Analog Guitar Effect with User Presets

Recallable and Reprogrammable Settings Based on a Roland Dimension-D, and Powered by the STM32F29I-DISC1

Jesse Rhew
jesse.rhew@gmail.com

Overview

The Roland Dimension-D was an analog rackmount effect unit popular in music production through the 1980s. It was famous for having only 5 buttons (one of which was an OFF) that engaged different modulation effects on the input audio signal.

Video Demonstration, Source: Retrosonic Pro Audio

Using an STM32 demo board, I recreated that functionality in a Dimension-style analog circuit, but without the Dimension-D's mechanical buttons. This enables me to put the full authentic effect into a smaller floor or desktop-based unit. The device had "secret settings" some studios used as signature sounds and were achieved by pressing more than one of the buttons down at once, managing to get them to lock into place despite the button array's shared mechanical reset. Some modern replications of the Dimension-D omit these secret settings, but I am able to add them back as well as saving my own bank of settings, and allowing the user to program the device and save their own personal settings.

Microcontroller Hardware Used

STM32F29I-DISC1

Processor: STM32F429ZIT
Arm® Cortex®-M4

Software Environment

Programmed and debugged via STM32CubeIDE using the STMicroelectronics STM32F4 Hardware Abstraction Layer.

Device Description

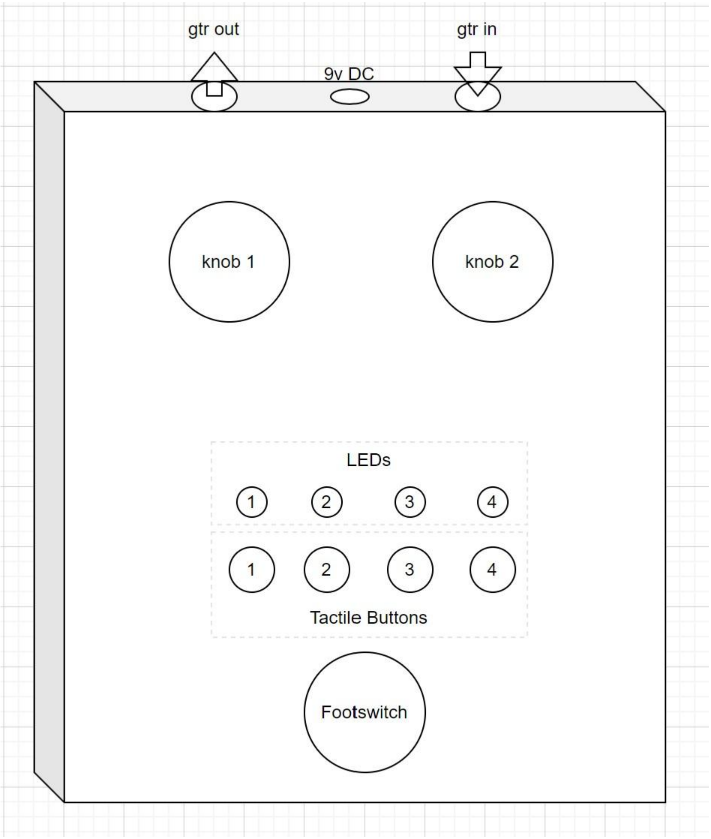


Fig 1 - Device UX Layout

The device will have 2 knobs consisting of potentiometers acting as a voltage divider between GND and a reference voltage (5 Vdc), each read by a different ADC pin on the microcontroller. Knob 1 will control the level of the DEPTH, Knob 2 will control the RATE. Settings are written to digital potentiometers in the analog signal-modulating circuit via I^2C. When a knob turn is detected (and is over a specified THRESHOLD), the unit snaps out of BUTTONS mode, goes into KNOBS mode, and is controlled entirely by the positions of the DEPTH and RATE knobs.

The footswitch will turn the unit from On (signal effect) mode to Off (guitar signal bypass) mode.

Factory settings and user-programmed settings are stored in flash memory, and re-written by holding a specific

The tactile buttons select a preset to be used. The unit reads the preset from memory, then writes the correct values to the device's internal digital potentiometers.

