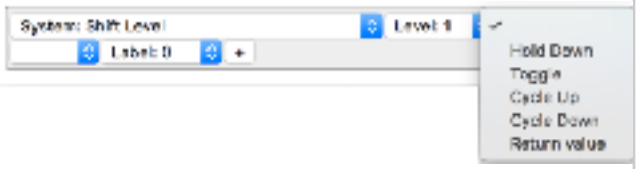
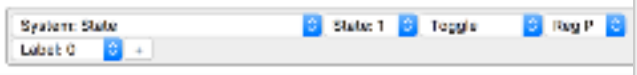
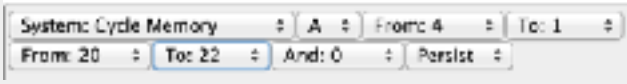
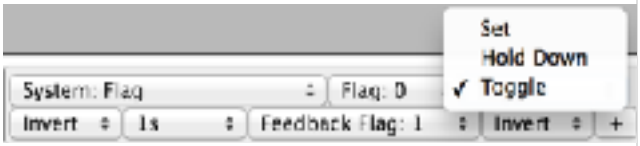
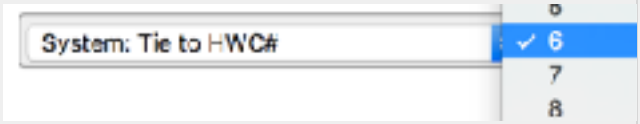
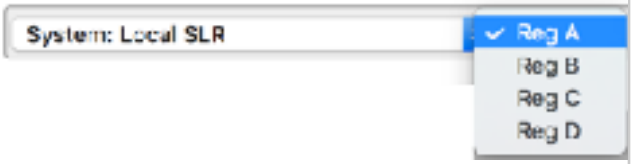
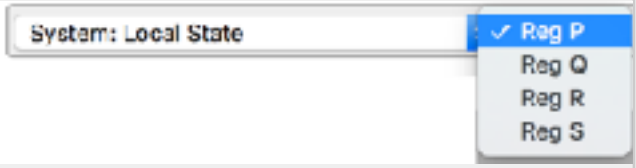
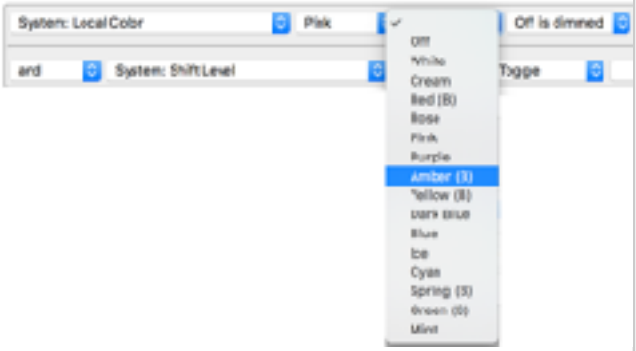
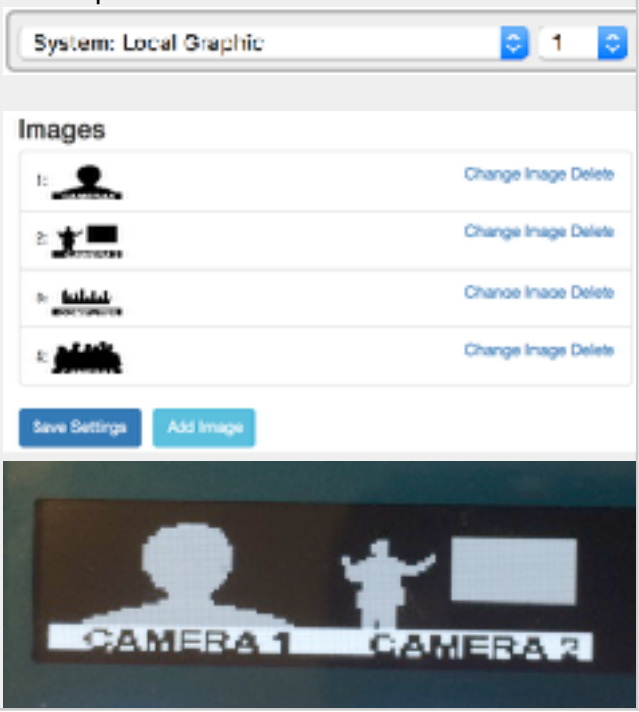
System Actions

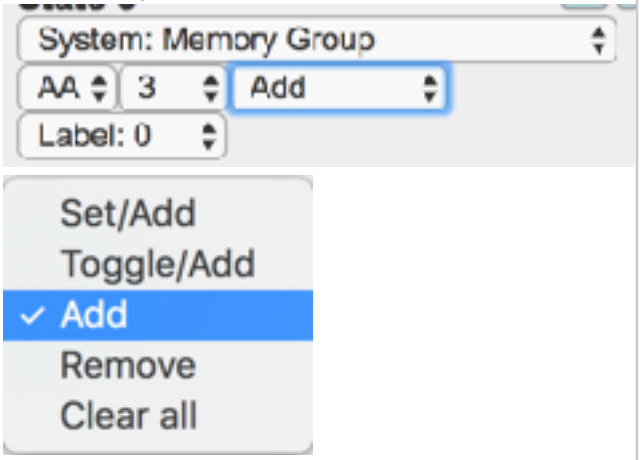
This is a table of system actions available for any UniSketch powered SKAARHOJ controller:

Shift Level 	Sets the controller shift level. *Binary triggers:* Sets the selected shift level for the given register (default or A-D). If Hold Down is selected, the shift level will fall back to Level 0 whenever the trigger is released. Toggle will set the shift level unless it's the same in which case it will set Level 0. If Cycle Up/Down modes are selected, a trigger will set the next/previous shift level up to the level selected. *Pulse inputs:* Will cycle through shift levels up to the selected level. *Analog inputs:* Will map the analog input range to shift levels up to the selected level. *Binary outputs:* On when the shift level matches selected source (or when trigger is held in Cycle modes) *Button colors:* Follows binary output: Highlighted, when on. Mode "Return value" means the selected level will be set if the binary value of the previous actions return value is true, otherwise the level will be set to zero. This is evaluated regardless of any triggers. This feature is a way to link a state of an external device to a shift level in the controller. Displays: Shows the selected shift level either as a number or "Off" (0) or "On" (1). If Cycle modes are selected, the display will reflect the current level (status). If a Label is selected other than "Label: 0" either a label string or an image (1-10) is used. Label strings will only overlay the default part (header, line1, line 2) if they are not empty.
State 	Sets the controller state. *Binary triggers:* Sets the selected state. If Hold Down is selected, the state will fall back State 0 whenever the trigger is released. Toggle will set the state unless its the same in which case it will set State 0. If Cycle Up/Down modes are selected, a trigger will set the next/previous state up to the state number selected. *Pulse inputs:* Will cycle through states up to the selected state number. *Analog inputs:* Will map the analog input range to state numbers up to the selected state. *Binary outputs:* On when the controller state matches the selected state (or when trigger is held in Cycle modes) *Button colors:* Follows binary output: Highlighted, when on. *Display text:* For displays and smart switches, the value will be shown as the preset label entered in the web interface if given, otherwise as a number. If a Label is selected other than "Label: 0" either a label string or an image (1-10) is used. Label strings will only overlay the default part (header, line1, line 2) if they are not empty.

