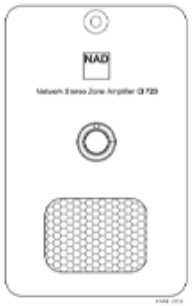
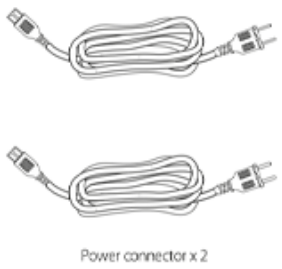
ИНСТРУКЦИЯ ПО ЭКСПЛУАТАЦИИ

Сетевой стереоусилитель NAD CI 720 (включая V2)

NAD CI 720





Power connector x 2

Ethernet cable

Speaker connector

Данный сетевой усилитель разработан компанией NAD Electronics (Канада) для профессионального применения в системах Custom Installation (CI). Усилитель предназначен для установки в технические стойки и коммерческие инсталляции: крупные жилые комплексы, рестораны, отели, офисные здания, конференц-залы . Усилитель не предназначен для использования в жилых помещениях в качестве настольного компонента из-за наличия активного вентилятора (требуется расчёт теплоотвода). Поддерживается интеграция с ведущими профессиональными системами управления: Control4, Crestron, RTI, URC, AMX.

Класс: однозонный сетевой усилитель для распределенных аудиосистем и коммерческих инсталляций

ЦЕЛЕВОЕ НАЗНАЧЕНИЕ (ПРОФЕССИОНАЛЬНОЕ НЕБЫТОВОЕ ПРИМЕНЕНИЕ)

Сетевой стереоусилитель NAD CI 720 V2 **предназначен исключительно для профессионального небытового использования** . Это специализированный компонент для систем Custom Installation (CI), используемый интеграторами в коммерческих и жилых многозонных проектах .

Основные профессиональные сценарии применения:

Сфера применения	Тип задач
Крупные жилые комплексы	Многозонное распределенное аудио, фоновое озвучивание
Коммерческие помещения	Рестораны, бары, отели, торговые центры, фитнес-клубы
Офисные здания	Фоновое озвучивание офисов, переговорных комнат, зон отдыха
Конференц-залы	Озвучивание презентаций, конференций
Образовательные учреждения	Озвучивание аудиторий, классов, кампусов
Профессиональные инсталляции	Системы до 64 зон с единым управлением через BluOS

Категорически не предназначено для: бытового использования в качестве настольного компонента в жилой комнате. Усилитель оснащен активным вентилятором и предназначен для монтажа в технические стойки .

Ключевые особенности для профессионального применения:

1. **CI-ориентированный дизайн:** Компактный модуль высотой 3U, возможность монтажа до 6 устройств в стойке RM720, активное термостатическое охлаждение .
2. **BluOS для многозонных систем:** Полная интеграция с экосистемой BluOS, поддержка до 64 зон с единым управлением через мобильное приложение .
3. **Программируемость через Python и API:** Полный контроль через IP API и веб-интерфейс для автоматизации и интеграции с профессиональными системами управления .
4. **Профессиональные интерфейсы:** Phoenix/Euroblock акустические терминалы для надежного монтажа, Ethernet (без Wi-Fi) для стабильного сетевого подключения .
5. **Корпоративные интеграции:** Сертифицированная поддержка Control4, Crestron, RTI, URC, AMX и других систем управления .

---

1. ВВЕДЕНИЕ И ОБЩИЕ СВЕДЕНИЯ

Настоящая инструкция предназначена для **квалифицированного персонала — системных интеграторов и инсталляторов**, использующих усилитель NAD CI 720 V2 в профессиональных инсталляциях. Усилитель построен на фирменной технологии HybridDigital (класс D) и оснащен встроенным сетевым стримером BluOS для интеграции в многозонные аудиосистемы .

Основные технические характеристики:

Параметр	Значение
Тип	Сетевой стереоусилитель для инсталляций
Выходная мощность (RMS)	2 x 60 Вт (8/4 Ом)
Пиковая мощность	2 x 165 Вт
THD+N	<0,005%
Соотношение сигнал/шум (SNR)	-110 дБА
Коэффициент демпфирования	>100

Параметр	Значение
Частотный диапазон	20 Гц - 20 кГц (±0,5 дБ)
Технология усиления	HybridDigital (класс D)
BluOS поддержка	До 64 зон, управление через приложение
Потребление (ожидание)	6 Вт (с сетью), 0,5 Вт (глубокий сон)
Максимальное потребление	150 Вт
Тепловыделение	4,9 Вт/ч / 16,7 BTU/ч

Подключения и разъемы:

Тип	Назначение
Аналоговый вход (RCA)	1 пара — линейный вход
Оптический вход (Toslink)	1 — цифровой вход
USB Type-A	Воспроизведение с USB-накопителей (FAT32)
Ethernet (RJ45)	Gigabit — проводное сетевое подключение (нет Wi-Fi!)
IR IN (3,5 мм)	Вход для внешнего ИК-приемника
Акустические выходы	Phoenix/Euroblock терминальная колодка
SUBW OUT (RCA)	Выход на активный сабвуфер