To access alternate button functions, Button 1 can be held down for 2 seconds and the LED lights will begin to blink. While still holding Button 1, Buttons 2 and 3 are used to move up or down to new "banks" of presets, indicated by a different color on the RGB LED array.

To write a new preset, the user must go to the USER bank, find their desired settings using the Knobs, then hold down the button they want to write their preset to for 5 seconds. During the hold time, the corresponding LED will swell, then a quick 3 blinks indicates that the new preset has been written. If the user lets go of the button during the LED swell, the unit will return to the previous state.

Presets can also include multiple buttons pressed at once. To select one of these “secret” presets, simply press two of the buttons at once. To write to a two-button preset, go to a USER bank, then write a preset by holding the desired two buttons as you would if writing to a single button.

The device is always on when connected to power, as guitar pedals are usually set to a desired setting, then left alone for long periods that might otherwise require a TIMEOUT in another system.

Power is supplied via a $9V_{DC}$ 2.1mm type-M connector.

Function Diagrams

Hardware Block Diagram

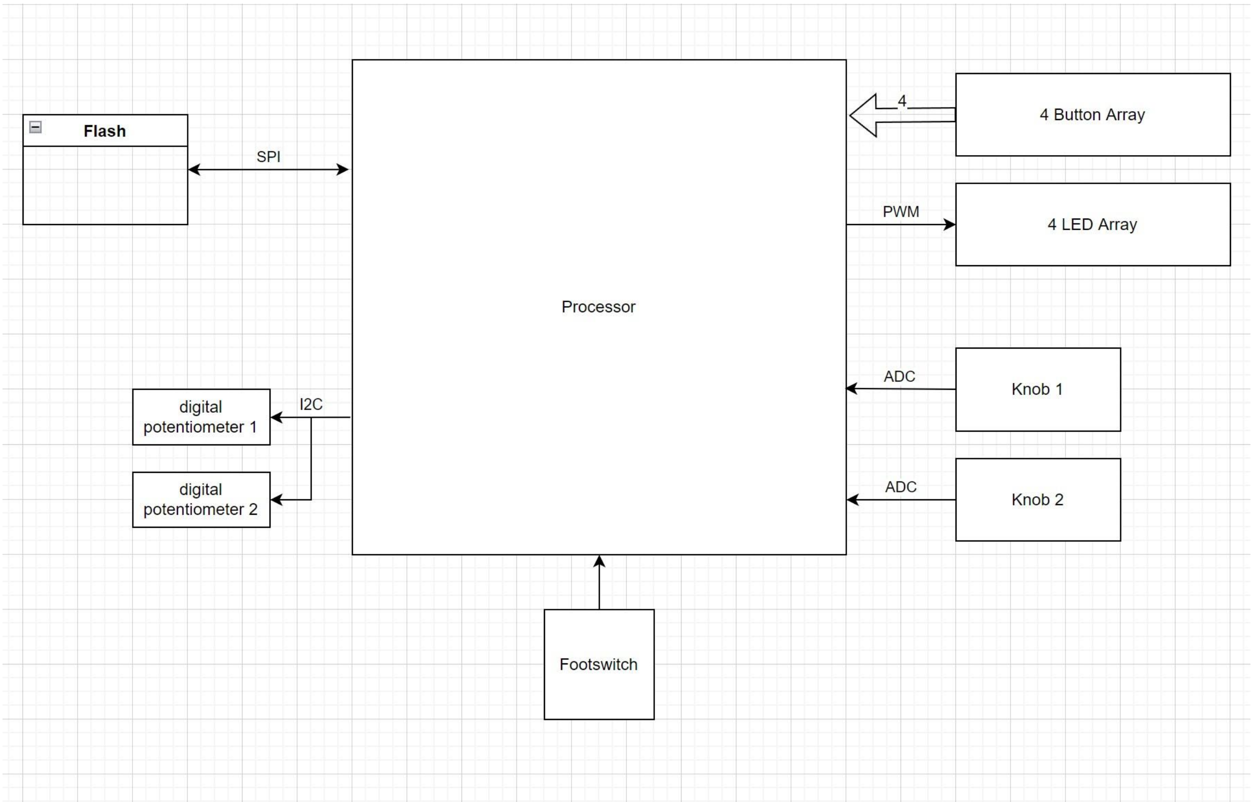


Fig 2 - Hardware Block Diagram

Software Block Diagram

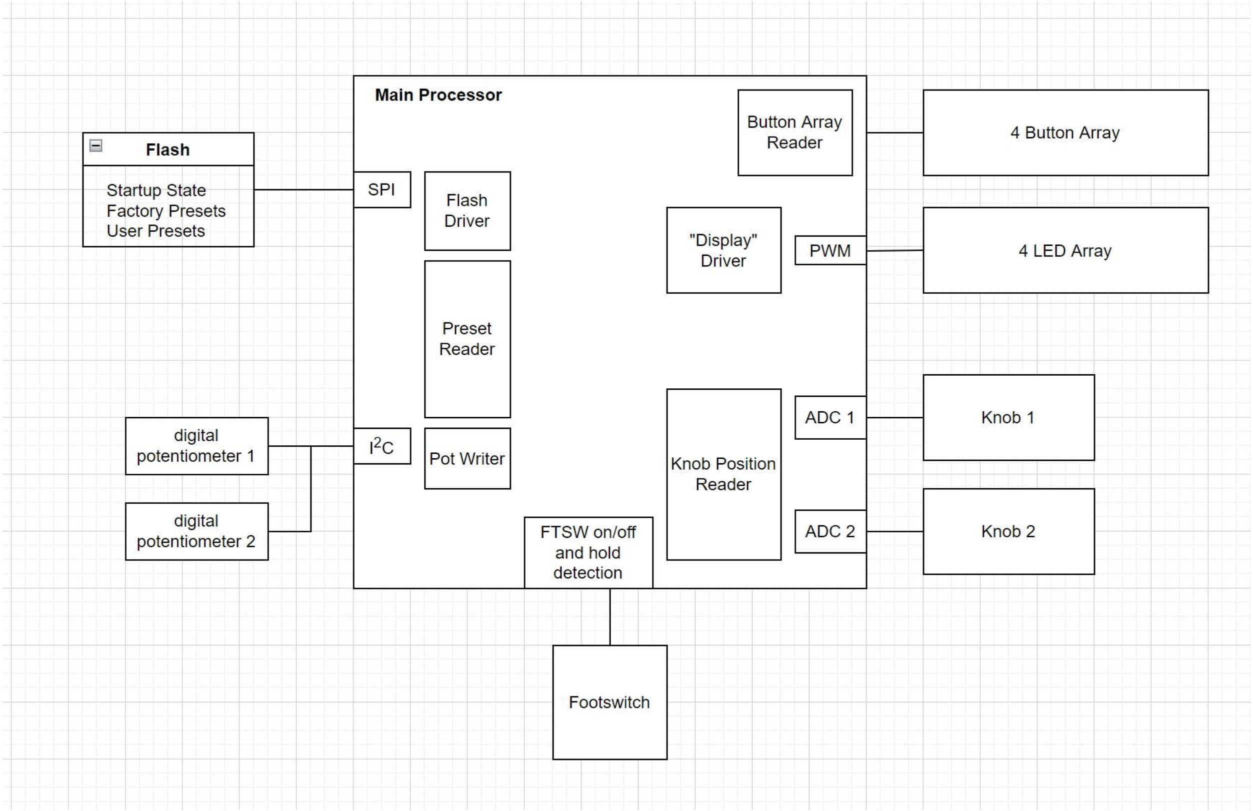


Fig 3 - Software Block Diagram

Hierarchy of Control

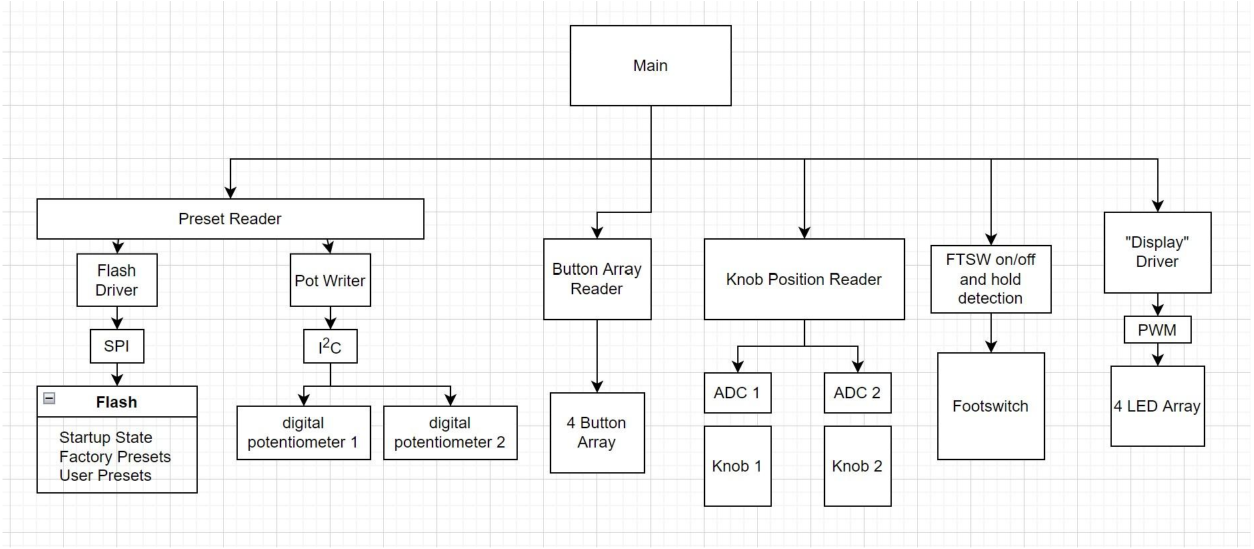
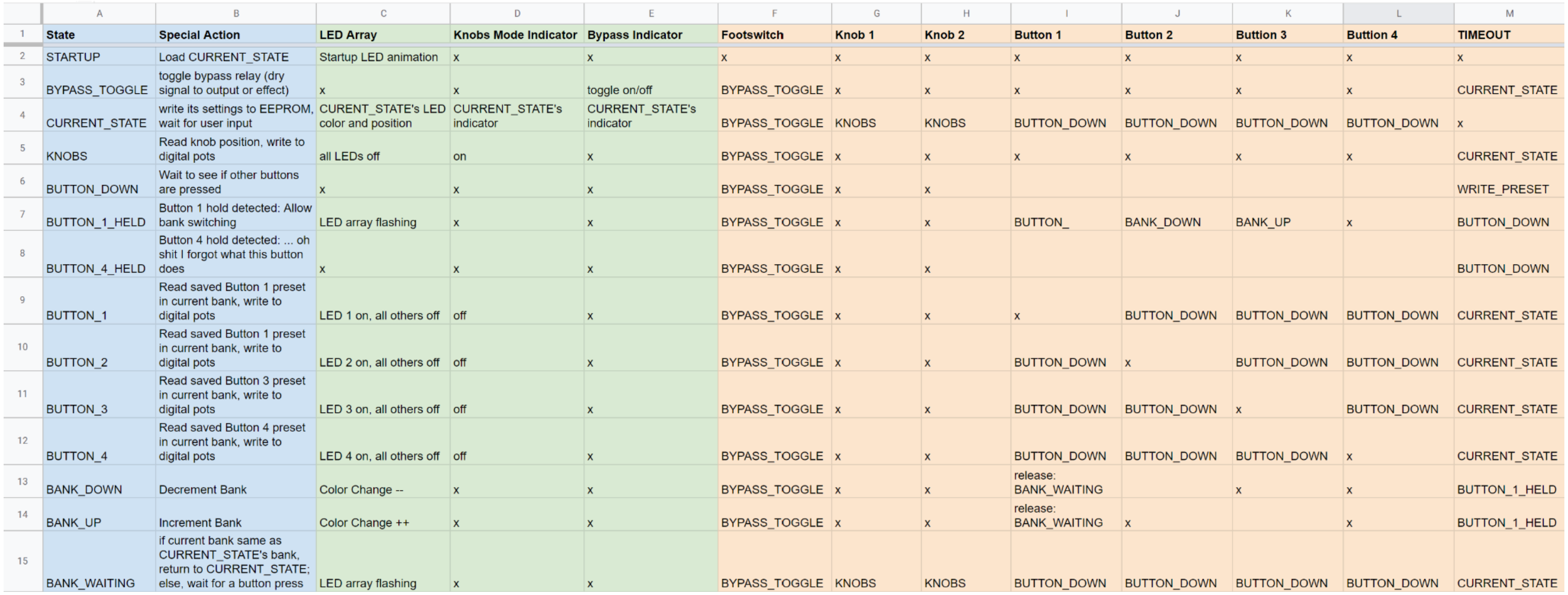


