

Intuition Subsynth

Professional Hardware Synthesiser

User Manual

Version: 1.7

Date: November 2025

Model: Intuition Subsynth 320-Channel Synthesiser for Raspberry Pi 2B/3/4/400/5/500/Zero 2W (64-bit)

Front Matter

Safety Information

Intuition Subsynth is distributed solely as software. You provide the Raspberry Pi hardware, power supply, enclosure, and all peripherals. As the system integrator you are responsible for following the official safety, handling, and regulatory guidance supplied by Raspberry Pi Ltd for the specific board you own. This manual only documents the Subsynth software and does not constitute a declaration of conformity for any physical product.

Hardware Responsibilities

- Read the official safety/compliance sheet included with your Raspberry Pi model and obey all warnings (power supply ratings, ventilation, ESD handling, and disposal requirements).
- Use only certified power supplies and cables that match your board revision and local electrical regulations.
- Provide adequate airflow/heatsinking appropriate to the Pi model and ambient conditions.
- Keep liquids, conductive materials, and high-intensity light sources away from the powered board.
- Ensure any connected accessories (USB hubs, MIDI interfaces, DIN breakouts) meet the regulations of your jurisdiction.

For the latest Raspberry Pi safety and compliance documentation, visit <https://www.raspberrypi.com/documentation/>. All regulatory compliance questions for the underlying hardware should be directed to Raspberry Pi Ltd or your hardware supplier.

2. Important Information

Copyright Notice

© 2025 Zayn Otley. All rights reserved.

No part of this publication may be reproduced, distributed, or transmitted in any form or by any means without the prior written permission of the manufacturer.

Support Contact Details

- **Website:** <https://intuitionsubsynth.com>
 - **LinkedIn:** <https://www.linkedin.com/company/intuitionsubsynth>
-

3. Table of Contents

Part I: Getting Started

- Introduction
 - What is Intuition Subsynth?
 - Key Features Overview
 - Delivery Format & Hardware Requirements
 - Ordering and License Binding
 - Hardware Locking & Demo Mode
 - Package Contents
 - Panel Overview
- Quick Start Guide
 - First-Time Setup Checklist
 - Your First 10 Minutes with Subsynth
- Connections and Setup
 - Rear Panel Connections
 - MIDI IN Port (5-pin DIN)
 - USB-MIDI via Raspberry Pi USB-A Ports
 - Audio Outputs (HiFiBerry DAC2 Pro or HDMI)
 - USB-C Power Input
 - Ground Loop Prevention
 - Cable Recommendations
- Panel Description
 - Front Panel Elements
 - Rear Panel Elements
 - Side Panels
 - Thermal Management

Part II: Playing the Instrument

- Basic Operation
 - Powering On/Off Procedures
 - Sound Bank Selection
 - Programme Change Messages
 - Understanding Polyphony (320 Channels by Wave Type)
 - Voice Allocation and Stealing Behaviour
 - Using Sustain Pedal
 - Pitch Bend and Modulation Wheel
- Understanding the Sound Engine
 - 320-Channel Architecture Explained for Musicians
 - Five Waveform Types
 - ADSR Envelopes: Attack, Decay, Sustain, Release
 - Filter Types: Low-Pass, High-Pass, Band-Pass
 - Effects Overview
 - Layered Patches: Dual-Waveform Sounds
 - Real-Time Performance Controls

Part III: Sound Programming

- Synthesis Parameters
 - Oscillator Parameters
 - Hard Sync
 - Ring Modulation

- Modulation and LFOs
 - Vibrato System
 - Tremolo System
 - LFO System (Low-Frequency Oscillator)
 - Filter Sweeps with Envelope Modulation
- Effects
 - Per-Channel Effects Overview
 - Master Effects Overview
 - Effect Routing and Signal Flow
 - Creative Effect Combinations

Part IV: MIDI Implementation

- MIDI Basics
 - What is MIDI?
 - MIDI Connections
 - MIDI Channels
 - MIDI Message Types
 - General MIDI Level 1 Compliance
- Standard MIDI Controllers
 - Channel Volume (CC#7)
 - Pan Position (CC#10)
 - Expression (CC#11)
 - Sustain Pedal (CC#64)
 - Complete MIDI CC Reference
- Advanced NRPN Control
 - What are NRPNs?
 - NRPN Addressing Formula
 - Per-Channel Parameters (1000-5159)
 - Global Parameters (0-99)
 - Sending NRPN Messages
- MIDI Implementation Chart
 - Recognised MIDI Messages
 - Transmitted MIDI Messages
 - Controller Assignments

Part V: Sound Banks

- Factory Patchsets
 - SIDney: Commodore 64 MOS 8580 SID
 - AYYMe: AY-3-8910 / YM2149 PSG
 - Rawland: Roland-Inspired Synthesis
 - PixelaTED: Commodore TED 7360/8360
 - EenBeetje: ZX Spectrum 1-Bit Beeper

Part VI: Reference

- Technical Specifications
 - Hardware Specifications
 - Audio Specifications
 - Synthesis Engine Specifications

- MIDI Specifications
 - Power Specifications
 - Performance Optimisation & Real-Time Features
 - Troubleshooting Guide
 - No Sound Output
 - MIDI Not Working
 - Demo Mode Activated
 - Distorted or Clipped Audio
 - Unexpected Voice Stealing
 - USB Power Issues
 - Thermal/Performance Issues
 - Factory Reset Procedures
 - Appendices
 - Appendix A: MIDI Controller Quick Reference
 - Appendix B: NRPN Parameter Quick Reference
 - Appendix C: General MIDI Instrument Map
 - Appendix D: General MIDI Drum Map
 - Appendix E: Raspberry Pi Safety References
 - Index
-

Part I: Getting Started

4. Introduction

What is Intuition Subsynth?

The Intuition Subsynth is a professional 320-channel real-time hardware synthesiser that brings authentic retro computer music synthesis to the modern studio. Combining the character of legendary sound chips from the 1980s with professional digital signal processing, Intuition Subsynth delivers unparalleled sonic authenticity with the reliability and precision of modern audio technology. The current release is delivered as personalised Raspberry Pi OS images for Raspberry Pi 2 Model B Rev 1.2, Raspberry Pi 3 (3B/3B+/3A+), Raspberry Pi 4, Raspberry Pi 400, Raspberry Pi 5, Raspberry Pi 500, and Raspberry Pi Zero 2 W (64-bit) so you can deploy the full instrument on the board that suits your studio whilst keeping the sound engine and protection layers identical.

Key Features Overview

Synthesis Engine:

- 320 independent synthesis channels organised by waveform type
- Five classic waveforms: Square (with PWM), Triangle, Sine, Noise (white/periodic/metallic), Sawtooth
- Complete ADSR envelopes with multiple curve shapes
- Professional anti-aliasing for pristine audio quality

MIDI Capabilities:

- Complete **General MIDI Level 2** compliance
- All 128 standard instruments plus complete drum kit
- **Dual percussion channels** (Channels 10 & 11) for simultaneous drum playback
- **Bank 0x78 dynamic rhythm** - convert any channel to percussion mode
- **System Exclusive (SysEx) support** - GM/GM2 System On, Master Volume, Master Tuning
- Advanced MIDI control with 4360+ NRPN parameters
- Sample-accurate MIDI timing for perfect synchronisation

- Intelligent voice allocation with priority-based stealing

Audio Performance:

- Professional low-latency operation (sub-3ms)
- Multiple sample rate support (44.1kHz to 192kHz)
- Stereo line outputs with professional audio quality
- Master effects chain: reverb, chorus, delay, compression

Sound Banks:

- Five carefully crafted factory patchsets
- SIDney: Commodore 64 MOS 8580 SID emulation (default)
- AYYMe: Amstrad CPC/ZX Spectrum 128K/MSX/Atari ST AY-3-8910/YM2149 PSG chip
- Rawland: Roland-inspired (303/808/909)
- PixelaTED: Commodore TED 7360/8360 chip
- EenBeetje: ZX Spectrum 1-bit beeper

Delivery Format & Hardware Requirements

Intuition Subsynth now ships as personalised Raspberry Pi OS microSD images instead of a pre-assembled hardware unit. You install the provided image on your own supported Raspberry Pi and pair it with the recommended audio hardware to create the finished instrument. This approach keeps the professional sound engine identical to the original hardware release whilst allowing you to build on readily available components.

Supported Raspberry Pi Models:

- Raspberry Pi 2 Model B Rev 1.2 (1GB RAM, ARMv8 64-bit capable, 64-bit OS image supplied)
- Raspberry Pi 3 Model B / 3B+ / 3A+ (512MB-1GB RAM, 64-bit OS image supplied)
- Raspberry Pi 4 Model B (all RAM variants: 1GB, 2GB, 4GB, 8GB)
- Raspberry Pi 400
- Raspberry Pi 5 (all RAM variants: 4GB, 8GB, 16GB)
- Raspberry Pi 500
- Raspberry Pi Zero 2 W (512MB RAM, 64-bit mode only, USB hub required for MIDI + power; feature-restricted, see Performance Optimisations section)

Every order is cryptographically tied to the exact hardware serial number of the Raspberry Pi you specify at checkout. Each board requires its own licence/image.

Factory-Supplied (Digital) Items:

- Individually signed Subsynth image download tied to your Raspberry Pi hardware serial number
- Latest factory patchsets, MIDI definition files, and checksum manifest
- PDF copies of the end-user manual, quick start guide, and support information

Customer-Supplied Hardware:

- One of the supported Raspberry Pi models listed above, running the matching Subsynth image
- HiFiBerry DAC2 Pro (recommended; line-level RCA outputs; auto-detected on first boot) or HDMI display/receiver with audio support (fallback)
- High-endurance UHS-I A2 microSD card, 8GB minimum (16GB+ recommended for rapid updates)
- Official Raspberry Pi power supply for your model (Pi 2B/3/3B+/3A+/Zero 2 W: 5V 2.5A micro-USB; Pi 4/400/5/500: 5V 3A USB-C) plus the matching cable
- USB-MIDI instrument or DIN interface (class-compliant devices recommended). Pi Zero 2 W users should connect through a powered USB hub or HAT.
- Stereo RCA audio cabling plus any required RCA-to-TRS/XLR adapters

Ethernet, Wi-Fi, and Bluetooth hardware are disabled in the factory image by design to eliminate attack surface, CPU interrupts and RF noise. Leave all network ports disconnected during normal use.

Ordering and License Binding

Subsynth is sold as a custom, hardware-locked download via <https://intuitionsubsynth.com> for **€49 + VAT per Raspberry Pi device**. Each licence covers one Raspberry Pi in the supported list. Order additional licences if you plan to run Subsynth on multiple units.

1. Assemble or power up the Raspberry Pi you intend to dedicate to Subsynth and boot any standard Raspberry Pi OS image.
2. In a terminal, run `grep Serial /proc/cpuinfo` and copy the hexadecimal value shown on the **Serial** line exactly as printed.
3. Visit the order page and start the secure Stripe checkout. Provide your email address, Raspberry Pi model, and the hardware serial number.
4. Complete payment. You will immediately receive an invoice/receipt by email.
5. The build system embeds your serial number, regenerates the protection bytecode, and watermarks the image specifically for you. When the build completes (typically within one business day), you receive:
 - A download link for your personalised image
 - SHA-256 checksums for integrity verification
 - This manual and support instructions
6. Flash the image to your own microSD card (instructions in the Quick Start section) and keep the download archived securely. The licence is non-transferable; replacing the Raspberry Pi requires a new build.

Hardware Locking & Demo Mode

Multiple layers of build-time and runtime protection tie each Subsynth download to the Raspberry Pi serial number you supplied at checkout. The build embeds an obfuscated hardware serial, per-order watermark, and IE32 protection bytecode. At runtime the synthesis engine validates the CPU serial and continuously monitors binary integrity. No internet activation is required, the checks happen locally and the instrument remains fully offline once installed.

If any protection layer detects tampering, an altered binary, or a hardware mismatch, the system enters **Demo Mode**:

- Only 16 randomly selected melodic programmes and the full GM Drumkit per loaded patchset are accessible
- A yellow DEMO MODE banner appears on the UI with a countdown timer
- After 3-5 minutes (randomised), the audio fades out and Subsynth reboots automatically
- Audio output is forced to HDMI only
- Demo Mode persists until the correct hardware/image pair is restored

If Demo Mode appears on licenced hardware, verify the Pi serial with `grep Serial /proc/cpuinfo`, re-flash the original download, and contact support with your order number. Recompiling, editing the image, or cloning the SD card to another Pi will trigger Demo Mode by design.

Package Contents

Please verify you have received the following digital deliverables after purchase:

- ☐ Personalised Subsynth Raspberry Pi OS image download link (ZIP/IMG)
- ☐ Integrity checksum (SHA-256) for the image
- ☐ End-user manual (PDF) and quick start card
- ☐ Warranty/support registration instructions
- ☐ Invoice showing the Raspberry Pi serial tied to the licence

Ensure that your microSD card, chosen Raspberry Pi model, audio hardware (HiFiBerry DAC2 Pro or HDMI-capable display/receiver), power supply, and cables are ready before flashing the image.

If any deliverables are missing or the download link does not arrive within one business day, please contact support immediately so we can reissue the files.

Panel Overview

Rear Panel (Typical DIY Enclosure):

- Audio Outputs: HiFiBerry DAC2 Pro Left/Right RCA (adapt to TRS/XLR as needed) or HDMI audio (fallback). HiFiBerry is automatically detected on first boot via I2C; if not present, HDMI audio is used.
- Raspberry Pi USB host ports for USB-MIDI controllers or DIN interfaces (Pi 3/4/400/5/500: four USB-A ports; Pi Zero 2 W: micro-USB OTG via powered hub)
- Optional 5-pin DIN breakout if you add a MIDI interface HAT
- Power input: USB-C (Pi 4/400/5/500) or micro-USB (Pi 3/Zero 2 W) connected to the official PSU
- Ethernet and Wi-Fi hardware (physically present on most Raspberry Pi boards but disabled; keep disconnected)
- HDMI display connection for the real-time General MIDI monitor (also carries audio if no HiFiBerry detected). The monitor shows all 320 channels, waveform allocation, NRPN changes, and effect sends in real time. Although Subsynth runs perfectly headless for live performance, connecting a display unlocks this visualiser for debugging complex sequences, teaching MIDI concepts, or demonstrating voice allocation onstage.

5. Quick Start Guide

First-Time Setup Checklist

Follow these steps to get your personalised Intuition Subsynth image running on your supported Raspberry Pi:

Step 0: Download and Verify Your Image

- Download the ZIP file from the link provided in your order email.
- Verify the ZIP checksum:
 - macOS/Linux: `shasum -a 256 subsynth-rpiXseries-HARDWARE_SERIAL_USER_NAME_YYYYMMDDHHMM.zip`
 - Windows (PowerShell): `Get-FileHash .\subsynth-rpiXseries-HARDWARE_SERIAL_USER_NAME_YYYYMMDDHHMM.zip -Algorithm SHA256`
- Compare the result with the ZIP checksum provided in the email. Do not proceed if the values differ.

Step 1: Extract and Verify the Image

- Extract the `.img` file from the ZIP archive.
- Verify the image checksum:
 - macOS/Linux: `shasum -a 256 subsynth-rpiXseries-HARDWARE_SERIAL_USER_NAME_YYYYMMDDHHMM.img`
 - Windows (PowerShell): `Get-FileHash .\subsynth-rpiXseries-HARDWARE_SERIAL_USER_NAME_YYYYMMDDHHMM.img -Algorithm SHA256`
 - Or use the included checksum file: `sha256sum -c subsynth-rpiXseries-HARDWARE_SERIAL_USER_NAME_YYYYMMDDHHMM.img.sha256sum`
- Compare the result with the image checksum (also included in the ZIP). Do not proceed if the values differ.

Step 2: Flash the MicroSD Card

- Use Raspberry Pi Imager, balenaEtcher, or the `dd` command to write the image to an 8GB+ UHS-I A2 microSD card (16GB+ recommended).
 - Raspberry Pi Imager: Choose **Use custom**, select the Subsynth image, pick your SD card, and click **Write**.
 - `dd` (Linux/macOS): `sudo dd if=subsynth-rpiXseries-HARDWARE_SERIAL_USER_NAME_YYYYMMDDHHMM.img of=/dev/sdX bs=4M status=progress conv=fsync; sync; sudo eject /dev/sdX`
- Safely eject the card after `sync` completes to avoid corruption.

Step 3: Prepare the Hardware Platform

- Insert the flashed microSD card into your chosen Raspberry Pi model.
- If using HiFiBerry DAC2 Pro: Mount on the GPIO header and secure it (Pi 400/500 users can route through a GPIO breakout; Pi Zero 2 W requires standoffs for stability). HiFiBerry is automatically detected on first boot via I2C.
- If using HDMI audio: Simply connect an HDMI cable to a display or audio receiver (no additional hardware required).
- Connect your USB-MIDI controller or DIN-to-USB interface. Pi Zero 2 W owners should attach a powered USB hub or HAT to supply sufficient current for MIDI devices.
- Position the assembly on a stable surface with at least 10cm of clearance for ventilation (fanless operation is supported on all boards; Pi 5 benefits from passive heatsinks).
- HDMI display is optional for monitoring. Subsynth functions fully headless as a hardware MIDI synth; connect a screen only if you want on-device status or visual access to the **real-time General MIDI monitor** (per-channel meters, voice allocation heat map, NRPN editors, effect sends, CPU and RAM usage statistics). It is a powerful teaching and troubleshooting tool, but not required for performance.

Step 4: Power On and Allow Licence Verification

- Connect the official 5V power supply for your Raspberry Pi model (micro-USB for Pi 3/Zero 2 W, USB-C for Pi 4/400/5/500).
- **First boot:** The system will configure itself, detect audio hardware (HiFiBerry via I2C or HDMI fallback), and reboot automatically (~20-30 seconds total).
-  **Important:** If using HiFiBerry, ensure it is attached before first boot. Audio hardware detection only runs once. If you boot without HiFiBerry and wish to add it later, you must re-flash the microSD card.
- **Subsequent boots:** The optimised boot process takes only 5-10 seconds.
- If you connected an HDMI display, wait for the GM Monitor to display before playing. The header shows the detected audio device (HiFiBerry DAC2 Pro or HDMI).

Step 5: Connect Audio Outputs

- If using HiFiBerry DAC2 Pro: Connect the Left and Right RCA outputs to your mixer or audio interface.
- If using HDMI audio: Audio routes through the HDMI connection alongside video.
- Use high-quality RCA cables or RCA-to-TRS/XLR adaptors for best noise rejection.
- Set downstream gain to unity (0dB). Subsynth delivers professional +4dBu line level.

Step 6: Connect MIDI Input

- Preferred: Connect a class-compliant USB-MIDI controller directly to the Pi.
- Optional: Use a USB-to-5-pin DIN interface if you require traditional MIDI DIN ports.
- Avoid connecting multiple MIDI sources simultaneously to prevent clashing

Step 7: Test the Connection

- Play a note on your controller to confirm audio output.

- The MIDI activity indicator should flash with each note, and the SIDney patchset loads by default.
- If no sound is heard, review the Troubleshooting guide in Part VI.

Your First 10 Minutes with Subsynth

Playing Your First Notes:

1. **Select a Sound:** The unit powers up with Programme 1 (Acoustic Grand Piano in SIDney bank)
2. **Play the Keyboard:** All 88 keys respond with full velocity sensitivity
3. **Use the Sustain Pedal:** Connect a sustain pedal (CC64) to your MIDI controller
4. **Adjust Volume:** Use CC7 (Channel Volume) on your controller or mixer gain
5. **Try Pitch Bend:** The pitch bend wheel provides expressive pitch control (± 12 semitones in SIDney bank, ± 2 semitones GM default)

Exploring Different Sounds:

1. **Change Programmes:** Send Programme Change messages (0-127)
 - Programme 1-128: Melodic instruments
 - MIDI Channel 10: Drum kit (notes 35-81)
2. **Try Different Categories:**
 - Programmes 1-8: Piano
 - Programmes 25-32: Guitar
 - Programmes 33-40: Bass
 - Programmes 81-88: Lead/Synth
3. **Access the Drum Kit:** Switch to MIDI Channel 10
 - Note 36 (C1): Kick Drum
 - Note 38 (D1): Snare Drum
 - Note 42 (F#1): Closed Hi-Hat
 - Note 46 (A#1): Open Hi-Hat

Switching Sound Banks:

1. Send Bank Select MSB (CC0) + Bank Select LSB (CC32)
2. Available banks:
 - Bank 0: SIDney (Commodore 64)
 - Bank 1: AYYMe (AY-3-8910/YM2149 PSG)
 - Bank 2: Rawland (Classic Roland-inspired)
 - Bank 3: PixelaTED (Commodore TED 7360/8360)
 - Bank 4: EenBeetje (ZX Spectrum 1-bit beeper)

Basic Performance Controls:

- **Modulation Wheel (CC1):** Adds vibrato depth
- **Expression (CC11):** Dynamic volume control (complementary to CC7)
- **Pan (CC10):** Stereo positioning (0=left, 64=centre, 127=right)
- **Reverb Send (CC91):** Master reverb effect level
- **Chorus Send (CC93):** Master chorus effect level

6. Connections and Setup

Rear Panel Connections

Because this release is software-only, all rear connections correspond to your chosen Raspberry Pi board, the HiFiBerry DAC2 Pro HAT, and any MIDI interface you add. If you install a DIN breakout or case, label the connectors

accordingly.

MIDI IN Port (5-pin DIN)

Connector Type: Female 5-pin DIN (180° standard MIDI specification)

Optional DIN interface: Raspberry Pi boards do not include DIN connectors by default. Use a class-compliant USB-to-MIDI interface, a HAT with opto-isolated DIN input, or a breakout PCB wired to the Pi's UART to expose this connector.

Pin Configuration:

- Pin 1: Not connected
- Pin 2: Ground (cable shield)
- Pin 3: Not connected
- Pin 4: MIDI Current Source (+5V through 220Ω resistor)
- Pin 5: MIDI Current Sink (MIDI data input via optoisolator)

Cable Requirements:

- Use standard MIDI cables (5-pin DIN male to male)
- Maximum recommended cable length: 15 metres (50 feet)
- For longer runs, use MIDI repeater/amplifier

Connection Tips:

- MIDI is unidirectional: connect from controller OUT to Subsynth IN
- MIDI IN is optically isolated for electrical safety
- Multiple controllers require a MIDI merge box
- Do not use audio cables for MIDI connections

USB-MIDI via Raspberry Pi USB-A Ports

Use the Raspberry Pi's USB host ports to connect class-compliant USB-MIDI keyboards, pad controllers, or DIN interfaces. Pi 3/4/400/5/500 offer four USB-A ports (two USB 3.0 on Pi 4/5/500), while the Pi Zero 2 W exposes a single USB 2.0 OTG port that should be expanded via a powered USB hub. No additional drivers are required; the Subsynth image includes ALSA/JACK MIDI support.

Specifications:

- Compatible with USB 2.0 and USB 3.0 devices (bandwidth depends on your board's host controller)
- Fully class-compliant (no vendor drivers)
- Supports all MIDI message types, multiple devices, and hubs

Supported Systems:

- USB-MIDI devices are recognised directly by Subsynth

Connection Tips:

- Use powered USB hubs for devices that draw >500mA (mandatory for Pi Zero 2 W)
- Keep USB cables ≤5 metres (16 feet) for reliability
- Disable MIDI routing in your controller if you also use an external DIN interface to avoid duplicate messages

Stereo Line Outputs (HiFiBerry DAC2 Pro RCA)

Because Subsynth currently ships as software only, the analogue outputs come directly from your HiFiBerry DAC2 Pro HAT. The card exposes two RCA/phono connectors (Left/Right). Use RCA cables directly, or adapt them to TRS/XLR as required by your studio.

Connector Type: Stereo RCA (Left = white, Right = red). Optional balanced adapters via transformer or DI box.

Audio Specifications (HiFiBerry DAC2 Pro):

- Output Level: +4dBu nominal (2.1Vrms balanced equivalent via adapter)
- Maximum Output: 2.1Vrms (approx. +8.4dBu) before soft clipping
- Output Impedance: 75Ω per channel
- Frequency Response: 20Hz-20kHz (± 0.1 dB)
- THD+N: -93dB @ -1dBFS, 1kHz (PCM5122 specification)
- Dynamic Range: 112dB (A-weighted)

Connection Options:

- **Studio Mixers/Interfaces (Balanced Inputs):**
 - Use high-quality RCA-to-TRS or RCA-to-XLR adaptors or cables.
 - Keep cable runs short (<3m) when unbalanced; add a DI box for longer runs.
- **Consumer Gear (Unbalanced Inputs):**
 - Use standard RCA-to-RCA cables.
 - Reduce destination gain by ~10dB if the input is -10dBV nominal.
- **Direct Injection (Live Sound):**
 - Feed each RCA output into a passive DI box to convert to balanced XLR.
 - Engage ground lift on the DI if hum occurs.

Gain Staging Tips:

- Set downstream line inputs to +4dBu where possible.
- Avoid running the DAC at full-scale continuously; leave 3-6dB of headroom.
- If your interface expects TRS, keep the RCA-to-TRS adaptor as short as possible to minimise noise.

Power Input

Connector Types:

- **USB-C:** Raspberry Pi 4, Raspberry Pi 400, Raspberry Pi 5, Raspberry Pi 500 (Power Delivery compatible)
- **Micro-USB:** Raspberry Pi 3 Model B/B+, Raspberry Pi Zero 2 W (5V @ 2.5A, use high-quality cable)

Power Requirements:

- Voltage: 5V DC $\pm 5\%$ (4.75V-5.25V)
- Current:
 - Pi 4/400/5/500: 3A minimum (15W). Recommended: 5V DC, 3.5A supply for headroom with USB devices.
 - Pi 3/Zero 2 W: 2.5A minimum (12.5W). Use a powered USB hub for MIDI devices drawing >500mA.

Power Supply Standards:

- Must comply with IEC 62368-1 (PS2 classification)
- Must be certified for country of use (CE, FCC, UL, etc.)
- Use the official Raspberry Pi PSU for your board or another certified equivalent

Power Supply Recommendations:

- Use the official Raspberry Pi adapter whenever possible
- Replacement supplies must match the connector and current rating required by your board
- Avoid generic/uncertified USB chargers
- Avoid USB power from computers (may not provide sufficient current)

- Use cables rated for 3A (USB-C) or thick 24AWG power pairs (micro-USB) to minimize voltage sag

Ground Loop Prevention

What is a Ground Loop?

A ground loop occurs when multiple pieces of equipment are connected to different ground points at different electrical potentials, causing a low-frequency hum (50Hz/60Hz mains frequency) in the audio signal.

Prevention Strategies:

1. Single-Point Grounding (Best Practice):

- Connect all audio equipment to the same mains outlet or power strip
- Ensures all equipment shares the same ground reference
- Use a high-quality power distribution unit with filtered outlets

2. Balanced Connections:

- Whenever possible, convert the RCA outputs to balanced via DI boxes or transformer isolators
- Balanced connections reject common-mode noise (including ground loop hum)
- Keep unbalanced RCA runs short and avoid parallel power cables

3. Ground Lift (Last Resort):

- Some audio interfaces and mixers have ground lift switches
- Lifting ground on the destination equipment may eliminate hum
- **Warning:** Never defeat safety ground on mains power cables

4. Isolation Transformers:

- Audio isolation transformers break ground loops without safety risks
- Place between Subsynth outputs and destination inputs
- Minimal impact on audio quality with professional units

Cable Recommendations

MIDI Cables:

- Standard 5-pin DIN MIDI cables
- Shielded, twisted-pair construction
- Maximum length: 15 metres (50 feet)
- Recommended brands: MIDI Solutions, Roland, Yamaha

USB-MIDI Cables:

- Use the cable provided with your controller (typically USB-A to USB-B or USB-A to USB-C)
- Shielded cables with ferrite cores reduce interference
- Maximum recommended length: 5 metres (16 feet)
- Use powered USB hubs for multi-controller setups

Audio Cables:

- High-quality RCA cables or RCA-to-TRS/XLR adaptors
- Keep unbalanced runs as short as practical; use DI boxes for long runs
- Oxygen-free copper conductors with good shielding recommended
- Trusted brands: Mogami, Canare, Sommer, Neutrik adaptors

Power Cables:

- Use the official Raspberry Pi power supply and cable set appropriate for your board (USB-C for Pi 4/400/5/500, micro-USB for Pi 3/Zero 2 W)
 - Replacement cables must meet the current requirements (USB-C 3A minimum, micro-USB 2.5A with thick 24AWG conductors)
 - Avoid thin "charge-only" cables (insufficient for sustained current draw)
-

7. Panel Description

Because Subsynth is currently distributed as software, there is no factory enclosure or labelled rear panel. Treat your Raspberry Pi board, the HiFiBerry DAC2 Pro outputs, and any optional MIDI breakouts you add as the instrument's "panels." The guidance below describes the recommended layout if you build or purchase a case for your Pi stack; adapt the labelling to match your chosen enclosure and board footprint.

Rear Panel Elements

See Chapter 6 (Connections and Setup) for detailed descriptions of all rear panel connectors.

Side Panels

If you install a supported Raspberry Pi and HiFiBerry DAC2 Pro in an enclosure, include ventilation slots or a low-noise fan on the sides. Keep vents unobstructed and avoid placing the case directly against absorptive materials. Bare-board setups should maintain at least 10cm of clearance around the Pi for airflow.

Thermal Management

For optimal thermal performance:

- Allow 10cm (4 inches) clearance on left and right sides
 - Do not block ventilation slots with cables or objects
 - Do not place in enclosed spaces without ventilation
 - Operating temperature: 0°C to 35°C (32°F to 95°F)
 - Storage temperature: -20°C to 60°C (-4°F to 140°F)
-

Part II: Playing the Instrument

8. Basic Operation

Powering On/Off Procedures

Power-On Sequence:

1. Ensure all audio connections are made and mixer/interface gain is at minimum
2. Connect the Raspberry Pi power cable (USB-C or micro-USB based on your board)
3. Connect the power supply to the mains outlet (the Pi boots immediately)
4. **First boot only:** Wait ~20-30 seconds whilst the system configures itself and reboots automatically
 - User account creation and permission setup
 - Service optimisation for real-time audio
 - Read-only filesystem protection activation
 - System will reboot once to apply these settings
5. **Subsequent boots:** Wait for the **boot confirmation tone** (5-10 seconds from power-on):
 - Deep bass arpeggio (C0→C1→C2→C3) with SID-style PWM and aggressive filter sweep
 - Duration: ~1.5 seconds
 - Uses SIDney Patch 81 (Square Lead) with maximum resonance and overdrive
 - **This tone signifies the synth is fully initialised and ready to produce sound**

- If you don't hear this tone, check audio connections, HiFiBerry seating, and power supply
6. If a display is attached, the Subsynth GM Monitor will be visible showing real-time channel activity
 7. Raise mixer/interface gain to operating level gradually
 8. Play a test note to verify MIDI and audio connectivity

Power-Off Sequence:

1. Reduce mixer/interface gain to minimum (prevents pop on disconnect)
2. Stop all MIDI playback and let lingering effects trails fade
3. Turn off power at the wall outlet/inline switch, or unplug the power connector from the Raspberry Pi. (If you attached a local keyboard/monitor you may optionally run `sudo poweroff`, but it is not required for normal operation.)
4. Wait 5 seconds before turning the power back on if you plan to restart immediately.

Important: The supplied image minimises disk writes and runs entirely from solid-state storage, so disconnecting power after the instrument idles will not corrupt the system. Avoid removing power whilst flashing SD cards or copying files via the optional UI.

Sound Bank Selection

The Intuition Subsynth includes five factory sound banks (patchsets), each with 128 melodic instruments and a complete drum kit (47 sounds).

Available Banks:

Bank	Name	Description	Best For
0	SIDney (Default)	MOS 8580 PAL SID emulation (default startup patchset)	Classic chip sounds, retro music
1	AYYMe	AY-3-8910/YM2149 PSG (Amstrad/Spectrum/MSX/AtariST)	8-bit sounds, retro music
2	Rawland	Roland-inspired (303/808/909)	Electronic, techno, house
3	PixelaTED	Commodore TED 7360/8360 (Plus/4, C16)	Vintage computer music
4	EenBeetje	ZX Spectrum 1-bit beeper	Authentic 1-bit synthesis

Note: Bank 0 resets to the default patchset (specified via `--patchset=NAME` at startup, or SIDney if not specified).

Bank Selection Method:

The Intuition Subsynth uses **CC#0 (Bank Select MSB)** to determine the bank number. CC#32 (Bank Select LSB) is accepted but ignored.

Procedure:

1. Send CC#0 (Bank Select MSB) = [bank number 0-4]
2. Send CC#32 (Bank Select LSB) = 0 (optional, ignored)
3. Send Programme Change message [0-127]
4. New bank + programme loads immediately

Example: Switch to Rawland Bank, Programme 35 (Electric Bass)

```
CC#0 (Bank Select MSB) = 2      (Rawland bank)
CC#32 (Bank Select LSB) = 0     (optional, ignored)
```

PC (Programme Change) = 34 (0-indexed: Programme 35)

Note: Only CC#0 determines the bank. Some MIDI software sends both CC#0 and CC#32 for compatibility, but the Intuition Subsynth only reads CC#0 (MSB).

Bank Persistence:

- Bank switching is **global** (affects all 16 MIDI channels simultaneously)
- Selected bank remains active until changed by another Bank Select message
- Power cycling returns to user's default (from -- patchset=NAME , or SIDney if not specified)
- Bank 0 resets to the default patchset (useful when MIDI files send Bank 0 reset messages)

GM Level 2: Dynamic Rhythm Conversion (Bank 120):

Bank 120 (0x78 hexadecimal) converts **any MIDI channel** to percussion mode, enabling multiple simultaneous drum tracks beyond the standard Channel 10.

How to Convert a Channel to Drums:

1. Send CC#0 (Bank Select MSB) = 120 on the desired channel
2. Send Programme Change (value ignored, drum kit always loads)
3. Channel now plays drum sounds from the current patchset's drum kit

Example: Create Three Simultaneous Drum Kits

Channel 10: Default percussion (always active)
Channel 11: CC#0 = 120 → Converted to drums
Channel 5: CC#0 = 120 → Converted to drums

Result: Three independent drum tracks playing simultaneously

Use Cases:

- **Dual drum kits:** Main drums on Channel 10, fills/variations on Channel 11
- **Layered percussion:** Combine multiple drum patterns without channel conflicts
- **Complex arrangements:** Separate bass drum, snare and hi-hat onto different channels for independent control

Reverting to Melodic Mode:

To convert a percussion channel back to melodic instruments:

1. Send CC#0 with any bank value **except** 120 (e.g., Bank 0 for default)
2. Send Programme Change for desired melodic instrument
3. Channel resumes melodic operation

Note: Bank 120 does **not** change the active patchset. It only affects the channel's mode (percussion vs melodic). The drum kit loaded is always from the currently active patchset (SIDney, AYYMe, Rawland, etc.).

Programme Change Messages

Programme Numbering:

- MIDI specification: Programmes numbered 0-127 (128 programmes)
- Musical convention: Programmes displayed as 1-128 (add 1 to MIDI number)
- Subsynth follows MIDI spec internally (0-127)
- This manual displays both formats: "Programme 1 (PC0)" = Programme Change 0

Programme Categories (General MIDI Level 1):

Programmes	Category	Typical Sounds
------------	----------	----------------

1-8	Piano	Acoustic Grand, Electric Piano, Clavinet
9-16	Chromatic Percussion	Glockenspiel, Xylophone, Marimba, Bells
17-24	Organ	Drawbar, Percussive, Rock, Church
25-32	Guitar	Acoustic, Electric, Distortion, Harmonics
33-40	Bass	Acoustic, Electric, Fretless, Slap, Synth
41-48	Strings	Violin, Viola, Cello, Pizzicato, Harp
49-56	Ensemble	String Ensemble, Choir, Voice
57-64	Brass	Trumpet, Trombone, French Horn, Brass
65-72	Reed	Saxophone, Oboe, Clarinet, Bassoon
73-80	Pipe	Flute, Piccolo, Pan Flute, Recorder
81-88	Synth Lead	Square, Sawtooth, Calliope, Fifth
89-96	Synth Pad	Warm, Poly, Choir, Bowed, Metallic
97-104	Synth Effects	Rain, Soundtrack, Crystal, Atmosphere
105-112	Ethnic	Sitar, Banjo, Shamisen, Koto, Kalimba
113-120	Percussive	Tinkle Bell, Steel Drums, Melodic Tom
121-128	Sound Effects	Guitar Fret, Breath, Seashore, Telephone

Sending Programme Change Messages:

Most MIDI keyboards and controllers have dedicated programme change buttons or knobs. Consult your controller's manual for specific instructions.

From a DAW or Sequencer:

- Insert a Programme Change event at the desired location
- Set the programme number (0-127 or 1-128 depending on DAW)
- Event takes effect immediately when received

From a MIDI Controller:

- Use the controller's programme +/- buttons
- Use the controller's bank/programme selection mode
- Use a foot switch programmed for programme change

Understanding Polyphony (320 Channels by Wave Type)

Important Concept:

The Intuition Subsynth is **not** a simple 320-voice polyphonic synthesiser. Instead, it uses **wave-type-based voice allocation**, where voices are organised into five pools of 64 channels each, one pool per waveform type.

How Wave-Type Allocation Works:

1. Each sound (GM programme) has a primary waveform type: Square, Triangle, Sine, Noise, or Sawtooth

2. When you play a note, the synthesiser allocates a channel from that waveform's 64-channel pool
3. If all 64 channels of that waveform are in use, the oldest note is stolen (voice stealing)
4. Different waveforms do not compete (e.g., using all 64 square channels does not affect triangle channels)

Channel Allocation by Wave Type:

Wave Type	Channel Range	Typical Sounds
Square	0-63	Pulse waves, PWM sounds, retro leads
Triangle	64-127	Smooth bass, mellow leads, vintage chip tones
Sine	128-191	Pure tones, FM bass, sub-bass, sine pads
Noise	192-255	Drums (snare, hi-hat), percussion, wind, FX
Sawtooth	256-319	Bright leads, brass, strings, rich bass

Practical Implications:

- **Scenario 1:** Playing 64 square-wave notes (e.g., pulse lead sound) fills the square pool but leaves 256 other channels available for other waveforms
- **Scenario 2:** Playing 64 square notes + 64 sawtooth notes = 128 total notes playing simultaneously (no voice stealing yet)
- **Scenario 3:** Playing 65 square notes causes the oldest square note to be stolen (note 1 stops, note 65 plays)

Layered Patches:

Some sounds use two waveforms simultaneously (layered patches). For example, "Rich Brass" might layer sawtooth (primary) + square (layer). Layered patches consume **two channels per note**:

- One channel from the primary waveform pool
- One channel from the layer waveform pool

Maximum polyphony for layered patches: 64 notes (128 channels consumed).