2. ТРЕБОВАНИЯ БЕЗОПАСНОСТИ

Перед эксплуатацией системы ознакомьтесь со следующими требованиями:

- **Электропитание:** Используйте оба прилагаемых съемных кабеля питания. Напряжение: соответствует региону. Основной метод изоляции — отключение кабеля питания от сети .
- **Вентиляция:** Обеспечьте свободный доступ воздуха. Рекомендуемый зазор: не менее 10 см слева, справа, сзади и сверху. Усилитель оснащен активным вентилятором, не блокируйте вентиляционные отверстия .
- **Температура:** Эксплуатируйте в сухих условиях при комнатной температуре.
- **Влажность:** Прибор не должен подвергаться воздействию капель или брызг. Не устанавливайте рядом с водой .
- **Чистка:** Отключите от сети перед чисткой. Используйте только сухую мягкую ткань. Не применяйте жидкости или аэрозоли .
- **Монтаж:** При установке в стойку RM720 используйте прилагаемые винты. При настенном монтаже используйте монтажные отверстия на боковых панелях .

⚠ **ПРОФЕССИОНАЛЬНОЕ ПРИМЕЧАНИЕ:** Усилитель **не поддерживает Wi-Fi** — только проводное подключение Ethernet (RJ45) . Это обеспечивает максимальную стабильность в профессиональных инсталляциях.

⚠ **ПРОФЕССИОНАЛЬНОЕ ПРИМЕЧАНИЕ:** Усилитель оснащен активным вентилятором (уровень шума 25 дБ) . Предназначен для установки в технические стойки/шкафы, а не для размещения в жилых комнатах.

⚠ **ПРИМЕЧАНИЕ ПО КАБЕЛЯМ:** Для Phoenix-терминалов рекомендуется использовать акустические кабели с моножилой (solid core) для надежной фиксации. Многожильные кабели могут не обеспечить стабильного контакта .

---

3. РАСПАКОВКА И ПРОВЕРКА КОМПЛЕКТАЦИИ

1. Извлеките компоненты из упаковки. **Сохраните оригинальную упаковку для транспортировки** — это самый безопасный контейнер для перемещения усилителя .
2. Проверьте комплектацию :

Компонент	Описание
NAD CI 720 V2	Основной блок усилителя
Съемные кабели питания	2 шт.

Компонент	Описание
Ethernet-кабель	Для проводного подключения к сети
Акустическая терминальная колодка	Phoenix/Euroblock
Винты для монтажа	Для установки в стойку RM720
Руководства	Краткое руководство по установке

3. Осмотрите корпус на предмет повреждений.

---

4. УСТАНОВКА И РАЗМЕЩЕНИЕ (ПРОФЕССИОНАЛЬНАЯ КОНФИГУРАЦИЯ)

4.1. Габаритные размеры и вес

Параметр	Значение
Ширина	70 мм
Высота	120 мм
Глубина	350 мм
Вес	1,6 кг
Высота в стойке	3U

4.2. Варианты установки

А. Монтаж в 19" стойку (рекомендовано для профессиональных инсталляций):

- Используйте опциональный монтажный комплект **RM720**
- В один комплект RM720 можно установить до **6 усилителей CI 720 V2** в пространстве 3U
- Закрепите каждый усилитель прилагаемыми винтами

Б. Настенный монтаж:

- Используйте два монтажных отверстия на боковых панелях усилителя
- Описание процедуры приведено в официальном руководстве

В. Настольная установка (только для временного размещения):

- Установите на прилагаемые ножки
- Обеспечьте зазор 10 см со всех сторон для вентиляции

4.3. Подключение питания

1. Перед подключением убедитесь, что усилитель выключен.
2. Подключите **оба** прилагаемых кабеля питания к разъемам питания на задней панели.
3. Подключите кабели к розеткам сети с заземлением.
4. Усилитель включится автоматически.

4.4. Подключение акустических систем

Усилитель использует **Phoenix/Euroblock терминальную колодку** для подключения колонок .

Процедура подключения:

1. Зачистите концы акустических кабелей (рекомендуется solid core кабель) .
2. Вставьте провода в соответствующие отверстия терминальной колодки (L+, L-, R+, R-).
3. Затяните винты для фиксации проводов.
4. Вставьте терминальную колодку в разъем на задней панели усилителя.

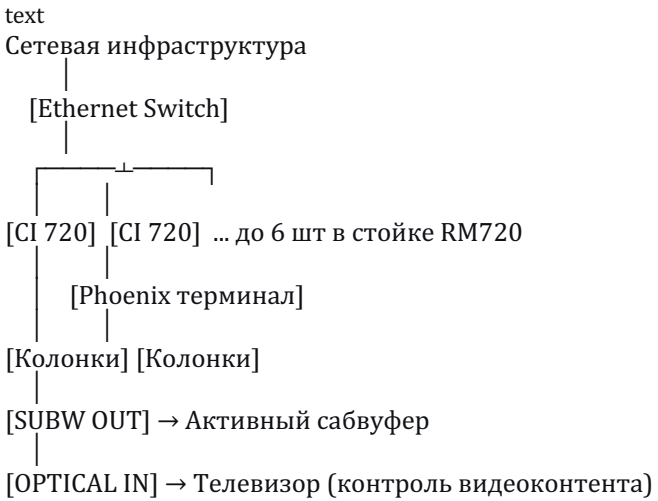
Рекомендация по кабелям: Используйте моножильные (solid core) акустические кабели, например AudioQuest SLiP серии. Многожильные кабели не обеспечивают надежной фиксации .

