

LEDBlinky
Arcade LED Control Software
Version 8.2
Created by Arzoo

Configuration Guide

Overview.....	3
LEDBlinky Website	5
Installation	6
How Do I Get This Thing To Work?	7
Configuring Other Emulators (not MAME)	9
It's Not Working, What Should I Do?	10
Running LEDBlinky in Stand-Alone (Command Line) Mode.....	11
LEDBlinky Core Application 	14
Generate LEDBlinky Input Map Application 	15
Menus	20
LEDBlinky Configuration Application 	21
Menus	21
Game Options	23
FE Options	28
MaLa / AtomicFE / GameEx / PinballX / HyperSpin / LaunchBox / Maximus Arcade / Attract-Mode / CoinOps / RetroFE Options	32
MAME	35
Audio.....	38
Misc Options	44
Integrations	50
Game Specific Animations	56
LEDBlinky Controls Editor 	57
Menus	58
Emulator and ROM/Game Lists	61
Edit Emulator.....	64
Edit ROM/Game and Controls.....	67
Edit Front-end (FE) and Controls	71
Edit Individual Control	73
How to Specify JDR Mode/Map Overrides	77
MAME Output System	78
LEDBlinky Output System	81
Appendix A: Supported LED Controllers.....	83
Appendix B: File Descriptions.....	84
Appendix C: Credits.....	89
Index.....	90

Overview

LEDBlinky is arcade LED control software for use with [supported LED controllers](#). With the LEDBlinky plug-in, you can light your control panel's LEDs to indicate which controls (buttons, joysticks, trackball, etc.) are active during game play (MAME or other emulators), plus many other features. Here's a list of the current features:

- Supports multiple LED controllers from multiple vendors. See [Appendix A](#) for a list of all supported LED controllers. You can use multiple combinations of any supported LED controller.
- Supports MaLa, AtomicFE, GameEx, PinballX, and Attract-Mode integration via plug-in OR stand-alone operation for HyperSpin, LaunchBox, Maximus Arcade, CoinOps, RetroFE, or any arcade Front-End (FE) software.
- Light active and/or inactive controls for MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, RetroFE user interface.
- Light active and/or inactive controls for MAME emulation.
- Light active and/or inactive controls for other emulators.
- Easily configure control attributes (color, intensity, spoken action, input codes, etc.) on a game by game basis, or define defaults for each emulator.
- Remapping controls (MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, RetroFE, or MAME) will automatically remap the associated LEDs.
- With RGB LEDs, you can specify colors for individual controls or using a pre-defined colors.ini file, match the original game control panel button colors. Colors or intensities can also be customized on a game-by-game basis.
- Use audio output (music or game sounds) to blink, fade, or animate LEDs – great for use with Jukebox software.
- Blink and speak front-end UI controls by pressing a pre-defined “Help” button.
- Blink and speak controls and/or high score when pausing a game and/or play an LED animation (selected, random, random montage, or specific to each game) or use audio output (music) to animate the LEDs. This is a MAME only feature.
- Flash start buttons when credits are available - this is a MAME only game dependent feature.
- Light start and coin buttons based on active player count for the current game.
- Full support for other MAME Outputs - light LEDs based on any output. Outputs can be linked to controls (P1_Button1, P2_Button2, etc.) or directly linked to an LED controller port. Can also send a Pixilecade command or web request.
- Extensive audio animation options let you completely customize how the LEDs blink to music or game sounds.
- When starting a game, LEDBlinky can play an LED animation (selected, random, or specific to each game), speak the game name, speak each button “action” while blinking the button in its correct color, speak the primary controls, speak the high score (MAME only), and speak a custom message. When speaking the game name, high score, or custom message, LEDs can blink in sync with the

speech. Can also send a Windows shell command, Pixlecade command, or web request.

- While playing a game, LEDBlinky can play a continuous LED animation (selected, random, random montage, or specific to each game) or use audio output (game sounds) to animate the LEDs. The LED animation will only effect unused controls.
- When starting or quitting the front-end, LEDBlinky can play an LED animation (selected or random), and speak a custom message. When speaking the custom message, LEDs can blink in sync with the speech. An LED animation can also play whenever the front-end UI is active (including audio animations). Can also send a Windows shell command, Pixlecade command, or web request.
- When changing game lists or emulator lists, LEDBlinky can play an LED animation (selected or random). Can also send a Windows shell command, Pixlecade command, or web request.
- When the Front-End and/or Screen Saver is active, LEDBlinky can play a continuous LED animation (selected, random, or random montage) or use audio output (music) to animate the LEDs. Random custom messages can be played at predefined intervals. Berzerk mode is also available for the random messages. Can also send a Windows shell command, Pixlecade command, or web request.
- Other speech features are available – Choose from multiple voices (downloadable) and set the voice rate and volume.
- Run (independent) LED animations for cabinet lights. These animations can run for all standard LEDBlinky events; FE Start, FE Quit, FE Active, FE Screensaver, Game Start, Game Pause, and Game Quit. Animations can also be specified for individual games. This feature requires an additional LED controller for the non-control panel LEDs.
- From the front-end user interface, game controls can be lit as you scroll through the game lists. This feature is only supported by MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, and RetroFE.
- Designate LEDs as “Always Active” for use with coin, start, or administration controls.
- Cocktail Mode lights all player controls for multi-player alternating games.
- Use the LED-Wiz built-in blinking effects.
- Set the GP-Wiz49 (for 49-Way joysticks) or Ultimarc UltraStik 360 joystick digital restriction based on the currently selected game’s primary control.
- Switch the Ultimarc ServoStik’s restrictor plate between 4-way and 8-way based on the currently selected game’s primary control.
- Support for pre-defined controller files (X-Arcade, SlikStik, etc).
- No Microsoft COM control dependencies.
- Use the LEDBlinky Animation Editor to create your own animations.
- Supports Pixelcade; LED/LCD Marquee for Arcade Machines.

LEDBlinky Website

For software updates and online support, please visit the [LEDBlinky website](#). I have also documented the software development, along with my MAME project and other arcade related stuff on my [Arcade Addiction](#) site.

Installation

Download the latest installation package from the [LEDBlinky website](#). Run the LEDBlinkySetup.exe program. If your front-end software is MaLa, GameEx, PinballX, AtomicFE, or Attract-Mode, then install LEDBlinky into the front-end \plugins folder and select the relevant front-end plugin. Otherwise, you may install LEDBlinky into any folder.

Note: If your front-end is LaunchBox or BigBox, do not install LEDBlinky in the \plugins folder. This may cause “Bad IL Format” errors when running LB/BB.

The LEDBlinky plug-in should not be used in conjunction with any other MaLa, AtomicFE, GameEx, PinballX, or Attract-Mode plug-in that controls the listed [LED controllers](#). If you have installed another LED plug-in, it should be disabled or removed from the *plugins* folder.

The LEDBlinky plug-in should not be used in conjunction with any native front-end feature that controls the [listed LED controllers](#). Please disable any MaLa, AtomicFE, GameEx, PinballX, or Attract-Mode LED features.

Optionally install RocketBlinky. RocketBlinky can auto-generate an LEDBlinky Controls file that supports over 160 systems. Please see the Readme.txt in the RocketBlinky folder for installation and support options. RocketBlinky is a 3rd-party tool and is not supported by LEDBlinky.

How to Upgrade

Download the latest installation package from the [LEDBlinky website](#). Run the LEDBlinkySetup.exe program. The installation folder must match your current \LEDBlinky folder. Only updated files will be installed and your existing configuration will not be altered.

Note: When the upgrade runs successfully, you will receive a message at the end with the upgraded version number.

Note: If you are upgrading an older version that was installed from a .zip file (not from the LEDBlinkySetup.exe), then the installation path should be set to the \LEDBlinky parent folder (the folder above \LEDBlinky). Otherwise, you will end up with \LEDBlinky\LEDBlinky. If this happens, delete the inner \LEDBlinky folder and run the setup again, pointing to the parent folder.

How Do I Get This Thing To Work?

Basic Setup (LEDBlinky Configuration Wizard)

Basic setup is recommended for anyone using LEDBlinky for the first time. Run the LEDBlinky Configuration Wizard (*LEDBlinkyConfigWizard.exe*). The Configuration Wizard will guide you through the process of creating your input map and setting other required options with step-by-step questions. To use the Configuration Wizard, you must have your control panel and all LED controllers connected. Using the wizard may take a bit longer, but you can stop at any time and the app will pick up where you left off the next time it's run.

When the Configuration Wizard is complete, run your front-end and a few MAME games to see LEDBlinky in action. You can then move on to the Advanced Setup to enable and configure additional features.

Note: Using the Configuration Wizard is optional.

Note: Do not use the Configuration Wizard if any of the following are true: Your system does not have any LEDs. You have any RGB LEDs that do not have all three leads wired to individual ports. You have more than one LED wired to the same port on a controller.

Advanced Setup

Advanced setup is recommended for more experienced LEDBlinky users. The advanced setup provides many more LEDBlinky features and is necessary for lighting controls for non-MAME games. You do not need to complete the basic setup prior to the advanced setup. Follow these general steps;

Step 1

 Run the Generate LEDBlinky Input Map application (*GenLEDBlinkyInputMap.exe*). Before you can use LEDBlinky, you must create your LED controller Input Map.

Note: This step is not necessary if you have previously completed the LEDBlinky Configuration Wizard.

The input map defines the relationship between each wired port on your LED controller(s), and the keyboard input code for a button or input codes for other controls (Joysticks, Trackball, etc). It also allows you to assign a Port Label and LED Type to each wired port. The Port Labels serve two purposes, they tie together three ports for RGB LEDs, and they provide an easy reference to the ports from within the other LEDBlinky tools. The LED Type defines LEDs as Single, Red, Green, or Blue.

For example; let's say LED controller ports 1, 2, and 3 are wired to an RGB LED under a button. That button is wired to the keyboard encoder and assigned to the letter "A".

We also have an LED under our TrackBall wired to port 4. The input map would have four entries:

Port	Port Label	LED Type	Input Codes
1	P1B1	Red	KEYCODE_A
2	P1B1	Green	KEYCODE_A
3	P1B1	Blue	KEYCODE_A
4	TRACKBALL	Single	MOUSECODE_1_ANALOG_X MOUSECODE_1_ANALOG_Y MOUSECO...
5			

A full description on how to use this application is provided in a later [section](#) of this document.

Note: The input map is not necessary if LEDBlinky will only be used to set the joystick digital restriction (JDR) mode for UltraStik 360 or 49-Way joysticks, or for use with the Pixelcade marquee.

Step 2

 Run the LEDBlinky Configuration app (*LEDBlinkyConfig.exe*). This app is used to configure all LEDBlinky features and options. If you have previously completed the Configuration Wizard, some LEDBlinky configuration options will already be set.

Step 2a

From the "FE" menu, select your Front-End software (current options: AtomicFE, Attract-Mode, CoinOps, GameEx, HyperSpin, LaunchBox, MaLa, Maximus Arcade, PinballX, RetroFE, or Other).

Step 2b

From the "MAME Config" tab; Set the paths to the required (and optional) MAME files and folders. If you wish to set the LED colors based on the actual game control panel, set the path to Colors.ini. A version of this file with authentic control panel colors supporting over 1200 ROMs (at last count) has been included with LEDBlinky. You may skip this step if you do not intend on running the MAME emulator.

Step 2c

From the "Game Options" tab; You can enable LEDBlinky to blink each button and speak the button "action". You can also configure LEDBlinky to play an LED animation, speak the game name, and speak a custom message, all before the game starts. Other Game options are available.

Step 2d

From the "FE Options" tab; You can configure various "Attract" features when the front-end starts, when switching lists, when the screen saver is active, and when the front-end quits. MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, or RetroFE specific features are also available.

Note: Features are only available for front-end's that support these events (Start, Screensaver, Quit); see [Stand-Alone Mode](#).

A full description on how to use this application is provided in a later [section](#) of this document.

Configuring Other Emulators (not MAME)

This is a manual process (or you can use RocketBlinky, see below). Since there is no way for LEDBlinky to know the control-input mapping (button assignments), or specific controls, or button colors for each emulator or individual game, you must provide the information manually using the LEDBlinky Controls Editor.

LEDBlinky and the Controls Editor provide a number of features to ease the configuration of emulators and games. Each time you play an unknown game (one for which no unique controls are defined), the Emulator and ROM/Game name is stored. From the Controls Editor import menu, you can display the list of unknown games and select which you wish to import. You can then define the controls. Controls for player 1 can be copied to players 2, 3, and 4. If the controls for one game are similar to another, you can copy the entire ROM/Game.

It is not necessary to define the controls for every game – using the Controls Editor you can define a default set of controls for each emulator (or even a default for all emulators).

A full description on how to use the LEDBlinky Controls Editor application is provided in a later [section](#) of this document.

LEDBlinky includes an optional third-party tool called RocketBlinky that can be used to auto-generate an LEDBlinky Controls file that supports over 160 systems. Please see the Readme.txt in the RocketBlinky folder for installation and support options.

It's Not Working, What Should I Do?

If LEDBlinky is missing critical information such as a required file or configuration value, a message will be displayed after you exit the application and provide you with the option to display the log file. LEDBlinky logs all errors to the *LEDBlinky.log* file in the *LEDBlinky* folder. These types of problems can be easily resolved by providing the missing files or configuration values.

If LEDBlinky is not displaying any errors, but it's still not working as expected, the first place to look for help is the [LEDBlinky Support](#) page.

Next, try using the LEDBlinky Troubleshooting application. The troubleshooting app (*LEDBlinkyTroubleshooter.exe*) will attempt to provide solutions for common issues such as the wrong buttons lighting up during game play. The app is self-explanatory and may provide a solution to your issue.

As another option, you can post questions to the Software group on the [ArcadeControls.com](#) forum. The current LEDBlinky support thread can be found [here](#). If you start a new thread, please include the word "LEDBlinky" in the subject.

To diagnose complex problems, LEDBlinky can generate a *Debug.log* file. Debug mode can be enabled from the LEDBlinky Configuration Application on the "Misc Options" tab. Additional *Debug_*.log* files will also be generated, all of which will be zipped into the *Debug.zip* file. Using the Debug mode may degrade performance and should only be enabled to actively diagnose a problem.

If you wish to contact me directly, please post a personal message to "[arzo0](#)" on the ArcadeControls.com forum. You can also reach me via email at arzo0@LEDBlinky.net.

If you would like my help diagnosing a problem, please follow these steps:

- 1) From the LEDBlinky Configuration app, check the "Enable Debug Log" option on the Misc Options tab.
- 2) Start your front-end and run through the process that's not working – like start a few games. Then quit your front-end.
- 3) Locate the *Debug.zip* file in the LEDBlinky folder and email it to me along with an explanation of what's not working.
- 4) Don't forget to turn off the "Enable Debug Log" option when you're done.

Running LEDBlinky in Stand-Alone (Command Line) Mode

Any front-end that supports the ability to launch an external application and pass parameters (e.g. Rom name) can use LEDBlinky. The LEDBlinky stand-alone program file is *LEDBlinky.exe*.

If your FE cannot launch an external application prior to starting a game, you can still use LEDBlinky with MAME. From the LEDBlinky configuration app, check the “Use MAME to Trigger the Game Start/Stop Events” option on the “MAME Config” tab. Then launch LEDBlinky with the FE Start parameter (see below) prior to starting your FE. You can even use this option without any FE - just using MAME or any variant that supports MAME output messages.

You should only enable the LEDBlinky features for events that your front-end supports. For example, if your front-end can launch *LEDBlinky.exe* when the front-end first starts, then you can use the “FE Startup Animation” and other Startup options. If your front-end can launch *LEDBlinky.exe* when it starts and stops a screensaver, then you can use the LEDBlinky screensaver options.

The LEDBlinky application can be launched multiple times (once for each event command). Don't worry, only one instance will remain in system memory.

LEDBlinky command line syntax:

FE Start

```
LEDBlinky.exe 1
```

FE Quit

```
LEDBlinky.exe 2
```

Game Start

```
LEDBlinky.exe <rom>  
LEDBlinky.exe <rom> <emulator>
```

Note: Use double quotes around parameters if the parameter values include spaces.

Note: If you start a game without specifying the emulator, the last selected emulator will be used, or MAME is the default.

Game Stop

```
LEDBlinky.exe 4
```

Screensaver Start

```
LEDBlinky.exe 5
```

Screensaver Stop

```
LEDBlinky.exe 6
```

List Change

```
LEDBlinky.exe 8
LEDBlinky.exe 8 <emulator>
```

Note: Use double quotes around parameters if the parameter values include spaces.

Note: If you don't specify the emulator, the last selected emulator will be used, or MAME is the default.

Animation Start

```
LEDBlinky.exe <animation>.lwax <option> <option>
LEDBlinky.exe random.lwax
LEDBlinky.exe audio.lwax <animation>.lwax
```

Note: <option> parameter can be SingleLoop (or S) or NoClear (or N). 'SingleLoop' runs the animation from beginning to end one time. 'NoClear' keeps any LEDs in their current state prior to running the animation (does not turn off all LEDs).

Note: For Audio Animations, the second parameter specifies the animation file used for the first frame.

Note: Animation files must reside in the LEDBlinky\lwa folder.

Animation Stop

```
LEDBlinky.exe 11 <option>
```

Note: <option> parameter can be NoClear (or N). 'NoClear' keeps any LEDs in their current state after stopping the animation (LEDs remain in last frame state).

Load MAME Controller File

```
LEDBlinky.exe 12 <controller file>
```

Note: If the filename is specified without the full path, the controller file must reside in the MAME cfg folder.

Note: LEDBlinky will only load values from the controller file that match the MAME defined XML schema (the same as any MAME cfg file).

Note: Input mappings defined in the default.cfg or <rom>.cfg files will take precedence over any mappings defined in the controller file.

Reset Ultrastik 360 (U360)

```
LEDBlinky.exe 13
LEDBlinky.exe 13 <map name>
```

Note: The Map Name refers to the .um files in the /jdr folder. Typical values: analog, joy2way, joy4way, joy8way, joydiag, vjoy2way. Default is joy8way.

Note: If no U360 joysticks are connected when LEDBlinky first starts, this command can be used to rescan/reinitialize the u360 joystick(s) – for example, when hot-swapping control panels.

Set Port(s)

```
LEDBlinky.exe 14 <port>,<intensity>
LEDBlinky.exe 14 <port label>,<R/G/B/S>,<intensity>
LEDBlinky.exe 14 <controller type>,<controller id>,<port>,<intensity>
```

Note: Intensity values are 0 – 48 (max). If an LED Controller supports a greater range of values (0 – 255 for example), the 0 – 48 value will be proportionally adjusted.

Note: The Controller Type can be specified as the LED Controller Name or 2 character abbreviation. Use the LEDBlinky Configuration [Help Menu](#) for a list of supported LED Controller names and abbreviations. Example: LEDWiz or LW, PACLED64 or PL, IPACUltimateIO or IP, PACDrive or PD, Howler or HO.

Note: If only the Port and Intensity is specified, the value will be set on all connected LED Controllers.

Note: When specifying more than one command, use the bar (|) symbol to separate each command.

Set Port(s) Examples:

```
LEDBlinky.exe 14 1,48
```

Sets port 1 to an intensity of 48 on all connected LED controllers.

```
LEDBlinky.exe 14 1,48|2,30
```

Sets port 1 to an intensity of 48 on all connected LED controllers.

Sets port 2 to an intensity of 30 on all connected LED controllers.

```
LEDBlinky.exe 14 BUTTON1,R,0|BUTTON1,G,48|BUTTON1,B,0
```

Sets port with label "BUTTON1" and LED Type "Red" to an intensity of 0.

Sets port with label "BUTTON1" and LED Type "Green" to an intensity of 48.

Sets port with label "BUTTON1" and LED Type "Blue" to an intensity of 0.

```
LEDBlinky.exe 14 LW,2,1,48|LEDWIZ,2,2,20
```

Sets port 1 to an intensity of 48 on LEDWiz ID2.

Sets port 2 to an intensity of 20 on LEDWiz ID2.

```
LEDBlinky.exe 14 1,48|TRACKBALL,S,20|HOWLER,1,96,10
```

Sets port 1 to an intensity of 48 on all connected LED controllers.

Sets port with label "TRACKBALL" and LED Type "Single" to an intensity of 20.

Sets port 96 to an intensity of 10 on Howler ID1.

Set Game (Light game controls)

```
LEDBlinky.exe 15
```

```
LEDBlinky.exe 15 <rom>
```

```
LEDBlinky.exe 15 <rom> <emulator>
```

Note: Set Game is the same as "Game Start" but only lights the controls (no other [Game Start](#) options are run).

Note: Use double quotes around parameters if the parameter values include spaces.

Note: If you don't specify the rom, the last game started will be used. If you don't specify the emulator, the last selected emulator will be used, or MAME is the default.

LEDBlinky Core Application

The LEDBlinky core application runs in the Windows system tray. It accepts commands using command-line parameters (see [above](#)) or via a plug-in interface.

To display the LEDBlinky version and registration information, right-click on the tray icon  and select “About LEDBlinky”.

Should you wish to close the LEDBlinky application, right-click on the tray icon  and select “Exit LEDBlinky”.