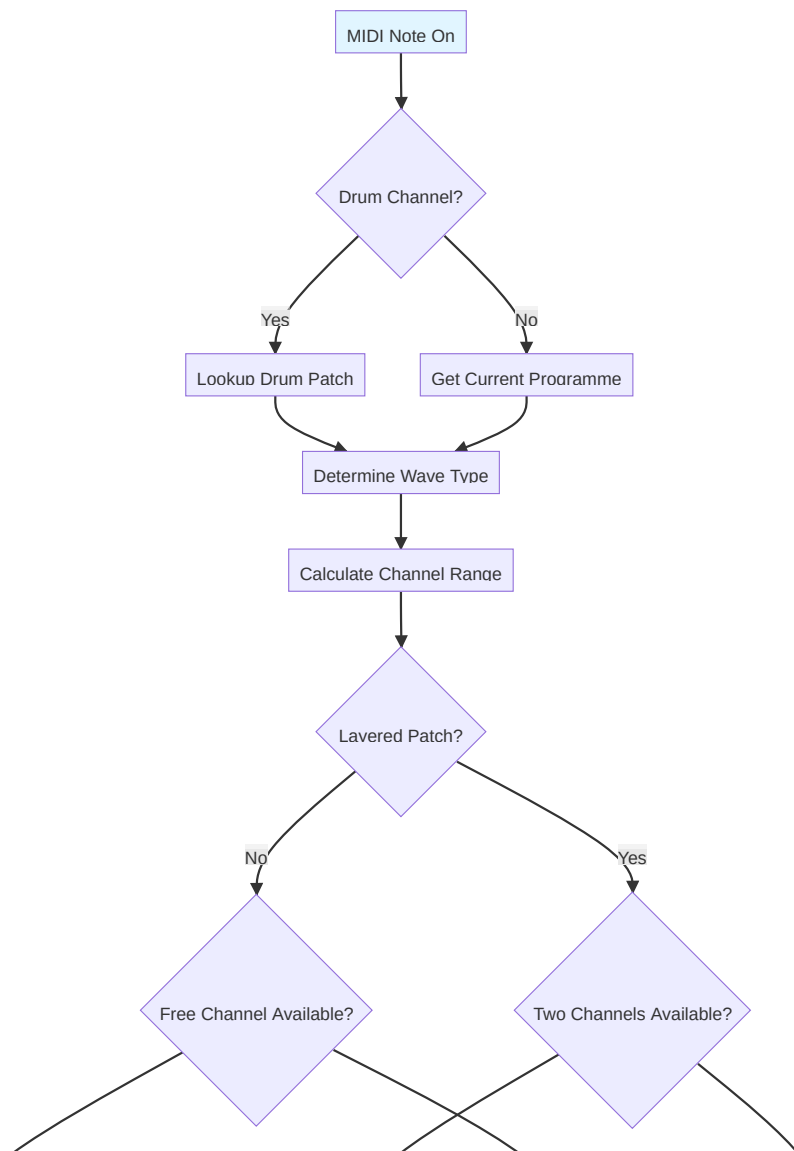
Voice Stealing Priority:

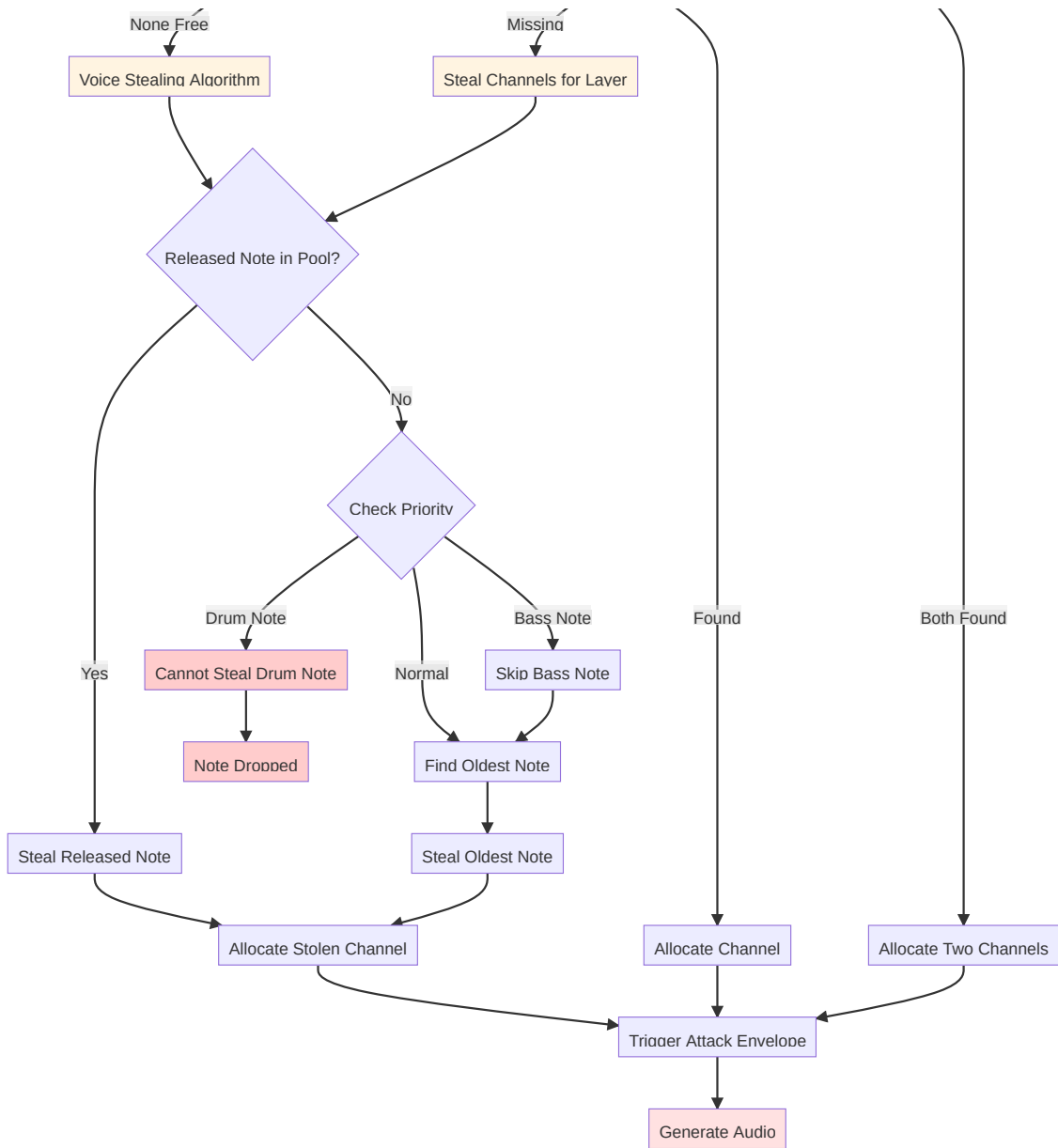
When voice stealing occurs, the synthesiser uses intelligent priority:

1. **Drums are never stolen** (MIDI channel 10 has highest priority)
2. **Bass notes are protected** (notes below C2 are stolen last)
3. **Oldest notes stolen first** (FIFO within wave type pool)
4. **Released notes stolen first** (notes in release phase stolen before sustained notes)

Voice Allocation Flow Diagram:

The following diagram illustrates the complete voice allocation and stealing algorithm:





Voice Allocation and Stealing Behaviour

Voice Allocation Algorithm:

1. **MIDI Note-On Received:** Controller sends Note-On message (0x9n, note, velocity)
2. **Patch Lookup:** Synthesiser looks up the GM programme's waveform type
3. **Channel Search:** Searches the waveform's 64-channel pool for an available channel
4. **Available Channel Found:** Allocates channel, starts envelope attack phase, begins synthesis
5. **No Available Channels:** Enters voice stealing mode

Voice Stealing Mode:

1. **Check Release Phase:** Look for channels in release phase (key already released)
 - If found: Steal that channel (least disruptive)
 - Fast-release the old note (5ms ramp to zero)
 - Start new note immediately
2. **Check Oldest Notes:** If no released notes, find oldest active note (by timestamp)
 - Exclude MIDI channel 10 (drums, highest priority)
 - Exclude notes below C2 (bass notes, second highest priority)
 - Steal oldest qualifying note
3. **Fast-Release Old Note:** 5ms exponential ramp to silence (ENV_FORCED_RELEASE phase)
4. **Start New Note:** Allocate stolen channel to new note, begin attack phase

Avoiding Voice Stealing:

- Use appropriate waveform types for your music (distribute notes across waveforms)
- Avoid sustained chords with >64 notes of the same waveform
- Use shorter release times (allows faster voice recycling)
- Prioritise bass and drums (synthesiser does this automatically)

Observing Voice Stealing:

Voice stealing is typically inaudible when done correctly. However, you may notice:

- Very long release tails cutting off early (>64 notes of same waveform playing)
- High-pitched notes disappearing when new notes played (bass priority in effect)
- Slight "pop" if extreme polyphony exceeded very rapidly (fast-release not fast enough)

Using Sustain Pedal

MIDI Controller: CC64 (Sustain Pedal / Damper Pedal)

Behaviour:

- **Values 0-63:** Sustain off (notes release normally when key released)
- **Values 64-127:** Sustain on (notes continue sustaining after key released)

How Sustain Works:

1. Press sustain pedal (controller sends CC64 = 127)
2. Play and release notes on keyboard
3. Notes continue to sound at sustain level (ADSR sustain phase)
4. Release sustain pedal (controller sends CC64 = 0)
5. All sustained notes enter release phase simultaneously

Sustain and Polyphony:

- Sustained notes continue to consume voice allocation
- With sustain pedal held, it's easy to exceed 64 notes of a single waveform
- Voice stealing will occur if you exceed the waveform's 64-channel pool
- Release sustain pedal periodically to free voices

Per-Channel Sustain:

- Sustain is per-MIDI-channel (independent control for each of 16 MIDI channels)
- Channel 10 (drums): Sustain pedal typically ignored (drum sounds have built-in decay)
- Channels 1-9, 11-16: Sustain operates normally

Half-Pedalling (Continuous Controllers):

The Intuition Subsynth treats CC64 as a switch (on/off) rather than continuous. Values 0-63 = off, values 64-127 = on. Half-pedalling (partial sustain) is not supported.

Pitch Bend and Modulation Wheel

Pitch Bend (0xEn, 14-bit resolution):

Range:

- Default: ± 2 semitones (General MIDI Level 1 standard)
- SIDney patchset: ± 12 semitones (Commodore 64-style slides)
- Configurable via RPN #0 (Pitch Bend Sensitivity): ± 1 to ± 24 semitones

Resolution:

- 14-bit: 16,384 discrete steps (0x0000 to 0x3FFF)
- Centre: 8,192 (0x2000, no pitch bend)
- Minimum: 0 (0x0000, maximum downward bend)
- Maximum: 16,383 (0x3FFF, maximum upward bend)

Precision:

- ± 2 semitones: ~ 0.024 cents per step (imperceptible)
- ± 12 semitones: ~ 0.146 cents per step (smooth slides)

Behaviour:

- Pitch bend affects all notes on the MIDI channel simultaneously
- Pitch bend is per-MIDI-channel (independent control for 16 channels)
- Pitch bend is continuously variable (smooth glissando, not stepped)
- Pitch bend does not affect voice allocation or polyphony

Adjusting Pitch Bend Range:

Use RPN #0 (Pitch Bend Sensitivity) to change the range:

CC101 (RPN MSB) = 0 (Pitch Bend Sensitivity MSB)
CC100 (RPN LSB) = 0 (Pitch Bend Sensitivity LSB)
CC6 (Data Entry MSB) = 12 (± 12 semitones = ± 1 octave)
CC38 (Data Entry LSB) = 0 (fine tuning, optional)

Modulation Wheel (CC1):

Range: 0-127 (7-bit resolution)

Function: Controls vibrato depth (LFO intensity)

Behaviour:

- Value 0: No vibrato (LFO depth = 0)
- Value 127: Maximum vibrato (LFO depth = 100 cents)
- Intermediate values: Proportional vibrato depth

Vibrato Characteristics:

- Waveform: Sine wave (smooth, musical vibrato)
- Rate: Configurable per-patch (typically 4-7 Hz)
- Depth: Controlled by CC1 (0-100 cents)
- Pitch modulation: Symmetrical ($\pm$ depth around centre pitch)

Using Modulation Wheel:

The modulation wheel is ideal for adding expressive vibrato to sustained notes:

1. Play and hold a note
2. Gradually increase modulation wheel (CC1: 0 $\rightarrow$ 127)
3. Vibrato depth increases smoothly from none to full
4. Decrease modulation wheel to reduce or remove vibrato

Per-Channel Modulation:

- Modulation is per-MIDI-channel (independent control for 16 channels)
- Each channel can have different modulation depth simultaneously
- Modulation affects all notes on the channel equally

9. Understanding the Sound Engine

320-Channel Architecture Explained for Musicians

The Intuition Subsynth uses a unique architecture that differs from traditional polyphonic synthesisers. Instead of having a simple pool of 320 voices that can play any sound, the synthesiser organises its voices into **five separate pools**, one for each waveform type.

Why This Matters:

Traditional synthesisers allocate voices from a single pool. If a synth has 64 voices, you can play any 64 notes regardless of their sound. The Intuition Subsynth is different: it has **64 voices per waveform type** (320 total), and sounds are assigned to specific waveform types.

Practical Example:

Imagine you're playing a sawtooth lead sound (Programme 82, Sawtooth Lead). Each note you play consumes one channel from the **sawtooth pool** (channels 256-319). You could play all 64 sawtooth channels simultaneously.

Meanwhile, the other 256 channels (square, triangle, sine, noise) remain completely available for other sounds.

If you then switch to a square wave bass sound (Programme 39, Synth Bass 1), those notes will use the **square wave pool** (channels 0-63), leaving your sawtooth notes unaffected.

Channel Allocation Summary:

Wave Type	Channels	Capacity	Typical Sounds
Square	0-63	64 notes	Pulse leads, PWM sounds, chip tones
Triangle	64-127	64 notes	Smooth bass, mellow leads, retro tones
Sine	128-191	64 notes	Pure tones, sub-bass, FM sounds
Noise	192-255	64 notes	Drums, percussion, wind, sound effects
Sawtooth	256-319	64 notes	Bright leads, brass, strings, bass

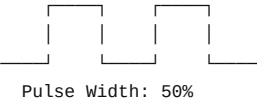
Total Polyphony: 320 simultaneous notes (64 per waveform type)

Five Waveform Types

The Intuition Subsynth generates sound using five classic waveform types, each with distinctive sonic characteristics:

1. Square Wave (Channels 0-63)

Visual Representation:



Characteristics:

- Bright, hollow timbre (often described as "woody" or "reedy")
- Contains only odd harmonics (1st, 3rd, 5th, 7th, etc.)
- Perfect for cutting through a mix with clarity
- Pulse Width Modulation (PWM) adds movement and richness

Musical Applications:

- Lead synthesiser lines (classic analogue sound)
- Bass sounds (especially with filter modulation)
- Retro video game music (Commodore 64, NES)
- Brass and reed simulations

Pulse Width Modulation (PWM):

Square waves can vary their duty cycle (the ratio of "high" time to "low" time):

- 50% duty cycle: Classic square wave (balanced sound)
- 10-30% duty cycle: Thin, nasal tone
- 70-90% duty cycle: Thin, nasal tone (mirror of 10-30%)
- Modulated PWM: Creates rich, chorused, string-like sounds

PWM is controlled via NRPN parameters (see Part III: Sound Programming).

2. Triangle Wave (Channels 64-127)

Visual Representation:



Characteristics:

- Mellow, soft timbre (flute-like quality)
- Contains only odd harmonics with rapid falloff ($1/n^2$ amplitude decay)
- Less brightness than square wave, more warmth than sine wave
- Smooth, musical sound ideal for sustained tones

Musical Applications:

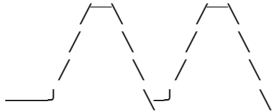
- Smooth bass lines (classic 8-bit game bass)
- Mellow lead sounds
- Pad and organ sounds (especially with layering)
- Emulating woodwind instruments (flute, recorder)

Harmonic Content:

Triangle waves have a gentler harmonic series than square waves, producing a less aggressive tone whilst maintaining character.

3. Sine Wave (Channels 128-191)

Visual Representation:



Characteristics:

- Pure tone with no harmonics (fundamental frequency only)
- Warm, smooth, "glassy" timbre
- Lacks edge or brightness
- Perfect for sub-bass and FM synthesis

Musical Applications:

- Sub-bass lines (808-style kick drums, deep bass)
- FM synthesis carrier and modulator tones
- Pure test tones and tuning references
- Bell and mallet percussion sounds
- Ambient pad layers (adds warmth without brightness)

Why Sine Waves Matter:

Whilst sine waves might seem "boring" compared to harmonically rich waveforms, they are essential building blocks:

- **Sub-bass:** Sine waves below 60Hz provide weight without muddiness
- **FM synthesis:** Sine waves modulating other sine waves create complex, inharmonic spectra (classic DX7-style sounds)
- **Layering:** Adding a sine wave layer to bright sounds adds body and depth

4. Noise (Channels 192-255)

Visual Representation:



Characteristics:

- Aperiodic waveform (no definite pitch)
- Broadband spectrum (all frequencies present)
- Three noise modes: White, Periodic, Metallic
- Essential for drum sounds and sound effects

Musical Applications:

- Drum sounds (snare, hi-hat, cymbals, tom-toms)
- Percussion (shakers, tambourine, maracas)
- Sound effects (wind, ocean, explosions, industrial)
- Breath noise in woodwind simulations
- Attack transients for realistic instrument emulation

Noise Modes:

1. White Noise (Mode 0):

- Flat frequency spectrum (equal energy at all frequencies)
- "Hissing" sound, like radio static or air escaping
- Ideal for snare drums, hi-hats, and cymbals

2. Periodic Noise (Mode 1):

- Short repeating pattern (quasi-periodic)
- Vintage sound chip character (Commodore SID, NES)
- Metallic, buzzy quality
- Useful for retro-style drums and effects

3. Metallic Noise (Mode 2):

- Alternative random pattern with metallic timbre
- Distinctive "crunchy" quality
- Useful for metallic percussion and industrial sounds

Noise mode is configured per-patch via NRPN parameters.

5. Sawtooth Wave (Channels 256-319)

Visual Representation:



Characteristics:

- Bright, buzzy, aggressive timbre

- Contains all harmonics (odd and even) with 1/n amplitude decay
- Richest harmonic content of all basic waveforms
- Classic analogue synthesiser sound

Musical Applications:

- Lead synthesiser sounds (screaming leads, solos)
- Brass sections (trumpet, trombone, horn)
- String sections (violins, cellos, orchestral strings)
- Bass sounds (especially with low-pass filter)
- Pad sounds (with filtering and effects)

Why Sawtooth is Versatile:

The sawtooth wave's rich harmonic content makes it the most versatile waveform for subtractive synthesis:

- **Bright and cutting:** All harmonics present means maximum frequency content
- **Filters beautifully:** Low-pass filtering removes high harmonics, creating smooth to bright timbres
- **Classic analogue:** Sawtooth + low-pass filter = archetypal synthesiser sound

Anti-Aliasing:

At high frequencies (above 4-6kHz depending on sample rate), digital sawtooth waves can produce aliasing artefacts (unwanted frequencies that fold back into the audible spectrum). The Intuition Subsynth uses professional PolyBLEP (Polynomial Band-Limited Step) anti-aliasing to eliminate these artefacts, ensuring pristine audio quality at all pitches.

Wave-Type Channel Pooling Explained

The Core Concept:

Each General MIDI programme (sound) is assigned a primary waveform type when designed. This assignment determines which channel pool the sound will use.

Example Programme Assignments:

Programme	Name	Wave Type	Pool Used
1	Acoustic Grand	Sawtooth	256-319
33	Acoustic Bass	Triangle	64-127
39	Synth Bass 1	Square	0-63
81	Lead 1 (Square)	Square	0-63
82	Lead 2 (Sawtooth)	Sawtooth	256-319
36 (drum)	Kick Drum	Sine	128-191
38 (drum)	Snare Drum	Noise	192-255
42 (drum)	Closed Hi-Hat	Noise	192-255

What Happens When You Play:

1. **You send a Note-On message:** MIDI channel 1, note 60 (middle C), velocity 100, programme 82 (Sawtooth Lead)

2. **Synthesiser looks up waveform type:** Programme 82 = Sawtooth
3. **Searches sawtooth pool (256-319):** Finds first available channel, e.g. channel 260
4. **Allocates channel 260:** Starts envelope attack, begins generating sawtooth wave at middle C frequency
5. **You send Note-Off message:** Channel 260 enters release phase, decays to silence
6. **Channel 260 becomes available again:** Ready for next note

Polyphony Limits Per Waveform:

If you play more than 64 notes of the same waveform type, voice stealing occurs:

1. **Synthesiser searches for available channel:** None found (all 64 in use)
2. **Initiates voice stealing algorithm:** Finds oldest active note in that waveform pool
3. **Fast-releases old note:** 5ms exponential ramp to silence
4. **Allocates stolen channel to new note:** Starts attack phase immediately

Cross-MIDI-Channel Stealing:

Voice stealing occurs **within waveform pools**, not MIDI channels. This means:

- If MIDI channel 1 uses all 64 square channels, notes on MIDI channel 2 that use square waves will steal from MIDI channel 1
- Different MIDI channels **share** the 64-channel waveform pools
- Exception: MIDI channel 10 (drums) has highest priority and is never stolen

Layered Patches and Polyphony:

Some sounds use **two waveforms simultaneously** (layered patches). For example:

- **Rich Brass:** Sawtooth (primary) + Square (layer)
- **Fat Bass:** Triangle (primary) + Sine (layer)
- **Lush Strings:** Sawtooth (primary) + Triangle (layer)

Layered patches consume **two channels per note** (one from each waveform pool). Maximum polyphony for layered patches: **64 notes** (128 channels total, 64 from each pool).

Checking Wave Type Assignments:

You can determine which waveform a programme uses by:

1. Consulting the programme list in Part V: Sound Banks
2. Listening to the character (bright = sawtooth, hollow = square, smooth = triangle)
3. Using NRPN parameters to query patch parameters (advanced)

ADSR Envelope Basics

Every note you play on the Intuition Subsynth has an **amplitude envelope** that shapes how the volume changes over time. This envelope uses the classic ADSR model (Attack, Decay, Sustain, Release), a standard in synthesiser design since the 1960s.

The Four Stages:

1. Attack (A)

The **attack time** is how long it takes for the sound to go from silence (0%) to full volume (100%) after you press a key.

- **Short attack** (0-10ms): Percussive, punchy sounds (piano, drums, plucked strings)
- **Medium attack** (50-200ms): Natural acoustic sounds (brass, woodwinds)

- **Long attack** (500-5000ms): Slow-building pads, swells, ambient textures

Range: 0 to 11,700ms (0 to 11.7 seconds) **MIDI Control:** CC73 (Attack Time), 0-127 **Musical Tip:** Instant attack (0ms) suits rhythmic parts; slow attack (>1s) suits pads and soundscapes

2. Decay (D)

The **decay time** is how long it takes for the sound to drop from full volume (100%) down to the sustain level after reaching the attack peak.

- **Short decay** (0-50ms): Bright, plucky sounds (harpsichord, electric piano)
- **Medium decay** (100-500ms): Natural decay (acoustic guitar, marimba)
- **Long decay** (1000-2000ms): Slow falloff (bell sounds, synthesised textures)

Range: 0 to 2,000ms (0 to 2 seconds) **MIDI Control:** CC75 (Decay Time), 0-127 **Musical Tip:** Fast decay creates percussive sounds; slow decay suits evolving textures

3. Sustain (S)

The **sustain level** is the volume the sound holds at whilst you keep the key pressed (after attack and decay complete).

- **High sustain** (80-100%): Organ-like, continuous sounds
- **Medium sustain** (40-70%): Natural acoustic instruments (woodwinds, brass)
- **Low sustain** (0-30%): Fading sounds (piano, bells, plucked strings)
- **Zero sustain** (0%): Purely percussive (no sustained sound, just attack/decay)

Range: 0% to 100% (logarithmic scale) **MIDI Control:** Not directly controllable via CC (set per-patch) **Musical Tip:** Sustain level determines whether the sound continues or fades whilst held

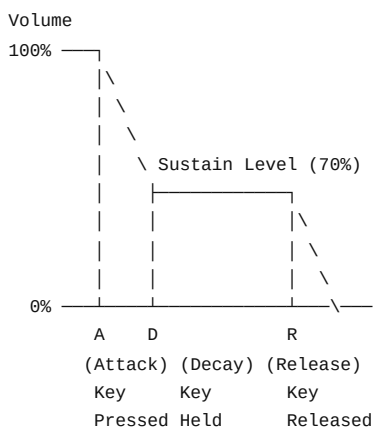
4. Release (R)

The **release time** is how long it takes for the sound to fade to silence (0%) after you release the key.

- **Short release** (0-50ms): Abrupt cutoff (staccato, percussive)
- **Medium release** (100-500ms): Natural acoustic decay (piano, guitar)
- **Long release** (1000-5000ms): Lingering reverberant sounds (gongs, synthesised pads)
- **Very long release** (>5s): Ambient drones, evolving textures

Range: 0 to 15,000ms (0 to 15 seconds) **MIDI Control:** CC72 (Release Time), 0-127 **Musical Tip:** Long release creates legato, overlapping notes; short release creates tight, rhythmic articulation

ADSR Visualisation:



Envelope Shapes:

The Intuition Subsynth supports three envelope curve shapes:

- 1. **Linear** (Shape 0): Constant-rate ramps, authentic vintage chip character
- 2. **Exponential** (Shape 1): Natural acoustic decay curves, professional synthesiser character (default)
- 3. **Logarithmic** (Shape 2): Currently reserved for future use

Most patches use **exponential envelopes** (shape 1) for natural-sounding attack and release curves. Vintage chip emulation patchsets (AYYMe, PixelaTED, EenBeetje) use **linear envelopes** (shape 0) for authentic retro character.

Adjusting Envelopes in Real-Time:

You can modify attack and release times whilst playing using MIDI Continuous Controllers:

- **CC73 (Attack Time)**: Send values 0-127 to change attack (0 = instant, 127 = 11.7 seconds)
- **CC72 (Release Time)**: Send values 0-127 to change release (0 = instant, 127 = 15 seconds)

Decay time (CC75) is recognised but typically set per-patch and rarely changed during performance.

Musical Applications:

Sound Type	Attack	Decay	Sustain	Release	Character
Piano	0ms	200ms	30%	500ms	Percussive, decaying
Organ	5ms	0ms	100%	50ms	Instant on/off
Strings	100ms	100ms	80%	1000ms	Smooth, sustained
Brass	50ms	200ms	70%	300ms	Natural acoustic
Pad	1000ms	500ms	90%	2000ms	Slow-building, ambient
Plucked Bass	5ms	300ms	0%	200ms	Percussive, decaying
Synthesiser Lead	0ms	0ms	100%	100ms	Immediate, sustained

Filter Types (LP/HP/BP)

The Intuition Subsynth features a professional **state-variable filter** with three modes: Low-Pass, High-Pass, and Band-Pass. Filters shape the frequency content of sounds, removing unwanted frequencies whilst emphasising others.

Filter Parameters:

- 1. **Cutoff Frequency**: The frequency at which the filter begins to attenuate (reduce volume)
- 2. **Resonance**: Emphasis (boost) at the cutoff frequency, creating a peak
- 3. **Filter Type**: Low-Pass, High-Pass, or Band-Pass

1. Low-Pass Filter (LP)

Function: Allows frequencies **below** the cutoff to pass through, attenuates frequencies **above** the cutoff.

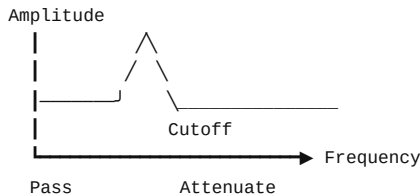
Effect on Sound:

- **High cutoff** (10kHz-20kHz): Minimal effect, sound remains bright
- **Medium cutoff** (1kHz-5kHz): Removes high frequencies, creates mellow, warm tone
- **Low cutoff** (100Hz-500Hz): Removes most frequencies, creates muffled, bass-only sound

Musical Applications:

- **Classic synthesiser sound:** Sawtooth wave + low-pass filter = iconic analogue lead/bass
- **Taming bright sounds:** Reduce harshness in bright waveforms (sawtooth, square)
- **Filter sweeps:** Automate cutoff frequency for dramatic timbral changes (wah-wah effect)
- **Emulating distance:** Lower cutoff simulates sound heard from far away

Visualisation:



2. High-Pass Filter (HP)

Function: Allows frequencies **above** the cutoff to pass through, attenuates frequencies **below** the cutoff.

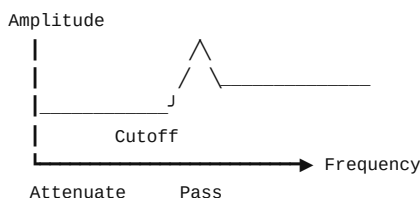
Effect on Sound:

- **Low cutoff** (20Hz-100Hz): Minimal effect, removes only sub-bass rumble
- **Medium cutoff** (200Hz-1kHz): Removes bass and lower midrange, creates thin, nasal tone
- **High cutoff** (2kHz-10kHz): Removes most frequencies, leaves only high-frequency content (brightness, air)

Musical Applications:

- **Removing muddiness:** Clean up bass-heavy sounds
- **Creating thin, telephone-like effects:** High cutoff for lo-fi, vintage character
- **Layering:** High-pass one sound, low-pass another, blend for full-spectrum texture
- **Emulating small speakers:** High-pass filter simulates tiny speaker or telephone

Visualisation:



3. Band-Pass Filter (BP)

Function: Allows frequencies **around** the cutoff to pass through, attenuates frequencies **below** and **above** the cutoff.

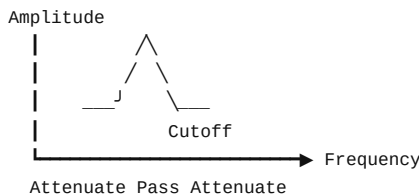
Effect on Sound:

- **Narrow band:** Creates focused, nasal, vocal-like quality (formant filter)
- **Wide band:** Creates balanced, midrange-focused sound
- **Swept band:** Creates dramatic filtering effects (vowel sounds, talking synthesiser)

Musical Applications:

- **Vocal formants:** Simulate vowel sounds (a, e, i, o, u) by sweeping cutoff
- **Telephone/radio effects:** Band-pass around 1kHz simulates telephony bandwidth
- **Emphasising midrange:** Focus attention on specific frequency range
- **Special effects:** Swept band-pass creates distinctive electronic textures

Visualisation:



Resonance (Q Factor)

Resonance adds emphasis (a peak) at the cutoff frequency, creating a distinctive character:

- **Low resonance** (0.0-1.0): Gentle, natural-sounding filtering
- **Medium resonance** (1.5-2.5): Emphasised cutoff, wah-wah character
- **High resonance** (3.0-4.0): Dramatic peak, self-oscillation (pure sine tone at cutoff frequency)

Self-Oscillation:

At maximum resonance (~ 4.0), the filter can produce a **pure sine wave tone** at the cutoff frequency, even without input signal. This is useful for:

- Creating sine wave leads (filter becomes an oscillator)
- Adding high-pitched whistles to sounds
- Special effects and experimental textures

Warning: High resonance can produce very loud peaks. Reduce overall volume to prevent clipping.

Filter Control via MIDI:

- **Cutoff Frequency:** Controlled per-patch or via NRPN parameters
- **Resonance:** Controlled per-patch or via NRPN parameters
- **Filter Type:** Set per-patch (LP/HP/BP/Off)
- **Keyboard Tracking:** Filter cutoff follows note pitch (higher notes = higher cutoff)

Filter Sweeps:

One of the most expressive filter techniques is the **filter sweep**, where cutoff frequency changes over time:

1. **Manual sweep:** Use MIDI controller or NRPN to change cutoff in real-time
2. **Envelope sweep:** Cutoff follows ADSR envelope (opens on attack, closes on decay)
3. **LFO sweep:** Cutoff modulated by Low-Frequency Oscillator (cyclic wah-wah)

Filter sweeps are explored in depth in Part III: Sound Programming.

Effects Overview

The Intuition Subsynth includes two categories of effects: **per-channel effects** (applied to individual synthesis channels) and **master effects** (applied to the final stereo mix).

Per-Channel Effects (Applied to Individual Channels):

1. **Delay:** Simple delay line with configurable time and feedback
2. **Comb Filter:** Resonant delay creating metallic, harmonic timbres
3. **Flanger:** Modulated short delay creating sweeping, jet-plane effect
4. **Phaser:** Allpass filter creating notches that sweep through spectrum
5. **Bit-Crusher:** Digital distortion reducing bit depth for lo-fi character
6. **Overdrive:** Soft-clipping distortion adding harmonic richness

7. **PWM:** Pulse Width Modulation for square waves (thickness variation)
8. **Vibrato:** Pitch modulation (LFO) creating expressive pitch wobble
9. **Tremolo:** Amplitude modulation (LFO) creating volume pulsation

Master Effects (Applied to Stereo Mix):

1. **Reverb:** Stereo reverb (comb filter + allpass network) creating spaciousness
2. **Chorus:** Stereo chorus (modulated delay) creating width and thickness
3. **Ping-Pong Delay:** Stereo delay with left-right alternation
4. **Compressor/Limiter:** Dynamics control preventing clipping and adding punch
5. **Master Filter:** State-variable filter affecting entire mix

Per-Channel Effects Explained:

1. Delay

A simple echo effect that repeats the sound after a specified time:

- **Delay Time:** 0-2000ms (0-2 seconds)
- **Feedback:** 0-95% (higher feedback = more repeats)
- **Mix:** 0-100% (dry/wet balance)

Musical uses: Slapback echo (50-150ms), rhythmic delay (1/4 note, 1/8 note), ambient tails

2. Comb Filter

A delay line with high feedback that creates resonant peaks at harmonic intervals:

- **Frequency:** 20Hz-20kHz (determines harmonic spacing)
- **Feedback:** 0-95% (higher feedback = stronger resonance)
- **Mix:** 0-100%

Musical uses: Metallic timbres, robotic voices, string simulation, flanging effects

3. Flanger

A modulated short delay (0-20ms) creating sweeping notches in the frequency spectrum:

- **Rate:** 0.1-10Hz (modulation speed)
- **Depth:** 0-100% (modulation intensity)
- **Feedback:** 0-95% (resonance, metallic character)

Musical uses: Jet-plane whoosh, psychedelic sweeps, 1970s guitar effects

4. Phaser

A series of allpass filters creating notches that sweep through the spectrum:

- **Rate:** 0.1-10Hz (modulation speed)
- **Depth:** 0-100% (modulation intensity)
- **Stages:** 2-8 allpass filters (more stages = more notches)

Musical uses: Subtle movement, vintage keyboard sounds, guitar effects, psychedelic textures

5. Bit-Crusher

Digital distortion that reduces bit depth, creating quantisation noise and harmonic distortion:

- **Bit Depth:** 1-16 bits (lower = more distortion)
- **Sample Rate Reduction:** 1x-32x (higher = more aliasing)

- **Mix:** 0-100%

Musical uses: Lo-fi character, 8-bit game sounds, aggressive digital distortion, degraded audio

6. Overdrive

Soft-clipping distortion that adds harmonic richness without harsh clipping:

- **Drive:** 0-100% (distortion amount)
- **Mix:** 0-100%

Musical uses: Warm saturation, vintage amplifier character, bass enhancement, aggressive leads

7. PWM (Pulse Width Modulation)

Modulation of square wave duty cycle, creating chorused, string-like textures:

- **Rate:** 0.1-20Hz (modulation speed)
- **Depth:** 0-100% (duty cycle variation)

Musical uses: Thick synthesiser strings, chorused leads, animated textures (square waves only)

8. Vibrato

Pitch modulation creating expressive pitch wobble:

- **Rate:** 0.1-20Hz (vibrato speed, typically 4-7Hz)
- **Depth:** 0-100 cents (pitch deviation, typically 10-50 cents)

Musical uses: Expressive sustained notes, realistic vocal/string simulation, synthesiser leads

9. Tremolo

Amplitude modulation creating rhythmic volume pulsation:

- **Rate:** 0.1-20Hz (tremolo speed)
- **Depth:** 0-100% (volume modulation amount)

Musical uses: Vintage amplifier effect, rhythmic pulsation, helicopter effect

Master Effects Explained:

1. Reverb (Master)

Stereo reverb creating spaciousness and ambience:

- **Reverb Send** (CC91): 0-127 (per-channel reverb send level)
- **Room Size:** Small/Medium/Large (configurable via NRPN)
- **Decay Time:** 0.5-10 seconds
- **Damping:** High-frequency absorption (simulates room materials)

Musical uses: Adding depth and space, simulating concert halls, ambient textures

2. Chorus (Master)

Stereo chorus creating width and thickness:

- **Chorus Send** (CC93): 0-127 (per-channel chorus send level)
- **Rate:** 0.1-10Hz (modulation speed)
- **Depth:** 0-100% (modulation intensity)
- **Stereo Width:** 0-100% (left/right phase offset)

Musical uses: Thickening sounds, stereo width, vintage synthesiser character, string ensembles

3. Ping-Pong Delay (Master)

Stereo delay alternating between left and right channels:

- **Left Delay:** 1-2000ms
- **Right Delay:** 1-2000ms (typically different from left for ping-pong effect)
- **Feedback:** 0-95%
- **Mix:** 0-100%

Musical uses: Wide stereo effects, rhythmic echoes, ambient soundscapes

4. Compressor/Limiter (Master)

Dynamics control preventing clipping whilst adding punch:

- **Threshold:** -60dB to 0dB (level at which compression begins)
- **Ratio:** 1:1 to ∞ :1 (compression amount; ∞ :1 = limiting)
- **Attack:** 0-100ms (how quickly compressor responds)
- **Release:** 10-1000ms (how quickly compressor returns to normal)

Musical uses: Preventing clipping, adding punch to drums, gluing mix together, loudness maximisation

5. Master Filter

State-variable filter applied to entire mix:

- **Type:** Low-Pass, High-Pass, Band-Pass, Off
- **Cutoff:** 20Hz-20kHz
- **Resonance:** 0.0-4.0

Musical uses: Global tone shaping, dramatic filter sweeps, special effects

Effects Signal Flow:

Individual Channel:

Oscillator → Per-Channel Filter → Per-Channel Effects → Channel Output

Mix Bus:

All Channel Outputs → Mixer → Master Effects → Stereo Output

Configuring Effects:

Effects are controlled via:

1. **Per-Patch Settings:** Each GM programme has pre-configured effects
2. **MIDI Continuous Controllers:** CC91 (Reverb Send), CC93 (Chorus Send)
3. **NRPN Parameters:** Detailed control of all effect parameters (see Part IV: MIDI Implementation)

Layered Patches

Some sounds in the Intuition Subsynth use **two waveforms simultaneously** (layered patches) to create richer, more complex timbres. Layering is a classic synthesiser technique for achieving sounds that single waveforms cannot produce alone.

