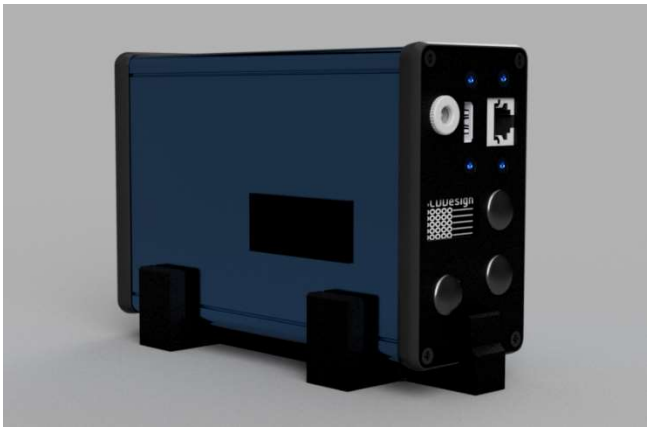
blueBox

User Guide



1 TABLE OF CONTENTS

2 SAFETY WARNING

The **blueBox** controller runs on low voltage (12VDC) but requires to be plugged in into standard 220VAC/110VAC mains power socket. Please follow safety guidelines when plugging the power supplies in the mains power source as it poses an electrical hazard that can cause harm or even death.

Risk of Electrical Shock! Inspect the power chords for damage. If any damage is noticed in any part of any of the power supplies and cables, replace them with an undamaged unit.

***Risk of Electrical Shock*!** Only plug the controller and the LED power supplies in mains power that was properly installed by qualified personnel.

Risk of Electrical Shock! Only plug the controller and the LED power supplies in mains power that is properly grounded.

Risk of Electrical Shock! The LED strips can be powered from multiple points. Please make sure to de-energize ALL power sources for the LED strips before servicing.

Risk of Electrical Shock! Only qualified personnel should perform service on the controller.

The **blueBox** controller and all its components are not designed to be installed in environments that require explosion-proof protection or in environments requiring greater than IP51 ingress protection rating.

****Product Laser Device Warning: Please Read Carefully****

The **mdb-measureKit variant** of the **blueBox** controller incorporates a Class I laser device. It is essential to understand and follow the safety guidelines outlined below to ensure your well-being and prevent any potential risks associated with laser technology.

1. Laser Safety Precautions:
   - Never look directly into the laser beam or expose your eyes or skin to the laser radiation.
   - Avoid any attempts to disassemble or modify the laser device. Only qualified personnel should perform maintenance or repairs.
   - Keep the laser device away from children and ensure they are supervised when in proximity to the product.

2. Eye Protection:
   - Although the laser in this device is classified as Class I, it is always recommended to exercise caution and avoid prolonged exposure to the laser beam, particularly for sensitive individuals, such as those with eye conditions.
   - If you experience any discomfort, eye irritation, or vision abnormalities while using the product, discontinue its use immediately and consult a healthcare professional.

3. Safe Usage:
   - Use the laser device strictly for its intended purpose as described in the product documentation.
   - Avoid pointing the laser beam towards reflective surfaces, including mirrors or shiny objects, as this can cause unintended reflections.
   - Do not use the laser device near flammable materials, volatile gases, or explosive substances.

4. Storage and Transportation:
   - When not in use, store the product in a safe place away from direct sunlight and extreme temperatures.
   - During transportation, protect the laser device from physical damage by using suitable packaging or a protective case.

5. Regulatory Compliance:
   - This product complies with relevant safety regulations and guidelines for Class I laser devices. However, failure to adhere to the instructions and warnings provided may result in potential hazards or compromise the safety features of the product.

   Please be aware that any misuse, negligence, or failure to follow these instructions could lead to serious injury or harm. We strongly advise you to familiarize yourself with the user manual, which provides comprehensive details on the safe handling and operation of the laser device.

   If you have any questions, concerns, or require further assistance regarding the safe usage of this product, please contact our customer support team immediately.

   By using this product, you acknowledge that you have read, understood, and agreed to comply with the above safety warnings and guidelines.

3 END USER LICENSE AGREEMENT

End User License Agreement for blueBox and mdb-lightKit

THIS END USER LICENSE AGREEMENT ("Agreement") is made between SCDDesign, LLC, a limited liability company organized under the laws of the state of Florida, United States ("Licensor"), and the end user ("End User" or "You") of the mdb-lightKit software ("Software") and the blueBox device ("Hardware").

1. License Grant. Licensor grants to End User a personal, non-exclusive, non-transferable license to use the Software and the Hardware in accordance with the terms of this Agreement. End User may only use the Software and Hardware for its intended purpose and may not reverse engineer, decompile or disassemble the Software.
2. Ownership. End User acknowledges that the Software and Hardware are the sole property of Licensor and that the End User has no rights in the Software or Hardware except as specifically set forth in this Agreement.
3. Warranty. Licensor warrants that for a period of 6 months from the date of purchase, the Hardware will be free from defects in materials and workmanship under normal use. If the Hardware fails to conform to this warranty, End User's sole and exclusive remedy shall be to return the Hardware to Licensor for repair or replacement. The warranty does not cover damage caused by End User's misuse, abuse, or unauthorized modification of the Hardware.
4. Disclaimer of Warranty. EXCEPT AS EXPRESSLY SET FORTH IN SECTION 3, THE SOFTWARE AND HARDWARE ARE PROVIDED "AS IS" AND LICENSOR DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT.
5. Limitation of Liability. IN NO EVENT SHALL LICENSOR BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION, DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, OR SPECIAL DAMAGES, WHETHER IN CONTRACT OR TORT, ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE OR HARDWARE.
6. Termination. This Agreement is effective until terminated. End User may terminate this Agreement at any time by destroying the Software and Hardware and all copies thereof. This Agreement will automatically terminate if End User breaches any term of this Agreement. Upon termination, End User shall destroy the Software and Hardware and all copies thereof.
7. General. This Agreement constitutes the entire agreement between the parties relating to the use of the Software and Hardware and supersedes all prior or contemporaneous understandings or agreements, whether written or oral, relating to such subject matter. This Agreement may not be amended except in writing signed by both parties. If any provision of this Agreement is held to be invalid or unenforceable, such provision shall be struck and the remaining provisions shall be enforced. This Agreement shall be binding upon and inure to the benefit of the parties and their respective successors and assigns.
8. Cost of Repair or Replacement. After the warranty has expired, End User will be responsible for the cost of repair or replacement of the Hardware.

End User agrees to be bound by the terms and conditions of this Agreement by using the Software and Hardware.

4 DESCRIPTION

The blueBox is a multi-purpose controller that can be configured in different variants or interconnectivity to multiple types of applications. Currently, following are the supported variants but can be expanded in the future:

- Web-cab Production Assistant
- Homag controllerMES
  - Kitting Rack
  - Production Flow
- Homag Automation automatic storage systems (storeTeq)
- Web-cab mdb-measureKit
- Web-cab mdb-feedbackKit
- Web-cab mdb-rfidKit
- SCDDesign mdb-lightKit

The function of the controller changes once the database type (variant) is defined in the parameters. The parameters and user functions are accessible via the web interface.

4.1 WEB-CAB PRODUCTION ASSISTANT

Each infeed scan of a part is visualized by the LEDs so that the operator(s) has a positive indicator to which infeed bin to insert the scanned part. When the bin is complete, the LEDs on the bins that are defined as outfeed bins will light with a different color to indicate that it is ready to assemble. The functions of the bins are not limited to kitting racks and can be extended to palletizing, workflow, or shipping by web-cab Production Assistant. Web-cab Production Assistant controls the LED activation via a database or file interface.

Infeed area

Outfeed Area

4.2 HOMAG CONTROLLERMES KITTING

Each feedback in the assembly area at the kitting rack will light the LEDs for the compartment to which the part needs to be inserted. Once the compartment has all the parts, the LEDs on the bin will indicate with a

different color to signal the operator that the parts in the compartment are ready to assemble. The functionalities of the bins are defined by the controllerMES user exits.

4.3 HOMAG CONTROLLERMES PRODUCTION FLOW

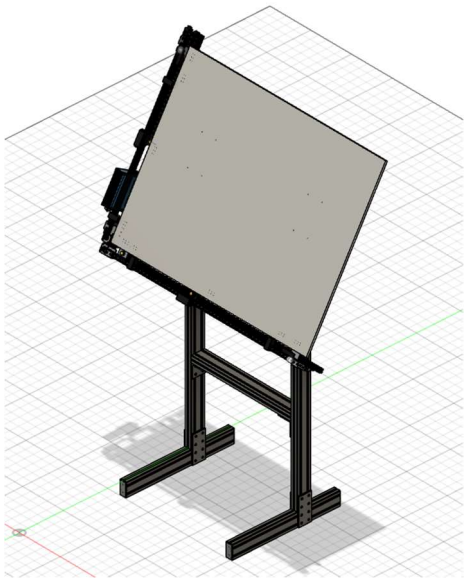
Each feedback or scan from the configured workstation will light the LEDS to indicate which transport system to put the part for the next production steps for the workpiece. The controllerMES user exit for the workstation will need to be modified to specify the sort criteria for the next production steps.

4.4 HOMAG AUTOMATION TLF (STORETEQ)

The controller will light the LEDs up to the board length that are at the infeed station to guide the material handler in the alignment. It also displays standard size markers and a custom size marker.

4.5 MDB-MEASUREKIT

The controller will gather range data from ranging sensors or stepper motors with proximity sensors to determine the length and width of a work piece. The operator puts the workpiece on the measuring table and the sensors will detect the edges to take dimensional information so that the data can be used for sorting, filtering, printing, or display.



mdb-measureKit with stand

4.6 MDB-FEEDBACKKIT

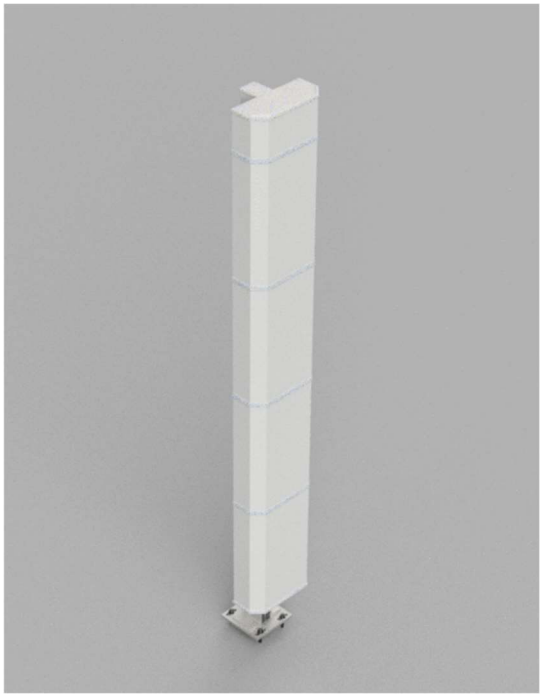
The controller will read the feedback from machines and create records of the feedback in the internal database table. The machine database or feedback file that contains the feedback for processed workpieces must be available.

Supported machines:

1. Homag CNC with legacy feedback table
    a. Venture
    b. BHX
    c. BHC
    d. BHP
    e. ABD
    f. centaTeq
2. Homag powerTouch control CNC
    a. Venture
    b. BHX
    c. BHC
    d. BHP
    e. ABD
    f. centaTeq
3. Homag Panel Dividing HPS Robot
4. Homag Panel Dividing saws
    a. HPP
    b. HKL
5. Homag edgeTeq edgebanders with feedback connections including KAL

4.7 MDB-RFIDKIT

The controller with attached RFID antennas can read RFID tags. The RFID tags are recorded in the database to be used by a superior system. When addressable LEDs are connected to the controller, the status is displayed by the LEDs at the front. An audible buzzer can also be activated to supplement the LEDs via an interface command.



5 WHAT IT COMES WITH

Web-cab Production Assistant, Homag controllerMES Kitting, mdb-lightKit

- 1x controller box with power supply
- 60 LED Strip assembly with LED channels, channel lens, and adapters
- 1x power supply (In 110-220VAC, Out 5V, 10Amps)
- 1x power supply (In 110-220VAC, Out 12V, 5Amps)
- 3x 3m 3-wire conductor cable

mdb-measureKit only

- 1x controller box with power supply (In 110-220VAC, Out 12V, 5Amps)
- 1x 2-axis Assembly
- 2x mounting spacers
- 1x Stand

mdb-feedbackKit only

- 1x controller box with power supply (In 110-220VAC, Out 12, 5Amps)

mdb-rfidKit only

- 1x controller box with power supply (In 110-220VAC, Out 12V, 5Amps)
- 1x Stand assembly with 2 long range RFID antennas

6 GENERAL NOTES

Shutdown the controller using the *Shutdown* link via the web interface**. Failure to do so can corrupt the data in the controller**. Avoid unplugging the controller without shutting down. Sometimes it is necessary to reboot the controller. The *Reboot* tool must be used from the web interface.

The Start/Restart App will restart the software.

When a parameter or bins configuration is changed, restart of the app is necessary. A reboot is not required.

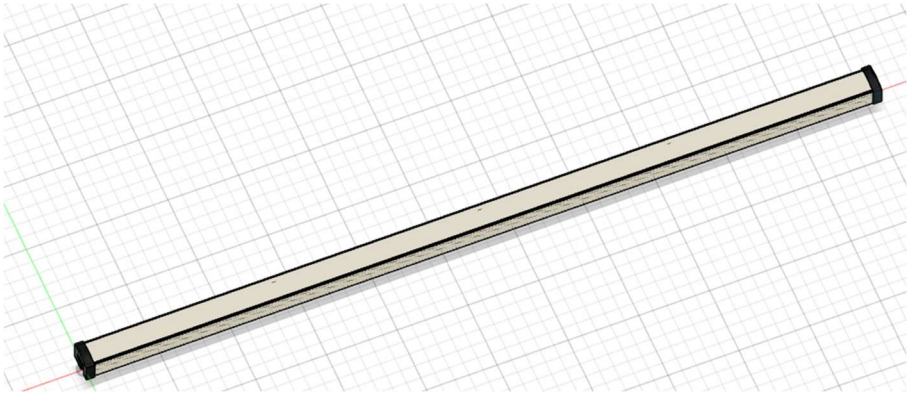
Before changing the parameters and the bins configuration, it is best to make a backup first using the *Make a Backup* tool from the web interface. When total app failure occurs, at least a working copy can be restored using the *Restore from Backup* tool.

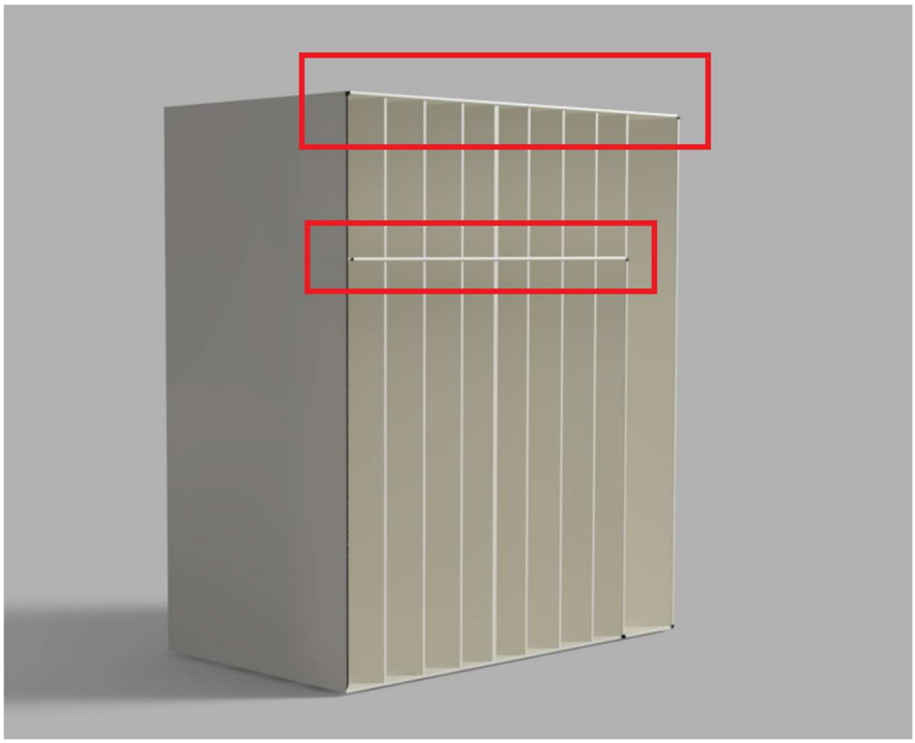
Distributors and partners for the blueBox may develop their own shutdown and reboot interfaces. Check the manual for specific applications that are provided by the distributors.

