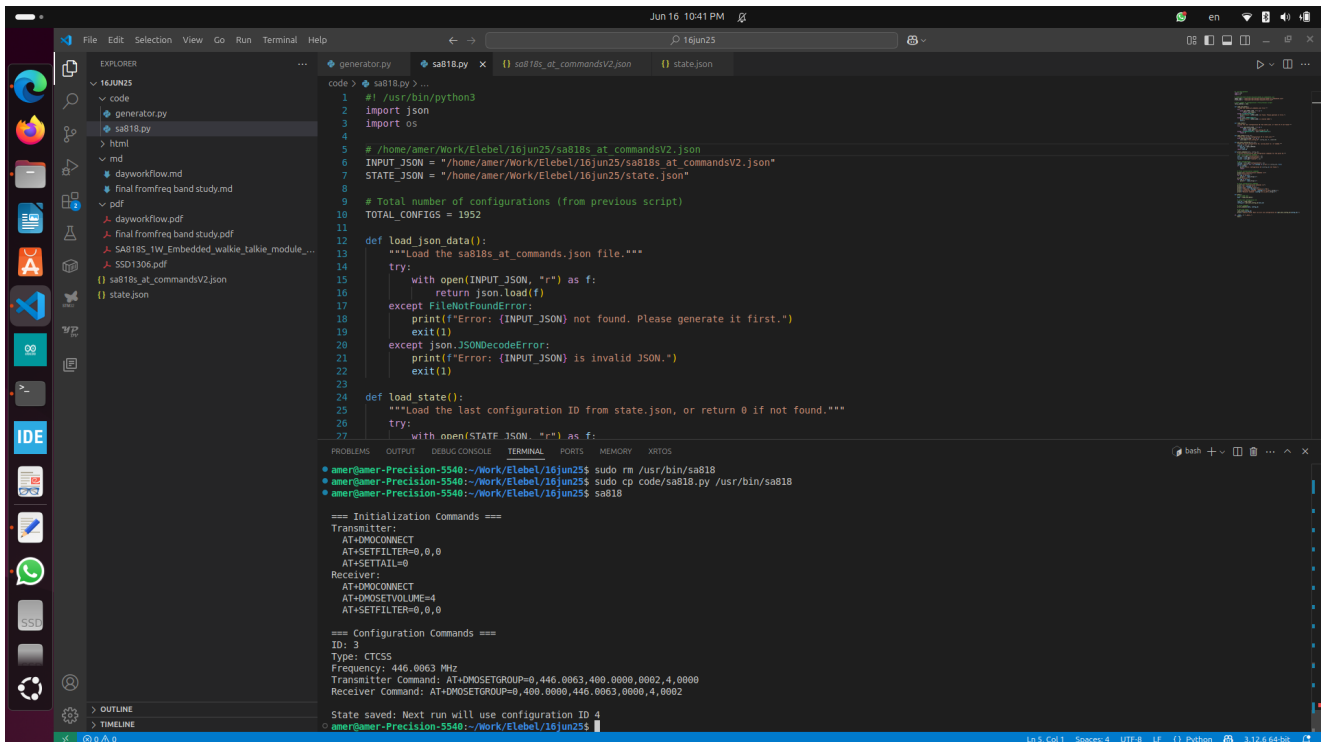
Note: PTT pin must be set to high (1) to lock in receive-only mode.

⚙ Usage Context

- **How to Use:**
 - Send initialization commands once via serial port (9600 bps, 8N1, datasheet Page 5) using a terminal (e.g., PuTTY) or script.
 - Example: For transmitter, send `AT+DMOCONNECT\r\n`, then `AT+SETFILTER=0,0,0\r\n`, then `AT+SETTAIL=0\r\n`.
 - Follow with an `AT+DMOSETGROUP` command from the configurations block (e.g., ID 1049: `AT+DMOSETGROUP=0,446.2000,400.0000,223I,4,0000\r\n` for transmitter).
 - Set H/L pin (Pin 7) low for 500 mW to comply with PMR446 in UAE.
- **Regulatory Notes:**
 - **UAE:** PMR446 is license-free (500 mW, 12.5 kHz, TDRA-approved equipment).
 - **Egypt:** PMR446 likely requires a license. Contact NTRA (info@tra.gov.eg) for confirmation and ETA certification.

Next Steps

- **Run the Generator:** If not done, run `generate_sa818s_at_commands.py` to create `sa818s_at_commands.json`.
- **Use the State Script:** Run the second script (`print_next_sa818s_config.py`) to print initialization commands and cycle through configurations, saving state in `state.json`.
- **Further Assistance:**
 - Need the initialization commands tested with a serial communication script?
 - Want an NTRA inquiry email draft for Egypt?
 - Prefer adjustments to initialization (e.g., different volume or filter settings)?



The screenshot shows a VS Code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a project structure with files like `generator.py`, `sa818.py`, `sa818s_at_commandsV2.json`, and `state.json`. The `sa818.py` file is open in the editor, showing Python code for loading JSON data and state. The terminal window shows the output of running the script, including initialization commands for a transmitter and receiver, and configuration commands for a CTCSS frequency.