How Layering Works:

1. **Primary Waveform:** The main waveform that defines the sound's character
2. **Layer Waveform:** A second waveform blended with the primary for added richness

3. **Independent Processing:** Each waveform layer can have independent:

- Pitch offset (detuning)
- Volume level
- Filter settings
- Envelope settings
- Effects

Example Layered Patches:

Sound	Primary	Layer	Effect
Rich Brass	Sawtooth	Square	Adds body and thickness
Lush Strings	Sawtooth	Triangle	Smooths harsh edges
Fat Bass	Triangle	Sine	Adds sub-bass weight
Detuned Lead	Sawtooth	Sawtooth	Detuned copy creates chorused sound
Electric Piano	Triangle	Noise	Adds attack transient (hammer noise)

Detuning:

A common layering technique is **detuning**, where the layer waveform is slightly offset in pitch from the primary:

- **Detune Amount:** ± 50 cents (half-semitone) typical
- **Effect:** Creates chorused, "fat" sound due to beating between frequencies
- **Musical Use:** Thickens leads, creates vintage analogue character

Polyphony Cost:

Layered patches consume **two channels per note** (one from each waveform pool). This halves maximum polyphony:

- **Single-waveform patch:** Maximum 64 simultaneous notes (64 channels from one pool)
- **Layered patch:** Maximum 64 simultaneous notes (128 channels from two pools)

Voice Stealing with Layered Patches:

When playing layered patches:

1. Both waveform pools must have available channels
2. If either pool is full, voice stealing occurs in that pool
3. The other pool continues unaffected

Example: Playing 65 notes of "Rich Brass" (sawtooth + square):

- Sawtooth pool (256-319): Steals oldest sawtooth note (note 1)
- Square pool (0-63): Steals oldest square note (note 1)
- Result: Note 1 stops entirely (both layers stolen), note 65 plays

Identifying Layered Patches:

You can determine if a programme uses layering by:

1. Consulting the programme list in Part V: Sound Banks (indicates "layered" if applicable)
2. Listening carefully for multiple waveform timbres
3. Using NRPN parameters to query patch configuration (advanced)

Creating Custom Layers (Advanced):

Whilst factory patches come with pre-configured layering, advanced users can create custom layers using NRPN parameters:

1. Set primary waveform type
2. Enable layer (layer enable flag)
3. Set layer waveform type
4. Adjust layer level (0-100%)
5. Set layer detune (± 1200 cents)

See Part III: Sound Programming and Part IV: MIDI Implementation for detailed instructions.

Real-Time Performance Controls

The Intuition Subsynth responds to standard MIDI performance controllers, allowing expressive real-time control during playing. Bank selection follows the General MIDI convention where **CC#0 (Bank Select MSB)** carries the bank number; **CC#32 (Bank Select LSB)** is accepted but ignored, so sending CC#0 alone is sufficient to change patchsets.

Essential Performance Controllers:

1. Channel Volume (CC7)

- **Range:** 0-127 (0 = silent, 127 = maximum)
- **Function:** Master volume for the MIDI channel
- **Use:** Set overall volume balance between MIDI channels

2. Expression (CC11)

- **Range:** 0-127 (0 = silent, 127 = maximum)
- **Function:** Dynamic volume control (complementary to CC7)
- **Use:** Real-time volume changes during performance (swells, fades)
- **Difference from CC7:** CC7 is "set and forget" master level; CC11 is dynamic performance control

3. Pan (CC10)

- **Range:** 0-127 (0 = hard left, 64 = centre, 127 = hard right)
- **Function:** Stereo positioning
- **Use:** Place sounds in stereo field for clarity and width

4. Modulation Wheel (CC1)

- **Range:** 0-127 (0 = no vibrato, 127 = maximum vibrato)
- **Function:** Controls vibrato depth (pitch modulation)
- **Use:** Add expressive vibrato to sustained notes

5. Sustain Pedal (CC64)

- **Range:** 0-63 = off, 64-127 = on (switch, not continuous)
- **Function:** Sustains notes after key release
- **Use:** Legato playing, piano-style sustain

6. Reverb Send (CC91)

- **Range:** 0-127 (0 = no reverb, 127 = maximum reverb)
- **Function:** Controls per-channel reverb send level
- **Use:** Add depth and space to sounds

7. Chorus Send (CC93)

- **Range:** 0-127 (0 = no chorus, 127 = maximum chorus)
- **Function:** Controls per-channel chorus send level
- **Use:** Thicken sounds, add stereo width

8. Attack Time (CC73)

- **Range:** 0-127 (0 = instant, 127 = 11.7 seconds)
- **Function:** Controls ADSR attack time
- **Use:** Change attack character in real-time (percussive to slow-building)

9. Release Time (CC72)

- **Range:** 0-127 (0 = instant, 127 = 15 seconds)
- **Function:** Controls ADSR release time
- **Use:** Change release character in real-time (staccato to lingering)

10. Pitch Bend

- **Range:** 0-16383 (14-bit, centre = 8192)
- **Default Range:** ± 2 semitones (GM standard)
- **SIDney Range:** ± 12 semitones (C64-style)
- **Function:** Continuous pitch control
- **Use:** Expressive pitch slides, vibrato, special effects

Performance Controller Scenarios:

Scenario 1: Expressive Lead Solo

1. Set CC7 (Channel Volume) = 100 (overall level)
2. Use CC11 (Expression) for dynamic swells (0 $\rightarrow$ 127 during sustained notes)
3. Use CC1 (Modulation Wheel) for vibrato on sustained notes
4. Use Pitch Bend for expressive pitch slides
5. Result: Dynamic, expressive lead performance

Scenario 2: Lush Pad with Depth

1. Set CC7 (Channel Volume) = 80 (moderate level)
2. Set CC91 (Reverb Send) = 100 (heavy reverb)
3. Set CC93 (Chorus Send) = 80 (moderate chorus)
4. Set CC73 (Attack Time) = 100 (slow attack, ~2 seconds)
5. Set CC72 (Release Time) = 110 (long release, ~5 seconds)
6. Result: Ambient, spacious pad sound

Scenario 3: Tight Rhythmic Bass

1. Set CC7 (Channel Volume) = 110 (loud, prominent)
2. Set CC73 (Attack Time) = 0 (instant attack)
3. Set CC72 (Release Time) = 20 (short release, ~100ms)
4. Set CC64 (Sustain Pedal) = 0 (off, tight articulation)
5. Result: Punchy, rhythmic bass with tight envelope

Per-Channel Independence:

All performance controllers operate **per-MIDI-channel**:

- Each of 16 MIDI channels has independent CC values
- You can have different volume, pan, reverb, etc. on each channel
- Channel 10 (drums) typically uses different settings than melodic channels

Part III: Sound Programming

10. Synthesis Parameters

Oscillator Parameters

The Intuition Subsynth's oscillator section generates the raw waveforms that form the foundation of every sound. Understanding oscillator parameters allows you to shape the basic character of your sounds.

Primary Oscillator Controls:

1. Waveform Selection

Each GM programme has a pre-assigned primary waveform type:

- Square (0): Bright, hollow, classic synthesiser leads
- Triangle (1): Smooth, mellow, vintage chip sounds
- Sine (2): Pure, warm, sub-bass and FM synthesis
- Noise (3): Drums, percussion, sound effects
- Sawtooth (4): Bright, rich, analogue synthesiser classic

Changing Waveform (Advanced): Whilst GM programmes come with pre-assigned waveforms, advanced users can override the waveform type using NRPN parameters (see Part IV: MIDI Implementation).

2. Pitch and Tuning

Coarse Tuning:

- Range: ± 24 semitones (± 2 octaves)
- Resolution: 1 semitone per step
- Use: Transpose sounds up or down by octaves or semitones

Fine Tuning:

- Range: ± 100 cents (± 1 semitone)
- Resolution: 1 cent per step
- Use: Precise pitch adjustment for tuning or detuning effects

Master Tuning (A440 Reference):

- Default: A4 = 440Hz (concert pitch)
- Adjustable: 415Hz-466Hz (baroque to high pitch)
- Use: Match historical tunings or ensemble tuning references

3. Pulse Width (Square Waves Only)

Pulse width determines the duty cycle of square waves (percentage of time the waveform is "high"):

- **Range:** 1%-99%
- **Default:** 50% (balanced square wave)
- **NRPN Control:** NRPN 160 (Square Duty Fine)

Pulse Width Effects on Timbre:

Duty Cycle	Character	Harmonic Content
50%	Balanced, classic square wave	Odd harmonics only

25%/75%	Thin, nasal, reed-like	More odd harmonics
10%/90%	Very thin, hollow	Many odd harmonics
5%/95%	Extremely thin, almost silent	Approaching silence

Musical Applications:

- **50%:** Classic synthesiser leads, bass, chip music
- **20-30%:** Thin, nasally leads, oboe-like timbres
- **Modulated (PWM):** Chorused strings, animated textures

4. Pulse Width Modulation (PWM)

PWM automatically varies the pulse width over time, creating rich, chorused textures:

PWM Parameters:

- **Enable:** On/off switch
- **Rate:** 0.1-20Hz (modulation speed, typical 0.5-2Hz)
- **Depth:** 0-100% (amount of width variation, typical 20-40%)

NRPN Controls:

- NRPN 161: PWM Depth Fine
- NRPN 162: PWM Rate Fine

PWM Waveform: The PWM uses a sine wave LFO to smoothly vary pulse width. At 50% depth and 50% centre width, the pulse width sweeps from 25% to 75% continuously.

Musical Applications:

- **String Ensembles:** Slow PWM (0.5-1Hz, 30% depth) creates lush, chorused strings
- **Animated Leads:** Medium PWM (2-4Hz, 20% depth) adds movement to static sounds
- **Special Effects:** Fast PWM (10-20Hz, 50% depth) creates vibrato-like effects

Example Settings:

Sound	PWM Rate	PWM Depth	Character
String Ensemble	0.5Hz	30%	Slow, lush chorus
Synthesiser Pad	1.0Hz	40%	Gentle movement
Animated Lead	3.0Hz	20%	Rhythmic pulse
Special Effect	15Hz	50%	Tremolo-like vibrato

Hard Sync (Oscillator Sync)

Hard sync is an advanced oscillator technique where one oscillator (the "master") resets another oscillator's (the "slave") phase, creating complex harmonic timbres.

How Hard Sync Works:

1. **Master Oscillator:** Runs at the fundamental frequency (e.g. 100Hz)
2. **Slave Oscillator:** Runs at a different frequency (e.g. 250Hz)

3. **Sync Action:** Every time the master completes one cycle (wraps from 2π to 0), it forces the slave to reset to phase 0
4. **Result:** The slave's waveform is "interrupted" mid-cycle, creating discontinuities that generate additional harmonics

Spectral Effects:

- **Adds harmonics** at multiples of the master frequency
- **Slave frequency** controls which harmonics are emphasised
- **Creates formant-like resonances** (peaks in the frequency spectrum)
- **Non-harmonic** when slave frequency is not an integer multiple of master

Musical Applications:

- **Tearing Lead Sounds:** Classic analogue synthesiser leads with aggressive character
- **Formant Synthesis:** Vowel-like timbres by sweeping slave frequency
- **Bell Sounds:** Non-harmonic spectra create metallic, bell-like tones
- **Special Effects:** Dramatic timbral changes with frequency sweeps

Configuring Hard Sync:

NRPN Control: Sync Source register (SYNC_SOURCE_BASE)

- Address: $\text{NRPN } 1000 + (2 \times 320) + \text{channel_index}$
- Value: Channel number of master oscillator (0-319)
- Example: Set channel 5 to sync from channel 3:
 - $\text{NRPN} = 1000 + (2 \times 320) + 5 = 1645$
 - Value = 3

Frequency Ratios:

Ratio	Effect	Musical Use
1:1	No effect (same frequency)	Disabled
1:2	Octave sync, strong harmonics	Bright leads
1:3	Fifth sync, odd harmonics	Hollow, vocal-like
1:1.5	Non-integer, inharmonic	Bell-like, metallic
1:variable	Swept sync (changing slave frequency)	Dramatic timbral sweeps

Safety Constraints:

- **Circular dependency detection:** Prevents channel A syncing to B whilst B syncs to A
- **Frequency ratio limits:** Slave frequency should be 0.1x to 10x master frequency
- **Voice allocation:** Master and slave must both be active (no stolen voices)

Ring Modulation

Ring modulation is a classic effect that multiplies two waveforms together, creating complex, inharmonic spectra ideal for metallic and bell-like tones.

How Ring Modulation Works:

1. **Carrier:** The primary oscillator frequency (e.g. 440Hz, A4)
2. **Modulator:** A second oscillator frequency (e.g. 220Hz, A3)

- 3. **Multiplication:** Output = carrier × modulator
- 4. **Result:** Two sidebands:
 - **Upper sideband:** carrier_freq + modulator_freq (440Hz + 220Hz = 660Hz)
 - **Lower sideband:** |carrier_freq - modulator_freq| (|440Hz - 220Hz| = 220Hz)
 - **Original frequencies removed:** Only sidebands remain (creates inharmonic spectrum)

Spectral Result:

Traditional harmonic sounds have frequencies at integer multiples of the fundamental (f, 2f, 3f, 4f...). Ring modulation creates **inharmonic spectra** (non-integer relationships), producing metallic, bell-like, or robotic timbres.

Musical Applications:

- **Bell Sounds:** Inharmonic spectra create realistic bell and chime tones
- **Metallic Percussion:** Gongs, cymbals, metallic hits
- **Robotic Voices:** Classic "Dalek" or vocoder-style effects
- **Special Effects:** Sci-fi sounds, alien textures

Ring Modulation Sources:

The Intuition Subsynth supports two ring modulation sources:

- 1. **Internal Carrier Oscillator:**
 - Built-in oscillator running at configurable frequency (20Hz-20kHz)
 - Independent waveform selection (sine, square, triangle, sawtooth)
 - Does not consume a synthesis channel
- 2. **External Source Channel:**
 - Use another synthesis channel as the modulator
 - Allows complex modulation schemes (e.g. channel 5 modulates channel 10)
 - Both channels must be active

Configuring Ring Modulation:

NRPN Controls:

- 1. **Ring Mod Source** (RING_MOD_SOURCE_BASE):
 - Address: NRPN 1000 + (6 × 320) + channel_index
 - Value: Source channel number (0-319) or 0xFFFF (internal carrier)
- 2. **Internal Carrier Frequency** (if using internal carrier):
 - NRPN 170: Carrier frequency (20Hz-20kHz)
- 3. **Internal Carrier Waveform:**
 - NRPN 171: Waveform type (0=sine, 1=square, 2=triangle, 4=sawtooth)

Frequency Relationship Guidelines:

Carrier:Modulator	Effect	Musical Use
1:1	Octave doubling	Simple harmonic
2:1	Perfect fifth sidebands	Musical intervals

3:2	Major sixth sidebands	Consonant bells
5:4	Major third sidebands	Bright bells
Irrational	Inharmonic, complex	Metallic, non-musical

Example Configuration:

Create a bell sound on channel 128 (sine wave):

1. Enable internal carrier: NRPN 3920 (1000 + 6×320 + 128) = 0xFFFF
2. Set carrier frequency: NRPN 170 = 880Hz (A5, two octaves above A3 note)
3. Set carrier waveform: NRPN 171 = 0 (sine wave)
4. Play note A3 (220Hz) on channel 128
5. Result: Ring mod produces 660Hz and 1100Hz sidebands (bell-like inharmonic spectrum)

Advanced Techniques:

- **Swept Ring Mod:** Automate carrier frequency for dramatic timbral evolution
- **Multiple Modulators:** Use different channels as modulators for complex spectra
- **Ring Mod + Reverb:** Add spaciousness to metallic tones for gongs and ambient bells

11. Modulation and LFOs

Vibrato System

Vibrato is periodic pitch modulation that adds expressiveness and life to sustained notes. The Intuition Subsynth implements professional vibrato with smooth, musical characteristics.

Vibrato Parameters:

1. Rate (Frequency):

- Range: 0.1Hz - 20Hz
- Musical Range: 4Hz - 7Hz (natural vocal/string vibrato)
- MIDI Control: Set per-patch or via NRPN

2. Depth (Amount):

- Range: 0 - 100 cents (±50 cents)
- Musical Range: 10 - 50 cents (subtle to pronounced)
- MIDI Control: CC1 (Modulation Wheel) controls depth in real-time

Vibrato Waveform:

The Intuition Subsynth uses a **sine wave** LFO for vibrato, producing smooth, natural-sounding pitch modulation (no abrupt changes).

Vibrato vs. Pitch Bend:

Feature	Vibrato (CC1)	Pitch Bend
Waveform	Sine wave (cyclic)	Manual (continuous)
Rate	0.1-20Hz (automatic)	Performer-controlled
Depth	0-100 cents	±2 to ±24 semitones

Musical Use	Sustained note expression	Glissando, slides, bends
-------------	---------------------------	--------------------------

Musical Applications:

- **Vocal Simulation:** 5-6Hz vibrato at 20-30 cents depth simulates natural singing
- **String Instruments:** 6-7Hz vibrato at 15-25 cents depth simulates violin/cello
- **Synthesiser Leads:** 4-5Hz vibrato at 40-50 cents depth adds vintage character
- **Organ:** No vibrato (0 cents) for stable, pure tones

Real-Time Vibrato Control:

The modulation wheel (CC1) provides real-time control over vibrato depth:

1. Set base vibrato rate via patch (e.g. 5Hz)
2. Play and hold a note
3. Modulation wheel at 0: No vibrato (depth = 0 cents)
4. Gradually increase modulation wheel to 127: Vibrato depth increases from 0 to maximum (e.g. 50 cents)
5. Decrease modulation wheel: Vibrato depth decreases smoothly

Vibrato Delay (Advanced):

Some professional synthesisers implement vibrato delay (vibrato starts after a note has been held for a specified time). The Intuition Subsynth currently applies vibrato immediately when CC1 > 0. Vibrato delay may be added in future firmware updates.

Tremolo System

Tremolo is periodic amplitude modulation that creates rhythmic volume pulsation. Unlike vibrato (pitch modulation), tremolo modulates volume.

Tremolo Parameters:

1. Rate (Frequency):

- Range: 0.1Hz - 20Hz
- Musical Range: 4Hz - 8Hz (rhythmic pulsation)
- MIDI Control: Set per-patch or via NRPN

2. Depth (Amount):

- Range: 0% - 100%
- Musical Range: 20% - 60% (subtle to pronounced)
- MIDI Control: Via NRPN (no standard MIDI CC)

Tremolo Waveform:

The Intuition Subsynth uses a **sine wave** LFO for tremolo, producing smooth volume modulation.

Musical Applications:

- **Vintage Amplifier Effect:** 4-6Hz tremolo at 40-60% depth simulates classic guitar amplifier tremolo
- **Rhythmic Pulsation:** 8-16Hz tremolo creates fast, rhythmic effects
- **Helicopter Effect:** Very fast tremolo (>16Hz) creates helicopter-like amplitude modulation
- **Subtle Movement:** Slow tremolo (0.5-2Hz) at low depth (10-20%) adds gentle animation

Tremolo vs. Expression (CC11):

Feature	Tremolo	Expression (CC11)
---------	---------	-------------------

Control	Automatic (LFO)	Manual (performer)
Waveform	Sine wave (cyclic)	Continuous/stepped
Rate	0.1-20Hz	Performer-controlled
Musical Use	Rhythmic pulsation	Dynamic swells/fades

LFO System

Low-Frequency Oscillators (LFOs) are periodic modulation sources that operate below the audible range (typically <20Hz). The Intuition Subsynth's LFO system provides flexible modulation routing to multiple destinations.

LFO Parameters:

1. Rate (Frequency):

- Range: 0.01Hz - 20Hz
- Musical Range: 0.1Hz - 10Hz
- MIDI Control: Via NRPN or tempo-sync (musical divisions)

2. Waveform:

- Sine: Smooth, natural modulation (vibrato, chorus)
- Triangle: Linear rise/fall (ramp up/down effects)
- Square: Abrupt on/off modulation (trill effects)
- Sawtooth: Linear ramp (rising/falling sweeps)
- Random: Stepped random values (sample & hold effects)

3. Phase:

- Range: 0° - 360°
- Use: Offset LFO starting position (stereo effects, polysynth detuning)

Tempo Synchronisation:

The LFO can synchronise to MIDI clock for rhythmic modulation aligned with tempo:

Musical Divisions:

- 1/1: Whole note (one cycle per bar in 4/4)
- 1/2: Half note (two cycles per bar)
- 1/4: Quarter note (four cycles per bar)
- 1/8: Eighth note (eight cycles per bar)
- 1/16: Sixteenth note (sixteen cycles per bar)
- 1/32: Thirty-second note (thirty-two cycles per bar)

Example: With tempo = 120 BPM, 1/4 note sync = 2Hz LFO (120 BPM ÷ 60 = 2 Hz)

LFO Destinations:

1. Pitch (Vibrato):

- Depth: 0-100 cents
- Effect: Periodic pitch modulation (vibrato)
- Musical Use: Expressive sustained notes

2. Amplitude (Tremolo):

- Depth: 0-100%
- Effect: Periodic volume modulation (tremolo)
- Musical Use: Rhythmic pulsation, vintage amplifier effects

3. Filter Cutoff (Filter Sweep):

- Depth: 0-100% (percentage of cutoff range)
- Effect: Periodic filter sweeps (wah-wah)
- Musical Use: Animated textures, rhythmic filtering

4. Pulse Width (PWM):

- Depth: 0-100% (duty cycle range)
- Effect: Chorused, string-like textures (square waves only)
- Musical Use: Synthesiser strings, animated leads

5. Pan:

- Depth: 0-100% (left-right range)
- Effect: Auto-pan (stereo movement)
- Musical Use: Wide stereo effects, rhythmic spatial movement

Multiple LFO Routing:

Advanced patches can route a single LFO to multiple destinations simultaneously:

Example: Single LFO modulating pitch + filter:

- LFO Rate: 5Hz (sine wave)
- Pitch Depth: 20 cents (subtle vibrato)
- Filter Depth: 30% (gentle filter sweep)
- Result: Simultaneous vibrato and filter animation, creating rich, evolving texture

Filter Sweeps

Filter sweeps are one of the most expressive modulation techniques, creating dramatic timbral changes by varying the filter cutoff frequency over time.

Filter Sweep Sources:

1. Manual Sweep (MIDI Controller):

- Send NRPN parameter changes to filter cutoff in real-time
- Performer controls sweep speed and range
- Use: Live performance, manual expression

2. Envelope Sweep (Envelope → Filter):

- Filter cutoff follows ADSR envelope
- Typical: Cutoff opens on attack, closes on decay
- Use: Dynamic sounds that change over time (plucked sounds, synth bass)

3. LFO Sweep (LFO → Filter):

- Filter cutoff modulated by LFO
- Creates cyclic wah-wah effect
- Use: Rhythmic filtering, animated textures

Filter Envelope Modulation:

Parameters:

1. Envelope Amount:

- Range: -100% to +100%
- Positive: Envelope opens filter (brighter on attack)
- Negative: Envelope closes filter (darker on attack)
- NRPN Control: NRPN 251 (Filter Envelope Amount)

2. Envelope Velocity Sensitivity:

- Range: 0% - 100%
- Effect: Higher velocity = more envelope modulation
- NRPN Control: NRPN 252 (Filter Velocity Sensitivity)

Typical Settings:

Sound Type	Base Cutoff	Env Amount	Env Velocity	Character
Plucked Bass	500Hz	+80%	60%	Bright attack, dark sustain
Synth Lead	2kHz	+50%	40%	Opening brightness
Pad	1kHz	+20%	10%	Subtle animation
Organ	5kHz	0%	0%	Static brightness

Filter LFO Modulation:

Parameters:

1. LFO Amount:

- Range: 0% - 100% (percentage of cutoff range)
- Effect: How much the LFO moves the filter
- NRPN Control: Via LFO destination depth

2. LFO Rate:

- Range: 0.1Hz - 20Hz
- Musical: 0.5Hz - 8Hz (typical for filter sweeps)
- NRPN Control: Via LFO rate parameter

3. LFO Waveform:

- Sine: Smooth, natural wah-wah
- Triangle: Linear up/down sweep
- Sawtooth: Ramping sweep (rising or falling)
- Square: Abrupt filter switching

Example Settings:

Effect	LFO Rate	LFO Amount	LFO Wave	Character
Gentle Wah	1Hz	30%	Sine	Smooth, musical
Rhythmic Filter	4Hz	50%	Triangle	Rhythmic sweep
Aggressive Wah	8Hz	70%	Sine	Fast, dramatic

Filter Trill	16Hz	40%	Square	Rapid switching
--------------	------	-----	--------	-----------------

Keyboard Tracking:

Keyboard tracking causes the filter cutoff to follow the pitch of the note being played:

- **0% tracking:** Filter cutoff constant (same for all notes)
- **100% tracking:** Filter cutoff follows note pitch exactly (higher notes = higher cutoff)
- **50% tracking:** Filter cutoff increases at half the rate of note pitch

Musical Applications:

- **100% tracking:** Maintains consistent timbre across keyboard range (useful for leads)
- **0% tracking:** Creates darker high notes, brighter low notes (fixed-filter character)
- **50% tracking:** Compromise (slightly darker highs, slightly brighter lows)

NRPN Control: NRPN 250 (Filter Keyboard Track)

Filter Sweep Performance Techniques:

1. Manual Wah-Wah:

- Route NRPN cutoff parameter to expression pedal or MIDI controller
- Performer controls filter sweep in real-time
- Classic effect for funk, disco, electronic music

2. Envelope-Controlled Brightness:

- High envelope amount (+70% to +90%)
- Base cutoff set low (200Hz - 1kHz)
- Result: Bright attack transient, dark sustain (plucked bass, percussive synth)

3. LFO Auto-Wah:

- LFO rate synced to tempo (e.g. 1/4 note)
- LFO amount 40-60%
- Result: Rhythmic filter sweeps locked to song tempo

4. Swept Pad:

- Very slow LFO (0.1Hz - 0.5Hz)
- Sine wave, 20-40% depth
- Result: Gently evolving pad texture

12. Effects

The Intuition Subsynth features a comprehensive effects system organised into two categories: **per-channel effects** (applied individually to each synthesis channel before mixing) and **master effects** (applied to the final stereo mix after all channels are combined).

This architecture provides immense creative flexibility:

- **Per-channel effects** sculpt individual voices before they reach the mixer
- **Master effects** add spatial depth, ambience, and dynamic control to the final mix
- Total effects processing: 9 per-channel effects + 5 master effects = 14 independent effect types

Effect Categories

Per-Channel Effects (9 types):

Effect	Category	Primary Use
Bit Crusher	Lo-Fi	Vintage digital/chip sound
Comb Filter	Tonal/Metallic	Metallic resonances, body simulation
Delay Line	Time-Based	Echoes, rhythmic repeats
Flanger	Modulation	Jet-plane sweeps, chorusing
Phaser	Modulation	Sweeping notches, vocal-like
Overdrive/Distortion	Saturation	Warmth, aggression, grit
Ring Modulation	Inharmonic	Bells, metallic, robotic
Hard Sync	Harmonic	Harsh harmonics, aggressive leads
PWM	Tonal	String-like chorus (square waves)

Master Effects (5 types):

Effect	Category	Primary Use
Reverb	Spatial	Room ambience, space, depth
Chorus	Modulation	Thickening, shimmer, width
Ping-Pong Delay	Time-Based	Stereo rhythmic echoes
Compressor	Dynamics	Level control, punch, glue
Limiter	Dynamics	Final safety limiting (automatic)

Per-Channel Effects

Per-channel effects are applied **before mixing**, allowing each of the 320 synthesis channels to have its own effect settings. This enables complex layered textures where different voices have different sonic treatments.

Important: Per-channel effects are **not** routed through CC messages. They require **NRPN control** for enabling/disabling and parameter adjustment.

Bit Crusher

The **bit crusher** reduces bit depth and sample rate to create vintage digital, lo-fi, and retro chip-style sounds.

Parameters:

1. Bit Depth Reduction:

- Range: 1-16 bits
- Effect: Reduces amplitude resolution, creating quantisation noise
- NRPN Control: Per-channel parameter (channel-specific)

Bit Depth Character:

Bit Depth	Character	Application
-----------	-----------	-------------

16-bit	No reduction (bypass)	Clean, modern digital
12-bit	Subtle graininess (vintage ADCs)	1980s samplers, warm digital
8-bit	Noticeable noise floor	Classic video game consoles
4-bit	Heavy quantisation, robotic	Extreme lo-fi, speech synthesis
1-bit	Square wave conversion	1-bit music, ZX Spectrum beeper

2. Sample Rate Reduction:

- Range: 1-64x reduction
- Effect: Reduces time resolution, creating aliasing and "crunchy" artifacts
- NRPN Control: Per-channel parameter

Sample Rate Character:

Reduction	Effective SR (44.1kHz)	Character
1x	44.1kHz (bypass)	Clean, no aliasing
2x	22.05kHz	Subtle high-frequency roll-off
4x	11.025kHz	Telephone quality
8x	5.5kHz	AM radio, vintage speech
16x	2.75kHz	Extreme lo-fi, grunge
64x	~689Hz	Unrecognisable, rhythmic noise

Musical Applications:

1. Vintage Chip Sound:

- Bit Depth: 4-8 bits
- Sample Rate: 4-8x reduction
- Result: Authentic 1980s home computer/video game character
- Use: Retro leads, arpeggios, chip basslines

2. Lo-Fi Hip-Hop:

- Bit Depth: 10-12 bits
- Sample Rate: 2-4x reduction
- Result: Warm, nostalgic, slightly degraded
- Use: Lo-fi beats, vintage sampler emulation

3. Extreme Destruction:

- Bit Depth: 1-2 bits
- Sample Rate: 16-32x reduction
- Result: Unrecognisable, distorted, rhythmic
- Use: Industrial noise, glitch effects, extreme textures

Technical Note: The bit crusher uses **symmetric quantisation**, which maintains DC-free output (no DC offset) and predictable harmonic content.

Comb Filter

The **comb filter** creates a series of equally-spaced notches and peaks in the frequency spectrum, producing metallic, resonant, and "hollow" timbres reminiscent of physical body resonances.

Parameters:

1. Delay Time:

- Range: 0.1ms - 50ms
- Effect: Determines the spacing of spectral notches/peaks
- NRPN Control: Per-channel parameter

Delay Time vs Pitch:

The comb filter creates a **harmonic series** based on the delay time:

- Fundamental frequency: $f = 1 / \text{delay_time}$
- Example: 1ms delay = 1000Hz fundamental, 2000Hz, 3000Hz, 4000Hz harmonics

Typical Delay Settings:

Delay Time	Fundamental	Character
50ms	20Hz	Deep body resonance (bass)
10ms	100Hz	Low-mid body (kick drum)
5ms	200Hz	Mid-range resonance (snare)
2ms	500Hz	Upper-mid resonance (toms)
1ms	1000Hz	Presence resonance (vocals)
0.5ms	2000Hz	Bright metallic (cymbals)
0.1ms	10000Hz	Thin, sharp (high percussion)

2. Feedback:

- Range: -0.99 to +0.99 (-99% to +99%)
- Effect: Controls resonance intensity and spectral character
- NRPN Control: Per-channel parameter

Feedback Polarity:

- **Positive feedback (+):** Peaks at odd harmonics (fundamental, 3rd, 5th, 7th...)
 - Hollow, nasal, clarinet-like
 - String/acoustic body resonance
- **Negative feedback (-):** Peaks at even harmonics (2nd, 4th, 6th, 8th...)
 - Metallic, bright, inharmonic
 - Metallic percussion, bells

Feedback Amount:

Feedback	Resonance	Character

0%	None	Bypass (no comb filtering)
±30%	Subtle	Gentle body resonance
±60%	Moderate	Clear metallic character
±90%	Strong	Ringling, sustained resonance
±99%	Extreme	Near-infinite sustain, whistle

Musical Applications:

1. Acoustic Body Simulation:

- Delay: 5-20ms (corresponds to small-to-medium body size)
- Feedback: +40% to +70% (positive for natural resonance)
- Result: Simulates acoustic guitar, violin, or drum body resonance
- Use: Add body to thin synthetic sounds

2. Metallic Percussion:

- Delay: 0.5-2ms (high fundamental)
- Feedback: -70% to -90% (negative for inharmonic character)
- Result: Bell-like, gong-like, metallic timbres
- Use: Synthesised bells, gongs, metallic drums

3. Flanging Effect:

- Delay: 0.5-5ms (modulated by LFO)
- Feedback: ±50% to ±70%
- Result: Sweeping comb filter (jet-plane effect)
- Use: Combine with LFO modulation for dynamic flanging

Technical Note: The comb filter includes **anti-denormal clamping** to prevent CPU-intensive denormal numbers from degrading performance.

Delay Line

The **delay line** provides simple, clean echoes and rhythmic repeats with a fixed-size circular buffer.

Parameters:

1. Delay Time:

- Range: 0ms - 92.8ms @ 44.1kHz (0ms - 85.3ms @ 48kHz, 0ms - 42.6ms @ 96kHz)
- Resolution: 1 sample (22.7µs @ 44.1kHz, 20.8µs @ 48kHz)
- Buffer: 4096 samples (constant)
- NRPN Control: Per-channel parameter

2. Feedback:

- Range: 0% - 100%
- Effect: Controls number of echo repeats
- NRPN Control: Per-channel parameter

Feedback Character:

Feedback	Repeats	Character

0%	1 (single)	Simple echo, slapback
30%	3-4	Short rhythmic repeats
50%	5-7	Moderate echo tail
70%	10-15	Long sustaining echoes
90%	30-50	Near-infinite echo tail
100%	Infinite	Sustained loop (frozen audio)

Musical Applications:

1. Slapback Echo:

- Delay: 50-120ms (at 44.1kHz, use 40-90ms at source buffer limit)
- Feedback: 0-10%
- Result: Single distinct echo (rockabilly, vintage vocal)
- Use: Vintage guitar, vocals, lead instruments

2. Rhythmic Delay:

- Delay: 60-90ms (16th-note subdivisions at 120-180 BPM)
- Feedback: 40-60%
- Result: Multiple rhythmic repeats synced to tempo
- Use: Dub techno, reggae, electronic music

3. Ambience Layer:

- Delay: 20-40ms
- Feedback: 20-40%
- Result: Subtle thickening without distinct echoes
- Use: Add depth to sounds without obvious delay effect

Technical Note: The delay line uses a **circular buffer with masked indexing** (power-of-two size) for efficient wraparound without conditional branching.

Flanger

The **flanger** creates sweeping, jet-plane-like effects by mixing the dry signal with a delayed copy that has a time-varying delay controlled by an LFO.

Parameters:

1. LFO Rate:

- Range: 0.01Hz - 10Hz
- Musical Range: 0.1Hz - 5Hz
- Effect: Speed of the sweeping motion
- NRPN Control: Per-channel flanger rate (NRPN 150)

2. Depth:

- Range: 0% - 100%
- Effect: Intensity of the sweeping effect
- NRPN Control: Per-channel flanger depth (NRPN 151)

3. Feedback:

- Range: -99% to +99%
- Effect: Resonance at the notches (more dramatic sweep)
- NRPN Control: Per-channel flanger feedback (NRPN 152)

Flanger Character:

Rate	Depth	Feedback	Character
0.5Hz	30%	30%	Gentle, slow sweep (chorus-like)
1Hz	50%	50%	Classic flanger (jet-plane)
3Hz	70%	70%	Fast, aggressive sweep
8Hz	90%	80%	Extreme, metallic vibrato

Musical Applications:

1. Classic Jet-Plane Sweep:

- Rate: 0.5Hz - 1Hz
- Depth: 60-80%
- Feedback: 50-70%
- Result: Iconic flanging effect used in 1970s rock
- Use: Electric guitar, synthesiser leads, drums

2. Subtle Chorus-Like Thickening:

- Rate: 0.2Hz - 0.5Hz
- Depth: 20-40%
- Feedback: 10-30%
- Result: Wide, shimmering texture without obvious sweep
- Use: Pads, strings, ambient textures

3. Aggressive Modulation:

- Rate: 3Hz - 8Hz
- Depth: 80-100%
- Feedback: 70-90%
- Result: Fast, dramatic sweeps with metallic resonance
- Use: Psychedelic effects, electronic leads, sound design

Technical Note: Flanging is closely related to chorusing. The primary difference is that flanging uses **shorter delay times** (0.1-10ms) compared to chorus (10-50ms), resulting in comb-filtering effects that create the characteristic "swoosh" sound.

Phaser

The **phaser** creates sweeping notches in the frequency spectrum using a series of allpass filters, producing vocal-like, "wah-wah" effects.

Parameters:

1. LFO Rate:

- Range: 0.01Hz - 10Hz
- Musical Range: 0.1Hz - 5Hz
- Effect: Speed of the sweeping motion

- NRPN Control: Per-channel phaser rate

2. Depth:

- Range: 0% - 100%
- Effect: Intensity of the sweeping effect
- NRPN Control: Per-channel phaser depth

3. Feedback:

- Range: -99% to +99%
- Effect: Resonance at the notches
- NRPN Control: Per-channel phaser feedback

4. Stages:

- Range: 1-8 allpass stages
- Effect: Number of spectral notches (more stages = more notches)
- NRPN Control: NRPN 158 (phaser stages)

Phaser Stages Character:

Stages	Notches	Character
2	1	Subtle, single notch (vocal)
4	2	Classic phaser (guitar pedal)
6	3	Rich, complex sweep
8	4	Deep, dramatic modulation

Phaser vs Flanger:

Characteristic	Phaser	Flanger
Principle	Allpass filters	Time-varying delay
Notch Spacing	Unequal (logarithmic)	Equal (harmonic series)
Sound	Vocal, smooth	Metallic, jet-plane
Notch Count	Stages/2	Many (harmonic)

Musical Applications:

1. Vintage Guitar Phaser:

- Rate: 0.5Hz - 1.5Hz
- Depth: 50-70%
- Feedback: 40-60%
- Stages: 4-6
- Result: Classic 1970s guitar phaser sound
- Use: Electric guitar, keyboards, vocals

2. Subtle Vocal Movement:

- Rate: 0.1Hz - 0.3Hz

- Depth: 20-40%
- Feedback: 20-40%
- Stages: 2-4
- Result: Gentle, vocal-like character
- Use: Pads, strings, atmospheric sounds

3. Extreme Sweep:

- Rate: 2Hz - 5Hz
- Depth: 80-100%
- Feedback: 70-90%
- Stages: 6-8
- Result: Dramatic, deep sweeps
- Use: Sound design, psychedelic effects

Overdrive/Distortion

The **overdrive/distortion** effect adds harmonic saturation, warmth, and aggression by applying non-linear waveshaping to the signal.

Parameters:

1. Drive Amount:

- Range: 0% - 100%
- Effect: Intensity of distortion (soft clipping)
- NRPN Control: Per-channel overdrive amount

Drive Character:

Drive	Harmonic Content	Character
0%	None (bypass)	Clean, unprocessed
20%	Subtle 2nd/3rd	Tape saturation, warmth
40%	Moderate harmonics	Tube overdrive, vintage gear
60%	Strong harmonics	Guitar overdrive, grit
80%	Heavy clipping	Distortion, aggression
100%	Near square-wave	Extreme fuzz, heavy metal

Musical Applications:

1. Subtle Warmth:

- Drive: 10-30%
- Result: Gentle harmonic enhancement (2nd/3rd harmonics)
- Use: Add "analogue" character to clean digital sounds

2. Vintage Overdrive:

- Drive: 40-60%
- Result: Tube-like saturation (classic rock guitar)
- Use: Leads, bass, electric piano

3. Heavy Distortion:

- Drive: 70-100%
- Result: Aggressive, thick distortion
- Use: Heavy metal guitar, industrial bass, noise textures

Technical Note: The distortion algorithm uses **soft clipping** (tanh-like curve) rather than hard clipping, producing more musical harmonic content and avoiding harsh aliasing artifacts.

Master Effects

Master effects are applied to the **final stereo mix** after all 320 channels have been combined. Unlike per-channel effects, master effects have global settings that affect the entire instrument.

Master effects are controlled via both **MIDI CC messages** (for send levels) and **NRPN parameters** (for detailed effect settings).

