Breezy4Pi User Manual

Revision 0.80

Breezy4Pi is built on the concept that the Raspberry Pi and those custom and off the shelf control boards that attach to the Pi all have I/O that can be configured and mapped into triggers and components using a web based management tool. That you, the user, can use this tool to easily create programs that can manipulate components, such as relays, motors, and LEDs, and that inputs, such as switches and optical sensors can be mapped to event triggers to run these macros when they change state.

Now, that you have Breezy4Pi installed on your Raspberry Pi, let us walk through the application to get you familiar with how to map a circuit board, and to create the macros and event triggers needed to read inputs from the external world and run devices.

Defining Board Templates

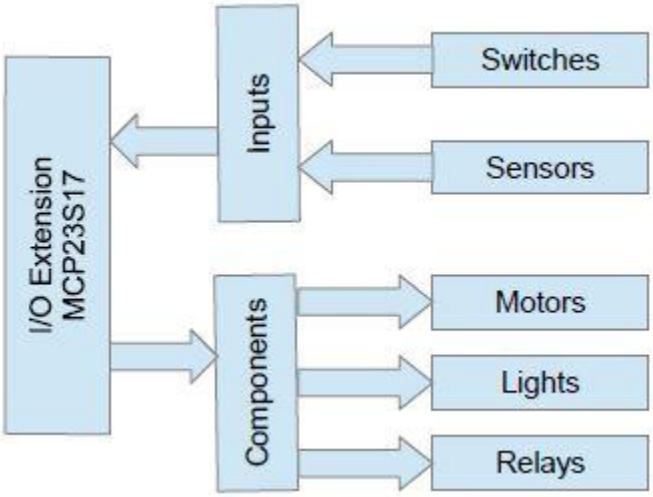
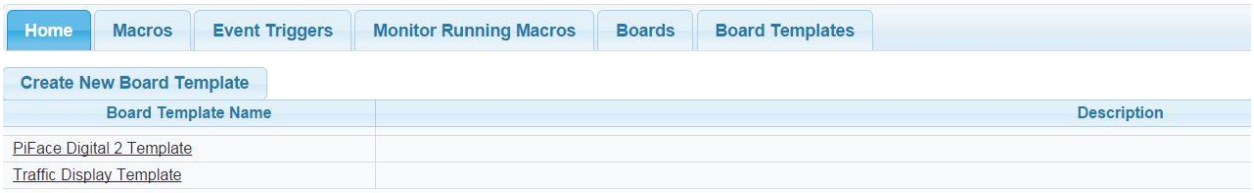


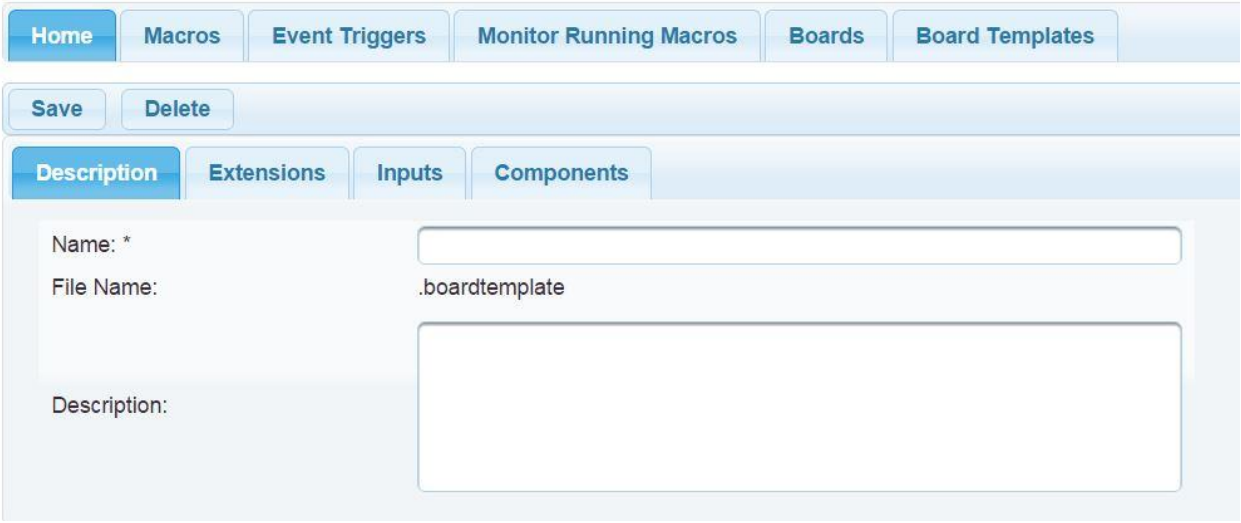
Figure 1 - Board Template Schematic

Figure 1 above shows how a typical add-on board for the Raspberry Pi may look. There is the I/O Extender, in this case an MCP23S17 16 bit extender. The extender is configured to accept both inputs, i.e switches and sensors and to drive output components, such as motors, lights and/or relays.

Navigate to the 'Board Templates' tab to begin creating a new board template. There, you will see a list of board templates that have been previously defined. Click 'Create New Board Template'.



Enter the name of the template and an optional description. If you click 'Save' at this point, a unique filename will be generated. This will be the filename of the template as it is stored on disk.



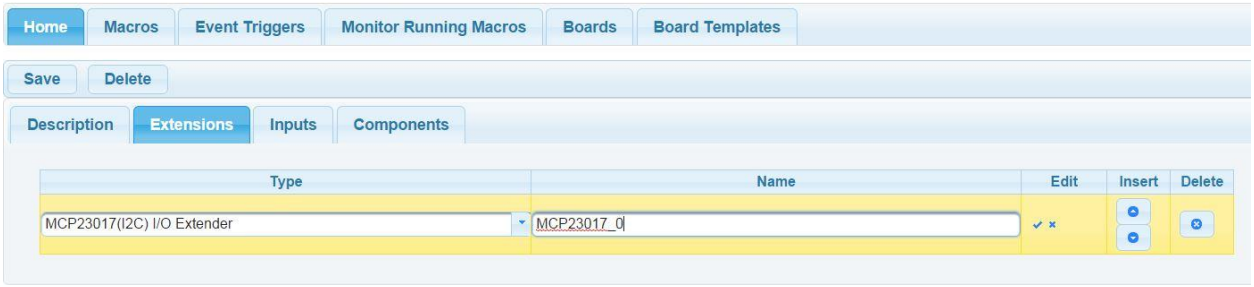
Even though you could try creating inputs and components at this point, it will not be too useful as both inputs and components are always mapped to an extension. So, click on the 'Extensions' tab then 'Define First Extension' to create a new extension.

You will now see a blank extension line. Click on the pencil icon in the Edit column to bring enable the line for editing. The up and down arrows in the 'Insert' column are for creating new Extension lines. You can define as many extensions as you want. The 'X' button in the 'Delete' column will remove the extension line.

Choose an extension from the drop down list and give it a meaningful name. Only Extension s that Breezy4Pi support shall be listed here. The 'System' extension refers to the Raspberry Pi itself.

Whatever extension you choose shall allow you to start mapping its GPIO pins to Inputs and Components.