4.5. Сетевое подключение

Важное ограничение: Усилитель **не поддерживает Wi-Fi** — используйте только проводное подключение Ethernet .

1. Подключите Ethernet-кабель к порту RJ45 на задней панели.
2. Подключите другой конец к сетевому коммутатору/маршрутизатору.
3. Убедитесь, что в сети работает DHCP-сервер для автоматического получения IP-адреса.

4.6. Типовая схема подключения



---

5. ПРОГРАММИРОВАНИЕ НА PYTHON (BLUOS API)

5.1. Общие сведения

Усилитель NAD CI 720 V2 имеет встроенный модуль BluOS и поддерживает **BluOS API** — RESTful веб-сервис, идентичный API устройств Bluesound и других NAD компонентов . API доступен через HTTP-запросы к встроенному веб-серверу.

Базовые параметры подключения:

- **Протокол:** HTTP
- **Порт:** 80
- **IP-адрес:** IP-адрес устройства в локальной сети
- **Формат ответа:** XML (key=value) или JSON
- **Доступна полная API документация** для интеграции с профессиональными системами

5.2. Определение IP-адреса устройства

Усилитель получает IP-адрес автоматически через DHCP. Для обнаружения используйте:

Метод 1: Через BluOS App

- Приложение BluOS для iOS/Android автоматически обнаруживает все устройства в сети

Метод 2: Через веб-интерфейс

- Откройте браузер и введите `http://bluos-XXXXXX.local` (XXXXXX — серийный номер устройства)

Метод 3: Через DHCP-сервер

- Проверьте список выданных IP-адресов на маршрутизаторе

Метод 4: Через UDP обнаружение (Python)

```
python
import socket
import requests
from typing import List, Dict, Optional

def discover_bluos_devices() -> List[Dict]:
    """
    Обнаружение устройств BluOS (включая NAD CI 720 V2)
    использует UPnP обнаружение (SSDP)
    """
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM, socket.IPPROTO_UDP)
    sock.settimeout(5)

    MCAST_ADDR = "239.255.255.250"
    MCAST_PORT = 1900
```

```
    search_message = (
        "M-SEARCH * HTTP/1.1\r\n"
        "HOST: 239.255.255.250:1900\r\n"
        "MAN: \"ssdp:discover\"\r\n"
        "MX: 3\r\n"
        "ST: urn:schemas-upnp-org:device:MediaRenderer:1\r\n\r\n"
    ).encode()

    sock.sendto(search_message, (MCAST_ADDR, MCAST_PORT))

    devices = []
    try:
        while True:
            data, addr = sock.recvfrom(1024)
            response = data.decode()
            if "Bluesound" in response or "bluos" in response.lower() or "NAD" in response:
                for line in response.split('\r\n'):
                    if line.lower().startswith('location:'):
                        location = line.split(':', 1)[1].strip()
                        devices.append({'ip': addr[0], 'location': location})
                        break
    except socket.timeout:
        pass
    finally:
        sock.close()

    return devices

def get_device_info(ip_address: str) -> Dict:
    """
    Получение подробной информации об устройстве (NAD CI 720 V2)
    """
    try:
        response = requests.get(f"http://{ip_address}:80/Status", timeout=5)
        info = {}
        for line in response.text.strip().split('\n'):
            if '=' in line:
                key, value = line.split('=', 1)
                info[key] = value
        return info
    except Exception as e:
        return {'error': str(e)}
```

```
# Пример использования
devices = discover_bluos_devices()
for device in devices:
    print(f"Найдено устройство: {device['ip']}")
    info = get_device_info(device['ip'])
    if 'modelName' in info:
        print(f"  Модель: {info['modelName']}")
        print(f"  Название: {info.get('name', 'unknown')}")
```