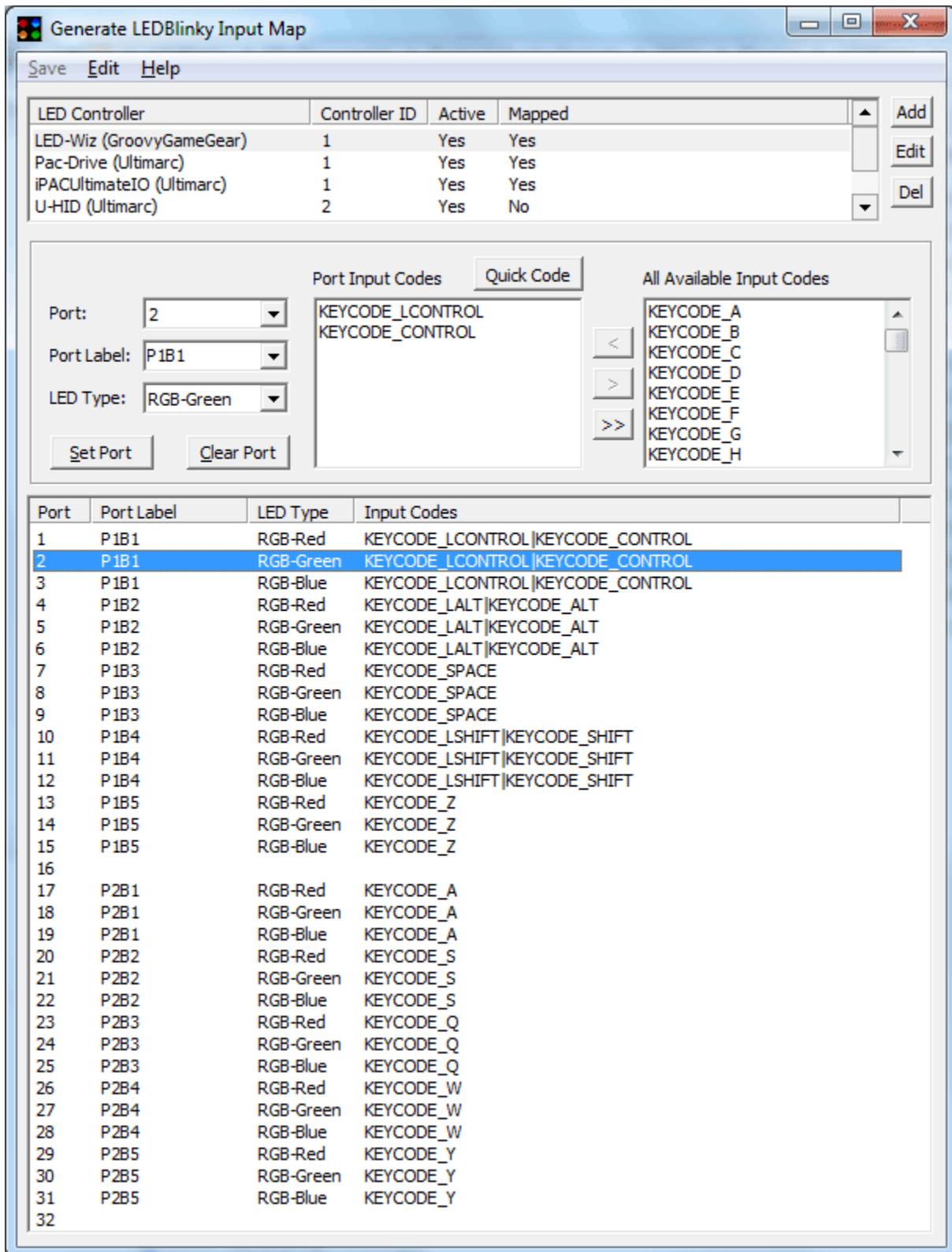
The LEDBlinky application can be launched multiple times but only one instance will remain in system memory.

Generate LEDBlinky Input Map Application

The input map defines the relationship between each wired port on your LED controller(s), and the keyboard or joystick input code for a Button or input codes for other controls (Joysticks, Trackball, etc.). This is a fixed relationship and should never change unless you rewire your LED controller(s), or rewire your input encoder, or reassign your input encoder values.

Think about it this way – Each LED button on your control panel is physically wired to a port on the LED controller. And each LED button is also physically wired to a port on your input encoder and assigned a keyboard or joystick value. The input map file ties the LED controller port to the assigned keyboard or joystick input code. LEDs for other controls (Joysticks, Trackball, etc.) may also be wired to the LED controller and these also have specific input codes.

The input map also defines a Port Label and LED Type for each wired port. The Port Labels serve two purposes, they tie together three ports for RGB LEDs, and they provide an easy reference to the ports from within the other LEDBlinky tools. The LED Type defines LEDs as Single or Red, Green, Blue for RGB.



Here's the general process for creating your input map. Each application feature is described in greater detail below.

1. Select an LED Controller. If your LED Controller is not currently active (connected to the PC), you can add it manually.
2. Select a Port. Optionally, you can click the port row on the lower list. The selected LED will light up on your control panel.
3. Select a Port Label from the list *or type your own*.
4. Select the LED Type (Single, Red, Green, or Blue).
5. Select one or more Input Codes from the list of all available Input Codes on the right. Optionally, you can click the "Quick Code" button and then press any button on your control panel or key on the keyboard to assign the Input Code.
6. Click the "Set Port" button.
7. Repeat for next Port. For RGB LEDs you can right-click a port, copy, and then paste.
8. Click the "Save" button.

LED Controller List

The top pane displays a list of all active and/or mapped LED controllers. Each LED controller has an associated ID (assigned from the manufacturer or set via the manufacturer provided software). An LED controller is considered Active when it is currently attached (via USB port) to your computer. An LED controller is considered mapped when you have defined a Port Label and LED Type for at least one port.

Note: No two LED controllers of the same type should have the same ID.

Add (LED Controller)

Manually add an LED controller. If your LED controller(s) are connected to your pc then they should already be listed and you should not need to add them. [LEDBlinky Output](#) (virtual LED Controller) must be added manually.

Edit (LED Controller ID)

Edit the LED controller ID. You should only need to change the LED controller ID when the ID is change in the controller firmware (via the manufacturer provided software). The "Edit" button is only enabled when a controller is selected.

Delete (LED Controller)

Delete an LED controller from the input map. You will be asked to confirm the delete action. The "Del" button is only enabled when a controller is selected.

Port

Each [LED controller](#) has a specific number of LED ports. Your control panel may not use all ports on each LED controller. A port can be selected from the drop-down list or by clicking on any port row in the lower pane.

Port Label

The port label serves two purposes – it ties together three ports for RGB LEDs, and it provides an easy reference to the ports from within the other LEDBlinky tools. You can select a port label from the drop-down list or enter your own label.

Note: You can enter your own label in the drop-down list.

When specifying the port label for an RGB LED, you must use the same label for the red, green, and blue ports. Port labels for single LEDs must be unique.

LED Type

The LED type specifies that a port is wired to a single color LED, or the red, green, or blue lead for an RGB LED.

Color Adjust

Color Adjust applies a global offset value (+/-) to the individual port so that color variations between buttons or other illuminated controls can be fine-tuned. The drop-down lists a range of positive and negative values based on the range of the current LED controller hardware. The selected value will be added or subtracted anytime the port is active.

Note: The Color Adjust drop-down list is only visible when Show Color Adjustments is selected from the Edit menu.

Port Input Codes

One or more input code can be assigned to each port. For LEDs that are illuminating a control that is wired to the keyboard encoder (usually buttons), you should only specify a single input code. For other controls not wired to the keyboard encoder (trackball, analog joysticks, spinners, etc.) you should specify all the input codes that could apply.

For example, in the sample screen shot above, five buttons have RGB LEDs, and the trackball has a single LED. Each of the buttons is assigned the same port label for all three colors (red, green, and blue), and the same input code. The trackball has its own unique port label and has been assigned the “TRACKBALL” input code. Analog controls such as a trackball can also be assigned analog input codes such as “MOUSECODE_1_ANALOG_X” and/or “MOUSECODE_1_ANALOG_Y”, etc.

Note: Assigning an input code to each port is not required if you are only creating LED animations for use with applications other than LEDBlinky.

Quick Code

Click this button to quickly select and assign a keyboard or joystick input code. When the button is depressed, it will flash ‘Press Button; then press any button on your control panel or key on the keyboard. The input code will be selected and added to the Port Input Codes list. Additionally, if the Port Label and LED Type have been specified and

validated, the values will be assigned to the Port (the same as clicking the “Set Port” button). To abort the Quick Code mode, click the button a second time.

Note: The Quick Code feature cannot be used to assign mouse, or gun codes – these must be selected manually from the list.

All Available Input Codes

This is the list of all possible input codes used by MAME. LEDBlinky also maps these input codes for use with MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, or RetroFE controls and other emulators.

Set Port

Click this button to assign the Port Label, LED Type, Input Code(s), and optional Color Adjust to the selected Port. Validation will confirm that the port label is unique for a single type LED or that the port label and color is unique for an RGB LED.

Clear Port

Clears the selected port.

LED Controller Port List

The lower pane displays a list of all ports for the selected LED controller (in the top pane). Selecting a port from the lower pane will fill all the port fields with any assigned data and light the associated LED (if wired).

Note: You can right-click any port in the lower pane to display the Copy/Paste menu. The “Copy Port” menu option is only enabled when an existing RGB port is selected. The “Paste Port” menu options are only enabled when a blank port is selected.

Note: The Color Adj. column is only visible when Show Color Adjustments is selected from the Edit menu.

Menus



Save

Saves the Input Map!

Edit

The Edit menu provides options to copy and paste port values for RGB LEDs. This allows you to quickly add the three ports required for each RGB LED.

1. Select a blank port wired to a RGB and set the port information (Port Label, LED Type, and Input Codes), and then click "Set Port".
2. From the Edit menu, click "Copy Port". You can also right-click the port and select "Copy Port" from the pop-up menu.
3. Select the second blank port wired to the RGB. From the Edit menu, click "Paste Port" for the correct color. You can also right-click the port and select "Paste Port".
4. Repeat for the final blank port wired to the RGB.

Note: The "Copy Port" menu option is only enabled when an existing RGB port is selected. The "Paste Port" menu options are only enabled when a blank port is selected.

Edit / Show Color Adjustments

The Edit menu "Show Color Adjustments" option will display an additional [Color Adjust](#) field (under the LED Type field) and an additional Color Adj. column (to the right of the Input Codes column). Color Adjustments allow a global offset value (+/-) to be applied to individual ports so that color variations between buttons or other illuminated controls can be fine-tuned.

Edit / Keyboard ID

The Edit menu "Keyboard ID" option allows the selection of the device ID when more than one keyboard or keyboard encoder is in use. The keyboard ID should only be changed if you are using more than one keyboard or keyboard encoder and your emulator(s) support multiple keyboards (MAME for example).

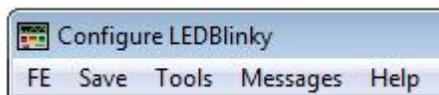
After selecting a Keyboard ID (other than the default), anytime you add a keyboard input code, the ID value will be included. LEDBlinky uses the syntax expected by MAME: `KEYCODE_<id>_<value>`. For example, if you are using Keyboard ID 2 and you add `KEYCODE_A`, the port input code will be `KEYCODE_2_A`.

Note: USB keyboards and keyboard encoders may change IDs depending on the order they are assigned by Windows.

LEDBlinky Configuration Application

The LEDBlinky Configuration application is used to configure all LEDBlinky features and options.

Menus



FE

Select your Front-End. Current options are “AtomicFE”, “Attract-Mode”, “CoinOps”, “GameEx”, “HyperSpin”, “LaunchBox”, “MaLa”, “Maximus Arcade”, “PinballX”, “RetroFE”, and “Other”. The MaLa tab, AtomicFE tab, GameEx tab, PinballX tab, HyperSpin tab, LaunchBox tab, Maximus tab, Attract-Mode tab, CoinOps tab, and RetroFE tab will be visible or hidden depending on the front-end selected.

Save

Save the current configuration. The LEDBlinky configuration file is *settings.ini* located in the LEDBlinky folder. The “Save” menu option is only enabled when one or more options have been modified. If you attempt to close the Configuration app prior to saving, you will be prompted to save the data.

Tools

Generate Input Map

This button runs the Generate LEDBlinky Input Map tool . You can also run this app directly - *GenLEDBlinkyInputMap.exe*. Before you can use the LEDBlinky plug-in, you must create an input map. A full description on how to use this application is provided in a later [section](#) of this document.

Controls Editor

This button runs the LEDBlinky Controls Editor . You can also run this app directly – *LEDBlinkyControlsEditor.exe*. A full description on how to use this application is provided in a later [section](#) of this document.

LED Animation Editor

This button runs the LEDBlinky Animation Editor . You can also run this app directly - *LEDBlinkyAnimationEditor.exe*. See the *AnimationEditor.pdf* file for detailed instructions on using the animation editor.

MAME Output Test

This button runs the MAME Output Test application . Use the MAME Output Test to view which outputs are generated by a MAME game and which controls (if any) are associated with the outputs.

Start the test app, then start MAME or MAME32, then start any game – the outputs and their associated controls will be listed whenever their state changes; On/Off.

Warning: You must start the test app prior to starting MAME or MAME32. If you start MAME or MAME32 first, it will most likely crash when the test app is started.

Messages

Check Now

Check server for new version or other messages. Requires internet access.

Check at Startup

Toggle option to check for messages when starting the LEDBlinky Configuration application. Requires internet access.

Note: LEDBlinky will always check for messages which can be viewed on the “About LEDBlinky” screen (right-click LEDBlinky icon in system tray). A message can be cleared by running the LEDBlinky Configuration Application with the “Check at Startup” option enabled, and then clicking the “Delete” button when the message is displayed.

Help

Installation and Configuration

Launches the LEDBlinky help documentation – this document!

Supported LED Controllers

Displays the names and abbreviations of all LED Controllers (hardware) [supported](#) by LEDBlinky.

Note: When using the LEDBlinky [command-line](#) mode to manually set one or more ports, the LED Controller parameter must be specified by its name or abbreviation exactly as listed here.

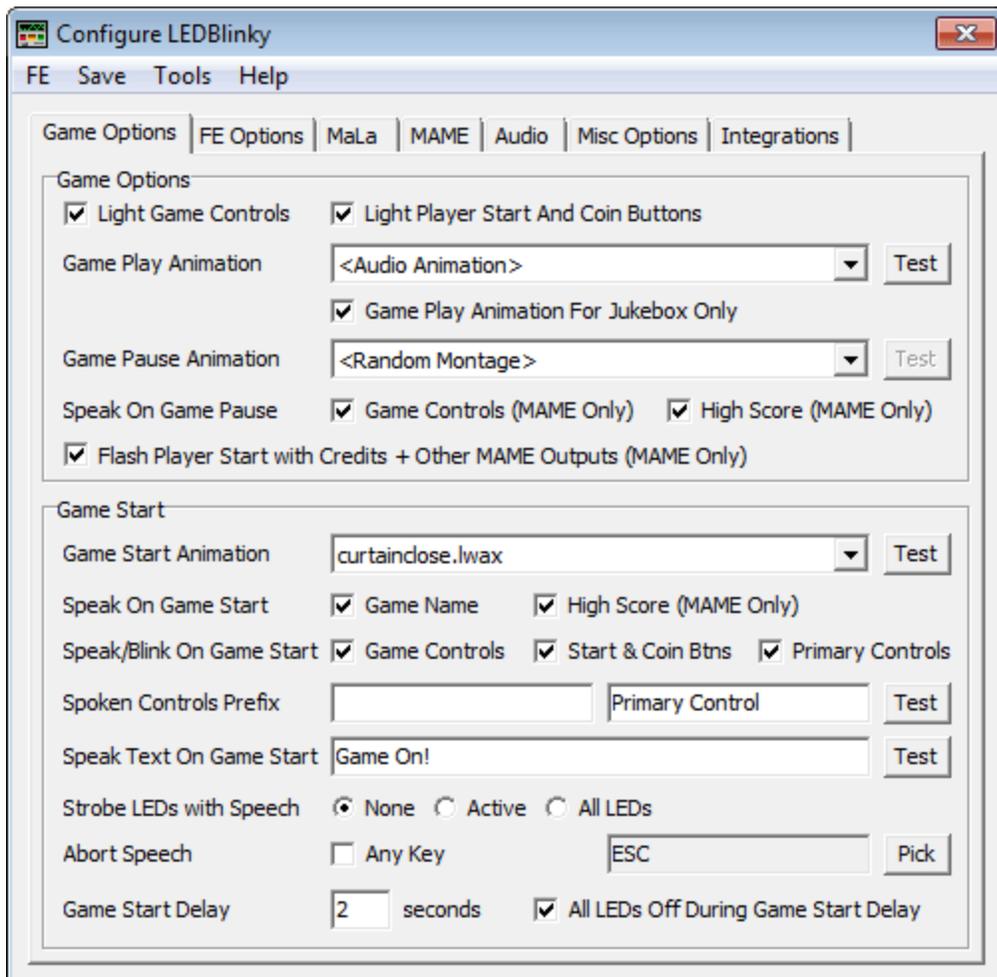
Support Key

Generates a single-use support key which may be necessary when requesting LEDBlinky support. Copy the value and paste into your email.

About

Displays the LEDBlinky and LEDBlinky support application’s version numbers.

Game Options



Light Game Controls

Check this box to light the controls used by MAME or any other emulator.

Light Player Start And Coin Buttons

Check this box to light the player Start and Coin buttons during game play and when using the MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, or RetroFE Demo Game Controls feature. This option is only enabled when the Light Game Controls option is checked.

The *controls.ini* file does not provide the Start and Coin controls. With this option, Start and Coin controls will be included for each game. For example, a two-player game will include Start1, Start2, Coin1, and Coin2.

Game Play Animation

Select the file name you wish to run when playing a game. The animation file list is populated with all *.lwax* files in the */lwa* folder. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

When <Random> is selected, a different file (from the list) will run each time a game is started. When <Random Montage> is selected, animations will be selected from the *lwa* folder and play a single loop one after the other - in effect randomly stringing all the animations together. When <Audio Animation> is selected, the actual animation along with other audio options must be selected on the [Audio Animation](#) tab.

Note: If the “Light Game Controls” option is checked, the animation will only affect non-active controls (controls not used by the selected game).

LEDBlinky can run any LWAX file generated using the LEDBLinky Animation Editor.

Game Play Animation For Jukebox Only

Check this option if you only want the Game Play Animation to run for emulators or games designated as a jukebox application. Use the Controls Editor [Jukebox](#) option to designate which Emulator or ROM/Game is a jukebox application.

Note: Any ROM/Game control group (including an emulator default control group) can be designated as a jukebox application for the purpose of this option. For example, let's assume you are using an audio animation for Game Play and the Game Play Animation For Jukebox Only option is checked. Using the Controls Editor you have designated your jukebox application. If you also have another game (not a jukebox) and you want the audio animation to run for that game, then just designate it as a jukebox!

Game Pause Animation

Select the file name you wish to run when pausing (P) a MAME game. The animation file list is populated with all *.lwax* files in the */lwa* folder. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

When <Random> is selected, a different file (from the list) will run each time MAME is paused. When <Random Montage> is selected, animations will be selected from the *lwa* folder and play a single loop one after the other - in effect randomly stringing all the animations together. When <Audio Animation> is selected, the actual animation along with other audio options must be selected on the [Audio Animation](#) tab.

Note: This feature only works for MAME version .118 or later.

If the “Speak and Blink Controls on Game Pause” option is checked, the game pause animation will start running after the control blinking is complete.

LEDBlinky can run any LWAX file generated using the LEDBLinky Animation Editor.

Speak On Game Pause; Game Controls (MAME Only)

Check this box to blink each button and have the Windows Text To Speech Synthesizer speak the button "action" when pausing (P) a MAME game. If you press pause a second time while the speech/blinking is still active, the feature will stop and game play will resume. Buttons and controls will be spoken based on the [Speak On Game Start; Game Controls](#) options selected.

Note: This feature only works for MAME version .118 or later.

Speak on Game Pause; High Score (MAME Only)

Check this box to have the Windows Text To Speech Synthesizer speak the top high score and initials when pausing (P) a MAME game.

Note: This feature only works for MAME version .118 or later.

Flash Player Start with Credits + Other MAME Outputs (MAME Only)

Check this to use MAME Outputs. MAME Outputs are game dependent – some games use them and some don't. The most common outputs are LED0 and LED1 which are most often used to flash the player start buttons when credits are available.

LEDBlinky uses the *MameOutputs.ini* file to map outputs to controls or LED controller ports. By default, LED0 and LED1 are mapped to Start1 and Start2 – if your player start buttons have LEDs, they will flash when credits are available for supported games (for example; Asteroids). Any other outputs you wish to use must be manually added to the *MameOutputs.ini* file.

MAME Outputs can also be used to trigger Pixelcade commands or execute a Web Request (http or https GET).

See the [MAME Output System](#) or the *MameOutputs.ini* file for a full details on how to configure MAME Outputs for use with LEDBlinky.

Note: This feature only works for MAME version .112 or later.

Game Start Animation

Select the file name you wish to run before the selected game starts. When <Random> is selected, a different file (from the list) will run each time a game starts. Only a single loop of the LED animation will be played (one pass through the file). The animation file list is populated with all *.lwx* files in the *lwa* folder. LEDBlinky can run any LWAX file generated using the LEDBlinky Animation Editor. Click the "Test" button to run the selected animation, and click the button a second time to stop the animation.

Speak On Game Start; Game Name

Check this box to have the Windows Text To Speech Synthesizer speak the game name when starting a game.

Speak On Game Start; High Score

Check this box to have the Windows Text To Speech Synthesizer speak the top high score and initials when starting a game.

Note: High Scores only works for MAME games and not all MAME games support high scores.

LEDBlinky uses the Hi2txt app to decode MAME high score files. The version of Hi2txt included with the LEDBlinky Installation may not be the latest version. For Hi2txt support and to download the latest C# version and database of supported games see; <https://greatstoneex.github.io/hi2txt-doc/>

Additionally, it may be necessary to enable high score functionality in MAME. Newer versions of MAME (0.172+) support high scores natively. For older versions of MAME, you may need a high score plugin. Please see Hi2txt and MAME documentation for more information.

Speak/Blink On Game Start; Game Controls

Check this box to blink each button and have the Windows Text To Speech Synthesizer speak the button "action" when starting a game.

Speak/Blinky On Game Start; Start and Coin Btns

Check this box to blink the Start and Coin buttons and have the Windows Text To Speech Synthesizer speak the button "action" when starting a game. This option is only enabled when the Speak/Blink On Game Start Game Controls option is checked and the [Light Game Controls](#) option is not checked or both the Light Game Controls and [Light Player Start And Coin Buttons](#) options are both checked. Basically, you can't light the Start and Coin buttons without also lighting the other game controls. And you can't speak the Start and Coin buttons without also speaking the other game controls.