Memory 	Sets memory registers A-L. If "Persist" is selected, the value will be stored in EEPROM and recalled from EEPROM upon booting the controller again (only register A-D). *Binary triggers:* Sets the selected value for the given register A-D. If Hold Down is selected, the value will fall back to the previous value whenever the trigger is released. Toggle will set the value, but on a subsequent trigger, it will fall back to the previous value. If Cycle Up/Down modes are selected, a trigger will set the next/previous value up to the value selected. *Pulse inputs:* Will cycle through values up to the selected value. *Analog inputs:* Will map the analog input range to values up to the selected value. *Binary outputs:* On when the memory value matches the selected value (or when trigger is held in Cycle modes) *Button colors:* Follows binary output: Highlighted, when on. *Display text:* For displays and smart switches, the value will be shown as a number. If a Label is selected other than "Label: 0" either a label string or an image (1-10) is used instead. Label strings will only overlay the default part (header, line1, line 2) if they are not empty.
Cycle Memory 	Sets memory registers A-D with values from specified ranges. If "Persist" is selected, the value will be stored in EEPROM and recalled from EEPROM upon booting the controller again. You can define two From-To ranges (values inclusive) and a single value ("And", must be different from zero). If both values in the range definitions are zero, the range is ignored. *Binary triggers:* Cycles to the next value given by the range line up. *Pulse inputs:* Will cycle through the values in the range line up. *Analog inputs:* Will map the analog input range to values in the range line up. *Binary outputs:* On when trigger is held in Cycle modes. *Button colors:* Follows binary output: Highlighted, when on. *Displays:* Shows the memory value.

Flag 	Sets a memory flag which is an internal binary value. This can be used to transport binary values around in the system. *Binary triggers:* Sets the selected flag. If Hold Down is selected, the flag will fall back to the previous value whenever the trigger is released. Toggle will set the flag, but on a subsequent trigger, it will fall back to the previous value. *Pulse inputs:* Will flip the value *Analog inputs:* Will clear/set the value when on either side of the middle of the analog value range. *Binary outputs:* On, if the selected feedback flag is set (but subject to modification by the second invert option). Notice that you must select the same feedback flag number as the flag-number if you want it to respond "intuitively". The idea of the feedback flag is to have a way to send a value out of the system but only reflect a confirmative return value. *Button colors:* Follows binary output: Highlighted, when on *Other features:* If "Invert" is selected, the flag will be cleared (binary zero) instead of set (binary 1) in any of the above cases. If any time frame is set, the value will fall back to the default after that period of time. *Display text:* For displays and smart switches, the status or function will be shown as On or Off (status us used for toggle switches). If a Label is selected other than "Label: 0" either a label string or an image (1-10) is used instead. Label strings will only overlay the default part (header, line1, line 2) if they are not empty.
Tie to HWC# 	This will tie an interface component to another interface component given by it's "Hwc#" number which is the number found in the web interface on every component. This feature is useful if you want a display to be linked to a button to display a label for it. But you can also copy functionality otherwise. If you tie a HWC to another HWC, you will execute its behavioural configuration, but not necessarily in the context it may be operating under in terms of shift, state and local colors set on section and module level. If you include other actions in a behaviour with this one, the will all be evaluated before you will evaluate the behaviour of the tied-to HWC. Has a transparent return value.
System Info	Displays system information, mainly status on connections.
No Action	No action. Will blank a display
Wait 1/10s second	Waits for a number of 1/10 second periods.
Custom Handler	Calls a custom handler (which need to be compiled into the firmware. (Special feature).
Inactivate	Inactivates the panel until pressed again. In inactive mode, no keypresses, turned knobs or pulled handles will result in any action. All displays and button colors will remain active though.
Stop Connect	If pressed, any device which is not yet connected will be disabled (until reboot of the controller).
Range Limiter	
Value Scaler	
Local Shift Register 	Defines which shift register is used in the context to evaluate shift level. The context can be a module/controller, a section and even a single behaviour. In case a local shift register is assigned on a regular HWC, it *has* to be the first action in the action list, otherwise the behavior will most likely be unintentional. This action does not depend on any trigger from the HWC, it will always be evaluated if inside the proper state and shift levels. Has a transparent return value.

Local State Register 	Defines which state register is used in the context to evaluate controller state. The context can be a module/controller or a section but it will not have any effect on regular HWCs. This action does not depend on any trigger from the HWC (obviously, since it's non functional on a regular HWC), it will always be evaluated if inside the proper state and shift levels. Has a transparent return value.
