* 1. **Загальний огляд компонентів системи**

Так як метою даного дипломного проекту є створення власного аналога моніторингової системи роботи ФЕС, необхідно зазначити певні технічні вимоги щодо розробки наступної. Насамперед, до таких вимог відносяться:

1. можливість роботи з обладнанням різних виробників;
2. портативність обладнання системи;
3. енергоефективність роботи системи;
4. можливість експорту даних для їх подальшого використання у технічному аналізі режимів роботи станції;
5. доступність використаних матеріалів на ринку України;
6. простота у використанні інтерфейсу програми для можливості роботи з ним персоналу без спеціальної підготовки;
7. відносно невелика вартість компонентів системи.

Для забезпечення належного виконання поставлених вимог, розроблена модель складається з чотирьох функціональних блоків, що дозволило спростити як процес розробки системи, так і подальшу експлуатацію окремих її елементів, дає змогу виконувати ремонт обладнання не порушуючи функціювання системи в цілому та легко розширювати існуючий функціонал системи, шляхом інтеграції в неї нових блоків. Отже, розглянемо дану модель більш детально.

Перший блок – блок метеоспостережень, розташований безпосередньо на об’єкті дослідження і призначений для зчитування даних з метеодатчиків та їх передачі по радіозв’язку в блок обробки інформації. Другий блок є блоком зняття електричних даних, здійснює регулярні опитування інвертора для зняття електричних показників станції й аналогічно першому транслює їх до третього блоку системи. До функціоналу третього блоку, що є блоком обробки інформації, відноситься отримання пакетів даних від фізично віддалених об’єктів дослідження, їх ідентифікації, первинної обробки та подальшої передачі на сервер. Четвертий блок – блок програмного забезпечення, що включає в себе REST-сервер, базу даних та клієнтський додаток.

* 1. **Розробка блоку метеоспостережень**

В основі роботи блоку метеоспостережень приймаємо мікроконтролер ATMega328, що обумовлено простотою використання та доступністю даного типу контролеру. Характеристики розглянутого мікроконтролера наведені у табл. 2.2.1. Для спрощення експлуатації мікроконтролера ATMega328 в розробленій схемі використовуються готові друковані плати марки Arduino.

**Таблиця 2.2.1.** – Характеристики мікроконтролера ATMega328

|  |  |  |  |
| --- | --- | --- | --- |
| **Параметр** | **Значення** | **Параметр** | **Значення** |
| Тактова частота | 0-20 МГц | Flash-пам'ять | 32 кб |
| SRAM-пам’ять | 2 кб | EEPROM-пам’ять | 1 кб |
| Напруга живлення | 1,8-5,5 В | Струм в режимі роботи | 0,2 мА |
| Загальна кількість портів | 23 | Струм в режимі сну | 0,75 мкА |
| Кількість ШИМ виходів | 6 | Кількість каналів АЦП | 6 |
| Кількість апаратних USART | 1 | Кількість апаратних SPI | 1 |

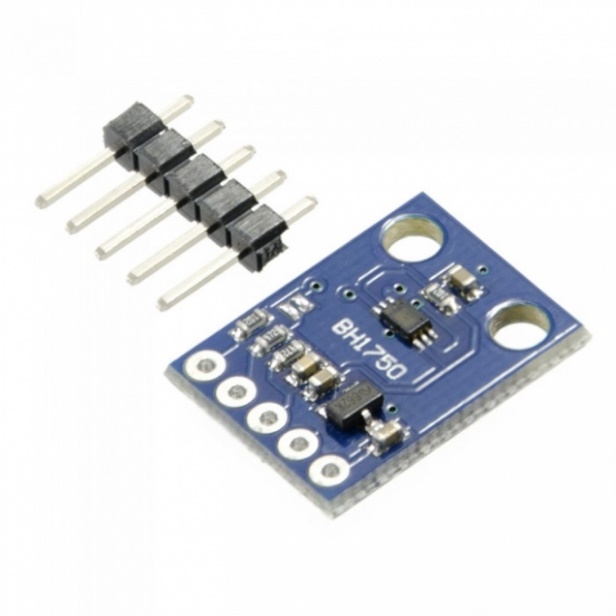
Як вже зазначалося вище, блок метеоспостережень розташований безпосередньо на об’єкті дослідження, тобто фотоелектричного модуля, і призначений для зчитування даних з метеодатчиків та передачі отриманої інформації по радіозв’язку в блок обробки інформації. Він складається з декількох цифрових метеодатчиків, плати Arduino Nano та радіомодуля nRF24L01. Цикл роботи даного блоку умовно можна поділити на декілька етапів:

1. дані, що знімають датчики, оцифровуються та передаються на мікроконтролер з певним інтервалом;
2. мікроконтролер формує пакет з отриманих даних, підписує його та передає на радіомодуль;
3. радіомодуль транслює пакети даних у зарезервований канал.