Speak/Blink On Game Start; Primary Controls

Check this box to have the Windows Text To Speech Synthesizer speak the primary control(s) when starting a game. The primary controls for each MAME game are defined in the *controls.ini* file by the "P1Controls" tag.

P1Controls example:

```
P1Controls=8-way Triggerstick+joy8way+P1_BUTTON1|Spinner+dial
```

In this example (for Tron), the primary controls are an 8-way joystick and a spinner. LEDBlinky uses the values to the left of the plus (+) for the spoken text. For Tron the spoken text will be; "8-way Tiggerstick and Spinner".

Primary controls for other emulators are defined using the LEDBlinky Controls Editor. The Controls Editor can also be used to override the values in the *controls.ini* file (for MAME).

Spoken Controls Prefix

The Spoken Controls Prefix text will be spoken before the button “actions” are blinked and spoken. The Spoken Primary Controls Prefix text will be spoken before the primary controls are spoken (after the buttons). Click the “Test” button to hear how the text will sound using the current [Text To Speech](#) Options.

Speak Text On Game Start

Enter any text you wish to have spoken prior to starting a game. Click the “Test” button to hear how the text will sound using the current [Text To Speech](#) Options.

Strobe LEDs with Speech

Select which LEDs (None, Active for selected game, or All) will blink as the Game Start, High Score, and Game Start Text messages are spoken. The LEDs will blink on an off based on the phonetics of the speech.

Abort Speech

You may abort the game start speech (game name, button actions, primary controls, etc.) by pressing any button or a selected button. You may also abort the MAME pause speech (button actions).

To select a specific Abort Speech button, uncheck “Any Key” and click the “Pick” button, then press any button on your control panel.

Game Start Delay (seconds)

The Game Start Delay will cause LEDBlinky to wait the specified time (in seconds) before continuing with any configured game start options (listed above). This can be used to allow the selected emulator/game to start before activating any LEDs. The maximum delay is 999 seconds.

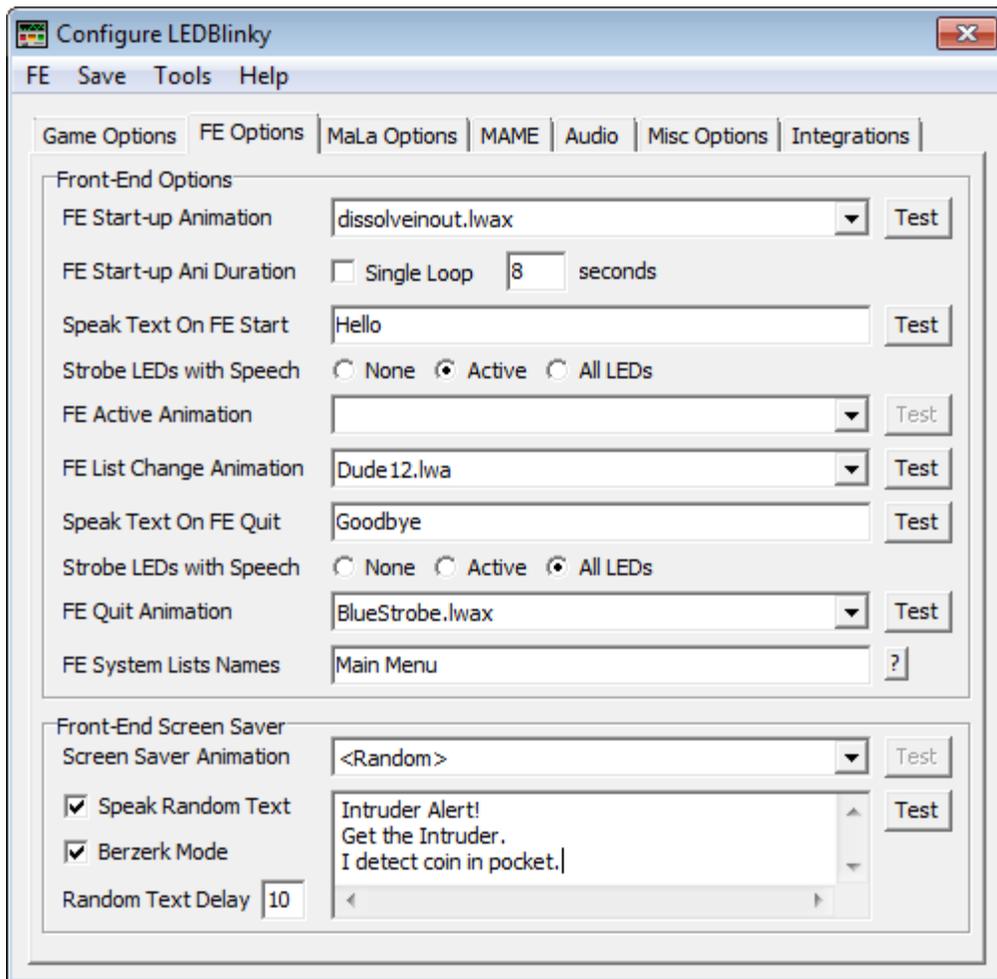
Note: Normally the Game Start Delay is set to zero (0).

Note: If another event (Game Quit for example) occurs during the Game Start Delay, the Game Start event will exit immediately and none of the features will activate.

All LEDs Off During Game Start Delay

With this option checked, any LEDs on when the Game Start event occurs will turn off. If the option is unchecked, any LEDs on when the Game Start event occurs will remain on for the duration of the game start delay. For example, if a FE Active Animation is running, it will continue to run until the game start delay is complete. The All LEDs Off During Game Start Delay is only enabled when a value greater than zero (0) is specified for the Game Start Delay.

FE Options



FE Start-up Animation

Select the file name you wish to run when the front-end first starts. When <Random> is selected, a different file (from the list) will run each time the front-end is started. Only a single loop of the LED animation will be played (one pass through the file). The animation file list is populated with all `.lwax` files in the `/lwa` folder. LEDBlinky can run any LWAX file generated using the LEDBlinky Animation Editor. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation. When <Audio Animation> is selected, the actual animation along with other audio options must be selected on the [Audio Animation](#) tab.

Note: During the FE Start-Up Animation, other events may queue up. These events will execute in order after the animation completes. A maximum of 10 events can be queued after which events will be dropped.

Note: Audio Animation cannot be used in conjunction with Single Loop Duration.

FE Start-Up Ani(mation) Duration

The start-up animation can run for a single loop of the animation or it can run for a specified duration (in seconds). Running the animation for a specified duration can be useful when you wish to sync the LED animation with a front-end video animation. The “FE Start-Up Animation Duration” options are only enabled then a “FE Start-Up Animation” has been selected. The maximum duration is 999 seconds.

Note: The FE Start-Up Ani Duration also applies to the [Cabinet LEDs Start Animation](#).

Speak Text On FE Start

Enter any text you wish to have spoken when the front-end first starts. Click the “Test” button to hear how the text will sound using the current [Text To Speech](#) Options.

Strobe LEDs with Speech

Select which LEDs (None, Active for selected game, or All) will blink as the front-end Start message is spoken. The LEDs will blink on an off based on the phonetics of the speech.

FE Active Animation

Select the file name you wish to run while the front-end is active (not during game play). When <Random> is selected, a different file (from the list) will run each time the front-end starts. The animation file list is populated with all *.lwax* files in the */wa* folder. LEDBlinky can run any LWAX file generated using the LEDBLinky Animation Editor. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

When <Audio Animation> is selected, the actual animation along with other audio options must be selected on the [Audio Animation](#) tab.

FE List Change Animation

Select the file name you wish to run each time the FE changes Emulators or Game lists. When <Random> is selected, a different file (from the list) will run each time the front-end is started. Only a single loop of the LED animation will be played (one pass through the file). The animation file list is populated with all *.lwax* files in the */wa* folder. LEDBlinky can run any LWAX file generated using the LEDBLinky Animation Editor. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

Note: This feature may not be fully supported by every front-end. For example, some front-ends may only play the list change animation when switching emulators, but not for game lists or category lists.

Speak Text On FE Quit

Enter any text you wish to have spoken when quitting the front-end. Click the “Test” button to hear how the text will sound using the current [Text To Speech](#) Options.

Strobe LEDs with Speech

Select which LEDs (None, Active for selected game, or All) will blink as the front-end Quit message is spoken. The LEDs will blink on an off based on the phonetics of the speech.

FE Quit Animation

Select the file name you wish to run when quitting the front-end. When <Random> is selected, a different file (from the list) will run each time the front-end exits. Only a single loop of the LED animation will be played (one pass through the file). The animation file list is populated with all `.lwax` files in the `/lwa` folder. LEDBlinky can run any LWAX file generated using the LEDBlinky Animation Editor. Click the "Test" button to run the selected animation, and click the button a second time to stop the animation.

FE System Lists Names

Front-End software typically displays two types of selection lists;

1. Lists of Games grouped by System. Additional sub-groups may be filtered by game attributes.
2. Lists of Systems (emulators). This is often the Main Menu.

When selecting an item from a list, the front-end will send a Game Selected event to LEDBlinky. The Game Selected event will include the "List Name" and the "Selected Item Name" as event parameters. LEDBlinky uses the "List Name" to identify the current System (emulator), and the "Selected Item Name" to identify the selected Game/ROM. This logic works fine for lists of Games grouped by System, but not for lists of Systems (e.g. Main Menu).

The "FE System Lists Names" field is used to provide a comma separated list of names of your FE System lists. LEDBlinky will use these values to identify when a System List is displayed and will use the selected item as the current System/Emulator (rather than the selected Game/ROM).

Names must be spelled exactly the same as the value passed by the Front-End. Capitalization is not important (case-insensitive). Multiple names must be separated with commas.

Screen Saver Animation

Select the file name you wish to run when the front-end is in screensaver mode. The animation file list is populated with all `.lwax` files in the `/lwa` folder. Click the "Test" button to run the selected animation, and click the button a second time to stop the animation.

When <Random> is selected, a different file (from the list) will run each time the front-end enters screensaver mode. When <Random Montage> is selected, animations will be selected from the `/lwa` folder and play a single loop one after the other - in effect randomly stringing all the animations together. When <Audio Animation> is selected, the actual animation along with other audio options must be selected on the [Audio Animation](#) tab.

LEDBlinky can run any LWAX file generated using the LEDBLinky Animation Editor. LED Animation files can also be generated using the LEDBLinky Animation Editor or manually.

Speak Random Text

Enter any text you wish to have spoken when the front-end is in screensaver mode. One line will be spoken randomly at predefined intervals (Random Text Delay). Click the “Test” button to hear how the text will sound using the current [Text To Speech](#) Options.

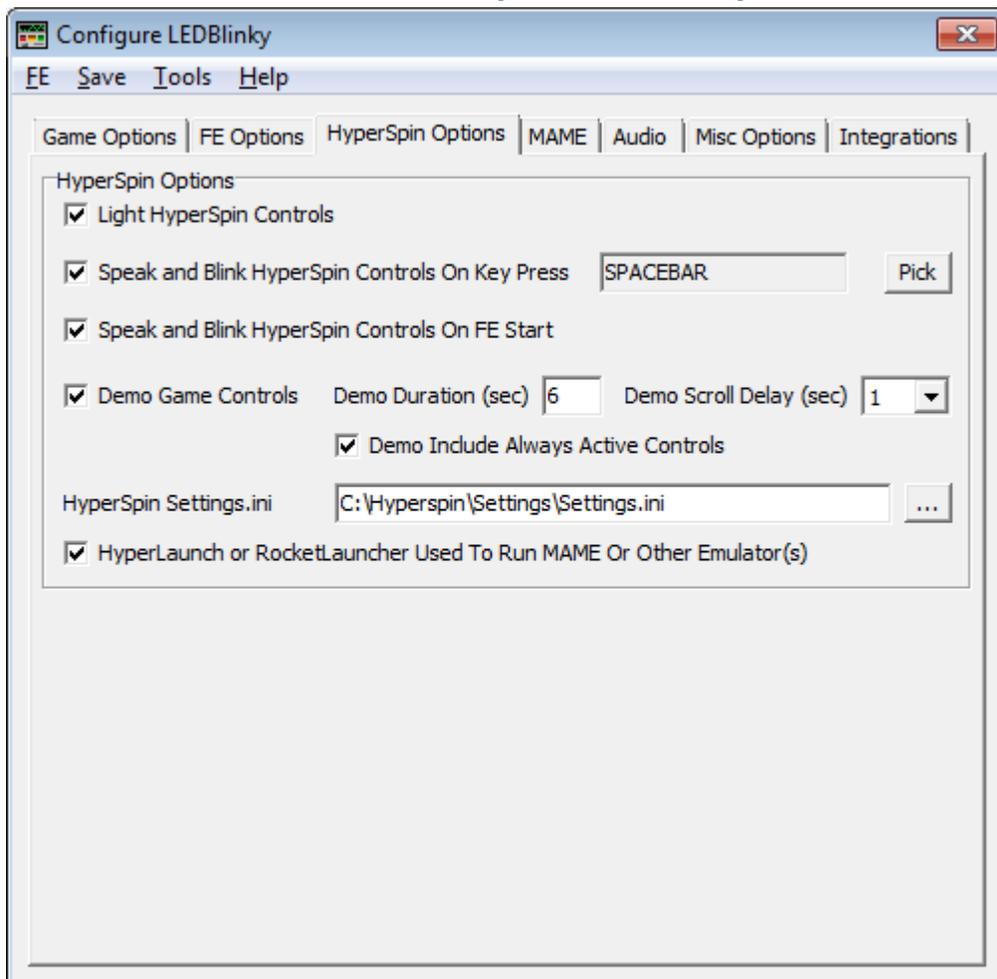
Berzerk Mode

With Berzerk Mode enabled, the speech rate will be randomly varied as each line of random text is spoken. For the best effect, use the Microsoft Sam voice (Windows XP only) which sounds similar to the original Berzerk voice!

Random Text Delay (seconds)

Enter the interval in seconds between random text messages. For example, a value of 60 will play a line of random text every minute while the screen saver is active.

MaLa / AtomicFE / GameEx / PinballX / HyperSpin / LaunchBox / Maximus Arcade / Attract-Mode / CoinOps / RetroFE Options



- Light MaLa Controls**
- Light AtomicFE Controls**
- Light GameEx Controls**
- Light PinballX Controls**
- Light HyperSpin Controls**
- Light LaunchBox Controls**
- Light Maximus Arcade Controls**
- Light Attract-Mode Controls**
- Light CoinOps Controls**
- Light RetroFE Controls**

Check this box if you wish to light the controls used by the MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, or RetroFE user interface.

Note: This feature is only useful if you have MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, or RetroFE controls mapped to buttons with LEDs.

Note: For GameEx, you must have “Enable Custom Keyboard Controls” = Yes in the GameEx Input settings.

Demo Game Controls

Check this box if you wish to briefly light the controls used by each game as you scroll through the MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, or RetroFE game lists.

Demo Duration (seconds)

The demo duration determines how many seconds a game’s controls will be lit while scrolling through the game list. For example, if the demo duration is set to 5 and you stop scrolling on Asteroids, the Asteroids buttons will light up for 5 seconds, after which the buttons return to the FE controls or turn off (depending on the selected feature).

Demo Scroll Delay (seconds)

The demo scroll delay is the time between when you stop scrolling and when the game’s controls light up. When set to zero, the game’s controls will light immediately as you stop on each game, but this may affect the scroll performance on slower CPUs.

Demo Include Always Active Controls

Include controls marked as [Always Active](#) when lighting controls for demo mode.

Speak And Blink MaLa Controls

Speak And Blink AtomicFE Controls

Speak And Blink GameEx Controls

Speak And Blink PinballX Controls

Speak And Blink HyperSpin Controls

Speak And Blink LaunchBox Controls

Speak And Blink Maximus Controls

Speak And Blinky Attract-Mode Controls

Speak And Blinky CoinOps Controls

Speak And Blinky RetroFE Controls

Check this box if you wish blink each button and have the Windows Text To Speech Synthesizer speak the button "action" when the FE is active.

To select the button (key) to active this feature, click the “Pick” button, then press any button on your control panel. The same button (key) can also be used to abort the speech.

HyperSpin Settings.ini

For HyperSpin you must select the location of the *settings.ini* file (usually located in the HyperSpin *Settings* folder).

Maximus Default.ini

For Maximus Arcade you must select the location of the *default.ini* file (usually located in the Maximus Arcade *Preferences* folder).

Note: For front-ends where LEDBlinky is installed into the front-end “plugins” folder, LEDBlinky will attempt to locate the front-end setting/configuration file. If the setting/configuration file cannot be located, you will need to provide the file path on the <FE> Options tab.

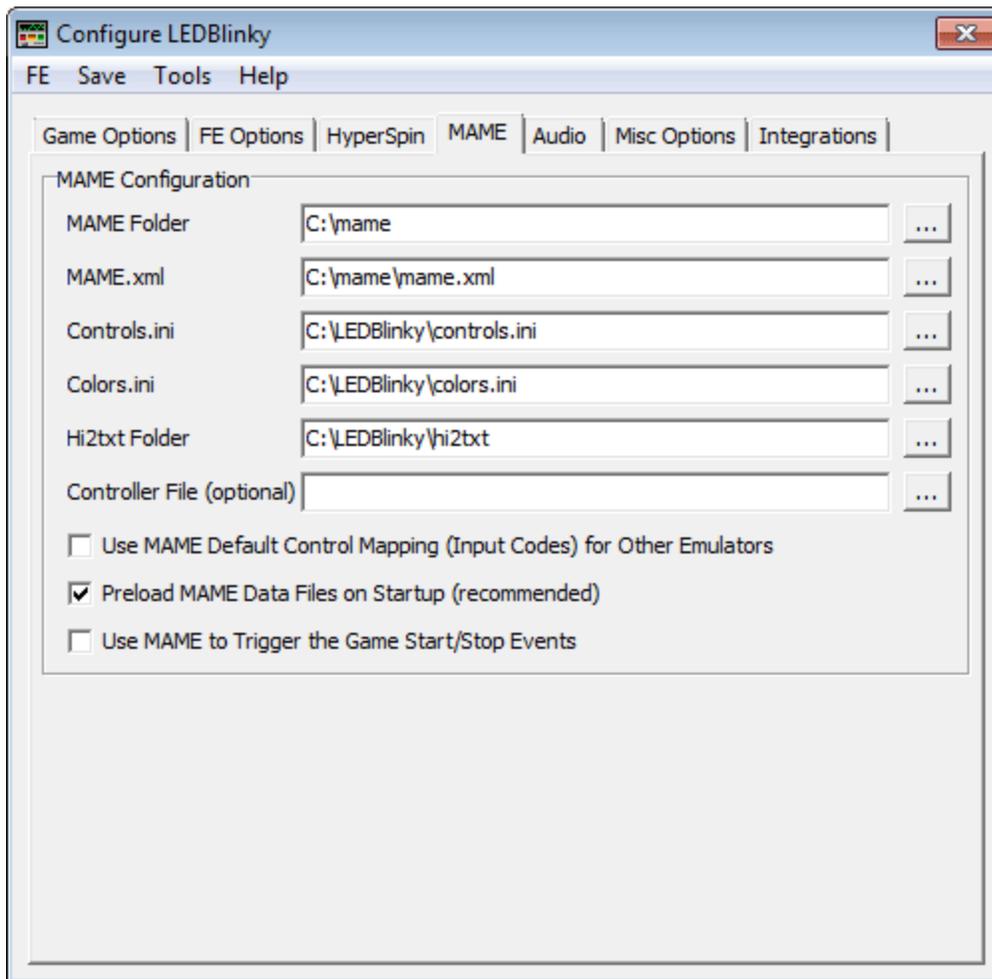
HyperLaunch or RocketLauncher Used To Run MAME Or Other Emulator(s)

When using HyperLaunch (version 2.0 or later) or RocketLauncher with HyperSpin to run one or more emulators (including MAME), this option should be checked. With this option enabled, LEDBlinky will ignore any invalid “Game Quit” commands sent from HyperSpin while HyperLaunch or RocketLauncher is active (in memory).

GameEx Screen Saver Starts Random Game**CoinOps Screen Saver Selects Random Game****RetroFE Screen Saver Selects Random Game**

GameEx, CoinOps, or RetroFE can be configured to randomly select or start games while the screen saver is active. Normally LEDBlinky will ignore these game select/starts and continue to run whichever (if any) [Front-End Screen Saver options](#) are enabled. With this option checked, LEDBlinky will respond to the random game select/starts and demo or run all configured [Game Options](#) (Game Start Animation, Speak/Blink On Game Start, Light Controls, etc.).

MAME



MAME Folder

Click the folder browse button  to select the folder where MAME is installed. You must specify this folder to light MAME controls and use other MAME related features.

MAME.xml

Click the file browse button  to select the *MAME.xml*. This file is generated by MAME using the `-listxml` command. You must specify this file to light MAME controls. This field is not visible (and not required) when using MaLa.

LEDBlinky will parse the *MAME.xml* file and create a minimized version. The initial parsing may add a delay when LEDBlinky first starts, but the next time LEDBlinky starts it will use the minimized version which should load very quickly.

Controls.ini

Click the file browse button  to select the *Controls.ini* file. You must specify this file to light MAME controls. A version of this file has been included with LEDBlinky and the path should be pre-populated.

Colors.ini

Click the file browse button  to select the *colors.ini* file. The *colors.ini* file is used to set game specific control colors (or intensities). A version of this file with authentic control panel colors supporting over 1200 ROMs (at last count) has been included with LEDBlinky and the path should be pre-populated

Note: Colors defined for an individual ROM/Game using the LEDBlinky Controls Editor will override the values in the *Colors.ini* file for that ROM/Game.