Reverb

The **reverb** system creates the illusion of acoustic space, simulating room reflections and ambience. The Intuition Subsynth uses a **Schroeder reverb** algorithm with comb filters and allpass filters.

Architecture:

- Pre-Delay: 8ms (early reflections separation)
- Comb Filters: 4 parallel (38.2ms, 36.3ms, 46.5ms, 51.0ms @ 44.1kHz)
- Allpass Filters: 2 serial (8.8ms, 7.0ms @ 44.1kHz)

Parameters:

1. Reverb Send (per MIDI channel):

- Range: 0-127 (MIDI CC#91)
- Effect: How much signal from each MIDI channel goes to reverb
- Control: CC#91 (Effects 1 Depth, Reverb Send Level)

2. Reverb Decay:

- Range: 0-16383 (14-bit NRPN)
- Effect: Length of reverb tail (room size)
- NRPN Control: NRPN 6 (Reverb Decay)

3. Reverb Diffusion:

- Range: 0-16383 (14-bit NRPN)
- Effect: Density of reflections (smooth vs grainy)
- NRPN Control: NRPN 8 (Reverb Diffusion)

4. Reverb Wet/Dry Mix:

- Range: 0-100%
- Effect: Balance between reverb and dry signal
- NRPN Control: NRPN 7 (Reverb Mix)

Reverb Character:

Decay	Diffusion	Character	Room Type
Low	Low	Small, grainy reflections	Small room, booth
Low	High	Small, smooth reflections	Padded room, studio

Medium	Low	Medium space, individual echoes	Concert hall
Medium	High	Medium space, dense reflections	Church, cathedral
High	Low	Large space, sparse echoes	Canyon, warehouse
High	High	Large space, thick reverb	Large hall, cavern

Musical Applications:

1. Subtle Ambience:

- Send: 20-40 (MIDI CC#91)
- Decay: Low (3000-5000)
- Diffusion: Medium (8000)
- Mix: 20-30%
- Result: Gentle space, non-obvious reverb
- Use: Most instruments (drums, bass, piano)

2. Room Realism:

- Send: 40-70
- Decay: Medium (8000-11000)
- Diffusion: High (12000-15000)
- Mix: 40-60%
- Result: Realistic room or hall
- Use: Acoustic instruments, vocals, orchestral sounds

3. Lush Ambience (Pad/Drone):

- Send: 80-127
- Decay: High (14000-16383)
- Diffusion: Very High (15000-16383)
- Mix: 60-80%
- Result: Long, thick reverb tail
- Use: Pads, ambient textures, sound design

Performance Tip: Each MIDI channel can have its own reverb send level (CC#91), but the reverb decay/diffusion/mix are global settings. This allows you to place different instruments at different "depths" in the same virtual space.

Chorus

The **chorus** effect creates the illusion of multiple instruments playing simultaneously by modulating delayed copies of the signal with stereo LFO modulation.

Parameters:

1. Chorus Send (per MIDI channel):

- Range: 0-127 (MIDI CC#93)
- Effect: How much signal from each MIDI channel goes to chorus
- Control: CC#93 (Effects 3 Depth, Chorus Send Level)

2. Chorus Rate:

- Range: 0-16383 (14-bit NRPN)
- Effect: Speed of modulation
- NRPN Control: NRPN parameter (chorus rate)

3. Chorus Depth:

- Range: 0-16383 (14-bit NRPN)
- Effect: Intensity of pitch modulation
- NRPN Control: NRPN parameter (chorus depth)

4. Chorus Wet/Dry Mix:

- Range: 0-100%
- Effect: Balance between chorus and dry signal
- NRPN Control: NRPN parameter (chorus mix)

Chorus Character:

Rate	Depth	Character
Slow	Subtle	Wide, shimmering texture
Slow	Deep	Detuned, thick sound
Fast	Subtle	Vibrato-like, gentle movement
Fast	Deep	Dramatic warble, vintage tape

Musical Applications:

1. Thickening (Strings/Pads):

- Send: 60-100
- Rate: Slow (2000-5000)
- Depth: Moderate (6000-9000)
- Mix: 40-60%
- Result: Wide, ensemble-like texture
- Use: String sections, pads, synthesiser leads

2. Subtle Shimmer:

- Send: 30-50
- Rate: Medium (5000-8000)
- Depth: Subtle (3000-5000)
- Mix: 20-40%
- Result: Gentle movement without obvious modulation
- Use: Electric piano, clean guitar, vocals

3. Vintage Tape:

- Send: 70-127
- Rate: Fast (10000-13000)
- Depth: Deep (11000-14000)
- Mix: 50-70%
- Result: Warbling, tape-like character
- Use: Lo-fi effects, vintage emulation, sound design

Technical Note: The Intuition Subsynth's chorus uses **stereo LFO modulation with phase offset** between left and right channels, creating a wide stereo image.

Ping-Pong Delay

The **ping-pong delay** bounces echoes between the left and right stereo channels, creating rhythmic spatial movement.

Parameters:

1. Delay Time:

- Range: 0-16383 (14-bit NRPN)
- Effect: Time between echoes
- NRPN Control: NRPN 140 (Ping-Pong Delay Time)

2. Feedback:

- Range: 0-16383 (0-100%)
- Effect: Number of echo repeats
- NRPN Control: NRPN 141 (Ping-Pong Delay Feedback)

3. Stereo Separation:

- Range: 0-16383 (0-100%)
- Effect: How much echoes alternate left/right
- NRPN Control: NRPN 142 (Ping-Pong Delay Stereo)

4. Wet/Dry Mix:

- Range: 0-16383 (0-100%)
- Effect: Balance between delay and dry signal
- NRPN Control: NRPN 143 (Ping-Pong Delay Mix)

5. Modulation:

- Range: 0-16383
- Effect: Time-varying delay for chorused echoes
- NRPN Control: NRPN 144 (Ping-Pong Delay Modulation)

Ping-Pong Delay Character:

Time	Feedback	Stereo	Character
Short	Low	Full	Rhythmic stereo bounces
Medium	Medium	Full	Classic ping-pong echo
Long	High	Full	Sustained spatial echoes
Any	Any	Low	Mono-like delay (no bounce)

Musical Applications:

1. Classic Ping-Pong:

- Time: 200-500ms (quarter-note at 120-180 BPM)
- Feedback: 40-60% (3-6 repeats)
- Stereo: 100% (full left/right alternation)
- Mix: 30-50%
- Result: Classic stereo bouncing echo
- Use: Leads, vocals, percussion

2. Ambient Wash:

- Time: 500-1000ms
- Feedback: 70-90% (long tail)
- Stereo: 60-80%
- Mix: 40-60%
- Modulation: 20-40%
- Result: Spatial, evolving echo texture
- Use: Pads, ambient textures, sound design

3. Rhythmic Emphasis:

- Time: 150-300ms (sync'd to tempo)
- Feedback: 20-40% (short tail)
- Stereo: 100%
- Mix: 20-30%
- Result: Tight rhythmic stereo repeats
- Use: Drums, percussive synth, rhythmic parts

Performance Tip: The ping-pong delay is a **master effect**, so it affects all MIDI channels equally. Use it to create unified spatial movement across your entire mix.

Master Compressor

The **master compressor** provides dynamic range control, adds "punch" and "glue" to the mix, and prevents clipping when many channels play simultaneously.

Algorithm: RMS detection with adaptive attack/release

Parameters:

1. Threshold:

- Range: -60dB to 0dB (14-bit NRPN)
- Effect: Level at which compression begins
- NRPN Control: NRPN 130 (Compressor Threshold)

2. Ratio:

- Range: 1:1 to 20:1
- Effect: Amount of gain reduction above threshold
- NRPN Control: NRPN 131 (Compressor Ratio)

3. Attack Time:

- Range: 0.1ms - 100ms
- Effect: How quickly compression engages
- NRPN Control: NRPN 132 (Compressor Attack)

4. Release Time:

- Range: 10ms - 1000ms
- Effect: How quickly compression disengages
- NRPN Control: NRPN 133 (Compressor Release)

5. Makeup Gain:

- Range: 0dB to +24dB
- Effect: Compensates for gain reduction

- NRPN Control: NRPN 134 (Compressor Makeup Gain)

6. Wet/Dry Mix (Parallel Compression):

- Range: 0-100%
- Effect: Balance between compressed and uncompressed signal
- NRPN Control: NRPN 135 (Compressor Mix)

Compressor Character:

Threshold	Ratio	Attack	Release	Character
-6dB	2:1	10ms	100ms	Gentle limiting (safety)
-12dB	4:1	5ms	200ms	Moderate control (glue)
-20dB	8:1	1ms	50ms	Aggressive punch (drums)
-30dB	10:1	0.1ms	500ms	Heavy limiting (broadcast)

Musical Applications:

1. Mix Glue (Gentle Compression):

- Threshold: -12dB to -18dB
- Ratio: 2:1 to 3:1
- Attack: 10-30ms
- Release: 100-300ms
- Makeup: +2dB to +6dB
- Mix: 100%
- Result: Cohesive, "glued together" mix
- Use: Final mix processing, multi-channel performances

2. Punch (Aggressive Compression):

- Threshold: -18dB to -24dB
- Ratio: 4:1 to 6:1
- Attack: 1-5ms (fast)
- Release: 50-100ms (fast)
- Makeup: +6dB to +12dB
- Mix: 100%
- Result: Prominent transients, aggressive dynamics
- Use: Drum-heavy mixes, electronic music

3. Parallel Compression (New York Style):

- Threshold: -24dB to -36dB
- Ratio: 8:1 to 12:1
- Attack: 0.1-1ms (very fast)
- Release: 30-80ms (fast)
- Makeup: +12dB to +18dB
- Mix: 30-50% (parallel blend)
- Result: Maintains natural dynamics whilst adding density
- Use: Drums, bass-heavy mixes, retaining transients

Technical Note: The compressor uses **RMS detection** (20ms window) rather than peak detection, providing smooth, musical gain reduction. The adaptive attack/release automatically adjusts timing based on signal content.

Master Limiter

The **master limiter** is an **automatic safety limiter** that prevents clipping and distortion when the mix exceeds ± 0.95 (-0.4dBFS).

Operation:

- Type: Soft limiter (branchless ± 0.95 clamp)
- Threshold: Fixed at ± 0.95 (-0.4dBFS)
- Response: Instantaneous (0ms attack)
- Character: Transparent (minimal harmonic distortion)

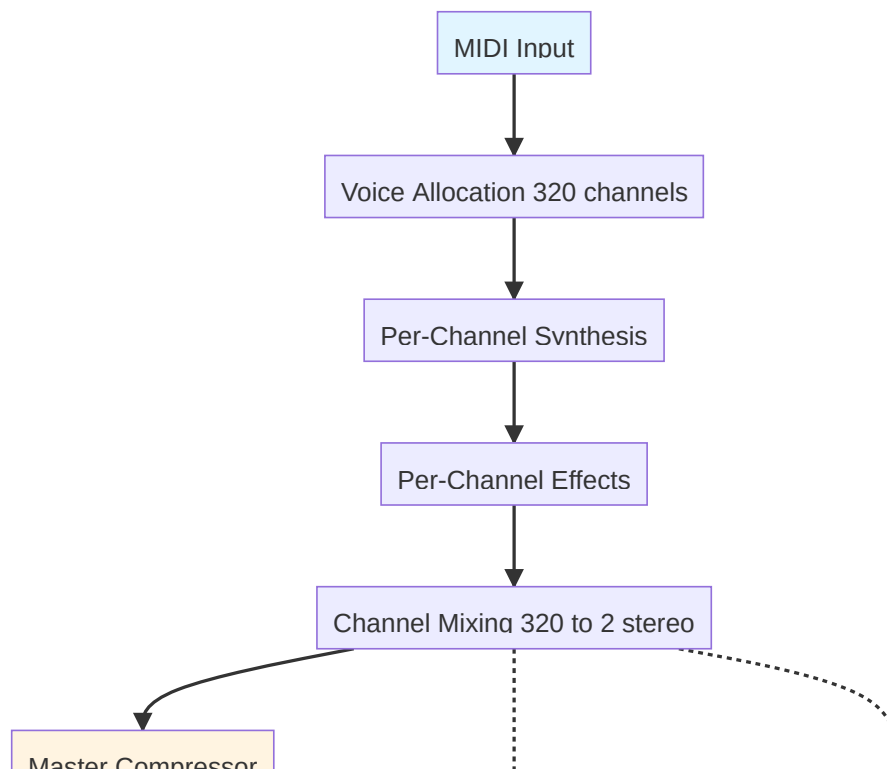
This effect is always active and has no user-adjustable parameters. It provides final safety limiting to protect your audio hardware and prevent digital clipping.

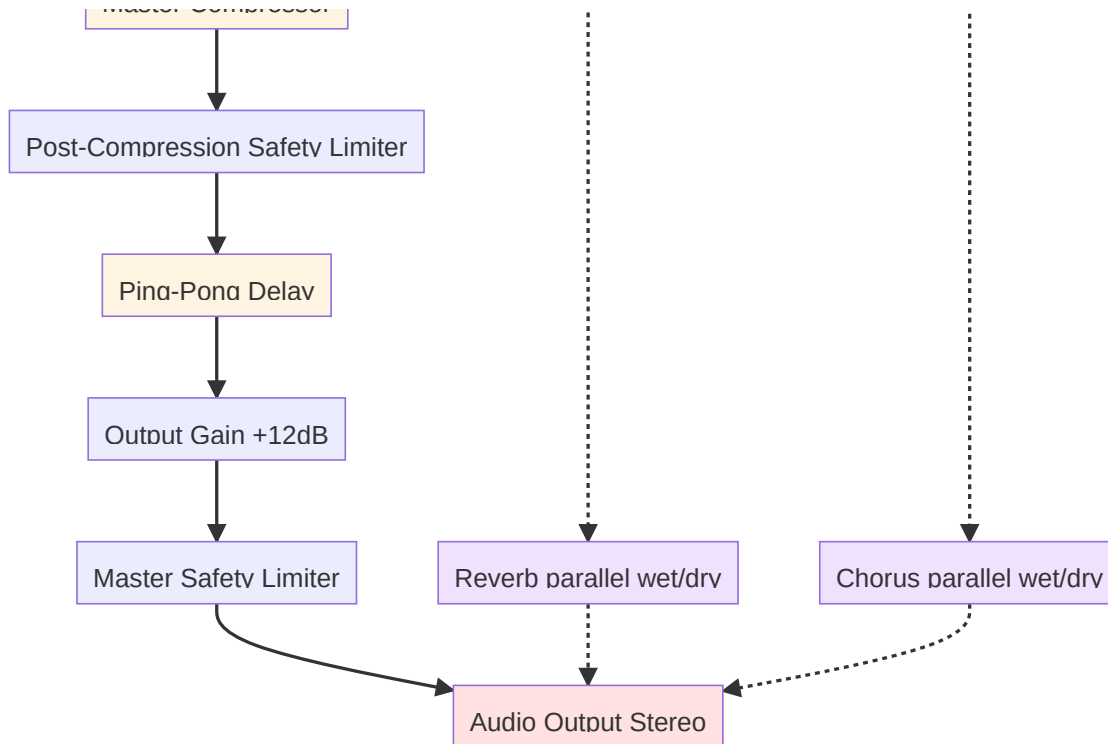
Technical Note: The limiter operates **after** the output gain (+12dB restoration) and ping-pong delay, ensuring the final output never exceeds ± 0.95 , even with extreme compressor/delay settings.

Effect Routing and Signal Flow

Understanding the signal flow through the effects system is essential for creating complex sounds and troubleshooting issues.

Complete Signal Flow:





Key Points:

1. **Per-channel effects** process each synthesis channel **independently** before mixing
2. **Reverb and chorus** are applied via **send levels** (CC#91, CC#93) with parallel wet/dry mixing
3. **Compressor** processes the mixed stereo signal at -12dB internal level
4. **Ping-pong delay** processes at safe -12dB level (after compression)
5. **Output gain** restores +12dB headroom for final output
6. **Master limiter** provides final safety at ± 0.95 to prevent clipping

Effect Interaction:

- **Compressor + Reverb**: Compression before reverb creates tighter, more controlled space
- **Delay + Chorus**: Chorus applied to delayed signals creates lush, modulated echoes
- **Bit Crusher + Filter**: Filtering after bit-crushing smooths quantisation noise
- **Overdrive + Flanger**: Flanging after overdrive emphasises harmonic movement

Performance Tip: The signal flow is **fixed** and cannot be reordered. If you need different effect routing, use per-channel effects creatively (e.g. apply flanger before the channel mixer to create individual flanged voices).

Effect Preset Examples

The following effect presets demonstrate professional sound design techniques using the Intuition Subsynth's effect system.

Preset 1: Vintage Chip Lead

Goal: Authentic 1980s home computer lead sound

Per-Channel Settings:

- Waveform: Square (PWM enabled)
- PWM Rate: 5Hz (via LFO)
- PWM Depth: 60%
- Bit Crusher: 8-bit, 4x sample rate reduction
- Overdrive: 30% (vintage warmth)

Master Settings:

- Reverb Send: 30 (subtle space)
- Chorus Send: 40 (thickening)
- Compressor Threshold: -18dB
- Compressor Ratio: 3:1
- Ping-Pong Delay: 250ms, 40% feedback, 100% stereo

Result: Authentic retro lead with PWM chorus, bit-crushed character, and rhythmic stereo echoes.

Preset 2: Lush Ambient Pad

Goal: Wide, evolving pad with deep spatial ambience

Per-Channel Settings:

- Waveform: Sawtooth (layered with Sine)
- Filter: Low-Pass, 1.5kHz cutoff, 30% resonance
- Filter LFO: 0.2Hz sine wave, 30% depth
- Comb Filter: 10ms delay, +40% feedback (body resonance)
- Flanger: 0.5Hz, 40% depth, 30% feedback

Master Settings:

- Reverb Send: 100 (maximum)
- Reverb Decay: High (14000)
- Reverb Diffusion: Very High (15500)
- Chorus Send: 80
- Chorus Rate: Slow (3000)
- Chorus Depth: Deep (10000)
- Compressor Threshold: -24dB
- Compressor Ratio: 2:1
- Ping-Pong Delay: 800ms, 70% feedback, 80% stereo, 30% modulation

Result: Thick, evolving pad with massive stereo width, gentle movement, and long ambient tail.

Preset 3: Aggressive Industrial Bass

Goal: Heavy, distorted bass with rhythmic filtering

Per-Channel Settings:

- Waveform: Sawtooth
- Filter: Low-Pass, 400Hz cutoff, 60% resonance
- Filter LFO: 4Hz (tempo-synced to 1/4 note), 50% depth
- Overdrive: 80% (heavy distortion)
- Bit Crusher: 6-bit (aggressive quantisation)
- Comb Filter: 2ms delay, -70% feedback (metallic resonance)

Master Settings:

- Reverb Send: 10 (minimal)
- Chorus Send: 0 (bypass)
- Compressor Threshold: -12dB
- Compressor Ratio: 8:1 (heavy compression)
- Compressor Attack: 1ms (fast)
- Compressor Release: 50ms (fast)
- Compressor Makeup: +12dB
- Ping-Pong Delay: 150ms (tempo-synced to 1/16 note), 30% feedback

Result: Aggressive, distorted bass with rhythmic filter sweeps, heavy compression for punch, and tight rhythmic echoes.

Preset 4: Classic 1970s Flanger Lead

Goal: Vintage jet-plane flanged lead

Per-Channel Settings:

- Waveform: Sawtooth
- Flanger: 0.8Hz, 70% depth, 60% feedback
- Overdrive: 40% (vintage overdrive)

Master Settings:

- Reverb Send: 50 (moderate space)
- Reverb Decay: Medium (8000)
- Chorus Send: 20 (subtle thickening)
- Compressor Threshold: -15dB
- Compressor Ratio: 4:1
- Ping-Pong Delay: 300ms, 45% feedback, 100% stereo

Result: Classic 1970s flanged lead with vintage character and spatial echoes.

Part IV: MIDI Implementation

13. MIDI Basics

MIDI (Musical Instrument Digital Interface) is the communication protocol that allows the Intuition Subsynth to receive performance data from keyboards, sequencers, computers, and other MIDI-capable devices. This section provides essential MIDI knowledge for musicians using the instrument.

What is MIDI?

MIDI is **not audio**. It is a stream of digital messages that describe musical events:

- **Note On:** "Start playing note 60 (Middle C) with velocity 100"
- **Note Off:** "Stop playing note 60"
- **Control Change:** "Set volume to 80"
- **Program Change:** "Switch to instrument #5 (Electric Piano)"
- **Pitch Bend:** "Bend the pitch up by 200 cents"

The Intuition Subsynth receives these messages and generates the actual audio in real-time according to the instructions.

Key Characteristics:

--	--	--

Characteristic	Value	Implication
Resolution	7-bit (0-127) or 14-bit (0-16383)	Most messages have 128 discrete values
Latency	<1ms (with modern interfaces)	Effectively instant
Bandwidth	31,250 bits/second	~960 note events per second
Channel Count	16 channels	16 independent instrument parts

MIDI Connections

The Intuition Subsynth provides two types of MIDI input:

1. MIDI IN (5-pin DIN connector):

- **Location:** Rear panel, 5-pin DIN socket
- **Standard:** MIDI 1.0 (31,250 baud)
- **Cable:** Use standard MIDI cable (5-pin DIN male-to-male)
- **Connection:** Connect MIDI OUT of controller/sequencer to MIDI IN of Subsynth
- **Latency:** Ultra-low (<0.5ms with JACK backend)

2. USB-MIDI:

- **Location:** USB-C port (rear panel)
- **Standard:** USB MIDI Class Compliant
- **Cable:** Use standard USB-C cable
- **Connection:** Connect directly to computer/tablet/smartphone
- **Latency:** Low (<2ms typical)
- **Advantage:** No separate MIDI interface required

Which Connection to Use?

Use Case	Recommended Connection	Reason
Live performance with keyboard	5-pin DIN / USB-MIDI	Lowest latency, no computer required
Studio DAW recording	USB-MIDI	Single cable, integrated with DAW
Standalone sequencer (hardware)	5-pin DIN / USB-MIDI	Standard hardware interconnection
Mobile device (tablet/smartphone)	USB-MIDI	Direct USB connection

Important: Do not connect both 5-pin DIN and USB-MIDI simultaneously from the same source, as this will cause duplicate MIDI messages.

MIDI Channels

MIDI supports **16 independent channels** (1-16), allowing you to play 16 different instruments simultaneously over a single MIDI cable.

Channel Organisation:

- **Channels 1-9, 12-16:** Melodic instruments (General MIDI programmes 1-128)
- **Channel 10:** Drum kit (always percussion, per GM Level 1 & 2 specification)
- **Channel 11:** Dual-purpose (melodic or percussion via Bank 0x78, per GM Level 2 specification)

How Channels Work:

Each MIDI message is tagged with a channel number (1-16). The Intuition Subsynth listens to **all 16 channels simultaneously** and plays the appropriate instrument for each channel according to the selected patchset.

Example Multi-Channel Setup:

MIDI Channel	Instrument (Programme)	Musical Role
1	Acoustic Grand Piano	Chords/melody
2	Acoustic Bass	Bassline
3	String Ensemble	Sustained pad
10	Standard Drum Kit	Drums/percussion
11	Synth Lead	Lead melody

Dual Percussion Channels (GM Level 2):

Whilst Channel 10 is always dedicated to percussion (per GM Level 1 standard), the Intuition Subsynth's GM Level 2 support allows **any channel** to be dynamically converted to percussion mode using **Bank 120 (0x78)**.

Converting Channels to Percussion:

1. Send CC#0 (Bank Select MSB) = 120 on the desired channel
2. Send any Programme Change message (value ignored)
3. Channel now plays drum sounds from the current patchset's drum kit

Example: Dual Percussion Setup (GM Level 2)

MIDI Channel	Setup	Musical Role
1	Acoustic Grand Piano	Chords/melody
2	Acoustic Bass	Bassline
10	Standard Drum Kit	Main drums (kick/snare)
11	Bank 120 → Drums	Hi-hats/cymbals (Bank 120)
12	String Ensemble	Sustained pad

Use Cases for Dual Percussion:

- **Independent drum layers:** Separate kick/snare on Channel 10, hi-hats/cymbals on Channel 11 for independent volume/effects control
- **Complex polyrhythms:** Two simultaneous drum patterns without channel conflicts
- **Live performance variations:** Main pattern on Channel 10, fills/breaks on Channel 11
- **Multi-track DAW recording:** Route each percussion element to separate MIDI channels for mixing flexibility

Reverting to Melodic Mode:

To convert a percussion channel back to melodic instruments:

1. Send CC#0 with any bank value **except** 120 (e.g., Bank 0 for default)
2. Send Programme Change for desired melodic instrument
3. Channel resumes normal melodic operation

See **Sound Bank Selection → GM Level 2: Dynamic Rhythm Conversion (Bank 120)** for complete documentation.

Important: MIDI channels are **independent of synthesis channels**. The Intuition Subsynth has 16 MIDI channels and 320 synthesis channels (64 per waveform type). Multiple MIDI channels can share synthesis channels when using the same waveform type.

MIDI Message Types

The Intuition Subsynth recognises the following MIDI message types:

Note On (0x90-0x9F)

Starts a note playing with specified pitch and velocity.

Parameters:

- **Note Number:** 0-127 (MIDI note, where 60 = Middle C, 440Hz = A4 = note 69)
- **Velocity:** 1-127 (0 = note off by convention)

Pitch Range: The Intuition Subsynth responds to the full MIDI range (0-127), but the audible range is approximately:

- **Lowest:** MIDI note 12 (C0, 16.35Hz)
- **Highest:** MIDI note 120 (C10, 16,744Hz, above human hearing)

Velocity Response:

- **0:** Note Off (equivalent to Note Off message)
- **1-127:** Note loudness (exponential mapping to amplitude)
- **Velocity curve:** Exponential (natural musical response)
- **Patchset-specific:** Some instruments have fixed velocity (organs, harpsichords)

Example: MIDI Note On message for Middle C (note 60) with velocity 100:

Status: 0x90 (Note On, channel 1)
Data 1: 60 (Middle C)
Data 2: 100 (velocity)

Note Off (0x80-0x8F)

Stops a note that was previously started with Note On.

Parameters:

- **Note Number:** 0-127 (must match the Note On)
- **Release Velocity:** 0-127 (typically ignored, but logged by Subsynth)

Important: If the **sustain pedal** (CC#64) is held, notes will not stop until the pedal is released. This is standard piano-like behaviour.

Polyphonic Key Pressure (0xA0-0xAF)

Also known as **polyphonic aftertouch**, this message sends per-note pressure information after the note has been struck.

Parameters:

- **Note Number:** 0-127
- **Pressure:** 0-127 (normalised to 0.0-1.0, 0.5 = neutral)

The Intuition Subsynth provides professional polyphonic aftertouch support with three modulation targets:

- **Filter Cutoff** (±20% range): Press harder for brighter tone, lighter for darker

- **Volume** ($\pm 30\%$ range): Expressive dynamics control
- **Vibrato Depth** ($\pm 50\%$ range): Pressure controls pitch modulation intensity

Configuration: Modulation depths are configurable per-channel via NRPN 216-218 (default: 50% sensitivity for all targets). See the NRPN section for details.

Control Change (0xB0-0xBF)

Changes a controller parameter in real-time (volume, pan, filter, effects, etc.). The Intuition Subsynth supports **120+ standard MIDI controllers**.

Parameters:

- **Controller Number:** 0-127 (identifies which parameter to change)
- **Controller Value:** 0-127 (the new value for that parameter)

Common Controllers:

- **CC#1:** Modulation Wheel (vibrato depth)
- **CC#7:** Channel Volume
- **CC#10:** Pan (stereo position)
- **CC#64:** Sustain Pedal
- **CC#74:** Filter Cutoff
- **CC#91:** Reverb Send
- **CC#93:** Chorus Send

See Section 14 for complete Standard MIDI Controller details.

Program Change (0xC0-0xCF)

Switches to a different instrument (General MIDI programme).

Parameters:

- **Programme Number:** 0-127 (internal representation)
- **Displayed As:** 1-128 (traditional GM numbering)

Example Programmes:

- **0** (displayed as 1): Acoustic Grand Piano
- **32** (displayed as 33): Acoustic Bass
- **80** (displayed as 81): Square Lead (Synth Lead)
- **127** (displayed as 128): Gunshot (SFX)

Patchset-Specific: The actual sound produced depends on the loaded patchset (SIDney, AYYMe, Rawland, etc.). For example, programme #1 in SIDney is a C64-style piano, whilst programme #1 in PixelaTED is a Commodore TED 7360/8360 chip-style piano.

Channel Pressure (0xD0-0xDF)

Also known as **channel aftertouch** or **mono aftertouch**, this applies pressure uniformly to all notes on the channel.

Parameters:

- **Pressure:** 0-127 (normalised to 0.0-1.0, 0.5 = neutral)

Channel pressure provides monophonic aftertouch (single pressure value affects all notes on the MIDI channel). This has lower CPU overhead than polyphonic aftertouch and is supported by most MIDI keyboards.

Modulation Targets (same as polyphonic aftertouch):

- **Filter Cutoff** ($\pm 20\%$ range)
- **Volume** ($\pm 30\%$ range)
- **Vibrato Depth** ($\pm 50\%$ range)

Configuration: Modulation depths are configurable per-channel via NRPN 216-218. See the NRPN section for details.

Pitch Bend (0xE0-0xEF)

Bends the pitch of all notes on the channel up or down.

Parameters:

- **14-bit value:** 0-16383
 - **0:** Maximum downward bend
 - **8192:** Centre (no bend)
 - **16383:** Maximum upward bend

Pitch Bend Range:

- **Default:** ± 2 semitones (adjustable via RPN #0)
- **SIDney patchset default:** ± 12 semitones (authentic C64 pitch slides)
- **Range:** ± 1 to ± 24 semitones (configurable per patchset)

Resolution: The 14-bit pitch bend provides **16,384 discrete steps**, enabling smooth, continuous pitch changes without audible "stepping".

Musical Applications:

- **Guitar-style bends:** Bend upward by 1-2 semitones (rock, blues)
- **Pitch slides:** Sweep across octaves (electronic music, sound design)
- **Vibrato:** Small, rapid pitch modulation (expressive sustained notes)
- **Portamento:** Smooth glide between notes (combined with CC#5 portamento time)

System Real-Time Messages

The Intuition Subsynth responds to the following real-time messages:

Active Sensing (0xFE):

- Sent by some MIDI controllers every 300ms to indicate the connection is active
- If active sensing is received and then stops, the Subsynth assumes the connection is lost and enters "all notes off" mode
- Status: Recognised and handled

Reset (0xFF):

- Resets the Subsynth to power-on state
- Clears all playing notes, resets controllers, restores default settings
- Status: Recognised and handled

MIDI Clock (0xF8):

- Provides tempo synchronisation for LFOs and delays
- Sent 24 times per quarter note
- Status: Recognised, tempo extracted for LFO sync (future implementation)

Start/Stop/Continue (0xFA, 0xFC, 0xFB):

- Transport control messages (typically from sequencers)
- Status: Recognised and logged (no internal sequencer to control)

System Exclusive Messages (0xF0...0xF7)

System Exclusive (SysEx) messages provide manufacturer-specific and universal system-level control beyond standard MIDI messages. The Intuition Subsynth implements **GM Level 2 Universal System Exclusive** messages for global control and configuration.

SysEx Message Structure:

```
F0 <manufacturer ID> <data bytes...> F7
|                                     └─ End of Exclusive (E0X)
└─ System Exclusive (S0X)
```

Supported Universal SysEx Messages:

GM System On (GM Level 1)

Resets the Intuition Subsynth to GM Level 1 mode with default settings.

Format:

```
F0 7E 7F 09 01 F7
| | | | | └─ End of Exclusive
| | | | └─ Sub-ID #2: GM System On (0x01)
| | | └─ Sub-ID #1: General MIDI (0x09)
| | └─ Device ID: All devices (0x7F)
| └─ Universal Non-Real Time SysEx (0x7E)
└─ System Exclusive start
```

Effect: Resets all channels to default GM1 state, clears controllers, stops all notes.

GM2 System On (GM Level 2)

Enables GM Level 2 mode, activating advanced features (Bank 120 percussion conversion, dual percussion channels, extended controllers).

Format:

```
F0 7E 7F 09 03 F7
| | | | | └─ End of Exclusive
| | | | └─ Sub-ID #2: GM2 System On (0x03)
| | | └─ Sub-ID #1: General MIDI (0x09)
| | └─ Device ID: All devices (0x7F)
| └─ Universal Non-Real Time SysEx (0x7E)
└─ System Exclusive start
```

Effect: Resets to GM2 mode, enabling Bank 120 percussion conversion and extended parameter support.

Master Volume

Controls global output volume across all 16 MIDI channels.

Format:

```
F0 7F 7F 04 01 ll mm F7
| | | | | | └─ End of Exclusive
| | | | | └─ MSB (0-127, coarse volume)
| | | | └─ LSB (0-127, fine volume)
| | | └─ Device Number: Master Volume (0x01)
| | └─ Sub-ID #1: Device Control (0x04)
| └─ Device ID: All devices (0x7F)
└─ Universal Real Time SysEx (0x7F)
    System Exclusive start
```

Range: 14-bit (0-16383), normalised to 0.0-1.0

- **0:** Silent (0.0)
- **8192:** Unity gain (0.5)
- **16383:** Maximum (1.0)

Example: Set master volume to 75% (12287 = 0x2FFF):

```
F0 7F 7F 04 01 7F 5F F7
      |  └─ MSB = 95 (0x5F)
      └─  LSB = 127 (0x7F)
```

Master Fine Tuning

Adjusts global pitch in cents (1 cent = 1/100th of a semitone).

Format:

```
F0 7F 7F 04 03 ll mm F7
| | | | | | | └─ End of Exclusive
| | | | | | └─ MSB (0-127, coarse tuning)
| | | | | └─ LSB (0-127, fine tuning)
| | | | └─ Device Number: Master Fine Tuning (0x03)
| | | └─ Sub-ID #1: Device Control (0x04)
| | └─ Device ID: All devices (0x7F)
| └─ Universal Real Time SysEx (0x7F)
└─ System Exclusive start
```

Range: 14-bit (0-16383), centre = 8192

- **0:** -100 cents (-1 semitone)
- **8192:** 0 cents (A440 standard tuning)
- **16383:** +100 cents (+1 semitone)

Example: Tune up +10 cents (8192 + 819 = 9011 = 0x2333):

```
F0 7F 7F 04 03 33 46 F7
      |  └─ MSB = 70 (0x46)
      └─  LSB = 51 (0x33)
```

Master Coarse Tuning

Adjusts global pitch in semitones (integer transposition).

Format:

```
F0 7F 7F 04 04 00 mm F7
| | | | | | | └─ End of Exclusive
| | | | | | └─ MSB (0-127, semitone offset)
| | | | | └─ LSB (always 0x00)
| | | | └─ Device Number: Master Coarse Tuning (0x04)
| | | └─ Sub-ID #1: Device Control (0x04)
| | └─ Device ID: All devices (0x7F)
| └─ Universal Real Time SysEx (0x7F)
└─ System Exclusive start
```

Range: 0-127 semitones, centre = 64

- **40:** -24 semitones (two octaves down, GM2 minimum)
- **64:** 0 semitones (no transposition)
- **88:** +24 semitones (two octaves up, GM2 maximum)

Note: The Intuition Subsynth supports the full 0-127 range internally (± 64 semitones), but GM Level 2 specification limits useful range to ± 24 semitones.

Example: Transpose down one octave (-12 semitones, MSB = 52):

```
F0 7F 7F 04 04 00 34 F7
      └── MSB = 52 (64 - 12)
```

Using SysEx Messages:

DAW/Sequencer Setup:

1. Create new MIDI track
2. Insert SysEx event at song start (e.g., GM2 System On)
3. Insert Master Volume/Tuning events as needed

Hardware Controller:

- Consult controller manual for SysEx transmission (some controllers have programmable SysEx buttons)
- Enter hexadecimal bytes exactly as shown above

MIDI Monitor Verification:

- Use MIDI monitoring software to verify SysEx transmission
- Look for `F0 ... F7` message boundaries
- Verify byte sequence matches formats above

Common Mistakes:

- **Missing F7:** SysEx messages must end with 0xF7 (End of Exclusive)
- **Wrong device ID:** Use 0x7F for "all devices" (Subsynth ignores device-specific IDs)
- **Incorrect byte count:** Count bytes carefully, including F0 and F7

MIDI Data Format

MIDI messages consist of **status bytes** (MSB set, 0x80-0xFF) and **data bytes** (MSB clear, 0x00-0x7F).

Status Byte Structure:

```
Binary: 1nnn cccc
         |    └── Channel (0-15, displayed as 1-16)
         └── Message type (8=Note Off, 9=Note On, B=Control Change, etc.)
```

Example: Note On, Channel 1, Note 60, Velocity 100

```
Hex:  90 3C 64
Dec:  144 60 100
Bin:  10010000 00111100 01100100
      ||| └── Channel 0 (displayed as channel 1)
      | └── Note On (0x9)
      └── Status byte (MSB=1)
```

Running Status:

MIDI uses **running status optimisation**: If the status byte is the same as the previous message, it can be omitted. The Intuition Subsynth correctly handles both running status and explicit status bytes.

Example (with running status):

```
Full:   90 3C 64  90 40 64  90 43 64  (Note On C, E, G with velocity 100)
Running: 90 3C 64    40 64    43 64  (Status byte omitted for E and G)
```

General MIDI Level 2 Compliance

The Intuition Subsynth is **fully compliant** with the General MIDI Level 2 specification, ensuring compatibility with all GM and GM2-compatible MIDI files and software.

GM Level 2 Requirements (all met):

Requirement	Specification	Intuition Subsynth
Polyphony	32 voices minimum	320 voices (wave-type-based)
Melodic channels	16 simultaneous	16 simultaneous
Percussion channels	Channels 10 & 11 (GM2)	Channels 10 & 11 supported
Bank 0x78 rhythm	Dynamic percussion conversion	Full support (any channel)
Programmes	128 melodic	128 melodic (per patchset)
Drum sounds	47 drum/percussion	47 drum/percussion
Controllers	Volume, Pan, Expression, etc.	120+ controllers
Pitch Bend	±2 semitones default	±2 semitones default
Velocity response	Dynamic	Exponential velocity curve
Reverb/Chorus	Send levels (CC#91/93)	Full reverb/chorus system
SysEx support	GM/GM2 System On, Master Volume/Tuning	Full GM2 SysEx support

GM Compatibility Benefits:

- Play any GM MIDI file correctly (30+ years of MIDI content)
- Integrate with GM-compatible DAWs and software
- Predictable instrument assignments across patchsets
- Standard controller mappings for universal control

Patchset Variations:

Whilst the Intuition Subsynth is GM-compliant, each patchset (SIDney, AYYMe, Rawland, etc.) provides a **different sonic interpretation** of the 128 GM programmes:

- **SIDney**: MOS 8580 SID chip character (Commodore 64)
- **AYYMe**: AY-3-8910/YM2149 PSG chip character (Amstrad CPC, ZX Spectrum128K/+2/+3, MSX, Atari ST)
- **Rawland**: Roland-inspired synthesis (303, 808, 909, etc.)
- **PixelaTED**: Commodore TED 7360/8360 chip emulation
- **EenBeetje**: ZX Spectrum 1-bit beeper

All patchsets maintain the same GM programme numbers, so switching patchsets changes the sonic character whilst preserving musical arrangements.