7 PREPARATION OF THE LED STRIPS

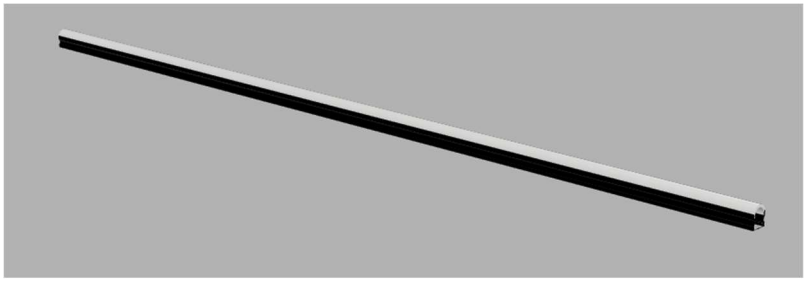
7.1 LED STRIP ASSEMBLY

Some variants of the blueBox controller comes with LED strip assemblies. The LED strip assemblies can be installed center of the bins and affixed with screw or with double-sided tape.

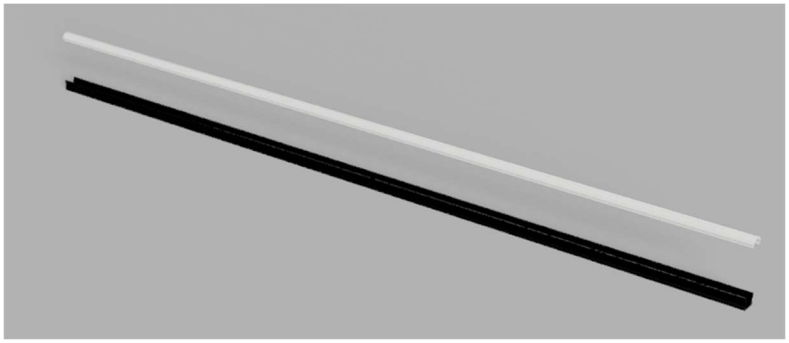




1. Prepare the aluminum U-Channels



a. Remove the diffusion lens from the U-channel.



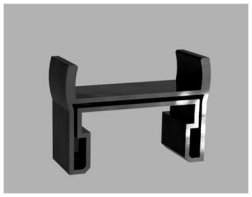
b. Affix the U-channel on the rack using screws. You must first drill a pilot hole on the aluminum channel using a drill bit appropriate for the size of the screws you are using. Leave approximately 5 mm of room on the left side of the first channel for the end plate and wire loom. Do not put the screws where the LED strips will be positioned (avoid the center of the bins). You may have to cut the U-channels to length for proper fit.

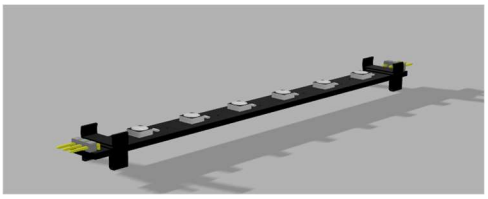
c. Install all required U-channels that cover the length of all the bins.

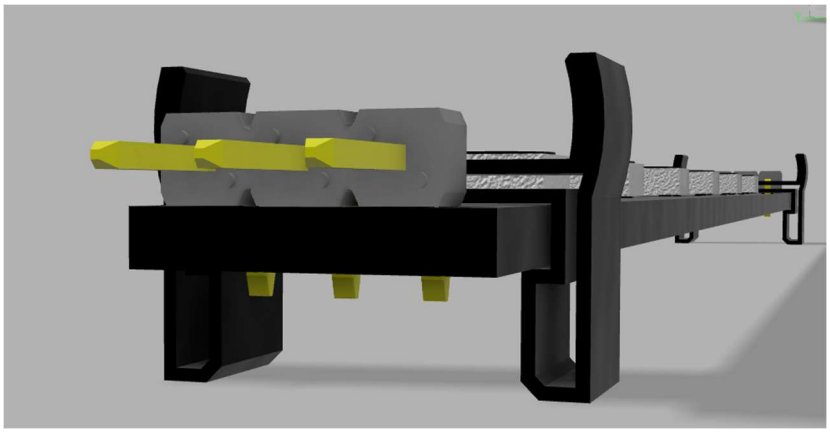
2. Prepare the first LED strip. The first LED strip should be the one with right angled male pins.

a. 

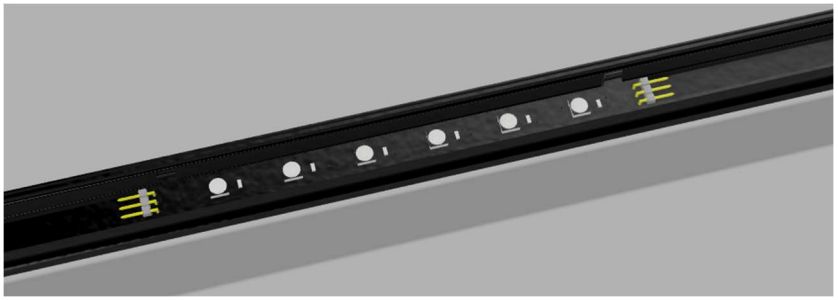
b. Install the LED strip clip/standoff on the left and on the right side of the LED strip.





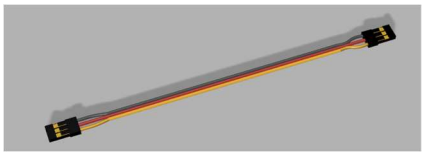


3. Find the center of the bin and insert the LED strip with the standoff/clip into the U-channel. There will be some friction – this is normal. Friction should be present so that the LED strip can stay in place.

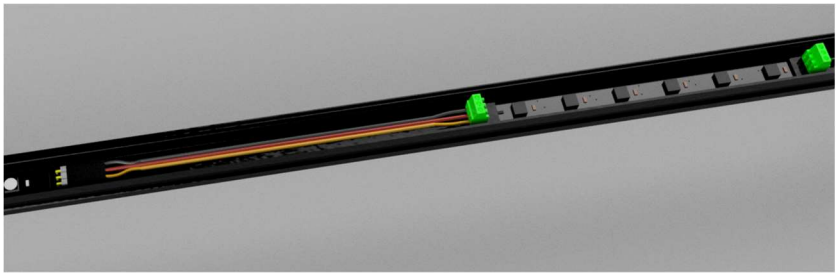


a. Install the LED strips with the standoff/clip on all the bins as required using the same methods above.

b. Now install the 3wire Dupont female-female connectors to connect all the LEDs. Make sure to orient the wires the same way so that the 5V, DI/DO, and GND are always using the same conductor color.



c. If the next LED strip has screw type terminals, cut the one female end of the Dupont connector, and terminate it with ferrules. Install the wires in the screw type terminal.



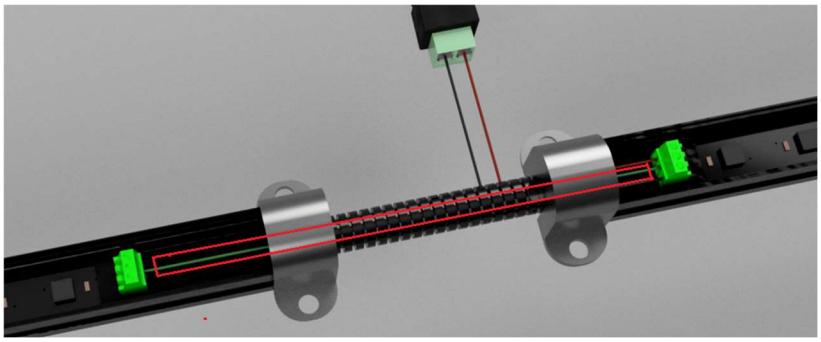
d. Use a 3wire conductor cable to connect to the next strip if the next strip is also a screw type terminal. Use the wire loom to protect the cable.

4. Test the functions. Once the functions of the LEDs are verified, Install the Lens/diffuser to the U-channels. Installation is now complete.

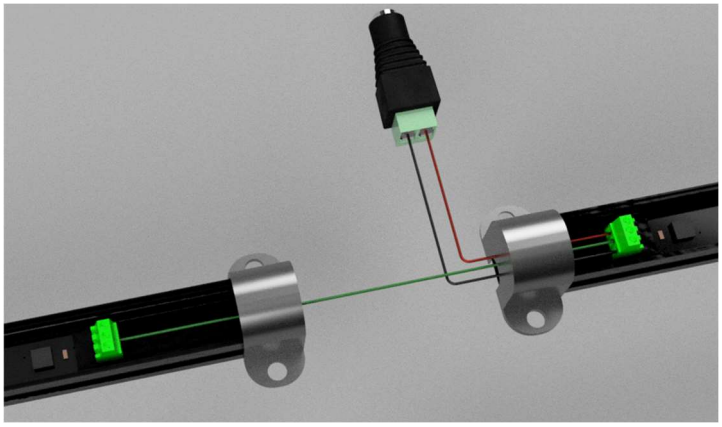
Adding Power Supply for the LED Strips

When there are more than 27 LED strips installed, additional LED power is required so that the color fidelity of the LEDs is maintained. Insufficient power can also make the LEDs behave erratically and cause damage to the LED diodes. For every 27 strips, additional power is normally required.

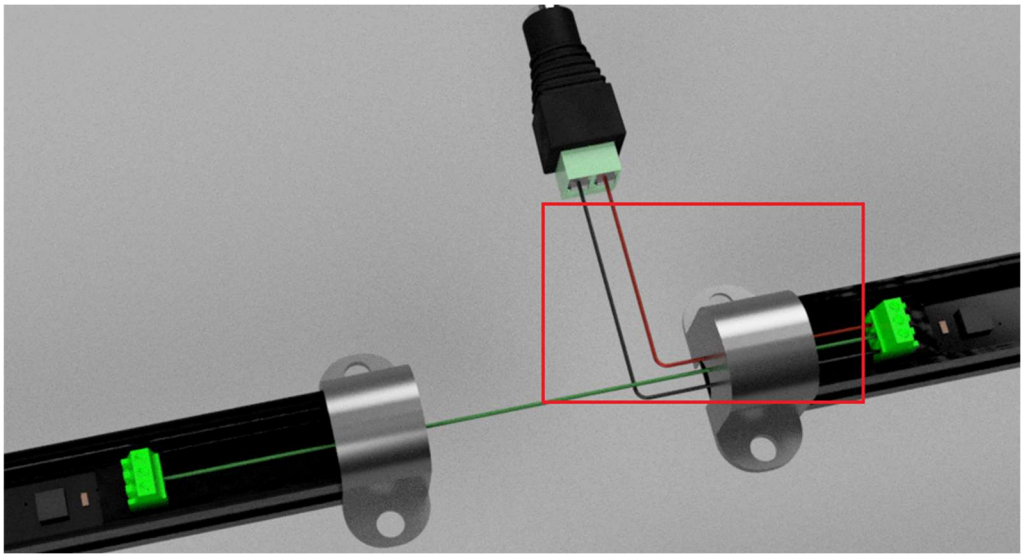
1. Connect **only** the DO signal from the last LED strip from the previous circuit to the first LED strip of the next circuit.



With the wire loom removed to visibility:



2. Install 2 wires to connect a power barrel connector to the terminal strip of the first LED strip of the next circuit. Ensure that the +5 of the LED strip is connected to the + of the barrel connector and the GND of the LED strip is connected to the – of the barrel connector.

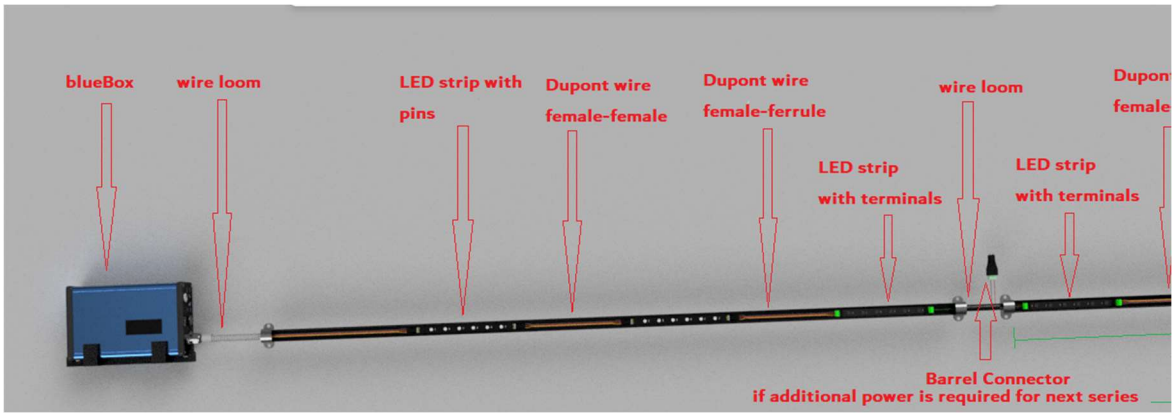


3. Install a wire loom to protect the wires.
4. Repeat this procedure every time additional power supply is required.

the LED strip assembly can come with screw terminals or male pins

Example Completed Configuration of LED Strips with Connection for an Additional Power Supply

This configuration is an example of how the LED strips should be connected.



When the LEDs are extended to the next series of strips or when additional power is required, LED strip with terminals or an appropriate 3 conductor wire Y-splitter is required as the last LED strip from the previous series and as the first LED strip of the next series.

Ex:

The last LED strip of the series for infeed bins should be an LED strip with terminals and should transition to another LED strip with terminals on the first LED strip for the outfeed bins.

OR

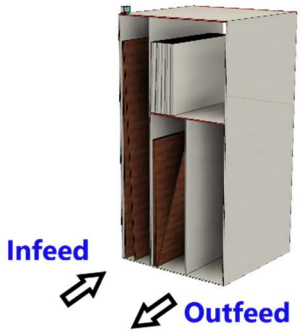
The last LED strip of the series and the first LED strip of the next series for should be connected to a 3 wire Y-splitter

8 INSTALLATION AND SETUP – DATABASE TYPE 1 (WEB-CAB PRODUCTION ASSISTANT), DATABASE TYPE 2 (HOMAG CONTROLLERMES KITTING), DATABASE TYPE 6/6.1 (HOMAG CONTROLLERMES PRODUCTION FLOW), AND DATABASE TYPE 9 (MDB-LIGHTKIT)

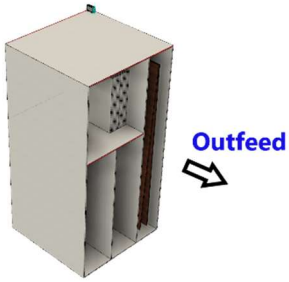
Installation of the LED strips must be customized to the configuration of the customer kitting rack, or staging area.

1. Install the LED strips to the total length of the bins (Production Assistant, cMES kitting)

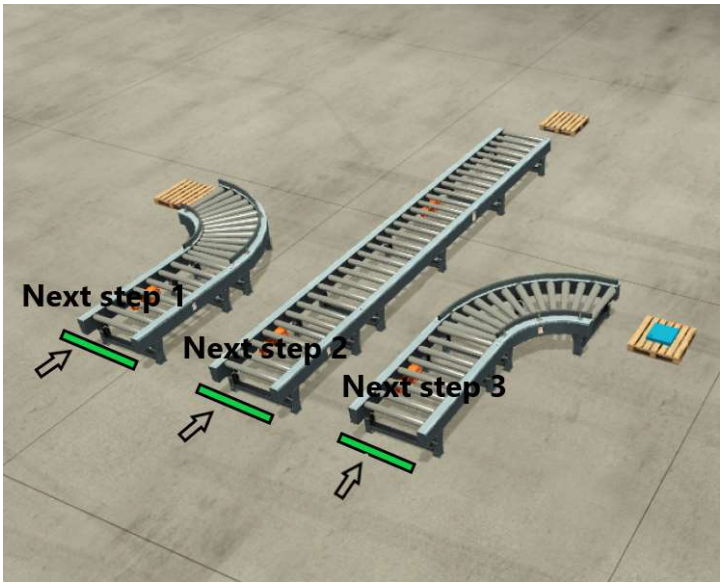
    a. If the infeed bins are the same as the outfeed bins (operator inserts and removes parts on the same side), only install LED strips on one side.



    b. If the infeed bins and outfeed bins are not on the same side (operators insert the parts on one side and remove parts that are ready to assemble on the opposite side), the LED strips must also be installed on the outfeed side.