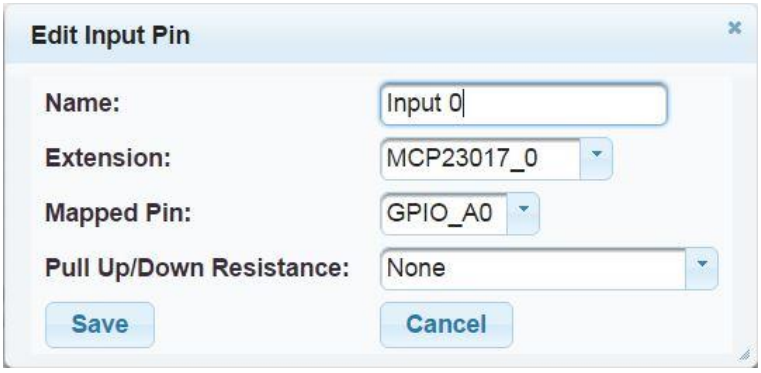
Click on the check mark to save the edit and close the edit box.



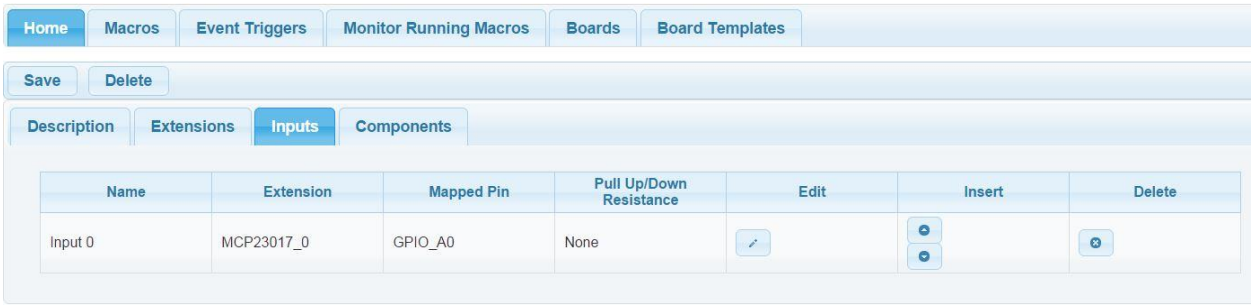
When you have all your extensions defined, you may now map the Inputs. Click on the 'Inputs' tab then click on 'Define First Input'. An Edit Input Pin mapping dialogue shall pop up.

Enter a meaningful name for this pin. Choose a previously defined extension. The Mapped Pin drop down shall auto populate with the valid GPIOs for this extension. Choose one, then select the desired Pull Up/ Pull Down/ None resistance for this pin. Some I/O extenders have built in pull up/down resistors that may be used. In this case, select Pull Up or Pull Down. If the board you're making a template of uses external pull ups/downs or doesn't have pull ups/downs, then select 'None'.

Click 'Save' to save changes and close the pop up.



Your changes now appear as a line in the Inputs screen. You shall see ‘Up/Down’ and ‘X’ buttons just like the in the Extension s edit screen in the Insert and Delete columns respectively. If you wish to change existing values, click on the pencil icon in the Edit column.

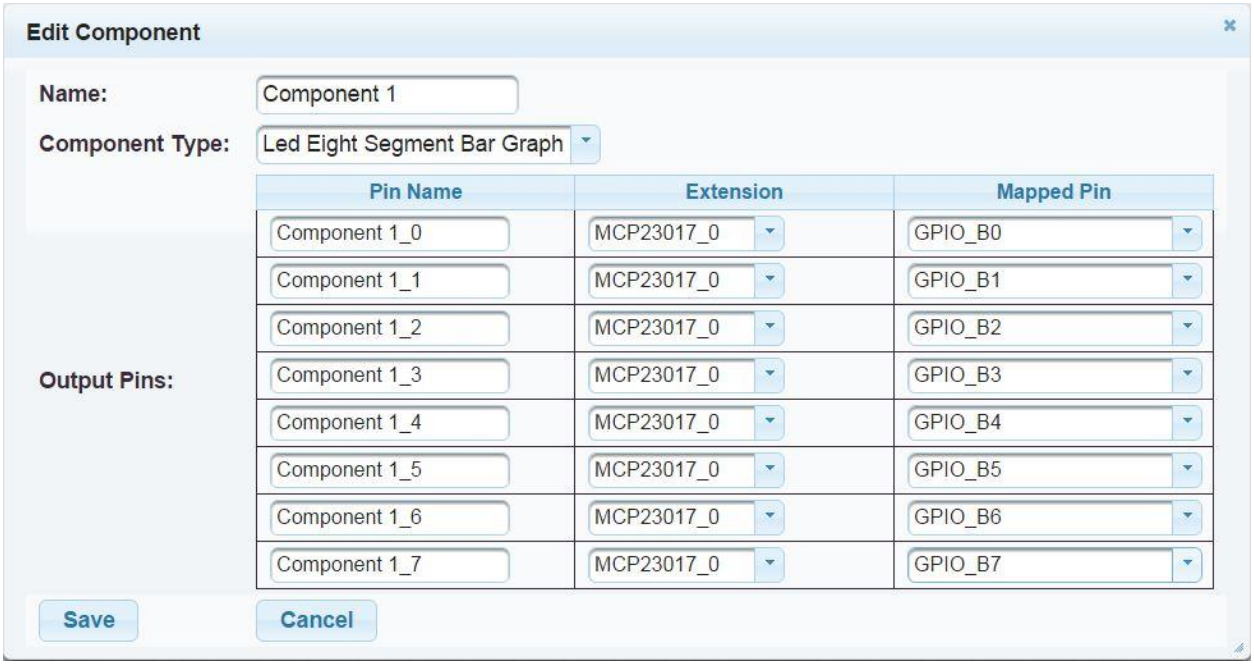


To define new components for this board template, click on ‘Components’ then ‘Define First Component’. The Edit Component dialogue shall pop up.

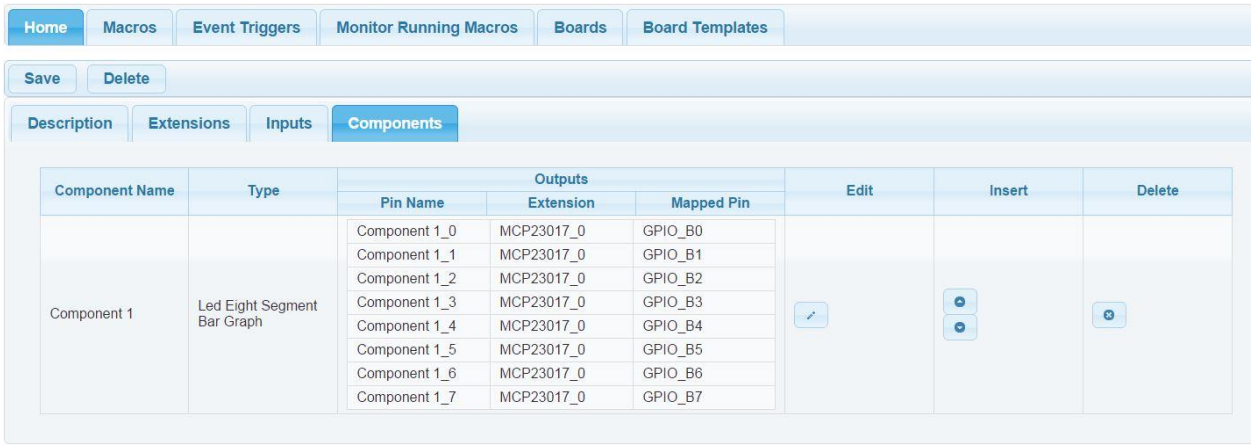
Enter a meaningful name for this component then select the type of component you wish to use. Each component type can map from one to many GPIO pins across one to many extensions. Once you select the desired component type the Output Pins section will auto populate with edit lines for each pin this component use.