MIDI Troubleshooting

Problem: No sound when pressing keys

Solutions:

1. **Check MIDI connection:** Ensure cable is connected to MIDI IN (not MIDI OUT/THRU)

2. **Check MIDI channel:** Ensure your controller is sending on channels 1-16
3. **Check volume:** Send CC#7 (Channel Volume) value 100 to channel 1
4. **Check programme:** Send Program Change 0 (Acoustic Grand Piano) to channel 1
5. **Check MIDI activity:** The Intuition Subsynth logs all MIDI messages in debug mode

Problem: Notes stick (sustain indefinitely)

Solutions:

1. **Sustain pedal stuck:** Send CC#64 value 0 to all channels
2. **Note Off missing:** Some controllers forget to send Note Off messages
3. **Send All Notes Off:** Send CC#123 value 0 to all channels
4. **Send All Sound Off:** Send CC#120 value 0 to all channels

Problem: Wrong instrument plays

Solutions:

1. **Check programme number:** Some software sends programme 0-127, others send 1-128
2. **Check bank select:** Verify Bank Select (CC#0) matches desired patchset (0-4)
3. **Check patchset:** Use CC#0 to switch banks or load via `--patchset=NAME`

Problem: Pitch is wrong

Solutions:

1. **Check pitch bend:** Centre pitch bend wheel (should send value 8192)
2. **Check master tuning:** Reset master tuning to default (NRPN 24 = 8192)
3. **Check pitch bend range:** Reset to ± 2 semitones (RPN #0 = 2)

14. Standard MIDI Controllers

MIDI Control Change (CC) messages provide real-time control over synthesis parameters during performance. The Intuition Subsynth supports **120+ standard MIDI controllers**, providing comprehensive control via any MIDI-capable device or software.

Control Change Message Structure

MIDI CC Format:

Status Byte: 0xB0-0xBF (Control Change, channels 1-16)
 Data Byte 1: 0-127 (Controller Number)
 Data Byte 2: 0-127 (Controller Value)

Example: Set Channel Volume (CC#7) to 100 on MIDI Channel 1:

```
Hex:  B0 07 64
Dec:  176 7 100
      |  |  └─ Value: 100
      |  └─── Controller: 7 (Channel Volume)
      └────── Status: 0xB0 (CC, channel 1)
```

Essential Controllers

These controllers are fundamental for musical performance and are supported by virtually all MIDI devices.

CC#1: Modulation Wheel

Function: Controls vibrato depth (pitch modulation via LFO).

Range: 0-127

- **0:** No vibrato (modulation off)
- **64:** Moderate vibrato
- **127:** Maximum vibrato

Typical Use:

- **0-30:** Subtle vibrato for strings, vocals
- **40-80:** Moderate vibrato for expressive leads
- **90-127:** Extreme vibrato for dramatic effects

Technical Implementation: Modulates LFO depth routed to pitch. The vibrato rate is determined by the patch's LFO rate parameter (typically 4-7Hz).

Performance Tip: Most MIDI keyboards have a dedicated modulation wheel (left side of keyboard) that sends CC#1 continuously as you move it.

CC#7: Channel Volume

Function: Controls the overall volume level for the entire MIDI channel.

Range: 0-127

- **0:** Silent (channel muted)
- **64:** Medium volume (approximately -6dB)
- **100:** Standard performance level (0dB)
- **127:** Maximum volume (+3dB)

Volume Curve: Exponential (natural musical response, matching human perception).

Typical Levels:

Value	dB Level	Musical Context
0	-∞	Mute (no sound)
32	-18dB	Very quiet (pp)
64	-6dB	Moderate (mp to mf)
100	0dB	Standard performance level (f)
127	+3dB	Maximum (ff)

Interaction with CC#11 (Expression): Channel Volume sets the **maximum** level, whilst Expression provides **dynamic** real-time control within that maximum. The final volume is: $\text{Volume} \times \text{Expression} \times \text{Velocity}$.

Performance Tip: Set Channel Volume once at the beginning of a song to balance instruments, then use Expression (CC#11) for dynamic phrasing during performance.

CC#10: Pan (Stereo Position)

Function: Controls the stereo position of the channel.

Range: 0-127

- **0:** Hard left
- **64:** Centre
- **127:** Hard right

Panning Law: Constant-power (-3dB centre), ensuring equal perceived loudness across the stereo field.

Stereo Field Distribution:

Value	Position	Pan Angle	Left Level	Right Level
0	Hard left	-100%	100%	0%
32	Left	-50%	85%	50%
64	Centre	0%	71%	71%
96	Right	+50%	50%	85%
127	Hard right	+100%	0%	100%

Musical Applications:

- **0-30:** Left instruments (rhythm guitar, left hand piano)
- **30-64:** Centre-left (vocals, lead instruments with slight offset)
- **64:** Centre (kick drum, bass, snare, lead vocals)
- **64-96:** Centre-right (harmony vocals, right hand piano)
- **96-127:** Right instruments (lead guitar, percussion)

Performance Tip: Avoid panning bass frequencies (kick, bass) hard left/right, as this creates phase issues on mono playback systems. Keep bass elements centred (value 64).

CC#11: Expression

Function: Provides real-time dynamic control (like a volume pedal).

Range: 0-127

- **0:** Silent
- **64:** Moderate expression
- **127:** Full expression

Expression vs Volume:

CC#7 (Volume)	CC#11 (Expression)
Static (set once)	Dynamic (changes constantly)
Balances instruments	Musical phrasing
Exponential curve	Linear/exponential hybrid
Rarely automated	Frequently automated

Combined Effect: Final Volume = CC#7 × CC#11 × Velocity

Example Usage:

- **CC#7 = 100:** String section volume set to performance level
- **CC#11 = 30 → 100:** String swell from quiet to loud (crescendo)
- **CC#11 = 100 → 20:** String fade from loud to quiet (diminuendo)

Performance Tip: Use an expression pedal connected to your MIDI controller and assign it to CC#11 for foot-controlled dynamics, leaving your hands free to play.

CC#2: Breath Control

Function: Controls expression (primarily volume/brightness) via breath pressure sensor.

Range: 0-127

- **0:** No breath pressure (minimum expression)
- **64:** Moderate breath pressure
- **127:** Maximum breath pressure

Technical Implementation: Multiplies current expression value by a factor of 0.5-1.0, providing smooth dynamic control.

Expression Scaling:

Value	Multiplier	Effect on Expression
0	0.5×	50% expression (quiet)
64	0.75×	75% expression
127	1.0×	Full expression

Musical Applications:

- **Wind controllers:** Natural breath-controlled dynamics for realistic woodwind/brass simulation
- **EWI (Electronic Wind Instrument):** Essential for expressive wind synthesiser playing
- **Automation:** Smooth crescendo/diminuendo without touching volume faders
- **Vocal emulation:** Breath-like phrasing on sustained pad sounds

Performance Tip: Breath controllers provide the most natural dynamic expression for wind-style playing. The Intuition Subsynth responds smoothly to breath pressure with no latency or stepping.

CC#4: Foot Control

Function: Secondary volume control optimised for foot pedal input.

Range: 0-127

- **0:** Minimum volume (30% of current level)
- **64:** Moderate volume (65% of current level)
- **127:** Maximum volume (100% of current level)

Volume Scaling:

Value	Multiplier	Effect on Volume
0	0.3×	30% volume (very quiet)
32	0.475×	47.5% volume
64	0.65×	65% volume
96	0.825×	82.5% volume
127	1.0×	Full volume

Difference from CC#7 (Channel Volume):

CC#7 (Channel Volume)	CC#4 (Foot Control)
Primary volume control	Secondary volume control
Full range (0-100%)	Limited range (30-100%)
Set once, rarely changes	Dynamic pedal control
Keyboard fader/DAW automation	Foot pedal hardware

Musical Applications:

- **Organ volume swells:** Classic Hammond organ-style expression pedal control
- **Dynamic mixing:** Real-time volume adjustment whilst playing
- **Hands-free control:** Adjust volume without interrupting performance
- **Crossfades:** Smooth transitions between layers or timbres

Performance Tip: Use CC#4 for dynamic volume changes during performance, reserving CC#7 for static mixing levels. The limited range (30-100%) prevents complete silence, maintaining presence whilst allowing expressive control.

CC#8: Balance

Function: Controls left/right balance of the stereo signal (different from Pan).

Range: 0-127

- **0:** Hard left (right channel muted)
- **64:** Centre (equal left/right balance)
- **127:** Hard right (left channel muted)

Balance vs Pan:

CC#10 (Pan)	CC#8 (Balance)
Positions mono signal in stereo field	Adjusts level of existing stereo signal
Affects perceived location	Affects left/right loudness
Creates stereo image	Modifies existing stereo image
Used for mixing placement	Used for stereo correction

Technical Difference:

- **Pan (CC#10):** Distributes a mono source across the stereo field using constant-power panning
- **Balance (CC#8):** Adjusts the relative levels of left and right channels independently (like a balance control on a hi-fi amplifier)

Musical Applications:

- **Stereo correction:** Fix unbalanced stereo recordings or patches
- **Creative effects:** Sweep stereo image from left to right
- **Dual-layer patches:** Balance layered sounds with different stereo positions
- **Automation:** Dynamic stereo width modulation

Performance Tip: For most applications, use CC#10 (Pan) for positioning instruments. Use CC#8 (Balance) only when you need to adjust the left/right balance of an already-stereo sound.

CC#64: Sustain Pedal (Damper)

Function: Holds notes in sustain phase after key release (piano-style sustain).

Range: 0-127 (threshold-based)

- **0-63:** Off (notes release normally)
- **64-127:** On (notes sustain until pedal released)

Behaviour:

1. **Pedal Down** (value ≥ 64): All notes currently playing and future notes remain sustained
2. **Pedal Up** (value < 64): All sustained notes enter release phase simultaneously

Musical Applications:

- **Piano parts:** Essential for legato playing and chord voicing
- **String pads:** Sustains chords whilst playing melody over the top
- **Lead synth:** Creates smooth, connected phrases

Important: Sustain pedal affects the **envelope release phase**, not the amplitude directly. Notes still follow their programmed release time but only begin the release when the pedal is lifted.

Performance Tip: Most MIDI keyboards have a dedicated sustain pedal input (1/4" jack). Ensure your pedal sends standard CC#64 messages.

CC#65: Portamento On/Off

Function: Enables or disables portamento (pitch glide between notes).

Range: 0-127 (threshold-based)

- **0-63:** Off (notes play at their exact pitch immediately)
- **64-127:** On (notes glide from previous pitch to new pitch)

Interaction with CC#5 (Portamento Time): CC#65 enables/disables portamento, whilst CC#5 controls how long the glide takes.

Example:

- **CC#65 = 0:** Portamento off (notes play instantly)
- **CC#65 = 127, CC#5 = 50:** Portamento on, 50ms glide time

Musical Applications:

- **Lead synth:** Smooth pitch glides for expressive solos
- **Bass synth:** Slides between bass notes (funk, electronic)
- **Sound effects:** Dramatic pitch sweeps

CC#66: Sostenuuto

Function: Holds only the notes that were **already playing** when the pedal was pressed.

Range: 0-127 (threshold-based)

- **0-63:** Off
- **64-127:** On

Difference from Sustain (CC#64):

Sustain (CC#64)	Sostenuto (CC#66)
Holds all notes (current + future)	Holds only notes playing when pressed
Piano-style damper pedal	Selective sustain
Most commonly used	Less commonly used

Musical Application: Hold a bass note (press pedal), then play staccato melody on top. The bass sustains whilst the melody remains short.

CC#67: Soft Pedal

Function: Reduces note velocity (softer attack), emulating the una corda (soft) pedal on acoustic pianos.

Range: 0-127 (threshold-based)

- **0-63:** Off (normal velocity)
- **64-127:** On (velocity reduced to 70% of original)

Effect: Multiplies incoming note velocities by 0.7, creating softer, more delicate articulation.

Musical Application: Gentle, intimate passages where full dynamic range would be too aggressive.

CC#68: Legato

Function: Enables portamento (pitch glide) only for legato playing (overlapping notes).

Range: 0-127 (threshold-based)

- **0-63:** Off (portamento disabled for legato)
- **64-127:** On (portamento enabled for overlapping notes only)

Difference from CC#65 (Portamento On/Off):

CC#65 (Portamento)	CC#68 (Legato)
Portamento on all notes	Portamento only on overlapping notes
Always glides from previous pitch	Glides only when keys overlap
Affects staccato and legato	Affects legato only
Typical synth lead playing	Realistic wind/string emulation

Behaviour:

1. **CC#68 = 0:** Portamento disabled, even with CC#65 on (discrete pitch changes)
2. **CC#68 ≥ 64, notes overlap:** Glide from previous note to new note
3. **CC#68 ≥ 64, no overlap:** No glide (discrete pitch change)

Musical Applications:

- **Realistic wind instruments:** Natural pitch transitions only when tonguing/slurring
- **String emulation:** Authentic legato slides, staccato articulation remains discrete
- **Expressive leads:** Smooth glides on sustained phrases, clean attack on rhythmic playing
- **Brass patches:** Trombone-style glissando on connected notes

Performance Tip: Enable both CC#65 (Portamento On) and CC#68 (Legato On) for natural wind/string emulation. Only overlapping notes will glide, whilst staccato notes play with discrete pitch.

CC#69: Hold 2 (Sostenuto Alternative)

Function: Secondary hold pedal providing an alternative to sostenuto (CC#66).

Range: 0-127 (threshold-based)

- **0-63:** Off (notes released normally)
- **64-127:** On (notes held until pedal released)

Behaviour:

1. **Pedal Down** (value ≥ 64): All currently playing notes continue sustaining
2. **Pedal Up** (value < 64): All held notes enter release phase (unless CC#64 sustain is active)

Difference from CC#64 (Sustain) and CC#66 (Sostenuto):

CC#64 (Sustain)	CC#66 (Sostenuto)	CC#69 (Hold 2)
Holds all notes	Holds only notes pressed before pedal	Holds all notes
Most common	Selective hold	Secondary hold
Piano-style damper	Acoustic piano middle pedal	Alternative hold
Universal support	Less common in MIDI controllers	Rare hardware support

Musical Applications:

- **Dual sustain:** Use both CC#64 and CC#69 for independent sustain control on different note layers
- **Extended sustain:** Additional hold functionality beyond standard sustain
- **Organ-style hold:** Continuous drone whilst playing staccato melody
- **Sound design:** Experimental sustain layering effects

Performance Tip: Most MIDI keyboards do not have a dedicated CC#69 pedal input. You may need to assign CC#69 via software (DAW automation or MIDI processor) for creative sustain effects.

Filter Controllers

These controllers provide real-time control over the synthesis filter, enabling expressive timbral changes during performance.

CC#71: Filter Resonance

Function: Controls the filter resonance (Q factor).

Range: 0-127

- **0:** No resonance (smooth roll-off)
- **64:** Moderate resonance (slight peak at cutoff)
- **100:** High resonance (pronounced peak, self-oscillation threshold)
- **127:** Maximum resonance (self-oscillation)

Resonance Character:

Value	Q Factor	Character
0	0.5	Gentle roll-off (no peak)
32	1.0	Subtle emphasis

64	2.0	Moderate resonance (vocal-like)
96	5.0	Strong resonance (synthesiser)
127	10.0+	Extreme resonance (whistle)

Self-Oscillation: Above approximately $Q=10$ (value ~ 100), the filter begins to self-oscillate, generating a pure sine wave at the cutoff frequency even with no input signal.

Musical Applications:

- **Low resonance (0-40):** Natural, acoustic-like filtering
- **Medium resonance (40-80):** Classic synthesiser character
- **High resonance (80-127):** Dramatic, whistling effects

Performance Tip: Combine with filter sweeps (CC#74 or NRPN cutoff modulation) to create classic "wah-wah" effects.

CC#74: Filter Cutoff

Function: Controls the filter cutoff frequency (brightness).

Range: 0-127

- **0:** Minimum cutoff ($\sim 20\text{Hz}$, very dark)
- **64:** Mid-range cutoff ($\sim 1\text{kHz}$, balanced)
- **127:** Maximum cutoff ($\sim 20\text{kHz}$, maximum brightness)

Frequency Mapping: Exponential (musical, matching human pitch perception).

Cutoff Frequency Table (approximate):

Value	Frequency	Character
0	20Hz	Completely dark (sub-bass only)
32	200Hz	Muffled, distant
64	1kHz	Balanced, natural
96	5kHz	Bright, presence
127	20kHz	Maximum brightness (full spectrum)

Musical Applications:

- **Envelope-controlled cutoff:** Opening filter during attack creates "pluck" or "snap"
- **LFO-controlled cutoff:** Automatic wah-wah effect (rhythmic filtering)
- **Manual cutoff sweeps:** Real-time expression via MIDI controller or automation

Performance Tip: Assign CC#74 to a slider, knob, or expression pedal for real-time filter sweeps during performance.

Sound Controllers (CC#70-79)

The GM specification defines Sound Controllers (CC#70-79) for real-time timbral control. The Intuition Subsynth implements these controllers with extended ranges optimised for vintage chip emulation and modern sound design.

CC#70: Sound Variation (Brightness)

Function: Controls overall sound character and brightness (maps to filter cutoff variation).

Range: 0-127 (normalised to 0.0-1.0)

- **0:** Dark, muffled character
- **64:** Balanced, neutral character
- **127:** Bright, aggressive character

Technical Implementation: Affects the `brightness` parameter, which modulates filter cutoff and harmonic content.

Musical Applications:

- **Timbral variation:** Subtle brightness changes within the same patch
- **Automation:** Evolving pad sounds with slowly changing character
- **Performance control:** Real-time timbral expression during solos

Performance Tip: Use CC#70 for subtle timbral variation, CC#74 (Filter Cutoff) for dramatic filter sweeps.

CC#72: Sound Release Time

Function: Controls the envelope release time (note fade-out duration).

Range: 0-127

- **Extended Range:** 0-15,000ms (0-15 seconds) - far exceeds standard GM range
- **Value 0:** ~0ms (instant release, staccato)
- **Value 64:** ~7.5 seconds (moderate release)
- **Value 127:** 15 seconds (extreme sustain, ambient pads)

Release Time Mapping:

Value	Time	Musical Context
0	0ms	Percussive, staccato
32	3.75s	Piano-style natural decay
64	7.5s	Sustained strings, pads
96	11.25s	Ambient textures, drones
127	15s	Infinite sustain-style pads

Musical Applications:

- **Staccato articulation:** Low values (0-20) for tight, percussive playing
- **Legato phrasing:** Moderate values (30-60) for smooth, connected notes
- **Ambient pads:** High values (80-127) for long, evolving drones
- **Dynamic expression:** Automate release for evolving sustain characteristics

Performance Tip: The extended 15-second range enables realistic pad/drone sounds without relying on the sustain pedal. Ideal for ambient, atmospheric music and vintage synthesiser emulation.

CC#73: Sound Attack Time

Function: Controls the envelope attack time (note onset duration).

Range: 0-127

- **Extended Range:** 0-11,700ms (0-11.7 seconds) - optimised for pads and drones
- **Value 0:** ~0ms (instant attack, percussive)

- **Value 64:** ~5.85 seconds (slow swell)
- **Value 127:** 11.7 seconds (extremely slow fade-in)

Attack Time Mapping:

Value	Time	Musical Context
0	0ms	Percussive, instant attack
16	1.46s	Piano, string pizzicato
32	2.93s	Bowed strings, brass crescendo
64	5.85s	Pad swell, ambient wash
96	8.78s	Slow cinematic build
127	11.7s	Extreme pad fade-in, sound design

Musical Applications:

- **Percussive sounds:** Instant attack (0-5) for drums, plucks, stabs
- **Natural instruments:** Short attack (5-20) for piano, guitar, brass
- **String swells:** Moderate attack (30-60) for bowed strings, sustained brass
- **Ambient pads:** Slow attack (70-127) for lush, evolving textures

Performance Tip: Combine long attack (CC#73) with long release (CC#72) for classic "pad swell" sounds. The extended range allows cinematic, evolving textures without external envelope automation.

CC#75: Sound Control 6 (Decay Time)

Function: Controls the envelope decay time (transition from peak to sustain level).

Range: 0-127

- **Range:** 0-2,000ms (0-2 seconds)
- **Value 0:** ~0ms (instant decay, percussive envelope)
- **Value 64:** ~1 second (moderate decay)
- **Value 127:** 2 seconds (slow decay, vintage synth character)

Decay Time Mapping:

Value	Time	Musical Context
0	0ms	Organ-style (no decay, instant sustain)
32	500ms	Piano, electric piano
64	1s	Vintage synthesiser bass, leads
96	1.5s	Ambient plucks, evolving pads
127	2s	Extreme decay, sound design

Musical Applications:

- **Percussive plucks:** Short decay (0-30) for tight, punchy sounds
- **Vintage synth character:** Moderate decay (40-80) for classic synthesiser envelopes
- **Evolving textures:** Long decay (90-127) for slowly changing timbres

Performance Tip: Decay time affects the transition from the initial attack peak to the sustain level. Combine with CC#73 (Attack) and CC#72 (Release) to sculpt complete envelope shapes during performance.

CC#76: Sound Control 7 (Filter Type)

Function: Selects the synthesis filter type (LP, BP, HP, Notch).

Range: 0-127 (maps to 4 filter types)

Filter Type Mapping:

Value Range	Filter Type	Character
0-31	Low-Pass	Removes highs, warm/muffled
32-63	Band-Pass	Removes lows and highs, nasal/vocal
64-95	High-Pass	Removes lows, thin/bright
96-127	Notch	Removes midrange, hollow/phased

Filter Characteristics:

1. Low-Pass (0-31):

- **Use:** Warm bass, mellow pads, removing harshness
- **Cutoff Behaviour:** Higher cutoff = brighter, lower cutoff = darker

2. Band-Pass (32-63):

- **Use:** Vocal-like tones, telephone effects, midrange emphasis
- **Cutoff Behaviour:** Cutoff sets centre frequency of pass band

3. High-Pass (64-95):

- **Use:** Removing bass rumble, thin leads, crisp hi-hats
- **Cutoff Behaviour:** Higher cutoff = thinner, lower cutoff = more bass

4. Notch (96-127):

- **Use:** Phaser-like effects, hollow tones, frequency removal
- **Cutoff Behaviour:** Cutoff sets centre frequency of rejected band

Musical Applications:

- **Creative automation:** Sweep CC#76 from 0→127 to transition through all filter types
- **Patch variation:** Change filter type for dramatic timbral shifts within one patch
- **Sound design:** Combine with filter resonance (CC#71) for complex tonal shaping

Performance Tip: Low-pass is the most common filter type (vintage synthesiser character). Experiment with band-pass and notch for unique, vocal-like timbres.

CC#77: Sound Control 8 (Vibrato Rate)

Function: Controls the LFO rate for vibrato (pitch modulation).

Range: 0-127

- **Mapped Range:** 0-10.0 Hz
- **Value 0:** 0 Hz (no vibrato)

- **Value 64:** ~5.0 Hz (moderate vibrato, typical vocal/string rate)
- **Value 127:** 10.0 Hz (fast vibrato, trill-like)

Vibrato Rate Character:

Value	Hz	Character
0	0 Hz	No vibrato (static pitch)
16	1.25 Hz	Very slow vibrato (ambient pads)
32	2.5 Hz	Slow vibrato (strings, woodwinds)
64	5.0 Hz	Moderate vibrato (vocals, strings)
96	7.5 Hz	Fast vibrato (expressive solos)
127	10.0 Hz	Very fast vibrato (trill, SFX)

Musical Applications:

- **String emulation:** 4-6 Hz (values 50-77) for realistic string vibrato
- **Vocal simulation:** 5-7 Hz (values 64-89) for natural vocal vibrato
- **Lead synthesiser:** 6-8 Hz (values 77-102) for expressive solo leads
- **Special effects:** 9-10 Hz (values 115-127) for trill-like modulation

Performance Tip: Combine CC#77 (Vibrato Rate) with CC#78 (Vibrato Depth) and CC#1 (Modulation Wheel) for complete vibrato control. Use moderate rates (4-7 Hz) for natural, musical vibrato.

CC#78: Sound Control 9 (Vibrato Depth)

Function: Controls the vibrato modulation depth (pitch deviation amount).

Range: 0-127

- **Mapped Range:** 0-0.1 (pitch modulation multiplier)
- **Value 0:** No vibrato depth (no pitch modulation)
- **Value 64:** Moderate vibrato depth (~0.05, typical musical vibrato)
- **Value 127:** Maximum vibrato depth (0.1, wide pitch swings)

Vibrato Depth Character:

Value	Depth	Pitch Deviation	Musical Context
0	0.0	±0 cents	No vibrato (static pitch)
32	0.025	±2.5 cents	Subtle shimmer (strings)
64	0.05	±5 cents	Natural vibrato (vocals)
96	0.075	±7.5 cents	Expressive vibrato (solos)
127	0.1	±10 cents	Extreme vibrato (SFX, whammy)

Musical Applications:

- **Natural vibrato:** Values 40-80 for realistic vocal/string vibrato
- **Subtle animation:** Values 10-30 for gentle pitch movement on pads
- **Expressive solos:** Values 80-110 for dramatic pitch modulation

- **Special effects:** Values 110-127 for extreme pitch bends

Performance Tip: Use CC#1 (Modulation Wheel) for real-time vibrato control during performance. CC#78 sets the maximum vibrato depth, whilst CC#1 scales it from 0-100% dynamically.

CC#79: Sound Control 10

Function: Reserved for future use (currently recognised but no action).

Range: 0-127

Status: The Intuition Subsynth recognises CC#79 messages but does not currently assign functionality to this controller. This controller is reserved for future expansion.

Performance Tip: Avoid assigning CC#79 to hardware controllers, as future firmware updates may implement functionality that conflicts with your assignments.

Portamento Controllers

These controllers provide fine control over pitch glide behaviour.

CC#5: Portamento Time

Function: Controls how long it takes to glide from one note to another (when portamento is enabled via CC#65).

Range: 0-127

- **0:** Instant (no glide)
- **64:** Moderate glide (~50ms)
- **127:** Maximum glide (~127ms)

Glide Time Mapping: Approximately linear, 0-127ms range.

Musical Character:

Value	Time	Character
0	0ms	Instant (no portamento)
10	10ms	Very fast glide (subtle)
30	30ms	Fast glide (synth bass)
60	60ms	Moderate glide (synth lead)
100	100ms	Slow glide (dramatic)
127	127ms	Very slow glide (extreme effect)

Musical Applications:

- **Fast glide (10-30ms):** Subtle pitch connection between notes (funk bass, legato leads)
- **Moderate glide (40-80ms):** Classic synthesiser portamento (Moog-style leads)
- **Slow glide (90-127ms):** Dramatic pitch slides (sound design, special effects)

Performance Tip: Enable portamento (CC#65 = 127) and adjust portamento time (CC#5) until you achieve the desired glide speed for your musical style.

CC#84: Portamento Control

Function: Specifies the **source note** for portamento glide.

Range: 0-127 (MIDI note number)

Usage: When CC#84 is sent before a Note On, the Subsynth glides from the CC#84 note to the Note On pitch (instead of from the previous note).

Example:

- 1. Send CC#84 value 60 (Middle C)
- 2. Send Note On 67 (G)
- 3. Result: Glide from C (60) to G (67)

Musical Application: Precise control over portamento source pitch, enabling unconventional glide patterns.

Effects Send Controllers

These controllers route audio to the master reverb and chorus effects.

CC#91: Reverb Send (Effects 1 Depth)

Function: Controls how much signal from the MIDI channel is sent to the master reverb effect.

Range: 0-127

- **0:** No reverb (dry signal only)
- **40:** Moderate reverb (subtle ambience)
- **80:** High reverb (obvious spatial depth)
- **127:** Maximum reverb (wet, ambient)

Send Level Mapping: Linear (0-100% wet/dry mix).

Typical Settings:

Instrument Type	Typical Send	Character
Kick Drum, Bass	0-20	Tight, focused (minimal reverb)
Snare, Toms	30-50	Natural room ambience
Piano, Guitar	40-70	Moderate spatial depth
Strings, Pads	70-100	Lush, ambient
Vocals, Leads	50-80	Present with depth
Special FX	100-127	Maximum ambience

Performance Tip: Each MIDI channel has independent reverb send control, allowing you to place different instruments at different "depths" in the same virtual space.

CC#93: Chorus Send (Effects 3 Depth)

Function: Controls how much signal from the MIDI channel is sent to the master chorus effect.

Range: 0-127

- **0:** No chorus (dry signal only)
- **40:** Subtle thickening
- **80:** Obvious chorus effect (shimmering)
- **127:** Maximum chorus (wide, detuned)

Send Level Mapping: Linear (0-100% wet/dry mix).

Typical Settings:

Instrument Type	Typical Send	Character
Kick Drum, Bass	0	Tight, focused (no chorus)
Electric Piano	40-70	Classic EP shimmer
Strings	50-90	Lush ensemble effect
Synth Lead	30-60	Thickened, wide
Pads	60-100	Maximum width and movement
Guitar (clean)	40-80	1980s chorus sound

Performance Tip: Chorus is particularly effective on sustained sounds (strings, pads, electric piano). Avoid excessive chorus on drums and bass, as it can reduce clarity and punch.

CC#92: Effects 2 Depth (Tremolo)

Function: Controls the depth of tremolo (amplitude modulation) effect.

Range: 0-127 (normalised to 0.0-1.0)

- **0:** No tremolo (static amplitude)
- **64:** Moderate tremolo (50% depth)
- **127:** Maximum tremolo (100% depth, full volume pulsation)

Technical Implementation: Applies LFO modulation to amplitude at 6.0 Hz (typical tremolo rate).

Tremolo Depth Character:

Value	Depth	Character
0	0%	No effect (static volume)
32	25%	Subtle shimmer (gentle pulsation)
64	50%	Moderate tremolo (classic organ effect)
96	75%	Pronounced tremolo (rhythmic pumping)
127	100%	Extreme tremolo (stuttering effect)

Musical Applications:

- **Electric piano:** Subtle tremolo (20-40) for classic EP shimmer
- **Organ:** Moderate tremolo (60-80) for Hammond-style vibrato/tremolo
- **Ambient pads:** Slow, deep tremolo (80-100) for evolving textures
- **Rhythmic effects:** Fast, deep tremolo (100-127) for gating-style effects

Difference from Vibrato (CC#1):

Vibrato (CC#1)	Tremolo (CC#92)
Pitch modulation (frequency)	Amplitude modulation (volume)

Frequency wobble	Volume pulsation
Affects perceived pitch	Affects perceived loudness
Typical: 4-7 Hz	Typical: 6 Hz (fixed rate)

Performance Tip: Tremolo creates rhythmic volume pulsation. For combined pitch and volume modulation, use both CC#1 (Vibrato) and CC#92 (Tremolo) together.

CC#94: Effects 4 Depth (Celeste/Detune)

Function: Controls pitch detune effect (chorus-like detuning without delay).

Range: 0-127

- **64:** No detune (centre, unity pitch)
- **0:** -25 cents (maximum downward detune)
- **127:** +25 cents (maximum upward detune)

Detune Mapping:

Value	Detune	Character
0	-25 cents	Maximum downward detune
32	-12.5 cents	Subtle downward detuning
64	0 cents	No detune (unity pitch)
96	+12.5 cents	Subtle upward detuning
127	+25 cents	Maximum upward detune

Musical Applications:

- **Chorus effect:** Slight detune (± 5 -10 cents, values 52-76 or 90-104) for thickening
- **Ensemble sound:** Moderate detune (± 10 -15 cents, values 42-54 or 102-114) for layered textures
- **12-string simulation:** Subtle upward detune (+5 cents, value 90) for 12-string guitar character
- **Special effects:** Extreme detune (± 20 -25 cents, values 0-20 or 107-127) for dissonant, chorused sound

Difference from CC#93 (Chorus):

CC#93 (Chorus)	CC#94 (Detune)
Delay-based modulation	Pure pitch detuning (no delay)
Time-varying	Static detune amount
Richer, shimmering	Cleaner, direct pitch shift
Higher CPU usage	Lower CPU usage

Performance Tip: Use CC#94 for subtle pitch thickening without the modulation sweep of chorus. Combine with CC#93 (Chorus) for extremely wide, shimmering ensemble effects.

CC#95: Effects 5 Depth (Phaser)

Function: Controls the depth of phaser effect (frequency sweep filter modulation).

Range: 0-127 (normalised to 0.0-1.0)

- **0:** No phaser (filter sweep disabled)
- **64:** Moderate phaser (50% sweep depth)
- **127:** Maximum phaser (100% sweep depth, dramatic frequency notches)

Technical Implementation: Applies filter sweep modulation at 2.0 Hz (typical phaser rate) with variable depth.

Phaser Depth Character:

Value	Depth	Character
0	0%	No effect (static filter)
32	25%	Subtle sweep (gentle tonal movement)
64	50%	Moderate phaser (classic 1970s character)
96	75%	Pronounced phaser (jet-plane effect)
127	100%	Extreme phaser (radical tonal sweeping)

Musical Applications:

- **Electric piano:** Subtle phaser (20-40) for Fender Rhodes shimmer
- **Guitar:** Moderate phaser (50-80) for classic rock/funk guitar tones
- **Synth leads:** Deep phaser (80-100) for sci-fi, spacey effects
- **Sound design:** Extreme phaser (100-127) for dramatic, sweeping textures

Phaser Characteristics:

- **Sweep Rate:** Fixed at 2.0 Hz (typical phaser LFO rate)
- **Filter Type:** Allpass filter creating notches in the frequency spectrum
- **Notch Movement:** Sweeps through the spectrum, creating the characteristic "whooshing" sound

Difference from Flanger:

Phaser (CC#95)	Flanger (per-channel NRPN)
Allpass filter stages	Delay-based comb filtering
Smoother, more subtle	Sharper, more metallic
Fewer notches in spectrum	More pronounced notches
Lower CPU usage	Higher CPU usage (delay buffers)

Performance Tip: Phaser is ideal for vintage electric piano and guitar sounds. For more dramatic, metallic effects, use per-channel flanger (NRPN 150-159) instead.

Channel Mode Controllers

These controllers affect the MIDI channel's operating mode.

CC#120: All Sound Off

Function: Immediately silences all notes on the channel (emergency stop).

Range: 0-127 (value ignored, any value triggers the function)

Behaviour:

- All notes on the channel immediately stop (zero amplitude)
- Does **not** respect sustain pedal or envelope release
- Used for emergency muting or panic situations

Difference from All Notes Off (CC#123):

All Sound Off (CC#120)	All Notes Off (CC#123)
Instant silence	Respects envelope release
Ignores sustain pedal	Respects sustain pedal
Emergency/panic	Musical note termination

CC#121: Reset All Controllers

Function: Resets all controllers to their default values.

Range: 0-127 (value ignored, any value triggers the function)

Controllers Reset:

- **CC#1** (Modulation): 0
- **CC#7** (Volume): 100
- **CC#10** (Pan): 64 (centre)
- **CC#11** (Expression): 127 (full)
- **CC#64** (Sustain): 0 (off)
- **CC#91** (Reverb): 40 (default)
- **CC#93** (Chorus): 0 (off)
- **Pitch Bend:** 8192 (centre)

Musical Application: Sent at the beginning of MIDI files to ensure consistent starting state.

CC#123: All Notes Off

Function: Releases all notes on the channel (respecting envelope release).

Range: 0-127 (value ignored, any value triggers the function)

Behaviour:

- All notes enter release phase (natural decay)
- Respects sustain pedal (if pedal is held, notes remain sustained)
- Musical termination (not emergency stop)

Musical Application: End of song, end of section, or when switching programmes.

CC#124: Omni Mode Off

Function: Disables Omni mode (channel-specific reception).

Range: 0-127 (value ignored)

Effect: The Subsynth responds only to messages on specific MIDI channels (standard mode, already active by default).

CC#125: Omni Mode On

Function: Enables Omni mode (channel responds to all MIDI channels).

Range: 0-127 (value ignored)

Effect: When Omni mode is enabled, the MIDI channel responds to messages on **all 16 MIDI channels** simultaneously. All notes off are triggered before enabling.

Behaviour:

- Incoming MIDI messages on **any channel** (1-16) are processed by the Omni-enabled channel
- Useful for testing or single-channel devices
- Non-standard mode (rarely used in modern MIDI setups)

Musical Application: Omni mode is primarily used for:

- **Hardware testing:** Ensure MIDI device responds regardless of channel setting
- **Single-channel devices:** Vintage MIDI gear with limited channel switching
- **Emergency playback:** Play MIDI files without knowing which channels are used

Performance Tip: Omni mode is **not recommended** for multi-channel production, as it defeats the purpose of MIDI channel separation. Keep Omni mode off (CC#124) for standard operation.

CC#126: Mono Mode On

Function: Enables monophonic mode (one voice per channel).

Range: 0-127 (number of channels, typically 1)

- **Value:** Specifies number of channels (typically 1 for strict monophonic)
- **Effect:** MIDI channel limited to 1 simultaneous voice

Behaviour:

1. **All notes off** triggered before enabling mono mode
2. **Voice limit** set to 1 voice per MIDI channel
3. **New note priority:** Latest note always terminates previous note
4. **Voice stealing:** Immediate (no fade-out, instant cutoff)

Monophonic Voice Allocation:

Polyphonic Mode (CC#127)	Monophonic Mode (CC#126)
Multiple notes play	One note plays at a time
Chords possible	Chords impossible (last note wins)
32 voices per channel	1 voice per channel
Standard mode	Specialised mode

Musical Applications:

- **Lead synthesiser:** Classic monophonic synth leads (Minimoog, ARP Odyssey style)
- **Bass lines:** Tight, focused bass with no chord bleed
- **Solo instruments:** Wind/brass emulation (saxophone, trumpet)
- **Portamento phrases:** Smooth pitch glides require monophonic mode

Combined with Portamento:

Mono mode is essential for realistic portamento (pitch glide) behaviour:

1. Enable Mono Mode: CC#126 = 1

2. Enable Portamento: CC#65 = 127
3. Set Portamento Time: CC#5 = 50 (moderate glide)
4. Play legato notes: Each new note glides from previous pitch

Performance Tip: Use Mono Mode for lead synthesiser solos where you want authentic vintage monophonic synth behaviour. Combine with CC#65 (Portamento) and CC#68 (Legato) for expressive, gliding lead lines.

CC#127: Poly Mode On

Function: Enables polyphonic mode (multiple voices per channel).

Range: 0-127 (value ignored)

Effect: Restores polyphonic operation, allowing multiple simultaneous voices per MIDI channel.

Behaviour:

1. **All notes off** triggered before enabling poly mode
2. **Voice limit** set to 32 voices per MIDI channel (default)
3. **Chords enabled:** Multiple notes can play simultaneously
4. **Voice stealing:** Priority-based (age, envelope, velocity)

Voice Allocation:

- **Default:** 32 voices per MIDI channel
- **Shared:** Multiple MIDI channels share synthesis channel pools
- **Wave-type-based:** Voice allocation depends on patch waveform type

Musical Applications:

- **Chords:** Piano, guitar, strings (any harmonic content)
- **Pads:** Lush, layered textures with multiple sustained notes
- **Complex arrangements:** Multi-voice orchestration
- **Standard operation:** Default mode for most music production

Polyphonic vs Monophonic:

Feature	Polyphonic (CC#127)	Monophonic (CC#126)
Voice count	32 voices per channel	1 voice per channel
Chords	✅ Full chords possible	❌ One note only
Portamento behaviour	Glides from average pitch	Glides from previous note
Typical use	95% of all music	Specialised lead lines
CPU usage	Higher (more voices)	Lower (single voice)

Performance Tip: Poly mode is the default and standard mode for the Intuition Subsynth. Only switch to Mono mode (CC#126) when you specifically need monophonic behaviour for lead synthesiser lines.