Отже, у першому етапі циклу роботи блока метеоспостережень, взаємодія йде між цифровими метеодатчиками та мікроконтролером ATMega328. До метеодатчиків, що наявні у базовій комплектації розробленої системи, відносяться: цифровий датчик температури DS18B20 (рис. 2.2.1. – а) та цифровий датчик освітленості GY-302 (рис. 2.2.1. – б). Обидва датчики встановлюються безпосередньо на поверхні фотомодуля і взаємодіють з мікроконтролером за визначеним алгоритмом. Передача даних з датчика освітленості виконується за допомогою інтерфейсу I2C, що є послідовною шиною даних для зв'язку інтегральних схем. В той час, як передача даних з датчика температури виконується через двонапрямлену шину зв’язку – публічний інтерфейс 1-Wire.



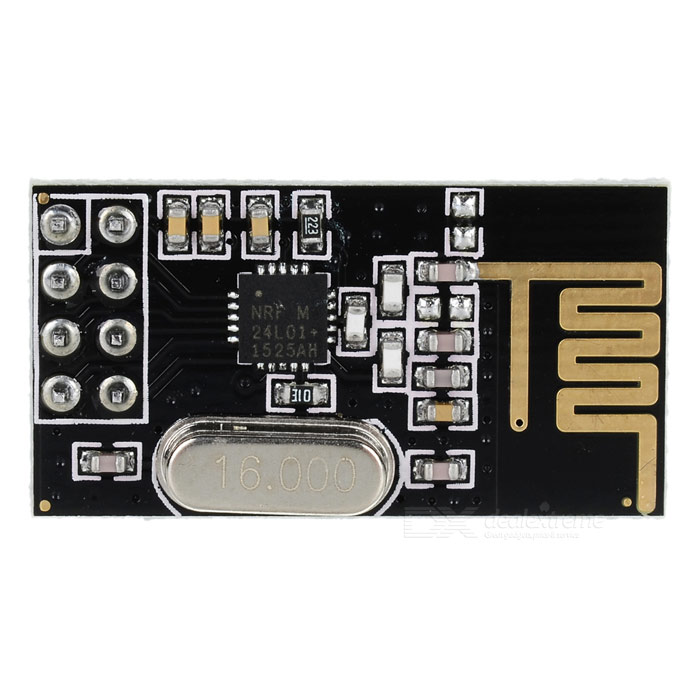
а)



б)

**Рисунок 2.2.1.** – Метеодатчики: а) цифровий датчик температури DS18B20; б) цифровий датчик освітленості GY-302.

У другому етапі мікроконтролер спочатку виконує первинну обробку інформаційних потоків, тобто формує пакети даних: ідентифікує значення з кожного датчика, підписує їх відміткою часу та здійснює кодування пакету у зручний для передачі формат. Далі за допомогою шини SPI, що по суті є синхронним послідовним повнодуплексним стандартом передачі даних, виконується відправка підготовлених пакетів до радіомодуля nRF24L01, який зображений на рисунку 2.2.2.



**Рисунок 2.2.2.** – Радіомодуль nRF24L01

Третій етап є завершальним етапом циклу роботи блоку метеоспостережень і реалізовує зв’язок останнього з блоком обробки інформації. В ході роботи даного етапу радіомодуль nRF24L01 транслює отримані пакети даних у заздалегідь зарезервований канал. Живлення радіомодуля виконується з п’яти вольтового виходу плати Arduino Nano через регулятор напруги 3,3 В AMS1117.

Схематичне зображення описаної вище схеми наведене на рисунку 2.2.3. Код прошивки мікроконтролера наведений у додатку 1.