5.3. Базовые API запросы

BluOS API использует HTTP GET запросы к эндпоинту `/Sync`. Полная спецификация API доступна в документе "BluOS Custom Integration API" .

```
python
import requests
from typing import Dict, Any, Optional, List

class NADCI720V2Controller:
    """
    Класс для управления NAD CI 720 V2 через BluOS API
    """

    def __init__(self, ip_address: str):
        """
        Инициализация контроллера

        Args:
            ip_address: IP-адрес усилителя в локальной сети
        """
        self.ip = ip_address
        self.base_url = f"http://{ip_address}:80"
        self._sync_endpoint = "/Sync"

    def _send_command(self, command: str, params: Optional[Dict] = None) -> Dict[str, Any]:
        """
        Отправка команды к устройству

        Args:
            command: Команда API
            params: Параметры команды
```

```
    Returns:
        Ответ в виде словаря
    """
    url = f"{self.base_url}{self._sync_endpoint}"
    payload = {'cmd': command}
    if params:
        payload.update(params)

    try:
        response = requests.get(url, params=payload, timeout=5)
        response.raise_for_status()

        result = {}
        for line in response.text.strip().split('\n'):
            if '=' in line:
                key, value = line.split('=', 1)
                result[key] = value
        return result
    except requests.exceptions.RequestException as e:
        print(f"Ошибка связи с устройством {self.ip}: {e}")
        return {'error': str(e)}

# === Управление воспроизведением ===

def play(self) -> Dict[str, Any]:
    """Запуск воспроизведения"""
    return self._send_command("play")

def pause(self) -> Dict[str, Any]:
    """Пауза"""
    return self._send_command("pause")

def stop(self) -> Dict[str, Any]:
    """Остановка"""
    return self._send_command("stop")

def next_track(self) -> Dict[str, Any]:
    """Следующий трек"""
    return self._send_command("skip")

def previous_track(self) -> Dict[str, Any]:
    """Предыдущий трек"""
    return self._send_command("back")
```

```
# === Управление громкостью ===

def set_volume(self, volume: int) -> Dict[str, Any]:
    """
    Установка уровня громкости

    Args:
        volume: Значение громкости (0-100)
    """
    volume = max(0, min(100, volume))
    return self._send_command("volume", {"level": volume})

def volume_up(self, step: int = 5) -> Dict[str, Any]:
    """Увеличение громкости"""
    current = self.get_volume()
    if 'volume' in current:
        new_volume = min(100, int(current['volume']) + step)
        return self.set_volume(new_volume)
    return {'error': 'Cannot get current volume'}

def volume_down(self, step: int = 5) -> Dict[str, Any]:
    """Уменьшение громкости"""
    current = self.get_volume()
    if 'volume' in current:
        new_volume = max(0, int(current['volume']) - step)
        return self.set_volume(new_volume)
    return {'error': 'Cannot get current volume'}

def mute(self, state: bool = True) -> Dict[str, Any]:
    """
    Управление отключением звука

    Args:
        state: True - отключить звук, False - включить
    """
    command = "mute" if state else "unmute"
    return self._send_command(command)

# === Получение информации ===

def get_volume(self) -> Dict[str, Any]:
    """Получение текущего уровня громкости"""
    return self._send_command("volume")
```

```python
def get_state(self) -> Dict[str, Any]:
    """Получение состояния проигрывателя"""
    return self._send_command("state")

def get_status(self) -> Dict[str, Any]:
    """Получение полного статуса устройства"""
    return self._send_command("status")

def get_now_playing(self) -> Dict[str, Any]:
    """Получение информации о текущем треке"""
    return self._send_command("NowPlaying")

# === Управление входами ===

def select_input(self, input_name: str) -> Dict[str, Any]:
    """
    Выбор входного источника

    Args:
        input_name: Имя источника (analog, optical, bluetooth)
    """
    input_map = {
        'analog': 'analog_in',
        'optical': 'optical_in',
        'bluetooth': 'bluetooth'
    }

    mapped_input = input_map.get(input_name.lower(), input_name)
    return self._send_command("select", {"input": mapped_input})

# === Управление воспроизведением с USB ===

def play_usb(self) -> Dict[str, Any]:
    """Воспроизведение музыки с USB-накопителя"""
    # USB-накопитель должен быть отформатирован в FAT32
    return self._send_command("play_usb")

# === Управление сабвуфером ===

def set_subwoofer(self, enabled: bool) -> Dict[str, Any]:
    """
    Включение/выключение выхода сабвуфера

    Args:
```

```
        enabled: True - включен, False - выключен
    """
    state = "1" if enabled else "0"
    return self._send_command("subwoofer", {"state": state})

# === Тон-компенсация ===

def set_bass(self, value: int) -> Dict[str, Any]:
    """
    Установка уровня низких частот

    Args:
        value: Значение от -6 до +6 дБ
    """
    value = max(-6, min(6, value))
    return self._send_command("bass", {"value": value})

def set_treble(self, value: int) -> Dict[str, Any]:
    """
    Установка уровня высоких частот

    Args:
        value: Значение от -6 до +6 дБ
    """
    value = max(-6, min(6, value))
    return self._send_command("treble", {"value": value})

# === Предустановки ===

def press_preset(self, preset_number: int) -> Dict[str, Any]:
    """
    Активация предустановки

    Args:
        preset_number: Номер предустановки (1-6)
    """
    if preset_number < 1 or preset_number > 6:
        return {'error': 'Preset number must be between 1 and 6'}
    return self._send_command(f"preset{preset_number}")

# === Группировка устройств (multi-room до 64 зон) ===

def group_devices(self, slave_ips: List[str]) -> Dict[str, Any]:
    """
```

```
    Группировка нескольких устройств BluOS для многозонного воспроизведения

    Args:
        slave_ips: Список IP-адресов устройств для группировки
    """
    result = {}
    for slave_ip in slave_ips:
        group_url = f"http://{slave_ip}:80/Sync?cmd=group&master={self.ip}"
        try:
            response = requests.get(group_url, timeout=5)
            result[slave_ip] = response.status_code == 200
        except requests.exceptions.RequestException as e:
            result[slave_ip] = False
            result[f"{slave_ip}_error"] = str(e)
    return result

def ungroup_devices(self) -> Dict[str, Any]:
    """Отмена группировки"""
    return self._send_command("ungroup")

# === Настройки устройства ===

def set_device_name(self, name: str) -> Dict[str, Any]:
    """
    Установка имени устройства

    Args:
        name: Новое имя устройства (отображается в приложении)
    """
    return self._send_command("setname", {"name": name})

def get_device_name(self) -> Dict[str, Any]:
    """Получение имени устройства"""
    return self._send_command("getname")
```