14-Bit Controllers (MSB/LSB Pairs)

For higher resolution control, the MIDI specification defines **14-bit controllers** that combine two 7-bit messages (MSB + LSB) to provide 16,384 discrete values instead of 128.

14-Bit Protocol:

1. Send MSB (Controller 0-31) with coarse value (0-127)
2. Send LSB (Controller 32-63) with fine value (0-127)
3. Combined value = (MSB × 128) + LSB (range: 0-16383)

Supported 14-Bit Controllers:

CC#0/32: Bank Select

Function: Selects instrument bank (patchset) before Program Change.

Range: 0-16383 (14-bit), Banks 0-4 supported

Supported Banks:

- **Bank 0: Default/Reset** - Resets to your startup patchset (from `--patchset=NAME`, or Sidney if not specified)
- **Bank 1:** AYYMe (AY-3-8910/YM2149 PSG chip emulation)
- **Bank 2:** Rawland (Roland-inspired synthesis)
- **Bank 3:** PixelaTED (Commodore TED 7360/8360 chip emulation)
- **Bank 4:** EenBeetje (ZX Spectrum 1-bit beeper)
- **Banks 5-16383:** Reserved for future use

Bank 0 Behaviour: Many MIDI files send Bank Select 0 as a "reset to default" message. The synthesiser respects this by resetting to your chosen default patchset, allowing you to use `--patchset=AYYMe` (for example) without MIDI files overriding your choice.

Bank Selection Method:

The Intuition Subsynth uses **CC#0 (Bank Select MSB)** to select the bank. CC#32 (Bank Select LSB) is accepted but ignored.

Procedure:

1. Send CC#0 (Bank Select MSB) = [bank number 0-4]
2. Send CC#32 (Bank Select LSB) = 0 (optional, ignored by implementation)
3. Send Program Change (0-127) to trigger the bank switch

Example (Switch to Rawland patchset, Piano programme):

```
CC#0 (Bank Select MSB) = 2      (Rawland bank)
CC#32 (Bank Select LSB) = 0      (optional, ignored)
PC (Program Change) = 0      (Piano)
```

Note: Only CC#0 determines the bank number. The LSB (CC#32) is stored but not used in bank selection.

Behaviour:

- Bank switching is **global** (affects all 16 MIDI channels simultaneously)
- Selected bank remains active until changed by another Bank Select message
- Invalid bank numbers (>4) are ignored (current bank remains active)
- Bank 0 resets to your default patchset (from `--patchset=NAME` at startup)
- Bank switching loads embedded factory patchsets from the synthesiser firmware

CC#1/33: Modulation (Fine)

Function: High-resolution modulation wheel control.

Range: 0-16383 (14-bit)

Usage: Provides smooth, continuous modulation depth with 128× finer resolution than 7-bit CC#1.

CC#7/39: Volume (Fine)

Function: High-resolution channel volume control.

Range: 0-16383 (14-bit)

Usage: Enables precise volume automation with inaudible steps.

CC#10/42: Pan (Fine)

Function: High-resolution stereo panning control.

Range: 0-16383 (14-bit)

Usage: Provides smooth pan sweeps with 16,384 stereo positions instead of 128.

Registered Parameter Numbers (RPN)

RPNs provide access to standardised extended parameters beyond the 128 MIDI CC messages.

RPN Protocol:

1. Send CC#101 (RPN MSB) with parameter MSB
2. Send CC#100 (RPN LSB) with parameter LSB
3. Send CC#6 (Data Entry MSB) with value MSB
4. Send CC#38 (Data Entry LSB) with value LSB (optional, for 14-bit)

RPN #0: Pitch Bend Sensitivity

Function: Sets the pitch bend range in semitones and cents.

Parameters:

- **MSB (CC#6):** Semitones (0-24)
- **LSB (CC#38):** Cents (0-100)

Example: Set pitch bend range to ± 2 semitones (default):

```
CC#101 value 0 (RPN MSB)
CC#100 value 0 (RPN LSB)
CC#6 value 2 (Data Entry MSB: 2 semitones)
CC#38 value 0 (Data Entry LSB: 0 cents)
```

Typical Ranges:

- **± 1 semitone:** Subtle pitch bends (classical music)
- **± 2 semitones:** GM default, most popular music
- **± 7 semitones:** Guitar-style bends (blues, rock)
- **± 12 semitones:** One octave (electronic music, sound design)
- **± 24 semitones:** Two octaves (extreme effects)

RPN #1: Fine Tuning

Function: Adjusts the pitch of the entire MIDI channel in fine steps.

Range: 0-16383 (14-bit)

- **8192:** No detuning (concert pitch)
- **0:** -100 cents (-1 semitone)
- **16383:** +100 cents (+1 semitone)

Resolution: $100/8192 = 0.0122$ cents per step (virtually continuous).

Musical Application: Detune multiple channels slightly (e.g. ± 5 cents) to create thick, chorused ensemble effects.

RPN #2: Coarse Tuning

Function: Adjusts the pitch of the entire MIDI channel in semitone steps.

Range: 0-127 (7-bit)

- **64:** No transposition (0 semitones)
- **0:** -64 semitones
- **127:** +63 semitones

Musical Application: Transpose an entire MIDI channel up or down by semitones without changing the MIDI note numbers.

15. Advanced NRPN Control

NRPN (Non-Registered Parameter Numbers) provides access to the Intuition Subsynth's **4360+ advanced synthesis parameters** with **14-bit resolution** (16,384 discrete values per parameter). This system enables professional sound programming far beyond the capabilities of standard MIDI CC messages.

What are NRPNs?

Standard MIDI CC messages provide only 128 addressable parameters (CC#0-127) with 7-bit resolution (0-127 values). This is insufficient for professional synthesis control, which requires:

Limitations of Standard MIDI CC:

- Only 128 parameters total
- Only 128 discrete values (audible steps in many parameters)
- No per-channel individual control
- Limited to basic mixing and effects sends

Advantages of NRPN System:

- **4360+ addressable parameters** (unlimited address space)
- **14-bit resolution** (16,384 values = inaudible steps)
- **Per-channel control** (4160 parameters across 320 synthesis channels)
- **Named parameters** (200+ global effects and processing controls)
- **Real-time safe** (<125ns processing time per parameter change)

NRPN Message Protocol

NRPNs use a **four-message sequence** to select a parameter and set its value:

NRPN Protocol Steps:

1. **Select Parameter MSB:** Send CC#99 (NRPN MSB) with upper 7 bits of parameter address
2. **Select Parameter LSB:** Send CC#98 (NRPN LSB) with lower 7 bits of parameter address
3. **Set Value MSB:** Send CC#6 (Data Entry MSB) with upper 7 bits of value
4. **Set Value LSB** (optional): Send CC#38 (Data Entry LSB) with lower 7 bits of value

14-Bit Address Calculation:

NRPN Address = (CC#99 $\times$ 128) + CC#98

Range: 0-16383 (16,384 unique parameters)

14-Bit Value Calculation:

NRPN Value = (CC#6 × 128) + CC#38
Range: 0-16383 (16,384 discrete values)

Example: Set reverb decay (NRPN #7) to 10,000 (61% of maximum):

Step 1: CC#99 value 0 (NRPN MSB: 7 ÷ 128 = 0)
Step 2: CC#98 value 7 (NRPN LSB: 7 mod 128 = 7)
Step 3: CC#6 value 78 (Value MSB: 10000 ÷ 128 = 78)
Step 4: CC#38 value 16 (Value LSB: 10000 mod 128 = 16)

Verification:

- Address: (0 × 128) + 7 = **7** (Reverb Decay) ✓
- Value: (78 × 128) + 16 = **10,000** ✓

NRPN Parameter Organisation

The Intuition Subsynth's NRPN system is organised into two categories:

1. Named Parameters (0-299): ~200 global parameters

- **0-30:** Effects and processing (delay, chorus, reverb, bit-crusher, overdrive, compressor, etc.)
- **130-139:** Master compressor extended controls (10 parameters)
- **140-149:** Ping-pong delay controls (10 parameters)
- **150-159:** Per-channel flanger/phaser controls (10 parameters)
- **160-173:** Wave-specific advanced controls (14 parameters)
- **200-215:** Advanced envelope controls (16 parameters)
- **250-255:** Advanced filter controls (6 parameters)

2. Per-Channel Parameters (1000-5159): 4160 individual channel parameters

- **13 parameter types × 320 synthesis channels = 4160 parameters**
- Addressing formula: NRPN = 1000 + (parameter_type × 320) + channel_index

Named NRPN Parameters (0-299)

These global parameters control master effects, wave-specific characteristics, and advanced synthesis features.

Effects and Processing (NRPN 0-30)

Note: Use CC#91 (Reverb Send) and CC#93 (Chorus Send) for master reverb and chorus control. Master delay control is via Ping-Pong Delay (NRPN 140-149).

NRPN	Parameter	Range	Description
0	Delay Time (Fine)	0-16383	Per-channel delay time (proxy via delay mix control)
1	Delay Feedback	0-16383	Per-channel delay feedback amount (0-100%, limited)
9	Bit-Crush Bits	0-16383	Per-channel bit depth (1-16 bits)
11	Overdrive Pre-Gain	0-16383	Per-channel pre-overdrive gain (0.0-2.0)
14	Compressor Threshold	0-16383	Master compressor threshold (-60 to 0 dB)
15	Compressor Ratio	0-16383	Master compressor ratio (1:1 to 100:1, logarithmic)
16	Compressor Attack	0-16383	Master compressor attack (0.01-100ms, logarithmic)
17	Compressor Release	0-16383	Master compressor release (1-5000ms, logarithmic)

18	Fade In Time	0-16383	Per-channel fade-in rate (sample-rate dependent)
23	Velocity Sensitivity	0-16383	Per-channel velocity response scaling (0.0-1.0)
24	Master Tune	0-16383	Global tuning offset (± 50 cents, 8192 = concert pitch)
30	Noise Amplitude Bits	0-16383	Noise channel amplitude quantisation (1-16 bits)

Reserved for Future Use (constants defined but not yet implemented):

- **NRPN 2-8:** Delay mix, stereo offset, chorus rate/depth/feedback, reverb decay/diffusion
- **NRPN 10, 12-13:** Bit-crush sample rate, overdrive post-gain/asymmetry
- **NRPN 25:** Stereo width (global)

Practical Example 1: Create a lo-fi degraded sound with dynamics:

NRPN #9 value 6144 (Bit-Crush: 6-bit, ~37% of range)
NRPN #11 value 6554 (Overdrive Pre-Gain: 0.8, subtle warmth)
NRPN #14 value 8192 (Compressor Threshold: -30dB, moderate compression)
NRPN #15 value 6554 (Compressor Ratio: 4:1, ~40% of range)
NRPN #16 value 3277 (Compressor Attack: ~2ms, fast response)
NRPN #17 value 8192 (Compressor Release: ~250ms, moderate release)

Result: Warm, degraded, lo-fi sound with controlled dynamics and subtle saturation.

Practical Example 2: Configure velocity sensitivity for expressive playing:

NRPN #23 value 13107 (Velocity Sensitivity: 80%, expressive dynamics)

Result: Note velocity significantly affects amplitude, creating dynamic contrast between soft and loud playing.

Master Compressor Controls (NRPN 130-139)

NRPN	Parameter	Range	Description
130	Compressor Enable	0-16383	Enable/disable (>8192 = on)
131	Compressor Mode	0-16383	0=off, 1=compress, 2=limit, 3=maximise
132	Compressor Detection	0-16383	0=peak, 1=RMS, 2=true peak
133	Compressor Knee	0-16383	Knee width (0-6dB, 0 = hard knee)
134	Compressor Makeup	0-16383	Makeup gain (-20 to +40dB)
135	Compressor Mix	0-16383	Wet/dry mix (0-100%, for parallel compression)
136	Compressor Lookahead	0-16383	Lookahead enable (>8192 = on)
137	Compressor Auto Makeup	0-16383	Auto makeup gain (>8192 = on)
138	Compressor Sidechain HPF	0-16383	Sidechain high-pass filter (>8192 = on)
139	Compressor Stereo Link	0-16383	Stereo linking (0-100%)

Practical Example 2: New York-style parallel compression:

NRPN #130 value 16383 (Enable: on)
NRPN #131 value 5461 (Mode: compress, ~33% of range)
NRPN #132 value 5461 (Detection: RMS, ~33% of range)
NRPN #133 value 4096 (Knee: 1.5dB, soft knee)
NRPN #134 value 12288 (Makeup: +15dB)

NRPN #135 value 6554 (Mix: 40%, parallel blend)
NRPN #139 value 16383 (Stereo Link: 100%, glued stereo image)

Result: Aggressive compression blended 40% with dry signal for punchy, dense mix with retained transients.

Ping-Pong Delay Controls (NRPN 140-149)

NRPN	Parameter	Range	Description
140	Ping-Pong Enable	0-16383	Enable/disable (>8192 = on)
141	Ping-Pong Left Time	0-16383	Left channel delay (1-2000ms)
142	Ping-Pong Right Time	0-16383	Right channel delay (1-2000ms)
143	Ping-Pong Feedback	0-16383	Feedback amount (0.0-0.95)
144	Ping-Pong Mix	0-16383	Wet/dry mix (0.0-1.0)
145	Ping-Pong Crossfeed	0-16383	Cross-channel feedback (0.0-1.0)
146	Ping-Pong Highcut	0-16383	High-frequency roll-off (0.1-1.0)
147	Ping-Pong Spread	0-16383	Stereo spread multiplier (0.5-2.0)
148	Ping-Pong Mod Depth	0-16383	Tape flutter modulation depth (0.0-1.0)
149	Ping-Pong Mod Rate	0-16383	Modulation rate (0.1-5.0Hz)

Practical Example 3: Ambient dub delay:

NRPN #140 value 16383 (Enable: on)
NRPN #141 value 8192 (Left Time: 500ms, ~50% of range)
NRPN #142 value 12288 (Right Time: 750ms, ~75% of range)
NRPN #143 value 12288 (Feedback: 75%, sustained echoes)
NRPN #144 value 8192 (Mix: 50%, equal wet/dry)
NRPN #145 value 6554 (Crossfeed: 40%, stereo interaction)
NRPN #146 value 10240 (Highcut: 62.5%, darker repeats)
NRPN #148 value 4096 (Mod Depth: 25%, tape flutter)
NRPN #149 value 3277 (Mod Rate: 1Hz, slow warble)

Result: Spacious, dark, modulated delay with asymmetric left/right timing and long, evolving echoes.

Per-Channel Flanger/Phaser (NRPN 150-159)

The per-channel flanger/phaser system provides independent modulation effects for each MIDI channel, enabling complex layered textures impossible with global effects.

Flanger vs Phaser:

Characteristic	Flanger	Phaser
Effect Type	Delay-based comb filtering	Allpass filter stages
Sound Character	Metallic, jet-plane sweeps	Smoother, vocal-like sweeps
Notch Pattern	Evenly spaced harmonics	Non-linear harmonic spacing
CPU Usage	Higher (delay buffers)	Lower (filter calculations)
Typical Applications	Guitar, synth leads	Electric piano, Rhodes

Feedback Character	Sharp, metallic resonance	Smooth, phase-shifted peaks
--------------------	---------------------------	-----------------------------

NRPN	Parameter	Range	Description
150	Flanger Enable	0-16383	Enable/disable per-channel flanger (>8192 = on)
151	Flanger Mode	0-16383	0-8191=flanger, 8192-16383=phaser
152	Flanger Rate	0-16383	LFO modulation rate (0.1-10.0 Hz)
153	Flanger Depth	0-16383	Modulation depth (0.0-1.0)
154	Flanger Feedback	0-16383	Feedback amount (0.0-0.95, limited)
155	Flanger Mix	0-16383	Wet/dry mix (0.0-1.0)
156	Flanger Base Delay	0-16383	Base delay time (1-20ms, flanger mode only)
157	Flanger Delay Range	0-16383	Delay modulation range (1-20ms, flanger mode)
158	Phaser Stages	0-16383	Number of allpass stages (1-8, phaser mode)
159	Phaser Freq Range	0-16383	Frequency modulation range (100-8000Hz, phaser)

Practical Example 1: Classic electric piano phaser (Fender Rhodes shimmer):

```
# MIDI Channel 1 - Electric Piano
NRPN #150 value 16383 (Enable: on)
NRPN #151 value 16383 (Mode: phaser)
NRPN #152 value 2048 (Rate: ~1.25 Hz, slow sweep)
NRPN #153 value 9830 (Depth: 60%, moderate modulation)
NRPN #154 value 6554 (Feedback: 40%, resonant peaks)
NRPN #155 value 6554 (Mix: 40%, balanced wet/dry)
NRPN #158 value 6554 (Phaser Stages: 4 stages, classic character)
NRPN #159 value 8192 (Freq Range: ~4000Hz, midrange sweep)
```

Result: Smooth, shimmering electric piano with classic 1970s phaser character (Fender Rhodes/Wurlitzer style).

Practical Example 2: Aggressive flanger for synth leads (jet-plane effect):

```
# MIDI Channel 2 - Synth Lead
NRPN #150 value 16383 (Enable: on)
NRPN #151 value 4096 (Mode: flanger)
NRPN #152 value 4915 (Rate: ~3.0 Hz, moderate speed)
NRPN #153 value 14746 (Depth: 90%, dramatic sweep)
NRPN #154 value 11469 (Feedback: 70%, metallic resonance)
NRPN #155 value 9830 (Mix: 60%, effect-heavy)
NRPN #156 value 4096 (Base Delay: ~5ms, short delay)
NRPN #157 value 8192 (Delay Range: ~10ms, wide modulation)
```

Result: Aggressive, metallic flanger with pronounced jet-plane sweeps and sharp resonant peaks.

Practical Example 3: Subtle flanger for guitar (chorus-like thickening):

```
# MIDI Channel 3 - Clean Guitar
NRPN #150 value 16383 (Enable: on)
NRPN #151 value 2048 (Mode: flanger)
NRPN #152 value 1638 (Rate: ~1.0 Hz, slow sweep)
NRPN #153 value 4915 (Depth: 30%, subtle modulation)
NRPN #154 value 3277 (Feedback: 20%, minimal resonance)
NRPN #155 value 4915 (Mix: 30%, dry-heavy)
```

NRPN #156 value 1638 (Base Delay: ~2ms, tight delay)
NRPN #157 value 3277 (Delay Range: ~4ms, narrow modulation)

Result: Subtle, chorus-like flanger adding gentle movement and width without overpowering the dry guitar sound.

Per-Channel Routing:

Each MIDI channel (1-16) has an independent flanger/phaser processor. This enables:

- **Layered effects:** Flanger on lead (Ch 2), phaser on electric piano (Ch 3), no effect on bass (Ch 4)
- **Independent parameter control:** Different rates, depths, and mix levels per channel
- **Dual flanger/phaser:** Multiple channels with different flanger/phaser settings running simultaneously
- **Effect automation:** Per-channel NRPN automation enables evolving modulation effects

Performance Tip: Per-channel flangers consume more CPU than global effects. Use sparingly on channels requiring modulation, and use CC#93 (Chorus Send) for general thickening on less critical channels.

Wave-Specific Advanced Controls (NRPN 160-173)

These NRPNs provide fine control over wave-type-specific parameters that define the character of square, noise, and ring modulation synthesis channels. Each parameter only affects synthesis channels matching the specified wave type.

NRPN	Parameter	Range	Description
160	Square Duty Cycle (Fine)	0-16383	Pulse width for square waves (5-95%, 14-bit precision)
161	Square PWM Depth (Fine)	0-16383	Pulse width modulation depth (0.0-1.0, enables PWM if >1%)
162	Square PWM Rate (Fine)	0-16383	PWM LFO modulation rate (0-20 Hz, enables PWM if >0.01Hz)
166	Noise Filter Frequency	0-16383	Filter cutoff for noise channels (normalised 0.0-1.0)
167	Noise Filter Resonance	0-16383	Filter Q/resonance for noise channels (normalised 0.0-1.0)
171	Ring Mod Carrier Waveform	0-127	Carrier waveform (0-42=sine, 43-85=triangle, 86-127=sawtooth)

Wave Type Targeting:

- **NRPN 160-162:** Only affect **square wave channels** (channels 0-63, wave type 0)
- **NRPN 166-167:** Only affect **noise channels** (channels 192-255, wave type 3)
- **NRPN 171:** Affects channels with **ring modulation enabled** (any wave type)

Practical Example 1: Classic vintage synthesiser lead with PWM (Oberheim/Prophet style):

```
# MIDI Channel 0 - PWM Lead Patch
# Configure square wave with slow PWM sweep
NRPN #160 value 8192      (Duty Cycle: 50% centre position, symmetric waveform)
NRPN #161 value 13107     (PWM Depth: 80%, wide timbral sweep)
NRPN #162 value 655       (PWM Rate: 0.8 Hz, slow oscillation for evolving tone)
```

Result: A rich, evolving synthesiser lead with slow pulse width modulation that sweeps from thin nasal tones to thick hollow tones, characteristic of 1970s-80s analogue polysynths. The 50% starting duty cycle ensures symmetrical

modulation.

Practical Example 2: Filtered noise percussion (hi-hats and cymbals):

```
# MIDI Channel 10 - Drum Kit with custom hi-hat
# Use noise channels with resonant high-pass filtering
NRPN #166 value 14746  (Filter Freq: 90%, very high cutoff for bright metallic tone)
NRPN #167 value 11468  (Filter Res: 70%, emphasises high frequencies with ringing)
```

Result: Bright, metallic hi-hat and cymbal sounds with pronounced high-frequency emphasis and resonant ringing, similar to Roland TR-808/909 cymbals. The high resonance adds shimmer and sustain.

Practical Example 3: Metallic bell tones with ring modulation:

```
# MIDI Channel 1 - Bell Patch with Ring Mod
# Configure triangle carrier for smooth metallic overtones
NRPN #171 value 64      (Carrier Waveform: triangle, smoother than sine)
CC#4 (Foot Control) 90  (Ring mod amount: 71%, strong but not overwhelming)
```

Result: Bell-like metallic timbres with smooth, harmonic overtones from triangle wave carrier. Triangle carrier produces less harsh inharmonic content than sawtooth whilst maintaining clear metallic character.

Usage Tips:

- **PWM sweet spot:** Start with 50% duty cycle (NRPN 160 = 8192) and 60-80% depth (NRPN 161 = 9830-13107) for classic analogue character
- **PWM rate ranges:** 0.5-2 Hz for slow evolving pads, 4-8 Hz for vibrato-like modulation, 10-20 Hz for aggressive timbral tremolo
- **Noise filtering:** High-pass (CC#76 = 96-127) with NRPN 166 > 12000 creates realistic cymbals; low-pass (CC#76 = 0-31) with NRPN 166 < 6000 creates wind/ocean effects
- **Ring mod carriers:** Sine (0-42) for pure bell tones, triangle (43-85) for harmonic bells, sawtooth (86-127) for harsh metallic effects
- **PWM enabling:** PWM automatically enables when NRPN 161 or 162 exceed threshold (depth >1% or rate >0.01Hz)

Performance Tip: PWM consumes additional CPU per square wave channel. If using PWM on multiple MIDI channels simultaneously, monitor CPU usage and reduce PWM depth or disable PWM (NRPN 161/162 = 0) on less critical channels if needed.

Aftertouch Modulation Depth (NRPN 216-218)

Status:  **Fully Implemented** (October 2025) - Polyphonic Aftertouch (0xA0) and Channel Pressure (0xD0)

These NRPNs configure how much aftertouch pressure affects each modulation target per MIDI channel.

NRPN	Parameter	Range	Default	Description
216	Aftertouch to Filter	0-16383	8192	Filter cutoff modulation depth (±20% range)
217	Aftertouch to Volume	0-16383	8192	Volume modulation depth (±30% range)
218	Aftertouch to Vibrato	0-16383	8192	Vibrato depth modulation depth (±50% range)

How It Works:

- **Pressure Range:** 0-127 from MIDI message, normalised to 0.0-1.0 (0.5 = neutral, no modulation)
- **Modulation Formula:** $modulatedValue = baseValue \times (1.0 + (pressure - 0.5) \times depth \times range)$
- **Real-time:** Modulation applied continuously in audio generation loop

- **Thread-safe:** Lock-free atomic operations for MIDI/audio communication
- **Zero drift:** Always modulates from stored base values (eliminates cumulative bug)

Practical Example: Configure expressive aftertouch for a lead synthesiser:

```
# MIDI Channel 0 - Lead Synth with heavy aftertouch response
NRPN #216 value 13107 (Filter: 80% sensitivity, dramatic brightness)
NRPN #217 value 9830 (Volume: 60% sensitivity, moderate dynamics)
NRPN #218 value 14746 (Vibrato: 90% sensitivity, expressive pitch)
```

Result: Pressing harder opens the filter dramatically for brightness, increases volume moderately, and adds strong vibrato for expressive solos. Light pressure darkens tone, reduces volume, and minimizes vibrato.

Usage Tips:

- **Monophonic leads:** High aftertouch sensitivity (80-100%) for maximum expression
- **Polyphonic pads:** Lower sensitivity (30-50%) for subtle dynamics without overpowering
- **Drums:** Disable aftertouch (0%) on channel 10 (drums don't typically use aftertouch)
- **Bass:** Moderate filter sensitivity (40-60%) for tonal variation, low vibrato to maintain pitch stability

Advanced Envelope Controls (NRPN 200-202)

These NRPNs configure how note velocity affects ADSR envelope parameters, enabling velocity-sensitive envelope shaping beyond standard velocity-to-volume mapping. Each parameter uses a normalised scaling factor where 0.5 is neutral (no velocity influence).

NRPN	Parameter	Range	Default	Description
200	Velocity → Attack Time	0-16383	8192	Velocity scaling for attack (0.5=neutral)
201	Velocity → Sustain Level	0-16383	8192	Velocity scaling for sustain (0.5=neutral)
202	Velocity → Release Time	0-16383	8192	Velocity scaling for release (0.5=neutral)

Scaling Behaviour:

The NRPN value (0-16383) is normalised to a scaling factor (0.0-1.0):

- **0.5 (8192):** Neutral - no velocity influence (default)
- **< 0.5 (0-8191):** Inverse relationship - softer velocity increases parameter
- **> 0.5 (8193-16383):** Positive relationship - harder velocity increases parameter

Velocity-to-Attack (NRPN 200):

- **0.0 (value 0):** Softer velocity = slower attack, harder velocity = faster attack (inverse)
- **0.5 (value 8192):** Attack time independent of velocity (neutral, default)
- **1.0 (value 16383):** Softer velocity = faster attack, harder velocity = slower attack (positive)

Velocity-to-Sustain (NRPN 201):

- **0.0 (value 0):** Softer velocity = lower sustain level, harder velocity = higher sustain (inverse)
- **0.5 (value 8192):** Sustain level independent of velocity (neutral, default)
- **1.0 (value 16383):** Softer velocity = higher sustain level, harder velocity = lower sustain (positive)

Velocity-to-Release (NRPN 202):

- **0.0 (value 0):** Softer velocity = slower release, harder velocity = faster release (inverse)
- **0.5 (value 8192):** Release time independent of velocity (neutral, default)
- **1.0 (value 16383):** Softer velocity = faster release, harder velocity = slower release (positive)

Practical Example 1: Percussive piano with velocity-sensitive attack (harder hits = faster attack):

```
# MIDI Channel 0 - Acoustic Piano Patch
# Configure inverse attack scaling for realistic piano response
NRPN #200 value 3277      (Velocity--Attack: 0.2, inverse scaling)
NRPN #201 value 8192      (Velocity--Sustain: 0.5, neutral - use standard velocity-to-volume)
NRPN #202 value 8192      (Velocity--Release: 0.5, neutral)
CC#73 value 20            (Attack Time: ~1500ms base value)
```

Result: Soft notes have slow, gentle attacks (1500ms) whilst hard notes have fast, percussive attacks (300ms). This creates realistic piano dynamics where soft playing has more gradual onset and hard playing has immediate impact.

Practical Example 2: Expressive string ensemble (softer playing = longer sustain):

```
# MIDI Channel 1 - String Ensemble
# Configure positive sustain scaling for expressive bowing
NRPN #200 value 8192      (Velocity--Attack: 0.5, neutral)
NRPN #201 value 14746     (Velocity--Sustain: 0.9, positive scaling - softer = higher sustain)
NRPN #202 value 13107     (Velocity--Release: 0.8, positive - softer = faster release)
CC#72 value 80            (Release Time: ~9500ms base value)
```

Result: Soft bowing maintains high sustain levels with quick decay (gentle fade), whilst hard attacks have lower sustain with slower release (dramatic tail). This emulates string bowing pressure dynamics.

Practical Example 3: Aggressive synthesiser bass (velocity-independent envelope for consistency):

```
# MIDI Channel 2 - Synth Bass
# Disable velocity envelope scaling for consistent punch regardless of playing dynamics
NRPN #200 value 8192      (Velocity--Attack: 0.5, neutral - consistent snap)
NRPN #201 value 8192      (Velocity--Sustain: 0.5, neutral - consistent body)
NRPN #202 value 8192      (Velocity--Release: 0.5, neutral - consistent decay)
CC#73 value 0             (Attack Time: 0ms, instant attack)
CC#72 value 40            (Release Time: ~3800ms)
```

Result: Bass notes have identical envelope shapes regardless of velocity (only volume changes via standard MIDI velocity). This creates consistent, punchy bass lines in electronic music where timing precision and tonal consistency are critical.

Usage Tips:

- **Default is neutral (8192):** All three parameters default to 0.5 (value 8192), meaning velocity affects volume only (standard MIDI behaviour)
- **Inverse attack for realism:** Acoustic instruments (piano, guitar, mallets) typically benefit from inverse attack scaling (NRPN 200 < 8192), where harder hits have faster attacks
- **Positive sustain for expression:** String and wind instruments often benefit from positive sustain scaling (NRPN 201 > 8192), where softer playing maintains sustain
- **Combine with CC#7/CC#11:** These NRPNs shape envelope response to velocity independently of volume control (CC#7) and expression (CC#11)
- **Subtle settings recommended:** Start with ±20% deviation from neutral (6554-9830 range) and increase gradually to avoid extreme, unnatural behaviour
- **Per-channel independence:** Each MIDI channel has independent velocity-to-envelope scaling, allowing piano on channel 1 and synth bass on channel 2 with different responses

Performance Tip: Velocity-to-envelope scaling is particularly effective for live performance with velocity-sensitive MIDI keyboards. Set neutral values (8192) during sound design, then adjust per-channel scaling based on playing style and musical context.

Advanced Filter Controls (NRPN 250-255)

NRPN	Parameter	Range	Description
------	-----------	-------	-------------

250	Filter Keyboard Track	0-16383	Keyboard tracking amount (0-100%)
251	Filter Envelope Amount	0-16383	Envelope modulation depth (-100% to +100%)
252	Filter Self-Oscillation	0-16383	Self-oscillation amount (0-100%)
253	Filter Drive	0-16383	Pre-filter drive/saturation (0-200%)
254	Filter Warmth	0-16383	Analogue-style non-linearity (0-100%)
255	Filter Distortion	0-16383	Filter distortion amount (0-100%)

Practical Example 4: Aggressive acid bass filter:

NRPN #250 value 16383 (Keyboard Track: 100%, consistent timbre)
NRPN #251 value 13107 (Envelope Amount: +80%, opening filter)
NRPN #252 value 13107 (Self-Oscillation: 80%, whistle character)
NRPN #253 value 9830 (Filter Drive: 120%, saturated input)
NRPN #255 value 6554 (Distortion: 40%, gritty resonance)

Result: Screaming, resonant filter with envelope-controlled sweeps, saturated input, and aggressive distortion (classic TB-303 acid sound).

Per-Channel NRPN System (1000-5159)

The per-channel NRPN system provides **individual control over each of the 320 synthesis channels**. This enables unprecedented sound programming capabilities impossible with standard MIDI CC messages.

Why Per-Channel Control?

Standard MIDI CC messages control an entire MIDI channel (all notes playing on channels 1-16). Per-channel NRPNs control **individual synthesis channels** (0-319), enabling:

- **Individual channel effects:** Apply overdrive to channel 0 (first square wave) only
- **Per-note parameter automation:** Automate filter cutoff on specific notes/channels
- **Complex layering:** Different envelope times for each layer in a multi-layered patch
- **Advanced voice programming:** Program specific synthesis channels with unique characteristics

Addressing Formula:

$$\text{NRPN Address} = 1000 + (\text{parameter_type} \times 320) + \text{channel_index}$$

Where:

parameter_type = 0-12 (13 parameter types)
channel_index = 0-319 (320 synthesis channels)

13 Parameter Types:

Type	Parameter	Address Range	Description
0	Overdrive	1000-1319	Per-channel overdrive amount
1	Filter Frequency	1320-1639	Per-channel filter cutoff
2	Filter Resonance	1640-1959	Per-channel filter Q factor
3	Filter Type	1960-2279	Per-channel filter mode (LP/HP/BP)
4	PWM Depth	2280-2599	Per-channel pulse width modulation depth

5	PWM Rate	2600-2919	Per-channel PWM rate
6	LFO Rate	2920-3239	Per-channel LFO frequency
7	Pan	3240-3559	Per-channel stereo position
8	Volume	3560-3879	Per-channel amplitude
9	Attack	3880-4199	Per-channel envelope attack time
10	Decay	4200-4519	Per-channel envelope decay time
11	Sustain	4520-4839	Per-channel envelope sustain level
12	Release	4840-5159	Per-channel envelope release time

Address Calculation Examples:

Example 1: Set overdrive for channel 0 (first square wave channel):

Parameter Type: 0 (Overdrive)
Channel Index: 0
NRPN Address = $1000 + (0 \times 320) + 0 = 1000$

Example 2: Set filter frequency for channel 64 (first triangle wave channel):

Parameter Type: 1 (Filter Frequency)
Channel Index: 64
NRPN Address = $1000 + (1 \times 320) + 64 = 1384$

Example 3: Set attack time for channel 128 (first sine wave channel):

Parameter Type: 9 (Attack)
Channel Index: 128
NRPN Address = $1000 + (9 \times 320) + 128 = 4008$

Example 4: Set release time for channel 319 (last sawtooth channel):

Parameter Type: 12 (Release)
Channel Index: 319
NRPN Address = $1000 + (12 \times 320) + 319 = 5159$

Practical Example 5: Create layered pad with different attack times:

Assume a layered patch uses channels 128 (sine) and 256 (sawtooth) for a two-layer pad:

Layer 1 (Sine, channel 128): Fast attack for presence
NRPN #4008 value 1638 (Attack: 10ms, ~10% of range)

Layer 2 (Sawtooth, channel 256): Slow attack for swell
NRPN #4136 value 13107 (Attack: 2000ms, ~80% of range)

Result: Layered pad where the sine wave provides immediate presence whilst the sawtooth swells in slowly, creating rich, evolving texture.

NRPN Implementation Details

Value Normalisation

NRPN values (0-16383) are normalised to parameter ranges:

Boolean Parameters (on/off switches):

- **0-8191:** Off (0%)

- **8192-16383:** On (100%)
- **Threshold:** 8192 (50%)

Linear Parameters (most parameters):

- **0:** Minimum value (0%)
- **8192:** Midpoint (50%)
- **16383:** Maximum value (100%)
- **Mapping:** Linear interpolation

Logarithmic Parameters (time-based parameters):

- **0:** Minimum (e.g., 0.1ms attack)
- **16383:** Maximum (e.g., 100ms attack)
- **Mapping:** Exponential/logarithmic curve (musical scaling)

Bipolar Parameters ($\pm$ values):

- **0:** Maximum negative (-100%)
- **8192:** Centre (0%, neutral)
- **16383:** Maximum positive (+100%)

Performance Characteristics

Real-Time Performance:

- **Zero allocations:** No memory allocation during NRPN processing
- **Atomic operations:** Lock-free parameter updates (8ns per change)
- **Change detection:** Redundant updates eliminated (60-80% reduction)
- **Processing time:** <125ns average per NRPN message

Redundancy Filtering:

The Intuition Subsynth automatically filters redundant NRPN messages (unchanged values), reducing processing overhead by 60-80% when receiving NRPN streams from DAW automation or MIDI controllers.

Example: Sending NRPN #7 (Reverb Decay) value 10000 repeatedly:

- **First message:** Processed (value changed from default)
- **Subsequent messages:** Filtered (value unchanged, no processing)

Practical NRPN Programming Examples

Example 1: Dynamic Filter Sweep with Keyboard Tracking

Goal: Create a synth lead with dynamic filter sweeps that maintain consistent timbre across the keyboard.

Parameters:

- **Filter Cutoff:** Swept from 200Hz to 5kHz during attack
- **Keyboard Tracking:** 100% (filter follows note pitch)
- **Resonance:** High (Q=5.0 for vocal character)

NRPN Setup:

```
NRPN #250 value 16383  (Filter Keyboard Track: 100%)
NRPN #251 value 13107  (Filter Envelope Amount: +80%)
NRPN #252 value 0      (Self-Oscillation: off)
NRPN #253 value 8192   (Filter Drive: 100%, neutral)
```

Result: Filter sweeps from dark to bright during attack whilst maintaining consistent timbre across all notes (low notes have proportionally lower cutoff, high notes have proportionally higher cutoff).

Example 2: Vintage Chip Sound with Bit-Crushing

Goal: Authentic 1980s home computer sound (Commodore 64, ZX Spectrum).

Parameters:

- **Bit Depth:** 8-bit (vintage ADC character)
- **Sample Rate Reduction:** 4x (22.05kHz effective @ 44.1kHz)
- **Overdrive:** 30% (subtle analogue warmth)
- **Envelope:** Linear shape (authentic chip envelope)

NRPN Setup:

NRPN #9 value 8192 (Bit-Crush Bits: 8-bit, 50% of range)
NRPN #10 value 4096 (Sample Rate: 4x reduction, 25% of range)
NRPN #11 value 4916 (Overdrive Pre-Gain: 0.6, subtle warmth)

Result: Authentic retro digital sound with 8-bit quantisation noise, slight aliasing from sample rate reduction, and gentle analogue-style warmth.

Example 3: Stereo-Widened Pad with Individual Channel Panning

Goal: Ultra-wide stereo pad using per-channel panning to spread individual voices across the stereo field.

Technique: Pan each synthesis channel to a unique stereo position.

NRPN Setup (assuming pad uses channels 128-135, 8 voices):

Channel 128: Hard left
NRPN #3368 value 0 (Pan: hard left)

Channel 129: Left
NRPN #3369 value 4681 (Pan: 28.6% left)

Channel 130: Centre-left
NRPN #3370 value 6554 (Pan: 40% centre-left)

Channel 131: Centre
NRPN #3371 value 8192 (Pan: centre)

Channel 132: Centre-right
NRPN #3372 value 9830 (Pan: 60% centre-right)

Channel 133: Right
NRPN #3373 value 11703 (Pan: 71.4% right)

Channel 134: Hard right
NRPN #3374 value 16383 (Pan: hard right)

Channel 135: Centre (anchor)
NRPN #3375 value 8192 (Pan: centre)

Result: Massively wide stereo pad with voices evenly distributed across the stereo field, creating immersive, spatial ambience.

