```
  # I'm on a Foxess AC1 5.0 and 28.2 kWh ECS4100 h7 battery stack (Manager 1.78, Master 1.70, Slave 1.02)
  # UK Octopus Cosy, 9 solar panels 300-450W off Enphase 7 series micro inverters, 2500W and just over 20kWh/day peak production

  load_today:
    - sensor.foxess_load_energy_today
  import_today:
    - sensor.foxess_grid_consumption_energy_today
  export_today:
    - sensor.foxess_feed_in_energy_today
  # I'm getting pv_total by monitoring the feed from my solar panels into my consumer unit with a ESPHome flashed Emporia Vue 2
  pv_today:
    - sensor.solar_daily_total


  # Controls/status - must by 1 per inverter
  #
  num_inverters: 1

  inverter_type: FoxESS
  inverter:
    name: "FoxESS"
    has_rest_api: False
    has_mqtt_api: False
    has_service_api: True
    output_charge_control: "power"
    has_charge_enable_time: False
    has_discharge_enable_time: False

    # Theoretically Target SoC should be configured with number.max_soc but this causes issues because
    # target_soc is not automatically reset to 100% after use by switch.predbat_inverter_soc_reset, and
    # this blocks all charging.
    has_target_soc: False

    # Reserve SoC will be configured with number.min_soc_on_grid
    has_reserve_soc: True
    reserve: number.foxess_min_soc_on_grid

    charge_time_format: "S"
    charge_time_entity_is_option: False
    soc_units: "%"
    num_load_entities: 1
    has_ge_inverter_mode: False
    time_button_press: False
    clock_time_format: "%Y-%m-%d %H:%M:%S"
    write_and_poll_sleep: 2
    has_time_window: False
```

```
    support_charge_freeze: true
    support_discharge_freeze: true


# Predbat statuses versus Fox work-modes
#
# PREDBAT           FOX
# Demand (Eco)      Self-Use
# This is the default, the load will be covered by solar and/or battery. Excess solar will
# charge the battery or be exported if the battery is full. This is described as 'Eco' Mode
# for GivEnergy inverters but other inverters use different terminology.
#
# Charging          Force Charge
# The battery charges from the grid and the grid also covers any load. Solar power will
# also be used to charge the battery.
#
# Freeze charging   Back-up (with reserve soc set to number.min_soc_on_grid)
# The battery is charging but the current battery level (SoC) is frozen (held). Think of
# it as a charge to the current battery level. The grid or solar covers any house load.
# If there is a shortfall of Solar power to meet house load, the excess house load is met
# from grid import, but if there is excess Solar power above the house load, the excess
# solar will be used to charge the battery,
#
# Hold charging     Back-up (with reserve soc set to number.min_soc_on_grid)
# A type of charge where the target SoC % is the same as the current SoC %, effectively the
# same as a charge freeze (but without being explicitly selected).
#
# No Charge         Self use
# A charge where the target SoC % is lower than the current battery SoC level so there will
# be no charging unless the usage is unexpectedly high.
#
# Exporting         Force Discharge
# The battery is being force-discharged. The house load will be covered by the battery and
# any excess is exported to the grid. Any solar generated will be exported.
#
# Freeze exporting  Feed-in First
# The battery is in demand mode, but with charging disabled. The
```

```
battery or solar covers the
# house load. As charging is disabled, if there is excess solar generated, the current SoC
# level will be held and the excess solar will be exported. If there is a shortfall of
# generated solar power to meet the house load, the battery will discharge to meet the extra
# load.
#
# Hold exporting    Self use
# The plan was to force export but the minimum battery level was reached and thus the battery
# is kept in Demand mode. If the battery level again gets above the threshold it will be changed
# back to Export mode.
#
# Calibration       Needs an automation
# Fox inverters do not automatically calibrate, so to do this an automation is needed if required.
# At least once every 7 days the battery should hit the 100% SoC mark. This can be done by setting
# the predbat min_keep, etc as needed if too much time has passed.
#
# Error
# If there is a configuration error or other problem, you should check the Predbat log file for more
# details.

  # Services to control charging/discharging
  # charge_start_service - Should be set to a service that is called when charging starts
  charge_start_service:
    service: select.select_option
    entity_id: select.foxess_work_mode
    option: "Force Charge"

  # charge_freeze_service - If your inverter supports charge freeze set to a service that starts this mode
  # The Back-up work mode does not exist on older firmware versions
  charge_freeze_service:
    service: select.select_option
    entity_id: select.foxess_work_mode
    option: "Back-up"

 # charge_stop_service - Should be set to a service that is called when charging/charge freeze stops
  charge_stop_service:
    service: select.select_option
    entity_id: select.foxess_work_mode
    option: "Self Use"

```

```
  # discharge_start_service - Should be set to a service that is called when force export (discharge) starts
  discharge_start_service:
    service: select.select_option
    entity_id: select.foxess_work_mode
    option: "Force Discharge"

  # discharge_stop_service - Should be set to a service that is called when export/export freeze stops
  discharge_stop_service:
    service: select.select_option
    entity_id: select.foxess_work_mode
    option: "Self Use"
#    option: "Feed-in First"
# Should be   option: "Self Use" but I like FiF better ###

  # discharge_freeze_service - If your inverter supports export freeze set to a service that starts this mode
  discharge_freeze_service:
    service: select.select_option
    entity_id: select.foxess_work_mode
    option: "Feed-in First"


  # If not using REST then instead set the Control here (one for each inverter)
  # You should keep this section even when using REST as a fallback if it fails and for charge curve calculations
  # charge_rate and discharge_rate come from the https://github.com/nathanmarlor/foxess_modbus/ integration, they are in kW but works without problems.
  charge_rate:
    - number.foxess_force_charge_power
  discharge_rate:
    - number.foxess_force_discharge_power
  battery_power:
    - sensor.foxess_invbatpower
  pv_power:
    - sensor.circuit_8_power
  load_power:
    - sensor.foxess_load_power
  # I'm getting grid_power by monitoring the mains feed into my consumer unit with a ESPHome flashed Emporia Vue 2
  grid_power:
    - sensor.mains_power
  grid_power_invert:
    - True
  soc_percent:
    - sensor.foxess_battery_soc
  # I'm setting soc_max manually via a number helper
  soc_max:
```

```
    - input_number.soc_max
  # I'm getting this from a template sensor (kWh, Stored Energy, Measurement to get statistics for the battery curve) off {{ ((float(states.sensor.foxess_battery_soc.state)/100) *float(states.sensor.foxess_bms_kwh_remaining.state)) }}
  soc_kw:
    - sensor.foxess_soc_kwh_remaining
  # I'm setting reserve manually via a number helper
  reserve:
    - input_number.reserve
  # I'm setting battery_min_soc manually via a number helper
  battery_min_soc:
    - input_number.battery_min_soc
  battery_temperature:
    - sensor.foxess_battery_temp


  # Inverter max AC limit (one per inverter). E.g for a 3.6kw inverter set to 3600
  # If you have a second inverter for PV only please add the two values together
  inverter_limit:
    - 5000

  # Set the maximum charge/discharge rate of the battery
  battery_rate_max:
    - 14112

  inverter_limit_charge:
   - 5000

  inverter_limit_discharge:
   - 5000
```