5.4. Примеры использования

Пример 1: Базовое управление усилителем

python

```
# Создание экземпляра контроллера для CI 720 V2
controller = NADCI720V2Controller("192.168.1.100")
```

```python
# Установка громкости для зоны
controller.set_volume(65)

# Выбор аналогового входа (подключение к источнику)
controller.select_input("analog")

# Включение сабвуфера при необходимости
controller.set_subwoofer(True)

# Запуск воспроизведения
controller.play()

# Получение информации о состоянии
info = controller.get_status()
print(f"Состояние: {info.get('state', 'unknown')}")
print(f"Громкость: {controller.get_volume().get('volume', '?')}%")
```

Пример 2: Управление многозонной системой

python
```python
class MultiZoneSystem:
    """
    Класс для управления несколькими зонами в профессиональной инсталляции
    """

    def __init__(self, zone_config: dict):
        """
        Args:
            zone_config: Словарь {название_зоны: IP-адрес}
        """
        self.zones = {name: NADCI720V2Controller(ip) for name, ip in zone_config.items()}

    def set_zone_volume(self, zone_name: str, volume: int) -> Dict:
        """Установка громкости в конкретной зоне"""
        if zone_name not in self.zones:
            return {'error': f'Zone {zone_name} not found'}
        return self.zones[zone_name].set_volume(volume)

    def set_all_volume(self, volume: int) -> Dict:
        """Установка громкости во всех зонах"""
        results = {}
        for name, controller in self.zones.items():
            results[name] = controller.set_volume(volume)
```

```
        return results

    def mute_zone(self, zone_name: str, mute: bool = True) -> Dict:
        """Отключение звука в зоне"""
        if zone_name not in self.zones:
            return {'error': f'Zone {zone_name} not found'}
        return self.zones[zone_name].mute(mute)

    def get_all_status(self) -> Dict:
        """Получение статуса всех зон"""
        status = {}
        for name, controller in self.zones.items():
            status[name] = {
                'volume': controller.get_volume(),
                'state': controller.get_state(),
                'now_playing': controller.get_now_playing()
            }
        return status

    def sync_all_inputs(self, input_name: str) -> Dict:
        """Синхронизация входов всех зон"""
        results = {}
        for name, controller in self.zones.items():
            results[name] = controller.select_input(input_name)
        return results

    def group_zones(self, master_zone: str, slave_zones: List[str]) -> Dict:
        """Группировка зон для синхронного воспроизведения"""
        if master_zone not in self.zones:
            return {'error': f'Master zone {master_zone} not found'}

        master_controller = self.zones[master_zone]
        slave_ips = [self.zones[zone].ip for zone in slave_zones if zone in self.zones]

        return master_controller.group_devices(slave_ips)

# Пример использования для коммерческой инсталляции
zones = {
    "lobby": "192.168.1.100",
    "restaurant": "192.168.1.101",
    "bar": "192.168.1.102",
    "corridor_1": "192.168.1.103",
    "corridor_2": "192.168.1.104"
}
```

```
multi_zone = MultiZoneSystem(zones)

# Установка фоновой музыки во всех зонах
multi_zone.set_all_volume(45)

# Группировка ресторана и бара для синхронной музыки
multi_zone.group_zones("restaurant", ["bar"])
```