Fig 4 - Hierarchy of Control

State Table



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	**State**	**Special Action**	**LED Array**	**Knobs Mode Indicator**	**Bypass Indicator**	**Footswitch**	**Knob 1**	**Knob 2**	**Button 1**	**Button 2**	**Buttion 3**	**Buttion 4**	**TIMEOUT**
2	STARTUP	Load CURRENT_STATE	Startup LED animation	x	x	x	x	x	x	x	x	x	x
3	BYPASS_TOGGLE	toggle bypass relay (dry signal to output or effect)	x	x	toggle on/off	BYPASS_TOGGLE	x	x	x	x	x	x	CURRENT_STATE
4	CURRENT_STATE	write its settings to EEPROM, wait for user input	CURENT_STATE's LED color and position	CURRENT_STATE's indicator	CURRENT_STATE's indicator	BYPASS_TOGGLE	KNOBS	KNOBS	BUTTON_DOWN	BUTTON_DOWN	BUTTON_DOWN	BUTTON_DOWN	x
5	KNOBS	Read knob position, write to digital pots	all LEDs off	on	x	BYPASS_TOGGLE	x	x	x	x	x	x	CURRENT_STATE
6	BUTTON_DOWN	Wait to see if other buttons are pressed	x	x	x	BYPASS_TOGGLE	x	x					WRITE_PRESET
7	BUTTON_1_HELD	Button 1 hold detected: Allow bank switching	LED array flashing	x	x	BYPASS_TOGGLE	x	x	BUTTON_	BANK_DOWN	BANK_UP	x	BUTTON_DOWN
8	BUTTON_4_HELD	Button 4 hold detected: ... oh shit I forgot what this button does	x	x	x	BYPASS_TOGGLE	x	x					BUTTON_DOWN
9	BUTTON_1	Read saved Button 1 preset in current bank, write to digital pots	LED 1 on, all others off	off	x	BYPASS_TOGGLE	x	x	x	BUTTON_DOWN	BUTTON_DOWN	BUTTON_DOWN	CURRENT_STATE
10	BUTTON_2	Read saved Button 1 preset in current bank, write to digital pots	LED 2 on, all others off	off	x	BYPASS_TOGGLE	x	x	BUTTON_DOWN	x	BUTTON_DOWN	BUTTON_DOWN	CURRENT_STATE
11	BUTTON_3	Read saved Button 3 preset in current bank, write to digital pots	LED 3 on, all others off	off	x	BYPASS_TOGGLE	x	x	BUTTON_DOWN	BUTTON_DOWN	x	BUTTON_DOWN	CURRENT_STATE
12	BUTTON_4	Read saved Button 4 preset in current bank, write to digital pots	LED 4 on, all others off	off	x	BYPASS_TOGGLE	x	x	BUTTON_DOWN	BUTTON_DOWN	BUTTON_DOWN	x	CURRENT_STATE
13	BANK_DOWN	Decrement Bank	Color Change --	x	x	BYPASS_TOGGLE	x	x	release: BANK_WAITING		x	x	BUTTON_1_HELD
14	BANK_UP	Increment Bank	Color Change ++	x	x	BYPASS_TOGGLE	x	x	release: BANK_WAITING	x		x	BUTTON_1_HELD
15	BANK_WAITING	if current bank same as CURRENT_STATE's bank, return to CURRENT_STATE; else, wait for a button press	LED array flashing	x	x	BYPASS_TOGGLE	KNOBS	KNOBS	BUTTON_DOWN	BUTTON_DOWN	BUTTON_DOWN	BUTTON_DOWN	CURRENT_STATE