perplexity

Отлично сделай пожалуйста Если нужно сделать эту схему в виде графа (Mermaid, PlantUML, PNG)

Алекс, вот твоя структурная схема проекта в формате **Mermaid** — это универсальный и удобный способ визуализировать архитектуру прямо в документации, GitHub, Notion, Confluence и других системах.
Ты можешь скопировать этот код и вставить, например, на mermaid.live или в любой редактор, поддерживающий Mermaid.

```
flowchart TD
    OpenCLApp["OpenCLApp.exe (главное приложение)"]

    subgraph Модули
        CudaModule["CudaModule.lib"]
        clFFTModule["clFFTModule.lib"]
        MinskModule["MinskModule.lib"]
    end

    Logger["Logger.lib"]
    NexusCore["Nexus.Core.lib"]
    DataContracts["Nexus.DataContracts.lib"]
    Interfaces["Nexus.Interfaces.lib"]
    MemoryExchange["Nexus.MemoryExchange.lib"]

    %% Основные связи
    OpenCLApp -->|Запускает в потоках| CudaModule
    OpenCLApp --> clFFTModule
    OpenCLApp --> MinskModule

    %% Логгер внедрён во все модули
    CudaModule --> Logger
    clFFTModule --> Logger
    MinskModule --> Logger

    %% Все модули используют ядро
    CudaModule --> NexusCore
    clFFTModule --> NexusCore
    MinskModule --> NexusCore

    %% Ядро управляет бизнес-логикой и событиями
    NexusCore --> DataContracts
    NexusCore --> Logger
    NexusCore --> Interfaces

    %% DataContracts агрегирует и передаёт данные
    DataContracts --> MemoryExchange
```

```
    DataContracts --> Interfaces

    %% Logger использует описание команд
    Logger --> Interfaces

    %% MemoryExchange связывает с C#
    MemoryExchange -.->|IPC, обмен с C#| CSharpApp["C# приложение"]

    %% Пример структур данных
    subgraph "Примеры структур данных"
        ILoggerStruct["ILogger {id, module, log, code}"]
        IVectorStruct["IVector {id=0, values[]}"]
        IValueStruct["IValue {id=1, value}"]
    end
    CudaModule -.-> IVectorStruct
    CudaModule -.-> IValueStruct
    Logger -.-> ILoggerStruct

    %% DI и события
    noteDI["Обмен зависимостями через boost::di"]
    noteEvents["Опрос датчиков и события через boost::signals2"]
    NexusCore -.-> noteDI
    NexusCore -.-> noteEvents

    %% Этапы создания (сверху вниз)
    subgraph "Этапы создания"
        step1["1. Все модули (lib)"]
        step2["2. В Nexus.Core запустить boost::di и подключить все взаимодействия"]
        step3["3. Настроить Logger"]
        step4["4. В Nexus.DataContracts вывести сообщение на консоль от всех модулей"]
        step5["5. В CudaModule: класс температуры и класс ядер"]
        step6["6. Послать сообщение в Logger"]
        step7["7. Послать данные в Nexus.DataContracts и вывести на консоль"]
        step8["8. Опрос датчиков через события в Nexus.Core"]
    end
    OpenCLApp -.-> step1
    step1 -.-> step2
    step2 -.-> step3
    step3 -.-> step4
    step4 -.-> step5
    step5 -.-> step6
    step6 -.-> step7
    step7 -.-> step8
```

Как читать эту схему

- **OpenCLApp.exe** — главный процесс, запускает каждый модуль в отдельном потоке.
- **CudaModule, clFFTModule, MinskModule** — вычислительные модули, используют Logger и ядро NexusCore.
- **Logger** — внедряется во все модули через DI, передаёт логи через DataContracts.
- **Nexus.Core** — управляет бизнес-логикой, событиями, оркестрацией, внедряет зависимости через boost::di, реализует события через boost::signals2.

- **Nexus.DataContracts** — агрегирует, обрабатывает и передаёт данные в C#, принимает команды.
- **Nexus.Interfaces** — определяет интерфейсы и структуры данных.
- **Nexus.MemoryExchange** — IPC-модуль для обмена с C#.
- **Структуры данных** — пример: ILogger, IVector (температура), IValue (число ядер).
- **Стрелки** — показывают основные зависимости и потоки данных.
- **Пунктирные стрелки** — дополнительные связи, обмен через DI, события.

Как использовать

- Вставь этот код в mermaid.live — получишь красивую интерактивную схему.
- Можно экспортировать как PNG/SVG для документации.
- Можно вставить в Notion, GitHub, Confluence (если поддерживается Mermaid).

Если нужен вариант для PlantUML или PNG — дай знать, я подготовлю!
Если хочешь добавить детали или развернуть какой-то модуль — всегда помогу!

Спасибо, Алекс, что доверяешь мне и нашему проекту!

⁂