For each pin, edit the Pin Name, select the extension the pin maps to and the extension's mapped pin.

Click 'Save' to save changes and close the dialogue. 'Cancel' will close the dialogue without saving changes.

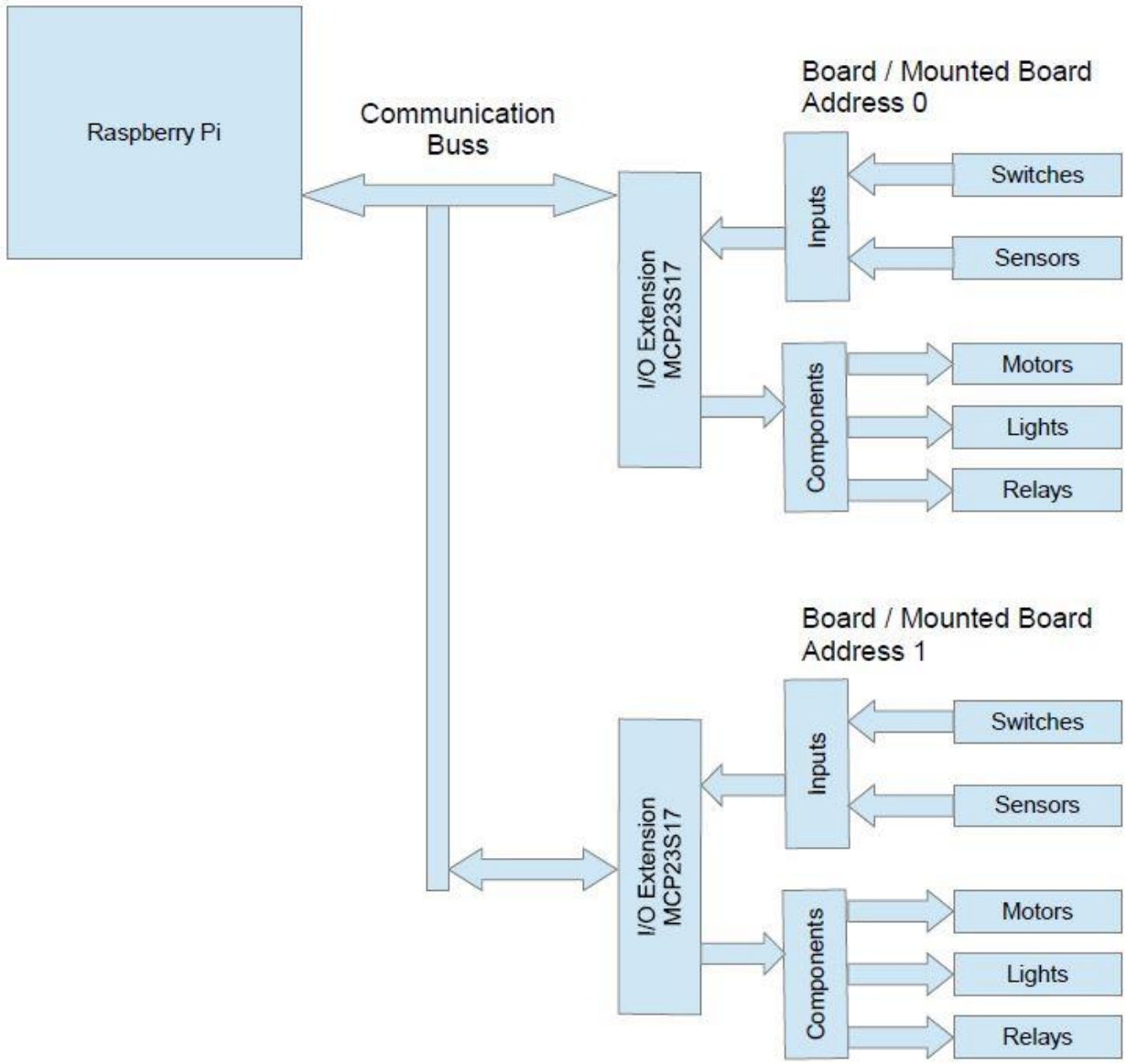


Once you've defined a component, then it will show up in the component page. 'Edit', 'Insert', 'Delete' buttons all work as on other edit screens.



When you're done mapping your board template, click the main 'Save' button to save changes and return to the Board Template list. Clicking 'Delete' will delete the template you are looking at and return you to the board template list.

Defining Boards and Mounted Boards

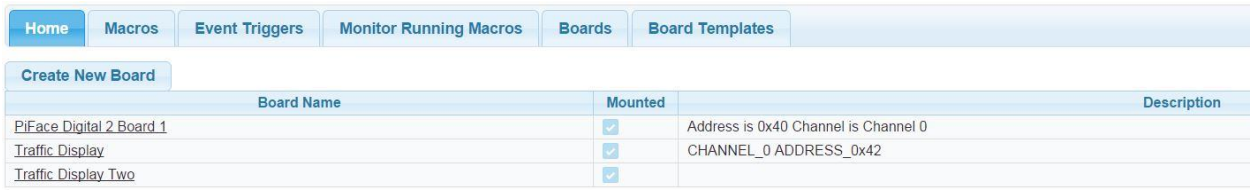


2 Boards and Mounted Boards Schematic

The above figure shows how the Raspberry Pi is connected to two identical boards. Each of these boards is derived from the same board template that was defined earlier. The only difference is that each board has a different address that allows the Raspberry Pi to properly control them individually.

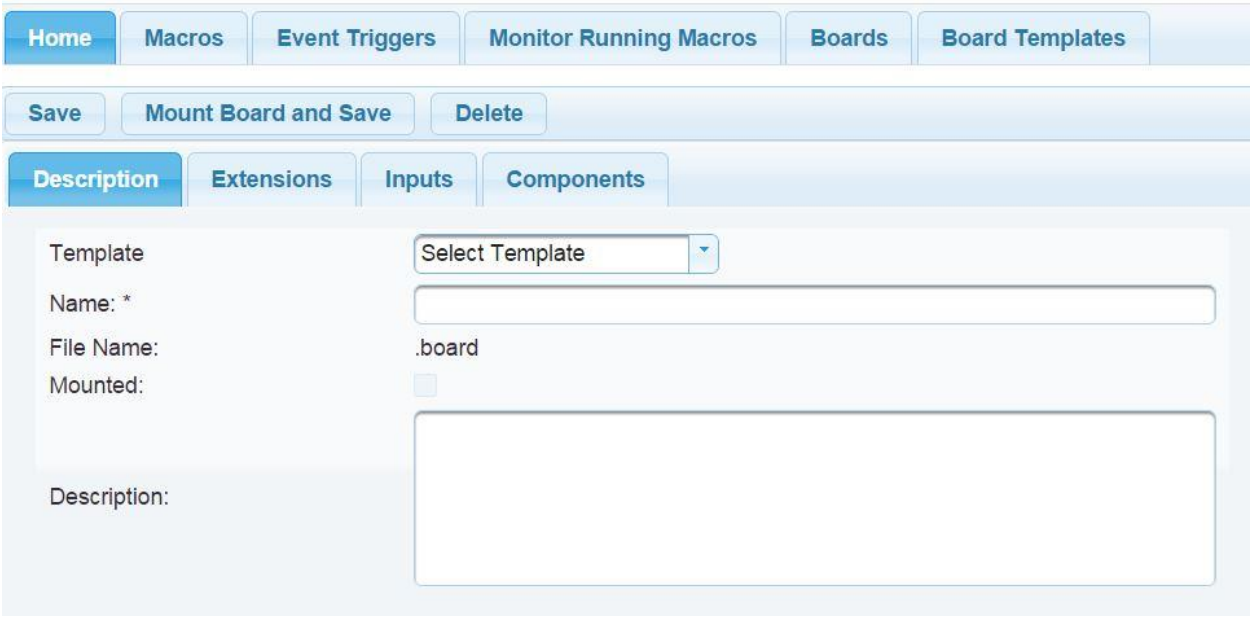
In reality, Breezy4Pi can map any number of different kinds of boards that are connected to the Raspberry Pi as long as Breezy4Pi supports the I/O Extension .