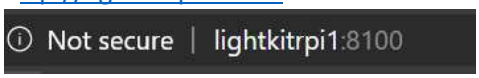


For Homag controllerMES Production flow, install it at your discretion on the next production step staging area.



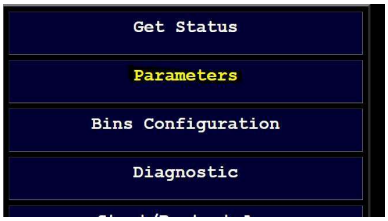
2. Connect the controller to the network via the LAN port
3. Apply power to the controller
    a. The controller will go through its boot up cycle and will perform a pixel check. Each pixel will illuminate (red) in sequence from pixel 0 to the last pixel.
    b. It is possible that the last pixel does not match what is installed if you have more than 300 pixels (2 LED strips) because by default, the controller is shipped configured for 750 pixels.
4. Connect to the controller to setup the parameters
    a. Open a web browser and on the address bar enter: http://ipaddress:8100

    b. Alternatively, you can reach the controller via its default hostname: http://lightkitrpi1:8100



    Note: if you have more than 1 controller in the network, you must use the IP address method

5. From the web interface, click on the *Parameters* link – this link is password protected. Please contact your distributor to commission the parameters.



a. Relevant parameter entries must be setup that pertains to Database Types. *The entries are case sensitive!*

i. Global Parameters – here the controller mode, SQL server and instance, and the TDS must be set

```
#-----------------------------------------------------#
#The running parameters used by the main mdb-lightKit
#The values must be customized per client.
#-----------------------------------------------------#
VERSION = "1.2020082701"

#In installation mode, the controller will continually display the las
#Set to false once the IP address for the controller is determined
#Every 10th pixel will also be illuminated starting at pixel 30
installation_mode = False

#####################
# Global Parameters  #
#####################
#set to true if the system will use database.  Set the database connec
#False if using a file interface.  Set the directory.
customer_name = 'Developer'
using_database = True
server = '172.16.1.36\HO2008'
database = 'lagerdb'
username = 'sa'
password = 'Homag'
port = '1433'
timeout = 5 #querry connection timeout

#MSSQL Database Connection TDS Version compatibility
"""
7.3.A = SQL Server 2008
7.3.B = SQL Server 2008 R2
7.4   = SQL Server 2012,
        SQL Server 2014,
        SQL Server 2016,
        SQL Server 2017,
        SQL Server 2019
"""
TDS_version = '7.4' #default is 7.4
use_tds = False #if the TDS version is required by SQL Server
```

a. Set the *installation_mode* to False to get out of installation mode if it is currently set to True
b. Set *production_mode* to True
c. Set the *customer_name*
d. The *using_database* entry should always be **True**
e. Set the *server* IP address and instance name
f. Set the user name and password
g. In many cases such as an older version of the SQL server, the TDS must be used and unset (**False**). Most SQL versions from MSSQL 2008R2 and higher requires this to be set to **True**. The default settings will work most of the time and requires no adjustment:

Default:

TDS_version = '**7.4'**

use_tds = **True**

On SQL EXPRESS versions, this must be set to **False**

h. The database type must be set to the appropriate setting

```
#Database types: 1   Production Coach   /  Web-Cab
#                                2   Homag controllerMES
#                                3   Homag Automation lagerdb
#                                4   Homag Sawteq
#                                5   3Tec
#                                6   Homag cMES production flow
#                                7   Homag woodFlex
#                                8   Reserved
#                                9   mdb-lightKit
database_type = 1

#bins import data
bins_data = "/opt/mdb-lightKit/data/rack.csv"

#numbers of LEDs in the strip (total)
number_of_pixels = 300
```

i. Set the total number of pixels

ii. Database Type 1, 6 and 9 Specific Parameters (web-cab Production Assistant, cMES, and mdb-lightKit) – specific parameters for the Database type must be customized

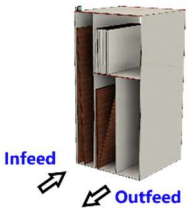
```
#########################################################################
#Database Type 1 and 9 Specific Parameters (web-cab and mdb-lightKit) #
#########################################################################
#infeed bins start and end (in bins)
infeed_bins_start = 1
infeed_bins_end = 60

#outfeed bins start and end (in bins)
outfeed_bins_start = 1
outfeed_bins_end = 60

#number of blinks infeed bin status
number_of_blinks_infeed_bin = 5

#number of blinks outfeed bin status
number_of_blinks_outfeed_bin = 5

#infeed and outfeed on the same side
infeed_outfeed_same_side = False

#width of pixels for infeed marquee
#the wider, the faster
#standard is (total_pixels/10)
width_of_infeed_marquee = 30
number_of_infeed_marquee = 3

#enable multi-color outfeed --only used when infeed_outfeed_same_side = True
enable_multi_color_outfeed = False

#Color definitions user1
color_infeed = (0,0,200) #infeed color
color_infeed_marquee = (0,0,50) #the running pointer color
color_outfeed = (0,200,0) #outfeed color
color_outfeed2 = (200,200,10) #outfeed color 2

#Color definitions user2
color_infeed_2 = (200,150,0) #infeed color
color_infeed_marquee_2 = (75,50,0) #the running pointer color
color_outfeed_2 = (150,0,150) #outfeed color
color_outfeed2_2 = (80,50,150) #outfeed color 2
```

a. Infeed bins start and end, Outfeed bins start and end
   i. Set the start infeed number and end
   ii. Set the outfeed bins start and end

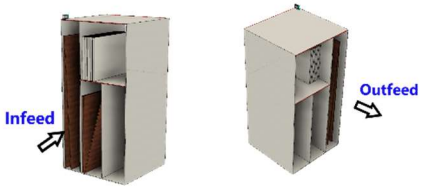
   It is possible that the infeed/outfeed start/end do not start at 1 or end at the last bin number if the customer has more than 1 controller. One controller can be monitoring half of the total bins and another controller for the remainder bins. This can be the case if the rack is very big and requires multiple controllers. It is also possible to have a setup where 1 controller controls the infeed area and another controller for the outfeed area.

Nonetheless, this is a special case that the controller can accommodate.

b. *number_of_blinks_infeed_bin, number_of_blinks_outfeed_bin*
    i. when a signal for the bin is received, the LEDs for the commanded bin will flash rapidly by this amount
c. *infeed_outfeed_same_side*
    i. Set to True if the infeed bins are the same bins as the outfeed bins (in the case of kitting racks)
        a. True



        b. False



d. *width_of_infeed_marquee*
    i. When an infeed command is received for the bin, the infeed side pixels will perform a running marquee towards the commanded bin. This parameter will set the number of pixels that runs. The more pixels is set, the faster the marquee will run. A typical setting is the total number of pixels divided by 10
        1. Ex.
            a. Total number of pixels = 750
            b. Set *width_of_infeed_marquee = 75*
e. *number_of_infeed_marquee*
    i. The number of times to repeat the running marquee
f. *enable_multi_color_outfeed*
    i. When this is set to **True,** the last outfeed bin commanded to be on will be a different color from the other outfeed bins that are already actively set to ON. This parameter is only active when the parameter *infeed_outfeed_same_side* is set to **True.** The default is set to **False**.

1. The main purpose of this parameter is for the operator to differentiate which was the last outfeed bin that became active when the infeed bins are on also the outfeed bins.

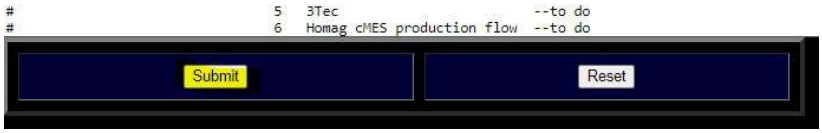
g. *indicate_outfeed_to_infeed*

i. When this parameter is set to **True**, the middle pixel of the infeed bin will indicate the status of the outfeed bin. This assumes that the outfeed bin has the same bin number as its corresponding infeed bin

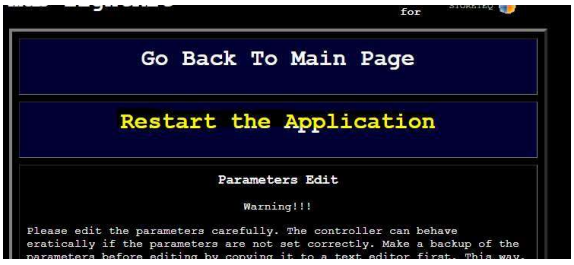
h. #Color definitions users

i. Different color schemes can be set for users so that each user can differentiate between each other

6. Once all the parameters have been set, save it by hitting submit.



7. Restart the Application



8. Now setup the bins by clicking on the Bins Configuration link from the web interface.

The bins can also be setup alternatively using the mdb-lightKit VI tool for Windows. You can download this tool via the web interface.

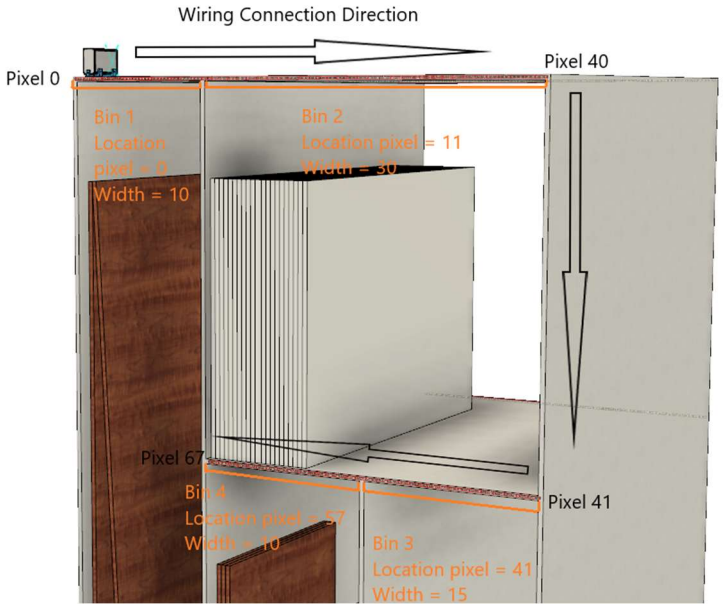
a. Enter the bins information in comma separated form
Bin Number, Location Pixel, Width in Pixels

```
Bin_Number,Location_Pixel,Width_Pixels
1,2,2
2,4,2
3,6,2
4,8,2
5,10,2
6,12,2
7,14,2
8,16,2
9,18,2
10,20,2
11,22,2
12,24,2
13,26,2
14,28,2
15,30,2
16,32,2
17,34,2
18,36,2
```

Bin_Number – the bin number. Each bin has its own number in sequence
Location_Pixel – the starting pixel number for the bin

Width_Pixels – the width of the bin in pixels

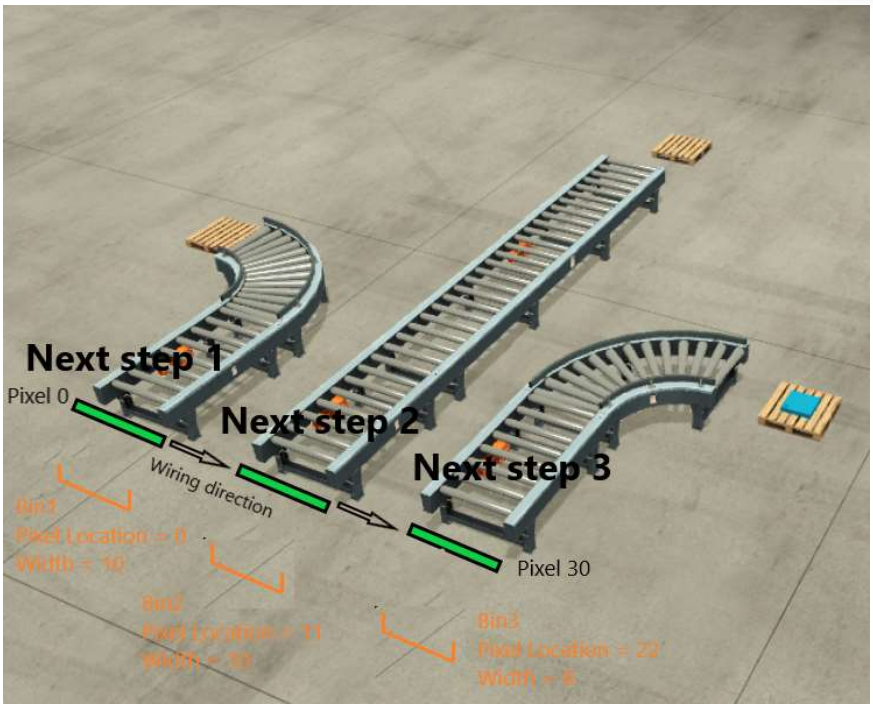
Example:



LED strips that have fixed number of LED pixels

follow the same setup procedure

In the case Homag controllerMES Production Flow (DB Type 6), Bin numbers are used to illuminate for the next production step staging area and therefore, the pixel configurations must also be set like above.



You can activate the installation aid to enter all the bins information. In the

Install Guides section of the web interface, there are some useful tools.

Markers On - activates every 10th pixel to aid with pixel counting

Pixels Check – activates a red running light to check individual pixels and to check the connections between LED strips

Cycle Infeed/Outfeed Bins – All the infeed/outfeed bins will cycle (2seconds ON, 2 seconds OFF) so the bins parameters can be verified visually

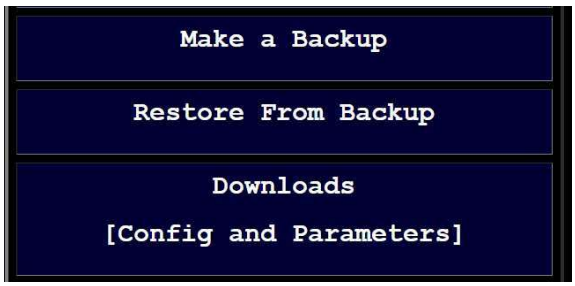
9. Save the parameter and Restart the application once again.

Once all the parameters and the bins configuration is set, the controller is ready for production operations.

Important!

Once a working system is in place, place make sure to create a backup using the web interface.

Download the config and the parameters and save it to a different media for safe keeping.


Make a Backup
Restore From Backup
Downloads
[Config and Parameters]

9 INSTALLATION AND SETUP – DATABASE TYPE 3 (HOMAG AUTOMATION TLF, STORETEQ)

1. Connect the controller to the workcell network via the LAN port
    a. If the workcell Cisco router is not directly accessible physically, the TLF storeTeq ethernet port 3 can be used if it is not in use
        i. Set the TLF storeTeq ethernet 3 to static IP address

            IP = 192.2.50.101

            Mask = 255.255.0.0

    b. If the ethernet 3 and the workcell Cisco router are not available, the controller can be connected to the customer network that can reach the TLF/storeTeq workcell. Additional parameters must be set for this to work properly – discussed further in the parameter settings section *hma_workcell_subnet* and *hma_router_ip*.
2. Apply power to the controller
    a. The controller will go through its boot up cycle and will perform a pixel check. Each pixel will illuminate red in sequence from pixel 0 to the last pixel.
        i. It is possible that the last pixel does not match what is installed if you have more than 300 pixels (2 LED strips) because by default, the controller is shipped set for 300 pixels.
        ii. It is possible that the controller will be in installation mode
            1. Every 10th pixel will be illuminated to aid with the setup of bins
            2. The OLED display on the controller box will display the IP address and Serial number of the controller.
                a. The IP address of the controller is

                    IP = 192.2.50.100 mask 255.255.0.0

                    Default Gateway = 192.2.2.180 DNS = 172.30.30.100

                b. If the controller is connected to the customer network, it will grab an IP address issued to it by the customer DHCP. Check the OLED display for the IP address or convert the first 30 pixels to the last octet of the IP address when the controller is installation mode.