If you wish to add additional entries to the *Colors.ini* file, valid colors are defined in the *Color-RGB.ini* file. Valid intensities are 0 – 48, or the following built-in blinking effects;
129 = Ramp Up / Ramp Down
130 = On / Off
131 = On / Ramp Down
132 = Ramp Up / On

Hi2txt Folder

Click the folder browse button  to select the folder where Hi2txt is installed. Hi2txt is a third-party app included with the LEDBlinky installation and the path should be pre-populated. You must specify this folder to use any high score related features.

Controller File

Controller files are usually supplied by Control Panel vendors (xArcade, SlikStik, etc.). They provide the default input controls for their layout. If you use a controller file when running MAME, set the path to that file.

Note: Do not confuse the Controller file with the Controls.ini file.

Use MAME Default Control Mapping for Other Emulators

Unfortunately, there is no way for LEDBlinky to know the control input codes (keycodes) for each non-MAME emulator or individual game. In this regard, you have two choices – define the input codes manually using the [LEDBlinky Controls Editor](#), or use MAME's control-input map for all other emulators (this option).

If your control panel always uses the same buttons mapped to the same input codes (keycodes) for all emulators including MAME, then you may wish to check this option. Remapping any buttons in MAME for 'All Games' will also be reflected for other emulators. If any of your emulators use control mapping that differs from MAME, then do not use this option.

Note: This only affects the buttons that LEDBlinky lights up. The actual keycode assignments must be changed for each individual emulator via its configuration features.

Preload MAME Data Files On Startup

With this option enabled the following files will be parsed and loaded into memory when LEDBlinky first starts: *mame.xml*, *controls.ini*, and *colors.ini*. When the option is not enabled, the files are parsed and loaded the first time the emulator (MAME) is selected by the front-end software.

Parsing and loading the data at startup (before any animations or other features are running) can significantly decrease the load time. For example, a large *mame.xml* file can take upwards of 60 seconds to load while the FE Active Animation is running, but the same file can load in 5 seconds at startup. The downside to preloading at startup is that there may be a short delay (approximately 5 - 10 seconds) before anything will light up.

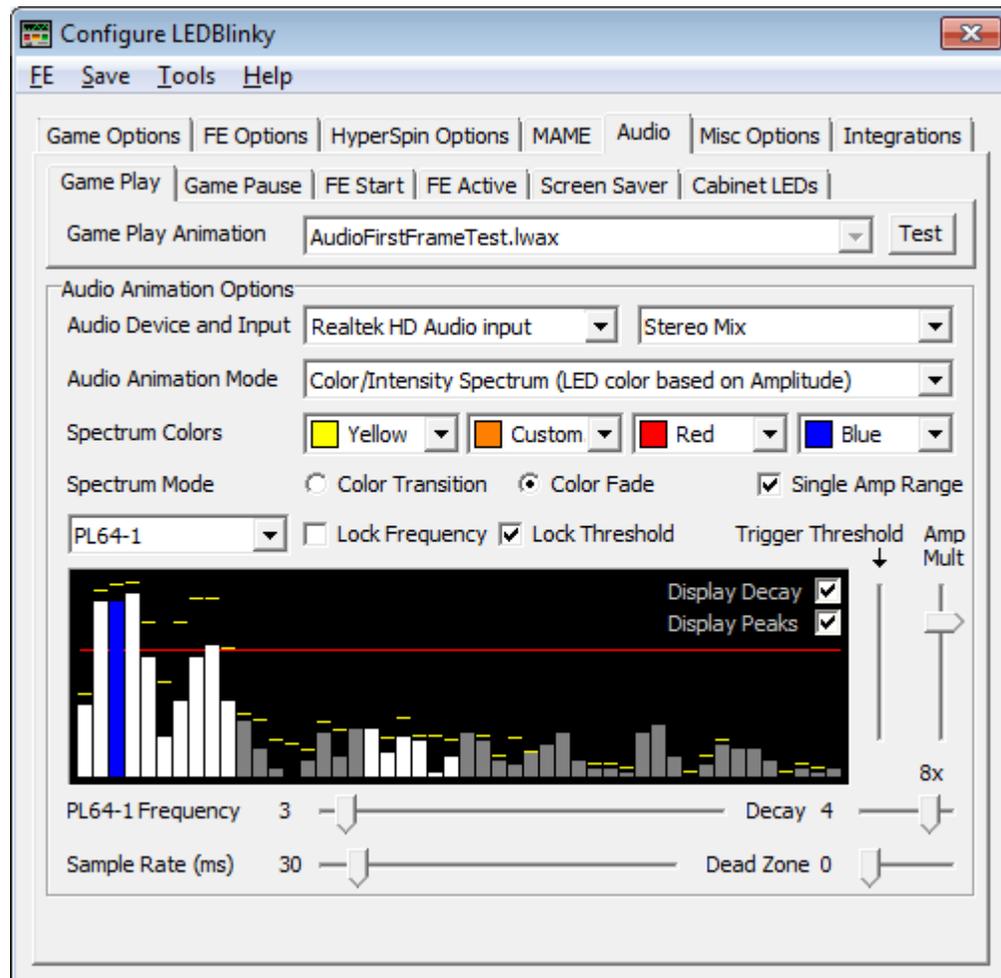
Note: If you are using MAME with your setup then it is recommended to preload the MAME files.

Use MAME to Trigger the Game Start/Stop Events

With this option checked, MAME output messages will be used to detect when a game (MAME only) is started and stopped. You can also use this option to launch LEDBlinky without any front-end, just using MAME or any MAME variant that supports MAME output messages.

Note: This option is only visible when "Other" is selected from the FE menu.

Audio



Audio Animations can be active when a game is played and/or paused, and/or when the FE is active, and/or when the FE screen saver is active, and/or for Cabinet LEDs.

Audio Animations are enabled by selecting <Audio Animation> from the following animation file drop-down lists:

- “Game Play Animation” on the “Game Options” tab.
- “Game Pause Animation” on the “Game Options” tab.
- “FE Active Animation” on the “FE Options” tab.
- “Screen Saver Animation” on the “FE Options” tab.
- Cabinet “Animation File” on the “Misc Options” tab in the Cabinet LEDs section.

Note: You may need to enable the “Stereo Mix” or “What You Hear” recording device to use the LEDBlinky audio animation features. This [LEDBlinky support link](#) has complete directions.

Game Play Animation
Game Pause Animation
FE Active Animation
Screen Saver Animation
Cabinet LEDs Animation

Select the animation file you wish to use for the Audio Animation.

Note: The “Blink Controls” animation mode only uses the first frame of the animation.

Note: No animation file is required for the “Color/Intensity Spectrum” animation mode.

Note: All Cabinet LED animations share the same audio animation.

Test

Click the “Test” button to start the audio animation testing with the selected options. Most options can be changed while actively testing. Click the button a second time to stop the test.

Note: You must have active audio input for the test to work – play some music!

Audio Device

Select the sound card used for the audio input.

Audio Input

Select the audio input source. Options here are dependent on the selected audio device (sound card).

Audio Animation Mode

Select the primary audio animation mode you wish to use. Other audio options may be enabled or disabled depending on which animation mode is selected. The following animation modes are available:

Pulse Animation

Using a selected frequency, the specified animation file will be advanced through the frames based on a trigger threshold or increasing amplitude. Basically, the animation can advance to the beat of the music.

Color/Intensity Spectrum

In this mode the color of each LED fades or transitions based on the frequency amplitude. Each control (LED) can be assigned to a different frequency. For RGB LEDs, a common color range can be defined. Basically, the color or brightness changes as the music gets louder or softer.

Blink Controls

Controls (LEDs) are toggled on or off based on a trigger threshold. Each control (LED) can be assigned to a different frequency and trigger threshold. For RGB LEDs, the color is set by the first frame of the selected animation - this allows you to define custom color layouts. Basically, the controls blink on or off based on a volume threshold.

Amplitude Animation

This mode sequentially assigns each frame of the animation to an amplitude. For example, amplitude level 1 uses frame 1, amplitude level 2 uses frame 2, amplitude level 3 uses frame 3, etc. By assigning a different frequency to each button, you can effectively design an animation that sets the color for every button at every amplitude value. This mode works best with animations that have approximately 50 frames.

Spectrum Colors

The Spectrum Colors lists allow you to define one, two, three, or four colors for use with the Color/Intensity Spectrum animation mode. For all other modes, these fields are disabled. To specify a color not in the list, select "Custom". Colors change left to right from low amplitude to high amplitude. See Spectrum Mode below for more details.

Note: For non RGB LEDs the intensity (brightness) will increase regardless of which colors are selected.

Note: The Spectrum Colors options are only enabled for the Color/Intensity Spectrum animation mode.

Spectrum Mode – Color Transition

When using the Color Spectrum animation mode, the Color Transition option uses the amplitude to transition between each adjacent color. Colors transition left to right from low amplitude to high amplitude. For example, if you select "Blue", "Yellow", "Red", each control (LED) will transition from blue to yellow and then yellow to red as the amplitude increases.

Spectrum Mode – Color Fade

When using the Color Spectrum animation mode, the Color Fade option uses the amplitude to fade up from one color to the next. Colors fade left to right from low amplitude to high amplitude. For example, if you select "Blue", "Magenta", "Red", "Red"; each control (LED) will fade up blue, then magenta, then red as the amplitude increases. Blue for the first 25%, magenta for the next 25%, and red for the last 50% (since Red was selected twice).

Note: The Spectrum Mode options are only enabled for the Color/Intensity Spectrum animation mode.

Single Amp Range

With this option enabled, the intensity (brightness) of each LED will increase from 0 to 100% across the color breakpoints. For example, if you have two colors selected, "Yellow" and "Red" – Yellow will light from 0 to 50% intensity, and then Red will light

from 50% to 100% intensity. With this option disabled, Yellow will light proportionally from 0 to 100%, and then Red will light proportionally from 0 to 100%.

Note: The Single Amplitude Range option is only enabled for the Color/Intensity Spectrum animation mode with the Spectrum Mode set to “Fade”.

Trigger Mode – Trigger Threshold

The Trigger Threshold option will pulse an animation or toggle (On/Off) the control based on a specific amplitude. Each control (LED) can be assigned to a different frequency and trigger threshold.

Use the Trigger Threshold slider on the right side of the Spectrum display to set the threshold. When actively testing, the selected threshold is displayed as a red horizontal line above each frequency. You can use the Dead Zone slider to reduce the sensitivity.

Trigger Mode – Increase Amplitude

The Increase Amplitude option will pulse an animation or toggle (On/Off) the control each time the amplitude changes from decreasing to increasing. This usually causes rapid animation pulses or control blinking. You can use the Dead Zone slider to reduce the sensitivity.

Note: The Trigger Mode options are not enabled for the Color/Intensity Spectrum animation mode.

Control Drop-down List (left side, above Spectrum display)

This drop-down list includes all Controls defined for you control panel. Selecting a control does not set any specific option; rather it allows you to assign other options (Frequency or Trigger Threshold) to that control.

For example, if you wish to set player one button one to the third frequency band, select “P1B1” from the drop-down list, then set the Frequency slider to value 3. The frequency and threshold value for each control will be retained regardless of which control is displayed.

Note: The Control list is not enabled when both Lock Frequency and Lock Threshold are checked.

Lock Frequency

Check Lock Frequency to use a single frequency across all controls (LEDs).

Note: Lock Frequency is not enabled for the Pulse Animation mode.

Lock Threshold

Check Lock Threshold to use a single trigger threshold across all controls (LEDs).

Note: Lock Threshold is only enabled for the Blink Controls animation mode.

Trigger Threshold

The Trigger Threshold determines when the animation will be pulsed (advanced) or when the control's LED will toggle on or off. Use the Trigger Threshold slider to adjust the trigger amplitude for the current control or all controls (see Lock Threshold).

When actively testing, the selected threshold is displayed as a **red** horizontal line above each frequency.

When Lock Threshold is unchecked, The Threshold slider label will display the name of the current control, otherwise the label will display "Trigger Threshold".

Note: The Trigger Threshold slider is not enabled for the Color/Intensity Spectrum animation mode or when the Increase Amplitude option is enabled.

Amp Mult

Use the Amplitude Multiplier slider to adjust the overall input amplitude. This adjustment depends on your PC and audio player's volume setting. Increasing the Amplitude Multiplier may cause low frequency amplitude values to occasionally clip (max out at the highest color/intensity).

Slide the Amplitude Multiplier to its lowest position to set "auto" mode. In auto mode the software will increase or decrease the multiplier dynamically to maintain an optimal value. The *settings.ini* file contains three values which effect how the auto Mult Amp mode works. The default values for these settings should be adequate – but they can be modified if you wish.

- AutoAmpMultUpdateRate – The rate at which the multiplier value is adjusted in milliseconds. The default value is 2000 (2 seconds).
- AutoAmpMultTargetAvgLow and AutoAmpMultTargetAvgHigh – The target amplitude range for which the multiplier remains stable. These are average amplitude values between 1 and 50. The defaults are 8 and 14. The average amplitude is calculated using the first 10 frequencies only.

Note: You should leave your PC's volume at a fixed level and use your speaker's volume control to adjust the music output level.

Display Decay

With the Display Decay option unchecked, the spectrum analyzer will display the amplitude drop-off (decay) in real-time. With the option checked, the amplitude drop-off will use a linear delay based on the value set by the Decay slider. This has the effect of smoothing the spectrum analyzer display.

Note: This option only affects the spectrum analyzer display – it has no effect on the LEDs.

Display Peaks

With this option checked, additional horizontal segments will display above each frequency (during active testing) to help visualize the amplitude peaks (momentary high values).

Note: This option only affects the spectrum analyzer display – it has no effect on the LEDs.

Trigger Frequency

The Trigger Frequency determines which frequency band is used to pulse (advance) the animation or blink the control's LED. Use the Trigger Frequency slider to select the frequency band for the current control or all controls (see Lock Frequency).

When actively testing, the selected frequency band is displayed as a **blue** or **red** bar. All other frequency bands assigned to controls are displayed as white bars. Any unassigned frequency bands are displayed as **gray** bars.

When Lock Frequency is unchecked, The Frequency slider label will display the name of the current control, otherwise the label will display "Trigger Frequency".

Decay

The Decay slider introduces a linear delay as the LED intensity drops off. This has the affect of smoothing the color/intensity changes. Higher values have less affect (less smoothing). Setting the value to zero (0) disables the decay and causes the color/intensity changes to occur in real-time (fast).

Note: The Decay slider is not enabled for the Pulse Animation mode.

Sample Rate (ms)

Use the Sample Rate slider to adjust how often (in milliseconds) the audio input is sampled. Higher values have the affect of slowing down the LED activity. The lowest value (10ms) will result in the LED activity most accurately responding to the audio input (music). PC's with less processing power may need to use a slightly higher value (30ms).

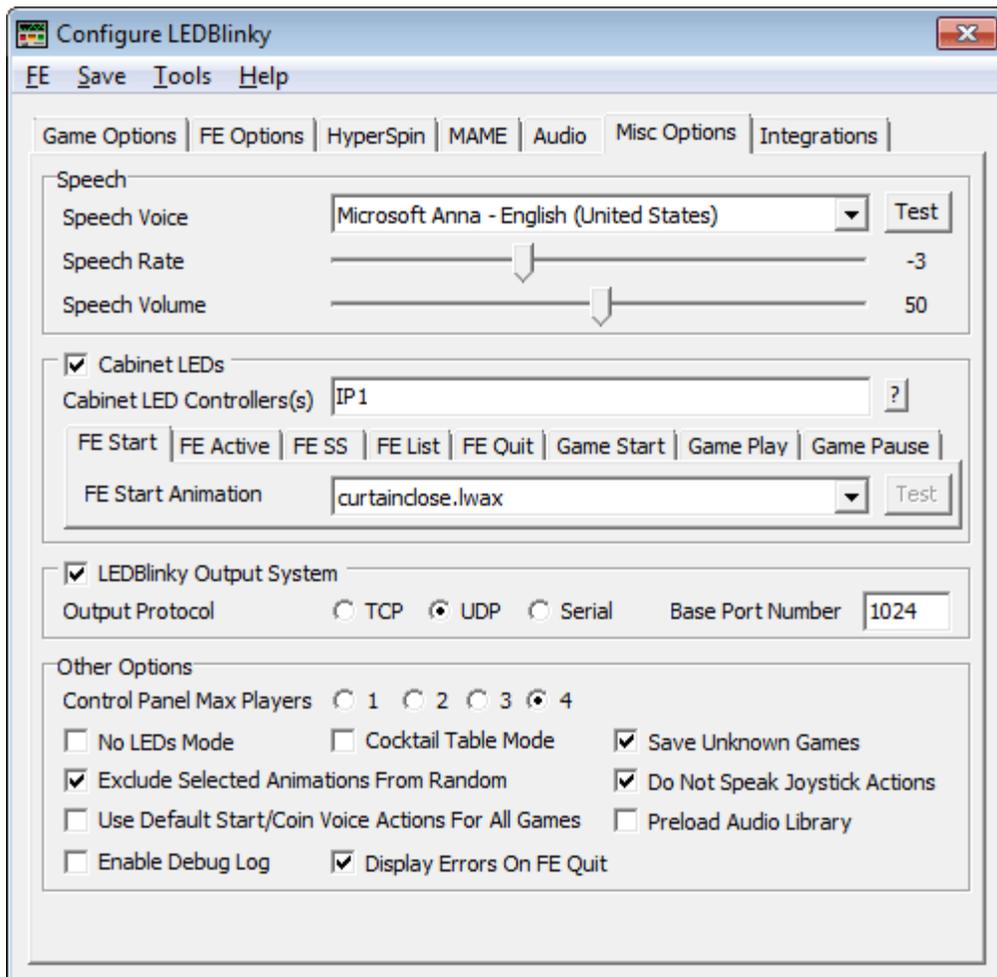
Dead Zone

The Dead Zone slider acts as a sensitivity adjustment. It is most effective when the "Increase Amplitude" option is enabled.

When actively testing, the selected threshold will be displayed as two **red** horizontal lines above each frequency – representing the top and bottom of the dead zone.

Note: The Dead Zone slider is not enabled for the Color/Intensity Spectrum animation mode.

Misc Options



Speech Voice

By default, the Windows XP Text To Speech Synthesizer comes with the Microsoft Sam voice and Windows 7 comes with Microsoft Anna. Microsoft also provides two other freely downloadable voices – Mary and Mike. Other third-party voices can be purchased and downloaded.

Speech Rate

Move the slider to set the voice speech rate from -10 to +10. The default is 0.

Speech Volume

Move the slider to set the voice speech volume from 0 to 100. The default is 100 (full volume).

Cabinet LEDs

LEDBlinky can run a secondary (independent) animation for cabinet or attract mode lighting. To use this feature, you must have additional LED controllers (one or more) wired to the cabinet LEDs.

Note: If you are only lighting your control panel LEDs, do not enable the Cabinet LEDs option.

LED Controller(s)

To use the secondary animation features, you must have additional LED controllers (one or more) wired to the cabinet LEDs. Each [LED controller](#) is assigned a unique ID. Specify the controller IDs used for cabinet lighting as a comma separated list of controller abbreviations and IDs. Click the help “?” button for a list of all LED controller abbreviations.

For example, let's say you have an LED-Wiz (ID2) and two Pac-LED64s (IDs 1 and 2) wired to your cabinet LEDs. You would specify LW2,PL1,PL2.

Cabinet LEDs Event Tabs

Cabinet LED animations can run for various events. To enable an animation for each event, click the associated tab and then select an animation file.

Note: If the same animation is selected for consecutive events, LEDBlinky will continue the animation as the event(s) change without interruption. For example, if you want to run an uninterrupted animation while the front-end or emulator is active, select the same animation for the FE Active, FE Screen Saver, Game Play, and Game Pause events.

Cabinet LEDs FE Start and Quit Animation

Select the file name you wish to run on the cabinet LEDs when the front-end first starts and/or quits. When <Random> is selected, a different file (from the list) will run each time the front-end is started. Only a single loop of the LED animation will be played (one pass through the file). Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

Cabinet LEDs FE Active Animation

Select the file name you wish to run on the cabinet LEDs while the front-end is active (not during game play). When <Random> is selected, a different file (from the list) will run each time the front-end starts. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

When <Audio Animation> is selected, the Cabinet LEDs Animation along with other audio options must be selected on the [Audio Animation](#) tab.

Cabinet LEDs FE SS (Screen Saver) Animation

Select the file name you wish to run on the cabinet LEDs while the front-end screen saver is active. When <Random> is selected, a different file (from the list) will run each time the screen saver starts. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

When <Audio Animation> is selected, the Cabinet LEDs Animation along with other audio options must be selected on the [Audio Animation](#) tab.

Cabinet LEDs FE List (Change) Animation

Select the file name you wish to run on the cabinet LEDs each time the front-end changes Emulators or Game lists. When <Random> is selected, a different file (from the list) will run each time the list is changed. Only a single loop of the LED animation will be played (one pass through the file). Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

Cabinet LEDs Game Start and Quit Animation

Select the file name you wish to run on the cabinet LEDs each time a game starts. When <Random> is selected, a different file (from the list) will run each time a game is started. Only a single loop of the LED animation will be played (one pass through the file). Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

Cabinet LEDs Game Play Animation

Select the file name you wish to run on the cabinet LEDs while playing a game (emulator is active). When <Random> is selected, a different file (from the list) will run each time a game is played. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

When <Audio Animation> is selected, the Cabinet LEDs Animation along with other audio options must be selected on the [Audio Animation](#) tab.

Cabinet LEDs Game Pause Animation

Select the file name you wish to run on the cabinet LEDs while a MAME game is paused. When <Random> is selected, a different file (from the list) will run each time a game is paused. Click the “Test” button to run the selected animation, and click the button a second time to stop the animation.

When <Audio Animation> is selected, the Cabinet LEDs Animation along with other audio options must be selected on the [Audio Animation](#) tab.

Note: This feature only works for MAME version .118 or later.

LEDBlinky Output System

The LEDBlinky Output system provides a mechanism for custom or 3rd party hardware to consume and respond to LEDBlinky data (the same as the currently supported LED controllers). The only requirement is that the custom hardware monitors LEDBlinky data over TCP, UDP, or Serial protocols. See [here](#) for more information.