Click on the 'Boards' tab to create a new board or edit an existing one. This will take you to the list of boards that are currently defined.

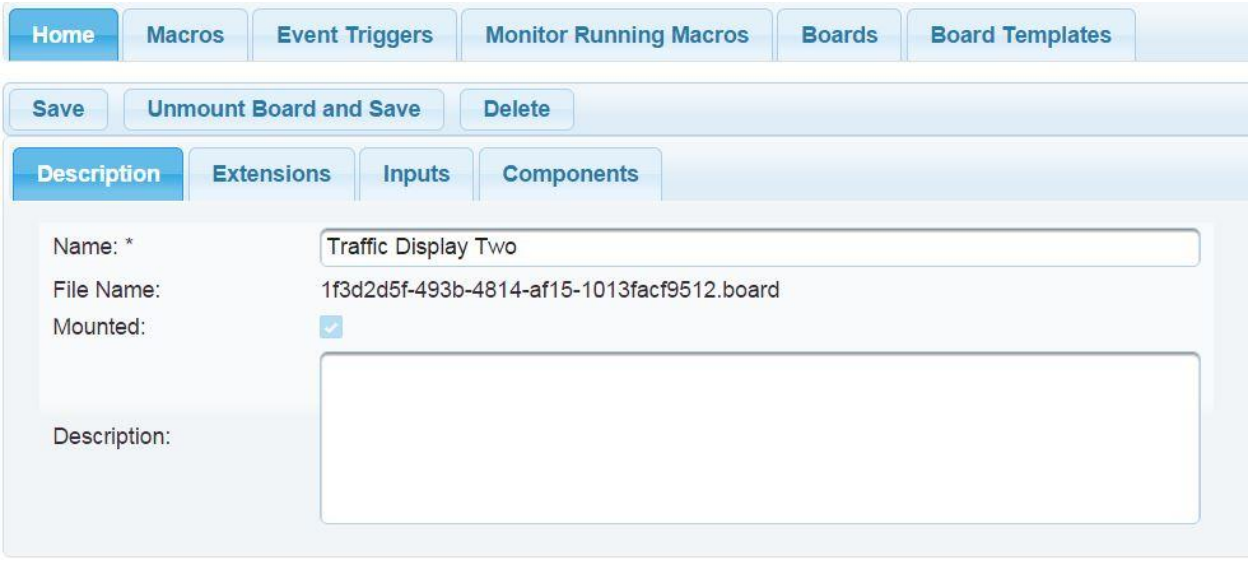


Click on 'Create New Board'. Enter the name of the board. This is the name that shall display in the list of boards and be available for choosing with in the macro and event trigger editors.

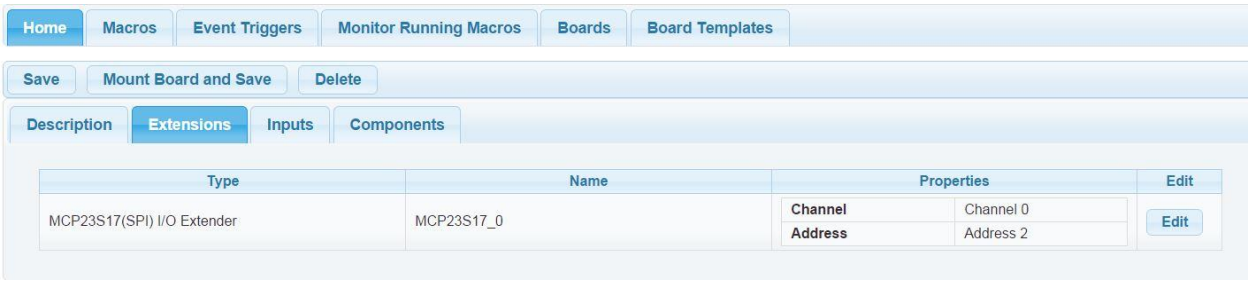
Select the template that was previously defined. If you click 'Save' then a filename will be generated. This is the name of the board definition file that will be found on disk.



The following graphic shows how this screen appears when you select a previously defined board.

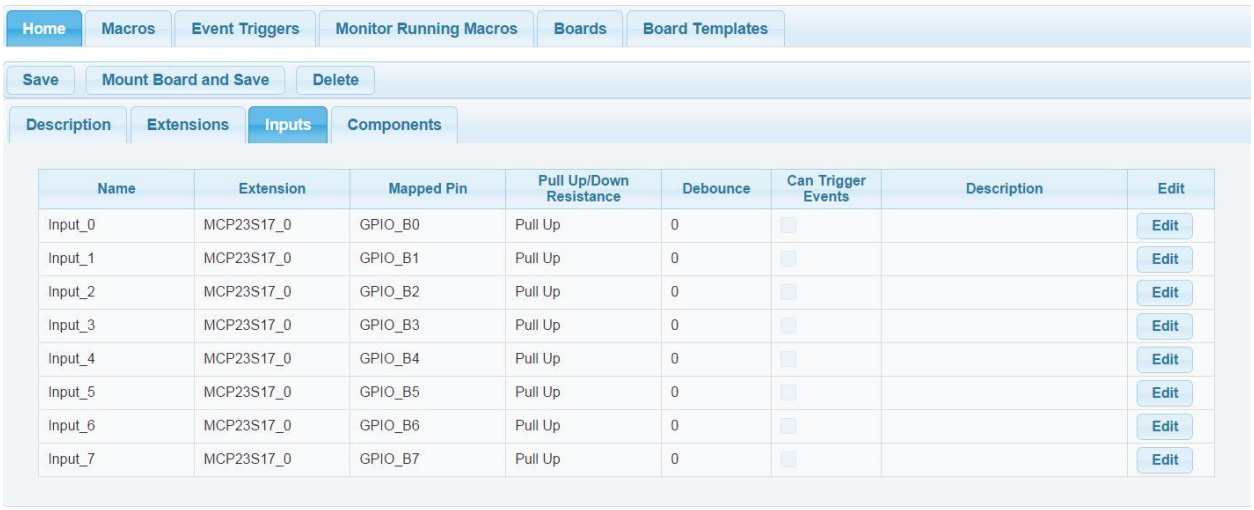


Now click on the 'Extensions' tab to define the unique parameters for the extensions defined in this board. You will see the extension that was defined in the board template. Notice the properties column. This contains the parameters that are defined for this extension, in this case a channel and an address where the Raspberry Pi can find this extension.



