|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Базовая кафедра №234 — Управляющих ЭВМ

**Отчет по практической работе**

|  |  |
| --- | --- |
| **Выполнил:**  Студент группыИКМО-05-23 | Миронов Д.С. |
| **Проверил:** | Плужнова Т.С. |

Москва, 2024

Ссылка на архив: <https://disk.yandex.ru/d/ea1iSB_iFe1XSg>

В варианте 20 задана конфигурация сервера OPC UA. Реализовать заданную конфигурацию сервера OPC UA. Динамически эмулировать сигналы. Организовать запись и сохранение необходимых сигналов.

Имена сигналов могут быть любыми. Тип сигналов указан в варианте.

Условные обозначения:

* R – чтение;
* W – запись;
* H – архивирование;

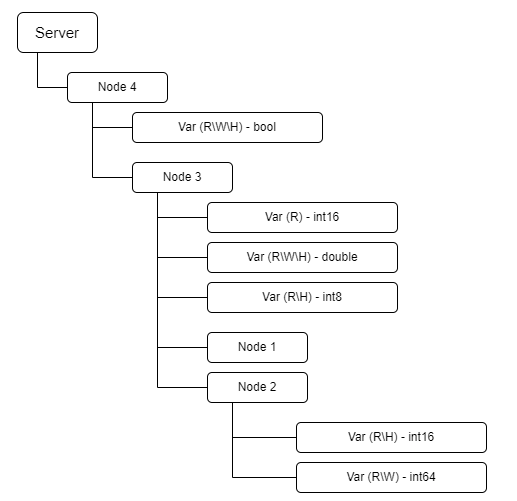


Рисунок 1 – 20 вариант работы

1. **Инициализация**

Необходимо инициализировать OPC UA сервер по адресу opc.tcp://0.0.0.0:4840/freeopcua/server/

Листинг 1

server = Server()

server.set\_endpoint(URL)

server.set\_server\_name("OPC UA Server Example")

1. **Структура**

Создаем структуру узлов

* 1. **Node 4**

Листинг 2

node4 = root\_node.add\_object("ns=2;i=1001", "Node4")

var\_rw\_bool=node4.add\_variable("ns=2;i=2001","Var\_RW\_H", False, ua.VariantType.Boolean)

var\_rw\_bool.set\_writable()

* Узел Node4 добавляется в корневую структуру (root\_node) с уникальным NodeId.
* Переменная Var\_RW\_H типа Boolean добавляется как дочерний узел.
* Метод set\_writable() делает переменную доступной для записи.
  1. **Node 3**

Листинг 3

node3 = root\_node.add\_object("ns=2;i=1002", "Node3")

var\_r\_int16 = node3.add\_variable("ns=2;i=2002", "Var\_R", 0, ua.VariantType.Int16)

var\_rwh\_double=node3.add\_variable("ns=2;i=2003","Var\_RW\_H",0.0, ua.VariantType.Double)

var\_rh\_int8 = node3.add\_variable("ns=2;i=2004", "Var\_RH", 0, ua.VariantType.Byte)

var\_r\_int16.set\_read\_only()

var\_rwh\_double.set\_writable()

* Узел Node3 содержит три переменные:
  + **Var\_R**: Только для чтения (Int16).
  + **Var\_RW\_H**: Доступна для записи и архивирования (Double).
  + **Var\_RH**: Только для чтения и архивирования (Byte).
  1. **Node 1 -> Node 2**

Листинг 4

node1 = root\_node.add\_object("ns=2;i=1003", "Node1")

node2 = node1.add\_object("ns=2;i=1004", "Node2")

var\_rh\_int16 = node2.add\_variable("ns=2;i=2005", "Var\_RH", 0, ua.VariantType.Int16)

var\_rw\_int64 = node2.add\_variable("ns=2;i=2006", "Var\_RW", 0, ua.VariantType.Int64)

var\_rh\_int16.set\_read\_only()

var\_rw\_int64.set\_writable()

* Узел Node1 содержит вложенный узел Node2.
* В Node2 добавлены:
  + **Var\_RH**: Только для чтения и архивирования (Int16).
  + **Var\_RW**: Доступна для записи (Int64).

1. **Обновление переменных**

Функция обновления переменных, в которой происходит генерация случайных значений для переменных, при помощи библиотеки random.

Значения переменных, указанных для архивирования (Var\_RW\_H, Var\_RW\_H\_double, Var\_RH\_int8, Var\_RH\_int16), сохраняются в таблицу базы данных вместе с текущим временем.

Листинг 5

# Функция обновления переменных

def update\_signals():

# Генерация случайных значений для переменных

var\_rw\_bool.set\_value(random.choice([True, False]))

var\_r\_int16.set\_value(random.randint(-32768, 32767))

var\_rwh\_double.set\_value(random.uniform(0, 100))

var\_rh\_int8.set\_value(random.randint(0, 255)) # Byte имеет диапазон от 0 до 255

var\_rh\_int16.set\_value(random.randint(-32768, 32767))

var\_rw\_int64.set\_value(random.randint(-9223372036854775808, 9223372036854775807))

# Сохранение в базу данных (архивируемые переменные)

timestamp = datetime.now()

DB.query("INSERT INTO signals (timestamp, signal\_name, value) VALUES (?, ?, ?)",

(timestamp, "Var\_RW\_H", var\_rw\_bool.get\_value()))

DB.query("INSERT INTO signals (timestamp, signal\_name, value) VALUES (?, ?, ?)",

(timestamp, "Var\_RW\_H\_double", var\_rwh\_double.get\_value()))

DB.query("INSERT INTO signals (timestamp, signal\_name, value) VALUES (?, ?, ?)",

(timestamp, "Var\_RH\_int8", var\_rh\_int8.get\_value()))

DB.query("INSERT INTO signals (timestamp, signal\_name, value) VALUES (?, ?, ?)",

(timestamp, "Var\_RH\_int16", var\_rh\_int16.get\_value()))

1. **Запуск сервера**

Листинг 6

server.start()

print("Server started at {}".format(server.endpoint))

Сервер запускается. Клиенты могут подключиться по указанному адресу.

1. **Основной цикл**

Листинг 7

try:

while True:

update\_signals()

time.sleep(1)

except KeyboardInterrupt:

print("Stopping server...")

finally:

server.stop()

conn.close()

В бесконечном цикле обновляются значения переменных каждую секунду.

При завершении программы сервер корректно останавливается, а соединение с базой данных закрывается.