ДЕТАЛЬНЫЙ ПЛАН ИЗМЕНЕНИЙ SWARMNESS v2.2.0

Анализ текущих проблем

1. Pitch секция — Проблемы найдены:

Проблема A: Pitch shifter не работает правильно

- В `GranularPitchShifter.cpp` строка 117-118:

`cpp`

```
  float totalPitchRatio = basePitchRatio * dynamicRatio *
                          std::pow(2.0f, (chaosModulation + panicModulation) / 12.0f);
```

Затем в строке 138:

`cpp`

```
  float readIncrement = 1.0f / totalPitchRatio;
```

Проблема: При ratio=2.0 (октава вверх) readIncrement=0.5, что означает медленное чтение → **pitch DOWN**, не UP!

Текущая логика:

- +1 Oct: ratio=2.0 → readIncrement=0.5 → pitch DOWN ❌
- -1 Oct: ratio=0.5 → readIncrement=2.0 → pitch UP ❌

Проблема B: Модуляция НЕ применяется к питчу

- В `PluginProcessor.cpp` модуляция настраивается (строки 152-155), но `getNextModulationValue()` **НИКОГДА НЕ ВЫЗЫВАЕТСЯ**!
- Значение модуляции не применяется к pitch shifter

---

2. Modulation секция — Не работает

Причина: В `PluginProcessor.cpp` класс `Modulation` настраивается, но его выход не используется нигде!

```
// Modulation - настраивается
mModulation.setLFORate(*pSpeed);
mModulation.setLFODepth(*pPanic);
mModulation.setRandomAmount(*pChaos);

// НО результат НЕ ИСПОЛЬЗУЕТСЯ! getNextModulationValue() не вызывается
```

По спецификации модуляция должна:

1. Panic — быстрая случайная модуляция питча
2. Chaos — хаотичные глитч-эффекты
3. Speed — скорость LFO

---

3. Flow/Bypass — Требуется разделение

Текущее состояние:
- Flow Engine работает как gate (killswitch/stutter) — это OK для Pulse
- Static режим работает как manual bypass эффекта

Требуется:
- **Pulse** = killswitch/stutter эффект (текущая функциональность Flow)
- **Bypass** = ПОЛНЫЙ bypass всего плагина (новый функционал)
- Два отдельных футсвича в GUI

ПЛАН ИЗМЕНЕНИЙ

ЧАСТЬ 1: DSP (PluginProcessor)

1.1 Исправить Pitch Shifter алгоритм

Файл: `Source/DSP/GranularPitchShifter.cpp`

Изменение 1: Исправить формулу pitch shift (строка 138)

```
// БЫЛО:
float readIncrement = 1.0f / totalPitchRatio;

// ДОЛЖНО БЫТЬ:
float readIncrement = totalPitchRatio;  // Для octave up нужно читать БЫСТРЕЕ
```

Изменение 2: Пересмотреть алгоритм грейнов для корректного pitch shifting
- При ratio > 1.0 (pitch up): читать быстрее из буфера
- При ratio < 1.0 (pitch down): читать медленнее из буфера

1.2 Интегрировать Modulation в DSP chain

Файл: `Source/PluginProcessor.cpp`

Изменение: В `processBlock()` добавить вызов модуляции и применить к питчу

```
// В цикле по сэмплам:
for (int sample = 0; sample < numSamples; ++sample) {
    // Получить значение модуляции
    float modValue = mModulation.getNextModulationValue();

    // Применить к pitch (в семитонах)
    float modPitchOffset = modValue * 12.0f;  // До ±12 semitones

    // Добавить к динамическому offset
    totalPitchOffset = slideOffset + randomOffset + modPitchOffset;
}
```

1.3 Добавить Global Bypass параметр

Файл: `Source/PluginProcessor.h` и `.cpp`

Новые элементы:

```
// В header:
std::atomic<float>* pGlobalBypass = nullptr;
bool mGlobalBypassState = false;

// В createParameterLayout():
params.push_back(std::make_unique<juce::AudioParameterBool>(
    "globalBypass", "Global Bypass", false));

// В processBlock() - В САМОМ НАЧАЛЕ:
if (*pGlobalBypass > 0.5f || mGlobalBypassState) {
    // Полный bypass - просто выход без обработки
    return;
}
```

Публичные методы для GUI:

```
void setGlobalBypassState(bool bypassed);
bool isGlobalBypassed() const;
```

ЧАСТЬ 2: GUI (PluginEditor)