Пример 3: Интеграция с профессиональной системой управления

python

```
from flask import Flask, request, jsonify

app = Flask(__name__)

# Конфигурация усилителей CI 720 V2
CI720_DEVICES = {
    "zone_a": "192.168.1.100",
    "zone_b": "192.168.1.101",
    "zone_c": "192.168.1.102"
}

# Хранение сессий контроллеров
controllers = {}

def get_controller(zone_name: str):
    """Получение или создание контроллера для зоны"""
    if zone_name not in CI720_DEVICES:
        return None
    if zone_name not in controllers:
        controllers[zone_name] = NADCI720V2Controller(CI720_DEVICES[zone_name])
    return controllers[zone_name]

@app.route('/api/ci720/<zone_name>/status', methods=['GET'])
def get_zone_status(zone_name):
    """Получение статуса зоны"""
    controller = get_controller(zone_name)
    if not controller:
        return jsonify({"error": "Zone not found"}), 404

    status = {
        'zone': zone_name,
        'ip': CI720_DEVICES[zone_name],
```

```
        'model': 'NAD CI 720 V2',
        'volume': controller.get_volume(),
        'state': controller.get_state(),
        'now_playing': controller.get_now_playing()
    }
    return jsonify(status)

@app.route('/api/ci720/<zone_name>/volume', methods=['GET', 'POST'])
def handle_volume(zone_name):
    """Управление громкостью"""
    controller = get_controller(zone_name)
    if not controller:
        return jsonify({"error": "Zone not found"}), 404

    if request.method == 'GET':
        volume = controller.get_volume()
        return jsonify(volume)
    elif request.method == 'POST':
        data = request.get_json()
        if 'level' not in data:
            return jsonify({"error": "Missing 'level' parameter"}), 400
        result = controller.set_volume(data['level'])
        return jsonify(result)

@app.route('/api/ci720/<zone_name>/input', methods=['POST'])
def handle_input(zone_name):
    """Выбор входного источника"""
    controller = get_controller(zone_name)
    if not controller:
        return jsonify({"error": "Zone not found"}), 404

    data = request.get_json()
    if 'source' not in data:
        return jsonify({"error": "Missing 'source' parameter"}), 400

    result = controller.select_input(data['source'])
    return jsonify(result)

@app.route('/api/ci720/zones', methods=['GET'])
def list_zones():
    """Список всех зон"""
    return jsonify(CI720_DEVICES)

@app.route('/api/ci720/preset', methods=['POST'])
```

```python
def apply_preset():
    """
    Применение предустановки ко всем зонам
    Пример: {"preset": 1, "zones": ["zone_a", "zone_b"]}
    """
    data = request.get_json()
    preset = data.get('preset')
    zones = data.get('zones', list(CI720_DEVICES.keys()))

    if not preset or preset < 1 or preset > 6:
        return jsonify({"error": "Invalid preset number (1-6)"}), 400

    results = {}
    for zone in zones:
        controller = get_controller(zone)
        if controller:
            results[zone] = controller.press_preset(preset)
        else:
            results[zone] = {"error": "Zone not found"}

    return jsonify(results)

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5005)
```

Пример 4: Автоматизация сценариев для коммерческих инсталляций

python

```python
class CommercialAutomation:
    """
    Автоматизация сценариев для коммерческих инсталляций
    """

    def __init__(self, ci720_ips: dict):
        """
        Args:
            ci720_ips: Словарь {зона: IP-адрес}
        """
        self.zones = {name: NADCI720V2Controller(ip) for name, ip in ci720_ips.items()}

    def morning_scenario(self):
        """Утренний сценарий открытия (фоновая музыка на низкой громкости)"""
        print("=== УТРЕННИЙ СЦЕНАРИЙ ===")
        for controller in self.zones.values():
```

```
            controller.set_volume(35)
            controller.select_input("analog")
            controller.play()

    def lunch_scenario(self):
        """Обеденный сценарий (средняя громкость, акцент на музыкальный сервис)"""
        print("=== ОБЕДЕННЫЙ СЦЕНАРИЙ ===")
        for controller in self.zones.values():
            controller.set_volume(55)
            # Предполагается, что предустановка 1 настроена на музыкальный плейлист
            controller.press_preset(1)

    def evening_scenario(self):
        """Вечерний сценарий (высокая громкость, атмосферная музыка)"""
        print("=== ВЕЧЕРНИЙ СЦЕНАРИЙ ===")
        for controller in self.zones.values():
            controller.set_volume(70)
            controller.press_preset(2)

    def closing_scenario(self):
        """Сценарий закрытия (выключение всех зон)"""
        print("=== ЗАКРЫТИЕ ===")
        for controller in self.zones.values():
            controller.pause()

    def zone_status_report(self) -> Dict:
        """Сводный отчет о состоянии всех зон"""
        report = {}
        for name, controller in self.zones.items():
            status = controller.get_status()
            report[name] = {
                'connected': True,
                'volume': controller.get_volume().get('volume', '?'),
                'input': status.get('input_name', 'unknown'),
                'state': status.get('state', 'unknown')
            }
        return report

# Пример использования для ресторана/бара
commercial = CommercialAutomation({
    "main_hall": "192.168.1.100",
    "bar": "192.168.1.101",
    "terrace": "192.168.1.102"
})
```

```
# Запуск утреннего сценария
commercial.morning_scenario()

# Получение отчета о состоянии
report = commercial.zone_status_report()
print(json.dumps(report, indent=2, ensure_ascii=False))
```