Local Color 	Sets the color for a button to something else than the default color schemes (default is yellow for bi-color, and white for RGB buttons). It affects SmartSwitches, Bi-color and RGB color buttons only. It also overrides special colors like red and green which are often returned for actions like setting sources on Program or Preview/Preset or recording. A local color action can be included anywhere among the actions executed for a HWC - it doesn't have to be the first action for instance. However, like any other action it is evaluated with respect to states and shift levels. Local color actions can be set also on section and controller level HWCs. The first parameter (in the example "Pink") will set the default color for both on and off (dimmed) state. If the second parameter (in the example "Amber") is set, this color will be used in the off-state of the button. The third parameter determines if the off-state of the new default color appears dimmed (default) or at full brightness. The color called "Default" will reset the color back to the default color schemes. All colors are designed to be distinctly different from each other on RGB buttons, but for bi-color buttons this is only guaranteed for those marked "(B)". For SmartSwitches the Local Color action will affect the smartswitch only if it's included as an action for the SmartSwitch HWC - not on section or module level. Furthermore, Default and Off settings may currently render unexpected results. The use or On or Off color is evaluated based on the binary return value of previous actions in the behavoir. This action does not depend on any trigger from the HWC, it will always be evaluated if inside the proper state and shift levels. Has a transparent return value.
Local Graphic 	Selects a graphic number to use for this button (in case it has a display). Graphics are uploaded as media on cores.skaarhoj.com for your controller. Files must be 64x32 pixels black and white. This action will always set the graphic no matter if a graphic has already been generated by a previous action in the behavoir. The consequence is that you can include Local Graphic before or after other actions and it will overrule their output (normally, only the first action will generate graphic content just like feedback color is generated by the first action that doesn't have a transparent return value). This action does not depend on any trigger from the HWC, it will always be evaluated if inside the proper state and shift levels. Has a transparent return value.

Local Label 	Selects a string number to use for this button as label (in case it has a display). Strings are entered as media on cores.skaarhoj.com for your controller. The strings for a controller can be used for many purposes (URLs, options, commands etc). If you use a string as a label, please format it according to "[Header] | [Line 1] | [Line 2]". You can omit header and line 2 if you want. Whitespace is respected, so you may want to exclude space from around the vertical lines. If two lines are shown, they can be up to 10 characters long (header too), but if a single line is shown, its 5 characters long. If "Is Status" is selected, the button label will be rendered with a solid title bar. This conceptually indicates that the label shows the current status of a value instead of merely what will happen if a button is pushed. This action will always set the graphic no matter if a graphic has already been generated by a previous action in the behavoir. The consequence is that you can include Local Graphic before or after other actions and it will overrule their output (normally, only the first action will generate graphic content just like feedback color is generated by the first action that doesn't have a transparent return value). However, notice the function of the "Clear" flag: If set, it will blank the display before setting labels, otherwise it will just set those labels (header, line1, line2) which are not empty. This action does not depend on any trigger from the HWC, it will always be evaluated if inside the proper state and shift levels. Has a transparent return value.
IP Request	
Memory Groups 	*Binary triggers:* Adds, Removes, Toggles the value to the memory group selected (AA or BB). If Set/Add mode is used, the value will be set by a quick push but added by a hold of 1 second. If Toogle/Add mode is used the value will be toggle as the only value in the memory group by a quick push clearing all others except if held down for 1 sec in which case it will not touch other values in the group. Add and Remove simply adds and removes without touching other values in the group. *Pulse inputs:* Will circle through the values from zero to the value given. *Binary outputs:* On when the selected value is found in the memory group (except for Remove which has it reversed and Clear all that is not active) *Button colors:* Follows binary output *Display text:* For displays and smart switches, the value will be shown as a number. If a Label is selected other than "Label: 0" either a label string or an image (1-10) is used instead. Label strings will only overlay the default part (header, line1, line 2) if they are not empty.