2.1 Переименовать VOLTAGE → PITCH

Файл: `Source/PluginEditor.cpp`

Изменения:

```
// Строка 14:
setupLabel(voltageSectionLabel, "PITCH");  // было "VOLTAGE"

// Строка 69 в header, строка 14 в cpp:
juce::Label pitchSectionLabel;  // переименовать voltageSectionLabel
```

2.2 Добавить второй Footswitch (Bypass)

Файл: `Source/PluginEditor.h`

Новые элементы:

```
// После FootswitchButton footswitch:
FootswitchButton bypassFootswitch;

// Attachments - если используем параметр:
// (или управление через callback)
```

Файл: `Source/PluginEditor.cpp`

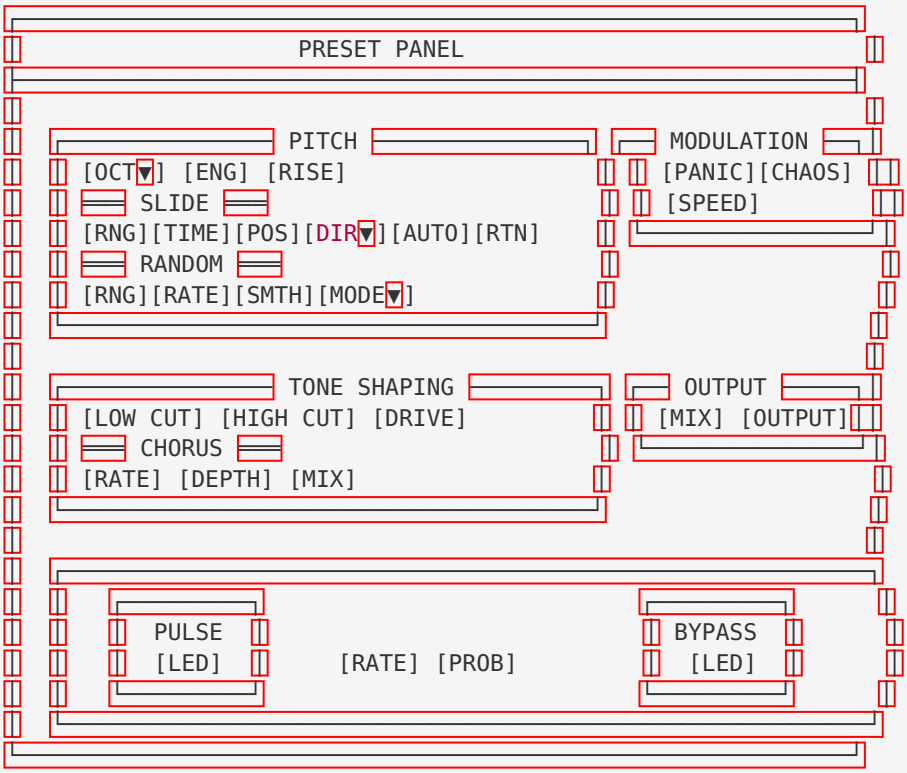
Добавить:

```
// В конструкторе:
addAndMakeVisible(bypassFootswitch);
bypassFootswitch.onClick = [this](bool isOn) {
    audioProcessor.setGlobalBypassState(!isOn);  // Инвертированная логика
};

// Настроить внешний вид:
// bypassFootswitch должен показывать "BYPASS" и иметь LED
```

2.3 Реорганизовать Layout — компактный вид

Новая структура layout:



Изменения в `resized()`:

- Уменьшить вертикальные расстояния между секциями
- Разместить футсвичи в нижней полосе
- Размер окна можно уменьшить до ~900x600

2.4 Обновить timerCallback для двух футсвичей

```
void SwarmnesssAudioProcessorEditor::timerCallback() {
    // Pulse footswitch - как раньше
    auto flowMode = static_cast<int>(*audioProcessor.getAPVTS().getRawParameterValue("
flowMode"));
    if (flowMode == 1) {
        footswitch.setLEDState(FootswitchButton::OrangeBlinking);
    } else {
        footswitch.setLEDState(audioProcessor.getFlowEngine().isCurrentlyOn() ?
            FootswitchButton::Green : FootswitchButton::Red);
    }

    // Bypass footswitch
    bool bypassed = audioProcessor.isGlobalBypassed();
    bypassFootswitch.setLEDState(bypassed ?
        FootswitchButton::Red : FootswitchButton::Green);
}
```