Output Protocol

Communication protocol used by the LEDBlinky Output system. TCP, UDP, or Serial options are available. The Output Protocol options are only enabled when LEDBlinky Output System is checked. See [here](#) for more information.

Base Port Number

Base Port Number used by LEDBlinky Output system for TCP or UDP protocols. Each LEDBlinky Output virtual controller is assigned an ID number. The ID number is added to the Base Port Number to determine the outgoing windows port used for data communication (TCP or UDP). The Base Port Number is only enabled when LEDBlinky Output System is checked and TCP or UDP is selected. See [here](#) for more information.

Baud Rate

Baud Rate used by LEDBlinky Output system for communication over the Serial/COM port(s). Using a higher baud rate is recommended if any LEDBlinky animations will be used. Even at the highest baud rate (115200), animations should not exceed 30 frames per second. The Baud Rate is only enabled when LEDBlinky Output System is checked and Serial is selected. See [here](#) for more information.

Control Panel Max Players

Set this value to the maximum number of players on your control panel for which you wish to light LEDs. This feature may prevent unexpected buttons from lighting up when your buttons are mapped to input codes that are normally used as defaults for other player buttons not available on your control panel.

For example, MAME uses (by default) keycodes I, J, K, and L for the player 3 joystick. If your control panel only supports two players and you use keycodes I, J, K, or L for your buttons, then these buttons may light incorrectly when you play a 4-player game. By setting the Control Panel Max Players to 2 for a two player control panel, you will avoid this problem.

No LEDs Mode

Enable No LEDs Mode ONLY when you are using LEDBlinky without any LED controllers. This is useful if you wish to use the LEDBlinky speech or Joystick Digital Restriction features but do not have any LEDs on your control panel.

Cocktail Table Mode

Enable Cocktail Mode if your cabinet is a cocktail table. Normally a multi-player *alternating* player game will only light the controls for player 1. With cocktail mode enabled, all player controls will light. For example, Galaxian will light the controls for Players 1 and 2.

Save Unknown Games

Unknown games/ROMs (games without any control configuration data) will not light or speak any controls. Saving unknown games will allow the [LEDBlinky Controls Editor](#) to

display and [import](#) these games. After a game is imported you can configure the controls.

Exclude Selected Animations From Random

If you have configured LEDBlinky to run a <Random> or <Random Montage> animation and also configured one or more other animation options to use a specific selected animation, then check this box to exclude the selected animation(s) from the random. For example, if the FE Start animation is “CurtainClose” and the Game Start animation is <Random>, then this option will prevent the CurtainClose animation from ever being selected randomly.

Do Not Speak Joystick Actions

If your control panel has illuminated joysticks and you have configured LEDBlinky to blink and [speak the game controls](#) on game start, each joystick may speak the four (4) directional actions. For example, while the joystick is blinking you might hear “up... down... left... right”, or “fire up... fire down...” etc. Check this option to disable speaking the joystick actions.

Use Default Start/Coin Voice Actions For All Games

Prior to LEDBlinky version 7.3 the Start and Coin button voice actions were globally defined as “Start Game” and “Insert Coin”. As of version 7.3 the Start and Coin button voice actions can be modified for each emulator and/or game using the Controls Editor. If a value is not defined in the Controls Editor, the global values will be used as a default. Enabling this option will only use the global default values (any values voice action values for Start or Coin buttons defined in the Controls Editor will be ignored).

Preload Audio Library

Enable Preload Audio Library when <Audio Animation> has not been selected for any animation feature and you are using an external command to start the audio animation. For example, your FE is using a pre-launch command or script to load the audio animation before launching a jukebox application.

Enable Debug Log

With the Debug Log enabled, LEDBlinky will add debug messages to *Debug.log* file located in the *LEDBlinky* folder. This should only be used when diagnosing a problem. Additional *Debug_*.log* files will also be generated and zipped into a *Debug.zip* file. This may slow LEDBlinky performance and should only be used when diagnosing a problem.

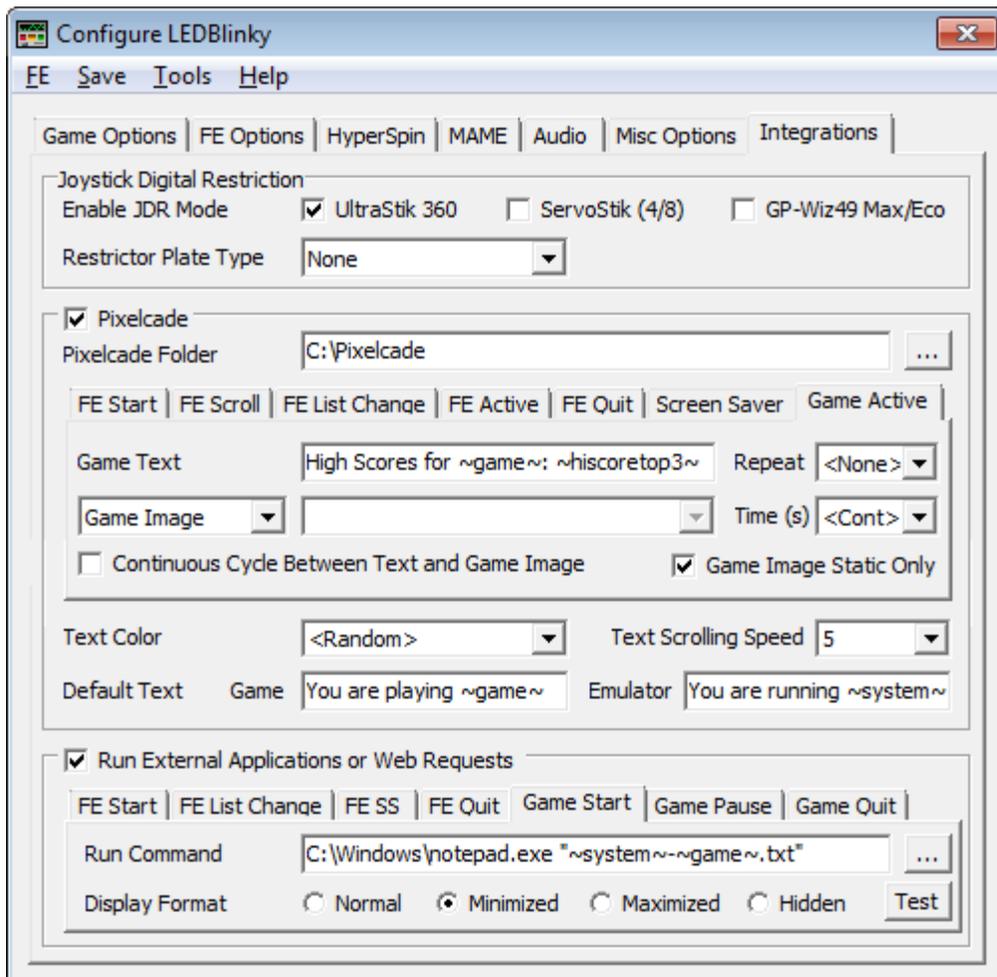
Note: Please include the *Debug.zip* file with any correspondence to diagnose problems.

With the Debug Log disabled, LEDBlinky will log all error messages to the *LEDBlinky.log* file.

Display Errors On FE Quit

LEDBlinky logs all errors to the *LEDBlinky.log* file in the *LEDBlinky* folder. If one or more errors occur during execution, LEDBlinky will display a message after you exit the front-end and provide you with the option to display the log file.

Integrations



Enable JDR Mode

Joystick Digital Restriction (JDR) allows digital joysticks such as an Ultimarc UltraStik 360 or 49-Way controlled via the Groovy Game Gear GP-Wiz49 Max/Eco, to be configured as is appropriate for the active game (2-way, 4-way, 8-way, etc.). An Ultimarc ServoStik or GRS tos428 physical restrictor plate can also be switched between 4-way and 8-way. When a game is started, the JDR mode is set based on the game's [primary control](#). When the front-end is active, the JDR mode is set to 4-way by default.

UltraStik 360

Check this box to enable JDR mode for any/all UltraStik 360 joysticks. Do not enable this feature if your control panel does not use any UltraStiks.

Each primary control type translates to an UltraStik 360 map file:

```
JOY2WAY = Joy2Way.um
VJOY2WAY = vJoy2Way.um
JOY4WAY = Joy4Way.um
```

JOY8WAY = Joy8Way.um
JOYDIAG = JoyDiag.um
STICK = Analog.um
DOUBLEJOY2WAY = Joy2Way.um
VDOUBLEJOY2WAY = vJoy2Way.um
DOUBLEJOY4WAY = Joy4Way.um
DOUBLEJOY8WAY = Joy8Way.um
TRACKBALL = Analog.um
DIAL = Analog.um
PADDLE = Analog.um
LIGHTGUN = Analog.um
PEDAL = Analog.um
PEDAL2 = Analog.um

See [here](#) for a description and how to set a custom UltraStik map (.um file) for a specific ROM/Game.

Restrictor Plate Type

Ultimarc offers physical restrictor plates which can be installed into your UltraStik 360 joysticks. If you have installed a restrictor plate, select the type from the list. Each value will compress the outer range of the joystick. Custom-1 has the most compression and Custom-15 the least compression. Custom-9 corresponds to the physical 4-Way/8-Way/Circular restrictor plate.

ServoStik

Check this box to enable automatic 4-way/8-way switching for any/all ServoStik joysticks. Do not enable this feature if your control panel does not use any ServoStiks.

The ServoStik physical restrictor plate will be rotated into 4-way mode for any game which uses a 4-way or 2-way joystick as its primary control. For all other games (or the front-end), the restrictor plate will be rotated into 8-way mode.

GRS tos428 Restrictor

Check this box to enable automatic 4-way/8-way switching for any/all GRS tos428 joystick restrictors (used with Sanwa Joysticks). Do not enable this feature if your control panel does not use any tos428 restrictors.

The tos428 physical restrictor plate will be rotated into 4-way mode for any game which uses a 4-way or 2-way joystick as its primary control. For all other games (or the front-end), the restrictor plate will be rotated into 8-way mode.

Pixelcade

Pixelcade is an LED/LCD Marquee for Arcade Machines. To use the LEDBlinky Pixelcade features, you must purchase the Pixelcade display panel(s) and install the Pixelcade software. Please visit the [Pixelcade website](#) for complete details. Check the option to enable Pixelcade features.

Pixelcade Folder

Click the folder browse button  to select the folder where Pixelcade is installed.

Pixelcade Event Tabs

LEDBlinky can display text and/or images on the Pixelcade panel(s) for the following events; FE Start, FE Scroll, FE Active, FE Quit, FE Screen Saver, and Game Active. All events with the exception of FE Scroll have the same configuration fields. See below for a description of each field.

Event - Text

Specify the text you wish to display on the Pixelcade. The text will scroll from right to left. You can use the following special keywords (each keyword must be surrounded by tilde character).

- **~game~** Replace keyword with the current Game/ROM name. If the Game/ROM name is not available, the [Default Text For Game](#) will be used.
- **~system~** Replace keyword with the current System/Emulator name. If the System/Emulator name is not available, the [Default Text For Emulator](#) will be used.
- **~date~** Replace keyword with the current date.
- **~time~** Replace keyword with the current time.
- **~highscore~** Replace keyword with high score and initials for current game.
- **~highscoretop3~** Replace keyword with top three (3) high scores and initials for current game.
- **~highscoreall~** Replace keyword with all high scores and initials for current game.

Note: High Score keyword replacement only works for MAME games and not all MAME games support high scores.

LEDBlinky uses the Hi2txt app to decode MAME high score files. The version of Hi2txt included with the LEDBlinky Installation may not be the latest version. For Hi2txt support and to download the latest C# version and database of supported games see; <https://greatstoneex.github.io/hi2txt-doc/>

Additionally, it may be necessary to enable high score functionality in MAME. Newer versions of MAME (0.172+) support high scores natively. For older versions of MAME, you may need a high score plugin. Please see Hi2txt and MAME documentation for more information.

Event - Text Scroll Repeat

How many times the specified text will repeatedly scroll across the Pixelcade. Select <None> to scroll the text a single time (no repeat). Select <Cont> to scroll the text continuously.

Event - Image Type

Each event type (with the exception of FE Scroll) will present four (4) image options;

1. No Image.

2. Marquee Image / Emulator Image / Game Image; The specific image depends on the selected event tab. If no image file is found, the display will remain blank.
3. Animation Image; The Pixelcade software installation includes a set of animated images. The animated image files are located in the Pixelcade \animations folder. Select any file or choose <Random>.
4. User Image; User provided custom image files must be located in the Pixelcade \user folder. Select any file or choose <Random>.

Note: The image will be displayed after scrolling the specified event text (if any).

Event - Select Image

Select Animation or User image, or choose <Random>. The “Select Image” field is only enabled when the Image Type is set to “Animation Image” or “User Image”.

Event - Image Display Repeat / Time

When displaying an animation, you can specify the number of times the animation will repeat. Select <None> to have the animation run a single time (no repeat). Select <Cont> to have the animation run continuously.

When displaying an image, you can specify the time (in seconds) the image will be displayed. Select <Cont> to display the image continuously.

Event FE Scroll - Display System/Game Image When Scrolling FE Lists

Enable this option to display the System or Game images as you scroll through your front-end lists. This feature works similar to the [Demo Game Controls](#) feature and both use the same “Scroll Delay” value.

For LEDBlinky to know which front-end lists are lists of Systems/Emulators, you must correctly set the [FE Systems Lists Names](#).

Event FE Scroll - Scroll Delay

The scroll delay is the time between when you stop scrolling and when the Pixelcade displays the system or game image. When set to zero, the image will be displayed immediately as you stop on each list item, but this may cause the Pixelcade to lag behind. The scroll delay value set here is the same as the [Demo Scroll Delay](#).

Event Screen Saver – Text and Image Mode

The Text and Image Mode provides the same options as the other Pixelcade events, allowing for custom scrolling text and/or displaying an image or animation.

Event Screen Saver – Ticker Mode

The Ticker Mode (only available for Screen Saver events) will scroll a predefined set of RSS feeds. The pixelcade-config or pixelcade-ticker applications (included with the Pixelcade installation) can be used to define from one to five RSS feeds.

Note: The Ticker Mode feature has been (temporarily) removed from Pixelcade.

Event Game Active – Continuous Cycle Between Text and Game/Animation/User Image

The Continuous Cycle mode (only available for Game Active events) will cycle between scrolling the custom specified text and displaying the selected image or animation. The text will scroll a single time (no repeat) and the image will display for the specified time or the animation will repeat the specified number of iterations, then the sequence will repeat.

Note: With this mode enabled you must specify both the Game Text and the Image or Animation. With this mode enabled you cannot select continuous <Cont> text scrolling.

Event Game Active – Game Image Static Only

Only display static game image even if animated gif is available for the selected game.

Text Color

Select the color for all scrolling text or select <Random>.

Text Scrolling Speed

Select the speed for all scrolling text. Higher values are faster.

Default Text for Game

If a game image file is not found, this text will be displayed. The same keywords can be used as for the [Pixelcade Text](#).

For example: *You are playing ~game~*

Default Text for Emulator

If a system/emulator image file is not found, this text will be displayed. The same keywords can be used as for the [Pixelcade Text](#).

For example: *~system~ Games*

Run External Applications or Web Requests

LEDBlinky can run an external application or execute a web request (http or https GET) on events such as when the FE is started or when a Game is started. Check the option to enable the Run External Application or Web Requests features.

Run External Applications or Web Requests Event Tabs

LEDBlinky can run an external application or execute a web request on the following events; FE Start, FE List Change, FE Screen Saver Start, FE Quit, Game Start, Game Pause, and Game Quit. This can be used for running batch or script files or any other executable.

Note: The commands/web requests will run at the end of each event after all other LEDBlinky processing is completed.

Run Command

The Run Command defines the complete path, executable file, and any additional parameters required to run the external command. Click the file browse button  to select any “runnable” file, and manually add additional parameters. Additional parameters must be enclosed in double-quotes. See example in screen-shot above.

For some events, you can use the following special keywords (each keyword must be surrounded by tilde character).

- **~game~** Replace keyword with the current Game/ROM name. If the Game/ROM name is not available, a blank will be used.
- **~system~** Replace keyword with the current System/Emulator name. If the System/Emulator name is not available, a blank will be used.

For Web Requests, the Run Command defines the URL for the request. The URL must be complete as would be executed from a web browser.

Display Format

When LEDBlinky runs an external application, the application can be started in one of four modes; Normal, Minimized, Maximized, and Hidden.

Note: When using the “Hidden” format, it is assumed the application will automatically close after it completes its task. If the application does not close, you can use the Windows Task Manager to kill the hidden process.

Note: The Display Format does not apply to Web Requests.

Click the Test button to confirm your Run Command is working as expected.

Game Specific Animations

LEDBlinky can run unique animations on a game-by-game basis. To use this feature, the animation files **MUST** have the same name as the game or rom, and you must have a default animation selected for the related animation option in the Configuration app. Place the game specific animations in the desired "event" folder under the /LEDBlinky/GameSpecific/ folder.

The following events folders can trigger a game specific animation:

- Game Start
- Game Play
- Game Play using an audio animation
- Game Pause
- Game Pause using an audio animation
- Cabinet LEDs Game Start
- Cabinet LEDs Game Play
- Cabinet LEDs Game Play using an audio animation
- Cabinet LEDs Game Pause
- Cabinet LEDs Game Pause using an audio animation

Note: [Cabinet LEDs](#) must be enabled to use any of the cabinet LED events.

For example, let's say you wish to run a specific animation when playing Joust; first you must have an animation selected for the [Game Play Animation](#) option - this is the default animation. Next, create (or copy) your Joust animation into the \LEDBlinky\lwa\GameSpecific\GamePlay folder. The file **MUST** have the exact same name as the game/rom the emulator is running. So for Joust, you would use "joust.lwax". That's it.

LEDBlinky Controls Editor

Use the LEDBlinky Controls Editor to configure the controls for any Emulator, Game, or Front-end. For each control you can specify the color or intensity, the voice 'action', and if necessary the input codes associated with the control. Controls can be defined as the default for an emulator, or individually for a specific game.

The primary need for the Controls Editor is to configure any non-MAME emulator or game. Since there is no way for LEDBlinky to know the control-input mapping (button assignments), or specific controls, or button colors for each non-MAME emulator, you must provide this information manually. Additionally, you can use the Controls Editor for MAME games to override values in the *Controls.ini* and *Colors.ini* files.

The Controls Editor provides a number of features to ease the configuration process. Each time you play an unknown game (one for which no unique controls are defined), the emulator and ROM/game name is stored. From the Controls Editor import menu, you can display the list of unknown games and select which you wish to import. You can then define the controls. Controls for player 1 can be copied to players 2, 3, and 4. If the controls for one game are similar to another, you can copy the entire ROM/Game.

The trick to getting your buttons to light for each emulator requires that you correctly define the Emulator and ROM/Game names and match the control Input Codes to the emulator's configuration. Each emulator passes (to LEDBlinky) the emulator name and game name when you launch a game. These values must be defined correctly – use the [Import Unknown Games](#) option. Using the emulator and game names, LEDBlinky will then match the controls you have defined (using the Controls Editor). But that's only half the job – to correctly light up the LEDs, the Input Codes (keycodes) for each control must match those used by the emulator. For example; if Player 1, Button 1 is configured by the emulator to use keyboard button "A", you must assign P1_BUTTON1 to KEYCODE_A. Detailed instructions are provided below.

Menus



Save

Save the controls configurations. The LEDBlinky controls configuration file is *LEDBlinkyControls.xml* located in the LEDBlinky folder. The “Save” menu option is only enabled when one or more controls have been added or modified. If you attempt to close the Controls Editor app prior to saving, you will be prompted to save the data.

Note: Do not edit the *LEDBlinkyControls.xml* file manually.

Copy ROM/Game

Copy all controls for any ROM/Game to a new ROM/Game name. The new set of controls can be created for any emulator. You must select a specific ROM/Game to enable the “Copy ROM/Game” menu option. After you click the menu option, you will be presented with a dialog to select the destination Emulator.

Note: If the destination Emulator has one or more [Unknown Games](#), the new [ROM / Game](#) name field will provide a drop-down list of unknown game names for that emulator. This makes it easy to copy the controls from a configured game to an unknown game.

Copy Player 1 Controls

Copy Player 1 Controls w/Overwrite

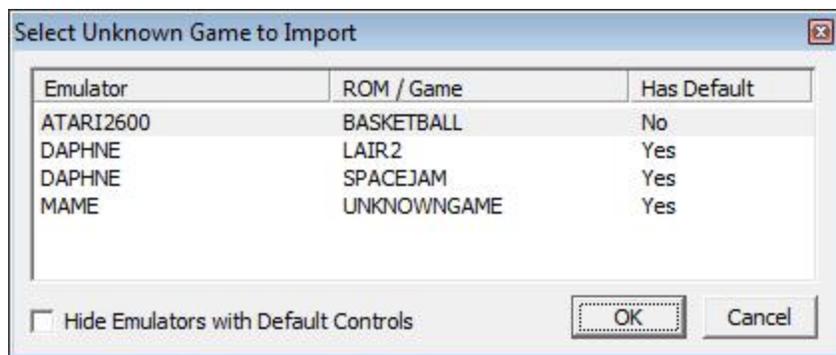
Copy the current set of Player 1 Controls to all other players for the selected game. Most multiplayer games use the same controls for each player – this feature allows you to define the controls for player 1 and then quickly copy them to all other players.

Without the Overwrite option, Player 1 controls will only be copied to other players for which no controls have been defined. With the Overwrite option, any existing player controls will be replaced with Player 1 controls.

Note: Input codes are never copied across players – they must be uniquely defined for each player control.

Import Unknown Games

When configuring new Emulators or Games, start with this option! Each time you play an unknown game, LEDBlinky stores the Emulator name and ROM/Game name as specified by your front-end. Click the “Import Unknown Games” menu option to display the unknown games list.