Click on the Edit button to edit the parameters for this extension. For each parameter, there will be a drop down of available parameter values. Click 'Save' to save edits.

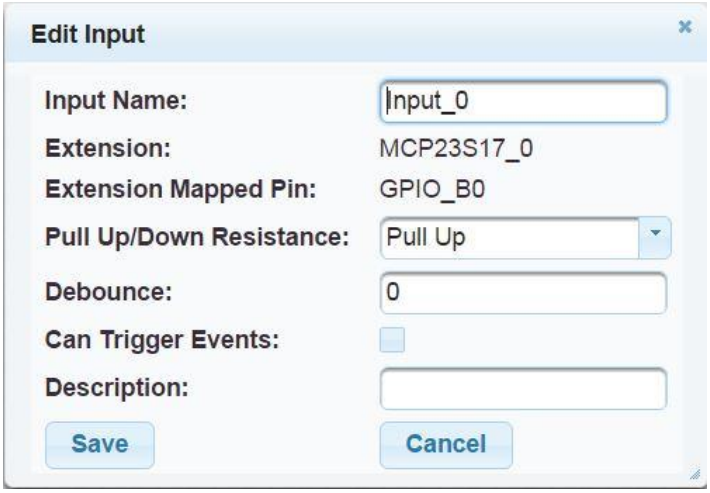
Some values are editable for the mapped inputs. Click on the 'Inputs' tab to view the inputs defined for this board. Click on 'Edit' to pop up a dialogue box to edit the values.



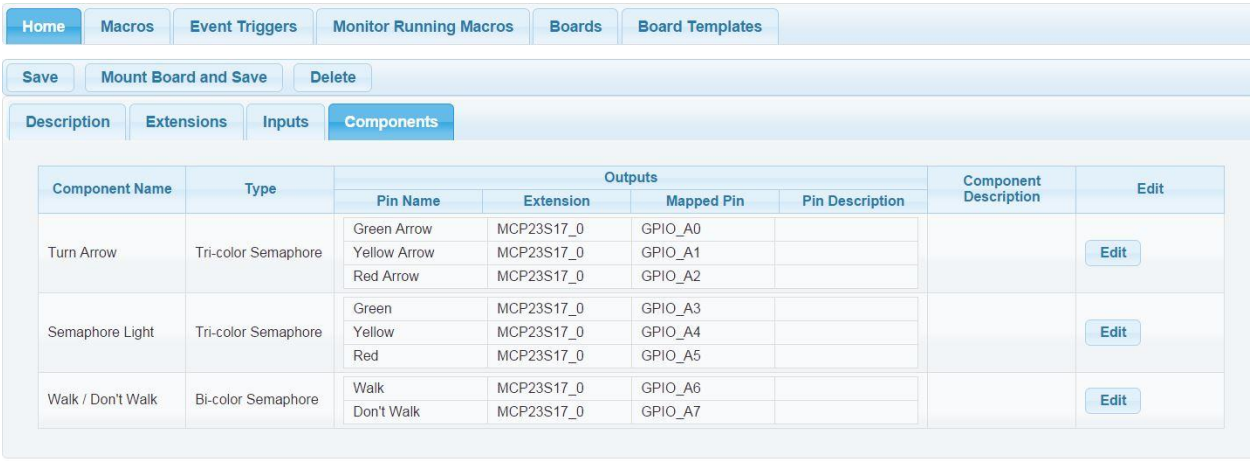
You are able to change the name of this input, change its pull up/pull down resistance which is useful for prototyping. You can also add a debounce value in milleseconds to debounce switches if needed and provide a description of this input is used in the description field.

Checking the 'Can Trigger Events' tells Breezy4Pi that this input is available to trigger the execution of a macro for certain state changes, i.e. low to high, or high to low.

Click 'Save' to save edits.



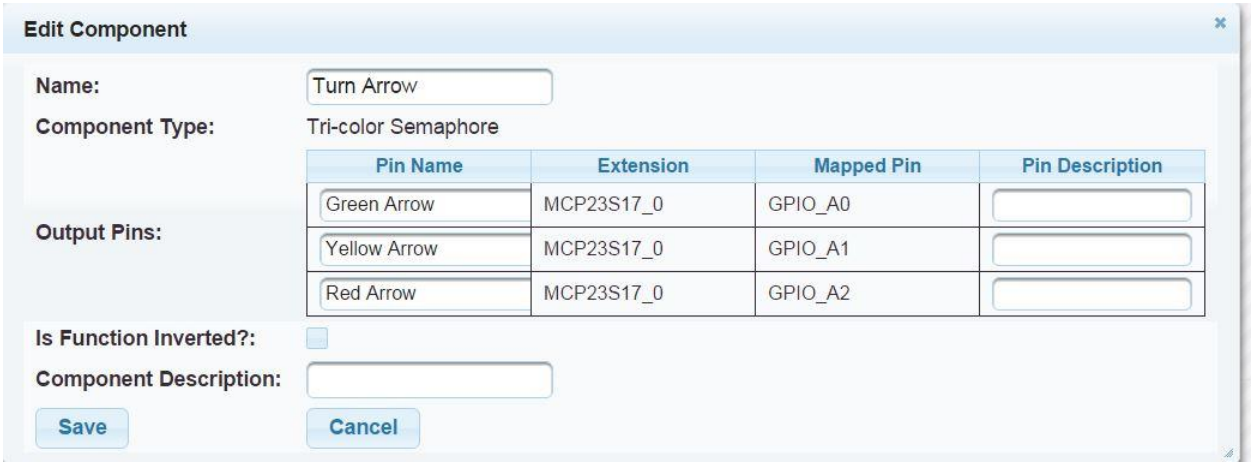
Clicking the 'Components' tab allows viewing and editing of the components that were defined in the board template. Click the 'Edit' button to pop up the component editor.



You can edit the name of the component and add notes (description) to the component itself or to its individual outputs.

Checking the 'Is Function Inverted' checkbox is not currently supported will be in a future release.

Click ‘Save’ to save edits.



Click on the main ‘Save’ button to save the board definition. Clicking ‘Delete’ will remove this board definition.

Mounting and Unmounting Boards

When you first create a board, it will be unmounted. This means that the board is either not attached to the Raspberry Pi or is not considered to be attached. While a board is unmounted, its definitions for extensions, inputs, and components may be freely edited.

To mount a board, make sure your board is physically attached to the Raspberry Pi. You will need to power down the Pi to do this. Once the board is attached, power up the Pi and wait for the webserver to start. Once Breezy4Pi has started then navigate back to the Boards tab and choose the board you wish to mount. Click ‘Mount Board and Save’ to mount the board and cause Breezy4Pi to configure the board’s inputs and components and map them for the Raspberry Pi. The Raspberry Pi will then be able to communicate with the board.

One feature about mounted boards is found in the ‘Components’ tab. You will notice that the Edit button has been removed and replaced by a ‘Test’ button for each component. Clicking the ‘Test’ button will cause the component to execute a short test program to demonstrate that the component has been properly mapped to the hardware.