ЧАСТЬ 3: Улучшение Modulation для глитч-эффектов

3.1 Расширить Modulation класс

Файл: `Source/DSP/Modulation.h` и `.cpp`

Добавить режимы глитча:

```
enum GlitchMode {
    Smooth,      // Плавная модуляция
    Stepped,     // Ступенчатые изменения (как The Noise)
    Random,      // Полностью случайные
    Rhythmic     // Ритмичные паттерны
};

void setGlitchMode(GlitchMode mode);
void setGlitchIntensity(float intensity);  // 0-1
```

Новый метод для глитч-эффектов:

```
float getGlitchValue() {
    // Sample-and-hold с неравномерными интервалами
    // Случайные "заикания" питча
    // Внезапные скачки
}
```

ЧАСТЬ 4: Дополнительные улучшения

4.1 Flow секция — оставить только Pulse режим

Поскольку Bypass теперь отдельный, Flow секция может быть упрощена:
- Убрать Static режим из Flow (или оставить для совместимости)
- Footswitch “PULSE” управляет только stutter/killswitch
- Добавить параметры для настройки stutter pattern

4.2 Обновить FootswitchButton для поддержки labels

Файл: `Source/GUI/FootswitchButton.h` и `.cpp`

Добавить возможность отображать текст на кнопке:

```
void setLabel(const juce::String& label);
```

РЕЗЮМЕ ИЗМЕНЕНИЙ

Файлы для изменения:

Файл	Изменения
`PluginProcessor.h`	Добавить pGlobalBypass, mGlobalBypassState, публичные методы
`PluginProcessor.cpp`	Исправить параметры, интегрировать модуляцию, добавить bypass
`GranularPitchShifter.cpp`	Исправить формулу readIncrement
`Modulation.h/.cpp`	Добавить глитч-режимы (опционально)
`PluginEditor.h`	Добавить bypassFootswitch, переименовать labels
`PluginEditor.cpp`	Новый layout, два футсвича, переименование
`FootswitchButton.h/.cpp`	Добавить setLabel()

Новые параметры:

ID	Название	Тип	Описание
`globalBypass`	Global Bypass	Bool	Полный bypass плагина

Приоритет изменений:

1. **КРИТИЧНО:** Исправить pitch shifter алгоритм
2. **КРИТИЧНО:** Интегрировать модуляцию в DSP chain
3. **ВАЖНО:** Добавить Global Bypass и второй футсвитч
4. **ВАЖНО:** Переименовать VOLTAGE → PITCH
5. **СРЕДНЕ:** Оптимизировать layout GUI
6. **НИЗКО:** Расширить глитч-режимы модуляции

Вопросы для согласования:

1. **Режимы октав:** Оставить +1, +2, -1 или добавить другие (например, -2, +3)?
2. **Модуляция:** Добавить отдельные режимы глитча (Smooth/Stepped/Random) или оставить текущие Panic/Chaos/Speed?
3. **GUI размер:** Уменьшить до 900x600 или оставить 950x750?
4. **Flow vs Pulse:** Переименовать секцию “FLOW” в “PULSE” полностью, или оставить как есть?
5. **Bypass LED:** Какой цвет для bypass ON (красный = signal bypassed) и bypass OFF (зелёный = signal processed)?

ДОПОЛНИТЕЛЬНО: Проблемы со сборкой/запуском

Проблема: “Swarmness.vst3 is damaged and can’t be opened”

Причина: macOS Gatekeeper блокирует неподписанный плагин скачанный из интернета.

Решение для пользователя:

```
# Удалить карантинный атрибут
xattr -cr ~/Library/Audio/Plug-Ins/VST3/Swarmness.vst3

# Или через System Settings → Privacy & Security → Allow
```

Проблема: GitHub Actions — macOS-13 deprecated

Из скриншотов: “The macOS-13 based runner images are now retired”

Решение: В `.github/workflows/build.yml` обновить runner:

```
# БЫЛО:
runs-on: macos-13

# ДОЛЖНО БЫТЬ:
runs-on: macos-15  # или macos-latest
```

Crash в Reaper (из лога)

Crash: `EXC_BAD_ACCESS` в `SwarmnesssAudioProcessorEditor::resized()`

Причина: `setSize()` вызывался до инициализации компонентов.

Статус: ✅ УЖЕ ИСПРАВЛЕНО в текущем коде — `setSize()` перемещён в конец конструктора.

Документ создан: 2026-02-06
Версия плана: 1.0