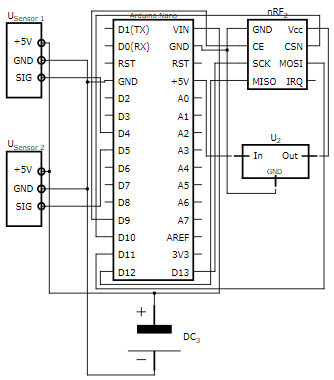


Рисунок 2.2.3. – Принципова схеми блоку метеоспостережень.

* 1. **Розробка блоку зняття електричних даних**

Для зменшення впливу фактору дискретності енергетичних потоків та забезпечення прогнозованості генерації потужності ФЕС можна використовувати моніторингові системи. Дані системи призначені для збору та аналізу даних щодо режимів роботи об’єкта дослідження. Зібранні дані можна умовно поділити на метеорологічні (інтенсивність сонячної радіації, вологість повітря, температура, тиск, швидкість вітра) та електричні (сила струму, напруга, частота). Збір електричних даних було вирішено здійснювати за допомогою готового та існуючого на українському ринку обладнання.

Для збору електричних даних щодо роботи ФЕС існує два основні варіанти:

1. використання зовнішньої гальванічної розв’язки з власними вимірювальними пристроями;
2. використання вбудованої в інвертор моніторингової системи.

Обидва із запропонованих варіантів мають як свої переваги, так і недоліки. Перевагами першого варіанта є не прив’язаність до конкретного бренду/виробника інверторів, можливість зекономити на виборі інвертора (без вбудованої системи моніторингу), можливість передачі даних за допомогою широкого вибору протоколів зв’язку. До його недоліків можна віднести: складність установки та обслуговування даного обладнання. Перевагами другого варіанта є надійність знятих даних (інвертор обов’язково підлягає метрологічній оцінці), простота установки та обслуговування. До його недоліків можна віднести: обмеження щодо використання тільки з конкретним брендом/виробником інверторів, дороговизна, мала кількість протоколів зв’язку, що підтримуються.

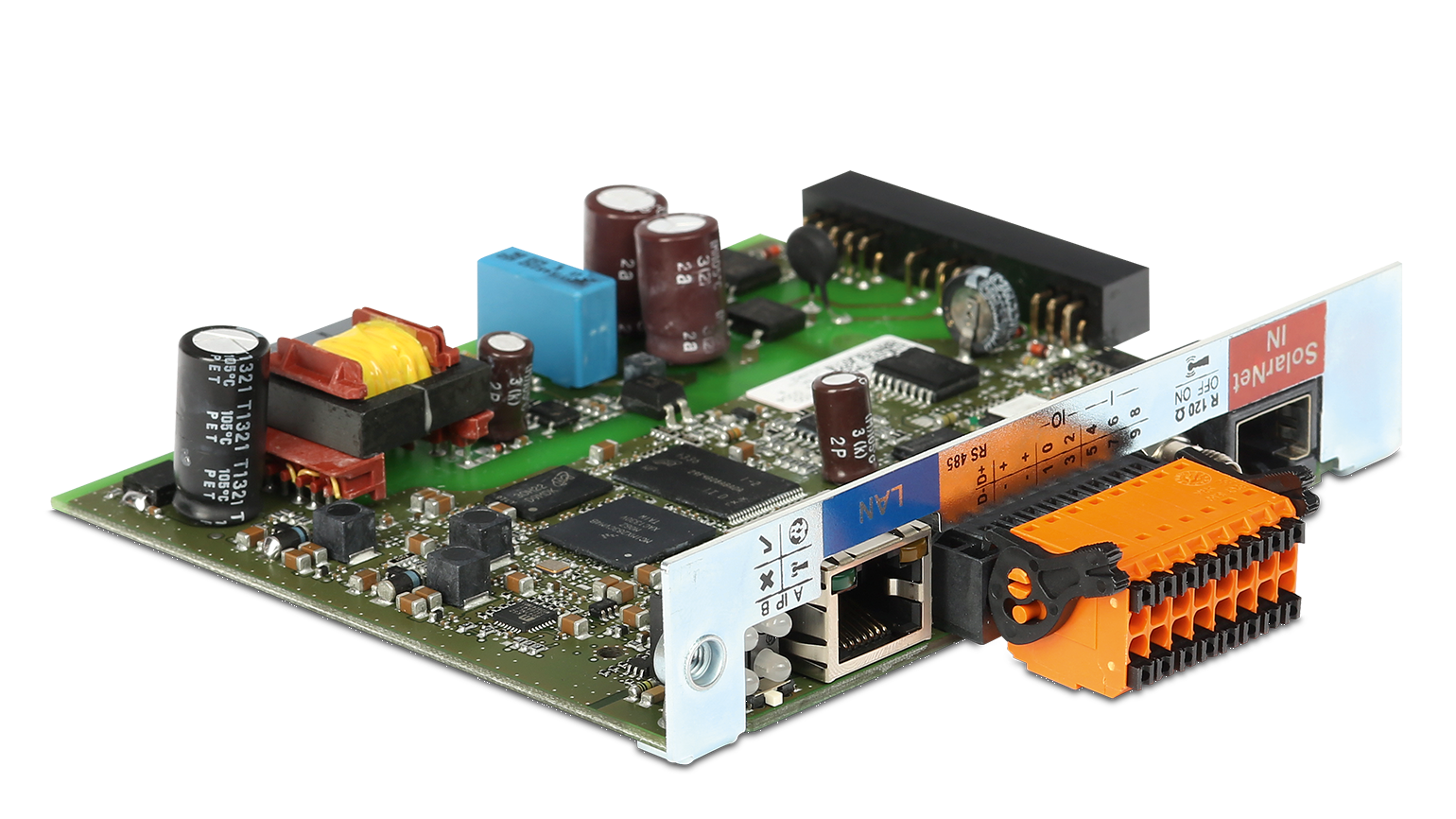
Отже, у ході експертної оцінки було обрано другий варіант. Для подальшого розгляду обираємо інвертор бренду Fronius серії ECO 25.0-3-S.