            Take note of the IP address if the controller is connected to the customer network instead of being connected in the machine workcell network.

3. Connect to the controller to setup the parameters
    a. Open the web browser and on the address bar enter: http://192.2.50.100:8100

    b. Alternatively, you can reach the controller via its default hostname: http://lightkitrpi1:8100

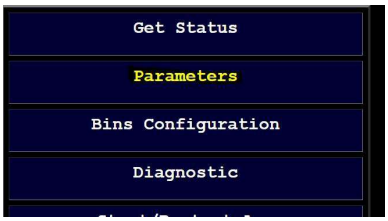
    

Note: if you have more than 1 controller in the network, you must use the IP address method

c. If the controller is connected via the customer network and using DHCP, you must used the issued IP address

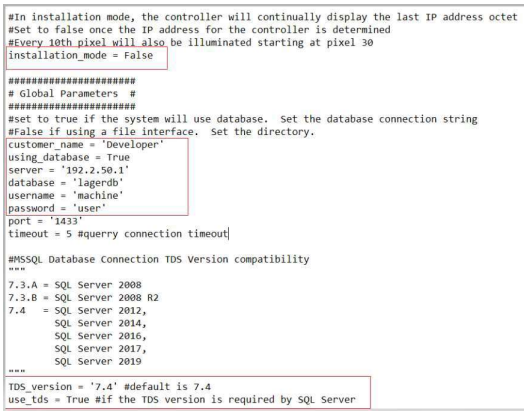
http://[ip address]:8100

4. From the web interface, click on the *Parameters* link – this link is password protected. Please contact your distributor to commission the parameters.



a. Relevant parameter entries must be setup that pertains to Database Type 3. *The entries are case sensitive!*

i. Global Parameters – here the controller mode, SQL server and instance, and the TDS must be set



```
#In installation mode, the controller will continually display the last IP address octet
#Set to false once the IP address for the controller is determined
#Every 10th pixel will also be illuminated starting at pixel 30
installation_mode = False

######################
# Global Parameters  #
######################
#set to true if the system will use database.  Set the database connection string
#False if using a file interface.  Set the directory.
customer_name = 'Developer'
using_database = True
server = '192.2.50.1'
database = 'lagerdb'
username = 'machine'
password = 'user'
port = '1433'
timeout = 5 #querry connection timeout

#MSSQL Database Connection TDS Version compatibility
"""
7.3.A = SQL Server 2008
7.3.B = SQL Server 2008 R2
7.4   = SQL Server 2012,
        SQL Server 2014,
        SQL Server 2016,
        SQL Server 2017,
        SQL Server 2019
"""
TDS_version = '7.4' #default is 7.4
use_tds = True #if the TDS version is required by SQL Server
```

a. Set the installation mode to False to get out of installation mode
b. Set *production_mode* to True
c. Set the customer name.
d. The *using_database* entry should always be **True**
e. Set the TLF/storeTeq db server IP address and instance name
    i. Default is 192.2.50.1
f. Set the user name and password
    i. Default is username = machine, password = user
g. In many cases such as an older version of the SQL server, the TDS must be used and unset (**False**). Most SQL versions from MSSQL 2008R2 and higher requires this to be set to **True**. The default settings will work most of the time and requires no adjustment:

Default:

TDS_version = '**7.4'**

use_tds = **True**

On SQL EXPRESS versions, this must be set to False

h. The database type must be set to **3**
i. Set the total number of pixels

```
#Database types: 1   Production Coach   /  Web-Cab
#                                  2   Homag controllerMES
#                                  3   Homag Automation lagerdb
#                                  4   Homag Sawteq
#                                  5   3Tec
#                                  6   Homag cMES production flow
#                                  7   Homag woodFlex
#                                  8   Reserved
#                                  9   mdb-lightKit
database_type = 3

#bins import data
bins_data = "/opt/mdb-lightKit/data/rack.csv"

#numbers of LEDs in the strip (total)
number_of_pixels = 300
```

ii. Database Type 3 Specific Parameters (Homag Automation TLF/storeTeq) – specific parameters for the Database type 3 must be customized

```
################################################
#Database Type 3 Specific Parameters (HMA)      #
################################################
hma_system_is_milliInch = True #true if the system is in inches
hma_number_of_infeed_places = 2  #number of infeed places
hma_overshoot_pixels = 0 #number of pixels to add at each end of the calculated board length
hma_pixel_pitch = 33.25  #distance between pixels
hma_center_pixels = (75, 225, 0, 0, 0) #center pixels of each infeed stations
hma_center_pixels_colors = ((255,0,0),(255,0,0),(255,0,0),(255,0,0),(255,0,0))  #colors of the center pixel marker
hma_infeed_place_color = (0,0,255) #color of the center marker
hma_enable_edge_markers = True  #enable the edge markers
hma_marker_8_colors = ((255,0,0),(255,0,0),(255,0,0),(255,0,0),(255,0,0),(255,0,0))  #colors for the standard 8 foot markers
hma_marker_10_colors = ((0,255,0),(0,255,0),(0,255,0),(0,255,0),(0,255,0),(0,255,0))  #colors for the standard 10 foot markers
hma_marker_12_colors = ((255,255,128),(255,255,128),(255,255,128),(255,255,128),(255,255,128),(255,255,128))  #colors for the standard 12 foot markers
hma_marker_custom_colors = ((0,100,100),(0,100,100),(0,100,100),(0,100,100),(0,100,100),(0,100,100))  #colors for custom markers
hma_marker_custom_size = 2768  #custom size to mark (in millimeters)
hma_infeed_place_color_marquee = (0,0,100)  #the color of running lights when changing sizes
hma_infeed_place_marquee_delay = 0.005 #speed of running marquee
hma_sleep_time = 2  #sleep time in minutes -turn off lights after db connection loss

#workcell network
hma_workcell_subnet = "10.101.0.0/16" #the workcell subnet
hma_router_ip = "172.16.0.1" #customer IP assigned to
```

a. *hma_system_is_milliInch* – set to True if the machine is in milliInch
b. *hma_number_of_infeed_places* – set the number of infeed places
c. *hma_overshoot_pixels* – when the board length is calculated, it is possible to add extra pixels to pad the calculated values. The machine can typically have up to 50mm of tolerance
d. *hma_pixel_pitch* – the pixels pitch. This is the distance between individual pixels. Accurate pitch is necessary for the proper calculation of the board length in pixels. The default:

33.25mm

It is possible that the LED strip provide can have a different pitch. Measure the pitch of the pixels with a caliper or get the pixel pitch data from the LED strip datasheet.

e. *hma_center_pixels* – the pixel number of the center pixels for each infeed stations. This parameter is a list. The first element is the center pixel for infeed station 1, the second is for infeed station 2 and etc. up to 5 infeed stations.
f. *hma_center_pixels_colors* – each infeed station can have different pixel colors. This parameter is a list of an RGB color list. The first element is for infeed station one and the last is for infeed station 5.
g. *hma_infeed_place_color* – the color of the individual infeed places, also in list form.
h. *hma_enable_edge_markers* – this parameter enables the standard size markers and the custom size markers for the infeed places (8ft, 10, ft, 12ft and custom size)
i. *hma_marker_8_colors* – colors for the standard 8 foot markers
j. *hma_marker_10_colors* – colors for the standard 10 foot markers
k. *hma_marker_12_colors* – colors for the standard 12 foot markers
l. *hma_marker_custom_colors* – colors for custom markers
m. *hma_marker_custom_size* – custom size to mark (in millimeters) that is not a standard size likt 8,10, or 12 ft boards
n. *hma_infeed_place_color_marquee* – the color of running lights when changing sizes
o. *hma_infeed_place_marquee_delay* – the speed of running marquee. This controls how fast the visual run lights when the board length is calculated and displayed.
p. *hma_sleep_time* – sleep time in minutes. This turn off lights after the database connection is lost. Effectively, if the machine is turned off, the LEDs will turn off after this parameter is reached.
q. *hma_workcell_subnet* – when the controller is not directly connected to the workcell network via the cisco router or the direct ethernet connection to the TLF/storeTeq computer, this must be set. The default is VLAN 1

10.101.0.0/16

r. *hma_router_ip* – when the controller is not directly connected to the workcell network via the cisco router or the direct ethernet connection to the TLF/storeTeq computer, this must be set to the IP address of the Cisco router. Otherwise, it will not be possible for the blueBox controller to connect to the database of the machine unless the machine database resides in the customer SQL server

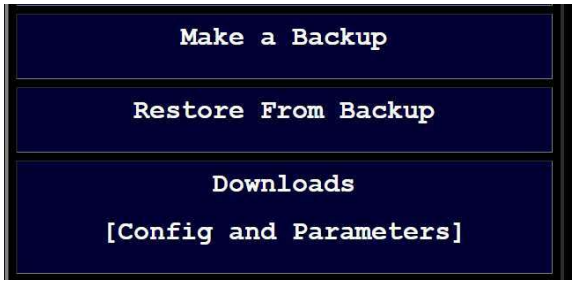
5. Save the parameter and Restart the application.

Once the parameters are set, the controller is ready for production operations.

Important!

Once a working system is in place, place make sure to create a back up using the web interface.

Download the config and the parameters and save it to a different media for safe keeping.

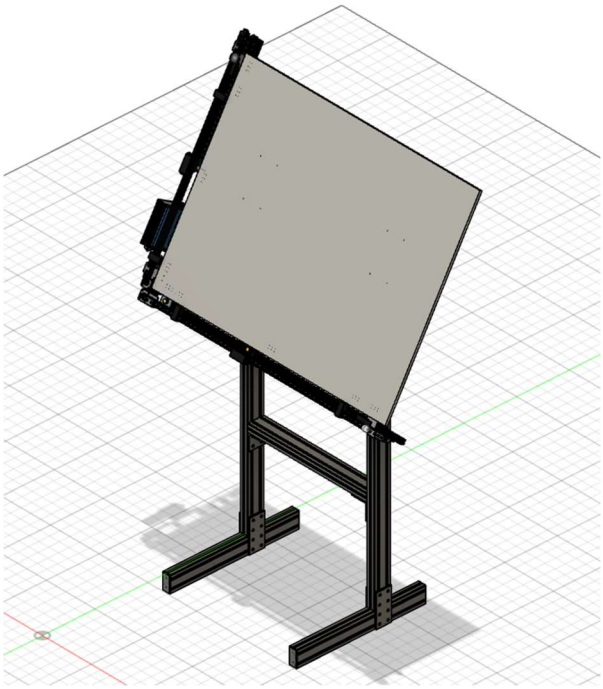


10 INSTALLATION AND SETUP – DATABASE TYPE 5 (MDB-MEASUREKIT)

If the controller is set to be an mdb-measureKit and sorting by LED is required, then first follow the instructions in section “INSTALLATION AND SETUP – DATABASE TYPE 1 (WEB-CAB PRODUCTION ASSISTANT), DATABASE TYPE 2 (HOMAG CONTROLLERMES KITTING), AND DATABASE TYPE 6 (HOMAG CONTROLLERMES PRODUCTION FLOW)” to prepare the LED strips and preparation of the sorting bins as necessary. This includes setting up the parameters for the database connections for the database types.

The customer is also responsible for providing a 1200mmx1200 ½” MDF table top (see sample table drawing in the appendix at the end of this document). When mounting the measuring assembly, they should be mounted in such that the sensors are not detecting the table or the aluminum extrusion. The sensor height should be adjusted to approximately 3-6mm below the workpiece.

During the ranging routine, the stepper motors will move to measure the workpiece. Keep your hands and fingers away from the moving carriage.



If Sensor 3 is used for the ranging trigger, set the sensor height should be adjusted to approximately 3-6mm below the workpiece. You can adjust the delay time for the detection of the workpiece by adjusting parameter *mk_sensor3_trigger_delay*

```
mk_sensor3_trigger_delay = 3.0
"""Delay the trigger signal for sensor3 (in seconds) -applies to mk_measure_mode = 4 only"""
```

1. Connect the controller to the network
2. Connect to the controller to setup the parameters
    a. Open the web browser and on the address bar enter:
    http://192.2.50.100:8100

    b. Alternatively, you can reach the controller via its default hostname:
    http://lightkitrpi1:8100

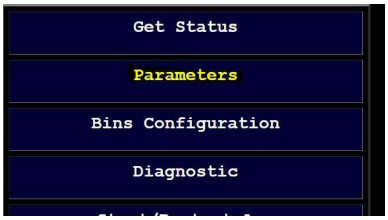
    

    Note: if you have more than 1 controller in the network, you must use the IP address method

    c. If the controller is connected via the customer network and using DHCP, you must used the issued IP address

    http://[ip address]:8100

3. From the web interface, click on the *Parameters* link – this link is password protected. Please contact your distributor to commission the parameters.



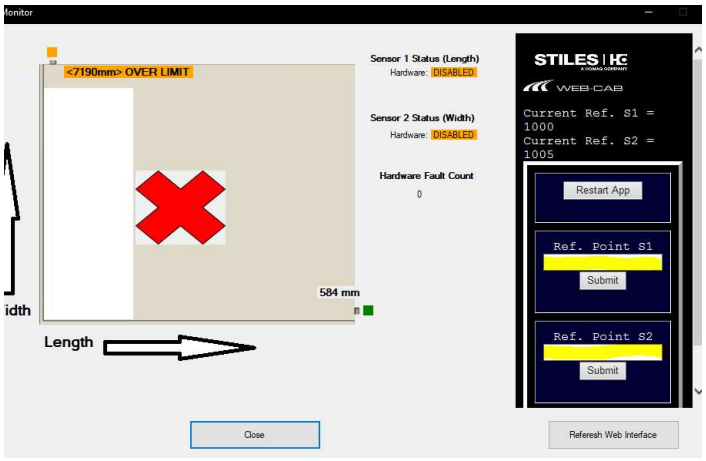
Relevant parameter entries must be setup that pertains to Database Type 5.

The entries are case sensitive!.

Parameter	Description	Notes / Possible Values
mk_sensors_type	Type of sensor being used	SPAD (VL53L0X, possible future TOF10120), Reflective Photosensor, Capacitive Proximity. Current support for VL53L0X ToF. Photosensors recommended due to operating voltage and accuracy.
mk_number_of_sensors	Number of sensors connected	2 for X and Y axes 3 for an additional trigger sensor. Adjustments via mdb-lightKit Visual Interface Tool.
mk_measurement_tolerance	Tolerance for measurement value	Value is subtracted from and added to the measurement for a range. Set in mdb-lightKit Visual Interface Tool.
mk_measurement_sample_size	Sample size for	Larger number for a slower response. Used with

Parameter	Description	Notes / Possible Values
	averaging measurement data	VL53L0X for averaging ranging data for accuracy and noise filtering.
mk_sort_with_light	Check bins for commands if set	Enables bin sorting based on criteria set by end-user programs. Command execution for sorting bins.
mk_sort_mode	Sorting mode when using mk_sort_with_light	1 = by dimensions 2 = by sort keys 3 = use bin numbers of db type 1 (webcab) 4 = db type 9 (mdb-lightKit). Modes 1 and 2 reserved for future development.
mk_debug_mode	Enable debug mode for sensor data logging	True to log sensor data each cycle. Adds 1 second to cycle time. Useful for commissioning sensors.
mk_sensor_cycle_time	Cycle time after reading sensor data	Value in seconds. 0.0 for real-time reading with TOF sensors. Set in mdb-lightKit Visual Interface Tool.
mk_sensor_quality	Quality mode for the sensor (VL53L0X)	1 = high accuracy (200ms) 2 = high speed (20ms). Set in mdb-lightKit Visual Interface Tool.
mk_measure_mode	Measurement Mode	1 = Continuous 2 = One Shot with Switch 3 = Database Trigger 4 = Sensor3 Trigger. Updates in MKInterfaces table for triggers.
mk_sensor3_trigger_range	Trigger range for additional sensor	Distance for sensor3 to be considered as triggered. Set in mdb-lightKit Visual Interface Tool.
mk_sensor3_trigger_delay	Trigger delay for additional sensor	Delay in seconds. Set in mdb-lightKit Visual Interface Tool.
mk_use_pixels	Use of LED strip for indicators	For visual indicators using an LED strip. Set in mdb-lightKit Visual Interface Tool.