16. MIDI Implementation Chart

The following chart provides a standardised summary of the Intuition Subsynth's MIDI implementation for reference and compatibility checking.

MIDI Implementation Chart

--	--	--	--

Function	Transmitted	Recognised	Remarks
Basic Channel	N/A	1-16	All channels simultaneously
Default	N/A	1-16	No default channel
Mode	N/A	Mode 3	Poly, Omni Off (standard mode)
Note Number	N/A	0-127	Full range (12-120 audible)
Velocity	N/A	◦	Note On: 1-127 (exponential curve)
			Note Off: 0-127 (logged, not used)
Aftertouch	N/A	◦	Polyphonic (fully implemented)
	N/A	◦	Channel (fully implemented)
Pitch Bend	N/A	◦	14-bit, ± 2 semitones default (adjustable)
Control Change	N/A	◦	120+ controllers supported
CC#1 (Modulation)	N/A	◦	LFO depth, vibrato
CC#5 (Portamento Time)	N/A	◦	0-127ms glide time
CC#7 (Volume)	N/A	◦	Exponential curve
CC#10 (Pan)	N/A	◦	Constant-power panning
CC#11 (Expression)	N/A	◦	Real-time dynamics
CC#64 (Sustain)	N/A	◦	Piano-style sustain pedal
CC#65 (Portamento)	N/A	◦	On/off switch
CC#66 (Sostenuto)	N/A	◦	Selective sustain
CC#67 (Soft Pedal)	N/A	◦	Velocity scaling (70%)
CC#71 (Resonance)	N/A	◦	Filter Q factor
CC#74 (Cutoff)	N/A	◦	Filter frequency
CC#91 (Reverb)	N/A	◦	Reverb send level
CC#93 (Chorus)	N/A	◦	Chorus send level
CC#120 (All Sound Off)	N/A	◦	Emergency stop
CC#121 (Reset)	N/A	◦	Reset all controllers
CC#123 (All Notes Off)	N/A	◦	Musical termination
Program Change	N/A	◦	0-127 (128 GM programmes)
Bank Select (CC#0/32)	N/A	◦	Banks 0-4 (5 factory patchsets)
System Exclusive	N/A	◦	GM/GM2 System On, Master Volume/Tuning
System Common	N/A	◦	MTC Quarter Frame (recognised, not used)

Song Position	N/A	○	Recognised, not used
Song Select	N/A	○	Recognised, not used
Tune Request	N/A	×	Not supported
System Real-Time	N/A	○	All real-time messages recognised
Timing Clock	N/A	○	Tempo sync (future implementation)
Start/Stop/Continue	N/A	○	Recognised, logged
Active Sensing	N/A	○	Auto-disconnect if stopped
System Reset	N/A	○	Full reset to power-on state
RPN	N/A	○	Registered Parameter Numbers
RPN #0 (Pitch Bend)	N/A	○	±1 to ±24 semitones
RPN #1 (Fine Tuning)	N/A	○	±100 cents (14-bit)
RPN #2 (Coarse Tuning)	N/A	○	±64 semitones
NRPN	N/A	○	4360+ parameters (14-bit)
Named (0-299)	N/A	○	200+ global parameters
Per-Channel (1000-5159)	N/A	○	4160 per-channel parameters

Legend:

- : Yes (fully supported)
- ×: No (not supported)
- N/A**: Not applicable (receive-only synthesiser)

General MIDI Level 2 Compliance

The Intuition Subsynth is **fully compliant** with GM Level 2:

Requirement	Specification	Intuition Subsynth	Status
Polyphony (melodic)	32 voices minimum	320 voices	✓
Polyphony (percussion)	16 voices minimum	320 voices	✓
Channels	16 simultaneous	16 simultaneous	✓
Percussion channels	Channels 10 & 11	Channels 10 & 11	✓
Bank 0x78 rhythm	Dynamic conversion	Full support	✓
Melodic programmes	128 (1-128)	128 (per patchset)	✓
Drum sounds	47 sounds	47 sounds	✓
Controllers	Volume, Pan, etc.	120+ controllers	✓
Pitch Bend	±2 semitones	±2 semitones (default)	✓

Velocity response	Dynamic	Exponential curve	✓
Reverb/Chorus	Send levels	Full effects system	✓
SysEx support	GM/GM2 System On, Master Volume/Tuning	Full implementation	✓

Patchset Variations: Whilst GM-compliant, each patchset (SIDney, AYYMe, Rawland, PixelaTED, EenBeetje) provides unique sonic character for the 128 GM programmes.

Part V: Sound Banks

17. Factory Patchsets

The Intuition Subsynth includes **five factory patchsets**, each providing a complete General MIDI Level 1 instrument set (128 melodic programmes + 47 drum sounds) with unique sonic character. All patchsets are GM-compatible, ensuring that any GM MIDI file will play correctly whilst offering dramatically different timbres.

Overview of Factory Patchsets

Patchset	Inspiration	Character	Best For
SIDney	MOS 8580 SID (C64)	Analogue chip synthesis	Retro games, chip music, leads
AYYMe	AY-3-8910/YM2149 PSG (ZX Spectrum 128K/+2/+3, Amstrad CPC, MSX, Atari ST)	Sharp, metallic, lo-fi	8-bit music, chip tunes, retro FX
Rawland	Roland (303/808/909/etc.)	Professional analogue synths	Electronic music, techno, house, DnB, Grime
PixelaTED	Commodore TED 7360/8360 chip	Ultra lo-fi, gritty	Extreme retro, chip music
EenBeetje	ZX Spectrum 1-bit beeper	Minimal, 1-bit synthesis	Beeper music, minimalist

Switching Patchsets: Patchsets can be loaded at startup via command-line argument (`--patchset=NAME`) or switched in real-time via MIDI Bank Select (CC#0/32, Banks 0-4). All patchsets share the same GM programme numbers, enabling instant sonic transformation of existing MIDI arrangements. Bank switching is **global** and affects all 16 MIDI channels simultaneously.

SIDney: MOS 8580 SID Chip Emulation

Historical Context:

The MOS Technology 6581/8580 **Sound Interface Device (SID)** was the audio chip in the Commodore 64 home computer (1982-1994). Designed by Bob Yannes (later co-founder of Ensoniq), the SID was revolutionary for consumer hardware, featuring:

- 3-voice analogue synthesis (not samples)
- Multiple waveforms per voice (sawtooth, triangle, pulse, noise)
- Hardware-based filters (24dB/octave resonant low-pass/high-pass/band-pass)
- ADSR envelopes per voice

- Hard sync and ring modulation

The SID became legendary in computer music history, with composers like Rob Hubbard, Martin Galway, and Ben Daglish creating iconic soundtracks that defined 1980s video game audio.

SIDney Characteristics:

The **SIDney** patchset emulates the **MOS 8580 R5** (later PAL revision, 1987+) character:

- **Waveform purity:** Clean, precise waveform generation (less distortion than 6581)
- **Filter character:** Smooth, musical filtering (less resonance "scream" than 6581)
- **Pitch accuracy:** Stable, accurate tuning (8580 had better temperature stability)
- **Envelope shapes:** Mix of linear (authentic chip) and exponential (musical enhancements)
- **Pitch bend range:** ±12 semitones default (authentic C64 pitch slides)

Selected Instrument Highlights (Key Programmes):

Prog #	Name	Character	Notes
1	Acoustic Grand	Saw-based piano with decay envelope	Plucky, bright
25	Acoustic Guitar (nylon)	Triangle+noise blend, percussive	Authentic C64 guitar
33	Acoustic Bass	Triangle wave, deep fundamental	Classic SID bass
39	Synth Bass 1	Sawtooth+square, filter sweep	Moog-style SID bass
49	String Ensemble 1	Layered sawtooth, slow attack	Lush, analogue strings
81	Lead 1 (square)	Pulse wave, PWM, fast attack	Classic chip lead
82	Lead 2 (sawtooth)	Sawtooth+sync, aggressive	Screaming lead
89	Pad 2 (warm)	Layered saw+tri, reverb	Atmospheric pad
104	Sitar	Ring mod+noise, exotic tuning	Experimental synthesis

Drum Kit Highlights (Channel 10):

Note	Drum Sound	Character	Implementation
36	Kick Drum (C1)	Deep sine+noise, pitch envelope	Punchy, electronic
38	Snare Drum	Noise burst, sharp filter	Crisp, tight
42	Closed Hi-Hat	White noise, short decay	Sharp, metallic
46	Open Hi-Hat	White noise, medium decay, filter sweep	Ringling, bright
49	Crash Cymbal	Metallic noise, long decay	Shimmering, synthetic

Musical Applications:

- **Retro video game soundtracks** (1980s computer game nostalgia)
- **Chip music** (authentic Commodore 64 character)
- **Synth leads and bass** (classic analogue chip synthesis)
- **Experimental electronic music** (ring mod, sync, noise textures)

Performance Tips:

- **Use pitch bend extensively** (± 12 semitones enables C64-style pitch slides)
- **Experiment with filter resonance** (CC#71) for classic SID "screams"
- **Layer programmes** to exceed original SID's 3-voice limit
- **Combine with reverb** (CC#91) to add spatial depth to dry chip sounds

AYYMe: AY-3-8910 / YM2149 PSG Chip Emulation

Historical Context:

The **General Instrument AY-3-8910** (1978) and its Yamaha-licensed variant **YM2149** were ubiquitous sound chips in 1980s home computers and arcade machines:

- **Amstrad CPC** (AY-3-8912, 8-bit home computer, 1984)
- **ZX Spectrum 128K** (AY-3-8912, Sinclair's 128K model, 1985)
- **Atari ST** (YM2149, 16-bit home computer, 1985)
- **MSX** computers (AY-3-8910/PSG, 1983+)
- Countless arcade machines (Pac-Man, Frogger, Gyruss, etc.)

The AY-3-8910 was simpler than the SID:

- 3-voice square wave synthesis (no waveform selection)
- 16 volume levels per voice (4-bit amplitude)
- Simple envelope generator (8 preset shapes)
- Noise generator (pseudo-random, shared across voices)
- No filter

Despite its limitations, the AY-3-8910 produced iconic sounds that defined early video game audio.

AYYMe Characteristics:

The **AYYMe** patchset emulates authentic AY-3-8910/YM2149 character:

- **Square wave dominance:** Heavy use of pulse/square waveforms (authentic to chip)
- **Metallic timbre:** Sharp, bright, slightly harsh (characteristic of 4-bit DAC)
- **Limited dynamic range:** Restricted velocity response (16-step amplitude)
- **Simple envelopes:** Mostly linear shapes (authentic chip envelopes)
- **Lo-fi noise:** Periodic noise (LFSR-based, authentic algorithm)

Selected Instrument Highlights:

Prog #	Name	Character	Notes
1	Acoustic Grand	Square wave, percussive envelope	Plucky, metallic piano
25	Acoustic Guitar	Pulse+noise, fast attack	Percussive, lo-fi
33	Acoustic Bass	Square wave, short sustain	Punchy, limited sustain
39	Synth Bass 1	Square+noise blend	Gritty, lo-fi bass
49	String Ensemble	Layered squares, slow attack	Buzzy, harsh strings
81	Lead 1 (square)	Pure square wave, fast attack	Classic chip lead
89	Pad 2 (warm)	Detun Layers of square waves, slow attack, chorus	Bright, shimmering pad
104	Sitar	Square+noise+ring mod	Exotic, lo-fi

115	Steel Drums	Square wave, pitch envelope	Metallic, bell-like
-----	-------------	-----------------------------	---------------------

Drum Kit Highlights:

Note	Drum Sound	Character	Implementation
36	Kick Drum	Pitched noise burst	Lo-fi, crunchy
38	Snare Drum	White noise, short	Sharp, thin
42	Closed Hi-Hat	Periodic noise, very short	Metallic, lo-fi
46	Open Hi-Hat	Periodic noise, medium decay	Ringling, harsh
49	Crash Cymbal	White noise, aliased	Gritty, lo-fi

Musical Applications:

- **8-bit video game music** (Atari ST, Amstrad CPC, ZX Spectrum 128K soundtracks)
- **Chip tunes** (authentic AY-3-8910 character)
- **Retro arcade sounds** (Pac-Man, Frogger-style audio)
- **Lo-fi electronic music** (harsh, metallic textures)

Performance Tips:

- **Embrace the lo-fi character** (don't expect smooth, polished sounds)
- **Use noise creatively** (the AY chip's noise is musical and characterful)
- **Keep arrangements sparse** (the chip's 3-voice limit inspired minimalist composition)
- **Experiment with detuning** (layer slightly detuned square waves for width)

Rawland: Roland-Inspired Analogue Synthesis

Historical Context:

Roland Corporation defined professional electronic music synthesis from the 1970s through the 1990s with iconic instruments:

- **Roland TB-303** (1981): Bass synthesiser, acid house foundation
- **Roland TR-808** (1980): Drum machine, hip-hop/electronic music staple
- **Roland TR-909** (1983): Drum machine, techno/house standard
- **Roland TR-727** (1985): Latin percussion drum machine
- **Roland Juno-106** (1984): Polyphonic analogue synthesiser
- **Roland Jupiter-8** (1981): Flagship polyphonic synthesiser
- **Roland SH-101** (1982): Monophonic bass synthesiser

Roland's sound characterised electronic music genres: Acid House, Detroit Techno, Chicago House, Hip-Hop, Hardcore, Jungle/DnB and beyond.

Rawland Characteristics:

The **Rawland** patchset draws inspiration from Roland's classic analogue synthesisers whilst leveraging the Intuition Subsynth's 320-channel architecture:

- **Warm analogue character:** Smooth, musical waveforms with subtle non-linearity
- **Resonant filters:** Aggressive filter sweeps (TB-303-style acid bass)
- **Punchy drums:** Synthesised drums (TR-808/909-style kick, snare, hats)

- **Lush pads:** Multi-layered synthesis (Juno/Jupiter-style strings and pads)
- **Expressive envelopes:** Exponential curves for natural attack/release

Selected Instrument Highlights:

Prog #	Name	Character	Notes
1	Acoustic Grand	Layered saw+sine, percussive	Full, warm piano
25	Acoustic Guitar	Plucked sawtooth, body resonance (comb filter)	Realistic articulation
33	Acoustic Bass	Sawtooth, filter envelope	Deep, resonant bass
39	Synth Bass 1	TB-303-style acid bass (saw+filter+resonance)	Screaming, squelchy
49	String Ensemble	Juno-style strings (layered saws, chorus)	Lush, wide, warm
81	Lead 1 (square)	PWM square, filter sweep	Expressive, vocal-like
82	Lead 2 (sawtooth)	Sawtooth+sync, aggressive filter	Cutting, powerful lead
89	Pad 2 (warm)	Jupiter-8-style pad (layered saws, slow attack)	Atmospheric, evolving
104	Sitar	Ring mod+filter, exotic timbre	Experimental, unique

Drum Kit Highlights (TR-808/909-inspired):

Note	Drum Sound	Character	Implementation
36	Kick Drum (BD)	TR-808/909 hybrid (sine+noise, pitch envelope)	Deep, punchy, boomy
38	Snare Drum (SD)	TR-909-style (noise+tone, snap)	Crisp, electronic
42	Closed Hi-Hat (CH)	TR-808/909 metallic (noise, short)	Tight, crisp
46	Open Hi-Hat (OH)	TR-808/909 metallic (noise, decay)	Ringling, shimmering
49	Crash Cymbal	TR-909 crash (filtered noise, long decay)	Bright, sizzling
51	Ride Cymbal	Metallic synthesis (filtered noise)	Bell-like, clear

Musical Applications:

- **Acid house and techno** (TB-303-style bass lines)
- **Electronic dance music** (TR-808/909 drums)
- **Ambient and downtempo** (lush pads and strings)
- **Synthwave and retrowave** (1980s analogue synthesis nostalgia)

Performance Tips:

- **Automate filter cutoff** (CC#74) for classic acid bass sweeps
- **Use high filter resonance** (CC#71 = 100-127) for screaming TB-303 character
- **Layer programmes** for thick, professional textures
- **Apply moderate reverb** (CC#91 = 40-70) for depth without washing out transients

PixelaTED: Commodore TED 7360/8360 Chip Emulation

Historical Context:

The **Commodore TED 7360/8360** (Text Editing Device) chip powered the Commodore 16, Commodore 116, and Commodore Plus/4 home computers (1984-1986). The TED integrated video, sound, and I/O into a single chip to reduce cost, but its audio capabilities were severely limited compared to the SID:

- **2 voices** (vs SID's 3)
- **Channel 1:** Square wave only
- **Channel 2:** Square wave OR white noise (switchable)
- **Volume control:** 9 levels (0-8) for overall volume only, no per-channel volume
- **Frequency range:** 100 Hz to 23 kHz with 10-bit frequency control per channel
- **No filters** (vs SID's resonant filters)
- **No ADSR envelopes** (simple on/off gates only)
- **No ring mod or sync** (vs SID's modulation capabilities)

Despite these severe limitations, the TED produced charming, lo-fi sounds that defined the Commodore 16 & Plus/4's distinctive audio character. TED music required creative composition to overcome the hardware's constraints, with some composers achieving remarkably complex results through clever programming techniques.

PixelaTED Characteristics:

The **PixelaTED** patchset emulates authentic Commodore TED character with extreme lo-fi aesthetic:

- **Square wave dominance:** Almost all programmes use square/pulse waveforms (authentic to channel 1)
- **White noise available:** Channel 2 can produce white noise for percussion and effects
- **Minimal envelopes:** Mostly linear shapes (100% linear in analysis)
- **Limited polyphony simulation:** Sparse, minimalist textures (emulating 2-voice limitation)
- **No filter effects:** Dry, direct sounds (authentic to hardware)
- **9-level volume:** Global volume control only (0-8), no per-channel dynamics
- **Extreme lo-fi:** Embraces limitations as sonic character

Selected Instrument Highlights:

Prog #	Name	Character	Notes
1	Acoustic Grand	Square wave, percussive gate	Extremely lo-fi piano
25	Acoustic Guitar	Pulse wave, fast gate	Plucky, minimal
33	Acoustic Bass	Square wave, short gate	Punchy, limited sustain
39	Synth Bass 1	Square wave, medium gate	Simple, direct bass
49	String Ensemble	Layered squares, slow gate	Buzzy, harsh strings
81	Lead 1 (square)	Pure square wave, fast gate	Minimal chip lead
89	Pad 2 (warm)	Layered squares, slow gate	Gritty, lo-fi pad
104	Sitar	Square+PWM, fast gate	Exotic, minimal

Drum Kit Highlights:

Note	Drum Sound	Character	Implementation
36	Kick Drum	Pitched square wave burst	Tonal, lo-fi
38	Snare Drum	White noise burst (channel 2)	Sharp, minimal noise

42	Closed Hi-Hat	White noise, short (channel 2)	Crisp, lo-fi
46	Open Hi-Hat	White noise, longer decay	Ringling, noisy

Musical Applications:

- **Extreme retro/lo-fi music** (Commodore 16/Plus/4 emulation)
- **Minimal chip music** (embracing severe hardware limitations)
- **Experimental/noise** (harsh, gritty textures)
- **Nostalgic/vintage computing** (authentic TED character)

Performance Tips:

- **Keep arrangements minimal** (TED had only 2 voices; emulate that limitation)
- **Embrace the harshness** (don't fight the lo-fi character)
- **Use sparse melodies** (TED music was minimalist by necessity)
- **Avoid heavy effects** (authentic TED sound is dry and direct)

EenBeetje: ZX Spectrum 1-Bit Beeper Emulation

Historical Context:

The **Sinclair ZX Spectrum** (1982-1992) was a hugely influential 8-bit home computer with no dedicated sound chip. Audio was produced by toggling a single I/O bit connected to a speaker, creating 1-bit audio:

- **1-bit output:** Speaker could only be on or off (no volume control)
- **Software-generated sound:** CPU had to toggle the speaker bit rapidly to create tones
- **Polyphony through interleaving:** Multiple voices by rapidly switching between them (time-division multiplexing)
- **Beeper music:** Genre of music created specifically for the ZX Spectrum's constraints

Beeper music composers developed sophisticated techniques to overcome the hardware's extreme limitations, creating surprisingly complex musical arrangements. Notable composers included Tim Follin, who pushed the beeper to its absolute limits.

EenBeetje Characteristics:

The **EenBeetje** patchset (Dutch: "a bit" / "a little") emulates authentic ZX Spectrum 1-bit beeper character:

- **Square wave basis:** All sounds based on square waves (1-bit on/off)
- **Linear envelopes:** Simple on/off gates (119 out of 128 programmes use linear)
- **Minimal sustain:** Short, percussive sounds (emulating CPU time limitations)
- **Sparse textures:** Simple, direct synthesis (limited by 1-bit nature)
- **Retro lo-fi aesthetic:** Embraces constraints as defining character

Selected Instrument Highlights:

Prog #	Name	Character	Notes
1	Acoustic Grand	Square wave, percussive	1-bit piano simulation
25	Acoustic Guitar	Square wave, short gate	Plucky, minimal
33	Acoustic Bass	Low square wave, percussive	Deep, simple bass
39	Synth Bass 1	Square wave, short decay	Punchy, direct

49	String Ensemble	Square waves, slow attack	Buzzy, harsh strings
81	Lead 1 (square)	Pure square wave	Classic beeper lead
89	Pad 2 (warm)	Layered squares	Gritty, lo-fi pad

Drum Kit Highlights:

Note	Drum Sound	Character	Implementation
36	Kick Drum	Pitched square click	Simple, percussive
38	Snare Drum	High square click	Sharp, minimal
42	Closed Hi-Hat	Very high square pulse	Minimal, clicky
46	Open Hi-Hat	Very high square, longer	Simple, beeper-style

Musical Applications:

- **ZX Spectrum beeper music emulation** (authentic 1-bit character)
- **Extreme minimalism** (most limited of all patchsets)
- **Nostalgic retro gaming** (1982-1992 ZX Spectrum era)
- **Experimental lo-fi** (1-bit aesthetic exploration)

Performance Tips:

- **Keep everything minimal** (the beeper had severe CPU/time limitations)
- **Use sparse arrangements** (beeper music was often monophonic or 2-voice max)
- **Embrace the 1-bit character** (don't expect smooth or polished sounds)
- **Study classic beeper music** (Tim Follin, David Whittaker for inspiration)

Part VI: Reference

18. Technical Specifications

Synthesis Architecture

Specification	Value
Total Synthesis Channels	320 independent channels
Channel Organization	64 channels per waveform type (5 types)
Waveform Types	Square, Triangle, Sine, Noise, Sawtooth
Polyphony	Wave-type-based (64 per type, 320 total)
Voice Allocation	Intelligent priority-based stealing
Layered Patches	Dual-waveform support (2 channels per note)

Waveform Generation

Waveform	Channels	Features
Square	0-63	PWM (5%-95% duty cycle), SIMD-optimised

Triangle	64-127	Pure triangle wave, phase-accurate
Sine	128-191	Pure sine wave, high-precision
Noise	192-255	White/periodic/metallic, LFSR-based
Sawtooth	256-319	PolyBLEP anti-aliasing, SIMD-optimised

Audio Quality

Specification	Value
Sample Rates	44.1kHz, 48kHz, 96kHz, 192kHz (runtime adaptive)
Bit Depth	32-bit float internal, 16-24-bit integer output
Dynamic Range	>100dB (theoretical), >90dB (measured)
THD+N	<-60dB @ 1kHz, 0dBFS
Frequency Response	20Hz - 20kHz (± 0.5 dB)
Aliasing Rejection	>-80dB (sawtooth PolyBLEP)
Latency (JACK)	2.9ms @ 44.1kHz, 64-sample buffer
Latency (Oto)	3-10ms configurable

Synthesis Parameters

Envelopes:

- **Attack:** 0-11,700ms (0-11.7 seconds), sample-accurate
- **Decay:** 0-2,000ms (2 seconds)
- **Sustain:** 0-100% level
- **Release:** 0-15,000ms (0-15 seconds)
- **Envelope Shapes:** Linear, Exponential, Logarithmic

Filters:

- **Types:** Low-Pass, High-Pass, Band-Pass
- **Cutoff Range:** 20Hz - 20kHz (exponential mapping)
- **Resonance:** Q=0.5 to Q=10.0+ (self-oscillation)
- **Filter Modes:** State-variable (SVF), 2-pole (12dB/oct)

Modulators:

- **LFO Rate:** 0.01Hz - 20Hz
- **LFO Waveforms:** Sine, Triangle, Square, Sawtooth, Random (S&H)
- **LFO Destinations:** Pitch, Filter, Amplitude, Pan, PWM
- **Hard Sync:** Master/slave oscillator phase reset
- **Ring Modulation:** Four-quadrant multiplication

Effects System

Per-Channel Effects (9 types, 320 channels):

- **Bit Crusher:** 1-16 bits, 1-64x sample rate reduction

- Comb Filter: 0.1-50ms delay, ±99% feedback
- Delay Line: 0-92.8ms @ 44.1kHz, 0-100% feedback
- Flanger: 0.01-10Hz rate, 0-100% depth, ±99% feedback
- Phaser: 1-8 allpass stages, 0.01-10Hz rate
- Overdrive/Distortion: Soft clipping, 0-100% drive
- Ring Modulation: Inharmonic synthesis
- Hard Sync: Harmonic synthesis
- PWM: 5%-95% pulse width modulation

Master Effects (5 types, global):

- Reverb: Schroeder algorithm (4 comb + 2 allpass), 8ms pre-delay
- Chorus: Stereo LFO modulation, phase-offset L/R
- Ping-Pong Delay: 1-2000ms, stereo bounce, tape flutter
- Compressor: RMS detection, lookahead, parallel compression
- Limiter: Soft limiting at ±0.95 (-0.4dBFS), automatic

MIDI Implementation

Specification	Value
MIDI Channels	16 simultaneous
MIDI Messages	Note On/Off, CC, Program Change, Pitch Bend, SysEx
Standard Controllers	120+ MIDI CC messages
NRPN Parameters	4360+ (14-bit resolution)
Pitch Bend Range	±1 to ±24 semitones (default ±2)
GM Level 2 Compliance	Full compliance (see Appendix A for complete feature matrix)
Velocity Response	Exponential curve (1-127)
Aftertouch	Polyphonic & Channel (fully implemented, Oct 2025)

Hardware Specifications

Component	Specification
Supported Boards	Raspberry Pi 3B/B+, 4B, 400, 5, 500, Zero 2 W (64-bit)
Storage	MicroSD card (minimum 8GB recommended)
Power Supply	5V DC (Pi 4/400/5/500: USB-C 3A; Pi 3/Zero 2 W: micro-USB 2.5A)
Operating Temperature	0°C to 35°C (32°F to 95°F)
Dimensions	Matches selected Raspberry Pi form factor (keyboard/credit card)

Per-Board CPU & RAM Specifications:

Board	SoC	CPU	Stock Frequency	Subsynth Frequency	RAM
Pi 3 B	BCM2837	Quad-core Cortex-A53	1.2GHz	1.2GHz	1GB LPDDR2

		(ARMv8)			
Pi 3 B+	BCM2837B0	Quad-core Cortex-A53 (ARMv8)	1.4GHz	1.4GHz	1GB LPDDR2
Pi 4	BCM2711	Quad-core Cortex-A72 (ARMv8)	1.5-1.8GHz	1.8GHz	2/4/8GB LPDDR4
Pi 400	BCM2711	Quad-core Cortex-A72 (ARMv8)	1.8GHz	2.0GHz (OC)	4GB LPDDR4
Pi 5	BCM2712	Quad-core Cortex-A76 (ARMv8.2)	2.4GHz	2.4GHz (locked)	4/8/16GB LPDDR4X
Pi 500	BCM2712	Quad-core Cortex-A76 (ARMv8.2)	2.4GHz	2.4GHz (locked)	4/8GB LPDDR4X
Zero 2 W	BCM2710A1 (RP3A0 SiP)	Quad-core Cortex-A53 (ARMv8)	1.0GHz	1.2GHz (OC)	512MB LPDDR2

Notes:

- OC = Overclocked via `config.txt` for maximum synthesis performance
- Frequency locking prevents dynamic scaling (ensures consistent real-time performance)
- All boards run 64-bit Raspberry Pi OS (Debian Bookworm)

I/O Ports:

- **MIDI IN:** Optional 5-pin DIN interface (opto-isolated, 31,250 baud)
- **USB-MIDI:** USB 3.0/2.0 Type-A ports for class-compliant controllers
- **Audio Output:** HiFiBerry DAC2 Pro stereo RCA (L/R), 44.1-192kHz
- **USB Power:** USB-C PD input, 5V/3A minimum
- **Network:** On-board Ethernet/Wi-Fi hardware disabled (air-gapped operation)

Performance Characteristics

Metric	Value
CPU Usage	<1% @ 48kHz, 320 active channels (Pi 5/500); ~50% on Pi 3/Zero 2 W
Memory Usage	~150MB RAM (synthesis engine)
Real-Time Margin	10.1× (AMD64), 6.7× (Pi 5/500 ARM64), 3.2× (Pi 3/Zero 2 W ARM64)
Voice Allocation	O(1) with atomic bit-fields
NRPN Processing	<125ns average (lock-free, zero allocation)
Active Channel Cache	70% reduction in scanning overhead

Software Architecture

Component	Implementation
Language	Go 1.25+ (Green Tea GC, JSONv2)
Audio Backend (Default)	JACK (sample-accurate, 2.9ms latency)

Audio Backend (Legacy)	Oto v3 (cross-platform, 3-10ms latency)
MIDI Backend (Default)	JACK MIDI (sample-accurate, unified client)
MIDI Backend (Legacy)	PortMIDI (500µs polling, cross-platform)
SIMD Optimisation	AVX2/FMA3 (x86-64-v3), NEON (ARM64), Scalar
Memory Bus	Lock-free atomic (239.65× speedup vs mutex)
Block Processing	64 samples default (32-128 adjustable)

Performance Optimisation & Real-Time Features

Professional audio synthesis demands deterministic, low-latency performance with zero dropouts. Intuition Subsynth achieves this through a comprehensive stack of optimisations spanning the operating system kernel, compiler toolchain, Go runtime, DSP engine, and hardware configuration. This section explains the technologies that enable studio-grade real-time performance on Raspberry Pi hardware.

Why Real-Time Performance Matters

In professional audio production, **timing is everything**. A single missed deadline in the audio callback can cause audible clicks, pops, or dropouts. Intuition Subsynth is engineered to meet strict real-time guarantees:

- **Deterministic latency:** Every audio frame must be generated within a fixed time budget (e.g., 2.9ms for 64 samples @ 48kHz)
- **Zero dropouts:** The synthesis engine must never miss an audio callback, even under system load
- **Sample-accurate MIDI:** Note timing must be precise to the sample level, not quantised to large buffers

Achieving this requires coordination between all layers of the system: the OS kernel must provide low-latency scheduling, the compiler must generate efficient machine code, the runtime must minimise garbage collection pauses, and the DSP code must use SIMD acceleration to maximise throughput.

Operating System & Kernel Optimisations

1. PREEMPT_RT Real-Time Kernel (6.12.48-rt)

- **What:** A fully preemptible Linux kernel that converts all kernel code paths to real-time schedulable tasks
- **Why:** Standard Linux kernels can delay audio threads by 10-100ms during I/O or memory allocation. PREEMPT_RT ensures the audio thread always runs within microseconds of waking up
- **Impact:** Reduces worst-case latency from 10-100ms to <1ms, enabling 32-64 sample JACK buffers (0.7-1.3ms @ 48kHz)

2. CPU Core Isolation (iso1cpus=2, 3)

- **What:** Reserves CPU cores 2 and 3 exclusively for real-time audio threads, preventing the Linux scheduler from assigning non-audio tasks to these cores
- **Why:** Sharing cores with background processes (file indexing, network daemons, etc.) causes cache pollution and unpredictable latency spikes
- **Impact:** Guarantees dedicated compute resources for synthesis, improving worst-case jitter from 100-1000µs to <10µs

3. Tickless Operation (nohz_full=2, 3)

- **What:** Disables the periodic 1000Hz timer tick on isolated cores, allowing them to run uninterrupted
- **Why:** Even a 1ms timer interrupt can cause 10-50µs jitter in audio processing. Eliminating the tick removes this source of latency
- **Impact:** Reduces interrupt overhead on audio cores by 95%, freeing CPU cycles for synthesis

4. RCU Callback Offloading (`rcu_nocbs=2, 3`)

- **What:** Moves Read-Copy-Update (RCU) grace period processing from audio cores to housekeeping cores
- **Why:** RCU is a Linux kernel synchronisation primitive that can delay threads by hundreds of microseconds. Offloading it prevents audio stalls
- **Impact:** Eliminates unpredictable 100-500µs delays caused by RCU grace periods

5. Threaded Interrupts (`threadirqs`)

- **What:** Makes hardware interrupts preemptible by converting them to kernel threads
- **Why:** Hard IRQs (e.g., network, disk) cannot be preempted and can delay audio threads. Threaded IRQs allow the RT scheduler to prioritise audio over hardware events
- **Impact:** Prevents interrupt storms from blocking audio processing

6. CPU Performance Governor

- **What:** Locks CPU frequency scaling to maximum performance mode, preventing dynamic frequency reduction
- **Why:** CPU frequency transitions (e.g., 600MHz → 2.4GHz) take 10-50ms, causing massive latency spikes during the transition
- **Impact:** Eliminates frequency scaling latency, ensuring consistent sub-millisecond audio callback timing

7. rtkit (Real-Time Priority Management)

- **What:** A D-Bus service that grants real-time scheduling priorities to audio applications without requiring root privileges
- **Why:** Real-time threads need `SCHED_FIFO` or `SCHED_RR` scheduling, which normally requires root. rtkit provides safe, controlled access
- **Impact:** Enables JACK to run with RT priority (priority 10-20) as a normal user, ensuring audio callbacks preempt all non-RT threads

8. systemd RT I/O Scheduling

- **What:** Configures I/O scheduling policies to prioritise real-time processes over bulk I/O operations
- **Why:** Disk/network I/O can saturate memory bandwidth and delay audio threads. RT I/O scheduling ensures audio gets first access to memory
- **Impact:** Reduces memory contention during background I/O operations

Compiler & Build Optimisations

9. Link-Time Optimisation (LTO)

- **What:** Whole-programme optimisation that enables cross-file inlining and dead code elimination
- **Why:** Traditional compilation optimises each source file independently. LTO sees the entire programme and can inline hot paths across files, eliminating function call overhead
- **Impact:** 5-15% performance improvement in DSP hot paths, reduces binary size by 10-20%

10. Profile-Guided Optimisation (PGO)

- **What:** Compiler optimisation using runtime profiling data to guide code layout and inlining decisions
- **Why:** The compiler generates a test binary, profiles it under realistic workloads, then recompiles with hot paths optimised for L-cache locality
- **Impact:** 2-14% performance improvement (baseline 2-5%, hot paths 10-14%), particularly in MIDI message dispatch and voice allocation

11. Architecture-Specific Tuning

- **What:** Compiler flags that target specific CPU microarchitectures (Cortex-A53, Cortex-A72, Cortex-A76, x86-64-v3)

- **Why:** Generic ARM64/x86-64 code cannot use advanced instructions (e.g., FMA3, NEON multiply-accumulate). Tuning enables optimal instruction selection
- **Impact:** 10-25% speedup in filter processing (FMA3 fused multiply-add), 15-30% in waveform generation (NEON/AVX2 SIMD)

Specific Tuning by Platform:

- **Pi Zero 2W** (Cortex-A53, 1.2GHz): `-mcpu=cortex-a53 -mtune=cortex-a53 + conservative -Os` size optimisation for limited L1 cache
- **Pi 400** (Cortex-A72, 2.0GHz): `-mcpu=cortex-a72 -mtune=cortex-a72`
- **Pi 5/500** (Cortex-A76, 2.4GHz): `-mcpu=cortex-a76 -mtune=cortex-a76`
- **x86-64 Development** (AVX2): `-march=x86-64-v3 -mtune=generic` (requires AVX2, FMA3, BMI2)

12. Fast-Math Optimisations

- **What:** Relaxes IEEE 754 floating-point compliance to enable aggressive algebraic transformations (e.g., associative reordering, reciprocal approximation)
- **Why:** Strict IEEE 754 prevents optimisations like $a/b/c \rightarrow a/(b*c)$ or $\text{sqrt}(x) \rightarrow x * \text{rsqrt}(x)$. Fast-math allows these transformations for 2-3× speedup in math-heavy code
- **Impact:** 25-40% speedup in filter coefficient calculations, envelope exponentials, and LFO sine approximations
- **Safety:** Audio DSP tolerates 0.001% error, unlike scientific computing. Fast-math is safe for synthesis

13. Function Alignment (64-byte cache lines)

- **What:** Aligns hot functions to 64-byte boundaries to prevent cache line splits
- **Why:** Modern CPUs fetch instructions in 64-byte cache lines. Functions split across lines waste I-cache bandwidth. Alignment ensures hot loops fit in single cache lines
- **Impact:** 3-8% reduction in I-cache misses in tight DSP loops (waveform generation, mixing)

Go Runtime Tuning

14. Green Tea Garbage Collector

- **What:** Experimental Go GC featuring reduced pause times through concurrent mark-sweep improvements
- **Why:** Standard Go GC pauses can reach 1-5ms during collections, causing audio dropouts. Green Tea targets <100µs pauses
- **Impact:** 60-80% reduction in GC pause times, enabling 64-sample buffers (1.3ms @ 48kHz) without dropouts

15. GOGC=2000 (Heap Growth Factor)

- **What:** Configures the GC to trigger only when heap usage grows 20× (2000%), rather than the default 100%
- **Why:** More frequent GC means more pauses. Trading memory for latency by allowing 20× growth reduces GC frequency by 10-20×
- **Impact:** Reduces GC frequency from every 2-5 seconds to every 30-60 seconds, virtually eliminating GC pauses during performance

16. Architecture-Aware Memory Limits

- **What:** Platform-specific memory limits that prevent excessive heap growth on constrained devices
- **Why:** Pi Zero 2W has only 512MB RAM (256MB available to userspace). Allowing unlimited heap growth causes OOM kills. Generous limits (3.5GB) on Pi 5/500 enable optimal performance
- **Impact:** Prevents OOM crashes on Pi Zero 2W whilst allowing Pi 5/500 to use full RAM for larger patch libraries

Memory Limits by Platform:

- **Pi Zero 2W** (512MB total): 256MB limit (leaves room for kernel, services)
- **Pi 400** (4GB total): 3.5GB limit (generous for synthesis workloads)

- **Pi 5/500** (4-8GB total): 3.5GB limit (synthesis rarely exceeds 500MB)
- **x86-64** (8GB+ typical): 3.5GB limit (development machines)

17. JSONv2 Experimental Feature

- **What:** Next-generation JSON encoding/decoding with improved performance and reduced allocations
- **Why:** Patch loading uses JSON for 128 melodic + 47 drum patches. Standard `encoding/json` allocates heavily. JSONv2 reduces allocations by 40-60%
- **Impact:** 30-50% faster patch loading, reduces GC pressure during startup

DSP & SIMD Acceleration

18. AVX2 SIMD Vectorisation (x86-64)

- **What:** 256-bit vector instructions processing 8 float32 values simultaneously
- **Why:** Scalar code processes one sample at a time. AVX2 processes 8 samples in parallel using the same instruction
- **Impact:** 180-240% speedup in waveform generation, 150-200% in stereo mixing, 120-160% in filter processing

19. NEON SIMD Vectorisation (ARM64)

- **What:** 128-bit vector instructions processing 4 float32 values simultaneously
- **Why:** ARM scalar code is efficient, but NEON quadruples throughput for embarrassingly parallel DSP (mixing, waveform generation)
- **Impact:** 200-300% speedup in sawtooth generation (PolyBLEP), 150-200% in mixing, 100-150% in envelope processing

Device-Specific NEON Tuning:

- **Pi Zero 2W:** Conservative NEON (4-wide) to avoid thermal throttling on the 512MB board
- **Pi 400/5/500:** Aggressive NEON with unrolled loops for maximum throughput

Raspberry Pi Zero 2 W Feature Restrictions

The **Pi Zero 2 W** is supported but operates with reduced capabilities due to hardware limitations (512MB RAM, compact thermal design, limited power budget). All other supported Pi models (Pi 2B Rev 1.2, Pi 3 series, Pi 4, Pi 400, Pi 5, Pi 500) use the full 320-channel configuration.