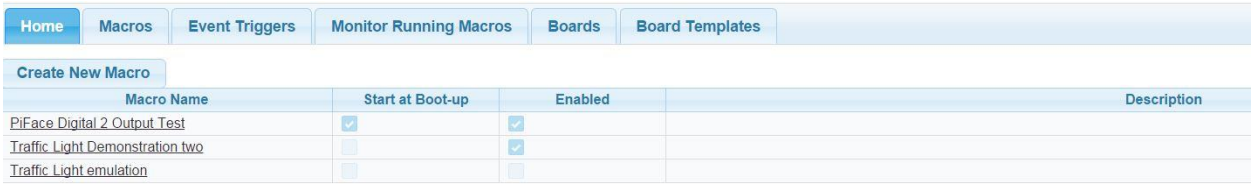
Unmounting a board will cause Breezy4Pi to unconfigure the associated input and output pins of the board. The board will then be invisible to the system except for editing purposes.

After mounting or unmounting boards, it is a good idea to restart the Tomcat webserver, or power cycle the Raspberry Pi to restart the server to clear out old data.

Defining Macros

Once the board or boards have been defined and mounted, then you will be able to create the macros used to execute components functions such as blinking lights, or turning on or off motors and to execute different program paths depending on input values. For example to keep looping in a macro until a certain input goes low.

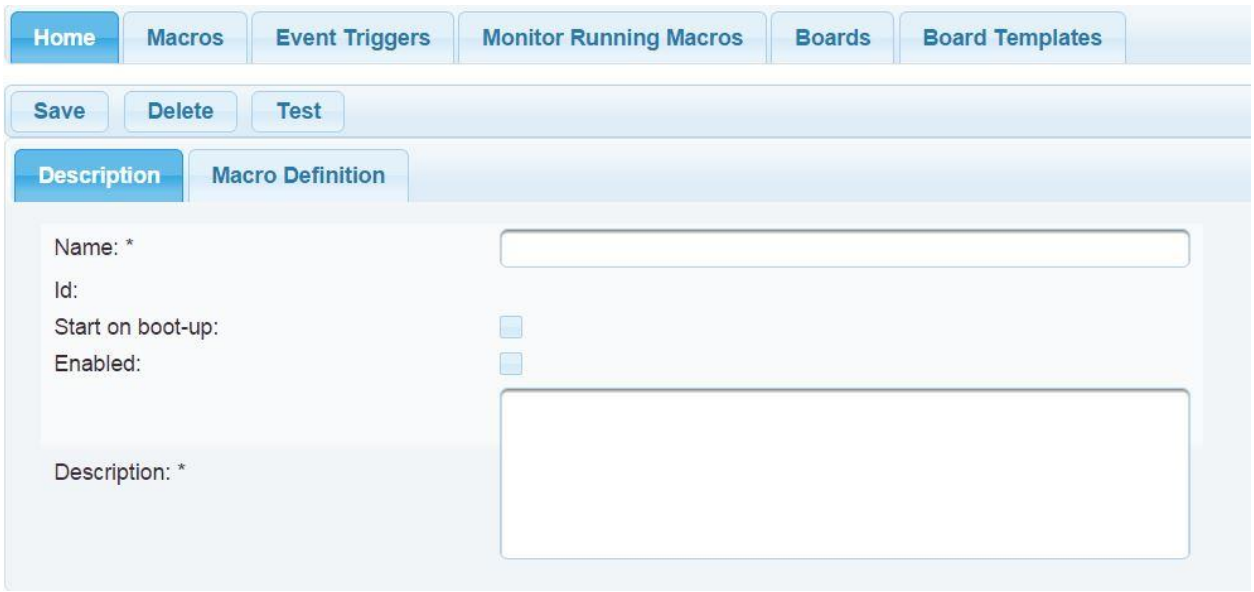
To create a macro, click on the 'Macros' tab then click on 'Create New Macro'. Clicking on an existing macro will allow you to edit it.



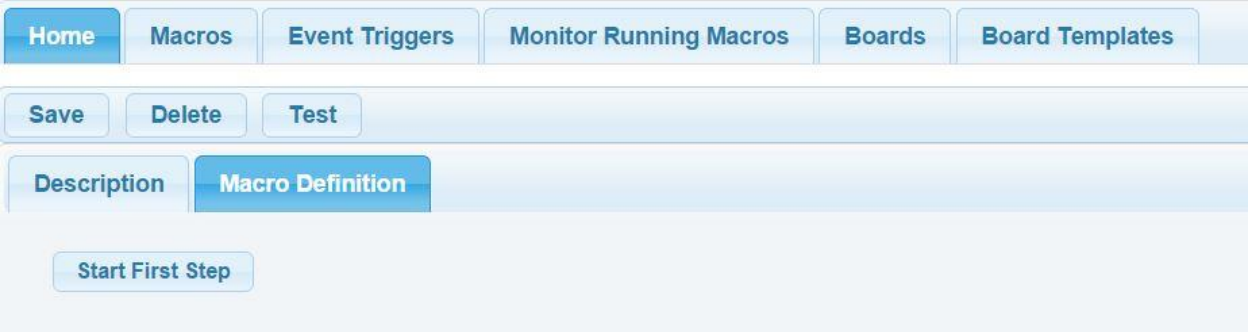
Enter the name you wish to give this macro. This is how the macro is defined in the list of macros and when you wish to associate an event trigger with this macro.

Checking 'Start on boot-up' tells Breezy4Pi that this macro should start upon restart of the Tomcat webserver.

Checking 'Enabled' allows the macro to be executed. This important for testing and event triggers.



Click on the 'Macro Definition' tab then the 'Start First Step' button to start creating your macro.



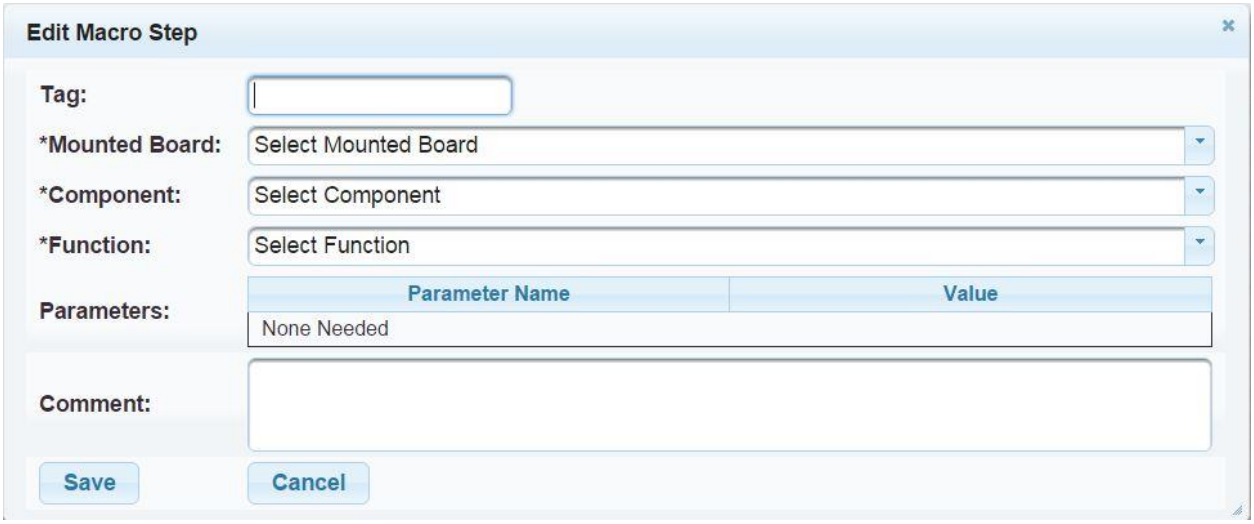
This will pop up the editor for this step.