Sensor offsets or reference can be set in the mdb-lightKit Visual Interface Tool:



Adjust the Reference Points Until the measurements are correct.

Special interface tool from web-cab Production Assistant is provided by web-cab for referencing and interfacing directly with the controller. Please refer to web-cab Production Assistant user's manual for further information.

During the boot up sequence and there are visualization pixels are installed:

Color	State	Description
Orange	Initialization	Indicates that the system is powering up and initializing.
Blue	Initialization Complete	System has completed the boot-up initialization.
Green	Ready for Reference	System is ready to begin referencing procedure.
Orange - Blinking	Referencing	Referencing process is underway.
White - Solid	Obstruction Detected	Sensors are obstructed; motors are inhibited from movement.
Blue - Solid	Reference Complete for Axis	Referencing for the specific axis is done.

During operations:

Color	State	Description
Blue - Marquee	Measurement/Movement Active	Measurement or movement is currently in progress.
Blue - Position	Indicative Positioning	Current measurement and position indicated by pixels.
Blue - Bright > Dim	Motors Sleeping	Indicates that the motors are

Color	State	Description
		in sleep mode.
Green - Solid	Successful Measurement	Measurement has been completed successfully.
Red - Blinking	Emergency Stop	An emergency stop has been activated for both axes.
Red - Solid	Error or Unsuccessful	An error has occurred, or the measurement was not successful.

Command to the LKInterface table will activate the pixels with user defined colors. This can be used as visual feedback from superior systems. For example, if a record with the measurement value is found, command output 1 status to 11 of the LKinterface table can activate the pixels solid green.

10.1 MDB-MEASUREKIT VARIANT PARAMETERS

The mdb-measureKit Variant parameters are the advanced set of parameters for motion control. Settings in in this parameter has a direct impact on the operation and performance of the axes.

10.1.1 Global Variables

Global Variables – Do not edit these variables because they are only used internally.

Parameter	Description
START_REF_RUN	Start reference run for the axis
MK_AXES_REFERENCED	Axes referenced marker
AXES_IN_REF	Axes currently in reference
AXES_IN_RANGING	Axes currently ranging
MK_HARDWARE_FAULT	Marker if hardware has a fault or if failed to initialize
OUTPUT_DATA1	Formatted output for sensor 1
OUTPUT_DATA1_MIN	Formatted output for sensor 1 minimum
OUTPUT_DATA1_MAX	Formatted output for sensor 1 maximum
OUTPUT_DATA2	Formatted output for sensor 2
OUTPUT_DATA2_MIN	Formatted output for sensor 2 minimum
OUTPUT_DATA2_MAX	Formatted output for sensor 2 maximum
PREVIOUS_LENGTH_PIXELS	Buffer for the previous calculated length
PREVIOUS_WIDTH_PIXELS	Buffer for the previous width pixels
MK_HARDWARE_FAULT_COUNT	Count hardware faults

10.1.2 General Parameters

The general section of the parameters is used for some miscellaneous and general options - change as required. The values used in this document are the default values.

Parameter	Value	Notes
LENGTH_PIXEL_START	0	Start pixel for length
LENGTH_PIXEL_END	35	End pixel for length
WIDTH_PIXEL_START	36	Start pixel for width
WIDTH_PIXEL_END	71	End pixel for width
PIXEL_PITCH	33.25	Distance between pixels in millimeters
PIXEL_COLOR	(0, 0, 200)	Color of the pixels while ranging (RGB value)
LENGTH_PIXEL_END_MEASURE_OFFSET	-1	Offset pixels for visualization of length measurement
WIDTH_PIXEL_END_MEASURE_OFFSET	-2	Offset pixels for visualization of width measurement
VISUALIZATION_OPTION	2	Visualization option 1 = blue marquee 2 = current position)

10.1.3 Speed Control and Motion Control Parameters

The speed control and motion control parameters are used to control the behavior of the axes motors. Change the parameters as required for optimized performance of the mdb-measureKit.

Changes in this section of the mdb-measureKit Variant Parameters can cause the axis to be inoperable or perform poorly (inaccurately). It can also cause some jamming and collision.

Parameter	Value	Notes / Calculations
VERY_SLOW	500	
SLOW	1000	
FAST	2000	
FASTEST	4000	
REFERENCE_SPEED	[FASTEST, VERY_SLOW, VERY_SLOW]	
RANGING_SPEED	[FASTEST, VERY_SLOW, VERY_SLOW]	
GOTO_SPEED	[FASTEST, FASTEST, FASTEST]	
MINIMUM_POS	[0.0, 120.0, 120.0]	
MAXIMUM_POS	[0, 1200.0, 1200.0]	
PARK_POS	[0.0, 600.0, 600.0]	
STEP_RES	1.8	Stepper motor resolution
STEP_SIZE	4	Step Size
STEPS_PER_REVOLUTION	(360/STEP_RES) * STEP_SIZE	Number of steps in 1 revolution (Original formula preserved)
PULLEY_TEETH	20	Number of teeth on the pulley
PULLEY_PITCH	2	Pitch size in mm
PITCH_DIAMETER	PULLEY_TEETH * PULLEY_PITCH / π	Formula to calculate pitch diameter
STEPS_PER_MM	STEPS_PER_REVOLUTION / (PULLEY_TEETH * PULLEY_PITCH)	Number of steps per mm (Original formula preserved)

Parameter	Value	Notes / Calculations
TOLERANCE_REFERENCE	0.2	Reference tolerance
TOLERANCE_MEASUREMENT	0.2	Measurement tolerance
TOLERANCE_REFERENCE_VALIDATION	0.1	Reference validation tolerance
ACCURACY_MODE	True	False = measure 1x True = ML/Statistics (3x)
REFERENCE_DIRECTION	0	0 = REV, 1 = FWD
REVERSING_WAIT_TIME	0.5	Wait time before reversing motor
SLEEP_TIMER	60	Sleep motors after n minutes of inactivity
MOTION_CONTROL_MODE	1	0 = no accel/decel control 1 = using time values 2 = using tables
ACCELERATION_TIME	0.05	Acceleration time (range 0.02 - 0.08)
DECELERATION_TIME	0.05	Deceleration time (nominal range 0.02 - 0.08)
MOTION_CONTROL_MODE_2_INTERVAL	0.01	Interval value for MOTION_CONTROL_MODE 2
USE_STEP_SIZE_TOLERANCE	False	When true, uses step size for validation; faster but less accurate

10.1.3.1 Recommended Speed Control and Motion Control Parameters

The smoothness of mdb-measureKit axes operation is greatly influenced by the speed/frequency setting. The listed values are provided as acceptable frequencies (in hertz) for different speed settings, ensuring smooth operation. Lower frequencies are typically associated with more precise, smoother movements, while higher frequencies can achieve faster, though potentially less smooth, operations. Users should select a frequency that matches the desired balance of speed and smoothness, considering the resolution and precision required.

Speed (Hz)	Notes
8000	Upper limit for high-speed operations
4000	
2000	
1600	
1000	Mid-range speed, balance of speed and control
800	
500	
400	
320	
250	
200	
160	Lower speed, smooth operations
100	

Speed (Hz)	Notes
80	
50	
40	Lower limit for precision at slow operations
20	
10	Minimum speed for ultra-fine control

Below is the step resolution for the stepper motors. In general, the higher the step size, the higher the resolution is per step.

STEP_SIZE	RESOLUTION	RECOMMENDED NOMINAL SPEED (FASTEST, FAST, SLOW, VERY SLOW)	VELOCITY AT FASTEST
1	0.2mm per step	1000, 500, 250, 100	200mm/sec
2	0.1mm per step	2000, 1000, 500, 250	200mm/sec
4	0.05mm per step	4000, 2000, 1000, 500	200mm/sec
8	0.025mm per step	8000, 4000, 2000, 1000	200mm/sec
16	0.0125mm per step	8000, 4000, 2000, 1000	100mm/sec

Below is the recommended speed settings for a given step size value

STEP_SIZE	REFERENCE SPEED	RANGINGSPEED	GOTO SPEED	TOLERANCE REFERENCE	TOLERANCE MEASUREMENT	TOLERANCE REFERENCE VALIDATION
1	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, ASTEST, FASTEST	.2 (MIN = .2)	.2 (MIN = .2)	.2 (MIN = .2)
2	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, ASTEST, FASTEST	.2 (MIN = .1)	.2 (MIN = .1)	.2 (MIN = .1)
4 (DEFAULT)	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, ASTEST, FASTEST	.2 (MIN = .05)	.2 (MIN = .05)	.1 (MIN = .05)
8	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, ASTEST, FASTEST	.1 (MIN = .025)	.1 (MIN = .025)	.1 (MIN = .025)
16	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, VERY_SLOW, VERY_SLOW	FASTEST, ASTEST, FASTEST	.1 (MIN = .0125)	.1 (MIN = .0125)	.1 (MIN = .0125)

The higher the STEP_SIZE, the more precise the measurement will be but the higher the STEP_SIZE, the weaker the motor becomes. It is important that the axes are very free when setting the STEP_SIZE higher than 4. Higher STEP_SIZE also makes the motors less jerky.

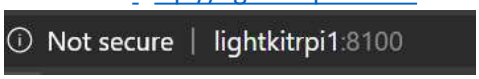
The speed can exceed the practical limit of the step size! Do not overspeed at a given step size or some steps can be lost and/or the motor cannot move properly.

11 INSTALLATION AND SETUP – DATABASE TYPE 7 (MDB-RFIDKIT)

When the controller is setup as database type 7 mdb-rfidKit variant, the controller can read RFID tags and record the values in the internal database table. The controller uses long range UHF antennas as RFID readers. The configuration can be customized based customer requirements. To configure, edit the configuration via the web interface:

1. Connect the controller to the network
2. Connect to the controller to setup the parameters
    a. Open the web browser and on the address bar enter: http://192.2.50.100:8100

    b. Alternatively, you can reach the controller via its default hostname: http://lightkitrpi1:8100



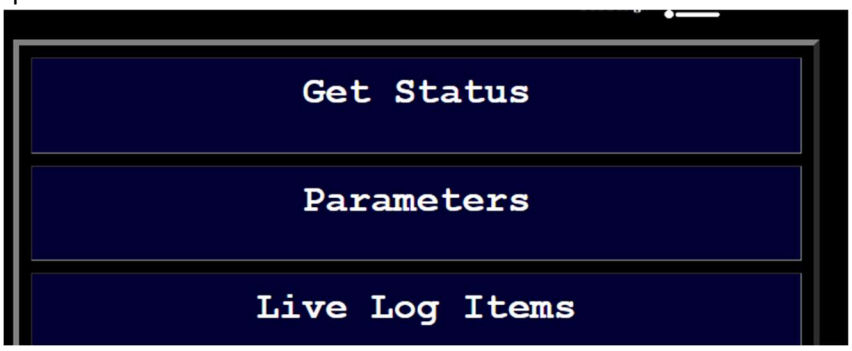
    Note: if you have more than 1 controller in the network, you must use the IP address method

    c. If the controller is connected via the customer network and using DHCP, you must used the issued IP address

    http://[ip address]:8100

4. From the web interface, click on the *Parameters* link – this link is password protected. Please contact your distributor to commission the parameters.

a.



5. Set the following parameter values:

    production_mode

    False

    - the RFID tags are simulated only. This is mainly used for development purposes

    True

- the controller will be in production mode and the RFID tags are read by the antenna. This is the normal mode of the controller in production

querry_interval

0.02 - N

- the refresh rate for reading the data from the RFID antenna. The default value is 0.05 seconds. This parameter should not be set lower than 0.02 seconds
- if the antenna type is set to type 2 or 3, the interval is ignored. New tag data is detected and processed as received

database_type

7

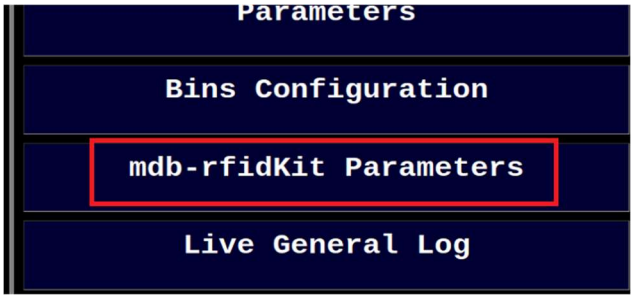
- the controller is configured as mdb-rfidKit variant

number_of_pixels

1 - N

- the number of pixels connected to the controller. This is optional. LED pixels are not required for the mdb-rfidKit variant to be operational. The default is 30

6. Set the following mdb-rfidKit parameters



antenna_type

1, 2 ,3

1

- Legacy antenna
- This antenna type requires a query command to receive data

2

- V2 type antenna
- The antenna sends the data automatically to the controller without request commands

3

- SR681 type
- The antenna sends the data automatically to the controller without request commands
- The power level and modulation values of the antenna can be adjusted via the mdb-rfidKit parameters

baud_rate

57600 or 115200

57600

- the default communication baud rate for legacy antennas. It is also the default baudrate for the SR681 type antennas (type 3)

115200

- the default communication baud rate for the v2 antennas

read_mode

0, 1, 2

0

- Command -- Antenna only activates and sends data when commanded

1

- Active -- Antenna sends the data automatically when data is present

2

- Passive -- Antenna reads tags but keeps in memory until commanded to send

Default = 0

Command mode is default

use_trigger

True, False

Use external sensor to read the data from SR681 antenna

refresh_rate

0.1 - n

If Antenna type is 3 and read mode is Command (1) or Passive (3)

Data is checked from the SR681 antenna at this interval in seconds

Default = 0.100

e*xpected_tag_length*

0 – N

0

- Disabled

Greater than 0

- When a number is entered here, the EPC ID or TAG ID length is checked if it matches this parameter. TAG ID or EPC ID length that does not match this parameter will be ignored. This is useful when you know the expected length of the TAG ID or EPC ID

ignore_tags

True or False

False

- Tag detection and recording is active. When this is set to False, the controller will record each tag detected.

True

- The detected tags will be ignored by the controller, effectively disabling the recording of detected tags. When this is set to True, no tag records will be written to the database.

This parameter can also be set by an fi_interface command.

- When the file command *accept_tags* is deposited in the fi_interface folder, this parameter is set to **False** automatically.
- When the file command *ignore_tags* is deposited in the fi_interface folder, this parameter is set to **True** automatically.

filter

[string value]

Tags containing the filter substring will be ignored.

Null or empty value will disable the filter. If this parameter is used, only tags that do not contain the filter string will be recorded and therefore no output file and no database record will be created

threshold

[hex value 1][hex value 2]

Default:

[0x01, 0x20]

- The signal threshold for tag detection. If the signal threshold is set too low, the system may detect and decode too many false signals, which can reduce its

accuracy. On the other hand, if the signal threshold is set too high, the system may not detect weak signals from tags that are far away or hidden behind obstacles, which can reduce its range. Adjust only when necessary.

verbose

True or False

False

- The controller will run in normal verbosity mode. Errors are logged into the database and functional logs are logged into the fo_interface/log.txt. This is the default mode

True

- The control will run in verbose mode. When the application is started from the command line, the shell will display additional information. Additional log entries will also be added to the log files. This is used for troubleshooting purposes.

dev_fi_trigger

True or False

False

- Fi_interface command “single_n” will not trigger a single read command

True

- Fi_interface command “single_n” will trigger a single read command to read the data from antenna in a single burst mode. Automatic data retrieval will be de-activated

Default = False

Set this parameter to False when not in use. This parameter is used in troubleshooting or development mode

After setting up the parameters, re-start the application from the web interface. The controller should now be operating as mdb-rfidKit variant. Changes to the parameters should now become active.