**Таблиця 4.1.** Номінальні параметри інвертора Fronius ECO 25.0-3-S [2]

|  |  |
| --- | --- |
| Вага | 35,7 кг |
| Розміри | 72.5 x 51 x 22.5 см |
| Номінальна потужність | 27 кВт |
| Кількість фаз | 3 |
| Кількість MPPT виходів | 1 |
| Інтерфейси підключення | RS-485, WiFI, Ethernet |
| Ступінь захисту від вологи | IP 66 |

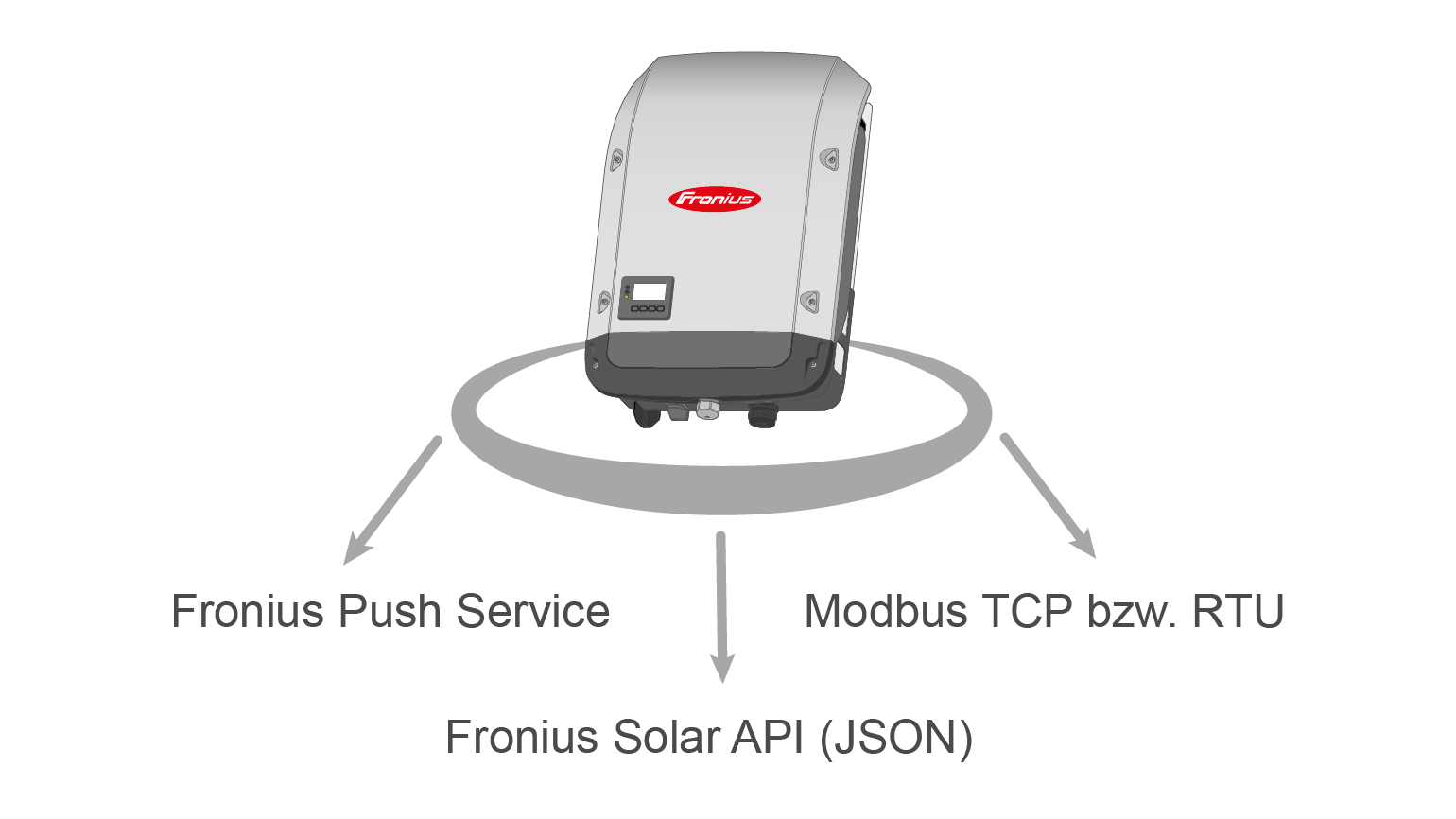
До даної моделі інвертора обираємо наступну внутрішню моніторингову систему – Fronius Datamaneger 2.0. Fronius Datamanager 2.0 – мережевий реєстратор даних, який поєднує у собі функціональність Fronius Com Card, реєстратора даних Fronius Web, Fronius Power Control Card і Fronius Modbus Card [3]. Даний реєстратор має вигляд друкованої плати та монтується усередині інвертора.