Пример 5: Веб-интерфейс и мониторинг

```
python
import time
import csv
from datetime import datetime
from typing import Dict

class CI720Monitor:
    """
    Мониторинг состояния усилителя для профессионального обслуживания
    """

    def __init__(self, ip_address: str, log_file: str = "ci720_monitor.csv"):
        self.controller = NADCI720V2Controller(ip_address)
        self.log_file = log_file

    def get_full_state(self) -> Dict:
        """Получение полного состояния устройства"""
        state = {
            'timestamp': datetime.now().isoformat(),
            'volume': self.controller.get_volume().get('volume', 'unknown'),
            'state': self.controller.get_state().get('state', 'unknown'),
            'input': self.controller.get_status().get('input_name', 'unknown')
        }

        # Получение информации о текущем треке
        now_playing = self.controller.get_now_playing()
        if 'title' in now_playing:
            state['title'] = now_playing.get('title', '')
            state['artist'] = now_playing.get('artist', '')
            state['album'] = now_playing.get('album', '')

        return state

    def log_to_csv(self, state: Dict):
```

```
        """Логирование состояния в CSV файл"""
        file_exists = False
        try:
            with open(self.log_file, 'r'):
                file_exists = True
        except FileNotFoundError:
            pass

        with open(self.log_file, 'a', newline='', encoding='utf-8') as f:
            writer = csv.DictWriter(f, fieldnames=state.keys())
            if not file_exists:
                writer.writeheader()
            writer.writerow(state)

    def monitor_continuously(self, interval_seconds: int = 60):
        """
        Непрерывный мониторинг для профессионального обслуживания

        Args:
            interval_seconds: Интервал между замерами в секундах
        """
        print(f"НАЧАЛО НЕПРЕРЫВНОГО МОНИТОРИНГА")
        print(f"Интервал: {interval_seconds} сек")
        print("=" * 60)

        try:
            while True:
                state = self.get_full_state()
                self.log_to_csv(state)
                print(f"[{state['timestamp']}] "
                      f"Громкость: {state['volume']}%, "
                      f"Состояние: {state['state']}, "
                      f"Вход: {state['input']}")
                time.sleep(interval_seconds)
        except KeyboardInterrupt:
            print("\nМОНИТОРИНГ ПРЕРВАН")
        finally:
            print(f"Лог сохранен в: {self.log_file}")

# Пример использования
monitor = CI720Monitor("192.168.1.100")
# monitor.monitor_continuously(interval_seconds=60)
```

5.5. Справочник команд BluOS API для CI 720 V2

Команда	Описание	Параметры
volume	Управление громкостью	level (0-100)
mute / unmute	Отключение/включение звука	-
play / pause / stop	Управление воспроизведением	-
skip / back	Следующий/предыдущий трек	-
state	Состояние проигрывателя	-
status	Полный статус	-
NowPlaying	Информация о текущем треке	-
select	Выбор входа	input (analog_in, optical_in, bluetooth)
subwoofer	Сабвуфер	state (0/1)
bass	Низкие частоты	value (-6 to +6 дБ)
treble	Высокие частоты	value (-6 to +6 дБ)
preset1 - preset6	Предустановки (1-6)	-
group	Группировка устройств	master (IP мастер-устройства)
ungroup	Отмена группировки	-
setname	Установка имени устройства	name
getname	Получение имени устройства	-

6. УПРАВЛЕНИЕ ЧЕРЕЗ BLUOS

6.1. BluOS App

Усилитель CI 720 V2 полностью интегрирован в экосистему BluOS и управляется через **BluOS App** .

Поддерживаемые платформы:

- iOS (iPhone/iPad)
- Android
- Windows
- macOS

Основные функции приложения:

- Управление воспроизведением (play/pause/skip)
- Регулировка громкости
- Выбор входных источников
- Настройка многозонной системы (до 64 зон)
- Доступ к музыкальным сервисам
- Воспроизведение локальной медиатеки

6.2. Поддерживаемые музыкальные сервисы и форматы

Стриминговые сервисы :

Сервис	**Поддержка**
Spotify	✓ (Spotify Connect)
TIDAL	✓ (TIDAL Connect)

Сервис	Поддержка
Qobuz	✓
Deezer	✓
Amazon Music HD	✓
Napster	✓
HighResAudio	✓
KKBox	✓

Интернет-радио :

- TuneIn Radio
- iHeartRadio
- Calm Radio
- Radio Paradise

Поддерживаемые аудиоформаты :

Формат	Поддержка
FLAC	До 24 бит/192 кГц
ALAC	До 24 бит/192 кГц
WAV	До 24 бит/192 кГц
AIFF	До 24 бит/192 кГц
MQA	Полная поддержка

Формат	Поддержка
MP3	До 320 кбит/с
AAC	До 320 кбит/с
OGG Vorbis	Да
WMA	Да

6.3. Apple AirPlay 2

Усилитель поддерживает **Apple AirPlay 2** для прямой потоковой передачи с устройств Apple .

Возможности:

- Потоковая передача с iPhone, iPad, Mac
- Синхронное воспроизведение на нескольких AirPlay 2-совместимых устройствах
- Управление через Центр управления iOS

6.4. USB-воспроизведение

Порт USB Type-A поддерживает воспроизведение музыки с внешних накопителей .

Требования:

- Форматирование: FAT32
- Поддерживаемые форматы: MP3, AAC, WMA, OGG, FLAC, ALAC, WAV, AIFF, MQA
- Максимальное разрешение: 24 бит/192 кГц

---

7. ПРОФЕССИОНАЛЬНЫЕ ФУНКЦИИ

7.1. Профессиональные системные интеграции

Усилитель NAD CI 720 V2 сертифицирован для интеграции с ведущими профессиональными системами управления :

Система	**Тип интеграции**
Control4	IP/API
Crestron	IP/API, RS232
RTI	IP/API
URC	IP/API
AMX	IP/API
ELAN	IP/API
Lutron	IP/API
PUSH	IP/API
KNX	IP/API

Доступна полная API документация :

- Полная спецификация BluOS Custom Integration API
- Примеры кода для основных систем управления
- Документация по IP командам

7.2. Активное охлаждение (Thermostatic Fan)

Усилитель оснащен **активным термостатическим вентилятором** :

Параметр	Значение
Тип	Активное охлаждение
Уровень шума	25 дБ
Управление	Термостатическое (включение по температуре)

Примечание: Наличие вентилятора делает усилитель непригодным для размещения в жилых комнатах. Устройство предназначено для установки в технические стойки и шкафы .