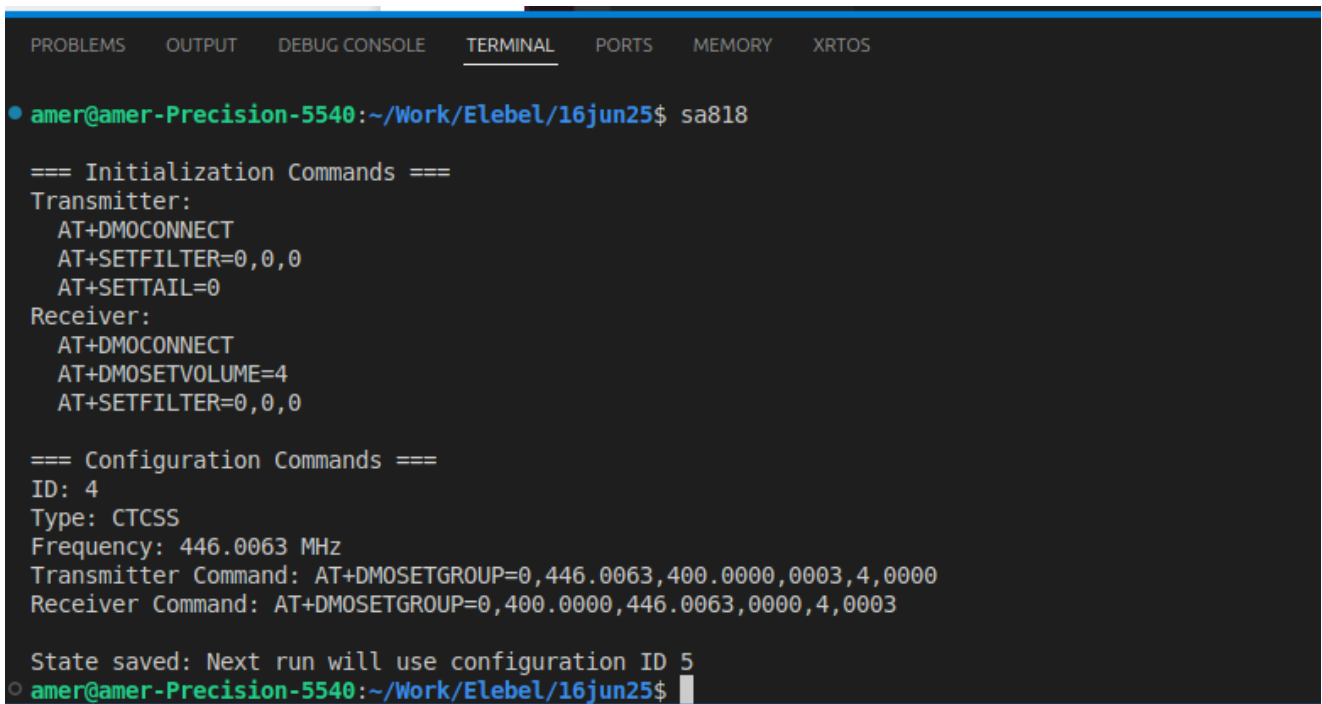
```
1 #! /usr/bin/python3
2 import json
3 import os
4
5 # /home/amer/Work/Ellebel/16jun25/sa818s_at_commandsV2.json
6 INPUT_JSON = "/home/amer/Work/Ellebel/16jun25/sa818s_at_commandsV2.json"
7 STATE_JSON = "/home/amer/Work/Ellebel/16jun25/state.json"
8
9 # Total number of configurations (from previous script)
10 TOTAL_CONFIGS = 1952
11
12 def load_json_data():
13     """Load the sa818s_at_commands.json file."""
14     try:
15         with open(INPUT_JSON, "r") as f:
16             return json.load(f)
17     except FileNotFoundError:
18         print(f"Error: {INPUT_JSON} not found. Please generate it first.")
19         exit(1)
20     except json.JSONDecodeError:
21         print(f"Error: {INPUT_JSON} is invalid JSON.")
22         exit(1)
23
24 def load_state():
25     """Load the last configuration ID from state.json, or return 0 if not found."""
26     try:
27         with open(STATE_JSON, "r") as f:
```

```
amer@amer-Precision-5540:~/Work/Ellebel/16jun25$ sudo rm /usr/bin/sa818
amer@amer-Precision-5540:~/Work/Ellebel/16jun25$ sudo cp code/sa818.py /usr/bin/sa818
amer@amer-Precision-5540:~/Work/Ellebel/16jun25$ sa818

=== Initialization Commands ===
Transmitter:
AT+DMOCONNECT
AT+SETFILTER=0,0,0
AT+SETTAIL=0
Receiver:
AT+DMOCONNECT
AT+DMOSETVOLUME=4
AT+SETFILTER=0,0,0

=== Configuration Commands ===
ID: 3
Type: CTCSS
Frequency: 446.0063 MHz
Transmitter Command: AT+DMOSETGROUP=0,446.0063,400.0000,0003,4,0000
Receiver Command: AT+DMOSETGROUP=0,400.0000,446.0063,0000,4,0003

State saved: Next run will use configuration ID 4
amer@amer-Precision-5540:~/Work/Ellebel/16jun25$
```



The screenshot shows a terminal window with the output of the `sa818` script. The output displays initialization commands for a transmitter and receiver, configuration commands for a CTCSS frequency, and the state saved for the next run.

```
amer@amer-Precision-5540:~/Work/Ellebel/16jun25$ sa818

=== Initialization Commands ===
Transmitter:
AT+DMOCONNECT
AT+SETFILTER=0,0,0
AT+SETTAIL=0
Receiver:
AT+DMOCONNECT
AT+DMOSETVOLUME=4
AT+SETFILTER=0,0,0

=== Configuration Commands ===
ID: 4
Type: CTCSS
Frequency: 446.0063 MHz
Transmitter Command: AT+DMOSETGROUP=0,446.0063,400.0000,0003,4,0000
Receiver Command: AT+DMOSETGROUP=0,400.0000,446.0063,0000,4,0003

State saved: Next run will use configuration ID 5
amer@amer-Precision-5540:~/Work/Ellebel/16jun25$
```