Встановивши даний реєстратор ми маємо можливість використовувати будь-який з трьох інтерфейсів зв’язку [3]:

1. RS-485 (англ. Recommended Standard 485), EIA-485 (англ. Electronic Industries Alliance 485) – стандарт передачі даних двопровідним напівдуплексним багатоточковим послідовним каналом зв'язку. У стандарті для передачі і прийому даних часто використовується одна і та ж вита пара дротів. Передача даних здійснюється за допомогою диференціальних сигналів. Різниця напруги однієї полярності між провідниками означає логічну одиницю, різниця іншої полярності – нуль. Стандарт не нормує формат інформаційних кадрів і протокол обміну. Найчастіше для передачі байтів даних використовуються ті ж фрейми, що і в інтерфейсі RS-232: стартовий біт, біти даних, біт паритету (якщо потрібно), стоповий біт.
2. Ethernet – сімейство протоколів стандарту IEEE 802.3 – це найпопулярніший стандарт серед кабельних комп'ютерних мереж, що працюють на фізичному та канальному рівні мережевої моделі OSI. Ethernet тісно пов'язаний з моделлю TCP/IP, оскільки у переважній більшості випадків служить для передачі IP-пакетів.
3. WiFi – загальновживана назва для стандарту IEEE 802.11 передачі цифрових потоків даних по радіоканалах. Дальність передавання інформації залежить від потужності передавача (яка в окремих моделях обладнання регулюються програмно), наявності та характеристики перешкод, типу антени.

Мережевий реєстратор Fronius Datamanager 2.0 має можливість передавати дані по кожному з приведених вище інтерфейсів використовуючи наступні технології: Modbus (TCP, RTU), HTTP API, push service (HTTP, FTP) [3].



Для зручності інтеграції даного мережевого реєстратора у запропоновану моніторингову систему скористаємося публічним інтерфейсом Ethernet для отримання даних за допомогою HTTP API. В даний час єдиним способом взаємодії з цим API є створення HTTP-запиту для певного CGI. Реалізоване API у Fronius Datamaneger 2.0 практично розділяє запити на два види: запити даних знятих в реальному часі та запити архівних даних. Запити даних в реальному часі отримуються безпосередньо з внутрішніх пристроїв інвертора, тому їх можна використовувати тільки тоді, коли інвертор підключений та не знаходиться у стані очікування [4]. Зняті дані передаються у форматі JSON (текстовий формат обміну даними).