7.3. Многозонное расширение

CI 720 V2 является однозонным устройством, но может быть интегрирован в **систему до 64 зон**с использованием нескольких усилителей и других BluOS-совместимых компонентов .

Архитектура многозонной системы:

- Каждый CI 720 V2 — одна независимая зона
- До 64 зон в единой системе
- Управление через единый интерфейс BluOS App
- Возможность группировки зон для синхронного воспроизведения

7.4. Гибридное усиление HybridDigital

Усилитель использует технологию **HybridDigital** — фирменную реализацию усилителей класса D от NAD .

Преимущества:

- Высокая эффективность (низкое энергопотребление)
- Минимальные искажения (<0,005%)
- Высокое соотношение сигнал/шум (-110 дБА)

- Компактный форм-фактор

7.5. Управление через IR

Усилитель оснащен входом IR IN (3,5 мм) для подключения внешнего ИК-приемника . Это позволяет интегрировать усилитель в системы с централизованным ИК-управлением.

7.6. Roon Ready

Усилитель сертифицирован как **Roon Ready**, что позволяет использовать его в профессиональных аудиосистемах с платформой Roon .

---

8. ПОИСК И УСТРАНЕНИЕ НЕИСПРАВНОСТЕЙ

Проблема	**Решение**
Усилитель не включается	Проверьте оба кабеля питания. Убедитесь, что розетки работают.
Нет звука	Проверьте акустические подключения (Phoenix терминал). Убедитесь, что выбран правильный вход. Проверьте, что громкость не на минимуме и не включен Mute.
Усилитель не подключается к сети	CI 720 V2 не поддерживает Wi-Fi! Используйте только проводное подключение Ethernet . Проверьте кабель Ethernet. Убедитесь, что DHCP включен.
Приложение не обнаруживает устройство	Убедитесь, что усилитель и устройство управления находятся в одной сети. Проверьте, что Ethernet подключен. Перезапустите приложение BluOS.

Проблема	Решение
Не работает USB-порт	Убедитесь, что USB-накопитель отформатирован в FAT32 . Проверьте поддерживаемые форматы файлов.
Греется/шум от вентилятора	Усилитель оснащен активным вентилятором — это нормально для профессиональных инсталляций . Обеспечьте вентиляционный зазор 10 см.
Нет звука с оптического входа	Проверьте, что источник выводит сигнал. Убедитесь, что выбран вход Optical. Проверьте качество оптического кабеля.
API не отвечает	Проверьте IP-адрес устройства. Убедитесь, что устройство находится в той же подсети. Порт 80 должен быть открыт.
Плохая фиксация акустических кабелей	Используйте моножильные (solid core) кабели. Многожильные кабели ненадежно фиксируются в Phoenix-терминалах .

---

9. ТЕХНИЧЕСКОЕ ОБСЛУЖИВАНИЕ

- **Чистка:** Отключите от сети перед чисткой. Используйте только сухую мягкую ткань. Не используйте жидкости или аэрозоли .
- **Вентиляция:** Регулярно проверяйте, чтобы вентиляционные отверстия не были заблокированы пылью. Усилитель оснащен активным вентилятором, который может притягивать пыль.
- **Обновление ПО:** Прошивка обновляется автоматически через BluOS App при подключении к Интернету. Белая вспышка LED индикатора указывает на доступность обновления .
- **Транспортировка:** Используйте оригинальную упаковку — это самый безопасный контейнер для перемещения усилителя .
- **Сервис:** При повреждениях обращайтесь к авторизованному сервисному центру NAD .

ПРИЛОЖЕНИЕ: ПОЛНЫЕ ТЕХНИЧЕСКИЕ СПЕЦИФИКАЦИИ

Параметр	Значение
Модель	NAD CI 720 V2
Тип	Сетевой стереоусилитель для инсталляций
Цвет	Черный (Black)
Технология усиления	HybridDigital (класс D)
Выходная мощность (RMS)	2 x 60 Вт (8/4 Ом)
Пиковая мощность	2 x 165 Вт
THD+N	<0,005%
Соотношение сигнал/шум (SNR)	-110 дБА
Частотный диапазон	20 Гц - 20 кГц (±0,5 дБ)
Поддерживаемые разрешения	24 бит / 192 кГц
BluOS зоны	До 64 зон в системе
Сетевое подключение	Gigabit Ethernet (RJ45) — HET Wi-Fi
Стриминговые протоколы	AirPlay 2, Spotify Connect, TIDAL Connect, Roon Ready
Потребление (ожидание)	6 Вт (с сетью), 0,5 Вт (глубокий сон)
Максимальное потребление	150 Вт
Тепловыделение	4,9 Вт/ч / 16,7 BTU/ч
Уровень шума вентилятора	25 дБ

Параметр	Значение
Габариты (Ш × В × Г)	70 × 120 × 350 мм
Высота в стойке	3U
Вес	1,6 кг
Артикул	CI720V2