Select the ROM/Game you wish to import and click “OK”. The controls editor lower pane will open with the game name pre-populated. Do not change ROM/Game Name. You can then proceed to add all the necessary controls.

Note: If the emulator does not exist for the selected game, it will be automatically added. You may wish to edit the Emulator Description.

If no default controls have been specified for the selected Emulator, you will be given the option to import the selected ROM/Game or create a default control group for the emulator. The default control group will be designated as <default>. A default control group is very useful when most or all games for a specific emulator use the same controls. The default controls are only used by LEDBlinky when no individual control group has been defined for the ROM/Game.

Note: You can add a Default Control Group at any time – but only one is allowed for each emulator.

Import MAME Games

The controls for most MAME games are predefined by the *Controls.ini*, *Colors.ini*, and *Mame.xml* files. If you wish to override the predefined control values, use the “Import MAME Games” menu option. You must know the exact ROM name to import the MAME game. After importing the game, you may edit any or all of the controls.

Note: Once a MAME game has been imported, LEDBlinky will not use any control values for that game from the *Controls.ini*, *Colors.ini*, and *Mame.xml* files. If you wish to revert back to the predefined control values, delete the game from the ROM/Game list.

Test LEDBlinky

Use the “Test LEDBlinky” menu option to test the configured game start options for any game. All Speech and LED options will activate exactly as if the game was started from your front-end. The “LEDBlinky menu option is only enabled when a ROM/Game is selected.

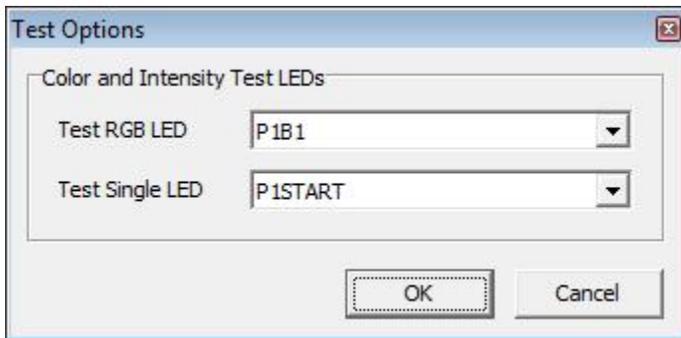


Click "Start Test" to simulate running the game. After the speech and LED activity has completed, you can click "Reset" to test again.

Note: All Game Start options are configured using the [LEDBlinky Configuration](#) app.

Test Options

Select the LEDs (RGB and Single) used for testing color and intensity when adding or editing controls, from the list of LED Labels as defined in your input map. Any LED on your control panel can be used for testing.



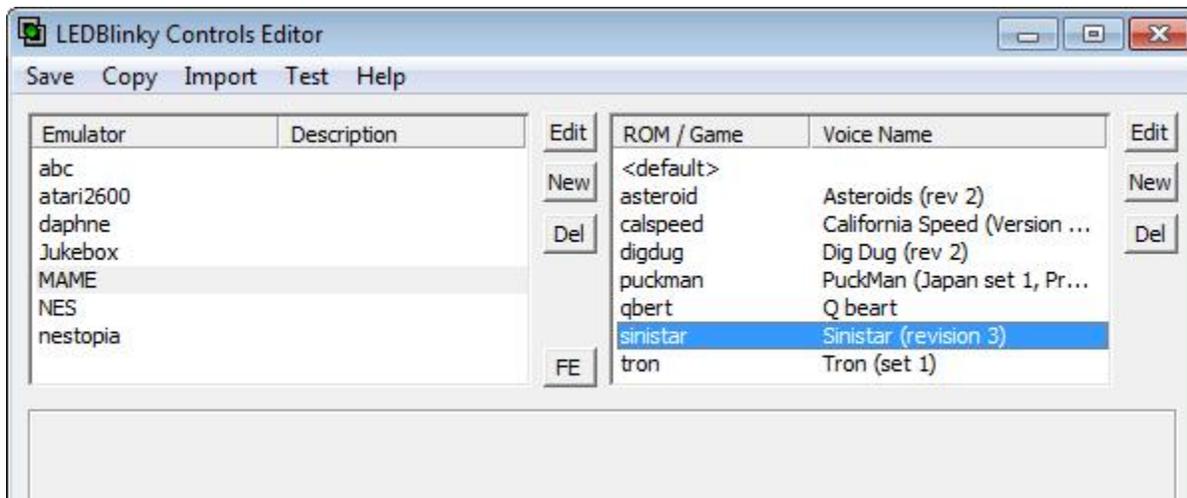
Help

Launches the LEDBlinky help documentation – this document!

Emulator and ROM/Game Lists

The LEDBlinky Controls Editor main window consists of an upper and lower pane. The upper pane contains the Emulator and ROM/Game lists. When the Controls Editor is first opened, the lower pane will be blank.

Note: Emulator and ROM/Game modifications (Edit/New/Delete) are not permanent until you click the “Save” menu option. If you wish to undo modifications, close the LEDBlinky Controls Editor without saving and then reopen the editor.



Emulator List

The left upper pane displays the list of emulators you have configured. Select an emulator in the list to load the list of games configured for that emulator. Double-clicking an emulator in the list will load the emulator into the lower pane for editing. Using the column headers, you can increase or decrease the width of each column.

Note: If you have an emulator designated as the default, it will be listed as “<default>”. See [below](#) for a description of the default emulator.

Edit (Emulator)

Click the “Edit” button to the right of the Emulator List to edit the selected emulator. This will load the emulator into the lower edit pane and disable the upper list panes. The “Edit” button is only enabled when an emulator is selected. See [below](#) for a description of the emulator fields.

Note: The upper list panes will remain disabled until the “OK” or “Cancel” button is clicked on the lower edit pane.

New (Emulator)

Click the “New” button to the right of the Emulator List to add a new emulator. This will display a blank emulator in the lower edit pane and disable the upper list panes. See [below](#) for a description of the emulator fields.

Note: Rather than manually adding a new emulator name, it is highly recommended that you use the [Import Unknown Games](#) option.

Note: The upper list panes will remain disabled until the “OK” or “Cancel” button is clicked on the lower edit pane.

Delete (Emulator)

Click the “Del(ete)” button to the right of the Emulator List to delete the selected emulator. You will be asked to confirm the delete action. The “Del” button is only enabled when an emulator is selected.

Note: Deleting an emulator will also delete all ROM/Games listed for that emulator.

FE

Click the “FE” button to edit the controls for your front-end. FE controls are similar to emulator controls. See [below](#) for a description of the FE controls.

Note: Use the LEDBlinky Configuration app to specify your front-end.

ROM / Game List

The right upper pane displays the list of games you have configured for the currently selected emulator. Select a ROM/Game in the list to enable the edit buttons. Double-clicking a ROM/Game in the list will load the game into the lower pane for editing. Using the column headers, you can increase or decrease the width of each column.

Note: If you have a game designated as the default for the selected emulator, it will be listed in the ROM/Game list as “<default>”. See [below](#) for a description of the default game.

Edit (ROM/Game)

Click the “Edit” button to the right of the ROM/Game List to edit the selected game. This will load the game into the lower edit pane and disable the upper list panes. The “Edit” button is only enabled when a game is selected. See [below](#) for a description of the game fields.

Note: The upper list panes will remain disabled until the “OK” or “Cancel” button is clicked on the lower edit pane.

New (ROM/Game)

Click the “New” button to the right of the ROM/Game List to add a new game. This will display a blank game in the lower edit pane and disable the upper list panes. See [below](#) for a description of the game fields.

Note: Rather than manually adding a new ROM/Game name, it is highly recommended that you use the [Import Unknown Games](#) option.

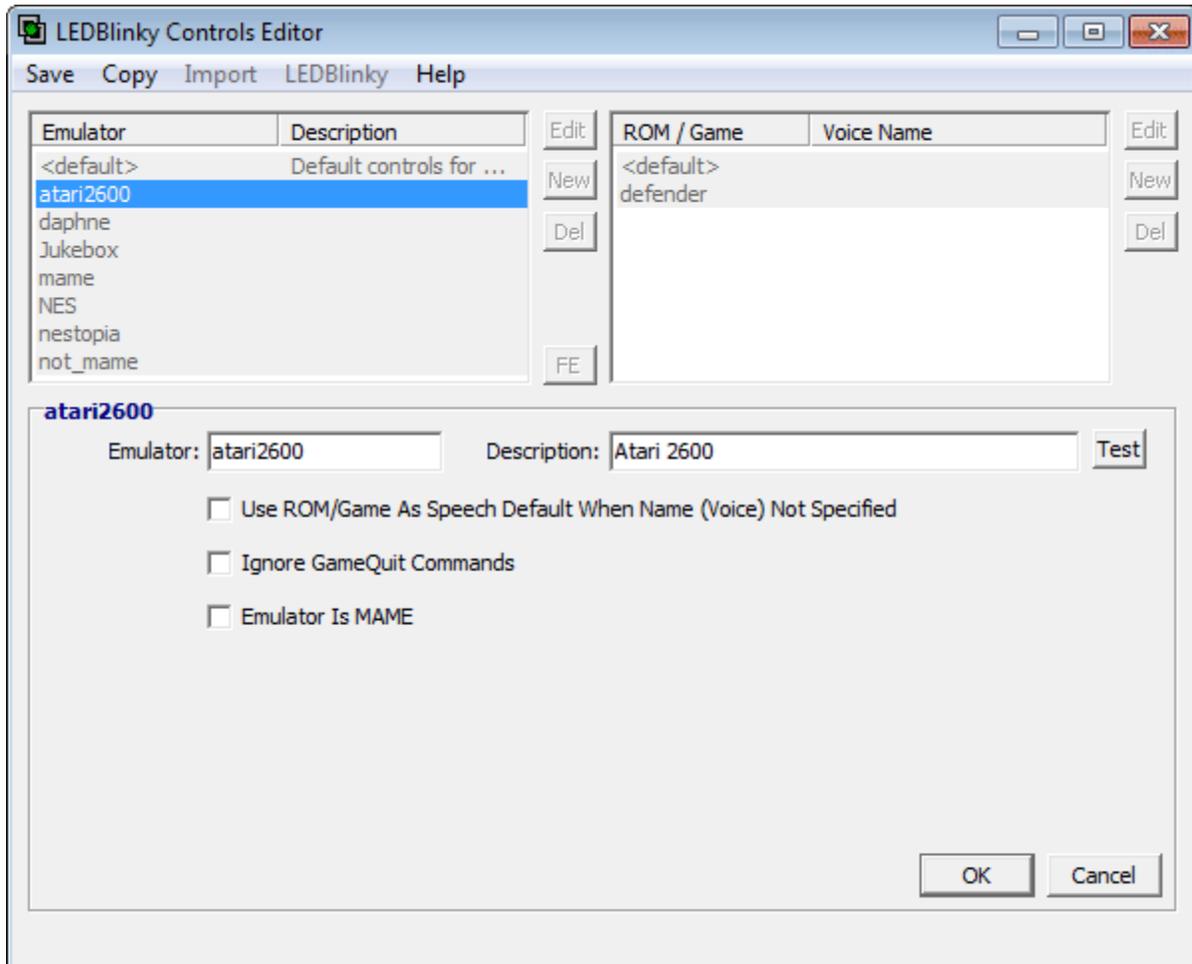
Note: The upper list panes will remain disabled until the “OK” or “Cancel” button is clicked on the lower edit pane.

Delete (ROM/Game)

Click the “Del(ete)” button to the right of the ROM/Game List to delete the selected game. You will be asked to confirm the delete action. The “Del” button is only enabled when a game is selected.

Edit Emulator

The LEDBlinky Controls Editor main window consists of an upper and lower pane. The lower “edit” pane is displayed when you add a new emulator or edit an existing emulator. While editing an emulator, the upper list panes will be disabled.



Emulator

Enter or modify the Emulator name. This value **MUST** match exactly the name passed by the front-end (to LEDBlinky). If you have an emulator checked as the default, the name will be listed as “<default>” and cannot be edited. If an emulator name has been loaded using the Import Unknown Games option, do not change this value.

Note: The Emulator name cannot include any spaces - spaces will automatically be replaced with underscores (_).

Note: Rather than manually adding a new emulator name, it is **highly recommended** that you use the [Import Unknown Games](#) option.

Note: The emulator with the exact name “mame” is a special exception. Controls configured under the emulator named “mame” will apply to ALL emulators identified, or specifically designated as MAME. Any emulator with “mame” (case independent) located anywhere in the name is automatically identified as MAME, unless specifically [designated as not MAME](#).

Description

Enter any description for the emulator. Currently this field is only used for reference. Click the “Test” button to hear how the text will sound using the current [Text To Speech](#) options – which is pointless because this text is not currently used by LEDBlinky 😊.

Default

Default emulator controls apply to any game for which no other game specific or emulator specific controls have been defined – these are global controls for all emulators (accept MAME).

The “Default” check box is only available when you are creating a “New” emulator and no other emulator has been designated as the default – you can have only one default emulator. Checking the “Default” option will set the Emulator name to “<default>”. You cannot alter the default emulator name.

Note: It is not necessary to create default (global) emulator controls – it is more accurate to define a set of [default](#) controls for each specific emulator.

OK

Click the “OK” button to accept the emulator modifications. Changes or additions will be reflected in the upper left list. After clicking “OK”, the lower edit pane will be blank and the upper list panes will be re-enabled.

Note: Modifications are not saved to the *LEDBlinkyControls.xml* file until the “Save” menu option is clicked.

Cancel

Click “Cancel” to abandon all emulator modifications. After clicking “Cancel”, the lower edit pane will be blank and the upper list panes will be re-enabled. You can also use the “ESC” key as a shortcut.

Use ROM/Game As Speech Default When Name (Voice) Not Specified

LEDBlinky normally speaks the name of a non-MAME game based on [Name \(Voice\)](#) value you assign when configuring a game. If a game is not configured or does not have a Name (Voice) value specified, LEDBlinky will not speak the name. With this option enabled, LEDBlinky will speak the [ROM/Game](#) value whenever a game is not configured or does not have a Name (Voice) value specified. This is useful for emulators that have human readable names for the game files.

Ignore GameQuit Commands

LEDBlinky will ignore all GameQuit commands from the front-end for the selected emulator. You should only enable this option if your front-end is sending false GameQuit commands immediately after launching a game. A symptom of this issue is your control panel returning to the front-end configuration while a game is still active. Be aware, when using this option your control panel may not immediately return to the front-end configuration upon game exit. This option is provided as a work-around.

Note: The Ignore GameQuit Commands option will not work when using a [Game Start Delay](#).

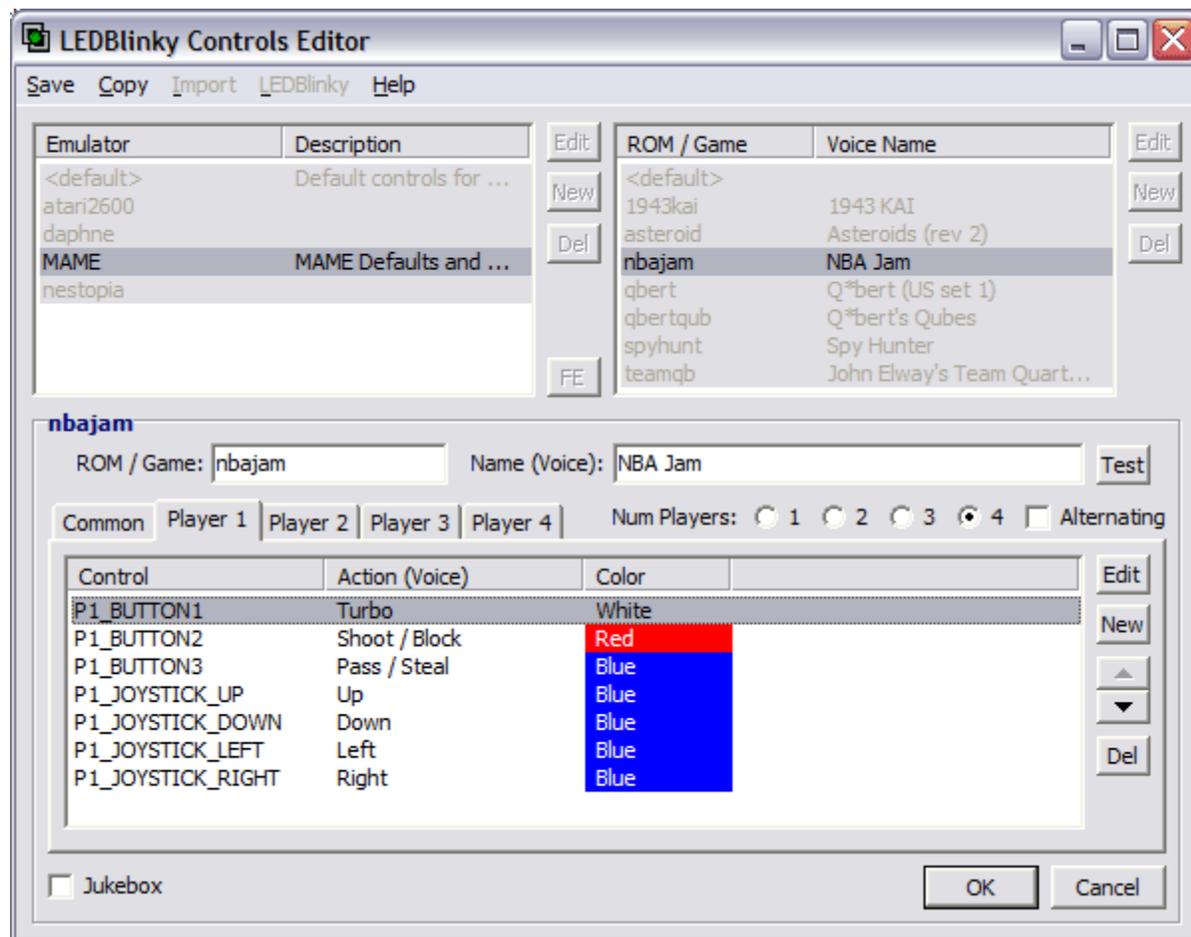
Emulator Is/Is Not MAME

LEDBlinky automatically identifies an emulator as MAME if “mame” (case independent) is located anywhere in the [Emulator](#) name. To override an emulator identified as MAME, select the “Emulator Is Not MAME” option. Any emulator that is not automatically identified as MAME can be configured as MAME by selecting the “Emulator Is MAME” option.

Note: The emulator with the exact name “mame” cannot be designated as Not MAME.

Edit ROM/Game and Controls

The LEDBlinky Controls Editor main window consists of an upper and lower pane. The lower “edit” pane is displayed when you add a new game or edit an existing game. While editing a game, the upper list panes will be disabled.



ROM / Game

Enter or modify the ROM/Game name. This value **MUST** match exactly the name passed by the front-end (to LEDBlinky). If you have a game checked as the default, the name will be listed as “<default>” and cannot be edited. If a game name has been loaded using the Import Unknown Games option, do not change this value.

Note: The ROM/Game name cannot include any spaces - spaces will automatically be replaced with underscores (_).

Note: Rather than manually adding a new game name, it is **highly recommended** that you use the [Import Unknown Games](#) or [Copy ROM/Game](#) menu options.

Note: When using the [Copy ROM/Game](#) menu option, if the selected Destination Emulator has one or more unknown games, the ROM / Game field will provide a drop-down list of unknown game names. This makes it easier to copy a configured game to an unknown game. You can select from the list or type your own name.

Name (Voice)

Enter the game name. The name text will be used to speak the game name when the [option](#) is enabled. Click the “Test” button to hear how the text will sound using the current [Text To Speech](#) options.

Note: You may need to alter the spelling to have the name sound phonetically correct.

Default

Default ROM/Game controls apply to any game for which no other game specific controls have been defined for the current emulator. The default control group also allows you to define controls which are “Always Active” – these controls will be included with every game under this emulator even when a game has its own set of controls defined.

The “Default” check box is only available when you are creating a “New” emulator and no other emulator has been designated as the default – you can have only one default emulator. Checking the “Default” option will set the Emulator name to “<default>”. You cannot alter the default emulator name.

Common Tab

All games have a “Common” player tab which lists all controls that are common across players – usually administration buttons or the [Primary Control](#) type. The “Common” tab also includes the “Default Active” and “Default Inactive” control values.

Player 1/2/3/4 Tabs

Use the “Player Tabs” to display the Controls List defined for each player. The number of player tabs is determined by the “Num Player” option (see below).

Num Players

Select the number of players for the game. Changing the number of players will add or remove “Player Tabs”.

Alternating

Check the “Alternating” option if a game uses one set of controls and alternates the players.

Note: Checking the “Alternating” option will change the “Player Tabs” to only display Common and Player 1 regardless of the “Num Players” setting.

Controls List

The Controls List displays the controls for the selected player (or Common). Select a control in the list to enable the edit buttons. Double-clicking a control in the list will display the [Control Edit](#) window. Using the column headers, you can increase or decrease the width of each column.

Note: If you have a control designated as the [Primary Control](#), the control name will be prefixed with <pri>.

Default Active/Inactive Controls

The first two controls in the “Common” controls list will be the “Default_Active” and “Default_Inactive” controls. These are not true controls – they are only used to define color or intensity. The “Default_Active” color/intensity is used for any active control (used by the game) that does not have a specific color/intensity defined. The “Default_Inactive” color/intensity can be used to light up any non-active control (not used by the game).

Note: The “Default Active/Inactive” controls cannot be deleted or reordered in the Control List, and they are never spoken.

Edit (Control)

Click the “Edit” button to the right of the Control List to edit the selected control. This will load and display the Control Edit window. The “Edit” button is only enabled when a control is selected. See [below](#) for a description of the control fields.

New (Control)

Click the “New” button to the right of the Control List to add a new control. This will display the Control Edit window. See [below](#) for a description of the control fields.

Move Control Up/Down

Use the Up/Down “Arrow” buttons to move the selected control up or down in the Controls List. The Arrow buttons are only enabled when a control is selected.

Note: The order of the Control List determines the order in which the control actions will be spoken before the game starts when the [option](#) is enabled.