Так як друковані плати марки Arduino, що використовуються у запропонованій моніторинговій системі, не мають вбудованого інтерфейсу Ethernet, який необхідний для обраного способу підключення мережевого реєстратора Fronius Datamanager 2.0, скористаємося розширювальною платою W5100 Ethernet Shield. Спілкування між Arduino і розширювальною платою побудовано за допомогою шини SPI, послідовному периферійному інтерфейсі, що є синхронним послідовним повнодуплексним стандартом передачі даних.

Запити до HTTP API мережевого реєстратора Fronius Datamanager 2.0 потребують наявність веб-клієнта. Для побудови наступного скористаємося бібліотекою «Ethernet.h», яка входить до стандартного пакету бібліотек Arduino IDE.

Наведемо приклад побудови веб-клієнта на базі Arduino:

#include <SPI.h>

#include <Ethernet.h>

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

char server[] = "www.google.com";

IPAddress ip(192, 168, 0, 177);

EthernetClient client;

void setup() {

Serial.begin(9600);

if (Ethernet.begin(mac) == 0) {

Ethernet.begin(mac, ip);

}

if (client.connect(server, 80)) {

makeRequest();

}

}

void makeRequest() {

client.println("GET / HTTP/1.1");

client.println("Host: www.google.com");

client.println("Connection: close");

client.println();

}

void loop() {

if (client.available()) {

char c = client.read();

Serial.print(c);

}

if (!client.connected()) {

client.stop();

// закінчити роботу:

while (true);

}

}

HTTP API запропоноване в Fronius Datamanager 2.0 реалізоване по типу локального REST-сервера (архітектурний стиль взаємодії компонентів розподіленого додатка в мережі). Для комунікації з даним API використовуються звичайні GET HTTP/1.1 запити (приклад створення такого запиту був наведений вище у реалізації веб-клієнта на базі Arduino). Усі публічні методи, що підтримує даний мережевий реєстратор наведені у документації до нього – «Fronius Solar API V1».

Наведемо приклад запита, що має повернути дані зняті з внутрішніх вимірювальних приладів інвертора у реальному часі [4]:

GET /solar\_api/v1/GetMeterRealtimeData.cgi HTTP/1.1

Host: 169.254.0.180

Наведемо приклад відповіді API на приведений вище запит [4]:

HTTP/1.1 200 OK

Content-type: “application/json”

{

"Body":{

"Data":{

"0":{

"TimeStamp":1406816001,

"Visible":1,

"PowerReal\_P\_Sum":-834.13,

"Meter\_Location\_Current":0,

"PowerReal\_P\_Phase\_1":-261.86,

"PowerReal\_P\_Phase\_2":-296.26,

"PowerReal\_P\_Phase\_3":-276.01,

"PowerReactive\_Q\_Sum":489.34,

"PowerReactive\_Q\_Phase\_1":169.63,

"PowerReactive\_Q\_Phase\_2":158.39,

"PowerReactive\_Q\_Phase\_3":161.32,

"Current\_AC\_Phase\_1":1.43,

"Current\_AC\_Phase\_2":1.522,

"Current\_AC\_Phase\_3":1.44,

"Voltage\_AC\_Phase\_1":233.1,

"Voltage\_AC\_Phase\_2":234.4,

"Voltage\_AC\_Phase\_3":233.8,

"Voltage\_AC\_PhaseToPhase\_12":404.9,

"Voltage\_AC\_PhaseToPhase\_23":405.5,

"Voltage\_AC\_PhaseToPhase\_31":404.3,

"Frequency\_Phase\_Average":50,

"PowerApparent\_S\_Sum":967,

"PowerFactor\_Sum":0.86,

"PowerFactor\_Phase\_1":0.83,

"PowerFactor\_Phase\_2":0.88,

"PowerFactor\_Phase\_3":0.86,

"EnergyReal\_WAC\_Sum\_Produced":33989,

"EnergyReal\_WAC\_Sum\_Consumed":1365,

"EnergyReactive\_VArAC\_Sum\_Produced":4020,

"EnergyReactive\_VArAC\_Sum\_Consumed":204310,

"EnergyReal\_WAC\_Plus\_Absolute":1365,

"EnergyReal\_WAC\_Minus\_Absolute":33989

}

}

}

}