12 INSTALLATION AND SETUP – DATABASE TYPE 8 (MDB-FEEDBACKKIT)

When the controller is setup as database type 8, then it will function as mdb-feedbackKit. The mdb-feedbackKit is used to create connections to machines or workstations that has feedback functions. The controller will query the feedback table of the machines or workstations and create a record of it in its own feedback database table. Single or multiple connections can be set with the only maximum limitation is the performance of the network connection and controller response time. The configuration of the connection can be customized based on connection requirements. To configure the connections, edit the configuration via the web interface:

1. Connect the controller to the network
2. Connect to the controller to setup the parameters
    a. Open the web browser and on the address bar enter:
    http://192.2.50.100:8100

    b. Alternatively, you can reach the controller via its default hostname:
    http://lightkitrpi1:8100

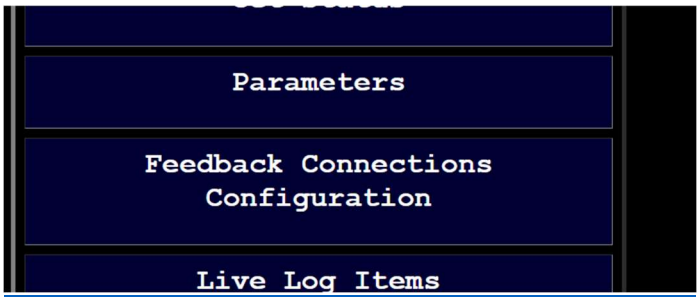
    

    Note: if you have more than 1 controller in the network, you must use the IP address method

    c. If the controller is connected via the customer network and using DHCP, you must used the issued IP address

    http://[ip address]:8100

7. From the web interface, click on the *Parameters* link – this link is password protected. Please contact your distributor to commission the parameters.



Set the *database_type* = 8

Now set the connections to the machines or workstations by clicking on Feedback Connections Configurations

Individual connections must be defined in the configuration file. Standard configurations are added in the configuration file for connections to Homag machines:

6. Homag CNC with legacy feedback table
    a. Venture
    b. BHX
    c. BHC
    d. BHP
    e. ABD
    f. centaTeq
7. Homag powerTouch control CNC
    a. Venture
    b. BHX
    c. BHC
    d. BHP
    e. ABD
    f. centaTeq
8. Homag Panel Dividing HPS Robot
9. Homag Panel Dividing saws
    a. HPP
    b. HKL
10. Homag edgeTeq edgebanders with feedback connections including KAL

The configuration of a connection in the Feedback Connections Configurations takes 4 steps:

1. Change the controller name

If the plant has multiple mdb-feedbackKit controllers, customize the controller name. The controller name is used as the “Source” field in the controller FeedBacks table.

```
#change controller name as necessary
controller_name = 'mdb-feedbackKit1'
```

2. Define the connection function

Standard connection definitions are provided, so you can choose which ones to use or copy as an initial connection definition.

```
def get_feedback_HomagCNC_powerTouch(pixel_number, interval):
    """Get feedback from CNC machine with powerTouch feedback database table.

    Args:
        pixel_number (int): the pixel number to display connection status
        interval (float, optional): The check interval in seconds.
    """

    #-----------------------------------------------------------------------------
    #                             1. Configure                                    -
    #-----------------------------------------------------------------------------

    #machine connection string
    machine_name = 'centaTeq'
    #connection connection
    server='172.16.1.201\SQLEXPRESS'
    database='WeekeProcessData'
    uid='machine'
    pwd='user'

    #select sql statement to get new records from the machine
    #limit results to improve performance
    select_string = """SELECT Top 10 *
                        FROM dbo.MESFeedbackItems
                        WHERE State = 10
                    """

    #the sql statement to update the machine feedback status
    #Modifify the field name for the WHERE clause as required
    update_string = """UPDATE dbo.MESFeedbackItems
                        SET State = 20
                        WHERE Reference = ?
                    """

    #other field values to insert in the records
    #modify as required
    reference_field_num = 1     #field number of the unique part ID
    step_code = 'machining'
    workcenter_code = machine_name
    quantity_good = 1           #quantity can come from machine feedbacks record also
    quantity_scrap = 0          #quantity can come from machine feedbacks record also
    quantity_rework = 0         #quantity can come from machine feedbacks record also
    source = controller_name
```

3. Configure the connection string, SQL statements, and default field values

Configure the connection string and the proper credentials for the machine or workstation where the feedback records are to be read from.

```
def get_feedback_HomagCNC_powerTouch(pixel_number, interval):
    """Get feedback from CNC machine with powerTouch feedback database table.

    Args:
        pixel_number (int): the pixel number to display connection status
        interval (float, optional): The check interval in seconds.
    """

    #-----------------------------------------------------------------------------
    #                              1. Configure                                  -
    #-----------------------------------------------------------------------------

    #machine connection string
    machine_name = 'centaTeq'
    #connection connection
    server='172.16.1.201\SQLEXPRESS'
    database='WeekeProcessData'
    uid='machine'
    pwd='user'

    #select sql statement to get new records from the machine
    #limit results to improve performance
    select_string = """SELECT Top 10 *
                        FROM dbo.MESFeedbackItems
                        WHERE State = 10
                    """

    #the sql statement to update the machine feedback status
    #Modifify the field name for the WHERE clause as required
    update_string = """UPDATE dbo.MESFeedbackItems
                        SET State = 20
                        WHERE Reference = ?
                    """

    #other field values to insert in the records
    #modify as required
    reference_field_num = 1     #field number of the unique part ID
    step_code = 'machining'
    workcenter_code = machine_name
    quantity_good = 1           #quantity can come from machine feedbacks record also
    quantity_scrap = 0          #quantity can come from machine feedbacks record also
    quantity_rework = 0         #quantity can come from machine feedbacks record also
    source = controller_name
```

Configure the SQL Statements next. Make sure to edit the database table names in the *select_string* and *update_string*. In the update_string, make sure to change the State value to the correct value for the machine or workstation. In general, the feedback connections to the machines will require you to mark the state or status of the record that is read. Thus, an update SQL statement string must be defined properly. Make sure to get the specification for the value in the status field from the manufacturer of the machine or workstation.

For example, in Homag machines “State = 10” means new record and “State = 20” means that the record has been read by a superior system. In this case, when the controller reads the record successfully, it must update the record status at the machine to 20.

```
def get_feedback_HomagCNC_powerTouch(pixel_number, interval):
    """Get feedback from CNC machine with powerTouch feedback database table.

    Args:
        pixel_number (int): the pixel number to display connection status
        interval (float, optional): The check interval in seconds.
    """

    #-----------------------------------------------------------------------------
    #                             1. Configure                                    -
    #-----------------------------------------------------------------------------

    #machine connection string
    machine_name = 'centaTeq'
    #connection connection
    server='172.16.1.201\SQLEXPRESS'
    database='WeekeProcessData'
    uid='machine'
    pwd='user'

    #select sql statement to get new records from the machine
    #limit results to improve performance
    select_string = """SELECT Top 10 *
                        FROM dbo.MESFeedbackItems
                        WHERE State = 10
                    """

    #the sql statement to update the machine feedback status
    #Modifify the field name for the WHERE clause as required
    update_string = """UPDATE dbo.MESFeedbackItems
                        SET State = 20
                        WHERE Reference = ?
                    """

    #other field values to insert in the records
    #modify as required
    reference_field_num = 1     #field number of the unique part ID
    step_code = 'machining'
    workcenter_code = machine_name
    quantity_good = 1           #quantity can come from machine feedbacks record also
    quantity_scrap = 0          #quantity can come from machine feedbacks record also
    quantity_rework = 0         #quantity can come from machine feedbacks record also
    source = controller_name
```

```
def get_feedback_HomagCNC_powerTouch(pixel_number, interval):
    """Get feedback from CNC machine with powerTouch feedback database table.

    Args:
        pixel_number (int): the pixel number to display connection status
        interval (float, optional): The check interval in seconds.
    """

    #-----------------------------------------------------------------------------
    #                             1. Configure                                    -
    #-----------------------------------------------------------------------------

    #machine connection string
    machine_name = 'centaTeq'
    #connection connection
    server='172.16.1.201\SQLEXPRESS'
    database='WeekeProcessData'
    uid='machine'
    pwd='user'

    #select sql statement to get new records from the machine
    #limit results to improve performance
    select_string = """SELECT Top 10 *
                        FROM dbo.MESFeedbackItems
                        WHERE State = 10
                    """

    #the sql statement to update the machine feedback status
    #Modifify the field name for the WHERE clause as required
    update_string = """UPDATE dbo.MESFeedbackItems
                        SET State = 20
                        WHERE Reference = ?
                    """

    #other field values to insert in the records
    #modify as required
    reference_field_num = 1     #field number of the unique part ID
    step_code = 'machining'
    workcenter_code = machine_name
    quantity_good = 1           #quantity can come from machine feedbacks record also
    quantity_scrap = 0          #quantity can come from machine feedbacks record also
    quantity_rework = 0         #quantity can come from machine feedbacks record also
    source = controller_name
```

Set the default fields as follows:

reference_field_num = 1

The field number of the unique partID or ReferenceID based on the result of the select_string statement

step_code = 'machining'

The step code for the machine. This is normally something like “cutting,” “machining,” “dowelling” etc. The value is customized per customer requirement.

workcenter_code = machine_name

This is typically set to the machine name defined previously but can be re-fined differently here based on customer requirements.

quantity_good = 1

Defaults to 1 as standard for machine connections. On connections other than machines, like a Quality Control workstation for example, this can be zero if it fails inspection.

quantity_scrap = 0

Defaults to 0. This can be set to 1, the *quantity_good* = 0, and *quantity_rework* = 0 for Quality Control stations for example.

quantity_rework = 0

Defaults to 0. This can be set to 1, the *quantity_good* = 0, and *quantity_scrap* = 0 for Quality Control stations for example.

source = controller_name

This defaults to the controller name previously defined. It is good practice to leave this as the controler name so that when a superior system reads that feedback record from the controller FeedBacks table, the record can be attributed to this controller.

4. Call the Connection Function for the connection definition

Call the function responsible for writing the records in the controller Feedbacks table. There is standard function for Homag machines. There is also another standard function for Homag Panel Dividing Saws such as HPS, HPP, and HKL, but more connection functions can be added as required. Contact the distributor for more information.

```
#-------------------------------------------------------------------------------
#                          2. Call The type8 function                          -
#                              and pass the parameters                         -
#-------------------------------------------------------------------------------
#now call the feedback function for this machine connection type
#modify the function name based on available functions for the machine feedback connection
is_type8_writeFeedback_homag_standard(select_string,
    update_string, \
    reference_field_num, \
    step_code, \
    workcenter_code, \
    quantity_good, \
    quantity_scrap, \
    quantity_rework, \
    source, \
    pixel_number, \
    server, \
    database, \
    uid, \
    pwd)

#-------------------------------------------------------------------------------
#                           3. Match the Function Parameter                    -
#                              with the task definition                        -
#-------------------------------------------------------------------------------
Timer(interval, get_feedback_HomagCNC_powerTouch, args = (pixel_number, interval)).start()

#End Definition
```

The standard functions currently defined:

is_type8_writeFeedback_homag_standard

is_type8_writeFeedback_homag_saw

```
def is_type8_writeFeedback_homag_standard(select_string, \
    update_string, \
    reference_field_num, \
    step_code, \
    workcenter_code, \
    quantity_good, \
    quantity_scrap, \
    quantity_rework, \
    source, \
    pixel_number, \
    db_server, \
    database_name, \
    user, \
    password):
    """For database type 8 writing new feedback records in standard way.
    Standard = no Data001xx being populated

    Args:
        select_string (str): select statement to querry the new feedback record from the machine
        update_string (str): the update statement to update the machine that the record has been rea
        reference_field_num (int): what field number is the reference/part ID
        step_code (str): the step code value for the machine (cut, band, bore, assy, fin, etc...)
        workcenter_code (str): the workstation code (BHX500, or Nesting1, etc)
        quantity_good (int): quantity good
        quantity_scrap (int): quantity scrap
        quantity_rework (int): quantity rework
        source (str): the name of the controller that created the record
        pixel_number (int): pixel to indicate status
        db_server (str): the IP address of the machine
        database_name (str): the database name for the feedback
        user (str): the connection user name
        password (str): the connection password
    """
```

is_type8_writeFeedback_homag_standard

```
def is_type8_writeFeedback_homag_saw(select_string, \
    update_string, \
    reference_field_num, \
    csv_field_number, \
    step_code, \
    workcenter_code, \
    quantity_good, \
    quantity_scrap, \
    quantity_rework, \
    source, \
    pixel_number, \
    db_server, \
    database_name, \
    user, \
    password):
    """For database type 8 reading from Homag Panel Dividing saws.  The PPCD_DATA (the amalgamation of all the fields in C
    is written in the Data001 field.

    Args:
        select_string (str): select statement to querry the new feedback record from the machine
        update_string (str): the update statement to update the machine that the record has been read
        reference_field_num (int): what field number is the reference/part ID
        csv_field_number (int): which field number is the csv values for Data001
        step_code (str): the step code value for the machine (cut, band, bore, assy, fin, etc...)
        workcenter_code (str): the workstation code (BHX500, or Nesting1, etc)
        quantity_good (int): quantity good
        quantity_scrap (int): quantity scrap
        quantity_rework (int): quantity rework
        source (str): the name of the controller that created the record
        pixel_number (int): pixel to indicate status
        db_server (str): the IP address of the machine
        database_name (str): the database name for the feedback
        user (str): the connection user name
        password (str): the connection password
    """
```

is_type8_writeFeedback_homag_saw

5. Make sure that the connection definition is correct in the thread call

```
#-------------------------------------------------------------------------------
#                          2. Call The type8 function                          -
#                              and pass the parameters                         -
#-------------------------------------------------------------------------------
#now call the feedback function for this machine connection type
#modify the function name based on available functions for the machine feedback connection
is_type8_writeFeedback_homag_standard(select_string,
    update_string, \
    reference_field_num, \
    step_code, \
    workcenter_code, \
    quantity_good, \
    quantity_scrap, \
    quantity_rework, \
    source, \
    pixel_number, \
    server, \
    database, \
    uid, \
    pwd)

#-------------------------------------------------------------------------------
#                          3. Match the Function Parameter                     -
#                              with the task definition                        -
#-------------------------------------------------------------------------------
Timer(interval, get_feedback_HomagCNC_powerTouch, args = (pixel_number, interval)).start()

#End Definition
```

```
def get_feedback_HomagCNC_powerTouch(pixel_number, interval):
    """Get feedback from CNC machine with powerTouch feedback database table.

    Args:
        pixel_number (int): the pixel number to display connection status
        interval (float, optional): The check interval in seconds.
    """

    #------------------------------------------------------------------------------
    #                              1. Configure                                    -
    #------------------------------------------------------------------------------

    #machine connection string
    machine_name = 'centaTeq'
    #connection connection
```

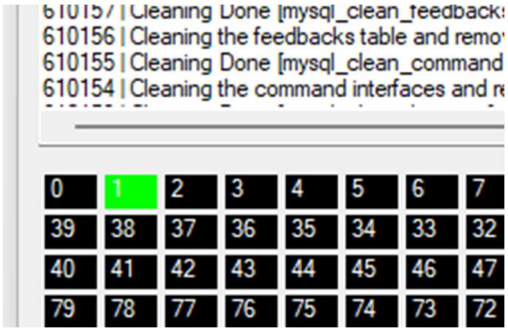
6. Include the definition in the call_active_tasks()

At the bottom of the configuration file, there is a defined function called *call_active_tasks.* Include the connection definition in the body of the call_active_tasks. Set the *pixel_number* and *interval* parameters.

```
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#**************************************************************************************
#                      Define Which Tasks are to be active                           *
#**************************************************************************************
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
def call_active_tasks():
    """Calls to tasks that are to be active.  Make function calls to all the defined tasks.
    Only defined tasks that are called here will be activated.
    """
    #call tasks
    get_feedback_HomagCNC_powerTouch(pixel_number=1, interval=1)
```

pixel_number = 1

The *pixel_number* is used by the controller to set an LED pixel if an LED strip is installed to display the connection status. For example machine connection definition 1 should have 1, for Machine 2 should be 2, etc.



An example: Here pixel number 1 is used as seen from the mdb-lightKit Visual Interface Tool

Green = Connection to machine or workstation is OK

Amber = Busy

Red = There is a problem with the connection. See the Log entries for more details on the error

Interval **=** 1.0

The interval time to re-check for new records in seconds. The default is 1 second. For machine connections where immediate record entry is required, it can be set to a shorter interval at the cost of more controller processing power requirements. On the other hand, for machines or workstations that do not require such fast-reading requirement, it can be set to a higher number. Take note that the query performance can vary based on network connection speed to the machine.