Delete (Control)

Click the “Del(ete)” button to the right of the Control List to delete the selected control. You will be asked to confirm the delete action. The “Del” button is only enabled when a control is selected.

Jukebox

Check to designate the game as a jukebox application. This option is used in conjunction with the [Game Play Animation For Jukebox Only option](#).

Note: Any ROM/Game control group (including an emulator default control group) can be designated as a jukebox application. For example, let's assume you are using an audio animation for Game Play and the Game Play Animation For Jukebox Only option is checked. Using the Controls Editor you have designated your jukebox application. If you also have another game (not a jukebox) and you want the audio animation to run for that game, then just designate it as a jukebox!

Global Pulse

The "Global Pulse" speed is only used for LED-Wiz blinking effects (intensities 129 to 132).

OK

Click the "OK" button to accept the ROM/Game modifications. Changes or additions will be reflected in the upper right list. After clicking "OK", the lower edit pane will be blank and the upper list panes will be re-enabled.

Note: Modifications are not saved to the *LEDBlinkyControls.xml* file until the "Save" menu option is clicked.

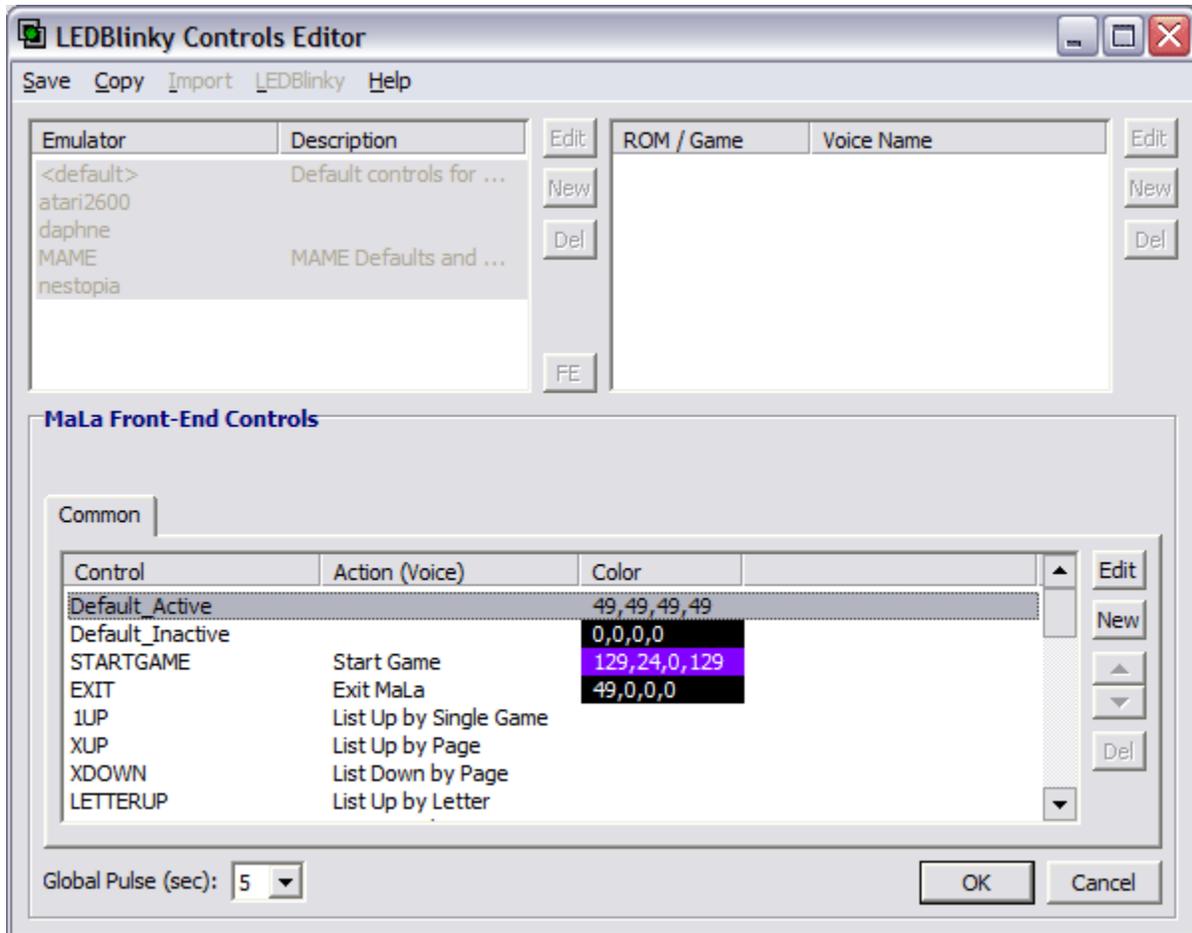
Cancel

Click "Cancel" to abandon all ROM/Game modifications. After clicking "Cancel", the lower edit pane will be blank and the upper list panes will be re-enabled. You can also use the "ESC" key as a shortcut.

Edit Front-end (FE) and Controls

LEDBlinky can light up the controls for MaLa, AtomicFE, GameEx, PinballX, HyperSpin, LaunchBox, Maximus Arcade, Attract-Mode, CoinOps, and RetroFE. Select your front-end from the Configuration app FE menu.

The LEDBlinky Controls Editor main window consists of an upper and lower pane. The lower “edit” pane is displayed when you edit the Front-end controls (click the “FE” button). While editing the FE, the upper list panes will be disabled.



Common Tab

Front-end controls are all listed under the “Common” player tab.

Controls List

The Controls List displays the controls for the front-end. Select a control in the list to enable the edit buttons. Double-clicking a control in the list will display the [Control Edit](#) window. Using the column headers, you can increase or decrease the width of each column.

By default, LEDBlinky will include the known controls for your selected front-end. If you do not wish to light up a specific control, you may delete it from the list or just remove the color/intensity for that control.

Edit / New / Move / Delete

These buttons work the same as for the ROM/Game controls. See [above](#).

OK

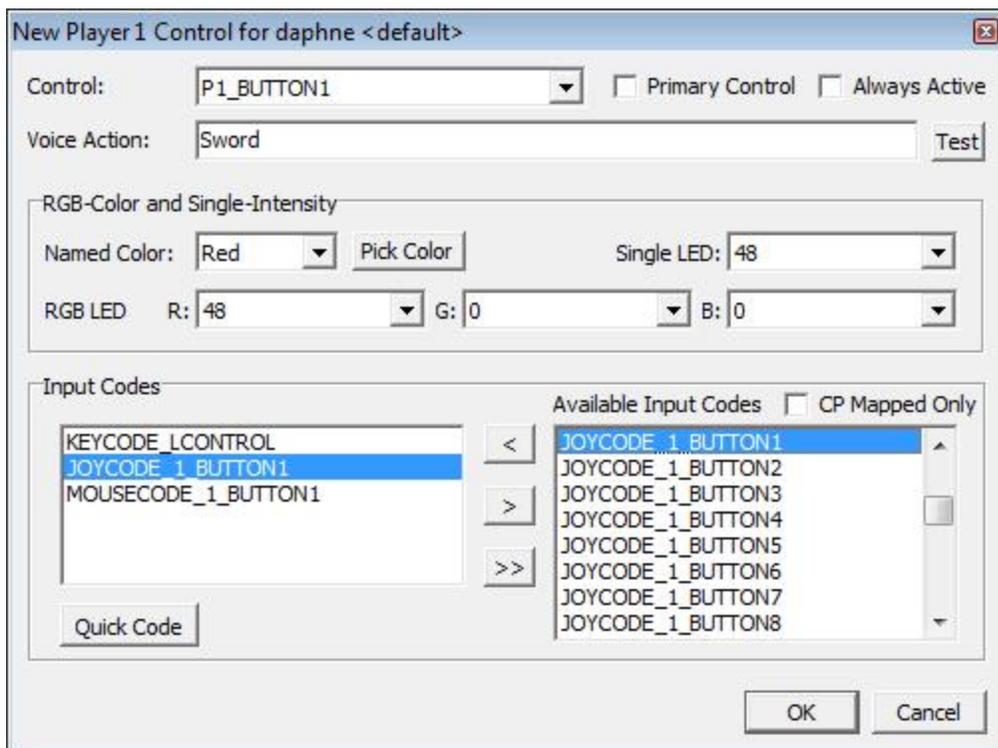
Click the “OK” button to accept the front-end control modifications. After clicking “OK”, the lower edit pane will be blank and the upper list panes will be re-enabled.

Note: Modifications are not saved to the *LEDBlinkyControls.xml* file until the “Save” menu option is clicked.

Cancel

Click “Cancel” to abandon all front-end control modifications. After clicking “Cancel”, the lower edit pane will be blank and the upper list panes will be re-enabled. You can also use the “ESC” key as a shortcut.

Edit Individual Control



Control

When adding a new control, select the “Control” name from the drop-down list. The list will only include controls that are available for the current player (1, 2, 3, 4, or Common).

Note: For non-MAME emulators the Control name has NO effect on which controls light up; only the Input Code(s) effect which controls light up.

Note: You should only add the same control more than once to specify a different action for combo-buttons. For example, if Button 1 and Button 2 can be pressed together for a special action, define each button twice; once for their unique action and once with the same action for the combo.

Primary Control

Check the “Primary Control” option to set the current control as the primary. Only one control for a game can be set as the primary. The “Name (Voice)” for the primary control is spoken separately from the rest of the controls (and it is not blinked).

The “Primary Control” is also used to determine the Joystick Digital Restriction mode for the Ultimarc UltraStik 360 (U360) and ServoStik, and older 49-Way joysticks.

Note: See here for a description on how to override the JDR mode.

Always Active

Check the “Always Active” option to insure that the control will always light up regardless if it is required by the game you are playing. This is useful for lighting administration buttons.

Note: The “Always Active” option is only available for the “Default” (<default>) controls group.

Voice Action

The “Voice Action” text will be used to speak the control name or action when the [option](#) is enabled. Click the “Test” button to hear how the text will sound using the current [Text To Speech](#) options. The Voice Action field will be disabled when the [Use Default Start/Coin Voice Actions For All Games](#) option is enabled.

Note: You may need to alter the spelling to have the name sound phonetically correct.

RGB-Color and Single-Intensity

Your control panel may have Single, RGB, or both types of LEDs. These have been defined using the [Generate LEDBlinky Input Map](#) application. If your control panel has only Single type LEDs, the RGB Color options will be disabled. If your control panel has only RGB type LEDs, the Single Intensity options will be disabled.

Note: If your control panel uses both RGB and Single type LEDs, you may want to define both Color and Single values for each control – conceivably you could remap a game function from a RGB LED button to a Single LED button or vice versa.

Named Color

A RGB color can be selected from the “Named Color” list. The individual RGB intensities will be set to match the selected color. The selected “Test RGB LED” will light up with the color.

Note: The “Named Color” option is only available if you are using an LED controller that supports variable intensities.

Pick Color

Click the “Pick Color” button to select a color from a pre-defined pallet. The individual RGB intensities will be set to match the selected color. If the selected color is also available as a named color, the “Named Color” value will be set. The selected “Test RGB LED” will light up with the color.

Note: RGB LEDs cannot accurately reproduce all the colors you see in the pallet. For example, gray tones do not look correct and black is impossible.

Note: The “Pick Color” option is only available if you are using an LED controller that supports variable intensities.

RGB LED Red / Green / Blue

The LED color can be defined by selecting the individual intensity values for the Red, Green, and Blue color components. The RGB LED selected on the [Test Options](#) menu will light up the specified color.

If your control panel uses any LED controller that supports variable intensities, then the intensity values will be 0 to 48. If your control panel uses any LED-Wiz controllers, then the intensity values will be 0 to 48, plus the built-in LED-Wiz blinking effects. If your control panel only uses LED controllers without variable intensities, then the intensity values will be *On* or *Off*.

Note: If you are using any blinking effects for RGB LEDs and your control panel has more than one LED controller, all three color leads for each RGB should be physically wired to the same LED controller. Wiring the colors across LED controllers will cause out-of-sync blinking effects with very strange results.

Single LED

Select the intensity value for the single color type LED. The Single LED selected on the [Test Options](#) menu will light up at the specified intensity.

If your control panel uses any LED controller that supports variable intensities, then the intensity values will be 0 to 48. If your control panel uses any LED-Wiz controllers, then the intensity values will be 0 to 48, plus the built-in LED-Wiz blinking effects. If your control panel only uses LED controllers without variable intensities, then the intensity values will be *On* or *Off*.

Input Codes

Defining the correct “Input Codes” for a control is critical to getting that control to light up! The “Input Codes” (keycodes) for each control must match those used by the emulator. For example; if Player 1, Button 1 is configured by the emulator to use keyboard button “A”, you must assign KEYCODE_A to the P1_BUTTON1 control.

Note: Defining “Input Codes” is only required for non-MAME emulator controls. MAME and Front-end input codes are pre-defined and the input code fields will be disabled.

Using MAME Default Input Codes Option

If you have checked the “Use MAME Default Control Mapping for Other Emulators” option, then all input code fields will be disabled. You can change this option from the Configuration app or click the “Question” button. See [here](#) for a full description of this option.

Quick Code

Click this button to quickly select and assign a keyboard or joystick input code. When the button is depressed, it will flash 'Press Button'; then press any button on your control panel or key on the keyboard. The input code will be selected and added to the Input Codes list. To abort the Quick Code mode, click the button a second time.

Note: The Quick Code feature cannot be used to assign analog joystick, mouse, or gun codes – these must be selected manually from the list.

Available Input Codes

Select an input code from the list of "Available Input Codes" and click the "<" button to add the value to the Input Codes list for the control. Use the ">" button to remove a single (selected) value or the ">>" button to remove all values from the Input Codes list for the control. Double-clicking an input code in either list will add or remove the value from the Input Codes list.

CP Mapped Only

Check the "CP Mapped Only" option to filter the "Available Input Codes" list to only input codes that you have configured on your control panel via the LEDBlinky Input Map. These are the only values you should be assigning to the controls.

OK

Click the "OK" button to accept the control modifications. Changes or additions will be reflected in the Control List.

Note: Modifications are not saved to the *LEDBlinkyControls.xml* file until the "Save" menu option is clicked.

Cancel

Click "Cancel" to abandon the control modifications. You can also use the "ESC" key as a shortcut.

How to Specify JDR Mode/Map Overrides

During normal operation, the “Primary Control” is used to determine the [Joystick Digital Restriction](#) mode for UltraStik 360 (U360) and 49-Way joysticks. Each primary control maps to a JDR mode/map. For example, all games assigned CONTROL_STICK as the primary control will use the u360 *Analog.um* map and/or the GP-Wiz49 Raw49 mode. (stick = analog? I know, it makes no sense). See [here](#) for a complete list of the Primary Control Mode/Map associations.

For the U360, you may wish to create your own maps using the UltraMap software provided by Ultimarc. Your own custom map can be substituted for any map included in the */LEDBlinky/JDR/UltraStik/* folder. For example, if you replace the *joy8way.um* file with your own version, all games that are assigned the CONTROL_JOY8WAY primary control will use your new map.

If you wish to assign a custom U360 map to one or more specific games, then follow these steps:

- 1) Put your new map file in the */LEDBlinky/JDR/UltraStik/* folder. Make sure it has a unique name – do not overwrite an existing map file.
- 2) Edit the LEDBlinkyControls.xml file using Notepad.
- 3) Locate the control group node for the game you wish to modify. For example;
`<controlGroup groupName="tron" numPlayers="2" alternating="1">`
- 4) Under the control node, locate the primary control node. If the primary control node does not exist, you must first add a primary control using the LEDBlinky Controls Editor. For example;
`<control name="CONTROL_JOY8WAY" primaryControl="1" />`
- 5) Add the UltraStik attribute and set equal to the name of your new map file. You do not need to include the “.um” file extension (but you can if you wish). For example;
`<control name="CONTROL_JOY8WAY" primaryControl="1" UltraStik="mymap" />`
- 6) Save and close the file.

MAME Output System

The MAME Output System is a mechanism by which MAME simulates game specific output signals. For example, some older games will flash the player start button(s) when credits are available. Other games may have had unique flashing buttons during game play. Additionally, MAME outputs are used when the emulator is started, when the emulator is quit, and when a game is paused.

LEDBlinky supports the MAME Output System through the use of the *MameOutputs.ini* file. The *MameOutputs.ini* file maps outputs to controls or LED controller ports. By default, LED0 and LED1 are mapped to Start1 and Start2 – if your player start buttons have LEDs, they will flash when credits are available for supported games (for example; Asteroids). Any other outputs you wish to use must be manually added to the *MameOutputs.ini* file.

The following MAME Outputs are most useful when used in conjunction with LEDBlink: LED0, LED1, LAMP0, LAMP1, LAMP2, ... LAMPn. To determine which (if any) outputs are generated by a game, use the [MAME Output Test](#) application (*MameOutputTest.exe*). Start the test app, then start MAME (or MAME32), then start any game – the outputs and associated controls will be listed whenever their state changes; On/Off.

The purpose of each output varies from game to game. Any MAME output state of zero (0) is considered OFF and a MAME output state of one (1) or greater is considered ON.

MAME Outputs can also be used to trigger [Pixelcade](#) commands.

MAME Outputs can also be used to trigger Web Requests (http or https GET). This could be used to send home automation commands such as turning lights on or off.

Any output can be added to the *MameOutputs.ini* file. To define outputs for a specific game, create an output group with the group name equal to the rom. For example, for Dig Dug the output group would be; [digdug]. The [default] group will be used for any game which does not have its own named output group.

Each output group consists of Key/Value pairs. Key/Value pairs must be formatted as follows:

```
<MAMEOutput> = <ControlName>
or
<MAMEOutput> = <LEDControllerType,LEDControllerID,Port,SingleIntensity>
or
<MAMEOutput> =
<LEDControllerType,LEDControllerID,Port,RedIntensity|
LEDControllerType,LEDControllerID,Port,GreenIntensity|
LEDControllerType,LEDControllerID,Port,BlueIntensity>
or
```

```

<MAMEOutput> =
<pixelcade, PixelcadeCommand, PixelcadeParameter, PixelcadeMAMEOutputState,
PixelcadeCommandTimer>
or
<MAMEOutput> =
<webrequest, WebRequestURL, WebRequestMAMEOutputState, WebRequestTimer>

```

LEDControllerType: LED Controller Name or Abbreviation. See the [Help Menu](#) for a list of supported LED Controller names and abbreviations. Example: LEDWiz or LW, PACLED64 or PL, IPACUltimateIO or IP, PACDrive or PD, Howler or HO.

LEDControllerID: This is the HardwareID value assigned to the LED Controller (LEDControllerType).

Port: First Port = 1. Maximum Port varies based on the LED Controller (LEDControllerType).

Intensity: 0-48, 129, 130, 131, 132 (values 129..132 are only supported by the GGG LEDWiz controller)

pixelcade: The first parameter for all Pixelcade commands must be "pixelcade".

PixelcadeCommand: One of the following Pixelcade commands; "user", "animation", "text", "console", "arcade", "clear", or "raw".

PixelcadeParameter:

For "user" PixelcadeParameter=<filename> from \Pixelcade\user folder.

For "animation" PixelcadeParameter=<filename> from \Pixelcade\animations folder.

For "text" PixelcadeParameter=text to display. Text color and scrolling speed will be set to values from Pixelcade configuration in LEDBlinkyConfig (Integrations Tab).

For "console", "arcade", and "clear" the PixelcadeParameter is not required. "console" displays the current console/emulator. "arcade" displays the current game/rom. "clear" clears the display.

For "raw" PixelcadeParameter=<Pixelcade API command>. See pixelcade.org/api V1 REST API. Do not prefix raw command with "http://localhost:8080".

PixelcadeMAMEOutputState (optional): 1 or 0. 1=on 0=off. LEDBlinky will send the Pixelcade command for the specified MAME output state. If not specified, command will be sent on 1(on) and clear on 0(off).

PixelcadeCommandTimer (optional): Timer value (specified in milliseconds).

Positive value; Throttle period during which Pixelcade commands will not be sent. This is useful for high frequency MAME outputs which may overload the Pixelcade display. After an output is received and a Pixelcade command is sent, all subsequent MAME outputs with Pixelcade commands will be ignored until the delay expires. The delay value should be slightly longer than the time it takes the Pixelcade to receive and process the command (display the image or animation).

Negative value; Delay until Pixelcade command is executed. If the same MAME output is received before the timer expires, the timer will start over. This is useful for updating the Pixelcade display after the MAME output activity has completed, for example, returning to the game image. The delay value should be longer than the MAME output repeat frequency.

Note: Some Pixelcade commands (such as animations) combined with high frequency MAME Outputs may cause the Pixelcade display to freeze or stop responding. In this case, use a positive PixelcadeCommandTimer value to throttle the repeat commands sent to the Pixelcade.

`webrequest`: The first parameter for all web requests must be "webrequest".

`WebRequestURL`: WebRequest URL must be prefixed with 'http' or 'https' and will be executed as a GET request.

`WebRequestMAMEOutputState`: 1 or 0. 1=on 0=off. LEDBlinky will send the WebRequest for the specified MAME output state.

`WebRequestTimer` (optional): Timer value (specified in milliseconds).

Positive value; Throttle period during which WebRequests will NOT be sent. This is useful for high frequency MAME outputs which may overload the web server. After an output is received and a WebRequest is sent, all subsequent MAME outputs with WebRequests will be ignored until the delay expires.

Negative value; Delay until WebRequest is executed. If the same MAME output is received before the timer expires, the timer will start over. This is useful for sending a WebRequest after the MAME output activity has completed. The delay value should be longer than the MAME output repeat frequency.

Note: Only three Pixelcade or WebRequest commands for each MAME output are supported (On, Off, and Delayed). If more than three are specified, prior values will be overwritten. Additionally, Pixelcade and WebRequest commands which specify a timer value are limited to two (2) timers (each), one for On/Off command and one for Delayed command. Any additional timer values specified will negate prior values. For example, if you have three Pixelcade commands with a timer value, only the second and third will be used.