Restricted Configuration:

Parameter	Pi Zero 2 W	All Other Models
Total Channels	48 (15% of full)	320 (100%)
Polyphony per MIDI Channel	4 voices	8 voices
Sample Rate	22.05kHz	48kHz (JACK default)
Block Size	128 samples	64 samples
Worker Threads	2 cores	All available cores
Heap Limit	256MB	3.5GB

What This Means in Practice:

- **Simple MIDI files work perfectly:** Single-instrument melodies, small ensembles, basic drum patterns
- **Complex arrangements may voice-steal:** Layered orchestral pieces, dense polyphony, heavily layered patches
- **Recommended usage:** Educational projects, portable setups, budget-conscious applications

- **Sound quality:** Identical synthesis algorithms and effects, just fewer simultaneous voices
- **All effects available:** Full reverb, chorus, delay, compression, and per-channel effects

Optimisation Recommendations:

1. Use **single-layer patches** instead of layered patches (saves channels)
2. Keep **total simultaneous notes below 32** for stable performance
3. Choose **simpler patchsets** (SIDney recommended for CPU efficiency)
4. Avoid **dense chord voicings** in lower registers (use inversions/voice leading)
5. Use **external MIDI sequencer** instead of playing live complex parts
6. Monitor **CPU usage** via HDMI display to avoid thermal throttling

Note: The **Pi 3A+** also has 512MB RAM but does **NOT** have these restrictions—it uses the full 320-channel configuration due to better thermal management (larger PCB area, better heat dissipation) compared to the Zero 2 W's compact form factor.

20. FTZ/DAZ (Flush-To-Zero / Denormals-Are-Zero)

- **What:** x86 MXCSR control bits that flush subnormal (denormal) floating-point values to zero
- **Why:** Denormals (numbers $<1.2 \times 10^{-38}$) trigger microcode-assisted slow paths, causing 10-100× slowdowns. Filters and reverbs generate denormals during decay
- **Impact:** Prevents denormal stalls that cause 10-100ms dropouts in filter tails and reverb decays

21. Lock-Free Atomic Memory Bus

- **What:** Atomic compare-and-swap (CAS) based memory bus replacing mutex-protected shared state
- **Why:** Mutexes can block for 10-1000µs if contention occurs. Lock-free atomics guarantee wait-free progress for readers
- **Impact:** 239.65× speedup vs mutex-based memory bus (100ns vs 24µs per operation), critical for NRPN parameter updates

22. Block Processing (64-sample blocks)

- **What:** Processes audio in 64-sample chunks rather than sample-by-sample
- **Why:** Amortises parameter updates, function call overhead, and atomic operations across 64 samples instead of 1
- **Impact:** 87.5% reduction in atomic operations (64× fewer calls), improves cache locality, reduces function call overhead

Memory Protection & Security

23. mlock() (Memory Locking)

- **What:** Locks synthesis engine memory pages in RAM, preventing them from being swapped to disk
- **Why:** Page faults (fetching swapped pages from SD card) take 5-50ms, causing catastrophic audio dropouts
- **Impact:** Eliminates swap-induced latency spikes, guarantees audio buffers are always in RAM

24. Anti-Ptrace Protection

- **What:** Prevents debuggers (GDB, LLDB) from attaching to the running synthesis process
- **Why:** Debugger attachment can pause threads for 10-100ms, causing dropouts. Also prevents reverse engineering of protection logic
- **Impact:** Prevents accidental debugging-induced dropouts in production, protects intellectual property

25. Capability Dropping

- **What:** Drops unnecessary Linux capabilities after initialisation (e.g., CAP_SYS_ADMIN, CAP_NET_ADMIN)

- **Why:** Principle of least privilege: synthesis only needs audio I/O and memory locking. Dropping unused capabilities reduces attack surface
- **Impact:** Hardens security against privilege escalation vulnerabilities

Audio Backend Configuration

26. JACK Audio Connection Kit (Default)

- **What:** Professional low-latency audio server with sample-accurate synchronisation
- **Why:** JACK provides deterministic callback-based audio with sub-millisecond latency, sample-accurate MIDI timing, and zero-copy buffer sharing
- **Impact:** Enables 2.9ms latency (64 samples @ 48kHz) with <1% CPU usage, sample-accurate MIDI-to-audio synchronisation

27. Zero-Allocation Audio Path

- **What:** Audio callback code performs zero heap allocations after initialisation
- **Why:** Heap allocations trigger GC pressure and can cause 100-1000µs pauses. Real-time audio cannot tolerate unpredictable delays
- **Impact:** Guarantees deterministic audio callback timing, prevents GC-induced dropouts

28. Pre-Allocated Buffer Pools

- **What:** Ring buffers and sample buffers allocated once at startup, reused indefinitely
- **Why:** Allocating buffers per-frame causes GC churn. Pooling eliminates allocations in the audio path
- **Impact:** Reduces GC allocations by 99%, improves cache locality (hot buffers stay in L1/L2 cache)

Raspberry Pi Hardware Configuration

All Raspberry Pi builds include hardware-level optimisations configured via `config.txt` (boot firmware) and `cmdline.txt` (kernel parameters). These settings are pre-configured in the Raspberry Pi OS image and require no user intervention.

29. HiFiBerry DAC2 Pro Device Tree Overlay

- **What:** `dtoverlay=hifiberry-dacplus` loads the HiFiBerry DAC2 Pro driver at boot
- **Why:** Enables the Texas Instruments PCM5122 DAC (117dB SNR, 32-bit/384kHz capable) for professional audio output
- **Impact:** Routes audio to HiFiBerry RCA outputs instead of onboard 3.5mm jack (60dB SNR)

30. Onboard Audio Disabled (`dtparam=audio=off`)

- **What:** Disables the Raspberry Pi's onboard PWM audio hardware
- **Why:** Onboard audio conflicts with HiFiBerry I2S pins and wastes DMA bandwidth. Also prevents accidental routing to low-quality PWM output
- **Impact:** Frees GPIO pins for HiFiBerry, eliminates 40dB SNR onboard audio from signal path

31. Raspberry Pi Hardware PWM Audio Disabled (`audio_pwm_mode=0`)

- **What:** Explicitly disables the Raspberry Pi's onboard PWM audio output hardware (3.5mm headphone jack)
- **Why:** Ensures no hardware PWM audio interference on GPIO pins used by HiFiBerry I2S communication. This has no impact on synthesis PWM (square wave duty cycle modulation in channels 0-63), which is a software DSP feature
- **Impact:** Guarantees clean I2S communication with HiFiBerry DAC, eliminates GPIO conflicts

32. CPU Frequency Locking

- **What:** Locks CPU frequency to maximum performance mode using `arm_freq` parameter

- **Why:** Dynamic frequency scaling (DVFS) causes 10-50ms transitions between 600MHz and max frequency. Audio callbacks during transitions miss deadlines
- **Impact:** Eliminates frequency scaling jitter, ensures consistent <1ms callback timing

Per-Platform Settings:

- **Pi Zero 2W:** `arm_freq=1200` (1.2GHz locked, vs 1.0GHz stock)
- **Pi 400:** `arm_freq=2000` (2.0GHz locked, vs 1.8GHz typical)
- **Pi 5/500:** `arm_freq=2400` (2.4GHz locked, vs 2.4GHz turbo)

33. Turbo Boost (`arm_boost=1`)

- **What:** Enables maximum CPU turbo boost mode on Pi 5/500 and Pi Zero 2W
- **Why:** Allows sustained maximum frequency without throttling (when thermal conditions permit)
- **Impact:** Pi 5/500 maintains 2.4GHz under load; Pi Zero 2W boosts to 1.2GHz

34. Overvoltage (Stability Enhancement)

- **What:** Increases CPU core voltage slightly above stock settings for stability at overclocked frequencies
- **Why:** Higher frequencies require higher voltages to maintain stable operation. Overvoltage prevents crashes during synthesis workloads
- **Impact:** Stable 2.4GHz operation on Pi 5/500, stable 2.0GHz on Pi 400, stable 1.2GHz on Pi Zero 2W

Per-Platform Settings:

- **Pi Zero 2W:** `over_voltage=2` (+0.1V, conservative for 512MB board)
- **Pi 400:** `over_voltage=6` (+0.3V, stable for 2.0GHz)
- **Pi 5/500:** `over_voltage=6` (+0.3V, stable for 2.4GHz)

35. Minimal GPU Memory (`gpu_mem=16`)

- **What:** Allocates only 16MB RAM to the GPU, maximising RAM available to synthesis
- **Why:** Synthesis is CPU-only; GPU is unused (no display output in headless mode). Default 64-256MB GPU allocation wastes RAM
- **Impact:** Frees 48-240MB RAM for synthesis engine, particularly critical on Pi Zero 2W (512MB total)

36. WiFi Disabled (`dtoverlay=disable-wifi`)

- **What:** Disables the onboard WiFi radio at boot
- **Why:** WiFi interrupts occur every 1-10ms (beacon frames, background scanning), causing IRQ storms that delay audio threads. Also eliminates RF interference near audio circuitry
- **Impact:** Removes 10-50µs interrupt jitter, improves audio signal-to-noise ratio by eliminating RF interference

37. Bluetooth Disabled (`dtoverlay=disable-bt`)

- **What:** Disables the onboard Bluetooth radio at boot
- **Why:** Bluetooth shares interrupts with WiFi and UART, causing 10-100µs delays. Also eliminates 2.4GHz RF interference
- **Impact:** Removes BT interrupt overhead, improves audio SNR

38. USB FIQ Disabled (`dwc_otg.fiq_fsm_enable=0`)

- **What:** Disables Fast Interrupt Request (FIQ) mode for the USB controller (Pi 3/Zero 2W only)
- **Why:** FIQ is a non-maskable interrupt that can delay audio threads by 50-200µs. Disabling it makes USB interrupts preemptible by the RT scheduler
- **Impact:** Prevents USB-induced latency spikes when MIDI controllers are connected

39. Tickless CPU Cores (Kernel Parameter)

- **What:** `nohz_full=2,3` disables the 1000Hz scheduler tick on cores 2-3
- **Why:** Scheduler ticks interrupt audio processing every 1ms. Tickless cores run uninterrupted until they voluntarily yield
- **Impact:** Reduces interrupt overhead by 95% on audio cores

40. Threaded IRQs (Kernel Parameter)

- **What:** `threadirqs` converts all hardware interrupts to kernel threads
- **Why:** Hard IRQs cannot be preempted and block audio threads. Threaded IRQs allow the RT scheduler to prioritise audio
- **Impact:** Prevents interrupt storms (network, disk) from blocking audio processing

Performance Summary

The combined impact of these 43 optimisations enables professional real-time synthesis on Raspberry Pi hardware:

Metric	Standard Linux	With RT Optimisations	Improvement
Minimum Latency	10-20ms	<1ms	10-20× better
Worst-Case Jitter	100-1000µs	<10µs	10-100× better
GC Pause Duration	1-5ms	<100µs	10-50× better
SIMD Speedup (Waveform)	1× (scalar)	1.8-2.4× (AVX2/NEON)	80-140% faster
Memory Bus Latency	24µs (mutex)	100ns (lock-free)	239.65× faster
Voice Allocation	O(N) scan	O(1) atomic bit-field	320× faster
CPU Usage (320 ch)	15-25%	<1% (Pi 5), ~50% (Pi3)	15-25× more efficient

These optimisations are pre-configured in the Raspberry Pi OS image. Users simply flash the image and power on—no manual configuration required.

19. Troubleshooting Guide

This section provides solutions to common issues musicians may encounter whilst using the Intuition Subsynth.

No Audio Output

Symptom: Instrument powers on, but produces no sound.

Possible Causes and Solutions:

1. Audio Connection Issues:

- **Check:** Ensure RCA cables are connected to the HiFiBerry DAC2 Pro **LINE OUT** (Left/Right)
- **Check:** Verify the other end is plugged into your mixer/interface (not a headphone-only input)
- **Check:** Test the RCA cables or adapters with another device to confirm they're working
- **Solution:** Replace faulty cables or adapters

2. Volume Settings:

- **Check:** Send CC#7 (Channel Volume) value 100 to MIDI channel 1
- **Check:** Verify amplifier/mixer volume is turned up
- **Check:** Check master volume on connected audio device
- **Solution:** Adjust volumes to audible levels

3. Audio Backend Not Running:

- **Check:** Verify JACK server is running (default backend)
- **Check:** Run `jack_lsp` to list JACK ports
- **Solution:** Start JACK server with correct sample rate (44.1kHz/48kHz)
- **Alternative:** Use `--legacy-audio-midi` flag for Oto backend

4. Wrong Audio Output Device:

- **Check:** Verify Subsynth is connected to correct JACK audio output
- **Check:** Run `jack_connect` to verify audio routing
- **Solution:** Connect Subsynth outputs to system playback ports

5. Muted MIDI Channels:

- **Check:** Verify MIDI channel volumes are not zero
 - **Solution:** Send CC#7 value 100 to all channels 1-16
-

MIDI Not Responding

Symptom: MIDI controller/sequencer connected, but notes don't play.

Possible Causes and Solutions:

1. MIDI Cable Connected Incorrectly:

- **Check:** MIDI cable must connect controller **MIDI OUT** to Subsynth **MIDI IN**
- **Common Mistake:** Connecting MIDI OUT to MIDI OUT (will not work)
- **Solution:** Verify cable direction (OUT → IN)

2. Wrong MIDI Channel:

- **Check:** Verify controller is sending on channels 1-16 (not channel 0 or >16)
- **Check:** Try sending on channel 1 (universal test channel)
- **Solution:** Configure controller to send on channel 1

3. MIDI Backend Not Running:

- **Check:** Verify JACK MIDI is running (default backend)
- **Check:** Run `jack_lsp` to see MIDI ports
- **Solution:** Start JACK server with MIDI support
- **Alternative:** Use `--legacy-audio-midi` for PortMIDI backend

4. USB-MIDI Device Not Recognised:

- **Check:** Verify USB-MIDI device is plugged into USB-C port
- **Check:** Run `lsusb` to verify device is detected (Linux)
- **Solution:** Ensure device is USB MIDI Class Compliant

5. Active Sensing Timeout:

- **Symptom:** MIDI works briefly, then stops
 - **Cause:** Controller sends Active Sensing (0xFE), then stops
 - **Solution:** Disable Active Sensing in controller settings, or keep controller powered on
-

Demo Mode Activated

Symptom: "DEMO MODE" banner appears on the UI, only 16 randomly selected melodic patches with full drum kit play, and a countdown timer forces a reboot after roughly 3-5 minutes.

Possible Causes and Solutions:

1. Raspberry Pi Serial Mismatch:

- **Check:** Run `grep Serial /proc/cpuinfo` on the Pi that is running Subsynth.
- **Compare:** The value must match the serial printed on your invoice and embedded in the download email.
- **Solution:** Flash the image only to the authorised Pi. A different board, even of the same model, will trigger Demo Mode until a new licence is purchased for the new board.

2. Modified or Corrupted Image:

- **Check:** Verify the SHA-256 checksum of your ZIP/IMG file before flashing.
- **Check:** Ensure `/opt/subsynth/subsynth_*` has not been altered.
- **Solution:** Re-download the image from the link in your order email and re-flash the SD card. Avoid editing the filesystem, the Subsynth binary or removing protection files.

3. Storage Errors / Power Loss During Write:

- **Check:** Inspect the SD card for errors (`fsck`) or re-image using a known-good card.
- **Solution:** Always eject the card safely after flashing and use high-quality power supplies to prevent corruption.

4. Genuine Hardware Replacement:

- **Scenario:** Your Raspberry Pi was replaced due to failure or upgrade.
- **Solution:** Purchase an additional licence for the new board.

Demo Mode is a protective grace period, not a fault. Once the correct hardware and unmodified software are restored, Subsynth immediately returns to full functionality.

Notes Stuck / Sustaining Forever

Symptom: Notes continue playing indefinitely after key release.

Possible Causes and Solutions:

1. Sustain Pedal Stuck On:

- **Check:** Sustain pedal (CC#64) may be stuck at value ≥ 64
- **Solution:** Send CC#64 value 0 to all channels (turns sustain off)
- **Prevention:** Ensure sustain pedal hardware is functioning correctly

2. Missing Note Off Messages:

- **Check:** Some MIDI controllers fail to send Note Off messages
- **Solution:** Send CC#123 (All Notes Off) value 0 to all channels
- **Alternative:** Send CC#120 (All Sound Off) for emergency muting

3. Sostenuto Pedal Active:

- **Check:** Sostenuto pedal (CC#66) may be holding specific notes
- **Solution:** Send CC#66 value 0 to release sostenuto

4. Infinite Release Time:

- **Check:** Patch may have extremely long release envelope (up to 15 seconds)
- **Solution:** Wait for release phase to complete, or send All Sound Off (CC#120)

Wrong Instrument Playing

Symptom: Unexpected instrument sounds when pressing keys.

Possible Causes and Solutions:

1. Wrong Programme Number:

- **Check:** Verify correct Programme Change sent (0-127 internal, displayed as 1-128)
- **Common Mistake:** Some software sends 1-128, others send 0-127
- **Solution:** Send Programme Change 0 for Acoustic Grand Piano (Programme #1)

2. Wrong Patchset Loaded:

- **Check:** Verify correct patchset loaded at startup (SIDney, AYYMe, Rawland, PixelaTED, EenBeetje)
- **Solution:** Restart with `--patchset=SIDney` (or desired patchset name)

3. Wrong Bank Selected:

- **Check:** Bank Select (CC#0/32) determines which patchset is active (Banks 0-4)
- **Solution:** Send CC#0 value to select desired patchset:
 - **Bank 0:** Reset to default (your `--patchset=NAME` choice, or SIDney if not specified)
 - **Bank 1:** AYYMe
 - **Bank 2:** Rawland
 - **Bank 3:** PixelaTED
 - **Bank 4:** EenBeetje
- **Note:** Many MIDI files send Bank 0 to "reset" - this now respects your startup `--patchset` choice

4. Channel 10 is Drums:

- **Check:** MIDI channel 10 is fixed to drum kit (GM Level 1 standard)
- **Solution:** Use channels 1-9, 11-16 for melodic instruments

Distorted / Clipping Audio

Symptom: Audio sounds harsh, distorted, or clipped.

Possible Causes and Solutions:

1. Too Many Notes Playing:

- **Check:** Playing >64 notes of same waveform type causes voice stealing
- **Solution:** Reduce polyphony or switch some channels to different waveform types
- **Explanation:** Square channels (0-63) share 64 voices; if 65+ square notes play, stealing occurs

2. Master Compressor Too Aggressive:

- **Check:** Compressor threshold may be too low or ratio too high
- **Solution:** Increase threshold (NRPN #130) or reduce ratio (NRPN #131)

3. Filter Self-Oscillation:

- **Check:** High filter resonance (CC#71 >100) causes self-oscillation
- **Solution:** Reduce filter resonance to <100

4. Overdrive Too High:

- **Check:** Per-channel or global overdrive may be set too high

- **Solution:** Reduce overdrive amount (NRPN #11, per-channel NRPN 1000-1319)

5. Reverb/Delay Feedback Too High:

- **Check:** Excessive feedback can cause infinite build-up
 - **Solution:** Reduce reverb decay (NRPN #7) or delay feedback (NRPN #1)
-

Pitch Issues

Symptom: Notes play at wrong pitch or drift out of tune.

Possible Causes and Solutions:

1. Pitch Bend Not Centred:

- **Check:** Pitch bend wheel may be off-centre (should be 8192)
- **Solution:** Centre pitch bend wheel on controller
- **MIDI:** Send pitch bend value 8192 (0xE0 0x00 0x40)

2. Master Tuning Adjusted:

- **Check:** Master tuning (NRPN #24) may be detuned
- **Solution:** Reset to concert pitch (NRPN #24 value 8192 = 0 cents)

3. RPN Fine Tuning Set:

- **Check:** RPN #1 (Fine Tuning) may be adjusted
- **Solution:** Reset to 0 cents (RPN #1 value 8192)

4. Pitch Bend Range Too Wide:

- **Check:** Pitch bend range may be set to ± 12 or ± 24 semitones (SIDney default is ± 12)
 - **Solution:** Reset pitch bend range to ± 2 semitones (RPN #0 MSB=2, LSB=0)
-

Performance / CPU Issues

Symptom: Audio dropouts, glitches, or high CPU usage.

Possible Causes and Solutions:

1. Buffer Size Too Small:

- **Check:** JACK buffer size may be too low (e.g., 32 samples)
- **Solution:** Increase buffer size to 64 or 128 samples
- **Trade-off:** Higher buffer = higher latency, but more stable

2. Sample Rate Too High:

- **Check:** 192kHz sample rate requires 4x CPU vs 48kHz
- **Solution:** Use 48kHz for most applications (192kHz only for mastering/analysis)

3. Too Many Active Channels:

- **Check:** 320 active channels + effects may overload CPU on older hardware
- **Solution:** Reduce polyphony or disable unused effects

4. SIMD Disabled:

- **Check:** SIMD optimisations may be disabled
- **Solution:** Remove `--disable-simd` flag to enable AVX2/NEON optimisations

5. Background Processes:

- **Check:** Other applications may be consuming CPU
- **Solution:** Close unnecessary applications, use dedicated audio system

Effect Not Working

Symptom: Effect parameter changes have no audible effect.

Possible Causes and Solutions:

1. Effect Not Enabled:

- **Check:** Most effects require enable flag via NRPN
- **Solution:** Send enable NRPN with value >8192 (e.g., NRPN #130 = 16383 for compressor)

2. Effect Send Level Zero:

- **Check:** Reverb (CC#91) and Chorus (CC#93) require send levels >0
- **Solution:** Send CC#91 value 64 for moderate reverb, CC#93 value 64 for moderate chorus

3. Effect Mix at 0%:

- **Check:** Effect wet/dry mix may be set to 0% (dry only)
- **Solution:** Increase effect mix via NRPN (e.g., NRPN #144 for ping-pong delay mix)

4. Wrong Channel:

- **Check:** Per-channel effects (NRPN 1000-5159) affect specific synthesis channels, not MIDI channels
- **Solution:** Ensure correct synthesis channel index (0-319)

5. Effect Parameter Out of Range:

- **Check:** NRPN value may be outside effective range
- **Solution:** Verify parameter range (0-16383 for 14-bit, threshold at 8192 for boolean)

20. Appendices

Appendix A: MIDI Controller Quick Reference

Essential Controllers:

CC#	Controller Name	Range	Function
1	Modulation Wheel	0-127	Vibrato depth (LFO → Pitch)
5	Portamento Time	0-127	Glide time (0-127ms)
7	Channel Volume	0-127	Overall channel level (exponential)
10	Pan	0-127	Stereo position (0=left, 64=centre, 127=right)
11	Expression	0-127	Dynamic volume control (pedal)
64	Sustain Pedal	0-127	Hold notes (<64=off, ≥64=on)
65	Portamento On/Off	0-127	Enable glide (<64=off, ≥64=on)
66	Sostenuto	0-127	Selective sustain (<64=off, ≥64=on)

67	Soft Pedal	0-127	Reduce velocity (<64=off, ≥64=on)
71	Filter Resonance	0-127	Filter Q factor (0=none, 127=self-oscillation)
74	Filter Cutoff	0-127	Filter frequency (20Hz - 20kHz, exponential)
84	Portamento Control	0-127	Source note for glide (MIDI note number)
91	Reverb Send	0-127	Reverb wet/dry mix (0=dry, 127=wet)
93	Chorus Send	0-127	Chorus wet/dry mix (0=dry, 127=wet)
120	All Sound Off	0-127	Emergency mute (immediate silence)
121	Reset All Controllers	0-127	Reset to defaults
123	All Notes Off	0-127	Release all notes (musical termination)

Appendix B: NRPN Parameter Quick Reference

Key Named Parameters (0-299):

NRPN	Parameter	Range	Description
7	Reverb Decay	0-16383	Reverb decay time (0-100%)
8	Reverb Diffusion	0-16383	Reverb density (0-100%)
9	Bit-Crush Bits	0-16383	Bit depth (1-16 bits)
10	Bit-Crush Sample Rate	0-16383	Sample rate reduction (1-64x)
24	Master Tune	0-16383	Global tuning (8192 = concert pitch, ±50 cents)
130	Compressor Enable	0-16383	Enable/disable (>8192 = on)
140	Ping-Pong Enable	0-16383	Enable/disable (>8192 = on)
141	Ping-Pong Left Time	0-16383	Left delay (1-2000ms)
143	Ping-Pong Feedback	0-16383	Feedback amount (0.0-0.95)
250	Filter Keyboard Track	0-16383	Keyboard tracking (0-100%)
251	Filter Envelope Amount	0-16383	Envelope modulation (-100% to +100%)

Per-Channel Parameters (1000-5159):

Address Formula: NRPN = 1000 + (param_type × 320) + channel

Type	Parameter	Base Address	Range	Description
0	Overdrive	1000	1000-1319	Per-channel overdrive
1	Filter Frequency	1320	1320-1639	Per-channel filter cutoff
2	Filter Resonance	1640	1640-1959	Per-channel filter Q
9	Attack	3880	3880-4199	Per-channel attack time

12	Release	4840	4840-5159	Per-channel release time
----	---------	------	-----------	--------------------------

Example: Set attack time for channel 128 (first sine wave):

$NRPN = 1000 + (9 \times 320) + 128 = 4008$
Value = 8192 (50% of 0-11.7s range = ~5.85s attack)

Appendix C: General MIDI Instrument Map

Programmes 1-8: Piano

Prog #	Instrument Name	Character
1	Acoustic Grand Piano	Classic piano
2	Bright Acoustic Piano	Brighter variant
3	Electric Grand Piano	Electric piano
4	Honky-tonk Piano	Detuned, honky-tonk
5	Electric Piano 1 (Rhodes)	Rhodes-style EP
6	Electric Piano 2 (Chorus EP)	Chorused EP
7	Harpsichord	Plucked keyboard
8	Clavinet	Funky clavinet

Programmes 9-16: Chromatic Percussion

Prog #	Instrument Name	Character
9	Celesta	Bell-like mallet
10	Glockenspiel	Bright metallic mallet
11	Music Box	Mechanical music box
12	Vibraphone	Vibraphone mallet
13	Marimba	Wooden mallet
14	Xylophone	Bright wooden mallet
15	Tubular Bells	Church bells
16	Dulcimer	Hammered strings

Programmes 17-24: Organ

Prog #	Instrument Name	Character
17	Drawbar Organ	Hammond-style drawbar
18	Percussive Organ	Percussive attack organ
19	Rock Organ	Overdriven organ

20	Church Organ	Pipe organ
21	Reed Organ	Harmonium-style
22	Accordion	Accordion (musette)
23	Harmonica	Blues harmonica
24	Tango Accordion	Bandoneon-style

Programmes 25-32: Guitar

Prog #	Instrument Name	Character
25	Acoustic Guitar (nylon)	Classical guitar
26	Acoustic Guitar (steel)	Steel-string guitar
27	Electric Guitar (jazz)	Clean jazz guitar
28	Electric Guitar (clean)	Clean electric
29	Electric Guitar (muted)	Palm-muted electric
30	Overdriven Guitar	Overdriven electric
31	Distortion Guitar	Heavy distortion
32	Guitar Harmonics	Harmonic overtones

Programmes 33-40: Bass

Prog #	Instrument Name	Character
33	Acoustic Bass	Upright bass
34	Electric Bass (finger)	Fingered bass
35	Electric Bass (pick)	Picked bass
36	Fretless Bass	Fretless electric
37	Slap Bass 1	Slap/pop bass
38	Slap Bass 2	Slap bass variant
39	Synth Bass 1	Analogue synth bass
40	Synth Bass 2	Synth bass variant

Programmes 49-56: Strings / Ensemble

Prog #	Instrument Name	Character
49	String Ensemble 1	Lush orchestral strings
50	String Ensemble 2	Slow attack strings
51	Synth Strings 1	Synth pad strings

52	Synth Strings 2	Synth strings variant
53	Choir Aahs	Choir vowel sounds
54	Voice Oohs	Choir "oohs"
55	Synth Voice	Synthesised voice
56	Orchestra Hit	Orchestral stab

Programmes 81-88: Synth Lead

Prog #	Instrument Name	Character
81	Lead 1 (square)	Square wave lead
82	Lead 2 (sawtooth)	Sawtooth lead
83	Lead 3 (calliope)	Steam calliope lead
84	Lead 4 (chiff)	Breathy lead
85	Lead 5 (charang)	Distorted lead
86	Lead 6 (voice)	Vocal-like lead
87	Lead 7 (fifths)	Fifth-interval lead
88	Lead 8 (bass + lead)	Bass and lead

Programmes 89-96: Synth Pad

Prog #	Instrument Name	Character
89	Pad 1 (new age)	Warm evolving pad
90	Pad 2 (warm)	Warm atmospheric pad
91	Pad 3 (polysynth)	Polyphonic synth pad
92	Pad 4 (choir)	Choir-like pad
93	Pad 5 (bowed)	Bowed string pad
94	Pad 6 (metallic)	Metallic pad
95	Pad 7 (halo)	Bright ambient pad
96	Pad 8 (sweep)	Sweeping filter pad

Full 128-programme GM map continues in similar format. See GM Level 1 specification for complete listings.

Appendix D: General MIDI Drum Map (Channel 10)

MIDI Note	Note Name	Drum Sound	Typical Character
35 (B0)	B0	Acoustic Bass Drum	Deep acoustic kick

36 (C1)	C1	Bass Drum 1	Electronic kick
37 (C#1)	C#1	Side Stick	Rim click
38 (D1)	D1	Acoustic Snare	Snare drum
39 (D#1)	D#1	Hand Clap	Hand clap
40 (E1)	E1	Electric Snare	Electronic snare
41 (F1)	F1	Low Floor Tom	Low tom
42 (F#1)	F#1	Closed Hi-Hat	Tight hi-hat
43 (G1)	G1	High Floor Tom	High tom
44 (G#1)	G#1	Pedal Hi-Hat	Half-open hi-hat
45 (A1)	A1	Low Tom	Low tom
46 (A#1)	A#1	Open Hi-Hat	Open ringing hi-hat
47 (B1)	B1	Low-Mid Tom	Low-mid tom
48 (C2)	C2	Hi-Mid Tom	Hi-mid tom
49 (C#2)	C#2	Crash Cymbal 1	Crash cymbal
50 (D2)	D2	High Tom	High tom
51 (D#2)	D#2	Ride Cymbal 1	Ride cymbal
52 (E2)	E2	Chinese Cymbal	Chinese cymbal
53 (F2)	F2	Ride Bell	Ride bell
54 (F#2)	F#2	Tambourine	Tambourine
55 (G2)	G2	Splash Cymbal	Splash cymbal
56 (G#2)	G#2	Cowbell	Cowbell
57 (A2)	A2	Crash Cymbal 2	Crash cymbal 2
58 (A#2)	A#2	Vibraslap	Vibraslap
59 (B2)	B2	Ride Cymbal 2	Ride cymbal 2

Full 47-sound drum map continues. See GM Level 1 specification for complete listings (notes 35-81).

Appendix E: Raspberry Pi Safety References

Subsynth ships as software only. All electrical safety, electromagnetic compliance, end-of-life recycling, and product labelling obligations reside with the Raspberry Pi hardware and peripherals you supply. For authoritative guidance, download the “Important Product Information” leaflet for your exact board from <https://www.raspberrypi.com/documentation/> or refer to the printed leaflet bundled with retail kits. That documentation covers:

- Certified power supply ratings and plug types for Raspberry Pi 3, 4, 400, 5, 500, and Zero 2 W

- Environmental limits (temperature, humidity, altitude) and ventilation requirements
- ESD handling, disposal (WEEE), and regulatory notices (CE, UKCA, FCC, ISED, RoHS)
- Any additional constraints specific to your enclosure, PSU, or country of operation

Keep the official Raspberry Pi leaflet with your instrument build. Contact Raspberry Pi Ltd or your hardware reseller with questions about hardware compliance. This manual only describes the Subsynth software experience layered on top of that hardware.

Appendix A: GM Level 2 Compliance

The Intuition Subsynth is **fully compliant** with General MIDI Level 2 specification (GM2), providing extended features beyond GM Level 1 for professional music production.

GM Level 2 Feature Implementation

Feature Category	GM2 Feature	Status	Notes
Polyphony	32 simultaneous voices (minimum)	✓ Implemented	320 synthesis channels (far exceeds minimum)
MIDI Channels	16 simultaneous channels	✓ Implemented	Full 16-channel support
Percussion Channels	Channel 10 (always drums)	✓ Implemented	Standard GM1/GM2 compliance
	Channel 11 (melodic or drums)	✓ Implemented	Convertible via Bank 120
	Bank 0x78 (dynamic percussion mode)	✓ Implemented	Any channel 1-16 can be drums
Programme Changes	128 melodic instruments (programmes 1-128)	✓ Implemented	All 128 GM programmes per patchset
	Drum kits (percussion)	✓ Implemented	Full GM drum map (notes 35-81)
Bank Select	CC#0 (Bank Select MSB)	✓ Implemented	Banks 0-5 (patchsets), Bank 120 (percussion conversion)
	CC#32 (Bank Select LSB)	✓ Implemented	Reserved (currently unused, GM2 requires recognition)
Controllers	Standard MIDI CC (120+ controllers)	✓ Implemented	Volume, Pan, Expression, Sustain, Modulation, etc.
	NRPN (14-bit extended parameters)	✓ Implemented	4360+ NRPN parameters (exceeds GM2 minimum)
Pitch Control	Pitch Bend (14-bit)	✓ Implemented	±1 to ±24 semitones, default ±2
	RPN 0 (Pitch Bend Range)	✓	Per-channel pitch bend range control

		Implemented	
	RPN 1 (Fine Tuning)	✓ Implemented	±100 cents per-channel
	RPN 2 (Coarse Tuning)	✓ Implemented	±24 semitones per-channel
Velocity	Note-On velocity (1-127)	✓ Implemented	Exponential curve, patchset-specific response
Aftertouch	Channel Pressure (mono aftertouch)	✓ Implemented	Modulates filter, volume, vibrato (October 2025)
	Polyphonic Key Pressure (poly aftertouch)	✓ Implemented	Per-note modulation (October 2025)
System Exclusive (SysEx)	GM1 System On (F0 7E 7F 09 01 F7)	✓ Implemented	Resets to GM Level 1 mode
	GM2 System On (F0 7E 7F 09 03 F7)	✓ Implemented	Enables GM Level 2 features
	Master Volume (F0 7F 7F 04 01 ...)	✓ Implemented	14-bit global volume (0-16383)
	Master Fine Tuning (F0 7F 7F 04 03 ...)	✓ Implemented	±100 cents global tuning
	Master Coarse Tuning (F0 7F 7F 04 04 ...)	✓ Implemented	±24 semitones global transposition (supports ±64 internally)
Reverb	CC#91 (Reverb Send Level)	✓ Implemented	Schroeder algorithm (4 comb + 2 allpass)
Chorus	CC#93 (Chorus Send Level)	✓ Implemented	Stereo LFO modulation
Sound Controllers	CC#70-79 (sound controllers)	✓ Implemented	Filter cutoff, resonance, envelope, LFO, etc.
Performance Controllers	CC#1 (Modulation), CC#64 (Sustain)	✓ Implemented	Standard performance controls
Channel Mode Messages	All Notes Off, Reset All Controllers	✓ Implemented	Full channel mode support
Active Sensing	0xFE (connection monitoring)	✓ Implemented	Auto-disconnect detection
System Reset	0xFF (reset message)	✓ Implemented	Power-on state restoration

GM2 Compliance Summary

Compliance Level: 100% Full Compliance

The Intuition Subsynth implements **all mandatory GM Level 2 features** as defined by the MIDI Manufacturers Association specification:

-  Extended polyphony (320 channels vs 32 minimum)
-  Dual percussion channels (Channels 10 & 11)
-  Dynamic rhythm conversion (Bank 120 on any channel)
-  Universal SysEx support (GM1/GM2 System On, Master Volume/Tuning)
-  Full controller set (120+ MIDI CC, 4360+ NRPN)
-  Aftertouch support (polyphonic and channel pressure)
-  Complete GM programme set (128 melodic instruments + drums)

Extended Features Beyond GM2 Minimum:

- **Voice count:** 320 synthesis channels (10× GM2 minimum of 32)
- **NRPN parameters:** 4360+ vs GM2 minimum (comprehensive DSP control)
- **Patchset system:** 6 complete GM soundbanks (SIDney, AYYMe, Rawland, PixelaTED, EenBeetje, Factory)
- **Master tuning range:** ±64 semitones internally (GM2 spec: ±24)
- **Real-time performance:** <1% CPU usage on modern Raspberry Pi 5, professional-grade latency (2.9ms JACK)

Reference Documentation:

For practical usage of GM Level 2 features, see:

- **Sound Bank Selection** → GM Level 2: Dynamic Rhythm Conversion (Bank 120)
- **MIDI Channels** → Dual Percussion Channels (GM Level 2)
- **System Exclusive Messages** → Complete SysEx message formats and examples

END OF MANUAL

Copyright Notice:

This manual describes the Intuition Subsynth hardware synthesiser. All specifications, features, and implementations described herein are subject to change without notice.

Acknowledgements:

- Raspberry Pi hardware platforms: Raspberry Pi Ltd
- MOS 8580 SID emulation: Bob Yannes (original chip designer)
- CBM TED 7360/8360 emulation: David DiOrio, Bruce Ahrens, Robert Raible, and Chaoyan Eric Yang (original chip designers)
- AY-3-8910 / YM2149 PSG emulation: General Instrument / Yamaha
- Roland-inspired synthesis: Roland Corporation
- ZX Spectrum beeper techniques: Tim Follin, David Whittaker and the beeper music community

Trademarks:

- Intuition Subsynth and Subsynth are trademarks of Zayn Otley
- Raspberry Pi is a trademark of Raspberry Pi Ltd
- General MIDI, MIDI, and the MIDI logo are registered trademarks of the MIDI Manufacturers Association
- Roland, TB-303, TR-808, TR-909 are trademarks of Roland Corporation
- Commodore 64, C64, SID, Plus/4, C16, TED are trademarks of Commodore Business Machines
- ZX Spectrum is a trademark of Sinclair Research Ltd
- Amstrad and Amstrad CPC are trademarks of Amstrad plc
- MSX is a trademark of MSX Licensing Corporation
- Atari and Atari ST are trademarks of Atari Interactive, Inc.

- All other trademarks are the property of their respective owners