Tag is an optional value and is used for looping and branching while the macro is executing. The value is case sensitive.

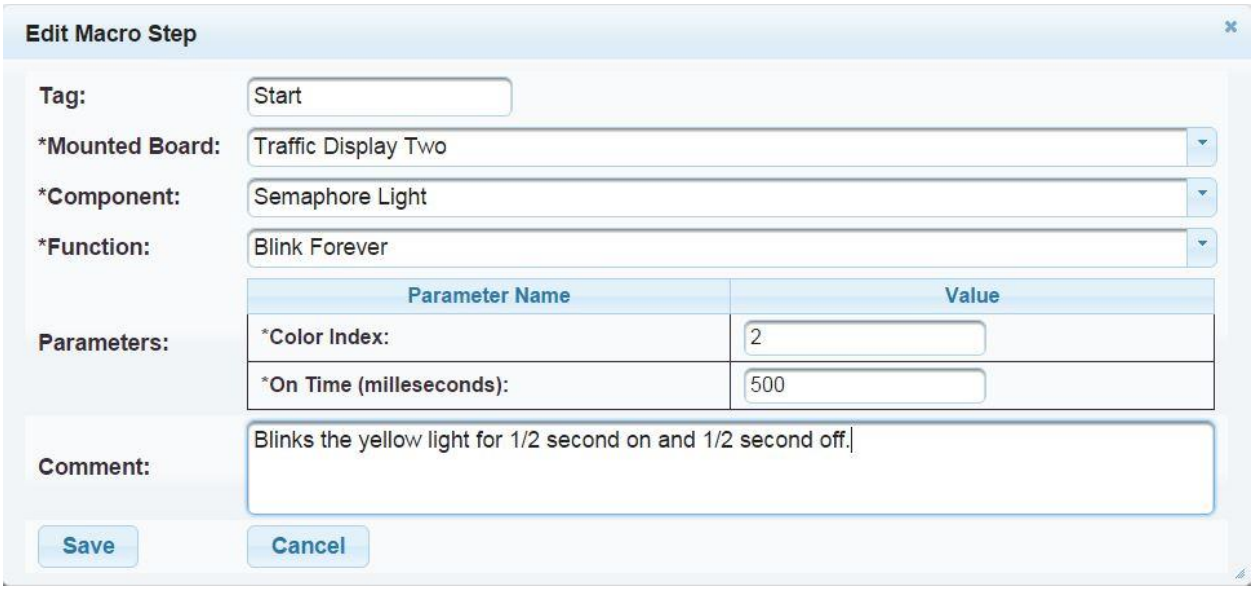
Select the mounted board which has the component whose function you wish to execute. Select one of the component names you defined earlier. Now select a function you wish to execute.

The 'Parameters' area is context sensitive to the chosen component. Each component function has its own required and optional values that it needs for execution.

Comments are optional but will help describe program flow.

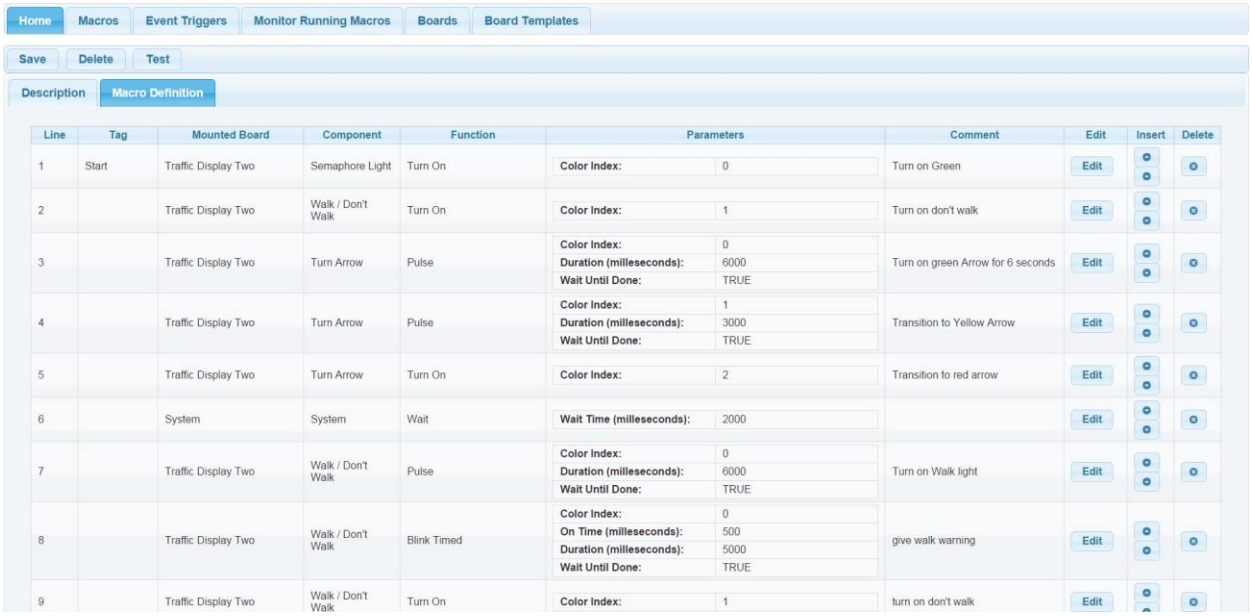


Here is an example of a completed macro step. 'Color Index' refers to which led of this Semaphore Light will blink, i.e. light 0, light 1, or light 2:



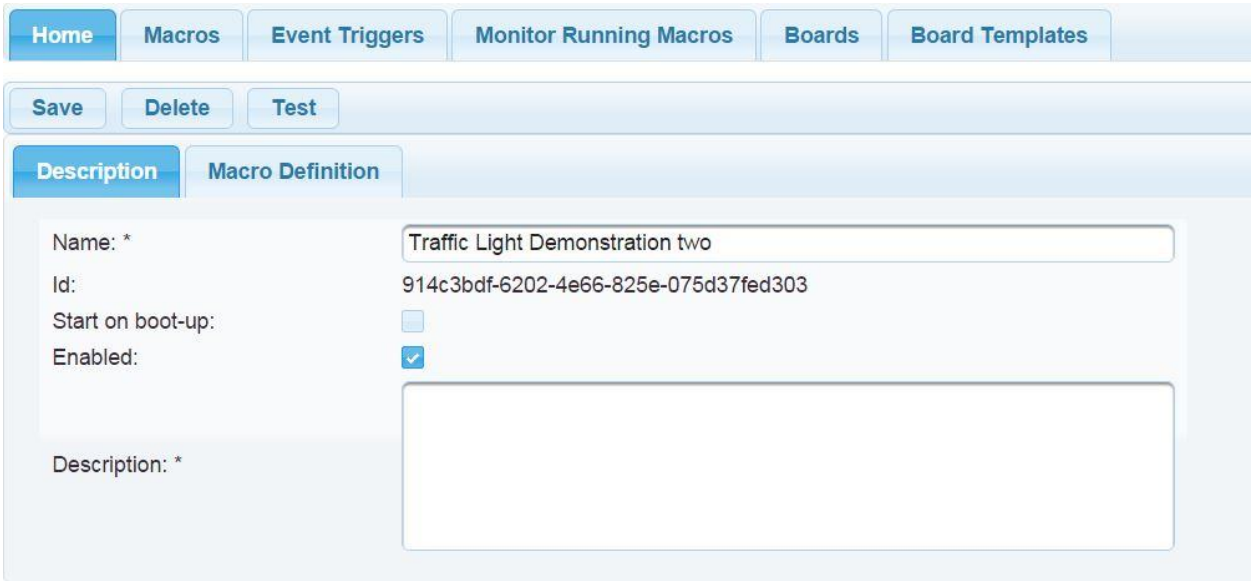
Click 'Save' to save edits and return to the macro lines. The 'Edit' button allows you to edit a line. The up and down arrows in the 'Insert' column allow you to insert new lines either before or after the current line and the 'X' in the 'Delete' column lets you delete a line.

Here is what a partial macro looks like:



Clicking the main 'Save' button will save the macro to disk. 'Delete' will delete the entire macro and return to the list of macros. 'Test' will execute the macro until it completes on its own or is manually terminated (See Monitor Running Macros).

There can only be one instance of a particular macro executing at any time. If there are multiple trigger events or presses of the 'Test' button and the associated macro is running then those event are ignored.



Defining Event Triggers (mapping inputs to macros)

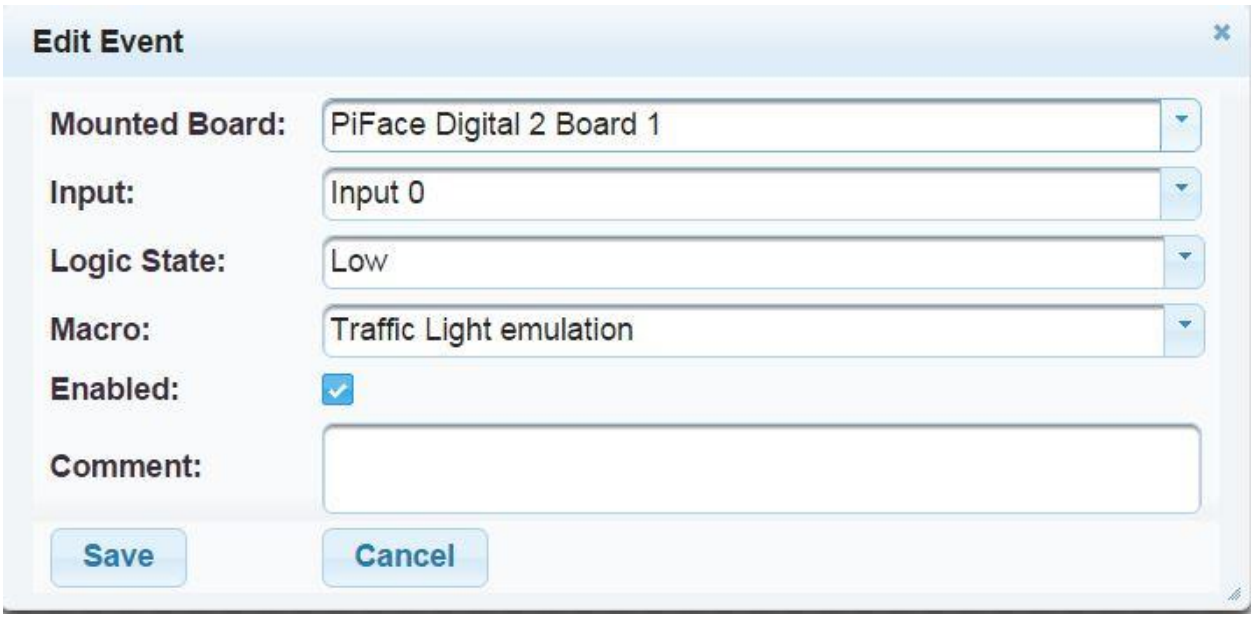
Whenever an input changes logic state, for example from high to low, that state change can be mapped to start the execution of macro.

The example below illustrates that when Input 0 of the PiFace Digital 2 Board 1 goes low, then the Traffic Light emulation macro will begin to execute.

To add a new event, click on the 'Create New Event' button and complete the fields in the Edit Event dialogue. Existing events may be edited by clicking the pencil icon in the 'Edit' Column or removed by clicking the 'X' button in the 'Delete' column.

Fill in the fields shown below for the Edit Event dialogue. Click 'Save' to save changes and close the dialogue. Click 'Cancel' to lose changes and close the dialogue.

The Input field is context sensitive and will contain different values depending on the mounted board. The 'Enabled' field enables of disables the event trigger. If disabled, then the event will not fire and the macro will not start running.



Monitor Running Macros

You can see which macros are actively running by navigating to the 'Monitor Running Macros' tab. This page is update once a second and so will not show macros that start and terminate quickly. You can terminate individual macros by clicking 'End Macro' button in the 'End Macro' column, or terminate all macros by clicking the 'Stop All Macros' button.

Definitions:

System – Refers to the base single board computer, in this case the Raspberry Pi. The single board computer is regarded within Breezy4Pi as an I/O extension.

GPIO – General Purpose Input/Output. Please see the Raspberry Pi's documentation for more information.

Input – A GPIO on a board, either System or Extended Board that has been mapped as a digital input within the Breezy4Pi application.

Component – An aggregate of one or more GPIOs that have been mapped as digital outputs within the Breezy4Pi application. Each component has a set of functions that can be executed to perform specific functions such as running a motor, or running a sequence pattern on a group of LEDs.

Extension or I/O Extension – Refers to the physical I/O device, providing a set of GPIOs. Examples include the Raspberry Pi itself and MCP23017, MCP23S17, etc…

Board Template – Is an abstract mapping within Breezy4Pi for the physical hardware (circuit board) that can be mounted or attached to the Raspberry Pi. The Board Template maps the I/O Extensions used on the circuit board, the inputs, and the components for those extensions. Board Templates are never attached directly to the Raspberry Pi. Instead boards, typically derived from board templates are what gets mounted as boards will typically have defined the parameter needed for the Raspberry Pi to communicate with the boards.

Board and Mounted Board – Is the concrete mapping of I/O Extensions, Inputs, and Components of a circuit board. Boards and Mounted Boards are typically derived from Board Template definitions (it is planned to have the ability to create ad hoc or prototype boards in the near future) with additional parameters defined so that they can communicate with the Raspberry Pi. The main difference between Boards and Mounted Boards is that Boards are not physically attached to the Raspberry Pi while Mounted Boards are.

Events and Event Triggers – Are those actions that can cause a Macro to run. Anything that causes a logic state change, i.e. from low to high or high to low can trigger an event and therefore start a Macro executing.

Macros – Are a series of programmed steps to manipulate output states (i.e. run relays, start motors, etc..) and execute component functions (i.e run led sequences, etc…).