Each MAME output can be repeated to assign more than one LED and/or Control and/or Pixelcade value. Spaces are allowed in the MAME Output (Key) and Pixelcade parameters (Value).

The *MameOutputs.ini* file has an example [sampleROM] MAME output group with key/value pairs.

LEDBlinky Output System

The LEDBlinky Output system provides a mechanism for custom or 3rd party hardware to consume and respond to LEDBlinky data (the same as the currently supported LED controllers). The only requirement is that the custom hardware monitors LEDBlinky data using TCP, UDP or Serial protocols.

To use the LEDBlinky Output system you must first create one or more LEDBlinky Output Controllers (virtual LED hardware controllers). From the [Generate LEDBlinky Input Map](#) app, use the [Add \(LED Controller\)](#) button. Select “LEDBlinky Output” from the LED Controller list and select the Controller ID.



For TCP or UDP protocols, the ID number is added to the [Base Port Number](#) to determine the outgoing Windows port used for data communication. For example, the default Base Port Number is 1024, so LEDBlinky Output Controller ID 1 would use port 1025.

For Serial/COM communications, the ID number is the COM Port number.

After adding the LEDBlinky Output Controller(s), the LEDBlinky Output system must be enabled and configured using the [LEDBlinky Configuration](#) app. On the Misc Options tab, check the [LEDBlinky Output](#) option. You must also choose the communications protocol, [TCP, UDP, or Serial](#). The communication protocol is dependent on the server (hardware) receiving the data. If you are building your own custom hardware (using a Raspberry Pi for example), UDP is faster and therefore recommended. For an Arduino, you could use the Serial/COM port.

After the configuration is complete, LEDBlinky (and any LEDBlinky support apps) will broadcast LED data using the selected protocol. For TCP or UDP protocols, LEDBlinky will send data to the local host IP (127.0.0.1) over the specified port (base port + ID). It is possible to configure an IP Address other than the local host (127.0.0.1) by manually editing the *settings.ini* file under the [LEDBlinkyOutput] group. For Serial communications, LEDBlinky will send data over the configured COM port(s).

Note: When using Serial communications, even at the highest baud rate (115200), animations should not exceed 30 frames per second.

LEDBlinky broadcasts data in JSON array format. There are only two data sets;

Single port: [`<port>`,`<intensity>`]

All ports: [`<intensity>`,`<intensity>`,`<intensity>`,...`<intensity>`]

The port range is 1-128 and the intensity range is 0-255.

For testing and to determine the data that LEDBlinky is broadcasting you can use the LEDBlinky Output Test app (*LEDBlinkyOutputTest.exe*). The test app acts as an LED controller and listens for data on the selected port using the configured protocol. When using TCP, the test app must be started before LEDBlinky (or any LEDBlinky support app) due to the required two-way handshaking. UDP does not utilize any handshaking and therefore the client and server can be started in any order.

Note: The LEDBlinky Output Test app does not work with the Serial communication protocol. Other third-party apps can be used to monitor Serial/COM ports for testing.

Appendix A: Supported LED Controllers

The LEDBlinky Configuration [Help Menu](#) has an option to display the list of supported LED Controller names and abbreviations.

Groovy Game Gear LED-Wiz

32 LED ports, variable intensity.

Ultimarc PacDrive

16 LED ports, on/off.

Ultimarc U-HID

16 LED ports, on/off

Ultimarc PacLED64

64 ports, variable intensity.

Ultimarc iPac Ultimate I/O

96 ports, variable intensity.

Ultimarc NanoLED

96 ports, variable intensity.

Note: The NanoLED firmware may require an upgrade for LEDBlinky to correctly identify the hardware. If your NanoLED is not running the required firmware, LEDBlinky will detect the NanoLED as a PacLED64. Please contact Ultimarc for the latest NanoLED firmware.

Note: The NanoLED performance may vary based on the type of LED array and the number of addressable LEDs in use.

Wolfware Tech Howler

96 ports, variable intensity.

Appendix B: File Descriptions

Colors.ini

This file defines game specific control colors (or intensities) for MAME. A version of the file with authentic control panel colors supporting over 1200 ROMs (at last count) has been included with LEDBlinky. An updated version may be available for downloaded from the ArcadeControls forum (although not likely).

Color-RGB.ini

This file contains a list of Color Names and the RGB (Red/Green/Blue) intensity values for each. Intensity values must be in the range of 0 to 48. Note: some colors cannot be reproduced by a RGB LED, for example; shades of Gray or Black. In these cases, substitute colors must be specified. This file is only used in conjunction with an optional *colors.ini* file for MAME.

Controls.ini

This file contains a list of MAME games and their associated details, options, and controls. Data in this file is considered supplemental to the *mame.xml* file, although LEDBlinky, when determining game controls, searches first in the *Controls.ini* file and second in the *mame.xml* file. *Controls.ini* was originally named *control.dat* and is a community provided file first defined by SirPoonga in 2007. An updated version may be available for downloaded from the ArcadeControls forum (although not likely).

GenLEDBlinkyInputMap.exe

Generate LEDBlinky Input Map Application. This app will allow you to generate or update the *LEDBlinkyInputMap.xml* file required by the LEDBlinky plug-in, Controls Editor, and Animation Editor. The Input Map file defines the connection between the LED controller ports, and the Input Codes (Keyboard, Trackball, Joysticks, etc). It also defines the three ports used by each RGB LED (if any). This mapping should never change unless you rewire your LED controller(s) or reprogram your keyboard encoder. See [here](#) for a complete description.

Hi2txt.exe

Hi2txt is a third-party app included with LEDBlinky to decode MAME high score files (.hi and .nvram). The version of Hi2txt included with the LEDBlinky Installation may not be the latest version. For Hi2txt support and to download the latest C# version and database of supported games see; <https://greatstoneex.github.io/hi2txt-doc/>

Hi2txt.zip

Hi2txt database of supported games.

HowlerDLL.dll

Howler Function Library. Provides the interface to the Wolfware Tech Howler LED controller.

LBkbh.dll

This dll provides LEDBlinky with the ability to abort the Game Start and Game Pause speech features by pressing any key. Keyboard monitoring is only active when LEDBlinky is speaking the Game Start and Game Pause options. No keystrokes are recorded or retained.

LEDBlinky.exe

Core LEDBlinky application. This app runs in the Windows system tray . It accepts commands using [command-line](#) parameters or via a plug-in interface. See [here](#) for a complete description.

LEDBlinky.mplugin

This is the MaLa plug-in.

LEDBlinky.nut

This is the Attract-Mode plug-in.

LEDBlinky.atoplug**LEDBlinky.plugcfg**

This is the AtomicFE plug-in and plug-in configuration file.

LEDBlinky_GX.dll

This is the GameEx 32-bit plug-in (dll).

LEDBlinky_GX64.dll

This is the GameEx 64-bit plug-in (dll).

LEDBlinky_PX.dll

This is the PinballX 32-bit plug-in (dll).

LEDBlinky_PX64.dll

This is the PinballX 64-bit plug-in (dll).

LEDBlinkyAnimationEditor.exe

LEDBlinky Animation Editor Application. This app will allow you to create or modify LED Animations. Although the Animation Editor can be used without the LEDBlinky plug-in, it does require an *LEDBlinkyInputMap.xml* file which can be generated using the [LEDBlinky Input Map Application](#). See the *Animation Editor.pdf* file for a complete description.

LEDBlinkyConfig.exe

LEDBlinky Configuration Application. This app will allow you to configure the LEDBlinky plug-in. It also provides quick access to the other LEDBlinky tools. See [here](#) for a complete description.

LEDBlinkyConfigWizard.exe

LEDBlinky Configuration Wizard. This app will guide you through the process of creating your input map and setting other required options with step-by-step questions. It provides a basic setup and is recommended for anyone using LEDBlinky for the first time. Using the Configuration Wizard is optional.

LEDBlinkyControls.xml

This file stores all LEDBlinky control related information – this includes Emulator controls, MAME control overrides, Front-end controls, control defaults, and other stuff. You should never need to edit this file manually and doing so may result in LEDBlinky errors. The *LEDBlinkyControls.xml* file is modified by the [LEDBlinky Controls Editor](#).

LEDBlinkyControlsEditor.exe

LEDBlinky Controls Editor. This app allows you to configure the controls for any Emulator, Game, or Front-end. For each control you can specify the color or intensity, the voice 'action', and if necessary the input codes associated with the control. Controls can be defined as the default for an emulator, or individually for a specific game. See [here](#) for a complete description.

LEDBlinkyTroubleshooter.exe

The LEDBlinky troubleshooting app will attempt to provide solutions for common issues such as the wrong buttons lighting up during game play. The app is self-explanatory and may provide additional information to help resolve your problem.

Ledwiz.dll

LED-Wiz Function Library (Created by MikeQ). Provides the interface to the Groovy Game Gear LED-Wiz LED controller.

LibUSB0.dll

Used exclusively by the UltraStik U360 Control library (ultrastik.dll). If you do not have any U360 joysticks, then this file may be deleted along with the /jdr/UltraStik folder.

Mame.dll

MAME Outputs Function Library (Ceated by headkaze © Copyright 2009 Headsoft www.headsoft.com.au). Provides the output interface to MAME (version .112 or later).

MameOutputs.ini

LEDBlinky uses the *MameOutputs.ini* file to map outputs to controls or LED controller ports. By default, LED0 and LED1 are mapped to Start1 and Start2 – if your player start buttons have LEDs, they will flash when credits are available for supported games (for example; Asteroids). Any other outputs you wish to use must be manually added to the *MameOutputs.ini* file.

MAME Outputs can also be used to trigger Pixelcade commands or execute a Web Request (http or https GET).

See the [MAME Output System](#) or the *MameOutputs.ini* file for a full details on how to configure MAME Outputs for use with LEDBlinky.

MameOutputTest.exe

This application will list all outputs generated by a MAME game. Start the application, then start MAME (or MAME32), then start any game. Outputs will be listed in real-time as they change state; On/Off.

NewInputCodes.ini

This file is used to add new MAME Input Codes. You should only edit this file when a new release of MAME adds new input codes. Each Input Code must be mapped to an existing LEDBlinky Base Input Code. Base input codes are defined in *LEDBlinkyControls.xml* file under the `<baseInputCodes>` node.

PacDrive32.dll

Ultimarc Function Library (Created by headkaze © Copyright 2009 Headsoft www.headsoft.com.au). Provides the interface to all Ultimarc LED controllers, and also ServoStik joysticks.

ServoStikTest.exe

ServoStik test app. Use this app to confirm LEDBlinky is detecting and communicating with all ServoStik joysticks.

Settings.ini

This file stores all LEDBlinky configuration settings. Use the LEDBlinky Configuration app to modify this file – do not edit manually.

SimpleLEDTest.exe

LED controller test app. Use this app to confirm that you LED controllers are functioning and wired correctly.

tos428.dll

GRS tos428 Function Library (Created by Stefan Burger of GRS). Provides the interface to the GRS tos428 Restrictor (for Sanwa joysticks).

tos428Test.exe

GRS tos428 test app. Use this app to confirm LEDBlinky is detecting and communicating with all GRS tos428 joystick restrictors.

UltraStik32.dll

UltraStik UltraStik 360 Function Library (Created by headkaze © Copyright 2009 Headsoft www.headsoft.com.au). Provides the interface to Ultimarc UltraStik 360 joysticks.

UltraStikTest.exe

UltraStik test app. Use this app to confirm LEDBlinky is detecting and communicating with all UltraStik joysticks.

ZipDll.dll

Freeware zip library designed by Eric Engler. Available for download here:

<http://www.geocities.com/SiliconValley/Network/2114/>

Also see: <http://www.info-zip.org/>

Appendix C: Credits

I'd like to thank the following persons for their help (directly and indirectly) with the LEDBlinky Arcade LED Control software and Animation Editor.

- Swindus and Loadman – The developers of MaLa; an excellent FE!
- Youki – The developer of AtomicFE; another excellent FE!
- Tom S. – The developer of GameEx and PinballX; two more excellent FEs!
- BadBoyBill. – The developer of HyperSpin; yet another excellent FE!
- Loadman – The pioneer of many MaLa plug-ins. Thanks for all your support and for not getting pissed-off when I decided to release a “similar” plug-in.
- Headkaze – Thanks for providing the colors.ini file – with that data, LEDBlinky can light up the buttons in their original arcade colors! Also, thanks for the PacDrive dll, Mame dll, UltraStik dll, and other code support.
- Kevin Jonas (SirPoonga) and Howard Casto – Thanks for providing Controls.dat. Without that data, many of LEDBlinky’s cool features would not be possible.
- Jason Carr – The developer of LaunchBox and BigBox; another excellent FE!
- Al Linke – The developer of Pixelcade.
- Pierre Monnot (GreatStone) – The developer of Hi2txt.
- phulshof – The developer of RetroFE.
- BritneysPAIRS – The developer of CoinOps.
- Glen Planamento and Stefan Burger of GRS (Glen’s Retro Show) – developers of the tos429 joystick restrictor.
- And many thanks to all the other forum members who have suggested new LEDBlinky features!

Index

- Abort Speech, 27
- Add (LED Controller), 17
- All Available Input Codes, 19
- All LEDs Off During Game Start Delay, 27
- Alternating, 68
- Always Active, 74
- Amp Mult, 42
- Amplitude Animation, 40
- Audio Animation Mode, 39
- Audio Device, 39
- Audio Input, 39
- Available Input Codes, 76
- Base Port Number, 47
- Baud Rate, 47
- Berzerk Mode, 31
- Blink Controls, 39
- Cabinet LEDs, 44
- Cabinet LEDs Animation, 39
- Cancel, 65, 70, 72, 76
- Clear Port, 19
- Cocktail Table Mode, 47
- CoinOps Screen Saver Selects Random Game, 34
- Color Adjust, 18
- Color/Intensity Spectrum, 39
- Color-RGB.ini, 84
- Colors.ini, 36, 84
- Common Tab, 68, 71
- Control, 73
- Control Drop-down List, 41
- Control Panel Max Players, 47
- Controller File, 36
- Controls Editor, 21
- Controls List, 69, 71
- Controls.ini, 36
- Copy Player 1 Controls, 58
- Copy ROM/Game, 58
- CP Wired Only, 76
- Dead Zone, 43
- Decay, 43
- Default, 65, 68
- Default Active/Inactive Controls, 69
- Delete (Control), 69
- Delete (Emulator), 62
- Delete (LED Controller), 17
- Delete (ROM/Game), 63
- Demo Duration, 33
- Demo Game Controls, 33
- Demo Include Always Active Controls, 33
- Demo Scroll Delay, 33
- Description, 65
- Display Decay, 42
- Display Errors On FE Quit, 49
- Display Format, 55
- Display Peaks, 43
- Do Not Speak Joystick Actions, 48
- Edit, 20
- Edit (Control), 69
- Edit (Emulator), 61
- Edit (LED Controller ID), 17
- Edit (ROM/Game), 62
- Edit / New / Move / Delete, 72
- Emulator, 64
- Emulator Is/Is Not MAME, 66
- Emulator List, 61
- Enable Debug Log, 48
- Enable JDR Mode, 50
- Event Game Active – Continuous Cycle Between Text and Game/Animation/User Image, 54
- Event Game Active – Game Image Static Only, 54
- Event Screen Saver – Text and Image Mode, 53
- Event Screen Saver – Ticker Mode, 53
- Exclude Selected Animations From Random, 48
- FE, 21, 62
- FE Active Animation, 29, 39, 45, 46
- FE List Change Animation, 29, 46
- FE Quit Animation, 30
- FE Start-Up Ani(mation) Duration, 29
- FE Start-up Animation, 28, 45, 46
- FE System Lists Names, 30
- Flash Player Start with Credits + Other MAME Outputs, 25
- Speak On Game Pause, 25
- Game Pause Animation, 24, 39
- Game Play Animation, 24, 39
- Game Play Animation For Jukebox Only, 24
- Game Start Animation, 25
- Game Start Delay, 27
- GameEx Screen Saver Starts Random Game, 34
- Generate Input Map, 21
- GenLEDBlinkyInputMap.exe, 84
- Global Pulse, 70
- GRS tos428 Restrictor, 51
- Help, 22
- Hi2txt, 26, 52
- Hi2txt Folder, 36
- HyperSpin Settings.ini, 33
- Ignore GameQuit Commands, 66
- Import MAME Games, 59
- Import Unknown Games, 58
- Input Codes, 75
- Jukebox, 69
- Keyboard ID, 20
- LBkbh.dll, 85
- LED Animation Editor, 21
- LED Controller List, 17
- LED Controller(s), 45
- LED Type, 18
- LEDBlinky Output System, 46, 81
- LEDBlinky.atoplug, 85
- LEDBlinky.exe, 85
- LEDBlinky.mplugin, 85
- LEDBlinky.nut, 85
- LEDBlinky.plugin, 85
- LEDBlinky_GX.dll, 85
- LEDBlinky_GX64.dll, 85
- LEDBlinky_PX.dll, 85
- LEDBlinky_PX64.dll, 85
- LEDBlinkyAnimationEditor.exe, 85
- LEDBlinkyConfig.exe, 85
- LEDBlinkyControls.xml, 86
- LEDBlinkyControlsEditor.exe, 86
- Ledwiz.dll, 86
- LibUSB0.dll, 86
- Light AtomicFE Controls, 32
- Light Attract-Mode Controls, 32
- Light CoinOps Controls, 32
- Light Game Controls, 23
- Light GameEx Controls, 32
- Light HyperSpin Controls, 32
- Light LaunchBox Controls, 32
- Light MaLa Controls, 32
- Light Maximus Arcade Controls, 32
- Light PinballX Controls, 32
- Light Player Start And Coin Buttons, 23

- Light RetroFE Controls, 32
- Lock Frequency, 41
- Lock Threshold, 41
- MAME Config, 35
- MAME Output System, 78
- MAME Output Test, 21
- Mame.dll, 86
- MAME.xml, 35
- MameOutputs.ini, 86
- MameOutputTest.exe, 87
- Maximus Default.ini, 34
- Move Control Up/Down, 69
- Name (Voice), 68, 74
- Named Color, 74
- New (Control), 69
- New (Emulator), 62
- New (ROM/Game), 63
- NewInputCodes.ini, 87
- No LEDs Mode, 47
- Num Players, 68
- OK, 65, 70, 72, 76
- Output Protocol, 47
- Pacdrive.dll, 87
- Pick Color, 74
- Pixelcade, 51
- Pixelcade Default Text for Emulator, 54
- Pixelcade Default Text for Game, 54
- Pixelcade Event Display System/Game Image When Scrolling FE Lists, 53
- Pixelcade Event Image Display Repeat / Time, 53
- Pixelcade Event Image Type, 52
- Pixelcade Event Scroll Delay, 53
- Pixelcade Event Select Image, 53
- Pixelcade Event Tabs, 52
- Pixelcade Event Text, 52
- Pixelcade Event Text Scroll Repeat, 52
- Pixelcade Folder, 51
- Pixelcade Text Color, 54
- Pixelcade Text Scrolling Speed, 54
- Player 1/2/3/4 Tabs, 68
- Port, 17
- Port Input Codes, 18
- Port Label, 18
- Preload Audio Library, 48
- Preload MAME Data Files On Startup, 37
- Primary Control, 73
- Pulse Animation, 39
- Quick Code, 18, 76
- Random Text Delay, 31
- Restrictor Plate Installed, 51
- RetroFE Screen Saver Selects Random Game, 34
- RGB LED Red / Green / Blue, 75
- RGB-Color and Single-Intensity, 74
- ROM / Game, 67
- ROM / Game List, 62
- Run Animation When Game is Active, 45
- Run Command, 55
- Run External Applications, 54
- Run External Applications Event Tabs, 54
- Sample Rate (ms), 43
- Save, 20, 21, 58
- Save Unknown Games, 47
- Screen Saver Animation, 30, 39
- ServoStikTest.exe, 87
- Set Port, 19
- Settings.ini, 87
- Show Color Adjustments, 20
- Single Amp Range, 40
- Single LED, 75
- Speak And Blink AtomicFE Controls, 33
- Speak and Blink Controls On Game Start, 26
- Speak And Blink GameEx Controls, 33
- Speak And Blink HyperSpin Controls, 33
- Speak And Blink LaunchBox Controls, 33
- Speak And Blink MaLa Controls, 33
- Speak And Blink Maximus Controls, 33
- Speak And Blink PinballX Controls, 33
- Speak And Blinky Attract-Mode Controls, 33
- Speak And Blinky CoinOps Controls, 33
- Speak And Blinky RetroFE Controls, 33
- Speak Name On Game Start, 25
- Speak on Game Pause; High Score, 25
- Speak On Game Start; High Score, 26
- Speak Primary Controls On Game Start, 26
- Speak Random Text, 31
- Speak Text On FE Quit, 29
- Speak Text On FE Start, 29
- Speak Text On Game Start, 27
- Speak/Blinky On Game Start; Start and Coin Btns, 26
- Spectrum Colors, 40
- Spectrum Mode – Color Fade, 40
- Spectrum Mode – Color Transition, 40
- Speech Rate, 44
- Speech Voice, 44
- Speech Volume, 44
- Spoken Controls Prefix, 27
- Strobe LEDs with Speech, 27, 29, 30
- Supported LED Controllers, 22
- Test, 39
- Test LEDBlinky, 59
- Test Options, 60
- Tools, 21
- tos428.dll, 87
- tos428Test.exe, 87
- Trigger Frequency, 43
- Trigger Mode – Increase Amplitude, 41
- Trigger Mode – Trigger Threshold, 41
- Trigger Threshold, 42
- UltraStik 360, 50, 51
- UltraStik.dll, 87
- UltraStikTest.exe, 87
- Use Default Start/Coin Voice Actions For All Games, 48
- Use MAME Default Control Mapping for Other Emulators, 36
- Use MAME to Trigger the Game Start/Stop Events, 37
- Use ROM/Game As Speech Default When Name (Voice) Not Specified, 65
- Using MAME Default Input Codes Option, 75
- ZipDll.dll, 88