The controller can connect to multiple machines and workstations if the connection definitions exist. The definitions can be copied if the definition names are changed, and the connection configured as outlined above.

13 WEB INTERFACE OPTIONS AND OPERATIONS

The options displayed in the web interface depends on the database type. Options and configuration links are that unrelated to the database type as set in the parameters are hidden. All parameter links are password protected.

Get Status – Displays the current status of the controller including the log. It will also display the current network configuration and network information of the controller

Live General Log – Displays the last 100 errors and controller messages in descending order and refreshes every 2 seconds

Live Function Log – Displays the last 100 messages related to the specific db type functions in descending order and refreshes every 2 seconds. All Functional type messages are logged in this screen and in the log.txt located in the fo_interface folder

Parameters – Settings of the parameters

Variant Specific Parameters – Any parameters specific to the configured blueBox variant

Bins Configuration – Setting of Bins in db type 1 and 9

Db type specific parameters – Settings that are specific to the configured db type of the blueBox

Diagnostic – displays the explanation of the diagnostic blink codes

Open Current Fatal Error Log – Opens the fatal error log (errors that crashed the main application)

Stop App – Stops the running appplication

Start/Restart App – start/restarts blueBox app. This does not restart the controller

Reboot – reboots the controller

Resstart SMB file Sharing Service – Restarts the SMB file server. When too much time has elapsed from inactivity or when the network connection was lost, it is possible that the SMB file server requires a re-start for the shared folders become accessible again.

Shutdown – shuts down the control. There are no further indication that the controller has shut down. **After pressing this button, wait 1 minute before unplugging the controller**.

Make A Backup –creates a backup of the controller app and settings

Downloads Config and Parameters – will download a zipped copy of the parameters and the bins configuration

Downloads mdb-lightKit VI Tool – Download the Visual Interface Tool for Windows

Install Guides (DB Types 1, 2, 6,9)

Markers On – turns on every 10th pixel

Markers Off – turns off every 10th pixel from *Markers On* command

Pixels check – runs the red running pixel

Turn OFF all Pixels – Turns off all active pixels

[Open User Guide] – Opens this user guide

Cycle Through Bins (DB Types 1, 2, 6,9)

Infeed Bins – turns on each infeed bins for 2 seconds in sequence

Outfeed Bins – turns on each outfeed bins for 2 seconds in sequence

Service Page – this is for service personnel only. This is a password protected page

Setup Wifi Connection

Setup the wifi credentials so the controller can connect to a wifi connection

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=US

network={
        ssid="ATT5yJD4zc_EXT_2_x"
        psk="3x8skh?qwg%s"
        key_mgmt=WPA-PSK
}
```

A reboot is required for the changes to take effect!

Open User Guide – opens this user manual

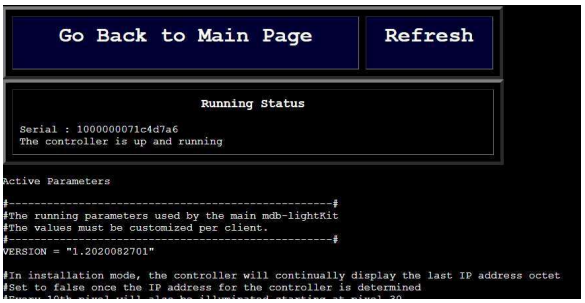
Open Electrical Schematic – Opens the Electrical Schematic

Open Mechanical Schematic – Opens the Mechanical Schematic

Service Page – Opens the password protected service page. This page is for service personnel only.

14 CONTROLLER STATUS

The status of the controller can be checked on the *Get Status* page of the web interface. A status file is also written in the fo_interface folder so that the controller status can be read by a superior system. The actual status is written in the file that can be opened by a text editor.



When the application is currently running, the message “The controller is up and running” will be displayed.

If the application is stopped, the message “The controller is not up” will be displayed.

The *Get Status* page will also display the active parameters running.

14.1 DIAGNOSTICS AND TROUBLESHOOTING

The controller uses the OLED display in the controller box to display the status and errors. Pixel 0 is also used by default to indicate the error status when the OLED display could not be read. Which pixel is used as a diagnostic pixel can be set in the **Diagnostic Specific Parameters** *diagnostic_pixel*. The diagnostics sequence of the diagnostic pixels is explained in the Diagnostic section of the web interface.

14.2 OLED DISPLAY

The OLED display on the controller will display relevant controller information when the main application is running.

Line 1: IP Address

Line 2: Access Point IP address

Line 3: current blueBox variant + blinking ¤ application status

The blinking ¤ signifies that the main application is currently running

Line 4: Status

TEST MODE -- The controller is in Test Mode

BUSY – The controller is busy with execution of functions

CONNECTING TO DB – The controller is in the connection to database phase

CONN FAILED – The connection to the database failed

CONN SUCCESS – The connection to the database was successful

DB NOT SUPPORTED – The variant type is currently not a supported type

EX ERR COUNT: num – Exceptions error. When there are exceptions during the execution of the main application, the number of errors is counted.

HEAT WARN xxC – Controller approaching heat limit, displays current controller temperature

INITIALIZING – The controller is in initialization phase

INSTALL MODE – The controller is running in install mode.

INVALID LICENSE – The application license number is invalid

INSTALL MODE – The controller is in Install Mode

LED TEST DONE – The marquee function completed for variants that has LEDs

MISSING LICENSE – The license file is missing

OVERHEAT xxC – Controller is overheating, displays current controller temperature

PREPAIRING FILES – The controller is creating operational files

READY FOR CMDS – The controller is ready and waiting for commands

mdb-lightKit Variant:

BUSY – The variant is busy with the queries. The main variant is running if this message is active together with the blinking ¤ character in Line 2

mdb-measureKit Variant:

[axis num] NOT CLEAR – there is a workpiece in the table and the axis cannot perform its referencing function

AXIS[axis num] REF – Axis[axis num] is referencing

AXIS[axis num] ERROR – Error in [axis num] while it was doing a function

AXIS FAULT – Fault detected in one of the axes

INVALID MODE – Parameter entry for *mk_measure_mode* is invalid

MODE [mode num] ACTIVE – mdb-measureKit parameter *mk_measure_mode* is in mode [mode num]

RANGING – the ranging/measure function is active

READING SENSORS – Sensors are being queried

REF ROUTINE [axis num] – Axis [axis num] is referencing

REF FAULT – Fault detected during the referencing routine

STANDING BY - variant is waiting for commands

SENSORS FAULT – Sensor fault(s) detected

SENSORS INIT – Sensors are initializing

mdb-rfidKit Variant:

COMM OK – Communications to the RFID Antennas have been established

INITIALING – Variant functions are initializing

INIT DONE – The initialization of variant functions complete

ERROR – There is a general error in the variant functions

LISTENING – The controller is listening for RFID Tags

OPEN COMM – The variant is establishing communication to the RFID antennas

mdb-feedbackKit Variant:

READING FB – Controller is reading from a feedback connection

mdb-hasKit variant (Homag automatic storage)

SLEEPING – The pixels are turned off to reduce power consumption

Line 5: Date Time

Line 6: Status 2

mdb-rfidKit Variant:

ANT[ant num]:

IDLE -- Antenna is waiting to detect tags

INIT – Antenna is in initialization phase

READY – Antenna is ready

RX – Antenna is receiving data

TX – Antenna is transmitting data

mdb-measureKit Variant:

Last measurement X: nn.n *Last Measurement Y*: nn.n *Sleep Counter*

Line 7: Controller internal Temperature, CPU Load, and number of threads running

Line 8: Controller Status

Status Numbers:

0	starting
10	initializing
	reading
20	initialized
	complete
30	controller not running
40	controller error
99	controlling stopping
100	estop activated

The OLED has a secondary and tertiary display mode.

Secondary mode – Application Status Watchdog display:

When the main application is not running, the OLED will display that the main application is not running. You can restart the main application using the web interface. Please refer to section 13 Web Interface Option and Operations.

Line 1: Ethernet IP Address

Line 2: Access Point IP Address

Line 3: “WARNING”

Line 4: “NOT RUNNING”

Line 5: “MAIN APP STOPPED”

Line 6: Date and Time

Line 7: CPU Temperature

Tertiary Mode – Emergency Display:

When there is a fatal error and the primary and secondary OLED display mode is not operational, the OLED display will activate its tertiary mode after 1 minute.

This mode will display the most basic information required to connect to the blueBox controller, and it will display critical information that can be used by service personnel for troubleshooting.

Line 1: Ethernet IP Address

Line 2: Access Point IP Address

Line 3: Core Voltage

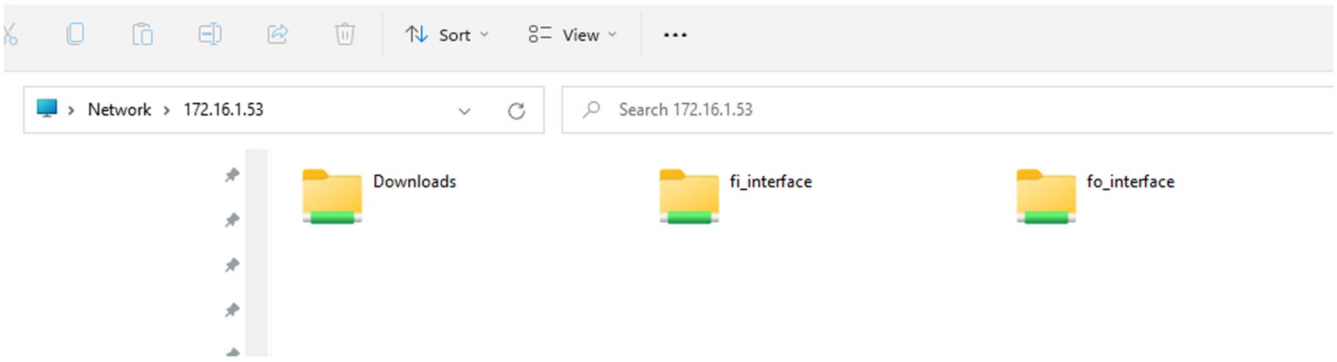
Line 4: Throttle Information

0x0 = CPU is running at full speed

0x00005 = CPU is throttled due to undervoltage

15 FILE INPUT AND FILE OUTPUT INTERFACES

The controller supports file input commands via the fi_interface folder. It also creates file outputs that are deposited in the fo_interface folders. These file interfaces can be used by a superior system to send commands to the controller or query the status using files.



15.1 FI_INTERFACE

The fi_interface can be used to send commands to the controller. When a blank file with a specific filename is deposited in the fi_interface folder, the controller will interpret the filenames as commands to be sent to the controller. Commands that are not valid will be ignored.

Available commands:

General:

on	Turn on every 10th pixel
off	Turn off pixels that were turned by the *on* fi_interface command
marquee	Activates the marquee function
all_off	Turns off all pixels
connect_wifi	The controller will attempt to connect to the wi-fi using the settings from the *Setup Wifi Connection* page of the web interface
kill_app	The main application will be stopped. No functions will execute. A restart of the application from the main web interface will be necessary to continue with the functions

mdb-lightKit Variant:

cycle_infeed	Turns on each infeed bin one by one at 1 second interval
cycle_outfeed	Turns off each outfeed bin one by one at 1 second interval
commitbinsparam	Commits the bins configuration. This is used if the bins configuration was edited by an external system such as the mdb-lightKit visual interface tool

enterconfig	The controller enters configuration mode. This command is only used by the visual interface tool
exitconfig	The controller exits the configuration mode. This command is only used by the visual interface tool

mdb-measureKit Variant:

disable_motors	The stepper motors of both axes will be disabled. All reference points will be lost. The motors will stay disabled until a re-start of the main application is executed
goto_x_n.n	The axis x will move to position n.n
goto_0_n	Starts the endurance run. The axes will go to random positions every 10 seconds for n cycles. This can be used to test the axes functions or put the mdb-measureKit in demonstration mode.
goto_0_n.n_m.m	Both axes will move simultaneously to positions n.n and m.m respectively: n.n = Position for axis 1 (Length) m.m = Position for axis 1 (Width) Example: goto_0_536.8_800.7 will move axis 1 (Length) to position 536.8 and axis 2 (Width) to 800.7 simultaneously
reference_axes	The mdb-measureKit will start the reference routine without a re-start of the application
sleep_axes	Puts the motors to sleep

mdb-rfidKit Variant:

ignore_tags	RFID tags detected will not be processed by the controller effectively de-activating the recording of RFID tags to the database
alarm_pitch_duration	When the mdb-rfidKit is equipped with a buzzer, it can be activated as an audible indicator with 3 different tones for a specific duration in seconds: *pitch* = "high" 2.4 kHz *pitch* = "low" 1.2 kHz *pitch* = "deep" 400 Hz Example alarm fi_interface command to activate the buzzer in high pitch for 3 seconds: Command:

alarm_high_3

accept_tags RFID tags detected will be processed by the controller

power_n_xx Power (field strength) of antenna *n* will be set to xx (0-30) when antenna is type 3.

The power level of the antenna affects the range of the RFID system, so you want to set it to the minimum level that provides reliable tag detection. It also affects the penetrating power. In general, increasing the power level will increase the range of the RFID system, but it may also increase the likelihood of interference and signal reflection, which can reduce its ability to penetrate obstacles such as walls and other objects. In RFID systems, the power level of the antenna is typically set to the minimum level that provides reliable tag detection, which balances the trade-off between range and penetration. If you need to increase the penetrating power of the RFID system, you can try adjusting the gain.

It's important to keep in mind that the ideal power level for an RFID system will depend on a variety of factors, such as the environment, the type of tags being used, and the specific requirements of the application. To find the best power level for your system, you may need to experiment with different settings and evaluate the performance of the system in your specific environment.

Effect: Increasing the transmit power of the RFID reader's RF signal will generally increase the read range of the system. This means that the RFID reader can communicate with RFID tags at greater distances.

Considerations: While higher power can extend the read range, it can also lead to increased interference and regulatory compliance issues. It's important to ensure that your RFID system complies with local regulations on RF emissions.

power_n_get Reads the current power settings for antenna *n*

The result of the command is appended in log.txt and *current_power_n* status file located in the *fo_interface* folder

modulation_n_preset The Mixer Gain and IF Amp Gain for antenna *n* will be set to *preset* values.

The mixer gain and IF (intermediate frequency) amplifier gain are important parameters that affect the overall performance of the RFID system. High mixer and IF amplifier gains can increase the sensitivity of the system, which can improve its ability to detect weak signals, but it may also increase the likelihood of interference and noise. On the

other hand, lower mixer and IF gains can reduce the sensitivity of the system and reduce the likelihood of interference, but it may also reduce the range of the system.

In general, the mixer and IF amplifier gain should be set to the minimum level that provides reliable tag detection, which balances the trade-off between sensitivity and range. The ideal mixer gain for an RFID system will depend on a variety of factors, such as the environment, the type of tags being used, and the specific requirements of the application. To find the best mixer gain for your system, you may need to experiment with different settings and evaluate the performance of the system in your specific environment.

Mixer Gain:

Effect: Mixer gain is typically associated with the receiver circuitry of the RFID reader. Increasing mixer gain can improve the ability of the reader to detect weak signals from RFID tags, which can be beneficial in noisy or challenging RF environments.

Considerations: Higher mixer gain can also increase the sensitivity to noise and interference. It's crucial to strike a balance to avoid false readings or misinterpretations of tag data.

IF Gain (Intermediate Frequency Gain):

Effect: IF gain amplifies the intermediate frequency signal in the reader's receiver circuit. Adjusting IF gain can help improve the signal-to-noise ratio of the received signal, enhancing the reader's ability to distinguish tag responses from background noise.

Considerations: Similar to mixer gain, increasing IF gain too much can amplify noise along with the RFID tag signals. Careful tuning is necessary to achieve optimal performance without introducing excessive noise.

preset:

low	Mixer Gain = 6 dBm	IF Amp Gain = 12 dBm
medium	Mixer Gain = 6 dBm	IF Amp Gain = 24 dBm
high	Mixer Gain = 12 dBm	IF Amp Gain = 36 dBm

low preset can be used for detecting tags within 500-1000mm of the antenna.

medium preset can be used for detecting tags within 1-2m of the antenna

high preset can be used for detecting tags within 2-4m of the antenna

modulation_1_custom_xx_yy

The Mixer Gain will be set to *xx* and the IF Amp Gain will be set to *yy* for antenna *n*

xx and *yy* values are index numbers of pre-defined values

Table 1: Mixer Gain index values

Value	Level
0	0dBm
1	3dBm
2	6dBm
3	9dBm
4	12dBm
5	15dBm
6	16dBm

Table 2:IF Amp Gain index values

Values	Level
0	12dBm
1	18dBm
2	21dBm
3	24dBm
4	27dBm
5	30dBm
6	36dBm
7	40dBm

Example usage of the custom modulation *fi_interface* command:

Command:

modulation_1__custom_02_04

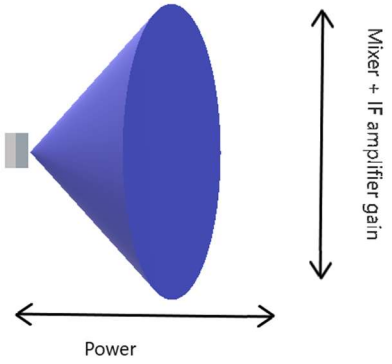
Result:

Antenna 1

Mixer Gain will be set to 6dBm

IF Amp Gain will be set to 27dBm

Practical effects of power, mixer gain, and IF amplifier gain settings



modulation_n_get Reads the current modulation settings for antenna *n*

The result of the command is appended in log.txt located in the *fo_interface* folder

read_mode_set_n_x Sets the read mode of the antenna *n* to x

x = 0 Command -Antenna only activates and sends data when commanded

x = 1 Active -Antenna sends the data automatically when data is present

x = 2 Passive -Antenna reads tags but keeps in memory until commanded to send

Default = 0

single_n When antenna is type 3 (SR681), this will send a single command to read the data from the antenna n. This is useful when troubleshooting data fidelity. This parameter is active when parameter *dev_fi_trigger* is set to True

basic_get_n When antenna is type 3 (SR681), this will send a command to read the current basic settings of antenna n.

basic_set_n_hhhh.. When antenna is type 3 (SR681), this will send a command to set the current basic settings of antenna n via hhhh. Hhhh is the data bytes for the settings. Refer to the manual for the proper protocol.

buzzer_n_x When antenna is type 3 (SR681), this will send a command to set the buzzer (beep) for antenna *n* to *x.*

x = 1 ON (beep at every tag detect)

x = 0 OFF (no beep)

serial_in_n When antenna is type 3 (SR681), the contents of this file will be sent to directly to antenna *n.* The content should have the command values in CSV format. The checksum should be omitted because the mdb-rfidKit will calculate it automatically before sending the command. The command values are in hex. The response for the command will be written in the fo interface filename *serial_in_response* located in the fo_interface folder.

Example content of the *serial_in_n file:*

7c,ff,ff,50,00,00

The basic format of the protocol

No.	1	2	3	4	5	6	7
byte	1	2	1	1	1	LENGTH	1
format	SOI	ADR	CID1	CID2	LENGTH	INFO	CHKSUM

No.	Symbol	significance	Remarks
1	SOI	START OF INFORMATION	Command(7CH) Response(CCH)
2	ADR	Equip address（1～65534）,(65535 public address,0 reserve address)	FFFFH
3	CID1	Control identification code (data type description)	
4	CID2	Command: control identification code (action type description) Response: RTN（Return code Table 2-3）	
5	LENGTH	INFO Data Length	
6	INFO	Command: Command information Response: Response data information	
7	CHKSUM	The checksum code	

The return code format

No.	RTN Value(HEX)	significance	Remarks
1	00H	Succeed	
2	01H	Fail	
	02H	Response massage for Command	
3	05H	Auto send to SU	

CID1 Codes

No.	Content	CID1	Remark
1	Read Type C UII	20H	
2	Read Type C Tag Data	21H	
3	Write Type C Tag Data	22H	
4	Lock Type C Tag	26H	
5	Kill/Recom Type C Tag	28H	
6	Encrypted Type C Tag	2AH	
7	Get Access EPC MATCH	2CH	
8	Set Access EPC MATCH	2DH	
9	Get Tx Power Level	50H	
10	Set Tx Power Level	51H	
11	Get Region	52H	
12	Set Region	53H	
13	Get current RF Channel	54H	
14	Set current RF Channel	55H	
15	Get Frequency Hopping Table *	56H	
16	Set Frequency Hopping Table *	57H	
17	Get Modulation *	58H	
18	Set Modulation *	59H	
19	Get Frequency Hopping status	5AH	
20	Set Frequency Hopping status	5BH	
21	Base Parameter	81H	
22	Antenna configuration *	83H	
23	Encryption method for Tag	84H	
24	Protocol Address	85H	
25	UART Baud rate *	86H	
26	Output Mode *	87H	
27	Reset System	D0H	
28	Update Registry	D2H	
29	Erase Registry	D3H	
30	Get GPIO Mode *	D6H	
31	Set GPIO Mode *	D7H	
32			

CID2 Codes

1	Senior command	00H
2	Set command	31H
3	Get command	32H

CID1 and CID2 Codes combination

No.	Content	CID1	CID2	Remarks
1	Read Type C UII	20H	00H	
2	Read Type C Tag Data	21H	00H	
3	Write Type C Tag Data	22H	00H	
4	Lock Type C Tag	26H	00H	
5	Kill/Recom Type C Tag	28H	00H	
6	Encrypted Type C Tag	2AH	00H	
7	Get Access EPC MATCH	2CH	00H	
8	Set Access EPC MATCH	2DH	00H	
9	Get Tx Power Level	50H	00H	
10	Set Tx Power Level	51H	00H	
11	Get Region	52H	00H	
12	Set Region	53H	00H	
13	Get current RF Channel	54H	00H	
14	Set current RF Channel	55H	00H	
15	Get Frequency Hopping Table *	56H	00H	
16	Set Frequency Hopping Table *	57H	00H	
17	Get Modulation *	58H	00H	
18	Set Modulation *	59H	00H	
19	Get Frequency Hopping status	5AH	00H	
20	Set Frequency Hopping status	5BH	00H	
21	Get Base Parameters	81H	32H	
22	Get Base Parameters	81H	31H	
23	Get Antenna configuration *	83H	32H	
24	Set Antenna configuration *	83H	31H	
25	Get Encryption method for Tag	84H	32H	
26	Set Encryption method for Tag	84H	31H	
27	Get Protocol Address	85H	32H	
28	Set Protocol Address	85H	31H	
29	Get UART Baudrate *	86H	32H	
30	Set UART Baudrate *	86H	31H	
31	Get Output Mode *	87H	32H	
32	Set Output Mode *	87H	31H	
33	Reset System	D0H	00H	
34	Update Registry	D2H	00H	
35	Erase Registry	D3H	00H	
36	Get GPIO Mode *	D6H	00H	
37	Set GPIO Mode *	D7H	00H	

15.2 FO_INTERFACE

The fo_interface can be used by a superior system to check the status and results of the controller variants. Some variants will deposit fo_interface files in the fo_interface folder that signifies functional items and function status.

This fo_interface files are meant to be read by the superior systems. Fo_interface files are also deleted and deposited automatically depending on the current controller operations. It is the responsibility of the superior system to read the files as needed. The content of the fi_interface file can also be read by the the superior system

Available fo_interface files:

General:

status	Controller status. This fo_interface file is always available.

mdb-measureKit Variant:

[sensor num]_ranging_done	Sensor num measurement complete
[axis num]_ranging_error	Error measuring axis num
[axis num]_reference_done	Axis num referencing routine done

mdb-rfidKit Variant:

[rfid tag]_[ant num].tag	Detected RFID tag from ant num
rfid_ok	The detected tag was written to the internal database successfully. The content of this file is the RFID Tag number
rfid_fault	The detected tag was not written to the internal database because it was faulty. The content of this file is the RFID Tag number
current_modulation_[ant num]	Contains the current antenna modulation settings for ant num after the successful execution of fi_interface command *modulation_1_get*
power_ok_[ant num]	Set power command was successful
power_fail_[ant num]	Set power command was not successful
modulation_ok_[ant num]	Set modulation settings command was successful
modulation_fail_[ant num]	Set modulation settings command was not successful
serial_in_response_[ant_num]	Content of file contains the response from *serial_in_[ant_num]* fi interface command
current_basic_settings_[ant_num]	Content of file contains the list of the current basic antenna settings

mdb-feedbackKit Variant:

[reference id].fb	When a feedback data is read from a machine, this file is created. The reference id is the feedback reference item number
feedback_ok	The feedback data from the machine is ok
feedback_fault	The feedback data from the machine is not ok or the wrong format

16 WIRELESS ACCESS POINT

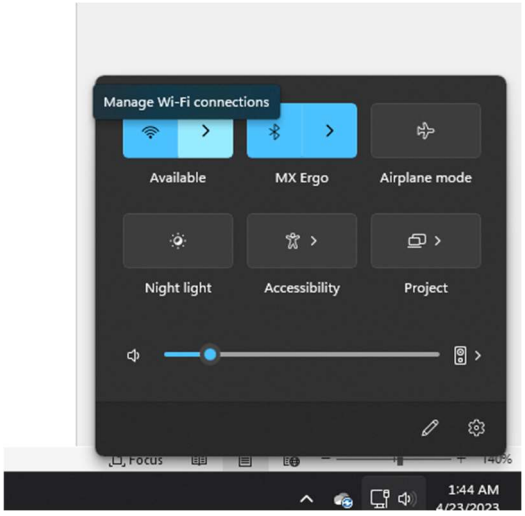
When connection to the blueBox using the Ethernet Port is not possible, the wireless Access Point can be used as peer-to-peer connection during commissioning. The blueBox controller will broadcast an SSID, "blueBox_serial number," so that it can be found by superior systems such as PCs with wireless adapters. The AP uses the 802.11n, 2.4Ghz WLAN channel.

Since the blueBox controller is fully enclosed in an aluminum enclosure, the connection may not be as reliable as a direct connection to the ethernet port because the signal strength is dampened significantly. It is therefore recommended to only use the wireless connection during commissioning or when a network connection to the ethernet port cannot be established.

16.1 ESTABLISH WIRELESS PEER-TO-PEER CONNECTION USING THE WIRELESS ACCESS POINT

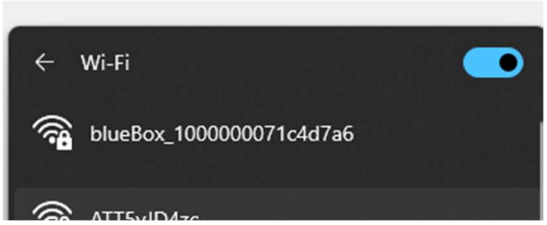
Below is a step-by-step instruction on how to connect to the blueBox controller's WAP using a PC running Windows 11 with a Wi-fi adapter. Follow the operating system instructions for other operating systems. Tablets, smartphones, and other smart devices that can run an internet browser can connect to the blueBox controller WAP.

1. Open your PC wireless connection settings.



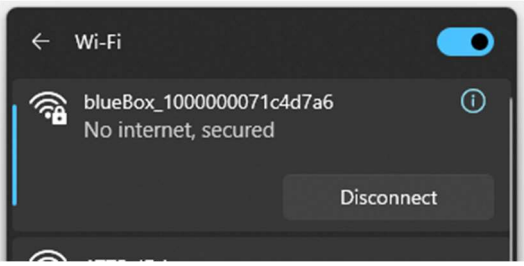
2. Locate the blueBox controller Access Point SSID.

   The SSID will be "blueBox_" plus the serial number of the controller.



3. Establish a connection and enter the password. Contact your dealer to get the password.

4. Once the connection is established, it should now be possible for the superior system (the PC) to connect to the controller using its AP IP address. The AP IP address is displayed on the blueBox controller OLED display line 2.



It is normal for the connection to indicate “No Internet, secured” because no network traffic is forwarded outside the blueBox controller from the superior system.

17 DIAGNOSTIC PIXEL BLINK CODES

Diagnostic blinks codes can be used to troubleshoot issues when the OLED display is not available to the operator. Refer to Diagnostic page of the web interface for details.

The Diagnostic Page also includes the last 100 entries in the LogItems table. After 50 of the same errors, the controller will stop logging the errors. The controller will resume normal operations and logging once the error is rectified.

18 TECHNICAL SUPPORT

SCDDesign, LLC customizes designs based on its value-added reseller's (VAR) needs and specifications. SCDDesign, LLC does not sell the blueBox controller directly to consumers.

For technical support or questions, please contact the value-added reseller (VAR) where you purchased the blueBox controller as its commissioned functions are customized to their specifications.

Contact WEB-CAB when the blueBox controller is configured as the following:

- Production Assistant (db type 1 and db type 9)
- measureKit (db type 5)
- scanKit/feedbackKIt (db type 8)
- mdb-rfidKit (db type 7)

Contact Stiles Machinery Inc when the blueBox controller is configured as the following:

- Homag controllerMES (db type 2)
- Homag controllerMES productionFlow (db type 6, 6.1)
- Homag Automation lagerdb (db type 3 and 3.1)

For non-VAR specific questions, please contact us at cs@scddesign.com.

19 TECHNICAL SPECIFICATIONS

Controller Power Supply

Description	Value
Power Input Voltage	100-240V AC
Power Frequency	50/60 Hz
Power	60W
Power Output Voltage	12V DC
Current	5A

Controller – Raspberry Pi 4b

- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model)
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 USB 3.0 ports; 2 USB 2.0 ports.
- Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards)
- 2 × micro-HDMI ports (up to 4kp60 supported)
- 2-lane MIPI DSI display port
- 2-lane MIPI CSI camera port
- 4-pole stereo audio and composite video port
- H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)
- OpenGL ES 3.1, Vulkan 1.0
- Micro-SD card slot for loading operating system and data storage
- 5V DC via USB-C connector (minimum 3A*)
- 5V DC via GPIO header (minimum 3A*)
- Power over Ethernet (PoE) enabled (requires separate PoE HAT)
- Operating temperature: 0 – 50 degrees C ambient

*__Raspberry Pi__ is a registered trademark of the __Raspberry Pi Foundation__

This product uses a Raspberry Pi as its main controller. Raspberry Pi is a trademark of the Raspberry Pi Foundation. We are not affiliated with or endorsed by the Raspberry Pi Foundation, but our use of the Raspberry Pi word mark is in accordance with the Raspberry Pi Foundation's guidelines for limited use without a license.

We would like to note that the unmodified Raspberry Pi used in our product is compliant with relevant regulations, including those of the Federal Communications Commission (FCC), as it meets the necessary requirements without modification.

Furthermore, our product meets all applicable compliance standards and regulations, as it has been designed and tested to ensure that it complies with relevant requirements. We take compliance seriously and are committed to ensuring that our product meets the highest standards of quality and safety.

LED Strip Assembly

Color	Black Fpcb
Shape	Linear
Material	FR4
Number of Lights	6 / 140mm
Voltage	5 Volts
Special Features	Individually addressable, Programmable
Light Direction	Adjustable
Power Source	DC
Batteries Included?	No
Batteries Required?	No
Type of Bulb	LED
Max Wattage	25mA per pixel (216mA per strip) or 0.9 watts per strip

WS2812B LED Strip (5m IP65)

Color	Black Fpcb
Shape	Linear
Material	Plastic
Number of Lights	150/5meters
Voltage	5 Volts
Special Features	Individually addressable, Programmable
Shade Material	Plastic
Light Direction	Adjustable
Power Source	DC
Batteries Included?	No
Batteries Required?	No
Type of Bulb	LED
Wattage	30 watts/5meters

WS2812B and LED Strip Power Supply – per 450 LEDs or per 25 LED Strip Assembly

Output Amps	10A
Output Volts	5V DC
Output Watts	75W
Input Volts	100~240V AC
Output Port	DC Jack
Output Channel	1

Stepper Motor Power Supply

Output Amps	8A
Output Volts	12V DC
Output Watts	96W
Input Volts	100~240V AC
Output Port	DC Jack
Output Channel	1

20 PROPOSITION 65 DISCLAIMER FOR THE STATE OF CALIFORNIA

WARNING: This product may contain chemicals to which are known to the State of California to cause cancer. For more information go to www.P65Warnings.ca.gov.